龙芯中科
LOONGSON TECHNOLOGY

LOONGSON

龙芯 3B1000 处理器用户手册（1）

多核处理器架构与寄存器描述

2012 年 5 月

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

www.loongson.cn

版权声明

免责声明

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

龙芯中科技术有限公司

Loongson Technology Corporation Limited

地址：北京市海淀区中关村科学院南路 10 号

No.10 Kexueyuan South Road, Zhongguancun Haidian District, Beijing

电话(Tel)：010-62546668

传真(Fax)：010-62600826

阅读指南

《龙芯 3B1000 处理器用户手册》分为 3 册：

第 1 册：“多核处理器架构与寄存器描述”，介绍龙芯 3B1000 多核处理器架构，主要包括多核处理器架构与寄存器描述；

第 2 册：“GS464V 处理器核”，从系统软件开发者角度详细介绍龙芯 3B1000 所采用的 GS464V 高性能处理器核；

第 3 册：“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

文档更新记录	文档编号:	
	文档名:	龙芯 3B1000 处理器用户手册（1）
	版本号	V1.3
	创建人:	研发中心
	创建日期 ：	2012-05-04

更新历史

序号.	更新日期	更新人	版本号	更新内容
1	2011-06-10	研发中心	V1.0	初稿完成
2	2012-02-14	研发中心	V1.1	增加了 PLL 时钟倍频说明
3	2012-02-23	研发中心	V1.2	增加 DDR 配置寄存器中的中断描述
4	2012-05-04	研发中心	V1.3	修改中断路由寄存器描述 增加 HT 配置寄存器描述 增加一级 XBAR 标号描述

目 录

图目录

表目录

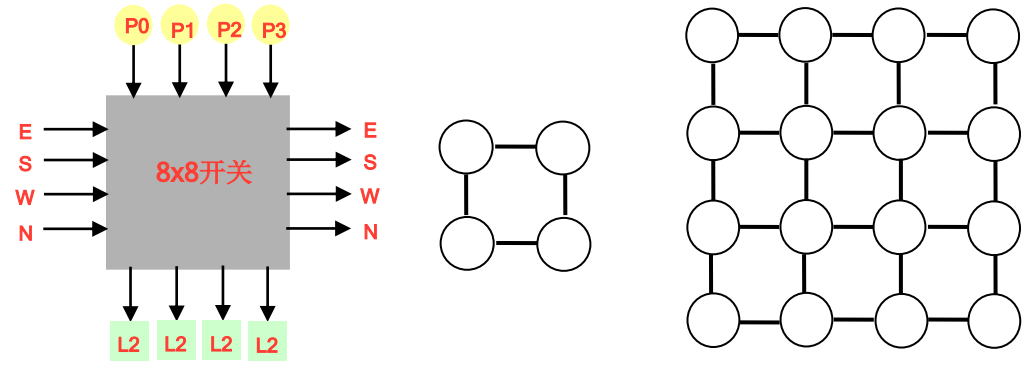
1 概述

1.1 龙芯系列处理器介绍

龙芯处理器主要包括三个系列。龙芯 1 号处理器及其 SOC 系列主要面向嵌入式应用，龙芯 2 号超标量处理器及其 SOC 系列主要面向桌面应用，龙芯 3 号多核处理器系列主要面向服务器和高性能机应用。根据应用的需要，其中部分龙芯 2 号也可以面向部分高端嵌入式应用，部分低端龙芯 3 号也可以面向部分桌面应用。上述三个系列将并行地发展。

龙芯 3 号多核系列处理器基于可伸缩的多核互连架构设计，在单个芯片上集成多个高性能处理器核以及大量的 2 级 Cache，还通过高速 I/O 接口实现多芯片的互连以组成更大规模的系统。

龙芯 3 号采用的可伸缩互连结构如下图 1-1 所示。龙芯 3 号片内及多片系统均采用二维 mesh 互连结构，其中每个结点由 8*8 的交叉开关组成，每个交叉开关连接四个处理器核以及四个二级 Cache，并与东（E）南（N）西（W）北（N）四个方向的其他结点互连。因此，2*2 的 mesh 可以连接 16 个处理器核，4*4 的 mesh 可以连接 64 个处理器核。



龙芯 3 号结点及二维互连结构，（a）结点结构，（b）2*2 的 mesh 网络连接 16 个处理器，（c）4*4 的 mesh 网络连接 64 个处理器。

图 1-1 龙芯 3 号系统结构

龙芯 3 号的结点结构如下图 1-2 所示。每个结点有两级 AXI 交叉开关连接处理器、二级 Cache、内存控制器以及 IO 控制器。其中第一级 AXI 交叉开关（称为 X1 Switch，简称 X1）连接处理器和二级 Cache。第二级交叉开关（称为 X2 Switch，简称 X2）连接二级 Cache

和内存控制器。

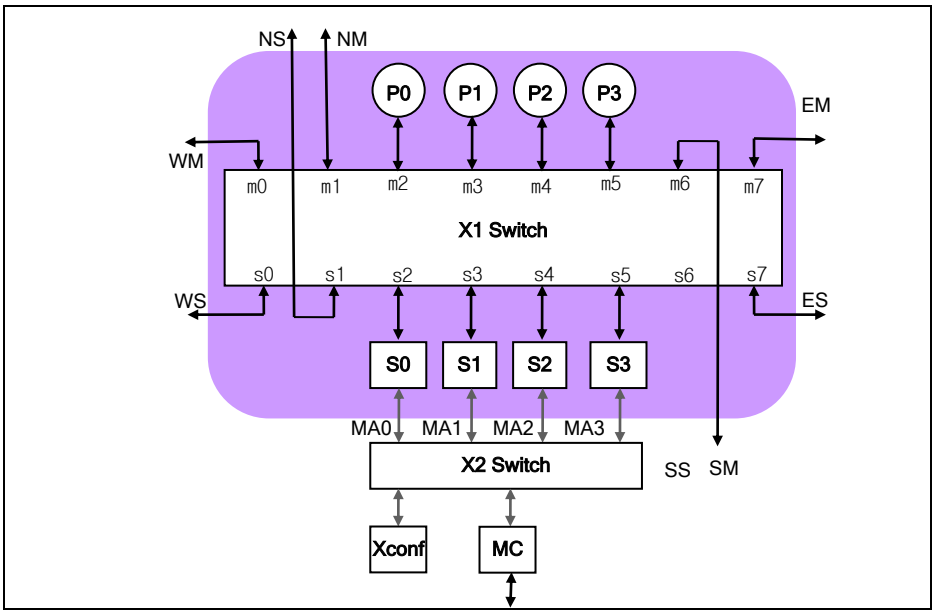


图 1-2 龙芯 3 号节点结构

在每个结点中，最多 8*8 的 X1 交叉开关通过四个 Master 端口连接四个 GS464V 处理器核（图中 P0、P1、P2、P3），通过四个 Slave 端口连接统一编址的四个 interleave 二级 Cache 块（图中 S0、S1、S2、S3），通过四对 Master/Slave 端口连接东、南、西、北四个方向的其他结点或 IO 结点（图中 EM/ES、SM/SS、WM/WS、NM/NS）。

X2 交叉开关通过四个 Master 端口连接四个二级 Cache，至少一个 Slave 端口连接一个内存控制器，至少一个 Slave 端口连接一个交叉开关的配置模块（Xconf）用于配置本结点的 X1 和 X2 的地址窗口等。还可以根据需要连接更多的内存控制器和 IO 端口等。

龙芯 3 号的互连系统只是定义上层协议，不对传输协议的实现做具体规定，因此，结点之间的互连即可以采用片上网络进行实现，也可以通过 I/O 控制链路实现多芯片的互连。在一个 4 结点 16 核系统为例，既可以通过 4 片 4 核芯片组成，也可以通过 2 片 8 核芯片，或基于一个单芯片 4 节点 16 核芯片组成。由于互连系统的物理实现对软件透明，上述 3 种配置的系统可以运行相同的操作系统进行。

1.2 龙芯 3B1000 简介

龙芯 3B1000 是龙芯 3 号多核处理器系列的第二款产品，是一个配置为双节点的 8 核处理器，采用 65nm 工艺制造，最高工作主频为 1GHz，主要技术特征如下：

- 片内集成 8 个 64 位的四发射超标量 GS464 高性能处理器核；
- 片内集成 4 MB 的分体共享二级 Cache(由 8 个体模块组成，每个体模块容量为 512KB) ；

- 通过目录协议维护多核及 I/O DMA 访问的 Cache 一致性；
- 片内集成 2 个 64 位 400MHz 的 DDR2/3 控制器；
- 片内集成 2 个 16 位 800MHz 的 HyperTransport 控制器；
- 每个 16 位的 HT 端口可以拆分成两个 8 路的 HT 端口使用。
- 片内集成 32 位 33MHz PCI；
- 片内集成 1 个 LPC、2 个 UART、1 个 SPI、16 路 GPIO 接口；

龙芯 3B1000 号芯片整体架构基于两级互连实现，结构如下图 1-3 所示。

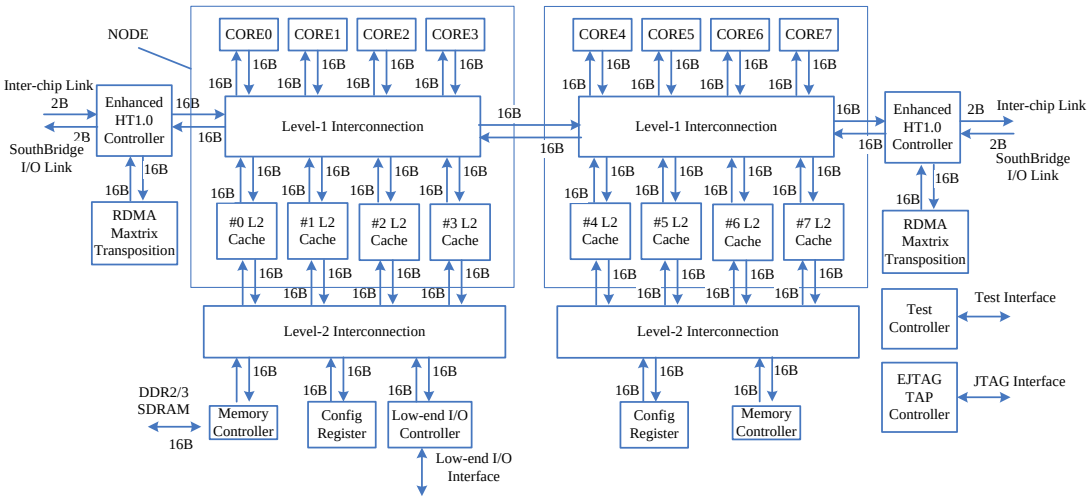


图 1-3 龙芯 3B1000 芯片结构

第一层互连采用两个相连的 6x6 交叉开关，分别用于连接四个 GS464V 处理器核（作为主设备）、四个二级 Cache 模块（作为从设备）、一个 HT 端口（每个端口使用一个 Master 和一个 Slave）以及相邻的节点。一级互连开关连接的 16 位 HT 控制器还可以作为两个 8 位的 HT 端口使用。HT 控制器和一级互联开关相连，其中的 DMA 控制器负责 HT 总线 IO 请求的 DMA 控制并负责片间一致性的维护。龙芯 3B1000 的 DMA 控制器还可以通过配置实现预取、矩阵转置及数据搬运功能。

第二级互连采用两个独立的 5x4 交叉开关，分别连接 4 个 2 级 Cache 模块(作为主设备)，DDR2 内存控制器、低速高速 I/O（包括 PCI、LPC、SPI 等）（节点二没有低速 IO）以及芯片内部的控制寄存器模块。

上述两级互连开关都采用读写分离的数据通道，数据通道宽度为 128bit，工作在与处理器核相同的频率，用以提供高速的片上数据传输。

基于龙芯 3B1000 可扩展互联架构，两片 8 核龙芯 3B1000 可以通过 HT 端口连接构成两芯片 16 核的 SMP 结构。

2 芯片配置与控制

2.1 芯片工作模式

龙芯 3B1000 有两种工作模式：

- 单芯片模式。系统只集成 1 片龙芯 3B1000，是一个两节点的非均匀访存多处理器系统（CC-NUMA）。
- 多芯片互连模式。系统中包含 2 片龙芯 3B1000，通过龙芯 3B1000 的 HT 端口进行互连，是一个四节点的非均匀访存多处理器系统（CC-NUMA）。

2.2 控制引脚说明

龙芯 3B1000 的控制引脚包括 DO_TEST、ICCC_EN、NODE_ID[1:0]、CLKSEL[15:0]、PCI_CONFIG。

表 2-1 控制引脚说明

<table>
<tr><th>信号</th><th>上下拉</th><th>作用</th></tr>
<tr><td>DO_TEST</td><td>上拉</td><td>1’b1 表示功能模式
1’b0 表示测试模式</td></tr>
<tr><td>ICCC_EN</td><td>下拉</td><td>1’b1 表示多芯片一致性互联模式
1’b0 表示单芯片模式</td></tr>
<tr><td>NODE_ID[1:0]</td><td></td><td>在多芯片一致性互连模式下表示处理器号
主处理器设为 2’b00；
从处理器设为 2’b10</td></tr>
<tr><td>CLKSEL[15:0]</td><td></td><td>上电时钟控制
HT 时钟控制
<table>
<tr><th>信</th><th>作用</th></tr>
<tr><td>CLKSEL[15]</td><td>1’b1 表示采用内部参考电压
1’b0 表示采用外部参考电压</td></tr>
<tr><td>CLKSEL[14]</td><td>1’b1 表示 HT PLL 采用差分时钟输入
1’b0 表示 HT PLL 采用普通时钟输入</td></tr>
<tr><td>CLKSEL[13:12]</td><td>2’b00 表示 PHY 时钟为 1.6GHZ/1
2’b01 表示 PHY 时钟为 3.2GHZ/2
2’b10 表示 PHY 时钟为普通输入时钟</td></tr>
</table></td></tr>
</table>

	2’b11 表示 PHY 时钟为差分输入时钟
CLKSEL[11:10]	2’b00 表示 HT 控制器时钟 200MHz 2’b01 表示 HT 控制器时钟 400MHz 2’b1x 表示 HT 控制器时钟为普通 入时钟

DDR 的时钟频率要求

MEMCLK 必须在 25MHz ~ 200MHz 之间

Mem_vco = MEMCLK * (20 + clksel[7:5]*2 + (clksel[7:5]==7)*2)

Mem_vco 的值必须在 700MHz ~ 3200MHz 之间

$DDR\_clock = Mem\_vco / (2^{(clksel[9:8] + 1)})$

DDR_clock 是最后内存的运行频率，可以参考下表的倍频系数。

内存控制器时钟控制

CLKSEL[9:5]	倍频系数	CLKSEL[9:5]	倍频系数
00000	10	10000	2.5
00001	11	10001	2.75
00010	12	10010	3
00011	13	10011	3.25
00100	14	10100	3.5
00101	15	10101	3.75
00110	16	10110	4
00111	18	10111	4.5
01000	5	11 00	1.25
01001	5.	11001	1.375
01010	6	11010	1.5
01011	6.5	11011	1.625
01100	7	11100	1.75
01101	7.5	11101	1.875
01110	8	11110	2
01111	9	11111	1

处理器核的时钟频率要求

SYSCLK 必须在 25MHz ~ 200MHz 之间

Sys_vco = SYSCLK * (25 + clksel[3:0])

Sys_vco 的值必须在 700MHz ~ 3200MHz 之间

$Core\_clock = Sys\_vco / (2^{clksel[4]})$

Core_clock 是最后内存的运行频率，可以参考下表的倍频系数。

处理器核时钟控制

CLKSEL[4:0]	倍频系数	CLKSEL[4:0]	倍频系数
00000	25	10000	12.5
00001	26	10001	13
00010	27	10010	13.5
00011	28	10011	14
00100	29	10100	14.5
00101	30	10101	15
00110	31	10110	15.5

00111	32	10111	16
010	33	11000	16.5
01001	34	11001	17
01010	35	11010	17.5
01011	36	11011	18
01100	37	11100	18.5
01101	38	11101	19
01110	39	11110	19.5
01111	40	11111	1

PCI_CONFIG[7:0]

IO 配置控制

7 HT 控制信号引脚电压控制位 1*

6:5 PCIX 总线速度选择*

4 PCIX 总线模式选择

3 PCI 主设备模式

2 系统从 PCI 空间启动

1 使用外部 PCI 仲裁

0 HT 控制信号引脚电压控制位 0*

注*:

6	5	4	PCIX 总线模式
0	0	0	PCI 33
0	1	1	PCI-X

注*:

7	0	HT 控制信号引脚电压，这些信号包括 HT_8x2，HT_Mode，HT_Powerok，HT_Rstn，HT_Ldt_Stopn，HT_Ldt_Reqn。
0	0	1.8v
0	1	Reserved
1	0	2.5v
1	1	3.3v

2.3 Cache 一致性

龙芯 3B1000 由硬件维护处理器以及通过 HT 端口接入的 I/O 之间的 Cache 一致性，但硬件不维护通过 PCI 接入到系统中的 I/O 设备的 Cache 一致性。在驱动程序开发时，对通过 PCI 接入的设备进行 DMA（Direct Memory Access）传输时，需要由软件进行 Cache 一致性维护。

2.4 系统节点级的物理地址空间分布

龙芯 3 号系列处理器的系统物理地址分布采用全局可访问的层次化寻址设计，以保证系统开发的扩展兼容。整个系统的物理地址宽度为 48 位。按照地址的高 4 位，整个地址空间被均匀分布到 16 个结点上，即每个结点分配 44 位地址空间。

龙芯 3B1000 采用双节点 8 核配置，因此龙芯 3B1000 芯片集成的 DDR 内存控制器、HT 总线、PCI 总线的对应地址都包含在从 0x0（含）至 0x2000_0000_0000（不含）的 44 位地

址空间内，具体各设备地址分布请参见后续相关章节。

表 2-2 节点级的系统全局地址分布

节点号	地址[47:44]位	起始地址	结束地址
0	0	0x0000_0000_0000	0x1000_0000_0000
1	1	0x1000_0000_0000	0x2000_0000_0000
2	2	0x2000_0000_0000	0x3000_0000_0000
3	3	0x3000_0000_0000	0x4000_0000_0000
4	4	0x4000_0000_0000	0x5000_0000_0000
5	5	0x5000_0000_0000	0x6000_0000_0000
6	6	0x6000_0000_0000	0x7000_0000_0000
7	7	0x7000_0000_0000	0x8000_0000_0000
8	8	0x8000_0000_0000	0x9000_0000_0000
9	9	0x9000_0000_0000	0xa000_0000_0000
10	0xa	0xa000_0000_0000	0xb000_0000_0000
11	0xb	0xb000_0000_0000	0xc000_0000_0000
12	0xc	0xc000_0000_0000	0xd000_0000_0000
13	0xd	0xd000_0000_0000	0xe000_0000_0000
14	0xe	0xe000_0000_0000	0xf000_0000_0000
15	0xf	0xf000_0000_0000	0x1_0000_0000_0000

在每个节点的内部，44 位地址空间又进一步均匀分布给结点内连接的可能最多 8 个设备。其中低 43 位地址由 4 个 2 级 Cache 模块所拥有，高 43 位地址则进一步按地址的[43:42]位分布给连接在 4 个方向端口的设备上。根据芯片和系统结构配置的不同，如果某端口上没有连接从设备，则对应的地址空间是保留地址空间，不允许访问。

表 2-3 节点内的地址分布

设备	地址[43:41]位	节点内起始地址	节点结束地址
2 级 Cache	0,1,2,3	0x000_0000_0000	0x800_0000_0000
东	4	0x800_0000_0000	0xa00_0000_0000
南	5	0xa00_0000_0000	0xc00_0000_0000
西	6	0xc00_0000_0000	0xe00_0000_0000
北	7	0xe00_0000_0000	0x1000_0000_0000

例如节点 0 的东端口设备的基地址为 0x0800_0000_0000，节点 1 的东端口设备的基地址为 0x1800_0000_0000，依次类推。

不同于方向端口的映射关系，龙芯 3B1000 可以根据实际应用的访问行为，来决定二级 Cache 的交叉寻址方式。节点内的 4 个 2 级 Cache 模块一共对应 43 位地址空间，而每个 2 级模块所对应的地址空间根据地址位的某两位选择位确定，并可以通过软件进行动态配置修改。系统中设置了名为 SCID_SEL 的配置寄存器来确定地址选择位，如下表所示。在缺省情况下采用[6:5]地位散列的方式进行分布，即地址[6:5]两位决定对应的 2 级 Cache 编号。节点 0 上的寄存器地址 0x3FF00400，节点 1 上的寄存器地址为 0x1000_3FF04400。两个节点上的散列方式可以分别配置。

表 2-4 二级缓存散列配置

SCID_SEL	地址位选择	SCID_SEL	地址位选择
4'h0	6: 5	4'h8	23:22
4'h1	9: 8	4'h9	25:24
4'h2	11:10	4'ha	27:26
4'h3	13:12	4'hb	29:28
4'h4	15:14	4'hc	31:30
4'h5	17:16	4'hd	33:32
4'h6	19:18	4'he	35:34
4'h7	21:20	4'hf	37:36

2.5 地址路由分布与配置

龙芯 3B1000 的路由主要通过系统的两级交叉开关实现。一级交叉开关可以对每个 Master 端口接收到的请求进行路由配置，每个 Master 端口都拥有 8 个地址窗口，可以完成 8 个地址窗口的目标路由选择。每个地址窗口由 BASE、MASK 和 MMAP 三个 64 位寄存器组成，BASE 以 K 字节对齐；MASK 采用类似网络掩码高位为 1 的格式；MMAP 的低三位表示对应目标 Slave 端口的编号，MMAP[4]表示允许取指，MMAP [5]表示允许块读，MMAP [7]表示窗口使能。

窗口命中公式: (IN_ADDR & MASK) == BASE

由于龙芯 3 号缺省采用固定路由，在上电启动时，配置窗口都处于关闭状态，使用时需要系统软件对其进行使能配置。

节点 0 的地址窗口转换寄存器如下表所示。节点 1 的地址窗口基址是在节点 0 的地址基础上增加 0x1000_0000_4000，即对应于 Core4_Win0_Base 的地址是 0x1000_3FF0_6000。

表 2-5 节点 0 一级交叉开关地址窗口寄存器表

地址	寄存器	地址	寄存器
0x3ff0_2000	CORE0_WIN0_BASE	0x3ff0_2100	CORE1_WIN0_BASE
0x3ff0_2008	CORE0_WIN1_BASE	0x3ff0_2108	CORE1_WIN1_BASE
0x3ff0_2010	CORE0_WIN2_BASE	0x3ff0_2110	CORE1_WIN2_BASE
0x3ff0_2018	CORE0_WIN3_BASE	0x3ff0_2118	CORE1_WIN3_BASE
0x3ff0_2020	CORE0_WIN4_BASE	0x3ff0_2120	CORE1_WIN4_BASE
0x3ff0_2028	CORE0_WIN5_BASE	0x3ff0_2128	CORE1_WIN5_BASE
0x3ff0_2030	CORE0_WIN6_BASE	0x3ff0_2130	CORE1_WIN6_BASE
0x3ff0_2038	CORE0_WIN7_BASE	0x3ff0_2138	CORE1_WIN7_BASE
0x3ff0_2040	CORE0_WIN0_MASK	0x3ff0_2140	CORE1_WIN0_MASK
0x3ff0_2048	CORE0_WIN1_MASK	0x3ff0_2148	CORE1_WIN1_MASK
0x3ff0_2050	CORE0_WIN2_MASK	0x3ff0_2150	CORE1_WIN2_MASK
0x3ff0_2058	CORE0_WIN3_MASK	0x3ff0_2158	CORE1_WIN3_MASK
0x3ff0_2060	CORE0_WIN4_MASK	0x3ff0_2160	CORE1_WIN4_MASK
0x3ff0_2068	CORE0_WIN5_MASK	0x3ff0_2168	CORE1_WIN5_MASK
0x3ff0_2070	CORE0_WIN6_MASK	0x3ff0_2170	CORE1_WIN6_MASK
0x3ff0_2078	CORE0_WIN7_MASK	0x3ff0_2178	CORE1_WIN7_MASK
0x3ff0_2080	CORE0_WIN0_MMAP	0x3ff0_2180	CORE1_WIN0_MMAP
0x3ff0_2088	CORE0_WIN1_MMAP	0x3ff0_2188	CORE1_WIN1_MMAP
0x3ff0_2090	CORE0_WIN2_MMAP	0x3ff0_2190	CORE1_WIN2_MMAP
0x3ff0_2098	CORE0_WIN3_MMAP	0x3ff0_2198	CORE1_WIN3_MMAP
0x3ff0_20a0	CORE0_WIN4_MMAP	0x3ff0_21a0	CORE1_WIN4_MMAP
0x3ff0_20a8	CORE0_WIN5_MMAP	0x3ff0_21a8	CORE1_WIN5_MMAP
0x3ff0_20b0	CORE0_WIN6_MMAP	0x3ff0_21b0	CORE1_WIN6_MMAP
0x3ff0_20b8	CORE0_WIN7_MMAP	0x3ff0_21b8	CORE1_WIN7_MMAP
0x3ff0_2200	CORE2_WIN0_BASE	0x3ff0_2300	CORE3_WIN0_BASE
0x3ff0_2208	CORE2_WIN1_BASE	0x3ff0_2308	CORE3_WIN1_BASE
0x3ff0_2210	CORE2_WIN2_BASE	0x3ff0_2310	CORE3_WIN2_BASE
0x3ff0_2218	CORE2_WIN3_BASE	0x3ff0_2318	CORE3_WIN3_BASE
0x3ff0_2220	CORE2_WIN4_BASE	0x3ff0_2320	CORE3_WIN4_BASE
0x3ff0_2228	CORE2_WIN5_BASE	0x3ff0_2328	CORE3_WIN5_BASE
0x3ff0_2230	CORE2_WIN6_BASE	0x3ff0_2330	CORE3_WIN6_BASE
0x3ff0_2238	CORE2_WIN7_BASE	0x3ff0_2338	CORE3_WIN7_BASE

0x3ff0_2240	CORE2_WIN0_MASK	0x3ff0_2340	CORE3_WIN0_MASK
0x3ff0_2248	CORE2_WIN1_MASK	0x3ff0_2348	CORE3_WIN1_MASK
0x3ff0_2250	CORE2_WIN2_MASK	0x3ff0_2350	CORE3_WIN2_MASK
0x3ff0_2258	CORE2_WIN3_MASK	0x3ff0_2358	CORE3_WIN3_MASK
0x3ff0_2260	CORE2_WIN4_MASK	0x3ff0_2360	CORE3_WIN4_MASK
0x3ff0_2268	CORE2_WIN5_MASK	0x3ff0_2368	CORE3_WIN5_MASK
0x3ff0_2270	CORE2_WIN6_MASK	0x3ff0_2370	CORE3_WIN6_MASK
0x3ff0_2278	CORE2_WIN7_MASK	0x3ff0_2378	CORE3_WIN7_MASK
0x3ff0_2280	CORE2_WIN0_MMAP	0x3ff0_2380	CORE3_WIN0_MMAP
0x3ff0_2288	CORE2_WIN1_MMAP	0x3ff0_2388	CORE3_WIN1_MMAP
0x3ff0_2290	CORE2_WIN2_MMAP	0x3ff0_2390	CORE3_WIN2_MMAP
0x3ff0_2298	CORE2_WIN3_MMAP	0x3ff0_2398	CORE3_WIN3_MMAP
0x3ff0_22a0	CORE2_WIN4_MMAP	0x3ff0_23a0	CORE3_WIN4_MMAP
0x3ff0_22a8	CORE2_WIN5_MMAP	0x3ff0_23a8	CORE3_WIN5_MMAP
0x3ff0_22b0	CORE2_WIN6_MMAP	0x3ff0_23B10000	CORE3_WIN6_MMAP
0x3ff0_22b8	CORE2_WIN7_MMAP	0x3ff0_23B10008	CORE3_WIN7_MMAP
0x3ff0_2400	EAST_WIN0_BASE	0x3ff0_2500	SOUTH_WIN0_BASE
0x3ff0_2408	EAST_WIN1_BASE	0x3ff0_2508	SOUTH_WIN1_BASE
0x3ff0_2410	EAST_WIN2_BASE	0x3ff0_2510	SOUTH_WIN2_BASE
0x3ff0_2418	EAST_WIN3_BASE	0x3ff0_2518	SOUTH_WIN3_BASE
0x3ff0_2420	EAST_WIN4_BASE	0x3ff0_2520	SOUTH_WIN4_BASE
0x3ff0_2428	EAST_WIN5_BASE	0x3ff0_2528	SOUTH_WIN5_BASE
0x3ff0_2430	EAST_WIN6_BASE	0x3ff0_2530	SOUTH_WIN6_BASE
0x3ff0_2438	EAST_WIN7_BASE	0x3ff0_2538	SOUTH_WIN7_BASE
0x3ff0_2440	EAST_WIN0_MASK	0x3ff0_2540	SOUTH_WIN0_MASK
0x3ff0_2448	EAST_WIN1_MASK	0x3ff0_2548	SOUTH_WIN1_MASK
0x3ff0_2450	EAST_WIN2_MASK	0x3ff0_2550	SOUTH_WIN2_MASK
0x3ff0_2458	EAST_WIN3_MASK	0x3ff0_2558	SOUTH_WIN3_MASK
0x3ff0_2460	EAST_WIN4_MASK	0x3ff0_2560	SOUTH_WIN4_MASK
0x3ff0_2468	EAST_WIN5_MASK	0x3ff0_2568	SOUTH_WIN5_MASK
0x3ff0_2470	EAST_WIN6_MASK	0x3ff0_2570	SOUTH_WIN6_MASK
0x3ff0_2478	EAST_WIN7_MASK	0x3ff0_2578	SOUTH_WIN7_MASK
0x3ff0_2480	EAST_WIN0_MMAP	0x3ff0_2580	SOUTH_WIN0_MMAP

0x3ff0_2488	EAST_WIN1_MMAP	0x3ff0_2588	SOUTH_WIN1_MMAP
0x3ff0_2490	EAST_WIN2_MMAP	0x3ff0_2590	SOUTH_WIN2_MMAP
0x3ff0_2498	EAST_WIN3_MMAP	0x3ff0_2598	SOUTH_WIN3_MMAP
0x3ff0_24a0	EAST_WIN4_MMAP	0x3ff0_25a0	SOUTH_WIN4_MMAP
0x3ff0_24a8	EAST_WIN5_MMAP	0x3ff0_25a8	SOUTH_WIN5_MMAP
0x3ff0_24b0	EAST_WIN6_MMAP	0x3ff0_25b0	SOUTH_WIN6_MMAP
0x3ff0_24b8	EAST_WIN7_MMAP	0x3ff0_25b8	SOUTH_WIN7_MMAP
0x3ff0_2600	WEST_WIN0_BASE	0x3ff0_2700	NORTH_WIN0_BASE
0x3ff0_2608	WEST_WIN1_BASE	0x3ff0_2708	NORTH_WIN1_BASE
0x3ff0_2610	WEST_WIN2_BASE	0x3ff0_2710	NORTH_WIN2_BASE
0x3ff0_2618	WEST_WIN3_BASE	0x3ff0_2718	NORTH_WIN3_BASE
0x3ff0_2620	WEST_WIN4_BASE	0x3ff0_2720	NORTH_WIN4_BASE
0x3ff0_2628	WEST_WIN5_BASE	0x3ff0_2728	NORTH_WIN5_BASE
0x3ff0_2630	WEST_WIN6_BASE	0x3ff0_2730	NORTH_WIN6_BASE
0x3ff0_2638	WEST_WIN7_BASE	0x3ff0_2738	NORTH_WIN7_BASE
0x3ff0_2640	WEST_WIN0_MASK	0x3ff0_2740	NORTH_WIN0_MASK
0x3ff0_2648	WEST_WIN1_MASK	0x3ff0_2748	NORTH_WIN1_MASK
0x3ff0_2650	WEST_WIN2_MASK	0x3ff0_2750	NORTH_WIN2_MASK
0x3ff0_2658	WEST_WIN3_MASK	0x3ff0_2758	NORTH_WIN3_MASK
0x3ff0_2660	WEST_WIN4_MASK	0x3ff0_2760	NORTH_WIN4_MASK
0x3ff0_2668	WEST_WIN5_MASK	0x3ff0_2768	NORTH_WIN5_MASK
0x3ff0_2670	WEST_WIN6_MASK	0x3ff0_2770	NORTH_WIN6_MASK
0x3ff0_2678	WEST_WIN7_MASK	0x3ff0_2778	NORTH_WIN7_MASK
0x3ff0_2680	WEST_WIN0_MMAP	0x3ff0_2780	NORTH_WIN0_MMAP
0x3ff0_2688	WEST_WIN1_MMAP	0x3ff0_2788	NORTH_WIN1_MMAP
0x3ff0_2690	WEST_WIN2_MMAP	0x3ff0_2790	NORTH_WIN2_MMAP
0x3ff0_2698	WEST_WIN3_MMAP	0x3ff0_2798	NORTH_WIN3_MMAP
0x3ff0_26a0	WEST_WIN4_MMAP	0x3ff0_27a0	NORTH_WIN4_MMAP
0x3ff0_26a8	WEST_WIN5_MMAP	0x3ff0_27a8	NORTH_WIN5_MMAP
0x3ff0_26b0	WEST_WIN6_MMAP	0x3ff0_27b0	NORTH_WIN6_MMAP
0x3ff0_26b8	WEST_WIN7_MMAP	0x3ff0_27b8	NORTH_WIN7_MMAP

在龙芯 3B1000 的二级 XBAR 中有 CPU 地址空间（包括 HT 空间）、DDR2 地址空间、以及 PCI 地址空间共三个 IP 相关的地址空间。地址窗口是供 CPU 和 PCI-DMA 两个具有

