龙芯中科
LOONGSON TECHNOLOGY

LOONGSON

龙芯 2K0500 处理器

用户手册

V1.0

2023年04月

龙芯中科技术股份有限公司

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
LOONGSON TECHNOLOGY

版权声明

免责声明

本文档仅提供阶段性信息，所含内容可根据产品的实际情况随时更新，恕不另行通知。如因文档使用不当造成的直接或间接损失，本公司不承担任何责任。

龙芯中科技术股份有限公司

Loongson Technology Corporation Limited
地址：北京市海淀区中关村环保科技示范园龙芯产业园 2 号楼
Building No.2, Loongson Industrial Park,
Zhongguancun Environmental Protection Park, Haidian District, Beijing
电话(Tel)：010-62546668
传真(Fax)：010-62600826

阅读指南

《龙芯 2K0500 处理器用户手册》主要介绍龙芯 2K0500 的架构与寄存器描述，对处理器系统架构、主要模块的功能与配置、寄存器列表及位域进行详细说明。

关于龙芯 2K0500 处理器所集成的 LA264 高性能处理器核的相关资料，请参阅《龙芯 LA264 处理器核用户手册》。

版本信息

版本信息	文档	龙芯 2K0500 处理器用户手册
	版本号	V1.0
	创建人	芯片研发部
版本历史		

序号.	版本号	更新内容
1	V1.0	第一版发布

目 录

图目录

表目录

1 概述

龙芯 2K0500 是龙芯 2 号处理器芯片，为面向多场景应用 SOC 型产品，其目标是为工控互联网应用、打印终端、BMC 等提供多种解决方案。龙芯 2K0500 采用 40nm 工艺实现，典型主频 500MHz。片内集成定点处理单元、浮点处理单元、图形图像处理单元，以及南桥、北桥等配套芯片组功能。

龙芯 2K0500 具有以下主要特性：

- 集成一个 LA264 双发射龙芯处理器核，L1 Cache(I/D) 32KB，L2 Cache 512KB
- 集成 1 个 32 位 DDR3 控制器
- 集成 2D GPU
- 集成两路显示控制器，最大分辨率可支持到 1920*1080@60Hz/24bit
- 集成 PCIE 2.0 控制器，支持 2 路 x1，支持 RC/EP 模式
- 集成 PCI 控制器，支持 HOST/DEVICE
- 集成 2 个 SATA2.0 接口
- 集成 2 个 10M/100M/1000M 自适应 GMAC，支持 RGMII
- 集成 4 个 USB2.0 HOST 接口，其中 1 个可配置为 OTG 接口
- 集成 1 个 USB3.0 HOST 接口，支持 OTG 模式
- 集成 1 个 8 位 NAND FLASH 控制器，支持 MLC
- 集成 6 个 SPI 控制器，1 路支持系统启动
- 集成 1 个 LPC 控制器，支持主、从模式
- 集成 6 路 I2C 控制器
- 集成 1 个打印机控制器
- 集成 AC97/HDA 控制器
- 集成 10 路 UART 串口
- 集成 2 个 SDIO 控制器
- 集成 4 个 CAN 控制器
- 集成 1 个 PS/2 控制器
- 集成 16 路 PWM 控制器
- 集成 155 路 GPIO 端口
- 集成 1 个温度传感器
- 集成 RTC/HPET 计时器
- 集成看门狗电路
- 集成 ACPI，支持 GMAC0 网络唤醒

- 集成动态功耗控制模块，支持 DVFS/DPM
- 集成中断控制器，支持灵活的中断设置

1.1 体系结构框图

龙芯 2K0500 内部采用多级总线结构。一级交叉开关连接一个处理器核、一个二级 Cache 以及 IO 子网络（Cache 访问路径）。二级 Cache 及 IODMA、内存控制器、图形媒体模块与 PCIE 、PCI、GMAC、USB、SATA、HDA、PRINT 等 IO 设备共享 128 位互联网络。低速外设（I2C/UART 等）作为一个集合加在南桥总线上。

龙芯 2K0500 芯片结构图如图 1-1 所示：

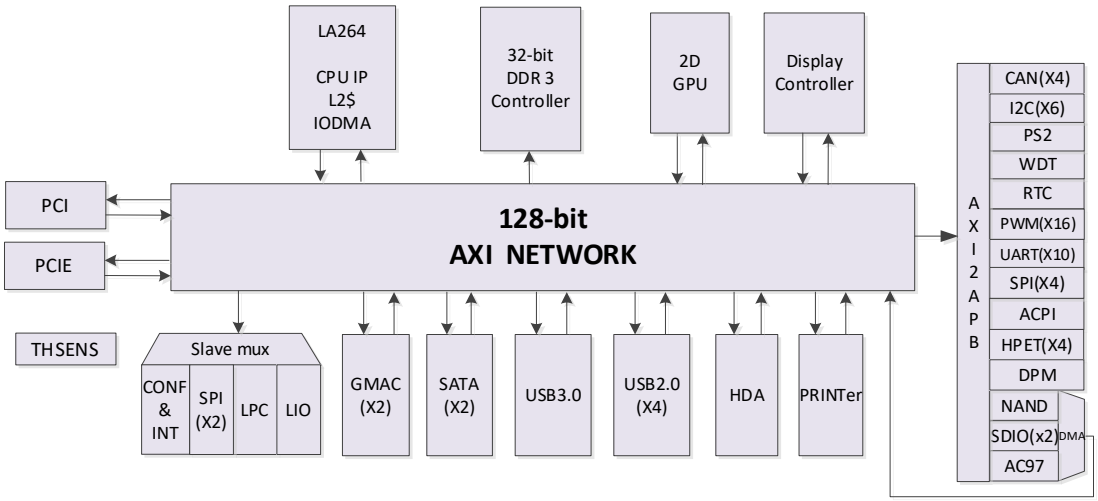


图1-1 龙芯 2K0500 芯片结构图

1.2 芯片主要功能

1.2.1 处理器核

- LA264
- LoongArch 指令体系结构
- 包括 1 个全流水的 64 位双精度浮点乘加部件
- 32KB 数据 Cache 和 32KB 的指令 Cache
- 512KB 共享二级 Cache
- 通过目录协议维护 I/O DMA 访问的 Cache 一致性
- JTAG 支持
- DVFS 支持

1.2.2 内存接口

- 32 位 DDR3 控制器，最高工作频率 533MHz

- 不支持 ECC
- 可配置为 32/16 位模式
- 支持命令调度

1.2.3 GPU

- 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
- 高质量缩放

1.2.4 显示控制器

- 双显示输出(DVO/VGA)
- 硬件光标
- 伽玛校正
- 输出抖动
- 最高像素时钟(DVO165MHz 1080p)
- 支持线性显示缓冲
- 上电序列控制

1.2.5 PCIE 接口

- 兼容 PCIE 2.0
- 双独立 X1 接口
- 双接口均支持 RC/EP

1.2.6 SATA 控制器

- 2 个独立 SATA 端口
- 支持 SATA 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 PCI 接口

- 兼容 PCI2.2，32 位总线宽度
- 既可以做 Host（SoC），又可以做 Device（南桥，外设资源部分删减）
- 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

1.2.8 USB 控制器

- 4 个独立的 USB2.0 的 HOST 端口，端口 0 可配置为 OTG 模式
- 1 个独立的 USB3.0 接口，支持 OTG 模式
- 内部 EHCI 控制和实现高速传输
- 内部 OHCI 控制和实现全速和低速传输

1.2.9 GMAC 控制器

- 两路 10/100/1000Mbps 自适应以太网 MAC
- 双网卡均兼容 IEEE 802.3
- 对外部 PHY 实现 RGMII 接口
- 半双工/全双工自适应
- Timestamp 功能
- 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，支持前置符生成与删除

1.2.10 LPC 控制器

- 兼容 LPC Rev1.1 标准
- 支持系统启动
- 支持主、从模式（从模式仅支持 RAM 存储）

1.2.11 AC97 控制器

- 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

1.2.12 HDA 控制器

- 支持 16，18 和 20 位采样精度支持可变速率
- 最高达 192KHz
- 7.1 频道环绕立体声输出

- 一路音频输入

1.2.13 NAND 控制器

- 最大支持单片 16GB NAND Flash
- 最大支持 4 个片选
- 支持 MLC
- 支持系统启动
- 支持 512/2K/4K/8K 页

1.2.14 SPI 控制器

- 双缓冲接收器
- 极性和相位可编程的串行时钟
- 主模式支持
- 支持到 4 个的变长字节传输
- 支持系统启动(仅 SPI0)
- 支持标准读、连续地址读、快速读、双路 I/O 等 SPI Flash 读模式

1.2.15 UART

- 2 个全功能 UART 和流控 TXD,RXD,CTS, RTS, DSR,DTR,DCD, RI
- 最多 10 个 UART 接口
- 在寄存器与功能上兼容 NS16550A
- 两路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

1.2.16 I2C 总线

- 与 PHILIPS I2C 标准相兼容
- 履行双向同步串行协议
- 实现主/从设备操作
- 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

1.2.17 打印接口

- 支持 JBIG85 解码
- 支持 8 路独立机芯控制
- 支持四色彩打功能

1.2.18 PWM

- 32 位计数器
- 支持脉冲生成及捕获
- 16 路控制器

1.2.19 SDIO 控制器

- 2 路独立 SDIO 控制器
- 兼容 SD Memory 2.0/MMC/SDIO 2.0 协议
- 1 路支持 SDIO 系统启动（SDIO0）

1.2.20 PS/2 控制器

- 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 HPET

- 32 位计数器
- 支持 1 个周期性中断
- 支持 2 个非周期性中断

1.2.22 CAN

- 四路 CAN 接口
- 兼容 CAN2.0 协议

1.2.23 RTC

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

1.2.24 GPIO

- 155 位 GPIO 引脚
- 128 路支持外部中断输入（GPIO0~127）
- 与其他接口复用

1.2.25 Watchdog

- 32 比特计数器及初始化寄存器
- 低功耗模式暂停功能

1.2.26 温度传感器

- 温度观测，支持温度范围：0～70° C
- 高低温中断

1.2.27 中断控制器

- 支持软件设置中断
- 支持电平与边沿触发
- 支持中断屏蔽与使能
- 支持多种中断分发模式

1.2.28 ACPI 功耗管理

- 处理器核动态频率电压调节
- 媒体处理器可关断
- 全芯片时钟门控
- PHY 可关断
- GMAC0 可网络唤醒
- 来电可自动启动

2 引脚定义

2.1 约定

龙芯 2K0500 的引脚进行了大量的功能复用。对于有复用关系的引脚，在介绍完其本身的功能之外会同时在下方给出其他复用功能的描述。

本章对龙芯 2K0500 引脚定义的说明使用以下约定：

- 信号名

信号名的选取以方便记忆和明确标识功能为原则。低有效信号以 N 结尾，高有效信号则不带 N。如无特别说明，以 ACPI 开头的信号位于 RSM 域；以 RTC 开头的信号位于 RTC 域；其它信号位于 SOC 域。

- 类型

信号的输入输出类型由一个代码表示，见表2-1。

表2-1 信号类型代码

代码	描述
A	模拟
DIFF I/O	双向差分
DIFF IN	差分输入
DIFF OUT	差分输出
I	输入
I/O	双向
O	输出
OD	开漏输出
P	电源
G	地

2.2 DDR3 接口

表2-2 DDR3 SDRAM控制器接口信号

信号名称	类型	描述
DDR_DQ[31:0]	I/O	DDR3 SDRAM 数据总线信号
DDR_DQSP[3:0] DDR_DQSN[3:0]	DIFF I/O	DDR3 SDRAM 数据选通
DDR_DQM[3:0]	O	DDR3 SDRAM 数据屏蔽
DDR_A[15:0]	O	DDR3 SDRAM 地址总线信号
DDR_BA[2:0]	O	DDR3 SDRAM 逻辑 BANK 地址信号
DDR_WEN	O	DDR3 SDRAM 写使能信号
DDR_CASN	O	DDR3 SDRAM 列地址选择信号
DDR_RASN	O	DDR3 SDRAM 行地址选择信号
DDR_SCSN[1:0]	O	DDR3 SDRAM 片选信号
DDR_CKE[1:0]	O	DDR3 SDRAM 时钟使能信号

DDR_CKP[1:0] DDR_CKN[1:0]	DIFF OUT	DDR3 SDRAM 差分时钟输出信号
DDR_ODT[1:0]	O	DDR3 SDRAM ODT 信号
DDR_RESETN	O	DDR3 SDRAM 复位控制信号

2.3 PCIE 接口

表2-3 PCIE 总线信号

信号名称	类型	描述
PCIE_REFCLKM PCIE_REFCLKP	DIFF IN	PCIE 100MHz 参考时钟输入
PCIE_CLKOUTP[1:0] PCIE_CLKOUTN[1:0]	DIFF OUT	PCIE 参考时钟输出
PCIE_RESREF	A	外部参考电阻，通过 200ohm(+/-1%)电阻连至地
PCIE_TXP[1:0] PCIE_TXM[1:0]	DIFF OUT	PCIE 差分数据输出
PCIE_RXP[1:0] PCIE_RXM[1:0]	DIFF IN	PCIE 差分数据输入
PCIE_PRSNT[1:0]	I	PCIE 插卡检测
PCIE_RSTN	O	PCIE 复位

2.4 PCI 接口

表2-4 PCI 总线信号

信号名称	类型	描述
PCI_AD[31:00]	I/O	PCI 数据地址线
PCI_RESETN	I/O	PCI 复位，需外部上拉
PCI_CBEN[3:0]	I/O	字节使能，需外部上拉
PCI_DEVSELN	I/O	设备选择，需外部上拉
PCI_FRAMEN	I/O	帧周期，需外部上拉
PCI_IDSEL	I	设备选择信号，当作为 PCI 主控制器时，该信号直接下拉处理；当作为 PCI 设备使用时，该信号作为设备选择和配置读写时的片选信号
PCI_IRDYN	I/O	主设备准备好，需外部上拉
PCI_PAR	I/O	校验位，需外部上拉
PCI_PERR	I/O	奇偶校验，需外部上拉
PCI_REQN[1:0]	I/O	外部设备总线请求输入/到外部仲裁器的总线请求占用输出信号，需外部上拉。当使用外部仲裁器时，该信号作为输出信号
PCI_GNTN[1:0]	I/O	外部仲裁器返回总线请求允许输入/到外部设备的 PCI 总线允许输出，需外部上拉。 当使用外部仲裁器时，该信号作为输入信号
PCI_SERR	I/O	系统错误报告，需外部上拉

PCI_STOPN	I/O	停止数据传送，需外部上拉
PCI_TRDYN	I/O	从设备准备好，需外部上拉

2.5 VGA 显示接口

表2-5 VGA 接口信号

信号名称	类型	描述
VGA_ROUT	A	VGA 红色通道输出
VGA_GOUT	A	VGA 绿色通道输出
VGA_BOUT	A	VGA 蓝色通道输出
VGA_HSYNC	O	VGA 水平同步
VGA_VSYNC	O	VGA 垂直同步
VGA_EN	O	VGA 使能
VGA_COMP	O	VGA 电压补偿，推荐 10nf 电容和 10uf 电容并联至 VGA_A3V3
VGA_REXT	A	外部参考电阻(推荐 510 ohm/1%)

2.6 DVO 显示接口

表2-6 DVO 接口信号

信号名称	类型	描述
LCD_CLK	O	DVO 时钟输出
LCD_HSYNC	O	DVO 水平同步
LCD_VSYNC	O	DVO 垂直同步
LCD_EN	O	DVO 数据有效
LCD_D[23:0]	O	DVO 显示数据

2.7 GMAC 接口

表2-7 GMAC 接口信号

信号名称	类型	描述
GMAC[1:0]_TX_CLK_O	O	RGMII 发送时钟输出
GMAC[1:0]_TX_CLK_I	I	RGMII 发送时钟输入(125MHz 备选时钟，可不接)
GMAC_TX_CTL	O	RGMII 发送控制
GMAC_TXD[3:0]	O	RGMII 发送数据
GMAC[1:0]_RX_CLK_I	I	RGMII 接收时钟
GMAC_RX_CTL	I	RGMII 接收控制
GMAC_RXD[3:0]	I	RGMII 接收数据
GMAC_MDCK	O	SMA 接口时钟，外部需上拉处理
GMAC_MDIO	I/O	SMA 接口数据，外部需上拉处理

2.8 SATA 接口

表2-8 SATA 接口信号

信号名称	类型	描述

信号名称	类型	描述
SATA_REFCLKP[1:0] SATA_REFCLKM[1:0]	I/O	100MHz 差分参考时钟输入(内部有备份时钟，通过软件控制双选；差分输入时内部已包含 100ohm 匹配输入电阻)
SATA_RESREF	A	外部参考电阻，通过 200ohm(+/-1%)电阻连至地
SATA[1:0]_TXP SATA[1:0]_TXN	DIFF OUT	SATA 差分数据输出
SATA[1:0]_RXP SATA[1:0]_RXP	DIFF IN	SATA 差分数据输入
SATA_LEDN[1:0]	OD	SATA 工作状态，低表示有数据传输

2.9 USB 接口

表2-9 USB 接口信号

信号名称	类型	描述
USB0_XI USB0_XO	I/O	12MHz 参考时钟晶振(晶振单端时钟由 XO 管脚输入，XI 管脚接地)
USB[3:0]_TXRTUNE	A	参考电阻，通过 200ohm/1%电阻连接到地
USB[3:0]_DP	I/O	USB D+，内部集成下拉电阻，外部可不做处理
USB[3:0]_DM	I/O	USB D-，内部集成下拉电阻，外部可不做处理
USB0_OVRCUR	I/O	USB0 过流检测，需注意该信号为高有效；其中 USB0 工作在 OTG 模式下为 OTG_VBUS 电源使能控制信号输出
USB[3:1]_OVRCUR	I	USB1~3 过流检测，需注意该信号为高有效
USB0_ID	I	USB0 OTG ID 输入
USB0_VBUS	A	USB0 OTG VBUS 5V 电压输入

2.10 USB3.0 接口

表2-10 USB3.0 接口信号

信号名称	类型	描述
U3_REFCLK_P U3_REFCLK_M	I	100MHz 差分参考时钟
U3_TX_P U3_TX_M	DIFF OUT	USB3.0 差分数据输出
U3_RX_P U3_RX_M	DIFF IN	USB3.0 差分数据输入
U3_REXT	A	参考电阻，通过 200ohm/1%电阻连接到地
U3_DP	I/O	USB2.0 D+
U3_DM	I/O	USB2.0 D-
U3_VBUS	A	USB OTG VBUS 输入

2.11 AC97/HDA 接口

表2-11 AC97/HDA 接口信号

信号名称	类型	描述
AC97_BITCLK	I/O	AC97 BITCLK 输入 HDA BITCLK 输出
AC97_SDATAI	I	AC97/HDA 数据输入
AC97_SDATAO	O	AC97/HDA 数据输出
AC97_SYNC	O	AC97/HDA 同步
AC97_RESET	O	AC97/HDA 复位

2.12 LPC 接口

表2-12 LPC 接口信号

信号名称	类型	描述
LPC_FRAMEn	I/O	LPC FRAME 控制信号，外部需上拉处理 主控：输出 ROM：输入
LPC_AD[3:0]	I/O	LPC 总线数据，外部需上拉处理
LPC_SIRQ	I/O	LPC 中断线，外部需上拉处理

2.13 SPI 接口

表2-13 SPI 接口信号

信号名称	类型	描述
SPI[1:0]_SCK	O	SPI 时钟输出
SPI[1:0]_CSn	O	SPI 片选，外部需上拉处理
SPI[1:0]_MOSI	O	SPI 数据输出
SPI[1:0]_MISO	I	SPI 数据输入

2.14 I^2C 接口

表2-14 I^2C 接口信号

信号名称	类型	描述
I2C_SCL	O	I2C 时钟，外部需上拉处理
I2C_SDA	I/O	I2C 数据，外部需上拉处理

2.15 UART 接口

表2-15 UART 接口信号

信号名称	类型	描述
UART[3:0]_TXD	O	串口数据输出
UART[3:0]_RXD	I	串口数据输入
UART[1:0]_RTS	O	串口数据传输请求
UART_DTR	O	串口初始化完成
UART_RI	I	外部 MODEM 探测到振铃信号

UART[1:0]_CTS	I	设备接收数据就绪
UART_DSR	I	设备初始化完成
UART_DCD	I	外部 MODEM 探测到载波信号

2.16 PRINT 接口

表2-16 PRINT 接口信号

信号名称	类型	描述
PR_INT	I	打印机机芯中断输入信号
PR0_CLK	O	打印机 0/1 号马达时钟输出
PR0_START	O	打印机 0/1 号马达开始信号
PR0_READY	I	打印机 0/1 号马达反馈信号
PR0_HSYNC	I	打印机 0/1 号机芯行同步信号
PR0_ENABLE	O	打印机 0 号机芯使能信号
PR0_SHOLD	O	打印机 0 号机芯数据保持信号
PR0_DATA	O	打印机 0 号机芯打印数据输出
PR1_ENABLE	O	打印机 1 号机芯使能信号
PR1_SHOLD	O	打印机 1 号机芯数据维持信号
PR1_DATA	O	打印机 1 号机芯打印数据输出
PR2_CLK	O	打印机 2/3 号马达时钟输出
PR2_START	O	打印机 2/3 号马达开始信号
PR2_READY	I	打印机 2/3 号马达反馈信号
PR2_HSYNC	I	打印机 2/3 号机芯行同步信号
PR2_ENABLE	O	打印机 2 号机芯使能信号
PR2_SHOLD	O	打印机 2 号机芯数据维持信号
PR2_DATA	O	打印机 2 号机芯打印数据输出
PR3_ENABLE	O	打印机 3 号机芯使能信号
PR3_SHOLD	O	打印机 3 号机芯数据维持信号
PR3_DATA	O	打印机 3 号机芯打印数据输出
PR4_CLK	O	打印机 4/5 号马达时钟输出
PR4_START	O	打印机 4/5 号马达开始信号
PR4_READY	I	打印机 4/5 号马达反馈信号
PR4_HSYNC	I	打印机 4/5 号机芯行同步信号
PR4_ENABLE	O	打印机 4 号机芯使能信号
PR4_SHOLD	O	打印机 4 号机芯数据维持信号
PR4_DATA	O	打印机 4 号机芯打印数据输出
PR5_ENABLE	O	打印机 5 号机芯使能信号
PR5_SHOLD	O	打印机 5 号机芯数据维持信号
PR5_DATA	O	打印机 5 号机芯打印数据输出
PR6_CLK	O	打印机 6/7 号马达时钟输出
PR6_START	O	打印机 6/7 号马达开始信号
PR6_READY	I	打印机 6/7 号马达反馈信号

PR6_HSYNC	I	打印机 6/7 号机芯行同步信号
PR6_ENABLE	O	打印机 6 号机芯使能信号
PR6_SHOLD	O	打印机 6 号机芯数据维持信号
PR6_DATA	O	打印机 6 号机芯打印数据输出
PR7_ENABLE	O	打印机 7 号机芯使能信号
PR7_SHOLD	O	打印机 7 号机芯数据维持信号
PR7_DATA	O	打印机 7 号机芯打印数据输出

2.17 NAND 接口

表2-17 NAND 接口信号

信号名称	类型	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，外部需上拉处理
NAND_RDY	I	NAND 准备好，外部需上拉处理
NAND_D[7:6]	I/O	NAND 数据线，其他数据线采用引脚复用

2.18 SDIO 接口

表2-18 SDIO 接口信号

信号名称	类型	描述
SDIO_CLK	O	SDIO 时钟输出
SDIO_CMD	I/O	SDIO 命令输入输出，外部需上拉处理
SDIO_DATA[3:0]	I/O	SDIO 数据信号，外部需上拉处理

2.19 LIO 接口

表2-19 LocalIO 接口信号

信号名称	类型	描述
LIO_A[22:0]	O	LIO 接口地址总线
LIO_data[15:0]	I/O	LIO 接口数据总线
LIO_CSn[1:0]	O	LIO 接口片选，外部需上拉处理
LIO_WRn	O	LIO 接口写信号
LIO_RDn	O	LIO 接口读信号

2.20 CAN 接口

表2-20 CAN 接口信号

信号名称	类型	描述
CAN0_RX	I	CAN 通道 0 数据接收
CAN0_TX	O	CAN 通道 0 数据发送
CAN1_RX	I	CAN 通道 1 数据接收

CAN1_TX	O	CAN 通道 1 数据发送

2.21 PS2 接口

表2-21 PS2 接口信号

信号名称	类型	描述
KB_CLK	I/O	键盘时钟，外部需上拉处理
KB_DAT	I/O	键盘数据，外部需上拉处理
MS_CLK	I/O	鼠标时钟，外部需上拉处理
MS_DAT	I/O	鼠标数据，外部需上拉处理

2.22 PWM 接口

表2-22 PWM 接口信号

信号名称	类型	描述
PWM[3:0]	I/O	PWM 信号输入输出

2.23 电源管理接口

表2-23 电源管理接口

信号名称	类型	描述
ACPI_SYSRSTn	I	系统复位
RTC_RSMRSTn	I	RSM 复位（RTC 域）,要求在 RSM 域电源稳定至少 1ms 后拉高，在 RSM 域电源降至 95%及以下时立即拉低
RTC_RTCRSTn	I	RTC 复位（RTC 域）,建议在 RTC 电源稳定至少 10ms 后再解除复位
ACPI_RINGn	I	振铃唤醒，外部需上拉处理
ACPI_WAKEn	I	PCIE 唤醒，外部需上拉处理
ACPI_PMEn	I	PCI 唤醒，外部需上拉处理
ACPI_LID	I	屏盖状态
ACPI_PWRTYPE	I	供电来源
ACPI_BATLOWn	I	电源电量低，外部需上拉处理
ACPI_SUSSTATn	O	低功耗状态，外部需上拉处理
ACPI_S3n	O	S3 状态，外部需上拉处理
ACPI_S4n	O	S4 状态，外部需上拉处理
ACPI_S5n	O	S5 状态，外部需上拉处理
ACPI_PLTRSTn	O	平台复位，外部需上拉处理
ACPI_SLPLANn	O	网络电源控制，外部需上拉处理
ACPI_PWRBTNn	I	电源开关，外部需上拉处理
ACPI_PWROK	I	电源有效，外部需上拉处理
ACPI_EN	I	ACPI 使能
ACPI_VSBGATE	O	DDR 接口 ACPI_VSBGATE 控制信号(使用该信号模式需在 ACPI 寄存器中配置使能)

2.24 电源地接口

表2-24 电源地接口

信号名称	类型	描述
VDD_NODE	P	NODE 域 1.1V 独立供电电源
VDD_CORE	P	CORE 域 1.1V 供电电源
VDD_RSM	P	RSM 电压域 1.1V 供电电源
PSU_1V1	P	PCIE/SATA/USB PHY 接口电压域 1.1V 供电电源
PLL_AVDD	P	PLL 模拟电压 1.2V 供电电源
PLL_DDR_AVDD	P	DDR PLL 模拟电压 1.2V 供电电源
PLL_NODE_AVDD	P	NODE PLL 模拟电压 1.2V 供电电源
DDR_VDDE	P	DDR3 电压域 1.5V 供电电源
DDR_VREF	P	DDR3 0.75V 参考电源
IO_3V3	P	IO PAD 电压域 3.3V 供电电源
PSU_3V3	P	PCIE/SATA/USB PHY 电压域 3.3V 供电电源
RSM_3V3	P	RSM 电压域 3.3V 供电电源
RTC_VDD	P	RTC 电压域供电电源
VGA_A3V3	P	VGA 接口 3.3V 供电电源
THSENS_AVDD	P	THSENS 3.3V 供电电源
VSS	G	接地
PLL_AVSS	G	PLL 模拟地
PLL_DDR_AVSS	G	DDR PLL 模拟地
PLL_NODE_AVSS	G	NODE PLL 模拟地

2.25 测试接口

表2-25 测试接口

信号名称	类型	描述
RTC_DOTESTn	I	测试模式控制(RTC 电压域) 0: 测试模式 1: 功能模式

2.26 JTAG 接口

表2-26 JTAG 接口

信号名称	类型	描述
JTAG_SEL	I	JTAG 选择(0: 测试 JTAG，1: 处理器核 JTAG)
JTAG_TCK	I	JTAG 时钟
JTAG_TDI	I	JTAG 数据输入，外部需上拉处理
JTAG_TMS	I	JTAG 模式，外部需上拉处理
JTAG_TRST	I	JTAG 复位，外部需下拉处理
JTAG_TDO	O	JTAG 数据输出

2.27 系统相关信号

表2-27 系统相关信号

信号名称	类型	描述
SYS_CLK	I	100MHz 系统参考时钟
SYS_TESTCLK	I	测试时钟
RTC_XI RTC_XO	I/O	RTC 参考时钟晶体(32.768KHz)
PCI_CLK	I	PCI/LPC 输入时钟，频率 33MHz(接口不使用时该时钟可不接，外部下拉处理)
SYS_INTn[1:0]	I/O	桥片模式下，为芯片中断输出，低电平有效，连接到主芯片的中断输入引脚；SoC 模式下可配置为 GPIO，作为外部中断输入，外部需上拉处理。 SYS_INTn0: INT1/2 SYS_INTn1: INT3/4/5

2.28 上电配置信号

表2-28 上电配置信号

信号名称	类型	描述
LCD_D[20:19]	I	PLL 时钟配置输入 00=低频模式 01=高频模式 10=软件模式 11=bypass 模式
{NAND_RD, NAND_CLE, PWM3}	I	启动选择输入 x00=SPI（SPI0 启动） x01=LPC x10=NAND 011=LIO 111=SDIO（SDIO0 启动）
LCD_D[9]	I	PCIE 参考时钟选择输入 0=内部 100MHz 时钟 1=外部 100MHz 时钟
LCD_D[5]	I	PCIE 端口 0 EP/RC 选择输入 0=RC 1=EP
LCD_D[16]	I	PCIE 端口 1 EP/RC 选择输入 0=RC 1=EP
LCD_D[0]	I	NAND ECC 功能使能输入, 1=enable 0=disable
{LCD_D[13],	I	启动 NAND 类型选择

LCD_D[10]}		00=512Mb(page 512B) 01=1Gb(page 2KB) 10=16Gb(page 4KB) 11=128Gb(page 8KB)
NAND_ALE	I	PCI 外部仲裁选择 0: 使用内部仲裁器 1: 使用外部仲裁器
NAND_WR	I	PCI 主模式选择 0: PCI 桥模式（芯片为 PCI 桥模式） 1: PCI 主模式（芯片为 SOC 模式） 两种模式主要区别在于主模式下 PCI_RESETn 为输出，桥模式下为输入。当选择 PCIX 时，桥模式控制器还会在复位后采样总线频率信息。
NAND_CE	I	PCIX 模式选择 0: PCI 模式 1: PCIX 模式
NAND_D[7:6]	I	PCIX 速度选择（PCI 模式时应为 0） 01: PCIX66 其它: 不可用

2.29 外设功能引脚复用

模块层次的功能复用关系如下表所示：

表2-29 功能引脚复用关系表

芯片主功能	第一复用	第二复用	第三复用	第四复用	上电默认功能(除启动引脚外)
sys_int[0]	-	-	gmac0_ptp_trig	-	GPIO0
sys_int[1]	-	-	gmac0_ptp_pps	-	GPIO1
vga_hsync	-	-	gmac1_ptp_trig	-	GPIO2
vga_vsync	-	pr_int	gmac1_ptp_pps	-	GPIO3
lcd_clk	can2_rx	pr0_clk	-	-	GPIO4
lcd_vsync	can2_tx	pr0_start	-	-	GPIO5
lcd_hsync	can3_rx	pr0_ready	-	-	GPIO6
lcd_en	can3_tx	pr0_enable	-	-	GPIO7
lcd_dat[0]	uart0_tx	pr0_shold	-	-	GPIO8
lcd_dat[1]	uart0_rx	pr0_data	-	-	GPIO9
lcd_dat[2]	uart0_rts	pr0_hsync	-	-	GPIO10
lcd_dat[3]	uart0_cts	pr1_enable	-	-	GPIO11
lcd_dat[4]	uart0_dsr	pr1_shold	-	-	GPIO12
lcd_dat[5]	uart0_dtr	pr1_data	-	-	GPIO13
lcd_dat[6]	uart0_dcd	pr2_clk	-	-	GPIO14
lcd_dat[7]	uart0_ri	pr2_start	-	-	GPIO15
lcd_dat[8]	uart1_rx	pr2_ready	-	-	GPIO16
lcd_dat[9]	uart1_tx	pr2_enable	-	-	GPIO17
lcd_dat[10]	uart1_rts	pr2_shold	-	-	GPIO18
lcd_dat[11]	uart1_cts	pr2_data	-	-	GPIO19

lcd_dat[12]	uart1_dsr	pr2_hsync	-	-	GPIO20
lcd_dat[13]	uart1_dtr	pr3_enable	-	-	GPIO21
lcd_dat[14]	uart1_dcd	pr3_shold	-	-	GPIO22
lcd_dat[15]	uart1_ri	pr3_data	-	-	GPIO23
lcd_dat[16]	-	pr4_clk	spi4_clk	-	GPIO24
lcd_dat[17]	-	pr4_start	spi4_miso	-	GPIO25
lcd_dat[18]	-	pr4_ready	spi4_mosi	uart0_rx	GPIO26
lcd_dat[19]	-	pr4_enable	spi4_cs	uart0_tx	GPIO27
lcd_dat[20]	-	pr4_shold	spi5_clk	uart0_rts	GPIO28
lcd_dat[21]	-	pr4_data	spi5_miso	uart0_cts	GPIO29
lcd_dat[22]	-	pr4_hsync	spi5_mosi	uart0_dsr	GPIO30
lcd_dat[23]	-	pr5_enable	spi5_cs	uart0_dtr	GPIO31
kb_clk	-	pr5_shold	spi3_clk	uart0_dcd	GPIO32
kb_dat	-	pr5_data	spi3_miso	uart0_ri	GPIO33
ms_clk	nand_rdy[2]	pr6_clk	spi3_mosi	pr_int	GPIO34
ms_dat	nand_ce[2]	pr6_start	spi3_cs	pr0_clk	GPIO35
ac97_datai	-	pr6_ready	pix0_scl	pr0_start	GPIO36
ac97_datao	-	pr6_enable	pix0_sda	pr0_ready	GPIO37
ac97_sync	-	pr6_shold	pix1_scl	pr0_enable	GPIO38
ac97_reset	-	pr6_data	pix1_sda	pr0_shold	GPIO39
spi0_clk	kb_clk	pr6_hsync	-	pr0_data	GPIO40
spi0_miso	kb_dat	pr7_enable	-	pr0_hsync	GPIO41
spi0_mosi	ms_clk	pr7_shold	-	pr1_enable	GPIO42
spi0_cs[0]	ms_dat	pr7_data	-	pr1_shold	GPIO43
spi1_clk	gmac1_tx[2]	gmac1_rx_ctl	nand_d[0]	pr1_data	GPIO44
spi1_miso	gmac1_tx[3]	gmac1_rx[0]	nand_d[1]	pr2_clk	GPIO45
spi1_mosi	gmac1_mdck	gmac1_rx[1]	nand_d[2]	pr2_start	GPIO46
spi1_cs[0]	gmac1_mdio	gmac1_rx[2]	nand_d[3]	pr2_ready	GPIO47
uart0_rx	gmac1_rx_ctl	-	scl0	pr2_enable	GPIO48
uart0_tx	gmac1_rx[0]	-	sda0	pr2_shold	GPIO49
uart0_rts	gmac1_rx[1]	pwm[0]	scl1	pr2_data	GPIO50
uart0_cts	gmac1_rx[2]	pwm[1]	sda1	pr2_hsync	GPIO51
uart0_dsr	gmac1_rx[3]	pwm[2]	scl2	pr3_enable	GPIO52
uart0_dtr	gmac1_tx_ctl	pwm[3]	sda2	pr3_shold	GPIO53
uart0_dcd	gmac1_tx[0]	pwm[4]	scl3	pr3_data	GPIO54
uart0_ri	gmac1_tx[1]	pwm[5]	sda3	pr4_clk	GPIO55
uart1_rx	gmac1_tx[2]	pwm[6]	spi0_clk	pr4_start	GPIO56
uart1_tx	gmac1_tx[3]	pwm[7]	spi0_miso	pr4_ready	GPIO57
uart1_rts	gmac1_mdck	pwm[8]	spi0_mosi	pr4_enable	GPIO58
uart1_cts	gmac1_mdio	pwm[9]	spi0_cs[0]	pr4_shold	GPIO59
uart2_tx	pix0_scl	pwm[10]	spi1_clk	pr4_data	GPIO60
uart2_rx	pix0_sda	pwm[11]	spi1_miso	pr4_hsync	GPIO61
uart3_tx	pix1_scl	pwm[12]	spi1_mosi	pr5_enable	GPIO62
uart3_rx	pix1_sda	pwm[13]	spi1_cs[0]	pr5_shold	GPIO63
scl0	nand_rdy[1]	pwm[14]	spi0_cs[3]	pr5_data	GPIO64
sda0	nand_ce[1]	pwm[15]	spi0_cs[2]	pr6_clk	GPIO65
can0_rx	nand_rdy[2]	sda2	spi0_cs[1]	pr6_start	GPIO66
can0_tx	nand_ce[2]	scl2	spi1_cs[3]	pr6_ready	GPIO67
can1_rx	nand_rdy[3]	sda3	spi1_cs[2]	pr6_enable	GPIO68

can1_tx	nand_ce[3]	scl3	spi1_cs[1]	pr6_shold	GPIO69
lpc_ad[0]	nand_d[0]	sda1	gmac1_rx_ctl	pr6_data	GPIO70
lpc_ad[1]	nand_d[1]	scl1	gmac1_rx[0]	pr6_hsync	GPIO71
lpc_ad[2]	nand_d[2]	sda2	gmac1_rx[1]	pr7_enable	GPIO72
lpc_ad[3]	nand_d[3]	scl2	gmac1_rx[2]	pr7_shold	GPIO73
lpc_frame	nand_d[4]	sda3	gmac1_rx[3]	pr7_data	GPIO74
lpc_serirq	nand_d[5]	scl3	-	-	GPIO75
nand_cle	-	-	-	pwm[0]	GPIO76
nand_ale	-	-	-	pwm[1]	GPIO77
nand_rd	-	-	-	pwm[2]	GPIO78
nand_wr	-	-	-	pwm[3]	GPIO79
nand_ce[0]	-	-	-	pwm[4]	GPIO80
nand_rdy[0]	gmac1_tx_ctl	-	-	pwm[5]	GPIO81
nand_d[6]	gmac1_tx[0]	-	-	pwm[6]	GPIO82
nand_d[7]	gmac1_tx[1]	-	-	pwm[7]	GPIO83
pwm[0]	can0_rx	gmac0_col	nand_rdy[1]	pwm[8]	GPIO84
pwm[1]	can0_tx	gmac0_crs	nand_ce[1]	pwm[9]	GPIO85
pwm[2]	gmac1_rx[3]	gmac1_col	nand_d[4]	pwm[10]	GPIO86
pwm[3]	-	gmac1_crs	nand_d[5]	pwm[11]	GPIO87
gmac0_rx_ctl	pwm[4]	-	uart1_dsr	pwm[12]	GPIO88
gmac0_rx[0]	pwm[5]	-	uart1_dtr	pwm[13]	GPIO89
gmac0_rx[1]	pwm[6]	-	uart1_dcd	pwm[14]	GPIO90
gmac0_rx[2]	pwm[7]	-	uart1_ri	pwm[15]	GPIO91
gmac0_rx[3]	pwm[8]	-	uart2_tx	nand_cle	GPIO92
gmac0_tx_ctl	pwm[9]	-	uart2_rx	nand_ale	GPIO93
gmac0_tx[0]	pwm[10]	-	uart3_tx	nand_rd	GPIO94
gmac0_tx[1]	pwm[11]	-	uart3_rx	nand_wr	GPIO95
gmac0_tx[2]	pwm[12]	pix0_scl	can2_rx	nand_ce[0]	GPIO96
gmac0_tx[3]	pwm[13]	pix0_sda	can2_tx	nand_rdy[0]	GPIO97
gmac0_mdck	pwm[14]	pix1_scl	can3_rx	nand_d[0]	GPIO98
gmac0_mdio	pwm[15]	pix1_sda	can3_tx	nand_d[1]	GPIO99
pci_ad[0]	pr_int	lioa[0]	pwm[0]	nand_d[2]	GPIO100
pci_ad[1]	pr0_clk	lioa[1]	pwm[1]	nand_d[3]	GPIO101
pci_ad[2]	pr0_start	lioa[2]	pwm[2]	nand_d[4]	GPIO102
pci_ad[3]	pr0_ready	lioa[3]	pwm[3]	nand_d[5]	GPIO103
pci_ad[4]	pr0_enable	lioa[4]	pwm[4]	nand_d[6]	GPIO104
pci_ad[5]	pr0_shold	lioa[5]	pwm[5]	nand_d[7]	GPIO105
pci_ad[6]	pr0_data	lioa[6]	pwm[6]	pix0_scl	GPIO106
pci_ad[7]	pr0_hsync	lioa[7]	pwm[7]	pix0_sda	GPIO107
pci_ad[8]	pr1_enable	lioa[8]	pwm[8]	pix1_scl	GPIO108
pci_ad[9]	pr1_shold	lioa[9]	pwm[9]	pix1_sda	GPIO109
pci_ad[10]	pr1_data	lioa[10]	pwm[10]	-	GPIO110
pci_ad[11]	pr2_clk	lioa[11]	pwm[11]	-	GPIO111
pci_ad[12]	pr2_start	lioa[12]	pwm[12]	-	GPIO112
pci_ad[13]	pr2_ready	lioa[13]	pwm[13]	spi2_clk	GPIO113
pci_ad[14]	pr2_enable	lioa[14]	pwm[14]	spi2_miso	GPIO114
pci_ad[15]	pr2_shold	lioa[15]	pwm[15]	spi2_mosi	GPIO115
pci_ad[16]	pr2_data	lio_data[0]	uart1_rx	spi2_cs	GPIO116
pci_ad[17]	pr2_hsync	lio_data[1]	uart1_tx	spi3_clk	GPIO117

pci_ad[18]	pr3_enable	lio_data[2]	uart1_rts	spi3_miso	GPIO118
pci_ad[19]	pr3_shold	lio_data[3]	uart1_cts	spi3_mosi	GPIO119
pci_ad[20]	pr3_data	lio_data[4]	uart1_dsr	spi3_cs	GPIO120
pci_ad[21]	pr4_clk	lio_data[5]	uart1_dtr	spi4_clk	GPIO121
pci_ad[22]	pr4_start	lio_data[6]	uart1_dcd	spi4_miso	GPIO122
pci_ad[23]	pr4_ready	lio_data[7]	uart1_ri	spi4_mosi	GPIO123
pci_ad[24]	pr4_enable	lio_data[8]	gmac0_col	spi4_cs	GPIO124
pci_ad[25]	pr4_shold	lio_data[9]	gmac0_crs	spi5_clk	GPIO125
pci_ad[26]	pr4_data	lio_data[10]	gmac1_col	spi5_miso	GPIO126
pci_ad[27]	pr4_hsync	lio_data[11]	gmac1_crs	spi5_mosi	GPIO127
pci_ad[28]	pr5_enable	lio_data[12]	-	spi5_cs	GPIO128
pci_ad[29]	pr5_shold	lio_data[13]	-	can0_rx	GPIO129
pci_ad[30]	pr5_data	lio_data[14]	-	can0_tx	GPIO130
pci_ad[31]	pr6_clk	lio_data[15]	-	can1_rx	GPIO131
pci_cbe[0]	pr6_start	lioa[16]	-	can1_tx	GPIO132
pci_cbe[1]	pr6_ready	lioa[17]	-	can2_rx	GPIO133
pci_cbe[2]	pr6_enable	lioa[18]	-	can2_tx	GPIO134
pci_cbe[3]	pr6_shold	lioa[19]	-	can3_rx	GPIO135
pci_frame	pr6_data	lioa[20]	-	can3_tx	GPIO136
pci_irdy	pr6_hsync	lioa[21]	-	-	GPIO137
pci_devsel	pr7_enable	lioa[22]	gmac1_mdck	-	GPIO138
pci_trdy	pr7_shold	liocsn[0]	gmac1_mdio	-	GPIO139
pci_stop	pr7_data	liocsn[1]	spi2_clk	-	GPIO140
pci_idsel	pix0_scl	liowrn	spi2_miso	-	GPIO141
pci_par	pix0_sda	liordn	spi2_mosi	-	GPIO142
pci_perr	pix1_scl	-	spi2_cs	sdio1_clk	GPIO143
pci_serr	pix1_sda	-	spi3_clk	sdio1_cmd	GPIO144
pci_req[0]	-	gmac0_ptp_trig	spi3_miso	sdio1_d[0]	GPIO145
pci_req[1]	-	gmac0_ptp_pps	spi3_mosi	sdio1_d[1]	GPIO146
pci_gnt[0]	-	gmac1_ptp_trig	spi3_cs	sdio1_d[2]	GPIO147
pci_gnt[1]	-	gmac1_ptp_pps	-	sdio1_d[3]	GPIO148
sdio_clk	lpc_ad[0]	-	-	-	GPIO149
sdio_cmd	lpc_ad[1]	-	-	-	GPIO150
sdio_d[0]	lpc_ad[2]	gmac0_ptp_trig	-	-	GPIO151
sdio_d[1]	lpc_ad[3]	gmac0_ptp_pps	-	-	GPIO152
sdio_d[2]	lpc_frame	gmac1_ptp_trig	-	-	GPIO153
sdio_d[3]	lpc_serirq	gmac1_ptp_pps	-	-	GPIO154

3 时钟结构

龙芯 2K0500 的时钟结构如图3-1 所示，片内主要由两路外部时钟输入，其中一路时钟频率 33MHz，作为 PCI/LPC 控制器主控时钟；另一路为一个 100MHz 固定频率时钟作为系统参考时钟输入，供片内 PLL 使用。芯片内部共有 5 个独立的 PLL，其中每个 PLL 最多可以提供 3 组频率上相互依赖的时钟输出。这 5 个 PLL 的用途分别为：

一个 NODE PLL 仅用于产生 node 时钟，该时钟为芯片的主要时钟，供 CPU 核、二级 Cache、一二级交叉开关以及 IO 子网络使用；

一个 DDR PLL 同时产生 DDR、NETWORK、PRINT 以及 HDA 的时钟；

一个 SOC PLL 同时产生 GPU 控制器内部时钟和 GMAC 控制器时钟，进一步产生 APB，SATA 以及 USB 的时钟；

两个 PIX PLL 各产生一路像素时钟供 DC 使用，以便支持双路独立显示；

除了内部的 PLL 之外，对于 SATA、PCIE、USB 采用 PHY 自身产生时钟的模块，也使用外部输入的参考时钟进行参考时钟通路设计。

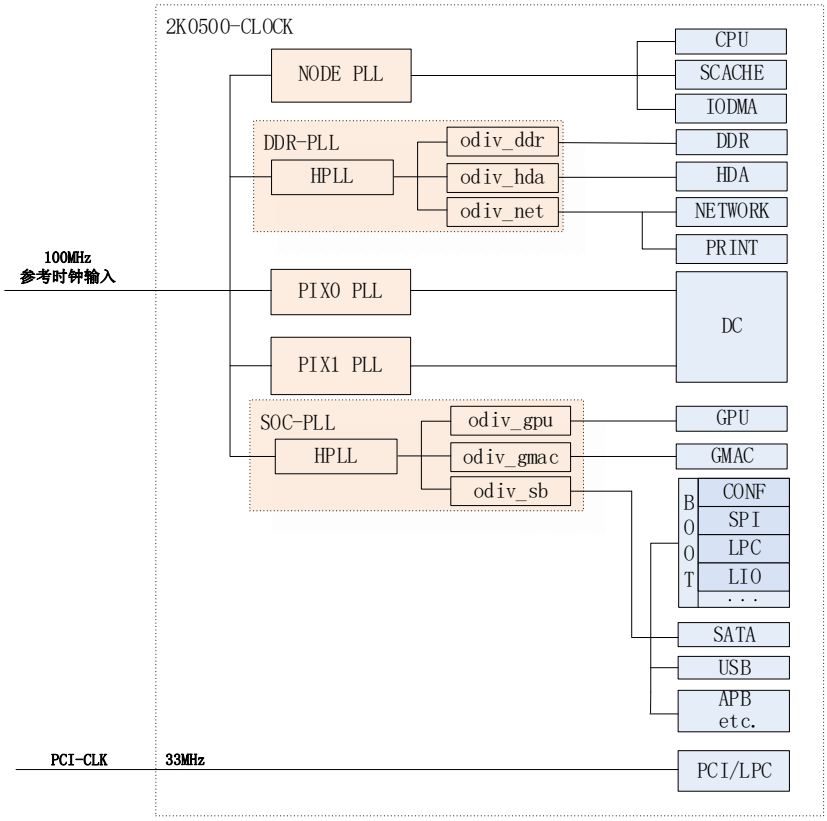


图3-1 时钟结构

3.1 NODE PLL

node clock 的产生结构图如下图所示，

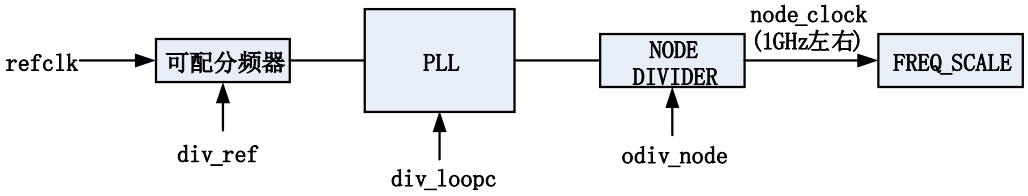


图3-2 NODE PLL 结构图

输出时钟频率的计算方式如下：

node_clock=refclk/div_ref*div_loopc/odiv_node；

node_clock 的工作频率在 500M~800MHz 左右，其 PLL 的分频系数以及倍频系数可以任意配置，但是需要保证可配分频器的输出 refclk/div_ref 在 20~40MHz 范围内，PLL 倍频值 refclk/div_ref*div_loopc 需要在 1.2GHz~3.2GHz。该限制对其他 4 个内部 PLL 也适用，所以下文不再赘述。

输出的时钟还可以经由 freq_scale 模块进行细粒度分频控制。具体分频方法请参考 5.5.21 节。

3.2 DDR PLL

DDR PLL 会输出三个时钟，分别为：

- ✓ ddr_clock 用于内存控制器，频率范围 400-600MHz
- ✓ network_clock 用于 NETWORK/DC/PRINT 模块，频率范围 200-400MHz
- ✓ hda_clock 用于 HDA 模块，频率固定为 24MHz

因为三个时钟共用一个 PLL，只是通过设置各自的 divout 值来实现不同的频率输出。所以在调整其中一个模块时钟时，如果对公用 PLL 的倍频系数进行了调整，那么需要注意对其他时钟的影响。

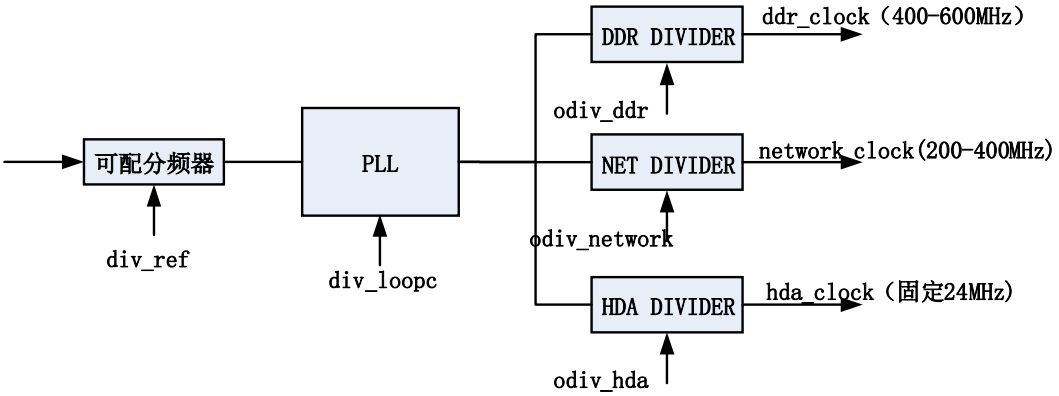


图3-3 DDR PLL 时钟结构

3.3 SOC PLL

SOC PLL 结构与 DDR PLL 结构基本相同，输出三个时钟，分别为：

- ✓ gpu_clock 用于 GPU 控制器，频率范围 200-300MHz

- ✓ sb_clock 用于 BOOT、SATA、USB、APB 模块，频率范围 100-200MHz
- ✓ gmac_clock 用于 GMAC 模块，频率固定为 125MHz

因为三个时钟共用一个 PLL，只是通过设置各自的 divout 值来实现不同的频率输出。所以在调整其中一个模块时钟时，如果对公用 PLL 的倍频系数进行了调整，那么需要注意对其他时钟的影响。

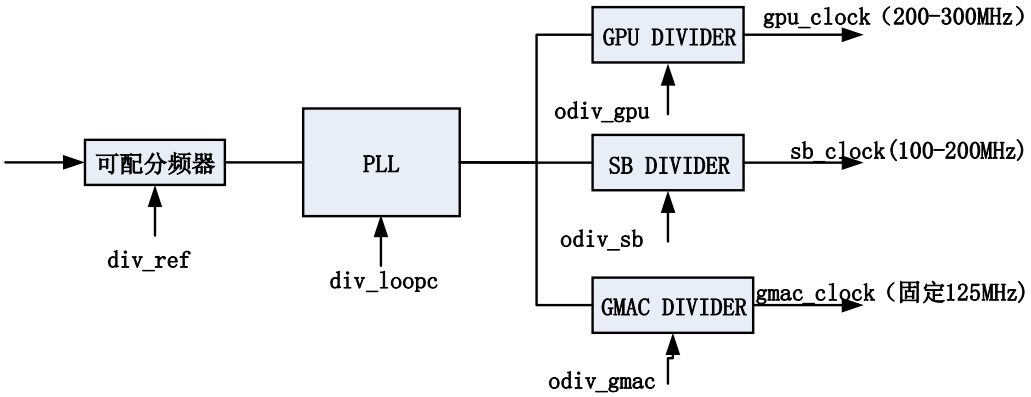


图3-4 SOC PLL 时钟结构

3.4 PIX PLL

PIX PLL 结构基本与上述 PLL 结构相同，但是不包含 freq_scale 模块。2K0500 内包含两个独立的 PIX PLL 用于双路显示输出。像素时钟 pix_clock 频率范围 100-200MHz。

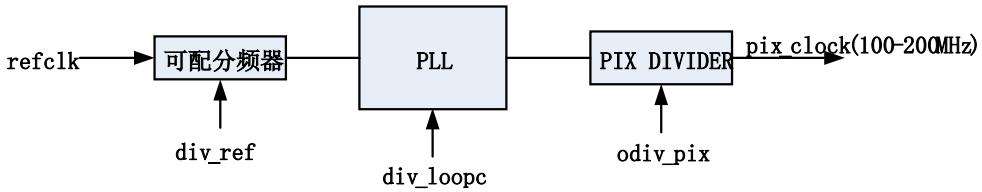


图3-5 PIX PLL 时钟结构

3.5 内部 PLL 配置方法

以上 5 个内部 PLL 提供硬件配置和软件配置两种配置方法。这两种配置方法通过LCD_D[20:19]的设置来区分。

3.5.1 硬件配置

具体如下表所示。

表 3-1 PLL 硬件配置

LCD_D[20:19]	00(硬件低频)	01（硬件高频）	10	11
NODE	500M	800M	软件配置	硬件 bypass 所有 PLL，所有时钟频率与参考时钟相同（100MHz）
DDR	480M	600M		
NETWORK	320M	400M		
GPU	200M	300M		

HDA	24M	24M		
DC	200M	300M		
PIX0	100M	200M		
PIX1	100M	200M		
GMAC	不可硬件配置	不可硬件配置		
SATA	100M	150M		
USB	100M	150M		
APB	100M	150M		

3.5.2 软件配置

当 LCD_D[20:19]设置为 2'b10 时表示 PLL 频率通过软件配置。这种配置下，默认对应的时钟频率为内部参考时钟频率，即所有 PLL 输出都是 SYS_REFCLK，需要在处理器启动过程中对时钟进行软件配置。各个时钟设置的过程应该按照以下方式：

1. 将对应的 PLL 的 PD 信号设置为 1；
2. 设置寄存器除了 sel_pll_*及 soft_set_pll 之外的其它寄存器，即这两个寄存器在设置的过程中写为 0；
3. 将对应的 PLL 的 PD 信号设置为 0；
4. 其他寄存器值不变，将 soft_set_pll 设置为 1；
5. 等待寄存器中的锁定信号 locked_*为 1；
6. 设置 sel_pll_*为 1，此时对应的时钟频率将切换为软件设置的频率。

另外，芯片内部配有设备时钟分频配置参数 freqscale，可供软件对部分设备时钟进行再次分频选择。

具体的配置寄存器说明请参考第五章。

3.6 USB 参考时钟

USB PHY 为 4 个独立的单端口 PHY，参考时钟输入提供以下 2 种方式供选择：

- ✓ 使用 1 个 12MHz 晶振输入，其它使用 0 号 PHY 的时钟输出作为参考源
- ✓ 不使用单独的参考时钟输入，都使用 50MHz 的内部参考时钟（100MHz 的 SYSCLK 经二分频后）

3.7 USB3.0 参考时钟

USB3.0 采用 1 个独立的 PHY，为了简化主板设计，提供以下 2 种方式供选择，使用引脚进行控制。

- ✓ 使用 100MHz 差分输入
- ✓ 不使用单独的参考时钟输入，使用 100MHz 的内部参考时钟

3.8 PCIE 参考时钟

PCIE 采用 1 个独立的 x1 PHY，为了简化主板设计，提供以下 2 种方式供选择，使用引脚进行控制。

- ✓ 使用 1 个 100MHz 差分输入
- ✓ 不使用单独的参考时钟输入，使用 100MHz 的内部参考时钟

3.9 SATA 参考时钟输入

SATA PHY 与 PCIE PHY 类似，也提供以下 2 种方式供选择，使用寄存器进行选择。

- ✓ 使用 2 个 100MHz 差分输入
- ✓ 不使用单独的参考时钟输入，都使用 100MHz 的内部参考时钟

3.10 时钟信号说明

表 3-2 统一介绍了龙芯 2K0500 的所有时钟信号说明。

表 3-2 2K0500 时钟信号说明

信号名称	频率(MHz)	类型	描述
SYSCLK	100	I/O	外接系统参考时钟晶振
RTC_XI RTC_XO	32.768K	I/O	RTC 参考时钟晶体
JTAG_TCK	33	I	JTAG 时钟
PCI_CLOCK	33~66	I	PCI 总线接口时钟输入(PCI/LPC)
TEST_CLOCK	100	I	TEST 测试时钟输入
PCIE_REFCLKp/n_I	100	DIFF IN	PCIE 参考时钟输入 如果使用内部 100M 时钟，则可悬空。
SATA0~1_REFCLKp/n_I	100	DIFF IN	SATA 两路参考时钟输入 如果使用内部 100M 时钟，则可悬空。
USB0~3_REFCLKp/n_I	25	DIFF IN	USB2.0 四路参考时钟输入 如果使用内部参考时钟，则可悬空。
U3_REFCLKp/n_I	100	DIFF IN	USB3.0 参考时钟输入 如果使用内部参考时钟，则可悬空。
DDR_CKp[1:0] DDR_CKn[1:0]	400	DIFF OUT	DDR3 SDRAM 差分时钟输出
PCIE_REFCLKp0/1 PCIE_REFCLKn0/1	100	DIFF OUT	PCIE 两路参考差分时钟输出
LCD_CLK	150	O	DVO 显示时钟输出
SPI_CLK	50	O	SPI 总线时钟输出
SDIO_CLK	50	O	SDIO 总线时钟输出
GMAC_RX_CLK GMAC_TX_CLK	125	I I/O	GMAC 网络接收、发送时钟
AC97_BCLK_I HDA_BCLK_O	12 24	I O	AC97、HDA 数据流时钟

内部时钟	NODE_CLOCK	500~800	G	NODE 模块时钟，供处理器核、SCACHE、IODMA、L1-XBAR 模块使用
	DDR_CLOCK	400~600	G	DDR 控制器时钟，供 DDR3 控制器使用
	NETWORK_CLOCK	200~400	G	供 NETWORK 互联结构使用
	PRINT_CLOCK	100~300	G	供 PRINT 打印接口控制器使用
	HDA_CLOCK	24	G	供 HDA 接口数据流时钟使用
	PIX0/1_CLOCK	100~200	G	供两路 DVO 显示时钟使用
	GPU_CLOCK	200~300	G	供 GPU 功能模块使用
	GMAC_CLOCK	125	G	供 GMAC 功能模块使用
	SB_CLOCK	100~200	G	供 HDA、BOOT 及 CONFBUS 模块内部控制器逻辑使用
	SATA_CLOCK	100~200	G	供 SATA 功能模块使用
	USB_CLOCK	100~200	G	USB、USB3.0 功能模块控制器时钟
	APB_CLOCK	80~150	G	供 APB 各个设备使用

4 电源管理

本章对电源管理的主体结构进行简要介绍，寄存器和使用方法参考第 29 章。

4.1 电源管理模块介绍

- 龙芯 2K0500 电源管理模块提供系统功耗管理实现机制。
- 支持 Advanced Configuration and Power Interface, Version 4.0a(ACPI),提供相应的功耗管理功能。
- 系统休眠与唤醒，支持 ACPI S3（待机到内存），ACPI S4（待机到硬盘），ACPI S5（软关机），并且支持电源失效检测和自动系统恢复。支持多种唤醒方式(GMAC0，电源开关等）
- 支持 Dynamic Power Management（DPM），动态性能功耗控制，支持动态关闭 NODE（CORE+SCACHE）、GPU、PCIE、SATA、USB2/3.0 控制器电源。
- 支持 Dynamic Voltage Frequency Scaling（DVFS），处理器核 DVFS 控制，由片内小核 LA132 处理器核独立控制。
- 系统时钟控制，模块时钟门控，多种方式调节频率。
- 提供温度管理控制功能。支持 3 级报警机制。

4.2 电源级别

表 4.1 显示了系统支持的 ACPI 状态及其相关说明。

表4.1 ACPI状态说明

状态	描述
G0/S0	全部工作，该模式下系统全部工作。
G1/S1	暂不支持
G1/S3	Suspend to RAM(STR)，上下文保存到内存
G1/S4	Suspend to Disk(STD)，保存到硬盘，除唤醒电路全部掉电
G2/S5	Soft off，只有唤醒电路上电
G3	Mechanical off，所有供电失效

4.3 控制引脚说明

表 4.2 为电源管理部分的 IO 信号描述。

表 4.2 控制引脚说明

名称	类型	描述	供电
ACPI_SYSRESETn	I	系统复位	RSM3V3
RTC_RSMRSTn	I	复位 Resume域逻辑，该信号需在 resume 域上电稳定后保持一段时间有效（推荐>5ms）	RTC_VDD
RTC_RSTn	I	电池更换后，重启RTC逻辑	RTC_VDD
ACPI_PLTRSTn	O	对系统平台其它设备进行复位	RSM3V3
ACPI_PWROK	I	主供电电源上电稳定，如有多个供电，该信号表示最后一个电源稳定。	RSM3V3

ACPI_PWRBTNn	I	电源按钮	RSM3V3
ACPI_RINGn	I	Modem唤醒信号	RSM3V3
ACPI_WAKEn	I	PCIE 边带唤醒信号	RSM3V3
ACPI_PMEn	I	PCI 系统唤醒信号	RSM3V3
ACPI_BATLOWn	I	电池电量低	RSM3V3
ACPI_LID	I	显示器开关信号	RSM3V3
ACPI_PWRTYPE	I	识别电池供电和电源供电,1指示AC Power；0指示System Battery	RSM3V3
ACPI_SUSSTATn	O	指示系统将要进入低功耗状态	RSM3V3
ACPI_S3n	O	STR，待机到内存指示信号	RSM3V3
ACPI_S4n	O	STD，待机到硬盘指示信号	RSM3V3
ACPI_S5n	O	Soft off，	RSM3V3
ACPI_SLPLANn	O	以太网 PHY 休眠指示信号	RSM3V3

5 芯片配置与控制

5.1 芯片工作模式

龙芯 2K0500 可以作为独立的 SoC 也可以作为 PCI 的桥片使用。本节就对这几种模式分别进行介绍。

5.1.1 独立 SoC

龙芯 2K0500 有丰富外围接口，配合若干外围器件，可简便地搭建一个单芯片计算机系统，一个常见的应用如图 5-1 所示。

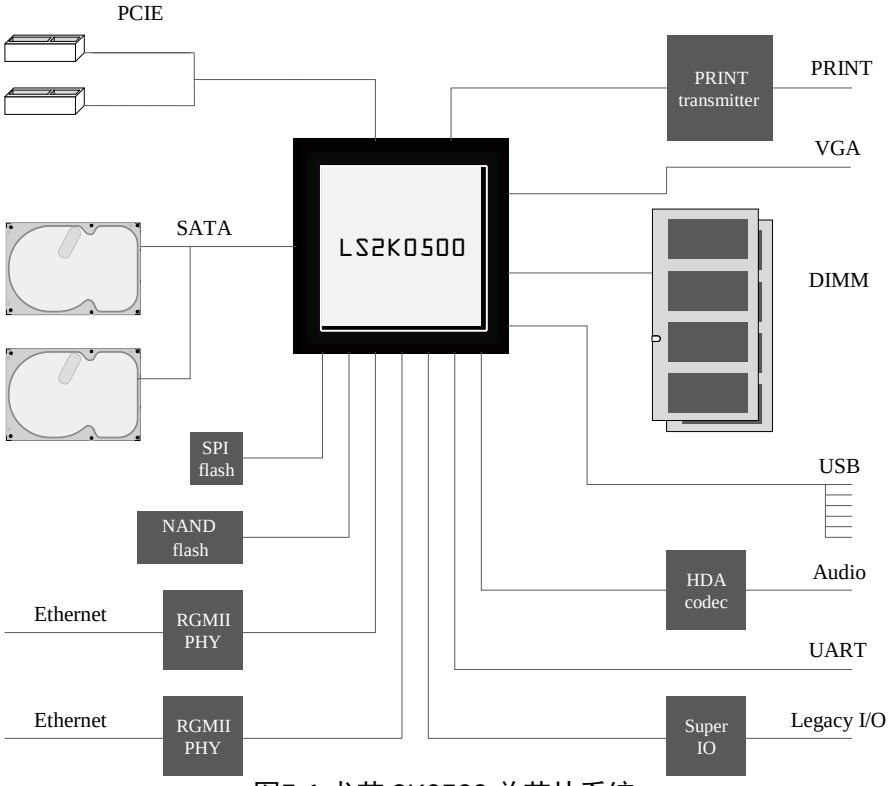


图5-1 龙芯 2K0500 单芯片系统

5.1.2 PCI 桥片

龙芯 2 号处理器使用 PCI 总线接口，可通过 2K0500 连接各种外设，形成两片系统。以龙芯 2 号芯片为例，图 5-2 给出一个搭建的样例。

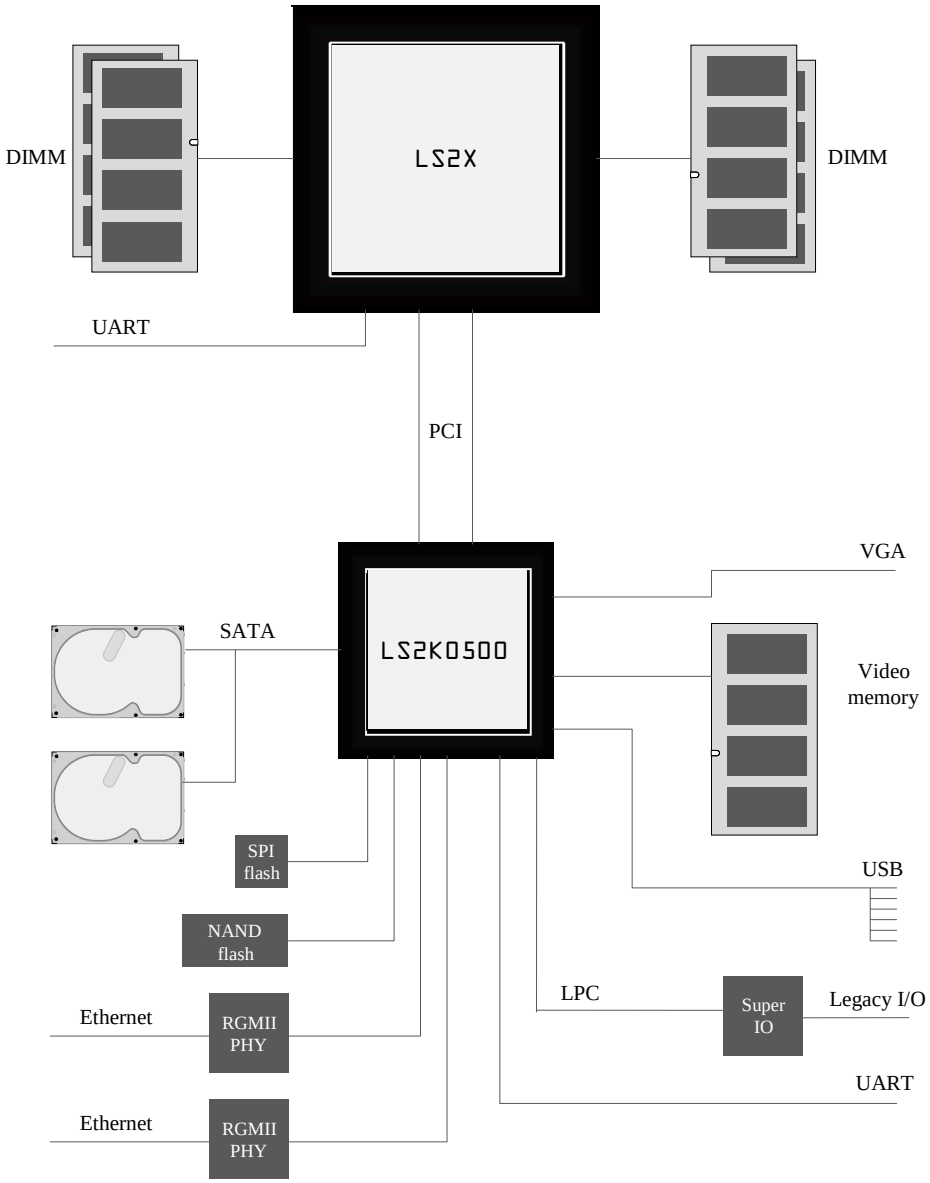


图5-2 龙芯 2 号处理器+2K0500 双芯片系统

在桥片模式下，龙芯 2K0500 桥片中断由 SYS_INTn0 和 SYS_INTn1 两引脚输出，送往 HOST 主芯片（龙芯 2 号处理器）。龙芯 2 号处理器需查询 2K0500 桥片的中断控制器得到详细的中断信息。

