# 龙芯 1A 处理器用户手册

*2012年11月*

*龙芯中科技术有限公司*



北京市海淀区温泉镇中关村环保科技示范园龙芯产业园2号楼 100095
Loongson Industrial Park, building 2，Zhongguancun environmental protection park
Haidian District，Beijing

www.loongson.cn

龙芯中科技术有限公司
Loongson Technology Corporation Limited

## 阅读指南

本文档主要介绍龙芯 1A 的系统架构以及寄存器描述。LS232 处理器核的详细说明参见《LS232 用户手册》。

# 修订历史

| 文档更新记录 | | 文档编号 | |
|---|---|---|---|
| | | 文档名 | 龙芯 1A 处理器用户手册 |
| | | 版本号 | V2.1 |
| | | 创建人 | 研发中心 |
| | | 创建日期 | 2012-10-11 |

**更新历史**

| 序号. | 更新日期 | 更新人 | 版本号 | 更新内容 |
|---|---|---|---|---|
| 1 | 2010-6-7 | 研发中心 | 1.0 | 1A 处理器初稿完成 |
| 2 | 2010-11-13 | 研发中心 | 1.1 | 增加了芯片引脚排布，DDR 控制器信息等 |
| 3 | 2010-11-15 | 研发中心 | 1.2 | 修改并进行标准排版 |
| 4 | 2010-11-15 | 研发中心 | 1.3 | 修订了第五章 DDR 的部分错误 |
| 5 | 2011-05-08 | 研发中心 | 1.4 | 修订了调试发现的错误 |
| 6 | 2011-05-17 | 研发中心 | 1.5 | 修订了多个小问题 |
| 7 | 2011-09-22 | 研发中心 | 1.6 | 修改了系统启动时钟配置<br>增加了 SRAM 控制器添加和复用<br>增加了 NAND ECC、BOOT 和中断<br>增加了 SDRAM 工作 16 位数据宽度配置<br>添加了 GMAC0/1 工作 MAC 模式复用功能<br>增加了复用使能寄存器添加控制位<br>增加了封装实现中各数据 PAD 的 delay |
| 8 | 2011-4-11 | 研发中心 | 1.7 | 添加 LPC 章节<br>ACPI 配置寄存器<br>NAND 部分寄存器说明修改<br>XTALI/O 与外部有源晶振、无源晶体连接方法 |
| 9 | 2011-4-20 | 研发中心 | V1.8 | PAD 封装位置和封装延迟<br>GMAC0/1 在 MII 模式下信号处理<br>Wdog 地址修改 |

| | | | | |
|---|---|---|---|---|
| | | | | USB 启动复位 |
| 9 | 2011-7-13 | 研发中心 | V1.9 | 增加 VGA、VR、USB 引脚连接说明<br>修正 LCD 颜色分量的错误<br>增加 SATA 时钟配置 |
| 10 | 2012-8-19 | 研发中心 | V2.0 | 增加 DDR 配置寄存器描述<br>修正 GPIO 复用章节<br>修改启动和时钟配置提到第 3 章<br>版面调整 |
| 11 | 2012-10-11 | 研发中心 | V2.1 | 修正 PLL 引脚定义<br>时钟架构图明确 LPC 模块时钟<br>去除 DDR 控制器 ECC 功能的描述<br>修正 NAND 颗粒容量大小的描述<br>增加 PWM 复用为 MAC 的配置 |

手册信息反馈: service@loongson.cn

# 目　录

# 图 目 录

# 表 目 录

# 1 概述

龙芯 1A 是基于 LS232 处理器核的高性价比单芯片系统，可广泛应用于工业控制、家庭网关、信息家电、安全应用等领域。

龙芯 1A 具有以下关键特性：

- 集成一个 LS232 双发射龙芯处理器核，指令和数据 L1 Cache 各 16KB
- 集成 2D GPU
- 集成两路 DC 控制器，最大分辨率可支持到 1920*1080@60Hz/24bit
- 集成 2 个 10M/100M/1000M 自适应 GMAC
- 集成 2 个 SATA2
- 集成 32 位 PCI，支持主从模式
- 集成 1 个 32 位/16 位 DDR2 控制器
- 集成 4 个 USB HOST 接口，兼容 USB2.0 和 USB1.1
- 集成 1 个 8 位 NAND FLASH 控制器，支持 4 个片选
- 集成中断控制器，支持灵活的中断设置
- 集成 2 个 SPI 控制器，支持主模式，SPI0 支持系统启动
- 集成 AC97 控制器
- 集成 1 个 LPC 控制器
- 集成 4 路 UART 串口
- 集成 1 路 PS/2(键盘和鼠标)
- 集成 3 路 I2C 控制器，兼容 SMBUS
- 集成 2 路 CAN 总线控制器
- 集成 88 路 GPIO 端口
- 集成 1 路 RTC 接口
- 集成 4 路 PWM 控制器
- 集成 ACPI
- 集成看门狗

## 1.1 体系结构框图

龙芯 1A 内部顶层结构由 AXI 交叉开关互连，其中 LS232、GPU/DC、PCI

和 AXI_MUX 作为主设备通过 4X4 交叉开关连接到系统；GPU/DC、AXI_MUX、PCI 和 DDR2 作为从设备通过 4X4 交叉开关连接到系统。在 AXI_MUX 内部实现了多个 AHB 和 APB 模块到顶层 AXI 交叉开关的连接，其中 DMA_MUX、GMAC0、GMAC1、USB 和 SATA 被 AXI_MUX 选择作为主设备访问交叉开关；AXI_MUX（包括 confreg、SPI0、SPI1）、 AXI2APB、SATA、GMAC0、GMAC1、USB 等作为从设备被来自 AXI_MUX 的主设备访问。在 AXI2APB 内部实现了系统对内部 APB 接口设备的访问，这些设备包括 Watch Dog、RTC、PWM、I2C、PS2、CAN、NAND、UART 等。

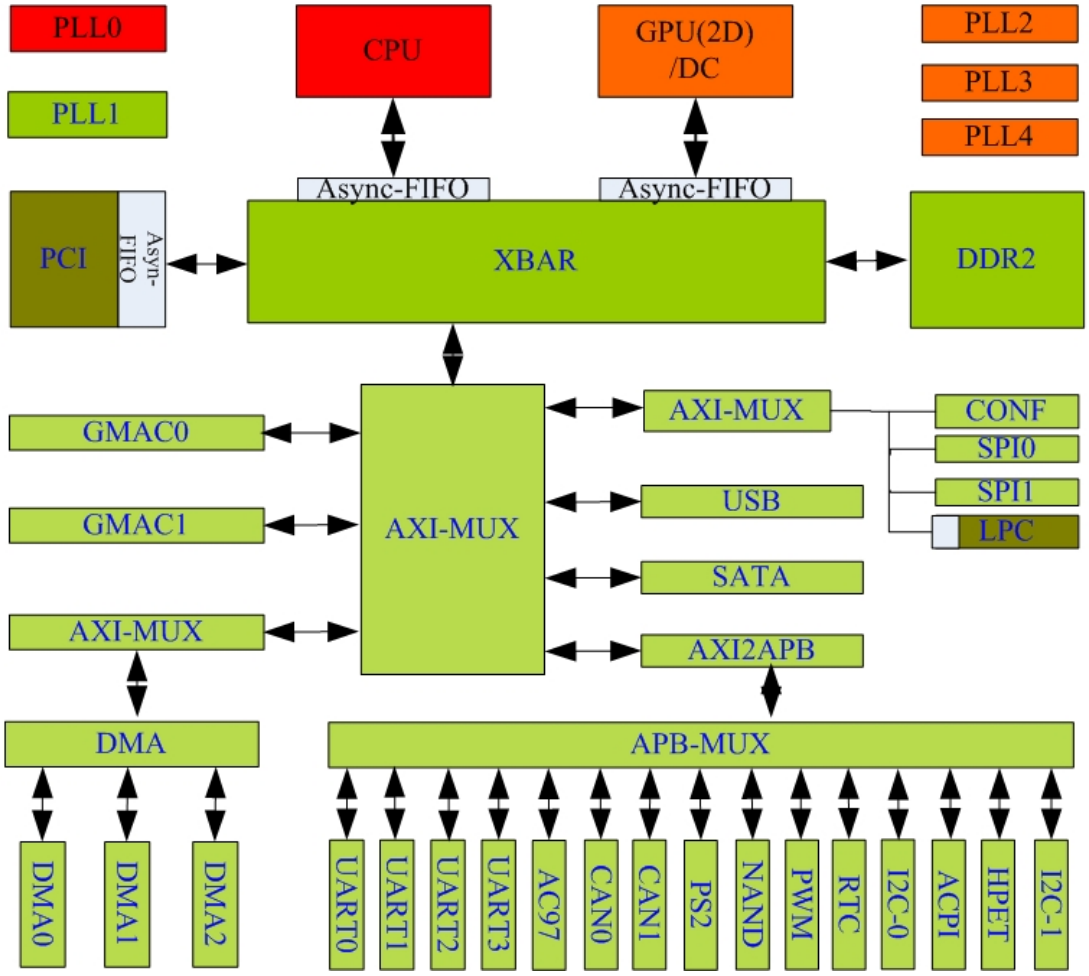



图 1-1 龙芯 1A 结构图

## 1.2 芯片主要功能

### 1.2.1 LS232 CPU

龙芯 232 核是一款实现 MIPS32 兼容且支持 EJTAG 调试的双发射处理器，通过采用转移预测、寄存器重命名、乱序发射、路预测的指令 CACHE、非阻塞的数据 CACHE、写合并收集等技术来提高流水线的效率。

- 双发射五级流水、乱序发射、乱序执行

- 16KB 指令 Cache+16KB 数据 Cache，4 路组相连，指令 CACHE 支持路预测
- 6 项 BRQ、16 项的 QUEUE
- 动态转移预测、地址返回栈
- 32 项 JTLB，4 项 ITLB 、8 项 DTLB
- 两个定点 ALU 部件。
- 浮点部件支持全流水的 64 位浮点加法和浮点乘法运算，硬件实现浮点除法运算
- 专门的 SIMD 型多媒体加速指令
- 支持非阻塞的 Cache 访问技术，4 项 load 队列、2 项 store 队列、3 项 miss 队列，最多容忍 5 条 store 指令 Cache 不命中和 4 条 load 指令 Cache 不命中
- 支持 cached store 指令的写合并
- 支持预取指令
- 支持流水线暂停模式
- 支持向量中断，可配置支持快速中断响应，最多 8 个时钟周期进入中断处理程序
- 支持 EJTAG 调试

### 1.2.2 DDR2

- 32 位/16 位 DDR2 控制器
- 遵守 DDR2 DDR 的行业标准（JESD79-2B）
- 支持最大 2 个物理内存 bank（由 2 个 DDR2 DDR 片选信号实现），一共含有 18 位的地址总线（即：15 位的行列地址总线和 3 位的逻辑 Bank 总线）
- 接口上命令、读写数据全流水操作
- 内存命令合并、排序提高整体带宽
- 配置寄存器读写端口，可以修改内存设备的基本参数
- 内建动态延迟补偿电路（DCC），用于数据的可靠发送和接收
- 支持 33-166MHZ 工作频率

### 1.2.3 PCI

- 兼容 PCI 2.2，32 位总线宽度，支持 33MHz 总线频率
- 既可以做 Host（SoC），又可以做 Device（南桥），上电时由外面 PAD 配置
- 作为 Host 最多支持 2 个 PCI 设备
- 作为 Device 时有三个 PCI 地址窗口：IO、Memory、Prefetchable Memory。其中 Prefetchable Memory 窗口用来访问片上 DDR2

### 1.2.4 2D GPU

- 通过 Futuremark 认证
- 动态电源管理
- 支持 BitBLT 和 Stretch BLT
- 矩形填充
- 硬件画线
- 单色字体渲染
- ROP2，ROP3，ROP4
- Alpha 混合
- 32Kx32K 坐标系统
- 90 度旋转
- 透明支持
- YUV 色域空间转换

### 1.2.5 LCD Controller

- 屏幕大小可达 1920*1080
- 硬件光标
- 伽玛校正
- 最高像素时钟 172MHz
- 支持线性显示缓冲
- 上电序列控制
- 具有 VGA 和 LCD 双接口；16 位,24 位支持

### 1.2.6 SATA

- 2 个独立 SATA2 的接口
- 支持 SATA 1 代 1.5Gbps 和 SATA2 代 3Gbps 的传输
- 兼容串行 ATA 2.6 规范和 AHCI 1.1 规范

### 1.2.7 USB2.0

- 4 个独立的 USB2.0 端口
- 兼容 USB1.1 和 USB2.0
- 内部 EHCI 控制和实现高速传输可达 480Mbps
- 内部 OHCI 控制和实现全速和低速传输 12Mbps 和 1.5Mbps

### 1.2.8 AC97

- 支持 16，18 和 20 位采样精度，支持可变速率
- 最高达 48KHz
- 2 频道立体声输出
- 支持麦克风输入

### 1.2.9 GMAC

- 两路 10/100/1000Mbps 自适应以太网控制器
- 双网卡均兼容 IEEE 802.3
- 对外部 PHY 实现 RGMII 和 MII 接口
- 半双工/全双工自适应
- 半双工时，支持碰撞检测与重发（CSMA/CD）协议
- 支持 CRC 校验码的自动生成与校验

### 1.2.10 LPC

- 兼容 LPC Rev1.1 标准
- 内置 FIFO

### 1.2.11 SPI

- 支持 2 路 SPI 接口
- SPI0 支持系统启动
- 极性和相位可编程的串行时钟
- 可在等待模式下对 SPI 进行控制

### 1.2.12 UART

- 集成 1 个全功能串口、1 个四线串口、2 个两线串口
- 在寄存器与功能上兼容 NS16550A
- 全双工异步数据接收/发送
- 可编程的数据格式
- 16 位可编程时钟计数器
- 支持接收超时检测
- 带仲裁的多中断系统

### 1.2.13 $I^2C$

- 兼容 SMBUS（100Kbps）
- 与 PHILIPS I2C 标准相兼容

- 只实现主设备操作
- 能够支持多主设备的总线
- 总线的时钟频率可编程
- 可以产生开始/停止/应答等操作
- 能够对总线的状态进行探测
- 支持低速和快速模式
- 支持 7 位寻址和 10 位寻址
- 支持时钟延伸和等待状态
-

### 1.2.14 PWM

- 提供 4 路可配置 PWM 输出
- 数据宽度 32 位
- 定时器功能
- 计数器功能

### 1.2.15 NAND

- 共 4 个片选 CS
- 支持 SLC/MLC
- 数据宽度 8bit
- 支持系统启动
- 支持 2KB 页

### 1.2.16 CAN

- 支持 2 个独立 CAN 总线接口
- 每路 CAN 接口均支持 CAN2.0A/B 协议
- 支持 CAN 协议扩展

### 1.2.17 RTC

- 计时精确到 0.1 秒
- 可产生 3 个计时中断
- 支持定时开关机功能

### 1.2.18 GPIO

- 88 路 GPIO

### 1.2.19 INT controller

- 支持软件设置中断

- 支持电平与边沿触发（上升或下降沿触发）
- 支持中断屏蔽与使能

#### 1.2.20 PS2

- 16 位可编程 5us 时钟计数器，8 位可编程 60us 时钟计数器
- 兼容第一套和第二套键盘扫描码
- 支持编码键盘和非编码键盘
- 支持二键式、三键式鼠标

#### 1.2.21 Watchdog

- 16 位计数器
- 低功耗模式暂停功能

#### 1.2.22 ACPI

- 兼容 ACPI 功耗管理规范
- 各个模块运行频率可调
- 多电压域设计（CORE、RTC、Resume）
- 全芯片 Clock gating
- 支持 STR,STD 等睡眠模式

#### 1.2.23 功耗

- 0.6-0.9W

## 1.3 文档约定与记号

### 1.3.1 数值表示

16 进制数表示为'h***或者 0x***，2 进制数表示为'b***，其它数字为 10 进制。

功能相同但标号有别的引脚（如 DDR_DQ0, DDR_DQ1, …）使用方括号加数字范围的形式简写（如 DDR_DQ[63:0]）。类似地，寄存器域也采用这种表示方式。

### 1.3.2 寄存器域

寄存器域以[寄存器名].[域名]的形式加以引用。如 chip_config0.uart_split 指芯片配置寄存器 0（chip_config0）的 uart_split 域。

### 1.3.3 地址说明

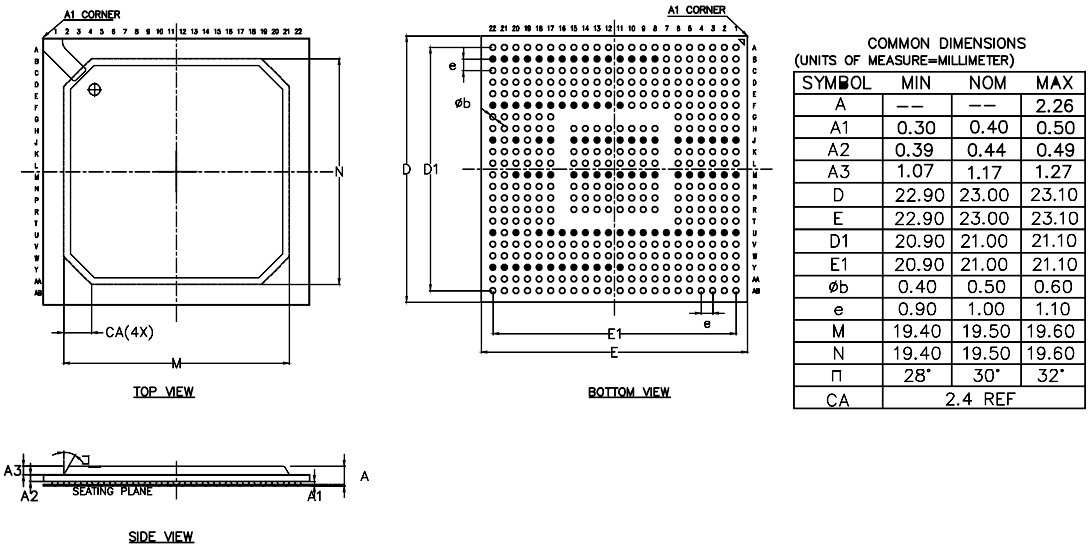
文档中出现的所有地址均为物理地址，软件访问须使用其所对应的物理地

址。例如 USB 寄存器基址在 0x1fe00000，软件可使用 0x1fe00000 进行访问。

# 2 芯片引脚定义

## 2.1 1A 引脚分布图

1A 采用 BGA448 封装形式，封装尺寸如下图所示：




COMMON DIMENSIONS
(UNITS OF MEASURE=MILLIMETER)

| SYMBOL | MIN | NOM | MAX |
|---|---|---|---|
| A | -- | -- | 2.26 |
| A1 | 0.30 | 0.40 | 0.50 |
| A2 | 0.39 | 0.44 | 0.49 |
| A3 | 1.07 | 1.17 | 1.27 |
| D | 22.90 | 23.00 | 23.10 |
| E | 22.90 | 23.00 | 23.10 |
| D1 | 20.90 | 21.00 | 21.10 |
| E1 | 20.90 | 21.00 | 21.10 |
| øb | 0.40 | 0.50 | 0.60 |
| e | 0.90 | 1.00 | 1.10 |
| M | 19.40 | 19.50 | 19.60 |
| N | 19.40 | 19.50 | 19.60 |
| Π | 28° | 30° | 32° |
| CA | 2.4 REF | | |

芯片的引脚 PAD 封装延迟如下表所示:

表 2-1 芯片引脚PAD的封装延迟列表

| Location | Pad Name | Distance |
|---|---|---|
| V20 | AC97_BIT_CLK | 11413.17 |
| W20 | AC97_DATA_I | 10434.28 |
| U20 | AC97_DATA_O | 11826.62 |
| AA20 | AC97_RESET | 15200.64 |
| Y20 | AC97_SYNC | 10361.56 |
| B21 | ACPI_SOC_EN | 12129.94 |
| A17 | ACPI_SYS_RSTN | 9671.55 |
| AB20 | CAN0_RX | 12813.47 |
| AB21 | CAN0_TX | 13394.77 |
| Y21 | CAN1_RX | 11902.90 |
| AA21 | CAN1_TX | 11772.66 |
| F10 | DDR_VREF_0V9 | |
| F13 | DDR_VREF_0V9 | |
| H06 | DDR_VREF_0V9 | |
| A02 | DDR2_A00 | 13699.23 |
| B02 | DDR2_A01 | 13966.17 |
| C02 | DDR2_A02 | 11445.94 |
| C01 | DDR2_A03 | 13391.30 |

| | | |
|---|---|---|
| D01 | DDR2_A04 | 11729.31 |
| A01 | DDR2_A05 | 15157.40 |
| B01 | DDR2_A06 | 14628.84 |
| A03 | DDR2_A07 | 15205.05 |
| B03 | DDR2_A08 | 12382.38 |
| A04 | DDR2_A09 | 11442.98 |
| B04 | DDR2_A10 | 11702.33 |
| C04 | DDR2_A11 | 9664.96 |
| C05 | DDR2_A12 | 10312.54 |
| D05 | DDR2_A13 | 6812.54 |
| D02 | DDR2_A14 | 13097.37 |
| A08 | DDR2_BA0 | 9157.88 |
| B08 | DDR2_BA1 | 8695.37 |
| C08 | DDR2_BA2 | 7626.47 |
| E08 | DDR2_CASN | 6828.24 |
| D07 | DDR2_CKE0 | 6975.04 |
| C03 | DDR2_CKE1 | 12536.62 |
| B05 | DDR2_CKN0 | 10785.75 |
| B06 | DDR2_CKN1 | 10697.88 |
| A05 | DDR2_CKP0 | 10864.72 |
| A06 | DDR2_CKP1 | 10782.53 |
| H03 | DDR2_DQ00 | 9547.11 |
| F03 | DDR2_DQ01 | 11830.75 |
| H02 | DDR2_DQ02 | 9330.07 |
| F01 | DDR2_DQ03 | 13377.23 |
| F02 | DDR2_DQ04 | 11919.73 |
| H01 | DDR2_DQ05 | 10405.97 |
| E02 | DDR2_DQ06 | 12762.04 |
| G01 | DDR2_DQ07 | 11692.57 |
| G03 | DDR2_DQ08 | 9527.24 |
| E05 | DDR2_DQ09 | 10155.00 |
| H05 | DDR2_DQ10 | 10571.16 |
| E03 | DDR2_DQ11 | 10575.38 |
| E04 | DDR2_DQ12 | 10254.01 |
| H04 | DDR2_DQ13 | 6856.05 |
| F05 | DDR2_DQ14 | 9238.00 |
| G04 | DDR2_DQ15 | 7030.55 |
| A10 | DDR2_DQ16 | 9351.10 |
| A12 | DDR2_DQ17 | 10519.92 |
| B10 | DDR2_DQ18 | 8413.83 |
| C12 | DDR2_DQ19 | 11037.38 |
| B12 | DDR2_DQ20 | 11543.40 |

| | | |
|---|---|---|
| C10 | DDR2_DQ21 | 7596.07 |
| A13 | DDR2_DQ22 | 10677.05 |
| B11 | DDR2_DQ23 | 8422.82 |
| E10 | DDR2_DQ24 | 6706.54 |
| D12 | DDR2_DQ25 | 6113.94 |
| D09 | DDR2_DQ26 | 9217.14 |
| E13 | DDR2_DQ27 | 7926.60 |
| D13 | DDR2_DQ28 | 7132.92 |
| E09 | DDR2_DQ29 | 8799.86 |
| E12 | DDR2_DQ30 | 5424.00 |
| D10 | DDR2_DQ31 | 7978.77 |
| E01 | DDR2_DQM0 | 11822.62 |
| F04 | DDR2_DQM1 | 8076.59 |
| B13 | DDR2_DQM2 | 8915.37 |
| D11 | DDR2_DQM3 | 6978.98 |
| G02 | DDR2_DQS0 | 11837.76 |
| G05 | DDR2_DQS1 | 7685.70 |
| A11 | DDR2_DQS2 | 10181.97 |
| E11 | DDR2_DQS3 | 7035.64 |
| D06 | DDR2_GATEI0 | 7117.97 |
| E06 | DDR2_GATEI1 | 7515.12 |
| C06 | DDR2_GATEO0 | 8058.26 |
| A07 | DDR2_GATEO1 | 10274.62 |
| B07 | DDR2_ODT0 | 8245.28 |
| C07 | DDR2_ODT1 | 7894.93 |
| D08 | DDR2_RASN | 7176.36 |
| B09 | DDR2_SCSN0 | 8497.46 |
| C09 | DDR2_SCSN1 | 7592.34 |
| A09 | DDR2_WEN | 9635.49 |
| R20 | EJTAG_TCK | 7199.93 |
| R19 | EJTAG_TDI | 5978.36 |
| R18 | EJTAG_TDO | 8092.47 |
| R21 | EJTAG_TMS | 7598.90 |
| R17 | EJTAG_TRST | 6530.17 |
| D16 | GMAC0_MDCK | 8036.07 |
| E16 | GMAC0_MDIO | 8076.27 |
| C15 | GMAC0_RX_CLK_I | 9882.61 |
| A14 | GMAC0_RX_CTL | 10427.23 |
| B14 | GMAC0_RX0 | 10924.88 |
| C14 | GMAC0_RX1 | 10452.44 |
| A15 | GMAC0_RX2 | 10879.00 |
| B15 | GMAC0_RX3 | 10308.49 |

| | | |
|---|---|---|
| F16 | GMAC0_TX_CLK_I | 11748.79 |
| D14 | GMAC0_TX_CLK_O | 11840.46 |
| F15 | GMAC0_TX_CTL | 11801.97 |
| E14 | GMAC0_TX0 | 11688.49 |
| F14 | GMAC0_TX1 | 11822.30 |
| D15 | GMAC0_TX2 | 11896.39 |
| E15 | GMAC0_TX3 | 11750.70 |
| Y13 | GMAC1_RX_CLK_I | 7824.49 |
| V13 | GMAC1_TX_CLK_I | 7804.13 |
| W13 | GMAC1_TX_CLK_O | 7768.01 |
| U19 | I2C_SCL | 9139.17 |
| V19 | I2C_SDA | 11282.29 |
| T21 | INTN0 | 8033.52 |
| R22 | INTN1 | 9257.69 |
| Y19 | KB_CLK | 10753.23 |
| W19 | KB_DAT | 9307.74 |
| AB12 | LCD_CLK | 9480.79 |
| Y08 | LCD_DAT_B0 | 7958.79 |
| AA08 | LCD_DAT_B1 | 9040.66 |
| AB08 | LCD_DAT_B2 | 9972.82 |
| U09 | LCD_DAT_B3 | 5328.00 |
| V09 | LCD_DAT_B4 | 7931.70 |
| W09 | LCD_DAT_B5 | 6824.36 |
| Y09 | LCD_DAT_B6 | 7656.92 |
| AA09 | LCD_DAT_B7 | 9396.81 |
| AB09 | LCD_DAT_R0 | 9950.36 |
| U10 | LCD_DAT_R1 | 6615.02 |
| V10 | LCD_DAT_R2 | 7304.87 |
| W10 | LCD_DAT_R3 | 8143.30 |
| Y10 | LCD_DAT_R4 | 7669.82 |
| AA10 | LCD_DAT_R5 | 8114.94 |
| AB10 | LCD_DAT_R6 | 9678.79 |
| U11 | LCD_DAT_R7 | 5287.98 |
| V11 | LCD_DAT_G0 | 12134.01 |
| W11 | LCD_DAT_G1 | 6166.92 |
| Y11 | LCD_DAT_G2 | 7653.00 |
| AA11 | LCD_DAT_G3 | 9149.44 |
| AB11 | LCD_DAT_G4 | 12217.86 |
| U12 | LCD_DAT_G5 | 9731.12 |
| V12 | LCD_DAT_G6 | 8813.05 |
| W12 | LCD_DAT_G7 | 7468.76 |
| U13 | LCD_EN | 6229.91 |

| | | |
|---|---|---|
| Y12 | LCD_HSYNC | 7398.96 |
| AA12 | LCD_VSYNC | 8522.69 |
| V22 | LPC_AD0 | 12820.22 |
| W22 | LPC_AD1 | 11257.76 |
| Y22 | LPC_AD2 | 12282.31 |
| AA22 | LPC_AD3 | 12159.16 |
| AB22 | LPC_FRAMEN | 14520.28 |
| W21 | LPC_SERIRQN | 9839.85 |
| AB19 | MS_CLK | 12831.70 |
| AA19 | MS_DAT | 12220.85 |
| AA13 | NAND_ALE | 8558.11 |
| AA14 | NAND_CE | 9029.91 |
| AB14 | NAND_CLE | 10255.44 |
| AA15 | NAND_D6 | 9024.57 |
| Y15 | NAND_D7 | 9005.68 |
| Y14 | NAND_RD | 7226.80 |
| AB15 | NAND_RDY | 10582.56 |
| AB13 | NAND_WR | 9659.67 |
| L04 | PCI_AD00 | 6256.20 |
| L05 | PCI_AD01 | 5189.41 |
| L01 | PCI_AD02 | 9291.07 |
| L02 | PCI_AD03 | 7726.21 |
| L03 | PCI_AD04 | 7143.69 |
| M04 | PCI_AD05 | 6092.11 |
| M05 | PCI_AD06 | 7649.18 |
| M01 | PCI_AD07 | 9854.33 |
| M03 | PCI_AD08 | 7111.98 |
| N04 | PCI_AD09 | 7056.12 |
| N05 | PCI_AD10 | 8763.36 |
| N06 | PCI_AD11 | 6210.61 |
| N01 | PCI_AD12 | 9117.11 |
| N02 | PCI_AD13 | 10148.51 |
| N03 | PCI_AD14 | 7071.31 |
| P04 | PCI_AD15 | 6070.87 |
| T05 | PCI_AD16 | 11493.92 |
| T06 | PCI_AD17 | 8753.70 |
| T01 | PCI_AD18 | 10440.20 |
| T02 | PCI_AD19 | 9178.78 |
| T03 | PCI_AD20 | 11299.34 |
| U04 | PCI_AD21 | 9280.32 |
| U05 | PCI_AD22 | 10820.03 |
| U01 | PCI_AD23 | 11979.22 |

| | | |
|---|---|---|
| U03 | PCI_AD24 | 11278.97 |
| V04 | PCI_AD25 | 9693.15 |
| V01 | PCI_AD26 | 11869.26 |
| V02 | PCI_AD27 | 11088.14 |
| V03 | PCI_AD28 | 11623.80 |
| W04 | PCI_AD29 | 10432.93 |
| W01 | PCI_AD30 | 12367.12 |
| W02 | PCI_AD31 | 10990.53 |
| M02 | PCI_CBEN0 | 9205.02 |
| P05 | PCI_CBEN1 | 5203.12 |
| T04 | PCI_CBEN2 | 9204.53 |
| U02 | PCI_CBEN3 | 9120.64 |
| AB02 | PCI_CLK | 14033.95 |
| R05 | PCI_DEVSELN | 9622.31 |
| R03 | PCI_FRAMEN | 8650.26 |
| W03 | PCI_GNTN0 | 11318.56 |
| AA03 | PCI_GNTN1 | 11613.41 |
| AB03 | PCI_IDSEL | 10825.16 |
| R02 | PCI_IRDYN | 9219.17 |
| Y03 | NC | 11483.75 |
| Y02 | NC | 15045.31 |
| AA02 | NC | 13378.91 |
| AA01 | NC | 13542.57 |
| P01 | PCI_PAR | 12718.01 |
| P03 | PCI_PERR | 11692.79 |
| Y04 | PCI_REQN0 | 9604.46 |
| AB01 | PCI_REQN1 | 14657.32 |
| Y01 | PCI_RESETN | 14306.63 |
| P02 | PCI_SERR | 14695.51 |
| R04 | PCI_STOPN | 14443.57 |
| R01 | PCI_TRDYN | 14747.22 |
| U21 | PWM0 | 13825.22 |
| V21 | PWM1 | 12236.33 |
| U22 | PWM2 | 14311.22 |
| T22 | PWM3 | 17682.90 |
| A16 | RSM_ACPI_EN | 9405.21 |
| F18 | RSM_BATLOWN | 7095.92 |
| E20 | RSM_LID | 9078.15 |
| E19 | RSM_PMEN | 9440.47 |
| D20 | RSM_PWRBTNN | 9737.94 |
| D22 | RSM_PLTRSTN | 11558.37 |
| D21 | RSM_RIN | 8207.42 |

| | | |
|---|---|---|
| C21 | RSM_S3N | 11826.05 |
| C22 | RSM_S4N | 11744.90 |
| E18 | RSM_SUS_STATN | 8998.88 |
| G18 | RSM_VDD1V2 | 8147.44 |
| F17 | RSM_VDD3V3 | 9779.25 |
| A22 | RTC_CLK_I | 13624.78 |
| B22 | RTC_CLK_O | 12924.10 |
| B20 | RTC_PWROK | 10331.35 |
| B19 | RTC_RSMRSTN | 10265.63 |
| B18 | RTC_RTCRSTN | 9371.05 |
| C18 | RTC_VDD3V3 | 22533.39 |
| C19 | RTC_VR_CEXT | 12687.61 |
| B17 | RTC_VR_PD | 10320.52 |
| D19 | RTC_VR_VBG | 8929.75 |
| C20 | RTC_VR_VOUT | 15319.30 |
| G22 | SATA_REFCLKN | 11096.36 |
| H22 | SATA_REFCLKP | 11115.50 |
| G19 | SATA_RESREF | 8830.26 |
| F21 | SATA0_RXN | 11073.21 |
| F22 | SATA0_RXP | 11121.74 |
| E22 | SATA0_TXN | 11687.82 |
| E21 | SATA0_TXP | 11598.34 |
| H20 | SATA1_RXN | 10441.06 |
| H21 | SATA1_RXP | 10536.01 |
| G21 | SATA1_TXN | 11598.11 |
| G20 | SATA1_TXP | 11505.47 |
| V15 | SPI0_CLK | 6709.34 |
| W15 | SPI0_CS | 7419.34 |
| V16 | SPI0_MISO | 7646.94 |
| U16 | SPI0_MOSI | 12137.03 |
| U15 | SPI1_CLK | 7604.55 |
| U14 | SPI1_CS | 10654.14 |
| V14 | SPI1_MISO | 6135.37 |
| W14 | SPI1_MOSI | 7665.50 |
| T17 | TEST_BIST_CLK | 6695.26 |
| T18 | TEST_CFG_MODEN | 9081.97 |
| P22 | TEST_JTAG_SEL | 9386.96 |
| T19 | TEST_PHY_CLK | 8207.63 |
| T20 | TEST_SCAN_MODEN | 10862.28 |
| AA18 | UART0_CTS | 10693.38 |
| AB18 | UART0_DCD | 10109.54 |
| Y17 | UART0_DSR | 8607.22 |

| | | |
|---|---|---|
| AB17 | UART0_DTR | 13294.27 |
| Y18 | UART0_RI | 9953.20 |
| W18 | UART0_RTS | 8862.13 |
| W17 | UART0_RX | 8387.47 |
| AA17 | UART0_TX | 9243.86 |
| AB16 | UART1_CTS | 10091.94 |
| AA16 | UART1_RTS | 8566.29 |
| W16 | UART1_RX | 7706.23 |
| Y16 | UART1_TX | 8278.90 |
| U18 | UART2_RX | 10857.70 |
| V18 | UART2_TX | 10993.82 |
| U17 | UART3_RX | 8823.21 |
| V17 | UART3_TX | 9055.94 |
| L19 | USB_AVDD33 | 21221.31 |
| J20 | USB0_DM | 7534.44 |
| J19 | USB0_DP | 7444.87 |
| P21 | USB0_OVRCUR | 8441.81 |
| J18 | USB0_REXT | 7652.20 |
| J22 | USB0_XI | 9731.39 |
| J21 | USB0_XO | 8439.53 |
| K20 | USB1_DM | 7079.68 |
| K19 | USB1_DP | 7022.81 |
| P20 | USB1_OVRCUR | 6742.87 |
| K18 | USB1_REXT | 7966.98 |
| K22 | USB1_XI | 9724.20 |
| K21 | USB1_XO | 8687.63 |
| M20 | USB2_DM | 6987.05 |
| M19 | USB2_DP | 6890.71 |
| P19 | USB2_OVRCUR | 5766.56 |
| M18 | USB2_REXT | 6918.23 |
| M22 | USB2_XI | 9503.46 |
| M21 | USB2_XO | 8448.87 |
| N20 | USB3_DM | 6946.45 |
| N19 | USB3_DP | 6962.71 |
| P18 | USB3_OVRCUR | 6528.85 |
| N18 | USB3_REXT | 6819.77 |
| N22 | USB3_XI | 9542.01 |
| N21 | USB3_XO | 8530.45 |
| Y07 | VGA_B | 7479.53 |
| AB06 | VGA_COMP | 9103.97 |
| W08 | VGA_EN | 6716.21 |
| AA07 | VGA_G | 10701.20 |

| | | |
|---|---|---|
| V08 | VGA_HSYNC | 10175.52 |
| AB07 | VGA_R | 7608.28 |
| AA06 | VGA_REXT | 8539.80 |
| W06 | VGA_VREFIN | 8049.06 |
| Y06 | VGA_VREFOUT | 8307.96 |
| W07 | VGA_VSYNC | 9077.90 |

## 2.2 系统相关引脚定义

表 2-2 系统时钟引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| XTALI | I | | 外部无源晶体时钟输入；<br>外部有源晶振悬空连接 |
| XTALO | O | | 外部无源晶体时钟回送；外部有源晶振输入 |
| RTC_CLK_I | I | | RTC 时钟晶体输入 |
| RTC_CLK_O | O | | RTC 时钟晶体回送 |
| PCI_CLK | I | | PCI 时钟，不论是否使用 PCI 都必须连接时钟。频率不高于 33MHz。 |
| INTn[1:0] | O | | 中断输出，内部中断的或。桥片模式下接到 2F 的中断输入引脚，SoC 模式下配置为 GPIO 则可作为 PCI 中断输入<br>INTn0: INT1/2<br>INTn1: INT3/4/5 |
| TEST_CFG_MODEN | I | | 测试信号，需外部上拉 |
| TEST_BIST_CLK | I | | 测试信号，可外部上拉或下拉 |
| TEST_PHY_CLK | I | | 测试信号，可外部上拉或下拉 |
| TEST_SCAN_MODEN | I | | 测试信号，需外部上拉 |

## 2.3 LCD 接口引脚定义

表 2-3 LCD接口引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| LCD_CLK | O | | LCD 时钟 |
| LCD_VSYNC | O | | LCD 列同步 |
| LCD_HSYNC | O | | LCD 行同步 |
| LCD_EN | O | | LCD 可视使能信号 |
| LCD_DAT_B[7:0] | O | | LCD 蓝色数据信号 7~0，7 为 MSB |
| LCD_DAT_G[7:0] | O | | LCD 绿色数据信号 7~0，7 为 MSB |
| LCD_DAT_R[7:0] | O | | LCD 红色数据信号 7~0，7 为 MSB |

## 2.4 VGA 引脚定义

表 2-4 VGA引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| VGA_B | O | | VGA 蓝色模拟信号 |
| VGA_G | O | | VGA 绿色模拟信号 |
| VGA_R | O | | VGA 红色模拟信号 |
| VGA_COMP | I | | 通过并联的 10nf 陶瓷电容和 10uf 钽电容连接至 VGA_A3V3 |
| VGA_EN | O | | 输出使能 |
| VGA_HSYNC | O | | 行同步 |
| VGA_REXT | I | | 外部电阻，510ohm 接地 |

| | | | |
|---|---|---|---|
| VGA_VREFIN | I | | 参考 1.22v 输入，接至 VGA_VREFOUT |
| VGA_VREFOUT | O | | 参考 1.22v 输出，通过 10nf 电容接地 |
| VGA_VSYNC | O | | 列同步 |

## 2.5 VR 引脚定义

表 2-5 VR引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| RTC_VDD3V3 | I | | 电源，外接 3.0V 电池 |
| RTC_VR_VOUT | O | | 外接 10nf 电容到地 |
| RTC_VR_CEXT | O | | 外接 4.7uf 电容到地 |
| RTC_VR_VBG | O | | 测试信号，需悬空 |
| RTC_VR_PD | I | | 关断模式，高有效。<br>正常使用需接地 |

## 2.6 DDR2 引脚定义

表 2-6 DDR引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| DDR2_DQ[31:00] | B | | 外部存储数据总线 |
| DDR2_A[14:00] | O | | 外部存储地址总线第 0 位 |
| DDR2_DQS[3:0] | B | | 输入输出数据 strobe 信号 |
| DDR2_DQM[3:0] | O | | 写数据屏蔽信号 |
| DDR2_CKp[1:0]<br>DDR2_CKn[1:0] | O | | 时钟信号 |
| DDR2_CKE[1:0] | O | | 时钟使能信号 |
| DDR2_ODT[1:0] | O | | ODT 信号 |
| DDR2_SCSn[1:0] | O | | 片选信号 |
| DDR2_BA[2:0] | O | | bank 选择信号 |
| DDR2_RASn | O | | 行选择 |
| DDR2_CASn | O | | 列选择 |
| DDR2_WEn | O | | 写信号 |
| DDR2_GATEI[1:0] | I | | 延迟测量，与 GATEO[1:0]相连 |
| DDR2_GATEO[1:0] | O | | 延迟测量，与 GATEI[1:0]相连 |

## 2.7 USB 引脚定义

表 2-7 USB引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| USB[3:0]_DM<br>USB[3:0]_DP | B | | USB 差分数据 |
| USB[3:0]_OVRCUR | I | | USB 过流，高有效 |
| USB[3:0]_REXT | | | USB 外部电阻，44.2 欧姆接地 |
| USB[3:0]_XI | | | USB 时钟，板级接地 |
| USB[3:0]_XO | I | | USB 时钟输入，四路时钟输入必须同相 |

## 2.8 EJTAG 引脚定义

表 2-8 JTAG引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| EJTAG_TCK | I | PU | TAP 时钟 |
| EJTAG_TRST | I | PU | TAP 复位，硬件需要下拉 |

| | | | |
|---|---|---|---|
| EJTAG_TDI | I | PU | TAP 数据输入 |
| EJTAG_TDO | O | | TAP 数据输出 |
| EJTAG_TMS | I | PU | TAP 工作模式 |
| TEST_JTAG_SEL | I | | 0：JTAG；1：EJTAG |

## 2.9 GMAC0 引脚定义

表 2-9 GMAC0引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| GMAC0_TX_CLK_I | I | | GMAC 125M 参考时钟输入 |
| GMAC0_TX_CLK_O | O | | GMAC 发送时钟输出 |
| GMAC0_TX0 | O | | GMAC 发送数据输出 0 |
| GMAC0_TX1 | O | | GMAC 发送数据输出 1 |
| GMAC0_TX2 | O | | GMAC 发送数据输出 2 |
| GMAC0_TX3 | O | | GMAC 发送数据输出 3 |
| GMAC0_TX_CTL | O | | GMAC 发送控制 |
| GMAC0_RX_CLK_I | I | | GMAC 接收时钟输入 |
| GMAC0_RX0 | I | | GMAC 接收数据输入 0 |
| GMAC0_RX1 | I | | GMAC 接收数据输入 1 |
| GMAC0_RX2 | I | | GMAC 接收数据输入 2 |
| GMAC0_RX3 | I | | GMAC 接收数据输入 3 |
| GMAC0_RX_CTL | I | | GMAC 接收控制 |
| GMAC0_MDC | O | | 读写 PHY 的时钟信号 |
| GMAC0_MDIO | B | | 读写 PHY 的数据信号 |

## 2.10 GMAC1 引脚定义

表 2-10 GMAC1引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| GMAC1_TX_CLK_I | I | | GMAC1125M 参考时钟输入 |
| GMAC1_TX_CLK_O | O | | GMAC1 发送时钟输出 |
| GMAC1_RX_CKL_I | I | | GMAC1 接收时钟输入 |

## 2.11 AC97 引脚定义

表 2-4 AC97引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| AC97_BIT_CLK | I | | AC97 时钟输入 |
| AC97_DATA_I | I | | AC97 数据输入 |
| AC97_DATA_O | O | | AC97 数据输出 |
| AC97_SYNC | O | | AC97 同步信号 |
| AC97_RESET | O | | AC97 复位信号 |

## 2.12 SPI 引脚定义

表 2-12 SPI引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| SPI0_CLK | O | | SPI0 时钟 |
| SPI0_MISO | I | | SPI0 输入数据 |
| SPI0_MOSI | O | | SPI0 输出数据 |
| SPI0_CS0 | O | | SPI0 选通信号 0 |
| SPI1_CLK | O | | SPI1 时钟 |

| | | | |
|---|---|---|---|
| SPI1_MISO | I | | SPI1 输入数据，需外部上拉 |
| SPI1_MOSI | O | | SPI1 输出数据 |
| SPI1_CS0 | O | | SPI1 选通信号 0 |

## 2.13 UART 引脚定义

表 2-53 UART引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| UART0_RX | I | | UART0 发送数据 |
| UART0_TX | O | | UART0 接收数据 |
| UART0_RTS | O | | UART0 请求发送 |
| UART0_CTS | I | | UART0 允许发送 |
| UART0_DSR | I | | UART0 设备准备好 |
| UART0_DTR | O | | UART0 终端准备好 |
| UART0_DCD | I | | UART0 载波检测 |
| UART0_RI | I | | UART0 振铃提示 |
| UART1_TX | O | | UART1 发送数据 |
| UART1_RX | I | | UART1 接收数据 |
| UART1_RTS | O | | UART1 请求发送 |
| UART1_CTS | I | | UART1 允许发送 |
| UART2_TX | O | | UART2 发送数据 |
| UART2_RX | I | | UART2 接收数据 |
| UART3_TX | O | | UART3 发送数据 |
| UART3_RX | I | | UART3 接收数据 |

## 2.14 I2C 引脚定义

表 2-64 2C引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| I2C_SCL | I | | 第一路 I2C 时钟 |
| I2C_SDA | B | | 第一路 I2C 数据 |

## 2.15 CAN 引脚定义

表 2-15 CAN引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| CAN0_RX | I | | CAN0 数据输入 |
| CAN0_TX | O | | CAN0 数据输出 |
| CAN1_RX | I | | CAN1 数据输入 |
| CAN1_TX | O | | CAN1 数据输出 |

## 2.16 NAND 引脚定义

表 2-16 NAND flash引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| NAND_CLE | O | | NAND 命令锁存 |
| NAND_ALE | O | | NAND 地址锁存 |
| NAND_RD | O | | NAND 读信号 |
| NAND_WR | O | | NAND 写信号 |
| NAND_CE | O | | NAND 片选信号 |
| NAND_RDY | I | | NAND 忙信号 |
| NAND_D6 | O | | NAND 数据信号 6 |

| | | | |
|---|---|---|---|
| NAND_D7 | O | | NAND 数据信号 7 |

## 2.17 PWM 引脚定义

表 2-17 PWM引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| PWM0 | O | | PWM0 波形输出 |
| PWM1 | O | | PWM1 波形输出 |
| PWM2 | O | | PWM2 波形输出 |
| PWM3 | O | | PWM3 波形输出 |

## 2.18 LPC 引脚定义

表 2-18 系统时钟引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| LPC_AD0 | I | | LPC 地址数据线 0 |
| LPC_AD1 | O | | LPC 地址数据线 1 |
| LPC_AD2 | I | | LPC 地址数据线 2 |
| LPC_AD3 | O | | LPC 地址数据线 3 |
| LPC_FRAMEn | I | | LPC 帧信号 |
| LPC_SERIRQn | B | | LPC 串行中断 |

## 2.19 PCI 引脚定义

表 2-19 系统时钟引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| PCI_AD[31:00] | BPCI | | PCI 数据地址线 |
| PCI_RESETn | BPCI | | 复位 |
| PCI_CBEn[3:0] | BPCI | | 字节使能 |
| PCI_DEVSELn | BPCI | | 设备选择 |
| PCI_FRAMEn | BPCI | | 帧周期 |
| PCI_IDSEL | I | | 设备选择信号 |
| PCI_IRDYn | BPCI | | 主设备准备好 |
| PCI_PAR | BPCI | | 校验位 |
| PCI_PERR | BPCI | | 奇偶校验 |
| PCI_REQn[1:0] | BPCI | | 总线占有请求 |
| PCI_GNTn[1:0] | BPCI | | 总线占有允许 |
| PCI_SERR | BPCI | | 系统错误报告 |
| PCI_STOPn | BPCI | | 停止数据传送 |
| PCI_TRDYn | BPCI | | 从设备准备好 |

## 2.20 SATA 引脚定义

表 2-20 SATA引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| SATA[1:0]_RXN | I | | SATA 数据差分接收 |
| SATA[1:0]_RXP | I | | SATA 数据差分接收 |
| SATA[1:0]_TXN | O | | SATA 数据差分输出 |
| SATA[1:0]_TXP | O | | SATA 数据差分输出 |
| SATA_REFCLKN | I | | 参考时钟，设置参考正文 |

| | | | |
|---|---|---|---|
| SATA_REFCLKP | | | |
| SATA_REXT | I | | 外部参考电阻，190 欧接地 |

## 2.21 ACPI 引脚定义

表 2-21 ACPI引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 | 电压域 |
|---|---|---|---|---|
| ACPI_SYS_RSTN | I | | 系统重启键 | CORE |
| RSM_BATLOWN | I | | 电池电量低 | RESUME |
| RSM_LID | I | | LCD 屏盖状态输入 | RESUME |
| RSM_RIN | I | | 输入 ring 信号，用来唤醒系统 | RESUME |
| RSM_PMEn | I | | PCI 唤醒系统信号 | RESUME |
| RTC_RSMRSTN | I | | 用来重启 resume 域的逻辑 | RTC |
| RTC_RTCRSTN | I | | 信号有效时，重启 RTC 域逻辑 | RTC |
| RTC_PWROK | I | | CORE well power ok | RTC |
| RSM_PWRBTNN | I | | 开机键 | RESUME |
| RSM_SUS_STATN | 0 | | 通知外设系统将进入睡眠状态 | RESUME |
| RSM_PLTRSTN | 0 | | 系统重启信号 | RESUME |
| RSM_S3N | 0 | | S3 信号关掉非关键电路的电源 | RESUME |
| RSM_S4N | 0 | | S4 信号用来关掉内存的电源 | RESUME |
| RSM_ACPI_EN | I | | ACPI 使能 | RESUME |
| RSM_SoC_EN | I | | 芯片工作在 SOC 模式 | RESUME |

## 2.22 PS2 引脚定义

表 2-22 PS2引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| KB_CLK | I | | 键盘时钟 |
| KB_DAT | O | | 键盘数据 |
| MS_CLK | I | | 鼠标时钟 |
| MS_DAT | O | | 鼠标数据 |

## 2.23 PLL 引脚定义

表 2-23 PLL引脚定义

| 信号名称 | 方向 | 上下拉 | 描述 |
|---|---|---|---|
| PLL_1_AVDD33 | I | | 3.3 伏模拟电源 |
| PLL_1_AVSS33 | I | | 3.3 伏模拟地 |
| PLL_2_AVDD33 | I | | 3.3 伏模拟电源 |
| PLL_2_AVSS33 | I | | 3.3 伏模拟地 |

## 2.24 电源/地引脚

表 2-7 引脚电源地

| 信号名称 | 电压值 | 电压域 | 描述 |
|---|---|---|---|
| VDD1V2 | 1.2v | CORE | CORE 域 |
| VDD1V8 | 1.8v | DDR2 | DDR2 电压域 |
| VDD3V3 | 3.3v | PAD | PAD 电压域 |
| VREF0V9 | 0.9v | 参考电源 | DDR2 参考电源 |
| VSS | 0v | 接地 | 接地 |

# 3 启动和时钟配置

## 3.1 上电配置引脚

龙芯 1A 具有多种启动方式及工作模式，由一些配置引脚的上拉或者下拉来进行选择。具体配置信息见下表（上拉为 1，下拉为 0）：

| 引脚名称 | 描述 |
|---|---|
| PWM[1:0] | CPU 初始频率配置<br>0: 5 倍频<br>1: 6 倍频<br>2: 7 倍频<br>3: 8 倍频 |
| PWM2 | DDR 初始频率配置<br>0: 5 倍频<br>1: 6 倍频 |
| {NAND_CLE,PWM3} | 启动源选择<br>‘b00: SPI 启动<br>‘b01: LPC 启动(仅支持 SST49LF040/080 类型的 Flash)<br>‘b1x: NAND 启动(复用 LPC 引脚) |
| LCD_DAT_B0 | NAND 启动模式选择<br>0: 普通模式<br>1: ECC 模式 |
| NAND_ALE | PCI 外部仲裁选择<br>0: 使用内部仲裁器<br>1: 使用外部仲裁器 |
| NAND_RD | PCI 启动支持(未经验证，暂不建议使用)<br>0: 关闭<br>1: 开启，复位后即开启 PCI 上 0x1fc00000 的 MEM 空间 |
| NAND_WR | PCI 主模式选择<br>0: 从模式<br>1: 主模式<br>两种模式的主要区别在于主模式下 PCI_RESETn 为输出，从模式的为输入。当选择 PCIX 时，从模式的控制器还会在复位后采样总线频率信息。 |
| NAND_CE | PCIX 模式选择<br>0: PCI 模式<br>1: PCIX 模式 |
| {NAND_D7,NAND_D6} | PCIX 速度选择（PCI 模式时应为 0）<br>1: PCIX66<br>其它: 不可用 |

## 3.2 时钟架构

龙芯 1A 内部有多个时钟域，如下图所示。来自 XTALI/O 的时钟送到内部 5 个 PLL，产生 CPU 等主要模块的工作时钟；USB、GMAC、SATA、PCI 等模块

在接口处使用各自的时钟（或参考时钟）。对于跨时钟域的数据路径，芯片内部设置了同步模块。芯片时钟输入引脚说明见表 XX。

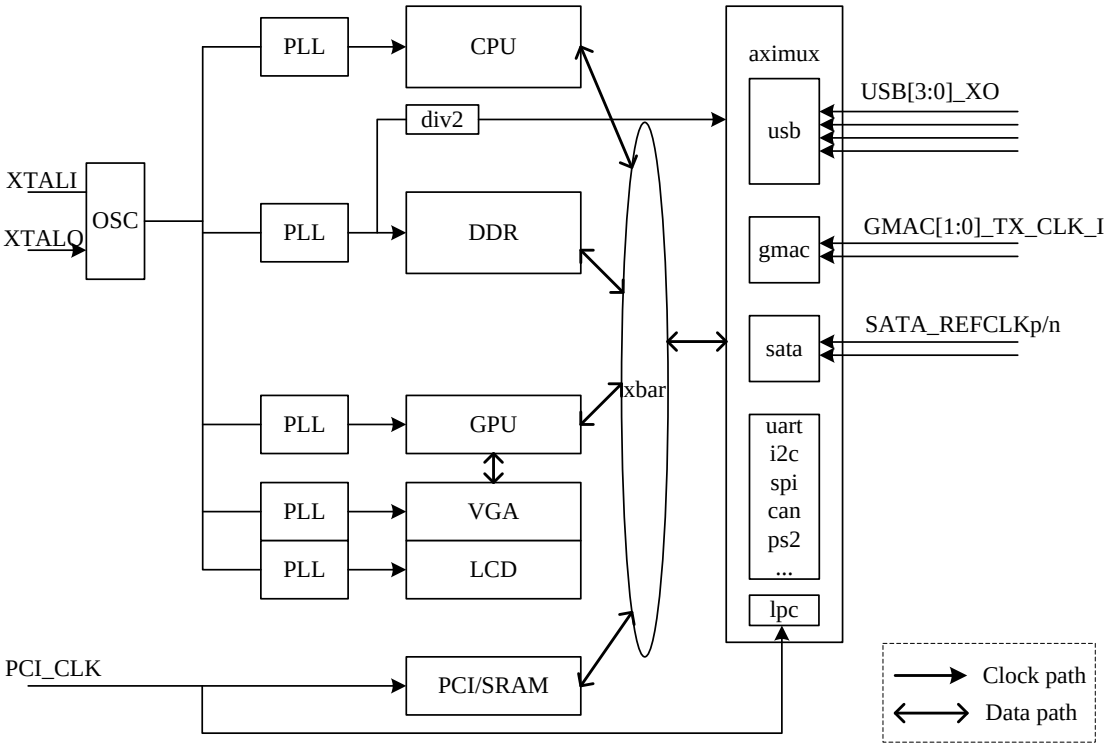



图 3-1 龙芯 1A 时钟架构

表 3-1 输入时钟

| 时钟引脚 | 类型 | 描述 |
|---|---|---|
| XTALI<br>XTALO | I/O | PLL 参考时钟，连接 25MHz~33MHz 晶体 |
| PCI_CLK | I | PCI/SRAM 控制器接口时钟，3.3V 全摆幅，最高频率 66Mhz |
| USB[3:0]_XO | I | USB PHY 参考时钟，3.3V 全摆幅，12MHz，四路同相（USB[3:0]_XI 不是时钟输入，需接地） |
| GMAC[1:0]TX_CLK_I | I | RGMII 接口参考时钟，3.3V 全摆幅，125MHz |
| SATA_REFCLKp<br>SATA_REFCLKn | DIFF IN | SATA PHY 参考时钟，频率 25/50/100/125MHz，差分共模点 0.175~2V，差分电压 0.35~0.85V，50ohm 输入阻抗 |

