Loongson 3A3000 / 3B3000 processor

User Manual

volume One

Multi-core processor architecture, register description and system software programming guide

V1.3

2017 Nian ***4*** Yue

Loongson Zhongke Technology Co., Ltd.

Copyright Notice

Disclaimer

This document only provides periodic information, and the content can be updated at any time according to the actual situation of the product without notice. Ruin The company does not assume any responsibility for direct or indirect losses caused by improper use of documents.

Loongson Zhongke Technology Co., Ltd.

Loongson Technology Corporation Limited

Address: Building 2, Longxin Industrial Park, Zhongguancun Environmental Protection Technology Demonstration Park, Haidian District, Beijing

Building No. 2, Loongson Industrial Park,

Zhongguancun Environmental Protection Park, Haidian District, Beijing

Telephone (Tel): 010-62546668

Fax: 010-62600826

Reading guide

"Godson 3A3000 / 3B3000 Processor User Manual" is divided into the first and second volumes.

"Loongson 3A3000 / 3B3000 Processor User Manual" is divided into two parts, the first part introduces Loongson 3A3000 / 3B3000

Multi-core processor architecture and register descriptions, on-chip system architecture, main module functions and configuration, register list and

The domain is described in detail.

Volume 2 of the "Loongson 3A3000 / 3B3000 Processor User Manual" introduces Loongson in detail from the perspective of system software developers

3A3000 / 3B3000 uses the GS464e high-performance processor core.

revise history

Document update record	Document name:	Loongson 3A3000 / 3B3000 Processor User Manual --volume One
	version number	V1.3
	founder:	Chip R & D Department
	Creation Date:	2017-04-13

Update history

Serial number	Updated	version number	update content
1	2016-06-14	V1.0	initial version
2	2016-09-14	V1.1	Updated some PLL configuration register descriptions, updated software and hardware changes descriptions
3	2016-11-25	V1.2	Add register description of BBGEN section
4	2017-04-13	V1.3	Add SPI address space description

Manual feedback: service@loongson.cn

You can also submit chip production to our company through the problem feedback website http://bugs.loongnix.org/

Problems in the use of products, and obtain technical support.

table of Contents

I

Page 6

II

Page 7

Figure catalog

Table directory

x

Page 15

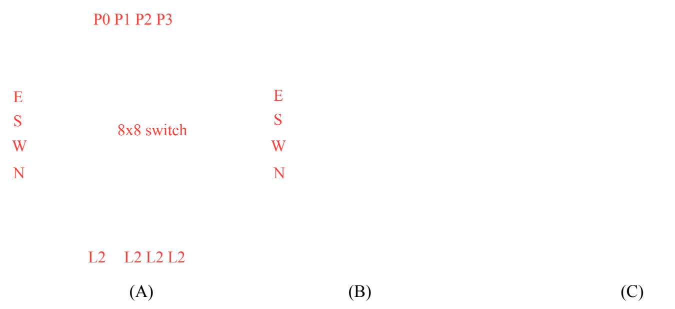
1 Overview

1.1 Introduction to Loongson series processors

Loongson processor mainly includes three series. Loongson No. 1 processor and its IP series are mainly for embedded applications. Core 2 superscalar processor and its IP series are mainly for desktop applications, and Godson 3 multi-core processor series is mainly for service Server and high-performance machine applications. According to the needs of the application, some of Loongson 2 can also face some high-end embedded applications, Some low-end Loongson 3 can also be used for some desktop applications. The above three series will be developed in parallel.

Loongson 3 multi-core series processors are based on a scalable multi-core interconnect architecture design, integrating multiple high-performance on a single chip The processor core and a large number of level 2 caches, and the interconnection of multiple chips through high-speed I / O interfaces to form a larger scale system.

The scalable interconnection structure adopted by Godson 3 is shown in Figure 1-1 below. Both the on-chip and multi-chip systems of Godson No. 3 adopt two Dimension mesh interconnection structure, where each node is composed of 8 * 8 crossbars, each crossbar is connected to four processor cores And four shared caches, and interconnect with other nodes in four directions of east (E) south (N) west (W) north (N). therefore, 2 * 2 meshes can be connected to 16 processor cores, and 4 * 4 meshes can be connected to 64 processor cores.



Loongson No. 3 node and two-dimensional interconnection structure, (a) node structure, (b) 2 * 2 mesh network connected to 16 processors, (c) The 4 * 4 mesh network connects 64 processors.

Figure 1-1 Loongson No. 3 system structure

The structure of Loongson No. 3 node is shown in Figure 1-2 below. Each node has two levels of AXI crossbars connected to the processor and shared

Cache, memory controller and IO controller. Among them, the first level AXI crossbar switch (called X1 Switch, referred to as X1)

Connect the processor and shared cache. The second level crossbar switch (called X2 Switch, referred to as X2 for short) is connected to share Cache and

Memory controller.

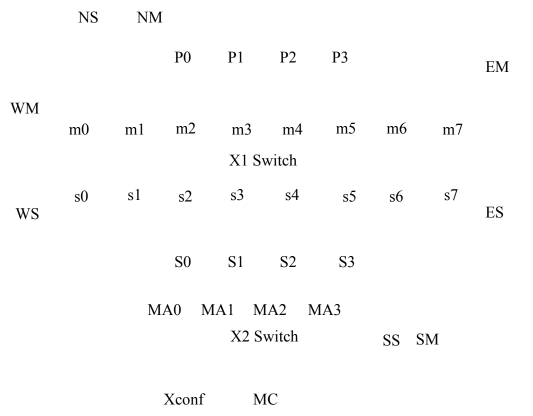


Figure 1-2 Loongson No. 3 node structure

In each node, up to 8 * 8 X1 crossbars are connected to four GS464 processor cores through four Master ports

(P0, P1, P2, P3 in the figure), connected to four interleave shared caches with four slave ports through four slave ports

Block (S0, S1, S2, S3 in the figure), connected to the four directions of east, south, west and north through four pairs of Master / Slave

Other nodes or IO nodes (EM / ES, SM / SS, WM / WS, NM / NS in the figure).

The X2 crossbar is connected to four shared caches through four Master ports, and one is connected to at least one Slave port

Memory controller, at least one Slave port connected to a crossbar configuration module (Xconf) is used to configure this node

The X1 and X2 address windows, etc. You can also connect more memory controllers and IO ports as needed.

1.2 Introduction to Godson 3A3000 / 3B3000

Loongson 3A3000 / 3B3000 is a process upgrade version of Loongson 3A2000 / 3B2000 quad-core processor.

Compared with the core 3A1000, PLL_AVDD is changed from 2.5V to 1.8V, which is more used than the Loongson 3A2000 / 3B2000

PLL_AVDD. Loongson 3A3000 / 3B3000 is a single-node 4-core processor, manufactured using 28nm process,

The main frequency is 1.2GHz-1.5GHz, and the main technical characteristics are as follows:

Page 17

- Four 64-bit super-scalar GS464e high-performance processor cores are integrated on-chip;
- On-chip integrated 8MB split shared three-level cache (composed of 4 individual modules, each module has a capacity of 2MB);
- Maintain the cache consistency of multi-core and I / O DMA access through the directory protocol;
- Two 64-bit DDR2 / 3 controllers with ECC and 667MHz are integrated on-chip;
- 3B3000 integrates two 16-bit 1.6GHz HyperTransport controllers (hereinafter referred to as HT);
- 3A3000 on-chip HT1 is a 16-bit 1.6GHz HT controller, HT0 is not available;
- Each 16-bit HT port is split into two 8-way HT ports for use.
- On-chip integrated 32-bit 33MHz PCI;
- Integrate 1 LPC, 2 UARTs, 1 SPI, 16 GPIO interfaces on-chip.

Compared with Loongson 3A2000 / 3B2000, the main improvements are as follows:

- Processor core microstructure upgrade;
- Memory controller structure and frequency upgrade;
- HT controller structure and frequency upgrade;
- The performance of the whole chip is optimized and improved.

The overall architecture of Loongson 3A3000 / 3B3000 chip is based on two-level interconnection. The structure is shown in Figures 1-3 below.

Figure 1-3 Loongson 3A3000 / 3B3000 chip structure

The first level interconnection uses a 6x6 crossbar switch, which is used to connect four GS464e cores (as a master device) and four shares Cache module (as a slave device), and two IO ports (each port uses a Master and a Slave).

Page 18

Each IO port connected to the first-level interconnect switch is connected to a 16-bit HT controller, and each 16-bit HT port can also Used as two 8-bit HT ports. The HT controller is connected to the first-level interconnect switch through a DMA controller. The DMA controller The controller is responsible for the DMA control of the IO and the maintenance of the consistency between the slices. The DMA controller of Godson 3 can also be configured Realize prefetching and matrix transposition or transfer.

The second level interconnection uses a 5x4 crossbar switch to connect 4 shared Cache modules (as the master device), two DDR2 / 3 Memory controller, low-speed high-speed I / O (including PCI, LPC, SPI, etc.) and configuration register module inside the chip.

The above two-level interconnect switches all use separate data channels for reading and writing. The width of the data channel is 128 bits.

The processor core has the same frequency to provide high-speed on-chip data transmission.

2 System configuration and control

2.1 Chip working mode

According to the structure of the system, Loongson 3A3000 / 3B3000 mainly includes three working modes:

- Single chip mode. The system contains only one Loongson 3A3000 / 3B3000, which is a symmetric multiprocessor system (SMP);
- Multi-chip interconnect mode. The system contains 2 or 4 Loongson 3A3000 / 3B3000, through Loongson

  The HT ports of 3A3000 / 3B3000 are interconnected, which is a non-uniform memory multiprocessor system (CC-NUMA)
- Large-scale interconnection model. Large-scale multi-chip expansion interconnection through dedicated expansion bridges, forming a large-scale non-uniform

  Uniform access to multi-processor systems (CC-NUMA).

2.2 Description of control pins

The main control pins include DO_TEST, ICCC_EN, NODE_ID [1: 0], CLKSEL [15: 0], PCI_CONFIG.

Table 2- 1 Control pin description

signal	Up and down	effect

DO_TEST	pull up	1'b1 means function mode 1'b0 means test mode
ICCC_EN	drop down	1'b1 means multi-chip consistent interconnect mode 1'b0 means single chip mode
NODE_ID [1: 0]		Indicates the processor number in multi-chip consistent interconnect mode
CLKSEL [15: 0]		HT clock control

signal	effect
CLKSEL [15]	1'b1 means the HT controller frequency is only set by hardware 1'b0 means HT controller frequency can be set by software
CLKSEL [14]	1'b1 means HT PLL uses normal clock input 1'b0 means HT PLL uses differential clock input
CLKSEL [13:12]	2'b00 means the PHY clock is 1.6GHZ 2'b01 indicates that the PHY clock is 3.2GHZ 2'b10 means the PHY clock is 1.2GHz 2'b11 means the PHY clock is 2.4GHz

signal	effect
CLKSEL [11:10]	2'b00 indicates that the HT controller clock is divided by 8 of the PHY clock 2'b01 indicates that the HT controller clock is divided by 4 of the PHY clock 2'b10 means the HT controller clock is divided by 2 of the PHY clock 2'b11 indicates that the HT controller clock is SYSCLOCK

Note: When CLKSEL [13:10] == 4'b1111, the HT controller clock is in bypass mode and used directly

External input 100MHz reference clock

MEM clock control

signal	effect
CLKSEL [9: 5]	5'b11111 means MEM clock directly uses memclk 5'b01111 indicates that the MEM clock is set by software. For the setting method, see Section 2.6 In other cases, the MEM clock is memclk * (clksel [8: 5] +30) / (clksel [9] +3) Note: memclk * (clksel [8: 5] +30) must be 1.2GHz ~ 3.2GHz memclk is the input reference clock, which must be 20 ~ 40MHz

CORE clock control

signal	effect
CLKSEL [4: 0]	5'b11111 indicates that the CORE clock directly uses sysclk 5'b011xx indicates that the CORE clock is set by software. For the setting method, see Instructions in Section 2.6. 5'b01111 is normal working mode, otherwise it is debugging mode 5'b0110x indicates that the processor interface is in asynchronous mode 5'b011x0 means delayed debug control mode In other cases, the CORE clock is sysclk * (clksel [3: 0] +30) / (clksel [4] +1) Note:

	sysclk * (clksel [3: 0] +30) must be 1.2GHz ~ 3.2GHz sysclk is the input reference clock, which must be 20 ~ 40MHz
PCI_CONFIG [7: 0]	IO configuration control 7 HT bus cold start is forced to 1.0 mode 6: 4 needs to be set to 000 3 PCI master mode 2 Need to be set to 0 1 Use external PCI arbitration 0 Use SPI boot function

2.3 Cache consistency

Loongson 3A3000 / 3B3000 maintains the cache between the processor and the I / O accessed through the HT port It is consistent, but the hardware does not maintain the cache consistency of the I / O devices connected to the system through PCI. During driver development, When DMA (Direct Memory Access) transmission is performed on a device connected via PCI, the software needs to perform Cache Consistency maintenance.

2.4 Distribution of physical address space at the node level of the system

The system physical address distribution of Loongson No. 3 series processors adopts a globally accessible hierarchical addressing design The extensions developed by the system are compatible. The physical address width of the entire system is 48 bits. According to the upper 4 bits of the address, the entire address sp It is evenly distributed to 16 nodes, that is, each node is allocated 44-bit address space.

Loongson 3A3000 / 3B3000 processor can directly use 4 chips to connect directly to build CC-NUMA system. The processor number is determined by the pin NODEID, and the address space of each chip is distributed as follows:

Table 2-2 Node-level system global address distribution

Chip node number (NODEID)	Address [47:44] bits	starting address	End address
0	0	0x0000_0000_0000	0x0FFF_FFFF_FFFF
1	1	0x1000_0000_0000	0x1FFF_FFFF_FFFF
2	2	0x2000_0000_0000	0x2FFF_FFFF_FFFF
3	3	0x3000_0000_0000	0x3FFF_FFFF_FFFF

Loongson 3A3000 / 3B3000 uses a single-node 4-core configuration, so Loongson 3A3000 / 3B3000 chip integrated DDR The corresponding addresses of the memory controller, HT bus, and PCI bus are included from 0x0 (inclusive) to 0x1000_0000_0000 (not The 44-bit address is in each node, and the 44-bit address space is further evenly distributed to the most likely connection within the node. 8 more devices. The lower 43 bits of addresses are owned by 4 shared cache modules, and the higher 43 bits of addresses are further The [43:42] bits are distributed to devices connected to the 4 directional ports. According to the different configuration of chip and system structure, if If there is no slave device connected to a port, the corresponding address space is reserved address space, and access is not allowed.

Page 22

The slave devices corresponding to the address space of the first-level crossbar in Loongson 3A3000 / 3B3000 are as follows:

Table 2-3 Address distribution in nodes

device	Address [43:41]	Start address within the node	Node end address
Shared Cache	0,1,2,3	0x000_0000_0000	0x7FF_FFFF_FFFF
HT0 controller	6	0xC00_0000_0000	0xDFF_FFFF_FFFF
HT1 controller	7	0xE00_0000_0000	0xFFF_FFFF_FFFF

Unlike the mapping relationship of direction ports, Loongson 3A3000 / 3B3000 can be determined according to the actual application access behavior

Set the cross-addressing mode of shared cache. The 4 shared Cache modules in the node correspond to a total of 43 address spaces, and

The address space corresponding to each module is determined according to one of the two selection bits of the address bit, and can be dynamically configured by software

modify. The configuration register named SCID_SEL is set in the system to determine the address selection bits, as shown in the following table. In default

In this case, it is distributed by means of [7: 6] status hash, that is, two bits of address [7: 6] determine the corresponding shared cache number.

The register address is 0x3FF00400.

Table 2-4 Address distribution in nodes

SCID_SEL	Address bit selection	SCID_SEL	Address bit selection
4'h0	7: 6	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 Distribution of physical address space within a node

The default distribution of the internal 44-bit physical address of each node of Loongson 3A3000 / 3B3000 processor is shown in the following table:

Table 2-2 44-bit physical address distribution in the node

starting address	End address	name	Explanation
0x0000_0000_0000	0x0000_0FFF_FFFF	RAM	Need to use two-level crossbar for mapping
0x0000_1000_0000	0x0000_1FFF_FFFF	Low speed IO	Need to use two-level crossbar for mapping

Page 23

0x2000_0000_0000	0x2FFF_FFFF_FFFF	2	2
0x3000_0000_0000	0x3FFF_FFFF_FFFF	3	3

Loongson 3A3000 / 3B3000 uses a single-node 4-core configuration, so Loongson 3A3000 / 3B3000 chip integrated DDR

2.6 Address Routing Distribution and Configuration

The routing of Loongson 3A3000 / 3B3000 is mainly realized through the two-stage crossbar of the system. First-level crossbar can

Each Master port receives requests for routing configuration, each Master port has 8 address windows, you can

Complete target routing in 8 address windows. Each address window consists of three 64-bit registers BASE, MASK and MMAP

Composed, BASE is aligned in K bytes; MASK adopts a format similar to the high bit of the netmask; the lower three bits of MMAP indicate the pair

According to the number of the target Slave port, MMAP [4] means to allow instruction fetch, MMAP [5] means to allow block read, MMAP [6] means

Allow interleaved access to Scache to be enabled, MMAP [7] means window is enabled.

Table 2-5 The space access attributes corresponding to the MMAP field

[7]	[6]	[5]	[4]
Window enable	Allow interleaved access to SCACHE, valid when the slave number is 0, according to the above A section of SCID_SEL configuration routes requests that hit window addresses	Block read	Allow fetching

Window hit formula: (IN_ADDR & MASK) == BASE

Since Loongson 3 uses fixed routing by default, the configuration window is closed when the power is turned on.

System software is required to enable and configure it.

The address window conversion register is shown in the table below.

Table 2-6 Register Table of Address Window of Primary Crossbar

address	register	address	register
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_23b0	CORE3_WIN6_MMAP
0x3ff0_22b8	CORE2_WIN7_MMAP	0x3ff0_23b8	CORE3_WIN7_MMAP

0x3ff0_2600	HT0_WIN0_BASE	0x3ff0_2700	HT1_WIN0_BASE
0x3ff0_2608	HT0_WIN1_BASE	0x3ff0_2708	HT1_WIN1_BASE
0x3ff0_2610	HT0_WIN2_BASE	0x3ff0_2710	HT1_WIN2_BASE
0x3ff0_2618	HT0_WIN3_BASE	0x3ff0_2718	HT1_WIN3_BASE
0x3ff0_2620	HT0_WIN4_BASE	0x3ff0_2720	HT1_WIN4_BASE
0x3ff0_2628	HT0_WIN5_BASE	0x3ff0_2728	HT1_WIN5_BASE
0x3ff0_2630	HT0_WIN6_BASE	0x3ff0_2730	HT1_WIN6_BASE
0x3ff0_2638	HT0_WIN7_BASE	0x3ff0_2738	HT1_WIN7_BASE
0x3ff0_2640	HT0_WIN0_MASK	0x3ff0_2740	HT1_WIN0_MASK
0x3ff0_2648	HT0_WIN1_MASK	0x3ff0_2748	HT1_WIN1_MASK
0x3ff0_2650	HT0_WIN2_MASK	0x3ff0_2750	HT1_WIN2_MASK
0x3ff0_2658	HT0_WIN3_MASK	0x3ff0_2758	HT1_WIN3_MASK
0x3ff0_2660	HT0_WIN4_MASK	0x3ff0_2760	HT1_WIN4_MASK
0x3ff0_2668	HT0_WIN5_MASK	0x3ff0_2768	HT1_WIN5_MASK
0x3ff0_2670	HT0_WIN6_MASK	0x3ff0_2770	HT1_WIN6_MASK
0x3ff0_2678	HT0_WIN7_MASK	0x3ff0_2778	HT1_WIN7_MASK
0x3ff0_2680	HT0_WIN0_MMAP	0x3ff0_2780	HT1_WIN0_MMAP

0x3ff0_2688	HT0_WIN1_MMAP	0x3ff0_2788	HT1_WIN1_MMAP
0x3ff0_2690	HT0_WIN2_MMAP	0x3ff0_2790	HT1_WIN2_MMAP
0x3ff0_2698	HT0_WIN3_MMAP	0x3ff0_2798	HT1_WIN3_MMAP
0x3ff0_26a0	HT0_WIN4_MMAP	0x3ff0_27a0	HT1_WIN4_MMAP
0x3ff0_26a8	HT0_WIN5_MMAP	0x3ff0_27a8	HT1_WIN5_MMAP
0x3ff0_26b0	HT0_WIN6_MMAP	0x3ff0_27b0	HT1_WIN6_MMAP
0x3ff0_26b8	HT0_WIN7_MMAP	0x3ff0_27b8	HT1_WIN7_MMAP

There are configuration register address space, DDR2 address space, and PCI address space in the second-level XBAR of Godson 3

There are three IP-related address spaces. The address window is for the CPU and PCI-DMA two IPs with master device functions

It is set for routing and address translation. Both CPU and PCI-DMA have 8 address windows, which can complete the target

The choice of address space and the conversion from source address space to target address space.

Each address window is composed of three 64-bit registers BASE, MASK and MMAP, BASE is aligned with K bytes, MASK

Using a format similar to the high-order bit of the netmask, MMAP contains the converted address, routing and enable control bits,

As shown in the following table:

[63:48]	[47:10]	[7: 4]	[3: 0]
Interleaved selection bit	Address after conversion	Window enable	Slave number

Among them, the device corresponding to the slave device number is shown in the following table:

Table 2-7 Correspondence between the slave device number and the module at the secondary XBAR

Slave number	Default value
0	No. 0 DDR2 / 3 controller
1	No. 1 DDR2 / 3 controller
2	Low-speed I / O (PCI, LPC, etc.)
3	Configuration register

The meaning of the window enable bit is shown in the following table:

Table 2-8 The space access attributes corresponding to the MMAP field

[7]	[6]	[5]	[4]
Window enable	Allow interleaved access to DDR, valid when the slave device number is 0, according to "interleaved Select bit configuration to route requests that hit the window address. The interleaving enable bit is required Greater than 10	Block read	Allow fetching

It should be noted that the window configuration of the first-level XBAR cannot perform address translation for Cache consistency requests, otherwise

The address at the SCache will be inconsistent with the address at the first-level cache of the processor, resulting in incorrect maintenance of Cache consistency.

Window hit formula: (IN_ADDR & MASK) == BASE

New address conversion formula: OUT_ADDR = (IN_ADDR & ~ MASK) | {MMAP [63:10], 10'h0}

The address window conversion register is as follows:

Table 2-9 Secondary XBAR address window conversion register table

address	register	description	Default value

Page 27

3ff0 0000 CPU_WIN0_BASE CPU window 0 base address	0x0
3ff0 0008 CPU_WIN1_BASE CPU window 1 base address	0x1000_0000
3ff0 0010 CPU_WIN2_BASE CPU window 2 base address	0x0
3ff0 0018 CPU_WIN3_BASE CPU window 3 base address	0x0
3ff0 0020 CPU_WIN4_BASE CPU window 4 base address	0x0
3ff0 0028 CPU_WIN5_BASE CPU window 5 base address	0x0
3ff0 0030 CPU_WIN6_BASE CPU window 6 base address	0x0
3ff0 0038 CPU_WIN7_BASE CPU window 7 base address	0x0
3ff0 0040 CPU_WIN0_MASK CPU window 0 mask	0xffff_ffff_f000_0000
3ff0 0048 CPU_WIN1_MASK CPU window 1 mask	0xffff_ffff_f000_0000
3ff0 0050 CPU_WIN2_MASK CPU window 2 mask	0x0
3ff0 0058 CPU_WIN3_MASK CPU window 3 mask	0x0
3ff0 0060 CPU_WIN4_MASK CPU window 4 mask	0x0
3ff0 0068 CPU_WIN5_MASK Mask of CPU window 5	0x0
3ff0 0070 CPU_WIN6_MASK CPU window 6 mask	0x0
3ff0 0078 CPU_WIN7_MASK CPU window 7 mask	0x0
3ff0 0080 CPU_WIN0_MMAP CPU window 0 new base address 0xf0	
3ff0 0088 CPU_WIN1_MMAP CPU window 1 new base address 0x1000_00f2	
3ff0 0090 CPU_WIN2_MMAP CPU window 2 new base address 0	
3ff0 0098 CPU_WIN3_MMAP CPU window 3 new base address 0	
3ff0 00a0 CPU_WIN4_MMAP CPU window 4 new base address 0x0	
3ff0 00a8 CPU_WIN5_MMAP CPU window 5 new base address 0x0	
3ff0 00b0 CPU_WIN6_MMAP CPU window 6 new base address 0	
3ff0 00b8 CPU_WIN7_MMAP CPU window 7 new base address 0	
3ff0 0100 PCI_WIN0_BASE PCI window 0 base address	0x8000_0000
3ff0 0108 PCI_WIN1_BASE PCI window 1 base address	0x0
3ff0 0110 PCI_WIN2_BASE PCI window 2 base address	0x0

Page 28

3ff0 0118 PCI_WIN3_BASE PCI window 3 base address	0x0
3ff0 0120 PCI_WIN4_BASE PCI window 4 base address	0x0
3ff0 0128 PCI_WIN5_BASE PCI window 5 base address	0x0
3ff0 0130 PCI_WIN6_BASE PCI window 6 base address	0x0

3ff0 0138 PCI_WIN7_BASE PCI window 7 base address 0x0

3ff0 0140 PCI_WIN0_MASK PCI window 0 mask 0xffff_ffff_8000_0000

3ff0 0148 PCI_WIN1_MASK Mask of PCI window 1 0x0

3ff0 0150 PCI_WIN2_MASK PCI window 2 mask 0x0

3ff0 0158 PCI_WIN3_MASK PCI window 3 mask 0x0

3ff0 0160 PCI_WIN4_MASK PCI window 4 mask 0x0

3ff0 0168 PCI_WIN5_MASK PCI window 5 mask 0x0

3ff0 0170 PCI_WIN6_MASK Mask of PCI window 6 0x0

3ff0 0178 PCI_WIN7_MASK Mask of PCI window 7 0x0

3ff0 0180 PCI_WIN0_MMAP PCI window 0 new base address 0xf0

3ff0 0188 PCI_WIN1_MMAP PCI window 1 new base address 0x0

3ff0 0190 PCI_WIN2_MMAP New base address of PCI window 2 0

3ff0 0198 PCI_WIN3_MMAP PCI window 3 new base address 0

3ff0 01a0 PCI_WIN4_MMAP PCI window 4 new base address 0x0

3ff0 01a8 PCI_WIN5_MMAP PCI window 5 new base address 0x0

3ff0 01b0 PCI_WIN6_MMAP New base address of PCI window 6 0

3ff0 01b8 PCI_WIN7_MMAP PCI window 7 new base address 0

According to the default register configuration, after the chip is started, the address range of 0x00000000-0x0fffffff of the CPU (256M) mapped to the address range of 0x00000000-0x0fffffff of DDR2, 0x10000000 of CPU-

0x1fffffff interval (256M) is mapped to PCI 0x10000000-0x1fffffff interval, PCIDMA 0x80000000

-The address range (256M) of 0x8fffffff is mapped to the address range of 0x00000000-0x0fffffff of DDR2.

The software can implement new address space routing and conversion by modifying the corresponding configuration registers.

In addition, when there is a read access to an illegal address due to CPU speculative execution, none of the eight address windows hit. The configuration register module returns all 0 data to the CPU to prevent the CPU from dying.

Table 2- 10 Level 2 XBAR default address configuration

Base address	High position	owner
0x0000_0000_0000_0000	0x0000_0000_0FFF_FFFF	No. 0 DDR controller
0x0000_0000_1000_0000	0x0000_0000_1FFF_FFFF	Low-speed I / O (PCI, etc.)

2.7 Chip configuration and sampling register

Chip configuration register (Chip_config) and chip sampling register in Godson 3A3000 / 3B3000 (Chip_sample) provides a mechanism to read and write the configuration of the chip.

Table 2-11 Chip Configuration Register (Physical Address 0x1fe00180)

Bit field	Field name	access	Reset value	description
3: 0-		RW	4'b7	Keep
4	MC0_disable_ddr2_confspace	RW	1'b0	Whether to disable MC0 DDR configuration space
5	-	RW	1'b0	Keep

6	-	RW	1'b0	Keep
7	MC0_ddr2_resetn	RW	1'b1	MC0 software reset (active low)
8	MC0_clken	RW	1'b1	Whether to enable MC0
9	MC1_disable_ddr2_confspace	RW	1'b0	Whether to disable MC1 DDR configuration space
10	-	RW	1'b0	Keep
11	-	RW	1'b0	Keep
12	MC1_ddr2_resetn	RW	1'b1	MC1 software reset (active low)
13	MC1_clken	RW	1'b1	Whether to enable MC1
26:24	HT0_freq_scale_ctrl	RW	3'b111	HT controller divide by 0
27	HT0_clken	RW	1'b1	Whether to enable HT0
30:28	HT1_freq_scale_ctrl	RW	3'b111	HT controller divided by 1
31	HT1_clken	RW	1'b1	Whether to enable HT1
42:40	Node0_freq_ ctrl	RW	3'b111	Node 0 frequency division
43	-	RW	1'b1	
46:44	Node1_freq_ ctrl	RW	3'b111	Node 1 frequency division
47	-	RW	1'b1	
63:56	Cpu_version	R	2'h39	CPU version
95:64				(air)
127: 96	Pad1v8_ctrl	RW	6'h780	1v8 pad control
other		R		Keep

Table 2- 12 Chip sampling register (physical address 0x1fe00190)

Bit field	Field name	access	Reset value	description
31: 0	Compcode_core	R		
47:32	Sys_clkseli	R		Onboard frequency setting
55:48	Bad_ip_core	R		core7-core0 is bad
57:56	Bad_ip_ddr	R		Whether 2 DDR controllers are bad
61:60	Bad_ip_ht	R		Whether 2 HT controllers are bad
83:80	Compcode_ok	R		
88	Thsens0_overflow	R		Temperature sensor 0 overflow (over 125 ℃)
89	Thsens1_overflow	R		Temperature sensor 1 overflow (over 125 ℃)
111: 96	Thsens0_out	R		Temperature sensor 0 Celsius Knot point temperature degree= Thens0_out * 731 / 0x4000-273 Temperature range -40 degrees – 125 degrees
127: 112	Thsens1_out	R		Temperature sensor 1 Celsius Knot point temperature degree= Thens1_out * 731 / 0x4000-273 Temperature range -40 degrees – 125 degrees
other		R		Keep

The following sets of software frequency multiplication setting registers are used to set the CLKSEL to software control mode (refer to section 2.2 CLKSEL setting method), the operating frequency of each clock. Among them, MEM CLOCK configuration corresponds to the memory controller and bus Clock frequency; CORE CLOCK corresponds to the clock frequency of the processor core, on-chip network and high-speed shared cache; HT CLOCK pair The HT controller clock frequency should be used.

Each clock configuration generally has two parameters, DIV_LOOPC and DIV_OUT. The final clock frequency is (reference clock * DIV_LOOPC) / DIV_OUT.

For the HT CLOCK configuration method is special, please refer to the specific configuration method in section 10.0.58.

In software control mode, the default corresponding clock frequency is the frequency of the external reference clock (for CORE CLOCK, it is The corresponding frequency of pin SYS_CLK; for MEM CLOCK, the frequency corresponding to pin MEM_CLK) Set the software for the clock during the operation. The process of setting each clock should follow the following methods:

1) The other registers in the setting register except SEL_PLL_ * and SOFT_SET_PLL, that is, these two

The register is written as 0 during the setting process;

2) Other register values remain unchanged, set SOFT_SET_PLL to 1;

3) Wait for the lock signal LOCKED_ * in the register to be 1;

4) Set SEL_PLL_ * to 1, and the corresponding clock frequency will be switched to the frequency set by the software.

In 3A3000 / 3B3000, two different PLLs can be used L1 L2

se al_m 耀 de PLL

Table 2- 13 Chip node and processor core software frequency multiplication setting register (physical address 0x1fe001b0)

Bit field	Field name	access	Reset value	description
0	SEL_PLL_NODE	RW	0x0	Node clock non-software bypass entire PLL
1	SEL_PLL_NODE	RW	0x0	Core clock non-software bypass entire PLL
2	SOFT_SET_PLL	RW	0x0	Allow software to set PLL
3	BYPASS_L1	RW	0x0	Bypass L1 PLL
15: 4	-	RW	0x0	-
16	LOCKED_L1	R	0x0	Whether L1 PLL is locked
17	LOCKED_L2	R	0x0	Whether L2 PLL is locked
18:17-		R	0x0	-
19	PD_L1	RW	0x0	Turn off L1 PLL
20	PD_L2	RW	0x0	Turn off L2 PLL
twenty one				
twenty two	Serial_mode	RW	0x0	0: Select L1 PLL as the main clock 1: Select L2 PLL as the main clock
twenty three	Serial_mode3	RW	0x0	0: Use NODE clock as the core clock 1: Select CORE clock as the core clock
25:24-		RW		-
31:26	L1_DIV_REFC	RW	0x1	L1 PLL input parameters
40:32	L1_DIV_LOOPC	RW	0x1	L1 PLL input parameters
41				
47:42	L1_DIV_OUT	RW	0x1	L1 PLL input parameters
50:48	L2_DIV_REFC	RW	0x1	L2 PLL input parameters
53:51				
63:54	L2_DIV_LOOPC	RW	0x1	L2 PLL input parameters
69:64	L2_DIV_OUT	RW	0x1	L2 PLL input parameters Only one bit is required
96	BBGEN_enable	RW	0x0	Bias enable
97	BBMUX_first	RW	0x0	Set to switch voltage mode first
99:98	BBMUX_SEL_0	RW	0x0	BBMUX_SEL_0 setting value
101: 100	BBGEN_feedback	RW	0x0	Disable BBGEN feedback signal
107: 104	BBGEN_vbbp_val	WO	0x0	Setting value of Vbbp
111: 108	BBGEN_vbbn_val	WO	0x0	Setting value of Vbbn
123: 122	BBMUX_SEL_1	RW	0x0	BBMUX_SEL_1 setting value
125: 124	BBMUX_SEL_2	RW	0x0	BBMUX_SEL_2 setting value

127: 126	BBMUX_SEL_3	RW	0x0	BBMUX_SEL_3 setting value
other	-	RW		Keep

Note: PLL ouput = (clk_ref * div_loopc) / div_out.