5.2 芯片初始化信号

龙芯 2K0500 初始化信号：复用芯片功能引脚，通过在系统复位期间采样外部上下拉的值得到配置信息。部分配置信息编码成 bootcfg，供软件判定上电状态。

表5-1 初始化配置信号

信号名称	bootcfg	描述

{NAND_RD, NAND_CLE, PWM3}	2:0	启动选择输入 x00=SPI x01=LPC x10=NAND 011=LIO 111=SDIO
{LCD_D[13], LCD_D[10]}	4:3	启动 NAND 类型选择 00=512Mb(page 512B) 01=1Gb(page 2KB) 10=16Gb(page 4KB) 11=128Gb(page 8KB)
LCD_D[0]	5	NAND ECC 功能使能输入, 1=enable 0=disable
LCD_D[5]	6	PCIE 端口 0 EP/RC 选择输入 0=RC 1=EP
LCD_D[20:19]	8:7	PLL 时钟配置输入 00=低频模式 01=高频模式 10=软件模式 11=bypass 模式
LCD_D[9]	9	PCIE 参考时钟选择输入 0=内部 100MHz 时钟 1=外部 100MHz 时钟
NAND_ALE	10	PCI 外部仲裁选择 0: 使用内部仲裁器 1: 使用外部仲裁器
LCD_D[16]	11	PCIE 端口 1 EP/RC 选择输入 0=RC 1=EP
NAND_WR	12	PCI 主模式选择（SOC 模式下 PCI 必须配置为主模式） 0: PCI 桥模式 1: PCI 主模式 两种模式的主要区别在于主模式下 PCI_RESETn 为输出，桥模式的为输入。当选择 PCIX 时，桥模式的控制器还会在复位后采样总线频率信息。
NAND_CE	13	PCIX 模式选择 0: PCI 模式 1: PCIX 模式

NAND_D[7:6]	15:14	PCIX 速度选择（PCI 模式时应为 0） 01: PCIX66 其它: 不可用

5.3 地址空间分配

龙芯 2K0500 的地址空间分为 CPU 和 DMA 两个视角：所有 CPU 可访问的设备编址在 CPU 地址空间上，谓之 CPU 视角；可通过 DMA 直接访问系统内存的主设备所见到的空间为 DMA 视角。表5-2 和表5-3 分别给出了这两个视角的具体定义。表格中未包含的地址空间均为系统保留，软件错误地访问保留空间将导致不可预知的后果。

CPU 可发生多种访问类型，包括字节(B)、半字(H)、字(W)、双字(D)、四字(Q)和块式(C)等。每个设备所支持的访问类型有限制，如果超出其范围同样会导致不可预知的后果。

表5-2 地址空间分配之 CPU 视角

地址空间(unmapped)	大小	功能	拓扑	访问
0x0000_0000 - 0x0fff_ffff	256M	DDR	DDR	BHWDQC
0x1000_0000 - 0x13ff_ffff	64M	PCI MEM0	PCI	BHW
0x1400_0000 - 0x15ff_ffff	32M	Reserved		
0x1600_0000 - 0x16ff_ffff	16M	PCIE IO	PCIE	B
0x1700_0000 - 0x170f_ffff	1M	PCI IO	PCI	B
0x1710_0000 - 0x1710_ffff	64K	PCI CFG		BHW
0x1711_0000 - 0x1711_00ff	256	PCI header		BHW
		Reserved		
0x1800_0000 - 0x19ff_ffff	32M	SPI0 MEM	CONF/BOOT	BHW
0x1a00_0000 - 0x1bff_ffff	32M	LIO		BHW
0x1c00_0000 - 0x1c0f_ffff	1M	boot		BHW
0x1c10_0000 - 0x1cff_ffff		Reserved		
0x1d00_0000 - 0x1dff_ffff	16M	LPC MEM		BHW
0x1e00_0000 - 0x1eff_ffff	16M	SPI1 MEM		BHW
0x1f00_0000 - 0x1f00_ffff	64K	GPU		W
0x1f01_0000 - 0x1f01_ffff	64K	DC		W
0x1f02_0000 - 0x1f02_ffff	64K	GMAC0		W
0x1f03_0000 - 0x1f03_ffff	64K	GMAC1		W
0x1f04_0000 - 0x1f04_ffff	64K	SATA		W
0x1f05_0000 - 0x1f05_ffff	64K	USB 2.0		W
0x1f06_0000 - 0x1f06_ffff	64K	USB 3.0		W
0x1f07_0000 - 0x1f07_ffff	64K	HDA		BHW
0x1f08_0000 - 0x1f0b_ffff	256K	OTG		W
0x1f0c_0000 - 0x1f0c_ffff	64K	PRINT		W
0x1f0d_0000 - 0x1f0e_ffff	128K	LPC IO/REG	CONF/BOOT	B
		Reserved		
0x1fd0_0000 - 0x1fd3_ffff	256K	SPI0 IO		B
0x1fd4_0000 - 0x1fd7_ffff	256K	SPI1 IO		B
0x1fe1_0000 - 0x1fe1_ffff	64k	confbus		BHW
		Reserved		
0x1ff0_0000 - 0x1fff_ffff	1M	APB		

地址空间(mapped)	大小	功能	拓扑	
0x4000_0000 - 0x7fff_ffff	1G	PCIE MEM	PCIE	BHW
0x2000_0000 - 0x23ff_ffff	64M	PCI MEM1		BHW
0x2400_0000 - 0x27ff_ffff	64M	PCI MEM2		BHW
0x8000_0000 - 0xffff_ffff	2G	DDR	DDR	BHWDQC
地址空间(APB)[19:12] Base 0x1ff0_0000	大小	功能	拓扑	
0x40000	16K	UART（10 路）	0~9(1KB*10)	B
0x44000	16K	CAN（4 路）	0~3(4KB*4)	B
0x48000	16K	I2C（6 路）	0~5(2KB*6)	B
0x4c000	16K	PS2		B
0x50000	16K	SPI（4 路）	0~3(4KB*4)	B
0x54000	16K	AC97		W
0x58000	16K	NAND		W
0x5c000	16K	PWM（16 路）	0~15(16B*16)	W
0x60000	16K	DPM		W
0x64000	16K	SDIO（2 路）	0~1(8KB*2)	W
0x68000	16K	HPET（4 个）	0~3(4KB*4)	W
0x6c000	16K	ACPI/RTC	ACPI:0x6c000 RTC:0x6c100	W

龙芯 2K0500 内部可发起 DMA 的主设备包括 GPU、DC、PCIE、USB、SATA、GMAC、HDA、PRINT、DMA 等，以上主设备请求可以访问路由到表5-3 中的所有空间。

表5-3 地址空间分配之 DMA 视角

地址空间(mapped)	大小	功能	拓扑	说明
0x0000_0000 - 0x7fff_ffff	2G	DDR	SoC 模式/ PCI 桥模式	同时设有 CACHE 访问配置位，可配置选择访问 DDR 或 CCIO，默认按照表中地址进行分配。
0x8000_0000 - 0xffff_ffff	2G	CCIO	SoC 模式	
0x8000_0000 - 0xffff_ffff	2G	PCI AM	PCI 桥模式	

5.4 时钟与复位控制

5.4.1 时钟配置概要

龙芯 2K0500 的片上时钟有硬件、软件两种配置模式。硬件配置模式下时钟生成完全不需要软件参与，但频率选择非常有限。软件配置模式下所有时钟相关的参数都可以改变，非常灵活，但在操作时需谨慎。本小节主要描述软件配置的流程。

在系统复位结束后，所有 PLL 输出均被旁路为参考时钟，分频器设置初始化为默认分频数，整个系统以最低速度运行。这时引导程序应当对片上的每个 PLL 进行以下操作：

1. 将对应的 PLL 的 PD 信号设置为 1；

2. 设置寄存器除了 sel_pll_*及 soft_set_pll 之外的其它寄存器，即这两个寄存器在设置的过程中写为 0；
3. 将对应的 PLL 的 PD 信号设置为 0；
4. 其他寄存器值不变，将 soft_set_pll 设置为 1；
5. 等待寄存器中的锁定信号 locked_*为 1；
6. 设置 sel_pll_*为 1，此时对应的时钟频率将切换为软件设置的频率。

如果后期有需要修改 PLL 参数，则要先切换时钟为参考时钟，然后按上述步骤再配一遍。

5.4.2 展频 PLL 配置

展频 PLL 概念性结构如图5-3 所示，输入时钟 Fin 经过输入分频器得到 Fref 送到倍频器，倍频器送出 Fvco，然后在输出前除以一个分频系数。展频 PLL 还带有一个展频控制器，能够用三角波对输出时钟的频率进行调制。

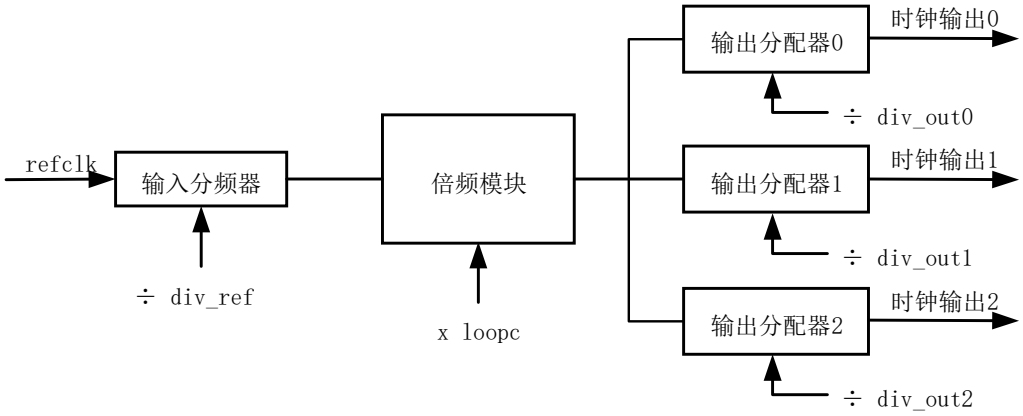


图5-3 展频 PLL 概念性结构

输出时钟频率的计算方式如下：clock_out = refclk / div_ref * loopc / divoutN；

其中，2K0500 的 refclk 固定为 100MHz，此外需要保证输入分频器的输出（refclk / div_ref）在 20 ~ 40MHz 的范围内，倍频模块倍频后的频率（refclk / div_ref * loopc）在 1.2GHz ~ 3.2GHz 的范围内。

PLL 相关的配置信号及说明见表 5-4。

表5-4 PLL 相关配置信号说明表

信号	位数	方向	说明
pll_div_out0	7	R/W	PLL 输出时钟 0 分频数
pll_div_out1	7	R/W	PLL 输出时钟 1 分频数
pll_div_out2	7	R/W	PLL 输出时钟 2 分频数
pll_loopc	9	R/W	PLL 倍频乘数
pll_div_ref	7	R/W	PLL 输入分频数
pll_locked	1	RO	PLL 锁定
sel_pll_out0	1	R/W	选择 PLL 输出时钟 0
sel_pll_out1	1	R/W	选择 PLL 输出时钟 1
sel_pll_out2	1	R/W	选择 PLL 输出时钟 2

set_pll_param	1	R/W	设置 PLL 配置参数
pll_bypass	1	R/W	PLL 内部 bypass
pll_pd	1	R/W	PLL powerdown

5.4.3 复位控制

龙芯 2K0500 内部有多种复位机制，用户可根据需要进行选择：

- 模块级复位：只针对一个模块发起，如 USB/ GPU 等。一般只在配置时钟完成，还未开始工作前进行。在工作后复位有使系统死锁的危险。
- 系统热复位：除芯片配置寄存器和时钟配置寄存器保持不变外，全系统复位。
- 系统冷复位：由看门狗或者电源管理模块发起，详见相关章节。

5.5 芯片配置寄存器

龙芯 2K0500 有大量的配置寄存器，多数分布于各个功能模块中，本节介绍芯片级的配置寄存器。

表5-5 芯片配置寄存器列表

地址	名称	描述
0x1fe10100	CHIP_CTRL0	芯片通用配置寄存器 0
0x1fe10104	CHIP_CTRL1	芯片通用配置寄存器 1
0x1fe10108	CHIP_CTRL2	芯片通用配置寄存器 2
0x1fe1010c	CHIP_CTRL3	芯片通用配置寄存器 3
0x1fe10110	CHIP_CTRL4	芯片通用配置寄存器 4
0x1fe10114	CHIP_CTRL5	芯片通用配置寄存器 5
0x1fe10120	CHIP_SAMP0	芯片采样参数寄存器 0
0x1fe10124	CHIP_SAMP1	芯片采样参数寄存器 1
0x1fe10128	CHIP_SAMP2	芯片采样参数寄存器 2
0x1fe1012c	CHIP_SAMP3	芯片采样参数寄存器 3
0x1fe10130	CHIP_HPT0	芯片计数寄存器低 32 位
0x1fe10134	CHIP_HPT1	芯片计数寄存器高 32 位
0x1fe10400	PLL_NODE_0	NODE 的 PLL 低 32 位配置
0x1fe10404	PLL_NODE_1	NODE 的 PLL 高 32 位配置
0x1fe10408	PLL_DDR_0	内存控制器的 PLL 低 32 位配置
0x1fe1040c	PLL_DDR_1	内存控制器的 PLL 高 32 位配置
0x1fe10410	PLL_SOC_0	SOC 的 PLL 低 32 位配置
0x1fe10414	PLL_SOC_1	SOC 的 PLL 高 32 位配置
0x1fe10418	PLL_PIX0_0	PIX0 的 PLL 低 32 位配置
0x1fe1041c	PLL_PIX0_1	PIX0 的 PLL 高 32 位配置
0x1fe10420	PLL_PIX1_0	PIX1 的 PLL 低 32 位配置

地址	名称	描述
0x1fe10424	PLL_PIX1_1	PIX1 的 PLL 高 32 位配置
0x1fe10428	FREQSCALE	设备时钟分频配置
0x1fe10430	GPIO0_OEN	GPIO0~31 位输出使能
0x1fe10434	GPIO1_OEN	GPIO32~63 位输出使能
0x1fe10438	GPIO0_IN	GPIO0~31 位输入值
0x1fe1043c	GPIO1_IN	GPIO32~63 位输入值
0x1fe10440	GPIO0_OUT	GPIO0~31 位输出值
0x1fe10444	GPIO1_OUT	GPIO32~63 位输出值
0x1fe10450	GPIO2_OEN	GPIO64~95 位输出使能
0x1fe10454	GPIO3_OEN	GPIO96~127 位输出使能
0x1fe10458	GPIO2_IN	GPIO64~95 位输入值
0x1fe1045c	GPIO3_IN	GPIO96~127 位输入值
0x1fe10460	GPIO2_OUT	GPIO64~95 位输出值
0x1fe10464	GPIO3_OUT	GPIO96~127 位输出值
0x1fe10470	GPIO4_OEN	GPIO128~154 位输出使能
0x1fe10478	GPIO4_IN	GPIO128~154 位输入值
0x1fe10480	GPIO4_OUT	GPIO128~154 位输出值
0x1fe10490	GPIO_CFG0	GPIO0~7 复用配置寄存器
0x1fe10494	GPIO_CFG1	GPIO8~15 复用配置寄存器
0x1fe10498	GPIO_CFG2	GPIO16~23 复用配置寄存器
0x1fe1049c	GPIO_CFG3	GPIO24~31 复用配置寄存器
0x1fe104a0	GPIO_CFG4	GPIO32~39 复用配置寄存器
0x1fe104a4	GPIO_CFG5	GPIO40~47 复用配置寄存器
0x1fe104a8	GPIO_CFG6	GPIO48~55 复用配置寄存器
0x1fe104ac	GPIO_CFG7	GPIO56~63 复用配置寄存器
0x1fe104b0	GPIO_CFG8	GPIO64~71 复用配置寄存器
0x1fe104b4	GPIO_CFG9	GPIO72~79 复用配置寄存器
0x1fe104b8	GPIO_CFG10	GPIO80~87 复用配置寄存器
0x1fe104bc	GPIO_CFG11	GPIO88~95 复用配置寄存器
0x1fe104c0	GPIO_CFG12	GPIO96~103 复用配置寄存器
0x1fe104c4	GPIO_CFG13	GPIO104~111 复用配置寄存器
0x1fe104c8	GPIO_CFG14	GPIO112~119 复用配置寄存器
0x1fe104cc	GPIO_CFG15	GPIO120~127 复用配置寄存器
0x1fe104d0	GPIO_CFG16	GPIO128~135 复用配置寄存器
0x1fe104d4	GPIO_CFG17	GPIO136~143 复用配置寄存器
0x1fe104d8	GPIO_CFG18	GPIO144~151 复用配置寄存器
0x1fe104dc	GPIO_CFG19	GPIO152~154 复用配置寄存器
0x1fe104e0	GPIO_INTEN0	GPIO0~31 中断输入使能寄存器
0x1fe104e4	GPIO_INTEN1	GPIO32~63 中断输入使能寄存器

地址	名称	描述
0x1fe104e8	GPIO_INTEN2	GPIO64~95 中断输入使能寄存器
0x1fe104ec	GPIO_INTEN3	GPIO96~127 中断输入使能寄存器
0x1fe10500	USB_PHY0	USB 的 PHY 配置寄存器 0
0x1fe10504	USB_PHY1	USB 的 PHY 配置寄存器 1
0x1fe10508	USB_PHY2	USB 的 PHY 配置寄存器 2
0x1fe1050c	USB_PHY3	USB 的 PHY 配置寄存器 3
0x1fe10550	PCIE_REG0	PCIE 的配置寄存器 0
0x1fe10554	PCIE_REG1	PCIE 的配置寄存器 1
0x1fe10558	PCIE_REG2	PCIE 的配置寄存器 2
0x1fe1055c	PCIE_REG3	PCIE 的配置寄存器 3
0x1fe10560	PCIEPHY0	PCIE 的 PHY 配置寄存器 0
0x1fe10564	PCIEPHY1	PCIE 的 PHY 配置寄存器 1
0x1fe10570	SATA0_REG0	SATA0 的配置寄存器 0
0x1fe10574	SATA0_REG1	SATA0 的配置寄存器 1
0x1fe10578	SATA1_REG0	SATA1 的配置寄存器 0
0x1fe1057c	SATA1_REG1	SATA1 的配置寄存器 1
0x1fe10580	SATA_BARCONF0	SATA 转换桥配置寄存器 0
0x1fe10584	SATA_BARCONF1	SATA 转换桥配置寄存器 1
0x1fe10590	SATA0_PHY0	SATA0 的 PHY 配置寄存器 0
0x1fe10594	SATA0_PHY1	SATA0 的 PHY 配置寄存器 1
0x1fe10598	SATA1_PHY0	SATA1 的 PHY 配置寄存器 0
0x1fe1059c	SATA1_PHY1	SATA1 的 PHY 配置寄存器 1
0x1fe10c00	CONFDMA0	DMA0 控制器的配置寄存器
0x1fe10c10	CONFDMA1	DMA1 控制器的配置寄存器
0x1fe10c20	CONFDMA2	DMA2 控制器的配置寄存器
0x1fe10c30	CONFDMA3	DMA3 控制器的配置寄存器
0x1fe11040	CORE_INTISR0	路由给 CORE 的低 32 位中断状态
0x1fe11044	INTISR0	低 32 位中断状态寄存器
0x1fe11048	CORE_INTISR1	路由给 CORE 的高 32 位中断状态
0x1fe1104c	INTISR1	高 32 位中断状态寄存器
0x1fe11400	ENTRY0_0	8 位中断路由寄存器[0--7]
0x1fe11408	ENTRY8_0	8 位中断路由寄存器[8--15]
0x1fe11410	ENTRY16_0	8 位中断路由寄存器[16--23]
0x1fe11418	ENTRY24_0	8 位中断路由寄存器[24--31]
0x1fe11420	INTISR_0	低 32 位中断状态寄存器

地址	名称	描述
0x1fe11424	INTIEN_0	低 32 位中断使能状态寄存器
0x1fe11428	INTSET_0	低 32 位设置使能寄存器
0x1fe1142c	INTCLR_0	低 32 位中断清除寄存器，清除使能寄存器和脉冲触发的中断
0x1fe11430	INTPOL_0	低 32 位极性设置寄存器(电平中断)
0x1fe11434	INTEDGE_0	低 32 位触发方式寄存器（1：脉冲触发；0：电平触发）
0x1fe11440	ENTRY0_1	8 位中断路由寄存器[32--39]
0x1fe11448	ENTRY8_1	8 位中断路由寄存器[40--47]
0x1fe11450	ENTRY16_1	8 位中断路由寄存器[48--55]
0x1fe11458	ENTRY24_1	8 位中断路由寄存器[56--63]
0x1fe11460	INTISR_1	高 32 位中断状态寄存器
0x1fe11464	INTIEN_1	高 32 位中断使能状态寄存器
0x1fe11468	INTSET_1	高 32 位设置使能寄存器
0x1fe1146c	INTCLR_1	高 32 位中断清除寄存器，清除使能寄存器和脉冲触发的中断
0x1fe11470	INTPOL_1	高 32 位极性设置寄存器(电平中断)
0x1fe11474	INTEDGE_1	高 32 位触发方式寄存器（1：脉冲触发；0：电平触发）
0x1fe11500	Thsens_int_ctrl_Hi0	温度传感器高温中断控制寄存器 0
0x1fe11504	Thsens_int_ctrl_Hi1	温度传感器高温中断控制寄存器 1
0x1fe11508	Thsens_int_ctrl_Lo0	温度传感器低温中断控制寄存器 0
0x1fe1150c	Thsens_int_ctrl_Lo1	温度传感器低温中断控制寄存器 1
0x1fe11510	Thsens_int_status0/clr0	温度传感器中断状态寄存器 0
0x1fe11514	Thsens_int_status1/clr1	温度传感器中断状态寄存器 1
0x1fe11520	Thsens_scale_hi0	温度传感器测量寄存器 0
0x1fe11524	Thsens_scale_hi1	温度传感器测量寄存器 1
0x1fe11528	Thsens_scale_lo0	保留
0x1fe1152c	Thsens_scale_lo1	保留
0x1fe13ff8	CHIP_CHIPID0	芯片识别号 0
0x1fe13ffc	CHIP_CHIPID1	芯片识别号 1

5.5.1 通用配置寄存器 0

通用配置寄存器0，包括 UART 模式的控制，以及 HDA、GMAC、PCIE、内存控制器、RTC 控制器及 LPC 控制器的配置等。

寄存器地址：0x1fe10100。

表5-6 通用配置寄存器 0

位域	名称	访问	缺省值	描述
31:30	usbclk_mode	RW	0x3	4个usb接口参考时钟模式选择： 00：保留； 01：保留； 10：USB0选择外部晶振时钟XO输入(时钟频率12MHz)，USB1~3选择由USB0输出时钟； 11：USB0选择内部PLL时钟产生(时钟频率50MHz)，USB1~3选择由USB0输出时钟；
29	lpc_slave_en	RW	0x0	LPC接口SLAVE模式使能： 1：LPC接口使能SLAVE；0：LPC接口关闭SLAVE。
28	Reserved	RO	0x0	-
27	extioint_enable	RW	0x0	扩展IO中断使能位，高电平有效
26	conf_rtc_restart	RW	0x0	内部振荡器重启控制
25:23	conf_rtc_ds	RW	0x0	内部振荡器接口控制信号
22:21	Reserved	RO	0x0	-
20:19	jtag_config	RW	0x0	JTAG接口复用模式配置： 00：LA264处理器JTAG接口； 01：LA132处理器JTAG接口； 10：LA264->LA132串行JTAG； 11：保留。
18	gmac_test_lpbk	RW	0x0	GMAC0~1接口loopback环回测试模式使能： 1：使能loopback环回测试模式； 0：关闭loopback环回测试模式。
17	gmac1_mii_sel	RW	0x0	GMAC1接口MII模式选择： 1：MII接口模式；0：RGMII接口模式
16	gmac0_mii_sel	RW	0x0	GMAC0接口MII模式选择： 1：MII接口模式；0：RGMII接口模式
15:14	sdio1_dma_sel	RW	0x0	SDIO1接口DMA复用选择方式： 0x00：DMA均不可用；0x01：复用DMA0； 0x10：复用DMA1；　0x11：复用DMA2。
13	pcie_clkdiv	RW	0x0	PCIE参考时钟二分频使能： 1：参考时钟使能二分频；0：参考时钟不分频。
12	ddr3_regs_default	RW	0x0	窗口不命中处理 0：关闭内存控制器的该功能 1：当所有窗口不命中时，由内存控制器给出响应，防止CPU卡死。
11	ddr3_regs_disable	RW	0x0	DDR配置空间关闭，高有效 DDR控制器在内存空间中开辟了一小段配置空间(1MB @0x0ff0,0000)，在关闭后软件就可以使用这段空间。为避免意外访问，建议在配置完成后及时关闭
10	hda_conf_cc	RW	0x0	HDA接口总线CACHE一致性使能： 1：使能CACHE一致性；0：关闭CACHE一致性。
9	hda_sel	RW	0x0	HDA选择AC97信号使能位： 1：HDA复用AC97信号引脚；0：HDA不复用AC97信号引脚。
8	hda_bclksel	RW	0x0	HDA信号时钟源端选择： 1：bclk选择usb接口48M时钟二分频产生； 0：bclk选择内部DDR-PLL时钟产生。
7:4	uart1_enable	RW	0x0	UART1对应的UART控制器模式及引脚复用关系(具体引脚复用分配参看芯片数据手册uart接口定义)：

				Bit0：对应uart1，保留； Bit1：为1对应uart7复用有效(若bit2为0则uart7复用为4线模式，若bit2为1则复用为2线模式)，为0对应uart1的8线模式，uart7复用无效； Bit2：为1对应uart8复用为2线模式，为0对应uart1的8线模式，uart8复用无效； Bit3：为1对应uart9复用为2线模式，为0对应uart1的8线模式，uart9复用无效。
3:0	uart0_enable	RW	0x0	UART0对应的UART控制器模式及引脚复用关系(具体引脚复用分配参看芯片数据手册uart接口定义)： Bit0：对应uart0，保留； Bit1：为1对应uart4复用有效(若bit2为0则uart4复用为4线模式，若bit2为1则复用为2线模式)，为0对应uart0的8线模式，uart4复用无效； Bit2：为1对应uart5复用为2线模式，为0对应uart0的8线模式，uart5复用无效； Bit3：为1对应uart6复用为2线模式，为0对应uart0的8线模式，uart6复用无效。

5.5.2 通用配置寄存器 1

通用配置寄存器1，包括对 USB、PCIE 的一致性及 LIO 控制器的配置等。

寄存器地址：0x1fe10104。

表5-7 通用配置寄存器 1

位域	名称	访问	缺省值	描述
31:29	Reserved	RO	0x0	-
28	lio_rom_width16	RW	0x0	ROM空间访问16位数据位宽配置位： 1：16位模式；0：8位模式。
27:23	lio_rom_count_init	RW	0x0	ROM空间访问数据读取延迟初始值,延迟范围1~32， 0：初值为31,；1：初值为30；...，31：初值为0。
22:21	lio_clk_period	RW	0x0	LIO总线访问数据读取延迟计数步长(时钟周期数)： 00：步长为1个周期；01：步长为4个周期； 10：步长为2个周期；11：步长为1个周期。
20:19	Reserved	RW	0x0	-
18	conf_pcie_cc	RW	0x0	PCIE接口总线CACHE一致性使能位： 1：使能CACHE一致性；0：关闭CACHE一致性。
17:10	conf_iodma_spare_rd	RW	0x0	iodma读操作最大数设置
9:6	conf_usb_flush_idle	RW	0x0	设置清空write buffer前空闲周期数
5	conf_usb_prefetch	RW	0x0	USB接口总线使能读预取
4	conf_usb_flush_wr	RW	0x0	USB接口总线设置写命令发出后是否清空read buffer
3	conf_usb_stop_waw	RW	0x0	USB接口总线是否允许在上一个写完成前发出写命令
2	conf_usb_stop_raw	RW	0x0	USB接口总线是否允许在上一个写完成前发出读命令
1	conf_usb_otg_sel	RW	0x0	USB0接口OTG模式配置位： 1：选择OTG模式；0：选择USB模式。
0	conf_usb_cc	RW	0x0	USB接口总线CACHE一致性使能位： 1：使能CACHE一致性；0：关闭CACHE一致性。

5.5.3 通用配置寄存器 2

通用配置寄存器2，包括各功能模块软复位的控制，以及模块 DMA 写请求按序执行使能位的配置等。

寄存器地址：0x1fe10108。

表5-8 通用配置寄存器 1

位域	名称	访问	缺省值	描述

31:30	Reserved	RO	0x0	-
29	apb_dma_order_en	RW	0x0	APB 内部互联读写请求按序执行使能位，高电平有效
28	print_order_en	RW	0x0	PRINT 内部互联读写请求按序执行使能位，高电平有效
27	hda_order_en	RW	0x0	HDA 内部互联读写请求按序执行使能位，高电平有效
26	usb3.0_order_en	RW	0x0	USB3内部互联读写请求按序执行使能位，高电平有效
25	usb2.0_order_en	RW	0x0	USB2内部互联读写请求按序执行使能位，高电平有效
24	sata_order_en	RW	0x0	SATA 内部互联读写请求按序执行使能位，高电平有效
23	gmac1_order_en	RW	0x0	GMAC1内部互联读写请求按序执行使能位，高电平有效
22	gmac0_order_en	RW	0x0	GMAC0内部互联读写请求按序执行使能位，高电平有效
21	dc_order_en	RW	0x0	DC 内部互联读写请求按序执行使能位，高电平有效
20	gpu_order_en	RW	0x0	GPU 内部互联读写请求按序执行使能位，高电平有效
19	pcie_order_en	RW	0x0	PCIE 内部互联读写请求按序执行使能位，高电平有效
18	pci_order_en	RW	0x0	PCI 内部互联读写请求按序执行使能位，高电平有效
17	ioregs_order_en	RW	0x0	IO 设备内部互联读写请求按序执行使能位，高电平有效
16	cpu_order_en	RW	0x0	CPU内部互联读写请求按序执行使能位，高电平有效
15	Reserved	RO	0x0	-
14	conf_rtc_timer_hspeed	RW	0x0	RTC计数器值快速访问使能配置位： 1：开启快速访问；0：关闭快速访问。
13	gpu_disable_ramcg	RW	0x0	GPU模块RAM时钟门控无效配置位
12	pcie_soft_rst	RW	0x0	PCIE模块软复位： 1：软复位有效；0：软复位解除。
11	sata_soft_rst	RW	0x0	SATA模块软复位： 1：软复位有效；0：软复位解除。
10	usb3_soft_rst	RW	0x0	USB3模块软复位： 1：软复位有效；0：软复位解除。
9	usb_soft_rst	RW	0x0	USB模块软复位： 1：软复位有效；0：软复位解除。
8	gpu_soft_rst	RW	0x0	GPU模块软复位： 1：软复位有效；0：软复位解除。
7:6	Reserved	RO	0x0	-
5	lpc_rom_8mbits	RW	0x1	LPC总线ROM访问8Mbits地址空间使能位 1：使能；0：关闭。
4:0	Reserved	RO	0x0	-

5.5.4 通用配置寄存器 3

通用配置寄存器3，包括对各功能模块 DMA 内部互联读写请求优先级的配置等。
寄存器地址：0x1fe1010c。

表5-9 通用配置寄存器3

位域	名称	访问	缺省值	描述
31:28	Reserved	RO	0x0	
27	apb_dma_read_upgrade	RW	0x0	APB 内部互联总线读请求优先级使能配置位（高电平有效，默认按序访问）
26	print_read_upgrade	RW	0x0	PRINT 内部互联总线读请求优先级使能配置位（高电平有效，默认按序访问）
25	hda_read_upgrade	RW	0x0	HDA 内部互联总线读请求优先级使能配置位（高电

				平有效，默认按序访问）
24	usb3.0_read_upgrade	RW	0x0	USB3内部互联总线读请求优先级使能配置位（高电平有效，默认按序访问）
23	usb2.0_read_upgrade	RW	0x0	USB2内部互联总线读请求优先级使能配置位（高电平有效，默认按序访问）
22	sata_read_upgrade	RW	0x0	SATA 内部互联总线读请求优先级使能配置位（高电平有效，默认按序访问）
21	gmac1_read_upgrade	RW	0x0	GMAC1内部互联总线读请求优先级使能配置位（高电平有效，默认按序访问）
20	gmac0_read_upgrade	RW	0x0	GMAC0内部互联总线读请求优先级使能配置位（高电平有效，默认按序访问）
19	dc_read_upgrade	RW	0x0	DC 内部互联总线读请求优先级使能配置位（高电平有效，默认按序访问）
18	gpu_read_upgrade	RW	0x0	GPU 内部互联总线读请求优先级使能配置位（高电平有效，默认按序访问）
17	pcie_read_upgrade	RW	0x0	PCIE 内部互联总线读请求优先级使能配置位（高电平有效，默认按序访问）
16	pci_read_upgrade	RW	0x0	PCI内部互联总线读请求优先级使能配置位（高电平有效，默认按序访问）
15:12	Reserved	RO	0x0	
11	apb_dma_write_upgrade	RW	0x0	APB 内部互联总线写请求优先级使能配置位（高电平有效，默认按序访问）
10	print_write_upgrade	RW	0x0	PRINT 内部互联总线写请求优先级使能配置位（高电平有效，默认按序访问）
9	hda_write_upgrade	RW	0x0	HDA 内部互联总线写请求优先级使能配置位（高电平有效，默认按序访问）
8	usb3.0_write_upgrade	RW	0x0	USB3内部互联总线写请求优先级使能配置位（高电平有效，默认按序访问）
7	usb2.0_write_upgrade	RW	0x0	USB2内部互联总线写请求优先级使能配置位（高电平有效，默认按序访问）
6	sata_write_upgrade	RW	0x0	SATA 内部互联总线写请求优先级使能配置位（高电平有效，默认按序访问）
5	gmac1_write_upgrade	RW	0x0	GMAC1内部互联总线写请求优先级使能配置位（高电平有效，默认按序访问）
4	gmac0_write_upgrade	RW	0x0	GMAC0内部互联总线写请求优先级使能配置位（高电平有效，默认按序访问）
3	dc_write_upgrade	RW	0x0	DC 内部互联总线写请求优先级使能配置位（高电平有效，默认按序访问）
2	gpu_write_upgrade	RW	0x0	GPU 内部互联总线写请求优先级使能配置位（高电平有效，默认按序访问）
1	pcie_write_upgrade	RW	0x0	PCIE内部互联总线写请求优先级使能配置位（高电平有效，默认按序访问）
0	pci_write_upgrade	RW	0x0	PCI内部互联总线写请求优先级使能配置位（高电平有效，默认按序访问）

5.5.5 通用配置寄存器 4

通用配置寄存器4，包括对各功能模块 DMA 内部互联总线 CACHE 一致性访问使能的配置等。

寄存器地址：0x1fe10110。

表5-10 通用配置寄存器4

位域	名称	访问	缺省值	描述
31	gpu_trans_en	RW	0x0	GPU内存读写地址高4位(Addr[31:28])映射窗口使能位： 0：地址映射窗口不使能； 1：地址映射窗口使能。

30:27	Reserved	RO	0x0	
28:25	gpu_trans_mask	RW	0x0	GPU内存读写地址映射屏蔽位(高电平有效)： 屏蔽位为高电平的有效位进行相应地址位(Addr[31:28])屏蔽处理。
24:21	gpu_trans_addr	RW	0x0	GPU内存读写地址重新映射地址，对应内存地址高4位(Addr[31:28])： addr_new[31:28]= ~trans_mask & addr[31:28] | trans_mask & trans_addr。
20:17	Reserved	RO	0x0	
16	io_coherent_enable	RW	0x0	IO设备内部互联CACHE访问使能位，： 1：使能设备CACHE访问配置有效，配置对应设备coherent位开启CACHE加速访问； 0：关闭设备CACHE访问配置，对应设备coherent位配置无效，此时各个设备可通过内部总线地址最高位(第32位)选择是否CACHE访问(1:开启，0:关闭)。
15:14	Reserved	RO	0x0	
13	dma_coherent	RW	0x0	DMA内部互联总线CACHE访问配置位(开启IO设备CACHE使能位后配置有效)： 1：开启CACHE加速访问；0：关闭CACHE加速访问
12	print_coherent	RW	0x0	PRINT内部互联总线CACHE访问配置位(开启IO设备CACHE使能位后配置有效)： 1：开启CACHE加速访问；0：关闭CACHE加速访问
11	hda_coherent	RW	0x0	HDA内部互联总线CACHE访问配置位(开启IO设备CACHE使能位后配置有效)： 1：开启CACHE加速访问；0：关闭CACHE加速访问
10	usb3_coherent	RW	0x0	USB3内部互联总线CACHE访问配置位(开启IO设备CACHE使能位后配置有效)： 1：开启CACHE加速访问；0：关闭CACHE加速访问
9	usb2_coherent	RW	0x0	USB2内部互联总线CACHE访问配置位(开启IO设备CACHE使能位后配置有效)： 1：开启CACHE加速访问；0：关闭CACHE加速访问
8	sata_coherent	RW	0x0	SATA内部互联总线CACHE访问配置位(开启IO设备CACHE使能位后配置有效)： 1：开启CACHE加速访问；0：关闭CACHE加速访问
7	gmac1_coherent	RW	0x0	GMAC1内部互联总线CACHE访问配置位(开启IO设备CACHE使能位后配置有效)： 1：开启CACHE加速访问；0：关闭CACHE加速访问
6	gmac0_coherent	RW	0x0	GMAC0内部互联总线CACHE访问配置位(开启IO设备CACHE使能位后配置有效)： 1：开启CACHE加速访问；0：关闭CACHE加速访问
5	dc_coherent	RW	0x0	DC内部互联总线CACHE访问配置位(开启IO设备CACHE使能位后配置有效)： 1：开启CACHE加速访问；0：关闭CACHE加速访问
4	gpu_coherent	RW	0x0	GPU内部互联总线CACHE访问配置位(开启IO设备CACHE使能位后配置有效)： 1：开启CACHE加速访问；0：关闭CACHE加速访问
3	pcie_coherent	RW	0x0	PCIE内部互联总线CACHE访问配置位(开启IO设备CACHE使能位后配置有效)： 1：开启CACHE加速访问；0：关闭CACHE加速访问
2	pci_coherent	RW	0x0	PCI内部互联总线CACHE访问配置位(开启IO设备CACHE使能位后配置有效)： 1：开启CACHE加速访问；0：关闭CACHE加速访问
1:0	Reserved	RO	0x0	

5.5.6 通用配置寄存器 5

通用配置寄存器5。

寄存器地址：0x1fe10114。

表5-11 通用配置寄存器5

位域	名称	访问	缺省值	描述
31:0	Reserved	RO	0x0	

5.5.7 芯片采样参数寄存器 0

芯片采样参数寄存器0，包括芯片启动配置相关信息。
寄存器地址：0x1fe10120。

表5-12 芯片采样参数寄存器0

位域	名称	访问	缺省值	描述
31:16	Reserved	RO	0x0	
15:14	pcix_speed	RO	0x0	PCIX模式下SPEED选择： 01：PCIX-66MHz模式；其他：保留。
13	pcix_mode	RO	0x0	PCI总线PCIX模式选择，高电平有效
12	pci_host_mode	RO	0x0	PCI总线主控、桥模式选择： 1：PCI主控模式，芯片为PCI主控端；； 0：PCI桥模式，芯片工作在PCI桥片模式下。
11	pcie1_ep	RO	0x0	PCIE1端口EP模式选择配置： 1：选择EP模式；0：选择RC模式。
10	pci_arb_ext	RO	0x0	PCI总线外部仲裁选择： 0：使用内部仲裁器；1：使用外部仲裁器。
9	pcie_refclk_sel	RO	0x0	PCIE参考时钟选择： 0：选择内部参考时钟；1：选择芯片PAD输入。
8:7	clk_sel	RO	0x0	芯片内部PLL输出时钟上电配置选择： 00：硬件低频时钟配置模式，PLL按照低频配置参数输出时钟(NODE:500M，DDR:480M，NET:320M)； 01：硬件高频时钟配置模式，PLL按照高频配置参数输出时钟(NODE:800M，DDR:600M，NET:400M)； 10：软件配置模式，PLL按照软件配置选择输出时钟； 11：硬件bypass模式，PLL输出时钟全部使用外部输入系统时钟。
6	pcie0_ep	RO	0x0	PCIE0端口EP模式选择配置： 1：选择EP模式；0：选择RC模式。
5	nand_boot_ecc	RO	0x0	选择NAND启动时，ECC开启使能配置： 1：ECC开启；0：ECC关闭。
4:3	nand_type	RO	0x0	NAND页大小配置： 00：512B；01：2KB；10：4KB；11：8KB。
2:0	boot_sel	RO	0x0	芯片启动选择方式： x00：SPI启动；x01：LPC启动；x10：NAND启动； 011：LIO启动；111：SDIO启动。

5.5.8 芯片采样参数寄存器 1~3

寄存器地址：0x1fe10124~0x1fe1012c。

表5-13 芯片采样参数寄存器1~3

位域	名称	访问	缺省值	描述
31:0	Reserved	RO	0x0	-

5.5.9 芯片高精度计数器 0

芯片计数寄存器0，64位高精度时钟计数器0~31位，工作频率为内部总线时钟频率。
寄存器地址：0x1fe10130。

表5-14 芯片高精度计数器0

位域	名称	访问	缺省值	描述

31:0	CHIP_HPT[31:0]	RW	0x0	64位高精度时钟计数器低32位

5.5.10 芯片高精度计数器 1

芯片计数寄存器1，64位高精度时钟计数器32~63位，工作频率为内部总线时钟频率。
寄存器地址：0x1fe10134。

表5-15 芯片高精度计数器1

位域	名称	访问	缺省值	描述
31:0	CHIP_HPT[63:32]	RW	0x0	64位高精度时钟计数器高32位

5.5.11 NODE PLL 时钟配置寄存器 0

NODE PLL 时钟配置寄存器0，用于 NODE PLL 时钟参数配置。
寄存器地址：0x1fe10400。

表5-16 NODE-PLL 时钟配置寄存器0

位域	名称	访问	缺省值	描述
31:30	Reserved	RO	0x0	
29:24	odiv_node	RW	0x0	NODE PLL分频系数配置：0~63
23:16	div_loopc	RW	0x0	PLL倍频系数：0~255
13:8	div_refc	RW	0x0	PLL参考时钟分频系数：0~63
7	pll_locked	RO	0x0	PLL锁定标志，1代表锁定
6	Reserved	RO	0x0	-
5	pd_pll	RW	0x0	PLL关电控制，1代表关电
4	bypass	RW	0x0	PLL时钟bypass控制，1代表bypass
3	pll_soft_set	RW	0x0	允许软件设置PLL，1代表允许软件配置
2:1	Reserved	RO	0x0	-
0	pll_sel_node	RW	0x0	NODE选择PLL时钟输出配置，1代表选择PLL时钟输出

5.5.12 NODE PLL 时钟配置寄存器 1

NODE PLL 时钟配置寄存器1。
寄存器地址：0x1fe10404。

表5-17 NODE-PLL 时钟配置寄存器1

位域	名称	访问	缺省值	描述
31:0	Reserved	RO	0x0	-

5.5.13 DDR PLL 时钟配置寄存器 0

DDR PLL 时钟配置寄存器0，用于 DDR PLL 时钟参数配置。
寄存器地址：0x1fe10408。

表5-18 DDR-PLL 时钟配置寄存器0

位域	名称	访问	缺省值	描述
31:30	Reserved	RO	0x0	-
29:24	odiv_ddr	RW	0x0	DDR PLL分频系数配置：0~63
23:16	div_loopc	RW	0x0	PLL倍频系数：0~255
13:8	div_refc	RW	0x0	PLL参考时钟分频系数：0~63
7	pll_locked	RO	0x0	PLL锁定标志，1代表锁定
6	Reserved	RO	0x0	-
5	pd_pll	RW	0x0	PLL关电控制，1代表关电
4	bypass	RW	0x0	PLL时钟bypass控制，1代表bypass
3	pll_soft_set	RW	0x0	允许软件设置PLL，1代表允许软件配置
2	pll_sel_hda	RW	0x0	HDA选择PLL时钟输出配置，1代表选择PLL时钟输出
1	pll_sel_network	RW	0x0	NETWORK选择PLL时钟输出配置，1代表选择PLL时钟

				输出
0	pll_sel_ddr	RW	0x0	DDR选择PLL时钟输出配置，1代表选择PLL时钟输出

5.5.14 DDR PLL 时钟配置寄存器 1

DDR PLL 时钟配置寄存器1，用于 DDR PLL 时钟参数配置。
寄存器地址：0x1fe1040c。

表5-19 DDR-PLL 时钟配置寄存器1

位域	名称	访问	缺省值	描述
31:14	Reserved	RO	0x0	-
13:8	odiv_hda	RW	0x0	HDA PLL分频系数配置：0~63
7:6	Reserved	RO	0x0	-
5:0	odiv_network	RW	0x0	NETWORK PLL分频系数配置：0~63

5.5.15 SOC PLL 时钟配置寄存器 0

SOC PLL 时钟配置寄存器0，用于 SOC PLL 时钟参数配置。
寄存器地址：0x1fe10410。

表5-20 SOC-PLL 时钟配置寄存器0

位域	名称	访问	缺省值	描述
31:30	Reserved	RO	0x0	-
29:24	odiv_gpu	RW	0x0	GPU PLL分频系数配置：0~63
23:16	div_loopc	RW	0x0	PLL倍频系数：0~255
13:8	div_refc	RW	0x0	PLL参考时钟分频系数：0~63
7	pll_locked	RO	0x0	PLL锁定标志，1代表锁定
6	Reserved	RO	0x0	-
5	pd_pll	RW	0x0	PLL关电控制，1代表关电
4	bypass	RW	0x0	PLL时钟bypass控制，1代表bypass
3	pll_soft_set	RW	0x0	允许软件设置PLL，1代表允许软件配置
2	pll_sel_gmac	RW	0x0	GMAC选择PLL时钟输出配置，1代表选择PLL时钟输出
1	pll_sel_sb	RW	0x0	SB选择PLL时钟输出配置，1代表选择PLL时钟输出
0	pll_sel_gpu	RW	0x0	GPU选择PLL时钟输出配置，1代表选择PLL时钟输出

5.5.16 SOC PLL 时钟配置寄存器 1

SOC PLL 时钟配置寄存器1，用于 SOC PLL 时钟参数配置。
寄存器地址：0x1fe10414。

表5-21 SOC-PLL 时钟配置寄存器1

位域	名称	访问	缺省值	描述
31:14	Reserved	RO	0x0	-
13:8	odiv_gmac	RW	0x0	GMAC PLL分频系数配置：0~63
7:6	Reserved	RO	0x0	-
5:0	odiv_sb	RW	0x0	SB PLL分频系数配置：0~63

5.5.17 PIX0 PLL 时钟配置寄存器 0

PIX0 PLL 时钟配置寄存器0，用于 PIX0 PLL 时钟参数配置。
寄存器地址：0x1fe10418。

表5-22 PIX0 -PLL 时钟配置寄存器0

位域	名称	访问	缺省值	描述
31:30	Reserved	RO	0x0	-
29:24	odiv_pix0	RW	0x0	PIX0 PLL分频系数配置：0~63
23:16	div_loopc	RW	0x0	PLL倍频系数：0~255

13:8	div_refc	RW	0x0	PLL参考时钟分频系数：0~63
7	pll_locked	RO	0x0	PLL锁定标志，1代表锁定
6	Reserved	RO	0x0	-
5	pd_pll	RW	0x0	PLL关电控制，1代表关电
4	bypass	RW	0x0	PLL时钟bypass控制，1代表bypass
3	pll_soft_set	RW	0x0	允许软件设置PLL，1代表允许软件配置
2:1	Reserved	RO	0x0	-
0	pll_sel_pix0	RW	0x0	PIX0选择PLL时钟输出配置，1代表选择PLL时钟输出

5.5.18 PIX0 PLL 时钟配置寄存器 1

PIX0 PLL 时钟配置寄存器1，用于 PIX0 PLL 时钟参数配置。

寄存器地址：0x1fe1041c

表5-23 PIX0 -PLL 时钟配置寄存器1

位域	名称	访问	缺省值	描述
31:0	Reserved	RO	0x0	-

5.5.19 PIX1 PLL 时钟配置寄存器 0

PIX1 PLL 时钟配置寄存器0，用于 PIX1 PLL 时钟参数配置。

寄存器地址：0x1fe10420。

表5-24 PIX1 -PLL 时钟配置寄存器0

位域	名称	访问	缺省值	描述
31:30	Reserved	RO	0x0	-
29:24	odiv_pix1	RW	0x0	PIX1 PLL分频系数配置：0~63
23:16	div_loopc	RW	0x0	PLL倍频系数：0~255
13:8	div_refc	RW	0x0	PLL参考时钟分频系数：0~63
7	pll_locked	RO	0x0	PLL锁定标志，1代表锁定
6	Reserved	RO	0x0	-
5	pd_pll	RW	0x0	PLL关电控制，1代表关电
4	bypass	RW	0x0	PLL时钟bypass控制，1代表bypass
3	pll_soft_set	RW	0x0	允许软件设置PLL，1代表允许软件配置
2:1	Reserved	RO	0x0	-
0	pll_sel_pix1	RW	0x0	PIX1选择PLL时钟输出配置，1代表选择PLL时钟输出

5.5.20 PIX1 PLL 时钟配置寄存器 1

PIX1 PLL 时钟配置寄存器1，用于 PIX1 PLL 时钟参数配置。

寄存器地址：0x1fe10424。

表5-25 PIX1 -PLL 时钟配置寄存器1

位域	名称	访问	缺省值	描述
31:0	Reserved	RO	0x0	-

5.5.21 设备时钟分频配置寄存器

设备时钟分频配置寄存器，分频计算公式为：fout=fin*(freqscale+1)/8，lsu_freqdiv 分频系数除外。

寄存器地址：0x1fe10428。

表5-26 设备时钟分频配置寄存器

位域	名称	访问	缺省值	描述
31:27	lsu_freqdiv	RW	0x1f	LSU机芯模块时钟输出分频系数：0~31(0为不分频)，该分频系数为LSU机芯模块时钟第二级分频配置，基准时钟为DDR-PLL中NETWORK输出时钟经第一级分频

				(print_freqscale)后输出时钟
26:24	print_freqscale	RW	0x7	PRINT时钟输出分频系数：0~7，该分频系数为LSU机芯模块时钟第一级分频配置，其基准时钟为DDR-PLL中NETWORK输出时钟
23	Reserved	RO	0x0	-
22:20	apb_freqscale	RW	0x7	APB时钟输出分频系数：0~7
19	Reserved	RO	0x0	-
18:16	usb_freqscale	RW	0x7	USB时钟输出分频系数：0~7
15	Reserved	RO	0x0	-
14:12	sata_freqscale	RW	0x7	SATA时钟输出分频系数：0~7
11	Reserved	RO	0x0	-
10:8	sb_freqscale	RW	0x7	SB时钟输出分频系数：0~7
7	Reserved	RO	0x0	-
6:4	gpu_freqscale	RW	0x7	GPU时钟输出分频系数：0~7
3	Reserved	RO	0x0	-
2:0	node_freqscale	RW	0x7	NODE时钟输出分频系数：0~7

5.5.22 GPIO0~31 输出使能寄存器

GPIO0~31输出使能寄存器。
寄存器地址：0x1fe10430。

表5-27 GPIO0~31输出使能寄存器

位域	名称	访问	缺省值	描述
31:0	GPIO_OEN[31:0]	RW	0xffffffff	对应GPIO0~31的方向控制： 0：输出；1：输入

5.5.23 GPIO32~63 输出使能寄存器

GPIO32~63输出使能寄存器。
寄存器地址：0x1fe10434。

表5-28 GPIO32~63输出使能寄存器

位域	名称	访问	缺省值	描述
31:0	GPIO_OEN[63:32]	RW	0xffffffff	对应GPIO32~63的方向控制： 0：输出；1：输入

5.5.24 GPIO0~31 输入值寄存器

GPIO0~31输入值寄存器。
寄存器地址：0x1fe10438。

表5-29 GPIO0~31输入值寄存器

位域	名称	访问	缺省值	描述
31:0	GPIO_IN[31:0]	RW	0x0	对应GPIO0~31的输入值

5.5.25 GPIO32~63 输入值寄存器

GPIO32~63输入值寄存器。
寄存器地址：0x1fe1043c。

表5-30 GPIO32~63输入值寄存器

位域	名称	访问	缺省值	描述
31:0	GPIO_IN[63:32]	RW	0x0	对应GPIO31~63的输入值

5.5.26 GPIO0~31 输出值寄存器

GPIO0~31输出值寄存器。
寄存器地址：0x1fe10440。

表5-31 GPIO0~31输出值寄存器

位域	名称	访问	缺省值	描述
31:0	GPIO_OUT[31:0]	RW	0x0	对应GPIO0~31的输出值

5.5.27 GPIO32~63 输出值寄存器

GPIO32~63输出值寄存器。
寄存器地址：0x1fe10444。

表5-32 GPIO32~63输出值寄存器

位域	名称	访问	缺省值	描述
31:0	GPIO_OUT[63:32]	RW	0x0	对应GPIO31~63的输出值

5.5.28 GPIO64~95 输出使能寄存器

GPIO64~95输出使能寄存器。
寄存器地址：0x1fe10450。

表5-33 GPIO64~95输出使能寄存器

位域	名称	访问	缺省值	描述
31:0	GPIO_OEN[95:64]	RW	0xffffffff	对应GPIO64~95的方向控制： 0：输出；1：输入

5.5.29 GPIO96~127 输出使能寄存器

GPIO96~127输出使能寄存器。
寄存器地址：0x1fe10454。

表5-34 GPIO96~127输出使能寄存器

位域	名称	访问	缺省值	描述
31:0	GPIO_OEN[127:96]	RW	0xffffffff	对应GPIO96~127的方向控制： 0：输出；1：输入

5.5.30 GPIO64~95 输入值寄存器

GPIO64~95输入值寄存器。
寄存器地址：0x1fe10458。

表5-35 GPIO64~95输入值寄存器

位域	名称	访问	缺省值	描述
31:0	GPIO_IN[95:64]	RW	0x0	对应GPIO64~95的输入值

5.5.31 GPIO96~127 输入值寄存器

GPIO96~127输入值寄存器。
寄存器地址：0x1fe1045c。

表5-36 GPIO96~127输入值寄存器

位域	名称	访问	缺省值	描述
31:0	GPIO_IN[127:96]	RW	0x0	对应GPIO96~127的输入值

5.5.32 GPIO64~95 输出值寄存器

GPIO64~95输出值寄存器。
寄存器地址：0x1fe10460。

表5-37 GPIO64~95输出值寄存器

位域	名称	访问	缺省值	描述
31:0	GPIO_OUT[95:64]	RW	0x0	对应GPIO64~95的输出值

5.5.33 GPIO96~127 输出值寄存器

GPIO96~127输出值寄存器。
寄存器地址：0x1fe10464。

表5-38 GPIO96~127输出值寄存器

位域	名称	访问	缺省值	描述
31:0	GPIO_OUT[127:96]	RW	0x0	对应GPIO96~127的输出值

5.5.34 GPIO128~154 输出使能寄存器

GPIO128~154输出使能寄存器。
寄存器地址：0x1fe10470。

表5-39 GPIO128~154输出使能寄存器

位域	名称	访问	缺省值	描述
31:27	Reserved	RO	0x0	
26:0	GPIO_OEN[154:128]	RW	0xffffffff	对应GPIO128~154的方向控制： 0：输出；1：输入

5.5.35 GPIO160~191 输出使能寄存器

GPIO160~191输出使能寄存器。
寄存器地址：0x1fe10474。

表5-40 GPIO160~191输出使能寄存器

位域	名称	访问	缺省值	描述
31:0	Reserved	RO	0x0	

5.5.36 GPIO128~154 输入值寄存器

GPIO128~154输入值寄存器。
寄存器地址：0x1fe10478。

表5-41 GPIO128~154输入值寄存器

位域	名称	访问	缺省值	描述
31:27	Reserved	RO	0x0	
26:0	GPIO_IN[154:128]	RW	0x0	对应GPIO128~154的输入值

5.5.37 GPIO160~191 输入值寄存器

GPIO160~191输入值寄存器。
寄存器地址：0x1fe1047c。

表5-42 GPIO160~191输入值寄存器

位域	名称	访问	缺省值	描述
31:0	Reserved	RO	0x0	

5.5.38 GPIO128~159 输出值寄存器

GPIO128~159输出值寄存器。
寄存器地址：0x1fe10480。

表5-43 GPIO128~159输出值寄存器

位域	名称	访问	缺省值	描述
31:27	Reserved	RO	0x0	
26:0	GPIO_OUT[154:128]	RW	0x0	对应GPIO128~159的输出值

5.5.39 GPIO160~191 输出值寄存器

GPIO160~191输出值寄存器。
寄存器地址：0x1fe10484。

表5-44 GPIO160~191输出值寄存器

位域	名称	访问	缺省值	描述
31:0	Reserved	RO	0x0	

5.5.40 GPIO0~7 复用配置寄存器

GPIO0~7复用配置寄存器。
寄存器地址：0x1fe10490。

表5-45 GPIO0~7复用配置寄存器

位域	名称	访问	缺省值	描述
31	Reserved	RO	0x0	-
30:28	GPIO7_MUX	RW	0x0	GPIO7引脚复用配置： 000：复用为GPIO7；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
27	Reserved	RO	0x0	-
26:24	GPIO6_MUX	RW	0x0	GPIO6引脚复用配置： 000：复用为GPIO6；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
23	Reserved	RO	0x0	-
22:20	GPIO5_MUX	RW	0x0	GPIO5引脚复用配置： 000：复用为GPIO5；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
19	Reserved	RO	0x0	-
18:16	GPIO4_MUX	RW	0x0	GPIO4引脚复用配置： 000：复用为GPIO4；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
15	Reserved	RO	0x0	-
14:12	GPIO3_MUX	RW	0x0	GPIO3引脚复用配置： 000：复用为GPIO3；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
11	Reserved	RO	0x0	-
10:8	GPIO2_MUX	RW	0x0	GPIO2引脚复用配置： 000：复用为GPIO2；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
7	Reserved	RO	0x0	-
6:4	GPIO1_MUX	RW	0x0	GPIO1引脚复用配置：

				000：复用为GPIO1；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
3	Reserved	RO	0x0	-
2:0	GPIO0_MUX	RW	0x0	GPIO0引脚复用配置： 000：复用为GPIO0；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。

5.5.41 GPIO8~15 复用配置寄存器

GPIO8~15复用配置寄存器。
寄存器地址：0x1fe10494。

表5-46 GPIO8~15复用配置寄存器

位域	名称	访问	缺省值	描述
31	Reserved	RO	0x0	-
30:28	GPIO15_MUX	RW	0x0	GPIO17引脚复用配置： 000：复用为GPIO15；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
27	Reserved	RO	0x0	-
26:24	GPIO14_MUX	RW	0x0	GPIO14引脚复用配置： 000：复用为GPIO14；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
23	Reserved	RO	0x0	-
22:20	GPIO13_MUX	RW	0x0	GPIO13引脚复用配置： 000：复用为GPIO13；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
19	Reserved	RO	0x0	-
18:16	GPIO12_MUX	RW	0x0	GPIO12引脚复用配置： 000：复用为GPIO12；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
15	Reserved	RO	0x0	-
14:12	GPIO11_MUX	RW	0x0	GPIO11引脚复用配置： 000：复用为GPIO11；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
11	Reserved	RO	0x0	-
10:8	GPIO10_MUX	RW	0x0	GPIO10引脚复用配置： 000：复用为GPIO10；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
7	Reserved	RO	0x0	-
6:4	GPIO9_MUX	RW	0x0	GPIO9引脚复用配置： 000：复用为GPIO9；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
3	Reserved	RO	0x0	-
2:0	GPIO8_MUX	RW	0x0	GPIO8引脚复用配置： 000：复用为GPIO8；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。

5.5.42 GPIO16~23 复用配置寄存器

GPIO16~23复用配置寄存器。
寄存器地址：0x1fe10498。

表5-47 GPIO16~23复用配置寄存器

位域	名称	访问	缺省值	描述
31	Reserved	RO	0x0	-
30:28	GPIO23_MUX	RW	0x0	GPIO23引脚复用配置： 000：复用为GPIO23；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
27	Reserved	RO	0x0	-
26:24	GPIO22_MUX	RW	0x0	GPIO22引脚复用配置： 000：复用为GPIO22；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
23	Reserved	RO	0x0	-
22:20	GPIO21_MUX	RW	0x0	GPIO21引脚复用配置： 000：复用为GPIO21；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
19	Reserved	RO	0x0	-
18:16	GPIO20_MUX	RW	0x0	GPIO20引脚复用配置： 000：复用为GPIO20；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
15	Reserved	RO	0x0	-
14:12	GPIO19_MUX	RW	0x0	GPIO19引脚复用配置： 000：复用为GPIO19；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
11	Reserved	RO	0x0	-
10:8	GPIO18_MUX	RW	0x0	GPIO18引脚复用配置： 000：复用为GPIO18；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
7	Reserved	RO	0x0	-
6:4	GPIO17_MUX	RW	0x0	GPIO17引脚复用配置： 000：复用为GPIO17；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
3	Reserved	RO	0x0	-
2:0	GPIO16_MUX	RW	0x0	GPIO16引脚复用配置： 000：复用为GPIO16；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。

5.5.43 GPIO24~31 复用配置寄存器

GPIO24~31复用配置寄存器。
寄存器地址：0x1fe1049c。

表5-48 GPIO24~31复用配置寄存器

位域	名称	访问	缺省值	描述
31	Reserved	RO	0x0	-
30:28	GPIO31_MUX	RW	0x0	GPIO31引脚复用配置： 000：复用为GPIO31；001：第一复用；

				010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
27	Reserved	RO	0x0	-
26:24	GPIO30_MUX	RW	0x0	GPIO30引脚复用配置： 000：复用为GPIO30；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
23	Reserved	RO	0x0	-
22:20	GPIO29_MUX	RW	0x0	GPIO29引脚复用配置： 000：复用为GPIO29；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
19	Reserved	RO	0x0	-
18:16	GPIO28_MUX	RW	0x0	GPIO28引脚复用配置： 000：复用为GPIO28；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
15	Reserved	RO	0x0	-
14:12	GPIO27_MUX	RW	0x0	GPIO27引脚复用配置： 000：复用为GPIO27；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
11	Reserved	RO	0x0	-
10:8	GPIO26_MUX	RW	0x0	GPIO26引脚复用配置： 000：复用为GPIO26；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
7	Reserved	RO	0x0	-
6:4	GPIO25_MUX	RW	0x0	GPIO25引脚复用配置： 000：复用为GPIO25；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
3	Reserved	RO	0x0	-
2:0	GPIO24_MUX	RW	0x0	GPIO24引脚复用配置： 000：复用为GPIO24；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。

5.5.44 GPIO32~39 复用配置寄存器

GPIO32~39复用配置寄存器。

寄存器地址：0x1fe104a0。

表5-49 GPIO32~39复用配置寄存器

位域	名称	访问	缺省值	描述
31	Reserved	RO	0x0	-
30:28	GPIO39_MUX	RW	0x0	GPIO39引脚复用配置： 000：复用为GPIO39；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
27	Reserved	RO	0x0	-
26:24	GPIO38_MUX	RW	0x0	GPIO38引脚复用配置： 000：复用为GPIO38；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
23	Reserved	RO	0x0	-
22:20	GPIO37_MUX	RW	0x0	GPIO37引脚复用配置： 000：复用为GPIO37；001：第一复用；

				010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
19	Reserved	RO	0x0	-
18:16	GPIO36_MUX	RW	0x0	GPIO36引脚复用配置： 000：复用为GPIO36；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
15	Reserved	RO	0x0	-
14:12	GPIO35_MUX	RW	0x0	GPIO35引脚复用配置： 000：复用为GPIO35；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
11	Reserved	RO	0x0	-
10:8	GPIO34_MUX	RW	0x0	GPIO34引脚复用配置： 000：复用为GPIO34；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
7	Reserved	RO	0x0	-
6:4	GPIO33_MUX	RW	0x0	GPIO33引脚复用配置： 000：复用为GPIO33；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
3	Reserved	RO	0x0	-
2:0	GPIO32_MUX	RW	0x0	GPIO32引脚复用配置： 000：复用为GPIO32；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。

5.5.45 GPIO40~47 复用配置寄存器

GPIO40~47复用配置寄存器。

寄存器地址：0x1fe104a4。

表5-50 GPIO40~47复用配置寄存器

位域	名称	访问	缺省值	描述
31	Reserved	RO	0x0	-
30:28	GPIO47_MUX	RW	0x0	GPIO47引脚复用配置： 000：复用为GPIO47；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
27	Reserved	RO	0x0	-
26:24	GPIO46_MUX	RW	0x0	GPIO46引脚复用配置： 000：复用为GPIO46；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
23	Reserved	RO	0x0	-
22:20	GPIO45_MUX	RW	0x0	GPIO45引脚复用配置： 000：复用为GPIO45；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
19	Reserved	RO	0x0	-
18:16	GPIO44_MUX	RW	0x0	GPIO44引脚复用配置： 000：复用为GPIO44；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
15	Reserved	RO	0x0	-
14:12	GPIO43_MUX	RW	0x0	GPIO43引脚复用配置： 000：复用为GPIO43；001：第一复用；