## 3.3 时钟配置

系统启动后软件可以通过以下寄存器调整 PLL 的频率输出。

表 3-2 时钟配置寄存器

| 寄存器 | 位域 | 名称 | 访问 | 描述 |
|---|---|---|---|---|
| corepll_cfg<br>0x1fe78030 | 15 | ddrcfg_w_en | W | DDR 时钟配置参数写使能<br>使用 sw 指令更新此寄存器时，只有往该位写 1 才能更新[7:4]的值 |
| | 11 | cpucfg_w_en | W | CPU 时钟配置参数写使能<br>使用 sw 指令更新此寄存器时，只有往该位写 1 才 |

| | | | | |
|---|---|---|---|---|
| | | | | 能更新[3:0]的值 |
| | 7 | ddrcfg_en | W | DDR 时钟频率配置使能，写 1 后 DDR 时钟倍频数才能产生作用 |
| | 6:4 | ddr_mul | W | DDR 时钟倍频数<br>0~7: 3~10 倍频 |
| | 3 | cpucfg_en | W | CPU 时钟频率配置使能，写 1 后 CPU 时钟倍频数才能产生作用 |
| | 2:0 | cpu_mul | W | CPU 时钟倍频数<br>0~7: 4~11 倍频 |
| gpupll_cfg<br>0x1fd00414 | 31:14 | gpu_frac | RW | 倍频数的小数部分 |
| | 13:12 | gpu_od | RW | 输出分频 |
| | 11:8 | gpu_n | RW | 输入分频 |
| | 7:0 | gpu_m | RW | 倍频分频 |
| lcdpll_cfg<br>0x1fd00410 | 31:14 | lcd_frac | RW | 倍频数的小数部分 |
| | 13:12 | lcd_od | RW | 输出分频 |
| | 11:8 | lcd_n | RW | 输入分频 |
| | 7:0 | lcd_m | RW | 倍频分频 |
| vgapll_cfg<br>0x1fd00410 | 31:14 | vga_frac | RW | 倍频数的小数部分 |
| | 13:12 | vga_od | RW | 输出分频 |
| | 11:8 | vga_n | RW | 输入分频 |
| | 7:0 | vga_m | RW | 倍频分频 |

GPU、LCD 和 VGA 三个 PLL 的配置形式相同，其输出频率计算方法为：

$$Fout = (Fref / n) * (m+frac/218) / 2od$$

在参数选取时应注意满足以下约束

1. 5MHz < Fref / n < 50MHz
2. 200MHz < Fref / n * m < 700MHz
3. GPU 频率不低于 LCD 和 VGA 频率

# 4 地址空间分配

## 4.1 一级 AXI 交叉开关上模块的地址空间

表 4-1 AXI 各模块地址分配

| 地址空间 | 设备 | 说明 |
|---|---|---|
| 0x0000,0000 - 0x0fff,ffff | DDR Slave 0 | 256MB |
| 0x1000,0000 - 0x13ff,ffff | PCI mem 空间 0 | 64MB |
| 0x1400,0000 - 0x17ff,ffff | PCI mem 空间 1 | 64MB |
| 0x1800,0000 - 0x1bff,ffff | PCI mem 空间 2 | 64MB |
| 0x1c00,0000 - 0x1c0f,ffff | PCI IO 空间 | 1MB |
| 0x1c10,0000 - 0x1c10,ffff | PCI 配置空间 | 64KB |
| 0x1c11,0000 - 0x1c11,00ff | PCI 控制寄存器地址空间 | 256B |
| 0x1c11,0100 - 0x1c1f,ffff | | RESERVED |
| 0x1c20,0000 - 0x1c2f,ffff | DC | 1MB |
| 0x1c30,0000 - 0x1c3f,ffff | GPU | 1MB |
| 0x1c40,0000 - 0x1eff,ffff | | RESERVED |
| 0x1f00,0000 - 0x1fff,ffff | AXI MUX Slave | 16MB |
| 0x2000,0000 - 0x3fff,ffff | | RESERVED |
| 0x4000,0000 - 0x7fff,ffff | DDR Slave 1 | 1GB |

## 4.2 AXI MUX 下各模块的地址空间

表 4-2 AXI-MUX 各模块地址分配

| 地址空间 | 设备 | 说明 |
|---|---|---|
| 0x1f00,0000 - 0x1f7f,ffff | SPI0-memory | 8MB |
| 0x1f80,0000 - 0x1fbf,ffff | SPI1-memory | 4MB |
| 0x1fc0,0000 - 0x1fcf,ffff | Boot | 1MB，根据系统启动方式映射到 SPI、LPC 或 NAND |
| 0x1fd0,0000 - 0x1fdf,ffff | CONFREG | 1MB |
| 0x1fe0,0000 - 0x1fe0,ffff | USB | 64KB |
| 0x1fe1,0000 - 0x1fe1,ffff | GMAC0 | 64KB |
| 0x1fe2,0000 - 0x1fe2,ffff | GMAC1 | 64KB |
| 0x1fe3,0000 - 0x1fe3,ffff | SATA | 64KB |
| 0x1fe4,0000 - 0x1fe7,ffff | APB-devices | 256KB |
| 0x1fe8,0000 - 0x1feb,ffff | SPI0-IO | 256KB |
| 0x1fec,0000 - 0x1fef,ffff | SPI1-IO | 256KB |
| 0x1ff0,0000 - 0x1fff,ffff | LPC-IO | 1MB |

## 4.3 APB 各模块的地址空间分配

表 4-3 APB 各模块地址分配

| 地址空间 | 模块 | 说明 |
|---|---|---|
| 0x1fe40000-0x1fe43fff | UART0 | 16KB |
| 0x1fe44000-0x1fe47fff | UART1 | 16KB |

| | | |
|---|---|---|
| 0x1fe48000-0x1fe4bfff | UART2 | 16KB |
| 0x1fe4c000-0x1fe4ffff | UART3 | 16KB |
| 0x1fe50000-0x1fe53fff | CAN0 | 16KB |
| 0x1fe54000-0x1fe57fff | CAN1 | 16KB |
| 0x1fe58000-0x1fe5bfff | I2C-0 | 16KB |
| 0x1fe5c000-0x1fe5ffff | PWM | 16KB |
| 0x1fe60000-0x1fe63fff | PS2 | 16KB |
| 0x1fe64000-0x1fe67fff | RTC | 16KB |
| 0x1fe68000-0x1fe6bfff | I2C-1 | 16KB |
| 0x1fe6c000-0x1fe6ffff | HPET | 16KB |
| 0x1fe70000-0x1fe73fff | I2C-2 | 16KB |
| 0x1fe74000-0x1fe77fff | AC97 | 16KB |
| 0x1fe78000-0x1fe7bfff | NAND | 16KB |
| 0x1fe7c000-0x1fe7ffff | ACPI | 16KB |

# 5 DDR2

龙芯 1A 集成了内存控制器，兼容 DDR2 SDRAM 标准（JESD79-2B）。龙芯 1A 提供 JESD79-2B 兼容的内存读写操作。

## 5.1 DDR2 SDRAM 控制器特性

龙芯 1A CPU 支持两个物理 Bank，通过两个片选信号和 18 位的地址总线（15 位行/列地址和 3 位逻辑 Bank 地址）实现最大地址空间是 64Gb (236)。

龙芯 1A 支持所有的与 JESD79-2B 兼容的内存颗粒。DDR2 控制器参数能被设置为支持指定的内存芯片类型。芯片选择信号（CS_n）的最大数目是 2。行地址（RAS_n）和列地址（CAS_n）的最大宽度分别是 15 和 14。还有 3 位的逻辑 bank 信号（BANK_n）。

CPU 内存的物理地址能被转换位行/列地址，见

表 5-1。例如，2 个 CS_n 信号，8 个 banks，12 位行地址和 12 位列地址。

表 5-1 DDR2 SDRAM行/列地址转换

| 35 31 | 30 30 | 29 18 | 17 15 | 14 3 | 2 0 |
|---|---|---|---|---|---|
| | CS_n | RAS_n | BANK_n | CAS_n | Byte |

内存控制器接收从处理器或外部设备发送的内存读写请求。无论是读还是写操作，内存控制器都处在 slave 状态。

内存控制器中实现了动态页管理功能。对于内存的一次存取，不需软件设计者的干预，控制器会在硬件电路上选择 Open Page/Close Page 策略。内存控制器特性包括：

- 全流水的命令和数据读写；
- 通过合并和重排序增加带宽；
- 通过丰富的寄存器读写端口修改基本的参数；
- 内置 Delay Compensation Circuit(DCC)，用来可靠的发送/接收数据；
- 频率：133MHz-333MHz;

## 5.2 DDR2 SDRAM 读协议

图 5-1 中显示 DDR2 SDRAM 读协议，命令（CMD）包括 RAS_n，CAS_n 和 WE_n。当一个读请求发生时，RAS_n=1，CAS_n=0，WE_n=1。

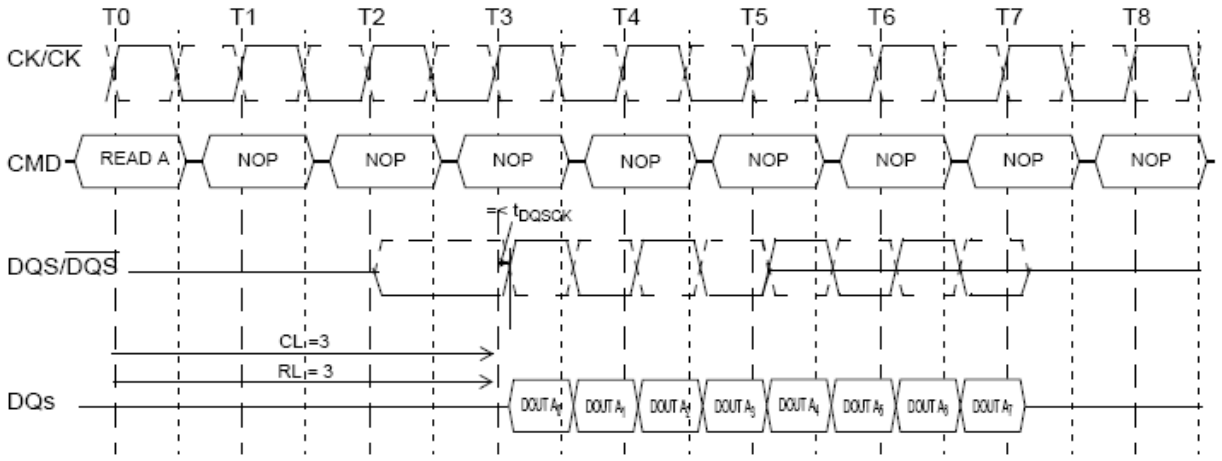



图 5-1 DDR2 SDRAM 读协议， Cas Latency = 3, Read Latency = 3, Burst Length = 8

## 5.3 DDR2 SDRAM 写协议

在图 5-2 中显示 DDR2 SDRAM 写协议，命令（CMD）包括 RAS_n，CAS_n 和 WE_n。当写请求发生时，RAS_n=1，CAS_n＝0，WE_n＝0。与读协议不同，DQM 用来识别需要被写的字节数。DQM 和 DQS 是同步的。

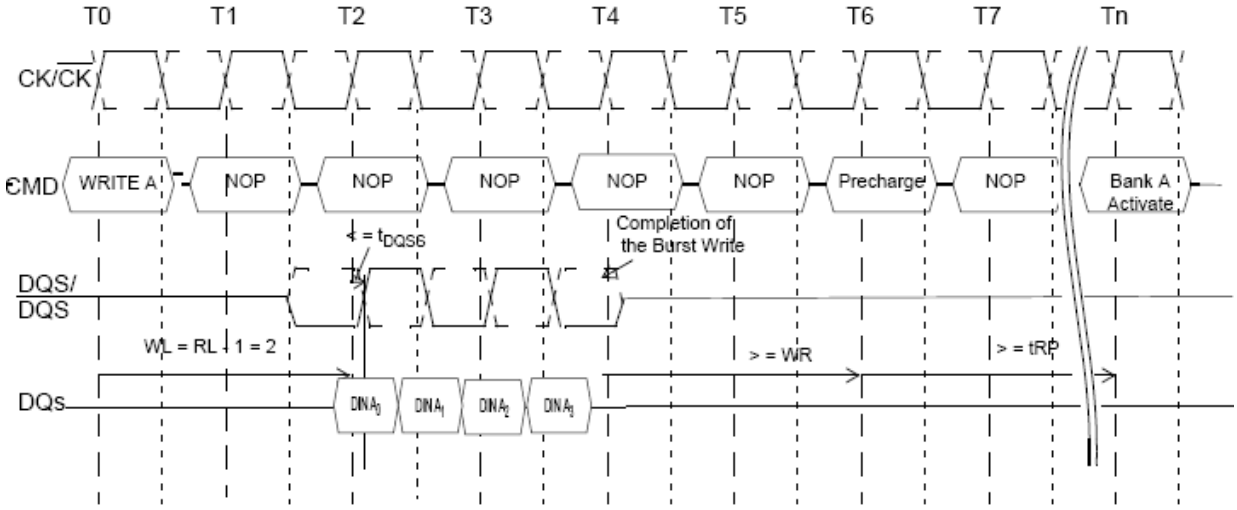



图 5-2 DDR2 SDRAM 写协议，Cas Latency = 3, Write Latency = Read Latency -1 = 2, Burst Length = 4.

## 5.4 DDR2 SDRAM 参数配置

由于系统中可能使用不同类型的 DDR2 SDRAM，因此，在系统上电复位以后，需要对 DDR2 SDRAM 进行配置。在 JESD79-2B 中规定了详细的配置操作和配置过程，在没有完成 DDR2 的内存初始化操作之前，DDR2 不可用。内存初始化操作执行顺序如下：

(1) 系统复位，aresetn 信号被置 0，此时控制器内部所有寄存器内容将被清除为初始值。

(2) 系统解复位，aresetn 信号被置为 1。

(3) 向配置寄存器地址发写指令，配置所有 29 个配置寄存器。此时如果写 CTRL_03，应将其中参数 START 设为 0。所有寄存器都必须正确配置才可以正常工作。

(4) 向配置寄存器 CTRL_03 中发写指令，将参数 START 设为 1。结束后内存控制器将自动对内存发起初始化指令。

具体的配置操作是对物理地址 0x0000 0000 0FFF FE00 相对应的 29 个 64 位寄存器写入相应的配置参数。一个寄存器可能会包括一个或多个参数的数据。这些配置寄存器及其包含的参数意义如下表（寄存器中未使用的位均为保留位），表中还给出了基于 DDR2 667 的一种寄存器配置方式，具体的配置可以根据实际情况再决定：

表 5-1 DDR2 SDRAM 配置参数寄存器格式

| 参数名称 | 位 | 缺省值 | 范围 | 描述 |
|---|---|---|---|---|
| CONF_CTL_00[31:0] Offset: 0x00 | | | | DDR2 667：0x00000101 |
| AREFRESH | 24:24 | 0x0 | 0x0-0x1 | 根据 auto_refresh_mode 参数的设置，向内存发起自动刷新命令（只写） |
| AP | 16:16 | 0x0 | 0x0-0x1 | 是否使能内存控制器自动刷新功能，置 1，表示内存访问为 CLOSE PAGE 方式。 |
| ADDR_CMP_EN | 8:8 | 0x0 | 0x0-0x1 | 是否允许命令队列重排序逻辑对地址冲突进行检测 |
| ACTIVE_AGING | 0:0 | 0x0 | 0x0-0x1 | 是否允许对命令队列中的命令进行 aging 记录，防止低优先级命令饿死 |
| CONF_CTL_00[63:32] Offset: 0x00 | | | | DDR2 667：0x01000100 |
| DDR2_SDRAM_MODE | 56:56 | 0x0 | 0x0-0x1 | 内存控制器 DDRI 和 DDRII 模式设置，对于 DDRII，应当置 1 |
| CONCURRENTAP | 48:48 | 0x0 | 0x0-0x1 | 是否允许控制器对一个 bank 进行 auto precharge 时，对另外一个 bank 发出命令。注：部分内存条不支持 |
| BANK_SPLIT_EN | 40:40 | 0x0 | 0x0-0x1 | 是否允许命令队列重排序逻辑对 bank 进行拆分（split） |
| AUTO_REFRESH_MODE | 32:32 | 0x0 | 0x0-0x1 | 设置 auto-refresh 是在下一个 burst 还是下一个命令边界发出 |
| CONF_CTL_01[31:0] Offset: 0x10 | | | | DDR2 667：0x00010000 |
| ECC_DISBALE_W_UC_ERR | 24:24 | 0x0 | 0x0-0x1 | 当检测到不可恢复的错误时，是否将 ECC 关闭 |
| DQS_N_EN | 16:16 | 0x0 | 0x0-0x1 | 是否使能差分 DQS |
| DLL_BYPASS_MODE | 8:8 | 0x0 | 0x0-0x1 | 是否使能 DLL BYPASS 模式，在 DLL BYPASS 模式下，所有 DLL 的参数将会使用带_BYPASS 结尾的参数，BYPASS 方式与普通方式下，计算的单位是不同的，带 BYPASS 的参数是以延迟线个数为单位，其它的参数是以周期的 1/128 为单位。通常情况下，不需要设置为 DLL_BYPASS_MODE。 |
| DLLLOCKREG | 0:0 | 0x0 | 0x0-0x1 | 指示 DLL 是否已锁定（只读），只有在 DLL 锁定之后，对内存发起的读写操作才能有效到达内存，所以，可以用本位判断第一次写内存的时机。 |
| CONF_CTL_01[63:32] Offset: 0x10 | | | | DDR2 667：0x00000000 |

| | | | | |
|---|---|---|---|---|
| FWC | 56:56 | 0x0 | 0x0-0x1 | 是否强制进行写检查，当这个参数设置后，内存控制器将用 xor_check_bits 参数指定的数与数据进行异或写入内存（只写） |
| FAST_WRITE | 48:48 | 0x0 | 0x0-0x1 | 是否允许控制器打开快速写功能。打开快速写功能后，控制器在未收到全部写数据后即向内存模块发出写命令。 |
| ENABLE_QUICK_SREFRESH | 40:40 | 0x0 | 0x0-0x1 | 是否使能快速自刷新。当这个参数使能后，内存的初始化未进行完就进入自刷新状态 |
| EIGHT_BANK_MODE | 32:32 | 0x0 | 0x0-0x1 | 指示内存模块是否有 8 个 bank |
| CONF_CTL_02[31:0] Offset: 0x20 | | | | DDR2 667：0x00000000 |
| NO_CMD_INIT | 24:24 | 0x0 | 0x0-0x1 | 在内存初始化过程中，是否禁止在内存模块的 tDLL 时间内发出其它命令 |
| INTRPTWRITENA | 16:16 | 0x0 | 0x0-0x1 | 是否允许用 autoprecharge 命令加上对同一 bank 的其它写命令打断前一个写命令 |
| INTRPTREADA | 8:8 | 0x0 | 0x0-0x1 | 是否允许用 autoprecharge 命令加上对同一 bank 的其它读命令打断前一个读命令 |
| INTRPTAPBURST | 0:0 | 0x0 | 0x0-0x1 | 是否允许对另一 bank 的其它命令打断当前的 auto-precharge 命令 |
| CONF_CTL_02[63:32] Offset: 0x20 | | | | DDR2 667：0x0**0**000**0**01 |
| PRIORITY_EN | 56:56 | 0x0 | 0x0-0x1 | 是否使能命令队列重排序逻辑使用优先级 |
| POWER_DOWN | 48:48 | 0x0 | 0x0-0x1 | 当使能这个参数时，内存控制器将用 pre-charge 命令关闭内存模块的所有页面，使时钟使能信号为低，不发送收到的所有命令，直到这个参数重新设置为 0 |
| PLACEMENT_EN | 40:40 | 0x0 | 0x0-0x1 | 是否使能命令重排序逻辑 |
| ODT_ADD_TURN_CLK_EN | 32:32 | 0x0 | 0x0-0x1 | 在对不同片选的快速背对背读或者写命令中间是否插入一个 turn-around 时钟。通常情况下，插入一个这样的周期是对内存是需要的。 |
| CONF_CTL_03[31:0] Offset: 0x30 | | | | DDR2 667：0x0**0**000100 |
| RW_SAME_EN | 24:24 | 0x0 | 0x0-0x1 | 在命令队列重排序逻辑中是否考虑对同一 bank 读写命令的重组 |
| REG_DIMM_EN | 16:16 | 0x0 | 0x0-0x1 | 是否使能 registered DIMM 内存模组 |
| REDUC | 8:8 | 0x0 | 0x0-0x1 | 是否只使用 32 位位宽的内存数据通道，须置为 1 |
| PWRUP_SREFRESH_EXIT | 0:0 | 0x0 | 0x0-0x1 | 是用 self-refresh 命令而不是用正常的内存初始化命令来脱离下电模式 |
| CONF_CTL_03[63:32] Offset: 0x30 | | | | DDR2 667：0x010**0**0000 |
| SWAP_PORT_RW_SAME_EN | 56:56 | 0x0 | 0x0-0x1 | 当 swap_en 使能时，该参数决定是否将同一端口上的类似命令进行交换 |
| SWAP_EN | 48:48 | 0x0 | 0x0-0x1 | 在使能命令队列重排序逻辑时，当高优先级命令到达时，是否将正在执行的命令与新命令交换 |
| START | 40:40 | 0x0 | 0x0-0x1 | 是否开始内存的初始化工作。需要在所有的参数配置完成之后，再设置该位，让内存进入初始化配置。在没有完成其它位的配置之前就配置该位，很可能导致内存访问错误。 |
| SREFRESH | 32:32 | 0x0 | 0x0-0x1 | 内存模块是否进入自刷新工作模式 |
| CONF_CTL_04[31:0] Offset: 0x40 | | | | DDR2 667：0x00010101 |
| WRITE_MODEREG | 24:24 | 0x0 | 0x0-0x1 | 是否写内存模块的 EMRS 寄存器（只写） |
| WRITEINTERP | 16:16 | 0x0 | 0x0-0x1 | 定义是否能用一个读命令取打断一个写突发 |
| TREF_ENABLE | 8:8 | 0x0 | 0x0-0x1 | 是否使能控制器内部的自动刷新功能，通常的情况下，应该将该位置 1 |
| TRAS_LOCKOUT | 0:0 | 0x0 | 0x0-0x1 | 是否在 tRAS 时间到期之前发出 auto-prechareg 命令 |
| CONF_CTL_04[63:32] Offset: 0x40 | | | | DDR2 667：0x01000202 |
| RTT_0 | 57:56 | 0x0 | 0x0-0x3 | 定义所有内存模块的片上终端电阻的阻值。这个值将在向内存发起初始化操作时写入内存颗粒。具体的配置应当参考相应的内存颗粒手册。<br>00 –disable 01 – 75ohm |

| | | | | |
|---|---|---|---|---|
| | | | | 10 – 150 ohm 11 -reserved |
| CTRL_RAW | 49:48 | 0x0 | 0x0-0x3 | 设置 ECC 的检错和纠错模式<br>2'b00 – 不使用 ECC<br>2'b01 – 只报错，不纠错<br>2'b10 – 没有使用 ECC 设备<br>2'b11 – 使用 ECC 报错纠错 |
| AXI0_W_PRIORITY | 41:40 | 0x0 | 0x0-0x3 | 设置 AXI0 端口写命令优先级 |
| AXI0_R_PRIORITY | 33:32 | 0x0 | 0x0-0x3 | 设置 AXI0 端口读命令优先级 |
| CONF_CTL_05[31:0] Offset: 0x50 | | | | DDR2 667：0x04050202 |
| COLUMN_SIZE | 26:24 | 0x0 | 0x0-0x7 | 设置实际列地址数和最大列地址数(14)之间的差值，应该根据具体的内存颗粒进行配置。<br>内存所用列地址数 = 14 - COLUMN_SIZE |
| CASLAT | 18:16 | 0x0 | 0x0-0x7 | 设置 CAS latency 值。应当根据具体的内存颗粒在不同的运行频率下进行配置。 |
| ADDR_PINS | 10:8 | 0x0 | 0x0-0x7 | 设置实际地址引脚数和最大地址数(15)之间的差值<br>内存所用地址线数 = 14 – ADDR_PINS |
| RTT_PAD_TERMINATION | 1:0 | 0x0 | 0x0-0x3 | 设置内存控制器 pad 的终端电阻阻值，与内存颗粒上的终端电阻相对应，具体的配置应当参考相应的内存颗粒手册。 |
| CONF_CTL_05[63:32] Offset: 0x50 | | | | DDR2 667：0x00000000 |
| Q_FULLNESS | 58:56 | 0x0 | 0x0-0x7 | 定义内存控制器命令队列中有多少命令时认为命令队列满 |
| PORT_DATA_ERROR_TYPE | 50:48 | 0x0 | 0x0-0x7 | 定义内存控制器端口上数据错误类型（只读）<br>位 0 – 突发数据个数大于 16<br>位 1 – 写数据交错<br>位 2 – ECC 2 位错 |
| OUT_OF_RANGE_TYPE | 42:40 | 0x0 | 0x0-0x7 | 定义发生越界访问时的错误类型（只读） |
| MAX_CS_REG | 34:32 | 0x4 | 0x0-0x4 | 定义控制器所用片选个数（只读） |
| CONF_CTL_06[31:0] Offset: 0x60 | | | | DDR2 667：0x03050203 |
| TRTP | 26:24 | 0x0 | 0x0-0x7 | 定义内存模组的读命令到 precharge 周期数，需要根据具体内存颗粒及运行频率进行配置。 |
| TRRD | 18:16 | 0x0 | 0x0-0x7 | 定义到不同 bank 的 active 命令时间间隔，需要根据具体内存颗粒及运行频率进行配置。 |
| TEMRS | 10:8 | 0x0 | 0x0-0x7 | 定义内存模组初始化时的 emrs 时间，一般内存颗粒配置的周期为 2。 |
| TCKE | 2:0 | 0x0 | 0x0-0x7 | 定义 CKE 信号最小脉宽 |
| CONF_CTL_06[63:32] Offset: 0x60 | | | | DDR2 667：0x0a040306 |
| APREBIT | 59:56 | 0x0 | 0x0-0xf | 定义用哪位地址线向内存发出 autoprecharge 命令，一般为 bit 10。 |
| WRLAT | 50:48 | 0x0 | 0x0-0x7 | 定义从写命令发出到接收到第一个数据的时间（按时钟周期数），一般为 CASLAT -1。 |
| TWTR | 42:40 | 0x0 | 0x0-0x7 | 定义从写命令切换到读命令所需要的时钟周期数，需要根据具体内存颗粒及运行频率进行配置。 |
| TWR_INT | 34:32 | 0x0 | 0x0-0x7 | 定义内存模组的写恢复时间，需要根据具体内存颗粒及运行频率进行配置。 |
| CONF_CTL_07[31:0] Offset: 0x70 | | | | DDR2 667：0x000f0a0b |
| ECC_C_ID | 27:24 | 0x0 | 0x0-0xf | 发生 1bit ECC 错的源 ID 号（只读） |
| CS_MAP | 19:16 | 0x0 | 0x0-0xf | 定义可用片选信号，本参数应当根据实际使用的片选个数进行正确的配置，不正确的配置将会导致错误的内存访问。该参数的四位分别对应于 CS0- CS3 |
| CASLAT_LIN_GATE | 11:8 | 0x0 | 0x0-0xf | 定义读命令返回数据时的 gate open 信号打开的时间，一般等于 CASLAT_LIN（以半个时钟周期为单位） |
| CASLAT_LIN | 3:0 | 0x0 | 0x0-0xf | 当板上走线延迟为 DDR2 时钟周期的<br>0.5~1.5 倍：CASLAT_LIN = CASLAT×2<br>小于 0.5 倍：CASLAT_LIN = CASLAT×2-1 |

| | | | | |
|---|---|---|---|---|
| | | | | 大于 1.5 倍：CASLAT_LIN = CASLAT×2+1<br>（以半个时钟周期为单位） |
| CONF_CTL_07[63:32] | Offset: 0x70 | | | DDR2 667：0x00000400 |
| MAX_ROW_REG | 59:56 | 0xf | 0x0-0xf | 系统最大行地址个数（只读） |
| MAX_COL_REG | 51:48 | 0xe | 0x0-0xe | 系统最大列地址个数（只读） |
| INITAREF | 43:40 | 0x0 | 0x0-0xf | 定义系统初始化时所需要执行的 autorefresh 命令个数 |
| ECC_U_ID | 35:32 | 0x0 | 0x0-0xf | 定义发生不可纠错的双字节错的源 ID 号（只读） |
| CONF_CTL_08[31:0] | Offset: 0x80 | | | DDR2 667：0x01020408 |
| ODT_RD_MAP_CS3 | 27:24 | 0x0 | 0x0-0xf | 定义 CS3 有读命令时，将指定的 CS 的 ODT 终端电阻有效，具体的配置应当参考相应的内存颗粒手册对于 ODT 配置的要求。该参数的四位分别对应于 CS0-CS3 |
| ODT_RD_MAP_CS2 | 19:16 | 0x0 | 0x0-0xf | 定义 CS2 有读命令时，将指定的 CS 的 ODT 终端电阻有效，具体的配置应当参考相应的内存颗粒手册对于 ODT 配置的要求。该参数的四位分别对应于 CS0-CS3 |
| ODT_RD_MAP_CS1 | 11:8 | 0x0 | 0x0-0xf | 定义 CS1 有读命令时，将指定的 CS 的 ODT 终端电阻有效，具体的配置应当参考相应的内存颗粒手册对于 ODT 配置的要求。该参数的四位分别对应于 CS0-CS3 |
| ODT_RD_MAP_CS0 | 3:0 | 0x0 | 0x0-0xf | 定义 CS0 有读命令时，将指定的 CS 的 ODT 终端电阻有效，具体的配置应当参考相应的内存颗粒手册对于 ODT 配置的要求。该参数的四位分别对应于 CS0-CS3 |
| CONF_CTL_08[63:32] | Offset: 0x80 | | | DDR2 667：0x01020408 |
| ODT_WR_MAP_CS3 | 59:56 | 0x0 | 0x0-0xf | 定义 CS3 有写命令时，将指定的 CS 的 ODT 终端电阻有效，具体的配置应当参考相应的内存颗粒手册对于 ODT 配置的要求。该参数的四位分别对应于 CS0-CS3 |
| ODT_WR_MAP_CS2 | 51:48 | 0x0 | 0x0-0xf | 定义 CS2 有写命令时，将指定的 CS 的 ODT 终端电阻有效，具体的配置应当参考相应的内存颗粒手册对于 ODT 配置的要求。该参数的四位分别对应于 CS0-CS3 |
| ODT_WR_MAP_CS1 | 43:40 | 0x0 | 0x0-0xf | 定义 CS1 有写命令时，将指定的 CS 的 ODT 终端电阻有效，具体的配置应当参考相应的内存颗粒手册对于 ODT 配置的要求。该参数的四位分别对应于 CS0-CS3 |
| ODT_WR_MAP_CS0 | 35:32 | 0x0 | 0x0-0xf | 定义 CS0 有写命令时，将指定的 CS 的 ODT 终端电阻有效，具体的配置应当参考相应的内存颗粒手册对于 ODT 配置的要求。该参数的四位分别对应于 CS0-CS3 |
| CONF_CTL_09[31:0] | Offset: 0x90 | | | DDR2 667：0x00000000 |
| PORT_DATA_ERROR_ID | 27:24 | 0x0 | 0x0-0xf | 端口上发生数据错误时的 ID 号（只读） |
| PORT_CMD_ERROR_TYPE | 19:16 | 0x0 | 0x0-0xf | 端口上发生命令错误的类型（只读）<br>位 0 – 数据位宽过大<br>位 1 – 关键字优先操作地址未对齐<br>位 2 – 关键字优先操作字数不是 2 幂<br>位 3 – narrow transform 出错 |
| PORT_CMD_ERROR_ID | 11:8 | 0x0 | 0x0-0xf | 端口上发生命令错误的 ID 号（只读） |
| OUT_OF_RANGE_SOURCE_ID | 3:0 | 0x0 | 0x0-0xf | 端口上发生越界访问错误时的 ID 号（只读） |
| CONF_CTL_09[63:32] | Offset: 0x90 | | | DDR2 667：0x0000060c |
| OCD_ADJUST_PUP_CS0 | 60:56 | 0x0 | 0x0-0x1f | 设置内存模组片选 0 OCD 上拉调整值。内存控制器将在初始化时根据这个参数的值向内存模组发出 OCD 调整命令 |
| OCD_ADJUST_PDN_CS0 | 52:48 | 0x0 | 0x0-0x1f | 设置内存模组片选 0 OCD 下拉调整值。内存控制器将在初始化时根据这个参数的值向内存模组发出 OCD |

| | | | | |
|---|---|---|---|---|
| | | | | 调整命令 |
| TRP | 43:40 | 0x0 | 0x0-0xf | 定义内存模组执行 pre-charge 所需要的时钟周期数，需要根据具体内存颗粒及运行频率进行配置。 |
| TDAL | 35:32 | 0x0 | 0x0-0xf | 当 auto-precharge 参数设置后，该参数定义了 auto-precharge 和 write recovery 时钟周期数。<br>TDAL = auto-precharge + write recovery<br>该参数仅在设置了 AP 之后才生效。 |
| CONF_CTL_10[31:0] | Offset: 0xa0 | | | DDR2 667：0x3f130200 |
| AGE_COUNT | 29:24 | 0x0 | 0x0-0x3f | 定义命令队列重排序逻辑使用 aging 算法时每个命令的 aging 初始值 |
| TRC | 20:16 | 0x0 | 0x0-0x1f | 定义对内存模组同一 bank 的 active 命令之间的时钟周期数，需要根据具体内存颗粒及运行频率进行配置。 |
| TMRD | 12:8 | 0x0 | 0x0-0x1f | 定义配置内存模组模式寄存器需要的时钟周期数，通常为 2 个周期 |
| TFAW | 4:0 | 0x0 | 0x0-0x1f | 定义内存模组 tFAW 参数，8 个逻辑 bank 时使用 |
| CONF_CTL_10[63:32] | Offset: 0xa0 | | | DDR2 667：0x1515153f |
| DLL_DQS_DELAY_2 | 62:56 | 0x0 | 0x0-0x7f | 定义读数据时 DQS2 的延迟百分比，每次增加一个时钟周期的 1/128，延迟 DQS 的目的是使延迟后的 DQS 可以落在数据信号的中心 |
| DLL_DQS_DELAY_1 | 54:48 | 0x0 | 0x0-0x7f | 定义读数据时 DQS1 的延迟百分比，每次增加一个时钟周期的 1/128，延迟 DQS 的目的是使延迟后的 DQS 可以落在数据信号的中心 |
| DLL_DQS_DELAY_0 | 46:40 | 0x0 | 0x0-0x7f | 定义读数据时 DQS0 的延迟百分比，每次增加一个时钟周期的 1/128，延迟 DQS 的目的是使延迟后的 DQS 可以落在数据信号的中心 |
| COMMAND_AGE_COUNT | 37:32 | 0x0 | 0x0-0x3f | 定义命令队列重排序逻辑使用 aging 算法时每个命令的 aging 初始值 |
| CONF_CTL_11[31:0] | Offset: 0xb0 | | | DDR2 667：0x15151515 |
| DLL_DQS_DELAY_6 | 30:24 | 0x0 | 0x0-0x7f | 定义读数据时 DQS6 的延迟百分比，每次增加一个时钟周期的 1/128，延迟 DQS 的目的是使延迟后的 DQS 可以落在数据信号的中心 |
| DLL_DQS_DELAY_5 | 22:16 | 0x0 | 0x0-0x7f | 定义读数据时 DQS5 的延迟百分比，每次增加一个时钟周期的 1/128，延迟 DQS 的目的是使延迟后的 DQS 可以落在数据信号的中心 |
| DLL_DQS_DELAY_4 | 14:8 | 0x0 | 0x0-0x7f | 定义读数据时 DQS4 的延迟百分比，每次增加一个时钟周期的 1/128，延迟 DQS 的目的是使延迟后的 DQS 可以落在数据信号的中心 |
| DLL_DQS_DELAY_3 | 6:0 | 0x0 | 0x0-0x7f | 定义读数据时 DQS3 的延迟百分比，每次增加一个时钟周期的 1/128，延迟 DQS 的目的是使延迟后的 DQS 可以落在数据信号的中心 |
| CONF_CTL_11[63:32] | Offset: 0xb0 | | | DDR2 667：0x5f7f1515 |
| WR_DQS_SHIFT | 62:56 | 0x0 | 0x0-0x7f | 定义写数据时 clk_wr 的延迟百分比，每次增加一个时钟周期的 1/128，延迟 CLK_WR 的目的是使发出的数据时钟提前四分之一个周期，从而比 DQS 提前四分之一个周期。 |
| DQS_OUT_SHIFT | 54:48 | 0x0 | 0x0-0x7f | 定义写数据时 DQS 的延迟百分比，每次增加一个时钟周期的 1/128，延迟 DQS 的目的是使发出的 DQS 与时钟对齐。从而可以落在数据的中心 |
| DLL_DQS_DELAY_8 | 46:40 | 0x0 | 0x0-0x7f | 定义读数据时 DQS8 的延迟百分比，每次增加一个时钟周期的 1/128，延迟 DQS 的目的是使延迟后的 DQS 可以落在数据信号的中心 |
| DLL_DQS_DELAY_7 | 38:32 | 0x0 | 0x0-0x7f | 定义读数据时 DQS7 的延迟百分比，每次增加一个时钟周期的 1/128，延迟 DQS 的目的是使延迟后的 DQS 可以落在数据信号的中心 |
| CONF_CTL_12[31:0] | Offset: 0xc0 | | | DDR2 667：0x15000000 |
| TRAS_MIN | 31:24 | 0x0 | 0x0-0xff | 定义内存模组行地址有效命令的最小时钟周期数 |

| | | | | |
|---|---|---|---|---|
| OUT_OF_RANGE_LENGTH | 23:16 | 0x0 | 0x0-0xff | 发生越界访问时的命令长度（只读） |
| ECC_U_SYND | 15:8 | 0x0 | 0x0-0xff | 发生 2bit 不可纠错误时的原因（只读） |
| ECC_C_SYND | 7:0 | 0x0 | 0x0-0xff | 发生 1bit 可纠错错误时的原因（只读） |
| CONF_CTL_12[63:32] | Offset: 0xc0 | | | DDR2 667：0x002a3c05 |
| DLL_DQS_DELAY_BYPASS_0 | 56:48 | 0x0 | 0x0-0x1ff | 定义 DLL bypass 模式下 dqs0 延迟线的个数 |
| TRFC | 47:40 | 0x0 | 0x0-0xff | 定义内存模组刷新操作需要的时钟周期数，需要根据具体内存颗粒及运行频率进行配置。 |
| TRCD_INT | 39:32 | 0x0 | 0x0-0xff | 定义内存模组 RAS 到 CAS 之间的时钟周期数，需要根据具体内存颗粒及运行频率进行配置。 |
| CONF_CTL_13[31:0] | Offset: 0xd0 | | | DDR2 667：0x002a002a |
| DLL_DQS_DELAY_BYPASS_2 | 24:16 | 0x0 | 0x0-0x1 | 定义 DLL bypass 模式下 dqs2 延迟线的个数 |
| DLL_DQS_DELAY_BYPASS_1 | 8:0 | 0x0 | 0x0-0x1 | 定义 DLL bypass 模式下 dqs1 延迟线的个数 |
| CONF_CTL_13[63:32] | Offset: 0xd0 | | | DDR2 667：0x002a002a |
| DLL_DQS_DELAY_BYPASS_4 | 56:48 | 0x0 | 0x0-0x1ff | 定义 DLL bypass 模式下 dqs4 延迟线的个数 |
| DLL_DQS_DELAY_BYPASS_3 | 40:32 | 0x0 | 0x0-0x1ff | 定义 DLL bypass 模式下 dqs3 延迟线的个数 |
| CONF_CTL_14[31:0] | Offset: 0xe0 | | | DDR2 667：0x002a002a |
| DLL_DQS_DELAY_BYPASS_6 | 24:16 | 0x0 | 0x0-0x1ff | 定义 DLL bypass 模式下 dqs6 延迟线的个数 |
| DLL_DQS_DELAY_BYPASS_5 | 8:0 | 0x0 | 0x0-0x1ff | 定义 DLL bypass 模式下 dqs5 延迟线的个数 |
| CONF_CTL_14[63:32] | Offset: 0xe0 | | | DDR2 667：0x002a002a |
| DLL_DQS_DELAY_BYPASS_8 | 56:48 | 0x0 | 0x0-0x1ff | 定义 DLL bypass 模式下 dqs8 延迟线的个数 |
| DLL_DQS_DELAY_BYPASS_7 | 40:32 | 0x0 | 0x0-0x1ff | 定义 DLL bypass 模式下 dqs7 延迟线的个数 |
| CONF_CTL_15[31:0] | Offset: 0xf0 | | | DDR2 667：0x00000004 |
| DLL_LOCK | 24:16 | 0x0 | 0x0-0x1ff | 指示 DLL 锁定时，延迟整个时钟周期所使用的延迟单元个数（只读） |
| DLL_INCREMENT | 8:0 | 0x0 | 0x0-0x1ff | 定义 DLL 进行锁定时，每次调整增加的延迟单元个数，不应为零，设小精度会比较高 |
| CONF_CTL_15[63:32] | Offset: 0xf0 | | | DDR2 667：0x00b4000a |
| DQS_OUT_SHIFT_BYPASS | 56:48 | 0x0 | 0x0-0x1ff | 定义 dqs out bypass 模式下 wr_dqs 延迟单元数 |
| DLL_START_POINT | 40:32 | 0x0 | 0x0-0x1ff | 定义 DLL 进行锁定时，延迟单元起始个数， 不应小于 4，不应过大，否则会使访存出错。 |
| CONF_CTL_16[31:0] | Offset: 0x100 | | | DDR2 667：0x00000087 |
| INT_ACK | 25:16 | 0x0 | 0x0-0x3ff | 将参数中位设置为 1 时，将会使对应位的清中断 |
| WR_DQS_SHIFT_BYPASS | 8:0 | 0x0 | 0x0-0x1ff | 定义 wr dqs bypass 模式下 wr_clk 延迟单元数 |
| CONF_CTL_16[63:32] | Offset: 0x100 | | | DDR2 667：0x00000000 |
| INT_STATUS | 58:48 | 0x0 | 0x0-0x7ff | 内存控制器发生中断的原因（只读）<br>位：0 – 一条访存指令地址超出内存实际物理空间<br>位：1 – 多条访存指令地址超出内存实际物理空间<br>位：2 – 一次 ECC 一位错发生<br>位：3 – 多次 ECC 一位错发生<br>位：4 – 一次 ECC 两位错发生<br>位：5 – 多次 ECC 两位错发生<br>位：6 – 控制器地址通道发生错误<br>位：7 – 控制器数据通道发生错误<br>位：8 – 内存初始化完成<br>位：9 – DLL 未锁定<br>位：10 – 以上任何一种中断发生 |
| INT_MASK | 42:32 | 0x0 | 0x0-0x7ff | 内存控制器中断的掩码位，与 INT_STATUS 位一一对应 |
| CONF_CTL_17[31:0] | Offset: 0x110 | | | DDR2 667：0x0000181b |
| EMRS1_DATA | 30:16 | 0x0 | 0x0-0x7ff | 定义内存控制器初始化内存模组时写入到内存模组 EMRS1 寄存器中的数据 |
| TREF | 13:0 | 0x0 | 0x0-0x3ff | 定义内存模组两次刷新命令的时钟间隔，需要根据具体内存颗粒及运行频率进行配置。 |
| CONF_CTL_17[63:32] | Offset: 0x110 | | | DDR2 667：0x00000000 |

| | | | | |
|---|---|---|---|---|
| EMRS2_DATA_1 | 62:48 | 0x0000 | 0x0-0x7fff | 定义内存模组初始化时，片选 1 对应的 EMRS2 数据 |
| EMRS2_DATA_0 | 46:32 | 0x0000 | 0x0-0x7fff | 定义内存模组初始化时，片选 0 对应的 EMRS2 数据 |
| CONF_CTL_18[31:0] | Offset: 0x120 | | | DDR2 667：0x00000000 |
| EMRS2_DATA_3 | 30:16 | 0x0000 | 0x0-0x7fff | 定义内存模组初始化时，片选 3 对应的 EMRS2 数据 |
| EMRS2_DATA_2 | 14:0 | 0x0000 | 0x0-0x7fff | 定义内存模组初始化时，片选 2 对应的 EMRS2 数据 |
| CONF_CTL_18[63:32] | Offset: 0x120 | | | DDR2 667：0x001c0000 |
| AXI0_EN_LT_WIDTH_INSTR | 63:48 | 0x0000 | 0x0-0xffff | 定义 AXI0 端口是否接收小于 64 位位宽的内存访问 |
| EMRS3_DATA | 46:32 | 0x0000 | 0x0-0x7fff | 定义内存模组初始化时 EMRS3 对应的数据 |
| CONF_CTL_19[31:0] | Offset: 0x130 | | | DDR2 667：0x00c8006b |
| TDLL | 31:16 | 0x0000 | 0x0-0xffff | 定义内存模组 DLL 锁定需要的时钟周期数 |
| TCPD | 15:0 | 0x0000 | 0x0-0xffff | 定义内存模组时钟有效到 precharge 之间的时钟周期数，需要根据具体内存颗粒及运行频率进行配置。 |
| CONF_CTL_19[63:32] | Offset: 0x130 | | | DDR2 667：0x48e10002 |
| TRAS_MAX | 63:48 | 0x0000 | 0x0-0xffff | 定义内存模组行有效命令的最大时钟周期数，需要根据具体内存颗粒及运行频率进行配置。 |
| TPDEX | 47:32 | 0x0000 | 0x0-0xffff | 定义内存模组掉电退出命令的时钟周期数 |
| CONF_CTL_20[31:0] | Offset: 0x140 | | | DDR2 667：0x00c8002f |
| TXSR | 31:16 | 0x0000 | 0x0-0xffff | 定义内存模组自刷新退出需要的时钟周期数 |
| TXSNR | 15:0 | 0x0000 | 0x0-0xffff | 定义内存模组 tXSNR 参数 |
| CONF_CTL_20[63:32] | Offset: 0x140 | | | DDR2 667：0x00000000 |
| XOR_CHECK_BITS | 63:48 | 0x0000 | 0x0-0xffff | 当 fwc 参数设定时，下次写操作的 check bit 将会与该参数进行异或后写入内存 |
| VERSION | 47:32 | 0x2041 | 0x2041 | 定义内存控制器版本号（只读） |
| CONF_CTL_21[31:0] | Offset: 0x150 | | | DDR2 667：0x00030d40 |
| ECC_C_ADDR[7:0] | 31:24 | 0x0000 | 0x0-0x1fffffff | 记录发生 1bit ECC 错误时的地址信息（只读） |
| TINIT | 23:0 | 0x0000 | 0x0-0xfffff | 定义内存模组初始化需要的时钟周期数，需要根据具体内存颗粒及运行频率进行配置。一般为 200us |
| CONF_CTL_21[63:32] | Offset: 0x150 | | | DDR2 667：0x00000000 |
| ECC_C_ADDR[36:8] | 60:32 | 0x0 | 0x0-0x1fffffff | 记录发生 1bit ECC 错误时的地址信息（只读） |
| CONF_CTL_22[31:0] | Offset: 0x160 | | | DDR2 667：0x00000000 |
| ECC_U_ADDR[31:0] | 31:0 | 0x0 | 0x0-0x1fffffff | 记录发生 2bit ECC 错误时的地址信息（只读） |
| CONF_CTL_22[63:32] | Offset: 0x160 | | | DDR2 667：0x00000000 |
| ECC_U_ADDR[36:32] | 36:32 | 0x0 | 0x0-0x1fffffff | 记录发生 2bit ECC 错误时的地址信息（只读） |
| CONF_CTL_23[31:0] | Offset: 0x170 | | | DDR2 667：0x00000000 |
| OUT_OF_RANGE_ADDR[31:0] | 31:0 | 0x0 | 0x0-0x1fffffff | 记录发生越界访问时的地址信息（只读） |
| CONF_CTL_23[63:32] | Offset: 0x170 | | | DDR2 667：0x00000000 |
| OUT_OF_RANGE_ADDR[36:32] | 36:32 | 0x0 | 0x0-0x1fffffff | 记录发生越界访问时的地址信息（只读） |
| CONF_CTL_24[31:0] | Offset: 0x180 | | | DDR2 667：0x00000000 |
| PORT_CMD_ERROR_ADDR[31:0] | 31:0 | 0x0 | 0x0-0x1fffffff | 记录端口发生命令错误时的地址信息（只读） |
| CONF_CTL_24[63:32] | Offset: 0x180 | | | DDR2 667：0x00000000 |
| PORT_CMD_ERROR_ADDR[36:32] | 36:32 | 0x0 | 0x0-0x1fffffff | 记录端口发生命令错误时的地址信息（只读） |
| CONF_CTL_25[31:0] | Offset: 0x190 | | | DDR2 667：0x00000000 |
| ECC_C_DATA[31:0] | 31:0 | 0x0 | 0x0-0x1fffffff | 记录发生 1bit ECC 错误时的数据信息（只读） |
| CONF_CTL_25[63:32] | Offset: 0x190 | | | DDR2 667：0x00000000 |
| ECC_C_DATA[63:32] | 63:32 | 0x0 | 0x0-0x1fffffff | 记录发生 1bit ECC 错误时的数据信息（只读） |

| CONF_CTL_26[31:0] | Offset: 0x1a0 | | | DDR2 667：0x00000000 |
|---|---|---|---|---|
| ECC_U_DATA[31:0] | 31:0 | 0x0 | 0x0-0x1ffffffff | 记录发生 2bit ECC 错误时的数据信息（只读） |
| CONF_CTL_26[63:32] | Offset: 0x1a0 | | | DDR2 667：0x00000000 |
| ECC_U_DATA[63:32] | 63:32 | 0x0 | 0x0-0x1ffffffff | 记录发生 2bit ECC 错误时的数据信息（只读） |
| CONF_CTL_27[31:0] | Offset: 0x1b0 | | | DDR2 667：0x00000000 |
| CKE_DELAY | 2:0 | 0x0 | 0x0-0x7 | CKE 有效延迟 |
| CONF_CTL_28[31:0] | Offset: 0x1c0 | | | DDR2 667：0x00000001 |
| UB_DIMM | 0:0 | 0x0 | 0x0-0x1 | 是否采用 unbuffered 内存条，对于普通内存条，应当置为 1，对于笔记本内存或是直接使用内存颗粒，应当置为 0，否则不能正常工作 |

对上述配置寄存器中一些位的说明如下：

(1) CONF_CTL_00 AP

此参数用于控制是否启用 Autoprecharge 功能，一旦启用 Autoprecharge，内存将在每条读写指令后关闭所操作页。在发送大量连续地址操作的时候如果打开此参数会导致性能的下降。

(2) CONF_CTL_00 CONCURRENTAP

此参数用于设置内存是否支持 Concurrent Autoprecharge，需要注意的是很多厂商的内存颗粒并不支持这种方式。

(3) CONF_CTL_03 SREFRESH

此参数用于设置内存进入 Self Refresh 工作方式。需要从 Self Refresh 方式返回时必须使这个参数置 0。

(4) CONF_CTL_07 CASLAT_LIN_GATE

此参数用于控制读数据返回时内存控制器对数据采样的时机，一般等于 CASLAT_LIN。根据主板走线带来的时钟信号和 DQS 信号间的偏差可加减 1。

(5) CONF_CTL_15 DLL_INCREMENT

此参数不应设为 0。

(6) CONF_CTL_15 DLL_START_POINT

此参数不应设为 0 或 1。且应小于锁定时得到正确 DLL_LOCK_VALUE 的 1.5 倍。

(7) CONF_CTL_28 UB_DIMM

此参数在使用 Unbuffered 内存条时，需要设置为 1，在直接使用内存颗粒时需要设置为 0。

## 5.5 DDR2 SDRAM 采样模式配置

在龙芯 1A 的 DDR2 SDRAM 控制器中，并通过延迟补偿电路（使用 DLL）来采样返回 DQS 的数据。因为内存控制器和 SDRAM 模块间的数据返回路径有延迟，所以必须引进一组控制信号用来测量延迟。

DDR2_GATE_I[1:0] 和 DDR2_GATE_O[1:0]的控制信号用于延迟测量。在 PCB 设计中，DDR2_GATE_I 和 DDR2_GATE_O 连接起来模拟 PCB 上的写延迟。这样，采样的精确性能够被保证。

## 5.6 DDR2 SDRAM PAD 驱动配置

表 5-2 DDR2 SDRAM PAD 驱动控制

| 寄存器地址 | 位 | PAD 控制位 | 对应被控制的 PAD |
|---|---|---|---|

| | | | |
|---|---|---|---|
| 0x1fd010c8 | [27:26] | DDR2_ssel[1:0] | DDR-CK/CONTRL/ADDR |
| | [29:28] | DDR2_ssel[3:2] | DQ[31:0] |
| | [31:30] | DDR2_ssel[5:4] | DQS[3:0] |
| 0x1fd010F8 | [27:26] | DDR2_tsel[1:0] | DQ[31:0] |
| | [29:28] | DDR2_tsel[3:2] | DQS[3:0] |
| | [30] | DDR2_st[0] | All except CK |

## 5.7 DDR2 16 位工作模式配置

DDR2 支持 16 位数据宽度的外部接口，配置工作如下：

1) 初始化 DDR 控制器，使用与 32 位模式相同的参数；

2) DISABLE_DDR_CONFSPACE 位置 1，关闭 DDR 控制器配置空间；

3) DDR32TO16EN 位置 1，使能 DDR16 位模式；

4) DDR2 16 位数据宽度模式正常使用。

表 5-3 DDR2 SDRAM 16 位数据宽度配置

| 寄存器地址 | 位 | 配置描述 |
|---|---|---|
| 0x1fd0_0420 | [1] | DISABLE_DDR_CONFSPACE |
| 0x1fd0_0420 | [0] | DDR32TO16EN |

# 6 PCI

龙芯 1A 的 PCI 控制器可以运行在 PCI 或者 PCIX 模式，其实现符合 PCI 2.3 和 PCI-X 1.0b 规范。龙芯 1A 在 PCI 总线上可以充当两种不同的角色：PCI 主桥（PCI host bridge）或者 PCI 设备（PCI device）。当龙芯 1A 作为 PCI 主桥时，龙芯 1A 可以在 PCI 总线上发起设备配置读/写、IO 读/写、MEM 读/写；也可以接收来自设备的 MEM 读/写请求。当龙芯 1A 作为 PCI 设备时，龙芯 1A 可以在 PCI 总线上向 PCI 主桥发起 MEM 读/写；也可以接收来自 PCI 主桥的设备配置读/写、IO 读/写、MEM 读/写。

## 6.1 总体描述

龙芯 1A 作为 PCI 主桥时，龙芯 1A 访问 PCI 空间的地址划分如表 6-1 所示。当龙芯 1A 作为 PCI 设备工作时，龙芯 1A 对。

表 6-1 中地址列表以外任何地址的访问将直接（不做任何地址转换）被发送到 PCI 总线上。

表 6-1龙芯1A访问PCI总线的地址空间划分

| 起始地址 | 大小 | 属性 |
|---|---|---|
| 0x10000000 | 64MB | PCI mem space 0 |
| 0x14000000 | 64MB | PCI mem space 1 |
| 0x18000000 | 64MB | PCI mem space 2 |
| 0x1c000000 | 1MB | PCI IO 空间 |
| 0x1c100000 | 64KB | PCI 设备配置空间 |
| 0x1c110000 | 256B | PCI 控制器的 pciheader 空间 |

