# 目 录


1 概述 ..... 7
2 特性 ..... 8
2.1 NUC101 特征 ..... 8
2.2 NUC100 特征 ..... 12
2.3 NUC120 特征 ..... 15
2.4 NUC130 特征 ..... 19
2.5 NUC140 特征 ..... 23
3 编号信息列表及管脚名称定义 ..... 27
3.1 产品选型指南 ..... 27
3.1.1 NUC101 产品选型指南 ..... 27
3.1.2 NUC100产品选型指南 ..... 28
3.1.3 NUC120产品选型指南 ..... 29
3.1.4 NUC130产品选型指南 ..... 30
3.1.5 NUC140产品选型指南 ..... 31
3.2 管脚配置 ..... 32
3.2.1 NUC101 LQFP 100 pin ..... 32
3.2.2 NUC101 LQFP 64 pin ..... 33
3.2.3 NUC101 LQFP 48 pin ..... 34
3.2.4 NUC100 LQFP 100 pin ..... 35
3.2.5 NUC100 LQFP 64 pin ..... 36
3.2.6 NUC100 LQFP 48 pin ..... 37
3.2.7 NUC120 LQFP 100 pin ..... 38
3.2.8 NUC120 LQFP 64 pin ..... 39
3.2.9 NUC120 LQFP 48 pin ..... 40
3.2.10 NUC130 LQFP 100 pin ..... 41
3.2.11 NUC130 LQFP 64 pin ..... 42
3.2.12 NUC130 LQFP 48 pin ..... 43
3.2.13 NUC140 LQFP 100 pin ..... 44
3.2.14 NUC140 LQFP 64 pin ..... 45
3.2.15 NUC140 LQFP 48 pin ..... 46
3.3 管脚功能描述 ..... 47
3.3.1 NUC101管脚定义 ..... 47
3.3.2 NUC100 管脚定义 ..... 54
3.3.3 NUC120 Pin Description ..... 61
3.3.4 NUC130 管脚定义 ..... 68
3.3.5 NUC140 管脚定义 ..... 76
4 框图 ..... 83

4.1 NUC101 框图 ..... 83
4.2 NUC100 框图 ..... 84
4.3 NUC120 框图 ..... 85
4.4 NUC130框图 ..... 86
4.5 NUC140框图 ..... 87
5 功能描述 ..... 88
5.1 ARM® Cortex™-M0 内核 ..... 88
5.2 系统管理器 ..... 90
5.2.1 概述 ..... 90
5.2.2 系统复位 ..... 90
5.2.3 系统内存 ..... 91
5.2.4 系统管理器控制寄存器 ..... 93
5.2.5 系统定时器 (SysTick) ..... 122
5.2.6 嵌套向量中断控制器 (NVIC) ..... 125
5.2.7 系统控制寄存器 ..... 151
5.3 时钟控制器 ..... 157
5.3.1 时钟发生器 ..... 157
5.3.2 系统时钟 & SysTick 时钟 ..... 158
5.3.3 外围设备时钟 ..... 159
5.3.4 掉电模式(深度休眠模式) 时钟 ..... 159
5.3.5 分频器输出 ..... 160
5.3.6 时钟控制寄存器列表 ..... 161
5.3.7 时钟控制寄存器描述 ..... 162
5.4 USB 设备控制器 ..... 181
5.4.1 概述 ..... 181
5.4.2 特征 ..... 181
5.4.3 框图 ..... 182
5.4.4 功能描述 ..... 183
5.4.5 与USB外设通信处理 ..... 184
5.4.6 内存图 ..... 185
5.4.7 USB控制寄存器描述 ..... 187
5.5 通用 I/O ..... 201
5.5.1 概述与特征 ..... 201
5.5.2 GPIO控制寄存器图 ..... 203
5.5.3 GPIO控制寄存器描述 ..... 205
5.6 I2C 总线控制器 (主机/从机) ..... 216
5.6.1 简介 ..... 216
5.6.2 I2C 寄存器 ..... 220

5.6.3 寄存器说明 ..... 224
5.6.4 寄存器描述 ..... 225
5.6.5 I2C 操作模式 ..... 233
5.6.6 数据传输5种操作 ..... 234
5.7 PWM 发生器和捕捉定时器 ..... 240
5.7.1 简介 ..... 240
5.7.2 特性 ..... 241
5.7.3 PWM发生器结构 ..... 242
5.7.4 PWM定时器操作 ..... 246
5.7.5 PWM 双缓存，自动重载以及单触发模式 ..... 247
5.7.6 可调占空比 ..... 248
5.7.7 死区发生器 ..... 249
5.7.8 捕捉模式 ..... 250
5.7.9 PWM定时器中断结构 ..... 251
5.7.10 PWM定时器开启步骤 ..... 252
5.7.11 PWM定时器关闭步骤 ..... 253
5.7.12 捕捉开始步骤 ..... 254
5.7.13 寄存器列表 ..... 255
5.7.14 寄存器描述 ..... 259
5.8 实时时钟(RTC) ..... 295
5.8.1 概述 ..... 295
5.8.2 RTC 特征 ..... 295
5.8.3 RTC模块框图 ..... 296
5.8.4 RTC功能描述 ..... 297
5.8.5 寄存器列表 ..... 299
5.8.6 寄存器描述 ..... 300
5.9 串行外围设备接口(SPI)控制器 ..... 315
5.9.1 概述 ..... 315
5.9.2 特性 ..... 315
5.9.3 SPI模块方块图 (主机/从机) ..... 316
5.9.4 SPI时序波形图(主机/从机) ..... 317
5.9.5 SPI 编程例程 ..... 319
5.9.6 改变时钟频率 ..... 321
5.9.7 SPI 串行总线控制寄存器列表 ..... 322
5.9.8 SPI (MICROWIRE) 控制寄存器描述 ..... 323
5.10 定时器控制器 ..... 333
5.10.1 通用定时器控制器 ..... 333
5.10.2 特征 ..... 333
5.10.3 定时器控制器框图 ..... 334
5.10.4 定时器控制器寄存器图 ..... 335

5.11 看门狗定时器 ..... 341
5.11.1 看门狗定时器控制寄存器图 ..... 343
5.12 UART 接口控制器 ..... 346
5.12.1 概述 ..... 346
5.12.2 UART控制器特性 ..... 348
5.12.3 框图 ..... 349
5.12.4 IrDA 模式 ..... 351
5.12.5 LIN (Local Interconnection Network) 模式 ..... 353
5.12.6 UART接口控制寄存器列表 ..... 354
5.12.7 UART 接口控制寄存器描述 ..... 356
5.13 局域控制网络总线(CAN Bus) ..... 381
5.13.1 简介 ..... 381
5.13.2 特性 ..... 381
5.13.3 框图 ..... 382
5.13.4 功能描述 ..... 383
5.13.5 控制寄存器列表 ..... 390
5.14 PS2设备控制器(PS2D) ..... 417
5.14.1 概述 ..... 417
5.14.2 特性 ..... 417
5.14.3 PS/2 系统框图 ..... 418
5.14.4 PS/2 通讯 ..... 419
5.14.5 TX FIFO 运作 ..... 422
5.14.6 PS2 设备控制寄存器图表 ..... 423
5.14.7 寄存器描述 ..... 424
5.15 I2S 控制器 ..... 431
5.15.1 概述 ..... 431
5.15.2 特征 ..... 431
5.15.3 I2S 框图 ..... 432
5.15.4 I2S 操作 ..... 433
5.15.5 FIFO 操作 ..... 434
5.15.6 I2S 控制寄存器表 ..... 435
5.15.7 I2S 控制寄存器描述 ..... 436
5.16 模拟数字转换(ADC) ..... 449
5.16.1 功能概述 ..... 449
5.16.2 特征 ..... 449
5.16.3 框图 ..... 450
5.16.4 操作步骤 ..... 452
5.16.5 ADC寄存器列表 ..... 459
5.16.6 ADC寄存器描述 ..... 460
5.17 模拟比较器 ..... 471

5.17.1 功能描述 ..... 471
5.17.2 特征 ..... 471
5.17.3 框图 ..... 471
5.17.4 操作步骤 ..... 472
5.17.5 寄存器列表 ..... 473
5.17.6 寄存器描述 ..... 474
5.18 PDMA 控制器 ..... 478
5.18.1 概述 ..... 478
5.18.2 特性 ..... 478
5.18.3 框图 ..... 479
5.18.4 功能描述 ..... 480
5.18.5 PDMA 控制器寄存器列表 ..... 481
5.18.6 PDMA 控制寄存器描述 ..... 482
6 FLASH 内存控制器 (FMC) ..... 505
6.1 概述 ..... 505
6.2 特征 ..... 505
6.3 Flash 内存控制器框图 ..... 506
6.4 Flash 内存结构 ..... 507
6.5 启动选择 ..... 509
6.6 数据 Flash ..... 509
6.7 用户配置 ..... 510
6.8 在系统编程(ISP) ..... 513
6.8.1 ISP 程序 ..... 513
6.9 Flash控制寄存器图 ..... 516
6.10 Flash 控制寄存器描述 ..... 517
7 电气特性 ..... 527
7.1 绝对最大额定值 ..... 527
7.2 DC 电气特性 ..... 528
7.3 AC电气特性 ..... 530
7.3.1 外部晶振 XTAL1 ..... 530
7.3.2 外部32KHz晶振 XTAL ..... 532
7.3.3 内部22.1184MHz RC晶振 ..... 532
7.3.4 内部10KHz RC 晶振 ..... 532
7.4 模拟量特性 ..... 533
7.4.1 1-MS/s 12-位 SARADC特性 ..... 533
7.4.2 LDO规格 & Power 管理 ..... 534
7.4.3 低压复位说明 ..... 535

7.4.4 欠压检测说明 ..... 535
7.4.5 上电复位说明(5V) ..... 535
7.4.6 温度传感器说明 ..... 536
7.4.7 比较器说明 ..... 536
7.4.8 USB PHY说明 ..... 537
8 封装定义 ..... 539
8.1.1 100L LQFP (14x14x1.4 mm footprint 2.0mm) ..... 539
8.1.2 64L LQFP (10x10x1.4mm footprint 2.0 mm ) ..... 540
8.1.3 48L LQFP (7x7x1.4mm footprint 2.0mm) ..... 541
9 版本信息 ..... 542

# 1 概述

The NUC1xx系列是32位的内嵌ARM® Cortex™-M0核的微控制器 ,适用于工业控制和需要丰富的通信接口的应用领域，Cortex™-M0是ARM最新的 32位嵌入式处理器，拥有与传统8051单片机之匹敌的价格优势. NUC1xx 系列包括 NUC100, NUC101, NUC120, NUC130 与NUC140.

NUC100 Cortex™-M0内核系列最高可运行至50MHz外部时钟，内建32K/64K/128K字节内建Flash存储器，4K/8K/16K字节内建SRAM。并内建有定时器，看门狗定时器，RTC，PDMA，UART，SPI/SSP，I2C, , $I^2S$ ,PWM定时器, GPIO, 12位ADC，模拟比较器，低电压复位和欠压检测功能。

NUC101 Cortex™-M0内核系列最高可运行至50MHz外部时钟，内建32K/64K/128K字节内建Flash存储器，4K/8K/16K字节内建SRAM。并内建有定时器，看门狗定时器，RTC，UART，SPI/SSP，I2C, PWM定时器, GPIO, LIN，USB2.0 FS设备，12位ADC，模拟比较器，低电压复位和欠压检测功能。

NUC120 Cortex™-M0内核系列最高可运行至50MHz外部时钟，内建2K/64K/128K字节内建Flash存储器，4K/8K/16K字节内建SRAM。并内建有定时器，看门狗定时器，RTC，PDMA，UART，SPI/SSP，I2C,I2S， PWM 定时器, GPIO, USB 2.0 FS设备，12位ADC，模拟比较器，低电压复位和欠压检测功能。

NUC130 Cortex™-M0内核系列最高可运行至50MHz外部时钟，内建32K/64K/128K字节内建Flash存储器，4K/8K/16K字节内建SRAM。并内建有定时器，看门狗定时器，RTC，PDMA，UART，SPI/SSP，I2C, $I^2S$,PWM定时器, GPIO, LIN, CAN,12位ADC，模拟比较器，低电压复位和欠压检测功能。

NUC140 Cortex™-M0内核系列最高可运行至50MHz外部时钟，内建32K/64K/128K字节内建Flash存储器，4K/8K/16K字节内建SRAM。并内建有定时器，看门狗定时器，RTC，PDMA，UART，SPI/SSP，I2C, $I^2S$, PWM Timer, GPIO, LIN, CAN, USB 2.0模块，12位ADC，模拟比较器，低电压复位和欠压检测功能。

# 2 特性

该器件的功能依赖于他们的子系列产品.

## 2.1 NUC101 特征

- 内核
  - ARM® Cortex™-M0 内核最高运行 50MHz.
  - 一个 24-位系统定时器.
  - 低功耗睡眠模式
  - 单指令 32 位硬件乘法器
  - 嵌套向量中断控制器 NVIC 用于控制 32 个中断源，每个中断源可设置为 4 个优先级
  - 支持串行线调试（SWD）及 2 个观察点/4 个断点
- 宽电压工作范围 由2.5V 至 5.5V
- FLASH ROM 存储器
  - 32K/64K/128K 字节 FLASH EPROM 用于存储程序代码
  - 4kB FLASH 用于存储 ISP 引导代码
  - 支持 ISP/IAP 编程升级
  - 512 字节一页 FLASH 擦除模式
  - 在 128K 字节系统中可配置数据 FLASH 地址区域，在 32K/64K 字节系统中包含 4K 字节数据 FLASH 区域
  - 在仿真界面下，支持 2 线 ICP 升级方式
  - 支持外部编程器并行高速编程模式
- SRAM 内存
  - 4K/8K/16K 字节内建 SRAM.
- 时钟控制
  - 针对不同应用可灵活选择时钟
  - 内建 22MHz 振荡器(精度可调整在 1%之内) 可用于系统运行，以及低功耗 10KHz RC 振荡器用于看门狗及睡眠模式唤醒等功能
  - 支持一组 PLL, 高至 50MHz, 用于更速的系统运行.
  - 外部 12MHz 晶振输入用于 USB 及精准的时序运作
  - 外部 32 kHz 晶振输入用于 RTC 及低功耗模式操作.
- GPIO
  - 四种 I/O 模式:
    - 准双向模式
    - 推挽输出模式
    - 开漏输出模式
    - 高阻输入模式
  - TTL/Schmitt 触发输入可选.
  - I/O 管脚可被定义为边沿/电平触发模式的中断源.
  - 支持大电流驱动/灌入 I/O.
- 定时器
  - 4 组带 8 位预分频的 24 位定时器.
  - 定时计数自动重载.

- 看门狗定时器
  - 由配置位定义默认打开/关闭模式。
  - 多选的时钟源
  - 溢出时间 8 种选择:输出由 ms 到 s (根据选择时钟源不同而不同)
  - WDT 可用作掉电模式/睡眠模式的唤醒。
  - 定时溢出后触发中断/复位功能选择
- RTC
  - 通过频率补偿寄存器(FCR) 支持软件频率补偿功能
  - 支持 RTC 计数(秒，分，小时) 及万年历功能(日，月，年)
  - 支持闹铃寄存器 (秒，分，小时，日，月，年)
  - 可分为 12 小时制或 24 小时制
  - 闰年自动识别
  - 支持秒级中断
  - 支持唤醒功能
- PWM/Capture
  - 内建四个 16 位 PWM 产生器,可输出 8 路 PWM 或 4 组互补 PWM
  - 每个 PWM 产生器配有一个 8 位时钟预分频, 8 位分频器和一个时钟选择器.另外有一个用于互补 PWM 的死区发生器
  - PWM 中断与 PWM 周期同步
  - 八路 16 位捕捉定时器(利用 PWM 定时器)提供 8 路输入的上升/下降沿的捕捉功能
  - 支持捕捉中断
- UART
  - 最多两组 16550 UART 装置.
  - UART 支持流程控制 (TX, RX, CTS and RTS)
  - UART0 带 64-字节 FIFO 用于高速模式
  - UART1 带 16-字节 FIFO 用于标准模式
  - 支持 IrDA(SIR)协议 及 LIN 总线功能
  - 可编程波特率发生器频率高至 1/16 系统时钟
  - 支持 PDMA 模式
- SPI
  - 最高支持 4 组 SPI 器件.
  - 当系统在 50MHz 运行时，主机速率高至 25 Mbps /从机高至 12.5Mbps
  - 支持 MICROWIRE 协议/SPI 总线协议(SSP)
  - 全双工同步串行数据传输
  - 1 至 32 位可改变数据长度传输模式
  - 可设置 MSB 或 LSB 在前的传输模式
  - 可设置 Rx 及 Tx 数据在上升沿或下降沿有效.
  - 当作为主机时 2 条从机片选线，作为从机时 1 条从机片选线
  - 32 位字节传输模式下的字节睡眠模式

- I2C
  - 支持 2 组 I2C 设备
  - 主/从机最高传输速率 1Mbit/s ( 高速模式 FM+ )
  - 主从机之间双向数据传输
  - 多主机总线支持 (无中心主机).
  - 多主机间同时传输数据仲裁，避免总线上串行数据损坏
  - 总线采用同步时钟,可实现设备之间以不同的速率传输
  - 可用同步时钟控制总线上数据暂停及恢复传送
  - 可编程的时钟适用于通用速率控制
  - I2C 总线上支持多地址辨识 (2 组从机地址被屏蔽时)
- $I^2S$
  - 外部音频 CODEC 接口
  - 可作主机也可作从机模式
  - 能处理 8, 16, 和 32 位字
  - 支持单声道和立体声的音频数据
  - 支持 $I^2S$ 和 最高有效位数据格式
  - 提供两组 8 字的 FIFO 数据缓存，一组用于发送，一组用于接收
  - 缓冲区超过可编程边界时，产生中断请求
  - 支持两组 DMA 请求，一组用于发送，另一组用于接收
- USB 2.0 全速设备
  - 1 组支持 USB 2.0 全速模块 12Mbps
  - 支持片内 USB 数据传输.
  - 四个中断事件共用一个中断源.
  - 支持控制传输（Control），批量传输（Bulk），中断传输(Interrupt)及同步传输(Isochronous)
  - 总线上无信号超过 3ms，支持自动暂停功能
  - –提供 6 个可编程端点(endpoints)
  - 512 字节内部 SRAM 作为 USB 的缓存区
  - 支持远程唤醒功能.

- ADC
  - 12 位 8 通道，转换速率达 800ksps.
  - 8 通道单端模式或 4 通道差分模式
  - 单一模式/单周期扫描模式/连续扫描模式
  - 每通道转换结果存放于独立寄存器内
  - 扫描使能通道
  - 门槛电压侦测
  - 有软件或外部管脚触发开始转换
- 模拟比较器
  - 2 组模拟比较器模块
  - 基准电位可选择外部输入或内部参考电压
  - 转换结束可作为中断触发条件
  - 支持掉电模式唤醒功能
- 内建温度传感器， 1℃ 分辨率.
- 欠压检测
  - 支持四级检测电压: 4.5V/3.8V/2.7V/2.2V
  - 支持欠压中断和复位选择
- 内建一组 LDO
- 低电压复位
- 工作温度: -40℃~85℃
- 封装:
  - 无铅封装 (RoHS)
    - LQFP 48-pin / 64-pin / 100-pin

## 2.2 NUC100 特征

- 内核
  - ARM® Cortex™-M0 内核最高运行 50MHz.
  - 一个 24-位系统定时器.
  - 低功耗睡眠模式
  - 单指令 32 位硬件乘法器
  - 嵌套向量中断控制器 NVIC 用于控制 32 个中断源，每个中断源可设置为 4 个优先级
  - 支持串行线调试（SWD）及 2 个观察点/4 个断点
- 宽电压工作范围 由2.5V 至 5.5V
- FLASH ROM 存储器
  - 32K/64K/128K 字节 FLASH ROM 用于存储程序代码
  - 4kB FLASH 用于存储 ISP 引导代码
  - 支持 ISP/IAP 编程升级
  - 512 字节一页 FLASH 擦除模式
  - 在 128K 字节系统中可配置数据 FLASH 地址区域，在 32K/64K 字节系统中包含 4K 字节数据 FLASH 区域
  - 在仿真界面下，支持 2 线 ICP 升级方式
  - 支持外部编程器并行高速编程模式.
- SRAM存储器
  - 4K/8K/16K 字节内建 SRAM.
  - 支持 PDMA 模式
- PDMA (Peripheral DMA)
  - 支持 9 通道 PDMA 用于 SRAM 和周边设备的自动数据传输.
- 时钟控制
  - 针对不同应用可灵活选择时钟
  - 内建 22MHz 振荡器(精度可调整在 1%之内) 可用于系统运行，以及低功耗 10KHz RC 振荡器用于看门狗及睡眠模式唤醒等功能
  - 支持一组 PLL, 高至 50MHz, 用于更速的系统运行.
  - 外部 12MHz 晶振输入用于 USB 及精准的时序运作
  - 外部 32 kHz 晶振输入用于 RTC 及低功耗模式操作.
- GPIO
  - 四种 I/O 模式:
    - 准双向模式
    - 推挽输出模式
    - 开漏输出模式
    - 高阻输入模式
  - TTL/Schmitt 触发输入可选.
  - I/O 管脚可被定义为边沿/电平触发模式的中断源.
  - 支持大电流驱动/灌入 I/O.

- 定时器
  - 4 组带 8 位预分频的 24 位定时器.
  - 定时计数自动重载.
- 看门狗定时器
  - 由配置位定义默认打开/关闭模式。
  - 多选的时钟源
  - 溢出时间 8 种选择:输出由 6ms ~ 3.0sec (根据选择时钟源不同而不同)
  - WDT 可用作掉电模式/睡眠模式的唤醒。
  - 定时溢出后触发中断/复位功能选择.
- RTC
  - 通过频率补偿寄存器(FCR) 支持软件频率补偿功能
  - 支持 RTC 计数(秒，分，小时) 及万年历功能(日，月，年)
  - 支持闹铃寄存器 (秒，分，小时，日，月，年)
  - 可分为 12 小时制或 24 小时制
  - 闰年自动识别
  - 支持秒级中断
  - 支持唤醒功能.
- PWM/捕捉/比较 定时器
  - 内建四个 16 位 PWM 产生器,可输出 8 路 PWM 或 4 组互补 PWM
  - 每个 PWM 产生器配有一个 8 位时钟预分频, 8 位分频器和一个时钟选择器.另外有一个用于互补 PWM 的死区发生器
  - PWM 中断与 PWM 周期同步
  - 八路 16 位捕捉定时器(利用 PWM 定时器)提供 8 路输入的上升/下降沿的捕捉功能
  - 支持捕捉中断.
- UART
  - 最多三组 16550 UART 装置.
  - UART 支持流程控制 (TX, RX, CTS and RTS)
  - UART0 带 64-字节 FIFO 用于高速模式
  - UART1/2 (可选) 带 16-字节 FIFO 用于标准模式
  - 支持 IrDA(SIR)协议
  - 可编程波特率发生器频率高至 1/16 系统时钟
  - 支持 PDMA 模式
- SPI
  - 最高支持 4 组 SPI 器件.
  - 主机速率高至 16 Mbps /从机高至 10Mbps.
  - 支持 MICROWIRE 协议/SPI 总线协议(SSP)
  - 全双工同步串行数据传输
  - 1 至 32 位可变数据长度传输模式
  - 可设置 MSB 或 LSB 在前的传输模式
  - 可设置 Rx 及 Tx 数据在上升沿或下降沿有效.
  - 当作为主机时 2 条从机片选线，作为从机时 1 条从机片选线
  - 32 位字节传输模式下的字节睡眠模式
  - 支持 PDMA 模式

- $I^2C$
  - 支持 2 组 I2C 设备
  - 主/从机最高传输速率 1Mbit/s ( 高速模式 FM+ )
  - 主从机之间双向数据传输
  - 多主机总线支持 (无中心主机).
  - 多主机间同时传输数据仲裁，避免总线上串行数据损坏
  - 总线采用同步时钟,可实现设备之间以不同的速率传输
  - 可用同步时钟控制总线上数据暂停及恢复传送
  - 可编程的时钟适用于不同速率控制
  - I2C 总线上支持多地址辨识 (2 组从机地址被屏蔽时)
- $I^2S$
  - 外部音频 CODEC 接口
  - 可作主机也可作从机模式
  - 能处理 8, 16, 和 32 位字
  - 支持单声道和立体声的音频数据
  - 支持 $I^2S$ 和 最高有效位数据格式
  - 提供两组 8 字的 FIFO 数据缓存，一组用于发送，一组用于接收
  - 缓冲区超过可编程边界时，产生中断请求
  - 支持两组 DMA 请求，一组用于发送，另一组用于接收
- ADC
  - 12 位 8 通道，转换速率达 800ksps.
  - 8 通道单端模式或 4 通道差分模式
  - 单一模式/单周期扫描模式/连续扫描模式
  - 每通道转换结果存放于独立寄存器内
  - 独立打开转换通道
  - 门槛电压侦测
  - 有软件或外部管脚触发开始转换
  - 支持 PDMA 模式
- 模拟比较器
  - 2 组模拟比较器模块
  - 基准电位可选择外部输入或内部频带间隙电压
  - 比较结果改变可作为中断触发条件
  - 支持掉电模式唤醒功能
- 内建温度传感器， 1℃ 分辨率.
- 欠压检测
  - 支持四级检测电压: 4.5V/3.8V/2.7V/2.2V
  - 支持欠压中断和复位选择
- 内建一组 LDO
- 低电压复位
- 工作温度: -40℃~85℃
- 封装: 无铅封装(RoHS)
  - LQFP 100-pin / 64-pin / 48-pin

## 2.3 NUC120 特征

- 内核
  - ARM® Cortex™-M0 内核最高运行 50MHz.
  - 一个 24-位系统定时器.
  - 低功耗睡眠模式
  - 单指令 32 位硬件乘法器
  - 嵌套向量中断控制器 NVIC 用于控制 32 个中断源，每个中断源可设置为 4 个优先级
  - 支持串行线调试（SWD）及 2 个观察点/4 个断点.
- 宽电压工作范围 由2.5V 至 5.5V
- FLASH ROM 存储器
  - 32K/64K/128K 字节 FLASH ROM 用于存储程序代码
  - 4kB FLASH 用于存储 ISP 引导代码
  - 支持 ISP/IAP 编程升级
  - 512 字节一页 FLASH 擦除模式
  - 在 128K 字节系统中可配置数据 FLASH 地址区域，在 32K/64K 字节系统中包含 4K 字节数据 FLASH 区域
  - 在仿真界面下，支持 2 线 ICP 升级方式
  - 支持外部编程器并行高速编程模式.
- SRAM存储器
  - 4K/8K/16K 字节内建 SRAM.
  - 支持 PDMA 模式
- PDMA (Peripheral DMA)
  - 支持 9 通道 PDMA 用于 SRAM 和周边设备的自动数据传输.
- 时钟控制
  - 针对不同应用可灵活选择时钟
  - 内建 22MHz 振荡器(精度可调整在 1%之内) 可用于系统运行，以及低功耗 10KHz 振荡器用于看门狗及睡眠模式唤醒等功能
  - 支持一组 PLL, 高至 50MHz, 用于更速的系统运行.
  - 外部 12MHz 晶振输入用于 USB 及精准的时序运作
  - 外部 32 kHz 晶振输入用于 RTC 及低功耗模式操作.
- GPIO
  - 四种 I/O 模式:
    - 准双向模式
    - 推挽输出模式
    - 开漏输出模式
    - 高阻输入模式
  - TTL/Schmitt 触发输入可选.
  - I/O 管脚可被定义为边沿/电平触发模式的中断源.
  - 支持大电流驱动/灌入 I/O.

- 定时器
  - 4 组带 8 位预分频的 24 位定时器.
  - 定时计数自动重载.
- 看门狗定时器
  - 由配置位定义默认打开/关闭模式。
  - 多选的时钟源
  - 溢出时间 8 种选择:输出由 6ms ~ 3.0sec (根据选择时钟源不同而不同)
  - WDT 可用作掉电模式/睡眠模式的唤醒。
  - 定时溢出后触发中断/复位功能选择.
- RTC
  - 通过频率补偿寄存器(FCR) 支持软件频率补偿功能
  - 支持 RTC 计数(秒，分，小时) 及万年历功能(日，月，年)
  - 支持闹铃寄存器 (秒，分，小时，日，月，年)
  - 可分为 12 小时制或 24 小时制
  - 闰年自动识别
  - 支持秒级中断
  - 支持唤醒功能.
- PWM/Capture
  - 内建四个 16 位 PWM 产生器,可输出 8 路 PWM 或 4 组互补 PWM
  - 每个 PWM 产生器配有一个 8 位时钟预分频, 8 位分频器和一个时钟选择器.另外有一个用于互补 PWM 的死区发生器
  - PWM 中断与 PWM 周期同步
  - 八路 16 位捕捉定时器(利用 PWM 定时器)提供 8 路输入的上升/下降沿的捕捉功能
  - 支持捕捉中断
- UART
  - 最多三组 16550 UART 装置.
  - UART 支持流程控制 (TX, RX, CTS and RTS)
  - UART0 带 64-字节 FIFO 用于高速模式
  - UART1/2 (可选) 带 16-字节 FIFO 用于标准模式
  - 支持 IrDA(SIR)协议
  - 可编程波特率发生器频率高至 1/16 系统时钟
  - 支持 PDMA 模式
- SPI
  - 最高支持 4 组 SPI 器件.
  - 主机速率高至 16 Mbps /从机高至 10Mbps
  - 支持 MICROWIRE 协议/SPI 总线协议(SSP)
  - 全双工同步串行数据传输
  - 1 至 32 位多种数据长度传输模式
  - 可设置 MSB 或 LSB 在前的传输模式
  - 可设置 Rx 及 Tx 数据在上升沿或下降沿有效.
  - 当作为主机时 2 条从机片选线，作为从机时 1 条从机片选线
  - 32 位字节传输模式下的字节睡眠模式
  - 支持 PDMA 模式

- I²C
  - 支持 2 组 I2C 设备
  - 主/从机最高传输速率 1Mbit/s ( 高速模式 FM+ )
  - 主从机之间双向数据传输
  - 多主机总线支持 (无中心主机).
  - 多主机间同时传输数据仲裁，避免总线上串行数据损坏
  - 总线采用同步时钟,可实现设备之间以不同的速率传输
  - 可用同步时钟控制总线上数据暂停及恢复传送
  - 可编程的时钟适用于不同速率控制
  - I2C 总线上支持多地址辨识 (2 组从机地址被屏蔽时)
- I²S
  - 外部音频 CODEC 接口
  - 可作主机也可作从机模式
  - 能处理 8, 16, 和 32 位字
  - 支持单声道和立体声的音频数据
  - 支持 I²S 和 最高有效位数据格式
  - 提供两组 8 字的 FIFO 数据缓存，一组用于发送，一组用于接收
  - 缓冲区超过可编程边界时，产生中断请求
  - 支持两组 DMA 请求，一组用于发送，另一组用于接收
- USB 2.0 全速设备模块
  - 1 组支持 USB 2.0 全速模块 12Mbps
  - 支持片内 USB 数据传输.
  - 作为 1 组中断源，提供四级等级.
  - 支持控制传输（Control），批量传输（Bulk），中断传输(Interrupt)及同步传输(Isochronous)
  - 当总线上无信号时，支持自动暂停 3ms 功能
  - 支持 6 组可编辑结束点(endpoints)
  - 512 字节内部 SRAM 作为 USB 的缓存区
  - 支持远程唤醒功能.
  - 支持 PDMA 模式
- ADC
  - 12 位 8 通道，转换速率达 800ksps.
  - 8 通道单端模式或 4 通道差分模式
  - 单一模式/单周期扫描模式/连续扫描模式
  - 每通道转换结果存放于独立寄存器内
  - 独立打开转换通道
  - 门槛电压侦测
  - 有软件或外部管脚触发开始转换
  - 支持 PDMA 模式
- 模拟比较器
  - 2 组模拟比较器模块
  - 基准电位可选择外部输入或内部参考电压
  - 转换结束可作为中断触发条件
  - 支持掉电模式唤醒功能

- 内建温度传感器， 1℃ 分辨率.
- 欠压检测
  - 支持四级检测电压: 4.5V/3.8V/2.7V/2.2V
  - 支持欠压中断和复位选择
- 内建一组 LDO
- 低电压复位
- 工作温度: -40℃~85℃
- 封装:
  - 无铅封装(RoHS)
    - LQFP 100-pin / 64-pin / 48-pin

## 2.4 NUC130 特征

- 内核
  - ARM® Cortex™-M0 内核最高运行 50MHz.
  - 一个 24-位系统定时器.
  - 低功耗睡眠模式
  - 单指令 32 位硬件乘法器
  - 嵌套向量中断控制器 NVIC 用于控制 32 个中断源，每个中断源可设置为 4 个优先级
  - 支持串行线调试（SWD）及 2 个观察点/4 个断点.
- 宽电压工作范围 由2.5V 至 5.5V
- FLASH ROM 存储器
  - 32K/64K/128K 字节 FLASH ROM 用于存储程序代码
  - 4kB FLASH 用于存储 ISP 引导代码
  - 支持 ISP/IAP 编程升级
  - 512 字节一页 FLASH 擦除模式
  - 在 128K 字节系统中可配置数据 FLASH 地址区域，在 32K/64K 字节系统中包含 4K 字节数据 FLASH 区域
  - 在仿真界面下，支持 2 线 ICP 升级方式
  - 支持外部编程器并行高速编程模式.
- SRAM存储器
  - 4K/8K/16K 字节内建 SRAM.
  - 支持 PDMA 模式
- PDMA (Peripheral DMA)
  - 支持 9 通道 PDMA 用于 SRAM 和周边设备的自动数据传输.
- 时钟控制
  - 针对不同应用可灵活选择时钟
  - 内建 22MHz 振荡器(精度可调整在 1%之内) 可用于系统运行，以及低功耗 10KHz 振荡器用于看门狗及睡眠模式唤醒等功能
  - 支持一组 PLL, 高至 50MHz, 用于更速的系统运行.
  - 外部 12MHz 晶振输入用于 USB 及精准的时序运作
  - 外部 32 kHz 晶振输入用于 RTC 及低功耗模式操作.
- GPIO
  - 四种 I/O 模式:
    - 准双向模式
    - 推挽输出模式
    - 开漏输出模式
    - 高阻输入模式
  - TTL/Schmitt 触发输入可选.
  - I/O 管脚可被定义为边沿/电平触发模式的中断源.
  - 支持大电流驱动/灌入 IO.

- 定时器
  - 4 组带 8 位预分频的 24 位定时器.
  - 定时计数自动重载.
- 看门狗定时器
  - 由配置位配置默认打开/关闭模式。
  - 多选的时钟源
  - 溢出时间 8 种选择:输出由 6ms ~ 3.0sec (根据选择时钟源不同而不同)
  - WDT 可用作掉电模式/睡眠模式的唤醒。
  - 定时溢出后触发中断/复位功能选择.
- RTC
  - 通过频率补偿寄存器(FCR) 支持软件频率补偿功能
  - 支持 RTC 计数(秒，分，小时) 及万年历功能(日，月，年)
  - 支持闹铃寄存器 (秒，分，小时，日，月，年)
  - 可分为 12 小时制或 24 小时制
  - 闰年自动识别
  - 支持秒级中断
  - 支持唤醒功能.
- PWM/Capture
  - 内建四个 16 位 PWM 产生器,可输出 8 路 PWM 或 4 组互补 PWM
  - 每个 PWM 产生器配有一个 8 位时钟预分频, 8 位分频器和一个时钟选择器.另外有一个用于互补 PWM 的死区发生器
  - PWM 中断与 PWM 周期同步
  - 八路 16 位捕捉定时器(利用 PWM 定时器)提供 8 路输入的上升/下降沿的捕捉功能
  - 支持捕捉中断
- UART
  - 三组 16550 UART 装置.
  - UART 支持流程控制 (TX, RX, CTS and RTS)
  - UART0 带 64-字节 FIFO 用于高速模式
  - UART1/2 (可选) 带 16-字节 FIFO 用于标准模式
  - 支持 IrDA(SIR)协议 及 LIN 总线功能
  - 可编程波特率发生器频率高至 1/16 系统时钟
  - 支持 PDMA 模式
- SPI
  - 最高支持 4 组 SPI 器件.
  - 主机速率高至 16 Mbps /从机高至 10Mbps
  - 支持 MICROWIRE 协议/SPI 总线协议(SSP)
  - 全双工同步串行数据传输
  - 1 至 32 位可变数据长度传输模式
  - 可设置 MSB 或 LSB 在前的传输模式
  - 可设置 Rx 及 Tx 数据在上升沿或下降沿有效.
  - 当作为主机时 2 条从机片选线，作为从机时 1 条从机片选线
  - 32 位字节传输模式下的字节睡眠模式
  - 支持 PDMA 模式

- $I^2C$
  - 支持 2 组 I2C 设备
  - 主/从机最高传输速率 1Mbit/s ( 高速模式 FM+ )
  - 主从机之间双向数据传输
  - 多主机总线支持 (无中心主机).
  - 多主机间同时传输数据仲裁，避免总线上串行数据损坏
  - 总线采用同步时钟,可实现设备之间以不同的速率传输
  - 可用同步时钟控制总线上数据暂停及恢复传送
  - 可编程的时钟适用于不同速率控制
  - I2C 总线上支持多地址辨识 (2 组从机地址被屏蔽时)
- $I^2S$
  - 外部音频 CODEC 接口
  - 可作主机也可作从机模式
  - 能处理 8, 16, 和 32 位字
  - 支持单声道和立体声的音频数据
  - 支持 $I^2S$ 和 最高有效位数据格式
  - 提供两组 8 字的 FIFO 数据缓存，一组用于发送，一组用于接收
  - 缓冲区超过可编程边界时，产生中断请求
  - 支持两组 DMA 请求，一组用于发送，另一组用于接收
- CAN 2.0
  - 2 组支持 CAN 2.0B 协议设备
  - 支持标准信息帧(11 位 ID)和扩展信息帧(29 位 ID)
  - 位传输速率最高至 1Mbits/s
  - 不归零码 NRZ 位编码/解码
  - 错误侦测及状态报告
    - 支持位错误，格式错误，位填充错误，15 位 CRC 侦测及应答错误中断
    - 每条 CAN 总线错误及传输/接受完成报告
  - 位时序同步
  - 支持扩展滤波方式
  - 支持睡眠模式唤醒功能
- ADC
  - 12 位 8 通道，转换速率达 800ksps.
  - 8 通道单端模式或 4 通道差分模式
  - 单一模式/单周期扫描模式/连续扫描模式
  - 每通道转换结果存放于独立寄存器内
  - 独立打开转换通道
  - 门槛电压侦测
  - 有软件或外部管脚触发开始转换
  - 支持 PDMA 模式
- 模拟比较器
  - 2 组模拟比较器模块
  - 基准电位可选择外部输入或内部参考电压
  - 转换结束可作为中断触发条件
  - 支持掉电模式唤醒功能

- 内建温度传感器， 1℃ 分辨率.
- 欠压检测
  - 支持四级检测电压: 4.5V/3.8V/2.7V/2.2V
  - 支持欠压中断和复位选择
- 内建一组 LDO
- 低电压复位
- 工作温度: -40℃~85℃
- 封装:
  - 无铅封装(RoHS)
    - LQFP 100-pin / 64-pin / 48-pin

## 2.5 NUC140 特征

- 内核
  - ARM® Cortex™-M0 内核最高运行 50MHz.
  - 一个 24-位系统定时器.
  - 低功耗睡眠模式
  - 单指令 32 位硬件乘法器
  - 嵌套向量中断控制器 NVIC 用于控制 32 个中断源，每个中断源可设置为 4 个优先级
  - 支持串行线调试（SWD）及 2 个观察点/4 个断点.
- 宽电压工作范围 由2.5V 至 5.5V
- FLASH ROM 存储器
  - 32K/64K/128K 字节 FLASH ROM 用于存储程序代码
  - 4kB FLASH 用于存储 ISP 引导代码
  - 支持 ISP/IAP 编程升级
  - 512 字节一页 FLASH 擦除模式
  - 在 128K 字节系统中可配置数据 FLASH 地址区域，在 32K/64K 字节系统中包含 4K 字节数据 FLASH 区域
  - 在仿真界面下，支持 2 线 ICP 升级方式
  - 支持外部编程器并行高速编程模式.
- SRAM 存储器
  - 8K/16K 字节内建 SRAM.
  - 支持 PDMA 模式
- PDMA (外围 DMA)
  - 支持 9 通道 PDMA 用于 SRAM 和周边设备的自动数据传输.
- 时钟控制
  - 针对不同应用可灵活选择时钟
  - 内建 22MHz 振荡器(精度可调整在 1%之内) 可用于系统运行，以及低功耗 10KHz 振荡器用于看门狗及睡眠模式唤醒等功能
  - 支持一组 PLL, 高至 50MHz, 用于更速的系统运行.
  - 外部 12MHz 晶振输入用于 USB 及精准的时序运作
  - 外部 32 kHz 晶振输入用于 RTC 及低功耗模式操作.
- GPIO
  - 四种 I/O 模式:
    - 准双向模式
    - 推挽输出模式
    - 开漏输出模式
    - 高阻输入模式
  - TTL/Schmitt 触发输入可选.
  - I/O 管脚可被定义为边沿/电平触发模式的中断源.
  - - 支持大电流驱动/灌入 I/O.

- 定时器
  - 4 组带 8 位预分频的 24 位定时器.
  - 定时计数自动重载.
- 看门狗定时器
  - 由配置位定义默认打开/关闭模式。
  - 多选的时钟源
  - 溢出时间 8 种选择:输出由 6ms ~ 3.0sec (根据选择时钟源不同而不同)
  - WDT 可用作掉电模式/睡眠模式的唤醒。
  - 定时溢出后触发中断/复位功能选择.
- RTC
  - 通过频率补偿寄存器(FCR) 支持软件频率补偿功能
  - 支持 RTC 计数(秒，分，小时) 及万年历功能(日，月，年)
  - 支持闹铃寄存器 (秒，分，小时，日，月，年)
  - 可分为 12 小时制或 24 小时制
  - 闰年自动识别
  - 支持秒级中断
  - 支持唤醒功能.
- PWM/Capture
  - 内建四个 16 位 PWM 产生器,可输出 8 路 PWM 或 4 组互补 PWM
  - 每个 PWM 产生器配有一个 8 位时钟预分频, 8 位分频器和一个时钟选择器.另外有一个用于互补 PWM 的死区发生器
  - PWM 中断与 PWM 周期同步
  - 八路 16 位捕捉定时器(利用 PWM 定时器)提供 8 路输入的上升/下降沿的捕捉功能
  - 支持捕捉中断
- UART
  - 最多三组 16550 UART 装置.
  - UART 支持流程控制 (TX, RX, CTS and RTS)
  - UART0 带 64-字节 FIFO 用于高速模式
  - UART1/2 (可选) 带 16-字节 FIFO 用于标准模式
  - 支持 IrDA(SIR)协议 及 LIN 总线功能
  - 可编程波特率发生器频率高至 1/16 系统时钟
  - 支持 PDMA 模式
- SPI
  - 最高支持 4 组 SPI 器件.
  - 主机速率高至 16 Mbps /从机高至 10Mbps
  - 支持 MICROWIRE 协议/SPI 总线协议(SSP)
  - 全双工同步串行数据传输
  - 1 至 32 位多种数据长度传输模式
  - 可设置 MSB 或 LSB 在前的传输模式
  - 可设置 Rx 及 Tx 数据在上升沿或下降沿有效.
  - 当作为主机时 2 条从机片选线，作为从机时 1 条从机片选线
  - 32 位字节传输模式下的字节睡眠模式
  - 支持 PDMA 模式

- $I^2C$
  - 支持 2 组 I2C 设备
  - 主/从机最高传输速率 1Mbit/s ( 高速模式 FM+ )
  - 主从机之间双向数据传输
  - 多主机总线支持 (无中心主机).
  - 多主机间同时传输数据仲裁，避免总线上串行数据损坏
  - 总线采用同步时钟,可实现设备之间以不同的速率传输
  - 可用同步时钟控制总线上数据暂停及恢复传送
  - 可编程的时钟适用于不同速率控制
  - I2C 总线上支持多地址辨识 (2 组从机地址被屏蔽时)
- $I^2S$
  - 外部音频 CODEC 接口
  - 可作主机也可作从机模式
  - 能处理 8, 16, 和 32 位字
  - 支持单声道和立体声的音频数据
  - 支持 $I^2S$ 和 最高有效位数据格式
  - 提供两组 8 字的 FIFO 数据缓存，一组用于发送，一组用于接收
  - 缓冲区超过可编程边界时，产生中断请求
  - 支持两组 DMA 请求，一组用于发送，另一组用于接收
- CAN 2.0
  - 2 组支持 CAN 2.0B 协议设备
  - 支持标准信息帧(11 位 ID)和扩展信息帧(29 位 ID)
  - 位传输速率最高至 1Mbits/s
  - 不归零码 NRZ 位编码/解码
  - 错误侦测及状态报告
    - 支持位错误，格式错误，位填充错误，15 位 CRC 侦测及应答错误中断
    - 每条 CAN 总线错误及传输/接受完成报告
  - 位时序同步
  - 支持扩展滤波方式
  - 支持睡眠模式唤醒功能
- USB 2.0 全速设备模块
  - 1 组支持 USB 2.0 全速模块 12Mbps
  - 支持片内 USB 数据传输.
  - 四个中断事件共用一个中断源.
  - 支持控制传输（Control），批量传输（Bulk），中断传输(Interrupt)及同步传输(Isochronous)
  - 总线上无信号超过 3ms，支持自动暂停功能
  - –提供 6 个可编程端点(endpoints)
  - 512 字节内部 SRAM 作为 USB 的缓存区
  - 支持远程唤醒功能.
  - 支持 PDMA 模式

- ADC
  - 12 位 8 通道，转换速率达 800ksps.
  - 8 通道单端模式或 4 通道差分模式
  - 单一模式/单周期扫描模式/连续扫描模式
  - 每通道转换结果存放于独立寄存器内
  - 独立打开转换通道
  - 门槛电压侦测
  - 有软件或外部管脚触发开始转换
  - 支持 PDMA 模式
- 模拟比较器
  - 2 组模拟比较器模块
  - 基准电位可选择外部输入或内部参考电压
  - 转换结束可作为中断触发条件
  - 支持掉电模式唤醒功能
- 内建温度传感器， 1℃ 分辨率.
- 欠压检测
  - 支持四级检测电压: 4.5V/3.8V/2.7V/2.2V
  - 支持欠压中断和复位选择
- 内建一组 LDO
- 低电压复位
- 工作温度: -40℃~85℃
- 封装:
  - 无铅封装 (RoHS)
    - LQFP 100-pin / 64-pin / 48-pin

# 3 编号信息列表及管脚名称定义

## 3.1 产品选型指南

### 3.1.1 NUC101 产品选型指南

| 编号 | Flash (KB) | SRAM (KB) | 各总线界面 | | | | | | Comp. | PWM | ADC | I/O | Timer | RTC | ISP ICP | 封装 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | UART | SPI | I2C | USB | LIN | PS2 | | | | | | | | |
| NUC101LE3AN | 128 | 16 | 1 | 3 | 1 | 1 | 1 | 1 | 0 | 2 | 0 | up to 31 | 4x32-bit | v | v | LQFP48 |
| NUC101LD2AN | 64 | 8 | 1 | 3 | 1 | 1 | 1 | 1 | 0 | 2 | 0 | up to 31 | 4x32-bit | v | v | LQFP48 |
| NUC101LC1AN | 32 | 4 | 1 | 3 | 1 | 1 | 1 | 1 | 0 | 2 | 0 | up to 31 | 4x32-bit | v | v | LQFP48 |
| NUC101RE3AN | 128 | 16 | 2 | 4 | 1 | 1 | 2 | 1 | 0 | 0 | 8X12-bit | up to 43 | 4x32-bit | v | v | LQFP64 |
| NUC101RD2AN | 64 | 8 | 2 | 4 | 1 | 1 | 2 | 1 | 0 | 0 | 8X12-bit | up to 43 | 4x32-bit | v | v | LQFP64 |
| NUC101RC1AN | 32 | 4 | 2 | 4 | 1 | 1 | 2 | 1 | 0 | 0 | 8X12-bit | up to 43 | 4x32-bit | v | v | LQFP64 |
| NUC101VE3AN | 128 | 16 | 2 | 4 | 1 | 1 | 2 | 1 | 2 | 8 | 8X12-bit | up to 76 | 4x32-bit | v | v | LQFP100 |
| NUC101VD2AN | 64 | 8 | 2 | 4 | 2 | 1 | 2 | 1 | 2 | 8 | 8X12-bit | up to 76 | 4x32-bit | v | v | LQFP100 |
| NUC101VC1AN | 32 | 4 | 2 | 4 | 2 | 1 | 2 | 1 | 2 | 8 | 8X12-bit | up to 76 | 4x32-bit | v | v | LQFP100 |

### 3.1.2 NUC100 产品选型指南

#### *3.1.2.1 NUC100 系列选型指南(中端系列)*

| 编号 | Flash | SRAM | 各总线界面 | | | $I^2S$ | PWM | Comp. | ADC | Timer | RTC | ISP ICP | I/O | 封装 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | UART | SPI/SSI | $I^2C$ | | | | | | | | | |
| NUC100LE3AN | 128 KB | 16 KB | 2 | 1 | 2 | 1 | 6 | 1 | 8x12-bit | 4x24-bit | v | v | up to 35 | LQFP48 |
| NUC100LD3AN | 64 KB | 16 KB | 2 | 1 | 2 | 1 | 6 | 1 | 8x12-bit | 4x24-bit | v | v | up to 35 | LQFP48 |
| NUC100RE3AN | 128 KB | 16 KB | 3 | 2 | 2 | 1 | 6 | 2 | 8x12-bit | 4x24-bit | v | v | up to 49 | LQFP64 |
| NUC100RD3AN | 64 KB | 16 KB | 3 | 2 | 2 | 1 | 6 | 2 | 8x12-bit | 4x24-bit | v | v | up to 49 | LQFP64 |
| NUC100VE3AN | 128 KB | 16 KB | 3 | 4 | 2 | 1 | 8 | 2 | 8x12-bit | 4x24-bit | v | v | up to 80 | LQFP100 |
| NUC100VD3AN | 64 KB | 16 KB | 3 | 4 | 2 | 1 | 8 | 2 | 8x12-bit | 4x24-bit | v | v | up to 80 | LQFP100 |
| NUC100VD2AN | 64 KB | 8 KB | 3 | 4 | 2 | 1 | 8 | 2 | 8x12-bit | 4x24-bit | v | v | up to 80 | LQFP100 |

#### *3.1.2.2 NUC100 系列选型指南(低端系列)*

**※ 下列器件不支持 PDMA**

| 编号 | Flash | SRAM | 各总线界面 | | | $I^2S$ | PWM | Comp. | ADC | Timer | RTC | ISP ICP | I/O | 封装 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | UART | SPI/SSI | $I^2C$ | | | | | | | | | |
| NUC100LD2AN | 64 KB | 8 KB | 2 | 1 | 2 | 1 | 4 | 1 | 8x12-bit | 4x24-bit | v | v | up to 35 | LQFP48 |
| NUC100LD1AN | 64 KB | 4 KB | 2 | 1 | 2 | 1 | 4 | 1 | 8X12-Bit | 4x24-bit | v | v | up to 35 | LQFP48 |
| NUC100LC1AN | 32 KB | 4 KB | 2 | 1 | 2 | 1 | 4 | 1 | 8X12-Bit | 4x24-bit | v | v | up to 35 | LQFP48 |
| NUC100RD2AN | 64 KB | 8 KB | 2 | 2 | 2 | 1 | 4 | 2 | 8x12-bit | 4x24-bit | v | v | up to 49 | LQFP64 |
| NUC100RD1AN | 64 KB | 4 KB | 2 | 2 | 2 | 1 | 4 | 2 | 8X12-Bit | 4x24-bit | v | v | up to 49 | LQFP64 |
| NUC100RC1AN | 32 KB | 4 KB | 2 | 2 | 2 | 1 | 4 | 2 | 8X12-Bit | 4x24-bit | v | v | up to 49 | LQFP64 |

### 3.1.3 NUC120 产品选型指南

#### *3.1.3.1 NUC120 系列带USB的选型指南(中端系列)*

| 编号 | Flash | SRAM | 各总线界面 | | | | I²S | PWM | Comp. | ADC | Timer | RTC | ISP ICP | I/O | 封装 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | UART | SPI/SSI | I²C | USB | | | | | | | | | |
| NUC120LE3AN | 128 KB | 16 KB | 2 | 1 | 2 | 1 | 1 | 4 | 1 | 8x12-bit | 4x24-bit | v | v | up to 31 | LQFP48 |
| NUC120LD3AN | 64 KB | 16 KB | 2 | 1 | 2 | 1 | 1 | 4 | 1 | 8x12-bit | 4x24-bit | v | v | up to 31 | LQFP48 |
| NUC120RE3AN | 128 KB | 16 KB | 2 | 2 | 2 | 1 | 1 | 6 | 2 | 8x12-bit | 4x24-bit | v | v | up to 45 | LQFP64 |
| NUC120RD3AN | 64 KB | 16 KB | 2 | 2 | 2 | 1 | 1 | 6 | 2 | 8x12-bit | 4x24-bit | v | v | up to 45 | LQFP64 |
| NUC120VE3AN | 128 KB | 16 KB | 3 | 4 | 2 | 1 | 1 | 8 | 2 | 8x12-bit | 4x24-bit | v | v | up to 76 | LQFP100 |
| NUC120VD3AN | 64 KB | 16 KB | 3 | 4 | 2 | 1 | 1 | 8 | 2 | 8x12-bit | 4x24-bit | v | v | up to 76 | LQFP100 |
| NUC120VD2AN | 64 KB | 8 KB | 3 | 4 | 2 | 1 | 1 | 8 | 2 | 8x12-bit | 4x24-bit | v | v | up to 76 | LQFP100 |

#### *3.1.3.2 NUC120 系列带USB的选型指南(低端系列)*

**※下列器件不支持PDMA**

| 编号 | Flash | SRAM | 各总线界面 | | | | I²S | PWM | Comp. | ADC | Timer | RTC | ISP ICP | I/O | 封装 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | UART | SPI/SSI | I²C | USB | | | | | | | | | |
| NUC120LD2AN | 64 KB | 8 KB | 2 | 1 | 2 | 1 | 1 | 4 | 1 | 8x12-bit | 4x24-bit | v | v | up to 31 | LQFP48 |
| NUC120LD1AN | 64 KB | 4 KB | 2 | 1 | 2 | 1 | 1 | 4 | 1 | 8X12-Bit | 4x24-bit | v | v | up to 31 | LQFP48 |
| NUC120LC1AN | 32 KB | 4 KB | 2 | 1 | 2 | 1 | 1 | 4 | 1 | 8X12-Bit | 4x24-bit | v | v | up to 31 | LQFP48 |
| NUC120RD2AN | 64 KB | 8 KB | 2 | 2 | 2 | 1 | 1 | 4 | 2 | 8x12-bit | 4x24-bit | v | v | up to 45 | LQFP64 |
| NUC120RD1AN | 64 KB | 4 KB | 2 | 2 | 2 | 1 | 1 | 4 | 2 | 8X12-Bit | 4x24-bit | v | v | up to 45 | LQFP64 |
| NUC120RC1AN | 32 KB | 4 KB | 2 | 2 | 2 | 1 | 1 | 4 | 2 | 8X12-Bit | 4x24-bit | v | v | up to 45 | LQFP64 |

### 3.1.4 NUC130 产品选型指南

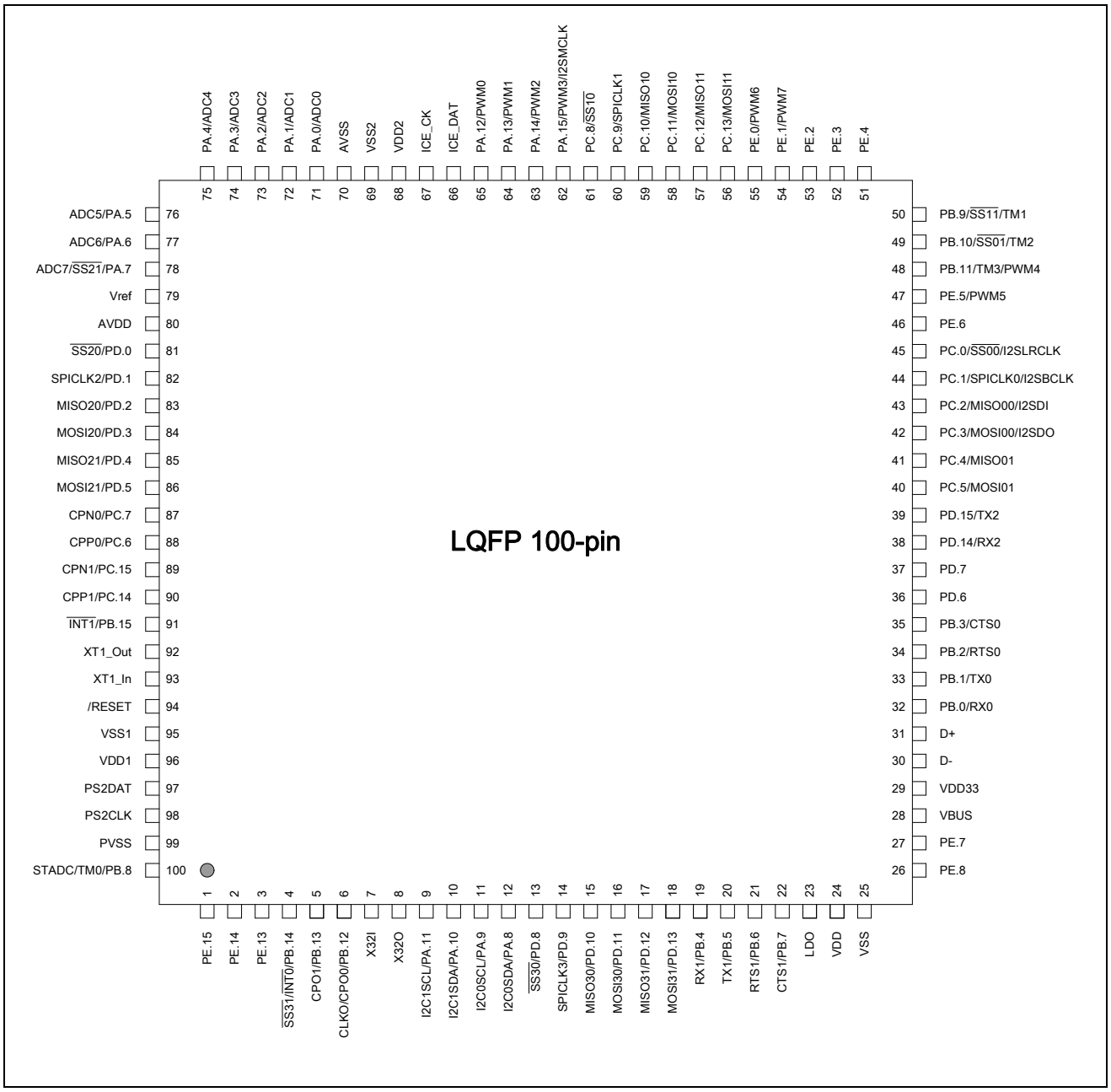
| 编号 | Flash | SRAM | 各总线界面 | | | | | $I^2S$ | PWM | Comp. | ADC | Timer | RTC | ISP ICP | I/O | 封装 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | UART | SPI/SSI | $I^2C$ | LIN | CAN | | | | | | | | | |
| NUC130LE3AN | 128 KB | 16 KB | 3 | 1 | 2 | 2 | 1 | 1 | 4 | 1 | 8x12-bit | 4x24-bit | v | v | up to 35 | LQFP48 |
| NUC130LD3AN | 64 KB | 16 KB | 3 | 1 | 2 | 2 | 1 | 1 | 4 | 1 | 8x12-bit | 4x24-bit | v | v | up to 35 | LQFP48 |
| NUC130LD2AN | 64 KB | 8 KB | 3 | 1 | 2 | 2 | 1 | 1 | 4 | 1 | 8x12-bit | 4x24-bit | v | v | up to 35 | LQFP48 |
| NUC130RE3AN | 128 KB | 16 KB | 3 | 2 | 2 | 2 | 1 | 1 | 6 | 2 | 8x12-bit | 4x24-bit | v | v | up to 49 | LQFP64 |
| NUC130RD3AN | 64 KB | 16 KB | 3 | 2 | 2 | 2 | 1 | 1 | 6 | 2 | 8x12-bit | 4x24-bit | v | v | up to 49 | LQFP64 |
| NUC130RD2AN | 64 KB | 8 KB | 3 | 2 | 2 | 2 | 1 | 1 | 6 | 2 | 8x12-bit | 4x24-bit | v | v | up to 49 | LQFP64 |
| NUC130VE3AN | 128 KB | 16 KB | 3 | 4 | 2 | 2 | 1 | 1 | 8 | 2 | 8x12-bit | 4x24-bit | v | v | up to 80 | LQFP100 |
| NUC130VD3AN | 64 KB | 16 KB | 3 | 4 | 2 | 2 | 1 | 1 | 8 | 2 | 8x12-bit | 4x24-bit | v | v | up to 80 | LQFP100 |
| NUC130VD2AN | 64 KB | 8 KB | 3 | 4 | 2 | 2 | 1 | 1 | 8 | 2 | 8x12-bit | 4x24-bit | v | v | up to 80 | LQFP100 |

### 3.1.5 NUC140 产品选型指南

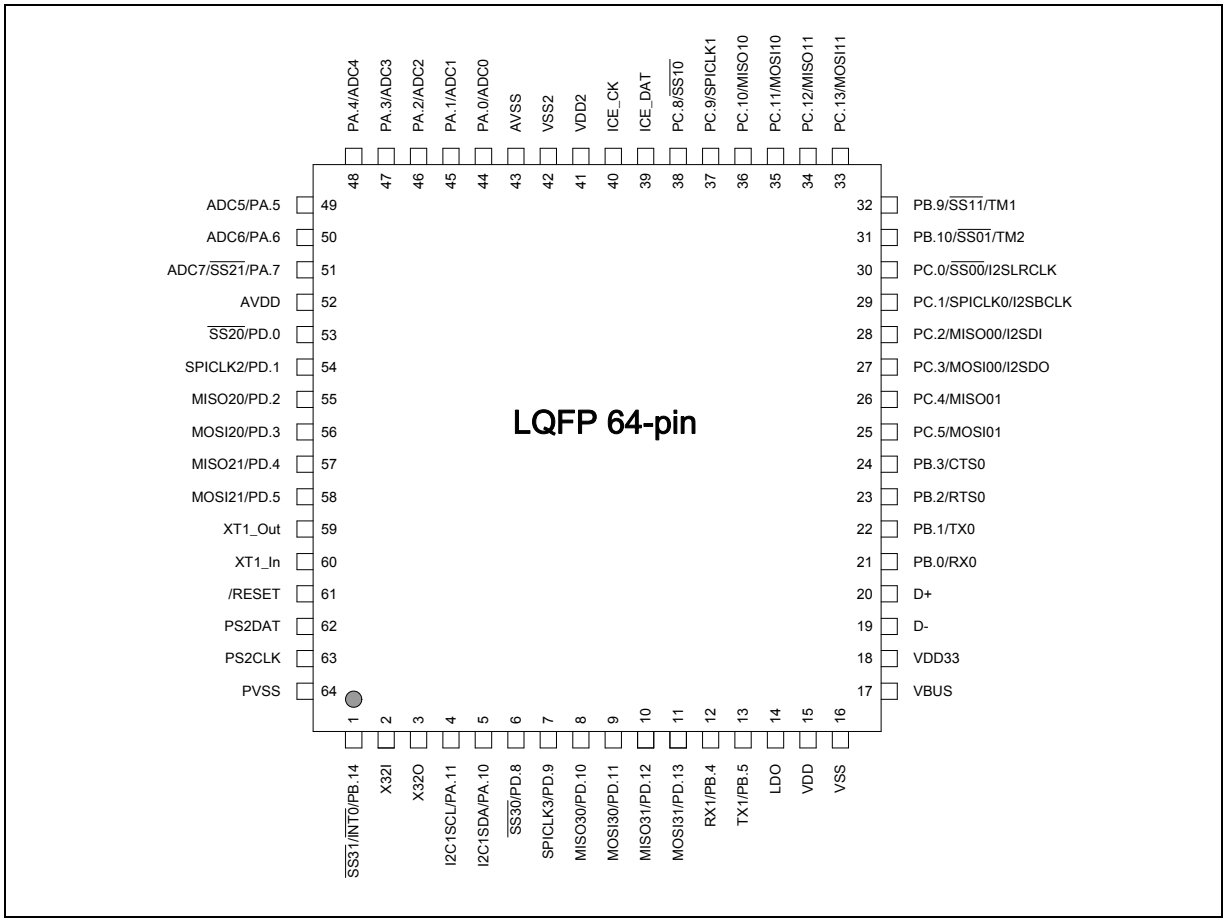
| 编号 | Flash | SRAM | 各总线界面 | | | | | | $I^2S$ | PWM | Comp. | ADC | Timer | RTC | ISP ICP | I/O | 封装 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | UART | SPI/SSI | $I^2C$ | USB | LIN | CAN | | | | | | | | | |
| NUC140LE3AN | 128 KB | 16 KB | 2 | 1 | 2 | 1 | 2 | 1 | 1 | 4 | 1 | 8x12-bit | 4x24-bit | v | v | up to 31 | LQFP48 |
| NUC140LD3AN | 64 KB | 16 KB | 2 | 1 | 2 | 1 | 2 | 1 | 1 | 4 | 1 | 8x12-bit | 4x24-bit | v | v | up to 31 | LQFP48 |
| NUC140LD2AN | 64 KB | 8 KB | 2 | 1 | 2 | 1 | 2 | 1 | 1 | 4 | 1 | 8x12-bit | 4x24-bit | v | v | up to 31 | LQFP48 |
| NUC140RE3AN | 128 KB | 16 KB | 3 | 2 | 2 | 1 | 2 | 1 | 1 | 4 | 2 | 8x12-bit | 4x24-bit | v | v | up to 45 | LQFP64 |
| NUC140RD3AN | 64 KB | 16 KB | 3 | 2 | 2 | 1 | 2 | 1 | 1 | 4 | 2 | 8x12-bit | 4x24-bit | v | v | up to 45 | LQFP64 |
| NUC140RD2AN | 64 KB | 8 KB | 3 | 2 | 2 | 1 | 2 | 1 | 1 | 4 | 2 | 8x12-bit | 4x24-bit | v | v | up to 45 | LQFP64 |
| NUC140VE3AN | 128 KB | 16 KB | 3 | 4 | 2 | 1 | 2 | 1 | 1 | 8 | 2 | 8x12-bit | 4x24-bit | v | v | up to 76 | LQFP100 |
| NUC140VD3AN | 64 KB | 16 KB | 3 | 4 | 2 | 1 | 2 | 1 | 1 | 8 | 2 | 8x12-bit | 4x24-bit | v | v | up to 76 | LQFP100 |
| NUC140VD2AN | 64 KB | 8 KB | 3 | 4 | 2 | 1 | 2 | 1 | 1 | 8 | 2 | 8x12-bit | 4x24-bit | v | v | up to 76 | LQFP100 |

## 3.2 管脚配置

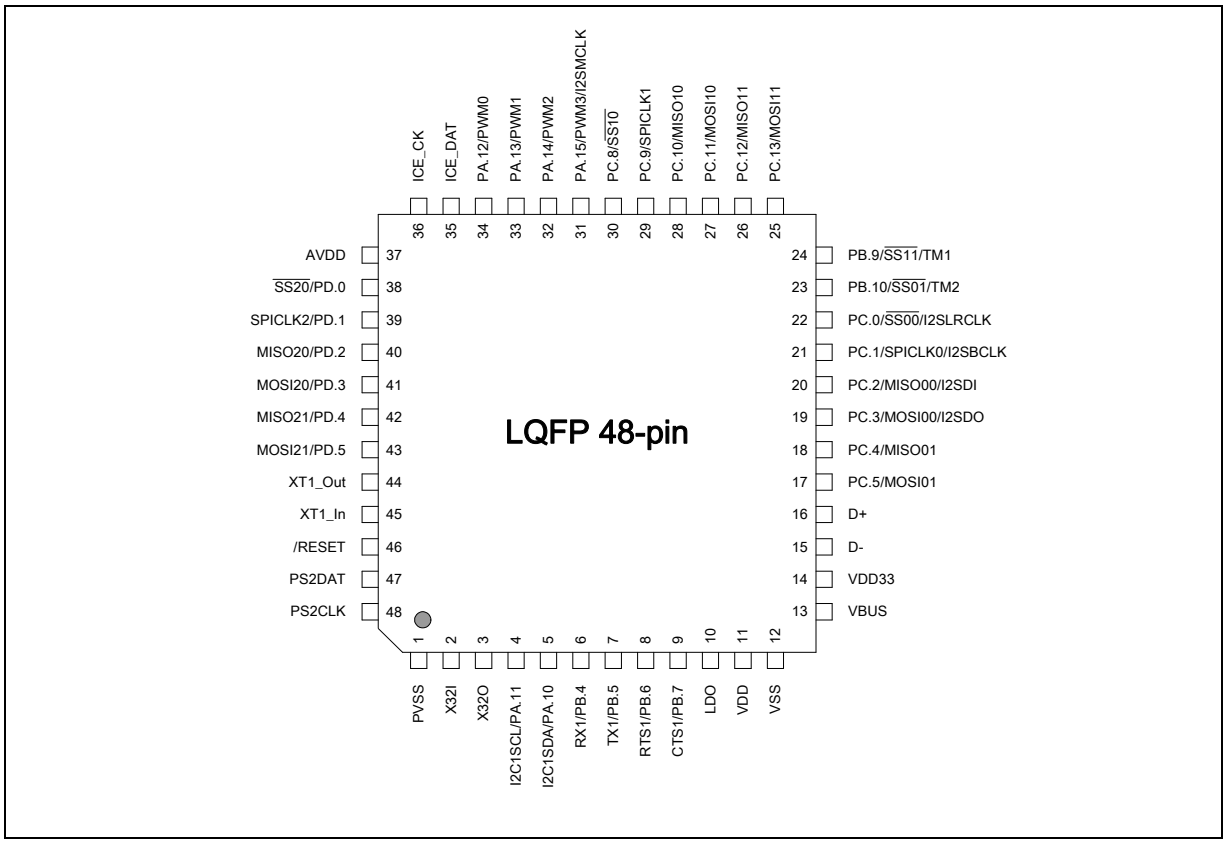
### 3.2.1 NUC101 LQFP 100 pin



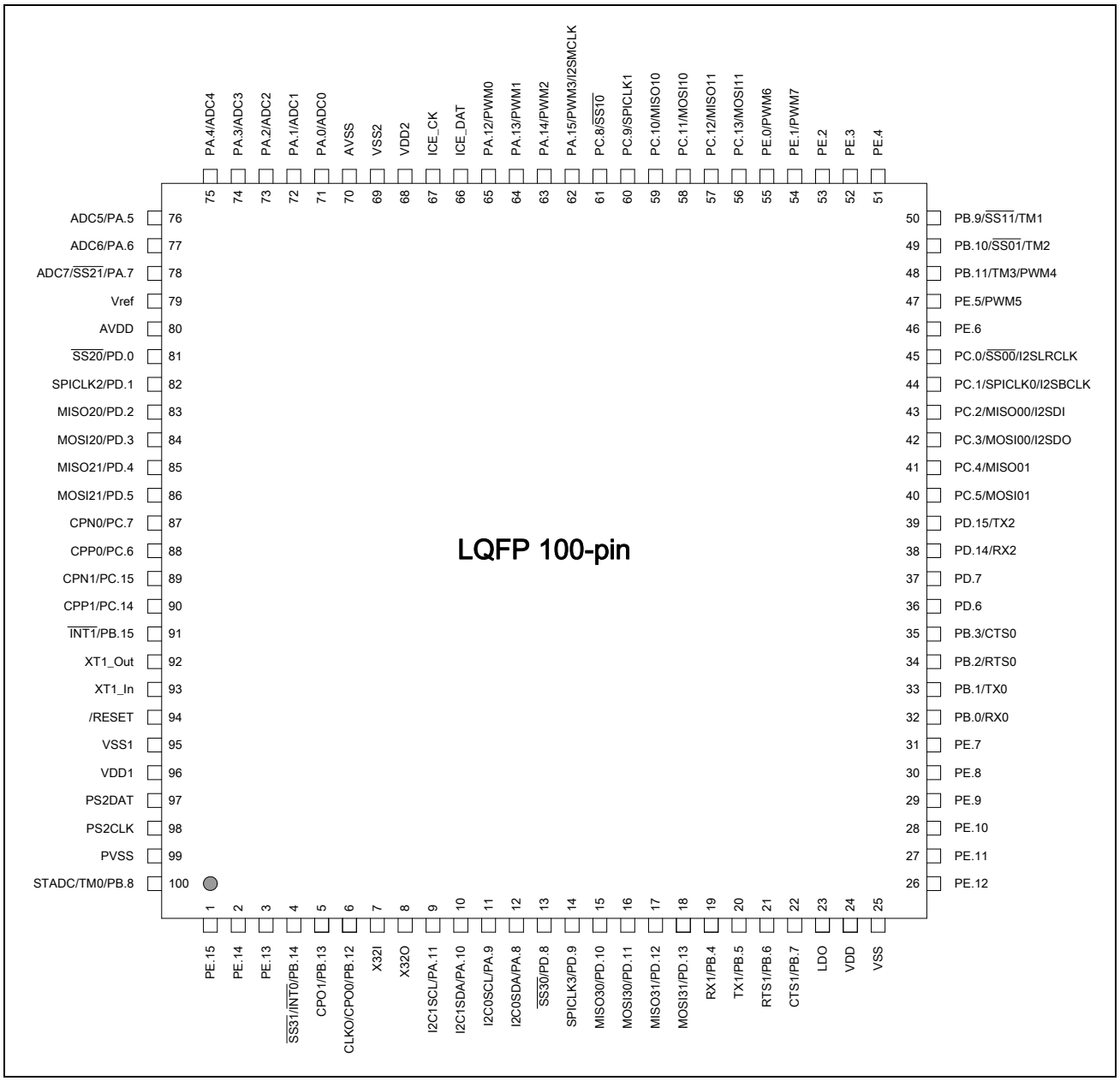
### 3.2.2 NUC101 LQFP 64 pin



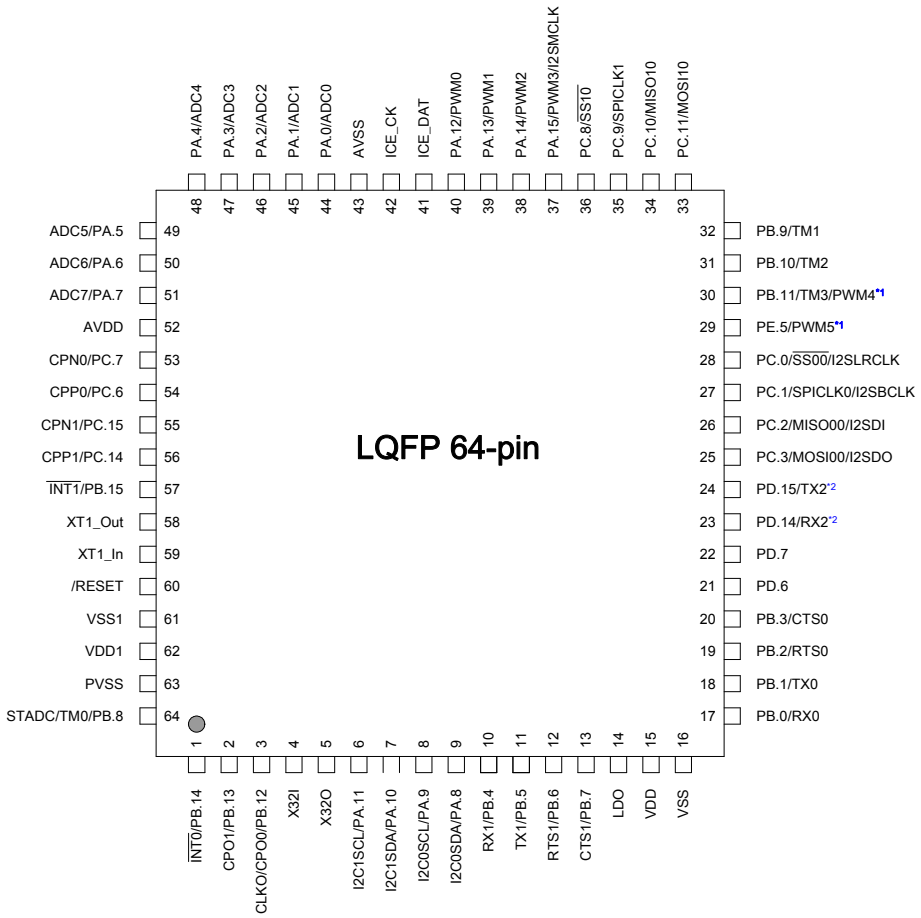
### 3.2.3 NUC101 LQFP 48 pin



### 3.2.4 NUC100 LQFP 100 pin

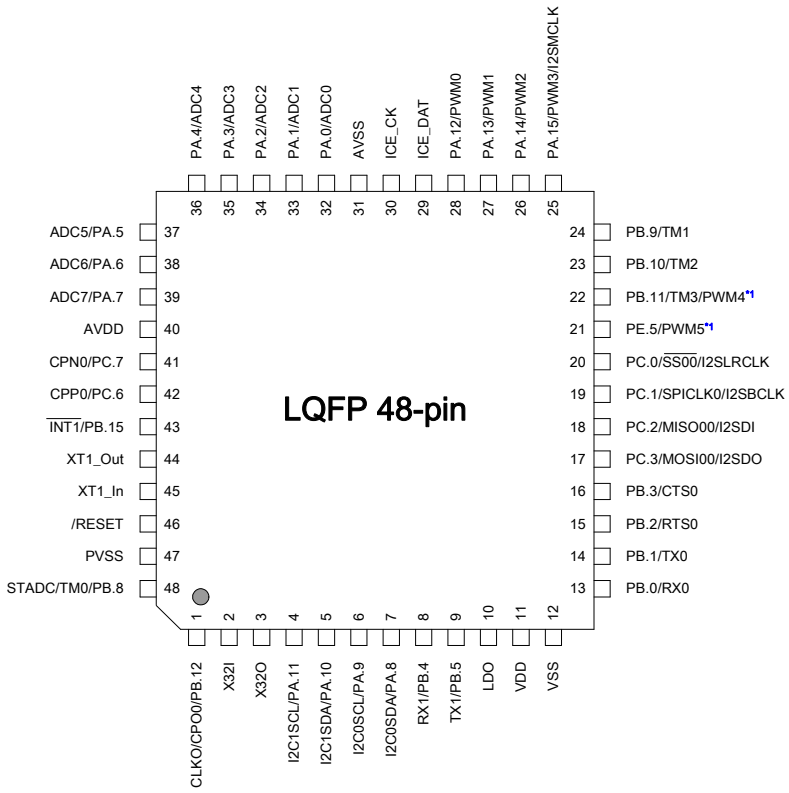


### 3.2.5 NUC100 LQFP 64 pin



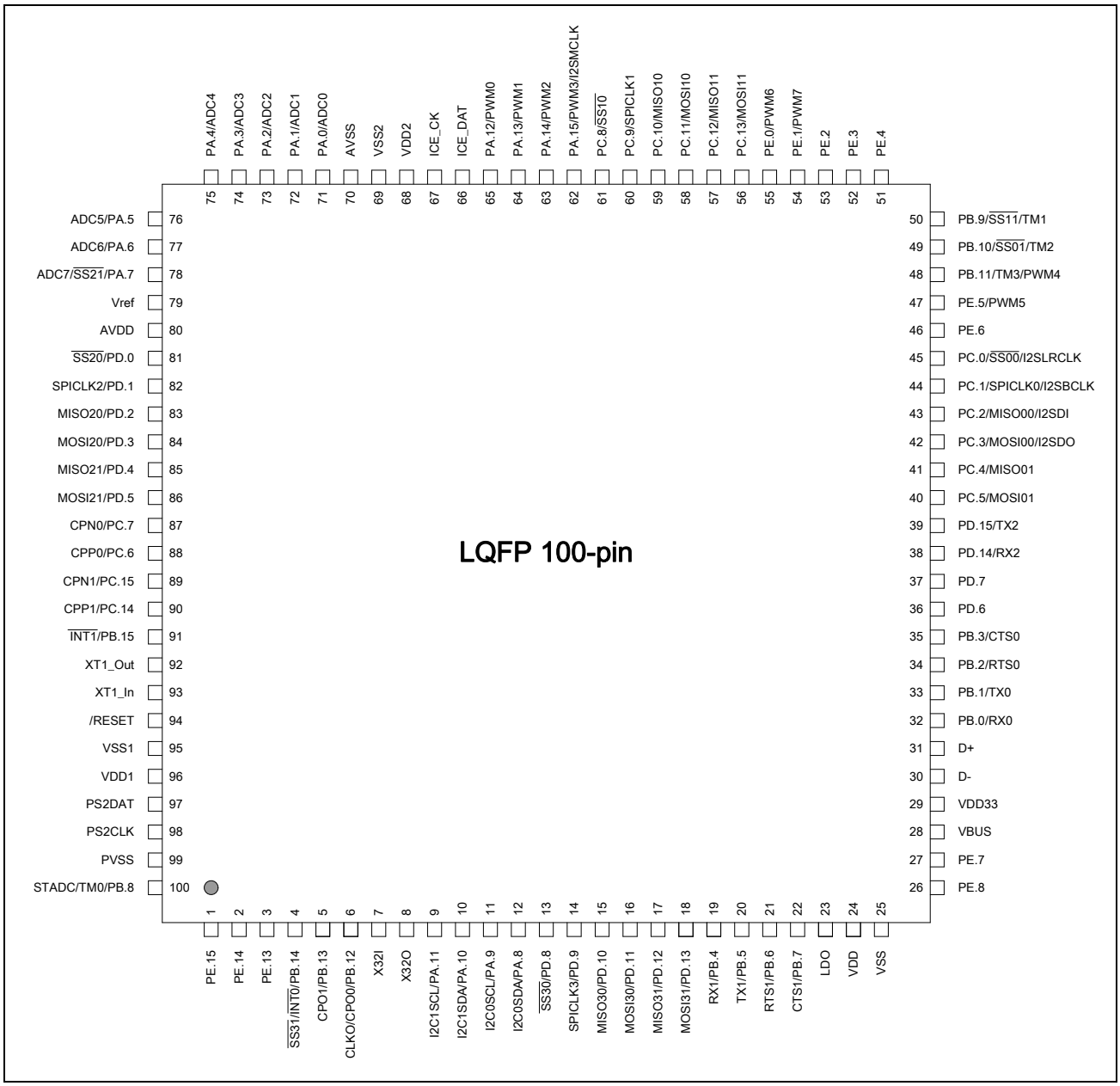

**注:**
**1. PWM4 及 PWM5 仅在中阶芯片中提供支持.**
**2. TX2 及 RX2 仅在中阶芯片中提供支持.**

### 3.2.6 NUC100 LQFP 48 pin

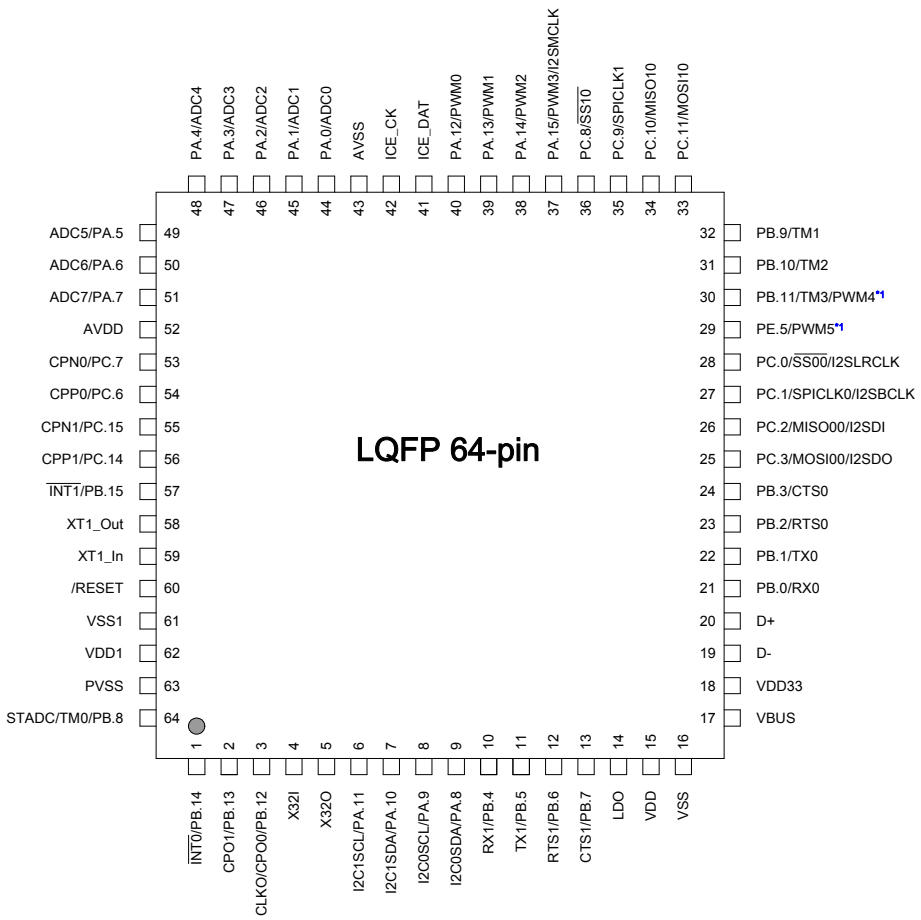



**注:**
**1. PWM4 及 PWM5 仅在中阶芯片中提供支持.**

### 3.2.7 NUC120 LQFP 100 pin



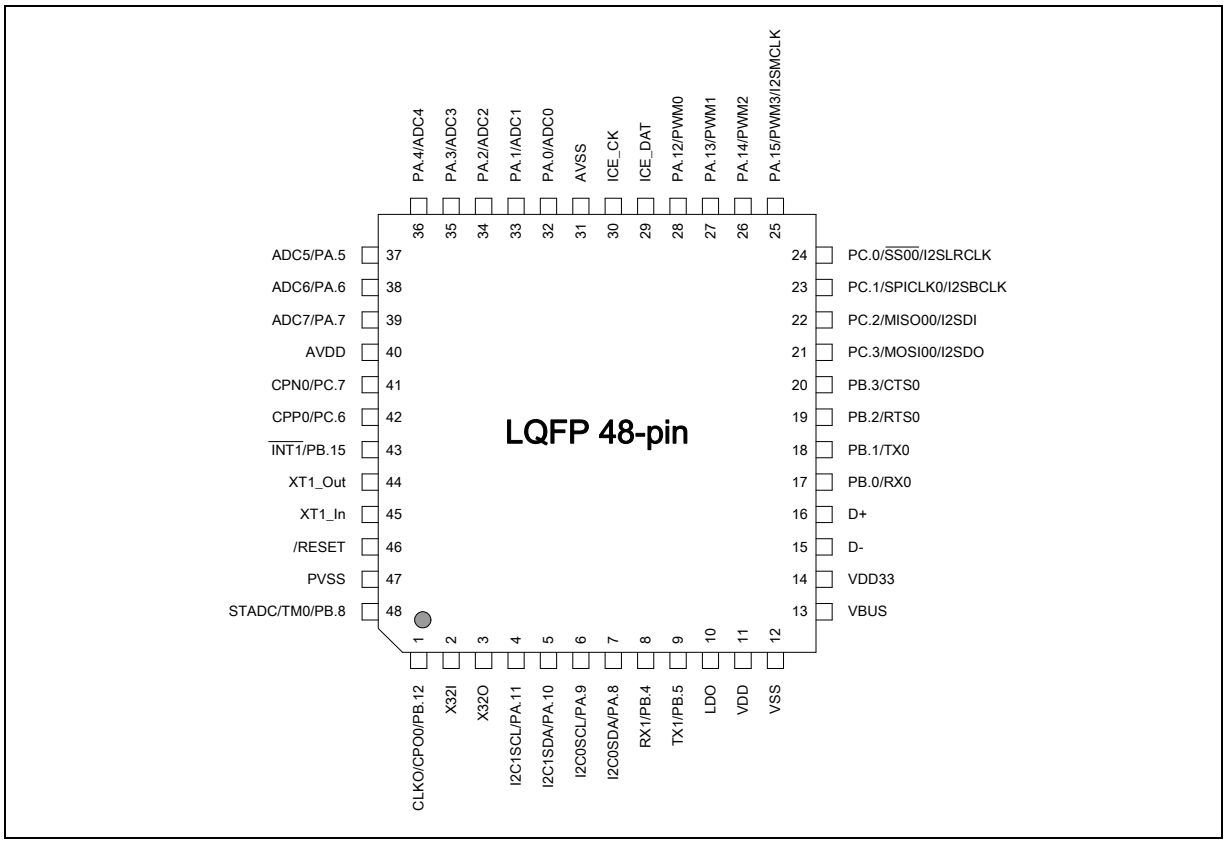
### 3.2.8 NUC120 LQFP 64 pin



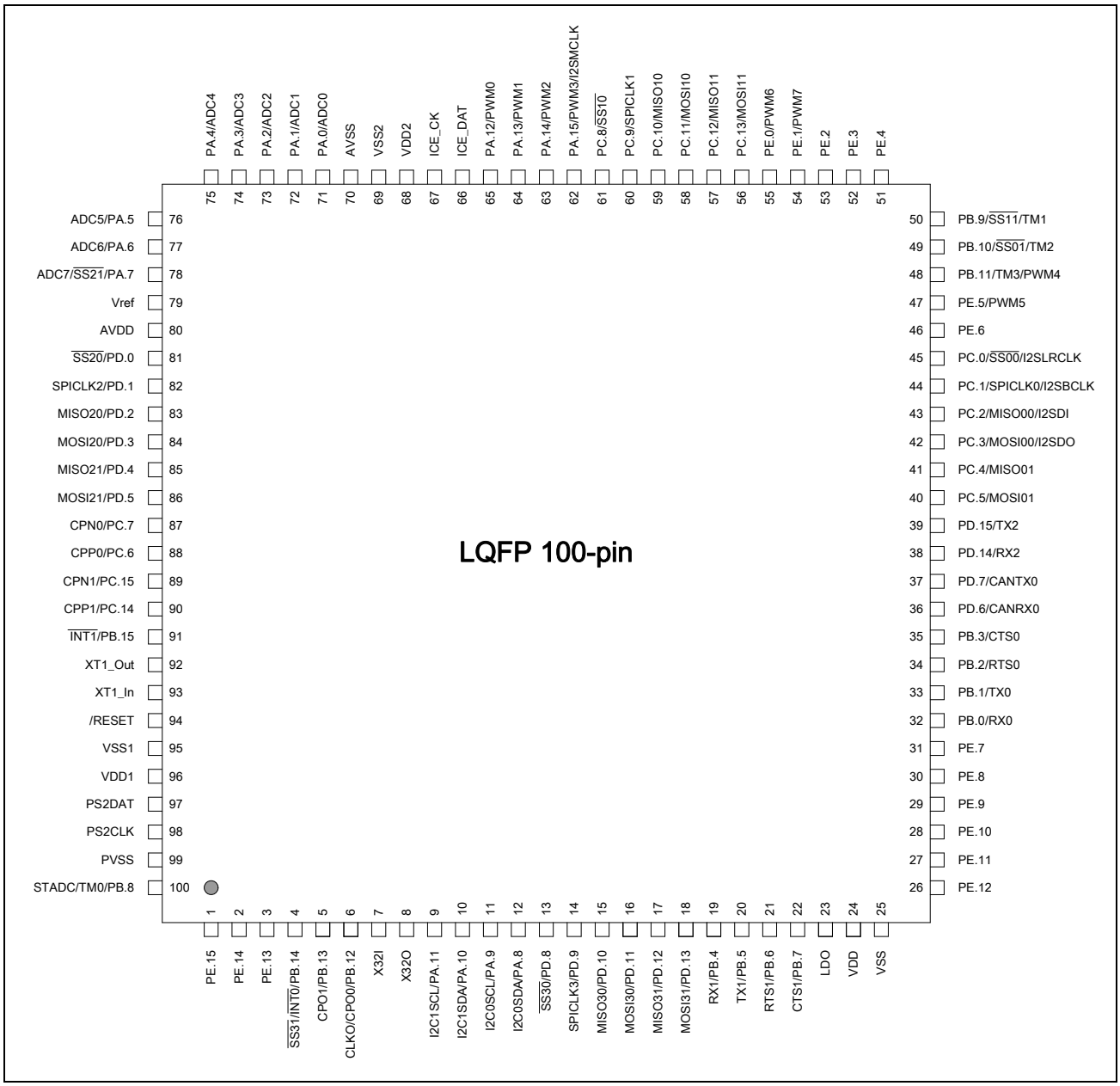

**注:**

**1. PWM4 及 PWM5 仅在中阶芯片中提供支持**

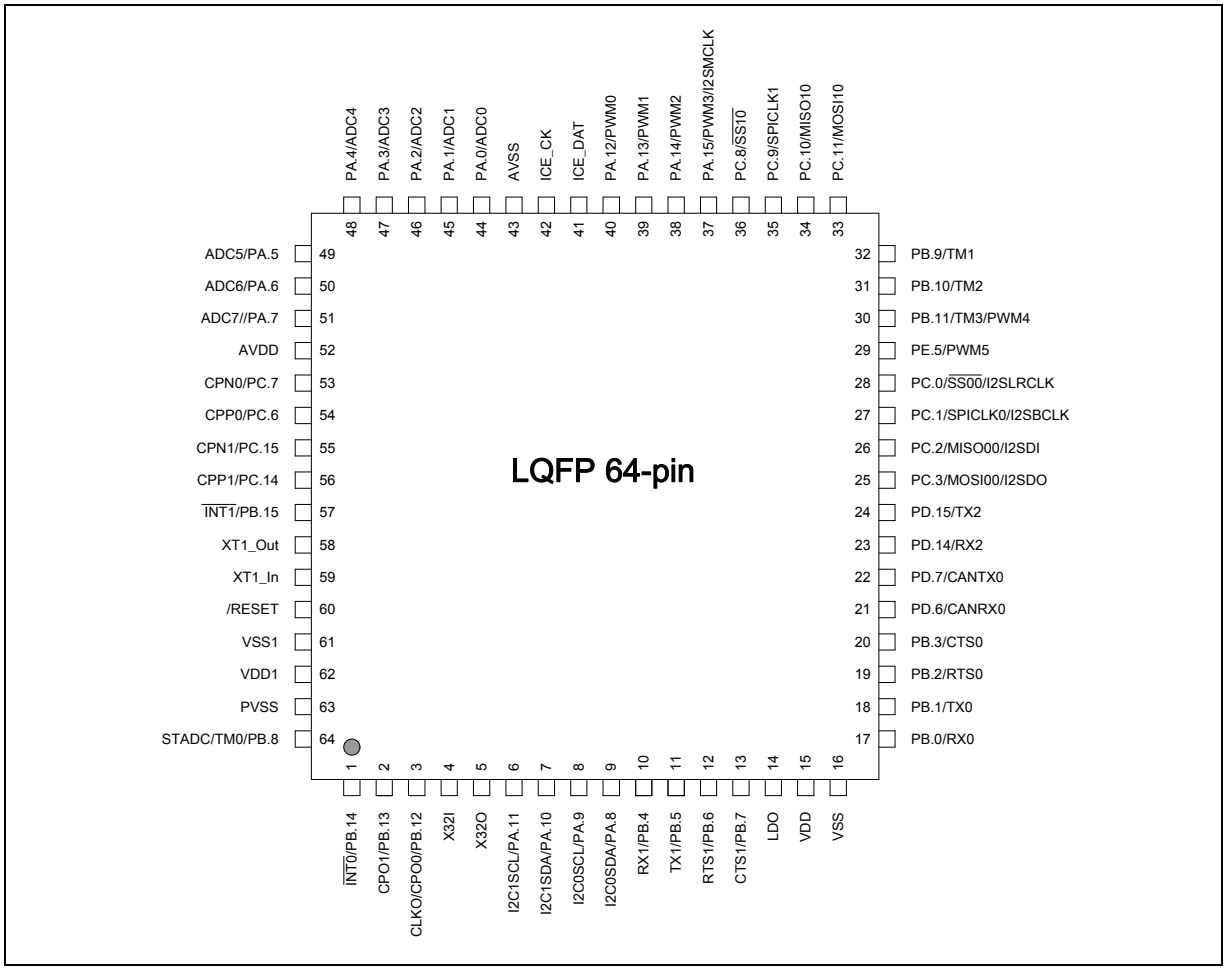
### 3.2.9 NUC120 LQFP 48 pin



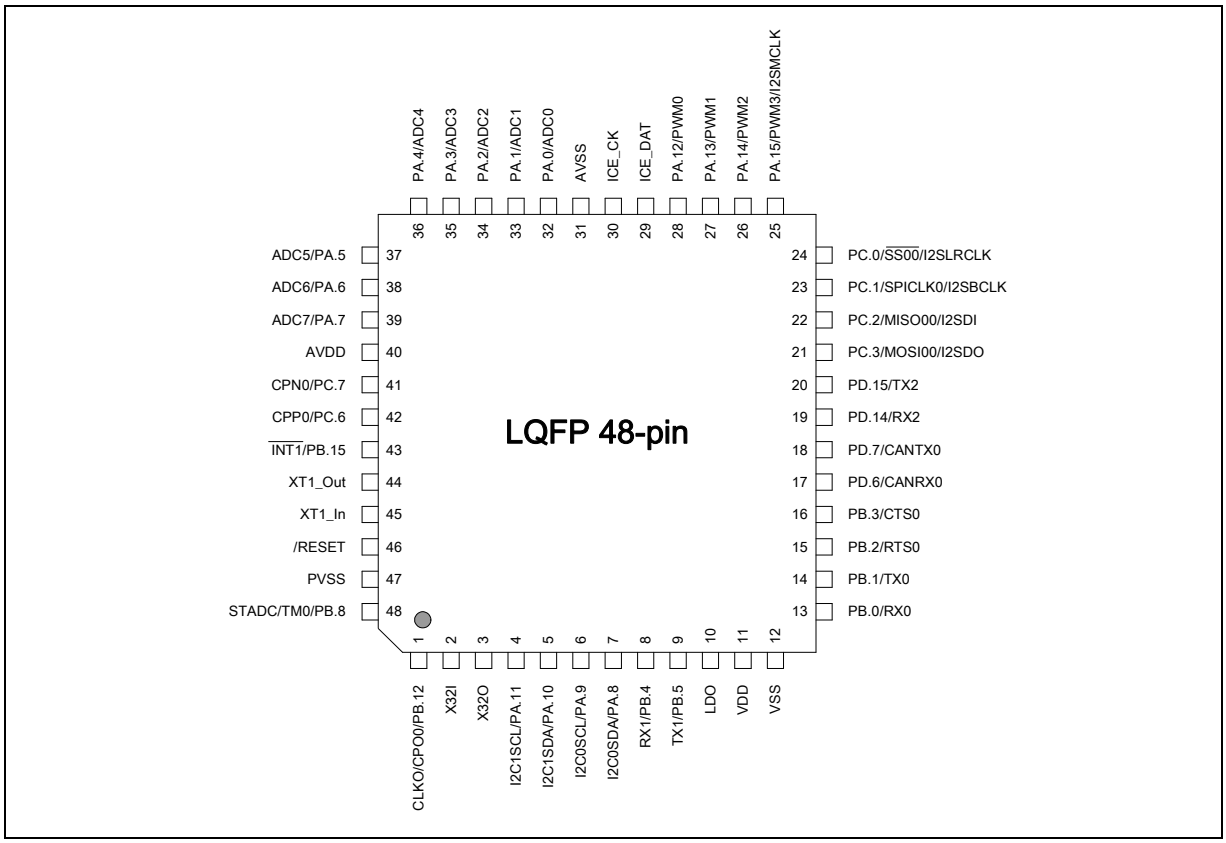
### 3.2.10 NUC130 LQFP 100 pin



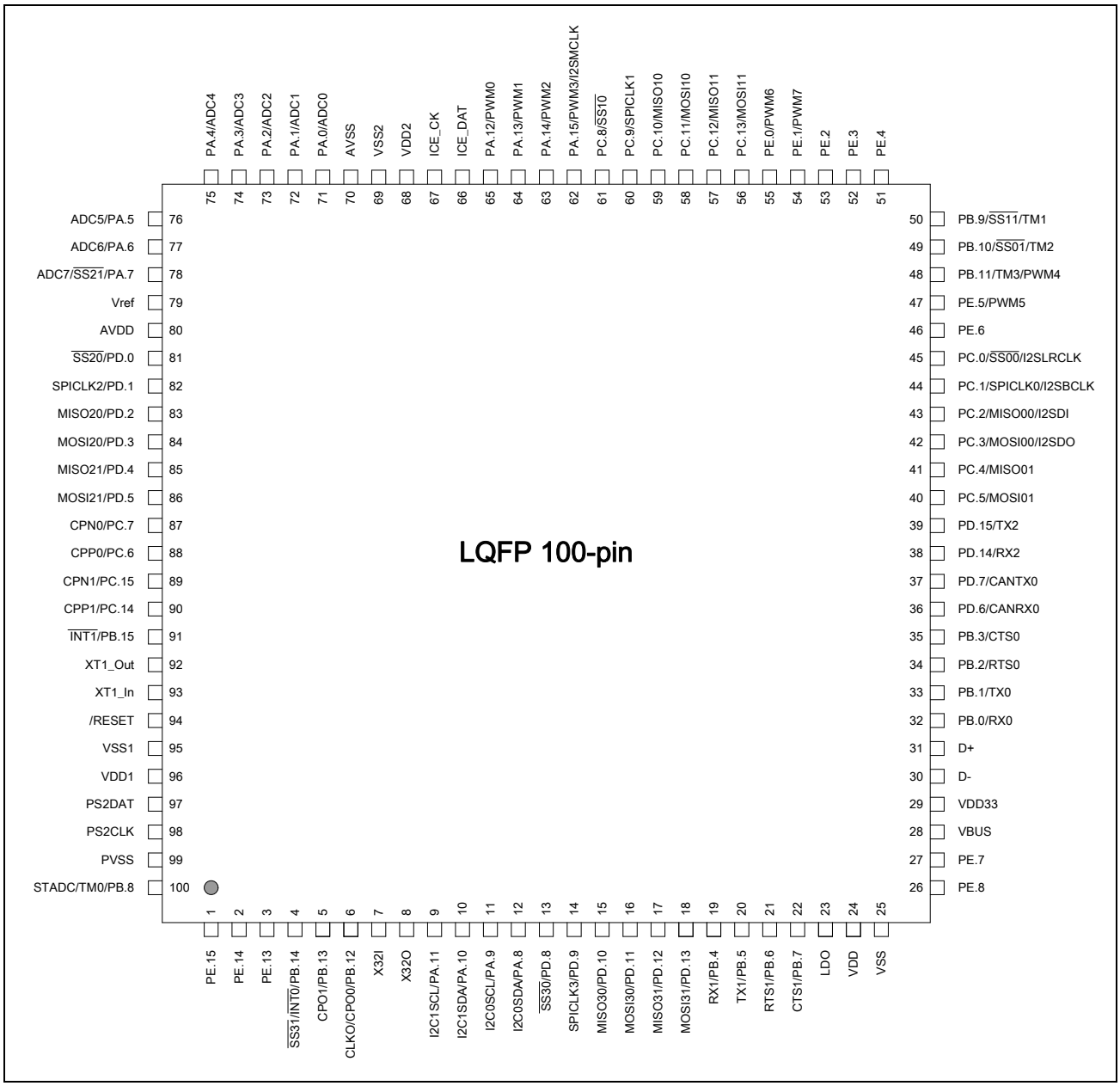
### 3.2.11 NUC130 LQFP 64 pin



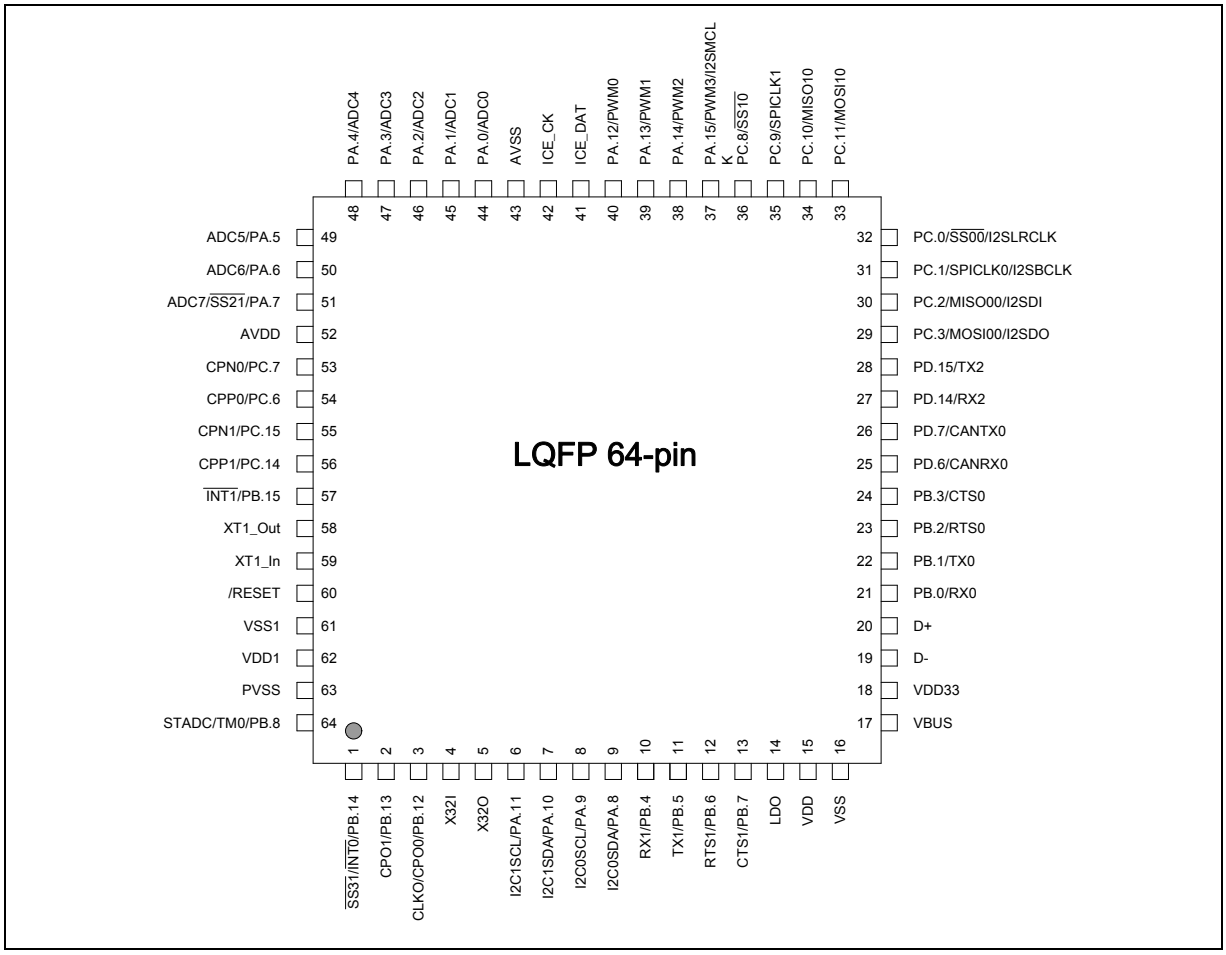
### 3.2.12 NUC130 LQFP 48 pin



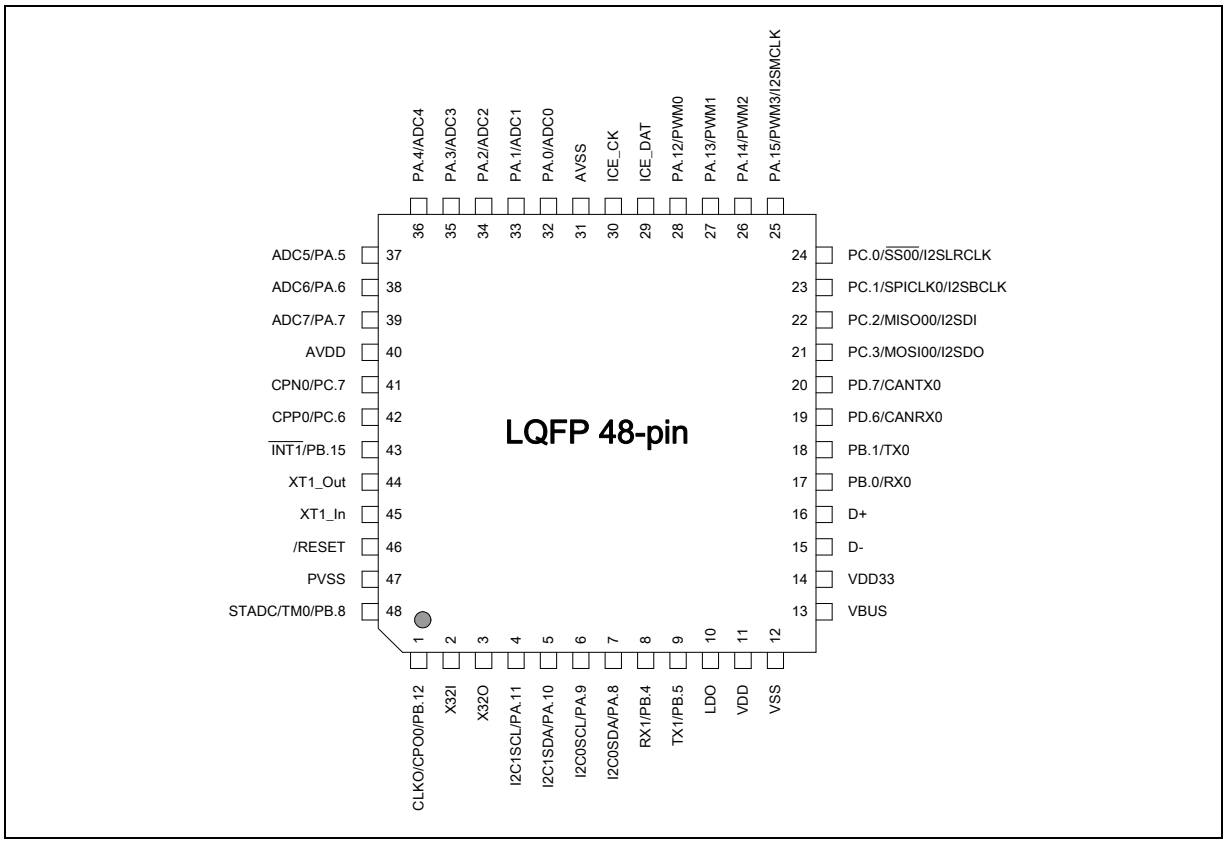
### 3.2.13 NUC140 LQFP 100 pin



### 3.2.14 NUC140 LQFP 64 pin



### 3.2.15 NUC140 LQFP 48 pin



## 3.3 管脚功能描述

### 3.3.1 NUC101 管脚定义

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| 1 | | | PE.15 | I/O | | 标准数字输入/输出管脚 |
| 2 | | | PE.14 | I/O | | 标准数字输入/输出管脚 |
| 3 | | | PE.13 | I/O | | 标准数字输入/输出管脚 |
| 4 | 1 | | PB.14 | I/O | | 标准数字输入/输出管脚 |
| | | | /INT0 | I | | /INT0: 外部中断1输入管脚 |
| | | | /SPISS31 | I/O | | /SPISS31: SPI3 2nd 从机选择脚 |
| 5 | | | PB.13 | I/O | | 标准数字输入/输出管脚 |
| | | | CPO1 | O | | 模拟比较器1输出管脚 |
| 6 | | | PB.12 | I/O | | 标准数字输入/输出管脚 |
| | | | CPO0 | O | | 模拟比较器0输出管脚 |
| | | | CLKO | O | | 分频器输出管理 |
| 7 | 2 | 2 | X32I | I | | 32.768KHz 外部晶振输入脚 |
| 8 | 3 | 3 | X32O | O | | 32.768KHz外部晶振输出脚 |
| 9 | 4 | 4 | PA.11 | I/O | | 标准数字输入/输出管脚 |
| | | | I2C1SCL | I/O | | I2C1SCL: I2C1 时钟脚 |
| 10 | 5 | 5 | PA.10 | I/O | | 标准数字输入/输出管脚 |
| | | | I2C1SDA | I/O | | I2C1SDA: I2C1数据输入输出脚 |
| 11 | | | PA.9 | I/O | | 标准数字输入/输出管脚 |
| | | | I2C0SCL | I/O | | I2C0SCL: I2C0时钟脚 |
| 12 | | | PA.8 | I/O | | 标准数字输入/输出管脚 |
| | | | I2C0SDA | I/O | | I2C0SDA: I2C0数据输入输出脚 |
| 13 | 6 | | PD.8 | I/O | | 标准数字输入/输出管脚 |
| | | | /SPISS30 | I/O | | /SPISS30: SPI3 从机选择脚 |
| 14 | 7 | | PD.9 | I/O | | 标准数字输入/输出管脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| | | | SPICLK3 | **I/O** | | SPICLK3: SPI3 串行时钟脚 |
| 15 | 8 | | PD.10 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MISO30 | **I** | | MISO30: SPI3 MISO(主机输入，从机输出) pin |
| 16 | 9 | | PD.11 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MOSI30 | **O** | | MOSI30: SPI3 MOSI(主机输出，从机输入) pin |
| 17 | 10 | | PD.12 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MISO31 | **I** | | MISO31: SPI3 2nd MISO(主机输入，从机输出) pin |
| 18 | 11 | | PD.13 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MOSI31 | **O** | | MOSI31: SPI3 2nd MOSI(主机输出，从机输入) pin |
| 19 | 12 | 4 | PB.4 | **I/O** | | 标准数字输入/输出管脚 |
| | | | RX1 | **I** | | RX1: UART1数据接收输入脚 |
| 20 | 13 | 5 | PB.5 | **I/O** | | 标准数字输入/输出管脚 |
| | | | TX1 | **O** | | TX1: UART1数据发送输出脚 |
| 21 | | 6 | PB.6 | **I/O** | | 标准数字输入/输出管脚 |
| | | | RTS1 | | | RTS1: UART1请求发送输出脚 |
| 22 | | 7 | PB.7 | **I/O** | | 标准数字输入/输出管脚 |
| | | | CTS1 | | | CTS1: UART1清发送输入脚 |
| 23 | 14 | 8 | LDO | **P** | | LDO 输出pin |
| 24 | 15 | 9 | VDD | **P** | | 电源(Power supply for I/O ports and LDO source for internal PLL and digital function) |
| 25 | 16 | 10 | VSS | **P** | | 地 |
| 26 | | | PE.8 | **I/O** | | 标准数字输入/输出管脚 |
| 27 | | | PE.7 | **I/O** | | 标准数字输入/输出管脚 |
| 28 | 17 | 13 | VBUS | **USB** | | USB HOST或HUB提供电源脚. |
| 29 | 18 | 14 | VDD33 | **USB** | | 内部电源输出 3.3V标准电位脚 |
| 30 | 19 | 15 | D- | **USB** | | USB 差分信号 D- |
| 31 | 20 | 16 | D+ | **USB** | | USB 差分信号 D+ |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| 32 | 21 | | PB.0 | I/O | | 标准数字输入/输出管脚 |
| | | | RX0 | I | | RX0: UART0数据接收输入脚 |
| 33 | 22 | | PB.1 | I/O | | 标准数字输入/输出管脚 |
| | | | TX0 | O | | TX0: UART0数据发送输出脚 |
| 34 | 23 | | PB.2 | I/O | | 标准数字输入/输出管脚 |
| | | | RTS0 | | | RTS0: UART0请求发送输出脚 |
| 35 | 24 | | PB.3 | I/O | | 标准数字输入/输出管脚 |
| | | | CTS0 | | | CTS0: UART0清发送输入脚 |
| 36 | | | PD.6 | I/O | | 标准数字输入/输出管脚 |
| 37 | | | PD.7 | I/O | | 标准数字输入/输出管脚 |
| 38 | | | PD.14 | I/O | | 标准数字输入/输出管脚 |
| 39 | | | PD.15 | I/O | | 标准数字输入/输出管脚 |
| 40 | 25 | 17 | PC.5 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI01 | O | | MOSI01: SPI0 2nd MOSI(主机输出，从机输入) pin |
| 41 | 26 | 18 | PC.4 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO01 | I | | MISO01: SPI0 2nd MISO(主机输入，从机输出) pin |
| 42 | 27 | 19 | PC.3 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI00 | O | | MOSI00: SPI0 MOSI(主机输出，从机输入) pin |
| | | | I2SDO | O | | I2SDO: I2S 数据输出 |
| 43 | 28 | 20 | PC.2 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO00 | I | | MISO00: SPI0 MISO(主机输入，从机输出) pin |
| | | | I2SDI | I | | I2SDI: I2S 数据输入 |
| 44 | 29 | 21 | PC.1 | I/O | | 标准数字输入/输出管脚 |
| | | | SPICLK0 | I/O | | SPICLK0: SPI0串行时钟脚 |
| | | | I2SBCLK | I/O | | I2SBCLK: I2S bit 时钟脚 |
| 45 | 30 | 22 | PC.0 | I/O | | 标准数字输入/输出管脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| | | | /SPISS00 | I/O | | /SPISS00: SPI0 从机选择脚 |
| | | | I2SLRCLK | I/O | | I2SLRCLK: I2S 左右声道时钟 |
| 46 | | | PE.6 | I/O | | 标准数字输入/输出管脚 |
| 47 | | | PE.5 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM5 | O | | PWM5: PWM 输出 |
| 48 | | | PB.11 | I/O | | 标准数字输入/输出管脚 |
| | | | TM3 | O | | TM3: Timer3 外部计数器输入 |
| | | | PWM4 | O | | PWM4: PWM 输出 |
| 49 | 31 | 23 | PB.10 | I/O | | 标准数字输入/输出管脚 |
| | | | TM2 | O | | TM2: Timer2 外部计数器输入 |
| | | | /SPISS01 | I/O | | /SPISS01: SPI0 2nd 从机选择脚 |
| 50 | 32 | 24 | PB.9 | I/O | | 标准数字输入/输出管脚 |
| | | | TM1 | O | | TM1: Timer1外部计数器输入 |
| | | | /SPISS11 | I/O | | /SPISS11: SPI1 2nd 从机选择脚 |
| 51 | | | PE.4 | I/O | | 标准数字输入/输出管脚 |
| 52 | | | PE.3 | I/O | | 标准数字输入/输出管脚 |
| 53 | | | PE.2 | I/O | | 标准数字输入/输出管脚 |
| 54 | | | PE.1 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM7 | O | | PWM7: PWM 输出 |
| 55 | | | PE.0 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM6 | O | | PWM6: PWM 输出 |
| 56 | 33 | 25 | PC.13 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI11 | O | | MOSI11: SPI1 2nd MOSI(主机输出，从机输入) pin |
| 57 | 34 | 26 | PC.12 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO11 | I | | MISO11: SPI1 2nd MISO(主机输入，从机输出) pin |
| 58 | 35 | 27 | PC.11 | I/O | | 标准数字输入/输出管脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| | | | MOSI10 | O | | MOSI10: SPI1 MOSI(主机输出，从机输入) pin |
| 59 | 36 | 28 | PC.10 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO10 | I | | MISO10: SPI1 MISO(主机输入，从机输出) pin |
| 60 | 37 | 29 | PC.9 | I/O | | 标准数字输入/输出管脚 |
| | | | SPICLK1 | I/O | | SPICLK1: SPI1串行时钟脚 |
| 61 | 38 | 30 | PC.8 | I/O | | 标准数字输入/输出管脚 |
| | | | /SPISS10 | I/O | | /SPISS10: SPI1 从机选择脚 |
| 62 | | 31 | PA.15 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM3 | O | | PWM3: PWM 输出脚 |
| | | | I2SMCLK | O | | I2SMCLK: I2S 主机时钟输出脚 |
| 63 | | 32 | PA.14 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM2 | O | | PWM2: PWM 输出脚 |
| 64 | | 33 | PA.13 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM1 | O | | PWM1: PWM 输出脚 |
| 65 | | 34 | PA.12 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM0 | O | | PWM0: PWM输出脚 |
| 66 | 39 | 35 | ICE_DAT | I/O | | 调试器的串行数据脚 |
| 67 | 40 | 36 | ICE_CLK | I | | 调试器的串行时钟脚 |
| 68 | 41 | | VDD2 | P | | 电源（Power supply for I/O ports and LDO source for internal PLL and digital circuit） |
| 69 | 42 | | VSS2 | P | | 地 |
| 70 | 43 | | AVSS | AP | | 模拟地 |
| 71 | 44 | | PA.0 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC0 | AI | | ADC0: ADC 模拟输入 |
| 72 | 45 | | PA.1 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC1 | AI | | ADC1: ADC模拟输入 |
| 73 | 46 | | PA.2 | I/O | | 标准数字输入/输出管脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| | | | ADC2 | **AI** | | ADC2: ADC模拟输入 |
| 74 | 47 | | PA.3 | **I/O** | | 标准数字输入/输出管脚 |
| | | | ADC3 | **AI** | | ADC3: ADC模拟输入 |
| 75 | 48 | | PA.4 | **I/O** | | 标准数字输入/输出管脚 |
| | | | ADC4 | **AI** | | ADC4: ADC模拟输入 |
| 76 | 49 | | PA.5 | **I/O** | | 标准数字输入/输出管脚 |
| | | | ADC5 | **AI** | | ADC5: ADC模拟输入 |
| 77 | 50 | | PA.6 | **I/O** | | 标准数字输入/输出管脚 |
| | | | ADC6 | **AI** | | ADC6: ADC 模拟输入 |
| 78 | 51 | | PA.7 | **I/O** | | 标准数字输入/输出管脚 |
| | | | ADC7 | **AI** | | ADC7: ADC 模拟输入 |
| | | | /SPISS21 | **I/O** | | /SPISS21: SPI2 2nd 从机选择脚 |
| 79 | | | Vref | **AP** | | ADC参考电压 |
| 80 | 52 | 37 | AVDD | **AP** | | 内部模拟电路的电源 |
| 81 | 53 | 38 | PD.0 | **I/O** | | 标准数字输入/输出管脚 |
| | | | /SPISS20 | **I/O** | | /SPISS20: SPI2 从机选择脚 |
| 82 | 54 | 39 | PD.1 | **I/O** | | 标准数字输入/输出管脚 |
| | | | SPICLK2 | **I/O** | | SPICLK2: SPI2 串行时钟脚 |
| 83 | 55 | 40 | PD.2 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MISO20 | **I** | | MISO20: SPI2 MISO(主机输入，从机输出) pin |
| 84 | 56 | 41 | PD.3 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MOSI20 | **O** | | MOSI20: SPI2 MOSI(主机输出，从机输入) pin |
| 85 | 57 | 42 | PD.4 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MISO21 | **I** | | MISO21: SPI2 2nd MISO(主机输入，从机输出) pin |
| 86 | 58 | 43 | PD.5 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MOSI21 | **O** | | MOSI21: SPI2 2nd MOSI(主机输出，从机输入) pin |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| 87 | | | PC.7 | **I/O** | | 标准数字输入/输出管脚 |
| | | | CPN0 | **I** | | CPN0: Comparator0 负端输入脚 |
| 88 | | | PC.6 | **I/O** | | 标准数字输入/输出管脚 |
| | | | CPP0 | **I** | | CPP0: Comparator0正端输入脚 |
| 89 | | | PC.15 | **I/O** | | 标准数字输入/输出管脚 |
| | | | CPN1 | **I** | | CPN1: Comparator1 负端输入脚 |
| 90 | | | PC.14 | **I/O** | | 标准数字输入/输出管脚 |
| | | | CPP1 | **I** | | CPP1: Comparator1 正端输入脚 |
| 91 | | | PB.15 | **I/O** | | 标准数字输入/输出管脚 |
| | | | /INT1 | **I** | | /INT1: 外部中断0输入脚 |
| 92 | 59 | 44 | XT1_OUT | **O** | | Crystal 输出脚 |
| 93 | 60 | 45 | XT1_IN | **I** | | Crystal 输入脚 |
| 94 | 61 | 46 | /RESET | **I** | | 外部复位输入：低有效，置低时，复位MCU至初始态。带内部上拉. |
| 95 | | | VSS1 | **P** | | 地 |
| 96 | | | VDD1 | **P** | | 电源（Power supply for I/O ports and LDO source for internal PLL and digital circuit） |
| 97 | 62 | 47 | PS2DAT | **I/O** | | PS2 数据脚 |
| 98 | 63 | 48 | PS2CLK | **I/O** | | PS2 时钟脚 |
| 99 | 64 | 1 | PVSS | **I/O** | | PLL 地 |
| 100 | | | PB.8 | **I/O** | | 标准数字输入/输出管脚 |
| | | | STADC | **I** | | STADC: ADC 外部触发输入. |
| | | | TM0 | **O** | | TM0: Timer0 外部计数器输入 |