				010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
11	Reserved	RO	0x0	-
10:8	GPIO42_MUX	RW	0x0	GPIO42引脚复用配置： 000：复用为GPIO42；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
7	Reserved	RO	0x0	-
6:4	GPIO41_MUX	RW	0x0	GPIO41引脚复用配置： 000：复用为GPIO41；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
3	Reserved	RO	0x0	-
2:0	GPIO40_MUX	RW	0x0	GPIO40引脚复用配置： 000：复用为GPIO40；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。

5.5.46 GPIO48~55 复用配置寄存器

GPIO48~55复用配置寄存器。

寄存器地址：0x1fe104a8。

表5-51 GPIO48~55复用配置寄存器

位域	名称	访问	缺省值	描述
31	Reserved	RO	0x0	-
30:28	GPIO55_MUX	RW	0x0	GPIO55引脚复用配置： 000：复用为GPIO55；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
27	Reserved	RO	0x0	-
26:24	GPIO54_MUX	RW	0x0	GPIO54引脚复用配置： 000：复用为GPIO54；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
23	Reserved	RO	0x0	-
22:20	GPIO53_MUX	RW	0x0	GPIO53引脚复用配置： 000：复用为GPIO53；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
19	Reserved	RO	0x0	-
18:16	GPIO52_MUX	RW	0x0	GPIO52引脚复用配置： 000：复用为GPIO52；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
15	Reserved	RO	0x0	-
14:12	GPIO51_MUX	RW	0x0	GPIO51引脚复用配置： 000：复用为GPIO51；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
11	Reserved	RO	0x0	-
10:8	GPIO50_MUX	RW	0x0	GPIO50引脚复用配置： 000：复用为GPIO50；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
7	Reserved	RO	0x0	-
6:4	GPIO49_MUX	RW	0x0	GPIO49引脚复用配置： 000：复用为GPIO49；001：第一复用；

				010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
3	Reserved	RO	0x0	-
2:0	GPIO48_MUX	RW	0x0	GPIO48引脚复用配置： 000：复用为GPIO48；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。

5.5.47 GPIO56~63 复用配置寄存器

GPIO56~63复用配置寄存器。

寄存器地址：0x1fe104ac。

表5-52 GPIO56~63复用配置寄存器

位域	名称	访问	缺省值	描述
31	Reserved	RO	0x0	-
30:28	GPIO63_MUX	RW	0x0	GPIO63引脚复用配置： 000：复用为GPIO63；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
27	Reserved	RO	0x0	-
26:24	GPIO62_MUX	RW	0x0	GPIO62引脚复用配置： 000：复用为GPIO62；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
23	Reserved	RO	0x0	-
22:20	GPIO61_MUX	RW	0x0	GPIO61引脚复用配置： 000：复用为GPIO61；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
19	Reserved	RO	0x0	-
18:16	GPIO60_MUX	RW	0x0	GPIO60引脚复用配置： 000：复用为GPIO60；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
15	Reserved	RO	0x0	-
14:12	GPIO59_MUX	RW	0x0	GPIO59引脚复用配置： 000：复用为GPIO59；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
11	Reserved	RO	0x0	-
10:8	GPIO58_MUX	RW	0x0	GPIO58引脚复用配置： 000：复用为GPIO58；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
7	Reserved	RO	0x0	-
6:4	GPIO57_MUX	RW	0x0	GPIO57引脚复用配置： 000：复用为GPIO57；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
3	Reserved	RO	0x0	-
2:0	GPIO56_MUX	RW	0x0	GPIO56引脚复用配置： 000：复用为GPIO56；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。

5.5.48 GPIO64~71 复用配置寄存器

GPIO64~71复用配置寄存器。

寄存器地址：0x1fe104b0。

表5-53 GPIO64~71复用配置寄存器

位域	名称	访问	缺省值	描述
31	Reserved	RO	0x0	-
30:28	GPIO71_MUX	RW	0x0	GPIO71引脚复用配置： 000：复用为GPIO71；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
27	Reserved	RO	0x0	-
26:24	GPIO70_MUX	RW	0x0	GPIO70引脚复用配置： 000：复用为GPIO70；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
23	Reserved	RO	0x0	-
22:20	GPIO69_MUX	RW	0x0	GPIO69引脚复用配置： 000：复用为GPIO69；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
19	Reserved	RO	0x0	-
18:16	GPIO68_MUX	RW	0x0	GPIO68引脚复用配置： 000：复用为GPIO68；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
15	Reserved	RO	0x0	-
14:12	GPIO67_MUX	RW	0x0	GPIO67引脚复用配置： 000：复用为GPIO67；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
11	Reserved	RO	0x0	-
10:8	GPIO66_MUX	RW	0x0	GPIO66引脚复用配置： 000：复用为GPIO66；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
7	Reserved	RO	0x0	-
6:4	GPIO65_MUX	RW	0x0	GPIO65引脚复用配置： 000：复用为GPIO65；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
3	Reserved	RO	0x0	-
2:0	GPIO64_MUX	RW	0x0	GPIO64引脚复用配置： 000：复用为GPIO64；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。

5.5.49 GPIO72~79 复用配置寄存器

GPIO72~79复用配置寄存器。

寄存器地址：0x1fe104b4。

表5-54 GPIO72~79复用配置寄存器

位域	名称	访问	缺省值	描述
31	Reserved	RO	0x0	-
30:28	GPIO79_MUX	RW	0x0	GPIO79引脚复用配置： 000：复用为GPIO79；001：第一复用；

				010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
27	Reserved	RO	0x0	-
26:24	GPIO78_MUX	RW	0x0	GPIO78引脚复用配置： 000：复用为GPIO78；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
23	Reserved	RO	0x0	-
22:20	GPIO77_MUX	RW	0x0	GPIO77引脚复用配置： 000：复用为GPIO77；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
19	Reserved	RO	0x0	-
18:16	GPIO76_MUX	RW	0x0	GPIO76引脚复用配置： 000：复用为GPIO76；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
15	Reserved	RO	0x0	-
14:12	GPIO75_MUX	RW	0x0	GPIO75引脚复用配置： 000：复用为GPIO75；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
11	Reserved	RO	0x0	-
10:8	GPIO74_MUX	RW	0x0	GPIO74引脚复用配置： 000：复用为GPIO74；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
7	Reserved	RO	0x0	-
6:4	GPIO73_MUX	RW	0x0	GPIO73引脚复用配置： 000：复用为GPIO73；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
3	Reserved	RO	0x0	-
2:0	GPIO72_MUX	RW	0x0	GPIO72引脚复用配置： 000：复用为GPIO72；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。

5.5.50 GPIO80~87 复用配置寄存器

GPIO80~87复用配置寄存器。

寄存器地址：0x1fe104b8。

表5-55 GPIO80~87复用配置寄存器

位域	名称	访问	缺省值	描述
31	Reserved	RO	0x0	-
30:28	GPIO87_MUX	RW	0x0	GPIO87引脚复用配置： 000：复用为GPIO87；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
27	Reserved	RO	0x0	-
26:24	GPIO86_MUX	RW	0x0	GPIO86引脚复用配置： 000：复用为GPIO86；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
23	Reserved	RO	0x0	-
22:20	GPIO85_MUX	RW	0x0	GPIO85引脚复用配置： 000：复用为GPIO85；001：第一复用；

				010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
19	Reserved	RO	0x0	-
18:16	GPIO84_MUX	RW	0x0	GPIO84引脚复用配置： 000：复用为GPIO84；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
15	Reserved	RO	0x0	-
14:12	GPIO83_MUX	RW	0x0	GPIO83引脚复用配置： 000：复用为GPIO83；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
11	Reserved	RO	0x0	-
10:8	GPIO82_MUX	RW	0x0	GPIO82引脚复用配置： 000：复用为GPIO82；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
7	Reserved	RO	0x0	-
6:4	GPIO81_MUX	RW	0x0	GPIO81引脚复用配置： 000：复用为GPIO81；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
3	Reserved	RO	0x0	-
2:0	GPIO80_MUX	RW	0x0	GPIO80引脚复用配置： 000：复用为GPIO80；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。

5.5.51 GPIO88~95 复用配置寄存器

GPIO88~95复用配置寄存器。

寄存器地址：0x1fe104bc。

表5-56 GPIO88~95复用配置寄存器

位域	名称	访问	缺省值	描述
31	Reserved	RO	0x0	-
30:28	GPIO95_MUX	RW	0x0	GPIO95引脚复用配置： 000：复用为GPIO95；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
27	Reserved	RO	0x0	-
26:24	GPIO94_MUX	RW	0x0	GPIO94引脚复用配置： 000：复用为GPIO94；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
23	Reserved	RO	0x0	-
22:20	GPIO93_MUX	RW	0x0	GPIO93引脚复用配置： 000：复用为GPIO93；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
19	Reserved	RO	0x0	-
18:16	GPIO92_MUX	RW	0x0	GPIO92引脚复用配置： 000：复用为GPIO92；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
15	Reserved	RO	0x0	-
14:12	GPIO91_MUX	RW	0x0	GPIO91引脚复用配置： 000：复用为GPIO91；001：第一复用；

				010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
11	Reserved	RO	0x0	-
10:8	GPIO90_MUX	RW	0x0	GPIO90引脚复用配置： 000：复用为GPIO90；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
7	Reserved	RO	0x0	-
6:4	GPIO89_MUX	RW	0x0	GPIO89引脚复用配置： 000：复用为GPIO89；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
3	Reserved	RO	0x0	-
2:0	GPIO88_MUX	RW	0x0	GPIO88引脚复用配置： 000：复用为GPIO88；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。

5.5.52 GPIO96~103 复用配置寄存器

GPIO96~103复用配置寄存器。
寄存器地址：0x1fe104c0。

表5-57 GPIO96~103复用配置寄存器

位域	名称	访问	缺省值	描述
31	Reserved	RO	0x0	-
30:28	GPIO103_MUX	RW	0x0	GPIO103引脚复用配置： 000：复用为GPIO103；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
27	Reserved	RO	0x0	-
26:24	GPIO102_MUX	RW	0x0	GPIO102引脚复用配置： 000：复用为GPIO102；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
23	Reserved	RO	0x0	-
22:20	GPIO101_MUX	RW	0x0	GPIO101引脚复用配置： 000：复用为GPIO101；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
19	Reserved	RO	0x0	-
18:16	GPIO100_MUX	RW	0x0	GPIO100引脚复用配置： 000：复用为GPIO100；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
15	Reserved	RO	0x0	-
14:12	GPIO99_MUX	RW	0x0	GPIO99引脚复用配置： 000：复用为GPIO99；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
11	Reserved	RO	0x0	-
10:8	GPIO98_MUX	RW	0x0	GPIO98引脚复用配置： 000：复用为GPIO98；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
7	Reserved	RO	0x0	-
6:4	GPIO97_MUX	RW	0x0	GPIO97引脚复用配置： 000：复用为GPIO97；001：第一复用；

				010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
3	Reserved	RO	0x0	-
2:0	GPIO96_MUX	RW	0x0	GPIO96引脚复用配置： 000：复用为GPIO96；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。

5.5.53 GPIO104~111 复用配置寄存器

GPIO104~111复用配置寄存器。

寄存器地址：0x1fe104c4。

表5-58 GPIO104~111复用配置寄存器

位域	名称	访问	缺省值	描述
31	Reserved	RO	0x0	-
30:28	GPIO111_MUX	RW	0x0	GPIO111引脚复用配置： 000：复用为GPIO111；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
27	Reserved	RO	0x0	-
26:24	GPIO110_MUX	RW	0x0	GPIO110引脚复用配置： 000：复用为GPIO110；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
23	Reserved	RO	0x0	-
22:20	GPIO109_MUX	RW	0x0	GPIO109引脚复用配置： 000：复用为GPIO109；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
19	Reserved	RO	0x0	-
18:16	GPIO108_MUX	RW	0x0	GPIO108引脚复用配置： 000：复用为GPIO108；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
15	Reserved	RO	0x0	-
14:12	GPIO107_MUX	RW	0x0	GPIO107引脚复用配置： 000：复用为GPIO107；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
11	Reserved	RO	0x0	-
10:8	GPIO106_MUX	RW	0x0	GPIO106引脚复用配置： 000：复用为GPIO106；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
7	Reserved	RO	0x0	-
6:4	GPIO105_MUX	RW	0x0	GPIO105引脚复用配置： 000：复用为GPIO105；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
3	Reserved	RO	0x0	-
2:0	GPIO104_MUX	RW	0x0	GPIO104引脚复用配置： 000：复用为GPIO104；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。

5.5.54 GPIO112~119 复用配置寄存器

GPIO112~119复用配置寄存器。

寄存器地址：0x1fe104c8。

表5-59 GPIO112~119复用配置寄存器

位域	名称	访问	缺省值	描述
31	Reserved	RO	0x0	-
30:28	GPIO119_MUX	RW	0x0	GPIO119引脚复用配置： 000：复用为GPIO119；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
27	Reserved	RO	0x0	-
26:24	GPIO118_MUX	RW	0x0	GPIO118引脚复用配置： 000：复用为GPIO118；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
23	Reserved	RO	0x0	-
22:20	GPIO117_MUX	RW	0x0	GPIO117引脚复用配置： 000：复用为GPIO117；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
19	Reserved	RO	0x0	-
18:16	GPIO116_MUX	RW	0x0	GPIO116引脚复用配置： 000：复用为GPIO116；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
15	Reserved	RO	0x0	-
14:12	GPIO115_MUX	RW	0x0	GPIO115引脚复用配置： 000：复用为GPIO115；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
11	Reserved	RO	0x0	-
10:8	GPIO114_MUX	RW	0x0	GPIO114引脚复用配置： 000：复用为GPIO114；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
7	Reserved	RO	0x0	-
6:4	GPIO113_MUX	RW	0x0	GPIO113引脚复用配置： 000：复用为GPIO113；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。
3	Reserved	RO	0x0	-
2:0	GPIO112_MUX	RW	0x0	GPIO112引脚复用配置： 000：复用为GPIO112；001：第一复用； 010：第二复用；　011：第三复用； 100：第四复用；　其他：引脚主功能。

5.5.55 GPIO120~127 复用配置寄存器

GPIO120~127复用配置寄存器。

寄存器地址：0x1fe104cc。

表5-60 GPIO120~127复用配置寄存器

位域	名称	访问	缺省值	描述
31	Reserved	RO	0x0	-
30:28	GPIO127_MUX	RW	0x0	GPIO127引脚复用配置： 000：复用为GPIO127；001：第一复用；

				010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
27	Reserved	RO	0x0	-
26:24	GPIO126_MUX	RW	0x0	GPIO126引脚复用配置： 000：复用为GPIO126；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
23	Reserved	RO	0x0	-
22:20	GPIO125_MUX	RW	0x0	GPIO125引脚复用配置： 000：复用为GPIO125；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
19	Reserved	RO	0x0	-
18:16	GPIO124_MUX	RW	0x0	GPIO124引脚复用配置： 000：复用为GPIO124；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
15	Reserved	RO	0x0	-
14:12	GPIO123_MUX	RW	0x0	GPIO123引脚复用配置： 000：复用为GPIO123；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
11	Reserved	RO	0x0	-
10:8	GPIO122_MUX	RW	0x0	GPIO122引脚复用配置： 000：复用为GPIO122；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
7	Reserved	RO	0x0	-
6:4	GPIO121_MUX	RW	0x0	GPIO121引脚复用配置： 000：复用为GPIO121；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
3	Reserved	RO	0x0	-
2:0	GPIO120_MUX	RW	0x0	GPIO120引脚复用配置： 000：复用为GPIO120；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。

5.5.56 GPIO128~135 复用配置寄存器

GPIO128~135复用配置寄存器。

寄存器地址：0x1fe104d0。

表5-61 GPIO128~135复用配置寄存器

位域	名称	访问	缺省值	描述
31	Reserved	RO	0x0	-
30:28	GPIO135_MUX	RW	0x0	GPIO135引脚复用配置： 000：复用为GPIO135；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
27	Reserved	RO	0x0	-
26:24	GPIO134_MUX	RW	0x0	GPIO134引脚复用配置： 000：复用为GPIO134；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
23	Reserved	RO	0x0	-
22:20	GPIO133_MUX	RW	0x0	GPIO133引脚复用配置： 000：复用为GPIO133；001：第一复用；

				010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
19	Reserved	RO	0x0	-
18:16	GPIO132_MUX	RW	0x0	GPIO132引脚复用配置： 000：复用为GPIO132；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
15	Reserved	RO	0x0	-
14:12	GPIO131_MUX	RW	0x0	GPIO131引脚复用配置： 000：复用为GPIO131；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
11	Reserved	RO	0x0	-
10:8	GPIO130_MUX	RW	0x0	GPIO130引脚复用配置： 000：复用为GPIO130；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
7	Reserved	RO	0x0	-
6:4	GPIO129_MUX	RW	0x0	GPIO129引脚复用配置： 000：复用为GPIO129；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
3	Reserved	RO	0x0	-
2:0	GPIO128_MUX	RW	0x0	GPIO128引脚复用配置： 000：复用为GPIO128；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。

5.5.57 GPIO136~143 复用配置寄存器

GPIO136~143复用配置寄存器。

寄存器地址：0x1fe104d4。

表5-62 GPIO136~143复用配置寄存器

位域	名称	访问	缺省值	描述
31	Reserved	RO	0x0	-
30:28	GPIO143_MUX	RW	0x0	GPIO143引脚复用配置： 000：复用为GPIO143；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
27	Reserved	RO	0x0	-
26:24	GPIO142_MUX	RW	0x0	GPIO142引脚复用配置： 000：复用为GPIO142；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
23	Reserved	RO	0x0	-
22:20	GPIO141_MUX	RW	0x0	GPIO141引脚复用配置： 000：复用为GPIO141；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
19	Reserved	RO	0x0	-
18:16	GPIO140_MUX	RW	0x0	GPIO140引脚复用配置： 000：复用为GPIO140；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
15	Reserved	RO	0x0	-
14:12	GPIO139_MUX	RW	0x0	GPIO139引脚复用配置： 000：复用为GPIO139；001：第一复用；

				010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
11	Reserved	RO	0x0	-
10:8	GPIO138_MUX	RW	0x0	GPIO138引脚复用配置： 000：复用为GPIO138；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
7	Reserved	RO	0x0	-
6:4	GPIO37_MUX	RW	0x0	GPIO137引脚复用配置： 000：复用为GPIO137；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
3	Reserved	RO	0x0	-
2:0	GPIO136_MUX	RW	0x0	GPIO136引脚复用配置： 000：复用为GPIO136；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。

5.5.58 GPIO144~151 复用配置寄存器

GPIO144~151复用配置寄存器。

寄存器地址：0x1fe104d8。

表5-63 GPIO144~151复用配置寄存器

位域	名称	访问	缺省值	描述
31	Reserved	RO	0x0	-
30:28	GPIO151_MUX	RW	0x0	GPIO151引脚复用配置： 000：复用为GPIO151；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
27	Reserved	RO	0x0	-
26:24	GPIO150_MUX	RW	0x0	GPIO150引脚复用配置： 000：复用为GPIO150；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
23	Reserved	RO	0x0	-
22:20	GPIO149_MUX	RW	0x0	GPIO149引脚复用配置： 000：复用为GPIO149；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
19	Reserved	RO	0x0	-
18:16	GPIO148_MUX	RW	0x0	GPIO148引脚复用配置： 000：复用为GPIO148；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
15	Reserved	RO	0x0	-
14:12	GPIO147_MUX	RW	0x0	GPIO147引脚复用配置： 000：复用为GPIO147；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
11	Reserved	RO	0x0	-
10:8	GPIO146_MUX	RW	0x0	GPIO146引脚复用配置： 000：复用为GPIO146；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
7	Reserved	RO	0x0	-
6:4	GPIO145_MUX	RW	0x0	GPIO145引脚复用配置： 000：复用为GPIO145；001：第一复用；

				010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
3	Reserved	RO	0x0	-
2:0	GPIO144_MUX	RW	0x0	GPIO144引脚复用配置： 000：复用为GPIO144；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。

5.5.59 GPIO152~159 复用配置寄存器

GPIO152~159复用配置寄存器。

寄存器地址：0x1fe104dc。

表5-64 GPIO152~159复用配置寄存器

位域	名称	访问	缺省值	描述
31:11	Reserved	RO	0x0	-
10:8	GPIO154_MUX	RW	0x0	GPIO154引脚复用配置： 000：复用为GPIO154；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
7	Reserved	RO	0x0	-
6:4	GPIO153_MUX	RW	0x0	GPIO153引脚复用配置： 000：复用为GPIO153；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。
3	Reserved	RO	0x0	-
2:0	GPIO152_MUX	RW	0x0	GPIO152引脚复用配置： 000：复用为GPIO152；001：第一复用； 010：第二复用； 011：第三复用； 100：第四复用； 其他：引脚主功能。

5.5.60 GPIO0~31 中断输入使能寄存器

GPIO0~31中断输入使能寄存器。

寄存器地址：0x1fe104e0。

表5-65 GPIO0~31中断输入使能寄存器

位域	名称	访问	缺省值	描述
31:0	GPIO_INTEN[31:0]	RW	0xffffffff	GPIO0~31引脚输入中断使能配置

5.5.61 GPIO32~63 中断输入使能寄存器

GPIO32~63中断输入使能寄存器。

寄存器地址：0x1fe104e4。

表5-66 GPIO32~63中断输入使能寄存器

位域	名称	访问	缺省值	描述
31:0	GPIO_INTEN[63:32]	RW	0xffffffff	GPIO32~63引脚输入中断使能配置

5.5.62 GPIO64~95 中断输入使能寄存器

GPIO64~95中断输入使能寄存器。

寄存器地址：0x1fe104e8。

表5-67 GPIO64~95中断输入使能寄存器

位域	名称	访问	缺省值	描述
31:0	GPIO_INTEN[95:64]	RW	0xffffffff	GPIO64~95引脚输入中断使能配置

5.5.63 GPIO96~127 中断输入使能寄存器

GPIO96~127中断输入使能寄存器。
寄存器地址：0x1fe104ec。

表5-68 GPIO96~127中断输入使能寄存器

位域	名称	访问	缺省值	描述
31:0	GPIO_INTEN[127:96]	RW	0xffffffff	GPIO96~127引脚输入中断使能配置

5.5.64 USB PHY 配置寄存器 0

USB PHY 配置寄存器0，配置 USB 接口0的电气特性。
寄存器地址：0x1fe10500。

表5-69 USB PHY 配置寄存器0

位域	名称	访问	缺省值	描述
31	Reserved	R/O	0	
30	cfg_fs_data_mod	R/W	0	低速模式数据有效使能控制 0：低速模式下，数据信号不受使能信号控制其有效性 1：低速模式下，数据信号受到使能信号控制其有效性
29	commononn0	R/W	0	通用模块关电模式 1：在suspend时，XO，Bias，PLL模块掉电；在睡眠时，Bias，PLL掉电 0：在suspend或睡眠时XO，Bias，PLL都有电
28	dmpulldown0	R/W	0	dm端口下拉电阻使能 1：D-使能 0：D-关闭
27	dppulldown0	R/W	0	dp端口下拉电阻使能 1：D+使能 0：D+关闭
26:25	txrestune0	R/W	0	该信号调整驱动电阻，用来补偿在发送器到电缆之间的寄生电阻 11：-4Ω 10：-2Ω 01：默认 00：+1.5Ω
24	txpreemppulsetune0	R/W	0	调整在高速模式下预加强电流在dp,dm电缆上的持续时间，高速收发器预加强电流持续时间根据时间单位来定义 1：1X，短持续时间 0：2X，长持续时间
23:22	txpreempamptune0	R/W	0	调整在高速模式从J到K或从K到J的跳变时提供的电流量，高速收发器预加强电流量根据电流量单位来定义 11：HS传输预加强3X电流 10：HS传输预加强2X电流 01：HS传输预加强1X电流 00：HS传输预加强电流关闭
21:20	txhsxvtune0	R/W	0	在高速模式下调整dp，dm信号交叉时的电压 11：default 10：+15mv 01：-15mv 00：reserved
19:18	txrisetune0	R/W	0	调整高速波形上升、下降沿次数 11：-10%

				10：default 01：+15% 00：+20%
17:14	txvreftune0	R/W	0	调整高速直流电压 1111：+8.75% 1110：+7.5% 1101：+6.25% 1100：+5% 1011：+3.75% 1010：+2.5% 1001：+1.25% 1000：default 0111：-1.25% 0110：-2.5% 0101：-3.75% 0100：-5% 0011：-6.25% 0010：-7.5% 0001：-8.75% 0000：-10%
13:10	txfslstune0	R/W	0	调整低、全速单端源阻抗 1111：-5% 0111：-2.5% 0011：default 0001：+2.5% 0000：+5%
9:7	sqrxtune0	R/W	0	调整门限电压来检测有效的高速数据 111：-20% 110：-15% 101：-10% 100：-5% 011：default 010：+5% 001：+10% 000：+15%
6:4	compdistune0	R/W	0	调整门限电压用于检测在主控制器上的一个断开连接事件 111：+4.5% 110：+3% 101：+1.5% 100：default 011：-1.5% 010：-3% 001：-4.5% 000：-6%
3:1	otgtune0	R/W	0	调整Vbus Valid的门限电压 111：+9% 110：+6% 101：+3% 100：default 011：-3% 010：-6% 001：-9% 000：-12%
0	cfg_en0	R/W	0	配置使能 0：使用硬件默认配置 1：使用该寄存器的软件配置

5.5.65 USB PHY 配置寄存器 1

USB PHY 配置寄存器1，配置 USB 接口1的电气特性。
寄存器地址：0x1fe10504。

表5-70 USB PHY 配置寄存器1

位域	名称	访问	缺省值	描述
31:30	Reserved	R/O	0	
29	commononn1	R/W	0	通用模块关电模式 1：在suspend时，XO，Bias，PLL模块掉电；在睡眠时，Bias，PLL掉电 0：在suspend或睡眠时XO，Bias，PLL都有电
28	dmpulldown1	R/W	0	dm端口下拉电阻使能 0：D-使能 1：D-关闭
27	dppulldown1	R/W	0	dp端口下拉电阻使能 0：D+使能 1：D+关闭
26:25	txrestune1	R/W	0	同上
24	txpreemppulsetune1	R/W	0	同上
23:22	txpreempamptune1	R/W	0	同上
21:20	txhsxvtune1	R/W	0	同上
19:18	txrisetune1	R/W	0	同上
17:14	txvreftune1	R/W	0	同上
13:10	txfslstune1	R/W	0	同上
9:7	sqrxtune1	R/W	0	同上
6:4	compdistune1	R/W	0	同上
3:1	otgtune1	R/W	0	同上
0	cfg_en1	R/W	0	配置使能 0: 使用硬件默认配置 1: 使用该寄存器的软件配置

5.5.66 USB PHY 配置寄存器 2

USB PHY 配置寄存器2，配置 USB 接口2的电气特性。
寄存器地址：0x1fe10508。

表5-71 USB PHY 配置寄存器2

位域	名称	访问	缺省值	描述
31:30	Reserved	R/O	0	
29	commononn2	R/W	0	通用模块关电模式 1：在suspend时，XO，Bias，PLL模块掉电；在睡眠时，Bias，PLL掉电 0：在suspend或睡眠时XO，Bias，PLL都有电
28	dmpulldown2	R/W	0	dm端口下拉电阻使能 1：D-使能 0：D-关闭
27	dppulldown2	R/W	0	dp端口下拉电阻使能 1：D+使能 0：D+关闭
26:25	txrestune2	R/W	0	同上
24	txpreemppulsetune2	R/W	0	同上
23:22	txpreempamptune2	R/W	0	同上
21:20	txhsxvtune2	R/W	0	同上
19:18	txrisetune2	R/W	0	同上

17:14	txvreftune2	R/W	0	同上
13:10	txfslstune2	R/W	0	同上
9:7	sqrxtune2	R/W	0	同上
6:4	compdistune2	R/W	0	同上
3:1	otgtune2	R/W	0	同上
0	cfg_en2	R/W	0	配置使能 0：使用硬件默认配置 1：使用该寄存器的软件配置

5.5.67 USB PHY 配置寄存器 3

USB PHY 配置寄存器3，配置 USB 接口3的电气特性。
寄存器地址：0x1fe1050c。

表5-72 USB PHY 配置寄存器3

位域	名称	访问	缺省值	描述
31:30	Reserved	R/O	0	
29	commononn3	R/W	0	通用模块关电模式 1：在suspend时，XO，Bias，PLL模块掉电；在睡眠时，Bias，PLL掉电 0：在suspend或睡眠时XO，Bias，PLL都有电
28	dmpulldown3	R/W	0	dm端口下拉电阻使能 0：D-使能 1：D-关闭
27	dppulldown3	R/W	0	dp端口下拉电阻使能 0：D+使能 1：D+关闭
26:25	txrestune3	R/W	0	同上
24	txpreemppulsetune3	R/W	0	同上
23:22	txpreempamptune3	R/W	0	同上
21:20	txhsxvtune3	R/W	0	同上
19:18	txrisetune3	R/W	0	同上
17:14	txvreftune3	R/W	0	同上
13:10	txfslstune3	R/W	0	同上
9:7	sqrxtune3	R/W	0	同上
6:4	compdistune3	R/W	0	同上
3:1	otgtune3	R/W	0	同上
0	cfg_en3	R/W	0	配置使能 0：使用硬件默认配置 1：使用该寄存器的软件配置

5.5.68 PCIE 配置寄存器 0

PCIE 配置寄存器0，配置 PCIE 接口 PHY 的控制信号。
寄存器地址：0x1fe10550。

表5-73 PCIE 配置寄存器0

位域	名称	访问	缺省值	描述
31:27	pcs_tx_deemph_gen1[4:0]	R/W	0x10	设置PCIE PHY在GEN1模式下的tx deemphasis值
26:24	phy_rx3_eq	R/W	0x2	LANE3 接收端equalizer设置
23:21	phy_rx2_eq	R/W	0x2	LANE2 接收端equalizer设置
20:18	phy_rx1_eq	R/W	0x2	LANE1 接收端equalizer设置
17:15	phy_rx0_eq	R/W	0x2	LANE0 接收端equalizer设置

14	pcs_common_clocks	R/W	0	common clock mode设置，只有PCIE链路两端使用相同参考时钟时才可设置
13	pcs_clk_req	R/W	0x1	表示PCIE PHY在P2状态下需要输出PIPE时钟
12	-	-	-	保留
11:5	phy_mpll_multiplier	R/W	0x19	PCIE PHY PLL的倍频数
4	-	-	-	保留
3	app_req_retry_en	R/W	0x0	保留
2	do_sleep	R/W	0x0	保留
1	power_fault	R/W	0x0	保留
0	slot_wake_n	R/W	0x1	保留

5.5.69 PCIE 配置寄存器 1

PCIE 配置寄存器1，配置 PCIE 接口 PHY 的控制信号。
寄存器地址：0x1fe10554。

表5-74 PCIE 配置寄存器1

位域	名称	访问	缺省值	描述
31:27	phy_tx0_term_offset	R/W	0	设置PCIE LANE0发送端阻抗在50欧左右的微调值
26:20	pcs_tx_swing_low	R/W	0x2	设置PCIE PHY发送低摆幅幅值
19:13	pcs_tx_swing_full	R/W	0x10	设置PCIE PHY发送满摆幅幅值
12:7	pcs_tx_deemph_gen2_6db	R/W	0x30	设置PCIE PHY在GEN2模式下的-6dB tx deemphasis值
6:1	pcs_tx_deemph_gen2_3p5db	R/W	0x20	设置PCIE PHY在GEN2模式下的-3.5dB tx deemphasis值
0	pcs_tx_deemph_gen1[5]	R/W	0x0	设置PCIE PHY在GEN1模式下的tx deemphasis值

5.5.70 PCIE 配置寄存器 2

PCIE 配置寄存器2，配置 PCIE 接口 PHY 的控制信号。
寄存器地址：0x1fe10558。

表5-75 PCIE 配置寄存器2

位域	名称	访问	缺省值	描述
31:25	Reserved	R/O	0	
24	phy_powerdown	R/W	0	设置PCIE PHY进入低功耗模式
23:20	pcie0_refclk_en	R/W	0	PCIE参考时钟输出使能
19	phy_rtune_req	R/W	0	发起resister tune请求
18	vreg_bypass	R/W	0	PCIE PHY内部voltage regulator使能位。Vph接2.5V电压时设置为1，Vph接3.3V电压时设置为0
17:15	phy_tx_vboost_lvl	R/W	0x4	Tx voltage boost level
14:10	phy_tx3_term_offset	R/W	0	设置PCIE LANE3发送端阻抗在50欧左右的微调值
9:5	phy_tx2_term_offset	R/W	0	设置PCIE LANE2发送端阻抗在50欧左右的微调值
4:0	phy_tx1_term_offset	R/W	0	设置PCIE LANE1发送端阻抗在50欧左右的微调值

5.5.71 PCIE 配置寄存器 3

PCIE 配置寄存器3，配置 PCIE 接口 PHY 的控制信号。
寄存器地址：0x1fe1055c。

表5-76 PCIE 配置寄存器3

位域	名称	访问	缺省值	描述

31:0	Reserved	R/O	0	

5.5.72 PCIE PHY 配置控制寄存器 0

PCIE PHY 配置控制寄存器0，配置 PCIE 接口 PHY 的内部寄存器。
寄存器地址：0x1fe10560。

表5-77 PCIE PHY 配置控制寄存器0

位域	名称	访问	缺省值	描述
31:16	phy_cfg_data	R/W	0	PHY配置读写数据。在写操作时，将数据先写入该寄存，然后再执行写操作；在读操作时，从PHY返回的读数据存储到该寄存器。
15:0	phycfg_addr	R/W	0	PHY配置地址

5.5.73 PCIE PHY 配置控制寄存器 1

PCIE PHY 配置控制寄存器1，配置 PCIE 接口 PHY 的内部寄存器。
寄存器地址：0x1fe10564。

表5-78 PCIE PHY 配置控制寄存器1

位域	名称	访问	缺省值	描述
31:7	Reserved	R/O	0	
6	phy_cfg_reset	R/W	0	PHY配置重置，高有效
5:3	phy_cfg_state	R	0	PHY配置状态机状态指示
2	phy_cfg_done	R/W	0	PHY一次访问完成，指示此次对PHY的读写完成。写完成表示写的数据已经写入PHY内部寄存器，读完成表示读的数据已经存储到write/read data寄存器
1	phy_cfg_disable	R/W	0x0	0--对该组寄存器读写会触发PHY配置操作 1-对该组寄存器读写不触发PHY配置操作，仅为简单的寄存器读写
0	phy_cfg_rw	R/W	0	开始读操作或写操作。为1时为写操作，为0时为读操作。

5.5.74 SATA0 PHY 配置寄存器 0

SATA0 PHY 配置寄存器0，配置 SATA0接口 PHY 的控制信号。
寄存器地址：0x1fe10570。

表5-79 SATA0 PHY 配置寄存器0

位域	名称	访问	缺省值	描述
31	P0_tx_amplitude_gen2[0]	R/W	0x1	GEN2模式下发送端摆幅设置
30:24	P0_tx_amplitude_gen1	R/W	0x7f	GEN1模式下发送端摆幅设置
23	Sata_prefetch	R/W	0x1	Sata接口预取
22	Sata_flush_wr	R/W	0x0	Sata接口读请求刷出写请求
21	Sata_stop_waw	R/W	0x0	Sata接口停止写后写
20	Sata_stop_raw	R/W	0x1	Sata接口停止写后读
19:16	Sata_flush_idle	R/W	0xf	Sata接口空闲写入周期数
15	Vreg_bypass	R/W	0x0	PHY内部继电器使能。Vph接2.5V电压时设置为1，Vph接3.3V电压时设置为0
14:12	P0_rx_eq	R/W	0x0	Rx equalizer设置
11				
10	Dma_coherent	R/W	0x1	dmacache一致性使能
9	P0_tx_invert	R/W	0x0	LANE0 发送端极性反向使能 1代表反向，0代表不反向

8	P0_rx_invert	R/W	0x0	LANE0 接收端极性反向使能 1代表反向，0代表不反向
7	Ssc_en	R/W	0x0	Spread Spectrum使能
6:4	Ssc_range	R/W	0x0	Spread SpectrumClock Range
3	P0_resetn	R/W	0x1	LANE0软复位，0有效
2	Phy_resetn	R/W	0x1	PHY软复位，0有效
1	Ref_use_pad	R/W	0x0	参考时钟输入选择： 0- 选择内部时钟 1- 选择外部时钟
0	Ref_ssp_en	R/W	0x1	PHY参考

5.5.75 SATA0 PHY 配置寄存器 1

SATA0 PHY 配置寄存器1，配置 SATA0接口 PHY 的控制信号。
寄存器地址：0x1fe10574。

表5-80 SATA0 PHY 配置寄存器1

位域	名称	访问	缺省值	描述
31	Phy_power_down	R/W	0x0	关闭SATA PHY
30:25	P0_tx_preemph_gen3	R/W	0x0	GEN3模式下发送端预加重设置
24:19	P0_tx_preemph_gen2	R/W	0x0	GEN2模式下发送端预加重设置
18:13	P0_tx_preemph_gen1	R/W	0x0	GEN1模式下发送端预加重设置
12:6	P0_tx_amplitude_gen3	R/W	0x7f	GEN3模式下发送端摆幅设置
5:0	P0_tx_amplitude_gen2 [6:1]	R/W	0x3f	GEN2模式下发送端摆幅设置

5.5.76 SATA1 PHY 配置寄存器 0

SATA1 PHY 配置寄存器0，配置 SATA1接口 PHY 的控制信号。
寄存器地址：0x1fe10578。

表5-81 SATA1 PHY 配置寄存器0

位域	名称	访问	缺省值	描述
31	P0_tx_amplitude_gen2 [0]	R/W	0x1	GEN2模式下发送端摆幅设置
30:24	P0_tx_amplitude_gen1	R/W	0x7f	GEN1模式下发送端摆幅设置
23	Sata_prefetch	R/W	0x1	Sata接口预取
22	Sata_flush_wr	R/W	0x0	Sata接口读请求刷出写请求
21	Sata_stop_waw	R/W	0x0	Sata接口停止写后写
20	Sata_stop_raw	R/W	0x1	Sata接口停止写后读
19:16	Sata_flush_idle	R/W	0xf	Sata接口空闲写入周期数
15	Vreg_bypass	R/W	0x0	PHY内部继电器使能。Vph接2.5V电压时设置为1，Vph接3.3V电压时设置为0
14:12	P0_rx_eq	R/W	0x0	Rx equalizer设置
11				
10	Dma_coherent	R/W	0x1	dmacache一致性使能
9	P0_tx_invert	R/W	0x0	LANE0 发送端极性反向使能 1代表反向，0代表不反向
8	P0_rx_invert	R/W	0x0	LANE0 接收端极性反向使能 1代表反向，0代表不反向
7	Ssc_en	R/W	0x0	Spread Spectrum使能

6:4	Ssc_range	R/W	0x0	Spread SpectrumClock Range
3	P0_resetn	R/W	0x1	LANE0软复位，0有效
2	Phy_resetn	R/W	0x1	PHY软复位，0有效
1	Ref_use_pad	R/W	0x0	参考时钟输入选择： 0- 选择内部时钟 1- 选择外部时钟
0	Ref_ssp_en	R/W	0x1	PHY参考

5.5.77 SATA1 PHY 配置寄存器 1

SATA1 PHY 配置寄存器1，配置 SATA1接口 PHY 的控制信号。
寄存器地址：0x1fe1057c。

表5-82 SATA1 PHY 配置寄存器1

位域	名称	访问	缺省值	描述
31	Phy_power_down	R/W	0x0	关闭SATA PHY
30:25	P0_tx_preemph_gen3	R/W	0x0	GEN3模式下发送端预加重设置
24:19	P0_tx_preemph_gen2	R/W	0x0	GEN2模式下发送端预加重设置
18:13	P0_tx_preemph_gen1	R/W	0x0	GEN1模式下发送端预加重设置
12:6	P0_tx_amplitude_gen3	R/W	0x7f	GEN3模式下发送端摆幅设置
5:0	P0_tx_amplitude_gen2 [6:1]	R/W	0x3f	GEN2模式下发送端摆幅设置

5.5.78 SATA 模块转换桥配置寄存器 0

SATA 模块转换桥配置寄存器0，配置 SATA 模块转换桥模式控制信号。
寄存器地址：0x1fe10580。

表5-83 模块转换桥配置寄存器0

位域	名称	访问	缺省值	描述
31	stop_cpu_rd_for_dma_wr	R/W	0x1	当存在未响应的DMA写请求时，使能对处理器读访问的阻塞
30	invld_pref_on_cpu_wr	R/W	0x1	当处理器写控制器内部寄存器时，无效掉预取的读数据
29	stop_cpu_wr_for_dma_wr	R/W	0x0	当存在未响应的DMA写请求时，使能对处理器写访问的阻塞（当bit31有效时才生效）
28:27	Reserved	R/O	0x0	
26:24	pref_cond	R/W	0x7	使能对4/16/32字节访问的预取
23:22	Reserved	R/O	0x0	
21:16	tolerant_cycle	R/W	0x8	当处理器需要读取控制器内部寄存器时，会阻止控制器进行DMA写操作。此参数配置的是允许处理器访问被阻塞的周期数（该值乘以4），当阻塞时间到达后，停止后续的DMA写操作。
15:12	pref_limit	R/W	0x8	预取请求地址边界配置。地址边界等于4K*2^N。
11:8	pref_max_num	R/W	0x8	预取的最大个数
7:4	pref_num_on_static	R/W	0x3	静态预取策略下，一次预取的个数
3	pref_static_en	R/W	0x0	使能静态预取策略
2	invld_pref_on_dma_wr	R/W	0x1	写请求无效所有的读预取数据
1	pref_disable	R/W	0x0	关闭预取
0	rd_wait_wr	R/W	0x0	读请求必须等待所有的写请求完成（写响应到达），即使读写地址不冲突。当读写地址冲突时，读请求

				会被强制等待写请求完成。

5.5.79 SATA 模块转换桥配置寄存器 1

SATA 模块转换桥配置寄存器1，配置 SATA 模块转换桥模式控制信号。
寄存器地址：0x1fe10584。

表5-84 模块转换桥配置寄存器1

位域	名称	访问	缺省值	描述
31:0	Reserved	-	-	保留

5.5.80 SATA0 PHY 配置访问寄存器 0

SATA0 PHY 配置访问寄存器0，用于控制对 SATA0 PHY 内部控制寄存器的访问操作。
寄存器地址：0x1fe10590。

表5-85 SATA0 PHY 配置访问寄存器0

位域	名称	访问	缺省值	描述
31:16	phy_cfg_data	R/W	0x0	PHY配置读写数据： 写操作时，将数据先写入该寄存器，然后再执行写操作；读操作时，从PHY返回的读数据存储到该寄存器
15:0	phy_cfg_addr	R/W	0x0	PHY配置地址

5.5.81 SATA0 PHY 配置访问寄存器 1

SATA0 PHY 配置访问寄存器1，用于控制对 SATA0 PHY 内部控制寄存器的访问操作。
寄存器地址：0x1fe10594。

表5-86 SATA0 PHY 配置访问寄存器1

位域	名称	访问	缺省值	描述
31:1	Reserved	-	-	保留
0	phy_cfg_R/W	R/W	0x0	开始读操作或写操作： 0：读操作；1：写操作

5.5.82 SATA1 PHY 配置访问寄存器 0

SATA1 PHY 配置访问寄存器0，用于控制对 SATA1 PHY 内部控制寄存器的访问操作。
寄存器地址：0x1fe10598。

表5-87 SATA1 PHY 配置访问寄存器0

位域	名称	访问	缺省值	描述
31:16	phy_cfg_data	R/W	0x0	PHY配置读写数据： 写操作时，将数据先写入该寄存器，然后再执行写操作；读操作时，从PHY返回的读数据存储到该寄存器
15:0	phy_cfg_addr	R/W	0x0	PHY配置地址

5.5.83 SATA1 PHY 配置访问寄存器 1

SATA1 PHY 配置访问寄存器1，用于控制对 SATA1 PHY 内部控制寄存器的访问操作。
寄存器地址：0x1fe1059c

表5-88 SATA1 PHY 配置访问寄存器1

位域	名称	访问	缺省值	描述

31:1	Reserved	-	-	保留
0	phy_cfg_R/W	R/W	0x0	开始读操作或写操作： 0：读操作；1：写操作

5.5.84 DMA0 配置寄存器

DMA0控制器的配置寄存器，该寄存器用来控制 DMA0控制器。
寄存器地址：0x1fe10c00。

表5-89 DMA0配置控制寄存器

位域	名称	访问	缺省值	描述
31:5	ask_addr	R/W	0	32位地址高27
4	dma_stop	R/W	0	停止DMA操作。DMA控制器完成当前数据读写后停止。
3	dma_start	R/W	0	开始DMA操作。DMA控制器读取描述符地址(ask_addr)后将该位清零。
2	ask_valid	R/W	0	DMA工作寄存器写回到(ask_addr)所指向的内存，完成后清零。
1:0	Reserved	R/O	0x0	保留

5.5.85 DMA1 配置寄存器

DMA1控制器的配置寄存器，该寄存器用来控制 DMA1控制器。
寄存器地址：0x1fe10c10。

表5-90 DMA1配置控制寄存器

位域	名称	访问	缺省值	描述
31:5	ask_addr	R/W	0	32位地址高27
4	dma_stop	R/W	0	停止DMA操作。DMA控制器完成当前数据读写后停止。
3	dma_start	R/W	0	开始DMA操作。DMA控制器读取描述符地址(ask_addr)后将该位清零。
2	ask_valid	R/W	0	DMA工作寄存器写回到(ask_addr)所指向的内存，完成后清零。
1:0	Reserved	R/O	0x0	保留

5.5.86 DMA2 配置寄存器

DMA2控制器的配置寄存器，该寄存器用来控制 DMA2控制器。
寄存器地址：0x1fe10c20。

表5-91 DMA2配置控制寄存器

位域	名称	访问	缺省值	描述
31:5	ask_addr	R/W	0	32位地址高27
4	dma_stop	R/W	0	停止DMA操作。DMA控制器完成当前数据读写后停止。
3	dma_start	R/W	0	开始DMA操作。DMA控制器读取描述符地址(ask_addr)后将该位清零。
2	ask_valid	R/W	0	DMA工作寄存器写回到(ask_addr)所指向的内存，完成后清零。
1:0	Reserved	R/O	0x0	保留

5.5.87 DMA3 配置寄存器

DMA3控制器的配置寄存器，该寄存器用来控制 DMA3控制器。
寄存器地址：0x1fe10c30。

表5-92 DMA3配置控制寄存器

位域	名称	访问	缺省值	描述
31:5	ask_addr	R/W	0	32位地址高27
4	dma_stop	R/W	0	停止DMA操作。DMA控制器完成当前数据读写后停止。
3	dma_start	R/W	0	开始DMA操作。DMA控制器读取描述符地址(ask_addr)后将该位清零。
2	ask_valid	R/W	0	DMA工作寄存器写回到(ask_addr)所指向的内存，完成后清零。
1:0	Reserved	R/O	0x0	保留

5.6 中断配置及路由

龙芯2K0500芯片最多支持64个中断源，以统一方式进行管理，如下图所示，任意一个IO中断源可以被配置为是否使能、触发的方式、以及被路由的目标处理器核中断脚。

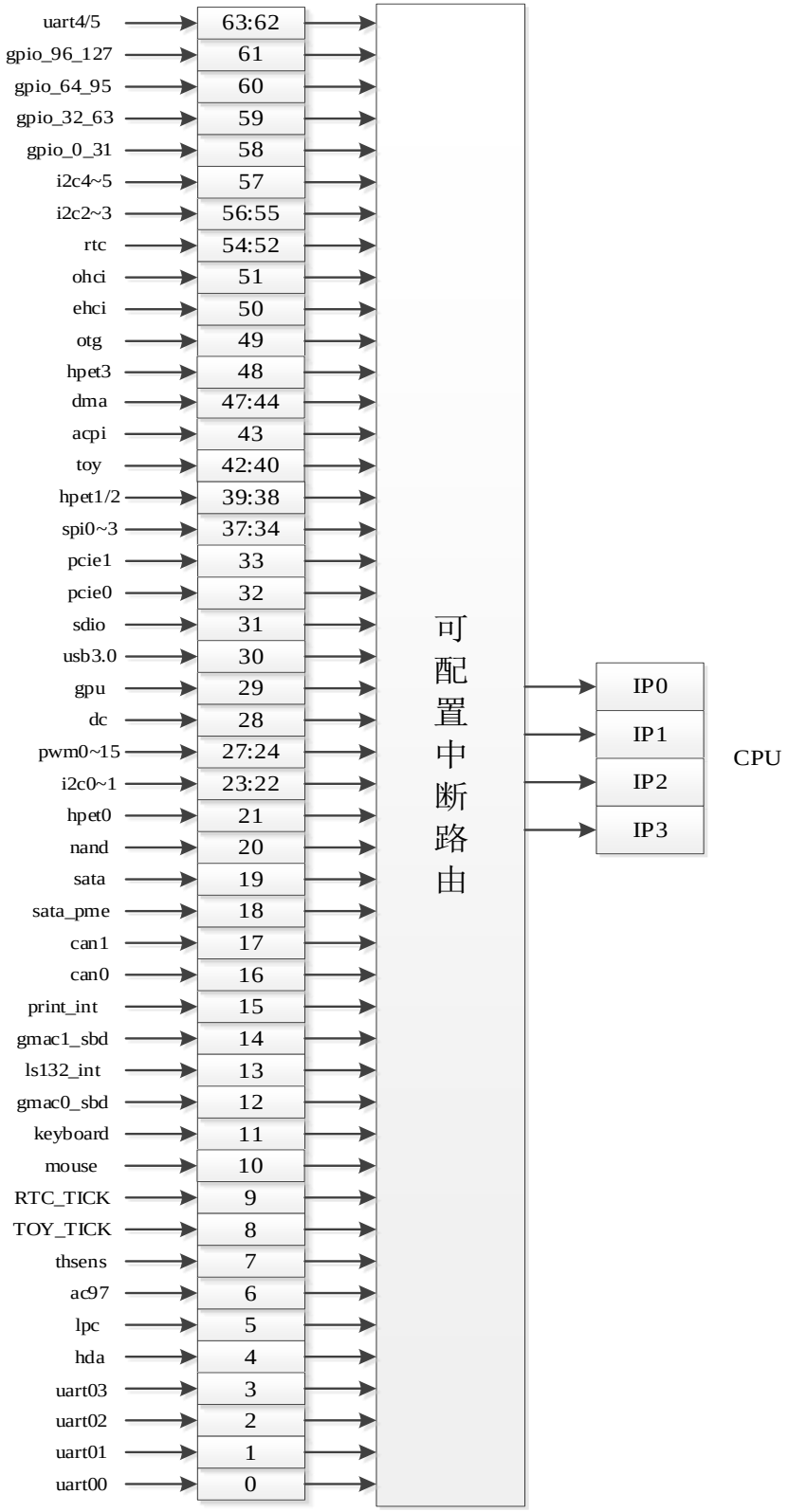


图5-4 龙芯 2K0500 处理器中断路由示意图

中断相关配置寄存器都是以位的形式对相应的中断线进行控制，中断控制位连接及属性配置见表 5-93。中断使能（Enable）的配置有三个寄存器：Intenset、Intenclr 和 Inten。Intenset 设置中断使能，Intenset 寄存器写 1 的位对应的中断被使能。Intenclr 清除中断使能，Intenclr 寄存器写 1 的位对应的中断被清除。Inten 寄存器读取当前各中断使能的情况。脉冲形式的中断信号由 Intedge 配置寄存器来选择，写 1 表示脉冲触发，写 0 表示电平触发。中断处理程序可以通过 Intenclr 的相应位来清除脉冲记录,在中断被清除后，需要配置相应的 Intenset 才能采集到该中断的下一次脉冲触发。

表5-93 芯片IO中断寄存器列表

地址	名称	描述
0x1fe11040	CORE_INTISR0	路由给CORE的低32位中断状态
0x1fe11044	INTISR0	低32位中断状态寄存器
0x1fe11048	CORE_INTISR1	路由给CORE的高32位中断状态
0x1fe1104c	INTISR1	高32位中断状态寄存器
0x1fe11400	ENTRY0_0	8位中断路由寄存器[0--7]
0x1fe11408	ENTRY8_0	8位中断路由寄存器[8--15]
0x1fe11410	ENTRY16_0	8位中断路由寄存器[16--23]
0x1fe11418	ENTRY24_0	8位中断路由寄存器[24--31]
0x1fe11420	INTISR_0	低32位中断状态寄存器
0x1fe11424	INTIEN_0	低32位中断使能状态寄存器
0x1fe11428	INTSET_0	低32位设置使能寄存器
0x1fe1142c	INTCLR_0	低32位中断清除寄存器，清除使能寄存器和脉冲触发的中断
0x1fe11430	INTPOL_0	低32位极性设置寄存器(电平中断)
0x1fe11434	INTEDGE_0	低32位触发方式寄存器（1：脉冲触发；0：电平触发）
0x1fe11440	ENTRY0_1	8位中断路由寄存器[32--39]
0x1fe11448	ENTRY8_1	8位中断路由寄存器[40--47]
0x1fe11450	ENTRY16_1	8位中断路由寄存器[48--55]
0x1fe11458	ENTRY24_1	8位中断路由寄存器[56--63]
0x1fe11460	INTISR_1	高32位中断状态寄存器
0x1fe11464	INTIEN_1	高32位中断使能状态寄存器
0x1fe11468	INTSET_1	高32位设置使能寄存器
0x1fe1146c	INTCLR_1	高32位中断清除寄存器，清除使能寄存器和脉冲触发的中断
0x1fe11470	INTPOL_1	高32位极性设置寄存器(电平中断)
0x1fe11474	INTEDGE_1	高32位触发方式寄存器（1：脉冲触发；0：电平触发）
0x1fe11500	Thsens_int_ctrl_Hi0	温度传感器高温中断控制寄存器0
0x1fe11504	Thsens_int_ctrl_Hi1	温度传感器高温中断控制寄存器1
0x1fe11508	Thsens_int_ctrl_Lo0	温度传感器低温中断控制寄存器0
0x1fe1150c	Thsens_int_ctrl_Lo1	温度传感器低温中断控制寄存器1
0x1fe11510	Thsens_int_status0/clr0	温度传感器中断状态寄存器0
0x1fe11514	Thsens_int_status1/clr1	温度传感器中断状态寄存器1
0x1fe11520	Thsens_scale_hi0	温度传感器测量寄存器0

0x1fe11524	Thsens_scale_hi1	温度传感器测量寄存器1
0x1fe11528	Thsens_scale_lo0	保留
0x1fe1152c	Thsens_scale_lo1	保留

5.6.1 中断触发类型

对于 2K0500 来说，DMA 控制器中断为脉冲触发类型，GPIO 中断根据需要可以配置成电平触发或者脉冲触发，其余中断均为电平触发类型。

5.6.2 中断相关寄存器描述

表5-94 中断控制寄存器属性

位域	访问属性/缺省值						
	Intedge	Inten	Intenset	Intenclr	Intpol	Intentry	中断源
0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Uart00
1	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Uart01
2	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Uart02
3	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Uart03
4	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Hda_int
5	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Lpc_int
6	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Ac97_int
7	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Thsens_int
8	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	TOY_TICK
9	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	RTC_TICK
10	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Mouse
11	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Keyboard
12	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Gmac0_sbd_int
13	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	LA132_int
14	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Gmac1_sbd_int
15	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Print _int
16	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Can0/1_int
17	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Can2/3_int
18	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Sata_pme
19	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Sata_int
20	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Nand_int
21	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Hpet0_int
22	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	I2c0_int

位域	访问属性/缺省值						
23	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	I2c1_int
24	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Pwm0~3_int
25	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Pwm4~7_int
26	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Pwm8~11_int
27	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Pwm12~15_int
28	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Dc_int
29	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Gpu_int
30	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Usb3_int
31	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Sdio0/1_int
32	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Pcie0_int
33	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Pcie1_int
34	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Spi0_int
35	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Spi1_int
36	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Spi2/3_int
37	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Spi4/5_int
38	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Hpet1_int
39	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Hpet2_int
40	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Toy0_int
41	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Toy1_int
42	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Toy2_int
43	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Acpi_int
44	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Dma0_int
45	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Dma1_int
46	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Dma2_int
47	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Dma3_int
48	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Hpet3_int
49	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Otg_int
50	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Ehci_int
51	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Ohci_int
52	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Rtc0_int
53	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Rtc1_int
54	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Rtc2_int
55	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	I2c2_int
56	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	I2c3_int
57	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	I2c4/5_int
58	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Gpio_0_31_int

位域	访问属性/缺省值						
59	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Gpio_32_63_int
60	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Gpio_64_95_int
61	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Gpio_96_127_int
62	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Uart4~6_int
63	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Uart7~9_int

表5-95 中断控制寄存器地址

名称	地址偏移	访问属性	缺省值	描述
Intisr_0	0x1fe11420	RO	NA	低 32 位中断状态寄存器
Inten_0	0x1fe11424	RO	NA	低 32 位中断使能状态寄存器
Intenset_0	0x1fe11428	WO	NA	低 32 位设置使能寄存器
Intenclr_0	0x1fe1142c	WO	NA	低 32 位清除使能寄存器和脉冲触发的中断
Intpol_0	0x1fe11430	WO	0x0	低 32 位中断极性选择寄存器（1：极性取反，0：极性相同）
Intedge_0	0x1fe11434	WO	0x0	低 32 位触发方式寄存器（1：脉冲触发；0：电平触发）
Intisr_1	0x1fe11460	RO	NA	高 32 位中断状态寄存器
Inten_1	0x1fe11464	RO	NA	高 32 位中断使能状态寄存器
Intenset_1	0x1fe11468	WO	NA	高 32 位设置使能寄存器
Intenclr_1	0x1fe1146c	WO	NA	高 32 位清除使能寄存器和脉冲触发的中断
Intpol_1	0x1fe11470	WO	0x0	高 32 位中断极性选择寄存器（1：极性取反，0：极性相同）
Intedge_1	0x1fe11474	WO	0x0	高 32 位触发方式寄存器（1：脉冲触发；0：电平触发）
CORE_IPISR	0x1fe11000	RO	NA	处理器核的 IPI_Status 寄存器
CORE_IPIEN	0x1fe11004	RW	0x0	处理器核的 IPI_Enalbe 寄存器
CORE_IPISET	0x1fe11008	WO	NA	处理器核的 IPI_Set 寄存器
CORE_IPI_CLR	0x1fe1100c	WO	NA	处理器核的 IPI_Clear 寄存器
CORE_INTISR0	0x1fe11040	RO	NA	路由给 CORE 的低 32 位中断状态
CORE_INTISR1	0x1fe11048	RO	NA	路由给 CORE 的高 32 位中断状态

5.6.3 中断路由寄存器描述

龙芯 2K0500 中断源可以选择路由到处理器核中断 INT0 到 INT3 中的任意一个。64 个 I/O 中断源中每一个都对应一个 8 位的路由控制器，其格式和地址如下表所示。路由寄存器采用向量的方式进行路由选择，如 0x40 表示路由到处理器的 INT2 上。

表5-96 中断路由寄存器的说明

位域	说　明
3:0	保留
7:4	路由的处理器核中断引脚向量号

表5-97 中断路由寄存器地址

名称	地址偏移	描述	名称	地址偏移	描述
Entry0	0x1fe11400	Uart00	Entry32	0x1fe11440	Pcie0_int
Entry1	0x1fe11401	Uart01	Entry33	0x1fe11441	Pcie1_int
Entry2	0x1fe11402	Uart02	Entry34	0x1fe11442	Spi0_int
Entry3	0x1fe11403	Uart03	Entry35	0x1fe11443	Spi1_int
Entry4	0x1fe11404	Hda_int	Entry36	0x1fe11444	Spi2_int
Entry5	0x1fe11405	Lpc_int	Entry37	0x1fe11445	Spi3_int
Entry6	0x1fe11406	Ac97_int	Entry38	0x1fe11446	Hpet1_int
Entry7	0x1fe11407	Thsens_int	Entry39	0x1fe11447	Hpet2_int
Entry8	0x1fe11408	TOY_TICK	Entry40	0x1fe11448	Toy_int0
Entry9	0x1fe11409	RTC_TICK	Entry41	0x1fe11449	Toy_int1
Entry10	0x1fe1140a	Mouse_int	Entry42	0x1fe1144a	Toy_int2
Entry11	0x1fe1140b	Keyboard_int	Entry43	0x1fe1144b	Acpi_int
Entry12	0x1fe1140c	Gmac0_sbd_int	Entry44	0x1fe1144c	Dma_int0
Entry13	0x1fe1140d	LA132_int	Entry45	0x1fe1144d	Dma_int1
Entry14	0x1fe1140e	Gmac1_sbd_int	Entry46	0x1fe1144e	Dma_int2
Entry15	0x1fe1140f	Print_int	Entry47	0x1fe1144f	Dma_int3
Entry16	0x1fe11410	Can0_int	Entry48	0x1fe11450	Hpet3_int
Entry17	0x1fe11411	Can1_int	Entry49	0x1fe11451	Otg_int
Entry18	0x1fe11412	Sata_pme	Entry50	0x1fe11452	Ehci_int
Entry19	0x1fe11413	Sata_int	Entry51	0x1fe11453	Ohci_int
Entry20	0x1fe11414	Nand_int	Entry52	0x1fe11454	Rtc_int0
Entry21	0x1fe11415	Hpet0_int	Entry53	0x1fe11455	Rtc_int1
Entry22	0x1fe11416	I2c_int0	Entry54	0x1fe11456	Rtc_int3
Entry23	0x1fe11417	I2c_int1	Entry55	0x1fe11457	I2c2_int
Entry24	0x1fe11418	Pwm_int0	Entry56	0x1fe11458	I2c3_int
Entry25	0x1fe11419	Pwm_int1	Entry57	0x1fe11459	I2c4_int
Entry26	0x1fe1141a	Pwm_int2	Entry58	0x1fe1145a	Gpio_int _0_31
Entry27	0x1fe1141b	Pwm_int3	Entry59	0x1fe1145b	Gpio_int _32_63
Entry28	0x1fe1141c	Dc_int	Entry60	0x1fe1145c	Gpio_int _64_95
Entry29	0x1fe1141d	Gpu_int	Entry61	0x1fe1145d	Gpio_int _96_127
Entry30	0x1fe1141e	Usb3_int	Entry62	0x1fe1145e	Uart4_int
Entry31	0x1fe1141f	Sdio_int	Entry63	0x1fe1145f	Uart5_int

5.6.4 GPIO 中断

龙芯 2K0500 有 155 个 GPIO 引脚，其中 128 个 GPIO 引脚可作为外部中断输入引脚使用，这些 GPIO 引脚与芯片内部中断引脚的对应关系如下：

表5-98 GPIO 中断

GPIO 引脚	中断引脚	说明
GPIO0~31	Gpio_int_0_31	GPIO0~GPIO31 复用一个中断引脚 Gpio_int_0_31
GPIO32~63	Gpio_int _32_63	GPIO32~GPIO63 复用一个中断引脚 Gpio_int_32_63
GPIO64~95	Gpio_int _64_95	GPIO64~GPIO95 复用一个中断引脚 Gpio_int_64_95
GPIO96~127	Gpio_int _96_127	GPIO96~GPIO127 复用一个中断引脚 Gpio_int_96_127

5.6.5 扩展 IO 中断

龙芯 2K0500 除了支持上述传统 IO 中断方式外，还增加了扩展 IO 中断方式，即：将 2K0500 芯片中所有 IO 设备中断，全部映射至全新的扩展 IO 中断向量，并增加相应扩展中断使能、中断状态、中断清除及路由等功能。该扩展 IO 中断向量最多支持 128 个 IO 设备，可独立于传统 IO 中断处理方式之外，填补原传统 IO 中断方式仅 64 个有限中断源的处理限制，提升 IO 中断使用的灵活性（注：GPIO 引脚外部中断输入扩展中断方式仅支持 GPIO0~123，其中，GPIO[4*n+3]，(n=0~30)引脚不支持扩展中断）。

扩展 IO 中断芯片内部采用全新扩展中断号，如下图所示，任意一个 IO 中断源可以被配置是否使能、清除、以及被路由的目标处理器核中断脚。

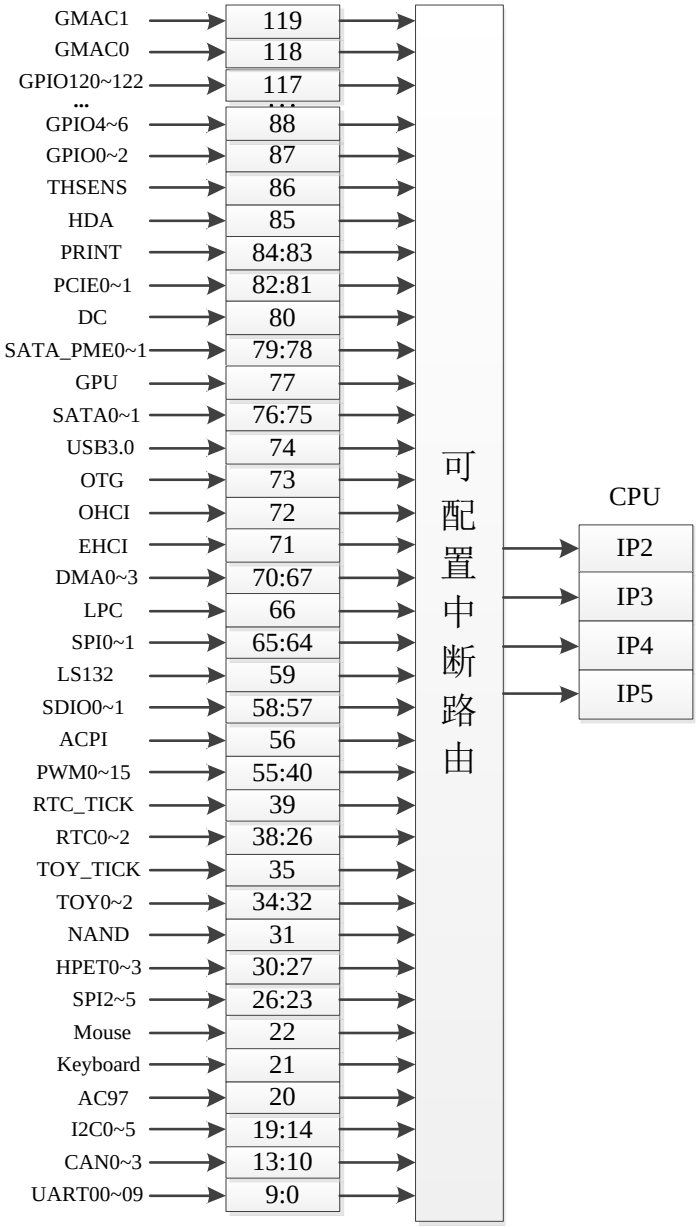


图5-5 龙芯 2K0500 处理器扩展 IO 中断路由示意图

2K0500 芯片在使用扩展 IO 中断前，需要使能“通用配置寄存器 0”中的扩展 IO 中断使能位。该寄存器配置地址为 0x1fe10100，对应配置位如下表：

位域	名称	访问	缺省值	描述
27	extioint_enable	RW	0x0	扩展IO中断使能位，高电平有效

扩展 IO 中断相关配置寄存器与传统中断寄存器类似，都是以位的形式对相应的中断线进行控制，中断控制位连接及属性配置见表 5-100。扩展中断配置寄存器主要有三种寄存器：扩展中断使能 EXTINT_IEN、扩展中断状态 EXTINT_ISR 和扩展中断清除 EXTINT_ICLR。EXTINT_IEN 设置扩展中断使能，相应寄存器写 1 的位对应的中断被使能。EXTINT_ISR 扩展中断状态，寄存器相应位为 1 表示对应 IO 中断有效(注：该状态不依赖于中断使能位是否被置起)。EXTINT_ICLR 扩展中断清除寄存器，与扩展中断状态寄存器共用，对应位写 1 清除