当龙芯 1A 访问 3 个 64MB 的 PCI mem 空间时，PCI 总线上的地址由寄存器 pcimap[5:0]、pcimap[11:6]、pcimap[17:12]分别形成 PCI mem space 0、1、2 这 3 个 PCI mem 地址窗口的高 6 位；当龙芯 1A 访问 PCI 的 IO 空间时，PCI 总线上地址的高 12 位被置为全 0；龙芯 1A 在发起配置空间读写前，应用程序应先配置好 17 位的 pcimap_cfg 寄存器，告诉控制器欲发起的配置操作的类型和高 16 位地址线上的值。然后对 0x1c100000 开始的 64K 空间进行读写即可访问对应设备的配置头。设备号由根据 pcimap_cfg[15:0]从低到高优先编码得到。PCI 设备配置操作地址生成见图 6-1。

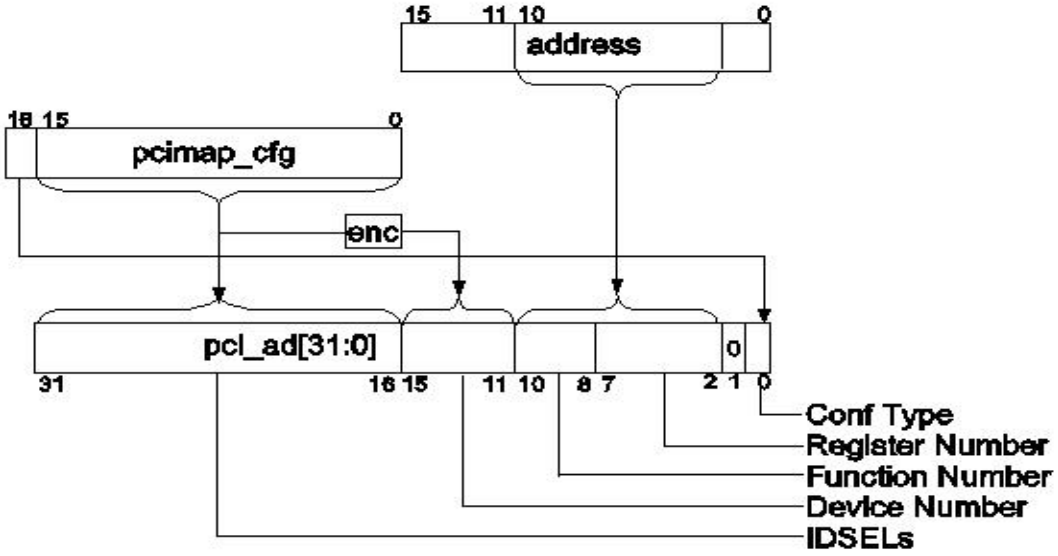



图 6-1　PCI 设备配置读写总线地址生成机制

```
#define pcibus_cfg_base    0xbc100000
#define pcimapcfg          0x1fd01120

unsigned int base0[16];

for(int i = 0; i< 16;i++)
{
      *((volatile unsigned int *)(pcimapcfg))= 0x1<< (16+i);
      base0[i] = *((volatile unsigned int *)(pcibus_cfg_base+0x10));
}
```

上面这段代码将读取 PCI 总线 0 上所有 PCI 设备的 base0 到数组 base0 中；

```
unsigned int devnum;
unsigned int funcnum = 0x0;
unsigned int busnum = 0x1;
for(devnum = 0; devnum<32; devnum++)
{
      *((volatile unsigned int *)(pcimapcfg))= 0x10000 | busnum;
      base0[i] = *((volatile unsigned int *)(pcibus_cfg_base+ (devnum << 11) +
(funcnum << 8)+ 0x10));
}
```

上面这段代码将读取 PCI 总线 1 上所有 PCI 设备的 base0 到数组 base0 中。

龙芯 1A 的 PCI 控制器的 pciheader 空间如

表 6-2 所示。

表 6-2龙芯1A的PCI控制器pciheader空间划分

| 字节 3 | 字节 2 | 字节 1 | 字节 0 | 地址 |
|---|---|---|---|---|
| Device ID | | Vendor ID | | 00 |
| Status | | Command | | 04 |

| | | | | |
|---|---|---|---|---|
| Class Code | | | Revision ID | 08 |
| BIST | Header Type | Latency Timer | CacheLine Size | 0C |
| Base Address Register 0 | | | | 10 |
| Base Address Register 1 | | | | 14 |
| Base Address Register 2 | | | | 18 |
| Base Address Register 3 | | | | 1C |
| Base Address Register 4 | | | | 20 |
| Base Address Register 5 | | | | 24 |
| | | | | 28 |
| Subsystem ID | | Subsystem Vendor ID | | 2C |
| | | | | 30 |
| | | | Capabilities Pointer | 34 |
| | | | | 38 |
| Maximum Latency | Minimum Grant | Interrupt Pin | Interrupt Line | 3C |
| ISR_40h | | | | 40 |
| ISR_44h | | | | 44 |
| ISR_48h | | | | 48 |
| ISR_4Ch | | | | 4C |
| ISR_50h | | | | 50 |
| ISR_54h | | | | 54 |
| ISR_58h | | | | 58 |
| PCIX Command Register | | | | E0 |
| PCIX Status Register | | | | E4 |

其中 00～3C 为 PCI type 0 header，40 开始的寄存器为龙芯 1A 自定义寄存器。龙芯 1A 的 pciheader 空间可以由龙芯 1A 以 0x1c110000 为基址进行访问，也可以在龙芯 1A 充当 PCI 设备时由外部的 PCI 主桥通过 PCI 设备配置读/写操作进行访问。

在龙芯 1A 的 pciheader 空间中有下列寄存器的使用方法和标准的 PCI 协议有所差异

## 6.2 寄存器描述

Command 寄存器：

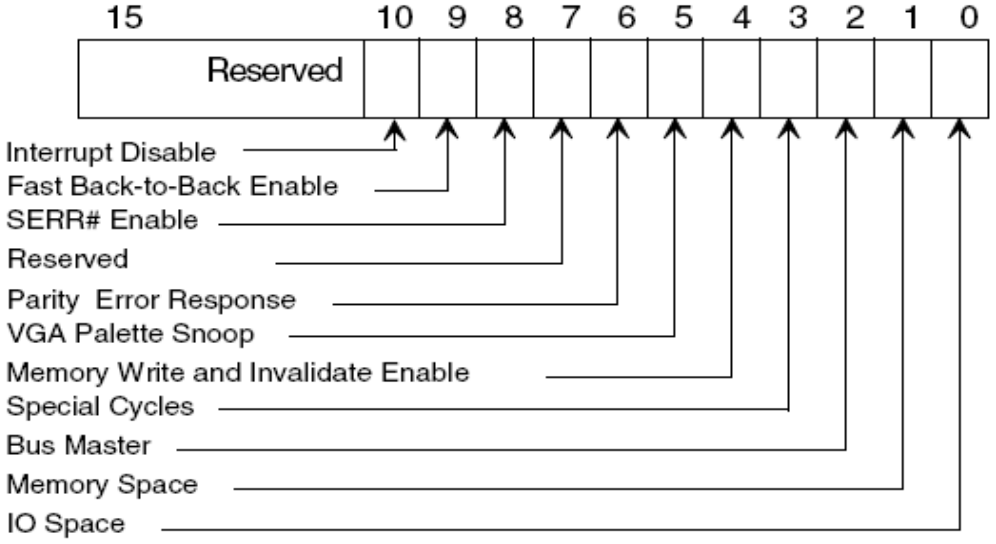


图 6-2 Command register layout

Command 寄存器的格式如图 6-2 所示。在龙芯 1A 中，Command 寄存器只有 0、1、2、6、8 这 5 位可以进行修改。龙芯 1A 的 PCI 控制器不支持 Fast Back-to-Back 功能。龙芯 1A 在使用 PCI 总线前必须将 Command 寄存器的 0、1、2 这 3 位值为有效（置为 1）。

Base Address Register 0～5：

龙芯 1A 有 3 个用于将龙芯 1A 所控制资源映射到 PCI 总线上的 64 位地址窗口。这 3 个 PCI 地址窗口分别被命名为 PCI base 0、1、2。这 3 个窗口的大小和映射的龙芯 1A 内部资源如。

表 6-3 所示。

表 6-3 龙芯1A内部资源在PCI总线上的映射

| 窗口名称 | 大小 | 访问方式 | 对应资源 |
|---|---|---|---|
| PCI base 0 | 2MB | MEM | GPU/DC |
| PCI base 1 | 16MB | MEM | AXI MUX Slave |
| PCI base 2 | 1GB | MEM | DDR |

PCI base 0 由 Base Address Register 1 和 Base Address Register 0 分别构成地址窗口的高 32 位和低 32 位。

PCI base 1 由 Base Address Register 3 和 Base Address Register 2 分别构成地址窗口的高 32 位和低 32 位。

PCI base 2 由 Base Address Register 5 和 Base Address Register 4 分别构成地址窗口的高 32 位和低 32 位。

Interrupt Line 和 Interrupt Pin：

由于龙芯 1A 的 PCI 控制器不包含相应的中断控制部分，这两个和中断相关的寄存器在龙芯 1A 中无意义。

在 40～E4 这些寄存器中，为了保证龙芯 1A 的 PCI 接口运行正常，必须对 ISR_4Ch 寄存器的第 31 位进行设置（ISR_4Ch 寄存器的第 31 位需要被置为 1），其它的寄存器可以不必进行如何配置。40～E4 这些寄存器的定义如下：

| 位域 | 字段名 | 访问 | 复位值 | 说明 |
|---|---|---|---|---|
| ISR_40 | | | | |
| 31 | tar_read_io | 读写<br>(写 1 清) | 0 | target 端收到对 IO 或者是不可预取区域的访问 |
| 30 | tar_read_discard | 读写<br>(写 1 清) | 0 | target 端的 delay 请求被丢弃 |
| 29 | tar_resp_delay | 读写 | 0 | target 访问何时给出 delay/split<br>0: 超时后 |

| | | | | |
|---|---|---|---|---|
| | | | | 1: 马上 |
| 28 | tar_delay_retry | 读写 | 0 | target 访问重试策略<br>0: 根据内部逻辑（见 29 位）<br>1: 马上重试 |
| 27 | tar_read_abort_en | 读写 | 0 | 若 target 对内部的读请求超时，是否让以 target-abort 回应 |
| 26:25 | Reserved | - | 0 | |
| 24 | tar_write_abort_en | 读写 | 0 | 若 target 对内部的写请求超时，是否让以 target-abort 回应 |
| 23 | tar_master_abort | 读写 | 0 | 是否允许 master-abort |
| 22:20 | tar_subseq_timeout | 读写 | 000 | target 后续延迟超时<br>000: 8 周期<br>其它: 不支持 |
| 19:16 | tar_init_timeout | 读写 | 0000 | target 初始延迟超时<br>PCI 模式下<br>0: 16 周期<br>1-7: 禁用计数器<br>8-15: 8-15 周期<br>PCIX 模式下超时计数固定为 8 周期，此处配置影响最大的 delay 访问数<br>0: 8 delay 访问<br>8: 1 delay 访问<br>9: 2 delay 访问<br>10: 3 delay 访问<br>11: 4 delay 访问<br>12: 5 delay 访问<br>13: 6 delay 访问<br>14: 7 delay 访问<br>15: 8 delay 访问 |
| 15:4 | tar_pref_boundary | 读写 | 000h | 可预取边界配置（以 16 字节为单位）<br>FFF: 64KB 到 16byte<br>FFE: 64KB 到 32byte<br>FF8: 64KB 到 128byte |
| 3 | tar_pref_bound_en | 读写 | 0 | 使用 tar_pref_boundary 的配置<br>0: 预取到设备边界<br>1: 使用 tar_pref_boundary |
| 2 | Reserved | - | 0 | |
| 1 | tar_splitw_ctrl | 读写 | 0 | target split 写控制<br>0: 阻挡除 Posted Memory Write 以外的访问<br>1: 阻挡所有访问，直至 split 完成 |
| 0 | mas_lat_timeout | 读写 | 0 | 禁用 mater 访问超时<br>0: 允许 master 访问超时<br>1: 不允许 |
| ISR_44 | | | | |
| 31:0 | Reserved | - | - | |
| ISR_48 | | | | |
| 31:0 | tar_pending_seq | 读写 | 0 | target 未处理完的请求号位向量<br>对应位写 1 可清 |
| ISR_4C | | | | |

| | | | | |
|---|---|---|---|---|
| 31:30 | Reserved | - | - | |
| 29 | mas_write_defer | 读写 | 0 | 允许后续的读越过前面未完成的写<br>(只对 PCI 有效) |
| 28 | mas_read_defer | 读写 | 0 | 允许后续的读写越过前面未完成的读<br>(只对 PCI 有效) |
| 27 | mas_io_defer_cnt | 读写 | 0 | 在外的最大 IO 请求数<br>0: 由控制<br>1: 1 |
| 26:24 | mas_read_defer_cnt | 读写 | 010 | master 支持在外读的最大数(只对 PCI 有效）<br>0: 8<br>1-7: 1-7<br>注：一个双地址周期访问占两项 |
| 23:16 | err_seq_id | 只读 | 00h | target/master 错误号 |
| 15 | err_type | 只读 | 0 | target/master 出错的命令类型<br>0: |
| 14 | err_module | 只读 | 0 | 出错的模块<br>0: target<br>1: master |
| 13 | system_error | 读写 | 0 | target/master 系统错（写 1 清） |
| 12 | data_parity_error | 读写 | 0 | target/master 数据奇偶错（写 1 清） |
| 11 | ctrl_parity_error | 读写 | 0 | target/master 地址奇偶错（写 1 清） |
| 10:0 | Reserved | - | - | |
| ISR_50 | | | | |
| 31:0 | mas_pending_seq | 读写 | 0 | master 未处理完的请求号位向量<br>对应位写 1 可清 |
| ISR_54 | | | | |
| 31:0 | mas_split_err | 读写 | 0 | split 返回出错的请求号位向量 |
| ISR_58 | | | | |
| 31:30 | Reserved | - | - | |
| 29:28 | tar_split_priority | 读写 | 0 | target split 返回优先级<br>0 最高, 3 最低 |
| 27:26 | mas_req_priority | 读写 | 0 | master 对外的优先级<br>0 最高, 3 最低 |
| 25 | Priority_en | 读写 | 0 | 仲裁算法(在 master 的访问和 target 的 split 返回间做仲裁）<br>0: 固定优先级<br>1: 轮转 |
| 24:18 | 保留 | - | - | |
| 17 | mas_retry_aborted | 读写 | 0 | master 重试取消（写 1 清） |
| 16 | mas_trdy_timeout | 读写 | 0 | master TRDY 超时计数 |
| 15:8 | mas_retry_value | 读写 | 00h | master 重试次数<br>0: 无限重试<br>1-255: 1-255 次 |
| 7:0 | mas_trdy_count | 读写 | 00h | master TRDY 超时计数器<br>0: 禁用<br>1-255: 1-255 拍 |

为了保证龙芯 1A 的 PCI 控制器正常工作，龙芯 1A 还有下列寄存器需要进

行相应的配置：

| | |
|---|---|
| pcimap | PCI 映射 |
| PCIX_Bridge_Cfg | PCI/X 桥相关配置 |
| pcimap_cfg | PCI 配置读写设备地址 |
| PCI_Hit0_Sel_L | PCI 窗口 0 控制低 32 位 |
| PCI_Hit0_Sel_H | PCI 窗口 0 控制高 32 位 |
| PCI_Hit1_Sel_L | PCI 窗口 1 控制低 32 位 |
| PCI_Hit1_Sel_H | PCI 窗口 1 控制高 32 位 |
| PCI_Hit2_Sel_L | PCI 窗口 2 控制低 32 位 |
| PCI_Hit2_Sel_H | PCI 窗口 2 控制高 32 位 |
| PXArb_Config | PCIX 仲裁器配置 |
| PXArb_Status | PCIX 仲裁器状态 |
| Pciconfigi | PCI 桥工作模式配置 |

PCI_Hit0_Sel_H 和 PCI_Hit0_Sel_L 分别构成龙芯 1A 的 PCI 窗口 0 的基址的高 32 位和低 32 位。这个地址窗口为一个 2MB 大小的地址窗口。其映射的资源是龙芯 1A 的 GPU/DC 资源。当龙芯 1A 作为南桥（PCI 设备）工作时，这两个 32 位寄存器的值分别应该被配置为：32'h7fff_ffff 、32'hffe0_0004。当龙芯 1A 作为 Soc（PCI 主桥）工作时 PCI 窗口 0 应该被关闭，这两个寄存器的值分别应该被设置为：32’h6、32’h6。

PCI_Hit1_Sel_H 和 PCI_Hit1_Sel_L 分别构成龙芯 1A 的 PCI 窗口 1 的基址的高 32 位和低 32 位。这个地址窗口为一个 16MB 大小的地址窗口。其映射的资源是龙芯 1A 中除 GPU/DC 外的其它 IP 资源。当龙芯 1A 作为南桥（PCI 设备）工作时，这两个 32 位寄存器的值分别应该被配置为：32'h7fff_ffff 、32'hff00_0004。当龙芯 1A 作为 Soc（PCI 主桥）工作时 PCI 窗口 1 应该被关闭，这两个寄存器的值分别应该被设置为：32’h6、32’h6。

PCI_Hit2_Sel_H 和 PCI_Hit2_Sel_L 分别构成龙芯 1A 的 PCI 窗口 2 的基址的高 32 位和低 32 位。这个地址窗口为一个大小可变的地址窗口（必须为 2 的幂次）。其映射的资源是龙芯 1A 中 DDR 内存空间。当龙芯 1A 映射到 PCI 总线上的 DDR 内存空间为 1GB 大小时，这两个 32 位寄存器的值分别应该被配置为：32'hffff_ffff 、32'hc000_0004。

| 偏移地址 | 位宽 | 寄存器 | 描述 | 读写特性 |
|---|---|---|---|---|
| 0x1fd0_110C | 32 | PCI_PXARB_CONFIG | 仲裁器配置 | R/W |
| 0x1fd0_1110 | 32 | PCI_PXARB_STATUS | 仲裁器状态 | R/W |
| 0x1fd0_1114 | 32 | pcimap | pcimap[5:0] 、 | R/W |

| | | | | |
|---|---|---|---|---|
| | | | pcimap[11:6] 、pcimap[17:12]分别形成 PCI mem space 0、1、2 这 3 个 PCI mem 地址窗口的高 6 位 | |
| 0x1fd0_1118 | 32 | PCIX_RGATE | 应该设置为 6'h18 | R/W |
| 0x1fd0_111C | 32 | PCIX_RELAX_EN | 应该设置为 0 | R/W |
| 0x1fd0_1120 | 32 | pcimap_cfg | 详见图 6-1 | R/W |
| 0x1fd0_1130 | 32 | PCI_Hit0_Sel_L | | R/W |
| 0x1fd0_1134 | 32 | PCI_Hit0_Sel_H | | R/W |
| 0x1fd0_1138 | 32 | PCI_Hit1_Sel_L | | R/W |
| 0x1fd0_1140 | 32 | PCI_Hit1_Sel_H | | R/W |
| 0x1fd0_1144 | 32 | PCI_Hit2_Sel_L | | R/W |
| 0x1fd0_1148 | 32 | PCI_Hit2_Sel_H | | R/W |

| pciap | | | | |
|---|---|---|---|---|
| 5:0 | trans_lo0 | 读写 | 0 | PCI_Mem_Lo0 窗口映射地址高 6 位 |
| 11:6 | trans_lo1 | 读写 | 0 | PCI_Mem_Lo1 窗口映射地址高 6 位 |
| 17:12 | trans_lo2 | 读写 | 0 | PCI_Mem_Lo2 窗口映射地址高 6 位 |
| 31:18 | 保留 | 只读 | 0 | |
| PCIX_Bridge_Cfg | | | | |
| 5:0 | pcix_rgate | 读写 | 6'h18 | PCIX 模式下向 DDR2 发读取数门限 |
| 6 | pcix_ro_en | 读写 | 0 | PCIX 桥是否允许写越过读 |
| 31:18 | 保留 | 只读 | 0 | |
| pcimap_cfg | | | | |
| 15:0 | dev_addr | 读写 | 0 | PCI 配置读写时 AD 线高 16 位 |
| 16 | conf_type | 读写 | 0 | 配置读写的类型 |
| 31:17 | 保留 | 只读 | 0 | |
| PCI_Hit*_Sel_* | | | | |
| 0 | 保留 | 只读 | 0 | |
| 2:1 | pci_img_size | 读写 | 2'b11 | 00：32 位；10：64 位；其它：无效 |
| 3 | pref_en | 读写 | 0 | 预取使能 |
| 11:4 | 保留 | 只读 | 0 | |
| 62:12 | bar_mask | 读写 | 0 | 窗口大小掩码（高位 1，低位 0） |
| 63 | burst_cap | 读写 | 1 | 是否允许突发传送 |
| PXArb_Config | | | | |
| 0 | device_en | 读写 | 1 | 外部设备允许 |
| 1 | disable_broken | 读写 | 0 | 禁用损坏的主设备 |
| 2 | default_mas_en | 读写 | 1 | 总线停靠到默认主设备<br>0：停靠到最后一个主设备<br>1：停靠到默认主设备 |
| 5:3 | default_master | 读写 | 0 | 总线停靠默认主设备号 |
| 7:6 | park_delay | 读写 | 0 | 从没有设备请求总线开始到触发停靠默认设备行为的延迟<br>00：0 周期<br>01：8 周期<br>10：32 周期<br>11：128 周期 |

| | | | | |
|---|---|---|---|---|
| 15:8 | level | 读写 | 8'h01 | 处于第一级的设备 |
| 23:16 | rude_dev | 读写 | 0 | 强制优先级设备<br>为 1 的位对应的 PCI 设备在得到总线后可以通过持续请求来占住总线 |
| 31:13 | 保留 | 只读 | 0 | |
| PXArb_Status | | | | |
| 7:0 | broken_master | 只读 | 0 | 损坏的主设备（改变禁用策略时清零） |
| 10:8 | Last_master | 只读 | 0 | 最后使用总线的主设备 |
| 31:11 | 保留 | 只读 | 0 | |

# 7 GPU

本章给出龙芯 1A 集成的高性能 2DGPU，该 GPU 拥有强大的图形处理能力。该 GPU 可以加速诸如 Symbian，Windows CE 和 Linux 的 GUI 性能，同时能够显著降低系统的总体功耗。

## 7.1 2D GPU 引擎

### 7.1.1 2D GPU 引擎框图

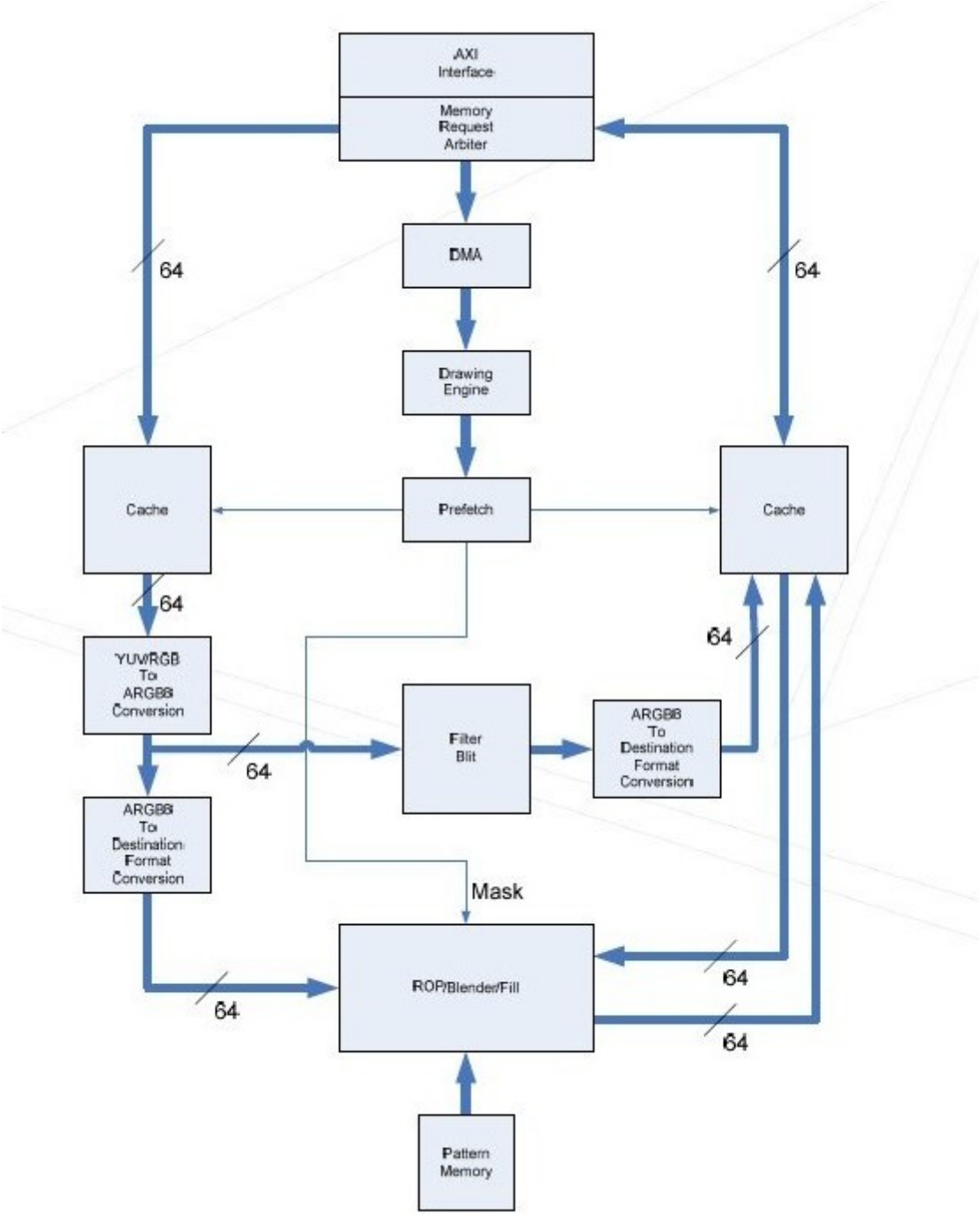


图 7-3　2D GPU 引擎框图

### 7.1.2　2D GPU 引擎支持的硬件图元操作

**Lines**

Lines 使用 Bresenham algorithm 进行渲染。在 Lines 时，仅仅 ROP2 和 ROP4 支持，并且在使用 ROP4 时，pattern 必须提供透明度掩码。在 Lines 时，裁剪操作基于像素级别。

**Rectangle Fill and Clear**

Rectangle Fill 操作用指定颜色填充矩形区域，该操作支持 ROP2 和 ROP4，在使用 ROP4 时，pattern 必须提供透明度掩码。

Clear 操作不需要 ROP 和 pattern 支持就可以填充矩形，一个 32bit 的值被用于填充整个矩形区域。

裁剪在 Rectangle Fill and Clear 中都是基于图元的。

**Stretch and Non-Stretch Bit Blit**

Bit Blit 操作将源内存区域的数据转移到目的内存区域。源内存和目的内存可以是相同或者不同的内存空间，源内存和目的内存可以是相同大小或者不同大小。若是不同大小，该操作就是 stretch 或者 shrink blit 操作。

Bit Blit 支持 ROP2, ROP3 和 ROP4。

修改后的 Bresenham algorithrm 被用于 fast stretching。Stretch factor 被定义为一个 15 fix-point 格式。Stretch blit 不允许重叠（源内存和目的内存共享）。Non stretch blit 允许重叠。

对于 Non-stretch blit，裁剪操作基于图元；对于 stretch blit，裁剪操作基于像素。

**Monchrome Expansion/Mask blit**

该操作与 bit blit 只有一点不同。该操作支持 monochrome mask 供 ROP4 选择。对于该操作，源和目的矩形必须同样大小。该操作不允许重叠，驱动程序必须保证 GPU 执行的命令在源和目的矩形上不重叠。

Mask blit 从显存中获取 color source, monochrome mask 从 command stream 中获取。

裁剪操作基于像素级别。

**Filter Blit**

该操作将矩形的源图像映射到大小不同的目的矩形。该操作支持高质量的 re-sampling filter。最大的 shrink 因子是 16，最大的 stretch 因子是 4。Filter Blit 不支持任何 ROP。裁剪基于像素级别。

**Rotation**

所有图元都支持 90 度旋转。

**Transparency Mode**

对于 monochrome expansion 支持：1 不透明 2 有条件透明（若当前像素匹配一特定值则透明）

对于 blits 支持：1 不透明 2 掩码透明（若当前像素的掩码为零则透明）3 有条件透明（同上）

**裁剪**

裁剪矩形支持所有图元。

**数据格式**

图形引擎支持的源数据格式：

A1R5G5B5、A4R4G4B4、A8R8G8B8、1-bit monochrome、R5G6B5、X1R5G5B5、X4R4G4B、 X8R8G8B8、UYUV YUV2、8-bit color index

**ARGB 数据转换**

对于 Filterblt，从 cache 中读取的数据会被转换成 A8R8G8B8 格式。对于其他图元，从 cache 中读取的数据会被转换成目的格式。写到 cache 中的数据总是目的格式。

**YUV to RGB 转换**

YUV 数据可以被转换成 RGB 格式的数据，一旦被转换，过程不可逆。

**颜色索引转换**

该功能仅仅适用于源数据，一个 256 的 look-up 表被用于索引数据，表格是可编程的。

**Alpha Blending**

**虚拟内存支持**

如果分配连续的物理内存不可能，GPU 可以支持虚拟地址。内存仲裁器包括一个虚拟地址到物理地址的转换模块。有两个含有 8 个入口的独立转换表分别用于源和目的地址的转换。为了使能该转换，31bit 必须被设置。内存仲裁器将会自动的转换。

MMU 有如下特征：

8entries per client(source,destination)

The entries are 20 bit wide

1 base address translation table address per client

Single level of translation table in memory

Bit 31 is used to indicate a virtual address

地址转换操作如下：

Incoming address: 31=1 30:12=lookup 11:0=offset

Base address: 31:12=base 11:0=0

4-byte request to AXI: lookup

<-{base[31:12],12'd0}+{11'd0,address[30:12],2'd0}

Resulting address: data<-{lookup[31:12],address[11:0]}

## 7.2 GPU 内部寄存器列表

| 参数名称 | 位 | 缺省值 | 范围 | 描述 |
|---|---|---|---|---|
| AQVertexElementCtrl [31:0] 地址: 0x1c200060 Reset Value:0x00000000 | | | | |
| FORMAT | 3:0 | 0x0 | 0x0-0xb | when vertex control element 0 is written // it is enabled and we disable all other VE controls // then we followed by writing to VE control N // (where N > 0) to enable the rest of the VE controls // The following definitions are general for D3D and AQ2 // Driver need to translate to the proper usage // BYTE = 0x0 // UBYTE = 0x1 // SHORT = 0x2 // USHORT = 0x3 // INT = 0x4 // UINT = 0x5 // DEC = 0x6 // UDEC = 0x7 // FLOAT = 0x8 // FLOAT16 = 0x9 // D3DCOLOR = 0xa // FIXED16DOT16 = 0xb |
| FETCH_BREAK | 7:7 | 0x0 | 0x0-0x1 | Enable a fetch break for the last element as well for any // other element that has a gap between itself and the next // element.<br>0=>disable, 1=>enable |
| SIZE | 13:12 | 0x0 | 0x0-0x1 | 1 --> one elements // 2 --> two elements // 3 --> three elements // 0 --> four element |
| NORMALIZE | 15:14 | 0x0 | 0x0-0x2 | 0 --> Disable // 1 --> D3D Normalization // 2 --> OES Normalization |
| OFFSET | 23:16 | 0x0 | | Offset into the vertex stream for the element. |
| FETCH_SIZE | 31:24 | 0x0 | | Only required when FetchBreak is enabled. Represents the // number of bytes to fetch |
| AQIndexStreamBaseAddr [31:0] 地址: 0x1c200644 Reset Value:0x00000000 | | | | |
| ADDRESS | 30:0 | 0x0 | | Base address for the index stream. |
| TYPE | 31:31 | 0x0 | 0x0-0x1 | 0=>SYSYEM 1=>VIRTUAL SYSYTEM |
| AQIndexStreamCtrl [31:0] 地址: 0x1c200648 Reset Value:0x00000000 | | | | |
| STRIDE | 0:0 | 0x0 | 0x0-0x1 | Stride of the index buffer: // 0 --> 8-bit indices. // 1 --> 16-bit indices |
| AQVertexStreamBaseAddr [31:0] 地址: 0x1c20064c Reset Value:0x00000000 | | | | |
| ADDRESS | 30:0 | 0x0 | | Base address for the vertex stream. |
| TYPE | 31:31 | 0x0 | 0x0-0x1 | 0=>SYSYEM 1=>VIRTUAL SYSYTEM |
| AQVertexStreamCtrl [31:0] 地址: 0x1c200650 Reset Value:0x00000000 | | | | |
| STRIDE | 8:0 | 0x0 | | Stride of the vertex stream in bytes |
| AQCmdBufferAddr [31:0] 地址: 0x1c200654 Reset Value:0x00000000 | | | | |
| ADDRESS | 30:0 | 0x0 | | Base address for the command buffer. |
| TYPE | 31:31 | 0x0 | 0x0-0x1 | 0=>SYSYEM 1=>VIRTUAL SYSYTEM |
| AQCmdBufferCtrl [31:0] 地址: 0x1c200658 Reset Value:0x00000000 | | | | |
| PRE_FETCH | 15:0 | 0x0 | | Number of 64-bit words to fetch from the command buffer |
| ENABLE | 16:16 | 0x0 | 0x0-0x1 | Enable the command parser.<br>0=>disable, 1=>enable |
| AQFEStatus [31:0] 地址: 0x1c20065c Reset Value:0x00000000 | | | | |

| | | | | |
|---|---|---|---|---|
| COMMAND_DATA | 0:0 | 0x0 | 0x0-0x1 | Status of the command parser. // 0 --> Idle // 1 --> Busy |
| AQFEDebugCurCmdAdr [31:0] 地址: 0x1c200664 Reset Value:0x00000000 | | | | |
| CUR_CMD_ADR | 31:3 | 0x0 | | |
| AQHiClockControl [31:0] 地址: 0x1c200000 Reset Value:0x00000100 | | | | |
| CLK3D_DIS | 0:0 | 0x0 | 0x0-0x1 | Disable 3D clk |
| CLK2D_DIS | 1:1 | 0x0 | 0x0-0x1 | Disable 2D clk |
| IDLE3_D | 16:16 | 0x0 | 0x0-0x1 | 3D pipe is idle |
| IDLE2_D | 17:17 | 0x0 | 0x0-0x1 | 2D pipe is idle |
| AQHiIdle [31:0] 地址: 0x1c200004 Reset Value:0x00000000 | | | | |
| IDLE_FE | 0:0 | 0x0 | 0x0-0x1 | FE is idle |
| IDLE_DE | 1:1 | 0x0 | 0x0-0x1 | DE is idle |
| IDLE_PE | 2:2 | 0x0 | 0x0-0x1 | PE is idle |
| AQAxiConfig [31:0] 地址: 0x1c200008 Reset Value:0x00000000 | | | | |
| AWID | 3:0 | 0x0 | | AxiConfig |
| ARID | 7:4 | 0x0 | | AxiConfig |
| AWCACHE | 11:8 | 0x0 | | AxiConfig |
| ARCACHE | 15:12 | 0x0 | | AxiConfig |
| AQAxiStatus [31:0] 地址: 0x1c20000c Reset Value:0x00000000 | | | | |
| DET_RD_ERR | 9:9 | 0x0 | | |
| DET_WR_ERR | 8:8 | 0x0 | | |
| RD_ERR_ID | 7:4 | 0x0 | | |
| WR_ERR_ID | 3:0 | 0x0 | | |
| AQIntrAcknowledge [31:0] 地址: 0x1c200010 Reset Value:0x00000000 | | | | |
| INTR_VEC | 31:0 | 0x0 | | Interrupt acknowledge register. Each bit represents a // corresponding event being triggered. Reading frmo this // register clears the outstanding interrupt. |
| AQIntrEnbl [31:0] 地址: 0x1c200014 Reset Value:0x00000000 | | | | |
| INTR_ENBL_VEC | 31:0 | 0x0 | | Interrupt enable register. Each bit enables a corresponding event |
| GCFeatures [31:0] 地址: 0x1c20001c Reset Value:0x00000000 | | | | |
| FAST_CLEAR | 0:0 | 0x0 | 0x0-0x1 | Fast clear.<br>0=>none 1=>available |
| SPECIAL_ANTI_ALIASING | 1:1 | 0x0 | 0x0-0x1 | Full-screen anti-aliasing.<br>0=>none 1=>available |
| PIPE_3D | 2:2 | 0x0 | 0x0-0x1 | 3D pipe.<br>0=>none 1=>available |
| DXT_TEXTURE_COMPRESSION | 3:3 | 0x0 | 0x0-0x1 | DXT texture compression.<br>0=>none 1=>available |
| DEBUG_MODE | 4:4 | 0x0 | 0x0-0x1 | Debug registers.<br>0=>none 1=>available |
| ZCOMPRESSION | 5:5 | 0x0 | 0x0-0x1 | Depth compression.<br>0=>none 1=>available |
| YUV420_FILTER | 6:6 | 0x0 | 0x0-0x1 | YUV 4:2:0 support in filter blit<br>0=>none 1=>available |
| MSAA | 7:7 | 0x0 | 0x0-0x1 | MSAA support<br>0=>none 1=>available |
| DC | 8:8 | 0x0 | 0x0-0x1 | Shows if there is a display controller in the IP.<br>0=>none 1=>available |
| PIPE_2D | 9:9 | 0x0 | 0x0-0x1 | Shows if there is 2D engine<br>0=>none 1=>available |
| ETC1_TEXTURE_COMPRESSION | 10:10 | 0x0 | 0x0-0x1 | ETC1 texture compression<br>0=>none 1=>available |
| FAST_SCALER | 11:11 | 0x0 | 0x0-0x1 | Shows if the IP has HD scaler<br>0=>none 1=>available |

| | | | | |
|---|---|---|---|---|
| HIGH_DYNAMIC_RANGE | 12:12 | 0x0 | 0x0-0x1 | Shows if the IP has HDR support<br>0=>none 1=>available |
| YUV420_TILER | 13:13 | 0x0 | 0x0-0x1 | YUV 4:2:0 tiler is available<br>0=>none 1=>available |
| MODULE_CG | 14:14 | 0x0 | 0x0-0x1 | Second level clock gating is available<br>0=>none 1=>available |
| MIN_AREA | 15:15 | 0x0 | 0x0-0x1 | IP is configured to have minimum area<br>0=>none 1=>available |
| NO_EZ | 16:16 | 0x0 | 0x0-0x1 | IP does not have early-Z<br>0=>none 1=>available |
| NO422_TEXTURE | 17:17 | 0x0 | 0x0-0x1 | IP does not have 422 texture input format<br>0=>none 1=>available |
| BUFFER_INTERLEAVING | 18:18 | 0x0 | 0x0-0x1 | IP supports interleaving depth and color buffers.<br>0=>none 1=>available |
| BYTE_WRITE_2D | 19:19 | 0x0 | 0x0-0x1 | Supports byte write in 2D<br>0=>none 1=>available |
| NO_SCALER | 20:20 | 0x0 | 0x0-0x1 | IP does not have 2D scaler<br>0=>none 1=>available |
| AQMemoryFePageTable [31:0] 地址: 0x1c200400 Reset Value:0x00000000 | | | | |
| BASE_ADDRESS | 31:12 | 0x0 | | Base address for FE virtual address lookup table |
| AQMemoryPePageTable [31:0] 地址: 0x1c200408 Reset Value:0x00000000 | | | | |
| BASE_ADDRESS | 31:12 | 0x0 | | Base address for color buffer virtual address lookup table |
| AQMemoryFe [31:0] 地址: 0x1c20041c Reset Value:0x00000000 | | | | |
| BASE_ADDRESS | 31:0 | 0x0 | | Base address for all FE memory requests (all addresses are added with this before going out of the chip) |
| AQMemoryPec [31:0] 地址: 0x1c200428 Reset Value:0x00000000 | | | | |
| BASE_ADDRESS | 31:0 | 0x0 | | Base address for all PE-Color memory requests (all addresses are added with this before going out of the chip) |
| gcMemoryFlush [31:0] 地址: 0x1c200430 Reset Value:0x00000000 | | | | |
| PAGE_TABLE | 62:56 | 0x0 | | Flush the page table cache<br>0=>DISABLE 1=>ENABLE |
| gcDbgCycleCounter [31:0] 地址: 0x1c200438 Reset Value:0x00000000 | | | | |
| COUNT | 31:0 | 0x0 | | Increments every cycle |
| gcDebugSignalsPe [31:0] 地址: 0x1c200454 Reset Value:0x00000000 | | | | |
| SIGNAL | 31:0 | 0x0 | | Signals according to select signal: // 0 -> pixel count killed by color pipe // 1 -> pixel count killed by depth pipe // 2 -> pixel count drawn by color pipe // 3 -> pixel count drawn by depth pipe // 4 -> debug signals for 3d_io, 2d_filter, 2d_fsm // 5 -> debug signals for cache2d_cntrl // 6 -> debug signals for cache2d_tag_alloc // 7 -> debug signals for cache3d_c_cntrl, cache3d_c_tag_alloc // 8 -> debug signals for cache3d_z_cntrl, cache3d_z_tag_alloc // 9 -> debug signals for pref_2d, pref_3d // a -> debug signals for cmd_state // F -> Signature = 0xbabef00d. |
| gcDebugSignalsMc [31:0] 地址: 0x1c200468 Reset Value:0x00000000 | | | | |

| | | | | |
|---|---|---|---|---|
| SIGNAL | 31:0 | 0x0 | | Signals according to select signal: // 0 -> Various signals from FC block. // 1 -> Total read req in terms of 8B from pipeline. // 2 -> Total read req in terms of 8B sent out from the IP. // 3 -> Total write req in terms of 8B from pipeline. // F -> Signature = 0x12345678 |
| gcDebugSignalsHi [31:0] 地址: 0x1c20046c Reset Value:0x00000000 | | | | |
| SIGNAL | 31:0 | 0x0 | | Signals according to select signal: // 0 -> Number of cycles AXI read request is stalled. // 1 -> Number of cycles AXI write request is stalled. // 2 -> Number of cycles AXI write data is stalled. // F -> Signature = 0xaaaaaaaa. |
| gcDebugControl0 [31:0] 地址: 0x1c200470 Reset Value:0x00000000 | | | | |
| FE | 3:0 | 0x0 | | Selects which set of 32 bit data to get from Fe. // Resets the counters if set to 0xf. |
| DE | 11:8 | 0x0 | | Selects which set of 32 bit data to get from De. // Resets the counters if set to 0xf. |
| gcDebugControl2 [31:0] 地址: 0x1c200478 Reset Value:0x00000000 | | | | |
| MC | 3:0 | 0x0 | | Selects which set of 32 bit data to get from Mc. // Resets the counters if set to 0xf. |
| HI | 11:8 | 0x0 | | Selects which set of 32 bit data to get from Hi. // Resets the counters if set to 0xf. |
| gcregEndianness0 [31:0] 地址: 0x1c200484 Reset Value:0x00000000 | | | | |
| WORD_SWAP | 31:0 | 0x0 | | Flip the words of 32 bit data. |
| gcregEndianness1 [31:0] 地址: 0x1c200488 Reset Value:0x00000000 | | | | |
| BYTE_SWAP | 31:0 | 0x0 | | Flip the bytes of 16 bit data. |
| gcregEndianness2 [31:0] 地址: 0x1c20048c Reset Value:0x00000000 | | | | |
| BIT_SWAP | 31:0 | 0x0 | | Flip the bits of 8 bit data. |

# 8 LCD

本章给出龙芯 1A 内 LCD 控制器（Display Controller）的配置使用。1A 的 LCD 支持 DVI 和 VGA 两路输出。

## 8.1 特性

- 支持格式转换
- 最大显示支持到 1920×1080@60Hz
- 同步信号可编程
- Gamma 调整查找表
- 颜色抖动（Dithering）

## 8.2 数据格式

Display Controller 支持以下数据格式：

R4G4B4 -> 12 bits per pixel
R5G5B5 -> 15 bits per pixel
R5G6B5 -> 16 bits per pixel
R8G8B8 -> 24 bits per pixel

## 8.3 寄存器

寄存器列表（寄存器共 32 位地址，高 12 位为 12'hbc2，低 20 位下表所示）：

| 寄存器名 | 对应于显示器 0<br>的配置地址 | 对应于显示器 1<br>的配置地址 |
|---|---|---|
| Frame Buffer configuration | {15' h92,5' h0} | {15' h92,5' h10} |
| Frame Buffer Address_0 | {15' h93,5' h0} | {15' h93,5' h10} |
| Frame Buffer Address_1 | {15' hAC,5' h0} | {15' hAC,5' h10} |
| Frame Buffer Stride | {15' h94,5' h0} | {15' h94,5' h10} |
| Frame Buffer Origin | {15' h95,5' h0} | {15' h95,5' h10} |
| Display Ditheronfiguration | {15' h9B,5' h0} | {15' h9B,5' h10} |
| Display Dither Table(low) | {15' h9C,5' h0} | {15' h9C,5' h10} |
| Display Dither Table(high) | {15' h9D,5' h0} | {15' h9D,5' h10} |
| Pane Configuration | {15' h9E,5' h0} | {15' h9E,5' h10} |
| Panel Timing | {15' h9F,5' h0} | {15' h9F,5' h10} |
| HDisplay | {15' hA0,5' h0} | {15' hA0,5' h10} |
| Hsync | {15' hA1,5' h0} | {15' hA1,5' h10} |
| VDisplay | {15' hA4,5' h0} | {15'hA4,5' h10} |
| VSync | {15' hA5,5' h0} | {15'hA5,5' h10} |
| Cursor Configuration | {16' h152,4' h0} | |
| Cursor Address | {16' h153,4' h0} | |
| Cursor Location | {16' h154,4' h0} | |

| | | |
|---|---|---|
| Cursor Background | {16’h155,4’h0} | |
| Cursor Foreground | {16’h156,4’h0} | |
| Gamma Index | {15’hA7,5’h0} | {15’hA7,5’h10} |
| Gamma Data | {15’hA8,5’h0} | {15’hA8,5’h10} |

| Frame Buffer Configuration | bit | 描述 | 初始值 |
|---|---|---|---|
| Reset | 20 | 写 0 reset | 0 |
| Gamma enable | 12 | 写 1 使能 | 0 |
| Switch Panel | 9 | 置 1 时，表示该显示单元的输出使用另外一个显示单元的输出，即如果对 0 号显示单元配置该位时表示 0 号单元的输出和输出控制信号复制于 1 号显示单元的输出，同理如果对 1 号显示单元配置该位表示 1 号显示单元的输出和输出控制信号复制于 0 号显示单元。 | 0 |
| Output enable | 8 | 写 1 使能输出，写 0 则不输出显示数据 | 0 |
| Format | [2:0] | 0 none<br>1 R4G4B4<br>2 R5G5B5<br>3 R5G6B5<br>4 R8G8B8 | 0 |

| Frame buffer address_0 | Bit | 描述 | 初始值 |
|---|---|---|---|
| Frame buffer address_0 | [31:0] | 内存中图像数据首地址 | 32’h0000_0000 |

| Frame buffer address_1 | Bit | 描述 | 初始值 |
|---|---|---|---|
| Frame buffer address_1 | [31:0] | 对于需要支持双 frame buffer 显示的情况，此时该寄存器可配置第二块 Frame Buffer 的地址，DC 运行时第一帧先从 Frame Buffer_0 取数据，第二帧从 Frame Buffer_1 取数据，第三帧再从 Frame Buffer_0 取数据，依此循环。对于不需要双 frame buffer 的情况，可将此 Frame Buffer Address_1 配置成和 Frame Buffer_0 一样的地址即可 | 32’h0000_0000 |

| Frame buffer stride | Bit | 描述 | 初始值 |
|---|---|---|---|
| Frame buffer stride | [31:0] | 显示屏一行的字节数 | 32’h0000_0000 |

| Frame buffer origin | Bit | 描述 | 初始值 |
|---|---|---|---|
| Frame buffer origin | [31:0] | 显示屏左侧原有字节数，一般配 0 即可 | 32’h0000_0000 |

| Display Dither Configuration | Bit | 描述 | 初始值 |
|---|---|---|---|
| Enable | 31 | 置 1 使能 dither 功能 | 0 |
| RedSize | [19:16] | 红色域宽度 | 4’b0000 |

| | | | |
|---|---|---|---|
| GreeenSize | [11:8] | 绿色域宽度 | 4’b0000 |
| BlueSize | [3:0] | 蓝色域宽度 | 4’b0000 |

| Display Dither Table | Bit | 描述 | 初始值 |
|---|---|---|---|
| Display Dither Table | [63:0] | 该寄存器有64位，而Display Controller的寄存器都是 32 位宽，所以实际上该寄存器为两个 32 位的寄存器。分为Display Dither Table(low)和 Display Dither Table(high)。这两个寄存器以像素点的X和Y坐标为索引，配置作为比较的数值。凡进入Dtiher处理模块的图像数据都会在Display Dither Talbe寄存器中被相应的索引到一个比较值，若输入数据的值的后四位大于该比较值则进行颜色增强。 | 64’h0000_0000 |
| Display Dither Table（low） | [31:0] | | |
| Y0_X0 | [3:0] | 坐标（0，0）处的比较值 | 4’b0000 |
| Y0_X1 | [7:4] | 坐标（1，0）处的比较值 | 4’b0000 |
| Y0_X2 | [11:8] | 坐标（2，0）处的比较值 | 4’b0000 |
| Y0_X3 | [15:12] | 坐标（3，0）处的比较值 | 4’b0000 |
| Y1_X0 | [19:16] | 坐标（0，1）处的比较值 | 4’b0000 |
| Y1_X1 | [23:20] | 坐标（1，1）处的比较值 | 4’b0000 |
| Y1_X2 | [27:24] | 坐标（2，1）处的比较值 | 4’b0000 |
| Y1_X3 | [31:28] | 坐标（3，1）处的比较值 | 4’b0000 |
| Display Dither Table（high） | [31:0] | | |
| Y2_X0 | [3:0] | 坐标（0，2）处的比较值 | 4’b0000 |
| Y2_X1 | [7:4] | 坐标（1，2）处的比较值 | 4’b0000 |
| Y2_X2 | [11:8] | 坐标（2，2）处的比较值 | 4’b0000 |
| Y2_X3 | [15:12] | 坐标（3，2）处的比较值 | 4’b0000 |
| Y3_X0 | [19:16] | 坐标（0，3）处的比较值 | 4’b0000 |
| Y3_X1 | [23:20] | 坐标（1，3）处的比较值 | 4’b0000 |
| Y3_X2 | [27:24] | 坐标（2，3）处的比较值 | 4’b0000 |
| Y3_X3 | [31:28] | 坐标（3，3）处的比较值 | 4’b0000 |

| Panel configuration | Bit | 描述 | 初始值 |
|---|---|---|---|
| ClockPolarity | 9 | 时钟极性，置1将时钟反向 | 0 |
| Clock | 8 | 时钟使能，置1使能时钟 | 1 |
| DE_Polarity | 1 | 数据使能极性，置1取反，一般设0 | 0 |
| DE | 0 | 数据使能，置1使能数据输出 | 1 |

| HDisplay | Bit | 描述 | 初始值 |
|---|---|---|---|
| Total | [27:16] | 显示屏一行的总体像素数（包括非显示区） | 12’b0 |
| DisplayEnd | [11:0] | 显示屏一行中显示区的像素数 | 12’b0 |

| HSync | Bit | 描述 | 初始值 |
|---|---|---|---|

| | | | |
|---|---|---|---|
| Polarity | 31 | HSync 信号的极性，置 1 取反，一般设 0 | 0 |
| Pulse | 30 | HSync 信号使能，置 1 只能 HSync 信号输出 | 1 |
| End | [27:16] | HSync 信号结束的像素数 | 12’b0 |
| Start | [11:0] | HSync 信号开始的像素数 | 12’b0 |

| VDisplay | Bit | 描述 | 初始值 |
|---|---|---|---|
| Total | [26:16] | 显示屏总体的行数（包括消隐区） | 11’b0 |
| DisplayEnd | [10:0] | 显示屏中显示区的行数 | 11’b0 |

| VSync | Bit | 描述 | 初始值 |
|---|---|---|---|
| Polarity | 31 | VSync 信号的极性，置 1 取反，一般设 0 | 0 |
| Pulse | 30 | VSync 信号使能，置 1 只能 VSync 信号输出 | 1 |
| End | [27:16] | VSync 信号结束的行数 | 12’b0 |
| Start | [11:0] | VSync 信号开始的行数 | 12’b0 |

| Cursor Configuration | Bit | 描述 | 初始值 |
|---|---|---|---|
| HotSpotX | [20:16] | 指针的“焦点”（作用点）的横坐标(在指针 32*32 的图案中的横坐标) | 5’b0 |
| HotSpotY | [12:8] | 指针的“焦点”（作用点）的纵坐标(在指针 32*32 的图案中的横坐标) | 5’b0 |
| Display | 4 | 指示指针存在于哪个显示单元中，0 表示在 0 号显示单元中，1 表示指针在 1 号显示单元中 | 0 |
| Format | [1:0] | 0 disabled<br>1 masked<br>2 A8R8G8B8 | 2’b0 |

| Cursor Address | Bit | 描述 | 初始值 |
|---|---|---|---|
| Cursor Address | [31：0] | 指针数据在内存中的基地址 | 32’b0 |

| Cursor Location | Bit | 描述 | 初始值 |
|---|---|---|---|
| Y | [26:16] | 指针的焦点在整个显示区的纵坐标 | 11’b0 |
| X | [10:0] | 指针的焦点在整个显示区的横坐标 | 11’b0 |
| Cursor Background | Bit | 描述 | 初始值 |
| Red | [23:16] | 指针单色模式下背景色的红色域 | 8’b0 |
| Green | [15:8] | 指针单色模式下背景色的绿色域 | 8’b0 |
| Blue | [7:0] | 指针单色模式下背景色的蓝色域 | 8’b0 |

| Cursor Foreground | Bit | 描述 | 初始值 |
|---|---|---|---|
| Red | [23:16] | 指针单色模式下前景色的红色域 | 8’b0 |
| Green | [15:8] | 指针单色模式下前景色的绿色域 | 8’b0 |
| Blue | [7:0] | 指针单色模式下前景色的蓝色域 | 8’b0 |

| Gamma Index | Bit | 描述 | 初始值 |
|---|---|---|---|
| Index | [7:0] | 表示从 0-255 颜色值之间的哪一项开始进行 Gamma 调整，一般设 0。只需配一次，此后该值硬件会自增。 | 8'b0 |

| Gamma Data | Bit | 描述 | 初始值 |
|---|---|---|---|
| Red | [23:16] | Gamma 调整的红色域，将 Gamma Index 指示的值调整为当前域的值 | 8'b0 |
| Green | [15:8] | Gamma 调整的绿色域，将 Gamma Index 指示的值调整为当前域的值 | 8'b0 |
| Blue | [7:0] | Gamma 调整的蓝色域，将 Gamma Index 指示的值调整为当前域的值 | 8'b0 |

LCD 支持 RGB444/555/565/888 位模式。在不同的 RGB 模式下，1A-PAD 和外部 LCD 数据线的连接方法如下所示（表中 GPIO 表示该 PAD 可以悬空，也可以复用为 GPIO）：