注:

1. 管脚类型 I= 数字输入, O=数字输出; AI=模拟输入; P=数字电源管脚; AP=模拟电源管教

### 3.3.2 NUC100 管脚定义

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| 1 | | | PE.15 | I/O | | 标准数字输入/输出管脚 |
| 2 | | | PE.14 | I/O | | 标准数字输入/输出管脚 |
| 3 | | | PE.13 | I/O | | 标准数字输入/输出管脚 |
| 4 | 1 | | PB.14 | I/O | | 标准数字输入/输出管脚 |
| | | | /INT0 | I | | /INT0: 外部中断1输入脚 |
| | | | /SPISS31 | I/O | | /SPISS31: SPI3 2nd 从机选择脚 |
| 5 | 2 | | PB.13 | I/O | | 标准数字输入/输出管脚 |
| | | | CPO1 | O | | 比较器1 输出脚 |
| 6 | 3 | 1 | PB.12 | I/O | | 标准数字输入/输出管脚 |
| | | | CPO0 | O | | 比较器0 输出脚 |
| | | | CLKO | O | | 分频器输出脚 |
| 7 | 4 | 2 | X32I | I | | 32.768KHz crystal输入脚 |
| 8 | 5 | 3 | X32O | O | | 32.768KHz crystal 输出脚 |
| 9 | 6 | 4 | PA.11 | I/O | | 标准数字输入/输出管脚 |
| | | | I2C1SCL | I/O | | I2C1SCL: I2C1时钟脚 |
| 10 | 7 | 5 | PA.10 | I/O | | 标准数字输入/输出管脚 |
| | | | I2C1SDA | I/O | | I2C1SDA: I2C1 数据输入/输出脚 |
| 11 | 8 | 6 | PA.9 | I/O | | 标准数字输入/输出管脚 |
| | | | I2C0SCL | I/O | | I2C0SCL: I2C0 时钟脚 |
| 12 | 9 | 7 | PA.8 | I/O | | 标准数字输入/输出管脚 |
| | | | I2C0SDA | I/O | | I2C0SDA: I2C0数据输入/输出脚 |
| 13 | | | PD.8 | I/O | | 标准数字输入/输出管脚 |
| | | | /SPISS30 | I/O | | /SPISS30: SPI3 从机选择脚 |
| 14 | | | PD.9 | I/O | | 标准数字输入/输出管脚 |
| | | | SPICLK3 | I/O | | SPICLK3: SPI3 串行时钟脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| 15 | | | PD.10 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO30 | I | | MISO30: SPI3 MISO(主机输入，从机输出) pin |
| 16 | | | PD.11 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI30 | O | | MOSI30: SPI3 MOSI(主机输出，从机输入) pin |
| 17 | | | PD.12 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO31 | I | | MISO31: SPI3 2nd MISO(主机输入，从机输出) pin |
| 18 | | | PD.13 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI31 | O | | MOSI31: SPI3 2nd MOSI(主机输出，从机输入) pin |
| 19 | 10 | 8 | PB.4 | I/O | | 标准数字输入/输出管脚 |
| | | | RX1 | I | | RX1: UART1数据接收输入脚 |
| 20 | 11 | 9 | PB.5 | I/O | | 标准数字输入/输出管脚 |
| | | | TX1 | O | | TX1: UART1数据发送输出脚 |
| 21 | 12 | | PB.6 | I/O | | 标准数字输入/输出管脚 |
| | | | RTS1 | | | RTS1: UART1请求发送输出脚 |
| 22 | 13 | | PB.7 | I/O | | 标准数字输入/输出管脚 |
| | | | CTS1 | | | CTS1: UART1清发送输入脚 |
| 23 | 14 | 10 | LDO | P | | LDO 输出脚 |
| 24 | 15 | 11 | VDD | P | | 电源（Power supply for I/O ports and LDO source for internal PLL and digital function） |
| 25 | 16 | 12 | VSS | P | | 地 |
| 26 | | | PE.12 | I/O | | 标准数字输入/输出管脚 |
| 27 | | | PE.11 | I/O | | 标准数字输入/输出管脚 |
| 28 | | | PE.10 | I/O | | 标准数字输入/输出管脚 |
| 29 | | | PE.9 | I/O | | 标准数字输入/输出管脚 |
| 30 | | | PE.8 | I/O | | 标准数字输入/输出管脚 |
| 31 | | | PE.7 | I/O | | 标准数字输入/输出管脚 |
| 32 | 17 | 13 | PB.0 | I/O | | 标准数字输入/输出管脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| | | | RX0 | I | | RX0: UART0数据接收输入脚 |
| 33 | 18 | 14 | PB.1 | I/O | | 标准数字输入/输出管脚 |
| | | | TX0 | O | | TX0: UART0数据发送输出脚 |
| 34 | 19 | 15 | PB.2 | I/O | | 标准数字输入/输出管脚 |
| | | | RTS0 | | | RTS0: UART0请求发送输出脚 |
| 35 | 20 | 16 | PB.3 | I/O | | 标准数字输入/输出管脚 |
| | | | CTS0 | | | CTS0: UART0清发送输入脚 |
| 36 | 21 | | PD.6 | I/O | | 标准数字输入/输出管脚 |
| 37 | 22 | | PD.7 | I/O | | 标准数字输入/输出管脚 |
| 38 | 23 | | PD.14 | I/O | | 标准数字输入/输出管脚 |
| 39 | 24 | | PD.15 | I/O | | 标准数字输入/输出管脚 |
| 40 | | | PC.5 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI01 | O | | MOSI01: SPI0 2nd MOSI(主机输出，从机输入) pin |
| 41 | | | PC.4 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO01 | I | | MISO01: SPI0 2nd MISO(主机输入，从机输出) pin |
| 42 | 25 | 17 | PC.3 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI00 | O | | MOSI00: SPI0 MOSI(主机输出，从机输入) pin |
| | | | I2SDO | O | | I2SDO: I2S 数据输出 |
| 43 | 26 | 18 | PC.2 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO00 | I | | MISO00: SPI0 MISO(主机输入，从机输出) pin |
| | | | I2SDI | I | | I2SDI: I2S 数据输入 |
| 44 | 27 | 19 | PC.1 | I/O | | 标准数字输入/输出管脚 |
| | | | SPICLK0 | I/O | | SPICLK0: SPI0 串行时钟脚 |
| | | | I2SBCLK | I/O | | I2SBCLK: I2S bit 时钟脚 |
| 45 | 28 | 20 | PC.0 | I/O | | 标准数字输入/输出管脚 |
| | | | /SPISS00 | I/O | | /SPISS00: SPI0 从机选择脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| | | | I2SLRCLK | I/O | | I2SLRCLK: I2S 左右声道时钟 |
| 46 | | | PE.6 | I/O | | 标准数字输入/输出管脚 |
| 47 | 29 | 21 | PE.5 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM5 | O | | PWM5: PWM 输出 |
| 48 | 30 | 22 | PB.11 | I/O | | 标准数字输入/输出管脚 |
| | | | TM3 | O | | TM3: Timer3 外部计数器输入 |
| | | | PWM4 | O | | PWM4: PWM 输出 |
| 49 | 31 | 23 | PB.10 | I/O | | 标准数字输入/输出管脚 |
| | | | TM2 | O | | TM2: Timer2 外部计数器输入 |
| | | | /SPISS01 | I/O | | /SPISS01: SPI0 2nd 从机选择脚 |
| 50 | 32 | 24 | PB.9 | I/O | | 标准数字输入/输出管脚 |
| | | | TM1 | O | | TM1: Timer1 外部计数器输入 |
| | | | /SPISS11 | I/O | | /SPISS11: SPI1 2nd 从机选择脚 |
| 51 | | | PE.4 | I/O | | 标准数字输入/输出管脚 |
| 52 | | | PE.3 | I/O | | 标准数字输入/输出管脚 |
| 53 | | | PE.2 | I/O | | 标准数字输入/输出管脚 |
| 54 | | | PE.1 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM7 | O | | PWM7: PWM 输出 |
| 55 | | | PE.0 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM6 | O | | PWM6: PWM 输出 |
| 56 | | | PC.13 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI11 | O | | MOSI11: SPI1 2nd MOSI(主机输出，从机输入) pin |
| 57 | | | PC.12 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO11 | I | | MISO11: SPI1 2nd MISO(主机输入，从机输出) pin |
| 58 | 33 | | PC.11 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI10 | O | | MOSI10: SPI1 MOSI(主机输出，从机输入) pin |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| 59 | 34 | | PC.10 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO10 | I | | MISO10: SPI1 MISO(主机输入，从机输出) pin |
| 60 | 35 | | PC.9 | I/O | | 标准数字输入/输出管脚 |
| | | | SPICLK1 | I/O | | SPICLK1: SPI1 串行时钟脚 |
| 61 | 36 | | PC.8 | I/O | | 标准数字输入/输出管脚 |
| | | | /SPISS10 | I/O | | /SPISS10: SPI1 从机选择脚 |
| 62 | 37 | 25 | PA.15 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM3 | O | | PWM3: PWM 输出脚 |
| | | | I2SMCLK | O | | I2SMCLK: I2S 主时钟输出脚 |
| 63 | 38 | 26 | PA.14 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM2 | O | | PWM2: PWM 输出 |
| 64 | 39 | 27 | PA.13 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM1 | O | | PWM1: PWM 输出 |
| 65 | 40 | 28 | PA.12 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM0 | O | | PWM0: PWM 输出 |
| 66 | 41 | 29 | ICE_DAT | I/O | | 调试器的串行数据脚 |
| 67 | 42 | 30 | ICE_CLK | I | | 调试器的串行时钟脚 |
| 68 | | | VDD2 | P | | （电源）Power supply for I/O ports and LDO source for internal PLL and digital circuit |
| 69 | | 31 | VSS2 | P | | 地 |
| 70 | 43 | | AVSS | AP | | 模拟电路地 |
| 71 | 44 | 32 | PA.0 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC0 | AI | | ADC0: ADC 模拟输入 |
| 72 | 45 | 33 | PA.1 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC1 | AI | | ADC1: ADC 模拟输入 |
| 73 | 46 | 34 | PA.2 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC2 | AI | | ADC2: ADC 模拟输入 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| 74 | 47 | 35 | PA.3 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC3 | AI | | ADC3: ADC 模拟输入 |
| 75 | 48 | 36 | PA.4 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC4 | AI | | ADC4: ADC 模拟输入 |
| 76 | 49 | 37 | PA.5 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC5 | AI | | ADC5: ADC 模拟输入 |
| 77 | 50 | 38 | PA.6 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC6 | AI | | ADC6: ADC 模拟输入 |
| 78 | 51 | 39 | PA.7 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC7 | AI | | ADC7: ADC 模拟输入 |
| | | | /SPISS21 | I/O | | /SPISS21: SPI2 2nd 从机选择脚 |
| 79 | | | Vref | AP | | ADC参考电压输入 |
| 80 | 52 | 40 | AVDD | AP | | 内部模拟电路电源 |
| 81 | | | PD.0 | I/O | | 标准数字输入/输出管脚 |
| | | | /SPISS20 | I/O | | /SPISS20: SPI2 从机选择脚 |
| 82 | | | PD.1 | I/O | | 标准数字输入/输出管脚 |
| | | | SPICLK2 | I/O | | SPICLK2: SPI2串行时钟脚 |
| 83 | | | PD.2 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO20 | I | | MISO20: SPI2 MISO(主机输入，从机输出) pin |
| 84 | | | PD.3 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI20 | O | | MOSI20: SPI2 MOSI(主机输出，从机输入) pin |
| 85 | | | PD.4 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO21 | I | | MISO21: SPI2 2nd MISO(主机输入，从机输出) pin |
| 86 | | | PD.5 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI21 | O | | MOSI21: SPI2 2nd MOSI(主机输出，从机输入) pin |
| 87 | 53 | 41 | PC.7 | I/O | | 标准数字输入/输出管脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| | | | CPN0 | I | | CPN0: Comparator0 负端输入脚 |
| 88 | 54 | 42 | PC.6 | I/O | | 标准数字输入/输出管脚 |
| | | | CPP0 | I | | CPP0: Comparator0 正端输入脚 |
| 89 | 55 | | PC.15 | I/O | | 标准数字输入/输出管脚 |
| | | | CPN1 | I | | CPN1: Comparator1 负端输入脚 |
| 90 | 56 | | PC.14 | I/O | | 标准数字输入/输出管脚 |
| | | | CPP1 | I | | CPP1: Comparator1正端输入脚 |
| 91 | 57 | 43 | PB.15 | I/O | | 标准数字输入/输出管脚 |
| | | | /INT1 | I | | /INT1: 外部中断0输入脚 |
| 92 | 58 | 44 | XT1_OUT | O | | Crystal 输出脚 |
| 93 | 59 | 45 | XT1_IN | I | | Crystal 输入脚 |
| 94 | 60 | 46 | /RESET | I | | 外部复位输入：低有效，置低时复位MCU至初始状态，带内部上拉. |
| 95 | 61 | | VSS1 | P | | 地 |
| 96 | 62 | | VDD1 | P | | 电源（Power supply for I/O ports and LDO source for internal PLL and digital circuit） |
| 97 | | | PS2DAT | I/O | | PS2 数据脚 |
| 98 | | | PS2CLK | I/O | | PS2 时钟脚 |
| 99 | 63 | 47 | PVSS | I/O | | PLL 地 |
| 100 | 64 | 48 | PB.8 | I/O | | 标准数字输入/输出管脚 |
| | | | STADC | I | | STADC: ADC外部触发输入 |
| | | | TM0 | O | | TM0: Timer0 外部计数器输入 |

注:

1. 管脚类型 I= 数字输入, O=数字输出; AI=模拟输入; P=数字电源管脚; AP=模拟电源管教

### 3.3.3 NUC120 Pin Description

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| 1 | | | PE.15 | I/O | | 标准数字输入/输出管脚 |
| 2 | | | PE.14 | I/O | | 标准数字输入/输出管脚 |
| 3 | | | PE.13 | I/O | | 标准数字输入/输出管脚 |
| 4 | 1 | | PB.14 | I/O | | 标准数字输入/输出管脚 |
| | | | /INT0 | I | | /INT0:外部中断1输入脚 |
| | | | /SPISS31 | I/O | | /SPISS31: SPI3 2nd 从机选择脚 |
| 5 | 2 | | PB.13 | I/O | | 标准数字输入/输出管脚 |
| | | | CPO1 | O | | 比较器1输出脚 |
| 6 | 3 | 1 | PB.12 | I/O | | 标准数字输入/输出管脚 |
| | | | CPO0 | O | | 比较器0输出脚 |
| | | | CLKO | O | | 分频器输出脚 |
| 7 | 4 | 2 | X32I | I | | 32.768KHz crystal 输入脚 |
| 8 | 5 | 3 | X32O | O | | 32.768KHz crystal 输出脚 |
| 9 | 6 | 4 | PA.11 | I/O | | 标准数字输入/输出管脚 |
| | | | I2C1SCL | I/O | | I2C1SCL: I2C1 时钟脚 |
| 10 | 7 | 5 | PA.10 | I/O | | 标准数字输入/输出管脚 |
| | | | I2C1SDA | I/O | | I2C1SDA: I2C1 数据输入/输出脚 |
| 11 | 8 | 6 | PA.9 | I/O | | 标准数字输入/输出管脚 |
| | | | I2C0SCL | I/O | | I2C0SCL: I2C0 时钟脚 |
| 12 | 9 | 7 | PA.8 | I/O | | 标准数字输入/输出管脚 |
| | | | I2C0SDA | I/O | | I2C0SDA: I2C0 时钟输入/输出脚 |
| 13 | | | PD.8 | I/O | | 标准数字输入/输出管脚 |
| | | | /SPISS30 | I/O | | /SPISS30: SPI3 从机选择脚 |
| 14 | | | PD.9 | I/O | | 标准数字输入/输出管脚 |
| | | | SPICLK3 | I/O | | SPICLK3: SPI3 串行时钟脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| 15 | | | PD.10 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MISO30 | **I** | | MISO30: SPI3 MISO(主机输入，从机输出) pin |
| 16 | | | PD.11 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MOSI30 | **O** | | MOSI30: SPI3 MOSI(主机输出，从机输入) pin |
| 17 | | | PD.12 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MISO31 | **I** | | MISO31: SPI3 2nd MISO(主机输入，从机输出) pin |
| 18 | | | PD.13 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MOSI31 | **O** | | MOSI31: SPI3 2nd MOSI(主机输出，从机输入) pin |
| 19 | 10 | 8 | PB.4 | **I/O** | | 标准数字输入/输出管脚 |
| | | | RX1 | **I** | | RX1: UART1数据接收输入脚 |
| 20 | 11 | 9 | PB.5 | **I/O** | | 标准数字输入/输出管脚 |
| | | | TX1 | **O** | | TX1: UART1数据发送输出脚 |
| 21 | 12 | | PB.6 | **I/O** | | 标准数字输入/输出管脚 |
| | | | RTS1 | | | RTS1: UART1请求发送输出脚 |
| 22 | 13 | | PB.7 | **I/O** | | 标准数字输入/输出管脚 |
| | | | CTS1 | | | CTS1: UART1清发送输入脚 |
| 23 | 14 | 10 | LDO | **P** | | LDO 输出脚 |
| 24 | 15 | 11 | VDD | **P** | | 电源（Power supply for I/O ports and LDO source for internal PLL and digital function） |
| 25 | 16 | 12 | VSS | **P** | | 地 |
| 26 | | | PE.8 | **I/O** | | 标准数字输入/输出管脚 |
| 27 | | | PE.7 | **I/O** | | 标准数字输入/输出管脚 |
| 28 | 17 | 13 | VBUS | **USB** | | USB HOST或HUB提供电源脚 |
| 29 | 18 | 14 | VDD33 | **USB** | | 内部电源输出 3.3V标准电位脚 |
| 30 | 19 | 15 | D- | **USB** | | USB 差分信号 D- |
| 31 | 20 | 16 | D+ | **USB** | | USB 差分信号 D+ |
| 32 | 21 | 17 | PB.0 | **I/O** | | 标准数字输入/输出管脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| | | | RX0 | I | | RX0: UART0数据接收输入脚 |
| 33 | 22 | 18 | PB.1 | I/O | | 标准数字输入/输出管脚 |
| | | | TX0 | O | | TX0: UART0数据发送输出脚 |
| 34 | 23 | | PB.2 | I/O | | 标准数字输入/输出管脚 |
| | | | RTS0 | | | RTS0: UART0请求发送输出脚 |
| 35 | 24 | | PB.3 | I/O | | 标准数字输入/输出管脚 |
| | | | CTS0 | | | CTS0: UART0清发送输入脚 |
| 36 | | 19 | PD.6 | I/O | | 标准数字输入/输出管脚 |
| 37 | | 20 | PD.7 | I/O | | 标准数字输入/输出管脚 |
| 38 | | | PD.14 | I/O | | 标准数字输入/输出管脚 |
| 39 | | | PD.15 | I/O | | 标准数字输入/输出管脚 |
| 40 | | | PC.5 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI01 | O | | MOSI01: SPI0 2nd MOSI(主机输出，从机输入) pin |
| 41 | | | PC.4 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO01 | I | | MISO01: SPI0 2nd MISO(主机输入，从机输出) pin |
| 42 | 25 | 21 | PC.3 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI00 | O | | MOSI00: SPI0 MOSI(主机输出，从机输入) pin |
| | | | I2SDO | O | | I2SDO: I2S 数据输出 |
| 43 | 26 | 22 | PC.2 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO00 | I | | MISO00: SPI0 MISO(主机输入，从机输出) pin |
| | | | I2SDI | I | | I2SDI: I2S 数据输入 |
| 44 | 27 | 23 | PC.1 | I/O | | 标准数字输入/输出管脚 |
| | | | SPICLK0 | I/O | | SPICLK0: SPI0 串行时钟脚 |
| | | | I2SBCLK | I/O | | I2SBCLK: I2S bit 时钟脚 |
| 45 | 28 | 24 | PC.0 | I/O | | 标准数字输入/输出管脚 |
| | | | /SPISS00 | I/O | | /SPISS00: SPI0 从机选择脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| | | | I2SLRCLK | I/O | | I2SLRCLK: I2S 左右声道时钟 |
| 46 | | | PE.6 | I/O | | 标准数字输入/输出管脚 |
| 47 | 29 | | PE.5 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM5 | O | | PWM5: PWM 输出 |
| 48 | 30 | | PB.11 | I/O | | 标准数字输入/输出管脚 |
| | | | TM3 | O | | TM3: Timer3 外部计数器输入 |
| | | | PWM4 | O | | PWM4: PWM 输出 |
| 49 | 31 | | PB.10 | I/O | | 标准数字输入/输出管脚 |
| | | | TM2 | O | | TM2: Timer2 外部计数器输入 |
| | | | /SPISS01 | I/O | | /SPISS01: SPI0 2nd 从机选择脚 |
| 50 | 32 | | PB.9 | I/O | | 标准数字输入/输出管脚 |
| | | | TM1 | O | | TM1: Timer1 外部计数器输入 |
| | | | /SPISS11 | I/O | | /SPISS11: SPI1 2nd 从机选择脚 |
| 51 | | | PE.4 | I/O | | 标准数字输入/输出管脚 |
| 52 | | | PE.3 | I/O | | 标准数字输入/输出管脚 |
| 53 | | | PE.2 | I/O | | 标准数字输入/输出管脚 |
| 54 | | | PE.1 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM7 | O | | PWM7: PWM output |
| 55 | | | PE.0 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM6 | O | | PWM6: PWM output |
| 56 | | | PC.13 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI11 | O | | MOSI11: SPI1 2nd MOSI(主机输出，从机输入) pin |
| 57 | | | PC.12 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO11 | I | | MISO11: SPI1 2nd MISO(主机输入，从机输出) pin |
| 58 | 33 | | PC.11 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI10 | O | | MOSI10: SPI1 MOSI(主机输出，从机输入) pin |
| 59 | 34 | | PC.10 | I/O | | 标准数字输入/输出管脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| | | | MISO10 | I | | MISO10: SPI1 MISO(主机输入，从机输出) pin |
| 60 | 35 | | PC.9 | I/O | | 标准数字输入/输出管脚 |
| | | | SPICLK1 | I/O | | SPICLK1: SPI1 串行时钟脚 |
| 61 | 36 | | PC.8 | I/O | | 标准数字输入/输出管脚 |
| | | | /SPISS10 | I/O | | /SPISS10: SPI1 从机选择脚 |
| 62 | 37 | 25 | PA.15 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM3 | O | | PWM3: PWM 输出脚 |
| | | | I2SMCLK | O | | I2SMCLK: I2S 主机时钟输出脚 |
| 63 | 38 | 26 | PA.14 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM2 | O | | PWM2: PWM 输出 |
| 64 | 39 | 27 | PA.13 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM1 | O | | PWM1: PWM 输出 |
| 65 | 40 | 28 | PA.12 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM0 | O | | PWM0: PWM 输出 |
| 66 | 41 | 29 | ICE_DAT | I/O | | 调试器的串行数据脚 |
| 67 | 42 | 30 | ICE_CLK | I | | 调试器的串行时钟脚 |
| 68 | | | VDD2 | P | | 电源（Power supply for I/O ports and LDO source for internal PLL and digital circuit） |
| 69 | | 31 | VSS2 | P | | 地 |
| 70 | 43 | | AVSS | AP | | 模拟电路地脚 |
| 71 | 44 | 32 | PA.0 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC0 | AI | | ADC0: ADC模拟输入 |
| 72 | 45 | 33 | PA.1 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC1 | AI | | ADC1: ADC 模拟输入 |
| 73 | 46 | 34 | PA.2 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC2 | AI | | ADC2: ADC模拟输入 |
| 74 | 47 | 35 | PA.3 | I/O | | 标准数字输入/输出管脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| | | | ADC3 | **AI** | | ADC3: ADC模拟输入 |
| 75 | 48 | 36 | PA.4 | **I/O** | | 标准数字输入/输出管脚 |
| | | | ADC4 | **AI** | | ADC4: ADC模拟输入 |
| 76 | 49 | 37 | PA.5 | **I/O** | | 标准数字输入/输出管脚 |
| | | | ADC5 | **AI** | | ADC5: ADC模拟输入 |
| 77 | 50 | 38 | PA.6 | **I/O** | | 标准数字输入/输出管脚 |
| | | | ADC6 | **AI** | | ADC6: ADC模拟输入 |
| 78 | 51 | 39 | PA.7 | **I/O** | | 标准数字输入/输出管脚 |
| | | | ADC7 | **AI** | | ADC7: ADC模拟输入 |
| | | | /SPISS21 | **I/O** | | /SPISS21: SPI2 2nd 从机选择脚 |
| 79 | | | Vref | **AP** | | ADC参考电压输入 |
| 80 | 52 | 40 | AVDD | **AP** | | 内部模拟电路电源 |
| 81 | | | PD.0 | **I/O** | | 标准数字输入/输出管脚 |
| | | | /SPISS20 | **I/O** | | /SPISS20: SPI2 从机选择脚 |
| 82 | | | PD.1 | **I/O** | | 标准数字输入/输出管脚 |
| | | | SPICLK2 | **I/O** | | SPICLK2: SPI2 串行时钟脚 |
| 83 | | | PD.2 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MISO20 | **I** | | MISO20: SPI2 MISO(主机输入，从机输出) pin |
| 84 | | | PD.3 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MOSI20 | **O** | | MOSI20: SPI2 MOSI(主机输出，从机输入) pin |
| 85 | | | PD.4 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MISO21 | **I** | | MISO21: SPI2 2nd MISO(主机输入，从机输出) pin |
| 86 | | | PD.5 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MOSI21 | **O** | | MOSI21: SPI2 2nd MOSI(主机输出，从机输入) pin |
| 87 | 53 | 41 | PC.7 | **I/O** | | 标准数字输入/输出管脚 |
| | | | CPN0 | **I** | | CPN0: Comparator0 负端输入脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| 88 | 54 | 42 | PC.6 | I/O | | 标准数字输入/输出管脚 |
| | | | CPP0 | I | | CPP0: Comparator0 正端输入脚 |
| 89 | 55 | | PC.15 | I/O | | 标准数字输入/输出管脚 |
| | | | CPN1 | I | | CPN1: Comparator1 负端输入脚 |
| 90 | 56 | | PC.14 | I/O | | 标准数字输入/输出管脚 |
| | | | CPP1 | I | | CPP1: Comparator1 正端输入脚 |
| 91 | 57 | 43 | PB.15 | I/O | | 标准数字输入/输出管脚 |
| | | | /INT1 | I | | /INT1: 外部 中断0输入脚 |
| 92 | 58 | 44 | XT1_OUT | O | | Crystal 输出脚 |
| 93 | 59 | 45 | XT1_IN | I | | Crystal 输入脚 |
| 94 | 60 | 46 | /RESET | I | | 外部复位输入：低有效，置低时复位MCU为初始状态，带内部上拉. |
| 95 | 61 | | VSS1 | P | | 地 |
| 96 | 62 | | VDD1 | P | | 电源（Power supply for I/O ports and LDO source for internal PLL and digital circuit） |
| 97 | | | PS2DAT | I/O | | PS2 数据脚 |
| 98 | | | PS2CLK | I/O | | PS2 时钟脚 |
| 99 | 63 | 47 | PVSS | I/O | | PLL 地 |
| 100 | 64 | 48 | PB.8 | I/O | | 标准数字输入/输出管脚 |
| | | | STADC | I | | STADC: ADC 外部触发输入. |
| | | | TM0 | O | | TM0: Timer0 外部计数器输入 |

注:

1. 管脚类型 I= 数字输入, O=数字输出; AI=模拟输入; P=数字电源管脚; AP=模拟电源管教

### 3.3.4 NUC130 管脚定义

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| 1 | | | PE.15 | I/O | | 标准数字输入/输出管脚 |
| 2 | | | PE.14 | I/O | | 标准数字输入/输出管脚 |
| 3 | | | PE.13 | I/O | | 标准数字输入/输出管脚 |
| 4 | 1 | | PB.14 | I/O | | 标准数字输入/输出管脚 |
| | | | /INT0 | I | | /INT0: 外部中断1输入脚 |
| | | | /SPISS31 | I/O | | /SPISS31: SPI3 2nd 从机选择脚 |
| 5 | 2 | | PB.13 | I/O | | 标准数字输入/输出管脚 |
| | | | CPO1 | O | | Comparator1 输出 |
| 6 | 3 | 1 | PB.12 | I/O | | 标准数字输入/输出管脚 |
| | | | CPO0 | O | | Comparator0 输出脚 |
| | | | CLKO | O | | 分频器输出脚 |
| 7 | 4 | 2 | X32I | I | | 32.768KHz crystal 输入脚 |
| 8 | 5 | 3 | X32O | O | | 32.768KHz crystal 输出脚 |
| 9 | 6 | 4 | PA.11 | I/O | | 标准数字输入/输出管脚 |
| | | | I2C1SCL | I/O | | I2C1SCL: I2C1 时钟脚 |
| 10 | 7 | 5 | PA.10 | I/O | | 标准数字输入/输出管脚 |
| | | | I2C1SDA | I/O | | I2C1SDA: I2C1 数据输入/输出脚 |
| 11 | 8 | 6 | PA.9 | I/O | | 标准数字输入/输出管脚 |
| | | | I2C0SCL | I/O | | I2C0SCL: I2C0 时钟脚 |
| 12 | 9 | 7 | PA.8 | I/O | | 标准数字输入/输出管脚 |
| | | | I2C0SDA | I/O | | I2C0SDA: I2C0 数据输入/输出脚 |
| 13 | | | PD.8 | I/O | | 标准数字输入/输出管脚 |
| | | | /SPISS30 | I/O | | /SPISS30: SPI3 从机选择脚 |
| 14 | | | PD.9 | I/O | | 标准数字输入/输出管脚 |
| | | | SPICLK3 | I/O | | SPICLK3: SPI3 串行时钟脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| 15 | | | PD.10 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO30 | I | | MISO30: SPI3 MISO(主机输入，从机输出) pin |
| 16 | | | PD.11 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI30 | O | | MOSI30: SPI3 MOSI(主机输出，从机输入) pin |
| 17 | | | PD.12 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO31 | I | | MISO31: SPI3 2nd MISO(主机输入，从机输出) pin |
| 18 | | | PD.13 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI31 | O | | MOSI31: SPI3 2nd MOSI(主机输出，从机输入) pin |
| 19 | 10 | 8 | PB.4 | I/O | | 标准数字输入/输出管脚 |
| | | | RX1 | I | | RX1: UART1数据接收输入脚 |
| 20 | 11 | 9 | PB.5 | I/O | | 标准数字输入/输出管脚 |
| | | | TX1 | O | | TX1: UART1数据发送输出脚 |
| 21 | 12 | | PB.6 | I/O | | 标准数字输入/输出管脚 |
| | | | RTS1 | | | RTS1: UART1请求发送输出脚 |
| 22 | 13 | | PB.7 | I/O | | 标准数字输入/输出管脚 |
| | | | CTS1 | | | CTS1: UART1清发送输入脚 |
| 23 | 14 | 10 | LDO | P | | LDO 输出脚 |
| 24 | 15 | 11 | VDD | P | | 电源（Power supply for I/O ports and LDO source for internal PLL and digital function） |
| 25 | 16 | 12 | VSS | P | | 地 |
| 26 | | | PE.12 | I/O | | 标准数字输入/输出管脚 |
| 27 | | | PE.11 | I/O | | 标准数字输入/输出管脚 |
| 28 | | | PE.10 | I/O | | 标准数字输入/输出管脚 |
| 29 | | | PE.9 | I/O | | 标准数字输入/输出管脚 |
| 30 | | | PE.8 | I/O | | 标准数字输入/输出管脚 |
| 31 | | | PE.7 | I/O | | 标准数字输入/输出管脚 |
| 32 | 17 | 13 | PB.0 | I/O | | 标准数字输入/输出管脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| | | | RX0 | I | | RX0: UART0数据接收输入脚 |
| 33 | 18 | 14 | PB.1 | I/O | | 标准数字输入/输出管脚 |
| | | | TX0 | O | | TX0: UART0数据发送输出脚 |
| 34 | 19 | 15 | PB.2 | I/O | | 标准数字输入/输出管脚 |
| | | | RTS0 | | | RTS0: UART0请求发送输出脚 |
| 35 | 20 | 16 | PB.3 | I/O | | 标准数字输入/输出管脚 |
| | | | CTS0 | | | CTS0: UART0清发送输入脚 |
| 36 | 21 | 17 | PD.6 | I/O | | 标准数字输入/输出管脚 |
| | | | CANRX0 | I | | CAN Bus0 RX 输入 |
| 37 | 22 | 18 | PD.7 | I/O | | 标准数字输入/输出管脚 |
| | | | CANTX0 | O | | CAN Bus0 TX 输出 |
| 38 | 23 | 19 | PD.14 | I/O | | 标准数字输入/输出管脚 |
| | | | CANRX1 | I | | CAN Bus1 RX 输入 |
| 39 | 24 | 20 | PD.15 | I/O | | 标准数字输入/输出管脚 |
| | | | CANTX1 | O | | CAN Bus1 TX 输出 |
| 40 | | | PC.5 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI01 | O | | MOSI01: SPI0 2nd MOSI(主机输出，从机输入) pin |
| 41 | | | PC.4 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO01 | I | | MISO01: SPI0 2nd MISO(主机输入，从机输出) pin |
| 42 | 25 | 21 | PC.3 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI00 | O | | MOSI00: SPI0 MOSI(主机输出，从机输入) pin |
| | | | I2SDO | O | | I2SDO: I2S 数据输出 |
| 43 | 26 | 22 | PC.2 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO00 | I | | MISO00: SPI0 MISO(主机输入，从机输出) pin |
| | | | I2SDI | I | | I2SDI: I2S 数据输入 |
| 44 | 27 | 23 | PC.1 | I/O | | 标准数字输入/输出管脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| | | | SPICLK0 | I/O | | SPICLK0: SPI0 串行时钟脚 |
| | | | I2SBCLK | I/O | | I2SBCLK: I2S bit 时钟脚 |
| 45 | 28 | 24 | PC.0 | I/O | | 标准数字输入/输出管脚 |
| | | | /SPISS00 | I/O | | /SPISS00: SPI0 从机选择脚 |
| | | | I2SLRCLK | I/O | | I2SLRCLK: I2S 左右声道时钟 |
| 46 | | | PE.6 | I/O | | 标准数字输入/输出管脚 |
| 47 | 29 | | PE.5 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM5 | O | | PWM5: PWM 输出 |
| 48 | 30 | | PB.11 | I/O | | 标准数字输入/输出管脚 |
| | | | TM3 | O | | TM3: Timer3 外部计数器输入 |
| | | | PWM4 | O | | PWM4: PWM 输出 |
| 49 | 31 | | PB.10 | I/O | | 标准数字输入/输出管脚 |
| | | | TM2 | O | | TM2: Timer2 外部计数器输入 |
| | | | /SPISS01 | I/O | | /SPISS01: SPI0 2nd 从机选择脚 |
| 50 | 32 | | PB.9 | I/O | | 标准数字输入/输出管脚 |
| | | | TM1 | O | | TM1: Timer1 外部计数器输入 |
| | | | /SPISS11 | I/O | | /SPISS11: SPI1 2nd 从机选择脚 |
| 51 | | | PE.4 | I/O | | 标准数字输入/输出管脚 |
| 52 | | | PE.3 | I/O | | 标准数字输入/输出管脚 |
| 53 | | | PE.2 | I/O | | 标准数字输入/输出管脚 |
| 54 | | | PE.1 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM7 | O | | PWM7: PWM 输出 |
| 55 | | | PE.0 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM6 | O | | PWM6: PWM 输出 |
| 56 | | | PC.13 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI11 | O | | MOSI11: SPI1 2nd MOSI(主机输出，从机输入) pin |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| 57 | | | PC.12 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO11 | I | | MISO11: SPI1 2nd MISO(主机输入，从机输出) pin |
| 58 | 33 | | PC.11 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI10 | O | | MOSI10: SPI1 MOSI(主机输出，从机输入) pin |
| 59 | 34 | | PC.10 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO10 | I | | MISO10: SPI1 MISO(主机输入，从机输出) pin |
| 60 | 35 | | PC.9 | I/O | | 标准数字输入/输出管脚 |
| | | | SPICLK1 | I/O | | SPICLK1: SPI1串行时钟脚 |
| 61 | 36 | | PC.8 | I/O | | 标准数字输入/输出管脚 |
| | | | /SPISS10 | I/O | | /SPISS10: SPI1 从机选择脚 |
| 62 | 37 | 25 | PA.15 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM3 | O | | PWM3: PWM 输出脚 |
| | | | I2SMCLK | O | | I2SMCLK: I2S 主机时钟输出脚 |
| 63 | 38 | 26 | PA.14 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM2 | O | | PWM2: PWM 输出 |
| 64 | 39 | 27 | PA.13 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM1 | O | | PWM1: PWM 输出 |
| 65 | 40 | 28 | PA.12 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM0 | O | | PWM0: PWM 输出 |
| 66 | 41 | 29 | ICE_DAT | I/O | | 调试器的串行数据脚 |
| 67 | 42 | 30 | ICE_CLK | I | | 调试器的串行时钟脚 |
| 68 | | | VDD2 | P | | 电源（Power supply for I/O ports and LDO source for internal PLL and digital circuit） |
| 69 | | 31 | VSS2 | P | | 地 |
| 70 | 43 | | AVSS | AP | | 模拟电路地 |
| 71 | 44 | 32 | PA.0 | I/O | | 标准数字输入/输出管脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| | | | ADC0 | **AI** | | ADC0: ADC模拟输入 |
| 72 | 45 | 33 | PA.1 | **I/O** | | 标准数字输入/输出管脚 |
| | | | ADC1 | **AI** | | ADC1: ADC 模拟输入 |
| 73 | 46 | 34 | PA.2 | **I/O** | | 标准数字输入/输出管脚 |
| | | | ADC2 | **AI** | | ADC2: ADC 模拟输入 |
| 74 | 47 | 35 | PA.3 | **I/O** | | 标准数字输入/输出管脚 |
| | | | ADC3 | **AI** | | ADC3: ADC 模拟输入 |
| 75 | 48 | 36 | PA.4 | **I/O** | | 标准数字输入/输出管脚 |
| | | | ADC4 | **AI** | | ADC4: ADC 模拟输入 |
| 76 | 49 | 37 | PA.5 | **I/O** | | 标准数字输入/输出管脚 |
| | | | ADC5 | **AI** | | ADC5: ADC 模拟输入 |
| 77 | 50 | 38 | PA.6 | **I/O** | | 标准数字输入/输出管脚 |
| | | | ADC6 | **AI** | | ADC6: ADC 模拟输入 |
| 78 | 51 | 39 | PA.7 | **I/O** | | 标准数字输入/输出管脚 |
| | | | ADC7 | **AI** | | ADC7: ADC 模拟输入 |
| | | | /SPISS21 | **I/O** | | /SPISS21: SPI2 2nd 从机选择脚 |
| 79 | | | Vref | **AP** | | ADC参考电压输入 |
| 80 | 52 | 40 | AVDD | **AP** | | 内部模拟电路电源 |
| 81 | | | PD.0 | **I/O** | | 标准数字输入/输出管脚 |
| | | | /SPISS20 | **I/O** | | /SPISS20: SPI2 从机选择脚 |
| 82 | | | PD.1 | **I/O** | | 标准数字输入/输出管脚 |
| | | | SPICLK2 | **I/O** | | SPICLK2: SPI2 串行时钟脚 |
| 83 | | | PD.2 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MISO20 | **I** | | MISO20: SPI2 MISO(主机输入，从机输出) pin |
| 84 | | | PD.3 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MOSI20 | **O** | | MOSI20: SPI2 MOSI(主机输出，从机输入) pin |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| 85 | | | PD.4 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO21 | I | | MISO21: SPI2 2nd MISO(主机输入，从机输出) pin |
| 86 | | | PD.5 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI21 | O | | MOSI21: SPI2 2nd MOSI(主机输出，从机输入) pin |
| 87 | 53 | 41 | PC.7 | I/O | | 标准数字输入/输出管脚 |
| | | | CPN0 | I | | CPN0: Comparator0 负端输入 |
| 88 | 54 | 42 | PC.6 | I/O | | 标准数字输入/输出管脚 |
| | | | CPP0 | I | | CPP0: Comparator0 天端输入 |
| 89 | 55 | | PC.15 | I/O | | 标准数字输入/输出管脚 |
| | | | CPN1 | I | | CPN1: Comparator1 负端输入 |
| 90 | 56 | | PC.14 | I/O | | 标准数字输入/输出管脚 |
| | | | CPP1 | I | | CPP1: Comparator1 正端输入 |
| 91 | 57 | 43 | PB.15 | I/O | | 标准数字输入/输出管脚 |
| | | | /INT1 | I | | /INT1: 外部中断0输入脚 |
| 92 | 58 | 44 | XT1_OUT | O | | Crystal 输出脚 |
| 93 | 59 | 45 | XT1_IN | I | | Crystal 输入脚 |
| 94 | 60 | 46 | /RESET | I | | 外部复位输入：低有效，置低复位MCU为初始状态，带内部上拉. |
| 95 | 61 | | VSS1 | P | | 地 |
| 96 | 62 | | VDD1 | P | | 电源（Power supply for I/O ports and LDO source for internal PLL and digital circuit） |
| 97 | | | PS2DAT | I/O | | PS2 数据脚 |
| 98 | | | PS2CLK | I/O | | PS2 时钟脚 |
| 99 | 63 | 47 | PVSS | I/O | | PLL 地 |
| 100 | 64 | 48 | PB.8 | I/O | | 标准数字输入/输出管脚 |
| | | | STADC | I | | STADC: ADC 外部触发输入. |
| | | | TM0 | O | | TM0: Timer0 外部计数器输入 |

注:

1. 管脚类型 I= 数字输入, O=数字输出; AI=模拟输入; P=数字电源管脚; AP=模拟电源管教

### 3.3.5 NUC140 管脚定义

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| 1 | | | PE.15 | I/O | | 标准数字输入/输出管脚 |
| 2 | | | PE.14 | I/O | | 标准数字输入/输出管脚 |
| 3 | | | PE.13 | I/O | | 标准数字输入/输出管脚 |
| 4 | 1 | | PB.14 | I/O | | 标准数字输入/输出管脚 |
| | | | /INT0 | I | | /INT0: 外部中断1输入脚 |
| | | | /SPISS31 | I/O | | /SPISS31: SPI3 2nd 从机选择脚 |
| 5 | 2 | | PB.13 | I/O | | 标准数字输入/输出管脚 |
| | | | CPO1 | O | | Comparator1 输出脚 |
| 6 | 3 | 1 | PB.12 | I/O | | 标准数字输入/输出管脚 |
| | | | CPO0 | O | | Comparator0 输出脚 |
| | | | CLKO | O | | 分频器输出脚 |
| 7 | 4 | 2 | X32I | I | | 32.768KHz crystal输入脚 |
| 8 | 5 | 3 | X32O | O | | 32.768KHz crystal 输出脚 |
| 9 | 6 | 4 | PA.11 | I/O | | 标准数字输入/输出管脚 |
| | | | I2C1SCL | I/O | | I2C1SCL: I2C1 时钟脚 |
| 10 | 7 | 5 | PA.10 | I/O | | 标准数字输入/输出管脚 |
| | | | I2C1SDA | I/O | | I2C1SDA: I2C1 数据输入/输出脚 |
| 11 | 8 | 6 | PA.9 | I/O | | 标准数字输入/输出管脚 |
| | | | I2C0SCL | I/O | | I2C0SCL: I2C0 时钟脚 |
| 12 | 9 | 7 | PA.8 | I/O | | 标准数字输入/输出管脚 |
| | | | I2C0SDA | I/O | | I2C0SDA: I2C0数据输入/输出脚 |
| 13 | | | PD.8 | I/O | | 标准数字输入/输出管脚 |
| | | | /SPISS30 | I/O | | /SPISS30: SPI3 从机选择脚 |
| 14 | | | PD.9 | I/O | | 标准数字输入/输出管脚 |
| | | | SPICLK3 | I/O | | SPICLK3: SPI3 串行时钟脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| 15 | | | PD.10 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MISO30 | **I** | | MISO30: SPI3 MISO(主机输入，从机输出) pin |
| 16 | | | PD.11 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MOSI30 | **O** | | MOSI30: SPI3 MOSI(主机输出，从机输入) pin |
| 17 | | | PD.12 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MISO31 | **I** | | MISO31: SPI3 2nd MISO(主机输入，从机输出) pin |
| 18 | | | PD.13 | **I/O** | | 标准数字输入/输出管脚 |
| | | | MOSI31 | **O** | | MOSI31: SPI3 2nd MOSI(主机输出，从机输入) pin |
| 19 | 10 | 8 | PB.4 | **I/O** | | 标准数字输入/输出管脚 |
| | | | RX1 | **I** | | RX1: UART1数据接收输入脚 |
| 20 | 11 | 9 | PB.5 | **I/O** | | 标准数字输入/输出管脚 |
| | | | TX1 | **O** | | TX1: UART1数据发送输出脚 |
| 21 | 12 | | PB.6 | **I/O** | | 标准数字输入/输出管脚 |
| | | | RTS1 | | | RTS1: UART1请求发送输出脚 |
| 22 | 13 | | PB.7 | **I/O** | | 标准数字输入/输出管脚 |
| | | | CTS1 | | | CTS1: UART1清发送输入脚 |
| 23 | 14 | 10 | LDO | **P** | | LDO 输出脚 |
| 24 | 15 | 11 | VDD | **P** | | 电源（Power supply for I/O ports and LDO source for internal PLL and digital function） |
| 25 | 16 | 12 | VSS | **P** | | 地 |
| 26 | | | PE.8 | **I/O** | | 标准数字输入/输出管脚 |
| 27 | | | PE.7 | **I/O** | | 标准数字输入/输出管脚 |
| 28 | 17 | 13 | VBUS | **USB** | | USB HOST或HUB提供电源脚 |
| 29 | 18 | 14 | VDD33 | **USB** | | 内部电源输出 3.3V标准电位脚 |
| 30 | 19 | 15 | D- | **USB** | | USB 差分信号 D- |
| 31 | 20 | 16 | D+ | **USB** | | USB 差分信号 D+ |
| 32 | 21 | 17 | PB.0 | **I/O** | | 标准数字输入/输出管脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| | | | RX0 | I | | RX0: UART0数据接收输入脚 |
| 33 | 22 | 18 | PB.1 | I/O | | 标准数字输入/输出管脚 |
| | | | TX0 | O | | TX0: UART0数据发送输出脚 |
| 34 | 23 | | PB.2 | I/O | | 标准数字输入/输出管脚 |
| | | | RTS0 | | | RTS0: UART0请求发送输出脚 |
| 35 | 24 | | PB.3 | I/O | | 标准数字输入/输出管脚 |
| | | | CTS0 | | | CTS0: UART0清发送输入脚 |
| 36 | 25 | 19 | PD.6 | I/O | | 标准数字输入/输出管脚 |
| | | | CANRX0 | I | | CAN Bus0 RX 输入 |
| 37 | 26 | 20 | PD.7 | I/O | | 标准数字输入/输出管脚 |
| | | | CANTX0 | O | | CAN Bus0 TX 输出 |
| 38 | 27 | | PD.14 | I/O | | 标准数字输入/输出管脚 |
| | | | CANRX1 | I | | CAN Bus1 RX 输入 |
| 39 | 28 | | PD.15 | I/O | | 标准数字输入/输出管脚 |
| | | | CANTX1 | O | | CAN Bus1 TX 输出 |
| 40 | | | PC.5 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI01 | O | | MOSI01: SPI0 2nd MOSI(主机输出，从机输入) pin |
| 41 | | | PC.4 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO01 | I | | MISO01: SPI0 2nd MISO(主机输入，从机输出) pin |
| 42 | 29 | 21 | PC.3 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI00 | O | | MOSI00: SPI0 MOSI(主机输出，从机输入) pin |
| | | | I2SDO | O | | I2SDO: I2S 数据输出 |
| 43 | 30 | 22 | PC.2 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO00 | I | | MISO00: SPI0 MISO(主机输入，从机输出) pin |
| | | | I2SDI | I | | I2SDI: I2S 数据输入 |
| 44 | 31 | 23 | PC.1 | I/O | | 标准数字输入/输出管脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| | | | SPICLK0 | I/O | | SPICLK0: SPI0 串行时钟脚 |
| | | | I2SBCLK | I/O | | I2SBCLK: I2S bit 时钟脚 |
| 45 | 32 | 24 | PC.0 | I/O | | 标准数字输入/输出管脚 |
| | | | /SPISS00 | I/O | | /SPISS00: SPI0 从机选择脚 |
| | | | I2SLRCLK | I/O | | I2SLRCLK: I2S 左右声道时钟 |
| 46 | | | PE.6 | I/O | | 标准数字输入/输出管脚 |
| 47 | | | PE.5 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM5 | O | | PWM5: PWM 输出 |
| 48 | | | PB.11 | I/O | | 标准数字输入/输出管脚 |
| | | | TM3 | O | | TM3: Timer3 外部计数器输入 |
| | | | PWM4 | O | | PWM4: PWM输出 |
| 49 | | | PB.10 | I/O | | 标准数字输入/输出管脚 |
| | | | TM2 | O | | TM2: Timer2 外部计数输入 |
| | | | /SPISS01 | I/O | | /SPISS01: SPI0 2nd 从机选择脚 |
| 50 | | | PB.9 | I/O | | 标准数字输入/输出管脚 |
| | | | TM1 | O | | TM1: Timer1 外部计数器输入 |
| | | | /SPISS11 | I/O | | /SPISS11: SPI1 2nd 从机选择脚 |
| 51 | | | PE.4 | I/O | | 标准数字输入/输出管脚 |
| 52 | | | PE.3 | I/O | | 标准数字输入/输出管脚 |
| 53 | | | PE.2 | I/O | | 标准数字输入/输出管脚 |
| 54 | | | PE.1 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM7 | O | | PWM7: PWM输出 |
| 55 | | | PE.0 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM6 | O | | PWM6: PWM输出 |
| 56 | | | PC.13 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI11 | O | | MOSI11: SPI1 2nd MOSI(主机输出，从机输入) pin |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| 57 | | | PC.12 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO11 | I | | MISO11: SPI1 2nd MISO(主机输入，从机输出) pin |
| 58 | 33 | | PC.11 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI10 | O | | MOSI10: SPI1 MOSI(主机输出，从机输入) pin |
| 59 | 34 | | PC.10 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO10 | I | | MISO10: SPI1 MISO(主机输入，从机输出) pin |
| 60 | 35 | | PC.9 | I/O | | 标准数字输入/输出管脚 |
| | | | SPICLK1 | I/O | | SPICLK1: SPI1 串行时钟脚 |
| 61 | 36 | | PC.8 | I/O | | 标准数字输入/输出管脚 |
| | | | /SPISS10 | I/O | | /SPISS10: SPI1 从机选择脚 |
| 62 | 37 | 25 | PA.15 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM3 | O | | PWM3: PWM输出 |
| | | | I2SMCLK | O | | I2SMCLK: I2S 主机时钟输出脚 |
| 63 | 38 | 26 | PA.14 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM2 | O | | PWM2: PWM 输出 |
| 64 | 39 | 27 | PA.13 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM1 | O | | PWM1: PWM 输出 |
| 65 | 40 | 28 | PA.12 | I/O | | 标准数字输入/输出管脚 |
| | | | PWM0 | O | | PWM0: PWM 输出 |
| 66 | 41 | 29 | ICE_DAT | I/O | | 调试器的串行数据脚 |
| 67 | 42 | 30 | ICE_CLK | I | | 调试器的串行时钟脚 |
| 68 | | | VDD2 | P | | 电源（Power supply for I/O ports and LDO source for internal PLL and digital circuit） |
| 69 | | 31 | VSS2 | P | | 地 |
| 70 | 43 | | AVSS | AP | | 模拟电路地 |
| 71 | 44 | 32 | PA.0 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC0 | AI | | ADC0: ADC 模拟输入 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| 72 | 45 | 33 | PA.1 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC1 | AI | | ADC1: ADC 模拟输入 |
| 73 | 46 | 34 | PA.2 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC2 | AI | | ADC2: ADC 模拟输入 |
| 74 | 47 | 35 | PA.3 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC3 | AI | | ADC3: ADC 模拟输入 |
| 75 | 48 | 36 | PA.4 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC4 | AI | | ADC4: ADC 模拟输入 |
| 76 | 49 | 37 | PA.5 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC5 | AI | | ADC5: ADC 模拟输入 |
| 77 | 50 | 38 | PA.6 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC6 | AI | | ADC6: ADC 模拟输入 |
| 78 | 51 | 39 | PA.7 | I/O | | 标准数字输入/输出管脚 |
| | | | ADC7 | AI | | ADC7: ADC 模拟输入 |
| | | | /SPISS21 | I/O | | /SPISS21: SPI2 2nd 从机选择脚 |
| 79 | | | Vref | AP | | ADC参考电压输入 |
| 80 | 52 | 40 | AVDD | AP | | 内部模拟电路电源 |
| 81 | | | PD.0 | I/O | | 标准数字输入/输出管脚 |
| | | | /SPISS20 | I/O | | /SPISS20: SPI2 从机选择脚 |
| 82 | | | PD.1 | I/O | | 标准数字输入/输出管脚 |
| | | | SPICLK2 | I/O | | SPICLK2: SPI2 串行时钟脚 |
| 83 | | | PD.2 | I/O | | 标准数字输入/输出管脚 |
| | | | MISO20 | I | | MISO20: SPI2 MISO(主机输入，从机输出) pin |
| 84 | | | PD.3 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI20 | O | | MOSI20: SPI2 MOSI(主机输出，从机输入) pin |
| 85 | | | PD.4 | I/O | | 标准数字输入/输出管脚 |

| 管脚号. | | | 管脚名称 | 管脚类型 | | 描述 |
|---|---|---|---|---|---|---|
| LQFP 100 | LQFP 64 | LQFP 48 | | | | |
| | | | MISO21 | I | | MISO21: SPI2 2nd MISO(主机输入，从机输出) pin |
| 86 | | | PD.5 | I/O | | 标准数字输入/输出管脚 |
| | | | MOSI21 | O | | MOSI21: SPI2 2nd MOSI(主机输出，从机输入) pin |
| 87 | 53 | 41 | PC.7 | I/O | | 标准数字输入/输出管脚 |
| | | | CPN0 | I | | CPN0: Comparator0 负端输入脚 |
| 88 | 54 | 42 | PC.6 | I/O | | 标准数字输入/输出管脚 |
| | | | CPP0 | I | | CPP0: Comparator0 正端输入脚 |
| 89 | 55 | | PC.15 | I/O | | 标准数字输入/输出管脚 |
| | | | CPN1 | I | | CPN1: Comparator1负端输入脚 |
| 90 | 56 | | PC.14 | I/O | | 标准数字输入/输出管脚 |
| | | | CPP1 | I | | CPP1: Comparator1正端输入脚 |
| 91 | 57 | 43 | PB.15 | I/O | | 标准数字输入/输出管脚 |
| | | | /INT1 | I | | /INT1: 外部中断0输入脚 |
| 92 | 58 | 44 | XT1_OUT | O | | Crystal 输出脚 |
| 93 | 59 | 45 | XT1_IN | I | | Crystal 输入脚 |
| 94 | 60 | 46 | /RESET | I | | 外部复位输入：低有效，置低复位MCU为初始状态，带内部上拉 |
| 95 | 61 | | VSS1 | P | | 地 |
| 96 | 62 | | VDD1 | P | | 电源（Power supply for I/O ports and LDO source for internal PLL and digital circuit） |
| 97 | | | PS2DAT | I/O | | PS2 数据脚 |
| 98 | | | PS2CLK | I/O | | PS2 时钟脚 |
| 99 | 63 | 47 | PVSS | I/O | | PLL 地 |
| 100 | 64 | 48 | PB.8 | I/O | | 标准数字输入/输出管脚 |
| | | | STADC | I | | STADC: ADC 外部触发输入. |
| | | | TM0 | O | | TM0: Timer0 外部计数器输入 |

注:

1. 管脚类型 I= 数字输入, O=数字输出; AI=模拟输入; P=数字电源管脚; AP=模拟电源管教

# 4 框图

## 4.1 NUC101 框图

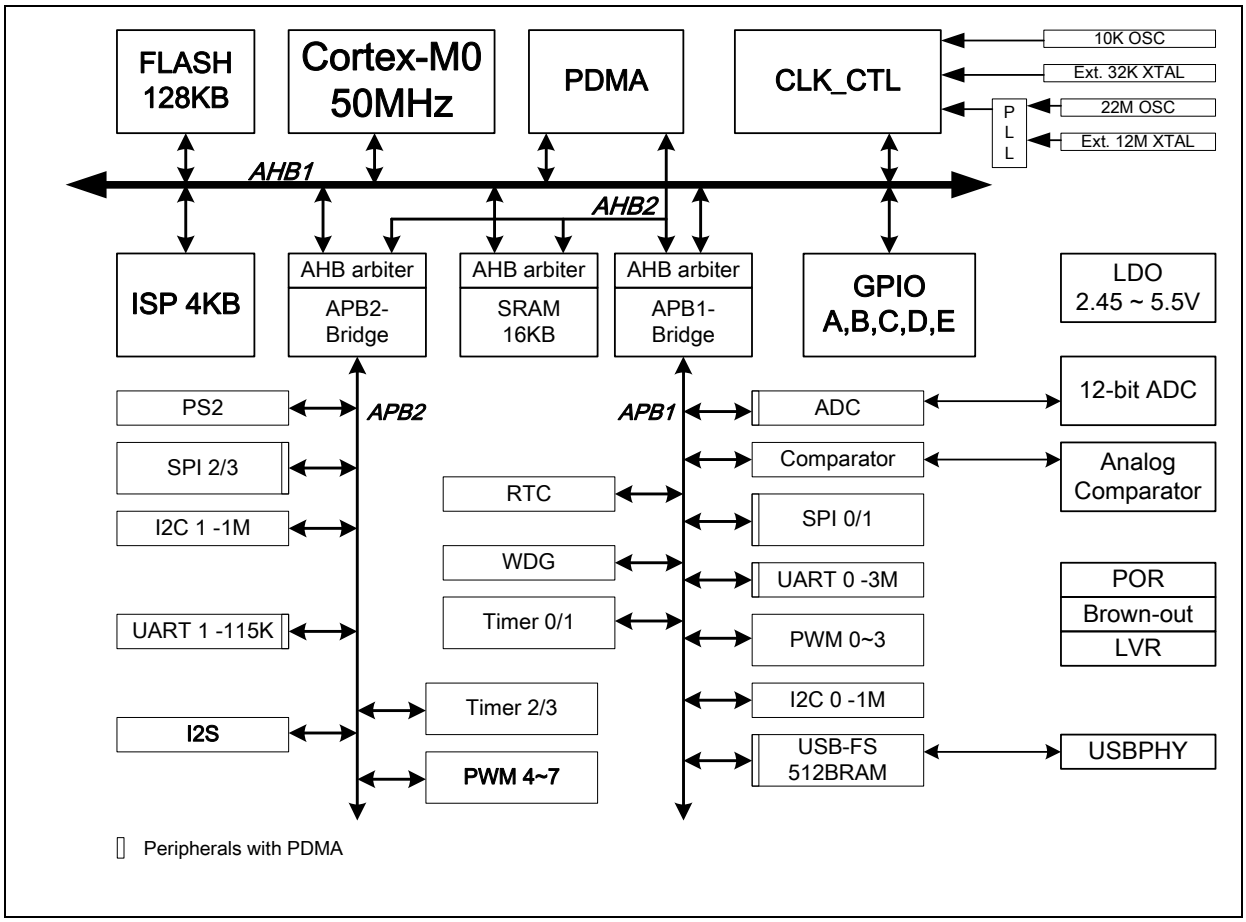



图 4-1 NUC101 框图

## 4.2 NUC100 框图

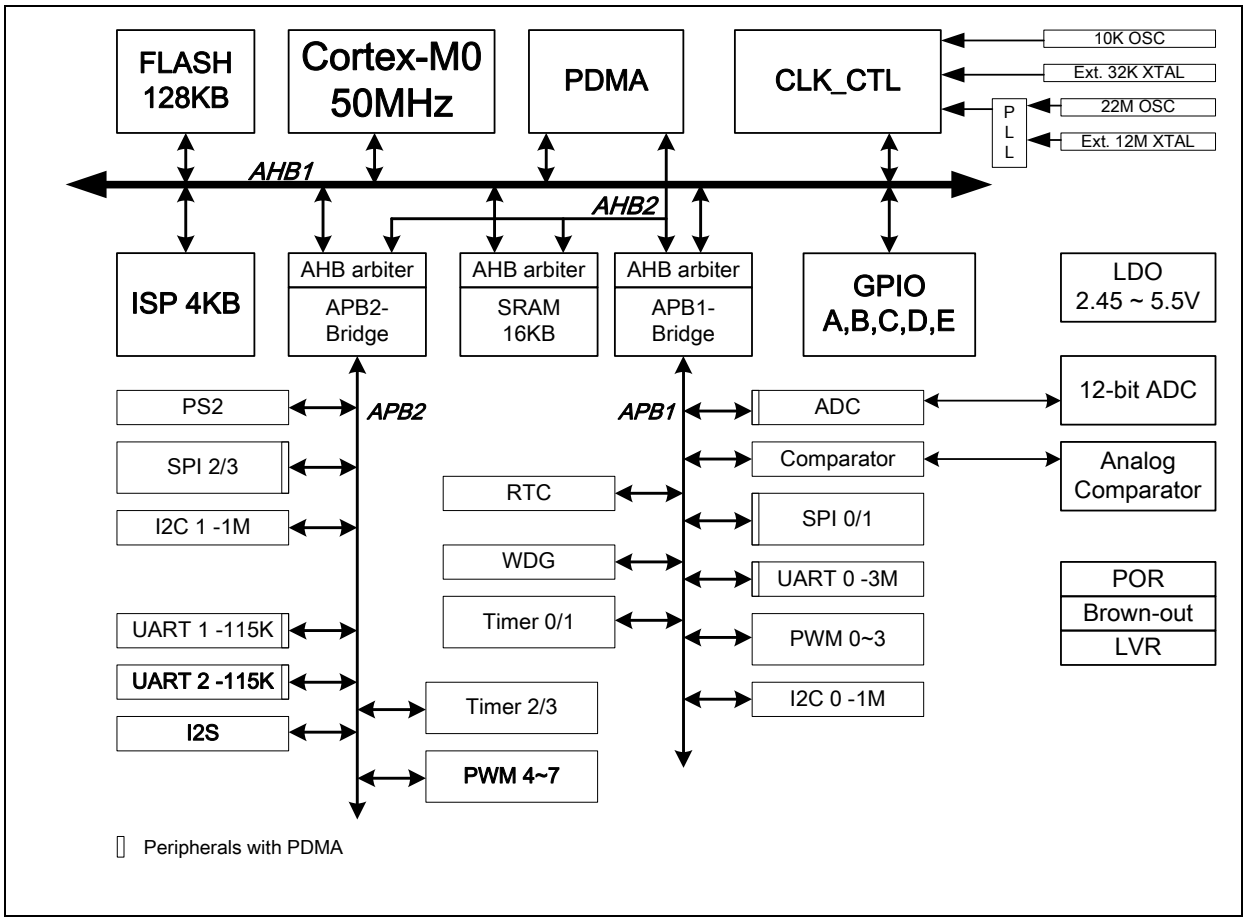



图 4-2 NUC100 框图

## 4.3 NUC120 框图

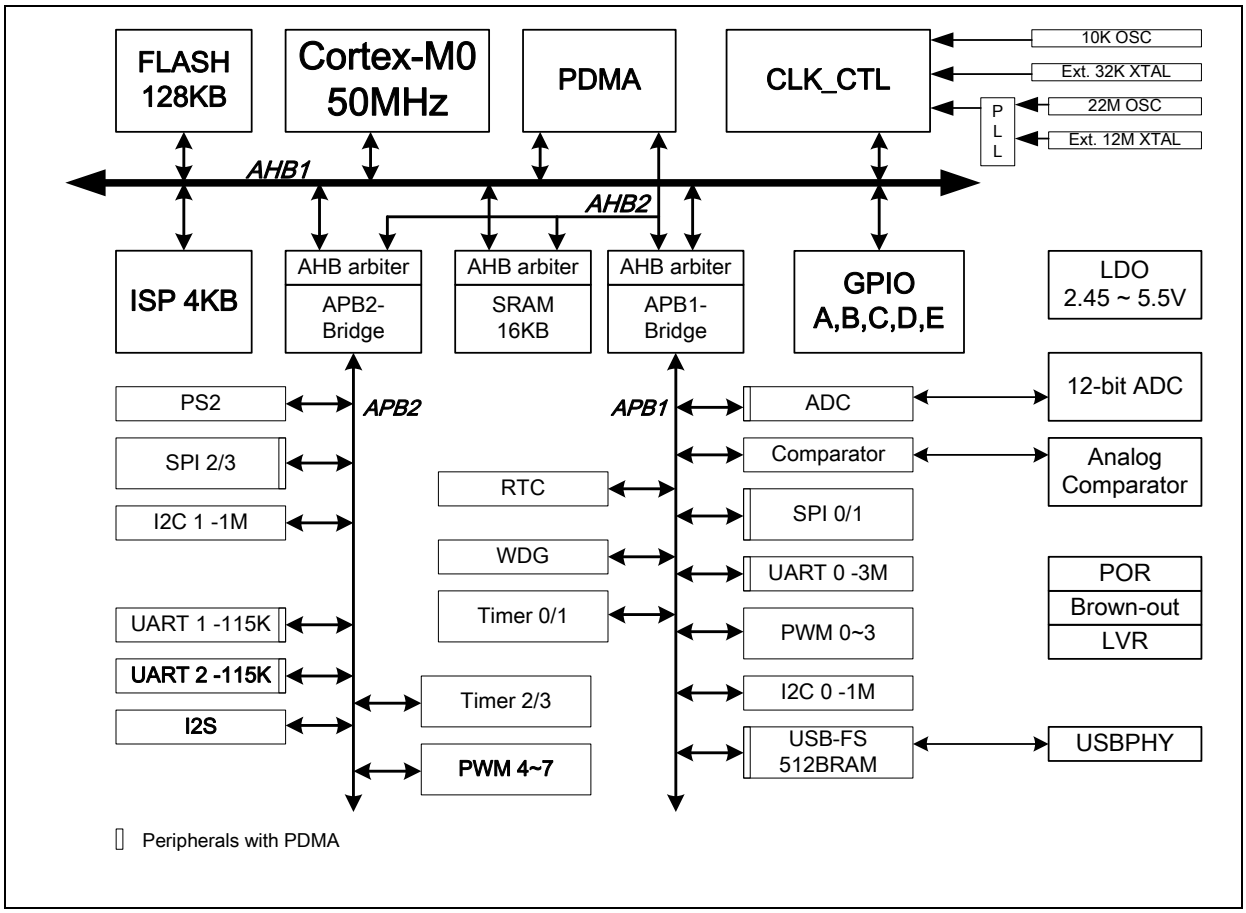



图 4-3 NUC120 框图

## 4.4 NUC130 框图

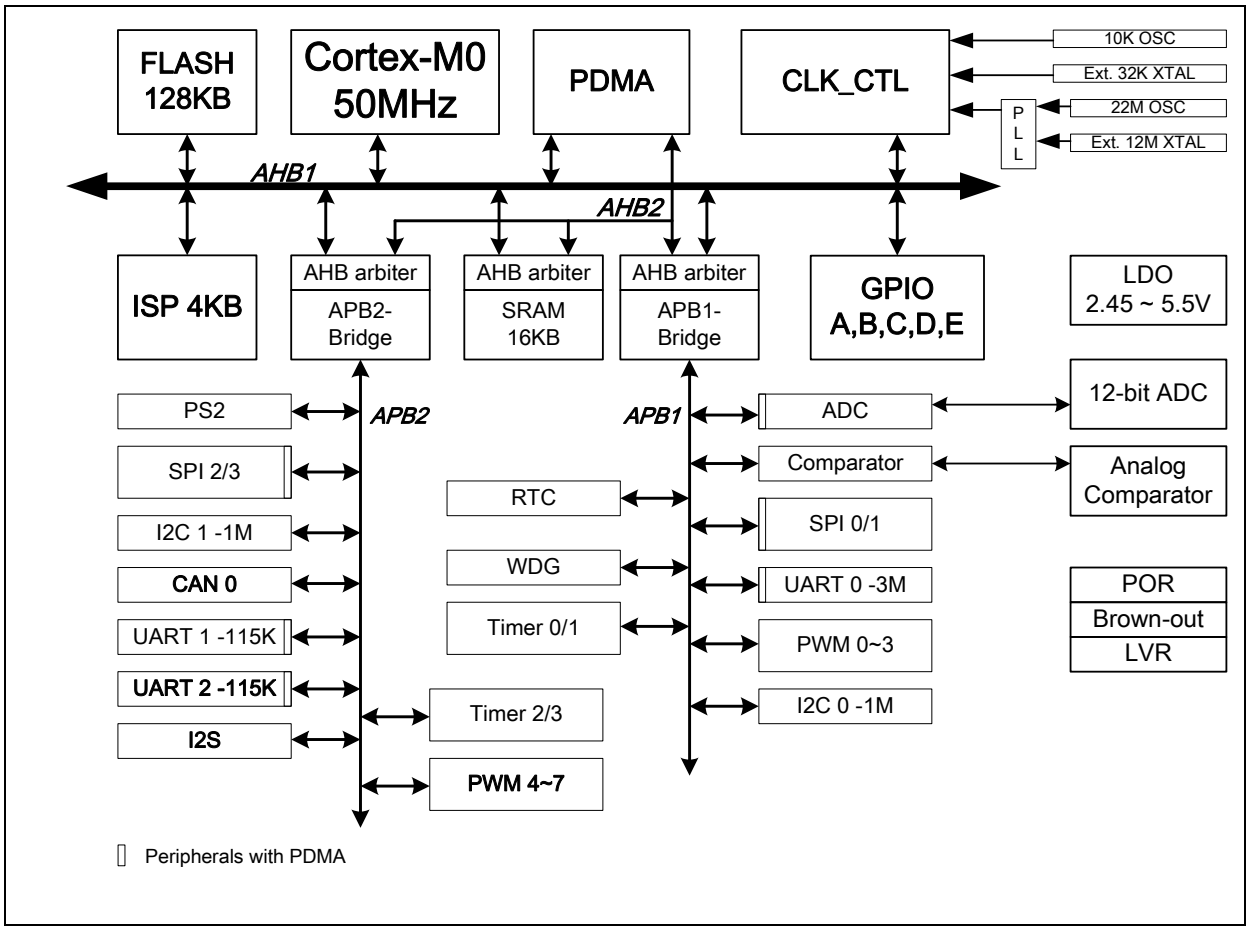



图 4-4 NUC130 框图

## 4.5 NUC140 框图

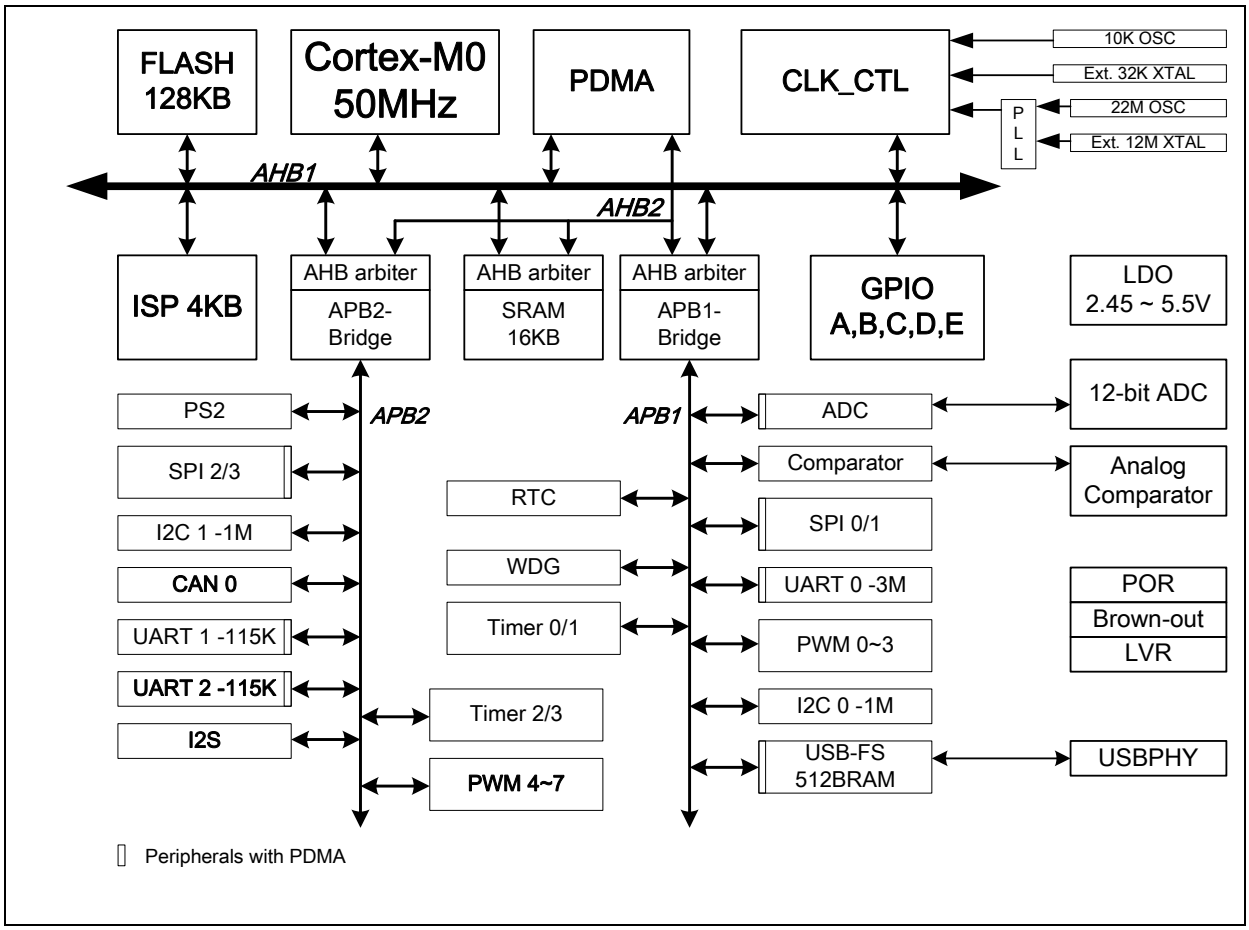



图 4-5 NUC140 框图

# 5 功能描述

## 5.1 ARM® Cortex™-M0 内核

Cortex™-M0处理器是32位多级可配置的RISC处理器。它有AMBA AHB-Lite接口和嵌套向量中断控制器（NVIC），具有可选的硬件调试功能，可以执行Thumb指令，并与其它Cortex-M系列兼容.

图 5-1 处理器的功能框图.

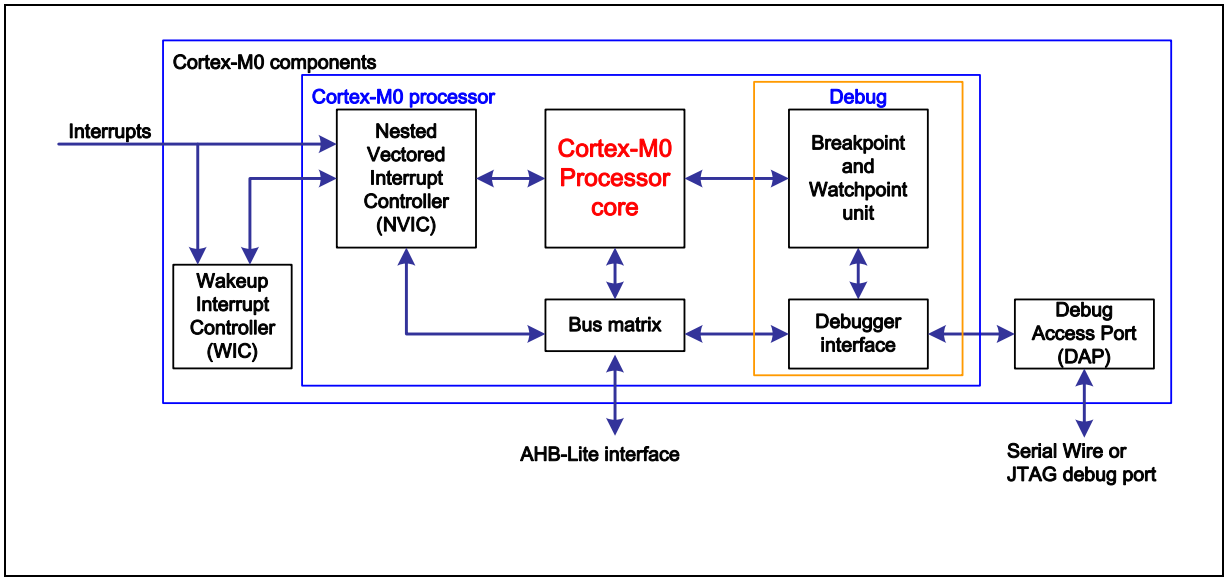



图 5-1 功能框图

设备提供:

- 低门数处理器特征:
  - ARMv6-M Thumb 指令集.
  - Thumb-2 技术.
  - ARMv6-M 兼容 24-bit SysTick 定时器.
  - 32-bit 硬件乘法器.
  - 系统支持小端（little-endian）数据.
  - 准确而及时的中断处理能力.
  - 加载/多路存储和多周期乘法可以实现快速中断处理.
  - C 应用程序二进制接口的异常兼容模式（C-ABI）.
    （C-ABI）完全允许用户使用 C 函数实现中断处理.
  - （WFI）可以进入低功耗的休眠模式，（WFE）指令可由休眠模式退出.
- NVIC 特征:
  - 32 个外部中断，具有 4 级优先级.
  - 不可屏蔽中断输入（NMI）.
  - 支持电平和脉冲中断触发.
  - 中断唤醒控制器(WIC), 支持极低功耗休眠模式..

- 调试
  - 四个硬件断点.
  - 两个观察点.
  - 用于非侵入式代码分析的程序计数采样寄存器.
  - 单步向量捕获能力.
- 总线接口:
  - 单一 32 位的 AMBA-3 ABH-Lite 系统接口.
  - 支持 DAP(Debug Access Port)的单一 32 位的从端口.

## 5.2 系统管理器

### 5.2.1 概述

系统管理器包括如下功能

- 系统内存
- 系统定时器（SysTick）
- 嵌套向量中断控制器（NVIC）
- 产品ID的系统管理寄存器
- 芯片、模块功能复位和多功能管脚控制的系统管理寄存器
- 欠压和复合芯片控制寄存器
- 对外围设备中断源的认别

### 5.2.2 系统复位

下列情况发生时，系统复位，复位标志由寄存器**RSTRC**读出.

- 上电复位
- 复位脚（/RESET）上有低电平
- 看门狗复位
- 低压复位
- 欠压检测复位
- Coretex-M0 单片机复位
- PMU 复位

### 5.2.3 系统内存

NUC1xx 提供4G字节的寻址空间，内存地址分配情况见表5-1，NUC1xx系列仅支持小端数据格式.

表 5-1 片上模块的地址空间分配

| 地址空间 | 标志 | 模块 |
|---|---|---|
| **Flash & SRAM内存空间** | | |
| 0x0000_0000 – 0x0001_FFFF | FLASH_BA | FLASH 内存空间 (128KB) |
| 0x2000_0000 – 0x2000_3FFF | SRAM_BA | SRAM 内存空间 (16KB) |
| **AHB 模块空间 (0x5000_0000 – 0x501F_FFFF)** | | |
| 0x5000_0000 – 0x5000_01FF | GCR_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 | GPIO_BA | GPIO控制寄存器 |
| 0x5000_8000 – 0x5000_BFFF | PDMA_BA | SRAM_APB DMA 控制寄存器 |
| 0x5000_C000 – 0x5000_FFFF | FMC_BA | Flash 内存控制寄存器 |
| **APB1 模块空间 (0x4000_0000 ~ 0x400F_FFFF)** | | |
| 0x4000_0000 – 0x4000_3FFF | INT_BA | 中断控制寄存器 |
| 0x4000_4000 – 0x4000_7FFF | WDT_BA | 看门狗控制寄存器 |
| 0x4000_8000 – 0x4000_BFFF | RTC_BA | RTC控制寄存器 |
| 0x4001_0000 – 0x4001_3FFF | TMR01_BA | Timer0/Timer1 控制寄存器 |
| 0x4002_0000 – 0x4002_3FFF | I2C0_BA | I2C0 接口控制寄存器 |
| 0x4003_0000 – 0x4003_3FFF | SPI0_BA | 带主/从功能的SPI0控制寄存器 |
| 0x4003_4000 – 0x4003_7FFF | SPI2_BA | 带主/从功能的SPI2 with 控制寄存器 |
| 0x4004_0000 – 0x4004_3FFF | PWM_BA0 | PWM0/1/2/3 控制寄存器 |

| | | |
|---|---|---|
| 0x4005_0000 – 0x4005_3FFF | UART0_BA | UART0控制寄存器 |
| 0x4006_0000 – 0x4006_3FFF | USBD_BA | USB FS 设备控制寄存器 |
| 0x400D_0000 – 0x400D_3FFF | ACMP_BA | 模拟比较控制寄存器 |
| 0x400E_0000 – 0x400E_FFFF | ADC0_BA | ADC控制寄存器 |
| **APB2 模块空间 (0x4010_0000 ~ 0x401F_FFFF)** | | |
| 0x4010_0000 – 0x4010_3FFF | PS2_BA | PS2接口控制寄存器 |
| 0x4011_0000 – 0x4011_3FFF | TMR23_BA | Timer2/Timer3控制寄存器 |
| 0x4012_0000 – 0x4012_3FFF | I2C1_BA | I2C1接口控制寄存器 |
| 0x4013_0000 – 0x4013_3FFF | SPI1_BA | 带主/从功能的SPI1控制寄存器 |
| 0x4013_4000 – 0x4013_7FFF | SPI3_BA | 带主/从功能的SPI3控制寄存器 |
| 0x4014_0000 – 0x4014_3FFF | PWM_BA4 | PWM4/5/6/7控制寄存器 |
| 0x4015_0000 – 0x4015_3FFF | UART1_BA | UART1 控制寄存器 |
| 0x4015_4000 – 0x4015_7FFF | UART2_BA | UART2控制寄存器 |
| 0x4018_0000 – 0x4018_3FFF | CAN0_BA | CAN0 Bus控制寄存器 |
| **系统控制空间 (0xE000_E000 ~ 0xE000_EFFF)** | | |
| 0xE000_E010 – 0xE000_E0FF | SCS_BA | System 定时器控制寄存器 |
| 0xE000_E100 – 0xE000_ECFF | SCS_BA | 外部中断控制器控制寄存器 |
| 0xE000_ED00 – 0xE000_ED8F | SCS_BA | System控制寄存器 |

### 5.2.4 系统管理器控制寄存器

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| GCR_BA = 0x5000_0000 | | | | |
| **RSTSRC** | GCR_BA+0x04 | R/W | 系统复位源寄存器 | 0x0000_00XX |
| **IPRSTC1** | GCR_BA+0x08 | R/W | IP复位控制寄存器1 | 0x0000_0000 |
| **IPRSTC2** | GCR_BA+0x0C | R/W | IP复位控制寄存器2 | 0x0000_0000 |
| 保留 | GCR_BA+0x14 | R/W | 保留 | 0x0000_0000 |
| **BODCR** | GCR_BA+0x18 | R/W | 欠压检测控制寄存器 | 0x0000_008x |
| **TEMPCR** | GCR_BA+0x1C | R/W | 温度传感器控制寄存器 | 0x0000_0000 |
| **GPA_MFP** | GCR_BA+0x30 | R/W | 多功能GPIOA 和输入类型控制寄存器 | 0x0000_0000 |
| **GPB_MFP** | GCR_BA+0x34 | R/W | 多功能GPIOB 和输入类型控制寄存器 | 0x0000_0000 |
| **GPC_MFP** | GCR_BA+0x38 | R/W | 多功能GPIOC 和输入类型控制寄存器 | 0x0000_0000 |
| **GPD_MFP** | GCR_BA+0x3C | R/W | 多功能GPIOD 和输入类型控制寄存器 | 0x0000_0000 |
| **GPE_MFP** | GCR_BA+0x40 | R/W | GPIOE输入类型控制寄存器 | 0x0000_0000 |
| **ALT_MFP** | GCR_BA+0x50 | R/W | USPI 多功能管脚控制寄存器 | 0x0000_0000 |
| **RegLockAddr** | GCR_BA+0x100 | R/W | 地址锁存寄存器 | 0x0000_0000 |
| **RCADJ** | GCR_BA+0x110 | R/W | RC 校验值 | 0xxxxx_xxxx |

### 系统复位源寄存器(RSTSRC)

该寄存器提供一些信息用于识别引起芯片上次操作复位的复位源.

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| RSTSRC | GCR_BA+04 | R/W | 系统复位源寄存器 | 0x0000_00xx |

| 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 | RCLR_PMU | RCLR_MCU | RCLR_BOD | RCLR_LVR | RCLR_WDG | RCLR_PAD | RCLR_POR |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | RSTS_PMU | RSTS_MCU | RSTS_BOD | RSTS_LVR | RSTS_WDG | RSTS_PAD | RSTS_POR |

| Bits | 描述 | |
|---|---|---|
| [31:15] | 保留 | 保留 |
| [14] | RCLR_PMU | 0:清相关的标志RSTS_PMU (RSTRC[6])<br>1:无效 |
| [13] | RCLR_MCU | 0:清相关的标志RSTS_MCU (RSTRC[5])<br>1:无效 |
| [12] | RCLR_BOD | 0: 清相关的标志RSTS_BOD (RSTRC[4])<br>1:无效 |
| [11] | RCLR_LVR | 0: 清相关的标志RSTS_LVR (RSTRC[3])<br>1:无效 |
| [10] | RCLR_WDG | 0: 清相关的标志RSTS_WDG (RSTRC[2])<br>1:无效 |

| | | |
|---|---|---|
| **[9]** | **RCLR_PAD** | 0: 清相关的标志RSTS_PAD (RSTRC[1])<br>1:无效 |
| **[8]** | **RCLR_POR** | 0: 清相关的标志RSTS_POR (RSTRC[0])<br>1:无效 |
| **[7]** | 保留 | 保留 |
| **[6]** | **RSTS_PMU** | RSTS_PMU标志位由PMU模块的"复位信号" 置1，用于说明当前中断源.<br>1: PMU发出复位信号使系统复位.<br>0: 没有PMU复位信号<br>给**RSTRC[14]** 写入 0，清该标志位 |
| **[5]** | **RSTS_MCU** | RSTS_MCU标志位由MCU Cortex_M0模块的" 复位信号"置1，用于说明当前中断源.<br>1: 向SYSRESTREQ(AIRCR[2] , Application Interrupt and Reset Control Register)写"1", MCU Cortex_M0 发送复位信号使系统复位.<br>0: 没有MCU复位信号<br>**给RSTRC[13]** 写入 0，清该标志位 |
| **[4]** | **RSTS_BOD** | RSTS_BOD标志位由欠压检测模块的" 复位信号"置1，用于说明当前中断源.<br>1: 欠压检验模块发出复位信号使系统复位.<br>0: 没有BOD复位信号<br>给**RSTRC[12]** 写入 0，清该标志位 |
| **[3]** | **RSTS_LVR** | RSTS_LVR标志位由低压复位模块的" 复位信号"置1，用于说明当前中断源.<br>1: 低压 LVR 模块发出复位信号使系统复位.<br>0: 没有LVR复位信号<br>给**RSTSRC[11] 写入** 0，清该标志位 |
| **[2]** | **RSTS_WDG** | RSTS_WDG标志位由看门狗模块的" 复位信号"置1，用于说明当前中断原.<br>1: 看门狗模块发出复位信号使系统复位.<br>0: 没有看门狗复位信号<br>给**RSTRC[10]** 写入 0，清该标志位 |
| **[1]** | **RSTS_PAD** | RSTS_PAD标志位由/RESET脚的" 复位信号"置1，用于说明当前中断源.<br>1: /RESET脚上发出复位信号使系统复位.<br>0: 没有/RESET复位信号<br>给**RSTRC[9]** 写入 0，清该标志位 |

| | | |
|---|---|---|
| **[0]** | **RSTS_POR** | RSTS_POR标志位由POR模块的” 复位信号”置1，用于说明当前的中断源.<br>1: 上电复位POR发出复位信号使系统复位.<br>0: 没有POR复位信号<br>给**RSTSRC[8]** 写入 0，清该标志位 |

### IP 复位控制寄存器1 (IPRSTC1)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| IPRSTC1 | GCR_BA+08 | R/W | IP 复位控制寄存器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 | | | | | PDMA_RST | CPU_RST | CHIP_RST |

| Bits | 描述 | |
|---|---|---|
| [31:3] | 保留 | 保留 |
| [2] | **PDMA_RST** | **PDMA 控制器复位**<br>该位置1，产生复位信号到PDMA，用户需要置0才能释放复位状态<br>0= 正常工作<br>1= PDMA IP 复位 |
| [1] | **CPU_RST** | **CPU 内核复位.**<br>该位置1，CPU复位，两个时钟周期后，该位返回“0”<br>该位是受保护的位，修改该位时，需要依次向0x5000_0100写入”59h”, “16h”, “88h”, 参考寄存器 RegLockAddr ,地址GCR_BA + 0x100<br>0 : 正常<br>1 : 复位CPU |
| [0] | **CHIP_RST** | **CHIP 复位.**<br>CHIP复位. 该位置1，CHIP复位，两个时钟周期后，该位自动清零.<br>CHIP_RST与上电复位一样，所有的芯片模块都复位，芯片设置从flash重新加载<br>该位是受保护的位，修改该位时，需要依次向0x5000_0100写入”59h”, “16h”, “88h”, 参考寄存器 RegLockAddr ,地址GCR_BA + 0x100<br>0 : 正常<br>1 : 复位CHIP |

### IP 复位控制寄存器2 (IPRSTC2)

置"1" , 产生异步复位信号给相应的IP. 清零时，从复位状态恢复

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| IPRST2 | GCR_BA+0C | R/W | IP 复位控制寄存器 2 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| 保留 | 保留 | I2S_RST | ADC_RST | USBD_RST | 保留 | 保留 | CAN0_RST |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| PS2_RST | ACMP_RST | PWM47_RST | PWM03_RST | 保留 | 保留 | UART1_RST | UART0_RST |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| SPI3_RST | SPI2_RST | SPI1_RST | SPI0_RST | 保留 | 保留 | I2C1_RST | I2C0_RST |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | 保留 | TMR3_RST | TMR2_RST | TMR1_RST | TMR0_RST | GPIO_RST | 保留 |

| Bits | 描述 | |
|---|---|---|
| [31:29] | 保留 | 保留 |
| [29] | **I2S_RST** | **I2S 控制器复位**<br>0= I2S 模块正常工作<br>1= I2S 模块复位 |
| [28] | **ADC_RST** | **ADC控制器复位**<br>"0": ADC 模块正常工作<br>"1": ADC 模块复位 |
| [27] | **USBD_RST** | **USB设备控制器复位**<br>"0": USB模块正常工作<br>"1": USB模块复位 |
| [26] | 保留 | 保留 |
| [25] | 保留 | 保留 |

| | | |
|---|---|---|
| [24] | **CAN0_RST** | **CAN0控制器复位**<br>“0”: CAN0正常工作<br>“1”: CAN0复位 |
| [23] | **PS2_RST** | **PS2控制器复位**<br>“0”: PS2 正常工作<br>“1”: PS2 复位 |
| [22] | **ACMP_RST** | **模拟比较器控制器复位**<br>“0”: 模拟比较器正常工作<br>“1”:模拟比较器复位 |
| [21] | **PWM47_RST** | **PWM47 控制器复位**<br>0= PWM47 模块正常工作<br>1= PWM47 模块复位 |
| [20] | **PWM03_RST** | **PWM03 控制器复位**<br>0= PWM03 模块正常工作<br>1= PWM03 模块复位 |
| [19:18] | 保留 | 保留 |
| [17] | **UART1_RST** | **UART1控制器复位**<br>0= UART1 正常工作<br>1= UART1 模块复位 |
| [16] | **UART0_RST** | **UART0控制器复位**<br>0= UART0 正常工作<br>1= UART0 模块复位 |
| [15] | **SPI3_ RST** | **SPI3控制器复位**<br>0= SPI3 正常工作<br>1= SPI3 模块复位 |
| [14] | **SPI2_ RST** | **SPI2控制器复位**<br>0= SPI2 正常工作<br>1= SPI2 模块复位 |
| [13] | **SPI1_ RST** | **SPI1控制器复位**<br>0= SPI1 正常工作<br>1= SPI1 模块复位 |

| | | |
|---|---|---|
| [12] | **SPI0_ RST** | **SPI0控制器复位**<br>0= SPI0 模块 正常工作<br>1= SPI0 模块复位 |
| [11:10] | 保留 | 保留 |
| [9] | **I2C1_RST** | **I2C1控制器复位**<br>0= I2C1 模块 正常工作<br>1= I2C1 模块复位 |
| [8] | **I2C0_RST** | **I2C0控制器复位**<br>0= I2C0 正常工作<br>1= I2C0 模块复位 |
| [7:6] | 保留 | 保留 |
| [5] | **TMR3_RST** | **Timer3控制器复位**<br>0= Timer3 正常工作<br>1= Timer3 模块复位 |
| [4] | **TMR2_RST** | **Timer2控制器复位**<br>0= Timer2 正常工作<br>1= Timer2 模块复位 |
| [3] | **TMR1_RST** | **Timer1控制器复位**<br>0= Timer1 正常工作<br>1= Timer1 模块复位 |
| [2] | **TMR0_RST** | **Timer0控制器复位**<br>0= Timer0 正常工作<br>1= Timer0 复位 |
| [1] | **GPIO_RST** | **GPIO控制器复位**<br>0= GPIO 正常工作<br>1= GPIO 复位 |
| [0] | 保留 | 保留 |

## 欠压检测控制寄存器(BODCR)

BODCR 控制寄存器的部分值在flash配置时已经初始化. 上电初始化完成之后，这些位被保护起来，用户需要改变BODCR的值，需要顺序向控制寄存器0x5000_0100写入“59h”, “16h” “88h”,不同的数据值或任何写操作于三个数据程序之间，都会使终止整个时序.

解锁后，用户可以检查0x5000_0100的bit0，该位为 ”1”, 表示已经解锁，为 ”0”, 表示锁定. 用户可以更新BODCR的值，可以向0x5000_0100写入任意数据来锁定BODCR.

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **BODCR** | GCR_BA+18 | R/W | 欠压检测控制寄存器 | 0x0000_00xx |

| 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** |
| **LVR_EN** | **BOD_OUT** | **BOD_LP** | **BOD_INTF** | **BOD_RSTEN** | **BOD_VL** | | **BOD_EN** |

| Bits | 描述 | |
|---|---|---|
| [31:8] | 保留 | 保留 |
| [7] | **LVR_EN** | **低压复位使能(write-protected bit)**<br>输入电源电压低于LVR电路设置时，LVR复位.<br>1= 使能低电压复位功能，使能该位100US后，LVR功能生效.**(default)**.<br>0= 禁止低电压复位功能 |
| [6] | **BOD_OUT** | **欠压检测输出的状态位**<br>1 = 欠压检测输出状态为 1, 检测到的电压低于 BOD_VL 设置. 若 BOD_EN 是“0”, 该位通常响应为 “0”<br>0 =欠压检测输出状态为0, t检测到的电压高于BOD_VL设置 |
| [5] | **BOD_LPM** | **低压模式下的欠压检测(write-protected bit)**<br>1 = 使能 BOD 低压模式<br>0 = BOD 工作于正常模式(默认)<br>BOD 在正常模式下消耗电流约为100uA, 低压模式下减少当前的1/10，但BOD响应速度变慢. |

| | | |
|---|---|---|
| [4] | **BOD_INTF** | **欠压检测中断标志**<br>1= 欠压检测到$V_{DD}$ 下降到BOD_VL 的设定电压或$V_{DD}$ 升到BOD_VL 的设定电压，该位设置为1，如果欠压中断被使能，则发生欠压中断.<br>0= 没有检测到任何电压由$V_{DD}$ 下降或上升至BOD_VL 设定值. |
| [3] | **BOD_RSTEN** | **欠压复位使能(initiated & write-protected bit)**<br>1 = 使能欠压复位功能，当欠压检测功能使能后，检测的电压低于threshold ，就使芯片复位<br>默认值由用户在配置flash控制寄存器时config0 bit[20]设置<br>0 = 全能欠压中断功能，当欠压检测功能使能后，检测的电压低于threshold ，就发送中断信号给MCU Cortex-M0<br>当BOD_EN使能，且中断声明时，该中断会持续到将BOD_EN设置为”0”. 通过禁止CPU中的NVIC以禁止BOD中断或者通过禁止BOD_EN禁止中断源可封锁CPU中断，如果需要BOD功能时，可重新使能BOD_EN功能 |
| [2:1] | **BOD_VL** | **欠压检测Threshold 电压选择 (initiated & write-protected bit)**<br>缺省值 由用户在配置FLASH控制寄存器config0 bit[22:21]时设定 |
| [0] | **BOD_EN** | **欠压检测使能(initiated & write-protected bit)**<br>缺省值 由用户在配置FLASH控制寄存器config0 bit[23]时设<br>1 = 使能欠压检测功能<br>0 =禁止欠压检测功能 |

| BOV_VL[1] | BOV_VL[0] | 欠压值 |
|---|---|---|
| 1 | 1 | 4.5V |
| 1 | 0 | 3.8V |
| 0 | 1 | 2.6V |
| 0 | 0 | 2.2V |

温度传感器控制寄存器(TEMPCR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| TEMPCR | GCR_BA+0x1C | R/W | 温度传感器控制寄存器 | 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 | | | | | | | VTEMP_EN |

| Bits | 描述 | |
|---|---|---|
| [31:16] | 保留 | 保留 |
| [9] | VTEMP_EN | 温度传感器使能<br>1= 使能温度传感器功能<br>0= 禁止温度传感器功能 (default) |

多功能GPIOA控制寄存器( (GPA_MFP)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| GPA_MFP | GCR_BA+30 | R/W | 多功能GPIOA 与输入类型控制寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:16] | **GPA_TYPEn** | 1 = 使能PA[15:0] I/O史密特触发输入<br>0 = 禁止PA[15:0] I/O史密特触发输入 |
| [15] | **GPA_MFP15** | **PA.15 Pin 功能选择**<br>1 = PA.15作为**PWM3**<br>0 = PA.15作为GPIOA[15] |
| [14] | **GPA_MFP14** | **PA.14 Pin 功能选择**<br>1 = PA.14作为**PWM2**<br>0 = PA.14作为GPIOA[14] |
| [13] | **GPA_MFP13** | **PA.13 Pin 功能选择**<br>1 = PA.13作为**PWM1**<br>0 = PA.13作为GPIOA[13] |
| [12] | **GPA_MFP12** | **PA.12 Pin 功能选择**<br>1 = PA.12作为**PWM0**<br>0 = PA.12作为GPIOA[12] |
| [11] | **GPA_MFP11** | **PA.11 Pin 功能选择**<br>1 = PA.11作为**I2C1 SCL**<br>0 = PA.11作为GPIOA[11] |

| | | |
|---|---|---|
| [10] | **GPA_MFP10** | **PA.10 Pin 功能选择**<br>1 = PA.10作为**I2C1 SDA**<br>0 = PA.10作为GPIOA[10] |
| [9] | **GPA_MFP9** | **PA.9 Pin 功能选择**<br>1 = PA.9作为**I2C0 SCL**<br>0 = PA.9作为GPIOA[9] |
| [8] | **GPA_MFP8** | **PA.8 Pin 功能选择**<br>1 = PA.8作为**I2C0 SDA**<br>0 = PA.8作为GPIOA[8] |
| [7] | **GPA_MFP7** | **PA.7 Pin 功能选择**<br>该引脚功能取决于GPA_MFP7 与PA7_S21(ALT_MFP[2]).<br>PA7_S21 / GPA_MFP[7] / PA.7 function<br>0 / 0 / **GPIO**<br>x / 1 / **ADC7**<br>1 / 0 / **/SS21 (SPI2)** |
| [6] | **GPA_MFP6** | **PA.6 Pin 功能选择**<br>1 = PA.6作为ADC6(模拟到数字转换通道6)<br>0 = PA.6作为GPIOA[6] |
| [5] | **GPA_MFP5** | **PA.5 Pin 功能选择**<br>1 = PA.5作为**ADC5**(模拟到数字转换通道5)<br>0 = PA.5作为 GPIOA[5] |
| [4] | **GPA_MFP4** | **PA.4 Pin 功能选择**<br>1 = PA.4作为**ADC4**(模拟到数字转换通道4)<br>0 = PA.4作为 GPIOA[4] |
| [3] | **GPA_MFP3** | **PA.3 Pin 功能选择**<br>1 = PA.3作为**ADC3**(模拟到数字转换通道3)<br>0 = PA.3作为GPIOA[3] |
| [2] | **GPA_MFP2** | **PA.2 Pin 功能选择**<br>1 = PA.2作为**ADC2**(模拟到数字转换通道2)<br>0 = PA.2作为PIOA[2] |

| | | |
|---|---|---|
| [1] | **GPA_MFP1** | **PA.1 Pin 功能选择**<br>1 = PA.1作为**ADC1**(模拟到数字转换通道1)<br>0 = PA.1作为GPIOA[1] |
| [0] | **GPA_MFP0** | **PA.0 Pin 功能选择**<br>1 = PA.0作为**ADC0**(模拟到数字转换通道0)<br>0 = PA.0作为GPIOA[0] |

### 多功能GPIOB控制寄存器(GPB_MFP)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **GPB_MFP** | GCR_BA+34 | R/W | 多功能GPIOB 与输入类型控制寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:16] | **GPB_TYPEn** | 1 = 使能GPB[15:0] I/O 史密特触发输入<br>0 = 禁止GPB[15:0] I/O史密特触发输入 |
| [15] | **GPB_MFP15** | **PB.15 Pin 功能选择**<br>1 = PB.15作为**外部中断 INT1**<br>0 = PB.15作为GPIOB[15] |
| [14] | **GPB_MFP14** | **PB.14 Pin 功能选择**<br>该引脚功能取决于GPB_MFP14 与PB14_S31(ALT_MFP[3]).<br>PB14_S31 \| GPB_MFP[14] \| PB.14 功能<br>0 \| 0 \| **GPIO**<br>x \| 1 \| **/INT1**<br>1 \| 0 \| **/SS31 (SPI3)** |
| [13] | **GPB_MFP13** | **PB.13 Pin 功能选择**<br>1 = PB.13作为CPO1 **(模拟比较器1 数字输出)**<br>0 = PB.13作为 GPIOB[13] |

<table>
<tr><td>[12]</td><td>GPB_MFP12</td><td>PB.12 Pin 功能选择<br>该引脚功能取决 GPB_MFP12 与 PB12_CLKO (ALT_MFP[10]).<br><table><tr><td>PB12_CLKO</td><td>GPB_MFP[12]</td><td>PB.12 功能</td></tr><tr><td>x</td><td>0</td><td>GPIO</td></tr><tr><td>0</td><td>1</td><td>CPO0</td></tr><tr><td>1</td><td>1</td><td>CLKO (Clock Driver output)</td></tr></table></td></tr>
<tr><td>[11]</td><td>GPB_MFP11</td><td>PB.11 Pin 功能选择<br>该引脚功能取决 GPB_MFP11 与 PB11_PWM4(ALT_MFP[4]).<br><table><tr><td>PB11_PWM4</td><td>GPB_MFP[11]</td><td>PB.11功能</td></tr><tr><td>x</td><td>0</td><td>GPIO</td></tr><tr><td>0</td><td>1</td><td>TM3</td></tr><tr><td>1</td><td>1</td><td>PWM4</td></tr></table></td></tr>
<tr><td>[10]</td><td>GPB_MFP10</td><td>PB.10 Pin 功能选择<br>该引脚功能取决 GPB_MFP10 与 PB10_S01(ALT_MFP[0]).<br><table><tr><td>PB10_S01</td><td>GPB_MFP[10]</td><td>PB.10 功能</td></tr><tr><td>0</td><td>0</td><td>GPIO</td></tr><tr><td>x</td><td>1</td><td>TM2</td></tr><tr><td>1</td><td>0</td><td>/SS01 (SPI0)</td></tr></table></td></tr>
<tr><td>[9]</td><td>GPB_MFP9</td><td>PB.9 Pin 功能选择<br>该引脚功能取决 GPB_MFP9 与 PB9_S11(ALT_MFP[1]).<br><table><tr><td>PB9_S11</td><td>GPB_MFP[9]</td><td>PB.9 功能</td></tr><tr><td>0</td><td>0</td><td>GPIO</td></tr><tr><td>x</td><td>1</td><td>TM1</td></tr><tr><td>1</td><td>0</td><td>/SS11 (SPI1)</td></tr></table></td></tr>
<tr><td>[8]</td><td>GPB_MFP8</td><td>PB.8 Pin 功能选择<br>1 = PB.8作为TM0 (定时器/计数器外部触发时钟输入)<br>0 = PB.8作为GPIOB[8]</td></tr>
</table>

| | | |
|---|---|---|
| [7] | **GPB_MFP7** | **PB.7 Pin 功能选择**<br>1 = PB.7作为**UART1 CTS#**<br>0 = PB.7作为GPIOB[7] |
| [6] | **GPB_MFP6** | **PB.6 Pin 功能选择**<br>1 = PB.6作为 **UART1 RTS#**<br>0 = PB.6作为GPIOB[6] |
| [5] | **GPB_MFP5** | **PB. 5 Pin 功能选择**<br>1 = PB.5作为**UART1 TX**<br>0 = PB.5作为GPIOB-5 |
| [4] | **GPB_MFP4** | **PB.4 Pin 功能选择**<br>1 = PB.4作为**UART1 RX**<br>0 = PB.4作为GPIOB[4] |
| [3] | **GPB_MFP3** | **PB.3 Pin 功能选择**<br>1 = PB.3作为**UART0 CTS#**<br>0 = PB.3作为GPIOB[3] |
| [2] | **GPB_MFP2** | **PB.2 Pin 功能选择**<br>1 = PB.2作为**UART0 RTS#**<br>0 = PB.2作为GPIOB[2] |
| [1] | **GPB_MFP1** | **PB.1 Pin 功能选择**<br>1 = PB.1作为**UART0 TX**<br>0 = PB.1作为GPIOB[1] |
| [0] | **GPB_MFP0** | **PB.0 Pin 功能选择**<br>1 = PB.0作为**UART0 RX**<br>0 = PB.0作为GPIOB[0] |

### 多功能GPIOC控制寄存器(GPC_MFP)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| GPC_MFP | GCR_BA+38 | R/W | 多功能GPIOC 与输入类型控制寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:16] | **GPC_TYPEn** | 1 = 使能GPC[n] I/O 史密特触发输入<br>0 = 禁止GPC[n] I/O史密特触发输入 |
| [15] | **GPC_MFP15** | **PC.15 Pin 功能选择**<br>1 = PC.15作为**CPN1** (模拟比较器-1模拟输入 CPN)<br>0 = PC.15作为 GPIOC[15] |
| [14] | **GPC_MFP14** | **PC.14 Pin 功能选择**<br>1 = PC.14作为**CPP1** (模拟比较器-1模拟输入CPP)<br>0 = PC.14作为GPIOC[14] |
| [13] | **GPC_MFP13** | **PC.13 Pin 功能选择**<br>1 = PC.13作为**SPI1 MOSI1** (主输出，从输入引脚-1)<br>0 = PC.13作为GPIOC[13] |
| [12] | **GPC_MFP12** | **PC.12 Pin 功能选择**<br>1 = PC.12作为**SPI1 MISO1** (主输入，从输出引脚-1)<br>0 = PC.12作为GPIOC[12] |
| [11] | **GPC_MFP11** | **PC.11 Pin 功能选择**<br>1 = PC.11作为**SPI1 MOSI0** (主输出，从输入引脚-0)<br>0 = PC.11作为GPIOC[11] |

| | | |
|---|---|---|
| [10] | **GPC_MFP10** | **PC.10 Pin 功能选择**<br>1 = PC.10作为**SPI1 MISO0** (主输入，从输出引脚-0)<br>0 = PC.10作为GPIOC[10] |
| [9] | **GPC_MFP9** | **PC.9 Pin 功能选择**<br>1 = PC.9作为 **SPI1 SPCLK**<br>0 = PC.9作为GPIOC[9] |
| [8] | **GPC_MFP8** | **PC.8 Pin 功能选择**<br>1 = PC.8作为**SPI1 SS10**<br>0 = PC.8作为GPIOC[8] |
| [7] | **GPC_MFP7** | **PC.7 Pin 功能选择**<br>1 = PC.7作为**CPN0** (模拟比较器-0模拟输入 CPN)<br>0 = PC.7作为GPIOC[7] |
| [6] | **GPC_MFP6** | **PC.6 Pin 功能选择**<br>1 = PC.6作为**CPP0** (模拟比较器-0模拟输入CPP)<br>0 = PC.6作为GPIOC[6] |
| [5] | **GPC_MFP5** | **PC.5 Pin 功能选择**<br>1 = PC.5作为**SPI0 MOSI1** (主输出，从输入引脚-1)<br>0 = PC.5作为GPIOC[4] |
| [4] | **GPC_MFP4** | **PC.4 Pin 功能选择**<br>1 = PC.4作为**SPI0 MISO1** (主输入，从输出引脚-1)<br>0 = PC.4作为 GPIOC[4] |
| [3] | **GPC_MFP3** | **PC.3 Pin 功能选择**<br>PC2_I2SDO(ALT_MFP[8]) 与 GPC_MFP[3] 决定PC.3的功能.<br>PC2_I2SDO \| GPC_MFP[3] \| PC.3 功能<br>x \| 0 \| **GPIO**<br>0 \| 1 \| **MOSI00 (SPI0)**<br>1 \| 1 \| **I2SDO** |

<table>
<tr>
<td>[2]</td>
<td>GPC_MFP2</td>
<td>PC.2 Pin 功能选择<br>PC2_I2SDI(ALT_MFP[7]) 与 GPC_MFP[2] 决定 PC.2的功能.
<table>
<tr><td>PC2_I2SDI</td><td>GPC_MFP[2]</td><td>PC.2 功能</td></tr>
<tr><td>x</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>MISO00 (SPI0)</td></tr>
<tr><td>1</td><td>1</td><td>I2SDI</td></tr>
</table>
</td>
</tr>
<tr>
<td>[1]</td>
<td>GPC_MFP1</td>
<td>PC.1 Pin 功能选择<br>PC1_I2SBCLK(ALT_MFP[6]) 与 GPC_MFP[1] 决定 PC.1的功能.
<table>
<tr><td>PC1_I2SBCLK</td><td>GPC_MFP[1]</td><td>PC.1 功能</td></tr>
<tr><td>x</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>SPICLK0 (SPI0)</td></tr>
<tr><td>1</td><td>1</td><td>I2SBCLK</td></tr>
</table>
</td>
</tr>
<tr>
<td>[0]</td>
<td>GPC_MFP0</td>
<td>PC.0 Pin 功能选择<br>PC0_I2SLRCLK(ALT_MFP[5]) 与 GPC_MFP[0] 决定 PC.0的功能.
<table>
<tr><td>PC0_I2SLRCLK</td><td>GPC_MFP[0]</td><td>PC.0 功能</td></tr>
<tr><td>x</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>/SS00 (SPI0)</td></tr>
<tr><td>1</td><td>1</td><td>I2SLRCLK</td></tr>
</table>
</td>
</tr>
</table>

多功能GPIOD控制寄存器(GPD_MFP)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| GPD_MFP | GCR_BA+3C | R/W | 多功能GPIOD 与输入类型控制寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:16] | **GPD_TYPEn** | 1 = 使能GPIOD[15:0] I/O 史密特触发输入<br>0 = 禁止GPIOD[15:0] I/O史密特触发输入 |
| [15] | **GPD_MFP15** | **PD.15 Pin 功能选择**<br>1= PD.15作为**UART TX2**<br>0= PD.15作为GPIOD[15] |
| [14] | **GPD_MFP14** | **PD.14 Pin 功能选择**<br>1= PD.14作为 **UART RX2**<br>0=PD.14作为 GPIOD[14] |
| [13] | **GPD_MFP13** | **PD.13 Pin 功能选择**<br>1= PD.13作为 **SPI3 MOSI1** (master output, slave input pin-1)<br>0= PD.13作为 GPIOD[13] |
| [12] | **GPD_MFP12** | **PD.12 Pin 功能选择**<br>1= PD.12作为**SPI3 MISO1** (master input, slave output pin-1)<br>0= PD.12作为GPIOD[12] |

| | | |
|---|---|---|
| [11] | **GPD_MFP11** | **PD.11 Pin 功能选择**<br>1= PD.11作为**SPI3 MOSI0** (master output, slave input pin-0)<br>0= PD.11作为GPIOD[11] |
| [10] | **GPD_MFP10** | **PD.10 Pin 功能选择**<br>1= PD.10作为**SPI3 MISO0** (master input, slave output pin-0)<br>0= PD.10作为GPIOD[10] |
| [9] | **GPD_MFP9** | **PD.9 Pin 功能选择**<br>1= PD.9作为**SPI3 SPCLK**<br>0= PD.9作为GPIOD-9 |
| [8] | **GPD_MFP8** | **PD.8 Pin 功能选择**<br>1= PD.8作为**SPI3 SS30**<br>0= PD.8作为GPIOD[8] |
| [7] | **GPD_MFP7** | **PD.7 Pin 功能选择**<br>1= PD.7作为**CAN0 TX**<br>0= PD.7作为GPIOD[7] |
| [6] | **GPD_MFP6** | **PD.6 Pin 功能选择**<br>1= PD.6作为**CAN0 RX**<br>0= PD.6作为GPIOD[6] |
| [5] | **GPD_MFP5** | **PD.5 Pin 功能选择**<br>1= PD.5作为**SPI2 MOSI1** (master output, slave input pin-1)<br>0= PD.5作为GPIOD[5] |
| [4] | **GPD_MFP4** | **PD.4 Pin 功能选择**<br>1= PD.4作为**SPI2 MISO1** (master input, slave output pin-1)<br>0= PD.4作为GPIOD[4] |
| [3] | **GPD_MFP3** | **PD.3 Pin 功能选择**<br>1= PD.3作为**SPI2 MOSI0** (master output, slave input pin-0)<br>0= PD.3作为GPIOD[3] |
| [2] | **GPD_MFP2** | **PD.2 Pin 功能选择**<br>1= PD.2作为**SPI2 MISO0** (master input, slave output pin-0)<br>0= PD.2作为GPIOD[2] |

| | | |
|---|---|---|
| [1] | **GPD_MFP1** | **PD.1 Pin 功能选择**<br>1= PD.1作为**SPI2 SPCLK**<br>0= PD.1作为GPIOD[1] |
| [0] | **GPD_MFP0** | **PD.0 Pin 功能选择**<br>1= PD.0作为**SPI2 SS20**<br>0= PD.0作为GPIOD[0] |

### 多功能GPIOE控制寄存器(GPE_MFP)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| GPE_MFP | GCR_BA+40 | R/W | GPIOE输入类型控制寄存器 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| GPE_TYPE[15:8] | | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| GPE_TYPE[7:0] | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| 保留 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | GPE_MFP5 | 保留 | | | GPE_MFP1 | GPE_MFP0 |

| Bits | 描述 | |
|---|---|---|
| [31:16] | **GPE_TYPEn** | 1 = 使能 GPE[15:0] I/O 史密特触发输入<br>0 = 禁止 GPE[15:0] I/O 史密特触发输入 |
| [15:0] | 保留 | 保留 |
| [5] | **GPE_MFP5** | **PE.5 Pin 功能选择**<br>1 = PE.5作为**PWM5**<br>0 = PE.5作为GPIOE[5] |
| [4:2] | 保留 | 保留 |
| [1] | **GPE_MFP1** | **PE.1 Pin 功能选择**<br>1 = PE.1作为**PWM7**<br>0 = PE.1作为GPIOE[1] |
| [0] | **GPE_MFP0** | **PE.0 Pin 功能选择**<br>1 = PE.0作为**PWM6**<br>0 = PE.0作为GPIOE[0] |

可选多功能引脚控制寄存器 (ALT_MFP)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| ALT_MFP | GCR_BA+50 | R/W | 可选多功能引脚控制寄存器 | 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 |
| | | | | | PB12_CLKO | PA15_I2SMCLK | PC3_I2SDO |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PC2_I2SDI | PC1_I2SBCLK | PC0_I2SLRCLK | PB11_PWM4 | PB14_S31 | PA7_S21 | PB9_S11 | PB10_S01 |

| Bits | 描述 | |
|---|---|---|
| [31:5] | 保留 | 保留 |
| [10] | **PB12_CLKO** | PB12_CLKO 与 GPB_MFP[12] 决定PB.12 的功能.<br>PB12_CLKO / GPB_MFP[12] / PB.12功能<br>x / 0 / GPIO<br>0 / 1 / CPO0<br>1 / 1 / CLKO (时钟分频输出) |
| [9] | **PA15_I2SMCLK** | A15_I2SMCLK 与 GPA_MFP[15] 决定PA.15 的功能.<br>PA15_I2SMCLK / GPA_MFP[15] / PA.15 功能<br>x / 0 / GPIO<br>0 / 1 / MISO00 (SPI0)<br>1 / 1 / I2SDI |

<table>
<tr>
<td>[8]</td>
<td>PC3_I2SDO</td>
<td>PC2_I2SDO 与 GPC_MFP[3] 决定PC.3 的功能.
<table>
<tr><td>PC2_I2SDO</td><td>GPC_MFP[3]</td><td>PC.3 功能</td></tr>
<tr><td>x</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>MOSI00 (SPI0)</td></tr>
<tr><td>1</td><td>1</td><td>I2SDO</td></tr>
</table>
</td>
</tr>
<tr>
<td>[7]</td>
<td>PC2_I2SDI</td>
<td>PC2_I2SDI 与 GPC_MFP[2] 决定PC.2 的功能.
<table>
<tr><td>PC2_I2SDI</td><td>GPC_MFP[2]</td><td>PC.2 功能</td></tr>
<tr><td>x</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>MISO00 (SPI0)</td></tr>
<tr><td>1</td><td>1</td><td>I2SDI</td></tr>
</table>
</td>
</tr>
<tr>
<td>[6]</td>
<td>PC1_I2SBCLK</td>
<td>PC1_I2SBCLK 与 GPC_MFP[1] 决定PC.1 的功能.
<table>
<tr><td>PC1_I2SBCLK</td><td>GPC_MFP[1]</td><td>PC.1 功能</td></tr>
<tr><td>x</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>SPICLK0 (SPI0)</td></tr>
<tr><td>1</td><td>1</td><td>I2SBCLK</td></tr>
</table>
</td>
</tr>
<tr>
<td>[5]</td>
<td>PC0_I2SLRCLK</td>
<td>PC0_I2SLRCLK 与 GPC_MFP[0] 决定PC.0 的功能.
<table>
<tr><td>PC0_I2SLRCLK</td><td>GPC_MFP[0]</td><td>PC.0 功能</td></tr>
<tr><td>x</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>/SS00 (SPI0)</td></tr>
<tr><td>1</td><td>1</td><td>I2SLRCLK</td></tr>
</table>
</td>
</tr>
<tr>
<td>[4]</td>
<td>PB11_PWM4</td>
<td>PB11_PWM4 与 GPB_MFP[11] 决定PB.11的功能.
<table>
<tr><td>PB11_PWM4</td><td>GPB_MFP[11]</td><td>PB.11 功能</td></tr>
<tr><td>x</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>TM3</td></tr>
<tr><td>1</td><td>1</td><td>PWM4</td></tr>
</table>
</td>
</tr>
</table>

<table>
<tr>
<td>[3]</td>
<td>PB14_S31</td>
<td>PB14_S31 与 GPB_MFP[14] 决定PB.14的功能.
<table>
<tr><td>PB14_S31</td><td>GPB_MFP[14]</td><td>PB.14 功能</td></tr>
<tr><td>x</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>/INT1</td></tr>
<tr><td>1</td><td>1</td><td>/SS31 (SPI3)</td></tr>
</table>
</td>
</tr>
<tr>
<td>[2]</td>
<td>PA7_S21</td>
<td>PA7_S21 与 GPA_MFP[7] 决定PA.7的功能.
<table>
<tr><td>PA7_S21</td><td>GPA_MFP[7]</td><td>PA.7 function</td></tr>
<tr><td>x</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>ADC7</td></tr>
<tr><td>1</td><td>1</td><td>/SS21 (SPI2)</td></tr>
</table>
</td>
</tr>
<tr>
<td>[1]</td>
<td>PB9_S11</td>
<td>PB9_S11 与 GPB_MFP[9] 决定PB.9的功能.
<table>
<tr><td>PB9_S11</td><td>GPB_MFP[9]</td><td>PB.9 功能</td></tr>
<tr><td>x</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>TM1</td></tr>
<tr><td>1</td><td>1</td><td>/SS11 (SPI1)</td></tr>
</table>
</td>
</tr>
<tr>
<td>[0]</td>
<td>PB10_S01</td>
<td>PB10_S01 与 GPB_MFP[10] 决定 GPB.10的功能.
<table>
<tr><td>PB10_S01</td><td>GPB_MFP[10]</td><td>PB.10 功能</td></tr>
<tr><td>x</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>TM2</td></tr>
<tr><td>1</td><td>1</td><td>/SS01 (SPI0)</td></tr>
</table>
</td>
</tr>
</table>

### 寄存器锁定键地址寄存器(RegLockAddr)

有些系统控制寄存器需要被保护起来，以防止误操作而影响芯片运行，这些寄存器在上电复位到用户解锁定之前是锁定的。用户可以连续依次写入“59h”, “16h” “88h”到0x5000_0100解锁定.