对应中断状态。处理器核增加扩展中断状态 CORE_EXTISR 和中断清除寄存器 CORE_EXTICLR，CORE_EXTISR 处理器核中断状态寄存器，当对应扩展 IO 中断使能位有效且 IO 中断有效时，该寄存器对应中断状态位被置起，表示对应 IO 中断可被处理器核接收。CORE_EXTICLR 中断清除寄存器与状态寄存器共用，对应位写 1 清除对应中断状态。

此外，扩展 IO 中断增加路由配置寄存器 EXTINT_MAP，该寄存器与传统 IO 中断路由类似，可将所有扩展 IO 中断按组分类路由至处理器核中断引脚向量，对应 CP0_Status 寄存器的 IP2 到 IP5。

表5-99 芯片扩展IO中断寄存器列表

地址	名称	描述
0x1fe11600	EXTINT_IEN0	扩展IO中断使能低64位[63:0]配置寄存器
0x1fe11608	EXTINT_IEN1	扩展IO中断使能高64位[127:64]配置寄存器
0x1fe11700	EXTINT_ISR0	扩展IO中断状态低64位[63:0]配置寄存器
0x1fe11708	EXTINT_ISR1	扩展IO中断状态高64位[127:64]配置寄存器
0x1fe11700	EXTINT_ICLR0	扩展IO中断清除低64位[63:0]配置寄存器
0x1fe11708	EXTINT_ICLR1	扩展IO中断清除高64位[127:64]配置寄存器
0x1fe11800	CORE_EXTISR0	处理器核扩展中断状态低64位[63:0]配置寄存器
0x1fe11808	CORE_EXTISR1	处理器核扩展中断状态高64位[127:64]配置寄存器
0x1fe11800	CORE_EXTICLR0	处理器核扩展中断清除低64位[63:0]配置寄存器
0x1fe11808	CORE_EXTICLR1	处理器核扩展中断清除高64位[127:64]配置寄存器
0x1fe11148	EXT_IOI_ACK	扩展IO中断处理完成握手配置寄存器
0x1fe114c0	EXTINT_MAP	扩展IO中断路由配置寄存器

2K0500 芯片扩展 IO 中断控制寄存器对应中断号及寄存器属性如下表所示：

表5-100 扩展 IO 中断控制寄存器属性

扩展 IO 中断号	访问属性/缺省值					
	EXTINT_IEN	EXTINT_ISR	EXTINT_ICLR	CORE_EXTISR	CORE_EXTICLR	中断源
0~9	RW / 0	R / 0	W / 0	R / 0	W / 0	UART0~9
10~13	RW / 0	R / 0	W / 0	R / 0	W / 0	CAN0~3
14~19	RW / 0	R / 0	W / 0	R / 0	W / 0	I2C0~5
20	RW / 0	R / 0	W / 0	R / 0	W / 0	AC97
21	RW / 0	R / 0	W / 0	R / 0	W / 0	Keyboard
22	RW / 0	R / 0	W / 0	R / 0	W / 0	Mouse
23~26	RW / 0	R / 0	W / 0	R / 0	W / 0	SPI2~5
27~30	RW / 0	R / 0	W / 0	R / 0	W / 0	HPET0~3

31	RW / 0	R / 0	W / 0	R / 0	W / 0	NAND
32~34	RW / 0	R / 0	W / 0	R / 0	W / 0	TOY0~2
35	RW / 0	R / 0	W / 0	R / 0	W / 0	TOY_TICK
36~38	RW / 0	R / 0	W / 0	R / 0	W / 0	RTC0~2
39	RW / 0	R / 0	W / 0	R / 0	W / 0	RTC_TICK
40~55	RW / 0	R / 0	W / 0	R / 0	W / 0	PWM0~15
56	RW / 0	R / 0	W / 0	R / 0	W / 0	ACPI
57~58	RW / 0	R / 0	W / 0	R / 0	W / 0	SDIO0~1
59	RW / 0	R / 0	W / 0	R / 0	W / 0	LA132
64~65	RW / 0	R / 0	W / 0	R / 0	W / 0	SPI0~1
66	RW / 0	R / 0	W / 0	R / 0	W / 0	LPC
67~70	RW / 0	R / 0	W / 0	R / 0	W / 0	DMA0~3
71	RW / 0	R / 0	W / 0	R / 0	W / 0	EHCI
72	RW / 0	R / 0	W / 0	R / 0	W / 0	OHCI
73	RW / 0	R / 0	W / 0	R / 0	W / 0	OTG
74	RW / 0	R / 0	W / 0	R / 0	W / 0	USB3.0
75~76	RW / 0	R / 0	W / 0	R / 0	W / 0	SATA0~1
77	RW / 0	R / 0	W / 0	R / 0	W / 0	GPU
78~79	RW / 0	R / 0	W / 0	R / 0	W / 0	SATA_PME0~1
80	RW / 0	R / 0	W / 0	R / 0	W / 0	DC
81~82	RW / 0	R / 0	W / 0	R / 0	W / 0	PCIE0~1
83~84	RW / 0	R / 0	W / 0	R / 0	W / 0	PRINT
85	RW / 0	R / 0	W / 0	R / 0	W / 0	HDA
86	RW / 0	R / 0	W / 0	R / 0	W / 0	THSENS
87~117	RW / 0	R / 0	W / 0	R / 0	W / 0	GPIO0~2
	RW / 0	R / 0	W / 0	R / 0	W / 0	GPIO4~6
	RW / 0	R / 0	W / 0	R / 0	W / 0	GPIO120~122
118	RW / 0	R / 0	W / 0	R / 0	W / 0	GMAC0
119	RW / 0	R / 0	W / 0	R / 0	W / 0	GMAC1

龙芯 2K0500 扩展 IO 中断源路由方式与传统 IO 中断类似，所有扩展 IO 中断源可以分组进行选择路由到处理器核中断 INT0 到 INT3 中的任意一个。128 个扩展 IO 中断源中每 32 个分为一组，每组对应一个 8 位的路由控制器，其格式和地址如下表所示。路由寄存器采用向量的方式进行路由选择，如 0x04 表示路由到处理器的 INT2 上。

表5-101 中断路由寄存器的说明

位域	说　明
3:0	路由的处理器核中断引脚向量号
7:4	保留

表5-102 中断路由寄存器地址

名称	地址偏移	描述
EXTINT_MAP0	0x1fe114c0	EXT_IOI[31:0]对应的处理器核中断引脚向量路由配置
EXTINT_MAP1	0x1fe114c1	EXT_IOI[63:32]对应的处理器核中断引脚向量路由配置
EXTINT_MAP2	0x1fe114c2	EXT_IOI[95:64]对应的处理器核中断引脚向量路由配置
EXTINT_MAP3	0x1fe114c3	EXT_IOI[127:96]对应的处理器核中断引脚向量路由配置

6 DDR3 控制器

龙芯 2K0500 处理器内部集成的 DDR3 SDRAM 内存控制器的设计遵守 DDR3 SDRAM 的行业标准（JESD79-2 和 JESD79-3），所实现的所有内存读/写操作都遵守 JESD79-2 及 JESD79-3 的规定。

6.1 访问地址

DDR3 控制器包括两个地址空间，分别如下：

表6-1 内存控制器地址空间分配

起始地址	结束名称	名称	说明
0x0FF0_0000	0x0FFF_FFFF	配置空间	当 mc_default_reg =1 时， 或 mc_default_reg = 0 且 mc_disable_reg = 0 时，为配置空间。 其它情况下为内存空间
其它		内存空间	使用 X2 的窗口配置路由至 DDR 的所有地址

具体的 mc_default_reg 和 mc_disable_reg 配置请参考 5.5.1 通用配置寄存器 0。

6.2 功能概述

龙芯 2K0500 处理器支持最大 2 个 CS（由 2 个片选信号实现），一共含有 19 位的地址总线（即：16 位的行列地址总线和 3 位的逻辑 Bank 总线）。

在具体选择使用不同内存芯片类型时，可以调整 DDR3 控制器参数设置进行支持。其中，支持的最大片选（CS_n）数为 2，行地址（RAS_n）数为 16，列地址（CAS_n）数为 16，逻辑体地址（BANK_n）数为 3。

CPU 发送的内存请求物理地址可以根据控制器内部不同的配置进行多种不同的地址映射。

龙芯 2K0500 处理器中内存控制器具有如下特征：

1. 接口上命令、读写数据全流水操作
2. 内存命令合并、排序提高整体带宽
3. 配置寄存器读写端口，可以修改内存设备的基本参数
4. 内建动态延迟补偿电路（DCC），用于数据的可靠发送和接收
5. 支持 133-533MHZ 工作频率

6.3 DDR3 控制器寄存器

由于系统中可能使用不同类型的 DDR3 SDRAM，因此，在系统上电复位以后，需要对 DDR3 SDRAM 进行配置。在 JESD79-2B 和 JESD79-3 中规定了详细的配置操作和配置过程，在没有完成 DDR3 的内存初始化操作之前，DDR3 不可用。内存初始化操作执行顺序如下：

1） 系统复位，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。

3） 向配置寄存器地址发写指令，配置所有 DDR3 配置寄存器。所有寄存器都必须正确配置才可以正常工作。

4） 配置结束后内存控制器将自动对内存发起初始化指令。

在龙芯 2K0500 处理器设计中，DDR3 SDRAM 的配置在系统主板初始化完成以后，需要使用内存之前，进行内存类型的配置。具体的配置操作是对物理地址 0x0ff0 0000 相对应的配置寄存器写入相应的配置参数。一个寄存器可能会包括多个、一个、部分参数的数据。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），具体的配置可以根据实际情况再决定：

表6-2 DDR3 SDRAM配置参数寄存器

	63:56	55:48	47:40	39:32	31:24	23:16	15:8	7:0
0x000	Dll_value_0(RD)/Dll_adj_cnt/dll_sync_disable/dll_close_disable		Dll_value_ck(RD)		Dll_init_done(RD)		Version(RD)	
0x008	Dll_value_4(RD)		Dll_value_3(RD)		Dll_value_2(RD)		Dll_value_1(RD)/capability(RD)	
0x010	Dll_value_8(RD)		Dll_value_7(RD)		Dll_value_6(RD)		Dll_value_5(RD)	
0x018	Dll_ck_3	Dll_ck_2	Dll_ck_1	Dll_ck_0	Dll_increment	Dll_start_point	Dll_bypass	Init_start
0x020	Dq_oe_end_0	Dq_oe_begin_0	Dq_stop_edge_0	Dq_start_edge_0	Rddata_delay_0	Rddqs_lt_half_0	Wrdqs_lt_half_0	Wrdq_lt_half_0
0x028	Rd_oe_end_0	Rd_oe_begin_0	Rd_stop_edge_0	Rd_start_edge_0	Dqs_oe_end_0	Dqs_oe_begin_0	Dqs_stop_edge_0	Dqs_start_edge_0
0x030	Enzi_end_0	Enzi_begin_0	Wrclk_sel_0	Wrdq_clkdelay_0	Odt_oe_end_0	Odt_oe_begin_0	Odt_stop_edge_0	Odt_start_edge_0
0x038	Enzi_stop_0	Enzi_start_0	Dll_oe_shorten_0	Dll_rddqs_n_0	Dll_rddqs_p_0	Dll_wrdqs_0	Dll_wrdata_0	Dll_gate_0
0x040	Dq_oe_end_1	Dq_oe_begin_1	Dq_stop_edge_1	Dq_start_edge_1	Rddata_delay_1	Rddqs_lt_half_1	Wrdqs_lt_half_1	Wrdq_lt_half_1
0x048	Rd_oe_end_1	Rd_oe_begin_1	Rd_stop_edge_1	Rd_start_edge_1	Dqs_oe_end_1	Dqs_oe_begin_1	Dqs_stop_edge_1	Dqs_start_edge_1
0x050	Enzi_end_1	Enzi_begin_1	Wrclk_sel_1	Wrdq_clkdelay_1	Odt_oe_end_1	Odt_oe_begin_1	Odt_stop_edge_1	Odt_start_edge_1
0x058	Enzi_stop_1	Enzi_start_1	Dll_oe_shorten_1	Dll_rddqs_n_1	Dll_rddqs_p_1	Dll_wrdqs_1	Dll_wrdata_1	Dll_gate_1
0x060	Dq_oe_end_2	Dq_oe_begin_2	Dq_stop_edge_2	Dq_start_edge_2	Rddata_delay_2	Rddqs_lt_half_2	Wrdqs_lt_half_2	Wrdq_lt_half_2
0x068	Rd_oe_end_2	Rd_oe_begin_2	Rd_stop_edge_2	Rd_start_edge_2	Dqs_oe_end_2	Dqs_oe_begin_2	Dqs_stop_edge_2	Dqs_start_edge_2
0x070	Enzi_end_2	Enzi_begin_2	Wrclk_sel_2	Wrdq_clkdelay_2	Odt_oe_end_2	Odt_oe_begin_2	Odt_stop_edge_2	Odt_start_edge_2
0x078	Enzi_stop_2	Enzi_start_2	Dll_oe_shorten_2	Dll_rddqs_n_2	Dll_rddqs_p_2	Dll_wrdqs_2	Dll_wrdata_2	Dll_gate_2
0x080	Dq_oe_end_3	Dq_oe_begin_3	Dq_stop_edge_3	Dq_start_edge_3	Rddata_delay_3	Rddqs_lt_half_3	Wrdqs_lt_half_3	Wrdq_lt_half_3
0x088	Rd_oe_end_3	Rd_oe_begin_3	Rd_stop_edge_3	Rd_start_edge_3	Dqs_oe_end_3	Dqs_oe_begin_3	Dqs_stop_edge_3	Dqs_start_edge_3
0x090	Enzi_end_3	Enzi_begin_3	Wrclk_sel_3	Wrdq_clkdelay_3	Odt_oe_end_3	Odt_oe_begin_3	Odt_stop_edge_3	Odt_start_edge_3
0x098	Enzi_stop_3	Enzi_start_3	Dll_oe_shorten_3	Dll_rddqs_n_3	Dll_rddqs_p_3	Dll_wrdqs_3	Dll_wrdata_3	Dll_gate_3
0x0A0	Dq_oe_end_4	Dq_oe_begin_4	Dq_stop_edge_4	Dq_start_edge_4	Rddata_delay_4	Rddqs_lt_half_4	Wrdqs_lt_half_4	Wrdq_lt_half_4
0x0A8	Rd_oe_end_4	Rd_oe_begin_4	Rd_stop_edge_4	Rd_start_edge_4	Dqs_oe_end_4	Dqs_oe_begin_4	Dqs_stop_edge_4	Dqs_start_edge_4
0x0B0	Enzi_end_4	Enzi_begin_4	Wrclk_sel_4	Wrdq_clkdelay_4	Odt_oe_end_4	Odt_oe_begin_4	Odt_stop_edge_4	Odt_start_edge_4
0x0B8	Enzi_stop_4	Enzi_start_4	Dll_oe_shorten_4	Dll_rddqs_n_4	Dll_rddqs_p_4	Dll_wrdqs_4	Dll_wrdata_4	Dll_gate_4
0x0C0	Dq_oe_end_5	Dq_oe_begin_5	Dq_stop_edge_5	Dq_start_edge_5	Rddata_delay_5	Rddqs_lt_half_5	Wrdqs_lt_half_5	Wrdq_lt_half_5
0x0C8	Rd_oe_end_5	Rd_oe_begin_5	Rd_stop_edge_5	Rd_start_edge_5	Dqs_oe_end_5	Dqs_oe_begin_5	Dqs_stop_edge_5	Dqs_start_edge_5
0x0D0	Enzi_end_5	Enzi_begin_5	Wrclk_sel_5	Wrdq_clkdelay_5	Odt_oe_end_5	Odt_oe_begin_5	Odt_stop_edge_5	Odt_start_edge_5
0x0D8	Enzi_stop_5	Enzi_start_5	Dll_oe_shorten_5	Dll_rddqs_n_5	Dll_rddqs_p_5	Dll_wrdqs_5	Dll_wrdata_5	Dll_gate_5

	63:56	55:48	47:40	39:32	31:24	23:16	15:8	7:0
0x0E0	Dq_oe_end_6	Dq_oe_begin_6	Dq_stop_edge_6	Dq_start_edge_6	Rddata_delay_6	Rddqs_lt_half_6	Wrdqs_lt_half_6	Wrdq_lt_half_6
0x0E8	Rd_oe_end_6	Rd_oe_begin_6	Rd_stop_edge_6	Rd_start_edge_6	Dqs_oe_end_6	Dqs_oe_begin_6	Dqs_stop_edge_6	Dqs_start_edge_6
0x0F0	Enzi_end_6	Enzi_begin_6	Wrclk_sel_6	Wrdq_clkdelay_6	Odt_oe_end_6	Odt_oe_begin_6	Odt_stop_edge_6	Odt_start_edge_6
0x0F8	Enzi_stop_6	Enzi_start_6	Dll_oe_shorten_6	Dll_rddqs_n_6	Dll_rddqs_p_6	Dll_wrdqs_6	Dll_wrdata_6	Dll_gate_6
0x100	Dq_oe_end_7	Dq_oe_begin_7	Dq_stop_edge_7	Dq_start_edge_7	Rddata_delay_7	Rddqs_lt_half_7	Wrdqs_lt_half_7	Wrdq_lt_half_7
0x108	Rd_oe_end_7	Rd_oe_begin_7	Rd_stop_edge_7	Rd_start_edge_7	Dqs_oe_end_7	Dqs_oe_begin_7	Dqs_stop_edge_7	Dqs_start_edge_7
0x110	Enzi_end_7	Enzi_begin_7	Wrclk_sel_7	Wrdq_clkdelay_7	Odt_oe_end_7	Odt_oe_begin_7	Odt_stop_edge_7	Odt_start_edge_7
0x118	Enzi_stop_7	Enzi_start_7	Dll_oe_shorten_7	Dll_rddqs_n_7	Dll_rddqs_p_7	Dll_wrdqs_7	Dll_wrdata_7	Dll_gate_7
0x120	Dq_oe_end_8	Dq_oe_begin_8	Dq_stop_edge_8	Dq_start_edge_8	Rddata_delay_8	Rddqs_lt_half_8	Wrdqs_lt_half_8	Wrdq_lt_half_8
0x128	Rd_oe_end_8	Rd_oe_begin_8	Rd_stop_edge_8	Rd_start_edge_8	Dqs_oe_end_8	Dqs_oe_begin_8	Dqs_stop_edge_8	Dqs_start_edge_8
0x130	Enzi_end_8	Enzi_begin_8	Wrclk_sel_8	Wrdq_clkdelay_8	Odt_oe_end_8	Odt_oe_begin_8	Odt_stop_edge_8	Odt_start_edge_8
0x138	Enzi_stop_8	Enzi_start_8	Dll_oe_shorten_8	Dll_rddqs_n_8	Dll_rddqs_p_8	Dll_wrdqs_8	Dll_wrdata_8	Dll_gate_8
0x140	Pad_ocd_clk	Pad_ocd_ctl	Pad_ocd_dqs	Pad_ocd_dq	Pad_enzi		Pad_en_ctl	Pad_en_clk
0x148	Pad_adj_code_dqs	Pad_code_dqs	Pad_adj_code_dq	Pad_code_dq		Pad_vref_internal	Pad_odt_se	Pad_modezi1v8
0x150		Pad_reset_po	Pad_adj_code_clk	Pad_code_lk	Pad_adj_code_cmd	Pad_code_cmd	Pad_adj_code_addr	Pad_code_addr
0x158		Pad_comp_code_o	Pad_comp_okn		Pad_comp_code_i	Pad_comp_mode	Pad_comp_tm	Pad_comp_pd
0x160	Rdfifo_empty(RD)		Overflow(RD)		Dram_init(RD)	Rdfifo_valid	Cmd_timming	Ddr3_mode
0x168	Ba_xor_row_offset	Addr_mirror	Cmd_delay	Burst_length	Bank/Cs_resync	Cs_zq	Cs_mrs	Cs_enable
0x170	Odt_wr_cs_map		Odt_wr_length	Odt_wr_delay	Odt_rd_cs_map		Odt_rd_length	Odt_rd_delay
0x178								
0x180	Lvl_resp_0(RD)	Lvl_done(RD)	Lvl_ready(RD)		Lvl_cs	tLVL_DELAY	Lvl_req(WR)	Lvl_mode
0x188	Lvl_resp_8(RD)	Lvl_resp_7(RD)	Lvl_resp_6(RD)	Lvl_resp_5(RD)	Lvl_resp_4(RD)	Lvl_resp_3(RD)	Lvl_resp_2(RD)	Lvl_resp_1(RD)
0x190	Cmd_a		Cmd_ba	Cmd_cmd	Cmd_cs	Status_cmd(RD)	Cmd_req(WR)	Command_mode
0x198			Status_sref(RD)	Srefresh_req	Pre_all_done(RD)	Pre_all_req(RD)	Mrs_done(RD)	Mrs_req(WR)
0x1A0	Mr_3_cs_0		Mr_2_cs_0		Mr_1_cs_0		Mr_0_cs_0	
0x1A8	Mr_3_cs_1		Mr_2_cs_1		Mr_1_cs_1		Mr_0_cs_1	
0x1B0	Mr_3_cs_2		Mr_2_cs_2		Mr_1_cs_2		Mr_0_cs_2	
0x1B8	Mr_3_cs_3		Mr_2_cs_3		Mr_1_cs_3		Mr_0_cs_3	
0x1C0	tRESET	tCKE	tXPR	tMOD	tZQCL	tZQ_CMD	tWLDQSEN	tRDDATA
0x1C8	tFAW	tRRD	tRCD	tRP	tREF	tRFC	tZQCS	tZQperiod
0x1D0	tODTL	tXSRD	tPHY_RDLAT	tPHY_WRLAT	tRAS_max			tRAS_min
0x1D8	tXPDLL	tXP	tWR	tRTP	tRL	tWL	tCCD	tWTR
0x1E0	tW2R_diffCS	tW2W_diffCS	tR2P_sameBA	tW2P_sameBA	tR2R_sameBA	tR2W_sameBA	tW2R_sameBA	tW2W_sameBA
0x1E8	tR2R_diffCS	tR2W_diffCS	tR2P_sameCS	tW2P_sameCS	tR2R_sameCS	tR2W_sameCS	tW2R_sameCS	tW2W_sameCS
0x1F0	Power_up	Age_step	tCPDED	Cs_map	Bs_config	Nc	Pr_r2w	Placement_en
0x1F8	Hw_pd_3	Hw_pd_2	Hw_pd_1	Hw_pd_0	Credit_16	Credit_32	Credit_64	Selection_en
0x200	Cmdq_age_16		Cmdq_age_32		Cmdq_age_64		tCKESR	tRDPDEN
0x208	Wfifo_age		Rfifo_age		Power_stat3	Power_stat2	Power_stat1	Power_stat0
0x210	Active_age		Cs_place_0	Addr_win_0	Cs_diff_0	Row_diff_0	Ba_diff_0	Col_diff_0
0x218	Fastpd_age		Cs_place_1	Addr_win_1	Cs_diff_1	Row_diff_1	Ba_diff_1	Col_diff_1
0x220	Slowpd_age		Cs_place_2	Addr_win_2	Cs_diff_2	Row_diff_2	Ba_diff_2	Col_diff_2
0x228	Selfref_age		Cs_place_3	Addr_win_3	Cs_diff_3	Row_diff_3	Ba_diff_3	Col_diff_3

	63:56	55:48	47:40	39:32	31:24	23:16	15:8	7:0
0x230	Win_mask_0				Win_base_0			
0x238	Win_mask_1				Win_base_1			
0x240	Win_mask_2				Win_base_2			
0x248	Win_mask_3				Win_base_3			
0x250		Cmd_monitor	Axi_monitor		Ecc_code(RD)	Ecc_enable	Int_vector	Int_enable
0x258								
0x260	Ecc_addr(RD)							
0x268	Ecc_data(RD)							
0x270	Lpbk_ecc_mask(RD)	Prbs_init			Lpbk_error(RD)	Prbs_23	Lpbk_start	Lpbk_en
0x278	Lpbk_ecc(RD)		Lpbk_data_mask(RD)		Lpbk_correct(RD)		Lpbk_counter(RD)	
0x280	Lpbk_data_r(RD)							
0x288	Lpbk_data_f(RD)							
0x290	Axi0_bandwidth_w				Axi0_bandwidth_r			
0x298	Axi0_latency_w				Axi0_latency_r			
0x2A0	Axi1_bandwidth_w				Axi1_bandwidth_r			
0x2A8	Axi1_latency_w				Axi1_latency_r			
0x2B0	Axi2_bandwidth_w				Axi2_bandwidth_r			
0x2B8	Axi2_latency_w				Axi2_latency_r			
0x2C0	Axi3_bandwidth_w				Axi3_bandwidth_r			
0x2C8	Axi3_latency_w				Axi3_latency_r			
0x2D0	Axi4_bandwidth_w				Axi4_bandwidth_r			
0x2D8	Axi4_latency_w				Axi4_latency_r			
0x2E0	Cmdq0_bandwidth_w				Cmdq0_bandwidth_r			
0x2E8	Cmdq0_latency_w				Cmdq0_latency_r			
0x2F0	Cmdq1_bandwidth_w				Cmdq1_bandwidth_r			
0x2F8	Cmdq1_latency_w				Cmdq1_latency_r			
0x300	Cmdq2_bandwidth_w				Cmdq2_bandwidth_r			
0x308	Cmdq2_latency_w				Cmdq2_latency_r			
0x310	Cmdq3_bandwidth_w				Cmdq3_bandwidth_r			
0x318	Cmdq3_latency_w				Cmdq3_latency_r			
0x320	tRESYNC_length	tRESYNC_shift	tRESYNC_max	tRESYNC_min	Pre_predict		tXS	tREF_low
0x328								tRESYNC_delay
0x330	Stat_en	Rdbuffer_max	Retry	Wr_pkg_num	Rwq_rb	Stb_en	Addr_new	tRDQidle
0x338				Rd_fifo_depth	Retry_cnt			
0x340	tREFretention					Ref_num	tREF_IDLE	Ref_sch_en
0x348								
0x350	Lpbk_data_en							
0x358						Lpbk_ecc_mask_en	Lpbk_ecc_en	Lpbk_data_mask_en
0x360			Int_ecc_cnt_fatal	Int_ecc_cnt_error	Ecc_cnt_cs_3	Ecc_cnt_cs_2	Ecc_cnt_cs_1	Ecc_cnt_cs_0
0x368								
0x370	Prior_age3		Prior_age2		Prior_age1		Prior_age_0	

	63:56	55:48	47:40	39:32	31:24	23:16	15:8	7:0
0x378							No_dead_inorder	Row_hit_place
0x380	Zq_cnt_1				Zq_cnt_0			
0x388	Zq_cnt_3				Zq_cnt_2			
0x390								Nc16_map
0x398								

7 PCIE 控制器

7.1 使用说明

龙芯 2K0500 在 PCIE 总线上既可以充当 RC（root complex）也可以充当 EP（end point）[1]。在充当 RC 时，龙芯 2K0500 的 PCIE 接口可以充当 2 个独立的 X1 的 PCIE 端口；在充当 EP 时，龙芯 2K0500 的 PCIE 接口可充当 1 个 X1 的 PCIE 接口。

图 7-1 为龙芯 2K0500 的 PCIE 控制器结构示意。龙芯 2K0500 的 PCIE 控制器包含 0～1 号，共 2 个 PCIE 端口。0～1 号端口均能以 X1 的方式工作，作为 RC 时 0～1 号端口可用，作为 EP 使用时，0～1 号端口均可用。每个 PCIE 端口均有自己独立的 PCI 地址空间。

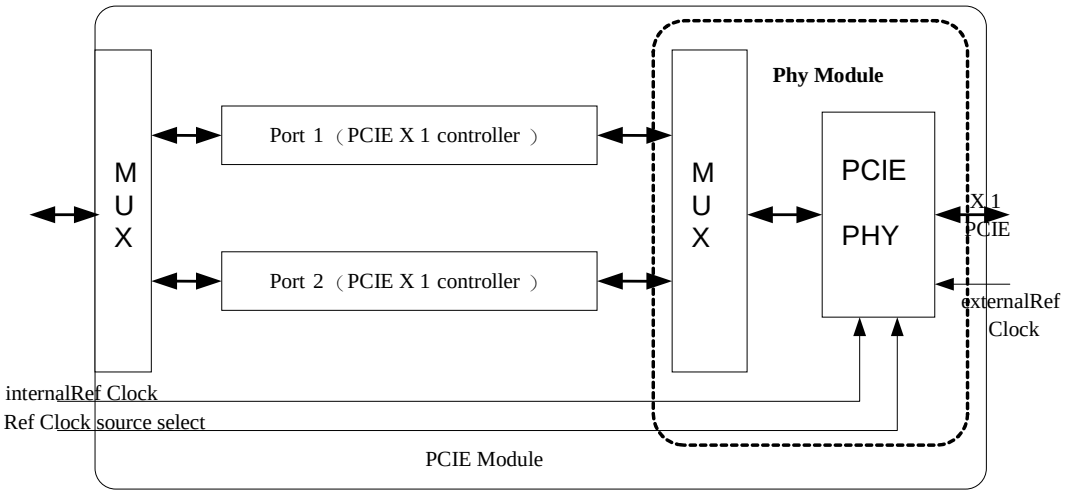


图 7-1 PCIE 控制器结构

7.2 访问地址

PCIE 控制器中每个端口有独立的配置头，其访问地址分别为：

设备	总线号	设备号	功能号	配置空间掩码	配置头访问首地址（64 位模式）	配置头访问首地址（32 位模式）	备注
PCIE Port0	0x0	0x0	0x0/1	0xFFF(64-bits) 0xFF(32-bits)	0xFE_0000_0000	0x1680_0000	TYPE1 类型配置头，仅有 32 位 BAR0，用于访问内部寄存器。对功能 1 的访问用于修改一些只读寄存器。
PCIE Port1	0x0	0x1	0x0/1	0xFFF(64-bits) 0xFF(32-bits)	0xFE_0000_0800	0x1680_0800	

[1] RC 相当于 PCI 总线上的 host bridge，EP 相当于 PCI 总线上的 device。管脚 LCD_D5/D16 分别控制 PCIE0/1，上拉时为 EP，下拉时为 RC。

PCIE 控制器内部寄存器的物理地址构成如下：

地址位	构成	备注
[63:12](64bits) / [31:8](32bits)	BAR_BASE	PCIE 控制器内 BAR0 配置
[11:00](64bits) / [7:0](32bits)	REG	内部寄存器地址

7.3 地址空间划分

龙芯 2K0500 处理器的 PCIE 控制器内有标准的 PCIE 配置头，因此 PCIE 控制器的内部寄存器以及其下游设备的地址空间都通过其配置头的信息来管理。配置头中地址相关的寄存器在 PCI 设备扫描时确定。

龙芯 2K0500 处理器的 PCIE 控制器既可以工作在 RC 模式下又可以工作在 EP 模式下，所以其配置头支持 TYPE0/TYPE1 类型。

每个 PCIE 端口作为 2K0500 中的独立设备，每个端口都包含一个 PCIE 配置头。对于每一个 PCIE 端口，其地址空间可以分为以下几部分：

配置头地址空间：该部分空间对应 PCIE 的配置头，通过配置请求来访问，最大 8KB(64 位)/ 256B(32 位)。其中低 4KB(64 位)/ 128B(32 位)通过将配置请求的 func 设为 0 来访问，用于访问标准配置头；高 4KB(64 位)/ 128B(32 位)通过将配置请求的 func 设为 1 来访问，用于改写标准配置头中的一些只读寄存器。

配置访问地址空间：该部分地址空间用于通过配置请求访问 PCIE 控制器的下游设备配置头信息。根据下游设备的 Bus 号，由 PCIE 控制器决定发送 TYPE0 类型还是 TYPE1 类型的配置访问。

以上两个地址空间的地址由配置地址空间基地址、BUS 号、设备号、功能号以及寄存器偏移地址计算得出，访问以字为单位。

PCIE 控制器内部寄存器空间：该部分地址空间用于访问 PCIE 控制器的内部寄存器。这些寄存器用于控制 PCIE 控制器的行为和特性，与 PCIE 配置头空间属于两个地址空间。该地址空间为 MEM 类型，32 位地址空间，大小为 4KB，基地址等于 32 位 BAR0 的值，该值在初始化时由 PCI 扫描软件分配。

MEM 地址空间：该部分地址空间包含了 PCIE 控制器下游设备的所有 MEM 地址空间。对于 32 位地址空间，由 PCIE 配置头的 memory base 和 memory limit 决定；对于 64 位地址空间，由 PCIE 配置头的 prefetchable memory base（组合 upper 32bits）和 prefetchable memory limit（组合 upper 32bits）决定。该段地址空间由 PCIE 配置头的 command 寄存器 bit1 位来使能控制。

IO 地址空间：该部分地址空间包含了 PCIE 控制器下游设备的所有 IO 地址空间。由 PCIE 配置头的 io base（组合 upper 16bits）和 io limit（组合 upper 16bits）决定。该

段地址空间由 PCIE 配置头的 command 寄存器 bit0 位来使能控制。

对于 MEM 地址空间和 IO 地址空间来说，如果在 X1 工作模式下，某个 X1 端口下游没有连接设备，通过设置 command 寄存器的 bit0 和 bit1 为 0 即可禁用其 MEM 和 IO 地址空间。

7.3.1 RC 模式下的地址空间划分

作为 RC 时所占用的空间分为 2 部分：cfg&ctrl、mem。

Mem 从 0x40000000 起始，大小为 1GB。这段空间处理器的物理地址和 PCIE 端口上的 PCI 地址相同。这段空间被用于访问需要映射较大 MEM 空间的设备。这段空间可以由 2 个 X1 端口分享。

表 7-1 MEM 地址空间分配

地址范围	大小	所属端口号
0x40000000~0x7fffffff	1GB	port0/1

Cfg&ctrl 空间从 0x16000000 起始，大小为 16MB。这个空间由 0～1 号 2 个 PCIE 端口分享。

龙芯 2K0500 处理器 PCIE 模块的内部寄存器在 cfg&ctrl 段中。龙芯 2K0500 的 PCIE 控制器内部的寄存器包含 3 部分：每个端口都有的 PCIE 头控制寄存器、每个端口都有的端口控制寄存器、phy 控制寄存器。其中对 phy 控制寄存器的访问以字节访问的形式进行，而其它的寄存器则以 32 位的 word 方式进行。

表 7-2 RC 模式下 cfg&ctrl 空间划分

起始地址	大小	用途	备注
0x16000000	4kB	对PCIE控制器port0的内部总线连接配置访问	MEM 地址类型
0x16001000	4kB	对PCIE控制器port1的内部总线连接配置访问	MEM 地址类型
0x16400000	4MB	对PCIE控制器PCII/O访问	IO 地址类型
0xfd_fc000000 (64 位模式)	32M		
0x16800000		对PCIE控制器port0/1本地的PCIE配置头空间进行访问	
0xfe_00000000 (64 位模式)			

7.3.2 EP 模式下的地址空间划分

在 EP 模式下，除了 cfg&ctrl、mem 这 2 段空间外的所有访问都被直接返回全‘F’。Mem 在 EP 模式下起始地址与 RC 模式相同，从 0x40000000 起始，大小为 1GB。这段空间处理器的物理地址和 PCIE 端口上的 PCI 地址相同。这段空间可以由 2 个 X1 端口分享，每个 PORT 端口 mem 空间分别为 512MB。

表 7-3 MEM 地址空间分配

地址范围	大小	所属端口号
0x40000000~0x5fffffff	512MB	port0

0x60000000~0x7fffffff	512MB	port1

cfg&ctrl 段地址的使用由表 7-4 给出。

表 7-4 EP 模式下 cfg&ctrl 空间划分

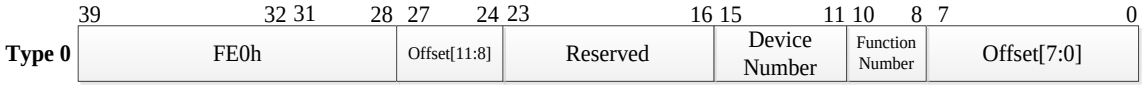
起始地址	大小	用途	备注
0x16000000	4kB	对PCIE控制器port0的内部总线连接配置访问	MEM 地址类型
0x16001000	4kB	对PCIE控制器port1的内部总线连接配置访问	MEM 地址类型
0x16400000	4MB	保留	
0x16800000		对PCIE控制器port0/1本地的PCIE配置头空间进行访问	
0xfe_00000000 (64 位模式)			

7.4 软件编程指南

本节给出龙芯 2K0500 的 PCIE 控制器的软件编程指南。

7.4.1 PCIE 配置头访问

基于前文对配置请求的介绍，对 PCIE 配置头的访问为 Type0 的访问，其格式为：



PCIE 各个 Port 的设备号（device number）分别为 0x0，0x1。根据设备号及所要访问的寄存器偏移地址(offset)即可得到对应寄存器的物理地址。功能号 Function Number 为 0 时访问配置头内寄存器，功能号 Function Number 为 1 时可以访问配置头空间内影子寄存器，用于改写配置头内的只读寄存器。

7.4.2 PCIE 链路建立(Linkup)

链路建立流程如下：

(1) 通过配置访问设置 Gen2 Control Register 寄存器中（0x80c）Directed Speed Change 为 1，PHY Tx Swing 为 0。注意配置地址格式，因为 offset 大于 8 位地址，需将高 4 位填到 bit24-bit27；

(2) 通过配置访问配置好 PCIE 的 BAR0 寄存器；

(3) 根据 BAR0 中配的地址通过 MEM 访问设置 PCIE 控制器内部寄存器 app_ltssm_enable(0x0)为 1，开始 link training 过程；

(4) 等待内部寄存器 Xmlh_ltssm_state(0xc)为 0x11；

(5) Linkup 成功。

7.4.3 TYPE1 类型配置访问

TYPE1 类型配置请求地址格式如下：

	39 32	31 28	27 24	23 16	15 11	10 8	7 0
Type 1	FE1h		Offset[11:8]	Bus Number	Device Number	Function Number	Offset[7:0]

发送 TYPE1 类型配置访问之前需要设置配置头的 Primary Bus Number、Secondary Bus Numer 和 Subordinate Bus Number。然后直接按照 TYPE1 地址格式发送读写请求即可，PCIE 控制器根据地址中的 Bus Number 与所配置的 Secondary Bus Numer 和 Subordinate Bus Number 决定发出 TYPE0 类型还是 TYPE1 类型的配置请求。

如果 Bus Number==Secondary Bus Numer，则发出 TYPE0 类型的配置请求。

如果 Bus Number> Secondary Bus Numer 并且 Bus Number <= Subordinate Bus Number，则发出 TYPE1 类型的配置请求。

7.4.4 PCIE PHY 配置方法

PCIE PHY 内部有一些可配置的寄存器，对这些寄存器的访问通过读写芯片配置寄存器中 PCIE PHY 配置寄存器来实现，具体步骤如下：

对于写请求

1. 设置 phy_cfg_disable 为 0x0
2. 设置所要访问的寄存器地址 phy_cfg_addr
3. 将要写的数据写入 phy_cfg_data
4. 设置寄存器 phy_cfg_rw 为 1，开始写内部寄存器
5. 等待 phy_cfg_done 为 1
6. 写数据完成。

对于读请求

7. 设置 phy_cfg_disable 为 0x0
8. 设置所要访问的寄存器地址 phy_cfg_addr
9. 设置寄存器 phy_cfg_rw 为 0，开始从内部寄存器读数
10. 等待 phy_cfg_done 为 1
11. 从 phy_cfg_data 中读出数据。

7.5 常用例程

本节给出龙芯 2K0500 的 PCIE 控制器的常用例程。当龙芯 2K0500 以 PCIE 总线上的 EP 方式工作时龙芯 2K0500 需要执行下面例程中的 pcie_link_init；而以方式 RC 工作时，龙芯 2K0500 需要先执行下面示例中的 pcie_link_init，再执行下面示例中的 pcie_header_init。随后，龙芯 2K0500 可以通过 cfg_device_read 和 cfg_device_write 这两个函数对端口 0 上的设备的 PCI Header 进行初始化。

```
unsigned int tmp_var;
unsigned int * cfg0_base = 0xfe00000000;
unsigned int * cfg1_base = 0xfe10000000;
unsigned int * mem_base = 0x9000000000000000;

void pcie_link_init(unsigned long bar,unsigned int dev_num, unsigned int func_num)
{
   unsigned long pcie_header_base = cfg0_base | (dev_num << 11) | (func_num <<8);
   unsigned long pcie_reg_base = mem_base + bar;
      // set port logic register of port 0
      // initiate speed change to PCIE Gen2 and set Tx to Low Swing
      tmp_var = *(volatile unsigned int *)(pcie_header_base + 0x80c);
      *(volatile unsigned int *)(pcie_header_base + 0x80c) = (tmp_var | 0x20000)&0xfffbffff;

      //start link training
      *(volatile unsigned int *)(pcie_reg_base) = 0xff200c;

      //wait link train end
      tmp_var = *(volatile unsigned int *)( pcie_reg_base +0xc);
      while((tmp_var&0x1f)!=0x11)
      {
            tmp_var = *(volatile unsigned int *)( pcie_reg_base +0xc);
      }
      printf("now PCIE port 0 link is start up\n");

}

void pcie_hot_reset(unsigned long bar)
{
      unsigned long pcie_reg_base = mem_base + bar;
      tmp_var = *(volatile unsigned int *)( pcie_reg_base);
      //enable soft reset
      *(volatile unsigned int *)( pcie_reg_base) = tmp_var |0x1000;
      //triger hot reset
      *(volatile unsigned int *)( pcie_reg_base +0x4) = 0x4;
}

void pcie_header_init(unsigned long bar,unsigned int dev_num, unsigned int func_num, unsigned
int io_base, unsigend int io_limit, unsigned int mem_base, unsigned int mem_limit, unsigned int
pref_mem_base, unsigned int pref_mem_limit, unsigned pref_mem_base_upper32, unsigned int
pref_mem_limit_upper32)
{//only used in RC mode
      unsigned long pcie_header_base = cfg0_base | (dev_num << 11) | (func_num <<8);
      unsigned long pcie_reg_base = mem_base + bar;
      //set master enable, io enable, mem enable, perr enable, serr enable
      *(volatile unsigned int *)( pcie_header_base + 0x4) = 0x147;

      //clear master abort, serr and perr status
//set IO space to be 16-bit address
//set 64 KB IO space: 0x0000 ~ 0xffff
      *(volatile unsigned int *)( pcie_header_base + 0x1c) = 0xf100f000;

      //set IO limit and IO base up 16 bit address
      *(volatile unsigned int *)( pcie_header_base + 0x30) = 0x0;

//set memory limit and memory base
```

```
        *(volatile unsigned int *)( pcie_header_base + 0x20) = 0x17f00000;

        //set prefetchable memory limit and base

        *(volatile unsigned int *)( pcie_header_base + 0x24) = 0x7ff04000;

        //set prefetchable base up 32 bit
        *(volatile unsigned int *)( pcie_header_base + 0x28) = pref_mem_base_upper32;
        //set prefetchable limit up 32 bit
        *(volatile unsigned int *)( pcie_header_base + 0x2c) = pref_mem_limit_upper32;

        //enable serr
        *(volatile unsigned int *)( pcie_header_base + 0x3c) = 0x20000;

        //enable system error on correctalbe error, non-fatal error and fatal error
        //enable PME interrupt
        *(volatile unsigned int *)( pcie_header_base + 0x8c) = 0xf;

        //enable correctalbe error, non-fatal error and fatal error
        *(volatile unsigned int *)( pcie_header_base + 0x12c) = 0x7;

        //enable ASPM L0s and L1

        *(volatile unsigned char *)( pcie_header_base + 0x80) = 0x3;
}
void cfg_device_read(
                                unsigned int bus_num,
                                unsigned int dev_num, unsigned int func_num,
unsigned int reg_id, unsigned int * read_data
                                )
{
unsigned long pcie_header_base = cfg1_base | (bus_num << 16) | (dev_num << 11) | (func_num
<<8);
            *(read_data) = *(volatile unsigned int *)( pcie_header_base + (reg_id<<2));
}
void cfg_device_write(
                                unsigned int bus_num,
                                unsigned int dev_num, unsigned int func_num,
unsigned int reg_id, unsigned int write_data
                                )
{
        unsigned long pcie_header_base = cfg1_base | (bus_num << 16) | (dev_num << 11) |
(func_num <<8);

        *(volatile unsigned int *)( pcie_header_base + (reg_id<<2)) = write_data;
}
```

8 PCI 控制器

龙芯2K0500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2K0500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2K0500作为PCI主桥时，龙芯2K0500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2K0500作为PCI设备时，龙芯2K0500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。

8.1 总体描述

龙芯2K0500作为PCI主桥时，龙芯2K0500访问PCI空间的地址划分如表8-1所示。当龙芯2K0500作为PCI设备工作时，龙芯2K0500对表 8-1 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 8-1 龙芯 2K0500 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
0x20000000	64MB	PCI mem space 1
0x24000000	64MB	PCI mem space 2
0x17000000	1MB	PCI IO空间
0x17100000	64KB	PCI设备配置空间
0x17110000	256B	PCI控制器的pciheader空间

当龙芯 2K0500 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 2K0500 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 2K0500 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 0x17100000 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 8-1。