The VCO frequency of the L1 PLL (the part in parentheses in the above formula) must be in the range 1.2GHz-3.2GHz. Should be It is also applicable to MEM PLL and HT PLL. The VCO frequency of the L2 PLL must be in the range 3.2GHz-6.4GHz.

Table 2- 14 Chip memory and HT clock software frequency multiplication setting register (physical address 0x1fe001c0)

Bit field	Field name	access	Reset value	description
0	SEL _MEM_PLL	RW	0x0	MEM clock non-software bypass entire PLL
1	SOFT_SET_MEM_PLL	RW	0x0	Allow software to set MEM PLL
2	BYPASS_MEM_PLL	RW	0x0	Bypass MEM_PLL
5: 3				
6	LOCKED_MEM_PLL	R	0x0	Whether MEM_PLL is locked
7	PD_MEM_PLL	RW	0x0	Turn off MEM PLL
13: 8	MEM_PLL_DIV_REFC	RW	0x1	MEM PLL input parameters When selecting NODE clock (NODE_CLOCK_SEL When 1), it is used as frequency division input
23:14	MEM_PLL_DIV_LOOPC	RW	0x41	MEM PLL input parameters
29:24	MEM_PLL_DIV_OUT	RW	0x0	MEM PLL input parameters
30	NODE_CLOCK_SEL	RW	0x0	0: Use MEM_PLL as the MEM clock 1: Use NODE_CLOCK as the crossover input
32	SEL_HT0_PLL	RW	0x0	HT0 non-software bypass PLL
33	SOFT_SET_HT0_PLL	RW	0x0	Allow software to set HT0 PLL
34	BYPASS_HT0_PLL	RW	0x0	Bypass HT0_PLL
35	LOCKEN_HT0_PLL	RW	0x0	Allow lock HT0 PLL
37:36	LOCKC_HT0_PLL	RW	0x0	Determine whether the HT0 PLL is locked with phase accuracy
38	LOCKED_HT0_PLL	R	0x0	Whether HT0_PLL is locked
45:40	HT0_DIV_HTCORE	RW	0x1	HT0 Core PLL input parameters
48	SEL_HT1_PLL	RW	0x0	HT1 non-software bypass PLL
49	SOFT_SET_HT1_PLL	RW	0x0	Allow software to set HT1 PLL
50	BYPASS_HT1_PLL	RW	0x0	Bypass HT1_PLL
51	LOCKEN_HT1_PLL	RW	0x0	Allow HT1 PLL to be locked
53:52	LOCKC_HT1_PLL	RW	0x0	Determine whether the HT1 PLL is locked with phase accuracy
54	LOCKED_HT1_PLL	R	0x0	Whether HT1_PLL is locked
61:56	HT1_DIV_HTCORE	RW	0x1	HT1 Core PLL input parameters
other		RW		Keep

Table 2- 15 Chip processor core software frequency division setting register (physical address 0x1fe001d0)

Bit field	Field name	access	Reset value	description
2: 0	core0_freqctrl	RW	0x7	Core 0 division control value
3	core0_en	RW	0x1	Core 0 clock enable
6: 4	core1_freqctrl	RW	0x7	Core 1 division control value

7	core1_en	RW	0x1	Core 1 clock enable
10: 8	core2_freqctrl	RW	0x7	Core 2 divider control value
11	core2_en	RW	0x1	Core 2 clock enable
14:12	core3_freqctrl	RW	0x7	Core 3 division control value
15	core3_en	RW	0x1	Core 3 clock enable
			Note:	The clock frequency value after the software frequency division is equal to the original Of (frequency division control value +1) / 8

3 GS464e processor core

GS464e is a four-launch 64-bit high-performance processor core. The processor core can be used as a single core for high-end embedded Applications and desktop applications can also be used as basic processor cores to form on-chip multi-core systems for server and high-performance applications use. Multiple GS464 cores in Loongson 3A3000 / 3B3000 and shared cache modules are formed via AXI interconnection network It is a multi-core structure with a distributed shared on-chip last-level cache. The main features of GS464 are as follows:

- MIPS64 compatible, support Godson extended instruction set;
- Four-shot superscalar structure, two fixed-point, two floating-point, and two memory access components;
- Each floating-point component supports full-pipe 64-bit / dual 32-bit floating-point multiply-add operations;
- The memory access component supports 128-bit memory access, the virtual address is 64 bits, and the physical address is 48 bits;
- Support register renaming, dynamic scheduling, branch prediction and other out-of-order execution technologies;
- 64 items are all connected, plus 8 groups connected to 1024 items, a total of 1088 items TLB, 64 items TLB, variable page size small;

- The size of the first-level instruction cache and data cache are 64KB, and the 4-way group is connected;
- Victim Cache is a private secondary cache with a size of 256KB and connected by 16 channels;
- Support Non-blocking access and Load-Speculation and other access optimization technologies;
- Support Cache consistency protocol, can be used for on-chip multi-core processor;
- Instruction Cache implements parity check, and Data Cache implements ECC check;
- Support the standard EJTAG debugging standard, which is convenient for hardware and software debugging;
- Standard 128-bit AXI interface.

The structure of GS464e is shown in the figure below. GS464e For more detailed introduction, please refer to GS464e user manual and MIPS64 User manual.

Figure 3- 1 GS464e structure diagram

4 Shared Cache (SCache)

The SCache module is a three-level cache shared by all processor cores within the Loongson 3A3000 / 3B3000 processor.

The main features of the SCache module include:

- Using 128-bit AXI interface.
- 16 items Cache access queue.
- Keywords first.
- Fastest 12 beats from receiving a read invalid request to returning data.
- Support Cache consistency protocol through the directory.
- It can be used for on-chip multi-core structure, and can also be directly connected with single processor IP.
- The 16-way group connection structure is adopted.
- Support ECC check.
- Support DMA consistent read and write and prefetch reading.
- Support 16 kinds of shared cache hashes.
- Support sharing cache by window lock.
- Ensure that read data returns atomicity.

Shared Cache module includes shared Cache management module scachemanage and shared Cache access module

scacheaccess. The Scachemanage module is responsible for processor access requests from the processor and DMA, and the shared cache

The TAG, directory and data are stored in the scacheaccess module. In order to reduce power consumption, Cache TAG,

The directory and data can be accessed separately. The shared Cache status bit and w bit are stored with the TAG, and the TAG is stored in the TAG RAM

In, the directory is stored in DIR RAM, and the data is stored in DATA RAM. Invalid request to access shared cache and read at the same time

Get out the TAGs and directories of all roads, and select the directories according to TAG, and read the data according to the hits. Replace request, re

The fill request and write back request only operate the TAG, directory and data along the way.

In order to improve the performance of some specific computing tasks, the shared cache adds a lock mechanism. Shares that fall in the locked area

The Cache block will be locked, so it will not be replaced by the shared Cache (unless the 16-way shared Cache is locked)

The four groups of lock window registers in the shared Cache module can be dynamically configured through the chip configuration register space, but

It must be ensured that one of the 16-way shared cache must not be locked. The size of each group of windows can be adjusted according to the mask,

However, it cannot exceed 3/4 of the size of the entire shared cache. In addition, when the shared cache receives the DMA write request, if it is written

The area is hit and locked in the shared cache, then the DMA write will be written directly to the shared cache instead of memory.

Table 4- 1 Shared Cache Lock Window Register Configuration

name	address	Bit field	description
Sl Yaock0_val d	0x3ff00200	[63:63]	Lock window 0 valid bits
Sl Yaock0_add	0x3ff00200	[47: 0]	No. 0 lock window lock address
Sl Yaock0_mask	0x3ff00240	[47: 0]	Lock window mask 0
Sl Yaock1_val d	0x3ff00208	[63:63]	Lock window 1 valid bit
Sl Yaock1_add	0x3ff00208	[47: 0]	Lock address of No. 1 lock window
Sl Yaock1_mask	0x3ff00248	[47: 0]	Lock window mask number 1
Sl Yaock2_val d	0x3ff00210	[63:63]	Lock window 2 valid bits
Sl Yaock2_add	0x3ff00210	[47: 0]	Lock address of No. 2 lock window
Sl Yaock2_mask	0x3ff00250	[47: 0]	Lock window mask number 2
Sl Yaock3_val d	0x3ff00218	[63:63]	Lock window 3 valid bits
Sl Yaock3_add	0x3ff00218	[47: 0]	Lock address of No. 3 lock window
Sl Yaock3_mask	0x3ff00258	[47: 0]	Lock window mask number 3

For example, when an address addr makes slock0_valid && ((addr & slock0_mask) == (slock0_addr & slock0_mask)) is 1, this address is locked by the lock window 0.

5 Matrix processing accelerator

Godson 3A3000 / 3B3000 has built-in two matrix processing accelerators independent of the processor core. Its basic functions are Through the configuration of the software, the function of transposing or moving the matrix stored in the memory from the source matrix to the target matrix can be realized. The two accelerators are integrated in the two HyperTransport controllers of Loongson 3A3000 / 3B3000, respectively. The crossbar realizes reading and writing of SCache and memory.

Since the order of elements in the same Cache line before transposition is scattered in the matrix after transposition, in order to improve read and write efficiency Multiple rows of data need to be read in, so that these data can be written in Cache row units in the transposed matrix, so A buffer area with a size of 32 lines is set in the module to achieve horizontal writing (reading from the source matrix to the buffer), Longitudinal readout (write from buffer to target matrix).

The working process of matrix processing is to first read in 32 rows of source matrix data, and then write the 32 rows of data into the target matrix, according to Go on again until the entire matrix is transposed or moved. The matrix processing accelerator can also only prefetch the source as needed The matrix is not written to the target matrix. In this way, the operation of prefetching the data into the SCache is realized.

The source matrix involved in transposing or moving may be a small matrix located in a large matrix. Therefore, the matrix address can be If it is not completely continuous, there will be gaps between the addresses between adjacent rows, and more programming control interfaces need to be implemented. Table 5-1 bel 5-4 shows the programming interface involved in matrix processing.

Table 5- 1 Matrix processing programming interface description

address	name	Attributes	Explanation
0x3ff00600	s ~ c_sta ~ t_add ~	RW	Source matrix start address
0x3ff00608	dst_sta t_add	RW	Target matrix start address
0x3ff00610	Yao w	RW	Number of elements in a row of the source matrix
0x3ff00618	c Yaol	RW	Number of elements in a column of the source matrix
0x3ff00620	length	RW	Row span of the large matrix where the source matrix is located (bytes)
0x3ff00628	w dth	RW	Row span of the large matrix where the target matrix is located (bytes)
0x3ff00630	t ans_ct l	RW	Transpose control register
0x3ff00638	t ans_status	RO	Transpose Status Register

Table 5- 2 Matrix processing register address description

address	name
0x3ff00600	S ~ c_sta ~ t_add ~ of transpose module 0
0x3ff00608	No. 0 transpose module's dst_sta ~ t_add ~
0x3ff00610	No Yao 0 of transpose module w
0x3ff00618	C Yaol of transpose module 0
0x3ff00620	Length of transposed module 0
0x3ff00628	W dth of transpose module 0
0x3ff00630	Transpose module No. 0 t ans_ct l
0x3ff00638	Transformer No. 0's t_ans_status
0x3ff00700	S ~ c_sta ~ t_add ~ of transpose module 1
0x3ff00708	Dst_sta t_add of transpose module 1

0x3ff00710	S c_ 耀 w_st de of transpose module 1
0x3ff00718	S c_last_ 耀 w_add of transpose module 1
0x3ff00720	The length of transpose module 1
0x3ff00728	W dth of transpose module 1
0x3ff00730	No. 1 transpose module's t ~ ans_ct ~ l
0x3ff00738	Transpose module No. 1's t_ans_status

Table 5- 3 t ans_ct l register description

Field	Explanation
0	Enable bit
1	Whether to write the target matrix. When it is 0, only the source matrix is prefetched, but the target matrix is not written.
2	After the source matrix is read, whether it is effectively interrupted.
3	After the target matrix is written, whether it is effectively interrupted,
7..4	A ~ cmd, read command internal control bit. When a cache is 4'hf, it must be set to 4'hc. It doesn't make sense when a cache is other value.
11..8	A Cache, read command internal control bit. When it is 4'hf, the cache path is used, and when it is 4'h0, the uncache path is used. other The value is meaningless.
15..12	Awcmd, write command internal control bit. When awcache is 4'hf, it must be set to 4'hb. Unintentional when awcache is other values Righteousness.
19..16	Awcache, write command internal control bit. When it is 4'hf, the cache path is used, and when it is 4'h0, the uncache path is used. other The value is meaningless.
21..20	Element size of matrix, 00 means 1 byte, 01 means 2 bytes, 10 means 4 bytes, 11 means 8 bytes
twenty two	t ans_yes, 1 means transpose; 0 means not transpose

Table 5-4 t ans_status register description

Field	Explanation
0	Source matrix read
1	The target matrix is written

6 Inter-processor interrupt and communication

Godson 3A3000 / 3B3000 implements 8 inter-core interrupt registers (IPI) for each processor core to support multiple core

Interrupt and communication between the processor cores during BIOS startup and operating system operation, the description and addresses are shown in Table 6-1 to Table 6-5.

Table 6-1: Inter-processor interrupt related registers and their functional description

name	Read and write permissions	description
IPI_Status	R	32-bit status register, if any bit is set and the corresponding bit is enabled, the The processor core INT4 interrupt line is set.
IPI_Enable	RW	32-bit enable register to control whether the corresponding interrupt bit is valid
IPI_Set	W	32 position register, write 1 to the corresponding bit, the corresponding STATUS register Bit is set
IPI_Clea	W	32-bit clear register, write 1 to the corresponding bit, the corresponding STATUS register Bit cleared 0
Ma lB Yao x0	RW	Cache register, used to transfer parameters at startup, according to 64 or 32 bit Uncache access.
Ma lB Yao x01	RW	Cache register, used to transfer parameters at startup, according to 64 or 32 bit Uncache access.
Ma lB Yao x02	RW	Cache register, used to transfer parameters at startup, according to 64 or 32 bit Uncache access.
Ma lB Yao x03	RW	Cache register, used to transfer parameters at startup, according to 64 or 32 bit Uncache access.

The interrupt-related registers and functions of Godson 3A3000 / 3B3000 and processor cores are described as follows:

Table 6- 2 Interrupt and Communication Register List of No. 0 Processor Core

name	address	Authority	description
C Yaocheng e0_IPI_Status	0x3ff01000	R	IPI_Status register of processor core 0
C Yaocheng e0_IPI_Enalbe	0x3ff01004	RW	IPI_Enalbe register of processor core 0
C Yaocheng e0_IPI_Set	0x3ff01008	W	IPI_Set register of processor core 0
C Yaocheng e0 _IPI_Clea	0x3ff0100c	W	IPI_Clea register of processor core 0
C Yao Cheng e0_Ma lB Yao x0	0x3ff01020	RW	IPI_Ma lB Yao x0 register of processor core 0
C Yao Yi e0_ Ma lB Yao x1	0x3ff01028	RW	IPI_Ma lB Yao x1 register of processor core 0
C Yao Cheng e0_ Ma lB Yao x2	0x3ff01030	RW	IPI_Ma lB Yao x2 register of processor core 0
C Yao Ya e0_ Ma lB Yao x3	0x3ff01038	RW	IPI_Ma lB Yao x3 register of processor core 0

Table 6-3 Internuclear Interrupt and Communication Register List of No. 1 Processor Core

name	address	Authority	description
C Yaoyi e1_IPI_Status	0x3ff01100	R	IPI_Status register of processor core 1
C Yaocheng e1_IPI_Enalbe	0x3ff01104	RW	IPI_Enalbe register of processor core 1
C Yaoyi e1_IPI_Set	0x3ff01108	W	IPI_Set register of processor core 1
C Yaocheng e1 _IPI_Clea	0x3ff0110c	W	IPI_Clea register of processor core 1
C Yao Ya e1_Ma lB Yao x0	0x3ff01120	R	IPI_Ma lB Yao x0 register of processor core 1
C Yao Yi e1_ Ma lB Yao x1	0x3ff01128	RW	IPI_Ma lB Yao x1 register of processor core 1
C Yao Yi e1_ Ma lB Yao x2	0x3ff01130	W	IPI_Ma lB Yao x2 register of processor core 1
C Yao Yi e1_ Ma lB Yao x3	0x3ff01138	W	IPI_Ma lB Yao x3 register of processor core 1

Table 6-4 Internuclear Interrupt and Communication Register List of No. 2 Processor Core

name	address	Authority	description
C Yaocheng e2_IPI_Status	0x3ff01200	R	IPI_Status register of processor core 2
C Yaocheng e2_IPI_Enalbe	0x3ff01204	RW	IPI_Enalbe register of processor core 2
C Yaoyi e2_IPI_Set	0x3ff01208	W	IPI_Set register of processor core 2
C Yaocheng e2 _IPI_Clea	0x3ff0120c	W	IPI_Clea register of processor core 2
C Yao Cheng e2_Ma lB Yao x0	0x3ff01220	R	IPI_Ma lB Yao x0 register of processor core 2
C Yao Ya e2_ Ma lB Yao x1	0x3ff01228	RW	IPI_Ma lB Yao x1 register of processor core 2
C Yao Ya e2_ Ma lB Yao x2	0x3ff01230	W	IPI_Ma lB Yao x2 register of processor core 2
C Yao Yi e2_ Ma lB Yao x3	0x3ff01238	W	IPI_Ma lB Yao x3 register of processor core 2

Table 6-5 List of Internuclear Interrupts and Communication Registers of Processor Core

name	address	Authority	description
C Yaocheng e3_IPI_Status	0x3ff01300	R	IPI_Status register of processor core 3
C Yaocheng e3_IPI_Enalbe	0x3ff01304	RW	IPI_Enalbe register of processor core 3
C Yaoyae3_IPI_Set	0x3ff01308	W	IPI_Set register of processor core 3
C Yaocheng e3 _IPI_Clea	0x3ff0130c	W	IPI_Clea register of processor core 3
C Yao Cheng e3_Ma lB Yao x0	0x3ff01320	R	IPI_Ma lB Yao x0 register of processor core 3
C Yao Ya e3_ Ma lB Yao x1	0x3ff01328	RW	IPI_Ma lB Yao x1 register of processor core 3
C Yao Ya e3_ Ma lB Yao x2	0x3ff01330	W	IPI_Ma lB Yao x2 register of processor core 3
C Yao Yi e3_ Ma lB Yao x3	0x3ff01338	W	IPI_Ma lB Yao x3 register of processor core 3

Listed above is the inter-core of a single-node multi-processor system composed of a single Loongson 3A3000 / 3B3000 chip

Break the relevant register list. When using multiple Loongson 3A3000 / 3B3000 interconnections to form a multi-node CC-NUMA system, each

The node in the chip corresponds to a system global node number, and the IPI register address of the processor core in the node is based on

The base address of the node is in a fixed offset relationship. For example, the IPI_Status address of processor core 0 in node 0 is

0x3ff01000, and the address of processor 0 of node 1 is 0x10003ff01000, and so on.

7 I / O interrupt

Loongson 3A3000 / 3B3000 chip supports up to 32 interrupt sources and is managed in a unified manner, as shown in Figure 7-1 below

Shows that any IO interrupt source can be configured to enable, trigger, and the target processor core to be routed

Interrupt the foot.

Figure 7- 1 Schematic diagram of Loongson 3A3000 / 3B3000 processor interrupt routing

Interrupt related configuration registers are used to control the corresponding interrupt lines in the form of bits.

See Table 7-1 for sexual configuration. The interrupt enable (Enable) configuration has three registers: Intenset, Intenclr and Inten.

Intenset sets the interrupt enable, and the interrupt corresponding to the bit written to 1 in the Intenset register is enabled. Intenclr clear interrupt

When enabled, the interrupt corresponding to the bit written to 1 in the Intenclr register is cleared. The Inten register reads the current status of each interrupt enable

condition. The interrupt signal in the form of a pulse (such as PCI_SERR) is selected by the Intedge configuration register. Writing 1 means that the pulse triggers

Send, write 0 means level trigger. The interrupt handler can clear the pulse record through the corresponding bit of Intenclr.

Table 7- 1 Interrupt Control Register

Bit field	Access properties / default				
	Intedge	Inten	Intenset	Intencl	Interrupt source
3: 0	RW / 0	R / 0	W / 0	W / 0	Sys_ nt0-3
7: 4	RO / 0	R / 0	RW / 0	RW / 0	PCI_INTn
8	RO / 0	R / 0	RW / 0	RW / 0	Mat ~ x_ nt0
9	RO / 1	R / 0	RW / 0	RW / 0	Mat x_ nt1
10	RO / 1	R / 0	RW / 0	RW / 0	Lpc
12: 11	RW / 0	Keep	Keep	Keep	Mc0-1
13	RW / 0	R / 0	RW / 0	RW / 0	Ba e
14	RW / 0	R / 0	RW / 0	RW / 0	Thsens nt
15	RW / 0	R / 0	RW / 0	RW / 0	Pc _pe
23: 16	RW / 0	R / 0	RW / 0	RW / 0	HT0 nt0-7
31: 24	RW / 0	R / 0	RW / 0	RW / 0	HT1 nt0-7

Table 7- 2 IO Control Register Address

name	Address offset	description
Int s	0x3ff01420	32-bit interrupt status register
Inten	0x3ff01424	32-bit interrupt enable status register

Intenset	0x3ff01428	32-bit setting enable register
Intencl	0x3ff0142c	32-bit clear enable register
Intedge	0x3ff01438	32-bit trigger mode register
CORE0_INTISR	0x3ff01440	32-bit interrupt status routed to CORE0
CORE1_INTISR	0x3ff01448	32-bit interrupt status routed to CORE1
CORE2_INTISR	0x3ff01450	32-bit interrupt status routed to CORE2
CORE3_INTISR	0x3ff01458	32-bit interrupt status routed to CORE3

Four processor cores are integrated in Loongson 3A3000 / 3B3000, the above 32-bit interrupt sources can be configured by software

Select the target processor core to be interrupted. Further, the interrupt source can be optionally routed to the processor core interrupt INT0 to INT3

Any one of the IP2 to IP5 corresponding to CP0_Status. Each of the 32 I / O interrupt sources corresponds to one 8

The bit routing controller, its format and address are shown in Table 7-3 and 7-4 below. The routing register is routed in a vector way

By choice, such as 0x48 indicates routing to INT2 of processor 3.

Page 46

Table 7- 3 Interrupt Routing Register Description

Bit field	Explanation
3: 0	Routed processor core vector number
7: 4	Routed processor core interrupt pin vector number

Table 7-4 Interrupt Routing Register Address

name	Address offset	description	name	Address offset	description
Ent y0	0x3ff01400	Sys_ nt0	Ent y16	0x3ff01410	HT0- nt0
Ent y1	0x3ff01401	Sys_ nt1	Ent y17	0x3ff01411	HT0- nt1
Ent2	0x3ff01402	Sys_ nt2	Ent y18	0x3ff01412	HT0- nt2
Ent y3	0x3ff01403	Sys_ nt3	Ent y19	0x3ff01413	HT0- nt3
Ent y4	0x3ff01404	Pc _ nt0	Ent y20	0x3ff01414	HT0- nt4
Ent y5	0x3ff01405	Pc _ nt1	Ent y21	0x3ff01415	HT0- nt5
Ent y6	0x3ff01406	Pc _ nt2	Ent 信 y22	0x3ff01416	HT0- nt6
Ent y7	0x3ff01407	Pc _ nt3	Ent y23	0x3ff01417	HT0- nt7
Ent y8	0x3ff01408	Mat x nt0	Ent y24	0x3ff01418	HT1- nt0
Ent y9	0x3ff01409	Mat x nt1	Ent y25	0x3ff01419	HT1- nt1
Ent y10	0x3ff0140a	Lpc nt	Ent y26	0x3ff0141a	HT1- nt2
Ent y11	0x3ff0140b	Mc0	Ent y27	0x3ff0141b	HT1- nt3
Ent y12	0x3ff0140c	Mc1	Ent y28	0x3ff0141c	HT1- nt4
Ent y13	0x3ff0140d	Ba e	Ent y29	0x3ff0141d	HT1- nt5
Ent y14	0x3ff0140e	Thsens nt	Ent y30	0x3ff0141e	HT1- nt6
Ent y15	0x3ff0140f	Pc _pe / se	Ent y31	0x3ff0141f	HT1- nt7

8 Temperature sensor

8.1 Real-time temperature sampling

Loongson 3A3000 / 3B3000 internally integrates two temperature sensors, which can be registered by sampling starting at 0x1FE00198

The device can be used for observation, and at the same time, it can be controlled by flexible high and low temperature interruption alarm or automatic frequency modulation functic

The corresponding bits of the sampling register are as follows (the base address is 0x1FE00198):

Table 8-1 Temperature sampling register description

Bit field	Field name	access	Reset value	description
twenty four	Thsens0_overflow	R		Temperature sensor 0 overflow (over 125 ℃)
25	Thsens1_overflow	R		Temperature sensor 1 overflow (over 125 ℃)
47:32	Thsens0_out	R		Temperature sensor 0 Celsius Knot point temperature degree= Thens0_out * 731 / 0x4000-273 Temperature range -40 degrees – 125 degrees
65:48	Thsens1_out	R		Temperature sensor 1 Celsius Knot point temperature degree= Thens1_out -* 731 / 0x4000-273 Temperature range -40 degrees – 125 degrees

Through the setting of the control register, it is possible to achieve interruptions above the preset temperature, interruptions below the preset temperature and high temperature

Automatic frequency reduction function.

8.2 High and low temperature interrupt trigger

For the high and low temperature interrupt alarm function, there are 4 groups of control registers to set their thresholds. Each set of register packets

Contains the following three control bits:

GATE: Set the threshold for high or low temperature. When the input temperature is higher than the high temperature threshold or lower than the low temperature threshold, i

Interruption

EN: interrupt enable control. The setting of this group of registers is valid after being set to 1;

SEL: Input temperature selection. At present, 3A3000 / 3B3000 integrate two temperature sensors inside, this register is used to configure

The temperature of which sensor is selected as input. You can use 0 or 1.

The high temperature interrupt control register contains 4 sets of setting bits for controlling high temperature interrupt trigger;

The device contains 4 sets of setting bits for controlling low temperature interrupt trigger. There is also a set of registers used to display the interrupt status, divided

Do not correspond to high temperature interrupt and low temperature interrupt, any write operation to this register will clear the interrupt status.

The specific descriptions of these registers are as follows:

Table 8- 2 High and low temperature interrupt register description

register	address	control	Explanation
			[7: 0]: Hi_gate0: high temperature threshold 0, an interrupt will be generated if this temperature is exceeded
			[8: 8]: Hi_en0: High temperature interrupt enable 0
			[11:10]: Hi_Sel0: Select the temperature sensor input source of high temperature interrupt
			[23:16]: Hi_gate1: high temperature threshold 1, exceeding this temperature will generate an interrupt
			[24:24]: Hi_en1: High temperature interrupt enable 1
			[27:26]: Hi_Sel1: Select the temperature sensor input source for high temperature interrupt 1
			[39:32]: Hi_gate2: High temperature threshold 2, above this temperature will generate an interrupt
			[40:40]: Hi_en2: High temperature interrupt enable 2
			[43:42]: Hi_Sel2: Select the temperature sensor input source for high temperature interrupt 2
			[55:48]: Hi_gate3: High temperature threshold 3, exceeding this temperature will generate interrupt
High temperature interrupt control register			[56:56]: Hi_en3: High temperature interrupt enable 3
Thsens_int_ctrl_Hi	0x3ff01460	RW	[59:58]: Hi_Sel3: Select the temperature sensor input source for high temperature interrupt 3
			[7: 0]: Lo_gate0: low temperature threshold 0, below this temperature will generate an interrupt
			[8: 8]: Lo_en0: Low temperature interrupt enable 0
			[11:10]: Lo_Sel0: Select the temperature sensor input source for low temperature interrupt 0
			[23:16]: Lo_gate1: low temperature threshold 1, below this temperature will generate an interrupt
			[24:24]: Lo_en1: Low temperature interrupt enable 1
			[27:26]: Lo_Sel1: Select the temperature sensor input source for low temperature interrupt 1
			[39:32]: Lo_gate2: Low temperature threshold 2, below this temperature will generate an interrupt
			[40:40]: Lo_en2: Low temperature interrupt enable 2
			[43:42]: Lo_Sel2: Select the temperature sensor input source for low temperature interrupt 2
			[55:48]: Lo_gate3: Low temperature threshold 3, below this temperature will generate an interrupt
Low temperature interrupt control register			[56:56]: Lo_en3: Low temperature interrupt enable 3
Thsens_int_ctrl_Lo	0x3ff01468	RW	[59:58]: Lo_Sel3: Select temperature sensor input source for low temperature interrupt 3
			Interrupt status register, write any value to clear the interrupt
Interrupt status register			[0]: High temperature interrupt trigger
Thsens_int_status / clr 0x3ff01470		RW	[1]: Low temperature interrupt trigger

8.3 High temperature automatic frequency reduction setting

In order to ensure the operation of the chip in a high-temperature environment, you can set the high-frequency automatic frequency reduction, so that the chip exceeds

In the range, it actively divides the clock to achieve the effect of reducing the chip turnover rate.

For the high temperature frequency reduction function, there are 4 sets of control registers to set its behavior. Each set of registers contains the following four

Control bit:

GATE: Set the threshold for high or low temperature. When the input temperature is higher than the high temperature threshold or lower than the low temperature threshold, i

Frequency division operation;

EN: interrupt enable control. The setting of this group of registers is valid after being set to 1;

SEL: Input temperature selection. At present, 3A3000 / 3B3000 integrate two temperature sensors inside, this register is used to configure

The temperature of which sensor is selected as input. You can use 0 or 1.

FREQ: frequency division number. When the frequency division operation is triggered, the frequency is adjusted to FREQ / 8 times the current clock frequency.

Table 8- 3 High-temperature down-frequency control register description

register	address	control	Explanation
			Four sets of setting priority from high to low
			[7: 0]: Scale_gate0: High temperature threshold 0, frequency will be reduced if this temperature is exceeded

High temperature down frequency control register Thsens_freq_scale	0x3ff01480	RW	[8: 8]: Scale_en0: High temperature frequency reduction enable 0 [11:10]: Scale_Sel0: Select the temperature sensor input source of high temperature down-conversion 0 [14:12]: Scale_freq0: frequency division value when frequency is reduced [23:16]: Scale_gate1: High temperature threshold 1, exceeding this temperature will reduce the frequency [24:24]: Scale_en1: High temperature frequency reduction enable 1 [27:26]: Scale_Sel1: Select the temperature sensor input source for high temperature down-conversion 1 [30:28]: Scale_freq1: frequency division value when frequency is reduced [39:32]: Scale_gate2: High temperature threshold value 2, if this temperature is exceeded, frequency will be reduced [40:40]: Scale_en2: High temperature frequency reduction enable 2 [43:42]: Scale_Sel2: Select the temperature sensor input source for high temperature down-conversion 2 [46:44]: Scale_freq2: frequency division value when frequency is reduced [55:48]: Scale_gate3: High temperature threshold 3, over this temperature will reduce the frequency [56:56]: Scale_en3: High temperature frequency reduction enable 3 [59:58]: Scale_Sel3: Select the temperature sensor input source for high temperature down-conversion 3 [62:60]: Scale_freq3: Frequency division value when frequency is reduced

9 DDR2 / 3 SDRAM controller configuration

The design of the integrated memory controller inside Loongson No. 3 processor complies with the industry standard of DDR2 / 3 SDRAM (JESD79-2 And JESD79-3). In the Godson 3 processor, all memory read / write operations are implemented in compliance with JESD79-2B and The provisions of JESD79-3.

9.1 DDR2 / 3 SDRAM controller function overview

Loongson No. 3 processor supports a maximum of 4 CS (implemented by 4 DDR2 SDRAM chip select signals, that is, two double-sided memory Article), contains a total of 19-bit address bus (ie: 16-bit row and column address bus and 3-bit logical Bank bus).

When Loongson No. 3 processor chooses to use different memory chip types, it can adjust the DDR2 / 3 controller parameter settings To support. Among them, the maximum number of chip selects (CS_n) supported is 4, the number of row addresses (RAS_n) is 16, and the column addresses (CAS_n) The number is 15, and the number of logical volume selection (BANK_n) is 3.

