LOONGSON

龙芯 1B 处理器用户手册

2011 年 5 月

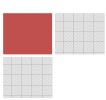
中国科学院计算技术研究所

龙芯中科技术有限公司

北京市海淀区科学院南路10号 100190
10 Kexueyuan South Road, Zhongguancun
Haidian District, Beijing

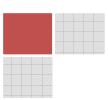
阅读指南

《龙芯 1B 处理器用户手册》主要介绍龙芯 1B 架构与寄存器描述,包括用户手册和软件编程指南两部分。软件编程指南介绍对 BIOS 和操作系统开发过程中的常见问题。



修订历史

日期	编者	版本	变化
2010-6-7	研发中心	1.0	1B 处理器初稿完成
2010-11-13	研发中心	1.1	增加了芯片引脚排布，DDR 控制器信息等
2010-11-15	研发中心	1.2	修改并进行标准排版
2010-11-15	研发中心	1.3	修正了第五章 DDR 的部分错误
2011-05-08	研发中心	1.4	修订了调试发现的错误
2011-05-17	研发中心	1.5	修订了多个小问题



目 录

图 目 录

表 目 录

1 概述

1B 芯片是基于 GS232 处理器核的单芯片系统，可广泛应用于工业控制、家庭网关、信息家电、医疗器械和安全应用等领域。1B 采用 SMIC0.13 微米工艺实现，采用 Wire Bond BGA256 封装。

1B 芯片具有以下关键特性：

- 集成一个 GS232 双发射龙芯处理器核，指令和数据 L1 Cache 各 8KB
- 集成一路 LCD 控制器，最大分辨率可支持到 1920*1080@60Hz/16bit
- 集成 2 个 10M/100M/1000M 自适应 MAC
- 集成 1 个 32 位 133MHz DDR2 控制器
- 集成 1 个 USB 2.0 接口，兼容 EHCI 和 OHCI
- 集成 1 个 8 位 NAND FLASH 控制器,最大支持 32GB
- 集成中断控制器，支持灵活的中断设置
- 集成 2 个 SPI 控制器，支持主从模式，支持系统启动
- 集成 AC97 控制器
- 集成 1 个全功能串口、1 个四线串口和 10 个两线串口
- 集成 3 路 I2C 控制器，兼容 SMBUS
- 集成 2 个 CAN 总线控制器
- 集成 62 个 GPIO 端口
- 集成 1 个 RTC 接口
- 集成 4 个 PWM 控制器
- 集成看门狗电路

1.1 体系结构框图

1B 芯片内部顶层结构由 AXI XBAR 交叉开关互连，其中 GS232、DC、AXI_MUX 作为主设备通过 3X3 交叉开关连接到系统； DC、AXI_MUX 和 DDR2 作为从设备通过 3X3 交叉开关连接到系统。在 AXI_MUX 内部实现了多个 AHB 和 APB 模块到顶层 AXI 交叉开关的连接，其中 DMA_MUX、GMAC0、GMAC1、USB 被 AXI_MUX 选择作为主设备访问交叉开关；AXI_MUX（包括 confreg、SPI0、SPI1）、 AXI2APB、

GMAC0、GMAC1、USB 等作为从设备被来自 AXI_MUX 的主设备访问。在 AXI2APB 内部实现了系统对内部 APB 接口设备的访问，这些设备包括 Watch Dog、RTC、PWM、I2C、CAN、NAND、UART 等。

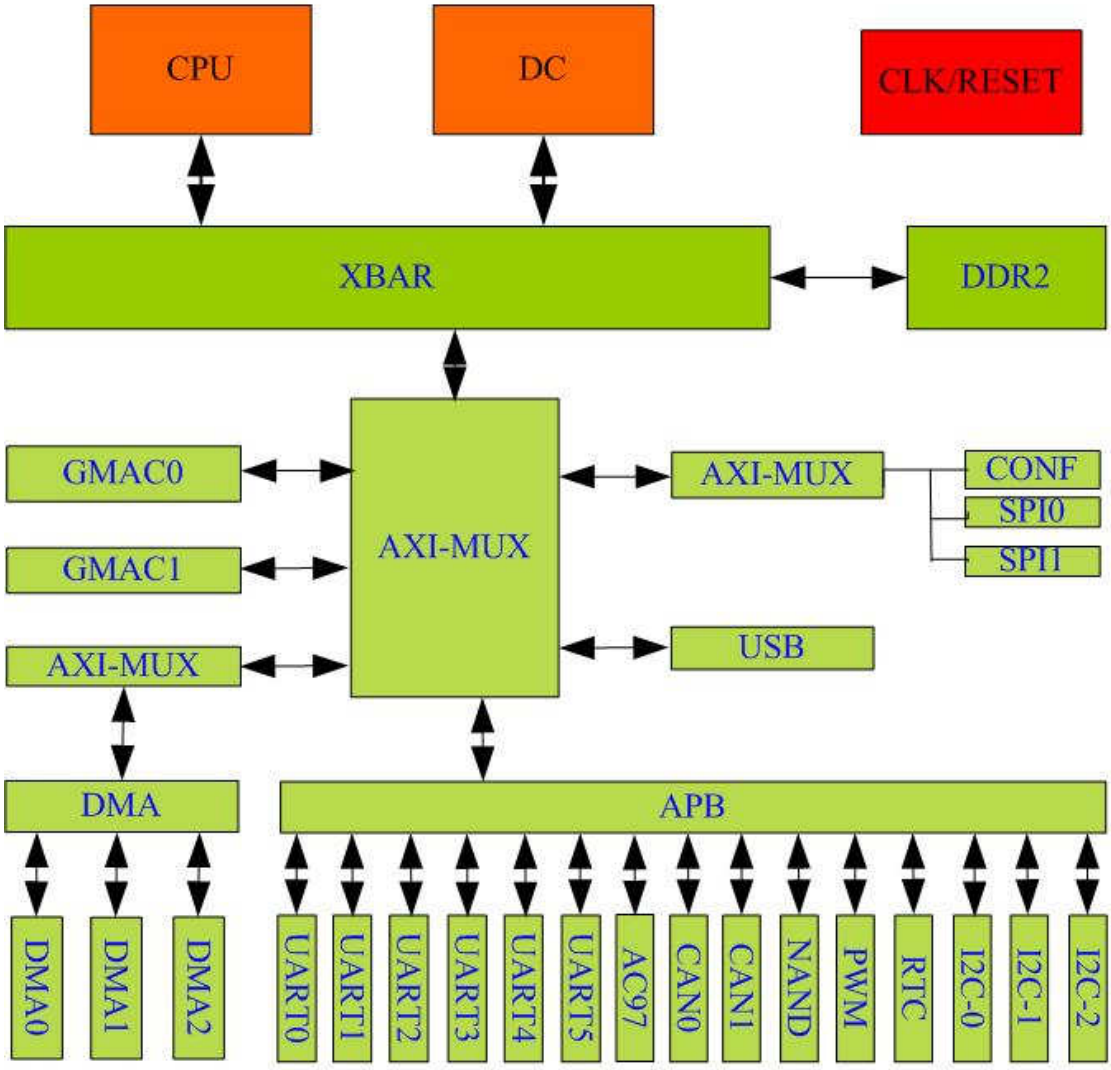


图 1-1 1B 芯片结构图

1.2 芯片主要功能

1B 芯片支持以下功能：

1.2.1 GS232 CPU

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

- 双发射五级流水、乱序发射、乱序执行

- 8KB 指令 Cache+8KB 数据 Cache，4 路组相连，指令 CACHE 支持路预测
- 6 项 BRQ、16 项的 QUEUE
- 动态转移预测、地址返回栈
- 32 项 JTLB，4 项 ITLB 、8 项 DTLB
- 两个定点 ALU 部件。
- 支持非阻塞的 Cache 访问技术，4 项 load 队列、2 项 store 队列、3 项 miss 队列，最多容忍 5 条 store 指令 Cache 不命中和 4 条 load 指令 Cache 不命中。
- 支持 cached store 指令的写合并和 uncache 写加速技术
- 支持 cache lock 技术和预取指令
- 支持流水线暂停模式
- 支持向量中断，可配置支持快速中断响应，最多 8 个时钟周期进入中断处理程序
- 支持 EJTAG 调试

1.2.2 DDR2

- 32 位 DDR2 控制器
- 遵守 DDR2 DDR 的行业标准（JESD79-2B）
- 一共含有 18 位的地址总线(即：15 位的行列地址总线和 3 位的逻辑 Bank 总线）。
- 接口上命令、读写数据全流水操作
- 内存命令合并、排序提高整体带宽
- 配置寄存器读写端口，可以修改内存设备的基本参数
- 内建动态延迟补偿电路（DCC），用于数据的可靠发送和接收
- 支持 33-133MHZ 工作频率

1.2.3 LCD Controller

- 屏幕大小可达 1920*1080
- 硬件光标

- 伽玛校正
- 最高像素时钟 172MHz
- 支持线性显示缓冲
- 上电序列控制
- 支持 16 位/24 位 LCD

1.2.4 USB2.0

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

1.2.5 AC97

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

1.2.6 GMAC

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

1.2.7 SPI

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

1.2.8 UART

- 集成 1 个全功能串口、1 个四线串口和 10 个两线串口
- 在寄存器与功能上兼容 NS16550A
- 全双工异步数据接收/发送
- 可编程的数据格式
- 16 位可编程时钟计数器
- 支持接收超时检测
- 带仲裁的多中断系统

1.2.9 I^2C

- 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

1.2.10 PWM

- 提供 4 路可配置 PWM 输出,
- 数据宽度 24 位
- 定时器功能
- 计数器功能

1.2.11 CAN

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

1.2.12 RTC

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

1.2.13 GPIO

- 62 位 GPIO
- 支持位操作

1.2.14 NAND

- 支持最大单颗 NAND FLASH 为 32GB
- 共 4 个片选 CS
- 数据宽度 8bit
- 支持 SLC

1.2.15 INT controller

- 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
- 支持固定中断优先级

1.2.16 Watchdog

- 16 比特计数器及初始化寄存器
- 低功耗模式暂停功能

1.2.17 功耗

- 0.5-0.9W

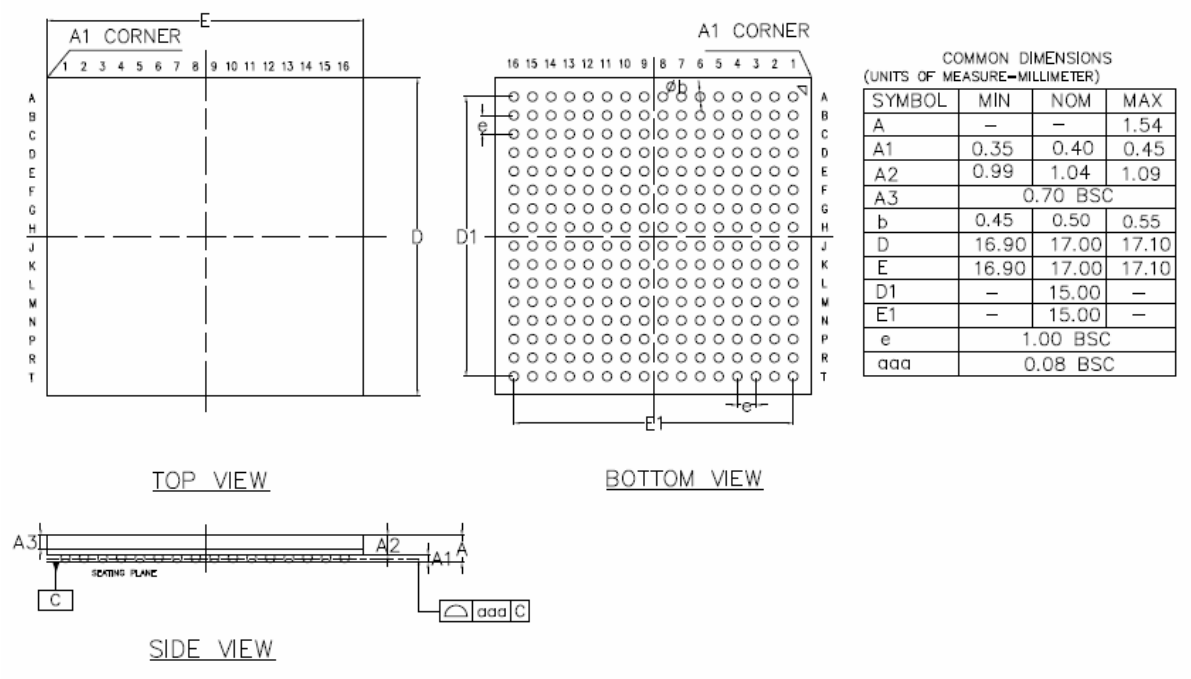
1.2.18 其它

- 测试访问口控制器 JTAG

2 芯片引脚定义

2.1 1B 引脚分布图

1B 采用 BGA256 封装形式，封装尺寸入下图所示：



COMMON DIMENSIONS
(UNITS OF MEASURE=MILLIMETER)

SYMBOL	MIN	NOM	MAX
A	–	–	1.54
A1	0.35	0.40	0.45
A2	0.99	1.04	1.09
A3		0.70 BSC	
b	0.45	0.50	0.55
D	16.90	17.00	17.10
E	16.90	17.00	17.10
D1	–	15.00	–
E1	–	15.00	–
e		1.00 BSC	
aaa		0.08 BSC	

芯片的引脚 PAD 的排布列如下表所示:

表 2- 1 芯片引脚 PAD 的排布列表

Pin Number	Net Name
A01	DDR2_A01
A02	DDR2_A05
A03	DDR2_A09
A04	DDR2_A13
A05	DDR2_CKN0
A06	DDR2_ODT0
A07	DDR2_BA2
A08	DDR2_SCSN0
A09	DDR2_DQ18
A10	DDR2_DQ23
A11	DDR2_DQS2
A12	DDR2_DQ17
A13	DDR2_DQM2
A14	RTC_CLK_I
A15	PLL_CPU_DVSS12

A16	PLL_CPU_DVDD12
B01	DDR2_A00
B02	DDR2_A04
B03	DDR2_A08
B04	DDR2_A12
B05	DDR2_CKP0
B06	DDR2_GATEO1
B07	DDR2_BA1
B08	DDR2_WEN
B09	DDR2_DQ21
B10	DDR2_DQ16
B11	DDR2_DQ19
B12	DDR2_DQ20
B13	DDR2_DQ22
B14	RTC_CLK_O
B15	PLL_CPU_AVSS33
B16	PLL_CPU_AVDD33
C01	DDR2_DQ06
C02	DDR2_A03
C03	DDR2_A07
C04	DDR2_A11
C05	DDR2_CKE0
C06	DDR2_GATEO0
C07	DDR2_BA0
C08	DDR2_CASN
C09	DDR2_DQ26
C10	DDR2_DQ31
C11	DDR2_DQM3
C12	DDR2_DQ25
C13	DDR2_DQ28
C14	RTC_VR_CEXT
C15	GMAC1_TX_CTL_O
C16	GMAC1_TX_CLK_O
D01	DDR2_DQM0
D02	DDR2_A02
D03	DDR2_A06
D04	DDR2_A10
D05	DDR2_A14
D06	DDR2_GATEI0
D07	DDR2_GATEI1
D08	DDR2_RASN
D09	DDR2_DQ29
D10	DDR2_DQ24

D11	DDR2_DQS3
D12	DDR2_DQ30
D13	DDR2_DQ27
D14	RTC_VR_VOUT
D15	UART1_CTS
D16	GMAC1_TX_CLK_I
E01	DDR2_DQ01
E02	DDR2_DQ09
E03	DDR2_DQ11
E04	DDR2_DQ12
E05	VREF_0V9
E06	VSS
E07	VDD1V8
E08	VSS
E09	VDD1V8
E10	VSS
E11	VDD1V8
E12	VREF_0V9
E13	RTC_VSS33
E14	RTC_VDD33
E15	UART1_RTS
E16	XTALI
F01	DDR2_DQ04
F02	DDR2_DQS1
F03	DDR2_DQM1
F04	DDR2_DQ14
F05	VSS
F06	VDD1V8
F07	VSS
F08	VDD1V8
F09	VSS
F10	VDD1V8
F11	VSS
F12	VDD1V8
F13	RTC_VDD33
F14	RTC_VSS33
F15	UART1_TX
F16	XTALO
G01	DDR2_DQS0
G02	DDR2_DQ03
G03	DDR2_DQ08
G04	DDR2_DQ15
G05	VDD1V2

G06	VSS
G07	VDD1V8
G08	VSS
G09	VDD3V3
G10	VSS
G11	UART5_RX
G12	UART5_TX
G13	I2C_SCL
G14	I2C_SDA
G15	UART1_RX
G16	SPI0_MISO
H01	DDR2_DQ00
H02	DDR2_DQ07
H03	DDR2_DQ13
H04	DDR2_DQ10
H05	VSS
H06	VDD1V2
H07	VSS
H08	VDD3V3
H09	VSS
H10	VDD3V3
H11	UART4_TX
H12	UART4_RX
H13	SPI0_CS3
H14	SPI0_CS0
H15	SPI0_MOSI
H16	SPI0_CLK
J01	DDR2_DQ05
J02	DDR2_DQ02
J03	VDD1V2
J04	VSS
J05	VDD1V2
J06	VSS
J07	VDD1V2
J08	VSS
J09	VDD3V3
J10	VSS
J11	UART3_RX
J12	UART3_TX
J13	SPI0_CS2
J14	SPI0_CS1
J15	AC97_DATA_O
J16	AC97_DATA_I

K01	VSS
K02	VDD1V2
K03	VSS
K04	VDD1V2
K05	VSS
K06	VDD1V2
K07	VSS
K08	VDD3V3
K09	VSS
K10	VDD3V3
K11	UART2_TX
K12	UART2_RX
K13	AC97_SYNC
K14	AC97_RESET
K15	AC97_BIT_CLK
K16	EJTAG_TDO
L01	VDD1V2
L02	VSS
L03	UART0_DCD
L04	UART0_RI
L05	USB_VDD1V2
L06	USB_AVSS33
L07	USB_AVDD33
L08	VSS
L09	VDD3V3
L10	VSS
L11	VDD3V3
L12	VSS
L13	EJTAG_TRST
L14	EJTAG_TMS
L15	EJTAG_TDI
L16	EJTAG_TCK
M01	UART0_DTR
M02	UART0_TX
M03	UART0_RX
M04	GMAC1_RX_CLK_I
M05	USB_VSS
M06	USB_AVDD33
M07	USB_AVSS33
M08	VDD3V3
M09	VSS
M10	VDD3V3
M11	VSS

M12	VDD3V3
M13	NAND_RD
M14	NAND_WR
M15	NAND_CE
M16	TEST_JTAG_SEL
N01	UART0_CTS
N02	UART0_RTS
N03	UART0_DSR
N04	LCD_DAT_B1
N05	LCD_DAT_B0
N06	USB0_DM
N07	USB0_DP
N08	TEST_CFG_MODEN
N09	GMAC0_RX_CTL_I
N10	GMAC0_RX1
N11	GMAC0_MDCK
N12	GMAC0_MDIO
N13	NAND_ALE
N14	NAND_CLE
N15	NAND_D7
N16	NAND_D6
P01	LCD_DAT_G0
P02	LCD_DAT_G1
P03	LCD_DAT_B2
P04	LCD_DAT_B3
P05	LCD_DAT_B4
P06	LCD_EN
P07	USB0_REXT
P08	SYS_RSTN
P09	GMAC0_RX0
P10	GMAC0_RX2
P11	GMAC0_RX3
P12	CAN1_RX
P13	CAN1_TX
P14	NAND_D3
P15	NAND_D4
P16	NAND_D5
R01	LCD_DAT_G2
R02	LCD_DAT_G4
R03	LCD_DAT_R0
R04	LCD_DAT_R2
R05	LCD_DAT_R4
R06	LCD_VSYNC

R07	USB0_XI
R08	GMAC0_RX_CLK_I
R09	GMAC0_TX2
R10	GMAC0_TX0
R11	GMAC0_TX_CLK_O
R12	CAN0_TX
R13	PWM0
R14	PWM3
R15	NAND_D2
R16	NAND_D1
T01	LCD_DAT_G3
T02	LCD_DAT_G5
T03	LCD_DAT_R1
T04	LCD_DAT_R3
T05	LCD_HSYNC
T06	LCD_CLK
T07	USB0_XO
T08	GMAC0_TX_CLK_I
T09	GMAC0_TX3
T10	GMAC0_TX1
T11	GMAC0_TX_CTL_O
T12	CAN0_RX
T13	PWM1
T14	PWM2
T15	NAND_RDY
T16	NAND_D0

2.2 系统相关引脚定义（6）

表 2-1 系统时钟引脚定义

No.	信号名称	方向	复用	上下拉	描述	电压域
1	XTALI	I			外部晶体时钟输入	core
2	XTALO	O			外部晶体时钟回送	core
3	RTC_CKI	I			RTC 时钟晶体输入	RTC
4	RTC_CKO	O			RTC 时钟晶体回送	RTC
5	TEST_CFG_MODEN	I			测试模块	core
6	SYS_RSTN	I			系统复位	core

2.3 LCD 引脚定义（20）

表 2-2 LCD 引脚定义

No.	信号名称	方向	复用	上下拉	描述	电压域
1	LCD_CLK	O			LCD 时钟	core
2	LCD_VSYNC	O			LCD 列同步	core
3	LCD_HSYNC	O			LCD 行同步	core
4	LCD_EN	O			LCD 可视使能信号	core
5	LCD_DAT_B0	O			LCD 蓝色数据信号 0	core
6	LCD_DAT_B1	O			LCD 蓝色数据信号 1	core
7	LCD_DAT_B2	O			LCD 蓝色数据信号 2	core
8	LCD_DAT_B3	O			LCD 蓝色数据信号 3	core
9	LCD_DAT_B4	O			LCD 蓝色数据信号 4	core
10	LCD_DAT_G0	O			LCD 绿色数据信号 0	core
11	LCD_DAT_G1	O			LCD 绿色数据信号 1	core
12	LCD_DAT_G2	O			LCD 绿色数据信号 2	core
13	LCD_DAT_G3	O			LCD 绿色数据信号 3	core
14	LCD_DAT_G4	O			LCD 绿色数据信号 4	core
15	LCD_DAT_G5	O			LCD 绿色数据信号 5	core
16	LCD_DAT_R0	O			LCD 红色数据信号 0	core
17	LCD_DAT_R1	O			LCD 红色数据信号 1	core
18	LCD_DAT_R2	O			LCD 红色数据信号 2	core
19	LCD_DAT_R3	O			LCD 红色数据信号 3	core
20	LCD_DAT_R4	O			LCD 红色数据信号 4	core

2.4 PLL 引脚定义（4）

表 2-3 PLL 引脚定义

No.	信号名称	方向	复用	上下拉	描述	电压域
1	PLL_DVDD12	I			1.2 伏数字电源	
2	PLL_DVSS12	I			1.2 伏数字地	
3	PLL_AVDD33	I			3.3 伏模拟电源	

4	PLL_AVSS33	I			3.3 伏模拟地	

2.5 VR 引脚定义（6）

表 2-4 VR 引脚定义

No.	信号名称	方向	复用	上下拉	描述	电压域
1	VR_VDDD-0	I			电源	VR_VDDD-0
2	VR_VDDA-0	I			电源	VR_VDDA-0
3	VR_VDDA-1	I			电源	VR_VDDA-1
4	VR_VDDD-1	I			电源	VR_VDDD-1
5	VR_VOUT	O			外接 10nf 电容	
6	VR_TOCAP	O			外接 4.7uf 电容	

2.6 DDR2 引脚定义（70）

表 2-5 DDR 引脚定义

No.	信号名称	方向	复用	上下拉	描述	电压域
1	DDR2_DQ00	B			外部存储数据总线第 0 位	
2	DDR2_DQ01	B			外部存储数据总线第 1 位	
3	DDR2_DQ02	B			外部存储数据总线第 2 位	
4	DDR2_DQ03	B			外部存储数据总线第 3 位	
5	DDR2_DQ04	B			外部存储数据总线第 4 位	
6	DDR2_DQ05	B			外部存储数据总线第 5 位	
7	DDR2_DQ06	B			外部存储数据总线第 6 位	
8	DDR2_DQ07	B			外部存储数据总线第 7 位	
9	DDR2_DQ08	B			外部存储数据总线第 8 位	
10	DDR2_DQ09	B			外部存储数据总线第 9 位	
11	DDR2_DQ10	B			外部存储数据总线第 10 位	
12	DDR2_DQ11	B			外部存储数据总线第 11 位	
13	DDR2_DQ12	B			外部存储数据总线第 12 位	
14	DDR2_DQ13	B			外部存储数据总线第 13 位	
15	DDR2_DQ14	B			外部存储数据总线第 14 位	
16	DDR2_DQ15	B			外部存储数据总线第 15 位	
17	DDR2_DQ16	B			外部存储数据总线第 16 位	
18	DDR2_DQ17	B			外部存储数据总线第 17 位	
19	DDR2_DQ18	B			外部存储数据总线第 18 位	
20	DDR2_DQ19	B			外部存储数据总线第 19 位	
21	DDR2_DQ20	B			外部存储数据总线第 20 位	
22	DDR2_DQ21	B			外部存储数据总线第 21 位	
23	DDR2_DQ22	B			外部存储数据总线第 22 位	
24	DDR2_DQ23	B			外部存储数据总线第 23 位	
25	DDR2_DQ24	B			外部存储数据总线第 24 位	
26	DDR2_DQ25	B			外部存储数据总线第 25 位	
27	DDR2_DQ26	B			外部存储数据总线第 26 位	
28	DDR2_DQ27	B			外部存储数据总线第 27 位	
29	DDR2_DQ28	B			外部存储数据总线第 28 位	
30	DDR2_DQ29	B			外部存储数据总线第 29 位	
31	DDR2_DQ30	B			外部存储数据总线第 30 位	

32	DDR2_DQ31	B			外部存储数据总线第 31 位	
33	DDR2_A00	O			外部存储地址总线第 0 位	
34	DDR2_A01	O			外部存储地址总线第 1 位	
35	DDR2_A02	O			外部存储地址总线第 2 位	
36	DDR2_A03	O			外部存储地址总线第 3 位	
37	DDR2_A04	O			外部存储地址总线第 4 位	
38	DDR2_A05	O			外部存储地址总线第 5 位	
39	DDR2_A06	O			外部存储地址总线第 6 位	
40	DDR2_A07	O			外部存储地址总线第 7 位	
41	DDR2_A08	O			外部存储地址总线第 8 位	
42	DDR2_A09	O			外部存储地址总线第 9 位	
43	DDR2_A10	O			外部存储地址总线第 10 位	
44	DDR2_A11	O			外部存储地址总线第 11 位	
45	DDR2_A12	O			外部存储地址总线第 12 位	
46	DDR2_A13	O			外部存储地址总线第 13 位	
47	DDR2_A14	O			外部存储地址总线第 14 位	
48	DDR2_DQS0	B			输入输出数据 strobe 信号	
49	DDR2_DQS1	B			输入输出数据 strobe 信号	
50	DDR2_DQS2	B			输入输出数据 strobe 信号	
51	DDR2_DQS3	B			输入输出数据 strobe 信号	
52	DDR2_DQM0	O			写数据屏蔽信号	
53	DDR2_DQM1	O			写数据屏蔽信号	
54	DDR2_DQM2	O			写数据屏蔽信号	
55	DDR2_DQM3	O			写数据屏蔽信号	
56	DDR2_CKp0	O			时钟信号	
57	DDR2_CKn0	O			时钟信号	
58	DDR2_CKE0	O			时钟有限信号	
59	DDR2_ODT0	O				
60	DDR2_SCSn0	O			片选信号	
61	DDR2_BA0	O			bank 选择信号	
62	DDR2_BA1	O			bank 选择信号	
63	DDR2_BA2	O			bank 选择信号	
64	DDR2_RASn	O			行选择	
65	DDR2_CASn	O			列选择	
66	DDR2_WEn	O			写信号	
67	DDR2_GATEI0	I				
68	DDR2_GATEI1	I				
69	DDR2_GATEO0	O				
70	DDR2_GATEO1	O				

2.7 USB 引脚定义(10)

表 2-6 USB 引脚定义

No.	信号名称	方向	复用	上下拉	描述	电压域
1	U_VDD33	I			3.3 伏模拟电源	
2	U_VSS33	I			模拟地	
3	U_REXT	I			外部调节电阻	
4	U_VSS33	I			模拟地	

5	U_VDD33	I			3.3 伏模拟电源	
6	U_DVDD	I			1.2 伏数字电源	
7	U_DVSS	I			数字地	
8	USB_XI	I			接地	
9	USB_XO	I			外部 12Mhz 参考时钟输入	
10	USB_DP	B			USB 差分数据	
11	USB_DM	B			USB 差分数据	

2.8 EJTAG 引脚定义(6)

表 2-7 JTAG 引脚定义

No.	信号名称	方向	复用	上下拉	描述	电压域
1	EJTAG_TCK	I		PU	TAP 时钟(内置上拉)	core
2	EJTAG_TRST	I		PU	TAP 复位(内置上拉)	core
3	EJTAG_TDI	I		PU	TAP 数据输入(内置上拉)	core
4	EJTAG_TDO	O			TAP 数据输出	core
5	EJTAG_TMS	I		PU	TAP 工作模式(内置上拉)	core
6	TEST_JTAG_SEL	I			JTAG/EJTAG 选择	core

2.9 GMAC0 引脚定义(15)

表 2-8 GMAC 引脚定义

No.	信号名称	方向	复用	上下拉	描述	电压域
1	GMAC0_TCKI	I			GMAC 传输时钟输入	core
2	GMAC0_TCKO	O			GMAC 传输时钟输出	core
3	GMAC0_TX0	O			GMAC 传输数据输出 0	core
4	GMAC0_TX1	O			GMAC 传输数据输出 1	core
5	GMAC0_TX2	O			GMAC 传输数据输出 2	core
6	GMAC0_TX3	O			GMAC 传输数据输出 3	core
7	GMAC0_TCTL	O			GMAC 传输控制	core
8	GMAC0_RCKI	I			GMAC 接收时钟输入	core
9	GMAC0_RX0	I			GMAC 接收数据输入 0	core
10	GMAC0_RX1	I			GMAC 接收数据输入 1	core
11	GMAC0_RX2	I			GMAC 接收数据输入 2	core
12	GMAC0_RX3	I			GMAC 接收数据输入 3	core
13	GMAC0_RCTL	I			GMAC 接受控制	core
14	GMAC0_MDC	O			读写 PHY 的时钟信号	core
15	GMAC0_MDIO	B			读写 PHY 的数据信号	core

2.10 GMAC1 引脚定义(4)

表 2-9 GMAC 引脚定义

No.	信号名称	方向	复用	上下拉	描述	电压域
1	GMAC1_TX_CLK_I	I			GMAC1 传输时钟输入	core
2	GMAC1_TX_CLK_O	O			GMAC1 传输时钟输出	core
3	GMAC1_RX_CLK_I	I			GMAC1 接收时钟输入	core
4	GMAC1_TX_CTL_O	O			GMAC1 传输控制	core

2.11 AC97 引脚定义(5)

表 2-10 AC97 引脚定义

No.	信号名称	方向	复用	上下拉	描述	电压域
1	AC97_BIT_CLK	I			AC97 时钟输入	core
2	AC97_DATA_I	I			AC97 数据输入	core
3	AC97_DATA_O	O			AC97 数据输出	core
4	AC97_SYNC	O			AC97 同步信号	core
5	AC97_RESET	O			AC97 复位信号	core

2.12 SPI 引脚定义(7)

表 2-11 SPI 引脚定义

No.	信号名称	方向	复用	上下拉	描述	电压域
1	SPI0_CLK	O			SPI0 时钟	core
2	SPI0_MISO	I			SPI0 主入从出数据	core
3	SPI0_MOSI	O			SPI0 主出从入数据	core
4	SPI0_CS0	O			SPI0 选通信号 0	core
5	SPI0_CS1	O			SPI0 选通信号 1	core
6	SPI0_CS2	O			SPI0 选通信号 2	core
7	SPI0_CS3	O			SPI0 选通信号 3	core

2.13 UART 引脚定义(20)

表 2-12 UART 引脚定义

No.	信号名称	方向	复用	上下拉	频率	描述	电压域
1	UART0_RX	I			1MHz	UART0 发送数据	core
2	UART0_TX	O			1MHz	UART0 接收数据	core
3	UART0_RTS	O			1MHz	UART0 请求发送	core
4	UART0_CTS	I			1MHz	UART0 允许发送	core
5	UART0_DSR	I			1MHz	UART0 设备准备好	core
6	UART0_DTR	O			1MHz	UART0 终端准备好	core
7	UART0_DCD	I			1MHz	UART0 载波检测	core
8	UART0_RI	I			1MHz	UART0 振铃提示	core
9	UART1_TX	O			1MHz	UART1 发送数据	core
10	UART1_RX	I			1MHz	UART1 接收数据	core
11	UART1_RTS	O			1MHz	UART1 请求发送	core
12	UART1_CTS	I			1MHz	UART1 允许发送	core
13	UART2_TX	O			1MHz	UART2 发送数据	core
14	UART2_RX	I			1MHz	UART2 接收数据	core
15	UART3_TX	O			1MHz	UART3 发送数据	core
16	UART3_RX	I			1MHz	UART3 接收数据	core

17	UART4_TX	O			1MHz	UART4 发送数据	core
18	UART4_RX	I			1MHz	UART4 接收数据	core
19	UART5_TX	O			1MHz	UART5 发送数据	core
20	UART5_RX	I			1MHz	UART5 接收数据	core

2.14 I2C 引脚定义(2)

表 2-13 I2C 引脚定义

No.	信号名称	方向	复用	上下拉	描述	电压域
1	I2C_SCL	O			第一路 I2C 时钟	core
2	I2C_SDA	B			第一路 I2C 数据	core

2.15 CAN 引脚定义(4)

表 2-14 CAN 引脚定义

No.	信号名称	方向	复用	上下拉	描述	电压域
1	CAN0_RX	I			CAN0 数据输入	core
2	CAN0_TX	O			CAN0 数据输出	core
3	CAN1_RX	I			CAN1 数据输入	core
4	CAN1_TX	O			CAN1 数据输出	core

2.16 NAND 引脚定义(14)

表 2-15 NAND 引脚定义

No.	信号名称	方向	复用	上下拉	描述	电压域
1	NAND_CLE	O			NAND 读信号	core
2	NAND_ALE	O			NAND 写信号	core
3	NAND_RD	O			NAND 命令锁存	core
4	NAND_WR	O			NAND 地址锁存	core
5	NAND_CE	O			NAND 片选信号	core
6	NAND_RDY	I			NAND 忙信号	core
7	NAND_D0	O			NAND 数据信号 0	core
8	NAND_D1	O			NAND 数据信号 1	core
9	NAND_D2	O			NAND 数据信号 2	core
10	NAND_D3	O			NAND 数据信号 3	core
11	NAND_D4	O			NAND 数据信号 4	core
12	NAND_D5	O			NAND 数据信号 5	core
13	NAND_D6	O			NAND 数据信号 6	core
14	NAND_D7	O			NAND 数据信号 7	core

2.17 PWM 引脚定义(4)

表 2-18 PWM 引脚定义

No.	信号名称	方向	复用	上下拉	描述	电压域

1	PWM0	O			PWM0 波形输出	core
2	PWM1	O			PWM1 波形输出	core
3	PWM2	O			PWM2 波形输出	core
4	PWM3	O			PWM3 波形输出	core

2.18 电源/地引脚(58)

表 2-19 电源地引脚

No.	信号名称	方向	描述	电压值	电压域	数目
1	VDD1V2		CORE 域	1.2v	CORE	10
2	VDD1V8		DDR2 电压域	1.8v	DDR2	8
3	VDD3V3		PAD 电压域	3.3v	PAD	11
4	VREF_0V9		DDR2 参考电源	0.9v	参考电源	2
4	VSS		接地	0v	接地	27

3 地址空间分配

本章给出 1B 芯片各模块的地址空间分配。

3.1 一级 AXI 交叉开关上模块的地址空间

表 3- 1 AXI 各模块地址分配

地址空间	模块	说明
0x0000,0000 – 0x0fff,ffff	DDR	256MB
0x1000,0000 – 0x1c19,ffff		**RESERVED**
0x1c20,0000 – 0x1c2f,ffff	DC Slave	1MB
0x1c30,0000 – 0x1eff,ffff		RESERVED
0x1f00,0000 – 0x1fff,ffff	AXI MUX Slave	16MB
0x2000,0000 – 0x7fff,ffff		RESERVED

3.2 AXI MUX 下各模块的地址空间

表 3-2 AXI MUX 各模块地址分配

地址空间	模块	说明
0xbf00,0000 – 0xbf7f,ffff	SPI0-memory	8MB
0xbf80,0000 – 0xbfbf,ffff	SPI1-memory	4MB
0xbfc0,0000 – 0xbfcf,ffff	SPI0	1MB
0xbfd0,0000 – 0xbfdf,ffff	CONFREG	1MB
0xbfe0,0000 – 0xbfe0,ffff	USB	64KB
0xbfe1,0000 – 0xbfe1,ffff	GMAC0	64KB
0xbfe2,0000 – 0xbfe2,ffff	GMAC1	64KB
0xbfe3,0000 – 0xbfe3,ffff		**RESERVED**
0xbfe4,0000 – 0xbfe7,ffff	APB-devices	256KB
0xbfe8,0000 – 0xbfeb,ffff	SPI0-IO	256KB
0xbfec,0000 – 0xbfef,ffff	SPI1-IO	256KB
0xbff0,0000 – 0xbfff,ffff		**RESERVED**

3.3 APB 各模块的地址空间分配

表 3-3 APB 各模块地址分配

地址空间	模块	说明
0xbfe40000-0xbfe43fff	UART0	16KB
0xbfe44000-0xbfe47fff	UART1	16KB
0xbfe48000-0xbfe4bfff	UART2	16KB

0xbfe4c000-0xbfe4ffff	UART3	16KB
0xbfe50000-0xbfe53fff	CAN0	16KB
0xbfe54000-0xbfe57fff	CAN1	16KB
0xbfe58000-0xbfe5bfff	I2C-0	16KB
0xbfe5c000-0xbfe5ffff	PWM	16KB
0xbfe60000-0xbfe63fff		**RESERVED**
0xbfe64000-0xbfe67fff	RTC	16KB
0xbfe68000-0xbfe6bfff	I2C-1	16KB
0xbfe6c000-0xbfe6ffff	UART4	16KB
0xbfe70000-0xbfe73fff	I2C-2	16KB
0xbfe74000-0xbfe77fff	AC97	16KB
0xbfe78000-0xbfe7bfff	NAND	16KB
0xbfe7c000-0xbfe7ffff	UART5	16KB

4 CPU

本章给出 1B 芯片内 CPU 的详细说明。GS232 核是一款实现 MIPS32 兼容且支持 DSP 扩展和 EJTAG 调试的双发射处理器，通过采用转移预测、寄存器重命名、乱序发射、路预测的指令 CACHE、非阻塞的数据 CACHE、写合并收集等技术来提高流水线的效率，形成了一款具有突出的性能价格比及性能功耗比的 32 位嵌入式处理器 IP。

GS232 处理器 IP 具有如下主要特点：

- MIPS32 兼容的 32 位处理器，包含可配置的 DSP 部件和 64 位浮点部件（FPU）。
- 高效的双发射+五级流水结构（取指、译码、发射、执行并写回、提交）
- 支持寄存器重命名、动态调度、转移预测等乱序发射、乱序执行技术。其中包含 6 项的转移队列、16 项重命名队列，采用 Gshare 转移猜测，BHT 为 256 项。
- 包含两个定点、一个浮点、一个访存四个功能部件。
- 定点部件支持 DSP 扩展指令的运算。
- 浮点部件支持全流水的 64 位浮点加法和浮点乘法运算，硬件实现浮点除法运算。
- 专门的 SIMD 型多媒体加速指令。
- 32 项全相联 JTLB，每项映射两页，页大小可变，并具有可执行位设置以防止缓冲区溢出攻击。
- 4 项的指令 TLB 和 8 项的数据 TLB，每项映射一页，页大小可变。
- 分离的一级指令 Cache 和数据 Cache，可配置为 1 路/2 路/4 路组相联，每路大小各为 4KB。
- 支持非阻塞的 Cache 访问技术，4 项 load 队列、2 项 store 队列、3 项 miss 队列，最多容忍 5 条 store 指令 Cache 不命中和 4 条 load 指令 Cache 不命中。
- 支持 cached store 指令的写合并和 uncache 写加速技术
- 支持 cache lock 技术和预取指令
- 软 IP 级可配置：是否包含 DSP 指令支持、浮点部件、指令和数据 Cache 的大小（4KB/8KB/16KB）、定点乘法规模（半规模/全规模/串行）等可配