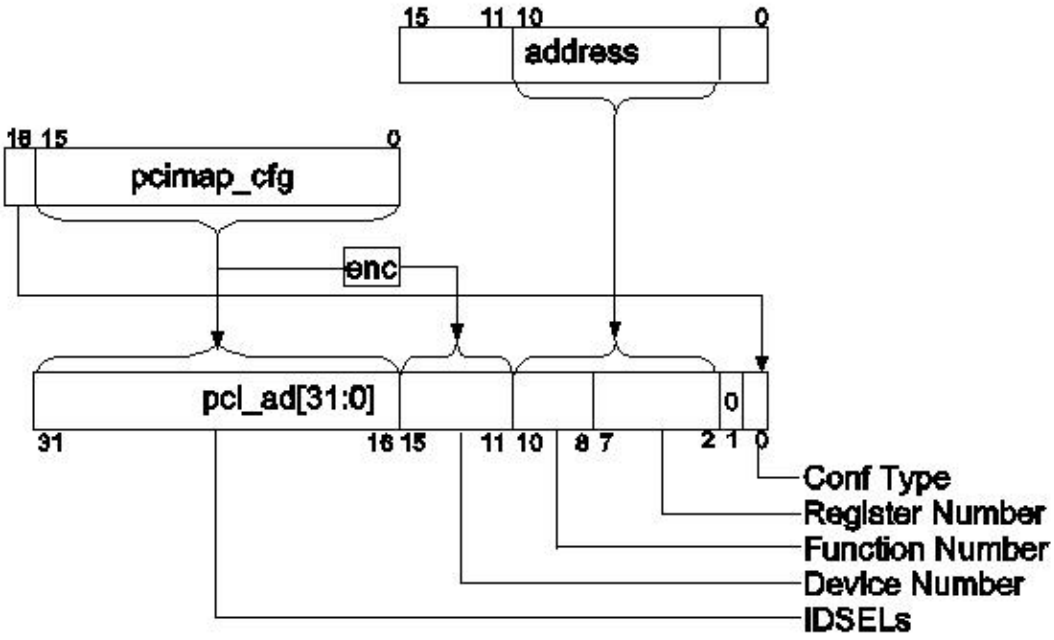


图8-1 PCI 设备配置读写总线地址生成机制

PCI 配置驱动程序事例简要描述：

```
#define pcibus_cfg_base 0xb7100000
#define pcimapcfg 0x1fe11120
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 2K0500 的 PCI 控制器的 pciheader 空间如表 8-2 所示。

表 8-2 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，0x40 开始的寄存器为龙芯 2K0500 自定义寄存器。龙芯 2K0500 的 pciheader 空间可以由龙芯 2K0500 以 0x17110000 为基址进行访问，也可以在龙芯 2K0500 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 2K0500 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。

8.2 寄存器描述

8.2.1 Command 寄存器

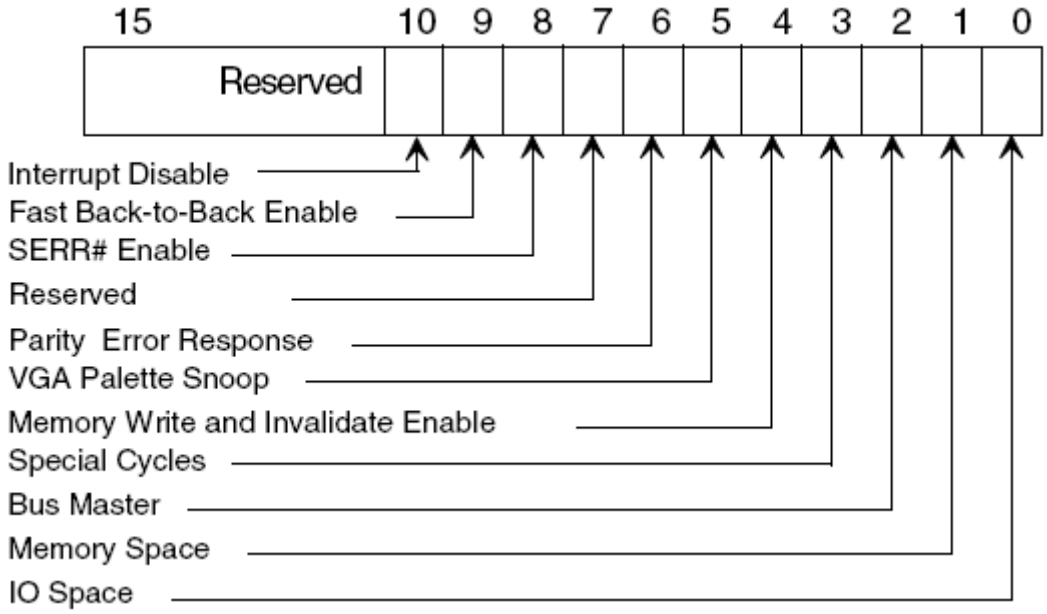


图8-2 Command register layout

Command 寄存器的格式如图 8-2 所示。在龙芯 2K0500 中，Command 寄存器只有 0、1、2、6、8 这 5 位可以进行修改。龙芯 2K0500 的 PCI 控制器不支持 Fast Back-to-Back 功能。龙芯 2K0500 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）。

8.2.2 Base Address Register 0～5

龙芯 2K0500 有 3 个用于将芯片所控制资源映射到 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 2K0500 内部资源如表 8-3 所示。

表 8-3 龙芯 2K0500 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。

8.2.3 Interrupt Line 和 Interrupt Pin

由于龙芯 2K0500 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 2K0500 中无意义。

在 40～E4 这些寄存器中，为了保证龙芯 2K0500 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：

表 8-4 龙芯 2K0500 PCI 总线 ISR 寄存器说明

位域	字段名	访问	复位值	说明
ISR_40				
31	tar_read_io	读写 (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 (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 0: 超时后 1: 马上
28	tar_delay_retry	读写	0	target访问重试策略 0: 根据内部逻辑（见29位） 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 000: 8周期 其它: 不支持
19:16	tar_init_timeout	读写	0000	target初始延迟超时 PCI模式下 0: 16周期 1-7: 禁用计数器 8-15: 8-15周期 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 0: 8 delay 访问 8: 1 delay 访问 9: 2 delay 访问 10: 3 delay 访问 11: 4 delay 访问 12: 5 delay 访问 13: 6 delay 访问 14: 7 delay 访问 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） FFF: 64KB到16byte FFE: 64KB到32byte FF8: 64KB到128byte

3	tar_pref_bound_en	读写	0	使用tar_pref_boundary的配置 0: 预取到设备边界 1: 使用tar_pref_boundary
2	Reserved	-	0	
1	tar_splitw_ctrl	读写	0	target split写控制 0: 阻挡除Posted Memory Write以外的访问 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 0: 允许master访问超时 1: 不允许
ISR_44				
31:0	Reserved	-	0	
ISR_48				
31:0	tar_pending_seq	读写	0	target未处理完的请求号位向量 对应位写1可清
ISR_4C				
31:30	Reserved	-	0	
29	mas_write_defer	读写	0	允许后续的读越过前面未完成的写 (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 (只对PCI有效)
27	mas_io_defer_cnt	读写	0	在外的最大IO请求数 0: 由控制 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） 0: 8 1-7: 1-7 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 0:
14	err_module	只读	0	出错的模块 0: target 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	-	0	
ISR_50				
31:0	mas_pending_seq	读写	0	master未处理完的请求号位向量 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	-	0	

29:28	tar_split_priority	读写	0	target split 返回优先级 0最高, 3最低
27:26	mas_req_priority	读写	0	master对外的优先级 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） 0: 固定优先级 1: 轮转
24:18	Reserved	-	0	
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 0: 无限重试 1-255: 1-255次
7:0	mas_trdy_count	读写	00h	master TRDY超时计数器 0: 禁用 1-255: 1-255拍

为保证龙芯 2K0500 的 PCI 控制器正常工作，下列寄存器需要进行相应的配置：

表 8-5 龙芯 2K0500 PCI 总线模式配置寄存器列表

PCI配置寄存器	寄存器描述
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 2K0500 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 2K0500 的 GPU/DC 资源。当龙芯 2K0500 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 2K0500 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 2K0500 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 2K0500 中除 GPU/DC 外的其它 IP 资源。当龙芯 2K0500 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 2K0500 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 2K0500 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 2K0500 中 DDR 内存空间。当龙芯 2K0500 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。

寄存器配置列表如下：

表 8-6 龙芯 2K0500 PCI 总线模式配置寄存器描述

偏移地址	位宽	寄存器	描述	读写特性
0x1fe1_110C	32	PCI_PXARB_CONFIG	仲裁器配置	R/W
0x1fe1_1110	32	PCI_PXARB_STATUS	仲裁器状态	R/W
0x1fe1_1114	32	pcimap	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	R/W
0x1fe1_1118	32	PCIX_RGATE	设置为6'h18	R/W
0x1fe1_111C	32	PCIX_RELAX_EN	设置为0	R/W
0x1fe1_1120	32	pcimap_cfg	详见图8-1	R/W
0x1fe1_1128	32	PCI_Hit0_Sel_L		R/W
0x1fe1_112c	32	PCI_Hit0_Sel_H		R/W
0x1fe1_1130	32	PCI_Hit1_Sel_L		R/W
0x1fe1_1134	32	PCI_Hit1_Sel_H		R/W
0x1fe1_1138	32	PCI_Hit2_Sel_L		R/W
0x1fe1_113c	32	PCI_Hit2_Sel_H		R/W

位域	字段名	访问	复位值	说明
pcimap				
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DDR发读取数门限
6	pcix_ro_en	读写	0	PCIX桥是否允许写越过读
31:7	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；其他：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	0	总线停靠到默认主设备 0:停靠到最后一个主设备 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 0:停靠到最后一个主设备 1:停靠到默认主设备
7:6	park_delay	读写	0	从没有设备请求总线到触发停靠默认设备行为的延迟 00:0周期 01:8周期 10:32周期 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 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	

8.3 PCI 桥片应用

龙芯 2K0500 作为 PCI 桥片应用模式时，为节省芯片整体功耗，通过外部供电电源关断可将 2K0500 处理器核进行断电处理，处理器核进入冷备模式。同时，该模式下对比 SOC 模式， 对部分功能接口进行删减，保留芯片部分功能模块，桥片模式下 2K0500 结构框图如下图所示：

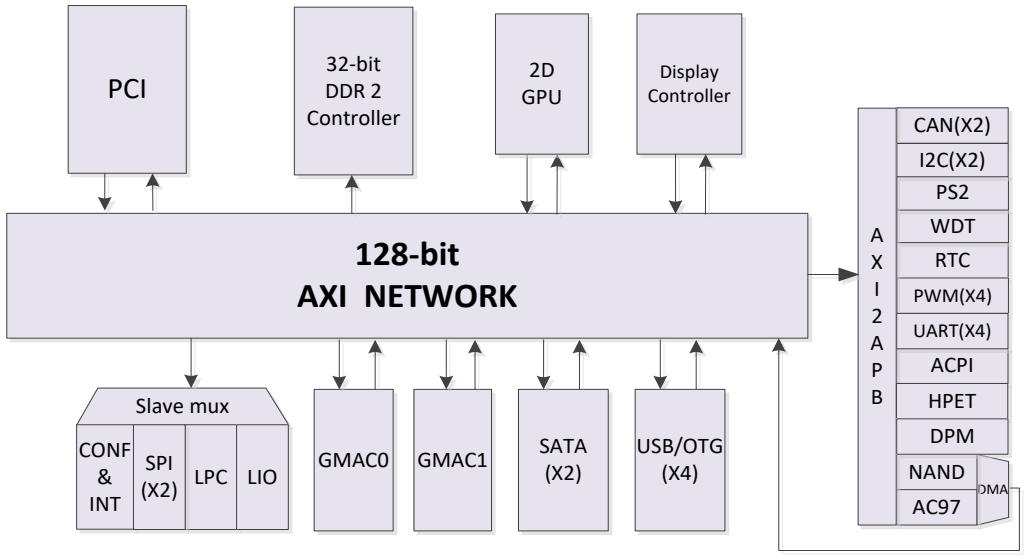


图8-3 2K0500 PCI 桥片模式结构框图

8.3.1 桥片模式地址映射

龙芯 2K0500 桥片模式下，通过 PCI-base0、1、2 三个 PCI 地址窗口将 2K0500 芯片内部设备资源映射到 PCI 总线 64 位地址窗口，这 3 个地址窗口地址由 PCI 控制器 header 寄存器中 base_addr_reg0~5 配置指定，其窗口大小与龙芯 2K0500 内部资源地址映射关系如表 8-7 所示。

表 8-7 龙芯 2K0500 内部资源对应 PCI 总线映射关系

窗口名称	窗口大小	对应资源	芯片内部地址映射	地址空间大小	访问方式
PCI base 0	2MB	DC	0x1c20_0000 - 0x1c2f_ffff	1MB	MEM
		GPU	0x1c30_0000 - 0x1c3f_ffff	1MB	
PCI base 1	16MB	SPI0-MEM	0x1f00_0000 - 0x1f7f_ffff	8MB	MEM
		SPI1-MEM	0x1f80_0000 - 0x1fbf_ffff	4MB	
		BOOT	0x1fc0_0000 - 0x1fcf_ffff	1MB	
		CONFREG	0x1fd0_0000 - 0x1fd3_ffff	256KB	
		DPM	0x1fd4_0000 - 0x1fd4_00ff	256B	
		LPC-REG	0x1fd4_0100 - 0x1fd4_01ff	256B	
		LPC-IO	0x1fd8_0000 - 0x1fd8_ffff	64KB	
		USB	0x1fe0_0000 - 0x1fe0_ffff	64KB	
		GMAC0	0x1fe1_0000 - 0x1fe1_ffff	64KB	
		GMAC1	0x1fe2_0000 - 0x1fe2_ffff	64KB	
		SATA	0x1fe3_0000 - 0x1fe3_ffff	64KB	
		APB-DEVs	0x1fe4_0000 - 0x1fe7_ffff	256KB	
		SPI0-IO	0x1fe8_0000 - 0x1feb_ffff	256KB	
		SPI1-IO	0x1fec_0000 - 0x1fef_ffff	256KB	
		LPC-MEM	0x1ff0_0000 - 0x1fff_ffff	1MB	
PCI base 2	1GB	DDR	0x4000_0000 - 0x7fff_ffff	1GB	MEM

表 8-8 桥片模式下 APB 设备资源地址分配

地址空间分配	大小	APB 设备	访问方式
0x1fe4_0000 - 0x1fe4_3fff	16K	UART0	B
0x1fe4_4000 - 0x1fe4_7fff	16K	UART1	B
0x1fe4_8000 - 0x1fe4_bfff	16K	UART2	B
0x1fe4_c000 - 0x1fe4_ffff	16K	UART3	B
0x1fe5_0000 - 0x1fe5_3fff	16K	CAN0	B
0x1fe5_4000 - 0x1fe5_7fff	16K	CAN1	B
0x1fe5_8000 - 0x1fe5_bfff	16K	I2C0	B
0x1fe5_c000 - 0x1fe5_ffff	16K	PWM	W
0x1fe6_0000 - 0x1fe6_3fff	16K	PS2	B
0x1fe6_4000 - 0x1fe6_7fff	16K	RTC	W
0x1fe6_8000 - 0x1fe6_bfff	16K	I2C1	B
0x1fe6_c000 - 0x1fe6_ffff	16K	HPET	W
0x1fe7_0000 - 0x1fe7_3fff	16K	I2C2	B
0x1fe7_4000 - 0x1fe7_7fff	16K	AC97	W
0x1fe7_8000 - 0x1fe7_bfff	16K	NAND	W

0x1fe7_c000 - 0x1fe7_ffff	16K	ACPI	W

桥片模式下，芯片内部 DMA 主设备包括 GPU、DC、GMAC、USB、SATA、DMA 等，该主设备可通过内部总线访问本地 DDR 内存资源，也可路由至 PCI 总线通过 AM 访问 PCI 主设备(HOST)内存资源，其访问地址分配如下表。

表 8-9 桥片模式下 DMA 设备访问地址

DMA 访问地址空间	大小	访问设备资源
0x0000_0000 - 0x7fff_ffff	2G	本地 DDR 内存
0x8000_0000 - 0xffff_ffff	2G	PCI 总线 AM 访问主设备内存

8.3.2 桥片模式配置寄存器

龙芯 2K0500 桥片模式下，芯片内部配置寄存器与 SOC 模式存在一些差异，包括配置地址和中断寄存器等，桥片模式下芯片配置寄存器如下表。

表 8-10 桥片模式芯片配置寄存器列表

地址	名称	描述
0x1fd00100	CHIP_CTRL0	芯片通用配置寄存器 0
0x1fd00104	CHIP_CTRL1	芯片通用配置寄存器 1
0x1fd00108	CHIP_CTRL2	芯片通用配置寄存器 2
0x1fd0010c	CHIP_CTRL3	芯片通用配置寄存器 3
0x1fd00110	CHIP_CTRL4	芯片通用配置寄存器 4
0x1fd00114	CHIP_CTRL5	芯片通用配置寄存器 5
0x1fd00120	CHIP_SAMP0	芯片采样参数寄存器 0
0x1fd00124	CHIP_SAMP1	芯片采样参数寄存器 1
0x1fd00128	CHIP_SAMP2	芯片采样参数寄存器 2
0x1fd0012c	CHIP_SAMP3	芯片采样参数寄存器 3
0x1fd00130	CHIP_HPT0	芯片计数寄存器低 32 位
0x1fd00134	CHIP_HPT1	芯片计数寄存器高 32 位
0x1fd00400	PLL_NODE_0	NODE 的 PLL 低 32 位配置
0x1fd00404	PLL_NODE_1	NODE 的 PLL 高 32 位配置
0x1fd00408	PLL_DDR_0	内存控制器的 PLL 低 32 位配置
0x1fd0040c	PLL_DDR_1	内存控制器的 PLL 高 32 位配置
0x1fd00410	PLL_SOC_0	SOC 的 PLL 低 32 位配置
0x1fd00414	PLL_SOC_1	SOC 的 PLL 高 32 位配置
0x1fd00418	PLL_PIX0_0	PIX0 的 PLL 低 32 位配置
0x1fd0041c	PLL_PIX0_1	PIX0 的 PLL 高 32 位配置
0x1fd00420	PLL_PIX1_0	PIX1 的 PLL 低 32 位配置
0x1fd00424	PLL_PIX1_1	PIX1 的 PLL 高 32 位配置
0x1fd00428	FREQSCALE	设备时钟分频配置
0x1fd010d0	GPIO0_OEN	GPIO0~31 位输出使能

地址	名称	描述
0x1fd010d4	GPIO1_OEN	GPIO32~63 位输出使能
0x1fd010d8	GPIO2_OEN	GPIO64~95 位输出使能
0x1fd010dc	GPIO3_OEN	GPIO96~127 位输出使能
0x1fd010e0	GPIO0_IN	GPIO0~31 位输入值
0x1fd010e4	GPIO1_IN	GPIO32~63 位输入值
0x1fd010e8	GPIO2_IN	GPIO64~95 位输入值
0x1fd010ec	GPIO3_IN	GPIO96~127 位输入值
0x1fd010f0	GPIO0_OUT	GPIO0~31 位输出值
0x1fd010f4	GPIO1_OUT	GPIO32~63 位输出值
0x1fd010f8	GPIO2_OUT	GPIO64~95 位输出值
0x1fd010fc	GPIO3_OUT	GPIO96~127 位输出值
0x1fd01200	GPIO4_OEN	GPIO128~154 位输出使能
0x1fd01208	GPIO4_IN	GPIO128~154 位输入值
0x1fd01210	GPIO4_OUT	GPIO128~154 位输出值
0x1fd010c0	GPIO_CFG0	GPIO0~7 复用配置寄存器
0x1fd010c4	GPIO_CFG1	GPIO8~15 复用配置寄存器
0x1fd010c8	GPIO_CFG2	GPIO16~23 复用配置寄存器
0x1fd010cc	GPIO_CFG3	GPIO24~31 复用配置寄存器
0x1fd01220	GPIO_CFG4	GPIO32~39 复用配置寄存器
0x1fd01224	GPIO_CFG5	GPIO40~47 复用配置寄存器
0x1fd01228	GPIO_CFG6	GPIO48~55 复用配置寄存器
0x1fd0122c	GPIO_CFG7	GPIO56~63 复用配置寄存器
0x1fd01230	GPIO_CFG8	GPIO64~71 复用配置寄存器
0x1fd01234	GPIO_CFG9	GPIO72~79 复用配置寄存器
0x1fd01238	GPIO_CFG10	GPIO80~87 复用配置寄存器
0x1fd0123c	GPIO_CFG11	GPIO88~95 复用配置寄存器
0x1fd01240	GPIO_CFG12	GPIO96~103 复用配置寄存器
0x1fd01244	GPIO_CFG13	GPIO104~111 复用配置寄存器
0x1fd01248	GPIO_CFG14	GPIO112~119 复用配置寄存器
0x1fd0124c	GPIO_CFG15	GPIO120~127 复用配置寄存器
0x1fd01250	GPIO_CFG16	GPIO128~135 复用配置寄存器
0x1fd01254	GPIO_CFG17	GPIO136~143 复用配置寄存器
0x1fd01258	GPIO_CFG18	GPIO144~151 复用配置寄存器
0x1fd0125c	GPIO_CFG19	GPIO152~154 复用配置寄存器
0x1fd01260	GPIO_INTEN0	GPIO0~31 中断输入使能寄存器
0x1fd01264	GPIO_INTEN1	GPIO32~63 中断输入使能寄存器
0x1fd01268	GPIO_INTEN2	GPIO64~95 中断输入使能寄存器
0x1fd0126c	GPIO_INTEN3	GPIO96~127 中断输入使能寄存器

地址	名称	描述
0x1fd00500	USB_PHY0	USB 的 PHY 配置寄存器 0
0x1fd00504	USB_PHY1	USB 的 PHY 配置寄存器 1
0x1fd00508	USB_PHY2	USB 的 PHY 配置寄存器 2
0x1fd0050c	USB_PHY3	USB 的 PHY 配置寄存器 3
0x1fd00570	SATA0_REG0	SATA0 的配置寄存器 0
0x1fd00574	SATA0_REG1	SATA0 的配置寄存器 1
0x1fd00578	SATA1_REG0	SATA1 的配置寄存器 0
0x1fd0057c	SATA1_REG1	SATA1 的配置寄存器 1
0x1fd00580	SATA_BARCONF0	SATA 转换桥配置寄存器 0
0x1fd00584	SATA_BARCONF1	SATA 转换桥配置寄存器 1
0x1fd00590	SATA0_PHY0	SATA0 的 PHY 配置寄存器 0
0x1fd00594	SATA0_PHY1	SATA0 的 PHY 配置寄存器 1
0x1fd00598	SATA1_PHY0	SATA1 的 PHY 配置寄存器 0
0x1fd0059c	SATA1_PHY1	SATA1 的 PHY 配置寄存器 1
0x1fd00c00	CONFDMA0	DMA0 控制器的配置寄存器
0x1fd00c10	CONFDMA1	DMA1 控制器的配置寄存器
0x1fd00c20	CONFDMA2	DMA2 控制器的配置寄存器
0x1fd00c30	CONFDMA3	DMA3 控制器的配置寄存器
0x1fd01040	INTISR_0	低 32 位中断状态寄存器
0x1fd01044	INTIEN_0	低 32 位中断使能状态寄存器
0x1fd01048	INTSET_0	低 32 位设置使能寄存器
0x1fd0104c	INTCLR_0	低 32 位中断清除寄存器，清除使能寄存器和脉冲触发的中断
0x1fd01050	INTPOL_0	低 32 位极性设置寄存器(电平中断)
0x1fd01054	INTEDGE_0	低 32 位触发方式寄存器（1：脉冲触发；0：电平触发）
0x1fd01058	INTISR_1	高 32 位中断状态寄存器
0x1fd0105c	INTIEN_1	高 32 位中断使能状态寄存器
0x1fd01060	INTSET_1	高 32 位设置使能寄存器
0x1fd01064	INTCLR_1	高 32 位中断清除寄存器，清除使能寄存器和脉冲触发的中断
0x1fd01068	INTPOL_1	高 32 位极性设置寄存器(电平中断)
0x1fd0106c	INTEDGE_1	高 32 位触发方式寄存器（1：脉冲触发；0：电平触发）
0x1fd01070	INTISR_2	GPIO0~31 中断状态寄存器
0x1fd01074	INTIEN_2	GPIO0~31 中断使能状态寄存器

地址	名称	描述
0x1fd01078	INTSET_2	GPIO0~31 中断设置使能寄存器
0x1fd0107c	INTCLR_2	GPIO0~31 中断清除寄存器，清除使能寄存器和脉冲触发的中断
0x1fd01080	INTPOL_2	GPIO0~31 中断极性设置寄存器（电平中断）
0x1fd01084	INTEDGE_2	GPIO0~31 中断触发方式寄存器（1：脉冲触发；0：电平触发）
0x1fd01088	INTISR_3	GPIO32~63 中断状态寄存器
0x1fd0108c	INTIEN_3	GPIO32~63 中断使能状态寄存器
0x1fd01090	INTSET_3	GPIO32~63 中断设置使能寄存器
0x1fd01094	INTCLR_3	GPIO32~63 中断清除寄存器，清除使能寄存器和脉冲触发的中断
0x1fd01098	INTPOL_3	GPIO32~63 中断极性设置寄存器（电平中断）
0x1fd0109c	INTEDGE_3	GPIO32~63 中断触发方式寄存器（1：脉冲触发；0：电平触发）
0x1fd010a0	INTISR_4	GPIO64~95 中断状态寄存器
0x1fd010a4	INTIEN_4	GPIO64~95 中断使能状态寄存器
0x1fd010a8	INTSET_4	GPIO64~95 中断设置使能寄存器
0x1fd010ac	INTCLR_4	GPIO64~95 中断清除寄存器，清除使能寄存器和脉冲触发的中断
0x1fd010b0	INTPOL_4	GPIO64~95 中断极性设置寄存器（电平中断）
0x1fd010b4	INTEDGE_4	GPIO64~95 中断触发方式寄存器（1：脉冲触发；0：电平触发）
0x1fd03ff8	CHIP_CHIPID0	芯片识别号 0
0x1fd03ffc	CHIP_CHIPID1	芯片识别号 1

8.3.3 桥片模式中断描述

龙芯 2K0500 桥片模式下，中断控制器与 SOC 模式下存在差异，桥片模式下芯片中断控制器共五个中断输出连接，分别对应 INT0、INT1、INT2、INT3、INT4。其中 INT0/1、INT2/3/4 还分别送到芯片输出引脚 SYS_INTn0、SYS_INTn1，用于在桥片模式下给主 HOST 处理器触发中断。

桥片模式下芯片支持 64 个内部中断和 96 个 GPIO 外部中断；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 96 个外部 GPIO 中断。具体如下表所示：

表 8-11 桥片模式 2K0500 中断源列表

中断编号	INT0	INT1	INT2	INT3	INT4
31	保留	保留	GPIO31	GPIO63	GPIO95
30	保留	保留	GPIO30	GPIO62	GPIO94
29	NAND_int	保留	GPIO29	GPIO61	GPIO93
28	TOY_TICK	保留	GPIO28	GPIO60	GPIO92
27	RTC_TICK	保留	GPIO27	GPIO59	GPIO91
26	TOY_INT2	保留	GPIO26	GPIO58	GPIO90
25	TOY_INT1	保留	GPIO25	GPIO57	GPIO89
24	TOY_INT0	保留	GPIO24	GPIO56	GPIO88
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

PCI 桥片模式下中断使用：首先，设置中断使能寄存器中相应位使能中断，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属性；最后，当发生中断时，通过中断状态寄存器查看相应中断源。

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。边沿触发方式下，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。

表 8-12 桥片模式 2K0500 中断寄存器列表

寄存器地址	位域	中断寄存器	功能描述	读写属性
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
0x1fd01064	32	INTCLR1	中断清空寄存器 1	R/W
0x1fd01068	32	INTPOL1	高电平触发中断使能寄存器 1	R/W
0x1fd0106c	32	INTEDGE1	边沿触发中断使能寄存器 1	R/W
0x1fd01070	32	INTISR2	中断控制状态寄存器 2	RO
0x1fd01074	32	INTIEN2	中断控制使能寄存器 2	R/W
0x1fd01078	32	INTSET2	中断置位寄存器 2	R/W
0x1fd0107c	32	INTCLR2	中断清空寄存器 2	R/W
0x1fd01080	32	INTPOL2	高电平触发中断使能寄存器 2	R/W
0x1fd01084	32	INTEDGE2	边沿触发中断使能寄存器 2	R/W
0x1fd01088	32	INTISR3	中断控制状态寄存器 3	RO
0x1fd0108c	32	INTIEN3	中断控制使能寄存器 3	R/W
0x1fd01090	32	INTSET3	中断置位寄存器 3	R/W
0x1fd01094	32	INTCLR3	中断清空寄存器 3	R/W
0x1fd01098	32	INTPOL3	高电平触发中断使能寄存器 3	R/W
0x1fd0109c	32	INTEDGE3	边沿触发中断使能寄存器 3	R/W
0x1fd010a0	32	INTISR4	中断控制状态寄存器 4	RO
0x1fd010a4	32	INTIEN4	中断控制使能寄存器 4	R/W
0x1fd010a8	32	INTSET4	中断置位寄存器 4	R/W
0x1fd010ac	32	INTCLR4	中断清空寄存器 4	R/W
0x1fd010b0	32	INTPOL4	高电平触发中断使能寄存器 4	R/W
0x1fd010b4	32	INTEDGE4	边沿触发中断使能寄存器 4	R/W
0x1fd01260	32	GPIO_INTEN0	GPIO0~31 中断输入使能寄存器	R/W
0x1fd01264	32	GPIO_INTEN1	GPIO32~63 中断输入使能寄存器	R/W
0x1fd01268	32	GPIO_INTEN2	GPIO64~95 中断输入使能寄存器	R/W

9 显示控制器

9.1 概述

显示控制器从内存中取帧缓冲和光标信息输出到外部显示接口上。

龙芯 2K0500 的显示控制器支持的特性包括：

1) 双路 DVO 接口显示，一路 DVO 接口，一路 VGA 接口
2) 每路显示最大支持至 1920x1080@60Hz
3) Monochrome、ARGB8888 两种模式硬件光标
4) RGB444、RGB555、RGB565、RGB888 四种色深
5) 输出抖动和伽马校正
6) 可切换的双路线性帧缓冲
7) 中断和软复位

9.2 寄存器访问地址和引脚说明

显示控制器寄存器基址为：0x1f01_0000。

对于显示控制器模块，使用时要注意将对应的芯片复用引脚设置为相应的接口功能。

与显示控制器接口相关的引脚复用设置可查询节 2.29 中的外设功能引脚复用关系表，并根据节 5.5.40~节 5.5.59 中 GPIO0~159 复用配置寄存器，配置相应的 GPIO 引脚复用关系，实现对应设备功能引脚。

10 GMAC 控制器

龙芯 2K0500 集成了两个 GMAC 控制器，即 GMAC0 和 GMAC1，二者在逻辑结构上完全相同。其中，GMAC0 接口支持系统唤醒功能，系统休眠时可通过 GMAC0 接口进行唤醒操作。

10.1 访问地址及引脚说明

GMAC 控制器寄存器包括 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 0x1f02_0000；GMAC0 的 DMA 寄存器的起始地址是 0x1f02_1000。

GMAC1 的 GMAC 寄存器的起始地址是 0x1f03_0000；GMAC1 的 DMA 寄存器的起始地址是 0x1f03_1000。

对于 GMAC 接口模块，使用时要注意将对应的芯片复用引脚设置为相应的接口功能。

与 GMAC 接口相关的引脚复用设置可查询 2.29 节中的外设功能引脚复用关系表，并根据节 5.5.40~节 5.5.59 中 GPIO0~159 复用配置寄存器，配置相应的 GPIO 引脚复用关系，实现对应设备功能引脚。

11 SATA 控制器

11.1 SATA 总体描述

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

11.2 SATA 控制器内部寄存器描述

SATA 的基地址是 0x1f040000，寄存器的定义和协议标准定义完全一致。

偏移地址	位宽	名称	描述
0x000	32	CAP	HBA 特性寄存器
0x004	32	GHC	全局 HBA 控制寄存器
0x008	32	IS	中断状态寄存器
0x00c	32	PI	端口寄存器
0x010	32	VS	AHCI 版本寄存器
0x014	32	CCC_CTL	命令完成合并控制寄存器
0x018	32	CCC_PORTS	命令完成合并端口寄存器
0x024	32	CAP2	HBA 特性扩展寄存器
0x0a0	32	BISTAFR	BIST 激活 FIS
0x0a4	32	BISTCR	BIST 控制寄存器
0x0a8	32	BISTCTR	BIST FIS 计数寄存器
0x0ac	32	BISTSR	BIST 状态寄存器
0x0b0	32	BISTDECR	BIST 双字错计数寄存器
0x0bc	32	OOBR	OOB 寄存器
0x0e0	32	TIMER1MS	1ms 计数寄存器
0x0e8	32	GPARAM1R	全局参数寄存器 1
0x0ec	32	GPARAM2R	全局参数寄存器 2
0x0f0	32	PPARAMR	端口参数寄存器
0x0f4	32	TESTR	测试寄存器
0x0f8	32	VERIONR	版本寄存器
0x0fc	32	IDR	ID 寄存器
0x100	32	P0_CLB	命令列表基地址低 32 位
0x104	32	P0_CLBU	命令列表基地址高 32 位
0x108	32	P0_FB	FIS 基地址低 32 位
0x10c	32	P0_FBU	FIS 基地址高 32 位
0x110	32	P0_IS	中断状态寄存器
0x114	32	P0_IE	中断使能寄存器
0x118	32	P0_CMD	命令寄存器

0x120	32	P0_TFD	任务文件数据寄存器
0x124	32	P0_SIG	签名寄存器
0x128	32	P0_SSTS	SATA 状态寄存器
0x12c	32	P0_SCTL	SATA 控制寄存器
0x130	32	P0_SERR	SATA 错误寄存器
0x134	32	P0_SACT	SATA 激活寄存器
0x138	32	P0_CI	命令发送寄存器
0x13c	32	P0_SNTF	SATA 命令通知寄存器
0x170	32	P0_DMACR	DMA 控制寄存器
0x178	32	P0_PHYCR	PHY 控制寄存器
0x17c	32	P0_PHYSR	PHY 状态寄存器
0x180	32	P1_CLB	命令列表基地址低 32 位
0x184	32	P1_CLBU	命令列表基地址高 32 位
0x188	32	P1_FB	FIS 基地址低 32 位
0x18c	32	P1_FBU	FIS 基地址高 32 位
0x190	32	P1_IS	中断状态寄存器
0x194	32	P1_IE	中断使能寄存器
0x108	32	P1_CMD	命令寄存器
0x1a0	32	P1_TFD	任务文件数据寄存器
0x1a4	32	P1_SIG	签名寄存器
0x1a8	32	P1_SSTS	SATA 状态寄存器
0x1ac	32	P1_SCTL	SATA 控制寄存器
0x1b0	32	P1_SERR	SATA 错误寄存器
0x1b4	32	P1_SACT	SATA 激活寄存器
0x1b8	32	P1_CI	命令发送寄存器
0x1bc	32	P1_SNTF	SATA 命令通知寄存器
0x1f0	32	P1_DMACR	DMA 控制寄存器
0x1f8	32	P1_PHYCR	PHY 控制寄存器
0x1fc	32	P1s_PHYSR	PHY 状态寄存器

12 USB 控制器

12.1 总体概述

2K0500 的 USB 主机端口特性如下：

- 兼容 USB Rev 1.1、USB Rev 2.0 协议、USB Rev 3.0 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
- 支持四个 USB2.0 端口，每个端口都可挂 LS、FS 或 HS 设备
- 支持一个 USB3.0 端口

USB 主机控制器模块包括一个支持高速设备的 EHCI 控制器，一个支持全速与低速设备的 OHCI 控制器。其中 EHCI 控制器处于主控地位，只有当挂上的设备是全速或低速设备时，才将控制权转交给 OHCI 控制器；当全速或低速设备拔掉时，控制权返回 EHCI 控制器。

同时 USB 控制器内部集成了 AHB 总线接口（与 AMBA Specification Revision 2.0 兼容），用来和内存/应用程序之间通信。USB 控制器与外部的互联结构图如下图所示：

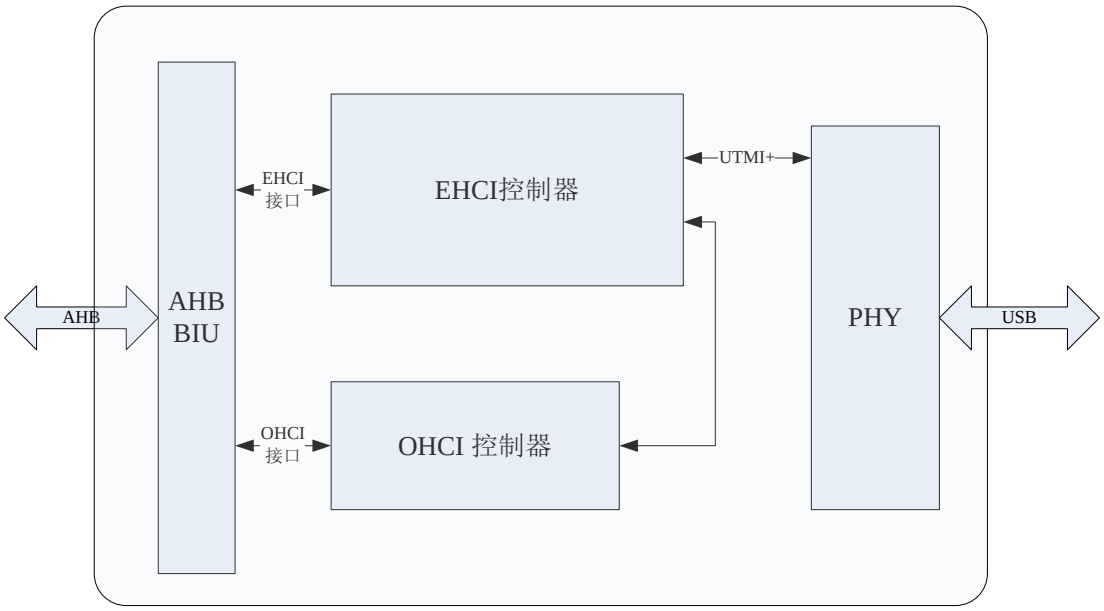


图 12-1 USB 主机控制器模块图

12.2 USB 配置寄存器

表12-1 USB控制器地址空间分布

地址空间	名称	大小
0x1f05,0000 – 0x1f05,7fff	EHCI 寄存器	32KB
0x1f05,8000 – 0x1f05,ffff	OHCI 寄存器	32KB

地址空间	名称	大小
0x1f06,0000 – 0x1f06,7fff	USB3.0 控制寄存器	32KB
0x1f06,8000 – 0x1f06,efff	USB3.0 PHY 接口寄存器	28KB
0x1f06,ff00 – 0x1f06,ffff	USB3.0 接口配置寄存器	256B

13 OTG 控制器

13.1 概述

2K0500 的 OTG 支持特性如下:

- 支持 HNP 与 SRP 协议；
- 内嵌 DMA，无需占用处理器带宽即可在 OTG 与外部存储之间移动数据；
- 在 device 模式下，为高速设备（480Mbps）；
- 在 host 模式下，仅能支持高速设备（480Mbps）；
- 在 device 模式下，支持 6 个双向的 endpoint，其中仅有默认的 endpoint0 支持控制传输；
- 在 device 模式下，最多同时支持 4 个 IN 方向的传输；
- 在 host 模式下，支持 12 个 channel，且软件可配置每个 channel 的方向；
- 在 host 模式下，支持 periodic OUT 传输。

13.2 寄存器访问地址

应用程序通过系统配置接口来读写 OTG 控制器里的控制与状态寄存器，这些寄存器都是 32 位宽，寄存器地址为 32 位对齐。

表13-1 OTG 配置寄存器访问地址

地址空间	名称	大小
0x1f08_0000 – 0x1f0b_ffff	OTG 配置寄存器	256KB

14 PRINT 控制器

龙芯 2K0500 集成一个打印机接口控制器，主要包括 LSU 机芯控制和 JBIG85 解码模块。其中，LSU 机芯控制模块同时支持 8 路机芯控制，可实现同时打印青、粉、黄、黑四种颜色彩打功能；JBIG85 解码模块可实现对由 jbig85 压缩算法压缩过的图片数据进行解码处理，可负责同时对青、粉、黄、黑四个颜色进行对应数据的解码工作。

14.1 功能概述

14.1.1 LSU 机芯控制

LSU 控制模块主要包含三个模块：AXI/APB 控制模块，LSU 激光头控制模块以及 MOTOR 马达控制模块。结构框图如图 14-1 所示：

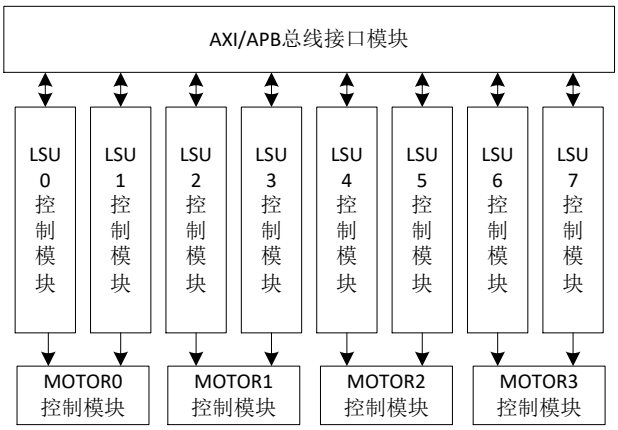


图 14-1 LSU 结构示意图

AXI/APB 控制模块主要负责配置打印相关的配置寄存器，从内存中取出需要打印的数据。8 个激光头共需配置 196 个 32 位配置寄存器，AXI 对 8 个激光头采取分时复用的策略，保证 8 个激光头可以公平的得到数据，保证不会出现某一激光头数据耗光。

LSU 控制模块共 8 个，分别独立控制 8 个激光头，其中两两组成一组，共分成四组。四组 LSU 控制模块可同时打印青粉黄黑四种颜色以达到彩打目的，每一组 LSU 控制模块可执行隔行打印，提高打印速度。LSU 控制模块经过配置可支持自左向右打印、自右向左打印、双面打印、多页打印、按字节翻转等操作。可根据打印机结构的不同，支持软件或传感器控制打印起始，并可配置精细调整打印延迟。LSU 控制模块输入时钟由 DDR-PLL 中 NETWORK 输出时钟经两级分频后输出产生，LSU 控制器配置参数中相关打印延迟参数、马达转速等配置基准时钟频率均与该输入时钟频率保持一致。

激光头接口包括 ENB，HSYNC，SH，VIDEO/DATA 组成。ENB 为龙芯 2K0500 的输出 GPO，使能激光头开关。HSYNC 为龙芯 2K0500 的输入 PWM，表示一线打印数据的开始/幅面对齐。SH 为龙芯 2K0500 的输出 PWM，表示打印数据窗口，低电平 VIDEO/DATA 无效，高电平 VIDEO/DATA 有效。VIDEO/DATA 为龙芯 2K0500 的输出 PWM+DMA，表示每线打印数据。

目前，打印控制模块最多支持 8 个激光头控制。可支持黑白，红黑，彩色打印机使用的激光头。支持激光头的接口示例如下：

a）黑白低速：黑单激光头：ENB0，HSYNC0，SH0，DATA0。

b）红黑低速：红单激光头：ENB0，HSYNC0，SH0，DATA0。黑单激光头：ENB2，HSYNC2，SH2，DATA2。

c）彩色高速：青双激光头：ENB0，HSYNC0，SH0，DATA0，ENB1，SH1，DATA1。粉双激光头：ENB2，HSYNC2，SH2，DATA2，ENB3，SH3，DATA3。黄双激光头：ENB4，HSYNC4，SH4，DATA4，ENB5，SH5，DATA5。黑双激光头：ENB6，HSYNC6，SH6，DATA6，ENB7，SH7，DATA7。

MOTOR 马达控制模块共 4 个，分别负责 4 组 LSU 控制模块。马达可根据打印机自身需求，精细调整马达转速，配合激光机打印速度。

马达接口包括 CLOCK/STEP，START/ENB，READY/LOCK 组成。START/ENB 为龙芯 2K0500 的输出 GPO，使能马达开关。CLOCK/STEP 为龙芯 2K0500 的输出 PWM，控制马达转速，马达静止没有 PWM 输出时，默认高电平。READY/LOCK 为龙芯 2K0500 的输入 GPI，表示马达启转稳定与否。

14.1.2 JBIG85 解码

JBIG85 解码单元包括 4 个模块，分别负责对青粉黄黑四个颜色进行对应数据的解码工作。每个模块包括主要包括两个子模块：主控制模块，JBIG85 算法解码模块。结构框图如图 14-2 所示：

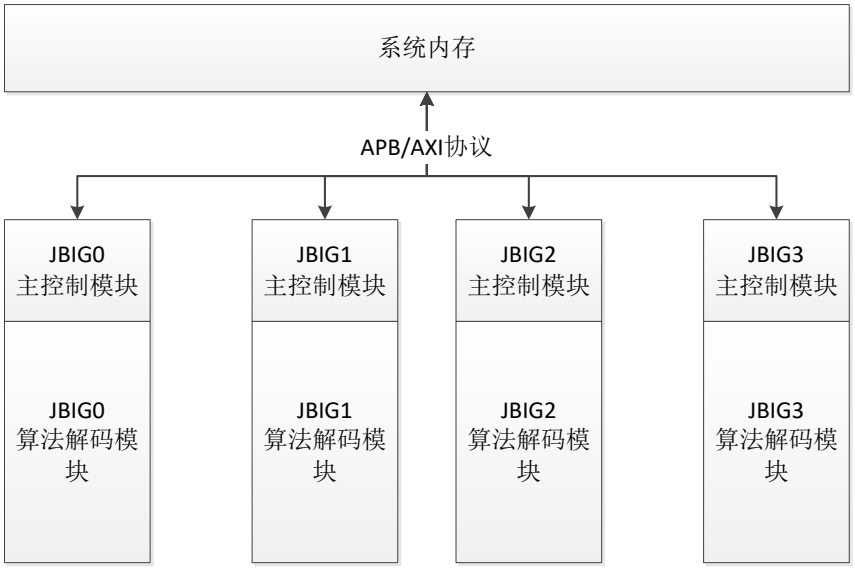


图 14-2 JBIG85 解码结构示意图

JBIG85 主控制模块包括三个功能：寄存器配置、输入数据读取，输出数据写回。配置寄存器通过配置总线进行寄存器读写，寄存器项包括青粉黄黑各模块使能寄存器、输入数据起始地址、输入数据长度、输出数据起始地址、多模块的中断传递类型、解码错误状态寄存器。输入数据保存在内存中，数据地址由 APB 进行配置，主控制模块通过 AXI 协议对数据进

行读取，并将读取的数据发送至 JBIG85 解码模块进行解码。输出数据通过 AXI 协议进行写回，写回基地址同样由 APB 进行配置，并且所有写回数据按照图片行数据进行 256 像素单位对齐。

JBIG85 算法解码模块主要实现对经由 jbig85 压缩算法压缩过的图片数据。

14.2 寄存器描述

打印机控制器内部寄存器的物理地址构成如下：

物理地址	设备	备注
0x1f0c_0000	LSU	机芯控制模块配置寄存器
0x1f0c_2000	JBIG85	JBIG85 解码模块配置寄存器

对于打印机接口模块，使用时要注意将对应的芯片复用引脚设置为相应的接口功能。

与打印机接口相关的引脚复用设置可查询节 2.29 中的外设功能引脚复用关系表，并根据节 5.5.40~节 5.5.59 中 GPIO0~159 复用配置寄存器，配置相应的 GPIO 引脚复用关系，实现对应设备功能引脚。

14.2.1 LSU 模块寄存器

LSU 机芯控制模块寄存器包括以下配置寄存器：

地址偏移	寄存器名称	描述
0x00	PO_CONTROL0	LSU0 重置控制寄存器
0x04	PO_CONTROL2	LSU2 重置控制寄存器
0x08	PO_CONTROL4	LSU4 重置控制寄存器
0x0c	PO_CONTROL6	LSU6 重置控制寄存器
0x10	PO_CONTROL1	LSU1 重置控制寄存器
0x14	PO_CONTROL3	LSU3 重置控制寄存器
0x18	PO_CONTROL5	LSU5 重置控制寄存器
0x1c	PO_CONTROL7	LSU7 重置控制寄存器
0x20	PO_SYNC_MODE0	LSU0 同步控制寄存器
0x24	PO_SYNC_MODE2	LSU2 同步控制寄存器
0x28	PO_SYNC_MODE4	LSU4 同步控制寄存器
0x2c	PO_SYNC_MODE6	LSU6 同步控制寄存器
0x30	PO_SYNC_MODE1	LSU1 同步控制寄存器
0x34	PO_SYNC_MODE3	LSU3 同步控制寄存器
0x38	PO_SYNC_MODE5	LSU5 同步控制寄存器
0x3c	PO_SYNC_MODE7	LSU7 同步控制寄存器
0x40	PO_OUTPUT_FORMAT0	LSU0 输出格式寄存器
0x44	PO_OUTPUT_FORMAT2	LSU2 输出格式寄存器
0x48	PO_OUTPUT_FORMAT4	LSU4 输出格式寄存器
0x4c	PO_OUTPUT_FORMAT6	LSU6 输出格式寄存器
0x50	PO_OUTPUT_FORMAT1	LSU1 输出格式寄存器
0x54	PO_OUTPUT_FORMAT3	LSU3 输出格式寄存器
0x58	PO_OUTPUT_FORMAT5	LSU5 输出格式寄存器
0x5c	PO_OUTPUT_FORMAT7	LSU7 输出格式寄存器
0x60	PO_LEFT_MARGIN0	LSU0 左边距寄存器

0x64	PO_LEFT_MARGIN2	LSU2 左边距寄存器
0x68	PO_LEFT_MARGIN4	LSU4 左边距寄存器
0x6c	PO_LEFT_MARGIN6	LSU6 左边距寄存器
0x70	PO_LEFT_MARGIN1	LSU1 左边距寄存器
0x74	PO_LEFT_MARGIN3	LSU3 左边距寄存器
0x78	PO_LEFT_MARGIN5	LSU5 左边距寄存器
0x7c	PO_LEFT_MARGIN7	LSU7 左边距寄存器
0x80	PO_SCAN_ACTIVE0	LSU0 横幅面寄存器
0x84	PO_SCAN_ACTIVE2	LSU2 横幅面寄存器
0x88	PO_SCAN_ACTIVE4	LSU4 横幅面寄存器
0x8c	PO_SCAN_ACTIVE6	LSU6 横幅面寄存器
0x90	PO_SCAN_ACTIVE1	LSU1 横幅面寄存器
0x94	PO_SCAN_ACTIVE3	LSU3 横幅面寄存器
0x98	PO_SCAN_ACTIVE5	LSU5 横幅面寄存器
0x9c	PO_SCAN_ACTIVE7	LSU7 横幅面寄存器
0xa0	PO_TOP_MARGIN0	LSU0 顶边距寄存器
0xa4	PO_TOP_MARGIN2	LSU2 顶边距寄存器
0xa8	PO_TOP_MARGIN4	LSU4 顶边距寄存器
0xac	PO_TOP_MARGIN6	LSU6 顶边距寄存器
0xb0	PO_TOP_MARGIN1	LSU1 顶边距寄存器
0xb4	PO_TOP_MARGIN3	LSU3 顶边距寄存器
0xb8	PO_TOP_MARGIN5	LSU5 顶边距寄存器
0xbc	PO_TOP_MARGIN7	LSU7 顶边距寄存器
0xc0	PO_PAGE_ACTIVE0	LSU0 纵幅面寄存器
0xc4	PO_PAGE_ACTIVE2	LSU2 纵幅面寄存器
0xc8	PO_PAGE_ACTIVE4	LSU4 纵幅面寄存器
0xcc	PO_PAGE_ACTIVE6	LSU6 纵幅面寄存器
0xd0	PO_PAGE_ACTIVE1	LSU1 纵幅面寄存器
0xd4	PO_PAGE_ACTIVE3	LSU3 纵幅面寄存器
0xd8	PO_PAGE_ACITVE5	LSU5 纵幅面寄存器
0xdc	PO_PAGE_ACTIVE7	LSU7 纵幅面寄存器
0xe0	PO_LSAUX_ENABLE0	LSU0 SH_DATA 使能寄存器
0xe4	PO_LSAUX_ENABLE2	LSU2 SH_DATA 使能寄存器
0xe8	PO_LSAUX_ENABLE4	LSU4 SH_DATA 使能寄存器
0xec	PO_LSAUX_ENABLE6	LSU6 SH_DATA 使能寄存器
0xf0	PO_LSAUX_ENABLE1	LSU1 SH_DATA 使能寄存器
0xf4	PO_LSAUX_ENABLE3	LSU3 SH_DATA 使能寄存器
0xf8	PO_LSAUX_ENABLE5	LSU5 SH_DATA 使能寄存器
0xfc	PO_LSAUX_ENABLE7	LSU7 SH_DATA 使能寄存器
0x100	PO_LSAUX_SHGOHI0	LSU0 SH 拉高寄存器
0x104	PO_LSAUX_SHGOHI2	LSU2 SH 拉高寄存器
0x108	PO_LSAUX_SHGOHI4	LSU4 SH 拉高寄存器
0x10c	PO_LSAUX_SHGOHI6	LSU6 SH 拉高寄存器
0x110	PO_LSAUX_SHGOHI1	LSU1 SH 拉高寄存器
0x114	PO_LSAUX_SHGOHI3	LSU3 SH 拉高寄存器
0x118	PO_LSAUX_SHGOHI5	LSU5 SH 拉高寄存器
0x11c	PO_LSAUX_SHGOHI7	LSU7 SH 拉高寄存器
0x120	PO_LSAUX_SHGOLO0	LSU0 SH 拉低寄存器
0x124	PO_LSAUX_SHGOLO2	LSU2 SH 拉低寄存器
0x128	PO_LSAUX_SHGOLO4	LSU4 SH 拉低寄存器

0x12c	PO_LSAUX_SHGOLO6	LSU6 SH 拉低寄存器
0x130	PO_LSAUX_SHGOLO1	LSU1 SH 拉低寄存器
0x134	PO_LSAUX_SHGOLO3	LSU3 SH 拉低寄存器
0x138	PO_LSAUX_SHGOLO5	LSU5 SH 拉低寄存器
0x13c	PO_LSAUX_SHGOLO7	LSU7 SH 拉低寄存器
0x140	PO_LSAUX_DATAGOHI0	LSU0 DATA 拉高寄存器
0x144	PO_LSAUX_DATAGOHI2	LSU2 DATA 拉高寄存器
0x148	PO_LSAUX_DATAGOHI4	LSU4 DATA 拉高寄存器
0x14c	PO_LSAUX_DATAGOHI6	LSU6 DATA 拉高寄存器
0x150	PO_LSAUX_DATAGOHI1	LSU1 DATA 拉高寄存器
0x154	PO_LSAUX_DATAGOHI3	LSU3 DATA 拉高寄存器
0x158	PO_LSAUX_DATAGOHI5	LSU5 DATA 拉高寄存器
0x15c	PO_LSAUX_DATAGOHI7	LSU7 DATA 拉高寄存器
0x160	PO_LSAUX_DATAGOLO0	LSU0 DATA 拉低寄存器
0x164	PO_LSAUX_DATAGOLO2	LSU2 DATA 拉低寄存器
0x168	PO_LSAUX_DATAGOLO4	LSU4 DATA 拉低寄存器
0x16c	PO_LSAUX_DATAGOLO6	LSU6 DATA 拉低寄存器
0x170	PO_LSAUX_DATAGOLO1	LSU1 DATA 拉低寄存器
0x174	PO_LSAUX_DATAGOLO3	LSU3 DATA 拉低寄存器
0x178	PO_LSAUX_DATAGOLO5	LSU5 DATA 拉低寄存器
0x17c	PO_LSAUX_DATAGOLO7	LSU7 DATA 拉低寄存器
0x180	PO_BASE0	LSU0 基地址寄存器
0x184	PO_BASE2	LSU2 基地址寄存器
0x188	PO_BASE4	LSU4 基地址寄存器
0x18c	PO_BASE6	LSU6 基地址寄存器
0x190	PO_BASE1	LSU1 基地址寄存器
0x194	PO_BASE3	LSU3 基地址寄存器
0x198	PO_BASE5	LSU5 基地址寄存器
0x19c	PO_BASE7	LSU7 基地址寄存器
0x1a0	PO_STRIDE0	LSU0 步长寄存器
0x1a4	PO_STRIDE2	LSU2 步长寄存器
0x1a8	PO_STRIDE4	LSU4 步长寄存器
0x1ac	PO_STRIDE6	LSU6 步长寄存器
0x1b0	PO_STRIDE1	LSU1 步长寄存器
0x1b4	PO_STRIDE3	LSU3 步长寄存器
0x1b8	PO_STRIDE5	LSU5 步长寄存器
0x1bc	PO_STRIDE7	LSU7 步长寄存器
0x1c0	PO_YSIZE0	LSU0 总线数寄存器
0x1c4	PO_YSIZE2	LSU2 总线数寄存器
0x1c8	PO_YSIZE4	LSU4 总线数寄存器
0x1cc	PO_YSIZE6	LSU6 总线数寄存器
0x1d0	PO_YSIZE1	LSU1 总线数寄存器
0x1d4	PO_YSIZE3	LSU3 总线数寄存器
0x1d8	PO_YSIZE5	LSU5 总线数寄存器
0x1dc	PO_YSIZE7	LSU7 总线数寄存器
0x1e0	PO_YVALUE0	LSU0 输出线数寄存器
0x1e4	PO_YVALUE2	LSU2 输出线数寄存器
0x1e8	PO_YVALUE4	LSU4 输出线数寄存器
0x1ec	PO_YVALUE6	LSU6 输出线数寄存器
0x1f0	PO_YVALUE1	LSU1 输出线数寄存器

0x1f4	PO_YVALUE3	LSU3 输出线数寄存器
0x1f8	PO_YVALUE5	LSU5 输出线数寄存器
0x1fc	PO_YVALUE7	LSU7 输出线数寄存器
0x200	PO_YVOLUME0	LSU0 剩余线数寄存器
0x204	PO_YVOLUME2	LSU2 剩余线数寄存器
0x208	PO_YVOLUME4	LSU4 剩余线数寄存器
0x20c	PO_YVOLUME6	LSU6 剩余线数寄存器
0x210	PO_YVOLUME1	LSU1 剩余线数寄存器
0x214	PO_YVOLUME3	LSU3 剩余线数寄存器
0x218	PO_YVOLUME5	LSU5 剩余线数寄存器
0x21c	PO_YVOLUME7	LSU7 剩余线数寄存器
0x220	PO_STINT_CLEAR	LSU0 打印开始中断清除寄存器
0x224	PO_STINT_SET	LSU2 打印开始中断置位寄存器
0x228	PO_STINT_ENABLE	LSU4 打印开始中断使能寄存器
0x22c	PO_STINT_STATUS	LSU6 打印开始中断状态寄存器
0x230	PO_ENDINT_CLEAR	LSU1 打印结束中断清除寄存器
0x234	PO_ENDINT_SET	LSU3 打印结束中断置位寄存器
0x238	PO_ENDINT_ENABLE	LSU5 打印结束中断使能寄存器
0x23c	PO_ENDINT_STATUS	LSU7 打印结束中断状态寄存器
0x240	PRINT_START_RESET	激光控制寄存器
0x250	FREQUENCY_TEMP_COUNT0	0 号马达转速寄存器
0x254	FREQUENCY_TEMP_COUNT1	1 号马达转速寄存器
0x258	FREQUENCY_TEMP_COUNT2	2 号马达转速寄存器
0x25c	FREQUENCY_TEMP_COUNT3	3 号马达转速寄存器
0x260	PO_BUFFER_CONTROL0	LSU0 镜像输出寄存器
0x264	PO_BUFFER_CONTROL2	LSU2 镜像输出寄存器
0x268	PO_BUFFER_CONTROL4	LSU4 镜像输出寄存器
0x26c	PO_BUFFER_CONTROL6	LSU6 镜像输出寄存器
0x270	PO_BUFFER_CONTROL1	LSU1 镜像输出寄存器
0x274	PO_BUFFER_CONTROL3	LSU3 镜像输出寄存器
0x278	PO_BUFFER_CONTROL5	LSU5 镜像输出寄存器
0x27c	PO_BUFFER_CONTROL7	LSU7 镜像输出寄存器
0x280	DOT_DELAY0	LSU0 点延迟寄存器
0x284	DOT_DELAY2	LSU2 点延迟寄存器
0x288	DOT_DELAY4	LSU4 点延迟寄存器
0x28c	DOT_DELAY6	LSU6 点延迟寄存器
0x290	DOT_DELAY1	LSU1 点延迟寄存器
0x294	DOT_DELAY3	LSU3 点延迟寄存器
0x298	DOT_DELAY5	LSU5 点延迟寄存器
0x29c	DOT_DELAY7	LSU7 点延迟寄存器
0x2a0	ENB_DELAY0	LSU0 使能延迟寄存器
0x2a4	ENB_DELAY2	LSU2 使能延迟寄存器
0x2a8	ENB_DELAY4	LSU4 使能延迟寄存器
0x2ac	ENB_DELAY6	LSU6 使能延迟寄存器
0x2b0	ENB_DELAY1	LSU1 使能延迟寄存器
0x2b4	ENB_DELAY3	LSU3 使能延迟寄存器
0x2b8	ENB_DELAY5	LSU5 使能延迟寄存器
0x2bc	ENB_DELAY7	LSU7 使能延迟寄存器
0x2c0	DMA_INT_CLEAR	DMA 中断关闭寄存器
0x2c4	DMA_INT_SET	DMA 中断开启寄存器

0x2c8	DMA_INT_ENABLE	DMA 中断屏蔽寄存器
0x2cc	DMA_INT_STATUS	DMA 中断状态寄存器
0x2d0	LSU_START_POL	START 电平控制寄存器
0x2d4	LSU_ENB_POL	ENB 电平控制寄存器
0x2d8	LSU_SH_POL	SH 电平控制寄存器
0x2dc	LSU_DATA_POL	DATA 电平控制寄存器
0x2e0	SENSOR_INT_POL	传感器中断控制寄存器
0x2e4	LSU_READY_STATUS	READY 状态反馈寄存器
0x300	PRINT_STDLY_COUNT0	LSU0 打印起始延迟寄存器
0x304	PRINT_STDLY_COUNT2	LSU2 打印起始延迟寄存器
0x308	PRINT_STDLY_COUNT4	LSU4 打印起始延迟寄存器
0x30c	PRINT_STDLY_COUNT6	LSU6 打印起始延迟寄存器
0x310	PRINT_STDLY_COUNT1	LSU1 打印起始延迟寄存器
0x314	PRINT_STDLY_COUNT3	LSU3 打印起始延迟寄存器
0x318	PRINT_STDLY_COUNT5	LSU5 打印起始延迟寄存器
0x31c	PRINT_STDLY_COUNT7	LSU7 打印起始延迟寄存器
0x320	PRINT_ENDDLY_COUNT0	LSU0 打印结束延迟寄存器
0x324	PRINT_ENDDLY_COUNT2	LSU2 打印结束延迟寄存器
0x328	PRINT_ENDDLY_COUNT4	LSU4 打印结束延迟寄存器
0x32c	PRINT_ENDDLY_COUNT6	LSU6 打印结束延迟寄存器
0x330	PRINT_ENDDLY_COUNT1	LSU1 打印结束延迟寄存器
0x334	PRINT_ENDDLY_COUNT3	LSU3 打印结束延迟寄存器
0x338	PRINT_ENDDLY_COUNT5	LSU5 打印结束延迟寄存器
0x33c	PRINT_ENDDLY_COUNT7	LSU7 打印结束延迟寄存器

各个配置寄存器具体描述如下：

(1)重置控制寄存器 0-7 (PO_CONTROL)

偏　移：　0x00,0x04,0x08,0x0c,0x10,0x14,0x18,0x1c

复位值：　32’h0

八个重置控制寄存器分别控制对应的八个 LSU。

位域	名称	访问	描述
31:2	reserved	-	保留
1	page_reset	RW	REG 保留配置控制位 向此位写入 1,REG 保留当前配置
0	local_reset	RW	REG 保留配置控制位 向此位写入 1,REG 保留当前配置

(2) 同步控制寄存器 0-7 (PO_SYNC_MODE)

偏　移：　0x20,0x24,0x28,0x2c,0x30,0x34,0x38,0x3c

复位值：　32’h0

八个同步控制寄存器分别控制对应的八个 LSU。

位域	名称	访问	描述
31:2	reserved	-	保留
1	line_sync_polarity	RW	HSYNC 输入控制位 1：HSYNC 上升沿有效，0：HSYNC 下降沿有效 0
0	reserved	-	保留

(3) 输出格式寄存器 0-7 (PO_OUTPUT_FORMAT)

偏　移：　0x40,0x44,0x48,0x4c,0x50,0x54,0x58,0x5c

复位值： 32’h0

八个输出格式控制寄存器分别控制对应的八个 LSU。

位域	名称	访问	描述
31:9	reserved	-	保留
8	byte_inversion	RW	字节内翻转控制位 0：字节不翻转输出，1：字节翻转输出
7:5	swizzle_control	RW	MSB/LSB 控制位 000：MSB 格式输出，001：LSB 格式输出
4	margin_color		页边像素填充控制位 1：页边填充黑点/打激光，0：页边填充白点/不打激光
3:0	reserved	-	保留

(4) 左边距寄存器 0-7 (PO_LEFT_MARGIN)

偏 移： 0x60,0x64,0x68,0x6c,0x70,0x74,0x78,0x7c

复位值： 32’h0

八个左边距寄存器分别控制对应的八个 LSU。

位域	名称	访问	描述
31:16	reserved	-	保留
15:0	left_margin	RW	左边距控制位，配置激光到达左幅面的点数

(5) 横幅面寄存器 0-7 (PO_SCAN_ACTIVE)

偏 移： 0x80,0x84,0x88,0x8c,0x90,0x94,0x98,0x9c

复位值： 32’h0

八个横幅面寄存器分别控制对应的八个 LSU。

位域	名称	访问	描述
31:24	reserved	-	保留
23:0	scan_active	RW	横幅面控制位，配置激光有效的每一行输出点数

(6) 顶边距寄存器 0-7 (PO_TOP_MARGIN)

偏 移： 0xa0,0xa4,0xa8,0xac,0xb0,0xb4,0xb8,0xbc

复位值： 32’h0

八个顶边距寄存器分别控制对应的八个 LSU。

位域	名称	访问	描述
31:2	reserved	-	保留
15:0	top_margin	RW	顶边距控制位，配置激光到达上幅面的线数

(7) 纵幅面寄存器 0-7 (PO_PAGE_ACTIVE)

偏 移： 0xc0,0xc4,0xc8,0xcc,0xd0,0xd4,0xd8,0xdc

复位值： 32’h0

八个纵幅面寄存器分别控制对应的八个 LSU。

位域	名称	访问	描述
31:24	reserved	-	保留
23:0	page_active	RW	纵幅面控制位，配置激光有效的输出线数

(8) SH/DATA 使能寄存器 0-7 (PO_LSAUX_ENABLE)

偏　移：　0xe0,0xe4,0xe8,0xec,0xf0,0xf4,0xf8,0xfc

复位值：　32’h1

八个 SH/DATA 使能寄存器分别控制对应的八个 LSU。

位域	名称	访问	描述
31:1	reserved	-	保留
0	enable	RW	使能控制位 1：SH/DATA 信号有效，0：SH/DATA 信号无效

(9) SH 拉高寄存器 0-7 (PO_LSAUX_SHGOHI)

偏　移：　0x100,0x104,0x108,0x10c,0x110,0x114,0x118,0x11c

复位值：　32’h0

八个 SH 拉高寄存器分别控制对应的八个 LSU。

位域	名称	访问	描述
31:0	gohi	RW	SH 拉高控制位，从 HSYNC 信号有效开始计数，经过一段延迟后将 SH 信号拉高

(10) SH 拉低寄存器 0-7 (PO_LSAUX_SHGOLO)

偏　移：　0x120,0x124,0x128,0x12c,0x130,0x134,0x138,0x13c

复位值：　32’h0

八个 SH 拉低寄存器分别控制对应的八个 LSU。

位域	名称	访问	描述
31:0	golo	RW	SH 拉低控制位，从 HSYNC 信号有效开始计数，经过一段延迟后将 SH 信号拉低

(11) DATA 拉高寄存器 0-7 (PO_LSAUX_DATAGOHI)

偏　移：　0x140,0x144,0x148,0x14c,0x150,0x154,0x158,0x15c

复位值：　32’h0

八个 DATA 拉高寄存器分别控制对应的八个 LSU。

位域	名称	访问	描述
31:0	gohi	RW	DATA 拉高控制位，从 HSYNC 信号有效开始计数，经过一段延迟后将 DATA 信号拉高

(12) DATA 拉低寄存器 0-7 (PO_LSAUX_DATAGOLO)

偏　移：　0x160,0x164,0x168,0x16c,0x170,0x174,0x178,0x17c

复位值：　32’h0

八个 DATA 拉低寄存器分别控制对应的八个 LSU。

位域	名称	访问	描述
31:0	golo	RW	DATA 拉低控制位，从 HSYNC 信号有效开始计数，经过一段延迟后将 DATA 信号拉低

(13) 基地址寄存器 0-7 (PO_BASE)

偏　移：　0x180,0x184,0x188,0x18c,0x190,0x194,0x198,0x19c

复位值：　32’h0

八个基地址寄存器分别控制对应的八个 LSU。

位域	名称	访问	描述
31:30	reserved	-	保留

29:0	base	RW	基地址控制位，设置对应的 LSU 数据的内存基地址

(14) 步长寄存器 0-7 (PO_STRIDE)

偏　移： 0x1a0,0x1a4,0x1a8,0x1ac,0x1b0,0x1b4,0x1b8,0x1bc

复位值： 32’h0

八个步长寄存器分别控制对应的八个 LSU。

位域	名称	访问	描述
31:30	reserved	-	保留
29:0	stride	RW	步长控制位，设置对应的 LSU 数据的内存步长

(15) 总线数寄存器 (PO_YSIZE)

偏　移： 0x1c0,0x1c4,0x1c8,0x1cc,0x1d0,0x1d4,0x1d8,0x1dc

复位值： 32’h0

八个总线数寄存器分别控制对应的八个 LSU。

位域	名称	访问	描述
31:16	reserved	-	保留
15:0	ysize	RW	总线数控制位，设置对应的 LSU 数据的内存总线数

(16) 输出线数寄存器 (PO_YVALUE)

偏　移： 0x1e0,1e4,1e8,1ec,1f0,1f4,1f8,1fc

复位值： 32’h0

八个已输出线数寄存器分别反馈对应的八个 LSU 的输出线数。

位域	名称	访问	描述
31:16	reserved	-	保留
15:0	yvalue	RO	已输出线数控制位，用于反馈给软件已经输出的线数

(17) 剩余线数寄存器 (PO_YVOLUME)

偏　移： 0x200,0x204,0x208,0x20c,0x210,0x214,0x218,0x21c

复位值： 32’h0

八个剩余线数寄存器分别反馈对应的八个 LSU 的剩余线数。

位域	名称	访问	描述
31:16	reserved	-	保留
15:0	yvolume	RO	剩余线数控制位，用于反馈给软件剩余的线数

(18) 打印开始中断清除寄存器 (PO_STINT_CLEAR)

偏　移： 0x220

复位值： 32’h0

位域	名称	访问	描述
31:24	reserved	-	保留
23	po7_stint_clear	RW	LSU7 打印开始中断清除控制: 控制 LSU7 的打印开始中断是否清除(高电平有效,中断使能开启时配置有效)
22	po5_stint_clear	RW	LSU5 打印开始中断清除控制: 控制 LSU5 的打印开始中断是否清除(高电平有效,中断使能开启时配置有效)
21	po3_stint_clear	RW	LSU3 打印开始中断清除控制: 控制 LSU3 的打印开始中断是否清除(高电平有效,中断使能开

			启时配置有效)
20	po1_stint_clear	RW	LSU1 打印开始中断清除控制: 控制 LSU1 的打印开始中断是否清除(高电平有效,中断使能开启时配置有效)
19	po6_stint_clear	RW	LSU6 打印开始中断清除控制: 控制 LSU6 的打印开始中断是否清除(高电平有效,中断使能开启时配置有效)
18	po4_stint_clear	RW	LSU4 打印开始中断清除控制: 控制 LSU4 的打印开始中断是否清除(高电平有效,中断使能开启时配置有效)
17	po2_stint_clear	RW	LSU2 打印开始中断清除控制: 控制 LSU2 的打印开始中断是否清除(高电平有效,中断使能开启时配置有效)
16	po0_stint_clear	RW	LSU0 打印开始中断清除控制: 控制 LSU0 的打印开始中断是否清除(高电平有效,中断使能开启时配置有效)
15:0	reserved	-	保留

(19) 打印开始中断开启寄存器 (PO_STINT_SET)

偏　移： 0x224

复位值： 32’h0

位域	名称	访问	描述
31:24	reserved	-	保留
23	po7_stint_set	RW	LSU7 打印开始中断置位控制: 控制 LSU7 的打印开始中断是否置位(高电平有效,中断使能开启时配置有效)
22	po5_stint_set	RW	LSU5 打印开始中断置位控制: 控制 LSU5 的打印开始中断是否置位(高电平有效,中断使能开启时配置有效)
21	po3_stint_set	RW	LSU3 打印开始中断置位控制: 控制 LSU3 的打印开始中断是否置位(高电平有效,中断使能开启时配置有效)
20	po1_stint_set	RW	LSU1 打印开始中断置位控制: 控制 LSU1 的打印开始中断是否置位(高电平有效,中断使能开启时配置有效)
19	po6_stint_set	RW	LSU6 打印开始中断置位控制: 控制 LSU6 的打印开始中断是否置位(高电平有效,中断使能开启时配置有效)
18	po4_stint_set	RW	LSU4 打印开始中断置位控制: 控制 LSU4 的打印开始中断是否置位(高电平有效,中断使能开启时配置有效)
17	po2_stint_set	RW	LSU2 打印开始中断置位控制: 控制 LSU2 的打印开始中断是否置位(高电平有效,中断使能开启时配置有效)
16	po0_stint_set	RW	LSU0 打印开始中断置位控制: 控制 LSU0 的打印开始中断是否置位(高电平有效,中断使能开启时配置有效)
15:0	reserved	-	保留

(20) 打印开始中断使能寄存器 (PO_STINT_ENABLE)

偏　移： 0x228

复位值： 32’h0

位域	名称	访问	描述
31:24	reserved	-	保留
23	po7_stint_enable	RW	LSU7 打印开始中断使能控制:

			控制 LSU7 的打印开始中断是否使能(高电平有效)
22	po5_stint_enable	RW	LSU5 打印开始中断使能控制: 控制 LSU5 的打印开始中断是否使能(高电平有效)
21	po3_stint_enable	RW	LSU3 打印开始中断使能控制: 控制 LSU3 的打印开始中断是否使能(高电平有效)
20	po1_stint_enable	RW	LSU1 打印开始中断使能控制: 控制 LSU1 的打印开始中断是否使能(高电平有效)
19	po6_stint_enable	RW	LSU6 打印开始中断使能控制: 控制 LSU6 的打印开始中断是否使能(高电平有效)
18	po4_stint_enable	RW	LSU4 打印开始中断使能控制: 控制 LSU4 的打印开始中断是否使能(高电平有效)
17	po2_stint_enable	RW	LSU2 打印开始中断使能控制: 控制 LSU2 的打印开始中断是否使能(高电平有效)
16	po0_stint_enable	RW	LSU0 打印开始中断使能控制: 控制 LSU0 的打印开始中断是否使能(高电平有效)
15:0	reserved	-	保留

(21) 打印开始中断状态寄存器 (PO_STINT_STATUS)

偏　移： 0x22c

复位值： 32’h0

位域	名称	访问	描述
31:24	reserved	-	保留
23	po7_stint_status	RO	LSU7 打印开始中断状态反馈: 显示 LSU7 的打印开始中断状态(高电平有效)
22	po5_stint_status	RO	LSU5 打印开始中断状态反馈: 显示 LSU5 的打印开始中断状态(高电平有效)
21	po3_stint_status	RO	LSU3 打印开始中断状态反馈: 显示 LSU3 的打印开始中断状态(高电平有效)
20	po1_stint_status	RO	LSU1 打印开始中断状态反馈: 显示 LSU1 的打印开始中断状态(高电平有效)
19	po6_stint_status	RO	LSU6 打印开始中断状态反馈: 显示 LSU6 的打印开始中断状态(高电平有效)
18	po4_stint_status	RO	LSU4 打印开始中断状态反馈: 显示 LSU4 的打印开始中断状态(高电平有效)
17	po2_stint_status	RO	LSU2 打印开始中断状态反馈: 显示 LSU2 的打印开始中断状态(高电平有效)
16	po0_stint_status	RO	LSU0 打印开始中断状态反馈: 显示 LSU0 的打印开始中断状态(高电平有效)
15:0	reserved	-	保留

(22) 打印结束中断清除寄存器 (PO_ENDINT_CLEAR)

偏　移： 0x230

复位值： 32’h0

位域	名称	访问	描述
31:24	reserved	-	保留
23	po7_endint_clear	RW	LSU7 打印结束中断清除控制: 控制 LSU7 的打印结束中断是否清除(高电平有效,中断使能开启时配置有效)
22	po5_endint_clear	RW	LSU5 打印结束中断清除控制: 控制 LSU5 的打印结束中断是否清除(高电平有效,中断使能开启时配置有效)
21	po3_endint_clear	RW	LSU3 打印结束中断清除控制: 控制 LSU3 的打印结束中断是否清除(高电平有效,中断使能开启时配置有效)

20	po1_endint_clear	RW	LSU1 打印结束中断清除控制: 控制 LSU1 的打印结束中断是否清除(高电平有效,中断使能开启时配置有效)
19	po6_endint_clear	RW	LSU6 打印结束中断清除控制: 控制 LSU6 的打印结束中断是否清除(高电平有效,中断使能开启时配置有效)
18	po4_endint_clear	RW	LSU4 打印结束中断清除控制: 控制 LSU4 的打印结束中断是否清除(高电平有效,中断使能开启时配置有效)
17	po2_endint_clear	RW	LSU2 打印结束中断清除控制: 控制 LSU2 的打印结束中断是否清除(高电平有效,中断使能开启时配置有效)
16	po0_endint_clear	RW	LSU0 打印结束中断清除控制: 控制 LSU0 的打印结束中断是否清除(高电平有效,中断使能开启时配置有效)
15:0	reserved	-	保留

(23) 打印结束中断开启寄存器 (PO_ENDINT_SET)

偏　移： 0x234

复位值： 32’h0

位域	名称	访问	描述
31:24	reserved	-	保留
23	po7_endint_set	RW	LSU7 打印结束中断置位控制: 控制 LSU7 的打印结束中断是否置位(高电平有效,中断使能开启时配置有效)
22	po5_endint_set	RW	LSU5 打印结束中断置位控制: 控制 LSU5 的打印结束中断是否置位(高电平有效,中断使能开启时配置有效)
21	po3_endint_set	RW	LSU3 打印结束中断置位控制: 控制 LSU3 的打印结束中断是否置位(高电平有效,中断使能开启时配置有效)
20	po1_endint_set	RW	LSU1 打印结束中断置位控制: 控制 LSU1 的打印结束中断是否置位(高电平有效,中断使能开启时配置有效)
19	po6_endint_set	RW	LSU6 打印结束中断置位控制: 控制 LSU6 的打印结束中断是否置位(高电平有效,中断使能开启时配置有效)
