

**LOONGSON**

龙芯 2K1000 处理器

用户手册

V1.1

2018 年 8 月

龙芯中科技术有限公司

自主决定命运，创新成就未来

北京市海淀区中关村环保科技示范园龙芯产业园 100095  
Loongson Industrial Park, Zhongguancun Environmental Protection Park,  
Haidian District, Beijing 100095, P.R.China



[www.loongson.cn](http://www.loongson.cn)

### 版权声明

本文档版权归龙芯中科技术有限公司所有，并保留一切权利。未经书面许可，任何公司和个人不得将此文档中的任何部分公开、转载或以其他方式散发给第三方。否则，必将追究其法律责任。

### 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

电话(Tel): 010-62546668

传真(Fax): 010-62600826

## 阅读指南

《龙芯 2K1000 处理器用户手册》主要介绍龙芯 2K1000 架构与寄存器描述，对芯片系统架构、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 2K1000 处理器所集成的 LS264 高性能处理器核的相关资料，请参阅《龙芯 LS264 处理器核用户手册》。

## 修订历史

文档更新记录		文档名:	龙芯 2K1000 处理器用户手册
		版本号:	V1.1
		创建人:	芯片研发部
		创建日期:	2017-7
更新历史			
序号.	更新日期	版本号	更新内容
1	2017-7	V1.0	第一版
2	2018-8	V1.1	<ol style="list-style-type: none"><li>修改 DVO09-12 引脚与 UART2 的复用关系</li><li>增加 hpet_int_mode 寄存器, LIO iopf_en 寄存器设置为保留</li><li>增加温度传感器的章节</li><li>增加 hpet1_int、hpet2_int、vpu_int 和 cam_int</li><li>增加 intpol 相关内容, 增加 intpol 和 intedge 可读属性</li><li>更新 LIO 读写时序</li><li>列举已验证 NAND 型号</li><li>列举已验证 SDIO 型号进行说明</li><li>增加 VPU 解码器</li><li>增加 CAMERA 接口控制器</li><li>其他细节的改动</li></ol>

手册信息反馈: [service@loongson.cn](mailto:service@loongson.cn)

也可通过问题反馈网站 <http://bugs.loongnix.org/> 向我司提交芯片产品使用过程中的问题，并获取技术支持。

## 目录

目录 .....	I
图目录 .....	IX
表目录 .....	X
1 概述 .....	1
1.1 体系结构框图 .....	1
1.2 芯片主要功能 .....	2
1.2.1 处理器核 .....	2
1.2.2 内存接口 .....	2
1.2.3 PCIE 接口 .....	3
1.2.4 GPU .....	3
1.2.5 显示控制器 .....	3
1.2.6 SATA 控制器 .....	4
1.2.7 USB2.0 控制器 .....	4
1.2.8 GMAC 控制器 .....	4
1.2.9 VPU 解码器 .....	4
1.2.10 CAMERA 控制器 .....	5
1.2.11 HDA 控制器 .....	5
1.2.12 I2S 控制器 .....	5
1.2.13 NAND 控制器 .....	5
1.2.14 SPI 控制器 .....	5
1.2.15 UART .....	6
1.2.16 I2C 总线 .....	6
1.2.17 PWM .....	6
1.2.18 HPET .....	6
1.2.19 RTC .....	7
1.2.20 Watchdog .....	7
1.2.21 中断控制器 .....	7
1.2.22 CAN .....	7
1.2.23 ACPI 功耗管理 .....	7
1.2.24 GPIO .....	7
1.2.25 加解密模块 .....	7
1.2.26 SDIO 控制器 .....	7
2 引脚定义 .....	9
2.1 约定 .....	9
2.2 DDR3 接口 .....	9
2.3 PCIE 接口 .....	10
2.4 DVO 显示接口 .....	10
2.5 GMAC 接口 .....	12
2.6 SATA 接口 .....	13
2.7 USB 接口 .....	13
2.8 CAMERA 接口 .....	13
2.9 HDA 接口 .....	14
2.10 SPI 接口 .....	15
2.11 I2C 接口 .....	15
2.12 UART 接口 .....	15
2.13 CAN 接口 .....	16
2.14 NAND 接口 .....	17
2.15 SDIO 接口 .....	17
2.16 GPIO .....	17
2.17 PWM .....	18



2.18	PLL 电源接口.....	18
2.19	电源管理接口.....	18
2.20	测试接口.....	19
2.21	EJTAG 接口.....	19
2.22	时钟信号.....	19
2.23	RTC 相关信号.....	19
2.24	系统相关信号.....	20
2.25	外设功能复用表.....	21
3	时钟结构.....	22
3.1	NODE PLL.....	22
3.2	PIX PLL.....	22
3.3	DDR PLL .....	23
3.4	DC PLL.....	23
3.5	内部 PLL 配置方法.....	24
3.5.1	硬件配置 .....	24
3.5.2	软件配置 .....	24
3.6	BOOT 时钟.....	25
3.7	USB 参考时钟 .....	25
3.8	PCIE 参考时钟 .....	25
3.9	SATA 参考时钟输入 .....	25
4	电源管理.....	26
4.1	电源管理模块介绍.....	26
4.2	电源级别.....	26
4.3	控制引脚说明.....	26
5	芯片配置寄存器.....	28
5.1	通用配置寄存器 0.....	34
5.2	通用配置寄存器 1.....	37
5.3	通用配置寄存器 2.....	38
5.4	APBDMA 配置寄存器.....	40
5.5	USB PHY0/1 配置寄存器.....	41
5.6	USB PHY2/3 配置寄存器.....	45
5.7	SATA 配置寄存器.....	46
5.8	NODE PLL 低 64 位配置寄存器.....	47
5.9	NODE PLL 高 64 位配置寄存器.....	48
5.10	DDR PLL 低 64 位配置寄存器 .....	48
5.11	DDR PLL 高 64 位配置寄存器 .....	49
5.12	DC PLL 低 64 位配置寄存器 .....	49
5.13	DC PLL 高 64 位配置寄存器 .....	50
5.14	PIX0 PLL 低 64 位配置寄存器 .....	50
5.15	PIX0 PLL 高 64 位配置寄存器 .....	51
5.16	PIX1 PLL 低 64 位配置寄存器 .....	52
5.17	PIX1 PLL 高 64 位配置寄存器 .....	52
5.18	FREQSCALE 配置寄存器.....	53
5.19	PCIE0 配置寄存器 0 .....	53
5.20	PCIE0 配置寄存器 1 .....	54
5.21	PCIE0 PHY 配置控制寄存器 .....	55
5.22	PCIE1 配置寄存器 0 .....	55
5.23	PCIE1 配置寄存器 1 .....	56
5.24	PCIE1 PHY 配置控制寄存器 .....	57
5.25	DMA 命令控制寄存器(DMA_ORDER) .....	57
5.26	PCICFG2_RECFCF 寄存器.....	58

5.27	PCICFG30_RECFCF 寄存器.....	58
5.28	PCICFG31_RECFCF 寄存器.....	59
5.29	PCICFG40_RECFCF 寄存器.....	60
5.30	PCICFG41_RECFCF 寄存器.....	60
5.31	PCICFG42_RECFCF 寄存器.....	61
5.32	PCICFG5_RECFCF 寄存器.....	61
5.33	PCICFG6_RECFCF 寄存器.....	62
5.34	PCICFG7_RECFCF 寄存器.....	62
5.35	PCICFG8_RECFCF 寄存器.....	63
5.36	PCICFGF_RECFCF 寄存器.....	64
5.37	PCICFG10_RECFCF 寄存器.....	64
5.38	PCICFG11_RECFCF 寄存器.....	65
6	地址空间分配 .....	66
6.1	一级交叉开关.....	67
6.2	二级交叉开关.....	67
6.3	IO 互连网络 .....	70
6.3.1	IO 设备的配置空间.....	71
6.3.2	APB 配置头 .....	78
6.3.3	GMAC0/1 配置头.....	79
6.3.4	USB OTG 配置头.....	79
6.3.5	USBEHCI 配置头.....	80
6.3.6	USB OHCI 配置头 .....	80
6.3.7	GPU 配置头.....	80
6.3.8	DC 配置头 .....	81
6.3.9	HDA 配置头 .....	81
6.3.10	SATA 配置头 .....	82
6.3.11	PCIE 配置头 .....	82
6.3.12	DMA 配置头.....	83
6.3.13	VPU 配置头 .....	83
6.3.14	CAMERA 配置头.....	84
6.3.15	N/A 处理.....	84
6.4	IODMA 请求 .....	84
6.5	APB 设备路由 .....	87
7	处理器核间中断与通信 .....	89
8	温度传感器 .....	91
8.1	实时温度采样.....	91
8.2	高低温中断触发.....	91
8.3	高温自动降频设置.....	92
9	I/O 中断 .....	94
9.1	中断触发类型.....	95
9.2	中断分发模式.....	95
9.3	中断相关寄存器描述.....	96
9.4	GPIO 中断 .....	100
9.5	MSI 中断.....	100
9.6	硬件中断负载均衡功能举例.....	101
10	SPI 控制器.....	103
10.1	访问地址.....	103
10.2	SPI 控制器结构.....	103
10.3	配置寄存器.....	103
10.3.1	控制寄存器(SPCR) .....	104
10.3.2	状态寄存器(SPSR).....	104

<b>10.3.3</b>	数据寄存器(TxFIFO/RxFIFO) .....	104
<b>10.3.4</b>	外部寄存器(SPER).....	104
<b>10.3.5</b>	参数控制寄存器(SFC_PARAM).....	105
<b>10.3.6</b>	片选控制寄存器(SFC_SOFTCS) .....	105
<b>10.3.7</b>	时序控制寄存器(SFC_TIMING).....	105
<b>10.4</b>	接口时序.....	106
<b>10.4.1</b>	SPI 主控制器接口时序.....	106
<b>10.4.2</b>	SPI Flash 访问时序 .....	106
<b>10.5</b>	软件编程指南.....	107
<b>10.5.1</b>	SPI 主控制器的读写操作 .....	107
<b>10.5.2</b>	硬件 SPI Flash 读 .....	108
<b>10.5.3</b>	混合访问 SPI Flash 和 SPI 主控制器 .....	108
<b>11</b>	LocalIO 控制器.....	109
<b>11.1</b>	访问地址及引脚复用 .....	109
<b>11.2</b>	LOCALIO 控制器功能概述 .....	109
<b>12</b>	DDR3 控制器 .....	111
<b>12.1</b>	访问地址.....	111
<b>12.2</b>	DDR3 SDRAM 控制器功能概述 .....	111
<b>12.3</b>	DDR3 SDRAM 读操作协议 .....	111
<b>12.4</b>	DDR3 SDRAM 写操作协议 .....	112
<b>12.5</b>	DDR3 控制器寄存器 .....	112
<b>13</b>	GPIO .....	116
<b>13.1</b>	GPIO 方向控制 .....	116
<b>13.2</b>	GPIO 输出设置 .....	116
<b>13.3</b>	GPIO 输入采样 .....	116
<b>13.4</b>	GPIO 中断使能 .....	116
<b>13.5</b>	GPIO 复用关系 .....	117
<b>14</b>	APB 设备 (Dev 2) .....	119
<b>14.1</b>	内部设备地址路由 .....	119
<b>15</b>	UART 控制器 .....	121
<b>15.1</b>	概述.....	121
<b>15.2</b>	访问地址及引脚复用 .....	121
<b>15.3</b>	控制器结构.....	121
<b>15.4</b>	寄存器描述.....	122
<b>15.4.1</b>	数据寄存器 (DAT) .....	122
<b>15.4.2</b>	中断使能寄存器 (IER) .....	122
<b>15.4.3</b>	中断标识寄存器 (IIR) .....	123
<b>15.4.4</b>	FIFO 控制寄存器 (FCR) .....	123
<b>15.4.5</b>	线路控制寄存器 (LCR) .....	124
<b>15.4.6</b>	MODEM 控制寄存器 (MCR) .....	124
<b>15.4.7</b>	线路状态寄存器 (LSR) .....	125
<b>15.4.8</b>	MODEM 状态寄存器 (MSR) .....	126
<b>15.4.9</b>	分频锁存器 .....	126
<b>16</b>	CAN .....	127
<b>16.1</b>	访问地址及引脚复用 .....	127
<b>16.2</b>	标准模式.....	127
<b>16.2.1</b>	控制寄存器 (CR) .....	128
<b>16.2.2</b>	命令寄存器 (CMR) .....	129
<b>16.2.3</b>	状态寄存器 (SR) .....	129
<b>16.2.4</b>	中断寄存器 (IR) .....	129
<b>16.2.5</b>	验收代码寄存器 (ACR) .....	130

16.2.6	验收屏蔽寄存器 (AMR) .....	130
16.2.7	发送缓冲区列表 .....	130
16.2.8	接收缓冲区列表 .....	131
16.3	扩展模式.....	131
16.3.1	模式寄存器 (MOD) .....	133
16.3.2	命令寄存器 (CMR) .....	134
16.3.3	状态寄存器 (SR) .....	134
16.3.4	中断寄存器 (IR) .....	135
16.3.5	中断使能寄存器 (IER) .....	135
16.3.6	仲裁丢失捕捉寄存器 .....	135
16.3.7	错误警报限制寄存器 (EMLR) .....	137
16.3.8	RX 错误计数寄存器 (RXERR) .....	137
16.3.9	TX 错误计数寄存器 (TXERR) .....	137
16.3.10	验收滤波器.....	137
16.3.11	RX 信息计数寄存器 (RMCR) .....	137
16.4	公共寄存器.....	138
16.4.1	总线定时寄存器 0 (BTR0) .....	138
16.4.2	总线定时寄存器 1 (BTR1) .....	138
16.4.3	输出控制寄存器 (OCR) .....	138
17	I2C 控制器 .....	139
17.1	概述.....	139
17.2	访问地址及引脚复用.....	139
17.3	I2C 控制器结构.....	139
17.4	I2C 控制器寄存器说明.....	140
17.4.1	分频锁存器低字节寄存器 (PRERlo) .....	140
17.4.2	分频锁存器高字节寄存器 (PRERhi) .....	140
17.4.3	控制寄存器 (CTR) .....	140
17.4.4	发送数据寄存器 (TXR) .....	141
17.4.5	接受数据寄存器 (RXR) .....	141
17.4.6	命令控制寄存器 (CR) .....	141
17.4.7	状态寄存器 (SR) .....	142
18	PWM 控制器 .....	143
18.1	概述.....	143
18.2	访问地址及引脚复用.....	143
18.3	寄存器描述.....	143
18.4	功能说明.....	144
18.4.1	脉宽调制功能 .....	144
18.4.2	脉冲测量功能 .....	144
18.4.3	防死区功能 .....	145
19	HPET 控制器.....	146
19.1	概述.....	146
19.2	访问地址.....	146
19.3	寄存器描述.....	146
20	NAND 控制器 .....	150
20.1	NAND 控制器结构描述 .....	150
20.2	访问地址及引脚复用.....	150
20.3	NAND 寄存器配置描述 .....	150
20.3.1	命令寄存器 NAND_CMD (偏移地址 0x00) .....	150
20.3.2	页内偏移地址寄存器 ADDR_C (偏移地址 0x04).....	151
20.3.3	页地址寄存器 ADDR_R (偏移地址 0x08).....	151
20.3.4	时序寄存器 NAND_TIMING (偏移地址 0x0C).....	151

<b>20.3.5</b>	ID 寄存器 ID_L (偏移地址 0x10).....	152
<b>20.3.6</b>	ID 和状态寄存器 STATUS & ID_H (偏移地址 0x14) .....	152
<b>20.3.7</b>	参数配置寄存器 NAND_PARAMETER (偏移地址 0x18) .....	152
<b>20.3.8</b>	操作数量寄存器 NAND_OP_NUM (偏移地址 0x1C).....	152
<b>20.3.9</b>	映射寄存器 CS_RDY_MAP (偏移地址 0x20) .....	153
<b>20.3.10</b>	DMA 读写数据寄存器 DMA_ADDRESS (偏移地址 0x40) .....	153
20.4	NAND ADDR 说明 .....	153
20.5	NAND-FLASH 读写操作举例.....	156
20.6	NAND ECC 说明.....	156
20.7	支持 NAND 型号 .....	158
21	电源管理模块.....	159
21.1	概述.....	159
21.2	访问地址.....	159
21.3	寄存器描述.....	159
22	RTC .....	169
22.1	概述.....	169
22.2	访问地址.....	169
22.3	寄存器描述.....	169
	<b>22.3.1</b> 寄存器地址列表 .....	169
	<b>22.3.2</b> SYS_TOYWRITE0.....	170
	<b>22.3.3</b> SYS_TOYWRITE1.....	170
	<b>22.3.4</b> SYS_TOYREAD0 .....	170
	<b>22.3.5</b> SYS_TOYREAD1 .....	170
	<b>22.3.6</b> SYS_TOYMATCH0/1/2.....	171
	<b>22.3.7</b> SYS_RTCCTRL .....	171
	<b>22.3.8</b> SYS_RTCWRITE .....	172
	<b>22.3.9</b> SYS_RTCREAD .....	172
	<b>22.3.10</b> SYS_RTCMATCH0/1/2 .....	172
23	加解密.....	173
23.1	DES .....	173
	<b>23.1.1</b> DES 功能概述.....	173
	<b>23.1.2</b> DES 访问地址: .....	173
	<b>23.1.3</b> DES 寄存器描述 .....	173
23.2	AES .....	174
	<b>23.2.1</b> AES 功能概述.....	174
	<b>23.2.2</b> AES 访问地址: .....	175
	<b>23.2.3</b> AES 寄存器描述 .....	175
23.3	RSA.....	177
	<b>23.3.1</b> RSA 访问地址: .....	177
23.4	RNG .....	177
	<b>23.4.1</b> RNG 访问地址: .....	177
24	SDIO 控制器 .....	178
24.1	功能概述.....	178
24.2	访问地址及引脚复用 .....	178
24.3	SDIO 协议概述 .....	178
24.4	寄存器描述.....	179
24.5	软件编程指南.....	185
	<b>24.5.1</b> SD Memory 卡软件编程说明 .....	185
	<b>24.5.2</b> SDIO 卡软件编程说明 .....	187
24.6	支持 SDIO 型号 .....	188
25	I2S 控制器 .....	189
25.1	概述.....	189



25.2	访问地址及引脚复用 .....	189
25.3	接口协议 .....	189
25.4	专用寄存器 .....	190
25.5	配置操作 .....	191
26	GMAC 控制器 (Dev 3) .....	193
26.1	访问地址及引脚复用 .....	193
27	OTG 控制器 (Dev 4, Fun 0) .....	194
27.1	概述 .....	194
27.2	访问地址 .....	194
28	USB 控制器 (Dev 4, Fun 1/2) .....	195
28.1	总体概述 .....	195
28.2	访问地址 .....	195
29	图形处理器 (Dev 5) .....	197
29.1	访问地址 .....	197
30	显示控制器 (Dev 6) .....	198
30.1	概述 .....	198
30.2	访问地址及引脚复用 .....	198
31	HDA 控制器 (Dev 7) .....	199
31.1	功能概述 .....	199
31.2	访问地址 .....	199
32	SATA 控制器 (Dev 8) .....	201
32.1	SATA 总体描述 .....	201
32.2	访问地址 .....	201
32.3	SATA 控制器内部寄存器描述 .....	201
33	PCIE 控制器 (Dev 9/A/B/C/D/E) .....	203
33.1	总体结构 .....	203
33.2	访问地址 .....	203
33.3	地址空间划分 .....	204
33.4	软件编程指南 .....	205
	<b>33.4.1 PCIE 控制器使能 .....</b>	205
	<b>33.4.2 PCIE 配置头访问 .....</b>	205
	<b>33.4.3 PCIE 链路建立(Linkup) .....</b>	205
	<b>33.4.4 TYPE1 类型配置访问 .....</b>	205
	<b>33.4.5 PCIE PHY 配置方法 .....</b>	206
	<b>33.5 常用例程 .....</b>	206
34	DMA 控制器 (Dev F) .....	209
34.1	DMA 控制器结构描述 .....	209
34.2	访问地址 .....	209
34.3	DMA 控制器与 APB 设备的交互 .....	210
34.4	DMA 描述符 .....	210
	<b>34.4.1 DMA_ORDER_ADDR_LOW .....</b>	210
	<b>34.4.2 DMA_SADDR .....</b>	210
	<b>34.4.3 DMA_DADDR .....</b>	211
	<b>34.4.4 DMA_LENGTH .....</b>	211
	<b>34.4.5 DMA_STEP_LENGTH .....</b>	211
	<b>34.4.6 DMA_STEP_TIMES .....</b>	212
	<b>34.4.7 DMA_CMD .....</b>	212
	<b>34.4.8 DMA_ORDER_ADDR_HIGH .....</b>	213
	<b>34.4.9 DMA_SADDR_HIGH .....</b>	213
35	VPU 控制器 (Dev 16) .....	214
35.1	访问地址 .....	214
36	CAMERA 接口控制器 (Dev 17) .....	215

36.1 功能概述.....	215
36.2 访问地址.....	215

## 图目录

图 1-1 龙芯 2K1000 结构图 .....	2
图 3-1NODE PLL 结构图 .....	22
图 3-2PIX PLL 结构图 .....	23
图 3-3DDR PLL 结构图 .....	23
图 3-4DC PLL 结构图 .....	24
图 3-5BOOT 时钟结构图 .....	25
图 6-1 二级交叉开关地址路由示意图 .....	67
图 6-2IO 互连结构图 .....	71
图 6-364 位配置访问地址格式 .....	71
图 6-432 位配置访问地址格式 .....	72
图 9-1 龙芯 2K1000 处理器中断路由示意图 .....	94
图 10-1 SPI 控制器结构 .....	103
图 10-2 SPI 主控制器接口时序 .....	106
图 10-3 SPI Flash 标准读时序 .....	106
图 10-4 SPI Flash 快速读时序 .....	107
图 10-5 SPI Flash 双向 I/O 读时序 .....	107
图 11-1LocalIO 读时序 .....	109
图 11-2LocalIO 写时序 .....	110
图 12-1DDR3 SDRAM 读操作协议 .....	112
图 12-2DDR3 SDRAM 写操作协议 .....	112
图 15-1 UART 控制器结构 .....	122
图 17-1 I2C 主控制器结构 .....	140
图 18-1 防死区功能 .....	145
图 24-1 SD 卡多块写操作示意图 .....	179
图 24-2 SD 卡多块读操作示意图 .....	179
图 24-3 SD Memory 卡初始化流程示意图 .....	186
图 25-1 I2S 传输协议 .....	190
图 28-1 USB 主机控制器模块图 .....	195
图 31-1 HDA 模块的整体设计框图 .....	199
图 33-1PCIE 控制器结构 .....	203
图 36-1 CAMERA 模块的整体设计框图 .....	215

## 表目录

表 2-1 信号类型代码 .....	9
表 2-2DDR3SDRAM 控制器接口信号.....	9
表 2-3PCIE 总线信号.....	10
表 2-4DVO 接口信号.....	10
表 2-5 DVO 接口 RGB 对应关系 .....	11
表 2-6DVO0 与 LIO 复用关系.....	11
表 2-7 DVO0 与 UART 复用关系 .....	11
表 2-8GMAC 接口信号.....	12
表 2-9 GMAC1 与 GPIO 复用关系 .....	12
表 2-10SATA 接口信号.....	13
表 2-11 SATA 与 GPIO 复用关系 .....	13
表 2-12USB 接口信号.....	13
表 2-13 CAMERA 接口信号 .....	13
表 2-14CAMERA 与 DVO1 复用关系 .....	14
表 2-15HDA 接口信号 .....	14
表 2-16HDA 与 I2S 复用关系.....	14
表 2-17HDA 与 GPIO 复用关系 .....	14
表 2-18 SPI 接口信号.....	15
表 2-19 I2C 接口信号 .....	15
表 2-20 I2C 与 GPIO 复用关系.....	15
表 2-21 UART 接口信号 .....	15
表 2-22UART 接口复用关系 .....	16
表 2-23 CAN 接口信号 .....	16
表 2-24 CAN 与 GPIO 复用关系 .....	16
表 2-25 NAND 接口信号 .....	17
表 2-26NAND 与 GPIO 复用关系.....	17
表 2-27 SDIO 接口信号 .....	17
表 2-28 SDIO 与 GPIO 复用关系.....	17
表 2-29GPIO 信号 .....	18
表 2-30 PWM 信号 .....	18
表 2-31 PWM 与 GPIO 复用关系 .....	18
表 2-32PLL 电源接口 .....	18
表 2-33 电源管理接口 .....	18

表 2-34 测试接口 .....	19
表 2-35 EJTAG 接口 .....	19
表 2-36 时钟信号 .....	19
表 2-37 时钟信号 .....	19
表 2-38 系统相关信号 .....	20
表 2-39 外设功能复用表 .....	21
表 3-1PLL 硬件配置 .....	24
表 4-1 ACPI 状态说明 .....	26
表 4-2 控制引脚说明 .....	26
表 5-1 芯片配置寄存器列表 .....	28
表 5-2 通用配置寄存器 0 .....	35
表 5-3 通用配置寄存器 1 .....	37
表 5-4 通用配置寄存器 2 .....	38
表 5-5 APBDMA 配置寄存器 .....	40
表 5-6USB 0/1 PHY 配置寄存器 .....	41
表 5-7USB 2/3 PHY 配置寄存器 .....	45
表 5-8 SATA 配置寄存器 .....	46
表 5-9 NODE PLL 低 64 位配置寄存器 .....	47
表 5-10 NODE PLL 高 64 位配置寄存器 .....	48
表 5-11 DDR PLL 低 64 位配置寄存器 .....	48
表 5-12 DDR PLL 高 64 位配置寄存器 .....	49
表 5-13 DC PLL 低 64 位配置寄存器 .....	49
表 5-14 DC PLL 高 64 位配置寄存器 .....	50
表 5-15 PIX0 PLL 低 64 位配置寄存器 .....	50
表 5-16 PIX0 PLL 高 64 位配置寄存器 .....	51
表 5-17 PIX1 PLL 低 64 位配置寄存器 .....	52
表 5-18 PIX1 PLL 高 64 位配置寄存器 .....	52
表 5-19 FRESCALE 配置寄存器 .....	53
表 5-20 PCIE0 配置寄存器 0 .....	53
表 5-21 PCIE0 配置寄存器 1 .....	54
表 5-22 PCIE0 PHY 配置寄存器 .....	55
表 5-23 PCIE1 配置寄存器 0 .....	55
表 5-24 PCIE1 配置寄存器 1 .....	56
表 5-25 PCIE1 PHY 配置寄存器 .....	57

表 5-26 DMA 命令控制寄存器.....	57
表 5-27 PCICFG2_RECFC 配置寄存器.....	58
表 5-28 PCICFG30_RECFC 配置寄存器.....	58
表 5-29 PCICFG31_RECFC 配置寄存器.....	59
表 5-30 PCICFG40_RECFC 配置寄存器.....	60
表 5-31 PCICFG41_RECFC 配置寄存器.....	60
表 5-32 PCICFG42_RECFC 配置寄存器.....	61
表 5-33 PCICFG5_RECFC 配置寄存器.....	61
表 5-34 PCICFG6_RECFC 配置寄存器.....	62
表 5-35 PCICFG7_RECFC 配置寄存器.....	62
表 5-36 PCICFG8_RECFC 配置寄存器.....	63
表 5-37 PCICFGf_RECFC 配置寄存器 .....	64
表 5-38 PCICFG10_RECFC 配置寄存器.....	64
表 5-39 PCICFG11_RECFC 配置寄存器.....	65
表 6-1 芯片地址空间划分 .....	66
表 6-2 一级交叉开关路由规则 .....	67
表 6-3 二级交叉开关处标号与所述模块的对应关系 .....	68
表 6-4 MMAP 字段对应的该空间访问属性 .....	68
表 6-5 二级交叉开关地址窗口转换寄存器表.....	68
表 6-6 各个设备的配置头访问对应关系.....	72
表 6-7 Type0 类型配置头 .....	73
表 6-8 Type0 的配置头寄存器.....	73
表 6-9 Type1 类型配置头 .....	75
表 6-10 Type1 的配置头寄存器 .....	76
表 6-11 APB 总线控制器的配置头缺省值 .....	78
表 6-12 GMAC0 控制器的配置头缺省值 .....	79
表 6-13 USB-OTG 控制器的配置头缺省值 .....	79
表 6-14 USB-EHCI 控制器的配置头缺省值 .....	80
表 6-15 USB-OHCI 控制器的配置头缺省值 .....	80
表 6-16 GPU 控制器的配置头缺省值 .....	81
表 6-17 DC 控制器的配置头缺省值 .....	81
表 6-18 HDA 控制器的配置头缺省值.....	81
表 6-19 SATA 控制器的配置头缺省值.....	82
表 6-20PCIE0 Port0 的配置头缺省值.....	82

表 6-21 DMA 控制器的配置头缺省值 .....	83
表 6-22 VPU 解码器的配置头缺省值 .....	83
表 6-23 CAMERA 控制器的配置头缺省值 .....	84
表 6-24 MMAP 字段对应的该空间访问属性 .....	85
表 6-25 IO 设备 DMA 访存地址转换寄存器表 .....	85
表 6-26 APB 设备地址译码 .....	87
表 7-1 处理器核间中断相关的寄存器及其功能描述 .....	89
表 7-2 0 号处理器核核间中断与通信寄存器列表 .....	89
表 7-3 1 号处理器核的核间中断与通信寄存器列表 .....	89
表 8-1 温度采样寄存器说明 .....	91
表 8-2 高低温中断寄存器说明 .....	92
表 8-3 高温降频控制寄存器说明 .....	93
表 9-1 中断控制寄存器属性 .....	96
表 9-2 中断控制寄存器地址 .....	97
表 9-3 中断路由寄存器的说明 .....	98
表 9-4 中断路由寄存器地址 .....	99
表 9-5 GPIO 中断 .....	100
表 9-6 MSI 中断相关寄存器 .....	100
表 10-1 SPI 控制器地址空间分配 .....	103
表 10-2 SPI 配置寄存器列表 .....	103
表 10-3 SPI 控制寄存器(SPCR) .....	104
表 10-4 SPI 状态寄存器(SPSR) .....	104
表 10-5 SPI 数据寄存器(TxFIFO/RXFIFO) .....	104
表 10-6 SPI 外部寄存器(SPER) .....	105
表 10-7 SPI 分频系数 .....	105
表 10-8 SPI 参数控制寄存器(SFC_PARAM) .....	105
表 10-9 SPI 片选控制寄存器(SFC_SOFTCS) .....	105
表 10-10 SPI 时序控制寄存器(SFC_TIMING) .....	105
表 11-1 LocalIO 地址空间分配 .....	109
表 12-1 内存控制器地址空间分配 .....	111
表 12-2 DDR3 控制器配置寄存器 .....	112
表 13-1 GPIO 配置寄存器 .....	116
表 13-2 GPIO 方向控制 .....	116
表 13-3 GPIO 输出设置 .....	116



表 13-4GPIO 输入采样 .....	116
表 13-5GPIO 中断使能 .....	116
表 13-6 GPIO 复用关系 .....	117
表 14-1 APB 配置访问信息 .....	119
表 14-2 APB 设备地址译码 .....	119
表 15-1 UART 控制器物理地址构成 .....	121
表 15-2 UART 数据寄存器 .....	122
表 15-3 UART 中断使能寄存器 .....	123
表 15-4 UART 中断标识寄存器 .....	123
表 15-5 UART 中断控制功能表 .....	123
表 15-6 UART 的 FIFO 控制寄存器 .....	124
表 15-7 UART 线路控制寄存器 .....	124
表 15-8 UART 的 MODEM 控制寄存器 .....	125
表 15-9 UART 线路状态寄存器 .....	125
表 15-10 UART 的 MODEM 状态寄存器 .....	126
表 15-11 UART 分频锁存器 1 .....	126
表 15-12 UART 分频锁存器 2 .....	126
表 16-1 CAN 内部寄存器物理地址构成 .....	127
表 18-1 PWM 寄存器列表 .....	143
表 18-2 PWM 控制寄存器设置 .....	143
表 25-1 寄存器定义 .....	190
表 25-2 标识寄存器 .....	190
表 25-3 配置寄存器 .....	190
表 25-4 控制寄存器 .....	191
表 34-1DMA ORDER 寄存器 .....	209

# 1 概述

龙芯 2K1000 处理器主要面向于网络应用，兼顾平板应用及工控领域应用。采用 40nm 工艺，片内集成 2 个 GS264 处理器核，主频 1GHz，64 位 DDR3 控制器，以及各种系统 IO 接口。

龙芯 2K1000 的主要特征如下：

- 片内集成两个 64 位的双发射超标量 GS264 处理器核，主频 1GHz
- 片内集成共享的 1MB 二级 Cache
- 片内集成 GPU
- 片内集成显示控制器，支持双路 DVO 显示
- 片内集成 64 位 533MHz 的 DDR3 控制器
- 片内集成 2 个 x4 PCIE2.0 接口；可以拆分为 6 个独立 x1 接口
- 片内集成 1 个 SATA2.0 接口
- 片内集成 4 个 USB2.0 接口
- 片内集成 2 个 RGMII 千兆网接口
- 片内集成 HDA/I2S 接口
- 片内集成 RTC/HPET 模块
- 片内集成 12 个 UART 控制器
- 片内集成 1 个 NAND 控制器
- 片内集成 2 个 CAN 控制器
- 片内集成 1 个 SDIO 控制器
- 片内集成 2 个 I2C 控制器
- 片内集成 1 个 LIO 控制器
- 片内集成 1 个 VPU 解码器
- 片内集成 1 个 CAMERA 接口控制器
- 片内集成 1 个温度传感器
- 集成动态功耗控制模块
- 采用 FC-BGA 封装

## 1.1 体系结构框图

龙芯 2K1000 的结构如图 1-1 所示。一级交叉开关连接两个处理器核、两个二级 Cache 以及 IO 子网络 (Cache 访问路径)。二级交叉开关连接两个二级 Cache、内存控制器、启动模块 (SPI 或者 LIO) 以及 IO 子网络 (Uncache 访问路径)。IO 子网络连接一级交叉开关，

以减少处理器访问延迟。IO 子网络中包括需要 DMA 的模块（PCIE、GMAC、SATA、USB、HDA/I2S、NAND、SDIO、DC、GPU、VPU、CAMERA 和加解密模块）和不需要 DMA 的模块，需要 DMA 的模块可以通过 Cache 或者 Uncache 方式访问内存。

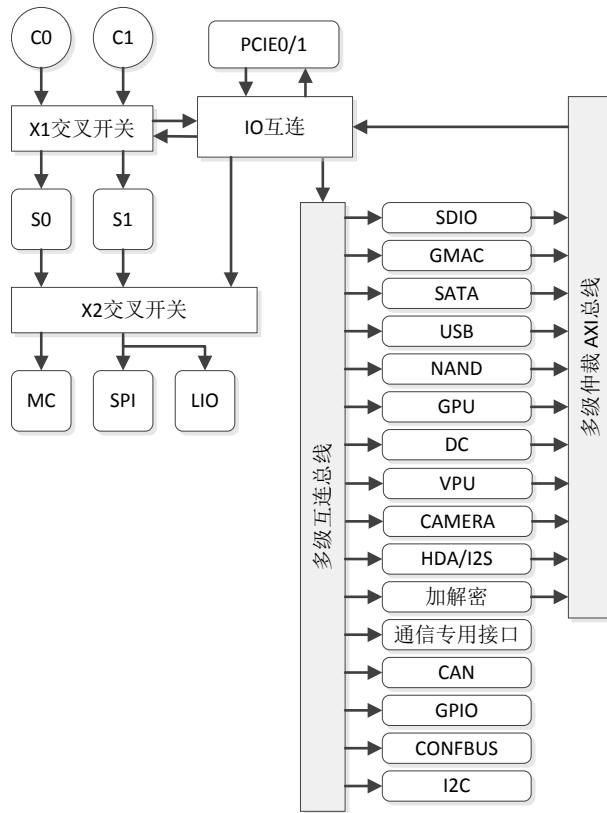


图 1-1 龙芯 2K1000 结构图

## 1.2 芯片主要功能

### 1.2.1 处理器核

- GS264
- MIPS64 R2 体系结构兼容
- 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
- 1M 共享二级 Cache
- 通过目录协议维护 I/O DMA 访问的 Cache 一致性
- EJTAG 支持

### 1.2.2 内存接口

- 64 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 PCIE 接口

- 兼容 PCIE 2.0
- 双独立 X4 接口
- 其中一路 X4 接口可以配置为 4 个 X1 接口
- 其中一路 X4 接口可以配置为 2 个 X1 接口

### 1.2.4 GPU

- 支持 OpenGL ES2.0, OpenGL ES 1.1
- 支持 OpenVG
- 通过 Futuremark 认证
- 动态电源管理
- 支持 BitBLT 和 Stretch BLT
- 矩形填充
- 硬件画线
- 单色字体渲染
- ROP2, ROP3, ROP4
- Alpha 混合
- 32Kx32K 坐标系统
- 90 度旋转
- 透明支持
- YUV 色域空间转换
- 高质量缩放

### 1.2.5 显示控制器

- 双 DVO 输出
- 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
- 低功耗管理

### 1.2.6 SATA 控制器

- 1 个 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
- 低功耗设计

### 1.2.7 USB2.0 控制器

- 4 个独立的 USB2.0 的 HOST 端口
- 其中端口 0 固定为 OTG 工作模式
- 兼容 USB1.1 和 USB2.0
- 内部 EHCI 控制和实现高速传输可达 480Mbps
- 内部 OHCI 控制和实现全速和低速传输
- 低功耗管理

### 1.2.8 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
- 支持网络开机

### 1.2.9 VPU 解码器

- 支持 H264
- 支持 H263
- 支持 SVC
- 支持 MPEG-4
- 支持 MPEG-2
- 支持 MPEG-1
- 支持 Sorenson Spark
- 支持 JPEG
- 支持 RV8、RV9、RV10
- 支持 VP6.0、VP6.1、VP6.2
- 支持 DivX Home Theater Profile Qualified、DivX3、DivX4、DivX5、DivX6

### 1.2.10 CAMERA 控制器

- 兼容 ITU-R BT 601/656 8-bit 模式外部接口（支持同步信号产生的同步或是嵌入式同步）
- 兼容 AMBA 2.0 AHB 接口
- AMBA Interface word aligned memory transfer （32bit 宽度）
- 使用内嵌的 DMA 方式进行存取数据操作
- 8-bit 视频数据输入，输入数据顺序固定，为 U01Y0V01Y1U23Y2V23Y3.....  
(因为这是最为常用的数据格式)
- 独立于图片尺寸的水平和垂直的尺寸设置
- 可编程水平、垂直同步信号极性
- 3 个异步 FIFO，大小分别为 16×32bit (for Y), 8×32bit (for U), 8×32bit (for V)

### 1.2.11 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
- 三路音频输入

### 1.2.12 I2S 控制器

- 支持 master 模式下 I2S 输入
- 支持 master 模式下 I2S 输出
- 支持 8、16、18、20、24、32 位宽
- 支持单声道和立体声道音频数据
- 支持(16、22.05、32、44.1、48)kHz 采样频率
- 支持 DMA 传输模式

### 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
- 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

- 3 个全功能 UART 和流控 TXD,RXD,CTS, RTS, DSR,DTR,DCD, RI
- 最多 12 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

- 兼容 SMBUS (100Kbps)
- 与 PHILIPS I2C 标准相兼容
- 履行双向同步串行协议
- 只实现主设备操作
- 能够支持多主设备的总线
- 总线的时钟频率可编程
- 可以产生开始/停止/应答等操作
- 能够对总线的状态进行探测
- 支持低速和快速模式
- 支持 7 位寻址和 10 位寻址
- 支持时钟延伸和等待状态

### 1.2.17 PWM

- 32 位计数器
- 支持脉冲生成及捕获
- 4 路控制器

### 1.2.18 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

### 1.2.19 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

### 1.2.20 Watchdog

- 32 比特计数器及初始化寄存器
- 低功耗模式暂停功能

### 1.2.21 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

### 1.2.22 CAN

- 两路 CAN 接口
- 复用 GPIO

### 1.2.23 ACPI 功耗管理

- 处理器核动态频率电压调节
- 全芯片时钟门控
- PHY 可关断
- USB/GMAC 可唤醒
- 来电可自动启动

### 1.2.24 GPIO

- 4 位专用 GPIO 引脚，56 位复用 GPIO 引脚
- 其余引脚与其他接口相复用，使用各个接口电压域

### 1.2.25 加解密模块

- AES、DES 算法支持
- RSA 算法支持

### 1.2.26 SDIO 控制器

- 1 路独立 SDIO 控制器
- 兼容 SD Memory 2.0/MMC/SDIO 2.0 协议

- 支持 SDIO 系统启动

## 2 引脚定义

龙芯 2K1000 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。

### 2.1 约定

本章对龙芯 2K1000 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。

- 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 DDR3SDRAM 控制器接口信号

信号名称	类型	描述
DDR_DQ[63:0]	I/O	DDR3 SDRAM 数据总线信号
DDR_DQS <sub>p</sub> [7:0] DDR_DQS <sub>n</sub> [7:0]	DIFF I/O	DDR3 SDRAM 数据选通
DDR_DQM[7:0]	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_CS <sub>n</sub> [3:0]	O	DDR3 SDRAM 片选信号
DDR_CKE[3:0]	O	DDR3 SDRAM 时钟使能信号

信号名称	类型	描述
DDR_CKp[7:0]	DIFF OUT	DDR3 SDRAM 差分时钟输出信号
DDR_CKn[7:0]	O	DDR3 SDRAM ODT 信号
DDR_ODT[3:0]	O	DDR3 SDRAM 复位控制信号
DDR_RESETn	O	DDR3 SDRAM 复位控制信号

## 2.3 PCIE 接口

表 2-3PCIE 总线信号

信号名称	类型	描述
PCIE[1:0]_REFCLKp	DIFF IN	PCIE 参考时钟输入
PCIE[1:0]_REFCLKn	DIFF OUT	PCIE0 参考时钟输出
PCIE0_REFCLKp[3:0]	DIFF OUT	PCIE0 参考时钟输出
PCIE0_REFCLKn[3:0]	DIFF OUT	PCIE1 参考时钟输出
PCIE1_REFCLKp[1:0]	DIFF OUT	PCIE1 参考时钟输出
PCIE1_REFCLKn[1:0]	A	外部参考电阻, 通过 200ohm(+/-1%)电阻连至地
PCIE[1:0]_REFRES	DIFF OUT	PCIE 差分数据输出
PCIE[1:0]_TXp[3:0]	DIFF IN	PCIE 差分数据输入
PCIE[1:0]_TXn[3:0]	I	PCIE0 插卡检测, bit3-1 任意一位置低时 PCIE0 工作在 4X1 模式, 否则工作在 1X4 模式
PCIE0_PRSNT[3:0]	I	PCIE1 插卡检测, bit1 置低时 PCIE1 工作在 2X1 模式, 否则工作在 1X4 模式
PCIE_RSTn	O	PCIE 复位

## 2.4 DVO 显示接口

表 2-4DVO 接口信号

信号名称	类型	描述
DVO[1:0]_CLKp	O	DVO 时钟输出
DVO[1:0]_CLKn	O	DVO 时钟输出, 与 DVO*_CLKp 相差 180° , 非差分关系
DVO[1:0]_HSYNC	O	DVO 水平同步
DVO[1:0]_VSYNC	O	DVO 垂直同步
DVO[1:0]_DE	O	DVO 数据有效
DVO[1:0]_D[23:0]	O	DVO 显示数据 [23:16]为 R 数据 [15:08]为 G 数据 [07:00]为 B 数据

DVO 接口数据信号与 RGB 对应关系如下:

表 2-5 DVO 接口 RGB 对应关系

DVO 接口信号	24 位模式	18 位模式
DVO_D0	B0	
DVO_D1	B1	
DVO_D2	B2	B0
DVO_D3	B3	B1
DVO_D4	B4	B2
DVO_D5	B5	B3
DVO_D6	B6	B4
DVO_D7	B7	B5
DVO_D8	G0	
DVO_D9	G1	
DVO_D10	G2	G0
DVO_D11	G3	G1
DVO_D12	G4	G2
DVO_D13	G5	G3
DVO_D14	G6	G4
DVO_D15	G7	G5
DVO_D16	R0	
DVO_D17	R1	
DVO_D18	R2	R0
DVO_D19	R3	R1
DVO_D20	R4	R2
DVO_D21	R5	R3
DVO_D22	R6	R4
DVO_D23	R7	R5

DVO0 接口与 LIO 以及 UART 有复用关系，如表 2-6 和表 2-7 DVO0 与 UART 复用关系所示。复用配置请参考表 5-3 通用配置寄存器 1 和表 5-4 通用配置寄存器 2。

表 2-6DVO0 与 LIO 复用关系

信号名称	复用名称	复用类型	复用信号描述
DVO0_CLKp	LIO_RDn	O	LIO RDn 输出
DVO0_CLKn	LIO_WRn	O	LIO WRn 输出
DVO0_HSYNC	LIO_DEN	O	LIO 数据使能
DVO0_VSYNC	LIO_DIR	O	LIO 方向控制，0 代表读，1 代表写
DVO0_DE	LIO_ADLOCK	O	LIO 地址/数据选择信号
DVO0_D[15:0]	LIO_AD[15:0]	I/O	LIO 双向 AD 信号
DVO0_D[22:16]	LIO_A[6:0]	O	LIO 地址低位
DVO0_D23	LIO_CSn	O	LIO 片选信号

表 2-7 DVO0 与 UART 复用关系

信号名称	复用名称	复用类型	复用信号描述
DVO0_HSYNC	UART1_TXD	O	串口数据输出
DVO0_VSYNC	UART1_RXD	I	串口数据输入
DVO0_DE	UART1_RTS	O	串口数据传输请求
DVO0_D00	UART1_DTR	O	串口初始化完成
DVO0_D01	UART1_RI	I	外部 MODEM 探测到振铃信号
DVO0_D02	UART1_CTS	I	设备接受数据就绪
DVO0_D03	UART1_DSR	I	设备初始化完成
DVO0_D04	UART1_DCD	I	外部 MODEM 探测到载波信号
DVO0_D05	UART2_TXD	O	串口数据输出
DVO0_D06	UART2_RXD	I	串口数据输入
DVO0_D07	UART2_RTS	O	串口数据传输请求
DVO0_D08	UART2_DTR	O	串口初始化完成
DVO0_D09	UART2_RI	I	外部 MODEM 探测到振铃信号
DVO0_D11	UART2_CTS	I	设备接受数据就绪
DVO0_D12	UART2_DSR	I	设备初始化完成
DVO0_D13	UART2_DCD	I	外部 MODEM 探测到载波信号

DVO1 接口与CAMERA接口有复用关系，参考和 2.8 节。

## 2.5 GMAC 接口

表 2-8GMAC 接口信号

信号名称	类型	描述
GMAC[1:0]_TXCK	O	RGMII 发送时钟
GMAC[1:0]_TCTL	O	RGMII 发送控制
GMAC[1:0]_TXD[3:0]	O	RGMII 发送数据
GMAC[1:0]_RXCK	I	RGMII 接收时钟
GMAC[1:0]_RCTL	I	RGMII 接收控制
GMAC[1:0]_RXD[3:0]	I	RGMII 接收数据
GMAC[1:0]_MDCK	O	SMA 接口时钟
GMAC[1:0]_MDIO	I/O	SMA 接口数据

GMAC1 接口与 GPIO 有复用关系，如下表所示。复用配置请参考表 5-2 通用配置寄存器 0。

表 2-9 GMAC1 与 GPIO 复用关系

信号名称	复用名称	复用类型	复用信号描述
GMAC1_TXCK	-	-	-
GMAC1_TCTL	GPIO13	I/O	通用输入输出 13
GMAC1_RXD[3:0]	GPIO[12:9]	I/O	通用输入输出 12-9
GMAC1_RXCK	-	-	-

信号名称	复用名称	复用类型	复用信号描述
GMAC1_RCTL	GPIO8	I/O	通用输入输出 8
GMAC1_RXD[3:0]	GPIO[7:4]	I/O	通用输入输出 7-4
GMAC1_MDCK	-	-	-
GMAC1_MDIO	-	-	-

## 2.6 SATA 接口

表 2-10SATA 接口信号

信号名称	类型	描述
SATA_REFCLKp	I	差分 100MHz 参考时钟输入(内部有备份时钟, 通过软件控制)
SATA_REFCLKn	A	外部参考电阻, 通过 200ohm(+/-1%)电阻连至地
SATA_TXp	DIFF OUT	SATA 差分数据输出
SATA_Txn	DIFF IN	SATA 差分数据输入
SATA_RXp	DIFF IN	SATA 差分数据输入
SATA_RXn	O	SATA 工作状态, 低表示有数据传输
SATA_LEDn	O	SATA 工作状态, 低表示有数据传输

SATA 接口的 SATA\_LEDn 与 GPIO 有复用关系, 如下表所示。复用配置请参考表 5-2 通用配置寄存器 0。

表 2-11 SATA 与 GPIO 复用关系

信号名称	复用名称	复用类型	复用信号描述
SATA_LEDn	GPIO14	I/O	通用输入输出 14

## 2.7 USB 接口

表 2-12USB 接口信号

信号名称	类型	描述
USB_XI USB_XO	I/O	必须在 USB_XO 上接晶振, USB_XI 保留不用
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+
USB[3:0]_DM	I/O	USB D-
USB[3:1]_OC	I	USB 过流检测输入, 需注意该信号为高有效
USB0_OC	O	OTG DRVVBUS 输出
USB0_ID	I	USB0 OTG ID 输入
USB0_VBUS	A	USB0 OTG VBUS 输入

## 2.8 CAMERA 接口

表 2-13 CAMERA 接口信号

信号名称	类型	描述
CAM_PCLK	I	像素时钟，被摄像头的处理器驱动
CAM_HSYNC	I	水平同步，被摄像头的处理器驱动
CAM_VSYNC	I	帧同步，被摄像头的处理器驱动
CAM_DATA[7:0]	I	像素数据，被摄像头的处理器驱动
CAM_CLOCK	O	XCLK 被摄像头控制器驱动，用于摄像头模块

注：控制信号都是单向的，只支持 camera 提供时钟和同步信号的模式

表 2-14CAMERA 与 DVO1 复用关系

信号名称	复用名称	复用类型	复用信号描述
CAM_PCLK	DVO1_CKN	I	DVO1 负时钟输出
CAM_HSYNC	DVO1_HSYNC	I	DVO1 水平同步信号
CAM_VSYNC	DVO1_VSYNC	I	DVO1 垂直同步信号
CAM_DATA[7:0]	DVO1_D[7:0]	I	DVO1 数据输出信号
CAM_CLOCK	DVO1_CKP	O	DVO1 正时钟输出

## 2.9 HDA 接口

表 2-15HDA 接口信号

信号名称	类型	描述
HDA_BITCLK	O	HDA BITCLK 输出
HDA_SDI0	I	HDA 数据输入，连接第一个 codec
HDA_SDI1	I	HDA 数据输入，连接第二个 codec
HDA_SDI2	I	HDA 数据输入，连接第三个 codec
HDA_SDO	O	HDA 数据输出
HDA_SYNC	O	HDA 同步
HDA_RESETn	O	HDA 复位

HDA 接口与、I2S 以及 GPIO 复用，具体复用关系如下。复用配置请参考表 5-2 通用配置寄存器 0。

表 2-16HDA 与 I2S 复用关系

信号名称	复用名称	复用类型	复用信号描述
HDA_BITCLK	I2S_BCLK	O	I2S bit 时钟
HDA_SDI0	I2S_DI	I	I2S 数据输入
HDA_SDI1	-	-	-
HDA_SDI2	-	-	-
HDA_SDO	I2S_DO	O	I2S 数据输出
HDA_SYNC	I2S_MCLK	O	I2S MCLK
HDA_RESETn	I2S_LR	O	I2S 左右声道选择

表 2-17HDA 与 GPIO 复用关系

信号名称	复用名称	复用类型	复用信号描述
HDA_BITCLK	GPIO24	I/O	通用输入输出 24
HDA_SDI0	GPIO28	I/O	通用输入输出 28
HDA_SDI1	GPIO29	I/O	通用输入输出 29
HDA_SDI2	GPIO30	I/O	通用输入输出 30
HDA_SDO	GPIO27	I/O	通用输入输出 27
HDA_SYNC	GPIO25	I/O	通用输入输出 25
HDA_RESETn	GPIO26	I/O	通用输入输出 26

## 2.10 SPI 接口

表 2-18 SPI 接口信号

信号名称	类型	描述
SPI_SCK	O	SPI 时钟输出
SPI_CS <sub>n</sub> 0	O	SPI 片选 0
SPI_CS <sub>n</sub> 1	O	SPI 片选 1
SPI_CS <sub>n</sub> 2	O	SPI 片选 2
SPI_CS <sub>n</sub> 3	O	SPI 片选 3
SPI_SDO	O	SPI 数据输出
SPI_SDI	I	SPI 数据输入

## 2.11 I2C 接口

表 2-19 I2C 接口信号

信号名称	类型	描述
I2C0_SCL	O	I2C0 时钟
I2C0_SDA	I/O	I2C0 数据
I2C1_SCL	O	I2C1 时钟
I2C1_SDA	I/O	I2C1 数据

I2C 与 GPIO 有复用，复用关系见下表。复用配置请参考表 5-2 通用配置寄存器 0。

表 2-20 I2C 与 GPIO 复用关系

信号名称	复用名称	复用类型	复用信号描述
I2C0_SCL	GPIO16	I/O	通用输入输出 16
I2C0_SDA	GPIO17	I/O	通用输入输出 17
I2C1_SCL	GPIO18	I/O	通用输入输出 18
I2C1_SDA	GPIO19	I/O	通用输入输出 19

## 2.12 UART 接口

表 2-21 UART 接口信号

信号名称	类型	描述

信号名称	类型	描述
UART_TXD	O	串口数据输出
UART_RXD	I	串口数据输入
UART_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_CTS	I	设备接受数据就绪
UART_DSR	I	设备初始化完成
UART_DCD	I	外部 MODEM 探测到载波信号

UART 与 DVO 接口有复用关系，具体见 2.4 节。

2K1000 仅有一个独立的全功能串口，该串口通过设置可以工作在 2x4 和 4x2 模式，各种模式的管脚对应关系如下。其它引脚复用的 UART 接口的内部复用关系也如下表所示。

表 2-22UART 接口复用关系

1x8	2x4	4x2
TXD0(O)	<b>TXD0(O)</b>	<b>TXD0(O)</b>
RTS0(O)	<b>RTS0(O)</b>	<b>TXD5(O)</b>
DTR0(O)	<i>TXD3(O)</i>	<i>TXD3(O)</i>
RXD0(I)	<b>RXD0(I)</b>	<b>RXD0(I)</b>
CTS0(I)	<b>CTS0(I)</b>	<b>RXD5(I)</b>
DSR0(I)	<i>RXD3(I)</i>	<i>RXD3(I)</i>
DCD0(I)	<i>CTS3(I)</i>	<b>RXD4(I)</b>
RI0(I)	<i>RTS3(O)</i>	<b>TXD4(O)</b>

## 2.13 CAN 接口

表 2-23 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

CAN 接口与 GPIO 有复用，如下表所示。复用配置请参考表 5-2 通用配置寄存器 0。

表 2-24 CAN 与 GPIO 复用关系

信号名称	复用名称	复用类型	复用信号描述
CAN0_RX	GPIO32	I/O	通用输入输出 32
CAN0_TX	GPIO33	I/O	通用输入输出 33
CAN1_RX	GPIO34	I/O	通用输入输出 34
CAN1_TX	GPIO35	I/O	通用输入输出 35

## 2.14 NAND 接口

表 2-25 NAND 接口信号

信号名称	类型	描述
NAND_CEn[3:0]	O	NAND 片选 3-0
NAND_CLE	O	NAND 命令锁存
NAND_ALE	I	NAND 地址锁存
NAND_WRn	O	NAND 写信号
NAND_RDn	I	NAND 读信号
NAND_RDYn[3:0]	I	NAND 准备好输入 3-0
NAND_D[7:0]	I/O	NAND 命令/地址/数据线

NAND 与 GPIO 有复用，复用关系见下表。复用配置请参考表 5-2 通用配置寄存器 0。

表 2-26 NAND 与 GPIO 复用关系

信号名称	复用名称	复用类型	复用信号描述
NAND_CEn[3:0]	GPIO[47:44]	I/O	通用输入输出 47-44
NAND_CLE	GPIO48	I/O	通用输入输出 48
NAND_ALE	GPIO49	I/O	通用输入输出 49
NAND_WRn	GPIO50	I/O	通用输入输出 50
NAND_RDn	GPIO51	I/O	通用输入输出 51
NAND_RDYn[3:0]	GPIO[55:52]	I/O	通用输入输出 55-52
NAND_D[7:0]	GPIO[63:56]	I/O	通用输入输出 63-56

## 2.15 SDIO 接口

表 2-27 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
SDIO_DATA[3:0]	I/O	SDIO 数据信号

SDIO 与 GPIO 有复用，复用关系见下表。复用配置请参考表 5-2 通用配置寄存器 0。

表 2-28 SDIO 与 GPIO 复用关系

信号名称	复用名称	复用类型	复用信号描述
SDIO_CLK	GPIO41	I/O	通用输入输出 41
SDIO_CMD	GPIO40	I/O	通用输入输出 40
SDIO_DATA[3:0]	GPIO[36:39]	I/O	通用输入输出 36-39

## 2.16 GPIO

下表仅列出专用的 4 个 GPIO 引脚信号，其他 GPIO 为复用信号，可参考其他信号定义。

默认情况下所有与 GPIO 复用的引脚为 GPIO 功能，且都为输入状态。

表 2-29GPIO 信号

信号名称	类型	描述
GPIO00	I/O	通用输入输出
GPIO01	I/O	通用输入输出
GPIO02	I/O	通用输入输出
GPIO03	I/O	通用输入输出

## 2.17 PWM

表 2-30 PWM 信号

信号名称	类型	描述
PWM[3:0]	O	PWM 输出

PWM 与 GPIO 有复用，复用关系如下。复用配置请参考表 5-2 通用配置寄存器 0。

表 2-31 PWM 与 GPIO 复用关系

信号名称	复用名称	复用类型	复用信号描述
PWM[3:0]	GPIO[23:20]	I/O	通用输入输出 23-20

## 2.18 PLL 电源接口

表 2-32PLL 电源接口

信号名称	类型	描述
PLL_NODE_VDD	P	NODE PLL 电源
PLL_SOC_VDD	P	SOC PLL 电源
PLL_DDR_VDD	P	DDR PLL 电源
PLL_PIX0_VDD	P	PIXEL0 PLL 电源
PLL_PIX1_VDD	P	PIXEL1 PLL 电源
PLL_NODE_VSS	P	NODE PLL 地
PLL_SOC_VSS	P	SOC PLL 地
PLL_DDR_VSS	P	DDR PLL 地
PLL_PIX0_VSS	P	PIXEL0 PLL 地
PLL_PIX1_VSS	P	PIXEL1 PLL 地

## 2.19 电源管理接口

表 2-33 电源管理接口

信号名称	类型	描述
ACPI_SYSRSTn	I	系统复位
ACPI_RINGn	I	振铃唤醒
ACPI_WAKEn	I	PCIE 唤醒
ACPI_LID	I	屏盖状态
ACPI_PWRTYPE	I	供电来源
ACPI_BATLOWn	I	电源电量低

信号名称	类型	描述
ACPI_SUSSTATn	O	低功耗状态
ACPI_S3n	O	S3 状态
ACPI_S4n	O	S4 状态
ACPI_S5n	O	S5 状态
ACPI_VID[5:0]	O	调压控制
ACPI_PLTRSTn	O	平台复位
ACPI_SLPLANn	O	网络电源控制
ACPI_PWRBTNn	I	电源开关
ACPI_PWROK	I	电源有效
ACPI_EN	I	ACPI 使能

## 2.20 测试接口

表 2-34 测试接口

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

## 2.21 EJTAG 接口

表 2-35 EJTAG 接口

信号名称	类型	描述
EJTAG_SEL	I	JTAG 选择(1: JTAG; 0: EJTAG)
EJTAG_TCK	I	JTAG 时钟
EJTAG_TDI	I	JTAG 数据输入
EJTAG_TMS	I	JTAG 模式
EJTAG_TRST	I	JTAG 复位
EJTAG_TDO	O	JTAG 数据输出

## 2.22 时钟信号

表 2-36 时钟信号

信号名称	类型	描述
SYS_SYSCLK	I	100MHz 参考时钟
SYS_TESTCLK	I	测试时钟输入， 默认不用连接

## 2.23 RTC 相关信号

表 2-37 时钟信号

信号名称	类型	描述
RTC_RSMRSTn	I	RSM 域复位, 要求在 RSM 域电源稳定 1ms 后拉高,

		在 RSM 域电源降至 95% 及以下时立即拉低。
RTC_RSTn	I	RTC 域复位,建议在 RTC 电源稳定 10ms 后再解除复位。
RTC_XI	I/O	32.768KHz 晶体输入
RTC_XO	I/O	32.768KHz 晶体输出

## 2.24 系统相关信号

表 2-38 系统相关信号

信号名称	类型	描述
SYS_CLKSEL[1:0]	I	PLL 时钟配置输入 00=低频模式 01=高频模式 10=软件模式(DFT) 11=bypass 模式
SYS_BOOTSEL[1:0]	I	启动选择输入 00=LIO 01=SPI(DFT) 10=SDIO 11=NAND
SYS_USBCLKMODE[1:0]	I	USB 时钟输入配置输入 00=保留 01=保留 10=one 12MHz clock input 11=use sysclk(DFT)
SYS_PCIECLKSEL	I	PCIE 参考时钟选择输入 {SYS_PCIECLKSEL, SYS_PCIECLKDIV}: 00=内部 100MHz 时钟 01=保留 10=外部 100MHz 时钟 11=外部 200MHz 时钟
SYS_PCIECLKDIV	I	PCIE 参考时钟选择输入 {SYS_PCIECLKSEL, SYS_PCIECLKDIV}: 00=内部 100MHz 时钟 01=保留 10=外部 100MHz 时钟 11=外部 200MHz 时钟
SYS_NANDRSRD	I	NAND ECC 功能使能输入, 1=enable 0=disable(DFT)
SYS_NANDTYPE[1:0]	I	启动 NAND 类型选择 00=512Mb(page 512B) 01=1Gb(page 2KB)

		10=16Gb(page 4KB) 11=128Gb(page 8KB)
--	--	---

## 2.25 外设功能复用表

模块层次的功能复用关系如下表所示：

表 2-39 外设功能复用表

功能 0	功能 1	功能 2	功能 3	功能 4	功能 5
DDR3					
PCIE					
SATA	GPIO(1)				
USB					
GMAC0					
GMAC1	GPIO(14)				
DVO0		Local Bus	UART1(8)	UART1(4)	UART1(2)
					UART8(2)
			UART2(8)	UART6(4)	UART6(2)
					UART7(2)
				UART2(4)	UART2(2)
					UART11(2)
				UART9(4)	UART9(2)
					UART10(2)
DVO1			CAMERA		
CAN	GPIO(4)				
HDA	GPIO(7)			I2S	
SPI					
RTC					
I2C	GPIO(4)				
			UART0(8)	UART0(4)	UART0(2)
					UART5(2)
				UART3(4)	UART3(2)
					UART4(2)
NAND	GPIO(16)				
EJTAG		JTAG			
	GPIO(4)				
PWM	GPIO(4)				
SDIO	GPIO(6)				
ACPI					

# 3 时钟结构

龙芯 2K1000 由一个 100MHz 时钟作为参考时钟，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；
- ✓ 两个 PIX PLL 各产生一个像素时钟供 DC 使用，以便支持双路独立显示；
- ✓ 一个 DC PLL 同时产生 DC 控制器的内部时钟和 GMAC 控制器时钟，进一步产生 APB, SATA 以及 USB 的时钟；
- ✓ 一个 DDR PLL 同时产生 DDR、GPU 以及 HDA 的时钟；

除了内部的 PLL 之外，对于 SATA、PCIE、USB 这几个采用 PHY 自己产生时钟的模块，也使用外部输入的同一个参考时钟进行了参考时钟通路设计。

还有一个启动时钟，直接使用 100MHz 参考时钟通过分频得到。

## 3.1 NODE PLL

node clock 的产生结构图如图 3-1 所示，

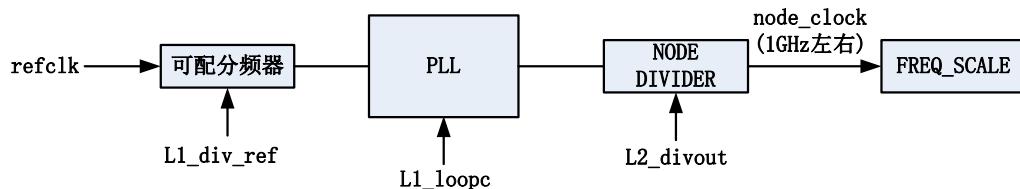


图 3-1 NODE PLL 结构图

输出时钟频率的计算方式如下：

$$\text{node\_clock} = \text{refclk} / \text{L1\_div\_ref} * \text{L1\_loopc} / \text{L2\_divout};$$

node\_clock 的工作频率在 1GHz 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/L1\_div\_ref 在 20~40MHz 范围内，PLL 倍频值 refclk/L1\_div\_ref\*L1\_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 5.18 节。

## 3.2 PIX PLL

pix pll 结构基本与 node pll 结构相同，但是不包含 freq\_scale 模块。2K1000 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 100-250MHz。

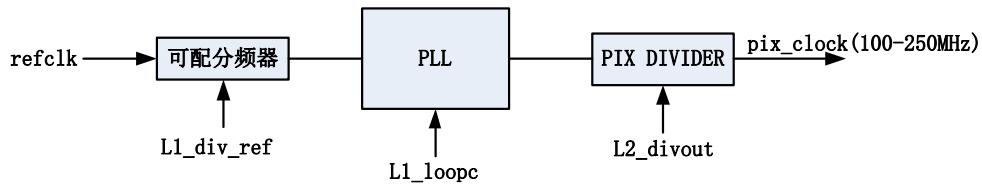


图 3-2PIX PLL 结构图

### 3.3 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 400-700MHz
- ✓ gpu\_clock 用于 GPU 模块，频率范围 300-500MHz
- ✓ 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L2\_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。

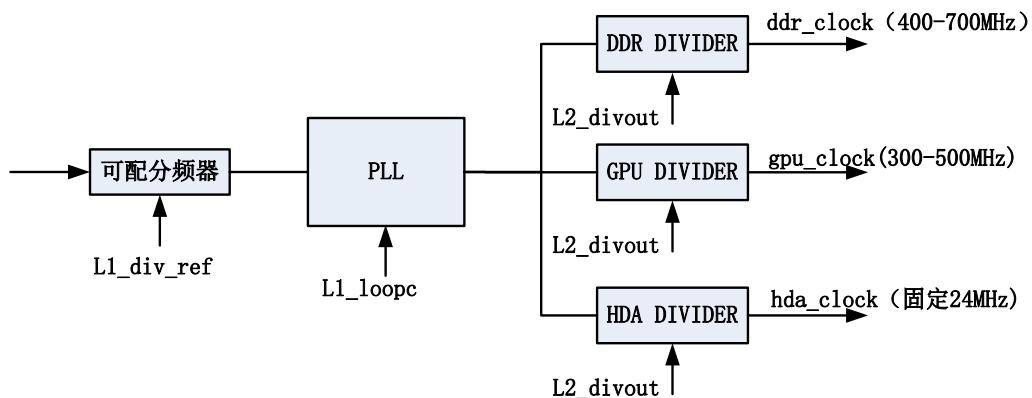


图 3-3DDR PLL 结构图

### 3.4 DC PLL

DC PLL输出两个时钟：

- ✓ dc\_clock 用于 DC 控制器，频率可设为 200MHz 左右，需保证其频率大于像素时钟。该时钟也作为 VPU 模块的主时钟。
- ✓ gmac\_clock 用于 GMAC 控制器，频率固定 125MHz。

APB、SATA 以及 USB 控制器的时钟由 125MHz gmac 时钟分频产生，所以这三个模块时钟频率最高为 125MHz，通过 freq\_scale 模块可以进一步调整分频系数。具体分频方法请参考 5.18 节。

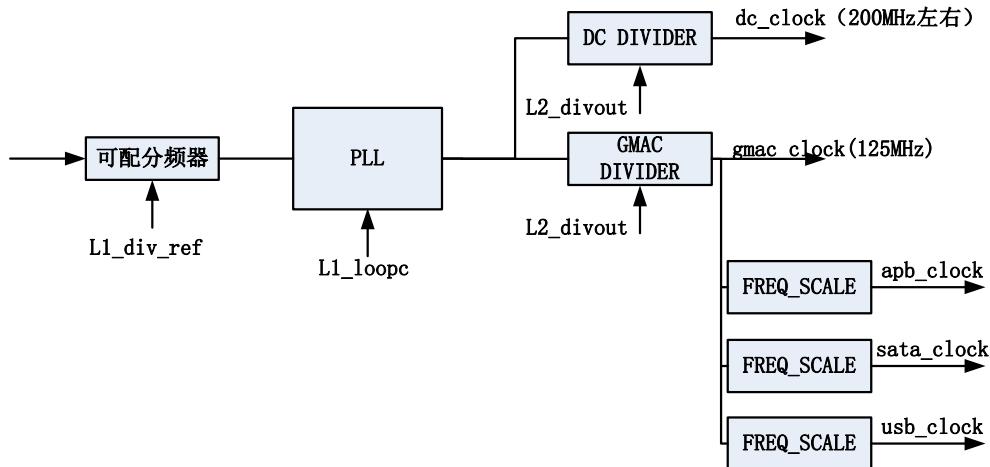


图 3-4DC PLL 结构图

### 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 SYS\_CLKSEL[1:0]的设置来区分。

#### 3.5.1 硬件配置

具体如下表所示。

表 3-1PLL 硬件配置

SYS_CLKSEL	00(硬件低频)	01 (硬件高频)	10	11
NODE	0.875G	1G	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		
GPU	320M	400M		
HDA	24M	24M		
DC	200M	400M		
PIX0	100M	200M		
PIX1	100M	200M		
GMAC	不可硬件配置	不可硬件配置		
SATA	85M	150M		
USB	85M	150M		
APB	85M	150M		

#### 3.5.2 软件配置

当 SYS\_CLKSEL 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外部参考时钟频率，即所有 PLL 输出都是 SYS\_SYSCLK，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。

具体的配置寄存器说明请参考第 5 章。

## 3.6 BOOT 时钟

BOOT 时钟通过 100MHz 参考时钟分频得到，用于 SPI 和 LIO 两个启动模块，同时该时钟也作为 CAMERA 模块的主时钟。



图 3-5 BOOT 时钟结构图

## 3.7 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)

## 3.8 PCIE 参考时钟

PCIE 采用两个独立的 x4 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 2 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 1 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

# 4 电源管理

本章对电源管理进行简单介绍，具体寄存器及使用方法请参考第 21 章。

## 4.1 电源管理模块介绍

- 龙芯 2K1000 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(USB, GMAC，电源开关等)。
- 动态性能功耗控制，支持处理器核 DVFS 控制，支持动态关闭媒体解码协处理器电源。
- 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，处理器子状态可由 Cx 或 Px 决定，媒体处理器由状态 Dx 决定。
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_SYSRSTn	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）	RTC2V5
RTC_RSTn	I	电池更换后，重启 RTC 逻辑	RTC2V5
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_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, 只有唤醒电路上电	RSM3V3
ACPI_SLPLANn	O	以太网 PHY 休眠指示信号	RSM3V3
ACPI_VID[5:0]	O	电压值指示信号	RSM3V3

# 5 芯片配置寄存器

龙芯 2K1000 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 5-1 芯片配置寄存器列表

地址	名称	描述
0x1fe10000	CPU_WIN0_BASE	cache 通路二级窗口 0 的基地址
0x1fe10008	CPU_WIN1_BASE	cache 通路二级窗口 1 的基地址
0x1fe10010	CPU_WIN2_BASE	cache 通路二级窗口 2 的基地址
0x1fe10018	CPU_WIN3_BASE	cache 通路二级窗口 3 的基地址
0x1fe10020	CPU_WIN4_BASE	cache 通路二级窗口 4 的基地址
0x1fe10028	CPU_WIN5_BASE	cache 通路二级窗口 5 的基地址
0x1fe10030	CPU_WIN6_BASE	cache 通路二级窗口 6 的基地址
0x1fe10038	CPU_WIN7_BASE	cache 通路二级窗口 7 的基地址
0x1fe10040	CPU_WIN0_MASK	cache 通路二级窗口 0 的掩码
0x1fe10048	CPU_WIN1_MASK	cache 通路二级窗口 1 的掩码
0x1fe10050	CPU_WIN2_MASK	cache 通路二级窗口 2 的掩码
0x1fe10058	CPU_WIN3_MASK	cache 通路二级窗口 3 的掩码
0x1fe10060	CPU_WIN4_MASK	cache 通路二级窗口 4 的掩码
0x1fe10068	CPU_WIN5_MASK	cache 通路二级窗口 5 的掩码
0x1fe10070	CPU_WIN6_MASK	cache 通路二级窗口 6 的掩码
0x1fe10078	CPU_WIN7_MASK	cache 通路二级窗口 7 的掩码
0x1fe10080	CPU_WIN0_MMAP	cache 通路二级窗口 0 的新基地址
0x1fe10088	CPU_WIN1_MMAP	cache 通路二级窗口 1 的新基地址
0x1fe10090	CPU_WIN2_MMAP	cache 通路二级窗口 2 的新基地址
0x1fe10098	CPU_WIN3_MMAP	cache 通路二级窗口 3 的新基地址
0x1fe100a0	CPU_WIN4_MMAP	cache 通路二级窗口 4 的新基地址
0x1fe100a8	CPU_WIN5_MMAP	cache 通路二级窗口 5 的新基地址
0x1fe100b0	CPU_WIN6_MMAP	cache 通路二级窗口 6 的新基地址
0x1fe100b8	CPU_WIN7_MMAP	cache 通路二级窗口 7 的新基地址
0x1fe10100	PCI_WIN0_BASE	uncache 通路二级窗口 0 的基地址
0x1fe10108	PCI_WIN1_BASE	uncache 通路二级窗口 1 的基地址
0x1fe10110	PCI_WIN2_BASE	uncache 通路二级窗口 2 的基地址
0x1fe10118	PCI_WIN3_BASE	uncache 通路二级窗口 3 的基地址
0x1fe10120	PCI_WIN4_BASE	uncache 通路二级窗口 4 的基地址
0x1fe10128	PCI_WIN5_BASE	uncache 通路二级窗口 5 的基地址
0x1fe10130	PCI_WIN6_BASE	uncache 通路二级窗口 6 的基地址
0x1fe10138	PCI_WIN7_BASE	uncache 通路二级窗口 7 的基地址
0x1fe10140	PCI_WIN0_MASK	uncache 通路二级窗口 0 的掩码

地址	名称	描述
0x1fe10148	PCI_WIN1_MASK	uncache 通路二级窗口 1 的掩码
0x1fe10150	PCI_WIN2_MASK	uncache 通路二级窗口 2 的掩码
0x1fe10158	PCI_WIN3_MASK	uncache 通路二级窗口 3 的掩码
0x1fe10160	PCI_WIN4_MASK	uncache 通路二级窗口 4 的掩码
0x1fe10168	PCI_WIN5_MASK	uncache 通路二级窗口 5 的掩码
0x1fe10170	PCI_WIN6_MASK	uncache 通路二级窗口 6 的掩码
0x1fe10178	PCI_WIN7_MASK	uncache 通路二级窗口 7 的掩码
0x1fe10180	PCI_WIN0_MMAP	uncache 通路二级窗口 0 的新地址
0x1fe10188	PCI_WIN1_MMAP	uncache 通路二级窗口 1 的新地址
0x1fe10190	PCI_WIN2_MMAP	uncache 通路二级窗口 2 的新地址
0x1fe10198	PCI_WIN3_MMAP	uncache 通路二级窗口 3 的新地址
0x1fe101a0	PCI_WIN4_MMAP	uncache 通路二级窗口 4 的新地址
0x1fe101a8	PCI_WIN5_MMAP	uncache 通路二级窗口 5 的新地址
0x1fe101b0	PCI_WIN6_MMAP	uncache 通路二级窗口 6 的新地址
0x1fe101b8	PCI_WIN7_MMAP	uncache 通路二级窗口 7 的新地址
0x1fe10480	PLL_SYS_0	系统的 PLL 低 64 位配置
0x1fe10488	PLL_SYS_1	系统的 PLL 高 64 位配置
0x1fe10490	PLL_DDR_0	内存控制器的 PLL 低 64 位配置
0x1fe10498	PLL_DDR_1	内存控制器的 PLL 高 64 位配置
0x1fe104a0	PLL_DC_0	DC 的 PLL 低 64 位配置
0x1fe104a8	PLL_DC_1	DC 的 PLL 高 64 位配置
0x1fe104b0	PLL_PIX0_0	PIX0 的 PLL 低 64 位配置
0x1fe104b8	PLL_PIX0_1	PIX0 的 PLL 高 64 位配置
0x1fe104c0	PLL_PIX1_0	PIX1 的 PLL 低 64 位配置
0x1fe104c8	PLL_PIX1_1	PIX1 的 PLL 高 64 位配置
0x1fe104d0	FREQSCALE	各设备的分频控制
0x1fe10500	GPIO0_OEN	GPIO 的低 64 位输出使能
0x1fe10508	GPIO1_OEN	保留
0x1fe10510	GPIO0_O	GPIO 的低 64 位输出值
0x1fe10518	GPIO1_O	保留
0x1fe10520	GPIO0_I	GPIO 的低 64 位输入值
0x1fe10528	GPIO1_I	保留
0x1fe10530	GPIO0_INT	GPIO 的低 64 位中断
0x1fe10538	GPIO1_INT	保留
0x1fe10580	PCIE0_REG0	PCIE0 的配置寄存器 0
0x1fe10588	PCIE0_REG1	PCIE0 的配置寄存器 1
0x1fe10590	PCIE0PHY	PCIE0 的 PHY 配置寄存器

地址	名称	描述
0x1fe105a0	PCIE1_REG0	PCIE1 的配置寄存器 0
0x1fe105a8	PCIE1_REG1	PCIE1 的配置寄存器 1
0x1fe105b0	PCIE1PHY	PCIE1 的 PHY 配置寄存器
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
0x1fe10c40	CONFDMA4	DMA4 控制器的配置寄存器
0x1fe11000	CORE0_IPISR	0 号处理器核的 IPI_Status 寄存器
0x1fe11004	CORE0_IPIEN	0 号处理器核的 IPI_Enable 寄存器
0x1fe11008	CORE0_IPISET	0 号处理器核的 IPI_Set 寄存器
0x1fe1100c	CORE0_IPICLR	0 号处理器核的 IPI_Clear 寄存器
0x1fe11020	CORE0_BUF0	0 号处理器核的 IPI_MailBox0 寄存器
0x1fe11028	CORE0_BUF1	0 号处理器核的 IPI_MailBox1 寄存器
0x1fe11030	CORE0_BUF2	0 号处理器核的 IPI_MailBox2 寄存器
0x1fe11038	CORE0_BUF3	0 号处理器核的 IPI_MailBox3 寄存器
0x1fe11040	CORE0_INTISR0	路由给 CORE0 的低 32 位中断状态
0x1fe11048	CORE0_INTISR1	路由给 CORE0 的高 32 位中断状态
0x1fe11100	CORE1_IPISR	1 号处理器核的 IPI_Status 寄存器
0x1fe11104	CORE1_IPIEN	1 号处理器核的 IPI_Enable 寄存器
0x1fe11108	CORE1_IPISET	1 号处理器核的 IPI_Set 寄存器
0x1fe1110c	CORE1_IPICLR	1 号处理器核的 IPI_Clear 寄存器
0x1fe11120	CORE1_BUF0	1 号处理器核的 IPI_MailBox0 寄存器
0x1fe11128	CORE1_BUF1	1 号处理器核的 IPI_MailBox1 寄存器
0x1fe11130	CORE1_BUF2	1 号处理器核的 IPI_MailBox2 寄存器
0x1fe11138	CORE1_BUF3	1 号处理器核的 IPI_MailBox3 寄存器
0x1fe11140	CORE1_INTISR0	路由给 CORE1 的低 32 位中断状态
0x1fe11148	CORE1_INTISR1	路由给 CORE1 的高 32 位中断状态
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

地址	名称	描述
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	保留
0x1fe11434	INTEDGE_0	低 32 位触发方式寄存器(1: 脉冲触发; 0: 电平触发)
0x1fe11438	BOUNCE_0	低 32 位中断分发模式控制，与 auto 合用。
0x1fe1143c	AUTO_0	低 32 位中断分发模式控制，与 bounce 合用。
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	保留
0x1fe11474	INTEDGE_1	高 32 位触发方式寄存器(1: 脉冲触发; 0: 电平触发)
0x1fe11478	BOUNCE_1	高 32 位中断分发模式控制，与 auto 合用。
0x1fe1147c	AUTO_1	高 32 位中断分发模式控制，与 bounce 合用。
0x1fe114a0	INT_MSI_0	保留
0x1fe114a8	INT_MSI_EN_0	保留
0x1fe114b0	INT_MSI_ADDR_0	MSI 地址 0
0x1fe114b4	INT_MSI_TRIGGER_EN_0	32 位 MSI 中断使能，每位对应一个写入 MSI 地址 0 的中断
0x1fe114e0	INT_MSI_1	保留
0x1fe114e8	INT_MSI_EN_1	保留
0x1fe114f0	INT_MSI_ADDR_1	MSI 地址 1
0x1fe114f4	INT_MSI_TRIGGER_EN_1	32 位 MSI 中断使能，每位对应一个写入 MSI 地址 1 的中断
0x1fe11500	Thsens_int_ctrl_Hi	温度传感器高温中断控制寄存器
0x1fe11508	Thsens_int_ctrl_Lo	温度传感器低温中断控制寄存器
0x1fe11510	Thsens_int_status/clr	温度传感器中断状态寄存器
0x1fe11520	Thsens_scale_hi	高温自动降频控制寄存器
0x1fe11528	Thsens_scale_lo	保留
0x1fe12400	XBAR_WIN4_BASE0	IO 设备（除 PCIE）DMA 访问窗口 0 基地址
0x1fe12408	XBAR_WIN4_BASE1	IO 设备（除 PCIE）DMA 访问窗口 1 基地址
0x1fe12410	XBAR_WIN4_BASE2	IO 设备（除 PCIE）DMA 访问窗口 2 基地址
0x1fe12418	XBAR_WIN4_BASE3	IO 设备（除 PCIE）DMA 访问窗口 3 基地址
0x1fe12420	XBAR_WIN4_BASE4	IO 设备（除 PCIE）DMA 访问窗口 4 基地址
0x1fe12428	XBAR_WIN4_BASE5	IO 设备（除 PCIE）DMA 访问窗口 5 基地址
0x1fe12430	XBAR_WIN4_BASE6	IO 设备（除 PCIE）DMA 访问窗口 6 基地址

地址	名称	描述
0x1fe12438	XBAR_WIN4_BASE7	IO 设备（除 PCIE）DMA 访问窗口 7 基址
0x1fe12440	XBAR_WIN4_MASK0	IO 设备（除 PCIE）DMA 访问窗口 0 掩码
0x1fe12448	XBAR_WIN4_MASK1	IO 设备（除 PCIE）DMA 访问窗口 1 掩码
0x1fe12450	XBAR_WIN4_MASK2	IO 设备（除 PCIE）DMA 访问窗口 2 掩码
0x1fe12458	XBAR_WIN4_MASK3	IO 设备（除 PCIE）DMA 访问窗口 3 掩码
0x1fe12460	XBAR_WIN4_MASK4	IO 设备（除 PCIE）DMA 访问窗口 4 掩码
0x1fe12468	XBAR_WIN4_MASK5	IO 设备（除 PCIE）DMA 访问窗口 5 掩码
0x1fe12470	XBAR_WIN4_MASK6	IO 设备（除 PCIE）DMA 访问窗口 6 掩码
0x1fe12478	XBAR_WIN4_MASK7	IO 设备（除 PCIE）DMA 访问窗口 7 掩码
0x1fe12480	XBAR_WIN4_MMAP0	IO 设备（除 PCIE）DMA 访问窗口 0 新基址
0x1fe12488	XBAR_WIN4_MMAP1	IO 设备（除 PCIE）DMA 访问窗口 1 新基址
0x1fe12490	XBAR_WIN4_MMAP2	IO 设备（除 PCIE）DMA 访问窗口 2 新基址
0x1fe12498	XBAR_WIN4_MMAP3	IO 设备（除 PCIE）DMA 访问窗口 3 新基址
0x1fe124a0	XBAR_WIN4_MMAP4	IO 设备（除 PCIE）DMA 访问窗口 4 新基址
0x1fe124a8	XBAR_WIN4_MMAP5	IO 设备（除 PCIE）DMA 访问窗口 5 新基址
0x1fe124b0	XBAR_WIN4_MMAP6	IO 设备（除 PCIE）DMA 访问窗口 6 新基址
0x1fe124b8	XBAR_WIN4_MMAP7	IO 设备（除 PCIE）DMA 访问窗口 7 新基址
0x1fe12500	XBAR_WIN5_BASE0	PCIE 设备 DMA 访问窗口 0 基址
0x1fe12508	XBAR_WIN5_BASE1	PCIE 设备 DMA 访问窗口 1 基址
0x1fe12510	XBAR_WIN5_BASE2	PCIE 设备 DMA 访问窗口 2 基址
0x1fe12518	XBAR_WIN5_BASE3	PCIE 设备 DMA 访问窗口 3 基址
0x1fe12520	XBAR_WIN5_BASE4	PCIE 设备 DMA 访问窗口 4 基址
0x1fe12528	XBAR_WIN5_BASE5	PCIE 设备 DMA 访问窗口 5 基址
0x1fe12530	XBAR_WIN5_BASE6	PCIE 设备 DMA 访问窗口 6 基址
0x1fe12538	XBAR_WIN5_BASE7	PCIE 设备 DMA 访问窗口 7 基址
0x1fe12540	XBAR_WIN5_MASK0	PCIE 设备 DMA 访问窗口 0 掩码
0x1fe12548	XBAR_WIN5_MASK1	PCIE 设备 DMA 访问窗口 1 掩码
0x1fe12550	XBAR_WIN5_MASK2	PCIE 设备 DMA 访问窗口 2 掩码
0x1fe12558	XBAR_WIN5_MASK3	PCIE 设备 DMA 访问窗口 3 掩码
0x1fe12560	XBAR_WIN5_MASK4	PCIE 设备 DMA 访问窗口 4 掩码
0x1fe12568	XBAR_WIN5_MASK5	PCIE 设备 DMA 访问窗口 5 掩码
0x1fe12570	XBAR_WIN5_MASK6	PCIE 设备 DMA 访问窗口 6 掩码
0x1fe12578	XBAR_WIN5_MASK7	PCIE 设备 DMA 访问窗口 7 掩码
0x1fe12580	XBAR_WIN5_MMAP0	PCIE 设备 DMA 访问窗口 0 新基址
0x1fe12588	XBAR_WIN5_MMAP1	PCIE 设备 DMA 访问窗口 1 新基址
0x1fe12590	XBAR_WIN5_MMAP2	PCIE 设备 DMA 访问窗口 2 新基址
0x1fe12598	XBAR_WIN5_MMAP3	PCIE 设备 DMA 访问窗口 3 新基址
0x1fe125a0	XBAR_WIN5_MMAP4	PCIE 设备 DMA 访问窗口 4 新基址
0x1fe125a8	XBAR_WIN5_MMAP5	PCIE 设备 DMA 访问窗口 5 新基址
0x1fe125b0	XBAR_WIN5_MMAP6	PCIE 设备 DMA 访问窗口 6 新基址