The physical address of the memory request sent by the CPU can be mapped to many different addresses according to different configurations inside the controller Shoot.

The memory control circuit integrated in the Loongson 3 processor only accepts memory read / write requests from the processor or external devices Demand, in all memory read / write operations, the memory control circuit is in the slave state.

The memory controller in Loongson No. 3 processor has the following characteristics:

- Full pipeline operation of commands and read and write data on the interface

- Memory command merging and sorting improve overall bandwidth
- Configure register read and write ports, you can modify the basic parameters of the memory device
- Built-in dynamic delay compensation circuit (DCC) for reliable transmission and reception of data
- The ECC function can detect 1-bit and 2-bit errors on the data path, and can perform self-correction on 1-bit errors.

  Error correction
- Support 133-667MHZ working frequency

9.2 DDR2 / 3 SDRAM read operation protocol

The protocol of DDR2 / 3 SDRAM read operation is shown in Figure 11-2. In the figure, the command (Command, referred to as CMD) is composed of RAS_n, CAS_n and WE_n are composed of three signals. For read operations, RAS_n = 1, CAS_n = 0, and WE_n = 1.

Figure 9- 1 DDR2 SDRAM read operation protocol

In the figure above, Cas Latency (CL) = 3, Read Latency (RL) = 3, and Burst Length = 8.

9.3 DDR2 / 3 SDRAM write operation protocol

The protocol of DDR2 / 3 SDRAM write operation is shown in Figure 11-3. The command CMD in the figure is composed of RAS_n, CAS_n and WE_n, It consists of three signals. For write operations, RAS_n = 1, CAS_n = 0, and WE_n = 0. In addition, unlike read operations, write The operation requires DQM to identify the mask of the write operation, that is, the number of bytes to be written. DQM is synchronized with the DQs signal in the figure.

Figure 9- 2 DDR2 SDRAM write operation protocol

In the above picture, Cas Latency (CL) = 3, Write Latency (WL) = Read Latency (RL) – 1 = 2, Burst Length = 4.

9.4 DDR2 / 3 SDRAM parameter configuration format

The parameter list and description of the memory controller software are as follows:

	63:56	55:48	47:40	39:32	31:24	23:16	15: 8	7: 0
0x000	dll_close_disable dll_sync_disable	Dll_adj_cnt	Dll_value_ck (RD)		Dll_init_done (RD)		Version (RD)	
0x008								
0x010								
0x018	Dll_ck_3	Dll_ck_2	Dll_ck_1	Dll_ck_0	Dll_increment	Dll_start_point	Dll_bypass	Init_start
0x020	Dq_oe_end_0	Dq_oe_begin_0	Dq_stop_edge_0	Dq_start_edge_0	Rddata_delay_0	Rddqs_lt_half_0	Wrdqs_lt_half_0	Wrdq_lt_half_0
0x028	Rd_oe_end_0	Rd_oe_begin_0	Rd_stop_edge_0	Rd_start_edge_0	Dqs_oe_end_0	Dqs_oe_begin_0	Dqs_stop_edge_0	Dqs_start_edge_0
0x030	Enzi_end_0	Enzi_begin_0	Wrclk_sel_0	Wrdq_clkdelay_0	Odt_oe_end_0	Odt_oe_begin_0	Odt_stop_edge_0	Odt_start_edge_0
0x038	Enzi_stop_0	Enzi_start_0	Dll_oe_shorten_0	Dll_rddqs_n_0	Dll_rddqs_p_0	Dll_wrdqs_0	Dll_wrdata_0	Dll_gate_0
0x040	Dq_oe_end_1	Dq_oe_begin_1	Dq_stop_edge_1	Dq_start_edge_1	Rddata_delay_1	Rddqs_lt_half_1	Wrdqs_lt_half_1	Wrdq_lt_half_1
0x048	Rd_oe_end_1	Rd_oe_begin_1	Rd_stop_edge_1	Rd_start_edge_1	Dqs_oe_end_1	Dqs_oe_begin_1	Dqs_stop_edge_1	Dqs_start_edge_1
0x050	Enzi_end_1	Enzi_begin_1	Wrclk_sel_1	Wrdq_clkdelay_1	Odt_oe_end_1	Odt_oe_begin_1	Odt_stop_edge_1	Odt_start_edge_1
0x058	Enzi_stop_1	Enzi_start_1	Dll_oe_shorten_1	Dll_rddqs_n_1	Dll_rddqs_p_1	Dll_wrdqs_1	Dll_wrdata_1	Dll_gate_1
0x060	Dq_oe_end_2	Dq_oe_begin_2	Dq_stop_edge_2	Dq_start_edge_2	Rddata_delay_2	Rddqs_lt_half_2	Wrdqs_lt_half_2	Wrdq_lt_half_2
0x068	Rd_oe_end_2	Rd_oe_begin_2	Rd_stop_edge_2	Rd_start_edge_2	Dqs_oe_end_2	Dqs_oe_begin_2	Dqs_stop_edge_2	Dqs_start_edge_2
0x070	Enzi_end_2	Enzi_begin_2	Wrclk_sel_2	Wrdq_clkdelay_2	Odt_oe_end_2	Odt_oe_begin_2	Odt_stop_edge_2	Odt_start_edge_2
0x078	Enzi_stop_2	Enzi_start_2	Dll_oe_shorten_2	Dll_rddqs_n_2	Dll_rddqs_p_2	Dll_wrdqs_2	Dll_wrdata_2	Dll_gate_2
0x080	Dq_oe_end_3	Dq_oe_begin_3	Dq_stop_edge_3	Dq_start_edge_3	Rddata_delay_3	Rddqs_lt_half_3	Wrdqs_lt_half_3	Wrdq_lt_half_3
0x088	Rd_oe_end_3	Rd_oe_begin_3	Rd_stop_edge_3	Rd_start_edge_3	Dqs_oe_end_3	Dqs_oe_begin_3	Dqs_stop_edge_3	Dqs_start_edge_3
0x090	Enzi_end_3	Enzi_begin_3	Wrclk_sel_3	Wrdq_clkdelay_3	Odt_oe_end_3	Odt_oe_begin_3	Odt_stop_edge_3	Odt_start_edge_3
0x098	Enzi_stop_3	Enzi_start_3	Dll_oe_shorten_3	Dll_rddqs_n_3	Dll_rddqs_p_3	Dll_wrdqs_3	Dll_wrdata_3	Dll_gate_3
0x0A0	Dq_oe_end_4	Dq_oe_begin_4	Dq_stop_edge_4	Dq_start_edge_4	Rddata_delay_4	Rddqs_lt_half_4	Wrdqs_lt_half_4	Wrdq_lt_half_4
0x0A8	Rd_oe_end_4	Rd_oe_begin_4	Rd_stop_edge_4	Rd_start_edge_4	Dqs_oe_end_4	Dqs_oe_begin_4	Dqs_stop_edge_4	Dqs_start_edge_4
0x0B0	Enzi_end_4	Enzi_begin_4	Wrclk_sel_4	Wrdq_clkdelay_4	Odt_oe_end_4	Odt_oe_begin_4	Odt_stop_edge_4	Odt_start_edge_4
0x0B8	Enzi_stop_4	Enzi_start_4	Dll_oe_shorten_4	Dll_rddqs_n_4	Dll_rddqs_p_4	Dll_wrdqs_4	Dll_wrdata_4	Dll_gate_4
0x0C0	Dq_oe_end_5	Dq_oe_begin_5	Dq_stop_edge_5	Dq_start_edge_5	Rddata_delay_5	Rddqs_lt_half_5	Wrdqs_lt_half_5	Wrdq_lt_half_5
0x0C8	Rd_oe_end_5	Rd_oe_begin_5	Rd_stop_edge_5	Rd_start_edge_5	Dqs_oe_end_5	Dqs_oe_begin_5	Dqs_stop_edge_5	Dqs_start_edge_5
0x0D0	Enzi_end_5	Enzi_begin_5	Wrclk_sel_5	Wrdq_clkdelay_5	Odt_oe_end_5	Odt_oe_begin_5	Odt_stop_edge_5	Odt_start_edge_5
0x0D8	Enzi_stop_5	Enzi_start_5	Dll_oe_shorten_5	Dll_rddqs_n_5	Dll_rddqs_p_5	Dll_wrdqs_5	Dll_wrdata_5	Dll_gate_5
0x0E0	Dq_oe_end_6	Dq_oe_begin_6	Dq_stop_edge_6	Dq_start_edge_6	Rddata_delay_6	Rddqs_lt_half_6	Wrdqs_lt_half_6	Wrdq_lt_half_6
0x0E8	Rd_oe_end_6	Rd_oe_begin_6	Rd_stop_edge_6	Rd_start_edge_6	Dqs_oe_end_6	Dqs_oe_begin_6	Dqs_stop_edge_6	Dqs_start_edge_6
0x0F0	Enzi_end_6	Enzi_begin_6	Wrclk_sel_6	Wrdq_clkdelay_6	Odt_oe_end_6	Odt_oe_begin_6	Odt_stop_edge_6	Odt_start_edge_6
0x0F8	Enzi_stop_6	Enzi_start_6	Dll_oe_shorten_6	Dll_rddqs_n_6	Dll_rddqs_p_6	Dll_wrdqs_6	Dll_wrdata_6	Dll_gate_6
0x100	Dq_oe_end_7	Dq_oe_begin_7	Dq_stop_edge_7	Dq_start_edge_7	Rddata_delay_7	Rddqs_lt_half_7	Wrdqs_lt_half_7	Wrdq_lt_half_7
0x108	Rd_oe_end_7	Rd_oe_begin_7	Rd_stop_edge_7	Rd_start_edge_7	Dqs_oe_end_7	Dqs_oe_begin_7	Dqs_stop_edge_7	Dqs_start_edge_7
0x110	Enzi_end_7	Enzi_begin_7	Wrclk_sel_7	Wrdq_clkdelay_7	Odt_oe_end_7	Odt_oe_begin_7	Odt_stop_edge_7	Odt_start_edge_7
0x118	Enzi_stop_7	Enzi_start_7	Dll_oe_shorten_7	Dll_rddqs_n_7	Dll_rddqs_p_7	Dll_wrdqs_7	Dll_wrdata_7	Dll_gate_7
0x120	Dq_oe_end_8	Dq_oe_begin_8	Dq_stop_edge_8	Dq_start_edge_8	Rddata_delay_8	Rddqs_lt_half_8	Wrdqs_lt_half_8	Wrdq_lt_half_8
0x128	Rd_oe_end_8	Rd_oe_begin_8	Rd_stop_edge_8	Rd_start_edge_8	Dqs_oe_end_8	Dqs_oe_begin_8	Dqs_stop_edge_8	Dqs_start_edge_8

0x130	Enzi_end_8	Enzi_begin_8	Wrclk_sel_8	Wrdq_clkdelay_8	Odt_oe_end_8	Odt_oe_begin_8	Odt_stop_edge_8	Odt_start_edge_8
0x138	Enzi_stop_8	Enzi_start_8	Dll_oe_shorten_8	Dll_rddqs_n_8	Dll_rddqs_p_8	Dll_wrdqs_8	Dll_wrdata_8	Dll_gate_8
0x140	Pad_ocd_clk	Pad_ocd_ctl	Pad_ocd_dqs	Pad_ocd_dq	Pad_enzi		Pad_en_ctl	Pad_en_clk
0x148	Pad_adj_code_dqs	Pad_code_dqs	Pad_adj_code_dq	Pad_code_dq		Pad_vref_internal	Pad_odt_se	Pad_modezi1v8
0x150		Pad_reset_po	Pad_adj_code_clk	Pad_code_lk	Pad_adj_code_cmd	Pad_code_cmd	Pad_adj_code_addr	Pad_code_addr
0x158		Pad_comp_code_o	Pad_comp_okn		Pad_comp_code_i	Pad_comp_mode	Pad_comp_tm	Pad_comp_pd
0x160	Rdfifo_empty (RD)		Overflow (RD)		Dram_init (RD)	Rdfifo_valid	Cmd_timming	Ddr3_mode
0x168	Ba_xor_row_offset	Addr_mirror	Cmd_delay	Burst_length	Bank / Cs_resync	Cs_zq	Cs_mrs	Cs_enable
0x170	Odt_wr_cs_map		Odt_wr_length	Odt_wr_delay	Odt_rd_cs_map		Odt_rd_length	Odt_rd_delay
0x178								
0x180	Lvl_resp_0 (RD)	Lvl_done (RD)	Lvl_ready (RD)		Lvl_cs	tLVL_DELAY	Lvl_req (WR)	Lvl_mode
0x188	Lvl_resp_8 (RD)	Lvl_resp_7 (RD)	Lvl_resp_6 (RD)	Lvl_resp_5 (RD)	Lvl_resp_4 (RD)	Lvl_resp_3 (RD)	Lvl_resp_2 (RD)	Lvl_resp_1 (RD)
0x190	Cmd_a		Cmd_ba	Cmd_cmd	Cmd_cs	Status_cmd (RD)	Cmd_req (WR)	Command_mode
0x198			Status_sref (RD)	Srefresh_req	Pre_all_done (RD)	Pre_all_req (RD)	Mrs_done (RD)	Mrs_req (WR)
0x1A0	Mr_3_cs_0		Mr_2_cs_0		Mr_1_cs_0		Mr_0_cs_0	
0x1A8	Mr_3_cs_1		Mr_2_cs_1		Mr_1_cs_1		Mr_0_cs_1	
0x1B0	Mr_3_cs_2		Mr_2_cs_2		Mr_1_cs_2		Mr_0_cs_2	
0x1B8	Mr_3_cs_3		Mr_2_cs_3		Mr_1_cs_3		Mr_0_cs_3	
0x1C0	tRESET	tCKE	tXPR	tMOD	tZQCL	tZQ_CMD	tWLDQSEN	tRDDATA
0x1C8	tFAW	tRRD	tRCD	tRP	tREF	tRFC	tZQCS	tZQperiod
0x1D0	tODTL	tXSRD	tPHY_RDLAT	tPHY_WRLAT	tRAS_max			tRAS_min
0x1D8	tXPDLL	tXP	tWR	tRTP	tRL	tWL	tCCD	tWTR
0x1E0	tW2R_diffCS	tW2W_diffCS	tR2P_sameBA	tW2P_sameBA	tR2R_sameBA	tR2W_sameBA	tW2R_sameBA	tW2W_sameBA
0x1E8	tR2R_diffCS	tR2W_diffCS	tR2P_sameCS	tW2P_sameCS	tR2R_sameCS	tR2W_sameCS	tW2R_sameCS	tW2W_sameCS
0x1F0	Power_up	Age_step	tCPDED	Cs_map	Bs_config	Nc	Pr_r2w	Placement_en
0x1F8	Hw_pd_3	Hw_pd_2	Hw_pd_1	Hw_pd_0	Credit_16	Credit_32	Credit_64	Selection_en
0x200	Cmdq_age_16		Cmdq_age_32		Cmdq_age_64		tCKESR	tRDPDEN
0x208	Wfifo_age		Rfifo_age		Power_stat3	Power_stat2	Power_stat1	Power_stat0
0x210	Active_age		Cs_place_0	Addr_win_0	Cs_diff_0	Row_diff_0	Ba_diff_0	Col_diff_0
0x218	Fastpd_age		Cs_place_1	Addr_win_1	Cs_diff_1	Row_diff_1	Ba_diff_1	Col_diff_1
0x220	Slowpd_age		Cs_place_2	Addr_win_2	Cs_diff_2	Row_diff_2	Ba_diff_2	Col_diff_2
0x228	Selfref_age		Cs_place_3	Addr_win_3	Cs_diff_3	Row_diff_3	Ba_diff_3	Col_diff_3
0x230	Win_mask_0				Win_base_0			
0x238	Win_mask_1				Win_base_1			
0x240	Win_mask_2				Win_base_2			
0x248	Win_mask_3				Win_base_3			
0x250		Cmd_monitor	Axi_monitor		Ecc_code (RD)	Ecc_enable	Int_vector	Int_enable
0x258								
0x260	Ecc_addr (RD)							
0x268	Ecc_data (RD)							

0x270	Lpbk_ecc_mask (RD)	Prbs_init			Lpbk_error (RD)	Prbs_23	Lpbk_start	Lpbk_en
0x278	Lpbk_ecc (RD)		Lpbk_data_mask (RD)		Lpbk_correct (RD)		Lpbk_counter (RD)	
0x280	Lpbk_data_r (RD)							
0x288	Lpbk_data_f (RD)							
0x290	Axi0_bandwidth_w				Axi0_bandwidth_r			
0x298	Axi0_latency_w				Axi0_latency_r			
0x2A0	Axi1_bandwidth_w				Axi1_bandwidth_r			
0x2A8	Axi1_latency_w				Axi1_latency_r			
0x2B0	Axi2_bandwidth_w				Axi2_bandwidth_r			
0x2B8	Axi2_latency_w				Axi2_latency_r			

0x2C0	Axi3_bandwidth_w				Axi3_bandwidth_r			
0x2C8	Axi3_latency_w				Axi3_latency_r			
0x2D0	Axi4_bandwidth_w				Axi4_bandwidth_r			
0x2D8	Axi4_latency_w				Axi4_latency_r			
0x2E0	Cmdq0_bandwidth_w				Cmdq0_bandwidth_r			
0x2E8	Cmdq0_latency_w				Cmdq0_latency_r			
0x2F0	Cmdq1_bandwidth_w				Cmdq1_bandwidth_r			
0x2F8	Cmdq1_latency_w				Cmdq1_latency_r			
0x300	Cmdq2_bandwidth_w				Cmdq2_bandwidth_r			
0x308	Cmdq2_latency_w				Cmdq2_latency_r			
0x310	Cmdq3_bandwidth_w				Cmdq3_bandwidth_r			
0x318	Cmdq3_latency_w				Cmdq3_latency_r			
0x320	tRESYNC_length	tRESYNC_shift	tRESYNC_max	tRESYNC_min	Pre_predict		tXS	tREF_low
0x328								tRESYNC_delay
0x330	Stat_en	Rdbuffer_max	Retry	Wr_pkg_num	Rwq_rb	Stb_en	Addr_new	tRDQidle
0x338				Rd_fifo_depth	Retry_cnt			
0x340	tREFretention					Ref_num	tREF_IDLE	Ref_sch_en
0x348								
0x350	Lpbk_data_en							
0x358						Lpbk_ecc_mask_en	Lpbk_ecc_en	Lpbk_data_mask_en
0x360			Int_ecc_cnt_fatal	Int_ecc_cnt_error	Ecc_cnt_cs_3	Ecc_cnt_cs_2	Ecc_cnt_cs_1	Ecc_cnt_cs_0
0x368								
0x370	Prior_age3		Prior_age2		Prior_age1		Prior_age_0	
0x378								Row_hit_place
0x380	Zq_cnt_1				Zq_cnt_0			
0x388	Zq_cnt_3				Zq_cnt_2			

9.5 Software Programming Guide

9.5.1 Initial operation

The initialization operation is started when the software writes 1 to the register Init_start (0x018). Set Init_start

Before the signal, all other registers must be set to the correct values.

The DRAM initialization process of software and hardware cooperation is as follows:

(1) The software writes correct configuration values to all registers, but Init_start (0x018) is in the process

Must be kept at 0;

(2) The software sets Init_start (0x018) to 1, which will lead to the start of hardware initialization;

(3) The initialization operation starts inside the PHY, and the DLL will try to perform the lock operation. If the lock is successful, you can

Dll_init_done (0x000) reads the corresponding status, and can read and write from Dll_value_ck (0x000)

The number of front lock delay lines; if the lock is not successful, the initialization will not continue (at this time, you can set

Dll_bypass (0x018) makes initialization continue to execute);

(4) After the DLL is locked (or bypass set), the controller will send the DRAM to the DRAM according to the initialization requirements of the corresponding DRAM

Issue the corresponding initialization sequence, such as the corresponding MRS command, ZQCL command, etc .;

(5) Software can judge whether the memory initialization operation is completed by sampling the Dram_init (0x160) register.

9.5.2 Control of reset pin

In order to more easily control the reset pin in STR and other states, you can use the reset_ctrl (0x150) register

For special reset pin (DDR_RESETn) control, there are two main control modes:

(1) In general mode, reset_ctrl [1: 0] == 2'b00. In this mode, the reset signal pin behaves as a

Compatible with general control modes. Connect DDR_RESETn directly to the corresponding pin on the memory slot on the motherboard. lead

The behavior of the feet is:

- When not powered: the pin status is low;
- At power-on: the pin status is low;
- When the controller starts to initialize, the pin state is high;
- During normal operation, the pin status is high.

The timing is shown below:

(2) Reverse mode, reset_ctrl [1: 0] == 2'b10. In this mode, the reset signal pin is in memory

In actual control, the effective level is opposite to the general control mode. So the DDR_RESETn needs to be

Connect to the corresponding pin on the memory slot through the inverter. The behavior of the pins is:

- When not powered: the pin status is low;
- At power-on: the pin status is low;
- When the controller starts to configure: the pin state is high;
- When the controller starts to initialize: the pin state is low;
- Normal operation: The pin state is low.

The timing is shown below:

(3) Reset inhibit mode, pm_reset_ctrl [1: 0] == 2'b01. In this mode, the reset signal pin

During the work of a memory, keep low level. Therefore, the motherboard needs to pass DDR_RESETn through the inverter and the memory

The corresponding pins on the slot are connected. The behavior of the pins is:

- Always low

The timing is shown below:

By cooperating with the latter two reset modes, it is possible to realize STR directly using the reset signal of the memory controller control. When the entire system is started from a shutdown state, use the method in (2) to use the memory module to reset normally and start jobs. When the system recovers from the STR, use the method in (3) to reconfigure the memory module so that it does not damage The original state of the memory module makes it resume normal operation.

9.5.3 Leveling

Leveling operation is in DDR3, used to intelligently configure the phase relationship between various signals in the read and write operations of the memory controller Operation. Usually it includes Write Leveling, Read Leveling and Gate Leveling. In this controller, Only Write Leveling and Gate Leveling are implemented, Read Leveling is not implemented, the software needs to pass judgment The correctness of reading and writing to achieve the functions completed by Read Leveling. In addition to the DQS phase operating during Leveling In addition to the bit and GATE phase, the configuration of writing DQ phase and reading DQ phase can be calculated according to these last confirmed phases Reset method.

9.5.3.1 Write Leveling

(1) Write Leveling is used to configure the phase relationship between writing DQS and clock. Software programming needs to refer to The next step.

(2) Complete the controller initialization, see the previous section;

(3) Set Dll_wrdqs_x (x = 0… 8) to 0;

(4) Set Lvl_mode (0x180) to 2'b01;

(5) Sampling the Lvl_ready (0x180) register, if it is 1, it means that the Write Leveling request can be started;

(6) Set Lvl_req (0x180) to 1;

(7) Sampling the Lvl_done (0x180) register, if it is 1, it means that a Write Leveling request is completed;

(8) Sampling Lvl_resp_x (0x180, 0x188) register, if it is 0, the corresponding Dll_wrdqs_x [6: 0]

Increase 1 and repeat 5-7; if it is 1, it means that the Write Leveling operation has been successful;

(9) At this time, the value of Dll_wrdqs_x should be the correct setting value.

(10) At this point, the Write Leveling operation ends. If in this process, Lvl_resp_x is found at the first sampling

Is 1, the result is problematic, you should check whether other registers have wrong settings.

The memory may include Wrdqs_lt_half, Dqs_start_edge, Dqs_stop_edge, Dqs_oe_begin,

Dqs_oe_end.

(11) Then set Wrdqs_lt_half_x according to whether the value of Dll_wrdqs_x is less than 0x40;

(12) Set Dll_wrdata_x according to whether the value of Dll_wrdqs_x is less than 0x20. If Dll_wrdqs_x>

0x20, Dll_wrdata_x = Dll_wrdqs_x – 0x20, otherwise Dll_wrdata_x = Dll_wrdqs_x

+ 0x60;

(13) Set Wrdata_lt_half_x according to whether the value of Dll_wrdata_x is less than 0x40;

(14) Determine whether the following conditions exist: different Dll_wrdata_x values are near 0x40, and there are edges crossing 0x40

The situation appears (refer to some Dll_wrdata_x is slightly less than 0x40, and some Dll_wrdata_x is slightly greater than

0x40). If this happens, set the corresponding Wrdata_lt_half_x == 0 data set

Write_clk_delay_x is 1. Then reduce the values of tPHY_WRDATA and tRDDATA by 1;

(15) Set Lvl_mode (0x180) to 2'b00 to exit Write Leveling mode;

9.5.3.2 Gate Leveling

Gate Leveling is used to configure the timing of enabling the sampling and reading DQS window in the controller. Refer to the following steps for software programming.

(1) Complete the controller initialization, see the previous section;

(2) Complete Write Leveling, see the previous section;

(3) Set Dll_gate_x (x = 0… 8) to 0;

(4) Set Lvl_mode (0x180) to 2'b10;

(5) Sampling the Lvl_ready (0x180) register, if it is 1, it means that the Gate Leveling request can be started;

(6) Set Lvl_req (0x180) to 1;

(7) Sampling the Lvl_done (0x180) register, if it is 1, it means that a Gate Leveling request is completed;

(8) Sampling Lvl_resp_x [0] (0x180, 0x188) register. If the first sample finds Lvl_resp_x [0]

Is 1, increase the corresponding Dll_gate_x [6: 0] by 1 and repeat 6-8 until the sampling result is 0

Otherwise, proceed to the next step;

(9) If the sampling result is 0, increase the corresponding Dll_gate_x [6: 0] by 1 and repeat 6-9; if it is

1, it means that the Gate Leveling operation has been successful;

(10) At this point, the Gate Leveling operation ends, and the sum of Dll_gate_x [6: 0] and Dll_wrdata_x [6: 0]

In fact, it is to read the phase relationship of DQS relative to the PHY internal clock. The following is based on the results of Leveling

Adjust each parameter.

(11) If the sum of Dll_gate_x [6: 0] and Dll_wrdata_x [6: 0] is less than 0x20 or greater than 0x60, then

Dll_rddqs_lt_halt is set to 1. Because the phase relationship of rddqs is actually equal to the read DQS in the input

Delay by 1/4 on the basis.

(12) At this time, if the value of Dll_gate_x is greater than 0x40, the value of Dll_gate_x is subtracted from 0x40; otherwise,

Just set it to 0.

(13) After the adjustment, perform two Lvl_req operations respectively, and observe Lvl_resp_x [7: 5] and

The value of Lvl_resp_x [4: 2] changes. If each increase is Burst_length / 2, continue to step 13

Operation; if not 4, you may need to add or subtract one to Rd_oe_begin_x

Burst_length / 2, it is likely that some fine-tuning of the value of Dll_gate_x

(14) Set Lvl_mode (0x180) to 2'b00 to exit Gate Leveling mode;

9.5.4 Initiate MRS commands separately

The order of MRS commands issued by the memory controller to the memory are:

MR2_CS0, MR2_CS1, MR2_CS2, MR2_CS3,

MR3_CS0, MR3_CS1, MR3_CS2, MR3_CS3,

MR1_CS0, MR1_CS1, MR1_CS2, MR1_CS3,

MR0_CS0, MR1_CS1, MR1_CS2, MR1_CS3.

Among them, whether the MRS command corresponding to CS is valid or not is determined by Cs_mrs, and only the corresponding chip select on Cs_mrs Is valid, the MRS command will be issued to the DRAM. The corresponding value of each MR is determined by the register Mr * _cs *. These values are also used for MRS commands when initializing memory.

The specific operations are as follows:

(1) Set the registers Cs_mrs (0x168) and Mr * _cs * (0x190 – 0x1B8) to the correct values;

(2) Set Command_mode (0x190) to 1 to make the controller enter the command sending mode;

(3) Sampling Status_cmd (0x190), if it is 1, it means the controller has entered command sending mode

Go to the next step, if it is 0, you need to continue to wait;

(4) Write Mrs_req (0x198) to 1, send MRS command to DRAM;

(5) Sampling Mrs_done (0x198), if it is 1, it means that the MRS command has been sent and can be exited, such as

If it is 0, you need to continue to wait;

(6) Set Command_mode (0x190) to 0 to make the controller exit the command sending mode.

9.5.5 Any operation control bus

The memory controller can send any command combination to the DRAM through the command sending mode, and the software can set Cmd_cs, Cmd_cmd, Cmd_ba, Cmd_a (0x168), issued to DRAM in command sending mode.

The specific operations are as follows:

(1) Set the registers Cmd_cs, Cmd_cmd, Cmd_ba, Cmd_a (0x190) to the correct values;

(2) Set Command_mode (0x190) to 1 to make the controller enter the command sending mode;

(3) Sampling Status_cmd (0x190), if it is 1, it means the controller has entered command sending mode

Go to the next step, if it is 0, you need to continue to wait;

(4) Write Cmd_req (0x190) to 1 to send commands to DRAM;

(5) Set Command_mode (0x190) to 0 to make the controller exit the command sending mode.

9.5.6 Self-loop test mode control

The self-loop test mode can be used in test mode or normal function mode, for this reason, the memory controller

Two sets of independent control interfaces are realized respectively, one set is used for direct control by the test port in the test mode, and the other set is used for

In the normal function mode, the configuration enabling test is performed by the register configuration module.

The multiplexing of these two sets of interfaces is controlled by the port test_phy. When test_phy is valid, the controller 's

The test_ * port is controlled, and the self-test at this time is completely controlled by the hardware; when test_phy is invalid, use software programming

The parameters of pm_ * are controlled. The specific signal meaning of using the test port can refer to the part of the same name in the register parameters.

The two sets of interfaces are basically the same in terms of control parameters, only the access points are different. Here we introduce the

制 方法。 Manufacturing methods. The specific operations are as follows:

(1) Set all the parameters of the memory controller correctly;

(2) Set the register Lpbk_en (0x270) to 1;

(3) Set the register Init_start (0x018) to 1;

(4) The sampling register Dll_init_done (0x000), if this value is 1, it means that the DLL is locked and can

Proceed to the next operation; if this value is 0, you need to continue to wait; (when using the test port for control

When you do n't see the output of this register, you do n't need to sample this register, but only need to

Wait here for a certain period of time to ensure that the DLL is locked, and then proceed to the next step);

(5) Set the register Lpbk_start (0x270) to 1; at this time, the self-loop test is officially started.

So far, since the loop test has started, the software needs to constantly check whether there is an error. The specific operations are as follows:

(6) Sampling register Lpbk_error (0x270), if this value is 1, it means there is an error, at this time you can pass

Observe the first error through Lpbk_ * and other observation registers (0x270, 0x278, 0x280, 0x288)

Incorrect data and correct data; if this value is 0, it means that no data error has occurred.

9.5.7 ECC function usage control

The ECC function is only available in 64-bit mode.

Ecc_enable includes the following 4 control bits:

Ecc_enable [0] controls whether the ECC function is enabled. Only when this valid bit is set, the ECC function will be enabled.

Ecc_enable [1] controls whether an error is reported through the read response path inside the processor, so that two ECC bits appear

Wrong read access can immediately lead to abnormal processor cores.

Ecc_enable [2] controls whether an error is reported through the write response path inside the processor, so that two ECC bits appear

Wrong write access (write after read) can immediately cause an exception to the processor core.

Ecc_enable [3] controls the trigger timing of recording error information in the register. These error messages are performed without software

In the case of processing, it will not be triggered continuously, and only the information of the first error will be recorded. This information includes Ecc_code,

Ecc_addr, Ecc_data. When Ecc_enable [3] is 0, as long as an ECC error occurs (including 1 bit error

And 2 bit error), this record will be triggered, when Ecc_enable [3] is 1, only ECC two bits appear

Wrong, this record will be triggered. And this "first time" refers to that the corresponding bit of the interrupt vector register is set. Just

That is, the access that caused the interruption was recorded.

In addition, ECC errors can also be notified to the processor core through interrupts. This interrupt is entered via Int_enable

行 控制。 Line control. The interrupt includes two vectors, Int_vector [0] indicates that an ECC error (including 1 bit error and 2 bit error) occurs,

Int_vecotr [1] indicates that two ECC errors have occurred. Int_vector is cleared by writing 1 to the corresponding bit.

10 HyperTransport controller

In Loongson 3A3000 / 3B3000, the HyperTransport bus is used to connect external devices and interconnect multiple chips.

When used for peripheral connection, the user program can freely choose whether to support IO Cache consistency (through the address window Uncache

Set, see section 10.5.13 for details): When configured to support Cache consistency mode, IO device accesses internal DMA

Transparent to the Cache level, that is, the consistency is automatically maintained by the hardware, without the need for software to maintain it through the program Cache instructi

When the HyperTransport bus is used for multi-chip interconnection, the HT0 controller (the initial address is

0x0C00_0000_0000 – 0x0DFF_FFFF_FFFF) can support the consistent transmission of Cache between chips through pin configuration,

The HT1 controller (the initial address is 0x0E00_0000_0000 – 0x0FFF_FFFF_FFFF) can be supported by software configuration

For consistent maintenance of Cache between chips, see section 10.7.

The HyperTransport controller supports up to 16-bit bidirectional width and 2.0GHz operating frequency. At the beginning of the system automatically

After initializing the connection, the user program can modify the corresponding configuration register in the protocol to achieve the width and running frequency.

Changes, and re-initialize, see section 10.1 for specific methods.