置以满足不同应用的要求。

- 标准的 EJTAG 调试标准，方便软硬件调试。
- 标准的 32 位 AMBA AHB 接口。
- 上述特点使得 GS232 处理器 IP 具有执行效率高，功耗和成本低的优势，可在面向嵌入式的应用领域中得到广泛采用。

4.1 MIPS32 指令系统结构

MIPS 公司开发的 MIPS 体系结构（ISA）已经发展了 6 个版本，依次分别为 MIPS I，MIPS II，MIPS III，MIPS IV，MIPS V 和 MIPS 32/64，版本之间向前兼容。目前最新的是 MIPS 32/64 体系结构的 Release2 版本，其中，MIPS32 体系结构基于 MIPSII 体系结构的指令集，并补充了 MIPS III，IV 和 V 中的部分指令增强其生成代码和移动数据的效率；MIPS64 体系结构基于 MIPS V 体系结构的指令集，兼容 MIPS 32 体系结构。

MIPS32 是 MIPS 公司为了统一 MIPS 指令系统的不同版本，在 MIPS II 的基础上进行扩充，并对系统态的指令进行规范而定义的。MIPS32 指令系统的用户态指令融合了不同指令系统的优点，系统态指令也更加规范。

GS232 处理器兼容 MIPS32 的 Release2 体系结构。MIPS32 主要包括 ISA（Instruction Set Architecture）、PRA（Privileged Resource Architecture）、ASEs（Application Specific Extensions）和 UDI（User Defined Instructions）四个部分，GS232 处理器实现了 ISA、PRA，以及 DSP ASE。

本章从 CPU，DSP，FPU，PRA 四个方面描述 GS232 指令结构系统，并分别给出指令列表，以及 GS232 实现的 CP0 寄存器的定义。GS232 与 MIPS32 Release1 兼容，具体的指令说明以及寄存器描述可以参见 MIPS32 用户手册。

4.1.1 CPU 寄存器

MIPS32 体系结构定义了如下 CPU 寄存器：

（1）32 个 32 位的通用寄存器。GPRs（general purpose registers）。其中有两个被赋予了特殊含义：R0，0 号通用寄存器，值永远为 0；R31，31 号通用寄存器，被 JAL，BLTZAL，BLTZALL，BGEZAL，和 BGEZALL 指令隐式的用作目标寄存器，存放返回地址。

（2）一对用于保存乘法、除法和乘加操作的结果的特殊寄存器。HI 和 LO。HI 寄存器用于存放乘或者除运算结果的高位；LO 寄存器用于存放乘或者除运算

结果的低位。

（3）一个特殊的程序计数器（PC）。这个寄存器程序不可直接访问。

4.1.2 CPU 指令集

MIPS32 体系结构的定义把 CPU 指令集按照功能划分为以下几组：存取、计算、跳转分支、协处理器和杂项指令。①Load 和 Store 访存指令在主存和通用寄存器之间移动数据。访存指令都是立即数指令（I 型），因为该指令模式所支持的唯一访存模式就是基址寄存器加上 16 位的对齐的偏移量。②Computational 计算型指令完成寄存器值的算术、逻辑、移位、乘法和除法操作。计算型指令包含了寄存器指令格式（R 型，操作数和运算结果均保存在寄存器中）和立即数指令格式（I 型，其中一个操作数为一个 16 位的立即数）。③Jump and Branch 跳转和分支指令改变程序的控制流。绝对地址跳转被称为“Jump(跳转)”(J 型或者 R 型),PC(指令计数器)相关的跳转指令被称为“Branch（分支)”（I 型)。④Coprocessor 协处理器指令完成协处理器内部的操作。协处理器的访存操作是 I 型指令。在 MIPS32 定义了 0 号协处理器（系统处理器）和 1 号协处理器（浮点协处理器），用户可以自定义 2 号协处理器的功能。⑤Special 特殊指令完成系统调用和断点操作。这些指令通常是 R 型的。⑥Exception 异常指令引起跳转，根据异常号比较结果跳转到通用异常处理向量。这些指令包括 R 型和 I 型指令格式。

GS232 IP 虽然与 MIPS32 Release2 版本兼容，从功能上实现了 MIPS32 体系结构规定的所有 CPU 指令，但是有些指令在实现了有细微的并不影响兼容性的差别，以下值得编程人员注意。

pref、prefx 指令。

表 4-1 到表 4-10 列出了 GS232IP 实现的 MIPS32 指令中的 CPU 指令。MIPS32 的浮点和 CP0 协处理器指令在后面章节介绍。

表 4-1 CPU 指令集：访存指令

OpCode	Description	MIPS ISA
LB	取字节	I
LBU	取无符号字节	I
LH	取半字	I
LHU	取无符号半字	I
LW	取字	I
LWU	取无符号字	I
LWL	取字左部	I

OpCode	Description	MIPS ISA
LWR	取字右部	I
LL	取标志处地址	I
SB	存字节	I
SH	存半字	I
SW	存字	I
SWL	存字左部	I
SWR	存字右部	I
SC	满足条件下存	I
SYNC	同步	I
PREF	预取	MIPS32

表 4-2 CPU 指令集：算术指令 (ALU 立即数)

OpCode	Description	MIPS ISA
ADDI	加立即数	I
ADDIU	加无符号立即数	I
SLTI	小于立即数设置	I
SLTIU	无符号小于立即数设置	I
ANDI	与立即数	I
ORI	或立即数	I
XORI	异或立即数	I
LUI	取立即数到高位	I

表 4-3 CPU 指令集：算术指令 (2 操作数)

OpCode	Description	MIPS ISA
CLO	计算前导 0 的个数	MIPS32
CLZ	计算前导 1 的个数	MIPS32

表 4-4 CPU 指令集：算术指令(3 操作数, R-型)

OpCode	Description	MIPS ISA
ADD	加	I
ADDU	无符号加	I
SUB	减	I
SUBU	无符号减	I
SLT	小于设置	I
SLTU	无符号小于设置	I
AND	与	I
OR	或	I

XOR	异或	I
NOR	或非	I

表 4-5 CPU 指令集：乘法和除法指令

OpCode	Description	MIPS ISA
MADD	乘加	MIPS32
MADDU	无符号乘加	MIPS32
MSUB	乘减	MIPS32
MSUBU	无符号乘减	MIPS32
MUL	乘（结果放到 GPR）	MIPS32
MULT	乘（结果放到 HI,LO）	I
MULTU	无符号乘	I
DIV	除	I
DIVU	无符号除	I
MFHI	从 hi 寄存器取数到通用寄存器	I
MTHI	从通用寄存器存数到 hi 寄存器	I
MFLO	从 lo 寄存器取数到通用寄存器	I
MTLO	从通用寄存器存数到 lo 寄存器	I

表 4-6 CPU 指令集：跳转和分支指令

Opcode	Description	MIPS ISA
J	跳转	I
JAL	立即数调用子程序	I
JR	跳转到寄存器指向的指令	I
JALR	寄存器调用子程序	I
BEQ	相等则跳转	I
BNE	不等则跳转	I
BLEZ	小于等于 0 跳转	I
BGTZ	大于 0 跳转	I
BLTZ	小于 0 跳转	I
BGEZ	大于或等于 0 跳转	I
BLTZAL	小于 0 调用子程序	I
BGEZAL	大于或等于 0 调用子程序	I
BEQL	相等则 Likely 跳转	II
BNEL	不等则 Likely 跳转	II
BLEZL	小于或等于 0 则 Likely 跳转	II
BGTZL	大于 0 则 Likely 跳转	II

Opcode	Description	MIPS ISA
BLTZL	小于 0 则 Likely 跳转	II
BGEZL	大于或等于 0 则 Likely 跳转	II
BLTZALL	小于 0 则 Likely 调用子程序	II
BGEZALL	大于或等于 0 则 Likely 调用子程序	II

表 4-7 CPU 指令集：移位指令

OpCode	Description	MIPS ISA
SLL	逻辑左移	I
SRL	逻辑右移	I
SRA	算术右移	I
SLLV	可变的逻辑左移	I
SRLV	可变的逻辑右移	I
SRAV	可变的算术右移	I

表 4-8 CPU 指令集：特殊指令

OpCode	Description	MIPS ISA
SYSCALL	系统调用	I
BREAK	断点	I
MOVZ	等于 0 时移动	MIPS32
MOVN	不等于 0 时移动	MIPS32
MOVT	浮点真时移动	MIPS32
MOVF	浮点假时移动	MIPS32

表 4-9 CPU 指令集：异常指令

OpCode	Description	MIPS ISA
TGE	大于或等于陷入	II
TGEU	无符号数大于或等于陷入	II
TLT	小于陷入	II
TLTU	无符号数小于陷入	II
TEQ	等于陷入	II
TNE	不等陷入	II
TGEI	大于或等于立即数陷入	II
TGEIU	大于或等于无符号立即数陷入	II
TLTI	小于立即数陷入	II
TLTIU	小于无符号立即数陷入	II
TEQI	等于立即数陷入	II
TNEI	不等于立即数陷入	II

表 4-10 CPU 指令集：CP0 指令

OpCode	Description	MIPS ISA
MFC0	从 CP0 寄存器取	I
MTC0	往 CP0 寄存器写	I
TLBR	读索引的 TLB 项	III
TLBWI	写索引的 TLB 项	III
TLBWR	写随机的 TLB 项	III
TLBP	在 TLB 中搜索匹配项	III
CACHE	Cache 操作	III
ERET	异常返回	III

4.1.3 CP0 指令集

下面给出 GS232 定义的 CP0 指令：

表 4-11 GS232 的 CP0 指令

OpCode	Description	MIPS ISA
MFC0	从 CP0 寄存器取	MIPS32
MTC0	写 CP0 寄存器	MIPS32
ERET	异常返回	MIPS32
DERET	Debug 返回	EJTAG
SDBBP	软件中断	EJTAG
CACHE 指令		
OpCode	Description	Target Cache
CACHE0	Index Invalidate	Instruction Cache
CACHE8	Index Store Tag	Instruction Cache
CACHE16	Hit Invalidate	Instruction Cache
CACHE28	Hit lock	Instruction Cache

CACHE1	Index WriteBack Invalidate	Data Cache
CACHE5	Index Load Tag	Data Cache
CACHE9	Index Store Tag	Data Cache
CACHE17	Hit Invalidate	Data Cache
CACHE21	Hit WriteBack Invalidate	Data Cache
Cache29	Hit lock	Data Cache

MIPS32 只对 CACHE 指令进行了必要的规范，其中有很多 CACHE 指令是推荐或可选的，GS232IP 只实现了 CACHE 指令中 MIPS32 规定必须要实现的部分。具体可参见上表。

4.1.4 存储空间

GS232IP 虚拟地址和物理地址都是 32 位，即 SEGBITS 和 PABITS 均为 32 。GS232IP 的实现可用地址空间分布如下表所示：

表 4-12 GS232IP 地址空间的分配

段名	虚拟地址范围	从不同模式下访问		
		用户模式	高级用户模式	核心模式
kseg3	0xFFFF FFFF-0xE000 0000	address error	address error	mapped
sseg Ksseg	0xDFFF FFFF-0xC000 0000	address error	address error	mapped
kseg1	0xBFFF FFFF-0xA000 0000	address error	address error	unmapped uncached
kseg0	0x9FFF FFFF-0x8000 0000	address	address error	unmapped
useg suseg kuseg	0x7FFF FFFF-0x0000 0000	mapped	mapped	ERL=1 unmapped ERL=0 mapped

注： 1，表中不包括 EJTAG 的 dseg 段 。

2，GS232IP 支持 4KB, 16KB, 64KB, 256KB, 1MB, 4MB, 16MB 大小的 TLB 页面

根据 MIPS32 的规范，GS232 的虚拟地址转换包括以下几种情况。①凡是

mapped 的段，都通过访问 TLB 获得物理地址以及相应页的状态。②kseg0 段的物理地址为虚地址减去 0x80000000 得到，该段是否 cached 根据 Config 寄存器的 K0 字段决定。③kseg1 的物理地址为虚地址减去 0xA0000000 得到，该段为 uncached 段。④当 ERL 为 1 时，最低 2G 空间 useg 的物理地址低 31 位直接从虚拟地址得到，并成为 uncached 访问方式。否则的话，useg 为 mapped,cached 的访问模式。

访问 TLB 的虚地址由两部分组成，一是 EntryHi 寄存器的 ASID 域，二是由地址运算部件给出的虚地址。该虚地址与 TLB 的每一项同时进行比较。如果有一个相符且有效，则从该项中抽取物理页号与偏移量（虚地址的低位）组成物理地址。如果没有相符或虽然相符但无效，则发生相应例外。

TLB 是一个全相联存储器，包括 32 项，提供 32 对奇/偶页面映像。页的大小为 4KB-16MB(以 4 的倍数递增)。TLB 的每一项共有 256 位。其主要内容如下表所示，其中 NE 为表示不可执行位，是 GS232 中为了防止缓冲区溢出攻击对每页增设的.NE 位为 1 时表示该页不可执行。

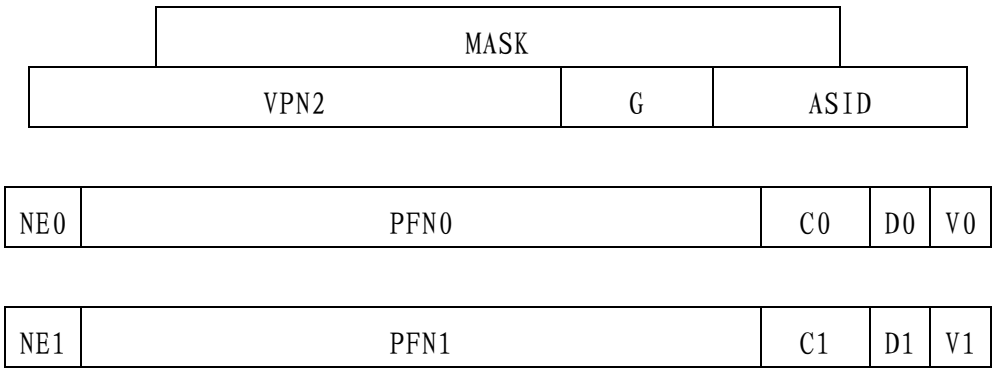


图 4-1 TLB 表项内容

由于 MIPS 的 TLB 每一项包括一个奇数页和一个偶数页，因此进行虚地址匹配时使用 VPN2，即虚地址的页号除以 2。MIPS 的页大小是可以变化的，由 TLB 中的 MASK 域决定。地址匹配时要考虑这个因素。虚地址匹配时，R 域也应该匹配，以区分不同的段。此外，ASID 域也应该匹配，除非 G 域为 1。如果进行上述匹配后发现 TLB 中没有与虚地址匹配的 TLB 项，则产生 TLBLR(取数)或 TLBSR（存数）例外。如果匹配成功但相应的 TLB 有效位 V 为 0，则产生 TLBLI（取数）或 TLBSI（存数）例外。对于存数操作，如果匹配成功且有效位为 1，但是 D 为 0，则产生 MOD 例外。

4.1.5 例外处理

龙芯 2 号 IP 的例外处理遵循 MIPS32 规范。下表给出了 GS232IP 实现的例外以及

例外时的寄存器修改。

表 4-13 例外编码及寄存器修改

	status_EXL	status_ERL	status_BEV	status_NMI	status_SR	cause_exco	debug_DM	bebug_5:0	EPC	ErrorEPC	DEPC	cacheErr	FCR_cause	Entry_hi	Context	Badvaddr
reset		1	1	0	0					*						
soft reset		1	0	0	1					*						
NMI		1	0	1	0					*						
cache error		1				30				*		*				
INT		1				0			*					*	*	*
MOD		1				1			*					*	*	*
TLBL		1				2			*					*	*	*
TLBS		1				3			*							*
ADEL		1				4			*							*
ADES		1				5			*							
IBE		1				6			*							
DBE		1				7			*							
SYS		1				8			*							
BP		1				9			*							
RI		1				10			*							
CPU		1				11			*							
OV		1				12			*							
TRAP		1				13			*							
FPE		1				15			*				*			
DSS							1	1		*						
DBP							1	2		*						
DDBL							1	4		*						

DDBS							1	8		*						
DIB							1	16		*						
DINT							1	32		*						

下表给出了 GS232IP 的不同例外的例外入口地址。如表中所示龙芯 1 号 IP 支持向量中断。

表 4-14 例外入口地址

Exception	Statuts_BEV	Status_EXL	Cause_IV	EJTAG ProbEn	Vector
Reset, Soft reset, NMI	X	X	X	X	0xBFC0 0000
EJTAG Debug	X	X	X	0	0xBFC0 0480
EJTAG Debug	X	X	X	1	0xFF20 0200
TLB Refill	0	0	X	X	0x8000 0000
TLB Refill	0	1	X	X	0x8000 0180
TLB Refill	1	0	X	X	0xBFC0 0200
TLB Refill	1	1	X	X	0xBFC0 0380
Cache error	0	X	X	X	0xA000 0100
Cache error	1	X	X	X	0BFC0 0300
Interrupt	0	0	0	X	0x8000 0180
Interrupt	0	0	1	X	0x8000 0200
Interrupt	1	0	0	X	0xBFC0 0380
Interrupt	1	0	1	X	0xBFC0 0400
All others	0	X	X	X	0x8000 0180
All others	1	X	X	X	0xBFC0 0380
‘x’ denotes don’t care					

4.1.6 CP0 寄存器

MIPS 定义了包括 CP0 在内的特权体系结构作为规范的一部分。为了适应所有的应用，MIPS 提供了规范的子集，允许根据需要只实现那些必要的特征，同时与 MIPS 体系结构兼容。GS232IP 实现了 MIPS32 Release1 特权体系结构中必须实现部分的全部特征，和可选择实现的部分特征：如 DEBUG 寄存器，Performance Counter 寄存器等。

本小节描述 GS232IP 实现的 CP0 指令以及 CP0 的寄存器定义。CP0 寄存器用于控制处理器的状态改变并报告处理器的当前状态。这些寄存器通过 MFC0 指令来读或 MTC0 指令来写。

当处理器运行在核心模式时或状态寄存器（Status 寄存器）中的第 28 位（CU0）被设置时，可以使用 CP0 指令。否则，执行 CP0 指令将产生“CP0 协处理器不可用例外”。

表 4-15 列出了 GS232IP 实现的所有 CP0 寄存器。

表 4-15 GS232IP 实现的 CP0 寄存器

0	Index	可写的寄存器，用于指定需要读/写的 TLB 表项
1	Random	用于 TLB 替换的伪随机计数器
2	EntryLo0	TLB 表项低半部分中对应于偶虚页的内容（主要是物理页号）
3	EntryLo1	TLB 表项低半部分中对应于奇虚页的内容（主要是物理页号）
4	Context	32 位寻址模式下指向内核的虚拟页转换表（PTE）
5	Page Mask	设置 TLB 页大小的掩码值
6	Wired	固定连线的 TLB 表项数目（指不用于随机替换的低端 TLB 表项）
7	HWREna	读硬件寄存器时用到的 mask 位(R2)
8	BadVaddr	错误的虚地址
9	Count	计数器
10	EntryHi	TLB 表项的高半部分内容（虚页号和 ASID）
11	Compare	计数器比较
12	Status(select0)	处理器状态寄存器
	IntCtl(select1)	控制扩展的中断功能（R2）
	SRSCtl(select2)	控制对影子寄存器的操作（R2）
	SRSMap(select3)	影子寄存器与中断向量的对应关系(R2)
13	Cause	最近一次例外的原因
14	EPC	例外程序计数器
15	PRID	处理器修订版本标识号
	Ebase	保存异常当 BEV 为 0 时的向量入口的基址
16	Config0	配置寄存器（Cache 大小等）
	Config1	配置寄存器
	Config2	在没有二级 cache 的 R2 实现中，仅表示实现

		Config3
	Config3	配置寄存器 （R2）
	Config6	配置分支预测的策略
17	LLAddr	链接读内存地址
18	WatchLo	虚地址空间访问陷阱地址
19	WatchHi	
20		
21		保留
22		保留
23	Debug	
24	DEPC	EJTAG debug 例外程序计数器
25	Performance Counter	性能计数器的高半部分
26		保留
27		保留
28	TagLo	CACHE TAG 寄存器的低半部分
29		
30	ErrorEPC	错误例外程序计数器
31	DESAVE	EJTAG debug 例外保存寄存器

Index 寄存器(0,select 0)

Index 寄存器是个 32 位可读/写的寄存器，其中最后六位的值用于索引 TLB 的表项。寄存器的最高位表示 TLB 探测(TLBP)指令执行是否成功。

Index 寄存器的值指示 TLB 读(TLBR)和 TLB 索引写(TLBWI)指令操作的 TLB 表项。

图 4-2 表示 Index 寄存器的格式，表 4-16 描述了 Index 寄存器各域的含义。

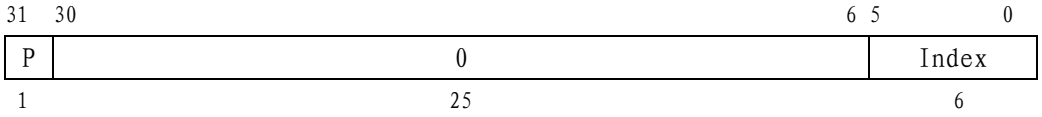


图 4-2 Index 寄存器

表 4-16 Index 寄存器各域描述

域	描述
P	探测失败。上一次 TLB 探测指令（TLBP）没有成功时置 1
Index	指示 TLB 读指令和 TLB 索引写指令操作的 TLB 表项的索引值
0	保留。必须按 0 写入，读时返回 0。

Random 寄存器(1,select1)

Random 寄存器是个只读寄存器，其中低六位索引 TLB 的表项。每执行完一条指令，该寄存器值减 1。同时，寄存器值在一个上界和一个下界之间浮动，上下界具体是：

- 下界等于保留给操作系统专用的 TLB 项数（即 Wired 寄存器的内容）。
- 上界等于整个 TLB 的项数减 1（最大为 32-1）。

Random 寄存器指示将由TLB随机写指令操作的TLB项。从这个目的来说，无需读此寄存器。但该寄存器是可读的，以验证处理器相应的操作是否正确。

为了简化测试，Random 寄存器在系统重起时置为上界。另外，当 Wired 寄存器被写时，该寄存器也要置为上界。

图 4-3 表示 Random 寄存器的格式，而表 4-17 描述 Random 寄存器各域的含义。

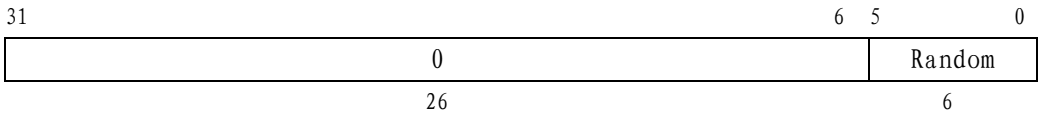


图 4-3 Random 寄存器

表 4-17 Random 寄存器各域

域	描述
Random	随机 TLB 索引值
0	保留。必须按 0 写入，读时返回 0。

EntryLo0 (2,select 0)以及 EntryLo1 (3, select 0)寄存器

EntryLo 寄存器包括两个相同格式的寄存器：

- EntryLo0 用于偶虚页
- EntryLo1 用于奇虚页

EntryLo0 和 EntryLo1 寄存器都是可读/写寄存器。当执行 TLB 读和写操作时，它们分别包括 TLB 项中奇偶页的物理页号（PFN）。图 4-4 表示这些寄存器的格式。

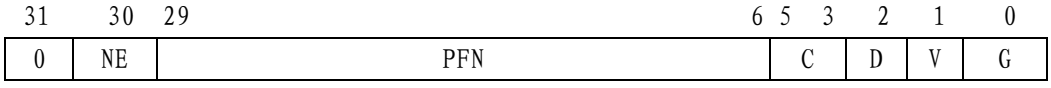


图 4-4 EntryLo0 和 EntryLo1 寄存器

EntryLo0 和 EntryLo1 寄存器的 PFN 域是 32 位物理地址中的高 28 位（39：12）。

表 4-18 EntryLo 寄存器域

域	描述
NE	不可执行位。1 表示不可执行，0 表示可执行。
PFN	页号，是物理地址的高位。
C	TLB 页的 Cache 一致性属性。

D	脏位。如果该位被设置，页面则标记为脏，也就是可写的。实际上这一位在软件中作为防止数据被更改的写保护使用。
V	有效位。当该位被设置时，说明 TLB 表项是有效的，否则将产生一个 TLBL 或 TLBS 例外。
G	全局位。当 EntryLo0 和 EntryLo1 中的 G 位都被设置为 1 时，处理器将在 TLB 查找时忽略 ASID。
0	保留。必须按 0 写入，读时返回 0。

在每个 TLB 表项中只有一个全局位，在 TLB 写操作中根据 EntryLo0[0]和 EntryLo1[0]的值写入。

Context (4, select 0)

Context 寄存器是一个读/写寄存器，它包含指向页表中某一项的指针。该页表是一个操作系统数据结构，存储虚拟地址到物理地址的转换。

当 TLB 缺失时，CPU 将根据缺失转换从页表中加载 TLB。一般情况下，操作系统使用 Context 寄存器寻址页表中当前页的映射。Context 寄存器复制 BadVAddr 寄存器中的部分信息，但是该信息被安排成一种利于软件 TLB 例外处理程序处理的形式。

图 4-5 显示了 Context 寄存器的格式；表 4-19 描述了上下文寄存器字段。

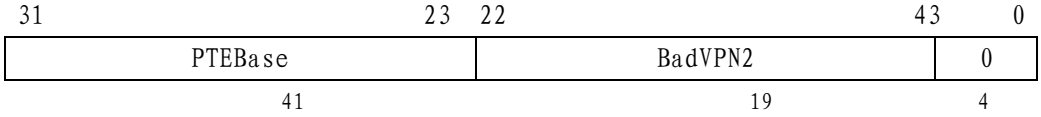


图 4-5 Context 寄存器

表 4-19 Context 寄存器域

域	描述
BadVPN2	当缺失时这一字段被硬件写。它包含最近不能进行有效转换的虚地址的虚页号(VPN)。
PTEBase	这一字段是操作系统使用的读/写字段。该字段写入的值允许操作系统将 Context 寄存器作为一个指向内存中当前页表的指针。
0	保留。必须按 0 写入，读时返回 0。

19 位的 BadVPN2 字段包含导致 TLB 缺失的虚地址的 31:13 位；第 12 位被排除是因为一个单一的 TLB 项映射到一个奇偶页对。对于一个 4K 字节的页尺寸，这一格式可以直接寻址 PTE 表项为 8 字节长且按对组织的页表。对于其它尺寸的页和 PTE，移动和屏蔽这个值可以产生合适的地址。

PageMask 寄存器(5, select 0)

PageMask 寄存器是个可读写的寄存器，在读写 TLB 的过程使用；它包含一个比较掩码，可为每个 TLB 表项设置不同的页大小，如表 4-20。该寄存器的格式如图 4-6。

TLB 读写操作使用该寄存器作为一个源或目的；当进行虚实地址转换时，TLB 中对应于 PageMask 寄存器的相应位指示虚地址位 24:13 中哪些位用于比较。当 MASK 域的值不是**错误！未找到引用源。**中的值时，TLB 的操作为未定义。0 域为保留，必须按 0 写入，读时返回 0。

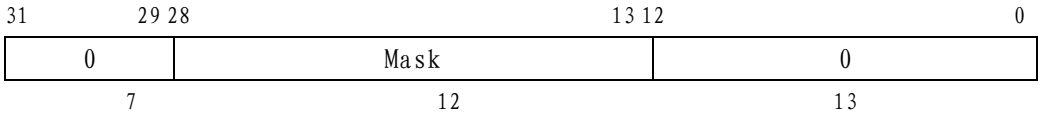


图 4-6 PageMask 寄存器

表 4-20 不同页大小的掩码（Mask）值

页大小	位															
	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13
4Kbytes	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16 Kbytes	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
64 Kbytes	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
256 Kbytes	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1
1 Mbytes	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
4 Mbytes	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
16M bytes	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1
64M bytes	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
256Mbytes	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Wired 寄存器(6, select 0)

Wired 寄存器是一个可读/写的寄存器，该寄存器的值指定了 TLB 中固定表项与随机表项之间的界限，如图 4-7 所示。Wired 表项是固定的、不可替换的表项，这些表项的内容不会被 TLB 写操作修改。而随机表项的内容可以被修改。

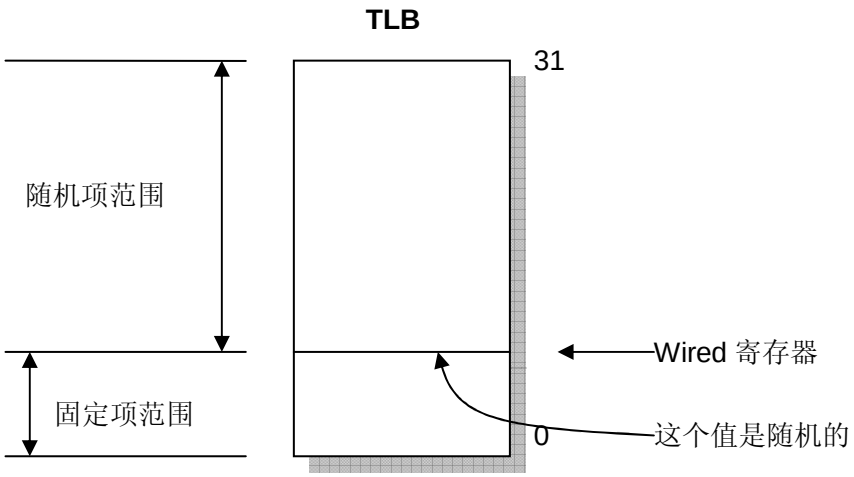


图 4-7 Wired 寄存器界限

Wired 寄存器在系统复位时置 0。写该寄存器的同时，Random 寄存器值要置为上限值（参阅前面的 Random 寄存器）。

图 4-8 表示 Wired 寄存器的格式；表 4-21 描述了寄存器的域。

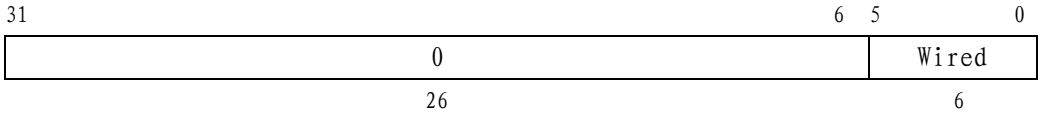


图 4-8 Wired 寄存器

表 4-21 Wired 寄存器域

域	描述
Wired	TLB 固定表项边界
0	保留。必须按 0 写入，读时返回 0。

HWREna 寄存器(7, select0)

硬件寄存器读使能寄存器（HWREna）包含一位的掩码用来决定当使用 RDHWR 指令时，对应的硬件寄存器是否可读。

图 4-9 显示了 HWREna 寄存器的格式，表 4-22 描述了 HWREna 寄存器各个域的意义。

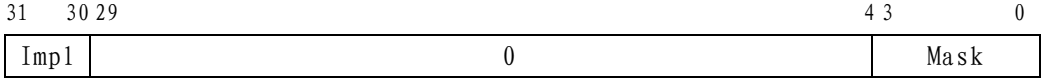


图 4-9 HWREna 寄存器

表 4-22 HWREna 寄存器域

域	描述
Impl	保留域，必须写为 0，且读时返回 0

Mask	读硬件寄存器的掩码位，若为 1，则对应的硬件寄存器可读，否则不可读
0	保留。必须按 0 写入，读时返回 0。

BadVAddr 寄存器(8, select0)

错误虚地址寄存器（BadVAddr）是一个只读寄存器，它记录了最近一次导致 TLB 或寻址错误例外的虚拟地址。除非发生软件复位，NMI 或 Cache 错误例外，BadVAddr 寄存器将一直保持不变。否则这个寄存器就是未定义的。

图 4-10 显示了错误虚地址寄存器的格式。

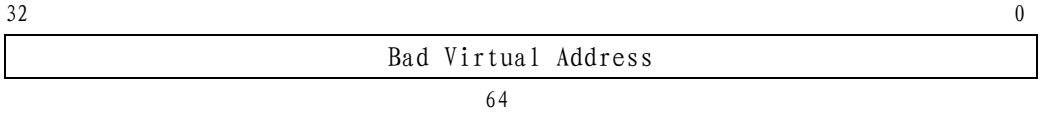


图 4-10 BadVAddr 寄存器

Count 寄存器(9,select0)以及 Compare 寄存器(11)

Count 寄存器和 Compare 寄存器都是 32 位读写寄存器，他们的模式如所示。

Count 寄存器作为一个实时的定时器工作，每两个时钟周期增 1。

Compare 寄存器用来在特定的时刻生成一个中断，该寄存器被写入一个值，图 4-11 并且不断地与 Count 寄存器中的值比较。一旦这两个值相等，Cause 寄存器里的中断位 IP[7]被设置。当 Compare 寄存器被再次写时这个中断位才被重置。

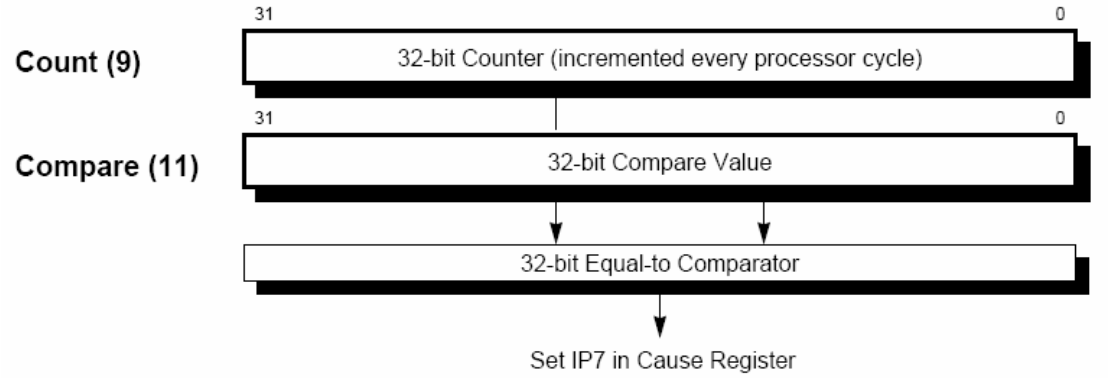


图 4-11 Count 寄存器和 Compare 寄存器

EntryHi 寄存器(10)

EntryHi 寄存器用于 TLB 读写时存放 TLB 表项的高位。

EntryHi 寄存器可以被 TLB Probe, TLB Write Random, TLB Write Indexed, 和 TLB Read Indexed 指令访问。

图 4-12 表示 EntryHi 寄存器的格式。表 4-23 表示 EntryHi 寄存器的域。

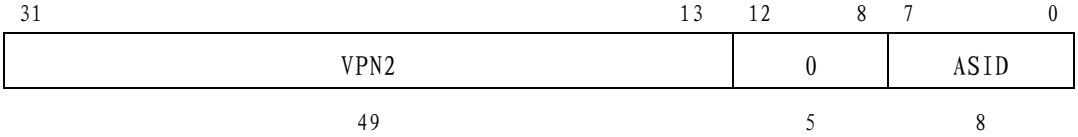


图 4-12 EntryHi 寄存器

表 4-23 EntryHi 寄存器域

域	描述
VPN2	虚页号除 2（映射到双页）；虚拟地址的高位。
ASID	地址空间标识域。一个 8 位的域；用于让多个进程共享 TLB；对于相同的虚页号，每个进程都与其他进程有不同的映射。
0	保留。必须按 0 写入，读时返回 0。

VPN2 域包含 32 位虚拟地址的 31:13 位。

当一个 TLB Refill，TLB Invalid，或 TLB Modified 例外发生时，没有匹配 TLB 表项的虚拟地址中虚拟页号（VPN2）和 ASID 将被加载到 EntryHi 寄存器。

Status 寄存器(12, select 0)

Status 寄存器(SR) 是一个读写寄存器，它包括操作模式，中断允许和处理器状态诊断。下面列表描述了一些更重要的 Status 寄存器字段；图 4-13 显示了整个寄存器的格式，包括域的描述。其中重要的域有：

- 8 位的中断屏蔽(IM)域控制 8 个中断条件的使能。中断在被触发之前必须被使能，在Status寄存器的中断屏蔽域和Cause寄存器的中断待定域相应的位都应该被置位。更多的信息，请参考 Cause 寄存器的中断待定（IP）域。
- 4 位的协处理器可用性（CU）域控制 4 个可能的协处理器的可用性。不管 CU0 位如何设置，在内核模式下 CP0 总是可用的。

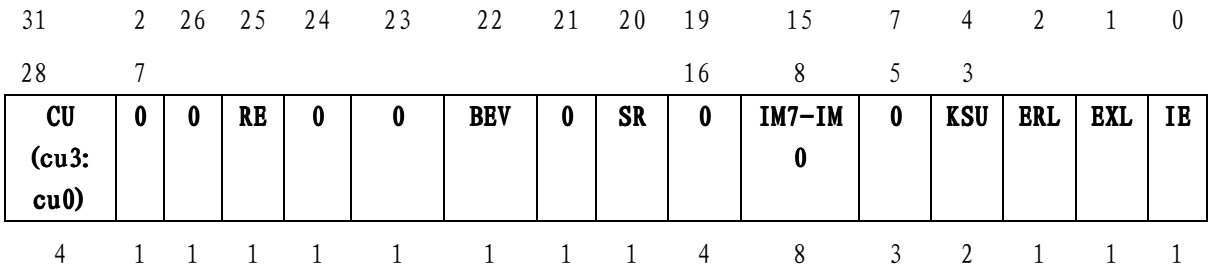


图 4-13 Status 寄存器

图 4-13 显示了 Status 寄存器的格式，表 4-24 描述了 Status 寄存器的域。

表 4-24 Status 寄存器域

域	描述
CU	控制 4 个协处理器单元的可用性。不管 CU0 位如何设置，在内核模式

	下 CP0 总是可用的。 1- 可用 0- 不可用 CU 域的初值是 0011
0	保留。必须按 0 写入，读时返回 0。
BEV	控制例外向量的入口地址 0 - 正常 1 - 启动
SR	1 表示有软复位例外发生
IM	中断屏蔽：控制每一个外部、内部和软件中断的使能。如果中断被使能，将允许它触发，同时 Cause 寄存器的中断 Pending 字段相应的位被置位。 0—禁止 1—允许
KSU	模式位 11 .未定义 10 .普通用户 01 .超级用户 00 .核心
ERL	错误级。当发生复位，软件复位，NMI 或 Cache 错误时处理器将重置此位。 0 .正常 1 .错误
EXL	例外级。当一个不是由复位，软件复位或 Cache 错误引发的例外产生时，处理器将设置该位。
IE	中断使能。 0 .禁用所有中断 1 .使能所有中断

Status 寄存器模式和访问状态

下面描述 Status 寄存器中用于设置模式和访问状态的域：

- **中断使能：**当符合以下条件时，中断被使能：
  - IE = 1 且
  - EXL = 0 且
  - ERL = 0。

如果遇到这些条件，IM 位的设置允许中断。

- **操作模式：**当处理器处于普通用户、内核和超级用户模式时需要设置下述位域。

- 当 KSU = 10_2, EXL = 0 和 ERL = 0 时处理器处于普通用户态模式下。
- 当 KSU = 01_2, EXL = 0 和 ERL = 0 时处理器处于超级用户态模式下。
- 当 KSU = 00_2, or EXL = 1 或者 ERL = 1 时处理器处于内核态模式下。

- **内核地址空间访问：** 当处理器处在内核模式时，可以访问内核地址空间。
- **超级用户地址空间访问：**当处理器处在内核模式或超级用户模式时，可以访问超级用户地址空间。
- **用户地址空间访问：**处理器在这三种操作模式下都可以访问用户地址空间。

Status 寄存器复位

复位时，Status 寄存器的值是 0x00400004。

IntCtl 寄存器(12,select 1)

在 Rlease2 的版本中，IntCtl 寄存器用来控制扩展的中断特性。包括向量中断和外部中断。图 4-14 列出了 IntCtl 寄存器的格式，表 4-25 描述了各个域的意义。