Master 功能的 IP 进行路由选择和地址转换而设置的。CPU 和 PCI-DMA 两者都拥有 8 个地址窗口，可以完成目标地址空间的选择以及从源地址空间到目标地址空间的转换。每个地址窗口由 BASE、MASK 和 MMAP 三个 64 位寄存器组成，BASE 以 K 字节对齐，MASK 采用类似网络掩码高位为 1 的格式，MMAP 的低三位是路由选择。

表 2-6 一级 XBAR 标号与所述模块的对应关系

标号	缺省值
0	二级缓存块 0
1	二级缓存块 1
2	二级缓存块 2
3	二级缓存块 3
4	相邻节点
5、6	保留
7	HT 模块 （对于节点 0，为 HT0；对于节点 1，对 HT1）

在二级 XBAR 处，标号与所述模块的对应关系如下表示对应新地址空间的编号（其中 DDR2 的标号为 0，PCI/Local IO 编号为 2，配置寄存器模块连接在端口 3 上）。

表 2-7 二级 XBAR 标号与所述模块的对应关系

标号	缺省值
0	0 号 DDR2/3 控制器
1	空
2	低速 I/O（PCI，LPC 等）
3	配置寄存器

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

表 2-8 MMAP 字段对应的该空间访问属性

[4]	[5]	[7]
允许取指	允许块读	窗口使能

二级 XBAR 的地址配置与一级 XBAR 的地址配置相比增加了地址转换的功能。相比之下，一级 XBAR 的窗口配置不能对 Cache 一致性的请求进行地址转换，否则在二级 Cache 处的地址会与处理器一级 Cache 处的地址不一致，导致 Cache 一致性的维护错误。

窗口命中公式: (IN_ADDR & MASK) == BASE

新地址换算公式: OUT_ADDR = (IN_ADDR & ~MASK) | {MMAP[63:10],10'h0}

节点 0 的地址窗口转换寄存器如下表。节点 1 的地址窗口基址是在节点 0 的地址基础上增加 0x1000_0000_4000，即对应于节点 1 的 CPU_Win0_Base 的地址是 0x1000_3FF0_4000。

表 2-9 二级 XBAR 地址窗口转换寄存器表

地址	寄存器	描述	缺省值
3ff0 0000	CPU_WIN0_BASE	CPU 窗口 0 的基地址	0x0
3ff0 0008	CPU_WIN1_BASE	CPU 窗口 1 的基地址	0x1000_0000
3ff0 0010	CPU_WIN2_BASE	CPU 窗口 2 的基地址	0x0
3ff0 0018	CPU_WIN3_BASE	CPU 窗口 3 的基地址	0x0
3ff0 0020	CPU_WIN4_BASE	CPU 窗口 4 的基地址	0x0
3ff0 0028	CPU_WIN5_BASE	CPU 窗口 5 的基地址	0x0
3ff0 0030	CPU_WIN6_BASE	CPU 窗口 6 的基地址	0x0
3ff0 0038	CPU_WIN7_BASE	CPU 窗口 7 的基地址	0x0
3ff0 0040	CPU_WIN0_MASK	CPU 窗口 0 的掩码	0xffff_ffff_f000_0000
3ff0 0048	CPU_WIN1_MASK	CPU 窗口 1 的掩码	0xffff_ffff_f000_0000
3ff0 0050	CPU_WIN2_MASK	CPU 窗口 2 的掩码	0x0
3ff0 0058	CPU_WIN3_MASK	CPU 窗口 3 的掩码	0x0
3ff0 0060	CPU_WIN4_MASK	CPU 窗口 4 的掩码	0x0
3ff0 0068	CPU_WIN5_MASK	CPU 窗口 5 的掩码	0x0
3ff0 0070	CPU_WIN6_MASK	CPU 窗口 6 的掩码	0x0
3ff0 0078	CPU_WIN7_MASK	CPU 窗口 7 的掩码	0x0
3ff0 0080	CPU_WIN0_MMAP	CPU 窗口 0 的新基地址	0xf0
3ff0 0088	CPU_WIN1_MMAP	CPU 窗口 1 的新基地址	0x1000_00f2
3ff0 0090	CPU_WIN2_MMAP	CPU 窗口 2 的新基地址	0
3ff0 0098	CPU_WIN3_MMAP	CPU 窗口 3 的新基地址	0
3ff0 00a0	CPU_WIN4_MMAP	CPU 窗口 4 的新基地址	0x0
3ff0 00a8	CPU_WIN5_MMAP	CPU 窗口 5 的新基地址	0x0
3ff0 00b0	CPU_WIN6_MMAP	CPU 窗口 6 的新基地址	0
3ff0 00b8	CPU_WIN7_MMAP	CPU 窗口 7 的新基地址	0
3ff0 0100	PCI_WIN0_BASE	PCI 窗口 0 的基地址	0x8000_0000
3ff0 0108	PCI_WIN1_BASE	PCI 窗口 1 的基地址	0x0
3ff0 0110	PCI_WIN2_BASE	PCI 窗口 2 的基地址	0x0
3ff0 0118	PCI_WIN3_BASE	PCI 窗口 3 的基地址	0x0

3ff0 0120	PCI_WIN4_BASE	PCI 窗口 4 的基地址	0x0
3ff0 0128	PCI_WIN5_BASE	PCI 窗口 5 的基地址	0x0
3ff0 0130	PCI_WIN6_BASE	PCI 窗口 6 的基地址	0x0
3ff0 0138	PCI_WIN7_BASE	PCI 窗口 7 的基地址	0x0
3ff0 0140	PCI_WIN0_MASK	PCI 窗口 0 的掩码	0xffff_ffff_8000_0000
3ff0 0148	PCI_WIN1_MASK	PCI 窗口 1 的掩码	0x0
3ff0 0150	PCI_WIN2_MASK	PCI 窗口 2 的掩码	0x0
3ff0 0158	PCI_WIN3_MASK	PCI 窗口 3 的掩码	0x0
3ff0 0160	PCI_WIN4_MASK	PCI 窗口 4 的掩码	0x0
3ff0 0168	PCI_WIN5_MASK	PCI 窗口 5 的掩码	0x0
3ff0 0170	PCI_WIN6_MASK	PCI 窗口 6 的掩码	0x0
3ff0 0178	PCI_WIN7_MASK	PCI 窗口 7 的掩码	0x0
3ff0 0180	PCI_WIN0_MMAP	PCI 窗口 0 的新基地址	0xf0
3ff0 0188	PCI_WIN1_MMAP	PCI 窗口 1 的新基地址	0x0
3ff0 0190	PCI_WIN2_MMAP	PCI 窗口 2 的新基地址	0
3ff0 0198	PCI_WIN3_MMAP	PCI 窗口 3 的新基地址	0
3ff0 01a0	PCI_WIN4_MMAP	PCI 窗口 4 的新基地址	0x0
3ff0 01a8	PCI_WIN5_MMAP	PCI 窗口 5 的新基地址	0x0
3ff0 01b0	PCI_WIN6_MMAP	PCI 窗口 6 的新基地址	0
3ff0 01b8	PCI_WIN7_MMAP	PCI 窗口 7 的新基地址	0

根据缺省的寄存器配置，芯片启动后，CPU 的 0x00000000 - 0x0fffffff 的地址区间（256M）映射到 DDR2 的 0x00000000 - 0x0fffffff 的地址区间，CPU 的 0x10000000 - 0x1fffffff 区间（256M）映射到 PCI 的 0x10000000 - 0x1fffffff 区间，PCIDMA 的 0x80000000 - 0x8fffffff 的地址区间（256M）映射到 DDR2 的 0x00000000 - 0x0fffffff 的地址区间。软件可以通过修改相应的配置寄存器实现新的地址空间路由和转换。

此外，当出现由于 CPU 猜测执行引起对非法地址的读访问时，8 个地址窗口都不命中，由配置寄存器模块返回全 0 的数据给 CPU，以防止 CPU 死等。

表 2-10 二级 XBAR 缺省地址配置

基地址	高位	所有者
0x0000_0000_0000_0000	0x0000_0000_1000_0000	0 号 DDR 控制器

0x0000_0000_1000_0000	0x0000_0000_2000_0000	低速 I/O（PCI 等）

2.6 芯片配置及采样寄存器

龙芯 3B1000 中的芯片配置寄存器(Chip_config)及芯片采样寄存器(chip_sample)提供了对芯片的配置进行读写的机制。

表 2-11 芯片配置寄存器（物理地址 0x1fe00180）

位域	字段名	访问	复位值	描述
2:0	Freq_scale_ctrl	RW	3’b111	处理器核分频
3	DDR_Clksel_en	RW	1’b0	是否使用软件配置 DDR 倍频
4	MC0_Disable_ddr2_confspace	RW	1’b0	是否禁用 MC0 DDR 配置空间
5	MC0_DDR_buffer_cpu	RW	1’b0	是否打开 MC0 DDR 读访问缓冲
6	MC0_read_buffer_bypass	RW	1’b0	是否绕过 MC0 读缓冲
7	MC0_ddr2_resetn	RW	1’b1	MC0 软件复位
8	MC0_Disable_ddr2_confspace	RW	1’b0	是否禁用 MC0 DDR 配置空间
9	MC1_DDR_buffer_cpu	RW	1’b0	是否打开 MC1 DDR 读访问缓冲
10	MC1_read_buffer_bypass	RW	1’b0	是否绕过 MC1 读缓冲
11	MC1_ddr2_resetn	RW	1’b1	MC1 软件复位
12	MC1_Disable_ddr2_confspace	RW	1’b0	是否禁用 MC1 DDR 配置空间
13	MC1_DDR_buffer_cpu	RW	1’b0	是否打开 MC1 DDR 读访问缓冲
26:24	HT_freq_scale_ctrl0	RW	3’b111	HT 控制器 0 分频
27	HT_clken0	RW	3’b0	是否禁用 HT0
30:28	HT_freq_scale_ctrl1	RW	3’b111	HT 控制器 1 分频
31	HT_clken1	RW	3’b0	是否禁用 HT1
44:40	DDR_Clksel	RW	5’b11111	软件配置 DDR 时钟倍频关系（当 DDR_Clksel_en 为 1 时有效）
48	Core0_en	RW	1’b1	是否启用处理器核 0
49	Core1_en	RW	1’b1	是否启用处理器核 1
50	Core2_en	RW	1’b1	是否启用处理器核 2
51	Core3_en	RW	1’b1	是否启用处理器核 3
52	Core0_en	RW	1’b1	是否启用处理器核 0

53	Core1_en	RW	1’b1	是否启用处理器核 1
54	Core2_en	RW	1’b1	是否启用处理器核 2
55	Core3_en	RW	1’b1	是否启用处理器核 3
其它		R		保留

表 2-12 芯片采样寄存器（物理地址 0x1fe00190）

位域	字段名	访问	复位值	描述
15:0	Pad2v5_ctrl	RW	16’h780	2v5pad 控制
31:16	Pad3v3_ctrl	RW	16’h780	3v3pad 控制
47:32	Sys_clksel	R		板上倍频设置
55:48	Bad_ip_core	R		core7-core0 是否坏
57:56	Bad_ip_ddr	R		2 个 DDR 控制器是否坏
61:60	Bad_ip_ht	R		2 个 HT 控制器是否坏
102:96	Thsens0_out	R		温度传感器 0 温度
103	Thsens0_overflow	R		温度传感器 0 温度上溢（超过 128 度）
110:104	Thsens1_out	R		温度传感器 1 温度
111	Thsens1_overflow	R		温度传感器 1 温度上溢（超过 128 度）
其它		R		保留

3 GS464V 处理器核

GS464V 是四发射 64 位的 MIPS 兼容的高性能处理器核，面向高性能计算和高端嵌入式等应用。GS464V 的主要特点是在 GS464 的基础上，将两个 64 位浮点功能部件替换成了两个 256 位向量功能部件，每个向量功能部件每拍可以完成 4 个 64 位运算，或 8 个 32 位运算，或 16 个 16 位运算，或 32 个 8 位运算。因此，GS464V 在 1G 主频下就可以达到 16GFlops 浮点（双精度）峰值性能。GS464V 在 GS464 支持的指令集的基础上，引入了大量新的 256 位向量指令，对媒体处理、信号处理、科学计算和加解密等进行了专门的支持。

在龙芯 3B1000 中的 8 个 GS464 核通过以及二级 Cache 模块通过 AXI 互联网络形成一个分布式共享二级 Cache 的多核结构。

GS464V 的主要特点如下：

- MIPS64 兼容；
- 提供 256 位向量指令；
- 提供 64 位 SIMD 型多媒体指令；
- 提供 X86 虚拟机指令；
- 四发射超标量结构，两个定点、两个向量、一个访存部件；
- 每个向量部件都支持 4 个全流水 64 位/8 个双 32 位定点或浮点乘加运算；
- 访存部件支持 128 位存储访问，虚地址和物理地址各为 48 位；
- 支持寄存器重命名、动态调度、转移预测等乱序执行技术；
- 64 项全相联 TLB，独立的 16 项指令 TLB，可变页大小；
- 一级指令 Cache64KB，4 路组相联；
- 一级数据 Cache32KB，2 路组相联；
- 支持 Non-blocking 访问及 Load-Speculation 等访存优化技术；
- 支持 Cache 一致性协议，可用于片内多核处理器；
- 指令 Cache 实现奇偶校验，数据 Cache 实现 ECC 校验；
- 支持标准的 EJTAG 调试标准，方便软硬件调试；
- 标准的 128 位 AXI 接口。

GS464V 的结构如下图所示。GS464V 更多的详细介绍请参考 GS464V 用户手册以及 MIPS64 用户手册。

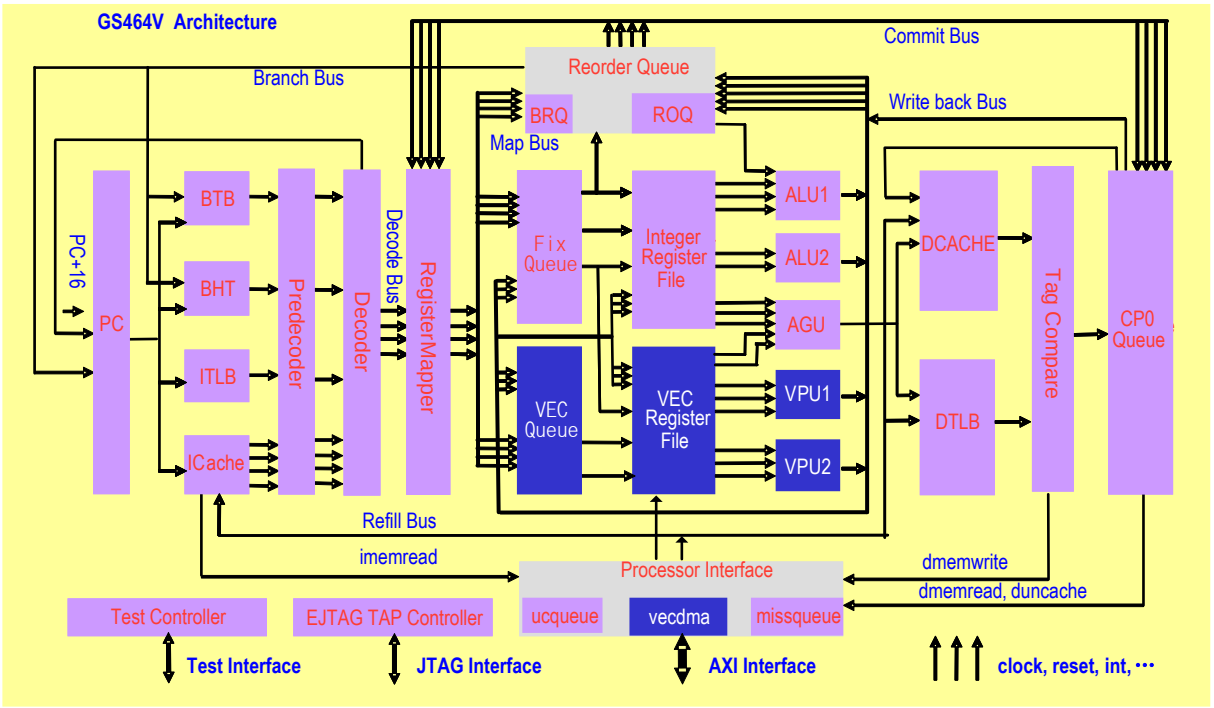


图 3-1 GS464V 结构图

4 二级 Cache

二级 Cache 模块是与 GS464 处理器 IP 配套设计的模块。该模块既可以和 GS464 对接，使 GS464 成为包括二级 Cache 在内的处理器 IP；也可以通过 AXI 网络连接多个 GS464 以及多个二级 Cache 模块，形成片内多处理器的 CMP 结构。二级 Cache 模块的主要特征包括：

- 通过目录支持 Cache 一致性协议。
- 采用四路组相联结构。
- 运行时可动态关闭。
- 支持 ECC 校验。
- 支持 DMA 一致性读写和预取读。
- 支持 16 种二级 Cache 散列方式。
- 支持按窗口锁二级 Cache。

为降低功耗，二级 Cache 的 TAG、目录和数据可以分开访问，二级 Cache 状态位、w 位与 TAG 一起存储，TAG 存放在 TAG RAM 中，目录存放在 DIR RAM 中，数据存放在 DATA RAM 中。失效请求访问二级 Cache，同时读出所有路的 TAG、目录和数据，并根据 TAG 来选出数据和目录。替换请求、重填请求和写回请求只操作一路的 TAG、目录和数据。

为提高一些特定计算任务的性能，二级 Cache 增加了锁机制。落在被锁区域中的二级 Cache 块会被锁住，因而不会被替换出二级 Cache（除非四路二级 Cache 中都是被锁住的块）。通过 confbus 可以对二级 Cache 模块内部的四组锁窗口寄存器进行动态配置，但必须保证四路二级 Cache 中一定有一路被锁住。因此二级 cache 锁住的区域大小没有限制，只要不超过二级 cache 大小的 3/4 就可以。此外当二级 Cache 收到 DMA 写请求时，如果被写的区域在二级 Cache 中命中且被锁住，那么 DMA 写将直接写入到二级 Cache 而不是内存。

表 4-1 二级 Cache 锁窗口寄存器配置

名称	地址	位域	描述
Slock0_valid	0x3ff00200	[63:63]	0 号锁窗口有效位
Slock0_addr	0x3ff00200	[47:0]	0 号锁窗口锁地址
Slock0_mask	0x3ff00240	[47:0]	0 号锁窗口掩码
Slock1_valid	0x3ff00208	[63:63]	1 号锁窗口有效位
Slock1_addr	0x3ff00208	[47:0]	1 号锁窗口锁地址
Slock1_mask	0x3ff00248	[47:0]	1 号锁窗口掩码
Slock2_valid	0x3ff00210	[63:63]	2 号锁窗口有效位
Slock2_addr	0x3ff00210	[47:0]	2 号锁窗口锁地址
Slock2_mask	0x3ff00250	[47:0]	2 号锁窗口掩码

Slock3_valid	0x3ff00218	[63:63]	3 号锁窗口有效位
Slock3_addr	0x3ff00218	[47:0]	3 号锁窗口锁地址
Slock3_mask	0x3ff00258	[47:0]	3 号锁窗口掩码

举例来说，当一个地址 addr 使得 slock0_valid && ((addr & slock0_mask) == (slock0_addr & slock0_mask))为 1 时，这个地址就被锁窗口 0 锁住了。

5 矩阵转置（加速器）

龙芯 3B1000 中内置了两个独立于处理器核的矩阵转置模块，其基本功能是通过软件的配置，实现对存放在内存中矩阵进行从源矩阵到目标矩阵的行列转置或搬移功能。两个矩阵转置模块分别集成在龙芯 3B1000 的两个 HyperTransport 控制器内部，通过 1 级交叉开关实现对二级 Cache 及内存的读写，并且能够维护 Cache 一致性。

完成转置功能时，由于转置前同一 Cache 块的元素顺序在转置后的矩阵中是分散的，如果提前读入多行数据，使得这些数据能够以 Cache 行为单位进行写入目标矩阵的地址空间，则能提高读写效率。因此矩阵转置模块中设置了一个大小为 32 行的缓存区，实现横向方式写入（从源矩阵读入到缓冲区），纵向读出（由缓冲区写入到目标矩阵）。

矩阵转置的工作过程为先读入 32 行源矩阵数据，再将该 32 行数据写入到目标矩阵，依次下去，直至完成整个矩阵的转置。矩阵转置模块还可以根据需要，仅进行预取源矩阵而不写目标矩阵，以此方式来实现对数据进行预取到 2 级 Cache 的操作。

与转置不同，进行矩阵搬移工作时可以直接以 Cache 块为单位进行读写操作。因为读写操作可能不是同时进行，需要对读回来的数据进行缓冲处理，因此矩阵搬移模块实现了一块 64*64bit 的缓存区，提供四个读端口和四个写端口。为了提高 Cache 块的读写效率，本模块要求源矩阵的起始地址，每行搬运字节数及行宽均是 32 字节的整数倍，对矩阵的列数则没有要求，并且硬件不考虑矩阵地址不对齐的情况。

转置和搬移操作涉及的源矩阵可能是位于一个大矩阵中的一个小矩阵，因此，其矩阵地址可能不是完全连续，相邻行之间的地址会有间隔，需要实现更多的编程控制接口。通过设置矩阵元素的大小，本模块还支持对源矩阵的行和列同时进行放大处理，但要求放大后的矩阵大小不能超出所在大矩阵。

下面表 5-1 到 5-4 说明了矩阵转置涉及到的编程接口。

表 5-1 矩阵转置编程接口说明

地址	名称	属性	说明
0x3ff00600	src_start_addr	RW	源矩阵起始地址
0x3ff00608	dst_start_addr	RW	目标矩阵起始地址
0x3ff00610	row	RW	源矩阵的一行元素个数
0x3ff00618	col	RW	源矩阵的一列元素个数
0x3ff00620	length	RW	源矩阵所在大矩阵的行跨度（字节）

0x3ff00628	width	RW	目标矩阵所在大矩阵的行跨度（字节）
0x3ff00630	trans_ctrl	RW	转置控制寄存器
0x3ff00638	trans_status	RO	转置状态寄存器

表 5-2 矩阵转置寄存器地址说明

地址	名称
0x3ff00600	0 号转置模块的 src_start_addr
0x3ff00608	0 号转置模块的 dst_start_addr
0x3ff00610	0 号转置模块的 row
0x3ff00618	0 号转置模块的 col
0x3ff00620	0 号转置模块的 length
0x3ff00628	0 号转置模块的 width
0x3ff00630	0 号转置模块的 trans_ctrl
0x3ff00638	0 号转置模块的 trans_status
0x3ff00700	1 号转置模块的 src_start_addr
0x3ff00708	1 号转置模块的 dst_start_addr
0x3ff00710	1 号转置模块的 src_row_stride
0x3ff00718	1 号转置模块的 src_last_row_addr
0x3ff00720	1 号转置模块的 length
0x3ff00728	1 号转置模块的 width
0x3ff00730	1 号转置模块的 trans_ctrl
0x3ff00738	1 号转置模块的 trans_status

表 5-3 trans_ctrl 寄存器的各位解释

字段	说明
0	fetch_en，是否允许读源矩阵。
1	store_en，是否允许写目标矩阵。为 0 时，转置过程只预取源矩阵，但不写目标矩阵。
2	源矩阵读取完毕后，是否有效中断。
3	目标矩阵写入完毕后，是否有效中断，
7..4	Arcmd，读命令内部控制位。当 arcache 为 4’hf 时，必须设为 4’he 或 4’hf，其中 4’he 表示读分配，4’hf 表示读不分配。当 arcache 为其它值时无意义。
11..8	Arcache，读命令内部控制位。为 4’hf 时，使用 cache 通路，为 4’h0 时，使用 uncache 通路。其它值无意义。

15..12	Awcmd，写命令内部控制位。当 awcache 为 4’hf 时，必须设为 4’hb。当 awcache 为其它值时无意义。
19..16	Awcache，写命令内部控制位。为 4’hf 时，使用 cache 通路，为 4’h0 时，使用 uncache 通路。其它值无意义。
21..20	element_width，矩阵的元素大小。00 表示 1 个字节，01 表示 2 个字节，10 表示 4 个字节，11 表示 8 个字节
22	trans_yes，为 1 表示进行转置；为 0 表示不转置

表 5-4 trans_status 寄存器的各位解释

字段	说明
0	源矩阵读取完毕
1	目标矩阵写入完毕

6 处理器核间中断与通信

龙芯 3B1000 为每个处理器核都实现了 8 个核间中断寄存器（IPI）以支持多核 BIOS 启动和操作系统运行时在处理器核之间进行中断和通信，其说明和地址见表 6-1 到表 6-5。

表 6-1 处理器核间中断相关的寄存器及其功能描述

名称	读写权限	描述
IPI_Status	R	32 位状态寄存器，任何一位有被置 1 且对应位使能情况下，处理器核 INT4 中断线被置位。
IPI_Enable	RW	32 位使能寄存器，控制对应中断状态位是否有效
IPI_Set	W	32 位中断置位寄存器，往对应的位写 1，则 STATUS 寄存器对应的位被置 1
IPI_Clear	W	32 位中断清除寄存器，往对应的位写 1，则 STATUS 寄存器对应的位被清 0
MailBox0	RW	缓存寄存器，供启动时传递参数使用，按 64 或者 32 位的 uncache 方式进行访问。
MailBox01	RW	缓存寄存器，供启动时传递参数使用，按 64 或者 32 位的 uncache 方式进行访问。
MailBox02	RW	缓存寄存器，供启动时传递参数使用，按 64 或者 32 位的 uncache 方式进行访问。
MailBox03	RW	缓存寄存器，供启动时传递参数使用，按 64 或者 32 位的 uncache 方式进行访问。

龙芯 3B1000 节点 0 的处理器核间中断相关的寄存器及其功能描述如下，节点 1 的处理器核相关寄存器的地址是在节点 0 的地址基础上增加 0x1000_0000_4000，如对应于 Core4_IPI_Status 的地址是 0x1000_3FF0_5000。

表 6-2 0 号节点核间中断与通信寄存器列表

名称	地址	权限	描述
Core0_IPI_Status	0x3ff01000	R	0 号处理器核的 IPI_Status 寄存器
Core0_IPI_Enalbe	0x3ff01004	RW	0 号处理器核的 IPI_Enalbe 寄存器
Core0_IPI_Set	0x3ff01008	W	0 号处理器核的 IPI_Set 寄存器
Core0 _IPI_Clear	0x3ff0100c	W	0 号处理器核的 IPI_Clear 寄存器
Core0_MailBox0	0x3ff01020	RW	0 号处理器核的 IPI_MailBox0 寄存器
Core0_ MailBox1	0x3ff01028	RW	0 号处理器核的 IPI_MailBox1 寄存器
Core0_ MailBox2	0x3ff01030	RW	0 号处理器核的 IPI_MailBox2 寄存器
Core0_ MailBox3	0x3ff01038	RW	0 号处理器核的 IPI_MailBox3 寄存器

表 6-3 1 号处理器核的核间中断与通信寄存器列表

名称	地址	权限	描述
Core1_IPI_Status	0x3ff01100	R	1 号处理器核的 IPI_Status 寄存器

Core1_IPI_Enalbe	0x3ff01104	RW	1 号处理器核的 IPI_Enalbe 寄存器
Core1_IPI_Set	0x3ff01108	W	1 号处理器核的 IPI_Set 寄存器
Core1 _IPI_Clear	0x3ff0110c	W	1 号处理器核的 IPI_Clear 寄存器
Core1_MailBox0	0x3ff01120	R	1 号处理器核的 IPI_MailBox0 寄存器
Core1_ MailBox1	0x3ff01128	RW	1 号处理器核的 IPI_MailBox1 寄存器
Core1_ MailBox2	0x3ff01130	W	1 号处理器核的 IPI_MailBox2 寄存器
Core1_ MailBox3	0x3ff01138	W	1 号处理器核的 IPI_MailBox3 寄存器

表 6-4 2 号处理器核的核间中断与通信寄存器列表

名称	地址	权限	描述
Core2_IPI_Status	0x3ff01200	R	2 号处理器核的 IPI_Status 寄存器
Core2_IPI_Enalbe	0x3ff01204	RW	2 号处理器核的 IPI_Enalbe 寄存器
Core2_IPI_Set	0x3ff01208	W	2 号处理器核的 IPI_Set 寄存器
Core2 _IPI_Clear	0x3ff0120c	W	2 号处理器核的 IPI_Clear 寄存器
Core2_MailBox0	0x3ff01220	R	2 号处理器核的 IPI_MailBox0 寄存器
Core2_ MailBox1	0x3ff01228	RW	2 号处理器核的 IPI_MailBox1 寄存器
Core2_ MailBox2	0x3ff01230	W	2 号处理器核的 IPI_MailBox2 寄存器
Core2_ MailBox3	0x3ff01238	W	2 号处理器核的 IPI_MailBox3 寄存器

表 6-5 3 号处理器核的核间中断与通信寄存器列表

名称	地址	权限	描述
Core3_IPI_Status	0x3ff01300	R	3 号处理器核的 IPI_Status 寄存器
Core3_IPI_Enalbe	0x3ff01304	RW	3 号处理器核的 IPI_Enalbe 寄存器
Core3_IPI_Set	0x3ff01308	W	3 号处理器核的 IPI_Set 寄存器
Core3 _IPI_Clear	0x3ff0130c	W	3 号处理器核的 IPI_Clear 寄存器
Core3_MailBox0	0x3ff01320	R	3 号处理器核的 IPI_MailBox0 寄存器
Core3_ MailBox1	0x3ff01328	RW	3 号处理器核的 IPI_MailBox1 寄存器
Core3_ MailBox2	0x3ff01330	W	3 号处理器核的 IPI_MailBox2 寄存器
Core3_ MailBox3	0x3ff01338	W	3 号处理器核的 IPI_MailBox3 寄存器

上面列出的是龙芯 3B1000 多处理器芯片的节点 0 核间中断相关寄存器列表。在采用两片龙芯 3B1000 互连构成四节点 CC-NUMA 系统时，每个芯片内的节点对应一个系统全局节点编号，节点内处理器核的 IPI 寄存器地址按上表与其所在节点的基地址成固定偏移关系。例如，0 号节点 0 号处理器核的 IPI_Status 地址为 0x3ff01000，其它节点内的 0 号处理器核地址偏移增加 0x1000_0000_4000，比如 1 号节点的 0 号处理器地址则为 0x10003ff05000，依次类推。

7 I/O 中断

龙芯 3B1000 芯片最多支持 64 个中断源，以统一方式进行管理，如下图 7-1 所示，任意一个 IO 中断源可以被配置为是否使能、触发的方式、以及被路由的目标处理器核中断脚。

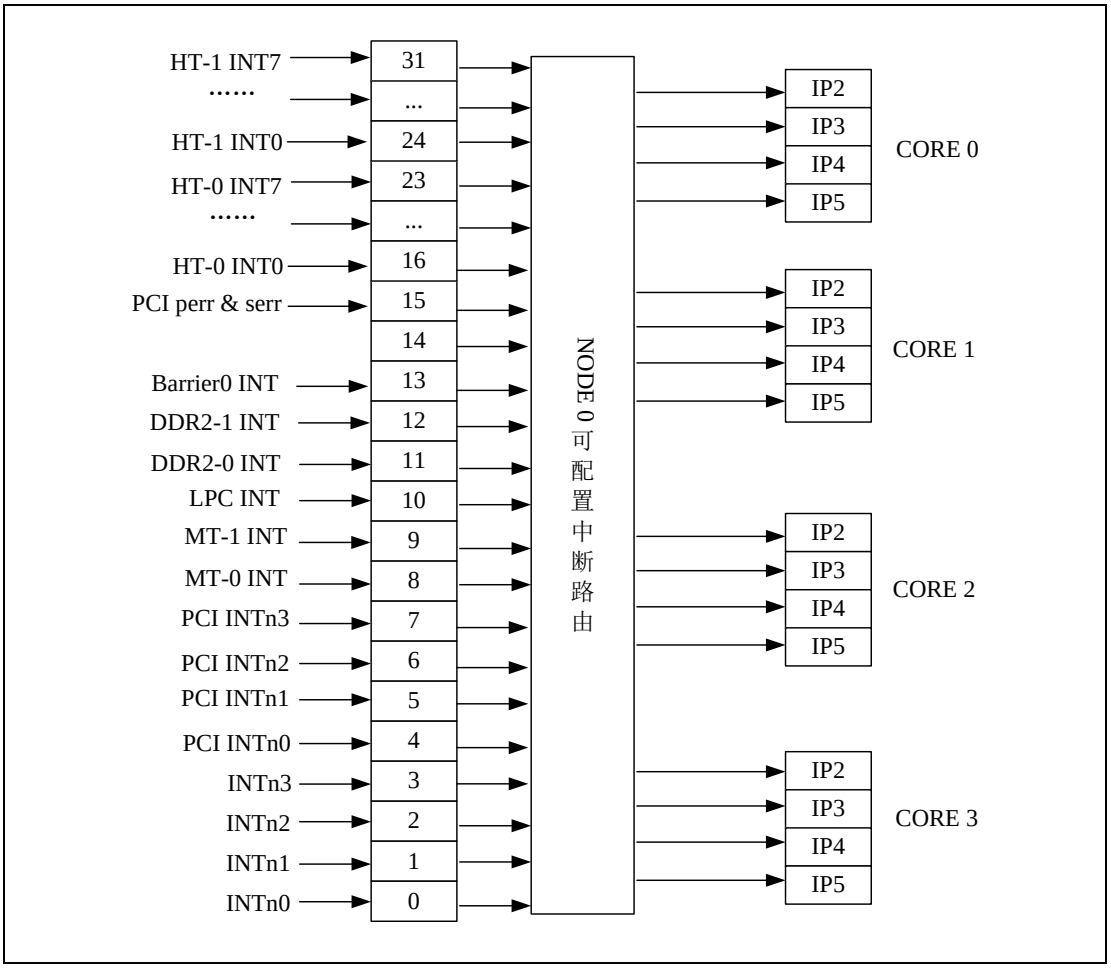


图 7-1 龙芯 3B1000 处理器中断路由示意图

中断相关配置寄存器都是以位的形式对相应的中断线进行控制，中断控制位连接及属性配置见下表 7-1。中断使能（Enable）的配置有三个寄存器：Intenset、Intenclr 和 Inten。Intenset 设置中断使能，Intenset 寄存器写 1 的位对应的中断被使能。Intenclr 清除中断使能，Intenclr 寄存器写 1 的位对应的中断被清除。Inten 寄存器读取当前各中断使能的情况。脉冲形式的中断信号（如 PCI_SERR）由 Intedge 配置寄存器来选择，写 1 表示脉冲触发，写 0 表示电平触发。中断处理程序可以通过 Intenclr 的相应位来清除脉冲记录。