地址	名称	描述
0x1fe125b8	XBAR_WIN5_MMAP7	PCIE 设备 DMA 访问窗口 7 新基址
0x1fe12810	CONFAML	QOS 控制使能寄存器
0x1fe12818	CONFAML0	AML0 的读操作带宽、延迟配置寄存器
0x1fe12820	CONFAML1	AML0 的写操作带宽、延迟配置寄存器
0x1fe12828	CONFAML2	AML1 的读操作带宽、延迟配置寄存器
0x1fe12830	CONFAML3	AML1 的写操作带宽、延迟配置寄存器
0x1fe12838	CONFAML4	AML2 的读操作带宽、延迟配置寄存器
0x1fe12840	CONFAML5	AML2 的写操作带宽、延迟配置寄存器
0x1fe12848	CONFAML6	AML3 的读操作带宽、延迟配置寄存器
0x1fe12850	CONFAML7	AML3 的写操作带宽、延迟配置寄存器
0x1fe12858	CONFAML8	AML4 的读操作带宽、延迟配置寄存器
0x1fe12860	CONFAML9	AML4 的写操作带宽、延迟配置寄存器
0x1fe12868	CONFAML10	AML5 的读操作带宽、延迟配置寄存器
0x1fe12870	CONFAML11	AML5 的写操作带宽、延迟配置寄存器
0x1fe12878	CONFAML12	AML6 的读操作带宽、延迟配置寄存器
0x1fe12880	CONFAML13	AML6 的写操作带宽、延迟配置寄存器
0x1fe12888	CONFAML14	AML7 的读操作带宽、延迟配置寄存器
0x1fe12890	CONFAML15	AML7 的写操作带宽、延迟配置寄存器
0x1fe12898	CONFAML16	AML0-3 的延迟屏蔽配置寄存器
0x1fe128a0	CONFAML17	AML4-7 的延迟屏蔽配置寄存器
0x1fe128a8	CONFAML18	AML0-3 的带宽屏蔽配置寄存器
0x1fe128b0	CONFAML19	AML4-7 的带宽屏蔽配置寄存器
0x1fe128b8	CONFAML20	AML0 的读操作实时带宽监测寄存器
0x1fe128c0	CONFAML21	AML0 的写操作实时带宽监测寄存器
0x1fe128c8	CONFAML22	AML1 的读操作实时带宽监测寄存器
0x1fe128d0	CONFAML23	AML1 的写操作实时带宽监测寄存器
0x1fe128d8	CONFAML24	AML2 的读操作实时带宽监测寄存器
0x1fe128e0	CONFAML25	AML2 的写操作实时带宽监测寄存器
0x1fe128e8	CONFAML26	AML3 的读操作实时带宽监测寄存器
0x1fe128f0	CONFAML27	AML3 的写操作实时带宽监测寄存器
0x1fe128f8	CONFAML28	AML4 的读操作实时带宽监测寄存器
0x1fe12900	CONFAML29	AML4 的写操作实时带宽监测寄存器
0x1fe12908	CONFAML30	AML5 的读操作实时带宽监测寄存器
0x1fe12910	CONFAML31	AML5 的写操作实时带宽监测寄存器
0x1fe12918	CONFAML32	AML6 的读操作实时带宽监测寄存器
0x1fe12920	CONFAML33	AML6 的写操作实时带宽监测寄存器
0x1fe12928	CONFAML34	AML7 的读操作实时带宽监测寄存器
0x1fe12930	CONFAML35	AML7 的写操作实时带宽监测寄存器
0x1fe13000	PCICFG_HEADER2	APB 设备的配置头寄存器(一般采用配置头访问形式，不建议使用配置寄存器的形式访问)

地址	名称	描述
0x1fe13040	PCICFG_HEADER30	GMAC0 设备的配置头寄存器(一般采用配置头访问形式, 不建议使用配置寄存器的形式访问)
0x1fe13080	PCICFG_HEADER31	GMAC1 设备的配置头寄存器(一般采用配置头访问形式, 不建议使用配置寄存器的形式访问)
0x1fe130c0	PCICFG_HEADER40	USB-OTG 设备的配置头寄存器(一般采用配置头访问形式, 不建议使用配置寄存器的形式访问)
0x1fe13100	PCICFG_HEADER41	USB-EHCI 设备的配置头寄存器(一般采用配置头访问形式, 不建议使用配置寄存器的形式访问)
0x1fe13140	PCICFG_HEADER42	USB-OHCI 设备的配置头寄存器(一般采用配置头访问形式, 不建议使用配置寄存器的形式访问)
0x1fe13180	PCICFG_HEADER5	GPU 设备的配置头寄存器(一般采用配置头访问形式, 不建议使用配置寄存器的形式访问)
0x1fe131c0	PCICFG_HEADER6	DC 设备的配置头寄存器(一般采用配置头访问形式, 不建议使用配置寄存器的形式访问)
0x1fe13200	PCICFG_HEADER7	HDA 设备的配置头寄存器(一般采用配置头访问形式, 不建议使用配置寄存器的形式访问)
0x1fe13240	PCICFG_HEADER8	SATA 设备的配置头寄存器(一般采用配置头访问形式, 不建议使用配置寄存器的形式访问)
0x1fe13280	N/A_HEADER	无效设备的配置头寄存器(一般采用配置头访问形式, 不建议使用配置寄存器的形式访问)
0x1fe132c0	PCICFG_HEADERf	DMA 控制器的配置头寄存器(一般采用配置头访问形式, 不建议使用配置寄存器的形式访问)
0x1fe13300	PCICFG_HEADER10	VPU 设备的配置头寄存器(一般采用配置头访问形式, 不建议使用配置寄存器的形式访问)
0x1fe13340	PCICFG_HEADER11	CAMERA 控制器的配置头寄存器(一般采用配置头访问形式, 不建议使用配置寄存器的形式访问)
0x1fe13800	PCICFG2_RECFCFG	APB 设备配置头空间的重配置使能寄存器
0x1fe13808	PCICFG30_RECFCFG	GMAC0 设备配置头空间的重配置使能寄存器
0x1fe13810	PCICFG31_RECFCFG	GMAC1 设备配置头空间的重配置使能寄存器
0x1fe13818	PCICFG40_RECFCFG	USB-OTG 设备配置头空间的重配置使能寄存器
0x1fe13820	PCICFG41_RECFCFG	USB-EHCI 设备配置头空间的重配置使能寄存器
0x1fe13828	PCICFG42_RECFCFG	USB-OHCI 设备配置头空间的重配置使能寄存器
0x1fe13830	PCICFG5_RECFCFG	GPU 设备配置头空间的重配置使能寄存器
0x1fe13838	PCICFG6_RECFCFG	DC 设备配置头空间的重配置使能寄存器
0x1fe13840	PCICFG7_RECFCFG	HDA 设备配置头空间的重配置使能寄存器
0x1fe13848	PCICFG8_RECFCFG	SATA 设备配置头空间的重配置使能寄存器
0x1fe13850	PCICFGf_RECFCFG	DMA 控制器配置头空间的重配置使能寄存器
0x1fe13858	PCICFG10_RECFCFG	VPU 设备配置头空间的重配置使能寄存器
0x1fe13860	PCICFG11_RECFCFG	CAMERA 控制器配置头空间的重配置使能寄存器
0x1fe13ff8	CHIP_ID	芯片版本号

上表中关于二级交叉开关以及 DMA 访问窗口相关的配置寄存器将在第 6 章介绍，中断相关寄存器将在第 7 章介绍，这里不再赘述。下面详细介绍其他的配置寄存器。

## 5.1 通用配置寄存器 0

通用配置寄存器 0，包括对管脚复用的控制，以及 HDA、USB、PCIE 的一致性、内存控制器、RTC 控制器及 LIO 控制器的配置等。

地址: 0x1fe10420

表 5-2 通用配置寄存器 0

位域	名称	访问	缺省值	描述
63	LIO big_mem	RW	0x0	LIO big_mem 模式控制
62	LIO iopf_en	RW	0x0	保留
61	LIO io_width_16	RW	0x1	IO 数据位宽: 0: 8 位 1: 16 位
60:56	LIO io_count_init_i	RW	0x1f	IO 数据读取延迟 (boot 时钟周期数)
55	LIO rom_width16	RW	0x1	Rom 数据位宽: 0: 8 位 1: 16 位
54:50	LIO rom_count_init_i	RW	0x1f	ROM 数据读取延迟 (boot 时钟周期数)
49:48	LIO clock_period_i	RW	0x0	内部等待计数器步长设置: 0: 步长为 1 1: 步长为 4 2: 步长为 2 3: 步长为 1
47	rtc_restart	RW	0x0	内部振荡器重启控制
46:44	rtc_ds	RW	0x0	内部振荡器接口控制信号
43:42	Reserved	RO	0x0	保留
41	mc_default_reg	RW	0x1	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。
40	mc_disable_reg	RW	0x0	DDR 配置空间关闭, 高效 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
39:37	Reserved	RO	0x0	保留
36	usbehci64_en	RW	0x0	USB 端口的 EHCI 控制器 64 位地址模式 0: 32 位地址模式 1: 64 位地址模式
35	hpet_int_mode	RW	0x0	HPET 中断模式 0: 正常模式, 即 timer0/1/2 的中断都映射到中断引脚 21 1: 中断分开模式, timer0/1/2 的中断分别映射到中断引脚 21/38/39
34	pcie_coherent	RW	0x1	PCIE 设备的 DMA 请求类别: 0: 非一致性请求 1: 为一致性请求

位域	名称	访问	缺省值	描述
33	usb_coherent	RW	0x1	USB 设备的 DMA 请求类别: 0: 非一致性请求 1: 为一致性请求
32	hda_coherent	RW	0x1	HDA 设备的 DMA 请求类别: 0: 非一致性请求 1: 为一致性请求
31:21	Reserved	RO	0x0	保留
20	sdio_sel	RW	0x0	SDIO 管脚复用控制: 0: 管脚为 GPIO 1: 管脚为 SDIO
19:18				
17:16	can_sel	RW	0x0	CAN 管脚复用控制: 当专用通信接口为 0 时 0: 管脚为 GPIO 1: 管脚为 CAN 否则, 管脚为专用通信接口
15:12	pwm_sel	RW	0x0	PWM 管脚复用控制: 0: 管脚为 GPIO 1: 管脚为 PWM
11:10	i2c_sel	RW	0x0	I2C 管脚复用控制: 0: 管脚为 GPIO 1: 管脚为 I2C
9	nand_sel	RW	0x0	NAND 管脚复用控制: 0: 管脚为 GPIO 1: 管脚为 NAND
8	sata_sel	RW	0x0	SATA 管脚复用控制: 0: 管脚为 GPIO 1: 管脚为 SATA
7	lio_sel	RW	0x0	当 dvo0_sel 为 0 时, LIO 管脚复用控制: 0: 管脚为 GPIO 1: 管脚为 LIO 注: dvo0_sel、lio_sel 和 uart1/2_sel 三者只能有一个为 1
6	i2s_sel	RW	0x0	表示 I2S 管脚复用控制: 0: 管脚为 HDA 或 GPIO 1: 管脚为 I2S 注: hda_sel、和 i2s_sel 只能有一个为 1
5	-	RW	0x0	保留
4	hda_sel	RW	0x0	HDA 管脚复用控制: 0: 管脚为 I2S 或 GPIO 1: 管脚为 HDA 注: hda_sel 和 i2s_sel 只能有一个为 1

位域	名称	访问	缺省值	描述
3	gmac1_sel	RW	0x1	GMAC1 管脚复用控制: 0: 管脚为 GPIO 1: 管脚为 GMAC1
2	gmac1_sdb_flowctrl	RW	0x0	GMAC1 的边带流控制, 当 GMAC 控制器的 EFC 使能时, 可以让 MAC 产生 Pause Control Frame 控制 Rx FIFO 的流量: 0: 关闭该功能 1: 使能该功能
1	gmac0_sdb_flowctrl	RW	0x0	GMAC0 的边带流控制, 当 GMAC 控制器的 EFC 使能时, 可以让 MAC 产生 Pause Control Frame 控制 Rx FIFO 的流量: 0: 关闭该功能 1: 使能该功能
0	gmac_coherence_en	RW	0x1	GMAC 设备的 DMA 请求类别: 0: 非一致性请求 1: 为一致性请求

## 5.2 通用配置寄存器 1

通用配置寄存器 1, 包括对管脚复用的控制以及流控制等。

地址: 0x1fe10428

表 5-3 通用配置寄存器 1

位域	名称	访问	缺省值	描述
63:60	delay_hda	RW	0x0	HDA 访存通路写命令堵塞时间
59:56	delay_pcie0	RW	0x4	PCIE0 访存通路写命令堵塞时间
55:52	delay_usb	RW	0x4	USB 访存通路写命令堵塞时间
51:48	delay_sata	RW	0x4	SATA 访存通路写命令堵塞时间
47:44	delay_dc	RW	0x2	DC 访存通路写命令堵塞时间
43:40	delay_gpu	RW	0x2	GPU 访存通路写命令堵塞时间
39:36	delay_dma	RW	0x4	DMA 访存写命令堵塞时间
35:32	delay_gmac	RW	0x4	GMAC 访存通路写命令堵塞时间
31:28	awmon_channel	RW	0x0	访存写命令通路 monitor 通道选择
27:26	awmon_select	RW	0x0	访存写命令通路 monitor 类型选择
25	awmon_clear	RW	0x0	访存写命令通路 monitor 清空
24	awmon_start	RW	0x0	访存写命令通路 monitor 开始工作
23:20	usb_flush_idle	RW	0xf	设置清空 write buffer 前空闲周期数
19	usb_prefetch	RW	0x1	使能读预取
18	usb_flush_wr	RW	0x0	设置写命令发出后是否清空 read buffer
17	usb_stop_waw	RW	0x0	是否允许在上一个写完成前发出写命令
16	usb_stop_raw	RW	0x1	是否允许在上一个写完成前发出读命令
15:14	Reserved	RO	0x0	保留

位域	名称	访问	缺省值	描述
13	uart2_sel	RW	0x0	当 dvo0_sel 为 0 时, UART2 管脚复用控制: 0: 管脚为 GPIO 1: 管脚为 UART2 注: dvo0_sel、lio_sel 和 uart1/2_sel 三者只能有一个为 1
12	uart1_sel	RW	0x0	当 dvo0_sel 为 0 时, UART1 管脚复用控制: 0: 管脚为 GPIO 1: 管脚为 UART1 注: dvo0_sel、lio_sel 和 uart1/2_sel 三者只能有一个为 1
11:8	uart2_enable	RW	0x1	UART2 对应的 UART 控制器: Bit0: 对应 uart2, 保留 Bit1: 为 1 对应 uart9, 为 0 对应 uart0 的 8 线模式 Bit2: 为 1 对应 uart10, 为 0 对应 uart1 的 8 线模式 Bit3: 为 1 对应 uart11, 为 0 对应 uart1 的 8 线模式
7:4	uart1_enable	RW	0x1	UART1 对应的 UART 控制器: Bit0: 对应 uart1, 保留 Bit1: 为 1 对应 uart6, 为 0 对应 uart0 的 8 线模式 Bit2: 为 1 对应 uart7, 为 0 对应 uart1 的 8 线模式 Bit3: 为 1 对应 uart8, 为 0 对应 uart1 的 8 线模式
3:0	uart0_enable	RW	0x1	UART0 对应的 UART 控制器: Bit0: 对应 uart0, 保留 Bit1: 为 1 对应 uart3, 为 0 对应 uart0 的 8 线模式 Bit2: 为 1 对应 uart4, 为 0 对应 uart1 的 8 线模式 Bit3: 为 1 对应 uart5, 为 0 对应 uart1 的 8 线模式

### 5.3 通用配置寄存器 2

通用配置寄存器 2, 包括对 GPU 的软复位, 管脚复用及 PCIE 设备的使能等。

地址: 0x1fe10430

表 5-4 通用配置寄存器 2

位域	名称	访问	缺省值	描述
63:32	Reserved	RO	0x0	保留

位域	名称	访问	缺省值	描述
31:28	cam_flush_idle	R/W	0xf	cam 接口空闲写入周期数
27	cam_prefetch	R/W	0x1	cam 接口预取
26	cam_flush_wr	R/W	0x0	cam 接口读请求刷出写请求
25	cam_stop_waw	R/W	0x0	cam 接口停止写后写
24	cam_stop_raw	R/W	0x1	cam 接口停止写后读
23:21	Reserved	RO	0x0	保留
20	vpu_disable	RW	0x0	VPU 模块使能信号, 1 关闭, 0 打开
19	cam_disable	RW	0x0	CAMERA 模块使能信号, 1 关闭, 0 打开
18	cam_soft_resetn	RW	0x0	CAMERA 模块软件复位, 低有效。
17	pcie1_enable	RW	0x0	PCIE1 控制器使能信号, 高有效。当不使能时软件扫描不到该 PCIE 桥。
16	pcie0_enable	RW	0x0	PCIE0 控制器使能信号, 高有效。当不使能时软件扫描不到该 PCIE 桥。
15:8	iodma_spare_rd	RW	0x0	iodma 读操作最大数设置
7	cam_coherent	RW	0x1	CAMERA 设备的 DMA 请求类别: 0: 非一致性请求 1: 为一致性请求
6	vpu_coherent	RW	0x1	VPU 设备的 DMA 请求类别: 0: 非一致性请求 1: 为一致性请求
5	cam_sel	RW	0x0	使能 CAMERA 的管脚功能: 0: 管脚保留 1: 管脚为 CAMERA 注: dvo1_sel、cam_sel 二者只能有一个为 1
4	dvo1_sel	RW	0x0	使能 DVO1 的管脚功能: 0: 管脚保留 1: 管脚为 DVO1 注: dvo1_sel、cam_sel 二者只能有一个为 1
3	dc_coherent	RW	0x1	DC 设备的 DMA 请求类别: 0: 非一致性请求 1: 为一致性请求
2	gpu_coherent	RW	0x1	GPU 设备的 DMA 请求类别: 0: 非一致性请求 1: 为一致性请求
1	dvo0_sel	RW	0x0	使能 DVO0 的管脚功能: 0: 管脚保留 1: 管脚为 DVO0 注: dvo0_sel、lio_sel 和 uart1/2_sel 三者只能有一个为 1

位域	名称	访问	缺省值	描述
0	gpu_soft_reset	RW	0x0	GPU 的软件复位: 1: 代表复位 0: 代表解复位

## 5.4 APBDMA 配置寄存器

APBDMA 配置寄存器用于给 APB 设备配置对应的 DMA 控制器。

地址: 0x1fe10438

表 5-5 APBDMA 配置寄存器

位域	名称	访问	缺省值	描述
63:30	Reserved	RO	0x0	保留
29:27	Reserved	RW	0x2	保留
26:24	Reserved	RW	0x1	保留
23:21	dma_sel7	RW	0x1	I2S 控制器接收端所用的 DMA 控制器: 0x0: 对应 DMA0 控制器 0x1: 对应 DMA1 控制器 0x2: 对应 DMA2 控制器 0x3: 对应 DMA3 控制器 0x4: 对应 DMA4 控制器 其他: 保留
20:18	dma_sel6	RW	0x0	I2S 控制器发送端所用的 DMA 控制器: 0x0: 对应 DMA0 控制器 0x1: 对应 DMA1 控制器 0x2: 对应 DMA2 控制器 0x3: 对应 DMA3 控制器 0x4: 对应 DMA4 控制器 其他: 保留
17:15	dma_sel5	RW	0x0	SDIO 控制器所用的 DMA 控制器: 0x0: 对应 DMA0 控制器 0x1: 对应 DMA1 控制器 0x2: 对应 DMA2 控制器 0x3: 对应 DMA3 控制器 0x4: 对应 DMA4 控制器 其他: 保留
14:12	dma_sel4	RW	0x1	DES 控制器写操作所用的 DMA 控制器: 0x0: 对应 DMA0 控制器 0x1: 对应 DMA1 控制器 0x2: 对应 DMA2 控制器 0x3: 对应 DMA3 控制器 0x4: 对应 DMA4 控制器 其他: 保留

位域	名称	访问	缺省值	描述
11:9	dma_sel3	RW	0x2	DES 控制器读操作所用的 DMA 控制器: 0x0: 对应 DMA0 控制器 0x1: 对应 DMA1 控制器 0x2: 对应 DMA2 控制器 0x3: 对应 DMA3 控制器 0x4: 对应 DMA4 控制器 其他: 保留
8:6	dma_sel2	RW	0x1	AES 控制器写操作所用的 DMA 控制器: 0x0: 对应 DMA0 控制器 0x1: 对应 DMA1 控制器 0x2: 对应 DMA2 控制器 0x3: 对应 DMA3 控制器 0x4: 对应 DMA4 控制器 其他: 保留
5:3	dma_sel1	RW	0x2	AES 控制器读操作所用的 DMA 控制器: 0x0: 对应 DMA0 控制器 0x1: 对应 DMA1 控制器 0x2: 对应 DMA2 控制器 0x3: 对应 DMA3 控制器 0x4: 对应 DMA4 控制器 其他: 保留
2:0	dma_sel0	RW	0x0	NAND 控制器所用的 DMA 控制器: 0x0: 对应 DMA0 控制器 0x1: 对应 DMA1 控制器 0x2: 对应 DMA2 控制器 0x3: 对应 DMA3 控制器 0x4: 对应 DMA4 控制器 其他: 保留

## 5.5 USB PHY0/1 配置寄存器

配置 USB 接口 0/1 相关的电气特性

地址: 0x1fe10440

表 5-6USB 0/1 PHY 配置寄存器

位域	名称	访问	缺省值	描述
61	usb_phy_cfg	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
60	usb_phy_cfg	R/W	0	dm 端口下拉电阻使能 0: D-使能 1: D-关闭

位域	名称	访问	缺省值	描述
59	usb_phy_cfg	R/W	0	dp 端口下拉电阻使能 0: D+使能 1: D+关闭
58:57	usb_phy_cfg	R/W	0	该信号调整了驱动电阻, 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
56	usb_phy_cfg	R/W	0	调整了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
55:54	usb_phy_cfg	R/W	0	调整了在高速模式从 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
53:52	usb_phy_cfg	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
51:50	usb_phy_cfg	R/W	0	调整了高速波形上升、下降沿次数 11: -10% 10: default 01: +15% 00: +20%

位域	名称	访问	缺省值	描述
49:46	usb_phy_cfg	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%
45:42	usb_phy_cfg	R/W	0	调整了低、全速单端源阻抗 1111: -5% 0111: -2.5% 0011: default 0001: +2.5% 0000: +5%
41:39	usb_phy_cfg	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%
38:36	usb_phy_cfg	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%

位域	名称	访问	缺省值	描述
35:33	usb_phy_cfg	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%
32	usb_phy_cfg	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
30	usb_phy_cfg	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	usb_phy_cfg	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	usb_phy_cfg	R/W	0	dm 端口下拉电阻使能 1: D-使能 0: D-关闭
27	usb_phy_cfg	R/W	0	dp 端口下拉电阻使能 1: D+使能 0: D+关闭
26:25	usb_phy_cfg	R/W	0	同上
24	usb_phy_cfg	R/W	0	同上
23:22	usb_phy_cfg	R/W	0	同上
21:20	usb_phy_cfg	R/W	0	同上
19:18	usb_phy_cfg	R/W	0	同上
17:14	usb_phy_cfg	R/W	0	同上
13:10	usb_phy_cfg	R/W	0	同上
9:7	usb_phy_cfg	R/W	0	同上
6:4	usb_phy_cfg	R/W	0	同上
3:1	usb_phy_cfg	R/W	0	同上
0	usb_phy_cfg	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

## 5.6 USB PHY2/3 配置寄存器

配置 USB 接口 2/3 相关的电气特性。

地址: 0x1fe10448

表 5-7USB 2/3 PHY 配置寄存器

位域	名称	访问	缺省值	描述
61	usb_phy_cfg	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
60	usb_phy_cfg	R/W	0	dm 端口下拉电阻使能 0: D-使能 1: D-关闭
59	usb_phy_cfg	R/W	0	dp 端口下拉电阻使能 0: D+使能 1: D+关闭
58:57	usb_phy_cfg	R/W	0	同上
56	usb_phy_cfg	R/W	0	同上
55:54	usb_phy_cfg	R/W	0	同上
53:52	usb_phy_cfg	R/W	0	同上
51:50	usb_phy_cfg	R/W	0	同上
49:46	usb_phy_cfg	R/W	0	同上
45:42	usb_phy_cfg	R/W	0	同上
41:39	usb_phy_cfg	R/W	0	同上
38:36	usb_phy_cfg	R/W	0	同上
35:33	usb_phy_cfg	R/W	0	同上
32	usb_phy_cfg	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
29	usb_phy_cfg	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	usb_phy_cfg	R/W	0	dm 端口下拉电阻使能 1: D-使能 0: D-关闭
27	usb_phy_cfg	R/W	0	dp 端口下拉电阻使能 1: D+使能 0: D+关闭
26:25	usb_phy_cfg	R/W	0	同上
24	usb_phy_cfg	R/W	0	同上

位域	名称	访问	缺省值	描述
23:22	usb_phy_cfg	R/W	0	同上
21:20	usb_phy_cfg	R/W	0	同上
19:18	usb_phy_cfg	R/W	0	同上
17:14	usb_phy_cfg	R/W	0	同上
13:10	usb_phy_cfg	R/W	0	同上
9:7	usb_phy_cfg	R/W	0	同上
6:4	usb_phy_cfg	R/W	0	同上
3:1	usb_phy_cfg	R/W	0	同上
0	usb_phy_cfg	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

## 5.7 SATA 配置寄存器

配置 SATA 的控制信号。

地址: 0x1fe10450

表 5-8 SATA 配置寄存器

位域	名称	访问	缺省值	描述
63	sata_phy_cfg	R/W	0	关闭 SATA PHY
62:57	sata_phy_cfg	R/W	0x3f	GEN3 模式下发送端预加重设置
56:51	sata_phy_cfg	R/W	0x3f	GEN2 模式下发送端预加重设置
50:45	sata_phy_cfg	R/W	0x3f	GEN1 模式下发送端预加重设置
44:38	sata_phy_cfg	R/W	0x7f	GEN3 模式下发送端摆幅设置
37:31	sata_phy_cfg	R/W	0x7f	GEN2 模式下发送端摆幅设置
30:24	sata_phy_cfg	R/W	0x7f	GEN1 模式下发送端摆幅设置
23	sata_phy_cfg	R/W	0x1	Sata 接口预取
22	sata_phy_cfg	R/W	0x0	Sata 接口读请求刷出写请求
21	sata_phy_cfg	R/W	0x0	Sata 接口停止写后写
20	sata_phy_cfg	R/W	0x1	Sata 接口停止写后读
19:16	sata_phy_cfg	R/W	0xf	Sata 接口空闲写入周期数
15	sata_phy_cfg	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	sata_phy_cfg	R/W	0x0	Rx equalizer 设置
11	sata_phy_cfg			
10	sata_phy_cfg	R/W	0x1	dma cache 一致性使能
9	sata_phy_cfg	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	sata_phy_cfg	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	sata_phy_cfg	R/W	0x0	Spread Spectrum 使能

位域	名称	访问	缺省值	描述
6:4	sata_phy_cfg	R/W	0x0	Spread Spectrum Clock Range
3	sata_phy_cfg	R/W	0x1	LANE0 软复位, 0 有效
2	sata_phy_cfg	R/W	0x1	PHY 软复位, 0 有效
1	sata_phy_cfg	R/W	0x1	参考时钟输入选择: 0- 选择内部时钟 1- 选择外部时钟
0	sata_phy_cfg	R/W	0x1	PHY 参考时钟使能

## 5.8 NODE PLL 低 64 位配置寄存器

该寄存器用来设置 NODE PLL，具体参数用法请参考第 3 章（时钟结构）。

地址: 0x1fe10480

表 5-9 NODE PLL 低 64 位配置寄存器

位域	名称	访问	缺省值	描述
63:54	L2_div_loop_node	R/W	0	保留
53:48	L2_div_refc_node	R/W	0	保留
47:42	L1_div_out	R/W	0	保留
41:32	L1_div_loopc	R/W	0	L1 PLL 倍频系数
31:26	L1_div_ref	R/W	0	L1 PLL 参考时钟分频系数
23	serial_mode3	R/W	0	保留
22	serial_mode	R/W	0	保留
21	pd_pll_L2_core	R/W	0	保留
20	pd_pll_L2_node	R/W	0	保留
19	pd_pll_L1	R/W	0	L1 PLL 关电控制, 1 代表关电
18	locked_L2_core	R/W	0	保留
17	locked_L2_node	R/W	0	保留
16	locked_L1	R	0	L1 PLL 锁定标志, 1 代表锁定
15:14	lockc_L2_core	R/W	0	保留
13:12	lockc_L2_node	R/W	0	保留
11:10	lockc_L1	R/W	0	判定 L1 PLL 是否锁定使用的相位的精度
9	locken_L2_core	R/W	0	保留
8	locken_L2_node	R/W	0	保留
7	locken_L1	R/W	0	允许锁定 L1 PLL
6	bypass_refin_L2	R/W	0	保留
5	bypass_L2_core	R/W	0	保留
4	bypass_L2_node	R/W	0	保留
3	bypass_L1	R/W	0	Bypass L1 PLL
2	pll_soft_set	R/W	0	允许软件设置 PLL
1	sel_pll_core	R/W	0	保留

位域	名称	访问	缺省值	描述
0	sel_pll_node	R/W	0	NODE 时钟非软件 bypass 整个 PLL

## 5.9 NODE PLL 高 64 位配置寄存器

该寄存器用来设置 NODE PLL，具体参数用法请参考第 3 章（时钟结构）。

地址：0x1fe10488

表 5-10 NODE PLL 高 64 位配置寄存器

位域	名称	访问	缺省值	描述
63:34	-	R/W	0	保留
33:28	L2_div_out_PCIE	R/W	0	保留
27:22	L2_div_out_core	R/W	0	保留
21:12	L2_div_loopc_core	R/W	0	保留
11:6	L2_div_refc_core	R/W	0	保留
5:0	L2_div_out_node	R/W	0	L2 NODE PLL 分频系数

## 5.10 DDR PLL 低 64 位配置寄存器

该寄存器用来设置 DDR PLL，具体参数用法请参考第 3 章（时钟结构）。

地址：0x1fe10490

表 5-11 DDR PLL 低 64 位配置寄存器

位域	名称	访问	缺省值	描述
63:54	-	R/W	0	保留
53:48	-	R/W	0	保留
47:42	L1_div_out	R/W	0	保留
41:32	L1_div_loopc	R/W	0	L1 PLL 倍频系数
31:26	L1_div_ref	R/W	0	L1 PLL 参考时钟分频系数
23	-	R/W	0	保留
22	-	R/W	0	保留
21	-	R/W	0	保留
20	-	R/W	0	保留
19	pd_pll_L1	R/W	0	L1 PLL 关电控制，1 代表关电
18	-	R/W	0	保留
17	-	R/W	0	保留
16	locked_L1	R	0	L1 PLL 锁定标志，1 代表锁定
15:14	-	R/W	0	保留
13:12	-	R/W	0	保留
11:10	lockc_L1	R/W	0	判定 L1 PLL 是否锁定使用的相位的精度
9	-	R/W	0	保留

位域	名称	访问	缺省值	描述
8	-	R/W	0	保留
7	locken_L1	R/W	0	允许锁定 L1 PLL
6	-	R/W	0	保留
5	-	R/W	0	保留
4	-	R/W	0	保留
3	bypass_L1	R/W	0	Bypass L1 PLL
2	pll_soft_set	R/W	0	允许软件设置 PLL
1	sel_pll_gpu	R/W	0	GPU/HDA 时钟非软件 bypass 整个 PLL
0	sel_pll_ddr	R/W	0	DDR 时钟非软件 bypass 整个 PLL

## 5.11 DDR PLL 高 64 位配置寄存器

该寄存器用来设置 DDR PLL，具体参数用法请参考第 3 章（时钟结构）。

地址：0x1fe10498

表 5-12 DDR PLL 高 64 位配置寄存器

位域	名称	访问	缺省值	描述
63:51	-	R/W	0	保留
50:44	L2_div_out_hda	R/W	0	L2 HDA PLL 分频系数
43:28	-	R/W	0	保留
27:22	L2_div_out_gpu	R/W	0	L2 GPU PLL 分频系数
21:12	-	R/W	0	保留
11:6	-	R/W	0	保留
5:0	L2_div_out_ddr	R/W	0	L2 DDR PLL 分频系数

## 5.12 DC PLL 低 64 位配置寄存器

该寄存器用来设置 DCPLL，具体参数用法请参考第 3 章（时钟结构）。

地址：0x1fe104a0

表 5-13 DC PLL 低 64 位配置寄存器

位域	名称	访问	缺省值	描述
63:54	-	R/W	0	保留
53:48	-	R/W	0	保留
47:42	L1_div_out	R/W	0	保留
41:32	L1_div_loopc	R/W	0	L1 PLL 倍频系数
31:26	L1_div_ref	R/W	0	L1 PLL 参考时钟分频系数
23	-	R/W	0	保留
22	-	R/W	0	保留
21	-	R/W	0	保留

位域	名称	访问	缺省值	描述
20	-	R/W	0	保留
19	pd_pll_L1	R/W	0	L1 PLL 关电控制, 1 代表关电
18	-	R/W	0	保留
17	-	R/W	0	保留
16	locked_L1	R	0	L1 PLL 锁定标志, 1 代表锁定
15:14	-	R/W	0	保留
13:12	-	R/W	0	保留
11:10	lockc_L1	R/W	0	判定 L1 PLL 是否锁定使用的相位的精度
9	-	R/W	0	保留
8	-	R/W	0	保留
7	locken_L1	R/W	0	允许锁定 L1 PLL
6	-	R/W	0	保留
5	-	R/W	0	保留
4	-	R/W	0	保留
3	bypass_L1	R/W	0	Bypass L1 PLL
2	pll_soft_set	R/W	0	允许软件设置 PLL
1	sel_pll_gmac	R/W	0	GMAC 时钟非软件 bypass 整个 PLL
0	sel_pll_dc	R/W	0	DC 时钟非软件 bypass 整个 PLL

## 5.13 DC PLL 高 64 位配置寄存器

该寄存器用来设置 DC PLL，具体参数用法请参考第 3 章（时钟结构）。

地址：0x1fe104a8

表 5-14 DC PLL 高 64 位配置寄存器

位域	名称	访问	缺省值	描述
63:34	-	R/W	0	保留
33:28	-	R/W	0	保留
27:22	L2_div_out_gma	R/W	0	L2 GMAC PLL 分频系数
21:12	-	R/W	0	保留
11:6	-	R/W	0	保留
5:0	L2_div_out_dc	R/W	0	L2 DC PLL 分频系数

## 5.14 PIX0 PLL 低 64 位配置寄存器

该寄存器用来设置 PIX0 PLL，具体参数用法请参考第 3 章（时钟结构）。

地址：0x1fe104b0

表 5-15 PIX0 PLL 低 64 位配置寄存器

位域	名称	访问	缺省值	描述
63:54	-	R/W	0	保留

位域	名称	访问	缺省值	描述
53:48	-	R/W	0	保留
47:42	L1_div_out	R/W	0	保留
41:32	L1_div_loopc	R/W	0	L1 PLL 倍频系数
31:26	L1_div_ref	R/W	0	L1 PLL 参考时钟分频系数
23	-	R/W	0	保留
22	-	R/W	0	保留
21	-	R/W	0	保留
20	-	R/W	0	保留
19	pd_pll_L1	R/W	0	L1 PLL 关电控制, 1 代表关电
18	-	R/W	0	保留
17	-	R/W	0	保留
16	locked_L1	R	0	L1 PLL 锁定标志, 1 代表锁定
15:14	-	R/W	0	保留
13:12	-	R/W	0	保留
11:10	lockc_L1	R/W	0	判定 L1 PLL 是否锁定使用的相位的精度
9	-	R/W	0	保留
8	-	R/W	0	保留
7	locken_L1	R/W	0	允许锁定 L1 PLL
6	-	R/W	0	保留
5	-	R/W	0	保留
4	-	R/W	0	保留
3	bypass_L1	R/W	0	Bypass L1 PLL
2	pll_soft_set	R/W	0	允许软件设置 PLL
1	-	R/W	0	保留
0	sel_pll_pix0	R/W	0	PIX0 时钟非软件 bypass 整个 PLL

## 5.15 PIX0 PLL 高 64 位配置寄存器

该寄存器用来设置 PIX0 PLL，具体参数用法请参考第 3 章（时钟结构）。

地址：0x1fe104b8

表 5-16 PIX0 PLL 高 64 位配置寄存器

位域	名称	访问	缺省值	描述
63:34	-	R/W	0	保留
33:28	-	R/W	0	保留
27:22	-	R/W	0	保留
21:12	-	R/W	0	保留
11:6	-	R/W	0	保留
5:0	L2_div_out_pix0	R/W	0	L2 PIX0 PLL 分频系数

## 5.16 PIX1 PLL 低 64 位配置寄存器

该寄存器用来设置 PIX1 PLL，具体参数用法请参考第 3 章（时钟结构）。

地址：0x1fe104c0

表 5-17 PIX1 PLL 低 64 位配置寄存器

位域	名称	访问	缺省值	描述
63:54	-	R/W	0	保留
53:48	-	R/W	0	保留
47:42	L1_div_out	R/W	0	保留
41:32	L1_div_loopc	R/W	0	L1 PLL 倍频系数
31:26	L1_div_ref	R/W	0	L1 PLL 参考时钟分频系数
23	-	R/W	0	保留
22	-	R/W	0	保留
21	-	R/W	0	保留
20	-	R/W	0	保留
19	pd_pll_L1	R/W	0	L1 PLL 关电控制，1 代表关电
18	-	R/W	0	保留
17	-	R/W	0	保留
16	locked_L1	R	0	L1 PLL 锁定标志，1 代表锁定
15:14	-	R/W	0	保留
13:12	-	R/W	0	保留
11:10	lockc_L1	R/W	0	判定 L1 PLL 是否锁定使用的相位的精度
9	-	R/W	0	保留
8	-	R/W	0	保留
7	locken_L1	R/W	0	允许锁定 L1 PLL
6	-	R/W	0	保留
5	-	R/W	0	保留
4	-	R/W	0	保留
3	bypass_L1	R/W	0	Bypass L1 PLL
2	pll_soft_set	R/W	0	允许软件设置 PLL
1	-	R/W	0	保留
0	sel_pll_pix1	R/W	0	PIX1 时钟非软件 bypass 整个 PLL

## 5.17 PIX1 PLL 高 64 位配置寄存器

该寄存器用来设置 PIX0 PLL，具体参数用法请参考第 3 章（时钟结构）。

地址：0x1fe104c8

表 5-18 PIX1 PLL 高 64 位配置寄存器

位域	名称	访问	缺省值	描述
----	----	----	-----	----

位域	名称	访问	缺省值	描述
63:34	-	R/W	0	保留
33:28	-	R/W	0	保留
27:22	-	R/W	0	保留
21:12	-	R/W	0	保留
11:6	-	R/W	0	保留
5:0	L2_div_out_pix1	R/W	0	L2 PIX1 PLL 分频系数

## 5.18 FREQSCALE 配置寄存器

该寄存器用于配置时钟的分频系数以及 CPU 核的时钟使能。

地址: 0x1fe104d0

表 5-19 FRESCALE 配置寄存器

位域	名称	访问	缺省值	描述
63:34	-	R/W	0	保留
33	core1_en	R/W	0x1	CPU 核 1 时钟使能
32	core0_en	R/W	0x1	CPU 核 0 时钟使能
22:20	apb_freqscale	R/W	0x7	apb 时钟分频系数 计算公式为 $8/(n+1)$ , 比如设置成 3 时代表 2 分频
18:16	usb_freqscale	R/W	0x7	usb 时钟分频系数, 计算方法同上
14:12	sata_freqscale	R/W	0x7	sata 时钟分频系数, 计算方法同上
10:8	boot_freqscale	R/W	0x7	boot 时钟分频系数, 计算方法同上
7:3	-	-	-	保留
2:0	node_freqscale	R/W	0x7	node 时钟分频系数

Freqscale 分频计算公式为:  $f = fref * (freqscale + 1) / 8$ .