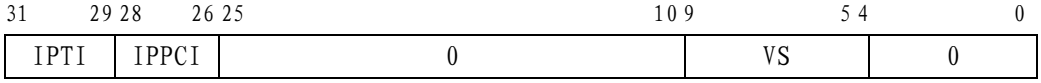


图 4-14 IntCtl 寄存器

表 4-25 IntCtl 寄存器域

域	描述
IPTI	对于向量中断而言，这个域表示时钟中断的中断号 Encoding / IP bit / Hardware Interrupt Source 2 / 2 / HW0 3 / 3 / HW1 4 / 4 / HW2 5 / 5 / HW3 6 / 6 / HW4 7 / 7 / HW5
IPPCI	表示 Performance Counter 的中断号。 Encoding / IP bit / Hardware Interrupt Source 2 / 2 / HW0 3 / 3 / HW1 4 / 4 / HW2 5 / 5 / HW3

	6 | 6 | HW4 7 | 7 | HW5
VS	在实现向量中断的版本中，这个域表示中断向量之间偏移差 Encoding | Spacing Between Vectors (hex) | Spacing Between Vectors (decimal) 0x00 | 0x000 | 0 0x01 | 0x020 | 32 0x02 | 0x40 | 64 0x04 | 0x80 | 128 0x08 | 0x100 | 256 0x10 | 0x200 | 512
0	保留。必须按 0 写入，读时返回 0。

SRSCtl 寄存器(12,select 2)

控制影子寄存器的读写。GS232 未实现影子寄存器。固所有域都写为 0。

31 30	29 26	25 22	21 18	17 16	15 12	11 10	9 6	5 4	3 0
0	HSS	0	EICSS	0	ESS	0	PSS	0	CSS

图 4-15SRSCtl 寄存器

表 4-26 SRSCtl 寄存器域

域	描述
HSS	表示实现的影子寄存器组数；若为 0 表示没有影子寄存器实现
EICSS	EIC 中断的影子寄存器组
ESS	异常的影子寄存器组
PSS	前一个影子寄存器组
CSS	当前影子寄存器组
0	保留。必须按 0 写入，读时返回 0。

SRSMap 寄存器(12,select 3)

用来反映影子寄存器与异常向量的对应关系。GS232 未实现影子寄存器。固所有域都写为 0。

31	0

图 4-16 SRSMap 寄存器

Cause 寄存器(13,select 0)

32 位的可读写 Cause 寄存器描述了最近一个例外发生的原因。

图 4-17 显示了这一寄存器的域，表 4-27Cause 寄存器域描述了 Cause 寄存器的域。一个 5 位例外码（ExcCode）指出了原因之一，如表 4-27 所示。

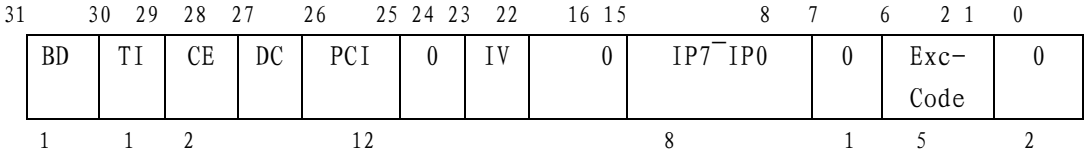


图 4-17 Cause 寄存器

表 4-27Cause 寄存器域

域	描述
BD	指出最后采用的例外是否在分支延时槽中。 1－延时槽 0－正常
CE	当发生协处理器不可用例外时协处理器的单元编号。
DC	关掉 Count 寄存器。DC=1 时关掉 count 寄存器
PCI	Performance counter 中断,用来指示是否有待处理的 PC 中断
IV	指示中断向量是否用普通的异常向量（0 表示是，1 表示用特殊向量）
IP	指出等待的中断。该位将保持不变直到中断撤除。IP0¯IP1 是软中断位，可由软件设置与清除。 1－中断等待 0－没有中断
ExcCode	例外码域 (见表5-11)
0	保留。必须按 0 写入，读时返回 0。

表 4-28 Cause 寄存器的 ExcCode 域

例外代码	Mnemonic	描述
0	INT	中断
1	MOD	TLB 修改例外
2	TLBL	TLB 例外（读或者取指令）
3	TLBS	TLB 例外（存储）
4	ADEL	地址错误例外（读或者取指令）
5	ADES	地址错误例外（存储）
6	IBE	总线错误例外（取指令）
7	DBE	总线错误例外（数据引用：读或存储）
8	SYS	系统调用例外
9	BP	断点例外
10	RI	保留指令例外
11	CPU	协处理器不可用例外

12	OV	算术溢出例外
13	TR	陷阱例外
14	–	保留
15	FPE	浮点例外
16－22	–	保留
23	WATCH	WATCH 例外
24－30	–	保留
31	–	保留

Exception Program Counter 寄存器(14, select0)

例外程序计数器（Exception Program Counter，简称 EPC）是一个读/写寄存器，它包括例外处理结束后的继续处理地址。

对于同步例外，EPC 寄存器的内容是下面之一：

- 指令虚地址，这是导致例外的直接原因，或者
- 之前的分支或者跳转指令（当指令在分支延时槽中，指令延时位在 Cause 寄存器中被置位）的虚地址。

当 Status 寄存器中的 EXL 位被置 1 时，处理器不写 EPC 寄存器。

图 4-18 显示了 EPC 寄存器的格式。

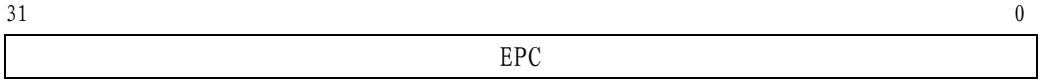


图 4-18 EPC 寄存器

Processor Revision Identifier (PRID)寄存器(15)

PRId 寄存器是个 32 的只读寄存器，该寄存器包含了标定处理器和 CP0 版本的实现版本和修订版本的信息。图 4-19 表示了该寄存器的格式；表 4-29 描述了该寄存器的域。

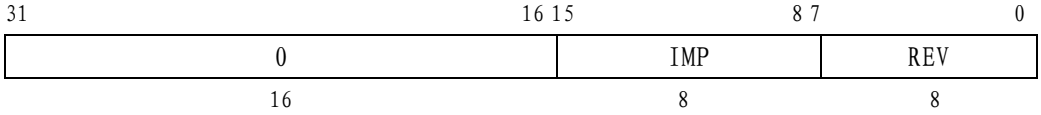


图 4-19 Processor Revision Identifier 寄存器

表 4-29 PRId 寄存器域

域	描述
IMP	实现版本号
REV	修订版本号

0	保留。必须按 0 写入，读时返回 0。

PRID 寄存器的低位（7：0 位）可用作修订版本的号码，而高位（15：8）位可用作实现版本的号码。龙芯 2F 实现版本号为 0x63，修订版本号为 0x02。

版本号码的表示格式为 Y.X，其中 Y（7：4 位）为主要版本号，而 X（3：0 位）为小版本号。

版本号码可以区分一些处理器的版本，但不能保证处理器的任何改动要体现在 PRID 寄存器中，换句话说，不能保证版本号的改动必须体现处理器的修改。因为这个原因，寄存器的值没有给出，而软件也不能依赖 PRID 寄存器中的版本号来标识处理器。

Config0 寄存器(16，select0)

Config 寄存器规定了 GS232 处理器中各种配置选择项；表 4-30 列出了这些选项。

由 Config 寄存器的位 31:3 所定义的一些配置选项，在复位时由硬件设置，而且作为只读状态位包括在 Config 寄存器中，用于软件的访问。其他配置选项（Config 寄存器的位 2:0）是可读/写的并且由软件所控制。在复位时这些域是没有定义的。

Config 寄存器的配置是受限的。Config 寄存器在 Cache 被使用之前应该由软件来初始化，并且，在做了任何改变后 Cache 应该重新初始化。

图 4-20 表示了 Config 寄存器的格式；表 4-30 Config 寄存器域描述了 Config 寄存器的域。Config 寄存器的初值为 0x00030932。

Config 寄存器

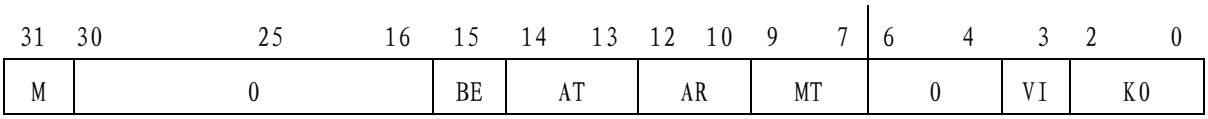


图 4-20 Config 寄存器

表 4-30 Config 寄存器域

域	描述
M	表示 config1 寄存器是否实现
BE	1：大尾端 0：小尾端
AT	0：MIPS32 兼容 1：仅能访问 3 2 位地址的 MIPS64 兼容 2：MIPS64 兼容 3：保留

AR	0: release1 1: release2 2-7: 保留
MT	MMU 类型
VI	指令 cache 是否是虚 cache
K0	Kseg0 的 Cache 一致性算法。 7 – Uncached Accelerated 3 – Cachable 2 – Uncached

Config1 寄存器(16，select1)

Config1 寄存器辅助 Config0 寄存器规定了 GS232 处理器中其他各种配置选择项；其中包括 Icache，Dcache 的各项配置参数，比如每路的组数，cache 行的大小，相关联数。如果 cache 行大小为 0，则表示 cache 没有实现。

31	30 25	24 22	21 19	18 16	15 13	12 10	9 7	6	5	4	3	2	1	0
M	MMU Size -1	IS	IL	IA	DS	DL	DA	C2	0	PC	WR	CA	EP	FP

图 4-21 Config 寄存器

表 4-31 Config 寄存器域

域	描述
M	用来指示是否实现了 config2 寄存器
MMU Size-1	TLB 表项大小减一
IS	Icache 每路的组数
IL	Icache 每行的大小
IA	Icache 组相联数
DS	Dcache 每路的组数
DL	Dcache 每行的大小
DA	Dcache 组相联数
C2	Coprocessor2 是否实现
PC	Performance Counter 是否实现
WR	Watch 寄存器是否实现
CA	代码压缩是否实现
EP	EJTAG 是否实现
FP	浮点功能单元是否实现
0	保留域，必须写 0

Config2 寄存器(16，select2)

Config2 定义了二级缓存的参数，因为 GS232 没有实现二级缓存，固此仅用此寄存器的最高位表示实现了 Config3 寄存器。

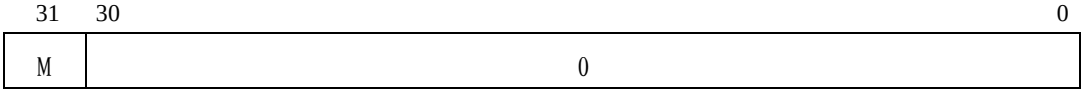


图 4-22 Config 寄存器

表 4-32 Config 寄存器域

域	描述
0	保留。必须按 0 写入，读时返回 0。
M	Config3 寄存器是否实现

Config3 寄存器(16，select3)

Config3 寄存器辅助 Config0 寄存器规定了 GS232 处理器中其他各种配置选择项；

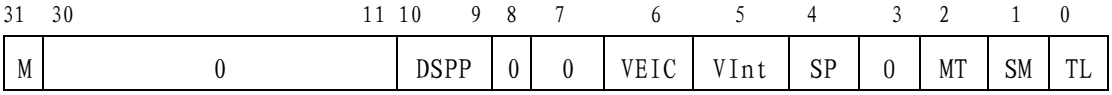


图 4-23 Config 寄存器

表 4-33 Config 寄存器域

域	描述
M	用来指示是否实现了 config2 寄存器
DSPP	DSP 是否实现
VEIC	是否实现了外部中断
VInt	是否实现了向量中断
SP	是否支持小物理页
MT	是否实现了 MTASE
SM	是否实现了 Smart ASE
TL	是否实现了 Trace 逻辑

Config6 寄存器(16，select6)

Config6 寄存器为 GS232 自己定义是用的控制寄存器，用来配置各种分支预测方式，以及表示是否实现实时中断。

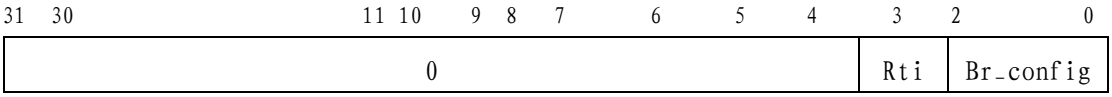


图 4-24 Config 寄存器

表 4-34 Config 寄存器域

<table>
<tr><th>域</th><th>描述</th></tr>
<tr><td>0</td><td>保留域</td></tr>
<tr><td>Rti</td><td>是否实现实时中断</td></tr>
<tr><td>Br_config</td><td>分支预测方式，具体如下：
<table>
<tr><td>编码</td><td>分支预测方式</td></tr>
<tr><td>3’b000</td><td>Gshare 索引 bht</td></tr>
<tr><td>3’b001</td><td>Pc 索引的 bht</td></tr>
<tr><td>3’b010</td><td>总是跳转</td></tr>
<tr><td>3’b011</td><td>总是不跳</td></tr>
<tr><td>3’b100</td><td>向前跳转</td></tr>
<tr><td>3’b101</td><td>向后跳转</td></tr>
</table></td></tr>
</table>

Load Linked Address (LLAddr)寄存器(17,select0)

可读/写寄存器

WatchLo 寄存器(18,select0)

WatchLo 与 WatchHi 寄存器共同构成 Watch 例外接口。WatchLo 是 32 位的寄存器，包含位于虚地址空间一个双字的虚地址。如果被使能，任何读或写这个位置都将引发一个 Watch 例外。这个特性是为了调试使用的。只有在 Status 寄存器的 EXL 和 ERL 位都为 0 时才发生 Watch 例外。

图 4-25 描述 WatchLo 寄存器的格式，表 4-35 描述了 WatchLo 寄存器的域。

图 4-25 WatchLo 寄存器

表 4-35 WatchLo 寄存器域

域	描述
VADDR	虚地址的 31：3 位
I	如果设成 1，则在取指时发生例外
R	如果设成 1，在 Load 时发生例外。
W	如果设成 1，在 Store 时发生例外。
0	保留。必须按 0 写入，读时返回 0。

WatchHi 寄存器(19,select0)

WatchHi 寄存器是 32 位的寄存器，包含与 WatchLo 中虚地址的其他信息，比如 ASID， G 位，Mask 位。如果 G 位为 1，那么任何与 WatchLo 匹配的虚地址都将引发 Watch 例外。如果 G 位为 0，那么只有当 WatchHi 寄存器中的 ASID 位与 EntryHi 寄存器中的 ASID 位匹配，且虚地址与 WatchLo 匹配才能引发 Watch 例外。

图 4-26 描述 WatchHi 寄存器的格式，表 4-36 描述了 WatchHi 寄存器的域。

WatchHi 寄存器

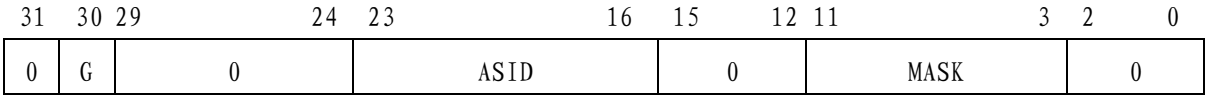


图 4-26 WatchHi 寄存器

表 4-36 WatchHi 寄存器域

域	描述
G	如果 G 位为 1，那么任何与 WatchLo 匹配的虚地址都将引发 Watch 例外。如果 G 位为 0，那么只有当 WatchHi 寄存器中的 ASID 位与 EntryHi 寄存器中的 ASID 位匹配，且虚地址与 WatchLo 匹配才能引发 Watch 例外
ASID	地址空间标识域
MASK	虚地址掩码
0	保留。必须按 0 写入，读时返回 0。

Performance Counter 寄存器(25)

GS232 处理器定义了两个性能计数器（Performance Counter），他们分别映射到 CP0 寄存器 25 号的 select 1 与 select 3 寄存器。对应的关联控制寄存器分别影射到 P0 寄存器 25 号的 select 0 与 select 2 寄存器。每个计数器都是 32 位的读/写寄存器，并且在每次关联控制域中可数事件发生时自增。每个计数器都可以独立对一种事件计数。

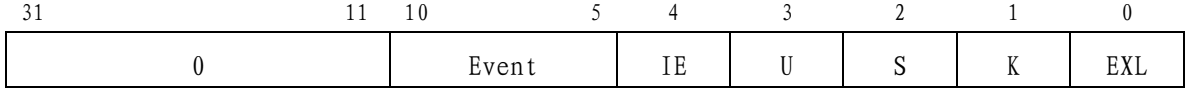


图 4-27 控制寄存器性能计数寄存器

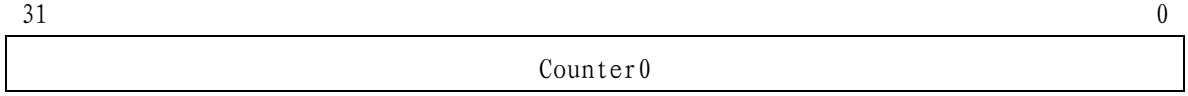


图 4-28 性能计数器寄存器

当计数器的首位（31 位）变成 1（计数器溢出）时，计数器将触发一个中

断 IP[6]，关联控制域使能中断。在计数器溢出后无论中断是否被告知，计数都将继续。表 4-37 描述 24 号寄存器控制域的格式。表 4-38 描述计数使能位的定义。表 4-39 描述计数器 0 和计数器 1 各自的事件。

表 4-37 控制域格式

[10:5]	[4]	[3:0]
Event 1 Select	IP[6] Interrupt Enable	计数使能位(K/S/U/EXL)

表 4-38 计数使能位定义

计数使能位	Count Qualifier(CP0 Status 寄存器域)
K	KSU = 0 (内核模式), EXL = 0, ERL = 0
S	KSU = 1 (超级用户模式), EXL = 0, ERL = 0
U	KSU = 2 (普通用户模式), EXL = 0, ERL = 0
EXL	EXL = 1, ERL = 0

表 4-39 计数器 0/1 事件

事件	内部信号	描述
0000	Cycles	周期
0001	Brbus.valid	分支指令
0010	Jrcount	JR 指令
0011	Jr31count	JR 指令并且域 rs=31
0100	Imemread.valid& Imemread_allow	一级 I-cache 缺失
0101	Rissuebus0.valid	Alu1 操作已发射
0110	Rissuebus2.valid	Mem 操作已发射
0111	Rissuebus3.valid	Falu1 操作已发射
1000	Brbus_bht	BHT 猜测指令
1001	Mreadreq.valid& Mreadreq_allow	从主存中读
1010	Fxqfull	固定发射队列满的次数
1011	Roqfull	重排队列满的次数
1100	Cp0qfull	CP0 队列满的次数
1101	Exbus.ex & Excode=34,35	T1b 重填例外
1110	Exbus.ex & Excode=0	例外
1111	Exbus.ex & Excode=63	内部例外

TagLo(28)寄存器

TagLo 寄存器是 32 位读/写寄存器，用于保存一级/ 二级 Cache 的标签和状态，使用 CACHE 和 MTC0 指令往 Tag 寄存器写。

图 4-29 显示了这些寄存器用于一级 Cache（P-Cache）操作的格式。

表 4-40 Cache Tag 寄存器域列出了 TagLo 和 TagHi 寄存器中域的定义。

TagLo 寄存器

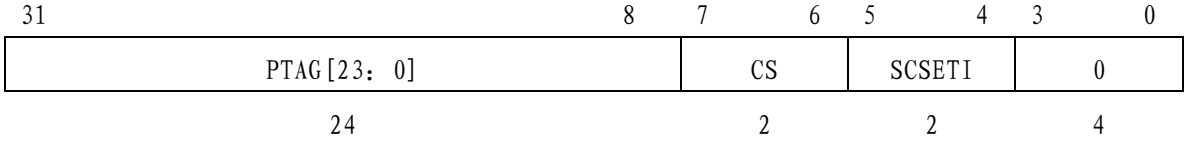


图 4-29 TagLo 寄存器(P-Cache)

表 4-40 Cache Tag 寄存器域

域	描述
PTAG	指定物理地址的 31:12 位。
CS	指定 Cache 的状态。
SCSETI	对应 Cache 行在二级 Cache 的组号（二级 Cache 该域为 0）
0	保留。必须按 0 写入，读时返回 0。

ErrorEPC 寄存器(30)

除了用于 ECC 和奇偶错误例外外，ErrorEPC 寄存器与 EPC 寄存器类似。它用于在复位、软件复位、和不可屏蔽中断（NMI）例外时存储程序计数器。

ErrorEPC 是一个读写寄存器，它包括处理一个错误后指令重新开始执行的虚拟地址。图 4-30 显示了 ErrorEPC 寄存器的格式。

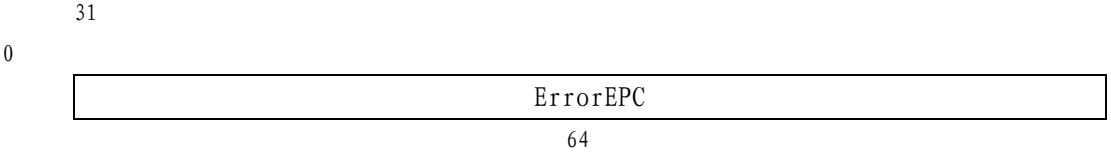


图 4-30 ErrorEPC 寄存器

4.2 CP0 指令

表 4-41 列出了 GS232 处理器定义的 CP0 指令。

表 4-41 CP0 指令

指令	描述
CACHE	CACHE 操作

DMFC0	从 CP0 取双字
DMTC0	将双字送到 CP0
ERET	例外返回
MFC0	从 CP0 取数据
MTC0	将数据送到 CP0
TLBP	查询 TLB 项
TLBR	用索引读 TLB 表项
TLBWI	用索引填充 TLB 表项
TLBWR	随机填充 TLB 表项

相关

龙芯处理器能够处理硬件中的流水线相关，包括 CP0 相关和访存相关，因此 CP0 指令并不需要 NOP 指令来校正指令序列。

4.3 EJTAG 设计

4.3.1 EJTAG 介绍

EJTAG 是 MIPS 公司制定的片上调试规范，它根据 IEEE1149.1 协议的基本构造和功能扩展而来。EJTAG 在被调试系统的处理器内部嵌入额外的控制模块，当满足一定的触发条件时进入一种特殊的调试模式。进入调试模式后，当前执行程序停止运行，处理器执行调试异常处理程序。在异常处理程序中，被调试系统可以进行各种 EJTAG 操作，并可以通过 TAP 与调试主机进行通信。被调试系统执行 DERET 指令后从调试模式退出。被调试系统退出调试模式后，从产生调试异常处继续执行指令。调试主机可以通过被调试系统外部特设的通信接口（TDI，TDO，TMS，TCK）访问一些处理器内部资源如寄存器，存储器等。调试主机通信端口与被调试系统调试通信接口通过一块简单的信号转换电路板即调试卡（probe）进行连接。下图为 EJTAG 调试的连接示意图。

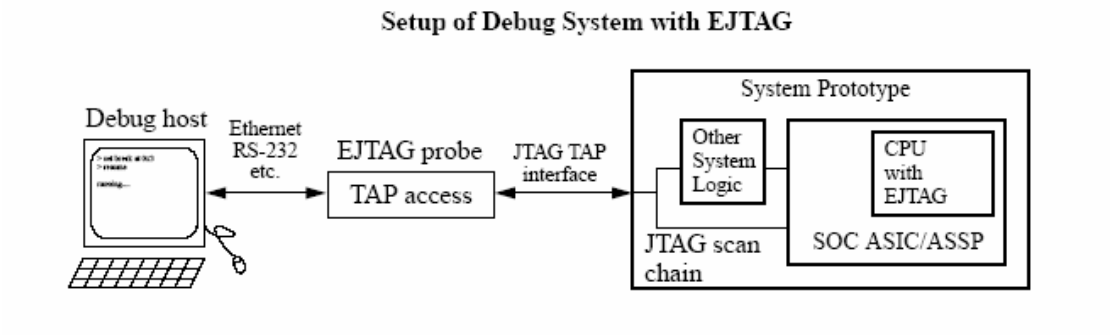


图 4-31 EJTAG 调试连接示意图

GS232IP 实现了 EJTAG3.1 的规范。主要提供了以下几个方面的功能：

调试例外和调试模式。为了能在处理器执行过程中观察处理器内部状态，EJTAG 规定了若干调试例外。当调试例外发生的时候，处理器进入调试模式。在调试模式下可以没有限制的访问协处理器，内存区域等，在调试模式下其他调试例外及外部中断等被忽略。例外处理程序在调试模式下执行，可以由调试主机提供也可以由被调试系统自己提供。龙芯 2 号 IP 实现了八种调试例外，分别是：DSS（单步调试异常），DINT（外部调试中断），DDBL（load 数据断点），DDBS（store 数据断点），DIB（指令断点）和 DBP（执行 SDBBP 指令例外），DDBSImpr(精确带值比较的 store 数据断点)，DDBLImpr（精确带值比较的 load 数据断点)。所有调试例外都是精确的，注意这里，DDBSImpr，DDBLImpr 与 EJTAG 规范规定的意义不同。

板外的 EJTAG 内存.EJTAG 允许处理器在调试模式下能够从系统之外存取指令或者数据。EJTAG 的这部分内存被映射到处理器的内存空间，从处理器看出去是被映射到 kseg3 段的虚拟地址。

调试断点指令。EJTAG 引入了一条新的断点指令 SDBBP，SDBBP 指令将处理器置于调试状态，然后从 EJTAG 内存的相应地址取出其相关的处理代码。

硬件断点。EJTAG 同时定义了两种硬件断点。一种是指令断点，是处理器从特定的虚拟地址取指的时候触发；另外一种是数据断点，当处理器从一个特定的虚拟地址存/取值的时候触发。GS232IP 实现了精确硬件断点，同时支持 2 个指令断点和 8 个数据断点（？？）。

DDBSImpr，DDBLImpr 例外的实现：不实现非精确硬件断点例外，通过分析发现，目前处理器的时间只有一种情况需要特殊处理才能避免发生非精确硬件断点例外，就是一个 load 操作不命中，同时硬件断点又需要值比较的时候，由于 nonblocking 的设计，后面如果上来了 store 或 MTC0 指令，就不能保证精确了。而其它情况都能立刻从硬件断点部件返回例外信号，中止该指令的操作，使其立刻返回操作队列。

GS232IP 中与 EJTAG 相关的硬件支持主要包括：调试控制寄存器（DCR）、硬件断点寄存器、支持 EJTAG 调试功能的处理器核扩展、EJTAG TAP 等部分。下面分几个部分分别详细介绍。

4.3.2 调试控制寄存器（Debug Control Register）

调试控制寄存器（Debug Control Register，以下简称 DCR）提供 EJTAG 调试的一些重要配置和状态信息。它的地址是 0XFFFF FFFF FF20 0000，也就

是 drseg 段偏移量为 0 处。在 GS232IP 中，DCR 是 32 位的。

DCR 提供了非调试模式下的外部中断和 NMI 控制、未决的 NMI 信号指示、指令断点和数据断点是否可用、PC 采样是否可用以及采样周期等主要功能。图 4-32 DCR 寄存器格式描述了 DCR 的格式，表 4-42 DCR 寄存器域具体介绍了 DCR 中各个域。

31 30	29	28 18	17	16	15 10	9	8 6	5	4	3	2	1	0
0	ENM	0	DataBrk	InstBrk	0	PCS	PCR	0	IntE	NMIE	NMIpend	SRstE	ProbEn

图 4-32 DCR 寄存器格式

表 4-42 DCR 寄存器域

域名	所在位	描述	读/写
ENM	29	处理器在 Kernel 态和调试模式下的尾端属性： 0：小尾端 1：大尾端	只读
DataBrk	17	是否实现了硬件数据断点： 0：未实现硬件数据断点 1：实现了硬件数据断点	只读
InstBrk	16	是否实现了硬件指令断点： 0：未实现硬件指令断点 1：实现硬件指令断点	只读
PCS	9	是否实现了 PC 采样机制： 0：未实现采样机制 1：实现了采样机制	只读
PCR[2:0]	8：6	PC 采样频率：PCR 的值 0 到 7 分别表示每 2 的 5 次方，2 的 6 方，2 的 7 方，2 的 8 方，2 的 9 方，2 的 10，2 的 11 和 2 的 12 次方拍进行一次采样。	可读 可写

IntE	4	是否允许非调试模式下的中断： 0：不允许中断 1：允许中断	可读 可写
NMIE	4	是否允许非调试模式下的 NMI 中断： 0：不允许 NMI 1：允许 NMI	可读 可写
NMIpend	4	未决 NMI 指示位： 0：无未决 NMI 1：有未决 NMI	只读
Proben	0	同 ECR 寄存器 Proben 域： 0：不允许访问 dmseg 1：允许访问 dmseg	只读
0	31：30 28：18 15：10 5：1	必须写入 0；读出时返回 0。	只读

4.3.3 硬件断点

GS232IP 提供了 2 个硬件指令断点和 8 个硬件数据断点。在非调试上的模式下，一旦断点条件被满足，根据相应断点控制寄存器的具体位置，发生硬件断点例外或设置断点标志位。在 GS232IP 中，指令和数据断点例外都是精确例外，数据断点例外不进行值的比较。图 4-33 是指令断点和数据断点的概况。表 4-43 是指令断点寄存器和数据断点寄存器在 drseg 中的偏移量。

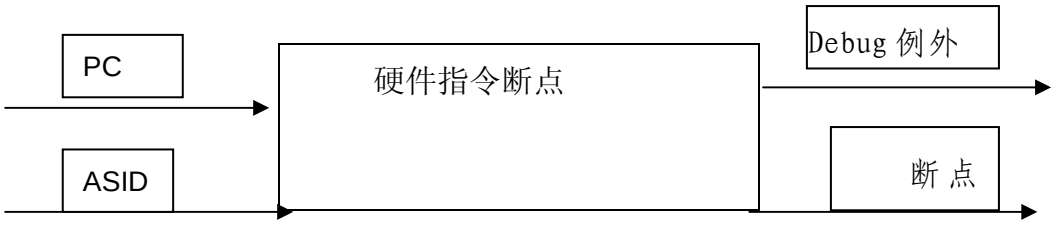


图 4-33 硬件指令、数据断点概况

需要特别指出的是，从断点例外返回会回到触发断点的指令。如果软件不在例外服务程序中禁用相应的断点，会导致重复发生断点例外。

表 4-43 硬件断点寄存器

寄存器简称	寄存器名	在 drseg 中偏移量
数据断点寄存器偏移量		
DBS	数据断点状态寄存器	0x2000
DBAn	数据断点地址寄存器 n	0x2100＋0x100×n
DBMn	数据断点掩码寄存器 n	0x2108＋0x100×n
DBASIDn	数据断点 ASID 寄存器 n	0x2110＋0x100×n
DBCn	数据断点控制寄存器 n	0x2118＋0x100×n
指令断点寄存器偏移量		
寄存器代码	寄存器描述	在 drseg 中的地址
IBS	指令断点状态寄存器	0x1000
IBAn	指令断点地址寄存器 n	0x1100＋0x100×n
IBMn	指令断点掩码寄存器 n	0x1108＋0x100×n
IBASIDn	指令断点 ASID 寄存器 n	0x1110＋0x100×n
IBCn	指令断点控制寄存器 n	0x1118＋0x100×n
注： n 代表第 n 个断点相关的寄存器（指令断点 n 从 0 到 3，数据断点 n 从 0 到 7）。		

图 4-34 描述了 IBS 的格式，表 4-44 具体介绍了 IBS 中的各个域。

31	30	29 28	27 24	23 16	15	14 0
0	ASIDsup	0	BCN	0	IBPTshare	BS[14：0]

图 4-34 IBS 寄存器格式

表 4-44 IBS 域描述

域		描　　述	读/写
域名	所在位		
ASIDsup	30	表示在指令断点中是否支持 ASID 的比较： 0：不支持 ASID 比较 1：支持 ASID 比较	只读
BCN	27：24	表示所实现的指令断点数： 0：保留 1—15：指令断点数	只读
BS[14:0]	14：0	断点标志位：BS[n]代表第 n 个指令断点匹配条件被满足了。在 GS232IP 中只有 BS[3:0]有效，BS[14:4]必须全部为 0.	可读/ 可写入 0
0	31 29：28 23：15	必须写入 0；读出时返回 0。	只读

图 4-35 描述了 IBAn 的格式，每个 IBAn 的内容表示一个指令断点比较的虚地址。

图 4-36 描述了 IBMn 的格式，每个 IBMn 的内容表示一个指令断点比较的地址掩码，IBMn 的每一位为“1”表示相应位不需要比较，为“0”表示相应位需要比较。

图 4-36 描述了 IBCn 的格式，表 4-45 具体介绍了 IBCn 中的各个域。

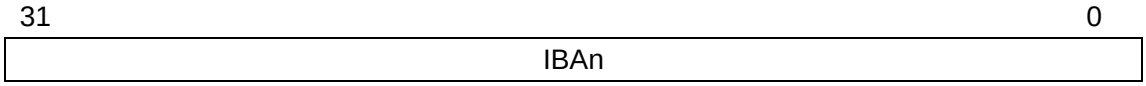


图 4-35 IBAn 寄存器格式

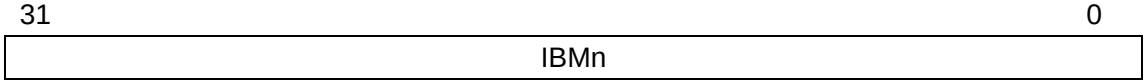


图 4-36 IBMn 寄存器格式

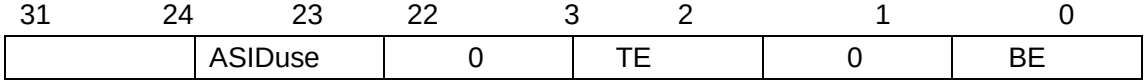


图 4-37 IBCn 寄存器格式

表 4-45 IBCn 域描述

域		描　　述	读/写
域名	所在位		
ASIDuse	23	表示在指令断点 n 中是否使用 ASID 的比较： 0：不使用 ASID 比较 1：使用 ASID 比较	可读 可写
TE	2	表示指令断点 n 匹配时是否会设 IBS 中的 BS[n]位： 0：不设 BS[n] 1：设 BS[n]	可读 可写
BE	0	表示断点 n 的匹配是否引发断点例外： 0：不引发断点例外 1：引发断点例外	可读/ 可写
0	31:24 22：3 1	必须写入 0；读出时返回 0。	只读

指令断点的匹配条件是：

IB_match = （ ! IBCn_ASIDuse | | (ASID = = IBASIDn_ASID) ） && ((IBMn_IBM |

¯(PC ^ IBAn_IBA) = = ¯0))

需要指出的是：当 BE 位被置为 1 时，如果断点匹配上，即使 TE 位为 0，IBS 中的 BS[n]位依然会被置为 1.因此在进入断点例外后，软件可以通过查询 BS 来判断是哪个断点引起了例外。

图 4-38 描述了 DBS 格式，表 4-46 具体介绍了 DBS 中的各个域。

31	30	29	28	27	24	23	15	14 0
0	ASIDsup	NoSVmatch	NoLVmatch	BCN	0	BS[14：0]		

图 4-38 DBS 寄存器格式

表 4-46 DBS 域描述

域		描　　述	读/写
域名	所在位		

ASIDsup	30	表示在指令断点是否支持 ASID 的比较： 0：不支持 ASID 比较 1：支持 ASID 比较	只读
NoSVmatch	29	表示是否在数据断点的存储操作进行值比较： 0：即比较地址也比较值 1：只比较地址	只读
NoLVmatch	28	表示是否在数据断点的 load 操作进行值比较： 0：即比较地址也比较值 1：只比较地址	只读
BCN	27：24	所实现的数据断点个数 0：未实现数据断点 1-15：数据断点个数	只读
BS[14:0]	14：0	断点标志位：BS[n]代表第 n 个数据断点匹配条件满足了。在 GS232IP 中只有 BS[7：0]有效，BS[14：8]必须全部为 0	可读/可写 0
0	31 23：15	必须写为 0，并且读出为 0	只读

图 4-39 描述了 DBAn 的格式，每个 DBAn 的内容表示一个数据断点比较的虚地址。

图 4-40 描述了 DBMn 的格式，每个 DBMn 的内容表示一个数据断点比较的地址掩码，DBMn 的每一位为“1”表示相应位不需要比较，为“0“表示相应位需要比较。

图 4-41 描述了 DBCn 的格式，表 4-47 具体介绍了 DBCn 中的各个域。

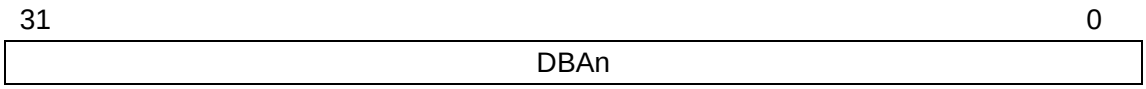


图 4-39 DBAn 寄存器格式

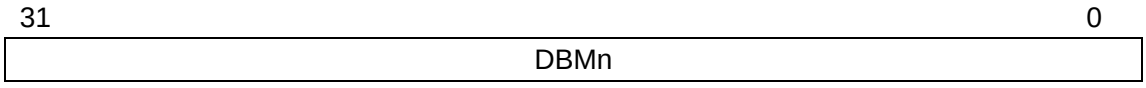


图 4-40 DBMn 寄存器格式

31	24	23	22	21	14	13	12	11	3	2	1	0
0	ASIDuse		0	BA[7:：0]		NoSB	NoLB	0		TE	0	BE

图 4-41 DBCn 寄存器格式

表 4-47 DBCn 域描述

域		描　　述	读写
域名	所在位		
ASIDuse	23	表示数据断点 n 的匹配是否需要比较 ASID： 0：不需要 ASID 比较 1：需要 ASID 比较	可读 可写
BAI[7:0]	21：14	BAI 中每一位对应双字中的一个字节，如果 BAI 中的位被置为 1，则相应的自己访问在匹配时被忽略： 0：对该字节的访问可以引发匹配 1：对该字节的访问在匹配时被忽略，不引发匹配 BAI 不全为 0 是匹配成功的必要条件	可读 可写
NoSB	13	表示 store 操作是否引发匹配： 0：store 操作可以引发匹配 1：store 操作不可以引发匹配	可读 可写
NoLB	12	表示 load 操作是否引发匹配： 0：load 操作可以引发匹配 1：load 操作不可以引发匹配	可读 可写
TE	2	表示数据断点 n 的匹配时是否会设 IBS 中的 BS[n]位： 0：不设 BS[n] 1：设 BS[n]为 1	可读 可写
BE	0	表示数据断点 n 的匹配是否会引发断点例外： 0：不引发断点例外 1：引发断点例外	可读 / 可写

0	31:24 22 3 1	必须写入 0；读出时返回 0。	只读

数据断点的匹配条件是：

DB_match = (((TYPE = = load) && !DBCn_NoLB) | | ((TYPE = =store) && !DBCn_NoSB)) && (!DBCn_ASIDuse | | (ASID = = DBASIDn_ASID))

&& ((DBMn_DBM | ¯ (ADDR ^ DBAn_DBA)) = = ¯0)

&& ((¯DBCn_BAI & BYTELANE) !=0)

需要指出的是：当 BE 位被置为 1 时，如果断点匹配上，即使 TE 位为 0，DBS 中的 BS[n]位依然会被置为 1.因此在进入断点例外后，软件可以通过查询 BS 来判断是那个断点引发了例外。

4.3.4 EJTAG 相关的处理器核扩展

（1）调试模式及其相关例外。调试模式只在发生调试例外的时候进入，执行 DERET 或处理器重置可以退出 Debug 模式。在调试模式下，可以像内核状态下存取处理器资源，除此之外调试模式还提供了额外的资源以方便调试。

调试例外优先级如下表所示：

表 4-48 调试例外优先级表

优先级	例外名称	例外类型
最高	Reset	非调试
	Soft reset	
	Debug Interrupt	调试
	DDBLImpr/DDSBImpr	
	NMI	非调试
	Interrupt	
	Debug Instruction Break	调试
	Address error on instruction fetch	非调试
	TLB refill on instruction fetch	
	TLB Invalid on instruction fetch	
	Cache Error on instruction fetch	
	Bus Error on instruction fetch	

	Debug Breakpoint (SDBBP)	调试
	Other excution-based exceptions	非调试
	Debug Data Break on Load/Store address match	调试
	Address error on data access	非调试
	TLB refill on data access	
	TLB Incvalid on data access	
	TLB modified on data access	
	Cache error on data access	
最低	Bus error on data access	