18	po4_endint_set	RW	LSU4 打印结束中断置位控制: 控制 LSU4 的打印结束中断是否置位(高电平有效,中断使能开启时配置有效)
17	po2_endint_set	RW	LSU2 打印结束中断置位控制: 控制 LSU2 的打印结束中断是否置位(高电平有效,中断使能开启时配置有效)
16	po0_endint_set	RW	LSU0 打印结束中断置位控制: 控制 LSU0 的打印结束中断是否置位(高电平有效,中断使能开启时配置有效)
15:0	reserved	-	保留

(24) 打印结束中断使能寄存器 (PO_ENDINT_ENABLE)

偏　移： 0x238

复位值： 32’h0

位域	名称	访问	描述
31:24	reserved	-	保留
23	po7_endint_enable	RW	LSU7 打印结束中断使能控制: 控制 LSU7 的打印结束中断是否使能(高电平有效)

22	po5_endint_enable	RW	LSU5 打印结束中断使能控制: 控制 LSU5 的打印结束中断是否使能(高电平有效)
21	po3_endint_enable	RW	LSU3 打印结束中断使能控制: 控制 LSU3 的打印结束中断是否使能(高电平有效)
20	po1_endint_enable	RW	LSU1 打印结束中断使能控制: 控制 LSU1 的打印结束中断是否使能(高电平有效)
19	po6_endint_enable	RW	LSU6 打印结束中断使能控制: 控制 LSU6 的打印结束中断是否使能(高电平有效)
18	po4_endint_enable	RW	LSU4 打印结束中断使能控制: 控制 LSU4 的打印结束中断是否使能(高电平有效)
17	po2_endint_enable	RW	LSU2 打印结束中断使能控制: 控制 LSU2 的打印结束中断是否使能(高电平有效)
16	po0_endint_enable	RW	LSU0 打印结束中断使能控制: 控制 LSU0 的打印结束中断是否使能(高电平有效)
15:0	reserved	-	保留

(25) 打印结束中断状态寄存器 (PO_ENDINT_STATUS)

偏　移： 0x23c

复位值： 32’h0

位域	名称	访问	描述
31:24	reserved	-	保留
23	po7_endint_status	RO	LSU7 打印结束中断状态反馈: 显示 LSU7 的打印结束中断状态(高电平有效)
22	po5_endint_status	RO	LSU5 打印结束中断状态反馈: 显示 LSU5 的打印结束中断状态(高电平有效)
21	po3_endint_status	RO	LSU3 打印结束中断状态反馈: 显示 LSU3 的打印结束中断状态(高电平有效)
20	po1_endint_status	RO	LSU1 打印结束中断状态反馈: 显示 LSU1 的打印结束中断状态(高电平有效)
19	po6_endint_status	RO	LSU6 打印结束中断状态反馈: 显示 LSU6 的打印结束中断状态(高电平有效)
18	po4_endint_status	RO	LSU4 打印结束中断状态反馈: 显示 LSU4 的打印结束中断状态(高电平有效)
17	po2_endint_status	RO	LSU2 打印结束中断状态反馈: 显示 LSU2 的打印结束中断状态(高电平有效)
16	po0_endint_status	RO	LSU0 打印结束中断状态反馈: 显示 LSU0 的打印结束中断状态(高电平有效)
15:0	reserved	-	保留

(26) 激光控制寄存器 (PO_START_RESET)

偏　移： 0x240

复位值： 32’h0

位域	名称	访问	描述
31:13	reserved	-	保留
12	lsu_reset	RW	激光器重置控制位，控制激光器状态重置 1-激光器状态重置，0-激光器解除重置
11	start_motor6/7	RW	启动马达 6/7 的启动配置位，高电平有效
10	start_motor4/5	RW	启动马达 4/5 的启动配置位，高电平有效
9	start_motor2/3	RW	启动马达 2/3 的启动配置位，高电平有效
8	start_motor0/1	RW	启动马达 0/1 的启动配置位，高电平有效
7	start_laser7	RW	启动激光器 7 的启动配置位，高电平有效
6	start_laser5	RW	启动激光器 5 的启动配置位，高电平有效
5	start_laser3	RW	启动激光器 3 的启动配置位，高电平有效

4	start_laser1	RW	启动激光器 1 的启动配置位，高电平有效
3	start_laser6	RW	启动激光器 6 的启动配置位，高电平有效
2	start_laser4	RW	启动激光器 4 的启动配置位，高电平有效
1	start_laser2	RW	启动激光器 2 的启动配置位，高电平有效
0	start_laser0	RW	启动激光器 0 的启动配置位，高电平有效

(27) 马达转速寄存器 (FREQUENCY_TEMP_COUNT)

偏　移： 0x250, 0x254, 0x258, 0x25c

复位值： 32’h0

四个马达转速寄存器分别控制对应的四个 MOTOR，该分频配置基准时钟为 LSU 机芯模块输入时钟，即：DDR-PLL 中 NETWORK 输出时钟经两级分频后所输出时钟。

位域	名称	访问	描述
31:0	frequency_temp_count	RW	马达转速时钟分频控制位，控制激光马达的转速时钟分频

(28) 镜像输出寄存器 (PO_BUFFER_CONTROL)

偏　移： 0x260,0x264,0x268,0x26c,0x270,0x274,0x278,0x27c

复位值： 32’h0

八个镜像输出寄存器分别控制对应的八个 LSU，以 128 比特位为基本单位。

位域	名称	访问	描述
31:7	reserved	-	保留
6	mirror	RW	镜像控制位: 1：数据自右向左，0：数据自左向右
5:0	reserved	-	保留

(29) 点延迟寄存器 0-7 (DOT_DELAY)

偏　移： 0x280,0x284,0x288,0x28c,0x290,0x294,0x298,0x29c

复位值： 32’h0

八个点延迟寄存器分别控制对应的八个 LSU。

位域	名称	访问	描述
31:16	reserved	-	保留
15:0	dot_delay	RW	点延迟控制位，控制每个点的延迟

(30) 使能延迟寄存器 (ENB_DELAY)

偏　移： 0x2a0,0x2a4,0x2a8,0x2ac,0x2b0,0x2b4,0x2b8,0x2bc

复位值： 32’h0

八个使能延迟寄存器分别控制对应的八个 LSU。

位域	名称	访问	描述
31:0	enb_delay	RW	使能延迟控制位: 控制激光器使能信号的延迟

(31) DMA 中断清除寄存器 (DMA_INT_CLEAR)

偏　移： 0x2c0

复位值： 32’h0

位域	名称	访问	描述
31:8	reserved	-	保留
7	po7_dma_int_clear	RW	LSU7 的 DMA 中断清除控制:

			控制 LSU7 的 DMA 中断是否清除(高电平有效,中断使能开启时配置有效)。
6	po5_dma_int_clear	RW	LSU5 的 DMA 中断清除控制: 控制 LSU5 的 DMA 中断是否清除(高电平有效,中断使能开启时配置有效)。
5	po3_dma_int_clear	RW	LSU3 的 DMA 中断清除控制: 控制 LSU3 的 DMA 中断是否清除(高电平有效,中断使能开启时配置有效)。
4	po1_dma_int_clear	RW	LSU1 的 DMA 中断清除控制: 控制 LSU1 的 DMA 中断是否清除(高电平有效,中断使能开启时配置有效)。
3	po6_dma_int_clear	RW	LSU6 的 DMA 中断清除控制: 控制 LSU6 的 DMA 中断是否清除(高电平有效,中断使能开启时配置有效)。
2	po4_dma_int_clear	RW	LSU4 的 DMA 中断清除控制: 控制 LSU4 的 DMA 中断是否清除(高电平有效,中断使能开启时配置有效)。
1	po2_dma_int_clear	RW	LSU2 的 DMA 中断清除控制: 控制 LSU2 的 DMA 中断是否清除(高电平有效,中断使能开启时配置有效)。
0	po0_dma_int_clear	RW	LSU0 的 DMA 中断清除控制: 控制 LSU0 的 DMA 中断是否清除(高电平有效,中断使能开启时配置有效)。

(32) DMA 中断置位寄存器 (DMA_INT_SET)

偏　移： 0x2c4

复位值： 32’h0

位域	名称	访问	描述
31:8	reserved	-	保留
7	po7_dma_int_set	RW	LSU7 的 DMA 中断置位控制: 控制 LSU7 的 DMA 中断是否置位(高电平有效,中断使能开启时配置有效)。
6	po5_dma_int_set	RW	LSU5 的 DMA 中断置位控制: 控制 LSU5 的 DMA 中断是否置位(高电平有效,中断使能开启时配置有效)。
5	po3_dma_int_set	RW	LSU3 的 DMA 中断置位控制: 控制 LSU3 的 DMA 中断是否置位(高电平有效,中断使能开启时配置有效)。
4	po1_dma_int_set	RW	LSU1 的 DMA 中断置位控制: 控制 LSU1 的 DMA 中断是否置位(高电平有效,中断使能开启时配置有效)。
3	po6_dma_int_set	RW	LSU6 的 DMA 中断置位控制: 控制 LSU6 的 DMA 中断是否置位(高电平有效,中断使能开启时配置有效)。
2	po4_dma_int_set	RW	LSU4 的 DMA 中断置位控制: 控制 LSU4 的 DMA 中断是否置位(高电平有效,中断使能开启时配置有效)。
1	po2_dma_int_set	RW	LSU2 的 DMA 中断置位控制: 控制 LSU2 的 DMA 中断是否置位(高电平有效,中断使能开启时配置有效)。
0	po0_dma_int_set	RW	LSU0 的 DMA 中断置位控制: 控制 LSU0 的 DMA 中断是否置位(高电平有效,中断使能开启时配置有效)。

(33) DMA 中断使能寄存器 (DMA_INT_ENABLE)

偏　移： 0x2c8

复位值： 32’h0

位域	名称	访问	描述
31:8	reserved	-	保留
7	po7_dma_int_enable	RW	LSU7 的 DMA 中断使能控制: 控制 LSU7 的 DMA 中断是否使能(高电平有效)。
6	po5_dma_int_enable	RW	LSU5 的 DMA 中断使能控制: 控制 LSU5 的 DMA 中断是否使能(高电平有效)。
5	po3_dma_int_enable	RW	LSU3 的 DMA 中断使能控制: 控制 LSU3 的 DMA 中断是否使能(高电平有效)。
4	po1_dma_int_enable	RW	LSU1 的 DMA 中断使能控制: 控制 LSU1 的 DMA 中断是否使能(高电平有效)。
3	po6_dma_int_enable	RW	LSU6 的 DMA 中断使能控制: 控制 LSU6 的 DMA 中断是否使能(高电平有效)。
2	po4_dma_int_enable	RW	LSU4 的 DMA 中断使能控制: 控制 LSU4 的 DMA 中断是否使能(高电平有效)。
1	po2_dma_int_enable	RW	LSU2 的 DMA 中断使能控制: 控制 LSU2 的 DMA 中断是否使能(高电平有效)。
0	po0_dma_int_enable	RW	LSU0 的 DMA 中断使能控制: 控制 LSU0 的 DMA 中断是否使能(高电平有效)。

(34) DMA 中断状态寄存器 (DMA_INT_STATUS)

偏 移： 0x2cc

复位值： 32’h0

位域	名称	访问	描述
31:8	reserved	-	保留
7	po7_dma_int_status	RW	LSU7 的 DMA 中断状态: 显示 LSU7 的 DMA 中断状态(高电平有效)。
6	po5_dma_int_status	RW	LSU5 的 DMA 中断状态: 显示 LSU5 的 DMA 中断状态(高电平有效)。
5	po3_dma_int_status	RW	LSU3 的 DMA 中断状态: 显示 LSU3 的 DMA 中断状态(高电平有效)。
4	po1_dma_int_status	RW	LSU1 的 DMA 中断状态: 显示 LSU1 的 DMA 中断状态(高电平有效)。
3	po6_dma_int_status	RW	LSU6 的 DMA 中断状态: 显示 LSU6 的 DMA 中断状态(高电平有效)。
2	po4_dma_int_status	RW	LSU4 的 DMA 中断状态: 显示 LSU4 的 DMA 中断状态(高电平有效)。
1	po2_dma_int_status	RW	LSU2 的 DMA 中断状态: 显示 LSU2 的 DMA 中断状态(高电平有效)。
0	po0_dma_int_status	RW	LSU0 的 DMA 中断状态: 显示 LSU0 的 DMA 中断状态(高电平有效)。

(35) 机芯启动电平控制寄存器 (LSU_START_POL)

偏 移： 0x2d0

复位值： 32’h0

位域	名称	访问	描述
31:8	reserved	-	保留
7	lsu7_start_pol	RW	LSU7 的 START 电平控制: 1：START 高电平有效，0：START 低电平有效
6	lsu5_start_pol	RW	LSU5 的 START 电平控制: 1：START 高电平有效，0：START 低电平有效
5	lsu3_start_pol	RW	LSU3 的 START 电平控制: 1：START 高电平有效，0：START 低电平有效
4	lsu1_start_pol	RW	LSU1 的 START 电平控制:

			1：START 高电平有效，0：START 低电平有效
3	lsu6_start_pol	RW	LSU6 的 START 电平控制: 1：START 高电平有效，0：START 低电平有效
2	lsu4_start_pol	RW	LSU4 的 START 电平控制: 1：START 高电平有效，0：START 低电平有效
1	lsu2_start_pol	RW	LSU2 的 START 电平控制: 1：START 高电平有效，0：START 低电平有效
0	lsu0_start_pol	RW	LSU0 的 START 电平控制: 1：START 高电平有效，0：START 低电平有效

(36) 机芯使能电平控制寄存器 (LSU_ENB_POL)

偏　移：　0x2d4

复位值：　32’h0

位域	名称	访问	描述
31:8	reserved	-	保留
7	lsu7_enb_pol	RW	LSU7 的 ENB 电平控制: 1：ENB 高电平有效，0：ENB 低电平有效
6	lsu5_enb_pol	RW	LSU5 的 ENB 电平控制: 1：ENB 高电平有效，0：ENB 低电平有效
5	lsu3_enb_pol	RW	LSU3 的 ENB 电平控制: 1：ENB 高电平有效，0：ENB 低电平有效
4	lsu1_enb_pol	RW	LSU1 的 ENB 电平控制: 1：ENB 高电平有效，0：ENB 低电平有效
3	lsu6_enb_pol	RW	LSU6 的 ENB 电平控制: 1：ENB 高电平有效，0：ENB 低电平有效
2	lsu4_enb_pol	RW	LSU4 的 ENB 电平控制: 1：ENB 高电平有效，0：ENB 低电平有效
1	lsu2_enb_pol	RW	LSU2 的 ENB 电平控制: 1：ENB 高电平有效，0：ENB 低电平有效
0	lsu0_enb_pol	RW	LSU0 的 ENB 电平控制: 1：ENB 高电平有效，0：ENB 低电平有效

(37) 机芯 SH 电平控制寄存器 (LSU_SH_POL)

偏　移：　0x2d8

复位值：　32’h0

位域	名称	访问	描述
31:8	reserved	-	保留
7	lsu7_sh_pol	RW	LSU7 的 SH 电平控制: 1：SH 高电平有效，0：SH 低电平有效
6	lsu5_sh_pol	RW	LSU5 的 SH 电平控制: 1：SH 高电平有效，0：SH 低电平有效
5	lsu3_sh_pol	RW	LSU3 的 SH 电平控制: 1：SH 高电平有效，0：SH 低电平有效
4	lsu1_sh_pol	RW	LSU1 的 SH 电平控制: 1：SH 高电平有效，0：SH 低电平有效
3	lsu6_sh_pol	RW	LSU6 的 SH 电平控制: 1：SH 高电平有效，0：SH 低电平有效
2	lsu4_sh_pol	RW	LSU4 的 SH 电平控制: 1：SH 高电平有效，0：SH 低电平有效
1	lsu2_sh_pol	RW	LSU2 的 SH 电平控制: 1：SH 高电平有效，0：SH 低电平有效
0	lsu0_sh_pol	RW	LSU0 的 SH 电平控制: 1：SH 高电平有效，0：SH 低电平有效

(38) 机芯 DATA 电平控制寄存器 (LSU_DATA_POL)

偏　移：　0x2dc

复位值：　32’h0

位域	名称	访问	描述
31:8	reserved	-	保留
7	lsu7_data_pol	RW	LSU7 的 DATA 电平控制: 1：DATA 高电平有效，0：DATA 低电平有效
6	lsu5_data_pol	RW	LSU5 的 DATA 电平控制: 1：DATA 高电平有效，0：DATA 低电平有效
5	lsu3_data_pol	RW	LSU3 的 DATA 电平控制: 1：DATA 高电平有效，0：DATA 低电平有效
4	lsu1_data_pol	RW	LSU1 的 DATA 电平控制: 1：DATA 高电平有效，0：DATA 低电平有效
3	lsu6_data_pol	RW	LSU6 的 DATA 电平控制: 1：DATA 高电平有效，0：DATA 低电平有效
2	lsu4_data_pol	RW	LSU4 的 DATA 电平控制: 1：DATA 高电平有效，0：DATA 低电平有效
1	lsu2_data_pol	RW	LSU2 的 DATA 电平控制: 1：DATA 高电平有效，0：DATA 低电平有效
0	lsu0_data_pol	RW	LSU0 的 DATA 电平控制: 1：DATA 高电平有效，0：DATA 低电平有效

(39) 传感器中断电平控制寄存器 (SENSOR_INT_POL)

偏　移：　0x2e0

复位值：　32’h0

位域	名称	访问	描述
31:1	reserved	-	保留
0	sensor_int_pol	RW	传感器中断电平控制: 1：INT 低电平有效，0：INT 高电平有效

(40) 机芯 READY 反馈寄存器 (LSU_READY_STATUS)

偏　移：　0x2e4

复位值：　32’h0

位域	名称	访问	描述
31:4	reserved	-	保留
3	lsu6/7_ready_status	RO	LSU6/7 的 READY 信号反馈: 显示 LSU6/7 的 READY 信号状态。
2	lsu4/5_ready_status	RO	LSU4/5 的 READY 信号反馈: 显示 LSU4/5 的 READY 信号状态。
1	lsu2/3_ready_status	RO	LSU2/3 的 READY 信号反馈: 显示 LSU2/3 的 READY 信号状态。
0	lsu0/1_ready_status	RO	LSU0/1 的 READY 信号反馈: 显示 LSU0/1 的 READY 信号状态。

(41) 打印起始延迟计数寄存器 (PRINT_STDLY_COUNT)

偏　移：　0x300，0x304，0x308，0x30c，0x310，0x314，0x318，0x31c

复位值：　32’h0

八个打印起始延迟计数器分别对应的八个 LSU。

位域	名称	访问	描述
31:0	print_stdly_count	RW	机芯打印起始延迟计数器: 分别对应每个机芯打印起始延迟计数值。

(42) 打印结束延迟计数寄存器 (PRINT_ENDDLY_COUNT)

偏　移：　0x320，0x324，0x328，0x32c，0x330，0x334，0x338，0x33c

复位值：　32’h0

八个打印结束延迟计数器分别对应的八个 LSU。

位域	名称	访问	描述
31:0	print_enddly_count	RW	机芯打印结束延迟计数器: 分别对应每个机芯打印结束延迟计数值。

14.2.2 JBIG85 解码模块寄存器

JBIG85 模块有 4 个相同的 JBIG 解码单元，四个模块均具有相同的配置寄存器格式。

物理地址	设备	备注
0x1f0c_2000	JBIG0	四个模块寄存器配置方式相同，区别在于寄存器基地址偏移不同
0x1f0c_2800	JBIG1	
0x1f0c_3000	JBIG2	
0x1f0c_3800	JBIG3	

JBIG85 每个模块包括 10 个可以通过配置总线进行读写的寄存器。

偏移地址	位宽	寄存器名称	读写	说明
0x0	32	jbig_top_cfg	R/W	JBIG 模块的整体配置位，
0x4	32	pscd_addr	R/W	当前 JBIG 模块待解码原始数据的存储地址
0x8	32	pscd_data_length	R/W	当前 JBIG 模块待解码数据的长度，单位为字节
0xc	32	output_addr	R/W	当前 JBIG 模块解码后数据存储地址
0x10	1	start	R/W	开始当前 JBIG 模块的解码，高电平有效
0x14	1	error_processed_reg	R/W	错误清除位，出现错误并处理结束后，写 1 清除
0x18	32	error_code_reg	R	错误码状态寄存器
0x1c	1	apb_clear	W	清空当前 JBIG 模块的所有配置项，高电平有效
0x20	1	int_reg_clear	W	清除当前 JBIG 模块的中断位，高电平有效
0x24	1	int_reg	R	读取当前 JBIG 模块的中断位

(1) jbig_top_cfg 寄存器

偏移地址：0x0。

位域	名称	复位值	访问	说明
31:9	Reserved	-	RO	保留
8	int_oen	1’b0	R_W	4 个 JBIG 模块的中断位合并为 1 位输出， 0：将 4 位中断位做 AND 操作，所有四个模块均为中断，才输出中断，可用作没有数据错误的正常打印方式。 1：将 4 位中断位做 OR 操作，所有模块有一个中断，则立即输出中断，可用作有数据错误的调试方式。

7:4	int_en	4’b0	R_W	配置 4 个 JBIG 模块的中断使能位 1：打开使能 0：关闭使能 [7]表示 JBIG3 [6]表示 JBIG2 [5]表示 JBIG1 [4]表示 JBIG0
3:0	jbig_en	4’b00	R/W	配置 4 个 JBIG 模块的使能位 1：打开使能 0：关闭使能 [3]表示 JBIG3 [2]表示 JBIG2 [1]表示 JBIG1 [0]表示 JBIG0

(2) error_code_reg 寄存器

偏移地址：0x18。

JBIG 错误码，数据 s->buffer 为 jbig85 数据前 20 字节数据，为当前 jbig85 数据的描述符，x0/y0/l0/mx/options 为通过 buffer 计算得来的数据。

位域	名称	复位值	访问	说明
31:24	jbig3_err_code	8’b0	R_W	JBIG3 的错误码，位域含义等同[7:0]
23:16	jbig2_err_code	8’b0	R_W	JBIG2 的错误码，位域含义等同[7:0]
15:8	jbig1_err_code	8’b0	R_W	JBIG1 的错误码，位域含义等同[7:0]
7:0	jbig0_err_code	8’b0	R/W	JBIG0 的错误码，数据 s 为 jbig85 数据的描述符，具体表示参考 jbig85_c_code [7:4] == 4’d6，输入数据包含错误输入 [3:0] == 4’d1：s->buffer[1] < s->buffer[0] [3:0] == 4’d2：s->buffer[3]!=0 [3:0] == 4’d3：s->buffer[18]&0xf0 != 0 [3:0] == 4’d4：s->buffer[19]&80 != 0 [3:0] == 4’d5：s->buffer[2] == 0 [3:0] == 4’d6：s->x0 == 0 [3:0] == 4’d7：s->y0 == 0 [3:0] == 4’d8：s->l0 == 0 [3:0] == 4’d9：s->mx > 127 [7:4] == 4’d7，输入数据不符合 JBIG85 格式 [3:0] == 4’d8：s->buffer[0] != 0 [3:0] == 4’d9：s->buffer[1] != 0 [3:0] == 4’d10：s->buffer[2] != 1 [3:0] == 4’d11：s->buffer[17] != 0 [3:0] == 4’d12：s->buffer[18] != 0 [3:0] == 4’d13：s->options &0x17 != 0

14.3 软件编程指南

14.3.1 LSU 控制编程说明

LSU 机芯控制寄存器，每一类寄存器共有八组，分别对应八个激光头。

当使用黑白打印时，只需要配置一组激光头，其他激光头无需配置；使用红黑打印时，需要配置两组激光头，其余无需配置；采用彩色打印时，单端打印头需要配置四组激光头，采用彩色差分打印需配置八组激光头。

在启动打印操作之前，需对激光头功能相关寄存器进行相应配置，具体相关功能项寄存器如下(注：以下配置寄存器单端打印 x=0/2/4/6，差分打印 x=0/1/2/3/4/5/6/7)：

（1）General 通用寄存器

相关寄存器	软件配置说明	
POx_Control	Local_Reset	信号关闭，REG 恢复默认值
	Page_Reset	信号关闭，REG 保留配置值
POx_Sync_Mode	Line_Sync_Polarity	检测 HSYNC 的上升沿还是下降沿(都是下降沿)
	Line_Sync_Source	HSYNC 是芯片输出还是输入(都是输入)
	Page_Sync_Select	开启关闭 PO0123
POx_Output_Mode	DataP_Enable	开启关闭 POx_DataP
	DataP_Clock	电平或差分
	DataN_Enable	开启关闭 POx_DataN
	DataN_Clock	电平或差分
POx_Output_Format	Swizzle_Control	MSB 或 LSB
	Margin_Color	页边像素填充（都是白点/不打激光）

（2）DOT 分频寄存器

相关寄存器	软件配置说明
POx_Dot_Clock_Divide	PO 内部时钟分频系数(基频 1104MHz,分频产生 10MHz~100MHz 内部时钟)

（3）INT 中断相关寄存器

相关寄存器	软件配置说明	
PO_IntFlag_Clear	PO0_INT_CLR	中断清除，在中断函数中关闭，避免连续触发
PO_IntFlag_Set	PO0_INT_SET	中断开启，DMA 输出完毕自动触发
PO_IntEnable	PO0_INT_EN	中断屏蔽，表示中断是否有效
PO_IntStatus(RO)	PO0_INT_STATUS	中断状态，应用程序读取用的

（4）Page Formatting 寄存器

相关寄存器	软件配置说明
POx_Left_Margin	左边距，POx_Dot_Clock_Divide 定义的时钟个数，调节左右幅面
POx_Scan_Active	横幅面，POx_Dot_Clock_Divide 定义的时钟个数，调节输出点数
POx_Top_Margin	顶边距，POx_Dot_Clock_Divide 定义的时钟个数，调节上下幅面
POx_Page_Active	纵幅面，POx_Dot_Clock_Divide 定义的时钟个数，调节输出线数

（5）LSync Auxiliary Signals 寄存器

相关寄存器	软件配置说明
POx_LSAux_Enable	SH 和 DATA 信号开关(一直开启)
POx_LSAux_GoHi0	SH 拉高，POx_Dot_Clock_Divide 定义的时钟个数， 从 Line_Sync_Polarity 定义的下降沿开始计数
POx_LSAux_GoLo1	SH 拉低，POx_Dot_Clock_Divide 定义的时钟个数， 从 Line_Sync_Polarity 定义的下降沿开始计数
POx_LSAux_GoHi1	VIDEO/DATA 拉高，POx_Dot_Clock_Divide 定义的时钟个数， 从 Line_Sync_Polarity 定义的下降沿开始计数
POx_LSAux_GoLo0	VIDEO/DATA 拉低，POx_Dot_Clock_Divide 定义的时钟个数， 从 Line_Sync_Polarity 定义的下降沿开始计数

（5）Pixel Output Memory Buffer 寄存器

相关寄存器	软件配置说明
POx_Base	当前页数据首地址
POx_Stride	当前页数据一线数据偏移量
POx_Ysize	当前页数据总线数
POx_Yvalue (RO)	当前页数据在 DMA 输出过程中，已输出线数
POx_Yvolume (RO)	当前页数据在 DMA 输出过程中，仍剩余线数
POx_Buffer_Control	左右镜像输出
POx_motor_frequency	配置马达频率

上述激光头功能配置寄存器，需要预先配置好，配置顺序不分先后。

在配置好上述寄存器后，可开始启动打印流程，软件具体操作如下：

1. 通过配置 start_reset 寄存器，首先启动激光器马达；
2. 待激光器马达转平稳后，配置 start_reset 寄存器启动激光器；
3. 此时激光器并不会输出数据，而是在等待中断信号的到来；
4. 通过配置一路 GPIO，启动加热管，对定影器进行加热；
5. 待定影器加热到一定温度，可以打印时，配置另一路 GPIO，启动卷纸开关；
6. 包含中断传感器的打印机待纸撞到打印机的中断开关时，根据配置的中断延时输出数据；
7. 不包含中断传感器的需配置中断配置寄存器；
8. 打印完毕，如连续打印则将数据 reset，无需停止马达；

9. 如果非连续打印，则将所有 reset。

14.3.2 JBIG 解码编程说明

操作 1：正常打印配置方式

1. 逐个配置四个 JBIG 模块的 apb_clear 寄存器，清空所有 jbig 配置项；
2. 逐个配置四个 JBIG 模块的 jbig_cfg 寄存器；
3. 配置需要启用的 JBIG 模块的 pscd_addr 寄存器，将待解压缩的 jbig 数据地址写入该寄存器；
4. 配置需要启用的 JBIG 模块的 pscd_length 寄存器，将待解压的 jbig 数据长度写入该寄存器；
5. 配置需要启用的 JBIG 模块的 output_addr 寄存器，将解压后的数据地址写入该寄存器；
6. 配置需要启用的 JBIG 模块的 start 寄存器，开始解压缩。

操作 2：打印结束后获取状态

1. 读取 int_reg 寄存器，如果值为 1，表示打印结束或者出现错误；
2. 读取 error_code_reg 寄存器，如果值为 0，表示没有错误，为正常结束；
3. 配置 int_reg_clear 寄存器；
4. 重复正常打印配置方式，继续下一页打印。

操作 3：打印错误及状态获取

1. 读取 int_reg 寄存器，如果值为 1，表示打印结束或者出现错误；
2. 读取 error_code_reg 寄存器，如果值不为 0，表示有错误，对照错误码表，确定错误类型；
3. 配置 int_reg_clear 寄存器；
4. 修改错误，重复正常打印配置方式，继续下一页打印。

15 HDA 控制器

15.1 功能概述

HDA 控制器主要功能包括各种输入、输出流组合，对 48KHZ 和 44.1KHZ 的采样频率的支持，初始化序列，命令控制通道等。

HDA 控制器主要包括 5 大功能模块，分别为 SDI，SDO，axi_master ,axi_slave,和 reg config。其中 axi_master 和 axi_salve 分别控制了 HDA 中 DMA 的读写通道和对 HDA 进行配置时的 AXI 总线控制情况。Reg config 主要的作用就是对 HDA 中的寄存器进行配置，控制 SDI SDO 的参数和运行情况。SDI 和 SDO 主要是对输入输入流的控制，包括 4 个输入流和 4 个输入流。

15.2 寄存器描述

HDA 控制器内部寄存器的物理地址构成如下：

地址空间	名称	大小
0x1f07_0000 – 0x1f07_ffff	HAD 配置寄存器	64KB

对于 HDA 接口模块，使用时要注意将对应的芯片复用引脚设置为相应的接口功能。

与 HDA 接口相关的引脚复用设置可查询 2.29 节中的外设功能引脚复用关系表，并根据节 5.5.40~节 5.5.59 中 GPIO0~159 复用配置寄存器，配置相应的 GPIO 引脚复用关系，实现对应设备功能引脚。

下表列举了控制器主要的寄存器参数信息。

15.2.1 协议定义的音频控制器寄存器集

偏移开始位置	偏移结束位置	符号	描述
00	01	GCAP	全局能力
02	02	VMIN	次要版本号
03	03	VMAJ	主要版本号
04	05	OUTPAY	输出的载荷能力
06	07	INPAY	输入的载荷能力
08	0B	GCTL	全局控制
0C	0D	WAKEEN	唤醒启用使能
0E	0F	WAKESTS	唤醒启用状态标志
10	11	GSTS	全局状态
12	17	Rsvd	保留
18	19	OUTSTRMPAY	输出流负载能力
1A	1B	INSTRMPAY	输入流负载能力
1C	1F	Rsvd	保留
20	23	INTCTL	中断控制
24	27	INTSTS	中断状态

28	2F	Rsvd	保留
30	33	WALCLK	时钟计数器
34	37	Rsvd	保留
38	3B	SSYNC	系统同步
3C	3F	Rsvd	保留
40	43	CORBLBASE	CORB 基地址的低位
44	47	CORBUBASE	CORB 基地址的高位
48	49	CORBWP	CORB 写指针
4A	4B	CORBRP	CORB 读指针
4C	4C	CORBCTL	CORB 控制寄存器
4D	4D	CORBSTS	CORB 状态寄存器
4E	4E	CORBSIZE	CORB 大小寄存器
4F	4F	Rsvd	保留
50	53	RIRBLBASE	RIRB 基地址的低位
54	57	RIRBUBASE	RIRB 基地址的高位
58	59	RIRBWP	RIRB 写指针
5A	5B	RINTCNT	RIRB 读指针
5C	5C	RIRBCTL	RIRB 控制寄存器
5D	5D	RIRBSTS	RIRB 状态寄存器
5E	5E	RIRBSIZE	RIRB 大小寄存器
5F	5F	Rsvd	保留
60	63	ICOI	立即命令输出接口
64	67	ICII	立即命令输入接口
68	69	ICIS	立即命令状态寄存器
6A	6F	Rsvd	保留
70	73	DPIBLBASE	DMA 在 buf 中的位置低地址
74	77	DPIBUBASE	DMA 在 buf 中的位置高地址
78	7F	Rsvd	保留
80	82	SD0CTL	输入流控制寄存器
83	83	SD0STS	输入流转台寄存器
84	87	SD0LPIB	输入流 link 在 buf 中的位置寄存器
88	8B	SD0CBL	输入流循环 buf 的长度
8C	8D	SD0LVI	输入流最后一个有效的位置
8E	8F	Rsvd	保留
90	91	SD0FIFOD	输入流 FIFO 的大小
92	93	SD0FMT	输入流格式
94	97	Rsvd	保留
98	9B	SD0BDPL	输入流 BDL 的低地址
9C	9F	SD0BDPU	输入流 BDL 的高地址

A0	A2	SD1CTL	输入流控制寄存器
A3	A3	SD1STS	输入流转台寄存器

A4	A7	SD1LPIB	输入流 link 在 buf 中的位置寄存器
A8	AB	SD1CBL	输入流循环 buf 的长度
AC	AD	SD1LVI	输入流最后一个有效的位置
AE	AF	Rsvd	保留
B0	B1	SD1FIFOD	输入流 FIFO 的大小
B2	B3	SD1FMT	输入流格式
B4	B7	Rsvd	保留
B8	BB	SD1BDPL	输入流 BDL 的低地址
BC	BF	SD1BDPU	输入流 BDL 的高地址

C0	C2	SD2CTL	输入流控制寄存器
C3	C3	SD2STS	输入流转台寄存器
C4	C7	SD2LPIB	输入流 link 在 buf 中的位置寄存器
C8	CB	SD2CBL	输入流循环 buf 的长度
CC	CD	SD2LVI	输入流最后一个有效的位置
CE	CF	Rsvd	保留
D0	D1	SD2FIFOD	输入流 FIFO 的大小
D2	D3	SD2FMT	输入流格式
D4	D7	Rsvd	保留
D8	DD	SD2BDPL	输入流 BDL 的低地址
DC	DF	SD2BDPU	输入流 BDL 的高地址

E0	E2	SD3CTL	输入流控制寄存器
E3	E3	SD3STS	输入流转台寄存器
E4	E7	SD3LPIB	输入流 link 在 buf 中的位置寄存器
E8	EB	SD3CBL	输入流循环 buf 的长度
EE	ED	SD3LVI	输入流最后一个有效的位置
EE	EF	Rsvd	保留
F0	F1	SD3FIFOD	输入流 FIFO 的大小
F2	F3	SD3FMT	输入流格式
F4	F7	Rsvd	保留
F8	FF	SD3BDPL	输入流 BDL 的低地址
FC	FF	SD3BDPU	输入流 BDL 的高地址

100	102	SD4CTL	输入流控制寄存器
103	103	SD4STS	输入流转台寄存器
104	107	SD4LPIB	输入流 link 在 buf 中的位置寄存器
108	10B	SD4CBL	输入流循环 buf 的长度
10C	10D	SD4LVI	输入流最后一个有效的位置
10E	10F	Rsvd	保留
110	111	SD4FIFOD	输入流 FIFO 的大小
112	113	SD4FMT	输入流格式

114	117	Rsvd	保留
118	11B	SD4BDPL	输入流 BDL 的低地址
11C	11F	SD4BDPU	输入流 BDL 的高地址

120	122	SD5CTL	输入流控制寄存器
123	123	SD5STS	输入流转台寄存器
124	127	SD5LPIB	输入流 link 在 buf 中的位置寄存器
128	12B	SD5CBL	输入流循环 buf 的长度
12C	12D	SD5LVI	输入流最后一个有效的位置
12E	12F	Rsvd	保留
130	131	SD5FIFOD	输入流 FIFO 的大小
132	133	SD5FMT	输入流格式
134	137	Rsvd	保留
138	13B	SD5BDPL	输入流 BDL 的低地址
13C	13F	SD5BDPU	输入流 BDL 的高地址

140	142	SD6CTL	输入流控制寄存器
143	143	SD6STS	输入流转台寄存器
144	147	SD6LPIB	输入流 link 在 buf 中的位置寄存器
148	14B	SD6CBL	输入流循环 buf 的长度
14C	14D	SD6LVI	输入流最后一个有效的位置
14E	14F	Rsvd	保留
150	151	SD6FIFOD	输入流 FIFO 的大小
152	155	SD6FMT	输入流格式
154	157	Rsvd	保留
158	15B	SD6BDPL	输入流 BDL 的低地址
15C	15F	SD6BDPU	输入流 BDL 的高地址

160	162	SD7CTL	输入流控制寄存器
163	163	SD7STS	输入流转台寄存器
164	167	SD7LPIB	输入流 link 在 buf 中的位置寄存器
168	16B	SD7CBL	输入流循环 buf 的长度
16C	16D	SD7LVI	输入流最后一个有效的位置
16E	16F	Rsvd	保留
170	171	SD7FIFOD	输入流 FIFO 的大小
172	177	SD7FMT	输入流格式
174	177	Rsvd	保留
178	17B	SD7BDPL	输入流 BDL 的低地址
17C	17F	SD7BDPU	输入流 BDL 的高地址

15.2.2 自定义的调试寄存器

SDO	16’h3000	{16’h0,sdo_sendState}	32 位	输出流的状态
	16’h3004	{16’h0,sdoDma4_State}	32 位	输出流 4DMA 状态机的状态
	16’h3008	{16’h0,sdoDma5_State }	32 位	输出流 5DMA 状态机的状态
	16’h300c	{16’h0,sdoDma6_State }	32 位	输出流 6DMA 状态机的状态
	16’h3010	{16’h0,sdoDma7_State }	32 位	输出流 7DMA 状态机的状态
SDI	16’h3014	{16’h0,sdi0_RecvState}	32 位	输入流 0 接受状态机的状态
	16’h3018	{16’h0,sdi1_RecvState}	32 位	输入流 1 接受状态机的状态
	16’h301c	{16’h0,sdi2_RecvState}	32 位	输入流 2 接受状态机的状态
	16’h3020	{16’h0,sdi_rirbState}	32 位	输入流的状态

15.3 内存数据结构

内存数据结构主要包括一下几个部分。

DMA Position in Current Buffer	在当前缓冲区的 DMA 的位置
Buffer Descriptor List	缓冲区描述符列表
Buffer Descriptor List Entry	缓冲区描述符列表条目
Command Output Ring Buffer	命令的输出环形缓冲区
Response Input Ring Buffer	响应输入环形缓冲区
Codec Verb and Response Structures	编解码器的命令和响应结构
Stream Format Structure	流格式结构

16 AC97 控制器

16.1 概述

在系统里一个 AC97 应用系统如 图 16-1 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。

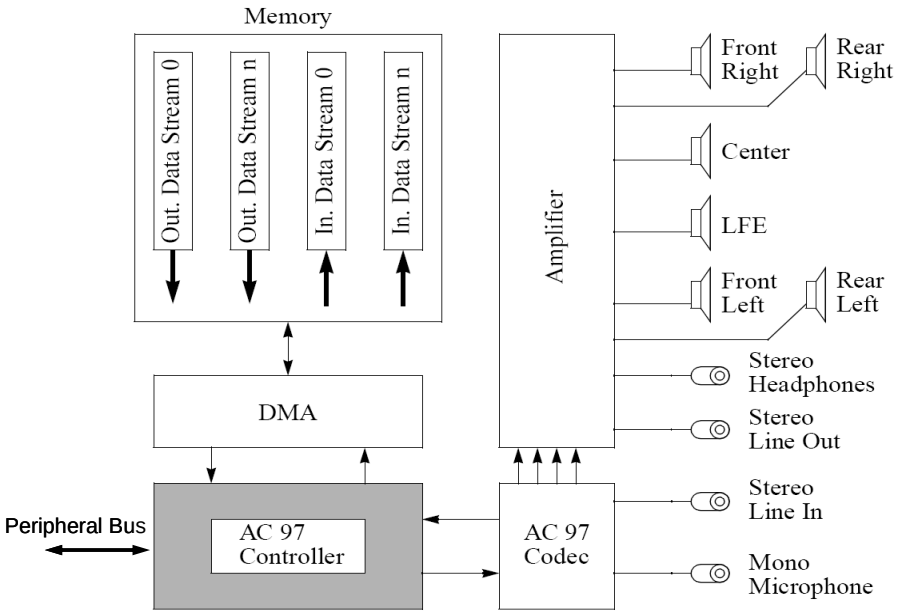


图 16-1 AC97应用系统

16.2 AC97 控制器寄存器

本模块寄存器物理地址基址为 0x1ff54000。

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
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

寄存器名	宽度	偏移量	描述
OC7	20	0x3c	保留
OC10	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
OC_INT_CLR	32	0x60	OC 中断清除寄存器
IC_INT_CLR	32	0x64	IC 中断清除寄存器
WR_INT_CLR	32	0x68	CODEC 写中断清除寄存器
RD_INT_CLR	32	0x6c	CODEC 读中断清除寄存器

对于 AC97 接口模块，使用时要注意将对应的芯片复用引脚设置为相应的接口功能。

与 AC97 接口相关的引脚复用设置可查询节 2.29 中的外设功能引脚复用关系表，并根据节 5.5.40~节 5.5.59 中 GPIO0~159 复用配置寄存器，配置相应的 GPIO 引脚复用关系，实现对应设备功能引脚。

16.2.1 CSR 寄存器

中文名： 状态配置寄存器
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	读此位返回 AC97 子系统的工作状态： 1：AC97 子系统挂起；0：正常工作状态 挂起状态下写入 1，将开始恢复操作直至进入正常工作状态。
0	RST_FORCE	1	W	AC97 冷启动 写入 1 会导致 AC97 Codec 冷启动

16.2.2 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	10	R/W	保留
23:16	Reserved	10	R/W	保留
15:10	OC1_CFG_R	10	R/W	输出通道1。右声道配置：
7:0	OC0_CFG_L	10	R/W	输出通道0。左声道配置：

16.2.3 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	10	R/W	保留
23:16	IC_CFG_MIC	10	R/W	输入通道2：MIC。声道配置
15:10	Reserved	10	R/W	保留
7:0	Reserved	10	R/W	保留

16.2.4 声道格式说明

OCC 和 ICC 寄存器中的声道配置格式相同， 即 16.2.2 中 OCC 寄存器的 OC0_CFG_L 和 OC0_CFG_R 域，以及 16.2.3 中 OCC 寄存器的 IC_CFG_MIC 域，声道格式说明如下：

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 1：DMA打开 0：DMA关闭
5:4	FIFO_THRES	2	R/W	FIFO门限 5：4 输出通道 输入通道 00 FIFO 1/4空 FIFO 1/4满 01 FIFO 1/2空 FIFO 1/2满 10 FIFO 3/4空 FIFO 3/4满 11 FIFO全空 FIFO全满
3:2	SW	2	R/W	采样位数 00：10位 10：16位
1	VSR	1	R/W	采样率 1采样率可变： 0）采样率固定：410KHz（
0	CH_EN	1	R/W	通道使能 1通道打开： 0通道关闭（或者进入节能状态）：

16.2.5 Codec 寄存器访问命令

中文名： Codec 访问命令寄存器
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： 1：读操作，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 寄存器值。 0：写操作。
30:23	Reserved	10	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

16.2.6 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： 0x54/510
复位值： 0x00410000

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
29:10	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

16.2.7 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

16.2.8 OC 中断清除寄存器

中文名： OC 中断清除寄存器
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 的中断状态对应位 bit[7:2]

16.2.9 IC 中断清除寄存器

中文名： IC 中断清除寄存器
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 的中断状态对应位 bit[31:30]

16.2.10 CODEC WRITE 中断清除寄存器

中文名： CODEC WRITE 中断清除寄存器
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]

16.2.11 CODEC READ 中断清除寄存器

中文名： CODEC READ 中断清除寄存器
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]

17 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
- 支持 Firmware Memory Read、Firmware Memory Write 访问类型（单字节）
- 支持 I/O read、I/O write 访问类型
- 支持 Memory 访问类型地址转换
- 支持 Serizlized IRQ 规范，提供 17 个中断源

17.1 LPC 地址空间

LPC 控制器包括四个地址空间：控制寄存器空间、MEM 空间、I/O 空间、BOOT 空间。

表17-1 LPC控制器地址空间分布

地址空间	名称	大小
0x1c00,0000 – 0x1c0f,ffff	LPC Boot	1MB
0x1d00,0000 – 0x1dff,ffff	LPC Memory	16MB
0x1f0d,0000 – 0x1f0d,ffff	LPC I/O	64KB
0x1f0e,0000 – 0x1f0e,00ff	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。这个地址空间支持 LPC Memory 访问类型，访问 1MB 的 Flash，映射到 LPC 总线后的地址为 0xfff0,0000~0xffff,ffff。

LPC 控制寄存器空间用来配置 LPC 控制器，起始地址为 0x1f0e,0000，大小为 256B。

LPC MEM 空间用来访问 LPC 总线上挂载的 Memory/Firmware Memory 设备。处理器发往 LPC MEM 空间的访问会被转换成 LPC 协议的 Memory 访问发往 LPC 总线。LPC 控制器发出哪种类型的 Memory 访问，由 LPC 控制器的控制寄存器决定。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。MEM 访问 8Mbits 地址空间使能配置参考节 5.5.3 通用配置寄存器 2(chip_ctrl2[5])设置，默认打开。

LPC I/O 空间用来访问 LPC 总线上挂载的 I/O 设备，LPC I/O 空间的地址从 PCI I/O 空间的 0 地址开始，大小为 64KB。处理器发往该空间的访问会被转换成 LPC 协议的 I/O 访问发到 LPC 总线，地址为地址空间低 16 位。

对于 LPC 接口模块，使用时要注意将对应的芯片复用引脚设置为相应的接口功能。

与 LPC 接口相关的引脚复用设置可查询节 2.29 中的外设功能引脚复用关系表，并根据节 5.5.40~节 5.5.59 中 GPIO0~159 复用配置寄存器，配置相应的 GPIO 引脚复用关系，实现对应设备功能引脚。

17.2 LPC 中断

LPC 控制器内部包括两类中断：SIRQ 中断和访问超时中断。LPC 控制器共支持 17 个 SIRQ 中断，对应中断相关寄存器的比特位[16:0]。访问超时中断对应中断相关寄存器的比特位[17]。

SIRQ 中断为电平触发中断，触发电平的值可由寄存器配置。软件应先配置好 SIRQ 中断的触发电平，然后再使能 LPC 控制器的 SIRQ 中断。SIRQ 中断不需要软件清除。

访问超时中断为边沿触发中断，因此，如果发生 LPC 访问超时中断，则软件需要写中断清除寄存器的 bit[17]来清除该中断。

17.3 LPC 控制寄存器

LPC 控制器配置寄存器共有 4 个 32 位寄存器，基址：0x1f0e_0000，寄存器含义见下表：

表17-2 LPC寄存器0

位域	名称	访问	初值	描述
31	SIRQ_EN	R/W	0	SIRQ 中断使能控制，高有效
30:22	Reserved	RO	-	-
23	LPC_MEM_TRANS_EN	R/W	0	LPC Memory 空间地址转换使能
22:16	LPC_MEM_TRANS	R/W	0	LPC Memory 空间地址转换控制
15:0	LPC_SYNC_TIMEOUT	R/W	0	LPC 访问超时计数 当小于 64 时硬件将当作 63 处理

表17-3 LPC寄存器1

位域	名称	访问	初值	描述
31	FIRMWARE_TYPE	R/W	0	LPC Memory 空间 Firmware Memory 访问类型设置 0: Memory； 1: Firmware Memory
30:18	Reserved	RO	-	-
17:0	LPC_INT_EN	R/W	0	中断使能，为 1 的位使能对应的中断源，高有效 17: timeout 16~0: sirq

表17-4 LPC寄存器2

位域	名称	访问	初值	描述
17:0	LPC_INT_SRC	R/W	0	LPC 中断源指示，每个比特位对应一个中断源。对于每个中断源， 0：没有中断； 1：有中断。

表17-5 LPC寄存器3

位域	名称	访问	初值	描述
31:18	Reserved	RO	-	-
17	LPC_TIMEOUT_INT_CLEAR	W	-	LPC 访问超时中断清除（写 1 清除）。比特 17 对应 LPC 访问超时中断，写 1 清除，写 0 无效。
16:0	Reserved	RO	-	-

18 PS/2 控制器

18.1 PS/2 控制器结构

PS/2 控制器的结构如下图所示。

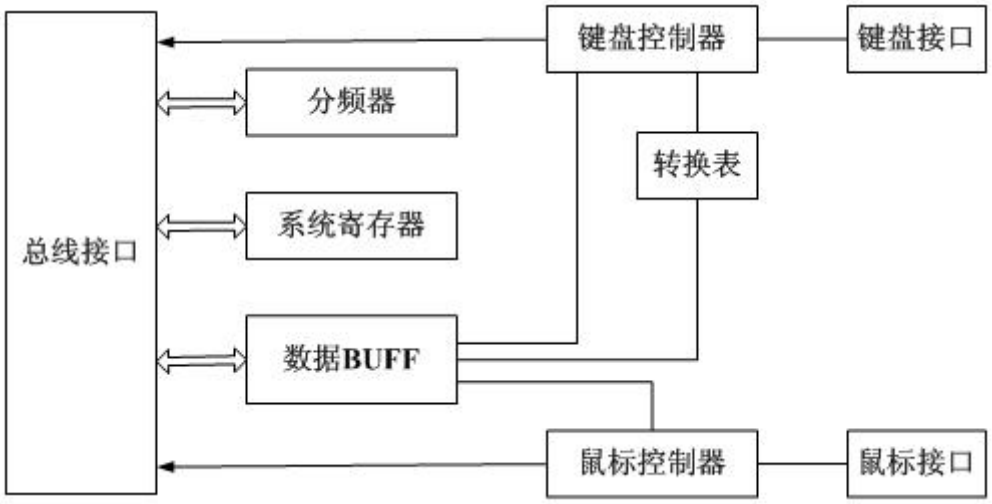


图18-1 PS/2 控制器结构

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。

18.2 配置寄存器

2K0500 的 PS/2 控制器中共包含如下 4 个寄存器，寄存器基地址是 0x1ff4c000:

表18-1 PS/2寄存器列表

寄存器名称	地址偏移	初 始值	寄存器宽度	访问类型	寄存器功能描述
接收寄存器	0x0	0x0	8 bits	RO	用于存放从键盘或鼠标接收到的数据
发送寄存器	0x0	0x0	8 bits	WO	用于存放要发送给键盘、鼠标的数据
状态寄存器	0x4	0x0	8 bits	RO	用于存放 8 位状态信息
命令寄存器	0x4	0x0	8 bits	R/W	用于存放 6 位控制信息
5us 计数低位寄存器	0x8	0x0	8 bits	R/W	用于配置 5us 时间计数值低 8 位，该计数周期为 PS/2 控制器时钟周期，需根据实际时钟周期配置该计数值。
5us 计数高位寄存器	0x9	0x0	8 bits	R/W	用于配置 5us 时间计数值高 8 位，该计数周期为 PS/2 控制器时钟周期，需根据实际时钟周期配置该计数值。
60us 计数键盘寄存器	0xa	0xc	8 bits	R/W	用于 60us 键盘计数值配置，需与 5us 计数寄存器配合，当 5us 计数值固定为 5us 时，该寄存器值为 12。

60us 计数鼠标寄存器	0xb	0xc	8 bits	R/W	用于 60us 鼠标计数值配置，需与 5us 计数寄存器配合，当 5us 计数值固定为 5us 时，该寄存器值为 12。

对于 PS/2 接口模块，使用时要注意将对应的芯片复用引脚设置为相应的接口功能。

与 PS/2 接口相关的引脚复用设置可查询节 2.29 中的外设功能引脚复用关系表，并根据节 5.5.40~节 5.5.59 中 GPIO0~159 复用配置寄存器，配置相应的 GPIO 引脚复用关系，实现对应设备功能引脚。

18.2.1 状态寄存器(SR)

表18-2 状态寄存器(SR)

位域	名称	访问	初值	描述
7	PERR	R/-	0	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”）。
6	TO	R/-	0	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。
5	MOBF	R/-	0	鼠标接收缓冲区满。 0：接收寄存器当前为空； 1：接收寄存器当前为满,存在自鼠标接收到的数据可供读取。
4	INH	R/-	0	设备通信禁止标志。 0：禁止与设备通信； 1：使能与设备通信。
3	A2	R/-	0	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发送寄存器； 1：上次对控制器的写操作写入的是命令寄存器。
2	SYS	R/-	0	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。
1	IBF	R/-	0	发送寄存器满标志。 0： 发送寄存器空，此时可以向控制器的发送寄存器写入数据； 1： 发送寄存器满，此时不允许向控制器的发送寄存器写入数据。

位域	名称	访问	初值	描述
0	OBF	R/-	0	接收寄存器满标志。 0： 接收寄存器当前为空； 1： 接收寄存器当前为满，存在数据数据可供读取。

18.2.2 命令寄存器(CR)

表18-3 命令寄存器(CR)

位域	名称	访问	初值	描述
7	Reserved	-	-	-
6	XLAT	R/W	0	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。
5	EN2	R/W	0	是否使能控制器与鼠标接口通信(注意与状态寄存器的 INH 位相对比)。 0：使能与鼠标接口通信； 1：禁止与鼠标接口通信。
4	EN	R/W	0	是否使能控制器与键盘接口通信(注意与状态寄存器的 INH 位相对比). 0：使能与键盘接口通信； 1：禁止与键盘接口通信。
3	Reserved	-	-	-
2	SYS	R/W	0	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。
1	INT2	R/W	0	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。
0	INT	R/W	0	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。

18.2.3 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。

下面给出软件操作实现流程描述

1. 发送命令给键盘。

通过直接将一个 Byte 写到发送寄存器，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)，接着再向发送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，再接着向发送寄存器中写入参数，如果该命令对应有返回值的话，返回值会放在接收寄存器中。

18.2.4 控制器命令列表

1. 发送到键盘的命令列表

a) 0xFF (Reset):引起键盘进入 Reset 模式。

b) 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。

c) 0xF5 (Disable)：键盘停止扫描,载入缺省值等待进一步指令。

d) 0xF4 (Enable)：使能键盘。

e) 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。

f) 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。

g) 0xEE (Echo)：回声命令，键盘用 0xEE 回应。

h) 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））

2. 发送到鼠标的命令列表

a) 0xFF(Reset)：复位鼠标命令。

b) 0xF6(Set Defaults)：设置鼠标的默认工作模式。

c) 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。

d) 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。

e) 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。

f) 0xF2(Get Device ID)：读取鼠标的设备 ID。

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

h) 0xEE(Set Wrap Mode)：设置鼠标进入 Wrap 模式。

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

j) 0xEB(Read Data)：读取鼠标采样到的位移数据包。

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

3. 发送到控制器的命令列表

a) 0x20：读命令寄存器。

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

c) 0xA7：禁止鼠标接口。

d) 0xA8：使能鼠标接口。

e) 0xA9：鼠标接口测试若通过则返回 0x00。

f) 0xAA：控制器自检若成功则返回 0x55。

g) 0xAB：键盘接口测试若通过则返回 0x00。

h) 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。

i) 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。

j) 0xD2：写键盘缓冲区命令。把紧随该命令的参数写到发送缓冲区就像是从键盘接收到的一样。

k) 0xD3：写鼠标缓冲区命令。把紧随该命令的参数写到发送缓冲区就像是从鼠标接收到的一样。

l) 0xD4：写鼠标设备命令。把紧随该命令的参数发给鼠标。

19 SPI 控制器

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。

19.1 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。2K0500 共集成 6 个 SPI 控制器，其中 SPI0~1 支持 IO/memory 两种空间访问，SPI2~5 仅支持 IO 空间访问。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 0x1c000000，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，该启动功能仅 SPI0 控制器支持。2K0500 芯片 6 个 SPI 控制器空间分布，如下表。

表19-1 SPI控制器地址空间分布

地址空间	名称	大小
0x1c00,0000 - 0x1c0f,ffff	SPI0 Boot	1MB
0x1800,0000 - 0x19ff,ffff	SPI0 MEM	32MB
0x1e00,0000 - 0x1eff,ffff	SPI1 MEM	16MB
0x1fd0,0000 - 0x1fd3,ffff	SPI0 IO/CFG	256KB
0x1fd4,0000 - 0x1fd7,ffff	SPI1 IO/CFG	256KB
0x1ff5,0000 - 0x1ff5,0fff	SPI2 CFG	4KB
0x1ff5,1000 - 0x1ff5,1fff	SPI3 CFG	4KB
0x1ff5,2000 - 0x1ff5,2fff	SPI4 CFG	4KB
0x1ff5,3000 - 0x1ff5,3fff	SPI5 CFG	4KB

对于 SPI 接口模块，使用时要注意将对应的芯片复用引脚设置为相应的接口功能。

与 PSPI 接口相关的引脚复用设置可查询节 2.29 中的外设功能引脚复用关系表，并根据节 5.5.40~节 5.5.59 中 GPIO0~159 复用配置寄存器，配置相应的 GPIO 引脚复用关系，实现对应设备功能引脚。

其结构如图19-1 所示，由 AXI/APB 内部总线接口、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，来自内部总线接口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)。

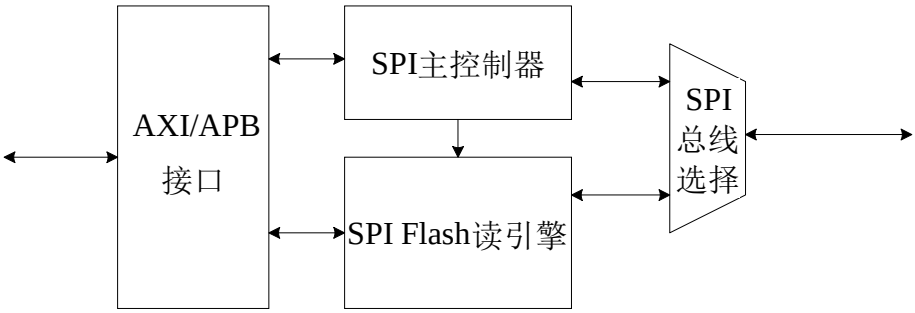


图19-1 SPI 控制器结构

19.2 配置寄存器

表19-2 SPI 配置寄存器列表

偏移	名称	描述	备注
0	SPCR	控制寄存器	SPI0~5 控制器均支持
1	SPSR	状态寄存器	SPI0~5 控制器均支持
2	TxFIFO/RxFIFO	数据寄存器	SPI0~5 控制器均支持
3	SPER	外部寄存器	SPI0~5 控制器均支持
4	SFC_PARAM	参数控制寄存器	仅 SPI0~1 控制器支持
	SPCS	SPI 软件片选配置	仅 SPI2~5 控制器支持
5	SFC_SOFTCS	片选控制寄存器	仅 SPI0~1 控制器支持
6	SFC_TIMING	时序控制寄存器	仅 SPI0~1 控制器支持

19.2.1 控制寄存器(SPCR)

表19-3 SPI 控制寄存器(SPCR)

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

19.2.2 状态寄存器(SPSR)

表19-4 SPI 状态寄存器(SPSR)

位域	名称	访问	初值	描述
7	spif	R/W	0	中断标志位 1 表示有中断申请，写 1 则清零
6	wcol	R/W	0	写寄存器溢出标志位 为 1 表示已经溢出,写 1 则清零
5	-	-	0	保留
4	busy	R	0	总线忙状态 0：总线空闲；1：总线忙状态

位域	名称	访问	初值	描述
3	wffull	R	0	写寄存器满标志 1 表示已经满
2	wfempty	R	1	写寄存器空标志 1 表示空
1	rffull	R	0	读寄存器满标志 1 表示已经满
0	rfempty	R	1	读寄存器空标志 1 表示空

19.2.3 数据寄存器(TxFIFO/RxFIFO)

表19-5 SPI 数据寄存器(TxFIFO/RXFIFO)

位域	名称	访问	初值	描述
7:0	TxFIFO RxFIFO	W R	-	数据发送端口 数据接收端口

19.2.4 外部寄存器(SPER)

表19-6 SPI 外部寄存器(SPER)

位域	名称	访问	初值	描述
7:4	icnt	R/W	0	传输完多少个字节后发中断 0000：1； 0001：2； 0010：3； 0011：4； 0100：5； 0101：6； 0110：7； 0111：8； 1000：9； 1001：10；1010：11；1011：12； 1100：13；1101：14；1110：15；1111：16。
3	-	-	-	保留
2	mode	R/W	0	spi 接口模式控制 0：采样与发送时机同时 1：采样与发送时机错开半周期
1:0	spre	R/W	0	与 spr 一起设定分频的比率

表19-7 SPI 分频系数

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

19.2.5 软件片选寄存器(SPCS)

表19-8 SPI 软件片选寄存器(SPCS)

位域	名称	访问	初值	描述
7:2	-	-	-	-
1	spi_csn_en	R/W	1	SPI 片选引脚输出使能，低电平有效
0	spi_csn	R/W	1	SPI 片选软件配置位，低电平有效

19.2.6 参数控制寄存器(SFC_PARAM)

表19-9 SPI 参数控制寄存器(SFC_PARAM)

位域	名称	访问	初值	描述
7:4	clk_div	R/W	2	时钟分频数选择 分频系数与{spre，spr}组合相同
3	dual_io	R/W	0	双 I/O 模式，优先级高于快速读
2	fast_read	R/W	0	快速读模式
1	burst_en	R/W	0	SPI flash 支持连续地址读模式

位域	名称	访问	初值	描述
0	memory_en	R/W	1	SPI flash 读使能，无效时 csn[0]可由软件控制。

19.2.7 片选控制寄存器(SFC_SOFTCS)

表19-10 SPI 片选控制寄存器(SFC_SOFTCS)

位域	名称	访问	初值	描述
7:4	csn	R/W	0	csn 引脚输出值
3:0	csen	R/W	0	为 1 时对应位的 csn 线由 7:4 位控制

19.2.8 时序控制寄存器(SFC_TIMING)

表19-11 SPI 时序控制寄存器(SFC_TIMING)

位域	名称	访问	初值	描述
7:4	-	-	-	保留
3	4byte_addr_en	R/W	0	4 字节地址访问使能位(32MB 空间大小扩展访问配置) 1：使能 4 字节地址访问，此时 SPI 可支持最大 32MB 地址空间寻址； 0：关闭 4 字节地址访问，此时 SPI 可支持最大 16MB 地址空间。
2	tFAST	R/W	0	SPI flash 读采样模式 0: 上沿采样，间隔半个 SPI 周期 1: 上沿采样，间隔一个 SPI 周期
1:0	tCSH	R/W	3	SPI Flash 的片选信号最短无效时间，以分频后时钟周期 T 计算 00: 1T 01: 2T 10: 4T 11: 8T

19.3 接口时序

19.3.1 SPI 主控制器接口时序

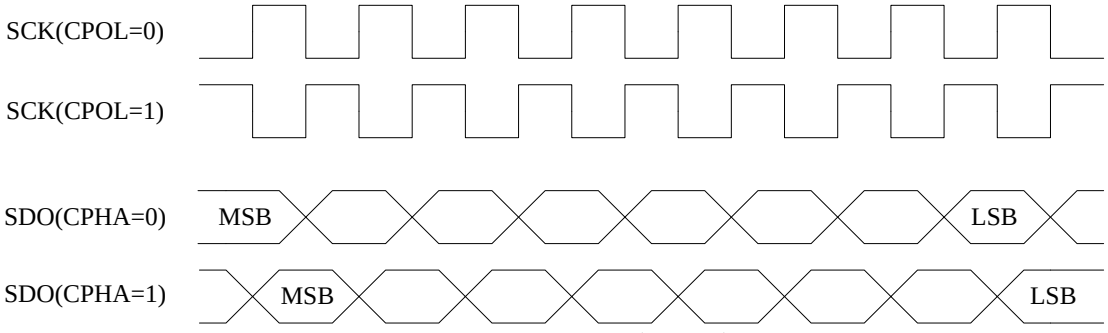


图19-2 SPI主控制器接口时序

19.3.2 SPI Flash 访问时序

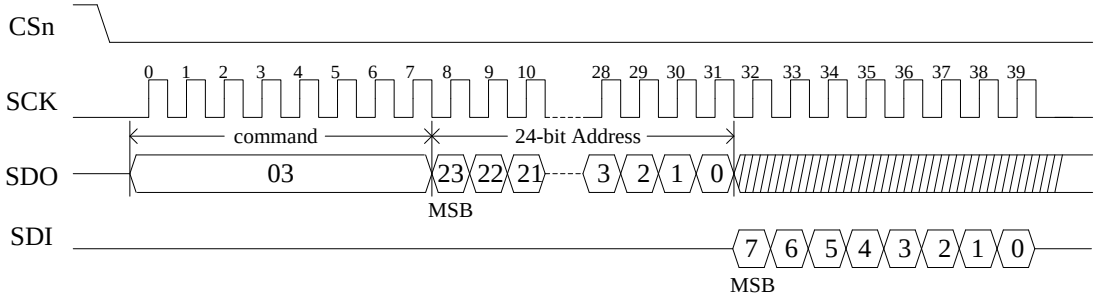


图19-3 SPI Flash 标准读时序

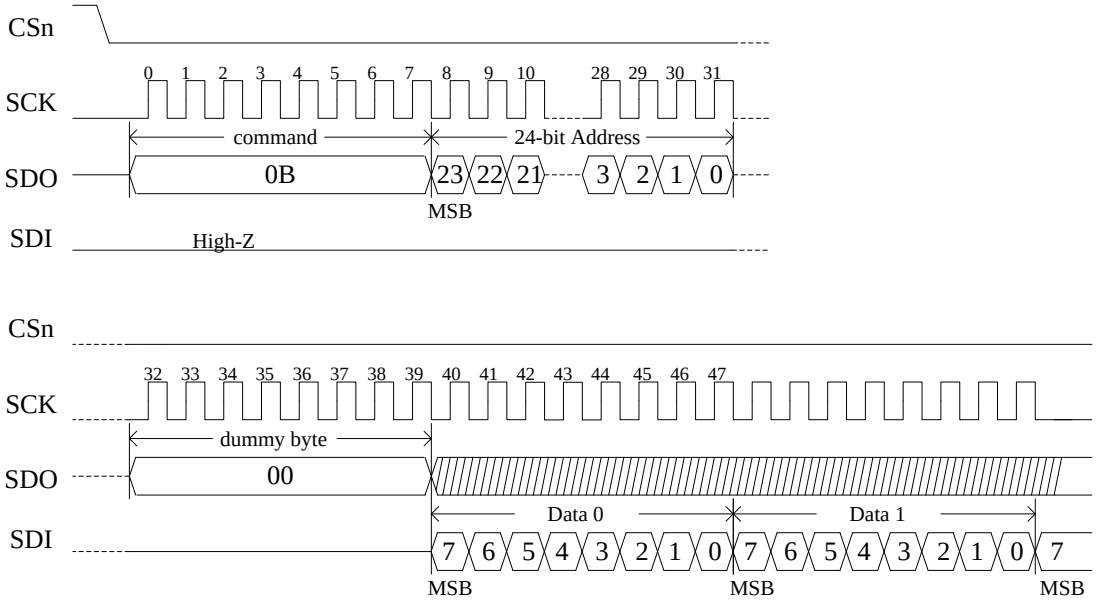


图19-4 SPI Flash 快速读时序

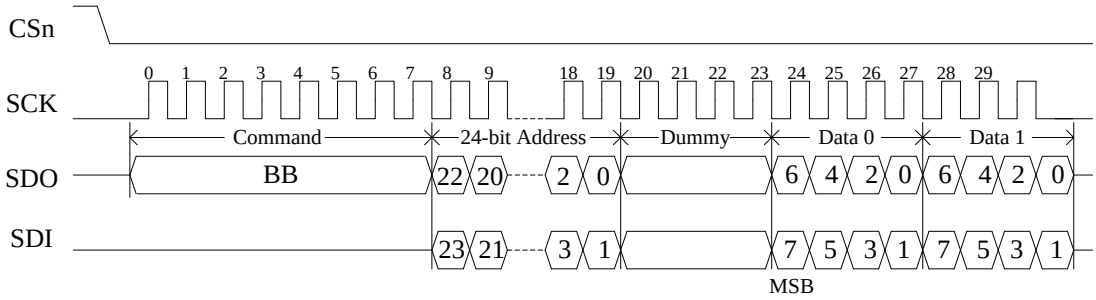


图19-5 SPI Flash 双向 I/O 读时序

19.4 使用指南

19.4.1 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

19.4.2 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。

19.4.3 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。

20 I2C 控制器

20.1 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，支持主设备和从设备模式，主要用于实现两个器件之间数据的交换，芯片默认配置为从设备模式。主设备模式时，控制器可通过轮询或中断方式工作；从设备模式时，控制器可通过中断方式工作。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。

控制器支持特性：(1) Standard_Mode，East_Mode；(2)Clock stretch as slave；(3)Clock Synchronization as master。不支持特性：(1) Hs_mode；(2)10-bit addressing；(3) 同时作为主/从设备。

20.2 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。

1) 时钟发生器模块：产生分频时钟，同步位命令的工作。

2) 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。

3) 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。

4) 数据移位寄存器模块：串行数据移位。

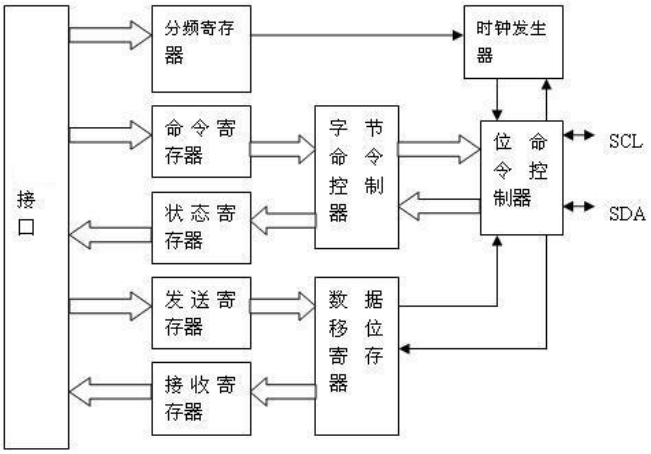


图 20-1 I2C主控制器结构

20.3 I2C 控制器寄存器说明

本芯片集成了六个 I2C 控制器 I2C-0~5，其中 I2C4~5 对应显示 I2C 接口 PIX0~1_I2C。

I2C-0：模块寄存器物理地址基址为 0x1ff48000，地址空间 2KB。

I2C-1：模块寄存器物理地址基址为 0x1ff48800，地址空间 2KB。

I2C-2：模块寄存器物理地址基址为 0x1ff49000，地址空间 2KB。

I2C-3：模块寄存器物理地址基址为 0x1ff49800，地址空间 2KB。

I2C-4：模块寄存器物理地址基址为 0x1ff4a000，地址空间 2KB，对应显示 PIX0_I2C。

I2C-5：模块寄存器物理地址基址为 0x1ff4a800，地址空间 2KB，对应显示 PIX1_I2C。

I2C 接口模块配置寄存器，如下表所示。

表 20-1 I2C 配置寄存器列表

地址偏移	寄存器名称	描述
0x00	PRERlo	分频系数低字节寄存器
0x01	PRERhi	分频系数高字节寄存器
0x02	CTR	控制寄存器
0x03	TXD/RXD	发送/接收数据寄存器
0x04	CR/SR	命令/状态寄存器
0x05	BLTOP	总线死锁时间寄存器

对于 I2C 接口模块，使用时要注意将对应的芯片复用引脚设置为相应的接口功能。

与 I2C 接口相关的引脚复用设置可查询节 2.29 中的外设功能引脚复用关系表，并根据节 5.5.40~节 5.5.59 中 GPIO0~159 复用配置寄存器，配置相应的 GPIO 引脚复用关系，实现对应设备功能引脚。

20.3.1 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

20.3.2 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

假设分频锁存器的值为 prescale，从 LPB 总线 PCLK 时钟输入的频率为 clock_a，SCL 总线的输出频率为 clock_s，则应满足如下关系：

clock_s = clock_a/4*(Prcescale+1)

20.3.3 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 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	ms	1	RW	主从模式选择位(默认从设备)： 0：从设备模式；1：主设备模式
4	txrok	1	RW	从设备发送数据准备好： 从设备模式时，当要发送的数据已写入 DR 时，此位置为 1，该位自动清零
3	rxrok	1	RW	从设备接收数据以读出： 从设备模式时，当 DR 收到的数据已经被读出时，此位置为 1，该位自动清零
2	Reserved	-	-	保留
1	Buslock_check_en	1	RW	总线死锁状态检查使能： 使能后，依据 buslock_top 寄存器规定的时间检查总线是否死锁，死锁状态持续周期达到{buslock_top,16’b0}后认定为产生死锁
0	Slv_autoreset_en	1	RW	总线死锁时从设备自动复位状态机使能： 使能时总线产生死锁后从设备会复位自身状态机，从而解除死锁，需要 buslock_check_en 使能

20.3.4 发送/接收数据寄存器（TXD/RXD）

中文名： 发送/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	TXD/RXD	8	R/W	发送/接收数据寄存器 写入时为总线待发送的数据；读出时为总线收到的数据

20.3.5 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	开始信号： 该位置 1 时，作为主设备下一次传输产生开始波形
6	STO	1	W	结束信号： 该位置 1 时，作为主设备下一次传输产生结束波形
5	RD	1	W	读信号： 该位置 1 时，作为主设备下一次传输为总线读请求
4	WR	1	W	写信号： 该位置 1 时，作为主设备下一次传输为总线写请求
3	ACK	1	W	主设备应答： 该位写 1 表示下一次读数据返回时应答 NACK，此时连续读请求结束
2	Recover	1	W	总线死锁恢复命令： 该位置 1 时作为主设备，看到总线死锁状态后，执行此命令接出死锁
1	Reserved	-	-	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’。bit 3 为 1 时表示此次传输结束时控制器不发送 ack，反之结束时发送 ack。

20.3.6 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	收到的应答位： 0：收到应答；1：收到 NACK
6	Busy	1	R	总线忙状态： 0：总线空闲；1：总线忙
5	AL	1	R	主设备失去仲裁： 该位为 1 表示主设备时序总线控制权
4	Slave_addressed	1	R	从设备被寻址： 该位为 1 表示从设备已被寻址成功
3	Slave_rw	1	R	从设备读写： 0：从设备被读；1：从设备被写

2	BUSLOCK	1	R	总线死锁： 该位为 1 表示总线出现死锁
1	TIP	1	R	传输进行状态： 作为主设备时，该位为 1 表示传输正在进行中
0	IF	1	R	中断标志位： 当传输完一个字节或主设备丢失仲裁时，中断标志为 1

20.3.7 总线死锁时间寄存器（BLTOP）

中文名：　　总线死锁时间寄存器

寄存器位宽：　[7：0]

偏移量：　　0x05

复位值：　　0xff

位域	位域名称	位宽	访问	描述
7:0	Buslock_top	8	RW	总线死锁时间： 死锁状态持续周期达到{buslock_top,16’b0}，认为产生死锁

20.3.8 从设备地址寄存器（SADDR）

中文名：　　从设备地址寄存器

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	Reserved	-	-	保留