解锁定后，用户可以检测0x5000_0100 的bit0，“1“表示已经解锁定，”0“表示锁定。用户可以更新目标寄存器的值，向“0x5000_0100”写入任何值，就可以重锁保护寄存器

该寄存器用于解锁定RegUnLock 和读RegUnLock的状态

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **RegLockAddr** | GCR_BA+100 | R/W | 寄存器锁定键地址寄存器 | 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 |
| | | | | | | | **RegUnLock** |

| Bits | 描述 | |
|---|---|---|
| [31:16] | 保留 | 保留 |
| [0] | **RegUnLock** | 1 = 受保护寄存器已解锁定<br>0 = 受保护寄存器已锁定,不能向目标寄存器中写入数据.<br>受保护的寄存器如下:<br>IPRST1 – 地址 0x5000_0008<br>BODCR – 地址 0x5000_0018<br>PORCR – 地址 0x5000_001C<br>RCADJ - 地址 0x5000_0110<br>PWRCON – 地址 0x5000_0200 (在电源唤醒中断被清时，bit[6]不受保护)<br>CLK_APBC bit[0] – 地址 0x5000_0208 (bit[0] 是看门狗时钟使能)<br>CLK_SEL0 – 地址 0x5000_0210 (选择 HCLK 与CPU STCLK 时钟源)<br>IrqTestMode – 地址 0x5000_0380 bit[7]<br>ISPCON – 地址 0x5000_C000 (Flash ISP 控制寄存器)<br>Watch_Dog – 地址 0x4000_4000<br>FATCON – 地址 0x5000_C018 |

### RC 校验控制寄存器(RCADJ)

| 寄存器 | 偏移量 | R/W | 描述 | 偏移后的值 |
|---|---|---|---|---|
| RCADJ | GCR_BA+110 | R/W | RC校验控制寄存器 | 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 | | RCADJ | | | | | |

| Bits | 描述 | |
|---|---|---|
| [31:6] | 保留 | 保留 |
| [5:0] | **RCADJ** | NUC1xx 内建22.1184MHz RC 振荡器,这些位存储一些经CP/FT校验的值, 用于提供 +/- 1% 频率精确度, 默认的中心频率22MHz是“10000b”<br><br>上电后，这些位由锁定电路保护起来。用户需要编辑RCADJ的内容时，可按如下方式写入解锁时序. 开锁时序是依次向0x5000_0100写入“59h”, “16h” “88h”. 不同的数据值或任何写操作于三个数据程序之间，都会使终止整个时序 |

### 5.2.5 系统定时器 (SysTick)

Cortex-M0 包含系统定时器, SysTick. SysTick 提供一种简单，24位计数器，可灵活控制。计数器可作如下几种不同应用，例：

- 一个RTOS 滴答定时器，频率可编程(如 100Hz)可调用SysTick程序.
- 一个高速报警定时器，使用内核时钟.
- 可变速率的报警或信号定时器– 其周期取决于所使用的参考时钟和动态计数器.
- 一个简单计数器，软件可用它测量时间.
- 一个内部时钟源控制. 计数标志位用于决定动作是否在设定期间内完成.

使能后，定时器从SysTick 当前寄存器(SYST_CVR)的值向下计数到0，下一个时钟边缘，重新加载寄存器(SYST_RVR) 的值。当计数器减到0时，标志位COUNTFLAG置位，在读数时，清COUNTFLAG标志位.

复位后，SYST_CVR 的值未知. 使能前，软件应该向寄存器写入0. 这样确保定时器以SYST_RVR中的值计数，而非任意值.

若SYST_RVR 是0 ，在重新加载后，定时器将保持当前值0.

详情请参考“ARM® Cortex™-M0 Technical Reference Manual” 与 “ARM® v6-M Architecture Reference Manual”.

#### *5.2.5.1 系统定时器控制寄存器列表*

**R: 只读, W: 只写, R/W: 可读写, W&C: 写1清零**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| SCS_BA = 0xE000_E000 | | | | |
| **SYST_CSR** | SCS_BA + 010 | R/W | SysTick 控制与状态 | 0x0000_0004 |
| **SYST_RVR** | SCS_BA + 014 | R/W | SysTick 重新加载值 | 0xXXXX_XXXX |
| **SYST_CVR** | SCS_BA + 018 | R/W | SysTick 当前值 | 0xXXXX_XXXX |

#### *5.2.5.2 系统定时器控制寄存器描述*

**SysTick 控制与状态（SYST_CSR）**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **SYST_CSR** | SCS_BA + 010 | R/W | SysTick 控制与状态 | 0x0000_0004 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | |
| **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |

| 保留 | | | | | | | COUNTFLAG |
|---|---|---|---|---|---|---|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| 保留 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | | | | CLKSRC | TICKINT | ENABLE |

| Bits | 描述 | |
|---|---|---|
| [31:17] | 保留 | 保留 |
| [16] | **COUNTFLAG** | 从上次该寄存器读取，如果定时器计数到0，则返回1.<br>计数由1到0时，COUNTFLAG 置位.<br>在读或向当前寄存器写时，COUNTFLAG 被清零. |
| [15:3] | 保留 | 保留 |
| [2] | **CLKSRC** | 1 : 如果没有外部时钟时，可用内核时钟作SysTick, 该位被读为1 ，不能写.<br>0 : 时钟源为外部参考时钟 |
| [1] | **TICKINT** | 1 : 向下计数到0将引起SysTick 异常而挂起. 清SysTick 当前寄存器的值将不会导致 SysTick 挂起.<br>0 : 向下计数到0不会引起SysTick异常而挂起. 软件设置COUNTFLAG 来确定是否已经发生计数到0 |
| [0] | **ENABLE** | 1 : 计数器运行于multi-shot manner.<br>0 : 禁止计数器 |

**SysTick 重新加载值寄存器（SYST_RVR）**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **SYST_RVR** | SCS_BA + 014 | R/W | SysTick 重新加载值寄存器 | 0xXXXX_XXXX |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| RELOAD[23:16] | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |

| RELOAD[15:8] | | | | | | | |
|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RELOAD[7:0] | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [31:24] | 保留 | 保留 |
| [23:0] | **RELOAD** | 当计数器达到0时，值加载到当前值寄存器. |

**SysTick 当前值寄存器（SYST_CVR）**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| SYST_CVR | SCS _BA + 018 | R/W | SysTick 当前值寄存器 | 0xXXXX_XXXX |

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

| Bits | 描述 | |
|---|---|---|
| [31:24] | 保留 | 保留 |
| [23:0] | **CURRENT** | 当前计数值，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，该寄存器为write-clear. 软件写入任何值将清寄存器为0. Unsupported bits RAZ (see SysTick Reload Value register). |

### 5.2.6 嵌套向量中断控制器 (NVIC)

Cortex-M0 提供中断控制器，用于总体管理异常，称之为“嵌套向量中断控制器(NVIC)”. 提供以下特征:

- 支持嵌套和向量中断
- 自动保存和恢复处理器状态
- 动态改变优先级
- 减少和确定中断时间

NVIC区分和处理所有支持的异常，所有异常在“处理器模式“处理. NVIC结构支持32(IRQ[31:0])4级离散中断优先级. 所有的中断和大多数系统异常可以配置为不同优先级. 当中断发生时，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.

当接受任何中断时，ISR的开始地址可从内存的向量表中取得. 不需要确定哪个中断被响应，也不要软件分配相关中断服务程序（ISR）的开始地址。当开始地址取得时，NVIC将自动保存处理状态到栈中，包括以下寄存器“PC, PSR, LR, R0~R3, R12” 的值. 在ISR结束时, NVIC 将从栈中恢复相关寄存器的值，进行正常操作,因此花费少量时处理中断请求.

NVIC支持末尾连锁“Tail Chaining” ，有效处理背对背中断“back-to-back interrupts”，以减少在切换ISR时的延迟时间. NVIC还支持迟到 “Late Arrival”，改善同时发生的ISR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（at the stage of state saving and starting address fetching），NVIC给优先级最高的，而没有延迟，从而提高了适时性.

详情请参考“ARM® Cortex™-M0 Technical Reference Manual” 与 “ARM® v6-M Architecture Reference Manual”.

#### *5.2.6.1 异常模式和系统中断列表*

NUC1xx 系列支持下表所列的异常模式. 与所有中断一样，软件可以对其中一些中断设置4级优先级. 最高优先级为“0”，最低优先级为“3”，所有用户可配置的优先级的默认值为“0”. 注意：优先级为“0”在整个系统中为第4优先级，排在“Reset”, “NMI” 与“Hard Fault”之后.

| *异常名称* | *向量号* | *优先级* |
|---|---|---|
| ***Reset*** | *1* | -3 |
| ***NMI*** | *2* | -2 |
| ***Hard Fault*** | *3* | -1 |

| | | |
|---|---|---|
| *保留* | *4 ~ 10* | **保留** |
| ***SVCall*** | *11* | 可配置 |
| *保留* | *12 ~ 13* | **保留** |
| ***PendSV*** | *14* | 可配置 |
| ***SysTick*** | *15* | 可配置 |
| ***Interrupt (IRQ0 ~ IRQ31)*** | *16 ~ 47* | 可配置 |

表 5-2 异常模式

| 向量号 | 中断号 (Bit in Interrupt Registers) | 中断名称 | 源 IP | 中断描述 |
|---|---|---|---|---|
| 0 ~ 15 | - | - | - | 系统异常 |
| 16 | 0 | **BOD_OUT** | Brown-Out | 欠压检测中断 |
| 17 | 1 | **WDT_INT** | WDT | Watch Dog Timer中断 |
| 18 | 2 | **EINT0** | GPIO | PB.14脚上的外部信号中断 |
| 19 | 3 | **EINT1** | GPIO | PB.15脚上的外部信号中断 |
| 20 | 4 | **GPAB_INT** | GPIO | PA[15:0] / PB[13:0]的外部信号中断 |
| 21 | 5 | **GPCDE_INT** | GPIO | PC[15:0]/PD[15:0]/PE[15:0] 的外部信号中断 |
| 22 | 6 | **PWMA_INT** | PWM0~3 | PWM0, PWM1, PWM2 与 PWM3 中断 |
| 23 | 7 | **PWMB_INT** | PWM4~7 | PWM4, PWM5, PWM6 与 PWM7中断 |
| 24 | 8 | **TMR0_INT** | TMR0 | Timer 0中断 |
| 25 | 9 | **TMR1_INT** | TMR1 | Timer 1中断 |
| 26 | 10 | **TMR2_INT** | TMR2 | Timer 2中断 |
| 27 | 11 | **TMR3_INT** | TMR3 | Timer 3中断 |
| 28 | 12 | **UART0_INT** | UART0 | UART0中断 |

| | | | | |
|---|---|---|---|---|
| 29 | 13 | **UART1_INT** | UART1 | UART1中断 |
| 30 | 14 | **SPI0_INT** | SPI0 | SPI0中断 |
| 31 | 15 | **SPI1_INT** | SPI1 | SPI1中断 |
| 32 | 16 | **SPI2_INT** | SPI2 | SPI2中断 |
| 33 | 17 | **SPI3_INT** | SPI3 | SPI3中断 |
| 34 | 18 | **I2C0_INT** | I2C0 | I2C0中断 |
| 35 | 19 | **I2C1_INT** | I2C1 | I2C1中断 |
| 36 | 20 | **CAN0_INT** | CAN0 | CAN0中断 |
| 37 | 21 | 保留 | 保留 | 保留 |
| 38 | 22 | 保留 | 保留 | 保留 |
| 39 | 23 | **USB_INT** | USBD | USB FS 设备中断 |
| 40 | 24 | **PS2_INT** | PS2 | PS2 中断 |
| 41 | 25 | **ACMP_INT** | ACMP | 模拟比较器0或模拟比较器1中断 |
| 42 | 26 | **PDMA_INT** | PDMA | PDMA 中断 |
| 43 | 27 | **I2S_INT** | I2S | I2S 中断 |
| 44 | 28 | **PWRWU_INT** | CLKC | 从掉电状态唤醒的时钟控制器中断 |
| 45 | 29 | **ADC_INT** | ADC0/1 | ADC 中断 |
| 46 | 30 | 保留 | 保留 | 保留 |
| 47 | 31 | **RTC_INT** | RTC | RTC中断 |

表 5-3 系统中断列表

#### *5.2.6.2 向量表*

响应中断时，处理器自动从向量表中取出ISR的起始地址. 对于ARMv6-M, 向量表的基地址为0x00000000. 向量表包括复位后栈的初始值，所有异常处理器的入口地址. 向量号表示处理异常的先后次序.

| 向量表字偏移量 | 描述 |
|---|---|
| 0 | SP_main – 主栈指针 |
| Vector Number | 异常入口指针，用向量号表示 |

表 5-4 向量表格式

*5.2.6.3 操作说明*

通过设置相应中断使能置位寄存器或清使能寄存器，可以使能NVIC中断或禁止NVIC中断,这些寄存器通过写1使能和写1清零，寄存器读取当前相应中断的使能状态，当中断禁止时，中断声明将使中断挂起，因此中断不被激活，如果在禁止时中断被激活，就保持在激活状态，直到通过复位或异常返回来清除。清使能位可以阻止新的相应中断被激活.

NVIC 中断可以使用互补的寄存器对来挂起/取消挂起以使能/禁止这些中断，这些寄存器分别为et-Pending Register 与 Clear-Pending , 可以写1使能和写1清零，这些寄存器返回当前相应中断的状态. 寄存器Clear-Pending 在中断响应时不影响执行状态.

NVIC中断依次更新32位寄存器中的各个8位字段（每个寄存器支持4个中断）.

与NVIC相关的通用寄存器都可以在内存系统控制空间设置，下一节将作出描述.

#### 5.2.6.4 NVIC控制寄存器

**R: 只读, W: 只写, R/W: 可读写, W&C: 写1清零**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| SCS_BA = 0xE000_E000 | | | | |
| **NVIC _ISER** | SCS_BA + 100 | R/W | IRQ0 ~ IRQ31 设置使能控制寄存器 | 0x0000_0000 |
| **NVIC_ICER** | SCS_BA + 180 | R/W | IRQ0 ~ IRQ31 清使能控制寄存器 | 0x0000_0000 |
| **NVIC_ISPR** | SCS_BA + 200 | R/W | IRQ0 ~ IRQ31 设置挂起控制寄存器 | 0x0000_0000 |
| **NVIC_ICPR** | SCS_BA + 280 | R/W | IRQ0 ~ IRQ31 清挂起控制寄存器 | 0x0000_0000 |
| **NVIC_IPR0** | SCS_BA + 400 | R/W | IRQ0 ~ IRQ3 优先级 控制寄存器 | 0x0000_0000 |
| **NVIC_IPR1** | SCS_BA + 404 | R/W | IRQ4 ~ IRQ7优先级控制寄存器 | 0x0000_0000 |
| **NVIC_IPR2** | SCS_BA + 408 | R/W | IRQ8 ~ IRQ11优先级控制寄存器 | 0x0000_0000 |
| **NVIC_IPR3** | SCS_BA + 40C | R/W | IRQ12 ~ IRQ15优先级控制寄存器 | 0x0000_0000 |
| **NVIC_IPR4** | SCS_BA + 410 | R/W | IRQ16 ~ IRQ19优先级控制寄存器 | 0x0000_0000 |
| **NVIC_IPR5** | SCS_BA + 414 | R/W | IRQ20 ~ IRQ23优先级控制寄存器 | 0x0000_0000 |
| **NVIC_IPR6** | SCS_BA + 418 | R/W | IRQ24 ~ IRQ27优先级控制寄存器 | 0x0000_0000 |
| **NVIC_IPR7** | SCS_BA + 41C | R/W | IRQ28 ~ IRQ31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 | 偏移量 | R/W | 描述 | 复位的值 |
|---|---|---|---|---|
| NVIC_ISER | SCS _BA + 100 | R/W | IRQ0 ~ IRQ31 设置使能控制寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:0] | SETENA | 使能1个或多个中断，每位代表从IRQ0 ~ IRQ31的中断号(向量号：16 ~ 47).<br>写1使能相关中断<br>写0无效<br>寄存器返回当前使能状态. |

### IRQ0 ~ IRQ31 清使能控制寄存器 (NVIC_ICER）

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| NVIC_ICER | SCS _BA + 180 | R/W | IRQ0 ~ IRQ31 清使能控制寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:0] | CLRENA | 禁止1个或多个中断，每位代表从IRQ0 ~ IRQ31的中断号 (向量号： 16 ~ 47).<br>写1禁止相应中断<br>写0无效<br>寄存器返回当前使能状态 |

### IRQ0 ~ IRQ31 设置挂起控制寄存器（NVIC_ISPR）

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **NVIC_ISPR** | SCS _BA + 200 | R/W | IRQ0 ~ IRQ31 设置挂起控制寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:0] | **SETPEND** | 软件写1，挂起相应中断.每位代表从IRQ0 ~ IRQ31 的中断号(向量号： 16 ~ 47).<br>写0无效<br>寄存器返回当前挂起状态 |

### IRQ0 ~ IRQ31 清挂起控制寄存器（NVIC_ICPR）

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| NVIC_ICPR | SCS _BA + 280 | R/W | IRQ0 ~ IRQ31清挂起控制寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:0] | CLRPEND | 写1清除相应中断挂起，每位代表从IRQ0 ~ IRQ31的中断号 (向量号： 16 ~ 47).<br>写0无效.<br>寄存器返回当前挂起状态. |

### IRQ0 ~ IRQ3 中断优先级寄存器（NVIC_IPR0）

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **NVIC_IPR0** | SCS _BA + 400 | 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** | | **保留** | | | | | |
| **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| **PRI_2** | | **保留** | | | | | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| **PRI_1** | | **保留** | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **PRI_0** | | **保留** | | | | | |

| Bits | 描述 | |
|---|---|---|
| [31:30] | **PRI_3** | IRQ3优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [23:22] | **PRI_2** | IRQ2优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [15:14] | **PRI_1** | IRQ1优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [7:6] | **PRI_0** | IRQ0优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |

### IRQ4 ~ IRQ7中断优先级寄存器（NVIC_IPR1）

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **NVIC_IPR1** | SCS _BA + 404 | R/W | IRQ4 ~ IRQ7中断优先级寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:30] | **PRI_7** | IRQ7优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [23:22] | **PRI_6** | IRQ6优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [15:14] | **PRI_5** | IRQ5优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [7:6] | **PRI_4** | IRQ4优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |

### IRQ8 ~ IRQ11中断优先级寄存器（NVIC_IPR2）

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| NVIC_IPR2 | SCS _BA + 408 | R/W | IRQ8 ~ IRQ11中断优先级寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:30] | **PRI_11** | IRQ11优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [23:22] | **PRI_10** | IRQ10优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [15:14] | **PRI_9** | IRQ9优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [7:6] | **PRI_8** | IRQ8优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |

### IRQ12 ~ IRQ15中断优先级寄存器（NVIC_IPR3）

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| NVIC_IPR3 | SCS _BA + 40C | R/W | IRQ12 ~ IRQ15中断优先级寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:30] | **PRI_15** | IRQ15优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [23:22] | **PRI_14** | IRQ14优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [15:14] | **PRI_13** | IRQ13优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [7:6] | **PRI_12** | IRQ12优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |

### IRQ16 ~ IRQ19中断优先级寄存器（NVIC_IPR4）

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| NVIC_IPR4 | SCS _BA + 410 | R/W | IRQ16 ~ IRQ19中断优先级寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:30] | **PRI_19** | IRQ19优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [23:22] | **PRI_18** | IRQ18优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [15:14] | **PRI_17** | IRQ17优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [7:6] | **PRI_16** | IRQ16优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |

### IRQ20 ~ IRQ23中断优先级寄存器（NVIC_IPR5）

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| NVIC_IPR5 | SCS _BA + 414 | R/W | IRQ20 ~ IRQ23中断优先级寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:30] | **PRI_23** | IRQ23优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [23:22] | **PRI_22** | IRQ22优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [15:14] | **PRI_21** | IRQ21优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [7:6] | **PRI_20** | IRQ20优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |

### IRQ24 ~ IRQ27中断优先级寄存器（NVIC_IPR6）

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| NVIC_IPR6 | SCS _BA + 418 | R/W | IRQ24 ~ IRQ27中断优先级寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:30] | **PRI_27** | IRQ27优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [23:22] | **PRI_26** | IRQ26优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [15:14] | **PRI_25** | IRQ25优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [7:6] | **PRI_24** | IRQ24优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |

### IRQ28 ~ IRQ31中断优先级寄存器（NVIC_IPR7）

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| NVIC_IPR7 | SCS _BA + 41C | R/W | IRQ28 ~ IRQ31中断优先级寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:30] | **PRI_31** | IRQ31优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [23:22] | **PRI_30** | IRQ30优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [15:14] | **PRI_29** | IRQ29优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |
| [7:6] | **PRI_28** | IRQ28优先级<br>“0” 表示最高优先级& “3” 表示最低优先级 |

### 5.2.6.5 中断源控制寄存器

除了NVIC相关的中断控制寄存器外，NUC1xx 系列还有一些特殊控制寄存器执行中断功能，包括“中断源识别”, ”NMI 源选择”与 “中断测试模式”. 描述如下.

**R: 只读, W: 只写, R/W: 可读写, W&C: 写1清零**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| INT_BA = 0x5000_0300 | | | | |
| **IRQ0_SRC** | INT_BA+0x00 | R | MCU IRQ0 (BOD) 中断源识别 | 0xXXXX_XXXX |
| **IRQ1_SRC** | INT_BA+0x04 | R | MCU IRQ1 (WDG) 中断源识别 | 0xXXXX_XXXX |
| **IRQ2_SRC** | INT_BA+0x08 | R | MCU IRQ2 ((EINT0) 中断源识别 | 0xXXXX_XXXX |
| **IRQ3_SRC** | INT_BA+0x0C | R | MCU IRQ3 (EINT1) 中断源识别 | 0xXXXX_XXXX |
| **IRQ4_SRC** | INT_BA+0x10 | R | MCU IRQ4 (GPA/B) 中断源识别 | 0xXXXX_XXXX |
| **IRQ5_SRC** | INT_BA+0x14 | R | MCU IRQ5 (GPC/D/E) 中断源识别 | 0xXXXX_XXXX |
| **IRQ6_SRC** | INT_BA+0x18 | R | MCU IRQ6 (PWM0/2) 中断源识别 | 0xXXXX_XXXX |
| **IRQ7_SRC** | INT_BA+0x1C | R | MCU IRQ7 (PWM1/3) 中断源识别 | 0xXXXX_XXXX |
| **IRQ8_SRC** | INT_BA+0x20 | R | MCU IRQ8 (TMR0) 中断源识别 | 0xXXXX_XXXX |
| **IRQ9_SRC** | INT_BA+0x24 | R | MCU IRQ9 (TMR1) 中断源识别 | 0xXXXX_XXXX |
| **IRQ10_SRC** | INT_BA+0x28 | R | MCU IRQ10 (TMR2) 中断源识别 | 0xXXXX_XXXX |
| **IRQ11_SRC** | INT_BA+0x2C | R | MCU IRQ11 (TMR3) 中断源识别 | 0xXXXX_XXXX |
| **IRQ12_SRC** | INT_BA+0x30 | R | MCU IRQ12 (URT0) 中断源识别 | 0xXXXX_XXXX |
| **IRQ13_SRC** | INT_BA+0x34 | R | MCU IRQ13 (URT1) 中断源识别 | 0xXXXX_XXXX |
| **IRQ14_SRC** | INT_BA+0x38 | R | MCU IRQ14 (SPI0) 中断源识别 | 0xXXXX_XXXX |
| **IRQ15_SRC** | INT_BA+0x3C | R | MCU IRQ15 (SPI1) 中断源识别 | 0xXXXX_XXXX |
| **IRQ16_SRC** | INT_BA+0x40 | R | MCU IRQ16 (SPI2) 中断源识别 | 0xXXXX_XXXX |
| **IRQ17_SRC** | INT_BA+0x44 | R | MCU IRQ17 (SPI3)) 中断源识别 | 0xXXXX_XXXX |
| **IRQ18_SRC** | INT_BA+0x48 | R | MCU IRQ18 (I2C0) 中断源识别 | 0xXXXX_XXXX |
| **IRQ19_SRC** | INT_BA+0x4C | R | MCU IRQ19 (I2C1) 中断源识别 | 0xXXXX_XXXX |
| **IRQ20_SRC** | INT_BA+0x50 | R | MCU IRQ20 (CAN0) 中断源识别 | 0xXXXX_XXXX |
| **IRQ21_SRC** | INT_BA+0x54 | R | MCU IRQ21 (保留) 中断源识别 | 0xXXXX_XXXX |
| **IRQ22_SRC** | INT_BA+0x58 | R | MCU IRQ22 (保留) 中断源识别 | 0xXXXX_XXXX |

| | | | | |
|---|---|---|---|---|
| **IRQ23_SRC** | INT_BA+0x5C | R | MCU IRQ23 (USBD) 中断源识别 | 0xXXXX_XXXX |
| **IRQ24_SRC** | INT_BA+0x60 | R | MCU IRQ24 (PS2) 中断源识别 | 0xXXXX_XXXX |
| **IRQ25_SRC** | INT_BA+0x64 | R | MCU IRQ25 (ACMP) 中断源识别 | 0xXXXX_XXXX |
| **IRQ26_SRC** | INT_BA+0x68 | R | MCU IRQ26 (PDMA) 中断源识别 | 0xXXXX_XXXX |
| **IRQ27_SRC** | INT_BA+0x6C | R | MCU IRQ27 (保留) 中断源识别 | 0xXXXX_XXXX |
| **IRQ28_SRC** | INT_BA+0x70 | R | MCU IRQ28 (PWRWU) 中断源识别 | 0xXXXX_XXXX |
| **IRQ29_SRC** | INT_BA+0x74 | R | MCU IRQ29 (ADC) 中断源识别 | 0xXXXX_XXXX |
| **IRQ30_SRC** | INT_BA+0x78 | R | MCU IRQ30 (保留) 中断源识别 | 0xXXXX_XXXX |
| **IRQ31_SRC** | INT_BA+0x7C | R | MCU IRQ31 (RTC) 中断源识别 | 0xXXXX_XXXX |
| **NMI_SEL** | 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 |

中断源识别寄存器(IRQn_SRC)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| IRQn_SRC | INT_BA+0x00<br>……..<br>INT_BA+0x7C | R | MCU IRQ0 (BOD)中断源识别<br>：<br>MCU IRQ31 (RTC)中断源识别 | 0xXXXX_XXXX |

| 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 | | | | INT_SRC[3] | INT_SRC[2:0] | | |

| 地址 | INT-Num | Bits | 描述 |
|---|---|---|---|
| **INT_BA+0x00** | **0** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: BOD_INT |
| **INT_BA+0x04** | **1** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: WDG_INT |
| **INT_BA+0x08** | **2** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: EINT0 –PB.14上的外部中断0 |
| **INT_BA+0x0C** | **3** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: EINT1 - PB.15上的外部中断1 |
| **INT_BA+0x10** | **4** | [2:0] | Bit2: 1'b0<br>Bit1: GPB_INT<br>Bit0: GPA_INT |
| **INT_BA+0x14** | **5** | [2:0] | Bit2: GPE_INT<br>Bit1: GPD_INT<br>Bit0: GPC_INT |
| **INT_BA+0x18** | **6** | [3:0] | Bit3: PWM3_INT<br>Bit2: PWM2_INT<br>Bit1: PWM1_INT<br>Bit0: PWM0_INT |
| **INT_BA+0x1C** | **7** | [3:0] | Bit3: PWM7_INT<br>Bit2: PWM6_INT<br>Bit1: PWM5_INT<br>Bit0: PWM4_INT |

| | | | |
|---|---|---|---|
| **INT_BA+0x20** | **8** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: TMR0_INT |
| **INT_BA+0x24** | **9** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: TMR1_INT |
| **INT_BA+0x28** | **10** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: TMR2_INT |
| **INT_BA+0x2C** | **11** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: TMR3_INT |
| **INT_BA+0x30** | **12** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: URT0_INT |
| **INT_BA+0x34** | **13** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: URT1_INT |
| **INT_BA+0x38** | **14** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: SPI0_INT |
| **INT_BA+0x3C** | **15** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: SPI1_INT |
| **INT_BA+0x40** | **16** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: SPI2_INT |

| | | | |
|---|---|---|---|
| **INT_BA+0x44** | **17** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: SPI3_INT |
| **INT_BA+0x48** | **18** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: I2C0_INT |
| **INT_BA+0x4C** | **19** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: I2C1_INT |
| **INT_BA+0x50** | **20** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: CAN0_INT |
| **INT_BA+0x54** | **21** | [2:0] | 保留 |
| **INT_BA+0x58** | **22** | [2:0] | 保留 |
| **INT_BA+0x5C** | **23** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: USBD_INT |
| **INT_BA+0x60** | **24** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: PS2_INT |
| **INT_BA+0x64** | **25** | [2:0] | Bit2: 1'b0<br>Bit1: ACMP1 _INT<br>Bit0: ACMP0_INT |
| **INT_BA+0x68** | **26** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: PDMA_INT |
| **INT_BA+0x6C** | **27** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: I2S_INT |

| | | | |
|---|---|---|---|
| **INT_BA+0x70** | **28** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: PWRWU_INT |
| **INT_BA+0x74** | **29** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: ADC_INT |
| **INT_BA+0x78** | **30** | [2:0] | 保留 |
| **INT_BA+0x7C** | **31** | [2:0] | Bit2: 1'b0<br>Bit1: 1'b0<br>Bit0: RTC_INT |

### NMI 中断源选择控制寄存器(NMI_SEL)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **NMI_SEL** | 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 |
|---|---|---|---|---|---|---|---|
| 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** |
| **INT_TEST** | 保留 | | NMI_SEL[4:0] | | | | |

| Bits | 描述 | |
|---|---|---|
| [31:8] | 保留 | 保留 |
| [7] | **INT_TEST** | 中断测试模式 |
| [4:0] | **NMI_SEL** | Cortex-M0的NMI 中断源可以从interrupt[31:0]中选择一个<br>NMI_SEL bit[4:0] 用于选择NMI 中断源 |

### MCU 中断请求源寄存器(MCU_IRQ)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| MCU_IRQ | INT_BA+0x84 | R/W | MCU中断请求源寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:0] | MCU_IRQ | **MCU IRQ Source 寄存器**<br>MCU_IRQ 从外围设备收集所有中断，MCU Cortex-M0产生同步中断，正常模式与测试模式.<br>正常模式： (由NMI_SEL 的 bit [7] = 0来控制)， MCU_IRQ 从外围设备收集所有中断和同步这些中断.<br>测试模式：阻止所有外围设备产生的中断, 此时向MCU送入bit31~bit0.<br>MCU_IRQ[n] 是 “0” : 置 MCU_IRQ[n] 为“1”， 向Cortex_M0 NVIC[n]发生一个中断.<br>MCU_IRQ[n] 是 “1”: (意味着有中断请求) 置位MCU_bit[n]将清中断<br>MCU_IRQ[n]是 “0” :无效 |

### 5.2.7 系统控制寄存器

Cortex™-M0的主要控制与状态特征由系统控制寄存器中的系统控制模块集中管理.

更多详情请参考 “ARM® Cortex™-M0 Technical Reference Manual” 与 “ARM® v6-M Architecture Reference Manual”.

**R: 只读, W: 只写, R/W: 可读写, W&C: 写1清零**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| SCS_BA = 0xE000_E000 | | | | |
| **CPUID** | SCS_BA + D00 | R | CPUID Base 寄存器 | 0x0000_0000 |
| **ICSR** | SCS_BA + D04 | R/W | 中断控制状态寄存器 | 0x0000_0000 |
| **SCR** | SCS_BA + D10 | R/W | 系统控制寄存器 | 0x0000_0000 |
| **SHPR2** | SCS_BA + D1C | R/W | 系统处理器优先级寄存器2 | 0x0000_0000 |
| **SHPR3** | SCS_BA + D20 | R/W | 系统处理器优先级寄存器3 | 0x0000_0000 |

### CPUID Base 寄存器 (CPUID)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **CPUID** | SCS_BA + D00 | R | CPUID Base 寄存器 | 0x 410CC200 |

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

| Bits | 描述 | |
|---|---|---|
| [31:24] | **IMPLEMENTER** | 由ARM分配执行码. ( ARM = 0x41) |
| [23:20] | 保留 | 保留 |
| [19:16] | **PART** | ARMv6-M 值为0xC |
| [15:4] | **PARTNO** | 值为 0xC20. |
| [3:0] | **REVISION** | 值为 0x0 |

### 中断控制状态寄存器(ICSR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| ICSR | SCS_BA + D04 | R/W | 中断控制状态寄存器 | 0x 00000000 |

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

| Bits | R/W | 描述 | |
|---|---|---|---|
| [31] | R/W | **NMIPENDSET** | 设置该位激活NMI,由于NMI是最高优先级，只要一注册，就被激活. 由当前状态读回(1 if Pending, 0 if not). |
| [28] | R/W | **PENDSVSET** | 设置PendSV 中断. 通常用于请求内容切换.由当前状态读回 (1 if Pending, 0 if not). |
| [27] | W | **PENDSVCLR** | 写1 清PendSV 中断. |
| [26] | R/W | **PENDSTSET** | 设置挂起SysTick. 由当前状态读回 (1 if Pending, 0 if not). |
| [25] | W | **PENDSTCLR** | 写1清除挂起SysTick. |
| [23] | R | **ISRPREEMPT** | 如果置位，挂起异常生效，由调试停止状态退出. |
| [22] | R | **ISRPENDING** | 表示外部配置中断是否挂起. |
| [20:12] | R | **VECTPENDING** | 表示最高优先级挂起异常的数目，挂起状态包括内存使能和掩膜寄存器，不包括PRIMASK。值为0时代表没有异常挂起。 |
| [8:0] | R | **VECTACTIVE** | 0: 线程模式<br>value > 1: 当前执行异常处理的数目. |

### 系统控制寄存器 (SCR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| SCR | SCS_BA + D10 | R/W | 系统控制寄存器 | 0x 00000000 |

| 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 | | | SEVONPEND | 保留 | SLEEPDEEP | SLEEPONEXIT | 保留 |

| Bits | 描述 | |
|---|---|---|
| [4] | SEVONPEND | 当使能时，中断由不活动到挂起，包括了唤醒事件（WFE指令）. |
| [2] | SLEEPDEEP | 提示从休眠中唤醒需要较长时间. |
| [1] | SLEEPONEXIT | 设置为1，在异常返回到线程模式时，内核进入休眠状态. |

### 系统处理器优先级寄存器2 (SHPR2)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| SHPR2 | SCS_BA + D1C | R/W | 系统处理器优先级寄存器2 | 0x 00000000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| PRI_11 | | 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 | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [31:30] | **PRI_11** | **系统处理器的优先级11 – SVCall**<br>“0” 表示最高优先级& “3” 表示最低优先级 |

### 系统处理器优先级寄存器3 (SHPR3)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| SHPR3 | SCS_BA + D20 | R/W | 系统处理器优先级寄存器3 | 0x 00000000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:30] | **PRI_15** | **系统处理器优先级15 – SysTick**<br>“0” 表示最高优先级 & “3” 表示最低优先级 |
| [23:22] | **PRI_14** | **系统处理器优先级14 – PendSV**<br>“0” 表示最高优先级 & “3” 表示最低优先级 |

## 5.3 时钟控制器

时钟控制器为芯片提供时钟源. 包括AMBA接口模块和所有外围设备时钟，如USB、UART、ADC、PS2等。该控制器执行电源控制功能, 包括个别时钟的关或开的控制寄存器，时钟源选择和分频. 这些功能可以减少额外的功耗，使得芯片工作在合适的时钟条件下. 置位Power-Down位后，CPU Cortex-M0执行WFI或WFE指令，芯片将进入掉电模式. 在掉电模式下，控制器关闭外部晶振和内部振荡器，以降低功耗到最小.

### 5.3.1 时钟发生器

时钟发生器由如下5个时钟源组成：

- 一个外部 32KHz 晶振
- 一个外部 12MHz 晶振
- 一个可编程的 PLL FOUT(PLL 由 12M 和 22M组成)
- 一个内部 22MHz RC 振荡器
- 一个内部 10KHz 振荡器

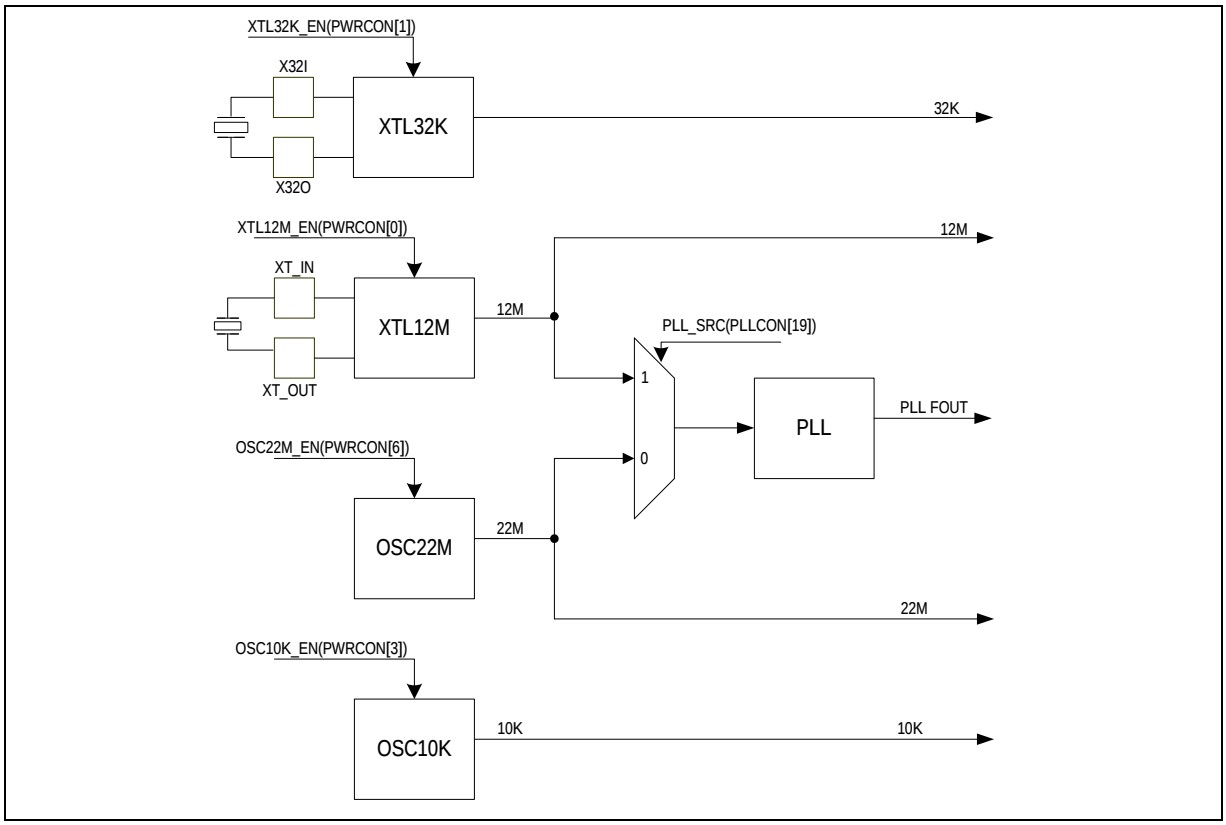



图 5-2 时钟发生器框图

### 5.3.2 系统时钟 & SysTick 时钟

系统时钟有5个时钟源，由时钟发生器产生。时钟源切换取决于寄存器HCLK_S(CLKSEL0[2:0])，如下图.

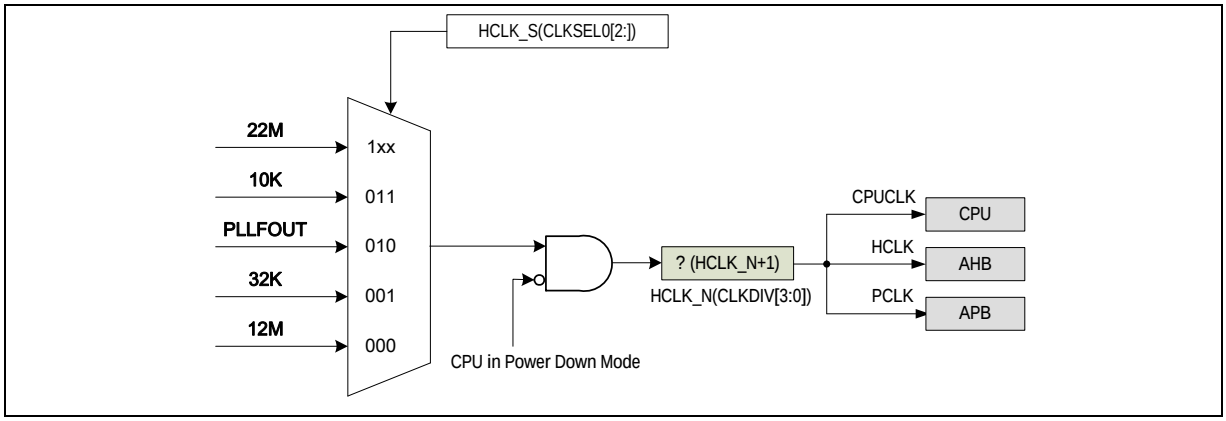



图 5-3 系统时钟框图

SysTick 时钟 (STCLK) 有5个时钟源，由时钟发生器产生。时钟源切换取决于寄存器STCLK_S(CLKSEL0[5:3]. 如下图.

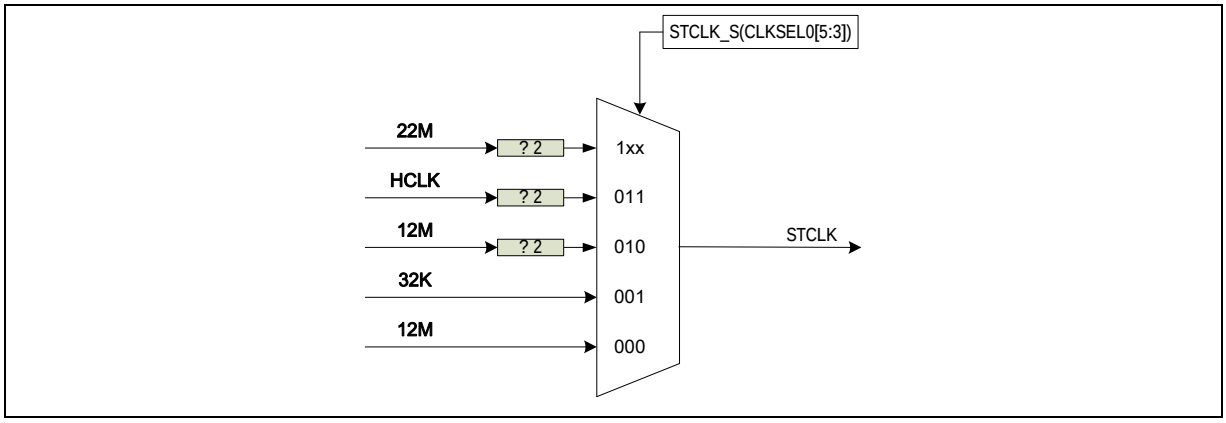



图 5-4 SysTick 时钟控制框图

### 5.3.3 外围设备时钟

不同的外设，其时钟依赖于不同的时钟源切换. 参阅寄存器CLKSEL1 & APBCLK 的描述.

### 5.3.4 掉电模式(深度休眠模式) 时钟

进入掉电模式后，一些时钟源、外设时钟和系统时钟被禁止，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
  - 内部 10K 振荡器时钟
  - 外部 32K 晶振时钟
- 外设时钟 (IP 采用 32K 或10K Hz作时钟源)
  - 看门狗时钟
  - RTC 时钟
  - Timer 0/1/2/3 时钟
  - PWM 时钟
  - ADC 时钟

### 5.3.5 分频器输出

该器件包含分频器，由16级2分频移位寄存器组成. 因此有16种分频选择从 $F_{in}/2^1$ 到 $F_{in}/2^{16}$ ，其中 Fin 为输入到时钟分频器的时钟频率.

输出公式： $F_{out} = F_{in}/2^{(N+1)}$，其中 $F_{in}$ 为输入时钟频率，$F_{out}$ 为时钟分频输出频率，N 为 FSEL(FRQDIV[3:0])中的4位值.

当 FDIV_EN(FRQDIV[4]) 被设为高, 将重置链计数器并开始计数，当FDIV_EN(FRQDIV[4])置0时，链计数器持续计数直到分频时钟达到低状态并停留在低状态.

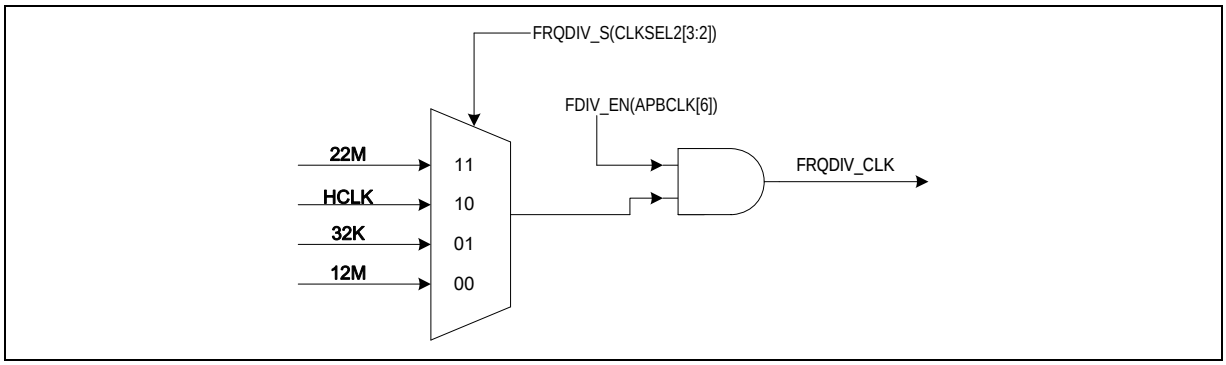



图 5-5 分频器的时钟源

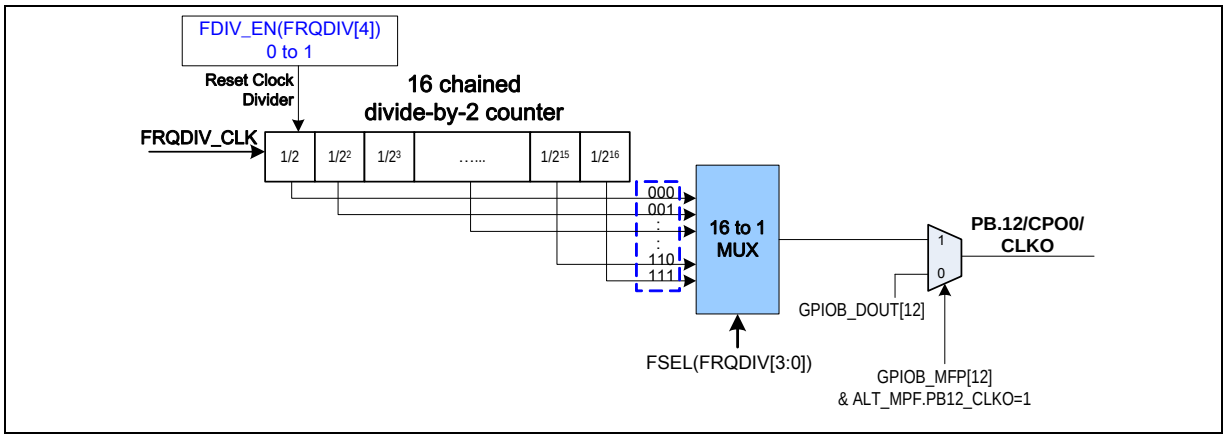



图 5-6 分频器框图

### 5.3.6 时钟控制寄存器列表

**R: 只读, W: 只写, R/W: 可读写**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **PWRCON** | CLK_BA + 00 | R/W | 系统掉电控制寄存器 | 0x0000_001x |
| **AHBCLK** | CLK_BA + 04 | R/W | AHB 设备时钟使能控制寄存器 | 0x0000_0001 |
| **APBCLK** | CLK_BA + 08 | R/W | APB 设备时钟使能控制寄存器 | 0x0000_000x |
| **CLKSEL0** | CLK_BA + 10 | R/W | 时钟源选择控制寄存器0 | 0xFFFF_FFFx |
| **CLKSEL1** | CLK_BA + 14 | R/W | 时钟源选择控制寄存器1 | 0xFFFF_FFFx |
| **CLKSEL2** | CLK_BA + 1C | R/W | 时钟源选择控制寄存器2 | 0xFFFF_FFFx |
| **CLKDIV** | CLK_BA_+ 18 | R/W | 时钟分频数目寄存器 | 0x0000_0000 |
| **PLLCON** | CLK_BA + 20 | R/W | PLL控制寄存器 | 0x0001_C22E |
| **FRQDIV** | GCR_BA + 24 | R/W | 分频器控制寄存器 | 0x0000_0000 |

### 5.3.7 时钟控制寄存器描述

**掉电控制寄存器（PWRCON）**

Ex除BIT[6]外, PWRCON的其他位都受保护, 解锁这些位，需要向0x5000_0100写入“59h”, “16h”, “88h”.
参考寄存器RegLockAddr，其地址是GCR_BA + 0x100

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **PWRCON** | CLK_BA + 00 | R/W | 系统掉电控制寄存器 | 0x0000_001x |

| 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** |
| | | | | | | | **PD_WAIT_CPU** |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **PWR_DOWN** | **PD_WU_STS** | **WINT_EN** | **WU_DLY** | **OSC10K_EN** | **OSC22M_EN** | **XTL32K_EN** | **XTL12M_EN** |

| Bits | 描述 | |
|---|---|---|
| [31:9] | 保留 | Reserve |
| [8] | **PD_WAIT_CPU (1’b0)** | **控制进入掉电模式的条件**<br>1 = 在PWR_DOWN与CPU执行WFE/WFI指令时，芯片进入掉电模式下.<br>0 = PWR_DOWN 置位时，芯片进入掉电模式 |
| [7] | **PWR_DOWN_EN (1’b0)** | **激活或使能系统掉电模式**<br>置 “1”，使能芯片的掉电模式，激活芯片的掉电依赖于PD_WAIT_CPU 位<br>(a) PD_WAIT_CPU 为 “0”,置位PWR_DOWN_EN 后，芯片进入掉电.<br>(b) PD_WAIT_CPU 为 “1”, 在CPU的休眠模式有效时,芯片仍在运行，然后才掉电<br>芯片由掉电唤醒，该位自动清零，在下次掉电时，用户需要重新置位该位.<br>掉电 模式下, LDO, 12M 晶振与22M OSC 被禁止， 32K与10K的使能不受该位控制<br>掉电时, PLL 与所有的AMBA时钟 (HCLKx, CPU clock 与 PCLKx) 也被禁, 忽略时钟源选择. 如果IPclock source从32k或10k时钟，IP engine clock 不受该位控制 (e.g. the RTC from 32K and the WDG from 10K)<br>1 = 芯片立即进入掉电模式 或等待CPU休眠命令<br>0 = 芯片工作于正常模式或CPU进入idle mode(休眠模式)<br>Ps: 参考表格5-2 |

| | | |
|---|---|---|
| [6] | **PD_WU_STS**<br>**(1'b0)** | **芯片掉电唤醒状态标志**<br>设置"掉电唤醒", 从掉电模式恢复<br>GPIO, USB, UART 或 RTC 被唤醒，该标志置位<br>写 0 清该位 |
| [5] | **PD_WU_INT_EN**<br>**(1'b0)** | **掉电模式唤醒的中断使能**<br>0 = 禁止<br>1 = 使能. 从掉电唤醒时，产生中断. |
| [4] | **WU_DLY**<br>**(1'b1)** | 使能唤醒延时计数器. 芯片由idle mode唤醒时，时钟控制会延迟64个时钟周期以等待12M或22M晶振稳定.<br>1 = 使能64个时钟周期的延迟<br>0 = 禁止64个时钟周期的延迟 |
| [3] | **OSC10K_EN**<br>**(1'b1)** | **内部 10KHz 振荡器控制**<br>1 = 使能10KHz 振荡器<br>0 = 禁止10KHz 振荡器 |
| [2] | **OSC22M_EN**<br>**(1'b1)** | **内部 22MHz振荡器控制**<br>1 = 使能22MHz振荡器<br>0 = 禁止22MHz振荡器 |
| [1] | **XTL32K_EN**<br>**(1'b0)** | **外部 32.768 KHz 晶振控制**<br>1 = 使能32.768KHz 晶振 (正常操作)<br>0 = 禁止32.768KHz 晶振 |
| [0] | **XTL12M_EN** | **外部 12MHz晶振控制**<br>该位的缺省值由flash控制器设置，用户配置寄存器config0 [26:24]. 当缺省时钟源为12MHz 晶振. 该 "1"<br>1 = 使能晶振<br>0 = 禁止晶振 |

表 5-5 掉电模式控制表

| 寄存器/指令<br>模式 | PWR_DOWN_EN | PD_WAIT_CPU | CPU 运行<br>WFE/WFI 指令 | 时钟门控 |
|---|---|---|---|---|
| **正常运行模式** | 1'b0 | 1'b0 | NO | 所有时钟根据寄存器配置正常运行 |
| **IDLE 模式**<br>**(CPU 进入休眠模式)** | 1'b0 | 1'b0 | YES | 仅CPU内部时钟运行 |
| **Power_down 模式** | 1'b1 | 1'b0 | NO | 大部分时钟停止运行，仅外部10K/32K及RTC/WDT/Timer/PWM/ADC 可能运行 |

| | | | | |
|---|---|---|---|---|
| **Power_down Mode**<br>**(CPU 进入深度休眠模式)** | 1’b1 | 1’b1 | YES | 大部分时钟停止运行，仅外部 10K/32K 及 RTC/WDT/Timer/PWM/ADC 可能运行. |

### AHB设备时钟使能控制寄存器 （AHBCLK）

该寄存器各位用于使能/禁止AMBA 时钟, AHB engine 和外围设备

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **AHBCLK** | CLK_BA + 04 | R/W | AHB 设备时钟使能控制寄存器 | 0x0000_0001 |

| 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** |
| | | | | | **ISP_EN** | **PDMA_EN** | **保留** |

| Bits | 描述 | |
|---|---|---|
| [31:3] | 保留 | 保留 |
| [2] | **ISP_EN** | **Flash ISP 控制器时钟使能控制.**<br>1 = 使能 the Flash ISP engine 时钟.<br>0 = 禁止the Flash ISP engine 时钟. |
| [1] | **PDMA_EN** | **PDMA 控制器时钟使能控制.**<br>PDMA时钟使能控制<br>1 = 使能 the PDMA engine 时钟.<br>0 = 禁止 the PDMA engine 时钟 |
| [0] | 保留 | 保留 |

### APB设备时钟使能控制寄存器（APBCLK）

这些寄存器各位用于使能/禁止APB engine和外设.

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| APBCLK | CLK_BA + 08 | R/W | APB设备时钟使能控制寄存器 | 0x0000_000x |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| PS2_EN | ACMP_EN | 保留 | ADC_EN | USBD_EN | 保留 | 保留 | CAN0_EN |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| PWM67_EN | PWM45_EN | PWM23_EN | PWM01_EN | 保留 | UART2_EN | UART1_EN | UART0_EN |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| SPI3_EN | SPI2_EN | SPI1_EN | SPI0_EN | 保留 | 保留 | I2C1_EN | I2C0_EN |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | FDIV_EN | TMR3_EN | TMR2_EN | TMR1_EN | TMR0_EN | RTC_EN | WDG_EN |

| Bits | 描述 | |
|---|---|---|
| [31] | **PS2_EN** | **PS2 时钟使能控制.**<br>1 = 使能PS2 APB 和engine 时钟<br>0 = 禁止PS2 APB 和engine 时钟 |
| [30] | **ACMP_EN** | **模拟比较器时钟使能控制.**<br>1 = 使能模拟比较器时钟<br>0 = 禁止模拟比较器时钟 |
| [29] | 保留 | 保留 |
| [28] | **ADC_EN** | **使能ADC时钟控制.**<br>置 “1” 使能ADC’s APB 和engine时钟<br>置 “0” 禁止ADC’s APB 和engine时钟 |
| [27] | **USBD_EN** | **USB FS设备控制器时钟使能控制**<br>置 “1” 使能 USB 设备控制器的PB 和engine时钟<br>置 “0” 禁止USB 设备控制器的PB 和engine时钟 |
| [26] | 保留 | 保留 |
| [25] | 保留 | 保留 |

| | | |
|---|---|---|
| [24] | **CAN0_EN** | **使能CAN 总线控制器0 时钟控制**<br>置 “1”使能CAN 总线控制器的 APB 和 engine 时钟<br>置 “0” 禁止CAN总线控制器的 APB 和 engine 时钟 |
| [23] | **PWM67_EN** | **PWM_67 时钟使能控制.**<br>置“1” 使能PWM67 APB 和engine时钟<br>置“0” 禁止PWM67 APB 和engine 时钟 |
| [22] | **PWM45_EN** | **PWM_45 时钟使能控制.**<br>置“1” 使能PWM45 APB 和engine 时钟<br>置“0” 禁止PWM45 APB 和engine 时钟 |
| [21] | **PWM23_EN** | **PWM_23 时钟使能控制.**<br>置“1” 使能PWM23 APB 和engine 时钟<br>置“0” 禁止PWM23 APB 和engine 时钟 |
| [20] | **PWM01_EN** | **PWM_01 时钟使能控制.**<br>置“1” 使能PWM01 APB 和engine 时钟<br>置“0” 禁止PWM01 APB 和engine时钟 |
| [19] | 保留 | 保留 |
| [18] | **UART2_EN** | **UART2 时钟使能控制.**<br>置“1” 使能UART2 APB 和engine时钟<br>置“0” 禁止UART2 APB 和engine时钟 |
| [17] | **UART1_EN** | **UART1 时钟使能控制.**<br>置“1” 使能UART1 APB 和engine时钟<br>置“0” 禁止UART1 APB 和engine时钟 |
| [16] | **UART0_EN** | **UART0 时钟使能控制.**<br>置“1” 使能UART0 APB 和engine时钟<br>置“0” 禁止UART0 APB 和engine时钟 |
| [15] | **SPI3_EN** | **SPI3 时钟使能控制.**<br>0 = 禁止<br>1 = 使能 |
| [14] | **SPI2_EN** | **SPI2 时钟使能控制.**<br>0 = 禁止<br>1 = 使能 |

| | | |
|---|---|---|
| [13] | **SPI1_EN** | **SPI1 时钟使能控制.**<br>0 = 禁止<br>1 = 使能 |
| [12] | **SPI0_EN** | **SPI0 时钟使能控制.**<br>0 = 禁止<br>1 = 使能 |
| [11:10] | 保留 | |
| [9] | **I2C1_EN** | **I2C1 时钟使能控制.**<br>0 = 禁止<br>1 = 使能 |
| [8] | **I2C0_EN** | **I2C0 时钟使能控制.**<br>0 = 禁止<br>1 = 使能 |
| [7] | 保留 | |
| [6] | **FDIV_EN** | **分频器输出时钟使能控制**<br>0 = 禁止<br>1 = 使能 |
| [5] | **TMR3_EN** | **Timer3 时钟使能控制**<br>0 = 禁止<br>1 = 使能 |
| [4] | **TMR2_EN** | **Timer2 时钟使能控制**<br>0 = 禁止<br>1 = 使能 |
| [3] | **TMR1_EN** | **Timer1 时钟使能控制**<br>0 = 禁止<br>1 = 使能 |
| [2] | **TMR0_EN** | **Timer0 时钟使能控制**<br>0 = 禁止<br>1 = 使能 |

| | | |
|---|---|---|
| [1] | **RTC_EN** | **Real-Time-Clock APB 接口时钟控制.**<br>该位仅用于控制RTC APB时钟，RTC时钟源由 32768Hz 晶振提供.<br>0 = 禁止<br>1 = 使能 |
| [0] | **WDCLK_EN** | **Watch Dog 时钟使能.**<br>该位是受保护的位，编程时，需要向0x5000_0100 依次写入“59h”, “16h”, “88h” 来解锁定，参考寄存器RegLockAddr<br>缺省值由flash控制设置，用户可配置寄存器congig0 bit[31]<br>0 = 禁止<br>1 = 使能 |

### 时钟源选择控制寄存器0 （CLKSEL0）

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **CLKSEL0**[1] | CLK_BA + 10 | R/W | 时钟源选择控制寄存器0 | 0xFFFF_FFFx[2] |

| 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 | | STCLK_S | | | HCLK_S | | |

| Bits | 描述 | |
|---|---|---|
| [31:6] | 保留 | **保留** |
| [5:3] | **STCLK_S** | **MCU Cortex_M0 SysTick 时钟源选择.**<br>受保护位，编程时，需要向0x5000_0100 依次写入“59h”, “16h”, “88h” 来解锁定，参考寄存器RegLockAddr 为GCR_BA + 0x100<br>000 = 保留<br>001 = 外部32KHz晶振<br>010 = 12MHz 晶振1/2分频<br>011 = HCLK**/2**<br>1xx = 内部 22MHz 振荡器 1/2分频 |

| | | |
|---|---|---|
| [2:0] | **HCLK_S** | **HCLK 时钟源选择.**<br>**注:**<br>1. 在时钟切换到相关时钟源(pre-select and new-select)时必须打开<br>2. 任何复位后，加载用户配置寄存器**CFOSC(Config0[26:24])的值，缺省值可为** 000b 或 111b.<br>3. 受保护位，编程时，需要向0x5000_0100 依次写入“59h”, “16h”, “88h” 来解锁定，参考寄存器RegLockAddr 为GCR_BA + 0x100<br>00 = 外部12MHz 晶振<br>001 = 外部32KHz 晶振<br>010 = PLL 时钟<br>011 = 内部 10KHz 振荡器时钟<br>1xx = 内部 22MHz振荡器时钟 |

### 时钟源选择控制寄存器1（CLKSEL1）

在时钟切换到相关的时钟源前，必须打开 (预选和新选).

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| CLKSEL1 | CLK_BA + 14 | R/W | 时钟源选择控制寄存器1 | 0xFFFF_FFFx |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| PWM23_S | | PWM01_S | | CAN_S | | UART_S | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | TMR3_S | | | R | TMR2_S | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| R | TMR1_S | | | R | TMR0_S | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | | | ADC_S | | WDG_S | |

| Bits | 描述 | |
|---|---|---|
| [31:30] | **PWM23_S** | **PWM3 与 PWM2的时钟源选择.**<br>PWM3 与 PWM2使用相同的时钟源 和相同的分频<br>00 = 外部12MHz 晶振<br>01 = 外部 32KHz 晶振<br>10 = HCLK<br>11 = 内部 22MHz 振荡器 |
| [29:28] | **PWM01_S** | **PWM1 与 PWM0的时钟源选择.**<br>PWM1 与 PWM0使用相同的时钟源 和相同的分频<br>00 = 外部12MHz 晶振<br>01 = 外部 32KHz 晶振<br>10 = HCLK<br>11 = 内部 22MHz 振荡器 |
| [27:26] | **CAN_S** | **CAN时钟源选择**<br>00 =外部12MHz 晶振<br>01 = PLL<br>1x =内部 22MHz 振荡器 |

| | | |
|---|---|---|
| [25:24] | **UART_S** | **UART时钟源选择.**<br>00 =外部12MHz 晶振<br>01 = PLL<br>1x =内部 22MHz 振荡器 |
| [23] | 保留 | 保留 |
| [22:20] | **TMR3_S** | **TIMER3 时钟源选择.**<br>000 =外部12MHz 晶振<br>001 =外部 32KHz 晶振<br>010 = HCLK<br>011 = 外部触发时钟<br>1xx =内部 22MHz 振荡器 |
| [19] | 保留 | 保留 |
| [18:16] | **TMR2_S** | **TIMER2时钟源选择.**<br>000 =外部12MHz 晶振<br>001 =外部 32KHz 晶振<br>010 = HCLK<br>011 = 外部触发时钟<br>1xx =内部 22MHz 振荡器 |
| [15] | 保留 | **保留** |
| [14:12] | **TMR1_S** | **TIMER1 时钟源选择.**<br>000 =外部12MHz 晶振<br>001 =外部 32KHz 晶振<br>010 = HCLK<br>011 = 外部触发时钟<br>1xx =内部 22MHz 振荡器 |
| [11] | 保留 | **保留** |
| [10:8] | **TMR0_S** | **TIMER0 时钟源选择.**<br>000 =外部12MHz 晶振<br>001 =外部 32KHz 晶振<br>010 = HCLK<br>011 = 外部触发时钟<br>1xx =内部 22MHz 振荡器 |
| [7:4] | 保留 | 保留 |
| [3:2] | **ADC_S** | **ADC 时钟源选择.**<br>00 =外部12MHz 晶振<br>01 = PLL<br>1x =内部 22MHz 振荡器 |

| | | |
|---|---|---|
| [1:0] | **WDG_S** | **WDG CLK 时钟源选择.**<br>受保护位，编程时，需要向0x5000_0100 依次写入“59h”, “16h”, “88h” 来解锁定，参考寄存器RegLockAddr为GCR_BA + 0x100<br>00 = 保留<br>01 =外部 32KHz 晶振<br>10 = HCLK/2048时钟<br>11 = 内部 10KHz振荡器时钟 |

### 时钟源选择控制寄存器2（CLKSEL2）

在时钟切换到相关的时钟源前，必须打开 (预选和新选).

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| CLKSEL2 | CLK_BA + 1C | R/W | 时钟源选择控制寄存器2 | 0xFFFF_FFFx |

| 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 |
| PWM67_S | | PWM45_S | | FRQDIV_S | | | |

| Bits | 描述 | |
|---|---|---|
| [31:8] | 保留 | 保留 |
| [7:6] | **PWM67_S** | **PWM6 与 PWM7的时钟源选择**.<br>– PWM6 与 PWM7使用相同的时钟源 和相同的分频<br>00 = 外部12MHz 晶振<br>01 = 外部 32KHz 晶振<br>10 = HCLK<br>11 = 内部 22MHz 振荡器 |
| [5:4] | **PWM45_S** | **PWM4 与 PWM5的时钟源选择**.<br>– PWM4 与 PWM5使用相同的时钟源 和相同的分频<br>00 = 外部12MHz 晶振<br>01 = 外部 32KHz 晶振<br>10 = HCLK<br>11 = 内部 22MHz 振荡器 |
| [3:2] | **FRQDIV_S** | **时钟分频器时钟源选择**<br>00 = 外部12MHz 晶振<br>01 = 外部 32KHz 晶振<br>10 = HCLK<br>11 = 内部 22MHz 振荡器 |

| | | |
|---|---|---|
| [1:0] | 保留 | 保留 |

### 时钟分频寄存器(CLKDIV)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **CLKDIV** | CLK_BA_+ 18 | R/W | 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** |
| ADC_N | | | | | | | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| CAN_N | | | | UART_N | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| USB_N | | | | HCLK_N | | | |

| Bits | 描述 | |
|---|---|---|
| [31:24] | 保留 | 保留 |
| [23:16] | **ADC_N** | ADC时钟频率=ADC时钟源频率/（ADC_N + 1） |
| [15:12] | **CAN_N** | APU 时钟频率 = (CAN 时钟源频率 ) / (CAN_N + 1) |
| [11:8] | **UART_N** | UART时钟频率 = (UART时钟源频率 ) / (UART_N + 1) |
| [7:4] | **USB_N** | USB 时钟频率 = (PLL 频率 ) / (USB_N + 1) |
| [3:0] | **HCLK_N** | HCLK 时钟频率 = (HCLK时钟源频率) / (HCLK_N + 1) |

### PLL 控制寄存器 （PLLCON）

The PLL的参考时钟源来12MHz的外部时钟输入或22MHz的内部振荡器，该寄存器用于控制PLL的输出频率和PLL的操作模式

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| PLLCON | CLK_BA + 20 | R/W | PLL 控制寄存器 | 0x0001_C22E |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 保留 | | | | PLL_SRC | OE | BP | PD |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| OUT_DV | | IN_DV | | | | | FB_DV |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| FB_DV | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [19] | PLL_SRC | **PLL时钟源选择**<br>0 = PLL 时钟源为22MHz 振荡器<br>1 = PLL时钟源为12MHz 的晶振 |
| [18] | OE | **PLL OE (FOUT enable)引脚控制**<br>0 = 使能 PLL FOUT<br>1 = PLL FOUT 为低 |
| [17] | BP | **PLL 旁路控制**<br>0 = PLL 正常模式 (default)<br>1 = PLL 时钟输出与时钟输入相同(XTALin) |
| [16] | PD | **掉电模式.**<br>设置PWRCON的IDLE位为”1”, PLL进入掉电模式<br>0 = PLL正常模式 (default)<br>1 = PLL 掉电模式 |
| [15:14] | OUT_DV | PLL 输出分频控制引脚 (PLL_OD[1:0]) |
| [13:9] | IN_DV | PLL输入分频控制引脚(PLL_R[4:0]) |

| | | |
|---|---|---|
| [8:0] | **FB_DV** | PLL 反馈分频控制引脚 (PLL_F[8:0]) |

**输出时钟频率设置**

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

约束条件:

1. $3.2MHz < FIN < 150MHz$
2. $800KHz < \frac{FIN}{2 * NR} < 8MHz$
3. $100MHz < FCO = FIN \times \frac{NF}{NR} < 200MHz$

   $120MHz < FCO$ is preferred

| *符号* | 说明 |
|---|---|
| *FOUT* | 输出时钟频率 |
| *FIN* | 输入 (参考) 时钟频率 |
| *NR* | 输入分频 (IN_DV + 2) |
| *NF* | 反馈分频 (FB_DV + 2) |
| *NO* | OUT_DV = “00” : NO = 1<br>OUT_DV = “01” : NO = 2<br>OUT_DV = “10” : NO = 2<br>OUT_DV = “11” : NO = 4 |

**Default Freq. Setting**

The default value : 0xC22E
FIN = 12Mhz
NR = (1+2) = 3
NF = (46+2) = 48
NO = 4
*FOUT* = 12/4 x 48 x 1/3 = 48Mhz

### 频率分频器控制寄存器(FRQDIV)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **FRQDIV** | CLK_BA+ 24 | R/W | 频率分频控制寄存器 | 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 | | | FDIV_EN | FSEL | | | |

| Bits | 描述 | |
|---|---|---|
| [31:5] | 保留 | 保留 |
| [4] | **FDIV_EN** | **频率分频器使能位**<br>0 = 禁止频率分频<br>1 = 使能频率分频 |
| [3:0] | **FSEL** | **分频器输出频率选择位**<br>输出频率的公式是<br>$F_{out} = F_{in}/2^{(N+1)}$,<br>$F_{in}$ 为输入时钟频率, $F_{out}$ 为分频器输出时钟频率, N 为FSEL[3:0]的值. |

## 5.4 USB 设备控制器

### 5.4.1 概述

NUC1xx 系列包含一组全速USB 2.0 设备控制器和收发器. 符合USB 2.0规范，支持control/bulk/interrupt/isochronous 传输类型.

在该设备控制器里，包含两个主接口：由USB PHY收发器出来的AHB总线与USB总线. AHB总线, 仅在实现从接口和CPU通过它编程控制寄存器。在该控制器内置有512字节的SRAM作为数据缓存。IN或OUT传输, 需要通过AHB从接口或SIE向SRAM写数据或从SRAM读数据. 用户需要通过BUFSEGx为每个端点缓存设置有效的SRAM地址.

USB设备控制器具有6个可配置的端点，每个端点需要正确配置，每个端点的长度在MXPLDx 中被定义。大多数的主机与设备的握手由硬件来处理。用户仅需要检查相关的USB事件所触发的中断标志寄存器(EVF)，来答应是哪种事件发生，并将请求数据送入缓存，这些数据由硬件送入主机。

软件禁用功能与适用于USB设备，用于模拟设备没有与主机相连的情况.

### 5.4.2 特征

下面是USB的一些特征.

- 遵循 全速USB2.0 设备 (12Mbps)
- 1个中断向量，4个中断事件.
- 支持Control/Bulk/Interrupt/Isochronous传输类型.
- 支持暂停功能，在没有总线活动时，暂停3ms.
- 为可配置的Control/Bulk/Interrupt/Isochronous传输类型提供6个端点和最大512字节的缓存.
- 提供远程唤醒功能.

### 5.4.3 框图

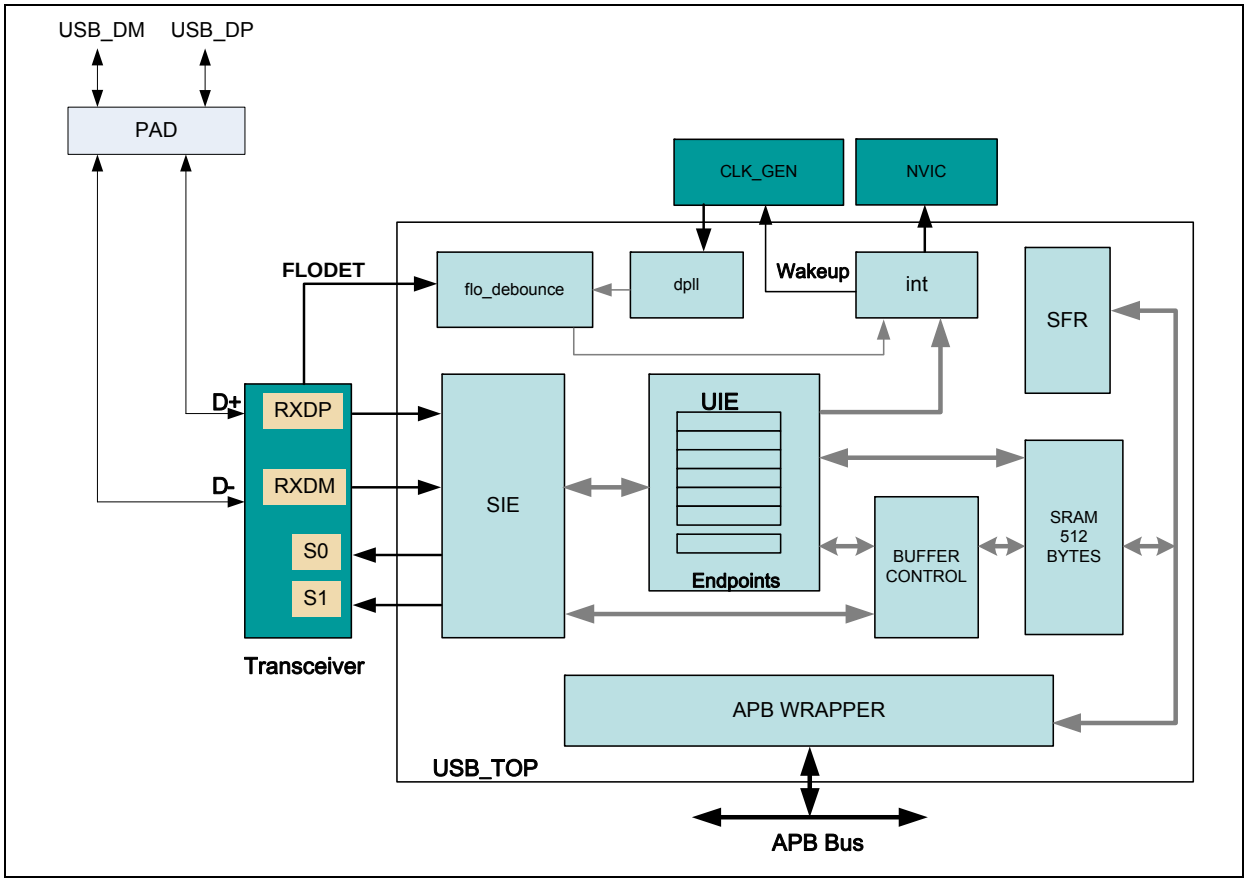



图 5-7 USB 框图

### 5.4.4 功能描述

#### *5.4.4.1 SIE (Serial Interface Engine)*

SIE是设备控制器的前端，处理最多的USB 协议包. 处理功能包括：

- 包识别，处理时序
- SOP, EOP, RESET, RESUME信号检测/产生
- Clock/Data分离
- NRZI 数据编解码与比特填塞（bit-stuffing）
- CRC产生和校验 (for Token and Data)
- Packet ID (PID) 产生和校验/解码
- 串-并/并-串转换

#### *5.4.4.2 UIE*

UIE 是端点管理. 包括Control, Bulk, Interrupt and Isochronous 在内的所有操作都在UIE中执行.

#### *5.4.4.3 数字锁相环DPLL（Digital Phase Lock Loop ）*

USB数据的比特率为12MHz, DPLL采用的48MHz的频率由时钟控制器产生，锁定RXDP与RXDM的输入数据，12MHz的比特率时钟也是由DPLL转换来的.

#### *5.4.4.4 浮动防反弹*

USB设备可以进行热插拔操作，为了监测USB设备从USB分离的状态，设备控制器提供了硬件防反弹以防止在USB插拔时产生的反弹问题，浮动检测中断产生于USB进行插拔操作的10ms后，用户可以通过读取“FLODET”寄存器的值，来答应USB插拔，可推定“FLODET”代表在当前总线上没有反弹，若用户要通过这个标志来检测USB的状态，则需要添加软件防反弹功能.

对于浮动检测功能，加一颗下拉电阻以避免不可预见的一些噪声.

#### *5.4.4.5 中断*

该USB提供1个中断信号和4个中断事件（WAKEUP, FLO, USB, BUS）. WAKEUP 中断仅用于停止唤醒, FLO 中断用于 USB 插拔, USB 事件告之用户一些USB请求,如IN ACK，OUT ACK等, BUS事件告之用户一些总线事件，如 suspend, resume 等. 用户比较在NVIC和IEF里设置相应的位以使能USB中断.

Wakeup 中断仅出现在停止唤醒之后. 在芯片进入掉电模式时，D+，D-和浮动检测脚上的任何变化都能唤醒芯片（假定USB唤醒功能使能时）. 若有不希望看到的变化在浮动脚上引起中断，在USB唤醒后，中断持续超过20MS.

USB 中断告之用户所有总线上的USB事件，用户可以读特殊功能寄存器“STS” 和 “EPTF” 来获知是哪类请求，到哪个端点和采取响应.

与USB中断相似，总线中断告之用户一些总线事件，如USB reset, suspend, time-out, and resume.，用户可以读特殊功能寄存器“ATTR”获取总线事件.

#### *5.4.4.6 省电*

当芯片进入掉电模式时，USB 自动关闭PHY省电，此外，在特殊环境下，用户可以给特殊功能寄存器ATTR［4］写入“0”关闭PHY进入省电状态

### 5.4.5 与USB外设通信处理

用户可以采用中断或检测USB_EVF来监测USB通信，在USB通信发生时，USB_EVF 由硬件置1，并向CPU发送中断请求，在没有中断时，用户可以检测USB_EVF来获取事件，以下是控制流的中断使能.

USB主机向设备控制器请示数据时，用户需要准备相关的数据给端点缓存. 在缓冲请求数据后，用户需要写入实际数据长度给MAXPLD寄存器. 一旦这个寄存器被写入数据，主机会发送IN标志和内部信号“In_Rdy”，“In_Rdy ”会由硬件自动重申.

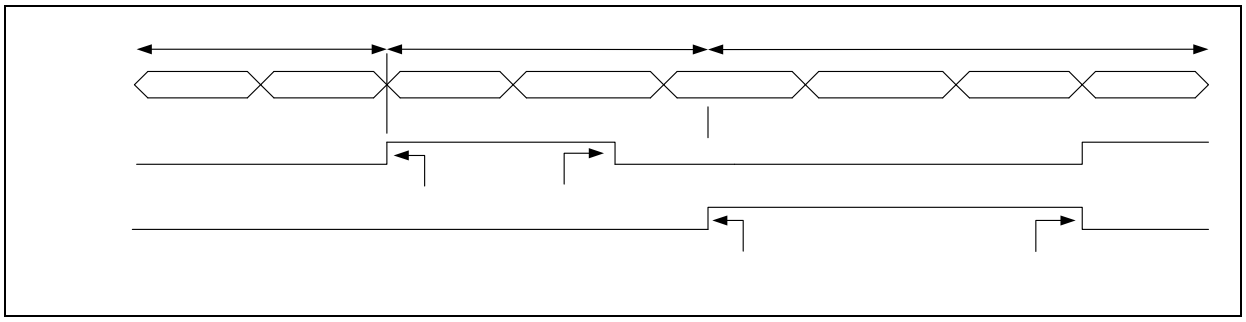


图 5-8 数据传入时间里传输流程图

USB主机要发送数据到设备控制器的OUT端点，硬件将这些数据存在指定的端点缓存里，通信完成后，硬件在MAXPLD记录数据长度，并发送“Out_Rdy”信号，这避免硬件在用户没有取走当前数据时接收下一个数据。一旦用户占用这个通信时，相关的寄存器“MAXPLD”由固件写入“Out_Rdy” 以再次接收下一次通信.

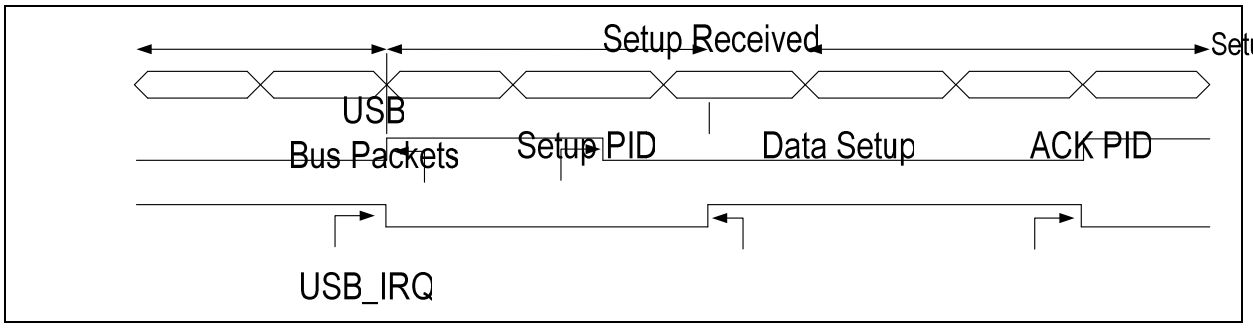



图 5-9 数据输出图

Set by Hardware Clear by Firmwa

In_Rdy

### 5.4.6 内存图

| 地址 | 大小 | 类型 | 描述 |
|---|---|---|---|
| 4006_0000h ~ 4006_00FFh | 256 Bytes | 寄存器 | 特殊功能寄存器 |
| 4006_0100h ~ 4006_02FFh | 512 Bytes | SRAM | USB缓存 |

**R: 只读, W: 只写, R/W: 可读写, W&C: 写1清零**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **USB_BA = 0x4006_0000** | | | | |
| **IEF** | USB_BA+0x000 | W<br>R/W | 中断使能标志 | 0x0000_0000 |
| **EVF** | USB_BA+0x004 | R/W&C | 中断事件标志 | 0x0000_0000 |
| **FADDR** | USB_BA+0x008 | R/W | 功能地址 | 0x0000_0000 |
| **STS** | USB_BA+0x00C | R | 系统状态 | 0x0000_00x0 |
| **ATTR** | USB_BA+0x010 | R/W<br>R | 总线状态 &归属 | 0x0000_0040 |
| **FLODETB** | USB_BA+0x014 | R | 浮动检测 | 0x0000_0000 |
| **BUFSEG** | USB_BA+0x018 | R/W | 缓存分割 | 0x0000_0000 |
| **BUFSEG0** | USB_BA+0x020 | R/W | 端点0的缓存分割 | 0x0000_0000 |
| **MXPLD0** | USB_BA+0x024 | R/W | 端点0的最大有效载荷 | 0x0000_0000 |
| **CFG0** | USB_BA+0x028 | R/W | 端点0的配置 | 0x0000_0000 |
| **CFGP0** | USB_BA+0x02C | R/W<br>W&C | 端点0的延迟控制寄存器与in/out准备清标志 | 0x0000_0000 |
| **BUFSEG1** | USB_BA+0x030 | R/W | 端点1的缓存分割 | 0x0000_0000 |
| **MXPLD1** | USB_BA+0x034 | R/W | 端点1的最大有效载荷 | 0x0000_0000 |
| **CFG1** | USB_BA+0x038 | R/W | 端点1的配置 | 0x0000_0000 |
| **CFGP1** | USB_BA+0x03C | R/W<br>W&C | 端点1的延迟控制寄存器与in/out准备清标志 | 0x0000_0000 |
| **BUFSEG2** | USB_BA+0x040 | R/W | 端点2的缓存分割 | 0x0000_0000 |
| **MXPLD2** | USB_BA+0x044 | R/W | 端点2的最大有效载荷 | 0x0000_0000 |
| **CFG2** | USB_BA+0x048 | R/W | 端点2的配置 | 0x0000_0000 |

| | | | | |
|---|---|---|---|---|
| **CFGP2** | USB_BA+0x04C | R/W<br>W&C | 端点2的延迟控制寄存器与in/out准备清标志 | 0x0000_0000 |
| **BUFSEG3** | USB_BA+0x050 | R/W | 端点3的缓存分割 | 0x0000_0000 |
| **MXPLD3** | USB_BA+0x054 | R/W | 端点3的最大有效载荷 | 0x0000_0000 |
| **CFG3** | USB_BA+0x058 | R/W | 端点3的配置 | 0x0000_0000 |
| **CFGP3** | USB_BA+0x05C | R/W<br>W&C | 端点3的延迟控制寄存器与in/out准备清标志 | 0x0000_0000 |
| **BUFSEG4** | USB_BA+0x060 | R/W | 端点4的缓存分割 | 0x0000_0000 |
| **MXPLD4** | USB_BA+0x064 | R/W | 端点4的最大有效载荷 | 0x0000_0000 |
| **CFG4** | USB_BA+0x068 | R/W | 端点4的配置 | 0x0000_0000 |
| **CFGP4** | USB_BA+0x06C | R/W<br>W&C | 端点4的延迟控制寄存器与in/out准备清标志 | 0x0000_0000 |
| **BUFSEG5** | USB_BA+0x070 | R/W | 端点5的缓存分割 | 0x0000_0000 |
| **MXPLD5** | USB_BA+0x074 | R/W | 端点5的最大有效载荷 | 0x0000_0000 |
| **CFG5** | USB_BA+0x078 | R/W | 端点5的配置 | 0x0000_0000 |
| **CFGP5** | USB_BA+0x07C | R/W<br>W&C | 端点5的In准备清标志 | 0x0000_0000 |
| **DRVSE0** | USB_BA+0x090 | R/W | 置“1”时，使USB PHY驱动SE0 | 0x0000_0001 |
| **PDMA** | USB_BA+0x0A4 | R/R | PDMA控制 寄存器 | 0x0000_0000 |

### 5.4.7 USB控制寄存器描述

**中断使能寄存器(IEF)**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| IEF | USB_BA+0x000 | W, R/W | 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 |
| INNAKen | 保留 | | | | | | wakeupEn |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | | | wakeup | fld | usb | bus |

| Bits | 描述 | | |
|---|---|---|---|
| [15] | **INNAKen** | W R/W | 1/0 在接到IN标志后，设备响应NAK，使能/禁止中断 |
| [8] | **wakeupEN** | R/W | 1/0 使能/禁止USB 唤醒功能 |
| [3] | **wakeup** | R/W | 1/0 使能/禁止 唤醒功能. |
| [2] | **fld** | R/W | 1/0使能/禁止 浮动检测中断 |
| [1] | **usb** | R/W | 1/0 使能/禁止USB事件中断 |
| [0] | **bus** | R/W | 1/0使能/禁止BUS事件中断. |

### 中断事件标志寄存器(EVF)

该寄存器是USB中断事件的标志寄存器，通过读STS，ATTR或FLODET清零.

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| EVF | USB_BA+0x004 | R/W&C | 中断事件标志 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| Setup | 保留 | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 保留 | | EPTF5 | EPTF4 | EPTF3 | EPTF2 | EPTF1 | EPTF0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| 保留 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | | | wakeup | fld | usb | bus |

| Bits | 描述 | | |
|---|---|---|---|
| [31] | **Setup** | R/W&C | 1 = Setup事件发生，向EVF[31]写1清标志. |
| [21] | **EPTF5** | R/W&C | 1 = USB 事件发生，通过STS[25:23]可知是哪类USB事件发生，向EVF[21] 或 EVF[1]写1清标志. |
| [20] | **EPTF4** | R/W&C | 1 = USB 事件发生，通过STS[22:20]可知是哪类USB事件发生，向EVF[20] 或 EVF[1]写1清标志. |
| [19] | **EPTF3** | R/W&C | 1 = USB 事件发生，通过STS[19:17]可知是哪类USB事件发生，向EVF[19] 或 EVF[1]写1清标志. |
| [18] | **EPTF2** | R/W&C | 1 = USB事件发生，通过STS[16:14]可知是哪类USB事件发生，向EVF[18] 或 EVF[1]写1清标志. |
| [17] | **EPTF1** | R/W&C | 1 = USB事件发生，通过STS[13:11]可知是哪类USB事件发生，向EVF[17] 或 EVF[1]写1清标志 |
| [16] | **EPTF0** | R/W&C | 1 = USB事件发生，通过STS[10:8]可知是哪类USB事件发生，向EVF[16] 或 EVF[1]写1清标志 |
| [3] | **wakeup** | R/W&C | 唤醒事件发生，向EVF［3］写1 清标志 |
| [2] | **fld** | R/W&C | 浮动检测事件发生，向EVF［2］写1 清标志 |
| [1] | **usb** | R/W&C | USB 事件发生，通过STS[6:4]可知是哪类USB事件发生，向EVF[1]写1或EPTF0~5 和Setup =0清标志. |
| [0] | **bus** | R/W&C | Bus 事件发生，通过ATTR[3:0]可知是哪类Bus事件发生，向EVF[0]写1清标志. |

### 功能地址寄存器(FADDR)

在USB总线上作为设备地址的七位值.

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **FADDR** | USB_BA+0x008 | R/W | 功能地址 | 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 | FADDR | | | | | | |

| Bits | 描述 | | |
|---|---|---|---|
| [6:0] | **FADDR** | W/R | USB设备的功能地址 |

### 系统状态寄存器 (STS)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **STS** | USB_BA+0x00C | R | 系统状态 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| 保留 | | | | | | STS5 | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| STS5 | STS4 | | | STS3 | | | STS2 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| STS2 | | STS1 | | | STS0 | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Overrun | 保留 | | | | | | |

| Bits | 描述 | | |
|---|---|---|---|
| [25:23] | **STS5** | R | **端点5的系统状态**<br>000 = In ACK<br>001 = In NAK<br>010 = Out 0 ACK<br>110 = Out 1 ACK<br>011 = Setup ACK<br>111 = 同步传输结束 |
| [22:20] | **STS4** | R | **端点4的系统状态**<br>000 = In ACK<br>001 = In NAK<br>010 = Out 0 ACK<br>110 = Out 1 ACK<br>011 = Setup ACK<br>111 =同步传输结束 |
| [19:17] | **STS3** | R | **端点3的系统状态**<br>000 = In ACK<br>001 = In NAK<br>010 = Out 0 ACK<br>110 = Out 1 ACK<br>011 = Setup ACK<br>111 =同步传输结束 |

| | | | |
|---|---|---|---|
| [16:14] | **STS2** | R | **端点2的系统状态**<br>000 = In ACK<br>001 = In NAK<br>010 = Out 0 ACK<br>110 = Out 1 ACK<br>011 = Setup ACK<br>111 =同步传输结束 |
| [13:11] | **STS1** | R | **端点1的系统状态**<br>000 = In ACK<br>001 = In NAK<br>010 = Out 0 ACK<br>110 = Out 1 ACK<br>011 = Setup ACK<br>111 =同步传输结束 |
| [10:8] | **STS0** | R | **端点0的系统状态**<br>000 = In ACK<br>001 = In NAK<br>010 = Out 0 ACK<br>110 = Out 1 ACK<br>011 = Setup ACK<br>111 =同步传输结束 |
| [7] | **Overrun** | R | 输出数据超过最大有效载荷或安装数据超过8字节 |
| [6:0] | 保留 | - | Obsolete |

总线状态&归属寄存器 (ATTR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| ATTR | USB_BA+0x010 | R/W, R | 总线状态&归属寄存器 | 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 | | | | | HSIZE_8 | PDB | DPpuen |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| enUSB | 保留 | RWakeup | enPHY | Timeout | Resume | Suspend | usbRST |

| Bits | 描述 | | |
|---|---|---|---|
| [10] | HSIZE_8 | R/W | CPU 取USB RAM大小选择<br>0: 由CPU传递到USB RAM的大小为Word.<br>1: 由CPU传递到USB RAM的大小为Byte. |
| [9] | PDB | R/W | 掉电 PHY, 低电平有效<br>0 = 关闭 PHY相关电路<br>1 = 打开 PHY相关电路 |
| [8] | DPpuen | R/W | 使能 DP上拉电阻<br>0 = 禁止<br>1 = 使能 |
| [7] | enUSB | R/W | 0 = 禁止 USB<br>1 = 使能 USB |
| [5] | RWakeUp | R/W | 1 =使USB总线处于K状态，以便远程唤醒. |
| [4] | enPHY | R/W | 0 = 禁止 PHY (Suspend state, 管理器在这种状态下仍起作用)<br>1 = 使能 PHY |
| [3] | Timeout | R | 无响应超过18位时间（No response more than 18 bits time） |
| [2] | Resume | R | 从暂停中恢复 |
| [1] | Suspend | R | Bus空闲超过3mS,表示 要么电缆没连接，要么主机正在休眠. |

| | | | |
|---|---|---|---|
| [0] | **usbRST** | R | SE0(single-ended 0) 超过2.5uS，总线复位. |

### 悬空检测寄存器 (FLODETB)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **FLODETB** | USB_BA+0x014 | R | 悬空检测 | 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 | | | | | | | **FLODETB** |

| Bits | 描述 | | |
|---|---|---|---|
| [0] | **FLODETB** | R | 1/0: 连接/悬空 |

### 缓冲分割寄存器(BUFSEG)

仅用于标记设置.

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **BUFSEG** | USB_BA+0x018 | R/W | 缓冲分割 | 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 | | | | | | | **BUFSEG** |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **BUFSEG** | | | | | 保留 | | |

| Bits | 描述 | | |
|---|---|---|---|
| [8:3] | **BUFSEG** | R/W | 仅用于标记设置.<br>总线缓冲有效的开始地址= {BUFSEG[8:3], 3'b000} |

缓冲分割寄存器(BUFSEGx) x = 0~5

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **BUFSEG0** | USB_BA+0x020 | R/W | **描述** | 0x0000_0000 |
| **BUFSEG1** | USB_BA+0x030 | R/W | 端点0的缓冲分割 | 0x0000_0000 |
| **BUFSEG2** | USB_BA+0x040 | R/W | 端点1的缓冲分割 | 0x0000_0000 |
| **BUFSEG3** | USB_BA+0x050 | R/W | 端点2的缓冲分割 | 0x0000_0000 |
| **BUFSEG4** | USB_BA+0x060 | R/W | 端点3的缓冲分割 | 0x0000_0000 |
| **BUFSEG5** | USB_BA+0x070 | R/W | 端点4的缓冲分割 | 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 | | | | | | | **BUFSEGx** |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **BUFSEGx** | | | | | 保留 | | |

| Bits | 描述 | | |
|---|---|---|---|
| [8:3] | **BUFSEGx** | R/W | USB缓冲有效的开始地址= {BUFSEG[8:3], 3'b000} |

最大有效载荷寄存器(MXPLDx) x = 0~5

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **MXPLD0** | USB_BA+0x024 | R/W | 端点0最大有效载荷 | 0x0000_0000 |
| **MXPLD1** | USB_BA+0x034 | R/W | 端点1最大有效载荷 | 0x0000_0000 |
| **MXPLD2** | USB_BA+0x044 | R/W | 端点2最大有效载荷 | 0x0000_0000 |
| **MXPLD3** | USB_BA+0x054 | R/W | 端点3最大有效载荷 | 0x0000_0000 |
| **MXPLD4** | USB_BA+0x064 | R/W | 端点4最大有效载荷 | 0x0000_0000 |
| **MXPLD5** | USB_BA+0x074 | R/W | 端点5最大有效载荷 | 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 | | | | | | | **MXPLD** |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **MXPLD** | | | | | | | |

| Bits | 描述 | | |
|---|---|---|---|
| [8:0] | **MXPLD** | R/W | **Read:** IN : 发送到主机的数据长度.<br>OUT : 从主机接收到数据的实际长度.<br>**Write:** IN : 发送到主机的数据长度， 声明 In_Rdy (IN buffer ready).<br>OUT : 从主机接收到数据的最大长度， 声明 Out_Rdy (OUT buffer ready).<br>**注：一旦MXPLD被写，数据包将在IN/OUT标志到达之后立即发送/接收.** |

### 配置寄存器(CFGx) x = 0~5

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **CFG0** | USB_BA+0x028 | R/W | 端点0的配置 | 0x0000_0000 |
| **CFG1** | USB_BA+0x038 | R/W | 端点1的配置 | 0x0000_0000 |
| **CFG2** | USB_BA+0x048 | R/W | 端点2的配置 | 0x0000_0000 |
| **CFG3** | USB_BA+0x058 | R/W | 端点3的配置 | 0x0000_0000 |
| **CFG4** | USB_BA+0x068 | R/W | 端点4的配置 | 0x0000_0000 |
| **CFG5** | USB_BA+0x078 | R/W | 端点5的配置 | 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 | | | | | | stall_ctl | 保留 |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| DSQ | state | | ISOCH | EPT | | | |

| Bits | 描述 | | |
|---|---|---|---|
| [9] | **stall_ctl** | R/W | 0 = 禁止自动清 stall<br>1 = 在setup阶段允许自动清stall |
| [7] | **DSQ** | R/W | 主机应答之后，IN标记自动指定为数据0或1. |
| [6:5] | **state** | R/W | 00 = 禁止端点<br>01 = Out 端点<br>10 = IN 端点<br>11 = 无定义 |
| [4] | **ISOCH** | R/W | 同步，无握手 |
| [3:0] | **EPT** | R/W | 端点号 |

额外配置寄存器(CFGPx) x = 0~5

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **CFGP0** | USB_BA+0x02C | R/W W&C | 延迟Stall控制寄存器和In/out 准备清端点0标志 | 0x0000_0000 |
| **CFGP 1** | USB_BA+0x03C | R/W W&C | 延迟Stall控制寄存器和In/out 准备清端点1标志 | 0x0000_0000 |
| **CFGP 2** | USB_BA+0x04C | R/W W&C | 延迟Stall控制寄存器和In/out 准备清端点2标志 | 0x0000_0000 |
| **CFGP 3** | USB_BA+0x05C | R/W W&C | 延迟Stall控制寄存器和In/out 准备清端点3标志 | 0x0000_0000 |
| **CFGP 4** | USB_BA+0x06C | R/W W&C | 延迟Stall控制寄存器和In/out 准备清端点4标志 | 0x0000_0000 |
| **CFGP 5** | USB_BA+0x07C | R/W W&C | 延迟Stall控制寄存器和In/out 准备清端点5标志 | 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 | | | | | | **stall** | **CFGP** |

| Bits | 描述 | | |
|---|---|---|---|
| [1] | **stall** | R/W | 置1， 使设备响应STALL |
| [0] | **CFGP** | W&C | IN: 写 ‘1’ 清in ready （由MXPLD置1）.<br>OUT: 写 ‘1’清out ready（由MXPLD置1） |

### 驱动 SE0 寄存器(DRVSE0)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **DRVSE0** | USB_BA+0x090 | R/W | USB PHY驱动SE0 | 0x0000_0001 |

| 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 | | | | | | | **DRVSE0** |

| Bits | 描述 | | |
|---|---|---|---|
| [0] | **DRVSE0** | R/W | 1/0: 驱动 se0/无 |

## 5.5 通用 I/O

### 5.5.1 概述与特征

这款MCU有80个通用I/O引脚，并共享着特殊功能。80个引脚分配在GPIOA, GPIOB, GPIOC, GPIOD 与 GPIOE五个口上，每个口最多16个引脚。每个引脚都是独立的，都有相应的寄存器来控制引脚模式与数据

I/O引脚上的I/O类型可由软件独立地配置为输入，输出，开漏或准双端模式. 所有的I/O引脚处于准双端模式，端口数据寄存器GPIOx_DOUT[15:0]的值为0x000_FFFF. 每个I/O引脚配置为弱上拉，即用一颗110KΩ~300KΩ的上拉电阻接到$V_{DD}$ 上.

#### *5.5.1.1 输入模式的说明*

**设置 GPIOx_OMD**(PMDn[1:0]) 为00b ，GPIOx port [n] 为输入模式，I/O引脚为三态（高阻），没有输出驱动能力。**GPIOx_PIN的值反映相当端口的状态.**

#### *5.5.1.2 输出模式的说明*

**设置 GPIOx_OMD**(PMDn[1:0]) 为 01b，GPIOx port [n] 为输入模式，I/O支持数字输出功能，有source/sink电流能力. GPIO_DOUT 相应位的值被送到相应引脚上.

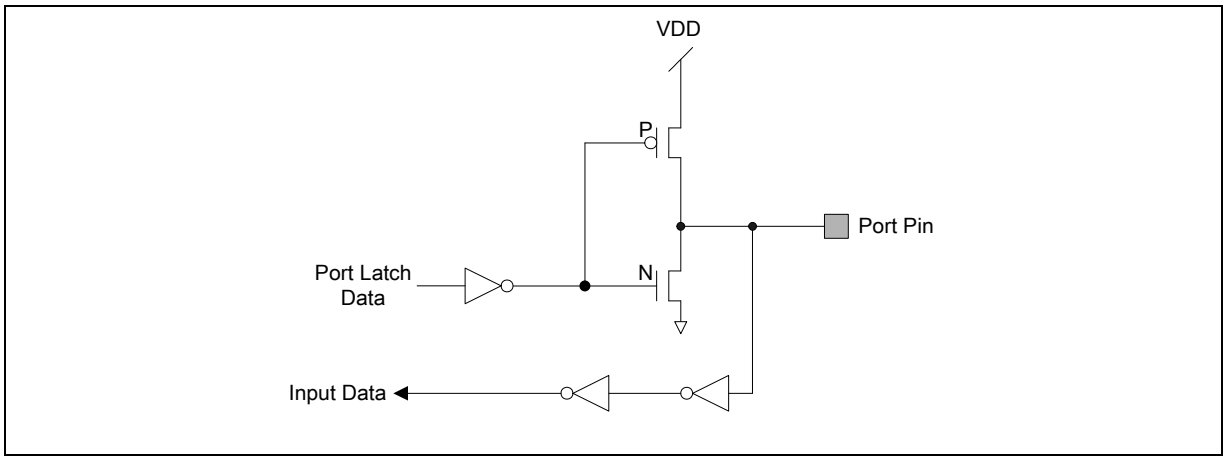



图 5-10 推挽输出

#### 5.5.1.3 开漏模式的说明

**设置 GPIOx_OMD**(PMDn[1:0]) 为 10b，GPIOx port [n]为开漏模式，I/O支持数字输出功能，仅有sink电流能力，一颗外加上接电阻用于驱动到高状态. 如果GPIOx_DOUT 相应位bit [n]的值为“0”，引脚上输出”low”. 如果GPIOx_DOUT 相应位bit [n]的值为“1”，该引脚输出为高，可以由内部上拉电阻或外部电阻控制.