表 7-1 中断控制寄存器

位域	访问属性/缺省值				
	Intedge	Inten	Intenset	Intenclr	中断源
3 : 0	RW / 0	R / 0	W / 0	W / 0	Sys_int0-3
7 : 4	RO / 0	R / 0	RW / 0	RW / 0	PCI_INTn

8	RO / 0	R / 0	RW / 0	RW / 0	Matrix_int0
9	RO / 1	R / 0	RW / 0	RW / 0	Matrix_int1
10	RO / 1	R / 0	RW / 0	RW / 0	Lpc
12 : 11	RW / 0	保留	保留	保留	Mc0-1
13	RW / 0	R / 0	RW / 0	RW / 0	Barrier
14	RW / 0	R / 0	RW / 0	RW / 0	保留
15	RW / 0	R / 0	RW / 0	RW / 0	Pci_perr
23 : 16	RW / 0	R / 0	RW / 0	RW / 0	HT0 int0-7
31 : 24	RW / 0	R / 0	RW / 0	RW / 0	HT1 int0-7

表 7-2 节点 0 IO 控制寄存器地址

名称	地址偏移	描述
Intisr	0x3ff01420	32 位中断状态寄存器
Inten	0x3ff01424	32 位中断使能状态寄存器
Intenset	0x3ff01428	32 位设置使能寄存器
Intenclr	0x3ff0142c	32 位清除使能寄存器
Intedge	0x3ff01438	32 位触发方式寄存器
CORE0_INTISR	0x3ff01440	路由给 CORE0 的 32 位中断状态
CORE1_INTISR	0x3ff01448	路由给 CORE1 的 32 位中断状态
CORE2_INTISR	0x3ff01450	路由给 CORE2 的 32 位中断状态
CORE3_INTISR	0x3ff01458	路由给 CORE3 的 32 位中断状态

节点 1 的处理器核相关寄存器的地址是在节点 0 的地址基础上增加 0x1000_0000_4000，如节点 1 的 Intisr 的地址是 0x1000_3FF0_5420。

在龙芯 3B1000 中集成了两个节点，每个节点包括 4 个处理器核，上述的 32 位中断源可以通过软件配置选择期望中断的目标处理器核。进一步，中断源可以选择路由到处理器核中断 IP2 到 IP5 中的任意一个，即对应 CP0_Status 的 IP2 到 IP5。每个节点内的 32 个 I/O 中断源中每一个都对应一个 8 位的路由控制器，其格式和地址如下表 7-3 和 7-4 所示。路由寄存器采用向量的方式进行路由选择，如 0x48 标示路由到 3 号处理器的 IP4 上。

表 7-3 中断路由寄存器的说明

位域	说 明
3:0	路由的处理器核向量号（第 0 位代表 0 号处理器核，第 1 位代表 1 号...）
7:4	路由的处理器核中断引脚向量号（第 0 位代表 IP2，第 1 位代表 IP3...）

表 7-4 中断路由寄存器地址

名称	地址偏移	描述	名称	地址偏移	描述

Entry0	0x3ff01400	Sys_int0	Entry16	0x3ff01410	HT0-int0
Entry1	0x3ff01401	Sys_int1	Entry17	0x3ff01411	HT0-int1
Entry2	0x3ff01402	Sys_int2	Entry18	0x3ff01412	HT0-int2
Entry3	0x3ff01403	Sys_int3	Entry19	0x3ff01413	HT0-int3
Entry4	0x3ff01404	Pci_int0	Entry20	0x3ff01414	HT0-int4
Entry5	0x3ff01405	Pci_int1	Entry21	0x3ff01415	HT0-int5
Entry6	0x3ff01406	Pci_int2	Entry22	0x3ff01416	HT0-int6
Entry7	0x3ff01407	Pci_int3	Entry23	0x3ff01417	HT0-int7
Entry8	0x3ff01408	Matrix int0	Entry24	0x3ff01418	HT1-int0
Entry9	0x3ff01409	Matrix int1	Entry25	0x3ff01419	HT1-int1
Entry10	0x3ff0140a	Lpc int	Entry26	0x3ff0141a	HT1-int2
Entry11	0x3ff0140b	Mc0	Entry27	0x3ff0141b	HT1-int3
Entry12	0x3ff0140c	Mc1	Entry28	0x3ff0141c	HT1-int4
Entry13	0x3ff0140d	Barrier	Entry29	0x3ff0141d	HT1-int5
Entry14	0x3ff0140e	保留	Entry30	0x3ff0141e	HT1-int6
Entry15	0x3ff0140f	Pci_perr/serr	Entry31	0x3ff0141f	HT1-int7

8 DDR2/3 SDRAM 控制器配置

龙芯 3B1000 处理器内部集成的内存控制器的设计遵守 DDR2 SDRAM 的行业标准（JESD79-2B）。在龙芯 3B1000 处理器中，所实现的所有内存读/写操作都遵守 JESD79-2B 及 JESD79-3 的规定。

8.1 DDR2 SDRAM 控制器功能概述

龙芯 3B1000 处理器支持最大 4 个 CS（由 4 个 DDR2 SDRAM 片选信号实现，即两个双面内存条），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）。最大支持的地址空间为 128GB（2^{37}）。

龙芯 3B1000 处理器在具体选择使用不同内存芯片类型时，可以调整 DDR2/3 控制器参数设置进行支持。其中，支持的最大片选（CS_n）数为 4，行地址（RAS_n）数为 15，列地址（CAS_n）数为 14，逻辑体选择（BANK_n）数为 3。

CPU 发送的内存请求物理地址将按照如下图所示的方法进行行列地址转换：

以 4GB 地址空间为例，按照下面的配置：

片选 = 4　　Bank 数 = 8

行地址数 = 12　列地址数 = 12

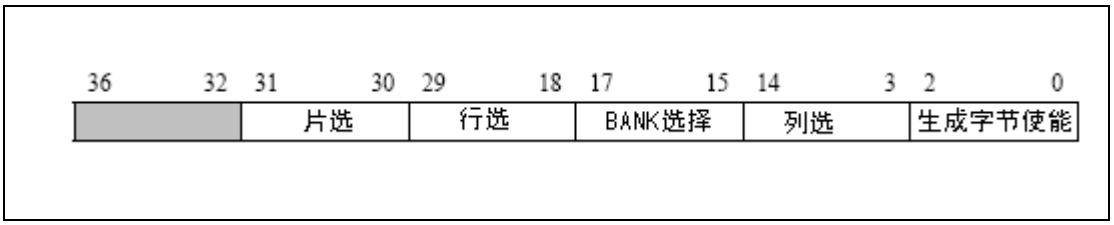


图 8-1 DDR2 SDRAM 行列地址与 CPU 物理地址的转换

龙芯 3B1000 处理器所集成的内存控制电路只接受来自处理器或者外部设备的内存读/写请求，在所有的内存读/写操作中，内存控制电路处于从设备状态（Slave State）。

龙芯 3B1000 处理器中内存控制器具有如下特征：

- 接口上命令、读写数据实现全流水操作
- 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
- ECC 功能可以对数据通路上的 1 位和 2 位错误进行检测，并能对 1 位错误进行自动纠错
- 支持 133-400MHZ 工作频率

8.2 DDR2 SDRAM 读操作协议

DDR2 SDRAM 读操作的协议如图 8-2 所示。在图中命令（Command，简称 CMD）由 RAS_n，CAS_n 和 WE_n 三个信号组成。对于读操作，RAS_n=1，CAS_n＝0，WE_n＝1。

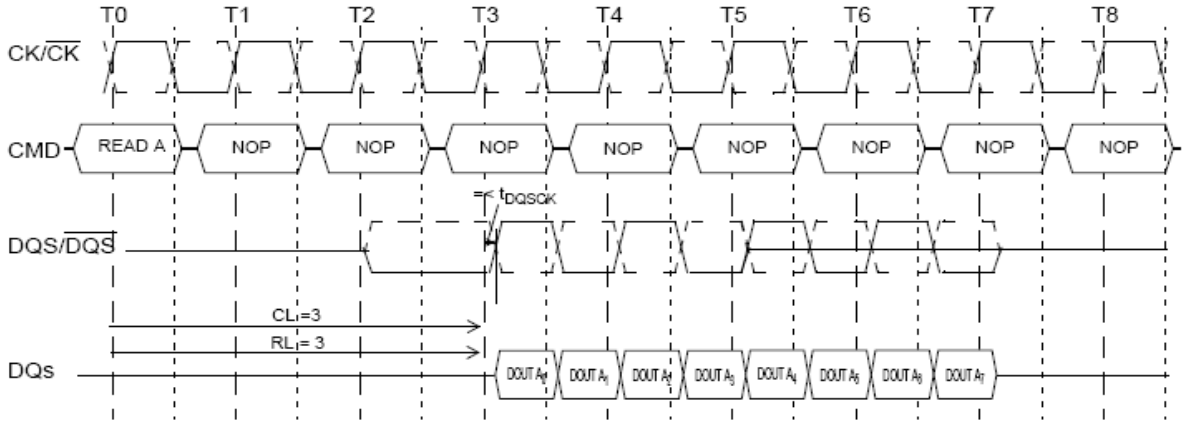


图 8-2 DDR2 SDRAM 读操作协议

上图中，Cas Latency (CL) = 3，Read Latency (RL) = 3，Burst Length = 8。

8.3 DDR2 SDRAM 参数配置格式

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

(1) 系统复位，此时控制器内部所有寄存器内容将被置为初始值。

(2) 系统解复位。

(3) 向配置寄存器地址发 64 位写指令，配置所有 180 个配置寄存器。此时如果写 CTRL_03，应将其中参数 START（位[40:40]） 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 64 位写指令。此时应将参数 START（位[40:40]）设为 1。结束后内存控制器将自动对内存发起初始化指令。

在龙芯 3B1000 处理器设计中，DDR2 SDRAM 需要在系统主板初始化完成以后，使用内存之前进行内存类型的配置。具体的配置操作是对物理地址 0x0000 0000 0FF0 0000 相对应的 180 个 64 位寄存器写入相应的配置参数。一个寄存器可能会包括多个、一个、部分参数的数据。这些配置寄存器及其包含的参数意义如下表（寄存器中未使用的位均为保留位），表中还给出了基于 DDR2 667 的一种寄存器配置方式，具体的配置可以根据实际情况决定：

表 8-1 DDR2 SDRAM 配置参数寄存器格式

参数名称	位	缺省值	范围	描述
CONF_CTL_00[63:0] Offset: 0x00			DDR2 667：0x0000010000000101	
CONCURRENTAP	48:48	0x0	0x0-0x1	是否允许控制器对一个 bank 进行 auto precharge 时，对另外一个 bank 发出命令。注：部分内存条不支持
BANK_SPLIT_EN	40:40	0x0	0x0-0x1	是否允许命令队列重排序逻辑对 bank 进行拆分（split）
AUTO_REFRESH_MODE	32:32	0x0	0x0-0x1	设置 auto-refresh 是在下一个 burst 还是下一个命令边界发出
AREFRESH	24:24	0x0	0x0-0x1	根据 auto_refresh_mode 参数的设置，向内存发起自动刷新命令（只写）
AP	16:16	0x0	0x0-0x1	是否使能内存控制器自动刷新功能，置 1，表示内存访问为 CLOSE PAGE 方式。
ADDR_CMP_EN	8:8	0x0	0x0-0x1	是否允许命令队列重排序逻辑对地址冲突进行检测
CONF_CTL_01[63:0] Offset: 0x10			DDR2 667：0x0000010100010000	
FWC	56:56	0x0	0x0-0x1	是否强制进行写检查，当这个参数设置后，内存控制器将用 xor_check_bits 参数指定的数与数据进行异或写入内存（只写）
FAST_WRITE	48:48	0x0	0x0-0x1	是否允许控制器打开快速写功能。打开快速写功能后，控制器在未收到全部写数据后即向内存模块发出写命令。
ENABLE_QUICK_SREFRESH	40:40	0x0	0x0-0x1	是否使能快速自刷新。当这个参数使能后，内存的初始化未进行完就进入自刷新状态
EIGHT_BANK_MODE	32:32	0x0	0x0-0x1	指示内存模块是否有 8 个 bank
ECC_DISBALE_W_UC_ERR	24:24	0x0	0x0-0x1	当检测到不可恢复的错误时，是否将 ECC 关闭
DQS_N_EN	16:16	0x0	0x0-0x1	设置 DQS 信号为单端还是差分信号。
DLLLOCKREG	0:0	0x0	0x0-0x1	指示 DLL 是否已锁定（只读），只有在 DLL 锁定之后，对内存发起的读写操作才能有效到达内存，所以，可以用本位判断第一次写内存的时机。
CONF_CTL_02[63:0] Offset: 0x20			DDR2 667：0x0100010100000000	
PRIORITY_EN	56:56	0x0	0x0-0x1	是否使能命令队列重排序逻辑使用优先级
POWER_DOWN	48:48	0x0	0x0-0x1	当使能这个参数时，内存控制器将用 pre-charge 命令关闭内存模块的所有页面，使时钟使能信号

				为低，不发送收到的所有命令，直到这个参数重新设置为 0
PLACEMENT_EN	40:40	0x0	0x0-0x1	是否使能命令重排序逻辑
ODT_ADD_TURN_CLK_EN	32:32	0x0	0x0-0x1	在对不同片选的快速背对背读或者写命令中间是否插入一个 turn-around 时钟。通常情况下，插入一个这样的周期是对内存是需要的。
NO_CMD_INIT	24:24	0x0	0x0-0x1	在内存初始化过程中，是否禁止在内存模块的 tDLL 时间内发出其它命令
INTRPTWRITENA	16:16	0x0	0x0-0x1	是否允许用 autoprecharge 命令加上对同一 bank 的其它写命令打断前一个写命令
INTRPTREADA	8:8	0x0	0x0-0x1	是否允许用 autoprecharge 命令加上对同一 bank 的其它读命令打断前一个读命令
INTRPTAPBURST	0:0	0x0	0x0-0x1	是否允许对另一 bank 的其它命令打断当前的 auto-precharge 命令
CONF_CTL_03[63:0]	Offset: 0x30		DDR2 667：0x0101010001000000	
SWAP_PORT_RW_SAME_EN	56:56	0x0	0x0-0x1	当 swap_en 使能时，该参数决定是否将同一端口上的类似命令进行交换
SWAP_EN	48:48	0x0	0x0-0x1	在使能命令队列重排序逻辑时，当高优先级命令到达时，是否将正在执行的命令与新命令交换
START	40:40	0x0	0x0-0x1	是否开始内存的初始化工作。需要在所有的参数配置完成之后，再设置该位，让内存进入初始化配置。在没有完成其它位的配置之前就配置该位，很可能导致内存访问错误。
SREFRESH	32:32	0x0	0x0-0x1	内存模块是否进入自刷新工作模式
RW_SAME_EN	24:24	0x0	0x0-0x1	在命令队列重排序逻辑中是否考虑对同一 bank 读写命令的重组
REG_DIMM_EN	16:16	0x0	0x0-0x1	是否使能 registered DIMM 内存模组
REDUC	8:8	0x0	0x0-0x1	是否只使用 32 位位宽的内存数据通道，通常情况下，不应设置该位
PWRUP_SREFRESH_EXIT	0:0	0x0	0x0-0x1	是用 self-refresh 命令而不是用正常的内存初始化命令来脱离下电模式
CONF_CTL_04[63:0]	Offset: 0x40		DDR2 667：0x0102010100010101	
RTT_0	57:56	0x0	0x0-0x3	使能内存模块的片上终端电阻。 00 －disable 其它－enable，电阻大小由 mrs_data 中的值决

				定
CTRL_RAW	49:48	0x0	0x0-0x3	设置 ECC 的检错和纠错模式 2’b00 - 不使用 ECC 2’b01 - 只报错，不纠错 2’b10 - 没有使用 ECC 设备 2’b11 - 使用 ECC 报错纠错
AXIO_W_PRIORITY	41:40	0x0	0x0-0x3	设置 AXIO 端口写命令优先级
AXIO_R_PRIORITY	33:32	0x0	0x0-0x3	设置 AXIO 端口读命令优先级
WRITE_MODEREG	24:24	0x0	0x0-0x1	是否写内存模块的 EMRS 寄存器（只写），每次写 1 时控制器就将配置参数中 emrs_data 与 mrs_data 发往内存。
WRITEINTERP	16:16	0x0	0x0-0x1	定义是否能用一个读命令取打断一个写突发
TREF_ENABLE	8:8	0x0	0x0-0x1	是否使能控制器内部的自动刷新功能，通常的情况下，应该将该位置 1
TRAS_LOCKOUT	0:0	0x0	0x0-0x1	是否在 tRAS 时间到期之前发出 auto-prechareg 命令
CONF_CTL_05[63:0]	Offset: 0x50		DDR2 667：0x0700000404050100	
Q_FULLNESS	58:56	0x0	0x0-0x7	定义内存控制器命令队列中有多少命令时认为命令队列满
PORT_DATA_ERROR_TYPE	50:48	0x0	0x0-0x7	定义内存控制器端口上数据错误类型（只读） 位 0 - 突发数据个数大于 16 位 1 - 写数据交错 位 2 - ECC 2 位错
OUT_OF_RANGE_TYPE	42:40	0x0	0x0-0x7	定义发生越界访问时的错误类型（只读）
MAX_CS_REG	34:32	0x4	0x0-0x4	定义控制器所用片选个数（只读）
COLUMN_SIZE	26:24	0x0	0x0-0x7	设置实际列地址数和最大列地址数(14)之间的差值，应该根据具体的内存颗粒进行配置。 内存所用列地址数 = 14 - COLUMN_SIZE
CASLAT	18:16	0x0	0x0-0x7	设置 CAS latency 值。应当根据具体的内存颗粒在不同的运行频率下进行配置。
ADDR_PINS	10:8	0x0	0x0-0x7	设置实际地址引脚数和最大地址数(15)之间的差值 内存所用地址线数 = 15 - ADDR_PINS

CONF_CTL_06[63:0]	Offset: 0x60		DDR2 667：0x0a04040603040003	
APREBIT	59:56	0x0	0x0-0xf	定义用哪位地址线向内存发出 autoprecharge 命令，一般为 bit 10。
WRLAT	50:48	0x0	0x0-0x7	写操作时写命令发出到接收到第一个数据的时间（按时钟周期数），同时决定何时使对应的 ODT 信号有效。 注：当 WRLAT = (CASLAT_LIN /2)时，会在不同 CS 读写之间加入一拍额外延迟。
TWTR	42:40	0x0	0x0-0x7	定义从写命令切换到读命令所需要的时钟周期数，需要根据具体内存颗粒及运行频率进行配置。
TWR_INT	34:32	0x0	0x0-0x7	定义内存模组的写恢复时间，需要根据具体内存颗粒及运行频率进行配置。
TRTP	26:24	0x0	0x0-0x7	定义内存模组的读命令到 precharge 周期数，需要根据具体内存颗粒及运行频率进行配置。
TRRD	18:16	0x0	0x0-0x7	定义到不同 bank 的 active 命令时间间隔，需要根据具体内存颗粒及运行频率进行配置。
TCKE	2:0	0x0	0x0-0x7	定义 CKE 信号最小脉宽
CONF_CTL_07[63:0]	Offset: 0x70		DDR2 667：0x0f0e0200000f0a0a	
MAX_ROW_REG	59:56	0xf	0x0-0xf	系统最大行地址个数（只读）
MAX_COL_REG	51:48	0xe	0x0-0xe	系统最大列地址个数（只读）
INITAREF	43:40	0x0	0x0-0xf	定义系统初始化时所需要执行的 autorefresh 命令个数。DDR2 时设为 2，DDR3 时设为 0。
CS_MAP	19:16	0x0	0x0-0xf	定义可用片选信号，本参数应当根据实际使用的片选个数进行正确的配置，不正确的配置将会导致错误的内存访问。该参数的四位分别对应于 CS0- CS3
CASLAT_LIN	3:0	0x0	0x0-0xf	当板上走线延迟为 DDR2 时钟周期的 0.5~1.5 倍：CASLAT_LIN = CASLAT×2 小于 0.5 倍：CASLAT_LIN = CASLAT×2-1 大于 1.5 倍：CASLAT_LIN = CASLAT×2+1 （以半个时钟周期为单位）
CONF_CTL_08[63:0]	Offset: 0x80		DDR2 667：0x0804020108040201	
ODT_WR_MAP_CS3	59:56	0x0	0x0-0xf	定义 CS3 有写命令时，将指定的 CS 的 ODT 终端

				电阻有效，具体的配置应当参考相应的内存颗粒手册对于 ODT 配置的要求。该参数的四位分别对应于 CS0- CS3
ODT_WR_MAP_CS2	51:48	0x0	0x0-0xf	定义 CS2 有写命令时，将指定的 CS 的 ODT 终端电阻有效，具体的配置应当参考相应的内存颗粒手册对于 ODT 配置的要求。该参数的四位分别对应于 CS0- CS3
ODT_WR_MAP_CS1	43:40	0x0	0x0-0xf	定义 CS1 有写命令时，将指定的 CS 的 ODT 终端电阻有效，具体的配置应当参考相应的内存颗粒手册对于 ODT 配置的要求。该参数的四位分别对应于 CS0- CS3
ODT_WR_MAP_CS0	35:32	0x0	0x0-0xf	定义 CS0 有写命令时，将指定的 CS 的 ODT 终端电阻有效，具体的配置应当参考相应的内存颗粒手册对于 ODT 配置的要求。该参数的四位分别对应于 CS0- CS3
ODT_RD_MAP_CS3	27:24	0x0	0x0-0xf	定义 CS3 有读命令时，将指定的 CS 的 ODT 终端电阻有效，具体的配置应当参考相应的内存颗粒手册对于 ODT 配置的要求。该参数的四位分别对应于 CS0- CS3
ODT_RD_MAP_CS2	19:16	0x0	0x0-0xf	定义 CS2 有读命令时，将指定的 CS 的 ODT 终端电阻有效，具体的配置应当参考相应的内存颗粒手册对于 ODT 配置的要求。该参数的四位分别对应于 CS0- CS3
ODT_RD_MAP_CS1	11:8	0x0	0x0-0xf	定义 CS1 有读命令时，将指定的 CS 的 ODT 终端电阻有效，具体的配置应当参考相应的内存颗粒手册对于 ODT 配置的要求。该参数的四位分别对应于 CS0- CS3
ODT_RD_MAP_CS0	3:0	0x0	0x0-0xf	定义 CS0 有读命令时，将指定的 CS 的 ODT 终端电阻有效，具体的配置应当参考相应的内存颗粒手册对于 ODT 配置的要求。该参数的四位分别对应于 CS0- CS3
CONF_CTL_09[63:0]	Offset: 0x90		DDR2 667：0x0000070d00000000	
OCD_ADJUST_PUP_CS0	60:56	0x0	0x0-0x1f	设置内存模组片选 0 OCD 上拉调整值。内存控制器将在初始化时根据这个参数的值向内存模组发出 OCD 调整命令
OCD_ADJUST_PDN_CS0	52:48	0x0	0x0-0x1f	设置内存模组片选 0 OCD 下拉调整值。内存控制

				器将在初始化时根据这个参数的值向内存模组发出 OCD 调整命令
TRP	43:40	0x0	0x0-0xf	定义内存模组执行 pre-charge 所需要的时钟周期数，需要根据具体内存颗粒及运行频率进行配置。
TDAL	35:32	0x0	0x0-0xf	当 auto-precharge 参数设置后，该参数定义了 auto-precharge 和 write recovery 时钟周期数。 TDAL = auto-precharge + write recovery 该参数仅在设置了 AP 之后才生效。
PORT_CMD_ERROR_TYPE	19:16	0x0	0x0-0xf	端口上发生命令错误的类型（只读） 位 0 - 数据位宽过大 位 1 - 关键字优先操作地址未对齐 位 2 - 关键字优先操作字数不是 2 幂 位 3 - narrow transform 出错
CONF_CTL_10[63:0]	Offset: 0xa0		DDR2 667：0x0000003f3f140612	
COMMAND_AGE_COUNT	37:32	0x0	0x0-0x3f	定义命令队列重排序逻辑使用 aging 算法时每个命令的 aging 初始值
AGE_COUNT	29:24	0x0	0x0-0x3f	定义命令队列重排序逻辑使用 aging 算法时每个命令的 aging 初始值
TRC	20:16	0x0	0x0-0x1f	定义对内存模组同一 bank 的 active 命令之间的时钟周期数，需要根据具体内存颗粒及运行频率进行配置。
TMRD	12:8	0x0	0x0-0x1f	定义配置内存模组模式寄存器需要的时钟周期数，通常为 2 个周期
TFAW	4:0	0x0	0x0-0x1f	定义内存模组 tFAW 参数，8 个逻辑 bank 时使用
CONF_CTL_12[63:0]	Offset: 0xc0		DDR2 667：0x00002c0511000000	
TRFC	47:40	0x0	0x0-0xff	定义内存模组刷新操作需要的时钟周期数，需要根据具体内存颗粒及运行频率进行配置。
TRCD_INT	39:32	0x0	0x0-0xff	定义内存模组 RAS 到 CAS 之间的时钟周期数，需要根据具体内存颗粒及运行频率进行配置。
TRAS_MIN	31:24	0x0	0x0-0xff	定义内存模组行地址有效命令的最小时钟周期数
OUT_OF_RANGE_LENGTH	23:16	0x0	0x0-0xff	发生越界访问时的命令长度（只读）

ECC_U_SYND	15:8	0x0	0x0-0xff	发生 2bit 不可纠错误时的原因（只读）
ECC_C_SYND	7:0	0x0	0x0-0xff	发生 1bit 可纠错错误时的原因（只读）
CONF_CTL_17[63:0]	Offset: 0x110		DDR2 667：0x0000000000000c2d	
TREF	13:0	0x0	0x0-0x3ff	定义内存模组两次刷新命令的时钟间隔，需要根据具体内存颗粒及运行频率进行配置。
CONF_CTL_18[63:0]	Offset: 0x120		DDR2 667：0x001c000000000000	
AXIO_EN_LT_WIDTH_INSTR	63:48	0x0000	0x0-0xffff	定义 AXIO 端口是否接收小于 64 位位宽的内存访问
CONF_CTL_19[63:0]	Offset: 0x130		DDR2 667：0x6d56000302000000	
TRAS_MAX	63:48	0x0000	0x0-0xffff	定义内存模组行有效命令的最大时钟周期数，需要根据具体内存颗粒及运行频率进行配置。
TPDEX	47:32	0x0000	0x0-0xffff	定义内存模组掉电退出命令的时钟周期数
TDLL	31:16	0x0000	0x0-0xffff	定义内存模组 DLL 锁定需要的时钟周期数
TCPD	15:0	0x0000	0x0-0xffff	定义内存模组时钟有效到 precharge 之间的时钟周期数，需要根据具体内存颗粒及运行频率进行配置。
CONF_CTL_20[63:0]	Offset: 0x140		DDR2 667：0x0000204002000030	
XOR_CHECK_BITS	63:48	0x0000	0x0-0xffff	当 fwc 参数设定时，下次写操作的 check bit 将会与该参数进行异或后写入内存
VERSION	47:32	0x2040	0x2040	定义内存控制器版本号（只读）
TXSR	31:16	0x0000	0x0-0xffff	定义内存模组自刷新退出需要的时钟周期数
TXSNR	15:0	0x0000	0x0-0xffff	定义内存模组 tXSNR 参数
CONF_CTL_21[63:0]	Offset: 0x150		DDR2 667：0x0000000000000000	
ECC_C_ADDR[36:8]	60:32	0x0	0x0-0x1fffffff	记录发生 1bit ECC 错误时的地址信息（只读）
ECC_C_ADDR[7:0]	31:24	0x0000	0x0-0x1fffffff	记录发生 1bit ECC 错误时的地址信息（只读）
TINIT	23:0	0x0000	0x0-0xffff	定义内存模组初始化需要的时钟周期数，需要根据具体内存颗粒及运行频率进行配置。一般为 200us。
CONF_CTL_22[63:0]	Offset: 0x160		DDR2 667：0x0000000000000000	
ECC_U_ADDR[36:32]	36:32	0x0	0x0-0x1fffffff	记录发生 2bit ECC 错误时的地址信息（只读）

ECC_U_ADDR[31:0]	31:0	0x0	0x0-0x1fffffffff	记录发生 2bit ECC 错误时的地址信息（只读）
CONF_CTL_23[63:0]	Offset: 0x170		DDR2 667：0x0000000000000000	
OUT_OF_RANGE_ADDR[36:32]	36:32	0x0	0x0-0x1fffffffff	记录发生越界访问时的地址信息（只读）
OUT_OF_RANGE_ADDR[31:0]	31:0	0x0	0x0-0x1fffffffff	记录发生越界访问时的地址信息（只读）
CONF_CTL_24[63:0]	Offset: 0x180		DDR2 667：0x0000000000000000	
PORT_CMD_ERROR_ADDR[36:32]	36:32	0x0	0x0-0x1fffffffff	记录端口发生命令错误时的地址信息（只读）
PORT_CMD_ERROR_ADDR[31:0]	31:0	0x0	0x0-0x1fffffffff	记录端口发生命令错误时的地址信息（只读）
CONF_CTL_25[63:0]	Offset: 0x190		DDR2 667：0x0000000000000000	
ECC_C_DATA[63:32]	63:32	0x0	0x0-0x1fffffffff	记录发生 1bit ECC 错误时的数据信息（只读）
ECC_C_DATA[31:0]	31:0	0x0	0x0-0x1fffffffff	记录发生 1bit ECC 错误时的数据信息（只读）
CONF_CTL_26[63:0]	Offset: 0x1a0		DDR2 667：0x0000000000000000	
ECC_U_DATA[63:32]	63:32	0x0	0x0-0x1fffffffff	记录发生 2bit ECC 错误时的数据信息（只读）
ECC_U_DATA[31:0]	31:0	0x0	0x0-0x1fffffffff	记录发生 2bit ECC 错误时的数据信息（只读）
CONF_CTL_27[63:0]	Offset: 0x1b0		DDR2 667：0x0000000000000000	
CKE_DELAY	2:0	0x0	0x0-0x7	CKE 有效延迟。 注：用于控制内部 srefresh_enter 命令的响应时间，对于龙芯 3 号无效。
CONF_CTL_29[63:0]	Offset: 0x1d0		DDR2 667：0x0103070400000101	
TDFI_PHY_WRLAT_BASE	59:56	0x0	0x0-0xf	设置 DDR PHY 中写数据需加入的延迟。对于龙芯 3 号这个值应为 2
TDFI_PHY_WRLAT	51:48	0x0	0x0-0xf	用于显示实际从写命令发出到写数据发出间隔的周期数（只读）
TDFI_PHY_RDLAT	44:40	0x0	0x0-0xf	设置读命令发出到读数据返回间隔的周期数
TDFI_CTRLUPD_MIN	35:32	0x4	0x0-0xf	保存 DFI Tctrlup_min 时间参数（只读）