在调试状态下，只有部分例外可以继续发出，有些例外要被屏蔽，具体情况请见下表：

表 4-49 例外屏蔽表

优先级	调试模式下发生的事件	调试模式处理
最高	Reset	同非调试模式
	Soft reset	
	Debug Interrupt	屏蔽
	DDBLImpr/DDSBImpr	
	NMI	
	Interrupt	
	Debug Instruction Break	
	Address error on instruction fetch	重新进入调试模式
	TLB refill on instruction fetch	
	TLB Invalid on instruction fetch	
	Cache Error on instruction fetch	
	Bus Error on instruction fetch	
	Debug Breakpoint (SDBBP)	如同执行 BREAK 重新进入调试模式
	Other excution-based exceptions	重新进入调试模

		式
	Debug Data Break on Load/Store address match	屏蔽
最低	Address error on data access	重新进入调试模式
	TLB refill on data access	
	TLB Incvalid on data access	
	TLB modified on data access	
	Cache error on data access	
	Bus error on data access	

（2）EJTAG 相关指令扩展。为了支持 EJTAG 调试功能，处理器增加了两条指令，SDBBP 和 DERET。SDBBP 产生一个调试断点异常；DERET 的功能是从调试例外中返回。

（3）内存映射的与调试有关的段。由于调试的 dseg 段分为 dmseg（EJTAG 内存）段和 dresg（EJTAG 寄存器）段。下表说明了段的划分和相关情况：

表 4-50 Dseg 划分

段名	字段名称	虚拟地址	Cache 属性
Dseg	dmseg	0xffff_ffff_ff20_0000 – 0xffff_ffff_ff2f_ffff	Uncached
	drseg	0xffff_ffff_ff30_0000 – 0xffff_ffff_ff3f_ffff	

下表列出了在调试模式下访问 0xffff_ffff_ff20_0000 到 0xffff_ffff_ff2f_ffff 的情况：

表 4-51 Dmseg 的访问情况

Debug_NoDCR	交易类型	DCR_ProbEn	Debug_LSNM	存取
1	x	不存在	0	Kernel 地址空间
0	Fetch	1	X	Dmseg
		0	X	
	Load/Store	1	0	Dmseg
			1	Kernel 地址空间

		0	0	Kernel 地址空间
			1	

下表列出了在调试模式下访问 0xffff_ffff_ff30_0000 到 0xffff_ffff_ff3f_ffff 的情况：

表 4-52 Drseg 的访问情况

Debug_NoDCR	交易类型	Debug_LSNM	存取
1	X	0	Kernel 模式的地址空间
0	Fetch	x	未定义
	Load/Store	0	Drseg
		1	Kernel 模式的地址空间

另外下表列出了不同情况下的调试例外中断入口地址：

表 4-53 调试例外中断入口地址

ECR 寄存器中的 ProbTrap 位	调试例外中断入口地址
0	0xffff_ffff_bfc0_0480
1	0xffff_ffff_ff20_0200(dmseg)

4.3.5 TAP 接口

（1）EJTAG TAP 包含的主要功能包括以下几个方面。①进行设备的识别和操作 EJTAG 调试功能；②dmseg 内存段的模拟；③ 重启后立即进入重启的处理，从处理器外部取得启动代码和数据；④ 从 Probe 得到调试中断请求。下图示表示了 TAP 包含的主要部分。

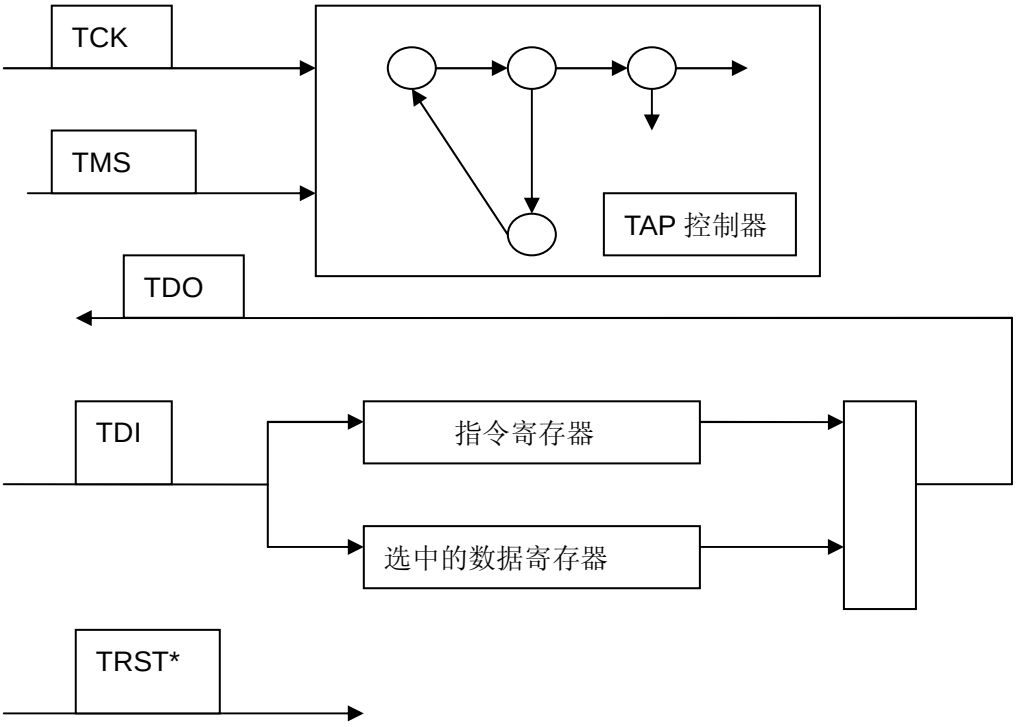


图 4-42 TAP 主要部分

TAP 由测试时钟（TCK）、测试模式选择（TMS）、测试数据输入（TDI）、测试数据输出（TDO）和测试重置（TRST*,低有效）组成。TCK 和 TMS 用于控制 TAP 控制器的状态，从而可以存取指令或者数据寄存器；通过 TDI 和 TDO 可以存取数据和指令寄存器；TRST*用于提供异步重置信号。关于 TAP 的详细信息可以参考 IEEE Std 1149.1—1990 TAP 和边界扫描架构标准，EJTAG 的 TAP 控制器部分与该标准保持兼容。

（2）指令寄存器及其指令。指令寄存器用于控制数据寄存器的选择和 EJTAGBOOT 标志的设置和清除。指令寄存器长度为 5 位，下表包含了 EJTAG 覆盖的指令：

表 4-54 EJTAG 指令

编码	指令	功能
0x01	IDCODE	选中设备 ID 器
0x03	IMPCODE	选中实现标志寄存器
0x08	ADDRESS	选中地址寄存器
0x09	DATA	选中数据寄存器
0x0a	CONTROL	选中 EJTAG 控制寄存器
0x0b	ALL	选中地址、数据、和控制寄存器

0x0c	EJTAGBOOT	使处理器 reset 后执行调试例外
0x0d	NORMALBOOT	使处理器 reset 后执行 reset 处理代码
0x0e	FASTDATA	选择数据和 fastdata 寄存器
0x14	PCSAMPLE	选中 PC 采样寄存器
全 1	BYPASS	选中旁路寄存器

其中指令 IDCODE、IMPCODE、ADDRESS、DATA、CONTROL 和 BYPASS 选中单个数据寄存器，如上表所示。ALL、EJTAGBOOT、NORMALBOOT 和 FASTDATA 指令稍微特殊一些，我们在下面进行描述。

ALL 指令

使用 ALL 指令 Address、Data 和 EJTAG 控制寄存器一次全部选中，如下图所示：

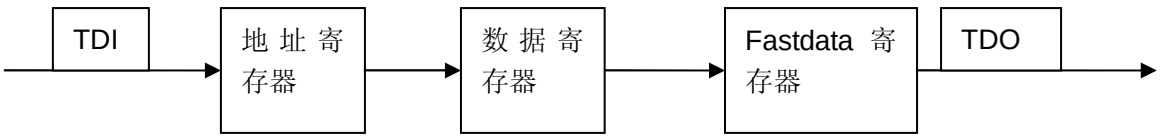


图 4-43 ALL 指令示意图

EJTAGBOOT 和 NORMALBOOT 指令

EJTAGBOOT 和 NORMALBOOT 指令用于控制处理器 reset 时，是否发生调试中断。如果 EJTAGBOOT 标志位存在，reset 时发生调试中断引发调试中断异常，处理器从调试例外异常处取值。同时，控制寄存中的 ProbTrap 位控制了调试例外入口的位置，可以通过 Probe 访问 dmseg 段，这样即使在正常的内存系统不能工作的情况下，也能启动处理器。

给出 EJTAGBOOT 或者 NORMALBOOT 指令的时候选中 Bypass 寄存器。EJTAG 控制寄存器的 EjtagBrk、ProbEn、ProbTrap 位跟随内部的 EJTAGBOOT 位。

FASTDATA 指令

FASTDATA 指令选中数据和 Fastdata 寄存器，如下图所示：

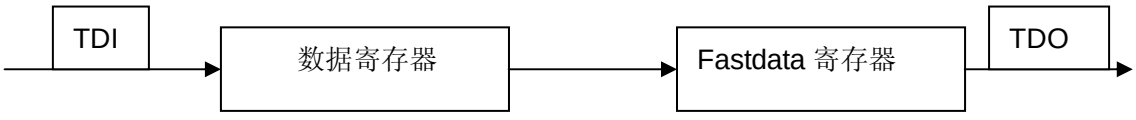


图 4-44 Fastdata 指令示意图

（3）数据寄存器。下表总结了 TAP 的数据寄存器。

表 4-55 TAP 数据寄存器

使用该寄存器的指令	寄存器名称	功能
IDCODE	设备 ID 寄存器	验证设备
MPCODE	实现寄存器	验证可以通过 TAP 访问的功能
DATA/ALL/FASTDATA	数据寄存器	处理器存取的数据
ADDRESS/ALL	地址寄存器	处理器存取的地址
CONTROL/ALL	EJTAG 控制寄存器	通过 TAP 访问的功能的控制寄存器
BYPASS1/EJTAGBOOT/NORMALBOOT	BYPASS 寄存器	提供 TAP 的一位的移位路径
FASTDATA	FASTDATA 寄存器	附着在前面的数据寄存器上值上表示下一次 PrAcc 位的值
PCSAMPLE	PC 采样寄存器	由 PC 采样逻辑使用

下面对数据寄存器值的含义做详细的说明：

设备识别寄存器（IDCODE）

设备识别寄存器是一个 32 位的只读寄存器，用来指明实现 EJTAG 的设备。这个寄存器也在 IEEE 1149. 1 标准中定义了。寄存器的格式和含义如下图和下表所示：

31 28	27 12	11 1	0
版本	部件编号	厂商 ID	1

图 4-45 IDCODE 寄存器格式

表 4-56 IDCODE 寄存器说明

域名	所在位	描述	读/写
版本	31：28		

部件编号	27：12	指明指定设备的部件编号	只读
生产产商	11：1	指明指定设备的生产厂商编号	只读
1	0		只读

- 实现寄存器（IMPCODE）

实现寄存器是一个 32 位的只读寄存器，用来指明 EJTAG 实现了的功能。寄存器值的格式和含义如下图和下表所示：

31 29 28 27 25 24 23 22 21 20 17 16 15 14 13 1 0

EJTAGver	R4k/R3k	0	DINTsup	0	ASIDsize	0	MIPS16e	0	NoDMA	0	MIPS 32/64

图 4-46 IMPCADE 寄存器示意图

表 4-57 IMPCODE 寄存器说明

域名	所在位	描述	读/写
EJTAGevr	31：29		
4k/R3k	28	特权环境的版本，0 为 R4k 特权环境	只读
DINTsup	24	指明是否支持来自 probe 的 DINT 0：不支持 1：支持	只读
ASIDsize	22：21	指明 ASID 的大小 0：为实现 1：6 位 2：8 位	只读
MIPS16e	16	指明处理器是否支持 MIPS16e ASE 0：不支持 1：支持	只读
NoDMA	14	指明没有 EJTAG DMA 支持： 0：不支持 1：支持	只读
MIPS 32/64	0	指明是 32 位还是 64 位处理器： 0：32 位 1：64 位	只读

0	27：25 23 20：17 15 13：1	写忽略；读返回零	只读

- 数据寄存器（DATA）

可以读写的数据寄存器在处理器存取的过程用于操作码和数据的传输。读取该寄存器的时候，只有当处理器阻塞在写操作上的时候，数据寄存器里面的值才有意义。当处理器有阻塞的读操作的时候，写入数据寄存器里面的值才有效。

寄存器值的格式如下图所示：

31

0

数据寄存器

图 4-47 数据寄存器格式

数据寄存器里面的数据有效位由存取的数据大小、地址等共同决定。也就是由地址和 ECR 寄存器中的 Psz 位等决定，Psz 位的含义由下表决定：

表 4-58 Psz 位的含义

Psz	大小	地址	小尾端							
		【2：0】	7	6	5	4	3	2	1	0
0	字节	000								■
		001							■	
		010						■		
		011					■			
		100				■				
0	字节	101			■					
		110		■						
		111	■							
1	半字	000							■	■
		010					■	■		
		100			■	■				
		110	■	■						

2	字	000					■	■	■	■
	5 字节	001				■	■	■	■	■
	6 自节	010			■	■	■	■	■	■
	7 字节	011		■	■	■	■	■	■	■
	字	100	■	■	■	■				
	5 字节	101	■	■	■	■	■			
	6 自节	110	■	■	■	■	■	■		
	7 字节	111	■	■	■	■	■	■	■	
3	3 字节	000						■	■	■
		010					■	■	■	
		100		■	■	■				
		110	■	■	■					
	双字	111	■	■	■	■	■	■	■	■
保留			n. a.							

- 地址寄存器（ADDRESS）

只读的地址寄存器提供处理器访问的地址，寄存器的长度为 48 位。

寄存器的格式如下图所示：

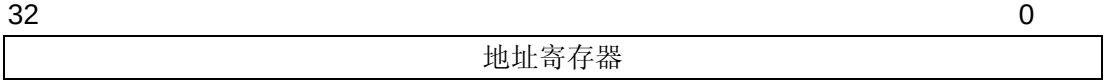


图 4-48 地址寄存器格式

- EJTAG 控制寄存器（ECR）

32 位的 EJTAG 控制寄存器可以用来处理处理器 reset 和 soft reset 信号、Debug 模式信号、处理器存取的开始、完成、大小，以及读写信号。除此之外 ECR 还控制：调试入口地址、允许调试中断请求、允许实现无关的处理器和外设 reset.

寄存器值的格式和含义如下图和下表所示：

Rocc	Psz	0	VPED	Doze	Halt	PerRst	PRnW	PrAcc	0	PrRst	ProbEn	ProbTrap	0	EjtagBrk	0	DM	0

图 4-49 ECR 格式

表 4-59 ECR 域描述

域名	所在位	描述	读/写
Rocc	31	当处理器在重置后，Rocc 位保持为 1.调试主机通过 TAP 清掉此位，表示检测到处理器重置行为。	读/可写 0
Psz	30：29	指明了当前处理器处理数据的大小	只读
VPED	23	在具有 MIPS MT ASE 的处理器上，该位表示 VPE 当前是否被禁用。 0：不支持 MT 或者支持，并且处于激活状态 1：禁用状态	只读
Doze	22	0：未处于低电压模式 1：处于低电压模式	只读
Halt	21	0：内部系统总线运行中 1：内部系统总线停止	只读
PerRst	20	用于控制外围 reset 0：无外围 reser 1：存在外围 reset	可读写
PRnW	19	表示当前阻塞的处理器存取类型。 0：处理器读，取值或者取数 1：处理器写	只读
PrAcc	18	用于只是是否有一阻塞的处理器存取，并用来控制处理器存取的完成。读的时候返回 0 表示没有等待的处理器存取；1 表示存在等待的处理器存取。如果在有等待的处理器存取的情况下，写入 0 表示处理器存取的完成。	可读/可写
PrRst	16	用于控制处理器 reset，该位不可用，只读	只读
ProbEn	15	用于控制 Probe 是否用来服务 dmseg 段的存取。 0：Probe 不服务 1：Probe 服务	可读写
ProbTrap	14	用于控制调试状态下的例外地址。 0：正常 0xffff_ffff_bfc0_0408	可读写

		1：dmseg 0xffff_ffff_ff20_0000	
EjtagBrk	12	写入 1 的时候请求一个调试中断例外，当处理器进入调试状态的时候硬件清掉该位	可读/可写 1
DM	3	指示当前处理器是否在调试模式。 0：非调试模式 1：调试模式	只读
0	28：24， 17， 13， 11：4， 2：0	返回 0	只读

- Fastdata 寄存器

Fastdata 寄存器的宽度为 1 位。移进的 Fastdata 寄存器的值表示了存取操作是否已经完成。Fastdata 寄存器只有 1 位，可读可写。

表 4-60 Sample 寄存器说明

域名	所在位	描述	读/写
TC	48：41		
ASID	40：33	采样的 PC 的地址空间	只读
PC	32：1	PC 值	只读
New	0	表示该寄存器是否被读过	可读/写 0

- Bypass 寄存器

Bypass 寄存器提供一位的只读寄存器，提供通过 TAP 访问的最短移位路径。该寄存器是 IEEE 1149. 1 的规定。

5 DDR2

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

5.1 DDR2 SDRAM 控制器特性

龙芯 1B 通过一个片选信号和 18 位的地址总线（15 位行/列地址和 3 位逻辑 Bank 地址）实现最大地址空间是 32G(2^{35})。

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

CPU 内存的物理地址能被转换位行/列地址，见表 5-1。例如，1 个 CS_n 信号，8 个 banks，12 位行地址和 12 位列地址。

表 5-1 DDR2 SDRAM 行/列地址转换

34 ... 30	29 ... 18	17 ... 15	14 ... 3	2 ... 0
	RAS_n	BANK_n	CAS_n	Byte

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

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

- 全流水的命令和数据读写；
- 通过合并和重排序增加带宽；
- 通过丰富的寄存器读写端口修改基本的参数；
- 内置 Delay Compensation Circuit(DCC)，用来可靠的发送/接收数据；
- 1 位和 2 位错误检测，通过 ECC 进行 1 位的错误修正;
- 频率：133MHz-333MHz;

5.2 DDR2 SDRAM 读协议

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

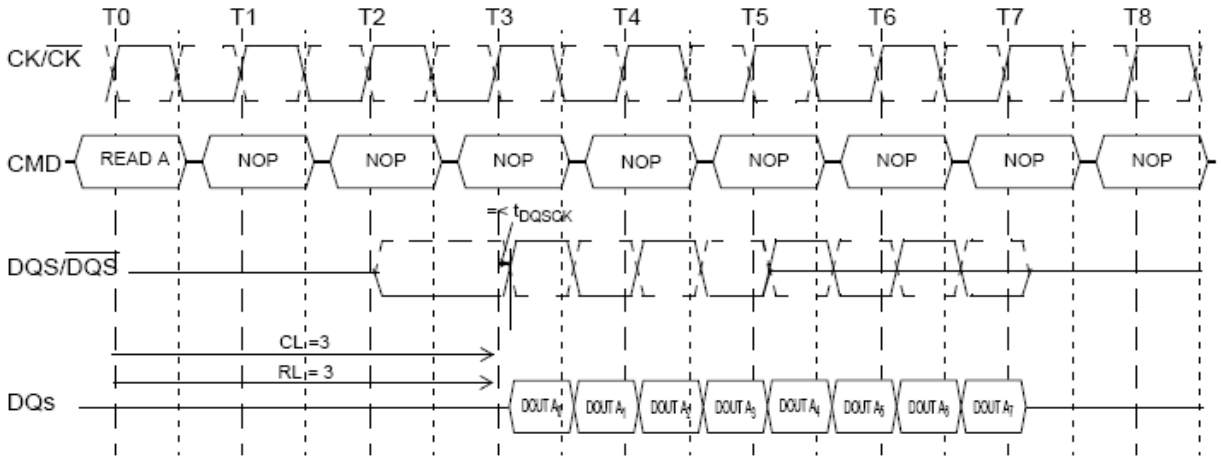


图 5-1 DDR2 SDRAM 读协议， Cas Latency = 3, Read Latency = 3, Burst Length = 8

5.3 DDR2 SDRAM 写协议

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

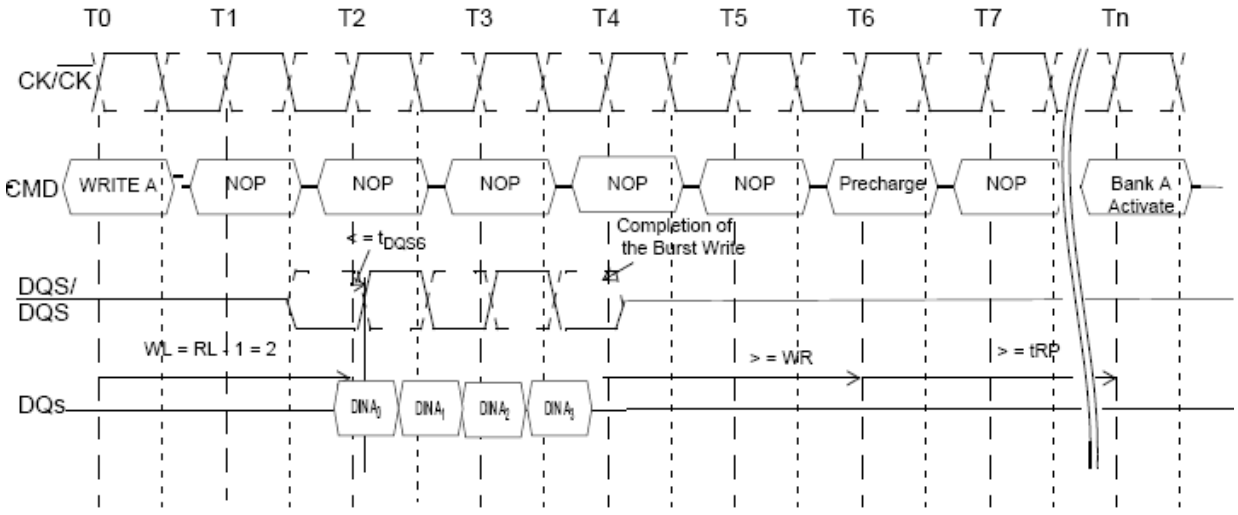


图 5-2 DDR2 SDRAM 写协议，Cas Latency = 3, Write Latency = Read Latency -1 = 2, Burst Length = 4.

5.4 DDR2 SDRAM 参数设置顺序

为了在系统中支持不同的 DDR2 SDRAM 颗粒，DDR2 SDRAM 需要在加电复位后配置。JESD79-2B 标准定义了详细的配置操作和过程。DDR2 在内存初始化前是不可用的，内存初始化顺序为：

1. 系统复位期间，aresetn 信号设置为 0，所有的寄存器内容将清除为却省值；
2. 系统复位释放，aresetn 信号设置为 1；
3. 向配置寄存器地址发 32 位写指令，配置所有 29 个配置寄存器。此时如果写 CTRL_03，应将其中参数 START 设为 0。所有寄存器都必须正确配置才可以正常工作。
4. 向配置寄存器 CTRL_03 中发 32 位写指令。此时应将参数 START 设为 1。结束后内存控制器将自动对内存发起初始化指令。

在系统母板初始化后，DDR2 SDRAM 控制器在内存使用前需要配置内存类型。特别的是需要将相应的配置参数写到对应于物理地址 0X0FFF FE00 的 29 个 64 位寄存器中。每个寄存器会包括一个、多个或部分的参数。

5.5 DDR2 SDRAM 采样模式配置

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

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

6 LCD

本章给出 1B 芯片内 LCD 控制器（Display Controller）的详细描述和配置。Display Controller 作为一个整体的模块，该模块读取指针数据和图像数据，通过对这些数据进行格式转换、颜色抖动、gamma 调整等步骤产生最终的数据输出，同时为两个显示处理单元产生同步信号和数据使能信号，最后将最终处理后的图像数据和同步信号发往显示接口。

6.1 特性

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

6.2 模块示意图

本模块示意图 6-1 显示了两个主要的块单元，分别对应 Display Controller 中的两个时钟域：

a） 核心时钟域，用于和内存交互。

b）显示单元时钟域，用于显示单元。

颜色抖动查找表含有 4x4 个条目，每个条目 4bits。它通过每个显示单元的 x[1:0] 和 y[1:0]索引。

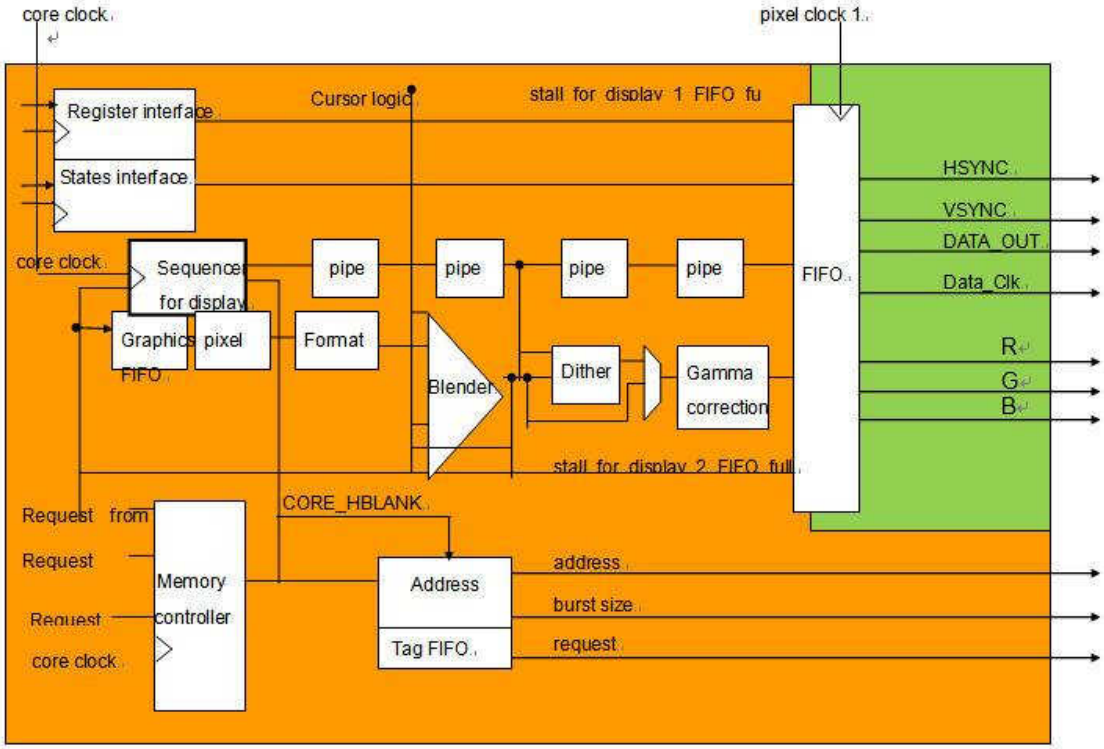


图 6-1 模块示意图

6.3 VTG 定序器

Video timing generation (VTG) 定序器包含两个运行在像素时钟域下的计数器，用这两个计数器 来产生 HSYNC，VSYNC，HBLANK,VBLANK 信号，如下图所示。

图 6-2 时序图

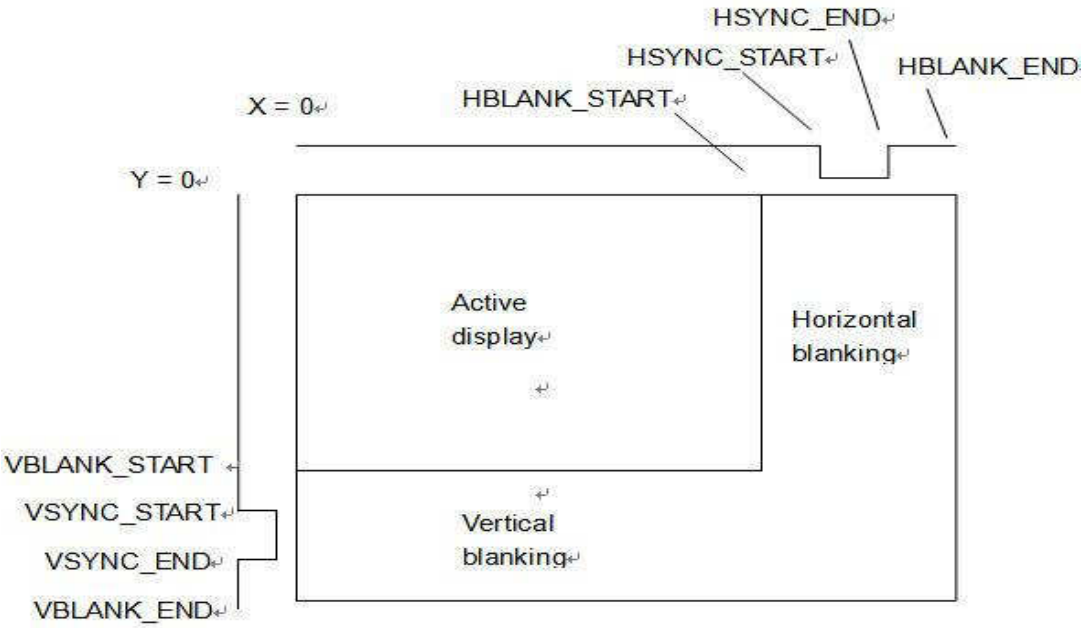


图 6-2 时序图

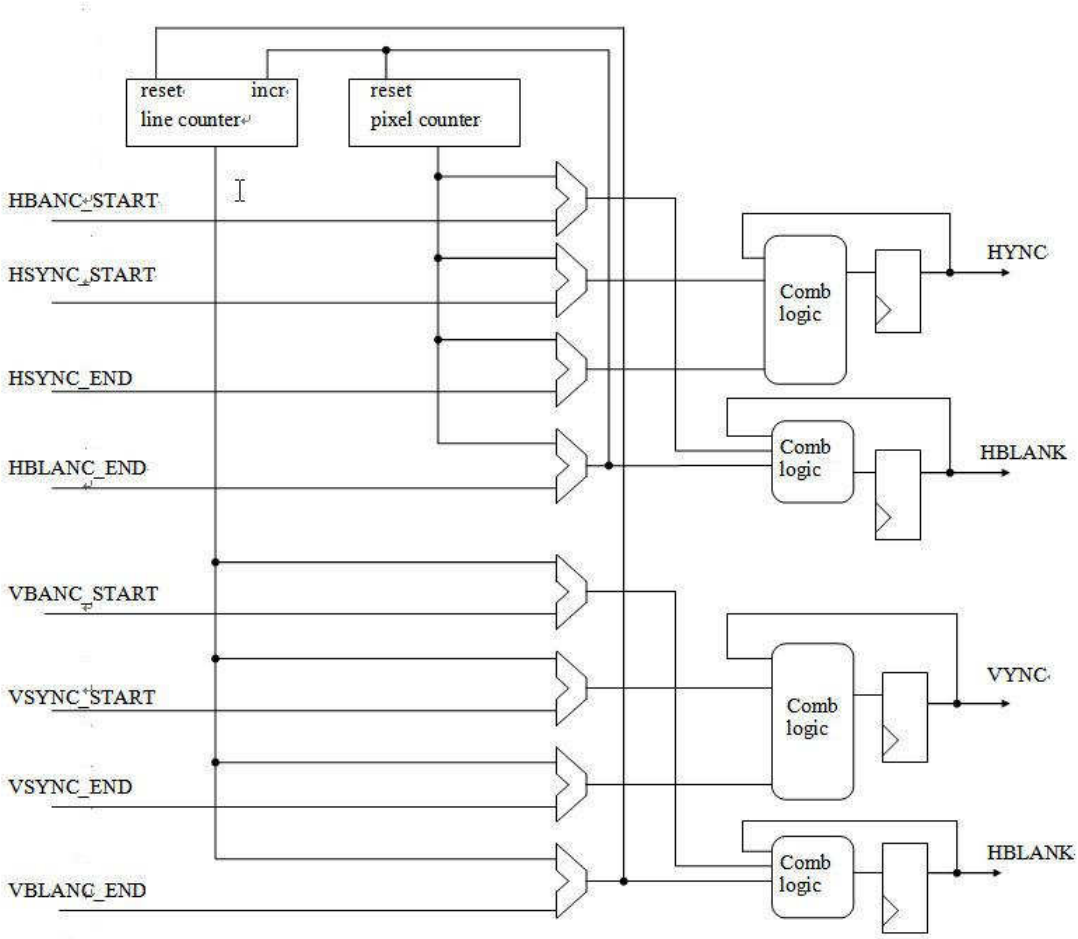


图 6-3 定序逻辑

6.4 合并模块

合并模块将负责合并主窗口（即一号或二号显示单元）像素和指针窗口像素。合并通过一个线性插值器完成。该插值器通过指针的 alpha 参数完成图像数据和指针数据的插值合并。如果没有指针数据提供，则插值器被绕过。

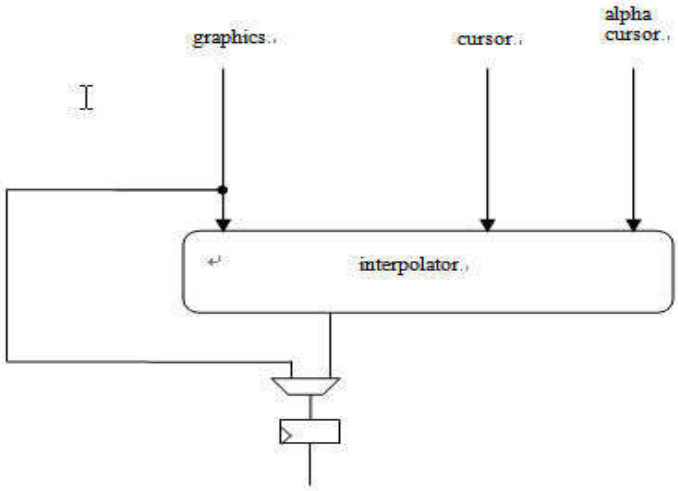


图 6-4 合并模块

6.5 格式转换

存储在内存中的数据可以有以下的格式：R4G4B4R5G5B5
R5G6B5
R8G8B8
Frame buffer 的数据需要被转换来适应成显示器需要的格式。
当把一个低位数转换成一个高位数时，相应所缺位数的 MSBs 将通过复制 LSBs 得到。

6.6 颜色抖动

颜色抖动功能用于根据一定规则来增强像素值。
在下面的这个例子中，将解释颜色抖动功能的实现步骤。
首先，确定是对哪一数据位进行增强（即该数据位加 1），为此需要配置寄存器 Display Dither Configuration。比如配置 RedSize 为 6（1 到 8 之间，包含 1 和 8），则表明对从 MSB 位数起第 6 位进行增强，即 RedColor[7:0]的 RedColor[2]位增强。GreenSize 和 BlueSize 同理。
其次，需要建立查找表，即配置寄存器 Display Dither Table。
查找表包含 16 个条目，即 16 个阈值，每个条目 4 位宽。
查找表通过屏幕像素计数器的横坐标 x 的最低两位 x[1:0]和纵坐标 y 的最低两位 y[1:0]进行索引，得到一个阈值 U[3:0]。
对应屏幕（x，y）位置的像素值的最低四位被拿来跟查到的这个阈值比较。
如果 RedColor[3:0] > U[3:0]，并且 RedColor[7:2]不是 6'b111111，则 RedColor[2]位加 1，实现颜色增强。

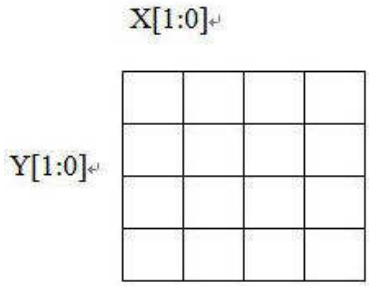


图 6-5 Dither 查找表

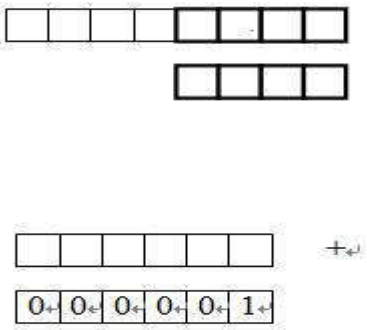


图 6-6 Dither 位操作

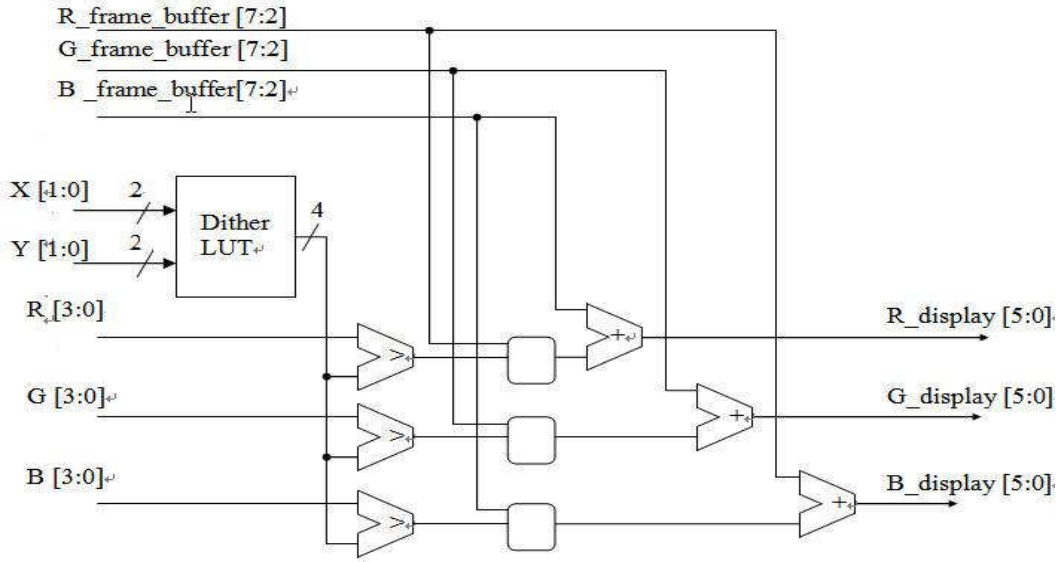


图 6-7 Dither 逻辑

6.7 Gamma 调整

Gamma 调整模块包含三个查找表，一个负责红色，一个负责绿色，一个负责蓝色。
查找表可以通过寄存器改写。查找表只可写。
考虑一个 Gamma 调整如下：（原色，调整色）。当设置 Gamma 颜色查找表时，我们应该按照颜色值大小顺序配置寄存器.。
如果我们想要一个调整为 （0，0）（1，2）（2，5）（3，6）……，先对寄存器 Gamma Index 配置 0，表示 gamma 调整从 0 开始，然后对 Gamma Data 寄存器配置 256 次完成整个配置过程：0，2，5，6……

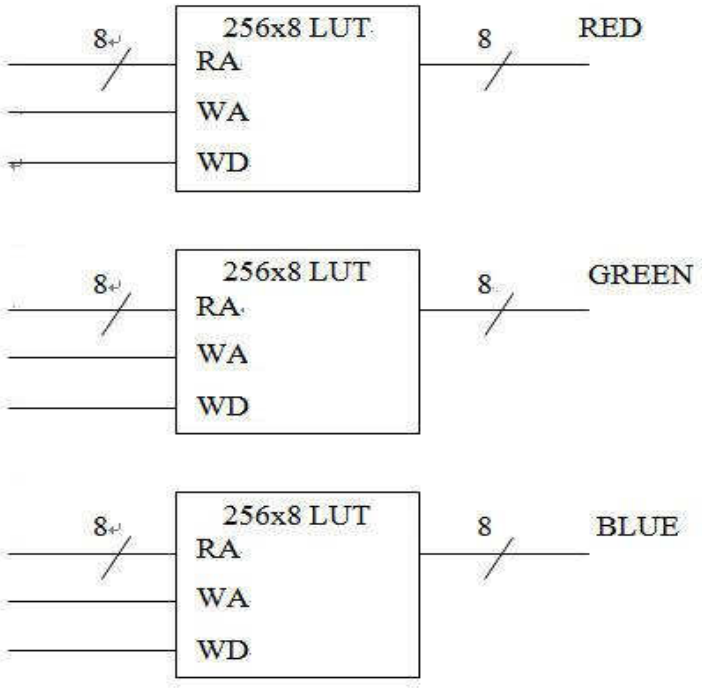


图 6-8 Gamma 调整查找表

6.8 访存

因为两个显示单元可以同时运行，它们有不同的时序和分辨率，所以两个显示单元在同一时刻可能需要不同的数据。Display Controller 必须分别的访存以给两个显示单元提供数据。
每一个屏幕显示包括主窗口和指针窗口。
注意指针窗口只能出现在一个显示单元中，即出现在一号显示单元或二号显示单元中。
因此有访存要求的实体共有三个：一号显示单元主窗口，二号显示单元主窗口，一个指针窗口。
访存仲裁即为这三者仲裁。
当想内存发出数据请求后，数据将会按照请求的顺序从内存收到。为了决定收到的数据属于哪个请求实体，我们对每一个请求附着一个标签并把这个标签放在 FIFO 里。当数据送到时，我们就从 FIFO 中弹出一个标签，标签包含两个域，请求 ID 和 burst 大小。

请求 ID 可以是 0(指针), 1(一号显示单元), 3(二号显示单元)。
Burst 大小可以使 0(8 字节) , 3(32 字节) ,15 (256 字节)。.