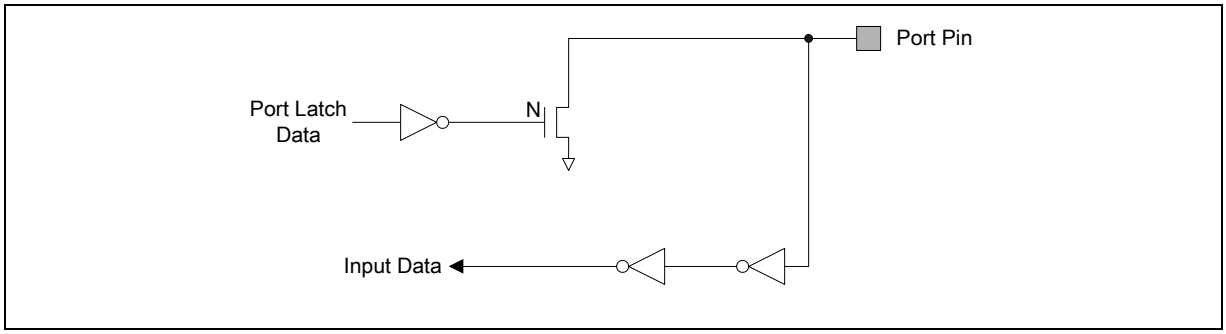



图 5-11 开漏输出

#### 5.5.1.4 准双端模式的说明

**设置GPIOx_OMD**(PMDn[1:0]) 为 11b，GPIOx port [n] 为准双端模式，I/O同时支持数字输出和输入功能，但source电流仅达数百uA. 要实现数字输入，需要先将**GPIOx_DOUT** 相应位置1，准双端输出是80C51及其派生产品所共有的模式. 若GPIOx_DOUT相应位bit[n]为”0”，引脚上输出为“低”. 若GPIOx_DOUT相应位bit[n]为”1”，该引脚将核对引脚值. 若引脚值为高，没有任何动作，若引脚值为低，该引脚在2个时钟周期内置高，然后禁止强输入驱动，引脚状态由内部上拉电阻控制. 注: 准双端模式的source 电流能力仅有200uA到30uA(相应VDD的电压从5.0V到2.5V)

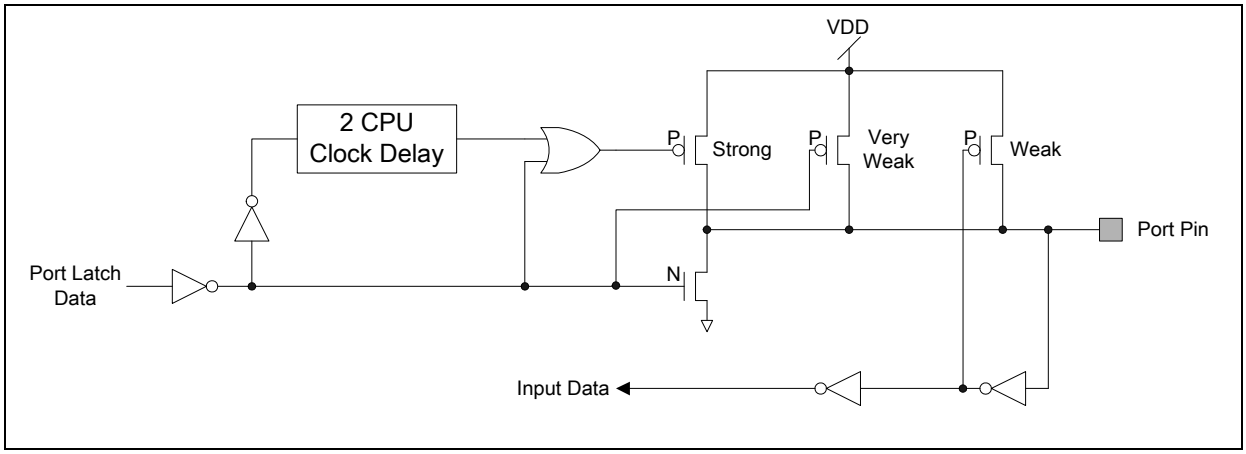



图 5-12 准双端 I/O 模式

### 5.5.2 GPIO控制寄存器图

**R: 只读, W: 只写, R/W: 可读写**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **GP_BA = 0x5000_4000** | | | | |
| **GPIOA_PMD** | GP_BA+0x000 | R/W | GPIO Port A Bit 模式控制 | 0xFFFF_FFFF |
| **GPIOA_OFFD** | GP_BA+0x004 | R/W | GPIO Port A Bit OFF 数字使能 | 0x0000_0000 |
| **GPIOA_DOUT** | GP_BA+0x008 | R/W | GPIO Port A 数据输出值 | 0x0000_FFFF |
| **GPIOA_DMASK** | GP_BA+0x00C | R/W | GPIO Port A 数据输出写屏蔽 | 0x0000_0000 |
| **GPIOA_PIN** | GP_BA+0x010 | R | GPIO Port A 管脚数值 | 0x0000_XXXX |
| **GPIOA_DBEN** | GP_BA+0x014 | R/W | GPIO Port A 防反弹使能 | 0x0000_0000 |
| **GPIOA_IMD** | GP_BA+0x018 | R/W | GPIO Port A 中断模式控制 | 0x0000_0000 |
| **GPIOA_IEN** | GP_BA+0x01C | R/W | GPIO Port A 中断使能 | 0x0000_0000 |
| **GPIOA_ISRC** | GP_BA+0x020 | R/WC | GPIO Port A中断源标志 | 0xXXXX_XXXX |
| **GPIOB_PMD** | GP_BA+0x040 | R/W | GPIO Port B Bit 模式使能 | 0xFFFF_FFFF |
| **GPIOB_OFFD** | GP_BA+0x044 | R/W | GPIO Port B Bit OFF 数字使能 | 0x0000_0000 |
| **GPIOB_DOUT** | GP_BA+0x048 | R/W | GPIO Port B 数据输出值 | 0x0000_FFFF |
| **GPIOB_DMASK** | GP_BA+0x04C | R/W | GPIO Port B数据输出写屏蔽 | 0x0000_0000 |
| **GPIOB_PIN** | GP_BA+0x050 | R | GPIO Port B 管脚数值 | 0x0000_XXXX |
| **GPIOB_DBEN** | GP_BA+0x054 | R/W | GPIO Port B 防反弹使能 | 0x0000_0000 |
| **GPIOB_IMD** | GP_BA+0x058 | R/W | GPIO Port B中断模式控制 | 0x0000_0000 |
| **GPIOB_IEN** | GP_BA+0x05C | R/W | GPIO Port B中断使能 | 0x0000_0000 |
| **GPIOB_ISRC** | GP_BA+0x060 | R/WC | GPIO Port B中断源标志 | 0xXXXX_XXXX |
| **GPIOC_PMD** | GP_BA+0x080 | R/W | GPIO Port C Bit模式使能 | 0xFFFF_FFFF |
| **GPIOC_OFFD** | GP_BA+0x084 | R/W | GPIO Port C Bit OFF 数字使能 | 0x0000_0000 |
| **GPIOC_DOUT** | GP_BA+0x088 | R/W | GPIO Port C 数据输出值 | 0x0000_FFFF |
| **GPIOC_DMASK** | GP_BA+0x08C | R/W | GPIO Port C数据输出写屏蔽 | 0x0000_0000 |
| **GPIOC_PIN** | GP_BA+0x090 | R | GPIO Port C管脚数值 | 0x0000_XXXX |
| **GPIOC_DBEN** | GP_BA+0x094 | R/W | GPIO Port C 防反弹使能 | 0x0000_0000 |
| **GPIOA_IMD** | GP_BA+0x098 | R/W | GPIO Port C 中断模式控制 | 0x0000_0000 |

| | | | | |
|---|---|---|---|---|
| **GPIOC_IEN** | GP_BA+0x09C | R/W | GPIO Port C 中断使能 | 0x0000_0000 |
| **GPIOC_ISRC** | GP_BA+0x0A0 | R/WC | GPIO Port C 中断源标志 | 0xXXXX_XXXX |
| **GPIOD_PMD** | GP_BA+0x0C0 | R/W | GPIO Port D Bit 模式使能 | 0xFFFF_FFFF |
| **GPIOD_OFFD** | GP_BA+0x0C4 | R/W | GPIO Port D Bit OFF 数字使能 | 0x0000_0000 |
| **GPIOD_DOUT** | GP_BA+0x0C8 | R/W | GPIO Port D 数据输出值 | 0x0000_FFFF |
| **GPIOD_DMASK** | GP_BA+0x0CC | R/W | GPIO Port D数据输出写屏蔽 | 0x0000_0000 |
| **GPIOD_PIN** | GP_BA+0x0D0 | R | GPIO Port D管脚数值 | 0x0000_XXXX |
| **GPIOD_DBEN** | GP_BA+0x0D4 | R/W | GPIO Port D 防反弹使能 | 0x0000_0000 |
| **GPIOD_IMD** | GP_BA+0x0D8 | R/W | GPIO Port D 中断模式控制 | 0x0000_0000 |
| **GPIOD_IEN** | GP_BA+0x0DC | R/W | GPIO Port D 中断使能 | 0x0000_0000 |
| **GPIOD_ISRC** | GP_BA+0x0E0 | R/WC | GPIO Port D 中断源标志 | 0xXXXX_XXXX |
| **GPIOE_PMD** | GP_BA+0x100 | R/W | GPIO Port E Bit 模式使能 | 0xFFFF_FFFF |
| **GPIOE_OFFD** | GP_BA+0x104 | R/W | GPIO Port E Bit OFF 数字使能 | 0x0000_0000 |
| **GPIOE_DOUT** | GP_BA+0x108 | R/W | GPIO Port E 数据输出值 | 0x0000_FFFF |
| **GPIOE_DMASK** | GP_BA+0x10C | R/W | GPIO Port E数据输出写屏蔽 | 0x0000_0000 |
| **GPIOE_PIN** | GP_BA+0x110 | R | GPIO Port E管脚数值 | 0x0000_XXXX |
| **GPIOE_DBEN** | GP_BA+0x114 | R/W | GPIO Port E防反弹使能 | 0x0000_0000 |
| **GPIOE_IMD** | GP_BA+0x118 | R/W | GPIO Port E中断模式控制 | 0x0000_0000 |
| **GPIOE_IEN** | GP_BA+0x11C | R/W | GPIO Port E中断使能 | 0x0000_0000 |
| **GPIOE_ISRC** | GP_BA+0x120 | R/WC | GPIO Port E中断源标志 | 0xXXXX_XXXX |
| **DBNCECON** | GP_BA+0x180 | R/W | 防反弹循环控制 | 0x0000_0020 |

### 5.5.3 GPIO控制寄存器描述

**GPIO Port [A/B/C/D/E] I/O模式控制(GPIOx_PMD)**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **GPIOA_PMD** | GP_BA+0x000 | R/W | GPIO Port A Pin 输出模式使能 | 0xFFFF_FFFF |
| **GPIOB_PMD** | GP_BA+0x040 | R/W | GPIO Port B Pin 输出模式使能 | 0xFFFF_FFFF |
| **GPIOC_PMD** | GP_BA+0x080 | R/W | GPIO Port C Pin 输出模式使能 | 0xFFFF_FFFF |
| **GPIOD_PMD** | GP_BA+0x0C0 | R/W | GPIO Port D Pin 输出模式使能 | 0xFFFF_FFFF |
| **GPIOE_PMD** | GP_BA+0x100 | R/W | GPIO Port E Pin 输出模式使能 | 0xFFFF_FFFF |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| PMD15 | | PMD14 | | PMD13 | | PMD12 | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| PMD11 | | PMD10 | | PMD9 | | PMD8 | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| PMD7 | | PMD6 | | PMD5 | | PMD4 | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PMD3 | | PMD2 | | PMD1 | | PMD0 | |

| Bits | 描述 | |
|---|---|---|
| [2n+1 :2n] | **PMDn** | **GPIOX I/O Pin[n] 模式控制**<br>GPIOx 的I/O类型<br>00 = GPIO port [n] pin 输入模式.<br>01 = GPIO port [n] pin 输出模式.<br>10 = GPIO port [n] pin 开漏模式.<br>11 = GPIO port [n] pin 准双端模式. |

### GPIO Port [A/B/C/D/E] Bit OFF数字寄存器使能(GPIOx_OFFD)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **GPIOA_OFFD** | GP_BA+0x004 | R/W | GPIO Port A Pin OFF 数字使能 | 0x0000_0000 |
| **GPIOB_OFFD** | GP_BA+0x044 | R/W | GPIO Port B Pin OFF数字使能 | 0x0000_0000 |
| **GPIOC_OFFD** | GP_BA+0x084 | R/W | GPIO Port C Pin OFF数字使能 | 0x0000_0000 |
| **GPIOD_OFFD** | GP_BA+0x0C4 | R/W | GPIO Port D Pin OFF数字使能 | 0x0000_0000 |
| **GPIOE_OFFD** | GP_BA+0x104 | R/W | GPIO Port E Pin OFF数字使能 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [16:31] | **OFFD** | **OFFD: GPIOx Pin[n] OFF 数字输入通道使能**<br>1 = 禁止 IO数字输入通道 (digital input tied to low)<br>0 = 使能 IO 数字输入通道 |
| [0:15] | 保留 | 保留 |

### GPIO Port [A/B/C/D/E] 数据输出值( (GPIOx_DOUT)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **GPIOA_DOUT** | GP_BA+0x008 | R/W | GPIO 端口 A数据输出值 | 0x0000_FFFF |
| **GPIOB_DOUT** | GP_BA+0x048 | R/W | GPIO 端口 B数据输出值 | 0x0000_FFFF |
| **GPIOC_DOUT** | GP_BA+0x088 | R/W | GPIO 端口C数据输出值 | 0x0000_FFFF |
| **GPIOD_DOUT** | GP_BA+0x0C8 | R/W | GPIO 端口D数据输出值 | 0x0000_FFFF |
| **GPIOE_DOUT** | GP_BA+0x108 | R/W | GPIO 端口E数据输出值 | 0x0000_FFFF |

| 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** |
| DOUT[15:8] | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| DOUT[7:0] | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [n] | **DOUT[n]** | **GPIOx Pin[n] 输出值**<br>GPIO配置成输出，输入，和准双端模式时，这些位控制GPIO引脚状态.<br>1 = 相应的输出模式使能位设置时，GPIO port [A/B/C/D/E] Pin[n] 为高.<br>0 =相应的输出模式使能位设置时，GPIO port [A/B/C/D/E] Pin[n] 为低. |

### GPIO Port [A/B/C/D/E] 数据输出写屏蔽(GPIOx _DMASK)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **GPIOA_DMASK** | GP_BA+0x00C | R/W | GPIO Port A 数据输出写屏蔽 | 0xXXXX_0000 |
| **GPIOB_DMASK** | GP_BA+0x04C | R/W | GPIO Port B 数据输出写屏蔽 | 0xXXXX_0000 |
| **GPIOC_DMASK** | GP_BA+0x08C | R/W | GPIO Port C 数据输出写屏蔽 | 0xXXXX_0000 |
| **GPIOD_DMASK** | GP_BA+0x0CC | R/W | GPIO Port D 数据输出写屏蔽 | 0xXXXX_0000 |
| **GPIOE_DMASK** | GP_BA+0x10C | R/W | GPIO Port E 数据输出写屏蔽 | 0xXXXX_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** |
| **DMASK[15:8]** | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **DMASK[7:0]** | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [n] | **DMASK[n]** | **Port [A/B/C/D/E] 数据输出写屏蔽**<br>用于保护相应寄存器GPIOx_DOUT bit[n]. 在设置DMASK bit[n] 为 “1”，相应DOUTn bit 被保护，写信号被屏蔽时，不能向保护位写数据<br>0 = 更新相应的GPIO_DOUT[n] 位<br>1 = 保护相应的GPIO_DOUT[n] 位 |

### GPIO Port [A/B/C/D/E] 管脚数据(GPIOx _PIN)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **GPIOA_PIN** | GP_BA+0x010 | R | GPIO Port A 管脚数据 | 0x0000_XXXX |
| **GPIOB_PIN** | GP_BA+0x050 | R | GPIO Port B 管脚数据 | 0x0000_XXXX |
| **GPIOC_PIN** | GP_BA+0x090 | R | GPIO Port C 管脚数据 | 0x0000_XXXX |
| **GPIOD_PIN** | GP_BA+0x0D0 | R | GPIO Port D 管脚数据 | 0x0000_XXXX |
| **GPIOE_PIN** | GP_BA+0x110 | R | GPIO Port E 管脚数据 | 0x0000_XXXX |

| 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** |
| **PIN[15:8]** | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **PIN[7:0]** | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [n] | **PIN[n]** | **Port [A/B/C/D/E] 管脚数据**<br>这些位的值为各个GPIO真实状态的反映 |

### GPIO Port [A/B/C/D/E] 防反弹使能(GPIOx _DBEN)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **GPIOA_DBEN** | GP_BA+0x014 | R/W | GPIO Port A 防反弹使能 | 0xXXXX_0000 |
| **GPIOB_DBEN** | GP_BA+0x054 | R/W | GPIO Port B 防反弹使能 | 0xXXXX_0000 |
| **GPIOC_DBEN** | GP_BA+0x094 | R/W | GPIO Port C 防反弹使能 | 0xXXXX_0000 |
| **GPIOD_DBEN** | GP_BA+0x0D4 | R/W | GPIO Port D 防反弹使能 | 0xXXXX_0000 |
| **GPIOE_DBEN** | GP_BA+0x114 | R/W | GPIO Port E 防反弹使能 | 0xXXXX_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** |
| **DBEN[15:8]** | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **DBEN[7:0]** | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [n] | **DBEN[n]** | **Port [A/B/C/D/E] 输入信号防反弹使能**<br>DBEN[n]用于使能相应位的防反弹功能. 如果信号脉冲宽度不能被两个连续的防反弹采样周期所采样，则被视为信号反弹，从而不触发中断.<br>DBEN[n] 仅用于边沿触发 “edge-trigger” 中断，不能电平“level trigger”触发中断<br>0 = 禁止 bit[n] 防反弹功能<br>1 = 使能 bit[n] 防反弹功能<br>防反弹功能对于边沿触发中断有效, 对于电平触发中断模式，防反弹功能全能位不起作用. |

### GPIO Port [A/B/C/D/E] 中断模式控制 (GPIOx_IMD)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **GPIOA_IMD** | GP_BA+0x018 | R/W | GPIO Port A中断模式控制 | 0xXXXX_0000 |
| **GPIOB_IMD** | GP_BA+0x058 | R/W | GPIO Port B中断模式控制 | 0xXXXX_0000 |
| **GPIOC_IMD** | GP_BA+0x098 | R/W | GPIO Port C中断模式控制 | 0xXXXX_0000 |
| **GPIOD_IMD** | GP_BA+0x0D8 | R/W | GPIO Port D中断模式控制 | 0xXXXX_0000 |
| **GPIOE_IMD** | GP_BA+0x118 | R/W | GPIO Port E中断模式控制 | 0xXXXX_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** |
| IMD[15:8] | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| IMD[7:0] | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [n] | **IMD[n]** | **Port [A/B/C/D/E] 中断模式控制**<br>IMD[n] 用于控制电平触发或边沿触发的中断. 若为边沿触发中断，触发源是控制防反弹，如果是电平触发中断，输入源由一个时钟采样并产生中断<br>0 = 边沿触发中断<br>1 = 电平触发中断<br>设置引脚为电平触发中断，仅需要在寄存器GPIOX_IEN 设置一个电平，若设置为既有电平触发，又有边沿触发，设置被忽略，不会产生中断<br>防反弹功能对于边沿触发中断有效，对于电平触发中断无效. |

### GPIO Port [A/B/C/D] 中断使能控制(GPIOx _IEN)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **GPIOA_IEN** | GP_BA+0x01C | R/W | GPIO Port A 中断使能 | 0x0000_0000 |
| **GPIOB_IEN** | GP_BA+0x05C | R/W | GPIO Port B 中断使能 | 0x0000_0000 |
| **GPIOC_IEN** | GP_BA+0x09C | R/W | GPIO Port C 中断使能 | 0x0000_0000 |
| **GPIOD_IEN** | GP_BA+0x0DC | R/W | GPIO Port D 中断使能 | 0x0000_0000 |
| **GPIOE_IEN** | GP_BA+0x11C | R/W | GPIO Port E 中断使能 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [n+16] | **IR_EN[n]** | **Port [A/B/C/D/E] 输入上升沿或输入高电平的中断使能**<br>IR_EN[x] 用于使能相应GPIO_PIN[x]输入的中断. 置 “1” 也可以使能引脚唤醒功能<br>设置 IR_EB[x] 位为 “1”:<br>如果中断是电平触发模式，输入PIN[x]的状态为高电平时，产生中断.<br>如果中断是边沿触发模式，输入PIN[x]的状态由低电平到高电平变化时，产生中断.<br>1 = 使能PIN[x]高电平或由低电平到高电平变化的中断<br>0 = 禁止PIN[x]高电平或由低电平到高电平变化的中断 |
| [n] | **IF_EN[n]** | **Port [A/B/C/D/E] 输入下降沿或输入低电平的中断使能**<br>IF_EN[n] 用于使能相应GPIO_PIN[n]输入的中断. 置 “1” 也可以使能引脚唤醒功能<br>设置 IF_EB[n] 位为 “1”:<br>如果中断是电平触发模式，输入PIN[n]的状态为低电平时，产生中断.<br>如果中断是边沿触发模式，输入PIN[n]的状态由高电平到低电平变化时，产生中断.<br>1 =使能PIN[n]低电平或由高电平到低电平变化的中断<br>0 =禁止PIN[n]低电平或由高电平到低电平变化的中断 |

### GPIO Port [A/B/C/D/E] 中断触发源 (GPIOx _ISRC)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **GPIOA_ISRC** | GP_BA+0x020 | R/WC | GPIO Port A中断触发源指标 | 0x0000_0000 |
| **GPIOB_ISRC** | GP_BA+0x060 | R/WC | GPIO Port B 中断触发源指标 | 0x0000_0000 |
| **GPIOC_ISRC** | GP_BA+0x0A0 | R/WC | GPIO Port C 中断触发源指标 | 0x0000_0000 |
| **GPIOD_ISRC** | GP_BA+0x0E0 | R/WC | GPIO Port D 中断触发源指标 | 0x0000_0000 |
| **GPIOE_ISRC** | GP_BA+0x120 | R/WC | GPIO Port E 中断触发源指标 | 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** |
| **IF_ ISRC[15:8]** | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **IF_ ISRC[7:0]** | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [n] | **ISRC[n]** | **Port [A/B/C/D/E]** 中断触发源指标<br>读 :<br>1 = GPIOx[n]产生中断<br>0 = GPIOx[n]没有中断<br>写 :<br>1= 清相应的中断标志<br>0= 无动作 |

### 中断防反弹周期控制(DBNCECON)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| DBNCECON | GP_BA+0x180 | R/W | 外部中断防反弹控制 | 0x0000_0020 |

| 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 | | ICLK_ON | DBCLKSRC | DBCLKSEL | | | |

| Bits | 描述 | |
|---|---|---|
| [5] | ICLK_ON | **中断时钟On模式**<br>如果禁止pin[n]中断，设置该位为“0” 将禁止中断产生时钟电路<br>0 = 如果中断GPIOA/B/C/D/E[n]被禁止，禁止时钟<br>1 = 总是使能中断产生时钟电路 |
| [4] | DBCLKSRC | **防反弹计数器时钟源选择**<br>1 = 防反弹计数器时钟源为内部10KHz 时钟<br>0 = 防反弹计数器时钟源为 HCLK |

| | | |
|---|---|---|
| [3:0] | **DBCLKSEL** | **防反弹采样周期选择**<br>**DBCLKSEL** \| **描述** \| **Descriptio**<br>0 \| 中断采样输入每 1 clocks一次 \| Sample int<br>1 \| 中断采样输入每 2 clocks一次 \| Sample int<br>2 \| 中断采样输入每4 clocks一次 \| Sample int<br>3 \| 中断采样输入每8 clocks一次 \| Sample int<br>4 \| 中断采样输入每16 clocks一次 \| Sample int<br>5 \| 中断采样输入每32 clocks一次 \| Sample int<br>6 \| 中断采样输入每64 clocks一次 \| Sample int<br>7 \| 中断采样输入每128 clocks一次 \| Sample int<br>8 \| 中断采样输入每256 clocks一次 \| Sample int<br>9 \| 中断采样输入每 2*256 clocks一次 \| Sample int<br>10 \| 中断采样输入每4*256clocks一次 \| Sample int<br>11 \| 中断采样输入每8*256 clocks一次 \| Sample int<br>12 \| 中断采样输入每16*256 clocks一次 \| Sample int<br>13 \| 中断采样输入每32*256 clocks一次 \| Sample int<br>14 \| 中断采样输入每64*256 clocks一次 \| Sample int<br>15 \| 中断采样输入每128*256 clocks一次 \| Sample int |

## 5.6 I2C 总线控制器 (主机/从机)

### 5.6.1 简介

I2C 为双线，双向串行总线，通过简单的连线方式达成器件间的数据通讯。标准I2C允许多主机器件在总线上达成，连接多于一个微控制器到I2C 总线，可能性意味着超过一个主机可以同时尝试初始化传输数据。为了避免由此产生混乱，发展出一个仲裁过程。 I2C 总线上数据的传输速率在标准模式下可达100kbit/s 在快速模式下可达400kbit/s 在高速模式，直至 1.0 Mbit/s 的增强高速模式。

I2C 总线支持任何IC，通过串行数据SDA 和串行时钟SCL 线，在连接到总线的器件间传递信息每个器件都有一个唯一的地址识别。总线数据**一字节、一字节**传输，每个字节包含8位，，每次传输可以发送的字节数量不受限制，每个字节后必须跟一个响应位，**首先传输的必须是数据的最高位MSB**。如果从机要完成一些其他功能后，才能接收或发送下一个完整的数据。字节可以使时钟线SCL 保持低电平迫使主机进入等待状态，当从机准备好接收下一个数据字节并释放时钟线SCL 后，数据传输继续。附图5-13为I2C总线时序.

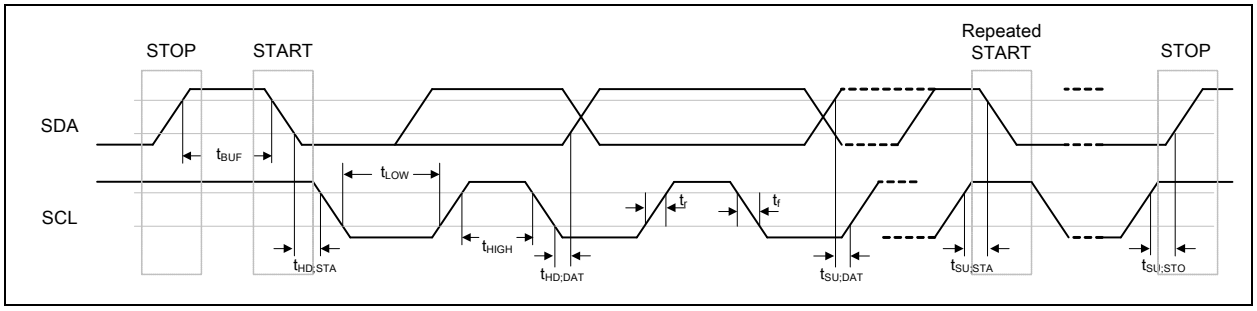



图 5-13 I2C 总线时序

符合I2C总线规范的设备,I2C端口自动处理字节传输，将I2CON的ENS1位设置为1，使能该端口. I2C H/W 接口: SDA (Px.y, 串行数据线) 与 SCL (Px.y, 串行时钟线). 引脚Px.y 与 Px.y 用于 I2C操作需要上拉电阻，因为这两个引脚为开漏脚. 在作为 I2C 端口使用时, 用户必须先将引脚设置为高.

I2C总线通过SDA 及 SCL与连接在总线上的设备间传输数据，总线的主要特为:

- AMBA APB界面兼容
- 支持Philips I2C 标准协议及支持主机模式
- 主 / 从机高至 1Mbit/s ( F M＋ )
- 主从机之间双向数据传输
- 多主机总线支持 (无中心主机).
- 总线上多主机间连续传输数据，实现无间断仲裁
- 总线上不同传输速率设备间，实现同步传输
- 总线上数据暂停传送及恢复传送时，实现同步时钟功能
- 内建14位溢出计数器，当I2C总线上闲置并计数器溢位，产生I2C中断

- 需要外部上拉确保高电平输出，但不支持独立器件电源关闭功能。
- 可编辑的时钟适用于不同速率控制
- 支持7位地址模式
- I2C总线上支持多地址辨识 (2组从机地址被隐藏时)

#### *5.6.1.1 $I^2C$ 协议*

通常标准I2C传输协议包含四个部分

1) 起始信号或重复起始信号
2) 从机地址传输
3) 数据传输
4) 停止信号

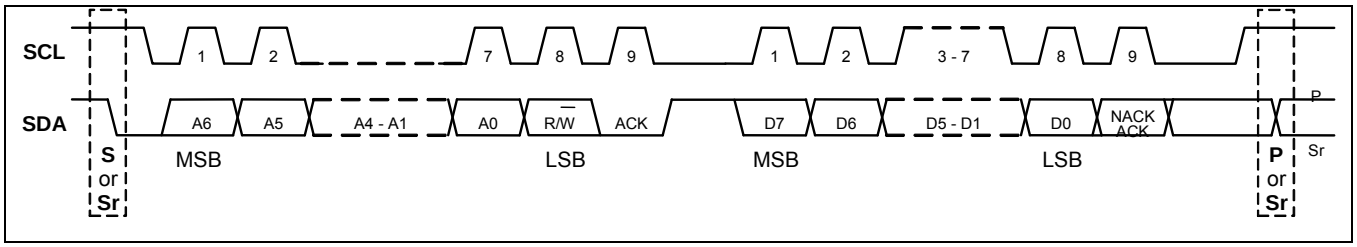



图 5-14 I2C 协议

#### *5.6.1.2 I2C总线上数据传输*

主机发出 从机接收7位地址(一个字节)

传输方向未改变

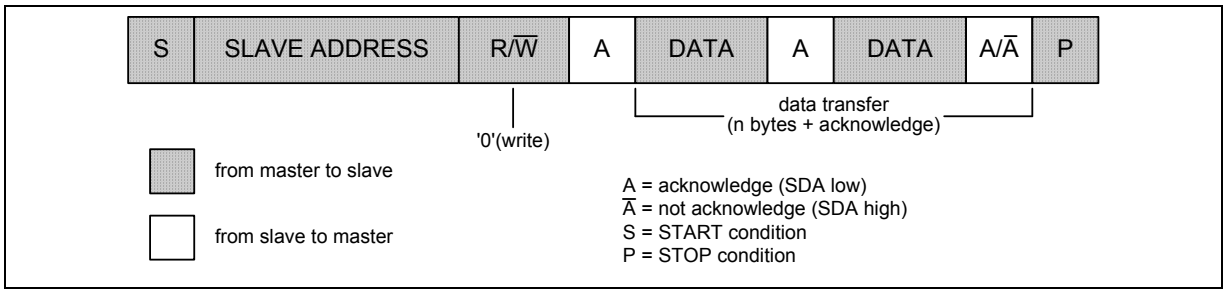



图 5-15 主机向从机传输数据

第一个字节后主机紧接着由从机读取数据(内容为从机地址)

传输方向改变

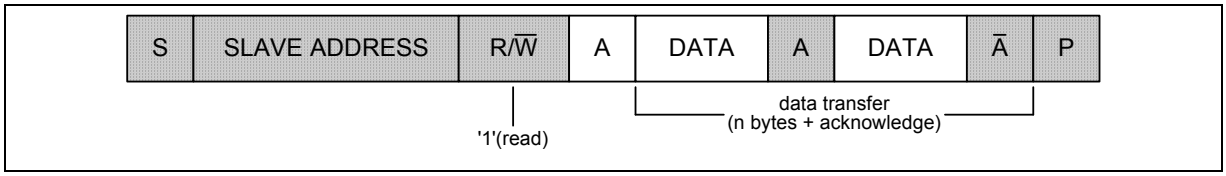



图 5-16 主机由从机读取地址

#### 5.6.1.3 *起始位或重复起始信号*

当总线处于空闲状态下，说明没有主机对总线发起传输请求(SCL 和SDA线同时为高)，主机可以对总线发出起始信号。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**，标示总线上产生起始信号，新的传输开始。

重复起始信号 (Sr) 。如果产生重复起始Sr 条件而不产生停止条件，总线会一直处于忙的状态。此时的起始条件S和重复起始Sr 条件在功能上是一样的.

#### 5.6.1.4 *停止信号*

主机向总线发出停止信号结束数据传送。停止信号，通常用**P-bit**表示, 当SCL线为高时，SDA线上出现由低do 高的信号，被定义为停止信号.

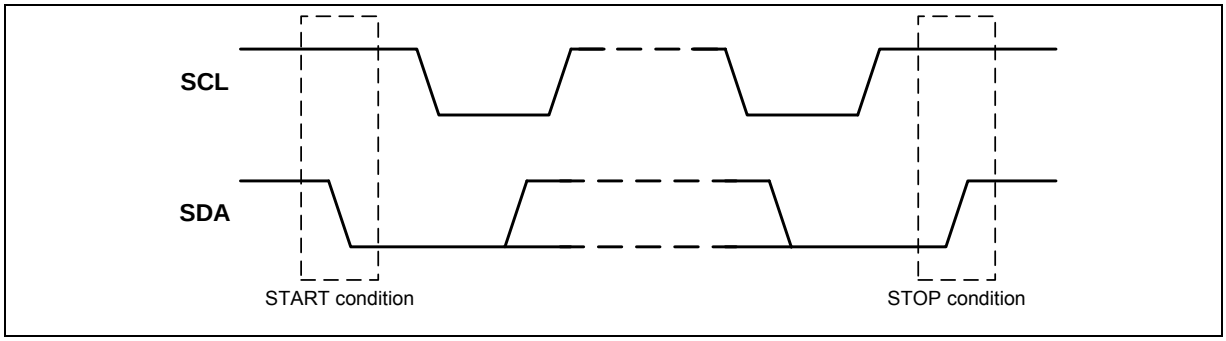



图 5-17 起始和停止信号

#### 5.6.1.5 *从机地址传输*

第一个字节的头7 位组成了从机地址，当发送了一个地址后，系统中的每个器件都在起始条件后将头7位与它自己的地址比较，如果一样器件会认为它被主机寻址，从而当SCL第9个时钟沿时，在SDA上发出低信号作为应答。至于是从机接收还是从机发送，由RW位决定.

#### 5.6.1.6 *数据传输*

当从机地址被成功识别，就可以根据RW所决定的方向，开始一字节以字节的数据传输，每字节最后带一个响应信号，如果从机上产生无响应信号**(NACK)**, 主机产生停止信号，或者产生重复起始信号开始新一轮的数据传输。

当主机作为接收器件时，发生无响应信号**(NACK)** ,从机释放SDA线，使主机产停止信号或重复起始信号.

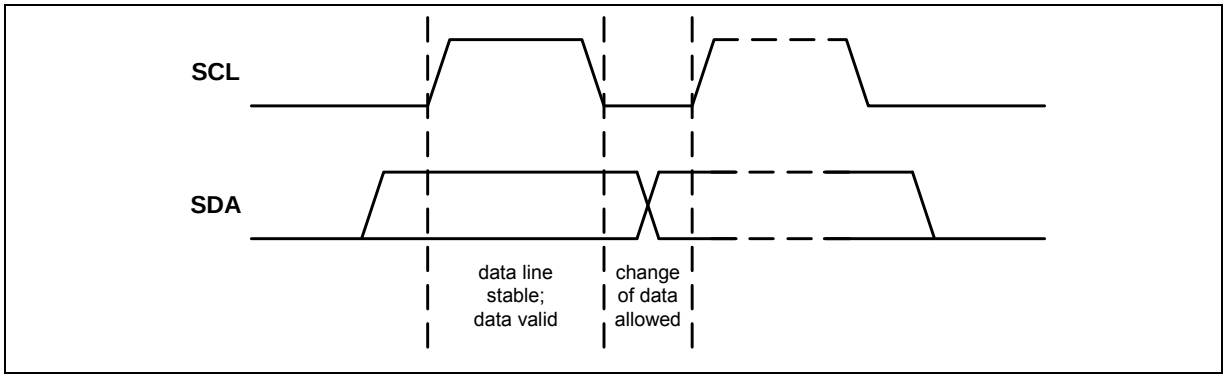



图 5-18 I2C 总线上位传输

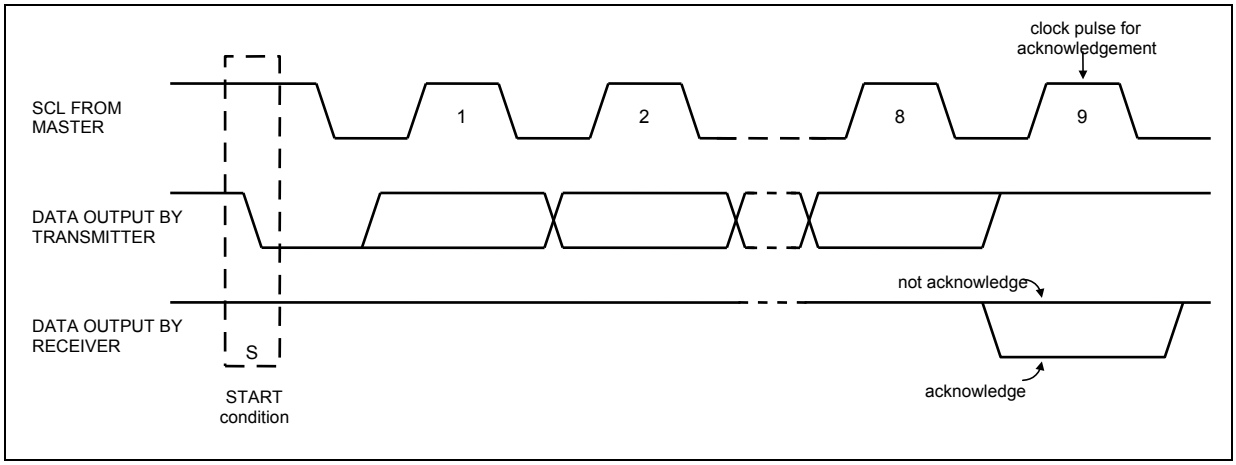



图 5-19 I2C 总线上应答信号

### 5.6.2 I2C 寄存器

NUC1xx共有6个特殊功能寄存，用于I2C通讯: **I2CON** (控制寄存器 C0H), **I2STATUS** (状态寄存器 BDH), **I2DAT** (数据寄存器 BCH), **I2ADDR** (地址寄存器 C1H), **I2CLK** (时钟速率寄存器 BEH) 和 **I2TOC** (定时计数寄存器 BFH) 。寄存器的 第31位至第8位都是保留的，不具备任何功能，读回值为0。

当ENS1置1，I2C口使能后，内部状态由 I2CON 和 I2C总线上状态控制。当有新的状态发生后，会存储到 I2STATUS, I2C 中断标志(SI) 也会自动置起。若此时 EI2 被设定为高，I2C中断会被响应。I2STATUS 内存储5个最常见的标志位，在SI由软件清除之前，最低三位时钟保持低。在NUC1XX上的地址为 4002_0000 和 4012_0000.

#### *5.6.2.1 地址寄存器 (I2ADDR)*

I2C 口内建4个从机地址寄存器。这四个寄存器当作为主机时不予响应。 在从机模式下，MCU会响应共有7个标志位，当I2ADDR 地址与接收的从机地址符合是，这些寄存器会根据状态产生相应的动作。

I2C 口支持“广播呼叫”功能。当GC位被置起，MCU按照广播呼叫方式响应。

**当GC位被置起，当MCU作为从时，总线上的总机发出信号后，默认按照00H地址的从机进行响应。然后按照广播呼叫模式工作。当作为主机时，向总线上发送00H地址时，AA位必须清除。**

I2C总线控制其支持多地址辨识，最多可辨识4组被隐藏的地址。当地址屏蔽寄存器置1，说明收到了一个地址未明的数据。当该位保持0，说明接收到的响应地址完全符合地址寄存器内的值.

| 符号 | 定义 | 地址 | MSB BIT31 | 符号 Bit7 | LSB Bit0 | 复位后的值 |
|---|---|---|---|---|---|---|
| I2ADDR0 | I2C ADDRESS0 | I2C_BA+04H | 保留 | ADDR0.6 ~ ADDR0.0 | GC0 | 0000 0000H |
| I2ADDR1 | I2C ADDRESS1 | I2C_BA+18H | 保留 | ADDR1.6 ~ ADDR1.0 | GC1 | 0000 0000H |
| I2ADDR2 | I2C ADDRESS2 | I2C_BA+1cH | 保留 | ADDR2.6 ~ ADDR2.0 | GC2 | 0000 0000H |
| I2ADDR3 | I2C ADDRESS3 | I2C_BA+20H | 保留 | ADDR3.6 ~ ADDR3.0 | GC3 | 0000 0000H |
| I2ADRM0 | I2C ADDRESS MASK0 | I2C_BA+24H | 保留 | ADRM0.6 ~ ADRM0.0 | 保留 | 0000 000XH |
| I2ADRM1 | I2C ADDRESS MASK1 | I2C_BA+28H | 保留 | ADRM1.6 ~ ADRM1.0 | 保留 | 0000 000XH |
| I2ADRM2 | I2C ADDRESS MASK2 | I2C_BA+2CH | 保留 | ADRM2.6 ~ ADRM2.0 | 保留 | 0000 000XH |
| I2ADRM3 | I2C ADDRESS MASK3 | I2C_BA+30H | 保留 | ADRM3.6 ~ ADRM3.0 | 保留 | 0000 000XH |

#### *5.6.2.2 数据寄存器 (I2DAT)*

该寄存器的内容是准备发送的或刚接收的串行数据一个字节的数据。只要不在移位处理的过程，CPU可以读写访问8-位可直接寻址的特殊功能寄存器。当SIO的状态设定后和串行中断标志(SI)置’1’；只要SI=‘1’，I2DAT中的数据一直是稳定的。在数据移出的过程中，总线上的数据同时也在移动；I2DAT 的内容一直是总线上出现的最后一个字节。在主机发送数据从机接收数据的模式中，不需要仲裁来保证I2DAT中的数据正确。

移位寄存器包含I2DAT和应答位9-位，应答位由SIO的硬件控制，CPU不能访问。I2DAT中的串行数据和应答位在串行时钟SCL线的上升沿移出。当一个字节被移位元到I2DAT后，I2DAT中的串行资料是可

以使用的，应答位(ACK或NACK)在第9个时钟返回。串行数据在每一个下降沿(SCL时钟)从I2DAT移出输出，在每一个上升沿(SCL时钟)资料移进I2DAT.

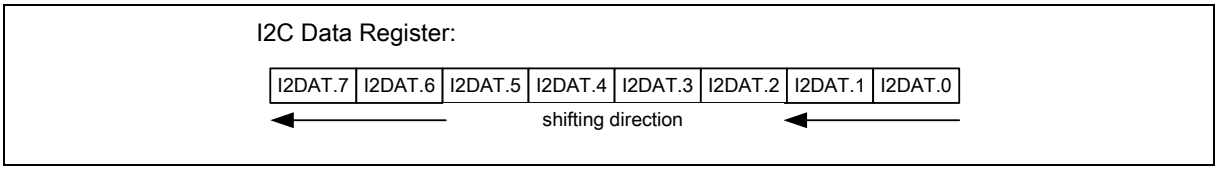



图 5-20 I2C 数据移位

| 符号 | 定义 | 地址 | MSB 符号 LSB<br>BIT3 Bit7 Bit0 | 复位后的值 |
|---|---|---|---|---|
| I2DAT | I2C数据寄存器 | I2C_BA+08H | 保留 I2DAT.7 ~ I2DAT.0 | I2DAT |

*5.6.2.3 控制寄存器 (I2CON)*

有2位会受到硬件的影响： SI位受SIO硬件的影响； 当串行中断请求和STO位被清除即条件显示在总线上，SI位被硬件置'1'。当ENS = "0"时，STO位被清除。

EI 中断使能.

ENSI 置位允许I2C串行功能。SDA和SCL口必须保持逻辑1状态。

STA I2C开始标志位. 置1，STA为高进入主机模式。如果总线为空闲，I2C 硬件检测I2C的总线状态和产生开始条件I2C

STO I2C为主机模式时，STO位置'1'， 将在I2C总线上输出STOP条件。当检测到总线上出现STOP条件。I2C硬件清除STO标志，在从机模式，STO标志被置'1'恢复总线错误条件。在改模式下没有STOP条件传输到I2C总线上。然而I2C 硬件动作好像有STOP条件已经被接收并切换到不可寻址的从接收模式。STO标志由硬件自动清除。如果I2C在主机模式 (在从机模式，I2C产生一个内部的STOP条件不传输到总线上)，如果STA和STO位同时被置'1'，STOP条件被传输到I2C总线。随后I2C传输开始条件。

SI I2C 端口 1 中断标志位. SI标志由硬件置'1' ，必须由软件清'0'；如果EA和ES位(在IE寄存器)被置'1'，串行中断请求，可以产生中断。SI被置'1' SCL在线的串行时钟为低的时间延长并且串行传输数据保持，在线的串行时钟为高的时间不受串行中断标志SI的影响。

AA 接收应答标志位

AA=1在应答时钟脉冲下，SCL上没有应答(SDA上高电平)：1) 接收到自己的地址；2) SIO为主机接收模式，已经接收一个数据。3) SIO为可寻址的从机接收模式，已经接收一个数据。

AA=0 在应答时钟脉冲下，SCL上没有应答(SDA上高电平)： 1) SIO为主机接收模式，已经接收一个数据。2) SIO为可寻址的从机接收模式，已经接收一个数据

| 符号 | 定义 | 地址 | MSB 位地址, 符号 LSB<br>Bit31 Bit7 Bit6 Bit5 Bit4 Bit3 Bit2 Bit0 | 复位后的值 |
|---|---|---|---|---|
| I2CON | I2C CONTROL REGISTER | I2C_BA+00H | 保留 EI ENS1 STA STO SI AA - - | 0000 0000H |

#### 5.6.2.4 状态寄存器(I2STATUS)

I2STATUS是一个8-位只读寄存器。低3位一直为0。其余是状态码。有23个可能的状态码，当I2STATUS的内容是F8H，没有串行中断请求。所有的其它I2STATUS值对应I2C 端口的状态。 当每一个进入状态，就会产生状态中断请求(SI = 1). 在SI被硬件置'1' 1个机器周期后或在SI被软件清除之后，有效状态码出现在I2STATUS中。

另外，00H状态表示总线错误。总线错误发生在START或STOP条件出现在帧结构不正确的位置。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.

| 符号 | 定义 | 地址 | MSB 位地址, 符号 LSB<br>Bit31 Bit7 Bit6 Bit5 Bit4 Bit3 Bit2 Bit0 | | 复位后的值 |
|---|---|---|---|---|---|
| I2STATUS | I2C STATUS REGISTER | I2C_BA+0CH | 保留 | 8 bit status data | 000000F8H |

#### 5.6.2.5 I2C时钟波特率位 (I2CLK)

当SIO在主机模式下，I2C数据的波特率由I2CLK寄存器设定。在从机模式下是很重要的；在从机模式下，SIO将自动与主机I2C设备时钟频率同步，可高达400 KHz。

I2C数据波特率设定是：I2C的数据波特率 = I2C = PCLK /(4x(I2CLK+1))，如果PCLK=16MHz, I2CLK = 40(28H)，I2C的数据波特率I2C = 16MHz /(4X (40 +1)) = 97.56K位/秒.

| 符号 | 定义 | 地址 | MSB 位地址, 符号 LSB<br>Bit31 Bit7 Bit6 Bit5 Bit4 Bit3 Bit2 Bit0 | | 复位后的值 |
|---|---|---|---|---|---|
| I2CLK | I2C CLOCK RATE | I2C_BA+10H | 保留 | I2CLK.7 ~ I2CLK.0 | 0000 0000H |

#### 5.6.2.6 I2C 超时计数寄存器(I2TOC)

当总线上闲置时，MCU提供一个14位超时的计数器。当计数功能使能后，计数器开始计数直至发生超时，TIF置1并要求MCU产生I2C中断或者清除ENT1为0关闭计数功能。当超时计数器使能，对SI标志置高会使计数器复位，再对SI清除为0会重新开始计数。如果I2C总线闲置，会使I2STATUS及SI标志不再更新。I2C中断发生中，该14位超时计数器同样会发生溢出。参考图5-16，用户可通过对TIF位软件清除.

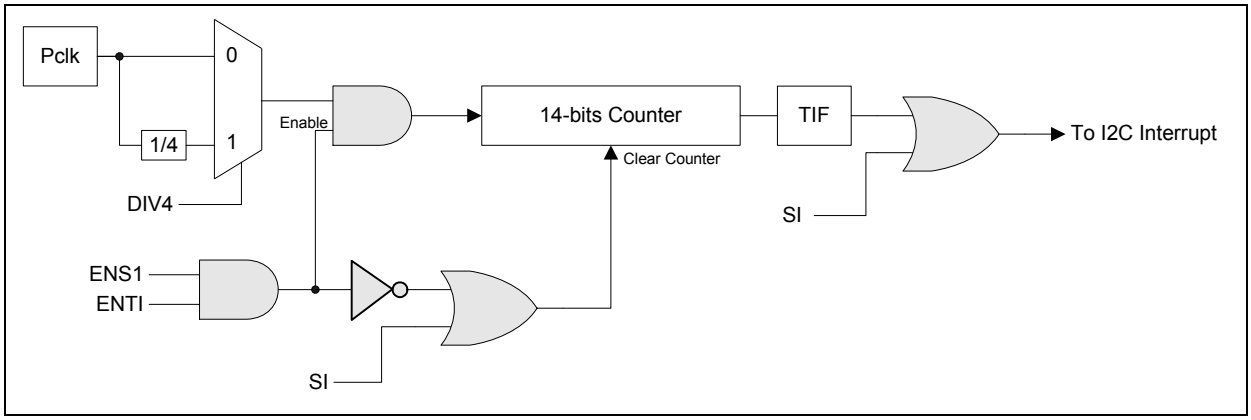



图 5-21: I2C 超时计数器方块图

| 符号 | 定义 | 地址 | MSB 位地址, 符号 LSB | | | | | | | | 复位后的值 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Bit31 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit0 | |
| I2TOC | I2C TIME-OUT COUNTER REGISTER | I2C_BA+14H | 保留 | | | | | ENTI | DIV4 | TIF | 0000 0000B |

### 5.6.3 寄存器说明

**R: 只读, W: 只写, R/W: 可读写**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **I2CON** | I2C_BA+0x00 | R/W | I2C 控制寄存器 | 0x0000_0000 |
| **I2CADRR0** | I2C_BA+0x04 | R/W | I2C 从机地址寄存器0 | 0x0000_0000 |
| **I2DAT** | I2C_BA+0x08 | R/W | I2C 数据寄存器 | 0x0000_0000 |
| **I2STATUS** | I2C_BA+0x0C | R | I2C 状态寄存器 | 0x0000_00F8 |
| **I2CLK** | I2C_BA+0x10 | R/W | I2C 时钟时钟分频寄存器 | 0x0000_0000 |
| **I2TOC** | I2C_BA+0x14 | R/W | I2C 超时控制寄存器 | 0x0000_0000 |
| **I2CADDR1** | I2C_BA+0x18 | R/W | 从机地址寄存器1 | 0x0000_0000 |
| **I2CADDR2** | I2C_BA+0x1C | R/W | 从机地址寄存器2 | 0x0000_0000 |
| **I2CADDR3** | I2C_BA+0x20 | R/W | 从机地址寄存器3 | 0x0000_0000 |
| **I2CADM0** | I2C_BA+0x24 | R/W | 从机隐藏地址寄存器0 | 0x0000_0000 |
| **I2CADM1** | I2C_BA+0x28 | R/W | 从机隐藏地址寄存器1 | 0x0000_0000 |
| **I2CADM2** | I2C_BA+0x2C | R/W | 从机隐藏地址寄存器2 | 0x0000_0000 |
| **I2CADM3** | I2C_BA+0x30 | R/W | 从机隐藏地址寄存器3 | 0x0000_0000 |

### 5.6.4 寄存器描述

**I2C 控制寄存器(I2CON)**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **I2CON** | I2C_BA+0x00H | 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** |
| **EI** | **ENSI** | **STA** | **STO** | **SI** | **AA** | **保留** | 保留 |

| Bits | 描述 | |
|---|---|---|
| [7] | **EI** | 使能中断<br>1 = 使能CPU中断功能<br>0 = 禁止CPU终端功能. |
| [6] | **ENSI** | I2C 控制使能/禁止<br>1 = 使能<br>0 = 禁止<br>当ENS=1 I2C串行功能使能，SDA1和SCL1 管脚必须置为1 |
| [5] | **STA** | I2C 起始标志<br>STA置1 ，进入主机模式，如果总线处于空闲状态，I2C硬件会送出起始信号或重复起始信号。 |
| [4] | **STO** | I2C 停止标志<br>当检测到总线上出现停止条件，I2C为主机模式时，STO位置'1'， 将在I2C总线上输出停止，STO标志由硬件自动清除。在从机模式，I2C硬件清除STO标志， STO标志被置'1'恢复总线"无地址"条件。表明将没有从机再从主机接收到数据。 |
| [3] | **SI** | I2C 中断标志位.<br>在S1STA寄存器上出现一个新的SIO信号时，SI将由硬件置位，如果EA和EI2C位被置'1'，I2C中断请求，可以产生中断。SI必须由软件清除。 |

| | | |
|---|---|---|
| [2] | **AA** | 接收应答标志位<br>0: 在下面情况下，在应答时钟脉冲下，SCL上没有应答(SDA上高电平)： 1) SIO为主机接收模式，已经接收一个数据。2) SIO为可寻址的从机接收模式，已经接收一个数据。<br>1: 在下面情况下，在应答时钟脉冲下，SCL上没有应答(SDA上高电平)：1) 接收到自己的地址；2) SIO为主机接收模式，已经接收一个数据。3) SIO为可寻址的从机接收模式，已经接收一个数据。 |

### I2C数据寄存器(I2DAT)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **I2DAT** | I2C_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** |
| I2DAT[7:0] | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [7:0] | **I2DAT** | **I2C** 数据寄存器<br>Bit[7:0] 为8位I2C数据. |

### I2C状态寄存器(I2STATUS )

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| I2STATUS | I2C_BA+0x0C | R/W | I2C 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 |
| I2STATUS[7:0] | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [7:0] | I2STATUS | **I2C状态寄存器**<br>低三位始终是0；高5位包含状态码。含状态码有23可能；当I2STATUS的值是F8H，表示没有串行断请求；其它的所有的I2STATUS值可以反映I2C的状态。当进入这些状态时会产生一个状态中断请求(SI=1)。一个有效的状态码在SI被硬件设为'1'后一个周期内反映到I2STATUS中；在SI被软件清'0'后一个周期内反映到I2STATUS中。另外，状态码是00H时表示总线错误；当'起始'或'结束'时出现帧结构错误时会产生总线错误。 |

### I2C 波特率控制寄存器 (I2CLK)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **I2CLK** | I2C_BA+0x10 | R/W | I2C 波特率控制寄存器 | 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** |
| **I2CLK[7:0]** | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [7:0] | **I2CLK** | **I2C** 波特率控制寄存器<br>I2C波特率 = PCLK /(4x(I2CLK+1)). |

### I2C 超时计数寄存器 (I2TOC)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| I2TOC | I2C_BA+0x14 | R/W | I2C超时计数寄存器 | 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 | | | | | ENTI | DIV4 | TIF |

| Bits | 描述 | |
|---|---|---|
| [2] | ENTI | **超时计数使能/禁止**<br>1 = 使能<br>0 = 禁止<br>SI被清0，14位超时计数寄存器开始计数，再对SI置1会使计数器复位，并重新打开计数功能。 |
| [1] | DIV4 | **超时计数输入时钟除4**<br>1 = 使能<br>0 = 禁止<br>使能后，溢出时间/4 |
| [0] | TIF | **超时标志**<br>1 = 硬件发生超时，可引发CPU的中断<br>0 = 软件清零 |

### I2C从机地址寄存器 (I2CADDRx)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **I2CADDR0** | I2C_BA+0x04 | R/W | 从机地址寄存器0 | 0x0000_0000 |
| **I2CADDR1** | I2C_BA+0x18 | R/W | 从机地址寄存器1 | 0x0000_0000 |
| **I2CADDR2** | I2C_BA+0x1C | R/W | 从机地址寄存器2 | 0x0000_0000 |
| **I2CADDR3** | I2C_BA+0x20 | R/W | 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** |
| **I2ADDR[7:1]** | | | | | | | **GC** |

| Bits | 描述 | |
|---|---|---|
| [7:1] | **I2ADDR** | I2C地址寄存器:<br>8051 uC可以直接对该8-位寄存器进行读/写操作。在主模式下寄存器的值没有意义；在从机模式下，高7位作为MCU本身的地址。如果地址符合硬件会自动应答。 |
| [0] | **GC** | 全呼功能.<br>0: 禁止全呼功能.<br>1: 允许全呼功能. |

### I2C从机隐藏地址寄存器(I2CADMx)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **I2CADM0** | I2C_BA+0x24 | R/W | I2C 从机隐藏地址寄存器0 | 0x0000_0000 |
| **I2CADM1** | I2C_BA+0x28 | R/W | I2C 从机隐藏地址寄存器1 | 0x0000_0000 |
| **I2CADM2** | I2C_BA+0x2C | R/W | I2C 从机隐藏地址寄存器2 | 0x0000_0000 |
| **I2CADM3** | I2C_BA+0x30 | R/W | I2C 从机隐藏地址寄存器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** |
| I2ADMx[7:1] | | | | | | | GC |

| Bits | 描述 | |
|---|---|---|
| [7:1] | I2ADMx | I2C 隐藏地址寄存器:<br>1 = 允许隐藏(接收到任何地址不予辨识)<br>0 = 禁止隐藏 (接收到的地址必须完全符合正确的地址内容)<br>I2C总线支持多隐藏地址辨识。当打开允许隐藏时，接收到的从机地址是否正确不予处理，当选择为禁止隐藏是，从机 地址必须完全符合其真实的地址才给与响应。 |

### 5.6.5 I2C 操作模式

5种操作模式：主机/传输，主机/接收，从机/传输，从机/接收和GC 模式。

在实际应用中，I2C端口可以作为主机和从机，有主机和从机I2CON中的STA，STO和AA置位后，SI被清除后，SIO 硬件才开始下一个动作，当一个动作结束，I2STATUS将会被更新并且同时SI被硬件置'1'。之后会进入中断服务程序(如果SI中断允许)，新状态码可以被用为软件判断哪一个中断进入中断服务程序.

#### *5.6.5.1 主机传输模式*

当SCL线上输出时钟信号时，SDA线上输出数据。主机产生起始信号，置位STA，主机向总线传输从机地址（一般为7位）+方向位。此时方向位(R/W)为0，并响应W。因此传输内容为SLA+W，形成完整的8位数据。然后等待接收应答（ACK）信号。接收到应答信号后，向被寻址到的从机发送数据，等待ACK信号。并查询起始或计数信号，决定是否继续传输.

#### *5.6.5.2 主机接收模式*

(R/W) 内容为1 ，响应R。此时传输内容为SLA+R. 当SCL线上输出时钟信号时，SDA线上输出数据。线上一次接收8位数据，每接收到一个字节，响应一个接收标志，并辨识是否有起始或结束标志.

#### *5.6.5.3 从机接收模式*

从串行线上接收SDA和SCL信号。每接收完一个字节，发送一个响应位，并辨识是否有起始或结束标志。由硬件识别从机地址.

#### *5.6.5.4 从机传输模式*

第一位接收从机接收模式。在该模式下，方向位用来标示传输方向，当SCL线上接收到时钟信号时，SDA脚上输出数据，并辨识是否有起始或停止标志.

### 5.6.6 数据传输 5 种操作

5种操作模式：主机/传输，主机/接收，从机/传输，从机/接收和GC 模式。I2CON中的STA，STO和AA位将决定SIO硬件下一次的状态在SI位清除后。置位后，SI被清除后，SIO 硬件才开始下一个动作，当一个动作结束，I2STATUS将会被更新并且同时SI被硬件置'1'。之后会进入中断服务程序(如果EA=1，EI2=1)，新状态码可以被用为软件判断哪一个中断进入中断服务程序。

数据传输每种模式如下图所示.

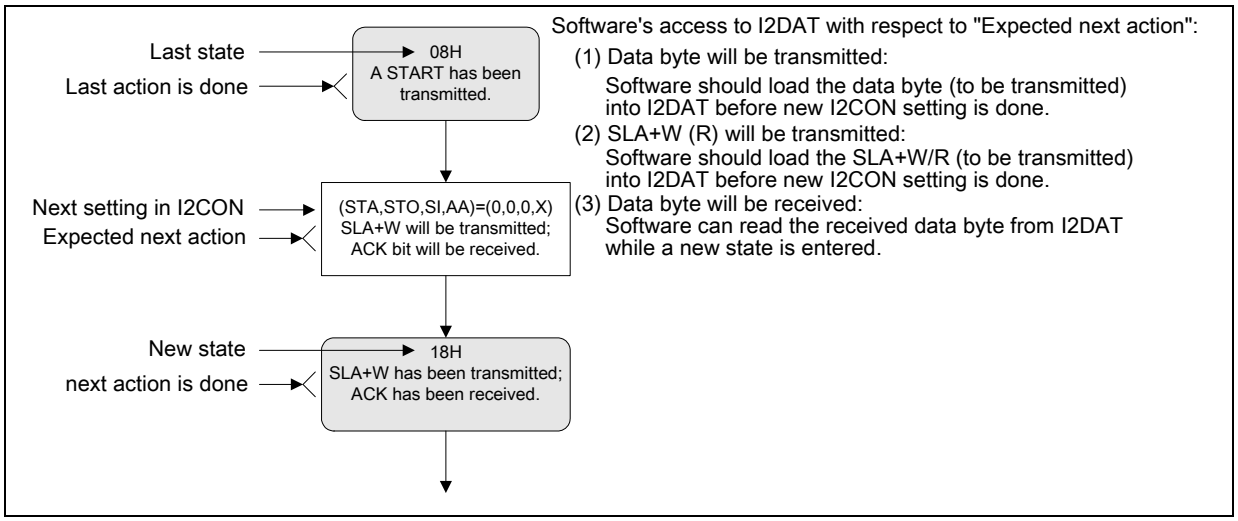



图 5-22 传输流程图

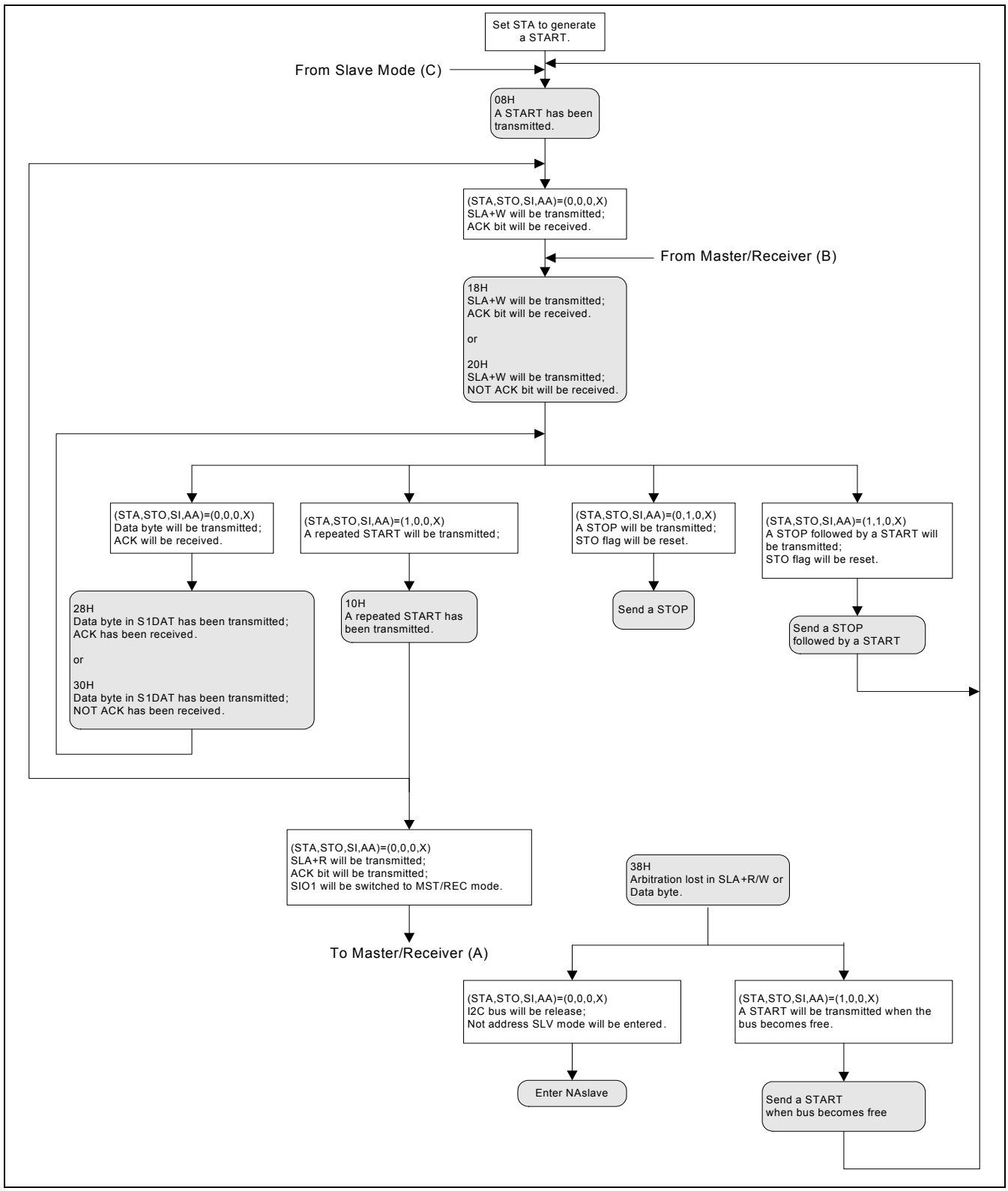

Set STA to generate a START.
From Slave Mode (C)
08H
A START has been transmitted.
(STA,STO,SI,AA)=(0,0,0,X)
SLA+W will be transmitted;
ACK bit will be received.
From Master/Receiver (B)
18H
SLA+W will be transmitted;
ACK bit will be received.
or
20H
SLA+W will be transmitted;
NOT ACK bit will be received.
(STA,STO,SI,AA)=(0,0,0,X)
Data byte will be transmitted;
ACK will be received.
(STA,STO,SI,AA)=(1,0,0,X)
A repeated START will be transmitted;
(STA,STO,SI,AA)=(0,1,0,X)
A STOP will be transmitted;
STO flag will be reset.
(STA,STO,SI,AA)=(1,1,0,X)
A STOP followed by a START will be transmitted;
STO flag will be reset.
28H
Data byte in S1DAT has been transmitted;
ACK has been received.
or
30H
Data byte in S1DAT has been transmitted;
NOT ACK has been received.
10H
A repeated START has been transmitted.
Send a STOP
Send a STOP followed by a START
(STA,STO,SI,AA)=(0,0,0,X)
SLA+R will be transmitted;
ACK bit will be transmitted;
SIO1 will be switched to MST/REC mode.
To Master/Receiver (A)
38H
Arbitration lost in SLA+R/W or Data byte.
(STA,STO,SI,AA)=(0,0,0,X)
I2C bus will be release;
Not address SLV mode will be entered.
(STA,STO,SI,AA)=(1,0,0,X)
A START will be transmitted when the bus becomes free.
Enter NAslave
Send a START when bus becomes free

图 5-23 主机传输模式

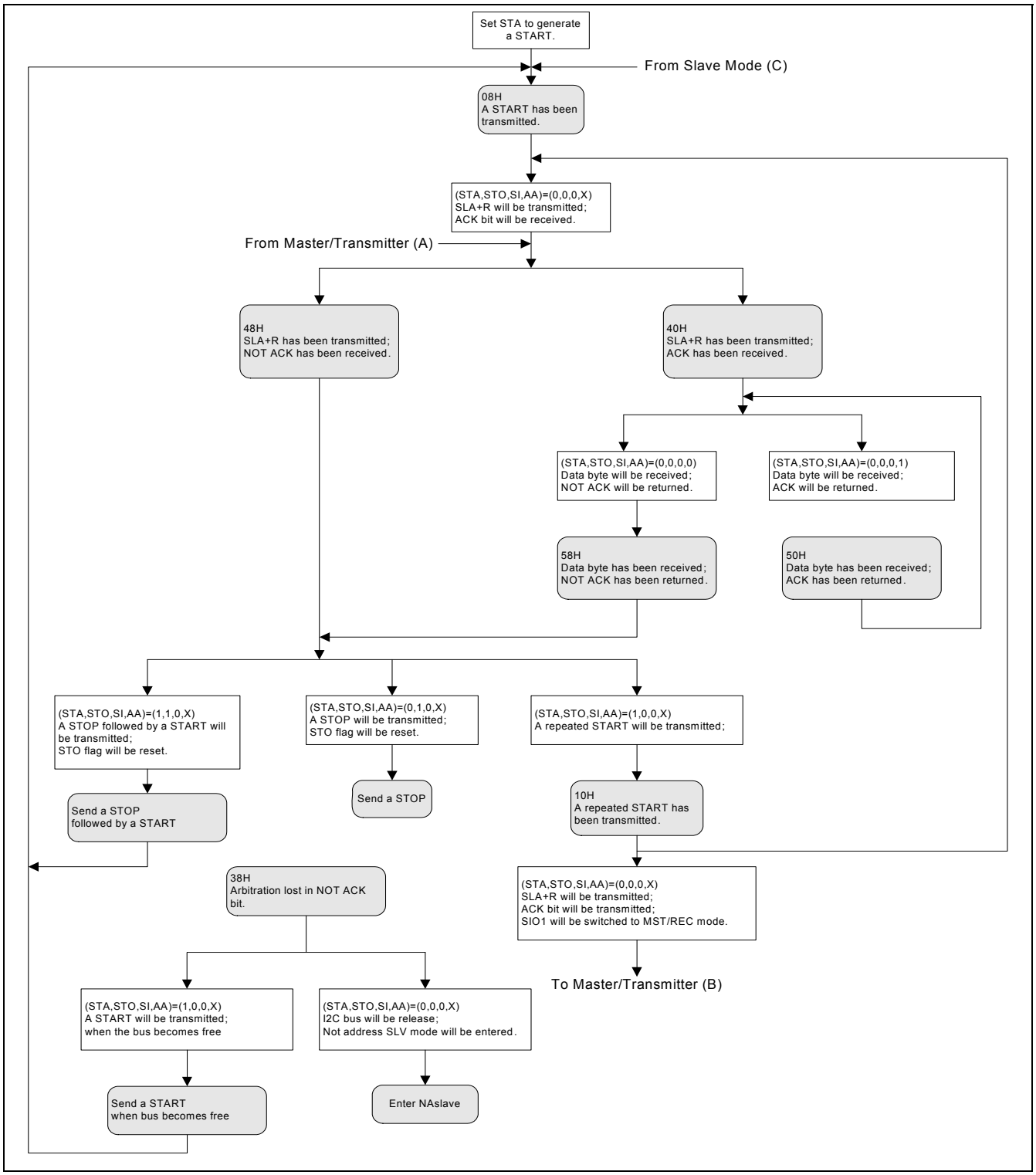

Set STA to generate a START.
From Slave Mode (C)
08H
A START has been transmitted.
(STA,STO,SI,AA)=(0,0,0,X)
SLA+R will be transmitted;
ACK bit will be received.
From Master/Transmitter (A)
48H
SLA+R has been transmitted;
NOT ACK has been received.
40H
SLA+R has been transmitted;
ACK has been received.
(STA,STO,SI,AA)=(0,0,0,0)
Data byte will be received;
NOT ACK will be returned.
(STA,STO,SI,AA)=(0,0,0,1)
Data byte will be received;
ACK will be returned.
58H
Data byte has been received;
NOT ACK has been returned.
50H
Data byte has been received;
ACK has been returned.
(STA,STO,SI,AA)=(1,1,0,X)
A STOP followed by a START will be transmitted;
STO flag will be reset.
(STA,STO,SI,AA)=(0,1,0,X)
A STOP will be transmitted;
STO flag will be reset.
(STA,STO,SI,AA)=(1,0,0,X)
A repeated START will be transmitted;
Send a STOP followed by a START
Send a STOP
10H
A repeated START has been transmitted.
38H
Arbitration lost in NOT ACK bit.
(STA,STO,SI,AA)=(0,0,0,X)
SLA+R will be transmitted;
ACK bit will be transmitted;
SIO1 will be switched to MST/REC mode.
To Master/Transmitter (B)
(STA,STO,SI,AA)=(1,0,0,X)
A START will be transmitted;
when the bus becomes free
(STA,STO,SI,AA)=(0,0,0,X)
I2C bus will be release;
Not address SLV mode will be entered.
Send a START when bus becomes free
Enter NAslave

图 5-24 主机接收模式

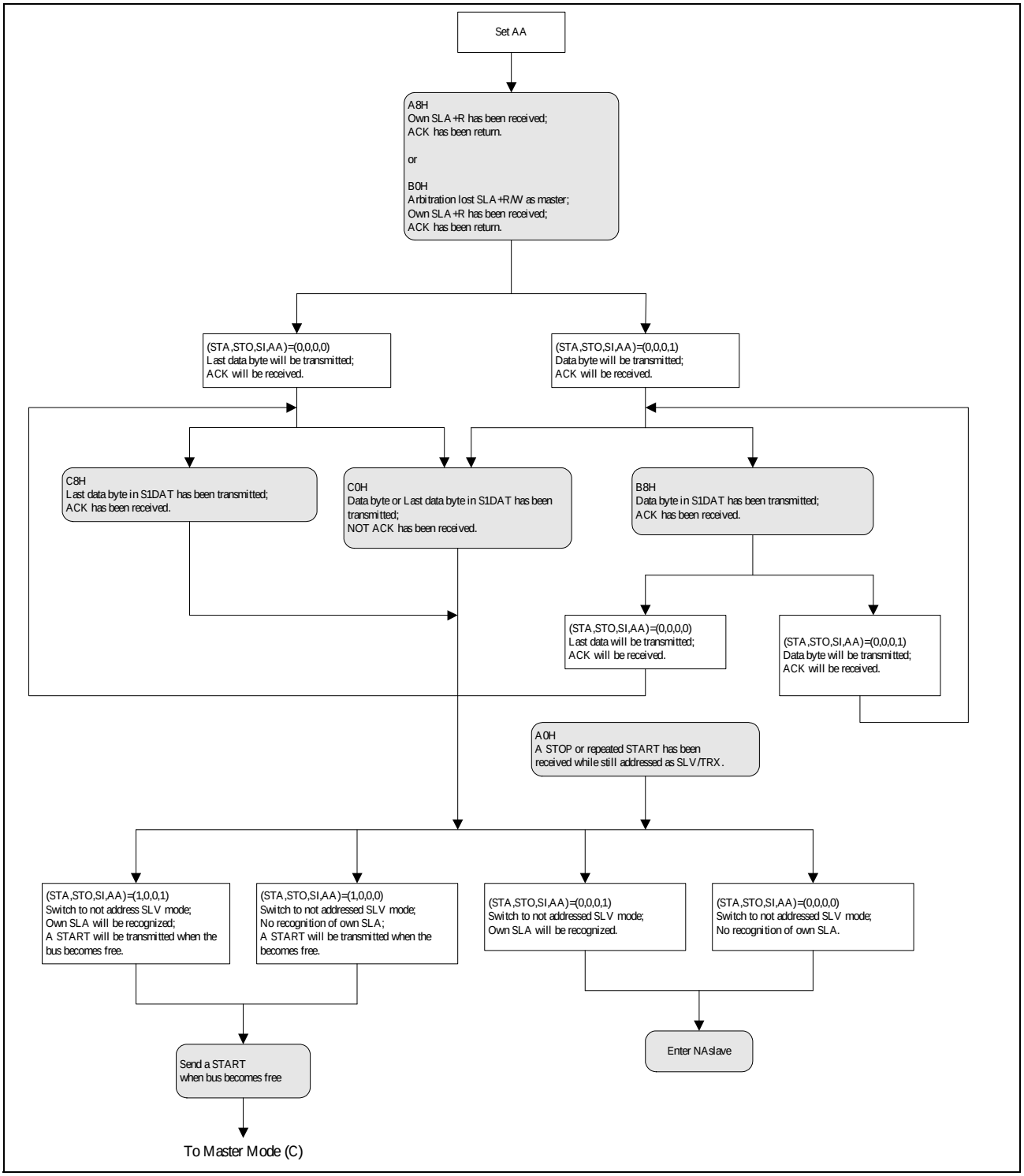

Set AA
A8H
Own SLA +R has been received;
ACK has been return.
or
B0H
Arbitration lost SLA +R/W as master;
Own SLA +R has been received;
ACK has been return.
(STA,STO,SI,AA)=(0,0,0,0)
Last data byte will be transmitted;
ACK will be received.
(STA,STO,SI,AA)=(0,0,0,1)
Data byte will be transmitted;
ACK will be received.
C8H
Last data byte in S1DAT has been transmitted;
ACK has been received.
C0H
Data byte or Last data byte in S1DAT has been transmitted;
NOT ACK has been received.
B8H
Data byte in S1DAT has been transmitted;
ACK has been received.
(STA,STO,SI,AA)=(0,0,0,0)
Last data will be transmitted;
ACK will be received.
(STA,STO,SI,AA)=(0,0,0,1)
Data byte will be transmitted;
ACK will be received.
A0H
A STOP or repeated START has been received while still addressed as SLV/TRX.
(STA,STO,SI,AA)=(1,0,0,1)
Switch to not address SLV mode;
Own SLA will be recognized;
A START will be transmitted when the bus becomes free.
(STA,STO,SI,AA)=(1,0,0,0)
Switch to not addressed SLV mode;
No recognition of own SLA;
A START will be transmitted when the becomes free.
(STA,STO,SI,AA)=(0,0,0,1)
Switch to not addressed SLV mode;
Own SLA will be recognized.
(STA,STO,SI,AA)=(0,0,0,0)
Switch to not addressed SLV mode;
No recognition of own SLA.
Send a START
when bus becomes free
Enter NAslave
To Master Mode (C)

图 5-25 从机传输模式

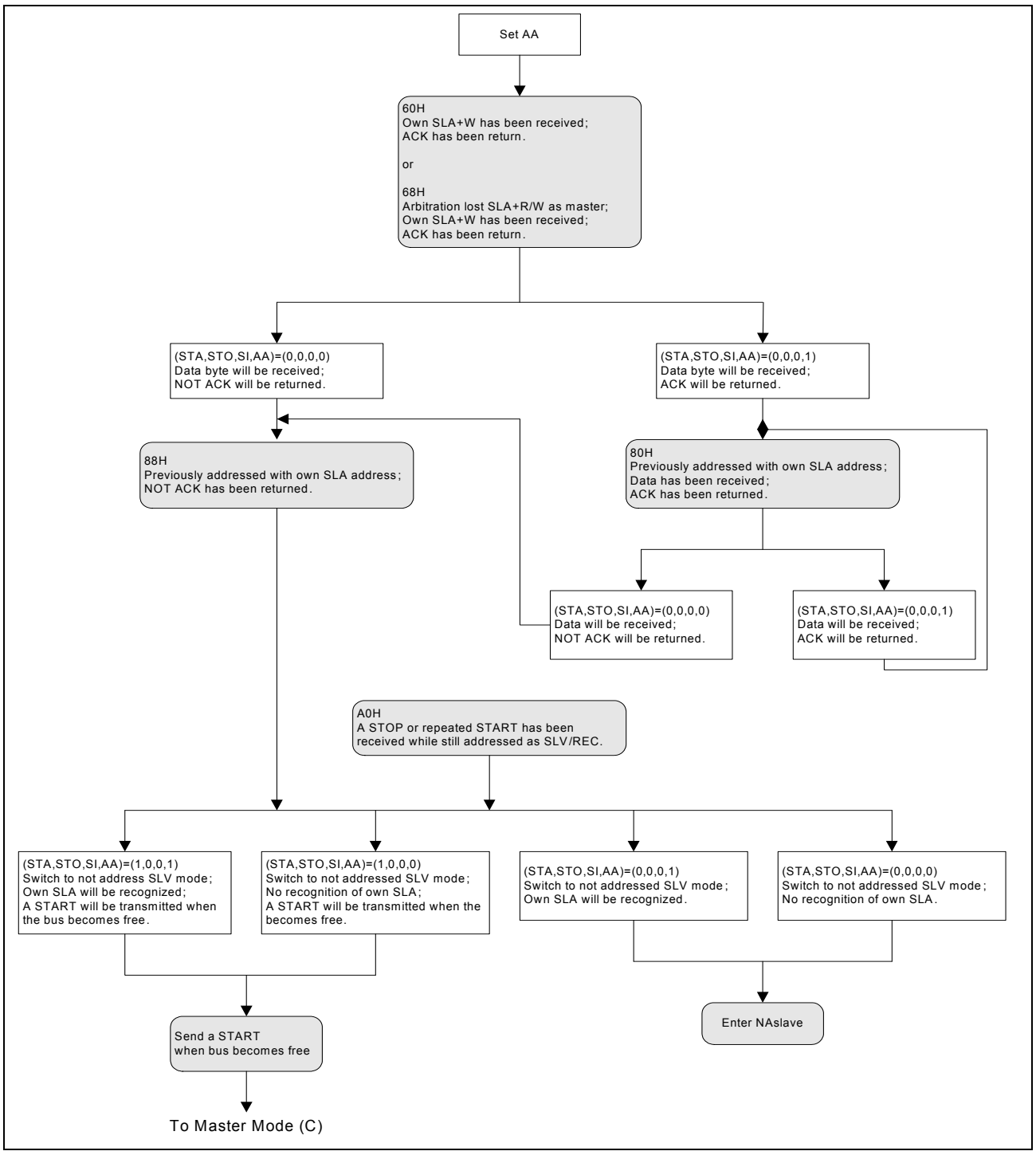

Set AA
60H
Own SLA+W has been received;
ACK has been return.
or
68H
Arbitration lost SLA+R/W as master;
Own SLA+W has been received;
ACK has been return.
(STA,STO,SI,AA)=(0,0,0,0)
Data byte will be received;
NOT ACK will be returned.
(STA,STO,SI,AA)=(0,0,0,1)
Data byte will be received;
ACK will be returned.
88H
Previously addressed with own SLA address;
NOT ACK has been returned.
80H
Previously addressed with own SLA address;
Data has been received;
ACK has been returned.
(STA,STO,SI,AA)=(0,0,0,0)
Data will be received;
NOT ACK will be returned.
(STA,STO,SI,AA)=(0,0,0,1)
Data will be received;
ACK will be returned.
A0H
A STOP or repeated START has been
received while still addressed as SLV/REC.
(STA,STO,SI,AA)=(1,0,0,1)
Switch to not address SLV mode;
Own SLA will be recognized;
A START will be transmitted when
the bus becomes free.
(STA,STO,SI,AA)=(1,0,0,0)
Switch to not addressed SLV mode;
No recognition of own SLA;
A START will be transmitted when the
becomes free.
(STA,STO,SI,AA)=(0,0,0,1)
Switch to not addressed SLV mode;
Own SLA will be recognized.
(STA,STO,SI,AA)=(0,0,0,0)
Switch to not addressed SLV mode;
No recognition of own SLA.
Send a START
when bus becomes free
Enter NAslave
To Master Mode (C)

图 5-26 从机接收模式

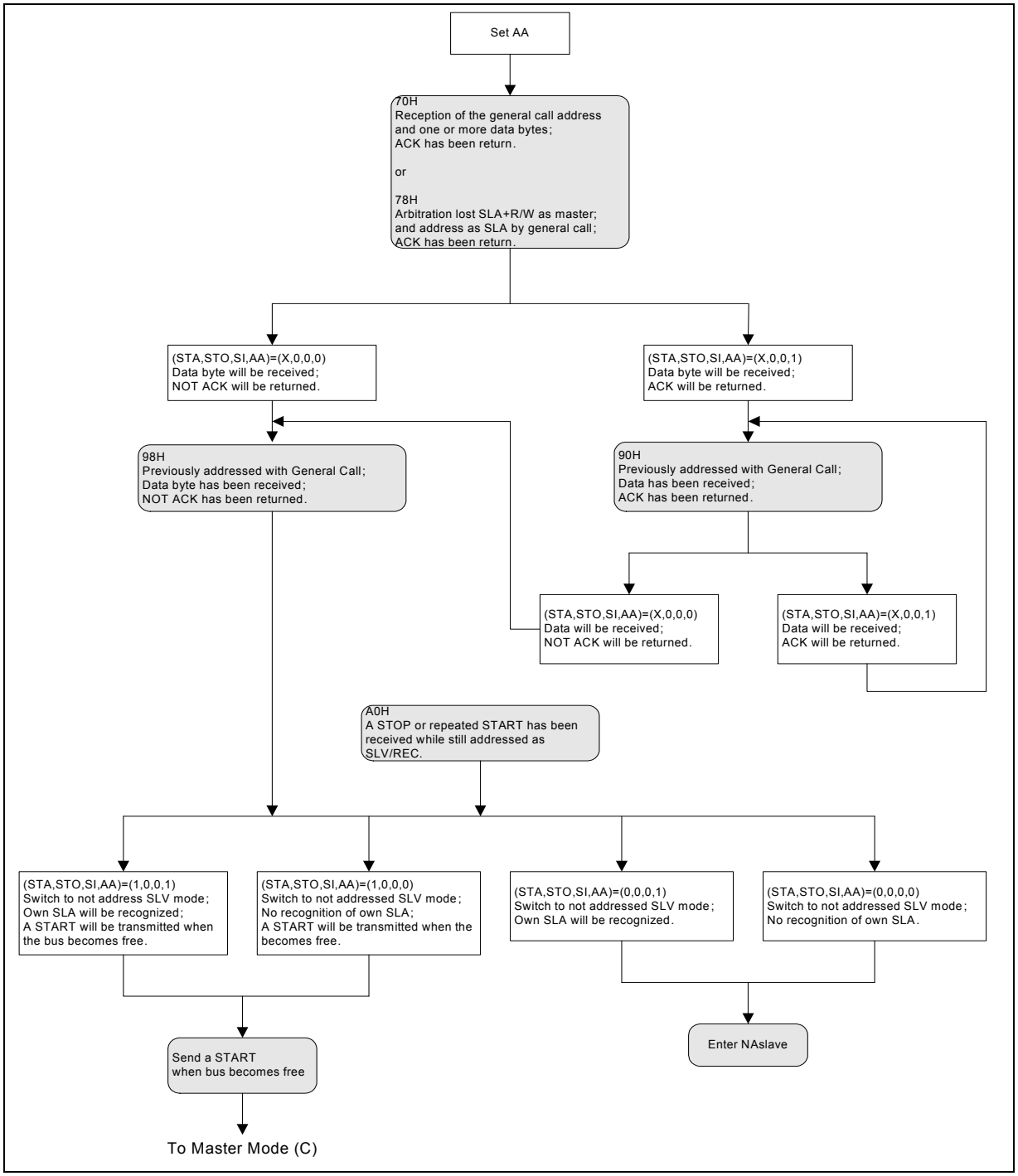

Set AA
70H
Reception of the general call address and one or more data bytes;
ACK has been return.
or
78H
Arbitration lost SLA+R/W as master;
and address as SLA by general call;
ACK has been return.
(STA,STO,SI,AA)=(X,0,0,0)
Data byte will be received;
NOT ACK will be returned.
(STA,STO,SI,AA)=(X,0,0,1)
Data byte will be received;
ACK will be returned.
98H
Previously addressed with General Call;
Data byte has been received;
NOT ACK has been returned.
90H
Previously addressed with General Call;
Data has been received;
ACK has been returned.
(STA,STO,SI,AA)=(X,0,0,0)
Data will be received;
NOT ACK will be returned.
(STA,STO,SI,AA)=(X,0,0,1)
Data will be received;
ACK will be returned.
A0H
A STOP or repeated START has been received while still addressed as SLV/REC.
(STA,STO,SI,AA)=(1,0,0,1)
Switch to not address SLV mode;
Own SLA will be recognized;
A START will be transmitted when the bus becomes free.
(STA,STO,SI,AA)=(1,0,0,0)
Switch to not addressed SLV mode;
No recognition of own SLA;
A START will be transmitted when the becomes free.
(STA,STO,SI,AA)=(0,0,0,1)
Switch to not addressed SLV mode;
Own SLA will be recognized.
(STA,STO,SI,AA)=(0,0,0,0)
Switch to not addressed SLV mode;
No recognition of own SLA.
Send a START
when bus becomes free
Enter NAslave
To Master Mode (C)

图 5-27 GC 模式

## 5.7 PWM 发生器和捕捉定时器

### 5.7.1 简介

包含4组PWM发生器，可配置为8个独立的PWM输出，PWM0~PWM7，或4组互补的PWM对, (PWM0, PWM1), (PWM2, PWM3), (PWM4, PWM5) 与 (PWM6, PWM7) ，带4个可编程的死区发生器. 每组PWM发生器带有8位预分频，一个时钟除频提供5级时钟源(1, 1/2, 1/4, 1/8, 1/16), 两个PWM定时器包括2个时钟选择，两个16位PWM向下计数计数器用于PWM period 周期控制，两个16位比较器用于PWM duty 周期控制以及死区发生. 4组PWM发生器提供8个独立的PWM中断标志，当PWM向下计数周期达到零时触发中断。PWM发生器可以定义为单触发模式或连续输出PWM波形.

当PCR.DZEN01置位, PWM0 与 PWM1形成互补的PWM周期，这一对PWM的timing, period, duty 和死区时间由PWM0定时器和死区发生器0决定. 同样, PWM 互补对(PWM2, PWM3), (PWM4, PWM5) 与 (PWM6, PWM7) 分别由 PWM2, PWM4 与 PWM6 定时器和死区发生器2，4，6控制，参考图 5-29和图 5-30查看PWM定时器内部结构.

为防止PWM输出抖动不稳定波形，16位向下计数计数器和16位比较器采用两组缓存。当用户向计数器/比较器寄存器内写入值，只有当计数器/比较器的值计数到0后，才会被重新写入计数器/比较器。该两组缓存可以避免PWM输出时产生干扰波形.

当16位向下计数计数器达到0时，中断请求产生。如果PWM定时器被定义为连续模式，当向下计数器达到0时，会自动重新导入设定值(CNRx)并从新开始运行下一个周期。如果定时器设为单触发模式，向下计数器停止计数，并产生中断请求.

比较器数据用于设定脉宽，计数器控制逻辑可以改变输出.

当PWM输出模块的输入捕捉功能使能，可同时用作捕捉功能。捕捉器0和PWM0使用同一个定时器，捕捉器1和PWM1使用另一组定时器，以此类推。在使用捕捉功能之前，必须预先配置PMW定时器。输入端口锁定**上升沿**，作为捕捉器的计数存入**CRLR**，并锁定PWM计数器的**下降沿**，存入**CFLR**。 设定CCR0[1] (上升沿触发中断有效)和CCR0[2](下降沿触发中断有效),可以使捕捉器通道0作为中断源。同样设定CCR0[17] 和CCR0[18]，可以设定通道1。通道2 和3同样通过设定CCR1[1],CCR1[2] 和CCR1[17], CCR1[18] 无论捕捉模块是否触发中断，PWM计数器都将重置

### 5.7.2 特性

#### *5.7.2.1 PWM 功能特性:*

- 4组PWM 发生器，每个支持8位预分频，一个时钟除频，PWM定时器，一个死区发生器和两个PWM输出
- 最多8通道PWM，或4组成对的PWM通道.
- 最高16位解析度.
- PWM 中断与PWM周期同步.
- 单触发模式或连续模式.
- 四组死区发生器

#### *5.7.2.2 捕捉功能模块:*

- 与PWM发生器共用定时器模块.
- 8路捕捉输入通道.
- 每个通道支持1个上升沿锁存寄存器(CRLR)，一个下降沿锁存寄存器 (CFLR) 和捕捉中断标志(CAPIFx)

### 5.7.3 PWM发生器结构

附件图片显示一组PWM发生器结构(定时器0和1为一组，一组为定时器2和3,等等).

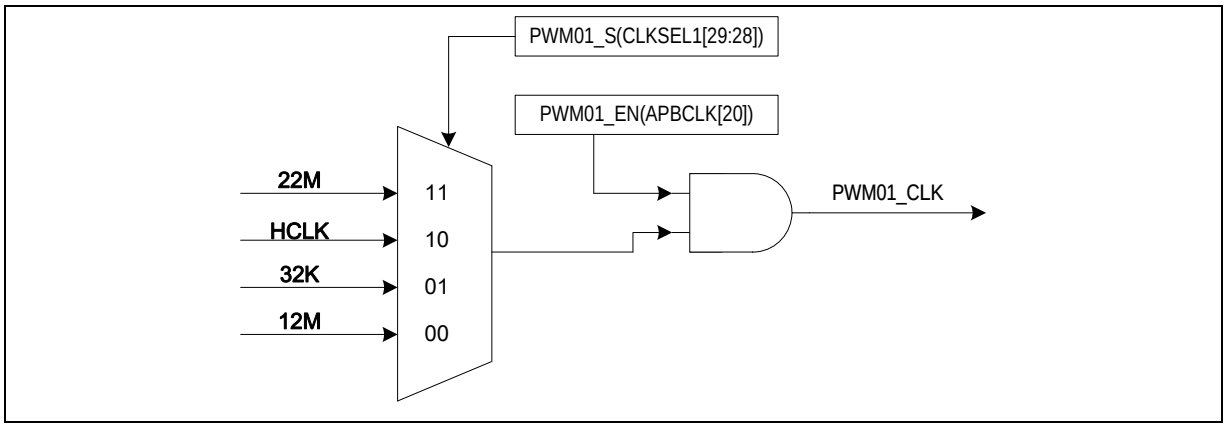



图 5-28 PWM 发生器 0 时钟源控制

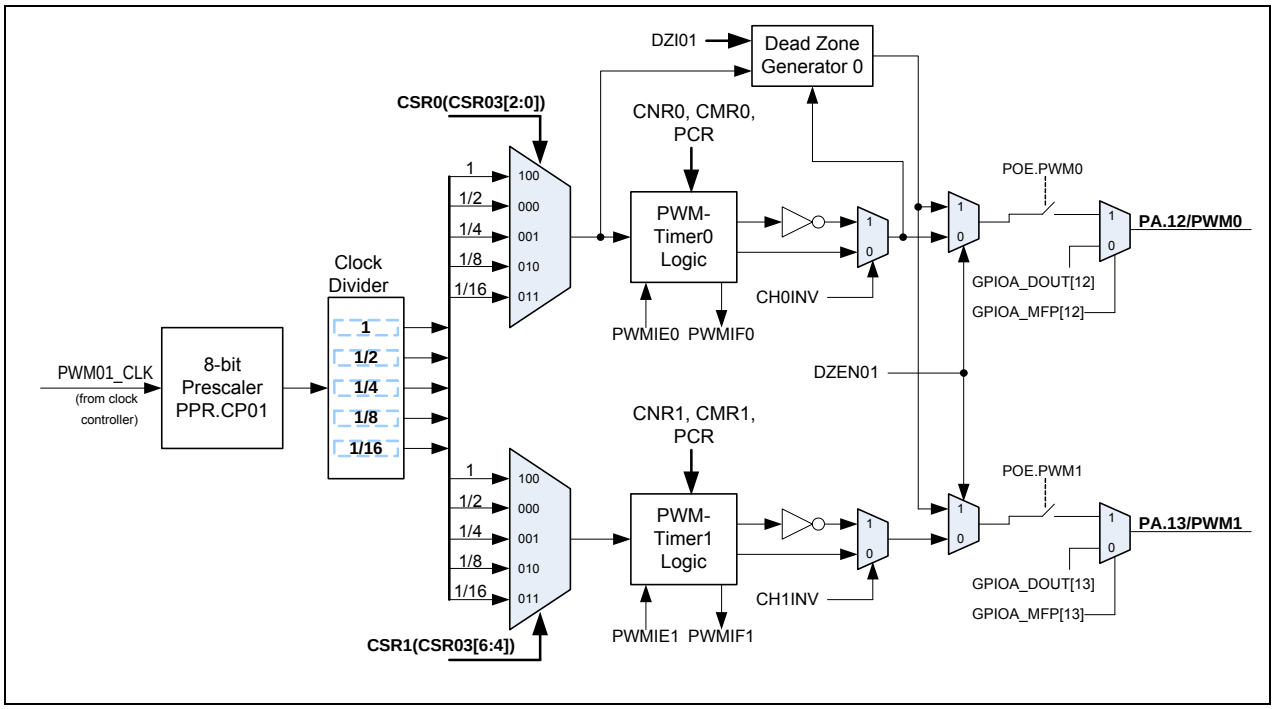



图 5-29 PWM 发生器 0 结构图

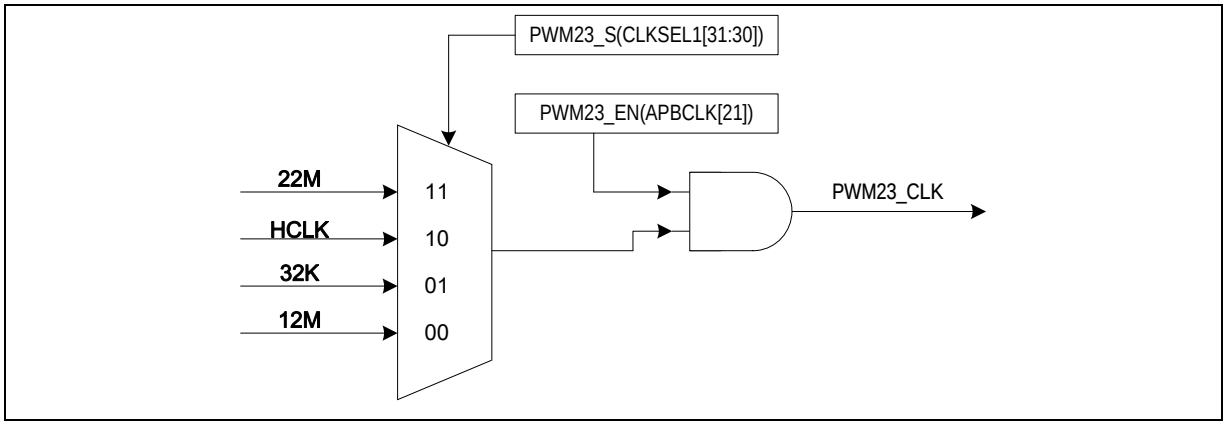



图 5-30 PWM 发生器 2 时钟源控制

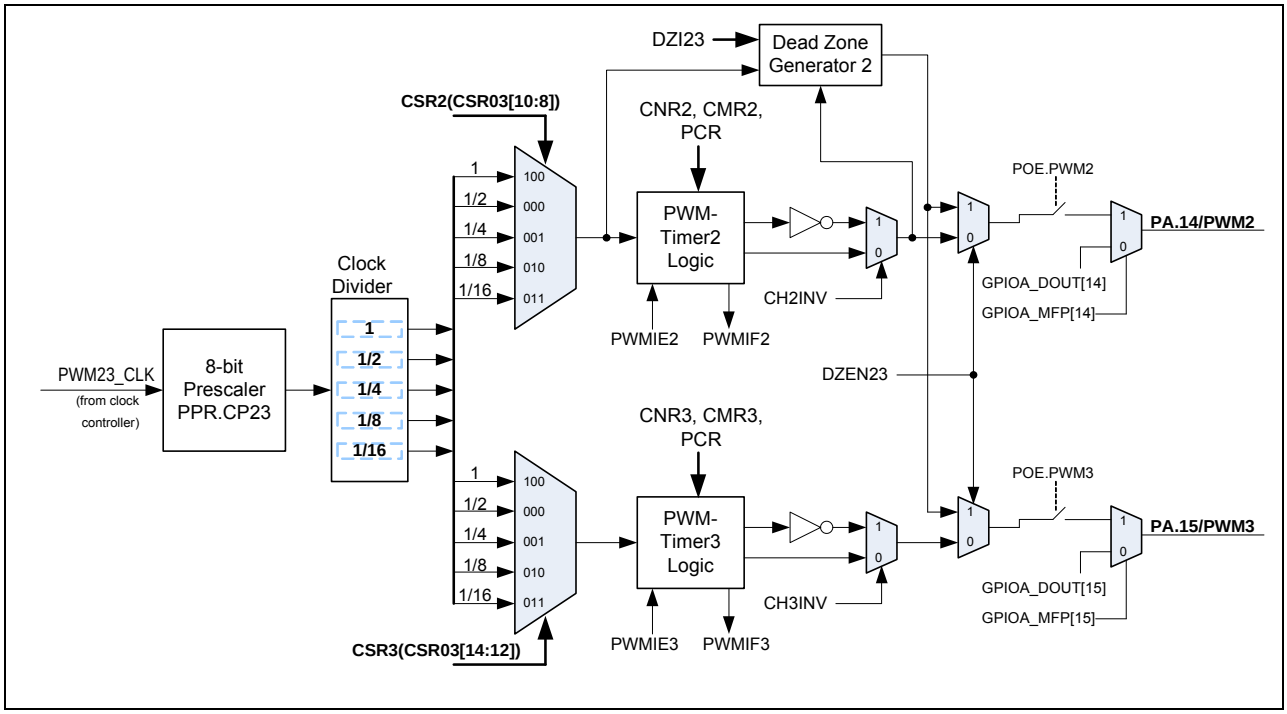



图 5-31 PWM 发生器 2 结构图

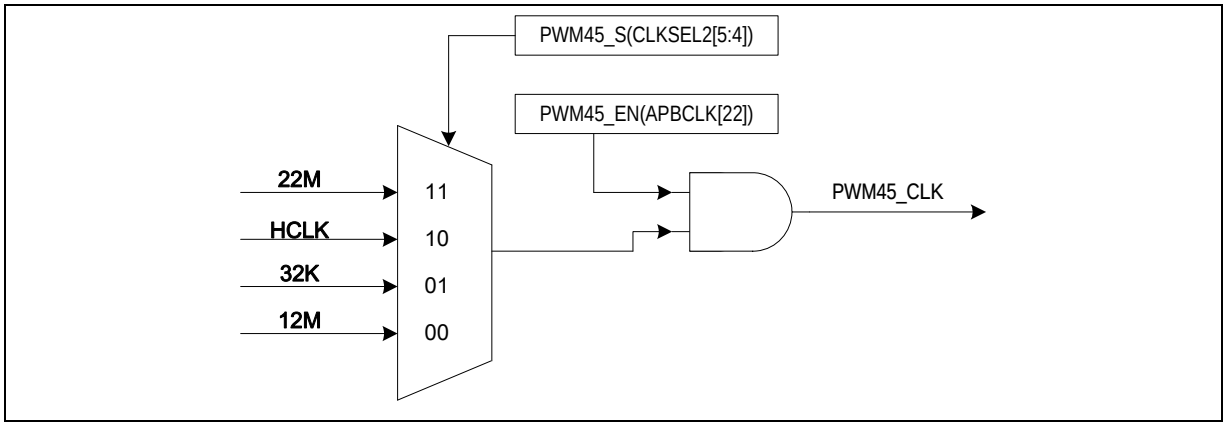



图 5-32 PWM 发生器 4 时钟源控制

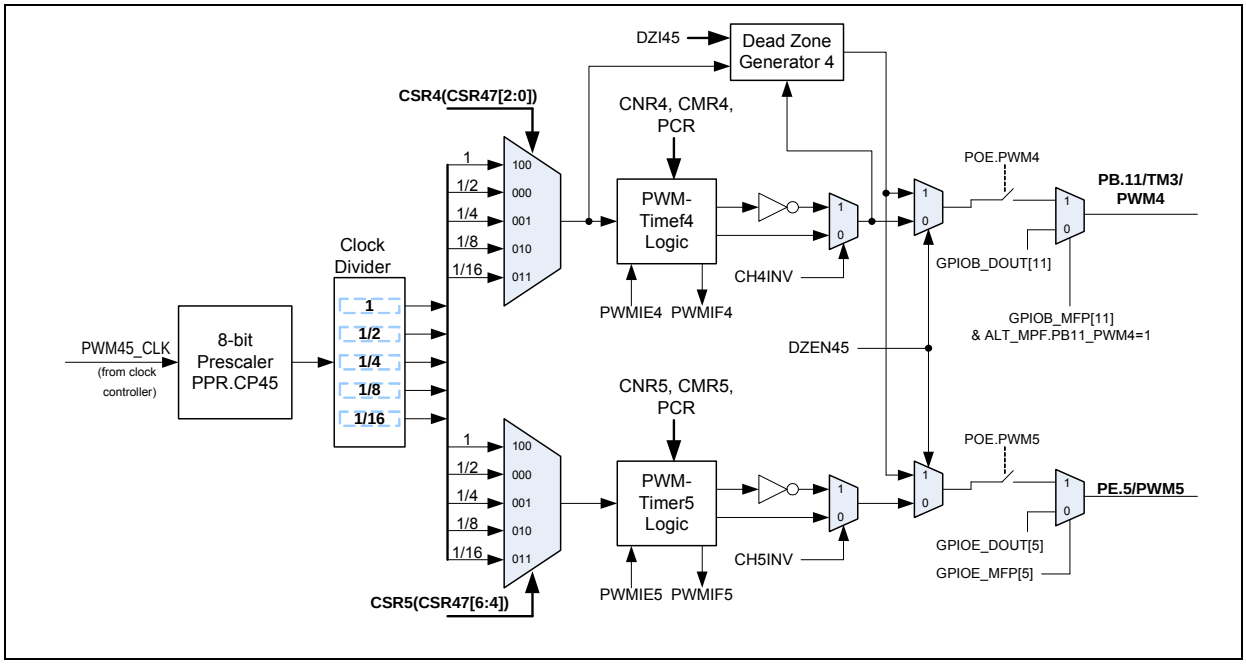



图 5-33 PWM 发生器 4 结构图

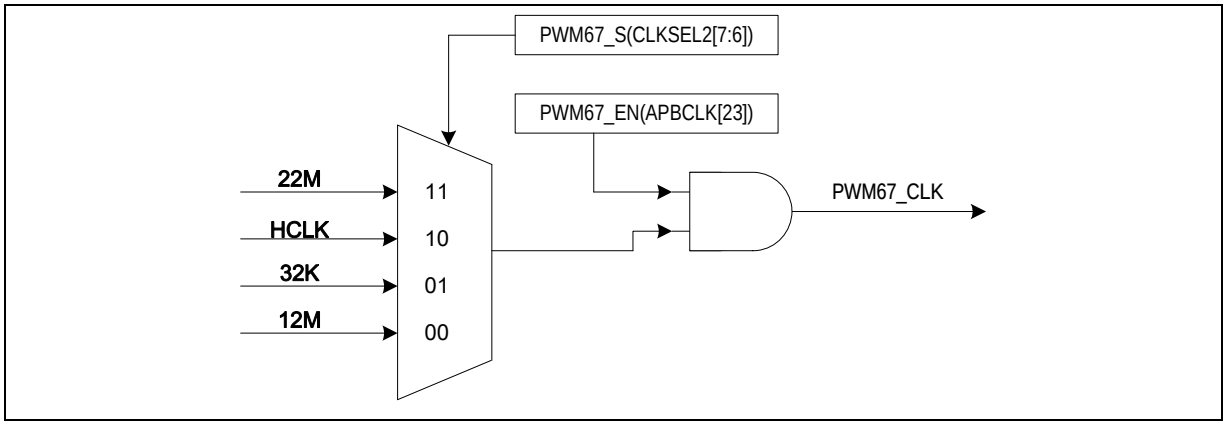



图 5-34 PWM 发生器 6 时钟源控制

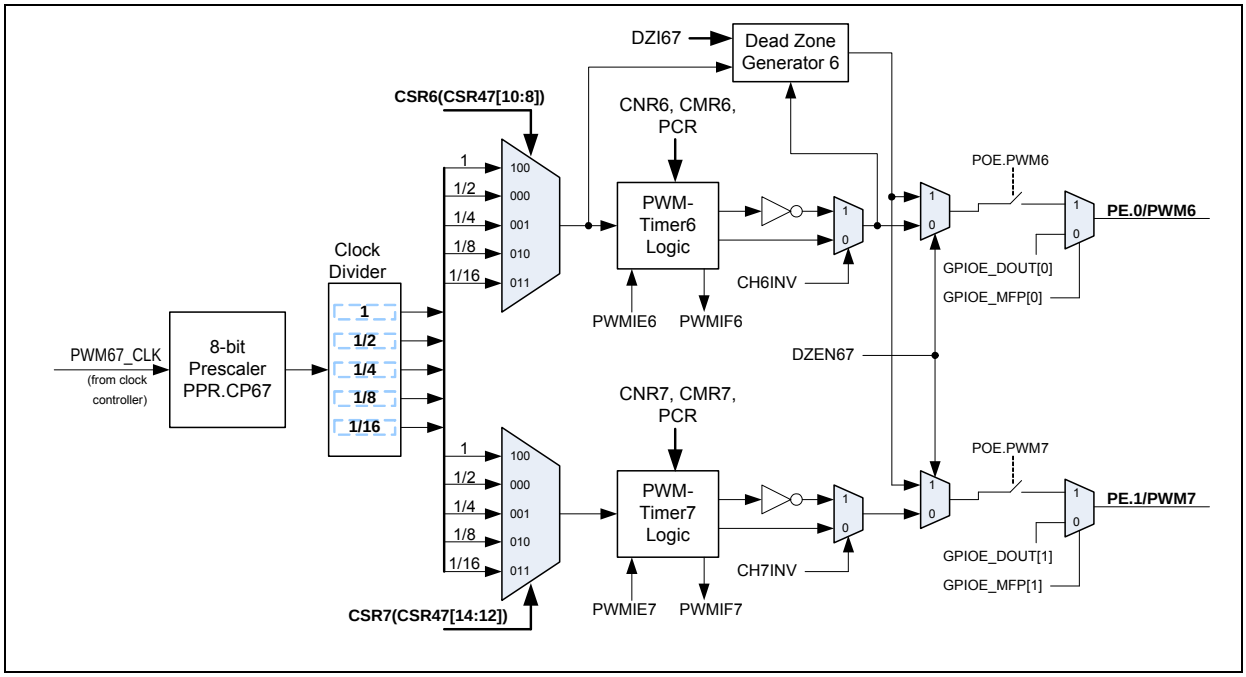



图 5-35 PWM 发生器 6 结构图

### 5.7.4 PWM定时器操作

PWM period 和duty 控制由向下计数的PWM寄存器(CNR)以及PWM比较寄存器(CMR)控制。PWM定时器工作时序如图 5-37. 脉宽调制的公式如下，相应的周期如图 5-36. 注：在PWM功能使能前，MCU相应管脚应配置我i输出模式(GPIOA_OMD).

- PWM 频率 = PWMxy_CLK/(prescale+1)*(clock divider)/(CNR+1); xy代表01, 23, 45 或 67, 取决于所选择的PWM通道.
- 占空比 = (CMR+1)/(CNR+1).
- CMR >= CNR: PWM 输出为高.
- CMR < CNR: PWM低脉宽= (CNR-CMR) unit[1]; PWM高脉宽=(CMR+1) unit.
- CMR = 0: PWM低脉宽 = (CNR) unit; PWM高脉宽= 1 unit

注: 1. 单位 = 一个PWM时钟周期.

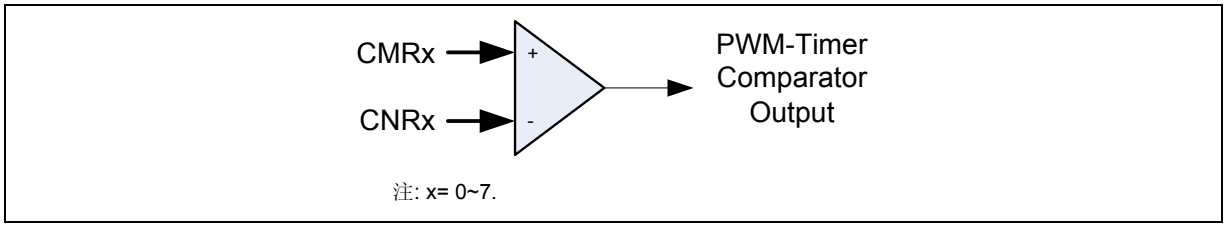



图 5-36 PWM 定时器内部比较器输出

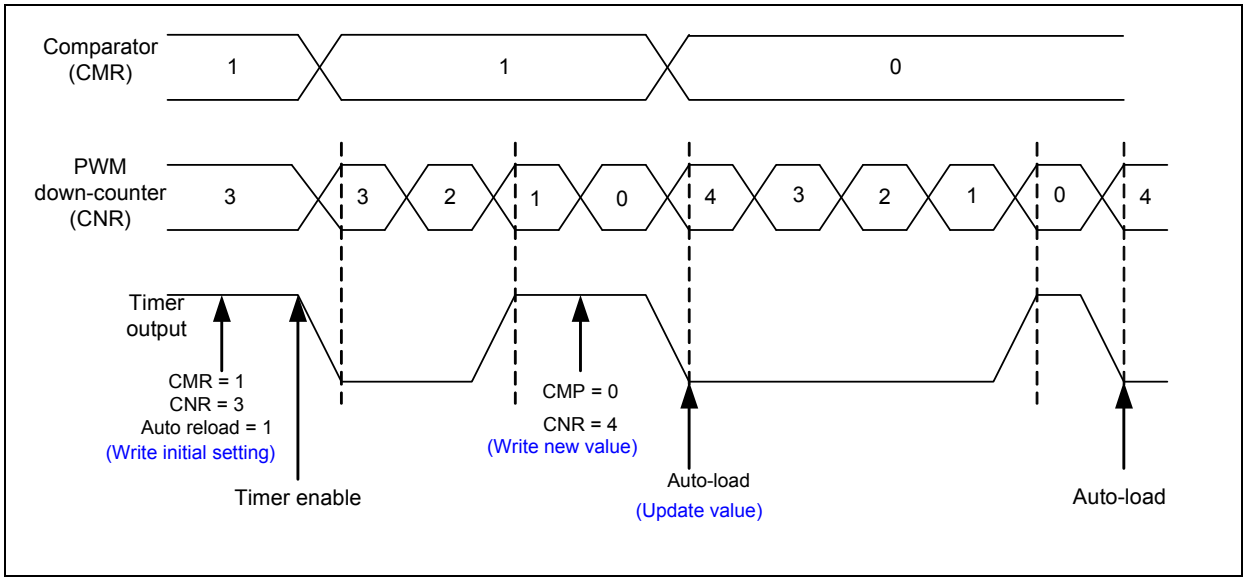



图 5-37 PWM 定时器操作时序

### 5.7.5 PWM 双缓存，自动重载以及单触发模式

NUC1XX 系列PWM定时器具有双缓存功能。寄存器预先设定的值，在一个周期完成后，可以自动重载。PWM计数器值写入CNR0~7，并可从PDR0~7内读出.

PWM 控制寄存器(PCR) 的CH0MOD 位定义PWM0是自动重载模式或是单触发模式。自动重载模式下，当PWM计数器计到0，MCU自动重载CNR0 值到PWM 计数器。如果CNR0 设定为0，PWM计数器计数到0后，暂停运行查询状态。如果此时CH0MOD设定为0 ，计数器停止运行。PWM1~PWM7 运行状态与PWM0 相同.

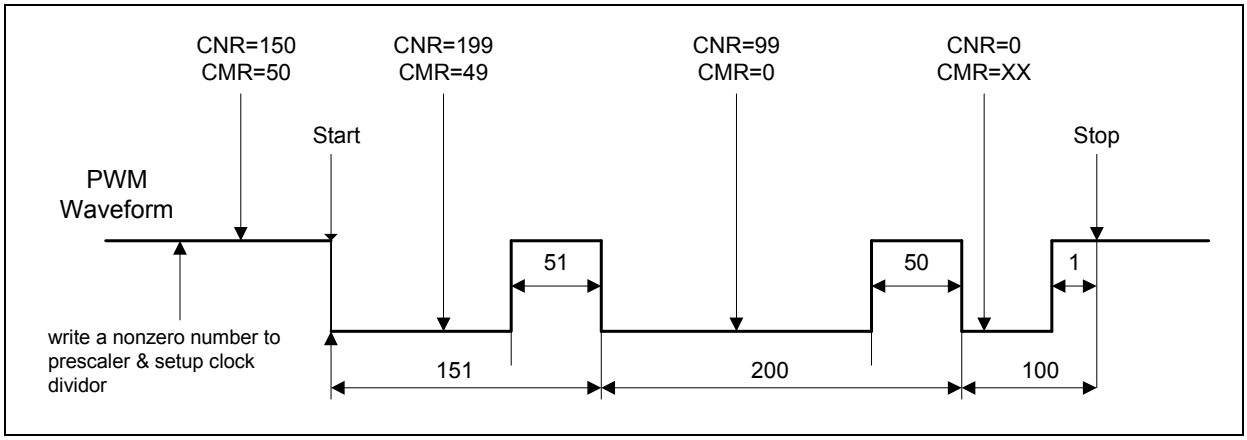



图 5-38 PWM 双缓存图解

### 5.7.6 可调占空比

双缓存允许CMR字当前运行时改写。下一个周期内值被导入运行.

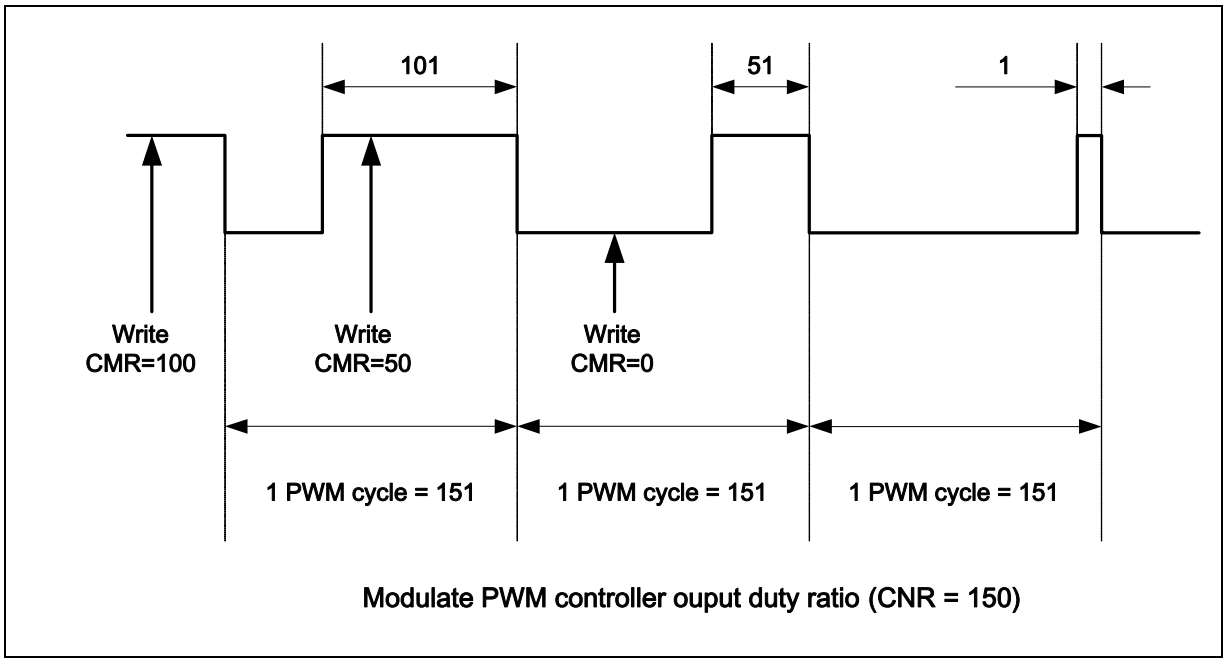



图 5-39 PWM 控制输出占空比

### 5.7.7 死区发生器

NUC1XX 系列提供PWM死区发生器，用于保护器件电源。该功能产生可编程的延迟时间到PWM上升沿输出，用户通过编程PPRx.DZI确定死区间隔.