DRAM_CLK_DISABLE	19:16	0x0	0x0-0xf	设置是否输出 DRAM 时钟信号，每位对应一个片选信号。0：输出时钟信号；1：禁止输出时钟信号。
ODT_ALT_EN	8:8	0x0	0x0-0x1	是否支持 CAS＝3 时的 ODT 信号。 注：对于龙芯 3 号，无效
DRIVE_DQ_DQS	0:0	0x0	0x0-0x1	设置当控制器空闲时是否驱动数据总线
CONF_CTL_30[63:0]	Offset: 0x1e0		DDR2 667：0x0c2d0c2d0c2d0205	
TDFI_PHYUPD_TYPE0	61:48	0x0000	0x0-0x3fff	这个值等于 TREF（只读）
TDFI_PHYUPD_RESP	45:32	0x0000	0x0-0x3fff	这个值等于 TREF（只读）
TDFI_CTRLUPD_MAX	29:16	0x0000	0x0-0x3fff	这个值等于 TREF（只读）
TDFI_RDDATA_EN_BASE	12:8	0x00	0x0-0x1f	DDR PHY 内部读命令发出到读返回的基本时间。对于龙芯 3 号这个值为 2
TDFI_RDDATA_EN	4:0	0x00	0x0-0x1f	用于显示从读命令发出到读数据返回的实际周期数
CONF_CTL_31[63:0]	Offset: 0x1f0		DDR2 667：0x0020008000000000	
DLL_CTRL_REG_0_0	63:32	0x00000	0x0-0xffffffff	第 0 数据组（DQ7-DQ0）DLL 控制信号 24： 控制内部 DLL 的使能信号，为 0 时 DLL 有效 23:16： 控制写数据（DQ）与 DQS 之间的相位关系，每个数值表示为（1/精度）＊360。在龙芯 3 号中，这个值一般为 1/4，即 8’h20 7:0： 控制内部 DLL 的精度，在龙芯 3 号中，这个值一般为 8’h80
DFT_CTRL_REG	7:0	0x00	0x0-0xff	测试使能信号，0x0 为正常工作模式
CONF_CTL_32[63:0]	Offset: 0x200		DDR2 667：0x0020008000200080	
DLL_CTRL_REG_0_2	63:32	0x000000	0x0-0xffffffff	第 2 数据组（DQ23-DQ16）DLL 控制信号 24： 控制内部 DLL 的使能信号，为 0 时 DLL 有效 23:16： 控制写数据（DQ）与 DQS 之间的相位关系，每个数值表示为（1/精度）＊360。在龙芯 3 号中，这个值一般为 1/4，即 8’h20 7:0： 控制内部 DLL 的精度，在龙芯 3 号中，这个值一般为 8’h80
DLL_CTRL_REG_0_1	31:0	0x0000	0x0-0xffff	第 1 数据组（DQ15-DQ8）DLL 控制信号

			ffff	24： 控制内部 DLL 的使能信号，为 0 时 DLL 有效 23:16： 控制写数据（DQ）与 DQS 之间的相位关系，每个数值表示为（1/精度）* 360。在龙芯 3 号中，这个值一般为 1/4，即 8’h20 7:0： 控制内部 DLL 的精度，在龙芯 3 号中，这个值一般为 8’h80
CONF_CTL_33[63:0]	Offset: 0x210		DDR2 667：0x0020008000200080	
DLL_CTRL_REG_0_4	63:32	0x000000	0x0-0xffffffff	第 4 数据组（DQ39-DQ32）DLL 控制信号 24： 控制内部 DLL 的使能信号，为 0 时 DLL 有效 23:16： 控制写数据（DQ）与 DQS 之间的相位关系，每个数值表示为（1/精度）* 360。在龙芯 3 号中，这个值一般为 1/4，即 8’h20 7:0： 控制内部 DLL 的精度，在龙芯 3 号中，这个值一般为 8’h80
DLL_CTRL_REG_0_3	31:0	0x00000	0x0-0xffffffff	第 3 数据组（DQ31-DQ24）DLL 控制信号 24： 控制内部 DLL 的使能信号，为 0 时 DLL 有效 23:16： 控制写数据（DQ）与 DQS 之间的相位关系，每个数值表示为（1/精度）* 360。在龙芯 3 号中，这个值一般为 1/4，即 8’h20 7:0： 控制内部 DLL 的精度，在龙芯 3 号中，这个值一般为 8’h80
CONF_CTL_34[63:0]	Offset: 0x220		DDR2 667：0x0020008000200080	
DLL_CTRL_REG_0_6	63:32	0x00000	0x0-0xffffffff	第 6 数据组（DQ55-DQ48）DLL 控制信号 24： 控制内部 DLL 的使能信号，为 0 时 DLL 有效 23:16： 控制写数据（DQ）与 DQS 之间的相位关系，每个数值表示为（1/精度）* 360。在龙芯 3 号中，这个值一般为 1/4，即 8’h20 7:0： 控制内部 DLL 的精度，在龙芯 3 号中，这个值一般为 8’h80
DLL_CTRL_REG_0_5	31:0	0x00000	0x0-0xffffffff	第 5 数据组（DQ47-DQ40）DLL 控制信号 24： 控制内部 DLL 的使能信号，为 0 时 DLL 有

				效 23:16： 控制写数据（DQ）与 DQS 之间的相位关系，每个数值表示为（1/精度）＊360。在龙芯 3 号中，这个值一般为 1/4，即 8’h20 7:0： 控制内部 DLL 的精度，在龙芯 3 号中，这个值一般为 8’h80
CONF_CTL_35[63:0]	Offset: 0x230		DDR2 667：0x0020008000200080	
DLL_CTRL_REG_0_8	63:32	0x00000	0x0-0xffffffff	第 8 数据组（DQ71-DQ64）DLL 控制信号 24： 控制内部 DLL 的使能信号，为 0 时 DLL 有效 23:16： 控制写数据（DQ）与 DQS 之间的相位关系，每个数值表示为（1/精度）＊360。在龙芯 3 号中，这个值一般为 1/4，即 8’h20 7:0： 控制内部 DLL 的精度，在龙芯 3 号中，这个值一般为 8’h80
DLL_CTRL_REG_0_7	31:0	0x0000	0x0-0xffffffff	第 7 数据组（DQ63-DQ56）DLL 控制信号 24： 控制内部 DLL 的使能信号，为 0 时 DLL 有效 23:16： 控制写数据（DQ）与 DQS 之间的相位关系，每个数值表示为（1/精度）＊360。在龙芯 3 号中，这个值一般为 1/4，即 8’h20 7:0： 控制内部 DLL 的精度，在龙芯 3 号中，这个值一般为 8’h80
CONF_CTL_36[63:0]	Offset: 0x240		DDR2 667：0x00001e0000001e00	
DLL_CTRL_REG_1_1	63:32	0x0000	0x0-0xffffffff	第 1 数据组 DLL 控制信号 15:8：读数据返回时，DQSn 的相位延迟。 5:0： DLL 测试控制信号，正常情况下为 8’h0
DLL_CTRL_REG_1_0	31:0	0x00000	0x0-0xffffffff	第 0 数据组 DLL 控制信号 15:8：读数据返回时，DQSn 的相位延迟。 5:0： DLL 测试控制信号，正常情况下为 8’h0
CONF_CTL_37[63:0]	Offset: 0x250		DDR2 667：0x00001e0000001e00	
DLL_CTRL_REG_1_3	63:32	0x0000	0x0-0xffffffff	第 3 数据组 DLL 控制信号 15:8：读数据返回时，DQSn 的相位延迟。 5:0： DLL 测试控制信号，正常情况下为 8’h0

DLL_CTRL_REG_1_2	31:0	0x000000	0x0-0xffff ffff	第 2 数据组 DLL 控制信号 15:8：读数据返回时，DQSn 的相位延迟。 5:0： DLL 测试控制信号，正常情况下为 8’h0
CONF_CTL_38[63:0]	Offset: 0x260		DDR2 667：0x00001e0000001e00	
DLL_CTRL_REG_1_5	63:32	0x00000	0x0-0xffff ffff	第 5 数据组 DLL 控制信号 15:8：读数据返回时，DQSn 的相位延迟。 5:0： DLL 测试控制信号，正常情况下为 8’h0
DLL_CTRL_REG_1_4	31:0	0x0000	0x0-0xffff ffff	第 4 数据组 DLL 控制信号 15:8:读数据返回时,DQSn 的相位延迟。5:0：DLL 测试控制信号，正常情况下为 8’h0
CONF_CTL_39[63:0]	Offset: 0x270		DDR2 667：0x00001e0000001e00	
DLL_CTRL_REG_1_7	63:32	0x0000	0x0-0xffff ffff	第 7 数据组 DLL 控制信号 15:8:读数据返回时,DQSn 的相位延迟。5:0：DLL 测试控制信号，正常情况下为 8’h0.
DLL_CTRL_REG_1_6	31:0	0x0000	0x0-0xffff ffff	第 6 数据组 DLL 控制信号 15:8：读数据返回时，DQSn 的相位延迟。 5:0： DLL 测试控制信号，正常情况下为 8’h0
CONF_CTL_40[63:0]	Offset: 0x280		DDR2 667：0x0000000000001e00	
DLL_OBS_REG_0_0	33:32	0x0	0x0-0x3	测试模式下的第 0 数据组 DLL 输出（只读）
DLL_CTRL_REG_1_8	31:0	0x00000	0x0-0xffff ffff	第 8 数据组 DLL 控制信号 15:8：读数据返回时，DQSn 的相位延迟。 5:0： DLL 测试控制信号，正常情况下为 8’h0
CONF_CTL_41[63:0]	Offset: 0x290		DDR2 667：0x0000000000000000	
DLL_OBS_REG_0_2	33:32	0x0	0x0-0x3	测试模式下的第 2 数据组 DLL 输出（只读）
DLL_OBS_REG_0_1	1:0	0x0	0x0-0x3	测试模式下的第 1 数据组 DLL 输出（只读）
CONF_CTL_42[63:0]	Offset: 0x2a0		DDR2 667：0x0x0000000000000000	
DLL_OBS_REG_0_4	33:32	0x0	0x0-0x3	测试模式下的第 4 数据组 DLL 输出（只读）
DLL_OBS_REG_0_3	1:0	0x0	0x0-0x3	测试模式下的第 3 数据组 DLL 输出（只读）
CONF_CTL_43[63:0]	Offset: 0x2b0		DDR2 667：0x0x0000000000000000	
DLL_OBS_REG_0_6	33:32	0x0	0x0-0x3	测试模式下的第 6 数据组 DLL 输出（只读）
DLL_OBS_REG_0_5	1:0	0x0	0x0-0x3	测试模式下的第 5 数据组 DLL 输出（只读）
CONF_CTL_44[63:0]	Offset: 0x2c0		DDR2 667：0x0000000000000000	

DLL_OBS_REG_0_8	33:32	0x0	0x0-0x3	测试模式下的第 8 数据组 DLL 输出（只读）
DLL_OBS_REG_0_7	1:0	0x0	0x0-0x3	测试模式下的第 7 数据组 DLL 输出（只读）
CONF_CTL_45[63:0]	Offset: 0x2d0		DDR2 667：0xf30029470000019d	
PHY_CTRL_REG_0_0	63:32	0x00000	0x0-0xffffffff	第 0 数据组时延控制. 28： 是否对读 DQS 使用去毛刺电路，指 gate 信号是否通过 PAD_feedback 延迟 27： 使用读 FIFO 有效信号自动控制读数据返回采样(1)，还是使用 26:24 中的固定时间采样(0) 26:24： 读数据返回采样完成时机，从内部时钟域采样的延迟。 21： 在 Read Leveling 模式下，采样数据总线的电平高低 20： 数据有效控制信号的电平，龙芯 3 号中为 0 19： 是否将写数据延迟再增加一周期 18： 读 DQS 采样是否提前 1/4 周期（与 clk_wr 同步） 17： 写数据/DQS 延迟是否增加半周期延迟 16： CAS 延迟是否为半周期 15:12：写 DQS 有效的起始时间，对于 DDR3 应该比 DDR2 提前一个周期打开，提供颗粒要求的 Preamble DQS 11:8： 写 DQS 有效的结束时间 6:4：写数据有效的起始时间 2:0： 写数据有效的结束时间
PAD_CTRL_REG_0	25:0	0x0000	0x0-0x3ffffff	引脚控制信号 25:22：对应 COMPZCP_dig 21:18：对应 COMPZCN_dig 17：对应引脚的 TQ1v8 16：对应内部反馈引脚的使能信号，低有效 15：对应内部反馈引脚的输出使能信号，低有效 14：对应数据选通引脚的输出使能信号，低有效 13：对应数据屏蔽引脚的输出使能信号，低有效 12：对应数据引脚的输出使能信号，低有效 11：对应引脚的 USEPAD

<table>
<tr><td></td><td></td><td></td><td></td><td>0：使用内部参考电压；
1：使用外部参考电压。
8：对应时钟引脚{1,3,5}的使能信号，高有效
7：对应时钟引脚{0,2,4}的使能信号，高有效
6：对应地址引脚的使能信号，低有效
5：对应引脚的 PROGB1v8
4：对应引脚的 PROGA1v8
用于控制引脚驱动能力
3：对应引脚的 ODTB
2：对应引脚的 ODTA
用于控制引脚 ODT 阻值大小
ODTA ODTB DDRII DDRIII
1 0 150 120
1 1 75 60
0 0 Disable Disable
1：对应引脚的 MODEZI1v8
对于龙芯 3 号应该设为 0。
0：对应引脚的 DDR1v8
0：对应 DDRII 的 1.8v 模式
1：对应 DDRIII 的 1.5v 模式</td></tr>
<tr><td>CONF_CTL_46[63:0]</td><td colspan="2">Offset: 0x2e0</td><td colspan="2">DDR2 667：0xf3002947f3002947</td></tr>
<tr><td>PHY_CTRL_REG_0_2</td><td>63:32</td><td>0x00000</td><td>0x0-0xffffffff</td><td>第 2 数据组时延控制.
28： 是否对读 DQS 使用去毛刺电路，指 gate 信号是否通过 PAD_feedback 延迟
27： 使用读 FIFO 有效信号自动控制读数据返回采样(1)，还是使用 26:24 中的固定时间采样(0)
26:24： 读数据返回采样完成时机，从内部时钟域采样的延迟。
21： 在 Read Leveling 模式下，采样数据总线的电平高低
20： 数据有效控制信号的电平，龙芯 3 号中为 0
19： 是否将写数据延迟再增加一周期
18： 读 DQS 采样是否提前 1/4 周期（与 clk_wr</td></tr>
</table>

				同步） 17： 写数据/DQS 延迟是否增加半周期延迟 16： CAS 延迟是否为半周期 15:12：写 DQS 有效的起始时间，对于 DDR3 应该比 DDR2 提前一个周期打开，提供颗粒要求的 Preamble DQS 11:8： 写 DQS 有效的结束时间 6:4：写数据有效的起始时间 2:0：写数据有效的结束时间
PHY_CTRL_REG_0_1	31:0	0x00000	0x0-0xffffffff	第 1 数据组时延控制. 28： 是否对读 DQS 使用去毛刺电路，指 gate 信号是否通过 PAD_feedback 延迟 27： 使用读 FIFO 有效信号自动控制读数据返回采样（1），还是使用 26:24 中的固定时间采样（0） 26:24： 读数据返回采样完成时机，从内部时钟域采样的延迟。 21： 在 Read Leveling 模式下，采样数据总线的电平高低 20： 数据有效控制信号的电平，龙芯 3 号中为 0 19： 是否将写数据延迟再增加一周期 18： 读 DQS 采样是否提前 1/4 周期（与 clk_wr 同步） 17： 写数据/DQS 延迟是否增加半周期延迟 16： CAS 延迟是否为半周期 15:12：写 DQS 有效的起始时间，对于 DDR3 应该比 DDR2 提前一个周期打开，提供颗粒要求的 Preamble DQS 11:8： 写 DQS 有效的结束时间 6:4：写数据有效的起始时间 2:0：写数据有效的结束时间
CONF_CTL_47[63:0]	Offset: 0x2f0		DDR2 667：0xf3002947f3002947	
PHY_CTRL_REG_0_4	63:32	0x00000	0x0-0xffffffff	第 4 数据组时延控制. 28： 是否对读 DQS 使用去毛刺电路，指 gate 信号是否通过 PAD_feedback 延迟

				27： 使用读 FIFO 有效信号自动控制读数据返回采样(1)，还是使用 26:24 中的固定时间采样(0) 26:24： 读数据返回采样完成时机，从内部时钟域采样的延迟。 21： 在 Read Leveling 模式下，采样数据总线的电平高低 20： 数据有效控制信号的电平，龙芯 3 号中为 0 19： 是否将写数据延迟再增加一周期 18： 读 DQS 采样是否提前 1/4 周期（与 clk_wr 同步） 17： 写数据/DQS 延迟是否增加半周期延迟 16： CAS 延迟是否为半周期 15:12：写 DQS 有效的起始时间，对于 DDR3 应该比 DDR2 提前一个周期打开，提供颗粒要求的 Preamble DQS 11:8： 写 DQS 有效的结束时间 6:4：写数据有效的起始时间 2:0：写数据有效的结束时间
PHY_CTRL_REG_0_3	31:0	0x0000	0x0-0xffffffff	第 3 数据组时延控制. 28： 是否对读 DQS 使用去毛刺电路，指 gate 信号是否通过 PAD_feedback 延迟 27： 使用读 FIFO 有效信号自动控制读数据返回采样(1)，还是使用 26:24 中的固定时间采样(0) 26:24： 读数据返回采样完成时机，从内部时钟域采样的延迟。 21： 在 Read Leveling 模式下，采样数据总线的电平高低 20： 数据有效控制信号的电平，龙芯 3 号中为 0 19： 是否将写数据延迟再增加一周期 18： 读 DQS 采样是否提前 1/4 周期（与 clk_wr 同步） 17： 写数据/DQS 延迟是否增加半周期延迟 16： CAS 延迟是否为半周期 15:12：写 DQS 有效的起始时间，对于 DDR3 应该比 DDR2 提前一个周期打开，提供颗粒要求的

				Preamble DQS 11:8： 写 DQS 有效的结束时间 6:4：写数据有效的起始时间 2:0：写数据有效的结束时间
CONF_CTL_48[63:0]	Offset: 0x300		DDR2 667：0xf3002947f3002947	
PHY_CTRL_REG_0_6	63:32	0x00000000	0x0-0xffffffff	第 6 数据组时延控制. 28： 是否对读 DQS 使用去毛刺电路，指 gate 信号是否通过 PAD_feedback 延迟 27： 使用读 FIFO 有效信号自动控制读数据返回采样(1)，还是使用 26:24 中的固定时间采样(0) 26:24： 读数据返回采样完成时机，从内部时钟域采样的延迟。 21： 在 Read Leveling 模式下，采样数据总线的电平高低 20： 数据有效控制信号的电平，龙芯 3 号中为 0 19： 是否将写数据延迟再增加一周期 18： 读 DQS 采样是否提前 1/4 周期（与 clk_wr 同步） 17： 写数据/DQS 延迟是否增加半周期延迟 16： CAS 延迟是否为半周期 15:12：写 DQS 有效的起始时间，对于 DDR3 应该比 DDR2 提前一个周期打开，提供颗粒要求的 Preamble DQS 11:8： 写 DQS 有效的结束时间 6:4：写数据有效的起始时间 2:0：写数据有效的结束时间
PHY_CTRL_REG_0_5	31:0	0x00000000	0x0-0xffffffff	第 5 数据组时延控制. 28： 是否对读 DQS 使用去毛刺电路，指 gate 信号是否通过 PAD_feedback 延迟 27： 使用读 FIFO 有效信号自动控制读数据返回采样(1)，还是使用 26:24 中的固定时间采样(0) 26:24： 读数据返回采样完成时机，从内部时钟域采样的延迟。 21： 在 Read Leveling 模式下，采样数据总线

				的电平高低 20： 数据有效控制信号的电平，龙芯 3 号中为 0 19： 是否将写数据延迟再增加一周期 18： 读 DQS 采样是否提前 1/4 周期（与 clk_wr 同步） 17： 写数据/DQS 延迟是否增加半周期延迟 16： CAS 延迟是否为半周期 15:12：写 DQS 有效的起始时间，对于 DDR3 应该比 DDR2 提前一个周期打开，提供颗粒要求的 Preamble DQS 11:8： 写 DQS 有效的结束时间 6:4：写数据有效的起始时间 2:0：写数据有效的结束时间
CONF_CTL_49[63:0]	Offset: 0x310		DDR2 667：0xf3002947f3002947	
PHY_CTRL_REG_0_8	63:32	0x00000	0x0-0xffffffff	第 8 数据组时延控制. 28： 是否对读 DQS 使用去毛刺电路，指 gate 信号是否通过 PAD_feedback 延迟 27： 使用读 FIFO 有效信号自动控制读数据返回采样(1)，还是使用 26:24 中的固定时间采样(0) 26:24： 读数据返回采样完成时机，从内部时钟域采样的延迟。 21： 在 Read Leveling 模式下，采样数据总线的电平高低 20： 数据有效控制信号的电平，龙芯 3 号中为 0 19： 是否将写数据延迟再增加一周期 18： 读 DQS 采样是否提前 1/4 周期（与 clk_wr 同步） 17： 写数据/DQS 延迟是否增加半周期延迟 16： CAS 延迟是否为半周期 15:12：写 DQS 有效的起始时间，对于 DDR3 应该比 DDR2 提前一个周期打开，提供颗粒要求的 Preamble DQS 11:8： 写 DQS 有效的结束时间 6:4：写数据有效的起始时间

				2:0：写数据有效的结束时间
PHY_CTRL_REG_0_7	31:0	0x0000	0x0-0xffffffff	第 7 数据组时延控制. 28： 是否对读 DQS 使用去毛刺电路，指 gate 信号是否通过 PAD_feedback 延迟 27： 使用读 FIFO 有效信号自动控制读数据返回采样(1)，还是使用 26:24 中的固定时间采样(0) 26:24： 读数据返回采样完成时机，从内部时钟域采样的延迟。 21： 在 Read Leveling 模式下，采样数据总线的电平高低 20： 数据有效控制信号的电平，龙芯 3 号中为 0 19： 是否将写数据延迟再增加一周期 18： 读 DQS 采样是否提前 1/4 周期（与 clk_wr 同步） 17： 写数据/DQS 延迟是否增加半周期延迟 16： CAS 延迟是否为半周期 15:12：写 DQS 有效的起始时间，对于 DDR3 应该比 DDR2 提前一个周期打开，提供颗粒要求的 Preamble DQS 11:8： 写 DQS 有效的结束时间 6:4：写数据有效的起始时间 2:0：写数据有效的结束时间
CONF_CTL_50[63:0]	Offset: 0x320		DDR2 667：0x07c0000007c00000	
PHY_CTRL_REG_1_1	63:32	0x00000	0x0-0xffffffff	第 1 数据组中 PAD 的终端电阻控制，发起读操作时，才会启用 31:28：终端电阻关闭时机控制，每个值表示半周期 27:24：终端电阻开启时机控制，从发送读命令后 4 拍开始计算 23：终端电阻的有效电平控制，对于龙芯 3 号应为 1 22：终端电阻的使能信号，为 1 时，使用动态方式控制终端电阻的使能；为 0 时，可以通过第 23 位 PAD 上的终端电阻永远有效（置 0）或永远无效（置 1）

				21：测试用信号，正常应为 0 20:16：测试用信号，正常应为 0 14:12：测试用信号，正常应为 0 11:8：读采样延时 1，其中只能 1 位有效，用于控制读 DQS 采样窗口关闭时机 7:0：读采样延时 0，其中只能 1 位有效，用于控制读 DQS 采样窗口打开时机
PHY_CTRL_REG_1_0	31:0	0x00000	0x0-0xffffffff	第 0 数据组中 PAD 的终端电阻控制，发起读操作时，才会启用 31:28：终端电阻关闭时机控制，每个值表示半周期 27:24：终端电阻开启时机控制，从发送读命令后 4 拍开始计算 23：终端电阻的有效电平控制，对于龙芯 3 号应为 1 22：终端电阻的使能信号，为 1 时，使用动态方式控制终端电阻的使能；为 0 时，可以通过第 23 位 PAD 上的终端电阻永远有效（置 0）或永远无效（置 1） 21：测试用信号，正常应为 0 20:16：测试用信号，正常应为 0 14:12：测试用信号，正常应为 0 11:8：读采样延时 1，其中只能 1 位有效，用于控制读 DQS 采样窗口关闭时机 7:0：读采样延时 0，其中只能 1 位有效，用于控制读 DQS 采样窗口打开时机
CONF_CTL_51[63:0]	Offset: 0x330		DDR2 667：0x07c0000007c00000	
PHY_CTRL_REG_1_3	63:32	0x00000	0x0-0xffffffff	第 3 数据组中 PAD 的终端电阻控制，发起读操作时，才会启用 31:28：终端电阻关闭时机控制，每个值表示半周期 27:24：终端电阻开启时机控制，从发送读命令后 4 拍开始计算 23：终端电阻的有效电平控制，对于龙芯 3 号应为 1

				22：终端电阻的使能信号，为 1 时，使用动态方式控制终端电阻的使能；为 0 时，可以通过第 23 位 PAD 上的终端电阻永远有效（置 0）或永远无效（置 1） 21：测试用信号，正常应为 0 20:16：测试用信号，正常应为 0 14:12：测试用信号，正常应为 0 11:8：读采样延时 1，其中只能 1 位有效，用于控制读 DQS 采样窗口关闭时机 7:0：读采样延时 0，其中只能 1 位有效，用于控制读 DQS 采样窗口打开时机
PHY_CTRL_REG_1_2	31:0	0x00000	0x0-0xffffffff	第 2 数据组中 PAD 的终端电阻控制，发起读操作时，才会启用 31:28：终端电阻关闭时机控制，每个值表示半周期 27:24：终端电阻开启时机控制，从发送读命令后 4 拍开始计算 23：终端电阻的有效电平控制，对于龙芯 3 号应为 1 22：终端电阻的使能信号，为 1 时，使用动态方式控制终端电阻的使能；为 0 时，可以通过第 23 位 PAD 上的终端电阻永远有效（置 0）或永远无效（置 1） 21：测试用信号，正常应为 0 20:16：测试用信号，正常应为 0 14:12：测试用信号，正常应为 0 11:8：读采样延时 1，其中只能 1 位有效，用于控制读 DQS 采样窗口关闭时机 7:0：读采样延时 0，其中只能 1 位有效，用于控制读 DQS 采样窗口打开时机
CONF_CTL_52[63:0]	Offset: 0x340		DDR2 667：0x07c0000007c00000	
PHY_CTRL_REG_1_5	63:32	0x00000	0x0-0xffffffff	第 5 数据组中 PAD 的终端电阻控制，发起读操作时，才会启用 31:28：终端电阻关闭时机控制，每个值表示半周期

				27:24：终端电阻开启时机控制，从发送读命令后 4 拍开始计算 23：终端电阻的有效电平控制，对于龙芯 3 号应为 1 22：终端电阻的使能信号，为 1 时，使用动态方式控制终端电阻的使能；为 0 时，可以通过第 23 位 PAD 上的终端电阻永远有效（置 0）或永远无效（置 1） 21：测试用信号，正常应为 0 20:16：测试用信号，正常应为 0 14:12：测试用信号，正常应为 0 11:8：读采样延时 1，其中只能 1 位有效，用于控制读 DQS 采样窗口关闭时机 7:0：读采样延时 0，其中只能 1 位有效，用于控制读 DQS 采样窗口打开时机
PHY_CTRL_REG_1_4	31:0	0x00000	0x0-0xffffffff	第 4 数据组中 PAD 的终端电阻控制，发起读操作时，才会启用 31:28：终端电阻关闭时机控制，每个值表示半周期 27:24：终端电阻开启时机控制，从发送读命令后 4 拍开始计算 23：终端电阻的有效电平控制，对于龙芯 3 号应为 1 22：终端电阻的使能信号，为 1 时，使用动态方式控制终端电阻的使能；为 0 时，可以通过第 23 位 PAD 上的终端电阻永远有效（置 0）或永远无效（置 1） 21：测试用信号，正常应为 0 20:16：测试用信号，正常应为 0 14:12：测试用信号，正常应为 0 11:8：读采样延时 1，其中只能 1 位有效，用于控制读 DQS 采样窗口关闭时机 7:0：读采样延时 0，其中只能 1 位有效，用于控制读 DQS 采样窗口打开时机
CONF_CTL_53[63:0]	Offset: 0x350		DDR2 667：0x07c0000007c00000	

PHY_CTRL_REG_1_7	63:32	0x00000	0x0-0xffff ffff	第 7 数据组中 PAD 的终端电阻控制，发起读操作时，才会启用 31:28：终端电阻关闭时机控制，每个值表示半周期 27:24：终端电阻开启时机控制，从发送读命令后 4 拍开始计算 23：终端电阻的有效电平控制，对于龙芯 3 号应为 1 22：终端电阻的使能信号，为 1 时，使用动态方式控制终端电阻的使能；为 0 时，可以通过第 23 位 PAD 上的终端电阻永远有效（置 0）或永远无效（置 1） 21：测试用信号，正常应为 0 20:16：测试用信号，正常应为 0 14:12：测试用信号，正常应为 0 11:8：读采样延时 1，其中只能 1 位有效，用于控制读 DQS 采样窗口关闭时机 7:0：读采样延时 0，其中只能 1 位有效，用于控制读 DQS 采样窗口打开时机
PHY_CTRL_REG_1_6	31:0	0x00000	0x0-0xffff ffff	第 6 数据组中 PAD 的终端电阻控制，发起读操作时，才会启用 31:28：终端电阻关闭时机控制，每个值表示半周期 27:24：终端电阻开启时机控制，从发送读命令后 4 拍开始计算 23：终端电阻的有效电平控制，对于龙芯 3 号应为 1 22：终端电阻的使能信号，为 1 时，使用动态方式控制终端电阻的使能；为 0 时，可以通过第 23 位 PAD 上的终端电阻永远有效（置 0）或永远无效（置 1） 21：测试用信号，正常应为 0 20:16：测试用信号，正常应为 0 14:12：测试用信号，正常应为 0 11:8：读采样延时 1，其中只能 1 位有效，用于

				控制读 DQS 采样窗口关闭时机 7:0：读采样延时 0，其中只能 1 位有效，用于控制读 DQS 采样窗口打开时机
CONF_CTL_54[63:0]	Offset: 0x360		DDR2 667：0x0800c00507c00000	
PHY_CTRL_REG_2	63:32	0x00000	0x0-0xffffffff	读写数据延迟控制 27：选择读数据缓冲类型，默认为 0 26：用于清除读返回缓冲区的数据，正常为 0 25：高速引脚使能，为 1 时，所有信号通过引脚向外传输的延迟减小 1 周期 16:13：设置读数据有效时机，从 FIFO 中收集数据返回控制器的延迟。如果从引脚到 FIFO 的延迟增加，这个值也必须增加 8：设置 DQS 信号输出是否为 DDR3 模式，DDR3 模式下，写 DQS 的 Preamble 将含有一个脉冲 5：测试模式信号，正常为 0 4：测试模式信号，正常为 0
PHY_CTRL_REG_1_8	31:0	0x00000	0x0-0xffffffff	第 8 数据组中的终端电阻控制 PAD 的终端电阻控制，发起读操作时，才会启用 31:28：终端电阻关闭时机控制，每个值表示半周期 27:24：终端电阻开启时机控制，从发送读命令后 4 拍开始计算 23：终端电阻的有效电平控制，对于龙芯 3 号应为 1 22：终端电阻的使能信号，为 1 时，使用动态方式控制终端电阻的使能；为 0 时，可以通过第 23 位 PAD 上的终端电阻永远有效（置 0）或永远无效（置 1） 21：测试用信号，正常应为 0 20:16：测试用信号，正常应为 0 14:12：测试用信号，正常应为 0 11:8：读采样延时 1，其中只能 1 位有效，用于控制读 DQS 采样窗口关闭时机 7:0：读采样延时 0，其中只能 1 位有效，用于控

				制读 DQS 采样窗口打开时机
CONF_CTL_55[63:0]	Offset: 0x370		DDR2 667：0x0000000000000000	
PHY_OBS_REG_0_1	63:32	0x00000	0x0-0xffff ffff	第 1 数据组测试用观测信号（只读）
PHY_OBS_REG_0_0	31:0	0x00000	0x0-0xffff ffff	第 0 数据组测试用观测信号（只读）
CONF_CTL_56[63:0]	Offset: 0x380		DDR2 667：0x0000000000000000	
PHY_OBS_REG_0_3	63:32	0x00000	0x0-0xffff ffff	第 3 数据组测试用观测信号（只读）
PHY_OBS_REG_0_2	31:0	0x00000	0x0-0xffff ffff	第 2 数据组测试用观测信号（只读）
CONF_CTL_57[63:0]	Offset: 0x390		DDR2 667：0x0000000000000000	
PHY_OBS_REG_0_5	63:32	0x000000	0x0-0xffff ffff	第 5 数据组测试用观测信号（只读）
PHY_OBS_REG_0_4	31:0	0x0000	0x0-0xffff ffff	第 4 数据组测试用观测信号（只读）
CONF_CTL_58[63:0]	Offset: 0x3a0		DDR2 667：0x0000000000000000	
PHY_OBS_REG_0_7	63:32	0x00000	0x0-0xffff ffff	第 7 数据组测试用观测信号（只读）
PHY_OBS_REG_0_6	31:0	0x00000	0x0-0xffff ffff	第 6 数据组测试用观测信号（只读）
CONF_CTL_59[63:0]	Offset: 0x3B10000		DDR2 667：0x0000000000000000	
PHY_OBS_REG_0_8	31:0	0x00000	0x0-0xffff ffff	第 8 数据组测试用观测信号（只读）
CONF_CTL_114[63:0]	Offset: 0x720		DDR2 667：0x0000000000000000	
RDLVL_GATE_REQ	56	0x0	0x0-0x1	用户请求读选通采样训练功能.（只写）
RDLVL_GATE_PREAMBLE_CHECK_EN	48	0x0	0x0-0x1	开启读选通采样训练时的前导采样检查
RDLVL_GATE_EN	40	0x0	0x0-0x1	使能 Read Leveling 时读选通采样训练，在初始化完成后会向颗粒发送命令，进行读 DQS 采样窗口的训练
RDLVL_EN	32	0x0	0x0-0x1	使能 Read Leveling 功能
RDLVL_BEGIN_DELAY_EN	24	0x0	0x0-0x1	使能 Read Leveling 寻找数据采样点功能

SWLVL_OP_DONE	8	0x0	0x0-0x1	用于指示软件 Leveling 是否完成（只读）
CONF_CTL_115[63:0]	Offset: 0x730		DDR2 667：0x0000000000000000	
RDLVL_OFFSET_DIR_7	56	0x0	0x0-0x1	第 7 数据组 Read Leveling 时中点的调整方向。为 0 时，中点计算为减去 rdlvl_offset_delay，为 1 则加。
RDLVL_OFFSET_DIR_6	48	0x0	0x0-0x1	第 6 数据组 Read Leveling 时中点的调整方向。为 0 时，中点计算为减去 rdlvl_offset_delay，为 1 则加。
RDLVL_OFFSET_DIR_5	40	0x0	0x0-0x1	第 5 数据组 Read Leveling 时中点的调整方向。为 0 时，中点计算为减去 rdlvl_offset_delay，为 1 则加。
RDLVL_OFFSET_DIR_4	32	0x0	0x0-0x1	第 4 数据组 Read Leveling 时中点的调整方向。为 0 时，中点计算为减去 rdlvl_offset_delay，为 1 则加。
RDLVL_OFFSET_DIR_3	24	0x0	0x0-0x1	第 3 数据组 Read Leveling 时中点的调整方向。为 0 时，中点计算为减去 rdlvl_offset_delay，为 1 则加。
RDLVL_OFFSET_DIR_2	16	0x0	0x0-0x1	第 2 数据组 Read Leveling 时中点的调整方向。为 0 时，中点计算为减去 rdlvl_offset_delay，为 1 则加。
RDLVL_OFFSET_DIR_1	8	0x0	0x0-0x1	第 1 数据组 Read Leveling 时中点的调整方向。为 0 时，中点计算为减去 rdlvl_offset_delay，为 1 则加。
RDLVL_OFFSET_DIR_0	0	0x0	0x0-0x1	第 0 数据组 Read Leveling 时中点的调整方向。为 0 时，中点计算为减去 rdlvl_offset_delay，为 1 则加。
CONF_CTL_116[63:0]	Offset: 0x740		DDR2 667：0x0100000000000000	
AXI1_PORT_ORDERING	57:56	0x0	0x0-0x3	内部端口 1 是否可乱序执行，对于龙芯 3 号无效
AXI0_PORT_ORDERING	49:48	0x0	0x0-0x3	内部端口 0 是否可乱序执行
WRLVL_REQ	40	0x0	0x0-0x1	用户请求开始 Write Leveling 训练功能.（只写）
WRLVL_INTERVAL_CT_EN	32	0x0	0x0-0x1	使能 Write Leveling 时间间隔功能
WEIGHTED_ROUND_ROBIN_WEIGHT_SHARING	24	0x0	0x0-0x1	Per-port pair shared arbitration for WRR
WEIGHTED_ROUND_ROBIN_LATENCY_ CONTROL	16	0x0	0x0-0x1	Free-running or limited WRR latency counters.