6.9 指针

Display Controller 支持硬件指针。当开启硬件指针时，可以下面两种格式：

1 XOR cursor

2 Full color αRGB

在 XOR 指针格式情况下，每个像素点使用 2 位。一位为 mask，一位为 XOR。Mask 位产生指针的形状。XOR 位决定该像素显示。

AND(mask_bit)	XOR(xor_bit)	CURSOR_COLOR
0	0	background color
0	1	foreground color
1	0	NOP
1	1	invert panel color

在αRGB 指针格式下，每个像素包含 8 位α，8位 R，8 位 G，8位B。
Alpha 部分表示插值系数。
指针有两个重要的点：左上点（top-left point）和作用点（hot spot）。
左上点用来作为指针地址的参考点。
作用点用来将鼠标按下动作确定到一个像素上。

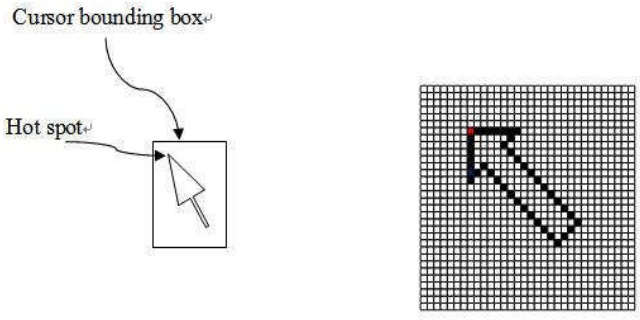


图 6-9 指针

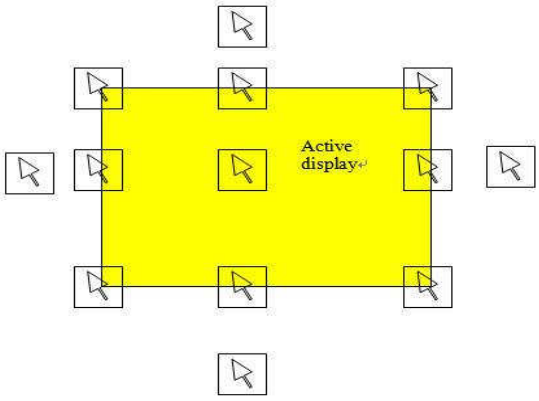


图 6-10 指针在屏幕各种位置示例

6.10 访存地址的生成

访存地址的生成通过核心时钟域的三个计数器完成。这些计数器是：

一号显示单元主窗口计数器

二号显示单元主窗口计数器

指针窗口计数器

因为主窗口和指针窗口都需要在第一行就使用数据，所以第一次的访存请求将发生在显示屏竖直空白区域（vertical blank）的一开始（即图 6-2 中的 VBLANK_START 位置，系统启动后计数器自动位于这一位置）。

在那之后，只要 FIFO 用完一个 burst 的数据量就会再次发起新的访存请求，当取满一帧的数据时，DMA 会暂停工作不再取数。直到计数器指示当前行数又到竖直空白区域（vertical blank）时，DMA 会再次启动发起数据请求为下一帧显示做准备。

显示单元的主窗口的将会产生 x 和 y 的坐标位置。

6.10.1 Bursts

Display Controller 的接口宽度为 64 为，所以根据 AXI 协议 burst size 为 3。

Display Controller 允许的 burst length 大小有：

Burst length	Transfer 个数	相应字节
0	1	8
3	4	32
15	32	256

对于 Burst 为 15 时，按照 AXI 协议，此时请求的 transfer 个数为 16，则相应字节数为 128 字节，但由于特殊的访存优化需要，所以本设计配合外部的上层模块将 15 定义为 32 个 transfer 个数。

6.10.2 数据格式

Display Controller 支持以下数据格式：

图像：

R4G4B4 -> 12 bits per pixel

R5G5B5 -> 15 bits per pixel

R5G6B5 -> 16 bits per pixel
R8G8B8 -> 24 bits per pixel

指针：
MASKED -> 2 bits per pixel
A8R8G8B8 -> 32 bits per pixel

6.10.3 内存组织

Frame buffer 的内存组织是线性的，即像素是顺序存放的。
显示单元的 burst 大小为 256 字节。
指针单元的 burst 大小为：A8R8G8B8 格式下为 32 字节，MASKED 格式下为 8 字节。

6.10.4 跨度

设 M = 一行中的显示像素个数（即寄存器“Hdisplay”中的“DisplayEnd”），
P = 每个像素的大小（即 2.8.2 所示 I）.
跨度 Stride = (M*P/256 + 1)*256。256 代表 DMA 每次 burst 的大小为 256 字节。
之所以对 M*P 除以 256 取整，是因为 M*P（即一行数据的总数据量）可能不能被 256 整除，这样在 DMA 取数时会造成取数错误。

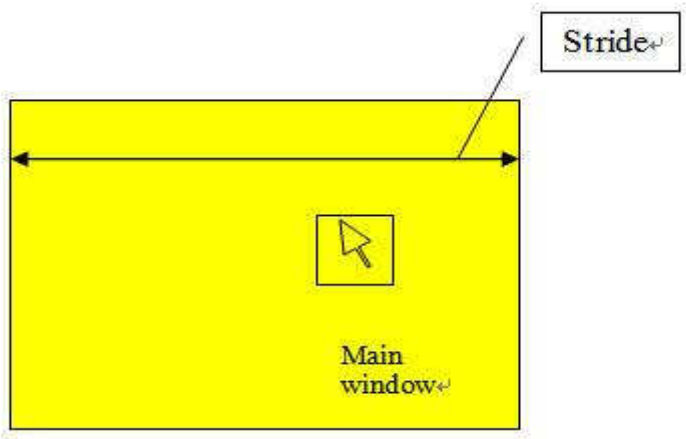


图 6-11 主窗口和指针

6.11 核心时钟域的定序器

定序器产生 x 和 y 的坐标位置，通过 x 和 y 来控制颜色抖动，控制指针以及控制主窗口和指针窗口的数据合并。因为核心时钟域的时钟频率高于像素时钟，所以在核心时钟域和像素时钟域之间有一个异步数据 FIFO，负责连接这两个时钟域。当此异步 FIFO 被填满时，核心时钟域的流水线就会被暂停，不允许数据继续前送。

6.12 输入数据 FIFOs

从内存来的数据被放进输入 FIFOs 中。Display Controller 中有两个输入 FIFOs：

一号主窗口 FIFO
指针窗口 FIFO

6.13 关于 VBLANK 的同步

某些寄存器的值保存在两套寄存器中。一套寄存器保存当前值，一套寄存器保存更改后的新值。新值对旧值的替换通过 VBLANK 同步。在每次 VBLANK 的开始，新值就会取代旧值。.

6.14 AXI 接口

Display Controller 接口符合 AXI 接口标准。
当配置寄存器时，注意寄存器的地址是第 4 位到第 19 位。如 awAddr[63:0]的有效地址位为 awAddr[19:4]，寄存器低 4 位恒为 0。.

6.15 寄存器

寄存器列表：

寄存器名	配置地址
Frame Buffer configuration	{15'h92,5'h0}
Frame Buffer Address_0	{15'h93,5'h0}
Frame Buffer Address_1	{15'hAC,5'h0}
Frame Buffer Stride	{15'h94,5'h0}
Frame Buffer Origin	{15'h95,5'h0}
Display Ditheronfiguration	{15'h9B,5'h0}
Display Dither Table(low)	{15'h9C,5'h0}
Display Dither Table(high)	{15'h9D,5'h0}
Pane Configuration	{15'h9E,5'h0}
Panel Timing	{15'h9F,5'h0}
HDisplay	{15'hA0,5'h0}
Hsync	{15'hA1,5'h0}
VDisplay	{15'hA4,5'h0}
VSync	{15'hA5,5'h0}
Cursor Configuration	{16'h152,4'h0}
Cursor Address	{16'h153,4'h0}
Cursor Location	{16'h154,4'h0}
Cursor Background	{16'h155,4'h0}
Cursor Foreground	{16'h156,4'h0}
Gamma Index	{15'hA7,5'h0}
Gamma Data	{15'hA8,5'h0}

Frame Buffer Configuration	bit	描述	初始值
Reset	20	写 0 reset	0

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

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

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

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

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

Display Dither Configuration	Bit	描述	初始值
Enable	31	置 1 使能 dither 功能	0
RedSize	[19:16]	红色域宽度	4’b0000
GreeenSize	[11:8]	绿色域宽度	4’b0000
BlueSize	[3:0]	蓝色域宽度	4’b0000

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

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

HDisplay	Bit	描述	初始值
Total	[27:16]	显示屏一行的总体像素数（包括非显示	12'b0

		区）	
DisplayEnd	[11:0]	显示屏一行中显示区的像素数	12’b0

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

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

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

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

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

Cursor Location	Bit	描述	初始值
Y	[26:16]	指针的焦点在整个显示区的纵坐标	11’b0
X	[10:0]	指针的焦点在整个显示区的横坐标	11’b0

Cursor Background	Bit	描述	初始值
Red	[23:16]	指针单色模式下背景色的红色域	8'b0
Green	[15:8]	指针单色模式下背景色的绿色域	8'b0
Blue	[7:0]	指针单色模式下背景色的蓝色域	8'b0

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

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

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

7 GMAC0

7.1 DMA 寄存器描述

GMAC 寄存器包括 GMAC 寄存器部分和 DMA 寄存器部分。GMAC0 的 GMAC 寄存器的起始地址是 0XBFE1_0000；GMAC0 的 DMA 寄存器的起始地址是 0XBFE1_1000。下面分别介绍 DMA 寄存器和 GMAC 寄存器的意义。

参数名称	位	缺省值	描述
Register0 (Bus Mode Register) Offset: 0x00			
Reserved 保留	31:27	0x0	保留，只读
MB: Mixed Burst 混合突发访问	26	0x0	当此位为高，FB 位为低时，AXI master 在突发访问长度大于 16 时采用 INCR 访问模式，当突发访问长度为 16 或者小于 16 时采用 FIX 访问模式。用户不用关心此位设置。
AAL:Address-Aligned Beats 地址对齐节拍	25	0x0	当此位和 FB 位同时为高时，AXI 接口的所有访问将对齐到起始地址的 LS 位。如果 FB 位为 0，首次访问地址访问不对齐，剩余的访问地址对齐。用户不用关心此位设置。
8XPBL Mode 是否使能 PBLX8 模式	24	0x0	此位为高时，GMAC DMA 的最大突发数据传输长度为 8,16,32,64,128 或者 256。最大突发长度取决于 PBL。用户不用关心此位设置。
USP:Use Separate PBL 使用分离的 PBL 值	23	0x0	此位为高时，PBL 值只应用于 TxDMA。此位为低时，PBL 值应用于 TxDMA 和 RxDMA。用户不用关心此位设置。
RPBL: RxDMA PBL RxDMA 突发传输长度	22:17	0x01	表示一次 RxDMA 传输的最大突发传输长度。只能为 1,2,4,8,16 和 32，其它值无效。
FB: Fixed Burst 定长突发传输长度使能	16	0x0	指定 AXI Master 接口是否采用 FIX 突发传输模式。用户不用关心此位设置。
PR: Rx:Tx priority ratio RxDMA 与 TxDMA	15:14	0x0	在 DA 位为 0 时起作用。 00：1：1 01：2：1 10：3：1

优先级比例			11：4：1
PBL:Programmable Burst Length 可编程突发传输长度	13:8	0x1	用户不用关心此设置。
ATDS:Alternate Descriptor size 是否使用 32 字节大小描述符	7	0x0	此位为 1 时使用 32 字节大小的描述符 此位为 0 时使用 16 字节大小的描述符
DSL: Descriptor Skip Length 描述符间隔距离	6:2	0x00	设置 2 个描述符间的距离。但此值为 0 时，默认为 DMA 描述符大小。
DA: DMA Arbitration scheme DMA 传输仲裁策略	1	0x0	0:在 RxDMA 和 TxDMA 间采用轮转仲裁机制 1: RxDMA 优先级高于 TxDMA 优先级。具体比值见 PR 值。
SWR:Software Reset 软件复位	0	0x1	此位置高 DMA 控制器将复位 GMAC 内部寄存器和逻辑。当复位结束时该位自动清零。
Register1 (Transmit Poll Demand Register) Offset: 0x04			
TPD: Transmit Poll Demand 传输轮询使能	31:0	0x0	向此值写入任意值，发送 DMA 控制器将会读取寄存器 18 对应的描述符。如果该描述符无效，DMA 传输将会停止。如果该描述符有效，DMA 传输将会继续。
Register2 (Receive Poll Demand Register) Offset: 0x08			
RPD: Receive Poll Demand 接收轮询使能	31:0	0x0	向此值写入任意值，接收 DMA 控制器将会读取寄存器 18 对应的描述符。如果该描述符无效，DMA 传输将会停止。如果该描述符有效，DMA 传输将会继续。
Register3 (Receive Descriptor List Address Register) Offset: 0x0C			
Start of Receive List 接收描述符起始地址	31:0	0x0	指向接收描述符首地址。

Register4 (Transmit Descriptor List Address Register) Offset: 0x10			
Start of Transmit List 发送描述符起始地址	31:0	0x0	指向发送描述符首地址
Register5 (Status Register) Offset: 0x14			
Reserved	31:30		保留，只读
TTI: Time-Stamp Trigger Interrupt 时间戳触发中断	29	0x0	时间戳模块触发中断。只读。
GPI:GMAC PMT Interrutp 电源管理模块触发中断	28	0x0	电源管理模块触发中断。只读。
GMI:GMAC MMC Interrupt MMC 模块触发中断	27	0x0	MMC 模块触发中断。只读。
GLI:GMAC Line interface Interrupt GMAC 模块线路触发中断	26	0x0	GMAC 模块的 PCS 或者 RGMII 模块触发中断。只读。
EB: Error Bits 错误位	25:23	0x0	23：1'b1 TxDMA 数据传输过程中发生错误 1'b0 RxDMA 数据传输过程中发生错误 24：1'b1 读传输错误 1'b0 写传输错误 25：1'b1 描述符访问错误 1'b0 数据缓存访问错误
TS:Transmit Process State 传输过程状态	22:20	0x0	3'b000:传输停止；复位或者停止命令发送 3'b001:正在进行；获取传输描述符 3'b010:正在进行；等待传输状态 3'b011:正在进行；从发送缓存读取数据并发送到传输 FIFO(TxFIFO) 3'b100:写入时间戳状态 3'b101:保留

			3’b110:挂起；传输描述符不可用或者传输缓存下溢。 3’b111:运行；关闭传输描述符。
RS:Receive Process State 接收过程状态	19:17	0x0	3’b000:停止；复位或者接收到停止命令 3’b001:运行；获取接收描述符。 3’b010:保留； 3’b011:运行；等待接收包。 3’b100:暂停；接收描述符不可用。 3’b101:运行；关闭接收描述符。 3’b110:时间戳写状态。 3’b111:运行；将包内容从接收缓存传输到系统内存。
NIS:Normal Interrutp Summary 正常中断汇总	16	0x0	提示系统是否存在正常中断。
AIS:Abnormal Interrutp Summary 异常中断汇总	15	0x0	提示系统是否存在异常中断。
ERI:Early Receive Interrutp 提前接收中断	14	0x0	提示 DMA 控制器已经把包的第一个数据写入接收缓存
FBI:Fatal Bus Error Interrutp 总线错误中断	13	0x0	提示总线错误，具体信息见[25:23]。当此位设置后 DMA 引擎停止总线访问操作。
Reserved	12:11	0x0	保留
ETI:Early Transmit Interrupt 提前发送中断	10	0x0	提示需要传输的以太网帧已经完全传输到 MTL 模块中的传输 FIFO
RWT:Receive Watchdog Timeout 接收看门狗超时	9	0x0	提示接收到一个大小超过 2048 字节的以太网帧。（当巨帧使能时，提示接收到大小超过 10240 字节的以太网帧）
RPS:Receive Process Stopped 接收过程停止	8	0x0	指示接收过程停止
RU:Receive Buffer Unavailable	7	0x0	指示接收缓存不可用

接收缓存不可用			
RI:Receive Interrupt 接收中断	6	0x0	指示帧接收完成。帧接收的状态信息已经写入接收描述符。接收处于运行状态。
UNF:Transmit Underflow 传输缓存下溢	5	0x0	指示帧发送过程中产生接收缓存下溢。
OVF:Receive Overflow 接收缓存上溢	4	0x0	指示帧接收过程中接收缓存上溢。
TJT:Transmit Jabber Timeout	3	0x0	
TU:Transmit Buffer Unavailable 传输缓存不可用	2	0x0	提示传输列表中的下一个描述符不能被 DMA 控制器访问。
TPS:Transmit Process Stopped 传输过程停止	1	0x0	提示传输过程停止
TI:Transmit Interrutp 传输完成中断	0	0x0	提示帧传输完成并且第一个描述符的 31 位置位。
Register6 (Operation Mode Register) Offset: 0x18			
Reserved 保留	31:27	0x0	保留
DT 关闭丢弃 TCP/IP Checksum 错误以太网帧的功能	26	0x0	此位为 1 时 GMAC 将不丢弃 checksum 错误的以太网帧。
RSF:Receive Store and Forward 接收存储转发	25	0x0	此位为 1 时 MTL 模块只接收已经全部存储在接收 FIFO 中的以太网帧。
DFF:Disable Flushing of Received Frames	24	0x0	此位为 1 时，接收 DMA 在接收描述符或者接收缓存不可用时不冲刷任何以太网帧。

关闭冲刷接收的以太网帧的功能			
RFA[2]:MSB of Threshold for Activating Flow Control 激活流控阈值	23	0x0	100：最大值减去 5KB 101：最大值减去 6KB 110：最大值减去 7KB 111：保留 （注：最大值为 8KB）
RFD[2]:MSB of Threshold for Deactivating Flow Control 关闭流控阈值	22	0x0	100：最大值减去 5KB 101：最大值减去 6KB 110：最大值减去 7KB 111：保留 （注：最大值为 8KB）
TSF:Transmit Store and Forward 发送存储转发	21	0x0	此位为 1 时，帧的发送只在帧的内容已经全部进入 MTL 的传输 FIFO 中。
FTF:Flush Transmit FIFO 冲刷传输 FIFO	20	0x0	此位为 1 时，传输控制逻辑复位为默认值，并且会导致发送 FIFO 里面的数据全部丢失。
Reserved	19:17	0x0	保留
TTC:Transmit Threshold Control 传输阈值控制	16:14	0x0	当帧大小超过此值时 MTL 将会传输该帧。 000: 64 字节 001: 128 字节 010: 192 字节 011: 256 字节 100: 40 字节 101: 32 字节 110: 24 字节 111: 16 字节
ST:Start/Stop Transmission Command 开始/停止传输命令	13	0x0	此位为 1，传输进入运行状态。 此位为 0，传输进入停止状态。
RFD:Threshold for	12:11	0x0	00: 最大值减去 1KB

deactivating flow control 关闭流控阈值			01: 最大值减去 2KB 10: 最大值减去 3KB 11: 最大值减去 4KB （最大值为 8KB）
RFA:Threshold for Activating flow control 激活流控阈值	10:9	0x0	00: 最大值减去 1KB 01: 最大值减去 2KB 10: 最大值减去 3KB 11: 最大值减去 4KB （最大值为 8KB）
EFC:Enable HW flow control 使能硬件流控	8	0x0	此位为 1 时，基于接收 FIFO 利用率的硬件流控电路生效。
FEF:Forward Error Frames 传输错误帧	7	0x0	此位为 1 时，接收错误帧（错误帧包括：CRC 错误，冲突错误，巨帧，看门狗超时，溢出等）
FUF:Forward Undersized Good Frames 接收无错误的小帧	6	0x0	此位为 1 时，接收 FIFO 将会接收没有错误但小于 64 字节的以太网帧。
Reserved	5	0x0	保留
RTC:Receive Threshold Control 接收阈值控制	4:3	0x0	MTL 传输接收 FIFO 中帧内容已经超过此项设置大小。 00: 64 字节 01:32 字节 10: 96 字节 11: 128 字节
OSF:Operate on Second Frame 是否操作第二个以太网帧	2	0x0	此位为高时，DMA 在第一个以太网帧的状态尚未写回时即可以开始处理第二个以太网帧。
SR:Start/Stop Receive 开始/停止接收	1	0x0	此位设置为高时，接收进入运行状态。 此位设置为低时，接收进入停止状态。
Reserved	0	0x0	保留

保留			
Register7 (Interrupt Enable Register) Offset: 0x1C			
Reserved 保留	31:17	0x0	保留
NIE:Normal Interrupt Summary Enable 正常中断汇总使能	16	0x0	此位为 1 时：正常中断使能 此位为 0 时：正常中断不使能
AIE:Abnormal Interrupt Summary Enable 非正常中断汇总使能	15	0x0	此位为 1 时：非正常中断使能。 此位为 0 时：非正常中断不使能。
ERE：Early Receive Interrupt Enable 早期接收中断使能	14	0x0	此位为高时：早期接收中断使能
FBE:Fatal Bus Error Enable 总线致命错误中断使能	13	0x0	此位为高时：总线致命错误中断使能。
Reserved 保留	12:11	0x0	保留
ETE:Early Transmit Interrupt Enable 早期传输中断使能	10	0x0	此位为高时：使能早期传输中断
RWE:Receive Watchdog Timeout Enable 接收看门狗超时中断使能	9	0x0	此位为高时：使能接收看门狗超时中断
RSE:Receive Stopped Enable 接收停止中断使能	8	0x0	此位为高时：使能接收停止中断。
RUE:Receive Buffer Unavailable	7	0x0	此位为高时：使能接收缓冲区不可用中断。

Enable 接收缓冲区不可用中断使能			
RIE:Receive Interrupt Enable 接收中断使能	6	0x0	此位为高时：使能接收完成中断
UNE:Underflow Interrupt Enable 传输 FIFO 下溢中断使能	5	0x0	此位为高时：使能传输 FIFO 下溢中断
OVE:Overflow Interrupt Enable 接收 FIFO 上溢中断使能	4	0x0	此位为高时：使能接收 FIFO 上溢中断。
TJE:Transmit Jabber Timeout Enable 传输 Jabber 超时中断使能	3	0x0	此位为高时：使能 Jabber 超时中断。
TUE:Transmit Buffer Unavailable Enable 传输缓存不可用中断使能	2	0x0	此位为高时：使能传输缓存不可用中断。
TSE:Transmit Stopped Enable 传输停止中断使能	1	0x0	此位为高时：使能传输停止中断。
TIE:Transmit Interrupt Enable 传输完成中断使能	0	0x0	此位为高时：使能传输完成中断。
Register8 (Missed Frame and Buffer Overflow Counter Register)　Offset: 0x20			
Reserved 保留	31:29	0x0	保留
Overflow bit for FIFO Overflow	28	0x0	FIFO 溢出指示位

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

Register20 (Current Host Transmit Buffer Address Register) Offset: 0x50			
Host Transmit Buffer Address Pointer 当前传输缓冲区主机地址指针	31:0	0x0	只读
Register21 (Current Host Receive Buffer Address Register) Offset: 0x54			
Host Receive Buffer Address Pointer 当前接收缓冲区主机地址指针	31:0	0x0	只读

7.2 GMAC 控制器寄存器描述

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

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

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

Filtering 源地址反转过滤			将会标记为源地址匹配失败。 此位为 0 时，和 SA 寄存器中源地址不匹配的以太网帧将会标记为源地址匹配失败。
PCF: Pass Control Frames 接收控制帧	7:6	0x0	00: GMAC 过滤所有控制帧 01: GMAC 接收除了 pause 帧以外的所有控制帧。 10: GMAC 接收所有控制帧。 11: GMAC 根据地址过滤情况接收控制帧
DBF: Disable Broadcast Frames 关闭广播帧	5	0x0	此位为 1 时，过滤所有接收的广播帧。 此位为 0 时，接收所有广播帧。
PM: Pass All Multicast 接收所有多播帧	4	0x0	此位为 1 时，接收所有多播帧。 此位为 0 时，过滤所有多播帧。
DAIF: DA Inverse Filtering 目标地址反转过滤	3	0x0	此位为 1 时，对单播和多播帧进行反向目标地址匹配。 此位为 0 时，对单播和多播帧进行正常目标地址匹配。
HMC: Hash Multicast 哈希多播过滤	2	0x0	此位为 1 时，对接收到的多播帧根据哈希表的内容进行目标地址过滤。
HUC: Hash Unicast 哈希单播过滤	1	0x0	此位为 1 时，对接收到的单播帧根据哈希表的内容进行目标地址过滤。
PR: Promiscuous Mode 混杂模式	0	0x0	接收所有以太网帧。
Register2 (Hash Table High Register) Offset: 0x0008			
HTH: Hash Table High 哈希表高位	31:0	0x0	哈希表的高 32 位。
Register3 (Hash Table Low Register) Offset: 0x000C			
HTL: Hash Table Low 哈希表低位	31:0	0x0	哈希表的低 32 位。
Register4 (GMII Address Register) Offset: 0x0010			

Reserved 保留	31:16	0x0	保留
PA: Physical Layer Address PHY 地址	15:11	0x0	此域选择需要访问 32 个 PHY 中的哪个。
GR: GMII Register 需要访问的 PHY 设备中的寄存器	10:6	0x0	此域选择需要访问的的 PHY 的哪个 GMII 配置寄存器。
Reserved 保留	5	0x0	保留
CR: CSR Clock Range CSR 时钟范围	4:2	0x0	此域决定 MDC 时钟是 clk_csr_i 时钟频率比例。 0000 clk_csr_i/42 0001 clk_csr_i/62 0010 clk_csr_i/16 0011 clk_csr_i/26 0100 clk_csr_i/102 0101 clk_csr_i/124 0110, 0111 Reserved
GW: GMII Write GMII 写	1	0x0	此位为 1 时，通过 GMII 数据寄存器对 PHY 进行写操作 此位为 0 时，通过 GMII 数据寄存器对 PHY 进行读操作。
GB: GMII Busy GMII 忙	0	0x0	对寄存器 4 和寄存器 5 写之前，此位应为 0。在写寄存器 4 之前此位必须先置 0。在访问 PHY 的寄存器时，应用程序需要将此位设置为 1，表示 GMII 接口上有写或者读操作正在进行。
Register5 (GMII Data Register) Offset: 0x0014			
Reserved 保留	31:16	0x0	保留
GD: GMII Data GMII 数据	15:0	0x0	此域保存了对 PHY 进行管理读访问操作的 16 位数据，或者对 PHY 进行管理写访问的 16 位数据。
Register6 (Flow Control Register) Offset: 0x0018			
PT: Pause Time	31:16	0x0	此域保存了需要填入传输控制帧中的暂停时间域。

暂停时间			
Reserved 保留	15:8	0x0	保留
DZPQ: Disable Zero-Quanta Pause 禁止零时间片暂停帧	7	0x0	此位为 1 时，禁止自动零时间片的暂停控制帧的产生。
Reserved 保留	6	0x0	保留
PLT: Pause Low Threshold 暂停帧的低阈值	5:4	0x0	此域用于设置暂停时间的阈值。 00: 暂停时间减少 4 个时间槽 01: 暂停时间减少 28 个时间槽 10: 暂停时间减少 144 个时间槽 11: 暂停时间减少 256 个时间槽 （一个时间槽为在 GMII/MII 接口上传输 512 比特或者 64 字节的时间）
UP: Unicast Pause Frame Detect 单播的暂停帧探测	3	0x0	此位为 1 时，GMAC 将会根据 MAC 地址 0 指定的本站单播地址来探测暂停帧。
RFE: Receive Flow Control Enable 接收流控使能	2	0x0	此位为 1 时，GMAC 将会解析接收到的暂停帧，并且按照暂停帧指定的时间暂停帧的发送。
TEF: Transmit Flow Control Enable 发送流控使能	1	0x0	在全双工模式下，此位为 1 时，GMAC 使能暂停帧的发送。 在半双工模式下，此位为 1 时，GMAC 使能反压操作。
FCB/BPA: Flow Control Busy/Backpressure Activate 流控忙/反压激活	0	0x0	此位为 1 时，在全双工模式下发起暂停控制帧的发送或在半双工模式下启动反压操作。
Register7 (VLAN Tag Register) Offset: 0x001C			
Reserved 保留	31:17	0x0	保留

ETV: Enable 12-Bit VLAN Tag Comparison 使能 12 位 VLAN Tag 比较	16	0x0	此位为 1 时，使用 12 位 VLAN Tag 而不是使用 16 位 VLAN Tag 用于以太网帧比较和过滤。
VL: VLAN Tag Identifier for Receive Frames 帧接收的 VLAN Tag 标识	15:0	0x0	此域保存 802.1Q 格式的 VLAN Tag，用于比较接收到的以太网帧的位于第 15 和第 16 个字节的 VLAN Tag。
Register8 (Version Register) Offset: 0x0020			
Reserved 保留	15:8	0x0	保留
Version 版本号	7:0	0x0	0X35
Register14 (Interrupt Status Register) Offset: 0x0038			
Reserved 保留	15:8	0x0	保留
MMC Receive Checksum Offload Interrupt Status MMC 接收校验和卸载状态中断	7	0x0	MMC 校验和卸载寄存器产生任何中断产生时，此位设置为 1。
MMC Transmit Interrupt Status MMC 传输中断	6	0x0	MMC 传输中断寄存器产生任何中断时，此位设置为 1。
MMC Receive Interrupt Status MMC 接收中断状态	5	0x0	MMC 接收中断寄存器产生任何中断时，此位设置为 1。
MMC Interrupt Status MMC 中断状态	4	0x0	7:5 的任何位为高时，此位设置为 1。
PMT Interrupt Status	3	0x0	在 Power Down 状态下，收到 magic 帧或在 Wake-on-LAN 帧时，此位设置为 1。

电源管理中断状态			
PCS Auto-Negotiation Complete PCS 自动协商完成	2	0x0	RGMII PHY 接口自动协商完成时，此位设置为 1。
PCS Link Status Changed PCS 链路状态变化	1	0x0	RGMII PHY 接口的链路状态发生任何变化时，此位设置为 1。
RGMII Interrupt Status RGMII 中断状态	0	0x0	RGMII 接口的链路状态发生任何变化时，此位设置为 1。
Register15 (Interrupt Mask Register) Offset: 0x003C			
Reserved 保留	15:10	0x0	保留
Time Stamp Interrupt Mask 时间戳中断使能	9	0x0	此位为 1 时，禁止时间戳发生的中断
Reserved 保留	8:4	0x0	保留
PMT Interrupt Mask 电源管理中断使能	3	0x0	此位为 1 时，禁止电源管理引起的中断。
PCS AN Completion Interrupt Mask PCS 自动协商完成中断使能	2	0x0	此位为 1 时，禁止 PCS 自动协商完成中断。
PCS Link Status Interrupt Mask PCS 链路状态中断使能	1	0x0	此位为 1 时，禁止由于 PCS 链路状态变化引起的中断。
RGMII Interrupt Mask RGMII 中断使能	0	0x0	此位为 1 时，禁止 RGMII 引起的中断。
Register16 (MAC Address0 High Register) Offset: 0x0040			

MO: Always 1 保留	31	0x0	保留
Reserved 保留	30:16	0x0	保留
MAC Address0[47:32] MAC 地址高 16 位	15:0	0x0	存放用于接收地址过滤和传输流控帧的 MAC 地址。
Register17 (MAC Address0 Low Register) Offset: 0x0044			
MAC Address0[31:0] MAC 地址低 32 位	31:0	0x0	存放用于接收地址过滤和传输流控帧的 MAC 地址。
Register18 (MAC Address1 High Register) Offset: 0x0048			
AE: Address Enable 地址使能	31	0x0	此位为 1 时，地址过滤模块使用第 2 个 MAC 地址用于完全地址过滤。此位为 0 时，地址过滤模块不使用第 2 个 MAC 地址用于地址过滤。
SA: Source Address 源 MAC 地址	30	0x0	此位为 1 时，MAC 地址 1 用于比较接收帧的源 MAC 地址。 此位为 0 时，MAC 地址 1 用于比较接收帧的目标 MAC 地址。
MBC: Mask Byte Control 掩模字节控制	29:24	0x0	此域用于比较每个 MAC 地址的字节掩模控制位。比如第 29 位用于掩码寄存器 18 的[15:8]这个字节。
Reserved 保留	23:16	0x0	保留。
MAC Address1[47:32] 第 2 个 MAC 地址的高 16 位	15:0	0xFFFF	
Register19 (MAC Address1 Low Register) Offset: 0x004C			
MAC Address1[31:0] 第 2 个 MAC 地址的低 32 位	31:0	0x0	
Register48 (AN Control Register) Offset: 0x00C0			

Reserved 保留	31:19	0x0	保留
SGMII RAL Control 保留	18	0x0	保留
LR: Lock to Reference 锁定到参考时钟	17	0x0	此位为 1 时，PHY 将其锁相环锁定到 125MHz 的参考时钟。
ECD: Enable Comma Detect 使能停顿探测	16	0x0	此位为 1 时，使能 PHY 的停顿探测和字重同步。
Reserved 保留	15	0x0	保留
ELE: External Loopback Enable 外部环回使能	14	0x0	此位为 1 时，使能 PHY 进入环回模式。
Reserved 保留	13	0x0	保留
ANE: Auto-Negotiation Enable 自动协商使能	12	0x0	此位为 1 时，GMAC 将会和链路对方进行自动协商。
Reserved 保留	11:10	0x0	保留
RAN: Restart Auto-Negotiation 重新进行自动协商	9	0x0	此位为 1 时，重新进行自动协商。
Reserved 保留	8:0	0x0	保留
Register49 (AN Status Register) Offset: 0x00C4			
Reserved 保留	31:9	0x0	保留
ES: Extended Status	8	0x0	只读，因为 GMAC 支持扩展状态信息。

扩展状态			
Reserved 保留	7:6	0x0	保留
ANC: Auto-Negotiation Complete 自动协商完成	5	0x0	只读，指示自动协商完成。
Reserved 保留	4	0x0	保留
ANA: Auto-Negotiation Ability 自动协商能力	3	0x0	只读，因为 GMAC 支持自动协商。
LS: Link Status 链路状态	2	0x0	此位为 1 时，指示链路连接上。 此位为 0 时，指示链接未连接。
Reserved 保留	1:0	0x0	保留。
Register50 (Auto-Negotiation Advertisement Register)　Offset: 0x00C8			
Reserved 保留	31:16	0x0	保留
NP: Next Page Support 下一页面支持	15	0x0	只读为 0，因为 GMAC 不支持下一页面。
Reserved 保留	14	0x0	保留
RFE: Remote Fault Encoding 远端错误编码	13:12	0x0	此 2 位指示链路对端发生错误，具体编码将 IEEE 802.3z 第 37.2.1.5 小节。
Reserved 保留	11:9	0x0	保留
PSE: Pause Encoding Pause 位编码	8:7	0x0	见 IEEE 802.3z 第 37.2.1.4 小节

HD: Half-Duplex 半双工	6	0x0	此位为 1 时，指示 GMAC 支持半双工。
FD: Full-Duplex 全双工	5	0x0	此位为 1 时，指示 GMAC 支持全双工。
Reserved 保留	4:0	0x0	保留
Register51 (Auto-Negotiation Link Partner Ability Register)　Offset: 0x00CC			
Reserved 保留	31:16	0x0	保留
NP: Next Page Support 下一页面支持	15	0x0	此位为 1 时，指示有更多下一页面信息可用 此位为 0 时，指示下一页面交换不可用。
ACK: Acknowledge 确认	14	0x0	指示在自动协商中，链路对端成功接收到 GMAC 的基本页面。
RFE: Remote Fault Encoding 远端错误编码	13:12	0x0	见 IEEE 802.3z 第 37.2.1.5 小节。
Reserved 保留	11:9	0x0	保留
PSE: Pause Encoding 对端 pause 状态编码	8:7	0x0	见 IEEE 802.3z 第 37.2.14 小节。
HD: Half-Duplex 半双工	6	0x0	指示对端可以运行在半双工模式。
FD: Full-Duplex 全双工	5	0x0	指示对端可以运行在全双工模式。
Reserved 保留	4:0	0x0	保留
Register52 (Auto-Negotiation Expansion Register)　Offset: 0x00D0			
Reserved 保留	31:3	0x0	保留
NPA: Next Page	2	0x0	只读为 0，因为 GMAC 不支持下一页面。

Ability 下一页面能力			
NPR: New Page Received 接收到新页面	1	0x0	此位为 1 时，指示 GMAC 接收到新页面。
Reserved 保留	0	0x0	保留
Register54 (SGMII/RGMII Status Register) Offset: 0x00D8			
Reserved 保留	31:4	0x0	保留
Link Status 链路状态	3	0x0	此位为 1 时，指示链路连接上。 此位为 0 时，指示链路未连接上。
Link Speed 链路速度	2:1	0x0	指示链路当前速度 00: 2.5MHz 01: 25MHz 10: 125MHz
Link Mode 链路模式	0	0x0	0: 半双工 1: 全双工

7.3 DMA 描述符

DMA 描述符是 GMAC 驱动和硬件的交互接口，记录了数据包的内存地址和传输状态。在此分别定义了发送描述符(Tx Desciptor)和接收描述符(Rx Descriptor)两种数据结构。两种描述符可以自由选择分别以环式(ring mode)或者链式(chain mode)相连，以供 GMAC 使用。

7.3.1 DMA 描述符的基本格式

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

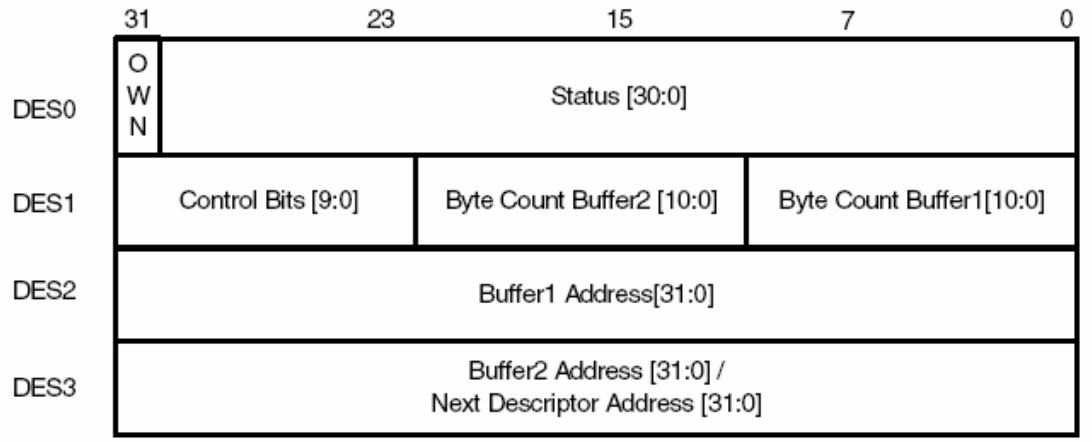


图 1　DMA 描述符的基本格式(小尾端 32 位总线)

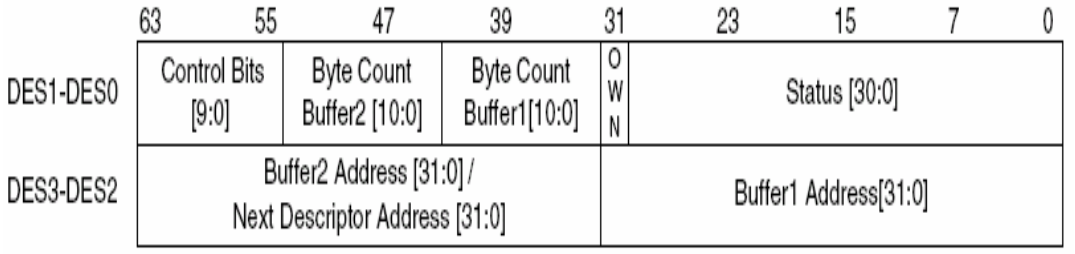


图 2　DMA 描述符的基本格式(小尾端 64 位总线)