The main features of Loongson 3A3000 / 3B3000 HyperTransport controller are as follows:

- Support HT1.0 / HT3.0 protocol
- Support 200/400/800/1600 / 2000MHz operating frequency
- HT1.0 supports 8-bit width
- HT3.0 supports 8/16 bit width
- Each HT controller (HT0 / HT1) can be configured as two 8-bit HT controllers
- The direction of bus control signals (including PowerOK, Rstn, LDT_Stopn) can be configured
- Peripheral DMA space Cache / Uncache can be configured
- It can be configured as Cache consistency mode when used for multi-chip interconnection

10.1 HyperTransport hardware setup and initialization

HyperTransport bus is composed of transmission signal bus and control signal pins, etc. The following table gives

HyperTransport bus related pins and their functional description.

Table 10-1 Hype T ansp Yaotong bus related pin signals

Pin	name	description
HT0_8x2	Bus width configuration	1: Configure the 16-bit HyperTransport bus as two independent 8-bit buses,

Page 63

		Controlled by two independent controllers, the address space is divided into
		HT0_Lo: address [40] = 0;
		HT0_Hi: address [40] = 1;
		0: Use the 16-bit HyperTransport bus as a 16-bit bus, by HT0_Lo control, the address space is the address of HT0_Lo, namely address [40] = 0; HT0_Hi all signals are invalid.

HT0_Lo_mode	Master mode	1: Set HT0_Lo as the master mode, in this mode, the bus control signal, etc. Driven by HT0_Lo, these control signals include HT0_Lo_Powerok, HT0_Lo_Rstn, HT0_Lo_Ldt_Stopn. In this mode, these controls The control signal can also be bidirectionally driven. At the same time this pin determines (negative) registration The initial value of the device "Act as Slave", when this register is 0, The Bridge bit in the packet on the HyperTransport bus is 1, otherwise it is 0. In addition, when this register is 0, if the HyperTransport bus When the requested address does not hit the receiving window of the controller, it will be regarded as P2P. Seek to send back to the bus again, if this register is 1, there is no hit, then make Respond to bad requests. 0: Set HT0_Lo to slave mode, in this mode, bus control signals, etc. Driven by the opposite device, these control signals include HT0_Lo_Powerok, HT0_Lo_Rstn, HT0_Lo_Ldt_Stopn. In this mode, these controls The control signal is driven by the other device. If it is not driven correctly, the Does not work correctly.
HT0_Lo_Powerok	Bus Powerok	HyperTransport bus Powerok signal, When HT0_Lo_Mode is 1, it is controlled by HT0_Lo; When HT0_Lo_Mode is 0, it is controlled by the opposite device.
HT0_Lo_Rstn	Bus Rstn	HyperTransport bus Rstn signal, When HT0_Lo_Mode is 1, it is controlled by HT0_Lo; When HT0_Lo_Mode is 0, it is controlled by the opposite device.
HT0_Lo_Ldt_Stopn	Bus Ldt_Stopn	HyperTransport bus Ldt_Stopn signal, When HT0_Lo_Mode is 1, it is controlled by HT0_Lo; When HT0_Lo_Mode is 0, it is controlled by the opposite device.
HT0_Lo_Ldt_Reqn	Bus Ldt_Reqn	HyperTransport bus Ldt_Reqn signal,
HT0_Hi_mode	Master mode	1: Set HT0_Hi to master mode, in this mode, bus control signals, etc. Driven by HT0_Hi, these control signals include HT0_Hi_Powerok, HT0_Hi_Rstn, HT0_Hi_Ldt_Stopn. In this mode, these controls The control signal can also be bidirectionally driven. At the same time this pin determines (negative) registration The initial value of the device "Act as Slave", when this register is 0, The Bridge bit in the packet on the HyperTransport bus is 1, otherwise it is 0. In addition, when this register is 0, if the HyperTransport bus When the requested address does not hit the receiving window of the controller, it will be regarded as P2P. Seek to send back to the bus again, if this register is 1, there is no hit, then make Respond to bad requests. 0: Set HT0_Hi to slave mode, in this mode, bus control signals, etc. Driven by the counterpart device, these control signals include HT0_Hi_Powerok,

Page 64

		HT0_Hi_Rstn, HT0_Hi_Ldt_Stopn. In this mode, these controls The control signal is driven by the other device. If it is not driven correctly, the Does not work correctly.
HT0_Hi_Powerok	Bus Powerok	HyperTransport bus Powerok signal, When HT0_Lo_Mode is 1, it is controlled by HT0_Hi; When HT0_Lo_Mode is 0, it is controlled by the opposite device. When HT0_8x2 is 1, control the upper 8-bit bus; When HT0_8x2 is 0, it is invalid.

HT0_Hi_Rstn	Bus Rstn	HyperTransport bus Rstn signal, When HT0_Lo_Mode is 1, it is controlled by HT0_Hi; When HT0_Lo_Mode is 0, it is controlled by the opposite device. When HT0_8x2 is 1, control the upper 8-bit bus; When HT0_8x2 is 0, it is invalid.
HT0_Hi_Ldt_Stopn	Bus Ldt_Stopn	HyperTransport bus Ldt_Stopn signal, When HT0_Lo_Mode is 1, it is controlled by HT0_Hi; When HT0_Lo_Mode is 0, it is controlled by the opposite device. When HT0_8x2 is 1, control the upper 8-bit bus; When HT0_8x2 is 0, it is invalid.
HT0_Hi_Ldt_Reqn	Bus Ldt_Reqn	HyperTransport bus Ldt_Reqn signal, When HT0_8x2 is 1, control the upper 8-bit bus; When HT0_8x2 is 0, it is invalid.
HT0_Rx_CLKp [1: 0] HT0_Rx_CLKn [1: 0] HT0_Tx_CLKp [1: 0] HT0_Tx_CLKp [1: 0]	CLK [1: 0]	HyperTransport bus CLK signal When HT0_8x2 is 1, CLK [1] is controlled by HT0_Hi CLK [0] is controlled by HT0_Lo When HT0_8x2 is 0, CLK [1: 0] is controlled by HT0_Lo
HT0_Rx_CTLp [1: 0] HT0_Rx_CTLn [1: 0] HT0_Tx_CTLp [1: 0] HT0_Tx_CTLn [1: 0]	CTL [1: 0]	HyperTransport bus CTL signal When HT0_8x2 is 1, CTL [1] is controlled by HT0_Hi CTL [0] is controlled by HT0_Lo When HT0_8x2 is 0, CTL [1] is invalid CTL [0] is controlled by HT0_Lo
HT0_Rx_CADp [15: 0] HT0_Rx_CADn [15: 0] HT0_Tx_CADp [15: 0] HT0_Tx_CADn [15: 0]	CAD [15: 0]	HyperTransport bus CAD signal When HT0_8x2 is 1, CAD [15: 8] is controlled by HT0_Hi CAD [7: 0] is controlled by HT0_Lo When HT0_8x2 is 0, CAD [15: 0] is controlled by HT0_Lo

The initialization of HyperTransport starts automatically after each reset is completed, and the HyperTransport bus after a cold start

It will automatically work at the lowest frequency (200MHz) and the smallest width (8bit), and try to initiate a bus initialization handshake. initialization

Whether it is in the completed state can be read from the register "Init Complete" (see Section 10.5.2). After initialization,

The width of the bus can be read from the registers "Link Width Out" and "Link Width In" (see Section 10.5.2).

After initialization, the user can rewrite the registers "Link Width Out", "Link Width In" and "Link

Freq ", at the same time, you need to configure the corresponding register of the other device. After the configuration is completed, you need to warm reset the bus or pass

The "HT_Ldt_Stopn" signal performs a reinitialization operation so that the rewritten value of the register takes effect. Reinitialized

After completion, the HyperTransport bus will work at the new frequency and width. It should be noted that both ends of HyperTransport

The configuration of the device needs to be one-to-one, otherwise the HyperTransport interface will not work properly.

10.2 HyperTransport protocol support

Godson 3A3000 / 3B3000's HyperTransport bus supports most commands in version 1.03 / 3.0 protocol, and

In addition, some extended instructions are added to the extended consistency protocol that supports multi-chip interconnection. In the above two modes,

The commands that the HyperTransport receiver can receive are shown in the following table. It should be noted that HyperTransport is not supported

Bus atomic operation commands.

Table 10- 2 Hype, T, and ansp commands that can be received by the receiving end

coding	aisle	command	Standard mode	Extension (consistency)
000000	-	NOP	Empty package or flow control	
000001	NPC	FLUSH	No operation	
x01xxx	NPC or PC	Write	bit 5: 0-Nonposted 1-Posted bit 2: 0 – Byte	bit 5: Must be 1, POSTED bit 2: 0 – Byte

			1 – Doubleword bit 1: Don't Care bit 0: Don't Care	1 – Doubleword bit 1: Don't Care bit 0: must be 1
01xxxx	NPC	Read	bit 3: Don't Care bit 2: 0 – Byte 1 – Doubleword bit 1: Don't Care bit 0: Don't Care	bit 3: Don't Care bit 2: 0 – Byte 1 – Doubleword bit 1: Don't Care bit 0: must be 1
110000	R	RdResponse	Read operation returns	
110011	R	TgtDone	Write operation returns	
110100	PC	WrCoherent	----	Write command extension
110101	PC	WrAddr	----	Write address extension
111000	R	RespCoherent	----	Read response extension
111001	NPC	RdCoherent	----	Read command extension
111010	PC	Broadcast	No operation	
111011	NPC	RdAddr	----	Read address extension
111100	PC	FENCE	Guaranteed order relationship	
111111	-	Sync / Error	Sync / Error	

For the sending end, the commands sent out in the two modes are shown in the following table.

Table 10- 3 Commands to be sent out in two modes

coding	aisle	command	Standard mode	Extension (consistency)
000000	-	NOP	Empty package or flow control	
x01x0x	NPC or PC	Write	bit 5: 0-Nonposted 1-Posted bit 2: 0 – Byte 1 – Doubleword bit 0: must be 0	bit 5: Must be 1, POSTED bit 2: 0 – Byte 1 – Doubleword bit 0: must be 1
010x0x	NPC	Read	bit 2: 0 – Byte 1 – Doubleword bit 0: Don't Care	bit 2: 0 – Byte 1 – Doubleword bit 0: must be 1
110000	R	RdResponse	Read operation returns	
110011	R	TgtDone	Write operation returns	
110100	PC	WrCoherent	----	Write command extension
110101	PC	WrAddr	----	Write address extension
111000	R	RespCoherent	----	Read response extension
111001	NPC	RdCoherent	----	Read command extension
111011	NPC	RdAddr	----	Read address extension
111111	-	Sync / Error	Will only forward	

Page 66

10.3 HyperTransport interrupt support

The HyperTransport controller provides 256 interrupt vectors, which can support Fix, Arbiter and other types of interrupts. However, there is no support for hardware automatic EOI. For the above two supported types of interrupts, the controller will Automatically write to the interrupt register, and perform interrupt notification to the system interrupt controller according to the setting of the interrupt mask register. With For the body's interrupt control, please see the interrupt control register set in Section 10.5.8.

In addition, the controller provides special support for PIC interrupts to speed up this type of interrupt processing.

A typical PIC interrupt is completed by the following steps: ① The PIC controller sends a PIC interrupt request to the system; ② The system Send the interrupt vector query to the PIC controller; ③ The PIC controller sends the interrupt vector number to the system; ④ The system clears the PIC controller The corresponding interrupt on the controller. Only after the above four steps are completed, the PIC controller will issue the next interrupt to the system. for Loongson 3A3000 / 3B3000 HyperTransport controller will automatically perform the first three steps of processing and interrupt the PIC to Write the corresponding position in the 256 interrupt vectors. After the software system has processed the interrupt, it needs to go to step 4. Management, that is, to issue a clear interrupt to the PIC controller. After that, the process of the next interrupt is started.

10.4 HyperTransport address window

10.4.1 HyperTransport space

In the Loongson 3A3000 / 3B3000 processor, the default address windows of the four HyperTransport interfaces are distributed as follows:

Table 10- 4 Address window distribution of the default 4 Hype, T, and Ansp interfaces

Base address	End address	size	definition
0x0C00_0000_0000	0x0CFF_FFFF_FFFF	1 Tbytes	HT0_LO window
0x0D00_0000_0000	0x0DFF_FFFF_FFFF	1 Tbytes	HT0_HI window
0x0E00_0000_0000	0x0EFF_FFFF_FFFF	1 Tbytes	HT1_LO window
0x0F00_0000_0000	0x0FFF_FFFF_FFFF	1 Tbytes	HT1_HI window

By default (not configured separately for the system address window), the software

HyperTransport interface to access, in addition, the software can also configure the address window on the crossbar

Implement access to it with other address spaces (see section 2.5 for details). 40 inside each HyperTransport interface

The address window distribution of the bit address space is shown in the following table.

Table 10- 5 Address window distribution inside the interface of Loongson No. 3 processor Hype-T-Tan-Ansp

Base address	End address	size	definition
0x00_0000_0000	0xFC_FFFF_FFFF	1012 Gbytes	MEM space
0xFD_0000_0000	0xFD_F7FF_FFFF	3968 Mbytes	Keep
0xFD_F800_0000	0xFD_F8FF_FFFF	16 Mbytes	Interrupt
0xFD_F900_0000	0xFD_F90F_FFFF	1 Mbyte	PIC interrupt response
0xFD_F910_0000	0xFD_F91F_FFFF	1 Mbyte	system message
0xFD_F920_0000	0xFD_FAFF_FFFF	30 Mbytes	Keep
0xFD_FB00_0000	0xFD_FBFF_FFFF	16 Mbytes	HT controller configuration space
0xFD_FC00_0000	0xFD_FDFF_FFFF	32 Mbytes	I / O space
0xFD_FE00_0000	0xFD_FFFF_FFFF	32 Mbytes	HT bus configuration space
0xFE_0000_0000	0xFF_FFFF_FFFF	8 Gbytes	Keep

10.4.2 Internal window configuration of HyperTransport controller

Loongson 3A3000 / 3B3000 processor provides a variety of rich address windows in HyperTransport

For user use, the functions and functions of these address windows are described in the following table.

Table 10-6 Address window provided in the interface of Loongson 3A3000 / 3B3000 processor Hype, T, ansp, Yaot

Address window	Number of windows	Accept bus	effect	Remarks
Receive window (See window configuration Section 10.5.7)	3	HyperTransport	Determine whether to receive HyperTransport Visits sent on the bus ask.	When in main bridge mode (ie configuration register Act_as_slave is 0), only falling Access in these address windows will be included The local bus responds, other visits will be Think it is P2P access and send it back HyperTransport bus; in the design When in standby mode (that is, in the configuration register act_as_slave is 1), only falls here Access in these address windows will be Received and processed by the line, other visits will be According to the agreement to return an error.
Post window (See window configuration 10.5.11)	2	Internal bus	Determine if it will be internal total line Correct HyperTransport Bus write access Post Write	External write visits that fall in these address spaces Question will be as Post Write. Post Write: HyperTransport protocol In this kind of write access does not need to wait for writing In response, that is, the controller sends to the bus

Page 68

			After this write access will enter the processor Row write access complete response.
Prefetch window (See window configu2ration 10.5.12)	Internal bus	Determine whether to receive Department 's Cache access Fetch access.	When the processor cores are executed out of order, the total Issue some guess read access or fetch Access, this access for some IO space it is wrong. By default, this Access to the HT controller will return directly without Visit the HyperTransport bus ask. Through these windows you can enable This type of access to the HyperTransport bus ask.
Uncache window (See window configu2ration 10.5.13)	HyperTransport	Determine whether to HyperTransport Access operations on the bus For internal Uncache access	Loongson 3A3000 / 3B3000 processor Department of IO DMA access, in the case of Access via Cache SCache judges that it is a hit, thus maintaining it IO consistency information. And through these windows The configuration can make hits in these windows Access directly in the way of Uncache Ask about memory without maintaining its IO through hardware Consistency information.

10.5 Configuration Register

The configuration register module is mainly used to control the configuration register access from the AXI SLAVE terminal or the HT RECEIVER terminal.

Ask for requests, perform external interrupt processing, and save a large number of configuration files that can be seen by the software for controlling the various working modes of

Memory.

First, the access and storage of configuration registers used to control various behaviors of the HT controller are in this module

The access offset address is 0xFD_FB00_0000 to 0xFD_FBFF_FFFF on the HT controller side. All software in the HT controller

The visible registers of the software are shown in the following table:

Table 10-7 Software visible register list

Offset address	name	description
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	Custom register	MISC
0x54	Receive Diagnostic Register	Used to diagnose the signal sampled at the receiving end
0x58	Interrupt routing mode selection register	Corresponding to 3 interrupt routing methods
0x5c	Receive buffer register	
0x60	Receive address window Configuration register	HT bus receive address window 0 enable (external access)
0x64		HT bus receive address window 0 base address (external access)
0x68		HT bus receive address window 1 enable (external access)
0x6c		HT bus receive address window 1 base address (external access)
0x70		HT bus receive address window 2 enable (external access)
0x74		HT bus receive address window 2 base address (external access)
0x148		HT bus receive address window 3 enable (external access)
0x14c		HT bus receive address window 3 base address (external access)
0x150		HT bus receive address window 4 is enabled (external access)

Page 69

0x154		HT bus receive address window 4 base address (external access)
0x80	Interrupt vector register	HT bus interrupt vector register [31: 0]
0x84		HT Bus Interrupt Vector Register [63:32]
0x88		HT Bus Interrupt Vector Register [95:64]
0x8c		HT bus interrupt vector register [127: 96]
0x90		HT bus interrupt vector register [159: 128]
0x94		HT Bus Interrupt Vector Register [191: 160]
0x98		HT Bus Interrupt Vector Register [223: 192]
0x9C		HT Bus Interrupt Vector Register [255: 224]
0xA0	Interrupt enable register	HT bus interrupt enable register [31: 0]
0xA4		HT bus interrupt enable register [63:32]
0xA8		HT bus interrupt enable register [95:64]
0xAC		HT bus interrupt enable register [127: 96]
0xB0		HT bus interrupt enable register [159: 128]
0xB4		HT bus interrupt enable register [191: 160]
0xB8		HT bus interrupt enable register [223: 192]
0xBC		HT bus interrupt enable register [255: 224]
0xC0	Interrupt Discovery & Configuration	Interrupt Capability
0xC4		DataPort
0xC8		IntrInfo [31: 0]
0xCC		IntrInfo [63:32]
0xD0	POST address window Configuration register	HT bus POST address window 0 enable (internal access)
0xD4		HT bus POST address window 0 base address (internal access)
0xD8		HT bus POST address window 1 enable (internal access)
0xDC		HT bus POST address window 1 base address (internal access)
0xE0	Prefetchable address window Configuration register	HT bus can be prefetched address window 0 enabled (internal access)
0xE4		HT bus prefetchable address window 0 base address (internal access)
0xE8		HT bus prefetch address window 1 enabled (internal access)
0xEC		Ht bus prefetchable address window 1 base address (internal access)
0xF0	Uncache address window Configuration register	HT bus Uncache address window 0 enable (external access)
0xF4		HT bus Uncache address window 0 base address (external access)
0xF8		HT bus Uncache address window 1 is enabled (external access)
0xFC		HT bus Uncache address window 1 base address (external access)
0x168		HT bus Uncache address window 2 enable (external access)
0x16C		HT bus Uncache address window 2 base address (external access)
0x170		HT bus Uncache address window 3 enable (external access)
0x174		HT bus Uncache address window 3 base address (external access)
0x158	P2P address window configuration register	HT bus P2P address window 0 enable (external access)
0x15C		HT bus P2P address window 0 base address (external access)
0x160		HT bus P2P address window 1 enable (external access)
0x164		HT bus P2P address window 1 base address (external access)
0x100	Sender buffer size register	Sender command buffer size register
0x104		Data buffer size register at the sending end
0x108	Buffer debug register on the sending side	Used to manually set the size of the sender buffer (for debugging)
0x10C	PHY matching configuration registers	for impedance PHY Configuration of the sending and receiving ends
0x110	Revision ID register	Used to configure the controller version
0x118	Error Retry Control Register	Retry Count Rollover, Short Retry Attempts
0x11C	Retry Count register	Used for error retransmission count in HyerTransport 3.0 mode
0x130	Link Train Register	HyperTransport 3.0 link initialization and link training control
0x134	Training 0 timeout short count send Register	Used for Training 0 short timer timeout threshold configuration
0x138	Training 0 Overtime long count Register	Used for Training 0 long count timeout threshold configuration
0x13C	Training 1 count register	Used for Training 1 count threshold configuration
0x140	Training 2 count register	For Training 2 count threshold configuration
0x144	Training 3 count register	Used for Training 3 count threshold configuration
0x178	Software frequency configuration register	Realize the frequency switching of the controller in the working process
0x17C	PHY configuration register	Used to configure PHY related physical parameters
0x180	Link initialization debug register	Used to ignore the PHY CDR lock signal and customize the waiting time
0x184	LDT debug register	It is used to configure the time from invalid LDT signal to link initialization

The specific meaning of each register is as follows:

10.5.1 Bridge Control

Offset: 0x3C

Reset value: 0x00000000

name: Bus Reset Control

Table 10- 8 Bus Reset Register Definition

Bit field	Bit field name	Bit width	reset value	Visit	description
31:23	Reserved	4	0x0		Keep
twenty two	Reset	12	0x0	R / W	Bus reset control: 0-> 1: Set HT_RSTn to 0, reset the bus 1-> 0: HT_RSTn is set to 1, the bus is reset
21: 0	Reserved	5	0x0		Keep

10.5.2 Capability Registers

Offset: 0x40

Reset value: 0x20010008

name: Command, Capabilities Pointer, Capability ID

Table 10- 9 C Yao mmand, Capab lt es P Yao nte, Capab l ty ID register definition

Bit field	Bit field name	Bit width	reset value	Visit	description
31:29	HOST / Sec	3	0x1	R	Command format is HOST / Sec
28:27	Reserved	2	0x0	R	Keep
26	Act as Slave	1	0x0 / 0x1	R / W	HOST / SLAVE mode The initial value is determined by the pin HOSTMODE HOSTMODE pull-up: 0 HOSTMODE drop-down: 1
25	Reserved	1	0x0		Keep
twenty four	Host Hide	1	0x0	Whether R / W prohibits register access from HT bus	
twenty thre	Reserved	1	0x0		Keep
22:18	Unit ID	5	0x0	R / W	In HOST mode: can be used to record the number of IDs used In SLAVE mode: record your own Unit ID
17	Double Ended	1	0x0	R	No dual HOST mode
16	Warm Reset	1	0x1	R	Bridge Control uses warm reset in reset
15: 8	Capabilities Pointer	8	0xa0	R	Next Cap register offset address
7: 0	Capability ID	8	0x08	R	HyperTransport capability ID

Offset: 0x44

Reset value: 0x00112000

name: Link Config, Link Control

Table 10-10 L nk C Yao nf g, L nk C Yao nt Ya Yao l Register definition

Bit field	Bit field name	Bit width	reset value	Visit	description
31	ht_phase_select _disable	1	0x0		Phase selection enable 0: enable phase selection function 1: Disable the phase selection function
30:28	Link Width Out	3	0x0	R / W	Sender width The value after cold reset is the maximum width of the current connection, write this post The value of the register will be the next warm reset or HT Effective after Disconnect 000: 8-bit mode 001: 16-bit mode
27	Reserved	1	0x0		Keep

26:24	Link Width In	3	0x0	R / W	Receiver width The value after cold reset is the maximum width of the current connection, write this post The value of the register will be the next warm reset or HT Effective after Disconnect
twenty three	Dw Fc out	1	0x0	R	The sender does not support double-word flow control
22:20	Max Link Width out	3	0x1	R	The maximum width of the sending end of the HT bus: 16bits
19	Dw Fc In	1	0x0	R	The receiver does not support double-word flow control
18:16	Max Link Width In	3	0x1	R	Maximum width of HT bus receiving end: 16bits
15:14	Reserved	2	0x0		Keep
13	LDTSTOP # Tristate Enable	1	0x1	R / W when	When the HT bus enters the HT Disconnect state, is it off Close HT PHY 1: Close 0: do not close
12:10	Reserved	3	0x0		Keep
9	CRC Error (hi)	1	0x0	R / W	CRC error in the upper 8 bits
8	CRC Error (lo)	1	0x0		CRC error occurred in the lower 8 bits of R / W
7	Trans off	1	0x0	R / W	HT PHY shutdown control When in 16-bit bus operating mode 1: Turn off high / low 8-bit HT PHY 0: enable the low 8-bit HT PHY, The upper 8-bit HT PHY is controlled by bit 0
6	End of Chain	0	0x0	R	HT bus end
5	Init Complete	1	0x0	R	Whether the HT bus initialization is completed
4	Link Fail	1	0x0	R	Indicates connection failure
3: 2	Reserved	2	0x0		Keep
1	CRC Flood Enable	1	0x0	R / W	Whether to flood the HT bus when a CRC error occurs
0	Trans off (hi)	1	0x0	R / W use	When the 16-bit HT bus runs an 8-bit protocol, High 8-bit PHY shutdown control 1: Turn off the upper 8-bit HT PHY 0: enable high 8-bit HT PHY

Offset: 0x48

Reset value: 0x80250023

name: Revision ID, Link Freq, Link Error, Link Freq Cap

Table 10-11 Rev s Yao n ID, L nk F eq, L nk E 耀 偡, L nk F eq eq Cap Register definition

Bit field	Bit field name	Bit width	reset value	Visit	description
31:16	Link Freq Cap	16	0x0025	R	Supported HT bus frequency, generated according to external PLL settings Different values
15:14	Reserved	2	0x0		Keep
13	Over Flow Error	1	0x0	R	HT bus packet overflow
12	Protocol Error	1	0x0	R / W	Agreement error, Refers to an unrecognized command received on the HT bus

Page 73

11: 8	Link Freq	4	0x0	R / W	HT bus operating frequency The value written to this register will be the next warm reset or HT Effective after Disconnect 0000: 200M 0010: 400M 0101: 800M
7: 0	Revision ID	8	0x23	R / W	version number: 1.03

Offset: 0x4C

Reset value: 0x00000002

name: Feature Capability

Table 10-12 Definition of Featu e Capab l ty register

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 9	Reserved	25	0x0		Keep
8	Extended Register 1		0x0	R	No
7: 4	Reserved	3	0x0		Keep
3	Extended CTL Time 1		0x0	R	No need
2	CRC Test Mode	1	0x0	R	not support
1	LDTSTOP #	1	0x1	R	Support LDTSTOP #
0	Isochronous Mode 1		0x0	R	not support

10.5.3 Custom register

Offset: 0x50

Reset value: 0x00904321

name: MISC

Table 10- 13 MISC register definition

Bit field	Bit field name	Bit width	reset value	Visit	description
31	Reserved	1	0x0		Keep
30	Ldt Stop Gen	1	0x0	R / W	makes the bus enter LDT DISCONNECT mode The correct method is: 0-> 1
29	Ldt Req Gen	1	0x0	R / W	from Wake up HT bus in LDT DISCONNECT, set LDT_REQ_n The correct way is to set 0 first and then set 0: 0-> 1 In addition, direct read and write requests to the bus can also be automatically Wake up bus

Page 74

28:24	Interrupt Index	5	0x0	R / W	To which redirects other than standard interrupts are redirected to In the interrupt vector (including SMI, NMI, INIT, INTA, INTB, INTC, INTD) A total of 256 interrupt vectors, this register indicates the interrupt direction

					The upper 5 bits of the quantity, the internal interrupt vector is as follows: 000: SMI 001: NMI 010: INIT 011: Reservered 100: INTA 101: INTB 110: INTC 111: INTD
twenty three	Dword Write	1	0x1	R / W	for 32/64/128/256 bit write access, whether to use Dword Write command format 1: Use Dword Write 0: Use Byte Write (with MASK)
twenty two	Coherent Mode	1	0x0	R	Whether it is processor consistency mode Determined by pin ICCC_EN
twenty one	Not Care Seqid	1	0x0	Does R / W	don't care about HT order relationship
20	Not Axi2Seqid	1	0x1	R	Whether to convert the commands on the Axi bus to different SeqIDs, If not converted, all read and write commands will use Fixed Fixed ID number in Seqid 1: No conversion 0: conversion
19:16	Fixed Seqid	4	0x0	R / W	When Not Axi2Seqid is valid, configure the Seqid
15:12	Priority Nop	4	0x4	R / W	HT bus Nop flow control packet priority
11: 8	Priority NPC	4	0x3	R / W	Non Post channel read and write priority
7: 4	Priority RC	4	0x2	R / W	Response channel reading and writing priority
3: 0	Priority PC	4	0x1	R / W	Post channel read and write priority 0x0: highest priority 0xF: lowest priority The priority of each channel is changed according to time. High priority strategy, the group register is used to configure each channel 'S initial priority

10.5.4 Receive diagnostic register

Offset:	0x54
Reset value:	0x00000000
name:	Receive Diagnostic Register

Table 10-14 Receive Diagnostic Register

Bit field	Bit field name	Bit width	reset value	Visit	description
0	Sample_en	1	0x0	R / W	Enable cad and ctl for sampling input 0x0: prohibited 0x1: enable
15: 8	x_ctl_catch	twenty four	0x0	R / W	Save the sampled input ctl (0, 2, 4, 6) Four phases corresponding to CTL0 sampling (1, 3, 5, 7) Four phases corresponding to CTL1 sampling
31:16	x_cad_phase_0	twenty four	0x0	R / W	save the input CAD [15: 0] value obtained by sampling

10.5.5 Interrupt routing mode selection register

Offset: 0x58

Reset value: 0x00000000

name: Interrupt routing mode selection register

Table 10-15 Interrupt Route Selection Register

Bit field	Bit field name	Bit width	reset value	Visit description
9: 8	ht_int_stripe	2	0x0	R / W corresponds to 3 interrupt routing methods, see 0 interrupt direction for details Volume register 0x0: ht_int_stripe_1 0x1: ht_int_stripe_2 0x2: ht_int_stripe_4

10.5.6 Receive buffer initial register

Offset: 0x5c

Reset value: 0x07778888

name: Receive buffer initialization configuration register

Table 10- 16 Receive buffer initial register

Bit field	Bit field name	Bit width	reset value	Visit description
27:24	rx_buffer_r_data	4	0x0	R / W Receive buffer read data buffer initialization information
23:20	rx_buffer_npc_data	4	0x0	R / W receive buffer npc data buffer initialization information
19:16	rx_buffer_pc_data	4	0x0	R / W receive buffer pc data buffer initialization information
15:12	rx_buffer_b_cmd	4	0x0	R / W receive buffer initialization command buffer initialization information
11: 8	rx_buffer_r_cmd	4	0x0	R / W receive buffer read command initialization information
7: 4	rx_buffer_npc_cmd	4	0x0	R / W receive buffer npc command buffer initialization information
3: 0	rx_buffer_pc_cmd	4	0x0	R / W receive buffer pc command buffer initialization information

10.5.7 Receive address window configuration register

The address window hit formula in the HT controller is as follows:

hit = (BASE & MASK) == (ADDR & MASK)

addr_out = TRANS_EN? TRANS | ADDR & ~ MASK: ADDR

It should be noted that when configuring the address window register, the high bit of MASK should be all 1, and the low bit should be all 0. 0 in MASK

The actual number of bits indicates the size of the address window.

The address in the receive address window is the address received on the HT bus. The HT address falling within the P2P window will be regarded as P2P

The command is forwarded back to the HT bus, and the HT address that falls within the normal receive window and is not in the P2P window will be sent to the CPU.

The command at its address will be forwarded back to the HT bus as a P2P command.