## 5.19 PCIE0 配置寄存器 0

配置 PCIE0 PHY 的控制信号。

地址: 0x1fe10580

表 5-20 PCIE0 配置寄存器 0

位域	名称	访问	缺省值	描述
63:59	pcie_phy_cfg	R/W	0	设置 PCIE LANE0 发送端阻抗在 50 欧左右的微调值
58:52	pcie_phy_cfg	R/W	0x2	设置 PCIE PHY 发送低摆幅幅值
51:45	pcie_phy_cfg	R/W	0x10	设置 PCIE PHY 发送满摆幅幅值
44:39	pcie_phy_cfg	R/W	0x30	设置 PCIE PHY 在 GEN2 模式下的 -6dB tx deemphasis 值
38:33	pcie_phy_cfg	R/W	0x20	设置 PCIE PHY 在 GEN2 模式下的 -3.5dB tx deemphasis 值

位域	名称	访问	缺省值	描述
32:27	pcie_phy_cfg	R/W	0x10	设置 PCIE PHY 在 GEN1 模式下的 tx deemphasis 值
26:24	pcie_phy_cfg	R/W	0x2	LANE3 接收端 equalizer 设置
23:21	pcie_phy_cfg	R/W	0x2	LANE2 接收端 equalizer 设置
20:18	pcie_phy_cfg	R/W	0x2	LANE1 接收端 equalizer 设置
17:15	pcie_phy_cfg	R/W	0x2	LANE0 接收端 equalizer 设置
14	pcie_phy_cfg	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	pcie_phy_cfg	R/W	0x1	表示 PCIE PHY 在 P2 状态下需要输出 PIPE 时钟
12	pcie_phy_cfg	-	-	保留
11:5	pcie_phy_cfg	R/W	0x19	PCIE PHY PLL 的倍频数
4	pcie_phy_cfg	-	-	保留
3	pcie_phy_cfg	R/W	0x0	保留
2	pcie_phy_cfg	R/W	0x0	保留
1	pcie_phy_cfg	R/W	0x0	保留
0	pcie_phy_cfg	R/W	0x1	保留

## 5.20 PCIE0 配置寄存器 1

配置 PCIE0 PHY 的接口信号。

地址: 0x1fe10588

表 5-21 PCIE0 配置寄存器 1

位域	名称	访问	缺省值	描述
63:25	-	-	-	保留
24	pcie_phy_cfg	R/W	0	设置 PCIE PHY 进入低功耗模式
23:20	pcie_phy_cfg	R/W	0	PCIE 参考时钟输出使能
19	pcie_phy_cfg	R/W	0	发起 resister tune 请求
18	pcie_phy_cfg	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	pcie_phy_cfg	R/W	0x4	Tx voltage boost level
14:10	pcie_phy_cfg	R/W	0	设置 PCIE LANE3 发送端阻抗在 50 欧左右的微调值
9:5	pcie_phy_cfg	R/W	0	设置 PCIE LANE2 发送端阻抗在 50 欧左右的微调值
4:0	pcie_phy_cfg	R/W	0	设置 PCIE LANE1 发送端阻抗在 50 欧左右的微调值

## 5.21 PCIE0 PHY 配置控制寄存器

通过这些寄存器配置 PCIE0 PHY 的内部寄存器。

地址: 0x1fe10590

表 5-22 PCIE0 PHY 配置寄存器

位域	名称	访问	缺省值	描述
63:39	-	-	-	保留
38	phy_cfg_reset	R/W	0	PHY 配置重置, 高有效
37:35	phy_cfg_state	R	0	PHY 配置状态机状态指示
34	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
33	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
32	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。
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

## 5.22 PCIE1 配置寄存器 0

配置 PCIE1 PHY 的控制信号。

地址: 0x1fe105a0

表 5-23 PCIE1 配置寄存器 0

位域	名称	访问	缺省值	描述
63:59	pcie_phy_cfg	R/W	0	设置 PCIE LANE0 发送端阻抗在 50 欧左右的微调值
58:52	pcie_phy_cfg	R/W	0x2	设置 PCIE PHY 发送低摆幅幅值
51:45	pcie_phy_cfg	R/W	0x10	设置 PCIE PHY 发送满摆幅幅值
44:39	pcie_phy_cfg	R/W	0x30	设置 PCIE PHY 在 GEN2 模式下的 -6dB tx deemphasis 值
38:33	pcie_phy_cfg	R/W	0x20	设置 PCIE PHY 在 GEN2 模式下的 -3.5dB tx deemphasis 值

位域	名称	访问	缺省值	描述
32:27	pcie_phy_cfg	R/W	0x10	设置 PCIE PHY 在 GEN1 模式下的 tx deemphasis 值
26:24	pcie_phy_cfg	R/W	0x2	LANE3 接收端 equalizer 设置
23:21	pcie_phy_cfg	R/W	0x2	LANE2 接收端 equalizer 设置
20:18	pcie_phy_cfg	R/W	0x2	LANE1 接收端 equalizer 设置
17:15	pcie_phy_cfg	R/W	0x2	LANE0 接收端 equalizer 设置
14	pcie_phy_cfg	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	pcie_phy_cfg	R/W	0x1	表示 PCIE PHY 在 P2 状态下需要输出 PIPE 时钟
12	pcie_phy_cfg	-	-	保留
11:5	pcie_phy_cfg	R/W	0x19	PCIE PHY PLL 的倍频数
4	pcie_phy_cfg	-	-	保留
3	pcie_phy_cfg	R/W	0x0	保留
2	pcie_phy_cfg	R/W	0x0	保留
1	pcie_phy_cfg	R/W	0x0	保留
0	pcie_phy_cfg	R/W	0x1	保留

## 5.23 PCIE1 配置寄存器 1

配置 PCIE1 PHY 的接口信号。

地址: 0x1fe105a8

表 5-24 PCIE1 配置寄存器 1

位域	名称	访问	缺省值	描述
63:25	-	-	-	保留
24	pcie_phy_cfg	R/W	0	设置 PCIE PHY 进入低功耗模式
23:20	pcie_phy_cfg	R/W	0	PCIE 参考时钟输出使能
19	pcie_phy_cfg	R/W	0	发起 resister tune 请求
18	pcie_phy_cfg	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	pcie_phy_cfg	R/W	0x4	Tx voltage boost level
14:10	pcie_phy_cfg	R/W	0	设置 PCIE LANE3 发送端阻抗在 50 欧左右的微调值
9:5	pcie_phy_cfg	R/W	0	设置 PCIE LANE2 发送端阻抗在 50 欧左右的微调值
4:0	pcie_phy_cfg	R/W	0	设置 PCIE LANE1 发送端阻抗在 50 欧左右的微调值

## 5.24 PCIE1 PHY 配置控制寄存器

通过这些寄存器配置 PCIE1 PHY 的内部寄存器。

地址: 0x1fe105b0

表 5-25 PCIE1 PHY 配置寄存器

位域	名称	访问	缺省值	描述
63:39	-	-	-	保留
38	phy_cfg_reset	R/W	0	PHY 配置重置, 高有效
37:35	phy_cfg_state	R	0	PHY 配置状态机状态指示
34	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
33	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
32	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。
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

## 5.25 DMA 命令控制寄存器(dma\_order)

该寄存器用来控制 DMA 控制器。2K1000 中包含 5 个 DMA 控制器, 它们的结构相同, 所以控制寄存器的内容也一致, 所以这里仅列出一个。

DMA0 地址: 0x1fe10c00

DMA1 地址: 0x1fe10c10

DMA2 地址: 0x1fe10c20

DMA3 地址: 0x1fe10c30

DMA4 地址: 0x1fe10c40

表 5-26 DMA 命令控制寄存器

位域	名称	访问	缺省值	描述
63:5	ask_addr	R/W	0	64 位地址高 59
5	-	-	0	保留

位域	名称	访问	缺省值	描述
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	axi_coherent	R/W	0	DMA 访问地址一致性使能
0	dma_64bit	R/W	0	DMA 控制器 64 位地址支持

## 5.26 PCICFG2\_RECFC 寄存器

PCICFG2\_RECFC 控制器用于使能 APB 总线控制器的配置头空间的重配置功能，表示对应的字节是否可写。该寄存器主要用于调试。

地址：0x1fe13800

表 5-27 PCICFG2\_RECFC 配置寄存器

位域	名称	访问	缺省值	描述
63:56	Pcicfg2_reconfig[63:56]	RW	0xff	APB 设备配置头空间 Byte63-56 的重配置使能
55:48	Pcicfg2_reconfig[55:48]	RW	0xff	APB 设备配置头空间 Byte55-48 的重配置使能
47:40	Pcicfg2_reconfig[47:40]	RW	0xff	APB 设备配置头空间 Byte47-40 的重配置使能
39:32	Pcicfg2_reconfig[39:32]	RW	0xff	APB 设备配置头空间 Byte39-32 的重配置使能
31:24	Pcicfg2_reconfig[31:24]	RW	0xff	APB 设备配置头空间 Byte31-24 的重配置使能
23:16	Pcicfg2_reconfig[23:16]	RW	0xff	APB 设备配置头空间 Byte23-16 的重配置使能
15:08	Pcicfg2_reconfig[15:08]	RW	0xff	APB 设备配置头空间 Byte15-08 的重配置使能
07:00	Pcicfg2_reconfig[07:00]	RW	0xf0	APB 设备配置头空间 Byte07-00 的重配置使能

## 5.27 PCICFG30\_RECFC 寄存器

PCICFG30\_RECFC 控制器用于使能 GMAC0 控制器的配置头空间的重配置功能，表示对应的字节是否可写。该寄存器主要用于调试。

地址：0x1fe13808

表 5-28 PCICFG30\_RECFC 配置寄存器

位域	名称	访问	缺省值	描述
63:56	Pcicfg30_reconfig[63:56]	RW	0xff	GMAC0 设备配置头空间 Byte63-56 的重配置使能
55:48	Pcicfg30_reconfig[55:48]	RW	0xff	GMAC0 设备配置头空间 Byte55-48 的重配置使能
47:40	Pcicfg30_reconfig[47:40]	RW	0xff	GMAC0 设备配置头空间 Byte47-40 的重配置使能
39:32	Pcicfg30_reconfig[39:32]	RW	0xff	GMAC0 设备配置头空间 Byte39-32 的重配置使能
31:24	Pcicfg30_reconfig[31:24]	RW	0xff	GMAC0 设备配置头空间 Byte31-24 的重配置使能
23:16	Pcicfg30_reconfig[23:16]	RW	0xff	GMAC0 设备配置头空间 Byte23-16 的重配置使能
15:08	Pcicfg30_reconfig[15:08]	RW	0xff	GMAC0 设备配置头空间 Byte15-08 的重配置使能
07:00	Pcicfg30_reconfig[07:00]	RW	0xf0	GMAC0 设备配置头空间 Byte07-00 的重配置使能

## 5.28 PCICFG31\_RECFCG 寄存器

PCICFG31\_RECFCG 控制器用于使能 GMAC1 控制器的配置头空间的重配置功能，表示对应的字节是否可写。该寄存器主要用于调试。

地址：0x1fe13810

表 5-29 PCICFG31\_RECFCG 配置寄存器

位域	名称	访问	缺省值	描述
63:56	Pcicfg31_reconfig[63:56]	RW	0xff	GMAC1 设备配置头空间 Byte63-56 的重配置使能
55:48	Pcicfg31_reconfig[55:48]	RW	0xff	GMAC1 设备配置头空间 Byte55-48 的重配置使能
47:40	Pcicfg31_reconfig[47:40]	RW	0xff	GMAC1 设备配置头空间 Byte47-40 的重配置使能
39:32	Pcicfg31_reconfig[39:32]	RW	0xff	GMAC1 设备配置头空间 Byte39-32 的重配置使能
31:24	Pcicfg31_reconfig[31:24]	RW	0xff	GMAC1 设备配置头空间 Byte31-24 的重配置使能
23:16	Pcicfg31_reconfig[23:16]	RW	0xff	GMAC1 设备配置头空间 Byte23-16 的重配置使能
15:08	Pcicfg31_reconfig[15:08]	RW	0xff	GMAC1 设备配置头空间 Byte15-08 的重配置使能
07:00	Pcicfg31_reconfig[07:00]	RW	0xf0	GMAC1 设备配置头空间 Byte07-00 的重配置使能



## 5.29 PCICFG40\_RECFC 寄存器

PCICFG40\_RECFC 控制器用于使能 USB-OTG 控制器的配置头空间的重配置功能，表示对应的字节是否可写。该寄存器主要用于调试。

地址：0x1fe13818

表 5-30 PCICFG40\_RECFC 配置寄存器

位域	名称	访问	缺省值	描述
63:56	Pcicfg40_reconfig[63:56]	RW	0xff	USB-OTG 设备配置头空间 Byte63-56 的重配置使能
55:48	Pcicfg40_reconfig[55:48]	RW	0xff	USB-OTG 设备配置头空间 Byte55-48 的重配置使能
47:40	Pcicfg40_reconfig[47:40]	RW	0xff	USB-OTG 设备配置头空间 Byte47-40 的重配置使能
39:32	Pcicfg40_reconfig[39:32]	RW	0xff	USB-OTG 设备配置头空间 Byte39-32 的重配置使能
31:24	Pcicfg40_reconfig[31:24]	RW	0xff	USB-OTG 设备配置头空间 Byte31-24 的重配置使能
23:16	Pcicfg40_reconfig[23:16]	RW	0xff	USB-OTG 设备配置头空间 Byte23-16 的重配置使能
15:08	Pcicfg40_reconfig[15:08]	RW	0xff	USB-OTG 设备配置头空间 Byte15-08 的重配置使能
07:00	Pcicfg40_reconfig[07:00]	RW	0xf0	USB-OTG 设备配置头空间 Byte07-00 的重配置使能

## 5.30 PCICFG41\_RECFC 寄存器

PCICFG41\_RECFC 控制器用于使能 USB-EHCI 控制器的配置头空间的重配置功能，表示对应的字节是否可写。该寄存器主要用于调试。

地址：0x1fe13820

表 5-31 PCICFG41\_RECFC 配置寄存器

位域	名称	访问	缺省值	描述
63:56	Pcicfg41_reconfig[63:56]	RW	0xff	USB-EHCI 设备配置头空间 Byte63-56 的重配置使能
55:48	Pcicfg41_reconfig[55:48]	RW	0xff	USB-EHCI 设备配置头空间 Byte55-48 的重配置使能
47:40	Pcicfg41_reconfig[47:40]	RW	0xff	USB-EHCI 设备配置头空间 Byte47-40 的重配置使能
39:32	Pcicfg41_reconfig[39:32]	RW	0xff	USB-EHCI 设备配置头空间 Byte39-32 的重配置使能
31:24	Pcicfg41_reconfig[31:24]	RW	0xff	USB-EHCI 设备配置头空间 Byte31-24 的重配置使能



位域	名称	访问	缺省值	描述
23:16	Pcicfg41_reconfig[23:16]	RW	0xff	USB-EHCI 设备配置头空间 Byte23-16 的重配置使能
15:08	Pcicfg41_reconfig[15:08]	RW	0xff	USB-EHCI 设备配置头空间 Byte15-08 的重配置使能
07:00	Pcicfg41_reconfig[07:00]	RW	0xf0	USB-EHCI 设备配置头空间 Byte07-00 的重配置使能

## 5.31 PCICFG42\_RECFCG 寄存器

PCICFG42\_RECFCG 控制器用于使能 USB-OHCI 控制器的配置头空间的重配置功能，表示对应的字节是否可写。该寄存器主要用于调试。

地址：0x1fe13828

表 5-32 PCICFG42\_RECFCG 配置寄存器

位域	名称	访问	缺省值	描述
63:56	Pcicfg42_reconfig[63:56]	RW	0xff	USB-OHCI 设备配置头空间 Byte63-56 的重配置使能
55:48	Pcicfg42_reconfig[55:48]	RW	0xff	USB-OHCI 设备配置头空间 Byte55-48 的重配置使能
47:40	Pcicfg42_reconfig[47:40]	RW	0xff	USB-OHCI 设备配置头空间 Byte47-40 的重配置使能
39:32	Pcicfg42_reconfig[39:32]	RW	0xff	USB-OHCI 设备配置头空间 Byte39-32 的重配置使能
31:24	Pcicfg42_reconfig[31:24]	RW	0xff	USB-OHCI 设备配置头空间 Byte31-24 的重配置使能
23:16	Pcicfg42_reconfig[23:16]	RW	0xff	USB-OHCI 设备配置头空间 Byte23-16 的重配置使能
15:08	Pcicfg42_reconfig[15:08]	RW	0xff	USB-OHCI 设备配置头空间 Byte15-08 的重配置使能
07:00	Pcicfg42_reconfig[07:00]	RW	0xf0	USB-OHCI 设备配置头空间 Byte07-00 的重配置使能

## 5.32 PCICFG5\_RECFCG 寄存器

PCICFG5\_RECFCG 控制器用于使能 GPU 控制器的配置头空间的重配置功能，表示对应的字节是否可写。该寄存器主要用于调试。

地址：0x1fe13830

表 5-33 PCICFG5\_RECFCG 配置寄存器

位域	名称	访问	缺省值	描述
63:56	Pcicfg5_reconfig[63:56]	RW	0xff	GPU 设备配置头空间 Byte63-56 的重配置使能

位域	名称	访问	缺省值	描述
55:48	Pcicfg5_reconfig[55:48]	RW	0xff	GPU 设备配置头空间 Byte55-48 的重配置使能
47:40	Pcicfg5_reconfig[47:40]	RW	0xff	GPU 设备配置头空间 Byte47-40 的重配置使能
39:32	Pcicfg5_reconfig[39:32]	RW	0xff	GPU 设备配置头空间 Byte39-32 的重配置使能
31:24	Pcicfg5_reconfig[31:24]	RW	0xff	GPU 设备配置头空间 Byte31-24 的重配置使能
23:16	Pcicfg5_reconfig[23:16]	RW	0xff	GPU 设备配置头空间 Byte23-16 的重配置使能
15:08	Pcicfg5_reconfig[15:08]	RW	0xff	GPU 设备配置头空间 Byte15-08 的重配置使能
07:00	Pcicfg5_reconfig[07:00]	RW	0xf0	GPU 设备配置头空间 Byte07-00 的重配置使能

### 5.33 PCICFG6\_RECFC 寄存器

PCICFG6\_RECFC 控制器用于使能 DC 控制器的配置头空间的重配置功能，表示对应的字节是否可写。该寄存器主要用于调试。

地址：0x1fe13838

表 5-34 PCICFG6\_RECFC 配置寄存器

位域	名称	访问	缺省值	描述
63:56	Pcicfg6_reconfig[63:56]	RW	0xff	DC 设备配置头空间 Byte63-56 的重配置使能
55:48	Pcicfg6_reconfig[55:48]	RW	0xff	DC 设备配置头空间 Byte55-48 的重配置使能
47:40	Pcicfg6_reconfig[47:40]	RW	0xff	DC 设备配置头空间 Byte47-40 的重配置使能
39:32	Pcicfg6_reconfig[39:32]	RW	0xff	DC 设备配置头空间 Byte39-32 的重配置使能
31:24	Pcicfg6_reconfig[31:24]	RW	0xff	DC 设备配置头空间 Byte31-24 的重配置使能
23:16	Pcicfg6_reconfig[23:16]	RW	0xff	DC 设备配置头空间 Byte23-16 的重配置使能
15:08	Pcicfg6_reconfig[15:08]	RW	0xff	DC 设备配置头空间 Byte15-08 的重配置使能
07:00	Pcicfg6_reconfig[07:00]	RW	0xf0	DC 设备配置头空间 Byte07-00 的重配置使能

### 5.34 PCICFG7\_RECFC 寄存器

PCICFG7\_RECFC 控制器用于使能 HDA 控制器的配置头空间的重配置功能，表示对应的字节是否可写。该寄存器主要用于调试。

地址：0x1fe13840

表 5-35 PCICFG7\_RECFC 配置寄存器

位域	名称	访问	缺省值	描述
63:56	Pcicfg7_reconfig[63:56]	RW	0xff	HDA 设备配置头空间 Byte63-56 的重配置使能

位域	名称	访问	缺省值	描述
55:48	Pcicfg7_reconfig[55:48]	RW	0xff	HDA 设备配置头空间 Byte55-48 的重配置使能
47:40	Pcicfg7_reconfig[47:40]	RW	0xff	HDA 设备配置头空间 Byte47-40 的重配置使能
39:32	Pcicfg7_reconfig[39:32]	RW	0xff	HDA 设备配置头空间 Byte39-32 的重配置使能
31:24	Pcicfg7_reconfig[31:24]	RW	0xff	HDA 设备配置头空间 Byte31-24 的重配置使能
23:16	Pcicfg7_reconfig[23:16]	RW	0xff	HDA 设备配置头空间 Byte23-16 的重配置使能
15:08	Pcicfg7_reconfig[15:08]	RW	0xff	HDA 设备配置头空间 Byte15-08 的重配置使能
07:00	Pcicfg7_reconfig[07:00]	RW	0xf0	HDA 设备配置头空间 Byte07-00 的重配置使能

## 5.35 PCICFG8\_RECFC 寄存器

PCICFG8\_RECFC 控制器用于使能 SATA 控制器的配置头空间的重配置功能，表示对应的字节是否可写。该寄存器主要用于调试。

地址：0x1fe13848

表 5-36 PCICFG8\_RECFC 配置寄存器

位域	名称	访问	缺省值	描述
63:56	Pcicfg8_reconfig[63:56]	RW	0xff	SATA 设备配置头空间 Byte63-56 的重配置使能
55:48	Pcicfg8_reconfig[55:48]	RW	0xff	SATA 设备配置头空间 Byte55-48 的重配置使能
47:40	Pcicfg8_reconfig[47:40]	RW	0xff	SATA 设备配置头空间 Byte47-40 的重配置使能
39:32	Pcicfg8_reconfig[39:32]	RW	0xff	SATA 设备配置头空间 Byte39-32 的重配置使能
31:24	Pcicfg8_reconfig[31:24]	RW	0xff	SATA 设备配置头空间 Byte31-24 的重配置使能
23:16	Pcicfg8_reconfig[23:16]	RW	0xff	SATA 设备配置头空间 Byte23-16 的重配置使能
15:08	Pcicfg8_reconfig[15:08]	RW	0xff	SATA 设备配置头空间 Byte15-08 的重配置使能
07:00	Pcicfg8_reconfig[07:00]	RW	0xf0	SATA 设备配置头空间 Byte07-00 的重配置使能



## 5.36 PCICFGf\_RECFCG 寄存器

PCICFGf\_RECFCG 控制器用于使能 DMA 控制器的配置头空间的重配置功能，表示对应的字节是否可写。该寄存器主要用于调试。

地址：0x1fe13850

表 5-37 PCICFGf\_RECFCG 配置寄存器

位域	名称	访问	缺省值	描述
63:56	Pcicfgf_reconfig[63:56]	RW	0xff	DMA 控制器配置头空间 Byte63-56 的重配置使能
55:48	Pcicfgf_reconfig[55:48]	RW	0xff	DMA 控制器配置头空间 Byte55-48 的重配置使能
47:40	Pcicfgf_reconfig[47:40]	RW	0xff	DMA 控制器配置头空间 Byte47-40 的重配置使能
39:32	Pcicfgf_reconfig[39:32]	RW	0xff	DMA 控制器配置头空间 Byte39-32 的重配置使能
31:24	Pcicfgf_reconfig[31:24]	RW	0xff	DMA 控制器配置头空间 Byte31-24 的重配置使能
23:16	Pcicfgf_reconfig[23:16]	RW	0xff	DMA 控制器配置头空间 Byte23-16 的重配置使能
15:08	Pcicfgf_reconfig[15:08]	RW	0xff	DMA 控制器配置头空间 Byte15-08 的重配置使能
07:00	Pcicfgf_reconfig[07:00]	RW	0xf0	DMA 控制器配置头空间 Byte07-00 的重配置使能

## 5.37 PCICFG10\_RECFCG 寄存器

PCICFG10\_RECFCG 控制器用于使能 VPU 解码器的配置头空间的重配置功能，表示对应的字节是否可写。该寄存器主要用于调试。

地址：0x1fe13858

表 5-38 PCICFG10\_RECFCG 配置寄存器

位域	名称	访问	缺省值	描述
63:56	Pcicfg10_reconfig[63:56]	RW	0xff	VPU 解码器配置头空间 Byte63-56 的重配置使能
55:48	Pcicfg10_reconfig[55:48]	RW	0xff	VPU 解码器配置头空间 Byte55-48 的重配置使能
47:40	Pcicfg10_reconfig[47:40]	RW	0xff	VPU 解码器配置头空间 Byte47-40 的重配置使能
39:32	Pcicfg10_reconfig[39:32]	RW	0xff	VPU 解码器配置头空间 Byte39-32 的重配置使能
31:24	Pcicfg10_reconfig[31:24]	RW	0xff	VPU 解码器配置头空间 Byte31-24 的重配置使能



位域	名称	访问	缺省值	描述
23:16	Pcicfg10_reconfig[23:16]	RW	0xff	VPU 解码器配置头空间 Byte23-16 的重配置使能
15:08	Pcicfg10_reconfig[15:08]	RW	0xff	VPU 解码器配置头空间 Byte15-08 的重配置使能
07:00	Pcicfg10_reconfig[07:00]	RW	0xf0	VPU 解码器配置头空间 Byte07-00 的重配置使能

## 5.38 PCICFG11\_RECFCG 寄存器

PCICFG11\_RECFCG 控制器用于使能 CAMERA 模块的配置头空间的重配置功能，表示对应的字节是否可写。该寄存器主要用于调试。

地址：0x1fe13860

表 5-39 PCICFG11\_RECFCG 配置寄存器

位域	名称	访问	缺省值	描述
63:56	Pcicfg11_reconfig[63:56]	RW	0xff	CAMERA 模块配置头空间 Byte63-56 的重配置使能
55:48	Pcicfg11_reconfig[55:48]	RW	0xff	CAMERA 模块配置头空间 Byte55-48 的重配置使能
47:40	Pcicfg11_reconfig[47:40]	RW	0xff	CAMERA 模块配置头空间 Byte47-40 的重配置使能
39:32	Pcicfg11_reconfig[39:32]	RW	0xff	CAMERA 模块配置头空间 Byte39-32 的重配置使能
31:24	Pcicfg11_reconfig[31:24]	RW	0xff	CAMERA 模块配置头空间 Byte31-24 的重配置使能
23:16	Pcicfg11_reconfig[23:16]	RW	0xff	CAMERA 模块配置头空间 Byte23-16 的重配置使能
15:08	Pcicfg11_reconfig[15:08]	RW	0xff	CAMERA 模块配置头空间 Byte15-08 的重配置使能
07:00	Pcicfg11_reconfig[07:00]	RW	0xf0	CAMERA 模块配置头空间 Byte07-00 的重配置使能

# 6 地址空间分配

龙芯 2K1000 的路由主要通过系统的两级交叉开关以及 IO 子网络组成。一级交叉开关采用固定路由；二级交叉开关可以对每个 Master 端口接收到的请求进行路由配置；IO 子网络采用 PCI 总线的拓扑结构，通过对软件扫描设备并配置寄存器基地址的方式来访问，因此每个设备都有一个标准的 PCI 配置头。

龙芯 2K1000 的地址空间与 PCI 定义的地址空间形式相同，主要分为三类：

1. 配置空间：该地址空间用于访问各个 IO 设备的 PCI 配置头，其地址组成符合 PCI 配置访问的地址组织形式；
2. IO 空间：该地址空间用于访问 PCI 协议定义的 IO 地址空间。在 2K1000 中只有 PCIE 有这段地址空间，用于通过 IO 类型的请求访问 PCIE 控制器的下游设备。
3. MEM 空间：除了以上两种地址空间之外的所有地址空间为 MEM 空间。

全芯片的地址空间划分如下表所示：

表 6-1 芯片地址空间划分

NAME	BASE	MASK	SIZE	备注
Memory	64'h0000_0000	64'hFFFF_FFFF_F000_0000	256M	内存空间
Memory mapped IO	64'h1000_0000	64'hFFFF_FFFF_F800_0000	128M	用于映射内部所使用的设备空间(BAR)
PCIE IO	64'h1800_0000	64'hFFFF_FFFF_FE00_0000	32M	用于映射 PCIE 控制器对外的 IO 空间
Type0	64'h1A00_0000	64'hFFFF_FFFF_FF00_0000	16M	Type0 配置空间
Type1	64'h1B00_0000	64'hFFFF_FFFF_FF00_0000	16M	Type1 配置空间
LIO Memory	64'h1C00_0000	64'hFFFF_FFFF_FC00_0000	56M	LIO 访问空间
BOOT	64'h1FC0_0000	64'hFFFF_FFFF_FFF0_0000	1M	启动空间，可映射至 SPI、NAND、SDIO 和 LIO 上
CONFIG	64'h1FE0_0000	64'hFFFF_FFFF_FFF0_0000	1M	芯片配置寄存器空间
SPI	64'h1FFF_0000	64'hFFFF_FFFF_FFFF_0000	64K	SPI 配置空间
Memory mapped IO	64'h4000_0000	64'hFFFF_FFFF_C000_0000	1G	PCIE MEM 空间
Memory	64'h8000_0000	64'hFFFF_FFFF_8000_0000	2G	内存空间
Memory	64'h1_0000_0000	64'hFFFF_FFFF_0000_0000	4G	内存空间
Memory	64'h2_0000_0000	64'hFFFF_FFFE_0000_0000	8G	内存空间
Memory mapped IO	64'h40_0000_0000	64'hFFFF_FFC0_0000_0000	256G	PCIE MEM 空间
Type0	64'hFE_0000_0000	64'hFFFF_FFFF_FF00_0000	256M	Type0 配置空间
Type1	64'hFE_1000_0000	64'hFFFF_FFFF_FF00_0000	256M	Type1 配置空间
Type0	64'hFE_2000_0000	64'hFFFF_FFFF_FF00_0000	256M	Type0 配置空间
Type1	64'hFE_3000_0000	64'hFFFF_FFFF_FF00_0000	256M	Type1 配置空间

## 6.1 一级交叉开关

一级交叉开关一共有 3 个主端口，分别连接 CORE0、CORE1 和 IO DMA。

一级开关的路由不可配置，具体的地址空间分配如下表所示。其中包含了 32 位和 64 位地址空间访问模式。

表 6-2 一级交叉开关路由规则

地址空间	大小	设备/路由目的空间	备注
0x1000_0000 - 0x17FF_FFFF	128M	I/O 设备的配置寄存器空间	需注意该段地址空间为 MEM 类型。各个 IO 设备的 BAR 地址在该地址空间。
0x1800_0000 - 0x19FF_FFFF	32M	PCIE 的 I/O 空间	32 位模式
0x1A00_0000 - 0x1BFF_FFFF	32M	各个 I/O 设备的配置头空间	32 位模式
0x1FC0_0000 - 0x1FCF_FFFF	1M	启动空间	BOOT
0x1FE0_0000 - 0x1FEF_FFFF	1M	Confbus	芯片配置空间
0x4000_0000 - 0x7FFF_FFFF	1G	32 位访问模式下 I/O 设备的 MEM 空间	32 位模式
0x40_0000_0000 - 0x4F_FFFF_FFFF	256G	64 位访问模式下 I/O 设备的 MEM 空间	64 位模式
0xFE_0000_0000 - 0xFE_FFFF_FFFF	4G	各个 I/O 设备的配置头空间	64 位模式
0xFD_FC00_0000 - 0xFD_FDFF_FFFF	32M	PCIE 的 I/O 空间	64 位模式
其他地址或上述空间若为 CACHED		Bit6 为 0，路由到 scache0； Bit6 为 1，路由到 scache1	由地址的第 6 位决定路由目的

## 6.2 二级交叉开关

龙芯 2K1000 的二级交叉开关连接两个二级 Cache、内存控制器、启动模块（SPI 或者 LIO）以及 IO 子网络（Uncache 访问路径），如下图所示：

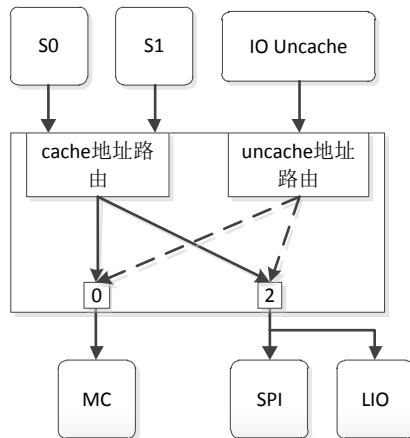


图 6-1 二级交叉开关地址路由示意图

地址窗口是供 SCACHE0、SCACHE1 和 IO DMA 三个具有主功能的 IP 进行路由选择和地址转换而设置的。CPU 和 IO DMA 的访问都拥有 8 个地址窗口，可以完成目标地址空间

的选择以及从源地址空间到目标地址空间的转换。每个地址窗口由 BASE、MASK 和 MMAP 三个 64 位寄存器组成，BASE 以 K 字节对齐，MASK 采用类似网络掩码高位为 1 的格式，MMAP 的低三位是路由选择。

在二级交叉开关处，标号与所述模块的对应关系如下表所示。

表 6-3 二级交叉开关处标号与所述模块的对应关系

标号	缺省值
0	DDR3 内存控制器
1	保留
2	启动模块（SPI 或 LIO）
3	保留

MMAP[4]表示允许取指，MMAP[5]表示允许块读，MMAP[7]表示窗口使能。

表 6-4 MMAP 字段对应的该空间访问属性

[4]	[5]	[7]
允许取指	允许块读	窗口使能

二级交叉开关的地址配置具有地址转换的功能。不需要维护 CACHE 一致性。

窗口命中公式:  $(IN\_ADDR \& MASK) == BASE$

新地址换算公式:  $OUT\_ADDR = (IN\_ADDR \& \sim MASK) | \{MMAP[63:10], 10'h0\}$

地址窗口转换寄存器如下表。

表 6-5 二级交叉开关地址窗口转换寄存器表

地址	寄存器	R/W	描述	缺省值
0x1fe10000	CPU_WIN0_BASE	RW	cache 通路二级窗口 0 的基址	0x1FC0_0000
0x1fe10008	CPU_WIN1_BASE	RW	cache 通路二级窗口 1 的基址	0x1000_0000
0x1fe10010	CPU_WIN2_BASE	RW	cache 通路二级窗口 2 的基址	0x0000_0000
0x1fe10018	CPU_WIN3_BASE	RW	cache 通路二级窗口 3 的基址	0x1_0000_0000
0x1fe10020	CPU_WIN4_BASE	RW	cache 通路二级窗口 4 的基址	0x0000_0000
0x1fe10028	CPU_WIN5_BASE	RW	cache 通路二级窗口 5 的基址	0x0000_0000
0x1fe10030	CPU_WIN6_BASE	RW	cache 通路二级窗口 6 的基址	0x0000_0000
0x1fe10038	CPU_WIN7_BASE	RW	cache 通路二级窗口 7 的基址	0x0000_0000
0x1fe10040	CPU_WIN0_MASK	RW	cache 通路二级窗口 0 的掩码	0xFFFF_FFFF_FFF0_0000
0x1fe10048	CPU_WIN1_MASK	RW	cache 通路二级窗口 1 的掩码	0xFFFF_FFFF_F000_0000
0x1fe10050	CPU_WIN2_MASK	RW	cache 通路二级窗口 2 的掩码	0xFFFF_FFFF_F000_0000
0x1fe10058	CPU_WIN3_MASK	RW	cache 通路二级窗口 3 的掩码	0xFFFF_FFFF_0000_0000
0x1fe10060	CPU_WIN4_MASK	RW	cache 通路二级窗口 4 的掩码	0x0000_0000

地址	寄存器	R/W	描述	缺省值
			掩码	
0x1fe10068	CPU_WIN5_MASK	RW	cache 通路二级窗口 5 的掩码	0x0000_0000
0x1fe10070	CPU_WIN6_MASK	RW	cache 通路二级窗口 6 的掩码	0x0000_0000
0x1fe10078	CPU_WIN7_MASK	RW	cache 通路二级窗口 7 的掩码	0x0000_0000
0x1fe10080	CPU_WIN0_MMAP	RW	cache 通路二级窗口 0 的新基址	0x1FC0_00F2
0x1fe10088	CPU_WIN1_MMAP	RW	cache 通路二级窗口 1 的新基址	0x1000_0082
0x1fe10090	CPU_WIN2_MMAP	RW	cache 通路二级窗口 2 的新基址	0x0000_00F0
0x1fe10098	CPU_WIN3_MMAP	RW	cache 通路二级窗口 3 的新基址	0x0000_00F0
0x1fe100a0	CPU_WIN4_MMAP	RW	cache 通路二级窗口 4 的新基址	0x0000_0000
0x1fe100a8	CPU_WIN5_MMAP	RW	cache 通路二级窗口 5 的新基址	0x0000_0000
0x1fe100b0	CPU_WIN6_MMAP	RW	cache 通路二级窗口 6 的新基址	0x0000_0000
0x1fe100b8	CPU_WIN7_MMAP	RW	cache 通路二级窗口 7 的新基址	0x0000_0000
0x1fe10100	PCI_WIN0_BASE	RW	uncache 通路二级窗口 0 的基址	0x0000_0000
0x1fe10108	PCI_WIN1_BASE	RW	uncache 通路二级窗口 1 的基址	0x0000_0000
0x1fe10110	PCI_WIN2_BASE	RW	uncache 通路二级窗口 2 的基址	0x0000_0000
0x1fe10118	PCI_WIN3_BASE	RW	uncache 通路二级窗口 3 的基址	0x0000_0000
0x1fe10120	PCI_WIN4_BASE	RW	uncache 通路二级窗口 4 的基址	0x0000_0000
0x1fe10128	PCI_WIN5_BASE	RW	uncache 通路二级窗口 5 的基址	0x0000_0000
0x1fe10130	PCI_WIN6_BASE	RW	uncache 通路二级窗口 6 的基址	0x0000_0000
0x1fe10138	PCI_WIN7_BASE	RW	uncache 通路二级窗口 7 的基址	0x0000_0000
0x1fe10140	PCI_WIN0_MASK	RW	uncache 通路二级窗口 0 的掩码	0xFFFF_FFFF_F000_0000
0x1fe10148	PCI_WIN1_MASK	RW	uncache 通路二级窗口 1 的掩码	0x0000_0000
0x1fe10150	PCI_WIN2_MASK	RW	uncache 通路二级窗口 2 的掩码	0x0000_0000
0x1fe10158	PCI_WIN3_MASK	RW	uncache 通路二级窗口 3 的掩码	0x0000_0000
0x1fe10160	PCI_WIN4_MASK	RW	uncache 通路二级窗口 4 的掩码	0x0000_0000
0x1fe10168	PCI_WIN5_MASK	RW	uncache 通路二级窗口 5 的掩码	0x0000_0000
0x1fe10170	PCI_WIN6_MASK	RW	uncache 通路二级窗口 6 的掩码	0x0000_0000
0x1fe10178	PCI_WIN7_MASK	RW	uncache 通路二级窗口 7 的掩码	0x0000_0000

地址	寄存器	R/W	描述	缺省值
0x1fe10180	PCI_WIN0_MMAP	RW	uncache 通路二级窗口 0 的新基址	0x0000_00F0
0x1fe10188	PCI_WIN1_MMAP	RW	uncache 通路二级窗口 1 的新基址	0x0000_0000
0x1fe10190	PCI_WIN2_MMAP	RW	uncache 通路二级窗口 2 的新基址	0x0000_0000
0x1fe10198	PCI_WIN3_MMAP	RW	uncache 通路二级窗口 3 的新基址	0x0000_0000
0x1fe101a0	PCI_WIN4_MMAP	RW	uncache 通路二级窗口 4 的新基址	0x0000_0000
0x1fe101a8	PCI_WIN5_MMAP	RW	uncache 通路二级窗口 5 的新基址	0x0000_0000
0x1fe101b0	PCI_WIN6_MMAP	RW	uncache 通路二级窗口 6 的新基址	0x0000_0000
0x1fe101b8	PCI_WIN7_MMAP	RW	uncache 通路二级窗口 7 的新基址	0x0000_0000

根据缺省的寄存器配置，芯片启动后，CPU 的 0x1FC0\_0000-0x1FCF\_FFFF 的地址区间（1M）映射到 BOOT 设备（SPI 或 LIO）存储空间的 0x00000000-0x000F\_FFFF 区间，CPU 的 0x10000000 - 0x1FFFFFFF 区间（256M）映射到 BOOT 设备的 0x10000000 - 0x1FFFFFFF 区间，CPU 的 0x00000000 - 0x0FFFFFFF 的地址区间（256M）映射到 DDR3 的 0x00000000 - 0x0FFFFFFF 的地址区间，CPU 的 0x100000000 - 0x1FFFFFFF 的地址区间（4G）映射到 DDR3 的 0x00000000 - 0xFFFFFFF 的地址区间。软件可以通过修改相应的配置寄存器实现新的地址空间路由和转换。

此外，当出现 8 个地址窗口都不命中时，由内存控制器模块返回数据。

### 6.3 IO 互连网络

2K1000 中的 IO 互连网络结构图如图 6-2 所示。为了便于说明，这里把 IO 设备分成 PCIE 设备（包括 PCIE0 与 PCIE1）和其他 IO 设备（除 PCIE 之外的所有其他设备）两部分。

IO 互连网络主要由以下两部分通路组成：

1. CPU 访问 IO 设备的通路，其地址路由形式主要包括：

- 配置访问：CPU 通过配置请求访问 IO 设备的配置头，其地址组织形式在 6.3.1 节有介绍；
- IO 访问：IO 访问用于访问 PCIE 控制器下游设备的 IO 地址空间，根据 PCIE 配置头的 IO Base 和 IO Limit 配置决定地址路由方式；
- 内部寄存器访问：通过各个 IO 设备配置头的 BAR 地址访问 IO 设备的内部配置寄存器，MEM 访问类型。
- PCIE 下游设备 MEM 访问：根据 PCIE 配置头的 Memory Base 和 Memory Limit、Prefetchable Memory Base 和 Prefetchable Memory Limit 进行地址路由，用于访问 PCIE 下游设备的 MEM 空间。



2. IO DMA 访问内存的通路，包括 CACHE 访问和 UNCACHE 访问。该部分地址路由方式在 6.4 节介绍。

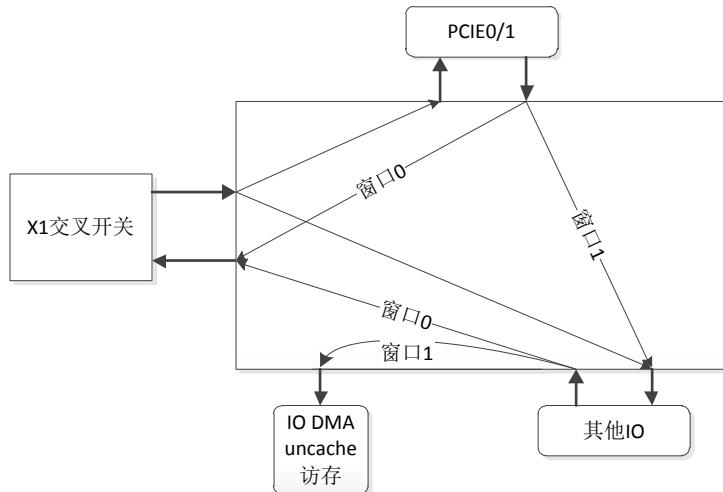


图 6-2IO 互连结构图

### 6.3.1 IO 设备的配置空间

这里需要先区分配置头空间和设备寄存器空间两个概念。配置头空间指的是每个 IO 设备的配置头所在的地址空间，而设备寄存器空间指的是配置头中的 BAR 所指定的设备配置寄存器的地址空间。访问 I/O 设备的设备寄存器空间需要先读取其配置头空间中的 BAR 地址作为其配置空间的起始地址，然后再对相应设备寄存器进行读写。

首先介绍 64 位模式下访问配置头空间的地址格式，龙芯 2K1000 中定义地址段 0xFE\_0000\_0000 – 0xFE\_1FFF\_FFFF 是 CPU 的 64 位配置地址空间，CPU 通过这段地址访问各个设备的 PCI 配置头。由地址的[63:28]决定配置头类型（0xFE0 是 Type0，0xFE1 是 Type1，其他保留）；[23:16]在 Type1 类型配置头访问时表示总线号（Bus Number），Type0 时保留；[15:11]表示设备号（Device Number）；[10:8]表示功能号（Function Number）；[27:24]和[7:0]组合起来表示偏移（offset），大小为 4K。下图是 CPU 访问 PCI 配置头空间的 64 位地址格式。

Type 0	39	32 31	28 27	24 23	16 15	11 10	8 7	0
		FE0h		Offset[11:8]	Reserved	Device Number	Function Number	Offset[7:0]
Type 1	39	32 31	28 27	24 23	16 15	11 10	8 7	0
		FE1h		Offset[11:8]	Bus Number	Device Number	Function Number	Offset[7:0]

图 6-364 位配置访问地址格式

为了保证系统的兼容性，还提供了 32 位地址模式下的配置地址格式，如下图所示。该模式下，地址偏移只有 8 位（即 256B）。其中，地址的[31:24]决定配置头类型（0x1A 是 Type0，0x1B 是 Type1）；[23:16]在 Type1 类型配置头访问时表示总线号（Bus Number），Type0 时保留；[15:11]表示设备号（Device Number）；[10:8]表示功能号（Function Number）；

[7:0]表示偏移 (offset)，大小为 256B。

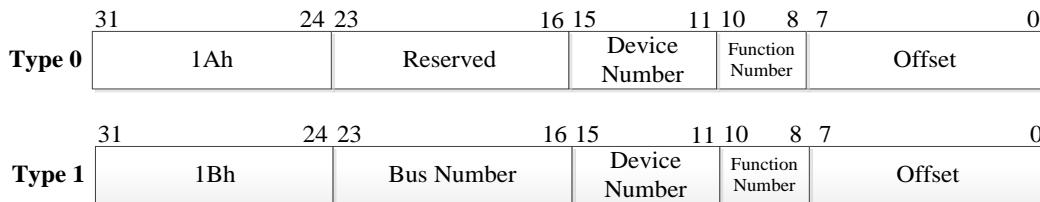


图 6-432 位配置访问地址格式

龙芯 2K1000 芯片中，具备配置头空间的设备包括：APB 总线控制器、GMAC0/1 控制器、USB 控制器、GPU 控制器、DC 控制器、HDA 控制器、SATA 控制器、PCIE0/1 控制器和 DMA 控制器。其中所有 APB 设备都作为一个整体挂在 APB 总线控制器上，由地址的[15:12]位进行译码，下文中将会做详细说明。各个设备的设备号、功能号以及配置头访问首地址如下表所示（该表只包括各个控制器本身的配置空间，不关注 PCIE 控制器的 Type1 访问等）。除了 PCIE 控制器之外，其他设备的配置头空间大小都只有 256B。