**配置如下所示，**

| 1A-PAD | RGB 444 | RGB 555 | RGB 565 | RGB 888 |
|---|---|---|---|---|
| LCD_DAT_B0 | GPIO | GPIO | GPIO | LCD_BLUE0 |
| LCD_DAT_B1 | GPIO | GPIO | GPIO | LCD_BLUE1 |
| LCD_DAT_B2 | GPIO | GPIO | GPIO | LCD_BLUE2 |
| LCD_DAT_B3 | GPIO | LCD_BLUE0 | LCD_BLUE0 | LCD_BLUE3 |
| LCD_DAT_B4 | LCD_BLUE0 | LCD_BLUE1 | LCD_BLUE1 | LCD_BLUE4 |
| LCD_DAT_B5 | LCD_BLUE1 | LCD_BLUE2 | LCD_BLUE2 | LCD_BLUE5 |
| LCD_DAT_B6 | LCD_BLUE2 | LCD_BLUE3 | LCD_BLUE3 | LCD_BLUE6 |
| LCD_DAT_B7 | LCD_BLUE3 | LCD_BLUE4 | LCD_BLUE4 | LCD_BLUE7 |
| LCD_DAT_G0 | GPIO | GPIO | GPIO | LCD_GREEN0 |
| LCD_DAT_G1 | GPIO | GPIO | GPIO | LCD_GREEN1 |
| LCD_DAT_G2 | GPIO | GPIO | LCD_GREEN0 | LCD_GREEN2 |
| LCD_DAT_G3 | GPIO | LCD_GREEN0 | LCD_GREEN1 | LCD_GREEN3 |
| LCD_DAT_G4 | LCD_GREEN0 | LCD_GREEN1 | LCD_GREEN2 | LCD_GREEN4 |
| LCD_DAT_G5 | LCD_GREEN1 | LCD_GREEN2 | LCD_GREEN3 | LCD_GREEN5 |
| LCD_DAT_G6 | LCD_GREEN2 | LCD_GREEN3 | LCD_GREEN4 | LCD_GREEN6 |
| LCD_DAT_G7 | LCD_GREEN3 | LCD_GREEN4 | LCD_GREEN5 | LCD_GREEN7 |
| LCD_DAT_R0 | GPIO | GPIO | GPIO | LCD_RED0 |
| LCD_DAT_R1 | GPIO | GPIO | GPIO | LCD_RED1 |
| LCD_DAT_R2 | GPIO | GPIO | GPIO | LCD_RED2 |
| LCD_DAT_R3 | GPIO | LCD_RED0 | LCD_RED0 | LCD_RED3 |
| LCD_DAT_R4 | LCD_RED0 | LCD_RED1 | LCD_RED1 | LCD_RED4 |
| LCD_DAT_R5 | LCD_RED1 | LCD_RED2 | LCD_RED2 | LCD_RED5 |
| LCD_DAT_R6 | LCD_RED2 | LCD_RED3 | LCD_RED3 | LCD_RED6 |
| LCD_DAT_R7 | LCD_RED3 | LCD_RED4 | LCD_RED4 | LCD_RED7 |

# 9 GMAC0

## 9.1 配置成 MAC 的连接和复用方式

GMAC0 控制器可以通过配置成百兆模式(MII)或千兆模式(RGMII)。如果外部连接百兆 PHY，需要复用 PWM0，PWM1 和 GMAC0_TX_CLK_O 三个 PAD，配置如下表：

| PAD | MAC 信号 | 配置位 | 复位值 |
|---|---|---|---|
| PWM0 | MAC_0_COL | GMA0_USE_PWM01 | 1’b0 |
| PWM1 | MAC_0_CRS | GMA0_USE_PWM01 | 1’b0 |
| GMAC0_TX_CLK_O | MAC_0_RX_ERR | GMA0_USE_TX_CLK | 1’b1 |

寄存器地址：0x1fd0_0420

| 配置位 | Bit 位 | 描述 |
|---|---|---|
| GMA0_USE_PWM01 | 8 | 1：百兆模式 MAC_COL/MAC_CRS 分别复用 PWM0/1<br>0：千兆模式 |
| GMA0_USE_TX_CLK | 10 | 1：Mii_0_RX_ERR 输入信号复用 GMAC0_TX_CLK_O<br>0：千兆模式 |

百兆模式下，如果外部 PHY 不提供 RX_ERR 信号，GMAC0_TX_CLK_O 需接地，不能悬空；

百兆模式下，如果外部 PHY 提供 RX_ERR 信号，GMAC0_TX_CLK_O 与 RX_ERR 连接。

## 9.2 DMA 寄存器描述

GMAC 内部集成独有的 DMA 控制器，专门配合 GMAC 数据传输；该 DMA 控制器不能被其他模块使用，GMAC 也无法使用外部其他 DMA（见 17 章）。GMAC 寄存器包括 GMAC 寄存器部分和 DMA 寄存器部分。GMAC0 的 GMAC 寄存器的起始地址是 0x1fe1_0000；GMAC0 的 DMA 寄存器的起始地址是 0x1fe1_1000。

下面分别介绍 DMA 寄存器和 GMAC 寄存器的意义。

| 参数名称 | 位 | 缺省值 | 描述 |
|---|---|---|---|
| Register0 (Bus Mode Register)　Offset: 0x00 | | | |
| Reserved<br>保留 | 31:27 | 0x0 | 保留，只读 |
| MB: Mixed Burst | 26 | 0x0 | 当此位为高，FB 位为低时，AXI master 在突发访问 |

| | | | |
|---|---|---|---|
| 混合突发访问 | | | 长度大于 16 时采用 INCR 访问模式，当突发访问长度为 16 或者小于 16 时采用 FIX 访问模式。用户不用关心此位设置。 |
| AAL:Address-Aligned Beats<br>地址对齐节拍 | 25 | 0x0 | 当此位和 FB 位同时为高时，AXI 接口的所有访问将对齐到起始地址的 LS 位。如果 FB 位为 0，首次访问地址访问不对齐，剩余的访问地址对齐。用户不用关心此位设置。 |
| 8XPBL Mode<br>是否使能 PBLX8 模式 | 24 | 0x0 | 此位为高时，GMAC DMA 的最大突发数据传输长度为 8,16,32,64,128 或者 256。最大突发长度取决于 PBL。用户不用关心此位设置。 |
| USP:Use Separate PBL<br>使用分离的 PBL 值 | 23 | 0x0 | 此位为高时，PBL 值只应用于 TxDMA。此位为低时，PBL 值应用于 TxDMA 和 RxDMA。用户不用关心此位设置。 |
| RPBL: RxDMA PBL<br>RxDMA 突发传输长度 | 22:17 | 0x01 | 表示一次 RxDMA 传输的最大突发传输长度。只能为 1,2,4,8,16 和 32，其它值无效。 |
| FB: Fixed Burst<br>定长突发传输长度使能 | 16 | 0x0 | 指定 AXI Master 接口是否采用 FIX 突发传输模式。用户不用关心此位设置。 |
| PR: Rx:Tx priority ratio<br>RxDMA 与 TxDMA 优先级比例 | 15:14 | 0x0 | 在 DA 位为 0 时起作用。<br>00：1：1<br>01：2：1<br>10：3：1<br>11：4：1 |
| PBL:Programmable Burst Length<br>可编程突发传输长度 | 13:8 | 0x1 | 用户不用关心此设置。 |
| ATDS:Alternate Descriptor size<br>是否使用 32 字节大小描述符 | 7 | 0x0 | 此位为 1 时使用 32 字节大小的描述符<br>此位为 0 时使用 16 字节大小的描述符 |
| DSL: Descriptor Skip Length<br>描述符间隔距离 | 6:2 | 0x00 | 设置 2 个描述符间的距离。但此值为 0 时，默认为 DMA 描述符大小。 |

| | | | |
|---|---|---|---|
| DA: DMA Arbitration scheme<br>DMA 传输仲裁策略 | 1 | 0x0 | 0:在 RxDMA 和 TxDMA 间采用轮转仲裁机制<br>1: RxDMA 优先级高于 TxDMA 优先级。具体比值见 PR 值。 |
| SWR:Software Reset<br>软件复位 | 0 | 0x1 | 此位置高 DMA 控制器将复位 GMAC 内部寄存器和逻辑。当复位结束时该位自动清零。 |
| Register1 (Transmit Poll Demand Register) Offset: 0x04 | | | |
| TPD: Transmit Poll Demand<br>传输轮询使能 | 31:0 | 0x0 | 向此值写入任意值，发送 DMA 控制器将会读取寄存器 18 对应的描述符。如果该描述符无效，DMA 传输将会停止。如果该描述符有效，DMA 传输将会继续。 |
| Register2 (Receive Poll Demand Register) Offset: 0x08 | | | |
| RPD: Receive Poll Demand<br>接收轮询使能 | 31:0 | 0x0 | 向此值写入任意值，接收 DMA 控制器将会读取寄存器 18 对应的描述符。如果该描述符无效，DMA 传输将会停止。如果该描述符有效，DMA 传输将会继续。 |
| Register3 (Receive Descriptor List Address Register) Offset: 0x0C | | | |
| Start of Receive List<br>接收描述符起始地址 | 31:0 | 0x0 | 指向接收描述符首地址。 |
| Register4 (Transmit Descriptor List Address Register) Offset: 0x10 | | | |
| Start of Transmit List<br>发送描述符起始地址 | 31:0 | 0x0 | 指向发送描述符首地址 |
| Register5 (Status Register) Offset: 0x14 | | | |
| Reserved | 31:30 | | 保留，只读 |
| TTI: Time-Stamp Trigger Interrupt<br>时间戳触发中断 | 29 | 0x0 | 时间戳模块触发中断。只读。 |
| GPI:GMAC PMT | 28 | 0x0 | 电源管理模块触发中断。只读。 |

| | | | |
|---|---|---|---|
| Interrutp<br>电源管理模块触发中断 | | | |
| GMI:GMAC MMC Interrupt<br>MMC 模块触发中断 | 27 | 0x0 | MMC 模块触发中断。只读。 |
| GLI:GMAC Line interface Interrupt<br>GMAC 模块线路触发中断 | 26 | 0x0 | GMAC 模块的 PCS 或者 RGMII 模块触发中断。只读。 |
| EB: Error Bits<br>错误位 | 25:23 | 0x0 | 23：1'b1 TxDMA 数据传输过程中发生错误<br>1'b0 RxDMA 数据传输过程中发生错误<br>24：1'b1 读传输错误<br>1'b0 写传输错误<br>25：1'b1 描述符访问错误<br>1'b0 数据缓存访问错误 |
| TS:Transmit Process State<br>传输过程状态 | 22:20 | 0x0 | 3'b000:传输停止；复位或者停止命令发送<br>3'b001:正在进行；获取传输描述符<br>3'b010:正在进行；等待传输状态<br>3'b011:正在进行；从发送缓存读取数据并发送到传输 FIFO(TxFIFO)<br>3'b100:写入时间戳状态<br>3'b101:保留<br>3'b110:挂起；传输描述符不可用或者传输缓存下溢。<br>3'b111:运行；关闭传输描述符。 |
| RS:Receive Process State<br>接收过程状态 | 19:17 | 0x0 | 3'b000:停止；复位或者接收到停止命令<br>3'b001:运行；获取接收描述符。<br>3'b010:保留；<br>3'b011:运行；等待接收包。<br>3'b100:暂停；接收描述符不可用。<br>3'b101:运行；关闭接收描述符。<br>3'b110:时间戳写状态。 |

| | | | |
|---|---|---|---|
| | | | 3’b111:运行；将包内容从接收缓存传输到系统内存。 |
| NIS:Normal Interrutp Summary<br>正常中断汇总 | 16 | 0x0 | 提示系统是否存在正常中断。 |
| AIS:Abnormal Interrutp Summary<br>异常中断汇总 | 15 | 0x0 | 提示系统是否存在异常中断。 |
| ERI:Early Receive Interrutp<br>提前接收中断 | 14 | 0x0 | 提示DMA控制器已经把包的第一个数据写入接收缓存 |
| FBI:Fatal Bus Error Interrutp<br>总线错误中断 | 13 | 0x0 | 提示总线错误，具体信息见[25:23]。当此位设置后DMA 引擎停止总线访问操作。 |
| Reserved | 12:11 | 0x0 | 保留 |
| ETI:Early Transmit Interrupt<br>提前发送中断 | 10 | 0x0 | 提示需要传输的以太网帧已经完全传输到 MTL 模块中的传输 FIFO |
| RWT:Receive Watchdog Timeout<br>接收看门狗超时 | 9 | 0x0 | 提示接收到一个大小超过2048字节的以太网帧。(当巨帧使能时，提示接收到大小超过 10240 字节的以太网帧) |
| RPS:Receive Process Stopped<br>接收过程停止 | 8 | 0x0 | 指示接收过程停止 |
| RU:Receive Buffer Unavailable<br>接收缓存不可用 | 7 | 0x0 | 指示接收缓存不可用 |
| RI:Receive Interrupt<br>接收中断 | 6 | 0x0 | 指示帧接收完成。帧接收的状态信息已经写入接收描述符。接收处于运行状态。 |
| UNF:Transmit Underflow<br>传输缓存下溢 | 5 | 0x0 | 指示帧发送过程中产生接收缓存下溢。 |
| OVF:Receive | 4 | 0x0 | 指示帧接收过程中接收缓存上溢。 |

| | | | |
|---|---|---|---|
| Overflow<br>接收缓存上溢 | | | |
| TJT:Transmit Jabber Timeout | 3 | 0x0 | |
| TU:Transmit Buffer Unavailable<br>传输缓存不可用 | 2 | 0x0 | 提示传输列表中的下一个描述符不能被 DMA 控制器访问。 |
| TPS:Transmit Process Stopped<br>传输过程停止 | 1 | 0x0 | 提示传输过程停止 |
| TI:Transmit Interrutp<br>传输完成中断 | 0 | 0x0 | 提示帧传输完成并且第一个描述符的 31 位置位。 |
| Register6 (Operation Mode Register) Offset: 0x18 | | | |
| Reserved<br>保留 | 31:27 | 0x0 | 保留 |
| DT<br>关闭丢弃 TCP/IP Checksum 错误以太网帧的功能 | 26 | 0x0 | 此位为 1 时 GMAC 将不丢弃 checksum 错误的以太网帧。 |
| RSF:Receive Store and Forward<br>接收存储转发 | 25 | 0x0 | 此位为 1 时 MTL 模块只接收已经全部存储在接收 FIFO 中的以太网帧。 |
| DFF:Disable Flushing of Received Frames<br>关闭冲刷接收的以太网帧的功能 | 24 | 0x0 | 此位为 1 时，接收 DMA 在接收描述符或者接收缓存不可用时不冲刷任何以太网帧。 |
| RFA[2]:MSB of Threshold for Activating Flow Control<br>激活流控阈值 | 23 | 0x0 | 100：最大值减去 5KB<br>101：最大值减去 6KB<br>110：最大值减去 7KB<br>111：保留<br>（注：最大值为 8KB） |

| | | | |
|---|---|---|---|
| RFD[2]:MSB of Threshold for Deactivating Flow Control<br>关闭流控阈值 | 22 | 0x0 | 100：最大值减去 5KB<br>101：最大值减去 6KB<br>110：最大值减去 7KB<br>111：保留<br>（注：最大值为 8KB） |
| TSF:Transmit Store and Forward<br>发送存储转发 | 21 | 0x0 | 此位为 1 时，帧的发送只在帧的内容已经全部进入 MTL 的传输 FIFO 中。 |
| FTF:Flush Transmit FIFO<br>冲刷传输 FIFO | 20 | 0x0 | 此位为 1 时，传输控制逻辑复位为默认值，并且会导致发送 FIFO 里面的数据全部丢失。 |
| Reserved | 19:17 | 0x0 | 保留 |
| TTC:Transmit Threshold Control<br>传输阈值控制 | 16:14 | 0x0 | 当帧大小超过此值时 MTL 将会传输该帧。<br>000: 64 字节<br>001: 128 字节<br>010: 192 字节<br>011: 256 字节<br>100: 40 字节<br>101: 32 字节<br>110: 24 字节<br>111: 16 字节 |
| ST:Start/Stop Transmission Command<br>开始/停止传输命令 | 13 | 0x0 | 此位为 1，传输进入运行状态。<br>此位为 0，传输进入停止状态。 |
| RFD:Threshold for deactivating flow control<br>关闭流控阈值 | 12:11 | 0x0 | 00: 最大值减去 1KB<br>01: 最大值减去 2KB<br>10: 最大值减去 3KB<br>11: 最大值减去 4KB<br>（最大值为 8KB） |
| RFA:Threshold for Activating flow | 10:9 | 0x0 | 00: 最大值减去 1KB<br>01: 最大值减去 2KB |

| | | | |
|---|---|---|---|
| control<br>激活流控阈值 | | | 10: 最大值减去 3KB<br>11: 最大值减去 4KB<br>（最大值为 8KB） |
| EFC:Enable HW flow control<br>使能硬件流控 | 8 | 0x0 | 此位为 1 时，基于接收 FIFO 利用率的硬件流控电路生效。 |
| FEF:Forward Error Frames<br>传输错误帧 | 7 | 0x0 | 此位为 1 时，接收错误帧（错误帧包括：CRC 错误，冲突错误，巨帧，看门狗超时，溢出等） |
| FUF:Forward Undersized Good Frames<br>接收无错误的小帧 | 6 | 0x0 | 此位为 1 时，接收 FIFO 将会接收没有错误但小于 64 字节的以太网帧。 |
| Reserved | 5 | 0x0 | 保留 |
| RTC:Receive Threshold Control<br>接收阈值控制 | 4:3 | 0x0 | MTL 传输接收 FIFO 中帧内容已经超过此项设置大小。<br>00: 64 字节<br>01:32 字节<br>10: 96 字节<br>11: 128 字节 |
| OSF:Operate on Second Frame<br>是否操作第二个以太网帧 | 2 | 0x0 | 此位为高时，DMA 在第一个以太网帧的状态尚未写回时即可以开始处理第二个以太网帧。 |
| SR:Start/Stop Receive<br>开始/停止接收 | 1 | 0x0 | 此位设置为高时，接收进入运行状态。<br>此位设置为低时，接收进入停止状态。 |
| Reserved<br>保留 | 0 | 0x0 | 保留 |
| Register7 (Interrupt Enable Register) Offset: 0x1C | | | |
| Reserved<br>保留 | 31:17 | 0x0 | 保留 |
| NIE:Normal | 16 | 0x0 | 此位为 1 时：正常中断使能 |

| | | | |
|---|---|---|---|
| Interrupt Summary Enable<br>正常中断汇总使能 | | | 此位为 0 时：正常中断不使能 |
| AIE:Abnormal Interrupt Summary Enable<br>非正常中断汇总使能 | 15 | 0x0 | 此位为 1 时：非正常中断使能。<br>此位为 0 时：非正常中断不使能。 |
| ERE：Early Receive Interrupt Enable<br>早期接收中断使能 | 14 | 0x0 | 此位为高时：早期接收中断使能 |
| FBE:Fatal Bus Error Enable<br>总线致命错误中断使能 | 13 | 0x0 | 此位为高时：总线致命错误中断使能。 |
| Reserved<br>保留 | 12:11 | 0x0 | 保留 |
| ETE:Early Transmit Interrupt Enable<br>早期传输中断使能 | 10 | 0x0 | 此位为高时：使能早期传输中断 |
| RWE:Receive Watchdog Timeout Enable<br>接收看门狗超时中断使能 | 9 | 0x0 | 此位为高时：使能接收看门狗超时中断 |
| RSE:Receive Stopped Enable<br>接收停止中断使能 | 8 | 0x0 | 此位为高时：使能接收停止中断。 |
| RUE:Receive Buffer Unavailable Enable<br>接收缓冲区不可用中断使能 | 7 | 0x0 | 此位为高时：使能接收缓冲区不可用中断。 |
| RIE:Receive | 6 | 0x0 | 此位为高时：使能接收完成中断 |

| | | | |
|---|---|---|---|
| Interrupt Enable<br>接收中断使能 | | | |
| UNE:Underflow Interrupt Enable<br>传输 FIFO 下溢中断使能 | 5 | 0x0 | 此位为高时：使能传输 FIFO 下溢中断 |
| OVE:Overflow Interrupt Enable<br>接收 FIFO 上溢中断使能 | 4 | 0x0 | 此位为高时：使能接收 FIFO 上溢中断。 |
| TJE:Transmit Jabber Timeout Enable<br>传输 Jabber 超时中断使能 | 3 | 0x0 | 此位为高时：使能 Jabber 超时中断。 |
| TUE:Transmit Buffer Unavailable Enable<br>传输缓存不可用中断使能 | 2 | 0x0 | 此位为高时：使能传输缓存不可用中断。 |
| TSE:Transmit Stopped Enable<br>传输停止中断使能 | 1 | 0x0 | 此位为高时：使能传输停止中断。 |
| TIE:Transmit Interrupt Enable<br>传输完成中断使能 | 0 | 0x0 | 此位为高时：使能传输完成中断。 |
| Register8 (Missed Frame and Buffer Overflow Counter Register)　Offset: 0x20 | | | |
| Reserved<br>保留 | 31:29 | 0x0 | 保留 |
| Overflow bit for FIFO Overflow Counter<br>FIFO 溢出指示位 | 28 | 0x0 | FIFO 溢出指示位 |

| | | | |
|---|---|---|---|
| Indicates the number of frames missed by the application<br>应用程序丢失的帧个数 | 27:17 | 0x0 | 指示应用程序丢失帧的个数 |
| Overflow bit for Missed Frame Counter<br>丢失帧个数溢出指示 | 16 | 0x0 | 提示丢失帧个数已经超过计数的最大值。 |
| Indicates the number of frames missed by the controller due to the Host Receive Buffer being unavailable<br>因为主机接收缓存不可用导致帧丢失的个数 | 15:0 | 0x0 | 指示因为主机接收缓存不可用导致帧丢失个数的计数。 |
| Register18 (Current Host Transmit Descriptor Register)　Offset: 0x48 | | | |
| Host Transmit Descriptor Address Pointer<br>当前发送描述符主机地址指针 | 31:0 | 0x0 | 只读 |
| Register19 (Current Host Receive Descriptor Register)　Offset: 0x4C | | | |
| Host Receive Descriptor Address Pointer<br>当前接收描述符主机地址指针 | 31:0 | 0x0 | 只读 |
| Register20 (Current Host Transmit Buffer Address Register)　Offset: 0x50 | | | |

| | | | |
|---|---|---|---|
| Host Transmit Buffer Address Pointer<br>当前传输缓冲区主机地址指针 | 31:0 | 0x0 | 只读 |
| Register21 (Current Host Receive Buffer Address Register) Offset: 0x54 | | | |
| Host Receive Buffer Address Pointer<br>当前接收缓冲区主机地址指针 | 31:0 | 0x0 | 只读 |

## 9.3 GMAC 控制器寄存器描述

| 参数名称 | 位 | 缺省值 | 范围 | 描述 |
|---|---|---|---|---|
| Register0 (MAC Configuration Register) Offset: 0x0000 | | | | |
| Reserved<br>保留 | 31:26 | 0x0 | | 保留 |
| TC: Transmit Configuration in RGMII<br>使能 RGMII 链路信息传输 | 24 | 0x0 | | 此位为高时，将会把双工模式，链路速度，链路以及链路连接/断开等信息通过 RGMII 接口传输给 PHY。 |
| WD: Watchdog Disable<br>关闭看门狗 | 23 | 0x0 | | 此位为高时，GMAC 将关闭接收端的看门狗定时器，可以接收最大 16384 字节的以太网帧。 |
| JD: Jabber Disable<br>关闭 Jabber 定时器 | 22 | 0x0 | | 此位为高时，GMAC 关闭发送过程中的 Jabber 定时器，可以发送最大 16384 字节的以太网帧。 |
| BE: Frame Burst Enable<br>帧突发传输使能 | 21 | 0x0 | | 此位为高时，GMAC 使能传输过程中的帧突发传输模式。 |
| JE: Jumbo Frame Enable<br>巨帧使能 | 20 | 0x0 | | 此位为高时，GMAC 使能巨帧（最大 9018 字节）的接收。 |

| | | | |
|---|---|---|---|
| IFG: Inter-Frame Gap<br>最小帧间距 | 19:17 | 0x0 | 设置传输过程中的最小帧间距。<br>000: 96 位时间<br>001: 88 位时间<br>010: 80 位时间<br>….<br>111: 40 位时间 |
| DCRS: Disable Carrier Sense During Transmission<br>传输过程中关闭载波冲突检测 | 16 | 0x0 | 此位为高时，MAC 忽略半双工模式下 CRS 信号的检测。 |
| PS: Port Select<br>端口选择 | 15 | 0x0 | 0: GMII (1000Mbps)<br>1: MII (10/100Mbps) |
| FES: Speed<br>快速以太网速度提示 | 14 | 0x0 | 0: 10Mbps<br>1: 100Mbps |
| DO: Disable Receive Own<br>关闭接收自己发出的以太网帧 | 13 | 0x0 | 此位为高时，GMAC 不接收半双工模式下 gmii_txen_o 有效的以太网帧。 |
| LM: Loopback Mode<br>使能环回模式 | 12 | 0x0 | 此位为高时，GMII/MII 工作在环回模式下。 |
| DM: Duplex Mode<br>使能全双工模式 | 11 | 0x0 | 此位为高时，GMAC 工作在全双工模式下，在全双工模式下可以同时发送和接收以太网帧。 |
| IPC: Checksum Offload<br>校验和卸载使能 | 10 | 0x0 | 此位为高时，GMAC 硬件计算接收到以太网帧的负载（payload）。还检查 IPV4 头的校验和是否正确。 |
| DR: Disable Retry<br>关闭重传 | 9 | 0x0 | 此位为高时，GMAC 在遇到冲突时不重传发送冲突的以太网帧，而只报告冲突错误。 |
| LUD: Link Up/Down<br>链路连接/链路断开 | 8 | 0x0 | 0：链路断开<br>1：链路连接 |

| | | | |
|---|---|---|---|
| ACS: Automatic Pad/CRC Stripping<br>以太网帧 Pad/CRC 自动去除 | 7 | 0x0 | 此位为 1 时，GMAC 中去除接收到的以太网帧的 Pad 和 FCS。 |
| BL: Back-Off Limit<br>回退限制 | 6:5 | 0x0 | 回退限制决定基于 slot 的延迟时间。<br>00: k=min(n,10)<br>01: k=min(n,8)<br>10: k=min(n,4)<br>11: k=min(n,1) |
| DC: Deferral Check<br>Deferral 检查 | 4 | 0x0 | 此位为 1 时，使能 deferral 检测功能。 |
| TE: Transmitter Enable<br>传输使能 | 3 | 0x0 | 此位为 1 时，使能 GMAC 传输功能。 |
| RE: Receiver Enable<br>接收使能 | 2 | 0x0 | 此位为 1 时，使能 GMAC 接收功能。 |
| Reserved | 1:0 | 0x0 | 保留。 |
| Register1 (MAC Frame Filter) Offset: 0x0004 | | | |
| RA: Receive All<br>接收全部 | 31 | 0x0 | 此位为 1 时，GMAC 接收模块把接收到的所有帧都发给应用程序，忽略源地址/目标地址过滤机制。 |
| Reserved<br>保留 | 30:11 | 0x0 | 保留 |
| HPF: Hash or Perfect Filter<br>哈希或者完全过滤 | 10 | 0x0 | 此位为 1 时，在哈希/完全过滤机制中匹配的以太网帧发送给应用。<br>此位为 0 时，只有在哈希过滤机制中匹配的以太网帧才发送给应用。 |
| SAF: Source Address Filter Enable<br>源地址过滤使能 | 9 | 0x0 | GMAC CORE 比较比较接收到以太网帧的源地址域和在 SA 寄存器中的值，如果匹配，接收状态寄存器中的 SAMatch 位设置为高。如果此位为 1，源地址匹配失败，GMAC CORE 将丢弃该以太网帧。<br>如果此位为 0，不管源地址匹配结果 GMAC CORE 都接收此帧，而匹配结果写入接收状态寄存器。 |

| | | | |
|---|---|---|---|
| SAIF: SA Inverse Filtering<br>源地址反转过滤 | 8 | 0x0 | 此位为 1 时，和 SA 寄存器中源地址匹配的以太网帧将会标记为源地址匹配失败。<br>此位为 0 时，和 SA 寄存器中源地址不匹配的以太网帧将会标记为源地址匹配失败。 |
| PCF: Pass Control Frames<br>接收控制帧 | 7:6 | 0x0 | 00: GMAC 过滤所有控制帧<br>01: GMAC 接收除了 pause 帧以外的所有控制帧。<br>10: GMAC 接收所有控制帧。<br>11: GMAC 根据地址过滤情况接收控制帧 |
| DBF: Disable Broadcast Frames<br>关闭广播帧 | 5 | 0x0 | 此位为 1 时，过滤所有接收的广播帧。<br>此位为 0 时，接收所有广播帧。 |
| PM: Pass All Multicast<br>接收所有多播帧 | 4 | 0x0 | 此位为 1 时，接收所有多播帧。<br>此位为 0 时，过滤所有多播帧。 |
| DAIF: DA Inverse Filtering<br>目标地址反转过滤 | 3 | 0x0 | 此位为 1 时，对单播和多播帧进行反向目标地址匹配。<br>此位为 0 时，对单播和多播帧进行正常目标地址匹配。 |
| HMC: Hash Multicast<br>哈希多播过滤 | 2 | 0x0 | 此位为 1 时，对接收到的多播帧根据哈希表的内容进行目标地址过滤。 |
| HUC: Hash Unicast<br>哈希单播过滤 | 1 | 0x0 | 此位为 1 时，对接收到的单播帧根据哈希表的内容进行目标地址过滤。 |
| PR: Promiscuous Mode<br>混杂模式 | 0 | 0x0 | 接收所有以太网帧。 |
| Register2 (Hash Table High Register) Offset: 0x0008 | | | |
| HTH: Hash Table High<br>哈希表高位 | 31:0 | 0x0 | 哈希表的高 32 位。 |
| Register3 (Hash Table Low Register) Offset: 0x000C | | | |
| HTL: Hash Table Low | 31:0 | 0x0 | 哈希表的低 32 位。 |

| | | | |
|---|---|---|---|
| 哈希表低位 | | | |
| Register4 (GMII Address Register)　Offset: 0x0010 | | | |
| Reserved<br>保留 | 31:16 | 0x0 | 保留 |
| PA: Physical Layer Address<br>PHY 地址 | 15:11 | 0x0 | 此域选择需要访问 32 个 PHY 中的哪个。 |
| GR: GMII Register<br>需要访问的 PHY 设备中的寄存器 | 10:6 | 0x0 | 此域选择需要访问的的 PHY 的哪个 GMII 配置寄存器。 |
| Reserved<br>保留 | 5 | 0x0 | 保留 |
| CR: CSR Clock Range<br>CSR 时钟范围 | 4:2 | 0x0 | 此域决定 MDC 时钟是 clk_csr_i 时钟频率比例。<br>0000 clk_csr_i/42<br>0001 clk_csr_i/62<br>0010 clk_csr_i/16<br>0011 clk_csr_i/26<br>0100 clk_csr_i/102<br>0101 clk_csr_i/124<br>0110, 0111 Reserved |
| GW: GMII Write<br>GMII 写 | 1 | 0x0 | 此位为 1 时，通过 GMII 数据寄存器对 PHY 进行写操作<br>此位为 0 时，通过 GMII 数据寄存器对 PHY 进行读操作。 |
| GB: GMII Busy<br>GMII 忙 | 0 | 0x0 | 对寄存器 4 和寄存器 5 写之前，此位应为 0。在写寄存器 4 之前此位必须先置 0。在访问 PHY 的寄存器时，应用程序需要将此位设置为 1，表示 GMII 接口上有写或者读操作正在进行。 |
| Register5 (GMII Data Register)　Offset: 0x0014 | | | |
| Reserved<br>保留 | 31:16 | 0x0 | 保留 |
| GD: GMII Data | 15:0 | 0x0 | 此域保存了对 PHY 进行管理读访问操作的 16 位数据，或者对 PHY 进行管理写访问的 16 位数据。 |

| | | | |
|---|---|---|---|
| GMII 数据 | | | |
| Register6 (Flow Control Register)　Offset: 0x0018 | | | |
| PT: Pause Time<br>暂停时间 | 31:16 | 0x0 | 此域保存了需要填入传输控制帧中的暂停时间域。 |
| Reserved<br>保留 | 15:8 | 0x0 | 保留 |
| DZPQ: Disable Zero-Quanta Pause<br>禁止零时间片暂停帧 | 7 | 0x0 | 此位为 1 时，禁止自动零时间片的暂停控制帧的产生。 |
| Reserved<br>保留 | 6 | 0x0 | 保留 |
| PLT: Pause Low Threshold<br>暂停帧的低阈值 | 5:4 | 0x0 | 此域用于设置暂停时间的阈值。<br>00: 暂停时间减少 4 个时间槽<br>01: 暂停时间减少 28 个时间槽<br>10: 暂停时间减少 144 个时间槽<br>11: 暂停时间减少 256 个时间槽<br>（一个时间槽为在 GMII/MII 接口上传输 512 比特或者 64 字节的时间） |
| UP: Unicast Pause Frame Detect<br>单播的暂停帧探测 | 3 | 0x0 | 此位为 1 时，GMAC 将会根据 MAC 地址 0 指定的本站单播地址来探测暂停帧。 |
| RFE: Receive Flow Control Enable<br>接收流控使能 | 2 | 0x0 | 此位为 1 时，GMAC 将会解析接收到的暂停帧，并且按照暂停帧指定的时间暂停帧的发送。 |
| TEF: Transmit Flow Control Enable<br>发送流控使能 | 1 | 0x0 | 在全双工模式下，此位为 1 时，GMAC 使能暂停帧的发送。<br>在半双工模式下，此位为 1 时，GMAC 使能反压操作。 |
| FCB/BPA: Flow Control Busy/Backpressure Activate | 0 | 0x0 | 此位为 1 时，在全双工模式下发起暂停控制帧的发送或在半双工模式下启动反压操作。 |

| | | | |
|---|---|---|---|
| 流控忙/反压激活 | | | |
| Register7 (VLAN Tag Register) Offset: 0x001C | | | |
| Reserved<br>保留 | 31:17 | 0x0 | 保留 |
| ETV: Enable 12-Bit VLAN Tag Comparison<br>使能 12 位 VLAN Tag 比较 | 16 | 0x0 | 此位为 1 时，使用 12 位 VLAN Tag 而不是使用 16 位 VLAN Tag 用于以太网帧比较和过滤。 |
| VL: VLAN Tag Identifier for Receive Frames<br>帧接收的 VLAN Tag 标识 | 15:0 | 0x0 | 此域保存 802.1Q 格式的 VLAN Tag，用于比较接收到的以太网帧的位于第 15 和第 16 个字节的 VLAN Tag。 |
| Register8 (Version Register) Offset: 0x0020 | | | |
| Reserved<br>保留 | 15:8 | 0x0 | 保留 |
| Version<br>版本号 | 7:0 | 0x0 | 0x35 |
| Register14 (Interrupt Status Register) Offset: 0x0038 | | | |
| Reserved<br>保留 | 15:8 | 0x0 | 保留 |
| MMC Receive Checksum Offload Interrupt Status<br>MMC 接收校验和卸载状态中断 | 7 | 0x0 | MMC 校验和卸载寄存器产生任何中断产生时，此位设置为 1。 |
| MMC Transmit Interrupt Status<br>MMC 传输中断 | 6 | 0x0 | MMC 传输中断寄存器产生任何中断时，此位设置为 1。 |
| MMC Receive Interrupt Status<br>MMC 接收中断状态 | 5 | 0x0 | MMC 接收中断寄存器产生任何中断时，此位设置为 1。 |

| | | | |
|---|---|---|---|
| MMC Interrupt Status<br>MMC 中断状态 | 4 | 0x0 | 7:5 的任何位为高时，此位设置为 1。 |
| PMT Interrupt Status<br>电源管理中断状态 | 3 | 0x0 | 在 Power Down 状态下，收到 magic 帧或在 Wake-on-LAN 帧时，此位设置为 1。 |
| PCS Auto-Negotiation Complete<br>PCS 自动协商完成 | 2 | 0x0 | RGMII PHY 接口自动协商完成时，此位设置为 1。 |
| PCS Link Status Changed<br>PCS 链路状态变化 | 1 | 0x0 | RGMII PHY 接口的链路状态发生任何变化时，此位设置为 1。 |
| RGMII Interrupt Status<br>RGMII 中断状态 | 0 | 0x0 | RGMII 接口的链路状态发生任何变化时，此位设置为 1。 |
| Register15 (Interrupt Mask Register) Offset: 0x003C | | | |
| Reserved<br>保留 | 15:10 | 0x0 | 保留 |
| Time Stamp Interrupt Mask<br>时间戳中断使能 | 9 | 0x0 | 此位为 1 时，禁止时间戳发生的中断 |
| Reserved<br>保留 | 8:4 | 0x0 | 保留 |
| PMT Interrupt Mask<br>电源管理中断使能 | 3 | 0x0 | 此位为 1 时，禁止电源管理引起的中断。 |
| PCS AN Completion Interrupt Mask<br>PCS 自动协商完成中断使能 | 2 | 0x0 | 此位为 1 时，禁止 PCS 自动协商完成中断。 |
| PCS Link Status Interrupt Mask | 1 | 0x0 | 此位为 1 时，禁止由于 PCS 链路状态变化引起的中断。 |

| | | | |
|---|---|---|---|
| PCS 链路状态中断使能 | | | |
| RGMII Interrupt Mask<br>RGMII 中断使能 | 0 | 0x0 | 此位为 1 时，禁止 RGMII 引起的中断。 |
| Register16 (MAC Address0 High Register) Offset: 0x0040 | | | |
| MO: Always 1<br>保留 | 31 | 0x0 | 保留 |
| Reserved<br>保留 | 30:16 | 0x0 | 保留 |
| MAC Address0[47:32]<br>MAC 地址高 16 位 | 15:0 | 0x0 | 存放用于接收地址过滤和传输流控帧的 MAC 地址。 |
| Register17 (MAC Address0 Low Register) Offset: 0x0044 | | | |
| MAC Address0[31:0]<br>MAC 地址低 32 位 | 31:0 | 0x0 | 存放用于接收地址过滤和传输流控帧的 MAC 地址。 |
| Register18 (MAC Address1 High Register) Offset: 0x0048 | | | |
| AE: Address Enable<br>地址使能 | 31 | 0x0 | 此位为 1 时，地址过滤模块使用第 2 个 MAC 地址用于完全地址过滤。此位为 0 时，地址过滤模块不使用第 2 个 MAC 地址用于地址过滤。 |
| SA: Source Address<br>源 MAC 地址 | 30 | 0x0 | 此位为 1 时，MAC 地址 1 用于比较接收帧的源 MAC 地址。<br>此位为 0 时，MAC 地址 1 用于比较接收帧的目标 MAC 地址。 |
| MBC: Mask Byte Control<br>掩模字节控制 | 29:24 | 0x0 | 此域用于比较每个 MAC 地址的字节掩模控制位。比如第 29 位用于掩码寄存器 18 的[15:8]这个字节。 |
| Reserved<br>保留 | 23:16 | 0x0 | 保留。 |
| MAC Address1[47:32]<br>第 2 个 MAC 地址的 | 15:0 | 0xFFFF | |

| | | | |
|---|---|---|---|
| 高 16 位 | | | |
| Register19 (MAC Address1 Low Register)　Offset: 0x004C | | | |
| MAC Address1[31:0]<br>第 2 个 MAC 地址的低 32 位 | 31:0 | 0x0 | |
| Register48 (AN Control Register)　Offset: 0x00C0 | | | |
| Reserved<br>保留 | 31:19 | 0x0 | 保留 |
| SGMII RAL Control<br>保留 | 18 | 0x0 | 保留 |
| LR: Lock to Reference<br>锁定到参考时钟 | 17 | 0x0 | 此位为 1 时，PHY 将其锁相环锁定到 125MHz 的参考时钟。 |
| ECD: Enable Comma Detect<br>使能停顿探测 | 16 | 0x0 | 此位为 1 时，使能 PHY 的停顿探测和字重同步。 |
| Reserved<br>保留 | 15 | 0x0 | 保留 |
| ELE: External Loopback Enable<br>外部环回使能 | 14 | 0x0 | 此位为 1 时，使能 PHY 进入环回模式。 |
| Reserved<br>保留 | 13 | 0x0 | 保留 |
| ANE: Auto-Negotiation Enable<br>自动协商使能 | 12 | 0x0 | 此位为 1 时，GMAC 将会和链路对方进行自动协商。 |
| Reserved<br>保留 | 11:10 | 0x0 | 保留 |
| RAN: Restart Auto-Negotiation | 9 | 0x0 | 此位为 1 时，重新进行自动协商。 |

| | | | |
|---|---|---|---|
| 重新进行自动协商 | | | |
| Reserved<br>保留 | 8:0 | 0x0 | 保留 |
| Register49 (AN Status Register)　Offset: 0x00C4 | | | |
| Reserved<br>保留 | 31:9 | 0x0 | 保留 |
| ES: Extended Status<br>扩展状态 | 8 | 0x0 | 只读，因为 GMAC 支持扩展状态信息。 |
| Reserved<br>保留 | 7:6 | 0x0 | 保留 |
| ANC: Auto-Negotiation Complete<br>自动协商完成 | 5 | 0x0 | 只读，指示自动协商完成。 |
| Reserved<br>保留 | 4 | 0x0 | 保留 |
| ANA: Auto-Negotiation Ability<br>自动协商能力 | 3 | 0x0 | 只读，因为 GMAC 支持自动协商。 |
| LS: Link Status<br>链路状态 | 2 | 0x0 | 此位为 1 时，指示链路连接上。<br>此位为 0 时，指示链接未连接。 |
| Reserved<br>保留 | 1:0 | 0x0 | 保留。 |
| Register50 (Auto-Negotiation Advertisement Register)　Offset: 0x00C8 | | | |
| Reserved<br>保留 | 31:16 | 0x0 | 保留 |
| NP: Next Page Support<br>下一页面支持 | 15 | 0x0 | 只读为 0，因为 GMAC 不支持下一页面。 |
| Reserved | 14 | 0x0 | 保留 |

| | | | |
|---|---|---|---|
| 保留 | | | |
| RFE: Remote Fault Encoding<br>远端错误编码 | 13:12 | 0x0 | 此 2 位指示链路对端发生错误，具体编码将 IEEE 802.3z 第 37.2.1.5 小节。 |
| Reserved<br>保留 | 11:9 | 0x0 | 保留 |
| PSE: Pause Encoding<br>Pause 位编码 | 8:7 | 0x0 | 见 IEEE 802.3z 第 37.2.1.4 小节 |
| HD: Half-Duplex<br>半双工 | 6 | 0x0 | 此位为 1 时，指示 GMAC 支持半双工。 |
| FD: Full-Duplex<br>全双工 | 5 | 0x0 | 此位为 1 时，指示 GMAC 支持全双工。 |
| Reserved<br>保留 | 4:0 | 0x0 | 保留 |
| Register51 (Auto-Negotiation Link Partner Ability Register)　Offset: 0x00CC | | | |
| Reserved<br>保留 | 31:16 | 0x0 | 保留 |
| NP: Next Page Support<br>下一页面支持 | 15 | 0x0 | 此位为 1 时，指示有更多下一页面信息可用<br>此位为 0 时，指示下一页面交换不可用。 |
| ACK: Acknowledge<br>确认 | 14 | 0x0 | 指示在自动协商中，链路对端成功接收到 GMAC 的基本页面。 |
| RFE: Remote Fault Encoding<br>远端错误编码 | 13:12 | 0x0 | 见 IEEE 802.3z 第 37.2.1.5 小节。 |
| Reserved<br>保留 | 11:9 | 0x0 | 保留 |
| PSE: Pause Encoding<br>对端 pause 状态编码 | 8:7 | 0x0 | 见 IEEE 802.3z 第 37.2.14 小节。 |

| | | | |
|---|---|---|---|
| HD: Half-Duplex<br>半双工 | 6 | 0x0 | 指示对端可以运行在半双工模式。 |
| FD: Full-Duplex<br>全双工 | 5 | 0x0 | 指示对端可以运行在全双工模式。 |
| Reserved<br>保留 | 4:0 | 0x0 | 保留 |
| Register52 (Auto-Negotiation Expansion Register)　Offset: 0x00D0 | | | |
| Reserved<br>保留 | 31:3 | 0x0 | 保留 |
| NPA: Next Page Ability<br>下一页面能力 | 2 | 0x0 | 只读为 0，因为 GMAC 不支持下一页面。 |
| NPR: New Page Received<br>接收到新页面 | 1 | 0x0 | 此位为 1 时，指示 GMAC 接收到新页面。 |
| Reserved<br>保留 | 0 | 0x0 | 保留 |
| Register54 (SGMII/RGMII Status Register)　Offset: 0x00D8 | | | |
| Reserved<br>保留 | 31:4 | 0x0 | 保留 |
| Link Status<br>链路状态 | 3 | 0x0 | 此位为 1 时，指示链路连接上。<br>此位为 0 时，指示链路未连接上。 |
| Link Speed<br>链路速度 | 2:1 | 0x0 | 指示链路当前速度<br>00: 2.5MHz<br>01: 25MHz<br>10: 125MHz |
| Link Mode<br>链路模式 | 0 | 0x0 | 0: 半双工<br>1: 全双工 |

## 9.4 DMA 描述符

DMA 描述符是 GMAC 驱动和硬件的交互接口，记录了数据包的内存地址和传输状态。在此分别定义了发送描述符(Tx Desciptor)和接收描述符(Rx

Descriptor)两种数据结构。两种描述符可以自由选择分别以环式(ring mode)或者链式(chain mode)相连，以供 GMAC 使用。

### 9.4.1 DMA 描述符的基本格式

每一个 DMA 描述符包含两个数据 buffer、两个字节计数 buffer 和两个指向数据 buffer 地址的指针。需要注意的是描述符的地址必须保证按照所连接的系统总线位宽对齐，同时保证与系统字节序相同(默认小尾端)。

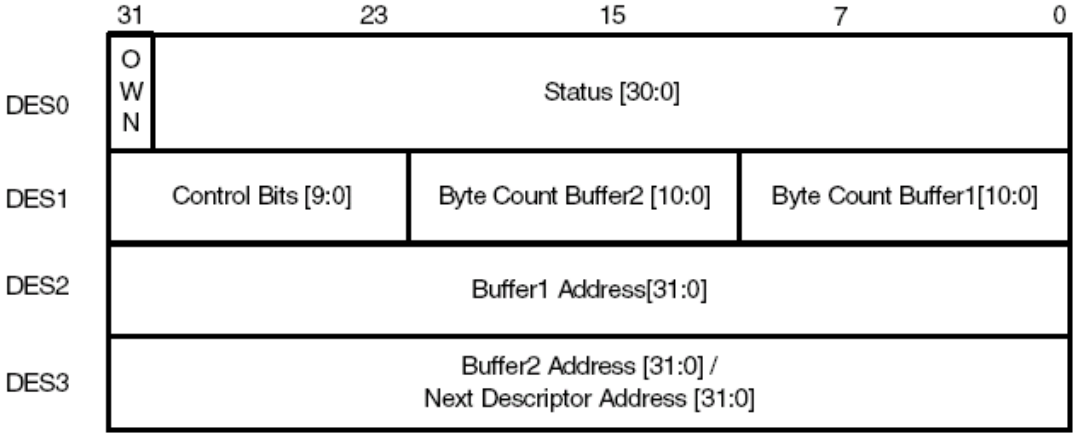



图 1　DMA 描述符的基本格式(小尾端 32 位总线)

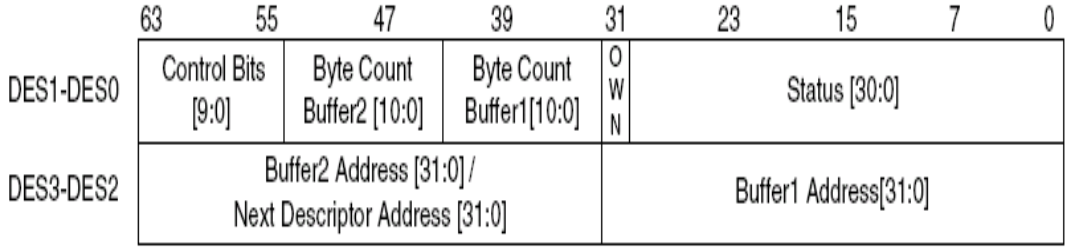



图 2　DMA 描述符的基本格式(小尾端 64 位总线)

### 9.4.2 DMA 接收描述符

GMAC 子系统在工作模式下需要至少两个接收描述符才能够正常的接收一个网络数据包。其内部的接收模块在处理一个网络数据包时，总是在同时尝试获取下一个接收描述符。每一个网络数据包被称为一个帧(frame)。

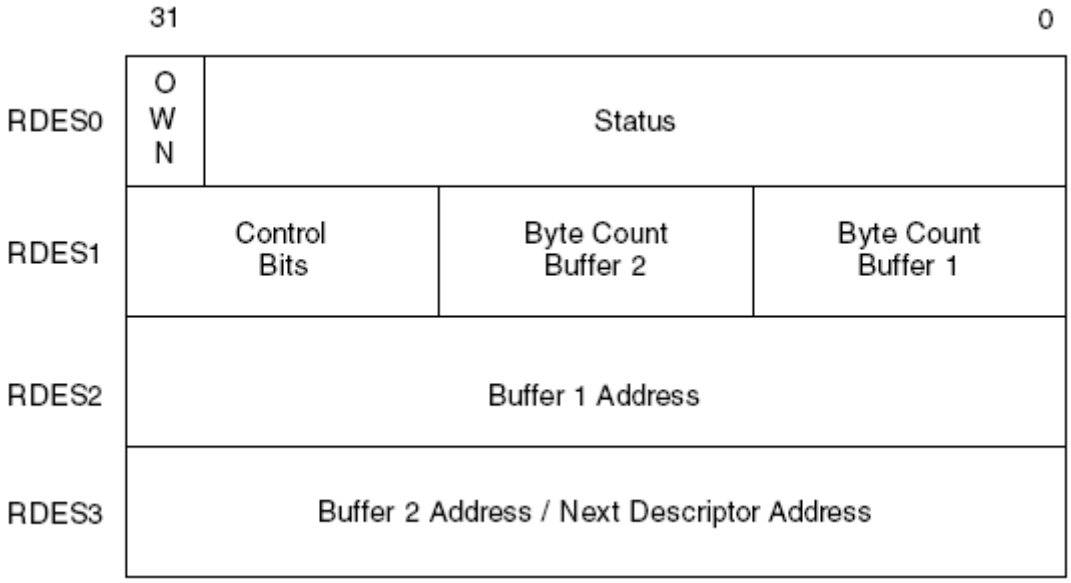



图 3 DMA 接收描述符的基本格式(小尾端 32 位总线)

### 9.4.3 RDES0

..... RDES0 包括了当前接收帧状态、长度以及该描述符的所有情况(主机或 DMA 拥有)。RDES0 的具体细节参见下表。

| RDES0 | 位 | |
|---|---|---|
| OWN<br>所有模式 | 31 | 该位为 1 时表示描述符当前属于 DMA 控制，0 表示属于主机控制。当 DMA 模块完成一次传输时，会将该位主动清 0 |
| AFM: Destination Address Filter Fai<br>目标地址过滤错误 I | 30 | 当该位为 1 时，表示当前数据帧目标地址不符合 GMAC 内部的帧目标地址过滤器 |
| FR: Frame length<br>帧长度 | 29:16 | 表示接收当前帧的长度，当 ES 位为 0 时有效 |
| ES: Error Summary<br>总体错误信息 | 15 | 指示当前帧是否出错，其值为 RDES[0]、RDES[1]、RDES[3]、RDES[4]、RDES[6]、RDES[7、RDES[11]、RDES[14]各位作或运算(OR)的结果 |
| DE: Descriptor Error<br>描述符错误 | 14 | 当该位为 1 时表示，当前描述符所指向的 buffer 与帧不相符或者 OWN 为 0(主机控制) |
| SAF: Source Address Filter Fail<br>源地址过滤错误 | 13 | 当该位为 1 时，表示当前数据帧的源地址不符合 GMAC 内部的帧源地址过滤器 |

| | | |
|---|---|---|
| LE: Length Error<br>长度错误 | 12 | 当该位为 1 时，表示当前接收帧长度与默认长度不符。当 Frame Type 位为 1 且 CRC Error 位为 0 时有效 |
| OE: Over Flow Error<br>溢出错误 | 11 | 当该位为 1 时，表示接收该帧时 GMAC 内部 RxFIFO 溢出 |
| VLAN: VLAN Tag<br>VLAN 标志 | 10 | 当该位为 1 时，表示该帧的类型为 VLAN |
| FS: First Desciptor<br>第一个描述符 | 9 | 当该位为 1 时，表示当前描述符所指向的 buffer 为当前接收帧的第一个保存 buffer |
| LS: Last Desciptor<br>最后一个描述符 | 8 | 当该位为 1 时，表示当前描述符所指向的 buffer 为当前接收帧的最后一个保存 buffer |
| IPC Checksum Error/Giant Frame<br>校验错误/超长帧 | 7 | 当该位为 1 时，如果 IPC 校验功能启用则表示当前帧的 IPv4 头校验值与帧内部校验域的值不相符。如果未启用则表示当前帧为一个超长帧(长度大于 1518 字节) |
| LC: late collision<br>后期冲突 | 6 | 当该位为 1 时，表示在半双工模式下，当前帧接收时发生了一个后期冲突 |
| FT: Frame Type<br>帧类型 | 5 | 当该位为 1 时，表示当前帧为一个以太网格式帧，为 0 时表示当前帧为一个 IEEE802.3 格式帧 |
| RWT: Receive Watchdog Timeout | 4 | 当该位为 1 时，表示当前时钟值超过了接收模块看门狗电路时钟的值，既接收帧超时 |
| RE: Receive Error<br>接收错误 | 3 | 当该位为 1 时，表示接收当前帧时内部模块出错。内部信号 rxer 置 1 且 rxdv 置 1 |
| DE: Dribble bit Error<br>奇数位错误 | 2 | 当该位为 1 时，表示接收帧长度不是整数，即总长度为奇数位，该位只有在 mii 模式下有效 |
| CE: CRC Error<br>接收 CRC 校验错误 | 1 | 当该位为 1 时，表示接收当前帧时内部 CRC 校验出错。该位只有在 last descriptor(RDES0[8])为 1 时有效 |
| RX MAC: Checksum/payload Checksum Error<br>接受校验/负载校验错误 | 0 | 当该位为 1 时，表示接收当前帧时内部 RX MAC 寄存器组 1-15 中存在一个匹配当前帧目的地址。为 0 时表示 RX MAC 寄存器组 0 匹配接受帧目的地址。如果 Full Checksum Offload Engine 启用时，为 1 表示该帧 TCP/UDP/ICMP 校验错误。该位为 1 时也可能表示当前帧实际接受长度与帧内部 |

| | | |
|---|---|---|
| | | 记载长度不相符。 |

### 9.4.4 RDES1

RDES1 记录了描述符所指向的 buffer 大小，以及描述符的组织格式(环形或链型)

| RDES1 | 位 | |
|---|---|---|
| Disable Intr in Completion<br>禁止完成后发中断 | 31 | 该位为 1 时表示该帧接收完成后将不会置起 STATUS 寄存器中 RI 位（CSR5[6]），这将会使得主机无法检测到该中断 |
| Reserved<br>保留 | 30:26 | |
| RER: Receive End of Ring<br>环型描述符结尾 | 25 | 该位为 1 时表示该描述符为环型描述符链表的最后一个，下一个描述符的地址为接收描述符链的基址 |
| RCH: Second Address Chained<br>第二个 buffer 地址指向下个链式描述符 | 24 | 该位为 1 时表示描述符中的第二个 buffer 地址指向的是下一个描述符的地址，为 0 时表示该地址指向第二个 buffer 地址<br>当该位为 1 时，RDES1[21-11]的值将没有意义，RDES1[25]比 RDES1[24]具有更高优先级(代表环型而不是链型) |
| Reserved<br>保留 | 23:22 | |
| RBS2: Receive Buffer Size 2<br>接收 buffer2 大小 | 21:11 | 该域表示数据 buffer2 的大小。根据系统总线的宽度 32/64/128，Buffer2 的大小应该为 4/8/16 的整数倍。如果不满足则会导致未知的结果。该域在 RDES1[24]为 0 时有效 |
| RBS2: Receive Buffer Size 1<br>接收 buffer1 大小 | 10:0 | 该域表示数据 buffer1 的大小。根据系统总线的宽度 32/64/128，Buffer1 的大小应该为 4/8/16 的整数倍。如果不满足则会导致未知的结果。该域一直有效。如果该域值为 0，DMA 则会自动访问 buffer2 或者下一个接收描述符 |

### 9.4.5 RDES2

该域记录了数据接收 buffer1 的地址。

| RDES2 | 位 | |
|---|---|---|
| Buffer1 Address Pointer | 31:0 | 该域记录了数据接收 buffer1 的 32 位物理地址。该物理地址没有默认的对齐要求。当 GMAC DMA 内部实现了总线数据 |

| | | |
|---|---|---|
| 接收 buffer1 地址 | | 32/64/128 位对齐，则该地址的低 2/3/4 位会被忽略 |

### 9.4.6 RDES3

该域记录了数据接收 buffer2 的地址。

| RDES3 | 位 | |
|---|---|---|
| Buffer2 Address Pointer<br>接收 buffer2 地址 | 31:0 | 该域记录了数据接收 buffer2 的 32 位物理地址。该物理地址没有默认的对齐要求。当 GMAC DMA 内部实现了总线数据 32/64/128 位对齐，则该地址的低 2/3/4 位会被忽略. 如果描述符是以链式连接，则该域记录的是下一个描述符的地址 |