Offset: 0x60

Reset value: 0x00000000

name: HT bus receive address window 0 enable (external access)

Table 10- 17 HT Bus Receive Address Window 0 Enable (External Access) Register Definition

Bit field	Bit field name	Bit width	reset value	Visit description
31	ht_rx_image0_en	1	0x0	R / W HT bus receives address window 0, enable signal

30	ht_rx_image0_ trans_en	1	0x0	R / W HT bus receives address window 0, mapping enable signal
29: 0	ht_rx_image0_ trans [53:24]	30	0x0	R / W HT bus receive address window 0, the mapped address [53:24]

Offset: 0x64

Reset value: 0x00000000

name: HT bus receive address window 0 base address (external access)

Table 10- 18 HT bus receive address window 0 base address (external access) register definition

Bit field	Bit field name	Bit width	reset value	Visit description
31:16	ht_rx_image0_ base [39:24]	16	0x0	R / W HT bus receive address window 0, address base address [39:24]
15: 0	ht_rx_image0_ mask [39:24]	16	0x0	R / W HT bus receive address window 0, address masked [39:24]

Offset: 0x68

Reset value: 0x00000000

name: HT bus receive address window 1 enable (external access)

Table 10- 19 HT bus receive address window 1 enable (external access) register definition

Bit field	Bit field name	Bit width	reset value	Visit description
31	ht_rx_image1_en	1	0x0	R / W HT bus receives address window 1, enable signal
30	ht_rx_image1_ trans_en	1	0x0	R / W HT bus receives address window 1, map enable signal
29: 0	ht_rx_image1_ trans [53:24]	30	0x0	R / W HT bus receive address window 1, the mapped address [53:24]

Offset: 0x6c

Reset value: 0x00000000

name: HT bus receive address window 1 base address (external access)

Table 10- 20 HT bus receive address window 1 base address (external access) register definition

Bit field	Bit field name	Bit width	reset value	Visit description
31:16	ht_rx_image1_ base [39:24]	16	0x0	R / W HT bus receive address window 1, address base address [39:24]
15: 0	ht_rx_image1_ mask [39:24]	16	0x0	R / W HT bus receive address window 1, address masked [39:24]

Offset: 0x70

Reset value: 0x00000000

name: HT bus receive address window 2 enable (external access)

Table 10- 21 HT Bus Receive Address Window 2 Enable (External Access) Register Definition

Bit field	Bit field name	Bit width	reset value	Visit description
31	ht_rx_image2_en	1	0x0	R / W HT bus receives address window 2, enable signal
30	ht_rx_image2_ trans_en	1	0x0	R / W HT bus receives address window 2, map enable signal

Page 78

Bit field	Bit field name	Bit width	reset value	Visit	description
29: 0	ht_rx_image2_ trans [53:24]	16	0x0	R / W	HT bus receive address window 2, the translated address [53:24]

Offset: 0x74

Reset value: 0x00000000

name: HT bus receive address window 2 base address (external access)

Table 10- 22 HT Bus Receive Address Window 2 Base Address (External Access) Register Definition

Bit field	Bit field name	Bit width	reset value	Visit	description
31:16	ht_rx_image2_ base [39:24]	16	0x0	R / W	HT bus receive address window 2, address base address [39:24]
15: 0	ht_rx_image2_ mask [39:24]	16	0x0	R / W	HT bus receive address window 2, address masked [39:24]

Offset: 0x148

Reset value: 0x00000000

name: HT bus receive address window 3 enable (external access)

Table 10- 23 HT Bus Receive Address Window 3 Enable (External Access) Register Definition

Bit field	Bit field name	Bit width	reset value	Visit	description
31	ht_rx_image3_en	1	0x0	R / W	HT bus receives address window 3, enable signal
30	ht_rx_image3_ trans_en	1	0x0	R / W	HT bus receives address window 3, mapping enable signal
29: 0	ht_rx_image3_ trans [53:24]	16	0x0	R / W	HT bus receive address window 3, the translated address [53:24]

Offset: 0x14C

Reset value: 0x00000000

name: HT bus receive address window 3 base address (external access)

Table 10- 24 HT Bus Receive Address Window 3 Base Address (External Access) Register Definition

Bit field	Bit field name	Bit width	reset value	Visit	description
31:16	ht_rx_image3_ base [39:24]	16	0x0	R / W	HT bus receive address window 3, address base address [39:24]
15: 0	ht_rx_image3_ mask [39:24]	16	0x0	R / W	HT bus receive address window 3, address masked [39:24]

Offset: 0x150

Page 79

Reset value: 0x00000000

name: HT bus receive address window 4 is enabled (external access)

Table 10- 25 HT Bus Receive Address Window 4 Enable (External Access) Register Definition

Bit field	Bit field name	Bit width	reset value	Visit description
31	ht_rx_image4_en	1	0x0	R / W HT bus receives address window 4, enable signal
30	ht_rx_image4_ trans_en	1	0x0	R / W HT bus receives address window 4, map enable signal
29: 0	ht_rx_image4_ trans [53:24]	16	0x0	R / W HT bus receive address window 4, the translated address [53:24]

Offset: 0x154

Reset value: 0x00000000

name: HT bus receive address window 4 base address (external access)

Table 10-26 HT Bus Receive Address Window 4 Base Address (External Access) Register Definition

Bit field	Bit field name	Bit width	reset value	Visit description
31:16	ht_rx_image4_ base [39:24]	16	0x0	R / W HT bus receive address window 4, address base address [39:24]
15: 0	ht_rx_image4_ mask [39:24]	16	0x0	R / W HT bus receive address window 4, address masked [39:24]

10.5.8 Interrupt Vector Register

A total of 256 interrupt vector registers, including the direct mapping of Fix, Arbiter and PIC interrupts on the HT bus

Up to this 256 interrupt vectors, other interrupts such as SMI, NMI, INIT, INTA, INTB, INTC, INTD can

To map to any 8-bit interrupt vector through [28:24] of register 0x50, the order of mapping is {INTD, INTC,

INTB, INTA, 1'b0, INIT, NMI, SMI}. At this time, the corresponding value of the interrupt vector is {Interrupt Index, internal

The amount [2: 0]}.

LS3A1000E and above, 256 interrupt vectors choose different mappings of register configuration according to interrupt routing

To different interrupt lines, the specific mapping method is:

ht_int_stripe_1:

[0,1,2,3 ...... 63] Corresponding to interrupt line 0 / HT HI Corresponding to interrupt line 4

[64,65,66,67 ... 127] Corresponding to interrupt line 1 / HT HI Corresponding to interrupt line 5

[128,129,130,131 ... 191] Corresponding to interrupt line 2 / HT HI Corresponding to interrupt line 6

[192,193,194,195 ... 255] corresponds to interrupt line 3 / HT HI corresponds to interrupt line 7

ht_int_stripe_2:

[0,2,4,6 ...... 126] Corresponding to interrupt line 0 / HT HI Corresponding to interrupt line 4

[1,3,5,7 ... 127] corresponds to interrupt line 1 / HT HI corresponds to interrupt line 5

[128,130,132,134 ... 254] Corresponding to interrupt line 2 / HT HI Corresponding to interrupt line 6

[129,131,133,135 ... 255] corresponds to interrupt line 3 / HT HI corresponds to interrupt line 7

ht_int_stripe_4:

[0,4,8,12 ... 252] corresponds to interrupt line 0 / HT HI corresponds to interrupt line 4

[1,5,9,13 ... 253] corresponds to interrupt line 1 / HT HI corresponds to interrupt line 5

[2,6,10,14 ... 254] corresponds to interrupt line 2 / HT HI corresponds to interrupt line 6

[3,7,11,15 ... 255] corresponds to interrupt line 3 / HT HI corresponds to interrupt line 7

The following description of the interrupt vector corresponds to ht_int_stripe_1, and the other two methods can be obtained from the above description.

For LS3A1000D and below, only ht_int_stripe_1 can be used.

Offset: 0x80
Reset value: 0x00000000
name: HT bus interrupt vector register [31: 0]

Table 10-27 HT bus interrupt vector register definition (1)

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	Interrupt_case [31: 0]	32	0x0	R / W	HT bus interrupt vector register [31: 0], Corresponding to interrupt line 0 / HT HI Corresponding to interrupt line 4

Offset: 0x84
Reset value: 0x00000000
name: HT Bus Interrupt Vector Register [63:32]

Table 10-28 Definition of HT Bus Interrupt Vector Register (2)

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	Interrupt_case [63:32]	32	0x0	R / W	HT bus interrupt vector register [63:32], Corresponding to interrupt line 0 / HT HI Corresponding to interrupt line 4

Offset: 0x88
Reset value: 0x00000000

name: HT Bus Interrupt Vector Register [95:64]

Table 10-29 HT Bus Interrupt Vector Register Definition (3)

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	Interrupt_case [95:64]	32	0x0	R / W	HT bus interrupt vector register [95:64], Corresponding to interrupt line 1 / HT HI Corresponding to interrupt line 5

Offset: 0x8c
Reset value: 0x00000000
name: HT bus interrupt vector register [127: 96]

Table 10- 30 HT Bus Interrupt Vector Register Definition (4)

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	Interrupt_case [127: 96]	32	0x0	R / W	HT bus interrupt vector register [127: 96], Corresponding to interrupt line 1 / HT HI Corresponding to interrupt line 5

Offset: 0x90
Reset value: 0x00000000
name: HT bus interrupt vector register [159: 128]

Table 10-31 HT bus interrupt vector register definition (5)

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	Interrupt_case [159: 128]	32	0x0	R / W	HT bus interrupt vector register [159: 128], Corresponding to interrupt line 2 / HT HI Corresponding to interrupt line 6

Offset: 0x94
Reset value: 0x00000000
name: HT Bus Interrupt Vector Register [191: 160]

Table 10-31 HT bus interrupt vector register definition (6)

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	Interrupt_case [191: 160]	32	0x0	R / W	HT bus interrupt vector register [191: 160], Corresponding to interrupt line 2 / HT HI Corresponding to interrupt line 6

Offset: 0x98
Reset value: 0x00000000
name: HT Bus Interrupt Vector Register [223: 192]

Table 10-32 HT Bus Interrupt Vector Register Definition (7)

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	Interrupt_case [223: 192]	32	0x0	R / W	HT bus interrupt vector register [223: 192], Corresponding to interrupt line 3 / HT HI Corresponding to interrupt line 7

Offset: 0x9c
Reset value: 0x00000000
name: HT Bus Interrupt Vector Register [255: 224]

Table 10-33 Definition of HT Bus Interrupt Vector Register (8)

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	Interrupt_case [255: 224]	32	0x0	R / W	HT bus interrupt vector register [255: 224], Corresponding to interrupt line 3 / HT HI Corresponding to interrupt line 7

10.5.9 Interrupt enable register

A total of 256 interrupt enable registers correspond to the interrupt vector registers. Set to 1 to enable the corresponding interrupt, set to 0

It is an interrupt mask.

The 256 interrupt vectors are mapped to different interrupt lines according to the different register configurations of the interrupt routing method.

Is mapped as:

ht_int_stripe_1:

[0,1,2,3 …… 63] Corresponding to interrupt line 0 / HT HI Corresponding to interrupt line 4

[64,65,66,67 ... 127] Corresponding to interrupt line 1 / HT HI Corresponding to interrupt line 5

[128,129,130,131 ... 191] Corresponding to interrupt line 2 / HT HI Corresponding to interrupt line 6

[192,193,194,195 ... 255] corresponds to interrupt line 3 / HT HI corresponds to interrupt line 7

ht_int_stripe_2:

[0,2,4,6 …… 126] Corresponding to interrupt line 0 / HT HI Corresponding to interrupt line 4

[1,3,5,7 ... 127] corresponds to interrupt line 1 / HT HI corresponds to interrupt line 5

[128,130,132,134 ... 254] Corresponding to interrupt line 2 / HT HI Corresponding to interrupt line 6

[129,131,133,135 ... 255] corresponds to interrupt line 3 / HT HI corresponds to interrupt line 7

ht_int_stripe_4:

[0,4,8,12 ... 252] corresponds to interrupt line 0 / HT HI corresponds to interrupt line 4

[1,5,9,13 ... 253] corresponds to interrupt line 1 / HT HI corresponds to interrupt line 5

[2,6,10,14 ... 254] corresponds to interrupt line 2 / HT HI corresponds to interrupt line 6

[3,7,11,15 ... 255] corresponds to interrupt line 3 / HT HI corresponds to interrupt line 7

The following description of the interrupt vector corresponds to ht_int_stripe_1, and the other two methods can be obtained from the above description.

Offset: 0xa0
Reset value: 0x00000000
name: HT bus interrupt enable register [31: 0]

Table 10-34 Definition of HT Bus Interrupt Enable Register (1)

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	Interrupt_mask [31: 0]	32	0x0	R / W	HT bus interrupt enable register [31: 0], Corresponding to interrupt line 0 / HT HI Corresponding to interrupt line 4

Offset: 0xa4
Reset value: 0x00000000
name: HT bus interrupt enable register [63:32]

Table 10-35 Definition of HT Bus Interrupt Enable Register (2)

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	Interrupt_mask [63:32]	32	0x0	R / W	HT bus interrupt enable register [63:32], Corresponding to interrupt line 0 / HT HI Corresponding to interrupt line 4

Offset: 0xa8
Reset value: 0x00000000
name: HT bus interrupt enable register [95:64]

Table 10- 36 HT Bus Interrupt Enable Register Definition (3)

Bit field	Bit field name	Bit width	Reset value	access	description
31: 0	Interrupt_mask [95:64]	32	0x0	R / W	HT bus interrupt enable register [95:64], Corresponding to interrupt line 1 / HT HI Corresponding to interrupt line 5

Offset: 0xac
Reset value: 0x00000000
name: HT bus interrupt enable register [127: 96]

Table 10-37 HT Bus Interrupt Enable Register Definition (4)

Bit field	Bit field name	Bit width	Reset value	access	description

31: 0	Interrupt_mask [127: 96]	32	0x0	R / W	HT bus interrupt enable register [127: 96], Corresponding to interrupt line 1 / HT HI Corresponding to interrupt line 5

Offset: 0xb0

Reset value: 0x00000000

name: HT bus interrupt enable register [159: 128]

Table 10-38 Definition of HT Bus Interrupt Enable Register (5)

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	Interrupt_mask [159: 128]	32	0x0	R / W	HT bus interrupt enable register [159: 128], Corresponding to interrupt line 2 / HT HI Corresponding to interrupt line 6

Offset: 0xb4

Reset value: 0x00000000

name: HT bus interrupt enable register [191: 160]

Table 10-39 Definition of HT Bus Interrupt Enable Register (6)

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	Interrupt_mask [191: 160]	32	0x0	R / W	HT bus interrupt enable register [191: 160], Corresponding to interrupt line 2 / HT HI Corresponding to interrupt line 6

Offset: 0xb8

Reset value: 0x00000000

name: HT bus interrupt enable register [223: 192]

Table 10- 40 HT bus interrupt enable register definition (7)

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	Interrupt_mask [223: 192]	32	0x0	R / W	HT bus interrupt enable register [223: 192], Corresponding to interrupt line 3 / HT HI Corresponding to interrupt line 7

Offset: 0xbc

Reset value: 0x00000000

name: HT bus interrupt enable register [255: 224]

Table 10- 41 HT Bus Interrupt Enable Register Definition (8)

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	Interrupt_mask [255: 224]	32	0x0	R / W	HT bus interrupt enable register [255: 224], Corresponding to interrupt line 3 / HT HI Corresponding to interrupt line 7

10.5.10 Interrupt Discovery & Configuration

Offset: 0xc0

Reset value: 0x80000008

name: Interrupt Capability

Table 10- 42 Inte upt Capab l ty Register Definition

Bit field	Bit field name	Bit width	reset value	Visit	description
31:24	Capabilities Pointer	8	0x80	R	Interrupt discovery and configuration block
23:16	Index	8	0x0	R / W	Read register offset address
15: 8	Capabilities Pointer	8	0x0	R	Capabilities Pointer
7: 0	Capability ID	8	0x08	R	Hypertransport Capablity ID

Offset: 0xc4
Reset value: 0x00000000
name: Dataport

Table 10- 43 Datap Yaot Register Definition

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	Dataport	32	0x0	R / W	When the previous register Index is 0x10, this register is read and written The result is the 0xa8 register, otherwise 0xac

Offset: 0xc8
Reset value: 0xF8000000
name: IntrInfo [31: 0]

Table 10-44 Definition of Int and Inf Yao Register (1)

Bit field	Bit field name	Bit width	reset value	Visit	description
31:24	IntrInfo [31:24]	32	0xF8	R	Keep
23: 2	IntrInfo [23: 2]	twenty two	0x0	R / W	IntrInfo [23: 2], when the PIC interrupt is issued, the value of IntrInfo Used to represent interrupt vector
1: 0	Reserved	2	0x0	R	Keep

Offset: 0xcc
Reset value: 0x00000000
name: IntrInfo [63:32]

Table 10-45 Int Yao Inf Yao register definition (2)

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	IntrInfo [63:32]	32	0x0	R	Keep

10.5.11 POST address window configuration register

For the address window hit formula, see section 10.5.7.

The address in this window is the address received on the AXI bus. All write accesses that fall in this window will be immediately in AXI B The channel returns and is sent to the HT bus in the format of the POST WRITE command. Instead of writing requests in this window, NONPOST WRITE is sent to the HT bus, and waits for the HT bus to respond before returning to the AXI bus.

Offset: 0xd0
Reset value: 0x00000000
name: HT bus POST address window 0 enable (internal access)

Table 10-46 HT Bus POST Address Window 0 Enable (Internal Access)

Bit field	Bit field name	Bit width	reset value	Visit	description
31	ht_post0_en	1	0x0	R / W	HT bus POST address window 0, enable signal
30	ht_depart0_en	1	0x0	R / W	HT access unpacking enable (corresponding to external CPU core uncache ACC operation window)
29:23	Reserved	14	0x0		Keep
15: 0	ht_post0_trans [39:24]	16	0x0	R / W	HT bus POST address window 0, the translated address [39:24]

Offset: 0xd4

Reset value: 0x00000000

name: HT bus POST address window 0 base address (internal access)

Table 10-47 HT bus POST address window 0 base address (internal access)

Bit field	Bit field name	Bit width	reset value	Visit	description
31:16	ht_post0_base [39:24]	16	0x0	R / W	HT bus POST address window 0, address base address [39:24]
15: 0	ht_post0_mask [39:24]	16	0x0	R / W	HT bus POST address window 0, address masked [39:24]

Offset: 0xd8

Reset value: 0x00000000

name: HT bus POST address window 1 enable (internal access)

Table 10-48 HT Bus POST Address Window 1 Enable (Internal Access)

Bit field	Bit field name	Bit width	reset value	Visit	description
31	ht_post1_en	1	0x0	R / W	HT bus POST address window 1, enable signal
30	ht_depart1_en	1	0x0	R / W	HT access unpacking enable (corresponding to external CPU core uncache ACC operation window)
29:16	Reserved	14	0x0		Keep
15: 0	ht_post1_trans [39:24]	16	0x0	R / W	HT bus POST address window 1, the translated address [39:24]

Offset: 0xdc

Reset value: 0x00000000

name: HT bus POST address window 1 base address (internal access)

Table 10-49 HT bus POST address window 1 base address (internal access)

Bit field	Bit field name	Bit width	reset value	Visit	description
31:16	ht_post1_base [39:24]	16	0x0	R / W	HT bus POST address window 1, address base address [39:24]
15: 0	ht_post1_mask [39:24]	16	0x0	R / W	HT bus POST address window 1, address masked [39:24]

10.5.12 Prefetchable address window configuration register

For the address window hit formula, see section 10.5.7.

The address in this window is the address received on the AXI bus. Only the instruction fetch instructions and CACHE access that fall in this window Is sent to the HT bus, other fetch instructions or CACHE access will not be sent to the HT bus, but will return immediately, if it is a read Command, it will return the corresponding number of invalid read data.

Offset: 0xe0

Reset value: 0x00000000

name: HT bus can be prefetched address window 0 enabled (internal access)

Table 10- 50 HT bus prefetchable address window 0 is enabled (internal access)

Bit field	Bit field name	Bit width	reset value	Visit	description
31	ht_prefetch0_en	1	0x0	R / W	HT bus can prefetch address window 0, enable signal
30:23	Reserved	15	0x0		Keep

15: 0	ht_prefetch0_trans [39:24]	16	0x0	R / W	HT bus can prefetch the address window 0, the translated address [39:24]

Offset: 0xe4

Reset value: 0x00000000

name: HT bus prefetchable address window 0 base address (internal access)

Table 10-51 HT Bus Prefetchable Address Window 0 Base Address (Internal Access)

Bit field	Bit field name	Bit width	reset value	Visit	description
31:16	ht_prefetch0_ base [39:24]	16	0x0	R / W	HT bus can pre-fetch address window 0, address base address [39:24] Bit address
15: 0	ht_prefetch0_ mask [39:24]	16	0x0	R / W	HT bus can prefetch address window 0, address masked [39:24]

Offset: 0xe8

Reset value: 0x00000000

name: HT bus prefetch address window 1 enabled (internal access)

Table 10-52 HT Bus Prefetchable Address Window 1 Enable (Internal Access)

Bit field	Bit field name	Bit width	reset value	Visit	description
31	ht_prefetch1_en	1	0x0	R / W	HT bus can prefetch address window 1, enable signal
30:23	Reserved	15	0x0		Keep
15: 0	ht_prefetch1_ trans [39:24]	16	0x0	R / W	HT bus can pre-fetch address window 1, the translated address [39:24]

Offset: 0xec

Reset value: 0x00000000

name: HT bus prefetch address window 1 base address (internal access)

Table 10-53 HT bus prefetchable address window 1 base address (internal access)

Bit field	Bit field name	Bit width	reset value	Visit	description
31:16	ht_prefetch1_ base [39:24]	16	0x0	R / W	HT bus can prefetch address window 1, address base address [39:24]
15: 0	ht_prefetch1_ mask [39:24]	16	0x0	R / W	HT bus can prefetch address window 1, address masked [39:24]

10.5.13 UNCACHE address window configuration register

For the address window hit formula, see section 10.5.7.

The address in this window is the address received on the HT bus. Read and write commands that fall into this window address will not be sent to SCACHE, will not make the first-level CACHE invalid, but will be sent directly to memory or other address space, that is The read and write commands in this address window will not maintain the CACHE consistency of IO. This window is mainly aimed at some Hitting operations that can increase the efficiency of storage, such as video memory access.

Offset: 0xf0
Reset value: 0x00000000
name: HT bus Uncache address window 0 enable (internal access)

Table 10-54 HT Bus Uncache Address Window 0 Enable (Internal Access)

Bit field	Bit field name	Bit width	reset value	Visit	description
31	ht_uncache0_en	1	0x0	R / W	HT bus uncache address window 0, enable signal
30	ht_uncache0_ trans_en	1	0x0	R / W	HT bus uncache address window 1, mapping enable signal
29: 0	ht_uncache0_ trans [53:24]	16	0x0	R / W	HT bus uncache address window 0, the translated address [53:24]

Offset: 0xf4
Reset value: 0x00000000
name: HT bus Uncache address window 0 base address (internal access)

Table 10-55 HT Bus Uncache Address Window 0 Base Address (Internal Access)

Bit field	Bit field name	Bit width	reset value	Visit	description
31:16	ht_uncache0_ base [39:24]	16	0x0	R / W	HT bus uncache address window 0, address base address [39:24]
15: 0	ht_uncache0_ mask [39:24]	16	0x0	R / W	HT bus uncache address window 0, address masked [39:24]

Offset: 0xf8
Reset value: 0x00000000
name: HT bus Uncache address window 1 is enabled (internal access)

Table 10-56 HT Bus Uncache Address Window 1 Enable (Internal Access)

Bit field	Bit field name	Bit width	reset value	Visit	description
31	ht_uncache1_en	1	0x0	R / W	HT bus uncache address window 1, enable signal
30	ht_uncache1_ trans_en	1	0x0	R / W	HT bus uncache address window 1, mapping enable signal
29: 0	ht_uncache1_ trans [53:24]	16	0x0	R / W	HT bus uncache address window 1, the translated address [53:24]

Offset: 0xfc
Reset value: 0x00000000
name: HT bus Uncache address window 1 base address (internal access)

Table 10-57 HT Bus Uncache Address Window 1 Base Address (Internal Access)

Bit field	Bit field name	Bit width	reset value	Visit	description

31:16	ht_uncache1_ base [39:24]	16	0x0	R / W HT bus uncache address window 1, address base address [39:24]
15: 0	ht_uncache1_ mask [39:24]	16	0x0	R / W HT bus uncache address window 1, address masked [39:24]

Offset: 0x168
Reset value: 0x00000000
name: HT bus Uncache address window 2 enable (internal access)

Table 10-58 HT Bus Uncache Address Window 2 Enable (Internal Access)

Bit field	Bit field name	Bit width	reset value	Visit description
31	ht_uncache1_en	1	0x0	R / W HT bus uncache address window 2, enable signal
30	ht_uncache1_ trans_en	1	0x0	R / W HT bus uncache address window 2, mapping enable signal
29: 0	ht_uncache1_ trans [53:24]	16	0x0	R / W HT bus uncache address window 2, the translated address [53:24]

Offset: 0x16c
Reset value: 0x00000000
name: HT bus Uncache address window 2 base address (internal access)

Table 10-59 HT Bus Uncache Address Window 2 Base Address (Internal Access)

Bit field	Bit field name	Bit width	reset value	Visit description
31:16	ht_uncache1_ base [39:24]	16	0x0	R / W HT bus uncache address window 2, address base address [39:24]

Bit field	Bit field name	Bit width	reset value	Visit description
15: 0	ht_uncache1_ mask [39:24]	16	0x0	R / W HT bus uncache address window 2, address masked [39:24]

Offset: 0x170
Reset value: 0x00000000
name: HT bus Uncache address window 3 enable (internal access)

Table 10- 60 HT Bus Uncache Address Window 3 Enable (Internal Access)

Bit field	Bit field name	Bit width	reset value	Visit description
31	ht_uncache1_en	1	0x0	R / W HT bus uncache address window 3, enable signal
30	ht_uncache1_ trans_en	1	0x0	R / W HT bus uncache address window 3, mapping enable signal
29: 0	ht_uncache1_ trans [53:24]	16	0x0	R / W HT bus uncache address window 3, the translated address [53:24]

Offset: 0x174
Reset value: 0x00000000
name: HT bus Uncache address window 3 base address (internal access)

Table 10-61 HT Bus Uncache Address Window 3 Base Address (Internal Access)

Bit field	Bit field name	Bit width	reset value	Visit description
31:16	ht_uncache1_	16	0x0	R / W HT bus uncache address window 3, address base address

	base [39:24]			
15: 0	ht_uncache1_ mask [39:24]	16	0x0	R / W HT bus uncache address window 3, address masked [39:24]

10.5.14 P2P address window configuration register

For the address window hit formula, see section 10.5.7.

The address in this window is the address received on the HT bus. The read and write commands at the address of this window are directly used as P2P

The command is forwarded back to the bus, which has the highest priority relative to the normal receive window and Uncache window.

Offset: 0x158
Reset value: 0x00000000
name: HT bus P2P address window 0 enable (external access)

Table 10-62 HT Bus P2P Address Window 0 Enable (External Access) Register Definition

Bit field	Bit field name	Bit width	reset value	Visit description
31	ht_rx_image2_en	1	0x0	R / W HT bus P2P address window 0, enable signal
30	ht_rx_image2_ trans_en	1	0x0	R / W HT bus P2P address window 0, mapping enable signal
29: 0	ht_rx_image2_ trans [53:24]	16	0x0	R / W HT bus P2P address window 0, translated address [53:24]

Offset: 0x15c
Reset value: 0x00000000
name: HT bus P2P address window 0 base address (external access)

Table 10-63 HT bus P2P address window 0 base address (external access) register definition

Bit field	Bit field name	Bit width	reset value	Visit description
31:16	ht_rx_image2_ base [39:24]	16	0x0	R / W HT bus P2P address window 1, address base address [39:24]
15: 0	ht_rx_image2_ mask [39:24]	16	0x0	R / W HT bus P2P address window 1, address masked [39:24]

Offset: 0x160
Reset value: 0x00000000
name: HT bus P2P address window 1 enable (external access)

Table 10-64 HT Bus P2P Address Window 1 Enable (External Access) Register Definition

Bit field	Bit field name	Bit width	reset value	Visit description
31	ht_rx_image2_en	1	0x0	R / W HT bus P2P address window 1, enable signal
30	ht_rx_image2_ trans_en	1	0x0	R / W HT bus P2P address window 1, mapping enable signal
29: 0	ht_rx_image2_ trans [53:24]	16	0x0	R / W HT bus P2P address window 1, the translated address [53:24]

Offset: 0x164
Reset value: 0x00000000
name: HT bus P2P address window 1 base address (external access)

Table 10-65 HT bus P2P address window 1 base address (external access) register definition

Bit field	Bit field name	Bit width reset value	Visit description

Bit field	Bit field name	Bit width	reset value	Visit description
31:16	ht_rx_image2_ base [39:24]	16	0x0	R / W HT bus P2P address window 1, address base address [39:24]
15: 0	ht_rx_image2_ mask [39:24]	16	0x0	R / W HT bus P2P address window 1, address masked [39:24]

10.5.15 Command send buffer size register

The command sending buffer size register is used to observe the number of buffers available for each command channel at the sending end.

Offset:	0x100
Reset value:	0x00000000
name:	Command send buffer size register

Table 10-66 Command Send Buffer Size Register

Bit field	Bit field name	Bit width	reset value	Visit	description
31:24	B_CMD_txbuffer	8	0x0	R	Number of B channel command buffers at the sending end
2316	R_CMD_txbuffer	8	0x0	R	Number of R channel command buffers at the sending end
15: 8	NPC_CMD_txbuffer 8		0x0	R	Number of NPC channel command buffers at the sending end
7: 0	PC_CMD_txbuffer	8	0x0	R	Number of PC channel command buffers at the sending end

10.5.16 Data transmission buffer size register

The data transmission buffer size register is used to observe the number of buffers available for each data channel at the sending end.

Offset:	0x104
Reset value:	0x00000000
name:	Data transmission buffer size register

Table 10-67 Data transmission buffer size register

Bit field	Bit field name	Bit width	reset value	access	description
31:24	Reserved	8	0x0	R	Keep
2316	R_DATA_txbuffer	8	0x0	R	Number of R channel data buffers at the sending end
15: 8	NPC_DATA_txbuffer 8		0x0	R	Number of NPC channel data buffers at the sending end
7: 0	PC_DATA_txbuffer	8	0x0	R	Number of PC channel data buffers at the sending end

10.5.17 Send buffer debug register

Send buffer debugging register is used to manually set the number of buffers at the sending end of the HT controller.

Adjust the number of different send buffers.

Offset: 0x108

Reset value: 0x00000000

name: Send buffer debug register

Table 10-68 Send Buffer Debug Register

Bit field	Bit field name	Bit width	reset value	Visit	description
31:30	Reserved	2	0x0	R	Keep
29	Tx_neg	1	0x0	R / W	Debugging symbols are cached on the sending end 0: increase the corresponding number 1: Reduce (number of corresponding registers + 1)
28	Tx_buff_adj_en	1	0x0	R / W	Buffer debugging enable register on the sending end 0-> 1: make the value of this register increase and decrease
27:24	R_DATA_txadj	4	0x0	R / W sender	R channel data buffer increase and decrease When tx_neg is 0, increase R_DATA_txadj; When tx_neg is 1, reduce R_DATA_txadj + 1
23:20	NPC_DATA_txadj	4	0x0	R / W sender	NPC channel data cache increase and decrease When tx_neg is 0, increase NPC_DATA_txadj; When tx_neg is 1, reduce NPC_DATA_txadj + 1
19:16	PC_DATA_txadj	4	0x0	R / W sender	Increase or decrease the number of PC channel data buffers When tx_neg is 0, add PC_DATA_txadj; When tx_neg is 1, reduce PC_DATA_txadj + 1
15:12	B_CMD_txadj	4	0x0	R / W sender	Increase and decrease the number of B channel command cache When tx_neg is 0, increase B_CMD_txadj; When tx_neg is 1, reduce B_CMD_txadj + 1
11: 8	R_CMD_txadj	4	0x0	R / W sender	R channel command cache increase or decrease the number When tx_neg is 0, increase R_CMD_txadj; When tx_neg is 1, reduce R_CMD_txadj + 1
7: 4	NPC_CMD_txadj	4	0x0	R / W sender	NPC channel command / data cache increase and decrease When tx_neg is 0, increase NPC_CMD_txadj; When tx_neg is 1, reduce NPC_CMD_txadj + 1
3: 0	PC_CMD_txadj	4	0x0	R / W sender	PC channel command cache increase and decrease When tx_neg is 0, increase PC_CMD_txadj; When tx_neg is 1, reduce PC_CMD_txadj + 1

10.5.18 PHY impedance matching control register

Used to control the impedance matching enable of the PHY, and set the impedance matching parameters at the transmitter and receiver

Offset: 0x10C

Reset value: 0x00000000

name: PHY impedance matching control register

Table 10-69 Impedance Matching Control Register

Bit field	Bit field name	Bit width	reset value	Visit	description
31	Tx_scanin_en	1	0x0	R / W	TX impedance matching enable
30	Rx_scanin_en	1	0x0	R / W	RX impedance matching enable
27:24	Tx_scanin_ncode	4	0x0	R / W	TX impedance matching scan input ncode
23:20	Tx_scanin_pcode	4	0x0	R / W	TX impedance matching scan input pcode
19:12	Rx_scanin_code	8	0x0	R / W	RX impedance matching scan input

10.5.19 Revision ID register

It is used to configure the controller version and configure it to a new version number, which takes effect through Warm Reset.

Offset: 0x110

Reset value: 0x00200000

name: RevisionID register

Table 10-70 Rev s Yao ID Register

Bit field	Bit field name	Bit width	reset value	Visit	description
31:24	Reserved	8	0x0	R	Keep
23:16	Revision ID	8	0x20	R / W	Revision ID control register 0x20: HyperTransport 1.00 0x60: HyperTransport 3.00
15: 0	Reserved	16	0x0	R	Keep

10.5.20 Error Retry Control Register

Used to enable error retransmission in HyerTransport 3.0 mode, configure the maximum number of Short Retry, display

Whether the Retry counter rolls over.

Offset: 0x118

Reset value: 0x00000000

name: Error Retry Control Register

Table 10-71 E Reach Ret Rey Control Register

Bit field	Bit field name	Bit width	reset value	Visit	description
31:10	Reserved	twenty two	0x0	R	Keep
9	Retry Count Rollover	1	0x0	R	Retry counter count rollover
8	Reserved	1	0x0	R	Keep
7: 6	Short Retry Attempts	2	0x0		The maximum number of Short Retry allowed by R / W

10.5.21 Retry Count register

Used for error retransmission count in HyerTransport 3.0 mode.