表 6-6 各个设备的配置头访问对应关系

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)	备注
<b>APB</b>	0x0	0x2	0x0	0xffff	0xFE_0000_1000	-
<b>GMAC0</b>	0x0	0x3	0x0	0x7ff	0xFE_0000_1800	GMAC 设备有 2 个功能，Func 0 是 GMAC0
<b>GMAC1</b>	0x0	0x3	0x1	0x7ff	0xFE_0000_1900	GMAC 设备有 2 个功能，Func 1 是 GMAC1
<b>USB-OTG</b>	0x0	0x4	0x0	0xffff	0xFE_0000_2000	USB 设备共 3 个功能：Func 0 是 OTG
<b>USB-EHCI</b>	0x0	0x4	0x1	0x7ff	0xFE_0000_2100	USB 设备共 3 个功能：Func 1 是 EHCI
<b>USB-OHCI</b>	0x0	0x4	0x2	0x7ff	0xFE_0000_2200	USB 设备共 3 个功能：Func 2 是 OHCI
<b>GPU</b>	0x0	0x5	0x0	0x3ffff	0xFE_0000_2800	-
<b>DC</b>	0x0	0x6	0x0	0xffff	0xFE_0000_3000	-
<b>HDA</b>	0x0	0x7	0x0	0xffff	0xFE_0000_3800	-
<b>SATA</b>	0x0	0x8	0x0	0xffff	0xFE_0000_4000	-
<b>PCIE0-Port0</b>	0x0	0x9	0x0	0xfff	0xFE_0000_4800	Func 1 和 Type1 的访问没有在此表中说明。 当 PCIE0 控制器为 X4 模式时，只能访问设备号为 0x9 的控制器。
<b>PCIE0-Port1</b>	0x0	0xa	0x0	0xfff	0xFE_0000_5000	
<b>PCIE0-Port2</b>	0x0	0xb	0x0	0xfff	0xFE_0000_5800	
<b>PCIE0-Port3</b>	0x0	0xc	0x0	0xfff	0xFE_0000_6000	
<b>PCIE1-Port0</b>	0x0	0xd	0x0	0xfff	0xFE_0000_6800	Func 1 和 Type1 的访问没有在此表中说明。 当 PCIE1 控制器为 X4 模式时，只能访问设备号为 0xd 的控制器。
<b>PCIE1-Port1</b>	0x0	0xe	0x0	0xfff	0xFE_0000_7000	
<b>DMA</b>	0x0	0xf	0x0	0xff	0xFE_0000_7800	-
<b>N/A</b>	无效的配置头访问					Vendor ID 为 16'hFFFF，表示没有有效设备

除 PCIE 外，所有的配置头空间都遵循 PCI Type0I 类型的格式，如下表所示，但是每个设备的缺省值不尽相同，后文中将展开介绍。

表 6-7 Type0I 类型配置头

31	24	23	16	15	8	7	0	ADDR						
Device ID			Vendor ID				00h							
Status			Command				04h							
Class Code				Revison ID			08h							
BIST	Header Type		Lantency Timer		Cache Line Size		0Ch							
Base Address 0							10h							
Base Address 1							14h							
Base Address 2							18h							
Base Address 3							1Ch							
Base Address 4							20h							
Base Address 5							24h							
CardBus CIS Pointer							28h							
Subsystem ID		Subsystem Vendor ID					2Ch							
Expansion ROM Base Address							30h							
Reserved				Capabilities Pointer			34h							
Reserved							38h							
Max_Lat	Min_Gnt	Interrupt Pin		Interrupt Line		3Ch								

表 6-8 Type0I 的配置头寄存器

寄存器名称	偏移地址	读/写	功能描述	说明 (除 PCIE 控制器)
Vendor ID 制造商 ID 号	0x0	RO	制造商 ID 号	缺省值都为 0x0014
Device ID 设备 ID 号	0x2	RO	设备 ID 号	
Command 命令寄存器	0x4	RW	Bit0: I/O 访问使能 Bit1: 存储器访问使能 Bit2: 主设备使能 Bit3: 专用周期监视 Bit4: 存储器写与使失效使能 Bit5: VGA 画板侦测	扫描配置地址后，需要使能相应设备的存储器访问使能



寄存器名称	偏移地址	读/写	功能描述	说明 (除 PCIE 控制器)
			Bit6: 奇偶校验错响应 Bit7: 等待周期控制 Bit8: SERR#使能 Bit9: 快速背靠背使能 其它: 保留	
Status 状态寄存器	0x6	RO/RW	Bit3-0: RO, 保留 Bit4: RO, 能力列表 Bit5: RO, 66MHz 能力 Bit6: 保留 Bit7: RO, 快速背靠背能力 Bit8: RW, 主设备数据奇偶校验错 Bit10-9: RO, 设备选择定时 Bit11: RW, 发出目标失败信号 Bit12: RW, 收到目标失败 Bit13: RW, 收到主设备失败 Bit14: RW, 发出系统错信号 Bit15: RW, 检测到奇偶错	
Revision ID 版本索引	0x8	RO	设备版本号	缺省值都为 0x0
Class Code 类代码	0x9	RO	Bit7-0: Prog.I/F 编程接口 Bit15-8: Subclass code, 子类码 Bit23-16: Class code 类别码	
Cache Line Size 缓存行大小	0xC	RO	龙芯 2K1000 的缓存行大小为 64Byte	缺省值都为 0x10
Latency Timer 等待计时器	0xD	RW		保留
Header Type 配置头类型	0xE	RO	除 PCIE 控制器外, 都是 Type0。 另外 bit7 表示多功能属性: 1-多功能设备 0-单功能设备	
BIST 内自建测试	0xF	RW	Bit3-0: 完成代码, 0-成功完成, 非 0-设备指定的错误 Bit5-4: 保留 Bit6: 起动 BIST Bit7: BIST 能力, 1- 提供 BIST 功能, 0-不提供	保留

寄存器名称	偏移地址	读/写	功能描述	说明 (除 PCIE 控制器)
Base Address Register 0 0 号基址寄存器	0x10	RW/RO	基地址寄存器 0- Bit0: RO, 1-I/O 基地址寄存器, 0-MEM 基地址寄存器  I/O 基地址寄存器: Bit1: 保留 Bit31-2: RO, 基地址单元 Bit63-32: 保留  MEM 基地址存储器: Bit2-1: RO, MEM 基地址寄存器-译码器宽度单元, 00-32 位, 10-64 位 Bit3: RO, 预提取属性 Bit64-4: 基地址单元	只支持 64 位模式
Base Address Register 1 1 号基址寄存器	0x18	RW	基址寄存器 1	只支持 64 位模式
Base Address Register 2 2 号基址寄存器	0x20	RW	基址寄存器 2	只支持 64 位模式
Cardbus CIS Pointer CIS 卡总线指针	0x28	RW	CIS 卡总线指针	保留
Subsystem Vendor ID 子系统版本号	0x2C	RO	子系统供应商 ID 号	保留
Subsystem ID 子系统版本号	0x2D	RO	子系统版本 ID 号	保留
Expansion ROM Base Address 扩展 ROM 基地址	0x30	RW	扩展 ROM 基地址寄存器 Bit0: 地址译码使能 Bit10-1: 保留 Bit31-11: 用于指定 ROM 的起始地址	保留
Capabilities Pointer 扩展能力指针	0x34	RW	扩展指针	保留
Reserved	0x35			保留
Reserved	0x38			保留
Interrupt Line 中断线寄存器	0x3C	RW	中断线寄存器	
Interrupt Pin 中断引脚寄存器	0x3D	RO	中断引脚寄存器	保留 (除 PCIE 控制器)
Min Gnt 最小突发期时间	0x3E	RW		保留 (除 PCIE 控制器)
Max Lat 获取 PCI 总线最大时间	0x3F	RW		保留 (除 PCIE 控制器)

PCIE 控制器的配置头为 Type1 类型，具体如下：

表 6-9 Type1 类型配置头

31	24	23	16	15	8	7	0	ADDR
Device ID				Vendor ID				00h
Status				Command				04h

Class Code			Revison ID	08h
BIST	Header Type	Lantency Timer	Cache Line Size	0Ch
Base Address 0			10h	
Base Address 1			14h	
Secondary Lantency Timer	Subordinate Bus #	Secondary Bus #	Primary Bus #	18h
Secondary Status		IO Limit	IO Base	1Ch
(Non-Prefetchable) Memory Limit		(Non-Prefetchable) Memory Base		20h
PrefetchableMemory Limit		PrefetchableMemory Base		24h
Prefetchable Memory Base Upper 32 Bits			28h	
Prefetchable Memory Limit Upper 32 Bits			2Ch	
IO Limit Upper 16 Bits	IO Base Upper 16 Bits			30h
Reserved			Capabilities Pointer	34h
Expansion ROM Base Address			38h	
Bridge Control	Interrupt Pin	Interrupt Line	3Ch	

表 6-10 Type1 的配置头寄存器

寄存器名称	偏移地址	读/写	功能描述	说明 (PCIE 控制器)
Vendor ID 制造商 ID 号	0x0	RO	制造商 ID 号	缺省值都为 0x0014
Device ID 设备 ID 号	0x2	RO	设备 ID 号	
Command 命令寄存器	0x4	RW	Bit0: I/O 访问使能 Bit1: 存储器访问使能 Bit2: 主设备使能 Bit3: 专用周期监视 Bit4: 存储器写与使失效使能 Bit5: VGA 画板侦测 Bit6: 奇偶校验错响应 Bit7: 等待周期控制 Bit8: SERR#使能 Bit9: 快速背靠背使能 其它: 保留	扫描配置地址后, 需要使能相应设备的存储器访问使能
Status 状态寄存器	0x6	RO/RW	Bit3-0: RO, 保留 Bit4: RO, 能力列表 Bit5: RO, 66MHz 能力 Bit6: 保留 Bit7: RO, 快速背靠背能力 Bit8: RW, 主设备数据奇偶校验错 Bit10-9: RO, 设备选择定时	

寄存器名称	偏移地址	读/写	功能描述	说明 (PCIE 控制器)
			Bit11: RW, 发出目标失败信号 Bit12: RW, 收到目标失败 Bit13: RW, 收到主设备失败 Bit14: RW, 发出系统错信号 Bit15: RW, 检测到奇偶错	
Revision ID 版本索引	0x8	RO	设备版本号	缺省值都为 0x0
Class Code 类代码	0x9	RO	Bit7-0: Prog.I/F 编程接口 Bit15-8: Subclass code, 子类码 Bit23-16: Class code 类别码	缺省值为 0x0b3000
Cache Line Size 缓存行大小	0xC	RO	缓存行大小	缺省值都为 0x00
Latency Timer 等待计时器	0xD	RW		保留
Header Type 配置头类型	0xE	RO	PCIE 控制器都是 Type1	缺省值为 0x01
BIST 内自建测试	0xF	RW	Bit3-0: 完成代码, 0-成功完成, 非 0-设备指定的错误 Bit5-4: 保留 Bit6: 起动 BIST Bit7: BIST 能力, 1-提供 BIST 功能, 0-不提供	保留
Base Address Register 0	0x10	RW/RO	基址寄存器 0- Bit0: RO, 1-I/O 基址寄存器, 0-MEM 基址寄存器  I/O 基址寄存器: Bit1: 保留 Bit31-2: RO, 基址单元 Bit63-32: 保留  MEM 基址存储器: Bit2-1: RO, MEM 基址寄存器-译码器宽度单元, 00-32 位, 10-64 位 Bit3: RO, 预提取属性 Bit64-4: 基址单元	64 位 MEM 空间
Primary Bus #	0x18	RW	Primary Bus 编号	默认为 0
Secondary Bus #	0x19	RW	Secondary Bus 编号	默认为 0
Subordinate Bus #	0x1A	RW	Subordinate Bus 编号	默认为 0
IO Base	0x1C	RW/RO	[3:0] RO, 0h-16bit,1h-32bit [7:4] RW	默认为 0x1
IO Limit	0X1D	RW/R0	[3:0] RO, 0h-16bit,1h-32bit [7:4] RW	默认为 0x1

寄存器名称	偏移地址	读/写	功能描述	说明 (PCIE 控制器)
Secondary Lantency Timer	0x1E	RO	Secondary Bus 状态寄存器	默认为 0
(Non-Prefetchable) Memory Base	0x20	RW/R0	[3:0] RO, must be 0 [15:4] RW	默认为 0x0
(Non-Prefetchable) Memory Limit	0x22	RW/R0	[3:0] RO, must be 0 [15:4] RW	默认为 0x0
Prefetchable Memory Base	0x24	RW/R0	[3:0] RO, 0h-32bit, 1h-64bit [15:4] RW	默认为 0x1
Prefetchable Memory Limit	0x26	RW/R0	[3:0] RO, 0h-32bit, 1h-64bit [15:4] RW	默认为 0x1
Prefetchable Memory Base Upper 32bits	0x28	RW	可预取 MEM 地址空间高 32 位	默认为 0
Prefetchable Memory Limit Upper 32bits	0x2C	RW	可预取 MEM 地址空间高 32 位	默认为 0
IO Base Upper 16 Bits	0x30	RW	IO 地址 Base 高 16 位	默认为 0
IO Limit Upper 16 Bits	0x32	RW	IO 地址 Limit 高 16 位	默认为 0
Capabilities Pointer 扩展能力指针	0x34	RW	扩展指针	默认为 0x40
Expansion ROM Base Address 扩展 ROM 基地址	0x38	RW	扩展 ROM 基地址寄存器 Bit0: 地址译码使能 Bit10-1: 保留 Bit31-11: 用于指定 ROM 的起始地址	保留
Interrupt Line 中断线寄存器	0x3C	RW	中断线寄存器	默认为 0xFF
Interrupt Pin 中断引脚寄存器	0x3D	RO	中断引脚寄存器	默认为 0x0
Bridge Control	0x3E	RW/RO	0(RW): Parity Error Response Enable 1(RW): SERR# Enable 2(RW): ISA Enable 3(RW): VGA Enable 4(RW): VGA 16bit Decode 5(RW): Master-Abort Mode 6(RW) Secondary Bus Reset 7(RW): Fast Back-to-Back Enable 8(RO): Primary Discard Timer 9: Secondary Discard Timer 10(RW1C): Discard Timer Status 11(RW): Discard Timer SERR# Enable 15:12 Reserved	默认为 0

### 6.3.2 APB 配置头

APB 总线控制器下挂载多个 APB 设备。该总线控制器作为一个设备具有相应的配置头空间。其配置头访问的基址为 0xFE\_0000\_1000。

表 6-11 APB 总线控制器的配置头缺省值

31	24	23	16	15	8	7	0	ADDR
			16'h7a02		16'h0014		00h	
			保留		保留		3'h0	04h

8'h 08	8'h80	8'h0	保留	08h
保留	8'h0	保留	8'h10	0Ch
24'h0			8'h4	10h
	32'h0			14h
	保留			18h
	保留			1Ch
	保留			20h
	保留			24h
	保留			28h
	保留			2Ch
	保留			30h
	保留			34h
	保留			38h
保留		0xFF		3Ch

### 6.3.3 GMAC0/1 配置头

GMAC0 控制器和 GMAC1 控制器的配置头缺省值一致。其配置头访问的基址分别为 0xFE\_0000\_1800 和 0xFE\_0000\_1900。

表 6-12 GMAC0 控制器的配置头缺省值

31	24	23	16	15	8	7	0	ADDR
		16'h7a03			16'h0014			00h
		保留			保留		3'h0	04h
	8'h02		8'h00		8'h00		保留	08h
	保留		8'h0		保留		8'h10	0Ch
		24'h0				8'h4		10h
			32'h0					14h
			保留					18h
			保留					1Ch
			保留					20h
			保留					24h
			保留					28h
			保留					2Ch
			保留					30h
			保留					34h
			保留					38h
		保留			0xFF			3Ch

### 6.3.4 USB OTG 配置头

该设备的配置头访问的基址为 0xFE\_0000\_2000。

表 6-13 USB-OTG 控制器的配置头缺省值

31	24	23	16	15	8	7	0	ADDR
		16'h7a04			16'h0014			00h
		保留			保留		3'h0	04h
	8'h 0c		8'h03		8'h80		保留	08h
	保留		8'h80		保留		8'h10	0Ch
		24'h0				8'h4		10h
			32'h0					14h
			保留					18h
			保留					1Ch
			保留					20h
			保留					24h
			保留					28h

保留		2Ch
保留		30h
保留		34h
保留		38h
保留	0xFF	3Ch

### 6.3.5 USB-EHCI 配置头

该设备的配置头访问的基址为 0xFE\_0000\_2100。

表 6-14 USB-EHCI 控制器的配置头缺省值

31	24	23	16	15	8	7	0	ADDR
		16'h7a14			16'h0014			00h
		保留			保留		3'h0	04h
	8'h 0c		8'h03		8'h20		保留	08h
	保留		8'h80		保留		8'h10	0Ch
			24'h0				8'h4	10h
				32'h0				14h
				保留				18h
				保留				1Ch
				保留				20h
				保留				24h
				保留				28h
				保留				2Ch
				保留				30h
				保留				34h
				保留				38h
				保留			0xFF	3Ch

### 6.3.6 USB OHCI 配置头

该设备的配置头访问的基址为 0xFE\_0000\_2200。

表 6-15 USB-OHCI 控制器的配置头缺省值

31	24	23	16	15	8	7	0	ADDR
		16'h7a24			16'h0014			00h
		保留			保留		3'h0	04h
	8'h 0c		8'h03		8'h10		保留	08h
	保留		8'h80		保留		8'h10	0Ch
			24'h0				8'h4	10h
				32'h0				14h
				保留				18h
				保留				1Ch
				保留				20h
				保留				24h
				保留				28h
				保留				2Ch
				保留				30h
				保留				34h
				保留				38h
				保留			0xFF	3Ch

### 6.3.7 GPU 配置头

该设备的配置头访问的基址为 0xFE\_0000\_2800。

表 6-16 GPU 控制器的配置头缺省值

31	24	23	16	15	8	7	0	ADDR
			16'h7a05			16'h0014		00h
		保留			保留		3'h0	04h
	8'h 03		8'h02		8'h00		保留	08h
	保留		8'h0		保留		8'h10	0Ch
			24'h0			8'h4		10h
			32'h0					14h
			保留					18h
			保留					1Ch
			保留					20h
			保留					24h
			保留					28h
			保留					2Ch
			保留					30h
			保留					34h
			保留					38h
		保留			0xFF			3Ch

### 6.3.8 DC 配置头

该设备的配置头访问的基址为 0xFE\_0000\_3000。

表 6-17 DC 控制器的配置头缺省值

31	24	23	16	15	8	7	0	ADDR
			16'h7a06			16'h0014		00h
		保留			保留		3'h0	04h
	8'h 03		8'h00		8'h00		保留	08h
	保留		8'h0		保留		8'h10	0Ch
			24'h0			8'h4		10h
			32'h0					14h
			保留					18h
			保留					1Ch
			保留					20h
			保留					24h
			保留					28h
			保留					2Ch
			保留					30h
			保留					34h
			保留					38h
		保留			0xFF			3Ch

### 6.3.9 HDA 配置头

该设备的配置头访问的基址为 0xFE\_0000\_3800。

表 6-18 HDA 控制器的配置头缺省值

31	24	23	16	15	8	7	0	ADDR
			16'h7a07			16'h0014		00h
		保留			保留		3'h0	04h
	8'h 04		8'h03		8'h00		保留	08h
	保留		8'h0		保留		8'h10	0Ch
			24'h0			8'h4		10h
			32'h0					14h
			保留					18h
			保留					1Ch

31	24	23	16	15	8	7	0	ADDR
				保留				20h
				保留				24h
				保留				28h
				保留				2Ch
				保留				30h
				保留				34h
				保留				38h
				保留		0xFF		3Ch

### 6.3.10 SATA 配置头

该设备的配置头访问的基址为 0xFE\_0000\_4000。

表 6-19 SATA 控制器的配置头缺省值

31	24	23	16	15	8	7	0	ADDR
			16'h7a08			16'h0014		00h
			保留		保留		3'h0	04h
		8'h 01	8'h06	8'h01		保留		08h
		保留	8'h0	保留		8'h10		0Ch
			24'h0			8'h4		10h
				32'h0				14h
				保留				18h
				保留				1Ch
				保留				20h
				保留				24h
				保留				28h
				保留				2Ch
				保留				30h
				保留				34h
				保留				38h
				保留		0xFF		3Ch

### 6.3.11 PCIE 配置头

龙芯 2K1000 中的两个 PCIE 共包含 6 个端口 (Port)，每个 Port 内都有一个 TYPE1 类型的配置头。其中

PCIE0 Port0 的配置头基址为 0xFE\_0000\_4800;

PCIE0 Port1 的配置头基址为 0xFE\_0000\_5000;

PCIE0 Port2 的配置头基址为 0xFE\_0000\_5800;

PCIE0 Port3 的配置头基址为 0xFE\_0000\_6000;

PCIE1 Port0 的配置头基址为 0xFE\_0000\_6800;

PCIE1 Port1 的配置头基址为 0xFE\_0000\_7000;

下表列出 PCIE0 Port0 的配置头缺省值，其他 Port 与 Port0 相比仅有 Device ID 的差别，Port1/2/3 的 Device ID 都是 16'h7a09。同样的，PCIE1 port0 的 Device ID 为 16'h7a19，Port1 的 Device ID 为 16'h7a09。

表 6-20PCIE0 Port0 的配置头缺省值

31	24	23	16	15	8	7	0	ADDR
			16'h7a19			16'h0014		00h

31	24	23	16	15	8	7	0	ADDR			
			16'h0010		16'h0000			04h			
			24'h0b0300			8'h01		08h			
8'h00		8'h01		8'h00		8'h00		0Ch			
		32'h4						10h			
		32'h0						14h			
8'h00		8'h00		8'h00		8'h00		18h			
16'h0			8'h01		8'h01			1Ch			
16'h0			16'h0					20h			
16'h0			16'h0					24h			
		32'h0						28h			
		32'h0						2Ch			
32'h0			32'h0					30h			
		Reserved			8'h40			34h			
		32'h0						38h			
16'h0			8'h01		0xFF			3Ch			

### 6.3.12 DMA 配置头

该设备的配置头访问的基址为 0xFE\_0000\_7800。

表 6-21 DMA 控制器的配置头缺省值

31	24	23	16	15	8	7	0	ADDR			
			16'h7a0f		16'h0014			00h			
			保留					04h			
8'h 08		8'h80		8'h00		保留		08h			
保留		8'h0		保留		8'h10		0Ch			
		24'h0		8'h4				10h			
		32'h0						14h			
		保留						18h			
		保留						1Ch			
		保留						20h			
		保留						24h			
		保留						28h			
		保留						2Ch			
		保留						30h			
		保留						34h			
		保留						38h			
保留			0xFF					3Ch			

### 6.3.13 VPU 配置头

该设备的配置头访问的基址为 0xFE\_0000\_8000。

表 6-22 VPU 解码器的配置头缺省值

31	24	23	16	15	8	7	0	ADDR			
			16'h7a16		16'h0014			00h			
			保留					04h			
8'h 04		8'h80		8'h00		保留		08h			
保留		8'h0		保留		8'h10		0Ch			
		24'h0		8'h4				10h			
		32'h0						14h			
		保留						18h			
		保留						1Ch			
		保留						20h			
		保留						24h			
		保留						28h			

31	24	23	16	15	8	7	0	ADDR
				保留				2Ch
				保留				30h
				保留				34h
				保留				38h
				保留		0xFF		3Ch

### 6.3.14 CAMERA 配置头

该设备的配置头访问的基址为 0xFE\_0000\_8800。

表 6-23 CAMERA 控制器的配置头缺省值

31	24	23	16	15	8	7	0	ADDR
			16'h7a26		16'h0014			00h
			保留		保留		3'h0	04h
8'h 04			8'h80		8'h00		保留	08h
保留			8'h0		保留		8'h10	0Ch
			24'h0				8'h4	10h
				32'h0				14h
				保留				18h
				保留				1Ch
				保留				20h
				保留				24h
				保留				28h
				保留				2Ch
				保留				30h
				保留				34h
				保留				38h
				保留		0xFF		3Ch

### 6.3.15 N/A 处理

当配置头访问的总线号、设备号、功能号和地址偏移无效时，写入操作执行，但是无效。读取操作得到的数据为全 1，表示该访问到无效设备。

N/A	位	读写	缺省值
N/A	63:0	RO	0xffff_ffff_ffff_ffff

## 6.4 IODMA 请求

IODMA 请求的路由方式由访存属性（CACHE/UNCACHE 类型，通过设备对应的 \*\_coherent 寄存器配置）以及 IO 互连网络的地址窗口共同决定。

PCIE 设备（包括所有 PCIE0 和 PCIE1 下游设备）与其他 IO 设备（除 PCIE 之外的所有设备）的 DMA 访问各拥有 8 组地址窗口，可以完成目标地址空间的选择以及从源地址空间到目标地址空间的转换。每个地址窗口由 BASE、MASK 和 MMAP 三个 64 位寄存器组成，BASE 以 K 字节对齐，MASK 采用类似网络掩码高位为 1 的格式。MMAP 各字段含义如下表所示，MMAP[5]表示是否允许 CACHE 访问类型命中，MMAP[6]表示映射后访问类型(0 为 UNCACHE, 1 为 CACHE)，MMAP[7]表示窗口使能。

表 6-24 MMAPI 字段对应的该空间访问属性

[4]	[5]	[6]	[7]
保留	允许 cache 访问命中	映射后访问类型	窗口使能

MMAPI[0]用于选择目标地址窗口，为 0 代表窗口 0，为 1 代表窗口 1。窗口 0 和窗口 1 的路由方式如图 6-2 所示。需注意，PCIE 设备与其他 IO 设备对于目标窗口 0 和目标窗口 1 的路由方式并不相同，PCIE 设备的目标窗口 0 路由到一级交叉开关，目标窗口 1 路由到其他 IO 设备；其他 IO 设备的目标窗口 0 路由到一级交叉开关，目标窗口 1 路由到 Uncache 访存通路，直接通过二级交叉开关访问内存。

判断窗口是否命中需要同时满足以下两个条件：

1. (IN\_ADDR & MASK) == BASE
2. ~IN\_CACHE | MMAPI[5]

其中 IN\_CACHE 为 DMA 的访存属性，通过设备对应的\*\_coherent 配置。如果 DMA 访问为 CACHE 类型，由 MMAPI[5]决定是否允许命中；如果 DMA 访问为 UNCACHED 类型，则不受 MMAPI[5]的控制。

IO 互连网络的地址窗口还具有地址转换和访存类型转换的作用。

窗口命中情况下，新地址换算公式：

$$\text{OUT\_ADDR} = (\text{IN\_ADDR} \& \sim\text{MASK}) | \{\text{MMAPI}[63:10], 10'h0\}$$

窗口命中情况下，新的访存类型由 MMAPI[6]指定。

所有窗口都不命中的情况下，默认路由到窗口 0，同时不做地址转换和访存类型转换。

地址窗口转换寄存器如下表。

表 6-25 IO 设备 DMA 访存地址转换寄存器表

地址	寄存器	R/W	描述	缺省值
0x1fe12400	XBAR_WIN4_BASE0	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 0 的基址	0x0
0x1fe12408	XBAR_WIN4_BASE1	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 1 的基址	0x0
0x1fe12410	XBAR_WIN4_BASE2	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 2 的基址	0x0
0x1fe12418	XBAR_WIN4_BASE3	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 3 的基址	0x0
0x1fe12420	XBAR_WIN4_BASE4	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 4 的基址	0x0
0x1fe12428	XBAR_WIN4_BASE5	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 5 的基址	0x0
0x1fe12430	XBAR_WIN4_BASE6	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 6 的基址	0x0
0x1fe12438	XBAR_WIN4_BASE7	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 7 的基址	0x0
0x1fe12440	XBAR_WIN4_MASK0	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 0 的掩码	0x0
0x1fe12448	XBAR_WIN4_MASK1	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 1 的掩码	0x0
0x1fe12450	XBAR_WIN4_MASK2	RW	除 PCIE 外的 IO 设备 DMA	0x0

地址	寄存器	R/W	描述	缺省值
			访问窗口 2 的掩码	
0x1fe12458	XBAR_WIN4_MASK3	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 3 的掩码	0x0
0x1fe12460	XBAR_WIN4_MASK4	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 4 的掩码	0x0
0x1fe12468	XBAR_WIN4_MASK5	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 5 的掩码	0x0
0x1fe12470	XBAR_WIN4_MASK6	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 6 的掩码	0x0
0x1fe12478	XBAR_WIN4_MASK7	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 7 的掩码	0x0
0x1fe12480	XBAR_WIN4_MMAP0	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 0 的新基址	0x0
0x1fe12488	XBAR_WIN4_MMAP1	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 1 的新基址	0x0
0x1fe12490	XBAR_WIN4_MMAP2	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 2 的新基址	0x0
0x1fe12498	XBAR_WIN4_MMAP3	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 3 的新基址	0x0
0x1fe124a0	XBAR_WIN4_MMAP4	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 4 的新基址	0x0
0x1fe124a8	XBAR_WIN4_MMAP5	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 5 的新基址	0x0
0x1fe124b0	XBAR_WIN4_MMAP6	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 6 的新基址	0x0
0x1fe124b8	XBAR_WIN4_MMAP7	RW	除 PCIE 外的 IO 设备 DMA 访问窗口 7 的新基址	0x0
0x1fe12500	XBAR_WIN5_BASE0	RW	PCIE 设备 DMA 访问窗口 0 的基址	0x0
0x1fe12508	XBAR_WIN5_BASE1	RW	PCIE 设备 DMA 访问窗口 1 的基址	0x0
0x1fe12510	XBAR_WIN5_BASE2	RW	PCIE 设备 DMA 访问窗口 2 的基址	0x0
0x1fe12518	XBAR_WIN5_BASE3	RW	PCIE 设备 DMA 访问窗口 3 的基址	0x0
0x1fe12520	XBAR_WIN5_BASE4	RW	PCIE 设备 DMA 访问窗口 4 的基址	0x0
0x1fe12528	XBAR_WIN5_BASE5	RW	PCIE 设备 DMA 访问窗口 5 的基址	0x0
0x1fe12530	XBAR_WIN5_BASE6	RW	PCIE 设备 DMA 访问窗口 6 的基址	0x0
0x1fe12538	XBAR_WIN5_BASE7	RW	PCIE 设备 DMA 访问窗口 7 的基址	0x0
0x1fe12540	XBAR_WIN5_MASK0	RW	PCIE 设备 DMA 访问窗口 0 的掩码	0x0
0x1fe12548	XBAR_WIN5_MASK1	RW	PCIE 设备 DMA 访问窗口 1 的掩码	0x0
0x1fe12550	XBAR_WIN5_MASK2	RW	PCIE 设备 DMA 访问窗口 2 的掩码	0x0
0x1fe12558	XBAR_WIN5_MASK3	RW	PCIE 设备 DMA 访问窗口 3 的掩码	0x0
0x1fe12560	XBAR_WIN5_MASK4	RW	PCIE 设备 DMA 访问窗口 4 的掩码	0x0
0x1fe12568	XBAR_WIN5_MASK5	RW	PCIE 设备 DMA 访问窗口 5 的掩码	0x0
0x1fe12570	XBAR_WIN5_MASK6	RW	PCIE 设备 DMA 访问窗口 6 的掩码	0x0

地址	寄存器	R/W	描述	缺省值
0x1fe12578	XBAR_WIN5_MASK7	RW	PCIE 设备 DMA 访问窗口 7 的掩码	0x0
0x1fe12580	XBAR_WIN5_MMAP0	RW	PCIE 设备 DMA 访问窗口 0 的新基址	0x0
0x1fe12588	XBAR_WIN5_MMAP1	RW	PCIE 设备 DMA 访问窗口 1 的新基址	0x0
0x1fe12590	XBAR_WIN5_MMAP2	RW	PCIE 设备 DMA 访问窗口 2 的新基址	0x0
0x1fe12598	XBAR_WIN5_MMAP3	RW	PCIE 设备 DMA 访问窗口 3 的新基址	0x0
0x1fe125a0	XBAR_WIN5_MMAP4	RW	PCIE 设备 DMA 访问窗口 4 的新基址	0x0
0x1fe125a8	XBAR_WIN5_MMAP5	RW	PCIE 设备 DMA 访问窗口 5 的新基址	0x0
0x1fe125b0	XBAR_WIN5_MMAP6	RW	PCIE 设备 DMA 访问窗口 6 的新基址	0x0
0x1fe125b8	XBAR_WIN5_MMAP7	RW	PCIE 设备 DMA 访问窗口 7 的新基址	0x0

以 PCIE MSI 中断的路由配置方法为例。因为系统分配给 MSI 的地址分别为 0x1fe114b0 和 0x1fe114f0，这两个地址都需要通过 uncache 方式访问，所以可以配置以下窗口：

XBAR\_WIN5\_BASE0 配置为 0x1FE1\_0000；

XBAR\_WIN5\_MASK0 配置为 0xFFFF\_FFFF\_FFFF\_0000；

XBAR\_WIN5\_MMAP0 配置为 0x81；

## 6.5 APB 设备路由

APB 总线控制器下挂载了 UART 控制器、CAN 控制器、I2C 控制器、PWM 控制器、RTC 控制器、HPET 控制器、NAND 控制器、ACPI 控制器、DES 控制器、AES 控制器、RSA 控制器、RNG 控制器、专用通信接口、SDIO 控制器和 I2S 控制器。由访问地址的[15:12]来进行路由，具体如下表所示。

表 6-26 APB 设备地址译码

地址[15:12]	设备	备注
0x0	UART * 12 CAN * 2	用地址的[11:8]做下一级地址译码： 0x0: UART0 0x1: UART1 0x2: UART2 0x3: UART3 0x4: UART4 0x5: UART5 0x6: UART6 0x7: UART7 0x8: UART8 0x9: UART9 0xA: UART10 0xB: UART11 0xC: CAN0 0xD: CAN1 0xE: NULL 0xF: NULL

0x1	I2C * 2	用地址[11]做下一级地址译码: 0x0: I2C0 0x1: I2C1
0x2	PWM	
0x3	Reserved	
0x4	HPET	
0x5	Reserved	
0x6	NAND	
0x7	ACPI/RTC	ACPI 的偏移地址为 0x7000 RTC 的偏移地址为 0x7800
0x8	DES	
0x9	AES	
0xA	RSA	
0xB	RNG	
0xC	SDIO	
0xD	I2S	
0xE	专用通信接口 E1	
0xF	NULL	

# 7 处理器核间中断与通信

龙芯 2K1000 为每个处理器核都实现了 8 个核间中断寄存器 (IPI) 以支持多核 BIOS 启动和操作系统运行时在处理器核之间进行中断和通信，其说明和地址见表 7-1 到表 7-3。

表 7-1 处理器核间中断相关的寄存器及其功能描述

名称	读写权限	描述
IPISR	R	32 位状态寄存器，任何一位有被置 1 且对应位使能情况下，处理器核 INT4 中断线被置位。
IPIEN	RW	32 位使能寄存器，控制对应中断位是否有效
IPiset	W	32 位置位寄存器，往对应的位写 1，则对应的 STATUS 寄存器位被置 1
IPI_CLR	W	32 位清除寄存器，往对应的位写 1，则对应的 STATUS 寄存器位被清 0
BUFO	RW	缓存寄存器，供启动时传递参数使用，按 64 或者 32 位的 uncache 方式进行访问。
BUF1	RW	缓存寄存器，供启动时传递参数使用，按 64 或者 32 位的 uncache 方式进行访问。
BUF2	RW	缓存寄存器，供启动时传递参数使用，按 64 或者 32 位的 uncache 方式进行访问。
BUF3	RW	缓存寄存器，供启动时传递参数使用，按 64 或者 32 位的 uncache 方式进行访问。

在龙芯 2K1000 与处理器核间中断相关的寄存器及其功能描述如下：

表 7-2 0 号处理器核核间中断与通信寄存器列表

名称	地址	权限	描述
CORE0_IPISR	0x1fe11000	R	0 号处理器核的 IPI_Status 寄存器
CORE0_IPIEN	0x1fe11004	RW	0 号处理器核的 IPI_Enalbe 寄存器
CORE0_IPiset	0x1fe11008	W	0 号处理器核的 IPI_Set 寄存器
CORE0_IPICLR	0x1fe1100c	W	0 号处理器核的 IPI_Clear 寄存器
CORE0_BUFO	0x1fe11020	RW	0 号处理器核的 IPI_MailBox0 寄存器
CORE0_BUF1	0x1fe11028	RW	0 号处理器核的 IPI_MailBox1 寄存器
CORE0_BUF2	0x1fe11030	RW	0 号处理器核的 IPI_MailBox2 寄存器
CORE0_BUF3	0x1fe11038	RW	0 号处理器核的 IPI_MailBox3 寄存器

表 7-3 1 号处理器核的核间中断与通信寄存器列表

名称	地址	权限	描述
CORE1_IPISR	0x1fe11100	R	1 号处理器核的 IPI_Status 寄存器
CORE1_IPIEN	0x1fe11104	RW	1 号处理器核的 IPI_Enalbe 寄存器
CORE1_IPiset	0x1fe11108	W	1 号处理器核的 IPI_Set 寄存器

CORE1_IPICLR	0x1fe1110c	W	1号处理器核的 IPI_Clear 寄存器
CORE1_BUF0	0x1fe11120	R	1号处理器核的 IPI_MailBox0 寄存器
CORE1_BUF1	0x1fe11128	RW	1号处理器核的 IPI_MailBox1 寄存器
CORE1_BUF2	0x1fe11130	W	1号处理器核的 IPI_MailBox2 寄存器
CORE1_BUF3	0x1fe11138	W	1号处理器核的 IPI_MailBox3 寄存器

# 8 温度传感器

## 8.1 实时温度采样

龙芯 2K1000 内部集成一个温度传感器，可以通过 0x1FE11510 开始的采样寄存器进行观测，同时，可以使用灵活的高低温中断报警或者自动调频功能进行控制。温度传感器在采样寄存器的对应位如下（基址为 0x1FE11510）：

表 8-1 温度采样寄存器说明

位域	字段名	访问	复位值	描述
0	Thsens_int_lo	RW		低温触发中断状态，写入任意值清除中断
1	Thsens_int_hi	RW		高温触发中断状态，写入任意值清除中断
3:2		R		保留
4	Thsens_overflow	R		温度传感器上溢（超过 125°C）
39:32	Thsens_out	R		温度传感器 0 摄氏温度 结点温度=Thens0_out -100 温度范围 -40 度 – 125 度
其它		R		保留

通过对控制寄存器的设置，可以实现超过预设温度中断、低于预设温度中断及高温自动降频功能。

## 8.2 高低温中断触发

对于高低温中断报警功能，分别有 4 组控制寄存器对其阈值进行设置。每组寄存器包含以下三个控制位：

GATE：设置高温或低温的阈值。当输入温度高于高温阈值或低于低温阈值时，将会产生中断；

EN：中断使能控制。置 1 之后该组寄存器的设置才有效；

SEL：输入温度选择。当前 2K1000 内部集成一个温度传感器，所以该位只可以配置成 0。

高温中断控制寄存器中包含 4 组用于控制高温中断触发的设置位；低温中断控制寄存器中包含 4 组用于控制低温中断触发的设置位。另外还有一组寄存器用于显示中断状态，分别对应于高温中断和低温中断，对该寄存器进行任意写操作将清除中断状态。

这几个寄存器的具体描述如下：

表 8-2 高低温中断寄存器说明

寄存器	地址	控制	说明
高温中断控制寄存器 Thsens_int_ctrl_Hi	0x1FE11500	RW	[7:0]: Hi_gate0: 高温阈值 0, 超过这个温度将产生中断 [8:8]: Hi_en0: 高温中断使能 0 [11:10]: Hi_Sel0: 选择高温中断 0 的温度传感器输入源 [23:16]: Hi_gate1: 高温阈值 1, 超过这个温度将产生中断 [24:24]: Hi_en1: 高温中断使能 1 [27:26]: Hi_Sel1: 选择高温中断 1 的温度传感器输入源 [39:32]: Hi_gate2: 高温阈值 2, 超过这个温度将产生中断 [40:40]: Hi_en2: 高温中断使能 2 [43:42]: Hi_Sel2: 选择高温中断 2 的温度传感器输入源 [55:48]: Hi_gate3: 高温阈值 3, 超过这个温度将产生中断 [56:56]: Hi_en3: 高温中断使能 3 [59:58]: Hi_Sel3: 选择高温中断 3 的温度传感器输入源
低温中断控制寄存器 Thsens_int_ctrl_Lo	0x1FE11508	RW	[7:0]: Lo_gate0: 低温阈值 0, 低于这个温度将产生中断 [8:8]: Lo_en0: 低温中断使能 0 [11:10]: Lo_Sel0: 选择低温中断 0 的温度传感器输入源 [23:16]: Lo_gate1: 低温阈值 1, 低于这个温度将产生中断 [24:24]: Lo_en1: 低温中断使能 1 [27:26]: Lo_Sel1: 选择低温中断 1 的温度传感器输入源 [39:32]: Lo_gate2: 低温阈值 2, 低于这个温度将产生中断 [40:40]: Lo_en2: 低温中断使能 2 [43:42]: Lo_Sel2: 选择低温中断 2 的温度传感器输入源 [55:48]: Lo_gate3: 低温阈值 3, 低于这个温度将产生中断 [56:56]: Lo_en3: 低温中断使能 3 [59:58]: Lo_Sel3: 选择低温中断 3 的温度传感器输入源
中断状态寄存器 Thsens_int_status/clr	0x1FE11510	RW	中断状态寄存器, 写任意值清除中断 [0]: 高温中断触发 [1]: 低温中断触发

### 8.3 高温自动降频设置

为了在高温环境中保证芯片的运行, 可以设置令高温自动降频, 使得芯片在超过预设范围时主动进行时钟分频, 达到降低芯片翻转率的效果。

对于高温降频功能, 有 4 组控制寄存器对其行为进行设置。每组寄存器包含以下四个控制位:

**GATE:** 设置高温或低温的阈值。当输入温度高于高温阈值或低于低温阈值时, 将触发分频操作;

**EN:** 高温降频使能控制。置 1 之后该组寄存器的设置才有效;

**SEL:** 输入温度选择。当前 2K1000 内部集成一个温度传感器, 所以该位只可以配置成 0。

**FREQ:** 分频数。当触发分频操作时, 将频率调整为当前时钟频率的 FREQ/8 倍。

表 8-3 高温降频控制寄存器说明

寄存器	地址	控制	说明
高温降频控制寄存器 Thsens_scale_hi	0x1FE11520	RW	<p>四组设置优先级由高到低</p> <p>[7:0]: Scale_gate0: 高温阈值 0, 超过这个温度将降频  [8:8]: Scale_en0: 高温降频使能 0  [11:10]: Scale_Sel0: 选择高温降频 0 的温度传感器输入源  [14:12]: Scale_freq0: 降频时的分频值  [23:16]: Scale_gate1: 高温阈值 1, 超过这个温度将降频  [24:24]: Scale_en1: 高温降频使能 1  [27:26]: Scale_Sel1: 选择高温降频 1 的温度传感器输入源  [30:28]: Scale_freq1: 降频时的分频值  [39:32]: Scale_gate2: 高温阈值 2, 超过这个温度将降频  [40:40]: Scale_en2: 高温降频使能 2  [43:42]: Scale_Sel2: 选择高温降频 2 的温度传感器输入源  [46:44]: Scale_freq2: 降频时的分频值  [55:48]: Scale_gate3: 高温阈值 3, 超过这个温度将降频  [56:56]: Scale_en3: 高温降频使能 3  [59:58]: Scale_Sel3: 选择高温降频 3 的温度传感器输入源  [62:60]: Scale_freq3: 降频时的分频值</p>