7.3.2　DMA 接收描述符

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

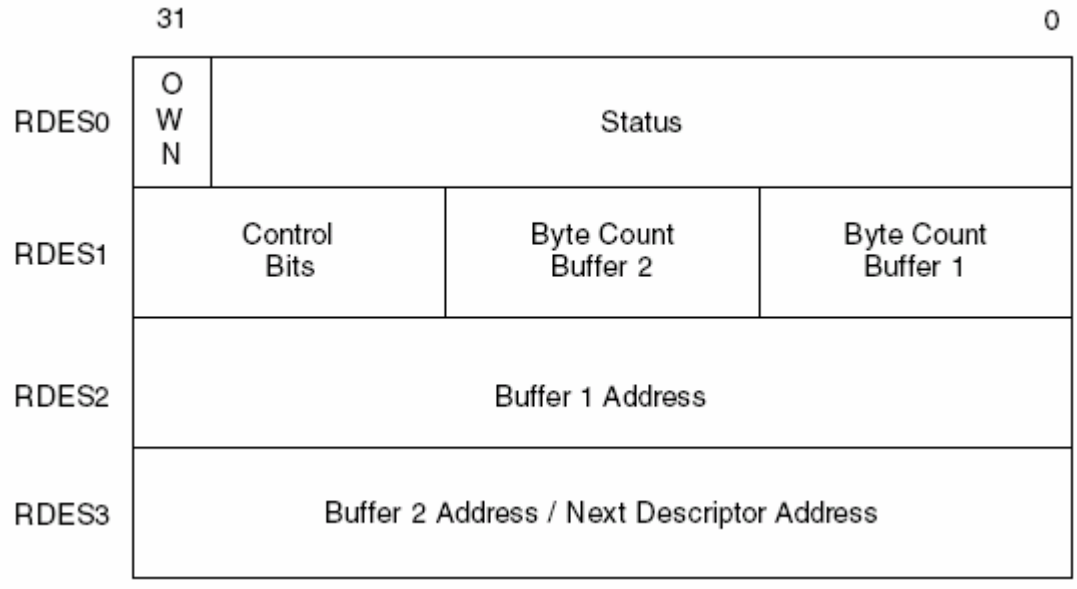


图 3 DMA 接收描述符的基本格式(小尾端 32 位总线)

7.3.3 RDES0

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

RDES0	位	
OWN 所有模式	31	该位为 1 时表示描述符当前属于 DMA 控制，0 表示属于主机控制。当 DMA 模块完成一次传输时，会将该位主动清 0
AFM: Destination Address Filter Fai 目标地址过滤错误 I	30	当该位为 1 时，表示当前数据帧目标地址不符合 GMAC 内部的帧目标地址过滤器
FR: Frame length 帧长度	29:16	表示接收当前帧的长度，当 ES 位为 0 时有效
ES: Error Summary 总体错误信息	15	指示当前帧是否出错，其值为 RDES[0]、RDES[1]、RDES[3]、RDES[4]、RDES[6]、RDES[7、RDES[11]、RDES[14]各位作或运算(OR)的结果
DE: Descriptor Error 描述符错误	14	当该位为 1 时表示，当前描述符所指向的 buffer 与帧不相符或者 OWN 为 0(主机控制)
SAF: Source Address Filter Fail 源地址过滤错误	13	当该位为 1 时，表示当前数据帧的源地址不符合 GMAC 内部的帧源地址过滤器
LE: Length Error 长度错误	12	当该位为 1 时，表示当前接收帧长度与默认长度不符。当 Frame Type 位为 1 且 CRC Error 位为 0 时有效
OE: Over Flow Error 溢出错误	11	当该位为 1 时，表示接收该帧时 GMAC 内部 RxFIFO 溢出
VLAN: VLAN Tag VLAN 标志	10	当该位为 1 时，表示该帧的类型为 VLAN
FS: First Desciptor 第一个描述符	9	当该位为 1 时，表示当前描述符所指向的 buffer 为当前接收帧的第一个保存 buffer
LS: Last Desciptor 最后一个描述符	8	当该位为 1 时，表示当前描述符所指向的 buffer 为当前接收帧的最后一个保存 buffer
IPC Checksum Error/Giant Frame	7	当该位为 1 时，如果 IPC 校验功能启用则表示当前帧的 IPv4 头校验值与帧内部校验域的值不相符。如果未启用则表示当

校验错误/超长帧		前帧为一个超长帧(长度大于 1518 字节)
LC: late collision 后期冲突	6	当该位为 1 时，表示在半双工模式下，当前帧接收时发生了一个后期冲突
FT: Frame Type 帧类型	5	当该位为 1 时，表示当前帧为一个以太网格式帧，为 0 时表示当前帧为一个 IEEE802.3 格式帧
RWT: Receive Watchdog Timeout	4	当该位为 1 时，表示当前时钟值超过了接收模块看门狗电路时钟的值，既接收帧超时
RE: Receive Error 接收错误	3	当该位为 1 时，表示接收当前帧时内部模块出错。内部信号 rxer 置 1 且 rxdv 置 1
DE: Dribble bit Error 奇数位错误	2	当该位为 1 时，表示接收帧长度不是整数，即总长度为奇数位，该位只有在 mii 模式下有效
CE: CRC Error 接收 CRC 校验错误	1	当该位为 1 时，表示接收当前帧时内部 CRC 校验出错。该位只有在 last descriptor(RDES0[8])为 1 时有效
RX MAC: Checksum/payload Checksum Error 接受校验/负载校验错误	0	当该位为 1 时，表示接收当前帧时内部 RX MAC 寄存器组 1-15 中存在一个匹配当前帧目的地址。为 0 时表示 RX MAC 寄存器组 0 匹配接受帧目的地址。如果 Full Checksum Offload Engine 启用时，为 1 表示该帧 TCP/UDP/ICMP 校验错误。该位为 1 时也可能表示当前帧实际接受长度与帧内部记载长度不相符。

7.3.4 RDES

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

RDES1	位	
Disable Intr in Completion 禁止完成后发中断	31	该位为 1 时表示该帧接收完成后将不会置起 STATUS 寄存器中 RI 位（CSR5[6]），这将会使得主机无法检测到该中断
Reserved 保留	30:26	
RER: Receive End of Ring 环型描述符结尾	25	该位为 1 时表示该描述符为环型描述符链表的最后一个，下一个描述符的地址为接收描述符链的基址
RCH: Second Address Chained	24	该位为 1 时表示描述符中的第二个 buffer 地址指向的是下一个描述符的地址，为 0 时表示该地址指向第二个 buffer 地址

第二个buffer地址指向下个链式描述符		当该位为 1 时，RDES1[21-11]的值将没有意义，RDES1[25]比 RDES1[24]具有更高优先级(代表环型而不是链型)
Reserved 保留	23:22	
RBS2: Receive Buffer Size 2 接收 buffer2 大小	21:11	该域表示数据 buffer2 的大小。根据系统总线的宽度 32/64/128，Buffer2 的大小应该为 4/8/16 的整数倍。如果不满足则会导致未知的结果。该域在 RDES1[24]为 0 时有效
RBS2: Receive Buffer Size 1 接收 buffer1 大小	10:0	该域表示数据 buffer1 的大小。根据系统总线的宽度 32/64/128，Buffer1 的大小应该为 4/8/16 的整数倍。如果不满足则会导致未知的结果。该域一直有效。如果该域值为 0，DMA 则会自动访问 buffer2 或者下一个接收描述符

7.3.5 RDES2

该域记录了数据接收 buffer1 的地址。

RDES2	位	
Buffer1 Address Pointer 接收 buffer1 地址	31:0	该域记录了数据接收 buffer1 的 32 位物理地址。该物理地址没有默认的对齐要求。当 GMAC DMA 内部实现了总线数据 32/64/128 位对齐，则该地址的低 2/3/4 位会被忽略

7.3.6 RDES3

该域记录了数据接收 buffer2 的地址。

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

7.3.7 DMA 发送描述符

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

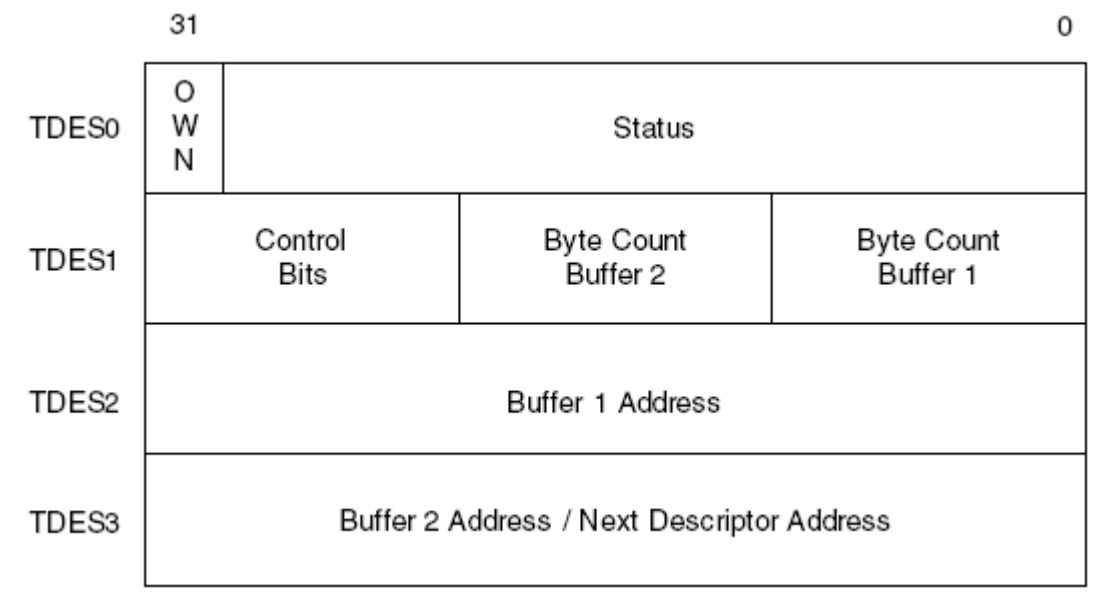


图 4 DMA 发送描述符的基本格式(小尾端 32 位总线)

7.3.8 TDES0

TDES0 包含了发送帧的状态和发送描述符的所属信息。

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

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

7.3.9 TDES1

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

TDES1	位	
IC: Interrption on Complete	31	该位为 1 时表示该帧接发送完成后将会置起 STATUS 寄存器中 TI 位(CSR5[0])

完成时中断		
LS: Last Segment 最后段	30	该位为 1 时表示当前 buffer 包含的是一帧数据的最后一段(如果帧分为多个段)
FS: First Segment 第一段	29	该位为 1 时表示当前 buffer 包含的是一帧数据的第一段(如果帧分为多个段)
CIC: Checksum Insertion Control 校验数据填充控制	28:27	该域控制内部模块是否在发送帧中填充校验数据。 值：2'b00: 不填充校验数据 2'b01: 填充 IPV4 头校验数据 2'b10: 在伪头数据(pseudo-header)存在的情况下，填充 TCP/UDP/ICMP 全校验数据 2'b11: 总是填充 TCP/UDP/ICMP 的全校验数据
DC: Disable CRC 禁止 CRC 校验	26	该位为 1 时 GMAC 硬件不在每个发送帧的结尾添加 CRC 校验数据
TER: Transmit End of Ring 环形描述符结尾	25	该位为 1 时表示该描述符为环型描述符链表的最后一个，下一个描述符的地址为发送描述符链的基址
TCH: Second Address Chained 第二个buffer地址指向下个链式描述符	24	该位为 1 时表示描述符中的第二个 buffer 地址指向的是下一个描述符的地址，为 0 时表示该地址指向第二个 buffer 地址 当该位为 1 时，TDES1[21-11]的值将没有意义，TDES1[25]比 TDES1[24]具有更高优先级(代表环型而不是链型)
DP: Dissable Pading 禁止填充	23	该位为 1 时表示 GMAC 将不会对长度小于 64 字节的数据包进行空数据填充
TTSE: Transmit Time Stamp Enable 启用发送时间戳	22	该位为 1 时表示将启用内部模块计算 IEEE1588 硬件时间戳计算，在 TDES1[29]为 1 时有效
TBS2: Transmit Buffer Size 2 发送 buffer2 大小	21:11	该域表示数据 buffer2 的大小。当 TDES1[24]为 1 时，该域无效
TBS1: Transmit Buffer Size 1 发送 buffer1 大小	10:0	该域表示数据 buffer1 的大小。该域一直有效。如果该域值为 0，DMA 则会自动访问 buffer2 或者下一个接收描述符

7.3.10 TDES2

该域记录了数据发送 buffer1 的地址。

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

7.3.11 TDES3

该域记录了数据发送 buffer2 的地址。

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

7.4 软件编程向导(Software Programming Guide):

DMA 初始化：

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

MAC 初始化：

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

c. 流控制忙/后压力启用

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

发送和接收的一般过程：

1. 检测到发送或接收中断后，查寻相应描述符来判断其是否属于主机，并读取描述符中的数据
2. 完成对描述符中数据的读取后，将描述符各位清 0 并设置其 OWN 位，使其继续发送/接收数据
3. 如果当前发送或接收描述符不属于 DMA(OWN=0)，则 DMA 模块会进入挂起状态。当有数据需要被发送或接收时，向 DMA Tx/Rx POLL 寄存器写 1 重新使能 DMA 模块。需要注意的是接收描述符在空闲时应该总是属于 DMA(OWN=1)

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

8 GMAC1

8.1 GMAC1 外部信号复用和配置

在 1B 芯片里，GMAC1 的大多数信号时通过复用来实现队外部的链接。

PAD	方向	描述	复用
GMAC1_MDCK	O	GMAC1 读写 PHY 的时钟信号	UART0_DCD
GMAC1_MDIO	I/O	GMAC1 读写 PHY 的数据信号	UART0_RI
GMAC1_RX_CTL	I	GMAC1 接收数据控制信号	UART0_RX
GMAC1_RX0	I	GMAC1 接收数据输入 0	UART0_TX
GMAC1_RX1	I	GMAC1 接收数据输入 1	UART0_RTS
GMAC1_RX2	I	GMAC1 接收数据输入 2	UART0_CTS
GMAC1_RX3	I	GMAC1 接收数据输入 3	UART0_DSR
GMAC1_TX0	O	GMAC1 传输数据输出 0	UART1_RX
GMAC1_TX1	O	GMAC1 传输数据输出 1	UART1_TX
GMAC1_TX2	O	GMAC1 传输数据输出 2	UART1_RTS
GMAC1_TX3	O	GMAC1 传输数据输出 3	UART1_CTS
GMAC1_TX_CLK_I	I	GMAC1 传输时钟输入	无复用
GMAC1_TX_CLK_O	O	GMAC1 传输时钟输出	
GMAC1_RX_CLK_I	I	GMAC1 接收时钟输入	
GMAC1_TX_CTL_O	O	GMAC1 输出数据控制信号	

从上表可以看出，GMAC1 的多数 PAD 通过复用 UART0 和 UART1 的信号实现。

这些复用的控制和配置如下：

GMAC1 复用 UART0 部分控制信号：GMAC1_USE_UART0;

GMAC1 复用 UART1 部分控制信号：GMAC1_USE_UART1;

参数名称	BIT 位	缺省值	描述
GPIO_MUX 基地址：0XBFD0_0420			
GMAC1_USE_UART0	3	0	UART0 的信号用来复用给 GMAC1 使用
GMAC1_USE_UART1	4	0	UART1 的信号用来复用给 GMAC1 使用

8.2 寄存器描述

GMAC 寄存器包括 GMAC 寄存器部分和 DMA 寄存器部分。GMAC1 的 GMAC 寄存器的起始地址是 0XBFE2_0000；GMAC1 的 DMA 寄存器的起始地址是 0XBFE2_1000。DMA 寄存器和 GMAC 寄存器的具体意义请参照第 7 章。

9 USB HOST

9.1 总体概述

1B 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

USB 主机控制器模块图如图 9-1、图 9-2 所示：

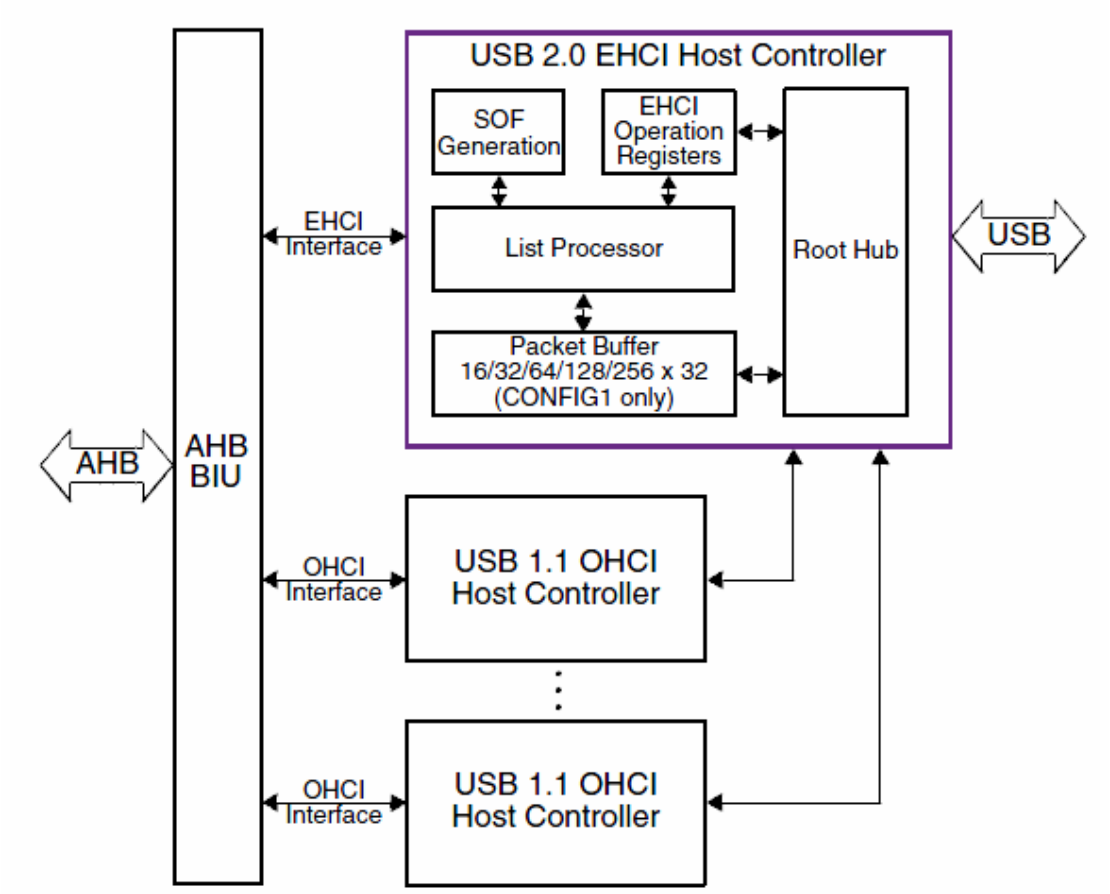


图 9-1 USB 主机控制器模块图

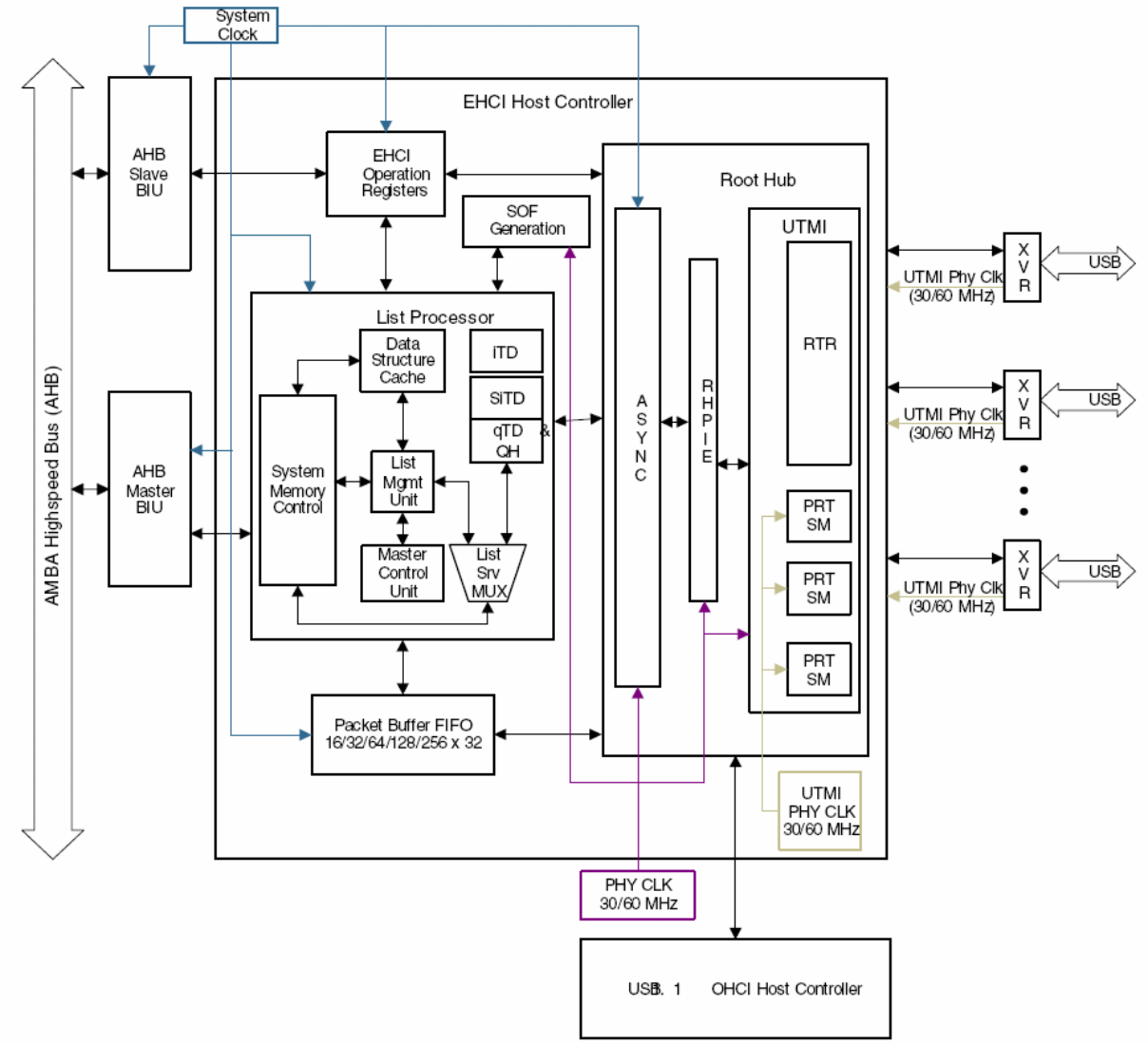


图 9-2 USB 主机控制器细节模块图（带 EHCI 控制器细节）

9.2 USB 主机控制器寄存器

9.2.1 EHCI 相关寄存器

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

9.2.2 Capability 寄存器

名称	地址	宽度	访问	说明
HCCAPBASE	0xbfe00000	32	RO	默认值为 32’h01000010
HCSPARAMS	0xbfe00004	32	RO	默认值为 32’h00001116
HCCPARAMS	0xbfe00008	32	RO	默认值为 32’h0000A010

（注：USBBase 固定为 EHCI slave 的起始地址 0xbfe00000）

9.2.3 Operational 寄存器

名称	地址	宽度	访问	说明
USBCMD	0xbfe00010	32	R/W、RO	USB 主机控制器的命令寄存器
USBSTS	0xbfe00014	32	R/W、RO	USB 主机控制器的状态寄存器
USBINTR	0xbfe00018	32	R/W	USB 主机控制器的中断设置寄存器
FRINDEX	0xbfe0001c	32	R/W	USB 主机控制器的帧索引寄存器
CTRLDSSEGMENT	0xbfe00020	32	R/W	存放 EHCI 控制数据结构的地址
PERIODICLISTBASE	0xbfe00024	32	R/W	存放周期数据帧表的起始地址
ASYNCLISTADDR	0xbfe00028	32	R/W	存放下一个要被执行的异步队列的起始地址
CONFIGFLAG	0xbfe00050	32	R/W	配置模式寄存器
PORTSC 1	0xbfe00054	32	R/W、RO	端口 1 状态和控制寄存器
PORTSC 2	0xbfe00058	32	R/W、RO	端口 2 状态和控制寄存器

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

9.2.4 EHCI 实现相关寄存器

EHCI 实现相关寄存器的详细描述如下。

名称	地址	宽度	访问	说明
INSNREG00	0xbfe00090	32	R/W	帧的长度配置寄存器
INSNREG01	0xbfe00094	32	R/W	数据包缓冲区 OUT/IN 阈值寄存器
INSNREG02	0xbfe00098	32	RO	数据包缓冲深度寄存器
INSNREG03	0xbfe0009c	32	RO ,R/W	参照寄存器详细描述
INSNREG04	0xbfe000a0	32	R/W	用于 Debug
INSNREG05	0xbfe000a4	32	RO, R/W	UTMI 配置（默认配置），控制和状态寄存器
INSNREG06	0xbfe000a8	32	RO	AHB 错误状态寄存器
INSNREG07	0xbfe000ac	32	RO	AHB Master 错误地址寄存器
INSNREG08	0xbfe000b0	32	RO	HSIC 使能寄存器

9.2.4.1 INSNREG00 寄存器（disable）

9.2.4.2 INSNREG01 寄存器

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

9.2.4.3 INSNREG02 寄存器

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

9.2.4.4 INSNREG03 寄存器

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

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

位域	访问	复位值	说明
31:6	Reserved	26'h0	保留
5	R/W	1'h0	置 1：禁止 automatic 功能，即当软件清除 Run/Stop 位时，USB 主机控制器会把挂起（Suspend）的端口唤醒 置 0：启用 automatic 功能，当 reset Run/Stop 位时，Suspend 信号会置为 1
4	R/W	1'h0	置 1：禁止 NAK reload 修复 置 0：启用 NAK reload 修复
3	Reserved	1'h0	保留
2	R/W	1'h0	置 1：缩短端口枚举（enumeration）时间（仿真）

1	R/W	1’h0	置 1： HCCPARAMS 寄存器的第 17、15:4、2:0 位均可写
0	R/W	1’h0	置 1： HCSPARAMS 寄存器可写

9.2.4.6 INSNRE05 寄存器

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

9.2.4.7 INSNREG06 寄存器

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

9.2.4.8 INSNREG07 寄存器

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

9.2.4.9 INSNREG08 寄存器

位域	访问	复位值	说明
31:0	RO	1’b0	HSIC 使能

9.3 OHCI 相关寄存器

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

9.3.1 Operational 寄存器

名称	地址	宽度	访问	说明
HcRevision	0xbfe08000	32	-	控制和状态
HcControl	0xbfe08004	32	-	
HcCommonStatus	0xbfe08008	32	-	
HcInterruptStatus	0xbfe0800C	32	-	
HcInterruptEnable	0xbfe08010	32	-	
HcInterruptDisable	0xbfe08014	32	--	
HcHCCA	0xbfe08018	32	-	内存指针
HcPeriodCuttentED	0xbfe0801C	32	-	
HcControlHeadED	0xbfe08020	32	–	
HcControlCurrentED	0xbfe08024	32	–	
HcBulkHeadED	0xbfe08028	32	–	
HcBulkCurrentED	0xbfe0802C	32	–	
HcDoneHead	0xbfe08030	32	–	
HcRmInterval	0xbfe08034	32	–	帧计数器
HcFmRemaining	0xbfe08038	32	–	
HcFmNumber	0xbfe0803C	32	–	
HcPeriodicStart	0xbfe08040	32	–	
HcLSThreshold	0xbfe08044	32	–	
HcRhDescriptorA	0xbfe08048	32	–	根集线器
HcRhDescriptorB	0xbfe0804C	32	–	
HcRhStatus	0xbfe08050	32	–	
HcRhPortStatus1	0xbfe08054	32	–	
HcRhPortStatus2	0xbfe08058	32	–	

9.3.2 OHCI 实现相关寄存器

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

名称	地址	宽度	访问	说明
INSNREG06	0xbfe08098	32	RO	AHB 错误状态寄存器
INSNREG07	0xbfe0809c	32	RO	AHB Master 错误地址寄存器

9.3.2.1 INSNREG06 寄存器

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

9.3.2.2 INSNREG07 寄存器

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

9.4 USB 主机控制器时序

本节介绍 USB 2.0 控制器通过 UTMI 传输和接收 USB 差分信号的时序。

9.4.1 数据接收时序

如图 9-3、图 9-4 所示，随着一个 SYNC 序列到来，rx_active 置 1 标志着数据接收的开始。rx_valid 和 rx_validh 置 1 标志着通过 UTMI 并行接收端口的 16 bit 数据已经准备好，在传输完最后一个包后，rx_valid 和 rx_validh 置 0。

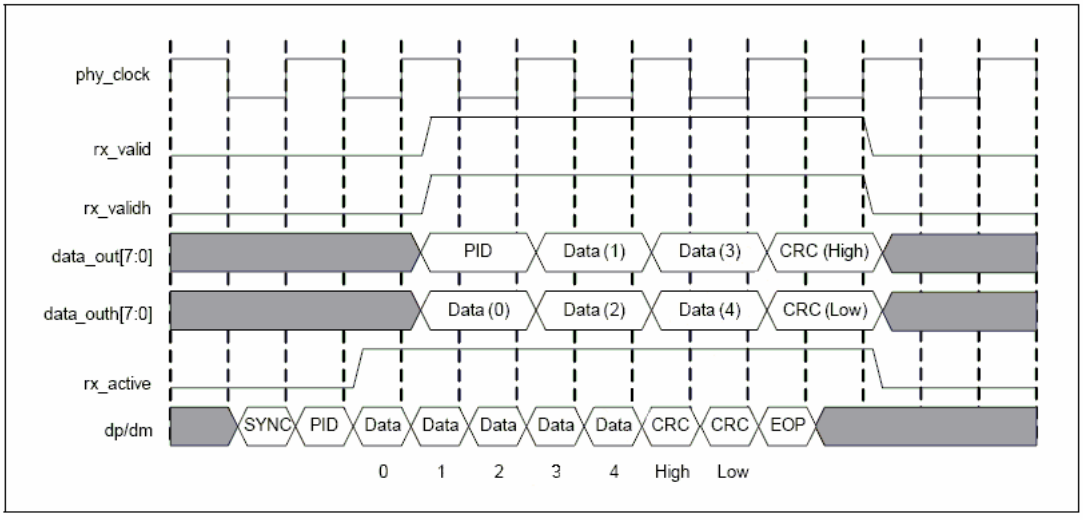


图 9-3 接收时序图（16 bit UTMI 接口，偶数个数据）

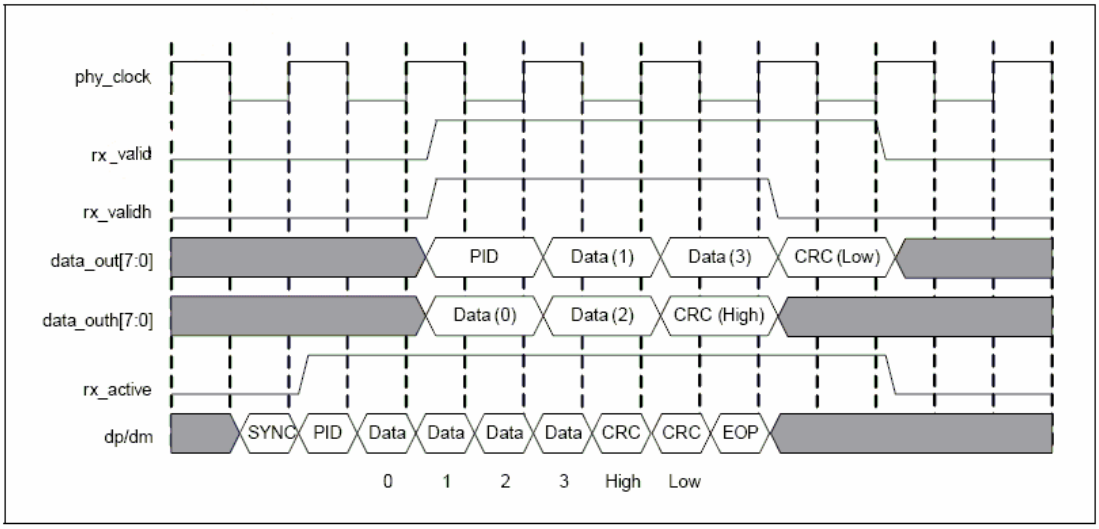


图 9-4 接收时序图（16 bit UTMI 接口，奇数个数据）

9.4.2 数据传输时序

如图 12-5、图 9-6 所示，tx_valid 和 tx_validh 在 SYNC 序列开始之前置 1，tx_ready 置 1 标志着数据可以被传输至 UTMI 并行发送端口，在传输完最后一个包后，tx_valid 和 tx_validh 置 0。

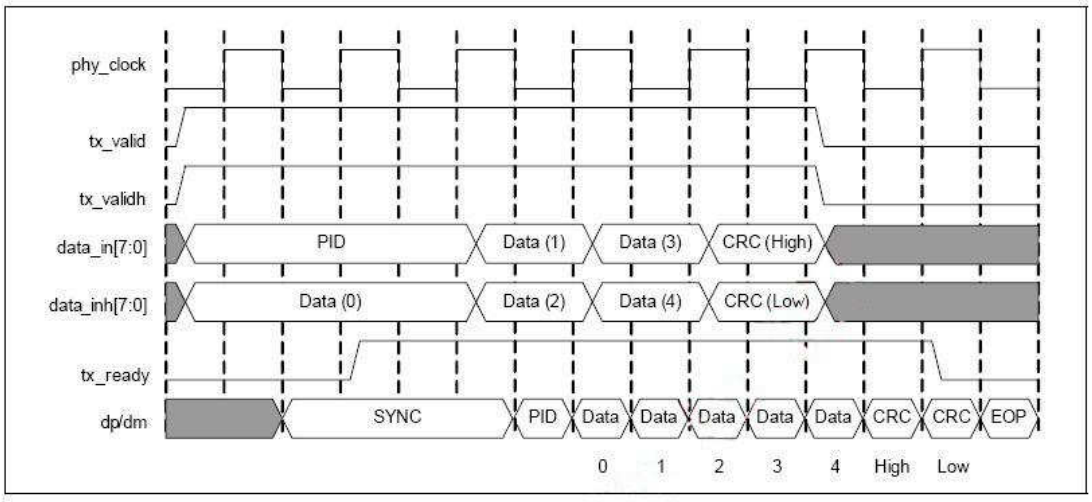


图 9-5 传输时序图（16 bit UTMI 接口，偶数个数据）

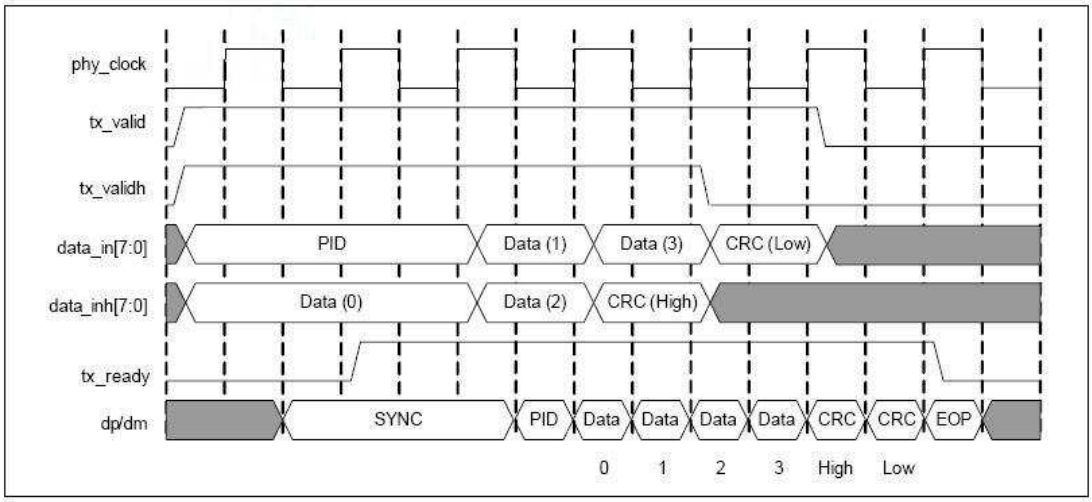


图 9-6　传输时序图（16bit UTMI 接口，奇数个数据）

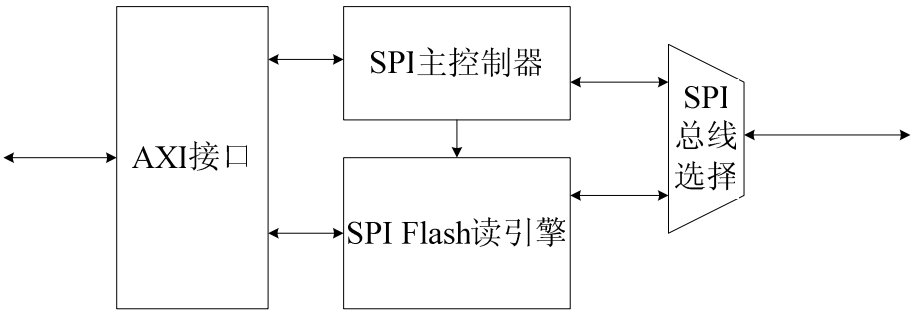
10 SPI0

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

10.1 SPI 控制器结构

1B 集成的 SPI 控制器仅可作为主控端，所连接的是从设备。其结构如下图所示，由一个 SPI 主控制器和 SPI Flash 读引擎组成。对于软件而言，SPI 控制器除了有若干 IO 寄存器外还有一段映射到 SPI Flash 的只读 memory 空间。如果将这段 memory 空间分配在 0xbfc00000，复位后不需要软件干预就可以直接访问，从而支持处理器从 SPI Flash 启动。SPI0 的 IO 寄存器的基地址 0xbfe80000，外部存储地址空间是 0xbf00, 0000 – 0xbf7f, ffff 共 8MB。

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



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

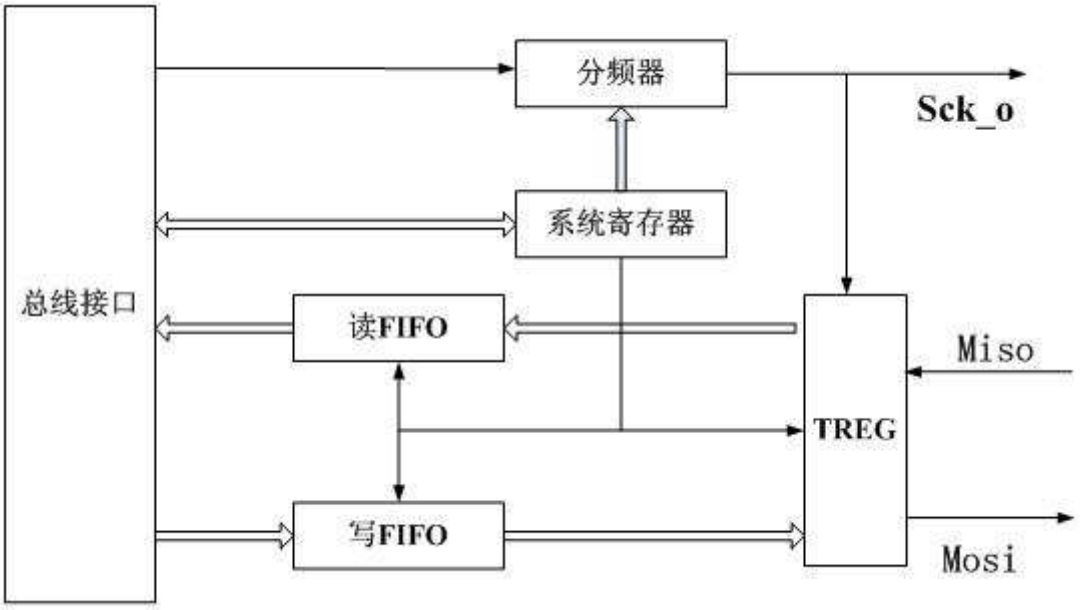


图 10-1 SPI 主控制器结构

10.2 SPI 控制器寄存器

10.2.1 控制寄存器（SPCR）

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

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

10.2.2 状态寄存器（SPSR）

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

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

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

10.2.3 数据寄存器（TxFIFO/RxFIFO）

中文名： 数据传输寄存器

寄存器位宽： [7：0]

偏移量： 0x02

复位值： 0x00

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

10.2.4 外部寄存器（SPER）

中文名： 外部寄存器

寄存器位宽： [7：0]