Offset: 0x11C

Reset value: 0x00000000

name: Retry Count register

Table 10- 72 Ret y C Yaount Register

Bit field	Bit field name	Bit width	reset value	Visit	description
31:20	Reserved	12	0x0	R	Keep
19:16	Rrequest delay	4	0x0	R / W	is used to control the random transmission of Rrequest transmission in consistency mode. Machine delay range 000: 0 Delay 001: Random delay 0-8 010: Random delay 8-15 011: Random delay 16-31 100: Random delay 32-63 101: Random delay 64-127 110: Random delay 128-255

					111: 0 Delay
15: 0	Retry Count	16	0x0	R	Retry count

10.5.22 Link Train Register

HyperTransport 3.0 link initialization and link training control register.

Offset: 0x130

Reset value: 0x00000070

name: Link Train Register

Table 10- 73 L nk T 偌 an Register

Bit field	Bit field name	Bit width	reset value	access	description
31:23	Reserved	9	0x0	R	Keep
22:21	Transmitter LS select 2		0x0	R / W	The sending end is at Disconnected or Inactive Link status: 2'b00 LS1 2'b01 LS0 2'b10 LS2 2'b11 LS3
14	Dsiable Cmd Throttling	1	0x0	R / W in	In HyperTransport 3.0 mode, any 4 by default Only one Non-info CMD can appear in consecutive DWS; 1'b0 Enable Cmd Throttling 1'b1 Disable Cmd Throttling
13:10	Reserved	4	0x0	R	Keep
8: 7	Receiver LS select	2	0x0		The R / W receiver is at Disconnected or Inactive Link status: 2'b00 LS1 2'b01 LS0 2'b10 LS2 2'b11 LS3
6: 4	Long Retry Count	3	0x7	R / W	Long Retry
3	Scrambling Enable	1	0x0		Whether R / W is enabled Scramble 0: Disable Scramble 1: enable Scramble
2	8B10B Enable	1	0x0		Whether R / W is enabled 8B10B 0: Disable 8B10B 1: enable 8B10B
1	AC	1	0x0	R	Whether AC mode is detected 0: AC mode is not detected 1: AC mode detected
0	Reserved	1	0x0	R	Keep

10.5.23 Training 0 timeout short timer register

It is used to configure Training 0 short-time timeout threshold in HyerTransport 3.0 mode, the counter clock frequency is HyperTransport3.0 link bus clock frequency is 1/4.

Offset: 0x134

Reset value: 0x00000080

name: Training 0 timeout short count register

Table 10-74 T an ng 0 Time-out short timer register

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	T0 time	32	0x8	R / W	Training 0 Timeout short timer register

10.5.24 Training 0 Time-out timer register

Used for Training 0 long counting timeout threshold in HyerTransport 3.0 mode, the counter clock frequency is

HyperTransport3.0 link bus clock frequency is 1/4.

Offset: 0x138
Reset value: 0x000fffff
name: Training 0 timeout long count register

Table 10-75 T an ng 0 timeout long count register

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	T0 time	32	0xfffff	R / W	Training 0 Time-out long count register

10.5.25 Training 1 count register

Used in Training 1 counting threshold in HyerTransport 3.0 mode, the counter clock frequency is

HyperTransport3.0 link bus clock frequency is 1/4.

Offset: 0x13C
Reset value: 0x0004fffff
name: Training 1 count register

Table 10- 76 T an ng 1 count register

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	T1 time	32	0x4fffff	R / W	Training 1 count register

10.5.26 Training 2 count register

Used in Training 2 counting threshold in HyerTransport 3.0 mode, the counter clock frequency is

HyperTransport3.0 link bus clock frequency is 1/4.

Offset: 0x144
Reset value: 0x0007fffff
name: Training 2 count register

Table 10- 77 T anan ng 2 count register

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	T2 time	32	0x7fffff	R / W	Training 2 count register

10.5.27 Training 3 count register

Used in Training 3 counting threshold in HyerTransport 3.0 mode, the counter clock frequency is

HyperTransport3.0 link bus clock frequency is 1/4.

Offset: 0x13C
name: Training 3 count register

Table 10- 78 T anan ng 3 count register

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 0	T3 time	32	0x7fffff	R / W	Training 3 Count register

10.5.28 Software frequency configuration register

It is used to switch the controller to the link frequency and controller frequency supported by any protocol and PLL during the working process;

The specific switching method is: on the premise of enabling the software configuration mode, set bit 1 of the software frequency configuration register, and

Write parameters related to the new clock, including div_refc and div_loop that determine the output frequency of the PLL

Frequency coefficients phy_hi_div and phy_lo_div, and the frequency division coefficient core_div of the controller. Then enter the warm

reset or LDT disconnect, the controller will automatically reset the PLL and configure new clock parameters.

The calculation formula of the clock frequency is:

HyperTransport 1.0:

PHY_LINK_CLK = 50MHz × div_loop / div_refc / phy_div

HT_CORE_CLK = 100MHz × div_loop / div_refc / core_div

HyperTransport 3.0:

PHY_LINK_CLK = 100MHz × div_loop / div_refc

HT_CORE_CLK = 100MHz × div_loop / div_refc / core_div

The time to wait for the PLL to relock is about 30us by default when the system clk is 33M;

Write a custom upper limit of wait count in the memory;

Offset: 0x178
Reset value: 0x00000000
name: Software frequency configuration register

Table 10-79 Software Frequency Configuration Register

Bit field	Bit field name	Bit width	reset value	Visit	description
31:27	PLL relock counter	5	0x0	R / W	counter upper limit configuration register, set counter select , The upper limit of the counter is {PLL_relock_counter, 5'h1f}, otherwise the upper limit of count 10'3ff
26	Counter select	1	0x0	R / W	Lock timer custom enable: 1'b0 uses the default upper counting limit; 1'b1 is calculated by PLL_relock_counter
25: 22	Soft_phy_lo_div	4	0x0	R / W	High PHY Divider
21: 18	Soft_phy_hi_div	4	0x0	R / W	Low PHY Divider
17: 16	Soft_div_refc	2	0x0	R / W	PLL internal frequency division factor

15: 9	Soft_div_loop	7	0x0	R / W	PLL internal frequency multiplication factor
8: 5	Soft_core_div	4	0x0	R / W	controller clock division factor
4: 2	Reserved	3	0x0	R	Keep
1	Soft cofig enable	1	0x0	R / W	Software configuration enable bit 1'b0 disable software frequency configuration 1'b1 Enable software frequency configuration
0	Reserved	1	0x0	R	Keep

10.5.29 PHY Configuration Register

Used to configure PHY related physical parameters. When the controller is used as two independent 8bit controllers, the upper PHY

And the lower PHY are independently controlled by the two controllers; when the controller acts as a 16bit controller, the upper and lower

The configuration parameters of the lower PHY are controlled by the lower controller;

Offset: 0x17C
Reset value: 0x83308000
name: PHY configuration register

Table 10- 80 PHY Configuration Register

Bit field	Bit field name	Bit width	reset value	Visit	description
31	Rx_ckpll_term	1	0x1	R / W	PLL to RX end on-chip transmission line termination impedance
30	Tx_ckpll_term	1	0x0	R / W	PLL to TX terminal on-chip transmission line termination impedance
29	Rx_clk_in_sel_	1	0x0	R / W	Clock PAD Clock selection for data PAD, HT1 Automatically select CLKPAD in mode: 1'b0 external clock source 1'b1 PLL clock
28	Rx_ckdll_sell	1	0x0	R / W	is used to lock DLL clock selection: 1'b0 PLL clock 1'b1 external clock source
27:26	Rx_ctle_bitc	2	0x0	R / W	PAD EQD high frequency gain
25:24	Rx_ctle_bitr	2	0x3	R / W	PAD EQD low frequency gain
23:22	Rx_ctle_bitlim	2	0x0	R / W	PAD EQD compensation limit
twenty one	Rx_en_ldo	1	0x1	R / W	LDO control 1'b0 LDO disabled 1'b1 LDO enable
20	Rx_en_by	1	0x1	R / W	BandGap control 1'b0 BandGap disabled 1'b1 BandGap enable
19: 17	Reserved	3	0x0	R	Keep
16:12	Tx_preenmp	5	0x08	R / W	PAD pre-emphasis control signal
11: 0	Reserved	12	0x0	R	Keep

10.5.30 Link initialization debug register

Used to configure whether to use the CDR provided by the PHY during the link initialization process in HyperTransport 3.0 mode

The lock signal is used as the link CDR completion flag; if the lock signal is ignored, the controller needs to count and wait

By default, the default CDR is completed.

Offset: 0x180

Reset value: 0x00000000

name: Link initialization debug register

Table 10-81 Link Initialization Debug Register

Bit field	Bit field name	Bit width	reset value	access	description
15	Cdr_ignore_enable 1		0x0	R / W	Whether to ignore the CRC lock during link initialization and pass the counter Wait for the count to complete: 1'b0 wait for CDR lock 1'b1 Ignore the CDR lock signal and wait through the counter
14:00	Cdr_wait_counter	15	0x0	R / W	Waiting for the upper limit of the counter count, based on the technology of the controller clock completion

10.5.31 LDT debug register

After the software changes the controller frequency, the timing of the LDT reconnect phase will be inaccurate, and the counter needs to be configured.

After the frequency is configured as software, the time between the LDT signal being invalid and the controller starting link initialization, the timing is based on the control

Clock.

Offset: 0x184

Reset value: 0x00000000

Page 102

name: LDT debug register

Table 10-82 LDT debug registers

Bit field	Bit field name	Bit width	reset value	Visit	description
31:16	Rx_wait_time	16	0x0	R / W	RX terminal waits for the initial value of the counter
15: 0	Tx_wait_time	16	0x0	R / W	TX terminal waits for the initial value of the counter

10.6 Access method of HyperTransport bus configuration space

The protocol of the HyperTransport interface software layer is basically the same as the PCI protocol. Since the access to the configuration space is directly

The underlying protocol is related, and the specific access details are slightly different. As listed in Table 10-5, the address range of the HT bus configuration space

The range is 0xFD_FE00_0000 to 0xFD_FFFF_FFFF. For configuration access in the HT protocol, the Godson 3A3000 / 3B3000

It is implemented in the following format:

Type 0:

Type 1:

Figure 10-1 Configuration access of HT protocol in Loongson 3A3000 / 3B3000

10.7 HyperTransport multiprocessor support

Loongson No. 3 processor uses HyperTransport interface for multi-processor interconnection and can be automatically maintained by hardware

Consistency request between 4 chips. The following provides two multiprocessor interconnection methods:

Four piece Loongson No. 3 interconnection structure

The four CPUs are connected in pairs to form a ring structure. Each CPU uses two 8-bit controllers of HT0 to connect with two adjacent chips,

Among them, HTx_LO is the master device, and HTx_HI is the slave device, and the interconnection structure as shown below is obtained:

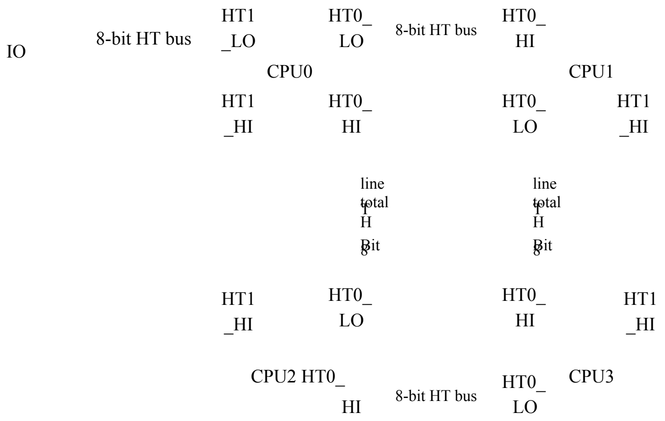


Figure 10- 2 Four-chip Loongson No. 3 interconnection structure

Loongson 3 interconnection routing

Loongson No. 3 interconnection routing adopts simple XY routing method. When routing, X followed by Y, taking four chips as an example, ID

The numbers are 00, 01, 10, and 11, respectively. If you send a request from 11 to 00, it is a route from 11 to 00, first go in the X direction,

Go from 11 to 10, then go in Y direction, and go from 10 to 00. And when the response of the request returns from 00 to 11, the routing first goes

X direction, from 00 to 01, and then Y direction, from 01 to 11. As you can see, these are two different routing lines. by

Due to the characteristics of this algorithm, we will adopt different methods when constructing the interconnection of two chips.

Two piece Loongson No. 3 interconnection structure

Due to the nature of the fixed routing algorithm, we have two different methods when constructing the interconnection of two chips. The first is to adopt

Use 8-bit HT bus interconnection. In this interconnection method, only 8-bit HT interconnection can be used between the two processors. Two chip numbers

They are 00 and 01 respectively. From the routing algorithm, we can know that when two chips access each other, they are interconnected by four chips.

Consistent 8-bit HT bus. As follows:

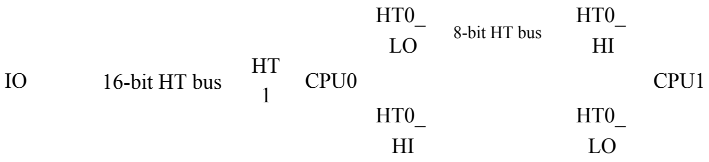


Figure 10--3 Two-chip Loongson No. 3 8-bit interconnection structure

However, our widest HT bus can use 16-bit mode, so the connection method to maximize bandwidth should be adopted

16-bit interconnect structure. In Godson III, as long as the HT0 controller is set to 16-bit mode, all are sent to the HT0 controller

Will be sent to HT0_LO instead of HT0_HI or HT0_LO according to the routing table.

We can use the 16-bit bus when interconnecting. Therefore, we only need to correctly configure the 16-bit mode of CPU0 and CPU1

You can use the 16-bit HT bus interconnection to set and connect the high and low bus correctly. And this interconnect structure can also be used 8

Bit HT bus protocol for mutual access. The resulting interconnection structure is as follows:

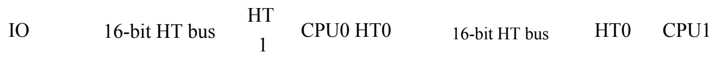


Figure 10--4 Two-chip Loongson No. 3 16-bit interconnection structure

11 Low-speed IO controller configuration

Loongson No. 3 I / O controller includes PCI controller, LPC controller, UART controller, SPI controller, GPIO

And configuration registers. These I / O controllers share an AXI port, and the CPU request is sent to the phase after address decoding

Should be the equipment.

11.1 PCI controller

The PCI controller of Loongson 3 can be used as the main bridge to control the entire system, or it can work as a common PC device. On the PCI bus. Its implementation conforms to the PCI 2.3 specification. The PCI controller of Godson 3 also has a built-in PCI arbiter.

The configuration header of the PCI controller is located at 256 bytes starting at 0x1FE00000, as shown in Table 11-1.

Table 11- 1 PCI controller configuration header

Byte 3	Byte 2	Byte 1	Byte 0	address
Dev ce ID		Vend Yao ID		00
Status		C Yao mmand		04
Class C Yaode			Rev s Yao n ID	08
BIST	Heade Type	Latency T me	CacheL ne S ze	0C
Base Add ess Reg ste 0				10
Base Add ess Reg ste 1				14
Base Add ess Reg ste 2				18
Base Add ess Reg ste 3				1C
Base Add ess Reg ste 4				20
Base Add ess Reg ste 5				twenty four
				28
Subsystem ID		Subsystem Vend Yao ID		2C
				30
			Capab lt es P Yao nte	34
				38
Max mum Latency	M n mum G ant	Inte upt P n	Inte upt L ne	3C
Implementat Yao n Spec fc Reg ste (ISR40)				40
Implementat Yaon Spec fc Reg ste (ISR44)				44
Implementat Yao n Spec fc Reg ste (ISR48)				48
Implementat Yao n Spec fc Reg ste (ISR4C)				4C
Implementat Yaon Spec fc Reg ste (ISR50)				50
Implementat Yao n Spec fc Reg ste (ISR54)				54
Implementat Yaon Spec fc Reg ste (ISR58)				58
				...
PCIX C Yaommand Reg ste				E0
PCIX Status Reg ste				E4

The PCIX controller of Loongson 3A3000 / 3B3000 supports three 64-bit windows, composed of {BAR1, BAR0}, {BAR3, BAR2}, {BAR5, BAR4} The base address of three pairs of register configuration windows 0, 1, 2. The size, enable, and other details of the window Three corresponding registers PCI_Hit0_Sel, PCI_Hit1_Sel, PCI_Hit2_Sel control, please refer to Table 2 for specific bit fields

Table 11- 2 PCI Control Register

Bit field	Field name	access	Reset value	Explanation
REG_40				
31	ta _ ead_ Yao	Read and write (Write 1 clear)	0	ta get end received access to IO or unprefetched area
30	ta _ ead_d sca d	Read and write (Write 1 clear)	0	The delay request on the get side of ta is discarded
29	ta _ esp_delay	Read and write	0	ta get When to give access delay / spl t 0: After timeout 1: right away
				ta get access retry strategy

28	ta _delay_ et y	Read and write	0	0: According to internal logic (see bit 29) 1: Retry now
27	ta _ ead_ab 耀 t_en	Read and write	0	If ta_get timeout for internal read request, whether to let ta_get-ab Yao-t respond
26:25	Rese ved	-	0	
twenty four	ta _w te_ab 耀 t_en	Read and write	0	If ta get times out for internal write request, whether to let ta get-ab Yao t respond
twenty three	ta _maste _ab 耀 t	Read and write	0	Whether to allow maste-ab Yaot
22:20	ta _subseq_t me Yaout	Read and write	000	ta get subsequent delay timeout 000: 8 cycles Other: Not supported
19:16	ta _n t_t me Yaout	Read and write	0000	ta get initial delay timeout In PCI mode 0: 16 cycles 1-7: Disable counter 8-15: 8-15 cycle In PCIX mode, the timeout count is fixed at 8 cycles. delay visits 0: 8 delay access 8: 1 delay access 9: 2 delay visit 10: 3 delay visit 11: 4 delay visit

				12: 5 delay visit 13: 6 delay visit 14: 7 delay visit 15: 8 delay visit
15: 4	ta _p ef_b Yao unda y	Read and write	000h	Prefetchable boundary configuration (in units of 16 bytes) FFF: 64KB to 16byte FFE: 64KB to 32byte FF8: 64KB to 128byte
3	ta _p ef_b 耀 und_en	Read and write	0	Use ta _p ef_b 耀 unda y configuration 0: prefetch to device boundary 1: Use ta _p ef_b 耀 unda y
2	Rese	-	0	
1	ta _spl tw_ct l	Read and write	0	ta get spl t write control 0: Block access except P Yao sted Mem Yao Cheng y W Cheng te 1: Block all access until spl t is completed
0	mas_lat_t meyaut	Read and write	0	Disable mate access timeout 0: Allow maste to access timeout 1: not allowed
REG_44				
31: 0	Rese ved	-	-	
REG_48				
31: 0	ta _pend ng_seq	Read and write	0	ta get unprocessed request number vector The corresponding bit can be cleared by writing 1
REG_4C				
31:30	Rese ved	-	-	
29	mas_w te_defe	Read and write	0	Allow subsequent reads to skip past unfinished writes (Only valid for PCI)
28	mas_ ead_defe	Read and write	0	Allow subsequent reads and writes to bypass previous unfinished reads (Only valid for PCI)
27	mas_ 耀 _defe _cnt	Read and write	0	Maximum number of IO requests out 0: controlled by 1: 1

26:24	mas_ ead_defe _cnt	Read and write	010	The maximum number of maste to support reading outside (only valid for PCI) 0: 8 1-7: 1-7 Note: A dual address cycle access accounts for two
23:16	e _seq_ d	Read only	00h	ta, get / maste, error number
15	e _type	Read only	0	ta 偡 get / maste erroneous command type 0:
14	e _m 耀 dule	Read only	0	The wrong module

				0: ta get 1: maste
13	system_e 耀	Read and write	0	ta / get / maste / system error (write 1 clear)
12	data_pa ty_e 耀	Read and write	0	ta / get / maste / data parity error (write 1 clear)
11	ct l_pa ty_e	Read and write	0	ta 偡 get / maste 偡 Address parity error (write 1 clear)
10: 0	Rese ved	-	-	
REG_50				
31: 0	mas_pend ng_seq	Read and write	0	maste unprocessed request number vector The corresponding bit can be cleared by writing 1
REG_54				
31: 0	mas_spl t_e	Read and write	0	spl t returns the wrong request number bit vector
REG_58				
31:30	Rese ved	-	-	
29:28	ta _spl t_p 耀 ty	Read and write	0	ta get spl t return priority 0 highest, 3 lowest
27:26	mas_ eq_p 耀 ty	Read and write	0	maste 0 highest, 3 lowest
25	P 耀 ty_en	Read and write	0	Arbitration algorithm (arbitration between the visit of maste and the return of spl t from ta) 0: fixed priority 1: rotation
24:18	Reserved	-	-	
17	mas_ et y_ab 耀 ted	Read and write	0	maste Retry cancellation (write 1 clear)
16	mas_t dy_t me Yaout	Read and write	0	maste ~ TRDY timeout count
15: 8	mas_ et y_value	Read and write	00h	maste 0: unlimited retry 1-255: 1-255 times
7: 0	mas_t dy_c Yaount	Read and write	00h	maste TRDY timeout counter 0: disabled 1-255: 1-255 beat

Before initiating configuration space read and write, the application program should first configure the PCIMap_Cfg register to tell the controller to initiate

The type of configuration operation and the value on the upper 16-bit address line. Then read and write the 2K space starting from 0x1fe80000

You can access the configuration header of the corresponding device. The device number is obtained by coding according to PCIMap_Cfg [15: 0] from low to high priority.

The configuration operation address generation is shown in Figure 11-1.

Page 109

Figure 11- 1 Configure read and write bus address generation

The PCI arbiter implements two-level round robin arbitration, bus docking, and isolation of damaged master devices. See its configuration and status

PXArb_Config and PXArb_Status registers. See Table 11-3 for the assignment of PCI bus request and response lines.

Table 11- 3 PCI / PCIX bus request and response line assignment

Request and answer line	description
0	Internal integrated PCI / PCIX controller
7: 1	External request 6 ~ 0

The rotation-based arbitration algorithm provides two levels, and the second level as a whole is scheduled as a member of the first level. When multiple

When the device applies for the bus at the same time, the first level device is rotated once, and the device with the highest priority in the second level can get the bus.

The arbiter is designed to be switched at any time as long as conditions permit. For some PCI devices that do not conform to the protocol,

Doing so may make it abnormal. Using mandatory priority allows these devices to occupy the bus through continuous requests.

Bus docking refers to whether or not to select one to give an enable signal when no device requests to use the bus. For already

As far as allowed devices are concerned, directly initiating bus operations can improve efficiency. The internal PCI arbiter provides two docking modes:

The last master device and the default master device. If you cannot dock in special occasions, you can set the arbiter to dock to

The default is No. 0 master device (internal controller), and the delay is 0.

Page 110

11.2 LPC controller

The LPC controller has the following characteristics:

- Comply with LPC1.1 specification
- Support LPC access timeout counter

- Supports Mem Yaoyue Read and Mem Yaoyyw access types
- Support F mwa e Mem Yao y Ready, F mwa e Mem Yao yyy W te Access type (single byte)
- Support I / O ead, I / O w te access type
- Support Mem Yaoyay access type address translation
- Support Se zed IRQ specification, provide 17 interrupt sources

The address space distribution of LPC controller is shown in Table 11-4:

Table 11- 4 LPC Controller Address Space Distribution

Address name	Address range	size
LPC B Yaoyao	0X1FC0_0000-0X1FD0_0000	1MByte
LPC Mem Yao Yi	0X1C00_0000-0X1D00_0000	16MByte
LPC I / O	0X1FF0_0000-0X1FF1_0000	64KByte
LPC Reg ste	0X1FE0_0200-0X1FE0_0300	256Byte

The LPC Boot address space is the address space that the processor first accesses when the system starts. When the PCI_CONFIG [0] pin is

When pulling down, the address of 0xBFC00000 is automatically routed to LPC. This address space supports LPC Memory or Firmware

Memory access type. The type of access issued at system startup is controlled by the LPC_ROM_INTEL pin. LPC_ROM_INTEL

LPC Firmware Memory access is issued when the pin is pulled up, LPC Memory is issued when the LPC_ROM_INTEL pin is pulled down

Type of access.

The LPC Memory address space is the address space accessed by the system with Memory / Firmware Memory. LPC controller

Which type of memory access is issued is determined by the configuration register LPC_MEM_IS_FWH of the LPC controller. The processor sends

Address translation to this address space can be performed. The converted address is controlled by the configuration register of the LPC controller

LPC_MEM_TRANS setting.

The processor's access to the LPC I / O address space is sent to the LPC bus according to the LPC I / O access type. Address is address

The space is 16 bits lower.

There are three 32-bit registers in the LPC controller configuration register. The meaning of the configuration register is shown in Table 11-5:

Table 11- 5 LPC Configuration Register Meaning

Bit field	Field name	Access	reset value	description
		REG0		
REG0 [31:31]	SIRQ_EN	Read and write	0	SIRQ enable control
REG0 [23:16]	LPC_MEM_TRANS	Read and write	0	LPC Mem Yao y space address translation control
REG0 [15: 0]	LPC_SYNC_TIMEOUT	Read and write	0	LPC access timeout counter
		REG1		
REG1 [31:31]	LPC_MEM_IS_FWH	Read and write	0	LPC Mem Yao yy space F y mwa y Mem 耀 y access type settings
REG1 [17: 0]	LPC_INT_EN	Read and write	0	LPC SIRQ interrupt enable
		REG2		
REG2 [17: 0]	LPC_INT_SRC	Read and write	0	LPC SIRQ interrupt source indication
		REG3		

REG3 [17: 0]	LPC_INT_CLEAR	write	0	LPC SIRQ interrupt clear

11.3 UART controller

The UART controller has the following features

- Full-duplex asynchronous data reception / transmission
- Programmable data format
- 16-bit programmable clock counter
- Support receive timeout detection
- Multi-interrupt system with arbitration
- Only work in FIFO mode
- Compatible with NS16550A in registers and functions

The chip integrates two UART interfaces, the function registers are exactly the same, but the access base address is different.

The base address of the physical address of the UART0 register is 0x1FE001E0.

The base address of the physical address of the UART1 register is 0x1FE001E8.

11.3.1 Data Register (**DAT**)

Chinese name: Data transfer register

Register bit width: [7: 0]

Offset: 0x00

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7: 0	Tx FIFO	8	W	Data transfer register

11.3.2 Interrupt enable register (**IER**)

Chinese name: Interrupt enable register

Register bit width: [7: 0]

Offset: 0x01

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7: 4	Rese	4	RW	Keep
3	IME	1	RW	M Yao dem status interrupt enable '0'-off '1'-open
2	ILE	1	RW	Receiver line status interrupt enable '0' – close '1' – open
1	ITxE	1	RW	Transfer save register is empty Interrupt enable '0' – close '1' – open
0	IRxE	1	RW	Receive valid data interrupt enable '0' – close '1' – open

11.3.3 Interrupt Identification Register (**IIR**)

Chinese name: Interrupt source register

Register bit width: [7: 0]

Offset: 0x02

Reset value: 0xc1

Bit field	Bit field name	Bit width	access	description
7: 4	Rese	4	R	Keep
3: 1	II	3	R	Interrupt source display bit, see the table below for details
0	INTp	1	R	Interrupt indication bit

Interrupt control function table

Bit 3	Bit 2	Bit 1	Priority interrupt type		Interrupt source	Interrupt reset control
0	1	1	1st	Receive line status	Parity, overflow, or frame error, or hit Interrupt	Read LSR
0	1	0	2nd	Received valid number according to	The number of characters in the FIFO reaches t 俧 gge 's level	Low number of characters in FIFO The value of t gge
1	1	0	2nd	Receive timeout	There is at least one character in the FIFO, But within 4 character time Operations, including read and write operations	Read receive FIFO
0	0	1	3 tad	Transfer, save, deposit The device is empty	Transfer save register is empty	Write data to THR or Multi IIR
0	0	0	4th	M Yao dem status	CTS, DSR, RI Yaoya DCD.	Read MSR

11.3.4 **FIFO** control register (**FCR**)

Chinese name: FIFO control register

Register bit width: [7: 0]

Offset: 0x02

Reset value: 0xc0

Bit field	Bit field name	Bit width	access	description
7: 6	TL	2	W	Receive tFIFO value of interrupt request from FIFO '00' – 1 byte '01' – 4 bytes '10' – 8 bytes '11' – 14 bytes
5: 3	Rese	3	W	Keep
2	Txset	1	W	'1' Clear the content of transmit FIFO, reset its logic
1	Rxset	1	W	'1' Clear the content of the receive FIFO, reset its logic
0	Rese	1	W	Keep

11.3.5 Line Control Register (**LCR**)

Chinese name: Line control register

Register bit width: [7: 0]

Offset: 0x03

Reset value: 0x03

Bit field	Bit field name	Bit width	access	description
7	dlab	1	RW	Divider latch access bit '1'-access to the operation divider latch '0'-access to normal operation register
6	bcb	1	RW	Interrupt control bit '1'-At this time the output of the serial port is set to 0 (interrupted state). '0'-normal operation
5	spb	1	RW	Specify parity '0' – no parity bit specified '1' – transmission and check parity if LCR [4] bit is 1 The bit is 0. If the LCR [4] bit is 0, transmit and check the parity The checkpoint is 1.
4	eps	1	RW	Parity bit selection '0' – There are an odd number of 1s in each character (including data and odd Even parity bit) '1' – there are an even number of 1s in each character
3	pe	1	RW	Parity bit enable '0' – no parity bit '1'-generate parity bit on output, judge odd on input Even parity
2	sb	1	RW	Define the number of generated stop bits '0' – 1 stop bit '1' – 1.5 stop bits when 5 characters long, others The length is 2 stop bits
1: 0	bec	2	RW	Set the number of digits for each character '00' – 5 digits '01' – 6 digits '10' – 7 digits '11' – 8 digits

11.3.6 **MODEM** control register (**MCR**)

Chinese name: M Yaodem control register

Register bit width: [7: 0]

Offset: 0x04

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7: 5	Rese	3	W	Keep
4	L Yaoyao p	1	W	Loopback mode control bit '0'-normal operation '1' – Loopback mode. In loopback mode, TXD outputs a Straight to 1, the output shift register is directly connected to the input shift register 器 中. The other connections are as follows. DTR → DSR RTS → CTS Out1 → RI Out2 → DCD
3	OUT2	1	W	Connect to DCD input in loopback mode
2	OUT1	1	W	Connect to RI input in loopback mode
1	RTSC	1	W	RTS signal control bit
0	DTRC	1	W	DTR signal control bit

11.3.7 Line Status Register (**LSR**)

Chinese name: Line status register

Register bit width: [7: 0]

Offset: 0x05

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7	ERROR	1	R	Error indication bit '1'-at least parity error, framing error or interruption The broken one. '0' – no errors
6	TE	1	R	Transmission is empty '1' – Both the transmission FIFO and the transmission shift register are empty. give Clear when the transmit FIFO writes data '0' – with data
5	TFE	1	R	Transmit FIFO bit empty representation bit '1' – The current transmit FIFO is empty, write data to the transmit FIFO Time zero '0' – with data

4	BI	1	R	Interrupt interruption bit '1'-Start bit + data + parity bit + stop bit received Is 0, that is interrupted '0'-no interruption
3	FE	1	R	Frame error indication bit '1' – received data has no stop bit '0' – no errors
2	PE	1	R	Parity bit error indicates bit '1'-The current received data has a parity error '0' – no parity error
1	OE	1	R	Data overflow indication bit '1'-There is data overflow '0' – no overflow
0	DR	1	R	Receive data valid representation bit '0' – No data in FIFO '1' – There is data in the FIFO

When reading this register, LSR [4: 1] and LSR [7] are cleared, and LSR [6: 5] is writing data to the transmit FIFO

Cleared according to the time, LSR [0] judges the receive FIFO.

11.3.8 **MODEM** status register (**MSR**)

Chinese name: M Yaodem Status Register
Register bit width: [7: 0]
Offset: 0x06
Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7	CDCD	1	R	Inverse of DCD input value, or connect to Out2 in loopback mode
6	CRI	1	R	Inverse of RI input value, or connect to OUT1 in loopback mode
5	CDSR	1	R	Inverse of DSR input value, or connect to DTR in loopback mode
4	CCTS	1	R	Inverse of CTS input value, or connect to RTS in loopback mode
3	DDCD	1	R	DDCD indicator
2	TERI	1	R	RI edge detection. RI state changes from low to high
1	DDSR	1	R	DDSR indicator
0	DCTS	1	R	DCTS indicator

11.3.9 Frequency divider latch

Chinese name: Divider 1

Register bit width: [7: 0]
Offset: 0x00
Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7: 0	LSB	8	RW	Store the lower 8 bits of the divider latch

Chinese name: Divider 2
Register bit width: [7: 0]
Offset: 0x01
Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7: 0	MSB	8	RW	Stores the upper 8 bits of the divider latch

11.4 SPI controller

The SPI controller has the following features:

- Full duplex synchronous serial data transmission
- Supports up to 4 variable-length byte transfers
- Main mode support
- Mode failure generates an error flag and issues an interrupt request
- Double buffer receiver
- Serial clock with programmable polarity and phase
- SPI can be controlled in wait mode
- Support boot from SPI

The physical address of the SPI controller register is 0x1FE00220.