RDLVL_REQ	8	0x0	0x0-0x1	用户请求开始 Read Leveling 训练功能.（只写）
RDLVL_OFFSET_DIR_8	0	0x0	0x0-0x1	第 8 数据组 Read Leveling 时中点的调整方向。为 0 时，中点计算为减去 rdlvl_offset_delay，为 1 则加。
CONF_CTL_117[63:0]	Offset: 0x750		DDR2 667：0x0100000101020101	
WRLVL_CS	57:56	0x0	0x0-0x3	指示当前 Write Leveling 操作的片选信号
SW_LEVELING_MODE	49:48	0x0	0x0-0x3	定义软件 Leveling 操作的模式
RDLVL_CS	41:40	0x0	0x0-0x3	指示当前 Read Leveling 操作的片选信号
AXI2_W_PRIORITY	33:32	0x0	0x0-0x3	内部端口 2 的写操作优先级，对于龙芯 3 号无效
AXI2_R_PRIORITY	25:24	0x0	0x0-0x3	内部端口 2 的读操作优先级，对于龙芯 3 号无效
AXI2_PORT_ORDERING	17:16	0x0	0x0-0x3	内部端口 2 是否可乱序执行，对于龙芯 3 号无效
AXI1_W_PRIORITY	9:8	0x0	0x0-0x3	内部端口 1 的写操作优先级，对于龙芯 3 号无效
AXI1_R_PRIORITY	1:0	0x0	0x0-0x3	内部端口 1 的读操作优先级，对于龙芯 3 号无效
CONF_CTL_118[63:0]	Offset: 0x760		DDR2 667：0x0303030000020002	
AXI0_PRIORITY2_RELATIVE_PRIORITY	59:56	0x0	0x0-0xf	内部端口 0 优先级 2 的命令的相对优先级
AXI0_PRIORITY1_RELATIVE_PRIORITY	51:48	0x0	0x0-0xf	内部端口 0 优先级 1 的命令的相对优先级
AXI0_PRIORITY0_RELATIVE_PRIORITY	43:40	0x0	0x0-0xf	内部端口 0 优先级 0 的命令的相对优先级
ADDRESS_MIRRORING	35:32	0x0	0x0-0xf	指示哪个片选支持 Address mirroring 功能
TDFI_DRAM_CLK_DISABLE	26:24	0x0	0x0-0x7	从内部时钟关闭到外部时钟关闭的延迟设置
BSTLEN	18:16	0x0	0x0-0x7	设置控制器上向内存模块发送的 Burst 长度值
ZQ_REQ	9:8	0x0	0x0-0x3	用户请求开始 ZQ 调整功能
ZQ_ON_SREF_EXIT	1:0	0x0	0x0-0x3	定义在退出自刷新模式时 ZQ 调整功能的模式
CONF_CTL_119[63:0]	Offset: 0x770		DDR2 667：0x0101010202020203	
AXI2_PRIORITY2_RELATIVE_PRIORITY	59:56	0x0	0x0-0xf	内部端口 2 优先级 2 的命令的相对优先级，对于龙芯 3 号无效
AXI2_PRIORITY1_RELATIVE_PRIORITY	51:48	0x0	0x0-0xf	内部端口 2 优先级 1 的命令的相对优先级，对于龙芯 3 号无效
AXI2_PRIORITY0_RELATIVE_PRIORITY	43:40	0x0	0x0-0xf	内部端口 2 优先级 0 的命令的相对优先级，对于龙芯 3 号无效

AXI1_PRIORITY3_RELATIVE_PRIORITY	35:32	0x0	0x0-0xf	内部端口 1 优先级 3 的命令的相对优先级，对于龙芯 3 号无效
AXI1_PRIORITY2_RELATIVE_PRIORITY	27:24	0x0	0x0-0xf	内部端口 1 优先级 2 的命令的相对优先级，对于龙芯 3 号无效
AXI1_PRIORITY1_RELATIVE_PRIORITY	19:16	0x0	0x0-0xf	内部端口 1 优先级 1 的命令的相对优先级，对于龙芯 3 号无效
AXI1_PRIORITY0_RELATIVE_PRIORITY	11:8	0x0	0x0-0xf	内部端口 1 优先级 0 的命令的相对优先级，对于龙芯 3 号无效
AXI0_PRIORITY3_RELATIVE_PRIORITY	3:0	0x0	0x0-0xf	内部端口 0 优先级 3 的命令的相对优先级
CONF_CTL_120[63:0]	Offset: 0x780		DDR2 667：0x0102020400040c01	
TDFI_DRAM_CLK_ENABLE	59:56	0x0	0x0-0xf	从内部时钟有效到输出时钟有效的延迟
TDFI_CTRL_DELAY	51:48	0x0	0x0-0xf	从时钟有效到输出命令之间的延迟
RDLVL_GATE_DQ_ZERO_COUNT	43:40	0x0	0x0-0xf	设置读选通采样训练时，表求由 1 到 0 的 0 的个数
RDLVL_DQ_ZERO_COUNT	35:32	0x0	0x0-0xf	设置读 Read Leveling 时，表求由 1 到 0 的 0 的个数
LOWPOWER_REFRESH_ENABLE	27:24	0x0	0x0-0xf	使能低功耗模式下的刷新功能
DRAM_CLASS	19:16	0x0	0x0-0xf	定义控制器外接内存类型 110：DDR3 100：DDR2
BURST_ON_FLY_BIT	11:8	0x0	0x0-0xf	对 DRAM 发出的模式配置中的 burst-on-fly 位
AXI2_PRIORITY3_RELATIVE_PRIORITY	3:0	0x0	0x0-0xf	内部端口 2 优先级 3 的命令的相对优先级，对于龙芯 3 号无效
CONF_CTL_121[63:0]	Offset: 0x790		DDR2 667：0x281900000f000303	
WLMRD	61:56	0x00	0x0-0x3f	从对 DRAM 发送模式配置到 Write Leveling 的延迟
WLDQSEN	53:48	0x00	0x0-0x3f	从对 DRAM 发送模式配置到 Write Leveling 的的选通数据采样延迟
LOWPOWER_CONTROL	44:40	0x00	0x0-0x1f	低功耗模式使能 Bit 4: power down Bit 3: power down external Bit 2: self refresh

				Bit 1: external Bit 0: internal
LOWPOWER_AUTO_ENABLE	36:32	0x00	0x0-0x1f	使能当控制器内闲时自动进入低功耗模式 控制位与 LOWERPOWER_CONTROL 相同
ZQCS_CHIP	27:24	0x0	0x0-0xf	定义下次 ZQ 时的有效片选
WRR_PARAM_VALUE_ERR	19:16	0x0	0x0-0xf	Errors/warnings related to the WRR parameters. （只读）
TDFI_WRLVL_DLL	15:8	0x00	0x0-0xff	读操作到 Write Leveling 更新延迟线数目的最小周期
TDFI_RDLVL_DLL	7:0	0x00	0x0-0xff	读操作到 Read Leveling 更新延迟线数目的最小周期
CONF_CTL_122[63:0]	Offset: 0x7a0		DDR2 667：0x0000000000000000	
SWLVL_RESP_6	63:56	0x00	0x0-0xff	第 6 数据组的 Leveling 响应
SWLVL_RESP_5	55:48	0x00	0x0-0xff	第 5 数据组的 Leveling 响应
SWLVL_RESP_4	47:40	0x00	0x0-0xff	第 4 数据组的 Leveling 响应
SWLVL_RESP_3	39:32	0x00	0x0-0xff	第 3 数据组的 Leveling 响应
SWLVL_RESP_2	31:24	0x00	0x0-0xff	第 2 数据组的 Leveling 响应
SWLVL_RESP_1	23:16	0x00	0x0-0xff	第 1 数据组的 Leveling 响应
SWLVL_RESP_0	15:8	0x00	0x0-0xff	第 0 数据组的 Leveling 响应
CONF_CTL_123[63:0]	Offset: 0x7b0		DDR2 667：0x0000000000000000	
OBSOLETE	63:16			
SWLVL_RESP_8	15:8	0x00	0x0-0xff	第 8 数据组的 Leveling 响应
SWLVL_RESP_7	7:0	0x00	0x0-0xff	第 7 数据组的 Leveling 响应
CONF_CTL_124[63:0]	Offset: 0x7c0		DDR2 667：0x0000000000000000	
OBSOLETE				
CONF_CTL_125[63:0]	Offset: 0x7d0		DDR2 667：0x0000000000000000	
RDLVL_GATE_CLK_ADJUST_3	63:56	0x00	0x0-0xff	第 3 数据组中，读采样训练的起始值
RDLVL_GATE_CLK_ADJUST_2	55:48	0x00	0x0-0xff	第 2 数据组中，读采样训练的起始值
RDLVL_GATE_CLK_ADJUST_1	47:40	0x00	0x0-0xff	第 1 数据组中，读采样训练的起始值
RDLVL_GATE_CLK_ADJUS	39:32	0x00	0x0-0xff	第 0 数据组中，读采样训练的起始值

T_0				
CONF_CTL_126[63:0]	Offset: 0x7e0		DDR2 667：0x0000000000000000	
RDLVL_GATE_CLK_ADJUST_8	39:32	0x00	0x0-0xff	第 8 数据组中，读采样训练的起始值
RDLVL_GATE_CLK_ADJUST_7	31:24	0x00	0x0-0xff	第 7 数据组中，读采样训练的起始值
RDLVL_GATE_CLK_ADJUST_6	23:16	0x00	0x0-0xff	第 6 数据组中，读采样训练的起始值
RDLVL_GATE_CLK_ADJUST_5	15:8	0x00	0x0-0xff	第 5 数据组中，读采样训练的起始值
RDLVL_GATE_CLK_ADJUST_4	7:0	0x00	0x0-0xff	第 4 数据组中，读采样训练的起始值
CONF_CTL_127[63:0]	Offset: 0x7f0		DDR2 667：0x0000000000000000	
OBSOLETE				
CONF_CTL_128[63:0]	Offset: 0x800		DDR2 667：0x0000000000000000	
OBSOLETE				
CONF_CTL_129[63:0]	Offset: 0x810		DDR2 667：0x0000000000000000	
OBSOLETE				
CONF_CTL_130[63:0]	Offset: 0x820		DDR2 667：0x0420000c20400000	
TDFI_WRLVL_RESPLAT	63:56	0x00	0x0-0xff	Write Leveling 选通到响应有效的周期数
TDFI_RDLVL_RESPLAT	39:32	0x00	0x0-0xff	Read Leveling 选通到响应有效的周期数
REFRESH_PER_ZQ	23:16	0x00	0x0-0xff	自动 ZQCS 命令之间的刷新命令数目
CONF_CTL_131[63:0]	Offset: 0x830		DDR2 667：0x0000000000000c0a	
TMOD	15:8	0x00	0x0-0xff	DRAM 模式配置后需空闲的周期数
CONF_CTL_132[63:0]	Offset: 0x840		DDR2 667：0x0000640064000000	
AXI1_PRIORITY_RELAX	49:40	0x000	0x0-0x3ff	内部端口 1 上触发优先控制放松的计数器值，对于龙芯 3 号无效
AXI0_PRIORITY_RELAX[9:8]	33:32	0x0	0x0-0x3	内部端口 0 上触发优先控制放松的计数器值
AXI0_PRIORITY_RELAX[7:0]	31:24	0x00	0x0-0xff	内部端口 0 上触发优先控制放松的计数器值
CONF_CTL_133[63:0]	Offset: 0x850		DDR2 667：0x0000000000000064	
OUT_OF_RANGE_SOURCE_	57:48	0x000	0x0-0x3ff	访问地址溢出请求的 ID 号（只读）

ID				
ECC_U_ID	41:32	0x000	0x0-0x3ff	访问出现 2 位错请求的 ID 号（只读）
ECC_C_ID	25:16	0x000	0x0-0x3ff	访问出现 1 位错请求的 ID 号（只读）
AXI2_PRIORITY_RELAX	9:0	0x000	0x0-0x3ff	内部端口 2 上触发优先控制放松的计数器值，对于龙芯 3 号无效
CONF_CTL_134[63:0]	Offset: 0x860		DDR2 667：0x0000004000000000	
ZQCS	43:32	0x000	0x0-0xfff	ZQCS 命令需要的周期数
PORT_DATA_ERROR_ID	25:16	0x000	0x0-0x3ff	内部端口数据错请求的 ID 号（只读）
PORT_CMD_ERROR_ID	9:0	0x000	0x0-0x3ff	内部端口命令错请求的 ID 号（只读）
CONF_CTL_135[63:0]	Offset: 0x870		DDR2 667：0x0000000000000000	
OBSOLETE				
CONF_CTL_136[63:0]	Offset: 0x880		DDR2 667：0x0000000000000000	
OBSOLETE				
CONF_CTL_137[63:0]	Offset: 0x890		DDR2 667：0x0000000000000000	
OBSOLETE				
CONF_CTL_138[63:0]	Offset: 0x8a0		DDR2 667：0x00000000001c001c	
LOWPOWER_INTERNAL_CNT	63:48	0x0000	0x0-0xffff	Counts idle cycles to self-refresh with memory and controller clk gating.
LOWPOWER_EXTERNAL_CNT	47:32	0x0000	0x0-0xffff	Counts idle cycles to self-refresh with memory clock gating.
AXI2_EN_SIZE_LT_WIDTH_INSTR	31:16	0x0000	0x0-0xffff	使能内部端口 2 上的各种窄访问，对于龙芯 3 号无效
AXI1_EN_SIZE_LT_WIDTH_INSTR	15:0	0x0000	0x0-0xffff	使能内部端口 1 上的各种窄访问，对于龙芯 3 号无效
CONF_CTL_139[63:0]	Offset: 0x8b0		DDR2 667：0x0000000000000000	
LOWPOWER_POWER_DOWN_CNT	15:0	0x0000	0x0-0xffff	进入 Power Down 模式前的空闲周期数
LOWPOWER_REFRESH_HOLD	31:16	0x0000	0x0-0xffff	在时钟门控模式下，内存控制器 re-lock DLL 前的空闲周期数
LOWPOWER_SELF_REFRESH_CNT	47:32	0x0000	0x0-0xffff	进入内存自刷新模式前的空闲周期数
CONF_CTL_140[63:0]	Offset: 0x8c0		DDR2 667：0x0004000000000000	
OBSOLETE				

CONF_CTL_141[63:0]	Offset：0x8d0		DDR2 667：0x00000000c8000000	
CKE_INACTIVE[31:8]	55:32	0x0000000	0x0-0xffffffff	从输出 DDR_RESET 有效到 CKE 有效的时间间隔高位
CKE_INACTIVE [7:0]	31:24	0x00	0x0-0xff	从输出 DDR_RESET 有效到 CKE 有效的时间间隔低位
WRLVL_STATUS	17:0	0x00000	0x0-0x3ffff	最近一次 Write Leveling 操作状态（只读）
CONF_CTL_142[63:0]	Offset：0x8e0		DDR2 667：0x0000000000000050	
TRST_PWRON	31:0	0x0000000	0x0-0xffffffff	从 start 有效 500 拍后到 DDR_RESET 有效之间的延迟
CONF_CTL_143[63:0]	Offset：0x8f0		DDR2 667：0x0000000020202080	
DLL_CTRL_REG_2 [32]	32:32	0x0	0x0-0x1	输出时钟 DLL 使能信号，高有效
DLL_CTRL_REG_2 [31:0]	31:0	0x00000000	0x0-0xffffffff	输出时钟 DLL 控制 31:24：输出时钟 DLL 上 CLK4 与 CLK5 的延迟 23:16：输出时钟 DLL 上 CLK2 与 CLK3 的延迟 15:8：输出时钟 DLL 上 CLK0 与 CLK1 的延迟 7:0：输出时钟 DLL 上精度值
CONF_CTL_144[63:0]	Offset：0x900		DDR2 667：0x0000000000000000	
RDLVL_ERROR_STATUS[37:32]	37:32	0x00	0x0-0x3f	指示 RDLVL 发生错误时的状态
RDLVL_ERROR_STATUS[31:0]	31:0	0x00000000	0x0-0xffffffff	指示 RDLVL 发生错误时的状态
CONF_CTL_145[63:0]	Offset：0x910		DDR2 667：0x0000000000000000	
RDLVL_GATE_RESP_MASK[63:32]	63:32	0x00000000	0x0-0xffffffff	采样训练中读返回屏蔽
RDLVL_GATE_RESP_MASK[31:0]	31:0	0x00000000	0x0-0xffffffff	采样训练中读返回屏蔽
CONF_CTL_146[63:0]	Offset：0x920		DDR2 667：0x0000000000000000	
RDLVL_GATE_RESP_MASK[71:64]	7:0	0x00	0x0-0xff	采样训练中读返回屏蔽
CONF_CTL_147[63:0]	Offset：0x930		DDR2 667：0x0000000000000000	
RDLVL_RESP_MASK[63:32]	63:32	0x00000000	0x0-0xffffffff	Read Leveling 中读返回屏蔽
RDLVL_RESP_MASK[31:0	31:0	0x00000000	0x0-0xffff	Read Leveling 中读返回屏蔽

]			ffff	
CONF_CTL_148[63:0]	Offset: 0x940		DDR2 667：0x0301010000050500	
TDFI_RDLVL_EN	59:56	0x0	0x0-0xf	Read Leveling 使能到 Read Leveling 读之间的最小周期数
W2R_SAMECS_DLY	50:48	0x0	0x0-0x7	对同一个片选信号，写到读之间的附加延迟
W2R_DIFFCS_DLY	42:40	0x0	0x0-0x7	对不同片选信号，写到读之间的附加延迟
LVL_STATUS	34:32	0x0	0x0-0x7	Write Leveling，Read Leveling 与采样训练请求的状态，用于 LVL_REQ 中断（只读）
RDLVL_EDGE	24	0x0	0x0-0x1	Read Leveling 操作中，指明 DQS 上升沿有效或下降沿有效
CKSRX	19:16	0x0	0x0-0x0	退出自刷新模式的时钟周期延迟
CKSRE	11:8	0x0	0x0-0x0	进入自刷新模式的时钟周期延迟
RDLVL_RESP_MASK[71:64]	7:0	0x00	0x0-0xff	Read Leveling 中读返回屏蔽
CONF_CTL_149[63:0]	Offset: 0x950		DDR2 667：0x0000000000000a03	
INT_MASK	42:24	0x00	0x0-0x7fff	中断屏蔽 [18] = 所有中断位的或； [17] = 用户发起的 DLL 同步结束标志； [16] = DLL 锁定信号改变（锁定与非锁定状态的切换）； [15] = 读采样时钟出错； [14] = 一个读写训练操作完成； [13] = 一个读写训练请求已经发起； [12] = 一个写训练结果出错； [11] = 一个读采样训练结果出错； [10] = 一个读训练结果出错； [9] = ODT 被使能，同时 CAS Latency 为 3； [8] = DRAM 初始化完成； [7] = 内部端口数据错误； [6] = 内部端口命令错误； [5] = 发现多次 ECC 两位错； [4] = 发现一次 ECC 两位错； [3] = 发现多次 ECC 一位错；

				[2] = 发现一次 ECC 一位错； [1] = 发现多次超出内存物理空间的访问； [0] = 发现一次超出内存物理空间的访问
TXPDLL	23:8	0x0000	0x0-0xffff	DRAM TXPDLL parameter in cycles.
TDFI_WRLVL_EN	3:0	0x0	0x0-0xf	Write Leveling 使能到 Write Leveling 读操作最小周期数
CONF_CTL_150[63:0]	Offset: 0x960		DDR2 667：0x0604000000000000	
RDLAT_ADJ	60:56	0x00	0x0-0x1f	PHY 读延迟周期
WRLAT_ADJ	51:48	0x0	0x0-0xf	PHY 写延迟周期
SWLVL_START	40	0x0	0x0-0x1	软件 Leveling 模式下开始操作（只写）
SWLVL_LOAD	32	0x0	0x0-0x1	软件 Leveling 模式下装入操作（只写）
SWLVL_EXIT	24	0x0	0x0-0x1	软件 Leveling 模式下退出操作（只写）
INT_STATUS	18:0	0x00	0x0-0x7ffff	中断状态（只读） [18] = 所有中断位的或； [17] = 用户发起的 DLL 同步结束标志； [16] = DLL 锁定信号改变（锁定与非锁定状态的切换）； [15] = 读采样时钟出错； [14] = 一个读写训练操作完成； [13] = 一个读写训练请求已经发起； [12] = 一个写训练结果出错； [11] = 一个读采样训练结果出错； [10] = 一个读训练结果出错； [9] = ODT 被使能，同时 CAS Latency 为 3； [8] = DRAM 初始化完成； [7] = 内部端口数据错误； [6] = 内部端口命令错误； [5] = 发现多次 ECC 两位错； [4] = 发现一次 ECC 两位错； [3] = 发现多次 ECC 一位错； [2] = 发现一次 ECC 一位错； [1] = 发现多次超出内存物理空间的访问； [0] = 发现一次超出内存物理空间的访问

CONF_CTL_151[63:0]	Offset: 0x970		DDR2 667：0x000000000003e805	
CONCURRENTAP_WR_ONLY	56	0x0	0x0-0x1	写操作之后读操作之前是否通过等待写恢复时间来阻止并发的 auto-precharge 操作
CKE_STATUS	48	0x0	0x0-0x1	指示 CKE_STATUS（只读）
INT_ACK	41:24	0x00	0x0-0x3fff f	中断清除（只写） [17] = 用户发起的 DLL 同步结束标志； [16] = DLL 锁定信号改变（锁定与非锁定状态的切换）； [15] = 读采样时钟出错； [14] = 一个读写训练操作完成； [13] = 一个读写训练请求已经发起； [12] = 一个写训练结果出错； [11] = 一个读采样训练结果出错； [10] = 一个读训练结果出错； [9] = ODT 被使能，同时 CAS Latency 为 3； [8] = DRAM 初始化完成； [7] = 内部端口数据错误； [6] = 内部端口命令错误； [5] = 发现多次 ECC 两位错； [4] = 发现一次 ECC 两位错； [3] = 发现多次 ECC 一位错； [2] = 发现一次 ECC 一位错； [1] = 发现多次超出内存物理空间的访问； [0] = 发现一次超出内存物理空间的访问
DLL_RST_DELAY	23:8	0x0000	0x0-0xffff	DLL 复位最小周期数
DLL_RST_ADJ_DLY	7:0	0x00	0x0-0xff	配置 DLL 精度到 DLL 复位结束的最小周期数
CONF_CTL_152[63:0]	Offset: 0x980		DDR2 667：0x0001010001000101	
ZQ_IN_PROGRESS	56	0x0	0x0-0x1	指示 ZQ 操作正在进行（只读）
ZQCS_ROTATE	48	0x0	0x0-0x1	使能 ZQCS（short ZQ）轮流校正。当该位为 0 时，每次 ZQCS 请求命令会对系统中所有的片选进行校正，当该位置为 1 时，系统在每个 ZQCS 命令到来时只对一个片选进行校正，系统会轮流校正所有的片选。ZQCS 和 REFRESH_PER_ZQ 参数的设

				置应该与该位一致
WRLVL_REG_EN	40	0x0	0x0-0x1	使能写 wrlvl_delay 寄存器
WRLVL_EN	32	0x0	0x0-0x1	使能控制器的 Write Leveling 功能
RESYNC_DLL_PER_AREF_EN	24	0x0	0x0-0x1	使能在每个刷新命令之后 DLL 自动同步
RESYNC_DLL	16	0x0	0x0-0x1	发起一个 DLL 同步命令（只写）
RDLVL_REG_EN	8	0x0	0x0-0x1	使能写 rdlvl_delay 寄存器
RDLVL_GATE_REG_EN	0	0x0	0x0-0x1	使能写 rdlvl_gate_delay 寄存器
CONF_CTL_153[63:0]	Offset: 0x990		DDR2 667：0x0101020202010100	
W2W_SAMECS_DLY	58:56	0x0	0x0-0x7	对同一个片选的写命令到写命令的附加延时时钟周期数
W2W_DIFFCS_DLY	50:48	0x0	0x0-0x7	对不同片选的写命令到写命令的附加延时时钟周期数
TBST_INT_INTERVAL	42:40	0x0	0x0-0x7	DRAM burst 中断间隔周期数
R2W_SAMECS_DLY	34:32	0x0	0x0-0x7	对同一个片选的读命令到写命令的附加延时时钟周期数
R2W_DIFFCS_DLY	26:24	0x0	0x0-0x7	对不同片选的读命令到写命令的附加延时时钟周期数
R2R_SAMECS_DLY	18:16	0x0	0x0-0x7	对同一个片选的读命令到读命令的附加延时时钟周期数
R2R_DIFFCS_DLY	10:8	0x0	0x0-0x7	对不同片选的读命令到读命令的附加延时时钟周期数
AXI_ALIGNED_STROBE_DISABLE	2:0	0x0	0x0-0x7	当 AXI 端口的事务具有以下特征之一时：一、事务的起始地址和结束地址按用户字对齐；二、事务长度为一个用户字（128 位），禁止 AXI Strobe，每位对应一个 AXI 端口。 当设为 0 时，写操作会按照读-修改-写的顺序执行； 当设为 1 时，写操作作为一个标准的写操作（非读-修改-写的顺序）
CONF_CTL_154[63:0]	Offset: 0x9a0		DDR2 667：0x0707040200060100	
TDFI_WRLVL_LOAD	63:56	0x0	0x0-0xff	写 Leveling 延时数有效到第一个写 Leveling Load 命令的最小时钟周期数
TDFI_RDLVL_LOAD	55:48	0x0	0x0-0xff	读 Leveling 延时数有效到第一个读 Leveling

				Load 命令的最小时钟周期数
TCKESR	44:40	0x0	0x0-0x1f	自刷新进入到退出时 CKE 保持为低电平的最小时钟周期数
TCCD	36:32	0x0	0x0-0x1f	CAS#到 CAS#命令的延时
ADD_ODT_CLK_DIFFTYPE_DIFFCS	28:24	0x0	0x0-0x1f	为了满足 ODT 时序，对不同片选的不同命令之间插入的附加时钟周期数
TRP_AB	19:16	0x0	0x0-0xf	对所有 bank 的 trp 时间
ADD_ODT_CLK_SAMETYPE_DIFFCS	11:8	0x0	0x0-0xf	为了满足 ODT 时序，对不同片选的同类型命令之间插入的附加时钟周期数
ADD_ODT_CLK_DIFFTYPE_SAMECS	3:0	0x0	0x0-0xf	为了满足 ODT 时序，对同一片选的不同命令之间插入的附加时钟周期数
CONF_CTL_155[63:0]	Offset: 0x9b0		DDR2 667：0x0200010000000000	
ZQINIT	59:48	0x0	0x0-0xfff	DRAM 初始化过程中 ZQ 命令需要的时钟周期数
ZQCL	43:32	0x0	0x0-0xfff	正常的 ZQCL 命令需要的时钟周期数，它应等于 ZQINIT 的一半
TDFI_WRLVL_WW	25:16	0x0	0x0-0x3ff	连续两次写 leveling 命令之间的最小时钟周期数
TDFI_RDLVL_RR	9:0	0x0	0x0-0x3ff	连续两次读 leveling 命令之间的最小时钟周期数
CONF_CTL_156[63:0]	Offset: 0x9c0		DDR2 667：0x0a52000000000000	
MR0_DATA_0	62:48	0x0	0x0-0x7fff	对应片选 0 的模式寄存器 0 配置值
TDFI_PHYUPD_TYPE3	45:32	0x0	0x0-0x3fff	保存 DFI Tphyupd_type3 参数（只读）
TDFI_PHYUPD_TYPE2	29:16	0x0	0x0-0x3fff	保存 DFI Tphyupd_type2 参数（只读）
TDFI_PHYUPD_TYPE1	13:0	0x0	0x0-0x3fff	保存 DFI Tphyupd_type1 参数（只读）
CONF_CTL_157[63:0]	Offset: 0x9d0		DDR2 667：0x00440a520a520a52	
MR1_DATA_0	62:48	0x0	0x0-0x7fff	对应片选 0 的模式寄存器 1 配置值
MR0_DATA_3	46:32	0x0	0x0-0x7fff	对应片选 3 的模式寄存器 0 配置值
MR0_DATA_2	30:16	0x0	0x0-0x7fff	对应片选 2 的模式寄存器 0 配置值
MR0_DATA_1	14:0	0x0	0x0-0x7fff	对应片选 1 的模式寄存器 0 配置值
CONF_CTL_158[63:0]	Offset: 0x9e0		DDR2 667：0x0000004400440044	
MR2_DATA_0	62:48	0x0	0x0-0x7fff	对应片选 0 的模式寄存器 2 配置值
MR1_DATA_3	46:32	0x0	0x0-0x7fff	对应片选 3 的模式寄存器 1 配置值
MR1_DATA_2	30:16	0x0	0x0-0x7fff	对应片选 2 的模式寄存器 1 配置值

MR1_DATA_1	14:0	0x0	0x0-0x7fff	对应片选 1 的模式寄存器 1 配置值
CONF_CTL_159[63:0]	Offset: 0x9f0		DDR2 667：0x0000000000000000	
MR3_DATA_0	62:48	0x0	0x0-0x7fff	对应片选 0 的模式寄存器 3 配置值
MR2_DATA_3	46:32	0x0	0x0-0x7fff	对应片选 3 的模式寄存器 2 配置值
MR2_DATA_2	30:16	0x0	0x0-0x7fff	对应片选 2 的模式寄存器 2 配置值
MR2_DATA_1	14:0	0x0	0x0-0x7fff	对应片选 1 的模式寄存器 2 配置值
CONF_CTL_160[63:0]	Offset: 0xa00		DDR2 667：0x00ff000000000000	
DFI_WRLVL_MAX_DELAY	63:48	0x0	0x0-0xffff	Hareware Write leveling 会使用的延迟线的最大级数
MR3_DATA_3	46:32	0x0	0x0-0x7fff	对应片选 3 的模式寄存器 3 配置值
MR3_DATA_2	30:16	0x0	0x0-0x7fff	对应片选 2 的模式寄存器 3 配置值
MR3_DATA_1	14:0	0x0	0x0-0x7fff	对应片选 1 的模式寄存器 3 配置值
CONF_CTL_161[63:0]	Offset: 0xa10		DDR2 667：0x0000000000000000	
RDLVL_BEGIN_DELAY_3	63:48	0x0	0x0-0xffff	第 3 数据组中，Read Leveling 时从第一个 1 到 0 的延迟单元数目
RDLVL_BEGIN_DELAY_2	47:32	0x0	0x0-0xffff	第 2 数据组中，Read Leveling 时从第一个 1 到 0 的延迟单元数目
RDLVL_BEGIN_DELAY_1	31:16	0x0	0x0-0xffff	第 1 数据组中，Read Leveling 时从第一个 1 到 0 的延迟单元数目
RDLVL_BEGIN_DELAY_0	15:0	0x0	0x0-0xffff	第 0 数据组中，Read Leveling 时从第一个 1 到 0 的延迟单元数目
CONF_CTL_162[63:0]	Offset: 0xa20		DDR2 667：0x0000000000000000	
RDLVL_BEGIN_DELAY_7	63:48	0x0	0x0-0xffff	第 7 数据组中，Read Leveling 时从第一个 1 到 0 的延迟单元数目
RDLVL_BEGIN_DELAY_6	47:32	0x0	0x0-0xffff	第 6 数据组中，Read Leveling 时从第一个 1 到 0 的延迟单元数目
RDLVL_BEGIN_DELAY_5	31:16	0x0	0x0-0xffff	第 5 数据组中，Read Leveling 时从第一个 1 到 0 的延迟单元数目
RDLVL_BEGIN_DELAY_4	15:0	0x0	0x0-0xffff	第 4 数据组中，Read Leveling 时从第一个 1 到 0 的延迟单元数目
CONF_CTL_163[63:0]	Offset: 0xa30		DDR2 667：0x0000000000000000	
RDLVL_DELAY_2	63:48	0x0	0x0-0xffff	第 2 数据组中，Read Leveling 使用的延迟单元数目
RDLVL_DELAY_1	47:32	0x0	0x0-0xffff	第 1 数据组中，Read Leveling 使用的延迟单元