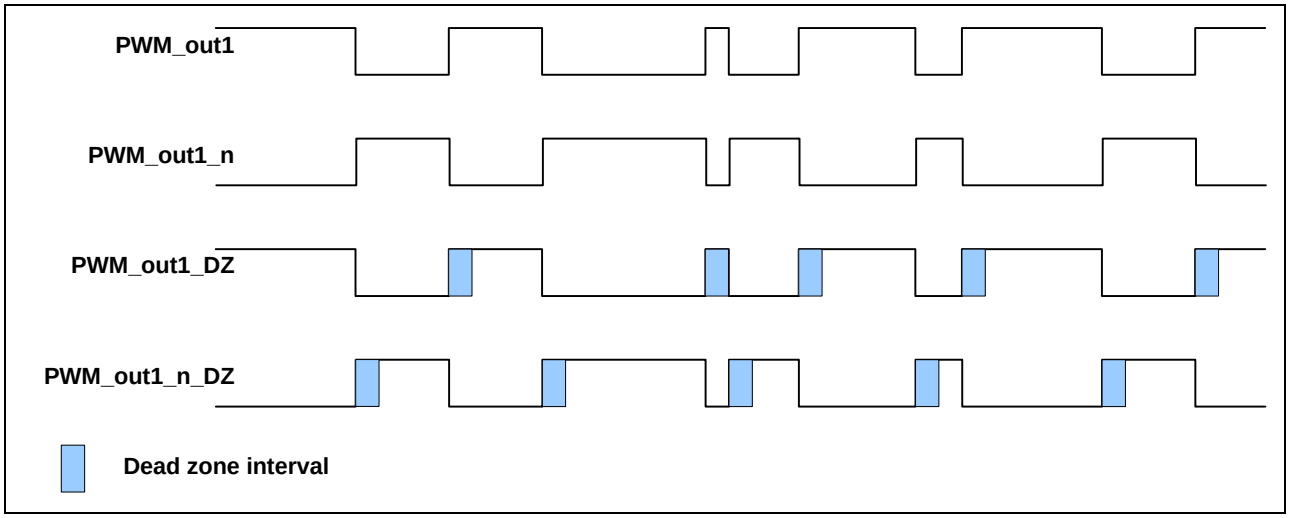



图 5-40 死区发生器操作

### 5.7.8 捕捉模式

捕捉器0和PWM0使用同一个定时器，捕捉器1和PWM1使用另一组定时器，以此类推。在使用捕捉功能之前，必须预先配置PMW定时器。输入端口锁定**上升沿**，作为捕捉器的计数存入**CRLR**，并锁定PWM计数器的**下降沿**，存入**CFLR**。 设定CCR0[1] (上升沿触发中断有效)和CCR0[2](下降沿触发中断有效), 可以使捕捉器通道0作为中断源。同样设定CCR0[17] 和CCR0[18]，可以设定通道1。通道2 和3同样通过设定CCR1[1],CCR1[2] 和CCR1[17], CCR1[18] 无论捕捉模块是否触发中断，PWM计数器都将重置。
注: 在捕捉功能使能前，相应的管脚必须配置为输入模式(GPIOA_OMD)

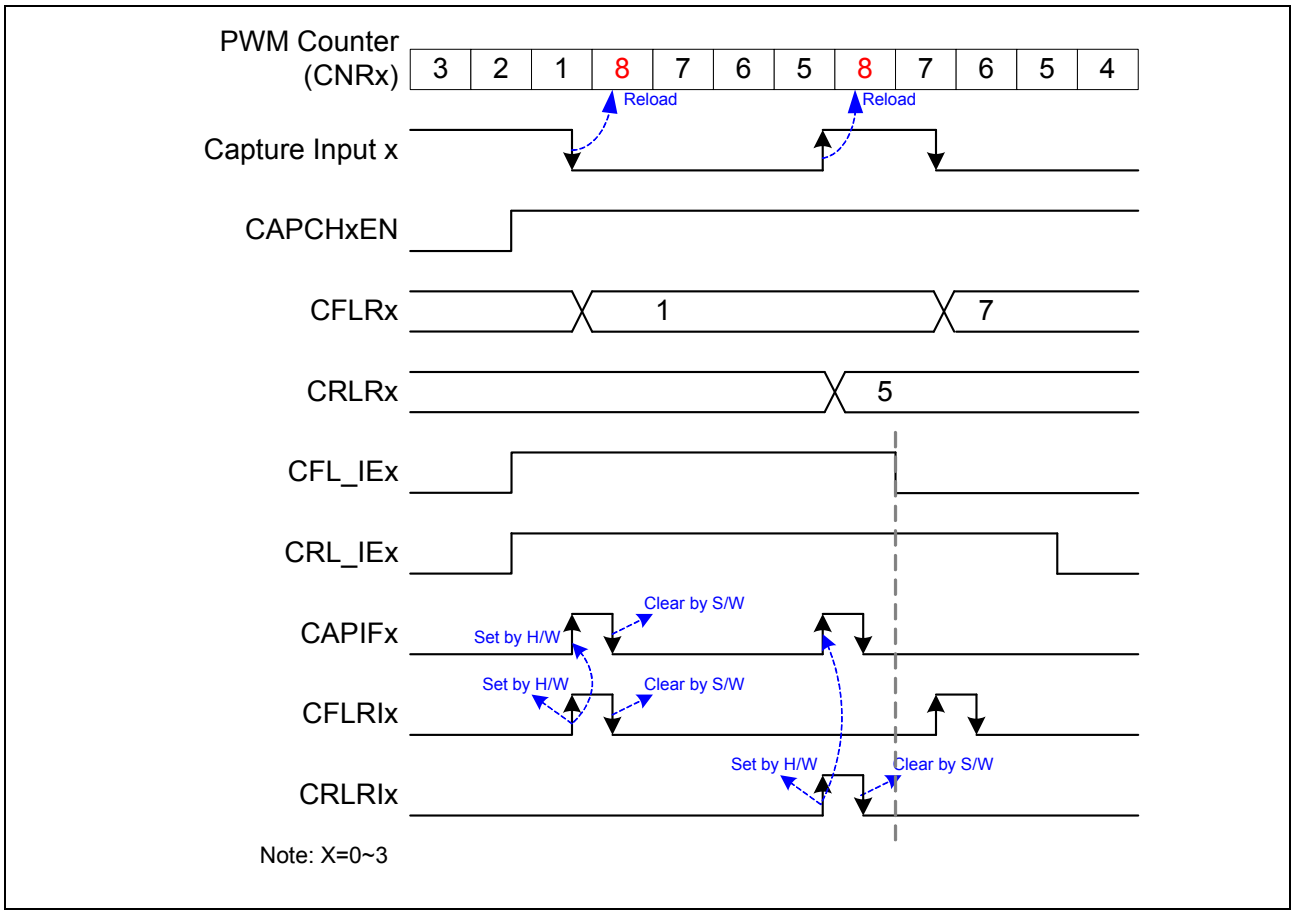



图 5-41 捕捉操作时序

在上述范例中，CNR 为8:

1. 当上升与下降沿中断同时使能，PWM计数器在中断发生同时进行数据重载。
2. 通道低脉宽为(CNR – CRLR).
3. 通道高脉宽为 (CNR - CFLR).

### 5.7.9 PWM定时器中断结构

NUC1XX提供PWM八级中断，PWM0_INT~PWM7_INT,增强型中断控制寄存器(AIC)配置。 PWM 0 与捕捉器0共用同一组中断。PWM1 与捕捉器1 共用同一组中断，以此类推。因此，PWM中断和捕捉中断不能同时发生。图 5-42 PWM定时器中断结构如下

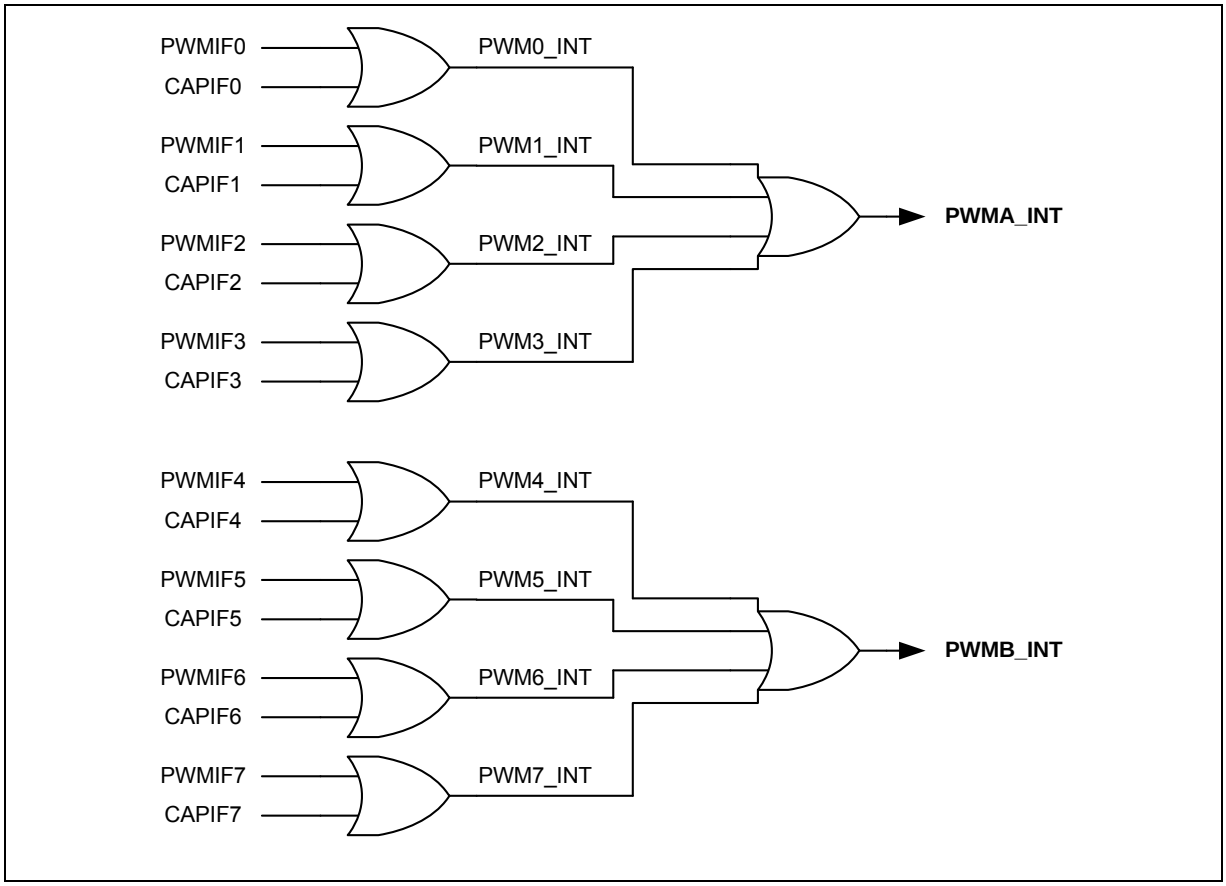



图 5-42 PWM 定时器中断结构图

### 5.7.10 PWM定时器开启步骤

下列步骤用于启动PWM

1. 配置时钟选择 (CSR)
2. 配置预分频(PPR)
3. 配置反向打开/关闭，死区打开/关闭，自动重载/单触发模式以及PWM定时器关闭 (PCR)
4. 配置比较器寄存器(CMR) 设定PWM duty.
5. 配置PWM计数器寄存器 (CNR) 设定period.
6. 配置中断使能寄存器 (PIER)
7. 配置PWM输出使能 (POE)
8. 配置相应的管脚GPIO (GPA_MFP)
9. 配置相应的管教到输出模式 (GPIOA_OMD)
10. 使能PWM定时器(PCR)

### 5.7.11 PWM定时器关闭步骤

**方式 1:**

设定16位向下计数计数器(CNR)为0 ， 并查看PDR状态。当PDR达到0 ， 关闭PWM定时器 (PCR的CHxEN位). ***(推荐)***

**方式2:**

设定16位向下计数计数器(CNR)为0，当中断条件发生。在中断内关闭PWM定时器(PCR的CHxEN位). ***(推荐)***

**方式3:**

直接关闭PWM定时器(PCR的CHxEN位). (不推荐)

### 5.7.12 捕捉开始步骤

1. 配置时钟选择(CSR)
2. 配置预分频(PPR)
3. 配置通道使能，上升/下降沿中断使能以及输入信号反向打开/关闭 (CCR0, CCR1)
4. 配置PWM计数器寄存器 (CNR)
5. 配置捕捉输入使能寄存器 (CAPENR)
6. 配置相应的GPIO管脚用于PWM功能 (GPA_MFP)
7. 配置相应的GPIO管脚于输入模式S (GPIOA_OMD)
8. 使能PWM定时器(PCR)

### 5.7.13 寄存器列表

**R: 只读, W: 只写, R/W: 可读写**, **C**: 仅在值为0时可写

**PWM0~3寄存器.**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **PWM_BA0 = 0x4004_0000** | | | | |
| **PPR03** | PWM_BA0+0x000 | R/W | PWM预分频寄存器 | 0x0000_0000 |
| **CSR03** | PWM_BA0+0x004 | R/W | PWM 时钟选择寄存器 | 0x0000_0000 |
| **PCR03** | PWM_BA0+0x008 | R/W | PWM 控制寄存器 | 0x0000_0000 |
| **CNR0** | PWM_BA0+0x00C | R/W | PWM 计数寄存器0 | 0x0000_0000 |
| **CMR0** | PWM_BA0+0x010 | R/W | PWM 比较寄存器0 | 0x0000_0000 |
| **PDR0** | PWM_BA0+0x014 | R | PWM 数据寄存器0 | 0x0000_0000 |
| **CNR1** | PWM_BA0+0x018 | R/W | PWM 计数寄存器1 | 0x0000_0000 |
| **CMR1** | PWM_BA0+0x01C | R/W | PWM 比较寄存器1 | 0x0000_0000 |
| **PDR1** | PWM_BA0+0x020 | R | PWM 数据寄存器1 | 0x0000_0000 |
| **CNR2** | PWM_BA0+0x024 | R/W | PWM 计数寄存器2 | 0x0000_0000 |
| **CMR2** | PWM_BA0+0x028 | R/W | PWM 比较寄存器2 | 0x0000_0000 |
| **PDR2** | PWM_BA0+0x02C | R | PWM 数据寄存器2 | 0x0000_0000 |
| **CNR3** | PWM_BA0+0x030 | R/W | PWM 计数寄存器3 | 0x0000_0000 |
| **CMR3** | PWM_BA0+0x034 | R/W | PWM 比较寄存器3 | 0x0000_0000 |
| **PDR3** | PWM_BA0+0x038 | R | PWM 数据寄存器3 | 0x0000_0000 |
| **PIER03** | PWM_BA0+0x040 | R/W | PWM 中断使能寄存器 | 0x0000_0000 |
| **PIIR03** | PWM_BA0+0x044 | R/C | PWM 中断标志寄存器 | 0x0000_0000 |
| **CCR0** | PWM_BA0+0x050 | R/W | 捕捉控制寄存器0 | 0x0000_0000 |
| **CCR2** | PWM_BA0+0x054 | R/W | 捕捉控制寄存器1 | 0x0000_0000 |
| **CRLR0** | PWM_BA0+0x058 | R/W | 捕捉上升沿锁存寄存器(通道0) | 0x0000_0000 |
| **CFLR0** | PWM_BA0+0x05C | R/W | 捕捉下降沿锁存寄存器(通道0) | 0x0000_0000 |
| **CRLR1** | PWM_BA0+0x060 | R/W | 捕捉上升沿锁存寄存器(通道1) | 0x0000_0000 |
| **CFLR1** | PWM_BA0+0x064 | R/W | 捕捉下降沿锁存寄存器(通道1) | 0x0000_0000 |

| | | | | |
|---|---|---|---|---|
| **CRLR2** | PWM_BA0+0x068 | R/W | 捕捉上升沿锁存寄存器(通道2) | 0x0000_0000 |
| **CFLR2** | PWM_BA0+0x06C | R/W | 捕捉下降沿锁存寄存器(通道2) | 0x0000_0000 |
| **CRLR3** | PWM_BA0+0x070 | R/W | 捕捉上升沿锁存寄存器(通道3) | 0x0000_0000 |
| **CFLR3** | PWM_BA0+0x074 | R/W | 捕捉下降沿锁存寄存器(通道3) | 0x0000_0000 |
| **CAPENR03** | PWM_BA0+0x078 | R/W | 捕捉输入使能寄存器 | 0x0000_0000 |
| **POE03** | PWM_BA0+0x07C | R/W | PWM 输出使能 | 0x0000_0000 |

**PWM4~7寄存器.**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| PWM_BA4 = 0x4014_0000 | | | | |
| PPR47 | PWM_BA4+0x000 | R/W | PWM 预分频寄存器 | 0x0000_0000 |
| CSR47 | PWM_BA4+0x004 | R/W | PWM 时钟选择寄存器 | 0x0000_0000 |
| PCR47 | PWM_BA4+0x008 | R/W | PWM 控制寄存器 | 0x0000_0000 |
| CNR4 | PWM_BA4+0x00C | R/W | PWM 计数寄存器4 | 0x0000_0000 |
| CMR4 | PWM_BA4+0x010 | R/W | PWM 比较寄存器4 | 0x0000_0000 |
| PDR4 | PWM_BA4+0x014 | R | PWM 数据寄存器4 | 0x0000_0000 |
| CNR5 | PWM_BA4+0x018 | R/W | PWM 计数寄存器5 | 0x0000_0000 |
| CMR5 | PWM_BA4+0x01C | R/W | PWM 比较寄存器5 | 0x0000_0000 |
| PDR5 | PWM_BA4+0x020 | R | PWM 数据寄存器5 | 0x0000_0000 |
| CNR6 | PWM_BA4+0x024 | R/W | PWM 计数寄存器6 | 0x0000_0000 |
| CMR6 | PWM_BA4+0x028 | R/W | PWM 比较寄存器6 | 0x0000_0000 |
| PDR6 | PWM_BA4+0x02C | R | PWM 数据寄存器6 | 0x0000_0000 |
| CNR7 | PWM_BA4+0x030 | R/W | PWM 计数寄存器7 | 0x0000_0000 |
| CMR7 | PWM_BA4+0x034 | R/W | PWM 比较寄存器7 | 0x0000_0000 |
| PDR7 | PWM_BA4+0x038 | R | PWM 数据寄存器7 | 0x0000_0000 |
| PIER47 | PWM_BA4+0x040 | R/W | PWM 中断使能寄存器 | 0x0000_0000 |
| PIIR47 | PWM_BA4+0x044 | R/C | PWM 中断标志寄存器 | 0x0000_0000 |
| CCR4 | PWM_BA4+0x050 | R/W | 捕捉控制寄存器4 | 0x0000_0000 |
| CCR6 | PWM_BA4+0x054 | R/W | 捕捉控制寄存器5 | 0x0000_0000 |
| CRLR4 | PWM_BA4+0x058 | R/W | 捕捉上升沿锁存寄存器(通道4) | 0x0000_0000 |
| CFLR4 | PWM_BA4+0x05C | R/W | 捕捉下降沿锁存寄存器(通道4) | 0x0000_0000 |
| CRLR5 | PWM_BA4+0x060 | R/W | 捕捉上升沿锁存寄存器(通道5) | 0x0000_0000 |
| CFLR5 | PWM_BA4+0x064 | R/W | 捕捉下降沿锁存寄存器(通道5) | 0x0000_0000 |

| | | | | |
|---|---|---|---|---|
| **CRLR6** | PWM_BA4+0x068 | R/W | 捕捉上升沿锁存寄存器(通道6) | 0x0000_0000 |
| **CFLR6** | PWM_BA4+0x06C | R/W | 捕捉下降沿锁存寄存器(通道6) | 0x0000_0000 |
| **CRLR7** | PWM_BA4+0x070 | R/W | 捕捉上升沿锁存寄存器(通道7) | 0x0000_0000 |
| **CFLR7** | PWM_BA4+0x074 | R/W | 捕捉下降沿锁存寄存器(通道7) | 0x0000_0000 |
| **CAPENR** | PWM_BA4+0x078 | R/W | 捕捉输入使能寄存器 | 0x0000_0000 |
| **POE** | PWM_BA4+0x07C | R/W | PWM 输出使能 | 0x0000_0000 |

### 5.7.14 寄存器描述

**PWM预分频寄存器(PPR03)**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **PPR03** | PWM_BA0+0x000 | R/W | PWM 预分频寄存器 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| **DZI23** | | | | | | | |
| **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| **DZI01** | | | | | | | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| **CP23** | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **CP01** | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [31:24] | **DZI23** | **PWM2 与 PWM3的死区间隔寄存器**<br>该8位寄存器决定死区长度.<br>每单位死区时间长度由时钟选择2决定。 |
| [23:16] | **DZI01** | **PWM0 与 PWM1的死区间隔寄存器**<br>该8位寄存器决定死区长度.<br>每单位死区时间长度由时钟选择0决定。 |
| [15:8] | **CP23** | **PWM定时器2 & 3的时钟预分频 2**<br>时钟输入计数器2&3之前，根据(CP23 + 1)除频<br>**如果CP23=0, 预分频器2输出时钟停止。** |
| [7:0] | **CP01** | **PWM定时器0 & 1的时钟预分频0**<br>时钟输入计数器0&1之前，根据(CP01 + 1)除频<br>**如果CP01=0, 预分频器0输出时钟停止。** |

### PWM 预分频寄存器(PPR47)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| PPR47 | PWM_BA4+0x000 | R/W | PWM预分频寄存器 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| DZI67 | | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| DZI45 | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| CP67 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CP45 | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [31:24] | **DZI67** | **PWM6 与 PWM7的死区间隔寄存器**<br>该8位寄存器决定死区长度.<br>每单位死区时间长度由时钟选择6决定. |
| [23:16] | **DZI45** | **PWM4 与 PWM5的死区间隔寄存器**<br>该8位寄存器决定死区长度.<br>每单位死区时间长度由时钟选择4决定. |
| [15:8] | **CP67** | **PWM定时器6 & 7的时钟预分频 6**<br>时钟输入计数器6&7之前，根据(CP67 + 1)除频<br>**如果CP67=0, 预分频器6输出时钟停止，因此PWM计数器6与7也停止。** |
| [7:0] | **CP45** | **PWM定时器4 &5的时钟预分频 0/4Clock prescaler 0/4**<br>时钟输入计数器4&5之前，根据(CP45 + 1)除频<br>**如果CP45=0, 预分频器0（4）输出时钟停止，因此PWM计数器4与5也停止.** |

### PWM时钟选择寄存器 (CSR03)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| CSR03 | PWM_BA0+0x004 | R/W | PWM时钟选择寄存器 | 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 | CSR3 | | | 保留 | CSR2 | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | CSR1 | | | 保留 | CSR0 | | |

| Bits | 描述 | |
|---|---|---|
| [31:15] | - | 保留 |
| [14:12] | CSR3 | **定时器3 时钟源选择**<br>为定时器3选择时钟输入.<br>CSR3 [14:12] / 输入时钟除频:<br>100 / 1<br>011 / 16<br>010 / 8<br>001 / 4<br>000 / 2 |
| [10:8] | CSR2 | **定时器2时钟源选择**<br>为定时器2选择时钟输入.<br>(表与CSR3相同) |
| [6:4] | CSR1 | **定时器1时钟源选择**<br>为定时器1选择时钟输入.<br>(表与CSR3相同) |

| | | |
|---|---|---|
| [2:0] | **CSR0** | **定时器0时钟源选择**<br>为定时器0选择时钟输入.<br>(表与CSR3相同) |

### PWM 时钟选择寄存器 (CSR47)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| CSR47 | PWM_BA4+0x004 | R/W | PWM时钟选择寄存器 | 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 | CSR7 | | | 保留 | CSR6 | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | CSR5 | | | 保留 | CSR4 | | |

| Bits | 描述 | |
|---|---|---|
| [31:15] | - | 保留 |
| [14:12] | CSR7 | **定时器7 时钟源选择**<br>为定时器7选择时钟输入.<br>CSR3 [14:12] / 输入时钟除频:<br>100 / 1<br>011 / 16<br>010 / 8<br>001 / 4<br>000 / 2 |
| [10:8] | CSR6 | **定时器6时钟源选择**<br>为定时器6选择时钟输入.<br>(表与CSR7相同) |
| [6:4] | CSR5 | **定时器5时钟源选择**<br>为定时器5选择时钟输入.<br>(表与CSR7相同) |

| | | |
|---|---|---|
| [2:0] | **CSR4** | **定时器4时钟源选择**<br>为定时器4选择时钟输入.<br>(表与CSR7相同) |

### PWM 控制寄存器(PCR03)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| PCR03 | PWM_BA0+0x008 | R/W | PWM 控制寄存器(PCR) | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| 保留 | | | | CH3MOD | CH3INV | 保留 | CH3EN |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 保留 | | | | CH2MOD | CH2INV | 保留 | CH2EN |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| 保留 | | | | CH1MOD | CH1INV | 保留 | CH1EN |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | DZEN23 | DZEN01 | CH0MOD | CH0INV | 保留 | CH0EN |

| Bits | 描述 | |
|---|---|---|
| [27] | **CH3MOD** | **PWM定时器3 自动重载/单触发模式选择**<br>1 = 自动重载模式<br>0 = 单触发模式<br>**注:** 如果该位由0置1，会使CNR3 和CMR3 清位. |
| [26] | **CH3INV** | **PWM定时器3反向打开/关闭**<br>1 = 反向打开<br>0 = 反向关闭 |
| [24] | **CH3EN** | **PWM定时器3 使能/禁止**<br>1 = 使能PWM功能<br>0 = 禁止PWM功能 |
| [19] | **CH2MOD** | **PWM定时器2 自动重载/单触发模式选择**<br>1 = 自动重载模式<br>0 = 单触发模式<br>**注:** 如果该位由0置1，会使CNR3 和CMR3 清位. |

| | | |
|---|---|---|
| [18] | **CH2INV** | **PWM定时器2反向打开/关闭**<br>1 = 反向打开<br>0 = 反向关闭 |
| [16] | **CH2EN** | **PWM定时器2 使能/禁止**<br>1 = 使能PWM功能<br>0 = 禁止PWM功能 |
| [11] | **CH1MOD** | **PWM定时器1 自动重载/单触发模式选择**<br>1 = 自动重载模式<br>0 = 单触发模式<br>**注:** 如果该位由0置1，会使CNR3 和CMR3 清位. |
| [10] | **CH1INV** | **PWM定时器1反向打开/关闭**<br>1 = 反向打开<br>0 = 反向关闭 |
| [8] | **CH1EN** | **PWM定时器1 使能/禁止**<br>1 = 使能PWM功能<br>0 = 禁止PWM功能 |
| [5] | **DZEN23** | **死区发生器2使能/禁止**<br>1 = 使能<br>0 = 禁止<br>**注**: 当死区发生器使能，PWM2和PWM3将成为相应一对输出. |
| [4] | **DZEN01** | **死区发生器0使能/禁止**<br>1 = 使能<br>0 = 禁止<br>**注**: 当死区发生器使能，PWM0和PWM1将成为相应一对输出. |
| [3] | **CH0MOD** | **PWM定时器 0 自动加载/单触发模式选择**<br>1 = 自动重载模式<br>0 = 单触发模式<br>**注:** 如果该位由0置1，会使CNR3 和CMR3 清位. |
| [2] | **CH0INV** | **PWM定时器0反向打开/关闭**<br>1 = 反向打开<br>0 = 反向关闭 |

| | | |
|---|---|---|
| [0] | **CH0EN** | **PWM定时器0 使能/禁止**<br>1 = 使能PWM功能<br>0 = 禁止PWM功能 |

### PWM 控制寄存器 (PCR47)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| PCR47 | PWM_BA4+0x008 | R/W | PWM 控制寄存器 (PCR) | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| 保留 | | | | CH7MOD | CH7INV | 保留 | CH7EN |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 保留 | | | | CH6MOD | CH6INV | 保留 | CH6EN |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| 保留 | | | | CH5MOD | CH5INV | 保留 | CH5EN |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | DZEN67 | DZEN45 | CH4MOD | CH4INV | 保留 | CH4EN |

| Bits | 描述 | |
|---|---|---|
| [27] | **CH7MOD** | **PWM定时器7 自动重载/单触发模式选择**<br>1 = 自动重载模式<br>0 = 单触发模式<br>**注:** 如果该位由0置1，会使CNR7 和CMR7 清位. |
| [26] | **CH7INV** | **PWM定时器7反向打开/关闭**<br>1 = 反向打开<br>0 = 反向关闭 |
| [24] | **CH7EN** | **PWM定时器7 使能/禁止**<br>1 = 使能PWM功能<br>0 = 禁止PWM功能 |
| [19] | **CH6MOD** | **PWM定时器6 自动重载/单触发模式选择**<br>1 = 自动重载模式<br>0 = 单触发模式<br>**注:** 如果该位由0置1，会使CNR6 和CMR6 清位. |

| | | |
|---|---|---|
| [18] | **CH6INV** | **PWM定时器6反向打开/关闭**<br>1 = 反向打开<br>0 = 反向关闭 |
| [16] | **CH6EN** | **PWM定时器6 使能/禁止**<br>1 = 使能PWM功能<br>0 = 禁止PWM功能 |
| [11] | **CH5MOD** | **PWM定时器5 自动重载/单触发模式选择**<br>1 = 自动重载模式<br>0 = 单触发模式<br>**注:** 如果该位由0置1，会使CNR5 和CMR5 清位. |
| [10] | **CH5INV** | **PWM定时器5反向打开/关闭**<br>1 = 反向打开<br>0 = 反向关闭 |
| [8] | **CH5EN** | **PWM定时器5使能/禁止**<br>1 = 使能PWM功能<br>0 = 禁止PWM功能 |
| [5] | **DZEN67** | **死区发生器2使能/禁止**<br>1 = 使能<br>0 = 禁止<br>**注**: 当死区发生器使能，PWM6和PWM7将成为相应一对输出. |
| [4] | **DZEN45** | **死区发生器0使能/禁止**<br>1 = 使能<br>0 = 禁止<br>**注**: 当死区发生器使能，PWM4和PWM5将成为相应一对输出 |
| [3] | **CH4MOD** | **PWM定时器4 自动重载/单触发模式选择**<br>1 = 自动重载模式<br>0 = 单触发模式<br>**注:** 如果该位由0置1，会使CNR4 和CMR4 清位. |
| [2] | **CH4INV** | **PWM定时器4反向打开/关闭**<br>1 = 反向打开<br>0 = 反向关闭 |

| | | |
|---|---|---|
| [0] | **CH4EN** | **PWM定时器4使能/禁止**<br>1 = 使能PWM功能<br>0 = 禁止PWM功能 |

### PWM 计数器寄存器7-0 (CNR7-0)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **CNR0** | PWM_BA0+0x00C | R/W | PWM 计数器寄存器 0 | 0x0000_0000 |
| **CNR1** | PWM_BA0+0x018 | R/W | PWM 计数器寄存器 1 | 0x0000_0000 |
| **CNR2** | PWM_BA0+0x024 | R/W | PWM 计数器寄存器 2 | 0x0000_0000 |
| **CNR3** | PWM_BA0+0x030 | R/W | PWM 计数器寄存器 3 | 0x0000_0000 |
| **CNR4** | PWM_BA4+0x00C | R/W | PWM 计数器寄存器 4 | 0x0000_0000 |
| **CNR5** | PWM_BA4+0x018 | R/W | PWM 计数器寄存器 5 | 0x0000_0000 |
| **CNR6** | PWM_BA4+0x024 | R/W | PWM 计数器寄存器 6 | 0x0000_0000 |
| **CNR7** | PWM_BA4+0x030 | R/W | PWM 计数器寄存器 7 | 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** |
| **CNR [15:8]** | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **CNR [7:0]** | | | | | | | |

| Bits | 描述 |
|---|---|

| | | |
|---|---|---|
| [15:0] | **CNR** | **PWM 计数器/定时器载入值**<br>CNR决定PWM的周期.<br>• PWM 频率 = PWMxy_CLK/(prescale+1)*(clock divider)/(CNR+1); xy代表01, 23, 45 or 67, 取决于所选择的PWM通道.<br>• 占空比 = (CMR+1)/(CNR+1).<br>• CMR >= CNR: PWM 输出高.<br>• CMR < CNR: PWM 低脉宽 = (CNR-CMR) unit; PWM高脉宽= (CMR+1) unit.<br>• CMR = 0: PWM低脉宽= (CNR) unit; PWM高脉宽= 1 unit<br>(Unit = one PWM clock cycle)<br>**注:**<br>CNR写入数据后将在下一个PWM周期生效. |

### PWM 比较寄存器7-0 (CMR7-0)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **CMR0** | PWM_BA0+0x010 | R/W | PWM 比较寄存器0 | 0x0000_0000 |
| **CMR1** | PWM_BA0+0x01C | R/W | PWM 比较寄存器1 | 0x0000_0000 |
| **CMR2** | PWM_BA0+0x028 | R/W | PWM 比较寄存器2 | 0x0000_0000 |
| **CMR3** | PWM_BA0+0x034 | R/W | PWM 比较寄存器3 | 0x0000_0000 |
| **CMR4** | PWM_BA4+0x010 | R/W | PWM 比较寄存器4 | 0x0000_0000 |
| **CMR5** | PWM_BA4+0x01C | R/W | PWM 比较寄存器5 | 0x0000_0000 |
| **CMR6** | PWM_BA4+0x028 | R/W | PWM 比较寄存器6 | 0x0000_0000 |
| **CMR7** | PWM_BA4+0x034 | R/W | PWM比较寄存器7 | 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** |
| **CMR[15:8]** | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **CMR[7:0]** | | | | | | | |

| Bits | 描述 |
|---|---|

| | | |
|---|---|---|
| [15:0] | **CMR** | **PWM比较寄存器**<br>CMR 决定 PWM 的占空比.<br>• PWM 频率 = PWMxy_CLK/(prescale+1)*(clock divider)/(CNR+1); xy代表 01, 23, 45 or 67, 取决于所选择的PWM通道.<br>• 占空比 = (CMR+1)/(CNR+1).<br>• CMR >= CNR: PWM输出高.<br>• CMR < CNR: PWM 低脉宽 = (CNR-CMR) unit; PWM 高脉宽= (CMR+1) unit.<br>• CMR = 0: PWM低脉宽= (CNR) unit; PWM高脉宽= 1 unit<br>(Unit =1个PWM时钟周期)<br>**注:**<br>CNR写入数据后将在下一个PWM周期生效. |

### PWM 数据寄存器 7-0 (PDR 7-0)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **PDR0** | PWM_BA0+0x014 | R | PWM 数据寄存器 0 | 0x0000_0000 |
| **PDR1** | PWM_BA0+0x020 | R | PWM 数据寄存器 1 | 0x0000_0000 |
| **PDR2** | PWM_BA0+0x02C | R | PWM 数据寄存器 2 | 0x0000_0000 |
| **PDR3** | PWM_BA0+0x038 | R | PWM 数据寄存器 3 | 0x0000_0000 |
| **PDR4** | PWM_BA4+0x014 | R | PWM 数据寄存器 4 | 0x0000_0000 |
| **PDR5** | PWM_BA4+0x020 | R | PWM 数据寄存器 5 | 0x0000_0000 |
| **PDR6** | PWM_BA4+0x02C | R | PWM 数据寄存器 6 | 0x0000_0000 |
| **PDR7** | PWM_BA4+0x038 | R | PWM 数据寄存器 7 | 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** |
| **PDR[15:8]** | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **PDR[7:0]** | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [15:0] | **PDR** | **PWM数据寄存器**<br>用来查询16位计数器当前值. |

### PWM 中断使能寄存器(PIER03)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| PIER03 | PWM_BA0+0x040 | R/W | PWM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 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | | | PIER3 | PIER2 | PIER1 | PIER0 |

| Bits | 描述 | |
|---|---|---|
| [3] | PWMIE3 | **PWM 定时器3 中断使能**<br>1 = 使能<br>0 = 禁止 |
| [2] | PWMIE2 | **PWM 定时器2 中断使能**<br>1 = 使能<br>0 = 禁止 |
| [1] | PWMIE1 | **PWM 定时器1 中断使能**<br>1 = 使能<br>0 = 禁止 |
| [0] | PWMIE0 | **PWM 定时器0 中断使能**<br>1 = 使能<br>0 = 禁止 |

### PWM 中断使能寄存器 (PIER47)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| PIER47 | PWM_BA4+0x040 | R/W | PWM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 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | | | PIER7 | PIER6 | PIER5 | PIER4 |

| Bits | 描述 | |
|---|---|---|
| [3] | PWMIE7 | **PWM 定时器7 中断使能**<br>1 = 使能<br>0 = 禁止 |
| [2] | PWMIE6 | **PWM 定时器6 中断使能**<br>1 = 使能<br>0 = 禁止 |
| [1] | PWMIE5 | **PWM 定时器5中断使能**<br>1 = 使能<br>0 = 禁止 |
| [0] | PWMIE4 | **PWM 定时器4 中断使能**<br>1 = 使能<br>0 = 禁止 |

### PWM中断标志寄存器(PIFR03)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| PIIR03 | PWM_BA0+0x044 | R/W | PWM中断标志寄存器 | 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 | | | | PWMIF3 | PWMIF2 | PWMIF1 | PWMIF0 |

| Bits | 描述 | |
|---|---|---|
| [3] | PWMIF3 | **PWM 定时器 3 中断标志**<br>当PWM3 向下计数至0，硬件自动将该位置1。软件清除该位。 |
| [2] | PWMIF2 | **PWM 定时器 2 中断标志**<br>当PWM2 向下计数至0，硬件自动将该位置1。软件清除该位。 |
| [1] | PWMIF1 | **PWM 定时器 1 中断标志**<br>当PWM1 向下计数至0，硬件自动将该位置1。软件清除该位。 |
| [0] | PWMIF0 | **PWM 定时器 0 中断标志**<br>当PWM0 向下计数至0，硬件自动将该位置1。软件清除该位。 |

**注:** 用户可通过改写PIIR相应的位来对中断标志清零。

### PWM 中断标志寄存器 (PIFR47)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| PIIR47 | PWM_BA4+0x044 | R/W | PWM中断标志寄存器 | 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 | | | | PWMIF7 | PWMIF6 | PWMIF5 | PWMIF4 |

| Bits | 描述 | |
|---|---|---|
| [3] | **PWMIF7** | **PWM 定时器 7中断标志**<br>当PWM7 向下计数至0，硬件自动将该位置1。软件清除该位。 |
| [2] | **PWMIF6** | **PWM 定时器 6中断标志**<br>当PWM6 向下计数至0，硬件自动将该位置1。软件清除该位。 |
| [1] | **PWMIF5** | **PWM 定时器5 中断标志**<br>当PWM5 向下计数至0，硬件自动将该位置1。软件清除该位。 |
| [0] | **PWMIF4** | **PWM 定时器4 中断标志**<br>当PWM4 向下计数至0，硬件自动将该位置1。软件清除该位。 |

**注:** 用户可通过改写PIIR相应的位来对中断标志清零。

### 捕捉控制寄存器 (CCR0)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| CCR0 | PWM_BA0+0x050 | R/W | 捕捉控制寄存器 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| CFLRI1 | CRLRI1 | 保留 | CAPIF1 | CAPCH1EN | FL_IE1 | RL_IE1 | INV1 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| 保留 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CFLRI0 | CRLRI0 | 保留 | CAPIF0 | CAPCH0EN | FL_IE0 | RL_IE0 | INV0 |

| Bits | 描述 | |
|---|---|---|
| [23] | **CFLRI1** | **CFLR1锁定方向标志位**<br>当输入通道1为向下计数传输，CFLR1 硬件置位，软件清除该位. |
| [22] | **CRLRI1** | **CRLR1锁定方向标志位**<br>当输入通道1为向上计数传输，CRLR1 硬件置位，软件清除该位. |
| [20] | **CAPIF1** | **捕捉器1中断标志**<br>当通道1内为向下传输，且向下传输中断打开，CAPIF1硬件置位，软件清除该位。<br>**注:** 如果该位为1，PWM计数器1 (CNR1)将不在重载下一个捕捉器中断，软件写0清除该位 |
| [19] | **CAPCH1EN** | **捕捉器通道1传输使能/禁止**<br>1 = 使能通道1捕捉器功能<br>0 = 禁止通道1捕捉器功能<br>当使能，捕捉锁定PWM计数器，并存储到CRLR (上升方式) 或CFLR (下降方式).<br>当禁止，捕捉器不更新CRLR 或CFLR, 并关闭通道1中断 |
| [18] | **CFL_IE1** | **通道1 向下计数中断使能**<br>1 = 使能向下计数中断<br>0 = 禁止向下计数中断<br>当使能，如果捕捉模式侦测到通道1是向下计数传输，触发中断. |

| | | |
|---|---|---|
| [17] | **CRL_IE1** | **通道1 向上计数中断使能**<br>1 = 使能向上计数中断<br>0 = 禁止向上计数中断<br>当使能，如果捕捉模式侦测到通道1是向上计数传输，触发中断 |
| [16] | **INV1** | **通道1反向打开/关闭**<br>1 = 反向打开。输入到寄存器的信号与通道上的实际信号点平反向。<br>0 = 反向关闭 |
| [7] | **CFLRI0** | **CFLR0锁定方向标志位**<br>当输入通道0为向下计数传输，CFLR0 硬件置位，软件清除该位。 |
| [6] | **CRLRI0** | **CRLR0锁定方向标志位**<br>当输入通道0为向上计数传输，CRLR0 硬件置位，软件清除该位。 |
| [4] | **CAPIF0** | **捕捉器0中断标志**<br>当通道0内为向下传输，且向下传输中断打开，CAPIF0硬件置位，软件清除该位。<br>**注:** 如果该位为0，PWM计数器0 (CNR0)将不在重载下一个捕捉器中断，软件写0清除该位 。 |
| [3] | **CAPCH0EN** | **捕捉器通道0传输使能/禁止**<br>1 = 使能通道0捕捉器功能<br>0 = 禁止通道0捕捉器功能<br>当使能，捕捉锁定PWM计数器，并存储到CRLR (上升方式) 或CFLR (下降方式).<br>当禁止，捕捉器不更新CRLR 或CFLR, 并关闭通道0中断。 |
| [2] | **CFL_IE0** | **通道0 向下计数中断使能**<br>1= 使能向下计数中断<br>0 = 禁止向下计数中断<br>当使能，如果捕捉模式侦测到通道0是向下计数传输，触发中断 |
| [1] | **CRL_IE0** | **通道0 向上计数中断使能**<br>1 = 使能向上计数中断<br>0 = 禁止向上计数中断<br>当使能，如果捕捉模式侦测到通道0是向上计数传输，触发中断。 |
| [0] | **INV0** | **通道0反向打开/关闭**<br>1 = 反向打开。输入到寄存器的信号与通道上的实际信号点平反向。<br>0 = 反向关闭 |

### 捕捉控制寄存器 (CCR2)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| CCR1 | PWM_BA0+0x054 | R/W | 捕捉控制寄存器 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| CFLRI3 | CRLRI3 | 保留 | CAPIF3 | CAPCH3EN | FL_IE3 | RL_IE3 | INV3 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| 保留 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CFLRI2 | CRLRI2 | 保留 | CAPIF2 | CAPCH2EN | FL_IE2 | RL_IE2 | INV2 |

| Bits | 描述 | |
|---|---|---|
| [23] | **CFLRI3** | **CFLR3锁定方向标志位**<br>当输入通道3为向下计数传输，CFLR3 硬件置位，软件清除该位。 |
| [22] | **CRLRI3** | **CRLR3锁定方向标志位**<br>当输入通道3为向上计数传输，CRLR3 硬件置位，软件清除该位。 |
| [20] | **CAPIF3** | **捕捉器3中断标志**<br>当通道3内为向下传输，且向下传输中断打开，CAPIF3硬件置位，软件清除该位。<br>**注:** 如果该位为0，PWM计数器3 (CNR3)将不在重载下一个捕捉器中断，软件写0清除该位 。 |
| [19] | **CAPCH3EN** | **捕捉器通道3传输使能/禁止**<br>1 = 使能通道3捕捉器功能<br>0 = 禁止通道3捕捉器功能<br>当使能，捕捉锁定PWM计数器，并存储到CRLR (上升方式) 或CFLR (下降方式).<br>当禁止，捕捉器不更新CRLR 或CFLR, 并关闭通道3中断。 |
| [18] | **CFL_IE3** | **通道3 向下计数中断使能**<br>1= 使能向下计数中断<br>0 = 禁止向下计数中断<br>当使能，如果捕捉模式侦测到通道3是向下计数传输，触发中断。 |

| | | |
|---|---|---|
| [17] | **CRL_IE3** | **通道3 向上计数中断使能**<br>1 = 使能向上计数中断<br>0 = 禁止向上计数中断<br>当使能，如果捕捉模式侦测到通道3是向上计数传输，触发中断。 |
| [16] | **INV3** | **通道3反向打开/关闭**<br>1 = 反向打开。输入到寄存器的信号与通道上的实际信号点平反向。<br>0 = 反向关闭 |
| [7] | **CFLRI2** | **CFLR2锁定方向标志位**<br>当输入通道2为向下计数传输，CFLR2 硬件置位，软件清除该位。 |
| [6] | **CRLRI2** | **CRLR2锁定方向标志位**<br>当输入通道2为向上计数传输，CRLR2 硬件置位，软件清除该位。 |
| [4] | **CAPIF2** | **捕捉器2中断标志**<br>当通道2内为向下传输，且向下传输中断打开，CAPIF2硬件置位，软件清除该位。<br>**注:** 如果该位为0，PWM计数器2 (CNR2)将不在重载下一个捕捉器中断，软件写0清除该位 。 |
| [3] | **CAPCH2EN** | **捕捉器通道2传输使能/禁止**<br>1 = 使能通道2捕捉器功能<br>0 = 禁止通道2捕捉器功能<br>当使能，捕捉锁定PWM计数器，并存储到CRLR (上升方式) 或CFLR (下降方式).<br>当禁止，捕捉器不更新CRLR 或CFLR, 并关闭通道2中断。 |
| [2] | **CFL_IE2** | **通道2 向下计数中断使能**<br>1= 使能向下计数中断<br>0 = 禁止向下计数中断<br>当使能，如果捕捉模式侦测到通道2是向下计数传输，触发中断。 |
| [1] | **CRL_IE2** | **通道2 向上计数中断使能**<br>1 = 使能向上计数中断<br>0 = 禁止向上计数中断<br>当使能，如果捕捉模式侦测到通道2是向上计数传输，触发中断。 |
| [0] | **INV2** | **通道2反向打开/关闭**<br>1 = 反向打开。输入到寄存器的信号与通道上的实际信号点平反向。<br>0 = 反向关闭 |

### 捕捉控制寄存器(CCR4)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| CCR4 | PWM_BA4+0x050 | R/W | 捕捉控制寄存器 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| CFLRI5 | CRLRI5 | 保留 | CAPIF5 | CAPCH5EN | FL_IE5 | RL_IE5 | INV5 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| 保留 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CFLRI4 | CRLRI4 | 保留 | CAPIF4 | CAPCH4EN | FL_IE4 | RL_IE4 | INV4 |

| Bits | 描述 | |
|---|---|---|
| [23] | **CFLRI5** | **CFLR5锁定方向标志位**<br>当输入通道5为向下计数传输，CFLR5 硬件置位，软件清除该位。 |
| [22] | **CRLRI5** | **CRLR5锁定方向标志位**<br>当输入通道5为向上计数传输，CRLR5 硬件置位，软件清除该位。 |
| [20] | **CAPIF5** | **捕捉器5中断标志**<br>当通道5内为向下传输，且向下传输中断打开，CAPIF5硬件置位，软件清除该位。<br>**注:** 如果该位为0，PWM计数器5 (CNR5)将不在重载下一个捕捉器中断，软件写0清除该位 。 |
| [19] | **CAPCH5EN** | **捕捉器通道2传输使能/禁止**<br>1 = 使能通道2捕捉器功能<br>0 = 禁止通道2捕捉器功能<br>当使能，捕捉锁定PWM计数器，并存储到CRLR (上升方式) 或CFLR (下降方式).<br>当禁止，捕捉器不更新CRLR 或CFLR, 并关闭通道5中断。 |
| [18] | **CFL_IE5** | **通道5 向下计数中断使能**<br>1= 使能向下计数中断<br>0 = 禁止向下计数中断<br>当使能，如果捕捉模式侦测到通道5是向下计数传输，触发中断。 |

| | | |
|---|---|---|
| [17] | **CRL_IE5** | **通道5 向上计数中断使能**<br>1 = 使能向上计数中断<br>0 = 禁止向上计数中断<br>当使能，如果捕捉模式侦测到通道5是向上计数传输，触发中断。 |
| [16] | **INV5** | **通道5反向打开/关闭**<br>1 = 反向打开。输入到寄存器的信号与通道上的实际信号点平反向。<br>0 = 反向关闭 |
| [7] | **CFLRI4** | **CFLR4锁定方向标志位**<br>当输入通道4为向下计数传输，CFLR4 硬件置位，软件清除该位。 |
| [6] | **CRLRI4** | **CRLR4锁定方向标志位**<br>当输入通道4为向上计数传输，CRLR4 硬件置位，软件清除该位。 |
| [4] | **CAPIF4** | **捕捉器4中断标志**<br>当通道4内为向下传输，且向下传输中断打开，CAPIF4硬件置位，软件清除该位。<br>**注:** 如果该位为0，PWM计数器4 (CNR4)将不在重载下一个捕捉器中断，软件写0清除该位 。 |
| [3] | **CAPCH4EN** | **捕捉器通道4传输使能/禁止**<br>1 = 使能通道4捕捉器功能<br>0 = 禁止通道4捕捉器功能<br>当使能，捕捉锁定PWM计数器，并存储到CRLR (上升方式) 或CFLR (下降方式).<br>当禁止，捕捉器不更新CRLR 或CFLR, 并关闭通道4中断。 |
| [2] | **CFL_IE4** | **通道4 向下计数中断使能**<br>1= 使能向下计数中断<br>0 = 禁止向下计数中断<br>当使能，如果捕捉模式侦测到通道4是向下计数传输，触发中断。 |
| [1] | **CRL_IE4** | **通道4向上计数中断使能**<br>1 = 使能向上计数中断<br>0 = 禁止向上计数中断<br>当使能，如果捕捉模式侦测到通道4是向上计数传输，触发中断。 |
| [0] | **INV4** | **通道4反向打开/关闭**<br>1 = 反向打开。输入到寄存器的信号与通道上的实际信号点平反向。<br>0 = 反向关闭 |

### 捕捉控制寄存器(CCR6)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| CCR6 | PWM_BA4+0x054 | R/W | 捕捉控制寄存器 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| CFLRI7 | CRLRI7 | 保留 | CAPIF7 | CAPCH7EN | FL_IE7 | RL_IE7 | INV7 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| 保留 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CFLRI6 | CRLRI6 | 保留 | CAPIF6 | CAPCH6EN | FL_IE6 | RL_IE6 | INV6 |

| Bits | 描述 | |
|---|---|---|
| [23] | **CFLRI7** | **CFLR7锁定方向标志位**<br>当输入通道7为向下计数传输，CFLR7 硬件置位，软件清除该位。 |
| [22] | **CRLRI7** | **CRLR7锁定方向标志位**<br>当输入通道7为向上计数传输，CRLR7 硬件置位，软件清除该位。 |
| [20] | **CAPIF7** | **捕捉器7中断标志**<br>当通道7内为向下传输，且向下传输中断打开，CAPIF7硬件置位，软件清除该位。<br>**注:** 如果该位为0，PWM计数器7 (CNR7)将不在重载下一个捕捉器中断，软件写0清除该位 。 |
| [19] | **CAPCH7EN** | **捕捉器通道7传输使能/禁止**<br>1 = 使能通道7捕捉器功能<br>0 = 禁止通道7捕捉器功能<br>当使能，捕捉锁定PWM计数器，并存储到CRLR (上升方式) 或CFLR (下降方式).<br>当禁止，捕捉器不更新CRLR 或CFLR, 并关闭通道7中断。 |
| [18] | **CFL_IE7** | **通道7 向下计数中断使能**<br>1= 使能向下计数中断<br>0 = 禁止向下计数中断<br>当使能，如果捕捉模式侦测到通道7是向下计数传输，触发中断。 |

| | | |
|---|---|---|
| [17] | **CRL_IE7** | **通道7 向上计数中断使能**<br>1 = 使能向上计数中断<br>0 = 禁止向上计数中断<br>当使能，如果捕捉模式侦测到通道7是向上计数传输，触发中断。 |
| [16] | **INV7** | **通道7反向打开/关闭**<br>1 = 反向打开。输入到寄存器的信号与通道上的实际信号点平反向。<br>0 = 反向关闭 |
| [7] | **CFLRI6** | **CFLR6锁定方向标志位**<br>当输入通道6为向下计数传输，CFLR6 硬件置位，软件清除该位。 |
| [6] | **CRLRI6** | **CRLR6锁定方向标志位**<br>当输入通道6为向上计数传输，CRLR6 硬件置位，软件清除该位。 |
| [4] | **CAPIF6** | **捕捉器6中断标志**<br>当通道6内为向下传输，且向下传输中断打开，CAPIF6硬件置位，软件清除该位。<br>**注:** 如果该位为0，PWM计数器6 (CNR6)将不在重载下一个捕捉器中断，软件写0清除该位 。 |
| [3] | **CAPCH6EN** | **捕捉器通道6传输使能/禁止**<br>1 = 使能通道6捕捉器功能<br>0 = 禁止通道6捕捉器功能<br>当使能，捕捉锁定PWM计数器，并存储到CRLR (上升方式) 或CFLR (下降方式).<br>当禁止，捕捉器不更新CRLR 或CFLR, 并关闭通道6中断。 |
| [2] | **CFL_IE6** | **通道6 向下计数中断使能**<br>1= 使能向下计数中断<br>0 = 禁止向下计数中断<br>当使能，如果捕捉模式侦测到通道6是向下计数传输，触发中断。 |
| [1] | **CRL_IE6** | **通道6 向上计数中断使能**<br>1 = 使能向上计数中断<br>0 = 禁止向上计数中断<br>当使能，如果捕捉模式侦测到通道6是向上计数传输，触发中断。 |
| [0] | **INV6** | **通道6反向打开/关闭**<br>1 = 反向打开。输入到寄存器的信号与通道上的实际信号点平反向。<br>0 = 反向关闭 |

### 捕捉上升沿锁存寄存器3-0 (CRLR3-0)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **CRLR0** | PWM_BA0+0x058 | R/W | 捕捉上升沿锁存寄存器(通道 0) | 0x0000_0000 |
| **CRLR1** | PWM_BA0+0x060 | R/W | 捕捉上升沿锁存寄存器(通道 1) | 0x0000_0000 |
| **CRLR2** | PWM_BA0+0x068 | R/W | 捕捉上升沿锁存寄存器(通道 2) | 0x0000_0000 |
| **CRLR3** | PWM_BA0+0x070 | R/W | 捕捉上升沿锁存寄存器(通道 3) | 0x0000_0000 |

注: 当CPU时钟低于PWM/Capture时钟时，不能对CRLRx进行改写。

| 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** |
| CRLR[15:8] | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| CRLR[7:0] | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [15:0] | **CRLR** | 捕捉上升沿锁存寄存器<br>通道0/1/2/3 向上传输时，锁存PWM计数器. |

### 捕捉上升沿锁存寄存器4-7 (CRLR4-7)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **CRLR4** | PWM_BA4+0x058 | R/W | 捕捉上升沿锁存寄存器（通道4) | 0x0000_0000 |
| **CRLR5** | PWM_BA4+0x060 | R/W | 捕捉上升沿锁存寄存器（通道5) | 0x0000_0000 |
| **CRLR6** | PWM_BA4+0x068 | R/W | 捕捉上升沿锁存寄存器（通道6) | 0x0000_0000 |
| **CRLR7** | PWM_BA4+0x070 | R/W | 捕捉上升沿锁存寄存器（通道7) | 0x0000_0000 |

注: 当CPU时钟低于PWM/Capture时钟时，不能对CRLRx进行改写

| 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** |
| **CRLR[15:8]** | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **CRLR[7:0]** | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [15:0] | **CRLR** | 捕捉上升沿锁存寄存器<br>通道4/5/6/7 向上传输时，锁存PWM计数器 |

### 捕捉下降沿锁存寄存器3-0 (CFLR3-0)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **CFLR0** | PWM_BA0+0x05C | R/W | 捕捉下降沿锁存寄存器（通道0) | 0x0000_0000 |
| **CFLR1** | PWM_BA0+0x064 | R/W | 捕捉下降沿锁存寄存器（通道1) | 0x0000_0000 |
| **CFLR2** | PWM_BA0+0x06C | R/W | 捕捉下降沿锁存寄存器（通道2) | 0x0000_0000 |
| **CFLR3** | PWM_BA0+0x074 | R/W | 捕捉下降沿锁存寄存器（通道3) | 0x0000_0000 |

注: 当CPU时钟低于PWM/Capture时钟时，不能对 CFLRx 进行改写.

| 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** |
| **CFLR[15:8]** | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **CFLR[7:0]** | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [15:0] | **CFLR** | 捕捉下降沿锁存寄存器<br>通道01/2/3 向下传输时，锁存PWM计数器 |

### 捕捉下降沿锁存寄存器4-7 (CFLR4-7)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **CFLR4** | PWM_BA4+0x05C | R/W | 捕捉下降沿锁存寄存器（通道4) | 0x0000_0000 |
| **CFLR5** | PWM_BA4+0x064 | R/W | 捕捉下降沿锁存寄存器（通道5) | 0x0000_0000 |
| **CFLR6** | PWM_BA4+0x06C | R/W | 捕捉下降沿锁存寄存器（通道6) | 0x0000_0000 |
| **CFLR7** | PWM_BA4+0x074 | R/W | 捕捉下降沿锁存寄存器（通道7) | 0x0000_0000 |

注: 当CPU时钟低于PWM/Capture时钟时，不能对CFLRx 进行改写.

| 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** |
| **CFLR[15:8]** | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **CFLR[7:0]** | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [15:0] | **CFLR** | 捕捉下降沿锁存寄存器<br>通道4/5/6/7 向下传输时，锁存PWM计数器 |

捕捉输入使能寄存器(CAPENR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| CAPENR | PWM_BA0+0x078 | R/W | 捕捉输入使能寄存器 | 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 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | | | CAPENR[3:0] | | | |

| Bits | 描述 | |
|---|---|---|
| [3:0] | CAPENR | 捕捉输入使能寄存器<br>4组捕捉输入。Bit0~Bit3 用于控制每个输入的打开 / 关闭。(4通道可同时使用)<br>0 = 关闭(GPA[15:12] 脚输入对捕捉器不产生影响)<br>1 = 打开(GPA[15:12] 脚输入捕捉使能)<br>CAPENR[3:0]<br>Bit 3210<br>Bit xxx1 ➔ 捕捉通道0 从 GPA [12] 输入<br>Bit xx1x ➔ 捕捉通道1 从 GPA [13] 输入<br>Bit x1xx ➔ 捕捉通道2 从 GPA [14] 输入<br>Bit 1xxx ➔ 捕捉通道3 从 GPA [15] 输入 |

### PWM输出使能寄存器(POE0)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| POE | PWM_BA0+0x07C | R/W | PWM输出使能寄存器 | 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 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | | | PWM3 | PWM2 | PWM1 | PWM0 |

| Bits | 描述 | |
|---|---|---|
| [3] | **PWM3** | PWM3 输出使能寄存器<br>1 = 使能PWM3 输出<br>0 = 禁止PWM3 输出<br>注:相应的GPIO管脚特性要根绝相应的PWM要求配置(参考 GPA_MFP) |
| [2] | **PWM2** | PWM2 输出使能寄存器<br>1 = 使能PWM2 输出<br>0 = 禁止PWM2 输出<br>注:相应的GPIO管脚特性要根绝相应的PWM要求配置(参考 GPA_MFP) |
| [1] | **PWM1** | PWM1 输出使能寄存器<br>1 = 使能PWM1 输出<br>0 = 禁止PWM1 输出<br>注:相应的GPIO管脚特性要根绝相应的PWM要求配置(参考 GPA_MFP) |
| [0] | **PWM0** | PWM0 输出使能寄存器<br>1 = 使能PWM0 输出<br>0 = 禁止PWM0 输出<br>注:相应的GPIO管脚特性要根绝相应的PWM要求配置(参考 GPA_MFP) |

### PWM输出使能寄存器(POE4)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| POE | PWM_BA4+0x07C | R/W | PWM输出使能寄存器 | 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 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | | | PWM7 | PWM6 | PWM5 | PWM4 |

| Bits | 描述 | |
|---|---|---|
| [3] | **PWM7** | PWM7 输出使能寄存器<br>1 = 使能PWM7 输出<br>0 = 禁止PWM7 输出<br>注:相应的GPIO管脚特性要根绝相应的PWM要求配置(参考 GPA_MFP) |
| [2] | **PWM6** | PWM6 输出使能寄存器<br>1 = 使能PWM6 输出<br>0 = 禁止PWM6 输出<br>注:相应的GPIO管脚特性要根绝相应的PWM要求配置(参考 GPA_MFP) |
| [1] | **PWM5** | PWM5 输出使能寄存器<br>1 = 使能PWM5 输出<br>0 = 禁止PWM5 输出<br>注:相应的GPIO管脚特性要根绝相应的PWM要求配置(参考 GPA_MFP) |
| [0] | **PWM4** | PWM4 输出使能寄存器<br>1 = 使能PWM4 输出<br>0 = 禁止PWM4 输出<br>注:相应的GPIO管脚特性要根绝相应的PWM要求配置(参考 GPA_MFP) |

## 5.8 实时时钟(RTC)

### 5.8.1 概述

实时时钟 (RTC)模块用于记录实时时间及日历功能。时钟源由外部32.768KHz晶振提供，管脚为X32I 及X32O 或者外接32.768KHz 频率的信号源。RTC支持时间格式 (秒，分，时)寄存器(TLR)以及日历格式(日，月，年)寄存器(CLR)。数据格式由BCD格式存取。模块支持闹钟功能，(TAR)寄存器用于指示时间闹铃，(CAR)用于指示日期闹铃。

RTC 模块支持时间记步及闹钟中断，通过设定TTR.TTR[2:0]，中断提供8级选择1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 及1 秒。当 RTC 计数器内的值TLR 和CLR 与TAR和CAR相当时，会引发中断标志(RIIR.AIF) ，当中断使能(RIER.AIER=1)，MCU进入相应中断运行。如果唤醒CPU功能使能(TWKE(TTR[3])=1)时，RTC与闹钟匹配时将CPU从休眠或掉电模式唤醒

### 5.8.2 RTC 特征

- 支持时间格式 (秒，分，时)以及日历格式(日，月，年)用于确认实时。
- 闹钟寄存器(秒，分，时，日，月，年)
- 12-小时或 24-小时模式可选择.
- 自动根据年计数格式分割
- 闰年自动识别
- 频率补偿寄存器 (FCR).
- 所有时间日期由BCD码组成
- 提供8级周期选择1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 及1 秒。
- 支持闹钟中断
- 支持CPU唤醒功能.

### 5.8.3 RTC模块框图

RTC模块框图如下:

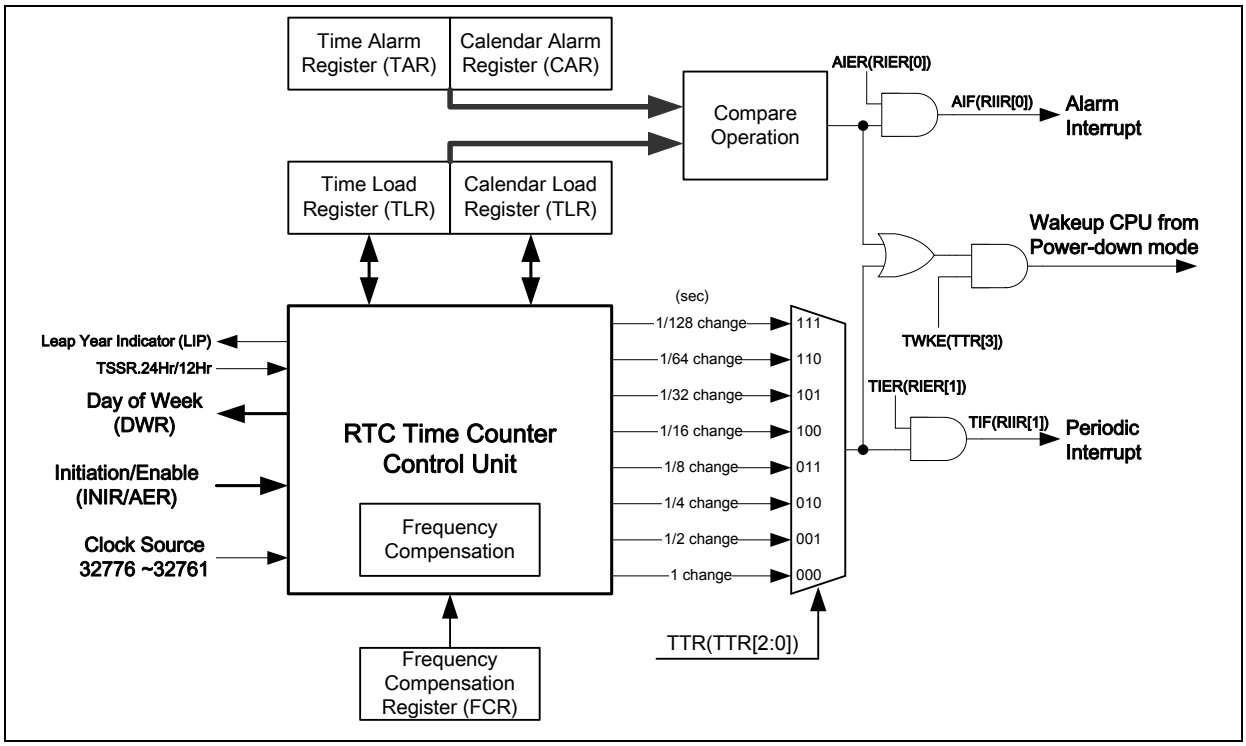



图 5-43 RTC 框图

### 5.8.4 RTC功能描述

#### *5.8.4.1 读写RTC寄存器*

基于RTC时钟与系统时钟采用不同的模块，用户对RTC寄存器写入后，必须等待2个RTC时钟周期(60us)后，寄存器内的值才会被更新。因此必须考虑TSSR, TAR 和TLR的写入步骤。

另外，用户必须确认RTC模块是否被禁止载入数据，RTC不会对DWR和CLR内的合理性进行确认.

#### *5.8.4.2 RTC初始化*

RTC模块上电后，可写入数据(0xa5eb1357) 至INIR用于重置。INIR根据硬件线路复位。一旦NIR 设定为0xa5eb1357，任何数据写入INIR都不能使RTC工作

#### *5.8.4.3 RTC读/写使能*

寄存器AER 位 15~0 作为RTC允许读/写密码，用于避免掉电时对RTC造成的误写。上电时AER 位15~0 被设定为0xa965 。一旦被设定，将会在512 RTC时钟后(大约15ms)产生作用，用户可通过读取AER.ENF 标志位来确认RTC是否正常工作

#### *5.8.4.4 频率补偿*

RTC FCR 允许软件对始终输入进行数字补偿，时钟输入频率必须在32776Hz 至32761Hz. 用户可通过GPIO测量RTC时钟的频率，并存储在内存中，当再次上电时用于数据检索。下列为频率输入的举例.

例1:

频率计数测量: 32773.65Hz ( > 32768 Hz)

整数部分: 32773 => 0x8005

FCR.Integer = 0x05 – 0x01 + 0x08 = 0x0c

分数部分: 0.65 x 60 = 39 => 0x27

FCR.Fraction = 0x27

例2

频率计数测量: 32765.27Hz ( ≦ 32768 Hz)

整数部分: 32765 => 0x7ffd

FCR.Integer = 0x0d – 0x01 – 0x08 = 0x04

分数部分: 0.27 x 60 = 16.2=> 0x10

FCR.Fraction = 0x10

#### *5.8.4.5 时间和日历计数*

TLR 和 CLR 用于载入时间和日历。TAR 和CAR 用于闹钟。由BCD格式呈现

#### *5.8.4.6 2/24 小时时间切换选择*

根据TSSR位0 选择12/24小时格式.

#### *5.8.4.7 一周日期格式*

RTC提供一周计日格式寄存器DWR. 内部的值由0至6 用于表示周日至周六

#### 5.8.4.8 时间周期中断

通过选择TTR.TTR[2:0]，中断具有8级选项1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 以及1秒。当周期中断寄存器RIER.AIER 置 1，MCU根据TTR寄存器内设定的值发生中断.

#### 5.8.4.9 时间闹钟中断

当 RTC 计数器TLR 和CLR等于闹钟设定时间 TAR 和 CAR，如果已设定中断使能(RIER.AIER=1)，闹钟中断标志(RIIR.AIF)被置位.

#### 5.8.4.10 应用指南:

1. TAR, CAR, TLR 和CLR 寄存器为BCD格式.
2. 用户需要确认，填入的数据为有效格式。例如， CLR = 201a (年), 13 (月), 00 (日), 或CLR 与DWR部匹配，等。
3. 复位状态:

| 寄存器 | 复位状态 |
|---|---|
| **AER** | 0 |
| **CLR** | 05/1/1 (年/月/日) |
| **TLR** | 00:00:00 (小时:分:秒) |
| **CAR** | 00/00/00 (年/月/日) |
| **TAR** | 00:00:00 (小时:分:秒) |
| **TSSR** | 1 (24小时格式) |
| **DWR** | 6 (星期六) |
| **RIER** | 0 |
| **RIIR** | 0 |
| **LIR** | 0 |
| **TTR** | 0 |
| **PWRTOUT** | 5555 |

4. 在TLR 和TAR中, 仅2位BCD码用于指示“年”。目前仅支持20xx年份，不支持19xx或21xx年.

### 5.8.5 寄存器列表

**R: 只读, W: 只写, R/W: 可读写**, **C**: 仅值为0时可写

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **RTC_BA = 0x4000_8000** | | | | |
| **INIR** | RTC_BA+0x000 | R/W | RTC 复位寄存器 | 0x0000_0000 |
| **AER** | RTC_BA+0x004 | R/W | RTC 读写使能寄存器 | 0x0000_0000 |
| **FCR** | RTC_BA+0x008 | R/W | RTC 频率补偿寄存器 | 0x0000_0700 |
| **TLR** | RTC_BA+0x00C | R/W | 时间载入寄存器 | 0x0000_0000 |
| **CLR** | RTC_BA+0x010 | R/W | 日历载入寄存器 | 0x0005_0101 |
| **TSSR** | RTC_BA+0x014 | R/W | 时间格式选择寄存器 | 0x0000_0001 |
| **DWR** | RTC_BA+0x018 | R/W | 一周日期寄存器 | 0x0000_0006 |
| **TAR** | RTC_BA+0x01C | R/W | 时间闹钟寄存器 | 0x0000_0000 |
| **CAR** | RTC_BA+0x020 | R/W | 日历闹钟寄存器 | 0x0000_0000 |
| **LIR** | RTC_BA+0x024 | R | 年分割指示寄存器 | 0x0000_0000 |
| **RIER** | RTC_BA+0x028 | R/W | RTC 中断使能寄存器 | 0x0000_0000 |
| **RIIR** | RTC_BA+0x02C | R/C | RTC 中断指示寄存器 | 0x0000_0000 |
| **TTR** | RTC_BA+0x030 | R/W | RTC时钟节拍寄存器 | 0x0000_0000 |

### 5.8.6 寄存器描述

**RTC 初始寄存器 (INIR)**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| INIR | RTC_BA+0x000 | R/W | RTC初始寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:0] | **INIR** | **RTC 初始化**<br>当RTC模块上电，RTC由初始状态开始运行，用户必须写入数值(0x a5eb1357) 至INIR用于释放所有的逻辑计数器。INIR act as hardware reset circuit. |
| [0] | **Active** | **RTC 活动状态**(只读),<br>0: RTC 在复位状态<br>1: RTC 正常运行 |

### RTC读写使能寄存器(AER)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| AER | RTC_BA+0x004 | R/W | RTC读写使能寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [16] | ENF | **RTC 寄存器写入使能标志(只读)**<br>1 = RTC 寄存器读/写使能<br>0 = RTC 寄存器读/写禁止<br>对AER[15:0]寄存器载入0xA965可对该位置位。128个RTC时钟后，或者当AER[15:0]不等于 0xA965时，自动清除。 |
| [15:0] | AER | **RTC 寄存器写入使能密码(只写)**<br>0xA965 = 允许RTC读写<br>Others = 禁止RTC读写 |

### RTC频率补偿寄存器(FCR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| FCR | RTC_BA+0x008 | R/W | 频率补偿寄存器 | 0x0000_0700 |

| 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 | | | | INTEGER | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | FRACTION | | | | | |

| Bits | 描述 | |
|---|---|---|
| [11:8] | INTEGER | 整数部分 |

| 整数部分值 | FCR[11:8] | 整数部分值 | FCR[11:8] |
|---|---|---|---|
| 32776 | 1111 | 32768 | 0111 |
| 32775 | 1110 | 32767 | 0110 |
| 32774 | 1101 | 32766 | 0101 |
| 32773 | 1100 | 32765 | 0100 |
| 32772 | 1011 | 32764 | 0011 |
| 32771 | 1010 | 32763 | 0010 |
| 32770 | 1001 | 32762 | 0001 |
| 32769 | 1000 | 32761 | 0000 |

| | | |
|---|---|---|
| [5:0] | FRACTION | 分数部分<br>公式= (分数部分值) x 60<br>注: 分数部分必须按照Hex格式数据填入。参考5.8.4.4 范例 |

注: RTC使能后，该寄存器值可被读回。

### RTC时间载入寄存器(TLR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| TLR | RTC_BA+0x00C | R/W | 时间载入寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [21:20] | **10HR** | 10 Hour Time Digit (0~3)[2] |
| [19:16] | **1HR** | 1 Hour Time Digit (0~9) |
| [14:12] | **10MIN** | 10 Min Time Digit (0~5) |
| [11:8] | **1MIN** | 1 Min Time Digit (0~9) |
| [6:4] | **10SEC** | 10 Sec Time Digit (0~5) |
| [3:0] | **1SEC** | 1 Sec Time Digit (0~9) |

注:

1. TLR 为二进制计数方式，RTC不对载入的值进行检测.
2. 括号内列出的为可接受的数值.

### RTC日历载入寄存器(CLR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| CLR | RTC_BA+0x010 | R/W | 日历载入寄存器 | 0x0005_0101 |

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

| Bits | 描述 | |
|---|---|---|
| [23:20] | **10YEAR** | 10-Year Calendar Digit (0~9) |
| [19:16] | **1YEAR** | 1-Year Calendar Digit (0~9) |
| [12] | **10MON** | 10-Month Calendar Digit (0~1) |
| [11:8] | **1MON** | 1-Month Calendar Digit (0~9) |
| [5:4] | **10DAY** | 10-Day Calendar Digit (0~3) |
| [3:0] | **1DAY** | 1-Day Calendar Digit (0~9) |

注:

3. TLR 为二进制计数方式，RTC不对载入的值进行检测。
4. 括号内列出的为可接受的数值.

### RTC时间格式选择寄存器 (TSSR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| TSSR | RTC_BA+0x014 | R/W | 时间格式选择寄存器 | 0x0000_0001 |

| 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 | | | | | | | 24hr/12hr |

| Bits | 描述 |
|---|---|

| | | |
|---|---|---|
| [0] | **24hr/12hr** | **24-小时/ 12-小时模式选择**<br>用于标示TLR 和TAR 为 24-小时 或12-小时模式<br>1 = 选择24-小时制时间模式<br>0 = 选择12-小时制时间模式，带AM /PM 指示 |

| 24-小时制 | 12-小时制 | 24-小时制 | 12-小时制<br>(PM 时间+ 20) |
|---|---|---|---|
| 00 | 12(AM12) | 12 | 32(PM12) |
| 01 | 01(AM01) | 13 | 21(PM01) |
| 02 | 02(AM02) | 14 | 22(PM02) |
| 03 | 03(AM03) | 15 | 23(PM03) |
| 04 | 04(AM04) | 16 | 24(PM04) |
| 05 | 05(AM05) | 17 | 25(PM05) |
| 06 | 06(AM06) | 18 | 26(PM06) |
| 07 | 07(AM07) | 19 | 27(PM07) |
| 08 | 08(AM08) | 20 | 28(PM08) |
| 09 | 09(AM09) | 21 | 29(PM09) |
| 10 | 10(AM10) | 22 | 30(PM10) |
| 11 | 11(AM11) | 23 | 31(PM11) |

### RTC一周日期寄存器(DWR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| DWR | RTC_BA+0x018 | R/W | 一周日期寄存器 | 0x0000_0006 |

| 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 | | | | | DWR | | |

| Bits | 描述 | |
|---|---|---|
| [2:0] | DWR | 一周日期寄存器 |

| 值 | 一周日期 |
|---|---|
| 0 | 星期日 |
| 1 | 星期一 |
| 2 | 星期二 |
| 3 | 星期三 |
| 4 | 星期四 |
| 5 | 星期五 |
| 6 | 星期六 |

### RTC时间闹钟寄存器(TAR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| TAR | RTC_BA+0x01C | R/W | 时间闹钟寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [21:20] | **10HR** | 10Hour Time Digit of Alarm Setting (0~3)[2] |
| [19:16] | **1HR** | 1 Hour Time Digit of Alarm Setting (0~9) |
| [14:12] | **10MIN** | 10 Min Time Digit of Alarm Setting (0~5) |
| [11:8] | **1MIN** | 1 Min Time Digit of Alarm Setting (0~9) |
| [6:4] | **10SEC** | 10 Sec Time Digit of Alarm Setting (0~5) |
| [3:0] | **1SEC** | 1 Sec Time Digit of Alarm Setting (0~9) |

注:

1. TLR 为二进制计数方式，RTC不对载入的值进行检测。
2. 括号内列出的为可接受的数值.
3. RTC单位使能后，该寄存器值可被读回.

### RTC日历闹钟寄存器(CAR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| CAR | RTC_BA+0x020 | R/W | 日历闹钟寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [23:20] | **10YEAR** | 10-Year Calendar Digit of Alarm Setting (0~9) |
| [19:16] | **1YEAR** | 1-Year Calendar Digit of Alarm Setting (0~9) |
| [12] | **10MON** | 10-Month Calendar Digit of Alarm Setting (0~1) |
| [11:8] | **1MON** | 1-Month Calendar Digit of Alarm Setting (0~9) |
| [5:4] | **10DAY** | 10-Day Calendar Digit of Alarm Setting (0~3) |
| [3:0] | **1DAY** | 1-Day Calendar Digit of Alarm Setting (0~9) |

注:

1. TLR 为二进制计数方式，RTC不对载入的值进行检测。
2. 括号内列出的为可接受的数值.
3. RTC单位使能后，该寄存器值可被读回.

### RTC年分割指示寄存器(LIR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| LIR | RTC_BA+0x024 | R | RTC年分割指示寄存器 | 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 | | | | | | | LIR |

| Bits | 描述 | |
|---|---|---|
| [0] | LIR | 按年分割指示寄存器(只读)<br>1 = 用于指示RTC按年分割<br>0 =用于指示RTC未按年分割 |

### RTC中断使能寄存器(RIER)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| RIER | RTC_BA+0x028 | R/W | RTC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 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | | | | | TIER | AIER |

| Bits | 描述 | |
|---|---|---|
| [1] | TIER | 时钟节拍中断使能<br>1 = RTC 时钟节拍中断使能<br>0 = RTC 时钟节拍中断禁止 |
| [0] | AIER | 闹钟中断使能<br>1 = RTC 闹钟中断使能<br>0 = RTC 闹钟中断禁止 |

### RTC中断指示寄存器(RIIR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| RIIR | RTC_BA+0x02C | R/C | RTC中断指示寄存器 | 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 | | | | | | TIF | AIF |

| Bits | 描述 | |
|---|---|---|
| [1] | TIF | **RTC 时钟节拍中断标志**<br>当RTC 时钟节拍中断使能，(RIER.TIER=1), RTC根据TTR[2:0]设定周期性得使TIF置1该位通过软件清除。 |
| [0] | AIF | **RTC 闹钟中断标志**<br>当RTC 闹钟中断使能， (RIER.AIER=1), RTC 根据TLR和CLR是否达到TAR 和 CAR的设定对AIF置1。该位通过软件清除。 |

### RTC时钟节拍寄存器(TTR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| TTR | RTC_BA+0x030 | R/C | RTC时钟节拍寄存器 | 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 | | | | TWKE | TTR[2:0] | | |

| Bits | 描述 | |
|---|---|---|
| [3] | TWKE | **RTC定时器唤醒CPU使能位**<br>**如果在CPU进入掉电模式之前，TWKE 被置位，当RTC或闹钟匹配发生时，CPU被RTC单元唤醒.**<br>1= 使能将CPU由掉电模式唤醒.<br>0= 禁止唤醒功能.<br>注: 1. 时间计数根据 TTR[2:0] 描述配置. |

<table>
<tr>
<td>[2:0]</td>
<td>TTR</td>
<td>时钟节拍寄存器<br>根据TTR寄存器设定产生中断.<br>
<table>
<tr><td>TTR[2:0]</td><td>时钟节拍(秒)</td></tr>
<tr><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1/2</td></tr>
<tr><td>2</td><td>1/4</td></tr>
<tr><td>3</td><td>1/8</td></tr>
<tr><td>4</td><td>1/16</td></tr>
<tr><td>5</td><td>1/32</td></tr>
<tr><td>6</td><td>1/64</td></tr>
<tr><td>7</td><td>1/128</td></tr>
</table>
注: RTC 使能后，该寄存器值可被读回e.</td>
</tr>
</table>

## 5.9 串行外围设备接口(SPI)控制器

### 5.9.1 概述

串行外围设备接口(SPI)控制器，支持高速串行通信的SPI模块，共支持四组双向主/从机模式传输。NUC1XX支持从外围设备进行串行的方式传输数据，每个SPI模块可支持2组外围设备，但不能同时进行传输。当(CNTRL[18])被置1时，同样可以当作从机来操作。

每组SPI控制器可以单独产生终端，当响应的中断位被置1，传输同时结束。响应中断的优先级可以通过软件编辑SSR[SS_LVL] 位达成。在主机模式下，对DIVIDER写入值，可以配置传输时钟的频率。SPI_CNTRL[23]寄存器的VARCLK_EN 位被使能，传输时钟可根据DIV和DIV2 ，被分为两组不同的频率。频率差别根据VARCLK来确定

该主/从机内核包含2组32位输出/输入数据缓存，可支持突发传输 (burst mode)模式，并支持各种不同长度的变量数据模式，最多可支持64 位数据传输。

通过配置SPI_CNTL[22]，MCU也可支持2位数据传输模式，当SPI_CNTL[22],寄存器的TWOB位使能，即可传输2位数据的输入/输出。第一位数据从SPI_TX0输出或从SPI_RX0输入，第二位数据从SPI_TX1输出或从SPI_RX1输入

### 5.9.2 特性

- 四组SPI传送控制器
- 支持主/从机模式
- 支持1位，2位串行输入/输出模式
- 最大传输字可达32位
- 主机模式下多种总线时钟频率选择
- 支持突发传输 (burst mode)模式传输，输入/输出时钟可分两组定时器。
- 支持MSB 或 LSB 为最先传输模式
- 主机模式2条从机器件选择，从机模式下1条从机器件选择
- 同一组时钟下，全静态同步设计
- 字节睡眠模式
- 支持2组可编辑频率时钟模组

### 5.9.3 SPI模块方块图 (主机/从机)

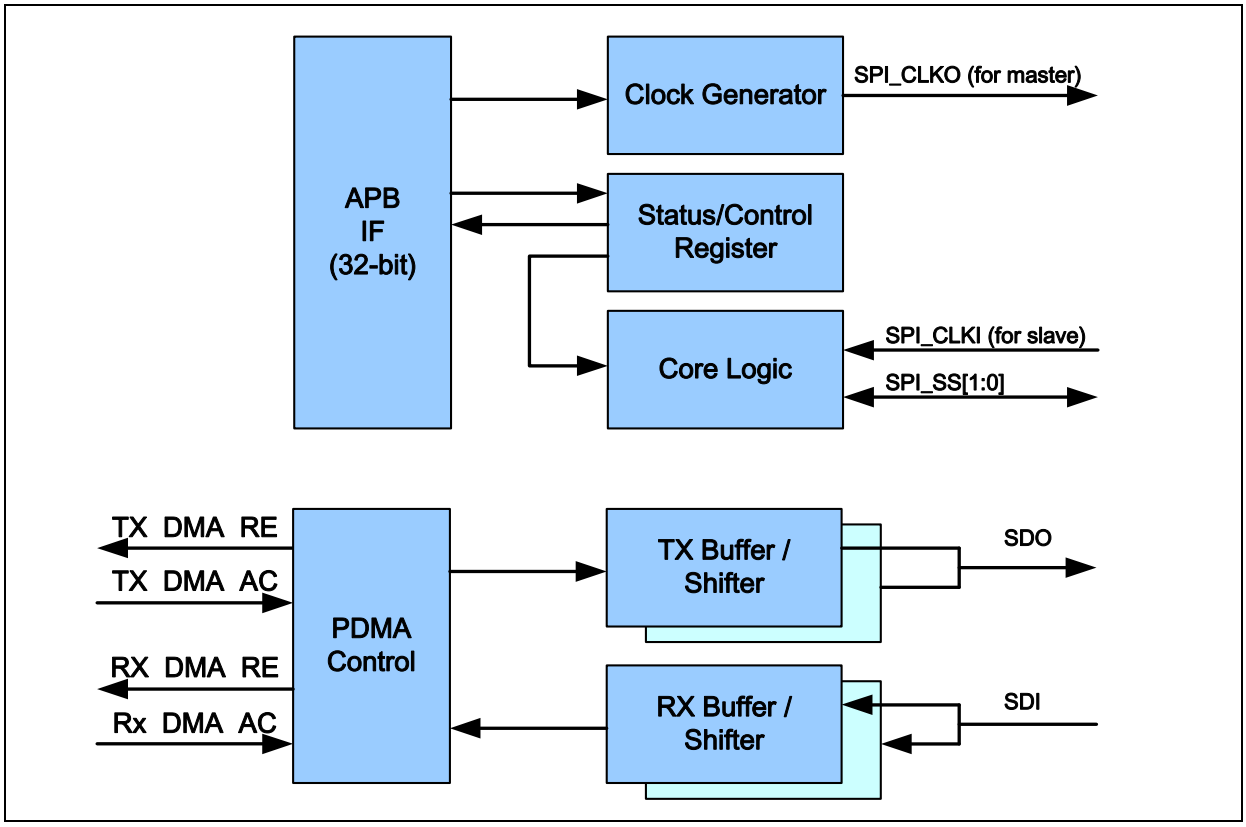



图 5-44 SPI 模块方块图 (主机/从机)

### 5.9.4 SPI时序波形图(主机/从机)

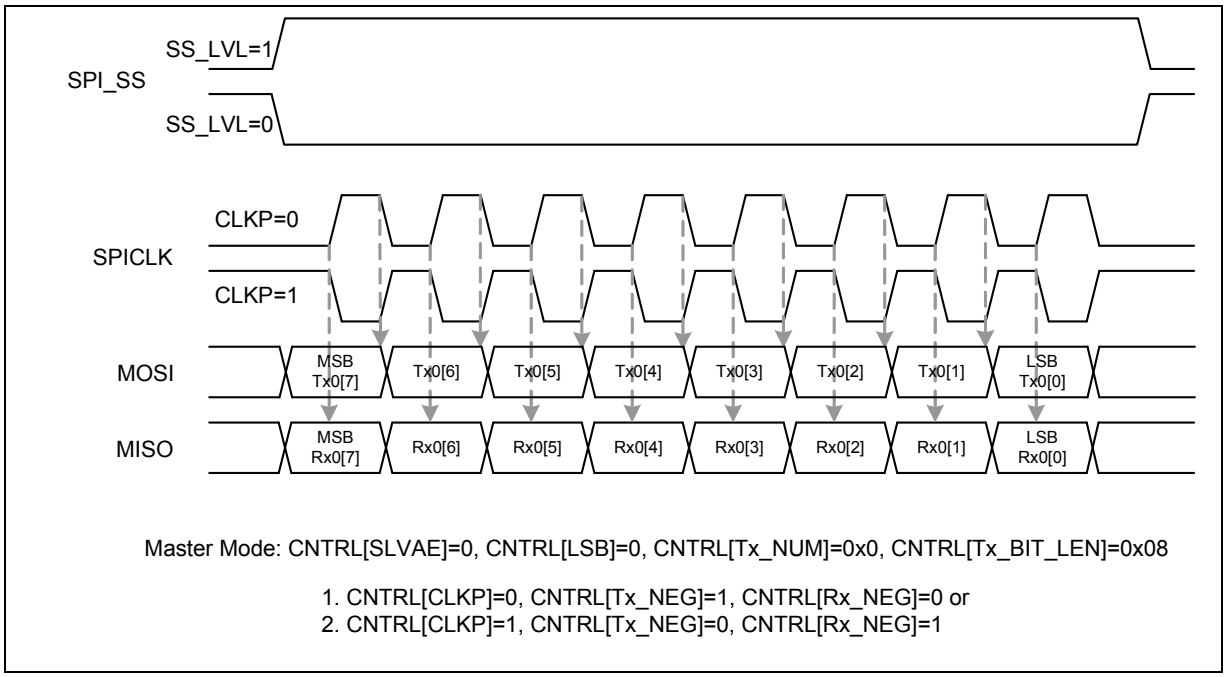



图 5-45 MICROWIRE/SPI 时序(主机)

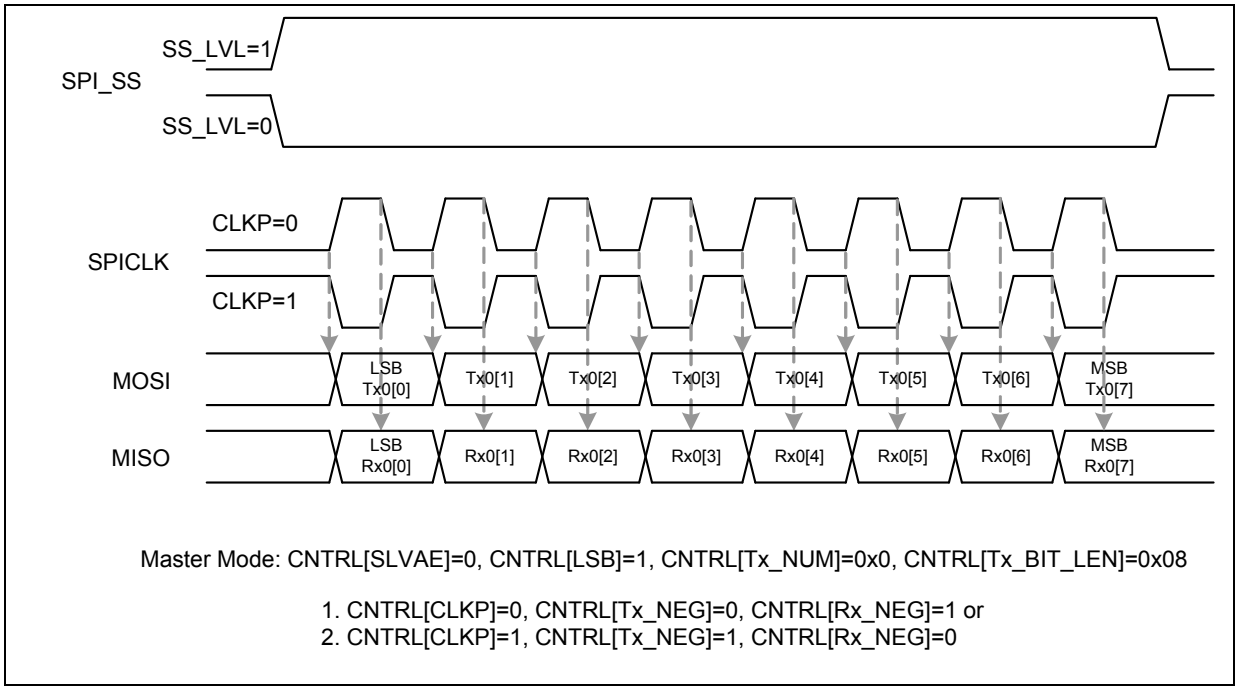



图 5-46 相位变化的 SCLK 时钟时序 (主机)

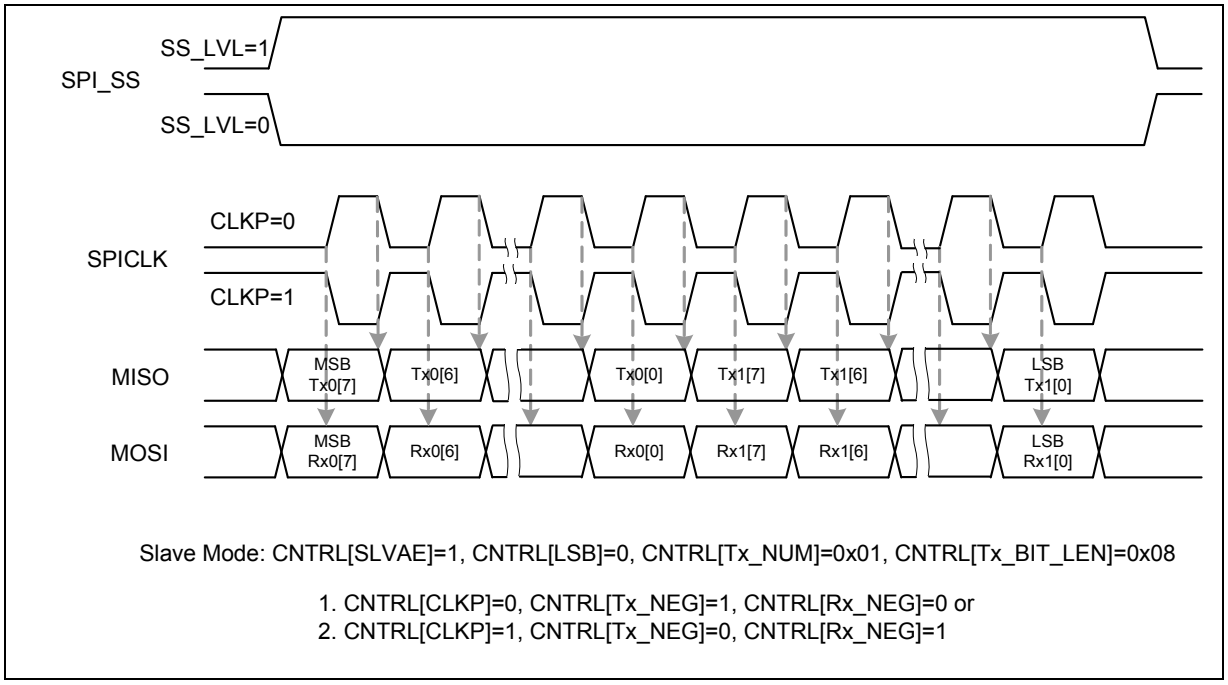



图 5-47 M ICROWIRE /SPI 时序 (从机)

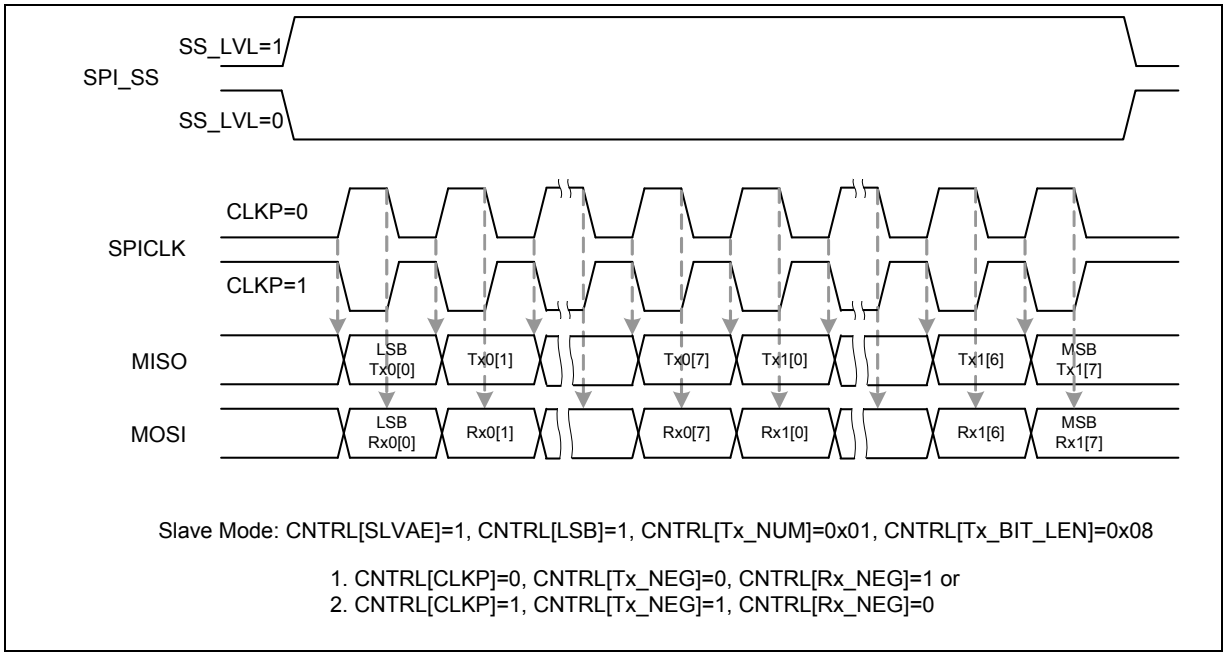



图 5-48 相位变化的 SCLK 时钟时序(从机)

### 5.9.5 SPI 编程例程

SPI作为主机时，与 一个从机设备通信如下:

- 数据在时钟上升沿锁存
- 数据在时钟下将沿传输
- 数据先传送MSB
- SCLK 空闲模式低响应
- 每次输出/接收一个字节
- 从机选择信号低响应

完成一组基础传输数据，请严格按照如下顺序: (连接的从机也必须配合下述流程)

1) 对DIVIDER内写入值，确定传输频率。

2) 写SSR，设定 ASS = 0, SS_LVL = 0 且SSR[0] 或SSR[1] =1 用以使能模块。

向从机发送(写)数据时:

3) 向SPI_Tx0[7:0]寄存器内写入值.

当由从机接受(读)数据时:

4) 对SPI_Tx0寄存器写入 0xFFFFFFFF

5) 写CNTRL, 设定 SLAVE = 0, CLKP = 0, Rx_NEG = 0, Tx_NEG = 1, Tx_BIT_LEN = 0x08, Tx_NUM = 0x0, LSB = 0, SLEEP = 0x0 及GO_BUSY = 1 用于开始传输.

-- 等待中断发生(确保此时 IE = 1) 或者查询GO_BUSY 位直至该位为 0 --

6) 从SPI_Rx0寄存器读出数据.

7) 重复由步骤3 开始的动作，获设SSR[0] 或SSR[1] 为 0 ，关闭模块结束传输.

当MCU作为从机时，支持外部主机读取MCU:

- 数据在时钟上升沿锁存
- 数据在时钟下将沿传输
- 数据先传送MSB
- SCLK 空闲模式高响应
- 每次输出/接收一个字节
- 从机选择信号高响应

完成一组基础传输数据，请严格按照如下顺序: (连接的主机也必须配合下述流程)

1) 写SSR, 设定SS_LVL = 1.

向主机发送(写)数据时:

2) 向SPI_Tx0[7:0]寄存器内写入值.

由主机接受(读)数据时:

3) 对SPI_Tx0寄存器写入 0xFFFFFFFF

4) 写CNTRL, 设定SLAVE = 1, CLKP = 1, Rx_NEG = 0, Tx_NEG = 1, Tx_BIT_LEN = 0x08, Tx_NUM = 0x0, LSB = 1, 及 GO_BUSY = 1 开始传输，等待外部主机发出从机选择，并且接受到时钟信号

-- 等待中断发生(确保此时 IE = 1) 或者查询GO_BUSY 位直至该位为 0 --

5) 从SPI_Rx0寄存器读出数据.

6) 重复由步骤2 开始的动作，传输剩余的数据.

### 5.9.6 改变时钟频率

在主机模式下，串行时钟频率可以编辑配置。当VARCLK_EN使能后，速率由VARCLK寄存器定义。如果VARCLK为0，则输出频率由DIVIDER决定，如果VARCLK=1，则输出频率由DIVIDER2 决定。下图用来说明串行时钟SPICLK与VARCLK, DIVIDER, DIVIDER2输出的关系。VARCLK内有两位来决定一个时钟。由VARCLK[31:30] 定义SCLKO的第一个时钟周期。时钟源由VARCLK定义，当重新定义后，必须等待一个周期，下一周期开始，时钟频率改变，例如SPICLK里有5个CLK1 ，则VARCLK需要设定9个“0”，在第10位设定为1用于下个时钟源切换到CLK2.

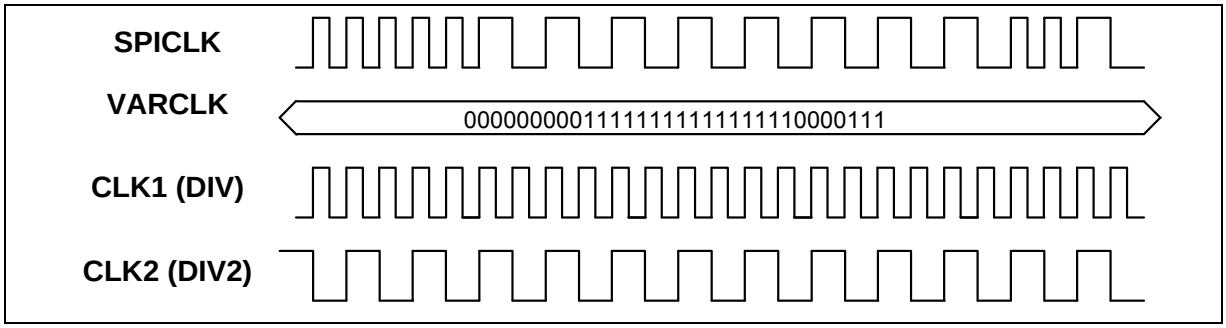



图 5-49 可变时钟频率

### 5.9.7 SPI 串行总线控制寄存器列表

**R: 只读, W: 只写, R/W: 可读写**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **SPI0_BA = 0x4003_0000**<br>**SPI1_BA = 0x4003_4000**<br>**SPI2_BA = 0x4013_0000**<br>**SPI3_BA = 0x4013_4000** | | | | |
| **SPI_CNTRL** | SPIx_BA + 0x00 | R/W | 控制及状态寄存器 | 0x0000_0004 |
| **SPI_DIVIDER** | SPIx_BA + 0x04 | R/W | 时钟除频寄存器 | 0x0000_0000 |
| **SPI_SSR** | SPIx_BA + 0x08 | R/W | 从机选择寄存器 | 0x0000_0000 |
| 保留 | SPIx_BA + 0x0C | N/A | 保留 | N/A |
| **SPI_Rx0** | SPIx_BA + 0x10 | R | 接收数据寄存器0 | 0x0000_0000 |
| **SPI_Rx1** | SPIx_BA + 0x14 | R | 接收数据寄存器1 | 0x0000_0000 |
| 保留 | SPIx_BA + 0x18 | N/A | 保留 | N/A |
| 保留 | SPIx_BA + 0x1C | N/A | 保留 | N/A |
| **SPI_Tx0** | SPIx_BA + 0x20 | W | 数据发送寄存器0 | 0x0000_0000 |
| **SPI_Tx1** | SPIx_BA + 0x24 | W | 数据发送寄存器1 | 0x0000_0000 |
| 保留 | SPIx_BA + 0x28 | N/A | 保留 | N/A |
| 保留 | SPIx_BA + 0x2C | N/A | 保留 | N/A |
| 保留 | SPIx_BA +0x30 | R/W | Dongle Joystick控制寄存器 | N/A |
| **SPI_VARCLK** | SPIx_BA +0x34 | R/W | 可变时钟标志寄存器 | 0x007f_ff87 |
| **SPI_DMA** | SPIx_BA+0x38 | R/W | SPI DMA 控制寄存器 | 0x0000_0000 |

**注 1: 由软件编写 CNTRL 寄存器, GO_BUSY 位必须最后写入.**

### 5.9.8 SPI (MICROWIRE) 控制寄存器描述

**SPI控制寄存器描述(SPI_CNTRL)**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| SPI_CNTRL | SPIx_BA + 0x00 | R/W | 控制及状态寄存器 | 0x0000_0004 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| VARCLK_EN | TWOB | 保留 | BYTE_ENDIAN | BYTE_SLEEP | SLAVE | IE | IF |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| SLEEP | | | | CLKP | LSB | Tx_NUM | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Tx_BIT_LEN | | | | | Tx_NEG | Rx_NEG | GO_BUSY |

| Bits | 描述 | |
|---|---|---|
| [31:24] | 保留 | 保留 |
| [23] | **VARCLK_EN** | **允许改变时钟(仅主机模式下)**<br>0 = 输出时钟根据DIV的值锁定。<br>1 = 输出时钟频率可变。由 VARCLK, DIVIDER 及DIVIDER2决定<br>注:当**VARIABLE CLOCK**使能,**Tx_BIT_LEN**必须设定为0x10 (16位模式) |
| [22] | **TWOB** | **两位数据写入/读出使能**<br>0 = 禁止两位数据写入/读出模式.<br>1 =使能两位数据写入/读出模式.<br>当使能TWOB, 数据写入 SDO[1:0]从SPI_TX1/0传出，数据由SPI_RX1/0接收并写入SDI[1:0].<br>注:当**TWOB**使能**, Tx_NUM**必须设定为0x00 |
| [21] | 保留 | 保留 |
| [20] | **BYTE_ENDIAN** | **BYTE ENDIAN**<br>0 = 禁止 BYTE ENDIAN 功能<br>1 = 允许 BYTE ENDIAN. 仅支持TX_BIT_LEN 内16, 24, 及32 位定义<br>当传输时以MSB先传输时，当TX_BIT_LEN=32位，TX缓冲区内的排列方式为 [BYTE0, BYTE1, BYTE2, BYTE3]格式，数据输出的方式为BYTE0, BYTE1, BYTE2, and BYTE3。，如果 TX_BIT_LEN=24位，TX缓冲区内的排列方式为 [BYTE3, BYTE0, BYTE1, BYTE2]，数据输出方式为BYTE0, BYTE1, 及BYTE2。16位格式如上。 |

| | | |
|---|---|---|
| [19] | **BYTE_SLEEP** | **BYTE SLEEP**<br>0 = word模式下传输时每个字节间隙不增加SLEEP。<br>1 = word模式下传输时，通过定义 SPI_CNTRL[15:12]，每个字节间增加SLEEP。<br>当**BYTE_SLEEP**使能**, Tx_BIT_LEN**必须设定为0x00 (32 位模式) |
| [18] | **SLAVE** | **SLAVE 模式标志**<br>0 = MCU作为主机模式<br>1 = MCU作为从机模式. |
| [17] | **IE** | **中断使能**<br>0 = 禁止MICROWIRE/SPI 中断.<br>1 = 使能MICROWIRE/SPI中断. |
| [16] | **IF** | **中断标志**<br>0 = 表示传输未结束<br>1 = 表示传输完成。当SPI使能，该位置1。<br>**注：** 该位用于读取状态，可以置1，但不能清0 。 |
| [15:12] | **SLEEP** | **暂缓间隙 (主机模式)**<br>该位用于编辑增加在两次成功传输内的空闲时间。默认值为0x0。当CNTRL[Tx_NUM] = 00, 传输间不增加任何时间。间隙时间根据如下的等式增加( 从当前SCLK的最后一个下降沿直至下一个上升沿)。<br>(CNTRL[SLEEP] + 2)*SCLK周期<br>SLEEP = 0x0 … 2 SCLK clock cycle<br>SLEEP = 0x1 … 3 SCLK clock cycle<br>……<br>SLEEP = 0xe … 16 SCLK clock cycle<br>SLEEP = 0xf … 17 SCLK clock cycle |
| [11] | **CLKP** | **时钟极性**<br>0 = SCLK 低电平闲置.<br>1 = SCLK 高电平闲置. |
| [10] | **LSB** | **优先传送LSB**<br>0 = 优先发送/接收MSB (根据CNTRL寄存器的Tx_BIT_LEN 决定TxX/RxX).<br>1 = 优先发送 LSB (TxX[0]位), 接收到的首位数居定为LSB 优先送入 Rx 寄存器 (RxX[0]位). |
| [9:8] | **Tx_NUM** | **发送/接收数量**<br>该寄存器用于标示一次成功传输中，传输的数量。<br>00 = 每次传输仅完成一次发送/接收<br>01 = 每次传输完成两次发送/接收<br>10 = 保留.<br>11 = 保留. |

| | | |
|---|---|---|
| [7:3] | **Tx_BIT_LEN** | **传输位长度**<br>该寄存器用于标示一次传输中，完成的传输长度，最高纪录32位。<br>Tx_BIT_LEN = 0x01 … 1 bit<br>Tx_BIT_LEN = 0x02 … 2 bits<br>……<br>Tx_BIT_LEN = 0x1f … 31 bits<br>Tx_BIT_LEN = 0x00 … 32 bits |
| [2] | **Tx_NEG** | **发送数据边沿反向位**<br>0 = SDO 信号在SPICLK的上升沿发送。<br>1 = SDO 信号在SPICLK的下降沿发送。 |
| [1] | **Rx_NEG** | **接收数据边沿反向位**<br>0 = SDI 信号在SPICLK上升沿接收.<br>1 = SDI 信号在SPICLK下降沿接收. |
| [0] | **GO_BUSY** | **通讯或忙状态标志**<br>0 = 写入0表明线上无通讯。<br>1 = 写入1表明线上有传输。该位置1说明线上仍有传输，传输完成该位自动清0。<br>**注**: 在对CNTRL寄存器的GO_GOBY置1之前，必须先配置相应的寄存器。在传输过程中再对其他寄存器进行配置，无法影响传输过程。 |

### SPI除频寄存器(SPI_DIVIDER)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **SPI_DIVIDER** | SPIx_BA + 0x04 | R/W | 时钟除频寄存器(仅主机模式) | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| **DIVIDER2[15:8]** | | | | | | | |
| **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| **DIVIDER2[7:0]** | | | | | | | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| **DIVIDER[15:8]** | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **DIVIDER[7:0]** | | | | | | | |

| Bits | 描述 |
|---|---|

| | | |
|---|---|---|
| [31:16] | **DIVIDER2** | **时钟除频2寄存器**<br>该寄存器根据PCLK内容进行除频，配置后SPICLK口输，具体频率如下图:<br>$f_{sclk} = \frac{f_{pclk}}{(DIVIDER+1)*2}$<br>**注:** 在主机模式下，建议DIVIDER2 内至少填入值1 |
| [15:0] | **DIVIDER** | **时钟除频寄存器**<br>该寄存器根据PCLK内容进行除频，配置后SPICLK口输，具体频率如下图::<br>$f_{sclk} = \frac{f_{pclk}}{(DIVIDER+1)*2}$<br>**注:** 在主机模式下，建议DIVIDER 至少填入数值1 。在从机模式下SPICLK内输入值应为PCLK的5倍。 |

从机选择寄存器(SPI_SSR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| SPI_SSR | SPI0_BA + 0x08 | R/W | 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 | | LTRIG_FLAG | SS_LTRIG | ASS | SS_LVL | SSR[1:0] | |

| Bits | 描述 | |
|---|---|---|
| [31:6] | 保留 | 保留 |
| [5] | LTRIG_FLAG | **电平触发标志**<br>在从机模式下SS_LTRIG置位，该标志能够表示是否达到要求。<br>1: 接收数量和接收位达到TX_NUM 及TX_BIT_LEN内的值.<br>0: 接收数量或接收位没有符合值。.<br>注：该位只读 |
| [4] | SS_LTRIG | **从机电平触发选择**<br>0: 从机输入边沿触发。该为默认值<br>1: 从机选择由电平触发。根据 SS_LVL选择是高电平/低电平触发。 |
| [3] | ASS | **自动从机选择(主机模式)**<br>0 = 该位清位，从机是否发出信号，由设置或清除SSR[1:0]寄存器决定。.<br>1 = 该位置位，SPI_SS[1:0]信号自动产生。这说明在SSR[1:0]寄存器内的从机选择信号，由SPI传输情况及CNTRL[GO_BUSY]内的状态决定并产生，并且根据信号收发一致持续到收发结束。 |
| [2] | SS_LVL | **从机选择触发电平选择**<br>该位决定SPI_SS[1:0]寄存器内信号根据哪个电平触发<br>0 = SPI_SS[1:0] 从机选择低电平时相应。<br>1 = The SPI_SS[1:0]从机选择 高电平时相应。 |

| | | |
|---|---|---|
| [1:0] | **SSR** | **从机选择寄存器(主机模式)**<br>当ASS位被清除，对该寄存器任何一位写1，将会激活SPI_SS [1:0]线，写0线上为非活动状态。<br>当ASS位被设置，对该寄存器任何一位写1，将会使SPI_SS [1:0]线上自动传输/接受数据。写0为非活动状态。 (由SS_LVL决定活动级).<br>注:仅在NUC1XX内使用SSR[1:0].<br>**注:** 在限定时间内，传输界面仅驱动一个器件/从机。因此，相应从机必须选中才能进行读写传输。<br>**注:** 在从机模式下，SPI_SS [0]同样被定义为器件/从机输入模式。输入SPI_SS[0]的从机选择信号必须根据SS_LVL的设定，由时钟边沿或电平触发，否则SPI模块不工作，直道下一个触发信号，或者SPI模块被重置。 |

### SPI数据接收寄存器(SPI_RX)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **SPI_Rx0** | SPIx_BA + 0x10 | R | 数据接收寄存器0 | 0x0000_0000 |
| **SPI_Rx1** | SPIx_BA + 0x14 | R | 数据接收寄存器1 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:0] | **Rx** | **数据接收寄存器**<br>数据接收寄存器内保存最后一次传输所接收的数据。数据的长度根据CNTRL寄存器内定义的长度决定。例如，CNTRL[Tx_BIT_LEN] 设定为 0x08 且CNTRL[Tx_NUM] 设定为0x0, Rx0[7:0] 内保存传输数据.<br>**注:** 数据接收寄存器应只用于读取，对该寄存器写入值会对数据发送寄存器产生影响，因为它们使用同一组flip-flops. |

### SPI数据发送寄存器(SPI_TX)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **SPI_Tx0** | SPIx_BA + 0x20 | W | 数据发送寄存器 0 | 0x0000_0000 |
| **SPI_Tx1** | SPIx_BA + 0x24 | W | 数据发送寄存器1 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:0] | **Tx** | **数据发送寄存器**<br>数据发送寄存器内存储下一次被发送的数据。数据的长度根据CNTRL寄存器内定义的长度决定。例如，CNTRL[Tx_BIT_LEN] 设定为 0x08 且CNTRL[Tx_NUM] 设定为0x0, Tx0[7:0] 内的数据将被发送。如果CNTRL[Tx_BIT_LEN] 设定为 0x00 且CNTRL[Tx_NUM] 设定为0x1,模块将确保2个32位数据发送/接收，与设定Tx0[31:0], Tx1[31:0]具有相同的效果。<br>**注:** RxX 和TxX 寄存器使用同一组flip-flops，这说明接收到的数据同样会存放到发送的模块中，如果对TxX寄存器内的值不进行修改，输入的数据将被再次发送。 |

### SPI可变时钟标志寄存器 (SPI_VARCLK)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| SPI_VARCLK | SPIx_BA + 0x34 | R/W | 时钟频率设定标志寄存器 | 0x007f_ff87 |

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

| Bits | 描述 | |
|---|---|---|
| [31:0] | VARCLK | 可变时钟标志<br>该寄存器内存储系统时钟SPICLK的频率。如果VARCLK =0, SPICLK的输出频率根据DIVIDER内的值设定，如果VARCLK = 1, SPICLK 输出频率根据DIVIDER2内的设定。 |

### DMA控制模式(DMACTL)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| SPI_DMA | SPIx_BA + 0x38 | R/W | SPI DMA 模式控制寄存器 | 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 | | | | | | RX_DMA_GO | TX_DMA_GO |

| Bits | 描述 | |
|---|---|---|
| [1] | RX_DMA_GO | **接收DMA 开始**<br>对该位置1，开始DMA接收模式，SPI模块自动切换为DMA模式接收。 |
| [0] | TX_DMA_GO | **发送DMA开始**<br>对该位置1，开始DMA发送模式，SPI模块自动切换为DMA发送。<br>如果采用DMA发送数据，将不会对SPI_CNTRL 寄存器的GO_BUSY位置位。DMA控制下，无论是否必要，SPI内的DMA模块都会对该位置位。 |

## 5.10 定时器控制器

### 5.10.1 通用定时器控制器

定时器模块包含4组，TIMER0~TIMER3 (TIMER0, TIMER1位于 AHB1 ， TIMER2 和 TIMER3 位于 AHB2), 提供用户便捷的计数定时功能。定时器模块可支持例如频率测量，计数，间隔时间测量，时钟产生，延迟时间等功能。定时器模块具有调节定时，可变计数周期以及相关的详细数据。定时器可产生中断信号并根据中断发生点记录当前数据.

### 5.10.2 特征

- 各通道有独立的时钟源(TMR0_CLK, TMR1_CLK, TMR2_CLK, TMR3_CLK).
- 时间溢出周期= (Period of timer clock input) * (8-bit Prescale + 1) * (24-bit TCMP)
- 最大计数周期= (1 / 25 MHz) * (2^8) * (2^24), if TCLK = 25 MHz.
- 内部24位向上计数器，通过TDR（定时器数据寄存器）可读取.

### 5.10.3 定时器控制器框图

每个通道带一个8位预分频计数器，一个24位向上计数器，一个24位比较寄存器和一个中断请求信号. 参阅图 5-50 的定时器控制框图. 每个通道有5个时钟源选项, 图 5-51 为时钟源控制功能.