### 9.4.7 DMA 发送描述符

发送描述符与接收描述符的格式基本相同。每个描述符的地址需要按照总线宽度(32/64/126 位)对齐。

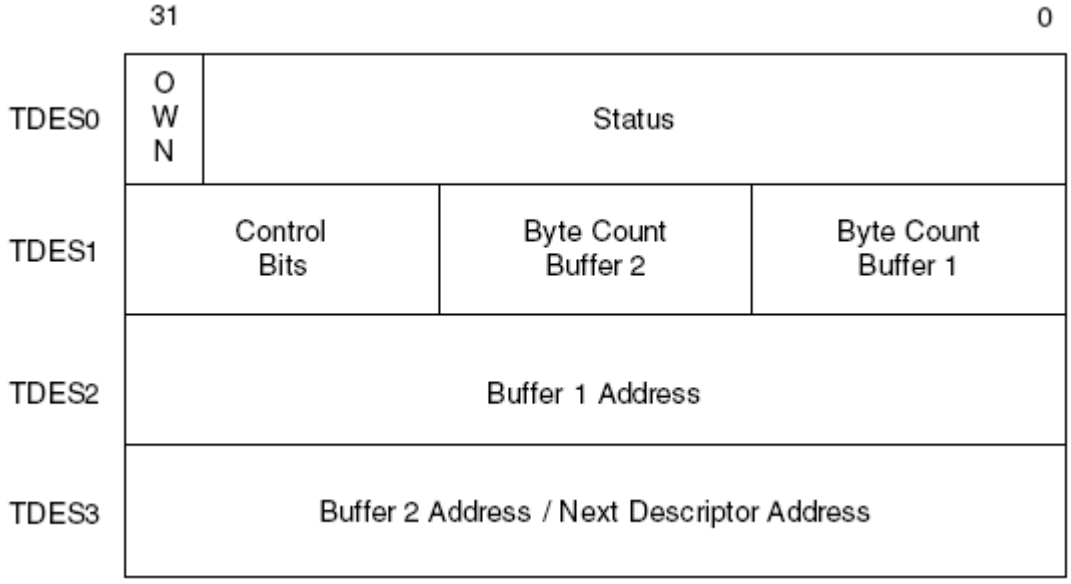



图 4 DMA 发送描述符的基本格式(小尾端 32 位总线)

### 9.4.8 TDES0

TDES0 包含了发送帧的状态和发送描述符的所属信息。

| TDES0 | 位 | |
|---|---|---|
| OWN<br>所属模式 | 31 | 该位为 1 时表示描述符当前属于 DMA 控制，0 表示属于主机控制。当 DMA 模块完成一次传输时，会将该位主动清 0 |
| Reserved<br>保留 | 30:18 | |
| TTSS: Tx Time Stamp Status<br>发送时间戳状态 | 17 | 当 IEEE1588 功能启用时，该位为 1 表示 TDES2 和 TDES3 中保存了该发送帧的时间戳信息。否则该位保留 |
| IHE: IP Header Error<br>IP 头错误 | 16 | 该位为 1 时表示内部校验模块发现该发送帧的 IP 头出错，并且不会对该域做任何修改 |
| ES: Error Summary<br>总体错误信息 | 15 | 指示当前帧是否出错，其值为 TDES[1]、TDES[2]、TDES[8]、TDES[9]、TDES[10]、TDES[11]、TDES[13]、TDES[14]各位作或运算(OR)的结果 |

| | | |
|---|---|---|
| JT: Jabber Timeout<br>Jabber 超时 | 14 | 该位为 1 时表示 GMAC 发送模块遇到了 Jabber 超时 |
| FF: Frame Flushed<br>帧刷新 | 13 | 该位为 1 时表示软件发出了一个刷新命令导致 DMA/MTL 将其内部的帧刷新掉 |
| PCE: Payload Checksum Error<br>负载校验错误 | 12 | 该位为 1 时表示内部负载校验模块再向发送帧中插入校验数据时出错。当负载校验模块启用时，该位有效 |
| LC: Loss of Carrier<br>载波丢失 | 11 | 该位为 1 时表示在发送该帧过程中载波丢失(gmii_crs 信号多个周期未置起) |
| NC: No Carrier<br>载波无效 | 10 | 该位为 1 时表示在发送过程中，PHY 的载波信号一直未置起 |
| LC: Late Collision<br>后期冲突 | 9 | 当该位为 1 时表示在半双工模式下，当前帧接收时发生了一个后期冲突 |
| EC: Excessive Collison<br>连续冲突 | 8 | 当该位为 1 时表示在发送当前帧的时候连续出现了 16 次冲突 |
| VF: VLAN Frame<br>VLAN 帧 | 7 | 该位为 1 时表示当前发送帧为一个 VLAN 帧 |
| CC: Collsion Count<br>冲突计数 | 6:3 | 该域表示当前帧在成功发送之前所遇到冲突次数的总数 |
| ED: Excessive Deferral<br>连续 Deferral | 2 | 该位为 1 时表示当前帧传输结束 |
| UF: Underflow Error<br>溢出错误 | 1 | 该位为 1 时表示当前帧传输时发生了溢出错误，即数据传输 buffer 过小或不可用 |
| DB: Defered Bit<br>帧刷新 | 0 | 该位为 1 时表示此次发送被延迟，只有在半双工模式下有效 |

### 9.4.9 TDES1

TDES1 包含了 buffer 大小以及其他一些控制描述符环型/链型连接的控制和状态位。

| TDES1 | 位 | |
|---|---|---|
| IC: Interrption on Complete<br>完成时中断 | 31 | 该位为1时表示该帧接发送完成后将会置起STATUS寄存器中TI位(CSR5[0]) |
| LS: Last Segment<br>最后段 | 30 | 该位为1时表示当前buffer包含的是一帧数据的最后一段(如果帧分为多个段) |
| FS: First Segment<br>第一段 | 29 | 该位为1时表示当前buffer包含的是一帧数据的第一段(如果帧分为多个段) |
| CIC: Checksum Insertion Control<br>校验数据填充控制 | 28:27 | 该域控制内部模块是否在发送帧中填充校验数据。<br>值：2'b00: 不填充校验数据<br>2'b01: 填充IPV4头校验数据<br>2'b10: 在伪头数据(pseudo-header)存在的情况下，填充TCP/UDP/ICMP全校验数据<br>2'b11: 总是填充TCP/UDP/ICMP的全校验数据 |
| DC: Disable CRC<br>禁止CRC校验 | 26 | 该位为1时GMAC硬件不在每个发送帧的结尾添加CRC校验数据 |
| TER: Transmit End of Ring<br>环形描述符结尾 | 25 | 该位为1时表示该描述符为环型描述符链表的最后一个，下一个描述符的地址为发送描述符链的基址 |
| TCH: Second Address Chained<br>第二个buffer地址指向下个链式描述符 | 24 | 该位为1时表示描述符中的第二个buffer地址指向的是下一个描述符的地址，为0时表示该地址指向第二个buffer地址<br>当该位为1时，TDES1[21-11]的值将没有意义，TDES1[25]比TDES1[24]具有更高优先级(代表环型而不是链型) |
| DP: Dissable Pading<br>禁止填充 | 23 | 该位为1时表示GMAC将不会对长度小于64字节的数据包进行空数据填充 |
| TTSE: Transmit Time Stamp Enable<br>启用发送时间戳 | 22 | 该位为1时表示将启用内部模块计算IEEE1588硬件时间戳计算，在TDES1[29]为1时有效 |
| TBS2: Transmit Buffer Size 2<br>发送buffer2大小 | 21:11 | 该域表示数据buffer2的大小。当TDES1[24]为1时，该域无效 |

| | | |
|---|---|---|
| TBS1: Transmit Buffer Size 1<br>发送 buffer1 大小 | 10:0 | 该域表示数据 buffer1 的大小。该域一直有效。如果该域值为 0，DMA 则会自动访问 buffer2 或者下一个接收描述符 |

### 9.4.10 TDES2

该域记录了数据发送 buffer1 的地址。

| TDES2 | 位 | |
|---|---|---|
| Buffer1 Address Pointer<br>发送 buffer1 地址 | 31:0 | 该域记录了数据接收 buffer1 的 32 位物理地址。该物理地址没有默认的对齐要求。当 GMAC DMA 内部实现了总线数据 32/64/128 位对齐，则该地址的低 2/3/4 位会被忽略 |

### 9.4.11 TDES3

该域记录了数据发送 buffer2 的地址。

| TDES3 | 位 | |
|---|---|---|
| Buffer2 Address Pointer<br>发送 buffer2 地址 | 31:0 | 该域记录了数据接收 buffer2 的 32 位物理地址。该物理地址没有默认的对齐要求。当 GMAC DMA 内部实现了总线数据 32/64/128 位对齐，则该地址的低 2/3/4 位会被忽略。如果描述符是以链式连接，则该域记录的是下一个描述符的地址 |

## 9.5 软件编程向导(Software Programming Guide):

**DMA 初始化：**

1. 软件重置(reset)GMAC
2. 等待重置完成(查询 DMA reg0[0])
3. 对 DMA reg0 的以下域进行编程
   a. MIX-BURST 和 AAL(DMA reg0[26]、[25])
   b. Fixed-burst 或者 undefined-burst(DMA reg0[16])
   c. Burst-length 和 Burst-mode
   d. Descriptor Length(只有当环形格式时有效)
   e. Tx 和 Rx 仲裁调度
4. 对 AXI Bus Mode Reg 进行编程
   a. 如果选择了 Fixed-burst，则需要在该寄存器内设置最大 burst length
5. 分别创建发送、接收描述符链，可以分别选择环形模式或者链型模式进行连接，并将接收描述符的 OWN 位设为 1(DMA 拥有)
6. 在软件启用 DMA 描述符之前，必须保证至少发送/接收描述符链中有三个描述符
7. 将发送、接收描述符链表的首地址写入 DMA reg3、4
8. 对 DMA reg6(DMA mode operation)中的以下位进行配置
   a. 接收/发送的 Store and Forward
   b. 接收/发送的阈值因子(Threshold Control)
   c. 启用流控制(hardware flow control enable)
   d. 错误帧和未识别的正确帧略过(forwarding enable)
   e. OSF 模式
9. 向 DMA reg6(Status reg)写 1.清除所有中断请求
10. 向 DMA reg7(interrupt enable reg)写 1，启用所有中断
11. 向 DMA reg6[1]、[13]中写 1，启用发送和接收 DMA

**MAC 初始化：**

1. 正确配置配套 PHY 芯片
2. 对 GMAC reg4(GMII Address Register)进行正确配置，使其能够正常访问 PHY 相关寄存器
3. 读取 GMAC reg5(GMII Data Register)获取当前 PHY 的链接(link)、速度(speed)、模式(双工)等信息
4. 配置 MAC 地址
5. 如果启用了 hash filtering，则需要对 hash filtering 进行配置
6. 对 GMAC reg1(Mac Frame filter)以下域进行配置，来进行帧过滤
7. 接收所有
8. 混杂模式(promiscuous mode)
9. 哈希或完美过滤(hash or perfect filter)
10. 组播、多播过滤设置等等
11. 对 GMAC reg6(Flow control register)以下域进行配置

a. 暂停时间和其他暂停控制位
b. 接收和发送流控制位
c. 流控制忙/后压力启用

12. 对中断掩码寄存器(Mac reg15)进行配置
13. 基于之前得到的线路信息(link,speed,mode)对 GMAC reg0 进行正确的配置
14. 设置 GMAC reg0[2]、[3]来启用 GMAC 中的发送、接收模块

**发送和接收的一般过程：**

检测到发送或接收中断后，查寻相应描述符来判断其是否属于主机，并读取描述符中的数据

完成对描述符中数据的读取后，将描述符各位清 0 并设置其 OWN 位，使其继续发送/接收数据

如果当前发送或接收描述符不属于 DMA(OWN=0)，则 DMA 模块会进入挂起状态。当有数据需要被发送或接收时，向 DMA Tx/Rx POLL 寄存器写 1 重新使能 DMA 模块。需要注意的是接收描述符在空闲时应该总是属于 DMA(OWN=1)

发送和接收描述符及对应 buffer 地址的实时信息可以通过查寻 DMA reg18、19、20、21 获得。

# 10 GMAC1

## 10.1 配置成 MAC 的连接和复用方式

GMAC1 控制器可以通过配置成百兆模式(MII)或千兆模式(RGMII)。如果外部连接百兆 PHY，需要复用 PWM2，PWM3 和 GMAC1_TX_CLK_O 三个 PAD，配置如下表：

| **PAD** | MAC 信号 | 配置位 | 复位值 |
|---|---|---|---|
| PWM2 | MAC_1_COL | GMA1_USE_PWM23 | 1’b0 |
| PWM3 | MAC_1_CRS | GMA1_USE_PWM23 | 1’b0 |
| GMAC1_TX_CLK_O | MAC_1_RX_ERR | GMA1_USE_TX_CLK | 1’b1 |

寄存器地址：0x1fd0_0420

| 配置位 | Bit 位 | 描述 |
|---|---|---|
| GMAC1_USE_PWM23 | 9 | 1：百兆模式 MAC_COL/MAC_CRS 分别复用 PWM2/3<br>0：千兆模式 |
| GMAC1_USE_TX_CLK | 11 | 1：MII_1_RX_ERR 输入信号复用 GMAC1_TX_CLK_O<br>0：千兆模式 |

百兆模式下，如果外部 PHY 不提供 RX_ERR 信号，GMAC1_TX_CLK_O 需接地，不能悬空；

百兆模式下，如果外部 PHY 提供 RX_ERR 信号， GMAC1_TX_CLK_O 与 RX_ERR 连接。

## 10.2 寄存器描述

GMAC 内部集成独有的 DMA 控制器，专门配合 GMAC 数据传输；该 DMA 控制器不能被其他模块使用，GMAC 也无法使用外部其他 DMA（见 17 章）。GMAC 寄存器包括 GMAC 寄存器部分和 DMA 寄存器部分。GMAC1 的 GMAC 寄存器的起始地址是 0x1fe2_0000；GMAC1 的 DMA 寄存器的起始地址是 0x1fe2_1000。

DMA 寄存器和 GMAC 寄存器的具体意义请参照第 9 章。

# 11 SATA

## 11.1 SATA 总体描述

本节介绍 SATA 的特性和总体结构：

SATA 的特性包括：

- 支持 SATA 1 代 1.5Gbps 和 SATA2 代 3Gbps 的传输
- 兼容串行 ATA 2.6 规范和 AHCI 1.1 规范

SATA 的结构图如下所示：

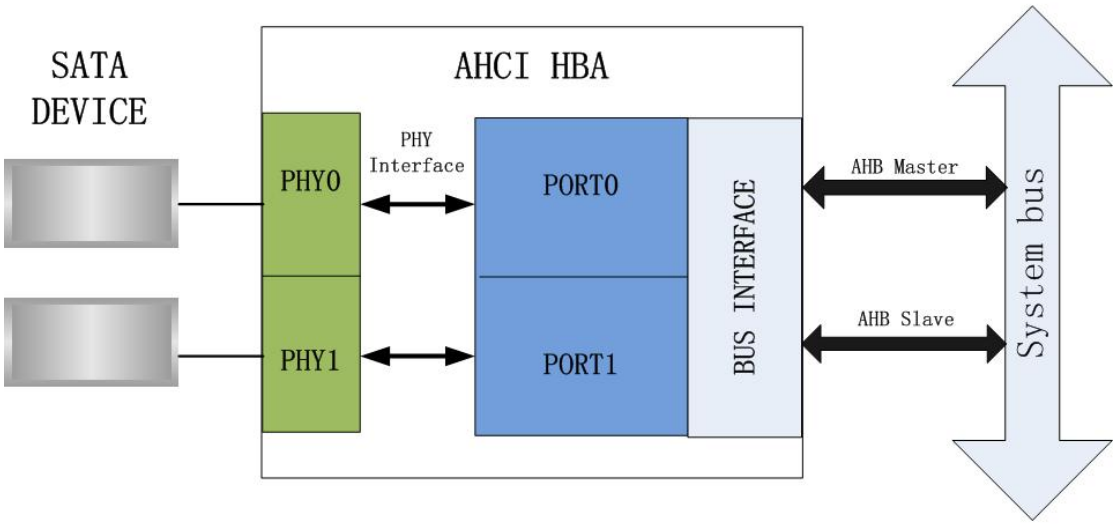



图 11-1 SATA 系统模块图

## 11.2 SATA 寄存器描述

SATA 的基地址是 0x1fe30000，寄存器的定义和协议标准定义完全一致，Linux 驱动只需要修改基地址就可以完成软件移植工作。如需自行修改驱动请参考 SATA 和 AHCI 规范。

| 地址 | 位宽 | 名称 | 描述 |
|---|---|---|---|
| 0x1fe3,0000 | 32 | CAP | HBA 特性寄存器 |
| 0x1fe3,0004 | 32 | GHC | 全局 HBA 控制寄存器 |
| 0x1fe3,0008 | 32 | IS | 中断状态寄存器 |
| 0x1fe3,000c | 32 | PI | 端口寄存器 |
| 0x1fe3,0010 | 32 | VS | AHCI 版本寄存器 |
| 0x1fe3,0014 | 32 | CCC_CTL | 命令完成合并控制寄存器 |
| 0x1fe3,0018 | 32 | CCC_PORTS | 命令完成合并端口寄存器 |

| | | | |
|---|---|---|---|
| 0x1fe3,0024 | 32 | CAP2 | HBA 特性扩展寄存器 |
| 0x1fe3,00A0 | 32 | BISTAFR | BIST 激活 FIS |
| 0x1fe3,00A4 | 32 | BISTCR | BIST 控制寄存器 |
| 0x1fe3,00A8 | 32 | BISTCTR | BIST FIS 计数寄存器 |
| 0x1fe3,00AC | 32 | BISTSR | BIST 状态寄存器 |
| 0x1fe3,00B0 | 32 | BISTDECR | BIST 双字错计数寄存器 |
| 0x1fe3,00BC | 32 | OOBR | OOB 寄存器 |
| 0x1fe3,00E0 | 32 | TIMER1MS | 1ms 计数寄存器 |
| 0x1fe3,00E8 | 32 | GPARAM1R | 全局参数寄存器 1 |
| 0x1fe3,00EC | 32 | GPARAM2R | 全局参数寄存器 2 |
| 0x1fe3,00F0 | 32 | PPARAMR | 端口参数寄存器 |
| 0x1fe3,00F4 | 32 | TESTR | 测试寄存器 |
| 0x1fe3,00F8 | 32 | VERIONR | 版本寄存器 |
| 0x1fe3,00FC | 32 | IDR | ID 寄存器 |
| 0x1fe3,0100 | 32 | P0_CLB | 命令列表基地址低 32 位 |
| 0x1fe3,0104 | 32 | P0_CLBU | 命令列表基地址高 32 位 |
| 0x1fe3,0108 | 32 | P0_FB | FIS 基地址低 32 位 |
| 0x1fe3,010c | 32 | P0_FBU | FIS 基地址高 32 位 |
| 0x1fe3,0110 | 32 | P0_IS | 中断状态寄存器 |
| 0x1fe3,0114 | 32 | P0_IE | 中断使能寄存器 |
| 0x1fe3,0118 | 32 | P0_CMD | 命令寄存器 |
| 0x1fe3,0120 | 32 | P0_TFD | 任务文件数据寄存器 |
| 0x1fe3,0124 | 32 | P0_SIG | 签名寄存器 |
| 0x1fe3,0128 | 32 | P0_SSTS | SATA 状态寄存器 |
| 0x1fe3,012C | 32 | P0_SCTL | SATA 控制寄存器 |
| 0x1fe3,0130 | 32 | P0_SERR | SATA 错误寄存器 |
| 0x1fe3,0134 | 32 | P0_SACT | SATA 激活寄存器 |
| 0x1fe3,0138 | 32 | P0_CI | 命令发送寄存器 |
| 0x1fe3,013C | 32 | P0_SNTF | SATA 命令通知寄存器 |
| 0x1fe3,0170 | 32 | P0_DMACR | DMA 控制寄存器 |
| 0x1fe3,0178 | 32 | P0_PHYCR | PHY 控制寄存器 |
| 0x1fe3,017C | 32 | P0_PHYSR | PHY 状态寄存器 |
| 0x1fe3,0180 | 32 | P1_CLB | 命令列表基地址低 32 位 |
| 0x1fe3,0184 | 32 | P1_CLBU | 命令列表基地址高 32 位 |
| 0x1fe3,0188 | 32 | P1_FB | FIS 基地址低 32 位 |
| 0x1fe3,018c | 32 | P1_FBU | FIS 基地址高 32 位 |
| 0x1fe3,0190 | 32 | P1_IS | 中断状态寄存器 |
| 0x1fe3,0194 | 32 | P1_IE | 中断使能寄存器 |
| 0x1fe3,0108 | 32 | P1_CMD | 命令寄存器 |
| 0x1fe3,01a0 | 32 | P1_TFD | 任务文件数据寄存器 |
| 0x1fe3,01a4 | 32 | P1_SIG | 签名寄存器 |
| 0x1fe3,01a8 | 32 | P1_SSTS | SATA 状态寄存器 |
| 0x1fe3,01aC | 32 | P1_SCTL | SATA 控制寄存器 |
| 0x1fe3,01b0 | 32 | P1_SERR | SATA 错误寄存器 |

| | | | |
|---|---|---|---|
| 0x1fe3,01b4 | 32 | P1_SACT | SATA 激活寄存器 |
| 0x1fe3,01b8 | 32 | P1_CI | 命令发送寄存器 |
| 0x1fe3,01bC | 32 | P1_SNTF | SATA 命令通知寄存器 |
| 0x1fe3,01f0 | 32 | P1_DMACR | DMA 控制寄存器 |
| 0x1fe3,01f8 | 32 | P1_PHYCR | PHY 控制寄存器 |
| 0x1fe3,01fC | 32 | P1s_PHYSR | PHY 状态寄存器 |

## 11.3 SATA 时钟配置

SATA 外部参考时钟可以是：25;50;100;125MHz,不同时钟频率配置不同见下表：

| Ref_clk | 0x1fd00418[27:26] | 0x1fd00418[24:20] | 0x1fd00418[19:18] |
|---|---|---|---|
| 25 MHz | 2’b01 | 5’b110 | 2’b10 |
| 50 MHz | 2’b00 | 5’b110 | 2’b10 |
| 100 MHz | 2’b10 | 5’b110 | 2’b10 |
| 125 MHz | 2’b10 | 5’b101 | 2’b00 |

# 12 USB HOST

## 12.1 总体概述

1A 的 USB 主机端口特性如下：

—兼容 USB Rev 1.1 、USB Rev 2.0 协议

—兼容 OHCI Rev 1.0 、EHCI Rev 1.0 协议

—支持 LS（Low Speed）、FS（Full Speed）和 HS（High Speed）的 USB 设备

LPC_REG[30]（基地址 0x1ff1_0204）是 USB 复位使能信号，USB 工作前需要置 1

USB 主机控制器模块图如所示：

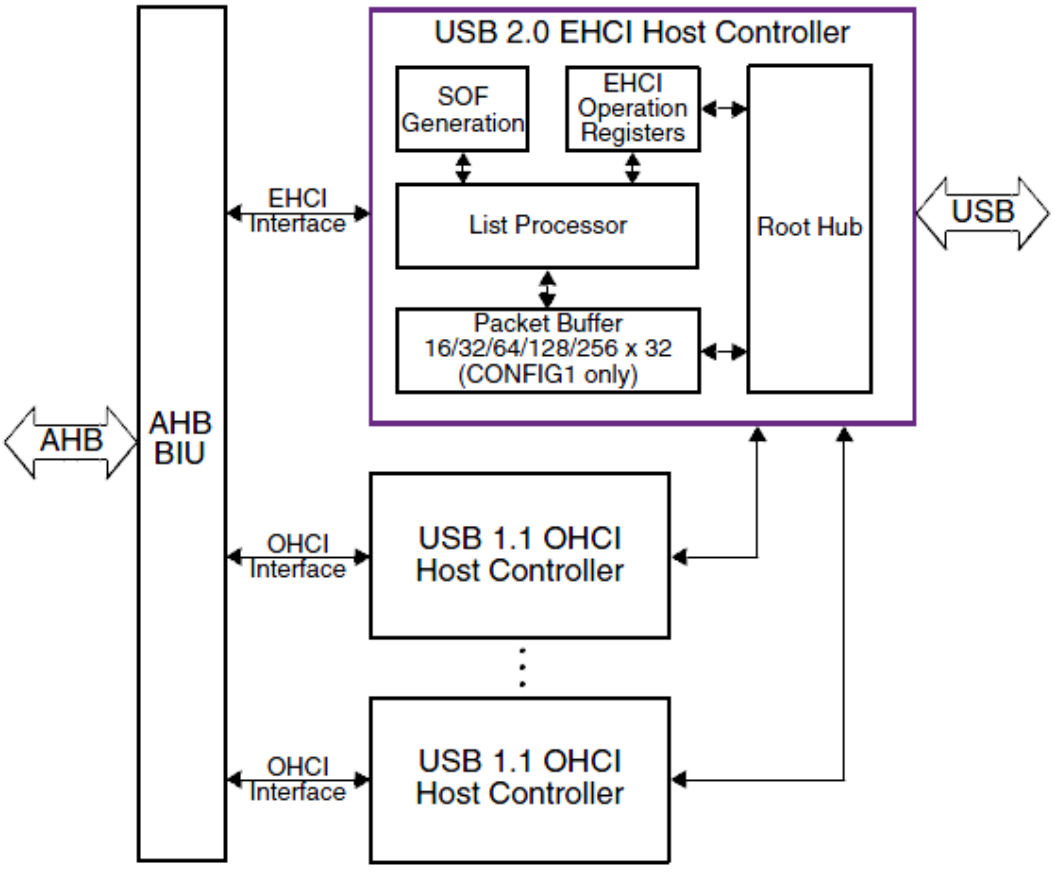



图 12-1 USB主机控制器模块图

## 12.2 USB 主机控制器寄存器

### 12.2.1 EHCI 相关寄存器

EHCI 的相关寄存器包括 Capability 寄存器、Operational 寄存器和，EHCI 实现相关寄存器。1A 的 USB 主机控制器兼容 EHCI Rev 1.0 协议，Capability 寄存器和 Operational 寄存器的详细信息参照 Enhanced Host Controller Interface Rev 1.0 Specification。

### 12.2.2 Capability 寄存器

| 名称 | 地址 | 宽度 | 访问 | 说明 |
|---|---|---|---|---|
| HCCAPBASE | 0x1fe00000 | 32 | RO | 默认值为 32’h01000010 |
| HCSPARAMS | 0x1fe00004 | 32 | RO | 默认值为 32’h00001116 |
| HCCPARAMS | 0x1fe00008 | 32 | RO | 默认值为 32’h0000A010 |

**（注：USBBase 固定为 EHCI slave 的起始地址 0x1fe00000）**

### 12.2.3 Operational 寄存器

| 名称 | 地址 | 宽度 | 访问 | 说明 |
|---|---|---|---|---|
| USBCMD | 0x1fe00010 | 32 | R/W、RO | USB 主机控制器的命令寄存器 |
| USBSTS | 0x1fe00014 | 32 | R/W、RO | USB 主机控制器的状态寄存器 |
| USBINTR | 0x1fe00018 | 32 | R/W | USB 主机控制器的中断设置寄存器 |
| FRINDEX | 0x1fe0001c | 32 | R/W | USB 主机控制器的帧索引寄存器 |
| CTRLDSSEGMENT | 0x1fe00020 | 32 | R/W | 存放 EHCI 控制数据结构的地址 |
| PERIODICLISTBASE | 0x1fe00024 | 32 | R/W | 存放周期数据帧表的起始地址 |
| ASYNCLISTADDR | 0x1fe00028 | 32 | R/W | 存放下一个要被执行的异步队列的起始地址 |
| CONFIGFLAG | 0x1fe00050 | 32 | R/W | 配置模式寄存器 |
| PORTSC 1 | 0x1fe00054 | 32 | R/W、RO | 端口 1 状态和控制寄存器 |
| PORTSC 2 | 0x1fe00058 | 32 | R/W、RO | 端口 2 状态和控制寄存器 |

**（注：USBOPBase 固定为 EHCI slave 的起始地址+ `h10）**

### 12.2.4 EHCI 实现相关寄存器

EHCI 实现相关寄存器的详细描述如下。

| 名称 | 地址 | 宽度 | 访问 | 说明 |
|---|---|---|---|---|
| INSNREG00 | 0x1fe00090 | 32 | R/W | 帧的长度配置寄存器 |
| INSNREG01 | 0x1fe00094 | 32 | R/W | 数据包缓冲区 OUT/IN 阈值寄存器 |
| INSNREG02 | 0x1fe00098 | 32 | RO | 数据包缓冲深度寄存器 |
| INSNREG03 | 0x1fe0009c | 32 | RO ,R/W | 参照寄存器详细描述 |
| INSNREG04 | 0x1fe000a0 | 32 | R/W | 用于 Debug |
| INSNREG05 | 0x1fe000a4 | 32 | RO, R/W | UTMI 配置（默认配置），控制和状态寄存器 |
| INSNREG06 | 0x1fe000a8 | 32 | RO | AHB 错误状态寄存器 |
| INSNREG07 | 0x1fe000ac | 32 | RO | AHB Master 错误地址寄存器 |
| INSNREG08 | 0x1fe000b0 | 32 | RO | HSIC 使能寄存器 |

#### 12.2.4.1 INSNREG00 寄存器（disable）

#### 12.2.4.2 INSNREG01 寄存器

| 位域 | 访问 | 复位值 | 说明 |
|---|---|---|---|
| 31:16 | R/W | 16’h0020 | OUT 阈值（单位是 4 bytes），一旦从系统内存中取出的数据量达到 OUT 阈值，就开始 USB 传输，最小为 16bytes |
| 15:0 | R/W | 16’h0020 | IN 阈值（单位是 4 bytes），一旦 Packet Buffer 里的数据量达到 IN 阈值，就开始向内存传输，最小为 16bytes |

#### 12.2.4.3 INSNREG02 寄存器

| 位域 | 访问 | 复位值 | 说明 |
|---|---|---|---|
| 31:12 | Reserved | 20’h0 | 保留 |
| 11:0 | RO | 12’h0020 | 数据包缓冲深度（单位是 4 bytes） |

#### 12.2.4.4 INSNREG03 寄存器

| 位域 | 访问 | 复位值 | 说明 |
|---|---|---|---|
| 31:13 | Reserved | 19’h0 | 保留 |
| 12:10 | RO | 3’h0 | 这个字段指定 phy_clks 的额外延时，这个延时被添加到“Tx-Tx turnaround Delay”中。 |
| 9 | RO | 1’h0 | 置 1：将迫使主机控制器在一帧的每一微帧中获取周期数据帧表，<br>置 0：主机控制器在一帧的微帧 0 中获取周期数据帧表 |
| 8:1 | R/W | 8’h0 | 时间可容忍偏移，这个字段用来指明为了容忍计算可用时间而要附加的字节数。计算可用时间为以后的传输弹性增加的， |

| | | | |
|---|---|---|---|
| | | | 用户程序默认不需要修改这个字段。 |
| 0 | RO | 1’h0 | Break Memory Transaction 模式<br>置 1：使能此功能<br>置 0：禁止此功能 |

#### 12.2.4.5 INSNRE04 寄存器（仅用于调试，软件不必更改此寄存器）

| 位域 | 访问 | 复位值 | 说明 |
|---|---|---|---|
| 31:6 | Reserved | 26’h0 | 保留 |
| 5 | R/W | 1’h0 | 置 1：禁止 automatic 功能，即当软件清除 Run/Stop 位时，USB 主机控制器会把挂起（Suspend）的端口唤醒<br>置 0：启用 automatic 功能，当 reset Run/Stop 位时，Suspend 信号会置为 1 |
| 4 | R/W | 1’h0 | 置 1：禁止 NAK reload 修复<br>置 0：启用 NAK reload 修复 |
| 3 | Reserved | 1’h0 | 保留 |
| 2 | R/W | 1’h0 | 置 1：缩短端口枚举（enumeration）时间（仿真） |
| 1 | R/W | 1’h0 | 置 1： HCCPARAMS 寄存器的第 17、15:4、2:0 位均可写 |
| 0 | R/W | 1’h0 | 置 1： HCSPARAMS 寄存器可写 |

#### 12.2.4.6 INSNRE05 寄存器

| 位域 | 访问 | 复位值 | 说明 |
|---|---|---|---|
| 31:18 | Reserved | 14’h0 | 保留 |
| 17 | RO | 1’h0 | 置 1：表示对这个寄存器进行了一个写操作，硬件正在执行<br>置 0：表示硬件已经执行完操作 |
| 16:13 | R/W | 5’h0 | 端口号 |
| 12 | R/W | 4’h1 | VControlLoadM<br>置 1：NOP<br>置 0：Load |
| 11:8 | R/W | 4’h0 | VControl |
| 7:0 | RO | 4’h0 | VStatus |

#### 12.2.4.7 INSNREG06 寄存器

| 位域 | 访问 | 复位值 | 说明 |
|---|---|---|---|

| | | | |
|---|---|---|---|
| 31 | R/W | 1'h0 | 一旦 AHB 出错即被捕获并置 1，写 0 清除该字段 |
| 30:12 | Reserved | 19'h0 | 保留 |
| 11:9 | RO | 3'h0 | AHB 出错时控制段 HBURST 的值 |
| 8:4 | RO | 5'h0 | AHB 出错的 burst 的预计节拍数 |
| 3:0 | RO | 4'h0 | 在当前 burst 下，AHB 出错前完成的节拍数 |

#### 12.2.4.8 INSNREG07 寄存器

| 位域 | 访问 | 复位值 | 说明 |
|---|---|---|---|
| 31:0 | RO | 32'h0 | AHB 出错时控制段的地址 |

#### 12.2.4.9 INSNREG08 寄存器

| 位域 | 访问 | 复位值 | 说明 |
|---|---|---|---|
| 31:0 | RO | 1'b0 | HSIC 使能 |

## 12.3 OHCI 相关寄存器

OHCI 的相关寄存器包括 Operational 寄存器和 OHCI 实现相关寄存器。1A 的 USB 主机控制器兼容 OHCI Rev 1.0 协议，Operational 寄存器的详细信息参照 Open Host Controller Interface Rev 1.0 Specification。

### 12.3.1 Operational 寄存器

| 名称 | 地址 | 宽度 | 访问 | 说明 |
|---|---|---|---|---|
| HcRevision | 0x1fe08000 | 32 | - | 控制和状态 |
| HcControl | 0x1fe08004 | 32 | - | |
| HcCommonStatus | 0x1fe08008 | 32 | - | |
| HcInterruptStatus | 0x1fe0800C | 32 | - | |
| HcInterruptEnable | 0x1fe08010 | 32 | - | |
| HcInterruptDisable | 0x1fe08014 | 32 | -- | |
| HcHCCA | 0x1fe08018 | 32 | - | 内存指针 |
| HcPeriodCuttentED | 0x1fe0801C | 32 | - | |
| HcControlHeadED | 0x1fe08020 | 32 | – | |
| HcControlCurrentED | 0x1fe08024 | 32 | – | |

| | | | | |
|---|---|---|---|---|
| HcBulkHeadED | 0x1fe08028 | 32 | – | |
| HcBulkCurrentED | 0x1fe0802C | 32 | – | |
| HcDoneHead | 0x1fe08030 | 32 | – | |
| HcRmInterval | 0x1fe08034 | 32 | – | 帧计数器 |
| HcFmRemaining | 0x1fe08038 | 32 | – | |
| HcFmNumber | 0x1fe0803C | 32 | – | |
| HcPeriodicStart | 0x1fe08040 | 32 | – | |
| HcLSThreshold | 0x1fe08044 | 32 | – | |
| HcRhDescriptorA | 0x1fe08048 | 32 | – | 根集线器 |
| HcRhDescriptorB | 0x1fe0804C | 32 | – | |
| HcRhStatus | 0x1fe08050 | 32 | – | |
| HcRhPortStatus1 | 0x1fe08054 | 32 | – | |
| HcRhPortStatus2 | 0x1fe08058 | 32 | – | |

### 12.3.2 OHCI 实现相关寄存器

除了标准的 OHCI 操作寄存器，还实现了两个额外寄存器（寄存器偏移 0x98 和 0x9C）用来报告 AHB 的错误状态。

| 名称 | 地址 | 宽度 | 访问 | 说明 |
|---|---|---|---|---|
| INSNREG06 | 0x1fe08098 | 32 | RO | AHB 错误状态寄存器 |
| INSNREG07 | 0x1fe0809c | 32 | RO | AHB Master 错误地址寄存器 |

#### 12.3.2.1 INSNREG06 寄存器

| 位域 | 访问 | 复位值 | 说明 |
|---|---|---|---|
| 31 | R/W | 1’h0 | 一旦 AHB 出错即被捕获并置 1，写 0 清除该字段 |
| 30:12 | RO | 19’h0 | 保留 |
| 11:9 | RO | 3’h0 | AHB 出错时控制段 HBURST 的值 |
| 8:4 | RO | 5’h0 | AHB 出错的 burst 的预计节拍数 |
| 3:0 | RO | 4’h0 | 在当前 burst 下，AHB 出错前完成的节拍数 |

#### 12.3.2.2 INSNREG07 寄存器

| 位域 | 访问 | 复位值 | 说明 |
|---|---|---|---|

| | | | |
|---|---|---|---|
| 31:0 | RO | 32’h0 | AHB 出错时控制段的地址 |

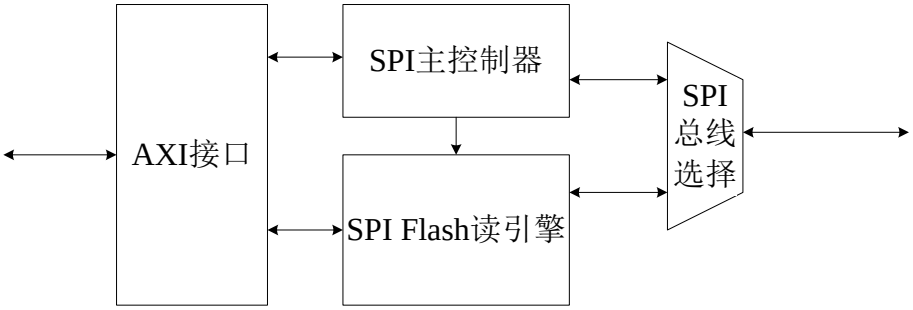
# 13 SPI0

串行外围设备接口 SPI 总线技术是 Motorola 公司推出的多种微处理器、微控制器以及外围设备之间的一种全双工、同步、串行数据接口标准。

## 13.1 SPI 控制器结构

本系统集成的 SPI 控制器仅可作为主控端，所连接的是从设备。其结构如下图所示，由一个 SPI 主控制器和 SPI Flash 读引擎组成。对于软件而言，SPI 控制器除了有若干 IO 寄存器外还有一段映射到 SPI Flash 的只读 memory 空间。如果将这段 memory 空间分配在 0x1fc00000，复位后不需要软件干预就可以直接访问，从而支持处理器从 SPI Flash 启动。SPI0 的 IO 寄存器的基地址 0x1fe80000，外部存储地址空间是 0x1f00,0000–0x1f7f,ffff 共 8MB。

本模块结构如下图所示，由 AXI 接口、简单的 SPI 主控制器、SPI Flash 读引擎和总线选择模块组成。根据访问的地址和类型，AXI 上的合法请求转发到 SPI 主控制器或者 SPI Flash 读引擎中(非法请求被丢弃)。




下图是 SPI 主控制器的结构，系统寄存器包括控制寄存器，状态寄存器和外部寄存器，分频器生成 SPI 总线工作的时钟信号，由于数据读、写缓冲器（FIFO）允许 SPI 同时进行串行发送和接收数据。

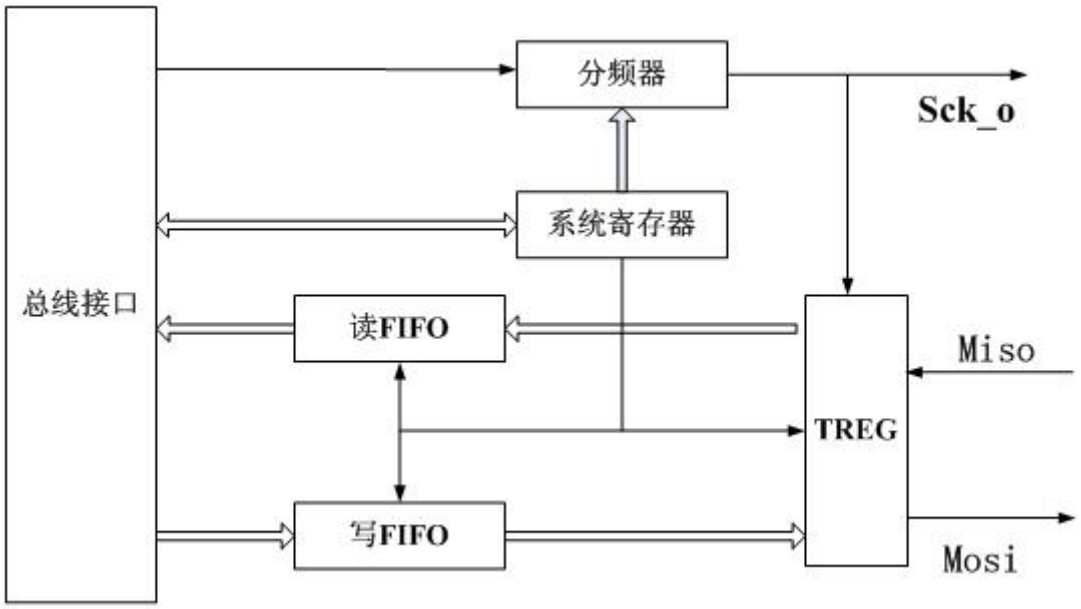



图 13-1 SPI 主控制器结构

## 13.2 SPI 控制器寄存器

### 13.2.1 控制寄存器（SPCR）

中文名： 控制寄存器
寄存器位宽： [7：0]
偏移量： 0x00
复位值： 0x10

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7 | Spie | 1 | RW | 中断输出使能信号 高有效 |
| 6 | spe | 1 | RW | 系统工作使能信号高有效 |
| 5 | Reserved | 1 | RW | 保留 |
| 4 | mstr | 1 | RW | master 模式选择位，此位一直保持 1 |
| 3 | cpol | 1 | RW | 时钟极性位 |
| 2 | cpha | 1 | RW | 时钟相位位 1 则相位相反，为 0 则相同 |
| 1:0 | spr | 2 | RW | sclk_o 分频设定，需要与 sper 的 spre 一起使用 |

### 13.2.2 状态寄存器（SPSR）

中文名： 状态寄存器
寄存器位宽： [7：0]
偏移量： 0x01
复位值： 0x05

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7 | spif | 1 | RW | 中断标志位 1 表示有中断申请，写 1 则清零 |
| 6 | wcol | 1 | RW | 写寄存器溢出标志位 为 1 表示已经溢出，写 1 则清零 |
| 5:4 | Reserved | 2 | RW | 保留 |
| 3 | wffull | 1 | RW | 写寄存器满标志 1 表示已经满 |

| | | | | |
|---|---|---|---|---|
| 2 | wfempty | 1 | RW | 写寄存器空标志 1 表示空 |
| 1 | rffull | 1 | RW | 读寄存器满标志 1 表示已经满 |
| 0 | rfempty | 1 | RW | 读寄存器空标志 1 表示空 |

### 13.2.3 数据寄存器（TxFIFO/RxFIFO）

中文名： 数据传输寄存器
寄存器位宽： [7：0]
偏移量： 0x02
复位值： 0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7:0 | Tx FIFO | 8 | W | 数据传输寄存器 |

### 13.2.4 外部寄存器（SPER）

中文名： 外部寄存器
寄存器位宽： [7：0]
偏移量： 0x03
复位值： 0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7:6 | icnt | 2 | RW | 在传输完多少个字节后送出中断申请信号<br>00 – 1 字节 01 - 2 字节<br>10 - 3 字节 11 - 3 字节 |
| 5:3 | Reserved | 3 | RW | 保留 |
| 2 | mode | 1 | RW | spi 接口模式控制<br>0: 采样与发送时机同时<br>1: 采样与发送时机错开半周期 |
| 1:0 | spre | 2 | RW | 与 Spr 一起设定分频的比率 |

分频系数（分频的源时钟频率是 DDR_CLK 的一半，参考第 29 章）：

| spre<br>spr | 00<br>00 | 00<br>01 | 00<br>10 | 00<br>11 | 01<br>00 | 01<br>01 | 01<br>10 | 01<br>11 | 10<br>00 | 10<br>01 | 10<br>10 | 10<br>11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 分频系数 | 2 | 4 | 16 | 32 | 8 | 64 | 128 | 256 | 512 | 1024 | 2048 | 4096 |

### 13.2.5 参数控制寄存器（SFC_PARAM）

中文名： SPI Flash 参数控制寄存器
寄存器位宽： [7：0]
偏移量： 0x04
复位值： 0x21

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7:4 | clk_div | 4 | RW | 时钟分频数选择（分频系数与{spre,spr}组合相同） |
| 3 | dual_io | 1 | RW | 使用双 I/O 模式，优先级高于快速读模式 |
| 2 | fast_read | 1 | RW | 使用快速读模式 |
| 1 | burst_en | 1 | RW | spi flash 支持连续地址读模式 |
| 0 | memory_en | 1 | RW | spi flash 读使能，无效时 csn[0]可由软件控制。 |

### 13.2.6 片选控制寄存器（SFC_SOFTCS）

中文名： SPI Flash 片选控制寄存器
寄存器位宽： [7：0]

偏移量：　　0x05
复位值：　　0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7:4 | csn | 4 | RW | csn 引脚输出值 |
| 3:0 | csen | 4 | RW | 为 1 时对应位的 cs 线由 7:4 位控制 |

### 13.2.7 时序控制寄存器（SFC_TIMING）

中文名：　　SPI Flash 时序控制寄存器
寄存器位宽：　[7：0]
偏移量：　　0x06
复位值：　　0x03

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7:2 | Reserved | 6 | RW | 保留 |
| 1:0 | tCSH | 2 | RW | SPI Flash 的片选信号最短无效时间，以分频后时钟周期 T 计算<br>00: 1T<br>01: 2T<br>10: 4T<br>11: 8T |

## 13.3 接口时序

**SPI 主控制器外部接口时序图**

如图 13-2 所示，SPI 主控制器发送数据时，数据提前半拍放在 MOSI 引线上，接着从设备端用时钟边沿锁存数据。根据时钟极性（CPOL）和时钟相位（CPHA）的设定，有 4 种可能的时序关系。

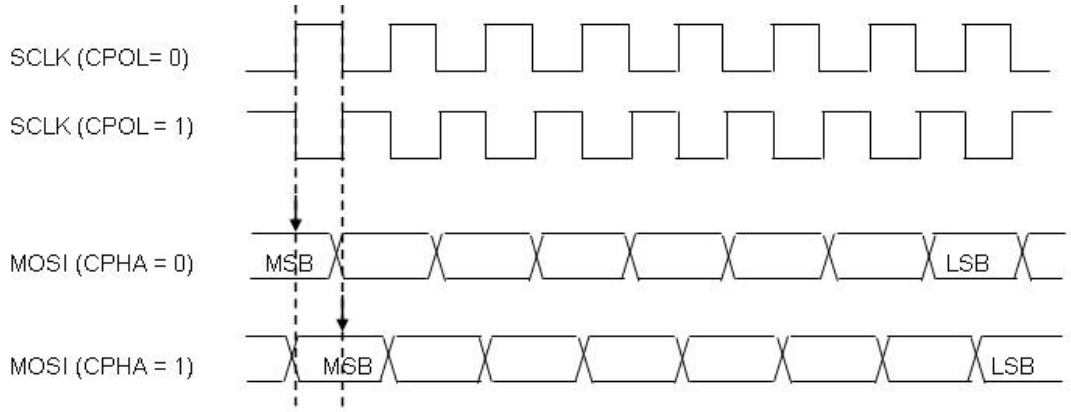
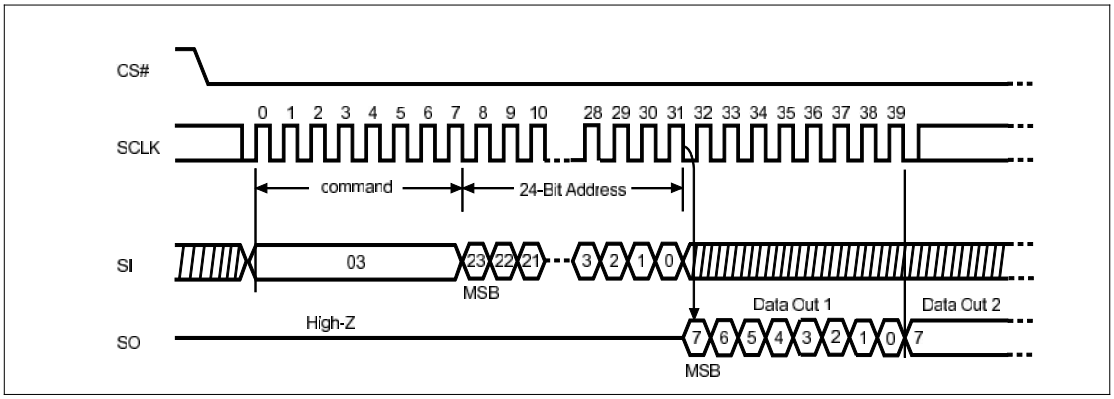


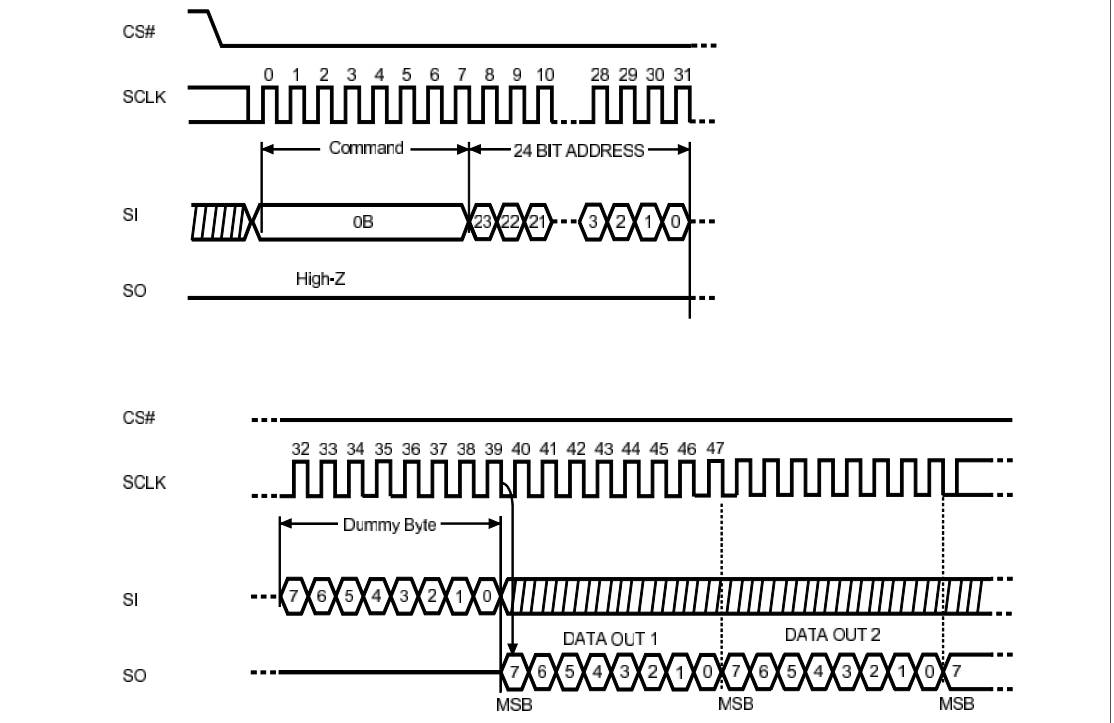

图 13-2 SPI时序图主控制器

**SPI Flash 访问时序图**

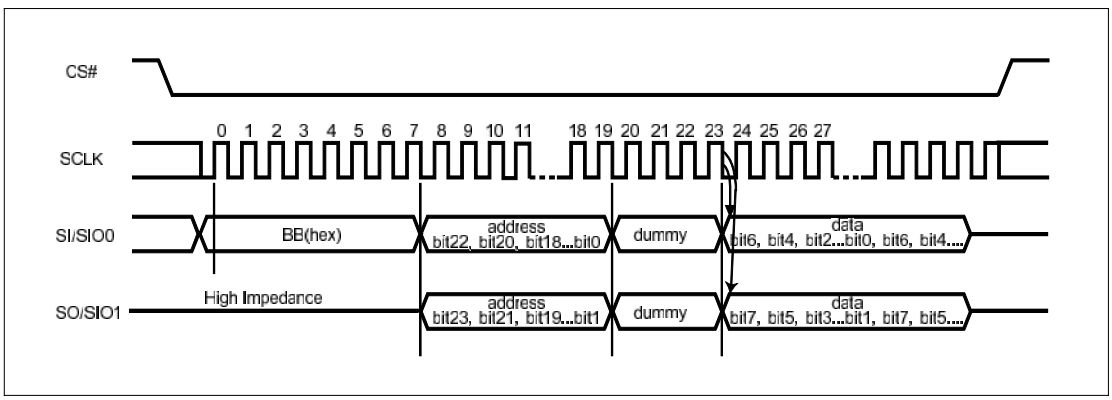
- 标准读模式


CS#
SCLK
0 1 2 3 4 5 6 7 8 9 10 28 29 30 31 32 33 34 35 36 37 38 39
command
24-Bit Address
SI
03
23 22 21 3 2 1 0
MSB
Data Out 1
Data Out 2
SO
High-Z
7 6 5 4 3 2 1 0 7
MSB

● 快速读模式


CS#
SCLK
0 1 2 3 4 5 6 7 8 9 10 28 29 30 31
Command
24 BIT ADDRESS
SI
0B
23 22 21 3 2 1 0
SO
High-Z
CS#
SCLK
32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47
Dummy Byte
SI
7 6 5 4 3 2 1 0
DATA OUT 1
DATA OUT 2
SO
7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7
MSB
MSB
MSB

● 双 I/O 模式


CS#
SCLK
0 1 2 3 4 5 6 7 8 9 10 11 18 19 20 21 22 23 24 25 26 27
SI/SIO0
BB(hex)
address
bit22, bit20, bit18...bit0
dummy
data
bit6, bit4, bit2...bit0, bit6, bit4....
SO/SIO1
High Impedance
address
bit23, bit21, bit19...bit1
dummy
data
bit7, bit5, bit3...bit1, bit7, bit5....