Table 11- 6 SPI controller address space distribution

Address name	Address range	size
SPI B Yaoyao	0X1FC0_0000-0X1FD0_0000	1MByte
SPI Mem Yao Yi	0X1D00_0000-0X1E00_0000	16MByte
SPI Reg ste	0X1FE0_0220-0X1FE0_0230	16Byte

The SPI Boot address space is the address space that the processor first accesses when the system starts. When the PCI_CONFIG [0] pin is

When pulling up, the address of 0xBFC00000 is automatically routed to the SPI.

The SPI Mem Yao space can also be directly accessed through the CPU's read request, its minimum 1M bytes and SPI BOOT space

overlapping.

11.4.1 Control Register (**SPCR**)

Chinese name: Control register
Register bit width: [7: 0]
Offset: 0x00
Reset value: 0x10

Bit field	Bit field name	Bit width	access	description
7	Sp e	1	RW	Interrupt output enable signal is high and effective
6	spe	1	RW	System work enable signal is highly effective

5	Rese	1	RW	Keep
4	mst	1	RW	maste mode selection bit, this bit keeps 1
3	cp Yaol	1	RW	Clock polarity bit
2	cpha	1	RW	Clock phase bit 1 is the opposite phase, and 0 is the same
1: 0	sp	2	RW	sclk_yao crossover setting, need to be used with spe spe

11.4.2 Status Register (**SPSR**)

Chinese name: Status register
Register bit width: [7: 0]
Offset: 0x01
Reset value: 0x05

Bit field	Bit field name	Bit width	access	description
7	sp f	1	RW	Interrupt flag bit 1 indicates that there is an interrupt request, write 1 to clear
6	wc Yaol	1	RW	Write register overflow flag bit is 1 indicates that it has overflowed, write 1 to clear
5: 4	Rese	2	RW	Keep
3	wffull	1	RW	Write register full flag 1 means full
2	wfempty	1	RW	Write register empty flag 1 means empty
1	Yffull	1	RW	Read register full flag 1 means full
0	Fempty	1	RW	Read register empty flag 1 means empty

11.4.3 Data Register (**TxFIFO**)

Chinese name: Data transfer register
Register bit width: [7: 0]
Offset: 0x02
Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7: 0	Tx FIFO	8	W	Data transfer register

11.4.4 External register (**SPER**)

Chinese name: External register
Register bit width: [7: 0]

Offset: 0x03
Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7: 6	cnt	2	RW	Send an interrupt request signal after how many bytes are transferred 00 – 1 byte 01-2 bytes 10-3 bytes 11-3 bytes
5: 2	Rese	4	RW	Keep
1: 0	sp	2	RW	Set the frequency division ratio together with Sp

Frequency division factor:

spre	00	00	00	00	01	01	01	01	10	10	10	10
spr	00	01	10	11	00	01	10	11	00	01	10	11
Frequency division factor	2	4	16	32	8	64	128	256	512	1024	2048	4096

11.4.5 Parameter control register (SFC_PARAM)

Chinese name: SPI Flash parameter control register
Register bit width: [7: 0]
Offset: 0x04
Reset value: 0x21

Bit field	Bit field name	Bit width	access	description
7: 4	clk_div	4	RW	Clock frequency division number selection (frequency division coefficient is the same as {spre, spr} combination)
3	dual_io	1	RW	Use dual I / O mode with higher priority than fast read mode
2	fast_read	1	RW	Use quick read mode
1	burst_en	1	RW	spi flash supports continuous address read mode
0	memory_en	1	RW	spi flash read enable, when invalid, csn [0] can be controlled by software.

11.4.6 Chip Select Control Register (SFC_SOFTCS)

Chinese name: SPI Flash Chip Select Control Register
Register bit width: [7: 0]
Offset: 0x05
Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7: 4	csn	4	RW	csn pin output value
3: 0	csen	4	RW	When it is 1, the corresponding cs line is controlled by 7: 4 bits

11.4.7 Timing control register (SFC_TIMING)

Chinese name: SPI Flash timing control register
Register bit width: [7: 0]
Offset: 0x06
Reset value: 0x03

Bit field	Bit field name	Bit width	access	description
7: 2	Reserved	6	RW	Keep

1: 0	tCSH	2	RW	The shortest invalid time of the chip select signal of SPI Flash, divided by frequency Clock period T calculation 00: 1T 01: 2T 10: 4T 11: 8T

11.5 IO controller configuration

The configuration register is mainly used to configure the address window, arbiter and GPIO controller of the PCI controller. Table 11- 7

These registers are listed, and Table 11-8 gives a detailed description of the registers. The base address of this part of the register is 0x1FE00100.

Table 11-7 IO Control Register

address	register	Explanation
00	P YaonCfg	Power-on configuration
04	GenCfg	General configuration
08	Keep	
0C	Keep	
10	PCIMap	PCI mapping
14	PCIX_B dge_Cfg	PCI / X bridge related configuration
18	PCIMap_Cfg	PCI configuration read and write device address
1C	GPIO_Data	GPIO data
20	GPIO_EN	GPIO direction
twenty four	Keep	
28	Keep	
2C	Keep	
30	Keep	

34	Keep	
38	Keep	
3C	Keep	
40	Mem_W n_Base_L	Prefetch the lower 32 bits of the base address of the window
44	Mem_W n_Base_H	Pre-fetch window base 32 higher bits
48	Mem_W n_Mask_L	Prefetchable window mask lower 32 bits
4C	Mem_W n_Mask_H	Pre-fetch window mask high 32 bits
50	PCI_H t0_Sel_L	PCI window 0 controls the lower 32 bits

54	PCI_H t0_Sel_H	PCI window 0 controls the upper 32 bits
58	PCI_H t1_Sel_L	PCI Window 1 controls the lower 32 bits
5C	PCI_H t1_Sel_H	PCI Window 1 controls the upper 32 bits
60	PCI_H t2_Sel_L	PCI Window 2 controls the lower 32 bits
64	PCI_H t2_Sel_H	PCI Window 2 controls the upper 32 bits
68	PXA b_C 耀 nf g	PCIX arbiter configuration
6C	PXA b_Status	PCIX arbiter status
70		
74		
78		
7C		
80	Ch p C Yao nf g	Chip configuration register
84		
88		
8C		
90	Ch p Sample	Chip sampling register

Table 11- 8 Register detailed description

Bit field	Field name	access	Reset value	Explanation
CR00: P Yao nCfg				
15: 0	pc x_bus_dev	Read only	l Yao_ad [7: 0]	In PCIX Agent mode, the total CPU usage Line, equipment number
15: 8	Keep	Read only	l Yao_ad [15: 8]	
23:16	p 耀 n_pc _c 耀 nf g	Read only	pc _c 耀 nf g	PCI_C Yao nf g pin value
31:24	Keep	Read only		
CR04: reserved				
31: 0	Keep	Read only	0	
CR08: reserved				
31: 0	Keep	Read only	0	
CR10: PCIMap				
5: 0	t ans_l 耀 0	Read-write	0	PCI_Mem_L Yao 0 Window map address high 6 bits
11: 6	t ans_l 耀 1	Read-write	0	PCI_Mem_L Yao1 Window map address high 6 bits

Page 124

17:12	t ans_l 耀 2	Read-write 0	PCI_Mem_L Yao 2 Window map address high 6 bits
31:18	Keep	Read only 0	

CR14: PCIX_B dge_Cfg

5: 0	pc x_ gate	Read and write 6'h18	Threshold for sending data to DDR2 in PCIX mode
6	pc x_ 耀 _en	Read-write 0	Does the PCIX bridge allow write over read
31:18	Keep	Read only 0	

CR18: PCIMap_Cfg

15: 0	dev_add	Read-write 0	The upper 16 bits of the AD line in PCI configuration
16	c 耀 nf_type	Read-write 0	Configure the type of read and write
31:17	Keep	Read only 0	

CR1C: GPIO_Data

15: 0	gp 耀 _ 耀 ut	Read-write 0	GPIO output data
31:16	gp Yao_n	Read-write 0	GPIO input data

CR20: GPIO_EN

15: 0	gp 耀 _en	Read and write FFFF	High is input, low output
31:16	Keep	Read only 0	

CR3C: reserved

31: 0	Keep	Read only 0	Keep

CR24, 2C, 30, 34, 38: reserved

See table 11-3

CR50,54 / 58,5C / 60,64: PCI_H t * _Sel_ *

0	Keep	Read only 0	
2: 1	pc _ mg_s ze	Read and write 2'b11	00: 32 bits; 10: 64 bits; others: invalid
3	p ef_en	Read-write 0	Prefetch enable
11: 4	Keep	Read only 0	
62:12	ba _mask	Read-write 0	Window size mask (high order 1, low order 0)
63	bu ~ st_cap	Read and write 1	Whether to allow burst transfer

CR68: PXA b_C 耀 nf g

0	dev ce_en	Read and write 1	Permitted by external equipment
1	d sable_b 耀 ken	Read-write 0	Disable damaged master device
2	default_mas_en	Read and write 1	The bus is docked to the default master 0: dock to the last master device 1: dock to the default master device
5: 3	default_maste	Read-write 0	Bus docking default master device number
7: 6	pa k_delay	Read and write 2'b11	Starting from no device requesting the bus to triggering the docking default Delay in device behavior 00: 0 cycles

Page 125

01: 8 cycles

			10: 32 cycles
			11: 128 cycles
15: 8	level	Read and write 8'h01	Equipment in the first level
			Mandatory priority device
23:16	ude_dev	Read-write 0	The PCI device corresponding to the 1 bit can be obtained after the bus
			To occupy the bus with continuous requests
31:13	Keep	Read only 0	
CR6C: PXA b_Status			
7: 0	b 耀 ken_maste	Read only 0	Damaged master device (cleared when changing the disable policy)
10: 8	Last_maste	Read only 0	Last master device using the bus
31:11	Keep	Read only 0	

CR80: Ch pc Yao nf g (see section 2.6)

CR90: Ch p Sample (see section 2.6)

CRA0: Ch p Sample (see section 2.6)

CRB0: PLL c Yao nf g (see section 2.6)

CRC0: PLL c Yao nf g (see section 2.6)

CRD0: C Yao Yi ec Yao nf g (see section 2.6)

12 Chip Configuration Register List

Name	ADDR	R / W	Description (NULL means no effect)	default value
CPU_WIN0_BASE	0x3ff00000	RW	Base address of CPU window 0	0x0
CPU_WIN1_BASE	0x3ff00008	RW	Base address of CPU window 1	0x1000_0000
CPU_WIN2_BASE	0x3ff00010	RW	Base address of CPU window 2	0x1000_8000_0000
CPU_WIN3_BASE	0x3ff00018	RW	Base address of CPU window 3	0x0
CPU_WIN4_BASE	0x3ff00020	RW	Base address of CPU window 4	0x0
CPU_WIN5_BASE	0x3ff00028	RW	Base address of CPU window 5	0x0

CPU_WIN6_BASE	0x3ff00030	RW	Base address of CPU window 6	0x0
CPU_WIN7_BASE	0x3ff00038	RW	Base address of CPU window 7	0x0
CPU_WIN0_MASK	0x3ff00040	RW	Mask of CPU window 0	0xffff_ffff_f000_00
CPU_WIN1_MASK	0x3ff00048	RW	Mask of CPU window 1	0xffff_ffff_f000_00
CPU_WIN2_MASK	0x3ff00050	RW	Mask of CPU window 2	0xffff_ffff_f000_00
CPU_WIN3_MASK	0x3ff00058	RW	Mask of CPU window 3	0x0
CPU_WIN4_MASK	0x3ff00060	RW	Mask of CPU window 4	0x0
CPU_WIN5_MASK	0x3ff00068	RW	Mask of CPU window 5	0x0
CPU_WIN6_MASK	0x3ff00070	RW	Mask of CPU window 6	0x0
CPU_WIN7_MASK	0x3ff00078	RW	Mask of CPU window 7	0x0

Page 127

CPU_WIN0_MMAP	0x3ff00080	RW	New base address of CPU window 0	0xf0
CPU_WIN1_MMAP	0x3ff00088	RW	New base address of CPU window 1	0x1000_00f2
CPU_WIN2_MMAP	0x3ff00090	RW	New base address of CPU window 2	0xf0
CPU_WIN3_MMAP	0x3ff00098	RW	New base address of CPU window 3	0x0
CPU_WIN4_MMAP	0x3ff000a0	RW	New base address of CPU window 4	0x0
CPU_WIN5_MMAP	0x3ff000a8	RW	New base address of CPU window 5	0x0
CPU_WIN6_MMAP	0x3ff000b0	RW	New base address of CPU window 6	0x0
CPU_WIN7_MMAP	0x3ff000b8	RW	New base address of CPU window 7	0x0
PCI_WIN0_BASE	0x3ff00100	RW	Base address of PCI window 0	0x8000_0000
PCI_WIN1_BASE	0x3ff00108	RW	Base address of PCI window 1	0x0
PCI_WIN2_BASE	0x3ff00110	RW	Base address of PCI window 2	0x0
PCI_WIN3_BASE	0x3ff00118	RW	Base address of PCI window 3	0x0
PCI_WIN4_BASE	0x3ff00120	RW	Base address of PCI window 4	0x0
PCI_WIN5_BASE	0x3ff00128	RW	Base address of PCI window 5	0x0
PCI_WIN6_BASE	0x3ff00130	RW	Base address of PCI window 6	0x0
PCI_WIN7_BASE	0x3ff00138	RW	Base address of PCI window 7	0x0
PCI_WIN0_MASK	0x3ff00140	RW	Mask of PCI window 0	0xffff_ffff_8000_00
PCI_WIN1_MASK	0x3ff00148	RW	Mask of PCI window 1	0x0
PCI_WIN2_MASK	0x3ff00150	RW	Mask of PCI window 2	0x0

Page 128

PCI_WIN3_MASK	0x3ff00158	RW	Mask of PCI window 3	0x0
PCI_WIN4_MASK	0x3ff00160	RW	Mask of PCI window 4	0x0
PCI_WIN5_MASK	0x3ff00168	RW	Mask of PCI window 5	0x0

PCI_WIN6_MASK	0x3ff00170	RW	Mask of PCI window 6	0x0
PCI_WIN7_MASK	0x3ff00178	RW	Mask of PCI window 7	0x0
PCI_WIN0_MMAP	0x3ff00180	RW	New base address of PCI window 0	0xf0
PCI_WIN1_MMAP	0x3ff00188	RW	New base address for PCI window 1	0x0
PCI_WIN2_MMAP	0x3ff00190	RW	New base address for PCI window 2	0x0
PCI_WIN3_MMAP	0x3ff00198	RW	New base address for PCI window 3	0x0
PCI_WIN4_MMAP	0x3ff001a0	RW	New base address of PCI window 4	0x0
PCI_WIN5_MMAP	0x3ff001a8	RW	New base address of PCI window 5	0x0
PCI_WIN6_MMAP	0x3ff001b0	RW	New base address of PCI window 6	0x0
PCI_WIN7_MMAP	0x3ff001b8	RW	New base address for PCI window 7	0x0
Slock0_addr	0x3ff00200	RW	Lock address of lock window 0 ([63]: valid, [47: 0]: addr)	0x0
Slock1_addr	0x3ff00208	RW	Lock address of lock window 1 ([63]: valid, [47: 0]: addr)	0x0
Slock2_addr	0x3ff00210	RW	Lock address of lock window 2 ([63]: valid, [47: 0]: addr)	0x0
Slock3_addr	0x3ff00218	RW	Lock address of lock window 3 ([63]: valid, [47: 0]: addr)	0x0
Slock0_mask	0x3ff00240	RW	Lock window mask 0 ([47: 0]: mask)	0x0
Slock1_mask	0x3ff00248	RW	Lock window mask 1 ([47: 0]: mask)	0x0

Page 129

Slock2_mask	0x3ff00250	RW	Lock window mask number 2 ([47: 0]: mask)	0x0
Slock3_mask	0x3ff00258	RW	Lock window mask number 3 ([47: 0]: mask)	0x0
BARRIER_SET	0x3ff00300	WO	barrier value plus 1	
BARRIER_CLR	0x3ff00308	WO	barrier value minus 1	
BARRIER_REF	0x3ff00310	RW	barrier threshold	0x0
BARRIER_CTRL	0x3ff00318	RW	bit [0]: barrier value addition / subtraction enable / barrier interrupt enable	0x0
BARRIER_VEC	0x3ff00320	RO	Current barrier value	
			24: ccsd_en	
			19:16: ccsd_id	
			8: xrouter_en	
			5: x2_pci_rdinterleave	
			4: x2_cpu_rdinterleave	
CONFSIGNAL_CR	0x3ff00400	RW	3: 0: scid_sel	0xffff_0000
gs3_HPT	0x3ff00408	RO	Counter incremented by 1 every clock cycle	
MTX0_SRC_START_ADDR	0x3ff00600	RW		0x0
MTX0_DST_START_ADDR	0x3ff00608	RW		0x0
MTX0_ORI_LENTH	0x3ff00610	RW		0x0
MTX0_ORI_WIDTH	0x3ff00618	RW		0x0
MTX0_SRC_ROW_STRIDE	0x3ff00620	RW		0x0

Page 130

MTX0_DST_ROW_STRIDE	0x3ff00628	RW	0x0
MTX0_TRANS_CTRL	0x3ff00630	RW	0x0
MTX1_SRC_START_ADDR	0x3ff00700	RW	0x0
MTX1_DST_START_ADDR	0x3ff00708	RW	0x0
MTX1_ORI_LENTH	0x3ff00710	RW	0x0
MTX1_ORI_WIDTH	0x3ff00718	RW	0x0
MTX1_SRC_ROW_STRIDE	0x3ff00720	RW	0x0
MTX1_DST_ROW_STRIDE	0x3ff00728	RW	0x0
MTX1_TRANS_CTRL	0x3ff00730	RW	0x0
SCache0_perfctrl0	0x3ff00800	RW	
SCache0_perfcnt0	0x3ff00808	RO	
SCache0_perfctrl1	0x3ff00810	RW	
SCache0_perfcnt1	0x3ff00818	RO	
SCache0_perfctrl2	0x3ff00820	RW	
SCache0_perfcnt2	0x3ff00828	RO	
SCache0_perfctrl3	0x3ff00830	RW	
SCache0_perfcnt3	0x3ff00838	RO	
SCache1_perfctrl0	0x3ff00900	RW	
SCache1_perfcnt0	0x3ff00908	RO	

Page 131

SCache1_perfctrl1	0x3ff00910	RW
SCache1_perfcnt1	0x3ff00918	RO
SCache1_perfctrl2	0x3ff00920	RW
SCache1_perfcnt2	0x3ff00928	RO
SCache1_perfctrl3	0x3ff00930	RW
SCache1_perfcnt3	0x3ff00938	RO
SCache2_perfctrl0	0x3ff00A00	RW
SCache2_perfcnt0	0x3ff00A08	RO
SCache2_perfctrl1	0x3ff00A10	RW
SCache2_perfcnt1	0x3ff00A18	RO
SCache2_perfctrl2	0x3ff00A20	RW
SCache2_perfcnt2	0x3ff00A28	RO
SCache2_perfctrl3	0x3ff00A30	RW
SCache2_perfcnt3	0x3ff00A38	RO
SCache3_perfctrl0	0x3ff00B00	RW
SCache3_perfcnt0	0x3ff00B08	RO
SCache3_perfctrl1	0x3ff00B10	RW
SCache3_perfcnt1	0x3ff00B18	RO

SCache3_perfctrl2	0x3ff00B20	RW		

Page 132

SCache3_perfcnt2	0x3ff00B28	RO		
SCache3_perfctrl3	0x3ff00B30	RW		
SCache3_perfcnt3	0x3ff00B38	RO		
Core0_IPI_Status	0x3ff01000	RO	IPI_Status register of processor core 0	
Core0_IPI_Enalbe	0x3ff01004	RW	IPI_Enalbe register of processor core 0	0x0
Core0_IPI_Set	0x3ff01008	WO	IPI_Set register of processor core 0	
Core0_IPI_Clear	0x3ff0100c	WO	IPI_Clear register of processor core 0	
Core0_MailBox0	0x3ff01020	RW	IPI_MailBox0 register of processor core 0	0x0
Core0_MailBox1	0x3ff01028	RW	IPI_MailBox1 register of processor core 0	0x0
Core0_MailBox2	0x3ff01030	RW	IPI_MailBox2 register of processor core 0	0x0
Core0_MailBox3	0x3ff01038	RW	IPI_MailBox3 register of processor core 0	0x0
Core0_int_interval	0x3ff01060	RW		
Core0_int_compare	0x3ff01068	RW		
Core1_IPI_Status	0x3ff01100	RO	IPI_Status register of processor core 1	
Core1_IPI_Enalbe	0x3ff01104	RW	IPI_Enalbe register of processor core 1	0x0
Core1_IPI_Set	0x3ff01108	WO	IPI_Set register of processor core 1	
Core1_IPI_Clear	0x3ff0110c	WO	IPI_Clear register of processor core 1	
Core1_MailBox0	0x3ff01120	RW	IPI_MailBox0 register of processor core 1	0x0
Core1_MailBox1	0x3ff01128	RW	IPI_MailBox1 register of processor core 1	0x0

Page 133

Core1_MailBox2	0x3ff01130	RW	IPI_MailBox2 register of processor core 1	0x0
Core1_MailBox3	0x3ff01138	RW	IPI_MailBox3 register of processor core 1	0x0
Core1_int_interval	0x3ff01160	RW		
Core1_int_compare	0x3ff01168	RW		
Core2_IPI_Status	0x3ff01200	RO	IPI_Status register of processor core 2	
Core2_IPI_Enalbe	0x3ff01204	RW	IPI_Enalbe register of processor core 2	0x0
Core2_IPI_Set	0x3ff01208	WO	IPI_Set register of processor core 2	
Core2_IPI_Clear	0x3ff0120c	WO	IPI_Clear register of processor core 2	
Core2_MailBox0	0x3ff01220	RW	IPI_MailBox0 register of processor core 2	0x0
Core2_MailBox1	0x3ff01228	RW	IPI_MailBox1 register of processor core 2	0x0
Core2_MailBox2	0x3ff01230	RW	IPI_MailBox2 register of processor core 2	0x0

Core2_MailBox3	0x3ff01238	RW	IPI_MailBox3 register of processor core 2	0x0
Core2_int_interval	0x3ff01260	RW		
Core2_int_compare	0x3ff01268	RW		
Core3_IPI_Status	0x3ff01300	RO	IPI_Status register of processor core 3	
Core3_IPI_Enalbe	0x3ff01304	RW	IPI_Enalbe register of processor core 3	0x0
Core3_IPI_Set	0x3ff01308	WO	IPI_Set register of processor core 3	
Core3_IPI_Clear	0x3ff0130c	WO	IPI_Clear register of processor core 3	
Core3_MailBox0	0x3ff01320	RW	IPI_MailBox0 register of processor core 3	0x0

Core3_MailBox1	0x3ff01328	RW	IPI_MailBox1 register of processor core 3	0x0
Core3_MailBox2	0x3ff01330	RW	IPI_MailBox2 register of processor core 3	0x0
Core3_MailBox3	0x3ff01338	RW	IPI_MailBox3 register of processor core 3	0x0
Core3_int_interval	0x3ff01360	RW		
Core3_int_compare	0x3ff01368	RW		
Int Entry [0--31]	0x3ff01400	RW	32 8-bit interrupt routing registers	0x0
Intisr	0x3ff01420	RO	32-bit interrupt status register	
Inten	0x3ff01424	RO	32-bit interrupt enable status register	
Intenset	0x3ff01428	WO	32-bit setting enable register	
Intenclr	0x3ff0142c	WO	32-bit clear enable register and pulse triggered interrupt	
Intpol	0x3ff01430	WO	useless	0x0
Intedge	0x3ff01434	WO	32-bit trigger mode register (1: pulse trigger; 0: level trigger)	0x0
CORE0_INTISR	0x3ff01440	RO	32-bit interrupt status routed to CORE0	
CORE1_INTISR	0x3ff01448	RO	32-bit interrupt status routed to CORE1	
CORE2_INTISR	0x3ff01450	RO	32-bit interrupt status routed to CORE2	
CORE3_INTISR	0x3ff01458	RO	32-bit interrupt status routed to CORE3	

Temperature sensor high temperature interrupt control register

[7: 0]: Hi_gate0: high temperature threshold 0, an interrupt will be generated if this temperature is exceeded

[8: 8]: Hi_en0: High temperature interrupt enable 0

[11:10]: Hi_Sel0: Select the temperature sensor input source of high temperature interrupt

Thsens_int_ctrl_Hi	0x3ff01460	RW	[23:16]: Hi_gate1: high temperature threshold 1, exceeding this temperature will generate an interrupt [24:24]: Hi_en1: High temperature interrupt enable 1 [27:26]: Hi_Sel1: Select the temperature sensor input source for high temperature interrupt 1 [39:32]: Hi_gate2: High temperature threshold 2, above this temperature will generate an interrupt [40:40]: Hi_en2: High temperature interrupt enable 2 [43:42]: Hi_Sel2: Select the temperature sensor input source for high temperature interrupt 2 [55:48]: Hi_gate3: High temperature threshold 3, exceeding this temperature will generate interrupt [56:56]: Hi_en3: High temperature interrupt enable 3 [59:58]: Hi_Sel3: Select the temperature sensor input source for high temperature interrupt 3

Thsens_int_ctrl_Lo	0x3ff01468	RW	Temperature sensor low temperature interrupt control register [7: 0]: Lo_gate0: low temperature threshold 0, below this temperature will generate an interrupt [8: 8]: Lo_en0: Low temperature interrupt enable 0 [11:10]: Lo_Sel0: Select the temperature sensor input source for low temperature interrupt 0 [23:16]: Lo_gate1: low temperature threshold 1, below this temperature will generate an interrupt [24:24]: Lo_en1: Low temperature interrupt enable 1 [27:26]: Lo_Sel1: Select the temperature sensor input source for low temperature interrupt 1 [39:32]: Lo_gate2: Low temperature threshold 2, below this temperature will generate an interrupt [40:40]: Lo_en2: Low temperature interrupt enable 2 [43:42]: Lo_Sel2: Select the temperature sensor input source for low temperature interrupt 2 [55:48]: Lo_gate3: Low temperature threshold 3, below this temperature will generate an interrupt [56:56]: Lo_en3: Low temperature interrupt enable 3 [59:58]: Lo_Sel3: Select temperature sensor input source for low temperature interrupt 3
Thsens_int_status / clr	0x3ff01470	RW	Interrupt status register, write any value to clear the interrupt [0]: High temperature interrupt trigger [1]: Low temperature interrupt trigger

Thsens_freq_scale	0x3ff01480	RW	Temperature sensor high-temperature down-frequency control register, four sets of setting priority from high to low [7: 0]: Scale_gate0: High temperature threshold 0, frequency will be reduced if this temperature is exceeded [8: 8]: Scale_en0: High temperature frequency reduction enable 0 [11:10]: Scale_Sel0: Select the temperature sensor input source of high temperature down-conversion 0 [14:12]: Scale_freq0: frequency division value when frequency is reduced [23:16]: Scale_gate1: High temperature threshold 1, exceeding this temperature will reduce the frequency [24:24]: Scale_en1: High temperature frequency reduction enable 1 [27:26]: Scale_Sel1: Select the temperature sensor input source for high temperature down-conversion 1 [30:28]: Scale_freq1: frequency division value when frequency is reduced [39:32]: Scale_gate2: High temperature threshold value 2, if this temperature is exceeded, frequency will be reduced [40:40]: Scale_en2: High temperature frequency reduction enable 2 [43:42]: Scale_Sel2: Select the temperature sensor input source for high temperature down-conversion 2 [46:44]: Scale_freq2: frequency division value when frequency is reduced [55:48]: Scale_gate3: High temperature threshold 3, over this temperature will reduce the frequency [56:56]: Scale_en3: High temperature frequency reduction enable 3 [59:58]: Scale_Sel3: Select the temperature sensor input source for high temperature down-conversion 3 [62:60]: Scale_freq3: Frequency division value when frequency is reduced

DFD_PARAM	0x3ff01500	RW	Debugging trigger condition enable [7: 0]: timer, trigger delay, set to 1 means to trigger immediately when the condition is met, set to 0 to prohibit touch Send, set to other values means that the number of beats delayed trigger after the condition is met +1 [15: 8]: trigger_en, trigger condition enable, corresponding to the enable control of 8 external trigger events
DFD_TRIGGER	0x3ff01508	WO	Software trigger, sending a write operation to this address will cause a software trigger condition, making Triggered after shooting
CORE0_AWCOND0	0x3ff01800	RW	CORE0 AXI interface AW trigger condition 0 setting [15: 0]: awid [19:16]: awlen [22:20]: awsize [24:23]: awburst [26:25]: awlock [30:27]: awcache [33:31]: awprot [37:34]: awcmd [41:38]: awdirqid [43:42]: awstate [47:44]: swscseti [48]: awvalid [49]: awready

Page 139

CORE0_AWMASK0	0x3ff01808	RW	CORE0 AXI interface AW trigger enable 0 is set, the highest bit is AW channel trigger enable [49: 0]: awmask [62]: awdata_en: trigger is allowed only when the wdata trigger condition of the same wid is met at the same time [63]: awchannel_en: enable trigger condition The trigger condition is (AW_IN & AWMASK) == (AWCOND & AWMASK) The trigger condition of AW must be satisfied by both COND0 and COND1
CORE0_AWCOND1	0x3ff01810	RW	[47: 0]: awaddr
CORE0_AWMASK1	0x3ff01818	RW	

Page 140

CORE0_ARCOND0	0x3ff01820	RW	CORE0's AXI interface AR trigger condition, similar to AW [15: 0]: arid [19:16]: arlen [22:20]: arsize [24:23]: arburst [26:25]: arlock [30:27]: arcache [33:31]: arprot [37:34]: arcmd [47:38]: arcpuno [48]: arvalid [49]: arready

			CORE0's AXI interface AR trigger enable 0 is set, the highest bit is the AR channel trigger enable [49: 0]: armask [62]: ardata_en: trigger is allowed only when the rdata trigger condition of the same rid is met
CORE0_ARMASK0	0x3ff01828	RW	[63]: archannel_en: enable trigger condition
CORE0_ARCOND1	0x3ff01830	RW	[47: 0]: araddr
CORE0_ARMASK1	0x3ff01838	RW	

			CORE0's AXI interface W trigger condition, similar to AW [15: 0]: wid [31:16]: wstrb [32]: wlast [33]: wvalid
CORE0_WCOND0	0x3ff01840	RW	[34]: wready
			CORE0's AXI interface W trigger enable 0 setting, the highest bit is the W channel trigger enable [49: 0]: wmask
CORE0_WMASK0	0x3ff01848	RW	[63]: wchannel_en: Trigger condition enable, no need to set when awdata_en is valid
CORE0_WCOND1	0x3ff01850	RW	
CORE0_WMASK1	0x3ff01858	RW	
CORE0_WCOND2	0x3ff01860	RW	
CORE0_WMASK2	0x3ff01868	RW	
			CORE0 AXI interface B trigger condition, similar to AW [15: 0]: bid [17:16]: bresp [18]: bvalid
CORE0_BCOND0	0x3ff01870	RW	[19]: ready

			CORE0's AXI interface B trigger enable 0 setting, the highest bit is the B channel trigger enable [19: 0]: bmask
CORE0_BMASK0	0x3ff01878	RW	[63]: bchannel_en
			CORE0 AXI interface R trigger condition, similar to AW [15: 0]: rid

CORE0_RCOND0	0x3ff01880	RW	[17:16]: rresp [18]: rlast [19]: rrequest [21:20]: rstate [25:22]: rscseti [26]: rvalid [27]: rready
CORE0_RMASK0	0x3ff01888	RW	CORE0's AXI interface R trigger enable 0 setting, the highest bit is the R channel trigger enable [27: 0]: rmask [63]: rchannel_en
CORE0_RCOND1	0x3ff01890	RW	
CORE0_RMASK1	0x3ff01898	RW	
CORE0_RCOND2	0x3ff018a0	RW	
CORE0_RMASK2	0x3ff018a8	RW	

TUD0_CONF0	0x3ff018e0	RW	TUD0 configuration register 0 [47: 0]: count_target [55:48]: monitor_enable
TUD0_CONF1	0x3ff018e8	RW	TUD0 configuration register 1 [2: 0]: DCDL_sel_signal [5: 3]: DCDL_sel_clock [9: 6]: signal_sel [13:10]: clok_sel [20:14]: reading_sel [21]: counter_clock_sel [22]: sticky [23]: reset_g [24]: stop [25]: start [26]: cg_en
TUD0_RESULT	0x3ff018f0	R	TUD0 result register
CORE1_AWCOND0	0x3ff01900	RW	CORE1 AXI interface AW trigger condition 0 setting