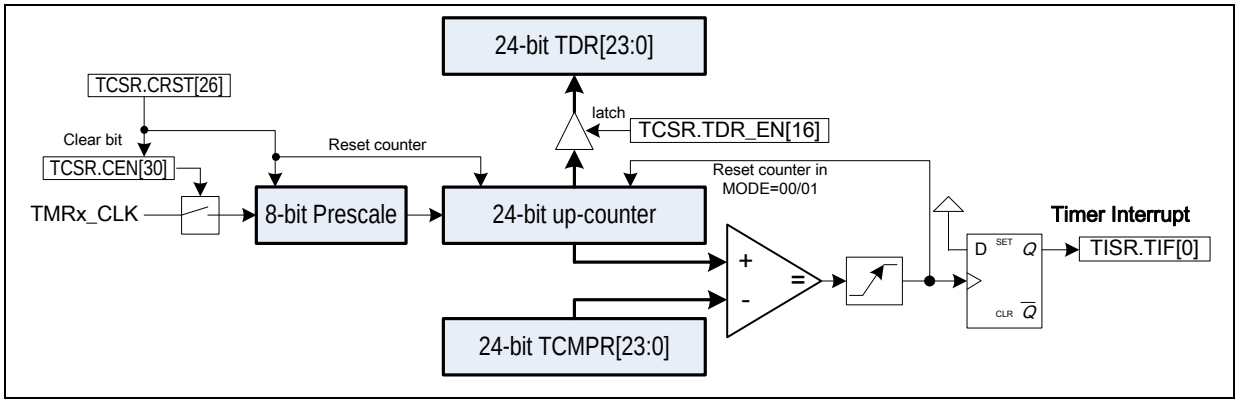



图 5-50 定时器控制器框图

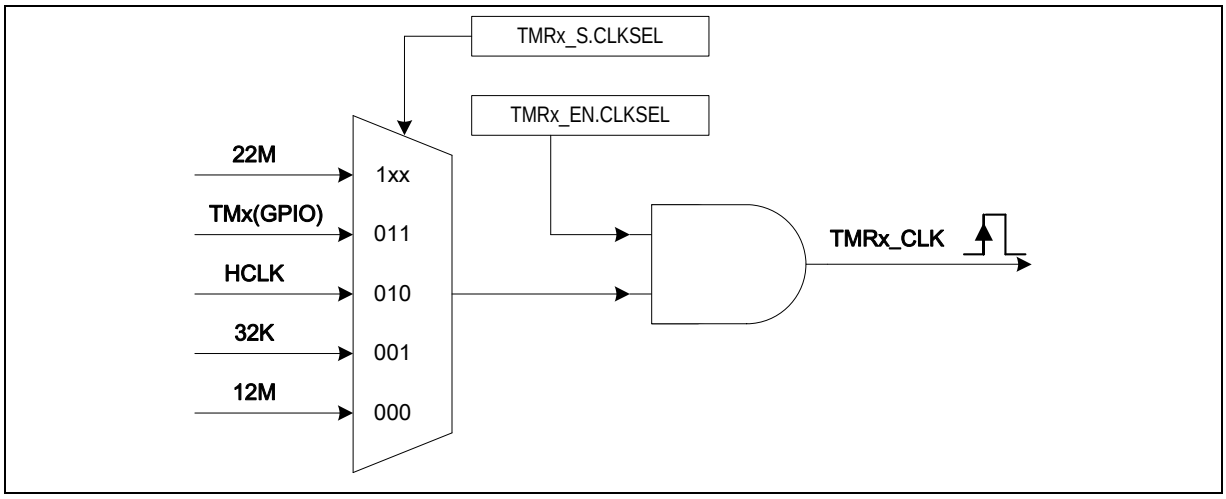



图 5-51 定时器控制的时钟源

### 5.10.4 定时器控制器寄存器图

**R: 只读, W: 只写, R/W: 可读写**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **TMR_BA01 = 0x4001_0000**<br>**TMR_BA23 = 0x4011_0000** | | | | |
| **TCSR0** | TMR_BA01+00 | R/W | Timer0 控制和状态寄存器 | 0x0000_0005 |
| **TCMPR0** | TMR_BA01+04 | R/W | Timer0 比较寄存器 | 0x0000_0000 |
| **TISR0** | TMR_BA01+08 | R/W | Timer0 中断状态寄存器 | 0x0000_0000 |
| **TDR0** | TMR_BA01+0C | R | Timer0 数据寄存器 | 0x0000_0000 |
| **TCSR1** | TMR_BA01+20 | R/W | Timer1 控制和状态寄存器 | 0x0000_0005 |
| **TCMPR1** | TMR_BA01+24 | R/W | Timer1 比较寄存器 | 0x0000_0000 |
| **TISR1** | TMR_BA01+28 | R/W | Timer1 中断状态寄存器 | 0x0000_0000 |
| **TDR1** | TMR_BA01+2C | R | Timer1 数据寄存器 | 0x0000_0000 |
| **TCSR2** | TMR_BA23+00 | R/W | Timer2 控制和状态寄存器 | 0x0000_0005 |
| **TCMPR2** | TMR_BA23+04 | R/W | Timer2 比较寄存器 | 0x0000_0000 |
| **TISR2** | TMR_BA23+08 | R/W | Timer2 中断状态寄存器 | 0x0000_0000 |
| **TDR2** | TMR_BA23+0C | R | Timer2 数据寄存器 | 0x0000_0000 |
| **TCSR3** | TMR_BA23+20 | R/W | Timer3 控制和状态寄存器 | 0x0000_0005 |
| **TCMPR3** | TMR_BA23+24 | R/W | Timer3 比较寄存器 | 0x0000_0000 |
| **TISR3** | TMR_BA23+28 | R/W | Timer3 中断状态寄存器 | 0x0000_0000 |
| **TDR3** | TMR_BA23+2C | R | Timer3 数据寄存器 | 0x0000_0000 |

定时器控制寄存器(TCSR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **TCSR0** | TMR_BA01+000 | R/W | Timer0 控制与状态寄存器 | 0x0000_0005 |
| **TCSR1** | TMR_BA01+020 | R/W | Timer1控制与状态寄存器 | 0x0000_0005 |
| **TCSR2** | TMR_BA23+000 | R/W | Timer2控制与状态寄存器 | 0x0000_0005 |
| **TCSR3** | TMR_BA23+020 | R/W | Timer3控制与状态寄存器 | 0x0000_0005 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| 保留 | CEN | IE | MODE[1:0] | | CRST | CACT | 保留 |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 保留 | | | | | | | TDR_EN |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| 保留 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PRESCALE[7:0] | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [31] | 保留 | 保留 |
| [30] | **CEN** | **计数器使能位**<br>0 = 停止/暂停计数<br>1 = 开始计数<br>**注1**: 设置CEN为1，使能24位计数器从上次停止的计数值继续计数.<br>**注2**: 在one-shot模式下(MODE[28:27]=00b)，当相应的定时中断产生时(IE[29]=1b)，该位由硬件自动清零. |
| [29] | **IE** | **中断使能**<br>0 = **禁止**定时器中断<br>1 = **使能**定时器中断<br>当定时器中断使能，当计数值与TCMPR寄存器内数值相同是，触发中断 |

<table>
<tr><td>[28:27]</td><td>MODE</td><td>定时器工作模式<br><table><tr><th>模式</th><th>定时器工作模式</th></tr><tr><td>00</td><td>当定时器定义为单触发模式(one-shot)时，定时器溢出仅触发中断一次(IE 使能),进入中断后CEN自动清除为0。.</td></tr><tr><td>01</td><td>T当定时器定义为周期模式(period)时，定时器每次溢出都触发中断(IE 使能).</td></tr><tr><td>10</td><td>保留.</td></tr><tr><td>11</td><td>定时器工作在连续计数模式，如果TDR = TCMPR (if IE is enabled)，将产生相应中断信号，24位向上计数器连续计数，除非重置.</td></tr></table></td></tr>
<tr><td>[26]</td><td>CRST</td><td>计数器重置<br>设置该位将重置定时器计数器，预分频和使CEN为0.<br>0 = 无动作.<br>1 = 重置定时器的预分频计数器，内部24位向上计数器和CEN位.</td></tr>
<tr><td>[25]</td><td>CACT</td><td>定时器工作状态<br>该位表示当前定时器计数器的状态。<br>0 = 定时器未工作。<br>1 = 定时器工作中。.</td></tr>
<tr><td>[24:17]</td><td>保留</td><td>保留</td></tr>
<tr><td>[16]</td><td>TDR_EN</td><td>数据锁存使能<br>当置位TDR_EN，TDR (Timer数据寄存器) 将每四个定时器时钟更新24位向上计数器的值<br>1 = Timer数据寄存器 更新使能<br>0 = Timer数据寄存器 禁止更新</td></tr>
<tr><td>[15:8]</td><td>保留</td><td>保留</td></tr>
<tr><td>[7:0]</td><td>PRESCALE</td><td>预分频计数器<br>时钟输入根据Prescale数值+1进行预分频。如果数值为0，不进行预分频</td></tr>
</table>

### 定时器比较寄存器(TCMPR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **TCMPR0** | TMR_BA01+004 | R/W | Timer0 比较寄存器 | 0x0000_0000 |
| **TCMPR1** | TMR_BA01+024 | R/W | Timer1比较寄存器 | 0x0000_0000 |
| **TCMPR2** | TMR_BA23+004 | R/W | Timer2比较寄存器 | 0x0000_0000 |
| **TCMPR3** | TMR_BA23+024 | R/W | Timer3比较寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [24:0] | **TCMP** | **定时器比较值**<br>TCMP是24位比较寄存器. 当内部24位向上计数器的值与TCMP的值相当时, 如果TCSR.IE[29]=1，就产生定时器中断请求. TCMP的值为定时器计数周期.<br>定时溢出周期= (Period of timer clock input) * (8-bit Prescale + 1) * (24-bit TCMP)<br>**注1:** 不能在TCMP里写0x0或0x1，否则内核将运行到未知状态.<br>**注2:** 无论CEN为0或1，软件向该寄存器写入新的值，TIMER将使用新的值并退出当前计数，开始重新计数. |

定时器中断状态寄存器(TISR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **TISR0** | TMR_BA01+08 | R/W | Timer0 中断状态寄存器 | 0x0000_0000 |
| **TISR1** | TMR_BA01+28 | R/W | Timer1 中断状态寄存器 | 0x0000_0000 |
| **TISR2** | TMR_BA23+08 | R/W | Timer2 中断状态寄存器 | 0x0000_0000 |
| **TISR3** | TMR_BA23+28 | R/W | Timer3 中断状态寄存器 | 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 | | | | | | | **TIF** |

| Bits | 描述 | |
|---|---|---|
| [31:1] | 保留 | 保留 |
| [0] | **TIF** | **定时器中断标志**<br>定时器中断状态位.<br>当内部24位计数器与TCMP的值匹配时，TIF由硬件置位，写1清该位. |

### Timer 数据寄存器 (TDR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **TDR0** | TMR_BA01+0C | R/W | Timer0数据寄存器 | 0x0000_0000 |
| **TDR1** | TMR_BA01+2C | R/W | Timer1数据寄存器 | 0x0000_0000 |
| **TDR2** | TMR_BA23+0C | R/W | Timer2数据寄存器 | 0x0000_0000 |
| **TDR3** | TMR_BA23+2C | R/W | Timer3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** |
| **TDR[23:16]** | | | | | | | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| **TDR[15:8]** | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **TDR[7:0]** | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [31:24] | 保留 | 保留 |
| [23:0] | **TDR** | **Timer**数据寄存器<br>TCSR.TDR_EN 置1时，内部24位定时器的值加载到TDR中，用户可以读取该寄存器的值. |

## 5.11 看门狗定时器

看门狗定时器是在软件出问题时执行系统复位功能，可以防止系统无限止地挂机，除此之外，看门狗定时器还可将CPU由掉电模式唤醒。看门狗定时器包含一个19位的自动运行的计数器，可编程其定时溢出间隔.

设置WTE(WDTCR[7])使能看门狗定时器和WDT计数器开始计数. 当计数器达到选择的定时溢出间隔，看门狗定时器中断标志WTIF被立即置位，并请求WDT中断（如果看门狗定时器中断使能位WTIE置位），同时，在溢出有一个有指定延时。用户必须设置WTR(WDTCR[0]) (Watchdog timer reset)为高，重置19位WDT计数器，防止CPU复位，WTR在WDT计数重置后自动由硬件清零。通过设置WTIS(WDTCR[10:8])选择8个定时溢出间隔. 如果在特殊延迟时间终止后，如果WDT计数没有被清零，看门狗定时将置位看门狗定时器重置标志(WTRF)为高并使CPU复位. 这个复位将持续64个WDT时钟，然后CPU重启，并从复位向量(0x0000 0000)执行程序. 用户可用软件拉低WTFR，WTRF将不被 看门狗重置.

表 5-6 看门狗定时溢出间隔选择

| WTIS | Interrupt Timeout | Watchdog Reset Timeout | WTR Timeout Interval (WDT_CLK=12MHz) | WTR Timeout Interval (WDT_CLK=32KHz) |
|---|---|---|---|---|
| 000 | $2^4$ WDT_CLK | ($2^4$ + 1024) WDT_CLK | 69.33 us | 31.7 ms |
| 001 | $2^6$ WDT_CLK | ($2^6$ + 1024) WDT_CLK | 72.53 us | 33.2 ms |
| 010 | $2^8$ WDT_CLK | ($2^8$ + 1024) WDT_CLK | 85.33 us | 39 ms |
| 011 | $2^{10}$ WDT_CLK | ($2^{10}$ + 1024) WDT_CLK | 170.67 us | 64 ms |
| 100 | $2^{12}$ WDT_CLK | ($2^{12}$ + 1024) WDT_CLK | 426.67 us | 160 ms |
| 101 | $2^{14}$ WDT_CLK | ($2^{14}$ + 1024) WDT_CLK | 1.45 ms | 544 ms |
| 110 | $2^{16}$ WDT_CLK | ($2^{16}$ + 1024) WDT_CLK | 5.55 ms | 2080 ms |
| 111 | $2^{18}$ WDT_CLK | ($2^{18}$ + 1024) WDT_CLK | 21.93 ms | 8224 ms |

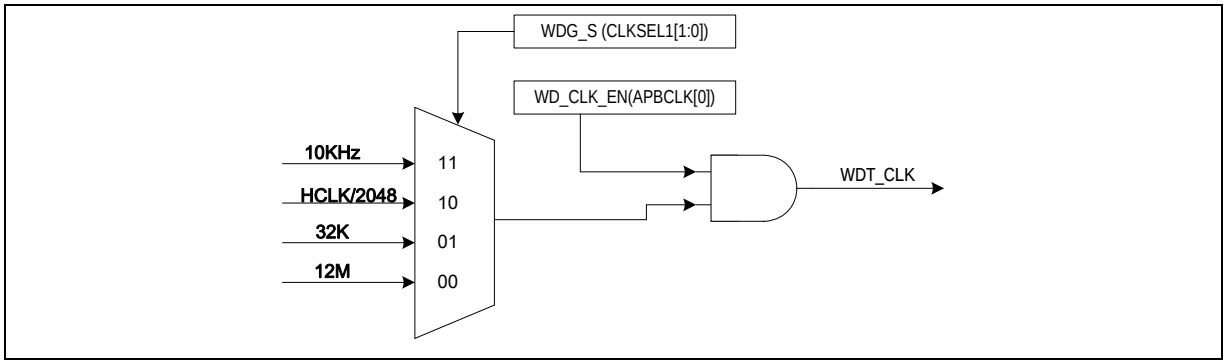



图 5-52 看门狗定时器时钟控制

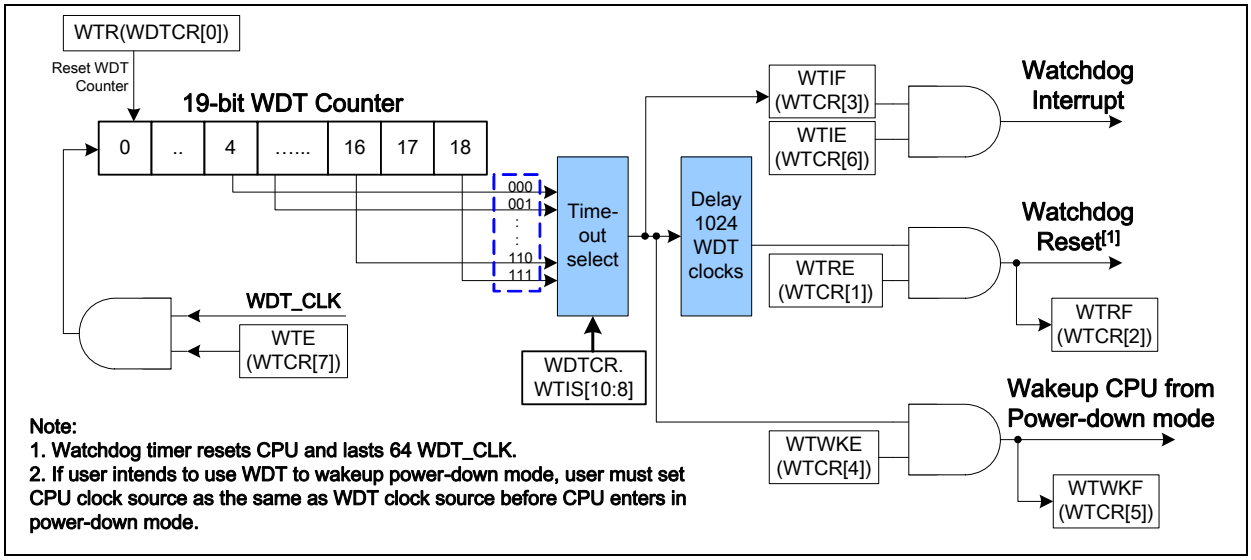



图 5-53 看门狗定时框图

### 5.11.1 看门狗定时器控制寄存器图

**R: 只读, W: 只写, R/W: 可读写**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **WDT_BA = 0x4000_4000** | | | | |
| **WTCR** | WDT_BA+00 | R/W | 看门狗定时器控制寄存器 | 0x0000_0700 |

**看门狗定时器控制寄存器(WTCR)**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **WTCR** | WDT_BA+000 | R/W | 看门狗定时器控制寄存器 | 0x0000_0700 |

注:

**该寄存器所有位都写保护. 要编程时，需要开锁时序，依次向寄存器RegLockAddr 写入“59h”, “16h”, 与 “88h”，RegLockAddr 的地址为GCR_BA + 0x100。**

| 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 | | | | | WTIS | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **WTE** | **WTIE** | **WTWKF** | **WTWKE** | **WTIF** | **WTRF** | **WTRE** | **WTR** |

| Bits | 描述 | |
|---|---|---|
| [31:11] | 保留 | 保留 |

<table>
<tr><td>[10:8]</td><td>WTIS</td><td>看门狗定时器间隔选择<br>选择看门狗定时器的定时溢出间隔<br>
<table>
<tr><th>WTIS</th><th>Interrupt Timeout</th><th>Watchdog Reset Timeout</th><th>WTR Timeout Interval (WDT_CLK=12MHz)</th></tr>
<tr><td>000</td><td>$2^4$ WDT_CLK</td><td>($2^4$ + 1008) WDT_CLK</td><td>69.33 us.</td></tr>
<tr><td>001</td><td>$2^6$ WDT_CLK</td><td>($2^6$ + 960) WDT_CLK</td><td>72.53 us.</td></tr>
<tr><td>010</td><td>$2^8$ WDT_CLK</td><td>($2^8$ + 768) WDT_CLK</td><td>85.33 us.</td></tr>
<tr><td>011</td><td>$2^{10}$ WDT_CLK</td><td>($2^{10}$ + 1024) WDT_CLK</td><td>170.67 us</td></tr>
<tr><td>100</td><td>$2^{12}$ WDT_CLK</td><td>($2^{12}$ + 1024) WDT_CLK</td><td>426.67 us</td></tr>
<tr><td>101</td><td>$2^{14}$ WDT_CLK</td><td>($2^{14}$ + 1024) WDT_CLK</td><td>1.45 ms</td></tr>
<tr><td>110</td><td>$2^{16}$ WDT_CLK</td><td>($2^{16}$ + 1024) WDT_CLK</td><td>5.55 ms</td></tr>
<tr><td>111</td><td>$2^{18}$ WDT_CLK</td><td>($2^{18}$ + 1024) WDT_CLK</td><td>21.93 ms</td></tr>
</table></td></tr>
<tr><td>[7]</td><td>WTE</td><td>看门狗定时器使能<br>0 = 禁止看门狗定时器功能(该动作重置内部计数器T)<br>1 = 使能看门狗定时器</td></tr>
<tr><td>[6]</td><td>WTIE</td><td>看门狗定时器中断使能<br>0 = 禁止看门狗定时器中断<br>1 =使能看门狗定时器中断</td></tr>
<tr><td>[5]</td><td>WTWKF</td><td>看门狗定时器唤醒标志<br>如果看门狗定时器引起CPU从掉电模式下唤醒，该位将被置高。必须由软件向该位写1清零<br>1 = CPU 由休眠或掉电模式被看门狗定时溢出唤醒.<br>0 = 看门狗定时器不能引起CPU唤醒.</td></tr>
<tr><td>[4]</td><td>WTWKE</td><td>看门狗定时器唤醒功能使能位<br>0 = 使能唤醒功能.<br>1 = 禁止唤醒功能.</td></tr>
<tr><td>[3]</td><td>WTIF</td><td>看门狗定时器中断标志<br>如果看门狗定时器中断使能，该位表示看门狗定时器中断发生，如果没有使能看门狗定时器中断，该位表示定时溢出周期已过.<br>0= 不发生看门狗定时器中断<br>1= 发生看门狗定时器中断<br>注: 1. 写1到该位，清零</td></tr>
</table>

| | | |
|---|---|---|
| [2] | **WTRF** | **看门狗定时器复位标志**<br>当看门狗定时器溢出引发复位，该位被置位，通过读取该位可以确认复位是否由看门狗引起。该位手动清除，如果**WTRE**禁止，看门狗定时器溢出对该位影响。<br>0 = 复位不是由看门狗定时器产生。<br>1 = 看门狗定时器引发复位<br>**注**: 该未只读，由软件清零. |
| [1] | **WTRE** | **看门狗定时器复位使能**<br>设定该位使能看门狗定时器复位功能。<br>0 = 禁止看门狗定时器复位功能<br>1 = 使能看门狗定时器复位功能 |
| [0] | **CLRWTR** | **看门狗定时器重置**<br>该位用于看门狗定时器溢出前，重置定时期值。如果**WTIE**设定使能，对WTR置位会引发看门狗定时器中断。如果**WTRE**使能，该位置位，在64个WDT时钟周期后，引发看门狗定时器复位。改为硬件清除。<br>0 = 无动作<br>1 = 重置看门狗定时器<br>**注**: 该位由硬件清除。 |

## 5.12 UART 接口控制器

MCU提供3个UART通道，UART0支持高速，UART1~2支持普通速度，另外UART0与UART1支持流控制.

### 5.12.1 概述

通用异步收/发器(UART) 从同外设收到数据的时候执行串到并的转换，从CPU收到数据的时候执行并到串的转换. 该串口同时支持LIN主机模式和 IrDA SIR 功能. **有7 种类型的中断**, 它们是, 发送FIFO 空中断(Int_THRE), 接收极限到达中断(Int_RDA), 线状态中断 (overrun error 或者校验错误或者framing error或者break 中断) (Int_RLS) , 超时中断(Int_Tout), MODEM 状态中断(Int_Modem) , 唤醒状态中断(Int_WakeUp)和LIN 接收停止侦测中断.

UART0 接口控制器一个内嵌64-byte 发送FIFO (TX_FIFO) 和 64-byte 接收 FIFO (RX_FIFO) 来降低CPU的中断数量；UART1~2内嵌 16-byte 发送FIFO (TX_FIFO) 和16-byte (每个字节加3比特的错误数据) 接收FIFO (RX_FIFO) 来降低CPU的中断数量. 在操作过程中CPU可以随时读UART的状态. 报告的状态信息包括已经被UART执行的传输操作的类型和条件, 也包括4 种错误条件(parity error, overrun error, framing error, 或break中断). UART包括一个可编程的波特率发生器，它可以将输入晶振除以一个除数来得到收发器需要的时钟. 波特率公式是 **Baud Rate = UART_CLK / M * [BRD + 2].** 其中M和BRD在波特率分频寄存器UA_BAUD中定义，见表 5-7 .

UART0与UART1 控制器支持自动流程控制功能用于 2 种low-level 信号, /CTS (clear-to-send)和 /RTS (request-to-send)，当使能自动流程控制时，据在UART 和外部驱动器(ex: Modem)之间的数据传输. UART 将禁止接收数据直到 UART asserts /RTS 外部驱动器. 当 Rx FIFO 的值和 UA_FCR[RTS_Tri_Lev] 相等, /RTS deasserted. UART 向外发送数据当UART 控制器侦测到 /CTS 从外部驱动器 asserted. 如果 /CTS 未被察觉, UART 将不向外发送数据.

UART 控制器提供 串行 IrDA (SIR, 串行红外) 功能 (用户需置位 UA_FUN_SEL[IrDA_EN] 使能 IrDA 功能). SIR 定义短程红外异步串行传输模式 1 开始位, 8 数据位, 和1 停止位. 最大数据速率 为 115.2 Kbps (半双工). IrDA SIR 包括 IrDA SIR 编码/解码协议.仅具有IrDA SIR 半双工协议. 不能同时传输和接收数据. IrDA SIR 物理层规定至少10ms 输出延时在传输和接收之间. **该特性由软件执行.**

UART 控制器具有**LIN(局域网) 功能.** LIN 模式提供设定LIN_EN bit 位 UA_FUN_SEL 寄存器. 在LIN 模式, 1 开始位 8 数据位和1 停止位相一致，同标准LIN 相同

表 5-7 UART 波特率公式

| DIV_X_EN | DIV_X_ONE | Divider X | BRD | 波特率公式 |
|---|---|---|---|---|
| Disable | 0 | B | A | UART_CLK / [16 * (A+2)] |
| Enable | 0 | B | A | UART_CLK / [(B+1) * (A+2)] , B must >= 8 |
| Enable | 1 | B | A | UART_CLK / (A+2), A must >=3 |

表 5-8 UART 波特率设置表

| System clock = 22.1184Mhz |
|---|

| Baud rate | Mode0 | Mode1 | Mode2 |
|---|---|---|---|
| 921600 | x | A=0,B=11 | A=22 |
| 460800 | A=1 | A=1,B=15<br>A=2,B=11 | A=46 |
| 230400 | A=4 | A=4,B=15<br>A=6,B=11 | A=94 |
| 115200 | A=10 | A=22,B=15<br>A=30,B=11 | A=190 |
| 57600 | A=22 | A=10,B=15<br>A=14,B=11 | A=382 |
| 38400 | A=34 | A=62,B=8<br>A=46,B=11<br>A=34,B=15 | A=574 |
| 19200 | A=70 | A=126,B=8<br>A=94,B=11<br>A=70,B=15 | A=1150 |
| 9600 | A=142 | A=254,B=8<br>A=190,B=11<br>A=142,B=15 | A=2302 |
| 4800 | A=286 | A=510,B=8<br>A=382,B=11<br>A=286,B=15 | A=4606 |

### 5.12.2 UART控制器特性

- UART控制支持3个通道， UART0, UART1 与 UART2.
- UART0/UART1/UART2 支持 64/16/16 字节FIFO，用来装载收/发的数据.
- UART0 与UART1支持自动流控制/流控制功能(/CTS, /RTS).
- 可编程的波特率发生器.
- 完全可编程的串口特性:
  - 5-, 6-, 7-, or 8-位特性.
  - 偶数, 奇数, 或者无校验产生和侦测.
  - 1-, 1&1/2, or 2-停止比特位.
  - 波特率发生器.
  - 开始位侦测.
- 支持 IrDA SIR 功能.
- 支持 LIN 主模式.

### 5.12.3 框图

UART时钟控制与框图如下.

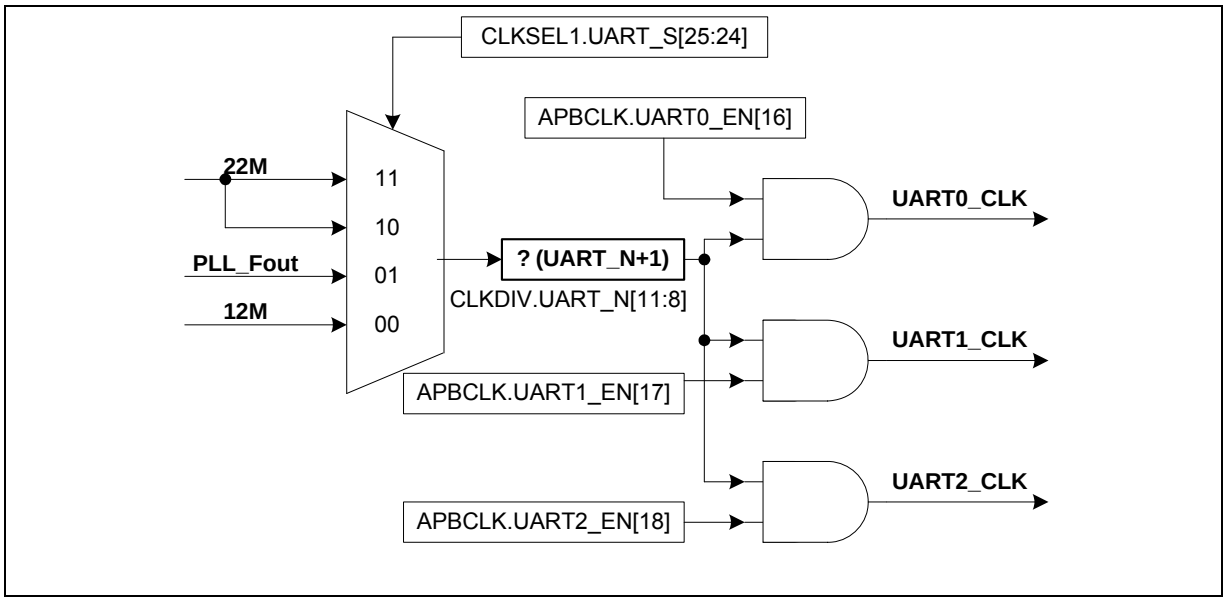



图 5-54 UART 时钟控制图

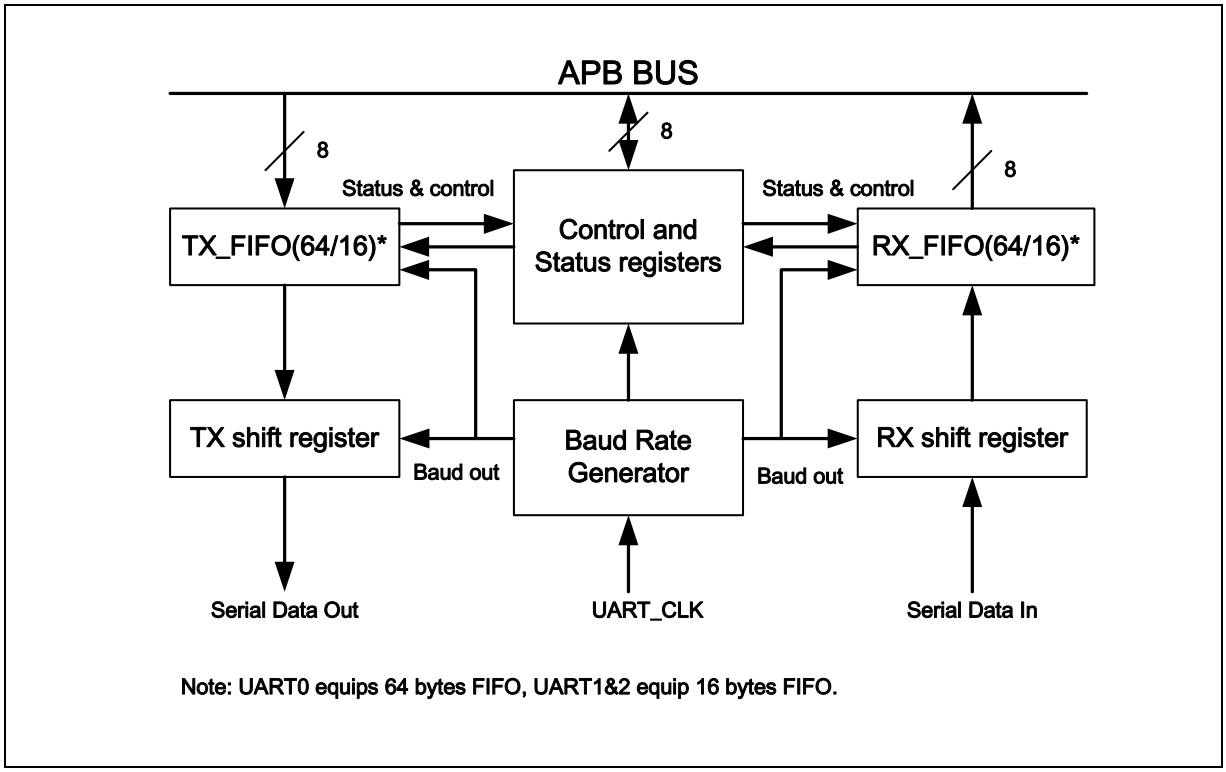



图 5-55 UART 框图

**TX_FIFO**
发送用一个64/16个字节的FIFO做缓存来降低CPU的中断数量..

**RX_FIFO**
接收用一个64/16个字节(每个字节加3个比特的错误比特)的FIFO做缓存来降低CPU的中断数量.

**TX 移位寄存器**
移动发送的数据串行输出

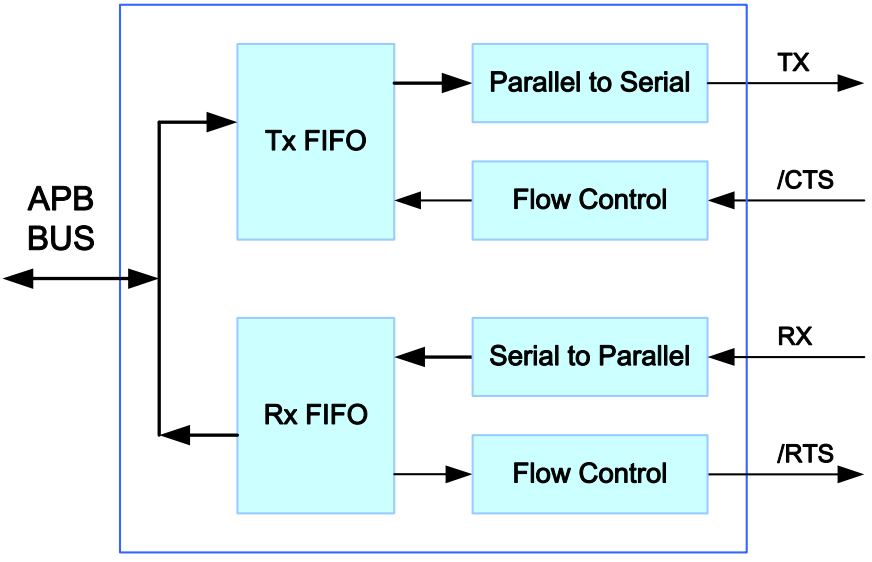
**RX 移位寄存器**
串行移动接收到的数据

**Modem 控制寄存器**
寄存器控制给MODEM 或者数据集或者一个MODEM模拟器的接口.

**波特率发生器**
将外部时钟除以一个除数来产生期望的内部时钟，参考表 5-7

**控制和状态寄存器**
此寄存器设定, 包括 FIFO 控制寄存器 (FCR), FIFO 状态寄存器 (FSR), 和排列状态寄存器 (LCR) 应用于传输和接收. 时间溢出控制寄存器 (TOR) 应用于识别时间溢出中断控制. 该寄存器包括中断控制使能寄存器 (IER) 和中断状态寄存器 (ISR) 来使能或者禁止中断响应并且识别发生的中断. 有六种中断: FIFO为空中断（INT_THRE），接收开始中断（INT_RDA），line 状态中断 (overrun error or 校验 error or framing error or break interrupt)（INT_RLS）, 定时溢出中断（INT_Tout），MODEM/唤醒状态中断（INT_Modem）和缓冲错误中断（INT_Buf_Err）

下图为自动流控制框图.




注: 仅支持 UART0 及 UART1

图 5-56 自动流控制框图

### 5.12.4 IrDA 模式

UART 支持 IrDA SIR (Serial Infrared) 传输编码 和接收解码, IrDA 模式通过设定 **IrDA_EN** 位 **UA_FUN_SEL** 寄存器.

IrDA 模式下, UA_BAUD[DIV_X_EN] 寄存器需禁止.

波特率 = Clock / (16 * BRD), BRD 为波特率分频 UA_BAUD 寄存器.

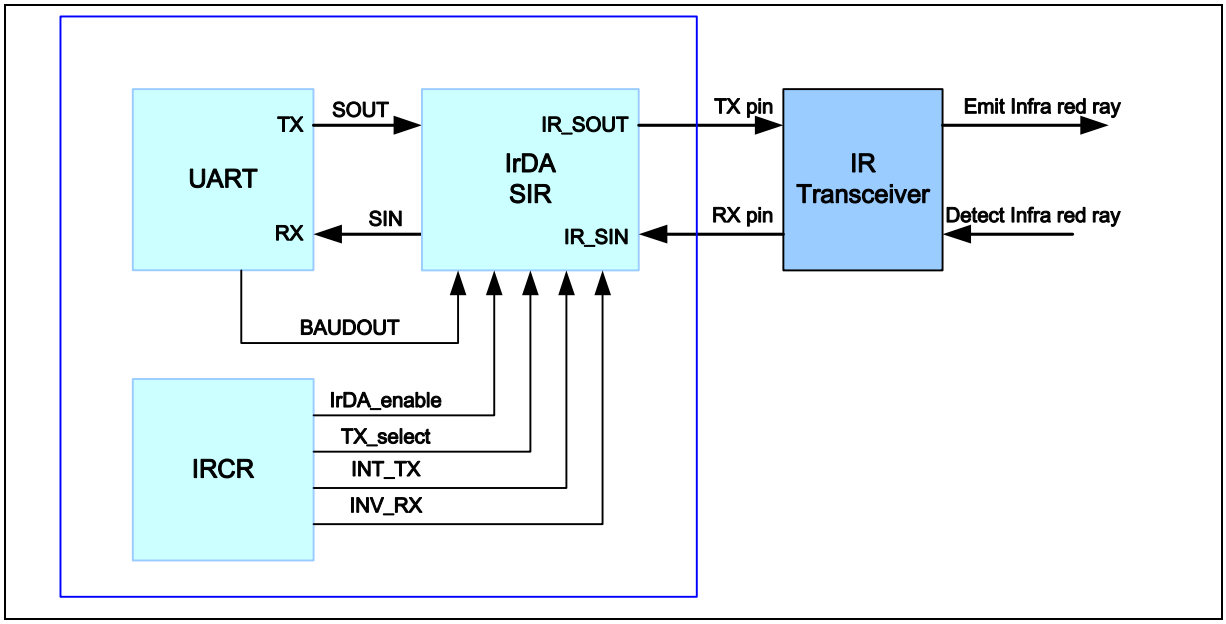



图 5-57 IrDA 框图

#### *5.12.4.1 IrDA SIR 传输编码*

IrDA SIR 传输编码调节 Non-Return-to Zero (NRZ) 传输位从 UART 输出. IrDA SIR 物理层指定使用 Return-to-Zero, 反向 (RZI) 调制配置表现在 0 作为下列脉冲. 被调整的脉冲输出 脉冲到外部输出驱动器和红外发射二极管.

在正常模式下, 传输脉冲的宽度为 3/16 波特率周期.

#### *5.12.4.2 IrDA SIR 接收解码*

IrDA SIR 接收解码 解调 return-to-zero 位 从输入探测器和输出 NRZ 连续位到 UART 作为数据输入. 解码器在空闲模式一般输入 高位. (因此, IRCR bit 5 默认设定为高)

当解码器输入为低时，开始位将被察觉

#### *5.12.4.3 IrDA SIR 运作*

IrDA SIR 编码/解码 提供 UART 数据流和半双工串行 SIR 将转换. IrDA编码/解码 波形图如下:

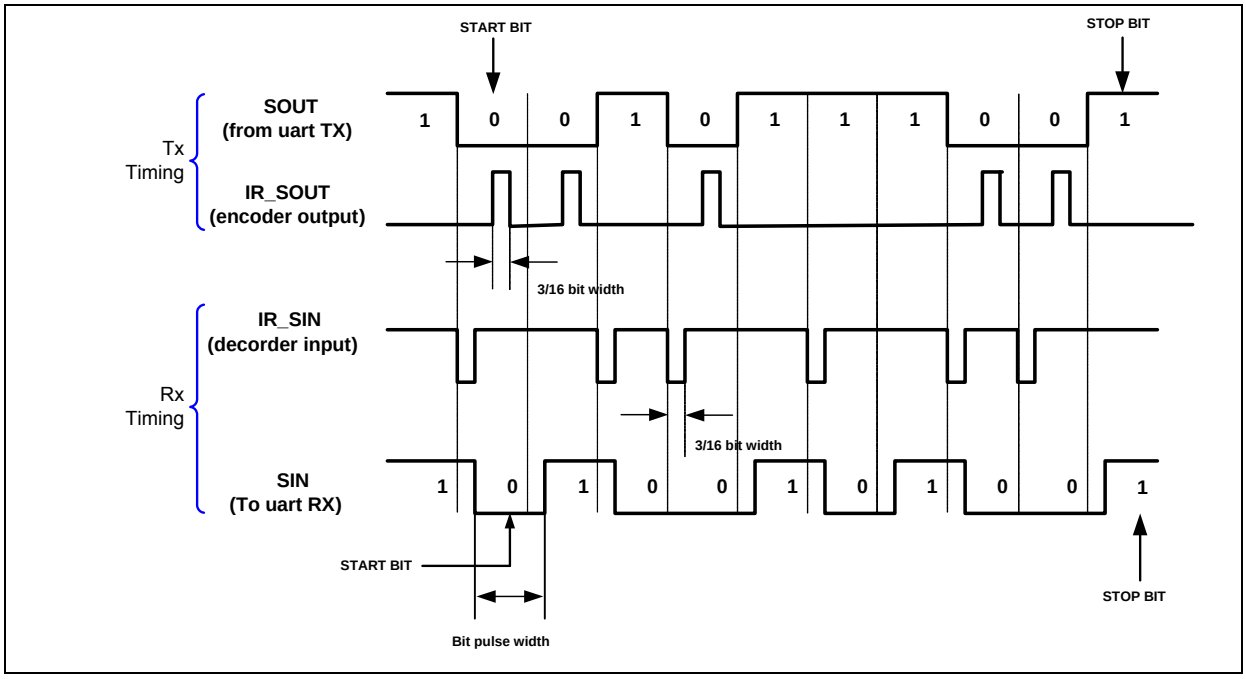



图 5-58 IrDA Tx/Rx 时序图

### 5.12.5 LIN (Local Interconnection Network) 模式

UART 支持 LIN 功能, LIN 模式通过选择设定**LIN_EN** 位 **UA_FUN_SEL** 寄存器. 在 LIN 模式下, 每字节开始位为 0 (显性), 下列8 位数据 (LSB is first) 结束位为 1 (隐性) 同标准 LIN 相同.

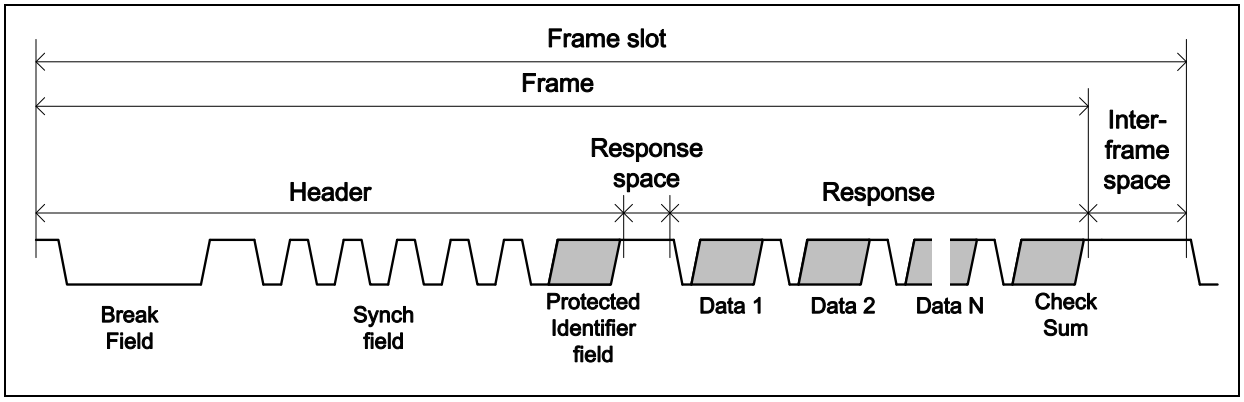



图 5-59 LIN 帧结构

LIN 总线传输 (Tx) 编程流程

1. 设定 **LIN_EN** 位 **UA_FUN_SEL** 寄存器 使能 LIN 总线模式.
2. 装载 **UA_LIN_BKFL** （**UA_LIN_BCNT** ）选择 break 区域长度. (break区域长度 为UA_LIN_BKF + 2).
3. 写 0x55 到 **UA_THR** 请求同步区域传输.
4. 写保护标志符值到 **UA_THR**，请求现场传输标识符.
5. 设定 **LIN_Tx_En** 位 **UA_LIN_BCNT**寄存器 开始传输 (当 break 现场操作完成, LIN_Tx_EN 将自动清除).
6. 当最后字节 STOP 位 THR 发送到总线, 硬件将 置位**TE_FLAG UA_FSR** 寄存器.
7. 写 N 字节数据 和检验 到 **UA_THR** ， 重复步骤5和6 传输数据.

LIN总线接收 (Rx) 编程流程

1. 设定 **LIN_EN** bit in **UA_FUN_SEL**寄存器使能 LIN 总线模式.
2. 设定 **LIN_Rx_En** bit in **UA_LIN_BCNT** 寄存器使能 LIN Rx 模式.
3. 等待 **LIN_Rx_Break_IF** in **UA_ISR** 标志位 检查 Rx 有无 Break 区域接收.
4. 等待 **RDA_IF** in **UA_ISR** 标志位 和读 **UART_RBR**寄存器.

### 5.12.6 UART接口控制寄存器列表

**R: 只读, W: 只写, R/W: 可读写**, **C**: 仅在值为0时可写

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **UART 基地址：**<br>**通道0 : UART0_BA (High Speed) = 0x4005_0000**<br>**通道1 : UART1_BA (Normal Speed)= 0x4015_0000**<br>**通道2 : UART2_BA (Normal Speed)= 0x4015_4000** | | | | |
| **UA_RBR** | UART0_BA + 0x00 | R | UART0接收数据缓存寄存器. | Undefined |
| | UART1_BA + 0x00 | R | UART1接收数据缓存寄存器. | Undefined |
| | UART2_BA + 0x00 | R | UART2接收数据缓存寄存器 | Undefined |
| **UA_THR** | UART0_BA + 0x00 | W | UART0发送保持寄存器. | Undefined |
| | UART1_BA + 0x00 | W | UART1发送保持寄存器. | Undefined |
| | UART2_BA + 0x00 | W | UART2发送保持寄存器. | Undefined |
| **UA_IER** | UART0_BA + 0x04 | R/W | UART0中断使能寄存器. | 0x0000_0000 |
| | UART1_BA + 0x04 | R/W | UART1中断使能寄存器. | 0x0000_0000 |
| | UART2_BA + 0x04 | R/W | UART2中断使能寄存器. | 0x0000_0000 |
| **UA_FCR** | UART0_BA + 0x08 | R/W | UART0 FIFO 控制寄存器. | 0x0000_0000 |
| | UART1_BA + 0x08 | R/W | UART1 FIFO 控制寄存器. | 0x0000_0000 |
| | UART2_BA + 0x08 | R/W | UART2 FIFO 控制寄存器. | 0x0000_0000 |
| **UA_LCR** | UART0_BA + 0x0C | R/W | UART0 Line 控制寄存器. | 0x0000_0000 |
| | UART1_BA + 0x0C | R/W | UART1 Line 控制寄存器. | 0x0000_0000 |
| | UART2_BA + 0x0C | R/W | UART2 Line 控制寄存器. | 0x0000_0000 |
| **UA_MCR** | UART0_BA + 0x10 | R/W | UART0 Modem 控制寄存器. | 0x0000_0000 |
| | UART1_BA + 0x10 | R/W | UART1 Modem 控制寄存器. | 0x0000_0000 |
| | UART2_BA + 0x10 | R/W | 保留. | 0x0000_0000 |
| **UA_MSR** | UART0_BA + 0x14 | R/W | UART0 Modem 状态寄存器. | 0x0000_0000 |
| | UART1_BA + 0x14 | R/W | UART1 Modem 状态寄存器. | 0x0000_0000 |
| | UART2_BA + 0x14 | R/W | 保留. | 0x0000_0000 |
| **UA_FSR** | UART0_BA + 0x18 | R/W | UART0 FIFO 状态寄存器. | 0x1040_4000 |

| | | | | |
|---|---|---|---|---|
| | UART1_BA + 0x18 | R/W | UART1 FIFO 状态寄存器. | 0x1040_4000 |
| | UART2_BA + 0x18 | R/W | UART2 FIFO 状态寄存器. | 0x1040_4000 |
| **UA_ISR** | UART0_BA + 0x1C | R/W | UART0 Interrupt 状态寄存器. | 0x0000_0002 |
| | UART1_BA + 0x1C | R/W | UART1 Interrupt 状态寄存器. | 0x0000_0002 |
| | UART2_BA + 0x1C | R/W | UART2 Interrupt状态寄存器 | 0x0000_0002 |
| **UA_TOR** | UART0_BA + 0x20 | R/W | UART0 定时溢出寄存器 | 0x0000_0000 |
| | UART1_BA + 0x20 | R/W | UART1定时溢出寄存器 | 0x0000_0000 |
| | UART2_BA + 0x20 | R/W | UART2定时溢出寄存器 | 0x0000_0000 |
| **UA_BAUD** | UART0_BA + 0x24 | R/W | UART0 波特率分频寄存器 | 0x0F00_0000 |
| | UART1_BA + 0x24 | R/W | UART1 波特率分频寄存器 | 0x0F00_0000 |
| | UART2_BA + 0x24 | R/W | UART2 波特率分频寄存器 | 0x0F00_0000 |
| **UA_IRCR** | UART0_BA + 0x28 | R/W | UART0 IrDA 控制寄存器. | 0x0000_0040 |
| | UART1_BA + 0x28 | R/W | UART1 IrDA 控制寄存器. | 0x0000_0040 |
| | UART2_BA + 0x28 | R/W | UART2 IrDA控制寄存器. | 0x0000_0040 |
| **UA_LIN_BCNT** | UART0_BA + 0x2C | R/W | UART0 LIN Break失败计数寄存器 | 0x0000_0000 |
| | UART1_BA + 0x2C | R/W | UART1 LIN Break失败计数寄存器 | 0x0000_0000 |
| | UART2_BA + 0x2C | R/W | UART2 LIN Break 失败计数寄存器 | 0x0000_0000 |
| **UA_FUN_SEL** | UART0_BA + 0x30 | R/W | UART0 功能选择寄存器. | 0x0000_0000 |
| | UART1_BA + 0x30 | R/W | UART1功能选择寄存器 | 0x0000_0000 |
| | UART2_BA + 0x30 | R/W | UART2功能选择寄存器 | 0x0000_0000 |

### 5.12.7 UART 接口控制寄存器描述

**接收缓冲寄存器(UA_RBR)**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| UA_RBR | UART0_BA + 0x00 | R | UART0接收缓冲寄存器 | Undefined |
| | UART1_BA + 0x00 | R | UART1接收缓冲寄存器 | Undefined |
| | UART2_BA + 0x00 | R | UART2接收缓冲寄存器 | Undefined |

| 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 |
| 8-bit 接收数据 | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [7:0] | **8-bit 接收数据** | 接收缓冲寄存器<br>通过读此寄存器, UART 将返回一组从 Rx pin接收到的 8-位数据 (LSB first). |

发送保持寄存器(UA_THR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| UA_THR | UART0_BA + 0x00 | W | UART0发送保持寄存器 | Undefined |
| | UART1_BA + 0x00 | W | UART1发送保持寄存器 | Undefined |
| | UART2_BA + 0x00 | W | UART2发送保持寄存器 | Undefined |

| 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 |
| 8-bit 发送数据 | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [7:0] | **8-bit 发送数据** | 发送保持寄存器<br>通过写该寄存器, UART 将通过Tx pin (LSB first) 发送 8-位数据. |

### 中断使能寄存器 (UA_IER)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| UA_IER | UART0_BA + 0x04 | R/W | UART0中断使能寄存器 | 0x0000_0000 |
| | UART1_BA + 0x04 | R/W | UART1中断使能寄存器 | 0x0000_0000 |
| | UART2_BA + 0x04 | R/W | UART2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 |
| DMA_Rx_EN | DMA_Tx_EN | Auto_CTS_EN | Auto_RTS_EN | Time_Out_EN | 保留 | | LIN_RX_BRK_IEN |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | Wake_EN | BUF_ERR_IEN | RTO_IEN | Modem_IEN | RLS_IEN | THRE_IEN | RDA_IEN |

| Bits | 描述 | |
|---|---|---|
| [31:16] | 保留 | 保留 |
| [15] | **DMA_Rx_EN** | **Rx DMA 使能**<br>该位使能或禁止Rx DMA 服务.<br>0 = 使能 Rx DMA.<br>1 = 禁止 Rx DMA. |
| [14] | **DMA_Tx_EN** | **Tx DMA 使能**<br>该位使能或禁止 Tx DMA 服务.<br>0 = 使能 Tx DMA.<br>1 = 禁止 Tx DMA. |
| [13] | **Auto_CTS_EN** | **CTS 自动流控制使能(not available in UART2 channel)**<br>1 = 使能 CTS **自动流控制**.<br>0 = 禁止 CTS **自动流控制**.<br>当 CTS 自动溢出控制使能, UART将向外部驱动器发送数据 当CTS输入允许(UART 将不发送数据 只到CTS 被证实. |

| | | |
|---|---|---|
| [12] | **Auto_RTS_EN** | **RTS 自动流控制使能**<br>1 = 使能 RTS 自动流控制.<br>0 = 禁止 RTS自动流控制.<br>当 RTS 自动流使能, Rx FIFO 的数值和 UA_FCR[RTS_Tri_Lev]相等, UART 将发出 RTS 信号 |
| [11] | **Time_Out_EN** | **Time-Out 计数器使能**<br>1 = 使能 Time-out 计数器.<br>0 = 禁止 Time-out 计数器.r. |
| [10:9] | 保留 | 保留 |
| [8] | **LIN_RX_BRK_IEN** | **LIN RX Break Field Detected 中断使能**<br>0 = 禁止Lin bus Rx break filed 中断.<br>1 = 使能Lin bus Rx break filed 中断. |
| [7] | 保留 | 保留 |
| [6] | **Wake_EN** | **唤醒 CPU 功能使能(not available in UART2 channel)**<br>0 = 禁止 UART 唤醒 CPU 功能<br>1 = 使能唤醒功能, 当系统在 deep sleep 模式下, 外部 /CTS 的改变将 CPU 从 deep sleep 模式下唤醒. |
| [5] | **BUF_ERR_IEN** | **Buffer Error 中断使能**<br>0 = 禁止INT_Buf_err中断<br>1 = 使能INT_Buf_err中断 |
| [4] | **RTO_IEN** | **Rx Time out 中断使能**<br>0 = 禁止INT_tout中断<br>1 = 使能INT_tout 中断 |
| [3] | **Modem_IEN** | **Modem 中断状态使能(not available in UART2 channel)**<br>0 = 禁止off INT_MOS中断<br>1 = 使能INT_MOS中断 |
| [2] | **RLS_IEN** | **线上接收中断状态使能**<br>0 = 禁止off INT_RLS中断<br>1 = 使能INT_RLS中断 |

| | | |
|---|---|---|
| [1] | **THRE_IEN** | **发送保持寄存器空中断使能**<br>0 = 禁止 INT_THRE中断<br>1 = 使能INT_THRE中断 |
| [0] | **RDA_IEN** | **可接收数据中断使能** .<br>0 = 禁止INT_RDA中断<br>1 = 使能INT_RDA中断 |

### FIFO控制寄存器(UA_FCR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| UA_FCR | UART0_BA + 0x08 | R/W | UART0 FIFO控制寄存器 | 0x0000_0000 |
| | UART1_BA + 0x08 | R/W | UART1 FIFO控制寄存器. | 0x0000_0000 |
| | UART2_BA + 0x08 | R/W | UART2 FIFO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 | | | | RTS_Tri_Lev | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| 保留 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RFITL | | | | 保留 | TFR | RFR | 保留 |

| Bits | 描述 | |
|---|---|---|
| [31:20] | 保留 | 保留 |
| [19:16] | RTS_Tri_Lev | RTS触发自动流程控制使用(not available in UART2 channel)<br>RTS_Tri_Lev / Trigger Level (Bytes):<br>0000: 01<br>0001: 04<br>0010: 08<br>0011: 14<br>0100: 30/14 (高速/常态)<br>0101: 46/14 (高速/常态)<br>0110: 62/14 (高速/常态)<br>others: 62/14 (高速/常态)<br>注: 该寄存器用于自动RTS流控制. |

<table>
<tr><td>[7:4]</td><td>RFITL</td><td>Rx FIFO中断 (INT_RDA) 触发级别<br>FIFO 接收字节数等于 RFITL 后RDA_IF 将被置位 (如果IER [RDA_IEN] 使能, 将产生中断).<br><table><tr><th>RFITL</th><th>INTR_RDA Trigger Level (Bytes)</th></tr><tr><td>0000</td><td>01</td></tr><tr><td>0001</td><td>04</td></tr><tr><td>0010</td><td>08</td></tr><tr><td>0011</td><td>14</td></tr><tr><td>0100</td><td>30/14 (高速/常态)</td></tr><tr><td>0101</td><td>46/14 (高速/常态)</td></tr><tr><td>0110</td><td>62/14 (高速/常态)</td></tr><tr><td>others</td><td>62/14 (高速/常态)</td></tr></table></td></tr>
<tr><td>[3]</td><td>保留</td><td>保留</td></tr>
<tr><td>[2]</td><td>Tx_RST</td><td>Tx 软件复位<br>当 Tx_RST 置位, FIFO 传输和Rx 内部状态将被清 0.<br>0 = 该位写 0 将无效.<br>1 =该位置位将复位 Tx 内部机器和指令状态.<br>注: 至少 3 UART时钟周期 自动清 0.</td></tr>
<tr><td>[1]</td><td>Rx_RST</td><td>Rx 软件复位<br>当 Rx_RST 置位, FIFO 传输和Rx 内部状态将被清 0.<br>0 = 该位写 0 将无效.<br>1 = 该位置位将复位 Rx 内部机器和指令状态.<br>注:至少 3 UART时钟周期 自动清 0.</td></tr>
<tr><td>[0]</td><td>保留</td><td>保留</td></tr>
</table>

### Line控制寄存器(UA_LCR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| UA_LCR | UART0_BA + 0x0C | R/W | UART0 Line控制寄存器 | 0x0000_0000 |
| | UART1_BA + 0x0C | R/W | UART1 Line控制寄存器 | 0x0000_0000 |
| | UART2_BA + 0x0C | R/W | UART2 Line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 | |

| Bits | 描述 | |
|---|---|---|
| [31:7] | 保留 | 保留 |
| [6] | **BCB** | **钳制控制位**<br>该位置位, 串行数据输出 (Tx) 将被迫间隔发送数据 (logic 0). 该位仅作用于Tx 对传输逻辑不起作用. |
| [5] | **SPE** | **Stick 奇偶使能**<br>0 = 禁止 stick 奇偶使能<br>1 = 当 PBE , EPE 和 SPE 置位, 奇偶位传输 检测被清除. 当 PBE 和 SPE 置位 并且 EPE 清除, 奇偶位传输 检测有效. |
| [4] | **EPE** | **Even 奇偶使能**<br>0 = 奇数逻辑 1's 传输 检测数据和奇偶位.<br>1 = 偶数逻辑 1's 传输 检测数据和奇偶位.<br>该位仅当 bit 3 (parity bit enable) 位 置位有效. |

| | | |
|---|---|---|
| [3] | **PBE** | **奇偶使能位**<br>0 = 当传输时 奇偶位 无(transmit data) 产生或检测 (receive data).<br>1 =串行数据"last data word bit" 和"stop bit"奇偶位产生或检测. |
| [2] | **NSB** | **“STOP bit” 数目**<br>0= 传递数据时 1“ STOP bit” 产生<br>1=传递数据时 1.5 “ STOP bit”产生( 5-bit word 长度被选择);<br>2 “STOP bit” 产生 6-, 7- 和 8-位 word 长度被选择. |
| [1:0] | **WLS** | **字长度选择**<br>**WLS[1:0]** / **Character length**<br>00 / 5 bits<br>01 / 6 bits<br>10 / 7 bits<br>11 / 8 bits |

### MODEM控制寄存器(UA_MCR) (not available in UART2 channel)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| UA_MCR | UART0_BA + 0x10 | R/W | UART0 Modem控制寄存器 | 0x0000_0000 |
| | UART1_BA + 0x10 | R/W | UART1 Modem控制寄存器 | 0x0000_0000 |
| | UART2_BA + 0x10 | R/W | 保留. | 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 | | RTS_St | 保留 | | | Lev_RTS | 保留 |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | | 保留 | 保留 | | RTS | 保留 |

| Bits | 描述 | |
|---|---|---|
| [31:14] | 保留 | 保留 |
| [13] | **RTS_St** | **RTS Pin 状态(not available in UART2 channel)**<br>该位表示 RTS 引脚状态 |
| [12:10] | 保留 | 保留 |
| [9] | **Lev_RTS** | **RTS 触发级别(not available in UART2 channel)**<br>该位改变 RTS 触发级别.<br>0= 低级触发<br>1= 高级触发 |
| [8:5] | 保留 | 保留 |
| [4] | 保留 | 保留 |
| [3:2] | 保留 | 保留 |

| | | |
|---|---|---|
| [1] | **RTS** | **RTS (Request-To-Send) 信号(not available in UART2 channel)**<br>0: 使能 RTS 管脚为 1 (如果 **Lev_RTS** 设定低级触发).<br>1: 使能 RTS 管脚为 0 (如果 **Lev_RTS** 设定低级触发).<br>0: 使能 RTS 管脚为 0 (如果 **Lev_RTS** 设定高级触发).<br>1: 使能 RTS 管脚为 1 (如果 **Lev_RTS** 设定高级触发). |

**Modem状态寄存器(UA_MSR) (not available in UART2 channel)**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| UA_MSR | UART0_BA + 0x14 | R/W | UART0 Modem状态寄存器 | 0x0000_0000 |
| | UART1_BA + 0x14 | R/W | UART1 Modem状态寄存器 | 0x0000_0000 |
| | UART2_BA + 0x14 | R/W | 保留. | 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 | | | | | | | Lev_CTS |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | | CTS_St | 保留 | | | DCTSF |

| Bits | 描述 | |
|---|---|---|
| [31:9] | 保留 | 保留 |
| [8] | **Lev_CTS** | **CTS 触发级别(not available in UART2 channel)**<br>该位可改变 CTS 触发级别.<br>0=低级触发<br>1=高级触发 |
| [7:5] | 保留 | 保留 |

| | | |
|---|---|---|
| [4] | **CTS_St** | **CTS Pin 状况(not available in UART2 channel)**<br>该位表示 CTS 管脚状态 |
| [3:1] | 保留 | 保留 |
| [0] | **DCTSF** | **侦测 CTS 状态改变标志位(not available in UART2 channel)**<br>只要 CTS 输入状态改变 该位置位, 可向 CPU 产生中断请求(使能 IER [Modem_IEN]).<br>**注:** 该位只读, 可写 '1' 清除. |

### FIFO状态寄存器(UA_FSR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| UA_FSR | UART0_BA + 0x18 | R/W | UART0 FIFO状态寄存器. | 0x1040_4000 |
| | UART1_BA + 0x18 | R/W | UART1 FIFO状态寄存器 | 0x1040_4000 |
| | UART2_BA + 0x18 | R/W | UART2 FIFO状态寄存器 | 0x1040_4000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| 保留 | | | TE_Flag | 保留 | | | Tx_Over_IF |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Tx_Full | Tx_Empty | Tx_Pointer | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| Rx_Full | Rx_Empty | Rx_Pointer | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | BIF | FEF | PEF | 保留 | | | Rx_Over_IF |

| Bits | 描述 | |
|---|---|---|
| [31:29] | 保留 | 保留 |
| [28] | **TE_Flag** | **传输清空标志位 (只读)**<br>当 Tx FIFO(UA_THR) 清空 STOP 位传输，该位由硬件自动置位.<br>当 Tx FIFO(UA_THR) 未清空 STOP 位未传输，该位由硬件自动清空.<br>注: 该位只读. |
| [27:25] | 保留 | 保留 |
| [24] | **Tx_Over_IF** | **Tx 溢出 Error 中断标志位 (只读)**<br>若 Tx FIFO(UA_THR) 充满I, 另外写 UA_THR 将引起 置1.<br>注: 该位只读, 可通过写'1' 清除该位. |
| [23] | **Tx_Full** | **全部 FIFO 传输 (只读)**<br>该位表示 Tx FIFO 是否充满.<br>当 Tx_Point 和 64/16(UART0/UART1)相等 该位置位, 反之由硬件清除. |

| | | |
|---|---|---|
| [22] | **Tx_Empty** | **发送FIFO 为空(只读)**<br>该位表示 Tx FIFO 是否充满.<br>当 Tx FIFO 传输到 Shift 寄存器, 硬件置位该位. 当写数据到THR (Tx FIFO not empty) 清除. |
| [21:16] | **Tx_Pointer** | **TX FIFO Pointer (只读)**<br>该位表示Tx FIFO 缓冲指示器. 当CPU 写 1位到 UA_THR, Tx_Pointer 增 1.当 Tx FIFO 传输 1 位到 Shift 寄存器, Tx_Pointer 减1. |
| [15] | **Rx_Full** | **接收 FIFO Full (只读)**<br>该位表示 Rx FIFO 是否充满.<br>当 Rx_Point 和 64/16(UART0/UART1)相等 该位置位, 反之由硬件清除 |
| [14] | **Rx_Empty** | **接收FIFO 为空(只读)**<br>该位表示 Rx FIFO是否充满.<br>当 Rx FIFO 最后字节 从CPU中读取, 硬件置位 该位. 当 UART 接收到新数据 该位清除. |
| [13:8] | **Rx_Pointer** | **Rx FIFO pointer (只读)**<br>该位表示 Rx FIFO 缓冲指示器. 当UART 从外部驱动器接收到 1 位数据, Rx_Pointer增 1. 当 Rx FIFO 通过 CPU 读 1位数据, Rx_Pointer 减1. |
| [7] | 保留 | 保留 |
| [6] | **BIF** | **钳制中断标志位**<br>当"spacing state" (logic 0) 内数据的长度大于输入全字传输(即"start bit" + data bits + parity + stop bits的所有时间)的时间，该位置1。软件清除该位。 |
| [5] | **FEF** | **Framing Error 标志位**<br>若 received character 无有效" stop bit" 该位将置位, CPU 写 1 到 该位 复位. |
| [4] | **PEF** | **Parity Error 标志位**<br>若 received character 无有效"parity bit" 该位将置位, CPU 写 1 到 该位 复位. |
| [3:1] | 保留 | 保留 |
| [0] | **Rx_Over_IF** | **Rx overflow Error IF (只读)**<br>当 Rx FIFO 溢出，该位置位.<br>当接收到的数据大于 Rx FIFO(UA_RBR) 范围, 64/16 UART0/UART1，该位置位.<br>注: 该位只读，CPU 写 1 到 该位 复位. |

中断状态控制寄存器(UA_ISR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| UA_ISR | UART0_BA + 0x1C | R/W | UART0中断状态控制寄存器 | 0x0000_0002 |
| | UART1_BA + 0x1C | R/W | UART1中断状态控制寄存器 | 0x0000_0002 |
| | UART2_BA + 0x1C | R/W | UART2中断状态控制寄存器 | 0x0000_0002 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| HW_LIN_Rx_Break_INT | 保留 | HW_Buf_Err_INT | HW_Tout_INT | HW_Modem_INT | HW_RLS_INT | 保留 | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| HW_LIN_Rx_Break_IF | 保留 | HW_Buf_Err_IF | HW_Tout_IF | HW_Modem_IF | HW_RLS_IF | 保留 | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| LIN_Rx_Break_INT | 保留 | Buf_Err_INT | Tout_INT | Modem_INT | RLS_INT | THRE_INT | RDA_INT |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| LIN_Rx_Break_IF | 保留 | Buf_Err_IF | Tout_IF | Modem_IF | RLS_IF | THRE_IF | RDA_IF |

| Bits | 描述 | |
|---|---|---|
| [31] | **HW_LIN_Rx_Break_INT** | **DMA 模式, LIN Bus Rx Break Field 侦测指示中断控制器**<br>将 LIN_RX_BRK_IEN 和 LIN_Rx_Break_IF进行“与”(AND)，然后在该位输出 |
| [30] | 保留 | 保留 |
| [29] | **HW_Buf_Err_INT** | **DMA 模式, Buffer Error 侦测指示中断控制器 (INT_Buf_err)**<br>将BUF_ERR_IEN 和Buf_Err_IF进行“与”(AND)，然后在该位输出 |
| [28] | **HW_Tout_INT** | **DMA 模式, Time Out 状态指示中断控制器 (INT_Tout)**<br>将RTO_IEN 和Tout_IF进行“与”(AND)，然后在该位输出 |
| [27] | **HW_Modem_INT** | **DMA模式，MODEM 状态指示中断控制器 (INT_MOS). (not available in UART2 channel)**<br>将Modem_IEN 和Modem_IF进行“与”(AND)，然后在该位输出 |
| [26] | **HW_RLS_INT** | **DMA 模式, 接收 Line 中断状态指示中断控制器(INT_RLS).**<br>将RLS_IEN 和RLS_IF进行“与”(AND)，然后在该位输出 |

| | | |
|---|---|---|
| [25:24] | 保留 | 保留 |
| [23] | **HW_LIN_Rx_ Break_IF** | **DMA模式, LIN Bus Rx Break Field Detect 中断标志位**<br>当Rx 接收 LIN Break Field 该位置位. 若 IER[LIN_RX_BRK_IEN]置位 LIN RX 钳制中断产生.<br>**注:** 该位只读，写 1 以清除. |
| [22] | 保留 | 保留 |
| [21] | **HW_Buf_Err_IF** | **DMA 模式, Buffer Error Interrupt Flag (只读)**<br>当 Tx or Rx FIFO 溢出(Tx_Over_IF or Rx_Over_IF is set) 该位置位. 当Buf_Err_IF 置位, 传输可能不正常. 若 IER[Buf_Err_IEN] 使能, 缓冲 error 中断产生.<br>**注:** 当 Tx_Over_IF和 Rx_Over_IF 被清空，该位清空. |
| [20] | **HW_Tout_IF** | **DMA 模式, Time out Interrupt Flag (只读)**<br>当 Rx FIFO未清空 同时时间溢出计数器和TOIC 相等 该位置位. 若IER[Tout_IEN] 使能, 中断产生.<br>**注:** 该位只读，用户可读UA_RBR (Rx is in active)清空. |
| [19] | **HW_Modem_IF** | **DMA 模式, MODEM Interrupt Flag (只读) (not available in UART2 channel)**<br>当 CTS pin 状态(DCTSF=1)改变 该位置位.若IER[Modem_IEN]使能, Modem 中断产生.<br>**注:** 该位只读 ， 当DCTSF 清空 通过写1 DCTSF该位复位. |
| [18] | **HW_RLS_IF** | **DMA 模式, 接收 Line 状态标志位 (只读)**<br>当 Rx 接收数据有奇偶 error 时 该位置位, framing error 或 break error (至少3 位, BIF, FEF 和 PEF, 置位). 若IER[RLS_IEN] 使能, RLS 中断产生.<br>**注:** 该位只读， BIF, FEF 和 PEF 被清空 该位复位. |
| [17:16] | 保留 | 保留 |
| [15] | **LIN_Rx_Break_INT** | **LIN Bus Rx Break Field 中断状态指示中断控制器**<br>将LIN_RX_BRK_IEN 和LIN_Rx_Break_IF进行“与”(AND)，然后在该位输出 |
| [14] | 保留 | 保留 |
| [13] | **Buf_Err_INT** | **Buffer Error 中断状态指示中断控制器(INT_Buf_err)**<br>将BUF_ERR_IEN 和Buf_Err_IF进行“与”(AND)，然后在该位输出 |
| [12] | **Tout_INT** | **Time Out 中断状态指示中断控制器(INT_Tout)**<br>将RTO_IEN 和Tout_IF 进行“与”(AND)，然后在该位输出 |
| [11] | **Modem_INT** | **MODEM 中断状态指示中断控制器(INT_MOS). (not available in UART2 channel)**<br>将Modem_IEN 和Modem_IF进行“与”(AND)，然后在该位输出 |

| | | |
|---|---|---|
| [10] | **RLS_INT** | **线上数据接收中断状态指示中断控制器(INT_RLS).**<br>将RLS_IEN 和 RLS_IF进行“与”(AND)，然后在该位输出 |
| [9] | **THRE_INT** | **发送保持寄存器 清空中断状态指示中断控制器(INT_THRE).**<br>将THRE_IEN 和THRE_IF进行“与”(AND)，然后在该位输出 |
| [8] | **RDA_INT** | **接收数据可用中断状态指示中断控制器(INT_RDA).**<br>对 RDA_IEN 和RDA_IF进行“与”(AND)，然后在该位输出 |
| [7] | **LIN_Rx_Break_IF** | **LIN Bus Rx Break Field 侦测标志位**<br>Rx接收 LIN Break Field 该位置位. 若 IER[LIN_RX_BRK_IEN] 使能 LIN RX 钳制中断产生.<br>**注:** 该位只读，用户可写 1 清该位. |
| [6] | 保留 | 保留 |
| [5] | **Buf_Err_IF** | **Buffer Error 中断标志位 (只读)**<br>Rx FIFO 溢出 (Tx_Over_IF or Rx_Over_IF is set) 该位 置位.当 Buf_Err_IF置位, 传输可能不正常. 若IER[Buf_Err_IEN] 使能, buffer error 中断产生.<br>**注:** Tx_Over_IF 和 Rx_Over_IF 被清除 该位清除. |
| [4] | **Tout_IF** | **Time Out 中断标志位 (只读)**<br>当 Rx FIFO 未清空 同时时间溢出计数器和TOIC 相等 该位置位. 若 IER[Tout_IEN] 使能, Tout 中断产生.<br>**注:** 用户可读 UA_RBR (Rx is in active) 清该位. |
| [3] | **Modem_IF** | **MODEM 中断标志位 (只读) (not available in UART2 channel)**<br>当 CTS pin 状态该位(DCTSF=1) 该位置位. 若IER[Modem_IEN] 使能, Modem 中断产生.<br>**注:** 该位只读，当写 1 到DCTSF 该位清除. |
| [2] | **RLS_IF** | **Receive Line中断标志位 (只读).**<br>当 Rx r接收数据 parity error, framing error 或 break error (至少 3 位, BIF, FEF 和 PEF, 置位t)该位清空.若 IER[RLS_IEN]使能, RLS 中断产生.<br>**注:** 该位只读， 当 BIF, FEF 和PEF 清除 该位复位. |
| [1] | **THRE_IF** | **Transmit Holding 寄存器 清空中断标志位(只读).**<br>当Tx FIFO 最后数据传输到 Transmitter Shift 寄存器 该位置位. 若 IER[THRE_IEN] 置位, THRE 中断产生.<br>**注:** 该位只读 写数据到 THR (Tx FIFO not empty) 该位清空. |

| | | |
|---|---|---|
| [0] | **RDA_IF** | **Receive Data Available中断标志位 (只读).**<br>当 Rx FIFO 数据和 RFITL 相等， RDA_IF 将置位. 若 IER[RDA_IEN] 使能, RDA 中断产生.<br>**注:** 该位只读， Rx FIFO 跌落到 极限 级别 该位清除 (RFITL). |

表 5-9 UART 中断源和标志位图表 （DMA 模式）

| UART 中断源 | 中断使能位 | 中断指示 中断控制 | 中断标志位 | 标志位清除 |
|---|---|---|---|---|
| LIN RX Break Field Detected interrupt | LIN_RX_BRK_IEN | HW_LIN_Rx_Break_INT | **HW_LIN_Rx_Break_IF** | Write ‘1’ to LIN_Rx_Break_IF |
| Buffer Error Interrupt INT_Buf_Err | BUF_ERR_IEN | HW_Buf_Err_INT | **HW_Buf_Err_IF =** (Tx_Over_IF or Rx_Over_IF) | Write ‘1’ to Tx_Over_IF/ Rx_Over_IF |
| Rx Timeout Interrupt INT_Tout | RTO_IEN | HW_Tout_INT | **HW_Tout_IF** | Read UA_RBR |
| Modem Status Interrupt INT_Modem | Modem_IEN | HW_Modem_INT | **HW_Modem_IF =** (DCTSF) | Write ‘1’ to DCTSF |
| Receive Line Status Interrupt INT_RLS | RLS_IEN | HW_RLS_INT | **HW_RLS_IF =** (BIF or FEF or PEF) | Write ‘1’ to BIF/FEF/PEF |
| Transmit Holding Register Empty Interrupt INT_THRE | THRE_IEN | HW_THRE_INT | **HW_THRE_IF** | Write UA_THR |
| Receive Data Available Interrupt INT_RDA | RDA_IEN | HW_RDA_INT | **HW_RDA_IF** | Read UA_RBR |

表 5-10 UART 软件模式下 UART 中断源和标志位图表

| UART 中断源 | 中断使能位 | 中断指示 中断控制 | 中断标志位 | 标志位清除 |
|---|---|---|---|---|
| LIN RX Break Field Detected interrupt | LIN_RX_BRK_IEN | LIN_Rx_Break_INT | **LIN_Rx_Break_IF** | Write '1' to LIN_Rx_Break_IF |
| Buffer Error Interrupt INT_Buf_Err | BUF_ERR_IEN | Buf_Err_INT | **Buf_Err_IF =** (Tx_Over_IF or Rx_Over_IF) | Write '1' to Tx_Over_IF/ Rx_Over_IF |
| Rx Timeout Interrupt INT_Tout | RTO_IEN | Tout_INT | **Tout_IF** | Read UA_RBR |
| Modem Status Interrupt INT_Modem | Modem_IEN | Modem_INT | **Modem_IF =** (DCTSF) | Write '1' to DCTSF |
| Receive Line Status Interrupt INT_RLS | RLS_IEN | RLS_INT | **RLS_IF =** (BIF or FEF or PEF) | Write '1' to BIF/FEF/PEF |
| Transmit Holding Register Empty Interrupt INT_THRE | THRE_IEN | THRE_INT | **THRE_IF** | Write UA_THR |
| Receive Data Available Interrupt INT_RDA | RDA_IEN | RDA_INT | **RDA_IF** | Read UA_RBR |

### 时间溢出寄存器(UA_TOR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| UA_TOR | UART0_BA + 0x20 | R/W | UART0时间溢出寄存器 | 0x0000_0000 |
| | UART1_BA + 0x20 | R/W | UART1时间溢出寄存器 | 0x0000_0000 |
| | UART2_BA + 0x20 | R/W | UART2时间溢出寄存器 | 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 |
| 保留 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | TOIC | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [31:7] | 保留 | 保留 |
| [6:0] | **TOIC** | **时间溢出中断比较器**<br>time out 计数器复位和状态计数(the counting clock = baud rate) 当 RX FIFO 接收到新数据. 一旦 time out 计数器 (TOUT_CNT)和中断比较器 (TOIC) 相等, 接收 time out 中断产生 (INTR_TOUT) 若 UA_IER [RTO_IEN]使能.. |

### 波特率分频寄存器(UA_BAUD)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| UA_BAUD | UART0_BA + 0x24 | R/W | UART0波特率分频寄存器 | 0x0F00_0000 |
| | UART1_BA + 0x24 | R/W | UART1波特率分频寄存器 | 0x0F00_0000 |
| | UART2_BA + 0x24 | R/W | UART2波特率分频寄存器 | 0x0F00_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| 保留 | | DIV_X_EN | DIV_X_ONE | Divider_X | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 保留 | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| 波特率分频(高字节) | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 波特率分频r (低字节) | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [31:30] | 保留 | 保留 |
| [29] | **DIV_X_EN** | **分频 X 使能**<br>BRD = 波特率分频,<br>**波特率= Clock / [ M * (BRD + 2) ]** ; 默认 M 为 16.<br>0 = 禁止分频 X (the equation of M = 16)<br>1 = 使能分频 X (the equation of M = X+1, but X must > 8).<br>**注**: 在IrDA 模式下 该位禁止 |
| [28] | **DIV_X_ONE** | **Divider X equal 1**<br>0 = Divider M = X (M = X+1, but Divider_X[27:24] must > 8)<br>1 = Divider M = 1 (M = 1, but BRD[15:0] must > 3).. |
| [27:24] | **Divider_X** | **分频 X**<br>M = X+1. |
| [23:16] | 保留 | 保留 |

| | | |
|---|---|---|
| [15:8] | **BRD (High Byte)** | **波特率 分频**<br>波特率分频 高位 |
| [7:0] | **BRD (Low Byte)** | **波特率 分频**<br>波特率分频 低位 |

| DIV_X_EN | DIV_X_ONE | Divider X | BRD | 波特率公式 |
|---|---|---|---|---|
| Disable | 0 | B | A | UART_CLK / [16 * (A+2)] |
| Enable | 0 | B | A | UART_CLK / [(B+1) * (A+2)] , B must >= 8 |
| Enable | 1 | B | A | UART_CLK / (A+2), A must >=3 |

### IrDA控制寄存器(IRCR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| UA_IRCR | UART0_BA + 0x28 | R/W | UART0 IrDA控制寄存器. | 0x0000_0040 |
| | UART1_BA + 0x28 | R/W | UART1 IrDA控制寄存器 | 0x0000_0040 |
| | UART2_BA + 0x28 | R/W | UART2 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 | INV_Rx | INV_Tx | 保留 | | 保留 | Tx_SELECT | 保留 |

| Bits | 描述 | |
|---|---|---|
| [31:7] | 保留 | 保留 |
| [6] | **INV_Rx** | **INV_Rx**<br>1= Rx 输入信号反转<br>0= 无反转 |
| [5] | **INV_Tx** | **INV_Tx**<br>1= Tx 输出信号反转<br>0=无反转 |
| [4:2] | 保留 | 保留 |
| [1] | **Tx_SELECT** | **Tx_SELECT**<br>1: 使能IrDA 发送<br>0: 使能 IrDA 接收 |
| [0] | 保留 | 保留 |

**注：**在 IrDA 模式，寄存器 UA_BAUD[DIV_X_EN]必须禁止 (波特方程必须Clock / 16 * (BRD)

### UART LIN Break Field计数寄存器(UA_LIN_BCNT)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| UA_LIN_BCNT | UART0_BA + 0x2C | R/W | UART0 LIN Break Failed计数寄存器 | 0x0000_0000 |
| | UART1_BA + 0x2C | R/W | UART1 LIN Break Failed计数寄存器. | 0x0000_0000 |
| | UART2_BA + 0x2C | R/W | UART2 LIN Break Failed计数寄存器 | 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 |
| LIN_Tx_EN | LIN_Rx_EN | 保留 | | UA_LIN_BKFL | | | |

| Bits | 描述 | |
|---|---|---|
| [31:8] | 保留 | 保留 |
| [7] | **LIN_Tx_EN** | **LIN TX 钳制模式使能**<br>1 = 使能 LIN Tx 钳制模式.<br>0 = 禁止 LIN Tx 钳制模式.<br>注:当Tx break field 传输运作完成, 该位自动清除. |
| [6] | **LIN_Rx_EN** | **LIN RX 使能**<br>1 =使能 LIN Rx 模式.<br>0 = 禁止 LIN Rx 模式. |
| [3:0] | **UA_LIN_BKFL** | **UART LIN Break Field 长度**<br>表示 4-bit LIN Tx break field 计数.<br>注: break field 长度为 UA_LIN_BKFL + 2 |

### UART功能选择寄存器(UA_FUN_SEL)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| UA_FUN_SEL | UART0_BA + 0x30 | R/W | UART0功能选择寄存器 | 0x0000_0000 |
| | UART1_BA + 0x30 | R/W | UART1功能选择寄存器 | 0x0000_0000 |
| | UART2_BA + 0x30 | R/W | UART2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 | | | | | | IrDA_EN | LIN_EN |

| Bits | 描述 | |
|---|---|---|
| [31:2] | 保留 | 保留 |
| [1] | **IrDA_EN** | **IrDA 功能使能**<br>1 =使能 IrDA功能.<br>0 =禁止 IrDA功能. |
| [0] | **LIN_EN** | **LIN 功能使能**<br>1 = 使能 LIN 功能.<br>0 = 禁止 LIN 功能. |

## 5.13 局域控制网络总线(CAN Bus)

### 5.13.1 简介

区域控制器 (CAN) 为串行通讯协议 具有多主机(multi-master) 和高效率实时控制功能，安全性能高 成本低. 应用于高速网络. 在汽车电子, 车身控制, 传感器, 防滑系统等,. CAN 连接速度可达 1Mbit/s.

在 CAN 系统中,一个节点未应用于任何系统配置信息 (station addresses).任何节点可增加到网络中，无需进行硬件和软件改变. 总线上信息发送为固定模式.当总线空闲时, 任何的连接单元可进行数据传递. 信息内容由 IDENTIFIER定义. IDENTIFIER 未指示信息的目的文件, 仅表示数据意义, 因此 网络中所有的节点可通过信息过滤功能判断.内部 CAN 网络保证通过或未通过所有节点信息同时被承认.

### 5.13.2 特性

- CAN 2.0B 协议兼容
- AMBA APB 总线特性兼容
- Multi-主机模式
- 支持 11-位 标志符作为 29-位 标志符
- 比特速率达 1Mbits/s
- NRZ 位译码
- 错误侦测: 位错误bit error, 填装错误 stuff error, 格式错误 form error, 15位 CRC 侦测, 和应答错误
- Listen only 模式 (无应答, 不相应错误标志)
- 滤波扩展(4-byte code, 4-byte mask)
- 每条CAN总线上发生错误产生相应的中断
- 扩展接收缓冲(8-byte FIFO)
- 唤醒功能

### 5.13.3 框图

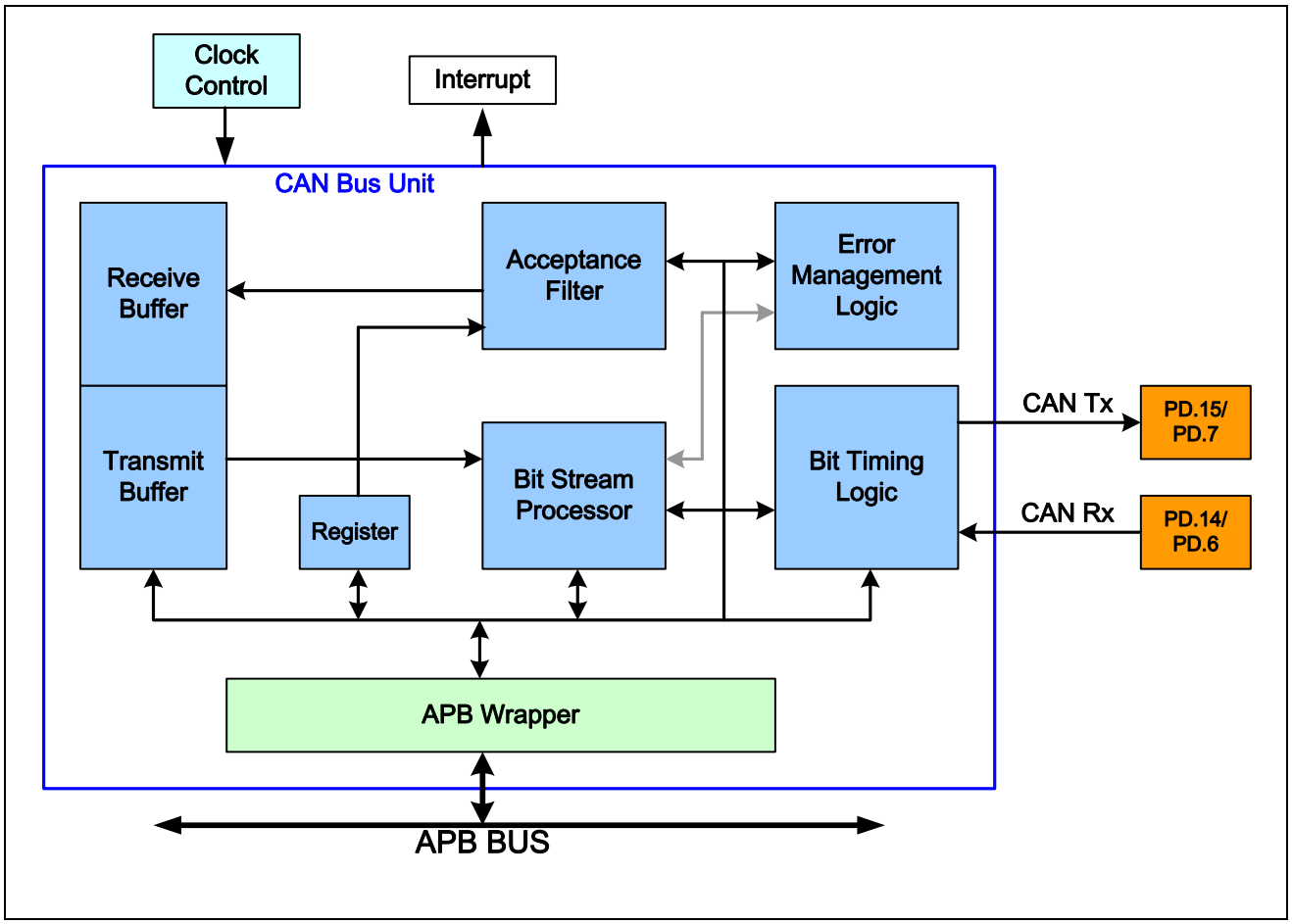



图 5-60 CAN 总线框图

### 5.13.4 功能描述

总线具有补充的2 路逻辑值,一 显性位逻辑 low level 一隐性位 逻辑 高电平.

*5.13.4.1 结构形式*

2种不同长度和格式的 IDENTIFIER: 11 位 IDENTIFIER 为标准结构.还包括 29 位 IDENTIFIER 为扩展结构.

*5.13.4.2 结构类型*

信息传递受 4 种不同结构类型控制:

1. **A DATA FRAME** 数据从传输到接收的运输.
2. **A REMOTE DATA FRAME** 通过总线单元向传输DATA FRAME IDENTIFIER发送请求.
3. **An ERROR FRAME** 传输通过 任何单元侦测总线 error.
4. **An OVERLOAD FRAME** 应用于提供额外延时 在当前和随后 DATA 或 REMOTE FRAMES 之间.

DATA FRAMES 和 REMOTE FRAMES 可应用于标准架构和扩展架构; 通过t INTERFRAME SPACE 把架构分隔.

*5.13.4.3 DATA 结构*

DATA FRAME 包括 7个不同位: START of FRAME, ARBITRATION FIELD, CONTROL FIELD, CRC FILED, ACK FIELD, and END OF FRAME. 数据区域长度可为 0.

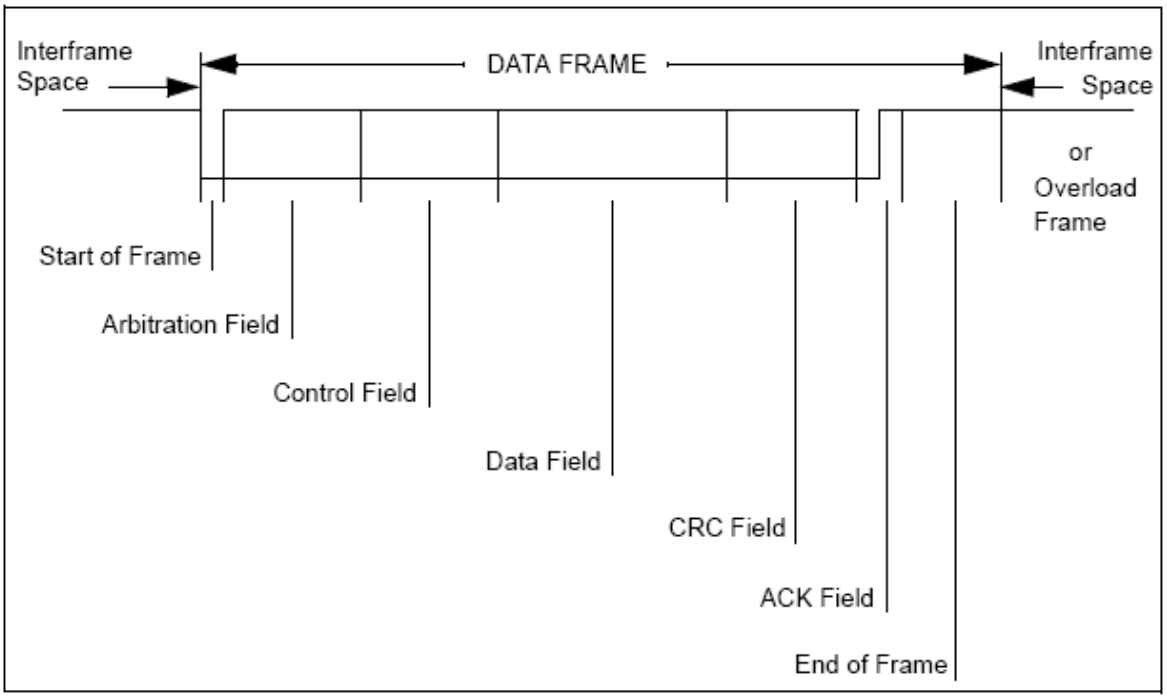



图 5-61 DATA 帧格式

1 **开始帧** (标准和扩展格式相同)

开始帧(SOF) 标志 DATA FRAMES 和 REMOTE FRAMES 开始. 包含单一 'dominant' 位. 当总线为空闲模式 标志数据允许传输. 所有状态和主要触发同步， 通过START OF FRAME 开始传输状态.

**2 FIELD 仲裁**

ARBITRATION FIELD 的格式在标准模式和扩展模式下不相同.

在标准模式下, ARBITRATION FIELD 包括 11 位 IDENTIFIER 和 RTR位. IDENTIFIER 表示 ID-28… ID-18.

在扩展模式下, ARBITRATION FIELD包括 29 位 IDENTIFIER, SRR位, IDE位, 和 RTR位. IDENTIFIER 表示 ID-28… ID-0.

为了区别标准格式和扩展格式 the 保留位 r1 用 IDE位表示.

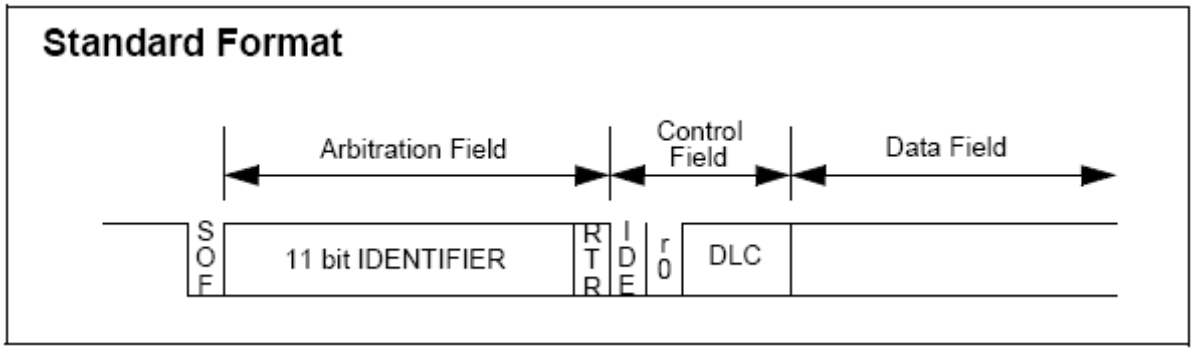



图 5-62 标准格式

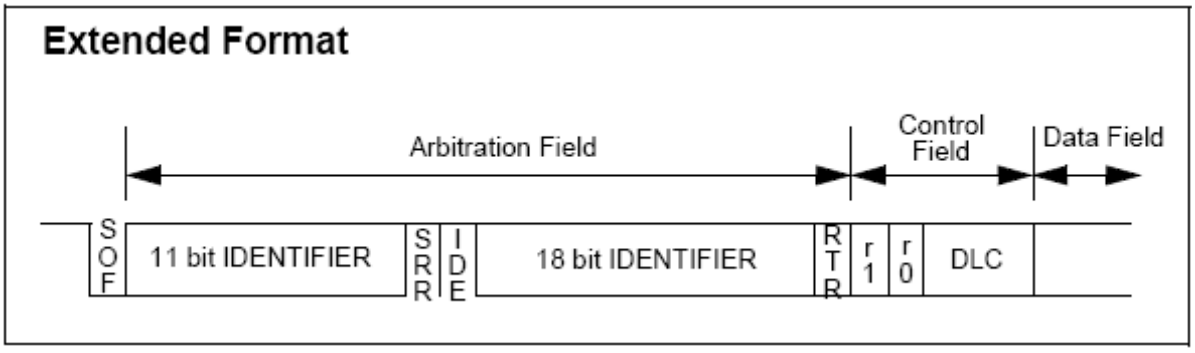



图 5-63 扩展格式

*3* **标志符**

A. 标准格式

IDENTIFIER的长度位11 位和 相应得基本ID. 该位传输次序从 ID-28 到 ID-18. 最小有意义位为 ID-18. 最多 7 位有意义位为 (ID-28 ~ ID-22) 必须为 'recessive'隐性状态.

B. 扩展格式

对比标准格式,扩展格式包括 29 位. 此格式包括 2 个部分: 11位 基础和 扩展的 ID 18 位.

B.1. Based ID

基本 ID 包括 11 位. 传输次序从 ID-28 ~ ID-18. 同标准模式标志符. 基本 ID 定义扩展结果优先级.

B.2. Extended ID

扩展 ID 包括 18 位. 传输次序为 ID-17 ~ ID-0.

在标准模式下, IDENTIFIER 遵循 RTR 位设定.

*RTR BIT:* 远程传输请求位 (标准和扩展格式相同)

DATA 帧, RTR 位具有 'dominant'统治权. 内部 REMOTE 帧 的RTR 位具有 'recessive'隐性状态. 外部扩展帧 ID 首先传输, 接下来的是 IDE位和 SRR 位. 外部 ID在 SRR 位后 进行传输.

*SRR BIT:* 替代远程请求位 (扩展格式)

SRR 为隐性位.传输方式同标准模式下的 RTR位 ,替代在标准模式下的 RTR位t.

*IDE BIT:* 标志符扩展位(扩展格式)

IDE 位包括 (a). ARBITRATION FIELD 用于扩展模式. (b). 控制标准模式现场.

4 **控制区域** : (标准和扩展格式相同)

控制区域 CONTROL FILED 包括 6 位. 此控制区域格式不同于标准格式和扩展格式. 标准格式包括 数据长度码, IDE 位, 用于传输'显性', 和保留位 r0. 扩展格式包括 数据长度码 2 位 保留位 r1 和 r0. 保留位发送 '显性',接收 '显性' 和'隐性' 位 将共同起作用.

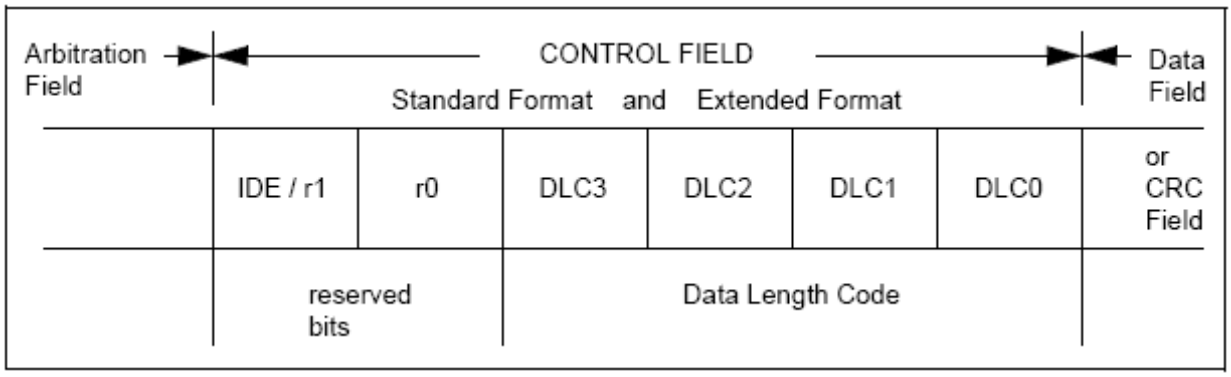



图 5-64 控制区域格式

*数据长度码 (DATA LENGTH CODE)*: (标准和扩展格式相同)

数据区域通过数据长度编码来表示. 数据长度编码包括 4 位宽度和传输控制区域位. 被认可的数据长度为 0~8.

5 **数据区域** (标准和扩展格式相同)

数据区域 DATA FIELD 由传输的数据桢组成.包括从 0 ~ 8 字节, 每组包括 8 位 首先进行 MSB 的传输..

6 **CRC区域**(标准和扩展格式相同)

包括CRC SEQUENCE ,遵循CRC DELIMITER

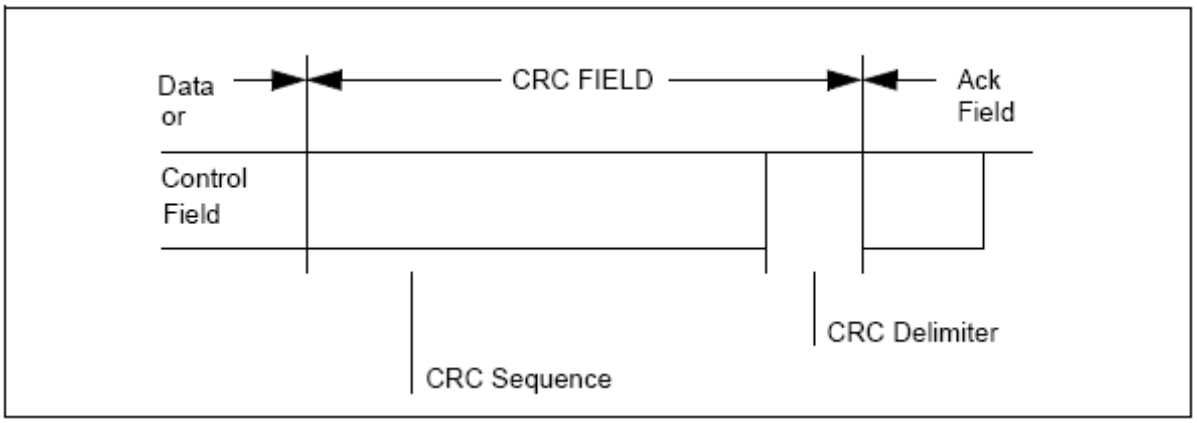



图 5-65 CRC 区域格式

*CRC顺序 (CRC SEQUENCE)* (标准和扩展格式相同)

该结构用于检测 循环冗余的代码,确保最适合的数据桢长度小于 127 位 (BCH 代码).

为了实现 CRC 计算, 多项式由分立项所组成, 多项式的系数由 de-stuffed 位 决定并组成开始桢, ARBITRATION FIELD, CONTROL FIELD, DATA FILED (if present) 和15位最低系数, 一直到 0. 多相式由下列分立项组成:

$$X^{15} + X^{14} + X^{10} + X^{8} + X^{7} + X^{4} + X^{3} + 1$$

多项分割后的其他项为 CRC SEQUENCE 总线传输结束.

*CRC分隔符 (CRC DELIMITER)* (标准和扩展格式相同)

CRC顺序遵循 CRC 分隔符由单一'隐性' 信号位组成.

7 **ACK 区域** (标准和扩展格式相同)

ACK区域为2位长度包括 ACK SLOT 和 ACK分隔符. 在 ACK区域模式下,传输发送 2 位 '隐性' 位.

RECEIVER 表示正确的接收到有效信息, 在ACK SLOT 期间 (发送'ACK')通过发送 '显性' 位向发送器件报告.

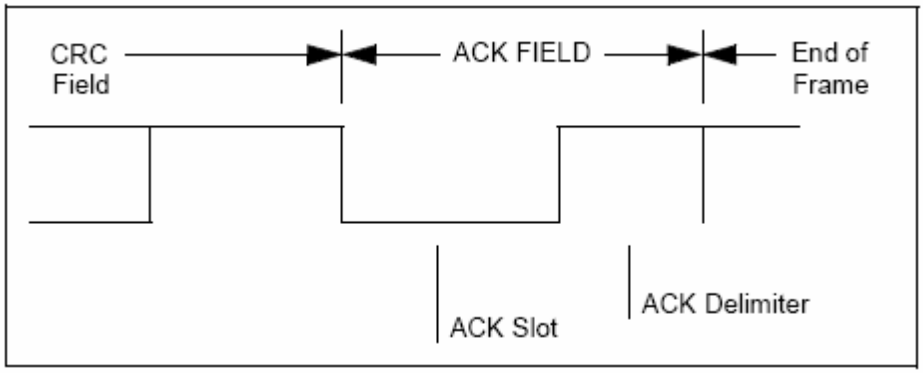



图 5-66 ACK 区域格式

*ACK SLOT:*

所有站点收到报告的匹配 CRC_SEQUENCE报告由内部 ACK SLOT通过由特定标注的发送器件'显性'位向'隐性'改变而控制 .

在ACK SLOT 期间 (发送 'ACK')通过发送 '隐性' 位 向发送器件报告

*ACK DELIMITER:*

ACK分隔符为ACK区域的第二位 为 '隐性' 位. 同理, ACK SLOT 被2 '隐性' 位(CRC 分隔符, ACK分隔符) 环绕.

8 **END OF FRAME** (标准和扩展格式相同)

每 DATA FRAME 和 REMOTE FRAME通过标志序列组成 7 位 '隐性' 位.

*5.13.4.4 REMOTE 帧*

作为有效的 RECEIVER 站点,用于确定各站点分别发送REMOTE FRAME,数据 开始传输. REMOTE FRAME 存在于标准模式和扩展模式中. 在此两类型下, 包括 6 个不同位: START OF FRAME, ARBITRATION FIELD, CONTROL FILED, CRC FILED, ACK FIELD, END OF FRAME.

相反对于 DATA FRAMES, RTR 位 REMOTE FRAME 为 '隐性'. 无 DATA FILED, 将不受到 DATA LENGTH CODE控制 可容忍的数据值为 0…8. 此为 DATA LENGTH CODE 对应的 DATA FRAME.

RTR 位的极性表示 传输结构为 DATA FRAME (RTR 位 '显性')或者 REMOTE FRAME (RTR 为 '隐性')

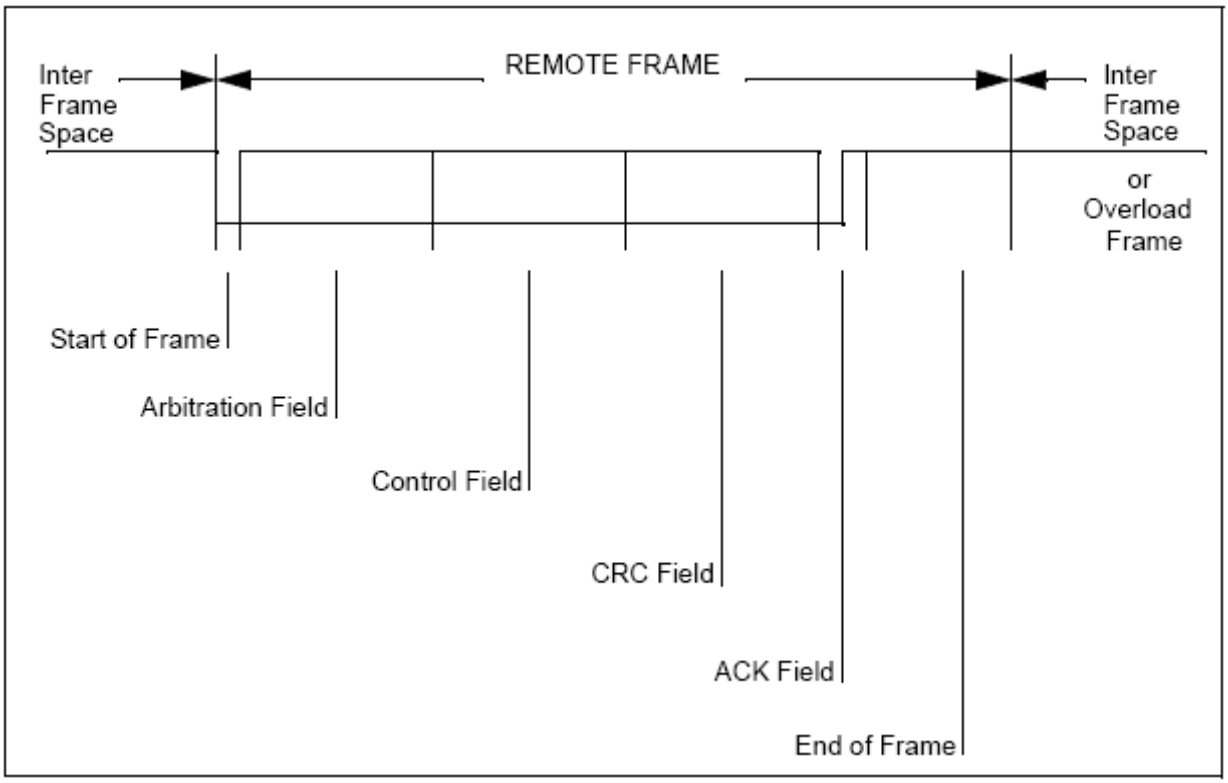



图 5-67 REMOTE FRAME 格式

*5.13.4.5 ERROR FRAME*

ERROR FRAME 包括 2 个不同区域. 开始区域支持不同节点的 ERROR FLAG 重叠.另外区域为 ERROR DELIMITER.

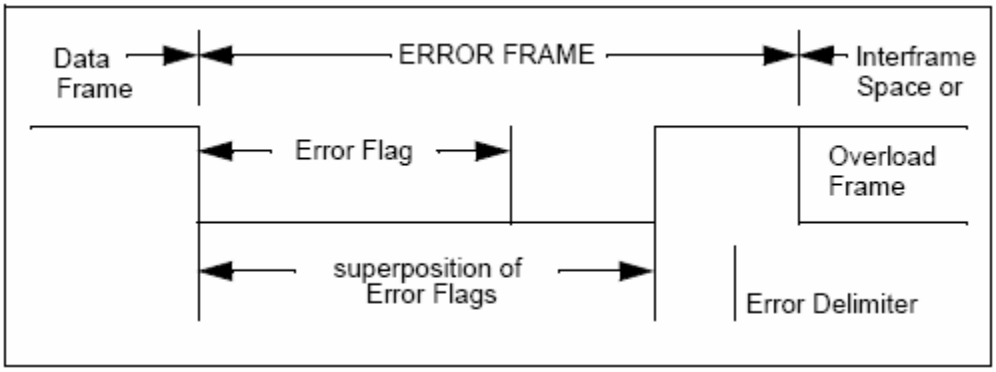



图 5-68 ERROR FRAME 格式

此为 2个 错误标志的标志位: ACTIVE ERROR FLAG 和 PASSIVE ERROR FLAG. ACTIVE ERROR FLAG 包括 6 位连续的 '显性' 位 . PASSIVE ERROR FLAG包括 6 位 连续的 '隐性' 位,除非改写通过 '显性' 位向另外节点.

ERROR DELIMITER 包括 8 ‘隐性’ 位.传输 ERROR FLAG 后, 每个节点发送 隐性’ 位 并且监控总线只到监控到‘隐性’ 位. 然后开始传输 7 位 以上的 ‘recessive’ 位.