## 9 I/O 中断

龙芯 2K1000 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。

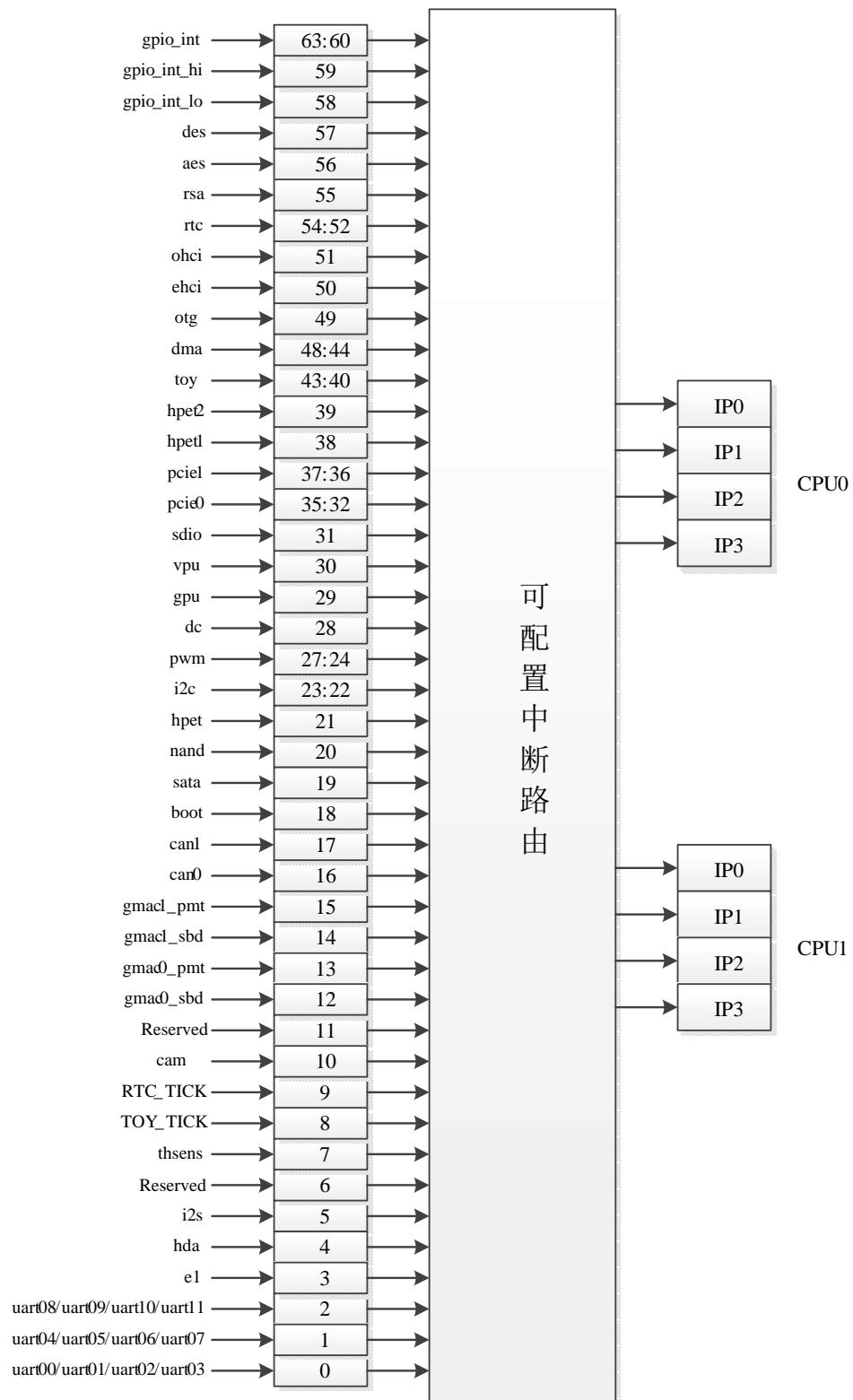


图 9-1 龙芯 2K1000 处理器中断路由示意图

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 9-1。中断使能（Enable）的配置有三个寄存器：Interset、Intenclr 和 Inten。Interset 设置中断使能，Interset 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 Interset 才能采集到该中断的下一次脉冲触发。

## 9.1 中断触发类型

对于 2K1000 来说，dma 控制器中断和 PCIE MSI 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。

## 9.2 中断分发模式

在龙芯 2K1000 中还增加了硬件中断负载均衡功能，使得中断可以在软件预设的几个处理器核上进行路由，分为三种模式：

1. 轮转分发模式（bounce），这种模式下每个新中断产生时，会按照 Entry 寄存器中处理器核向量的配置，路由到下一个处理器核上。
2. 忙碌分发模式——当前处理器忙碌则跳转到下一个（0->1->2->3）候选处理器核（auto bounce），这种模式下每个新中断产生时，会首先检测原有中断的处理器核上是否已有未被处理的中断，如果没有，则继续中断该核，如果存在未被处理的中断，则按照 Entry 寄存器中处理器核向量的配置，路由到下一个处理器核上。
3. 空闲分发模式——跳转到空闲的候选处理器核（auto），这种模式下，每个新中断产生时，会按照 Entry 寄存器中处理器核向量的配置，首先检测下一个处理器核上是否已有未被处理的中断，如果没有，则中断该核；如果存在未被处理的中断，则继续检测下一个处理器核。依此类推，直到将所有 Entry 寄存器中处理器核向量都检查一遍，如果仍旧没有空闲的处理器核，那么该中断将在最初的处理器核上处理，不论该核是否有未被处理的中断。

硬件负载均衡使能由寄存器 Intbounce 和 Intauto 共同控制。如果仅当 Intbounce 有效，而 Intauto 对应位为无效，则采用轮转分发模式；在 Intbounce 对应位有效的情况下，如果 Intauto 对应位也有效，则采用忙碌分发模式；在仅当 Intauto 有效时，则采用空闲分发模式。当 Intbounce 或 Intauto 对应位有效的情况下，Entry 寄存器代表可用的处理器核。当 Intbounce 和 Intauto 对应位均无效的情况下，Entry 寄存器中处理器核向量配置只能有一个处理器核，且该中断路由到该核上。需要注意的是，一旦配置完 Intbounce 和 Intauto 后不应该在运行中途修改。

### 9.3 中断相关寄存器描述

表 9-1 中断控制寄存器属性

位域	访问属性/缺省值							中断源
	Intedge	Intpol	Inten	Intensem	Intenclr	Intbounce	Intauto	
0	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Uart00-uart03
1	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Uart04-uart07
2	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Uart08-uart11
3	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	专用通信接口
4	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Had_int
5	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	I2s_int
6	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Reserved
7	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Thsens_int
8	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	TOY_TICK
9	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	RTC_TICK
10	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	cam_int
11	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Reserved
12	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Gmac0_sbd_int
13	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Gmac0_pmt_int
14	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Gmac1_sbd_int
15	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Gmac1_pmt_int
16	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Can0_int
17	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Can1_int
18	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Spi_int
19	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Sata_int
20	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Nand_int
21	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Hpet_int
23:22	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	I2c_int
27:24	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Pwm_int
28	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Dc_int
29	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Gpu_int
30	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	vpu_int
31	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Sdio_int
35:32	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Pcie0_int
37:36	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Pcie1_int
38	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	hpet1_int
39	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	hpet2_int

位域	访问属性/缺省值							
43:40	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Toy_int
48:44	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Dma_int[4:0]
49	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Otg_int
50	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Ehci_int
51	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Ohci_int
54:52	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Rtc_int
55	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Rsa_int
56	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Aes_int
57	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Des_int
58	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Gpio_int_lo
59	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Gpio_int_hi
63:60	RW / 0	RW / 0	R / 0	W / 0	W / 0	RW / 0	RW / 0	Gpio_int

表 9-2 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
Intensem_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	WR	0x0	中断极性控制：0 代表低电平，1 代表高电平
Intedge_0	0x1fe11434	WR	0x0	低 32 位触发方式寄存器（1：脉冲触发；0：电平触发）
Intbounce_0	0x1fe11438	WO	0x0	低 32 位中断分发模式控制，与 auto 合用。 {auto, bounce}： 2'b00-固定分发模式 2'b01-轮转分发模式 2'b10-空闲分发模式 2'b11-忙碌分发模式
Intauto_0	0x1fe1143c	WO	0x0	低 32 位中断分发模式控制，与 bounce 合用。 {auto, bounce}： 2'b00-固定分发模式 2'b01-轮转分发模式 2'b10-空闲分发模式 2'b11-忙碌分发模式
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
Intensem_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	WR	0x0	中断极性控制：0 代表低电平，1 代表高电平
Intedge_1	0x1fe11474	WR	0x0	高 32 位触发方式寄存器（1：脉冲触发；0：电平触发）
Intbounce_1	0x1fe11478	WO	0x0	高 32 位中断分发模式控制，与 auto 合用。 {auto, bounce}：

名称	地址偏移	访问属性	缺省值	描述
				2'b00-固定分发模式 2'b01-轮转分发模式 2'b10-空闲分发模式 2'b11-忙碌分发模式
Intauto_1	0x1fe11464	RO	0x0	高 32 位中断分发模式控制，与 bounce 合用。 {auto, bounce}: 2'b00-固定分发模式 2'b01-轮转分发模式 2'b10-空闲分发模式 2'b11-忙碌分发模式
CORE0_IPISR	0x1fe11000	RO	NA	0 号处理器核的 IPI_Status 寄存器
CORE0_IPIEN	0x1fe11004	RW	0x0	0 号处理器核的 IPI_Enalbe 寄存器
CORE0_IPISET	0x1fe11008	WO	NA	0 号处理器核的 IPI_Set 寄存器
CORE0_IPI_CLR	0x1fe1100c	WO	NA	0 号处理器核的 IPI_Clear 寄存器
CORE0_BUF0	0x1fe11020	RW	0x0	0 号处理器核的 IPI_MailBox0 寄存器
CORE0_BUF1	0x1fe11028	RW	0x0	0 号处理器核的 IPI_MailBox1 寄存器
CORE0_BUF2	0x1fe11030	RW	0x0	0 号处理器核的 IPI_MailBox2 寄存器
CORE0_BUF3	0x1fe11038	RW	0x0	0 号处理器核的 IPI_MailBox3 寄存器
CORE0_INTISR0	0x1fe11040	RO	NA	路由给 CORE0 的低 32 位中断状态
CORE0_INTISR1	0x1fe11048	RO	NA	路由给 CORE0 的高 32 位中断状态
CORE1_IPISR	0x1fe11100	RO	NA	1 号处理器核的 IPI_Status 寄存器
CORE1_IPIEN	0x1fe11104	RW	0x0	1 号处理器核的 IPI_Enalbe 寄存器
CORE1_IPISET	0x1fe11108	WO	NA	1 号处理器核的 IPI_Set 寄存器
CORE1_IPI_CLR	0x1fe1110c	WO	NA	1 号处理器核的 IPI_Clear 寄存器
CORE1_BUF0	0x1fe11120	RW	0x0	1 号处理器核的 IPI_MailBox0 寄存器
CORE1_BUF1	0x1fe11128	RW	0x0	1 号处理器核的 IPI_MailBox1 寄存器
CORE1_BUF2	0x1fe11130	RW	0x0	1 号处理器核的 IPI_MailBox2 寄存器
CORE1_BUF3	0x1fe11138	RW	0x0	1 号处理器核的 IPI_MailBox3 寄存器
CORE1_INTISR0	0x1fe11140	RO	NA	路由给 CORE1 的低 32 位中断状态
CORE1_INTISR1	0x1fe11148	RO	NA	路由给 CORE1 的高 32 位中断状态

龙芯 2K1000 中集成了 2 个处理器核，上述的 64 位中断源可以通过软件配置选择期望中断的目标处理器核。更进一步，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，即对应 CP0\_Status 的 IP2 到 IP5。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 0x42 表示路由到 1 号处理器的 INT2 上。

表 9-3 中断路由寄存器的说明

位域	说明
3:0	路由的处理器核向量号

7:4

路由的处理器核中断引脚向量号

表 9-4 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	Uart00-03	Entry32	0x1fe11440	Pcie0_int0
Entry1	0x1fe11401	Uart04-07	Entry33	0x1fe11441	Pcie0_int1
Entry2	0x1fe11402	Uart08-11	Entry34	0x1fe11442	Pcie0_int2
Entry3	0x1fe11403	专用通信接口	Entry35	0x1fe11443	Pcie0_int3
Entry4	0x1fe11404	Hda_int	Entry36	0x1fe11444	Pcie1_int0
Entry5	0x1fe11405	I2s_int	Entry37	0x1fe11445	Pcie1_int1
Entry6	0x1fe11406	Reserved	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	cam	Entry42	0x1fe1144a	Toy_int2
Entry11	0x1fe1140b	Reserved	Entry43	0x1fe1144b	Toy_int3
Entry12	0x1fe1140c	Gmac0_sbd_int	Entry44	0x1fe1144c	Dma_int0
Entry13	0x1fe1140d	Gmac0_pmt_int	Entry45	0x1fe1144d	Dma_int1
Entry14	0x1fe1140e	Gmac1_sbd_int	Entry46	0x1fe1144e	Dma_int2
Entry15	0x1fe1140f	Gmac1_pmt_int	Entry47	0x1fe1144f	Dma_int3
Entry16	0x1fe11410	Can0_int	Entry48	0x1fe11450	Dma_int4
Entry17	0x1fe11411	Can1_int	Entry49	0x1fe11451	Otg_int
Entry18	0x1fe11412	Spi_int	Entry50	0x1fe11452	Ehci_int
Entry19	0x1fe11413	Sata_int	Entry51	0x1fe11453	Ohci_int
Entry20	0x1fe11414	Nand_int	Entry52	0x1fe11454	Rtc_int0
Entry21	0x1fe11415	Hpet_int	Entry53	0x1fe11455	Rtc_int1
Entry22	0x1fe11416	I2c_int0	Entry54	0x1fe11456	Rtc_int3
Entry23	0x1fe11417	I2c_int1	Entry55	0x1fe11457	Rsa_int
Entry24	0x1fe11418	Pwm_int0	Entry56	0x1fe11458	Aes_int
Entry25	0x1fe11419	Pwm_int1	Entry57	0x1fe11459	Des_int
Entry26	0x1fe1141a	Pwm_int2	Entry58	0x1fe1145a	Gpio_int_lo
Entry27	0x1fe1141b	Pwm_int3	Entry59	0x1fe1145b	Gpio_int_hi
Entry28	0x1fe1141c	Dc_int	Entry60	0x1fe1145c	Gpio_int0
Entry29	0x1fe1141d	Gpu_int	Entry61	0x1fe1145d	Gpio_int1
Entry30	0x1fe1141e	vpu	Entry62	0x1fe1145e	Gpio_int2

名称	地址偏移	描述	名称	地址偏移	描述
Entry31	0x1fe1141f	Sdio_int	Entry63	0x1fe1145f	Gpio_int3

## 9.4 GPIO 中断

龙芯 2K1000 有 60 个 GPIO 引脚，具体请参考 13.5 节。这些 GPIO 引脚与中断引脚的对应关系如下：

表 9-5 GPIO 中断

GPIO 引脚	中断引脚	说明
GPIO0	Gpio_int0	专用 GPIO 引脚，与中断引脚一一对应，需设置中断使能，参考 13.4 节
GPIO1	Gpio_int1	
GPIO2	Gpio_int2	
GPIO3	Gpio_int3	
GPIO[31:04]	Gpio_int_lo	GPIO31-GPIO4 复用中断引脚 Gpio_int_lo
GPIO[63:32]	Gpio_int_hi	GPIO63-GPIO32 复用中断引脚 Gpio_int_hi

## 9.5 MSI 中断

龙芯 2K1000 的 PCIE 控制器支持 MSI 中断类型。当接收到 MSI 中断时，PCIE 控制器把对应的中断请求转发到中断控制器。中断控制器将 MSI 中断的 Message Data 译码成中断线的形式，与现有的中断引脚复用。

龙芯 2K1000 分配给 MSI 的地址有两个，分别对应寄存器 INT\_MSI\_ADDR\_0 (0x1fe114b0) 和 INT\_MSI\_ADDR\_1 (0x1fe114f0)，每个地址可以接受 32 个中断消息，分别与现有的中断引脚低 32 位和高 32 位复用。与 MSI 相关的寄存器如下表所示：

表 9-6 MSI 中断相关寄存器

0x1fe114a0	INT_MSI_0	保留
0x1fe114a8	INT_MSI_EN_0	保留
0x1fe114b0	INT_MSI_ADDR_0	MSI 地址 0
0x1fe114b4	INT_MSI_TRIGGER_EN_0	32 位 MSI 中断使能，每位对应一个写入 MSI 地址 0 的中断
0x1fe114e0	INT_MSI_1	保留
0x1fe114e8	INT_MSI_EN_1	保留
0x1fe114f0	INT_MSI_ADDR_1	MSI 地址 1
0x1fe114f4	INT_MSI_TRIGGER_EN_1	32 位 MSI 中断使能，每位对应一个写入 MSI 地址 1 的中断

在接收 MSI 中断之前需要先将对应的 MSI 中断使能位设为 1。另外，MSI 中断为脉冲触发类型。

MSI 中断配置流程：

1. 配置设备的 MSI 相关 capability，系统分配的 MSI 地址为 INT\_MSI\_ADDR\_0 (0x1fe114b0) 和 INT\_MSI\_ADDR\_1 (0x1fe114f0)；

2. 配置中断触发类型 INTEDGE 寄存器，将 MSI 中断设置为脉冲触发，即对应位设为 1；
3. 配置对应的 INT\_MSI\_TRIGGER\_EN 寄存器，即将对应的 MSI 中断使能位设为 1；
4. 配置对应的 INTENSET 寄存器，即将对应的中断使能位设为 1；
5. 读对应的中断状态 INTISR 寄存器，获取相应的中断信息；
6. 读到对应的中断后，配置对应的 INTENCLR 寄存器，将其对应位设为 1，清除中断；
7. 再重新将对应的中断使能位(INTENSET 寄存器)设为 1。

## 9.6 硬件中断负载均衡功能举例

### 1. 轮转分发模式 (bounce)

步骤	处理器核向量配置	Bounce	Auto	触发中断	处理器核	清除中断	说明
1	Entry0[3:0] = 4'b0011 Entry1[3:0] = 4'b0010	32'b1	32'b0				Int0 的 bounce 有效
2				Int0	Core0		路由到配置中最右边的处理器核，即 Core0
3						Int0	
4				Int1	Core1		使 Core1 上有未被处理的中断
5				Int0	Core1		仍旧路由到 Entry 配置的下一个处理器核，即 Core1
6						Int0	
7				Int0	Core0		路由到 Entry 配置的下一个处理器核，即 Core0
8						Int0 Int1	

### 2. 忙碌分发模式

步骤	处理器核向量配置	Bounce	Auto	触发中断	处理器核	清除中断	说明
1	Entry0[3:0] = 4'b0011 Entry1[3:0] = 4'b0001 Entry2[3:0] = 4'b0010	32'b1	32'b1				Int0 的 bounce 和 auto 有效
2				Int0	Core0		路由到 Entry 配置中最右边的处理器核，即 Core0
3				Int1	Core0		使 Core0 上有未被处理的中断
4				Int2	Core1		使 Core1 上有未被处理的中断
5						Int0	
6				Int0	Core1		由于 Core0 上有未被处理的中断，路由到 Entry 配置的下一个处理器核，即 core1。
7						Int0	
8				Int0	Core0		由于 Core1 上有未被处理的中断，路由到 Entry 配置的下一个处理器核，即 Core0
9						Int0 Int1	
10				Int0	Core0		Core0 上没有未被处理的中断，继续中断该核，即

						Core0
11					Int0 Int2	

### 3. 空闲分发模式

步骤	处理器核向量配置	Bounce	Auto	触发中断	处理器核	清除中断	说明
1	Entry0[3:0] = 4'b0011 Entry1[3:0] = 4'b0001 Entry2[3:0] = 4'b0010	32'b0	32'b1				Int0 的 auto 有效
2				Int0	Core0		路由到 Entry 配置中最右边的处理器核, 即 Core0
3				Int2	Core1		使 Core1 上有未被处理的中断
4						Int0	
5				Int0	Core0		由于 Core1 上有未被处理的中断, 而 Core0 上没有未被处理的中断, 所以路由到 Entry 配置的下一个空闲的处理器核, 即 Core0。
6						Int0	
7				Int1	Core0		使 Core0 上有未被处理的中断
8				Int0	Core0		由于 Entry 配置的核 (Core0 和 Core1) 都有未被处理的中断, 只能继续中断该核, 即 Core0
9						Int2 Int0	
10				Int0	Core1		由于 Core1 空闲, 所以路由到 Entry 配置的下一个空闲的处理器核, 即 Core1。
11						Int0 Int1	

# 10 SPI 控制器

串行外围设备接口 SPI 总线技术是多种微处理器、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。

## 10.1 访问地址

SPI 控制器包括两个地址空间，分别如下：

表 10-1 SPI 控制器地址空间分配

起始地址	结束地址	名称	说明
0x1FC0_0000	0x1FCF_FFFF	启动空间	当引脚设置为 SPI 启动时可访问，其它情况下不可访问
0xFFFF_0220	0xFFFF_022F	配置寄存器空间	

## 10.2 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。对于软件而言，SPI 控制器除了有若干 IO 寄存器外还有一段映射到 SPI Flash 的只读 memory 空间。如果将这段 memory 空间分配在 0x1fc00000，复位后不需要软件干预就可以直接访问，从而支持处理器从 SPI Flash 启动。SPI 的 IO 寄存器的基址 0xffff0220。

其结构如图 10-1 所示，由 AXI 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)。

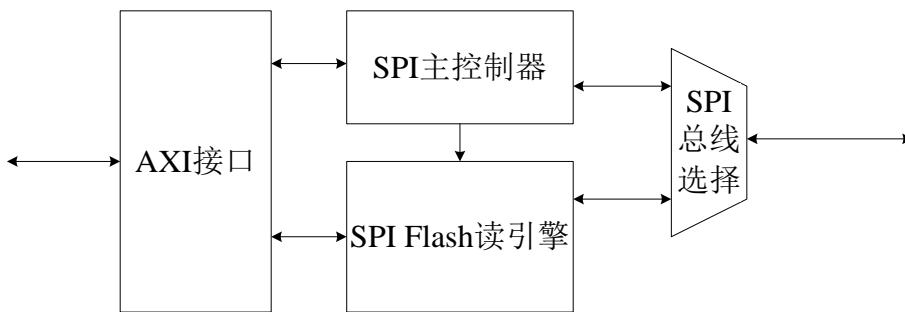


图 10-1 SPI 控制器结构

## 10.3 配置寄存器

表 10-2 SPI 配置寄存器列表

偏移	名称	描述
0	SPCR	控制寄存器
1	SPSR	状态寄存器
2	TxFIFO/RxFIFO	数据寄存器

偏移	名称	描述
3	SPER	外部寄存器
4	SFC_PARAM	参数控制寄存器
5	SFC_SOFTCS	片选控制寄存器
6	SFC_TIMING	时序控制寄存器

### 10.3.1 控制寄存器(SPCR)

偏移地址: 0x0

表 10-3 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

### 10.3.2 状态寄存器(SPSR)

偏移地址: 0x1

表 10-4 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
5:4	-	-	0	保留
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

### 10.3.3 数据寄存器(TxFIFO/RxFIFO)

偏移地址: 0x2

表 10-5 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

### 10.3.4 外部寄存器(SPER)

偏移地址: 0x3

表 10-6 SPI 外部寄存器(SPER)

位域	名称	访问	初值	描述
7:6	icnt	R/W	0	传输完多少个字节后发中断 00: 1 01: 2 10: 3 11: 4
5:3	-	-	-	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 10-7 SPI 分频系数

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

### 10.3.5 参数控制寄存器(SFC\_PARAM)

偏移地址: 0x4

表 10-8 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
0	memory_en	R/W	1	SPI flash 读使能, 无效时 csn[0] 可由软件控制。

### 10.3.6 片选控制寄存器(SFC\_SOFTCS)

偏移地址: 0x5

表 10-9 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

### 10.3.7 时序控制寄存器(SFC\_TIMING)

偏移地址: 0x6

表 10-10 SPI 时序控制寄存器(SFC\_TIMING)

位域	名称	访问	初值	描述
7:3	-	-	-	保留

位域	名称	访问	初值	描述
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

## 10.4 接口时序

### 10.4.1 SPI 主控制器接口时序

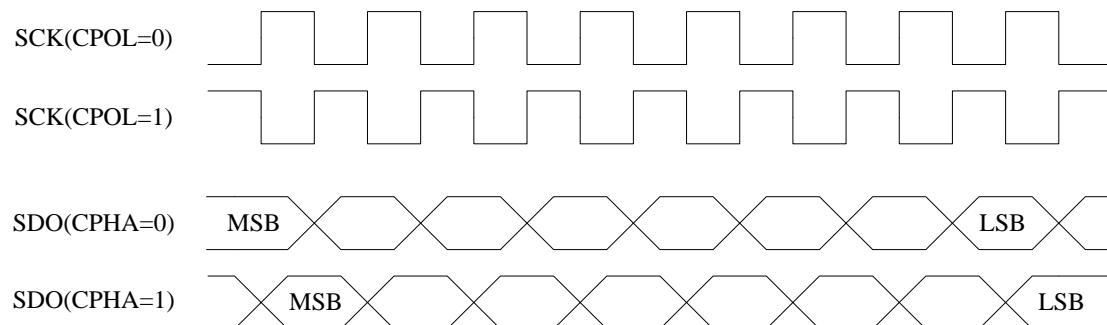


图 10-2 SPI 主控制器接口时序

### 10.4.2 SPI Flash 访问时序

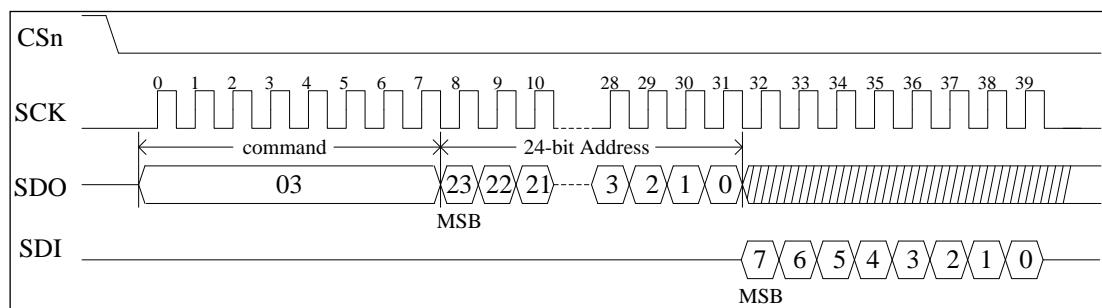


图 10-3 SPI Flash 标准读时序

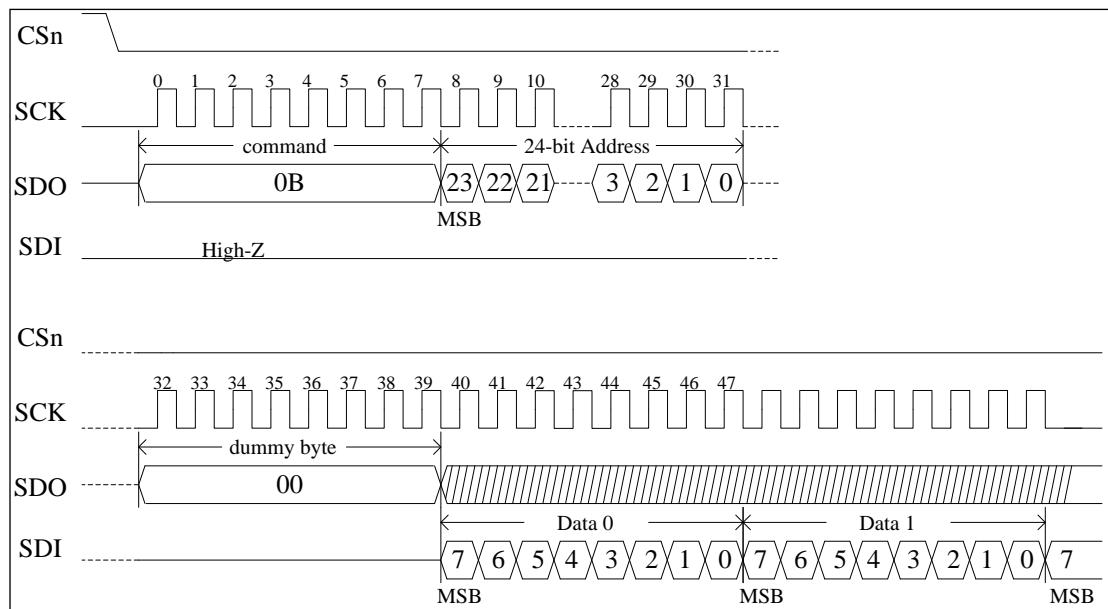


图 10-4 SPI Flash 快速读时序

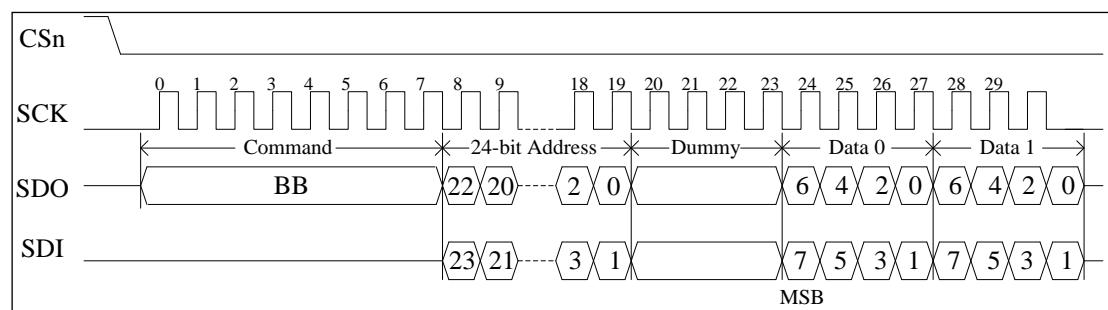


图 10-5 SPI Flash 双向 I/O 读时序

## 10.5 软件编程指南

### 10.5.1 SPI 主控制器的读写操作

- 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

- 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。

- 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

### 10.5.2 硬件 SPI Flash 读

- 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。

- 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。

### 10.5.3 混合访问 SPI Flash 和 SPI 主控制器

- 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。

# 11 LocalIO 控制器

## 11.1 访问地址及引脚复用

表 11-1 LocalIO 地址空间分配

起始地址	结束名称	名称	说明
0x1FC0_0000	0x1FCF_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
0x1C00_0000	0x1FBF_FFFF		
0x1FD0_0000	0x1FDF_FFFF	存储空间	
0x1FF0_0000	0x1FFF_FFFF		

对于 LocalIO 模块，使用时要注意将对应的引脚设置为相应功能。

与 LocalIO 相关的引脚设置寄存器为 5.1 节中的 lio\_sel。

## 11.2 LocalIO 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 10-1，图 10-2。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。

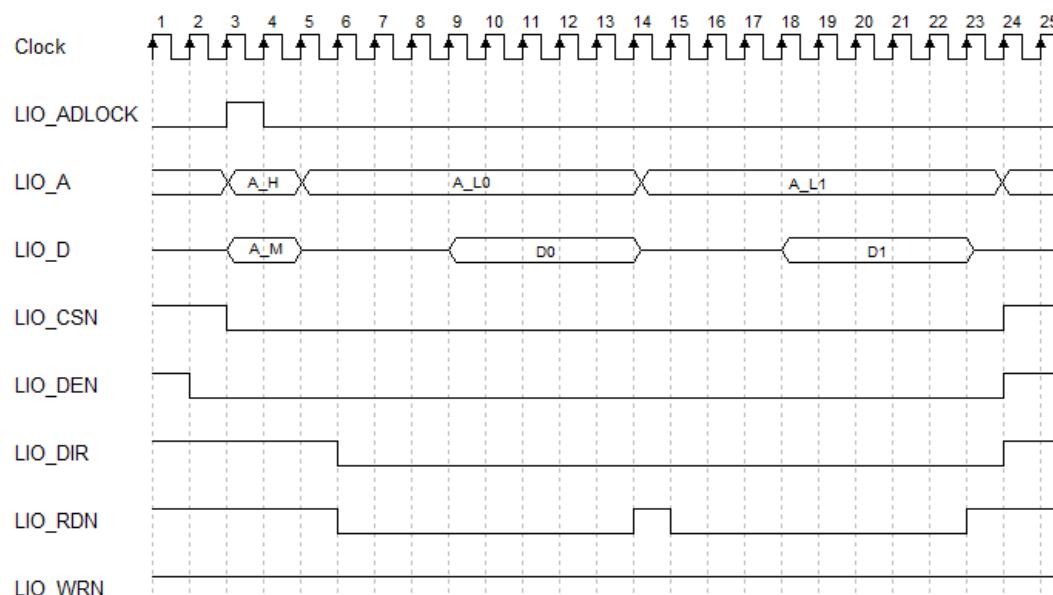


图 11-1 LocalIO 读时序

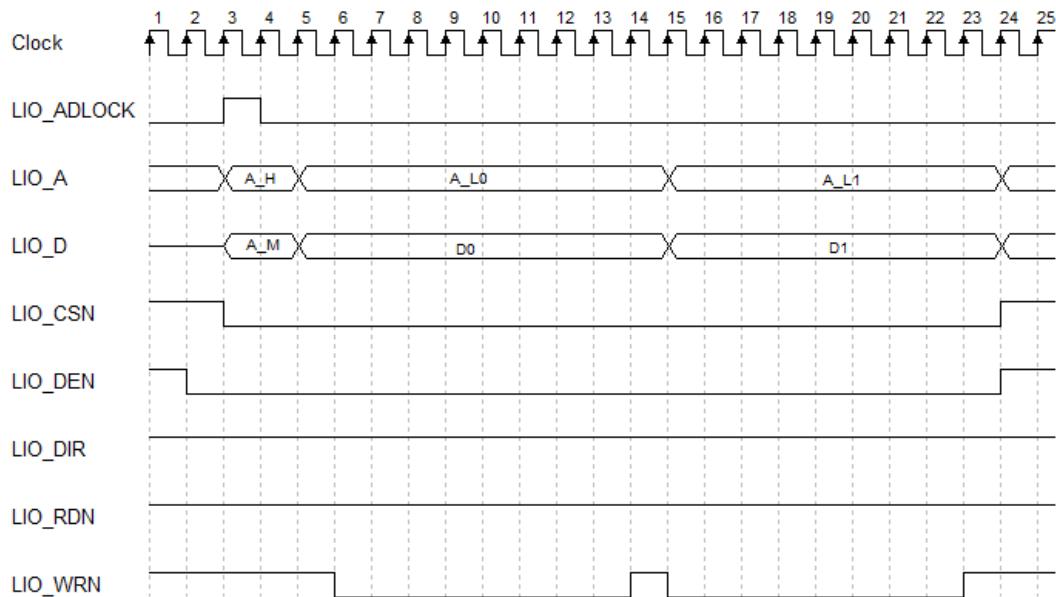


图 11-2LocalIO 写时序

说明：

- 图中 clock 信号实际并不存在，只是为了方便时序描述
- A\_H 代表地址 bit23-bit29(8 位模式)/bit24-bit30(16 位模式)
- A\_M 代表地址 bit7-bit22(8 位模式)/bit8-bit23(16 位模式)
- A\_L 代表低 7 位地址
- 在 big\_mem 设置为 0 时，第 4 拍波形不存在，第 4 拍之后的波形向前推一拍
- LIO\_WRN 和 LIO\_RDN 低有效时间与 LIO clock\_period\_i 设置有关：
  - LIO clock\_period\_i 设置为 1 时-低电平持续 8 拍；
  - LIO clock\_period\_i 设置为 2 时-低电平持续 16 拍；
  - LIO clock\_period\_i 设置为 3/0 时-低电平持续 32 拍；
- 一次 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。

## 12 DDR3 控制器

龙芯 2K1000 处理器内部集成的内存控制器的设计遵守 DDR3 SDRAM 的行业标准 (JESD79-3)。所实现的所有内存读/写操作都遵守 JESD79-3 的规定。

### 12.1 访问地址

DDR3 控制器包括两个地址空间，分别如下：

表 12-1 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； 或当 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 配置请参考表 5-2 通用配置寄存器 0。

### 12.2 DDR3 SDRAM 控制器功能概述

龙芯 2K1000 处理器支持最大 4 个 CS (由 4 个片选信号实现，即两个双面内存条)，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 4，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 2K1000 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

### 12.3 DDR3 SDRAM 读操作协议

DDR3 SDRAM 读操作的协议如图 12-1 所示。在图中命令 (Command，简称 CMD) 由 RAS\_n, CAS\_n 和 WE\_n，共三个信号组成。对于读操作，RAS\_n=1, CAS\_n=0, WE\_n =1。

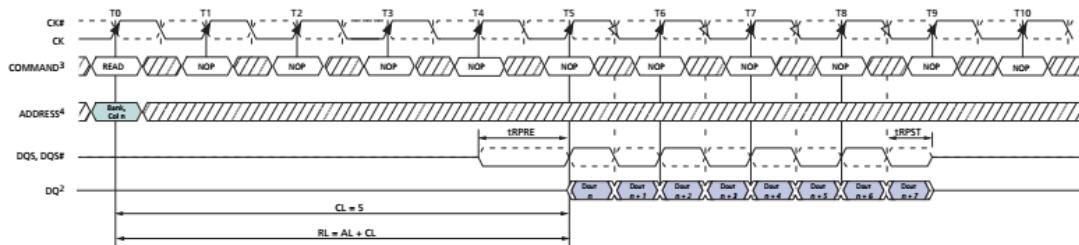


图 12-1 DDR3 SDRAM 读操作协议

上图中, Cas Latency (CL) = 5, Read Latency (RL) = 5, Burst Length = 8。

## 12.4 DDR3 SDRAM 写操作协议

DDR3 SDRAM 写操作的协议如图 12-2 所示。在图中命令 CMD 是由 RAS\_n, CAS\_n 和 WE\_n, 共三个信号组成的。对于写操作, RAS\_n=1, CAS\_n=0, WE\_n=0。另外, 与读操作不同, 写操作需要 DQM 来标识写操作的掩码, 即需要写入的字节数。DQM 与图中 DQs 信号同步。

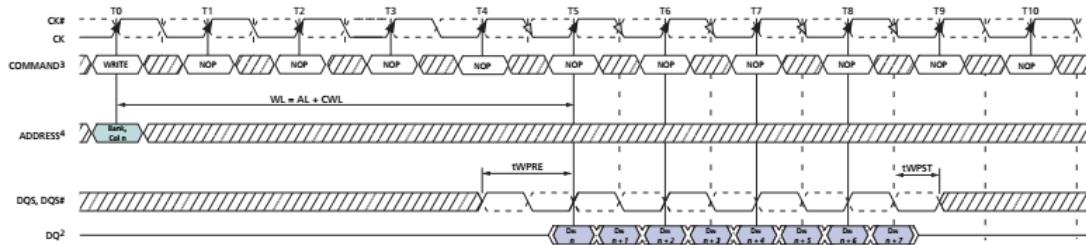


图 12-2 DDR3 SDRAM 写操作协议

上图中, Cas Latency (CL) = 5, Write Latency (WL) = 5, Burst Length = 8。

## 12.5 DDR3 控制器寄存器

DDR3 控制器的配置寄存器内容如表 12-2 所示。

表 12-2 DDR3 控制器配置寄存器

	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	Rddqs_lt_half_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	Rddqs_lt_half_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	Rddqs_lt_half_2	Rddqs_lt_half_2	Wrdqs_lt_half_2	Wrdq_lt_half_2

	63:56	55:48	47:40	39:32	31:24	23:16	15:8	7:0
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
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_otp_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
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
0x178								
0x180	Lvl_resp_0(RD)	Lvl_done(RD)	Lvl_ready(RD)			Lvl_cs	tLVL_DELAY	Lvl_req(WR)
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	

	63:56	55:48	47:40	39:32	31:24	23:16	15:8	7:0						
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						
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									

	63:56	55:48	47:40	39:32	31:24	23:16	15:8	7:0
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	Rdbufbuffer_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	
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								

## 13 GPIO

龙芯 2K1000 共有 60 个 GPIO 引脚，4 个为专用 GPIO，其余 56 个与其他功能复用。下面具体介绍与 GPIO 相关的配置寄存器。

表 13-1 GPIO 配置寄存器

地址	名称	描述
0x1fe10500	GPIO0_OEN	GPIO 输出使能，低有效
0x1fe10508	GPIO1_OEN	保留
0x1fe10510	GPIO0_O	GPIO 输出值
0x1fe10518	GPIO1_O	保留
0x1fe10520	GPIO0_I	GPIO 输入值
0x1fe10528	GPIO1_I	保留
0x1fe10530	GPIO0_INTEN	GPIO 的低 64 位中断使能
0x1fe10538	GPIO1_INTEN	保留

### 13.1 GPIO 方向控制

地址：0x1fe10500

表 13-2 GPIO 方向控制

位域	名称	访问	初值	描述
63:0	GPIO0_OEN	R/W	64'hFFFF_FFFF_FFFF_FFFF	0 为输出，1 为输入

### 13.2 GPIO 输出设置

地址：0x1fe10510

表 13-3 GPIO 输出设置

位域	名称	访问	初值	描述
63:0	GPIO0_O	R/W	0	0 输出低电平，1 输出高电平

### 13.3 GPIO 输入采样

地址：0x1fe10520

表 13-4 GPIO 输入采样

位域	名称	访问	初值	描述
63:0	GPIO0_I	R	-	反映 GPIO 引脚的值

### 13.4 GPIO 中断使能

地址：0x1fe10530

表 13-5 GPIO 中断使能

位域	名称	访问	初值	描述
63:0	GPIO0_INTEN	R/W	0	中断使能位，每一位对应一个 GPIO 引脚

## 13.5GPIO 复用关系

GPIO 与其他功能的复用关系如下表所示：

表 13-6 GPIO 复用关系

GPIO 编号	复用信号	备注
63	NAND_D7	
62	NAND_D6	
61	NAND_D5	
60	NAND_D4	
59	NAND_D3	
58	NAND_D2	
57	NAND_D1	
56	NAND_D0	
55	NAND_RDYn3	
54	NAND_RDYn2	
53	NAND_RDYn1	
52	NAND_RDYn0	
51	NAND_RDn	
50	NAND_WRn	
49	NAND_ALE	
48	NAND_CLE	
47	NAND_CEn3	
46	NAND_CEn2	
45	NAND_CEn1	
44	NAND_CEn0	
43	-	保留
42	-	保留
41	SDIO_CLK	
40	SDIO_CMD	
39	SDIO_DATA3	
38	SDIO_DATA2	
37	SDIO_DATA1	
36	SDIO_DATA0	
35	CAN1_TX	默认为 GPIO 功能，使用 CAN 时需要设置 can_sel[1] 为 1， can_sel 配置参考表 5-2 通用配置寄存器 0 设置。
34	CAN1_RX	
33	CAN0_TX	默认为 GPIO 功能，使用 CAN 时需要设置 can_sel[1] 为 1， can_sel 配置参考表 5-2 通用配置寄存器 0 设置。
32	CAN0_RX	
31	-	保留
30	HDA_SDI2	默认为 GPIO 功能，使用 HDA 时需要设置 hda_sel 为 1， 参

29	HDA_SDI1	考表 5-2 通用配置寄存器 0 设置。
28	HDA_SDI0	
27	HDA_SDO	
26	HDA_RESETn	
25	HDA_SYNC	
24	HDA_BITCLK	
23	PWM3	默认为 GPIO 功能，使用 PWM 时需要设置 pwm_sel[3] 为 1，参考表 5-2 通用配置寄存器 0 设置。
22	PWM2	默认为 GPIO 功能，使用 PWM 时需要设置 pwm_sel[2] 为 1，参考表 5-2 通用配置寄存器 0 设置。
21	PWM1	默认为 GPIO 功能，使用 PWM 时需要设置 pwm_sel[1] 为 1，参考表 5-2 通用配置寄存器 0 设置。
20	PWM0	默认为 GPIO 功能，使用 PWM 时需要设置 pwm_sel[0] 为 1，参考表 5-2 通用配置寄存器 0 设置。
19	I2C1_SDA	默认为 GPIO 功能，使用 I2C 时需要设置 i2c_sel[1] 为 1，参考表 5-2 通用配置寄存器 0 设置。
18	I2C1_SCL	
17	I2C0_SDA	默认为 GPIO 功能，使用 I2C 时需要设置 i2c_sel[1] 为 1，参考表 5-2 通用配置寄存器 0 设置
16	I2C0_SCL	
15	-	保留
14	SATA_LEDn	默认为 GPIO 功能，使用 SATA 时需要设置 sata_sel 为 1，参考表 5-2 通用配置寄存器 0 设置
13	GMAC1_TCTL	默认为 GPIO 功能，使用 GMAC1 时需要设置 gmac1_sel 为 1，gmac1_sel 配置参考表 5-2 通用配置寄存器 0 设置
12	GMAC1_TXD3	
11	GMAC1_TXD2	
10	GMAC1_TXD1	
9	GMAC1_RXD0	
8	GMAC1_RCTL	
7	GMAC1_RXD3	
6	GMAC1_RXD2	
5	GMAC1_RXD1	
4	GMAC1_RXD0	
3	无复用	专用 GPIO 引脚
2	无复用	专用 GPIO 引脚
1	无复用	专用 GPIO 引脚
0	无复用	专用 GPIO 引脚