偏移量： 0x03

复位值： 0x00

位域	位域名称	位宽	访问	描述
7:6	icnt	2	RW	在传输完多少个字节后送出中断申请信号 00 – 1 字节 01 - 2 字节 10 - 3 字节 11 - 3 字节
5:3	Reserved	3	RW	保留
2	mode	1	RW	spi 接口模式控制 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	2	RW	与 Spr 一起设定分频的比率

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

10.2.5 参数控制寄存器（SFC_PARAM）

中文名： SPI Flash 参数控制寄存器

寄存器位宽： [7：0]

偏移量： 0x04

复位值： 0x21

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

10.2.6 片选控制寄存器（SFC_SOFTCS）

中文名： SPI Flash 片选控制寄存器
寄存器位宽： [7：0]
偏移量： 0x05
复位值： 0x00

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

10.2.7 时序控制寄存器（SFC_TIMING）

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

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

10.3 接口时序

SPI 主控制器外部接口时序图

如图 10-2 所示，SPI 主控制器发送数据时，数据提前半拍放在 MOSI 引线上，接着从设备端用时钟边沿锁存数据。根据时钟极性（CPOL）和时钟相位（CPHA）的设定，有 4 种可能的时序关系。

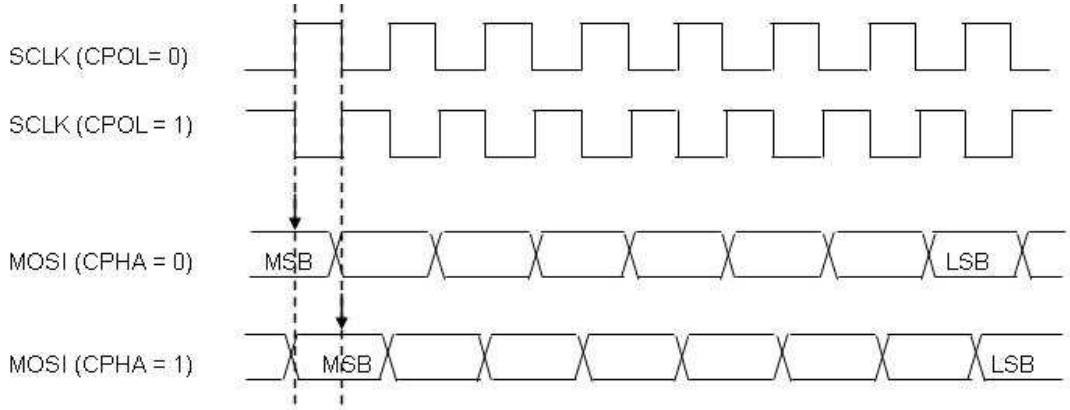
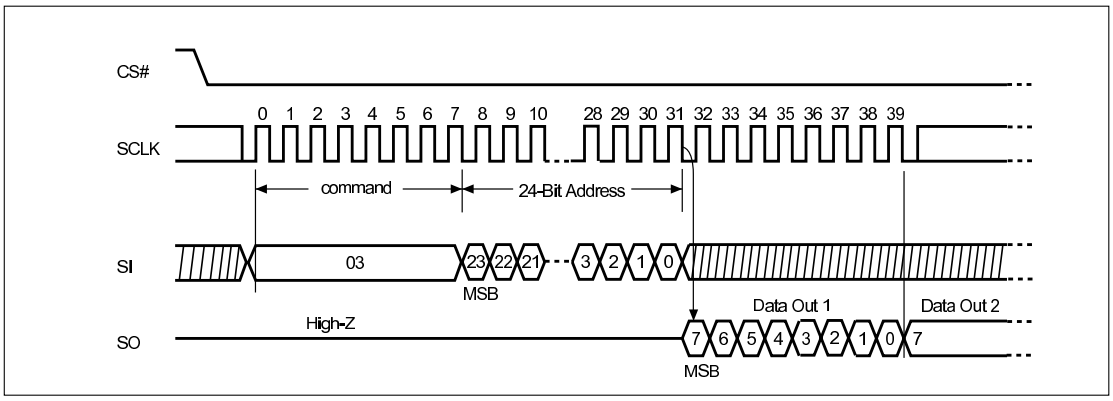


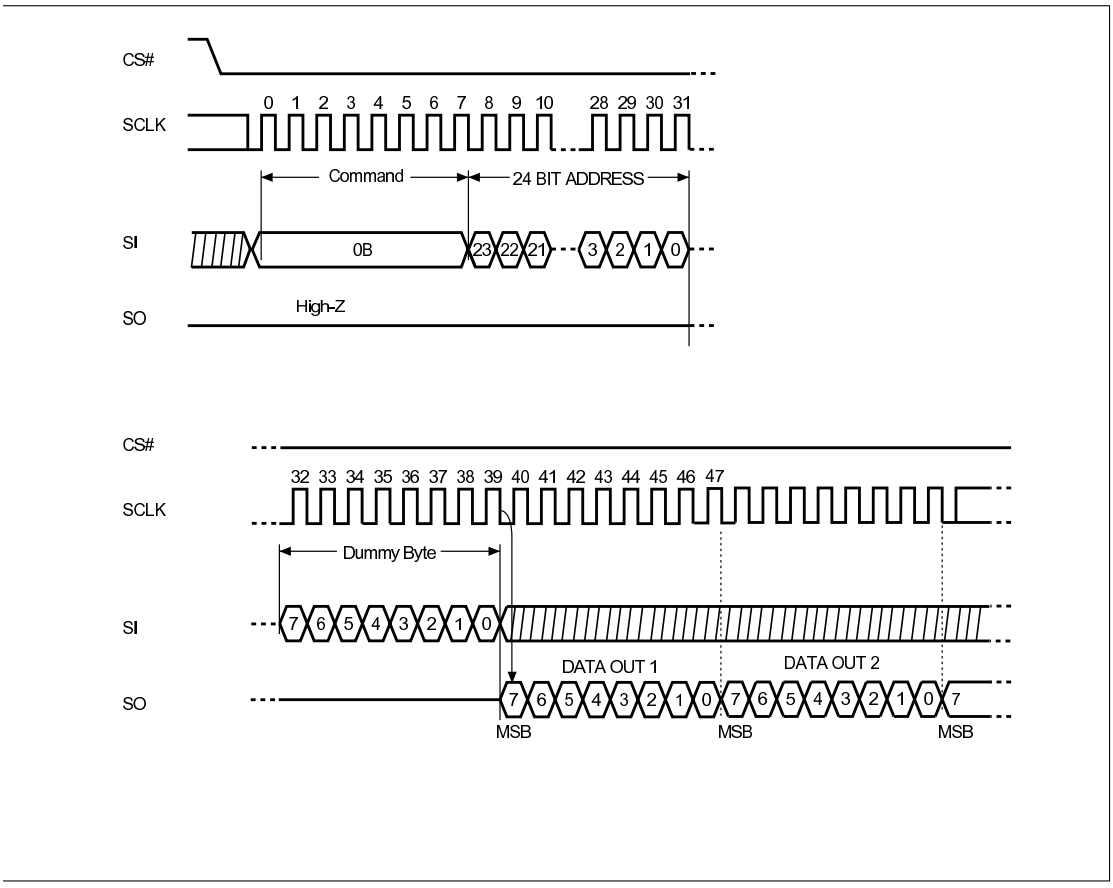
图 10-2 SPI 主控制器时序图

SPI Flash 访问时序图

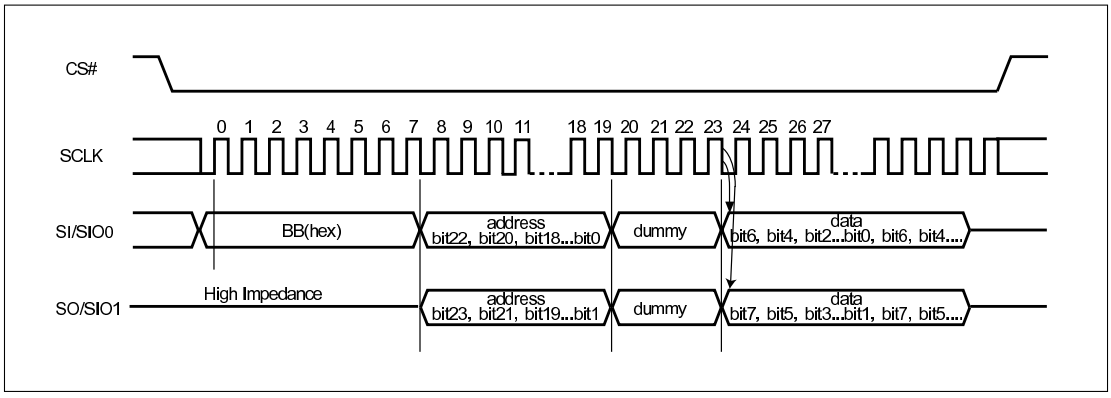
- 标准读模式



- 快速读模式



- 双 I/O 模式



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

10.4 SPI Flash 控制器使用指南

SPI 主控制器的读写操作

1. 模块初始化

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

2. 模块的发送/传输操作

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

3. 中断处理

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

硬件 SPI Flash 读

1. 初始化

- 将 SFC_PARAM 的 memory_en 位写 1。当 SPI 被选为启动设备时此位复位为 1。

- 设置读参数(时钟分频、连续地址读、快速读、双 I/O、tCSH 等)。这些参数复位值均为最保守的值。

2. 更改参数

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

混合访问 SPI Flash 和 SPI 主控制器

1. 对 SPI Flash 进行读以外的访问

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

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

11 SPI1

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

11.1 SPI 主控制器结构

SPI1 和 SPI0 的实现完全一样，系统启动地址不会映射到 SPI1 控制器，所以 SPI1 不支持系统启动。SPI1 的外部存储地址空间是 0xbf80,0000 - 0xbfbf,ffff 共 4MB。所有结构和配置相关请参考第 10 章信息。

12 Conf and Interrupt

12.1 配置和中断控制器总体描述

1B 芯片内置简单、灵活的中断控制器。1B 芯片的中断控制器除了管理 GPIO 输入的中断信号外，中断控制器还处理内部事件引起的中断。所有的中断寄存器的位域安排相同，一个中断源对应其中一位。中断控制器共四个中断输出连接 CPU 模块，分别对应 INT0, INT1, INT2, INT3。芯片支持 64 个内部中断和 64 个 GPIO 的中断；其中 INT0 和 INT1 分别对应于 64 个内部中断的前后 32 位，INT2 和 INT3 对应于 62 个外部 GPIO 中断。具体如下表所示：

	INT0	INT1	INT2	INT3
31	保留	保留	保留	保留
30	UART5	保留	GPIO30	保留
29	UART4	保留	GPIO29	GPIO61
28	TOY_TICK	保留	GPIO28	GPIO60
27	RTC_TICK	保留	GPIO27	GPIO59
26	TOY_INT2	保留	GPIO26	GPIO58
25	TOY_INT1	保留	GPIO25	GPIO57
24	TOY_INT0	保留	GPIO24	GPIO56
23	RTC_INT2	保留	GPIO23	GPIO55
22	RTC_INT1	保留	GPIO22	GPIO54
21	RTC_INT0	保留	GPIO21	GPIO53
20	PWM3	保留	GPIO20	GPIO52
19	PWM2	保留	GPIO19	GPIO51
18	PWM1	保留	GPIO18	GPIO50
17	PWM0	保留	GPIO17	GPIO49
16	保留	保留	GPIO16	GPIO48
15	DMA2	保留	GPIO15	GPIO47
14	DMA1	保留	GPIO14	GPIO46
13	DMA0	保留	GPIO13	GPIO45
12	保留	保留	GPIO12	GPIO44
11	保留	保留	GPIO11	GPIO43
10	AC97	保留	GPIO10	GPIO42
9	SPI1	保留	GPIO09	GPIO41
8	SPI0	保留	GPIO08	GPIO40
7	CAN1	保留	GPIO07	GPIO39
6	CAN0	保留	GPIO06	GPIO38
5	UART3	保留	GPIO05	GPIO37
4	UART2	保留	GPIO04	GPIO36
3	UART1	Gmac1	GPIO03	GPIO35

2	URAT0	Gmac0	GPIO02	GPIO34
1	保留	Ohci	GPIO01	GPIO33
0	保留	Ehci	GPIO00	GPIO32

1B 互联基于 XBAR，XBAR 对各主设备的窗口进行配置，每个主设备配置 8 各窗口，每个窗口的寄存器包括 BASE，MASK，MMAP 组成。

12.2 中断控制器寄存器描述

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

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

偏移地址	位	寄存器	描述	读写特性
0xbfd01040	32	INTISR0	中断控制状态寄存器 0	RO
0xbfd01044	32	INTIEN0	中断控制使能寄存器 0	R/W
0xbfd01048	32	INTSET0	中断置位寄存器 0	R/W
0xbfd0104c	32	INTCLR0	中断清空寄存器 0	R/W
0xbfd01050	32	INTPOL0	高电平触发中断使能寄存器 0	R/W
0xbfd01054	32	INTEDGE0	边沿触发中断使能寄存器 0	R/W
0xbfd01058	32	INTISR1	中断控制状态寄存器 1	RO
0xbfd0105c	32	INTIEN1	中断控制使能寄存器 1	R/W
0xbfd01060	32	INTSET1	中断置位寄存器 1	R/W
0xbfd01064	32	INTCLR1	中断清空寄存器 1	R/W
0xbfd01068	32	INTPOL1	高电平触发中断使能寄存器 1	R/W
0xbfd0106c	32	INTEDGE1	边沿触发中断使能寄存器 1	R/W
0xbfd01070	32	INTISR2	中断控制状态寄存器 2	RO
0xbfd01074	32	INTIEN2	中断控制使能寄存器 2	R/W
0xbfd01078	32	INTSET2	中断置位寄存器 2	R/W
0xbfd0107c	32	INTCLR2	中断清空寄存器 2	R/W
0xbfd01080	32	INTPOL2	高电平触发中断使能寄存器 2	R/W
0xbfd01084	32	INTEDGE2	边沿触发中断使能寄存器 2	R/W
0xbfd01088	32	INTISR3	中断控制状态寄存器 3	RO
0xbfd0108c	32	INTIEN3	中断控制使能寄存器 3	R/W
0xbfd01090	32	INTSET3	中断置位寄存器 3	R/W
0xbfd01094	32	INTCLR3	中断清空寄存器 3	R/W

0xbfd01098	32	INTPOL3	高电平触发中断使能寄存器 3	R/W
0xbfd0109c	32	INTEDGE3	边沿触发中断使能寄存器 3	R/W
0xbfd010c0	32	GPIOCFG0	GPIO 配置寄存器 0	R/W
0xbfd010c4	32	GPIOCFG1	GPIO 配置寄存器 1	R/W
0xbfd010d0	32	GPIOOE0	GPIO 配置寄存器输人使能 0	R/W
0xbfd010d4	32	GPIOOE1	GPIO 配置寄存器输入使能 1	R/W
0xbfd010e0	32	GPIOIN0	GPIO 配置寄存器输入寄存器 0	R/W
0xbfd010e4	32	GPIOIN1	GPIO 配置寄存器输入寄存器 1	R/W
0xbfd010f0	32	GPIOOUT0	GPIO 配置寄存器输出寄存器 0	R/W
0xbfd010f4	32	GPIOOUT1	GPIO 配置寄存器输出寄存器 1	R/W
0xbfd01160	32	ORDER_REG_ADDR	DMA 模块控制寄存器位	

XBAR 上各主设备地址窗口的配置如下表所示：

偏移地址	位	寄存器	描述	读写特性
0xbfd00000	64	CPU_WIN0_BASE	Cpu 配置窗口 0 基地址	R/W
0xbfd00008	64	CPU_WIN1_BASE	Cpu 配置窗口 1 基地址	R/W
0xbfd00010	64	CPU_WIN2_BASE	Cpu 配置窗口 2 基地址	R/W
0xbfd00018	64	CPU_WIN3_BASE	Cpu 配置窗口 3 基地址	R/W
0xbfd00020	64	CPU_WIN4_BASE	Cpu 配置窗口 4 基地址	R/W
0xbfd00028	64	CPU_WIN5_BASE	Cpu 配置窗口 5 基地址	R/W
0xbfd00030	64	CPU_WIN6_BASE	Cpu 配置窗口 6 基地址	R/W
0xbfd00038	64	CPU_WIN7_BASE	Cpu 配置窗口 7 基地址	R/W
0xbfd00040	64	CPU_WIN0_MASK	Cpu 配置窗口 0 掩码地址	R/W
0xbfd00048	64	CPU_WIN1_MASK	Cpu 配置窗口 1 掩码地址	R/W
0xbfd00050	64	CPU_WIN2_MASK	Cpu 配置窗口 2 掩码地址	R/W
0xbfd00058	64	CPU_WIN3_MASK	Cpu 配置窗口 3 掩码地址	R/W
0xbfd00060	64	CPU_WIN4_MASK	Cpu 配置窗口 4 掩码地址	R/W
0xbfd00068	64	CPU_WIN5_MASK	Cpu 配置窗口 5 掩码地址	R/W
0xbfd00070	64	CPU_WIN6_MASK	Cpu 配置窗口 6 掩码地址	R/W
0xbfd00078	64	CPU_WIN7_MASK	Cpu 配置窗口 7 掩码地址	R/W
0xbfd00080	64	CPU_WIN0_MMAP	Cpu 配置窗口 0 映射地址	R/W
0xbfd00088	64	CPU_WIN1_MMAP	Cpu 配置窗口 1 映射地址	R/W
0xbfd00090	64	CPU_WIN2_MMAP	Cpu 配置窗口 2 映射地址	R/W
0xbfd00098	64	CPU_WIN3_MMAP	Cpu 配置窗口 3 映射地址	R/W
0xbfd000a0	64	CPU_WIN4_MMAP	Cpu 配置窗口 4 映射地址	R/W
0xbfd000a8	64	CPU_WIN5_MMAP	Cpu 配置窗口 5 映射地址	R/W
0xbfd000b0	64	CPU_WIN6_MMAP	Cpu 配置窗口 6 映射地址	R/W
0xbfd000b8	64	CPU_WIN7_MMAP	Cpu 配置窗口 7 映射地址	R/W

13 DMA

13.1 DMA 控制器结构描述

DMA 来进行 DDR2 与 APB 设备间数据搬移工作，大大节省了资源提高了系统数据传输的效率，我们设计的是单通道传输方式由 CPU 可配置的 DMA 结构。DMA 挂在 AXI_MST_MUX 接口上，作为 AXI_MST_MUX 的一个主设备，来读写 DDR2；又挂在 AXI_SLV_MUX 接口上，作为 AXI_SLV_MUX 的一个从设备，被 CPU 配置。这样的结构方便软硬件的结合能更好的控制 DMA 的数据传输。

DMA 的传送数据的过程由三个阶段组成：

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

b) 数据传送：在 DMA 控制器的控制下自动完成。

c) 传送结束处理：发送中断请求。

本 DMA 控制器是一个基于 AXI 总线的单通道、可配置的 DMA 控制器 IP 核，主要功能就是在 1B 芯片上集成了 DMA 功能，专门负责在 DDR2 与 APB 设备间搬运数据。本 DMA 控制器限定为以字为单位的数据搬运。根据 DMA 的定义，设计了下个描述符地址寄存器、源地址寄存器、目的地址寄存器、传送字数计数器、传送步长间隔、传送循环次数、DMA 控制逻辑等必备寄存器；设计了符合 AXI 总线协议的总线接口用来接 AXI 设备，符合 APB 总线协议的总线接口用来接 APB 设备。DMA 的缓存大小为 128Byte（32x4Byte），以字为单位读写。如果以字节为单位读写，DMA 的逻辑面积很大，为减少读写 DMA 缓存的逻辑使用以字为单位的读、写。

CPU 通过 AXI 总线接口配置 DMA 的寄存器，将来自于 ddr2 或 APB 设备的数据保存在缓存中，将缓存中的数据写入要对应的内存或设备中去，最后发送 DMA 传输结束信号。在 DMA 传输过程中，CPU 可以监听 DMA 的工作状态。

13.2 DMA 控制器与 APB 设备的交互

在 1B 设计中，使用 DMA 的 APB 设备包括 NAND，AC97， 每个设备都有单独的 DMA 控制器。在 DMA 控制器的设计中有一小部分代码是用来处理 AC97 写操作的。AC97 的写通道是双通道，且 AC97 写使能的判断条件的 DMA_DADDR[31]=1，

DMA_DADDR[29:28]域为 AC97 写模式选择域，判断 AC97 写操作是字节、半字或者字操作，与 AC97 的写模式配置一致。所以，如果是写 AC97 的写操作，需要在配置 DMA 描述符时，将 DAM_DADDR[31]配置为 1，将 DMA_DADDR[29:28]配置为需要的 AC97 写模式。

13.3 DMA 控制器

DMA_ORDER_ADDR
中文名： 下一个描述符地址寄存器
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	存储器内部下一个描述符地址寄存器
0	Dma_order_en	1	R/W	描述符是否有效信号

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

13.3.1 DMA_SADDR

中文名： 内存地址寄存器
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

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

13.3.2 DMA_DADDR

中文名： 设备地址寄存器
寄存器位宽： [31：0]
偏移地址： 0x8
复位值： 0x00000000

位域	位域名称	位宽	访问	描述
31		1	R/W	AC97 写使能，“1”表示是写操作
30		1	R/W	0:mono 1: 2 stero
29:28		2	R/W	AC97 写模式,0: 1byte,1: 2byte,2: 4byte

27:0	dma_daddr	32	R/W	DMA 操作的 APB 设备地址

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

13.3.3 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。

13.3.4 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。

13.3.5 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。

13.3.6 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 ddr2 写设备，“0”为读设备写 ddr2
11:8	dma_write_state	4	R/W	DMA 写数据状态
7:4	dma_read_state	4	R/W	DMA 读数据状态
3	dma_trans_over	1	R/W	DMA 执行完被配置的所有描述符操作
2	dma_single_trans_over	1	R/W	DMA 执行完一次描述符操作
1	dma_int	1	R/W	DMA 中断信号
0	dma_int_mask	1	R/W	DMA 中断是否被屏蔽掉
位域	位域名称	位宽	访问	描述

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

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

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

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

Read_state	【3:0】	描述
Read_idle	4’h0	读状态正处于空闲状态
Read_ready	4’h1	接收到开始 dma 操作的 start 信号后，进入准备好状态，开始读描述符
Get_order	4’h2	向内存发出读描述符请求，等待内存应答

Read_order	4'h3	内存接收读描述符请求，正在执行读操作
Finish_order_end	4'h4	内存读完 dma 描述符
R_ddr_wait	4'h5	Dma 向内存发出读数据请求，等待内存应答
Read_ddr	4'h6	内存接收 dma 读数据请求，正在执行读数据操作
Read_ddr_end	4'h7	内存完成 dma 的一次读数据请求
Read_dev	4'h8	Dma 进入读设备状态
Read_dev_end	4'h9	设备返回读数据，结束此次读设备请求
Read_step_end	4'ha	结束一次 step 操作，step times 减 1

13.3.7 ORDER_ADDR_IN

中文名： 在 confreg 模块，存放第一个 DMA 描述符地址寄存器
寄存器位宽： [31：0]
地址： 0xbfd01160
复位值： 0x00000000

位域	位域名称	位宽	访问	描述
31:6	Ask_addr	26	R/W	第一个 DMA 描述符的地址，地址双字对齐
5	保留	1		
4	dma_stop	1	R/W	用户请求停止 DMA 操作; 完成当前数据读写操作后，停止操作
3	dma_start	1	R/W	可以开始读描述符链的第一个 DMA 描述符; 当第一个描述符相关的寄存器读回后，该位清零
2	Ask_valid	1	R/W	用户请求将当前 DMA 操作的相关信息写回到指定的内存地址; 当用户写回 DMA 操作相关信息后，该位清零。
1:0	Dev_num	2	R	2'b00 nand flash 2'b01 AC97 read device 2'b10 AC97 write device

说明：每次 DMA 操作，DMA_ORDER_ADDR 寄存器存放的下个描述符的地址和有效位。第一个描述符的地址存在 ORDER_ADDR_IN 寄存器中，该寄存器在 confreg 模块维护，由 CPU 来配置。如果 ask_valid=1，表示 CPU 要侦听 DMA 操作，此时要将 DMA 控制器寄存器的值写回到 ask_addr 指向的内存中。如果 dma_start=1，表示开始 DMA 操作，DMA 先从 ask_addr 指向的内存地址读描述符，然后根据描述符的信息开始执行 DMA 操作。

14 UART

14.1 概述

1B集成了 12 个UART核，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。

14.2 UART 控制器结构

UART 控制器有发送和接收模块（Transmitter and Receiver）、MODEM 模块、中断仲裁模块（Interrupt Arbitrator）、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：

- **发送和接收模块：**负责处理数据帧的发送和接收。发送模块是将 FIFO 发送队列中的数据按照设定的格式把并行数据转换为串行数据帧，并通过发送端口送出去。接收模块则监视接收端信号，一旦出现有效开始位，就进行接收，并实现将接收到的异步串行数据帧转换为并行数据，存入 FIFO 接收队列中，同时检查数据帧格式是否有错。UART 的帧结构是通过行控制寄存器（LCR）设置的，发送和接收器的状态被保存在行状态寄存器（LSR）中
- **MODEM 模块：**MODEM 控制寄存器（MCR）控制输出信号 DTR 和 RTS 的状态。MODEM 控制模块监视输入信号 DCD,CTS,DSR 和 RI 的线路状态，并将这些信号的状态记录在 MODEM 状态寄存器（MSR）的相对应位中。
- **中断仲裁模块：**当任何一种中断条件被满足，并且在中断使能寄存器（IER）中相应位置 1，那么 UART 的中断请求信号 UAT_INT 被置为有效状态。为了减少和外部软件的交互，UART 把中断分为四个级别，并且在中断标识寄存器（IIR）中标识这些中断。四个级别的中断按优先级级别由高到低的排列顺序为，接收线路状态中断；接收数据准备好中断；传送拥有寄存器为空中断；MODEM 状态中断。
- **访问寄存器模块：**当 UART 模块被选中时，CPU 可通过读或写操作访问被地址线选中的寄存器。

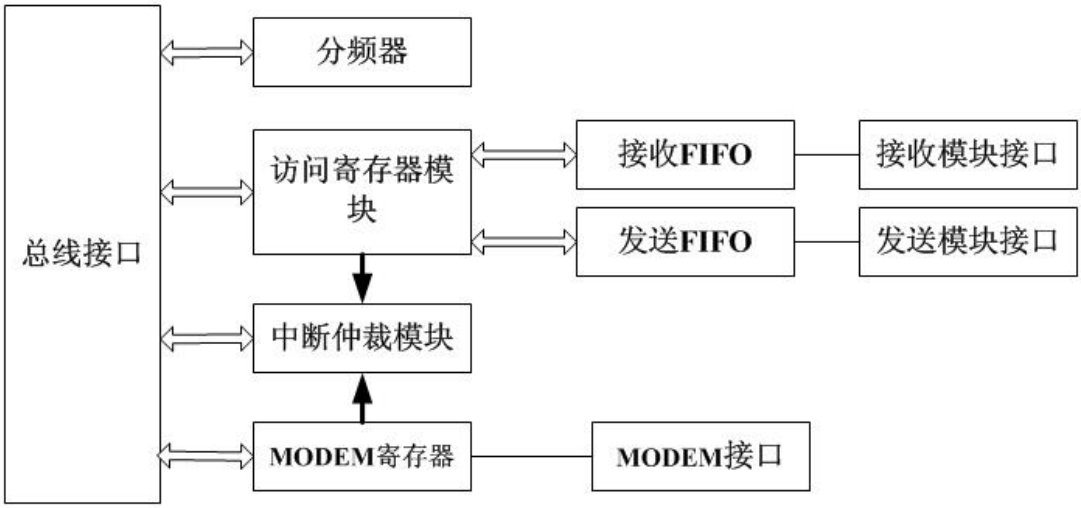


图 14-1　UART 控制器结构

14.3　UART 控制器寄存器

1B 内共 12 个并行工作的 UART 接口，其功能寄存器完全一样，只是访问基址不一样。

UART0　寄存器物理地址基址为　0xbfE40000。
UART0_1 寄存器物理地址基址为 0xbfE41000。
UART0_2 寄存器物理地址基址为 0xbfE42000。
UART0_3 寄存器物理地址基址为 0xbfE43000。
UART1　寄存器物理地址基址为　0xbfE44000。
UART1_1 寄存器物理地址基址为 0xbfE45000。
UART1_2 寄存器物理地址基址为 0xbfE46000。
UART1_3 寄存器物理地址基址为 0xbfE47000。
UART2　寄存器物理地址基址为 0xbfE48000。
UART3　寄存器物理地址基址为 0xbfE4c000。
UART4　寄存器物理地址基址为 0xbfE6c000。
UART5　寄存器物理地址基址为 0xbfE7c000。

UART0 和 UART1 都实现了一分四功能，UART0 有 8 个 PAD；UART1 只有 4 个 PAD，同时利用 CAN0 和 CAN1 的 4 个 PAD。所以在 CAN0/CAN1 不用的时候，1B 最多可以提供出来 12 个两线 UART。

0XBFE7_8038:uart_split

位域	位域名称	访问	描述
1	Uart1_split	R/W	UART1 被分成四个独立两线 UART
0	Uart0_split	R/W	UART0 被分成四个独立两线 UART

00XBFD0_0424: uart1_use_can

5	UART1_3_USE_CAN1	UART1_3 利用 CAN1 实现
4	UART1_2_USE_CAN0	UART1_2 利用 CAN0 实现

14.3.1 数据寄存器（DAT）

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

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

14.3.2 中断使能寄存器（IER）

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

位域	位域名称	位宽	访问	描述
7:4	Reserved	4	RW	保留
3	IME	1	RW	Modem 状态中断使能 ‘0’– 关闭 ‘1’ – 打开
2	ILE	1	RW	接收器线路状态中断使能 ‘0’ – 关闭 ‘1’– 打开
1	ITxE	1	RW	传输保存寄存器为空中断使能 ‘0’ – 关闭 ‘1’– 打开
0	IRxE	1	RW	接收有效数据中断使能 ‘0’– 关闭 ‘1’ – 打开

14.3.3 中断标识寄存器（IIR）

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

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

中断控制功能表

Bit 3	Bit 2	Bit 1	优先级	中断类型	中断源	中断复位控制
0	1	1	1st	接收线路状态	奇偶、溢出或帧错误，或打断中断	读 LSR
0	1	0	2nd	接收到有效数据	FIFO 的字符个数达到 trigger 的水平	FIFO 的字符个数低于 trigger 的值
1	1	0	2nd	接收超时	在 FIFO 至少有一个字符，但在 4 个字符时间内没有任何操作，包括读和写操作	读接收 FIFO

0	0	1	3rd	传输保存寄存器为空	传输保存寄存器为空	写数据到 THR 或者多 IIR
0	0	0	4th	Modem 状态	CTS, DSR, RI or DCD.	读 MSR

14.3.4 FIFO 控制寄存器（FCR）

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

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

14.3.5 线路控制寄存器（LCR）

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

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

				‘0’ – 1 个停止位 ‘1’ – 在 5 位字符长度时是 1.5 个停止位，其他长度是 2 个停止位
1:0	bec	2	RW	设定每个字符的位数 ‘00’ – 5 位 ‘01’ – 6 位 ‘10’ – 7 位 ‘11’ – 8 位

14.3.6 MODEM 控制寄存器（MCR）

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

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

14.3.7 线路状态寄存器（LSR）

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

位域	位域名称	位宽	访问	描述
7	ERROR	1	R	错误表示位 ‘1’ – 至少有奇偶校验位错误，帧错误或打断中断的一个。 ‘0’ – 没有错误
6	TE	1	R	传输为空表示位 ‘1’ – 传输 FIFO 和传输移位寄存器都为空。给传输 FIFO 写数据时清零 ‘0’ – 有数据
5	TFE	1	R	传输 FIFO 位空表示位 ‘1’ – 当前传输 FIFO 为空，给传输 FIFO 写数据时清零 ‘0’ – 有数据

位域	位域名称	位宽	访问	描述
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 ‘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。

14.3.8 MODEM 状态寄存器 （MSR）

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

位域	位域名称	位宽	访问	描述
7	CDCD	1	R	DCD 输入值的反，或者在回环模式中连到 Out2
6	CRI	1	R	RI 输入值的反，或者在回环模式中连到 OUT1
5	CDSR	1	R	DSR 输入值的反，或者在回环模式中连到 DTR
4	CCTS	1	R	CTS 输入值的反，或者在回环模式中连到 RTS
3	DDCD	1	R	DDCD 指示位
2	TERI	1	R	RI 边沿检测。RI 状态从低到高变化
1	DDSR	1	R	DDSR 指示位
0	DCTS	1	R	DCTS 指示位

14.3.9 分频锁存器

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

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

中文名： 分频锁存器 2

寄存器位宽： [7：0]
偏移量： 0x01
复位值： 0x00

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

15 CAN

15.1 概述

1B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。

本芯片集成了两个CAN 总线控制器，引脚复用 GPIO[41:38]，对应关系如下表。

CAN0_RX	GPIO66
CAN0_TX	GPIO67
CAN1_RX	GPIO68
CAN1_TX	GPIO69

两个 CAN 总线控制器的中断连接到中断控制的第一组寄存器中，其中 can0 的中断对应 bit6，can1 的中断对应 bit7。参考第 12 章的说明。

CAN0 总线控制器的寄存器基地址为 0xbfe50000 开始的 16KB;

CAN1 总线控制器的寄存器基地址为 0xbfe54000 开始的 16KB

15.2 CAN 控制器结构

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

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

为接收FIFO（长 64 字节）的一个窗口可被CPU 访问

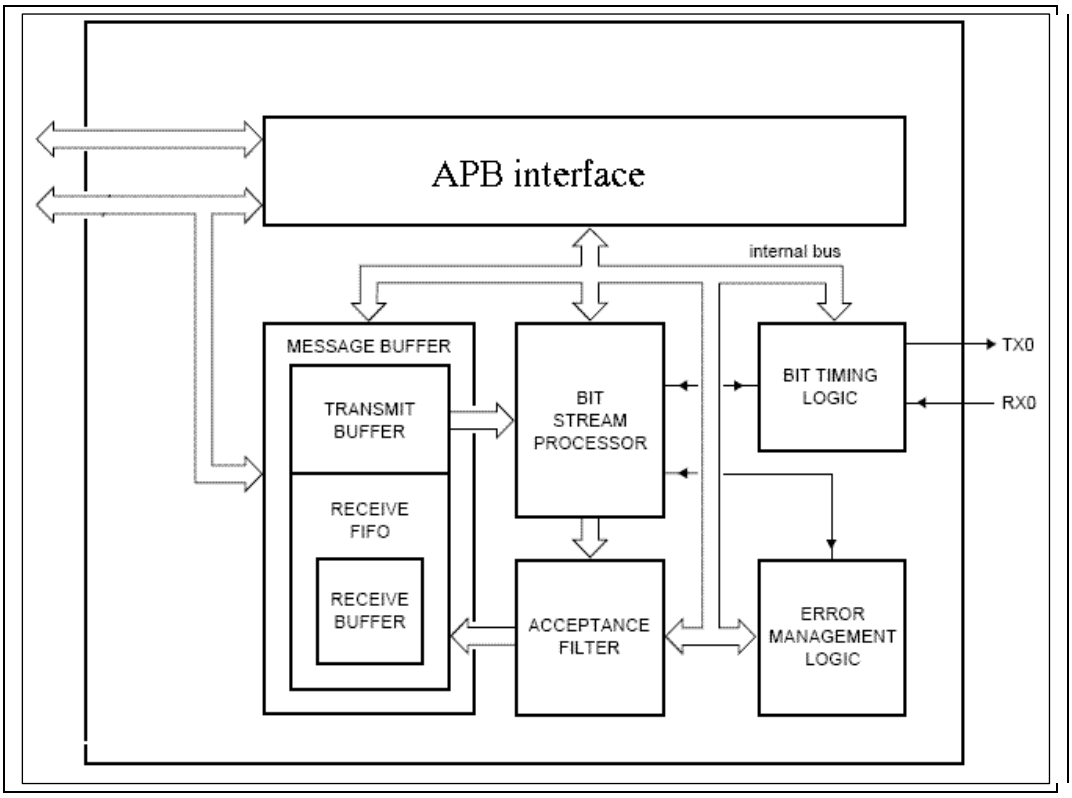


图 15-1 CAN 主控制器结构

本模块包括两个功能完全相同，能够同时工作的 CAN 模块；他们寄存器完全一样，所不同的是他们的物理地址基址。他们的地址基址分别为 ：

CAN #0 的物理地址基址为 0xbf004400；

CAN #1 的物理地址基址为 0xbf004300。

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

15.3 标准模式

15.3.1 标准模式地址表

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

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

CAN 地址	段	工作模式		复位模式	
		读	写	读	写

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

15.3.2 控制寄存器（CR）

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

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

复位请求位被设为 0 后控制器将会等待

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

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

化总线造成的。

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

15.3.3 命令寄存器（CMR）

中文名： 命令寄存器

寄存器位宽： [7：0]

偏移量： 0x01

复位值： 0x00

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

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

15.3.4 状态寄存器（SR）

中文名： 状态寄存器

寄存器位宽： [7：0]

偏移量： 0x02

复位值： 0x00

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

15.3.5 中断寄存器（IR）

中文名： 中断寄存器

寄存器位宽： [7：0]

偏移量： 0x03

复位值： 0x00

位域	位域名称	位宽	访问	描述
7：5	Reserved	1	R	保留
4	WUI	1	R	唤醒中断

3	DOI	1	R	数据溢出中断
2	EI	1	R	错误中断
1	TI	1	R	发送中断
0	RI	1	R	接收中断

15.3.6 验收代码寄存器（ACR）

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

在复位情况下，该寄存器是可以读写的。

位域	位域名称	位宽	访问	描述
7:0	AC	8	RW	ID 验收代码

15.3.7 验收屏蔽寄存器（AMR）

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

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

位域	位域名称	位宽	访问	描述
7:0	AM	8	RW	ID 屏蔽位

15.3.8 发送缓冲区列表

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

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

15.3.9 接收缓冲区列表

接收缓冲区的配置和发送缓冲区的一样，只是地址变为 20－29。

15.4 扩展模式

15.4.1 扩展模式地址表

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

15.4.2 模式寄存器（MOD）

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

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

复位请求位被设为 0 后控制器将会等待

a）一个总线空闲信号（11 个弱势位），如果前一次复位请求是硬件复位或CPU 初始复位。
b）128 个总线空闲，如果前一次复位请求是CAN控制器在重新进入总线开启模式前初始化总线造成的。

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

15.4.3 命令寄存器（CMR）

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

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

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

15.4.4 状态寄存器（SR）

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

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

15.4.5 中断寄存器（IR）

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

位域	位域名称	位宽	访问	描述

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

15.4.6 中断使能寄存器（IER）

中文名： 中断使能寄存器

寄存器位宽： [7：0]

偏移量： 0x04

复位值： 0x00

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

15.4.7 仲裁丢失捕捉寄存器（IER）

中文名： 仲裁丢失捕捉寄存器

寄存器位宽： [7：0]

偏移量： 0xB

复位值： 0x00

位域	位域名称	位宽	访问	描述
7：5	—	3	R	保留
4	BITNO4	1	R	第四位
3	BITNO3	1	R	第三位
2	BITNO2	1	R	第二位
1	BITNO1	1	R	第一位
0	BITNO0	1	R	第零位

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

15.4.8 错误警报限制寄存器（EMLR）

中文名： 错误警报限制寄存器

寄存器位宽： [7：0]

偏移量： 0xD

复位值： 0x60

位域	位域名称	位宽	访问	描述
7：0	EML	8	RW	错误警报阀值

15.4.9 RX 错误计数寄存器（RXERR）

中文名： RX 错误计数寄存器

寄存器位宽： [7：0]

偏移量： 0xE
复位值： 0x60

位域	位域名称	位宽	访问	描述
7：0	RXERR	8	R	接收错误计数

15.4.10 TX 错误计数寄存器（TXERR）

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

位域	位域名称	位宽	访问	描述
7：0	TXERR	8	R	发送错误计数

15.4.11 验收滤波器

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

15.4.12 RX 信息计数寄存器（RMCR）

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

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

15.5 公共寄存器

15.5.1 总线定时寄存器 0（BTR0）

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

注：在复位模式是可以读写的，工作模式是只读的

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

15.5.2 总线定时寄存器 1（BTR1）

中文名： 总线定时寄存器 1
寄存器位宽： [7：0]
偏移量： 0x07
复位值： 0x00

位域	位域名称	位宽	访问	描述

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

15.5.3 输出控制寄存器（OCR）

中文名： 输出控制寄存器

寄存器位宽： [7：0]