### 5.13.5 控制寄存器列表

**R: 只读, W: 只写, R/W: 可读写**, **C**: 仅值为0时可写

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **CAN_BA0 = 0x4018_0000**<br>**CAN_BA1 = 0x4018_4000** | | | | |
| **MODE** | CAN_BA+0x00 | R/W | 模式寄存器 | 0x0000_0000 |
| **COMMANDR** | CAN_BA+0x04 | R/W | 命令寄存器 | 0x0000_0000 |
| **BSR** | CAN_BA+0x08 | R | 总线状态寄存器 | 0x0000_0140 |
| **INTR** | CAN_BA+0x0C | R/W | 中断状态寄存器 | 0x0000_0000 |
| **INTEN** | CAN_BA+0x10 | R/W | 中断使能寄存器 | 0x0000_0000 |
| **BTIM** | CAN_BA+0x14 | R/W | 总线位时序寄存器 | 0x0000_1100 |
| **ERRCR** | CAN_BA+0x20 | R | 接收错误计数寄存器 | 0x0000_0000 |
| **RECNTR** | CAN_BA+0x28 | R | 发送错误计数寄存器 | 0x0000_0000 |
| **TECNTR** | CAN_BA+0x2C | R/W | 发送祯信息寄存器 | 0x0000_0000 |
| **TXFINFO** | CAN_BA+0x30 | R/W | 发送标示符寄存器 | 0x0000_0000 |
| **TXID** | CAN_BA+0x34 | R/W | 发送数据寄存器 A | 0x0000_0000 |
| **TXDATA_A** | CAN_BA+0x38 | R/W | 发送数据寄存器 B | 0x0000_0000 |
| **TXDATA_B** | CAN_BA+0x3C | R/W | 接收错误计数寄存器 | 0x0000_0000 |
| **RXFINFO** | CAN_BA+0x40 | R | 接收祯信息寄存器 | 0x0000_0000 |
| **RXID** | CAN_BA+0x44 | R | 接收标示符寄存器 | 0x0000_0000 |
| **RXDATA_A** | CAN_BA+0x48 | R | 接收数据寄存器 A | 0x0000_0000 |
| **RXDATA_B** | CAN_BA+0x4C | R | 接收数据寄存器 B | 0x0000_0000 |
| **ACR** | CAN_BA+0x50 | R/W | 验收代码寄存器 | 0x0000_0000 |
| **AMR** | CAN_BA+0x54 | R/W | 验收屏蔽寄存器 | 0xFFFF_FFFF |

### 模式寄存器(MR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| MODE | 0x00 | R/W | CAN总线操作模式寄存器 | 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 | | | | | | LOM | RSTM |

| Bits | 描述 | |
|---|---|---|
| [31:2] | 保留 | 保留 |
| 1 | **LOM** | **Listen Only Mode**<br>使能该位, 可接收输入数据 但如果 ID frame 和默认值相匹配 将不能对 ACK 信号做出相应.<br>1 = 使能 Listen Only Mode.<br>0 = 无效 |
| 0 | **RSTM** | **复位模式**<br>禁止转播功能 复位 H/W 进入 idle 状态.<br>1 = 复位模式<br>0 = 无效 |

### 命令寄存器(COMMAND)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| COMMAND | 0x04 | R/W | 命令寄存器 | 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 |
| CAN_EN | WAKEUP_EN | OVERLOAD_EN | 保留 | | | ABRT | TR |

| Bits | 描述 | |
|---|---|---|
| [31:8] | 保留 | 保留 |
| 7 | **CAN_EN** | **CAN BUS 使能**<br>1 = 使能 CAB BUS 控制单元<br>0 = 禁止 CAN Bus |
| 6 | **WAKEUP_EN** | **唤醒使能**<br>1 = 使能唤醒功能<br>当系统在休眠模式下,可使能唤醒系统.在CAN Bus中 同时拴牢接收信号 (RX) 从 'recessive' ~ 'dominant' 并且 WAKEUP_EN 置 1, 将唤醒系统.<br>注: 当系统 wakeup, 该位必须清除 在用户请中断标志位之前.<br>0 = 禁止唤醒功能. |
| [5] | **OVERFLOAD_EN** | **OVERLOAD 使能**<br>1 = 使能 OVERLOAD FRAME<br>OVERFLOW FRAME 将进入 CAN总线.<br>0 = 禁止 OVERFLOW FRAME |
| [4:2] | 保留 | 保留 |
| 1 | **ABRT** | **中断自动重传输功能**<br>1 =当信息传输失败,中断信息在次传输.<br>0 =当信息传输失败,信息在次传输. |

| | | |
|---|---|---|
| 0 | **TR** | **传输请求**<br>1 = 信息将传输<br>0 = 无效 |

### 总线状态寄存器(BSR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| BSR | 0x08 | R | 总线状态寄存器 | 0x0000_0140 |

| 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 | | | | | | EPS | EAS |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| BS2 | BS | TS | RS | TCS | 保留 | | |

| Bits | 描述 | |
|---|---|---|
| [31:10] | 保留 | 保留 |
| 9 | **EPS** | **Error 被动状态**<br>1 =总线保持在Error 被动状态<br>0 =总线不保持在Error 被动状态 |
| 8 | **EAS** | **Error 主动状态**<br>1 = 总线保持在Error 主动状态<br>0 =总线不保持在Error 主动状态 |
| 7 | **BS2** | **总线ON/OFF 状态**<br>1 = 总线在 OFF 状态<br>0 = 总线在 ON 状态; ACTIVE 或 PASSIVE 状态 |
| 6 | **BS** | **总线空闲状态标志**<br>1 = 总线空闲; 控制器和总线无相关<br>0 = 总线上无任何触发信号; |
| 5 | **TS** | **发送状态标志**<br>1 =总线传输信息<br>0 = 空闲 |
| 4 | **RS** | **接收状态标志**<br>1 = 控制器接收数据<br>0 = 空闲 |

| | | |
|---|---|---|
| 3 | **TCS** | **传输完成状态标志**<br>1 = 当前被请求信息已经成功传递<br>0 =不完全 |
| [2:0] | 保留 | 保留 |

### 中断状态寄存器(INTR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **INTR** | 0x0C | R/W | 中断状态寄存器 | 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 |
| BEI | ALI | 保留 | WUI | 保留 | | TI | RI |

| Bits | 描述 | |
|---|---|---|
| [31:8] | 保留 | 保留 |
| 7 | **BEI** | **总线Error 中断标志**<br>1 : 当节点侦测到CAN总线上 error 并且 BEIE 置位，内部中断使能。通过软件清除.<br>0: 无总线 error 中断 |
| 6 | **ALI** | **Arbitration Lost 中断标志**<br>1 = 该位置位 当节点失去仲裁 成为接收方 并且 ALIE 位置位使能内部中断 寄存器. 通过软件清除.<br>0: 无 arbitration lost 中断 |
| 5 | 保留 | 保留 |
| 4 | **WUI** | **Wake-Up 中断标志**<br>1 = 当节点为睡眠模式和总线侦测活跃 WUIE 置位 内部中断使能 该位将置位. 通过写 1 请该位。如果进入睡眠模式后，该中断可用于唤醒。 WAKEUP_EN 位将清除.<br>0 = 无唤醒中断 |
| [3:2] | 保留 | 保留 |
| 1 | **TI** | **发送中断标志**<br>1 = 传输完成 TIE 位 置位 使能内部中断 该位置位. 通过写 1 请该位.<br>0 = 无传输完成信息 |

| | | |
|---|---|---|
| 0 | **RI** | **接收中断标志**<br>1 = 当接收完成 RIE 位 置位 使能内部中断 该位置位. 通过写 1 请该位.<br>0 = 无接收 RXFIFO 信息. |

### 中断使能寄存器(INTEN)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **INTEN** | 0x10 | R/W | 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 | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **BEIE** | **ALIE** | **保留** | **WUIE** | **保留** | | **TIE** | **RIE** |

| Bits | 描述 | |
|---|---|---|
| [31:8] | 保留 | 保留 |
| 7 | **BEIE** | **总线Error 中断使能**<br>1 = 总线error 中断使能.<br>0 = 禁止 |
| 6 | **ALIE** | **Arbitration Lost 中断使能**<br>1 = arbitration lost 中断使能.<br>0 = 禁止 |
| 5 | 保留 | 保留 |
| 4 | **WUIE** | **Wake-Up 中断使能**<br>1 = wake-up 中断使能.<br>0 = 禁止 |
| [3:2] | 保留 | 保留 |
| 1 | **TIE** | **发送中断使能**<br>1 = 传输完成 中断使能.<br>0 = 禁止 |

| | | |
|---|---|---|
| 0 | **RIE** | **接收中断使能**<br>1 = 接收完成中断使能.<br>0 = 禁止 |

总线位时序寄存器(BTIMR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| BTIMR | 0x14 | R/W | 总线位时序寄存器 | 0x0000_1100 |

| 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 |
| SAMP | TSEG2 | | | | TSEG1[4:2] | | |
| 7 | 6 | 5 | 7 | 6 | 5 | 7 | 6 |
| TSEG1[1:0] | | SJW | | 保留 | | | |

| Bits | 描述 | |
|---|---|---|
| [31:16] | 保留 | 保留 |
| 15 | **SAMP** | **取样点数量**<br>1 = 3 倍; 一个 bit 位 总线取样 3 次; 装载到TSEG1 + 1, TSEG1 和 TSEG1 - 1.<br>推荐 low/medium 总线速度 ,使总线过滤功能有效<br>0 = 单一; 总线 取样装载到 (TSEG1 + 1);推荐总线高速. |
| [14:11] | **TSEG2** | **时间区域2**<br>(TSEG1 + 1) 和 (TSEG2 + 1) 默认每 bi 时钟周期和指定 取样点.<br>(TSEG2 +1) 定义 取样每位位置,在SYNC SEG 前.<br>注: 3段包括:SYNC, TSEG1, 和 TSEG2. TSEG2 的值大于 (TSEG1/2, 2). |
| [10:6] | **TSEG1** | **时间区域 1**<br>(TSEG1 + 1) 定义周期 在SYNC 段和取样点(不包括 SYNC 段)之间. TSEG1 最大值为 16 最小值为 2. |
| [5:4] | **SJW** | **同步跳转宽度**<br>在不同节点 时钟源之间 补偿相位, 硬件可在每位中调节时间单元的上限. SJW 每位时钟周期的最大数 缩短和延长时间单元的上限. SJW 最大值需要小于{TSEG1, TSEG2}. (参考 BTIM 块描述) |
| [3:0] | 保留 | 保留 |

### 错误捕捉寄存器(ERRCR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| ERRCR | 0x20 | R | 错误捕捉寄存器 | 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 | ID18_NM | ID11_NM | STUFF_ERRS | FORM_ERRS | CRC_ERRS | ACK_ERRS | BIT_ERRS |

| Bits | 描述 | |
|---|---|---|
| [31:7] | 保留 | 保留 |
| 6 | **ID18_NM** | **ID18 无匹配状态**<br>1 = 11-bits识别不匹配接收数据. CRC error 或 写‘0’清除 状态将被更新.<br>0 = 11-bits 识别匹配接收数据. |
| 5 | **ID11_NM** | **ID11 无匹配状态**<br>1 = 11-bits识别不匹配接收数据. CRC error 或 写‘0’清除 状态将被更新.<br>0 = 11-bits识别匹配接收数据. |
| 4 | **STUFF_ERRS** | **Stuff 错误**<br>1 = A Stuff Error将被侦测 当 信息 6th 连续位时间和 和 bit level相等 通过位填充的方式编码.<br>0 = 无 stuff error |
| 3 | **FROM_ERRS** | **Form 错误**<br>1 = A Form Error 将被侦测 当 fixed-form 区域包括一个或多个违法 位. fixed-form 包括 CRC 分隔符, ACK 分隔符, Error 信息分隔符,溢出分隔符和 EOF<br>0 = 无固定形式 error |

| | | |
|---|---|---|
| 2 | **CRC_ERRS** | **CRC 错误**<br>1 = CRC 结果序列由 CRC 通过传输完成. 接收CRC 和传输相同.如果计算结果值和 CRC 值不相匹配, A CRC Error 将被侦测.<br>0 = 无 CRC error |
| 1 | **ACK_ERRS** | **应答错误**<br>1 = 应答错误通过传输被侦测到， 无论是否监控到 ‘dominant’位在 ACK SLOT期间.<br>0 = 无应答错误 |
| 0 | **BIT_ERRS** | **Bit Error**<br>1 = 总线发送节点 同时监控总线. A Bit Error 同时被侦测, 当监控位值和 发送位值不相匹配.<br>0 = 无 bit error |

接收错误计数寄存器(RECNTR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| RECNTR | 0x28 | R | 接收错误计数寄存器 | 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 |
| RECNT[7:0] | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [31:8] | 保留 | 保留 |
| [7:0] | RECNT | **接收错误计数器**<br>RX error 接收寄存器 反映当前接收到的 error 计数器. 在运行模式下, 该寄存器展现 CPU 为只读寄存器. (当总线上发生off , RECNT初始化为0) |

### 发送错误计数寄存器(TECNTR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **TECNTR** | 0x2C | R | 发送错误计数寄存器 | 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** |
| TECNT[7:0] | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [31:8] | 保留 | 保留 |
| [7:0] | **TECNT** | **发送错误计数器**<br>发送错误计数器寄存器 反映当前传输 error 计数器的值. 在运行模式下, 该寄存器展现 CPU 为只读寄存器. |

发送祯信息寄存器(TXFINFO)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| TXFINFO | 0x30 | R/W | 发送祯信息寄存器 | 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 |
| TXFF | TXRTR | TXDLC[5:0] | | | | | |

| Bits | 描述 | |
|---|---|---|
| [31:8] | 保留 | 保留 |
| 7 | TXFF | **发送祯格式**<br>1 = 传输格式为扩展模式<br>0 =传输格式为标准模式 |
| 6 | TXRTR | **远端传输请求**<br>1 = 传输格式为 REMOTE FRAME (标准格式同扩展格式)<br>0 = 传输格式为 DATA FRAME |

| | | |
|---|---|---|
| [5:0] | **TXDLC** | **发送数据长度码**<br>通过 TXDLC 表示传输数据的长度.<br>Number of Data Bytes / DLC[3] / DLC[2] / DLC[1] / DLC[0]<br>0 / d / d / d / d<br>1 / d / d / d / r<br>2 / d / d / r / d<br>3 / d / d / r / r<br>4 / d / r / d / d<br>5 / d / r / d / r<br>6 / d / r / r / d<br>7 / d / r / r / r<br>8 / r / d / d / d |

发送标示符寄存器(TXID)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| TXID | 0x34 | R/W | 发送标示符寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:3] | TXID | 发送标示符<br>TXID[28:18] 为 11-bit 标志符 应用于标准格式<br>TX[17:0] is the18-bit 标志符 应用于扩展格式 |
| [2:0] | 保留 | 保留 |

发送数据_A 寄存器(TXDATA_A)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| TXDATA_A | 0x38 | R/W | 发送数据_A 寄存器 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| TXDATA4 | | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| TXDATA3 | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| TXDATA2 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| TXDATA1 | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [31:24] | TXDATA4 | **发送数据缓存 4**<br>TXDATA4 为 4th 数据传输缓冲器. 若 TXDLC = 6'h4,数据缓冲器中的内容将发送到 Data Filed |
| [23:16] | TXDATA3 | **发送数据缓存 3**<br>TXDATA4 为 3rd 数据传输缓冲器. 若 TXDLC = 6'h3, 数据缓冲器中的内容将发送到 Data Filed |
| [15:8] | TXDATA2 | **发送数据缓存 2**<br>TXDATA4 为 2nd 数据传输缓冲器 . 若 TXDLC = 6'h2, 数据缓冲器中的内容将发送到 Data Filed |
| [7:0] | TXDATA1 | **发送数据缓存 1**<br>TXDATA4 为 1st 数据传输缓冲器 . 若 TXDLC = 6'h1, 数据缓冲器中的内容将发送到 Data Filed |

发送数据_B 寄存器(TXDATA_B)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| TXDATA_B | 0x3C | R/W | 发送数据_B 寄存器 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| TXDATA8 | | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| TXDATA7 | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| TXDATA6 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| TXDATA5 | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [31:24] | **TXDATA8** | **发送数据缓存 8**<br>TXDATA4 为 8th 数据传输缓冲器. 若 TXDLC = 6'h8, 数据缓冲器中的内容将发送到 Data Filed |
| [23:16] | **TXDATA7** | **发送数据缓存 7**<br>TXDATA4 为 7th 数据传输缓冲器 . 若 TXDLC = 6'h7, 数据缓冲器中的内容将发送到 Data Filed |
| [15:8] | **TXDATA6** | **发送数据缓存 6**<br>TXDATA4 为 6th 数据传输缓冲器. 若 TXDLC = 6'h6, 数据缓冲器中的内容将发送到 Data Filed |
| [7:0] | **TXDATA5** | **发送数据缓存 5**<br>TXDATA4 为 5th 数据传输缓冲器. 若 TXDLC = 6'h5, 数据缓冲器中的内容将发送到 Data Filed |

### 接收祯信息寄存器(RXFINFO)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **RXFINFO** | 0x40 | R | 接收祯信息寄存器 | 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** |
| **RXIDE** | **RXRTR** | 保留 | | **RXDLC[3:0]** | | | |

| Bits | 描述 | |
|---|---|---|
| [31:8] | 保留 | 保留 |
| 7 | **RXIDE** | **接收标示扩展位**<br>1: 接收标志符为扩展模式<br>0: 接收标志符为标准模式 |
| 6 | **RXRTR** | **远端接收传输请求位**<br>1: 接收格式为 REMOTE FRAME (标准模式同扩展模式)<br>0: 接收格式为 DATA FRAME |
| [5:4] | 保留 | 保留 |

| | | |
|---|---|---|
| [3:0] | **RTXDLC** | **接收数据长度码**<br>通过 RTXDLC 表示接收数据的长度.<br>Number of Data Bytes / DLC[3] / DLC[2] / DLC[1] / DLC[0]<br>0 / d / d / d / d<br>1 / d / d / d / r<br>2 / d / d / r / d<br>3 / d / d / r / r<br>4 / d / r / d / d<br>5 / d / r / d / r<br>6 / d / r / r / d<br>7 / d / r / r / r<br>8 / r / d / d / d |

### 接收标志寄存器(RXID)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **RXID** | 0x44 | R | 接收标志寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:3] | **RXID** | **接收标志**<br>RXID[28:18] 为 11-bit 标志符 用于标准模式<br>TX[17:0] 为18-bit标志符 用于扩展模式 |
| [2:0] | 保留 | 保留 |

### 接收数据_A 寄存器(RXDATA_A)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| RXDATA_A | 0x48 | R | 接收数据_A 寄存器 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| RXDATA4 | | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| RXDATA3 | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| RXDATA2 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RXDATA1 | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [31:24] | **RXDATA4** | **接收数据缓冲器 4**<br>RXDATA4 为 4th 接收数据缓冲区. 若 RXDLC = 4'h4, 数据缓冲器中的内容将发送到 Data Filed |
| [23:16] | **RXDATA3** | **接收数据缓冲器 3**<br>RXDATA4 为 3rd 接收数据缓冲区. 若 RXDLC = 4'h3, 数据缓冲器中的内容将发送到 Data Filed |
| [15:8] | **RXDATA2** | **接收数据缓冲器 2**<br>RXDATA4 为 2nd 接收数据缓冲区. 若 RXDLC = 4'h2, 数据缓冲器中的内容将发送到 Data Filed |
| [7:0] | **RXDATA1** | **接收数据缓冲器 1**<br>RXDATA4 为 1st 接收数据缓冲区. 若 RXDLC = 4'h1, 数据缓冲器中的内容将发送到 Data Filed |

### 接收数据_B 寄存器(RXDATA_B)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **RXDATA_B** | 0x4C | R | 接收数据_B 寄存器 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| **RXDATA8** | | | | | | | |
| **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| **RXDATA7** | | | | | | | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| **RXDATA6** | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **RXDATA5** | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [31:24] | **RXDATA8** | **接收数据缓冲器 8**<br>RXDATA4 为 8th 接收数据缓冲区. 若 RXDLC = 4'h8, 数据缓冲器中的内容将发送到 Data Filed |
| [23:16] | **RXDATA7** | **接收数据缓冲器 7**<br>RXDATA4 为 7th 接收数据缓冲区. 若 RXDLC = 4'h7, 数据缓冲器中的内容将发送到 Data Filed |
| [15:8] | **RXDATA6** | **接收数据缓冲器 6**<br>RXDATA4 为 6th 接收数据缓冲区. 若 RXDLC = 4'h6, 数据缓冲器中的内容将发送到 Data Filed |
| [7:0] | **RXDATA5** | **接收数据缓冲器 5**<br>RXDATA4 为 5th 接收数据缓冲区. 若 RXDLC = 4'h5, 数据缓冲器中的内容将发送到 Data Filed |

### 验收代码寄存器 (ACR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **ACR** | 0x50 | R/W | 验收代码寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:3] | **ACR** | **验收代码寄存器**<br>该位模仿被接收到的信息 (identifier) 验收代码寄存器详细说明.<br>ACR[31:21] 应用于识别 11-bit 接收信息.<br>ACR[20:3] 应用于识别 18-bit 接收信息 |
| [2:0] | 保留 | 保留 |

### 验收屏蔽寄存器(AMR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **AMR** | 0x54 | R/W | 验收屏蔽寄存器 | 0xFFFF_FFFF |

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

| Bits | 描述 | |
|---|---|---|
| [31:3] | **AMR** | **验收屏蔽寄存器**<br>该位模仿被接收到的信息 (identifier) acceptance code 寄存器详细说明.<br>0 = 相应位始终通过。<br>1 = 根据ACR位判别接收数据是否被屏蔽。<br>.ACR[31:21] 应用于识别 11-bit 接收信息.<br>ACR[20:3] 应用于识别 18-bit 接收信息 |
| [2:0] | 保留 | 保留 |

## 5.14 PS2 设备控制器(PS2D)

### 5.14.1 概述

PS/2 设备控制器为 PS/2通讯提供基本时序控制. 在设备和主机之间的通讯CLK 和 DATA pins控制. 不同于 PS/2 键盘和鼠标设备控制器, 接收/传输 代码需要固件进行代码转换. 启动控制器发送CLK 信号 在接收到发送请求后,但主机最终控制结束通讯. 当设备读取到 主机的上升沿后，设备向主机发送数据. 16 位 FIFO 应用于减少 CPU干扰. S/w 可选择 1 ~ 16 位 用于连续传输.

### 5.14.2 特性

- APB 界面兼容
- 主机通讯禁止和发送请求侦测
- 接收 帧错误侦测
- 可编程1 ~ 16 位传输缓冲 以减少 CPU 干扰
- 双数据缓冲功能
- S/W override 总线

### 5.14.3 PS/2 系统框图

PS/2 设备驱动器 包括APB 界面和时序逻辑， 用于DATA 和 CLK lines.

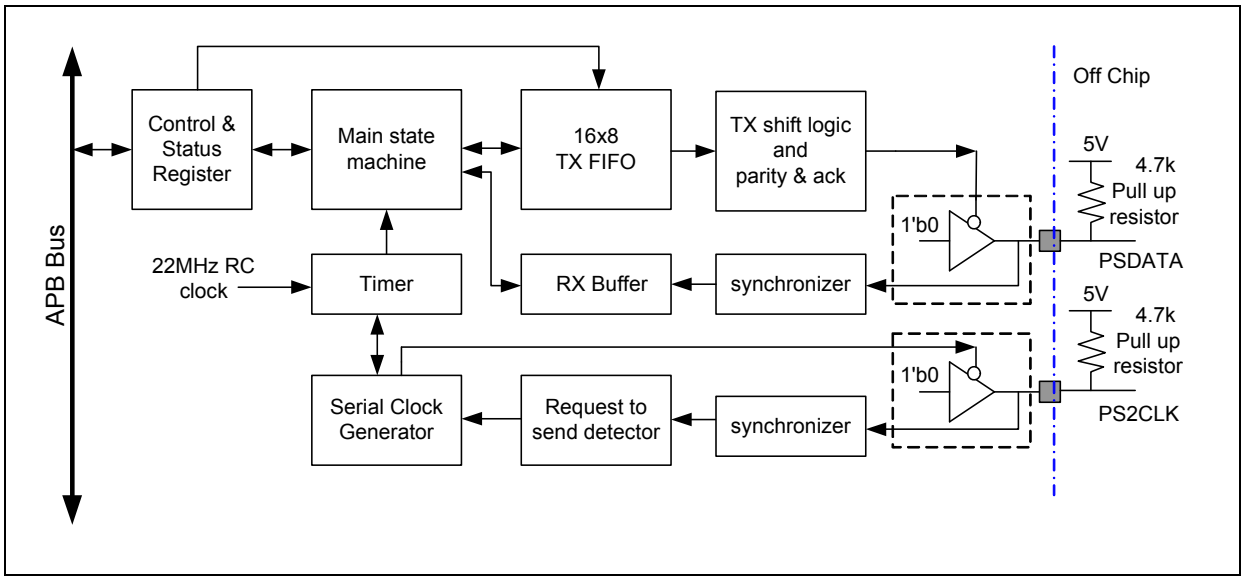



图 5-69 PS/2 设备框图

### 5.14.4 PS/2 通讯

PS/2 设备具有 双向双向同步 串行协议.总线为 "Idle" 模式 当2 lines 都为高 (open-collector). 该状态为设备允许开始 DATA 传输. 主机控制总线 ， 下拉CLK line 为低 将禁止通讯.

设备始终产生 CLK 信号. 如果主机需要发送 DATA, 下拉CLK line 为低需，禁止从设备进行通讯. 主机来DATA 为低 释放 CLK. "Request-to-Send" 状态和设备信号开始 CLK 脉冲.

| DATA | CLK | 总线状态 |
|---|---|---|
| High | High | Idle |
| High | Low | 禁止通讯 |
| Low | High | 主机请求发送 |

所有数据每次传输 1 字节，每字节包括 11-12 位. 如下:

- 1 开始位. 一直为 0.
- 8 数据位.
- 1 奇偶位(odd parity).
- 1 停止位. 该位一直 1.
- 1 应答位 (主机~设备通讯)

如果1's数据位和复位（0）为偶数，将通过奇偶位将1's 数据位加上奇偶位合计成奇数(odd parity.) .应用于error 侦测. 设备需要检查该位，如果有错误将接收到不正常的命令 并做出回应.主机可通过拉底Clock line 至少100ms 后，将禁止通讯.如果11$^{th}$ 时钟脉冲之前停止传输, 设备将异常停止当前传输 ， 主机释放Clock，将准备传输当前数据.为了有充足的应用于主机软件解码，传输逻辑封锁 通过设定RXINT 位, S/w 必须清RXINT 位 开始传输. S/W 可写 CLRFIFO 为 1 复位 FIFO 指示器.

**设备向主机传输**

设备应用于连续的 11-位 架构. 如下:

- 1 个开始位. 内容始终为 0.
- 8 数据位. 最高位最先传输
- 1 奇偶位(odd parity).
- 1 停止位. 该位内容一直为1.

当CLK为高，设备写 1 位数据到总线, CLK为低时，通过主机读数据. 举例如下.

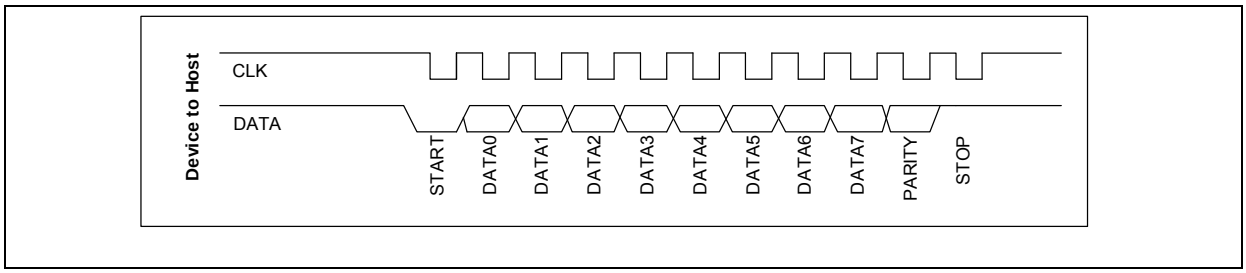



图 5-70 数据格式 设备向主机传输

**主机向设备传输**

首先, PS/2 设备一直产生 CLK 信号. 如果主机希望发送DATA, 首先需设定CLK 和 DATA在 "Request-to-send" 如下状态：

- 拉底CLK 至少100 us 禁止通讯
- 应用 "Request-to-send" 拉 DATA 为低, 释放CLK.

设备 每 10 ms 以内需 不间断的监控状态.当设备监控到此状态，将开始产生CLK 信号和8位DATA位 1 位停止位. 主机改变DATA line 当 CLK line 为底.当 CLK 为高，通过设备读数据.停止位接收后，设备发出应答信号，使DATA line 为低 并且产生CLK 脉冲. 如果11th CLK 脉冲后主机未释放DATA line, 设备将继续产生CLK脉冲 只到 DATA line释放 (设备将产生一次 error.)

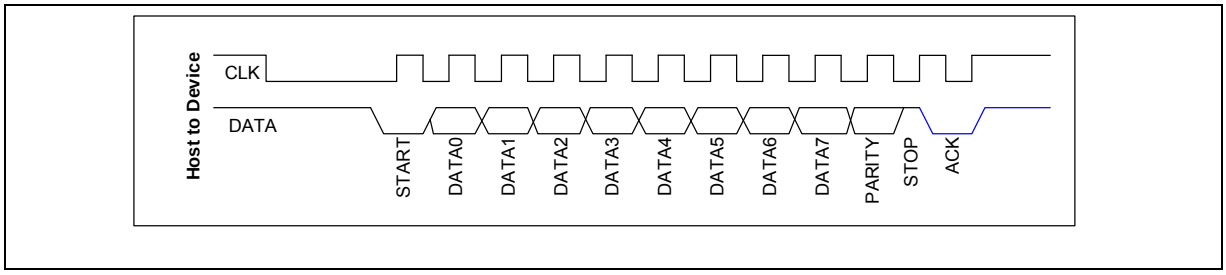



图 5-71 数据格式 主机向设备传输

主机和外设 DATA 和 CLK 详细时序框图如下:

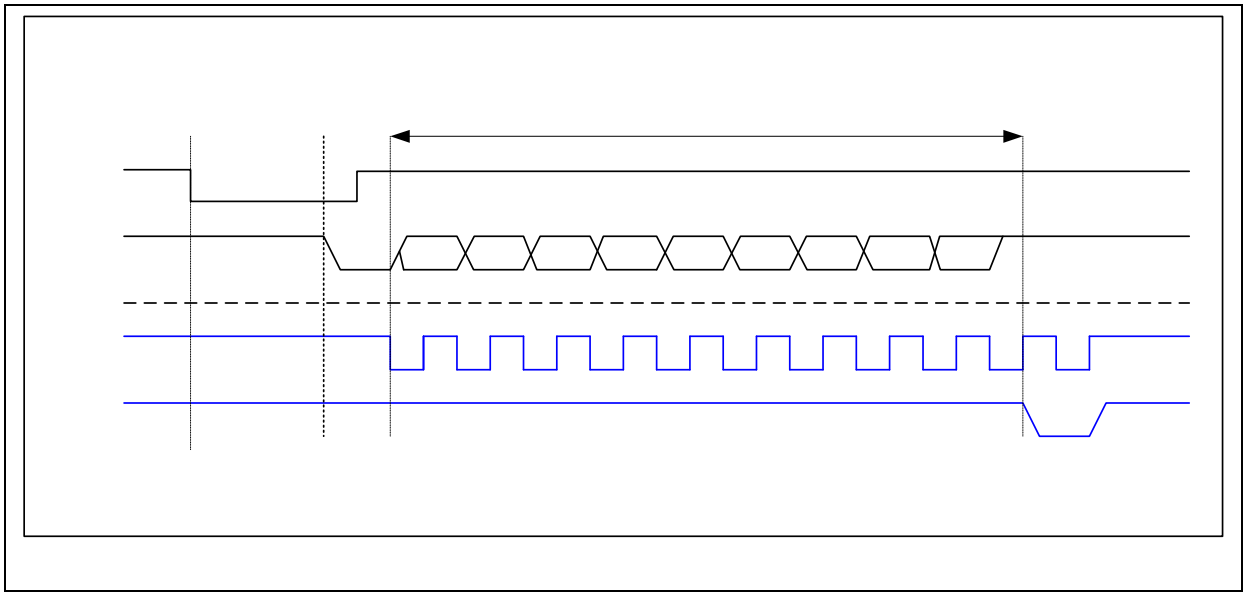
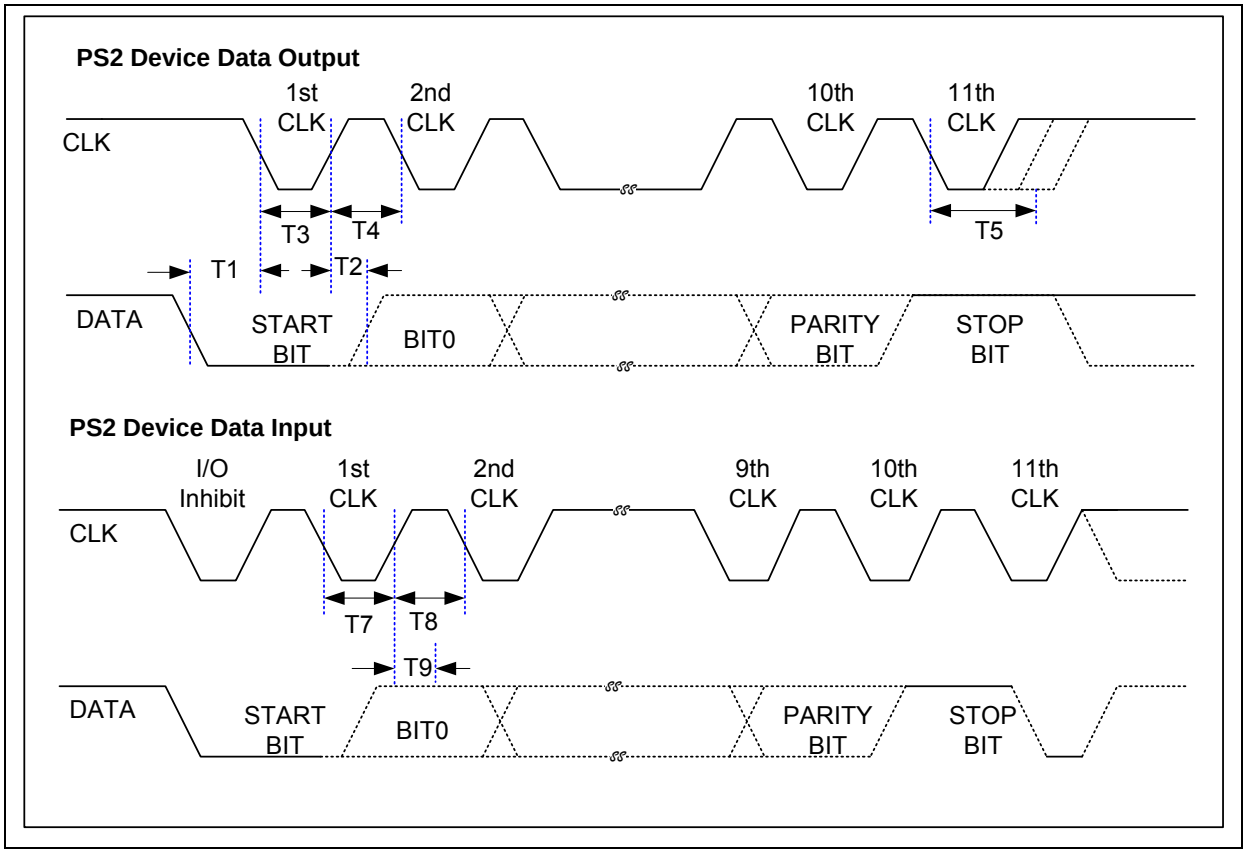


图 5-72 PS/2 数据格式

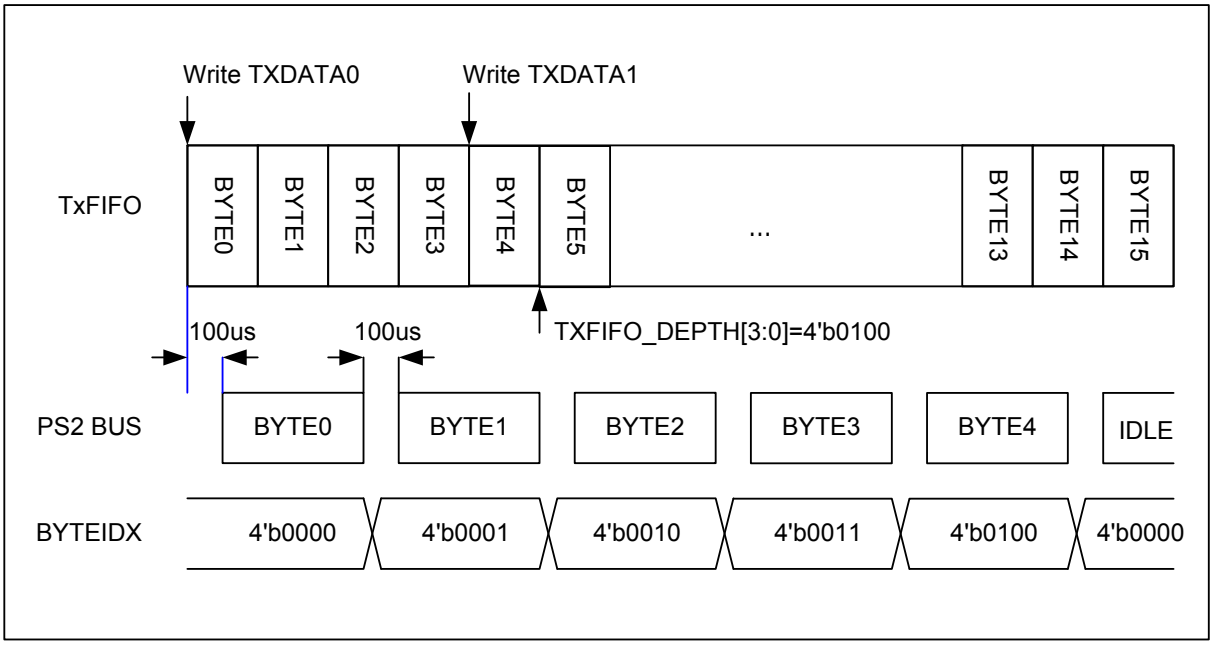
#### 5.14.4.1 PS2总线时序规范




| 符号 | 时序参数 | Min | Max |
|---|---|---|---|
| T1 | CLK 下降沿 DATA 传输 | 5us | 25us |
| T2 | CLK 上升沿 DATA 传输 | 5us | T4-5us |
| T3 | CLK 持续无效 | 30us | 50us |
| T4 | CLK持续有效 | 30us | 50us |
| T5 | clock 11后 辅助设备禁止其他传输 | >0 | 50us |
| T7 | CLK 持续无效 | 30us | 50us |
| T8 | CLK持续有效 | 30us | 50us |
| T9 | CLK 传输从无效到有效时间, 辅助设备取样时间 | 5us | 25us |

### 5.14.5 TX FIFO 运作

写PS2TXDATA0 寄存器状态开始设备和主机将通讯. S/w 需定义TXFIFO 长度 在向TX FIFO传输数据前. 1st START 位置位PS2 总线 100us 后S/w 写 TX FIFO,如果超过4 字节数据需要发送, s/w 可写剩余数据到PS2TXDATA1-3 在 4th 字节数据传输完成前. 2次连续字节将延时100us.



### 5.14.6 PS2 设备控制寄存器图表

**R: 只读, W: 只写, R/W: 可读写，C:仅在值为0时可写**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **Base Address: 0x4010_0000** | | | | |
| **PS2CON** | PS2_BA+0x000 | R/W | PS2 控制寄存器 | 0x0000_0000 |
| **PS2TXDATA0** | PS2_BA+0x004 | R/W | PS2 发送数据寄存器 0 | 0x0000_0000 |
| **PS2TXDATA1** | PS2_BA+0x008 | R/W | PS2 发送数据寄存器 0 | 0x0000_0000 |
| **PS2TXDATA2** | PS2_BA+0x00C | R/W | PS2 发送数据寄存器 0 | 0x0000_0000 |
| **PS2TXDATA3** | PS2_BA+0x010 | R/W | PS2 发送数据寄存器 0 | 0x0000_0000 |
| **PS2RXDATA** | PS2_BA+0x014 | R | PS2 接收数据寄存器 | 0x0000_0000 |
| **PS2STATUS** | PS2_BA+0x018 | R/W | PS2 状态寄存器 | 0x0000_0083 |
| **PS2INTID** | PS2_BA+0x01C | R/W | PS2 中断判定寄存器 | 0x0000_0000 |

### 5.14.7 寄存器描述

**PS2控制寄存器(PS2CON)**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| PS2CON | PS2_BA + 0x00 | R/W | PS2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 | | | | FPS2DAT | FPS2CLK | OVERRIDE | CLRFIFO |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ACK | TXFIFO_DEPTH | | | | RXINTEN | TXINTEN | PS2EN |

| Bits | 描述 | |
|---|---|---|
| [31:11] | 保留 | 保留 |
| [11] | **FPS2DAT** | **Force PS2DATA Line**<br>强制 PS2DATA低或高，无视设备控制器的内部状态（若OVERRIDE =1）<br>1 = 强制 PS2DATA 高<br>0 = 强制 PS2DATA 低 |
| [10] | **FPS2CLK** | **Force PS2CLK Line**<br>强制 PS2CLK line 低或高，无视设备控制器的内部状态（若OVERRIDE =1）<br>1 = 强制 PS2DATA line 高<br>0 = 强制 PS2DATA line 低 |
| [9] | **OVERRIDE** | **软件无视PS2 CLK/DATA Pin 状态**<br>1 = PS2CLK 和 PS2DATA pins受S/W控制<br>0 = PS2CLK 和 PS2DATA pins 不受S/W控制. |

| | | |
|---|---|---|
| [8] | **CLRFIFO** | **清 TX FIFO**<br>写 1 停止设备向主机传输. TXEMPTY 位 PS2STATUS 将置位 BYTEIDEX 将清0 无论数据缓冲器中 是否有数据. 数据缓冲器将不被清除.<br>1 = 清 FIFO<br>0 =无效 |
| [7] | **ACK** | **应答使能**<br>1 = 如果奇偶error 或停止位未接收到, 12$^{th}$ 时钟将不发送应答.<br>0 = 在12th 时钟 一直发送应答位到主机到设备通讯. |
| [6:3] | **TXFIFODIPTH** | **FIFO 传输数据长度**<br>16 位缓冲器 应用于数据传输. S/W 可定义 FIFO 长度从 1 ~ 16 字节.<br>0 = 1字节<br>1 = 2字节<br>…<br>14 = 15字节<br>15 = 16字节 |
| [2] | **RXINTEN** | **使能接收中断**<br>1 =使能数据接收完成中断<br>0 = 禁止数据接收完成中断 |
| [1] | **TXINTEN** | **使能传输中断**<br>1 = 使能数据传输完成中断<br>0 = 禁止数据传输完成中断 |
| [0] | **PS2EN** | **使能 PS2 设备**<br>使能 PS2 设备<br>1 = 使能<br>0 = 禁止 |

### PS2发送数据寄存器0-3 (PS2TXDATA0-3)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **PS2TXDATA0** | PS2_BA + 0x04 | R/W | PS2发送数据寄存器0 | 0x0000_0000 |
| **PS2TXDATA1** | PS2_BA + 0x08 | R/W | PS2发送数据寄存器1 | 0x0000_0000 |
| **PS2TXDATA2** | PS2_BA + 0x0c | R/W | PS2发送数据寄存器2 | 0x0000_0000 |
| **PS2TXDATA3** | PS2_BA + 0x10 | R/W | PS2发送数据寄存器3 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:0] | **TXDATA** | **发送数据**<br>如果总线在 IDLE 状态 设备和主机通讯 向寄存器写数据. S/w 需使能 PS2EN 在写数据到 TX 缓冲器前. |

### PS2接收数据寄存器(PS2RXDATA )

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **PS2RXDATA3** | PS2_BA + 0x14 | R/W | PS2接收数据寄存器 | 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** |
| RXDATA[7:0] | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [31:8] | 保留 | 保留 |
| [7:0] | **PS2RXDATA** | **接收数据**<br>应用于主机启动传输, 应答位发送后, 接收数据将从shift 寄存器向 PS2RXDATA 寄存器中拷贝. 下一字节接收完成前，CPU 需读寄存器，否则数据将无效， RXOVF 位 PS2STATUS[6]将置 1. |

### PS2状态寄存器(PS2STATUS)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| PS2STATUS | PS2_BA + 0x18 | R/W | PS2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 | | | | BYTEIDX[3:0] | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| TXEMPTY | RXOVF | TXBUSY | RXBUSY | RXPARTY | FRAMERR | PS2DATA | PS2CLK |

| Bits | 描述 | |
|---|---|---|
| [31:12] | 保留 | 保留 |
| [11:8] | BYTEIDX | 字节目录<br>表示 shift 寄存器 8位数据传输.当 FIFO 所有数据传输，该位清 0.<br>只读位. |

| BYTEIDX | DATA Transmit | BYTEIDX | DATA Transmit |
|---|---|---|---|
| 0000 | TXDATA0[7:0] | 1000 | TXDATA2[7:0] |
| 0001 | TXDATA0[15:8] | 1001 | TXDATA2[15:8] |
| 0010 | TXDATA0[23:16] | 1010 | TXDATA2[23:16] |
| 0011 | TXDATA0[31:24] | 1011 | TXDATA2[31:24] |
| 0100 | TXDATA1[7:0] | 1100 | TXDATA3[7:0] |
| 0101 | TXDATA1[15:8] | 1101 | TXDATA3[15:8] |
| 0110 | TXDATA1[23:16] | 1110 | TXDATA3[23:16] |
| 0111 | TXDATA1[31:24] | 1111 | TXDATA3[31:24] |

| | | |
|---|---|---|
| [7] | **TXEMPTY** | **TX FIFO 空**<br>当 S/w 写任何数据大到 PS2TXDATA0-3， TXEMPTY 位置 0，如果PS2EN 使能. 当传输 数据和 FIFODEPTH 相等， TXEMPTY 将置 1.<br>1 = FIFO 为空<br>0 = 数据传输<br>只读. |
| [6] | **RXOVF** | **RX Buffer 改写**<br>1 =新数据来时， PS2RXDATA 数据改写.<br>0 =无 改写<br>写 1 清该位. |
| [5] | **TXBUSY** | **发送忙**<br>该位表示 PS2 驱动器当前发送数据.<br>0 = 空闲状态.<br>1 = 当前发送数据<br>只读 位. |
| [4] | **RXBUSY** | **接收忙**<br>该位表示 PS2 驱动器当前接收数据.<br>0 =空闲状态.<br>1 = 当前接收数据.<br>只读 位. |
| [3] | **RXPARITY** | **接收奇偶**<br>该位表示最后接收数据奇偶位. (odd parity).<br>只读 位. |
| [2] | **FRAMERR** | **帧错误**<br>应用于主机驱动传输, 如果 STOP 位 (logic 1)未接收到 frame error. 如果 frame error 产生, DATA line 保持低状态 12th clock. 此时, S/w 将无视 PS2CLK 发送时钟 只到 PS2DATA 释放高状态. 然后, 驱动器发送“Resend” 命令到主机.<br>1 = 出现帧错误<br>0 = 未发生帧错误<br>写 1 清该位. |
| [1] | **PS2DATA** | **DATA Pin状态**<br>该位表示同步和取样后，PS2DATA line 状态. |

| | | |
|---|---|---|
| [0] | **PS2CLK** | **CLK Pin 状态**<br>该位表示同步后， PS2CLK line 状态. |

**PS2中断识别寄存器(PS2INTID)**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **PS2INTID** | PS2_BA + 0x1C | R/W | PS2中断识别寄存器 | 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 | | | | | | **PS2DATA** | **PS2CLK** |

| Bits | 描述 | |
|---|---|---|
| [31:3] | 保留 | 保留 |
| [1] | **TXINT** | **传输中断**<br>当 STOP 位传输后，该位置 1. 如果 TXINTEN 位置 1，中断产生.<br>1 = 传输中断发生<br>0 = 无中断<br>写 1 清该位为 0. |
| [0] | **RXINT** | **接收中断**<br>当应答位发送，该位置位.如果 RXINTEN置 1，中断产生.<br>1 = 接收中断发生<br>0 = 无中断<br>写 1 清该位为 0. |

## 5.15 I2S 控制器

### 5.15.1 概述

I2S控制器由IIS协议与外部音频CODEC组成，两个8字的FIFO分别用于读与写通道，可以处理8～32位字大小。DMA控制器处理数据在FIFO与内存之间传输.

### 5.15.2 特征

- 兼容APB接口
- I2S 可工作于8位主机或从机
- 可处理8, 16, 和32 位字大小.
- 支持单声道和立体声的音频数据.
- 支持I2S和MSB 校验数据格式.
- 提供两个8字的FIFO数据缓冲，一个用于发送，一个用于接收.
- 当缓冲超过可编程边界时，产生中断请求.
- 两个DMA请求，一个用于发送，一个用于接收.

### 5.15.3 I2S 框图

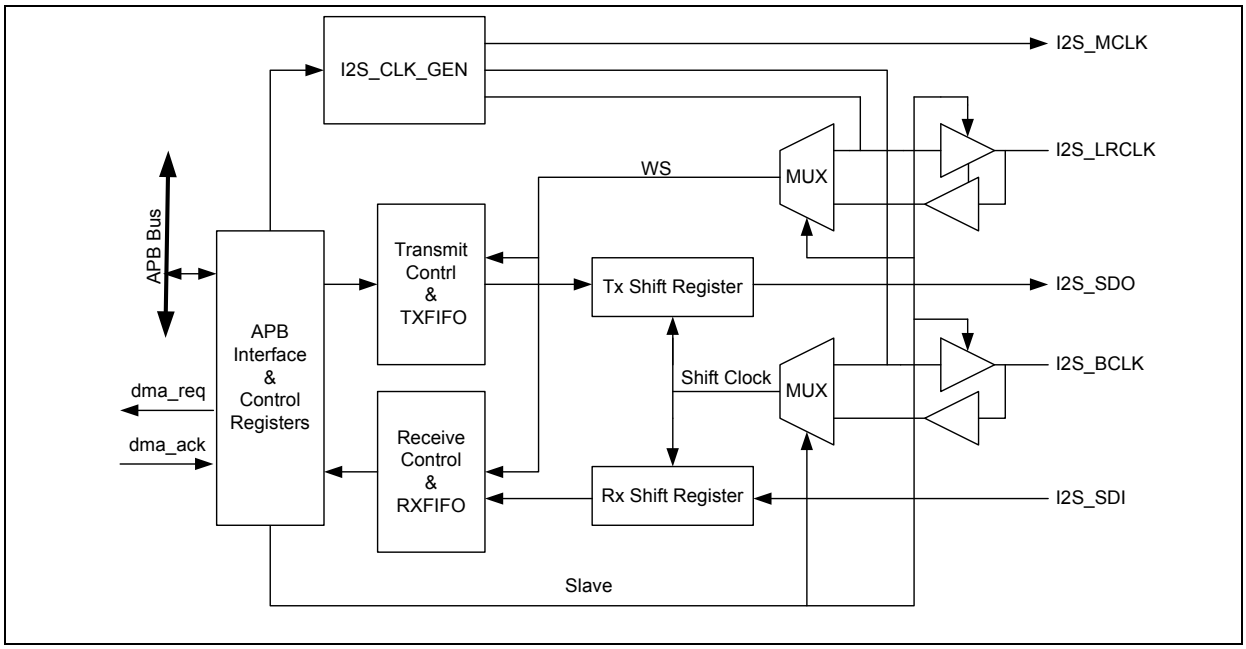



图 5-73 I2S 控制器框图

### 5.15.4 I2S 操作

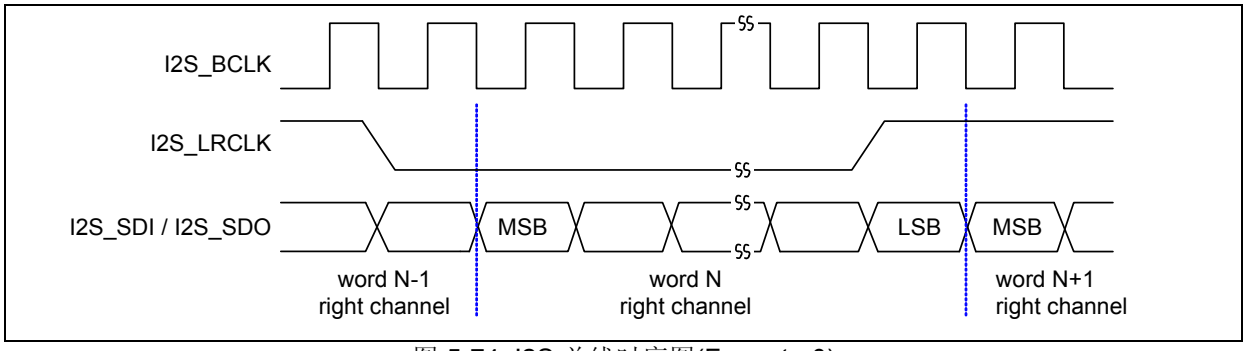



图 5-74 I2S 总线时序图(Format =0)

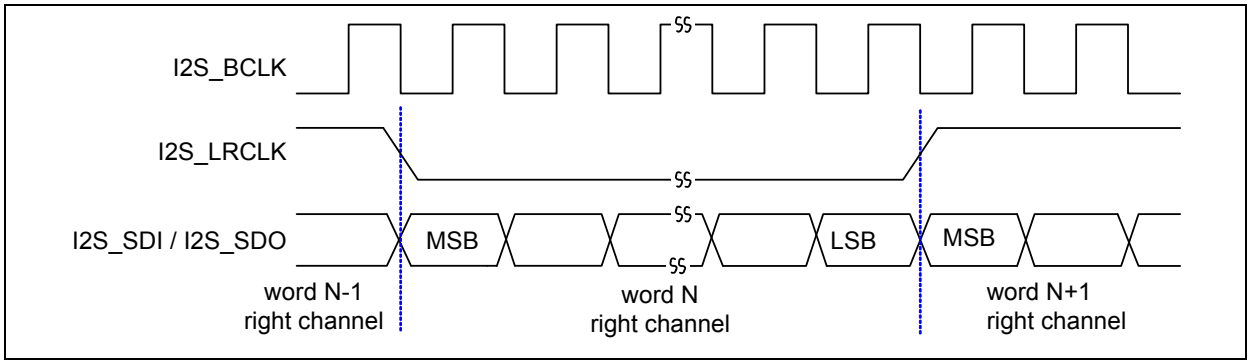



图 5-75 MSB 校验时序图(Format=1)

### 5.15.5 FIFO 操作

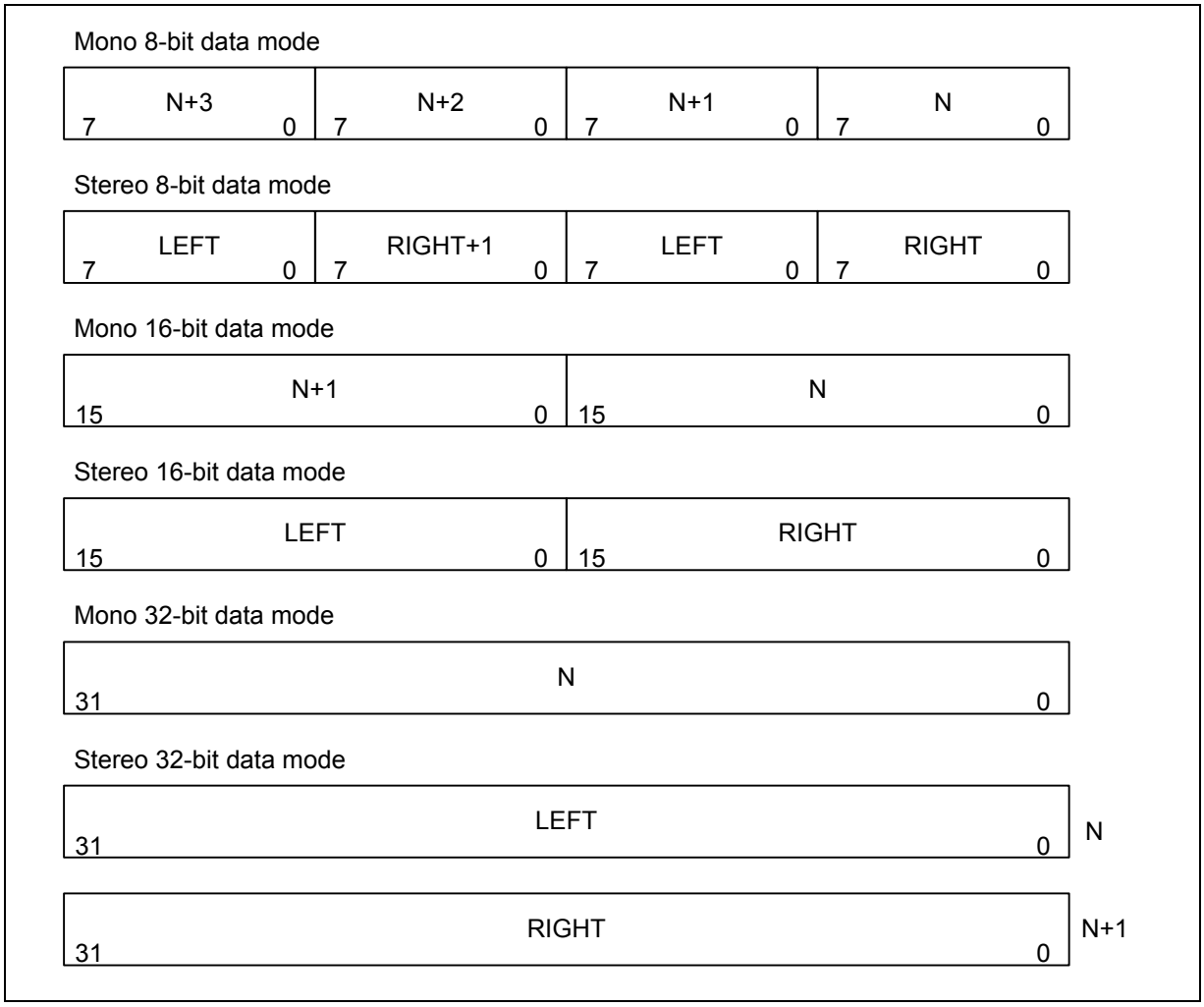



图 5-76 用于可变I2S控制器的FIFO 控制器

### 5.15.6 I2S 控制寄存器表

**R: 只读, W: 只写, R/W: 可读写**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **I2S_BA = 0x401A_0000** | | | | |
| **I2S_CON** | I2S_BA+0x00 | R/W | I2S 控制寄存器 | 0x0000_0000 |
| **I2S_CLKDIV** | I2S_BA+0x04 | R/W | I2S 时钟分频寄存器 | 0x0000_0000 |
| **I2S_IE** | I2S_BA+0x08 | R/W | I2S 中断使能寄存器 | 0x0000_0000 |
| **I2S_STATUS** | I2S_BA+0x0C | R/W | I2S 状态寄存器 | 0x0000_0000 |
| **I2S_TXFIFO** | I2S_BA+0x10 | R/W | I2S 发送FIFO寄存器 | 0x0000_0000 |
| **I2S_RXFIFO** | I2S_BA+0x14 | R/W | I2S 接收FIFO寄存器 | 0x0000_0000 |

### 5.15.7 I2S 控制寄存器描述

**I2S 控制寄存器 (I2S_CON)**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| I2S_CON | I2S_BA + 0x00 | R/W | I2S 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 | 保留 | RXDMA | TXDMA | CLR_RXFIFO | CLR_TXFIFO | LCHZCEN | RCHZCEN |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| MCLKEN | RXTH[2:0] | | | TXTH[2:0] | | | SLAVE |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| FORMAT | MONO | WORDWIDTH[1:0] | | MUTE | RXEN | TXEN | I2SEN |

| Bits | 描述 | |
|---|---|---|
| [31:22] | 保留 | 保留 |
| [21] | **RXDMA** | **使能 DMA 接收**<br>当使能 RX DMA, 如果FIFO没空，I2S 请求DMA从接收FIFO向SRAM传输数据.<br>1 = 使能 RX DMA<br>0 = 禁止 RX DMA |
| [20] | **TXDMA** | **使能 DMA 发送**<br>当使史能 TX DMA, 当FIFO没空，I2S 请求DMA 从SRAM向发送FIFO传输数据.<br>1 = Enable TX DMA<br>0 = Disable TX DMA |
| [19] | **CLR_RXFIFO** | **清接收 FIFO**<br>写“1”清接收FIFO,内部指针重置FIFO到开始指针，RXFIFO_LEVEL[3:0] 返回0，接收FIFO为空.<br>该位由硬件自动清零，读时返回0. |

| | | |
|---|---|---|
| [18] | **CLR_TXFIFO** | **清发送 FIFO**<br>写“1”清发送FIFO, 内部指针重置FIFO开始指针， TXFIFO_LEVEL[3:0] 返回0，发送FIFO 为空，但在发送FIFO中的数据不变.<br>该位由硬件自动清零，读时返回0. |
| [17] | **LCHZCEN** | **使能左声道过零检测**<br>如果该位置 1，当左声道数据符号位改变或下一个移动数据位都为 0，则寄存器 I2S_STATUS 的 LZCF 标志置 1<br>1 = 使能左声道过零检测<br>0 = 禁止左声道过零检测 |
| [16] | **RCHZCEN** | **使能右声道过零检测**<br>如果该位置 1，当右声道数据符号位改变或下一个移动数据位全为 0，则寄存器 I2S_STATUS 的 LZCF 标志置 1.<br>1 = 使能右声道过零检测<br>0 = 禁止右声道过零检测 |
| [15] | **MCLKEN** | **使能主时钟**<br>如果NUC100 外部晶振时钟频率为2*N*256fs, 由软件可编程寄存器I2S_CLKDIV 的MCLK_DIV[2:0]位，以得到音频CODEC所需的256fs的时钟.<br>1 = 使能主时钟<br>0 = 禁止主时钟 |
| [14:12] | **RXTH[2:0]** | **接收 FIFO 的阈值水平**<br>当缓冲里的接收数据字等于或大于阈值水平时，RXTHI标志置位.<br>000 =FIFO接收0个字数据<br>001 = FIFO接收1个字数据<br>010 = FIFO接收2个字数据<br>011 = FIFO接收3个字数据<br>100 = FIFO接收4个字数据<br>101 = FIFO接收5个字数据<br>110 = FIFO接收6个字数据<br>111 = FIFO接收7个字数据 |

| | | |
|---|---|---|
| [11:9] | **TXTH[2:0]** | **发送 FIFO 的阈值水平**<br>如果保存在发送FIFO的数据字（32位）等于或小于阈值水平，TXTHI标志置位.<br>000 =发送FIFO中有0个字数据<br>001 =发送FIFO中有1个字数据<br>010 =发送FIFO中有2个字数据<br>011 =发送FIFO中有3个字数据<br>100 =发送FIFO中有4个字数据<br>101 =发送FIFO中有5个字数据<br>110 =发送FIFO中有6个字数据<br>111 =发送FIFO中有7个字数据 |
| [8] | **SLAVE** | **从机模式**<br>I2S 可以工作于主机模式或从机模式. 主机械下，I2S_BCLK 与I2S_LRCLK都作为由NUC100到音频CODEC芯片的输出模式和发送位时钟，在从机模式，I2S_BCLK 与I2S_LRCLK 都为输入模式，I2S_BCLK 与 I2S_LRCLK信号从外部音频CODEC芯片接收.<br>1 = 从机模式<br>0 = 主机模式 |
| [7] | **FORMAT** | **数据模式**<br>1 = MSB 检验数据格式<br>0 = I2S 数据模式 |
| [6] | **MONO** | **单声道数据**<br>1 = 单声道数据格式<br>0 = 立体声数据格式 |
| [5:4] | **WORDWIDTH[1:0]** | **字宽度**<br>00 = data is 8 bit<br>11 = data is 16 bit<br>10 = data is 24 bit<br>11 = data is 32 bit |
| [3] | **MUTE** | **使能发送静音**<br>1=发送通道0<br>0 = 发送数据由缓冲移动 |
| [2] | **RXEN** | **接收使能**<br>1 = 使能数据接收<br>0 = 禁止数据接收 |

| | | |
|---|---|---|
| [1] | **TXEN** | **发送使能**<br>1 = 使能数据发送<br>0 = 禁止数据发送 |
| [0] | **I2SEN** | **使能 I2S 控制器**<br>1 = 使能<br>0 = 禁止 |

### I2S 时钟分频(I2S_CLKDIV)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **I2S_CLKDIV** | I2S_BA + 0x04 | R/W | I2S 时钟分频控制寄存器 | 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** |
| BCLK_DIV [7:0] | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| 保留 | | | | | MCLK_DIV[2:0] | | |

| Bits | 描述 | |
|---|---|---|
| [31:16] | 保留 | 保留 |
| [15:8] | **BCLK_DIV [7:0]** | **位时钟分频**<br>如果I2S 工作于主机模式，NUC100提供位时钟。软件可以编程这些位产生采样时钟频率.<br>*F_BCLK = F_I2SCLK /(2x(BCLK_DIV + 1))* |
| [7:3] | 保留 | 保留 |
| [2:0] | **MCLK_DIV[2:0]** | **主时钟分频**<br>如果NUC100的外部晶振频率(2xMCLK_DIV)x256fs ， 则软件可编程这些位产生256fs时钟频率到音频CODEC芯片. 如果MCLK_DIV设置为0， MCLK 与外部时钟输入相同.<br>如：采样率为48khz , NUC100 的外部晶振时钟为12.288MHz, 设置 MCLK_DIV=0.<br>*F_MCLK = F_I2SCLK/(2x(MCLK_DIV))* |

### I2S 中断使能寄存器(I2S_IE)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| I2S_IE | I2S_BA + 0x08 | R/W | I2S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 | | | LZCIE | RZCIE | TXTHIE | TXOVFIE | TXUDFIE |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | | | | RXTHIE | RXOVFIE | RXUDFIE |

| Bits | 描述 | |
|---|---|---|
| [31:13] | 保留 | 保留 |
| [12] | **LZCIE** | **左声道过零检测中断使能**<br>如果该位设为1，左声道有过零信号，产生中断<br>1 = 使能中断<br>0 = 禁止中断 |
| [11] | **RZCIE** | **右声道过零检测中断使能**<br>1 = 使能中断<br>0 = 禁止中断 |
| [10] | **TXTHIE** | **发送 FIFO 阈值水平中断使能**<br>如果该位设置为1，发送FIFO中的数据少于TXTH[2:0],产生中断.<br>1 = 使能中断<br>0 = 禁止中断 |
| [9] | **TXOVFIE** | **发送 FIFO 溢出中断使能**<br>如果该位设置为1，发送FIFO溢出标志设置为1，发生中断<br>1 = 使能中断<br>0 = 禁止中断 |

| | | |
|---|---|---|
| [8] | **TXUDFIE** | **发送 FIFO 下溢中断使能**<br>如果该位设置为1，发送FIFO下溢标志设置为1时，中断发生.<br>1 = 使能中断<br>0 = 禁止中断 |
| [7:3] | 保留 | 保留 |
| [2] | **RXTHIE** | **接收 FIFO 阈值水平中断**<br>当接收FIFO的数据字等于或大于RXTH[2:0] 和 RXTHI 位设置为1。如果RXTHIE位使能，中断发生.<br>1 = 使能中断<br>0 = 禁止中断 |
| [1] | **RXOVFIE** | **接收 FIFO 溢出中断使能**<br>1 = 使能中断<br>0 = 禁止中断 |
| [0] | **RXUDFIE** | **接收 FIFO 下溢中断使能**<br>如果软件读接收FIFO，当为空时，寄存器I2SSTATUS 的RXUDF 标志设置为1.<br>1 = 使能中断<br>0 = 禁止中断 |

### I2S 状态 寄存器(I2S_STATUS)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| I2S_STATUS | I2S_BA + 0x0C | R/W | I2S 状态寄存器 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| TX_LEVEL[3:0] | | | | RX_LEVEL[3:0] | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| LZCF | RZCF | TXBUSY | TXEMPTY | TXFULL | TXTHF | TXOVF | TXUDF |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| 保留 | | | RXEMPTY | RXFULL | RXTHF | RXOVF | RXUDF |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | | | RIGHT | I2SRXINT | I2STXINT | I2SINT |

| Bits | 描述 | |
|---|---|---|
| [31:28] | **TX_LEVEL** | **发送 FIFO 水平**<br>这些位表示在发送FIFO中数据字的数目<br>0000 = 无数据<br>0001 = 在发送 FIFO有1个字<br>….<br>1000 = 在发送 FIFO有81个 |
| [27:24] | **RX_LEVEL** | **接收 FIFO 水平**<br>**这些位表示在接收 FIFO 中数据字的数目**<br>0000 = 无数据<br>0001 = 在接收FIFO中有1个字<br>….<br>1000 =在接收FIFO中有8个字 |
| [23] | **LZCF** | **左声道过零标志**<br>表示左声道下一个采样数据符号位已经改变或所有数据位为0.<br>1 = 左声道过零检测<br>0 = 没有过零<br>该位只读. |

| | | |
|---|---|---|
| [22] | **RZCF** | **右声道过零标志**<br>表示右声道下一个采样数据符号位已经改变或所有数据位为0.<br>1 = 右声道过零检测<br>0 = 没有过零<br>该位只读. |
| [21] | **TXBUSY** | **发送忙**<br>在所有数据在发送FIFO中时，移动缓冲移出的时候，该位清零，当第一个数据加载到移位缓冲时，该位置1<br>1 = 发送移位缓冲忙<br>0 = 发送移位缓冲空<br>该位只读. |
| [20] | **TXEMPTY** | **发送 FIFO 空**<br>该位反映发送FIFO中的数据字号是0<br>0 = 空<br>1 = 非空<br>该位只读. |
| [19] | **TXFULL** | **发送 FIFO 满**<br>该位反映发送FIFO中数据字号是8<br>0 = 满.<br>1 = 不满.<br>该位只读 |
| [18] | **TXTHF** | **发送 FIFO 开始标志**<br>当发送FIFO中的数据字等于或低于TXTH[2:0]中所设的开始值， TXTHF 变成1. 并保持为1直到在软件写寄存器TXFIFO使TXFIFO_LEVEL[3:0] 高于TXTH[1:0].<br>1 = FIFO中的数据字等于或低于阈值水平<br>0 = FIFO中的数据字高于阈值水平<br>该位只读 |
| [17] | **TXOVF** | **发 送 FIFO 溢出标志**<br>当发送FIFO满时且该位置1，向发送FIFO中写数据<br>1 = 溢出<br>0 = 没有溢出<br>软件写1清该位 |

| | | |
|---|---|---|
| [16] | **TXUDF** | **发送 FIFO 下溢标志**<br>发送FIFO为空，且移位逻辑硬件从数据FIFO读数据，引起该位置1.<br>1 = 下溢<br>0 = 无下溢<br>软件写1清该位 |
| [15:13] | 保留 | 保留 |
| [12] | **RXEMPTY** | **接收 FIFO 空**<br>该位反映接收FIFO中的数据字号为0<br>1 = 空<br>0 = 非空<br>该位只读. |
| [11] | **RXFULL** | **接收 FIFO 空**<br>该位反映接收FIFO中的数据字号为8<br>1 = 满<br>0 = 不满<br>该位只读. |
| [10] | **RXTHF** | **接收 FIFO 开始标志**<br>接收FIFO中的数据字等于或高于RXTH[2:0]中所设定的开始值， RXTHF 位变为1，在软件读RXFIFO 寄存器时，保持为1走到RXFIFO_LEVEL[3:0] 低于 RXTH[1:0]<br>1 = FIFO中的数据字等于或高于阈值水平<br>0 = FIFO中的数据字低于阈值水平<br>该位只读 |
| [9] | **RXOVF** | **接收 FIFO 溢出标志**<br>当接收FIFO满且接收硬件试图写接收FIFO写数据，该位置1，第一个缓冲数据被覆盖.<br>1 = 发生溢出<br>0 = 没有发生溢出<br>该位写 1 清零 |
| [8] | **RXUDF** | **接收 FIFO 下溢标志**<br>当为空时，读接收FIFO，该位置1，表示发生下溢.<br>1 = 发生下溢<br>0 = 没有 发生下溢<br>写 1 清该位 |
| [7:4] | 保留 | 保留 |

| | | |
|---|---|---|
| [3] | **RIGHT** | **右声道**<br>该位表示当前发送数据属于右声道<br>1 = 右声道<br>0 = 左声道<br>该位只读 |
| [2] | **I2SRXINT** | **I2S 接收中断**<br>1 = 接收中断<br>0 = 没 有接收中断<br>该位只读 |
| [1] | **I2STXINT** | **I2S 发送中断**<br>1 = 发送中断<br>0 = 没有发送中断<br>该位只读 |
| [0] | **I2SINT** | **I2S 中断标志**<br>1 = I2S 中断<br>0 =没有I2S 中断<br>I2STXINT 与I2SRXINT 的“或”<br>该位只读. |

### I2S 发送FIFO (I2S_TXFIFO)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| I2S_TXFIFO | I2S_BA + 0x10 | R/W | I2S 发送FIFO | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:0] | TXFIFO | **发送 FIFO 寄存器**<br>I2S 包括8字（8x32 bit）的数据缓冲用于数据发送，向该寄存器写数据以准备发送数据，余下的数字号由寄存器I2S_STATUS 中的TX_LEVEL[3:0]定义 |

### I2S 接收FIFO (I2S_RXFIFO)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **I2S_RXFIFO** | I2S_BA + 0x14 | R/W | I2S 接收 FIFO | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:0] | **RXFIFO** | **接收 FIFO 寄存器**<br>I2S包括8字 (8x32 bit)数据缓冲用于数据接收. 读该寄存器得到FIFO中的数据，余下的数字号在寄存器I2S_STATUS中的 RX_LEVEL[3:0]定义. |

## 5.16 模拟数字转换(ADC)

### 5.16.1 功能概述

NUC1XX 系列包含 一个12-bit 8通道逐次逼近式 模拟 – 数字转换器 (SAR A/D converter). A/D 转换器支持 三种操作模式: 单一, 单周期扫描 和连续扫描模式. 2 种扫描模式: 单周期扫描 和连续扫描模式. 开始A/D 转换可软件设定和外部STADC/PB.8 pin.

注：使能 ADC功能前，模拟输入引脚输入类型.

### 5.16.2 特征

- 模拟输入电压: 0~Vref (Max to 5.0V).
- 12-bits resolution 和 10-bits 精确度保证.
- 多达 8 路单端输入通道或4路差分输入.
- 最大 ADC 时钟频率 20MHz.
- 高达1M SPS 转换速率, 转换时间少于 1us.
- 三种运作模式
  - 单一模式: 单通道 A/D 转换
  - 单周期扫描模式: 使能通道连续转换
  - 连续扫描模式: 使能通道重复转换
- A/D 转换开始条件
  - 软件置ADST 位
  - 外部 pin STADC
- 每通道转换结果存储在数据寄存器内，并带有valid/overrun标志。
- 转换结果可和指定的值相比较 当转换值和设定值相匹配时，用户设定产生中断请求.
- 通道 7 支持 3 输入源：外部模拟电压, 内部规定电压和内部温度传感器输出.
- 支持自身校正功能 减少转换的误差.

### 5.16.3 框图

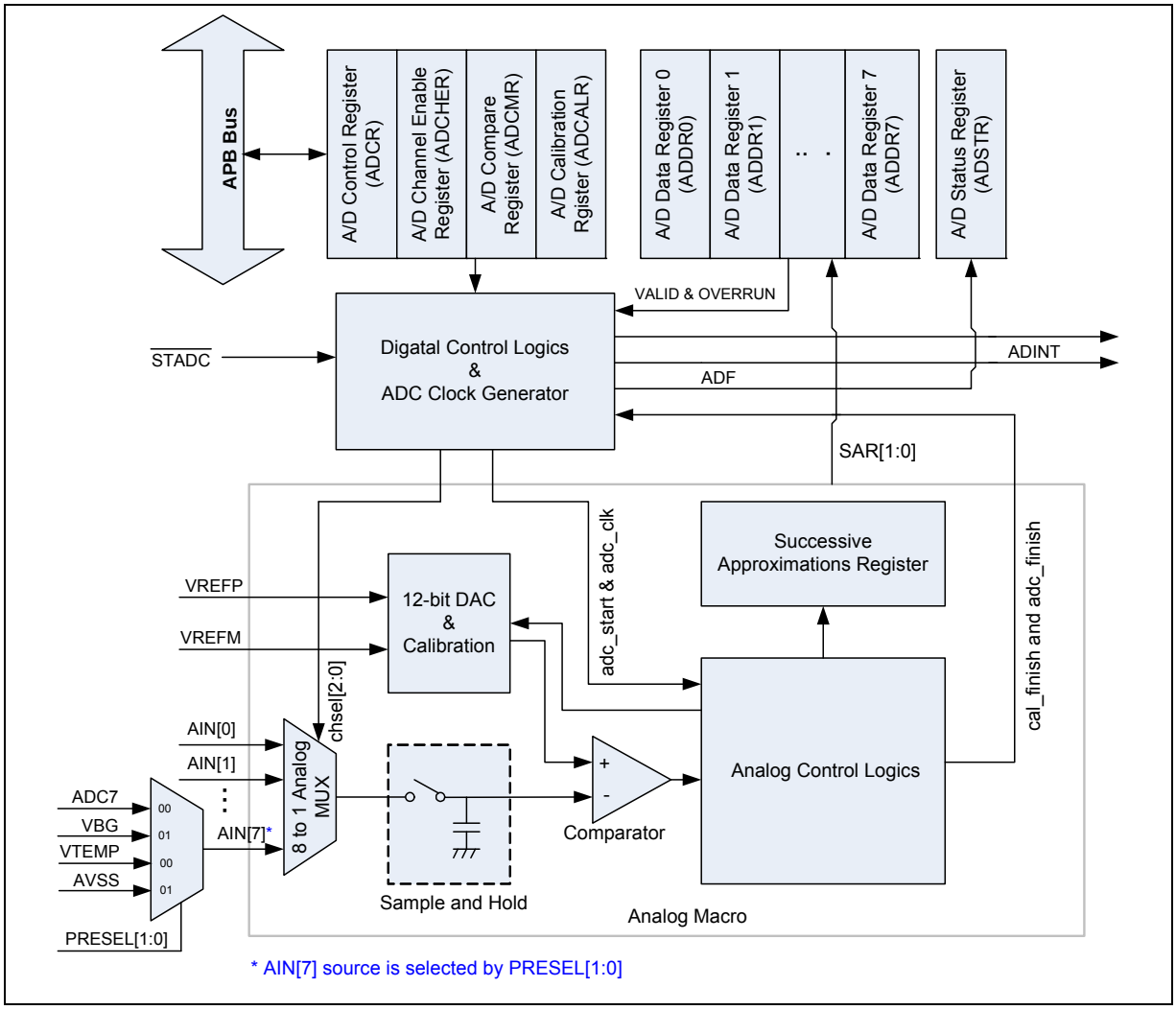



图 5-77 ADC 控制器框图

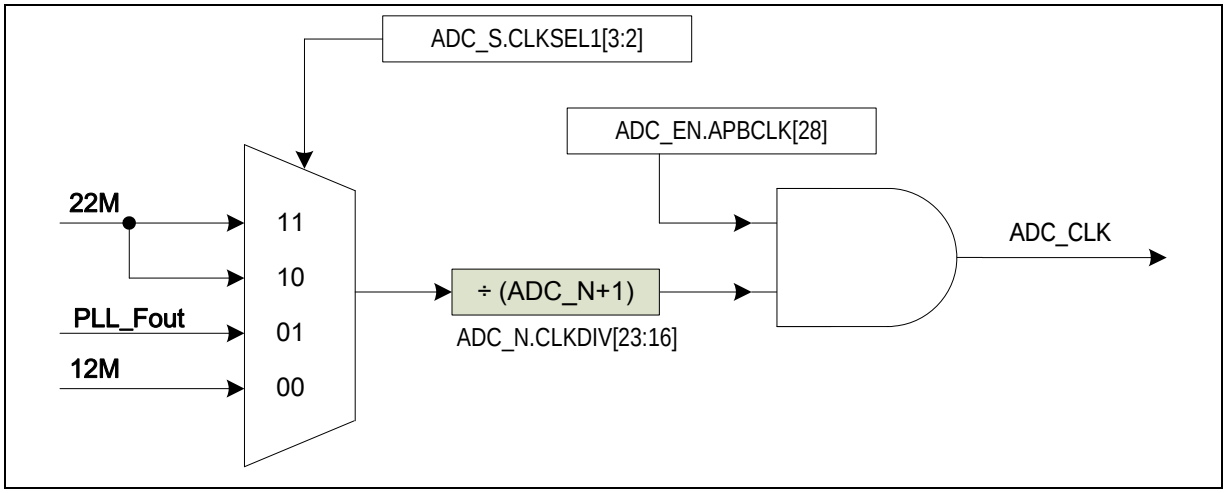



图 5-78 ADC 时钟控制

### 5.16.4 操作步骤

A/D 转换器为 12-bit 逐次逼近式. A/D 具有自身校正功能 减少转换的误差, 用户可写 1 到 CALEN 位（ ADCALR 寄存器） 使能自身校正功能, 当内部校正完成 CAL_DONE 为高. ADC 具有 2 种操作模式: 单一模式和扫描模式. 两类扫描模式: 连续扫描模式和单周期模式. 当改变运行模式或模拟输入通道使能时, 为了防止错误的操作, 软件需清 ADST 位为 0 (ADCR register). A/D 转换将停止 并进入 idle 模式.

#### *5.16.4.1 自校正*

用户置位 CALEN位( ADCALR 寄存器) 使能自身校正功能.运作过程需要127 ADC 时钟完成校正. CALEN 置位后, 软件需等待 CAL_DONE 位设定通过内部硬件. 时序图如下:

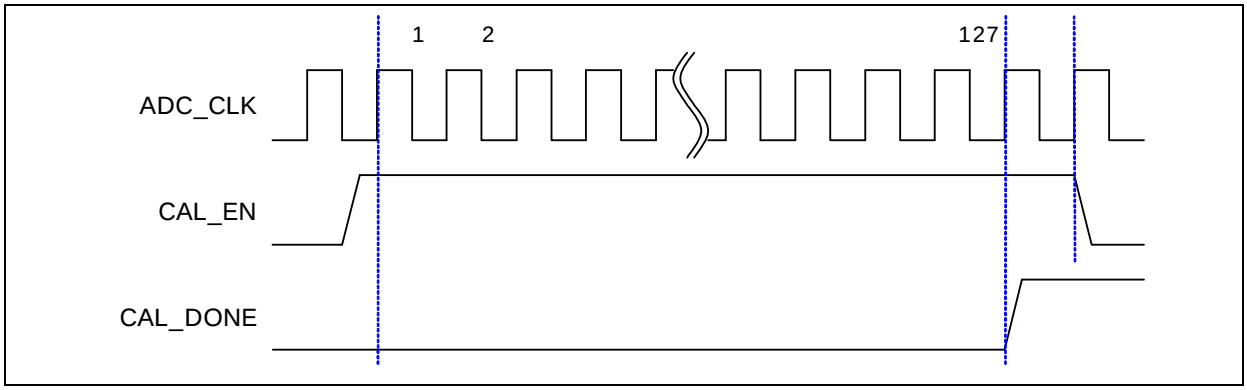



图 5-79 ADC 转换器自身校正时序图

#### *5.16.4.2 ADC时钟发生器*

最大采样率达800KHz 转换时间小于1.25us. 一次A/D转换需要20个ADC时钟周期完成. ADC有三个时钟源，可由ADC_S(CLKSEL[3:2])选择, ADC 时钟频率按如下公式进行8位预分频:

ADC时钟频率= (ADC clock source frequency) / (ADC_N+1);
8-bit ADC_N 在寄存器CLKDIV[23:16]中设置.

通常来说，软件设置ADC_S 与 ADC_N 获得 16MHZ 或稍低于16MHZ.

#### *5.16.4.3 单触发模式*

在单触发模式下, A/D 转换遵循单通道模式，运作流程如下:

#### *5.16.4.4 单触发模式*

在单触发模式下, A/D 转换遵循单通道模式，运作流程如下:

1. 当 ADCR 的ADST 置位开始A/D 转换，可通过软件或外部触发输入.
2. 当 A/D 转换完成, A/D 转换的数据值将传递给相应通道.
3. A/D 转换完成, ADSR 的ADF 位置位. 若此时ADIE 位置位, 将产生ADINT 中断请求.
4. ADST 位为高 当A/D 转换时. A/D 转换结束, ADST 位自动清 0 ， A/D 转换器进入 idle 模式. ADST 清 0 当 A/D 转换时, A/D 转换将自动停止 进入 idle 模式.

**注:** 在单触发模式时，如果软件使能多于一个通道，最小通道被转换，其他通道被忽略.

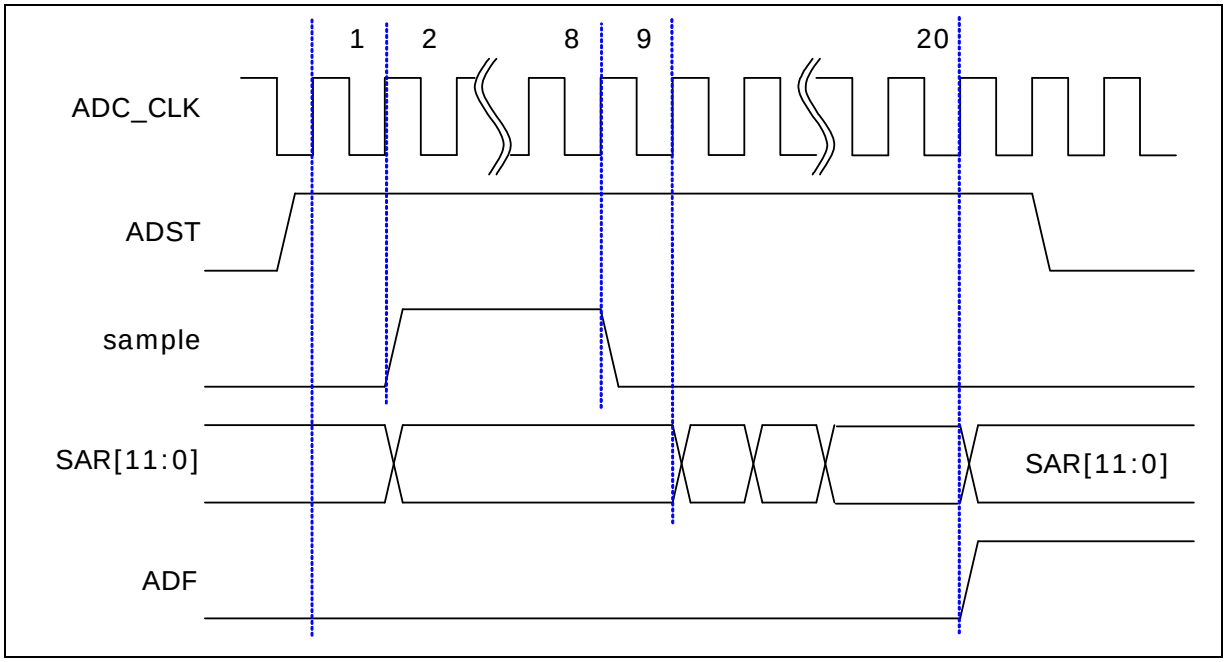



图 5-80 单一转换模式时序图

*5.16.4.5 连续转换模式*

在连续模式下, 可通过设定CHEN 位 ADCHER 寄存器 (maximum 8 channels for ADC) 使能 A/D 转换. 运作流程如下:

1. 软件置位 ADCR 寄存器的 ADST 位或外部触发输入, 开始 最小数目的 A/D 转换.
2. 每路 A/D 转换完成后, A/D 转换数值将装载到相应数据寄存器中.
3. 当被选择的通道数 都转换完成后， ADF 位 （ADSR 寄存器）置位. 若此时 ADIE 置位, 将产生 ADINT 中断请求.当 A/D 转换完成后. 使能的 1st 通道将开始新的转换.
4. ADST 一直保持 1，步骤2 ~ 3 执行. 当 ADST 位清 0, A/D 转换将停止 A/D 转换器 进入 idle 模式.

使能通道(0, 2, 3 and 7) 连续模式时序图如下:

(这个例子仅适用于ADC)

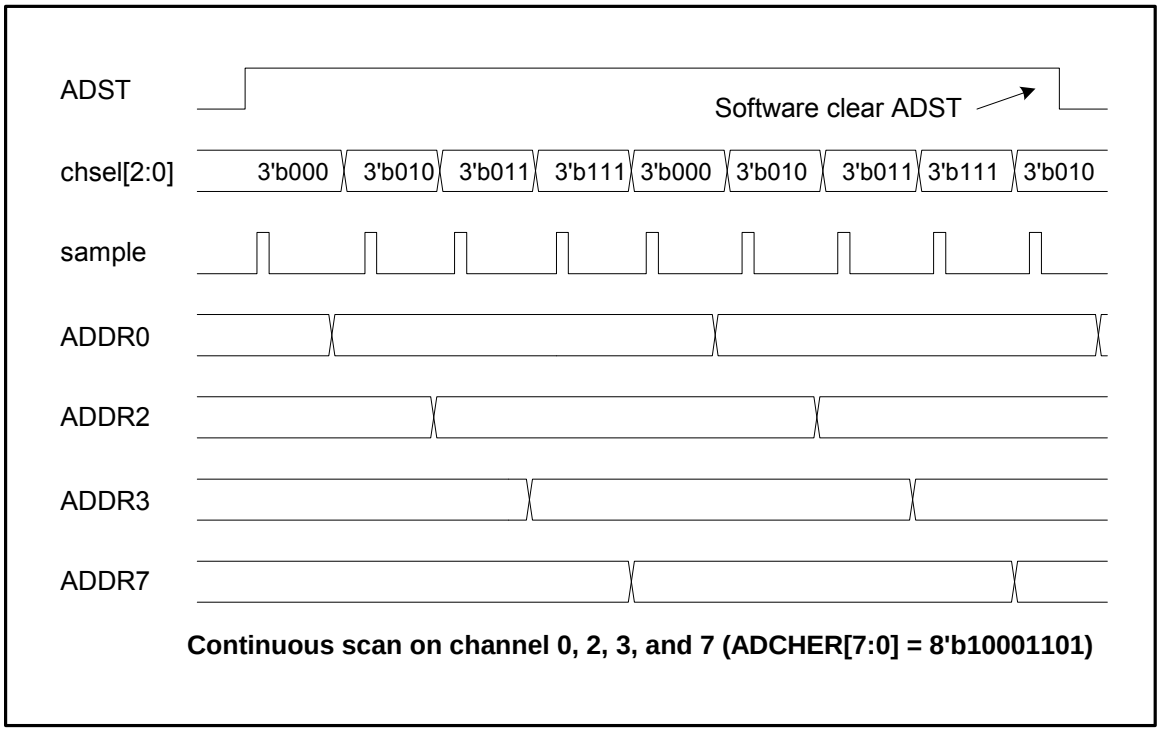



图 5-81 连续扫描下使能通道时序图

*5.16.4.6 单周期扫描模式*

在单周期模式下， A/D 将从最小通道向最大通道 进行转换，具体流程如下：

1. 软件置位 ADCR 寄存器的 ADST 位或外部触发输入, 开始 最小数目的 A/D 转换..
2. 每路 A/D 转换完成后, A/D 转换数值将装载到相应数据寄存器中..
3. 当被选择的通道数 都转换完成后， ADF 位 （ADSR 寄存器）置位. 若此时 ADIE 置位, 将产生 ADINT 中断请求.当 A/D 转换完成后. 使能的 1st 通道将开始新的转换.
4. A/D 转换结束, ADST 位自动清 0 ，A/D 转换器进入 idle 模式. ADST 清 0 当 A/D 转换时, A/D 转换将自动停止 进入 idle 模式.

使能通道(0, 2, 3 and 7) 单周期模式时序图如下:

(这个例子仅适用于 ADC)

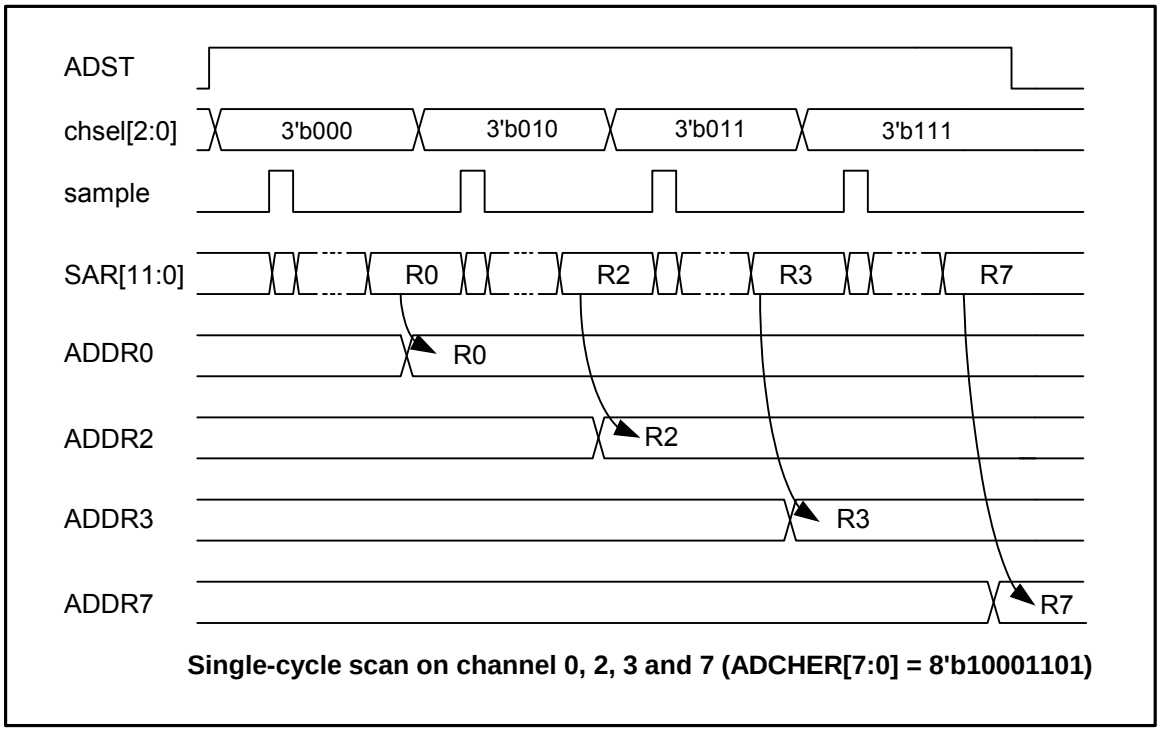



图 5-82 单周期扫描下使能通道时序图

*5.16.4.7 输入取样和 A/D 转换时间*

当 ADCR 寄存器 ADST 位置位后，A/D 开始转换延时 (Td), 而后开始转换. 由于 ADC 时钟通过 PCLK 位进行 (N+1)分频, 最大延时时间从 APB 写 A/D 开始模拟输入取样时间为 2N PCLKs. 延时 时间如下:

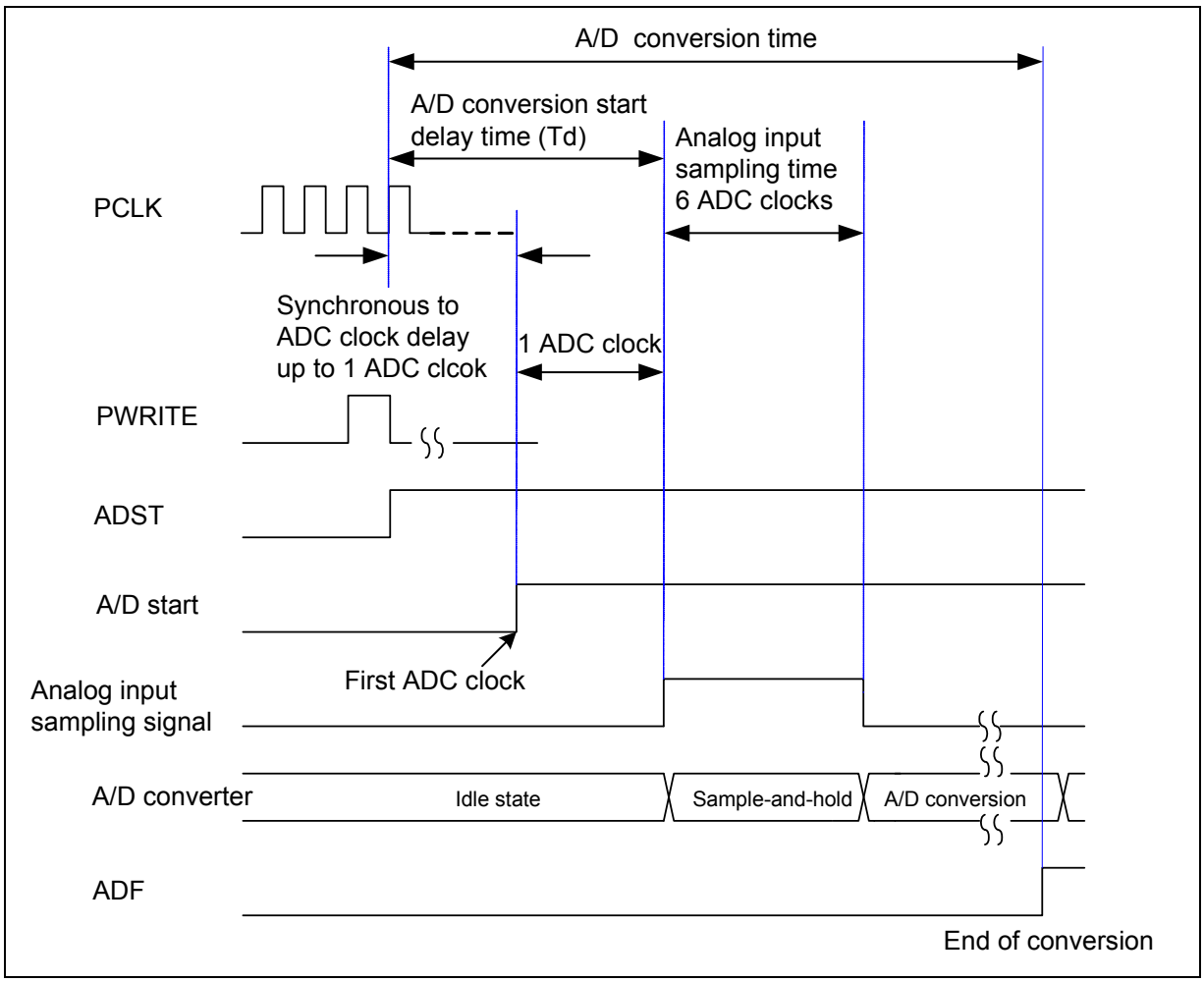



图 5-83 转换开始延时 时序框图

A/D 转换可通过外部 pin 脚触发请求.当 ADCR.TRGEN 置位 使能 ADC 外部触发功能, 通过设定 TRGS[1:0] 位为 00b 从 STADC pin 选择外部触发引脚输入. 软件设定 TRGCOND[1:0] 选择 **上升/下降沿 或低/高** 触发. 8-bit 取样计数器应用于抗尖峰脉冲. 若选择 level trigger 条件, STADC 需保持默认状态至少 8 个PCLKs. ADST 位置位 1 在 9$^{th}$ PCLK ， 开始新的转换.外部触发输入状态为pull at low (or high state) level trigger模式状态下 进行连续转换. 仅当外部触发条件消失 才停止. 若选择边缘触发模式，thigh and low 状态至少需保持 4 PLCKs. 其它操作模式, 单一和扫描模式, 同 ADST 置位状态

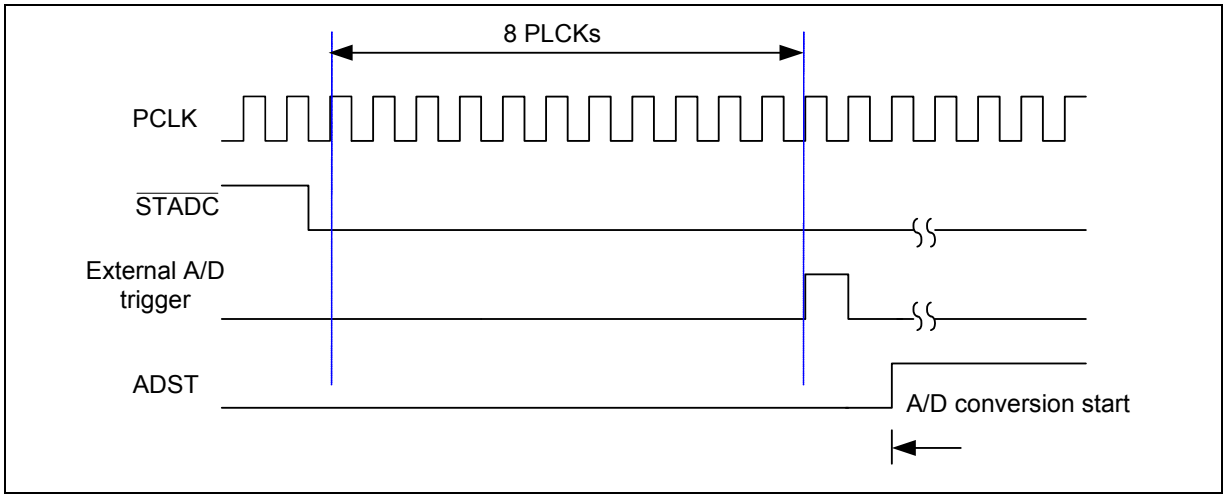



图 5-84 外部 A/D 转换触发时序框图

### *5.16.4.8 比较模式下AD转换结果监控*

NUC1XX 系列提供比较寄存器 ADCMPR0 和 1 监控 A/D 转换模块的 2路转换结果值, 可参考 图 5-85. 可通过软件设定CMPCH[2:0] 和 CMPCOND 位 监控选定的通道，检查转换置小于或大于 等于 CMPD[11:0]指定值. 当比较结果和设定值相匹配，比较计数器将加 1，当计数器的值和设定值 CMPMATCNT位匹配 CMPF 位将置 1, 如果CMPIE 置位 将残生 ADINT 中断请求. 在扫描模式下 软件可应用于监控外部模拟输入 pin 脚电压变化.具体逻辑框图如下:

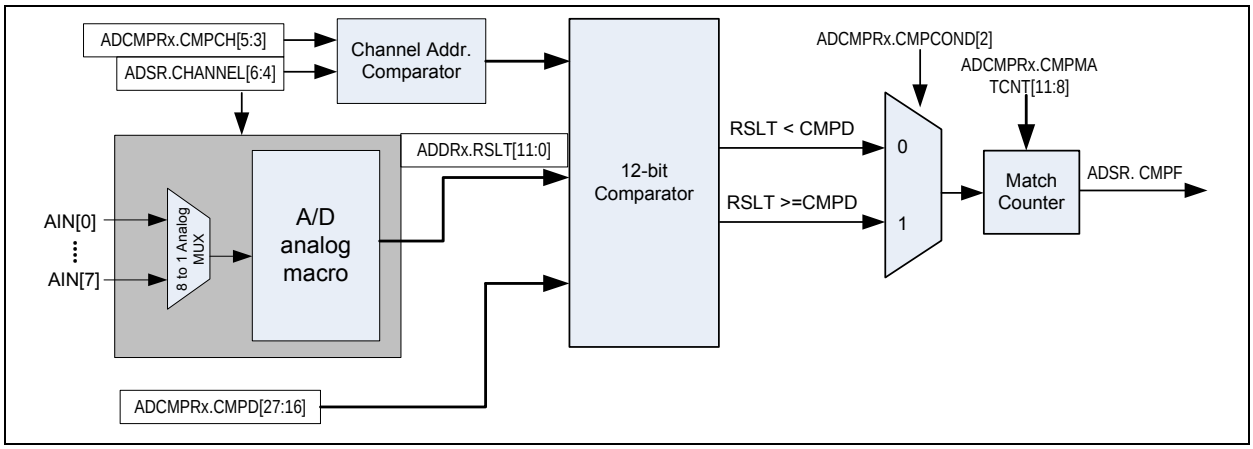



图 5-85 A/D 转换结果监控框图

### *5.16.4.9 中断源*

A/D 转换结束时产生ADF（ADSR寄存器）. 若 ADIE 位（ADCR 寄存器）置位，将产生 ADINT 中断请求.若 CMPIE位 使能, 当 A/D 转换结果同 ADCMPR 寄存器设定值相匹配, 将产生 ADINT 中断请求系统可清 CMPF 和 ADF 位 禁止中断请求.

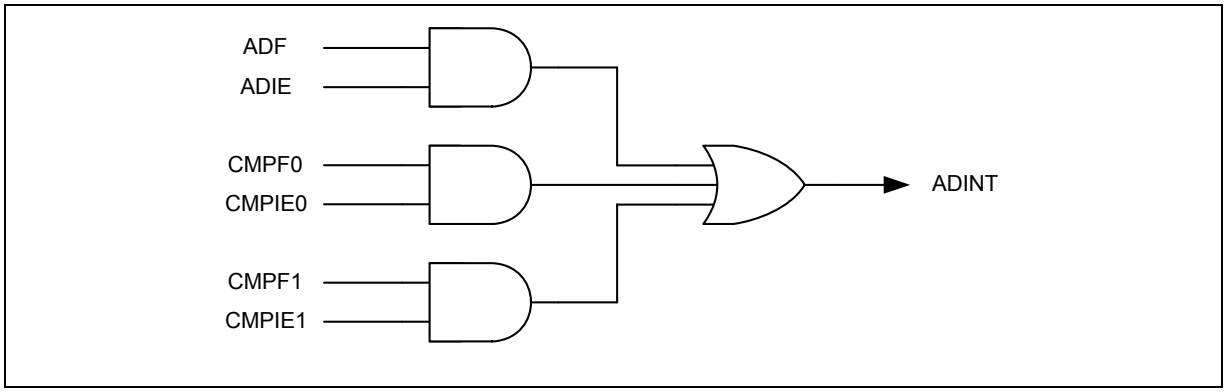



图 5-86 A/D 控制器中断

*5.16.4.10 外设DMA 请求*

当 A/D 转换完成 结果装载到 ADDR 寄存器 VALID 置位. 如果 ACDR 寄存器 PTEN 位置位, ADC 控制器将产生 PDMA 请求 (P_nDRQ) 以便数据传输. 所有转换结果可通过读 PDMA 而无需干挠 CPU ,保证系统可进行连续的转换.

### 5.16.5 ADC寄存器列表

**R: 只读, W: 只写, R/W: 可读写，C:仅在值为0时可写**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **ADC_BA = 0x400E_0000** | | | | |
| **ADDR0** | ADC_BA+0x00 | R | A/D数据寄存器 0 | 0x0000_0000 |
| **ADDR1** | ADC_BA+0x04 | R | A/D数据寄存器 1 | 0x0000_0000 |
| **ADDR2** | ADC_BA+0x08 | R | A/D数据寄存器 2 | 0x0000_0000 |
| **ADDR3** | ADC_BA+0x0C | R | A/D数据寄存器 3 | 0x0000_0000 |
| **ADDR4** | ADC_BA+0x10 | R | A/D数据寄存器 4 | 0x0000_0000 |
| **ADDR5** | ADC_BA+0x14 | R | A/D数据寄存器 5 | 0x0000_0000 |
| **ADDR6** | ADC_BA+0x18 | R | A/D数据寄存器 6 | 0x0000_0000 |
| **ADDR7** | ADC_BA+0x1C | R | A/D数据寄存器 7 | 0x0000_0000 |
| **ADCR** | ADC_BA+0x20 | R/W | A/D控制寄存器 | 0x0000_0000 |
| **ADCHER** | ADC_BA+0x24 | R/W | A/D 通道使能寄存器 | 0x0000_0000 |
| **ADCMPR0** | ADC_BA+0x28 | R/W | A/D 比较寄存器0 | 0x0000_0000 |
| **ADCMPR1** | ADC_BA+0x2C | R/W | A/D比较寄存器1 | 0x0000_0000 |
| **ADSR** | ADC_BA+0x30 | R/W | A/D 状态寄存器 | 0x0000_0000 |
| **ADCALR** | ADC_BA+0x34 | R/W | A/D 校准寄存器 | 0x0000_0000 |

### 5.16.6 ADC寄存器描述

**A/D数据寄存器(ADDR0 ~ ADDR7)**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **ADDR0** | ADC_BA+0x00 | R | A/D数据寄存器 0 | 0x0000_0000 |
| **ADDR1** | ADC_BA+0x04 | R | A/D数据寄存器 1 | 0x0000_0000 |
| **ADDR2** | ADC_BA+0x08 | R | A/D数据寄存器 2 | 0x0000_0000 |
| **ADDR3** | ADC_BA+0x0c | R | A/D数据寄存器 3 | 0x0000_0000 |
| **ADDR4** | ADC_BA+0x10 | R | A/D数据寄存器 4 | 0x0000_0000 |
| **ADDR5** | ADC_BA+0x14 | R | A/D数据寄存器 5 | 0x0000_0000 |
| **ADDR6** | ADC_BA+0x18 | R | A/D数据寄存器 6 | 0x0000_0000 |
| **ADDR7** | ADC_BA+0x1C | R | A/D数据寄存器 7 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | |
| **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| 保留 | | | | | | VALID | OVERRUN |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| 保留 | | | | RSLT [11:8] | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| RSLT[7:0] | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [31:18] | 保留 | - |
| [17] | **VALID** | **有效标志位**<br>1 = RSLT[11:0] 位数据 有效.<br>0 = RSLT[11:0] 位数据 无效.<br>相应模拟通道 转换完成后，将该位置位，读ADDR 寄存器后，该位由硬件清除. |
| [16] | **OVERRUN** | **结束运行标志位**<br>1 = RSLT[11:0] 数据 被覆盖.<br>0 = RSLT[11:0] 数据 新近转换结果.<br>新的转换结果下载到 寄存器之前，若 RSLT[11:0] 的数据没有被读取, OVERRUN 将置 1. 读ADDR 寄存器后，该位由硬件清除. |

| | | |
|---|---|---|
| [15:12] | 保留 | - |
| [11:0] | **RSLT** | **A/D 转换**结果<br>包括 12 位AD 转换结果. |

### A/D控制寄存器(ADCR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **ADCR** | ADC_BA+0x20 | R/W | ADC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 | | | | **ADST** | **DIFMEN** | **PTEN** | **TRGEN** |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **TRGCOND** | | **TRGS** | | **ADMD** | | **ADIE** | **ADEN** |

| Bits | 描述 | |
|---|---|---|
| [31:12] | 保留 | - |
| [11] | **ADST** | **A/D 转换开始**<br>1 =**转换开始**.<br>0 = 转换结束或A/D转进入空闲状态.<br>ADST 位置位有下列 2 种方式: 软件设定和外部 pin STADC. ADST 将被硬件自动清除在指定通道 单周期模式和单一模式结束后. 在连续扫描模式下, A/D 转换将一直进行 只到软件写该位或系统复位. |
| [10] | **DIFFEN** | **A/D 差分输入模式使能**<br>1 = A/D 为差分输入模式<br>0 = A/D 为单端输入模式<br>转换结束，结果存储在通道0, 1, 2 与 3. |

| 差分对输入通道 | 单端输入通道 | |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 2 | 3 |
| 2 | 4 | 5 |
| 3 | 6 | 7 |

| | | |
|---|---|---|
| [9] | **PTEN** | **PDMA 传送使能**<br>1 = 使能 PDMA 数据传送到 ADDR 0~7<br>0 = 禁止 PDMA 数据传送.<br>当 A/D 转换完成后, 转换的结果将自动装载到 ADDR 0~7, 软件使能该位 可使产生 PDMA 数据传送请求.<br>当 PTEN=1, 软件需设定 ADIE=0 禁止中断. PDMA 可写 ADDR0-7 寄存器 通过 block 或 single 传送模式.<br>适用于 ADC0 和ADC1, ADDR0-3 可通过 PDMA可写 |
| [8] | **TRGE** | **外部触发使能**<br>使能或禁止A/D 转换（通过外部STADC pin）<br>1= 使能<br>0= 禁止 |
| [7:6] | **TRGCOND** | **外部触发条件**<br>该 2 位 决定外部pin STADC 触发状态为(level 或 edge状态). 该信号必须保持至少8 PCLKS的稳定状态用于电平触，4 PCLKS 的高和低状态.<br>00 = 低级<br>01 = 级<br>10 = 下降沿<br>11 = 上升沿 |
| [5:4] | **TRGS** | **硬件触发源**<br>00 = A/D 转换开始（设定外部STADC pin）<br>其它 =保留<br>改变 TRGS 前，软件禁止TRGE 和 ADST.<br>在硬件触发模式下, ADST 位可由硬件触发源设定,软件为最高优先级 设定ADST 位.. |
| [3:2] | **ADMD** | **A/D 转换操作模式**<br>00 = 单一转换<br>01 = 保留<br>10 = 单周期扫描<br>11 = 连续扫描<br>当改变操作模式时, 软件将首先禁止 ADST 位.. |
| [1] | **ADIE** | **A/D 中断使能**<br>1 = 使能 A/D 中断功能<br>0 = 禁止 A/D 中断功能<br>A/D 转换结束产生中断使能 当 ADIE 置位，A/D 转换结束产生中断使能. |
| [0] | **ADEN** | **A/D 转换使能**<br>1 = 使能<br>0 = 禁止<br>开始 A/D 转换功能时,该位需 置位. 该位为 0 将禁止 A/D 转换模拟电路的电源供给. |

### A/D 通道使能寄存器 (ADCHER)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| ADCHER | ADC_BA+0x24 | R/W | A/D 通道使能 | 0x0000_0000 |

在使用ADC0 与 ADC1, 仅寄存器ADCHER0-3 有效, ADCHER4-7不能被设置.

| 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 | | | | | | PRESEL[1:0] | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CHEN7 | CHEN6 | CHEN5 | CHEN4 | CHEN3 | CHEN2 | CHEN1 | CHEN0 |

| Bits | 描述 | |
|---|---|---|
| [31:10] | 保留 | - |
| [9:8] | **PRESEL[1:0]** | **模拟输入通道 7 选择位**<br>00: 模拟输入通道 7<br>01: Bandgap (VBG) 模拟输入<br>10: VTEMP 模拟输入<br>11:模拟地 |
| [7] | **CHEN7** | **模拟输入通道 7**<br>1 = 使能<br>0 = 禁止 |
| [6] | **CHEN6** | **模拟输入通道 6**<br>1 = 使能<br>0 = 禁止 |
| [5] | **CHEN5** | **模拟输入通道 5**<br>1 = 使能<br>0 = 禁止 |

| | | |
|---|---|---|
| [4] | **CHEN4** | **模拟输入通道 4**<br>1 = 使能<br>0 = 禁止 |
| [3] | **CHEN3** | **模拟输入通道 3**<br>1 = 使能<br>0 = 禁止 |
| [2] | **CHEN2** | **模拟输入通道 2**<br>1 = 使能<br>0 = 禁止 |
| [1] | **CHEN1** | **模拟输入通道 1**<br>1 = 使能<br>0 = 禁止 |
| [0] | **CHEN0** | **模拟输入通道 0**<br>1 = 使能<br>0 = 禁止<br>当 CHEN1~7 设定为 0时，该位使能.<br>**在单一模式下，软件使能多通道模式, 仅进行最小通道转换，其他通道将被忽视.** |

### A/D比较寄存器0/1 (ADCMPR0/1)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **ADCMPR0** | ADC_BA+0x28 | R/W | A/D比较寄存器0 | 0x0000_0000 |
| **ADCMPR1** | ADC_BA+0x2C | R/W | A/D比较寄存器1 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| 保留 | | | | CMPD[11:8] | | | |
| **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| CMPD[7:0] | | | | | | | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| 保留 | | | | CMPMATCNT | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| 保留 | | CMPCH | | | CMPCOND | CMPIE | CPMEN |

| Bits | 描述 | |
|---|---|---|
| [31:28] | 保留 | - |
| [27:16] | **CMPD** | **比较数值**<br>此12 位数值将和 指定通道的转换结果相比较,在scan模式下(without imposing a load on software)软件可应用于监控外部模拟输入 pin 电压跃迁 |
| [15:12] | 保留 | - |
| [11:8] | **CMPMATCNT** | **比较匹配值**<br>当指定A/D通道的转换值和比较条件相匹配, 内部计数器将相应的加1.当内部计数器的值达到设定值时, (CMPMATCNT +1) 硬件将置位CMPF 位. |
| [5:3] | **CMPCH** | **Compare 通道选择**<br>000 = 选择 通道 0 转换结果.<br>001 = 选择 通道 1 转换结果.<br>010 = 选择 通道 2 转换结果.<br>011 = 选择 通道 3 转换结果.<br>100 = 选择 通道 4 转换结果.<br>101 = 选择 通道 5 转换结果.<br>110 = 选择 通道 6 转换结果.<br>111 = 选择 通道 7 转换结果. |

| | | |
|---|---|---|
| [2] | **CMPCOND** | **比较条件**<br>1 = CMPF 位置位 当转换结果和CMPD[11:0]相等.<br>0 = CMPF 位置位 当转换结果和CMPD[11:0]相等.<br>1= 设置比较条件即当12位A/D转换结果大于或等于12位CMPD(ADCMPRx[27:16])，内部匹配计数器加1.<br>0=设置比较条件即当12位A/D转换结果小于12位CMPD(ADCMPRx[27:16])，内部匹配计数器减1.<br>**注:** 当内部计数器的值达到(CMPMATCNT +1), CMPF 置位. |
| [1] | **CMPIE** | **比较中断使能**<br>1 = 使能<br>0 =禁止<br>ADDR 中转换的数值小于(等于) MPD[11:0]的值, CMPF 位有效,CMPIE置 1, 产生比较中断请求. |
| [0] | **CMPEN** | **比较使能**<br>1 = 比较使能.<br>0 =比较禁止.<br>该位置位 使能 比较 CMPD[11:0] 特定通道的转换值 当转换数据下载到ADDR 寄存器. |

### A/D状态寄存器 (ADSR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **ADSR** | ADC_BA+0x30 | R/W | ADC 状态寄存器 | undefined |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | |
| **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| **OVERRUN** | | | | | | | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| **VALID** | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| 保留 | **CHANNEL** | | | **BUSY** | **CMPF1** | **CMPF0** | **ADF** |

| Bits | 描述 | |
|---|---|---|
| [31:24] | 保留 | - |
| [23:16] | **OVERRUN** | **结束运行标志位**<br>It is a mirror to OVERRUN bit in ADDRx |
| [15:8] | **VALID** | **数据有效标志位**<br>It is a mirror of VALID bit in ADDRx |
| [7] | 保留 | - |
| [6:4] | **CHANNEL** | **当前转换通道**<br>BUSY=1 表示进行转换中. 当BUSY=0, 表示可进行下次转换.<br>只读位... |
| [3] | **BUSY** | **BUSY/IDLE**<br>1 = A/D 转换忙碌<br>0 = A/D 转换空闲<br>该位反应ADST 位(ADCR).<br>只读位. |

| | | |
|---|---|---|
| [2] | **CMPF1** | **较标志位**<br>选择 A/D 转换通道 结果和ADCMPR1相匹配 该位置位. 写 1 清该位.<br>1 = ADDR 转换结果和 ADCMPR1相匹配<br>0 = ADDR 转换结果和 ADCMPR1不匹配 |
| [1] | **CMPF0** | **比较标志位**<br>选择 A/D 转换通道 结果和ADCMPR 相匹配 该位置位. 写 1 清该位.<br>1 = ADDR 转换结果和 ADCMPR0相匹配<br>0 = ADDR 转换结果和 ADCMPR0不匹配 |
| [0] | **ADF** | **A/D 转换结束标志位**<br>状态标志位 指示A/D 转换的结果<br>ADF 置位有下列2 种条件.<br>1. A/D 转换结束在 单一模式.<br>2. A/D 转换结束在某通道扫描模式.<br>清该位有下列 2 种条件:<br>1. 该位置位<br>2. 当PDMA 传送完成. |

### A/D校准寄存器 (ADCALR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **ADCALR** | ADC_BA+0x34 | R/W | A/D校准寄存器 | 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 | | | | | | **CALDONE** | **CALEN** |

| Bits | 描述 | |
|---|---|---|
| [31:2] | 保留 | - |
| [1] | **CALDONE** | **校准完成标志 (只读)**<br>1 = A/D 转换 自身校准.<br>0 = A/D转换无自身校准 或 自身校准进行中.（若CALEN 位置位）.<br>CALEN 位写0, CALDONE 位将于硬件直接清除 在设定96 ADC clocks 后 ， 该位可直读. |
| [0] | **CALEN** | **自身校准 功能**<br>1 = 使能 自身校准<br>0 = 禁止 自身校准<br>软件置位 该位使能 A/D 转换执行自身校准 功能.需要 127 ADC 时钟 完成校准功能. CALDONE 置位后 该位需保持为高，清该位将禁止自身校准 功能. |

## 5.17 模拟比较器

### 5.17.1 功能描述

NUC1XX 系列包括2 路模拟比较器. 该可应用于不同电压配置. 当正级输入大于负极输入时，为逻辑1，反之为逻辑 0. 每一路比较器可配置成：当输入值发生改变时 产生中断.系统框图如：图 5-87.

使能模拟比较器功能前，每路模拟输入脚需配置为输入类型管教

### 5.17.2 特征

- 模拟输入电压: 0~5.0V
- 软件使能（ hysterisis功能）
- 2 路模拟比较器（带内部输入参考电压和极性选择）
- 1 路模拟比较器输入中断请求

### 5.17.3 框图

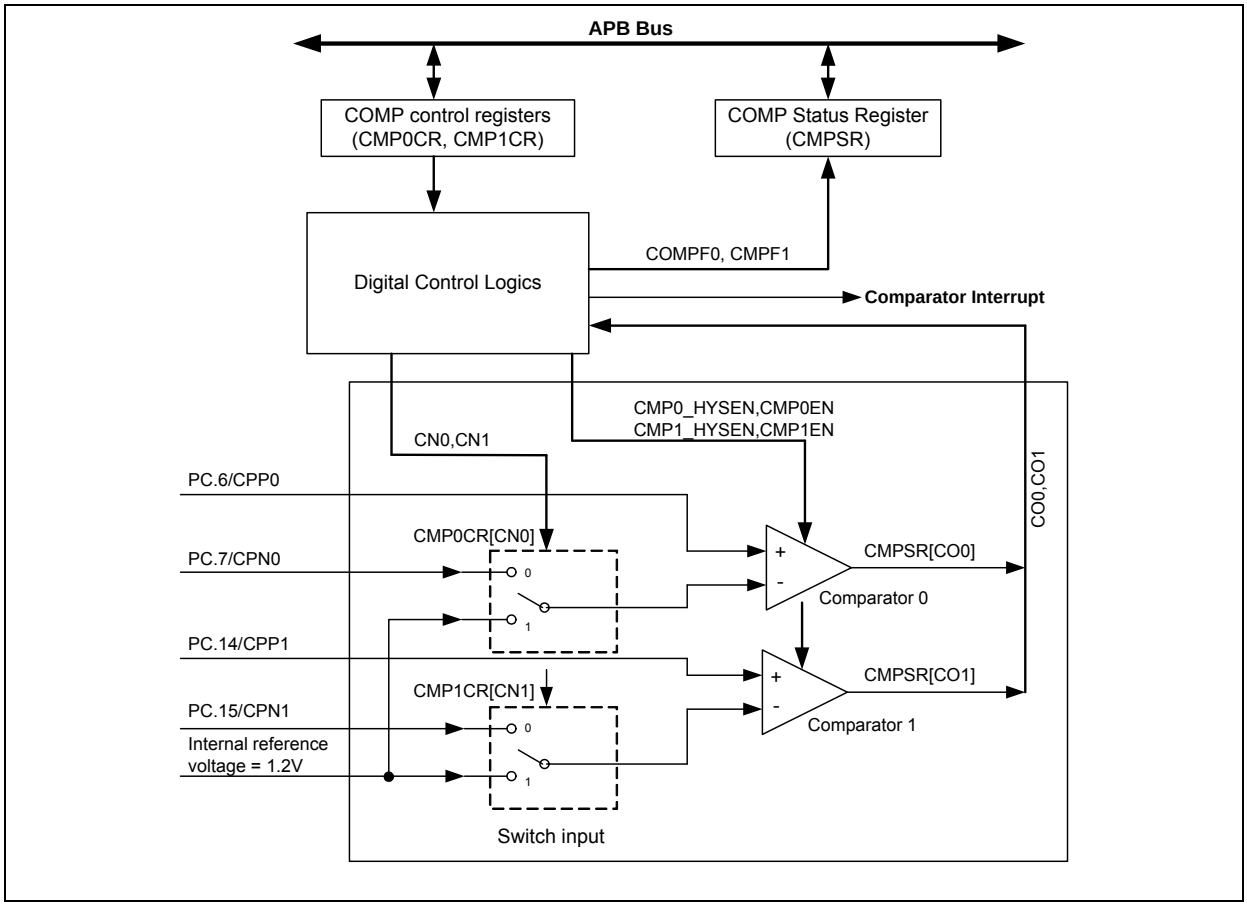



图 5-87 模拟比较器框图

### 5.17.4 操作步骤

*5.17.4.1 中断源*

比较器产生输出 CO1(CO2) CMPSR 寄存器（通过PCLK采样）如果 CMP0IE(CMP1IE) 位（CMP0CR(CMP1CR) ）设定 引起比较器的输出CO0(CO1)状态改变， 将引起比较器标志位 CMPF0(CMPF1) 置位 并产生比较器中断请求. 软件可 写 0 到 CMP0 位 CMPF1 以停止中断请求.