在所有模式下，若没有使能连续地址读，则 CS 将在传输完一个字节数据后拉高。

## 13.4 SPI Flash 控制器使用指南

**SPI 主控制器的读写操作**

**1. 模块初始化**

- 停止 SPI 控制器工作，对控制寄存器 spcr 的 spe 位写 0
- 重置状态寄存器 spsr，对寄存器写入 8'b1100_0000
- 设置外部寄存器 sper，包括中断申请条件 sper[7:6]和分频系数 sper[1:0]，具体参考寄存器说明
- 配置 SPI 时序，包括 spcr 的 cpol、cpha 和 sper 的 mode 位。mode 为 1 时是标准 SPI 实现，为 0 时为兼容模式。
- 配置中断使能，spcr 的 spie 位
- 启动 SPI 控制器，对控制寄存器 spcr 的 spe 位写 1

**2. 模块的发送/传输操作**

- 往数据传输寄存器写入数据
- 传输完成后从数据传输寄存器读出数据。由于发送和接收同时进行，即使 SPI 从设备没有发送有效数据也必须进行读出操作。

**3. 中断处理**

- 接收到中断申请
- 读状态寄存器 spsr 的值，若 spsr[2]为 1 则表示数据发送完成，若 spsr[0]为 1 则表示已经接收数据
- 读或写数据传输寄存器
- 往状态寄存器 spsr 的 spif 位写 1，清除控制器的中断申请

**硬件 SPI Flash 读**

**1. 初始化**

- 将 SFC_PARAM 的 memory_en 位写 1。当 SPI 被选为启动设备时此位复位为 1。
- 设置读参数(时钟分频、连续地址读、快速读、双 I/O、tCSH 等)。这些参数复位值均为最保守的值。

**2. 更改参数**

如果所使用的 SPI Flash 支持更高的频率或者提供增强功能，修改相应参数可以大大加快 Flash 的访问速度。参数的修改不需要关闭 SPI Flash 读使能(memory_en)。具体参考寄存器说明。

**混合访问 SPI Flash 和 SPI 主控制器**

**1. 对 SPI Flash 进行读以外的访问**

将 SPI Flash 读使能关闭后，软件就可直接控制 csn[0]，并通过 SPI 主控制器访问 SPI 总线。这意味着在进行此操作时，不能从 SPI Flash 中取指。

除了读以外，SPI Flash 还实现了很多命令(如擦除、写入)，具体参见相关 Flash 的文档。

# 14 SPI1

串行外围设备接口 SPI 总线技术是 Motorola 公司推出的多种微处理器、微控制器以及外围设备之间的一种全双工、同步、串行数据接口标准。

## 14.1 SPI 主控制器结构

SPI1 和 SPI0 的实现完全一样，系统启动地址不会映射到 SPI1 控制器，所以 SPI1 不支持系统启动。SPI1 的 IO 寄存器的基地址 0x1fec0000，SPI1 的外部存储地址空间是 0x1f80,0000–0x1fbf,ffff 共 4MB。所有结构和配置相关请参考第 13 章信息。

# 15 中断

## 15.1 中断控制器总体描述

龙芯1A内置简单、灵活的中断控制器。龙芯1A的中断控制器除了管理GPIO输入的中断信号外，中断控制器还处理内部事件引起的中断。所有的中断寄存器的位域安排相同，一个中断源对应其中一位。中断控制器共五个中断输出连接CPU模块，分别对应INT0, INT1, INT2, INT3, INT4。INT0/1、INT2/3/4还分别送到芯片输出引脚INTn0、INTn1，用于在桥片模式下给主处理器发中断。

芯片支持64个内部中断和96个GPIO的中断；其中INT0和INT1分别对应于64个内部中断的前后32位，INT2 、INT3 和INT4对应于88个外部GPIO中断。具体如下表所示：

| | INT0 | INT1 | INT2 | INT3 | INT4 |
|---|---|---|---|---|---|
| 31 | 保留 | 保留 | GPIO31 | GPIO63 | 保留 |
| 30 | 保留 | 保留 | GPIO30 | GPIO62 | 保留 |
| 29 | NAND_int | 保留 | GPIO29 | GPIO61 | 保留 |
| 28 | TOY_TICK | 保留 | GPIO28 | GPIO60 | 保留 |
| 27 | RTC_TICK | 保留 | GPIO27 | GPIO59 | 保留 |
| 26 | TOY_INT2 | 保留 | GPIO26 | GPIO58 | 保留 |
| 25 | TOY_INT1 | 保留 | GPIO25 | GPIO57 | 保留 |
| 24 | TOY_INT0 | 保留 | GPIO24 | GPIO56 | 保留 |
| 23 | RTC_INT2 | 保留 | GPIO23 | GPIO55 | GPIO87 |
| 22 | RTC_INT1 | 保留 | GPIO22 | GPIO54 | GPIO86 |
| 21 | RTC_INT0 | 保留 | GPIO21 | GPIO53 | GPIO85 |
| 20 | PWM3 | 保留 | GPIO20 | GPIO52 | GPIO84 |
| 19 | PWM2 | 保留 | GPIO19 | GPIO51 | GPIO83 |
| 18 | PWM1 | 保留 | GPIO18 | GPIO50 | GPIO82 |
| 17 | PWM0 | 保留 | GPIO17 | GPIO49 | GPIO81 |
| 16 | LPC_int | 保留 | GPIO16 | GPIO48 | GPIO80 |
| 15 | DMA2 | 保留 | GPIO15 | GPIO47 | GPIO79 |
| 14 | DMA1 | 保留 | GPIO14 | GPIO46 | GPIO78 |
| 13 | DMA0 | 保留 | GPIO13 | GPIO45 | GPIO77 |
| 12 | KB_int | 保留 | GPIO12 | GPIO44 | GPIO76 |
| 11 | MS_int | 保留 | GPIO11 | GPIO43 | GPIO75 |
| 10 | AC97 | 保留 | GPIO10 | GPIO42 | GPIO74 |
| 9 | SPI1 | 保留 | GPIO09 | GPIO41 | GPIO73 |
| 8 | SPI0 | 保留 | GPIO08 | GPIO40 | GPIO72 |
| 7 | CAN1 | 保留 | GPIO07 | GPIO39 | GPIO71 |
| 6 | CAN0 | 保留 | GPIO06 | GPIO38 | GPIO70 |
| 5 | UART3 | GPU_INT | GPIO05 | GPIO37 | GPIO69 |
| 4 | UART2 | SATA_INT | GPIO04 | GPIO36 | GPIO68 |
| 3 | UART1 | Gmac1 | GPIO03 | GPIO35 | GPIO67 |
| 2 | URAT0 | Gmac0 | GPIO02 | GPIO34 | GPIO66 |

| | | | | | |
|---|---|---|---|---|---|
| 1 | HPET_int | Ohci | GPIO01 | GPIO33 | GPIO65 |
| 0 | ACPI_int | Ehci | GPIO00 | GPIO32 | GPIO64 |

## 15.2 中断控制器寄存器描述

中断的使用首先要设置中断使能寄存器中相应的位来使能该中断，系统复位时默认不使能中断。然后设置中断触发类型寄存器、中断极性控制寄存器和中断输出控制寄存器相应的属性。最后当发生中断时，通过中断状态寄存器查看相应的中断源。

中断触发方式分为电平触发与边沿触发两种，电平触发方式时，中断控制器内部不寄存外部中断，此时对中断处理的响应完成后只需要清除对应设备上的中断就可以清除对 CPU 的相应中断。例如，上行网口向 CPU 发出接收包中断，网络驱动处理中断后，只要清除上行网口内部的中断寄存器中的中断状态，就可以清除 CPU 中断控制器的中断状态，而不需要通过对应的 INT_CLR 对 CPU 进行清中断。但是在边沿触发的方式下，中断控制器会寄存外部中断，此时软件处理中断时，需要通过写对应的 INT_CLR，清除 CPU 中断控制器内部的对应中断状态。另外，在边沿触发的情况下，用户可以通过写 INT_SET 位强置中断控制器的对应中断状态。

| 偏移地址 | 位 | 寄存器 | 描述 | 读写特性 |
|---|---|---|---|---|
| 0x1fd01040 | 32 | INTISR0 | 中断控制状态寄存器 0 | RO |
| 0x1fd01044 | 32 | INTIEN0 | 中断控制使能寄存器 0 | R/W |
| 0x1fd01048 | 32 | INTSET0 | 中断置位寄存器 0 | R/W |
| 0x1fd0104c | 32 | INTCLR0 | 中断清空寄存器 0 | R/W |
| 0x1fd01050 | 32 | INTPOL0 | 高电平触发中断使能寄存器 0 | R/W |
| 0x1fd01054 | 32 | INTEDGE0 | 边沿触发中断使能寄存器 0 | R/W |
| 0x1fd01058 | 32 | INTISR1 | 中断控制状态寄存器 1 | RO |
| 0x1fd0105c | 32 | INTIEN1 | 中断控制使能寄存器 1 | R/W |
| 0x1fd01060 | 32 | INTSET1 | 中断置位寄存器 1 | R/W |
| 0x0x1fd01064 | 32 | INTCLR1 | 中断清空寄存器 1 | R/W |
| 0x0x1fd01068 | 32 | INTPOL1 | 高电平触发中断使能寄存器 1 | R/W |
| 0x0x1fd0106c | 32 | INTEDGE1 | 边沿触发中断使能寄存器 1 | R/W |
| 0x0x1fd01070 | 32 | INTISR2 | 中断控制状态寄存器 2 | RO |
| 0x0x1fd01074 | 32 | INTIEN2 | 中断控制使能寄存器 2 | R/W |
| 0x0x1fd01078 | 32 | INTSET2 | 中断置位寄存器 2 | R/W |
| 0x0x1fd0107c | 32 | INTCLR2 | 中断清空寄存器 2 | R/W |
| 0x0x1fd01080 | 32 | INTPOL2 | 高电平触发中断使能寄存器 2 | R/W |
| 0x0x1fd01084 | 32 | INTEDGE2 | 边沿触发中断使能寄存器 2 | R/W |
| 0x0x1fd01088 | 32 | INTISR3 | 中断控制状态寄存器 3 | RO |
| 0x0x1fd0108c | 32 | INTIEN3 | 中断控制使能寄存器 3 | R/W |
| 0x0x1fd01090 | 32 | INTSET3 | 中断置位寄存器 3 | R/W |
| 0x0x1fd01094 | 32 | INTCLR3 | 中断清空寄存器 3 | R/W |
| 0x0x1fd01098 | 32 | INTPOL3 | 高电平触发中断使能寄存器 3 | R/W |

| | | | | |
|---|---|---|---|---|
| 0x0x1fd0109c | 32 | INTEDGE3 | 边沿触发中断使能寄存器 3 | R/W |
| 0x0x1fd010a8 | 32 | INTISR4 | 中断控制状态寄存器 4 | RO |
| 0x0x1fd010ac | 32 | INTIEN4 | 中断控制使能寄存器 4 | R/W |
| 0x0x1fd010a0 | 32 | INTSET4 | 中断置位寄存器 4 | R/W |
| 0x0x1fd010a4 | 32 | INTCLR4 | 中断清空寄存器 4 | R/W |
| 0x0x1fd010a8 | 32 | INTPOL4 | 高电平触发中断使能寄存器 4 | R/W |
| 0x0x1fd010ac | 32 | INTEDGE4 | 边沿触发中断使能寄存器 4 | R/W |
| 0x0x1fd010c0 | 32 | GPIOCFG0 | GPIO 配置寄存器 0 | R/W |
| 0x0x1fd010c4 | 32 | GPIOCFG1 | GPIO 配置寄存器 1 | R/W |
| 0x0x1fd010c8 | 32 | GPIOCFG2 | GPIO 配置寄存器 2 | R/W |
| 0x0x1fd010d0 | 32 | GPIOOE0 | GPIO 配置寄存器输入使能 0 | R/W |
| 0x0x1fd010d4 | 32 | GPIOOE1 | GPIO 配置寄存器输入使能 1 | R/W |
| 0x0x1fd010d8 | 32 | GPIOOE2 | GPIO 配置寄存器输入使能 2 | R/W |
| 0x0x1fd010e0 | 32 | GPIOIN0 | GPIO 配置寄存器输入寄存器 0 | R/W |
| 0x0x1fd010e4 | 32 | GPIOIN1 | GPIO 配置寄存器输入寄存器 1 | R/W |
| 0x0x1fd010e8 | 32 | GPIOIN2 | GPIO 配置寄存器输入寄存器 2 | R/W |
| 0x0x1fd010f0 | 32 | GPIOOUT0 | GPIO 配置寄存器输出寄存器 0 | R/W |
| 0x0x1fd010f4 | 32 | GPIOOUT1 | GPIO 配置寄存器输出寄存器 1 | R/W |
| 0x0x1fd010f8 | 32 | GPIOOUT2 | GPIO 配置寄存器输出寄存器 2 | R/W |
| 0x0x1fd01160 | 32 | ORDER_REG_ADDR | DMA 模块控制寄存器位 | |

# 16 SRAM

LS1A 集成片上 SRAM 控制器，数据宽度 16 位。

## 16.1 SRAM 控制器复用连接

SRAM 控制器没有专门的输出输入 PAD,SRAM 所有的 PAD 复用 PCI 实现，下表给出了 SRAM 复用 PCI PAD 的连接关系。

| SRAM | PCI | 说明 |
|---|---|---|
| SRAM_WEn | PCI_TRDY | 写信号 |
| SRAM_OEn | PCI_IRDY | 读信号 |
| SRAM_CSn[1] | PCI_DEVSEL | 片选信号 0 |
| SRAM_CSn[0] | PCI_FRAME | 片选信号 1 |
| SRAM_ADDR[23] | PCI_SERR | 地址 23 |
| SRAM_ADDR[22] | PCI_PERR | 地址 22 |
| SRAM_ADDR[21] | PCI_PAR | 地址 21 |
| SRAM_ADDR[20] | PCI_STOP | 地址 20 |
| SRAM_ADDR[19:16] | PCI_CBE[3:0] | 地址 19：16 |
| SRAM_ADDR[15:0] | PCI_AD[31:16] | 地址 15：0 |
| SRAM_DATA[15:0] | PCI_AD[15:0] | 数据 15：0 |

## 16.2 SRAM 控制器工作

由于 SRAM 所有的 PAD 复用 PCI 实现，下表给出了 SRAM 复用 PCI PAD 的配置方法。配置地址：0x1fd01108，复位值为 0。

| 寄存器位 | 配置寄存器 | 描述 |
|---|---|---|
| 0 | SRAM_SEL | 1’b1，SRAM 控制器使用 PAD<br>1’b0，PCI 控制器使用 PAD |
| 2：1 | Clock_period_i | 单位延迟的时钟周期数(PCI 时钟)<br>01: 1<br>10: 2<br>11: 4<br>00: 4 |
| 7:3 | Ram_count_init_i | 计数单位见，访问延迟delay_unit |
| 8 | Ram_width | sram 数据宽度<br>0: 8bit<br>1: 16bit |

# 17 DMA

## 17.1 DMA 控制器结构描述

DMA 来进行 DDR2 与设备间数据搬移工作，提高系统数据传输的效率。本章介绍的 DMA 是专用 DMA，DMA 共有三路，分别对应 NAND、AC97 播放、AC97 录音的数据传输。

DMA 传送数据的过程由三个阶段组成：

a) 传送前的预处理：由 CPU 完成以下步骤：配置 DMA 描述符相关的寄存器。

b) 数据传送：在 DMA 控制器的控制下自动完成。

c) 传送结束处理：发送中断请求。

本 DMA 控制器限定为以字为单位的数据搬运。根据 DMA 的定义，设计了下个描述符地址寄存器、源地址寄存器、目的地址寄存器、传送字数计数器、传送步长间隔、传送循环次数、DMA 控制逻辑等必备寄存器。DMA 的缓存大小为 128Byte（32x4Byte），以字为单位读写。

CPU 通过配置 DMA 寄存器，将来自于 DDR2 或设备的数据保存在缓存中，将缓存中的数据写入要对应的内存或设备中去，最后发送 DMA 传输结束信号。在 DMA 传输过程中，CPU 可以随时监听 DMA 的工作状态。

## 17.2 DMA 控制器与 APB 设备的交互

在 1A 中，使用 DMA 的 APB 设备包括 NAND，AC97， 每个设备都有单独的 DMA 控制器。AC97 的写通道是双通道，且 AC97 写使能的判断条件的 DMA_DADDR[31]=1， DMA_DADDR[29:28]域为 AC97 写模式选择域，判断 AC97 写操作是字节、半字或者字操作，与 AC97 的写模式配置一致。所以，如果是写 AC97 的写操作，需要在配置 DMA 描述符时，将 DAM_DADDR[31]配置为 1，将 DMA_DADDR[29:28]配置为需要的 AC97 写模式。

## 17.3 DMA 控制器

### 17.3.1 ORDER_ADDR_IN

中文名： 该寄存器广播到三路 DMA，被选中的 DMA 根据寄存器的配置开始工作
寄存器位宽： [31：0]
地址： 0x1fd01160
复位值： 0x00000000

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 31:6 | Ask_addr | 26 | R/W | 被选中 DMA 第一个描述符地址的高 26 位，低 6 位为 0；相当于 26 为的 Ask_addr 左移 6 位。 |
| 5 | 保留 | 1 | | |

| | | | | |
|---|---|---|---|---|
| 4 | dma_stop | 1 | R/W | 用户请求停止 DMA 操作；<br>完成当前数据读写操作后，停止操作 |
| 3 | dma_start | 1 | R/W | 可以开始读描述符链的第一个 DMA 描述符；<br>当第一个描述符相关的寄存器读回后，该位清零 |
| 2 | Ask_valid | 1 | R/W | 用户请求将当前 DMA 操作的相关信息写回到指定的内存地址；<br>当用户写回 DMA 操作相关信息后，该位清零。 |
| 1:0 | Dev_num | 2 | R | 2'b00 nand flash<br>2'b01 AC97 read device<br>2'b10 AC97 write device |

说明：

第一个描述符的地址在 ORDER_ADDR_IN 寄存器中，该寄存器由 CPU 来配置，也就是 Ask_addr 左移 6 位后组成了所有描述符寄存器的基地址。

每次 DMA 操作，DMA_ORDER_ADDR 寄存器存放的下个描述符的地址和有效位。

如果 ask_valid=1，表示 CPU 要侦听 DMA 操作，此时要将 DMA 控制器寄存器的值写回到 ask_addr 指向的内存中。

如果 dma_start=1，表示开始 DMA 操作，DMA 先从 ask_addr 指向的内存地址读描述符，然后根据描述符的信息开始执行 DMA 操作。

### 17.3.2 DMA_ORDER_ADDR

中文名： 下一个描述符地址寄存器
寄存器位宽： [31：0]
基地址： Ask_addr<<6
偏移地址： 0x0
复位值： 0x00000000

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 31:1 | dma_order_addr | 31 | R/W | 存储器内部下一个描述符地址寄存器 |
| 0 | Dma_order_en | 1 | R/W | 描述符是否有效信号 |

说明：存储下一个 DMA 描述符的地址，dma_order_en 是下个 DMA 描述符的使能位，如果该位为 1 表示下个描述符有效，该位为 0 表示下个描述符无效，不执行操作，地址 16 字节对齐。在配置 DMA 描述符时，该寄存器存放的是下个描述符的地址，执行完该次 DMA 操作后，通过判断 dma_order_en 信号确定是否开始下次 DMA 操作。

### 17.3.3 DMA_SADDR

中文名： 内存地址寄存器
寄存器位宽： [31：0]
基地址： Ask_addr<<6
偏移地址： 0x4
复位值： 0x00000000

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 31:0 | dma_saddr | 32 | R/W | DMA 操作的内存地址 |

说明：DMA 操作分为：从内存中读数据，保存在 DMA 控制器的缓存中，由 APB 发请求来访问 DMA 缓存中的数据，该寄存器指定了读 ddr2 的地址；从 APB 设备读数据保存在 DMA 缓存中，当 DMA 缓存中的字超过一定数目，就往内存中写，该寄存器指定了写内存的地址。

### 17.3.4 DMA_DADDR

中文名： 设备地址寄存器
寄存器位宽： [31：0]
基地址： Ask_addr<<6

偏移地址： 0x8
复位值： 0x00000000

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 31 | | 1 | R/W | AC97 写使能，“1”表示是写操作 |
| 30 | | 1 | R/W | 0:mono 1: 2 stero |
| 29:28 | | 2 | R/W | AC97 写模式,0: 1byte,1: 2byte,2: 4byte |
| 27:0 | dma_daddr | 32 | R/W | DMA 操作的 APB 设备地址 |

说明：从内存中读数据，保存在 DMA 控制器的缓存中，由 APB 发请求来访问 DMA 缓存中的数据，该寄存器指定了写 APB 设备的地址；从 APB 设备读数据保存在 DMA 缓存中，当 DMA 缓存中的字超过一定数目，就往内存中写，该寄存器指定了读 APB 设备的地址。

### 17.3.5 DMA_LENGTH

中文名： 长度寄存器
寄存器位宽： [31：0]
基地址： Ask_addr<<6
偏移地址： 0xc
复位值： 0x00000000

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 31:0 | dma_length | 32 | R/W | 传输数据长度寄存器 |

说明：代表一块被搬运内容的长度，单位是字。当搬运完 length 长度的字之后，开始下个 step 即下一个循环。开始新的循环，则再次搬运 length 长度的数据。当 step 变为 1，单个 DMA 描述符操作结束，开始读下个描述符。

### 17.3.6 DMA_STEP_LENGTH

中文名： 间隔长度寄存器
寄存器位宽： [31：0]
基地址： Ask_addr<<6
偏移地址： 0x10
复位值： 0x00000000

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 31:0 | dma_step_length | 32 | R/W | 数据传输间隔长度寄存器 |

说明：间隔长度说明两块被搬运内存数据块之间的长度，前一个 step 的结束地址与后一个 step 的开始地址之间的间隔。

### 17.3.7 DMA_STEP_TIMES

中文名： 循环次数寄存器
寄存器位宽： [31：0]
基地址： Ask_addr<<6
偏移地址： 0x14
复位值： 0x00000000

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 31:0 | dma_step_times | 32 | R/W | 数据传输循环次数寄存器 |

说明：循环次数说明在一次 DMA 操作中需要搬运的块的数目。如果只想搬运一个连续的数据块，循环次数寄存器的值可以赋值为 1。

### 17.3.8 DMA_CMD

中文名： 控制寄存器

寄存器位宽： [31：0]
基地址： Ask_addr<<6
偏移地址： 0x18
复位值： 0x00000000

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 14:13 | Dma_cmd | 2 | R/W | 源、目的地址生成方式 |
| 12 | dma_r_w | 1 | R/W | DMA 操作类型，“1”为读 ddr2 写设备，“0”为读设备写 ddr2 |
| 11:8 | dma_write_state | 4 | R/W | DMA 写数据状态 |
| 7:4 | dma_read_state | 4 | R/W | DMA 读数据状态 |
| 3 | dma_trans_over | 1 | R/W | DMA 执行完被配置的所有描述符操作 |
| 2 | dma_single_trans_over | 1 | R/W | DMA 执行完一次描述符操作 |
| 1 | dma_int | 1 | R/W | DMA 中断信号 |
| 0 | dma_int_mask | 1 | R/W | DMA 中断是否被屏蔽掉 |
| 位域 | 位域名称 | 位宽 | 访问 | 描述 |

说明：dma_single_trans_over=1 指一次 DMA 操作执行结束，此时 length=0 且 step_times=1，开始取下个 DMA 操作的描述符。下个 DMA 操作的描述符地址保存在 DMA_ORDER_ADDR 寄存器中，如果 DMA_ORDER_ADDR 寄存器中 dma_order_en=0，则 dma_trans_over=1，整个 dma 操作结束，没有新的描述符要读；如果 dma_order_en=1，则 dma_trans_over 置为 0，开始读下个 dma 描述符。dma_int 为 DMA 的中断，如果没有中断屏蔽，在一次配置的 DMA 操作结束后发生中断。CPU 处理完中断后可以直接将其置低，也可以等到 DMA 进行下次传输时自动置低。dma_int_mask 为对应 dma_int 的中断屏蔽。dma_read_state 说明了 DMA 当前的读状态。dma_write_state 说明了 DMA 当前的写状态。

DMA 写状态(WRITE_STATE[3:0])描述，DMA 包括以下几个写状态：

| Write_state | 【3:0】 | 描述 |
|---|---|---|
| Write_idle | 4’h0 | 写状态正处于空闲状态 |
| W_ddr_wait | 4’h1 | Dma 判断需要执行读设备写内存操作，并发起写内存请求，但是内存还没准备好响应请求，因此 dma 一直在等待内存的响应 |
| Write_ddr | 4’h2 | 内存接收了 dma 写请求，但是还没有执行完写操作 |
| Write_ddr_end | 4’h3 | 内存接收了 dma 写请求，并完成写操作，此时 dma 处于写内存操作完成状态 |
| Write_dma_wait | 4’h4 | Dma 发出将 dma 状态寄存器写回内存的请求，等待内存接收请求 |
| Write_dma | 4’h5 | 内存接收写 dma 状态请求，但是操作还未完成 |
| Write_dma_end | 4’h6 | 内存完成写 dma 状态操作 |
| Write_step_end | 4’h7 | Dma 完成一次 length 长度的操作（也就是说完成一个 step） |

DMA 读状态(READ_STATE[3:0])描述，DMA 包括以下几个读状态：

| Read_state | 【3:0】 | 描述 |
|---|---|---|
| Read_idle | 4’h0 | 读状态正处于空闲状态 |
| Read_ready | 4’h1 | 接收到开始 dma 操作的 start 信号后，进入准备好状态，开始读描述符 |
| Get_order | 4’h2 | 向内存发出读描述符请求，等待内存应答 |
| Read_order | 4’h3 | 内存接收读描述符请求，正在执行读操作 |
| Finish_order_end | 4’h4 | 内存读完 dma 描述符 |
| R_ddr_wait | 4’h5 | Dma 向内存发出读数据请求，等待内存应答 |
| Read_ddr | 4’h6 | 内存接收 dma 读数据请求，正在执行读数据操作 |
| Read_ddr_end | 4’h7 | 内存完成 dma 的一次读数据请求 |
| Read_dev | 4’h8 | Dma 进入读设备状态 |
| Read_dev_end | 4’h9 | 设备返回读数据，结束此次读设备请求 |

| | | |
|---|---|---|
| Read_step_end | 4’ha | 结束一次 step 操作，step times 减 1 |

# 18 UART

## 18.1 概述

1A 集成了四个 UART 核。UART 控制器提供与 MODEM 或其他外部设备串行通信的功能，例如与另外一台计算机，以 RS232 为标准使用串行线路进行通信。 该控制器在设计上能很好地兼容国际工业标准半导体设备 16550A。

## 18.2 UART 控制器结构

UART 控制器有发送和接收模块（Transmitter and Receiver）、MODEM 模块、中断仲裁模块（Interrupt Arbitrator）、访问寄存器模块（Register Access Control），这些模块之间的关系见下图所示。主要模块功能及特征描述如下：

**发送和接收模块**：负责处理数据帧的发送和接收。发送模块是将 FIFO 发送队列中的数据按照设定的格式把并行数据转换为串行数据帧，并通过发送端口送出去。接收模块则监视接收端信号，一旦出现有效开始位，就进行接收，并实现将接收到的异步串行数据帧转换为并行数据，存入 FIFO 接收队列中，同时检查数据帧格式是否有错。UART 的帧结构是通过行控制寄存器（LCR）设置的，发送和接收器的状态被保存在行状态寄存器（LSR）中

**MODEM 模块**：MODEM 控制寄存器（MCR）控制输出信号 DTR 和 RTS 的状态。MODEM 控制模块监视输入信号 DCD,CTS,DSR 和 RI 的线路状态，并将这些信号的状态记录在 MODEM 状态寄存器（MSR）的相对应位中。

**中断仲裁模块**：当任何一种中断条件被满足，并且在中断使能寄存器（IER）中相应位置 1，那么 UART 的中断请求信号 UAT_INT 被置为有效状态。为了减少和外部软件的交互，UART 把中断分为四个级别，并且在中断标识寄存器（IIR）中标识这些中断。四个级别的中断按优先级级别由高到低的排列顺序为，接收线路状态中断；接收数据准备好中断；传送拥有寄存器为空中断；MODEM 状态中断。

**访问寄存器模块**：当 UART 模块被选中时，CPU 可通过读或写操作访问被地址线选中的寄存器。

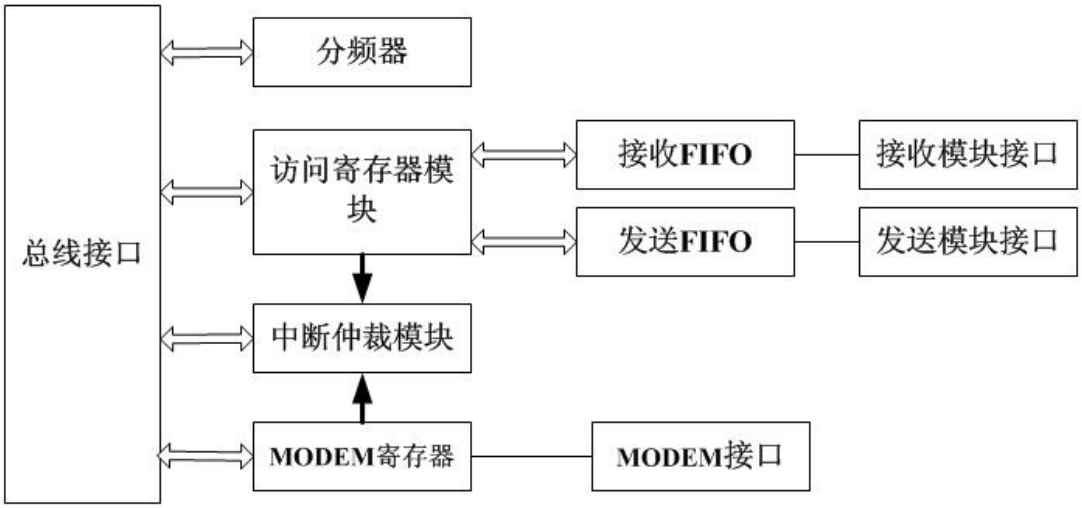



图 17-3 UART控制器结构

## 18.3 UART 寄存器描述

1A 内四个并行工作的 UART 接口，其功能寄存器完全一样，只是访问基址不一样。

- UART0 寄存器物理地址基址为 0x1fe40000。
- UART1 寄存器物理地址基址为 0x1fe44000。
- UART2 寄存器物理地址基址为 0x1fe48000。
- UART3 寄存器物理地址基址为 0x1fe4c000。

### 18.3.1 数据寄存器（DAT）

中文名： 数据传输寄存器
寄存器位宽： [7：0]
偏移量： 0x00
复位值： 0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7:0 | Tx FIFO | 8 | W | 数据传输寄存器 |

### 18.3.2 中断使能寄存器（IER）

中文名： 中断使能寄存器
寄存器位宽： [7：0]
偏移量： 0x01
复位值： 0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7:4 | Reserved | 4 | RW | 保留 |
| 3 | IME | 1 | RW | Modem 状态中断使能 ‘0’ – 关闭 ‘1’ – 打开 |

| | | | | |
|---|---|---|---|---|
| 2 | ILE | 1 | RW | 接收器线路状态中断使能 ‘0’– 关闭‘1’– 打开 |
| 1 | ITxE | 1 | RW | 传输保存寄存器为空中断使能 ‘0’– 关闭 ‘1’ – 打开 |
| 0 | IRxE | 1 | RW | 接收有效数据中断使能 ‘0’– 关闭 ‘1’– 打开 |

### 18.3.3 中断标识寄存器（IIR）

中文名： 中断源寄存器
寄存器位宽： [7：0]
偏移量： 0x02
复位值： 0xc1

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7:4 | Reserved | 4 | R | 保留 |
| 3:1 | II | 3 | R | 中断源表示位，详见下表 |
| 0 | INTp | 1 | R | 中断表示位 |

中断控制功能表：

| Bit 3 | Bit 2 | Bit 1 | 优先级 | 中断类型 | 中断源 | 中断复位控制 |
|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 1st | 接收线路状态 | 奇偶、溢出或帧错误，或打断中断 | 读 LSR |
| 0 | 1 | 0 | 2nd | 接收到有效数据 | FIFO 的字符个数达到 trigger 的水平 | FIFO 的字符个数低于 trigger 的值 |
| 1 | 1 | 0 | 2nd | 接收超时 | 在 FIFO 至少有一个字符，但在 4 个字符时间内没有任何操作，包括读和写操作 | 读接收 FIFO |
| 0 | 0 | 1 | 3rd | 传输保存寄存器为空 | 传输保存寄存器为空 | 写数据到 THR 或者多 IIR |
| 0 | 0 | 0 | 4th | Modem 状态 | CTS, DSR, RI or DCD. | 读 MSR |

### 18.3.4 FIFO 控制寄存器（FCR）

中文名： FIFO 控制寄存器
寄存器位宽： [7：0]
偏移量： 0x02
复位值： 0xc0

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7:6 | TL | 2 | W | 接收 FIFO 提出中断申请的 trigger 值 ‘00’ – 1 字节 ‘01’ – 4 字节 ‘10’ – 8 字节 ‘11’ – 14 字节 |
| 5:3 | Reserved | 3 | W | 保留 |
| 2 | Txset | 1 | W | ‘1’ 清除发送 FIFO 的内容，复位其逻辑 |
| 1 | Rxset | 1 | W | ‘1’ 清除接收 FIFO 的内容，复位其逻辑 |
| 0 | Reserved | 1 | W | 保留 |

### 18.3.5 线路控制寄存器（LCR）

中文名：　　线路控制寄存器
寄存器位宽：　[7：0]
偏移量：　　0x03
复位值：　　0x03

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7 | dlab | 1 | RW | 分频锁存器访问位<br>‘1’ – 访问操作分频锁存器<br>‘0’ – 访问操作正常寄存器 |
| 6 | bcb | 1 | RW | 打断控制位<br>‘1’ – 此时串口的输出被置为 0(打断状态).<br>‘0’ – 正常操作 |
| 5 | spb | 1 | RW | 指定奇偶校验位<br>‘0’ – 不用指定奇偶校验位<br>‘1’ – 如果 LCR[4]位是 1 则传输和检查奇偶校验位为 0。如果 LCR[4]位是 0 则传输和检查奇偶校验位为 1。 |
| 4 | eps | 1 | RW | 奇偶校验位选择<br>‘0’ – 在每个字符中有奇数个 1（包括数据和奇偶校验位）<br>‘1’ –在每个字符中有偶数个 1 |
| 3 | pe | 1 | RW | 奇偶校验位使能<br>‘0’ – 没有奇偶校验位<br>‘1’ – 在输出时生成奇偶校验位，输入则判断奇偶校验位 |
| 2 | sb | 1 | RW | 定义生成停止位的位数<br>‘0’ – 1 个停止位<br>‘1’ – 在 5 位字符长度时是 1.5 个停止位，其他长度是 2 个停止位 |
| 1:0 | bec | 2 | RW | 设定每个字符的位数<br>‘00’ – 5 位　‘01’ – 6 位<br>‘10’ – 7 位　‘11’ – 8 位 |

### 18.3.6 MODEM 控制寄存器（MCR）

中文名：　　Modem 控制寄存器
寄存器位宽：　[7：0]
偏移量：　　0x04
复位值：　　0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7:5 | Reserved | 3 | W | 保留 |
| 4 | Loop | 1 | W | 回环模式控制位<br>‘0’ – 正常操作<br>‘1’ –回环模式。在在回环模式中，TXD 输出一直为 1，输出移位寄存器直接连到输入移位寄存器中。其他连接如下。<br>DTR ➔ DSR<br>RTS ➔ CTS<br>Out1 ➔ RI<br>Out2 ➔ DCD |
| 3 | OUT2 | 1 | W | 在回环模式中连到 DCD 输入 |
| 2 | OUT1 | 1 | W | 在回环模式中连到 RI 输入 |

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 1 | RTSC | 1 | W | RTS 信号控制位 |
| 0 | DTRC | 1 | W | DTR 信号控制位 |

### 18.3.7 线路状态寄存器（LSR）

中文名：　　线路状态寄存器
寄存器位宽：　[7：0]
偏移量：　　0x05
复位值：　　0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7 | ERROR | 1 | R | 错误表示位<br>‘1’ – 至少有奇偶校验位错误，帧错误或打断中断的一个。<br>‘0’ – 没有错误 |
| 6 | TE | 1 | R | 传输为空表示位<br>‘1’ – 传输 FIFO 和传输移位寄存器都为空。给传输 FIFO 写数据时清零<br>‘0’ – 有数据 |
| 5 | TFE | 1 | R | 传输 FIFO 位空表示位<br>‘1’ – 当前传输 FIFO 为空，给传输 FIFO 写数据时清零<br>‘0’ – 有数据 |
| 4 | BI | 1 | R | 打断中断表示位<br>‘1’ –接收到 起始位＋数据＋奇偶位＋停止位都是 0，即有打断中断<br>‘0’ – 没有打断 |
| 3 | FE | 1 | R | 帧错误表示位<br>‘1’ – 接收的数据没有停止位<br>‘0’ – 没有错误 |
| 2 | PE | 1 | R | 奇偶校验位错误表示位<br>‘1’ – 当前接收数据有奇偶错误<br>‘0’ – 没有奇偶错误 |
| 1 | OE | 1 | R | 数据溢出表示位<br>‘1’ – 有数据溢出<br>‘0’ – 无溢出 |
| 0 | DR | 1 | R | 接收数据有效表示位<br>‘0’ – 在 FIFO 中无数据<br>‘1’ – 在 FIFO 中有数据 |

对这个寄存器进行读操作时，LSR[4:1]和 LSR[7]被清零，LSR[6:5]在给传输 FIFO 写数据时清零，LSR[0]则对接收 FIFO 进行判断。

### 18.3.8 MODEM 状态寄存器 （MSR）

中文名：　　Modem 状态寄存器
寄存器位宽：　[7：0]
偏移量：　　0x06
复位值：　　0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7 | CDCD | 1 | R | DCD 输入值的反，或者在回环模式中连到 Out2 |

| | | | | |
|---|---|---|---|---|
| 6 | CRI | 1 | R | RI 输入值的反，或者在回环模式中连到 OUT1 |
| 5 | CDSR | 1 | R | DSR 输入值的反，或者在回环模式中连到 DTR |
| 4 | CCTS | 1 | R | CTS 输入值的反，或者在回环模式中连到 RTS |
| 3 | DDCD | 1 | R | DDCD 指示位 |
| 2 | TERI | 1 | R | RI 边沿检测。RI 状态从低到高变化 |
| 1 | DDSR | 1 | R | DDSR 指示位 |
| 0 | DCTS | 1 | R | DCTS 指示位 |

### 18.3.9 分频锁存器

中文名：　　分频锁存器 1
寄存器位宽：[7：0]
偏移量：　　0x00
复位值：　　0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7:0 | LSB | 8 | RW | 存放分频锁存器的低 8 位 |

中文名：　　分频锁存器 2
寄存器位宽：[7：0]
偏移量：　　0x01
复位值：　　0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7:0 | MSB | 8 | RW | 存放分频锁存器的高 8 位 |

模块中被分频时钟 clock_a 的频率是 DDR_clk 频率的的一半（DDR_clk 配置见 29 章）；假设分频锁存器的值为 prescale，波特率为 clock_baud（波特率根据用户需要和外部 UART 连接特性确定），则应满足如下关系：

Prcescale = clock_a/(16*clock_baud)

或者 Prcescale = DDR_clk/(32*clock_baud)

# 19 CAN

## 19.1 概述

1A 集成了两路 CAN 接口控制器。CAN 总线是由发送数据线 TX 和接收数据线 RX 构成的串行总线，可发送和接收数据。器件与器件之间进行双向传送，最高传送速率 1Mbps。

两个 CAN 总线控制器的中断连接到中断控制的第一组寄存器中，其中 can0 的中断对应 bit6，can1 的中断对应 bit7。参考第 15 章的说明。

CAN0 总线控制器的寄存器基地址为 0x1fe50000 开始的 16KB;

CAN1 总线控制器的寄存器基地址为 0x1fe54000 开始的 16KB。

## 19.2 CAN 控制器结构

下图为 CAN 主控制器的结构，主要模块有 APB 总线接口、位流处理单元、位时序逻辑、错误管理逻辑、接收滤波和数据缓存区。

1. **APB总线接口：**接收APB总线的指令和返回数据。
2. **位流处理单元：**实现对发送缓存器、接收FIFO和CAN总线之间数据流的控制，同时还执行错误检测、总线仲裁、数据填充和错误处理等功能。
3. **位时序逻辑：**监视串口的CAN 总线和处理与总线有关的位时序。还提供了可编程的时间段来补偿传播延迟时间、相位转换（例如由于振荡漂移）和定义采样点和一位时间内的采样次数。
4. **错误管理逻辑：**判断传输的CRC错误并对错误计数。
5. **接收滤波：**把接收的识别码的内容相比较以决定是否接收信息
6. **数据缓存区：**接收缓冲器是验收滤波器和CPU 之间的接口，用来储存从CAN 总线上接收和接收的信息。接收缓冲器（13 个字节）作为接收FIFO（长 64 字节）的一个窗口可被CPU 访问

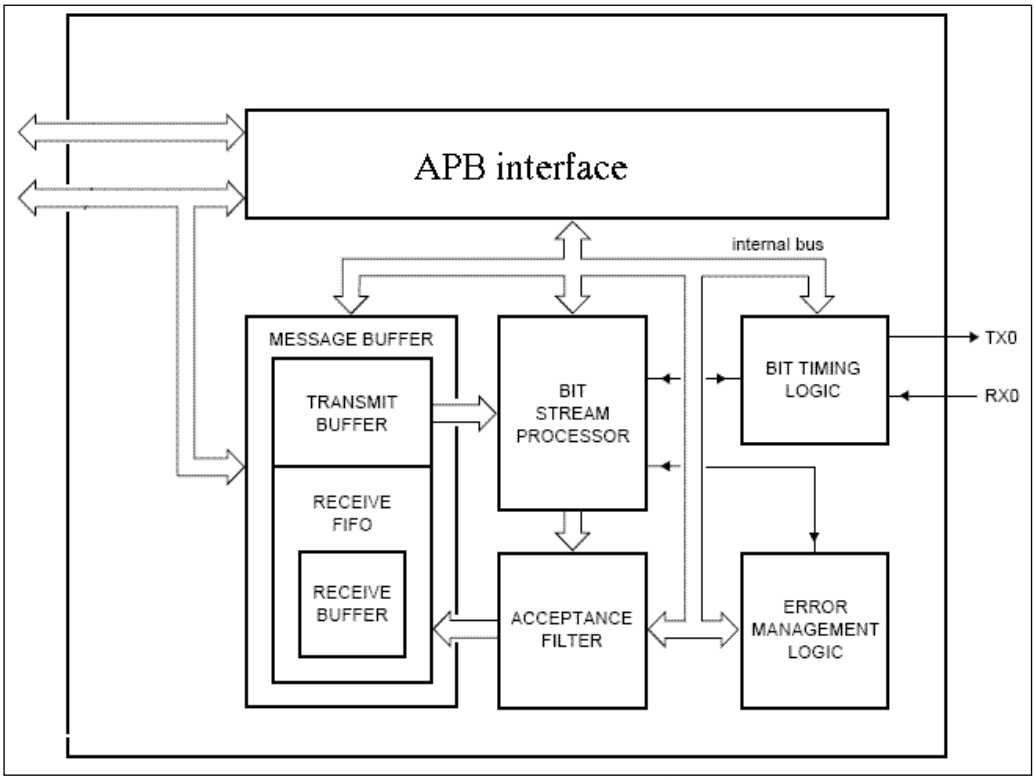



图 19-1 CAN主控制器结构

CAN 支持两种工作模式，即标准模式和扩展模式。工作模式通过命令寄存器中的 CAN 模式位来选择。复位默认是标准模式。

## 19.3 标准模式

### 19.3.1 标准模式地址表

地址区包括控制段和信息缓冲区，控制段在初始化载入是可被编程来配置通讯参数的，应发送的信息会被写入发送缓冲器，成功接收信息后，微控制器从接收缓冲器中读取接收的信息，然后释放空间以做下一步应用。

初始载入后，寄存器的验收代码，验收屏蔽，总线定时寄存器 0 和 1 以及输出控制就不能改变了。只有控制寄存器的复位位被置高时，才可以访问这些寄存器。在复位模式和工作模式两种不同的模式中，访问寄存器是不同的。当硬件复位或控制器掉线，状态寄存器的总线状态位时会自动进入复位模式。工作模式是通过置位控制寄存器的复位请求位激活的。

| CAN 地址 | 段 | 工作模式 | | 复位模式 | |
|---|---|---|---|---|---|
| | | 读 | 写 | 读 | 写 |
| 0 | | 控制 | 控制 | 控制 | 控制 |
| 1 | | FF | 命令 | FF | 命令 |
| 2 | | 状态 | — | 状态 | — |
| 3 | | FF | — | 中断 | — |
| 4 | | FF | — | 验收代码 | 验收代码 |
| 5 | | FF | — | 验收屏蔽 | 验收屏蔽 |
| 6 | 控制 | FF | — | 总线定时 0 | 总线定时 0 |

| | | | | | |
|---|---|---|---|---|---|
| 7 | | FF | — | 总线定时 1 | 总线定时 1 |
| 8 | | 保留 | 保留 | 保留 | 保留 |
| 9 | | 保留 | 保留 | 保留 | 保留 |
| 10 | | ID(10-3) | ID(10-3) | FF | — |
| 11 | | ID(2-0), RTR,DLC | ID(2-0), RTR,DLC | FF | — |
| 12 | | 数据字节 1 | 数据字节 1 | FF | — |
| 13 | | 数据字节 2 | 数据字节 2 | FF | — |
| 14 | | 数据字节 3 | 数据字节 3 | FF | — |
| 15 | | 数据字节 4 | 数据字节 4 | FF | — |
| 16 | | 数据字节 5 | 数据字节 5 | FF | — |
| 17 | | 数据字节 6 | 数据字节 6 | FF | — |
| 18 | | 数据字节 7 | 数据字节 7 | FF | — |
| 19 | 发送缓冲器 | 数据字节 8 | 数据字节 8 | FF | — |
| 20 | | ID(10-3) | ID(10-3) | FF | — |
| 21 | | ID(2-0), RTR,DLC | ID(2-0), RTR,DLC | FF | — |
| 22 | | 数据字节 1 | 数据字节 1 | FF | — |
| 23 | | 数据字节 2 | 数据字节 2 | FF | — |
| 24 | | 数据字节 3 | 数据字节 3 | FF | — |
| 25 | | 数据字节 4 | 数据字节 4 | FF | — |
| 26 | | 数据字节 5 | 数据字节 5 | FF | — |
| 27 | | 数据字节 6 | 数据字节 6 | FF | — |
| 28 | | 数据字节 7 | 数据字节 7 | FF | — |
| 29 | 接收缓冲器 | 数据字节 8 | 数据字节 8 | FF | — |

### 19.3.2 控制寄存器（CR）

中文名： 控制寄存器
寄存器位宽： [7：0]
偏移量： 0x00
复位值： 0x01

读此位的值总是逻辑 1。在硬启动或总线状态位设置为 1（总线关闭）时，复位请求位被置为 1。如果这些位被软件访问，其值将发生变化而且会影响内部时钟的下一个上升沿，在外部复位期间微控制器不能把复位请求位置为 0。如果把复位请求位设为 0，微控制器就必须检查这一位以保证外部复位引脚不保持为低。复位请求位的变化是同内部分频时钟同步的。读复位请求位能够反映出这种同步状态。

复位请求位被设为 0 后控制器将会等待

a) 一个总线空闲信号（11 个弱势位），如果前一次复位请求是硬件复位或 CPU 初始复位。

b) 128 个总线空闲，如果前一次复位请求是 CAN 控制器在重新进入总线开启模式前初始化总线造成的。

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7：5 | Reserve | 3 | — | 保留 |
| 4 | OIE | 1 | RW | 溢出中断使能 |
| 3 | EIE | 1 | RW | 错误中断使能 |
| 2 | TIE | 1 | RW | 发送中断使能 |
| 1 | RIE | 1 | RW | 接收中断使能 |
| 0 | RR | 1 | RW | 复位请求 |

### 19.3.3 命令寄存器（CMR）

中文名：　　命令寄存器
寄存器位宽：[7：0]
偏移量：　　0x01
复位值：　　0x00

命令寄存器对微控制器来说是只写存储器如果去读这个地址返回值是 1111 1111

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7 | EFF | 1 | W | 扩展模式 |
| 6：5 | Reserve | 2 | — | 保留 |
| 4 | GTS | 1 | W | 睡眠 |
| 3 | CDO | 1 | W | 清除数据溢出 |
| 2 | RRB | 1 | W | 释放接收缓冲器 |
| 1 | AT | 1 | W | 中止发送 |
| 0 | TR | 1 | W | 发送请求 |

### 19.3.4 状态寄存器（SR）

中文名：　　状态寄存器
寄存器位宽：[7：0]
偏移量：　　0x02
复位值：　　0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7 | BS | 1 | R | 总线状态 |
| 6 | ES | 1 | R | 出错状态 |
| 5 | TS | 1 | R | 发送状态 |
| 4 | RS | 1 | R | 接收状态 |
| 3 | TCS | 1 | R | 发送完毕状态 |
| 2 | TBS | 1 | R | 发送缓存器状态 |
| 1 | DOS | 1 | R | 数据溢出状态 |
| 0 | RBS | 1 | R | 接收缓存器状态 |

### 19.3.5 中断寄存器（IR）

中文名：　　中断寄存器
寄存器位宽：[7：0]
偏移量：　　0x03
复位值：　　0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| 7：5 | Reserved | 1 | R | 保留 |
| 4 | WUI | 1 | R | 唤醒中断 |
| 3 | DOI | 1 | R | 数据溢出中断 |
| 2 | EI | 1 | R | 错误中断 |
| 1 | TI | 1 | R | 发送中断 |
| 0 | RI | 1 | R | 接收中断 |

### 19.3.6 验收代码寄存器（ACR）

中文名：　　验收代码寄存器
寄存器位宽：　[7：0]
偏移量：　　0x04
复位值：　　0x00

在复位情况下，该寄存器是可以读写的。

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7:0 | AC | 8 | RW | ID 验收代码 |

### 19.3.7 验收屏蔽寄存器（AMR）

中文名：　　验收屏蔽寄存器
寄存器位宽：　[7：0]
偏移量：　　0x05
复位值：　　0x00

验收代码位 AC 和信息识别码的高 8 位 ID.10-ID.3 相等且与验收屏蔽位 AM 的相应位相或为 1 时数据可以接收。在复位情况下，该寄存器是可以读写的。

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7:0 | AM | 8 | RW | ID 屏蔽位 |

### 19.3.8 发送缓冲区列表

缓冲器是用来存储微控制器要 CAN 控制器发送的信息，它被分为描述符区和数据区。发送缓冲器的读/写只能由微控制器在工作模式下完成，在复位模式下读出的值总是 FF。

| 地址 | 区 | 名称 | 数据位 |
|---|---|---|---|
| 10 | 发送缓冲器 | 识别码字节 1 | ID(10-3) |
| 11 | | 识别码字节 2 | ID(2-0)，RTR,DLC |
| 12 | | TX 数据 1 | TX 数据 1 |
| 13 | | TX 数据 2 | TX 数据 2 |
| 14 | | TX 数据 3 | TX 数据 3 |
| 15 | | TX 数据 4 | TX 数据 4 |
| 16 | | TX 数据 5 | TX 数据 5 |
| 17 | | TX 数据 6 | TX 数据 6 |
| 18 | | TX 数据 7 | TX 数据 7 |
| 19 | | TX 数据 8 | TX 数据 8 |

### 19.3.9 接收缓冲区列表

接收缓冲区的配置和发送缓冲区的一样，只是地址变为 20－29。

## 19.4 扩展模式

### 19.4.1 扩展模式地址表

| CAN 地址 | 工作模式 | | 复位模式 | |
|---|---|---|---|---|
| | 读 | 写 | 读 | 写 |
| 0 | 控制 | 控制 | 控制 | 控制 |
| 1 | 0 | 命令 | 0 | 命令 |
| 2 | 状态 | — | 状态 | — |
| 3 | 中断 | — | 中断 | — |
| 4 | 中断使能 | 中断使能 | 中断使能 | 中断使能 |
| 5 | — | — | 验收屏蔽 | 验收屏蔽 |
| 6 | 总线定时 0 | — | 总线定时 0 | 总线定时 0 |
| 7 | 总线定时 1 | — | 总线定时 1 | 总线定时 1 |
| 8 | 保留 | 保留 | 保留 | 保留 |
| 9 | 保留 | 保留 | 保留 | 保留 |
| 10 | 保留 | 保留 | 保留 | 保留 |
| 11 | 仲裁丢失捕捉 | — | 仲裁丢失捕捉 | — |
| 12 | 错误代码捕捉 | — | 错误代码捕捉 | — |
| 13 | 错误警报限制 | — | 错误警报限制 | — |
| 14 | RX 错误计数器 | — | RX 错误计数器 | — |
| 15 | TX 错误计数器 | — | TX 错误计数器 | — |
| 16 | RX 帧信息 | TX 帧信息 | 验收代码 0 | 验收代码 0 |
| 17 | RX 识别码 1 | TX 识别码 1 | 验收代码 1 | 验收代码 1 |
| 18 | RX 识别码 2 | TX 识别码 2 | 验收代码 2 | 验收代码 2 |
| 19 | RX 识别码 3 | TX 识别码 3 | 验收代码 3 | 验收代码 3 |
| 20 | RX 识别码 4 | TX 识别码 4 | 验收屏蔽 0 | 验收屏蔽 0 |
| 21 | RX 数据 1 | TX 数据 1 | 验收屏蔽 1 | 验收屏蔽 1 |
| 22 | RX 数据 2 | TX 数据 2 | 验收屏蔽 2 | 验收屏蔽 2 |
| 23 | RX 数据 3 | TX 数据 3 | 验收屏蔽 3 | 验收屏蔽 3 |
| 24 | RX 数据 4 | TX 数据 4 | — | — |
| 25 | RX 数据 5 | TX 数据 5 | — | — |
| 26 | RX 数据 6 | TX 数据 6 | — | — |
| 27 | RX 数据 7 | TX 数据 7 | — | — |
| 28 | RX 数据 8 | TX 数据 8 | — | — |
| 29 | RX 信息计数器 | — | RX 信息计数器 | — |

### 19.4.2 模式寄存器（MOD）

中文名： 模式寄存器
寄存器位宽： [7：0]
偏移量： 0x00
复位值： 0x01

读此位的值总是逻辑 1。在硬启动或总线状态位设置为 1（总线关闭）时，复位请求位被置为 1。如果这些位被软件访问，其值将发生变化而且会影响内部时钟的下一个上升沿，在外部复位期间微控制器不能把复位请求位置为 0。如果把复位请求位设为 0，微控制器就必须检查这一位以保证外部复位引脚不保持为

低。复位请求位的变化是同内部分频时钟同步的。读复位请求位能够反映出这种同步状态。

复位请求位被设为 0 后控制器将会等待

a) 一个总线空闲信号（11 个弱势位），如果前一次复位请求是硬件复位或 CPU 初始复位。

b) 128 个总线空闲，如果前一次复位请求是 CAN 控制器在重新进入总线开启模式前初始化总线造成的。

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7：5 | Reserve | 3 | — | 保留 |
| 4 | SM | 1 | RW | 睡眠模式 |
| 3 | AFM | 1 | RW | 单/双滤波模式 |
| 2 | STM | 1 | RW | 正常工作模式 |
| 1 | LOM | 1 | RW | 只听模式 |
| 0 | RM | 1 | RW | 复位模式 |

### 19.4.3 命令寄存器（CMR）

中文名： 命令寄存器
寄存器位宽： [7：0]
偏移量： 0x01
复位值： 0x00

命令寄存器对微控制器来说是只写存储器如果去读这个地址返回值是 1111 1111

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7 | EFF | 1 | W | 扩展模式 |
| 6：5 | Reserve | 2 | — | 保留 |
| 4 | SRR | 1 | W | 自接收请求 |
| 3 | CDO | 1 | W | 清除数据溢出 |
| 2 | RRB | 1 | W | 释放接收缓冲器 |
| 1 | AT | 1 | W | 中止发送 |
| 0 | TR | 1 | W | 发送请求 |

### 19.4.4 状态寄存器（SR）

中文名： 状态寄存器
寄存器位宽： [7：0]
偏移量： 0x02
复位值： 0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7 | BS | 1 | R | 总线状态 |
| 6 | ES | 1 | R | 出错状态 |
| 5 | TS | 1 | R | 发送状态 |
| 4 | RS | 1 | R | 接收状态 |
| 3 | TCS | 1 | R | 发送完毕状态 |

| | | | | |
|---|---|---|---|---|
| 2 | TBS | 1 | R | 发送缓存器状态 |
| 1 | DOS | 1 | R | 数据溢出状态 |
| 0 | RBS | 1 | R | 接收缓存器状态 |

### 19.4.5 中断寄存器（IR）

中文名：　　中断寄存器
寄存器位宽：　[7：0]
偏移量：　　0x03
复位值：　　0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7 | BEI | 1 | R | 总线错误中断 |
| 6 | ALI | 1 | R | 仲裁丢失中断 |
| 5 | EPI | 1 | R | 错误消极中断 |
| 4 | WUI | 1 | R | 唤醒中断 |
| 3 | DOI | 1 | R | 数据溢出中断 |
| 2 | EI | 1 | R | 错误中断 |
| 1 | TI | 1 | R | 发送中断 |
| 0 | RI | 1 | R | 接收中断 |

### 19.4.6 中断使能寄存器（IER）

中文名：　　中断使能寄存器
寄存器位宽：　[7：0]
偏移量：　　0x04
复位值：　　0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7 | BEIE | 1 | RW | 总线错误中断使能 |
| 6 | ALIE | 1 | RW | 仲裁丢失中断使能 |
| 5 | EPIE | 1 | RW | 错误消极中断使能 |
| 4 | WUIE | 1 | RW | 唤醒中断使能 |
| 3 | DOIE | 1 | RW | 数据溢出中断使能 |
| 2 | EIE | 1 | RW | 错误中断使能 |
| 1 | TIE | 1 | RW | 发送中断使能 |
| 0 | RIE | 1 | RW | 接收中断使能 |