6:0	saddr	7	RW	从设备地址： 作为从设备模式时，存放总线地址

21 UART 控制器

21.1 概述

2K0500 集成了十个 UART 控制器，通过 APB 总线与总线桥通信。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。

21.2 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）、和访问寄存器模块（Register Access Control），这些模块之间的关系见下图所示。

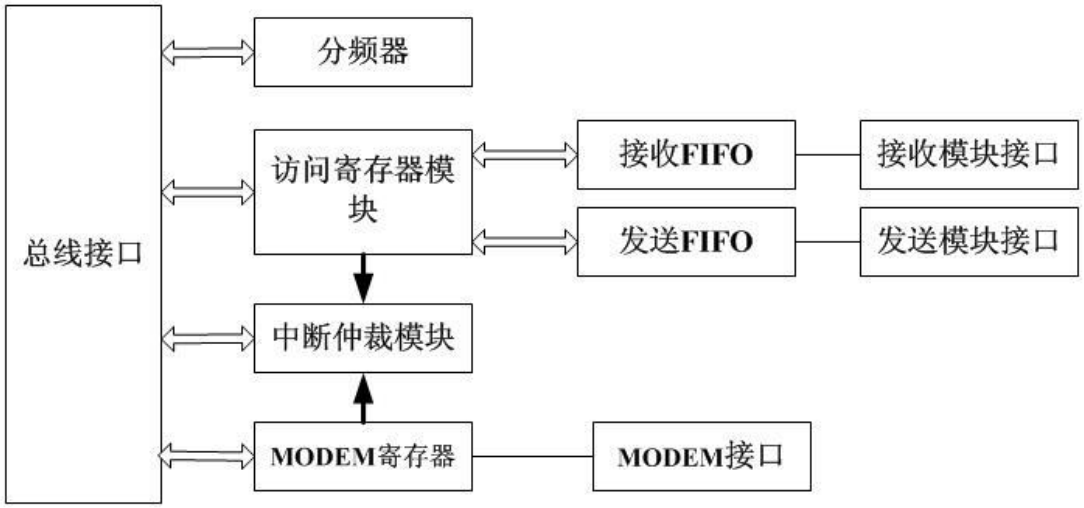


图21-1 UART控制器结构

主要模块功能及特征描述如下：

1） 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

2） 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。

3） 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。

4） 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。

21.3 寄存器描述

龙芯 2K0500 中有 10 个 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 0x1ff40000。
- UART1寄存器物理地址基址为 0x1ff40400。
- UART2寄存器物理地址基址为 0x1ff40800。
- UART3寄存器物理地址基址为 0x1ff40c00。
- UART4寄存器物理地址基址为 0x1ff41000。
- UART5寄存器物理地址基址为 0x1ff41400。
- UART6寄存器物理地址基址为 0x1ff41800。
- UART7寄存器物理地址基址为 0x1ff41c00。
- UART8寄存器物理地址基址为 0x1ff42000。
- UART9寄存器物理地址基址为 0x1ff42400。

UART 接口模块配置寄存器，如下表所示。

表 21-1 UART 配置寄存器列表

地址偏移	寄存器名称	描述
0x00	DAT/DL_L	数据寄存器/分频系数低 8 位
0x01	IER/DL_H	中断使能寄存器/分频系数高 8 位
0x02	IIR	中断标识寄存器
0x02	FCR/DL_D	FIFO 控制寄存器/分频系统小数位
0x03	LCR	线路控制寄存器
0x04	MCR	Modem 控制寄存器
0x05	LSR	线路状态寄存器
0x06	MSR	Modem 状态寄存器

对于 UART 接口模块，使用时要注意将对应的芯片复用引脚设置为相应的接口功能。

与 UART 接口相关的引脚复用设置可查询节 2.29 中的外设功能引脚复用关系表，并根据节 5.5.40~节 5.5.59 中 GPIO0~159 复用配置寄存器，配置相应的 GPIO 引脚复用关系，实现对应设备功能引脚。

21.3.1 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	数据寄存器： 读该寄存器为收到的数据；写此寄存器将准备发送数据写入发送 FIFO

21.3.2 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

21.3.3 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	中断未处理状态 该位为低表示存在未处理的中断

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

21.3.4 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	Txreset	1	W	该位写 1，清除发送 FIFO 内容并复位其逻辑
1	Rxreset	1	W	该位写 1，清除接收 FIFO 内容并复位其逻辑
0	Reserved	1	W	保留

21.3.5 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 ‘1’ – 访问操作分频锁存器 ‘0’ – 访问操作正常寄存器
6	bcb	1	RW	打断控制位 ‘1’ – 此时串口的输出被置为 0(打断状态). ‘0’ – 正常操作
5	spb	1	RW	指定奇偶校验位 ‘0’ – 不用指定奇偶校验位 ‘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 ‘0’ – 在每个字符中有奇数个 1（包括数据和奇偶校验位） ‘1’ – 在每个字符中有偶数个 1
3	pe	1	RW	奇偶校验位使能 ‘0’ – 没有奇偶校验位 ‘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 ‘0’ – 1 个停止位 ‘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 ‘00’ – 5 位　　‘01’ – 6 位 ‘10’ – 7 位　　‘11’ – 8 位

21.3.6 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	Infrared	1	W	红外载波使能： 0：关闭；1：使能
6	Rx_pol	1	W	载波使能时 RX 极性 0：正常，低电平有效；1：反转，高电平有效
5	Infrared_pol	1	W	载波使能时 TX 极性 0：正常，低电平有效；1：反转，高电平有效
4	Loop	1	W	回环模式控制位 ‘0’ － 正常操作 ‘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。 DTR　　DSR RTS　　CTS Out1　　RI 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
1	RTSC	1	W	RTS 信号控制位
0	DTRC	1	W	DTR 信号控制位

21.3.7 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 ‘1’ － 至少有奇偶校验位错误，帧错误或打断中断的一个。 ‘0’ － 没有错误
6	TE	1	R	传输为空表示位 ‘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 ‘0’ － 有数据
5	TFE	1	R	传输 FIFO 位空表示位 ‘1’ － 当前传输 FIFO 为空，给传输 FIFO 写数据时清零 ‘0’ － 有数据
4	BI	1	R	打断中断表示位 ‘1’ －接收到 起始位＋数据＋奇偶位＋停止位都是 0，即有打断中断 ‘0’ － 没有打断
3	FE	1	R	帧错误表示位

位域	位域名称	位宽	访问	描述
				‘1’ – 接收的数据没有停止位 ‘0’ – 没有错误
2	PE	1	R	奇偶校验位错误表示位 ‘1’ – 当前接收数据有奇偶错误 ‘0’ – 没有奇偶错误
1	OE	1	R	数据溢出表示位 ‘1’ – 有数据溢出 ‘0’ – 无溢出
0	DR	1	R	接收数据有效表示位 ‘0’ – 在 FIFO 中无数据 ‘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。

21.3.8 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

21.3.9 分频锁存器

中文名： 分频锁存器低 8 位

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	DL_L	8	RW	存放分频锁存器的低 8 位数值

中文名： 分频锁存器高 8 位

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	DL_H	8	RW	存放分频锁存器的高 8 位数值

中文名：　　分频锁存器小数位

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	DL_D	8	RW	存放分频锁存器的小数部分二进制值 例如，分频小数值为 0.45，DL_D=0x73(即：0x100*0.45)

UART 分频由 DL_H、DL_L、DL_D 三个寄存器组成分频值寄存器 DL，则 UART 波特率为：CLKin/16*DL。例如，输入为 10MHz 时钟，目标波特率为 115200，则 DL=5.42535，故，DL_H=0x0，DL_L=0x5，DL_D=0x6c(即：0x100*0.42535)。波特率配置值与期望值的误差在 5%以内，否则无法识别所有数据位，将导致串口显示乱码。

22 LocalIO 控制器

22.1访问地址及引脚复用

表 22-1 LocalIO 地址空间分配

起始地址	结束名称	名称	说明
0x1C00_0000	0x1C0F_FFFF	启动空间	当引脚设置为 LIO 启动时可访问，其它情况下不可访问
0x1A00_0000	0x1BFF_FFFF	存储空间	32MB 大小

LocalIO 总线地址共 23 位，支持 8 位/16 位数据位宽，采用双片选控制，可支持最大存储容量 16MB(8 位，CS0/低 8MB，CS1/高 8MB)和 32MB(16 位，CS0/低 16MB，CS1/高 16MB)。总线访问参数配置可查询节 5.5.2 中 CHIP_CTRL1[31:21]配置项，主要包括配置 LIO 总线访问延迟参数，即：LIO 总线读写控制信号 RDn/WRn 有效电平时长，该延迟参数范围为 1~32 个总线周期，具体配置有：(1)总线访问延迟计数步长周期数 clk_period[1:0](即：一次计数累加周期数，0：步长为 1；1：步长为 4；2：步长为 2；3：步长为 1)；(2)访问延迟初始值 rom_count[4:0](即：访问延迟计数以该初始值开始计数，0：初值为 31；1：初值为 30；依次类推...，31：初值为 0)；(3)LIO 总线 8/16 位数据宽度配置项等。

对于 LocalIO 接口模块，使用时要注意将对应的芯片复用引脚设置为相应的接口功能。

与 LocalIO 接口相关的引脚复用设置可查询节 2.29 中的外设功能引脚复用关系表，并根据节 5.5.40~节 5.5.59 中 GPIO0~159 复用配置寄存器，配置相应的 GPIO 引脚复用关系，实现对应设备功能引脚。

22.2LocalIO 控制器功能概述

LocalIO 控制器提供了简单外设访问接口，主要用于连接系统启动 ROM。它对外提供一个片选，具有可配置的数据位宽和访问延迟。其中 wait 参数指 liord 或 liowr 信号为低的周期数减一，读写时序可参考图 22-1，图 22-2。当数据位宽为 16 时，送出的地址由 CPU 物理地址右移一位得到。

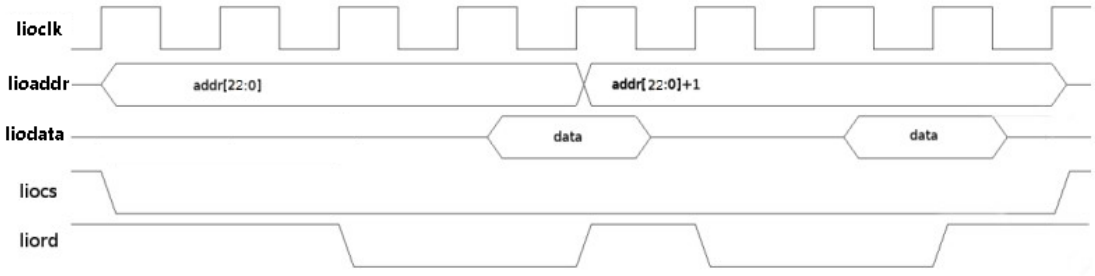


图 22-1 LocalIO 读时序

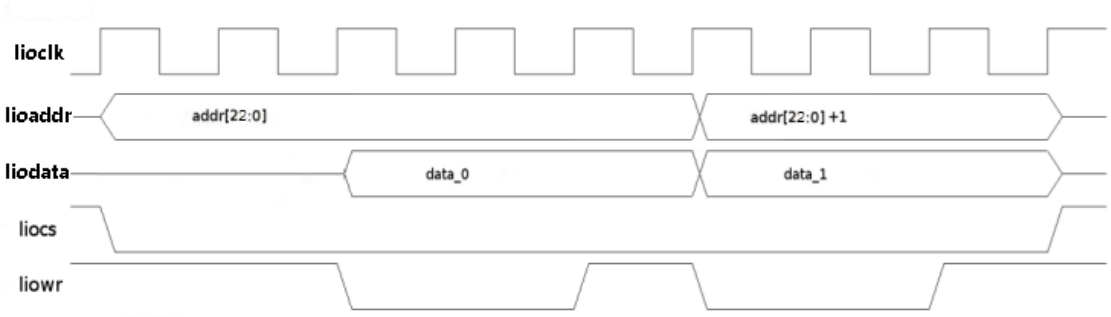


图 22-2 LocalIO 写时序

读写时序说明：

- 图中 LIOCLK 信号实际并不存在，属于内部总线时钟，只是方便时序描述；
- liord 和 liowr 低有效时间与 LIO clk_period 设置有关：

  LIO clk_period 设置为 1 时-有效电平持续最长 8 拍(计数初值从 0 开始时)；

  LIO clk_period 设置为 2 时-有效电平持续最长 16 拍(计数初值从 0 开始时)；

  LIO clk_period 设置为 3/0 时-有效电平持续最长 32 拍(计数初值从 0 开始时)；
- 单次 CS 有效期间可能出现多次读写操作，以上时序图仅作为一种示例。

23 NAND 控制器

23.1 NAND 控制器结构描述

NAND FLASH 控制器最大支持单片 16GB FLASH 的容量，最大页大小 8KB，芯片最多支持 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（非 ECC 模式）。

系统启动模式管脚配置如下表所示：

启动模式	配置	说明
ECC 模式	-	不可用于启动
普通启动	外部 NAND_CLE 上拉，PWM3、LCD_D0 下拉	外部 NAND FLASH 第一个 page 的数据为普通原始数据

23.2 NAND 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
0XBFF5_8000	NAND_CMD
0XBFF5_8004	ADDR_C
0XBFF5_8008	ADDR_R
0XBFF5_800C	NAND_TIMING
0XBFF5_8010	ID_L
0XBFF5_8014	STATUS & ID_H
0XBFF5_8018	NAND_PARAMETER
0XBFF5_801C	NAND_OP_NUM
0XBFF5_8020	CS_RDY_MAP
0XBFF5_8040	DMA access address

对于 NAND 接口模块，使用时要注意将对应的芯片复用引脚设置为相应的接口功能。

与 NAND 接口相关的引脚复用设置可查询节 2.29 中的外设功能引脚复用关系表，并根据节 5.5.40~节 5.5.59 中 GPIO0~159 复用配置寄存器，配置相应的 GPIO 引脚复用关系，实现对应设备功能引脚。

23.2.1 命令寄存器 NAND_CMD

位	位域名	读写	描述
31	DMA_REQ	R/-	非 ECC 模式下 NAND 发出 DMA 请求
30	ECC_DMA_REQ	R/-	ECC 模式下 NAND 发出 DMA 请求
29:25	STATUS	R/-	内部状态机（供测试用）
24		R/-	Reserved
23：20	NAND_CE	R/-	外部 NAND 芯片片选情况，四位分别对应片外四个片选，0 表示选中
19：16	NAND_RDY	R/-	外部 NAND 芯片 RDY 情况，对应关系和 NAND_CE 一致，1 表示准

			备好
15			Reserved
14	wait_ecc	R/W	为 1 表示等待 ECC 读完成（用于 ECC 读）
13	INT_EN	R/W	NAND 中断使能信号，为 1 表示使能中断
12	RS_WR	R/W	为 1 表示写操作时候 ECC 功能开启
11	RS_RD	R/W	为 1 表示读操作时候 ECC 功能开启
10	done	R/W	为 1 表示操作完成，需要软件清零
9	Spare	R/W	为 1 表示操作发生在 NAND 的 SPARE 区
8	Main	R/W	为 1 表示操作发生在 NAND 的 MAIN 区
7	Read status	R/W	为 1 表示读 NAND 的状态操作
6	Reset	R/W	为 1 表示 Nand 复位操作
5	read id	R/W	为 1 表示读 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	为 1 表示擦除操作
2	write operation	R/W	为 1 表示写操作
1	read operation	R/W	为 1 表示读操作
0	command valid	R/W	为 1 表示命令有效，操作完成后硬件自动清零

23.2.2 页内偏移地址寄存器 ADDR_C

位	位域名	读写	描述
31:14		R/-	Reserved
13:0	Nand_Col	R/W	读、写、擦除操作起始地址页内地址（必须以字对齐，为 4 的倍数），和页大小对应关系如下： 512Bytes：只需要填充[8:0] 2K：需要填充[11:0]，[11]表示 spare 区，[10:0]表示页内偏移地址 4K：需要填充[12:0]，[12]表示 spare 区，[11:0]表示页内偏移地址 8K：需要填充[13:0]，[13]表示 spare 区，[12:0]表示页内偏移地址

23.2.3 页地址寄存器 ADDR_R

位	位域名	读写	描述
31:25		R/-	Reserved
24:0	Nand_Row	R/W	读、写、擦除操作起始地址页地址，地址组成如下： {片选，页数} 其中片选固定为 2 位，页数根据实际的单片颗粒容量确定，如 1M 页则为[19:0]，[21:20]用于选择 4 片中的第几片

23.2.4 时序寄存器 NAND_TIMING

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

23.2.5 ID 寄存器 ID_L

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

23.2.6 ID 和状态寄存器 STATUS & ID_H

位	位域名称	访问	
31:24		R/-	Reserved
23:16	STATUS	R/-	NAND 设备当前的读写完成状态
15:0	ID[47:32]	R/-	ID 高 16 位

23.2.7 参数配置寄存器 NAND_PARAMETER

位	位域名称	访问	描述
31:20		R/-	Reserved
29:16	op_scope	R/W	每次能操作的范围，配置如下： 1.操作 main 区，配置为一页的 main 区大小 2.操作 spare 区，配置为一页的 spare 区大小 3.操作 main 加 spare 区，配置为一页的 main 区加上 spare 区大小
15		R/-	Reserved
14:12	ID_number	R/W	ID 号的字节数
11:8	外部颗粒容量大小	R/W	0：1Gb（2K 页） 1：2Gb（2K 页） 2：4Gb（2K 页） 3：8Gb（2K 页） 4：16Gb（4K 页） 5：32Gb（8K 页） 6：64Gb(8K 页) 7：128Gb(8K 页) 9：64Mb(512B 页) a:128Mb(512B 页) b:256Mb(512B 页) c:512Mb(512B 页) d:1Gb(512B 页) (bit)
7:0		R/-	Reserved

23.2.8 操作数量寄存器 NAND_OP_NUM

位	位域名	读写	描述
31:0	NAND_OP_NUM	R/W	NAND 读写操作 Byte 数(非 ECC 模式下必须以字对齐，为 4 的倍数；ECC 模式下，配置成 527*N+2，其中 512*N 为 main 区大小)；擦除为块数

23.2.9 映射寄存器 CS_RDY_MAP

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 4’b0001:NAND_RDY[0] 4’b0010:NAND_RDY[1] 4’b0100:NAND_RDY[2] 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 4’b0001:NAND_CS[0] 4’b0010:NAND_CS[1] 4’b0100:NAND_CS[2] 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 4’b0001:NAND_RDY[0] 4’b0010:NAND_RDY[1] 4’b0100:NAND_RDY[2] 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 4’b0001:NAND_CS[0] 4’b0010:NAND_CS[1] 4’b0100:NAND_CS[2] 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 4’b0001:NAND_RDY[0] 4’b0010:NAND_RDY[1] 4’b0100:NAND_RDY[2] 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 4’b0001:NAND_CS[0] 4’b0010:NAND_CS[1] 4’b0100:NAND_CS[2] 4’b1000:NAND_CS[3]
7:0		R/-	reserved

23.2.10 DMA 读写数据寄存器 DMA_ADDRESS

位	位域名	读写	描述
31：0	DMA_ADDRESS	R/W	NAND 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

23.3 NAND ADDR 说明

以 2K 页的 NAND flash 为例，定义如下:

每一页的 main 区大小为 2KB，spare 区大小为 64B

main_op = NAND_CMD[8];

spare_op = NAND_CMD[9];

addr_in_page ={ A11, A10.. A2, A1, A0}=ADDR_C

page_number ={ …A30,A29,A28,A27…A13,A12}= ADDR_R

NAND flash 的 main 区总容量计算公式为

容量=$2^{(ADDR\_C-1)} * 2^{(ADDR\_R)}$*8bit = 2K*$2^{(ADDR\_R)}$*8bit

NAND 地址空间示例如下表：

	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	….	….

(注：2K 页的 1Gb 容量 NAND flash 对应的 Row 最大值为 A27，发送地址给 NAND flash 时只用发 Column1~2 和 Row1~2，不用发 Row3。配置 NAND 参数时，注意不要配错型号，否则可能会读不出数据并且控制器会死等)

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 =0 & main_op =0) equal to (spare_op =0 & main_op =1); 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 & main_op=0; 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 & main_op =1; 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=0 & main_op=0), (equal to spare_op=0 & main_op=1); 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 and main_op =0; 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 & main_op =1; 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

512B 页大小的 NAND flash 和 2KB 页大小配置类似，但在以下几个地方会有不同，需要注意：

每一页的 main 区大小为 512B，spare 区大小为 16B。其中 main 区分为两个 256B 区，每个 256B 区通过 A0~A7 来寻址。读写操作时，通过发送命令 0x00、0x01 和 0x50 来选择是在哪个 256B 区或者 spare 区（软件不必关心，硬件自动选择，如配置 NAND 控制器写 0x100 时，硬件会自动发送到高 256B 区）。

发送地址命令顺序如下：

	I/O	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Column1	1st Cycle	A0	A1	A2	A3	A4	A5	A6	A7
Row1	2nd Cycle	A9	A10	A11	A12	A13	A14	A15	A16
Row2	3rd Cycle	A17	A18	A19	A20	A21	A22	A23	A24
Row3（注）	4th Cycle	A25	A26	*L	*L	*L	*L	*L	*L

（注：Nand flash 容量为 64Mb、128Mb 和 256Mb 时，对应的最大列地址 ADDR_R 分

别为 A22、A23 和 A24，只用发送三次地址命令 Column1 和 Row1~2，不用发送 Row3；容量为 512Mb 和 1Gb 时，需要发送 Row3）

4K/8K 页大小的配置和 2K 页配置一样，4K 页的 main 区大小为 4KB，spare 区大小为 128B；8K 页的 main 区大小为 8KB，spare 区大小为 640B。都需要发送五次地址命令。

23.4 NAND-flash 读写操作举例

命令寄存器的‘command valid’位不能与其他读写使能位同时置位，要先设置好要进行的操作，最后才置‘command valid’位。

例如：现在要读 Main 区的数据，那么操作分成以下两步：

a、先 NAND_CMD = 0x102

b、再 NAND_CMD = 0x103

23.5 NAND ECC 说明

硬件集成 ECC 功能，ECC 采用 RS(527,512)方法进行编码和解码，即每 512Byte 的有效数据对应 15Byte 的 ECC 编码。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，需要每次操作一个 Page，即使原始数据不够一个 Page；
2. 原始数据存储在 main 区，ECC 编码存储在 spare 区。ECC 码从 spare 区的第三个字节开始存储，防止缺陷标记区 defect area 被覆盖，该功能由硬件完成，软件只需按下文要求配置参数即可；
3. NAND 的 op_num 参数与 DMA 控制器的操作数的关系在不同情况下处理方式不一样，具体见下文；
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；
5. ECC 模式下，不支持 512B 页大小。

在软件控制上，写操作与读操作流程有些差别，下面给出具体操作步骤：

写操作：

a) 设置 NAND_CMD[12]为 1，表示接下来写操作为 ECC 写
b) 设置 NAND_CMD[8]和 NAND_CMD[9]为 1，表示同时操作 main 区和 spare 区
c) 设置 NAND_OP_NUM 为（main 区大小+main 区数据对应的 ECC+2）
d) 设置 ADDR_C 为 0
e) 设置其他寄存器
f) 设置 NAND_CMD[2]表示写操作
g) 设置 NAND_CMD[0]开始写操作

h) 等待 NAND_CMD[10]完成

以上每步的设置需要保证对应寄存器的其他位保持不变。对应的 DMA 控制器的操作数配置为 main 区大小/4。

读操作：

读操作比写操作复杂一些，其需要两个步骤来完成。第一个步骤先将 spare 区的 ECC 码读到 NAND 控制器的 FIFO 中暂存，具体步骤为：

1. 设置 NAND_CMD[9]和 NAND_CMD[11],表示接下来的操作在 spare 区进行，而且读回来的数据是为 ECC 解码做准备
2. 设置 NAND_OP_NUM 为 ECC 码 byte 数
3. 设置 ADDR_C 为 main 区大小
4. 设置 NAND_CMD[14]
5. 设置其他寄存器
6. 设置 NAND_CMD[1]表示读操作
7. 设置 NAND_CMD[0]开始读操作
8. 等待 NAND_CMD[10]完成

该步骤不需要 DMA 控制器，因此不必启动 DMA 控制器。

第二个步骤，读 main 区数据，NAND 控制器会同步将译码后的数据返回给 DMA 控制器，其步骤为：

1. 设置 NAND_CMD[8],同时设置 NAND_CMD[9]为无效，表示接下来操作在 main 区，同时需保证 NAND_CMD[11]为 1
2. 设置 NAND_OP_NUM 为 main 区大小
3. 设置 ADDR_C 为 0
4. 设置 NAND_CMD[1]表示读操作
5. 设置 NAND_CMD[0]开始读操作
6. 等待 NAND_CMD[10]完成

该步骤中 DMA 控制器的操作数配置为 main 区大小/4。

需注意在整个读操作过程中保持 NAND_CMD[11](rs_rd)为 1，否则 NAND 控制器的 FIFO 会被复位。

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 6 个 10bit 单元，这些 10bit 单元内部出错的位数可以是 1-10 个。10bit 单元指的是对于每个 512Byte 原始数据+15Byte ECC 码数据，从 bit0 开始把每个连续的 10bit 数据作为一个单元来看待。

24 CAN

龙芯 2K0500 集成了四路 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。

24.1 访问地址及引脚复用

四个 CAN 控制器内部寄存器的物理地址构成如下：

表 24-1 CAN 内部寄存器物理地址构成

地址	设备	备注
0x1ff4_4000	CAN0	4KB 寄存器配置空间
0x1ff4_5000	CAN1	4KB 寄存器配置空间
0x1ff4_6000	CAN2	4KB 寄存器配置空间
0x1ff4_7000	CAN3	4KB 寄存器配置空间

对于 CAN 模块，使用时要注意将对应的芯片复用引脚设置为相应的接口功能。

与 CAN 相关的引脚复用设置可查询节 2.29 中的外设功能引脚复用关系表，并根据节 5.5.40~节 5.5.59 中 GPIO0~159 复用配置寄存器，配置相应的 GPIO 引脚复用关系，实现对应设备功能引脚。

24.2 标准模式

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。

在标准模式下，CAN 控制器将 ID[10:3]的值和验收代码的值相同的消息包存入接收缓冲区。如果验收屏蔽码的某位为 1，则验收码对应的位不参与对 ID 的检查。

表 24-2 CAN 控制器标准模式下寄存器定义

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	控制	FF	—	验收屏蔽	验收屏蔽

CAN 地址	段	工作模式		复位模式	
6		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	—

24.2.1 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。

表 24-3 CAN 控制器标准模式下的控制寄存器格式

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

24.2.2 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 0xff

表 24-4 CAN 控制器标准模式下命令寄存器格式

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

24.2.3 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

表 24-5 CAN 控制器标准模式下状态寄存器格式

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

24.2.4 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

表 24-6 CAN 控制器标准模式下中断寄存器格式

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

24.2.5 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。

表 24-7 CAN 验收代码寄存器

位域	位域名称	位宽	访问	描述
7:0	AC	8	RW	ID 验收代码

24.2.6 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。

表 24-8 CAN 验收屏蔽寄存器

位域	位域名称	位宽	访问	描述
7:0	AM	8	RW	ID 屏蔽位

24.2.7 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 0xff。

表 24-9 CAN 控制器标准模式下发送缓冲区格式

地址	区	名称	数据位
10		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	发送缓冲器	TX 数据 8	TX 数据 8

24.2.8 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。

24.3 扩展模式

在扩展模式下，允许使用 11 位 ID 的标准帧和 29 位 ID 的扩展帧（是标准帧还是扩展帧由 TX 帧信息的最高位 IDE 位确定）。

在扩展模式下,CAN 控制器可以接收 11 位 ID 的标准帧也可以接收 29 位 ID 的扩展帧。在接收不同格式的帧的时候，验收代码 0～验收代码 3（code0 ~ code3）所检查的内容有所不同。验收屏蔽码 0~验收屏蔽码 3（mask0 ~ mask3）对应验收代码 0～验收代码 3。如果验收屏蔽码的某 1 位为 1，则对应的验收代码的那一位就不参与对接收包的 ID 的检查。

在扩展模式下，CAN 控制器可以对接收到的消息包进行单滤波也可以进行双滤波。在进行单滤波时，验收代码 0～验收代码 3 仅对单一 ID 进行过滤。在进行双滤波时，验收代码 0～验收代码 3 可以对两个不同的 ID 进行。CAN 控制器只将 ID 符合滤波条件的消息包存入接收缓冲区。

对 11 位 ID 包的单滤波：

允许将 rdata0 和 rdata1 用来扩展对 ID 的验收长度

(rx_id[10:3] == code0)&&(rtr == code1[4])&&(rx_id[2:0] == code1[7:5])&&(rx_data0 == code2)&&(rx_data1 == code3)

对 11 位 ID 包的双滤波：

(rx_id[10:3] == code0)&&(rtr == code1[4])&&(rx_id[2:0] == code1[7:5])&&(rx_data0 == {code1[3:0],code3[3:0]})

或者

(rx_id[10:3] == code2)&&(rtr == code3[4])&&(rx_id[2:0] == code3[7:5])

对 29 位 ID 包的单滤波：

(rx_id[28:21] == code0) && (rx_id[20:13] == code1)&&(rx_id[12:5] == code2)&&(rtr == code3[2])&&(rx_id[4:0] == code3[7:3])

对 29 位 ID 的双滤波：

(rx_id[28:21] == code0) && (rx_id[20:13] == code1)

或者

(rx_id[28:21] == code2) && (rx_id[20:13] == code3)

表 24-10 扩展模式下 CAN 控制器的地址列表

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 {IDE,RTR,2'h0,DLC[3:0]}	TX 帧信息 {IDE,RTR,2'h0,DLC[3:0]}	验收代码 0 Id[28:21]（扩展帧） Id[10:3](非扩展帧)	验收代码 0
17	RX_Id[28:21] （扩展帧） RX_Id[10:3] （非扩展帧）	TX_Id[28:21]（扩展帧） TX_Id[10:3]（非扩展帧）	验收代码 1 Id[20:13]（扩展帧） {Id[2:0],RTR,4'h0}(非扩展帧，单滤波) {Id[2:0],RTR,data0[7：4] }(非扩展帧，双滤波) {Id[2:0],RTR,4'h0] }(非扩展帧，单滤波)	验收代码 1
18	RX_Id[20:13]（扩展帧） {RX_Id[2:0],5'h0}(非扩展帧)	TX_Id[20:13]（扩展帧） {TX_Id[2:0],5'h0}(非扩展帧)	验收代码 2 Id2[28:21]（扩展帧，双滤波） Id[12:5]（扩展帧，单滤波） Id2[10:3](非扩展帧，双滤波)	验收代码 2

CAN 地	工作模式		复位模式	
			Data0（非扩展帧，单滤波）	
19	RX Id[12:5]（扩展帧） RX data0(非扩展帧)	TX Id[12:5]（扩展帧） TX data0(非扩展帧)	验收代码 3 Id2[20:13]（扩展帧，双滤波） {id[4:0],RTR,2'h0}（扩展帧，单滤波） {id2[2:0],RTR,data0[3:0]}(非扩展帧，双滤波) Data1（非扩展帧，单滤波）	验收代码 3
20	{RX_id[4:0],3'h0}（扩展帧） RX data1(非扩展帧)	{TX_id[4:0],3'h0}（扩展帧） TX data1(非扩展帧)	验收屏蔽 0（不判断为 1 的那位的 id 值）	验收屏蔽 0
21	RX data0(扩展帧) RX data2(非扩展帧)	TX data0(扩展帧) TX data2(非扩展帧)	验收屏蔽 1	验收屏蔽 1
22	RX data1(扩展帧) RX data3(非扩展帧)	TX data1(扩展帧) TX data3(非扩展帧)	验收屏蔽 2	验收屏蔽 2
23	RX data2(扩展帧) RX Data4(非扩展帧)	TX data2(扩展帧) TX Data4(非扩展帧)	验收屏蔽 3	验收屏蔽 3
24	RX data3(扩展帧) RX data5(非扩展帧)	TX data3(扩展帧) TX data5(非扩展帧)	—	—
25	RX data4(扩展帧) RX data6(非扩展帧)	TX data4(扩展帧) TX data6(非扩展帧)	—	—
26	RX data5(扩展帧) RX data7(非扩展帧)	TX data5(扩展帧) TX data7(非扩展帧)	—	—
27	RX data6(扩展帧)	TX data6(扩展帧)	—	—
28	RX data7(扩展帧)	TX data7(扩展帧)	—	—
29	RX 信息计数器	—	RX 信息计数器	—

24.3.1 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。

表 24-11 CAN 控制器扩展模式下的模式寄存器格式

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(0 为双滤波,仅复位模式可写)
2	STM	1	RW	正常工作模式(1 为自测试模式,仅复位模式可写)
1	LOM	1	RW	只听模式（仅复位模式可写）
0	RM	1	RW	复位模式

24.3.2 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 0xff

表 24-12 CAN 控制器扩展模式下命令寄存器格式

位域	位域名称	位宽	访问	描述
7	EFF	1	W	扩展模式（仅在复位模式下可写）
6：5	Reserve	2	—	保留
4	SRR	1	W	自接收请求（和 TR 不能同时为 1）
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（和 SRR 不能同时为 1）

24.3.3 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

表 24-13 CAN 控制器扩展模式下状态寄存器格式

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

24.3.4 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

表 24-14 CAN 控制器扩展模式下中断寄存器格式

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

24.3.5 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

表 24-15 CAN 控制器扩展模式下中断使能寄存器格式

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

24.3.6 仲裁丢失捕捉寄存器

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

表 24-16 CAN 控制器扩展模式下仲裁丢失捕捉寄存器格式

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

24.3.7 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

表 24-17 CAN 错误劲爆限制寄存器

位域	位域名称	位宽	访问	描述
7：0	EML	8	RW	错误警报阀值

24.3.8 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

表 24-18 CAN 的 RX 错误计数寄存器

位域	位域名称	位宽	访问	描述
7：0	RXERR	8	R	接收错误计数

24.3.9 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

表 24-19 CAN 的 TX 错误计数寄存器

位域	位域名称	位宽	访问	描述
7：0	TXERR	8	R	发送错误计数

24.3.10 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。

24.3.11 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

表 24-20 CAN 的 RX 错信息计数寄存器

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

24.4 公共寄存器

1bit time ＝ internal_clock_time * ((BRP + 1) * 2) *(1+ (TESG2 + 1) + (TESG1 + 1))

24.4.1 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。

表 24-21 CAN 总线定时寄存器 0

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

24.4.2 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

表 24-22 CAN 总线定时寄存器 1

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

24.4.3 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

表 24-23 CAN 输出控制寄存器

位域	位域名称	位宽	访问	描述
7：0	OCR	8	RW	保留（未使用）

25 SDIO 控制器

25.1 功能概述

龙芯 2K0500 集成了两个 SDIO 控制器，用于 SD Memory 和 SDIO 卡的读写，支持 SD Memory 卡启动。SDIO 控制器特性如下：

- 兼容SD 存储卡规格（2.0版本）
- 兼容SDIO卡规格（2.0版本）
- 8字（32字节）数据发送/接收FIFO
- 扩展的256位SD卡状态寄存器
- 8位预分频逻辑（频率=系统时钟/(p+1)）
- DMA数据传输模式
- 1位/4位（宽总线）的SD模式

25.2 访问地址及引脚复用

SDIO 控制器内部寄存器的物理地址构成如下：

表 25-1 SDIO 内部寄存器物理地址构成

地址	设备	备注
0x1ff6_4000	SDIO0	8KB 大小寄存器配置空间
0x1ff6_6000	SDIO1	8KB 大小寄存器配置空间

对于 SDIO 接口模块，使用时要注意将对应的芯片复用引脚设置为相应的接口功能。

与 SDIO 接口相关的引脚复用设置可查询节 2.29 中的外设功能引脚复用关系表，并根据节 5.5.40~节 5.5.59 中 GPIO0~159 复用配置寄存器，配置相应的 GPIO 引脚复用关系，实现对应设备功能引脚。

25.3 SDIO 协议概述

SDIO 是一个串行通信方式，主设备和从设备通过消息传递来实现数据和状态的传输。写多块数据过程如下：

1. 主设备通过命令线发送写命令消息给从设备
2. 从设备接收完消息之后通过命令线发送应答消息给主设备
3. 主设备接收到正确的应答消息后，通过数据线发送一块数据(512K Byte 或者更多)给从设备，并且检测数据线忙状态
4. 从设备接收到正确的数据后会进入编程状态，此时将数据线置为忙状态，不再响应主设备的数据请求
5. 主设备检测到从设备编程完成，继续发送下一块数据。

6. 主设备发送完最后一块数据时，通过命令线发送停止命令给从设备，收到正确应答之后完成这次多块写操作。

多块读操作的过程和多块写操作的过程类似。

25.4 寄存器描述

SDIO 控制器的寄存器详细说明如下:

寄存器名称	偏移地址	读/写(R/W)	功能描述	复位值
SDI_CON	0x00	R/W	SDIO 控制寄存器	0x0

SDI_CON	位	缺省值	描述
•	31:9	0x0	
soft_rst	8	0x0	软件复位，整个模块复位。复位完成后硬件自动清零
Reserved	7:1	0x0	
enclk	0	0x0	SD 时钟输出使能

寄存器名称	地址	读/写(R/W)	功能描述	复位值
SDI_PRE	0x04	R/W	SDIO 预分频寄存器	0x1

SDI_PRE	位	缺省值	描述
Reserved	31:8	0x0	
Sdi_pre	7:0	0x1	SDIO 时钟预分频值，输出频率=PCLK/预分频值

寄存器名称	偏移地址	读/写(R/W)	功能描述	复位值
SDI_CMD_ARG	0x08	R/W	SDIO 命令参数寄存器	0x0

SDI_CMD_ARG	位	缺省值	描述
sdi_cmd_arg	31:0	0x0	命令参数

寄存器名称	偏移地址	读/写(R/W)	功能描述	复位值
SDI_CMD_CON	0x0c	R/W	SDIO 命令控制寄存器	0x0

SDI_CMD_ARG	位	缺省值	描述
Reserved	31:18	0x0	
func_num_abort	17:15	0x0	SDIO 卡时中断的功能号，用于多块读写时，硬件自动发送停止命令。如果 auto_stop_en 为 0，则此位无效

sdio_en	14	0x0	SDIO 使能信号。用于多块读写时，硬件自动发送停止命令，为 1 时发送 CMD52，为 0 是发送 CMD12。如果 auto_stop_en 为 0，则此位无效
check_on	13	0x0	是否检查 CRC，为 1 时有效
Auto_stop_en	12	0x0	硬件自动发送停止命令，多块读写时，是否硬件自动发送停止命令，为 1 时有效
Reserved	11	0x0	
long_rsp	10	0x0	是否为 136 位长响应，为 1 时表示长消息回复
Wait_rsp	9	0x0	决定是否主机等待相应，为 1 是表示等待消息回复
CMST	8	0x0	命令开始，置 1 时开始，命令结束后硬件自动清零
cmd_index	7:0	0x0	带开始 2 位的命令索引（共 8 位）

寄存器名称	偏移地址	读/写(R/W)	功能描述	复位值
SDI_CMD_STA	0x10	RO	SDIO 命令状态寄存器	0x0

SDI_CMD_STA	位	缺省值	描述
Reserved	31:13	0x0	
cmd_sent_fin	14	0x0	命令发送完成（包含响应）标志位，为 1 表示命令发送完成及响应完成
auto_stop	13	0x0	硬件自动发送停止命令标志位，为 1 表示硬件自动发送停止命令，为 0 则没有
rsp_crc_err	12	0x0	响应 CRC 错误，接收到的响应 CRC 错误。为 1 时表示响应 CRC 错误，为 0 时未发现
cmd_end	11	0x0	命令发送完成（不关心响应）。为 1 时表示命令发送完成，为 0 时未完成。
cmd_tout	10	0x0	命令超时。命令响应超时（64 个时钟周期），或者 R1b 类型的命令，忙等待超时，为 1 时表示响应超时，为 0 时未超时。
rsp_fin	9	0x0	响应结束，接收完成从设备的返回信息。为 1 时表示响应结束，为 0 时未完成。
cmd_on	8	0x0	命令传输标志位。为 1 时表示传输进行中，为 0 表示结束。
rsp_index	7:0	0x0	从设备返回的带开始 2 位的响应索引（共 8 位）

寄存器名称	偏移地址	读/写(R/W)	功能描述	复位值
SDI_RSP0	0x14	RO	SDIO 命令响应寄存器 0	0x0

SDI_RESP0	位	缺省值	描述
sdi_resp0	31:0	0x0	卡状态[31:0]（短），卡状态[127:96]（长）长响应

			的配置间 sdi_cmd_con[10]

寄存器名称	偏移地址	读/写(R/W)	功能描述	复位值
SDI_RSP1	0x18	RO	SDIO 命令响应寄存器 1	0x0

SDI_RESP1	位	缺省值	描述
sdi_resp1	31:0	0x0	未使用（短），卡状态[95:64]（长）长响应的配置间 sdi_cmd_con[10]

寄存器名称	偏移地址	读/写(R/W)	功能描述	复位值
SDI_RSP2	0x1c	RO	SDIO 命令响应寄存器 2	0x0

SDI_RESP2	位	缺省值	描述
sdi_resp2	31:0	0x0	未使用（短），卡状态[63:32]（长）长响应的配置间 sdi_cmd_con[10]

寄存器名称	偏移地址	读/写(R/W)	功能描述	复位值
SDI_RSP3	0x20	RO	SDIO 命令响应寄存器 3	0x0

SDI_RESP3	位	缺省值	描述
sdi_resp3	31:0	0x0	未使用（短），卡状态[31:0]（长）长响应的配置间 sdi_cmd_con[10]

寄存器名称	偏移地址	读/写(R/W)	功能描述	复位值
SDI_DTIMER	0x24	R/W	SDIO 命令数据超时寄存器	0x0

SDI_DTIMER	位	缺省值	描述
Reserved	31:24	0x0	
sdi_dtimer	23:0	0x0	数据超时计数值，用分频后的时钟计数

寄存器名称	偏移地址	读/写(R/W)	功能描述	复位值
SDI_BSIZE	0x28	R/W	SDIO 块大小寄存器	0x0

SDI_BSIZE	位	缺省值	描述
Reserved	31:12	0x0	
sdi_bsize	11:0	0x0	块大小值（0~4095）

寄存器名称	偏移地址	读/写(R/W)	功能描述	复位值
SDI_DAT_CON	0x2c	R/W	SDIO 数据控制寄存器	0x0

SDI_DAT_CON	位	缺省值	描述
Reserved	31:21	0x0	
resume_rw	20	0x0	SDIO 挂起回复读写标志位。为 1 时，SDIO 挂起后恢复之前的写操作；为 0 时，恢复之前的读操作
IO_resume	19	0x0	SDIO 恢复请求。在 SDIO 设备进入挂起状态后，将此位写 1，并且 IO_suspend 位写 0 后，SDIO 设备恢复之前的操作。
IO_suspend	18	0x0	SDIO 挂起请求。写 1 后控制器会在合适的时机发送 CMD52 命令，通知 SDIO 设备进入挂起状态。恢复操作时需要将此位写 0.
RwaitReq	17	0x0	读等待请求。写 1 后控制器会在合适的时机将 DAT2 拉低，通知 SDIO 设备进入读等待状态。写 0 后恢复之前的读操作。
wide_mode	16	0x0	位宽选择位。为 1 表示 4 线模式，为 0 表示单线模式。
DMA_en	15	0x0	DMA 使能。为 1 时表示使能 DMA，为 0 表示禁止 DMA
DTST	14	0x0	数据传输开始，写 1 时数据传输开始，数据传输结束后硬件清零。
Reserved	13:12	0x0	
Blk_num	11:0	0x0	读写操作的块数。

寄存器名称	偏移地址	读/写(R/W)	功能描述	复位值
SDI_DAT_CNT	0x30	R/W	SDIO 数据计数寄存器	0x0

SDI_DAT_CNT	位	缺省值	描述
Reserved	31:24	0x0	
blk_num_cnt	23:12	0x0	当前传输块的字节数
blk_cnt	11:0	0x0	当前传输的块数

寄存器名称	偏移地址	读/写(R/W)	功能描述	复位值
SDI_DAT_STA	0x34	RO	SDIO 数据状态寄存器	0x0

SDI_DAT_STA	位	缺省值	描述
Reserved	31:17	0x0	
suspend_on	16	0x0	为 1 时表示正在挂起状态

rst_suspend	15	0x0	为 1 表示正在挂起复位。用于 SDIO 设备挂起后，控制器复位 FIFO 和 DMA 请求
R1b_tout	14	0x0	为 1 表示 R1b 类型命令超时
data_start	13	0x0	为 1 表示数据传输开始
R1b_fin	12	0x0	检测到带 busy 状态的命令完成。当发送带 busy 状态的命令时，此位为 0；当 busy 状态结束时变成 1
auto_stop	11	0x0	为 1 时表示硬件正在自动发送停止命令
Reserved	10	0x0	
r_wait_req	9	0x0	读等待发生。发送读等待请求信号到 SDIO 卡
SDIO_int	8	0x0	SDIO 中断标志位。为 1 表示检测到中断
crc_sta	7	0x0	数据发送后，从设备返回 CRC 错误
dat_crc	6	0x0	数据接收 CRC 错误
dat_tout	5	0x0	数据传输超时。为 1 时表示数据超时。
dat_fin	4	0x0	数据传输结束标志位（比如编程时）。为 1 时标志忙结束
busy_fin	3	0x0	编程错误标志位（比如编程时）。为 1 时标志忙结束
prog_err	2	0x0	编程错误标志位，为 1 时表示编程错误
tx_dat_on	1	0x0	Tx 数据发送中，为 1 时表示正在发送，为 0 时发送完成
rx_dat_on	0	0x0	Rx 数据接收中，为 1 时表示正在接收，为 0 时发送完成。

寄存器名称	偏移地址	读/写(R/W)	功能描述	复位值
SDI_FIFO_STA	0x38	RO	SDIO FIFO 状态寄存器	0x0

SDI_FIFO_STA	位	缺省值	描述
Reserved	31:12	0x0	
tx_full	11	0x0	Tx FIFO 满标志位
tx_empty	10	0x0	Tx FIFO 空标志位
Reserved	9	0x0	
rx_full	8	0x0	Rx FIFO 满标志
rx_empty	7	0x0	Rx FIFO 空标志位
Reserved	6:0	0x0	

寄存器名称	偏移地址	读/写(R/W)	功能描述	复位值
SDI_INT_MASK	0x3c	R/W	SDIO 中断寄存器	0x0

SDI_INT_MASK	位	缺省值	描述

Reserved	31:10	0x0	
R1b_fin_int	9	0x0	检测到 busy 结束中断，写 1 清零
rsp_crc_int	8	0x0	命令响应 CRC 错误中断，写 1 清零
cmd_tout_int	7	0x0	命令超时中断，写 1 清零
cmd_fin_int	6	0x0	发送完成中断，硬件清零
SDIO_int	5	0x0	检测到 SDIO 中断，写 1 清零
prog_err_int	4	0x0	SD 卡编程错误中断，写 1 清零
crc_sta_int	3	0x0	数据发送后从设备返回 CRC 错误中断，写 1 清零
dat_crc_int	2	0x0	数据接收 CRC 错误中断，写 1 清零
dat_tout_int	1	0x0	数据超时中断，写 1 清零
dat_fin_int	0	0x0	数据完成中断，硬件清零

寄存器名称	偏移地址	读/写(R/W)	功能描述	复位值
SDI_DAT	0x40	RO	SDIO 命令数据寄存器	0x0

SDI_DAT	位	缺省值	描述
sdi_dat	31:0	0x0	SDIO 控制器发送或者接收的数据（用于 DMA 操作）

寄存器名称	偏移地址	读/写(R/W)	功能描述	复位值
SDI_INT_EN	0x64	R/W	SDIO 中断寄使能存器	0x0

SDI_INT_EN	位	缺省值	描述
Reserved	31:10	0x0	
R1b_fin_int_en	9	0x0	Busy 结束中断使能，为 1 时有效
rsp_crc_int_en	8	0x0	命令响应 CRC 错误中断使能，为 1 时有效
cmd_tout_int_en	7	0x0	命令超时中断使能，为 1 时有效
cmd_fin_int_en	6	0x0	命令发送完成中断使能，为 1 时有效
SDIO_int_en	5	0x0	SDIO 中断使能，为 1 时有效
prog_err_int_en	4	0x0	SD 卡编程错误中断使能，为 1 时有效
crc_sta_int_en	3	0x0	数据发送后从设备返回 CRC 错误中断使能，为 1 时有效
dat_cec_int_en	2	0x0	数据接收 CRC 错误中断使能，为 1 时有效
dat_tout_int_en	1	0x0	数据超时中断使能，为 1 时有效
dat_fin_int_en	0	0x0	数据完成中断使能，为 1 时有效

25.5 软件编程指南

25.5.1 SD Memory 卡软件编程说明

SD Memory 卡要想正常工作，必须要先初始化。初始化的过程需要发送不同的命令序列来配置从设备。初始化的流程示意图如下：

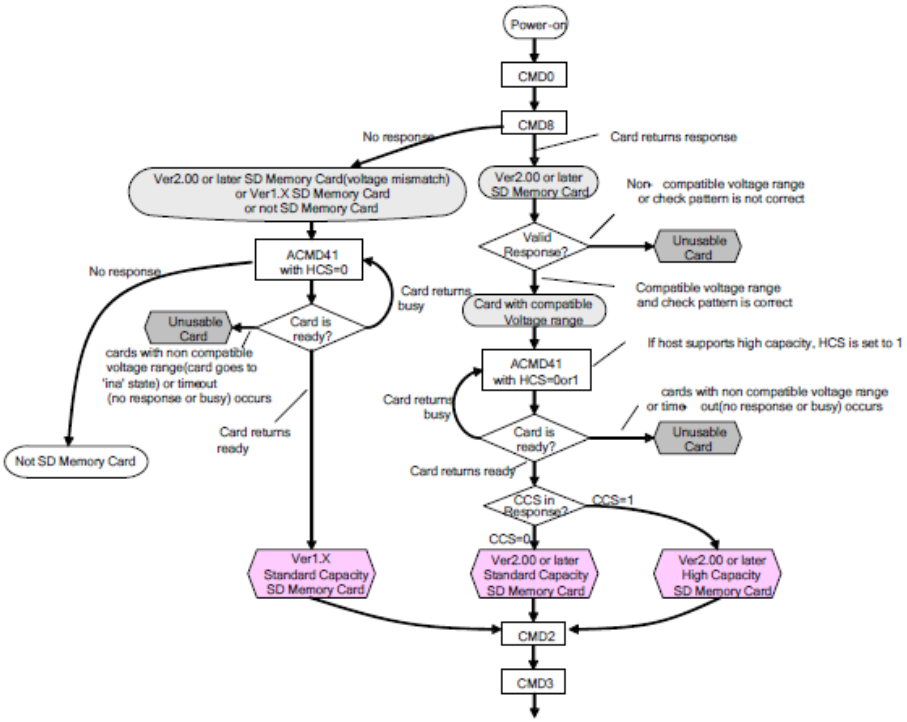


图 25-1SD Memory 卡初始化流程示意图

初始化完成之后就可以正常工作了。

配置寄存器的流程如下：

1. 配置 sdi_con，使能输出时钟

2. 配置 sdi_pre，设置一个分频系数，如果时序不满足，可以设置输出反向时钟来调整时序。

3. 配置 sdi_int_en，使能命令、数据完成及其他中断。

4. 按照上图初始化流程初始化控制器

发送命令的配置寄存器过程如下：

- 根据发送的命令，配置 cmd_arg 寄存器
- 配置 cmd_con 寄存器，发送命令
- 读 sdi_int_msk 寄存器，检查是否传输完成，是否有错误
- 如果需要，读 sdi_rsp 寄存器

初始化的流程如下：

CMD0 → CMD8 → ACMD41(即 CMD55 → CMD41) → CMD2 → CMD3

→ CMD7 → ACMD6（用于配置是否用 4bit 数据线传输）

5. 进行数据操作之前需要配置 Bsize 寄存器，Dtimer 寄存器

6. 数据的操作必须要配置 DMA，配置 dat_con 寄存器并配置 DMA（注：读操作时先配置 DMA，写操作时先配置 dat_con）

7. 读 sdi_int_msk 寄存器，检测是否传输完成，是否有错误。

8. 没有错误则完成一次数据传输，不需要软件发送停止命令

25.5.2 SDIO 卡软件编程说明

SDIO 卡的初始化流程和 SD memory 卡不同，其初始化流程如下：

1. 配置 sdi_con，使能输出时钟。

2. 配置 sdi_pre，设置一个分频系数，如果时序不满足，可以设置输出反向

3. 时钟来调整时序。

4. 配置 sdi_int_en，使能命令、数据完成及其他中断。

5. 初始化流程如下：

发送命令的配置寄存器过程如下：

- 根据发送的命令，配置 cmd_arg 寄存器
- 配置 cmd_con 寄存器，发送命令
- 读 sdi_int_msk 寄存器，检查是否传输完成，是否有错误
- 如果需要，读 sdi_rsp 寄存器

初始化的流程如下（对 CCCR 的操作）：

6. CMD52（复位） CMD5（等待上电完成） CMD3(获取 RCA) CMD7（选择相应 RCA 的卡） CMD52（配置是否用 4bit 数据线传输） CMD52（配置读写数据的块大小） CMD52（打开 IO 中断使能）进行数据操作之前需要配置 Bsize 寄存器，Dtimer 寄存器

7. 数据的操作必须要配置 DMA，配置 dat_con 寄存器并配置 DMA（注：读操作时先配置 DMA，写操作时先配置 dat_con）

8. 发送读写数据的命令时，如果需要硬件自动发送停止命令，需要配置 auto_stop 和 sdio_en。读写数据时需要先读写支持的 Function，配置相应的 FBR 的指针寄存器，再发送多块读写（CMD53）或者单块读写（CMD52）命令进行读写。

9. 读 sdi_int_msk 寄存器，检测是否传输完成，是否有错误。

10. 没有错误则完成一次数据传输，不需要软件发送停止命令

11. 如果检测到 IO 中断，控制器会置起响应的中断，但是不会停止当前的操作。

12. 对于读等待，控制器会在合适的时机将 DAT2 拉低，通知 SDIO 卡停止发送数据。所以如果当前正在传输数据过程中，控制器可能会在当前一块数据传输结束后，发出读等待信号，这时才不会接收下一块数据。

13. 对于挂起和恢复操作。有可能出现挂起嵌套情况，比如说操作 1 被操作 2 中断而挂

起，然后操作 2 被操作 3 中断而挂起。挂起时中断的现场需要软件保存（如当前的读写标志位，当前传输的块数，地址等），进行入栈操作。恢复时需要软件再按相应的顺序出栈。

26 PWM 控制器

26.1 概述

2K0500 芯片实现了 16 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和一路待测脉冲输入信号。系统时钟高达 125MHz，计数寄存器和参考寄存器均 32 位数据宽度。

26.2 访问地址及引脚复用

PWM 控制器内部寄存器的物理地址构成如下：

地址	设备	备注
0x1ff5_c000	PWM0~15	每个 PWM 占用 16B 寄存器配置空间： 0x1ff5_c000-pwm0, 0x1ff5_c010-pwm1, …

对于 PWM 接口模块，使用时要注意将对应的芯片复用引脚设置为相应的接口功能。

与 PWM 接口相关的引脚复用设置可查询节 2.29 中的外设功能引脚复用关系表，并根据节 5.5.40~节 5.5.59 中 GPIO0~159 复用配置寄存器，配置相应的 GPIO 引脚复用关系，实现对应设备功能引脚。

26.3 寄存器描述

每路控制器共有五个寄存器，具体描述如下：

表 26-1 PWM 寄存器列表

名称	地址	宽度	访问	说明
Low_buffer	Base + 0x4	32	R/W	低脉冲缓冲寄存器
Full_buffer	Base + 0x8	32	R/W	脉冲周期缓冲寄存器
CTRL	Base + 0xC	11	R/W	控制寄存器

表 26-2 PWM 控制寄存器设置

位域	名称	访问	复位值	说明
0	EN	R/W	0	计数器使能位 置 1 时：PWM 内部脉冲计数器开始计数（PWM 开始输出指定波形） 置 0 时：PWM 内部脉冲计数器停止计数（对应引脚仍保持原配置波形输出）
2：1	-	Reserved	2’b0	预留
3	OE	R/W	0	脉冲输出使能控制位,低有效 置 0 时：脉冲输出使能 置 1 时：脉冲输出屏蔽
4	SINGLE	R/W	0	单脉冲控制位 置 1 时：脉冲仅产生一次 置 0 时：脉冲持续产生

5	INTE	R/W	0	中断使能位 置 1 时：当 full_pulse 到 1 时送中断 置 0 时：不产生中断
6	INT	R/W	0	中断位 读操作：1 表示有中断产生,0 表示没有中断 写入 1：清中断
7	RST	R/W	0	使得 Low_level 和 full_pluse 计数器重置 置 1 时：计数器重置（从 buffer 读，输出低电平） 置 0 时：计数器正常工作
8	CAPTE	R/W	0	测量脉冲使能 置 1 时：测量脉冲模式 置 0 时：非测量脉冲模式（一般而言则是脉冲输出模式）
9	INVERT	R/W	0	输出翻转使能 置 1 时：使脉冲在输出去发生信号翻转（周期以高电平开始） 置 0 时：使脉冲保持原始输出（周期以低电平开始）
10	DZONE	R/W	0	防死区功能使能 置 1 时：该计数模块需要启用防死区功能 置 0 时：该模块无需防死区功能

26.4 功能说明

26.4.1 脉宽调制功能

Low_buffer 和 Full_buffer 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，模块内部的 low_level 和 full_pulse 寄存器分别从 Low_buffer 和 Full_buffer 缓冲寄存器中读取初值，之后在系统时钟驱动下不断自减（初始输出低电平）。当 low_level 寄存器到达 1 之后，输出变为高电平，此时 full_pulse 仍在自减。当 full_pulse 寄存器到达 1 之后，输出变为低电平，low_level 和 full_pulse 又分别从 Low_buffer 和 Full_buffer 缓冲寄存器中读取初值，然后重新开始不断自减，控制器就产生连续不断的脉冲宽度输出。当 full_pulse 寄存器的值等于 1 的时候，可以配置产生一个中断，从而作为定时器使用。

例：如果要产生宽度为系统时钟周期 50 倍的高脉宽和 90 倍的低脉宽，在 low_buffer 中应该配置初始值 90，在 full_buffer 寄存器中配置初始值(50+90)=140.

值得说明的是，由于两个缓冲寄存器的写入有先后之分，在某些特殊的情况下（比如写入时刻刚好是旧脉冲结束时）会使得输出脉冲有异于预期。推荐的做法是在向缓冲寄存器写入新数前，将控制寄存器 EN 位写 0，在写入新数之后再将 EN 位写 1。值得说明的是，即使没有重写 EN 位，紊乱的脉冲输出最多只会维持一个周期。

如果对两个缓冲寄存器都写 0，则输出为低电平；如果对 low_buffer 写 0，对 full_buffer 写 1，则输出高电平；如果写入 Low_buffer 的值不小于 full_buffer，则输出低电平。但这三类数值都是不推荐的。

此外，缓冲寄存器的数值写入应当先于 CTRL 控制寄存器。

26.4.2 脉冲测量功能

待测脉冲信号连在 PWM 输入信号接口上，在设置完 CTRL 控制寄存器后，在系统时钟的驱动下，Low_level 和 full_pulse 寄存器开始不断自增。当检测到输入脉冲信号上跳变时，将 Low_level 寄存器的值传送到 low_buffer 寄存器中；当检测到输入脉冲信号下跳变时，将 full_pulse 寄存器的值传送到 full_buffer 寄存器中，并将 Low_level 和 full_pulse 寄存器置 1，重新开始计数。

例：如果要输入脉冲为系统时钟 50 倍的高脉宽和 90 倍的低脉宽，在 low_buffer 中最终读出的值为 90，在 full_buffer 寄存器中读出的值为(50+90)=140.

待测脉冲应当是周期信号，且脉冲周期不应超出 32 位计数器能计量的范围。

每次测量均是从下跳变开始，到下一个下跳变结束。由于测量及缓冲的需要，在连续测量两个脉冲周期后，low_buffer 和 full_buffer 寄存器中存储的才是正确的脉冲参数。

若出现持续的周期超过 0xFFFF_FFF9 的脉冲，控制寄存器 INT 位会被置 1，表示待测脉冲超出了计量范围。

26.4.3 防死区功能

四路 PWM 都配备了防死区功能，可以防止四路脉冲输出同时发生跳变。

将四路模块分别标记为 PWM_0、PWM_1、PWM_2、PWM_3，它们的优先级为 0>1>2>3，即若要同时产生跳变，在 PWM_0 跳变之后 PWM_1 才能跳变（低优先级的信号被“抹去”一个或多个系统时钟），依此类推。该优先级是固化的，不可配置。

一个典型的防死区示例如下（PWM_*为未开防死区的输出，PWM_*’为打开防死区后的输出）：

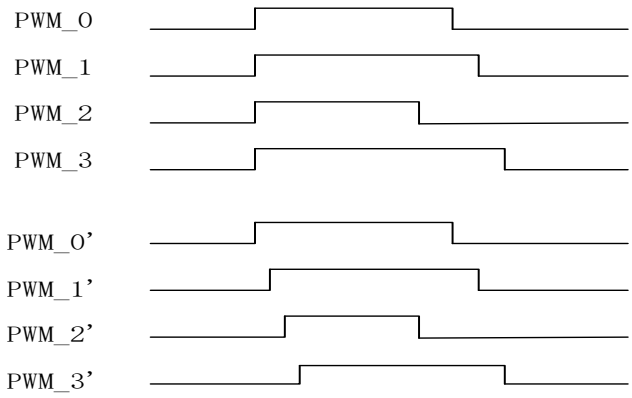


图 26-1 防死区功能

27 HPET 控制器

27.1 概述

2K0500 芯片实现了 4 个 HPET 定时控制器，HPET（High Precision Event Timer, 高精度事件定时器）定义了一组新的定时器，这组定时器被操作系统使用，用来给线程调度，内核以及多媒体定时器服务器等产生中断。操作系统可以将不同的定时器分配给不同的应用程序使用。通过配置，每个定时器都能独立产生中断。

这组定时器由一个向上累加的主计时器（up-counter）以及一组比较器构成。这个计时器以 APB 接口频率向上累加，因此当软件两次读取计时器的值时，除非遇到计时器溢出，否则第二次读取的值总是比第一次读取的值大。而每个定时器都包含一个 match 寄存器以及一个比较器。当 match 寄存器的值与主计时器相等时，那么定时器产生中断。部分定时器可产生周期性中断。

HPET 模块包括一个主计数器（main count）以及三个比较器（comparator），且他们的宽度都是 32 位。在这三个比较器中，有且仅有一个比较器支持周期性中断（periodic-capable）；这三个比较器都支持非周期性中断。

27.2 访问地址

HPET 控制器内部寄存器的物理地址构成如下：

地址	设备	备注
0x1ff6_8000	HPET0	4KB 寄存器配置空间
0x1ff6_9000	HPET1	4KB 寄存器配置空间
0x1ff6_a000	HPET2	4KB 寄存器配置空间
0x1ff6_b000	HPET3	4KB 寄存器配置空间

27.3 寄存器描述

下表列出了 HPET 的寄存器:

寄存器偏移地址	寄存器	类型
000-007h	General Capabilities and ID Register	只读
008-00Fh	Reserved	
010-017h	General Configuration Register	读/写
018-01Fh	Reserved	R/WC
020-027h	General Interrupt Status Register	R/W
028-0EFh	Reserved	
0F0-0F7h	Main Counter Value Register	R/W
100-107h	Timer 0 Configuration and Capability Register	R/W
108-10Fh	Timer 0 Comparator Value Register	R/W
110-11Fh	Reserved	

120-127h	Timer 1 Configuration and Capability Register	R/W
128-12Fh	Timer 1 Comparator Value Register	R/W
130-13Fh	Reserved	
140-147h	Timer 2 Configuration and Capability Register	R/W
148-14Fh	Timer 2 Comparator Value Register	R/W
150-15Fh	Reserved	

若系统在状态转换过程中需要保存这些寄存器的的值以便随后恢复，那么操作系统负责保存这些寄存器的值，硬件无需保存这些寄存器的值。因此当系统处于 S3，S4，S5 状态时，这些寄存器无需维持。

General Capabilities and ID Register

位	名称	描　述	读写特性
63：32	COUNTER_CLK_PERIOD	Main Counter Tick Period：这个域标示了主计时器的计时频率，以 fps（10^-15s）为单位。这个值必须大于 0，且小于或等于 05F5E100（100ns，即 10MHz）	RO
31：16	VENDOR_ID		RO
15：14	Reserved		
13	COUNT_SIZE_CAP	Counter Size:主计时器的宽度； 0：32 bits 1：64 bits	RO
12:8	NUM_TIM_CAP	Num of Timer：定时器的个数；这个域的值指示最后一个定时器的编号，GS 南桥芯片的 HPET 有三个定时器，因此这个域的值是 2。	RO
7:0	REV_ID	版本号； 不可为 0	RO

General Configuaration Register

位	名称	描述	读写特性
63：1	Reserved		
0	ENABLE_CNF	Overal Enable； 用来使能所有定时器产生中断。如果为 0，主计时器停止计时且所有的定时器都不再产生中断。 0：主计时器停止计时且所有的定时器都不再产生中断； 1：主计时器计时且允许定时器产生中断；	R/W

General Interrupt Status Register

位	名称	描述	读写特性
63：3	Reserved		
2	T2_INT_STS	Timer 2 Interrupt Active:功能同 T0_INT_STS	R/WC
1	T1_INT_STS	Timer 1 Interrupt Active:功能同 T0_INT_STS	R/WC
0	T0_INT_STS	Timer 0 Interrupt Active:功能依赖于这个定时器的中断	R/WC

		触发模式是电平触发还是边沿触发： 如果是电平触发模式： 这位默认是 0。当对应的定时器发生中断，那么有硬件将其置 1.一旦被置位，软件往这位写 1 将会清空这位。往这位写 0，则无意义。 如果边沿触发模式： 软件将忽略这位。软件通常往这位写 0.	

各个定时器的中断触发模式由各自 Configuartion and Capability 寄存器的 Tn_TYPE_CNF 位确定。

Main Counter Value Register

位	名称	描述	读写特性
63：32	Reserved		
31：0	Main_Counter_Val	主计时器的值；只有当主计时器停止计时时，才允许修改这个寄存器的值。	R/W

Timer N Configuration and Capabilities Registe

位	名称	描述	读写特性
63：9	Reserved		
8	Tn_32MODE_CNF	Timer n 32-bit 模式（N 为 0-2）。当定时器为 32 位时，这位为 0，且只读	RO
7	Reserved		RO
6	Tn_VAL_SET_CNF	Timer N Value Set（N 为 0-2）:只有能产生周期性中断的定时器才会使用这个域。通过对这位写 1，软件能直接修改周期性定时期的累加器。软件无需对这位清 0 GS 南桥芯片中只有 Timer 0 能产生周期性中断，因此对 Timer0 来讲，这位是可读可写。而对于 Timer1，Timer2，这位默认为 0，且为只读。	R/W
5	Tn_SIZE_CAP	Timer N Size；Timer N 的宽度（N 为 0-2）。 0：32 位宽。	RO
4	Tn_PER_INT_CAP	Timer N Periodic Interrupt Capable（N 为 0-2）： 1：定时器能产生周期性中断； 0：定时器不能产生周期性中断；	RO
3	Tn_TYPE_CNF	Timer N type（N 为 0-2）： 如果对应的 Tn_PER_INT_CAP 位为 0，那么这位为只读，且默认为 0. 若对应的 Tn_PER_INT_CAP 位为 1,那么这位可读可写。用作使能相应的定时器产生周期性中断。	R/W

		1：使能定时器产生周期性中断 0：使能定时器产生非周期性中断	
2	Tn_INT_ENB_CNF	Timer N interrupt Enable（N 为 0-2）:使能定时器产生中断	R/W
1	Tn_INT_TYPE_CNF	Timer N Interrupt Type（N 为 0-2）： 0：定时器的中断触发模式为边沿触发；这意位着对应的定时器将产生边沿触发中断。若另外的的中断产生，那么将产生另外的边沿。 1：定时器的中断触发模式为电平触发；这意味着对应的定时器将产生电平触发中断。这个中断将一直有效直到被软件清掉(General Interrupt Status Register)。	
0	Reserved		

Timer N Comparator Value Register

位	名称	描述	读写特性
63：32	Reserved		
31：0	Tn_Com_VAL	Tn_Comparator value（N 为 0-2）：定时器比较器的值； 当对应的定时器配置为非周期性模式时： ◆ 这个寄存器的值将与主计时器寄存器的值做比较； ◆ 若主计时器的值与比较器的值相等时，则产生定时中断（如果；对应的中断使能打开）。 ◆ 比较器的值不会因为中断的产生而发生变化 若对应的定时器配置为周期性模式时： • 当主计时器的值域比较器的值相等时，产生中断（如果对应的中断使能被打开）； • 如果产生中断，那么比较器的值将累加最后一次软件写入比较器的值。比如当比较器的值被写入 0x0123h， • 那么当主计时器的值为 0x123h 时，产生中断； • 比较器的值被硬件修改为 0x246h； • 当主计时器的值达到 0x246h 时，产生另外一个中断； • 比较器的值被硬件修改为 0x369h。 ◆ 只要产生中断，那么比较器的值都会累加；直到比较器的值达到最大（0xffffffff），那么累加器的值将会继续累加。比如当比较器的值是 FFFF0000h，而最后一次由软件写入比较器的值是 20000。当中断发生后，比较器的值变为 00010000h。	R/W

28 DMA 控制器

28.1 DMA 控制器结构描述