# 14 APB 设备 (Dev 2)

APB 设备的配置空间基本信息如下：

表 14-1 APB 配置访问信息

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)	备注
APB	0x0	0x2	0x0	0xFFFF	0xFE_0000_1000	-

## 14.1 内部设备地址路由

APB 总线控制器下挂载了 UART 控制器、CAN 控制器、I2C 控制器、PWM 控制器、RTC 控制器、HPET 控制器、NAND 控制器、ACPI 控制器、DES 控制器、AES 控制器、RSA 控制器、RNG 控制器、SDIO 控制器和 I2S 控制器。由访问地址的[15:12]来进行路由，具体如下表所示。

表 14-2 APB 设备地址译码

地址[15:12]	设备	备注
0x0	UART * 12 CAN * 2	用地址的[11:8]做下一级地址译码： 0x0: UART0 0x1: UART1 0x2: UART2 0x3: UART3 0x4: UART4 0x5: UART5 0x6: UART6 0x7: UART7 0x8: UART8 0x9: UART9 0xA: UART10 0xB: UART11 0xC: CAN0 0xD: CAN1 0xE: NULL 0xF: NULL
0x1	I2C * 2	用地址[11]做下一级地址译码： 0x0: I2C0 0x1: I2C1
0x2	PWM * 4	用地址[7:4]做下一级地址译码： 0x0: PWM0 0x1: PWM1 0x2: PWM2 0x3: PWM3
0x3	Reserved	
0x4	HPET	
0x5	Reserved	
0x6	NAND	

0x7	ACPI/RTC	ACPI 的偏移地址为 0x7000 RTC 的偏移地址为 0x7800
0x8	DES	
0x9	AES	
0xA	RSA	
0xB	RNG	
0xC	SDIO	
0xD	I2S	
0xE	专用通信接口	
0xF	NULL	

后续章节将分别对 APB 总线的各个设备控制器进行介绍。

# 15 UART 控制器

## 15.1 概述

2K1000 集成了 12 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。

其中，UART0、UART3、UART4、UART5 复用 UART0 接口；UART1、UART6、UART7、UART8 复用 UART1 接口；UART2、UART9、UART10、UART11 复用 UART2 接口。参见 2.25 节。

## 15.2 访问地址及引脚复用

各个 UART 控制器内部寄存器的物理地址构成如下：

表 15-1 UART 控制器物理地址构成

地址位	构成	备注
[27:16]	BAR_BASE	Device 2 的基址寄存器值
[15:12]	0	固定为 0
[11:08]	UART 号	0x0 - 0xB，分别表示各个 UART 号
[07:00]	REG	内部寄存器地址

对于各个 UART，使用时要注意将对应的引脚设置为相应功能。

与 UART 相关的引脚设置寄存器为 5.2 节中的 uart1\_sel、uart2\_sel、uart0\_enable、uart1\_enable 及 uart2\_enable。

## 15.3 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。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。

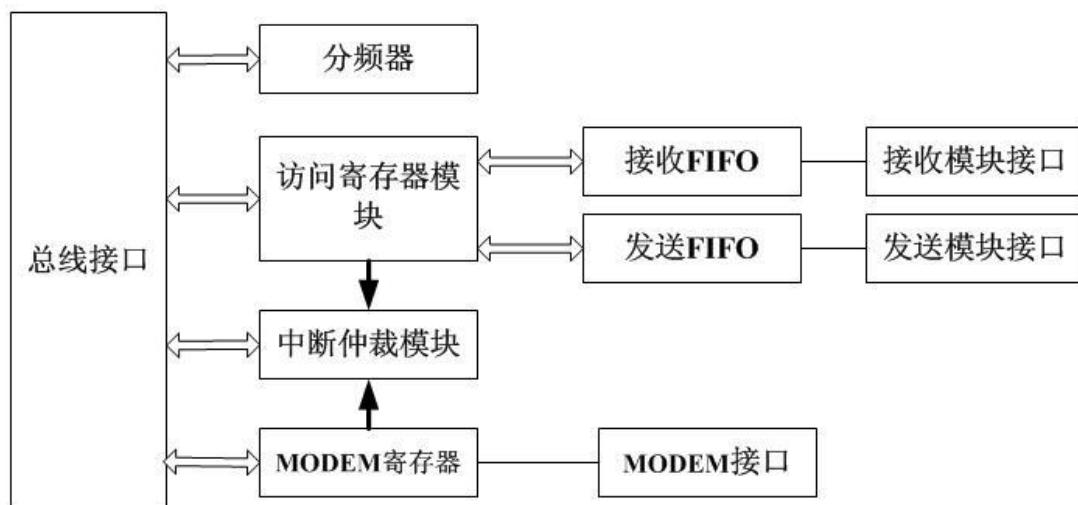


图 15-1 UART 控制器结构

## 15.4 寄存器描述

### 15.4.1 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

表 15-2 UART 数据寄存器

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

### 15.4.2 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

表 15-3 UART 中断使能寄存器

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

### 15.4.3 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

表 15-4 UART 中断标识寄存器

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

中断控制功能表:

表 15-5 UART 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	1 <sup>st</sup>	接收线路状态	奇偶、溢出或帧错误，或打断中断	读 LSR
0	1	0	2 <sup>nd</sup>	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	2 <sup>nd</sup>	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	3 <sup>rd</sup>	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	4 <sup>th</sup>	Modem 状态	CTS, DSR, RI or DCD.	读 MSR

### 15.4.4 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

表 15-6 UART 的 FIFO 控制寄存器

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

#### 15.4.5 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

表 15-7 UART 线路控制寄存器

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

#### 15.4.6 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

表 15-8 UART 的 MODEM 控制寄存器

位域	位域名称	位宽	访问	描述
7:5	Reserved	3	W	保留
4	Loop	1	W	回环模式控制位 ‘0’ - 正常操作 ‘1’ - 回环模式。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

#### 15.4.7 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

表 15-9 UART 线路状态寄存器

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 ‘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

位域	位域名称	位宽	访问	描述
				‘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。

#### 15.4.8 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

表 15-10 UART 的 MODEM 状态寄存器

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

#### 15.4.9 分频锁存器

中文名： 分频锁存器 1

寄存器位宽： [7: 0]

偏移量： 0x00

复位值： 0x00

表 15-11 UART 分频锁存器 1

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

中文名： 分频锁存器 2

寄存器位宽： [7: 0]

偏移量： 0x01

复位值： 0x00

表 15-12 UART 分频锁存器 2

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

# 16 CAN

龙芯 2K1000 集成了两路 CAN 接口控制器。CAN 总线是由发送数据线 TX 和接收数据线 RX 构成的串行总线，可发送和接收数据。器件与器件之间进行双向传送，最高传送速率 1Mbps。

## 16.1 访问地址及引脚复用

两个 CAN 控制器内部寄存器的物理地址构成如下：

表 16-1 CAN 内部寄存器物理地址构成

地址位	构成	备注
[27:16]	BAR_BASE	Device 2 的基址寄存器值
[15:12]	0	固定为 0
[11:08]	CAN 编号	分别为 0xC 和 0xD
[07:00]	REG	内部寄存器地址

对于 CAN 模块，使用时要注意将对应的引脚设置为相应功能。

与 CAN 相关的引脚设置寄存器为 5.1 节中的 can\_sel。

## 16.2 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 16-2 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	控制
1		FF	命令	FF	命令
2		状态	—	状态	—
3		FF	—	中断	—
4		FF	—	验收代码	验收代码
5		FF	—	验收屏蔽	验收屏蔽
6		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

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

### 16.2.1 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 13-3 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
----	------	----	----	----

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

### 16.2.2 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 13-4 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

### 16.2.3 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 13-5 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

### 16.2.4 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 13-6 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

### 16.2.5 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 16-7 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

### 16.2.6 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 16-8 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

### 16.2.7 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 13-9 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	发送缓冲器	识别码字节 1	ID(10-3)

11		识别码字节 2	ID(2-0), RTR,DLC
12		TX 数据 1	TX 数据 1
13		TX 数据 2	TX 数据 2
14		TX 数据 3	TX 数据 3
15		TX 数据 4	TX 数据 4
16		TX 数据 5	TX 数据 5
17		TX 数据 6	TX 数据 6
18		TX 数据 7	TX 数据 7
19		TX 数据 8	TX 数据 8

### 16.2.8 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。

## 16.3 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 13-10 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,DL C[3:0]}	TX 帧信息 {IDE,RTR,2'h0,D LC[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,da ta0[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) Data0 (非扩 展帧，单滤波)	验收代码 2
19	RX_Id[12:5] (扩 展)	TX_Id[12:5] (扩 展)	验收代码 3	验收代码 3

CAN 地址	工作模式		复位模式	
	帧) RX data0(非扩展帧)	帧) TX data0(非扩展帧)	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)	
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	—

### 16.3.1 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 13-11 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

### 16.3.2 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 13-12 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)

### 16.3.3 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 13-13 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

#### 16.3.4 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 13-14 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

#### 16.3.5 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 13-15 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

#### 16.3.6 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 13-16 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RTTR位

### 16.3.7 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 16-17 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

### 16.3.8 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 16-18 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

### 16.3.9 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 16-19 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

### 16.3.10 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。

### 16.3.11 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 16-20 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

## 16.4 公共寄存器

1bit time = internal\_clock\_time \* ((BRP + 1) \* 2) \* (1 + (TESG2 + 1) + (TESG1 + 1))

### 16.4.1 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 16-21 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

### 16.4.2 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 16-22 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

### 16.4.3 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 16-23 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）

# 17 I2C 控制器

## 17.1 概述

本章给出 I2C 的详细描述和配置使用。本系统芯片集成了 I2C 接口，主要用于实现两个器件之间数据的交换。I2C 总线是由数据线 SDA 和时钟 SCL 构成的串行总线，可发送和接收数据。器件与器件之间进行双向传送，最高传送速率 400kbps。

## 17.2 访问地址及引脚复用

两个 I2C 控制器内部寄存器的物理地址构成如下：

地址位	构成	备注
[27:16]	BAR_BASE	Device 2 的基址寄存器值
[15:12]	1	固定为 1
[11]	I2C 编号	0x0 代表 I2C0; 0x1 代表 I2C1
[10:03]	0	保留
[02:00]	REG	内部寄存器地址

对于 I2C 模块，使用时要注意将对应的引脚设置为相应功能。

与 I2C 相关的引脚设置寄存器为 5.1 节中的 i2c\_sel。

## 17.3 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 APB 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。

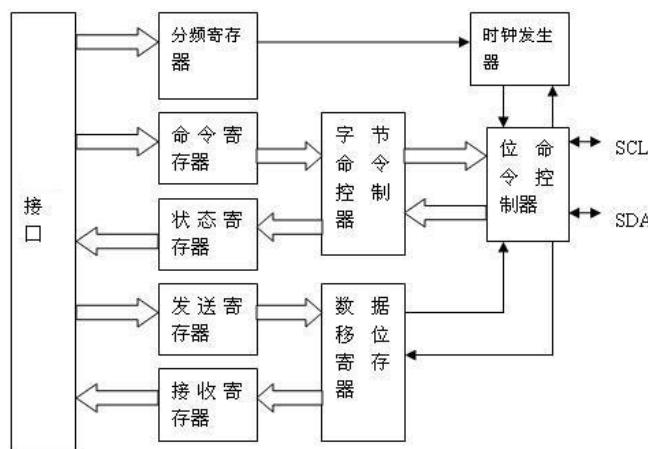


图 17-1 I2C 主控制器结构

## 17.4 I2C 控制器寄存器说明

本芯片集成了两个 I2C 控制器 I2C-0、I2C-1。

### 17.4.1 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

### 17.4.2 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

$$\text{Prcescale} = \text{clock\_a}/(4*\text{clock\_s})-1$$

### 17.4.3 控制寄存器 (CTR)

中文名：控制寄存器

寄存器位宽: [7: 0]

偏移量: 0x02

复位值: 0x00

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

#### 17.4.4 发送数据寄存器 (TXR)

中文名: 发送寄存器

寄存器位宽: [7: 0]

偏移量: 0x03

复位值: 0x00

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

#### 17.4.5 接受数据寄存器 (RXR)

中文名: 接收寄存器

寄存器位宽: [7: 0]

偏移量: 0x03

复位值: 0x00

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

#### 17.4.6 命令控制寄存器 (CR)

中文名: 命令寄存器

寄存器位宽: [7: 0]

偏移量: 0x04

复位值: 0x00

位域	位域名称	位宽	访问	描述
7	STA	1	W	产生 START 信号
6	STO	1	W	产生 STOP 信号
5	RD	1	W	产生读信号
4	WR	1	W	产生写信号
3	ACK	1	W	产生应答信号

2:1	Reserved	2	W	保留
0	IACK	1	W	产生中断应答信号

都是在 I2C 发送数据后硬件自动清零。对这些位读操作时候总是读回 ‘0’。 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。

#### 17.4.7 状态寄存器 (SR)

中文名：状态寄存器

寄存器位宽： [7: 0]

偏移量： 0x04

复位值： 0x00

位域	位域名称	位宽	访问	描述
7	RxACK	1	R	收到应答位 1 没收到应答位 0 收到应答位
6	Busy	1	R	I2c 总线忙标志位 1 总线在忙 0 总线空闲
5	AL	1	R	当 I2C 核失去 I2C 总线控制权时，该位置 1
4:2	Reserved	3	R	保留
1	TIP	1	R	指示传输的过程 1 表示正在传输数据 0 表示数据传输完毕
0	IF	1	R	中断标志位，一个数据传输完，或另外一个器件发起数据传输，该位置 1

# 18 PWM 控制器

## 18.1 概述

2K1000 芯片里实现了四路脉冲宽度调节/计数控制器，以下简称 PWM。每一路 PWM 工作和控制方式完全相同。每路 PWM 有一路脉冲宽度输出信号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。

## 18.2 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：

地址位	构成	备注
[27:16]	BAR_BASE	Device 2 的基址寄存器值
[15:12]	2	固定为 2
[11:08]	0	保留
[07:04]	PWM 编号	取值 0x0-0x3 分别代表 PWM0-PWM3
[03:00]	REG	内部寄存器地址

对于 PWM 模块，使用时要注意将对应的引脚设置为相应功能。与 PWM 相关的引脚设置寄存器为 5.1 节中的 pwm\_sel。

## 18.3 寄存器描述

每路控制器共有五个寄存器，具体描述如下：

表 18-1 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 18-2 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：CNTR 用来计数 置 0 时：CNTR 停止计数（输出保持）
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_pulse 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

## 18.4 功能说明

### 18.4.1 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。

### 18.4.2 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。

#### 18.4.3 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）：

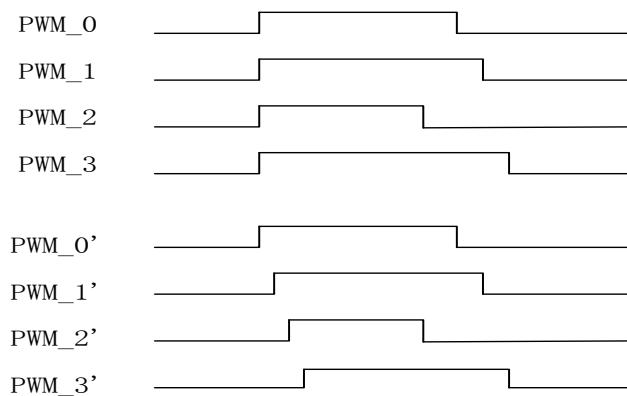


图 18-1 防死区功能

# 19 HPET 控制器

## 19.1 概述

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。这个计时器以固定的频率 (125MHz) 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。

## 19.2 访问地址

HPET 控制器内部寄存器的物理地址构成如下：

地址位	构成	备注
[27:16]	BAR_BASE	Device 2 的基地址寄存器值
[15:12]	4	固定为 4
[01:00]	REG	内部寄存器地址

## 19.3 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_PER IOD	Main Counter Tick Period: 这个域标示了主计时器的计时频率，以 fs (10^-15s) 为单位。这个值必须大于 0，且小于或等于 0x05F5E100 (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，2K1000 中有三个定时器，因此这个域的值是 2。	RO
7:0	REV_ID	版本号；不可为 0	RO

### General Configuration Register

位	名称	描述	读写特性
63: 1	Reserved		
0	ENABLE_CNF	Overall 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触发模式是电平触发还是边沿触发：	R/WC



		<p>如果是电平触发模式：</p> <p>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。</p> <p>往这位写 0，则无意义。</p> <p>如果边沿触发模式：</p> <p>软件将忽略这位。软件通常往这位写 0.</p>	
--	--	--	--

各个定时器的中断触发模式由各自 Configuration 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 Register

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  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。  1: 使能定时器产生周期性中断	R/W

		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	<p>Timer N Interrupt Type (N 为 0-2) :</p> <p>0: 定时器的中断触发模式为边沿触发；这意味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。</p> <p>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)。</p>	
0	Reserved		

### Timer N Comparator Value Register

位	名称	描述	读写特性
63: 32	Reserved		
31: 0	Tn_Com_VAL	<p>Tn_ComPARATOR value (N 为 0-2) : 定时器比较器的值；</p> <p>当对应的定时器配置为非周期性模式时：</p> <ul style="list-style-type: none"> <li>◆ 这个寄存器的值将与主计时器寄存器的值做比较；</li> <li>◆ 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）。</li> <li>◆ 比较器的值不会因为中断的产生而发生变化</li> </ul> <p>若对应的定时器配置为周期性模式时：</p> <ul style="list-style-type: none"> <li>• 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）；</li> <li>• 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；</li> <li>• 比较器的值被硬件修改为 0x246h；</li> <li>• 当主计时器的值达到 0x246h 时，产生另外一个中断；</li> <li>• 比较器的值被硬件修改为 0x369h。</li> <li>◆ 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。</li> </ul>	R/W

# 20 NAND 控制器

## 20.1 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 SPI_CK、SPI_CS、 NAND_RD 全部下拉	外部 NAND FLASH 第一个 page 的数据普通原始数据

## 20.2 访问地址及引脚复用

NAND 控制器内部寄存器的物理地址构成如下：

地址位	构成	备注
[27:16]	BAR_BASE	Device 2 的基址寄存器值
[15:12]	6	固定为 6
[11]	0	保留
[10:00]	REG	内部寄存器地址

对于 NAND 模块，使用时要注意将对应的引脚设置为相应功能。

与 NAND 相关的引脚设置寄存器为 5.1 节中的 nand\_sel。

## 20.3 NAND 寄存器配置描述

NAND 内部的寄存器设置如下：

偏移地址	寄存器名称
0x00	NAND_CMD
0x04	ADDR_C
0x08	ADDR_R
0x0C	NAND_TIMING
0x10	ID_L
0x14	STATUS & ID_H
0x18	NAND_PARAMETER
0x1C	NAND_OP_NUM
0x20	CS_RDY_MAP
0x40	DMA access address

### 20.3.1 命令寄存器 NAND\_CMD (偏移地址 0x00)

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_R_EQ	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
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

### 20.3.2 页内偏移地址寄存器 ADDR\_C (偏移地址 0x04)

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

### 20.3.3 页地址寄存器 ADDR\_R (偏移地址 0x08)

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

### 20.3.4 时序寄存器 NAND\_TIMING (偏移地址 0x0C)

位	位域名	读写	描述
31:16		R/-	Reserved

45 8 :7 0	Hold cycle Wait cycle	R/W R/W	NAND 命令有效需等待的周期数，缺省 4 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
--------------	--------------------------	------------	--

### 20.3.5 ID 寄存器 ID\_L (偏移地址 0x10)

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

### 20.3.6 ID 和状态寄存器 STATUS & ID\_H (偏移地址 0x14)

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

### 20.3.7 参数配置寄存器 NAND\_PARAMETER (偏移地址 0x18)

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

### 20.3.8 操作数量寄存器 NAND\_OP\_NUM (偏移地址 0x1C)

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

### 20.3.9 映射寄存器 CS\_RDY\_MAP (偏移地址 0x20)

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\_rdy1/2/3 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

### 20.3.10 DMA 读写数据寄存器 DMA\_ADDRESS (偏移地址 0x40)

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

## 20.4 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

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

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						
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						
Page 2	op						
Page 3	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。

## 20.5 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

## 20.6 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：

1. 设置 NAND\_CMD[12]为 1，表示接下来写操作为 ECC 写
2. 设置 NAND\_CMD[8]和 NAND\_CMD[9]为 1，表示同时操作 main 区和 spare 区
3. 设置 NAND\_OP\_NUM 为 (main 区大小+main 区数据对应的 ECC+2)
4. 设置 ADDR\_C 为 0

5. 设置其他寄存器
6. 设置 NAND\_CMD[2]表示写操作
7. 设置 NAND\_CMD[0]开始写操作
8. 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。

## 20.7 支持 NAND 型号

本节列出经过验证的可支持的 NAND 型号，其它类型的 NAND 颗粒未经验证，不保证与本控制器兼容。

- 镁光 L84A\_64Gb\_128Gb\_256Gb\_AsyncSync\_NAND: 可用于存储，不可用于启动
- 三星 K9F1G08U0C: 可用于存储和启动

# 21 电源管理模块

## 21.1 概述

龙芯 2K1000 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(USB, GMAC, 电源开关等)
- 动态性能功耗控制，支持处理器核 DVFS 控制，支持动态关闭媒体解码协处理器电源。
- 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。

## 21.2 访问地址

电源管理模块内部寄存器的物理地址构成如下：

地址位	构成	备注
[27:16]	BAR_BASE	Device 2 的基地址寄存器值
[15:12]	7	固定为 7
[11]	ACPI/RTC	0x0 代表 ACPI; 0x1 代表 RTC
[10:00]	REG	内部寄存器地址

## 21.3 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。

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)

### ■ 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	<b>PWRBTN_LVL - RO</b> 该位指示当前 PWRBTNn 信号状态。
24	<b>PWRTYP - RO</b> 该位指示供电模式 1: AC (适配器) 0: Battery (电池)
23:9	保留
8:0	<b>TEMP_NOW - RO.</b> 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
31:8	保留
7	<b>USB_GMAC_OK - R/W</b> 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	<b>CTT_STS - R/WC</b> 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	<b>CTT_EN - R/W</b> 使能 temperature trip 保护机制
4	<b>LID_OPEN - RO</b> 显示屏状态检测位 1: 显示屏打开 0: 显示屏合上
3	保留
2	<b>SRS (System Reset Status) - R/WC</b> 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	<b>PWROK_FLR (PWROK Failure) - R/WC</b> 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	<b>DRAM_INIT - R/W</b> 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	<b>WOL_EN - R/W</b> 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	<b>WOL_BAT_EN - R/W</b> 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	<b>S3_ASSERTION_WIDTH - R/W</b> 这 2 bit 值代表 S3n 信号从有效到重新无效的最长时间间隔。 11: 1s 10: 125ms 01: 1ms 00: 60us
4:3	<b>S4_ASSERTION_WIDTH - R/W</b> 这 2 bit 值代表 S4n 信号从有效到重新无效的最长时间间隔。 11: 4 s 10: 2 s 01: 1 s 00: 125 us
2	<b>S4_ASSERTION_EN - R/W</b> 0: S4n 信号从有效到重新无效的间隔为几个 RTC 周期。 1: S4n 信号从有效到重新无效的间隔为 S4_ASSERTION_WIDTH 决定。
1	<b>PWR_FLR (Power Failure) - R/WC.</b> 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	<b>AFTERG3_EN - R/W</b> 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	<b>WAK_STS (Wake Status) - R/WC</b> 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	<b>PCIEXP_WAKE_STS - R/WC.</b> 1: PCIE 唤醒事件发生 0: 软件写 1 将该位清除	Resume
13:12	Reserved	
11	<b>PRBTNOR_STS (Power Button Override Status) - R/WC</b> 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 置位 1。	RTC
10	<b>RTC_STS (RTC Status) - R/WC</b> 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	<b>PWRBTN_STS (Power Button Status) - R/WC</b> 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
7:1	保留	
0	<b>TMROF_STS (PM Timer Overflow Status) - R/WC</b> 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	<b>PCIEXP_WAKE_DIS - R/W</b> 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	<b>RTC_EN (RTC Event Enable) - R/W</b> RTC 唤醒和中断使能	rtc
9	保留	
8	<b>PWRBTN_EN (Power Button Enable) - R/W.</b> 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	<b>TMROF_EN (PM Timer Overflow Enable) - R/W.</b>	SOC

	如果该位置位, TMROF_STS 将产生中断。	
--	--------------------------	--

### ■ 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	<b>SLP_EN (Sleep Enable) - R/W.</b> 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	<b>SLP_TYP (Sleep Type) - R/W.</b> 该 3bit 表示系统的休眠状态。 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。	rtc
9:1	Reserved	
0	<b>INT_EN - R/W</b> 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	<b>TMR_VAL (Timer Value) - RO.</b> 计数器计数, 周期 8ns。当 23 位翻转时, 置位 TNROF_STS 位。 推荐使用 HPET。

### ■ P\_CNT : Processor Control Register

地址偏移	电压域	属性
0x1C	SOC	R/W, RO

位域	描述
31:5	保留
4	<b>THTL_EN - R/W</b> 当系统处于 C0 状态时, 使能该位可以通过 clock throttling 对 CPU 时钟进行控制。
3:1	<b>THTL_DTY - R/W</b> 该 3-bit 确定 throttling 的百分比, throttling 周期是 1024 APB 时钟。 000: no throttling 001: 87.5% throttling 010: 75% throttling 011: 62.5% throttling 100: 50% throttling 101: 37.5% throttling 110: 25% throttling 111: 12.5% throttling
0	保留

### ■ GPE0\_STS : General Purpose Event0 Status Register

地址偏移	电压域	属性
0x28	RESUME	R/WC

位域	描述
31:16	保留
15:10	<b>USB[6:1]_STS - R/WC.</b> 只有第 10 位有意义，15:11 位暂无意义。 0: 软件写 1 将该位清除。 1: 当 USB 发生 wake 事件时这些位被置位，当 USBx_EN 位有效时，产生唤醒事件或中断。
9	保留
8	<b>RI_STS - R/WC.</b> 0: 软件写 1 将该位清除。 1: 当 RIn 信号有效时被置位。
7	<b>BATLOW_STS - R/WC.</b> 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	<b>GMAC1_STS - R/WC.</b> 0: 软件写 1 将该位清除。 1: 当 GMAC1 发生 wake 事件时这些位被置位，当 GMAC1_EN 位有效时，产生唤醒事件或中断。
5	<b>GMAC0_STS - R/WC.</b> 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	<b>LID_STS - R/WC.</b> 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	<b>CTW_STS - R/WC.</b> CPU thermal warning 发生
2	<b>CTA_STS - R/WC.</b> CPU thermal alert 发生
1	<b>PWRSWITCH_STS - R/WC.</b> 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
31:16	保留	
15:10	<b>USB[6:1]_EN - R/W.</b> 0: 无效 1: 使能 USBx_STS 产生唤醒事件，当回到 S0 将产生中断信号。	
9	保留	
8	<b>RI_EN - R/W.</b> 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	<b>BATLOW_EN - R/W.</b> 0: 无效 1: 使能 BATLOWn 产生中断事件。	RTC
6	<b>GMAC1_EN - R/W.</b> 0: 无效 1: 使能 GMAC1_STS 产生唤醒事件，当回到 S0 将产生中断信号。	RTC
5	<b>GMAC0_EN - R/W.</b> 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	<b>LID_EN - R/W.</b> 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	<b>CTW_EN - R/W</b> 使能 CPU THERMAL WARNING 产生中断。	
2	<b>CTA_EN - R/W</b> 使能 CPU THERMAL ALERT 产生中断。	
1	<b>PWRSWITCH_EN - R/W</b> 使能 PWRSWITCH_STS 产生中断。	
0	<b>LID_POL - R/W</b> 该位设置 LID 的极性。	

### ■ 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	<b>WD_EN - R/W</b> Watch dog 功能使能
0	<b>OS_RST - R/W</b> 软件写该位使系统复位。

### ■ 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

### ■ 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 。

### ■ THSENS\_CNT : CPU Thermal Sensor Control Register

地址偏移	电压域	属性
0x4C	SOC	R/W

位域	描述
31:24	<b>WARNING_TMP - R/W</b> CPU 温度超过该值, 如果被使能, 将产生中断。如果温度位降低到该值以下, 不会重复产生中断。 推荐操作: 系统采取降低功耗操作。
23:16	<b>ALERT_TMP - R/W</b> CPU 温度超过该值, 如果被使能, 将产生中断。如果温度位降低到该值以下, 不会重复产生中断。 推荐操作: 采取正常关闭系统操作

15:8	<b>TRIP_TMP - R/W</b> CPU 温度超过该值，如果被使能，系统无条件进入 G2/S5 状态。
7:0	<b>OFFSET - R/W</b> 设置温度传感器的校正偏移（软件），最高位为符号位。

### ■ 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

### ■ DPM\_CFG : Dynamic Power Management Configuration

地址偏移	电压域	属性
0x400	SOC	R/W

位域	描述
31:7	保留
6	<b>DPM_EN_GMAC1 - R/W</b> 使能 GMAC1 的时钟电源管理
5	<b>DPM_EN_GMAC0 - R/W</b> 使能 GMAC0 的时钟电源管理
4	<b>DPM_EN_SATA - R/W</b> 使能 SATA 的时钟电源管理
3	<b>DPM_EN_PCIE1 - R/W</b> 使能 PCIe1 的时钟电源管理
2	<b>DPM_EN_PCIE0 - R/W</b> 使能 PCIe0 的时钟电源管理
1	<b>DPM_EN_GPU - R/W</b> 使能 GPU 的时钟电源管理
0	<b>DPM_EN_DC - R/W</b> 使能 DC 的时钟电源管理

### ■ DPM\_STS : Dynamic Power Management Status

地址偏移	电压域	属性
0x404	SOC	RO

位域	描述
31	保留
30	<b>DPM_RUNNING_GMAC1 - R/W</b> GMAC1 正在进行时钟电源状态转换
29	<b>DPM_RUNNING_GMAC0 - R/W</b> GMAC0 正在进行时钟电源状态转换
28	<b>DPM_RUNNING_SATA - R/W</b> SATA 正在进行时钟电源状态转换
27	<b>DPM_RUNNING_PCIE1 - R/W</b> PCIE1 正在进行时钟电源状态转换
26	<b>DPM_RUNNING_PCIE0 - R/W</b>

	PCIE0 正在进行时钟电源状态转换
25	<b>DPM_RUNNING_GPUDC - R/W</b> GPU 和 DC 正在进行时钟电源状态转换
24	<b>DPM_RUNNING_GPUDC - R/W</b> GPU 和 DC 正在进行时钟电源状态转换
23:14	保留
13:12	<b>DPM_STS_GMAC1 - R/W</b> GMAC1 的时钟电源状态 00: D0 ALL_ON 正常工作 01: D1 STP_CLK 停时钟 11: D3 ALL_OFF 电源关断 (保留, 未实现)
11:10	<b>DPM_STS_GMAC0 - R/W</b> GMAC0 的时钟电源状态 00: D0 ALL_ON 正常工作 01: D1 STP_CLK 停时钟 11: D3 ALL_OFF 电源关断 (保留, 未实现)
9:8	<b>DPM_STS_SATA - R/W</b> SATA 的时钟电源状态 00: D0 ALL_ON 正常工作 01: D1 STP_CLK 停时钟 11: D3 ALL_OFF 电源关断 (保留, 未实现)
7:6	<b>DPM_STS_PCIE1 - R/W</b> PCIE1 的时钟电源状态 00: D0 ALL_ON 正常工作 01: D1 STP_CLK 停时钟 11: D3 ALL_OFF 电源关断 (保留, 未实现)
5:4	<b>DPM_STS_PCIE0 - R/W</b> PCIE0 的时钟电源状态 00: D0 ALL_ON 正常工作 01: D1 STP_CLK 停时钟 11: D3 ALL_OFF 电源关断 (保留, 未实现)
3:2	<b>DPM_STS_GPU - R/W</b> GPU 的时钟电源状态 00: D0 ALL_ON 正常工作 01: D1 STP_CLK 停时钟 11: D3 ALL_OFF 电源关断 (保留, 未实现)
1:0	<b>DPM_STS_DC - R/W</b> DC 的时钟电源状态 00: D0 ALL_ON 正常工作 01: D1 STP_CLK 停时钟 11: D3 ALL_OFF 电源关断 (保留, 未实现)

## ■ DPM\_CNT : Dynamic Power Management Control

地址偏移	电压域	属性
0x408	SOC	R/W

位域	描述
31:20	保留
19:18	<b>DPM_PCIE1(CG) - RW</b> PCIE1 的两个 PORT 的独立 clock gating 使能
17:14	<b>DPM_PCIE0(CG) - RW</b> PCIE0 的四个 PORT 的独立 clock gating 使能
13:12	<b>DPM_TGT_GMAC1 - R/W</b> GMAC1 的时钟电源控制 00: D0 ALL_ON 正常工作 01: D1 STP_CLK 停时钟 11: D3 ALL_OFF 电源关断 (保留, 未实现)
11:10	<b>DPM_TGT_GMAC0 - R/W</b>

	GMAC0 的时钟电源状态 00: D0 ALL_ON 正常工作 01: D1 STP_CLK 停时钟 11: D3 ALL_OFF 电源关断（保留，未实现）
9:8	<b>DPM_TGT_SATA - R/W</b> SATA 的时钟电源状态 00: D0 ALL_ON 正常工作 01: D1 STP_CLK 停时钟 11: D3 ALL_OFF 电源关断（保留，未实现）
7:6	<b>DPM_TGT_PCIE1 - R/W</b> PCIE1 的时钟电源状态 00: D0 ALL_ON 正常工作 01: D1 STP_CLK 停时钟 11: D3 ALL_OFF 电源关断（保留，未实现）
5:4	<b>DPM_TGT_PCIE0 - R/W</b> PCIE0 的时钟电源状态 00: D0 ALL_ON 正常工作 01: D1 STP_CLK 停时钟 11: D3 ALL_OFF 电源关断（保留，未实现）
3:2	<b>DPM_TGT_GPU - R/W</b> GPU 的时钟电源状态 00: D0 ALL_ON 正常工作 01: D1 STP_CLK 停时钟 11: D3 ALL_OFF 电源关断（保留，未实现）
1:0	<b>DPM_TGT_DC - R/W</b> DC 的时钟电源状态 00: D0 ALL_ON 正常工作 01: D1 STP_CLK 停时钟 11: D3 ALL_OFF 电源关断（保留，未实现）

## ■ DVFS\_CFG : Dynamic Voltage Frequency Scaling Config Register

地址偏移	电压域	属性
0x410	SOC	R/W, RO

位域	描述
31:28	保留
27:22	<b>VID_DEFAULT - RO</b> 上电时默认 VID 值
21:12	<b>VOLT_ADJUST_TIME - R/W</b> 该部分值表示 DVFS 电压转换时的延时值，该延时值期间可进行保护操作。 范围：0—1000us。
11:8	<b>FREQ_ADJUST_TIME - R/W</b> 该部分值表示 DVFS 电压转换时的延时值，单位为 apb 总线时钟周期
7:6	<b>DVFS_CNT_1 (DVFS Control 1) - R/W</b> DVFS 转换时何时进行保护操作 00: 不进行保护操作。 01: 电压转换阶段进行保护操作。 10: 频率变换是进行保护操作。 11: 电压和频率转换阶段都进行保护操作。
5:4	<b>DVFS_CNT_0 (DVFS Control 2) - R/W</b> DVFS 转换时保护操作类型 00: 停时钟。 01: 使用备份时钟。 10: 保留。 11: 对时钟设置不做操作。
3	保留
2	<b>VID_VALID - R/W</b> 使能电压控制

1	<b>PWROK_MASK_EN - R/W</b> 如果有效，在 DVFS 电压转换阶段时屏蔽 PWROK 信号
0	<b>DVFS_EN - R/W</b> 该位使能处理器核 DVFS 功能 0: 无效 DVFS 使能 1: 使能 DVFS 功能

### ■ DVFS\_STS : Dynamic Voltage Frequency Scaling Status Register

地址偏移	电压域	属性
0x414	SOC	RO

位域	描述
31:23	保留
22:16	<b>FEQ_STS - RO</b> CPU 分频器分频值。
9:4	<b>VID_STS - RO</b> 当前 VID 值
3:1	保留
0	<b>CPU_DVFS_STS(DVFS status) - RO.</b> 0: DVFS 控制器空闲，可进行 DVFS 操作 1: 系统正在进行 DVFS 转换。保留

### ■ DVFS\_CNT : Dynamic Voltage Frequency Scaling Control Register

地址偏移	电压域	属性
0x418	SOC	R/W

位域	描述
31:23	保留
22:16	<b>FEQ_TGT - R/W</b> 修改 CPU 分频器分频值。 22:update en 21:DIV enable 20:16 DIV number
15:10	保留
9:4	<b>VID_TGT - R/W</b> 设置目标电压值（目标 VID）
3	保留
2	<b>DVFS_POL - R/W</b> 写 1 表示需提升性能，升压升频。 写 0 表示降低性能节约功耗，降压降频。
1	<b>VID_UPDATE_EN R/W</b> 如果不改变电压 VID 值，将该位置 0（即只进行频率转换）
0	<b>DVFS_START - R/W</b> 写 1 开始一个 DVFS 转换。

# 22 RTC

## 22.1 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。

## 22.2 访问地址

电源管理模块内部寄存器的物理地址构成如下：

地址位	构成	备注
[27:16]	BAR_BASE	Device 2 的基址寄存器值
[15:12]	7	固定为 7
[11]	ACPI/RTC	0x0 代表 ACPI; 0x1 代表 RTC
[10:00]	REG	内部寄存器地址

## 22.3 寄存器描述

RTC 模块寄存器和 ACPI 模块位于同一地址空间内，其偏移为 0x800，其基地址由 BAR 给定，所有寄存器位宽均为 32 位。

### 22.3.1 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_rtctrl	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

### 22.3.2 SYS\_TOYWRITE0

中文名: TOY 计数器低 32 位数值

寄存器位宽: [31: 0]

偏移量: 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

### 22.3.3 SYS\_TOYWRITE1

中文名: TOY 计数器高 32 位数值

寄存器位宽: [31: 0]

偏移量: 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

### 22.3.4 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		月, 范围 1~12
25:21	TOY_DAY	R		日, 范围 1~31
20:16	TOY_HOUR	R		小时, 范围 0~23
15:10	TOY_MIN	R		分, 范围 0~59
9:4	TOY_SEC	R		秒, 范围 0~59
3:0	TOY_MILLISEC	R		0.1 秒, 范围 0~9

### 22.3.5 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		年, 范围~0 16383

### 22.3.6 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		年, 范围~0 16383
25:22	MONTH	RW		月, 范围 1~12
21:17	DAY	RW		日, 范围 1~31
16:12	HOUR	RW		小时, 范围 0~23
11:6	MIN	RW		分, 范围 0~59
5:0	SEC	RW		秒, 范围 0~59

### 22.3.7 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_rtcrewrite 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

### 22.3.8 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	RTC	W		

### 22.3.9 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	RTC	R		

### 22.3.10SYS\_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	RTC	RW		

# 23 加解密

## 23.1 DES

### 23.1.1 DES 功能概述

DES 控制器采用 32 位的 APB 接口，支持使用 DES/TDEA 算法进行加/解密，并支持使用 APB 接口进行 DMA 操作。DES 控制器采用了 OpenCore 的面积节约方案的 DES3 加/解密单元作为实现加/解密的运算单元。这个运算单元每 16 个时钟周期完成一次 DES 加/解密，每 48 个时钟周期完成一次 TDEA 加/解密。为了减少加/解密的等待时间，DES 控制器使用 2 个时钟域分别处理 APB 接口的操作和进行 DES 加/解密计算（DES 加解密使用的时钟的频率更高）。两个不同的时钟域通过 2 个 4 项的 64bit 位宽异步 FIFO 交换加/解密运算前后的数据。

DES 控制器在使用前需要先配置密钥以及 Command 寄存器。控制器有 3 个使用 64bit 格式存储的密钥：Key0、Key1、Key2。其中，Key1、Key2 仅在 TDEA 算法是需要进行配置。控制器不检查密钥中的校验位。待运算（加密还是解密由 Command[1]的值确定）的数据通过 Data\_low 和 Data\_high 写入运算数据 FIFO。DES 运算模块从运算数据 FIFO 读取数据后进行加/解密运算迭代，迭代的结果被送入运算结果 FIFO。通过 APB 接口查询 Command[4]可以获知运算结果是否就绪。当 Command[4]的值为 0 时，可以从 APB 接口通过 Data\_low 和 Data\_high 读取运算结果 FIFO 中的运算结果。

#### 23.1.2 DES 访问地址：

地址位	构成	备注
[27:16]	BAR_BASE	Device 2 的基地址寄存器值
[15:12]	8	固定为 8
[11:06]	0	保留
[05:00]	REG	内部寄存器地址

#### 23.1.3 DES 寄存器描述

DES 的寄存器列表及寄存器说明如下：

地址偏移	名称	说明
0x0	Key0_low	DES 密钥的低 32 位/TDEA 密钥 0 的低 32 位
0x4	Key0_high	DES 密钥的高 32 位/TDEA 密钥 0 的高 32 位
0x8	Key1_low	TDEA 密钥 1 的低 32 位
0xc	Key1_high	TDEA 密钥 1 的高 32 位
0x10	Key2_low	TDEA 密钥 2 的低 32 位
0x14	Key2_high	TDEA 密钥 2 的高 32 位
0x18	Data_low	非 DMA 模式（Command[2] = 1'b0）：待加/解密数据低 32 位的写入端口；加/解密后数据低 32 位的读出端口。

		DMA 模式 (Command[2] = 1'b1): Data_low 和 Data_high 不做区分。先写入/读取的是数据的低 32 位, 后写入/读取的是数据的高 32 位。
0x1c	Data_high	非 DMA 模式 (Command[2] = 1'b0): 待加/解密数据高 32 位的写入端口; 加/解密后数据高 32 位的读出端口。 DMA 模式 (Command[2] = 1'b1): Data_low 和 Data_high 不做区分。先写入/读取的是数据的低 32 位, 后写入/读取的是数据的高 32 位。
0x20	Command	命令和状态控制寄存器
0x24	Rev	保留
0x28		
0x2c		

Comand 寄存器位域说明:

位域	复位值	名称	属性	说明
0	0	des3	RW	0: 使用 DES 算法 1: 使用 TDEA 算法
1	0	decrypt	RW	0: 加密操作 1: 解密操作
2	0	dma_start	RW	写入 1 启动 DMA 操作, 写入 0 无影响 在 DMA 操作完成前此位保持为 1 当 DMA 操作完成时此位自动清零
3	0	dma_done	RW1C	当 DMA 操作完成时, 此位置 1 向此位写入 1, 则清零
4	1	dout_empty	RO	0: 数据读 FIFO 非空 1: 数据读 FIFO 为空
5	0	din_full	RO	0: 数据写 FIFO 未满 1: 数据写 FIFO 已满
7:6	0	Rev	RO	保留
31:8	0	dma_count	RW	需要进行 DMA 加/解密的 64 位数的个数

## 23.2 AES

### 23.2.1 AES 功能概述

AES 控制器采用 32 位的 APB 接口, 支持使用 128-bit KEY、192-bit KEY、256-bit KEY 进行 AES 算法加/解密, 并支持使用 APB 接口进行 DMA 操作。在使用 DMA 功能时, 支持 CTR 模式进行加/解密。AES 控制器有 3 个主要模块: KEY 扩展模块、加密模块、解密模块。加/解密运算模块每 11 个时钟周期完成一次 128-bit KEY 的 AES 加/解密, 每 13 个时钟周期完成一次 192-bit KEY 的 AES 加/解密, 每 15 个时钟周期完成一次 256-bit KEY 的 AES 加/解密。