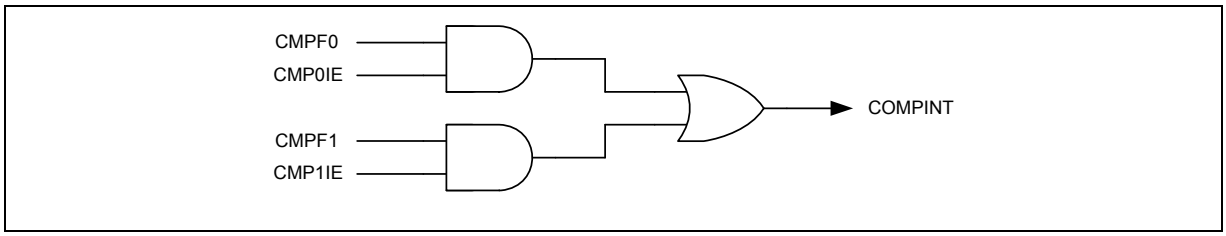



图 5-88 模拟比较器输入中断源

### 5.17.5 寄存器列表

**R: 只读, W: 只写, R/W: 可读写，C:仅在值为0时可写**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **CMP_BA = 0x400D_0000** | | | | |
| **CMP0CR** | CMP_BA+0x00 | R/W | CMP0 控制寄存器 | 0x0000_0000 |
| **CMP1CR** | CMP_BA+0x04 | R/W | CMP1 控制寄存器 | 0x0000_0000 |
| **CMPSR** | ADC_BA+0x08 | R/W | CMP 状态寄存器 | 0x0000_0000 |

### 5.17.6 寄存器描述

**CMP0 控制寄存器 (CMP0CR)**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **CMP0CR** | CMP_BA+0x00 | R/W | CMP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** |
| - | - | - | **CN0** | - | **CMP0_HYSEN** | **CMP0IE** | **CMP0EN** |

| Bits | 描述 | |
|---|---|---|
| [31:5] | 保留 | 保留 |
| [4] | **CN0** | **比较器0 负极输入选择**<br>1 =内部比较器参考电压(Vref=1.2V) 选择作为 比较器的正极输入.<br>0 = 比较器参考 PC.7/CPN0 选择作为 比较器的正极输入. |
| [3] | 保留 | 保留 |
| [2] | **CMP0_HYSEN** | **CMP0 Hysterisis 使能**<br>1 = 使能CMP0 Hysterisis 功能 （ 比较器1 典型 range 电压值位 20mV）.<br>0 =禁止 CMP0 Hysterisis 功能 (默认). |
| [1] | **CMP0IE** | **CMP0 中断使能**<br>1 =使能 CMP0 中断使能<br>0 =禁止 CMP0 中断使能<br>CMP0 转换结束 产生中断请求（CMP0IE 位置位）. |
| [0] | **CMP0EN** | **比较器0使能**<br>1 = 使能<br>0 = 禁止<br>CMP0EN 置位后，比较器等待10 us 开始运作. |

### CMP1控制寄存器(CMP1CR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **CMP1CR** | CMP_BA+0x04 | R/W | CMP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** |
| **RESco1ERVED** | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **-** | **-** | **-** | **CN1** | **-** | **CMP1_HYSEN** | **CMP1IE** | **CMP1EN** |

| Bits | 描述 | |
|---|---|---|
| [31:5] | 保留 | 保留 |
| [4] | **CN1** | **比较器1 负极输入选择**<br>1 = 比较器内部参考电压(Vref=1.2V) 作为比较器正极输入电压 .<br>0 = 比较器参考pin CNG2作为比较器正极输入电压 |
| [3] | 保留 | 保留 |
| [2] | **CMP1_HYSEN** | **CMP1 Hysterisis 使能**<br>1 = 使能CMP1 Hysterisis 功能（at comparator 1 that the typical range is 20mV）<br>0 =禁止 CMP1 Hysterisis 功能 (Default). |
| [1] | **CMP1IE** | **CMP1 中断使能**<br>1 =使能CMP1 中断功能<br>0 = 禁止 CMP1中断功能<br>CMP1 转换结束中断请求产生 当CMP1IE 置位 |

| | | |
|---|---|---|
| [0] | **CMP1EN** | **比较器1使能**<br>1 = 使能<br>0 = 禁止<br>CMP1EN置位后，比较器输出需 10 us后 稳定. |

### CMP状态寄存器(CMPSR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **CMPSR** | CMP_BA+0x08 | R/W | CMP状态寄存器 | undefined |

| 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 | | | | **CO1** | **CO0** | **CMPF1** | **CMPF0** |

| Bits | 描述 | |
|---|---|---|
| [31:4] | 保留 | 保留 |
| [3] | **CO1** | **比较器1 输出**<br>APB 时钟同步 允许软件访问. 比较器禁止(CMP0EN = 0) 清该位. |
| [2] | **CO0** | **比较器0 输出**<br>APB 时钟同步 允许软件访问. 比较器禁止(CMP0EN = 0) 清该位. |
| [1] | **CMPF1** | **比较器1 标志位**<br>比较器输出状态改变时，该位将于硬件置位. 当中断级别足够并且中断使能，该位将引起硬件中断.**A write 0 清 CMPF1.** |
| [0] | **CMPF0** | **比较器0 标志位**<br>比较器输出状态改变时，该位将于硬件置位. 当中断级别足够并且中断使能，该位将引起硬件中断. **A write 0清CMPF0.** |

## 5.18 PDMA 控制器

### 5.18.1 概述

NUC1XX 包括指导访问存储器的外设 (PDMA) 控制器,用于APB和存储器间的数据交换. PDMA 具有 9 路DMA (外设-存储器 或存储器 -外设 或存储器-存储器) 通道. 每路 PDMA 通道 (PDMA CH0~CH8), 具有一组缓冲器用于外设 APB IP 和 存储器间 缓冲数据传递.

软件 设定禁止 PDMA [PDMACEN] 位 可禁止PDMA的运作. PDMA 运行通过软件检测或接收到内部 PDMA 中断, CPU认识运作完成. NUC1XX PDMA 控制器可增加源和目的地址

### 5.18.2 特性

- AMBA AHB master/slave 模式兼容, 用于数据传输和寄存器 读/写.
- PDMA 支持 32-位 源目的地址增加. (目标地址为address increment, fixed 或wrap around).

### 5.18.3 框图

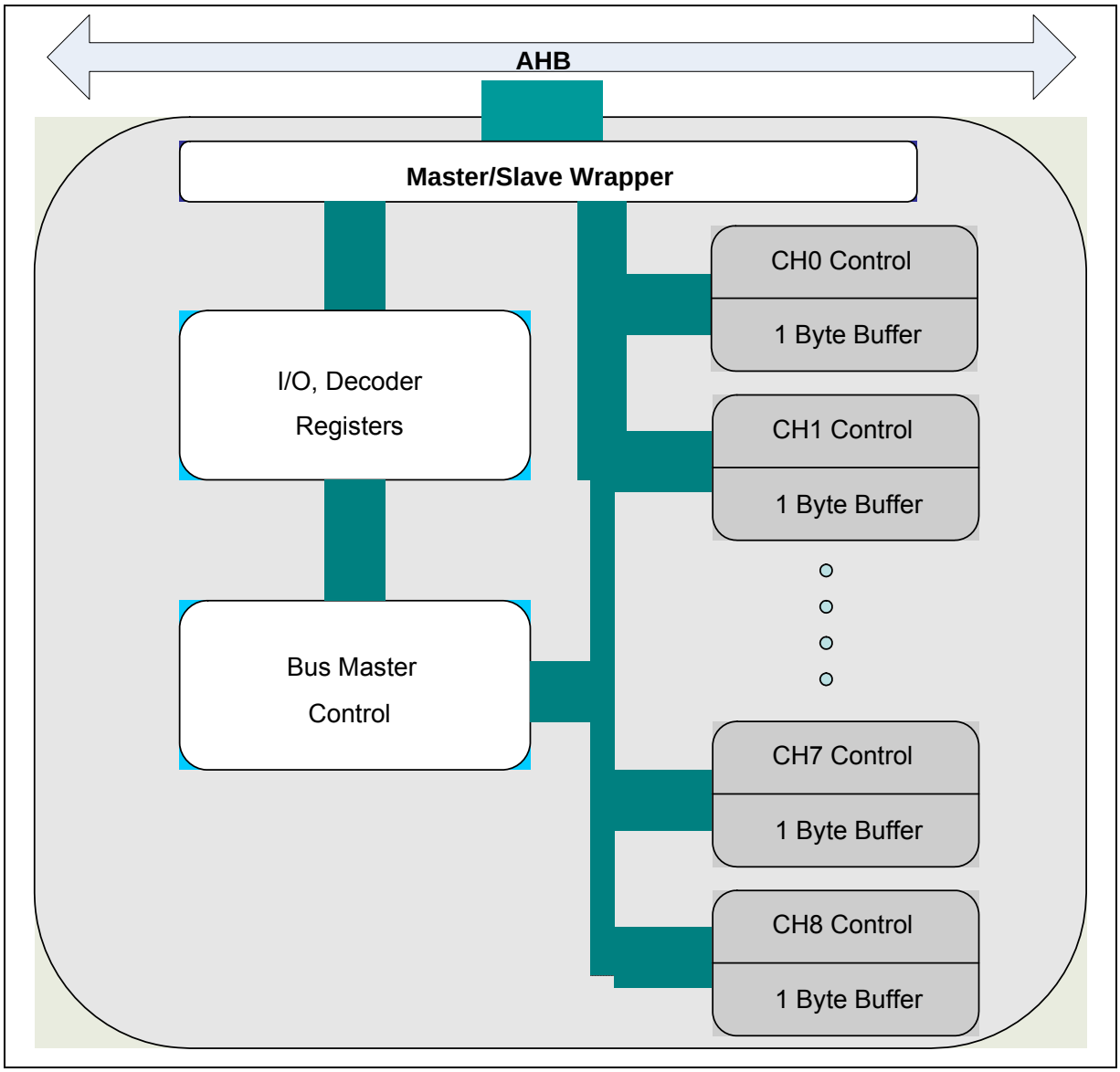



图 5-89 PDMA 控制器框图

### 5.18.4 功能描述

PDMA 控制器具有 9 路相关联 DMA 应用于 外设-到-存储器、存储器-到-外设 或 存储器-到-存储器. 每路 PDMA 通道, 在 APB IP 和存储器之间有传递缓冲器.

CPU 通过软件轮流检测或 接收到内部 PDMA 中断，可侦测到 PDMA 完成.对于来源和目的地址而言, NUC1XX PDMA 控制器具有 三种模式: incrementing, fixed 和 wraparound 模式.

NUC1XX PDMA 模拟通道无说先设定, 因此用户需通过设定PDMA_PDSSR0 和 PDMA_PDSSR1先行配置每路的设 PDMA 通道.

软件需使能 DMA 通道 PDMA [PDMACEN] 并且写有效的来源地址到PDMA_SARx 寄存器, 、 目的地址到 PDMA_DSABx 寄存器, 传输计算数到 PDMA_BCRx 寄存器. 以及DMA_CSRx PDMA [Trig_EN] 触发. 如果源地址和目的地址非 wrap around 模式, PDMA 将继续传递 直到 PDMA_CBCRx 为 0 (在 wrap around 模式下，当 PDMA_CBCRx 为 0 ，PDMA 将重装载 PDMA_CBCRx 直到软件禁止 PDMA_CSRx[DMACEN]), 如果 error 发生在 PDMA 运作时, 通道将停止 直到 软件清除 error 条件 设定 PDMA_CSRx [SW_RST] 复位 PDMA 通道和设定 PDMA_CSRx [EDMACEN] 和 [Trig_EN] 位 再次重新开始. 当 error发生在 PDMA 运作时, 该通道将停止直到 软件清除 error 条件 设定 PDMA_CSRx [SW_RST] 复位 PDMA 通道和设定 PDMA_CSRx [EDMACEN] 和 [Trig_EN] 位 再次重新开始.

PDMA (外设-到-存储器、存储器-到-外设) 模式, DMA 可以在 外设 APB IP (ex: UART, SPI, ADC….) 和 Memory 将传递数据.

### 5.18.5 PDMA 控制器寄存器列表

**R: 只读, W: 只写, R/W: 可读写，C:仅在值为0时可写**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **PDMA_BA** = 0x5000_8000<br>**PDMA_BA_ch0** = 0x5000_8000 **PDMA_BA_ch1** = 0x5000_8100 **PDMA_BA_ch2** = 0x5000_8200<br>**PDMA_BA_ch3** = 0x5000_8300 **PDMA_BA_ch4** = 0x5000_8400 **PDMA_BA_ch5** = 0x5000_8500<br>**PDMA_BA_ch6** = 0x5000_8600 **PDMA_BA_ch7** = 0x5000_8700 **PDMA_BA_ch8** = 0x5000_8800 | | | | |
| **PDMA_CSR** | PDMA_BA_chx + 0x00 | R/W | PDMA 控制寄存器 | 0x0000_0000 |
| **PDMA_SAR** | PDMA_BA_chx + 0x04 | R/W | PDMA 源地址寄存器 | 0x0000_0000 |
| **PDMA_DAR** | PDMA_BA_chx + 0x08 | R/W | PDMA 目的地址寄存器 | 0x0000_0000 |
| **PDMA_BCR** | PDMA_BA_chx + 0x0C | R/W | PDMA 发送字节计数寄存器 | 0x0000_0000 |
| **PDMA_POINT** | PDMA_BA_chx + 0x10 | R | PDMA 内部数据指示器 | 0xXXXX_0000 |
| **PDMA_CSAR** | PDMA_BA_chx + 0x14 | R | PDMA 当前源目的地址 | 0x0000_0000 |
| **PDMA_CDAR** | PDMA_BA_chx + 0x18 | R | PDMA 当前目的地址寄存器 | 0x0000_0000 |
| **PDMA_CBCR** | PDMA_BA_chx + 0x1C | R | PDMA 当前传输字节计数寄存器 | 0x0000_0000 |
| **PDMA_IER** | PDMA_BA_chx + 0x20 | R/W | PDMA 中断使能寄存器 | 0x0000_0001 |
| **PDMA_ISR** | PDMA_BA_chx + 0x24 | R/W | PDMA 中断状态寄存器 | 0x0000_0000 |
| **PDMA_SBUF0** | PDMA_BA_chx + 0x80 | R | PDMA 共享缓冲 FIFO 0 | 0x0000_0000 |
| **PDMA_BA_GCR** = 0x5000_8F00 | | | | |
| **PDMA_GCRCSR** | PDMA_BA_GCR + 0x00 | R/W | PDMA全局控制寄存器 | 0x0000_0000 |
| **PDMA_PDSSR0** | PDMA_BA_GCR + 0x04 | R/W | PDMA 服务选择控制寄存器 0 | 0XFFFF_FFFF |
| **PDMA_PDSSR1** | PDMA_BA_GCR + 0x08 | R/W | PDMA 服务选择控制寄存器 1 | 0xFFFF_FFFF |
| **PDMA_GCRISR** | PDMA_BA_GCR + 0x0C | R/W | PDMA全局中断寄存器 | 0x0000_0000 |
| **PDMA_PDSSR2** | PDMA_BA_GCR + 0x10 | R/W | PDMA 服务选择控制寄存器2 | 0x0000_00FF |

### 5.18.6 PDMA 控制寄存器描述

**PDMA控制和状态寄存器(PDMA_CSR)**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **PDMA_CSR0** | PDMA_BA+0x000 | R/W | PMAC 控制和状态寄存器 CH0 | 0x0000_0000 |
| **PDMA_CSR1** | PDMA_BA+0x100 | R/W | PMAC 控制和状态寄存器 CH1 | 0x0000_0000 |
| **PDMA_CSR2** | PDMA_BA+0x200 | R/W | PMAC 控制和状态寄存器 CH2 | 0x0000_0000 |
| **PDMA_CSR3** | PDMA_BA+0x300 | R/W | PMAC 控制和状态寄存器 CH3 | 0x0000_0000 |
| **PDMA_CSR4** | PDMA_BA+0x400 | R/W | PMAC 控制和状态寄存器 CH4 | 0x0000_0000 |
| **PDMA_CSR5** | PDMA_BA+0x500 | R/W | PMAC 控制和状态寄存器 CH5 | 0x0000_0000 |
| **PDMA_CSR6** | PDMA_BA+0x600 | R/W | PMAC 控制和状态寄存器 CH6 | 0x0000_0000 |
| **PDMA_CSR7** | PDMA_BA+0x700 | R/W | PMAC 控制和状态寄存器 CH7 | 0x0000_0000 |
| **PDMA_CSR8** | PDMA_BA+0x800 | R/W | PMAC 控制和状态寄存器 CH8 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | |
| **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| **Trig_EN** | 保留 | | **APB_TWS** | | 保留 | | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| **WAR_BCR_SEL** | | | | 保留 | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **DAD_SEL** | | **SAD_SEL** | | **MODE_SEL** | | **SW_RST** | **PDMACEN** |

| Bits | 描述 | |
|---|---|---|
| [31:24] | 保留 | 保留 |
| [23] | **Trig_EN** | **Trig_EN**<br>0 = 无效.<br>1 = 使能 PDMA 数据读写 传输.<br>**注:**当 PDMA 传输完成, 该位自动清除.<br>如果总线 error 发生, 所有 PDMA 传输将停止. 软件复位 PDMA 通道, 重新开始. |
| [22:21] | 保留 | 保留 |

| | | |
|---|---|---|
| [20:19] | **APB_TWS** | **外设传输宽度选择**<br>00 = 一个字(32 bits)为单位传输 用于 PDMA 运作.<br>01 = 一字节(8 bits)为单位传输 用于 PDMA 运作.<br>10 = 一个半字 (16 bits) 为单位传输用于 PDMA 运作.<br>11 = 保留.<br>**注:** 该项仅支持**MODE_SEL**内的IP到存储器模式 (APB-to-Memory)和存储器到 IP 模式 (Memory-to-APB) |
| [18:16] | 保留 | 保留 |
| [15:12] | **WAR_BCR_SEL** | **Wrap around 发送字节计数中断选择**<br>0001 = PDMA_CBCR 为 0 中断产生.<br>0100 = 当 PDMA_CBCR 值和 1/2 PDMA_BCR相等 中断产生.<br>0000 = 无 中断.<br>**注:** 仅当**PDMA_CSRx [SAD_SEL]** 等于2'b11 或**PDMA_CSRx [DAD_SEL]** 等于2'b11, 该位有实际意义 |
| [11:8] | 保留 | 保留 |
| [7:6] | **DAD_SEL** | **传输目的地址方向选择**<br>00 =传输目的地址 持续增加.<br>01 = 保留.<br>10 =传输目的地址固定(适用于多数据 传递到单一目的地址).<br>11 = 传递目的地址(当 PDMA_CBCR 为 0 , PDMA_CDAR 和 PDMA_CBCR 寄存器由 PDMA_DAR 和 PDMA_BCR 自动清 0 . PDMA 将开始其它传递无软件触发 直到 PDMA_EN 禁止. 当 PDMA_EN 禁止, PDMA 将完成传递，但PDMA_BUF 中保留的数据将无法传递给目的地址). |
| [5:4] | **SAD_SEL** | **传输源地址方向选择**<br>00 =传输源地址 持续增加.<br>01 =保留.<br>10 = 传输源地址固定(适用于单数据 传递到多目的地址).<br>11 = 传递源地址 (当 PDMA_CBCR 为 0 , PDMA_CSAR 和 PDMA_CBCR 将通过 PDMA_SAR 和 PDMA_BCR 寄存器自动更新. 将开始其它传递无软件触发 直到 PDMA_EN 禁止. 当 PDMA_EN 禁止, PDMA 将完成传递，但PDMA_BUF 中保留的数据将无法传递给目的地址). |
| [3:2] | **MODE_SEL** | **PDMA 模式选择**<br>00 =存储器到存储器模式 (Memory-to-Memory).<br>01 = IP到存储器模式 (APB-to-Memory).<br>10 = 存储器到 IP 模式 (Memory-to-APB). |
| [1] | **SW_RST** | **软件产生复位**<br>0 = 该位写 0 无效.<br>1 = 写 1 将复位内部机器和指令状态. 控制寄存器的内容将不被清除. 该位在 很少周期后自动清除. |
| [0] | **PDMACEN** | **PDMA 通道使能**<br>该位置位 使能 PDMA's 运作. 如果该位被清除, PDMA 清忽视所有 PDMA 请求 并且强制主机总线 进入 IDLE 状态.<br>**注:** SW_RST 将清该位. |

### PDMA传输源地址寄存器(PDMA_SAR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **PDMA_SAR0** | PDMA_BA+0x004 | R/W | PDMA 传输源地址寄存器 CH0 | 0x0000_0000 |
| **PDMA_SAR1** | PDMA_BA+0x104 | R/W | PDMA 传输源地址寄存器 CH1 | 0x0000_0000 |
| **PDMA_SAR2** | PDMA_BA+0x204 | R/W | PDMA 传输源地址寄存器 CH2 | 0x0000_0000 |
| **PDMA_SAR3** | PDMA_BA+0x304 | R/W | PDMA 传输源地址寄存器 CH3 | 0x0000_0000 |
| **PDMA_SAR4** | PDMA_BA+0x404 | R/W | PDMA 传输源地址寄存器 CH4 | 0x0000_0000 |
| **PDMA_SAR5** | PDMA_BA+0x504 | R/W | PDMA 传输源地址寄存器 CH5 | 0x0000_0000 |
| **PDMA_SAR6** | PDMA_BA+0x604 | R/W | PDMA 传输源地址寄存器 CH6 | 0x0000_0000 |
| **PDMA_SAR7** | PDMA_BA+0x704 | R/W | PDMA 传输源地址寄存器 CH7 | 0x0000_0000 |
| **PDMA_SAR8** | PDMA_BA+0x804 | R/W | PDMA 传输源地址寄存器 CH8 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:0] | **PDMA_SAR** | **PDMA 发送源地址寄存器**<br>该寄存器用于指向 32位PDMA源地址.<br>**注:** 源地址必须为字排列 |

### PDMA传输目的地址寄存器(PDMA_DAR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **PDMA_DAR0** | PDMA_BA+0x008 | R/W | PDMA 传输目的地址寄存器 CH0 | 0x0000_0000 |
| **PDMA_DAR1** | PDMA_BA+0x108 | R/W | PDMA 传输目的地址寄存器 CH1 | 0x0000_0000 |
| **PDMA_DAR2** | PDMA_BA+0x208 | R/W | PDMA 传输目的地址寄存器 CH2 | 0x0000_0000 |
| **PDMA_DAR3** | PDMA_BA+0x308 | R/W | PDMA 传输目的地址寄存器 CH3 | 0x0000_0000 |
| **PDMA_DAR4** | PDMA_BA+0x408 | R/W | PDMA 传输目的地址寄存器 CH4 | 0x0000_0000 |
| **PDMA_DAR5** | PDMA_BA+0x508 | R/W | PDMA 传输目的地址寄存器 CH5 | 0x0000_0000 |
| **PDMA_DAR6** | PDMA_BA+0x608 | R/W | PDMA 传输目的地址寄存器 CH6 | 0x0000_0000 |
| **PDMA_DAR7** | PDMA_BA+0x708 | R/W | PDMA 传输目的地址寄存器 CH7 | 0x0000_0000 |
| **PDMA_DAR8** | PDMA_BA+0x808 | R/W | PDMA传输目的地址寄存器 CH8 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:0] | **PDMA_DAR** | **PDMA 传输目的地址寄存器**<br>32位 PDMA目的地址.<br>**注：**源目的地址必须为一列（ word 对齐）. |

### PDMA传输计数寄存器(PDMA_BCR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **PDMA_BCR0** | PDMA_BA+0x00C | R/W | PDMA 传输计数寄存器 CH0 | 0x0000_0000 |
| **PDMA_BCR1** | PDMA_BA+0x10C | R/W | PDMA 传输计数寄存器 CH1 | 0x0000_0000 |
| **PDMA_BCR2** | PDMA_BA+0x20C | R/W | PDMA 传输计数寄存器 CH2 | 0x0000_0000 |
| **PDMA_BCR3** | PDMA_BA+0x30C | R/W | PDMA 传输计数寄存器 CH3 | 0x0000_0000 |
| **PDMA_BCR4** | PDMA_BA+0x40C | R/W | PDMA 传输计数寄存器 CH4 | 0x0000_0000 |
| **PDMA_BCR5** | PDMA_BA+0x50C | R/W | PDMA 传输计数寄存器 CH5 | 0x0000_0000 |
| **PDMA_BCR6** | PDMA_BA+0x60C | R/W | PDMA 传输计数寄存器 CH6 | 0x0000_0000 |
| **PDMA_BCR7** | PDMA_BA+0x70C | R/W | PDMA 传输计数寄存器 CH7 | 0x0000_0000 |
| **PDMA_BCR8** | PDMA_BA+0x80C | R/W | PDMA 传输计数寄存器 CH8 | 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** |
| PDMA_BCR [15:8] | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| PDMA_BCR [7:0] | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| [31:24] | 保留 | 保留 |
| [15:0] | **PDMA_BCR** | **PDMA 传输计数寄存器**<br>16位PDMA. 传输计数寄存器<br>**注:** 当 memory-to-memory (PDMA_CSR [MODE_SEL] = 2'b00) 模式, 传递数据须为一列. |

### PDMA内部缓冲指示寄存器(PDMA_POINT)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **PDMA_POINT0** | PDMA_BA+0x010 | R | PDMA 内部缓冲指示寄存器 CH0 | 0xXXXX_0000 |
| **PDMA_POINT1** | PDMA_BA+0x110 | R | PDMA 内部缓冲指示寄存器 CH1 | 0xXXXX_0000 |
| **PDMA_POINT2** | PDMA_BA+0x210 | R | PDMA 内部缓冲指示寄存器 CH2 | 0xXXXX_0000 |
| **PDMA_POINT3** | PDMA_BA+0x310 | R | PDMA 内部缓冲指示寄存器 CH3 | 0xXXXX_0000 |
| **PDMA_POINT4** | PDMA_BA+0x410 | R | PDMA 内部缓冲指示寄存器 CH4 | 0xXXXX_0000 |
| **PDMA_POINT5** | PDMA_BA+0x510 | R | PDMA 内部缓冲指示寄存器 CH5 | 0xXXXX_0000 |
| **PDMA_POINT6** | PDMA_BA+0x610 | R | PDMA 内部缓冲指示寄存器 CH6 | 0xXXXX_0000 |
| **PDMA_POINT7** | PDMA_BA+0x710 | R | PDMA 内部缓冲指示寄存器 CH7 | 0xXXXX_0000 |
| **PDMA_POINT8** | PDMA_BA+0x810 | R | PDMA 内部缓冲指示寄存器 CH8 | 0xXXXX_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 | | | | | | PDMA_POINT | |

| Bits | 描述 | |
|---|---|---|
| [31:2] | 保留 | 保留 |
| [1:0] | **PDMA_POINT** | **PDMA 内部缓冲指示寄存器 (Read Only)**<br>内部缓冲指示器. |

### PDMA源地址寄存器(PDMA_CSAR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **PDMA_CSAR0** | PDMA_BA+0x014 | R | PDMA 源地址寄存器 CH0 | 0x0000_0000 |
| **PDMA_CSAR1** | PDMA_BA+0x114 | R | PDMA 源地址寄存器 CH1 | 0x0000_0000 |
| **PDMA_CSAR2** | PDMA_BA+0x214 | R | PDMA 源地址寄存器 CH2 | 0x0000_0000 |
| **PDMA_CSAR3** | PDMA_BA+0x314 | R | PDMA 源地址寄存器 CH3 | 0x0000_0000 |
| **PDMA_CSAR4** | PDMA_BA+0x414 | R | PDMA 源地址寄存器 CH4 | 0x0000_0000 |
| **PDMA_CSAR5** | PDMA_BA+0x514 | R | PDMA 源地址寄存器 CH5 | 0x0000_0000 |
| **PDMA_CSAR6** | PDMA_BA+0x614 | R | PDMA 源地址寄存器 CH6 | 0x0000_0000 |
| **PDMA_CSAR7** | PDMA_BA+0x714 | R | PDMA 源地址寄存器 CH7 | 0x0000_0000 |
| **PDMA_CSAR8** | PDMA_BA+0x814 | R | PDMA 源地址寄存器 CH8 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:0] | **PDMA_CSAR** | **PDMA 源地址寄存器 (只读)**<br>此区域表示 PDMA 传递故障时的源目的地址. |

### PDMA目的地址寄存器(PDMA_CDAR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **PDMA_CDAR0** | PDMA_BA+0x018 | R | PDMA 目的地址寄存器 CH0 | 0x0000_0000 |
| **PDMA_CDAR1** | PDMA_BA+0x118 | R | PDMA 目的地址寄存器 CH1 | 0x0000_0000 |
| **PDMA_CDAR2** | PDMA_BA+0x218 | R | PDMA 目的地址寄存器 CH2 | 0x0000_0000 |
| **PDMA_CDAR3** | PDMA_BA+0x318 | R | PDMA 目的地址寄存器 CH3 | 0x0000_0000 |
| **PDMA_CDAR4** | PDMA_BA+0x418 | R | PDMA 目的地址寄存器 CH4 | 0x0000_0000 |
| **PDMA_CDAR5** | PDMA_BA+0x518 | R | PDMA 目的地址寄存器 CH5 | 0x0000_0000 |
| **PDMA_CDAR6** | PDMA_BA+0x618 | R | PDMA 目的地址寄存器 CH6 | 0x0000_0000 |
| **PDMA_CDAR7** | PDMA_BA+0x718 | R | PDMA 目的地址寄存器 CH7 | 0x0000_0000 |
| **PDMA_CDAR8** | PDMA_BA+0x818 | R | PDMA 目的地址寄存器 CH8 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:0] | **PDMA_CDAR** | **PDMA 目的地址寄存器 (Read Only)**<br>此区域表示 PDMA 传递故障时的源目的地址. |

### PDMA当前字节计数寄存器(PDMA_CBCR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **PDMA_CBCR0** | PDMA_BA+0x01C | R | PDMA当前字节计数寄存器CH0 | 0x0000_0000 |
| **PDMA_CBCR1** | PDMA_BA+0x11C | R | PDMA当前字节计数寄存器CH1 | 0x0000_0000 |
| **PDMA_CBCR2** | PDMA_BA+0x21C | R | PDMA当前字节计数寄存器CH2 | 0x0000_0000 |
| **PDMA_CBCR3** | PDMA_BA+0x31C | R | PDMA当前字节计数寄存器CH3 | 0x0000_0000 |
| **PDMA_CBCR4** | PDMA_BA+0x41C | R | PDMA当前字节计数寄存器CH4 | 0x0000_0000 |
| **PDMA_CBCR5** | PDMA_BA+0x51C | R | PDMA当前字节计数寄存器CH5 | 0x0000_0000 |
| **PDMA_CBCR6** | PDMA_BA+0x61C | R | PDMA当前字节计数寄存器CH6 | 0x0000_0000 |
| **PDMA_CBCR7** | PDMA_BA+0x71C | R | PDMA当前字节计数寄存器CH7 | 0x0000_0000 |
| **PDMA_CBCR8** | PDMA_BA+0x81C | R | PDMA当前字节计数寄存器CH8 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:24] | 保留 | 保留 |
| [23:0] | **PDMA_CBCR** | **PDMA 位元计数寄存器 (Read Only)**<br>PDMA 计数器当前保持的字节. |

### PDMA中断使能控制寄存(PDMA_IER)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **PDMA_IER0** | PDMA_BA+0x020 | R/W | PDMA 中断使能控制寄存器 CH0 | 0x0000_0001 |
| **PDMA_IER1** | PDMA_BA+0x120 | R/W | PDMA 中断使能控制寄存器 CH1 | 0x0000_0001 |
| **PDMA_IER2** | PDMA_BA+0x220 | R/W | PDMA 中断使能控制寄存器 CH2 | 0x0000_0001 |
| **PDMA_IER3** | PDMA_BA+0x320 | R/W | PDMA 中断使能控制寄存器 CH3 | 0x0000_0001 |
| **PDMA_IER4** | PDMA_BA+0x420 | R/W | PDMA 中断使能控制寄存器 CH4 | 0x0000_0001 |
| **PDMA_IER5** | PDMA_BA+0x520 | R/W | PDMA 中断使能控制寄存器 CH5 | 0x0000_0001 |
| **PDMA_IER6** | PDMA_BA+0x620 | R/W | PDMA 中断使能控制寄存器 CH6 | 0x0000_0001 |
| **PDMA_IER7** | PDMA_BA+0x720 | R/W | PDMA 中断使能控制寄存器 CH7 | 0x0000_0001 |
| **PDMA_IER8** | PDMA_BA+0x820 | R/W | PDMA 中断使能控制寄存器 CH8 | 0x0000_0001 |

| 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 | | | | | **WAR_IE** | **BLKD_IE** | **TABORT_IE** |

| Bits | 描述 | |
|---|---|---|
| [31:3] | 保留 | 保留 |
| [2] | **WAR_IE** | **Wrap Around 中断使能**<br>0 = 禁止Wrap around中断<br>1 =使能 Wrap Around中断 |
| [1] | **BLKD_IE** | **PDMA Transfer Done 中断使能**<br>0 = 禁止 PDMA 传递时中断<br>1 = 允许 PDMA 传递时中断. |

| | | |
|---|---|---|
| [0] | **TABORT_IE** | **PDMA 读/写 异常中断使能**<br>0 = 禁止 PDMA 传递时 异常中断.<br>1 = 允许 PDMA 传递时 异常中断. |

### PDMA中断状态寄存器(PDMA_ISR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **PDMA_ISR0** | PDMA_BA+0x024 | R/W | PDMA 中断状态寄存器 CH0 | 0x0x0x_0000 |
| **PDMA_ISR1** | PDMA_BA+0x124 | R/W | PDMA 中断状态寄存器 CH1 | 0x0x0x_0000 |
| **PDMA_ISR2** | PDMA_BA+0x224 | R/W | PDMA 中断状态寄存器 CH2 | 0x0x0x_0000 |
| **PDMA_ISR3** | PDMA_BA+0x324 | R/W | PDMA 中断状态寄存器 CH3 | 0x0x0x_0000 |
| **PDMA_ISR4** | PDMA_BA+0x424 | R/W | PDMA 中断状态寄存器 CH4 | 0x0x0x_0000 |
| **PDMA_ISR5** | PDMA_BA+0x524 | R/W | PDMA 中断状态寄存器 CH5 | 0x0x0x_0000 |
| **PDMA_ISR6** | PDMA_BA+0x624 | R/W | PDMA 中断状态寄存器 CH6 | 0x0x0x_0000 |
| **PDMA_ISR7** | PDMA_BA+0x724 | R/W | PDMA 中断状态寄存器 CH7 | 0x0x0x_0000 |
| **PDMA_ISR8** | PDMA_BA+0x824 | R/W | PDMA 中断状态寄存器 CH8 | 0x0x0x_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31] | **INTR** | **中断管脚状态 (只读)**<br>PDMA通道x中断 管脚状态. |
| [30:16] | 保留 | 保留 |
| [15] | **Busy** | **PDMA 传递进展(只读)**<br>0 = PDMA 传递无进展.<br>1 = PDMA 传递进展. |

| | | |
|---|---|---|
| [14:12] | 保留 | 保留 |
| [11:8] | **WAR_BCR_IF** | **Wrap around 传递字节计时器中断标志位**<br>0001 = PDMA_CBCR 为 0 标志.<br>0100 = PDMA_CBCR 为 1/2 PDMA_BCR.<br>**注:** 该位直读, 可写 '1' 清除. |
| [1] | **BLKD_IF** | **Block 传递完成 中断标志位**<br>该位预示 SD主机完成所有 data-in 和 data-out. 若 CRC-16 error 或 incorrect CRC 的状态为妨碍多数据传递, 传递将停止并且该位置位.<br>0 = 未完成.<br>1 = 完成.<br>**注:** 该位直读, 可写 '1' 清除. |
| [0] | **TABORT_IF** | **PDMA 读/写 目标异常中断标志位**<br>0 = 无总线 ERROR 应答响应.<br>1 = 总线 ERROR应答响应.<br>**注:** 该位直读, 可写 '1' 清除. |

**注:** PDMA_ISR [TABORT_IF] 表示主机总线接收ERROR响应与否, 如果主机总线接收发生，意味着目标异常发生， PDMAC 将停止传输和响应该事件，软件进入IDLE状态. 当目标异常发生，软件必须复位PDMAC, 并重传这些数据.

### PDMA 共享缓冲 FIFO 0 (PDMA_SBUF0_cx)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **PDMA_SBUF0_c0** | PDMA_BA+0x080 | R | PDMA 共享缓冲 FIFO 0 寄存器 CH0 | 0x0000_0000 |
| **PDMA_SBUF0_c1** | PDMA_BA+0x180 | R | PDMA 共享缓冲 FIFO 0 寄存器 CH1 | 0x0000_0000 |
| **PDMA_SBUF0_c2** | PDMA_BA+0x280 | R | PDMA 共享缓冲 FIFO 0 寄存器 CH2 | 0x0000_0000 |
| **PDMA_SBUF0_c3** | PDMA_BA+0x380 | R | PDMA 共享缓冲 FIFO 0 寄存器 CH3 | 0x0000_0000 |
| **PDMA_SBUF0_c4** | PDMA_BA+0x480 | R | PDMA 共享缓冲 FIFO 0 寄存器 CH4 | 0x0000_0000 |
| **PDMA_SBUF0_c5** | PDMA_BA+0x580 | R | PDMA 共享缓冲 FIFO 0 寄存器 CH5 | 0x0000_0000 |
| **PDMA_SBUF0_c6** | PDMA_BA+0x680 | R | PDMA 共享缓冲 FIFO 0 寄存器 CH6 | 0x0000_0000 |
| **PDMA_SBUF0_c7** | PDMA_BA+0x780 | R | PDMA 共享缓冲 FIFO 0 寄存器 CH7 | 0x0000_0000 |
| **PDMA_SBUF0_c8** | PDMA_BA+0x880 | R | PDMA 共享缓冲 FIFO 0 寄存器 CH8 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| **PDMA_SBUF0 [31:24]** | | | | | | | |
| **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| **PDMA_SBUF0 [23:16]** | | | | | | | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| **PDMA_SBUF0 [15:8]** | | | | | | | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **PDMA_SBUF0 [7:0]** | | | | | | | |

| Bits | 描述 | |
|---|---|---|
| **[31:0]** | **PDMA_SBUF0** | **PDMA共享缓冲 FIFO 0 (只读)**<br>每通道拥有专属的1个字(32位)缓存. |

### PDMA全局控制和状态寄存器(PDMA_GCRCSR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| PDMA_GCRCSR | PDMA_BA_GCR + 0x000 | R/W | PDMA全局控制和状态寄存器 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 保留 | | | | | | | HCLK8_EN |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| HCLK7_EN | HCLK6_EN | HCLK5_EN | HCLK4_EN | HCLK3_EN | HCLK2_EN | HCLK1_EN | HCLK0_EN |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | | | | | | PDMA_RST |

| Bits | 描述 | |
|---|---|---|
| [31:17] | 保留 | 保留 |
| [16] | **HCLK8_EN** | **PDMA 控制通道 8 时钟使能控制**<br>0 = 禁止<br>1 = 使能 |
| [15] | **HCLK7_EN** | **PDMA 控制通道 7 时钟使能控制**<br>0 = 禁止<br>1 = 使能 |
| [14] | **HCLK6_EN** | **PDMA 控制通道 6 时钟使能控制**<br>0 = 禁止<br>1 = 使能 |
| [13] | **HCLK5_EN** | **PDMA 控制通道 5 时钟使能控制**<br>0 = 禁止<br>1 = 使能 |
| [12] | **HCLK4_EN** | **PDMA 控制通道 4 时钟使能控制**<br>0 = 禁止<br>1 = 使能 |

| | | |
|---|---|---|
| [11 | **HCLK3_EN** | **PDMA 控制通道 3 时钟使能控制**<br>0 = 禁止<br>1 = 使能 |
| [10 | **HCLK2_EN** | **PDMA 控制通道 2 时钟使能控制**<br>0 = 禁止<br>1 = 使能 |
| [9] | **HCLK1_EN** | **PDMA 控制通道 1 时钟使能控制**<br>0 = 禁止<br>1 = 使能 |
| [8] | **HCLK0_EN** | **PDMA 控制通道 0 时钟使能控制**<br>0 = 禁止<br>1 = 使能 |
| [7:1] | 保留 | 保留 |
| [0] | **PDMA_RST** | **PDMA 软件复位**<br>0 = 写 0 该位将无结果.<br>1 = 写 1 将复位内部指令. 控制寄存器将不会被清零. 该位在数机器周期后 自动清零.<br>**注:** 该位可复位所有信道 (global reset). |

### PDMA服务选择控制寄存器 0 (PDSSR0)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| PDSSR0 | PDMA_BA_GCR +0x04 | R/W | EDMA服务选择控制寄存器 0 | 0xFFFF_FFFF |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| SPI3_TXSEL | | | | SPI3_RXSEL | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| SPI2_TXSEL | | | | SPI2_RXSEL | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| SPI1_TXSEL | | | | SPI1_RXSEL | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SPI0_TXSEL | | | | SPI0_RXSEL | | | |

| Bits | 描述 | |
|---|---|---|
| [31:28] | **SPI3_TXSEL** | **PDMA SPI3 Tx 选择**<br>PDMA 信道连接片上外设 SPI3 TX. 软件配置 TX 信道 通过设定 SPI3_TXSEL. 该信道配置同 SPI0_RXSEL. 可以参考 SPI0_RXSEL配置设定. |
| [27:24] | **SPI3_RXSEL** | **PDMA SPI3 Rx 选择**<br>PDMA 信道连接片上外设 SPI3 RX. 软件配置 RX 信道 通过设定 SPI3_RXSEL. 该信道配置同 SPI0_RXSEL . 可以参考 SPI0_RXSEL配置设定. |
| [23:20] | **SPI2_TXSEL** | **PDMA SPI2 Tx 选择**<br>PDMA 信道连接片上外设 SPI2 TX. 软件配置 TX 信道 通过设定 SPI2_TXSEL. 该信道配置同 SPI0_RXSEL. 可以参考 SPI0_RXSEL配置设定. |
| [19:16] | **SPI2_RXSEL** | **PDMA SPI2 Rx 选择**<br>PDMA 信道连接片上外设 SPI2 RX. 软件配置 RX 信道 通过设定 SPI2_RXSEL. 该信道配置同 SPI0_RXSEL. 可以参考 SPI0_RXSEL配置设定. |
| [15:12] | **SPI1_TXSEL** | **PDMA SPI1 Tx 选择**<br>PDMA 信道连接片上外设 SPI1 TX. 软件配置 TX 信道 通过设定 SPI1_TXSEL. 该信道配置同 SPI0_RXSEL. 可以参考 SPI0_RXSEL配置设定. |
| [11:8] | **SPI1_RXSEL** | **PDMA SPI1 Rx 选择**<br>PDMA 信道连接片上外设 SPI1 RX. 软件配置 RX 信道 通过设定 SPI1_RXSEL. 该信道配置同 SPI0_RXSEL. 可以参考 SPI0_RXSEL配置设定. |

| | | |
|---|---|---|
| [7:4] | **SPI0_TXSEL** | **PDMA SPI0 Tx 选择**<br>PDMA 信道连接片上外设 SPI0 TX. 软件配置 TX 信道 通过设定 SPI0_TXSEL. 该信道配置同 SPI0_RXSEL. 可以参考 SPI0_RXSEL配置设定. |
| [3:0] | **SPI0_RXSEL** | **PDMA SPI0 Rx 选择**<br>PDMA 信道连接片上外设 SPI0 RX. 可通过软件改变 RX 通道设定（ SPI0_RXSEL）<br>4’b0000: CH0<br>4’b0001: CH1<br>4’b0010: CH2<br>4’b0011: CH3<br>4’b0100: CH4<br>4’b0101: CH5<br>4’b0110: CH6<br>4’b0111: CH7<br>4’b1000: CH8<br>Others ：保留<br>**注：**Ex : SPI0_RXSEL = 4’b0110, 意味 SPI0_RX 和 PDMA_CH6相连接 |

### PDMA服务选择控制寄存器1 (PDSSR1)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| PDSSR1 | PDMA_BA_GCR +0x08 | R/W | EDMA服务选择控制寄存器1 | 0xFFFF_FFFF |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| ADC_TXSEL | | | | ADC_RXSEL | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| USB_TXSEL | | | | USB_RXSEL | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| UART1_TXSEL | | | | UART1_RXSEL | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| UART0_TXSEL | | | | UART0_RXSEL | | | |

| Bits | 描述 | |
|---|---|---|
| [31:28] | **ADC_TXSEL** | **PDMA ADC Tx 选择**<br>PDMA 信道和外设ADC TX相连接. 软件可配置 TX 信道 通过设定 XSEL. 该信道配置同 UART0_RXSEL. 可参考UART0_RXSEL解释 |
| [27:24] | **ADC_RXSEL** | **PDMA ADC Rx 选择**<br>PDMA 信道和外设ADC RX相连接. 软件可配置 RX信道 通过设定 ADC_RXSEL. 该信道配置同 UART0_RXSEL. 可参考UART0_RXSEL解释 |
| [23:20] | **USB_TXSEL** | **PDMA USB Tx 选择**<br>PDMA 信道和外设USB TX相连接. 软件可配置 TX信道 通过设定 SB_TXSEL. 该信道配置同 UART0_RXSEL. 可参考UART0_RXSEL解释 |
| [19:16] | **USB_RXSEL** | **PDMA USB Rx 选择**<br>PDMA 信道和外设USB RX相连接. 软件可配置 RX信道 通过设定 USB_RXSEL. 该信道配置同 UART0_RXSEL. 可参考UART0_RXSEL解释 |
| [15:12] | **UART1_TXSEL** | **PDMA UART1 Tx 选择**<br>PDMA 信道和外设UART1 TX相连接. 软件可配置 TX信号 通过设定UART1_TXSEL. T 该信道配置同 UART0_RXSEL . 可参考UART0_RXSEL解释 |
| [11:8] | **UART1_RXSEL** | **PDMA UART1 Rx 选择**<br>PDMA 信道和外设UART1 RX相连接. 软件可配置 RX信道 通过设定UART1_RXSEL. 该信道配置同UART0_RXSEL. 可参考UART0_RXSEL解释 |

| | | |
|---|---|---|
| [7:4] | **UART0_TXSEL** | **PDMA UART0 Tx 选择**<br>PDMA 信道和外设UART0 TX相连接. 软件可配置TX信号道 通过设定UART0_TXSEL. 该信道配置同 UART0_RXSEL . 可参考UART0_RXSEL解释 |
| [3:0] | **UART0_RXSEL** | PDMA 信道和外设UART0 RX相连接. 软件可配置RX 通过设定 UART0_RXSEL<br>4'b0000: CH0<br>4'b0001: CH1<br>4'b0010: CH2<br>4'b0011: CH3<br>4'b0100: CH4<br>4'b0101: CH5<br>4'b0110: CH6<br>4'b0111: CH7<br>4'b1000: CH8<br>Others ：保留<br>**注：**Ex : UART0_RXSEL = 4'b0110, 意味 UART0_RX和PDMA_CH6连接 |

### PDMA全局中断状态寄存器 (PDMA_GCRISR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| PDMA_GCRISR | PDMA_BA_GCR + 0x00C | R/W | PDMA 全局中断状态寄存器 | 0x0000_0000 |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| INTR | 保留 | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 保留 | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| 保留 | | | | | | | INTR8 |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| INTR7 | INTR6 | INTR5 | INTR4 | INTR3 | INTR2 | INTR1 | INTR0 |

| Bits | 描述 | |
|---|---|---|
| [31] | **INTR** | **中断引脚状态 (只读)**<br>该位为 PDMA 控制器中断引脚状态.<br>**注:** 该位只读 |
| [30:9] | 保留 | 保留 |
| [8] | **INTR8** | **中断引脚 8 状态 (只读)**<br>该位为 PDMA 控制器中断引脚 8 状态.<br>**注:** 该位只读 |
| [7] | **INTR7** | **中断引脚 7 状态 (只读)**<br>该位为 PDMA 控制器中断引脚 7 状态.<br>**注:** 该位只读 |
| [6] | **INTR6** | **中断引脚 6 状态 (只读)**<br>该位为 PDMA 控制器中断引脚 6 状态.<br>**注:** 该位只读 |
| [5] | **INTR5** | **中断引脚 5 状态 (只读)**<br>该位为 PDMA 控制器中断引脚 5 状态.<br>**注:** 该位只读 |

| | | |
|---|---|---|
| [4] | **INTR4** | **中断引脚 4 状态 (只读)**<br>该位为 PDMA 控制器中断引脚4 状态.<br>**注:** 该位只读 |
| [3] | **INTR3** | **中断引脚 3 状态 (只读)**<br>该位为 PDMA 控制器中断引脚 3 状态.<br>**注:** 该位只读 |
| [2] | **INTR2** | **中断引脚 2 状态 (只读)**<br>该位为 PDMA 控制器中断引脚 2 状态.<br>**注:** 该位只读 |
| [1] | **INTR1** | **中断引脚 1 状态 (只读)**<br>该位为 PDMA 控制器中断引脚 1 状态.<br>**注:** 该位只读 |
| [0] | **INTR0** | **中断引脚 0 状态 (只读)**<br>该位为 PDMA 控制器中断引脚 0 状态.<br>**注:** 该位只读 |

### PDMA 服务选择控制寄存器2 (PDSSR2)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| PDSSR2 | PDMA_BA_GCR +0x10 | R/W | EDMA服务选择控制寄存器2 | 0x0000_00FF |

| 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 |
| I2S_TXSEL | | | | I2S_RXSEL | | | |

| Bits | 描述 | |
|---|---|---|
| [31:8] | 保留 | 保留 |
| [7:4] | **I2S_TXSEL** | **PDMA I2S Tx选择**<br>PDMA 信道和外设I2S TX相连接. 软件可配置 TX 信道 通过设定I2S_TXSEL. 该信道配置同 I2S _RXSEL. 可参考I2S _RXSEL解释 |
| [3:0] | **I2S_RXSEL** | **PDMA I2S Rx 选择**<br>PDMA 信道连接片上外设 **I2S** RX. 可通过软件改变 RX 通道设定（ I2S _RXSEL）<br>4’b0000: CH0<br>4’b0001: CH1<br>4’b0010: CH2<br>4’b0011: CH3<br>4’b0100: CH4<br>4’b0101: CH5<br>4’b0110: CH6<br>4’b0111: CH7<br>4’b1000: CH8<br>其他 ：保留<br>**注：**Ex : I2S _RXSEL = 4’b0110, 意味 I2S _RX 和 PDMA_CH6相连接 |

# 6 FLASH 内存控制器 (FMC)

## 6.1 概述

NUC1XX 系列具有128/64/32k 字节的片上FLASH，用于存储应用程序（APROM）,用户可以通过ISP/IAP更新FLASH中的程序. 在系统编程 (ISP) 允许用户更新焊接在PCB板上的芯片中的程序. 上电后，通过设置Config0确定 Cortex-M0 CPU 从APROM或LDROM读取代码. 对于64/32K 字节的器件，NUC1XX 系列为用户提供4k字节的数据FLASH用于存储一些应用所需的数据. 对于128k 字节的器件, 数据flash与128K的程序内存共享，其开始地址是可配置的，由用户在Config1中定义. 用户可以根据其应用的需要来定义数据FLASH的大小.

## 6.2 特征

- 可兼容AHB 接口
- 零等待状态，可达50MHz的不连续的地址访问
- 128/64/32kB 应用程序内存 (APROM)
- 4kB 在系统编程 (ISP) 加载程序内存 (LDROM)
- 可配置的或不易失的4kB数据FLASH，带有512字节页擦除单元
- 对于128K程序内存的器件，有起始地址和内存大小可编程的数据FLASH
- 在系统编程 (ISP)/在应用编程 (IAP) 用于更新片上FLASH EPROM

## 6.3 Flash 内存控制器框图

FLASH内存控制器由AHB从接口，ISP控制逻辑，烧写接口和FLASH宏接口时序控制逻辑组成. FLASH内存控制器框图如下所示:

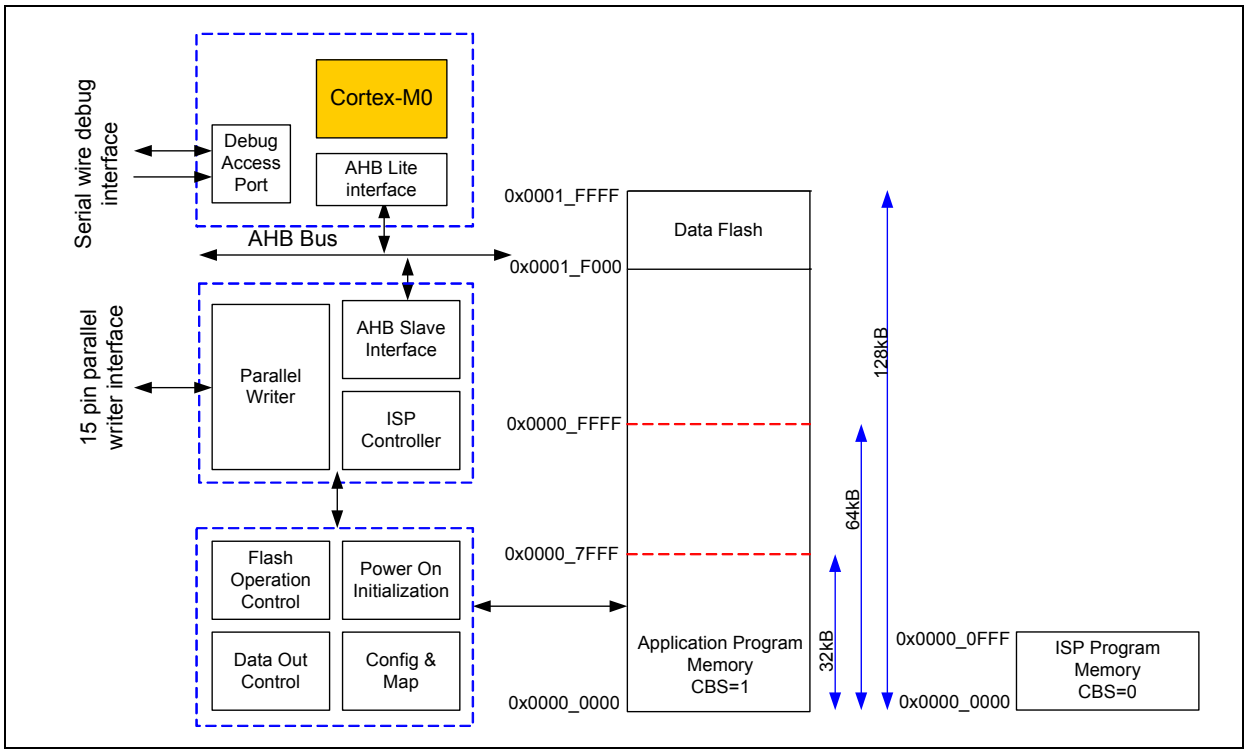



图 6-1 Flash 内存控制框图

## 6.4 Flash 内存结构

NUC1XX 的flash 内存由程序内存（128/64/32kB），数据FLASH，ISP加载程序内存，用户配置组成. 用户配置块提供几个字节控制系统逻辑，如flash安全加密，启动选择，欠压电平，数据FLASH基地址等. 在上电期间，由FLASH内存加载到相应的控制寄存器中，用户根据烧写器的应用要求设置这些位，不用将芯片从PCB板上取下，数据FLASH的开始地址和大小可由用户根据应用定义，对于64/32KB的FLASH内存器件，其大小为4KB，开始地址为0x0001_F000..

表 6-1内存地址图

| 区块名称 | 大小 | 开始地址 | 结束地址 |
|---|---|---|---|
| AP-ROM | 32/64/(128-0.5*N)kB | 0x0000_0000 | 0x0000_7FFF (32kB)<br>0x0000_FFFF (64kB)<br>DFBADR-1 (128kB if DFEN=0) |
| 保留 | 896kB | 0x0002_0000 | 0x000F_FFFF |
| Data Flash | 4/4/0.5*N KB | 0x0001_F000<br>DFBADR | 0x0001_FFFF |
| LD-ROM | 4 KB | 0x0010_0000 | 0x0010_0FFF |
| 用户配置 | 2 words | 0x0030_0000 | 0x0030_0004 |

Flash内存组成如下所示:

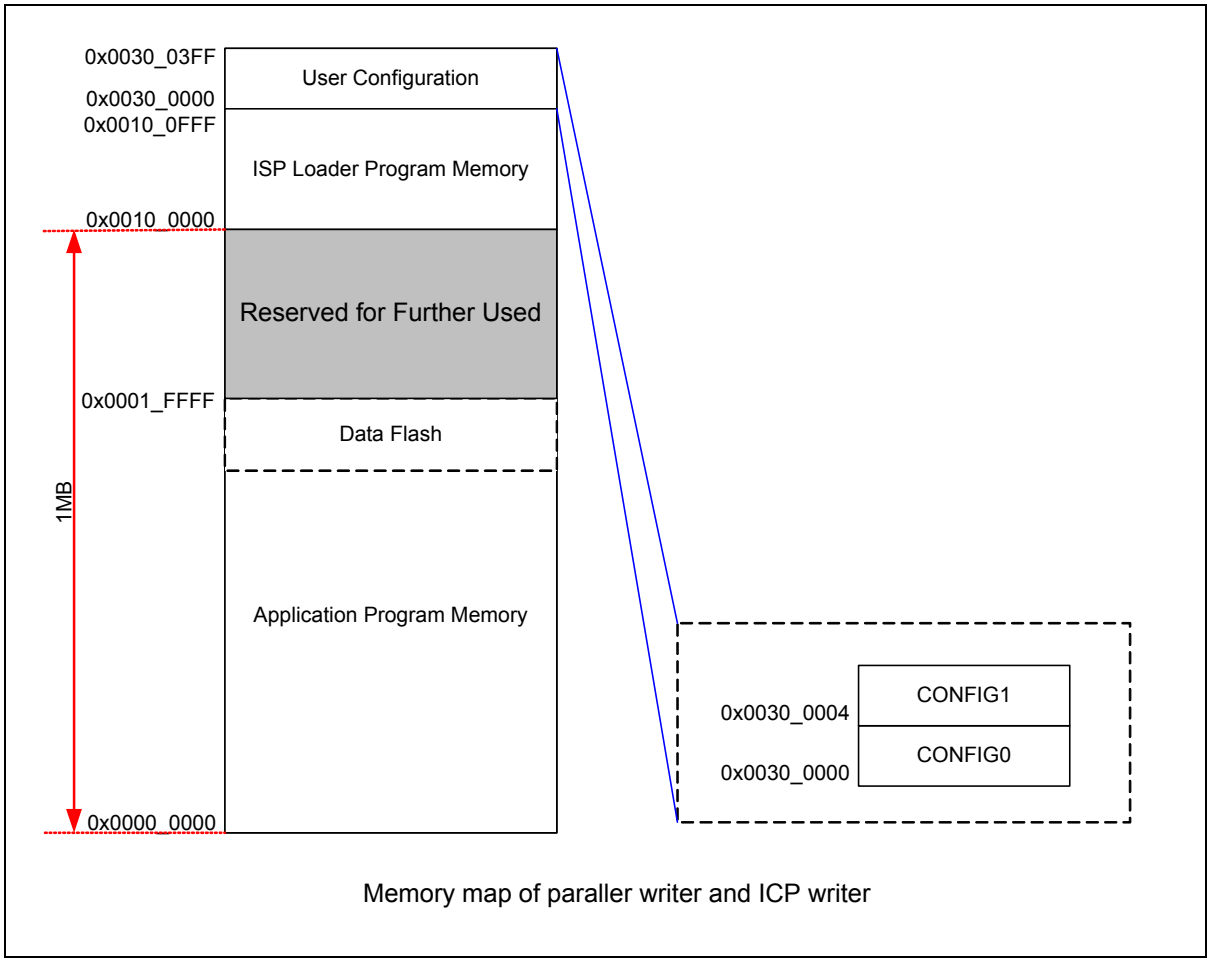



图 6-2 Flash 内存组成

## 6.5 启动选择

NUC1XX 提供在系统编程 (ISP) 特征，允许用户直接更新PCB板上芯片中的程序. 提供4kB程序内存用于存储ISP固件. 用户设置Config0以选择从AP_ROM或LD_ROM开始.

## 6.6 数据 Flash

NUC1XX 为用户提供数据FLASH. 通过ISP程序读/写. 擦除单元为512字节. 若要改变一个字，需要先把所有128字拷贝到另外页或SRAM中. 对于 128k flash 器件，数据flash与应用程序共享128K的内存, 若Config0的DNDF位使能，数据FLASH的基地址由DFBADR定义，应用程序的内存大小为(128-0.5*N)kB，对于64/32KB的FLASH内存器件，数据FLASH的大小为4KB，开始地址为0x0001_F000.

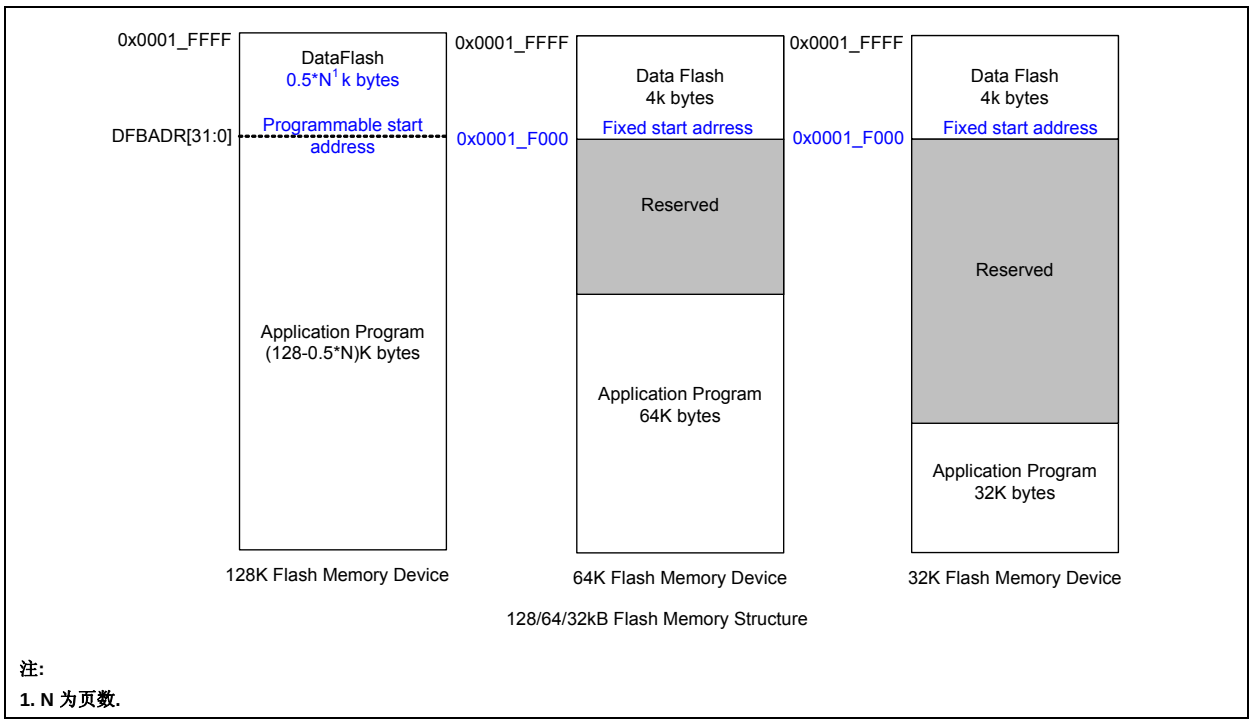



图 6-3 Flash 内存结构

## 6.7 用户配置

**Config0 (地址 = 0x0030_0000)**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| CWDTEN | - | - | CKF | - | CFOSC | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| CBODEN | CBOV1 | CBOV0 | CBORST | - | - | - | - |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| - | - | - | - | - | - | - | - |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CBS | - | - | - | - | - | LOCK | DFEN |

| Bits | 描述 | |
|---|---|---|
| [31] | **CWDTEN** | **看门狗 使能**<br>0= 上电后，使能看门狗<br>1=上电后，禁止看门狗 |
| [30:29] | 保留 | 保留 |
| [28] | **CKF** | **XT1时钟滤波使能**<br>0 = 禁止时钟滤波<br>1 = 使能时钟滤波 |
| [27] | 保留 | 保留 for further used |
| [26:24] | **CFOSC** | **复位后CPU时钟源选择**<br>FOSC[2:0] 时钟源<br>000 外部 12MHz 晶振时钟<br>111 内部 RC 22MHz 振荡器时钟<br>Others **保留**<br>复位后，CFOSC的值将被加载到系统寄存器CLKSEL0.HCLK_S[2:0]. |
| [23] | **CBODEN** | **欠压检测使能**<br>0= 上电后，使能欠压检测<br>1= 上电后，禁止欠压检测 |

<table>
<tr><td>[22:21]</td><td>CBOV1-0</td><td>欠压极端电压选择<table><tr><td>CBOV1</td><td>CBOV0</td><td>欠压电压</td></tr><tr><td>0</td><td>0</td><td>4.5v</td></tr><tr><td>0</td><td>1</td><td>3.8v</td></tr><tr><td>1</td><td>0</td><td>2.6v</td></tr><tr><td>1</td><td>1</td><td>2.2v</td></tr></table></td></tr>
<tr><td>[20]</td><td>CBORST</td><td>欠压复位使能<br>0 = 上电后，使能欠压复位<br>1 = 上电后，禁止欠压复位</td></tr>
<tr><td>[19:8]</td><td>保留</td><td>保留</td></tr>
<tr><td>[7]</td><td>CBS</td><td>启动选项<br>0 = 芯片由LDROM启动<br>1 = 芯片由APROM启动</td></tr>
<tr><td>[6:2]</td><td>保留</td><td>保留</td></tr>
<tr><td>[1]</td><td>LOCK</td><td>安全加密<br>0 = 加密FLASH数据<br>1 = 解除Flash 数据加密<br>当FLASH数据被加密，仅有器件ID， Config0 和Config1 可被烧写器和ICP通过串口调试接口读取. 其他数据锁定为0xFFFFFFFF. 无论数据是否锁定，ISP 都可以读取.</td></tr>
<tr><td>[0]</td><td>DFEN</td><td>数据FLASH使能<br>0 = 使能数据FLASH<br>1 = 禁止数据FLASH</td></tr>
</table>

**Config1 (Address = 0x0030_0004)**

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

| Bits | 描述 | |
|---|---|---|
| [31:20] | 保留 | **保留**<br>强行给这些保留位写入0x00 |
| [19:0] | **DFBA** | **数据FLASH的基地址**<br>对于 128kB flash 器件, 其数据FLASH的基地址由用户定义.片上FLASH擦除单元为512字节, bit8-0位为0 , 对于64/32KB的器件，不采用这种配置. |

## 6.8 在系统编程(ISP)

程序内存和数据FLASH支持硬件编程和在系统编程 (ISP). 硬件编程模式采用批量写，以方便量产阶段进行编程。若产品还在开发阶段或终端用户需要升级固件时，硬件编程模式不是很方便，ISP模式能更好地适用于这种情况。NUC1XX 支持 ISP 模式，即通过软件控制来对器件重新编程. 而且，这也使得更新固件得以广泛应用。

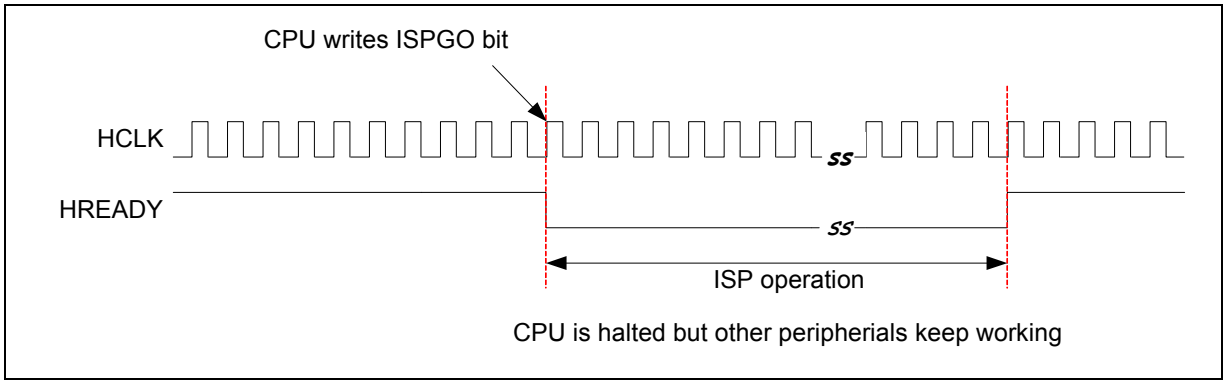
ISP 可以在没有将微控器从系统中取下来的情况下执行编程. 各种接口使得LDROM更容易更新程序代码. 最常用的方法是通过UART执行ISP，PC一般都是通过串口传输新 的APROM代码. LDROM接收后，通过ISP，重新对APROM编程. Nuvoton 提供用于NUC1XX的ISP 固件和 PC 应用程序. 用户采用Nuvoton ISP工具可以非常方便地执行ISP

### 6.8.1 ISP 程序

NUC1XX 支持从APROM 或 LDROM启动. 用户想更新APROM中的应用程序时，可以写BS=1，并开始软件复位使芯片由LDROM启动. 向ISPEN写入1开始ISP功能. 在向ISPCON寄存器写数据之前，S/W 需要向全局控制寄存器（GCR, 0x5000_0100）的寄存器 RegLockAddr 写入0x59, 0x16 和 0x88，这个过程用于保护FLASH内存免受意外更改.

向ISPGO向入数据后，要检查几个错误条件. 如果错误条件产生时，ISP操作失败，其失败标志置位，ISPFF标志由软件清零，而不会在下次ISP操作时被覆盖，即使ISPFF保持为“1“，下一次ISP也可以开始. 建议在每次ISP操作后，如果ISPFF被设置为1了，就软件检查ISPFF并将其清零。.

ISPGO置位，CPU将等待ISP操作结束，在此期间，外设仍然在工作，如果有中断请求时，CPU仍然会先执行完ISP后再响应中断.




**注** NUC1XX 允许用户通过ISP更新CONFIG的值, 基于对应用程序安全考虑，软件在擦除CONFIG时，要先页擦除APROM，否则CONFIG不能被擦除.

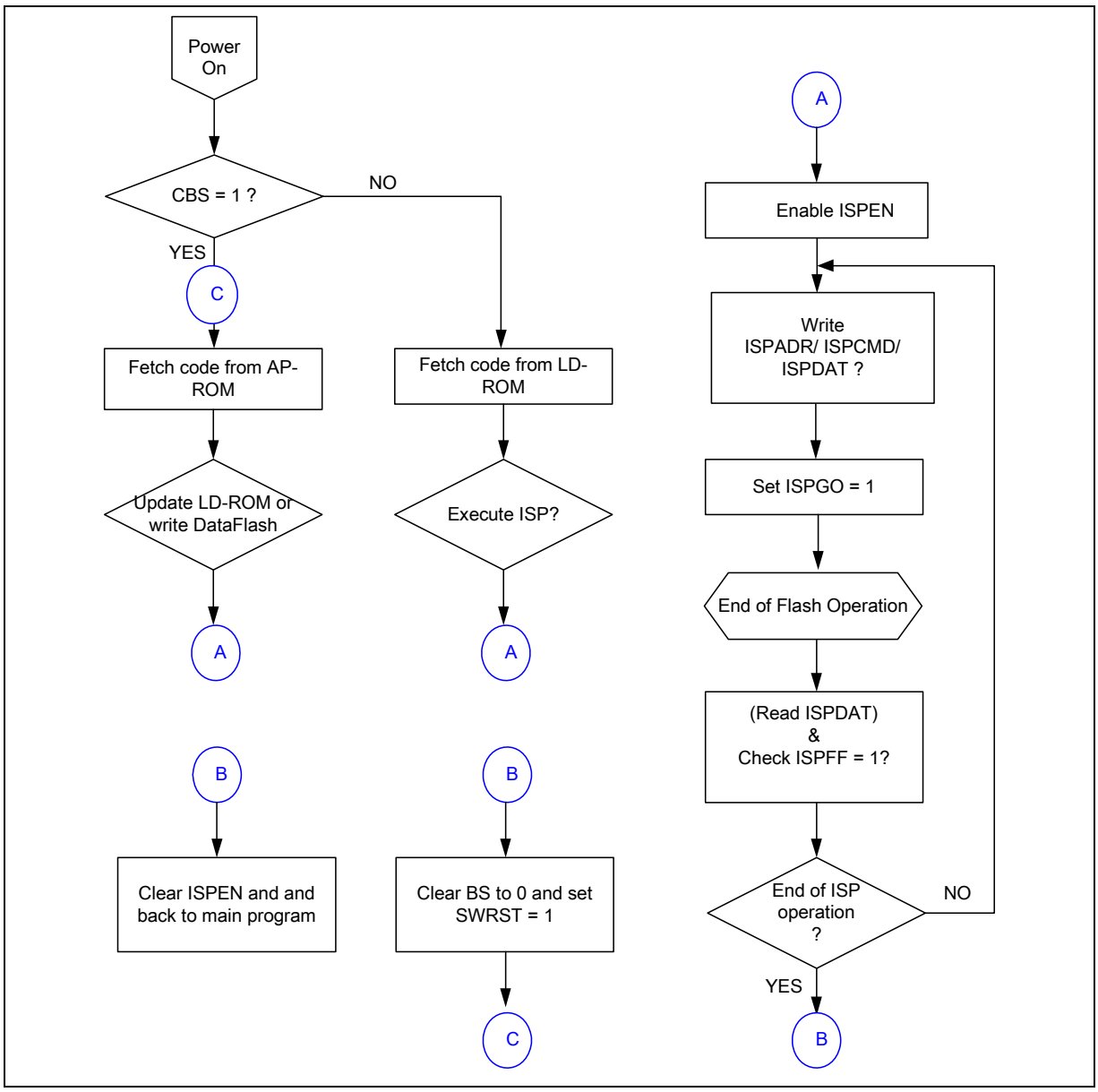

Power On
CBS = 1 ?
NO
YES
C
Fetch code from AP-ROM
Update LD-ROM or write DataFlash
A
B
Clear ISPEN and and back to main program
Fetch code from LD-ROM
Execute ISP?
A
B
Clear BS to 0 and set SWRST = 1
C
A
Enable ISPEN
Write ISPADR/ ISPCMD/ ISPDAT ?
Set ISPGO = 1
End of Flash Operation
(Read ISPDAT) & Check ISPFF = 1?
End of ISP operation ?
NO
YES
B

表 6-2 ISP模式

| ISP Mode | ISPCMD | | | ISPADR | | | ISPDAT |
|---|---|---|---|---|---|---|---|
| | FOEN | FCEN | FCTRL[3:0] | A21 | A20 | A[19:0] | D[31:0] |
| Standby | 1 | 1 | X | x | x | x | x |
| Read Company ID | 0 | 0 | 1011 | x | x | x | Data out D[31:0] = 0x0000_00DA |
| Read Device ID | 0 | 0 | 1100 | x | x | Address in A[19:0] = 0x00000 | Data out D[31:0] = 0x0000_0E80 |
| FLASH Page Erase | 1 | 0 | 0010 | 0 | A20 | Address in A[19:0] | x |
| FLASH Program | 1 | 0 | 0001 | 0 | A20 | Address in A[19:0] | Data in D[31:0] |
| FLASH Read | 0 | 0 | 0000 | 0 | A20 | Address in A[19:0] | Data out D[31:0] |
| CONFIG Page Erase | 1 | 0 | 0010 | 1 | 1 | Address in A[19:0] | x |
| CONFIG Program | 1 | 0 | 0001 | 1 | 1 | Address in A[19:0] | Data in D[31:0] |
| CONFIG Read | 0 | 0 | 0000 | 1 | 1 | Address in A[19:0] | Data out D[31:0] |

## 6.9 Flash控制寄存器图

**R: 只读, W: 只写, R/W: 可读写**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **Base Address (FMC_BA) : 0x5000_C000** | | | | |
| **ISPCON** | FMC_BA+0x000 | R/W | ISP控制寄存器 | 0x0000_0000 |
| **ISPADR** | FMC_BA+0x004 | R/W | ISP地址寄存器 | 0x0000_0000 |
| **ISPDAT** | FMC_BA+0x008 | R/W | ISP数据寄存器 | 0x0000_0000 |
| **ISPCMD** | FMC_BA+0x00C | R/W | ISP命令寄存器 | 0x0000_0000 |
| **ISPTRG** | FMC_BA+0x010 | R/W | ISP触发寄存器 | 0x0000_0000 |
| **DFBADR** | FMC_BA+0x014 | R | 数据FLASH开始地址(AP ROM小于128KB) | 0x0001_f000 |
| **DFBADR** | FMC_BA+0x014 | R | 数据FLASH开始地址(AP ROM 等于 128KB) | 0x0000_0000 |
| **FATCON** | FMC_BA+0x018 | R/W | FLASH访问窗口控制寄存器 | 0x0000_0000 |

## 6.10 Flash 控制寄存器描述

**ISP控制寄存器(ISPCON)**

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **ISPCON** | FMC_BA+0x00 | R/W | ISP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** |
| **-** | **ET2** | **ET1** | **ET0** | **-** | **PT2** | **PT1** | **PT0** |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **SWRST** | **ISPFF** | **LDUEN** | **-** | **-** | **-** | **BS** | **ISPEN** |

| Bits | 描述 | |
|---|---|---|
| [31:15] | 保留 | 保留 |
| [14:12] | **ET[2:0]** | **Flash擦除时间**<br>ET[2] / ET[1] / ET[0] / **擦除时间**(ms)<br>0 / 0 / 0 / 20 (default)<br>0 / 0 / 1 / 25<br>0 / 1 / 0 / 30<br>0 / 1 / 1 / 35<br>1 / 0 / 0 / 3<br>1 / 0 / 1 / 5<br>1 / 1 / 0 / 10<br>1 / 1 / 1 / 15 |
| [11] | 保留 | 保留 |

<table>
<tr><td>[8:10]</td><td>PT[2:0]</td><td>Flash编程时间
<table>
<tr><td>PT[2]</td><td>PT[1]</td><td>PT[0]</td><td>编程时间 (us)</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>40</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>45</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>50</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>55</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>20</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>25</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>30</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>35</td></tr>
</table></td></tr>
<tr><td>[7]</td><td>SWRST</td><td>软件复位<br>写1执行软件复位.<br>复位完成后由硬件清零.</td></tr>
<tr><td>[6]</td><td>ISPFF</td><td>ISP失败标志<br>当ISP满足下列条件时，该位由硬件置位:<br>(1) APROM 写入本身.<br>(2) LDROM 写入本身.<br>(3) MCU在APROM中运行时，CONFIG 被擦除或编程.<br>(4) 定义地址无效，如超过正常范围.<br>写 1 清标志.</td></tr>
<tr><td>[5]</td><td>LDUEN</td><td>LDROM更新使能<br>LDROM 更新使能位.<br>1 = MCU在APROM中运行时，LDROM 被更新.<br>0 = 禁止LDROM更新</td></tr>
<tr><td>[2]</td><td>保留</td><td>保留</td></tr>
<tr><td>[1]</td><td>BS</td><td>启动选择<br>置位/清零该位选择下次是由LDROM启动还是由APROM启动, 该位可作为MCU启动状态标志,用于检查MCU是由LDROM还是APROM启动的.<br>1 = 由LDROM启动<br>0 = 由 APROM启动</td></tr>
</table>

| | | |
|---|---|---|
| [0] | **ISPEN** | **ISP 使能**<br>ISP 使能位，设置该位可以使能ISP功能.<br>1 = 使能 ISP 功能<br>0 = 禁止 ISP 功能 |

### ISP地址(ISPADR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| ISPADR | FMC_BA+ 0x04 | R/W | ISP 地址寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:0] | **ISPADR** | **ISP 地址**<br>NUC1XX 系列内置32kx32 的flash, 仅支持字编程. 执行ISP功能时, ISPARD[1:0] 必须为00b. |

### ISPDAT (ISP数据寄存器)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| **ISPDAT** | FMC_BA+ 0x08 | R/W | ISP数据寄存器 | 0x0000_0000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:0] | **ISPDAT** | **ISP 数据**<br>ISP操作之前，写数据到该寄存器<br>ISP读操作后，可从该寄存器读数据 |

### ISP命令(ISPCMD)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| ISPCMD | FMC_BA+ 0x1C | R/W | ISP 命令寄存器 | 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 | | FOEN | FCEN | FCTRL3 | FCTRL2 | FCTRL1 | FCTRL0 |

| Bits | 描述 | |
|---|---|---|
| [31:6] | 保留 | 保留 |
| [5:0] | **FOEN, FCEN, FCTRL** | **ISP 命令**<br>ISP 命令与写模式相似，但不支持整颗芯片擦除.<br><br>操作模式 / FOEN / FCEN / FCTRL[3:0]<br>待机: 1 1 0 0 0 0<br>读: 0 0 0 0 0 0<br>编程: 1 0 0 0 0 1<br>页擦除: 1 0 0 0 1 0<br>读 CID: 0 0 1 0 1 1<br>读 DID: 0 0 1 1 0 0 |

### ISP 触发控制寄存器(ISPTRG)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| ISPTRG | FMC_BA+ 0x10 | R/W | ISP 触发控制寄存器 | 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 | | | | | | | ISPGO |

| Bits | 描述 | |
|---|---|---|
| [31:1] | 保留 | 保留 |
| [0] | **ISPGO** | **ISP开始触发**<br>写 1 开始ISP操作，当ISP操作结束后，该位由硬件自动清零.<br>1 = ISP 正在执行<br>0 = ISP 操作结束 |

### 数据FLASH基地址寄存器(DFBADR)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| DFBADR | FMC_BA+ 0x14 | R | 数据FLASH基地址 | 0x0001_F000 |

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

| Bits | 描述 | |
|---|---|---|
| [31:0] | **DFBA** | **数据FLASH基地址**<br>该寄存器为数据FLASH开始地址寄存器,只读.<br>对于128kB flash内存的器件, 数据FLASH的大小由用户配置定义, 对于64/32KB的器件，寄存器的值在芯片上电后由Config1加载，地址为0x01_f000 |

### Flash访问时间控制寄存器(FATCON)

| 寄存器 | 偏移量 | R/W | 描述 | 复位后的值 |
|---|---|---|---|---|
| FATCON | 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 |
|---|---|---|---|---|---|---|---|
| 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 | | | | FATS[2:0] | | | FPSEN |

| Bits | 描述 | |
|---|---|---|
| [31:4] | 保留 | 保留 |
| [3:1] | **FATS** | **FLASH访问时间窗口选择**<br>这些位决定FLASH的检测放大器持续有效时间，对于tsmc 0.25um 的内置FLASH，其值约为40ns. |

| FATS | Access Time window (ns) |
|---|---|
| 000 | 40 |
| 001 | 50 |
| 010 | 60 |
| 011 | 70 |
| 100 | 80 |
| 101 | 90 |
| 110 | 100 |
| 111 | 保留 |

| | | |
|---|---|---|
| [0] | **FPSEN** | **Flash 省电使能**<br>在FLASH内存访问时间大约为40ns,如果CPU时钟低于50MHz, S/W可以使能省电功能.<br>1 = 使能FLASH省电功能<br>0 = 禁止FLASH省电功能 |

# 7 电气特性

## 7.1 绝对最大额定值

| 参数 | 符号 | 最小值 | 最大值 | 单位 |
|---|---|---|---|---|
| 直流电源电压 | VDD–VSS | -0.3 | +7.0 | V |
| 输入电压 | VIN | VSS-0.3 | VDD+0.3 | V |
| 晶振频率 | $1/t_{CLCL}$ | 0 | 40 | MHz |
| 工作温度 | TA | -40 | +85 | °C |
| 贮存温度 | TST | -55 | +150 | °C |
| $V_{DD}$最大流入电流 | | - | 120 | mA |
| $V_{SS}$最大流出电流 | | | 120 | mA |
| 单一管脚最大灌电流 | | | 35 | mA |
| 单一管脚最大源电流 | | | 35 | mA |
| 所有管脚最大灌电流总合 | | | 100 | mA |
| 所有管脚最大源电流总合 | | | 100 | mA |

**注:** 上表所列的条件中，其极限值可能对器件的稳定有反作用.

## 7.2 DC 电气特性

(VDD-VSS=3.3V, TA = 25°C, $F_{OSC}$ = 50Mhz 在无特别说明的情况下.)

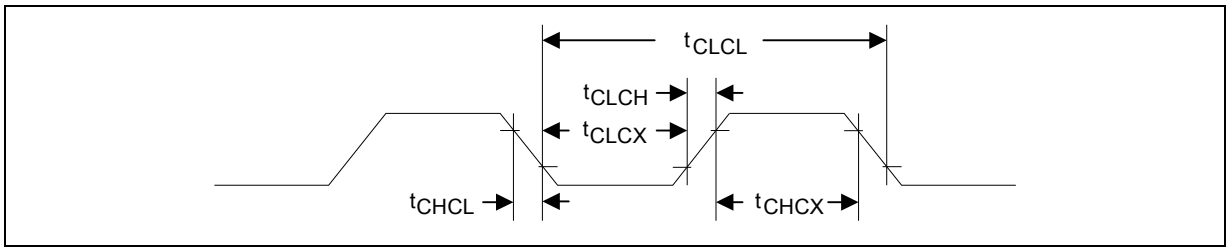
| 参数 | 符号. | 明细表 | | | | 测试条件 |
|---|---|---|---|---|---|---|
| | | 最小值 | 典型值 | 最大值 | 单位 | |
| 工作电压 | $V_{DD}$ | 2.5 | | 5.5 | V | $V_{DD}$ =2.5V ~ 5.5V up to 50MHz |
| 电源地 | $V_{SS}$<br>$AV_{SS}$ | -0.3 | | | V | |
| LDO 输出电压<br>(bypass = 0) | $V_{LDO}$ | -10% | 2.45 | +10% | V | $V_{DD}$ > 2.7V |
| LDO 输出电压<br>(bypass = 0) | $V_{LDO}$ | -10% | $V_{DD}$ | +10% | V | $V_{DD}$ < 2.7V |
| 模拟工作电压 | $AV_{DD}$ | 0 | | $V_{DD}$ | V | |
| 模拟参考电压 | Vref | 0 | | $AV_{DD}$ | V | |
| 普通模式下的工作电流 | $I_{DD1}$ | | 50 | | mA | $V_{DD}$= 5.5V@50Mhz , enable all IP |
| | $I_{DD2}$ | | 33 | | mA | $V_{DD}$=5.5V@50Mhz , disable all IP |
| | $I_{DD3}$ | | 44 | | mA | $V_{DD}$ = 3V@50Mhz , enable all IP |
| | $I_{DD4}$ | | 27 | | mA | $V_{DD}$ = 3V@50Mhz , disable all IP |
| 空闲模式下的工作电流<br>(Idle 模式) | $I_{IDLE1}$ | | 35 | | mA | $V_{DD}$= 5.5V@50Mhz , enable all IP |
| | $I_{IDLE2}$ | | 17 | | mA | $V_{DD}$=5.5V@50Mhz , disable all IP |
| | $I_{IDLE3}$ | | 33 | | mA | $V_{DD}$ = 3V@50Mhz , enable all IP |
| | $I_{IDLE4}$ | | 15 | | mA | $V_{DD}$ = 3V@50Mhz , disable all IP |
| 掉电模式下工作电流<br>(深度休眠模式) | $I_{PWD1}$ | | 20 | | μA | $V_{DD}$ = 5.5V, No load<br>@ Disable BOV function |
| | $I_{PWD2}$ | | 18 | | μA | $V_{DD}$ = 3.3V, No load<br>@ Disable BOV function |
| | $I_{PWD3}$ | | 25 | | μA | $V_{DD}$ = 5.5V, RTC run<br>@ Disable BOV function |
| | $I_{PWD4}$ | | 23 | | μA | $V_{DD}$ = 3.3V, RTC run<br>@ Disable BOV function |
| PA~PE输入电流 | $I_{IN1}$ | -60 | - | +15 | μA | $V_{DD}$ = 5.5V, $V_{IN}$ = 0V or $V_{IN}$=$V_{DD}$ |
| RESET[1]脚输入电流 | $I_{IN2}$ | -55 | -45 | -30 | μA | $V_{DD}$ = 5.5V, $V_{IN}$ = 0.45V |
| PA~PE 输入漏电流 | $I_{LK}$ | -2 | - | +2 | μA | $V_{DD}$ = 5.5V, 0<$V_{IN}$<$V_{DD}$ |
| PA~PE 逻辑1至0转换时电流<br>(准双向模式) | $I_{TL}$ [3] | -650 | - | -200 | μA | $V_{DD}$ = 5.5V, $V_{IN}$<2.0V |
| P0, P1, P2, P3, P4输入低电压 | $V_{IL1}$ | -0.3 | - | 1.0 | V | $V_{DD}$ = 4.5V |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | -0.3 | - | 0.6 | | $V_{DD}$ = 2.5V |
| P0, P1, P2, P3, P4 输入高电压 (TTL 输入) | $V_{IH1}$ | 2.2 | - | $V_{DD}$ +0.2 | V | $V_{DD}$ = 5.5V |
| | | 1.5 | - | $V_{DD}$ +0.2 | | $V_{DD}$ =3.0V |
| XT1脚 输入低电平[*2] | $V_{IL3}$ | 0 | - | 0.8 | V | $V_{DD}$ = 4.5V |
| | | 0 | - | 0.4 | | $V_{DD}$ = 3.0V |
| XT1脚 输入高电平[*2] | $V_{IH3}$ | 3.5 | - | $V_{DD}$ +0.2 | V | $V_{DD}$ = 5.5V |
| | | 2.4 | - | $V_{DD}$ +0.2 | | $V_{DD}$ = 3.0V |
| X32I 脚 输入低电平[*2] | $V_{IL4}$ | 0 | - | 0.8 | | |
| | | 0 | - | 0.4 | | |
| X32O脚 输入高电平[*2] | $V_{IH4}$ | 3.5 | - | $V_{DD}$ +0.2 | | |
| | | 2.4 | - | $V_{DD}$ +0.2 | | |
| RESET脚 负向门槛电压 (Schmitt输入) | $V_{ILS}$ | -0.5 | - | $0.3V_{DD}$ | V | |
| RESET脚 正向门槛电压 (Schmitt输入) | $V_{IHS}$ | $0.7V_{DD}$ | - | $V_{DD}$+0.5 | V | |
| 内部 /RST 脚上拉电阻 | $R_{RST}$ | 50 | | 100 | KΩ | |
| 迟滞电压 | $V_{HY}$ | | $0.2V_{DD}$ | | V | |
| PA~PE 源电流 (准双向模式) | $I_{SR11}$ | -300 | -370 | -450 | μA | $V_{DD}$ = 4.5V, $V_S$ = 2.4V |
| | $I_{SR12}$ | -50 | -70 | -90 | μA | $V_{DD}$ = 2.7V, $V_S$ = 2.2V |
| | $I_{SR12}$ | -40 | -60 | -80 | μA | $V_{DD}$ = 2.5V, $V_S$ = 2.0V |
| PA~PE 源电流 (推挽模式) | $I_{SR21}$ | -20 | -24 | -28 | mA | $V_{DD}$ = 4.5V, $V_S$ = 3.0V |
| | $I_{SR22}$ | -4 | -6 | -8 | mA | $V_{DD}$ = 2.7V, $V_S$ = 2.2V |
| | $I_{SR22}$ | -3 | -5 | -7 | mA | $V_{DD}$ = 2.5V, $V_S$ = 2.0V |
| PA~PE 灌电流 (准双向及推挽模式) | $I_{SK1}$ | 10 | 16 | 20 | mA | $V_{DD}$ = 4.5V, $V_S$ = 0.45V |
| | $I_{SK1}$ | 7 | 10 | 13 | mA | $V_{DD}$ = 2.7V, $V_S$ = 0.45V |
| | $I_{SK1}$ | 6 | 9 | 12 | mA | $V_{DD}$ = 2.5V, $V_S$ = 0.45V |
| BOV_VL [1:0] =00b 时 复位电压 | $V_{BO2.2}$ | 2.1 | 2.2 | 2.3 | V | |
| BOV_VL [1:0] =01b 时 复位电压 | $V_{BO2.7}$ | 2.6 | 2.7 | 2.8 | V | |
| BOV_VL [1:0] =10b 时 复位电压 | $V_{BO3.8}$ | 3.7 | 3.8 | 3.9 | V | |
| BOV_VL [1:0] =11b 时 复位电压 | $V_{BO4.5}$ | 4.4 | 4.5 | 4.6 | V | |
| BOD电压迟滞范围 | $V_{BH}$ | 30 | - | 150 | mV | $V_{DD}$ = 2.4V~5.5V |

**注s:**

1. /RST 管脚为 Schmitt 触发 输入模式.
2. XTAL1 为 CMOS 输入模式.
3. 当P0, P1, P2, P3 及 P4 管脚被外部由1驱动到0时， 可作来输出电流的源端，在$V_{DD}$=5.5V时, 输出电流达到最大值， Vin 接近2V .

## 7.3 AC电气特性




*注: 占空比为 50%.*

| 参数 | 符号 | 最小值 | 典型值 | 最大值 | 单位 | 条件 |
|---|---|---|---|---|---|---|
| 时钟高电平时间 | $t_{CHCX}$ | 20 | - | - | nS | |
| 时钟低电平时间 | $t_{CLCX}$ | 20 | - | - | nS | |
| 时钟上升沿时间 | $t_{CLCH}$ | - | - | 10 | nS | |
| 时钟下降沿时间 | $t_{CHCL}$ | - | - | 10 | nS | |

### 7.3.1 外部晶振 XTAL1

| 参数 | 符号 | 最小值 | 典型值 | 最大值 | 单位 |
|---|---|---|---|---|---|
| 输入时钟频率 | 外部晶振 | 4 | 12 | 24 | MHz |
| 温度 | - | -40 | - | 85 | ℃ |
| VDD | - | 2.5 | 5 | 5.5 | V |
| 工作电流 | 12MHz@ $V_{DD}$ = 5V | - | 5 | - | mA |

#### 7.3.1.1 外部晶振电流应用电路

| 晶振 | C1 | C2 | R |
|---|---|---|---|
| 4MHz ~ 24 MHz | 不需要 | 不需要 | 不需要 |

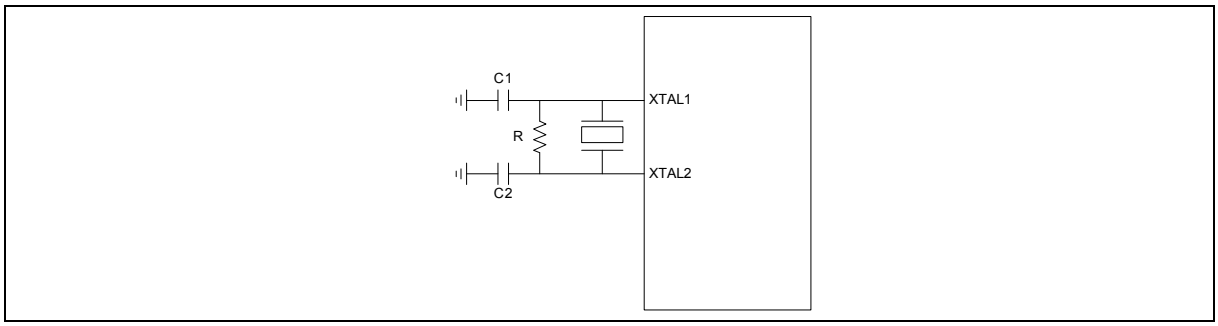



图 7-1 外部晶振电流应用电路

### 7.3.2 外部 32KHz晶振 XTAL

| 参数 | 符号 | 最小值 | 典型值 | 最大值 | 单位 |
|---|---|---|---|---|---|
| 输入时钟频率 | 外部晶振 | - | 32 | - | KHz |
| 温度 | - | -40 | - | 85 | ℃ |
| VDD | - | 2.5 | - | 5.5 | V |
| 工作电流 | $V_{DD}$ = 5V | - | 5 | - | uA |

### 7.3.3 内部 22.1184MHz RC晶振

| 参数 | 符号 | 最小值 | 典型值 | 最大值 | 单位 |
|---|---|---|---|---|---|
| 提供电压[1] | - | 2.5 | - | 5.5 | V |
| 标准频率 | - | - | 22.1184 | | MHz |
| 经过校准标准内部频率偏差率 | +25 C; $V_{DD}$ =5V | -1 | - | +1 | % |
| | -40 C~+85 C; $V_{DD}$=2.5V~5.5V | -3 | - | +3 | % |
| 未经校准标准内部频率偏差率 | -40 C~+85 C; $V_{DD}$=2.5V~5.5V | -25 | - | +25 | % |
| 工作电流 | $V_{DD}$ =5V | - | 500 | - | uA |

### 7.3.4 内部 10KHz RC 晶振

| 参数 | 符号 | 最小值 | 典型值 | 最大值 | 单位 |
|---|---|---|---|---|---|
| 提供电压[1] | - | 2.5 | - | 5.5 | V |
| 标准频率 | - | - | 10 | - | KHz |
| 经过校准标准内部频率偏差率 | +25 C; $V_{DD}$ =5V | -30 | - | +30 | % |
| | -40 C~+85 C; $V_{DD}$=2.5V~5.5V | -50 | - | +50 | % |
| 工作电流 | $V_{DD}$ =5V | - | 5 | - | uA |

**注:**

1. 内部的工作电压来自LDO.

## 7.4 模拟量特性

### 7.4.1 1-MS/s 12-位 SARADC特性

| 参数 | 符号 | 最小值 | 典型值 | 最大值 | 单位 |
|---|---|---|---|---|---|
| 分辨率 | - | - | - | 12 | Bit |
| 非线性差分错误 | DNL | - | ±3 | - | LSB |
| 非线性整型错误 | INL | - | ±4 | - | LSB |
| 补偿错误 | EO | - | ±1 | 10 | LSB |
| 增益错误 (传输增益) | EG | - | 1 | 1.005 | - |
| 一致性 | - | Guaranteed | | | - |
| ADC 时钟频率 | FADC | - | - | 20 | MHz |
| 校准时间 | TCAL | - | 127 | - | Clock |
| 取样时间 | TS | - | 7 | - | Clock |
| 转换时间 | TADC | - | 13 | - | Clock |
| 采样率 | FS | - | - | 800 | Ksps |
| 工作电压 | $V_{LDO}$ | - | 2.5 | - | V |
| | VADD | 3 | - | 5.5 | V |
| 工作电流(平均) | IDD | - | 0.5 | - | mA |
| | IDDA | - | 1.5 | - | mA |
| 参考电压 | VREFP | - | VDDA | - | V |
| 参考电流(平均) | IREFP | - | 1 | - | mA |
| 输入电压范围 | VIN | 0 | - | VREFP | V |
| 电容流量 | CIN | - | 5 | - | pF |

### 7.4.2 LDO规格 & Power 管理

| 参数 | 最小值 | 典型值 | 最大值 | 单位 | 条件 |
|---|---|---|---|---|---|
| 输入电压 | 2.7 | 5 | 5.5 | V | $V_{DD}$ 输入电压 |
| 输出电压 (bypass=0) | -10% | 2.45 | +10% | V | LDO 输出电压 |
| 输出电压 (bypass=1) | -10% | 输入电压 | +10% | V | 输入电压< 2.7V |
| 温度 | -40 | 25 | 85 | oC | |
| 静态电流 (PD=0, bypass=0) | - | 100 | - | uA | |
| 静态电流 (PD=1, bypass=0) | - | 5 | - | uA | |
| 静态电流 (PD=1, bypass=1) | - | 5 | - | uA | |
| Iload (PD=0) | - | - | 100 | mA | |
| Iload (PD=1) | - | - | 100 | uA | |
| Cbp | - | 1u | - | F | Resr=1ohm |
| 负载 | - | 250p | - | F | |

注:

1. 建议接一颗10uF 或更大的电容和一颗 100nF 旁路电容在VDD与VSS之间.
2. 为保证电源稳定，要在LDO与VSS之间接一颗4.7uF 或更大的电容. 再加一颗100nF 的旁路电容在LDO与VSS之间有助于抑制输出噪声.

### 7.4.3 低压复位说明

| 参数 | 条件 | 最小值 | 典型值 | 最大值 | 单位 |
|---|---|---|---|---|---|
| 操作电压 | - | 1.7 | - | 5.5 | V |
| 静态电流 | VDD5V=5.5V | - | - | 5 | uA |
| 温度 | - | -40 | 25 | 85 | ℃ |
| 极限电压 | 温度=25° | 1.7 | 2.0 | 2.3 | V |
| | 温度=-40° | - | 2.4 | - | V |
| | 温度=85° | - | 1.6 | - | V |
| 迟滞 | - | 0 | 0 | 0 | V |

### 7.4.4 欠压检测说明

| 参数 | 条件 | 最小值 | 典型值 | 最大值 | 单位 |
|---|---|---|---|---|---|
| 操作电压 | - | 2.5 | - | 5.5 | V |
| 静态电流 | AVDD=5.5V | - | - | 125 | μA |
| 温度 | - | -40 | 25 | 85 | ℃ |
| 欠压电压 | BOV_VL[1:0]=11 | 4.4 | 4.5 | 4.6 | V |
| | BOV_VL [1:0]=10 | 3.7 | 3.8 | 3.9 | V |
| | BOV_VL [1:0]=01 | 2.6 | 2.7 | 2.8 | V |
| | BOV_VL [1:0]=00 | 2.1 | 2.2 | 2.3 | V |
| 迟滞 | - | 30m | - | 150m | V |

### 7.4.5 上电复位说明(5V)

| 参数 | 条件 | 最小值 | 典型值 | 最大值 | 单位 |
|---|---|---|---|---|---|
| 温度 | - | -40 | 25 | 85 | ℃ |
| 复位电压 | V+ | - | 2 | - | V |
| 静态电流 | Vin>复位电压 | - | 1 | - | nA |

### 7.4.6 温度传感器说明

| 参数 | 最小值 | 典型值 | 最大值 | 单位 | 条件 |
|---|---|---|---|---|---|
| 电源 | 2.5 | - | 5.5 | V | |
| 温度 | -40 | - | 125 | ℃ | |
| 电流消耗 | 6.4 | - | 10.5 | uA | |
| 增益 | -1.95 | -2 | -2.05 | mV/℃ | |
| 偏移量 | 688 | 708 | 730 | mV | Temp=0 ℃ |

**注:**

1. 内部工作电压来自LDO.

### 7.4.7 比较器说明

| 参数 | 最小值 | 典型值 | 最大值 | 条件 |
|---|---|---|---|---|
| 温度 | -40℃ | 25 ℃ | 85℃ | - |
| VDD | 2.4 | 3 | 5.5 | - |
| VDD 电流 | - | 20uA | 40uA | 20uA@VDD=3V |
| 输入偏移电压 | - | 5mV | 15mV | - |
| 输出偏差 | 0.1 | - | VDD-0.1 | - |
| 共模输入范围 | 0.1 | - | VDD-1.2 | - |
| DC 增益 | - | 70dB | - | - |
| 延迟 | - | 200ns | - | @VCM=1.2V & VDIFF=0.1V |
| 比较电压 | 10mV | 20mV | - | 20mV@VCM=1V<br>50mV@VCM=0.1V<br>50mV@VCM=VDD-1.2<br>@10mV for non-hysteresis |
| 迟滞 | - | ±10mV | - | One bit control<br>W/O & W. hysteresis<br>@VCM=0.4V ~ VDD-1.2V |
| 唤醒时间 | - | - | 2us | @CINP=1.3V<br>CINN=1.2V |

### 7.4.8 USB PHY说明

#### *7.4.8.1 USB DC 电气特性*

| 符号 | 参数 | 条件 | 最小值 | 典型值 | 最大值 | 单位 |
|---|---|---|---|---|---|---|
| $V_{IH}$ | 输入高 (driven) | | 2.0 | | | V |
| $V_{IL}$ | 输入低 | | | | 0.8 | V |
| $V_{DI}$ | 差分输入 | \|PADP-PADM\| | 0.2 | | | V |
| $V_{CM}$ | 差分同模范围 | Includes $V_{DI}$ range | 0.8 | | 2.5 | V |
| $V_{SE}$ | 单端接收器极限 | | 0.8 | | 2.0 | V |
| | 接收器滞后 | | | 200 | | mV |
| $V_{OL}$ | 输出低 (driven) | | 0 | | 0.3 | V |
| $V_{OH}$ | 输出高 (driven) | | 2.8 | | 3.6 | V |
| $V_{CRS}$ | 输出信号串扰电压 | | 1.3 | | 2.0 | V |
| $R_{PU}$ | 上拉电阻 | | 1.425 | | 1.575 | kΩ |
| $R_{PD}$ | 下拉电阻 | | 14.25 | | 15.75 | kΩ |
| $V_{TRM}$ | 上行端口上的上拉电阻的极限电压 ($R_{PU}$) | | 3.0 | | 3.6 | V |
| $Z_{DRV}$ | 驱动输出阻抗* | 稳态驱动* | | 10 | | Ω |
| $C_{IN}$ | 发射器容值 | Pin to GND | | | 20 | pF |

*驱动输出阻抗不包括串联电阻阻抗.

#### 7.4.8.2 USB 全速驱动器电气特性

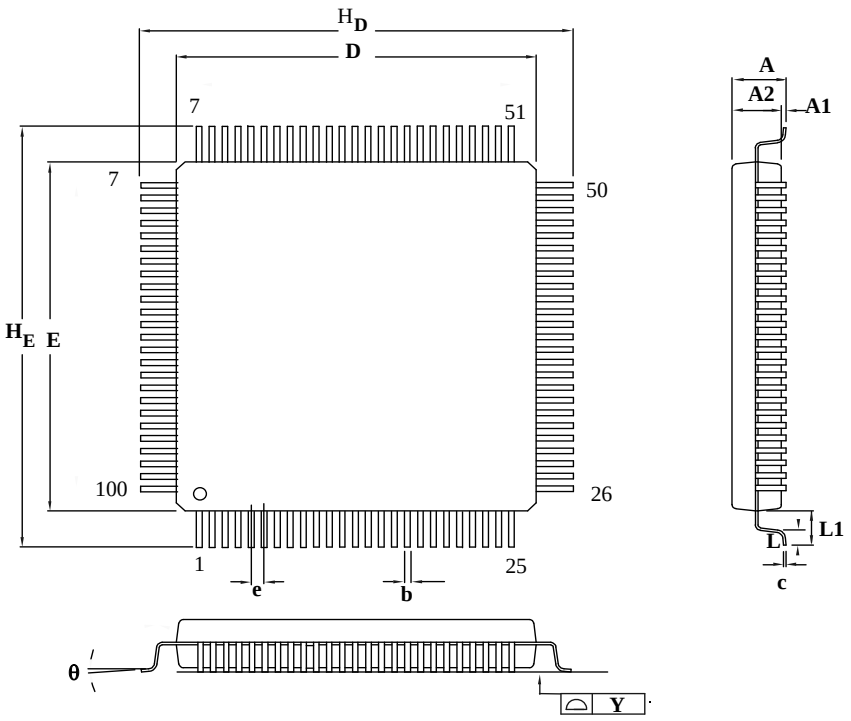
| 符号 | 参数 | 条件 | 最小值 | 典型值 | 最大值 | 单位 |
|---|---|---|---|---|---|---|
| $T_{FR}$ | 上升时间 | $C_L$=50p | 4 | | 20 | ns |
| $T_{FF}$ | 下降时间 | $C_L$=50p | 4 | | 20 | ns |
| $T_{FRFF}$ | 上升与下降时间比值 | $T_{FRFF}=T_{FR}/T_{FF}$ | 90 | | 111.11 | % |

#### 7.4.8.3 USB功耗

| 符号 | 参数 | 条件 | 最小值 | 典型值 | 最大值 | 单位 |
|---|---|---|---|---|---|---|
| $I_{VDDREG}$ (全速) | VDDD 和 VDDREG 供给电流(稳态) | 待机 | | 50 | | uA |
| | | 输入模式 | | | | uA |
| | | 输出模式 | | | | uA |

# 8 封装定义

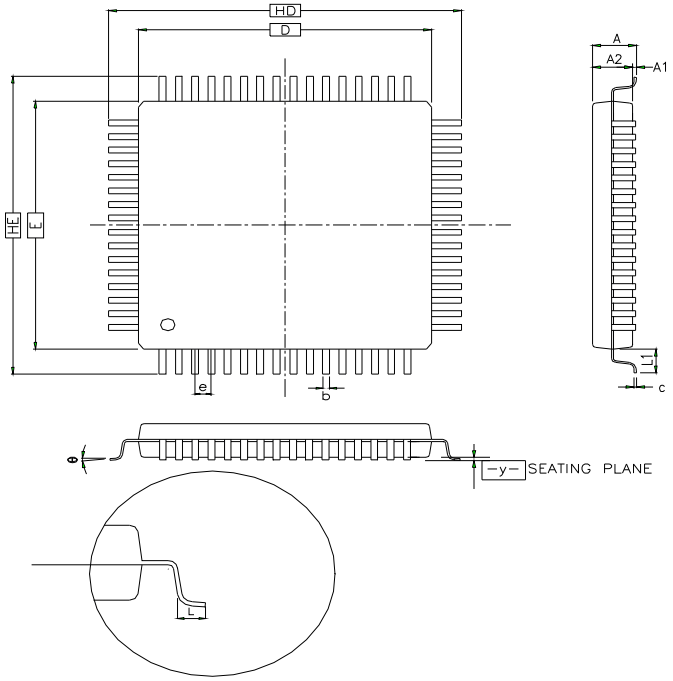
### 8.1.1 100L LQFP (14x14x1.4 mm footprint 2.0mm)




**Controlling Dimension : Millimeters**

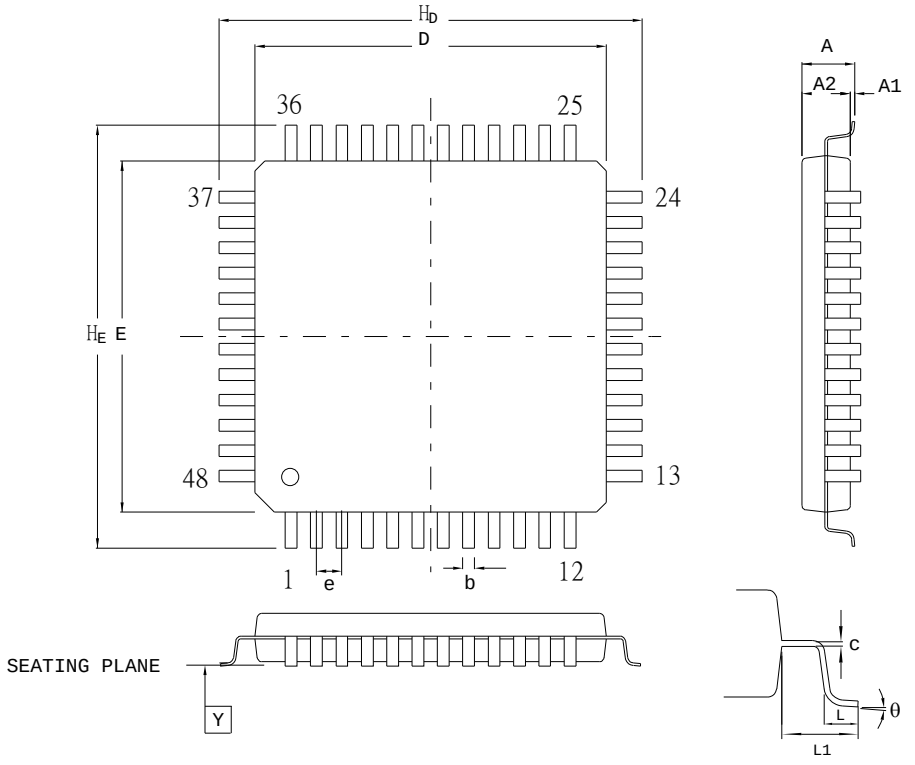
| Symbol | Dimension in inch | | | Dimension in mm | | |
|---|---|---|---|---|---|---|
| | Min | Nom | Max | Min | Nom | Max |
| A | — | — | 0.063 | — | — | 1.60 |
| A1 | 0.002 | — | — | 0.05 | — | — |
| A2 | 0.053 | 0.055 | 0.057 | 1.35 | 1.40 | 1.45 |
| b | 0.007 | 0.009 | 0.011 | 0.17 | 0.22 | 0.27 |
| c | 0.004 | 0.006 | 0.008 | 0.10 | 0.15 | 0.20 |
| D | 0.547 | 0.551 | 0.556 | 13.90 | 14.00 | 14.10 |
| E | 0.547 | 0.551 | 0.556 | 13.90 | 14.00 | 14.10 |
| e | — | 0.020 | — | — | 0.50 | — |
| $H_D$ | 0.622 | 0.630 | 0.638 | 15.80 | 16.00 | 16.20 |
| $H_E$ | 0.622 | 0.630 | 0.638 | 15.80 | 16.00 | 16.20 |
| L | 0.018 | 0.024 | 0.030 | 0.45 | 0.60 | 0.75 |
| L1 | — | 0.039 | — | — | 1.00 | — |
| y | — | — | 0.004 | — | — | 0.10 |
| θ | 0° | — | 7° | 0° | — | 7° |

### 8.1.2 64L LQFP (10x10x1.4mm footprint 2.0 mm )




| Symbol | Dimension in inch | | | Dimension in mm | | |
|---|---|---|---|---|---|---|
| | Min | Nom | Max | Min | Nom | Max |
| A | — | — | 0.063 | — | — | 1.60 |
| $A_1$ | 0.002 | — | 0.006 | 0.05 | — | 0.15 |
| $A_2$ | 0.053 | 0.055 | 0.057 | 1.35 | 1.40 | 1.45 |
| b | 0.007 | 0.008 | 0.011 | 0.17 | 0.20 | 0.27 |
| c | 0.004 | —— | 0.008 | 0.09 | —— | 0.20 |
| D | — | 0.393 | — | — | 10.00 | — |
| E | — | 0.393 | — | — | 10.00 | — |
| e | — | 0.020 | — | — | 0.50 | — |
| $H_D$ | — | 0.472 | — | — | 12.00 | — |
| $H_E$ | — | 0.472 | — | — | 12.00 | — |
| L | 0.018 | 0.024 | 0.030 | 0.45 | 0.60 | 0.75 |
| $L_1$ | — | 0.039 | — | — | 1.00 | — |
| y | — | 0.004 | — | — | 0.10 | — |
| θ | 0° | 3.5° | 7° | 0° | 3.5° | 7° |

### 8.1.3 48L LQFP (7x7x1.4mm footprint 2.0mm)




Controlling dimension : Millimeters

| Symbol | Dimension in inch | | | Dimension in mm | | |
|---|---|---|---|---|---|---|
| | Min | Nom | Max | Min | Nom | Max |
| A | — | — | — | — | — | — |
| $A_1$ | 0.002 | 0.004 | 0.006 | 0.05 | 0.10 | 0.15 |
| $A_2$ | 0.053 | 0.055 | 0.057 | 1.35 | 1.40 | 1.45 |
| b | 0.006 | 0.008 | 0.010 | 0.15 | 0.20 | 0.25 |
| c | 0.004 | 0.006 | 0.008 | 0.10 | 0.15 | 0.20 |
| D | 0.272 | 0.276 | 0.280 | 6.90 | 7.00 | 7.10 |
| E | 0.272 | 0.276 | 0.280 | 6.90 | 7.00 | 7.10 |
| e | 0.014 | 0.020 | 0.026 | 0.35 | 0.50 | 0.65 |
| $H_D$ | 0.350 | 0.354 | 0.358 | 8.90 | 9.00 | 9.10 |
| $H_E$ | 0.350 | 0.354 | 0.358 | 8.90 | 9.00 | 9.10 |
| L | 0.018 | 0.024 | 0.030 | 0.45 | 0.60 | 0.75 |
| $L_1$ | — | 0.039 | — | — | 1.00 | — |
| Y | — | — | 0.004 | — | — | 0.10 |
| θ | 0° | — | 7° | 0° | — | 7° |

# 9 版本信息

| 版本 | 日期 | 页码 | 描述 |
|---|---|---|---|
| V1.00 | 2/ 9, 2010 | - | 初次发布 |

## Important Notice

**Nuvoton products are not designed, intended, authorized or warranted for use as components in systems or equipment intended for surgical implantation, atomic energy control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, combustion control instruments, or for other applications intended to support or sustain life. Further more, Nuvoton products are not intended for applications wherein failure of Nuvoton products could result or lead to a situation wherein personal injury, death or severe property or environmental damage could occur.**

**Nuvoton customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Nuvoton for any damages resulting from such improper use or sales.**

---

**Please 注 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.**