### 19.4.7 仲裁丢失捕捉寄存器（IER）

中文名：　　仲裁丢失捕捉寄存器
寄存器位宽：　[7：0]
偏移量：　　0xB
复位值：　　0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7：5 | — | 3 | R | 保留 |
| 4 | BITNO4 | 1 | R | 第四位 |
| 3 | BITNO3 | 1 | R | 第三位 |
| 2 | BITNO2 | 1 | R | 第二位 |
| 1 | BITNO1 | 1 | R | 第一位 |

| | | | | |
|---|---|---|---|---|
| 0 | BITNO0 | 1 | R | 第零位 |

| 位 | | | | | 十进制值 | 功能 |
|---|---|---|---|---|---|---|
| ALC. 4 | ALC. 3 | ALC. 2 | ALC. 1 | ALC. 0 | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 仲裁丢失在识别码的bit1 |
| 0 | 0 | 0 | 0 | 1 | 1 | 仲裁丢失在识别码的bit2 |
| 0 | 0 | 0 | 1 | 0 | 2 | 仲裁丢失在识别码的bit3 |
| 0 | 0 | 0 | 1 | 1 | 3 | 仲裁丢失在识别码的bit4 |
| 0 | 0 | 1 | 0 | 0 | 4 | 仲裁丢失在识别码的bit5 |
| 0 | 0 | 1 | 0 | 1 | 5 | 仲裁丢失在识别码的bit6 |
| 0 | 0 | 1 | 1 | 0 | 6 | 仲裁丢失在识别码的bit7 |
| 0 | 0 | 1 | 1 | 1 | 7 | 仲裁丢失在识别码的bit8 |
| 0 | 1 | 0 | 0 | 0 | 8 | 仲裁丢失在识别码的bit9 |
| 0 | 1 | 0 | 0 | 1 | 9 | 仲裁丢失在识别码的bit10 |
| 0 | 1 | 0 | 1 | 0 | 10 | 仲裁丢失在识别码的bit11 |
| 0 | 1 | 0 | 1 | 1 | 11 | 仲裁丢失在SRTR位 |
| 0 | 1 | 1 | 0 | 0 | 12 | 仲裁丢失在IDE位 |
| 0 | 1 | 1 | 0 | 1 | 13 | 仲裁丢失在识别码的bit12 |
| 0 | 1 | 1 | 1 | 0 | 14 | 仲裁丢失在识别码的bit13 |
| 0 | 1 | 1 | 1 | 1 | 15 | 仲裁丢失在识别码的bit14 |
| 1 | 0 | 0 | 0 | 0 | 16 | 仲裁丢失在识别码的bit15 |
| 1 | 0 | 0 | 0 | 1 | 17 | 仲裁丢失在识别码的bit16 |
| 1 | 0 | 0 | 1 | 0 | 18 | 仲裁丢失在识别码的bit17 |
| 1 | 0 | 0 | 1 | 1 | 19 | 仲裁丢失在识别码的bit18 |
| 1 | 0 | 1 | 0 | 0 | 20 | 仲裁丢失在识别码的bit19 |
| 1 | 0 | 1 | 0 | 1 | 21 | 仲裁丢失在识别码的bit20 |
| 1 | 0 | 1 | 1 | 0 | 22 | 仲裁丢失在识别码的bit21 |
| 1 | 0 | 1 | 1 | 1 | 23 | 仲裁丢失在识别码的bit22 |
| 1 | 1 | 0 | 0 | 0 | 24 | 仲裁丢失在识别码的bit23 |
| 1 | 1 | 0 | 0 | 1 | 25 | 仲裁丢失在识别码的bit24 |
| 1 | 1 | 0 | 1 | 0 | 26 | 仲裁丢失在识别码的bit25 |
| 1 | 1 | 0 | 1 | 1 | 27 | 仲裁丢失在识别码的bit26 |
| 1 | 1 | 1 | 0 | 0 | 28 | 仲裁丢失在识别码的bit27 |
| 1 | 1 | 1 | 0 | 1 | 29 | 仲裁丢失在识别码的bit28 |
| 1 | 1 | 1 | 1 | 0 | 30 | 仲裁丢失在识别码的bit29 |
| 1 | 1 | 1 | 1 | 1 | 31 | 仲裁丢失在RTR位 |

### 19.4.8 错误警报限制寄存器（EMLR）

中文名： 错误警报限制寄存器
寄存器位宽： [7：0]
偏移量： 0xD
复位值： 0x60

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7：0 | EML | 8 | RW | 错误警报阀值 |

### 19.4.9 RX 错误计数寄存器（RXERR）

中文名：　　RX 错误计数寄存器
寄存器位宽：　[7：0]
偏移量：　　0xE
复位值：　　0x60

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7：0 | RXERR | 8 | R | 接收错误计数 |

### 19.4.10 TX 错误计数寄存器（TXERR）

中文名：　　TX 错误计数寄存器
寄存器位宽：　[7：0]
偏移量：　　0xF
复位值：　　0x60

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7：0 | TXERR | 8 | R | 发送错误计数 |

### 19.4.11 验收滤波器

在验收滤波器的帮助下，只有当接收信息中的识别位和验收滤波器预定义的值相等时，CAN 控制器才允许将已接收信息存入 RXFIFO。验收滤波器由验收代码寄存器和验收屏蔽寄存器定义。在模式寄存器中选择单滤波器模式或者双滤波器模式。具体的配置可以参考 SJA1000 的数据手册。

### 19.4.12 RX 信息计数寄存器（RMCR）

中文名：　　RX 信息计数寄存器
寄存器位宽：　[7：0]
偏移量：　　0x1D
复位值：　　0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7：0 | RMCR | 8 | R | 接收的数据帧计数器 |

## 19.5　公共寄存器

### 19.5.1 总线定时寄存器 0（BTR0）

中文名：　　总线定时寄存器
寄存器位宽：　[7：0]
偏移量：　　0x06
复位值：　　0x00

注：在复位模式是可以读写的，工作模式是只读的

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7：6 | SJW | 8 | RW | 同步跳转宽度 |
| 5：0 | BRP | 8 | RW | 波特率分频系数 |

### 19.5.2 总线定时寄存器 1（BTR1）

中文名：　　总线定时寄存器 1
寄存器位宽：　[7：0]

偏移量：　　0x07
复位值：　　0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7 | SAM | 1 | RW | 为 1 时三次采样，否则是一次采用 |
| 6：4 | TESG2 | 3 | RW | 一个 bit 中的时间段 2 的计数值 |
| 3：0 | TSEG1 | 4 | RW | 一个 bit 中的时间段 1 的计数值 |

### 19.5.3 输出控制寄存器（OCR）

中文名：　　输出控制寄存器
寄存器位宽：　[7：0]
偏移量：　　0x08
复位值：　　0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7：0 | OCR | 8 | RW | 保留 |

# 20 AC97

## 20.1 概述

在系统里一个 AC97 应用系统如 图 19-1 所示。在一个片上系统中,与 AC97 控制器相连的有 3 部分:一是外设总线,接收来自微处理器的控制信息以及配置信息;二是 AC97 Codec,多媒体数字信号编解码器,该解码器对 PCM 信号进行调制,输出人耳接受的模拟声音或者把真实的声音转换为 PCM 信号,转换通过 D/A 转换器实现;三是 DMA 引擎,通过 DMA 的方式写或读 AC97 控制器内部的 FIFO,实现 PCM 音频数据的不间断操作。DMA 是通过微处理器配置的,从处理器设定的内存区域搬运数据给 FIFO 或者把 FIFO 的数据搬运到设定的内存区域。

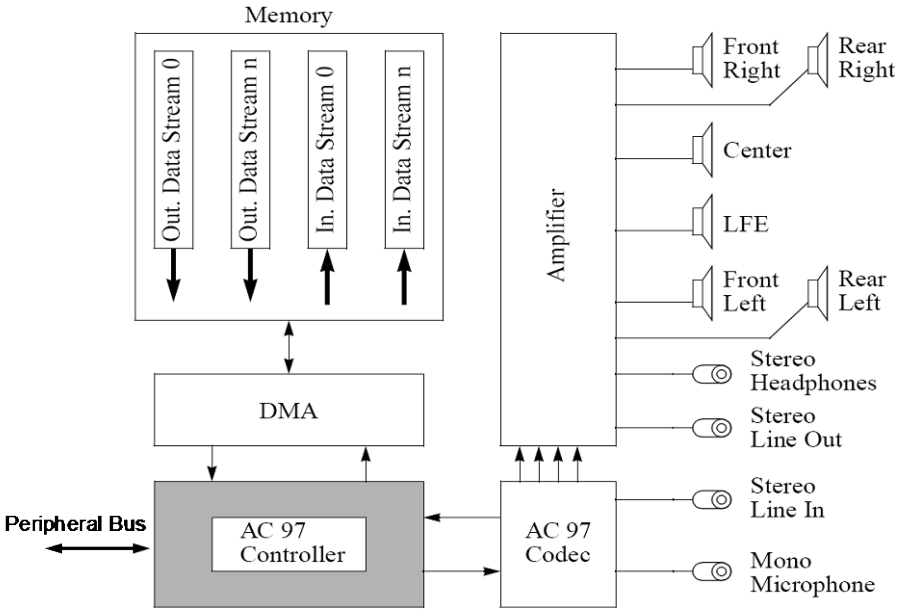



图 20-1 AC97应用系统

## 20.2 AC97 控制器寄存器

本模块寄存器物理地址基址为 0x1fe74000。

| 寄存器名 | 宽度 | 偏移量 | 描述 |
|---|---|---|---|
| CSR | 2 | 0x00 | 配置状态寄存器 |
| OCC0 | 24 | 0x04 | 输出通道配置寄存器 0 |
| OCC1 | 24 | 0x08 | 保留 |
| OCC2 | 24 | 0x0c | 保留 |
| ICC | 24 | 0x10 | 输入通道配置寄存器 |
| CODEC_ID | 32 | 0x14 | Codec ID 寄存器 |
| CRAC | 32 | 0x18 | Codec 寄存器访问命令 |
| OC0 | 20 | 0x20 | 输出声道 0 |

| 寄存器名 | 宽度 | 偏移量 | 描述 |
|---|---|---|---|
| OC1 | 20 | 0x24 | 输出声道 1 |
| OC2 | 20 | 0x28 | 保留 |
| OC3 | 20 | 0x2c | 保留 |
| OC4 | 20 | 0x30 | 保留 |
| OC5 | 20 | 0x34 | 保留 |
| OC6 | 20 | 0x38 | 保留 |
| OC7 | 20 | 0x3c | 保留 |
| OC8 | 20 | 0x40 | 保留 |
| IC0 | 20 | 0x44 | 保留 |
| IC1 | 20 | 0x48 | 保留 |
| IC2 | 20 | 0x4c | 输入声道 2 |
| INTRAW | 32 | 0x54 | 中断状态寄存器 |
| INTM | 32 | 0x58 | 中断掩膜 |
| INTS | 32 | 0x5c | 保留 |

### 20.2.1 CSR 寄存器

中文名： 配置状态寄存器
寄存器位宽： [31：0]
偏移量： 0x00
复位值： 0x00000000

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 31:2 | Reserved | 30 | RO | 保留 |
| 1 | RESUME | 1 | R/W | 挂起，读此位返回现在 AC97 子系统的状态<br>1：AC97 子系统挂起<br>0：正常工作状态<br>在挂起状态下，写入 1 到该位，将会开始恢复操作。 |
| 0 | RST_FORCE | 1 | W | AC97 冷启动<br>写入 1 会导致 AC97 Codec 冷启动 |

### 20.2.2 OCC 寄存器

中文名： 输出通道配置寄存器
寄存器位宽： [31：0]
偏移量： 0x04
复位值： 0x00004141

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 31:24 | Reserved | 8 | R/W | 保留 |
| 23:16 | Reserved | 8 | R/W | 保留 |
| 15:8 | OC1_CFG_R | 8 | R/W | 输出通道 1：右声道配置。 |
| 7:0 | OC0_CFG_L | 8 | R/W | 输出通道 0：左声道配置。 |

### 20.2.3 ICC 寄存器

中文名： 输入通道配置寄存器
寄存器位宽： [31：0]
偏移量： 0x10
复位值： 0x00410000

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 31:24 | Reserved | 8 | R/W | 保留 |
| 23:16 | IC_CFG_MIC | 8 | R/W | 输入通道 2：MIC 声道配置。 |
| 15:8 | Reserved | 8 | R/W | 保留 |
| 7:0 | Reserved | 8 | R/W | 保留 |

### 20.2.4 声道格式说明

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7 | Reserved | 1 | R/W | 保留 |
| 6 | DMA_EN | 1 | R/W | DMA 使能<br>1：DMA 打开<br>0：DMA 关闭 |
| 5:4 | FIFO_THRES | 2 | R/W | FIFO 门限<br>5：4 输出通道 输入通道<br>00 FIFO 1/4 空 FIFO 1/4 满<br>01 FIFO 1/2 空 FIFO 1/2 满<br>10 FIFO 3/4 空 FIFO 3/4 满<br>11 FIFO 全空 FIFO 全满 |
| 3:2 | SW | 2 | R/W | 采样位数<br>00：8 位<br>10：16 位 |
| 1 | VSR | 1 | R/W | 采样率<br>1：采样率可变<br>0：采样率固定（48KHz） |
| 0 | CH_EN | 1 | R/W | 通道使能<br>1：通道打开<br>0：通道关闭（或者进入节能状态） |

### 20.2.5 Codec 寄存器访问命令

中文名： Codec 寄存器访问命令
寄存器位宽： [31：0]
偏移量： 0x18
复位值： 0x00000000

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| 31 | CODEC_WR | 1 | R/W | 读/写选择<br>1：读，读取数据时，先设置 CODEC_WR 为读方式，并在 CODEC_ADR 设置欲访问的寄存器地址；等到返回数据完成中断时再读 CODEC_DAT 寄存器读取值。<br>0：写 |
| 30:23 | Reserved | 8 | R | 保留 |
| 22:16 | CODEC_ADR | 7 | R/W | Codec 寄存器地址 |
| 15:0 | CODEC_DAT | 16 | R/W | Codec 寄存器数据 |

### 20.2.6 中断状态寄存器/中断掩膜寄存器

中文名：　　中断状态/中断掩膜寄存器
寄存器位宽：　[31：0]
偏移量：　　0x54/58
复位值：　　0x00000000

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 31 | IC_FULL | 1 | R/W | 输入通道 2：FIFO 满 |
| 30 | IC_TH_INT | 1 | R/W | 输入通道 2：FIFO 达到门限 |
| 29:8 | Reserved | 22 | R/W | 保留 |
| 7 | OC1_FULL | 1 | R/W | 输出通道 1：FIFO 满 |
| 6 | OC1_EMPTY | 1 | R/W | 输出通道 1：FIFO 空 |
| 5 | OC1_ TH_INT | 1 | R/W | 输出通道 1：FIFO 达到门限 |
| 4 | OC0_FULL | 1 | R/W | 输出通道 0：FIFO 满 |
| 3 | OC0_EMPTY | 1 | R/W | 输出通道 0：FIFO 空 |
| 2 | OC0_ TH_INT | 1 | R/W | 输出通道 0：FIFO 达到门限 |
| 1 | CW_DONE | 1 | R/W | Codec 寄存器写完成 |
| 0 | CR_DONE | 1 | R/W | Codec 寄存器读完成 |

### 20.2.7 中断状态/清除寄存器

中文名：　　中断状态/清除寄存器
寄存器位宽：　[31：0]
偏移量：　　0x5c
复位值：　　0x00000000

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 31:0 | INT_CLR | 32 | RO | 屏蔽后的中断状态寄存器，对本寄存器的读操作将清除寄存器 0x54 中的所有中断状态 |

### 20.2.8 OC 中断清除寄存器

中文名：　　OC 中断清除
寄存器位宽：　[31：0]
偏移量：　　0x60
复位值：　　0x00000000

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 31:0 | INT_OC_CLR | 32 | RO | 对本寄存器的读操作将清除寄存器 0x54 中的所有 output channel 的中断状态对应的 bit[7:2] |

### 20.2.9 IC 中断清除寄存器

中文名： IC 中断清除
寄存器位宽： [31：0]
偏移量： 0x64
复位值： 0x00000000

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 31:0 | INT_IC_CLR | 32 | RO | 对本寄存器的读操作将清除寄存器 0x54 中的所有 input channel 的中断状态对应的 bit[31:30] |

### 20.2.10 CODEC WRITE 中断清除寄存器

中文名： CODEC WRITE 中断清除
寄存器位宽： [31：0]
偏移量： 0x68
复位值： 0x00000000

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 31:0 | INT_CW_CLR | 32 | RO | 对本寄存器的读操作将清除寄存器 0x54 中的中 bit[1] |

### 20.2.11 CODEC READ 中断清除寄存器

中文名： CODEC READ 中断清除
寄存器位宽： [31：0]
偏移量： 0x6c
复位值： 0x00000000

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 31:0 | INT_CR_CLR | 32 | RO | 对本寄存器的读操作将清除寄存器 0x54 中的中 bit[0] |

# 21 PS/2

## 21.1 PS/2 控制器结构

PS/2 控制器的结构如下图所示。

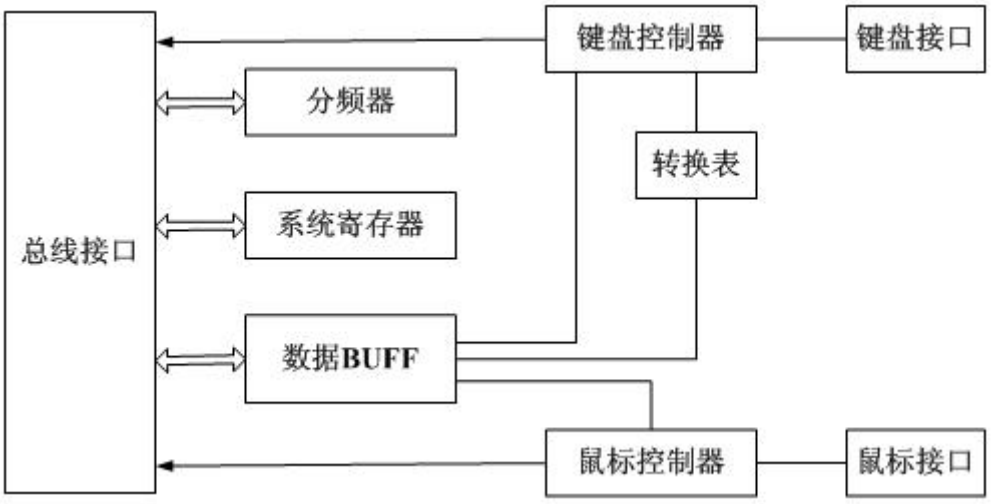



图 21-1 PS/2 控制器结构

PS/2 键盘和鼠标履行一种双向同步串行协议，换句话说每次数据线上发送一位数据，并且每次在时钟线上发一个脉冲，数据就被读入。键盘/鼠标可以发送数据到主机，而主机也可以发送数据到设备，但主机总是在总线上有优先权，它可以在任何时候抑制来自于键盘/鼠标的通讯，只要把时钟拉低即可。基地址是 0x1fe60000。

### 21.1.1 接口寄存器描述

在我们所开发的 PS/2 控制器中共包含如下 4 个寄存器:

| 存器名称 | 寄存器宽度 | 寄存器访问类型 | 寄存器功能描述 |
|---|---|---|---|
| 接收寄存器 | 8 bit | RO | 用于存放从键盘，或鼠标接收到的数据 |
| 发送寄存器 | 8 bit | WO | 用于存放要发送给键盘，鼠标的数据 |
| 状态寄存器 | 8 bit | RO | 用于存放 8 位状态信息 |
| 命令寄存器 | 8 bit | R/W | 用于存放 6 位控制信息 |

### 21.1.2 状态寄存器的位描述:

| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|
| PERR | TO | MOBF | INH | A2 | SYS | IBF | OBF |

各位含义解释如下:

PERR：校验错误标志位。

0：从设备端接收到的数据奇校验正确

1：从设备端接收到的数据奇校验出错，或是收到的设备对控制器所发送的命令响应不正确（控制器向设备发送的命令是”RESEND”,但设备回应的也是”RESEND”）。

TO：超时错误标志位。

0：无超时错误。

1：在设备与控制器的通讯过程中出现了超时错误。

MOBF：鼠标输出缓冲区满。

0：输出寄存器当前为空。

1：输出寄存器当前为满,存在自鼠标接收到的数据可供读取。

INH：设备通信禁止标志。

0：禁止与设备的通信。

1：使能与设备的通信。

A2：地址线,控制器内部使用。

0：上次对控制器的写操作写入的是输入寄存器。

1：上次对控制器的写操作写入的是命令寄存器。

SYS：系统标志Post 读取这个标记以判断当前是上电复位还是软件复位。

0： 当前处于上电自检过程。

1： 系统已经完成了上电自检过程。

IBF：输入寄存器满标志。

0： 输入寄存器空，此时可以向控制器的输入寄存器写入数据。

1： 输入寄存器满，此时不允许向控制器的输入寄存器写入数据。

OBF：输出寄存器满标志。

0： 输出寄存器当前为空。

1： 输出寄存器当前为满，存在数据数据可供读取。

### 21.1.3 命令寄存器的位描述

| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|
| Reserved | XLAT | _EN2 | _EN | Reserved | SYS | INT2 | INT |

各位含义解释如下：(其中 Bit7 和 Bit3 保留不用，故忽略掉)

XLAT：控制器在收到来自键盘的扫描码数据时是否执行扫描码集 2 到扫描码集 1 的映射转换。

0：不进行扫描码转换。

1：进行扫描码转换。

_EN2：是否允许控制器与鼠标的接口(注意与状态寄存器的 INH 位相对比!)。

0：使能与鼠标的接口。

1：禁止与鼠标的接口。

_EN：是否允许控制器与键盘的接口(注意与状态寄存器的 INH 位相对比!).

0：使能与键盘的接口。

1：禁止与键盘的接口。

SYS：用于同步设置状态寄存器的 SYS 位。

0：设置自检程序执行冷启动检测。

1：设置自检程序执行热启动检测。

INT2：是否允许在接收到来自鼠标的数据时产生鼠标中断信号。

0：禁止产生鼠标中断信号。

1：使能产生鼠标中断信号。

INT：是否允许在接收到来自键盘的数据时产生键盘中断信号。

0：禁止产生键盘中断信号。

1：使能产生键盘中断信号。

### 21.1.4 控制器命令描述

通过访问 PS2 控制器的接口寄存器来实现命令控制,不外乎是三类操作：发命令给键盘，发命令给鼠标，发命令给控制器自身。

在软件中应当怎样实现这三类功能呢？下面给出实现流程的描述

1. 发送命令给键盘。
   通过直接将一个 Byte 写到输入寄存器，就可以实现向键盘发送指定命令的功能。
2. 发送命令给鼠标。
   发命令给鼠标要特殊一些，首先要向命令寄存器写入一个控制命令(D4H)，接着再向输入寄存器写入一个命令，这个命令就会被发送给鼠标。
3. 发送命令给控制器自身。
   直接向命令寄存器写入命令即可。如果该命令需要参数的话，再接着向输入寄存器中写入参数，如果该命令对应有返回值的话，返回值会放在输出寄存器中。

### 21.1.5 命令列表

#### 21.1.5.1 发送到键盘的命令列表

0xFF (Reset): 引起键盘进入Reset 模式。

0xF6 (Set Default)： 载入缺省的机打速率/延时（10.9cps/500ms） 按键类型(所有按键都使能机打/通码/断码) 以及使用第二套扫描码集。

0xF5 (Disable)： 键盘停止扫描,载入缺省值等待进一步指令。

0xF4 (Enable)： 使能键盘。

0xF3 (Set Typematic Rate/Delay)： 主机在这条命令后会发送一个字节的参数来定义机打速率和延时0xF2 (Read ID) 读取键盘设备ID。

0xF0 (Set Scan Code Set) 主机在这个命令后发送一个字节的参数以决定键盘使用哪套扫描码集, 参数字节可以是0x01，0x02或0x03，分别对应选择扫描码集第一套,第二套或第三套。如果要获得键盘当前正在使用的扫描码集只要发送带0x00参数的本命令即可。

0xEE (Echo)：回声命令, 键盘用0xEE回应。

0xED (Set/Reset LEDs)：主机在本命令后跟随一个参数字节用于设置键盘上
Num Lock, Caps Lock, and Scroll Lock LED 的状态。

**（注：除了”ECHO”命令以外，发送给键盘的所有命令都应当获得键盘的回应消息（FAH））**

#### 21.1.5.2 发送到鼠标的命令列表

0xFF(Reset)：复位鼠标命令。

0xF6(Set Defaults)：设置鼠标的默认工作模式。

0xF5(Disable Data Reporting)：禁止鼠标的数据报告功能并复位它的位移计数器。

0xF4(Enable Data Reporting)： 使能鼠标的数据报告功能并复位它的位移计数器这条命令只对Stream 模式下的数据报告有效。

0xF3(Set Sample Rate)： 设置鼠标采样率。鼠标用0xFA 回应,然后从主机读入一个或更多字节作为新的采样速率。在收到采样速率后鼠标再次用应答0xFA 回应并复位它的位移计数器。有效的采样速率是10, 20, 40,60, 80, 100和200采样点/秒。

0xF2(Get Device ID)： 读取鼠标的设备ID。

0xF0(Set Remote Mode)： 设置鼠标进入Remote模式。

0xEE(Set Wrap Mode)： 设置鼠标进入Wrap模式。

0xEC(Reset Wrap Mode)：重设鼠标的工作模式为进入Wrap模式之前的模式。

0xEB(Read Data)： 读取鼠标采样到的位移数据包。

0xEA(Set Stream Mode)：设置鼠标的工作模式为Stream 模式。

#### 21.1.5.3 发送到控制器的命令列表

0x20： 读命令寄存器。

0x60： 写命令寄存器。将紧随该命令发送的参数(通过写入输入寄存器实现)写到命令寄存器中。

0xA7： 禁止鼠标接口。

0xA8： 使能鼠标接口。

0xA9： 鼠标接口测试若通过则返回0x00。

0xAA： 控制器自检若成功则返回0x55。

0xAB： 键盘接口测试若通过则返回0x00。

0xAD： 禁止键盘接口。设置命令字节的_EN位并禁止所有控制器与键盘的通讯。

0xAE: 使能键盘接口。清除命令字节的_EN位并重新使能与键盘的通讯。

0xD2： 写键盘缓冲区命令。把紧随该命令的参数写到输出缓冲区就像是从键盘接收到的一样。

0xD3： 写鼠标缓冲区命令。把紧随该命令的参数写到输出缓冲区就像是从鼠标接收到的一样。

0xD4： 写鼠标设备命令。把紧随该命令的参数发给鼠标。

# 22 I2C

## 22.1 概述

本章给出 I2C 的详细描述和配置使用。本系统芯片集成了 I2C 接口，主要用于实现两个器件之间数据的交换。I2C 总线是由数据线 SDA 和时钟 SCL 构成的串行总线，可发送和接收数据。器件与器件之间进行双向传送，最高传送速率 400kbps。龙芯 1A 共集成 3 路 I2C 接口，其中第二路和第三路分别通过 CAN0 和 CAN1 复用实现。复用配置参加 24 章 MUX 寄存器小节。

## 22.2 I²C 控制器结构

I2C 主控制器的结构，主要模块有，时钟发生器（Clock Generator）、字节命令控制器（Byte Command Controller）、位命令控制器（Bit Command controller）、数据移位寄存器（Data Shift Register）。其余为 LPB 总线接口和一些寄存器。

时钟发生器模块：产生分频时钟，同步位命令的工作。

字节命令控制器模块：将一个命令解释为按字节操作的时序，即把字节操作分解为位操作。

位命令控制器模块：进行实际数据的传输，以及位命令信号产生。

数据移位寄存器模块：串行数据移位。

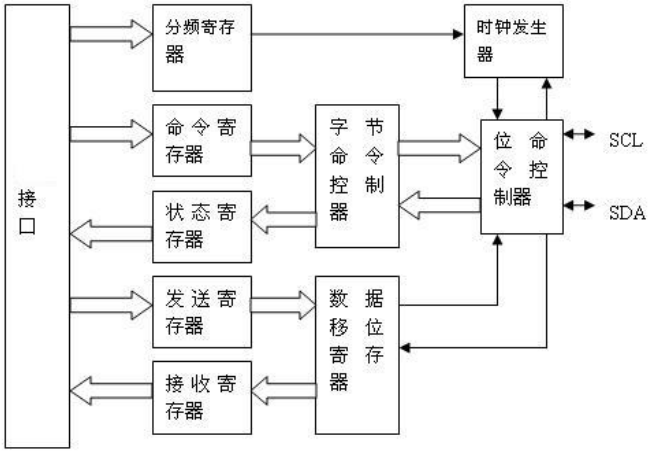



图 21-1 I2C主控制器结构

## 22.3 I²C 控制器寄存器说明

I2C-0 模块寄存器物理地址基址为：0x1fe58000,地址空间 16KB。

I2C-1 模块寄存器物理地址基址为：0x1fe68000,地址空间 16KB。

I2C-2 模块寄存器物理地址基址为：0x1fe70000,地址空间 16KB。

### 22.3.1 分频锁存器低字节寄存器（PRERlo）

中文名：　　分频锁存器低字节寄存器
寄存器位宽：　[7：0]
偏移量：　　0x00
复位值：　　0xff

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7:0 | PRERlo | 8 | RW | 存放分频锁存器的低 8 位 |

### 22.3.2 分频锁存器高字节寄存器（PRERhi）

中文名：　　分频锁存器高字节寄存器
寄存器位宽：　[7：0]
偏移量：　　0x01
复位值：　　0xff

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7:0 | PRERhi | 8 | RW | 存放分频锁存器的高 8 位 |

模块中被分频时钟 clock_a 的频率是 DDR_clk 频率的的一半（DDR_clk 配置见 29 章）；假设分频锁存器的值为 prescale，SCL 总线的输出频率为 clock_s（该时钟根据用户需要和外部 I2C 设备特性确定），则应满足如下关系：

Prcescale　=　clock_a/(5*clock_s)-1

或者 Prcescale　=　DDR_clk/(10*clock_s)-1

### 22.3.3 控制寄存器（CTR）

中文名：　　控制寄存器
寄存器位宽：　[7：0]
偏移量：　　0x02
复位值：　　0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7 | EN | 1 | RW | 模块工作使能位 为 1 正常工作模式, 0 对分频寄存器进行操作 |
| 6 | IEN | 1 | RW | 中断使能位 为 1 则打开中断 |
| 5:0 | Reserved | 6 | RW | 保留 |

### 22.3.4 发送数据寄存器（TXR）

中文名：　　发送寄存器
寄存器位宽：　[7：0]
偏移量：　　0x03
复位值：　　0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7:1 | DATA | 7 | W | 存放下个将要发送的字节 |
| 0 | DRW | 1 | W | 当数据传送时，该位保存的是数据的最低位；<br>当地址传送时，该位指示读写状态 |

### 22.3.5 接受数据寄存器（RXR）

中文名：　　接收寄存器
寄存器位宽：　[7：0]
偏移量：　　0x03
复位值：　　0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7:0 | RXR | 8 | R | 存放最后一个接收到的字节 |

### 22.3.6 命令控制寄存器（CR）

中文名：　　命令寄存器
寄存器位宽：　[7：0]
偏移量：　　0x04
复位值：　　0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7 | STA | 1 | W | 产生 START 信号 |
| 6 | STO | 1 | W | 产生 STOP 信号 |
| 5 | RD | 1 | W | 产生读信号 |
| 4 | WR | 1 | W | 产生写信号 |
| 3 | ACK | 1 | W | 产生应答信号 |
| 2:1 | Reserved | 2 | W | 保留 |
| 0 | IACK | 1 | W | 产生中断应答信号 |

都是在 I2C 发送数据后硬件自动清零。对这些位读操作时候总是读回‘0’。

### 22.3.7 状态寄存器（SR）

中文名：　　状态寄存器
寄存器位宽：　[7：0]
偏移量：　　0x04
复位值：　　0x00

| 位域 | 位域名称 | 位宽 | 访问 | 描述 |
|---|---|---|---|---|
| 7 | RxACK | 1 | R | 收到应答位<br>1 没收到应答位<br>0 收到应答位 |
| 6 | Busy | 1 | R | $I^2c$ 总线忙标志位<br>1 总线在忙<br>0 总线空闲 |
| 5 | AL | 1 | R | 当 $I^2C$ 核失去 $I^2C$ 总线控制权时，该位置 1 |
| 4:2 | Reserved | 3 | R | 保留 |
| 1 | TIP | 1 | R | 指示传输的过程 |

| | | | | |
|---|---|---|---|---|
| | | | | 1 表示正在传输数据<br>0 表示数据传输完毕 |
| 0 | IF | 1 | R | 中断标志位，一个数据传输完，或另外一个器件发起数据传输，该位置 1 |

# 23 PWM

## 23.1 概述

龙芯 1A 里实现了四路脉冲宽度调节/计数控制器，以下简称 PWM．每一路 PWM 工作和控制方式完全相同．每路 PWM 有一路脉冲宽度输出信号 (pwm_o)．系统时钟高达 100MHz，计数寄存器和参考寄存器均 24 位数据宽度，使得芯片非常适合高档电机的控制。

四路 PWM 控制器系统的基地址具体如下：

表 23-1四路控制器描述

| 名称 | 基地址（Base） | 中断号 |
|---|---|---|
| PWM0 | 0x1fe5:C000 | 18 |
| PWM1 | 0x1fe5:C010 | 19 |
| PWM2 | 0x1fe5:C020 | 20 |
| PWM3 | 0x1fe5:C030 | 21 |

每路控制器共有四个控制寄存器，具体描述如下：

表 23-2 控制寄存器描述

| 名称 | 地址 | 宽度 | 访问 | 说明 |
|---|---|---|---|---|
| CNTR | Base + 0x0 | 24 | R/W | 主计数器 |
| HRC | Base + 0x4 | 24 | R/W | 高脉冲定时参考寄存器 |
| LRC | Base + 0x8 | 24 | R/W | 低脉冲定时参考寄存器 |
| CTRL | Base + 0xC | 8 | R/W | 控制寄存器 |

## 23.2 PWM 寄存器说明

实现脉冲宽度功能

CNTR 寄存器可以由系统编程写入获得初始值，系统编程写入完毕后，CNTR 寄存器在系统时钟驱动下不断自加，当到达 LRC 寄存器的值后清 0．然后重新开始不断自加，控制器就产生连续不断的脉冲宽度输出．

表 23-3 主计数器设置

| 位域 | 访问 | 复位值 | 说明 |
|---|---|---|---|
| 23：0 | R/W | 0x0 | 主计数器 |

HRC 寄存器由系统写入，当 CNTR 寄存器的值等于 HRC 的值的时候，控制器产生高脉冲电平．

表 23-4 高脉冲计数器设置

| 位域 | 访问 | 复位值 | 说明 |
|---|---|---|---|
| 23：0 | R/W | 0x0 | 高脉冲计数器 |

LRC 寄存器由系统写入，当 CNTR 寄存器的值等于ＬRC 的值的时候，控制器产生低脉冲电平．

表 23-5 低脉冲计数器设置

| 位域 | 访问 | 复位值 | 说明 |
|---|---|---|---|
| 23：0 | R/W | 0x0 | 低脉冲计数器 |

例：如果要产生宽度为系统始终周期 50 倍的高脉宽和 90 倍的低脉宽，在 HRC 中应该配置初始值(90-1)=89，在 LRC 寄存器中配置初始值(50+90-1)=139.

当工作在定时器模式下，CNTR 记录内部系统时钟．HRC 和 LRC 寄存器的初始值系统编程写入，当 CNTR 寄存器的值等于 HRC 或者 LRC 的时候，芯片会产生一个中断，这样就实现了定时器功能.

CTRL 控制寄存器，在上面三种工作模式下，控制寄存器的功能不变，根据功能需求选择不同的配置.

表 23-6 控制寄存器设置

| 位域 | 访问 | 复位值 | 说明 |
|---|---|---|---|
| 0 | R/W | 0 | EN，主计数器使能位<br>置 1 时：CNTR 用来计数<br>置 0 时：CNTR 停止计数 |
| 2：1 | Reserved | 2’b0 | 预留 |
| 3 | R/W | 0 | OE,脉冲输出使能控制位,低有效<br>置 0 时：脉冲输出使能<br>置 1 时：脉冲输出屏蔽 |
| 4 | R/W | 0 | SINGLE，单脉冲控制位<br>置 1 时：脉冲仅产生一次<br>置 0 时：脉冲持续产生 |
| 5 | R/W | 0 | INTE，中断使能位<br>置 1 时：当 CNTR 计数到 LRC 和 CNTR 后送中断<br>置 0 时：不产生中断 |
| 6 | R/W | 0 | INT，中断位<br>读操作：1 表示有中断产生,0 表示没有中断<br>写入 1：清中断 |
| 7 | R/W | 0 | CNTR_RST,使得 CNTR 计数器清零<br>置 1 时：CNTR 计数器清零<br>置 0 时：CNTR 计数器正常工作 |

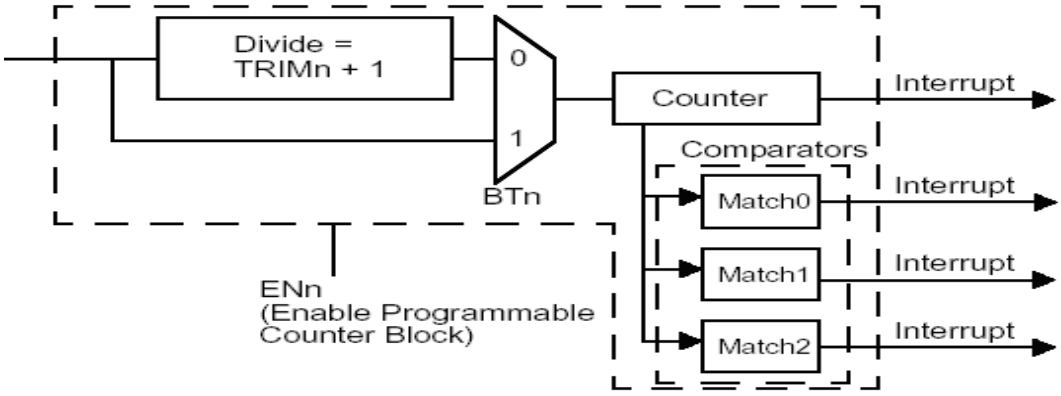
# 24 RTC

## 24.1 概述

龙芯 1A 实时时钟（RTC）单元可以在主板上电后进行配置，当主板断电后，该单元任然运作，可以仅靠板上的电池供电就正常运行。RTC 单元运行时功耗仅几个微瓦。

RTC 由外部 32.768KHZ 晶振驱动，内部经可配置的分频器分频后，该时钟用来计数，年月日，时分秒等信息被更新。同时该时钟也用于产生各种定时和计数中断。

RTC 单元由计数器和定时器组成，其构架如下图所示：




## 24.2 RTC 电源配置

龙芯1A（的电源管理单元ACPI控制着 ）RTC如果，单元的读写通道ACPI中未对RTC则无法对，使能RTC要使用，所以 。写操作单元进行读RTC模块首先需要在硬件上对ACPI然后才能在软件层使用，部分进行正确配置RTC详细配置参考 。模块ACPI
。手册部分说明

## 24.3 寄存器描述

RTC 模块寄存器位于 0x1fe64000——0x1fe67fff 的 16KB 地址空间内，其基地址为 0x1fe64000，所有寄存器位宽均为 32 位。

### 24.3.1 寄存器地址列表

| 名称 | 地址 | 位宽 | RW | 描述 | 复位值 |
|---|---|---|---|---|---|

| | | | | | |
|---|---|---|---|---|---|
| sys_toytrim | 0x1fe64020 | 32 | RW | 对 32.768kHz 的分频系数（计数器时钟） | |
| sys_toywrite0 | 0x1fe64024 | 32 | W | TOY 低 32 位数值写入 | |
| sys_toywrite1 | 0x1fe64028 | 32 | W | TOY 高 32 位数值写入 | |
| sys_toyread0 | 0x1fe6402C | 32 | R | TOY 低 32 位数值读出 | |
| sys_toyread1 | 0x1fe64030 | 32 | R | TOY 高 32 位数值读出 | |
| sys_toymatch0 | 0x1fe64034 | 32 | RW | TOY 定时中断 0 | |
| sys_toymatch1 | 0x1fe64038 | 32 | RW | TOY 定时中断 1 | |
| sys_toymatch2 | 0x1fe6403C | 32 | RW | TOY 定时中断 2 | |
| sys_rtcctrl | 0x1fe64040 | 32 | RW | TOY 和 RTC 控制寄存器 | |
| sys_rtctrim | 0x1fe64060 | 32 | RW | 对 32.768kHz 的分频系数(定时器时钟) | |
| sys_rtcwrite0 | 0x1fe64064 | 32 | R | RTC 定时计数写入 | |
| sys_rtcread0 | 0x1fe64068 | 32 | W | RTC 定时计数读出 | |
| sys_rtcmatch0 | 0x1fe6406C | 32 | RW | RTC 时钟定时中断 0 | |
| sys_rtcmatch1 | 0x1fe64070 | 32 | RW | RTC 时钟定时中断 1 | |
| sys_rtcmatch2 | 0x1fe64074 | 32 | RW | RTC 时钟定时中断 2 | |

注意：其中 sys_toytrim 及 sys_rtctrim 寄存器复位后，其值不确定，如果不需要对外部晶振进行分频，请对这两个寄存器清零，这样 RTC 模块才能正常计时工作。

### 24.3.2 SYS_TOYWRITE0

中文名： TOY 计数器低 32 位数值
寄存器位宽： [31：0]
偏移量： 0x20
复位值： 0x00000000

| 位域 | 位域名称 | 访问 | 缺省 | 描述 |
|---|---|---|---|---|
| 31:26 | TOY_MONTH | W | | 月，范围 1~12 |
| 25:21 | TOY_DAY | W | | 日，范围 1~31 |
| 20:16 | TOY_HOUR | W | | 小时，范围 0~23 |
| 15:10 | TOY_MIN | W | | 分，范围 0~59 |
| 9:4 | TOY_SEC | W | | 秒，范围 0~59 |
| 3:0 | TOY_MILLISEC | W | | 0.1 秒，范围 0~9 |

### 24.3.3 SYS_TOYWRITE1

中文名： TOY 计数器高 32 位数值
寄存器位宽： [31：0]
偏移量： 0x24
复位值： 0x00000000

| 位域 | 位域名称 | 访问 | 缺省 | 描述 |
|---|---|---|---|---|
| 31:0 | TOY_YEAR | W | | 年，范围 0～16383 |

### 24.3.4 SYS_TOYMATCH0/1/2

中文名： TOY 计数器中断寄存器 0/1/2
寄存器位宽： [31：0]
偏移量： 0x34/38/3C
复位值： 0x00000000

| 位域 | 位域名称 | 访问 | 缺省 | 描述 |
|---|---|---|---|---|
| 31:26 | YEAR | RW | | 年，范围 0～16383 |
| 25:22 | MONTH | RW | | 月，范围 1~12 |
| 21:17 | DAY | RW | | 日，范围 1~31 |
| 16:12 | HOUR | RW | | 小时，范围 0~23 |
| 11:6 | MIN | RW | | 分，范围 0~59 |
| 5:0 | SEC | RW | | 秒，范围 0~59 |

### 24.3.5 SYS_RTCCTRL

中文名： RTC 定时器中断寄存器 0/1/2
寄存器位宽： [31：0]
偏移量： 0x40
复位值： 0x00000000

| 位域 | 位域名称 | 访问 | 缺省 | 描述 |
|---|---|---|---|---|
| 31:24 | 保留 | R | 0 | 保留，置 0 |
| 23 | ERS | R | 0 | REN(bit13)写状态 |
| 22:21 | 保留 | R | 0 | 保留，置 0 |
| 20 | RTS | R | 0 | Sys_rtctrim 写状态 |
| 19 | RM2 | R | 0 | Sys_rtcmatch2 写状态 |
| 18 | RM2 | R | 0 | Sys_rtcmatch2 写状态 |
| 17 | RM0 | R | 0 | Sys_rtcmatch0 写状态 |
| 16 | RS | R | 0 | Sys_rtcwrite 写状态 |
| 15 | 保留 | R | 0 | 保留，置 0 |
| 14 | BP | R/W | 0 | 旁路 32.768k 晶振<br>0:选择晶振输入；<br>1: GPIO8 来驱动计数器，这是测试模式，GPIO8 通过外部时钟或者 GPIO8 控制器。 |
| 13 | REN | R/W | 0 | 0：RTC 禁止；1：RTC 使能 |
| 12 | BRT | R/W | 0 | 旁路 RTC 分频<br>0:正常操作；<br>1:RTC 直接被 32.768k 晶振驱动 |
| 11 | TEN | R/W | 0 | 0：TOY 禁止；1：TOY 使能 |
| 10 | BTT | R/W | 0 | 旁路 TOY 分频<br>0:正常操作；<br>1:TOY 直接被 32.768k 晶振驱动 |
| 9 | 保留 | R | 0 | 保留，置 0 |
| 8 | EO | R/W | 0 | 0: 32.768k 晶振禁止； |

| | | | | |
|---|---|---|---|---|
| | | | | 1：32.768k 晶振使能 |
| 7 | ETS | R | 0 | TOY 使能写状态 |
| 6 | 保留 | R | 0 | 保留，置 0 |
| 5 | 32S | R | 0 | 0：32.768k 晶振不工作；<br>1：32.768k 晶振正常工作。 |
| 4 | TTS | R | 0 | Sys_toytrim 写状态 |
| 3 | TM2 | R | 0 | Sys_toymatch2 写状态 |
| 2 | TM1 | R | 0 | Sys_toymatch1 写状态 |
| 1 | TM0 | R | 0 | Sys_toymatch0 写状态 |
| 0 | TS | R | 0 | Sys_toywrite 写状态 |

### 24.3.6 SYS_RTCMATCH0/1/2

中文名： RTC 定时器中断寄存器 0/1/2
寄存器位宽： [31：0]
偏移量： 0x6C/70/74
复位值： 0x00000000

| 位域 | 位域名称 | 访问 | 缺省 | 描述 |
|---|---|---|---|---|
| 31:26 | YEAR | RW | | 年，范围 0～16383 |
| 25:22 | MONTH | RW | | 月，范围 1~12 |
| 21:17 | DAY | RW | | 日，范围 1~31 |
| 16:12 | HOUR | RW | | 小时，范围 0~23 |
| 11:6 | MIN | RW | | 分，范围 0~59 |
| 5:0 | SEC | RW | | 秒，范围 0~59 |

# 25 NAND

## 25.1 NAND 控制器结构描述

NAND FLASH 控制器最多支持 4 个片选和 4 个 RDY 信号，控制器支持 SLC 和 MLC 两种类型 FLASH 的操作，NAND FLASH 控制器支持系统启动，启动模式包括 ECC 模式启动和普通模式启动。

系统启动模式选择包括两种，如下表所示：

| 启动模式 | 配置 | 说明 |
|---|---|---|
| ECC 模式启动 | NAND_CLE 外部上拉<br>LCD_DAT_B0 外部上拉 | 外部 NAND FLASH 中的第一个 page 的内容必须是原始数据经过 RS(204,188)译码后生成的数据 |
| 普通启动 | NAND_CLE 外部上拉<br>LCD_DAT_B0 外部下拉 | 外部 NAND FLASH 第一个 page 的数据普通原始数据,复用 SPI1 和 PWM23 |

## 25.2 NAND 控制器寄存器配置描述

NAND 内部的寄存器的设置如下：

| 地址 | 寄存器名称 |
|---|---|
| 0x1fe7_8000 | NAND_CMD |
| 0x1fe7_8004 | ADDR_C |
| 0x1fe7_8008 | ADDR_R |
| 0x1fe7_800C | NAND_TIMING |
| 0x1fe7_8010 | ID_L |
| 0x1fe7_8014 | STATUS & ID_H |
| 0x1fe7_8018 | NAND_PARAMETER |
| 0x1fe7_801C | NAND_OP_NUM |
| 0x1fe7_8020 | CS_RDY_MAP |
| 0x1fe7_8040 | DMA access address |

### 25.2.1 NAND_CMD（地址：0x1fe7_8000）

| 位 | 位域名 | 读写 | 描述 |
|---|---|---|---|
| 31:26 | | R/- | Reserved |
| 25 | DMA_REQ | R/- | 非 ECC 模式下 NAND 发出 DMA 请求 |
| 24 | ECC_DMA_REQ | R/- | ECC 模式下 NAND 发出 DMA 请求 |
| 23：20 | NAND_CE | R/- | 外部 NAND 芯片片选情况 |
| 19：16 | NAND_RDY | R/- | 外部 NAND 芯片 RDY 情况 |
| 15：14 | | | Reserved |
| 13 | INT_EN | R/W | NAND 中断使能信号 |
| 12 | RS_WR | R/W | 写操作时候 ECC 功能开启 |
| 11 | RS_RD | R/W | 读操作时候 ECC 功能开启 |
| 10 | done | R/W | 操作完成 |
| 9 | Spare | R/W | 操作发生在 NAND 的 SPARE 区 |
| 8 | Main | R/W | 操作发生在 NAND 的 MAIN 区 |

| | | | |
|---|---|---|---|
| 7 | Read status | R/W | 读 NAND 的状态 |
| 6 | Reset | R/W | Nand 复位操作 |
| 5 | read id | R/W | 读 ID 操作 |
| 4 | blocks erase | R/W | 连续擦除标志，缺省 0；1 有效，连续擦擦块的数目由 nand_op_num 决定 |
| 3 | erase operation | R/W | 擦除操作 |
| 2 | write operation | R/W | 写操作 |
| 1 | read operation | R/W | 读操作 |
| 0 | command valid | R/W | 命令有效 |

25.2.2 ADDR_C（地址：0x1fe7_8004）

| 位 | 位域名 | 读写 | 描述 |
|---|---|---|---|
| 31:12 | | R/- | Reserved |
| 11:0 | Nand_address[11:0] | R/W | 读、写、擦除操作起始地址页内地址 |

25.2.3 ADDR_R（地址：0x1fe7_8008）

| 位 | 位域名 | 读写 | 描述 |
|---|---|---|---|
| 31:25 | | R/- | Reserved |
| 24:0 | Nand_address[33:12] | R/W | 读、写、擦除操作起始地址页地址 |

25.2.4 NAND_TIMING（地址：0x1fe7_800C）

| 位 | 位域名 | 读写 | 描述 |
|---|---|---|---|
| 31:16 | | R/- | Reserved |
| 15：8 | Hold cycle | R/W | NAND 命令有效需等待的周期数，缺省 4 |
| 7：0 | Wait cycle | R/W | NAND 一次读写所需总时钟周期数，缺省 18,ECC 模式下配置为 8’hb |

25.2.5 ID_L（地址：0x1fe7_8010）

| 位 | 位域名 | 读写 | 描述 |
|---|---|---|---|
| 31：0 | ID[31:0] | R/- | ID[31:0] |

25.2.6 STATUS & ID_H（地址：0x1fe7_8014）

| 位 | 位域名称 | 访问 | |
|---|---|---|---|
| 31:16 | | R/- | Reserved |
| 15:8 | STATUS | R/- | NAND 设备当前的读写完成状态 |
| 7:0 | ID[40:32] | R/- | ID 高 8 位 |

25.2.7 NAND_PARAMETER（地址：0x1fe7_8018）

| 位 | 位域名 | 读写 | 描述 |
|---|---|---|---|
| 31:12 | | R/- | Reserved |
| 11:8 | 外部颗粒容量大小 | R/W | 1：1Gb<br>2：2Gb<br>3：4Gb<br>4：8Gb<br>5：16Gb<br>6：32Gb<br>7：64Gb<br>8：128Gb |
| 7:0 | | R/- | Reserved |

25.2.8 NAND_OP_NUM（地址：0x1fe7_801C）

| 位 | 位域名 | 读写 | 描述 |
|---|---|---|---|
| 31：0 | NAND_OP_NUM | R/W | NAND 读写操作 Byte 数；擦除为块数 |

25.2.9 CS_RDY_MAP（地址：0x1fe7_8020）

NAND 的 4 个 CS 由所访问的地址硬件自动生成，CS0/RDY0 对应最低一块空间，CS1/RDY1 对应次低一块空间，其它以此类推。如果需要调整外部芯片和 NAND 地址的关系，可通过设置本寄存器，对 cs_rdy 1/2/3 进行重新映射。

| 位 | 位域名 | 读写 | 描述 |
|---|---|---|---|
| 31：28 | rdy3_sel | R/W | rdy3 信号从芯片引脚到 NAND 控制器的映射<br>4’b0001:NAND_RDY[0]<br>4’b0010:NAND_RDY[1]<br>4’b0100:NAND_RDY[2]<br>4’b1000:NAND_RDY[3] |
| 27：24 | cs3_sel | R/W | cs3 信号从 NAND 控制器到芯片引脚的映射<br>4’b0001:NAND_CS[0]<br>4’b0010:NAND_CS[1]<br>4’b0100:NAND_CS[2]<br>4’b1000:NAND_CS[3] |
| 23：20 | rdy2_sel | R/W | rdy2 信号从芯片引脚到 NAND 控制器的映射<br>4’b0001:NAND_RDY[0]<br>4’b0010:NAND_RDY[1]<br>4’b0100:NAND_RDY[2]<br>4’b1000:NAND_RDY[3] |
| 19：16 | cs2_sel | R/W | cs2 信号从 NAND 控制器到芯片引脚的映射<br>4’b0001:NAND_CS[0]<br>4’b0010:NAND_CS[1]<br>4’b0100:NAND_CS[2]<br>4’b1000:NAND_CS[3] |
| 15：12 | rdy1_sel | R/W | rdy1 信号从芯片引脚到 NAND 控制器的映射<br>4’b0001:NAND_RDY[0]<br>4’b0010:NAND_RDY[1]<br>4’b0100:NAND_RDY[2]<br>4’b1000:NAND_RDY[3] |
| 11：8 | cs1_sel | R/W | cs1 信号从 NAND 控制器到芯片引脚的映射<br>4’b0001:NAND_CS[0]<br>4’b0010:NAND_CS[1]<br>4’b0100:NAND_CS[2]<br>4’b1000:NAND_CS[3] |
| 7:0 | | R/- | Reserved |

25.2.10 DMA_ADDRESS（地址：0x1fe7_8040）

| 位 | 位域名 | 读写 | 描述 |
|---|---|---|---|
| 31：0 | DMA_ADDRESS | R/W | DMA 读写 NAND flash 数据（ID/STATUS 除外）时候的访问地址，读/写地址相同，读写方向通过 DMA 配置实现 |

## 25.3 NAND ADDR 说明

定义:
main_op = NAND_CMD[8];
spare_op = NAND_CMD[9];
addr_in_page ={ A11, A10.. A2, A1, A0}=ADDR_C
page_number ={ …A32,A31,A30,A29,A28…A13,A12}= ADDR_R
NAND 地址空间示例

| | I/O | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|---|
| Column1 | 1st Cycle | A0 | A1 | A2 | A3 | A4 | A5 | A6 | A7 |
| Column2 | 2nd Cycle | A8 | A9 | A10 | **A11** | *L | *L | *L | *L |
| Row1 | 3rd Cycle | A12 | A13 | A14 | A15 | A16 | A17 | A18 | A19 |
| Row2 | 4th Cycle | A20 | A21 | A22 | A23 | A24 | A25 | A26 | A27 |
| Row3 | 5th Cycle | A28 | A29 | A30 | A31 | A32 | A33 | …. | …. |

对系统板上 NAND 颗粒来说，如果仅仅操作 spare 区，A11=1 是唯一标志。所以软件配置内部寄存器时，需要配置 A11 和 spare_op 均为 1(见 Examples5) ，错误的示例见 Examples2。

对系统板上 NAND 颗粒来说，如果仅仅操作 main 区，A11=0 是唯一标志.；所以软件配置内部寄存器时，需要配置 A11 和 spare_op 均为 0（见 Examples1），错误的示例见 Examples4。

对系统板上 NAND 颗粒来说，如果操作 main+spare 区，A11 可以为 0（见 Examples3）；也可以为 1（见 Examples6）。

Examples1:（非 ECC 模式下。NAND 颗粒中一个 page 的数据只能位于 0x0-0x83f，第一个 op 表示读写开始的数据，接下来的 op 表示随后的读写数据；NO_op 表示不能被本次 NAND 配置读写的数据）