				数目
RDLVL_DELAY_0	31:16	0x0	0x0-0xffff	第 0 数据组中，Read Leveling 使用的延迟单元数目
RDLVL_BEGIN_DELAY_8	15:0	0x0	0x0-0xffff	第 8 数据组中，Read Leveling 时从第一个 1 到 0 的延迟单元数目
CONF_CTL_164[63:0]	Offset: 0xa40		DDR2 667：0x0000000000000000	
RDLVL _DELAY_6	63:48	0x0	0x0-0xffff	第 6 数据组中，Read Leveling 使用的延迟单元数目
RDLVL_DELAY_5	47:32	0x0	0x0-0xffff	第 5 数据组中，Read Leveling 使用的延迟单元数目
RDLVL_DELAY_4	31:16	0x0	0x0-0xffff	第 4 数据组中，Read Leveling 使用的延迟单元数目
RDLVL_DELAY_3	15:0	0x0	0x0-0xffff	第 3 数据组中，Read Leveling 使用的延迟单元数目
CONF_CTL_165[63:0]	Offset: 0xa50		DDR2 667：0x0000000000000000	
RDLVL_END_DELAY_1	63:48	0x0	0x0-0xffff	第 1 数据组中，Read Leveling 时从第一个 0 到 1 的延迟单元数目
RDLVL_END_DELAY_0	47:32	0x0	0x0-0xffff	第 0 数据组中，Read Leveling 时从第一个 0 到 1 的延迟单元数目
RDLVL_DELAY_8	31:16	0x0	0x0-0xffff	第 8 数据组中，Read Leveling 使用的延迟单元数目
RDLVL_DELAY_7	15:0	0x0	0x0-0xffff	第 7 数据组中，Read Leveling 使用的延迟单元数目
CONF_CTL_166[63:0]	Offset: 0xa60		DDR2 667：0x0000000000000000	
RDLVL_END_DELAY_5	63:48	0x0	0x0-0xffff	第 5 数据组中，Read Leveling 时从第一个 0 到 1 的延迟单元数目
RDLVL_END_DELAY_4	47:32	0x0	0x0-0xffff	第 4 数据组中，Read Leveling 时从第一个 0 到 1 的延迟单元数目
RDLVL_END_DELAY_3	31:16	0x0	0x0-0xffff	第 3 数据组中，Read Leveling 时从第一个 0 到 1 的延迟单元数目
RDLVL_END_DELAY_2	15:0	0x0	0x0-0xffff	第 2 数据组中，Read Leveling 时从第一个 0 到 1 的延迟单元数目
CONF_CTL_167[63:0]	Offset: 0xa70		DDR2 667：0x0000000000000000	
RDLVL_GATE_DELAY_0	63:48	0x0	0x0-0xffff	第 0 数据组中，采样时机到选通信号上升沿的延

				迟单元个数
RDLVL_END_DELAY_8	47:32	0x0	0x0-0xffff	第 8 数据组中，Read Leveling 时从第一个 0 到 1 的延迟单元数目
RDLVL_END_DELAY_7	31:16	0x0	0x0-0xffff	第 7 数据组中，Read Leveling 时从第一个 0 到 1 的延迟单元数目
RDLVL_END_DELAY_6	15:0	0x0	0x0-0xffff	第 6 数据组中，Read Leveling 时从第一个 0 到 1 的延迟单元数目
CONF_CTL_168[63:0]	Offset: 0xa80		DDR2 667：0x0000000000000000	
RDLVL_GATE_DELAY_4	63:48	0x0	0x0-0xffff	第 4 数据组中，采样时机到选通信号上升沿的延迟单元个数
RDLVL_GATE_DELAY_3	47:32	0x0	0x0-0xffff	第 3 数据组中，采样时机到选通信号上升沿的延迟单元个数
RDLVL_GATE_DELAY_2	31:16	0x0	0x0-0xffff	第 2 数据组中，采样时机到选通信号上升沿的延迟单元个数
RDLVL_GATE_DELAY_1	15:0	0x0	0x0-0xffff	第 1 数据组中，采样时机到选通信号上升沿的延迟单元个数
CONF_CTL_169[63:0]	Offset: 0xa90		DDR2 667：0x0000000000000000	
RDLVL_GATE_DELAY_8	63:48	0x0	0x0-0xffff	第 8 数据组中，采样时机到选通信号上升沿的延迟单元个数
RDLVL_GATE_DELAY_7	47:32	0x0	0x0-0xffff	第 7 数据组中，采样时机到选通信号上升沿的延迟单元个数
RDLVL_GATE_DELAY_6	31:16	0x0	0x0-0xffff	第 6 数据组中，采样时机到选通信号上升沿的延迟单元个数
RDLVL_GATE_DELAY_5	15:0	0x0	0x0-0xffff	第 5 数据组中，采样时机到选通信号上升沿的延迟单元个数
CONF_CTL_170[63:0]	Offset: 0xaa0		DDR2 667：0x0000ffff00000010	
RDLVL_MIDPOINT_DELAY_0	63:48	0x0	0x0-0xffff	当 Hardware read leveling 模块使能时，等于 rdlvl_begin_delay_0 和 rdlvl_end_delay_0 的中间值，否则，等于 rdlvl_delay_0(只读)
RDLVL_MAX_DELAY	47:32	0x0	0x0-0xffff	Read Leveling 延迟线的最大数目
RDLVL_GATE_REFRESH_INTERVAL	31:16	0x0	0x0-0xffff	两次自动 Gate Training 之间的最大刷新命令数（应设为 0）
RDLVL_GATE_MAX_DELAY	15:0	0x0	0x0-0xffff	采样延迟线的最大数目
CONF_CTL_171[63:0]	Offset: 0xab0		DDR2 667：0x0000000000000000	

RDLVL_MIDPOINT_DELAY_4	63:48	0x0	0x0-0xffff	当 Hardware read leveling 模块使能时，等于 rdlvl_begin_delay_4 和 rdlvl_end_delay_4 的中间值，否则，等于 rdlvl_delay_4(只读)
RDLVL_MIDPOINT_DELAY_3	47:32	0x0	0x0-0xffff	当 Hardware read leveling 模块使能时，等于 rdlvl_begin_delay_3 和 rdlvl_end_delay_3 的中间值，否则，等于 rdlvl_delay_3(只读)
RDLVL_MIDPOINT_DELAY_2	31:16	0x0	0x0-0xffff	当 Hardware read leveling 模块使能时，等于 rdlvl_begin_delay_2 和 rdlvl_end_delay_2 的中间值，否则，等于 rdlvl_delay_2(只读)
RDLVL_MIDPOINT_DELAY_1	15:0	0x0	0x0-0xffff	当 Hardware read leveling 模块使能时，等于 rdlvl_begin_delay_1 和 rdlvl_end_delay_1 的中间值，否则，等于 rdlvl_delay_1(只读)
CONF_CTL_172[63:0]	Offset: 0xac0		DDR2 667：0x0000000000000000	
RDLVL_MIDPOINT_DELAY_8	63:48	0x0	0x0-0xffff	当 Hardware read leveling 模块使能时，等于 rdlvl_begin_delay_8 和 rdlvl_end_delay_8 的中间值，否则，等于 rdlvl_delay_8(只读)
RDLVL_MIDPOINT_DELAY_7	47:32	0x0	0x0-0xffff	当 Hardware read leveling 模块使能时，等于 rdlvl_begin_delay_7 和 rdlvl_end_delay_7 的中间值，否则，等于 rdlvl_delay_7(只读)
RDLVL_MIDPOINT_DELAY_6	31:16	0x0	0x0-0xffff	当 Hardware read leveling 模块使能时，等于 rdlvl_begin_delay_6 和 rdlvl_end_delay_6 的中间值，否则，等于 rdlvl_delay_6(只读)
RDLVL_MIDPOINT_DELAY_5	15:0	0x0	0x0-0xffff	当 Hardware read leveling 模块使能时，等于 rdlvl_begin_delay_5 和 rdlvl_end_delay_5 的中间值，否则，等于 rdlvl_delay_5(只读)
CONF_CTL_173[63:0]	Offset: 0xad0		DDR2 667：0x0000000000000000	
RDLVL_OFFSET_DELAY_3	63:48	0x0	0x0-0xffff	第 3 数据组中，到 Read Leveling 中点的偏移
RDLVL_OFFSET_DELAY_2	47:32	0x0	0x0-0xffff	第 2 数据组中，到 Read Leveling 中点的偏移
RDLVL_OFFSET_DELAY_1	31:16	0x0	0x0-0xffff	第 1 数据组中，到 Read Leveling 中点的偏移
RDLVL_OFFSET_DELAY_0	15:0	0x0	0x0-0xffff	第 0 数据组中，到 Read Leveling 中点的偏移
CONF_CTL_174[63:0]	Offset: 0xae0		DDR2 667：0x0000000000000000	
RDLVL_OFFSET_DELAY_7	63:48	0x0	0x0-0xffff	第 7 数据组中，到 Read Leveling 中点的偏移
RDLVL_OFFSET_DELAY_6	47:32	0x0	0x0-0xffff	第 6 数据组中，到 Read Leveling 中点的偏移
RDLVL_OFFSET_DELAY_5	31:16	0x0	0x0-0xffff	第 5 数据组中，到 Read Leveling 中点的偏移
RDLVL_OFFSET_DELAY_4	15:0	0x0	0x0-0xffff	第 4 数据组中，到 Read Leveling 中点的偏移

CONF_CTL_175[63:0]	Offset: 0xaf0		DDR2 667：0x0000000000000000	
WRLVL_DELAY_1	63:48	0x0	0x0-0xffff	第 1 数据组中，控制写 DQS 经 DLL 延迟数
WRLVL_DELAY_0	47:32	0x0	0x0-0xffff	第 0 数据组中，控制写 DQS 经 DLL 延迟数
RDLVL_REFRESH_INTERVAL	31:16	0x0	0x0-0xffff	两次自动 Read Leveling 之间的最大刷新命令数（应设为 0）
RDLVL_OFFSET_DELAY_8	15:0	0x0	0x0-0xffff	第 8 数据组中，到 Read Leveling 中点的偏移
CONF_CTL_176[63:0]	Offset: 0xb00		DDR2 667：0x0000000000000000	
WRLVL_DELAY_5	63:48	0x0	0x0-0xffff	第 5 数据组中，控制写 DQS 经 DLL 延迟数
WRLVL_DELAY_4	47:32	0x0	0x0-0xffff	第 4 数据组中，控制写 DQS 经 DLL 延迟数
WRLVL_DELAY_3	31:16	0x0	0x0-0xffff	第 3 数据组中，控制写 DQS 经 DLL 延迟数
WRLVL_DELAY_2	15:0	0x0	0x0-0xffff	第 2 数据组中，控制写 DQS 经 DLL 延迟数
CONF_CTL_177[63:0]	Offset: 0xb10		DDR2 667：0x0000000000000000	
WRLVL_REFRESH_INTERVAL	63:48	0x0	0x0-0xffff	两次自动 Write Leveling 之间的最大刷新命令数（应设为 0）
WRLVL_DELAY_8	47:32	0x0	0x0-0xffff	第 8 数据组中，控制写 DQS 经 DLL 延迟数
WRLVL_DELAY_7	31:16	0x0	0x0-0xffff	第 7 数据组中，控制写 DQS 经 DLL 延迟数
WRLVL_DELAY_6	15:0	0x0	0x0-0xffff	第 6 数据组中，控制写 DQS 经 DLL 延迟数
CONF_CTL_178[63:0]	Offset: 0xb20		DDR2 667：0x00000c2d00000c2d	
TDFI_RDLVL_RESP	63:32	0x0	0x0-0xffff	保存 DFI Trdlvl_resp 时间参数
TDFI_RDLVL_MAX	31:0	0x0	0x0-0xffff	保存 DFI Trdlvl_max 时间参数
CONF_CTL_179[63:0]	Offset: 0xb30		DDR2 667：0x00000c2d00000c2d	
TDFI_WRLVL_RESP	63:32	0x0	0x0-0xffff	保存 DFI Twrlvl_resp 时间参数
TDFI_WRLVL_MAX	31:0	0x0	0x0-0xffff	保存 DFI Twrlvl_max 时间参数

9 HyperTransport 控制器

龙芯 3B1000 中，HyperTransport 总线用于实现外部设备连接以及多芯片互联。用于外设连接时，可由用户程序自由选择是否支持 IO Cache 一致性（通过地址窗口 Uncache 进行设置，详见 9.4.2 节）：当配置为支持 Cache 一致性模式时，IO 设备对内 DMA 的访问对于 Cache 层次透明，即由硬件自动维护其一致性,而无需软件通过程序 Cache 指令进行维护；当 HyperTransport 总线用于多芯片互联时，HT0 控制器(初始地址为 0x0C00_0000_0000 – 0x0DFF_FFFF_FFFF)可通过硬件自动维护各个 CPU 之间的 Cache 一致性，而 HT1 控制器(初始地址为 0x0E00_0000_0000 – 0x0FFF_FFFF_FFFF)不支持片间 Cache 一致性维护，详见 9.6 节。

HyperTransport 控制器最高支持双向 16 位宽度以及 800MHz 运行频率。在系统自动初始化建立连接后，用户程序可以通过修改协议中相应的配置寄存器，实现对宽度和运行频率的更改，并重新进行初始化，具体方法见 9.1 节。

龙芯 3B1000 HyperTransport 控制器的主要特征如下：

- 支持 200/400/800MHz 运行频率
- 支持 8/16 位宽度
- 每个 HT 控制器（HT0/HT1）可以配置为两个 8 位 HT 控制器
- 总线控制信号（包括 PowerOK，Rstn，LDT_Stopn）方向可配置
- 外设 DMA 空间 Cache/Uncache 可配置
- HT0 控制器用于多片互联时可配置为 Cache 一致性模式

9.1 HyperTransport 硬件设置及初始化

HyperTransport 总线由传输信号总线和控制信号引脚等组成，下表给出了 HyperTransport 总线相关的引脚及其功能描述。

表 9-1 HyperTransport 总线相关引脚信号

引脚	名称	描述
{PCI_Config[7], PCI_Config[0]}	HT 周边信号电压控制	00：将 HyperTransport 周边信号作为 1.8v 信号，这些信号包括 HT_8x2，HT_Mode，HT_Powerok，HT_Rstn，HT_Ldt_Stopn，HT_Ldt_Reqn。 01：保留。 10：将 HyperTransport 周边信号作为 2.5v 信号。 11：将 HyperTransport 周边信号作为 3.3v 信号。
HT0_8x2	总线宽度配置	1：将 16 位 HyperTransport 总线配置为两个独立的 8 位总线，分别由两个独立的控制器控制，地址空间的区分为 HT0_Lo：address[40] = 0; HT0_Hi：address[40] = 1; 0:将 16 位 HyperTransport 总线作为一个 16 位总线使用,由 HT0_Lo

		控制，地址空间为 HT0_Lo 的地址，即 address[40] =0；HT0_Hi 所有信号无效。
HT0_Lo_mode	主设备模式	1：将 HT0_Lo 设为主设备模式，这个模式下，总线控制信号等由 HT0_Lo 驱动，这些控制信号包括 HT0_Lo_Powerok，HT0_Lo_Rstn，HT0_Lo_Ldt_Stopn。这个模式下，这些控制信号也可以为双向驱动。同时这个引脚决定（取反）寄存器“Act as Slave”的初始值，这个寄存器为 0 时，HyperTransport 总线上的包中的 Bridge 位为 1，否则为 0。另外，这个寄存器为 0 时，如果 HyperTransport 总线上的请求地址没有在控制器的接收窗口命中时，将作为 P2P 请求重新发回总线，如果这个寄存器为 1 时，没有命中，则作为错误请求做出响应。 0：将 HT0_Lo 设为从设备模式，这个模式下，总线控制信号等由对方设备驱动，这些控制信号包括 HT0_Lo_Powerok，HT0_Lo_Rstn，HT0_Lo_Ldt_Stopn。这个模式下，这些控制信号由对方设备驱动，如果没有被正确驱动，则 HT 总线不能正确工作。
HT0_Lo_Powerok	总线 Powerok	HyperTransport 总线 Powerok 信号， HT0_Lo_Mode 为 1 时，由 HT0_Lo 控制； HT0_Lo_Mode 为 0 时，由对方设备控制。
HT0_Lo_Rstn	总线 Rstn	HyperTransport 总线 Rstn 信号， HT0_Lo_Mode 为 1 时，由 HT0_Lo 控制； HT0_Lo_Mode 为 0 时，由对方设备控制。
HT0_Lo_Ldt_Stopn	总线 Ldt_Stopn	HyperTransport 总线 Ldt_Stopn 信号， HT0_Lo_Mode 为 1 时，由 HT0_Lo 控制； HT0_Lo_Mode 为 0 时，由对方设备控制。
HT0_Lo_Ldt_Reqn	总线 Ldt_Reqn	HyperTransport 总线 Ldt_Reqn 信号，
HT0_Hi_mode	主设备模式	1：将 HT0_Hi 设为主设备模式，这个模式下，总线控制信号等由 HT0_Hi 驱动，这些控制信号包括 HT0_Hi_Powerok，HT0_Hi_Rstn，HT0_Hi_Ldt_Stopn。这个模式下，这些控制信号也可以为双向驱动。同时这个引脚决定（取反）寄存器“Act as Slave”的初始值，这个寄存器为 0 时，HyperTransport 总线上的包中的 Bridge 位为 1，否则为 0。另外，这个寄存器为 0 时，如果 HyperTransport 总线上的请求地址没有在控制器的接收窗口命中时，将作为 P2P 请求重新发回总线，如果这个寄存器为 1 时，没有命中，则作为错误请求做出响应。 0：将 HT0_Hi 设为从设备模式，这个模式下，总线控制信号等由对方设备驱动，这些控制信号包括 HT0_Hi_Powerok，HT0_Hi_Rstn，HT0_Hi_Ldt_Stopn。这个模式下，这些控制信号由对方设备驱动，如果没有被正确驱动，则 HT 总线不能正确工作。
HT0_Hi_Powerok	总线 Powerok	HyperTransport 总线 Powerok 信号， HT0_Lo_Mode 为 1 时，由 HT0_Hi 控制； HT0_Lo_Mode 为 0 时，由对方设备控制。 HT0_8x2 为 1 时，控制高 8 位总线； HT0_8x2 为 0 时，无效。
HT0_Hi_Rstn	总线 Rstn	HyperTransport 总线 Rstn 信号， HT0_Lo_Mode 为 1 时，由 HT0_Hi 控制； HT0_Lo_Mode 为 0 时，由对方设备控制。 HT0_8x2 为 1 时，控制高 8 位总线； HT0_8x2 为 0 时，无效。
HT0_Hi_Ldt_Stopn	总线 Ldt_Stopn	HyperTransport 总线 Ldt_Stopn 信号， HT0_Lo_Mode 为 1 时，由 HT0_Hi 控制； HT0_Lo_Mode 为 0 时，由对方设备控制。 HT0_8x2 为 1 时，控制高 8 位总线； HT0_8x2 为 0 时，无效。
HT0_Hi_Ldt_Reqn	总线 Ldt_Reqn	HyperTransport 总线 Ldt_Reqn 信号， HT0_8x2 为 1 时，控制高 8 位总线； HT0_8x2 为 0 时，无效。

HT0_Rx_CLKp[1:0] HT0_Rx_CLKn[1:0] HT0_Tx_CLKp[1:0] HT0_Tx_CLKp[1:0]	CLK[1:0]	HyperTransport 总线 CLK 信号 HT0_8x2 为 1 时，CLK[1]由 HT0_Hi 控制 CLK[0]由 HT0_Lo 控制 HT0_8x2 为 0 时，CLK[1:0]由 HT0_Lo 控制
HT0_Rx_CTLp[1:0] HT0_Rx_CTLn[1:0] HT0_Tx_CTLp[1:0] HT0_Tx_CTLn[1:0]	CTL[1:0]	HyperTransport 总线 CTL 信号 HT0_8x2 为 1 时，CTL[1]由 HT0_Hi 控制 CTL[0]由 HT0_Lo 控制 HT0_8x2 为 0 时，CTL[1]无效 CTL[0]由 HT0_Lo 控制
HT0_Rx_CADp[15:0] HT0_Rx_CADn[15:0] HT0_Tx_CADp[15:0] HT0_Tx_CADn[15:0]	CAD[15:0]	HyperTransport 总线 CAD 信号 HT0_8x2 为 1 时，CAD[15:8]由 HT0_Hi 控制 CAD[7:0]由 HT0_Lo 控制 HT0_8x2 为 0 时，CAD[15:0]由 HT0_Lo 控制

HyperTransport 的初始化在每次复位完成后自动开始，冷启动后 HyperTransport 总线将自动工作在最低频率（200MHz)与最小宽度(8bit)，并尝试进行总线初始化握手。初始化是否已处于完成状态可以由寄存器“Init Complete”（见 9.5.2 节）读出。初始化完成后，总线的宽度可以由寄存器“Link Width Out”与“Link Width In”（见 9.5.2 节）读出。初始化完成后，用户可重写寄存器“Link Width Out”、“Link Width In”以及“Link Freq”，同时还需要配置对方设备的相应寄存器，配置完成后需要热复位总线或者通过“HT_Ldt_Stopn”信号进行重新初始化操作，以便使寄存器重写后的值生效。重新初始化完成后 HyperTransport 总线将工作在新的频率和宽度。需要注意的是，HyperTransport 两端的设备的配置需要一一对应，否则将使得 HyperTransport 接口不能正常工作。

9.2 HyperTransport 协议支持

龙芯 3B1000 的 HyperTransport 总线支持 1.03 版协议中的大部分命令，并且在支持多芯片互联的扩展一致性协议中加入了一些扩展指令。在以上两种模式下，HyperTransport 接收端可接收的命令如下表所示。需要注意的是，不支持 HyperTransport 总线的原子操作命令。

表 9-2 HyperTransport 接收端可接收的命令

编码	通道	命令	标准模式	扩展（一致性）
000000	-	NOP	空包或流控	
000001	NPC	FLUSH	无操作	
x01xxx	NPC or PC	Write	bit 5: 0 - Nonposted 1 - Posted bit 2: 0 - Byte 1 - Doubleword bit 1: Don’t Care bit 0: Don’t Care	bit 5: 必为 1, POSTED bit 2: 0 - Byte 1 - Doubleword bit 1: Don’t Care bit 0: 必为 1
01xxxx	NPC	Read	bit 3: Don’t Care bit 2: 0 - Byte 1 - Doubleword bit 1: Don’t Care	bit 3: Don’t Care bit 2: 0 - Byte 1 - Doubleword bit 1: Don’t Care

			bit 0：Don’t Care	bit 0：必为 1
110000	R	RdResponse	读操作返回	
110011	R	TgtDone	写操作返回	
110100	PC	WrCoherent	----	写命令扩展
110101	PC	WrAddr	----	写地址扩展
111000	R	RespCoherent	----	读响应扩展
111001	NPC	RdCoherent	----	读命令扩展
111010	PC	Broadcast	无操作	
111011	NPC	RdAddr	----	读地址扩展
111100	PC	FENCE	保证序关系	
111111	-	Sync/Error	Sync/Error	

对于发送端，在两种模式下会向外发送的命令如下表所示。

表 9-3 两种模式下会向外发送的命令

编码	通道	命令	标准模式	扩展（一致性）
000000	-	NOP	空包或流控	
x01x0x	NPC or PC	Write	bit 5：0 - Nonposted 1 - Posted bit 2：0 - Byte 1 - Doubleword bit 0：必为 0	bit 5：必为 1，POSTED bit 2：0 - Byte 1 - Doubleword bit 0：必为 1
010x0x	NPC	Read	bit 2：0 - Byte 1 - Doubleword bit 0：Don’t Care	bit 2：0 - Byte 1 - Doubleword bit 0：必为 1
110000	R	RdResponse	读操作返回	
110011	R	TgtDone	写操作返回	
110100	PC	WrCoherent	----	写命令扩展
110101	PC	WrAddr	----	写地址扩展
111000	R	RespCoherent	----	读响应扩展
111001	NPC	RdCoherent	----	读命令扩展
111011	NPC	RdAddr	----	读地址扩展
111111	-	Sync/Error	只会转发	

9.3 HyperTransport 中断支持

HyperTransport 控制器提供了 256 个中断向量，可以支持 Fix，Arbiter 等类型的中断，但是，没有对硬件自动 EOI 提供支持。对于以上两种支持类型的中断，控制器在接收之后会自动写入中断寄存器中，并根据中断屏蔽寄存器的设置对系统中断控制器进行中断通知。具体的中断控制请见 9.5 节中的中断控制寄存器组。

另外，控制器对 PIC 中断做了专门的支持，以加速该类型的中断处理。

一个典型的 PIC 中断由下述步骤完成：①PIC 控制器向系统发送 PIC 中断请求；②系统向 PIC 控制器发送中断向量查询；③PIC 控制器向系统发送中断向量号；④系统清除 PIC 控制器上的对应中断。只有上述 4 步都完成后，PIC 控制器才会对系统发出下一个中断。对于

龙芯 3B1000 HyperTransport 控制器，将自动进行前 3 步的处理，并将 PIC 中断向量写入 256 个中断向量中的对应位置。而软件系统在处理了该中断之后，需要进行第 4 步处理，即向 PIC 控制器发出清中断。之后开始下一个中断的处理过程。

9.4 HyperTransport 地址窗口

9.4.1 HyperTransport空间

龙芯 3B1000 处理器中，默认的 4 个 HyperTransport 接口的地址窗口分布如下：

表 9-4 默认的 4 个 HyperTransport 接口的地址窗口分布

基地址	结束地址	大小	定义
0x0E00_0000_0000	0x0EFF_FFFF_FFFF	1 Tbytes	HT0_LO 窗口
0x0F00_0000_0000	0x0FFF_FFFF_FFFF	1 Tbytes	HT0_HI 窗口
0x1E00_0000_0000	0x1EFF_FFFF_FFFF	1 Tbytes	HT1_LO 窗口
0x1F00_0000_0000	0x1FFF_FFFF_FFFF	1 Tbytes	HT1_HI 窗口

在默认情况下（未对系统地址窗口另行配置），软件根据上述地址空间对各个 HyperTransport 接口进行访问，此外，软件还可以通过对交叉开关上的地址窗口进行配置实现用其它的地址空间对其进行访问（详见 2.5 节）。每个 HyperTransport 接口的内部 40 位地址空间其地址窗口分布如下表所示。

表 9-5 龙芯 3 号处理器 HyperTransport 接口内部的地址窗口分布

基地址	结束地址	大小	定义
0x00_0000_0000	0xFC_FFFF_FFFF	1012 Gbytes	MEM 空间
0xFD_0000_0000	0xFD_F7FF_FFFF	3968 Mbytes	保留
0xFD_F800_0000	0xFD_F8FF_FFFF	16 Mbytes	中断
0xFD_F900_0000	0xFD_F90F_FFFF	1 Mbyte	PIC 中断响应
0xFD_F910_0000	0xFD_F91F_FFFF	1 Mbyte	系统信息
0xFD_F920_0000	0xFD_FAFF_FFFF	30 Mbytes	保留
0xFD_FB00_0000	0xFD_FBFF_FFFF	16 Mbytes	HT 控制器配置空间
0xFD_FC00_0000	0xFD_FDFF_FFFF	32 Mbytes	I/O 空间
0xFD_FE00_0000	0xFD_FFFF_FFFF	32 Mbytes	HT 总线配置空间
0xFE_0000_0000	0xFF_FFFF_FFFF	8 Gbytes	保留

9.4.2 HyperTransport控制器内部窗口配置

龙芯 3B1000 处理器的 HyperTransport 接口中提供了多种丰富的地址窗口供用户使用，这些地址窗口的作用和功能描述如下表所示。

表 9-6 龙芯 3B1000 处理器 HyperTransport 接口中提供的地址窗口

地址窗口	窗口数	接受总线	作用	备注
接收窗口（窗口配置见 9.5.4 节）	3	HyperTransport	判断是否接收 HyperTransport 总线上发出的访问。	处于主桥模式时（即配置寄存器中 act_as_slave 为 0），只有落在这些地址窗口中的访问会被内部总线所响应，其它访问将会被认为是 P2P 访问重新发回到 HyperTransport 总线上；处于设备模式时（即配置寄存器中 act_as_slave 为 1），只有落在这些地址窗口中的访问会被内部总线所接收并处理，其它访问将会按照协议给出错误返回。
Post 窗口（窗口配置见 9.5.8 节）	2	内部总线	判断是否将内部总线对 HyperTransport 总线的写访问作为 Post Write	落在这些地址空间中的对外写访问将作为 Post Write。 Post Write：HyperTransport 协议中，这种写访问不需要等待写完成响应，即在控制器向总线发出这个写访问之后就将对处理器进行写访问完成响应。
可预取窗口（窗口配置见 9.5.9 节）	2	内部总线	判断是否接收内部的 Cache 访问，取指访问。	当处理器核乱序执行时，会对总线发出一些猜测读访问或是取指访问，这种访问对于某些 IO 空间是错误的。在默认情况下，这种访问 HT 控制器将直接返回而不对 HyperTransport 总线进行访问。通过这些窗口可以使能对 HyperTransport 总线的这类访问。
Uncache 窗口（窗口配置见 9.5.10 节）	2	HyperTransport	判断是否将 HyperTransport 总线上的访问作为对内部的 Uncache 访问	龙芯 3B1000 处理器内部的 IO DMA 访问，在情况下将作为 Cache 方式访问经由二级 Cache 判断是命中，从而维护其 IO 一致性信息。而通过这些窗口的配置，可以使在这些窗口命中的访问以 Uncache 的方式直接访问内存，而不通过硬件维护其 IO 一致性信息。

9.5 配置寄存器

配置寄存器模块主要用于控制从 AXI SLAVE 端或是 HT RECEIVER 端到达的配置寄存器访问请求，进行外部中断处理，并保存了大量软件可见的用于控制系统各种工作方式的配置寄存器。

首先，用于控制 HT 控制器各种行为的配置寄存器的访问与存储都在本模块中，本模块的访问偏移地址在 AXI 端为 0xFD_FB00_0000 到 0xFD_FBFF_FFFF。本模块中所有软件可见寄存器如下表所示：

表 9-7 本模块中所有软件可见寄存器

偏移地址	名称	描述

0x30		
0x34		
0x38		
0x3c	Bridge Control	Bus Reset Control
0x40	Capability Registers	Command, Capabilities Pointer, Capability ID
0x44		Link Config, Link Control
0x48		Revision ID, Link Freq, Link Error, Link Freq Cap
0x4c		Feature Capability
0x50	自定义寄存器	MISC
0x54		
0x58		
0x5c		
0x60	接收地址窗口配置寄存器	HT 总线接收地址窗口 0 使能（外部访问）
0x64		HT 总线接收地址窗口 0 基址（外部访问）
0x68		HT 总线接收地址窗口 1 使能（外部访问）
0x6c		HT 总线接收地址窗口 1 基址（外部访问）
0x70		HT 总线接收地址窗口 2 使能（外部访问）
0x74		HT 总线接收地址窗口 2 基址（外部访问）
0x78		
0x7c		
0x80	中断向量寄存器	HT 总线中断向量寄存器[31:0]
0x84		HT 总线中断向量寄存器[63:32]
0x88		HT 总线中断向量寄存器[95:64]
0x8c		HT 总线中断向量寄存器[127:96]
0x90		HT 总线中断向量寄存器[159:128]
0x94		HT 总线中断向量寄存器[191:160]
0x98		HT 总线中断向量寄存器[223:192]
0x9C		HT 总线中断向量寄存器[255:224]
0xA0	中断使能寄存器	HT 总线中断使能寄存器[31:0]
0xA4		HT 总线中断使能寄存器[63:32]
0xA8		HT 总线中断使能寄存器[95:64]
0xAC		HT 总线中断使能寄存器[127:96]
0xB0		HT 总线中断使能寄存器[159:128]
0xB4		HT 总线中断使能寄存器[191:160]
0xB8		HT 总线中断使能寄存器[223:192]
0xBC		HT 总线中断使能寄存器[255:224]
0xC0	Interrupt Discovery & Configuration	Interrupt Capability
0xC4		DataPort
0xC8		IntrInfo[31:0]
0xCC		IntrInfo[63:32]
0xD0	POST 地址窗口配置寄存器	HT 总线 POST 地址窗口 0 使能（内部访问）
0xD4		HT 总线 POST 地址窗口 0 基址（内部访问）
0xD8		HT 总线 POST 地址窗口 1 使能（内部访问）
0xDC		HT 总线 POST 地址窗口 1 基址（内部访问）
0xE0	可预取地址窗口配置寄存器	HT 总线可预取地址窗口 0 使能（内部访问）
0xE4		HT 总线可预取地址窗口 0 基址（内部访问）
0xE8		HT 总线可预取地址窗口 1 使能（内部访问）

0xEC		Ht 总线可预取地址窗口 1 基址（内部访问）
0xF0	Uncache 地址窗口配置寄存器	HT 总线 Uncache 地址窗口 0 使能（内部访问）
0xF4		HT 总线 Uncache 地址窗口 0 基址（内部访问）
0xF8		HT 总线 Uncache 地址窗口 1 使能（内部访问）
0xFC		HT 总线 Uncache 地址窗口 1 基址（内部访问）