龙芯 2K0500 中 DMA 用来实现内存与 APB 设备之间数据搬移，可以节省资源提高系统数据传输的效率。

DMA 的传送数据的过程由三个阶段组成：

a) 传送前的预处理：由 CPU 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 控制器是一个基于 AXI 总线的单通道、可配置的 DMA 控制器 IP 核，主要功能就是在芯片上集成了 DMA 功能，专门负责在内存与 APB 设备间搬运数据。本 DMA 控制器限定为以字（4Byte）为单位的数据搬运。

CPU 通过一个通用寄存器（dma_order）向 DMA 发命令。DMA 根据命令内容，从内存读取描述符启动 DMA 直接操作，或者将 DMA 状态写入内存，或者停止 DMA。

dma_order 寄存器为芯片配置寄存器，四个 DMA 配置地址分别为 0x1fe10c00，0x1fe10c10，0x1fe10c20，0x1fe10c30，在 5.5.84 节有介绍，下表列出相同内容，方便查看。

表 28-1 DMA ORDER 寄存器

位域	名称	访问	缺省值	描述
31:5	ask_addr	R/W	0	64 位地址的高 59 位
4	dma_stop	R/W	0	停止 DMA 操作。DMA 控制器完成当前数据读写后停止。
3	dma_start	R/W	0	开始 DMA 操作。DMA 控制器读取描述符地址(ask_addr)后将些位清零。
2	ask_valid	R/W	0	DMA 工作寄存器写回到(ask_addr)所指向的内存，完成后清零。
1:0	Reserved	-	-	-

28.2 DMA 控制器与 APB 设备的交互

2K0500 中包含 4 个 DMA 控制器，DMA0~3 默认配置下分别供 APB 设备中 NAND、AC97 写操作(播音输出)、AC97 读操作(录音输入)及 SDIO0 接口使用。其中 NAND 和 SDIO 各需要一个 DMA 控制器，而 AC97 模块需要两个 DMA 控制器分别控制 DMA 写操作(播音输出)和 DMA 读操作(录音输入)。根据应用场景不同，需要通过芯片配置寄存器来设置 APB 设备使用哪个 DMA 控制器，默认配置为：NAND 采用 DMA0 控制器，AC97 写操作(播音输出)使用 DMA1 控制器，AC97 读操作(录音输入)使用 DMA2 控制器，SDIO0 使用 DMA3 控制器，SDIO1 复用其中 DMA0~2 控制器，通过 5.5.1 节通用配置寄存器 0(CHIP_CTRL0[15:14])配置实现（注：该复用配置有效时，对应 DMA 控制器只能用于 SDIO1，

不能供原接口使用）。如果是写 AC97 的操作(播音输出)，需要在配置 DMA 描述符时，将 DAM_DADDR[31]配置为 1，将 DMA_DADDR[30:28]配置为需要的模式。

28.3 DMA 描述符

28.3.1 DMA_ORDER_ADDR_LOW

中文名： 下一个描述符低位地址寄存器

寄存器位宽： [31：0]

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	存储器内部下一描述符地址寄存器（低 32 位）
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。在 64 位地址模式下，该寄存器存储低 32 位地址。

28.3.2 DMA_SADDR

中文名： 内存低位地址寄存器

寄存器位宽： [31：0]

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（低 32 位）

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 ddr3 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。在 64 位地址模式下，该寄存器存储低 32 位地址。

28.3.3 DMA_DADDR

中文名： 设备地址寄存器

寄存器位宽： [31：0]

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(播音输出)

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	28	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。使用 DMA 的 APB 设备都有对应的数据 buffer 地址，比如 NAND 控制器的数据 buffer 偏移地址为 0x40，默认的设备地址就可以配置成 0x1fe06040。

28.3.4 DMA_LENGTH

中文名： 长度寄存器

寄存器位宽： [31：0]

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。

28.3.5 DMA_STEP_LENGTH

中文名： 间隔长度寄存器

寄存器位宽： [31：0]

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。

28.3.6 DMA_STEP_TIMES

中文名： 循环次数寄存器

寄存器位宽： [31：0]

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。

28.3.7 DMA_CMD

中文名： 控制寄存器

寄存器位宽： [31：0]

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 ddr3 写设备，“0”为读设备写 ddr3
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

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

28.3.8 DMA_ORDER_ADDR_HIGH

中文名： 下一个描述符高位地址寄存器

寄存器位宽： [31：0]

偏移地址： 0x20

复位值： 0x00000000

位域	位域名称	位宽	访问	描述
31:0	dma_order_addr	32	R/W	存储器内部下一个描述符地址寄存器(高 32 位)

28.3.9 DMA_SADDR_HIGH

中文名： 内存高位地址寄存器

寄存器位宽： [31：0]

偏移地址： 0x24

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(高 32 位)

29 电源管理模块

29.1 概述

龙芯 2K0500 电源管理模块提供系统功耗管理实现机制。支持 Advanced Configuration and Power Interface, Version 4.0a(ACPI),提供相应的功耗管理功能。

- 系统休眠与唤醒，支持 ACPI S3（待机到内存），ACPI S4（待机到硬盘），ACPI S5（软关机），并且支持电源失效检测和自动系统恢复。支持多种唤醒方式(GMAC0，电源开关等）
- 支持 Dynamic Power Management（DPM），动态性能功耗控制，支持动态关闭 NODE（CORE+SCACHE）、GPU、PCIE、SATA、USB2/3.0 控制器电源。
- 支持 Dynamic Voltage Frequency Scaling（DVFS），处理器核 DVFS 控制，由片内小核 LA132 处理器核独立控制。
- 系统时钟控制，模块时钟门控，多种方式调节频率。
- 提供温度管理控制功能。支持 3 级报警机制。

29.2 动态电源管理

龙芯 2K0500 支持芯片内部各大功能模块动态时钟门控、电源开关管理功能，主要包括动态电压频率调节(DVFS)和动态电源管理(DPM)技术。具体功能划分如下表：

表 29-1 芯片各功能模块电源管理划分

功能模块	DVFS	动态时钟门控	动态电源管理	PHY 低功耗	说明
NODE (CORE+SCACHE)	支持	支持	-	-	通过片内小核 LA132 调节 NODE 时钟频率、关断和外部供电电压值
DDR	-	支持	-	-	
PCI	-	支持	-	-	
PCIE	-	支持	支持	支持	PCIE 模块电源关断，PHY 接口自动进入低功耗模式
GPU	-	支持	支持	-	
DC	-	支持	-	-	
GMAC0/1	-	支持	-	-	
USB2.0/3.0	-	支持	支持	支持	USB 模块电源关断，PHY 接口自动进入低功耗模式

功能模块	DVFS	动态时钟门控	动态电源管理	PHY 低功耗	说明
SATA	-	支持	支持	支持	SATA 模块电源关断，PHY 接口自动进入低功耗模式
HDA	-	支持	-	-	
PRINT	-	支持	-	-	

29.2.1 DVFS 功能描述

2K0500 芯片支持处理器核(CORE+SCACHE)动态电压频率调节(DVFS)控制，由片内小核 LA132 处理器核控制实现，控制结构框图如下图所示：

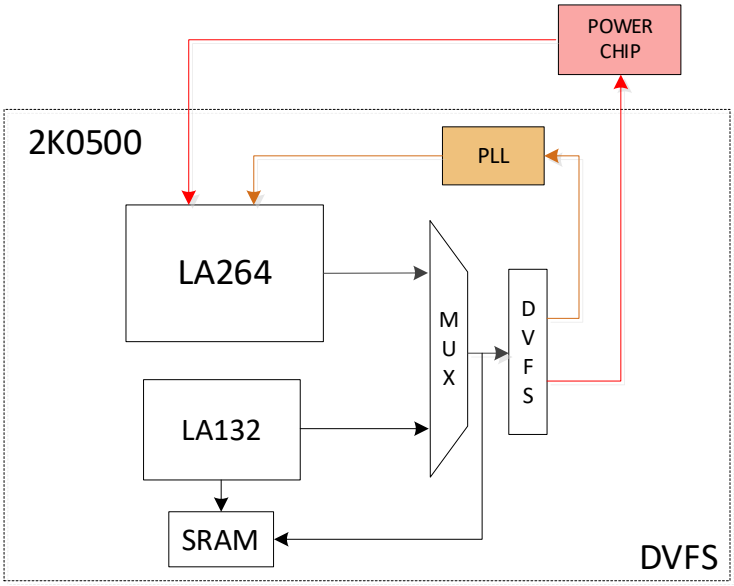


图29-1 DVFS 控制结构框图

动态调频调压由 LA132 独立执行专用程序实现，芯片内部配有 16kB 指令 SRAM 供 LA132 启动取指，启动地址为：0x1c00_0000~0x1c00_3fff；同时，除取指访问外，该内部 SRAM 可供 LA132 作为少量数据 SRAM 存储，存储地址为 0x1000_0000~0x1000_3fff。此外，2K0500 芯片 LA264 可通过 APB 总线访问该内部 SRAM(访问地址：0x1ff7_0000~0x1ff7_3fff)，并负责将 LA132 启动程序通过上述地址写入该 SRAM 中，启动程序初始化完成后，2K0500 芯片再打开 LA132 时钟使能(DVFS_CFG[16]:la132_clken 写‘0’有效)，解除 LA132 复位(DVFS_CFG[17]:la132_soft_rstn 写‘1’解除)，此时，LA132 处理器核开始启动并从 SRAM 中取指，运行特定控制程序。

2K0500 处理器核动态调频调压控制具体实现:

（1） LA264 处理器核**频率调节**,主要通过 LA132 配置 DVFS 控制寄存器(DVFS_CTRL)实现，LA132 通过访问 DVFS 控制寄存器(访问地址：0x1ff6_0024,与 LA264 访问地址相同)，首先，配置 DVFS 使能位有效，此时可通过配置 CLKBYP 位先将处理器核时钟 BYPASS 为系统参考时钟(100MHz)，再配置 CLK_ODIV 分频参数，对应处理器核时钟分频参数，进行时钟分

频配置，待选择合适分频参数后，解除 CLKBYP 配置，处理器核恢复正常时钟运行。

（2） LA264 处理器核**电压调节**,主要通过 LA132 控制芯片内部一路 I2C 总线，可选择 I2C0~5 中任一路(访问地址:0x1ff4_8000~0x1ff4_a800,与 LA264 访问地址相同),访问芯片外部 I2C 接口电源芯片(该电源芯片只负责 LA264 处理器核 VDD_NODE 电压)，通过改变该电源芯片参数配置，实现输出不同电压值，供 2K0500 芯片处理器核使用。

2K0500 芯片处理器核(CORE+SCACHE)动态电压频率调节(DVFS)具体操作流程图如下图所示：

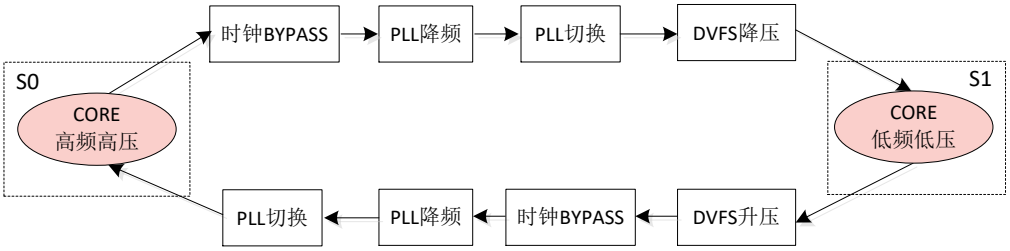


图29-2 处理器核 DVFS 操作流程图

软件操作：

首先，2K0500 芯片通过 APB 总线将 LA132 启动程序通过指定地址(0x1ff7_0000~0x1ff7_3fff)写入 LA132 内部 SRAM 中，待启动程序初始化完成后，2K0500 芯片再打开 LA132 时钟使能(DVFS_CFG[16]:la132_clken 写‘0’有效),解除 LA132 复位（DVFS_CFG[17]:la132_soft_rstn 写‘1’解除），此时，LA132 处理器核开始启动并从 SRAM 中取指，运行特定控制程序。

2K0500 处理器核进入低压低功耗工作模式，首先需将处理器核进行降频处理，具体操作为：1）使能 DVFS，配置 dvfs_en 有效；2）将处理器核时钟 bypass 为系统参考时钟，配置 dvfs_clkbyp 有效；3）配置 dvfs_clkodiv 时钟输出分频参数(最高位需为 1)；4）切换 PLL 时钟,配置 dvfs_clkbyp 无效;5)LA132 降低处理核外部供电电压至特定值,完成 2K0500 处理器核低频低压工作模式切换。

与此类似，2K0500 处理器核进入高频高性能工作模式，需先将处理器核供电进行升压，再对其工作频率进行升频处理，并进行相应时钟切换，完成 2K0500 处理器核高频高压模式切换。

29.2.2 DPM 功能描述

2K0500 芯片支持各大功能模块动态电源调节(DPM)控制，通过 DPM 模块实现各个功能模块动态时钟门控、电源开关状态管理，其控制结构框图如下图所示：

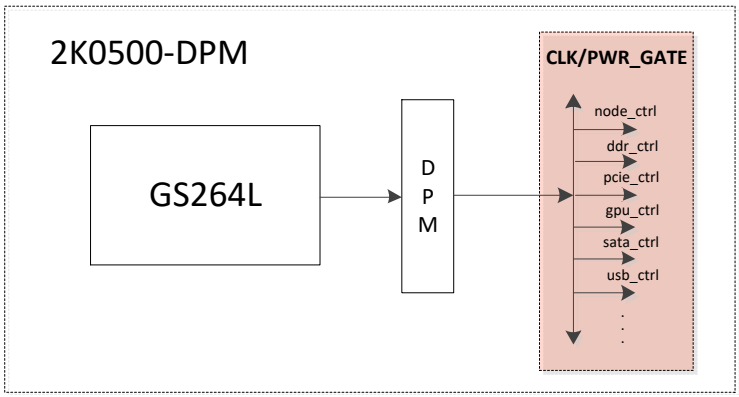


图29-3 DPM 控制结构框图

2K0500 芯片各功能模块动态电源调节(DPM)，模块时钟、电源状态动态切换流程图如下图所示：

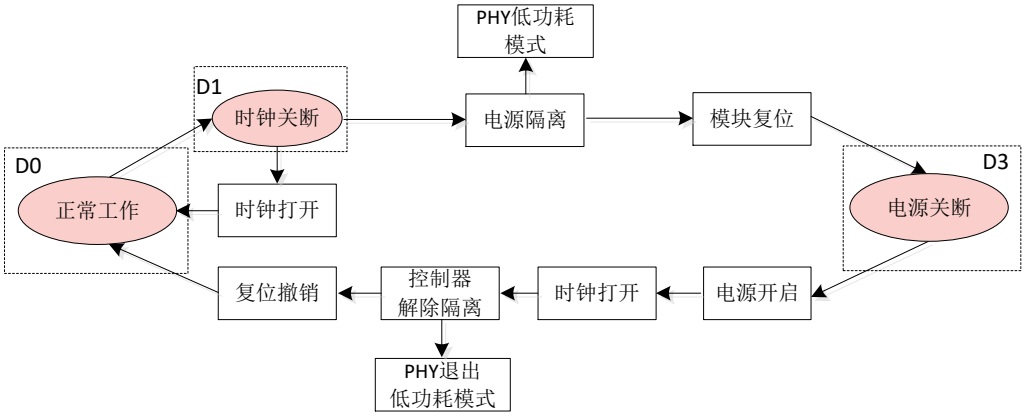


图29-4 功能模块 DPM 状态切换流程图

其中，D0 状态为模块时钟、电源全开正常工作模式；D1 状态为模块时钟关闭、电源开启静态模式；D3 状态为模块时钟、电源全部关闭(若功能模块包括 PHY 接口，PHY 处于低功耗模式)。

软件操作：2K0500 芯片特定功能模块正常工作模式(D0 状态)下，首先，使能 DPM 模式调节，配置 DPM_EN 相应使能位有效；然后，选择目标状态，若进行时钟关闭、电源关断操作直接选择 D1、D3 状态，配置 DPM_TGT 对应模块目标状态值，完成模块时钟、电源关断切换，且可由 DPM_STS 进行当前状态软件确认。

与此类似，若需解除当前关断状态，时钟关断状态 D1 下，配置 DPM_TGT 目标状态值完成状态恢复，并由 DPM_STS 进行软件确认；若解除电源关断状态，首先需配置 WAIT_TIME 上电、复位等待时间参数，并确认 PWRUP_SEL 应答模式，再配置 DPM_TGT 目标状态值完成状态恢复，并由 DPM_STS 进行软件确认。

29.3 寄存器描述

本节介绍电源管理控制器相关寄存器，使用方法可参见下一节描述。

电源管理控制(ACPI)的物理基地址为：0x1FF6_C000 — 0x1FF6_C0FF (256B)；

动态电源管理(DPM/DVFS)的物理基地址为：0x1FF6_0000 — 0x1FF6_00FF (256B)；

寄存器电压域表示寄存器的该位所属电压域。

寄存器属性简写包括：

R/W（可读可写），RO（只读），

R/WC（可读，写清除），WO（只写，读无效）

29.3.1 ACPI 寄存器描述

ACPI 配置寄存器基址为：0x1ff6_c000。

■ PMCON_SOC : SOC General PM Configuration Register

地址偏移	电压域	属性
0x00	SOC	R/W，RO

位域	描述
25	**PWRBTN_LVL - RO** 该位指示当前 PWRBTNn 信号状态。
24	**PWRTYP - RO** 该位指示供电模式 1：AC（适配器） 0：Battery（电池）
23:9	保留
8:0	**TEMP_NOW - RO**. CPU 内部温度传感器温度值，第 8 位为溢出位。

■ PMCON_RESUME : RESUME General PM Configuration Register

地址偏移	电压域	属性
0x04	RESUME	R/W，RO，R/WC

位域	描述
31:14	保留
13	**VSB_GATEn_EN - R/W** 是否使能 VSB_GATEn 功能。0：关闭；1：使能。 如果 RSMRSTn 有效过，该位为 1。重新上电后由系统配置该位。如果主板使用 VSB_GATEn 引脚作为电源管理控制信号，系统软件必须将该位写 1。
12:11	**VSB_GATEn_DLY - R/W** 用来控制 S0 到 S3 以及 S3 到 S0 时，VSB_GATEn 信号相对 S3n 的持续时间（休眠时提前的时间，唤醒时延后的时间）。 2'b00：休眠时提前 31.25ms，唤醒时延后 125ms； 2'b01：休眠时提前 62.5ms， 唤醒时延后 250ms； 2'b10：休眠时提前 125ms， 唤醒时延后 500ms； 2'b11：休眠时提前 250ms， 唤醒时延后 1s； 如果 RSMRSTn 有效过，该字段为 2'b0。重新上电后由系统配置该字段。
10:9	保留
8	**USBPHY_LP_EN - R/W** 进入 G2/S5 状态时使能 USB PHY 进入低功耗状态。
7	**USB_GMAC_OK - R/W** 如果 RSMRSTn 有效过，该位为 0，表示 USB 和 GMAC 没有配置，不能唤醒系统。重新上电后由系统配置该位。
6	**CTT_STS - R/WC** 系统在 S0 状态时发生 temperature trip，系统进入 G2/S5 状态，该位为重新上电后系统检测记录事件状态
5	**CTT_EN - R/W** 使能 temperature trip 保护机制
4	**LID_OPEN - RO** 显示屏状态检测位

	1：显示屏打开 0：显示屏合上
3	保留
2	**SRS (System Reset Status) – R/WC** 0：SYS_RESETn 没有被按下 1: SYS_RESETn 被按下过，系统重新复位后需检查此位并作出相应清除操作。
1	**PWROK_FLR (PWROK Failure) – R/WC** 当系统在 S0 状态时，PWROK 信号变无效该位置 1，软件通过写 1 将该位清除。
0	**DRAM_INIT – R/W** 该位不影响硬件功能，PMON 在进行 DRAM 初始化前将该位置 1，结束 DRAM 初始化后将该位写 0，软件可通过此位检查 DRAM 初始化是否被打断过。

■ PMCON_RTC : RTC General PM Configuration Register

地址偏移	电压域	属性
0x08	RTC	R/W，R/WC

位域	描述
31:9	保留
8	**WOL_EN – R/W** enable wake on LAN don’t shut off SLP_LANn, use with WOL_BAT_EN 使能 wake on LAN，与 WOL_BAT_EN 共同使用
7	**WOL_BAT_EN – R/W** 如果系统使用电池供电，如果该位置 1，使能 WOL，如果该位置 0，不论 WOL_EN，WOL 无效
6:5	**S3_ASSERTION_WIDTH – R/W** 这 2 bit 值代表 S3n 信号从有效到重新无效的最小时间间隔。 11: 2s 10: 125ms 01: 2ms 00: 120us
4:3	**S4_ASSERTION_WIDTH – R/W** 这 2 bit 值代表 S4n 信号从有效到重新无效的最小时间间隔。 11: 4 seconds 10: 3 seconds 01: 2 seconds 00: 1 seconds
2	**S4_ASSERTION_EN – R/W** 0: S4n 信号从有效到重新无效的间隔为 64RTC 周期。 1: S4n 信号从有效到重新无效的间隔为 S4_ASSERTION_WIDTH 决定。
1	**PWR_FLR (Power Failure) – R/WC**. 该位在 RTC 域，只能被 RTC_RSTn 复位。 如果置 1 表示系统发生过电源失效（进入 G3 状态），即除 RTC 以外所有供电失效过（RSMRSTn 有效过），软件通过写 1 将该位清除。
0	**AFTERG3_EN – R/W** 该位决定系统进入 G3 状态后重新供电后的动作。 0：系统在供电恢复后将自动回复到 S0 状态。 1：系统将恢复到 S5 状态，如果发生电源失效时系统在 S4 状态，重新供电后系统恢复到 S4 状态。 该位会被 power button override 和 thermal trip 事件置 1。

■ PM1_STS : Power Management 1 Status Register

地址偏移	电压域	属性
0x0C	RESUME/RTC/SOC	R/WC

位域	描述	电压域
31:16	Reserved	
15	**WAK_STS (Wake Status) – R/WC** 0：软件写 1 将该位清除。 1：如果系统从任何一个休眠状态被唤醒事件唤醒，硬件将该位置 1。	Resume

14	**PCIEXP_WAKE_STS － R/WC.** 1：PCIE 唤醒事件发生 0：软件写 1 将该位清除	Resume
13:12	Reserved	
11	**PRBTNOR_STS (Power Button Override Status) － R/WC** 0：软件写 1 将该位清除 1：当 power button override 发生时，该位置 1，系统无条件进入 G2/S5 状态，同时将 AFTERG3_EN 位置 1。	RTC
10	**RTC_STS (RTC Status) － R/WC** 0：软件写 1 将该位清除 1: 当 RTC 产生 alarm 时该位置 1。此外当 RTC_EN 有效时，该位产生唤醒事件。	Resume
9	Reserved	
8	**PWRBTN_STS (Power Button Status) － R/WC** 0：软件写 1 将该位清除。Thermal trip 会清除该位。 1：当按下 PWRBTNn 保持 16ms 以上（4s 以下）时，该位会置 1。 在 S0 状态时，当 PWRBTN_EN 和 PWRBTN_STS 同时有效时，将产生中断。 在 S3-S5 任何休眠状态时，如果 PWRBTN_STS 置位，系统将恢复。	Resume
7:5	保留	
4	**BM_STS (Bus Master Status) － R/WC** 0：软件写 1 将该位清除 1：该位监视 bus master 上的非一致性访问。	SOC
3:1	保留	SOC
0	**TMROF_STS (PM Timer Overflow Status) － R/WC** 0：软件写 1 将该位清除 1：当 24bit 计数器（周期 8ns）的最高位翻转时，该位置 1，该记时功能推荐使用 HPET 完成。	SOC

■ PM1_EN : Power Management 1 Enable Register

地址偏移	电压域	属性
0x10	RESUME/RTC/SOC	R/W

位域	描述	电压域
31:15	保留	
14	**PCIEXP_WAKE_DIS － R/W** 当置位时，不产生 PCIE 唤醒事件，但是该位的值不影响 PCIEXP_WAKE_STS 的值。	resume
13:11	保留	
10	**RTC_EN (RTC Event Enable) － R/W** RTC 唤醒和中断使能	rtc
9	保留	
8	**PWRBTN_EN (Power Button Enable) － R/W.** PWRBTN 中断事件产生使能，该位不影响 PWRBTN 唤醒事件产生。	resume
7:1	保留	
0	**TMROF_EN (PM Timer Overflow Enable) － R/W.** 如果该位置位，TMROF_STS 将产生中断。	SOC

■ PM1_CNT : Power Management 1 Control Register

地址偏移	电压域	属性
0x14	RESUME/RTC/SOC	R/W

位域	描述	电压域
31:14	保留	
13	**SLP_EN (Sleep Enable) － R/W.** 该位写 1 将会使系统进入 SLP_TYP 声明的休眠状态，进入相关休眠状态后该位自动恢复为 0。	resume
12:10	**SLP_TYP (Sleep Type) － R/W.** 该 3bit 表示系统的休眠状态。	rtc

	000: 表示 S0 状态。 001: Reserved. 010: Reserved. 011: Reserved. 100: Reserved. 101: Suspend-to-RAM. S3n 信号有效，进入 S3 状态。 110: Suspend-to-Disk. S3n, S4n 信号有效，进入 S4 状态。 111: Soft Off. S3n, S4n, S5n 信号有效，进入 S5 状态。	
9:2	Reserved	
1	**BM_RLD (Bus master Reload) – R/W** 如果置位，使能 BM_STS 产生打断事件，使得系统从 C2/C3 状态恢复。	SOC
0	**INT_EN – R/W** 中断使能开关，使能电源管理控制器中断信号的产生。	SOC

■ PM1_TMR : Power Management 1 Timer

地址偏移	电压域	属性
0x18	SOC	RO

位域	描述
31:24	保留
23:0	**TMR_VAL (Timer Value) – RO.** 计数器计数，周期 8ns。当 23 位翻转时，置位 TNROF_STS 位。 推荐使用 HPET。

■ GPE0_STS : General Purpose Event0 Status Register

地址偏移	电压域	属性
0x28	RESUME	R/WC

位域	描述
31:9	保留
8	**RI_STS – R/WC.** 0：软件写 1 将该位清除。 1：当 RIn 信号有效时被置位。
7	**BATLOW_STS – R/WC.** 0：软件写 1 将该位清除。 1：当 BATLOWn 信号有效时被置位 如果 BATLOW_EN 有效，BATLOW_STS 将产生中断。该位不产生唤醒事件。
6	保留
5	**GMAC0_STS – R/WC.** 0：软件写 1 将该位清除。 1：当 GMAC0 发生 wake 事件时这些位被置位，当 GMAC0_EN 位有效时，产生唤醒事件或中断。
4	**LID_STS – R/WC.** 0：软件写 1 将该位清除。 1：当 LID_EN 位有效时，产生唤醒事件。
3	**CTW_STS – R/WC**. CPU thermal warning 发生
2	**CTA_STS – R/WC**. CPU thermal alert 发生
1	**PWRSWITCH_STS – R/WC.** 供电状态发生变化，PWRTYP 变化。该位会产生中断。
0	保留

■ GPE0_EN : General Purpose Event0 Status Register

地址偏移	电压域	属性
0x2C	RESUME/RTC	R/W

位域	描述	电压域
31:9	保留	
8	**RI_EN - R/W.** 0：无效 1：使能 RIn_STS 产生唤醒事件，当回到 S0 将产生中断信号。	rtc
7	**BATLOW_EN - R/W.** 0: 无效 1: 使能 BATLOWn 产生中断事件。	rtc
6	保留	rtc
5	**GMAC0_EN - R/W.** 0：无效 1：使能 GMAC0_STS 产生唤醒事件，当回到 S0 将产生中断信号。	
4	**LID_EN - R/W.** 0：无效 1：使能 LID_STS 产生唤醒事件， S0 状态时将产生中断信号。	
3	**CTW_EN - R/W** 使能 CPU THERMAL WARNING 产生中断。	
2	**CTA_EN - R/W** 使能 CPU THERMAL ALERT 产生中断。	
1	**PWRSWITCH_EN - R/W** 使能 PWRSWITCH_STS 产生中断。	
0	**LID_POL - R/W** 该位设置 LID 的极性。 0: LID 为低时置位 LID_STS bit. 1: LID 为高时置位 LID_STS bit.	

■ RST_CNT : Reset Control Register

地址偏移	电压域	属性
0x30	SOC	R/W

位域	描述
31:2	保留
1	**WD_EN - R/W** Watch dog 功能使能
0	**OS_RST - R/W** 软件写该位使系统复位。

■ GEN_RTC_1 : General RTC Register 1

地址偏移	电压域	属性
0x50	RTC	R/W

位域	描述
31:0	RTC 通用寄存器

■ GEN_RTC_2 : General RTC Register 2

地址偏移	电压域	属性
0x54	RTC	R/W

位域	描述
31:0	RTC 通用寄存器

29.3.2 WDT 寄存器描述

看门狗(WDT)配置寄存器基址为：0x1ff6_c000。

■ WD_SET : Watch Dog Set Register

地址偏移	电压域	属性
0x34	SOC	WO

位域	描述
31:1	保留
0	写该位将重填 watch dog 计数器，配置该位需首先配置 WDT 使能位(即:RST_CNT[1])。

■ WD_Timer : Watch Dog Timer Register

地址偏移	电压域	属性
0x38	SOC	R/W

位域	描述
31:0	该寄存器的值为 watch dog 重填的值，复位后为全 1。

29.3.3 DPM/DVFS 寄存器描述

DPM/DVFS 配置寄存器基址为：0x1ff6_0000。

■ DPM_EN : DPM Enable Register

地址偏移	电压域	属性
0x00	SOC	R/W

位域	描述	电压域
31:13	保留	
12	**PRINT_EN – R/W** PRINT 模块动态电源管理(DPM)使能位，使能 PRINT 模块动态电源管理，DPM 可对 PRINT 模块进行动态时钟或电源关断控制。	SOC
11	**HDA_EN – R/W** HDA 模块动态电源管理(DPM)使能位，使能 HDA 模块动态电源管理，DPM 可对 HDA 模块进行动态时钟或电源关断控制。	SOC
10	**USB3_EN – R/W** USB3 模块动态电源管理(DPM)使能位，使能 USB3 模块动态电源管理，DPM 可对 USB3 模块进行动态时钟或电源关断控制。	SOC
9	**USB_EN – R/W** USB 模块动态电源管理(DPM)使能位，使能 USB 模块动态电源管理，DPM 可对 USB 模块进行动态时钟或电源关断控制。	SOC
8	**SATA_EN – R/W** SATA 模块动态电源管理(DPM)使能位，使能 SATA 模块动态电源管理，DPM 可对 SATA 模块进行动态时钟或电源关断控制。	SOC
7	**GMAC1_EN – R/W** GMAC1 模块动态电源管理(DPM)使能位，使能 GMAC1 模块动态电源管理，DPM 可对 GMAC1 模块进行动态时钟或电源关断控制。	SOC
6	**GMAC0_EN – R/W** GMAC0 模块动态电源管理(DPM)使能位，使能 GMAC0 模块动态电源管理，DPM 可对 GMAC0 模块进行动态时钟或电源关断控制。	SOC
5	**DC_EN – R/W** DC 模块动态电源管理(DPM)使能位，使能 DC 模块动态电源管理，DPM 可对 DC 模块进行动态时钟或电源关断控制。	SOC
4	**GPU_EN – R/W** GPU 模块动态电源管理(DPM)使能位，使能 GPU 模块动态电源管理，DPM 可对 GPU 模块进行动态时钟或电源关断控制。	SOC
3	**PCIE_EN – R/W** PCIE 模块动态电源管理(DPM)使能位，使能 PCIE 模块动态电源管理，DPM	SOC

	可对 PCIE 模块进行动态时钟或电源关断控制。	
2	**PCI_EN – R/W** PCI 模块动态电源管理(DPM)使能位，使能 PCI 模块动态电源管理，DPM 可对 PCI 模块进行动态时钟或电源关断控制。	SOC
1	**DDR_EN – R/W** DDR 模块动态电源管理(DPM)使能位，使能 DDR 模块动态电源管理，DPM 可对 DDR 模块进行动态时钟或电源关断控制。	SOC
0	**NODE_EN – R/W** NODE 模块动态电源管理(DPM)使能位，使能 NODE 模块动态电源管理，DPM 可对 NODE 模块进行动态时钟或电源关断控制。	SOC

■ PWRUP_SEL : POWERUP Select Register

地址偏移	电压域	属性
0x04	SOC	R/W

位域	描述	电压域
31:13	保留	
12	**PRINT_SEL – R/W(暂不支持)** PRINT 模块上电完成应答选择： 1：选择 PRINT 模块本身上电完成标志；0：选择 print_pwrup_time 计数结束标志。	SOC
11	**HDA_SEL – R/W(暂不支持)** HDA 模块上电完成应答选择： 1：选择 HDA 模块本身上电完成标志；0：选择 hda_pwrup_time 计数结束标志。	SOC
10	**USB3_SEL – R/W** USB3 模块上电完成应答选择： 1：选择 USB3 模块本身上电完成标志；0：选择 usb3_pwrup_time 计数结束标志。	SOC
9	**USB_SEL – R/W** USB 模块上电完成应答选择： 1：选择 USB 模块本身上电完成标志；0：选择 usb_pwrup_time 计数结束标志。	SOC
8	**SATA_SEL – R/W** SATA 模块上电完成应答选择： 1：选择 SATA 模块本身上电完成标志；0：选择 sata_pwrup_time 计数结束标志。	SOC
7	**GMAC1_SEL – R/W(暂不支持)** GMAC1 模块上电完成应答选择： 1：选择 GMAC1 模块本身上电完成标志；0：选择 gmac1_pwrup_time 计数结束标志。	SOC
6	**GMAC0_SEL – R/W(暂不支持)** GMAC0 模块上电完成应答选择： 1：选择 GMAC0 模块本身上电完成标志；0：选择 gmac0_pwrup_time 计数结束标志。	SOC
5	**DC_SEL – R/W(暂不支持)** DC 模块上电完成应答选择： 1：选择 DC 模块本身上电完成标志；0：选择 dc_pwrup_time 计数结束标志。	SOC
4	**GPU_SEL – R/W** GPU 模块上电完成应答选择： 1：选择 GPU 模块本身上电完成标志；0：选择 gpu_pwrup_time 计数结束标志。	SOC
3	**PCIE_SEL – R/W** PCIE 模块上电完成应答选择： 1：选择 PCIE 模块本身上电完成标志；0：选择 pcie_pwrup_time 计数结束标志。	SOC
2	**PCI_SEL – R/W(暂不支持)** PCI 模块上电完成应答选择： 1：选择 PCI 模块本身上电完成标志；0：选择 pci_pwrup_time 计数结束标志。	SOC
1	**DDR_SEL – R/W(暂不支持)** DDR 模块上电完成应答选择： 1：选择 DDR 模块本身上电完成标志；0：选择 ddr_pwrup_time 计数结束标志。	SOC
0	**NODE_SEL – R/W(暂不支持)** NODE 模块上电完成应答选择： 1：选择 NODE 模块本身上电完成标志；0：选择 node_pwrup_time 计数结束标志。	SOC

■ DPM_TGT : DPM Target Register

地址偏移	电压域	属性
0x08	SOC	R/W

位域	描述	电压域
31:13	保留	
25:24	**PRINT_ TGT - R/W** PRINT 模块动态电源目标状态选择： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态(暂不支持)。	SOC
23:22	**HDA_ TGT - R/W** HDA 模块动态电源目标状态选择： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态(暂不支持)。	SOC
21:20	**USB3_ TGT - R/W** USB3 模块动态电源目标状态选择： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态。	SOC
19:18	**USB_ TGT - R/W** USB 模块动态电源目标状态选择： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态。	SOC
17:16	**SATA_ TGT - R/W** SATA 模块动态电源目标状态选择： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态。	SOC
15:14	**GMAC1_ TGT - R/W** GMAC1 模块动态电源目标状态选择： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态(暂不支持)。	SOC
13:12	**GMAC0_ TGT - R/W** GMAC0 模块动态电源目标状态选择： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态(暂不支持)。	SOC
11:10	**DC_ TGT - R/W** DC 模块动态电源目标状态选择： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态。	SOC
9:8	**GPU_ TGT - R/W** GPU 模块动态电源目标状态选择： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态。	SOC
7:6	**PCIE_ TGT - R/W** PCIE 模块动态电源目标状态选择： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态。	SOC
5:4	**PCI_ TGT - R/W** PCI 模块模块动态电源目标状态选择： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态(暂不支持)。	SOC
3:2	**DDR_ TGT - R/W** DDR 模块模块动态电源目标状态选择： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态(暂不支持)。	SOC
1:0	**NODE_TGT - R/W** NODE 模块动态电源目标状态选择： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态(暂不支持)。	SOC

■ DPM_STS : DPM Status Register

地址偏移	电压域	属性
0x0c	SOC	RO

位域	描述	电压域
31:13	保留	
25:23	**PRINT_ STS - RO** PRINT 模块动态电源当前状态： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态(暂不支持)。	SOC
23:22	**HDA_ STS - RO** HDA 模块动态电源当前状态： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态(暂不支持)。	SOC
21:20	**USB3_ STS - RO** USB3 模块动态电源当前状态： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态。	SOC
19:18	**USB_ STS - RO** USB 模块动态电源当前状态： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态。	SOC
17:16	**SATA_ STS - RO** SATA 模块动态电源当前状态： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态。	SOC
15:14	**GMAC1_ STS - RO** GMAC1 模块动态电源当前状态： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态(暂不支持)。	SOC
13:12	**GMAC0_ STS - RO** GMAC0 模块动态电源当前状态： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态(暂不支持)。	SOC
11:10	**DC_ STS - RO** DC 模块动态电源当前状态： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态。	SOC
9:8	**GPU_ STS - RO** GPU 模块动态电源当前状态： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态。	SOC
7:6	**PCIE_ STS - RO** PCIE 模块动态电源当前状态： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态。	SOC
5:4	**PCI_ STS - RO** PCI 模块模块动态电源当前状态： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态(暂不支持)。	SOC
3:2	**DDR_ STS - RO** DDR 模块模块动态电源当前状态： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态(暂不支持)。	SOC
1:0	**NODE_ STS - RO** NODE 模块动态电源当前状态： 00：全部开启状态；01：模块时钟关断状态；10：保留；11：模块电源关断状态(暂不支持)。	SOC

■ WAIT_TIME0：Wait Time0 Register

地址偏移	电压域	属性
0x10	SOC	R/W

位域	描述	电压域
31:28	**PCIE_PWRUP_TIME - R/W** PCIE 模块上电完成等待计数值：0~15。	SOC
27:24	**PCIE_RST_WTIME - R/W**	SOC

	PCIE 模块复位完成等待计数值：0~15。	
23:20	**PCI_PWRUP_TIME – R/W** PCI 模块上电完成等待计数值：0~15。	SOC
19:16	**PCI_RST_WTIME – R/W** PCI 模块复位完成等待计数值：0~15。	SOC
15:12	**DDR_PWRUP_TIME – R/W** DDR 模块上电完成等待计数值：0~15。	SOC
11:8	**DDR_RST_WTIME – R/W** DDR 模块复位完成等待计数值：0~15。	SOC
7:4	**NODE_PWRUP_TIME – R/W** NODE 模块上电完成等待计数值：0~15。	SOC
3:0	**NODE_RST_WTIME – R/W** NODE 模块复位完成等待计数值：0~15。	SOC

■ WAIT_TIME1 : Wait Time1 Register

地址偏移	电压域	属性
0x14	SOC	R/W

位域	描述	电压域
31:28	**GMAC1_PWRUP_TIME – R/W** GMAC1 模块上电完成等待计数值：0~15。	SOC
27:24	**GMAC1_RST_WTIME – R/W** GMAC1 模块复位完成等待计数值：0~15。	SOC
23:20	**GMAC0_PWRUP_TIME – R/W** GMAC0 模块上电完成等待计数值：0~15。	SOC
19:16	**GMAC0_RST_WTIME – R/W** GMAC0 模块复位完成等待计数值：0~15。	SOC
15:12	**DC_PWRUP_TIME – R/W** DC 模块上电完成等待计数值：0~15。	SOC
11:8	**DC_RST_WTIME – R/W** DC 模块复位完成等待计数值：0~15。	SOC
7:4	**GPU_PWRUP_TIME – R/W** GPU 模块上电完成等待计数值：0~15。	SOC
3:0	**GPU_RST_WTIME – R/W** GPU 模块复位完成等待计数值：0~15。	SOC

■ WAIT_TIME2 : Wait Time2 Register

地址偏移	电压域	属性
0x18	SOC	R/W

位域	描述	电压域
31:28	**HDA_PWRUP_TIME – R/W** HDA 模块上电完成等待计数值：0~15。	SOC
27:24	**HDA_RST_WTIME – R/W** HDA 模块复位完成等待计数值：0~15。	SOC
23:20	**USB3_PWRUP_TIME – R/W** USB3 模块上电完成等待计数值：0~15。	SOC
19:16	**USB3_RST_WTIME – R/W** USB3 模块复位完成等待计数值：0~15。	SOC
15:12	**USB_PWRUP_TIME – R/W** USB 模块上电完成等待计数值：0~15。	SOC
11:8	**USB_RST_WTIME – R/W** USB 模块复位完成等待计数值：0~15。	SOC
7:4	**SATA_PWRUP_TIME – R/W** SATA 模块上电完成等待计数值：0~15。	SOC
3:0	**SATA_RST_WTIME – R/W**	SOC

	SATA 模块复位完成等待计数值：0~15。	

■ WAIT_TIME3 : Wait Time3 Register

地址偏移	电压域	属性
0x1c	SOC	R/W

位域	描述	电压域
31:8	保留	
7:4	**PRINT_PWRUP_TIME – R/W** PRINT 模块上电完成等待计数值：0~15。	SOC
3:0	**PRINT_RST_WTIME – R/W** PRINT 模块复位完成等待计数值：0~15。	SOC

■ DVFS_CFG : DVFS Config Register

地址偏移	电压域	属性
0x20	SOC	R/W

位域	描述	电压域
31:30	保留	
29:24	**la132_int – R/W** la132 处理器核 5 个外部中断软件输入，高电平有效(默认值均为 0)	SOC
23:21	保留	
20	**la132_nmi – R/W** la132 处理器核 NMI 中断软件输入配置位，高电平有效(默认值为 0)	SOC
19:18	保留	
17	**la132_clken – R/W** la132 处理器核时钟使能配置位，低电平使能(默认打开，初值为 0)	SOC
16	**la132_soft_rstn – R/W** la132 处理器核软复位配置位，低电平复位有效(默认复位有效，初值为 0)。	SOC
15:0	保留	

■ DVFS_CTRL : DVFS Control Register

地址偏移	电压域	属性
0x24	SOC	R/W

位域	描述	电压域
31:15	保留	
14:8	**DVFS_CLK_ODIV – R/W** NODE 模块 PLL 时钟输出分频参数配置，最高位(odiv[14])为软件参数配置使能位(高有效)： 对应 PLL 时钟配置参数 ODIV[5:0]：0~63	SOC
7:3	保留	
2	**DVFS_CLK_OFF – R/W** NODE 模块时钟关断配置位，高电平时钟关断，默认值为 0。	SOC
1	**DVFS_CLKBYP – R/W** NODE 模块时钟 BYPASS 配置位，默认值为 0： 1：NODE 时钟 BYPASS 为系统参考时钟；0：NODE 时钟为 PLL 输出时钟。	SOC
0	**DVFS_EN – R/W** NODE 模块动态调频调压(DVFS)使能位，高电平有效，默认值为 0。	SOC

■ CG_CFG : Clkgate Config Register

地址偏移	电压域	属性
0x28	SOC	R/W

位域	描述	电压域

31:6	保留	
5:4	**USB_CG_EN － R/W** USB/OTG 时钟关断控制有效位，高电平时钟可关断，默认值为 00： [4]：USB0~3 时钟关断控制有效位；[5]：OTG 时钟关断控制有效位。	SOC
3:2	**PCIE_CG_EN － R/W** PCIE0~1 时钟关断控制有效位，高电平时钟可关断，默认值为 00： [0]：PCIE0 时钟关断控制有效位；[1]：PCIE1 时钟关断控制有效位。	SOC
1:0	**SATA_CG_EN － R/W** SATA0~1 时钟关断控制有效位，高电平时钟可关断，默认值为 00： [0]：SATA0 时钟关断控制有效位；[1]：SATA1 时钟关断控制有效位。	SOC

30 RTC

30.1 概述

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，该单元仍然运作，可以仅靠板上的电池供电就正常运行。RTC 单元运行时电流仅几个微安。

RTC 包含振荡器，结合外部 32.768KHZ 晶体产生工作时钟。该时钟用于时间信息的维护以及产生各种定时和计数中断。

RTC 模块中包含两个计数器，分别为 TOY（Time of Year）计数器和 RTC 计数器。其中 TOY 计数器按年月日时分秒计数，精度为以 0.1 秒；RTC 计数器以 32.768KHz 时钟计数，宽度为 32 位。

30.2 寄存器描述

RTC 模块寄存器位于 0x1ff6c100——0x1ff6c1ff 的 256B 地址空间内，其基地址为 0X1ff6c100，所有寄存器位宽均为 32 位。

30.2.1 寄存器地址列表

名称	偏移地址	位宽	RW	描述
sys_toytrim	0x20	32	RW	软件必须初始化为 0
sys_toywrite0	0x24	32	W	TOY 低 32 位数值写入
sys_toywrite1	0x28	32	W	TOY 高 32 位数值写入
sys_toyread0	0x2C	32	R	TOY 低 32 位数值读出
sys_toyread1	0x30	32	R	TOY 高 32 位数值读出
sys_toymatch0	0x34	32	RW	TOY 定时中断 0
sys_toymatch1	0x38	32	RW	TOY 定时中断 1
sys_toymatch2	0x3C	32	RW	TOY 定时中断 2
sys_rtcctrl	0x40	32	RW	TOY 和 RTC 控制寄存器 软件必须初始化
sys_rtctrim	0x60	32	RW	软件必须初始化为 0
sys_rtcwrite0	0x64	32	W	RTC 定时计数写入
sys_rtcread0	0x68	32	R	RTC 定时计数读出
sys_rtcmatch0	0x6C	32	RW	RTC 时钟定时中断 0
sys_rtcmatch1	0x70	32	RW	RTC 时钟定时中断 1
sys_rtcmatch2	0x74	32	RW	RTC 时钟定时中断 2

30.2.2 SYS_TOYWRITE0

中文名： TOY 计数器低 32 位数值

寄存器位宽： [31：0]

偏移量： 0x24

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

30.2.3 SYS_TOYWRITE1

中文名： TOY 计数器高 32 位数值

寄存器位宽： [31：0]

偏移量： 0x28

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

30.2.4 SYS_TOYREAD0

中文名： TOY 计数器低 32 位数值

寄存器位宽： [31：0]

偏移量： 0x2C

复位值： 0x00000000

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

30.2.5 SYS_TOYREAD1

中文名： TOY 计数器高 32 位数值

寄存器位宽： [31：0]

偏移量： 0x30

复位值： 0x00000000

位域	位域名称	访问	缺省	描述
31:0	TOY_YEAR	R	0	年，范围 0～16383

30.2.6 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	0	年，范围 0～16383
25:22	MONTH	RW	0	月，范围 1~12
21:17	DAY	RW	0	日，范围 1~31
16:12	HOUR	RW	0	小时，范围 0~23
11:6	MIN	RW	0	分，范围 0~59
5:0	SEC	RW	0	秒，范围 0~59

30.2.7 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： 无

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	保留	R	0	保留，置 0
13	REN	R/W	0	RTC 使能，高有效。需要初始化为 1
12	保留	R	0	保留，置 0
11	TEN	R/W	0	TOY 使能，高有效。需要初始化为 1
10	保留	R	0	保留，置 0
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	保留	R	0	保留，置 0
6	保留	R	0	保留，置 0
5	32S	R	0	0：32.768k 晶振不工作； 1：32.768k 晶振正常工作。
4	保留	R	0	保留，置 0
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

30.2.8 SYS_RTCWRITE

中文名： RTC 计数器写入端口

寄存器位宽： [31：0]

偏移量： 0x64

复位值： 0x00000000

位域	位域名称	访问	缺省	描述
31:0	RTCWRITE	W	-	RTC 计数器写入寄存器

30.2.9 SYS_RTCREAD

中文名： RTC 计数器读出端口

寄存器位宽： [31：0]

偏移量： 0x68

复位值： 0x00000000

位域	位域名称	访问	缺省	描述
31:0	RTCREAD	R	0	RTC 计数器读出寄存器

30.2.10 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	RTCMATCH0/1/2	RW	0	RTC 定时比较寄存器 0/1/2

31 GPIO

龙芯 2K0500 共有 155 个 GPIO 引脚，全部与其他功能引脚复用。其中，GPIO0~127 支持外部中断输入。下面具体介绍与 GPIO 相关的配置寄存器。

表 31-1 GPIO 配置寄存器

地址	名称	描述
0x1fe10430	GPIO0_OEN	GPIO0~31 位输出使能
0x1fe10434	GPIO1_OEN	GPIO32~63 位输出使能
0x1fe10438	GPIO0_IN	GPIO0~31 位输入值
0x1fe1043c	GPIO1_IN	GPIO32~63 位输入值
0x1fe10440	GPIO0_OUT	GPIO0~31 位输出值
0x1fe10444	GPIO1_OUT	GPIO32~63 位输出值
0x1fe10450	GPIO2_OEN	GPIO64~95 位输出使能
0x1fe10454	GPIO3_OEN	GPIO96~127 位输出使能
0x1fe10458	GPIO2_IN	GPIO64~95 位输入值
0x1fe1045c	GPIO3_IN	GPIO96~127 位输入值
0x1fe10460	GPIO2_OUT	GPIO64~95 位输出值
0x1fe10464	GPIO3_OUT	GPIO96~127 位输出值
0x1fe10470	GPIO4_OEN	GPIO128~154 位输出使能
0x1fe10478	GPIO4_IN	GPIO128~154 位输入值
0x1fe10480	GPIO4_OUT	GPIO128~154 位输出值
0x1fe104e0	GPIO_INTEN0	GPIO0~31 中断输入使能寄存器
0x1fe104e4	GPIO_INTEN1	GPIO32~63 中断输入使能寄存器
0x1fe104e8	GPIO_INTEN2	GPIO64~95 中断输入使能寄存器
0x1fe104ec	GPIO_INTEN3	GPIO96~127 中断输入使能寄存器
0x1fe10490 ~ 0x1fe104dc	GPIO_CFG0 ~ GPIO_CFG19	GPIO0~154 复用配置寄存器

31.1GPIO 方向控制

表 31-2 GPIO 方向控制

地址	位域	名称	访问	初值	描述
0x1fe10430	31:0	GPIO0_OEN	R/W	32'hFFFF_FFFF	GPIO0~31 位输出使能: 0 为输出，1 为输入
0x1fe10434	31:0	GPIO1_OEN	R/W	32'hFFFF_FFFF	GPIO32~63 位输出使能: 0 为输出，1 为输入
0x1fe10450	31:0	GPIO2_OEN	R/W	32'hFFFF_FFFF	GPIO64~95 位输出使能: 0 为输出，1 为输入
0x1fe10454	31:0	GPIO3_OEN	R/W	32'hFFFF_FFFF	GPIO96~127 位输出使能: 0 为输出，1 为输入
0x1fe10470	31:0	GPIO4_OEN	R/W	32'hFFFF_FFFF	GPIO127~154 位输出使能: 0 为输出，1 为输入

31.2GPIO 输出设置

表 31-3 GPIO 输出设置

地址	位域	名称	访问	初值	描述
0x1fe10440	31:0	GPIO0_OUT	R/W	0	GPIO0~31 位输出值: 0 输出低电平，1 输出高电平
0x1fe10444	31:0	GPIO1_OUT	R/W	0	GPIO32~63 位输出值: 0 输出低电平，1 输出高电平
0x1fe10460	31:0	GPIO2_OUT	R/W	0	GPIO64~95 位输出值: 0 输出低电平，1 输出高电平
0x1fe10464	31:0	GPIO3_OUT	R/W	0	GPIO96~127 位输出值: 0 输出低电平，1 输出高电平
0x1fe10480	31:0	GPIO4_OUT	R/W	0	GPIO127~154 位输出值: 0 输出低电平，1 输出高电平

31.3GPIO 输入采样

表 31-4 GPIO 输入采样

地址	位域	名称	访问	初值	描述
0x1fe10438	31:0	GPIO0_IN	R	0	GPIO0~31 位输入值: 0 输入低电平，1 输入高电平
0x1fe1043c	31:0	GPIO1_IN	R	0	GPIO32~63 位输入值: 0 输入低电平，1 输入高电平
0x1fe10458	31:0	GPIO2_IN	R	0	GPIO64~95 位输入值: 0 输入低电平，1 输入高电平
0x1fe1045c	31:0	GPIO3_IN	R	0	GPIO96~127 位输入值: 0 输入低电平，1 输入高电平
0x1fe10478	31:0	GPIO4_IN	R	0	GPIO127~154 位输入值: 0 输入低电平，1 输入高电平

31.4GPIO 中断使能

表 31-5 GPIO 中断使能

地址	位域	名称	访问	初值	描述
0x1fe104e0	31:0	GPIO_INTEN0	R/W	0	GPIO0~31 位中断使能位，高有效，32 位引脚输入中断共享 1 位中断状态位，可在中断控制器状态寄存器查询
0x1fe104e4	31:0	GPIO_INTEN1	R/W	0	GPIO32~63 位中断使能位,高有效，32 位引脚输入中断共享 1 位中断状态位，可在中断控制器状态寄存器查询
0x1fe104e8	31:0	GPIO_INTEN2	R/W	0	GPIO64~95 位中断使能位,高有效，32 位引脚输入中断共享 1 位中断状态位，可在中断控制器状态寄存器查询
0x1fe104ec	31:0	GPIO_INTEN3	R/W	0	GPIO96~127 位中断使能位，高有效，32 位引脚输入中断共享 1 位中断状态位,可在中断控制器状态寄存器查询

31.5GPIO 复用关系

GPIO 与其他功能的复用关系如下表所示：

表 31-6 GPIO 复用关系

GPIO 编号	芯片信号 (主功能)	第一复用	第二复用	第三复用	第四复用
GPIO0	sys_int[0]	-	-	gmac0_ptp_trig	-
GPIO1	sys_int[1]	-	-	gmac0_ptp_pps	-

GPIO2	vga_hsync	-	-	gmac1_ptp_trig	-
GPIO3	vga_vsync	-	pr_int	gmac1_ptp_pps	-
GPIO4	lcd_clk	can2_rx	pr0_clk	-	-
GPIO5	lcd_vsync	can2_tx	pr0_start	-	-
GPIO6	lcd_hsync	can3_rx	pr0_ready	-	-
GPIO7	lcd_en	can3_tx	pr0_enable	-	-
GPIO8	lcd_dat[0]	uart0_tx	pr0_shold	-	-
GPIO9	lcd_dat[1]	uart0_rx	pr0_data	-	-
GPIO10	lcd_dat[2]	uart0_rts	pr0_hsync	-	-
GPIO11	lcd_dat[3]	uart0_cts	pr1_enable	-	-
GPIO12	lcd_dat[4]	uart0_dsr	pr1_shold	-	-
GPIO13	lcd_dat[5]	uart0_dtr	pr1_data	-	-
GPIO14	lcd_dat[6]	uart0_dcd	pr2_clk	-	-
GPIO15	lcd_dat[7]	uart0_ri	pr2_start	-	-
GPIO16	lcd_dat[8]	uart1_rx	pr2_ready	-	-
GPIO17	lcd_dat[9]	uart1_tx	pr2_enable	-	-
GPIO18	lcd_dat[10]	uart1_rts	pr2_shold	-	-
GPIO19	lcd_dat[11]	uart1_cts	pr2_data	-	-
GPIO20	lcd_dat[12]	uart1_dsr	pr2_hsync	-	-
GPIO21	lcd_dat[13]	uart1_dtr	pr3_enable	-	-
GPIO22	lcd_dat[14]	uart1_dcd	pr3_shold	-	-
GPIO23	lcd_dat[15]	uart1_ri	pr3_data	-	-
GPIO24	lcd_dat[16]	-	pr4_clk	spi4_clk	-
GPIO25	lcd_dat[17]	-	pr4_start	spi4_miso	-
GPIO26	lcd_dat[18]	-	pr4_ready	spi4_mosi	uart0_rx
GPIO27	lcd_dat[19]	-	pr4_enable	spi4_cs	uart0_tx
GPIO28	lcd_dat[20]	-	pr4_shold	spi5_clk	uart0_rts
GPIO29	lcd_dat[21]	-	pr4_data	spi5_miso	uart0_cts
GPIO30	lcd_dat[22]	-	pr4_hsync	spi5_mosi	uart0_dsr
GPIO31	lcd_dat[23]	-	pr5_enable	spi5_cs	uart0_dtr
GPIO32	kb_clk	-	pr5_shold	spi3_clk	uart0_dcd
GPIO33	kb_dat	-	pr5_data	spi3_miso	uart0_ri
GPIO34	ms_clk	nand_rdy[2]	pr6_clk	spi3_mosi	pr_int
GPIO35	ms_dat	nand_ce[2]	pr6_start	spi3_cs	pr0_clk
GPIO36	ac97_datai	-	pr6_ready	pix0_scl	pr0_start
GPIO37	ac97_datao	-	pr6_enable	pix0_sda	pr0_ready
GPIO38	ac97_sync	-	pr6_shold	pix1_scl	pr0_enable
GPIO39	ac97_reset	-	pr6_data	pix1_sda	pr0_shold
GPIO40	spi0_clk	kb_clk	pr6_hsync	-	pr0_data
GPIO41	spi0_miso	kb_dat	pr7_enable	-	pr0_hsync
GPIO42	spi0_mosi	ms_clk	pr7_shold	-	pr1_enable
GPIO43	spi0_cs[0]	ms_dat	pr7_data	-	pr1_shold
GPIO44	spi1_clk	gmac1_tx[2]	gmac1_rx_ctl	nand_d[0]	pr1_data
GPIO45	spi1_miso	gmac1_tx[3]	gmac1_rx[0]	nand_d[1]	pr2_clk
GPIO46	spi1_mosi	gmac1_mdck	gmac1_rx[1]	nand_d[2]	pr2_start
GPIO47	spi1_cs[0]	gmac1_mdio	gmac1_rx[2]	nand_d[3]	pr2_ready
GPIO48	uart0_rx	gmac1_rx_ctl	-	scl0	pr2_enable
GPIO49	uart0_tx	gmac1_rx[0]	-	sda0	pr2_shold
GPIO50	uart0_rts	gmac1_rx[1]	pwm[0]	scl1	pr2_data

GPIO51	uart0_cts	gmac1_rx[2]	pwm[1]	sda1	pr2_hsync
GPIO52	uart0_dsr	gmac1_rx[3]	pwm[2]	scl2	pr3_enable
GPIO53	uart0_dtr	gmac1_tx_ctl	pwm[3]	sda2	pr3_shold
GPIO54	uart0_dcd	gmac1_tx[0]	pwm[4]	scl3	pr3_data
GPIO55	uart0_ri	gmac1_tx[1]	pwm[5]	sda3	pr4_clk
GPIO56	uart1_rx	gmac1_tx[2]	pwm[6]	spi0_clk	pr4_start
GPIO57	uart1_tx	gmac1_tx[3]	pwm[7]	spi0_miso	pr4_ready
GPIO58	uart1_rts	gmac1_mdck	pwm[8]	spi0_mosi	pr4_enable
GPIO59	uart1_cts	gmac1_mdio	pwm[9]	spi0_cs[0]	pr4_shold
GPIO60	uart2_tx	pix0_scl	pwm[10]	spi1_clk	pr4_data
GPIO61	uart2_rx	pix0_sda	pwm[11]	spi1_miso	pr4_hsync
GPIO62	uart3_tx	pix1_scl	pwm[12]	spi1_mosi	pr5_enable
GPIO63	uart3_rx	pix1_sda	pwm[13]	spi1_cs[0]	pr5_shold
GPIO64	scl0	nand_rdy[1]	pwm[14]	spi0_cs[3]	pr5_data
GPIO65	sda0	nand_ce[1]	pwm[15]	spi0_cs[2]	pr6_clk
GPIO66	can0_rx	nand_rdy[2]	sda2	spi0_cs[1]	pr6_start
GPIO67	can0_tx	nand_ce[2]	scl2	spi1_cs[3]	pr6_ready
GPIO68	can1_rx	nand_rdy[3]	sda3	spi1_cs[2]	pr6_enable
GPIO69	can1_tx	nand_ce[3]	scl3	spi1_cs[1]	pr6_shold
GPIO70	lpc_ad[0]	nand_d[0]	sda1	gmac1_rx_ctl	pr6_data
GPIO71	lpc_ad[1]	nand_d[1]	scl1	gmac1_rx[0]	pr6_hsync
GPIO72	lpc_ad[2]	nand_d[2]	sda2	gmac1_rx[1]	pr7_enable
GPIO73	lpc_ad[3]	nand_d[3]	scl2	gmac1_rx[2]	pr7_shold
GPIO74	lpc_frame	nand_d[4]	sda3	gmac1_rx[3]	pr7_data
GPIO75	lpc_serirq	nand_d[5]	scl3	-	-
GPIO76	nand_cle	-	-	-	pwm[0]
GPIO77	nand_ale	-	-	-	pwm[1]
GPIO78	nand_rd	-	-	-	pwm[2]
GPIO79	nand_wr	-	-	-	pwm[3]
GPIO80	nand_ce[0]	-	-	-	pwm[4]
GPIO81	nand_rdy[0]	gmac1_tx_ctl	-	-	pwm[5]
GPIO82	nand_d[6]	gmac1_tx[0]	-	-	pwm[6]
GPIO83	nand_d[7]	gmac1_tx[1]	-	-	pwm[7]
GPIO84	pwm[0]	can0_rx	gmac0_col	nand_rdy[1]	pwm[8]
GPIO85	pwm[1]	can0_tx	gmac0_crs	nand_ce[1]	pwm[9]
GPIO86	pwm[2]	gmac1_rx[3]	gmac1_col	nand_d[4]	pwm[10]
GPIO87	pwm[3]	-	gmac1_crs	nand_d[5]	pwm[11]
GPIO88	gmac0_rx_ctl	pwm[4]	-	uart1_dsr	pwm[12]
GPIO89	gmac0_rx[0]	pwm[5]	-	uart1_dtr	pwm[13]
GPIO90	gmac0_rx[1]	pwm[6]	-	uart1_dcd	pwm[14]
GPIO91	gmac0_rx[2]	pwm[7]	-	uart1_ri	pwm[15]
GPIO92	gmac0_rx[3]	pwm[8]	-	uart2_tx	nand_cle
GPIO93	gmac0_tx_ctl	pwm[9]	-	uart2_rx	nand_ale
GPIO94	gmac0_tx[0]	pwm[10]	-	uart3_tx	nand_rd
GPIO95	gmac0_tx[1]	pwm[11]	-	uart3_rx	nand_wr
GPIO96	gmac0_tx[2]	pwm[12]	pix0_scl	can2_rx	nand_ce[0]
GPIO97	gmac0_tx[3]	pwm[13]	pix0_sda	can2_tx	nand_rdy[0]
GPIO98	gmac0_mdck	pwm[14]	pix1_scl	can3_rx	nand_d[0]
GPIO99	gmac0_mdio	pwm[15]	pix1_sda	can3_tx	nand_d[1]

GPIO100	pci_ad[0]	pr_int	lioa[0]	pwm[0]	nand_d[2]
GPIO101	pci_ad[1]	pr0_clk	lioa[1]	pwm[1]	nand_d[3]
GPIO102	pci_ad[2]	pr0_start	lioa[2]	pwm[2]	nand_d[4]
GPIO103	pci_ad[3]	pr0_ready	lioa[3]	pwm[3]	nand_d[5]
GPIO104	pci_ad[4]	pr0_enable	lioa[4]	pwm[4]	nand_d[6]
GPIO105	pci_ad[5]	pr0_shold	lioa[5]	pwm[5]	nand_d[7]
GPIO106	pci_ad[6]	pr0_data	lioa[6]	pwm[6]	pix0_scl
GPIO107	pci_ad[7]	pr0_hsync	lioa[7]	pwm[7]	pix0_sda
GPIO108	pci_ad[8]	pr1_enable	lioa[8]	pwm[8]	pix1_scl
GPIO109	pci_ad[9]	pr1_shold	lioa[9]	pwm[9]	pix1_sda
GPIO110	pci_ad[10]	pr1_data	lioa[10]	pwm[10]	-
GPIO111	pci_ad[11]	pr2_clk	lioa[11]	pwm[11]	-
GPIO112	pci_ad[12]	pr2_start	lioa[12]	pwm[12]	-
GPIO113	pci_ad[13]	pr2_ready	lioa[13]	pwm[13]	spi2_clk
GPIO114	pci_ad[14]	pr2_enable	lioa[14]	pwm[14]	spi2_miso
GPIO115	pci_ad[15]	pr2_shold	lioa[15]	pwm[15]	spi2_mosi
GPIO116	pci_ad[16]	pr2_data	lio_data[0]	uart1_rx	spi2_cs
GPIO117	pci_ad[17]	pr2_hsync	lio_data[1]	uart1_tx	spi3_clk
GPIO118	pci_ad[18]	pr3_enable	lio_data[2]	uart1_rts	spi3_miso
GPIO119	pci_ad[19]	pr3_shold	lio_data[3]	uart1_cts	spi3_mosi
GPIO120	pci_ad[20]	pr3_data	lio_data[4]	uart1_dsr	spi3_cs
GPIO121	pci_ad[21]	pr4_clk	lio_data[5]	uart1_dtr	spi4_clk
GPIO122	pci_ad[22]	pr4_start	lio_data[6]	uart1_dcd	spi4_miso
GPIO123	pci_ad[23]	pr4_ready	lio_data[7]	uart1_ri	spi4_mosi
GPIO124	pci_ad[24]	pr4_enable	lio_data[8]	gmac0_col	spi4_cs
GPIO125	pci_ad[25]	pr4_shold	lio_data[9]	gmac0_crs	spi5_clk
GPIO126	pci_ad[26]	pr4_data	lio_data[10]	gmac1_col	spi5_miso
GPIO127	pci_ad[27]	pr4_hsync	lio_data[11]	gmac1_crs	spi5_mosi
GPIO128	pci_ad[28]	pr5_enable	lio_data[12]	-	spi5_cs
GPIO129	pci_ad[29]	pr5_shold	lio_data[13]	-	can0_rx
GPIO130	pci_ad[30]	pr5_data	lio_data[14]	-	can0_tx
GPIO131	pci_ad[31]	pr6_clk	lio_data[15]	-	can1_rx
GPIO132	pci_cbe[0]	pr6_start	lioa[16]	-	can1_tx
GPIO133	pci_cbe[1]	pr6_ready	lioa[17]	-	can2_rx
GPIO134	pci_cbe[2]	pr6_enable	lioa[18]	-	can2_tx
GPIO135	pci_cbe[3]	pr6_shold	lioa[19]	-	can3_rx
GPIO136	pci_frame	pr6_data	lioa[20]	-	can3_tx
GPIO137	pci_irdy	pr6_hsync	lioa[21]	-	-
GPIO138	pci_devsel	pr7_enable	lioa[22]	gmac1_mdck	-
GPIO139	pci_trdy	pr7_shold	liocsn[0]	gmac1_mdio	-
GPIO140	pci_stop	pr7_data	liocsn[1]	spi2_clk	-
GPIO141	pci_idsel	pix0_scl	liowrn	spi2_miso	-
GPIO142	pci_par	pix0_sda	liordn	spi2_mosi	-
GPIO143	pci_perr	pix1_scl	-	spi2_cs	sdio1_clk
GPIO144	pci_serr	pix1_sda	-	spi3_clk	sdio1_cmd
GPIO145	pci_req[0]	-	gmac0_ptp_trig	spi3_miso	sdio1_d[0]
GPIO146	pci_req[1]	-	gmac0_ptp_pps	spi3_mosi	sdio1_d[1]
GPIO147	pci_gnt[0]	-	gmac1_ptp_trig	spi3_cs	sdio1_d[2]
GPIO148	pci_gnt[1]	-	gmac1_ptp_pps	-	sdio1_d[3]

GPIO149	sdio_clk	lpc_ad[0]	-	-	-
GPIO150	sdio_cmd	lpc_ad[1]	-	-	-
GPIO151	sdio_d[0]	lpc_ad[2]	gmac0_ptp_trig	-	-
GPIO152	sdio_d[1]	lpc_ad[3]	gmac0_ptp_pps	-	-
GPIO153	sdio_d[2]	lpc_frame	gmac1_ptp_trig	-	-
GPIO154	sdio_d[3]	lpc_serirq	gmac1_ptp_pps	-	-

注：2K0500 芯片共 155 个 GPIO 引脚，全部与其他功能引脚复用，以上芯片复用引脚在上电启动过程，除启动相关功能引脚外（与启动方式关联），其他引脚均默认配置为 GPIO 输入状态，芯片启动完成后可由软件配置选择其他复用关系。GPIO 引脚复用关系通过 GPIO_CFG0 ~ GPIO_CFG19 配置寄存器相应配置位，选择引脚复用功能（其中，0：GPIO；1：第一复用；2：第二复用；3：第三复用；4：第四复用；其他：芯片主功能）。