(spare_op = 1’b0 & main_op =1’b0) equal to (spare_op = 1’b0 & main_op =1’b1); ADDR_C =0x30

| Data in a page | 0 | 0x30 | …. | 0x7ff | 0x800 | 0x830 | 0x83f |
|---|---|---|---|---|---|---|---|
| Page 0 | NO_op | op | op | op | NO_op | NO_op | NO_op |
| Page 1 | op | op | op | op | NO_op | NO_op | NO_op |
| Page 2 | op | op | op | op | NO_op | NO_op | NO_op |

Examples2:
spare_op=1’b1 & main_op=1’b0; ADDR_C = 0x30 (配置出错!! 开始操作不在 spare 区，下图是可能的错误访问顺序)

| Data in a page | 0 | 0x30 | …. | 0x7ff | 0x800 | 0x830 | 0x83f |
|---|---|---|---|---|---|---|---|
| Page 0 | NO_op | op | op | op | op | op | op |
| Page 1 | NO_op | NO_op | NO_op | NO_op | op | op | op |
| Page 2 | NO_op | NO_op | NO_op | NO_op | op | op | op |
| Page 3 | NO_op | NO_op | NO_op | NO_op | op | op | op |

Examples3:
spare_op = 1’b1 & main_op =1’b1; ADDR_C = 0x30

| Data in a page | 0 | 0x30 | …. | 0x7ff | 0x800 | 0x830 | 0x83f |
|---|---|---|---|---|---|---|---|
| Page 0 | NO_op | op | op | op | op | op | op |
| Page 1 | op | op | op | op | op | op | op |
| Page 2 | op | op | op | op | op | op | op |

Examples4:

(spare_op=1'b0 & main_op=1'b0), (equal to spare_op=1'b0 & main_op=1'b1); ADDR_C =0x830: (配置出错!! 开始操作在 spare 区，下图是可能的错误访问顺序)

| Data in a page | 0 | 0x30 | …. | 0x7ff | 0x800 | 0x830 | 0x83f |
|---|---|---|---|---|---|---|---|
| Page 0 | NO_op | NO_op | NO_op | NO_op | NO_op | NO_op | NO_op |
| Page 1 | NO_op | op | op | op | op | NO_op | NO_op |
| Page 2 | op | op | op | op | op | NO_op | NO_op |
| Page 3 | op | op | op | op | op | NO_op | NO_op |

Examples5:

spare_op = 1'b1 and main_op =1'b0; ADDR_C = 0x830

| Data in a page | 0 | 0x30 | …. | 0x7ff | 0x800 | 0x830 | 0x83f |
|---|---|---|---|---|---|---|---|
| Page 0 | NO_op | NO_op | NO_op | NO_op | NO_op | op | op |
| Page 1 | NO_op | NO_op | NO_op | NO_op | op | op | op |
| Page 2 | NO_op | NO_op | NO_op | NO_op | op | op | op |

Examples6:

spare_op = 1'b1 & main_op =1'b1; ADDR_C = 0x830

| Data in a page | 0 | 0x30 | …. | 0x7ff | 0x800 | 0x830 | 0x83f |
|---|---|---|---|---|---|---|---|
| Page 0 | NO_op | NO_op | NO_op | NO_op | NO_op | op | op |
| Page 1 | op | op | op | op | op | op | op |
| Page 2 | op | op | op | op | op | op | op |
| Page 3 | op | op | op | op | op | op | op |

## 25.4 NAND ECC 说明

硬件集成 ECC 功能，ECC 采用 RS(204,188)方法进行编码和解码，在配置软件过程中需要注意以下几点：

1. 每次读写 NAND 的时候，推荐配置 PAGE 的页内部地址（ADDR_C）为 0；
2. NAND 每个 PAGE 有 2048Bytes，采用 RS(204,188)方式编解码后，只会用到前面的 2040Bytes，会有 8 个 Bytes 不用；采用 ECC 后 NAND 利用率为 188/204；
3. 在配置操作数的时候，如果每次操作一个页面，请配置 NAND 里面的 op_num 为 204 的倍数(byte 为单位)；在配置 DMA 控制器时候，操作数为 47(188/4)的倍数(word 为单位)。
4. ECC 操作和 OOB 操作可以分开，比如对一个页完成 ECC 读/写后可以对其 OOB 进行操作。

可以在 ECC 操作完成后通过普通方式读回所有内容，包括原始数据和 ECC 校验增加的数据（此时配置操作数 op_num 和 DMA 相同）。

校验能力说明：最多可以纠错 8 个 Bytes，这些 Bytes 内部出错的位数可以是 1-8 个。

第一行数据共出错 64bit，恰好是 8 个 Bytes，可以纠错；最后一行数据虽然错 9bits，分散在 9Bytes 中，无法纠错。

| 原始数据（204Bytes） | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | … | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 数据 1 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | ff | ff | ff | ff | … | 可纠错 |
| 数据 2 | 1 | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | … | 可纠错 |
| 数据 3 | fe | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | … | 可纠错 |
| 数据 4 | 1 | 2 | 3 | 4 | 5 | 6 | 07 | 08 | ff | ff | ff | ff | … | 可纠错 |
| 数据 5 | fe | fe | fe | fe | fe | fe | fe | fe | fe | ff | ff | ff | … | 不可纠错 |

# 26 ACPI

## 26.1 概述

a) 支持 ACPI 协议 3.0b，提供功耗管理

ACPI 24 位的计数器；软件发起处理器的降频工作模式；

SCI（System Control Interrupt）中断的产生；

b) PCI PME#为唤醒系统信号。

c) 系统时钟控制

ACPI C2 状态通过停止处理器的时钟（STPCLK#）来停止处理器执行指令；

ACPI C3 状态：停止处理器时钟（时钟生成器不给处理器时钟），但是不停止内存的时钟；

d) 系统睡眠状态控制

ACPI S3 状态：挂起到内存（STR）;

ACPI S4 状态：挂起到磁盘（STD）;

ACPI G2/S5 状态：软挂起（soft off）;

Power Failure Detection and Recovery;

## 26.2 全系统的功耗状态描述（power state）

下表定义了基于龙芯 1A 的系统的功耗状态（power state）。这些状态的名字命名规则与 ACPI 是一样的。其中 G 代表全局状态，S 代表 Sleep, C 代表 cpu。

| 状态/子状态 | 描述 |
|---|---|
| G0/S0/C0 | Full on: 处理器工作状态。为了降低全系统的功耗，各个设备能被单独关闭。Cx 状态定义了处理器不同的工作状态。在 C0 状态下，龙芯 1A 的功耗管理模块通过 STPCLK#信号降低处理器的工作频率（throttling），从而进一步将低功耗。龙芯 1A 只支持软件发起处理器降频操作（没有温度管理）。 |
| G0/S0/C1 | Auto-Halt：处理器执行 WAIT 指令后，处理将不再执行指令（但是还有时钟）;(在需要硬件维护 cache 一致性的情况下，处理器需要侦听总线以保证 cache 内容的正确)。 |
| G0/S0/C2 | Stop-Grant: 送给处理器的 STPCLKn 信号有效（当作为 1A 时为 PAD 输出，当作为 SoC 为送给 CPU 的内部信号）。处理器的时钟被停掉，处理器不再执行指令。处理器维持这个状态直到 STPCLKn 无效。(在需要硬件维护 cache 一致性的情况下，处理器需要侦听总线以保证 cache 内容的正确) |
| G0/S0/C3 | Stop-Clock:送给处理器的 STPCLK#信号有效；处理器进入 STOP-Grant 模式，然后送给板子的 STP-CPU#信号有效，告之时钟生成模块停止给处理器产生时钟。(在需要硬件维护 cache 一致性的情况下，处理器需要侦听总线以保证 cache 内容的正确) |

| | |
|---|---|
| G1/S3 | Suspend-To-RAM(STR):挂起到内存。系统的上下文保存在内存中。除了需要唤醒系统的逻辑（保存在 Resume 域）外，其他所有设备的电源将被关掉（包括内存控制器）。内存本身不断电，内存的内容保持不变，且维持自刷新。除了 RTC 和 Resume 的时钟外，所有的时钟都被停掉。 |
| G1/S4 | Suspend-To-Disk(STD):系统的上下文保存在硬盘中（硬盘本身不断电）。除了需要唤醒系统的逻辑外（RTC 和 Resume），其他所有设备的电源将被关掉（包括内存控制器以及内存本身）。除了 RTC 和 Resume 的时钟外，所有的时钟都被停掉。 |
| G2/S5 | Soft Off: 系统的上下文无需保存，除了需要唤醒系统的逻辑外（RTC 和 Resume）外，所有的电源都被关掉（包括硬盘）。除了 RTC 和 Resume 的时钟外，所有的时钟都被停掉。当被唤醒后，系统需要完整的启动过程。 |
| G3 | Mechanical OFF(MOFF): 系统的上下文无需保存，除了 RTC 外，其他所有的电源都被关掉（包括 Resume），所以没有唤醒事件。当系统的供电被拔出或电池供电不足，以及用户按下关机键时，系统进入这个状态。当系统的供电恢复时，根据 GEN_PMCON3 寄存器的 AFTERG3 位，来决定切换到哪个状态。 |

下表列出了系统在不同状态下的转换规则。

| 当前状态 | 状态转换事件 | 下一个状态 |
|---|---|---|
| G0/S0/C0 | 处理器执行 wait 指令<br>读 Level 2 寄存器<br>读 Level 3 寄存器<br>SLP_EN 被置位<br>Power button override<br>供电失效或机械关机 | G0/S0/C1<br>G0/S0/C2<br>G0/S0/C3<br>G1/Sx 或 G2/S5<br>G2/S5<br>G3 |
| G0/S0/C1 | 任意异常事件<br>STPCLK#有效<br>Power button override<br>供电失效 | G0/S0/C0<br>G0/S0/C2<br>G2/S5<br>G3 |
| G0/S0/C2 | 任意异常事件<br>Power button override<br>供电失效 | G0/S0/C0<br>G2/S5<br>G3 |
| G0/S0/C3 | 任意异常事件<br>Power button override<br>供电失效 | G0/S0/C0<br>G2/S5<br>G3 |
| G1/Sx | 任意被使能的唤醒事件<br>Power button override<br>供电失效 | G0/S0/C0<br>G2/S5<br>G3 |
| G2/S5 | 任意被使能的唤醒事件<br>供电失效 | G0/S0/C0<br>G3 |
| G3 | 供电恢复 | S0/C0（reboot）或 G2/S5 状态 |

## 26.3 龙芯龙芯 1A 的电源域

| 电源域 | 描述 |
|---|---|
| CORE | 由主供电电源（main power supply）供电.当系统处于 S3，S4，S5 或 G3 状态时，这个电源域的供电将被断掉； |
| Resume | 由主供电电源供电。当系统当系统处于 S3，S4，S5 状态时，这个电源域的供电需要维持 |

| RTC | 相对于主供电电源而言，这有电池供电。 |
|---|---|

在龙芯 1A 中，RTC well 的电路包括 RTC 模块以及 ACPI 模块中的一部分；Resume Well 的电路包括那些可以唤醒系统的逻辑以及 ACPI 模块中的一部分；CORE well 的电路包括其他所有不需要唤醒系统的逻辑以及 ACPI 的一部分。

## 26.4 ACPI 控制寄存器

这些寄存器中的每一位都有各自读写特性，用以下符号标示：

RO：只读；

WO: 只写；

R/W: 可读可写；

R/WC: 可读且写 1 清 0. 即当往这位写 1 时，当往这位写 1 时，将这位清 0；写 0 则不会产生效果；

R/WO：可读且仅能写一次。即这为在上电后，仅能被写一次。之后将变成只读属性；

以下寄存器分布在不同的电源域中，也有的寄存器的不同位也会分布在不同的电源域中，而处在不同电源域中的寄存器只能被那个特定电源域的重启信号给复位。比如处于 Resume 域的寄存器位只能被 RSMRSTn 复位， 而处于 RTC 电源域的寄存器只能被 RTCRSTn 复位。

下表列出了南桥功耗管理寄存器的地址映射以及读写特性。

| 偏移地址 | 寄存器 | 描述 | 默认值 | 读写特性 |
|---|---|---|---|---|
| 0x30 | GEN_PMCON_1 | general power management configuration1 | 0000h | RO |
| 0x34 | GEN_PMCON_2 | general power management configuration2 | 00h | R/WC<br>R/W |
| 0x38 | GEN_PMCON_3 | general power management configuration3 | 00h | R/W<br>R/WC |
| 0x00 | PM1_STS | power management1 status | | R/WC |
| 0x04 | PM1_EN | power management1 enbale | | R/W |
| 0x08 | PM1_CNT | power management control | | WO<br>R/W<br>WO |
| 0x0C | PM1_TMR | power management timer | | RO |
| 0x10 | PROC_CNT | processor control | | R/W |
| 0x14 | LVL2 | level2 register | | RO |
| 0x18 | LVL3 | level3 register | | RO |
| 0x20 | GPE0_STS | general purpose event0 status | | R/WC |
| 0x24 | GPE0_EN | general purpose event0 enable | | R/W |
| 0x50 | CPU_INIT | CPU initialization | | R/W |
| 0x44 | RST_CNT | Reset Control register | | R/W |

### 26.4.1 GEN_PMCON_1: General PM Configuration1 Register

| 偏移地址 | 读写特性 | 默认值 | 大小 | 用处 | 电源域 |
|---|---|---|---|---|---|
| 0x30 | RO | 0000h | 32bits | ACPI | Core Well |

| 位 | 描述 | 读写特性 | 电源域 |
|---|---|---|---|
| 9 | **PWRBTN_LVL**：power button level;<br>这位记录了 PPWBTNn 信号（开机键）的状态。<br>0：low<br>1：high | RO | Core Well |
| 31：8；7：0 | 保留位； | RO | Core Well |

### 26.4.2 GEN_PMCON_2: General PM Configuration2 Register

| 偏移地址 | 读写特性 | 默认值 | 大小 | 用处 | 电源域 |
|---|---|---|---|---|---|
| 0x34 | R/WC, R/W | 0000h | 32bits | ACPI | Resume Well |

| 位 | 描述 | 读 写特性 | 电源域 |
|---|---|---|---|
| 2 | **SRS**: system reset status;<br>0：SYS_RSTn 键（重启键）未按下；<br>1：SYS_RSTn 键（重启键）按下；PMON 读这位，若这位被置位，那么将其清空。 | R/WC | Resume Well |
| 1 | **PWROK_FLR**: powerok failure；<br>0：软件向这位写 1 时，将这位清 0<br>1：当系统处于 S0 状态时，若 PWROK 信号拉低，这将这位置 1； | R/WC | Resume Well |
| 0 | **DRAM_INIT**：这一位在任何情况下都不影响硬件功能。PMON 在初始化内存之前会将这位置 1，当完成内存初始化之后会将其清 0。所以在启动时软件能根据这一位的值来判断之前 PMON 在初始化内存的时候是否被重启给打断。 | R/W | Resume Well |
| 31：3 | 保留位； | RO | Resume Well |

### 26.4.3 GEN_PMCON_3: General PM Configuration3 Register

| 偏移地址 | 读写特性 | 默认值 | 大小 | 用处 | 电源域 |
|---|---|---|---|---|---|
| 0x38 | R/WC, R/W | 0000h | 32bits | ACPI | RTC Well |

| 位 | 描述 | 读 写特性 | 电 源域 |
|---|---|---|---|
| 31：3 | 保留位； | RO | RTC Well |
| 6：5 | **S3 assertion width**:这两位指定了 S3n 最小的有效时间；用来保证主供电电源已经完成了 power-cycled。即当系统进入 S3 状态后，非关键电路的电源会被断掉，但是若立即上电可能会带来一些问题，所以需要规定 S3 有效的最小时间，用来保证再次上电不会带来问题。<br>00: 60-100us;<br>01: 1-1.2ms;<br>10: 50-50.2ms<br>11:2-2.0002s | R/W | RTC Well |
| 4：3 | **S4_assertion _width**：这两位指定了 S4n 最小的有效时间（DRAM has been fully power-cycled）；S4n 用来控制内存是否需要供电，当 S4n 有效时表示内存不需要供电。在这期间不能立即给内存供电，必须在 S4n 有效时间达到这两位制定的值后，才能将 S4n 拉 | R/W | RTC Well |

| | | | |
|---|---|---|---|
| | 高。<br>11:2s<br>10:3s<br>01:4s<br>10:5s | | |
| 2 | **S4 assertion en:**<br>0: S4n 信号的最低有效时间为 4 个 RTC clock；<br>1：S4n 信号的最低有效时间由 S4_assertion_width 来指定； | R/WC | RTC Well |
| 1 | **PWR_FLR**：这位在 RTC 电源域中，所以只能被 RTCRSTn 复位。<br>0：表示自从上次清 0 后，系统没有掉电（即 RSMRSTn 未有效）；软件往这位写 1 会将这位清 0；<br>1：表示系统掉电过（RSMRSTn 有效） | R/WC | RTC Well |
| 0 | **AFTERG3_EN**：用来决定当系统掉电后（G3 状态）电源恢复时，系统将进入什么状态；<br>0：当系统重新上电后，系统进入 S0 状态（自动重启）；<br>1：当系统重新商店后，系统进入 S5 状态（即关机状态 soft off）；除非当系统在掉电时处于 S4 状态，那么系统上电后将恢复到 S4 状态。<br>在 S5 状态下，只有开机事件（Power button）才能作为唤醒系统的事件；<br>这一位只能被 RTCRSTn 复位 | R/W | RTC Well |

### 26.4.4 PM1_STS: Power Management 1 Status Register

| 偏移地址 | 读写特性 | 默认值 | 大小 | 用处 | 电源域 |
|---|---|---|---|---|---|
| 0x00 | R/WC | 0000h | 32bits | ACPI | Resume Well；<br>RTC Well;<br>Core Well |

当 bit8 或 bit10 被置位时，若 PM1_EN 寄存器相应位为 1 且系统处于睡眠状态（S3-S5），那么南桥的电源管理模块将产生唤醒时间（wake event）；若这时系统处于工作状态（S0），那么将产生 SCI 中断事件。

| 位 | 描述 | 读写特性 | 电源域 |
|---|---|---|---|
| 31：16 | 保留位 | RO | Resume Well |
| 15 | **WAK_STS:**<br>0：软件往这位写 1，将清空这位；<br>1：当系统处于睡眠状态时，且一个被使能的唤醒事件发生时，这位被置 1； | R/WC | Resume Well |
| 14:12 | 保留位 | RO | |
| 11 | **PBO_STS:** power button override status;<br>0：软件往这位写 1，将清空这位;<br>1:任何时候当 power button override 事件发生时（开机键被按下超过 4s 的时间），这位都将被置 1。power button override 事件发生时，系统将无条件进入 S5 状态，且将 AFTERG3_EN 位置 1。这位在掉电的时候仍维持原值，所以不会被重启，以及 RSMRSTn 复位。只有 RTCRSTn 才会复位这一位。 | R/WC | RTC Well |
| 10 | **RTC_STS:** | R/WC | Resume |

| | | | |
|---|---|---|---|
| | 0: 软件往这位写 1，将清空这位;<br>1: 若 RTC 模块产生报警（alarm）,则置这位。若系统处于睡眠状态且 PM1_EN 位为 1，那么将产生唤醒事件； 若系统处于工作状态下，若 PM1_EN 的响应位为 1，那么产生 SCI 中断。 | | Well |
| 9 | 保留位 | RO | Resume Well |
| 8 | **PWRBTN_STS:** power button status;<br>0: 软件往这位写 1，将清空这位;且若 PWRBTNn 信号维持有效超过 4s（即开机键按下超过 4s），则这位将被硬件清空；<br>1：当 PWRBTNn 维持有效时间超过 16ms 时，这位将被置 1；在 S0 状态下，若 PWRBTN_STS 与 PM1_EN 都为 1，那么将产生 SCI 中断；若系统处于睡眠状态下，若 PWRBTN_STS 位为 1，那么将产生唤醒事件（PWRBTN_EN 位是否为 1）. | R/WC | Resume Well |
| 7：5 | **保留位** | RO | |
| 4 | **BM_STS:** Bus Master Status<br>0：软件往这位写 1，将清空这位;<br>1：当有对 DDR busmaster 请求时，置 1； | | Core Well |
| 3:1 | **保留位；** | RO | |
| 0 | **TMROF_STS**：（PM Timer Overflow Status）<br>0：SCI 中断处理程序将这位清 0；<br>1：当 PM_TMR 的第 24 位发生翻转时，将这位置 1；且若系统处于工作状态（S0），且 PM1_EN 的相应位为 1，那么产生 SCI 中断。 | R/WC | Core well |

### 26.4.5 PM1_EN: Power Management 1 Enable Register

| 偏移地址 | 读写特性 | 默认值 | 大小 | 用处 | 电源域 |
|---|---|---|---|---|---|
| 0x04 | R/W | 0000h | 32bits | ACPI | Resume Well；<br>RTC Well;<br>Core Well |

| 位 | 描述 | 读 写特性 | 电源域 |
|---|---|---|---|
| 31 ：11 | 保留位 | | |
| 10 | **RTC_EN: RTC Event Enable；**这一位用来允许 RTC 事件产生功耗管理事件。位于 RTC 电源域中，只能被 RTCRSTn 复位。<br>0: 当 RTC_STS 有效时，不产生 SCI 中断或唤醒事件；<br>1: 当 RTC_STS 有效时，产生 SCI 中断或唤醒事件； | R/W | RTC Well |
| 9 | 保留位 | RO | |
| 8 | **PWRBTN_EN:** Power Button Enable; 用来使能 PWRBTN_STS 时候能产生 SCI 中断。这一位对 PWRBTN_STS 是否能产生唤醒事件没有影响。<br>0: enable<br>1：disable | R/W | Resume Well |
| 7：1 | **保留位** | RO | |
| 0 | **TMROF_EN**：（PM Timer Overflow Enable）；用来使能 TMROF_STS 是否产生 SCI 中断。<br>0：disable；<br>1：enable | R/W | Core Well |

### 26.4.6 PM1_CNT: Power Management 1 Control Register

| 偏移地址 | 读写特性 | 默认值 | 大小 | 用处 | 电源域 |
|---|---|---|---|---|---|
| 0x08 | R/W, RO | 0000h | 32bits | ACPI | Resume Well；RTC Well |

| 位 | 描述 | 读 写特性 | 电源域 |
|---|---|---|---|
| 31：14 | 保留位 | | |
| 13 | **SLP_EN: Sleep Enable;**这位置为 1 后，系统将进入睡眠状态，具体睡眠的类型由 SLP_TYP 来决定； | WO | Resume Well |
| 12:10 | **SLP_TYP: Sleep Type;**这三位来决定当 SLP_EN 为 1 时，系统进入的睡眠状态。这三位处于 RTC 电源域里，所以只能被 RTCRSTn 复位。<br>000: S0 状态；<br>001：Reserved<br>010：Reserved<br>011：Reserved<br>100：Reserved<br>101：Suspend-to-RAM，挂起到内存，对应 S3 状态；当系统进入 S3 状态时，将有效 S3n 信号；<br>110：Suspend-to-Disk，挂起到磁盘，对应 S4 状态；当系统进入 S4 状态时，将有效 S3n，S4n 信号；<br>111：Soft off。对应 S5 状态；当系统进入 S5 状态时，将有效 S3n，S4n，S5n 信号 | R/W | RTC Well |
| 9：2 | 保留位 | RO | |
| 1 | **BM_RLD:** Bus master Reload; 用来使能 DDR bus master 请求是否使 CPU 退出 C3 状态。<br>0: enable<br>1：disable | R/W | Core Well |
| 0 | **保留位** | RO | |

### 26.4.7 PM1_TMR: Power Management1 Timer Register

| 偏移地址 | 读写特性 | 默认值 | 大小 | 用处 | 电源域 |
|---|---|---|---|---|---|
| 0x0C | RO | 0000h | 32bits | ACPI | Core Well |

| 位 | 描述 | 读写特性 | 电 源域 |
|---|---|---|---|
| 31 ：24 | 保留位 | RO | |
| 23：0 | **TMR_VAL**: Timer Value; 以 1/4 RTC 时钟为频率计数。只要当系统处于 S0 状态，就开始计数。这个寄存器被 PLTRSTn 复位。当第 23 位发生翻转时，将 TMROF_STS 置位；若 TMROF_EN 位为 1，那么将产生 SCI 中断。 | RO | Core Well |

### 26.4.8 PROC_CNT: Processor Control Register

| 偏移地址 | 读写特性 | 默认值 | 大小 | 用处 | 电源域 |
|---|---|---|---|---|---|
| 0x10 | R/W , RO | 0000h | 32bits | ACPI | Core Well |

| 位 | 描述 | 读写特性 | 电源域 |
|---|---|---|---|
| 31：5 | 保留位 | RO | |
| 4 | **THTL_EN**: Throttling Enable；当 cpu 处于 C0 状态时，若置位，则使能处理器降频运行（通过控制 CPU_STPCLKn 来达到降频运行的效果）。由 THTL_DTY 指定处理器运行的频率。 | R/W | |
| 3:1 | **THTL_DTY**: Throttling duty; 用来指定当 THTL_EN 为 1 时处理器运行的频率。南桥芯片中以 1024 个 pclk 为一个周期，来对处理器做降频处理。具体工作原理参考第 5 章。<br>000：保留<br>001：处理运行的工作频率为与原来的 1/8；<br>010：处理运行的工作频率为与原来的 2/8；<br>011：处理运行的工作频率为与原来的 3/8；<br>100：处理运行的工作频率为与原来的 4/8；<br>101：处理运行的工作频率为与原来的 5/8；<br>110：处理运行的工作频率为与原来的 6/8；<br>111：处理运行的工作频率为与原来的 7/8； | R/W | Core Well |

### 26.4.9 LVL2: Level 2 Register

| 偏移地址 | 读写特性 | 默认值 | 大小 | 用处 | 电源域 |
|---|---|---|---|---|---|
| 0x14 | RO | 0000h | 32bits | ACPI | Core Well |

| 位 | 描述 | 读写特性 | 电源域 |
|---|---|---|---|
| 31：0 | 读这个寄存器时，返回全 0；写这个寄存器则不会产生任何效果。当读这个寄存器时，将会产生"处理器进入 C2 状态"的效果。即 CPU_STPCLKn 会一直有效，直到发生中断事件，那么处理器才会返回工作状态（C0）。 | RO | Core well |

### 26.4.10 LVL3: Level 3 Register

| 偏移地址 | 读写特性 | 默认值 | 大小 | 用处 | 电源域 |
|---|---|---|---|---|---|
| 0x18 | RO | 0000h | 32bits | ACPI | Core Well |

| 位 | 描述 | 读写特性 | 电源域 |
|---|---|---|---|
| 31：0 | 读这个寄存器时，返回全 0；写这个寄存器则不会产生任何效果。当读这个寄存器时，将会产生"处理器进入 C3 状态"的效果。即 CPU_STPn 会一直有效，直到发生中断事件，那么处理器才会返回工作状态（C0）。 | RO | Core well |

### 26.4.11 GPE0_STS - General Purpose Event0 Status Register

| 偏移地址 | 读写特性 | 默认值 | 大小 | 用处 | 电源域 |
|---|---|---|---|---|---|
| 0x20 | R/W | 0000h | 32bits | ACPI | Resume Well |

| 位 | 描述 | 读 写特性 | 电源域 |
|---|---|---|---|
| 31：16 | **GPIO_STS[15:0]：**<br>0: 软件写 1，将清空此位；<br>1：当 GPIO 作为输入，且对应的 GPIO 信号为高时，将对应的位置 1；若 GPE0_EN 对应的位为 1，那么当系统处于睡眠状态时，产生唤醒事件；否则若系统处于工作状态，那么产生 SCI 中断事件。 | R/WC | Resume well |
| 15 | **USB6_STS：**<br>0: 软件写 1，清空此位；<br>1：当 USB 的端口 6 发生了唤醒事件，那么将这位置 1；若同时 GPE0_EN 对应的 USB6_EN 位为 1，那么将产生唤醒事件致使系统从睡眠状态中退出。<br>USB 端口支持的唤醒事件包括：设备的插入，拔出；J-K 状态的转换； | R/WC | Resume Well |
| 14 | **USB5_STS：**<br>0: 软件写 1，清空此位；<br>1：当 USB 的端口 5 发生了唤醒事件，那么将这位置 1；若同时 GPE0_EN 对应的 USB5_EN 位为 1，那么将产生唤醒事件致使系统从睡眠状态中退出。<br>USB 端口支持的唤醒事件包括：设备的插入，拔出；J-K 状态的转换； | R/WC | Resume Well |
| 13 | **USB4_STS：**<br>0: 软件写 1，清空此位；<br>1：当 USB 的端口 4 发生了唤醒事件，那么将这位置 1；若同时 GPE0_EN 对应的 USB4_EN 位为 1，那么将产生唤醒事件致使系统从睡眠状态中退出。<br>USB 端口支持的唤醒事件包括：设备的插入，拔出；J-K 状态的转换； | R/WC | Resume Well |
| 12 | **USB3_STS：**<br>0: 软件写 1，清空此位；<br>1：当 USB 的端口 3 发生了唤醒事件，那么将这位置 1；若同时 GPE0_EN 对应的 USB3_EN 位为 1，那么将产生唤醒事件致使系统从睡眠状态中退出。<br>USB 端口支持的唤醒事件包括：设备的插入，拔出；J-K 状态的转换； | R/WC | Resume Well |
| 11 | **USB2_STS：**<br>0: 软件写 1，清空此位；<br>1：当 USB 的端口 2 发生了唤醒事件，那么将这位置 1；若同时 GPE0_EN 对应的 USB2_EN 位为 1，那么将产生唤醒事件致使系统从睡眠状态中退出。<br>USB 端口支持的唤醒事件包括：设备的插入，拔出；J-K 状态的转换； | R/WC | Resume Well |
| 10 | **USB1_STS：**<br>0: 软件写 1，清空此位；<br>1：当 USB 的端口 1 发生了唤醒事件，那么将这位置 1；若同时 GPE0_EN 对应的 USB1_EN 位为 1，那么将产生唤醒事件致使系统从睡眠状态中退出。 | R/WC | Resume Well |

| | | | |
|---|---|---|---|
| | USB 端口支持的唤醒事件包括：设备的插入，拔出；J-K 状态的转换； | | |
| 9 | **PME_STS：PCI Management Event**<br>0: 软件写 1，清空此位<br>1: PCI 设备产生功耗管理事件(即 PMEn 有效);若 PME_EN 位为 1，且系统处于工作状态，那么将产生 SCI 中断。若系统处于睡眠状态（S3-S5）,那么将产生唤醒事件； | R/WC | Resume Well |
| 8 | **RI_STS：**<br>0: 软件写 1，清空此位<br>1: RIn 信号有效则置此位。 | R/WC | Resume Well |
| 7 | **BatLow_STS：**<br>0: 软件写 1，清空此位<br>1: 当 BATLOWn 信号有效时（电池电量不足），置此位为 1；若 BATLOW_EN 位有效，那么将会产生 SCI 中断。注意电池电量不足事件，只能产生 SCI 中断，而不会产生唤醒事件。 | R/WC | Resume Well |
| 6 | **GMAC_STS：**<br>0: 软件写 1，清空此位<br>1: 当 GMAC 控制器产生了唤醒系统的事件时，将此位置 1；若对应的 GMAC_EN 位为 1，那么将产生唤醒事件。 | R/WC | Resume Well |
| 5 | **LID_STS：**<br>0: 软件写 1，清空此位<br>1: LID 有效信号与 LID_POL 一致时，将这位置位。且若 LID_EN 为 1，那么 LID_STS 置位将产生唤醒事件（系统处于睡眠状态）或者 SCI 中断（系统处于工作状态）； | R/WC | Resume Well |
| 4：0 | **保留位** | R/WC | Resume Well |

### 26.4.12 GPE0_EN - General Purpose Event0 Enable Register

| 偏移地址 | 读写特性 | 默认值 | 大小 | 用处 | 电源域 |
|---|---|---|---|---|---|
| 0x24 | R/WC | 0000h | 32bits | ACPI | Resume Well |

| 位 | 描述 | 读 写特性 | 电源域 |
|---|---|---|---|
| 31 ：16 | **GPIO_EN[15:0]：**<br>0: disable<br>1: enable；使能对应的 GPIO_STS 产生唤醒事件或者是 SCI 中断 | R/W | Resume well |
| 15 | **USB6_EN：**<br>0: disable<br>1：enable；使能对应的 USB6_STS 产生唤醒事件；当系统从睡眠状态返回到工作状态时，将产生 SCI 中断。 | R/W | Resume Well |
| 14 | **USB5_EN：**<br>0: disable<br>1：enable；使能对应的 USB5_STS 产生唤醒事件；当系统从睡眠状态返回到工作状态时，将产生 SCI 中断。 | R/W | Resume Well |
| 13 | **USB4_EN：**<br>0: disable<br>1：enable；使能对应的 USB4_STS 产生唤醒事件；当系统 | R/W | Resume Well |

| | | | |
|---|---|---|---|
| | 从睡眠状态返回到工作状态时，将产生 SCI 中断。 | | |
| 12 | **USB3_EN：**<br>0: disable<br>1：enable；使能对应的 USB3_STS 产生唤醒事件；当系统从睡眠状态返回到工作状态时，将产生 SCI 中断。 | R/W | Resume Well |
| 11 | **USB2_EN：**<br>0: disable<br>1：enable；使能对应的 USB2_STS 产生唤醒事件；当系统从睡眠状态返回到工作状态时，将产生 SCI 中断。 | R/W | Resume Well |
| 10 | **USB1_EN：**<br>0: disable<br>1：enable；使能对应的 USB1_STS 产生唤醒事件；当系统从睡眠状态返回到工作状态时，将产生 SCI 中断。 | R/W | Resume Well |
| 9 | **PME_EN：**<br>0: disable<br>1：enable；使能对应的 PME_STS 产生唤醒事件或者是 SCI 中断事件； | R/W | Resume Well |
| 8 | **RI_EN：**<br>0: disable<br>1：enable；使能对应的 RI_STS 产生唤醒事件或者是 SCI 中断 | R/W | RTC Well |
| 7 | **BatLow_EN：**<br>0: disable<br>1：enable；使能对应的 BatLow_STS 产生 SCI 中断。 | R/W | Resume Well |
| 6 | **GMAC_EN：**<br>0: disable<br>1：enable；使能对应的 GMAC_STS 产生唤醒事件或者是 SCI 中断 | R/W | Resume Well |
| 5 | **LID_EN：**<br>0: disable<br>1：enable；使能对应的 LID_STS 产生唤醒事件或者是 SCI 中断 | R/W | Resume Well |
| 4：1 | **保留位** | RO | |
| 0 | **LID_POL: lid 信号有效电平** | R/W | Resume Well |

### 26.4.13 CPU_INIT: CPU Initialization Register

| 偏移地址 | 读写特性 | 默认值 | 大小 | 用处 | 电源域 |
|---|---|---|---|---|---|
| 0x50 | RO，R/W | 0000h | 32bits | ACPI | Core Well |

| 位 | 描述 | 读写特性 | 电源域 |
|---|---|---|---|
| 31：1 | 保留位 | RO | |
| 0 | **INIT_NOW**: 当从 0 切换到 1 时，南桥芯片将立即软重启 CPU；且维持 soft_rstn 有效 2 个 RTC 时钟。 | R/W | Core Well |

### 26.4.14 RST_CNT: Reset Control Register

| 偏移地址 | 读写特性 | 默认值 | 大小 | 用处 | 电源域 |
|---|---|---|---|---|---|
| 0x44 | RO,R/W | 0000h | 32bits | ACPI | Core Well |

| 位 | 描述 | 读 写 | 电 源 |
|---|---|---|---|

| | | 特性 | 域 |
|---|---|---|---|
| 31：2 | 保留位 | RO | |
| 1 | **RST_CPU**: Reset CPU；当从 0 切换到 1，南桥芯片将根据 SYS_RST 位的值，来软重启或者硬重启系统； | R/W | Core Well |
| 0 | **SYS_RST: System Reset；**<br>0：当 RST_CPU 位从 0 切换到 1 时，南桥芯片软重启系统（仅重启 CPU）。且维持 soft_rstn 有效 2 个 RTC 时钟；<br>1：当 RST_CPU 位从 0 切换到 1 时，南桥芯片将硬重启系统。即有效 PLTRST 与 SUS_STATn，且维持有效时间为 16ms。注意硬重启的过程中，S3n，S4n，S5n 无效。 | R/W | Core Well |

# 27 看门狗

## 27.1 概述

在系统中看门狗定时器（WDT，Watch Dog Timer）实际上是一个计数器，一般给看门狗一个大数，程序开始运行后看门狗开始倒计数。如果程序运行正常，过一段时间 CPU 应发出指令让看门狗复位，重新开始倒计数。如果看门狗减到 0 就认为程序没有正常工作，强制整个系统复位。下图是看门狗的实现，系统对看门狗进行配置，看门狗内部有个计数器，同时看门狗里面的比较器比较计数器值是否为零，如果为零就发出软复位信号让系统重启。

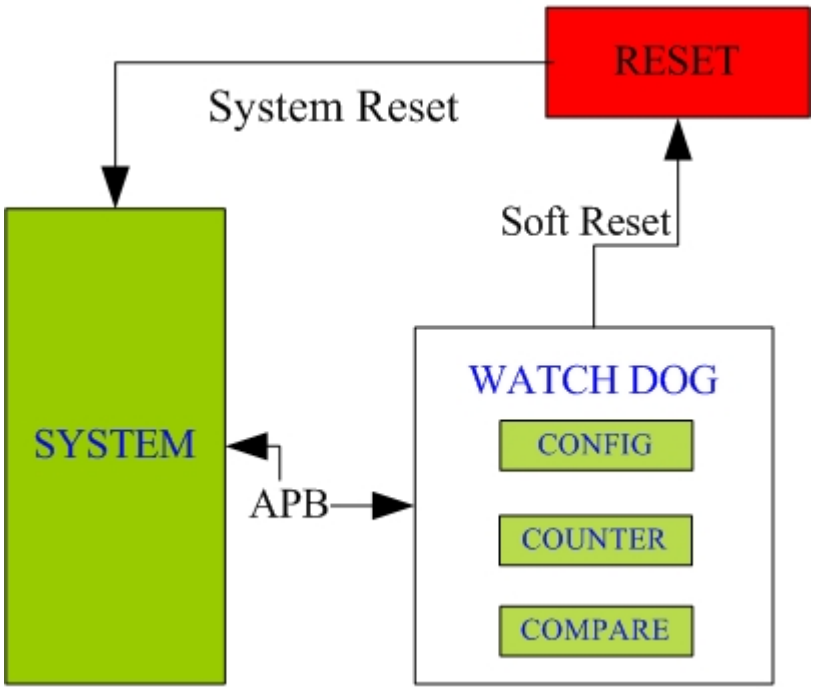



图 27-1看门狗的结构图

## 27.2 WATCH DOG 寄存器描述

看门狗逻辑可编程寄存器主要有三个，这些寄存器描述如下：

### 27.2.1 WDT_EN 地址：（0x1fe7_c060）

| 位 | 位域名 | 读写 | 描述 |
|---|---|---|---|
| 31:1 | | | Reserved |
| 0 | WDT_EN | R/W | 看门狗使能 |

### 27.2.2 WDT_SET（地址：0x1fe7_c068）

| 位 | 位域名 | 读写 | 描述 |
|---|---|---|---|

| | | | |
|---|---|---|---|
| 31:1 | | | Reserved |
| 0 | WDT_SET | R/W | 看门狗中计数器设置 |

27.2.3 WDT_timer（地址：0x1fe7_c064）

| 位 | 位域名 | 读写 | 描述 |
|---|---|---|---|
| 31:0 | WDT_timer | R/W | 看门狗计数器计数值 |

系统这三个寄存器的设置顺序：系统先配置看门狗使能位 WDT_EN；然后配置看门狗开始计数器的初始值 WDT_TIMER，该值保持在一个特别的寄存器中；当系统设置 WDT_SET 后，计数器开始计数。

看门狗不实现低功耗功能，看门狗功能的工作和硬件设计没有关系。如果需要看门狗工作，软件需要定时去更新器计数器的数值

# 28 LPC 控制器

LPC 控制器具有以下特性：

- 符合 LPC1.1 规范
- 支持 LPC 访问超时计数器
- 支持 Memory Read、Memory Write 访问类型（单字节）
- 支持 I/O read、I/O write 访问类型
- 支持 Serizlized IRQ 规范，提供 17 个中断源

LPC 控制器内部的地址空间分布见：

表28-1：

表28-1 LPC控制器地址空间分布

| 地址空间 | 名称 | 大小 |
|---|---|---|
| 0x1fc0,0000 - 0x1fcf,ffff | LPC Boot | 1MB |
| 0x1ff0,0000 - 0x1ff0,ffff | LPC I/O | 64KB |
| 0x1ff1,0200 - 0x1ff1,02ff | LPC regs | 256B |

LPC Boot 地址空间是系统启动时处理器最先访问的地址空间。这个地址空间使用 LPC Memory 访问类型，支持 4Mbit 的 Flash(如 SST49LF040)，映射到 LPC 总线后的地址为 0xfff8,0000~0xffff,ffff。

LPC Memory 地址空间是系统用 Memory/Firmware Memory 访问的地址空间。LPC 控制器发出哪种类型的 Memory 访问，由 LPC 控制器的配置寄存器 LPC_MEM_IS_FWH 决定。处理器发往这个地址空间的地址可以进行地址转换。转换后的地址由 LPC 控制器的配置寄存器 LPC_MEM_TRANS 设置。

处理器发往 LPC I/O 地址空间的访问按照 LPC I/O 访问类型发往 LPC 总线。

地址为地址空间低 16 位。

LPC 控制器配置寄存器共有 4 个 32 位寄存器。配置寄存器的含义见下表：

表27-28-2 LPC 寄存器 0

基地址：0x1ff1_0200

| 位域 | 名称 | 访问 | 初值 | 描述 |
|---|---|---|---|---|
| 31 | sirq_en | R/W | 0 | SIRQ 使能，高有效 |
| 23:16 | mem_trans | R/W | 0 | LPC Memory 空间地址转换控制 |
| 15:0 | timeout | R/W | 0 | LPC 访问超时计数<br>当小于 64 时硬件将当作 63 处理 |

3-28表 LPC 寄存器 1

基地址：0x1ff1_0204

| 位域 | 名称 | 访问 | 初值 | 描述 |
|---|---|---|---|---|
| 31 | mem_fwh | R/W | 0 | LPC Memory 空间访问类型控制<br>0: Memory<br>1: Firmware Memory |
| 17:0 | int_en | R/W | 0 | 中断使能，为 1 的位使能对应的中断源<br>17: timeout<br>16~0: sirq |

4-28表 LPC 寄存器 2

基地址：0x1ff1_0208

| 位域 | 名称 | 访问 | 初值 | 描述 |
|---|---|---|---|---|
| 17:0 | int_src | R/W | 0 | 中断状态 |

5-28表 LPC 寄存器 3

基地址：0x1ff1_020c

| 位域 | 名称 | 访问 | 初值 | 描述 |
|---|---|---|---|---|
| 17 | int_clr | W | - | 写 1 清除超时中断状态 |

# 29 复用和 GPIO

## 29.1 GPIO 结构描述

GPIO 为芯片应用提供了灵活外部接口；部分 PAD 通过 MUX 实现，从而在 BGA448 封装情况下提供丰富的外部功能。

| PAD | 第一复用 | 第二复用 | 第三复用 | 第四复用 |
|---|---|---|---|---|
| INTN0 | GPIO00 | | | |
| INTN1 | GPIO01 | | | |
| VGA_HSYNC | GPIO02 | | | |
| VGA_VSYNC | GPIO03 | | | |
| LCD_CLK | GPIO04 | | | |
| LCD_VSYNC | GPIO05 | | | |
| LCD_HSYNC | GPIO06 | | | |
| LCD_EN | GPIO07 | | | |
| LCD_DAT_B0 | GPIO08 | | | |
| LCD_DAT_B1 | GPIO09 | | | |
| LCD_DAT_B2 | GPIO10 | | | |
| LCD_DAT_B3 | GPIO11 | | | |
| LCD_DAT_B4 | GPIO12 | | | |
| LCD_DAT_B5 | GPIO13 | | | |
| LCD_DAT_B6 | GPIO14 | | | |
| LCD_DAT_B7 | GPIO15 | | | |
| LCD_DAT_G0 | GPIO16 | | | |
| LCD_DAT_G1 | GPIO17 | | | |
| LCD_DAT_G2 | GPIO18 | | | |
| LCD_DAT_G3 | GPIO19 | | | |
| LCD_DAT_G4 | GPIO20 | | | |
| LCD_DAT_G5 | GPIO21 | | | |
| LCD_DAT_G6 | GPIO22 | | | |
| LCD_DAT_G7 | GPIO23 | | | |
| LCD_DAT_R0 | GPIO24 | | | |
| LCD_DAT_R1 | GPIO25 | | | |
| LCD_DAT_R2 | GPIO26 | | | |
| LCD_DAT_R3 | GPIO27 | | | |
| LCD_DAT_R4 | GPIO28 | | | |
| LCD_DAT_R5 | GPIO29 | | | |
| LCD_DAT_R6 | GPIO30 | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| LCD_DAT_R7 | GPIO31 | | | | | | |
| KB_CLK | GPIO32 | | | | | | |
| KB_DAT | GPIO33 | | | | | | |
| MS_CLK | GPIO34 | NAND2_RDY | NAND2_USE_MS | | | | |
| MS_DAT | GPIO35 | NAND2_CS | | | | | |
| AC97_DATA_I | GPIO36 | | | | | | |
| AC97_DATA_O | GPIO37 | | | | | | |
| AC97_SYNC | GPIO38 | | | | | | |
| AC97_RESET | GPIO39 | | | | | | |
| SPI0_CLK | GPIO40 | | | | | | |
| SPI0_MISO | GPIO41 | | | | | | |
| SPI0_MOSI | GPIO42 | | | | | | |
| SPI0_CS | GPIO43 | | | | | | |
| SPI1_SCLK | GPIO44 | NAND_D0 | NAND_D03_USE_SPI1 | GMAC1_TX2 | GMAC1_USE_SPI1 | GMAC1_RX_CTL | NAND_D03_USE_SPI1 |
| SPI1_MISO | GPIO45 | NAND_D1 | | GMAC1_TX3 | | GMAC1_RX0 | |
| SPI1_MOSI | GPIO46 | NAND_D2 | | GMAC1_MDCK | | GMAC1_RX1 | |
| SPI1_CS | GPIO47 | NAND_D3 | | GMAC1_MDIO | | GMAC1_RX2 | |
| UART0_RX | GPIO48 | GMAC1_RX_CTL | GMAC1_USE_UART0 | | | | |
| UART0_TX | GPIO49 | GMAC1_RX0 | | | | | |
| UART0_RTS | GPIO50 | GMAC1_RX1 | | | | | |
| UART0_CTS | GPIO51 | GMAC1_RX2 | | | | | |
| UART0_DSR | GPIO52 | GMAC1_RX3 | | | | | |
| UART0_DTR | GPIO53 | GMAC1_TX_CTL | | | | | |
| UART0_DCD | GPIO54 | GMAC1_TX0 | | | | | |
| UART0_RI | GPIO55 | GMAC1_TX1 | | | | | |
| UART1_RX | GPIO56 | GMAC1_TX2 | GMAC1_USE_UART1 | | | | |
| UART1_TX | GPIO57 | GMAC1_TX3 | | | | | |
| UART1_RTS | GPIO58 | GMAC1_MDCK | | | | | |
| UART1_CTS | GPIO59 | GMAC1_MDIO | | | | | |
| UART2_TX | GPIO60 | | | | | | |
| UART2_RX | GPIO61 | | | | | | |
| UART3_TX | GPIO62 | | | | | | |
| UART3_RX | GPIO63 | | | | | | |
| SCL | GPIO64 | | | | | | |
| SDA | GPIO65 | | | | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| CAN0_RX | GPIO66 | SDA_2 | I2C2_USE_CAN | SPI0_CS2 | SPI0_USE_CAN0_RX | | |
| CAN0_TX | GPIO67 | SCL_2 | | SPI0_CS1 | SPI0_USE_CAN0_TX | | |
| CAN1_RX | GPIO68 | SDA_3 | I2C3_USE_CAN | SPI1_CS2 | SPI1_USE_CAN1_RX | NAND3_RDY | NAND3_USE_CAN |
| CAN1_TX | GPIO69 | SCL_3 | | SPI1_CS1 | SPI1_USE_CAN1_TX | NAND3_CE | |
| LPC_AD0 | GPIO70 | NAND_D0 | NAND_D03_USE_LPC | GMAC1_RX_CTL | NAND_D03_USE_LPC | | |
| LPC_AD1 | GPIO71 | NAND_D1 | | GMAC1_RX0 | | | |
| LPC_AD2 | GPIO72 | NAND_D2 | | GMAC1_RX1 | | | |
| LPC_AD3 | GPIO73 | NAND_D3 | | GMAC1_RX2 | | | |
| LPC_FRAME | GPIO74 | NAND_D4 | NAND_D45_USE_LPC | GMAC1_RX3 | NAND_D45_USE_LPC | | |
| LPC_SERIRQ | GPIO75 | NAND_D5 | | GMAC1_TX_CTL | | | |
| NAND_CLE | GPIO76 | GMAC1_RX_CTL | GMAC1_USE_NAND | | | | |
| NAND_ALE | GPIO77 | GMAC1_RX0 | | | | | |
| NAND_RD | GPIO78 | GMAC1_RX1 | | | | | |
| NAND_WR | GPIO79 | GMAC1_RX2 | | | | | |
| NAND_CE | GPIO80 | GMAC1_RX3 | | | | | |
| NAND_RDY | GPIO81 | GMAC1_TX_CTL | | | | | |
| NAND_D6 | GPIO82 | GMAC1_TX0 | | | | | |
| NAND_D7 | GPIO83 | GMAC1_TX1 | | | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| PWM0 | GPIO84 | NAND1_RDY | NAND1_USE_PWM | | | MAC0_COL | GMAC0_USE_PWM01 |
| PWM1 | GPIO85 | NAND1_CE | | | | MAC0_CRS | |
| PWM2 | GPIO86 | NAND_D4 | NAND_D45_USE_PWM | GMAC1_RX3 | NAND_D45_USE_PWM | MAC1_COL | GMAC1_USE_PWM23 |
| PWM3 | GPIO87 | NAND_D5 | | GMAC1_TX_CTL | | MAC1_CRS | |

注 1：芯片作为南桥时候，INTN0/INTN1 中断输出到处理器；芯片做 SoC 的时候，INTN0/INTN1 可以作为 GPIO，也可以作为外部 PCI_INTA/PCI_INTB 的输入。

## 29.2 GPIO 寄存器描述

| 偏移地址 | 宽 | 寄存器 | 描述 | 读写 | 描述 |
|---|---|---|---|---|---|
| 0x1fd010C0 | 32 | GPIOCFG0 | 配置寄存器 0<br>复位值<br>0xffffffff | R/W | GPIOCFG0[31:0] 分别对应 GPIO31:GPIO0<br>1:对应 PAD 为 GPIO 功能<br>0:对应 PAD 为普通功能 |
| 0x1fd010C4 | 32 | GPIOCFG1 | 配置寄存器 1<br>复位值<br>0xfffff0ff | R/W | GPIOCFG1[31:0] 分别对应 GPIO63:GPIO32<br>1:对应 PAD 为 GPIO 功能<br>0:对应 PAD 为普通功能 |
| 0x1fd010C8 | 32 | GPIOCFG2 | 配置寄存器 2<br>复位值<br>0x00f0003f | R/W | GPIOCFG2[23:0] 分别对应 GPIO87:GPIO64<br>1:对应 PAD 为 GPIO 功能<br>0:对应 PAD 为普通功能 |
| 0x1fd010D0 | 32 | GPIOOE0 | 输入使能寄存器 0<br>复位值<br>0xffffffff | R/W | GPIOOE0[31:0] 分别对应 GPIO31:GPIO0<br>1:对应 GPIO 被控制为输入<br>0:对应 GPIO 被控制为输出 |
| 0x1fd010D4 | 32 | GPIOOE1 | 输入使能寄存器 1<br>复位值<br>0xfffff0ff | R/W | GPIOOE1[31:0] 分别对应 GPIO63:GPIO32<br>1:对应 GPIO 被控制为输入<br>0:对应 GPIO 被控制为输出 |
| 0x1fd010D8 | 32 | GPIOOE2 | 输入使能寄存器 2<br>复位值<br>0x00f0003f | R/W | GPIOOE2[23:0] 分别对应 GPIO87:GPIO64<br>1:对应 GPIO 被控制为输入<br>0:对应 GPIO 被控制为输出 |
| 0x1fd010E0 | 32 | GPIOIN0 | 输入寄存器 0 | R | GPIOIN0[31:0] 分别对应 GPIO31:GPIO0<br>1:对应 GPIO 输入值为 1<br>0:对应 GPIO 输入值为 0 |
| 0x1fd010E4 | 32 | GPIOIN1 | 输入寄存器 1 | R | GPIOIN1[31:0] 分别对应 |

| | | | | | |
|---|---|---|---|---|---|
| | | | | | GPIO63:GPIO32<br>1:对应 GPIO 输入值为 1<br>0:对应 GPIO 输入值为 0 |
| 0x1fd010E8 | 32 | GPIOIN2 | 输入寄存器 2 | R | GPIOIN2[23:0] 分别对应 GPIO87:GPIO64<br>1:对应 GPIO 输入值为 1<br>0:对应 GPIO 输入值为 0 |
| 0x1fd010F0 | 32 | GPIOOUT0 | 配置输出寄存器 0<br>复位值<br>0x0 | R/W | GPIOOUT0[31:0] 分别对应 GPIO31:GPIO0<br>1:对应 GPIO 输出值为 1<br>0:对应 GPIO 输出值为 0 |
| 0x1fd010F4 | 32 | GPIOOUT1 | 配置输出寄存器 1<br>复位值<br>0x0 | R/W | GPIOOUT1[31:0] 分别对应 GPIO63:GPIO32<br>1:对应 GPIO 输出值为 1<br>0:对应 GPIO 输出值为 0 |
| 0x1fd010F8 | 32 | GPIOOUT2 | 配置输出寄存器 2<br>复位值<br>0x0 | R/W | GPIOOUT2[23:0] 分别对应 GPIO87:GPIO64<br>1:对应 GPIO 输出值为 1<br>0:对应 GPIO 输出值为 0 |

## 29.3 MUX 寄存器描述

GPIO_MUX_CTRL0 基地址 0x1fd0_0420,寄存器的描述如下，当管脚配置为 GPIO 功能时，MUX 寄存器的配置不起作用：

| 位 | 描述 | 读写特性 |
|---|---|---|
| 31 | NAND3_USE_CAN1 | R/W |
| 30 | NAND2_USE_MS | R/W |
| 29 | NAND1_USE_PWM01 | R/W |
| 28 | NAND_D45_USE_PWM23 | R/W |
| 27 | NAND_D45_USE_LPC | R/W |
| 26 | NAND_D03_USE_SPI1 | R/W |
| 25 | NAND_D03_USE_LPC | R/W |
| 24 | GMAC1_SHUT | R/W |
| 23 | GMAC0_SHUT | R/W |
| 22 | SATA_SHUT | R/W |
| 21 | USB_SHUT | R/W |
| 20 | GPU_SHUT | R/W |
| 19 | DDR2_SHUT | R/W |
| 18 | VGA_USE_PCI | R/W |
| 17 | I2C3_USE_CAN0 | R/W |
| 16 | I2C2_USE_CAN1 | R/W |
| 15 | SPI0_USE_CAN0_TX | R/W |
| 14 | SPI0_USE_CAN0_RX | R/W |
| 13 | SPI1_USE_CAN1_TX | R/W |
| 12 | SPI1_USE_CAN1_RX | R/W |
| 11 | GMAC1_USE_TXCLK | R/W |
| 10 | GMAC0_USE_TXCLK | R/W |

| | | |
|---|---|---|
| 9 | GMAC1_USE_PWM23 | R/W |
| 8 | GMAC0_USE_PWM01 | R/W |
| 7 | GMAC1_USE_UART1 | R/W |
| 6 | GMAC1_USE_UART0 | R/W |
| 5 | GMAC1_USE_SPI1 | R/W |
| 4 | GMAC1_USE_NAND | R/W |
| 3 | | 保留 |
| 2 | PCI_REQ2_USE_GMAC1 | R/W |
| 1 | DISABLE_DDR2_CONFSPACE | R/W |
| 0 | DDR32TO16EN | R/W |