每个寄存存器的具体含义如下节如示：

9.5.1 Bridge Control

偏移量： 0x3C
复位值： 0x00000000
名称： Bus Reset Control

位域	位域名称	位宽	复位值	访问	描述
31:23	Reserved	4	0x0		保留
22	Reset	12	0x0	R/W	总线复位控制： 0->1：HT_RSTn 置 0，总线复位 1->0：HT_RSTn 置 1，总线解复位
21:18	Reserved	4	0x0		保留
17	B_interleave	1	0x0	R/W	写响应通道是否允许乱序执行 当使用多片互连模式时，必须置 1
16	Nop_interleave	1	0x0	R/W	流控通道是否允许乱序执行 当使用多片互连模式时，必须置 1
15:0	Reserved	16	0x0		保留

9.5.2 Capability Registers

偏移量： 0x40
复位值： 0x20010008
名称： Command，Capabilities Pointer，Capability ID

位域	位域名称	位宽	复位值	访问	描述
31:29	HOST/Sec	3	0x1	R	Command 格式为 HOST/Sec
28:27	Reserved	2	0x0	R	保留
26	Act as Slave	1	0x0 /0x1	R/W	HOST/SLAVE 模式 初始值由引脚 HOSTMODE 决定 HOSTMODE 上拉：0 HOSTMODE 下拉：1
25	Reserved	1	0x0		保留
24	Host Hide	1	0x0	R/W	是否禁止来自 HT 总线的寄存器访问
23	Reserved	1	0x0		保留
22:18	Unit ID	5	0x0	R/W	HOST 模式时：可用于记录使用 ID 个数 SLAVE 模式时：记录自身 Unit ID
17	Double Ended	1	0x0	R	不采用双 HOST 模式
16	Warm Reset	1	0x1	R	Bridge Control 中 reset 采用热复位方式
15:8	Capabilities Pointer	8	0xa0	R	下一个 Cap 寄存器偏移地址
7:0	Capability ID	8	0x08	R	HyperTransport capability ID

偏移量：　　0x44
复位值：　　0x00112000
名称：　　Link Config，Link Control

位域	位域名称	位宽	复位值	访问	描述
31	Reserved	1	0x0		保留
30:28	Link Width Out	3	0x0	R/W	发送端宽度 冷复位后的值为当前连接的最大宽度，写入此寄存器的值将会在下次热复位或是 HT Disconnect 之后生效 000：8 位方式 001：16 位方式
27	Reserved	1	0x0		保留
26:24	Link Width In	3	0x0	R/W	接收端宽度 冷复位后的值为当前连接的最大宽度，写入此寄存器的值将会在下次热复位或是 HT Disconnect 之后生效
23	Dw Fc out	1	0x0	R	发送端不支持双字流控
22:20	Max Link Width out	3	0x1	R	HT 总线发送端最大宽度：16bits
19	Dw Fc In	1	0x0	R	接收端不支持双字流控
18:16	Max Link Width In	3	0x1	R	HT 总线接收端最大宽度：16bits
15:14	Reserved	2	0x0		保留
13	LDTSTOP# Tristate Enable	1	0x1	R/W	当 HT 总线进入 HT Disconnect 状态时，是否关闭 HT PHY 1：关闭 0：不关闭
12:10	Reserved	3	0x0		保留
9	CRC Error (hi)	1	0x0	R/W	高 8 位发生 CRC 错
8	CRC Error (lo)	1	0x0	R/W	低 8 位发生 CRC 错
7	Trans off	1	0x0	R/W	HT PHY 关闭控制 处于 16 位总线工作方式时 1：关闭 高/低 8 位 HT PHY 0：使能 低 8 位 HT PHY， 高 8 位 HT PHY 由 bit 0 控制
6	End of Chain	0	0x0	R	HT 总线末端
5	Init Complete	1	0x0	R	HT 总线初始化是否完成
4	Link Fail	1	0x0	R	指示连接失败
3:2	Reserved	2	0x0		保留
1	CRC Flood Enable	1	0x0	R/W	发生 CRC 错误时，是否 flood HT 总线
0	Trans off (hi)	1	0x0	R/W	使用 16 位 HT 总线运行 8 位协议时， 高 8 位 PHY 关闭控制 1：关闭 高 8 位 HT PHY 0：使能 高 8 位 HT PHY

偏移量：　　0x48
复位值：　　0x80250023
名称：　　Revision ID，Link Freq，Link Error，Link Freq Cap

位域	位域名称	位宽	复位值	访问	描述
31:16	Link Freq Cap	16	0x0025	R	支持的 HT 总线频率 200Mhz，400Mhz，800Mhz，
15:14	Reserved	2	0x0		保留
13	Over Flow Error	1	0x0	R	HT 总线包溢出
12	Protocol Error	1	0x0	R/W	协议错误， 指 HT 总线上收到不可识别的命令
11:8	Link Freq	4	0x0	R/W	HT 总线工作频率 写入此寄存器的值后将在下次热复位或是 HT Disconnect 之后生效 0000：200M 0010：400M 0101：800M
7:0	Revision ID	8	0x23	R/W	版本号：1.03

偏移量：　　0x4C
复位值：　　0x00000002
名称：　　Feature Capability

位域	位域名称	位宽	复位值	访问	描述
31:9	Reserved	25	0x0		保留
8	Extended Register	1	0x0	R	没有
7:4	Reserved	3	0x0		保留
3	Extended CTL Time	1	0x0	R	不需要
2	CRC Test Mode	1	0x0	R	不支持
1	LDTSTOP#	1	0x1	R	支持 LDTSTOP#
0	Isochronous Mode	1	0x0	R	不支持

9.5.3 自定义寄存器

偏移量：　　0x50
复位值：　　0x20010008
名称：　　MISC

位域	位域名称	位宽	复位值	访问	描述
31	Reserved	1	0x0		保留
30	Ldt Stop Gen	1	0x0	R/W	使总线进入 LDT DISCONNECT 模式 正确的方法是：0 -> 1
29	Ldt Req Gen	1	0x0	R/W	从 LDT DISCONNECT 中唤醒 HT 总线，设置 LDT_REQ_n 正确的方法是先置 0 再置 1：0 -> 1 除此之外，直接向总线发出读写请求也可以自动唤醒总线

位域	位域名称	位宽	复位值	访问	描述
28:24	Interrupt Index	5	0x0	R/W	将除了标准中断之外的其它中断重定向到哪个中断向量中（包括 SMI，NMI，INIT，INTA，INTB，INTC，INTD） 总共 256 个中断向量，本寄存器表示的是中断向量的高 5 位，内部中断向量如下： 000：SMI 001：NMI 010：INIT 011：Reserved 100：INTA 101：INTB 110：INTC 111：INTD
23	Dword Write	1	0x1	R/W	对于 32/64/128/256 位的写访问，是否采用 Dword Write 命令格式 1：使用 Dword Write 0：使用 Byte Write（带 MASK）
22	Coherent Mode	1	0x0	R	是否是处理器一致性模式 由引脚 ICCC_EN 决定
21	Not Care Seqid	1	0x0	R/W	是否不关心 HT 序关系
20	Not Axi2Seqid	1	0x1	R	是否把 Axi 总线上的命令转换成不同的 SeqID，如果不转换，则所有的读写命令都会采用 Fixed Seqid 中的固定 ID 号 1：不转换 0：转换
19:16	Fixed Seqid	4	0x0	R/W	当 Not Axi2Seqid 有效时，配置 HT 总线发出的 Seqid
15:12	Priority Nop	4	0x4	R/W	HT 总线 Nop 流控包优先级
11:8	Priority NPC	4	0x3	R/W	Non Post 通道读写优先级
7:4	Priority RC	4	0x2	R/W	Response 通道读写先级
3:0	Priority PC	4	0x1	R/W	Post 通道读写优先级 0x0：最高优先级 0xF：最低优先级 对于各个通道的优先级均采用根据时间变化提高的优先级策略，该组存器用于配置各个通道的初始优先级

9.5.4 接收地址窗口配置寄存器

HT 控制器中的地址窗口命中公式如下：

```
hit = ( BASE & MASK ) == ( ADDR & MASK )
addr_out = TRANS_EN ? TRANS | ADDR & ~MASK : ADDR
```

需要说明的是，配置地址窗口寄存器时，MASK 高位应全为 1，低位应全为 0。MASK 中 0 的实际位数表示的就是地址窗口的大小。

接收地址窗口的地址为 HT 总线上接收的地址。落在本窗口内的 HT 地址将被发往 CPU 内，其它地址的命令将作为 P2P 命令被转发回 HT 总线。

偏移量： 0x60
复位值： 0x00000000
名称： HT 总线接收地址窗口 0 使能（外部访问）

位域	位域名称	位宽	复位值	访问	描述
31	ht_rx_image0_en	1	0x0	R/W	HT 总线接收地址窗口 0，使能信号
30	ht_rx_image0_ trans_en	1	0x0	R/W	HT 总线接收地址窗口 0，映射使能信号
29:23	Reserved	14	0x0		保留
15:0	ht_rx_image0_ trans[39:24]	16	0x0	R/W	HT 总线接收地址窗口 0，映射后地址的[39:24]

偏移量： 0x64
复位值： 0x00000000
名称： HT 总线接收地址窗口 0 基址（外部访问）

位域	位域名称	位宽	复位值	访问	描述
31:16	ht_rx_image0_ base[39:24]	16	0x0	R/W	HT 总线接收地址窗口 0，地址基址的[39:24]
15:0	ht_rx_image0_ mask[39:24]	16	0x0	R/W	HT 总线接收地址窗口 0，地址屏蔽的[39:24]

偏移量： 0x68
复位值： 0x00000000
名称： HT 总线接收地址窗口 1 使能（外部访问）

位域	位域名称	位宽	复位值	访问	描述
31	ht_rx_image1_en	1	0x0	R/W	HT 总线接收地址窗口 1，使能信号
30	ht_rx_image1_ trans_en	1	0x0	R/W	HT 总线接收地址窗口 1，映射使能信号
29:23	Reserved	14	0x0		保留
15:0	ht_rx_image1_ trans[39:24]	16	0x0	R/W	HT 总线接收地址窗口 1，映射后地址的[39:24]

偏移量： 0x6c
复位值： 0x00000000
名称： HT 总线接收地址窗口 1 基址（外部访问）

位域	位域名称	位宽	复位值	访问	描述
31:16	ht_rx_image1_ base[39:24]	16	0x0	R/W	HT 总线接收地址窗口 1，地址基址的[39:24]
15:0	ht_rx_image1_ mask[39:24]	16	0x0	R/W	HT 总线接收地址窗口 1，地址屏蔽的[39:24]

偏移量： 0x70
复位值： 0x00000000
名称： HT 总线接收地址窗口 2 使能（外部访问）

位域	位域名称	位宽	复位值	访问	描述
31	ht_rx_image2_en	1	0x0	R/W	HT 总线接收地址窗口 2，使能信号

位域	位域名称	位宽	复位值	访问	描述
30	ht_rx_image2_trans_en	1	0x0	R/W	HT 总线接收地址窗口 2，映射使能信号
29:23	Reserved	14	0x0		保留
15:0	ht_rx_image2_trans[39:24]	16	0x0	R/W	HT 总线接收地址窗口 2，转译后地址的[39:24]

偏移量： 0x74
复位值： 0x00000000
名称： HT 总线接收地址窗口 2 基址（外部访问）

位域	位域名称	位宽	复位值	访问	描述
31:16	ht_rx_image2_base[39:24]	16	0x0	R/W	HT 总线接收地址窗口 2，地址基址的[39:24]
15:0	ht_rx_image2_mask[39:24]	16	0x0	R/W	HT 总线接收地址窗口 2，地址屏蔽的[39:24]

9.5.5 中断向量寄存器

中断向量寄存器共 256 个，其中除去 HT 总线上的 Fix、Arbiter 以及 PIC 中断直接映射到此 256 个中断向量之中，其它的中断，如 SMI，NMI，INIT，INTA，INTB，INTC，INTD 可以通过寄存器 0x50 的[28:24]映射到任意一个 8 位中断向量上去，映射的顺序为{INTD,INTC,INTB，INTA，1’b0，INIT，NMI，SMI}。此时中断向量对应值为{Interrupt Index,内部向量[2:0]}。

偏移量： 0x80
复位值： 0x00000000
名称： HT 总线中断向量寄存器[31:0]

位域	位域名称	位宽	复位值	访问	描述
31:0	Interrupt_case[31:0]	32	0x0	R/W	HT 总线中断向量寄存器[31:0]， 对应中断线 0 /HT HI 对应中断线 4

偏移量： 0x84
复位值： 0x00000000
名称： HT 总线中断向量寄存器[63:32]

位域	位域名称	位宽	复位值	访问	描述
31:0	Interrupt_case[63:32]	32	0x0	R/W	HT 总线中断向量寄存器[63:32]， 对应中断线 0 /HT HI 对应中断线 4

偏移量： 0x88
复位值： 0x00000000
名称： HT 总线中断向量寄存器[95:64]

位域	位域名称	位宽	复位值	访问	描述

位域	位域名称	位宽	复位值	访问	描述
31:0	Interrupt_case [95:64]	32	0x0	R/W	HT 总线中断向量寄存器[95:64]， 对应中断线 1 /HT HI 对应中断线 5

偏移量： 0x8c
复位值： 0x00000000
名称： HT 总线中断向量寄存器[127:96]

位域	位域名称	位宽	复位值	访问	描述
31:0	Interrupt_case [127:96]	32	0x0	R/W	HT 总线中断向量寄存器[127:96]， 对应中断线 1 /HT HI 对应中断线 5

偏移量： 0x90
复位值： 0x00000000
名称： HT 总线中断向量寄存器[159:128]

位域	位域名称	位宽	复位值	访问	描述
31:0	Interrupt_case [159:128]	32	0x0	R/W	HT 总线中断向量寄存器[159:128]， 对应中断线 2 /HT HI 对应中断线 6

偏移量： 0x94
复位值： 0x00000000
名称： HT 总线中断向量寄存器[191:160]

位域	位域名称	位宽	复位值	访问	描述
31:0	Interrupt_case [191:160]	32	0x0	R/W	HT 总线中断向量寄存器[191:160]， 对应中断线 2 /HT HI 对应中断线 6

偏移量： 0x98
复位值： 0x00000000
名称： HT 总线中断向量寄存器[223:192]

位域	位域名称	位宽	复位值	访问	描述
31:0	Interrupt_case [223:192]	32	0x0	R/W	HT 总线中断向量寄存器[223:192]， 对应中断线 3 /HT HI 对应中断线 7

偏移量： 0x9c
复位值： 0x00000000
名称： HT 总线中断向量寄存器[255:224]

位域	位域名称	位宽	复位值	访问	描述
31:0	Interrupt_case [255:224]	32	0x0	R/W	HT 总线中断向量寄存器[255:224]， 对应中断线 3 /HT HI 对应中断线 7

9.5.6 中断使能寄存器

中断使能寄存器共 256 个，与中断向量寄存器一一对应。 置 1 为对应中断打开，置 0 则为中断屏蔽。

偏移量： 0xa0

复位值： 0x00000000
名称： HT 总线中断使能寄存器[31:0]

位域	位域名称	位宽	复位值	访问	描述
31:0	Interrupt_mask [31:0]	32	0x0	R/W	HT 总线中断使能寄存器[31:0]， 对应中断线 0 /HT HI 对应中断线 4

偏移量： 0xa4
复位值： 0x00000000
名称： HT 总线中断使能寄存器[63:32]

位域	位域名称	位宽	复位值	访问	描述
31:0	Interrupt_mask [63:32]	32	0x0	R/W	HT 总线中断使能寄存器[63:32]， 对应中断线 0 /HT HI 对应中断线 4

偏移量： 0xa8
复位值： 0x00000000
名称： HT 总线中断使能寄存器[95:64]

位域	位域名称	位宽	复位值	访问	描述
31:0	Interrupt_mask [95:64]	32	0x0	R/W	HT 总线中断使能寄存器[95:64]， 对应中断线 1 /HT HI 对应中断线 5

偏移量： 0xac
复位值： 0x00000000
名称： HT 总线中断使能寄存器[127:96]

位域	位域名称	位宽	复位值	访问	描述
31:0	Interrupt_mask [127:96]	32	0x0	R/W	HT 总线中断使能寄存器[127:96]， 对应中断线 1 /HT HI 对应中断线 5

偏移量： 0xb0
复位值： 0x00000000
名称： HT 总线中断使能寄存器[159:128]

位域	位域名称	位宽	复位值	访问	描述
31:0	Interrupt_mask [159:128]	32	0x0	R/W	HT 总线中断使能寄存器[159:128]， 对应中断线 2 /HT HI 对应中断线 6

偏移量： 0xb4
复位值： 0x00000000
名称： HT 总线中断使能寄存器[191:160]

位域	位域名称	位宽	复位值	访问	描述
31:0	Interrupt_mask [191:160]	32	0x0	R/W	HT 总线中断使能寄存器[191:160]， 对应中断线 2 /HT HI 对应中断线 6

偏移量： 0xb8
复位值： 0x00000000
名称： HT 总线中断使能寄存器[223:192]

位域	位域名称	位宽	复位值	访问	描述
31:0	Interrupt_mask [223:192]	32	0x0	R/W	HT 总线中断使能寄存器[223:192]， 对应中断线 3 /HT HI 对应中断线 7

偏移量：　　0xbc
复位值：　　0x00000000
名称：　　HT 总线中断使能寄存器[255:224]

位域	位域名称	位宽	复位值	访问	描述
31:0	Interrupt_mask [255:224]	32	0x0	R/W	HT 总线中断使能寄存器[255:224]， 对应中断线 3 /HT HI 对应中断线 7

9.5.7 Interrupt Discovery & Configuration

偏移量：　　0xc0
复位值：　　0x80000008
名称：　　Interrupt Capability

位域	位域名称	位宽	复位值	访问	描述
31:24	Capabilities Pointer	8	0x80	R	Interrupt discovery and configuration block
23:16	Index	8	0x0	R/W	读寄存器偏移地址
15:8	Capabilities Pointer	8	0x0	R	Capabilities Pointer
7:0	Capability ID	8	0x08	R	Hypertransport Capablity ID

偏移量：　　0xc4
复位值：　　0x00000000
名称：　　Dataport

位域	位域名称	位宽	复位值	访问	描述
31:0	Dataport	32	0x0	R/W	当上一寄存器 Index 为 0x10 时，本寄存器读写结果为 0xa8 寄存器，否则为 0xac

偏移量：　　0xc8
复位值：　　0xF8000000
名称：　　IntrInfo[31:0]

位域	位域名称	位宽	复位值	访问	描述
31:24	IntrInfo[31:24]	32	0xF8	R	保留
23:2	IntrInfo[23:2]	22	0x0	R/W	IntrInfo[23:2]，当发出 PIC 中断时，IntrInfo 的值用来表示中断向量
1:0	Reserved	2	0x0	R	保留

偏移量：　　0xcc
复位值：　　0x00000000
名称：　　IntrInfo[63:32]

位域	位域名称	位宽	复位值	访问	描述
31:0	IntrInfo[63:32]	32	0x0	R	保留

9.5.8 POST地址窗口配置寄存器

地址窗口命中公式详见 9.5.4 节。

本窗口的地址是 AXI 总线上接收到的地址。落在本窗口的所有写访问将立即在 AXI B 通道返回，并以 POST WRITE 的命令格式发给 HT 总线。而不在本窗口的写请求则以 NONPOST WRITE 的方式发送到 HT 总线，并等待 HT 总线响应后再返回 AXI 总线。

偏移量： 0xd0
复位值： 0x00000000
名称： HT 总线 POST 地址窗口 0 使能（内部访问）

位域	位域名称	位宽	复位值	访问	描述
31	ht_post0_en	1	0x0	R/W	HT 总线 POST 地址窗口 0，使能信号
30:23	Reserved	15	0x0		保留
15:0	ht_post0_trans [39:24]	16	0x0	R/W	HT 总线 POST 地址窗口 0,转译后地址的[39:24]

偏移量： 0xd4
复位值： 0x00000000
名称： HT 总线 POST 地址窗口 0 基址（内部访问）

位域	位域名称	位宽	复位值	访问	描述
31:16	ht_post0_base [39:24]	16	0x0	R/W	HT 总线 POST 地址窗口 0，地址基址的[39:24]
15:0	ht_post0_mask [39:24]	16	0x0	R/W	HT 总线 POST 地址窗口 0，地址屏蔽的[39:24]

偏移量： 0xd8
复位值： 0x00000000
名称： HT 总线 POST 地址窗口 1 使能（内部访问）

位域	位域名称	位宽	复位值	访问	描述
31	ht_post1_en	1	0x0	R/W	HT 总线 POST 地址窗口 1，使能信号
30:23	Reserved	15	0x0		保留
15:0	ht_post1_trans [39:24]	16	0x0	R/W	HT 总线 POST 地址窗口 1,转译后地址的[39:24]

偏移量： 0xdc
复位值： 0x00000000
名称： HT 总线 POST 地址窗口 1 基址（内部访问）

位域	位域名称	位宽	复位值	访问	描述
31:16	ht_post1_base [39:24]	16	0x0	R/W	HT 总线 POST 地址窗口 1，地址基址的[39:24]
15:0	ht_post1_mask [39:24]	16	0x0	R/W	HT 总线 POST 地址窗口 1，地址屏蔽的[39:24]

9.5.9 可预取地址窗口配置寄存器

地址窗口命中公式详见 9.5.4 节。

本窗口的地址是 AXI 总线上接收到的地址。落在本窗口的取指指令以及 CACHE 访问才会被发往 HT 总线，其它的取指或 CACHE 访问将不会被发往 HT 总线，而是立即返回，如果是读命令，则会返回相应个数的无效读数据。

偏移量： 0xe0
复位值： 0x00000000
名称： HT 总线可预取地址窗口 0 使能（内部访问）

位域	位域名称	位宽	复位值	访问	描述
31	ht_prefetch0_en	1	0x0	R/W	HT 总线可预取地址窗口 0，使能信号
30:23	Reserved	15	0x0		保留
15:0	ht_prefetch0_trans [39:24]	16	0x0	R/W	HT 总线可预取地址窗口 0，转译后地址的[39:24]

偏移量： 0xe4
复位值： 0x00000000
名称： HT 总线可预取地址窗口 0 基址（内部访问）

位域	位域名称	位宽	复位值	访问	描述
31:16	ht_prefetch0_base[39:24]	16	0x0	R/W	HT 总线可预取地址窗口 0，地址基址的[39:24]位地址
15:0	ht_prefetch0_mask[39:24]	16	0x0	R/W	HT 总线可预取地址窗口 0，地址屏蔽的[39:24]

偏移量： 0xe8
复位值： 0x00000000
名称： HT 总线可预取地址窗口 1 使能（内部访问）

位域	位域名称	位宽	复位值	访问	描述
31	ht_prefetch1_en	1	0x0	R/W	HT 总线可预取地址窗口 1，使能信号
30:23	Reserved	15	0x0		保留
15:0	ht_prefetch1_trans[39:24]	16	0x0	R/W	HT 总线可预取地址窗口 1，转译后地址的[39:24]

偏移量： 0xec
复位值： 0x00000000
名称： HT 总线可预取地址窗口 1 基址（内部访问）

位域	位域名称	位宽	复位值	访问	描述
31:16	ht_prefetch1_base[39:24]	16	0x0	R/W	HT 总线可预取地址窗口 1，地址基址的[39:24]
15:0	ht_prefetch1_mask[39:24]	16	0x0	R/W	HT 总线可预取地址窗口 1，地址屏蔽的[39:24]

9.5.10 UNCACHE地址窗口配置寄存器

地址窗口命中公式详见 9.5.4 节。

本窗口的地址是 HT 总线上接收到的地址。落在本窗口地址的读写命令，将不会被送往二级 CACHE，也不会使一级 CACHE 发生失效，而是会被直接送至内存或是其它的地址空间，也即该地址窗口中的读写命令将不会维持 IO 的 CACHE 一致性。该窗口主要针对一些不会在 CACHE 中命中所以可以提高存问效率的操作，如显存的访问等。

偏移量： 0xf0
复位值： 0x00000000
名称： HT 总线 Uncache 地址窗口 0 使能（内部访问）

位域	位域名称	位宽	复位值	访问	描述
31	ht_uncache0_en	1	0x0	R/W	HT 总线 uncache 地址窗口 0，使能信号
30	ht_uncache0_trans_en	1	0x0	R/W	HT 总线 uncache 地址窗口 1，映射使能信号
29:23	Reserved	14	0x0		保留
15:0	ht_uncache0_trans[39:24]	16	0x0	R/W	HT 总线 uncache 地址窗口 0，转译后地址的[39:24]

偏移量： 0xf4
复位值： 0x00000000
名称： HT 总线 Uncache 地址窗口 0 基址（内部访问）

位域	位域名称	位宽	复位值	访问	描述
31:16	ht_uncache0_base[39:24]	16	0x0	R/W	HT 总线 uncache 地址窗口 0,地址基址的[39:24]
15:0	ht_uncache0_mask[39:24]	16	0x0	R/W	HT 总线 uncache 地址窗口 0,地址屏蔽的[39:24]

偏移量： 0xf8
复位值： 0x00000000
名称： HT 总线 Uncache 地址窗口 1 使能（内部访问）

位域	位域名称	位宽	复位值	访问	描述
31	ht_uncache1_en	1	0x0	R/W	HT 总线 uncache 地址窗口 1，使能信号
30	ht_uncache1_trans_en	1	0x0	R/W	HT 总线 uncache 地址窗口 1，映射使能信号
29:23	Reserved	14	0x0		保留
15:0	ht_uncache1_trans[39:24]	16	0x0	R/W	HT 总线 uncache 地址窗口 1，转译后地址的[39:24]

偏移量： 0xfc
复位值： 0x00000000
名称： HT 总线 Uncache 地址窗口 1 基址（内部访问）

位域	位域名称	位宽	复位值	访问	描述
31:16	ht_uncache1_base[39:24]	16	0x0	R/W	HT 总线 uncache 地址窗口 1,地址基址的[39:24]

位域	位域名称	位宽	复位值	访问	描述
15:0	ht_uncache1_mask[39:24]	16	0x0	R/W	HT 总线 uncache 地址窗口 1，地址屏蔽的[39:24]

9.5.11 HyperTransport总线配置空间的访问方法

HyperTransport 接口软件层的协议与 PCI 协议基本一致，由于配置空间的访问直接与底层协议相关，具体访问细节略有不同。在表 9-5 中已列出，HT 总线配置空间的地址范围是 0xFD_FE00_0000 ～0xFD_FFFF_FFFF。对于 HT 协议中的配置访问，在龙芯 3B1000 中采用如下格式实现。

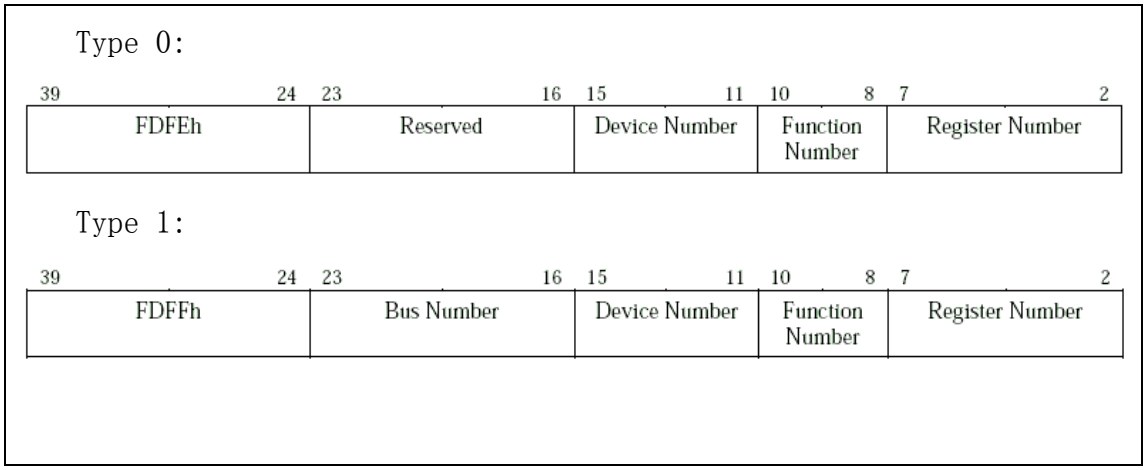


图 9-1 龙芯 3B1000 中 HT 协议的配置访问

9.6 HyperTransport 多处理器支持

龙芯 3B1000 处理器使用 HyperTransport 接口进行多处理器互联，并且硬件可以自动维护 2 个芯片之间的一致性请求。下面介绍两片龙芯 3B1000 处理器的互联方法。

两片龙芯 3 号互联结构

根据固定路由算法的特性，我们在构建两片处理器互联时，有两种不同的方法。首先是采用 8 位 HT 总线互联。在这种互联方式下，两个处理器之间只能采用 8 位 HT 互联。如下所示：

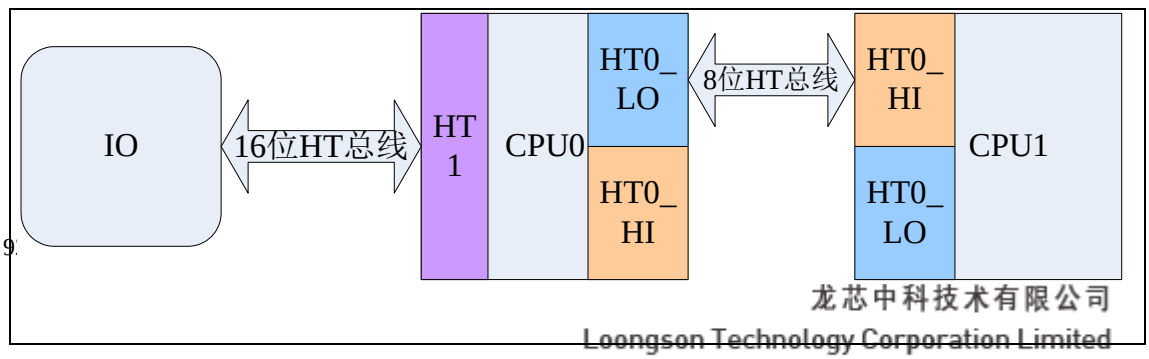


图 9-2 两片龙芯 3B1000 8 位互联结构

此外，龙芯 3B1000 的 HT 总线最高可以采用 16 位宽度，因此还可以采用最大化带宽的 16 位互联方式。将龙芯 3B1000 中的 HT0 控制器设置为 16 位模式后，所有发到 HT0 控制器的命令都会被发往 HT0_LO，而不会像第一种互联方式一样按照路由表分别发至 HT0_HI 或是 HT0_LO。所以只需正确配置 CPU0 与 CPU1 的 16 位模式，并正确连接高低位总线，即可使用 16 位的 HT 总线互联方式。同时，该互联结构也支持使用 8 位的 HT 总线协议进行相互访问。该互联结构如下图所示：

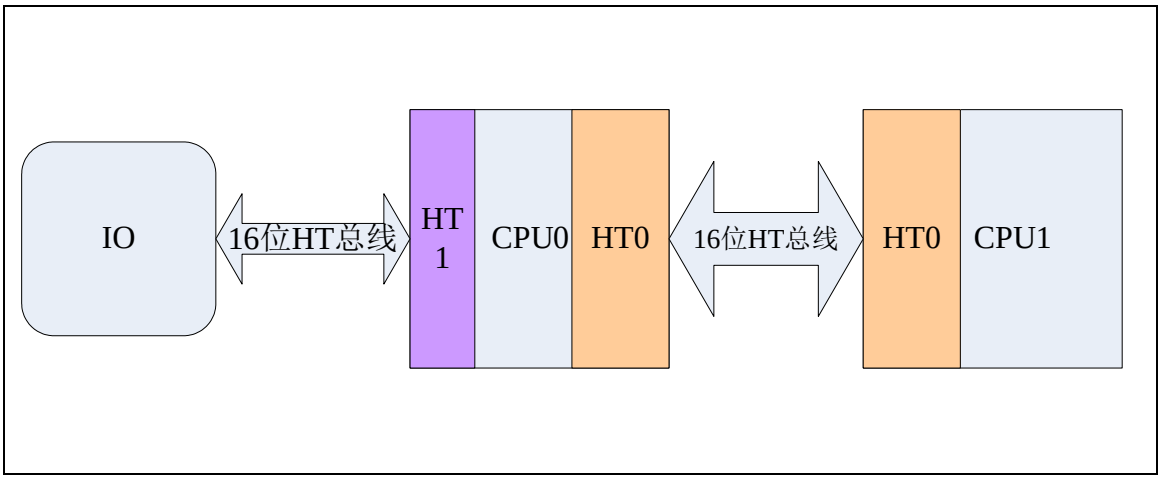


图 9-3 两片龙芯 3B1000 16 位互联结构

10 低速 IO 控制器配置

龙芯 3B1000 I/O 控制器包括 PCI/PCI-X 控制器、LPC 控制器、UART 控制器、SPI 控制器、GPIO 以及配置寄存器。这些 I/O 控制器共享一个交叉开关端口，CPU 的请求经过地址译码后发送到相应的设备。

10.1 PCI/PCI-X 控制器

龙芯 3B1000 的 PCI/PCI-X 控制器既可以作为主桥控制整个系统，也可以作为普通 PCI/PCI-X 设备工作在 PCI/PCI-X 总线上。它的实现符合 PCI-X 1.0b 和 PCI 2.3 规范。龙芯 3B1000 的 PCI/PCI-X 控制器还内置了 PCI 仲裁器。

PCI/PCI-X 控制器的配置头位于 0x1FE00000 开始的 256 字节，如表 10-1 所示。

表 10-1 PCIX 控制器配置头

字节 3	字节 2	字节 1	字节 0	地址
Device ID		Vendor ID		00
Status		Command		04
Class Code			Revision ID	08
BIST	Header Type	Latency Timer	CacheLine Size	0C
Base Address Register 0				10
Base Address Register 1				14
Base Address Register 2				18
Base Address Register 3				1C
Base Address Register 4				20
Base Address Register 5				24
				28
Subsystem ID		Subsystem Vendor ID		2C
				30
			Capabilities Pointer	34
				38
Maximum Latency	Minimum Grant	Interrupt Pin	Interrupt Line	3C
Implementation Specific Register(ISR40)				40
Implementation Specific Register(ISR44)				44
Implementation Specific Register(ISR48)				48
Implementation Specific Register(ISR4C)				4C
Implementation Specific Register(ISR50)				50
Implementation Specific Register(ISR54)				54
Implementation Specific Register(ISR58)				58
				...