解密。为了减少加/解密的等待时间，AES 控制器使用 2 个时钟域分别处理 APB 接口的操作和进行 AES 加/解密计算（AES 加解密使用的时钟的频率更高）。KEY 扩展模块工作在速度较慢的 APB 时钟域。两个不同的时钟域通过 2 个 4 项的 128bit 位宽异步 FIFO 交换加/解密运算前后的数据。

AES 控制器在使用前需要先配置密钥以及 Command 寄存器。AES 的密钥、明文和密文均以小尾端的格式进行存储。控制器使用 8 个 32bit 寄存器以小尾端的格式存储密钥：Key0、Key1、Key2、key3、key4、key5、key6、key7。其中，Key4、Key5 仅在 192-bit KEY 和 256-bit KEY 时需要进行配置；Key6、Key7 仅在 256-bit KEY 时需要进行配置。待运算（加密还是解密由 Command[1]的值确定）的数据通过 4 个 32 位寄存器地址（Data0、Data1、Data2、Data3）写入运算数据 FIFO。AES 运算模块从运算数据 FIFO 读取数据后进行加/解密运算迭代，迭代的结果被送入运算结果 FIFO。通过 APB 接口查询 Command[4]可以获知运算结果是否就绪。当 Command[4]的值为 0 时，可以从 APB 接口通过 Data0、Data1、Data2、Data3 读取运算结果 FIFO 中的运算结果。

### 23.2.2 AES 访问地址：

地址位	构成	备注
[27:16]	BAR_BASE	Device 2 的基地址寄存器值
[15:12]	9	固定为 9
[11:06]	0	保留
[05:00]	REG	内部寄存器地址

### 23.2.3 AES 寄存器描述

AES 的寄存器列表及寄存器说明如下：

地址偏移	名称	说明
0x0	Key0	AES 密钥的 Key[31:0]，以小尾端形式存储的 AES 密钥的最低 32 位
0x4	Key1	AES 密钥的 Key[63:32]
0x8	Key2	AES 密钥的 Key[95:64]
0xc	Key3	AES 密钥的 Key[127:96]，在使用 128-bit Key 时为以小尾端形式存储的 AES 密钥的最高 32 位
0x10	Key4	AES 密钥的 Key[159:128]，仅在使用 192/256-bit 的 Key 时使用
0x14	Key5	AES 密钥的 Key[191:160]，仅在使用 192/256-bit 的 Key 时使用，在使用 192-bit Key 时为以小尾端形式存储的 AES 密钥的最高 32 位
0x18	Key6	AES 密钥的 Key[223:192]，仅在使用 256-bit 的 Key 时使用
0x1c	Key7	AES 密钥的 Key[255:224]，仅在使用 256-bit 的 Key 时使用，在使用 256-bit Key 时为以小尾端形式存储的 AES 密钥的最高 32 位

0x20	Data0	非 DMA 模式(Command[2]=1'b0):待加/解密数据最低 32 位的写入端口和加/解密后数据最低 32 位的读出端口 DMA 模式(Command[2]=1'b1):Data0-Data3 不做区分, 数据按照从低到高的小尾端顺序写入或读出
0x24	Data1	非 DMA 模式(Command[2]=1'b0):待加/解密数据 63~32 位的写入端口和加/解密后数据高 63~32 位的读出端口 DMA 模式(Command[2]=1'b1):Data0-Data3 不做区分, 数据按照从低到高的小尾端顺序写入或读出
0x28	Data2	非 DMA 模式(Command[2]=1'b0):待加/解密数据 95~64 位的写入端口和加/解密后数据高 95~64 位的读出端口 DMA 模式(Command[2]=1'b1):Data0-Data3 不做区分, 数据按照从低到高的小尾端顺序写入或读出
0x2c	Data3	非 DMA 模式(Command[2]=1'b0):待加/解密数据的最高 32 位 (127~96) 的写入端口和加/解密后数据最高 32 位 (127~96) 的读出端口 DMA 模式(Command[2]=1'b1):Data0-Data3 不做区分, 数据按照从低到高的小尾端顺序写入或读出
0x30	Ctr_init_val	使用 CTR 模式时 CTR 计数器的初始值。要使此寄存器的值发生作用需要先配置此寄存器的值, 再向 command[0]写入 0。
0x34	Command	命令和状态控制寄存器
0x38	Rev	保留
0x3c		

#### Comand 寄存器位域说明:

位域	复位值	名称	属性	说明
0	0	Ctr_mode	RW	0: 使用普通的 AES 算法进行加/解密 1: 使用 CTR 模式进行 AES 算法的加/解密
1	0	decrypt	RW	0: 加密操作 1: 解密操作
2	0	dma_start	RW	写入 1 启动 DMA 操作, 写入 0 无影响 在 DMA 操作完成前此位保持为 1 当 DMA 操作完成时此位自动清零
3	0	dma_done	RW1C	当 DMA 操作完成时, 此位置 1, 中断输出信号变为高电平 向此位写入 1, 则清零, 中断输出信号变为低电平
4	1	dout_empty	RO	0: 数据读 FIFO 非空 1: 数据读 FIFO 为空

5	0	din_full	RO	0: 数据写 FIFO 未满 1: 数据写 FIFO 已满
7:6	0	Kr_mode	RW	0: 128-bit KEY 1: 192-bit KEY 2: 256-bit KEY 3: 保留
31:8	0	dma_count	RW	需要进行 DMA 加/解密的 128 位数的个数

## 23.3 RSA

### 23.3.1 RSA 访问地址:

地址位	构成	备注
[27:16]	BAR_BASE	Device 2 的基址寄存器值
[15:12]	0xA	固定为 0xA
[11]	0	保留
[10:00]	REG	内部寄存器地址

## 23.4 RNG

### 23.4.1 RNG 访问地址:

地址位	构成	备注
[27:16]	BAR_BASE	Device 2 的基址寄存器值
[15:12]	0xB	固定为 0xB
[11:00]	0	保留

直接通过以上地址访问 RNG 会返回一个 32 位随机数。

# 24 SDIO 控制器

## 24.1 功能概述

龙芯 2K1000 集成了一个 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

## 24.2 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：

地址位	构成	备注
[27:16]	BAR_BASE	Device 2 的基址寄存器值
[15:12]	0xC	固定为 0xC
[11:08]	0	保留
[07:00]	REG	内部寄存器地址

对于 SDIO 模块，使用时要注意将对应的引脚设置为相应功能。

与 SDIO 相关的引脚设置寄存器为 5.1 节中的 `sdio_sel`。

## 24.3 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。如下图是一个写多块数据的示意框图，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。

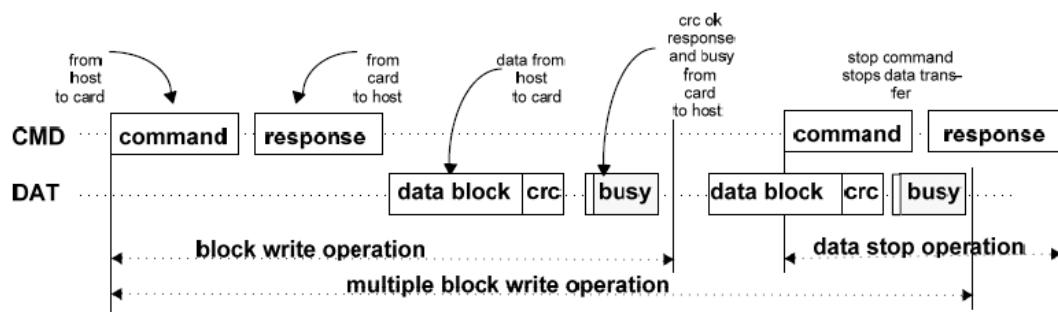


图 24-1 SD 卡多块写操作示意图

多块读操作的过程和多块写操作的过程类似（等补充说明）。

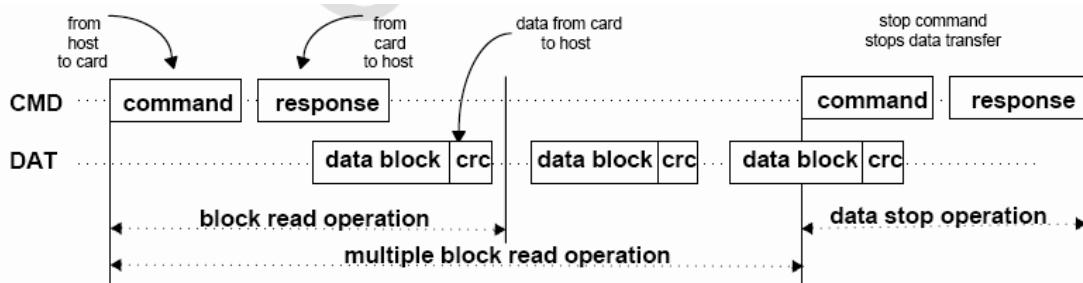


图 24-2 SD 卡多块读操作示意图

不同的命令都有统一的格式。一般命令的格式如下表，1bit 起始位，1bit 传输方向位，6bit 命令序号，32bit 命令参数，7bit CRC 检验位再加上 1bit 结束位。

Bit position	47	46	[45:40]	[39:8]	[7:1]	0
Width (bits)	1	1	6	32	7	1
Value	'0'	'1'	x	x	x	'1'
Description	start bit	transmission bit	command index	argument	CRC7	end bit

其中 command index 对应命令的序号，比如命令 0，cmdindex 为 0，命令 55，cmdindex 为 37。不同命令的参数可能不同，具体参考 SD 协议规范。

## 24.4 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 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

## 24.5 软件编程指南

### 24.5.1 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：

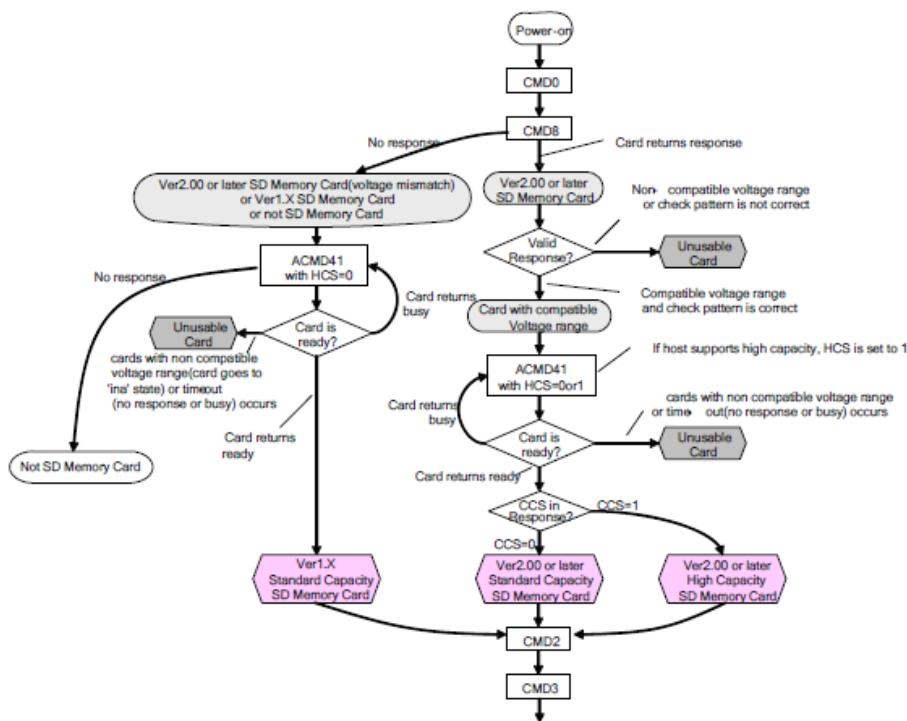


图 24-3 SD Memory 卡初始化流程示意图

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

#### 24.5.2 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。  
所以如果目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。

## 24.6 支持 SDIO 型号

本节列出经过验证的可支持的 SDIO 卡型号，其它类型的 SDIO 卡未经验证，不保证与本控制器兼容。

- Mem 卡：Kingston SD-CO2G SDC/2GB
- IO 卡 (wifi)：maxwell sd8686

# 25 I2S 控制器

## 25.1 概述

龙芯 2K1000 中 I2S 控制器，数据宽度是 32 位，支持 DMA 传输，支持多家公司的 codec 芯片。I2S 控制器仅支持主模式，由 I2S 产生位时钟信号和左右声道选择时钟信号。I2S 的功能特性包括：

1. 支持 8、16、20、24、32 位的音频数据采样位宽。
2. 支持 8、16、20、24、32 位的左右声道处理字宽。
3. 包含两个缓存 FIFO，FIFO 的缓存容量为 8bytes。
4. I2S 的中断处理模式可配，在 I2S 的发送和接收中断功能都使能后，当两个通道的缓存 fifo 为满仍要写以及为空仍要读时，则向 CPU 发出中断信号。
5. I2S 可以为 codec 芯片提供系统时钟，时钟频率可配。

## 25.2 访问地址及引脚复用

I2S 控制器内部寄存器的物理地址构成如下：

地址位	构成	备注
[27:16]	BAR_BASE	Device 2 的基地址寄存器值
[15:12]	0xD	固定为 D
[11:08]	0	保留
[07:00]	REG	内部寄存器地址

对于 I2S 模块，使用时要注意将对应的引脚设置为相应功能。

与 I2S 相关的引脚设置寄存器为 5.1 节中的 i2s\_sel。

## 25.3 接口协议

I2S 发送器和接收器的操作时序如图 25-1 所示。发送器和接收器的传输操作时序要求都是在信道选择信号（WS）改变之后的第二个位时钟开始传输下一帧数据，数据先传 MSB 位，再传输 LSB 位。发送器和接收器能处理的字位宽可以不一致，若发送器所需发送的数据的位宽比系统所支持的数据位宽要短，则 LSB 补零发送，反之，则忽略部分 LSB 数据；同理，对接受器而言，当接收到的数据的位宽比自己能处理的位宽小时，则 LSB 补零接收，反之，则忽略接收部分 LSB 数据。所以，对收发的数据而言，MSB 是具有固定的，而 LSB 则要取决于要收发的数据的字长已经系统所配置的字长位宽。

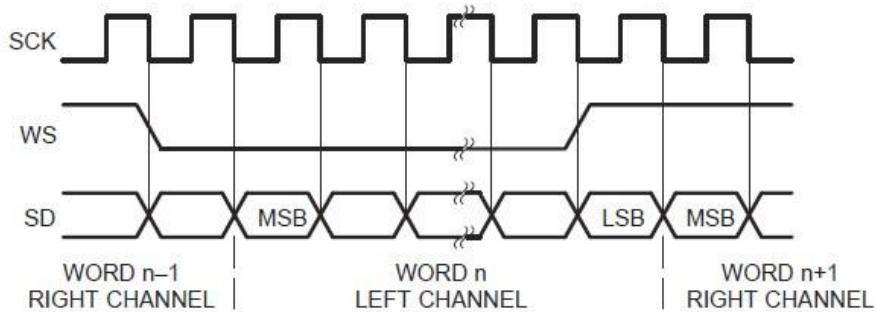


图 25-1 I2S 传输协议

## 25.4 专用寄存器

I2S 包含 5 个寄存器，定义如下表所示。

表 25-1 寄存器定义

寄存器名称	偏移地址	读/写(R/W)	功能描述	复位值
IISVersion	0xd000	R/W	I2S 标识寄存器	32'h0
IISConfig	0xd004	R/W	I2S 配置寄存器	32'h0
IISControl	0xd008	R/W	I2S 控制寄存器	32'h0
IISRxDATA	0xd00c	R/W	I2S 接收数据寄存器（用于 DMA 接收数据）	32'h0
IISTxDATA	0xd010	R/W	I2S 发送数据寄存器（用于 DMA 发送数据）	32'h0

接收标识寄存器允许主控机读取接收器的相关工作信息。它标识了 IIS 的地址位宽，数据位宽以及版本号等信息。

表 25-2 标识寄存器

寄存器名称	偏移地址	读/写(R/W)	功能描述
<b>IISVersion</b>	位	缺省值	描述
ADRW	9:8	2'h0	地址总线宽度： 00: 地址宽度 8 位 01: 地址宽度 16 位 10: 地址宽度 32 位 11: 地址宽度 64 位
DATW	5:4	2'h0	数据宽度： 00: 数据宽度 8 位 01: 数据宽度 16 位 10: 数据宽度 32 位 11: 数据宽度 64 位
VER	3:0	4'h0	I2S 版本号

接收配置寄存器是配置 I2S 的声道字长，音频数据的采样深度以及各个时钟的分频系数的。

表 25-3 配置寄存器

寄存器名称	偏移地址	读/写(R/W)	功能描述
<b>IISConfig</b>	位	缺省值	描述
LR_LEN	31:24	'h0	左右声道处理的字长。

寄存器名称	偏移地址	读/写(R/W)	功能描述
RES_DEPTH	23:16	'h0	采样深度设置： IIS 采样数据长度，有效范围为 8-32,如果发送或者接收到的数据宽度小于采样数据长度，则低位补 0；如果发送或者接收到的数据宽度大于采样数据长度，则低位忽略。
BCLK_RATIO	15:8	'h0	位时钟(BCLK)分频系数： 位时钟分频系数，分频数为总线时钟频率除以 $2x(RATIO+1)$
MCLK_RATIO	7:0	'h0	系统时钟(MCLK)分频系数， 系统时钟分频系数，分频数为总线时钟频率除以 $2x(RATIO+1)$ ,系统时钟作为 Codec 的 sysclk

控制寄存器用于配置 IIS 的工作使能信号，缓存 FIFO 的存储状态以及中断相关信息状态。

表 25-4 控制寄存器

寄存器名称	偏移地址	读/写(R/W)	功能描述
<b>I2SControl</b>	位	缺省值	描述
MASTER	15	'h0	1: IIS 工作于主模式
MSB/LSB	14	'h0	1: 高位在左端 0: 高位在右端
RX_EN	13	'h0	控制器接收使能，为 1 时有效，开始接收数据
TX_EN	12	'h0	控制器发送使能，为 1 时有效，开始发送数据
RX_DMA_EN	11	'h0	DMA 接收使能，为 1 时有效
Reserved	10: 8	'h0	
TX_DMA_EN	7	'h0	DMA 发送使能，为 1 时有效
Reserved	6:2	'h0	
RX_INT_EN	1	'h0	RX 中断使能，为 1 时使能中断，为 0 时禁止
TX_INT_EN	0	'h0	TX 中断使能，为 1 时使能中断，为 0 时禁止

## 25.5 配置操作

I2S 正常工作，需要先配置好 CODEC 芯片，然后配置 I2S 控制器的的配置寄存器和控制寄存器。

2K1000 芯片通过 I2S 接口和 CODEC 芯片通信，CODEC 芯片作为 I2S 总线上的从设备，详细地址、寄存器和配置方法请参考具体 CODEC 芯片的数据手册。配置完 CODEC 之后，需要对 I2S 控制器进行配置。可以将一些配置信息写入标志寄存器 (I2SVersion)，以供查询。先配置好 I2SConfig 寄存器，再配置 I2SControl 寄存器。

I2SConfig 寄存器建议 LR\_LEN 和 RES\_DEPTH 配成一样，不至于在传输过程中丢数据或者空数据。BCK 时钟根据配置 CODEC 的采样频率、采样深度和倍频系数来计算，计算公式如下：

$BCK=256xfs$ (或者  $512xfs$ )或者( $768xfs$ ); (详细见 CODEC 手册推荐配置)

$BCK\_RATIO=Freq\_APB/(256xfs)/2-1;$

双通道的话，计算公式下：

BCK = RES\_DEPTH x 2 xfs;

BCK\_RATIO= Freq\_APB / (RES\_DEPTH x 2 x fs) /2 – 1;

其中 fs 为配置的采样频率（即音频文件的码率）。发送和读取数据时，需要先配置好 DMA，再配置控制器，以免发生数据丢失。DMA 复用的配置方法见第 5.4 节。

MCK 时钟根据配置 CODEC 的采样频率和倍频系数来计算，计算公式如下：

MCK = 256xfs(或者 512xfs) 或者(768xfs); (详细见 CODEC 手册推荐配置)

MCK\_RATIO= Freq\_APB / (256 x fs) /2 – 1。

# 26 GMAC 控制器 (Dev 3)

龙芯 2K1000 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，分别为 Device 3 的不同功能（Function）。其配置空间基本信息如下：

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）	备注
GMAC0	0x0	0x3	0x0	0x7FFF	0xFE_0000_1800	
GMAC1	0x0	0x3	0x1	0x7FFF	0xFE_0000_1900	

## 26.1 访问地址及引脚复用

各个 GMAC 控制器内部寄存器的物理地址构成如下：

地址位	构成	备注
[63:15]	BAR_BASE	GMAC0 为 Device 3、FUNC 0 的基地址寄存器值 GMAC1 为 Device 3、FUNC 1 的基地址寄存器值
[14:00]	REG	内部寄存器地址

对于 GMAC1，使用时要注意将对应的引脚设置为相应功能。

与 GMAC1 相关的引脚设置寄存器为 5.1 节中的 gmac1\_sel。

# 27 OTG 控制器 (Dev 4, Fun 0)

## 27.1 概述

2K1000 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;

## 27.2 访问地址

2K1000 的 Device 4 包含 3 个功能 (Function), 分别为 OTG 控制器 (Function0)、EHCI 控制器 (Function1) 和 OHCI 控制器 (Function2), 其配置空间基本信息如下:

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)	备注
OTG	0x0	0x4	0x0	0x3_FFFF	0xFE_0000_2000	
EHCI	0x0	0x4	0x1	0x7FFF	0xFE_0000_2100	
OHCI	0x0	0x4	0x2	0x7FFF	0xFE_0000_2200	

OTG 控制器内部寄存器的物理地址构成如下:

地址位	构成	备注
[63:18]	BAR_BASE	Device 4、FUNC 0 的基址寄存器值
[17:00]	REG	内部寄存器地址

# 28 USB 控制器 (Dev 4, Fun 1/2)

## 28.1 总体概述

2K1000 的 USB 主机端口特性如下：

- 兼容 USB Rev 1.1、USB Rev 2.0 协议
- 兼容 OHCI Rev 1.0、EHCI Rev 1.0 协议
- 支持 LS (Low Speed)、FS (Full Speed) 和 HS (High Speed) 的 USB 设备
- 支持四个端口，每个端口都可挂 LS、FS 或 HS 设备

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：

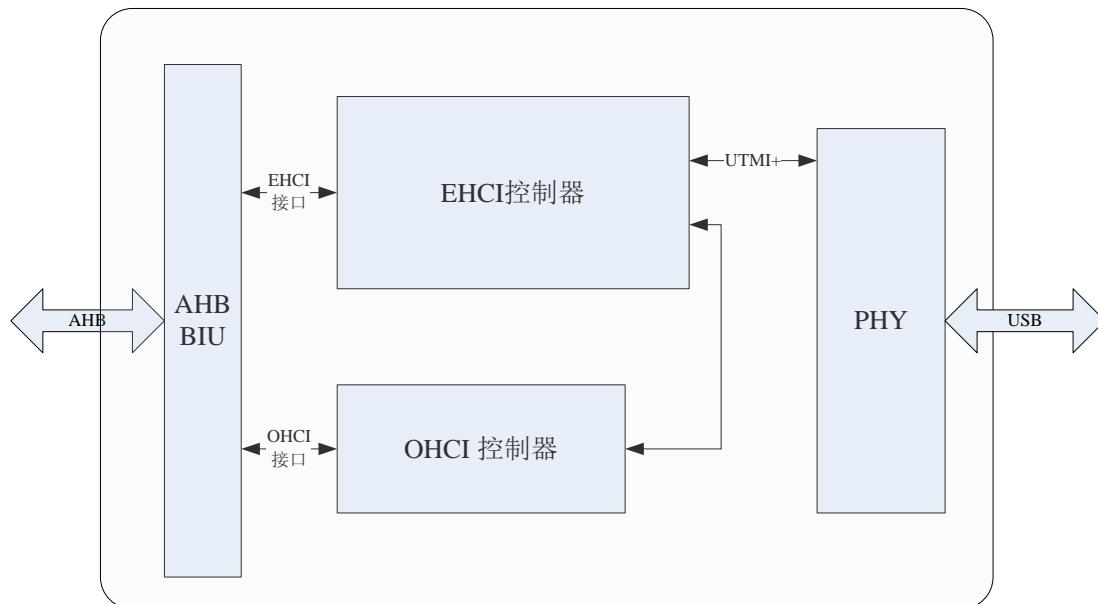


图 28-1 USB 主机控制器模块图

## 28.2 访问地址

2K1000 的 Device 4 包含 3 个功能 (Function) , 分别为 OTG 控制器 (Function0)、EHC1 控制器 (Function1) 和 OHCI 控制器 (Function2) , 其配置空间基本信息如下：

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)	备注
OTG	0x0	0x4	0x0	0x3_FFFF	0xFE_0000_2000	

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)	备注
EHCI	0x0	0x4	0x1	0x7FFF	0xFE_0000_2100	
OHCI	0x0	0x4	0x2	0x7FFF	0xFE_0000_2200	

EHCI 控制器内部寄存器的物理地址构成如下：

地址位	构成	备注
[63:15]	BAR_BASE	Device 4、FUNC 1 的基地址寄存器值
[14:00]	REG	内部寄存器地址

OHCI 控制器内部寄存器的物理地址构成如下：

地址位	构成	备注
[63:15]	BAR_BASE	Device 4、FUNC 2 的基地址寄存器值
[14:00]	REG	内部寄存器地址

# 29 图形处理器（Dev 5）

## 29.1 访问地址

2K1000 的 Device 5 为图形处理器 GPU，其配置空间基本信息如下：

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）	备注
GPU	0x0	0x5	0x0	0x3_FFFF	0xFE_0000_2800	

GPU 内部寄存器的物理地址构成如下：

地址位	构成	备注
[63:18]	BAR_BASE	Device 5、FUNC 0 的基址寄存器值
[17:00]	REG	内部寄存器地址

# 30 显示控制器 (Dev 6)

## 30.1 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 2K1000 的显示控制器支持的特性包括：

- 1) 双路 DVO 接口显示
- 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

## 30.2 访问地址及引脚复用

2K1000 的 Device 6 为显示控制器(DC),其配置空间基本信息如下:

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)	备注
DC	0x0	0x6	0x0	0xFFFF	0xFE_0000_3000	

DC 控制器内部寄存器的物理地址构成如下:

地址位	构成	备注
[63:16]	BAR_BASE	Device 6、FUNC 0 的基址寄存器值
[15:00]	REG	内部寄存器地址

对于 DC 控制器, 使用时要注意将对应的引脚设置为相应功能。

与 DC 相关的引脚设置寄存器为 5.3 节中的 dvo0\_sel、dvo1\_sel。

# 31 HDA 控制器 (Dev 7)

## 31.1 功能概述

HDA 控制器兼容 High Definition Audio Specification Revision 1.0a，主要的功能包括各种输入、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 控制器的整体设计框架包括了 5 个大的模块，分别为 SDI，SDO，axi\_master, axi\_slave, 和 reg config。其中 axi\_master 和 axi\_slave 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 个输出流。

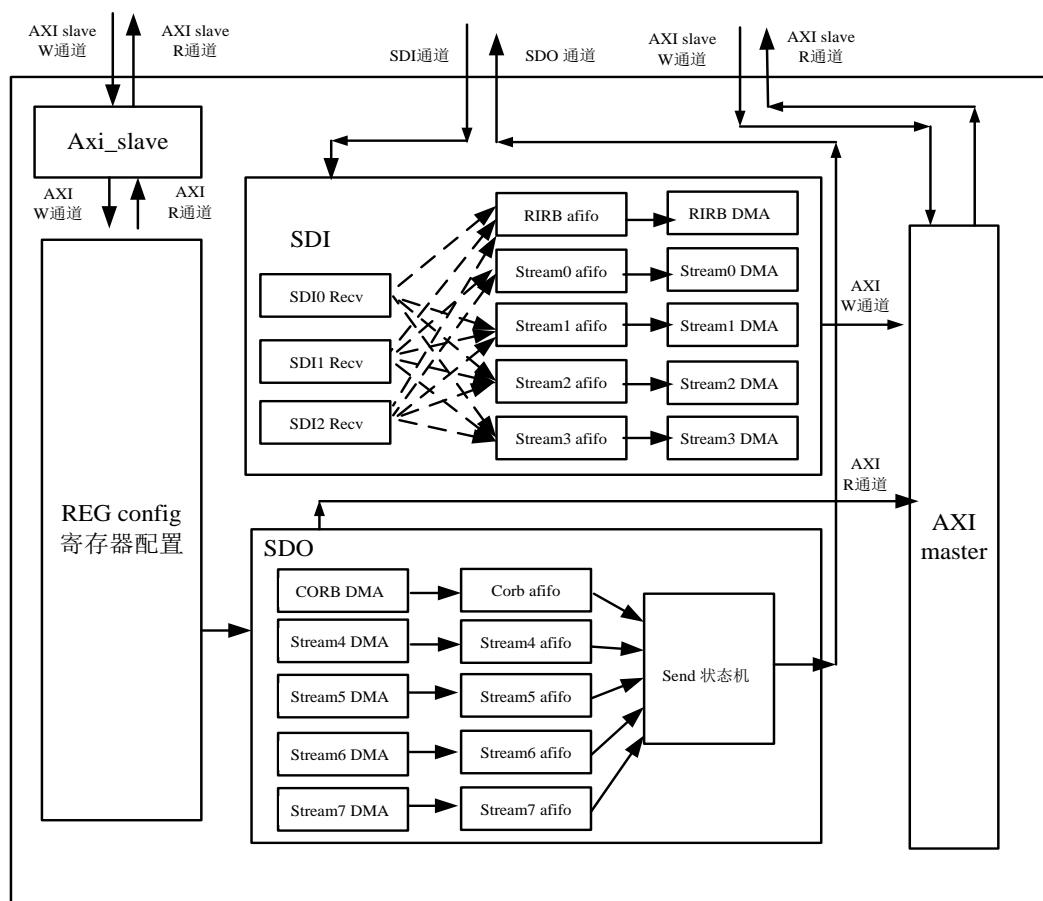


图 31-1 HDA 模块的整体设计框图

## 31.2 访问地址

2K1000 的 Device 7 为 HDA，其配置空间基本信息如下：

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)	备注
----	-----	-----	-----	--------	-------------------	----

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）	备注
HDA	0x0	0x7	0x0	0xFFFF	0xFE_0000_3800	

HDA 控制器内部寄存器的物理地址构成如下：

地址位	构成	备注
[63:16]	BAR_BASE	Device 7、FUNC 0 的基地址寄存器值
[15:00]	REG	内部寄存器地址

对于 HDA 控制器，使用时要注意将对应的引脚设置为相应功能。

与 HDA 相关的引脚设置寄存器为 5.1 节中的 hda\_sel。

# 32 SATA 控制器 (Dev 8)

## 32.1 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

## 32.2 访问地址

2K1000 的 Device8 为 SATA 控制器,其配置空间基本信息如下:

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)	备注
SATA	0x0	0x8	0x0	0xFFFF	0xFE_0000_4000	

SATA 控制器内部寄存器的物理地址构成如下:

地址位	构成	备注
[63:16]	BAR_BASE	Device 8、FUNC 0 的基地址寄存器值
[15:00]	REG	内部寄存器地址

## 32.3 SATA 控制器内部寄存器描述

SATA 的基址是由 SATA 的 BAR0 给定，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	BISTSER	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	命令列表地址低 32 位
0x104	32	P0_CLBU	命令列表地址高 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	命令列表地址低 32 位
0x184	32	P1_CLBU	命令列表地址高 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

# 33 PCIE 控制器(Dev 9/A/B/C/D/E)

## 33.1 总体结构

龙芯 2K1000 有两个 PCIE 控制器，其中一个 PCIE 控制器既可以作为一个 X4 的 PCIE 端口也可以作为 4 个独立的 X1 PCIE 端口；另一个 PCIE 控制器既可以作为一个 X4 的 PCIE 端口也可以作为 2 个独立的 X1 PCIE 端口，作为 X1 端口时，仅 LANE0 和 LANE1 可用，LANE2 和 LANE3 不可用。

龙芯 2K1000 的 PCIE 控制器仅可以作为 RC 使用，不能作为 EP。

龙芯 2K1000 的 PCIE 控制器结构如图 33-1 所示。其中一个 PCIE 控制器包含 0~3 号，共 4 个 PCIE 端口。0 号端口可以以 X4/X1 的方式工作，1~3 号端口仅能以 X1 的方式工作。另一个 PCIE 控制器仅包含 0 和 1 号 PCIE 端口。每个 PCIE 端口均有自己独立的 PCI 地址空间。

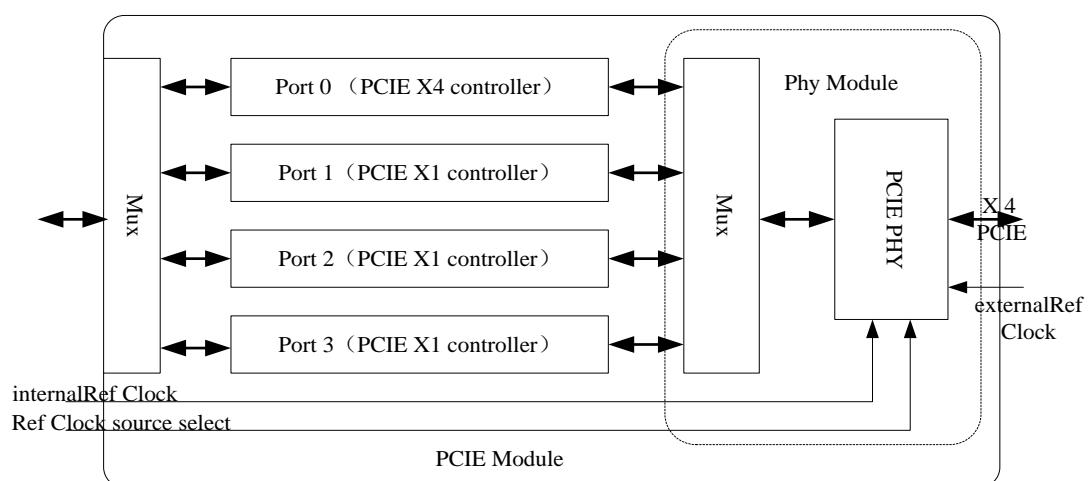


图 33-1PCIE 控制器结构

## 33.2 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)	备注
PCIE0 Port0	0x0	0x9	0x0/1	0xFFFF	0xFE_0000_4800	TYPE1 类型配 置头，仅 有 64 位 BAR0， 用于访 问内部 寄存器。 对功能 1 的访
PCIE0 Port1	0x0	0xA	0x0/1	0xFFFF	0xFE_0000_5000	
PCIE0 Port2	0x0	0xB	0x0/1	0xFFFF	0xFE_0000_5800	
PCIE0 Port3	0x0	0xC	0x0/1	0xFFFF	0xFE_0000_6000	

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）	备注
PCIE1 Port0	0x0	0xD	0x0/1	0xFFFF	0xFE_0000_6800	
PCIE1 Port1	0x0	0xE	0x0/1	0xFFFF	0xFE_0000_7000	同用于修改一些只读寄存器。

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]	BAR_BASE	PCIE 控制器内 BAR0 配置
[11:00]	REG	内部寄存器地址

### 3.3 地址空间划分

龙芯 2K1000 中的 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。

因为 2K1000 的 PCIE 控制器仅可以工作在 RC 模式下，所以其配置头为 TYPE1 类型。

每个 PCIE 端口作为 2K1000 中的独立设备，每个端口都包含一个 PCIE 配置头。当 PCIE 工作在 X4 模式时，其他 X1 的端口软件不可见，只有当 PCIE 工作在 X1 模式时才可以访问其他 X1 端口。

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。其中低 4KB 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 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，64 位地址空间，大小为 4KB，基地址等于 64 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。

## 33.4 软件编程指南

### 33.4.1 PCIE 控制器使能

芯片配置寄存器中通用配置寄存器 3（表 5-4）bit16 和 bit17 控制两个 PCIE 控制器的使能。在使用对应 PCIE 控制器时需要先将其使能才能访问该控制器以及下游设备的所有地址空间，包括对控制器的配置访问。

### 33.4.2 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：

Type 0	39	32 31	28 27	24 23	16 15	11 10	8 7	0
		FE0h		Offset[11:8]	Reserved	Device Number	Function Number	Offset[7:0]

PCIE 各个 Port 的设备号（device number）分别为 0x9, 0xa, 0xb, 0xc, 0xd, 0xe。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。

### 33.4.3 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。

### 33.4.4 TYPE1 类型配置访问

TYPE1 类型配置请求地址格式如下：

Type 1	39	32 31	28 27	24 23	16 15	11 10	8 7	0
		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。

### 33.4.5 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。

## 33.5 常用例程

本节给出龙芯 2K1000 的 PCIE 控制器的常用例程。龙芯 2K1000 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 2K1000 可以通过 cfg\_device\_read 和 cfg\_device\_write 这两个函数对下游设备的 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 longpcie_header_base = cfg0_base| (dev_num << 11)| (func_num <<8);
    unsigned longpcie_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)&0xfffffbffff;

    //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 longpcie_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, unsigned int io_limit, unsigned int mem_base, unsigned int mem_limit, unsigned int pref_mem_base, unsigned int pref_mem_limit, unsigned int pref_mem_base_upper32, unsigned int pref_mem_limit_upper32)
{//only used in RC mode
    unsigned longpcie_header_base = cfg0_base| (dev_num << 11)| (func_num <<8);
    unsigned longpcie_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 longpcie_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 longpcie_header_base = cfg1_base| (bus_num << 16) | (dev_num << 11)| (func_num  
<<8);  
    *(volatile unsigned int *) ( pcie_header_base + (reg_id<<2)) = write_data;  
}
```

# 34 DMA 控制器 (Dev F)

## 34.1 DMA 控制器结构描述

龙芯 2K1000 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：

- 传送前的预处理：由 CPU 配置 DMA 描述符相关的寄存器。
- 数据传送：在 DMA 控制器的控制下自动完成。
- 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，在 5.25 节有介绍，下表列出相同内容，方便查看。

表 34-1DMA ORDER 寄存器

位域	名称	访问	缺省值	描述
63:5	ask_addr	R/W	0	64 位地址的高 59 位
5	-	-	0	保留
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	axi_uncoherent	R/W	0	DMA 访问地址非一致性使能，设置为 1 代表 uncache 访问，设置为 0 代表 cache 访问。
0	dma_64bit	R/W	0	DMA 控制器 64 位地址支持

## 34.2 访问地址

2K1000 的 Device F 为 DMA 控制器，其配置空间基本信息如下：

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）	备注
DMA	0x0	0xF	0x0	0xFF	0xFE_0000_7800	

DMA 控制器内部寄存器的物理地址构成如下：

地址位	构成	备注
[63:08]	BAR_BASE	Device F、FUNC 0 的基址寄存器值
[07:00]	REG	内部寄存器地址

## 34.3 DMA 控制器与 APB 设备的交互

2K1000 中包含 5 个 DMA 控制器，使用 DMA 的 APB 设备包括 NAND、I2S、SDIO 以及加解密模块。其中 NAND 和 SDIO 各需要一个 DMA 控制器，而 I2S、加密模块、解密模块各需要两个 DMA 控制器分别控制 DMA 写操作和 DMA 读操作。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。

另外，2K1000 的 DMA 控制器支持 64 位地址空间，这主要通过 dma\_order[0] 来控制，当该位设置为 1 时表示 DMA 控制器工作在 64 位地址空间，反之为 32 位地址空间。在 64 位地址模式下，需要扩展 DMA\_ORDER\_ADDR 和 DMA\_SADDR 为 64 位寄存器。

## 34.4 DMA 描述符

### 34.4.1 DMA\_ORDER\_ADDR

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。

### 34.4.2 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。

#### 34.4.3 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	保留
30		1	R/W	保留
29:28		2	R/W	保留
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。

#### 34.4.4 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。

#### 34.4.5 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。

#### 34.4.6 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。

#### 34.4.7 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

#### 34.4.8 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)

#### 34.4.9 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)

# 35 VPU 控制器 (Dev 16)

## 35.1 访问地址

2K1000 的 Device 16 为 VPU 控制器, 其配置空间基本信息如下:

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)	备注
VPU	0x0	0x10	0x0	0x1_FF	0xFE_0000_8000	

VPU 内部寄存器的物理地址构成如下:

地址位	构成	备注
[63:09]	BAR_BASE	Device 16、FUNC 0 的基地址寄存器值
[08:00]	REG	内部寄存器地址

# 36 CAMERA 接口控制器 (Dev 17)

## 36.1 功能概述

Camera Interface 模块是视频输入转换存储模块。该模块一端接通用的 video camera 设备，另一端接 AHB 总线。实现了将 Camera 捕捉到的数据进行转换、并通过 DMA 存储到 memory 中。该 IP 支持 ITU-R BT 601/656 8-bit 模式，仅支持 YUV 格式输入。可以将 camera 产生的 YUV 三种颜色信号经打包成组后分别由 DMA 方式传输到 ESRAM 中的三片存储区。该设备支持图像的镜像和翻转，以便适应手持式设备在移动环境中图像的捕捉。可变的同步信号极性使得可以兼容各种摄像头外设。Camera Interface 兼容 AMBA 规范， AHB SLAVE 接口，用于读取软件配置数据和设置三种颜色数据在 ESRAM 中存放的基址和它们占用的存储空间， AHB MASTER 只有写接口，用于将打包后的数据写到存储区中。

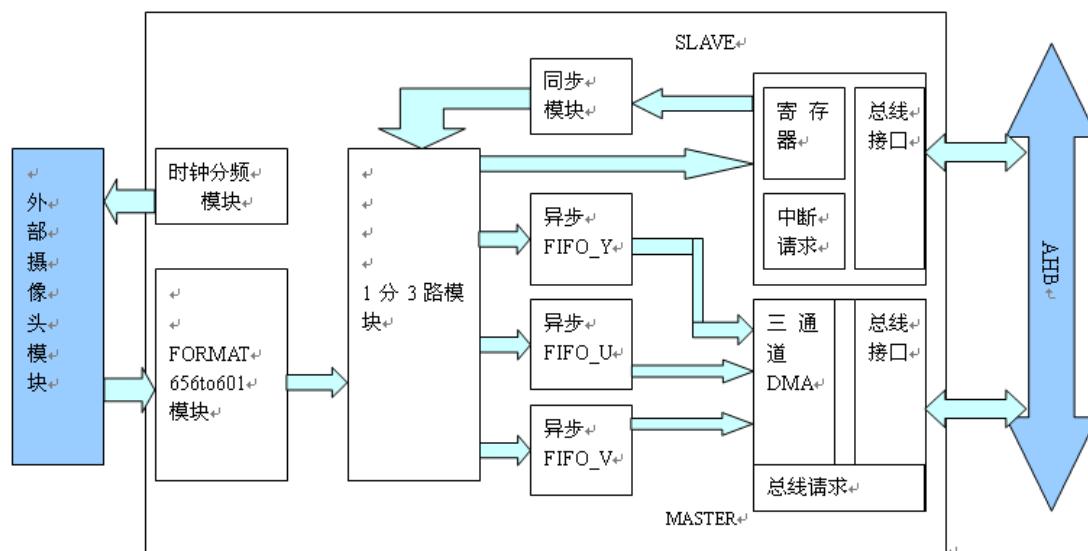


图 36-1 CAMERA 模块的整体设计框图

## 36.2 访问地址

2K1000 的 Device 17 为 CAMERA 接口控制器，其配置空间基本信息如下：

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）	备注
CAMERA	0x0	0x11	0x0	0xFF	0xFE_0000_8800	

CAMERA 控制器内部寄存器的物理地址构成如下：

地址位	构成	备注
[63:08]	BAR_BASE	Device 11、FUNC 0 的基地址寄存器值
[07:00]	REG	内部寄存器地址