偏移量： 0x08

复位值： 0x00

位域	位域名称	位宽	访问	描述
7：0	OCR	8	RW	保留

16 AC97

16.1 AC97 结构描述

在系统里一个 AC97 应用系统如图 16-1 所示。在一个片上系统中，与 AC97 控制器相连的有 3 部分：一是外设总线，接收来自微处理器的控制信息以及配置信息；二是 AC97 Codec，多媒体数字信号编解码器，该解码器对 PCM 信号进行调制，输出人耳接受的模拟声音或者把真实的声音转换为 PCM 信号，转换通过 D/A 转换器实现；三是 DMA 引擎，通过 DMA 的方式写或读 AC97 控制器内部的 FIFO，实现 PCM 音频数据的不间断操作。DMA 是通过微处理器配置的，从处理器设定的内存区域搬运数据给 FIFO 或者把 FIFO 的数据搬运到设定的内存区域。

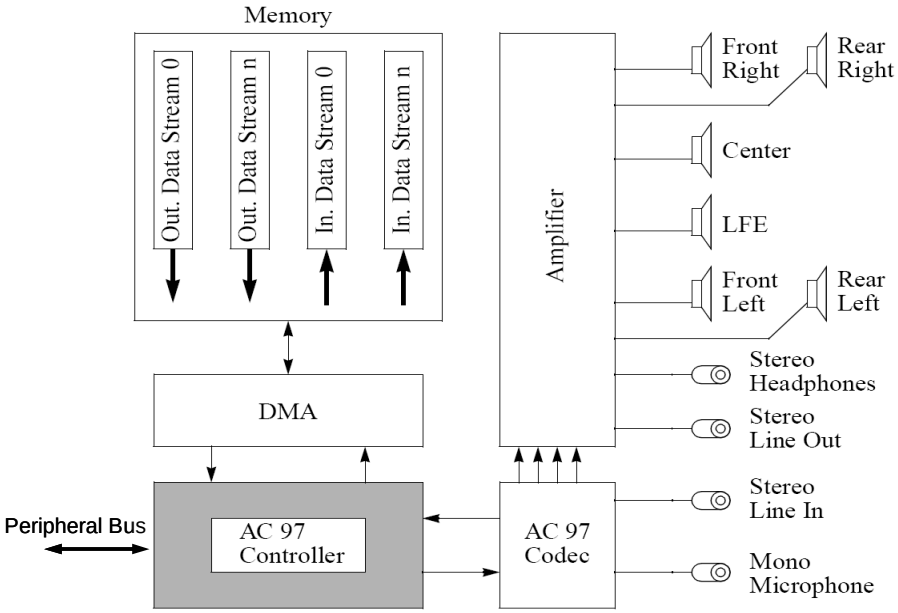


图 16-1 AC97 应用系统

16.2 AC97 控制器寄存器

本模块寄存器物理地址基址为 0xbfE74000。

寄存器名	宽度	偏移量	描述
CSR	2	0x00	配置状态寄存器
OCC0	24	0x04	输出通道配置寄存器 0
OCC1	24	0x08	保留
OCC2	24	0x0c	保留
ICC	24	0x10	输入通道配置寄存器
CODEC_ID	32	0x14	Codec ID 寄存器

寄存器名	宽度	偏移量	描述
CRAC	32	0x18	Codec 寄存器访问命令
OC0	20	0x20	输出声道 0
OC1	20	0x24	输出声道 1
OC2	20	0x28	保留
OC3	20	0x2c	保留
OC4	20	0x30	保留
OC5	20	0x34	保留
OC6	20	0x38	保留
OC7	20	0x3c	保留
OC8	20	0x40	保留
IC0	20	0x44	保留
IC1	20	0x48	保留
IC2	20	0x4c	输入声道 2
INTRAW	32	0x54	中断状态寄存器
INTM	32	0x58	中断掩膜
INTS	32	0x5c	保留

16.2.1 CSR 寄存器

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

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

16.2.2 OCC 寄存器

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

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

16.2.3 ICC 寄存器

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

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

16.2.4 （输入输出）通道寄存器配置

中文名： 通道配置寄存器
寄存器位宽： [31：0]
偏移量： 0x14
复位值： 0x00410000

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

16.2.5 Codec 寄存器访问命令

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

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

16.2.6 中断状态寄存器/中断掩膜寄存器

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

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

16.2.7 中断状态/清除寄存器

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

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

16.2.8 OC 中断清除寄存器

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

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

16.2.9 IC 中断清除寄存器

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

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

16.2.10 CODEC WRITE 中断清除寄存器

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

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

16.2.11 CODEC READ 中断清除寄存器

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

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

17 I2C

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I^2C接口，主要用于实现两个器件之间数据的交换。I^2C 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。1B芯片共集成 3 路I2C接口，其中第二路和第三路分别通过CAN0 和CAN1 复用实现。复用配置参加 23 章MUX寄存器小节。

I2C_SDA1	CAN0_RX
I2C_SCL1	CAN0_TX
I2C_SDA2	CAN1_RX
I2C_SCL2	CAN1_TX

17.2 I^2C 控制器结构

I^2C 主控制器的结构，主要模块有，时钟发生器（Clock Generator）、字节命令控制器（Byte Command Controller）、位命令控制器（Bit Command controller）、数据移位寄存器（Data Shift Register）。其余为LPB总线接口和一些寄存器。

1. **时钟发生器模块：**产生分频时钟，同步位命令的工作。
2. **字节命令控制器模块：**将一个命令解释为按字节操作的时序，即把字节操作分解为位操作。
3. **位命令控制器模块：**进行实际数据的传输，以及位命令信号产生。
4. **数据移位寄存器模块：**串行数据移位。

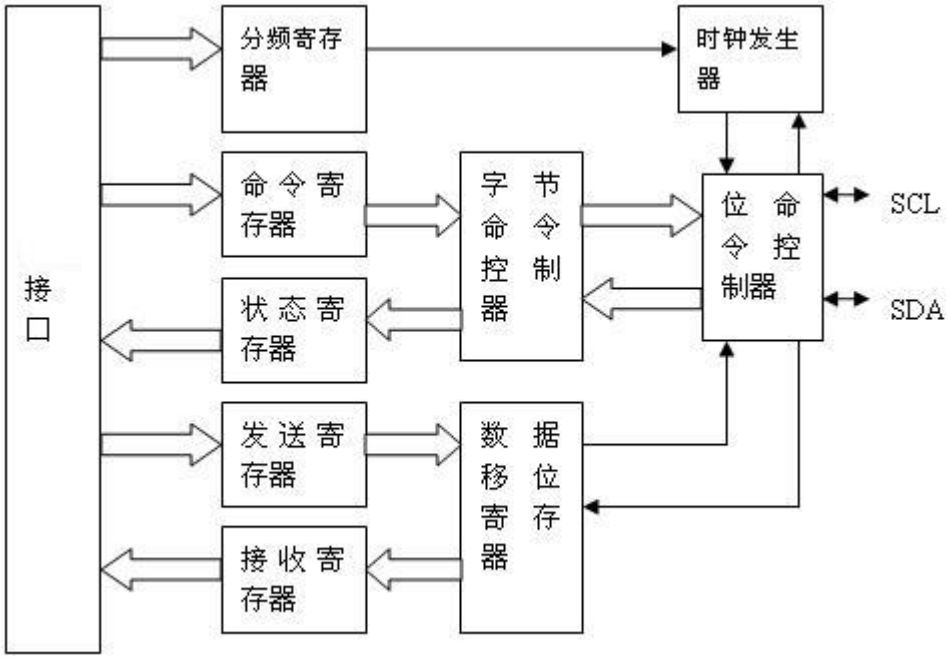


图 17-1 I²C 主控制器结构

17.3 I²C 控制器寄存器说明

I2C-0 模块寄存器物理地址基址为：0xbfe58000,地址空间 16KB。
I2C-1 模块寄存器物理地址基址为：0xbfe68000,地址空间 16KB。
I2C-2 模块寄存器物理地址基址为：0xbfe70000,地址空间 16KB。

17.3.1 分频锁存器低字节寄存器（PRERlo）

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

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

17.3.2 分频锁存器高字节寄存器（PRERhi）

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

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

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

Prcescale = clock_a/(5*clock_s)-1

17.3.3 控制寄存器（CTR）

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

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

17.3.4 发送数据寄存器（TXR）

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

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

17.3.5 接受数据寄存器（RXR）

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

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

17.3.6 命令控制寄存器（CR）

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

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

都是在 I^2C 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’。

17.3.7 状态寄存器（SR）

中文名： 状态寄存器

寄存器位宽： [7：0]
偏移量： 0x04
复位值： 0x00

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

18.1 概述

1B 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 (pwm_o)．系统时钟高达 100MHz，计数寄存器和参考寄存器均 24 位数据宽度，使得芯片非常适合高档电机的控制。

四路 PWM 控制器系统的基地址具体如下：

表 18-1 四路控制器描述

名称	基地址（Base）	中断号
PWM0	0XBFE5:C000	18
PWM1	0XBFE5:C010	19
PWM2	0XBFE5:C020	20
PWM3	0XBFE5:C030	21

每路控制器共有四个控制寄存器，具体描述如下：

表 18-2 控制寄存器描述

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

18.2 PWM 寄存器说明

（1） 实现脉冲宽度功能

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

表 18-3 主计数器设置

位域	访问	复位值	说明
23：0	R/W	0x0	主计数器

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

表 18-4 高脉冲计数器设置

位域	访问	复位值	说明
23：0	R/W	0x0	高脉冲计数器

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

表 18-5 低脉冲计数器设置

位域	访问	复位值	说明
23：0	R/W	0x0	低脉冲计数器

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

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

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

表 18-6 控制寄存器设置

位域	访问	复位值	说明
0	R/W	0	EN，主计数器使能位 置 1 时：CNTR 用来计数 置 0 时：CNTR 停止计数
2：1	Reserved	2’b0	预留
3	R/W	0	OE,脉冲输出使能控制位,低有效 置 0 时：脉冲输出使能 置 1 时：脉冲输出屏蔽
4	R/W	0	SINGLE，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	R/W	0	INTE，中断使能位 置 1 时：当 CNTR 计数到 LRC 和 CNTR 后送中断 置 0 时：不产生中断
6	R/W	0	INT，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	R/W	0	CNTR_RST,使得 CNTR 计数器清零 置 1 时：CNTR 计数器清零 置 0 时：CNTR 计数器正常工作

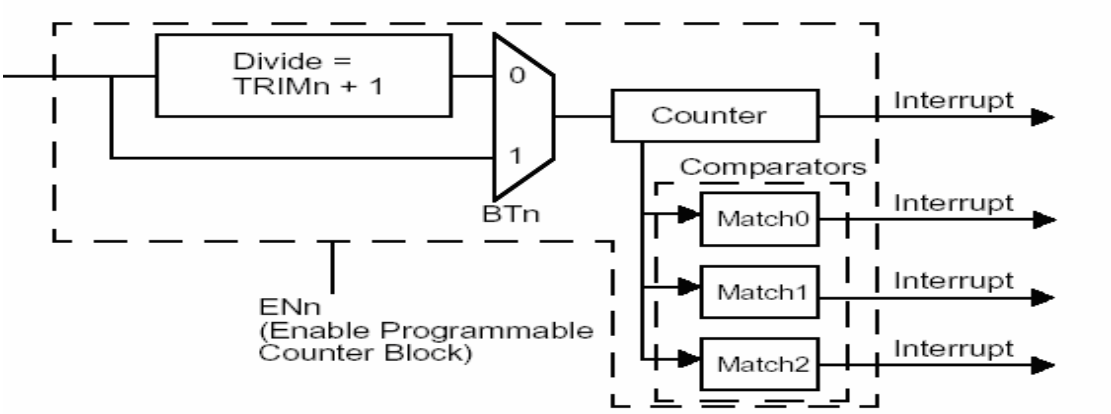
19 RTC

19.1 概述

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

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

RTC 单元由计数器和定时器组成，其构架如下图所示：



19.2 RTC 电源配置

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

19.3 寄存器描述

RTC 模块寄存器位于 0xbfe64000——0xbfe67fff 的 16KB 地址空间内，其基地址为 0xbfe64000，所有寄存器位宽均为 32 位。

19.3.1 寄存器地址列表

名称	地址	位宽	RW	描述	复位值
sys_toytrim	0xbfE64020	32	RW	对 32.768kHz 的分频系数（计数器时钟）	
sys_toywrite0	0xbfE64024	32	W	TOY 低 32 位数值写入	

sys_toywrite1	0xbfE64028	32	W	TOY 高 32 位数值写入	
sys_toyread0	0xbfE6402C	32	R	TOY 低 32 位数值读出	
sys_toyread1	0xbfE64030	32	R	TOY 高 32 位数值读出	
sys_toymatch0	0xbfE64034	32	RW	TOY 定时中断 0	
sys_toymatch1	0xbfE64038	32	RW	TOY 定时中断 1	
sys_toymatch2	0xbfE6403C	32	RW	TOY 定时中断 2	
sys_rtcctrl	0xbfE64040	32	RW	TOY 和 RTC 控制寄存器	
sys_rtctrim	0xbfE64060	32	RW	对 32.768kHz 的分频系数（定时器时钟）	
sys_rtcwrite0	0xbfE64064	32	R	RTC 定时计数写入	
sys_rtcread0	0xbfE64068	32	W	RTC 定时计数读出	
sys_rtcmatch0	0xbfE6406C	32	RW	RTC 时钟定时中断 0	
sys_rtcmatch1	0xbfE64070	32	RW	RTC 时钟定时中断 1	
sys_rtcmatch2	0xbfE64074	32	RW	RTC 时钟定时中断 2	

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

19.3.2 SYS_TOYWRITE0

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

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

19.3.3 SYS_TOYWRITE1

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

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

19.3.4 SYS_TOYMATCH0/1/2

中文名： TOY 计数器中断寄存器 0/1/2
寄存器位宽： [31：0]

偏移量： 0x34/38/3C
复位值： 0x00000000

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

19.3.5 SYS_RTCCTRL

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

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

5	32S	R	0	0：32.768k 晶振不工作； 1：32.768k 晶振正常工作。
4	TTS	R	0	Sys_toytrim 写状态
3	TM2	R	0	Sys_toymatch2 写状态
2	TM1	R	0	Sys_toymatch1 写状态
1	TM0	R	0	Sys_toymatch0 写状态
0	TS	R	0	Sys_toywrite 写状态

19.3.6 SYS_RTCMATCH0/1/2

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

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

20 NAND

20.1 NAND 控制器结构描述

NAND FLASH 控制器最大支持 32GB FLASH 的容量，芯片最多支持 4 个片选和 4 个 RDY 信号。

20.2 NAND 控制器寄存器配置描述

NAND 内部的寄存器的设置如下：

地址	寄存器名称
BFE7_8000	NAND_CMD
BFE7_8004	ADDR_L
BFE7_8008	ADDR_H
BFE7_800C	NAND_TIMING
BFE7_8010	ID_L
BFE7_8014	STATUS & ID_H
BFE7_8018	NAND_PARAMETER
BFE7_801C	NAND_OP_NUM
BFE7_8020	CS_RDY_MAP
BFE7_8040	DMA access address

20.2.1 NAND_CMD（地址：BFE7_8000）

位	位域名	读写	描述
31:11		R/-	Reserved
23：20	NAND_CE	R/-	外部 NAND 芯片片选情况
19：16	NAND_RDY	R/-	外部 NAND 芯片 RDY 情况
15：11		R/-	Reserved
10	done	R/-	操作完成
9	Spare	R/W	操作发生在 NAND 的 SPARE 区
8	Main	R/W	操作发生在 NAND 的 MAIN 区
7	Read status	R/W	读 NAND 的状态
6	Reset	R/W	Nand 复位操作
5	read id	R/W	读 ID 操作
4	blocks erase	R/W	连续擦除标志，缺省 0；1 有效，连续擦擦块的数目由 nand_op_num 决定
3	erase operation	R/W	擦除操作
2	write operation	R/W	写操作
1	read operation	R/W	读操作
0	command valid	R/W	命令有效

20.2.2 ADDR_L（地址：BFE7_8004）

位	位域名	读写	描述
31:0	Nand_address[31:0]	R/W	读、写、擦除操作起始地址低 32 位

20.2.3 ADDR_H（地址：BFE7_8008）

位	位域名	读写	描述
31:8		R/-	Reserved
7:0	Nand_address[39:32]	R/W	读、写、擦除操作起始地址高 8 位

20.2.4 NAND_TIMING（地址：BFE7_800C）

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

20.2.5 ID_L（地址：BFE7_8010）

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

20.2.6 STATUS & ID_H（地址：BFE7_8014）

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

20.2.7 NAND_PARAMETER（地址：BFE7_8018）

位	位域名	读写	描述
31:20		R/-	Reserved
19:16	外部颗粒容量大小	R/W	1：2Gb 2：4Gb 3：8Gb 4：16Gb 5：32Gb 6：64Gb 7：128Gb 8：256Gb
15:0		R/-	Reserved

20.2.8 NAND_OP_NUM（地址：BFE7_801C）

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

20.2.9 CS_RDY_MAP（地址：BFE7_8020）

位	位域名	读写	描述
31：28	NAND_RDY[3]	R/W	4’b0001:NAND_RDY_inter[0] 4’b0010:NAND_RDY_inter[1]

			4’b0100:NAND_RDY_inter[2] 4’b1000:NAND_RDY_inter[3]
27：24	NAND_CE[3]	R/W	4’b0001:NAND_CS_inter[0] 4’b0010:NAND_CS_inter[1] 4’b0100:NAND_CS_inter[2] 4’b1000:NAND_CS_inter[3]
23：20	NAND_RDY[2]	R/W	4’b0001:NAND_RDY_inter[0] 4’b0010:NAND_RDY_inter[1] 4’b0100:NAND_RDY_inter[2] 4’b1000:NAND_RDY_inter[3]
19：16	NAND_CS[2]	R/W	4’b0001:NAND_CS_inter[0] 4’b0010:NAND_CS_inter[1] 4’b0100:NAND_CS_inter[2] 4’b1000:NAND_CS_inter[3]
15：12	NAND_RDY[1]	R/W	4’b0001:NAND_RDY_inter[0] 4’b0010:NAND_RDY_inter[1] 4’b0100:NAND_RDY_inter[2] 4’b1000:NAND_RDY_inter[3]
11：8	NAND_CS[1]	R/W	4’b0001:NAND_CS_inter[0] 4’b0010:NAND_CS_inter[1] 4’b0100:NAND_CS_inter[2] 4’b1000:NAND_CS_inter[3]
7:0		R/-	Reserved

DMA 对 NAND 操作的地址空间是地址：0XBFE7_8040，读写通过内部 APB-WRITE 信号来区分。

20.3 NAND ADDR 说明

Define: nand_addr_in_flash_chip = {…A33,A32,A31,A30,A29,A28…….A1,A0};
nand_addr_config_by_software = 0xbfe78004[31:0];
spare_op = 0xbfe78000[9];
main_op = 0xbfe78000[8];

NAND 地址空间示例

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

A11 是区别 spare 区的唯一标志。
在控制器里，如果不访问 spare 区，A11 被硬件写 0；
在控制器里面访问 spare 区或者同时 main+spare 区，A11 通过软件配置实现；
所以:nand_addr_in_flash_chip = spare ? nand_addr_config_by_software :
{nand_addr_config_by_software[30:11],1’b0, nand_addr_config_by_software [10:0]};

如果仅仅访问 spare 区，需要配置 spare=1 & main_op=0 & 0xbfe78004[11]=1，也就是当软件配置的地址的第 11 位为 0，永远不能直接访问 spare 区;

Examples:

(spare_op = 1'b0 & main_op =1'b0) equal to (spare_op = 1'b0 & main_op =1'b1);

nand_addr_config_by_software =0x30: addr_in_nand_flash = 0x30

Data	0	0x30	.	0x7ff	0x800	0x830	0x83f	0x840	0xfff
Row0	NO_op	op	op	op	NO_op	NO_op	NO_op	NO_op	NO_op
Row1	op	op	op	op	NO_op	NO_op	NO_op	NO_op	NO_op
Row2	op	op	op	op	NO_op	NO_op	NO_op	NO_op	NO_op

spare_op=1'b1 & main_op=1'b0;(配置有问题，开始操作不在 spare 区，最好 0x***800 开始) nand_addr_config_by_software =0x30: addr_in_nand_flash = 0x30

Data	0	0x30	.	0x7ff	0x800	0x830	0x83f	0x840	0xfff
Row0	NO_op	op	op	op	op	op	op	NO_op	NO_op
Row1	NO_op	NO_op	NO_op	NO_op	op	op	op	NO_op	NO_op
Row2	NO_op	NO_op	NO_op	NO_op	op	op	op	NO_op	NO_op
Row3	NO_op	NO_op	NO_op	NO_op	op	op	op	NO_op	NO_op

spare_op = 1'b1 & main_op =1'b1;

nand_addr_config_by_software =0x30: addr_in_nand_flash = 0x30

Data	0	0x30	.	0x7ff	0x800	0x830	0x83f	0x840	0xfff
Row0	NO_op	op	op	op	op	op	op	NO_op	NO_op
Row1	op	op	op	op	op	op	op	NO_op	NO_op
Row2	op	op	op	op	op	op	op	NO_op	NO_op

(spare_op=1'b0 & main_op=1'b0), (equal to spare_op=1'b0 & main_op=1'b1);

nand_addr_config_by_software =0x830: addr_in_nand_flash = 0x1030(It is different!!!!)

Data	0	0x30	.	0x7ff	0x800	0x830	0x83f	0x840	0xfff
Row0	NO_op	NO_op	NO_op	NO_op	NO_op	NO_op	NO_op	NO_op	NO_op
Row1	NO_op	op	op	op	op	NO_op	NO_op	NO_op	NO_op
Row2	op	op	op	op	op	NO_op	NO_op	NO_op	NO_op
Row3	op	op	op	op	op	NO_op	NO_op	NO_op	NO_op

spare_op = 1'b1 and main_op =1'b0;

nand_addr_config_by_software =0x830: addr_in_nand_flash = 0x830

Data	0	0x30	.	0x7ff	0x800	0x830	0x83f	0x840	0xfff
Row0	NO_op	NO_op	NO_op	NO_op	NO_op	op	op	NO_op	NO_op
Row1	NO_op	NO_op	NO_op	NO_op	op	op	op	NO_op	NO_op
Row2	NO_op	NO_op	NO_op	NO_op	op	op	op	NO_op	NO_op

spare_op = 1'b1 & main_op =1'b1;

nand_addr_config_by_software =0x830: addr_in_nand_flash = 0x830

Data	0	0x30	.	0x7ff	0x800	0x830	0x83f	0x840	0xfff

Row0	NO_op	NO_op	NO_op	NO_op	NO_op	op	op	NO_op	NO_op
Row1	op	op	op	op	op	op	op	NO_op	NO_op
Row2	op	op	op	op	op	op	op	NO_op	NO_op
Row3	op	op	op	op	op	op	op	NO_op	NO_op

20.4 Nand-flash 读写操作举例：

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

21 WATCHDOG

21.1 概述

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

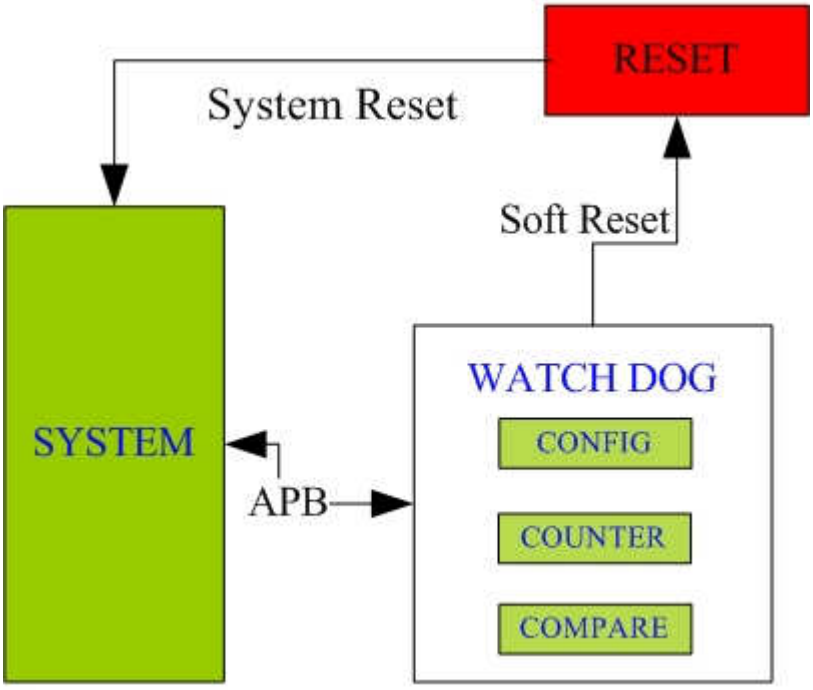


图 21-1 看门狗的结构图

21.2 WATCH DOG 寄存器描述

看门狗逻辑可编程寄存器主要有三个，这些寄存器描述如下：

21.2.1 WDT_EN 地址：（0XBFE5_C060）

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

21.2.2 WDT_SET（地址：0XBFE5_C064）

位	位域名	读写	描述
31:1			Reserved
0	WDT_SET	R/W	看门狗中计数器设置

21.2.3 WDT_timer（地址：0XBFE5_C068）

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

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

22 Clock Management

22.1 Clock 模块结构描述

时钟模块用来产生系统主要的三个时钟：CPU_clk、DDR_clk 和 DC_clk。系统中集成了一个 PLL，PLL 在系统 RESET 时从外部 PAD 的状态获取初始配置，该 PLL 在进入系统后可以再次配置。PLL 产生一个高频输出 PLL_clk，系统需要的 CPU_clk、DDR_clk 和 DC_clk 均由此高频输出时钟分频而来。其工作结构图如下：

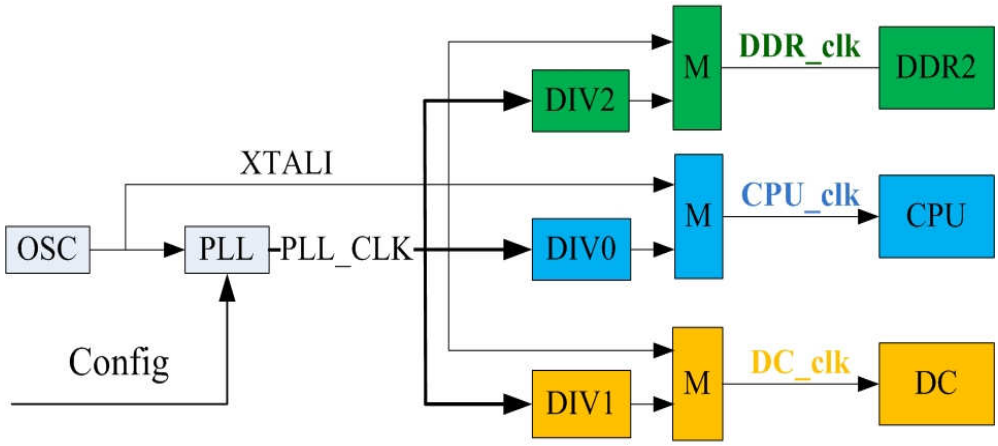


图 22-1 1B 时钟产生模块

22.2 Clock 配置描述

在系统 RESET 状态下，时钟模块利用外部 PAD 状态选择 PLL 的配置。具体如下表所示：

表 22-1

PAD	描述	对 PLL 配置作用
NAND_D[5:0]	PLL 输出频率控制	PLL 输出频率： (NAND_D[5:0]+12)*33/2Mhz
NAND_D6	CPU 和 DDR 频率通路选择，系统复位启动过程中，CPU 和 DDR 频率相同。	CPU_clk 和 DDR_clk 1：33Mhz 0：(NAND_D[5:0]+12)*33/4Mhz
NAND_D7		

系统启动后，软件可以精确配置 PLL 的频率，这些参数如下表所示：

PLL_FREQ:基地址 0XBFE7_8030:

Bit 位	寄存器描述
[17:0]	PLL 输出频率： (12+PLL_FREQ[5:0]+ (PLL_FREQ[17:8]/1024))*33/2Mhz

同时，软件可以对内部的 CPU/DDR/DC 时钟频率单独配置。配置过程：对应时钟 Bypass 位置 1，让其切换到 33Mhz 的外部输入，然后让对应逻辑 RST，最后配置需要分频的倍数，以产生目标时钟。恢复过程：先把 Bypass 位清零，让对应时钟恢

复到分频的目标时钟。时钟频率的配置过程中使用了 Glitch free 的电路，确保系统能够正常稳定工作。对应寄存器详细描述如下：

PLL_DIV_PARAM: 基地址 0XBFE7_8034:

Bit 位	寄存器描述	读写
31	DC_DIV 使能	R/W
30:26	DC_DIV	R/W
25	CPU_DIV 使能	R/W
24:20	CPU_DIV	R/W
19	DDR_DIV 使能	R/W
18:14	DDR_DIV	R/W
13	DC_BYPASS 使能	R/W
12	DC_BYPASS	R/W
11	DDR_BYPASS 使能	R/W
10	DDR_BYPASS	R/W
9	CPU_BYPASS 使能	R/W
8	CPU_BYPASS	R/W
7：6	保留	保留
5	DC_RST 使能	R/W
4	DC_RST	R/W
3	DDR_RST 使能	R/W
2	DDR_RST	R/W
1	CPU_RST 使能	R/W
0	CPU_RST	R/W

For external configuration D7:D0 = 0X80, OSC_CLK=33Mhz

CPU_clk = 100MHz, DDR_clk = 100MHz, dc_clk = 22Mhz

An example:

0xbfe7_8034:0x3f00: cpu/ddr/dc clock to 33Mhz in bypass mode

0xbfe7_8030:0xc : pll_clk to 400MHz

0xbfe7_8034:0x8ff00, ddr_clk_div enable, and ddr_div_num =3, still in bypass mode, ddr_clk =33Mhz, cpu_div_num and dc_div_num use the default value to 2 and 9.

0xbfe7_8034:0x2a00, cpu_clk= 400/2=200MHz ,dc_clk=400/9=44.4MHz , and ddr_clk = 400/3=133Mhz

0xbfe7_8034:0x0, clock config for cpu/ddr/dc is okay.

23 GPIO and MUX

23.1 GPIO 结构描述

GPIO 给芯片的设计和应用提供了灵活外部接口；部分 PAD 通过 MUX 实现，从而在 BGA256 封装情况下提供丰富的外部功能。

PAD	PAD 描述	GPIO	第一复用	第二复用	第三复用	UART
PWM0	PWM0 波形输出	GPIO00	NAND_CS*	SPI1_CSN[1]	UART0_RX	UART0_RX
PWM1	PWM1 波形输出	GPIO01	NAND_RDY*	SPI1_CSN[2]	UART0_TX	UART0_TX
PWM2	PWM2 波形输出	GPIO02	NAND_CS*		UART0_CTS	UART0_CTS
PWM3	PWM3 波形输出	GPIO03	NAND_RDY*		UART0_RTS	UART0_RTS
LCD_CLK	LCD 时钟	GPIO04				
LCD_VSYNC	LCD 列同步	GPIO05				
LCD_HSYNC	LCD 行同步	GPIO06				
LCD_EN	LCD 使能信号	GPIO07				
LCD_DAT_B0	LCD_BLUE0	GPIO08			UART1_RX	UART1_RX
LCD_DAT_B1	LCD_BLUE1	GPIO09				
LCD_DAT_B2	LCD_BLUE2	GPIO10				
LCD_DAT_B3	LCD_BLUE3	GPIO11				
LCD_DAT_B4	LCD_BLUE4	GPIO12				
LCD_DAT_G0	LCD_GREEN0	GPIO13			UART1_CTS	UART1_CTS
LCD_DAT_G1	LCD_GREEN1	GPIO14			UART1_RTS	UART1_RTS
LCD_DAT_G2	LCD_GREEN2	GPIO15				
LCD_DAT_G3	LCD_GREEN3	GPIO16				
LCD_DAT_G4	LCD_GREEN4	GPIO17				
LCD_DAT_G5	LCD_GREEN5	GPIO18				
LCD_DAT_R0	LCD_RED0	GPIO19			UART1_TX	UART1_TX
LCD_DAT_R1	LCD_RED1	GPIO20				
LCD_DAT_R2	LCD_RED2	GPIO21				
LCD_DAT_R3	LCD_RED3	GPIO22				
LCD_DAT_R4	LCD_RED4	GPIO23				
SPI0_CLK	SPI0 时钟	GPIO24				
SPI0_MISO	SPI0 主入从出	GPIO25				
SPI0_MOSI	SPI0 主出从入	GPIO26				
SPI0_CS0	SPI0 选通信号 0	GPIO27				
SPI0_CS1	SPI0 选通信号 1	GPIO28				
SPI0_CS2	SPI0 选通信号 2	GPIO29				
SPI0_CS3	SPI0 选通信号 3	GPIO30				
SCL	第一路 I2C 时钟	GPIO32				
SDA	第一路 I2C 数据	GPIO33				
AC97_SYNC	AC97 同步信号	GPIO34				

AC97_RST	AC97 复位信号	GPIO35			
AC97_DI	AC97 数据输入	GPIO36			
AC97_DO	AC97 数据输出	GPIO37			
CAN0_RX	CAN0 数据输入	GPIO38	SDA2	SPI1_CSN0	Uart1_DSR UART1_2RX
CAN0_TX	CAN0 数据输出	GPIO39	SCL2	SPI1_CLK	Uart1_DTR UART1_2TX
CAN1_RX	CAN1 数据输入	GPIO40	SDA3	SPI1_SD0	Uart1_DCD UART1_3RX
CAN1_TX	CAN1 数据输出	GPIO41	SCL3	SPI1_SDI	Uart1_RI UART1_3TX
UART0_RX	UART0 发送数据	GPIO42	LCD_DAT22	GMAC1_RCTL	UART0_0RX
UART0_TX	UART0 接收数据	GPIO43	LCD_DAT23	GMAC1_RX0	UART0_0TX
UART0_RTS	UART0 请求发送	GPIO44	LCD_DAT16	GMAC1_RX1	UART0_1TX
UART0_CTS	UART0 允许发送	GPIO45	LCD_DAT17	GMAC1_RX2	UART0_1RX
UART0_DSR	UART0 设备准备好	GPIO46	LCD_DAT18	GMAC1_RX3	UART0_2RX
UART0_DTR	UART0 终端准备好	GPIO47	LCD_DAT19		UART0_2TX
UART0_DCD	UART0 载波检测	GPIO48	LCD_DAT20	GMAC1_MDCK	UART0_3RX
UART0_RI	UART0 振铃提示	GPIO49	LCD_DAT21	GMAC1_MDIO	UART0_3TX
UART1_RX	UART1 接收数据	GPIO50		GMAC1_TX0	NAND_RDY* UART1_0RX
UART1_TX	UART1 发送数据	GPIO51		GMAC1_TX1	NAND_CS* UART1_0TX
UART1_RTS	UART1 请求发送	GPIO52		GMAC1_TX2	NAND_CS* UART1_1TX
UART1_CTS	UART1 允许发送	GPIO53		GMAC1_TX3	NAND_RDY* UART1_1RX
UART2_RX	UART2 接收数据	GPIO54			**UART2_RX**
UART2_TX	UART2 发送数据	GPIO55			**UART2_TX**
UART3_RX	UART3 接收数据	GPIO56			**UART3_RX**
UART3_TX	UART3 发送数据	GPIO57			**UART3_TX**
UART4_RX	UART4 接收数据	GPIO58			**UART4_RX**
UART4_TX	UART4 发送数据	GPIO59			**UART4_TX**
UART5_RX	UART5 接收数据	GPIO50			**UART5_RX**
UART5_TX	UART5 发送数据	GPIO51			**UART5_TX**

Note1：NAND_CS*/NAND_RDY*表示 NAND_CS1/2/3, NAND_RDY1/2/3 可以配置选择。

1B 中 UART0 有 8 个 PAD，UART1 有 4 个 PAD，UART2/3/4/5 各 2 个，UART 本身 PAD 共 20 个，UART 一共可以提供了 10 个两线 UART（最后一列红色+黑色）。

UART0 和 UART1 都实现了一分四功能，UART0 有 8 个 PAD；UART1 只有 4 个 PAD，这个时候利用了 CAN0 和 CAN1 的 4 个 PAD。所以在 CAN0/CAN1 不用的时候，1B 最多可以提供出来 12 个两线 UART（最后一列红色+黑色+绿色）。

当 GMAC1 复用 UART1&UART0 的时候，UART1 可以复用 CAN0/CAN1/LCD(0.5.6.11)实现全功能串口。此时 UART0 复用 PWM 实现两路 2 线串口，UART2/3/4/5 保留不动。在 GMAC1 复用的时候，1B 还可以有 1 个全功能和 6 个两线串口，或者 10 个两线串口。

23.2 GPIO 寄存器描述

偏移地址	位	寄存器	描述	读写特性
0xbfd010C0	32	GPIOCFG0	GPIO 配置寄存器 0	R/W

0xbfd010C4	32	GPIOCFG1	GPIO 配置寄存器 1	R/W
0xbfd010D0	32	GPIOOE0	GPIO 配置寄存器输入使能 0	R/W
0xbfd010D4	32	GPIOOE1	GPIO 配置寄存器输入使能 1	R/W
0xbfd010E0	32	GPIOIN0	GPIO 配置寄存器输入寄存器 0	R/W
0xbfd010E4	32	GPIOIN1	GPIO 配置寄存器输入寄存器 1	R/W
0xbfd010F0	32	GPIOOUT0	GPIO 配置寄存器输出寄存器 0	R/W
0xbfd010F4	32	GPIOOUT1	GPIO 配置寄存器输出寄存器 1	R/W

23.3 MUX 寄存器描述

GPIO_MUX_CTRL0 基地址 0XBFD0_0420,寄存器的描述如下：

位	描述	读写特性
31：26		保留
25	I2C_USE_CAN1	R/W
24	I2C_USE_CAN0	R/W
23	NAND3_USE_UART5	R/W
22	NAND3_USE_UART4	R/W
21	NAND3_USE_UART1_DAT	R/W
20	NAND3_USE_UART1_CTS	R/W
19	NAND3_USE_PWM23	R/W
18	NAND3_USE_PWM01	R/W
17	NAND2_USE_UART5	R/W
16	NAND2_USE_UART4	R/W
15	NAND2_USE_UART1_DAT	R/W
14	NAND2_USE_UART1_CTS	R/W
13	NAND2_USE_PWM23	R/W
12	NAND2_USE_PWM01	R/W
11	NAND1_USE_UART5	R/W
10	NAND1_USE_UART4	R/W
9	NAND1_USE_UART1_DAT	R/W
8	NAND1_USE_UART1_CTS	R/W
7	NAND1_USE_PWM23	R/W
6	NAND1_USE_PWM01	R/W
5		保留
4	GMAC1_USE_UART1	R/W
3	GMAC1_USE_UART0	R/W
2	LCD_USE_UART0_DAT	R/W
1	LCD_USE_UART15	R/W
0	LCD_USE_UART0	R/W

GPIO_MUX_CTRL1 基地址 0XBFD0_0424,寄存器的描述如下：

位	描述	读写特性
31	USB_reset	R/W
30:25		保留
24	SPI1_CS_USE_PWM01	R/W
23	SPI1_USE_CAN	R/W

22:13		保留
12	GMAC0_SHUT	R/W
11	USB_SHUT	R/W
10:6		保留
5	UART1_3_USE_CAN1	R/W
4	UART1_2_USE_CAN0	R/W
3:0		保留

24 AC/DC

24.1 时钟系统

1B 芯片内部工作部分包括三个电源域（power planes），如下表所示

表 24-1 1B 电源域

电源域	描述
Core	由主供电电源（main power supply）供电.当系统处于 S3，S4，S5 或 G3 状态时，这个电源域的供电将被断掉
RTC	系统断电时候，由外部电池供电；系统工作情况下，供电从电池切换到部普通电源
DDR	DDR2 工作所需电源

24.2 系统复位

1B 芯片在系统上电复位时，需要将某些芯片引脚上下拉进行配置。如下表格所示的全部引脚：

表 24-2 **1B** 上电配置引脚汇总

配置引脚	上下拉	功能
EJTAG_TRST	上拉	TAP 复位
EJTAG_TCK	上拉	TAP 时钟
EJTAG_TDI	上拉	TAP 数据输入
EJTAG_TMS	上拉	TAP 工作模式
SYS_RSTN	上拉	系统复位