PCIX Command Register	E0
PCIX Status Register	E4

龙芯 3B1000 的 PCIX 控制器支持三个 64 位窗口，由{BAR1, BAR0}、{BAR3, BAR2}、{BAR5, BAR4}三对寄存器配置窗口 0、1、2 的基址。窗口的大小、使能以及其它细节由另外三个对应寄存器 PCI_Hit0_Sel, PCI_Hit1_Sel, PCI_Hit2_Sel 控制，具体位域请参见表 10-2。

表 10-2 PCI 控制寄存器

位域	字段名	访问	复位值	说明
REG_40				
31	tar_read_io	读写 (写 1 清)	0	target 端收到对 IO 或者是不可预取区域的访问
30	tar_read_discard	读写 (写 1 清)	0	target 端的 delay 请求被丢弃
29	tar_resp_delay	读写	0	target 访问何时给出 delay/split 0: 超时后 1: 马上
28	tar_delay_retry	读写	0	target 访问重试策略 0: 根据内部逻辑（见 29 位） 1: 马上重试
27	tar_read_abort_en	读写	0	若 target 对内部的读请求超时，是否允许 target-abort 回应
26:25	Reserved	-	0	
24	tar_write_abort_en	读写	0	若 target 对内部的写请求超时，是否允许 target-abort 回应
23	tar_master_abort	读写	0	是否允许 master-abort
22:20	tar_subseq_timeout	读写	000	target 后续延迟超时 000: 8 周期 其它: 不支持
19:16	tar_init_timeout	读写	0000	target 初始延迟超时 PCI 模式下 0: 16 周期 1-7: 禁用计数器 8-15: 8-15 周期 PCIX 模式下超时计数固定为 8 周期，此处配置影响最大的 delay 访问数 0: 8 delay 访问 8: 1 delay 访问 9: 2 delay 访问 10: 3 delay 访问 11: 4 delay 访问 12: 5 delay 访问 13: 6 delay 访问 14: 7 delay 访问 15: 8 delay 访问

15:4	tar_pref_boundary	读写	000h	可预取边界配置（以 16 字节为单位） FFF: 64KB 到 16byte FFE: 64KB 到 32byte FF8: 64KB 到 128byte
3	tar_pref_bound_en	读写	0	使用 tar_pref_boundary 的配置 0: 预取到设备边界 1: 使用 tar_pref_boundary
2	Reserved	-	0	
1	tar_splitw_ctrl	读写	0	target split 写控制 0: 阻挡除 Posted Memory Write 以外的访问 1: 阻挡所有访问，直至 split 完成
0	mas_lat_timeout	读写	0	禁用 mater 访问超时 0: 允许 master 访问超时 1: 不允许
REG_44				
31:0	Reserved	-	-	
REG_48				
31:0	tar_pending_seq	读写	0	target 未处理完的请求号位向量 对应位写 1 可清
REG_4C				
31:30	Reserved	-	-	
29	mas_write_defer	读写	0	允许后续的读越过前面未完成的写 (只对 PCI 有效)
28	mas_read_defer	读写	0	允许后续的读写越过前面未完成的读 (只对 PCI 有效)
27	mas_io_defer_cnt	读写	0	在外的最大 IO 请求数 0: 由控制 1: 1
26:24	mas_read_defer_cnt	读写	010	master 支持在外读的最大数(只对 PCI 有效） 0: 8 1-7: 1-7 注：一个双地址周期访问占两项
23:16	err_seq_id	只读	00h	target/master 错误号
15	err_type	只读	0	target/master 出错的命令类型 0:
14	err_module	只读	0	出错的模块 0: target 1: master
13	system_error	读写	0	target/master 系统错（写 1 清）
12	data_parity_error	读写	0	target/master 数据奇偶错（写 1 清）
11	ctrl_parity_error	读写	0	target/master 地址奇偶错（写 1 清）
10:0	Reserved	-	-	

REG_50				
31:0	mas_pending_seq	读写	0	master 未处理完的请求号位向量 对应位写 1 可清
REG_54				
31:0	mas_split_err	读写	0	split 返回出错的请求号位向量
REG_58				
31:30	Reserved	-	-	
29:28	tar_split_priority	读写	0	target split 返回优先级 0 最高, 3 最低
27:26	mas_req_priority	读写	0	master 对外的优先级 0 最高, 3 最低
25	Priority_en	读写	0	仲裁算法（在 master 的访问和 target 的 split 返回间做仲裁） 0: 固定优先级 1: 轮转
24:18	保留	-	-	
17	mas_retry_aborted	读写	0	master 重试取消（写 1 清）
16	mas_trdy_timeout	读写	0	master TRDY 超时计数
15:8	mas_retry_value	读写	00h	master 重试次数 0: 无限重试 1-255: 1-255 次
7:0	mas_trdy_count	读写	00h	master TRDY 超时计数器 0: 禁用 1-255: 1-255 拍

在发起配置空间读写前，应用程序应先配置好 PCIMap_Cfg 寄存器，告诉控制器欲发起的配置操作的类型和高 16 位地址线上的值。然后对 0x1fe80000 开始的 2K 空间进行读写即可访问对应设备的配置头。设备号根据 PCIMap_Cfg[15:0]从低到高优先编码得到。

配置操作地址生成见图 10-1。

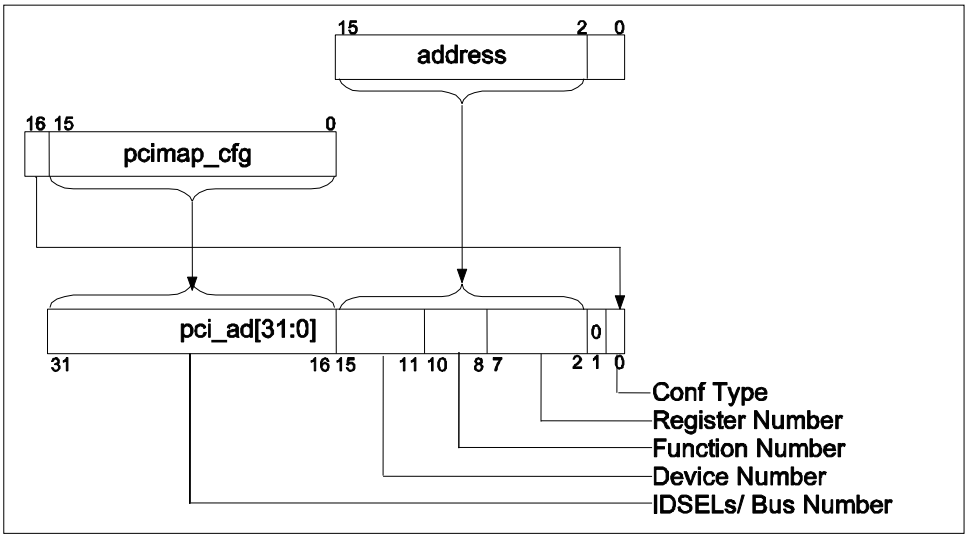


图 10-1 配置读写总线地址生成

PCI/PCIX 仲裁器实现了两级轮转仲裁、总线停靠和损坏主设备的隔离。其配置和状态见 PXArb_Config 和 PXArb_Status 寄存器。PCI/PCIX 总线请求与应答线分配见表 10-3。

表 10-3 PCI/PCIX 总线请求与应答线分配

请求与应答线	描述
0	内部集成 PCI/PCIX 控制器
7:1	外部请求 6~0

基于轮转的仲裁算法提供两个级别，第二级整体作为第一级中的一员一起调度。当多个设备同时申请总线时每轮转完一次第一级设备，第二级中优先级最高的设备可以得到总线。

仲裁器被设计成任何时候只要条件允许就可以切换，对于一些不符合协议的 PCI 设备，这样做可能会使之不正常。使用强制优先级可以让这些设备通过持续请求来占有总线。

总线停靠是指当没有设备请求使用总线时是否选择其中一个给出允许信号。对于已经得到允许的设备而言，直接发起总线操作能够提高效率。龙芯 2F 的 PCI 仲裁器提供两种停靠模式：最后一个主设备和默认主设备。如果在特殊场合下不能够停靠，可以将仲裁器设置为停靠到默认 0 号主设备（内部控制器），且依靠延迟为 0。

10.2 LPC 控制器

LPC 控制器具有以下特性：

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

LPC 控制器的地址空间分布见表 10-4：

表 10-4 LPC 控制器地址空间分布

地址名称	地址范围	大小
LPC Boot	0X1FC0_0000-0X1FD0_0000	1MByte
LPC Memory	0X1C00_0000-0X1E00_0000	32MByte
LPC I/O	0X1FF0_0000-0X1FF1_0000	64KByte
LPC Register	0X1FE0_0200-0X1FE0_0300	256Byte

LPC Boot 地址空间是系统启动时处理器最先访问的地址空间。这个地址空间支持 LPC Memory 或 Firmware Memory 访问类型。系统启动时发出哪种访问类型由 LPC_ROM_INTEL 引脚控制。LPC_ROM_INTEL 引脚上拉时发出 LPC Firmware Memory 访问，LPC_ROM_INTEL 引脚下拉时发出 LPC Memory 访问类型。

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

处理器发往 LPC I/O 地址空间的访问按照 LPC I/O 访问类型发往 LPC 总线。地址为地址空间低 16 位。

LPC 控制器配置寄存器共有 3 个 32 位寄存器，基地址为 0x1FE0_0200。配置寄存器的含义见表 10-5:

表 10-5 LPC 配置寄存器含义

位域	字段名	访问	复位值	说明
REG0				
REG0[31:31]	SIRQ_EN	读写	0	SIRQ 使能控制
REG0[23:23]	LPC_MEM_TRANS_EN	读写	0	LPC Memory 空间地址转换控制使能控制
REG0[22:16]	LPC_MEM_TRANS	读写	0	LPC Memory 空间地址转换控制，对应于 LPC 总线地址的[31:25]
REG0[15:0]	LPC_SYNC_TIMEOUT	读写	0	LPC 访问超时计数器
REG1				
REG1[31:31]	LPC_MEM_IS_FWH	读写	0	LPC Memory 空间 Firmware Memory 访问类型设置
REG1[17:0]	LPC_INT_EN	读写	0	LPC SIRQ 中断使能
REG2				
REG2[17:0]	LPC_INT_SRC	读写	0	LPC SIRQ 中断源指示
REG3				
REG3[17:0]	LPC_INT_CLEAR	写	0	LPC SIRQ 中断清除

10.3 UART 控制器

UART 控制器具有以下特性

- 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
- 仅工作在 FIFO 方式

- 在寄存器与功能上兼容 NS16550A

本模块有两个并行工作 UART 接口，功能寄存器完全一样，只是访问基址不同。

UART0 寄存器物理地址基址为 0x1FE001E0。

UART1 寄存器物理地址基址为 0x1FE001E8。

10.3.1 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

10.3.2 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

10.3.3 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	中断源标志位，详见下表
0	INTp	1	R	中断标志位

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

10.3.4 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

10.3.5 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’ – 读写分频锁存器 ‘0’ – 读写正常寄存器
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

10.3.6 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

10.3.7 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	传输为空标志位 ‘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 为空标志位 ‘1’ – 当前传输 FIFO 为空，给传输 FIFO 写数据时清零 ‘0’ – 有数据
4	BI	1	R	打断中断标志位 ‘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	帧错误标志位 ‘1’ – 接收的数据没有停止位 ‘0’ – 没有错误
2	PE	1	R	奇偶校验位错误标志位 ‘1’ – 当前接收数据有奇偶错误 ‘0’ – 没有奇偶错误
1	OE	1	R	数据溢出标志位

				'1' – 有数据溢出 '0' – 无溢出
0	DR	1	R	接收数据有效标志位 '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。

10.3.8 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

10.3.9 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

10.4 SPI Flash 控制器

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

SPI 控制器具有以下特性：

- 全双工同步串口数据传输
- 支持到 4 个的变长字节传输
- 主模式支持
- 极性和相位可编程的串行时钟
- 支持系统启动
- 支持标准读、连续地址读、快速读、双路 I/O 等 SPI Flash 读模式

龙芯 3A 中 SPI 控制器模块寄存器物理地址基址为 0x1FE001F0。

龙芯 3B1000 中 SPI 控制器模块寄存器物理地址基址为 0x1FE00220。

10.4.1 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 高电平有效
6	spe	1	RW	系统工作使能信号高电平有效

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，为 1 位则相位相反，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

10.4.2 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，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，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，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

10.4.3 数据寄存器（TxFIFO）

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

10.4.4 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请求信号 00 -1 字节 01 - 2 字节 10 -3 字节 11 - 3 字节
5:3	Reserved	4	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 spr	00 00	00 01	00 10	00 11	01 00	01 01	01 10	01 11	10 00	10 01	10 10	10 11
分频系数	2	4	16	32	8	64	128	256	512	1024	2048	4096

10.4.5 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。

10.4.6 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

10.4.7 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

10.4.8 SPI Flash控制器结构

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。

SPI Flash 控制器是在一个简单 SPI 控制器的基础上增加专门的硬件逻辑，使得控制器对外(软件)除了有若干 IO 寄存器外还有一段只读 memory 空间。这段 memory 空间映射到 SPI Flash 中，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。

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)。

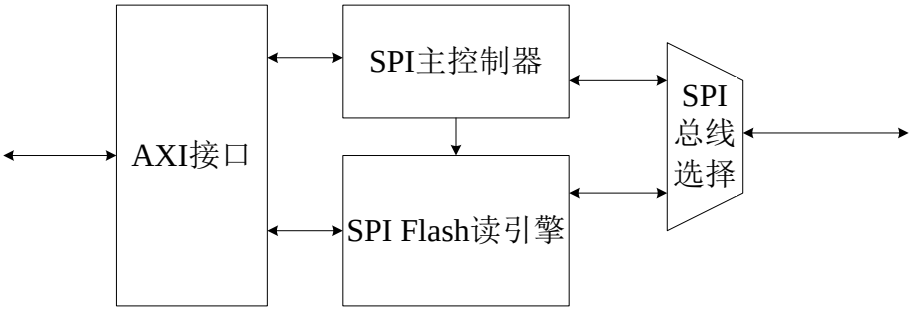


图 10-2 SPI Flash 控制器结构

SPI 主控制器只接收单字节的读写访问，实现了 SPI 主设备的功能，并为 SPI Flash 读引擎提供参数配置。SPI 主控制器的结构如图 10-3 所示。系统寄存器包括控制寄存器，状态寄存器、外部寄存器和 SPI Flash 相关控制寄存器。分频器生成 SPI 总线工作的时钟信号，数据读、写缓冲器（FIFO）允许 SPI 同时进行串行发送和接收数据。

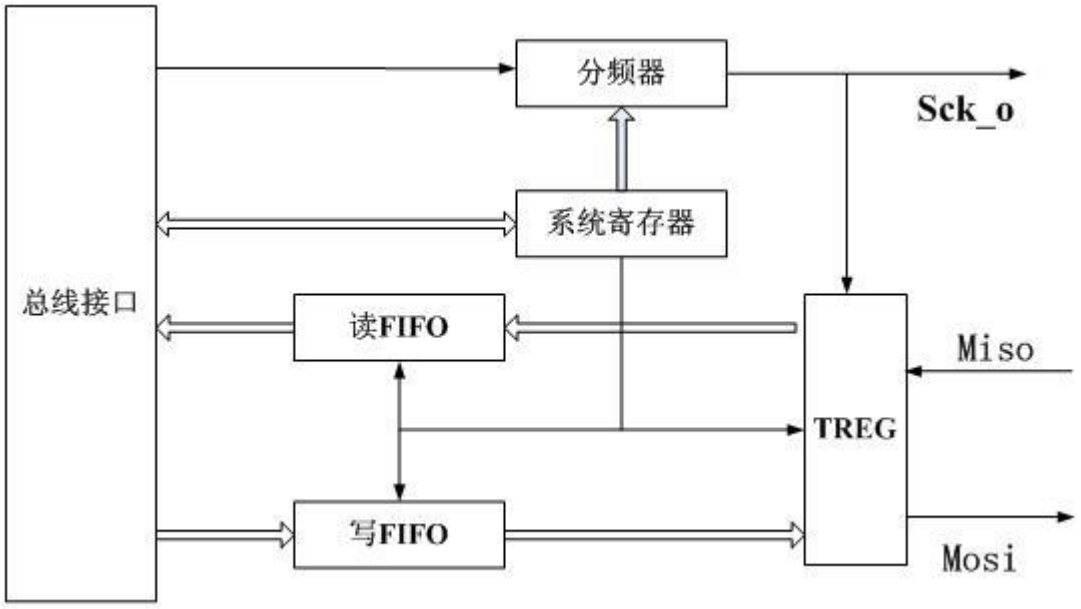


图 10-3 SPI 主控制器结构

SPI Flash 读引擎结构如图 10-3 所示。它将读请求打包成 SPI Flash 的读命令，读出数据后返回。为了加快接口速度，SPI Flash 生产厂商在最早的 SPI Flash 协议基础上作了若干扩展。根据受支持的情况和实现开销，本引擎实现了三种增强模式，分别为

- 连续地址读：在传送完一个字节数据后 SPI Flash 自动准备好下一个地址的数据，

只要片选不拉高就可以继续传输。

- 快速读模式：可工作在高频率的读模式，在命令地址后还附了一个字节的空数据供数据读出。
- 双 I/O 模式：在发完命令编码后，SDI 和 SDO 两根线不再遵循 SPI 协议，而是同时同向进行数据传输。

双 I/O 模式与快速读模式互斥，二者只能选其一。连续地址读则与其它两种模式可共存。

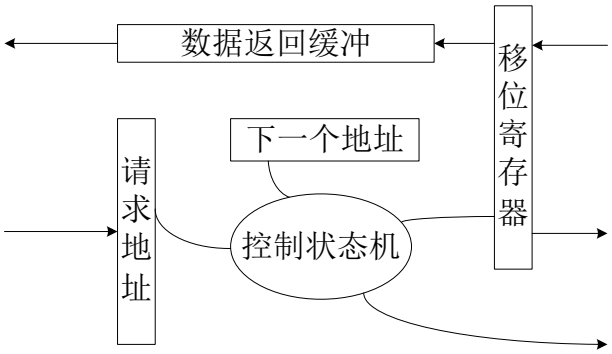


图 10-4 SPI Flash 读引擎结构

10.4.9 SPI主控制器外部接口时序图

如图 10-5 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(sper.mode=1)。

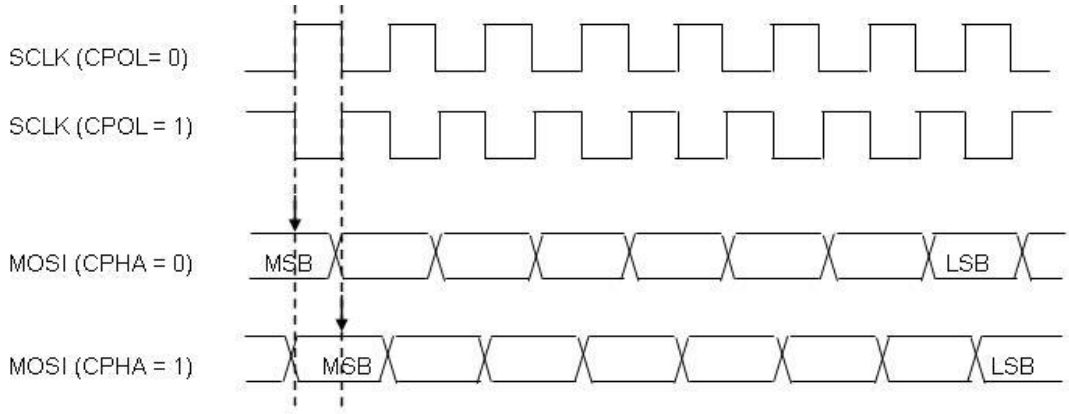


图 10-5 SPI 主控制器时序图

SPI Flash 访问时序图

- 标准读模式

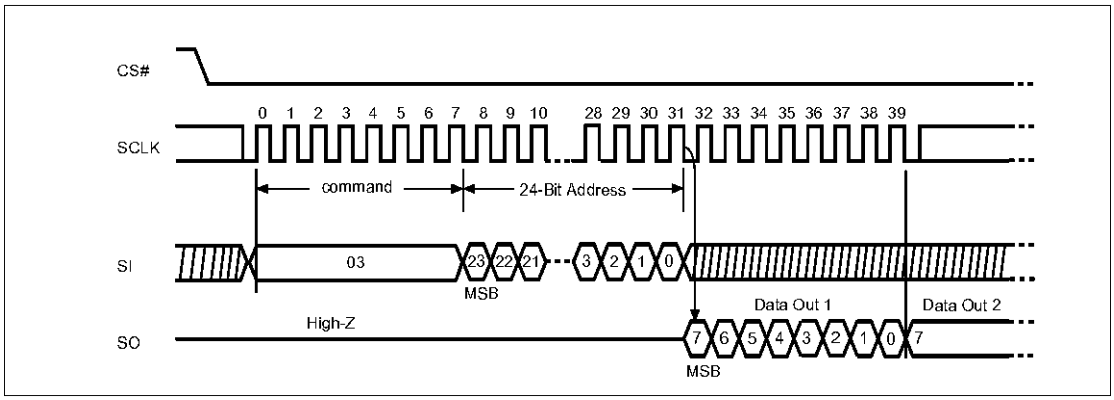


图 10-6 SPI 标准读时序

- 快速读模式

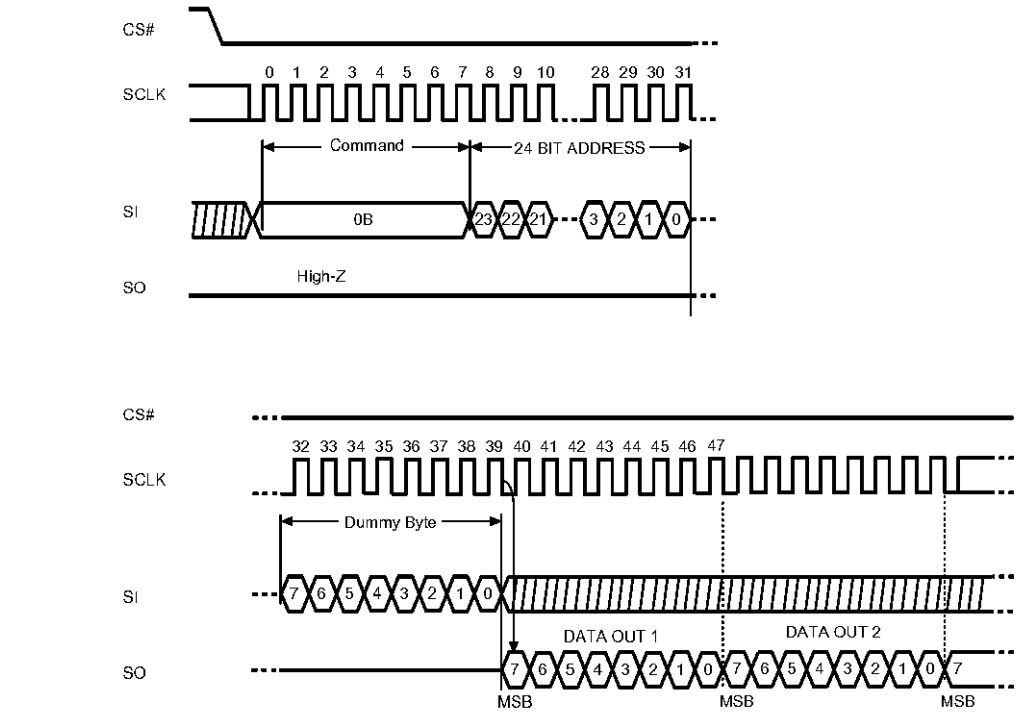


图 10-7 SPI 快速读时序

- 双 I/O 模式

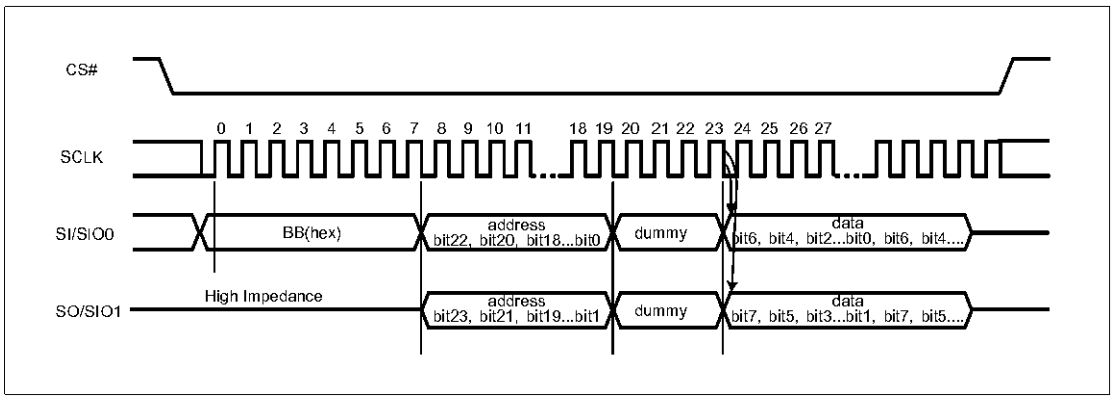


图 10-8 SPI 双 I/O 读时序

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。

10.4.10 SPI Flash控制器使用指南

本模块寄存器物理地址基址为 0x1F004000。

SPI 主控制器的读写操作

1. 模块初始化

- 设置片选控制寄存器（SFC_SOFTCS），并使能控制寄存器
- 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. 使用多个 SPI 外设

SPI Flash 控制器提供 4 个软件可控制的片选 cs[3:0]，其中 cs[0]专用于 SPI Flash。在龙芯 3B1000 中，SPI Flash 控制器的片选与 GPIO[3:0]对应。软件可通过 SFC_SOFTCS 控制寄存器设置 cs[3:1]，来访问所选择的 SPI 设备，当 SFC_SOFTCS 的 csen 有效时，对应的 GPIO 位由 SPI Flash 控制器控制。如果软件选中了其它 SPI 设备，此时又出现 SPI Flash 读访问(比如取指)，cs[3:1]会自动无效，Flash 访问结束后恢复原值。

2. 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 cs[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)，具体参见相关文档。

10.5 IO 控制器配置

配置寄存器主要用于配置 PCI/PIC-X 控制器的地址窗口、仲裁器以及 GPIO 控制器。表 10-6 列出了这些寄存器，表 10-7 给出寄存器的详细说明。这部分寄存器基地址为 0x1FE00100。

表 10-6 IO 控制寄存器

地 址	寄存器	说 明
00	PonCfg	上电配置
04	GenCfg	常规配置
08	保留	
0C	保留	
10	PCIMap	PCI 映射
14	PCIX_Bridge_Cfg	PCI/X 桥相关配置
18	PCIMap_Cfg	PCI 配置读写设备地址
1C	GPIO_Data	GPIO 数据
20	GPIO_EN	GPIO 方向
24	保留	
28	保留	
2C	保留	
30	保留	
34	保留	
38	保留	
3C	保留	
40	Mem_Win_Base_L	可预取窗口基址低 32 位
44	Mem_Win_Base_H	可预取窗口基址高 32 位
48	Mem_Win_Mask_L	可预取窗口掩码低 32 位
4C	Mem_Win_Mask_H	可预取窗口掩码高 32 位
50	PCI_Hit0_Sel_L	PCI 窗口 0 控制低 32 位
54	PCI_Hit0_Sel_H	PCI 窗口 0 控制高 32 位
58	PCI_Hit1_Sel_L	PCI 窗口 1 控制低 32 位

5C	PCI_Hit1_Sel_H	PCI 窗口 1 控制高 32 位
60	PCI_Hit2_Sel_L	PCI 窗口 2 控制低 32 位
64	PCI_Hit2_Sel_H	PCI 窗口 2 控制高 32 位
68	PXArb_Config	PCIX 仲裁器配置
6C	PXArb_Status	PCIX 仲裁器状态
70		
74		
78		
7C		
80	Chip Config	芯片配置寄存器
84		
88		
8C		
90	Chip Sample	芯片采样寄存器

表 10-7 寄存器详细描述

位域	字段名	访问	复位值	说明
CR00: PonCfg				
15:0	pcix_bus_dev	只读	lio_ad[7:0]	PCIX Agent 模式下 CPU 取指所使用的总线、设备号
15:8	保留	只读	lio_ad[15:8]	
23:16	pon_pci_configi	只读	pci_configi	PCI_Configi 引脚值
31:24	保留	只读		
CR04: 保留				
31:0	保留	只读	0	
CR08: 保留				
31:0	保留	只读	0	
CR10: PCIMap				
5:0	trans_lo0	读写	0	PCI_Mem_Lo0 窗口映射地址高 6 位
11:6	trans_lo1	读写	0	PCI_Mem_Lo1 窗口映射地址高 6 位
17:12	trans_lo2	读写	0	PCI_Mem_Lo2 窗口映射地址高 6 位
31:18	保留	只读	0	
CR14: PCIX_Bridge_Cfg				
5:0	pcix_rgate	读写	6'h18	PCIX 模式下向 DDR2 发读取数门限

6	pcix_ro_en	读写	0	PCIX 桥是否允许写越过读
31:18	保留	只读	0	
CR18: PCIMap_Cfg				
15:0	dev_addr	读写	0	PCI 配置读写时 AD 线高 16 位
16	conf_type	读写	0	配置读写的类型
31:17	保留	只读	0	
CR1C: GPIO_Data				
3:0	gpio_out	读写	0	GPIO 输出数据
15:4	保留	只读	0	
19:16	gpio_in	读写	0	GPIO 输入数据
31:20	保留	只读	0	
CR20: GPIO_EN				
3:0	gpio_en	读写	F	高为输入，低输出
31:4	保留	只读	0	
CR3C: 保留				
31:0	保留	只读	0	保留
CR24,2C,30,34,38:保留				
见表 11-3				
CR50,54/58,5C/60,64: PCI_Hit*_Sel_*				
0	保留	只读	0	
2:1	pci_img_size	读写	2'b11	00: 32 位；10: 64 位；其它: 无效
3	pref_en	读写	0	预取使能
11:4	保留	只读	0	
62:12	bar_mask	读写	0	窗口大小掩码（高位 1，低位 0）
63	burst_cap	读写	1	是否允许突发传送
CR68:PXArb_Config				
0	device_en	读写	1	外部设备允许
1	disable_broken	读写	0	禁用损坏的主设备
2	default_mas_en	读写	1	总线停靠到默认主设备 0：停靠到最后一个主设备 1：停靠到默认主设备
5:3	default_master	读写	0	总线停靠默认主设备号
7:6	park_delay	读写	2'b11	从没有设备请求总线开始到触发停靠默认设备行为的延迟 00: 0 周期 01: 8 周期 10: 32 周期 11: 128 周期
15:8	level	读写	8'h01	处于第一级的设备

23:16	rude_dev	读写	0	强制优先级设备 为 1 的位对应的 PCI 设备在得到总线后可以通过持续请求来占住总线
31:13	保留	只读	0	
CR6C: PXArb_Status				
7:0	broken_master	只读	0	损坏的主设备（改变禁用策略时清零）
10:8	Last_master	只读	0	最后使用总线的主设备
31:11	保留	只读	0	
CR80: Chip config				
2:0	Freq_scale_ctrl	读写	3’b111	处理器核分频
3	DDR_Clksel_en	读写	1’b0	是否使用软件配置 DDR 倍频
8	Disable_ddr2_confspace	读写	1’b0	是否禁用 DDR 配置空间
9	DDR_buffer_cpu	读写	1’b0	是否打开 DDR 读访问缓冲
12	Core0_en	读写	1’b1	是否启用处理器核 0
13	Core1_en	读写	1’b1	是否启用处理器核 1
14	Core2_en	读写	1’b1	是否启用处理器核 2
15	Core3_en	读写	1’b1	是否启用处理器核 3
16	Mc0_en	读写	1’b1	是否启用 DDR 控制器 0
17	Mc1_en	读写	1’b1	是否启用 DDR 控制器 1
18	DDR_reset0	读写	1’b1	软件 reset DDR 控制器 0
19	DDR_reset1	读写	1’b1	软件 reset DDR 控制器 1
28:24	DDR_Clksel	读写	5’b11111	软件配置 DDR 时钟倍频关系（当 DDR_Clksel_en 为 1 时有效）
31:29	HT_freq_scale_ctrl0	读写	3’b111	HT 控制器 0 分频
34:32	HT_freq_scale_ctrl0	读写	3’b111	HT 控制器 1 分频
其它		只读		保留
CR90: Chip Sample				
15:0	Pad2v5_ctrl	读写	16’h780	2v5pad 控制
31:16	Pad3v3_ctrl	读写	16’h780	3v3pad 控制
47:32	Sys_clksel	只读		板上倍频设置
51:48	Bad_ip_core	只读		4 个处理器核是否坏
53:52	Bad_ip_ddr	只读		2 个 DDR 控制器是否坏
57:56	Bad_ip_ht	只读		2 个 HT 控制器是否坏
102:96	Thsens0_out	只读		温度传感器 0 温度
103	Thsens0_overflow	只读		温度传感器 0 温度上溢（超过 128 度）
110:104	Thsens1_out	只读		温度传感器 1 温度
111	Thsens1_overflow	只读		温度传感器 1 温度上溢（超过 128 度）
其它		只读		保留