CORE1_AWMASK0	0x3ff01908	RW	CORE1 AXI interface AW trigger enable 0 is set, the highest bit is AW channel trigger enable The trigger condition is (AW_IN & AWMASK) == (AWCOND & AWMASK)
CORE1_AWCOND1	0x3ff01910	RW	The trigger condition of AW must be satisfied by both COND0 and COND1
CORE1_AWMASK1	0x3ff01918	RW	
CORE1_ARCOND0	0x3ff01920	RW	CORE1's AXI interface AR trigger condition, similar to AW
CORE1_ARMASK0	0x3ff01928	RW	
CORE1_ARCOND1	0x3ff01930	RW	
CORE1_ARMASK1	0x3ff01938	RW	
CORE1_WCOND0	0x3ff01940	RW	CORE1's AXI interface W trigger condition, similar to AW
CORE1_WMASK0	0x3ff01948	RW	
CORE1_WCOND1	0x3ff01950	RW	
CORE1_WMASK1	0x3ff01958	RW	
CORE1_WCOND2	0x3ff01960	RW	
CORE1_WMASK2	0x3ff01968	RW	
CORE1_BCOND0	0x3ff01970	RW	CORE1's AXI interface B trigger condition, similar to AW
CORE1_BMASK0	0x3ff01978	RW	
CORE1_RCOND0	0x3ff01980	RW	CORE1's AXI interface R trigger condition, similar to AW
CORE1_RMASK0	0x3ff01988	RW	

CORE1_RCOND1	0x3ff01990	RW	
CORE1_RMASK1	0x3ff01998	RW	
CORE1_RCOND2	0x3ff019a0	RW	
CORE1_RMASK2	0x3ff019a8	RW	
TUD1_CONF0	0x3ff019e0	RW	TUD1 configuration register 0 [47: 0]: count_target [55:48]: monitor_enable
TUD1_CONF1	0x3ff019e8	RW	TUD0 configuration register 1 [2: 0]: DCDL_sel_signal [5: 3]: DCDL_sel_clock [9: 6]: signal_sel [13:10]: clok_sel [20:14]: reading_sel [21]: counter_clock_sel [22]: sticky [23]: reset_g [24]: stop [25]: start [26]: cg_en

TUD1_RESULT	0x3ff019f0	R	TUD1 result register
CORE2_AWCOND0	0x3ff01a00	RW	CORE2 AXI interface AW trigger condition 0 setting CORE2's AXI interface AW trigger enable 0 setting, the highest bit is AW channel trigger enable The trigger condition is
CORE2_AWMASK0	0x3ff01a08	RW	(AW_IN & AWMASK) == (AWCOND & AWMASK)
CORE2_AWCOND1	0x3ff01a10	RW	The trigger condition of AW must be satisfied by both COND0 and COND1
CORE2_AWMASK1	0x3ff01a18	RW	
CORE2_ARCOND0	0x3ff01a20	RW	CORE2's AXI interface AR trigger condition, similar to AW
CORE2_ARMASK0	0x3ff01a28	RW	
CORE2_ARCOND1	0x3ff01a30	RW	
CORE2_ARMASK1	0x3ff01a38	RW	
CORE2_WCOND0	0x3ff01a40	RW	CORE2's AXI interface W trigger condition, similar to AW
CORE2_WMASK0	0x3ff01a48	RW	
CORE2_WCOND1	0x3ff01a50	RW	
CORE2_WMASK1	0x3ff01a58	RW	
CORE2_WCOND2	0x3ff01a60	RW	
CORE2_WMASK2	0x3ff01a68	RW	
CORE2_BCOND0	0x3ff01a70	RW	CORE2 AXI interface B trigger condition, similar to AW

CORE2_BMASK0	0x3ff01a78	RW	
CORE2_RCOND0	0x3ff01a80	RW	CORE2's AXI interface R trigger condition, similar to AW
CORE2_RMASK0	0x3ff01a88	RW	
CORE2_RCOND1	0x3ff01a90	RW	
CORE2_RMASK1	0x3ff01a98	RW	
CORE2_RCOND2	0x3ff01aa0	RW	
CORE2_RMASK2	0x3ff01aa8	RW	
TUD2_CONF0	0x3ff01ae0	RW	TUD2 configuration register 0 [47: 0]: count_target [55:48]: monitor_enable

Page 148

TUD2_CONF1	0x3ff01ae8	RW	TUD0 configuration register 1 [2: 0]: DCDL_sel_signal [5: 3]: DCDL_sel_clock [9: 6]: signal_sel [13:10]: clok_sel [20:14]: reading_sel [21]: counter_clock_sel [22]: sticky [23]: reset_g [24]: stop [25]: start [26]: cg_en
TUD2_RESULT	0x3ff01af0	R	TUD2 result register
CORE3_AWCOND0	0x3ff01b00	RW	CORE3 AXI interface AW trigger condition 0 setting CORE3 AXI interface AW trigger enable 0 is set, the highest bit is AW channel trigger enable The trigger condition is
CORE3_AWMASK0	0x3ff01b08	RW	(AW_IN & AWMASK) == (AWCOND & AWMASK)
CORE3_AWCOND1	0x3ff01b10	RW	The trigger condition of AW must be satisfied by both COND0 and COND1
CORE3_AWMASK1	0x3ff01b18	RW	

Page 149

CORE3_ARCOND0	0x3ff01b20	RW	CORE3's AXI interface AR trigger condition, similar to AW
CORE3_ARMASK0	0x3ff01b28	RW	
CORE3_ARCOND1	0x3ff01b30	RW	
CORE3_ARMASK1	0x3ff01b38	RW	
CORE3_WCOND0	0x3ff01b40	RW	CORE3's AXI interface W trigger condition, similar to AW
CORE3_WMASK0	0x3ff01b48	RW	

CORE3_WCOND1	0x3ff01b50	RW	
CORE3_WMASK1	0x3ff01b58	RW	
CORE3_WCOND2	0x3ff01b60	RW	
CORE3_WMASK2	0x3ff01b68	RW	
CORE3_BCOND0	0x3ff01b70	RW	CORE3 AXI interface B trigger condition, similar to AW
CORE3_BMASK0	0x3ff01b78	RW	
CORE3_RCOND0	0x3ff01b80	RW	CORE3's AXI interface R trigger condition, similar to AW
CORE3_RMASK0	0x3ff01b88	RW	
CORE3_RCOND1	0x3ff01b90	RW	
CORE3_RMASK1	0x3ff01b98	RW	
CORE3_RCOND2	0x3ff01ba0	RW	
CORE3_RMASK2	0x3ff01ba8	RW	

TUD3_CONF0	0x3ff01be0	RW	TUD3 configuration register 0 [47: 0]: count_target [55:48]: monitor_enable
TUD3_CONF1	0x3ff01be8	RW	TUD0 configuration register 1 [2: 0]: DCDL_sel_signal [5: 3]: DCDL_sel_clock [9: 6]: signal_sel [13:10]: clok_sel [20:14]: reading_sel [21]: counter_clock_sel [22]: sticky [23]: reset_g [24]: stop [25]: start [26]: cg_en
TUD3_RESULT	0x3ff01bf0	R	TUD3 result register
TUD4_CONF0	0x3ff01ce0	RW	TUD4 configuration register 0 [47: 0]: count_target [55:48]: monitor_enable

TUD4_CONF1	0x3ff01ce8	RW	TUD4 configuration register 1 [2: 0]: DCDL_sel_signal [5: 3]: DCDL_sel_clock [8: 6]: signal_sel [11: 9]: clock_sel [18:12]: reading_sel [19]: counter_clock_sel [20]: sticky [21]: reset_g [22]: stop [23]: start [24]: cg_en
TUD4_RESULT	0x3ff01cf0	R	TUD4 result register
TUD5_CONF0	0x3ff01de0	RW	TUD5 configuration register 0 [47: 0]: count_target [55:48]: monitor_enable

TUD5_CONF1	0x3ff01de8	RW	TUD5 configuration register 1 [2: 0]: DCDL_sel_signal [5: 3]: DCDL_sel_clock [8: 6]: signal_sel [11: 9]: clock_sel [18:12]: reading_sel [19]: counter_clock_sel [20]: sticky [21]: reset_g [22]: stop [23]: start [24]: cg_en
TUD5_RESULT	0x3ff01df0	R	TUD5 result register
HT0_AWCOND0	0x3ff01e00	RW	HT0 AXI interface AW trigger condition 0 setting
HT0_AWMASK0	0x3ff01e08	RW	HT0's AXI interface AW trigger enable 0 setting, the highest bit is AW channel trigger enable The trigger condition is (AW_IN & AWMASK) == (AWCOND & AWMASK)
HT0_AWCOND1	0x3ff01e10	RW	The trigger condition of AW must be satisfied by both COND0 and COND1
HT0_AWMASK1	0x3ff01e18	RW	

Page 153

HT0_ARCOND0	0x3ff01e20	RW	HT0's AXI interface AR trigger condition, similar to AW
HT0_ARMASK0	0x3ff01e28	RW	
HT0_ARCOND1	0x3ff01e30	RW	
HT0_ARMASK1	0x3ff01e38	RW	
HT0_WCOND0	0x3ff01e40	RW	HT0's AXI interface W trigger condition, similar to AW
HT0_WMASK0	0x3ff01e48	RW	
HT0_WCOND1	0x3ff01e50	RW	
HT0_WMASK1	0x3ff01e58	RW	
HT0_WCOND2	0x3ff01e60	RW	
HT0_WMASK2	0x3ff01e68	RW	
HT0_BCOND0	0x3ff01e70	RW	HT0's AXI interface B trigger condition, similar to AW
HT0_BMASK0	0x3ff01e78	RW	
HT0_RCOND0	0x3ff01e80	RW	HT0's AXI interface R trigger condition, similar to AW
HT0_RMASK0	0x3ff01e88	RW	
HT0_RCOND1	0x3ff01e90	RW	
HT0_RMASK1	0x3ff01e98	RW	
HT0_RCOND2	0x3ff01ea0	RW	
HT0_RMASK2	0x3ff01ea8	RW	
HT1_AWCOND0	0x3ff01f00	RW	HT1 AXI interface AW trigger condition 0 setting

Page 154

			HT1's AXI interface AW trigger enable 0 setting, the highest bit is AW channel trigger enable
			The trigger condition is
HT1_AWMASK0	0x3ff01f08	RW	(AW_IN & AWMASK) == (AWCOND & AWMASK)
HT1_AWCOND1	0x3ff01f10	RW	The trigger condition of AW must be satisfied by both COND0 and COND1
HT1_AWMASK1	0x3ff01f18	RW	
HT1_ARCOND0	0x3ff01f20	RW	HT1's AXI interface AR trigger condition, similar to AW
HT1_ARMASK0	0x3ff01f28	RW	
HT1_ARCOND1	0x3ff01f30	RW	
HT1_ARMASK1	0x3ff01f38	RW	
HT1_WCOND0	0x3ff01f40	RW	HT1's AXI interface W trigger condition, similar to AW
HT1_WMASK0	0x3ff01f48	RW	
HT1_WCOND1	0x3ff01f50	RW	
HT1_WMASK1	0x3ff01f58	RW	
HT1_WCOND2	0x3ff01f60	RW	

HT1_WMASK2	0x3ff01f68	RW		
HT1_BCOND0	0x3ff01f70	RW	HT1's AXI interface B trigger condition, similar to AW	
HT1_BMASK0	0x3ff01f78	RW		
HT1_RCOND0	0x3ff01f80	RW	HT1's AXI interface R trigger condition, similar to AW	
HT1_RMASK0	0x3ff01f88	RW		

HT1_RCOND1	0x3ff01f90	RW		
HT1_RMASK1	0x3ff01f98	RW		
HT1_RCOND2	0x3ff01fa0	RW		
HT1_RMASK2	0x3ff01fa8	RW		
CORE0_WIN0_BASE	0x3ff02000	RW	First-level crossbar address window	0x0
CORE0_WIN1_BASE	0x3ff02008	RW	First-level crossbar address window	0x0
CORE0_WIN2_BASE	0x3ff02010	RW	First-level crossbar address window	0x0
CORE0_WIN3_BASE	0x3ff02018	RW	First-level crossbar address window	0x0
CORE0_WIN4_BASE	0x3ff02020	RW	First-level crossbar address window	0x0
CORE0_WIN5_BASE	0x3ff02028	RW	First-level crossbar address window	0x0
CORE0_WIN6_BASE	0x3ff02030	RW	First-level crossbar address window	0x0
CORE0_WIN7_BASE	0x3ff02038	RW	First-level crossbar address window	0x0
CORE0_WIN0_MASK	0x3ff02040	RW	First-level crossbar address window	0x0
CORE0_WIN1_MASK	0x3ff02048	RW	First-level crossbar address window	0x0
CORE0_WIN2_MASK	0x3ff02050	RW	First-level crossbar address window	0x0
CORE0_WIN3_MASK	0x3ff02058	RW	First-level crossbar address window	0x0
CORE0_WIN4_MASK	0x3ff02060	RW	First-level crossbar address window	0x0
CORE0_WIN5_MASK	0x3ff02068	RW	First-level crossbar address window	0x0
CORE0_WIN6_MASK	0x3ff02070	RW	First-level crossbar address window	0x0

CORE0_WIN7_MASK	0x3ff02078	RW	First-level crossbar address window	0x0
CORE0_WIN0_MMAP	0x3ff02080	RW	First-level crossbar address window	0x0
CORE0_WIN1_MMAP	0x3ff02088	RW	First-level crossbar address window	0x0
CORE0_WIN2_MMAP	0x3ff02090	RW	First-level crossbar address window	0x0
CORE0_WIN3_MMAP	0x3ff02098	RW	First-level crossbar address window	0x0
CORE0_WIN4_MMAP	0x3ff020a0	RW	First-level crossbar address window	0x0
CORE0_WIN5_MMAP	0x3ff020a8	RW	First-level crossbar address window	0x0

CORE0_WIN6_MMAP	0x3ff020b0	RW	First-level crossbar address window	0x0
CORE0_WIN7_MMAP	0x3ff020b8	RW	First-level crossbar address window	0x0
CORE1_WIN0_BASE	0x3ff02100	RW	First-level crossbar address window	0x0
CORE1_WIN1_BASE	0x3ff02108	RW	First-level crossbar address window	0x0
CORE1_WIN2_BASE	0x3ff02110	RW	First-level crossbar address window	0x0
CORE1_WIN3_BASE	0x3ff02118	RW	First-level crossbar address window	0x0
CORE1_WIN4_BASE	0x3ff02120	RW	First-level crossbar address window	0x0
CORE1_WIN5_BASE	0x3ff02128	RW	First-level crossbar address window	0x0
CORE1_WIN6_BASE	0x3ff02130	RW	First-level crossbar address window	0x0
CORE1_WIN7_BASE	0x3ff02138	RW	First-level crossbar address window	0x0
CORE1_WIN0_MASK	0x3ff02140	RW	First-level crossbar address window	0x0
CORE1_WIN1_MASK	0x3ff02148	RW	First-level crossbar address window	0x0

CORE1_WIN2_MASK	0x3ff02150	RW	First-level crossbar address window	0x0
CORE1_WIN3_MASK	0x3ff02158	RW	First-level crossbar address window	0x0
CORE1_WIN4_MASK	0x3ff02160	RW	First-level crossbar address window	0x0
CORE1_WIN5_MASK	0x3ff02168	RW	First-level crossbar address window	0x0
CORE1_WIN6_MASK	0x3ff02170	RW	First-level crossbar address window	0x0
CORE1_WIN7_MASK	0x3ff02178	RW	First-level crossbar address window	0x0
CORE1_WIN0_MMAP	0x3ff02180	RW	First-level crossbar address window	0x0
CORE1_WIN1_MMAP	0x3ff02188	RW	First-level crossbar address window	0x0
CORE1_WIN2_MMAP	0x3ff02190	RW	First-level crossbar address window	0x0
CORE1_WIN3_MMAP	0x3ff02198	RW	First-level crossbar address window	0x0
CORE1_WIN4_MMAP	0x3ff021a0	RW	First-level crossbar address window	0x0
CORE1_WIN5_MMAP	0x3ff021a8	RW	First-level crossbar address window	0x0
CORE1_WIN6_MMAP	0x3ff021b0	RW	First-level crossbar address window	0x0
CORE1_WIN7_MMAP	0x3ff021b8	RW	First-level crossbar address window	0x0
CORE2_WIN0_BASE	0x3ff02200	RW	First-level crossbar address window	0x0
CORE2_WIN1_BASE	0x3ff02208	RW	First-level crossbar address window	0x0
CORE2_WIN2_BASE	0x3ff02210	RW	First-level crossbar address window	0x0
CORE2_WIN3_BASE	0x3ff02218	RW	First-level crossbar address window	0x0
CORE2_WIN4_BASE	0x3ff02220	RW	First-level crossbar address window	0x0

CORE2_WIN5_BASE	0x3ff02228	RW	First-level crossbar address window	0x0
CORE2_WIN6_BASE	0x3ff02230	RW	First-level crossbar address window	0x0
CORE2_WIN7_BASE	0x3ff02238	RW	First-level crossbar address window	0x0
CORE2_WIN0_MASK	0x3ff02240	RW	First-level crossbar address window	0x0
CORE2_WIN1_MASK	0x3ff02248	RW	First-level crossbar address window	0x0
CORE2_WIN2_MASK	0x3ff02250	RW	First-level crossbar address window	0x0
CORE2_WIN3_MASK	0x3ff02258	RW	First-level crossbar address window	0x0
CORE2_WIN4_MASK	0x3ff02260	RW	First-level crossbar address window	0x0
CORE2_WIN5_MASK	0x3ff02268	RW	First-level crossbar address window	0x0
CORE2_WIN6_MASK	0x3ff02270	RW	First-level crossbar address window	0x0
CORE2_WIN7_MASK	0x3ff02278	RW	First-level crossbar address window	0x0
CORE2_WIN0_MMAP	0x3ff02280	RW	First-level crossbar address window	0x0
CORE2_WIN1_MMAP	0x3ff02288	RW	First-level crossbar address window	0x0
CORE2_WIN2_MMAP	0x3ff02290	RW	First-level crossbar address window	0x0
CORE2_WIN3_MMAP	0x3ff02298	RW	First-level crossbar address window	0x0
CORE2_WIN4_MMAP	0x3ff022a0	RW	First-level crossbar address window	0x0
CORE2_WIN5_MMAP	0x3ff022a8	RW	First-level crossbar address window	0x0
CORE2_WIN6_MMAP	0x3ff022b0	RW	First-level crossbar address window	0x0
CORE2_WIN7_MMAP	0x3ff022b8	RW	First-level crossbar address window	0x0

Page 159

CORE3_WIN0_BASE	0x3ff02300	RW	First-level crossbar address window	0x0
CORE3_WIN1_BASE	0x3ff02308	RW	First-level crossbar address window	0x0
CORE3_WIN2_BASE	0x3ff02310	RW	First-level crossbar address window	0x0
CORE3_WIN3_BASE	0x3ff02318	RW	First-level crossbar address window	0x0
CORE3_WIN4_BASE	0x3ff02320	RW	First-level crossbar address window	0x0
CORE3_WIN5_BASE	0x3ff02328	RW	First-level crossbar address window	0x0
CORE3_WIN6_BASE	0x3ff02330	RW	First-level crossbar address window	0x0
CORE3_WIN7_BASE	0x3ff02338	RW	First-level crossbar address window	0x0
CORE3_WIN0_MASK	0x3ff02340	RW	First-level crossbar address window	0x0
CORE3_WIN1_MASK	0x3ff02348	RW	First-level crossbar address window	0x0
CORE3_WIN2_MASK	0x3ff02350	RW	First-level crossbar address window	0x0
CORE3_WIN3_MASK	0x3ff02358	RW	First-level crossbar address window	0x0
CORE3_WIN4_MASK	0x3ff02360	RW	First-level crossbar address window	0x0
CORE3_WIN5_MASK	0x3ff02368	RW	First-level crossbar address window	0x0
CORE3_WIN6_MASK	0x3ff02370	RW	First-level crossbar address window	0x0
CORE3_WIN7_MASK	0x3ff02378	RW	First-level crossbar address window	0x0
CORE3_WIN0_MMAP	0x3ff02380	RW	First-level crossbar address window	0x0
CORE3_WIN1_MMAP	0x3ff02388	RW	First-level crossbar address window	0x0
CORE3_WIN2_MMAP	0x3ff02390	RW	First-level crossbar address window	0x0

Page 160

CORE3_WIN3_MMAP	0x3ff02398	RW	First-level crossbar address window	0x0
CORE3_WIN4_MMAP	0x3ff023a0	RW	First-level crossbar address window	0x0
CORE3_WIN5_MMAP	0x3ff023a8	RW	First-level crossbar address window	0x0
CORE3_WIN6_MMAP	0x3ff023b0	RW	First-level crossbar address window	0x0
CORE3_WIN7_MMAP	0x3ff023b8	RW	First-level crossbar address window	0x0
EAST_WIN0_BASE	0x3ff02400	RW	First-level crossbar address window	0x0
EAST_WIN1_BASE	0x3ff02408	RW	First-level crossbar address window	0x0
EAST_WIN2_BASE	0x3ff02410	RW	First-level crossbar address window	0x0
EAST_WIN3_BASE	0x3ff02418	RW	First-level crossbar address window	0x0
EAST_WIN4_BASE	0x3ff02420	RW	First-level crossbar address window	0x0
EAST_WIN5_BASE	0x3ff02428	RW	First-level crossbar address window	0x0
EAST_WIN6_BASE	0x3ff02430	RW	First-level crossbar address window	0x0
EAST_WIN7_BASE	0x3ff02438	RW	First-level crossbar address window	0x0
EAST_WIN0_MASK	0x3ff02440	RW	First-level crossbar address window	0x0
EAST_WIN1_MASK	0x3ff02448	RW	First-level crossbar address window	0x0
EAST_WIN2_MASK	0x3ff02450	RW	First-level crossbar address window	0x0
EAST_WIN3_MASK	0x3ff02458	RW	First-level crossbar address window	0x0
EAST_WIN4_MASK	0x3ff02460	RW	First-level crossbar address window	0x0
EAST_WIN5_MASK	0x3ff02468	RW	First-level crossbar address window	0x0

Page 161

EAST_WIN6_MASK	0x3ff02470	RW	First-level crossbar address window	0x0
EAST_WIN7_MASK	0x3ff02478	RW	First-level crossbar address window	0x0
EAST_WIN0_MMAP	0x3ff02480	RW	First-level crossbar address window	0x0
EAST_WIN1_MMAP	0x3ff02488	RW	First-level crossbar address window	0x0
EAST_WIN2_MMAP	0x3ff02490	RW	First-level crossbar address window	0x0
EAST_WIN3_MMAP	0x3ff02498	RW	First-level crossbar address window	0x0
EAST_WIN4_MMAP	0x3ff024a0	RW	First-level crossbar address window	0x0
EAST_WIN5_MMAP	0x3ff024a8	RW	First-level crossbar address window	0x0
EAST_WIN6_MMAP	0x3ff024b0	RW	First-level crossbar address window	0x0
EAST_WIN7_MMAP	0x3ff024b8	RW	First-level crossbar address window	0x0
SOUTH_WIN0_BASE	0x3ff02500	RW	First-level crossbar address window	0x0
SOUTH_WIN1_BASE	0x3ff02508	RW	First-level crossbar address window	0x0
SOUTH_WIN2_BASE	0x3ff02510	RW	First-level crossbar address window	0x0
SOUTH_WIN3_BASE	0x3ff02518	RW	First-level crossbar address window	0x0
SOUTH_WIN4_BASE	0x3ff02520	RW	First-level crossbar address window	0x0

SOUTH_WIN5_BASE	0x3ff02528	RW	First-level crossbar address window	0x0
SOUTH_WIN6_BASE	0x3ff02530	RW	First-level crossbar address window	0x0
SOUTH_WIN7_BASE	0x3ff02538	RW	First-level crossbar address window	0x0
SOUTH_WIN0_MASK	0x3ff02540	RW	First-level crossbar address window	0x0

SOUTH_WIN1_MASK	0x3ff02548	RW	First-level crossbar address window	0x0
SOUTH_WIN2_MASK	0x3ff02550	RW	First-level crossbar address window	0x0
SOUTH_WIN3_MASK	0x3ff02558	RW	First-level crossbar address window	0x0
SOUTH_WIN4_MASK	0x3ff02560	RW	First-level crossbar address window	0x0
SOUTH_WIN5_MASK	0x3ff02568	RW	First-level crossbar address window	0x0
SOUTH_WIN6_MASK	0x3ff02570	RW	First-level crossbar address window	0x0
SOUTH_WIN7_MASK	0x3ff02578	RW	First-level crossbar address window	0x0
SOUTH_WIN0_MMAP	0x3ff02580	RW	First-level crossbar address window	0x0
SOUTH_WIN1_MMAP	0x3ff02588	RW	First-level crossbar address window	0x0
SOUTH_WIN2_MMAP	0x3ff02590	RW	First-level crossbar address window	0x0
SOUTH_WIN3_MMAP	0x3ff02598	RW	First-level crossbar address window	0x0
SOUTH_WIN4_MMAP	0x3ff025a0	RW	First-level crossbar address window	0x0
SOUTH_WIN5_MMAP	0x3ff025a8	RW	First-level crossbar address window	0x0
SOUTH_WIN6_MMAP	0x3ff025b0	RW	First-level crossbar address window	0x0
SOUTH_WIN7_MMAP	0x3ff025b8	RW	First-level crossbar address window	0x0
WEST_WIN0_BASE	0x3ff02600	RW	First-level crossbar address window	0x0
WEST_WIN1_BASE	0x3ff02608	RW	First-level crossbar address window	0x0
WEST_WIN2_BASE	0x3ff02610	RW	First-level crossbar address window	0x0
WEST_WIN3_BASE	0x3ff02618	RW	First-level crossbar address window	0x0

WEST_WIN4_BASE	0x3ff02620	RW	First-level crossbar address window	0x0
WEST_WIN5_BASE	0x3ff02628	RW	First-level crossbar address window	0x0
WEST_WIN6_BASE	0x3ff02630	RW	First-level crossbar address window	0x0
WEST_WIN7_BASE	0x3ff02638	RW	First-level crossbar address window	0x0
WEST_WIN0_MASK	0x3ff02640	RW	First-level crossbar address window	0x0
WEST_WIN1_MASK	0x3ff02648	RW	First-level crossbar address window	0x0
WEST_WIN2_MASK	0x3ff02650	RW	First-level crossbar address window	0x0

WEST_WIN3_MASK	0x3ff02658	RW	First-level crossbar address window	0x0
WEST_WIN4_MASK	0x3ff02660	RW	First-level crossbar address window	0x0
WEST_WIN5_MASK	0x3ff02668	RW	First-level crossbar address window	0x0
WEST_WIN6_MASK	0x3ff02670	RW	First-level crossbar address window	0x0
WEST_WIN7_MASK	0x3ff02678	RW	First-level crossbar address window	0x0
WEST_WIN0_MMAP	0x3ff02680	RW	First-level crossbar address window	0x0
WEST_WIN1_MMAP	0x3ff02688	RW	First-level crossbar address window	0x0
WEST_WIN2_MMAP	0x3ff02690	RW	First-level crossbar address window	0x0
WEST_WIN3_MMAP	0x3ff02698	RW	First-level crossbar address window	0x0
WEST_WIN4_MMAP	0x3ff026a0	RW	First-level crossbar address window	0x0
WEST_WIN5_MMAP	0x3ff026a8	RW	First-level crossbar address window	0x0
WEST_WIN6_MMAP	0x3ff026b0	RW	First-level crossbar address window	0x0

WEST_WIN7_MMAP	0x3ff026b8	RW	First-level crossbar address window	0x0
NORTH_WIN0_BASE	0x3ff02700	RW	First-level crossbar address window	0x0
NORTH_WIN1_BASE	0x3ff02708	RW	First-level crossbar address window	0x0
NORTH_WIN2_BASE	0x3ff02710	RW	First-level crossbar address window	0x0
NORTH_WIN3_BASE	0x3ff02718	RW	First-level crossbar address window	0x0
NORTH_WIN4_BASE	0x3ff02720	RW	First-level crossbar address window	0x0
NORTH_WIN5_BASE	0x3ff02728	RW	First-level crossbar address window	0x0
NORTH_WIN6_BASE	0x3ff02730	RW	First-level crossbar address window	0x0
NORTH_WIN7_BASE	0x3ff02738	RW	First-level crossbar address window	0x0
NORTH_WIN0_MASK	0x3ff02740	RW	First-level crossbar address window	0x0
NORTH_WIN1_MASK	0x3ff02748	RW	First-level crossbar address window	0x0
NORTH_WIN2_MASK	0x3ff02750	RW	First-level crossbar address window	0x0
NORTH_WIN3_MASK	0x3ff02758	RW	First-level crossbar address window	0x0
NORTH_WIN4_MASK	0x3ff02760	RW	First-level crossbar address window	0x0
NORTH_WIN5_MASK	0x3ff02768	RW	First-level crossbar address window	0x0
NORTH_WIN6_MASK	0x3ff02770	RW	First-level crossbar address window	0x0
NORTH_WIN7_MASK	0x3ff02778	RW	First-level crossbar address window	0x0
NORTH_WIN0_MMAP	0x3ff02780	RW	First-level crossbar address window	0x0
NORTH_WIN1_MMAP	0x3ff02788	RW	First-level crossbar address window	0x0

NORTH_WIN2_MMAP	0x3ff02790	RW	First-level crossbar address window	0x0
NORTH_WIN3_MMAP	0x3ff02798	RW	First-level crossbar address window	0x0
NORTH_WIN4_MMAP	0x3ff027a0	RW	First-level crossbar address window	0x0
NORTH_WIN5_MMAP	0x3ff027a8	RW	First-level crossbar address window	0x0
NORTH_WIN6_MMAP	0x3ff027b0	RW	First-level crossbar address window	0x0
NORTH_WIN7_MMAP	0x3ff027b8	RW	First-level crossbar address window	0x0

Page 166

13 Software and Hardware Design Guidelines

Loongson 3A3000 / 3B3000 processor pins downward compatible with Loongson 3A1000 processor, but the corresponding hardware and software needs

Make some configuration changes to enable the original compatibility mode, or open some new features of Godson 3A3000 / 3B3000,

This chapter focuses on the software and hardware settings of the Loongson 3A3000 / 3B3000 processor compared with Loongson 3A1000 / 2000

the difference.

13.1 Hardware modification guide

1. The original CORE_PLL_AVDD and DDR_PLL_AVDD (2.5v) are now 1.8v. If you use the original 3A1000 motherboard,
   These two power supplies need to be changed from 2.5v to 1.8v. And these pins on 3A2000 / 3B2000 (including
   HT0 / 1_PLL_AVDD) is NC, if you consider the compatibility with 3A2000 / 3B2000, these power supply voltages
   Modified to 1.8v, or adopt 1.8v / 2.5v configurable design;

2. The original MC0 / 1_COMP_REF_RES is changed to NC pin. If the original 3A motherboard is used, no modification is required;
   (Consistent with 3A2000)

3. The original HT0 / 1_PLL_REF is changed to NC pin. If you use the original 3A motherboard, you do n't need to modify it; (with 3A2000
   Consistent)

4. The original MC0 / 1_COMP_REF_GND is changed to MC0 / 1_A15. If you use the original 3A motherboard, you can
   Modified; but if connected to a memory module, it can support a larger capacity memory; (same as 3A2000)

5. The function controlled by PCI_CONFIG [0] is changed to SPI startup enable. After setting to 1, it can be started from SPI FLASH. If make

Use the original 3A motherboard, it needs to be set to 0, start from LPC FLASH; if the motherboard already has SPI FLASH, you can GPIO [0] as SPI_CS connection, and set PCI_CONFIG [0] to 1, start from SPI FLASH; (and 3A2000 Consistent)

6. The function controlled by PCI_CONFIG [7] is changed to forced HT1.0 mode. After setting it to 1, HT starts directly in 1.0 mode.

If you use a 3A780E motherboard, you currently need to set it to 1; if you use a 3A2H motherboard, no special settings are required; (and 3A2000 consistent)

7. For Godson 3A3000 / 3B3000A / B / C, CLKSEL [15:10] needs to be set to 6'b000000 or 6'b000001, and

Reconfigure the frequency using software in PMON;

8. CLKSEL [9: 5] needs to be set to 5'b01111; use PMON to set the memory frequency. For Godson 3A3000 / 3B3000A / B / C, PMON must use NODE clock to divide frequency when frequency configuration; 3A2000 consistent)

9. CLKSEL [4: 0] needs to be set to 5'b01111; use PMON to set the processor core frequency. For Godson 3A3000 / 3B3000A / B / C, PMON must use L2 PLL as the main clock when configuring the frequency; (and 3A2000 Consistent)

10. For the 3A2H motherboard, you need to remove the pull-up resistors on HT0 / 1_powerok and HT0 / 1_resetn; (the original pull-up The resistance of 300 ohms is not suitable for 3A and can also be removed) (same as 3A2000)

13.2 Frequency setting instructions

In order to be basically compatible with the frequency configuration of Godson 3A1000, the hardware frequency configuration range of Godson 3A3000 / 3B3000 is relatively

Narrow, in order to obtain a wider frequency range and better clock quality, it is mainly used in PMON in Godson 3A3000 / 3B3000

The software configuration method is the same as that of Godson 3B1500. Please refer to the PMON source code for the specific configuration method.

1. The frequency setting is completely set by the software, there is no need to modify CLKSEL when changing the frequency;

2. Stable working frequency of 1.25V core voltage: processor core frequency is set to 1400MHz, memory frequency is set to 700MHz, HT controller is set to 800MHz, HT bus 800MHz / 1600MHz;

3. For Godson 3A3000 / 3B3000A / B / C, NODE CLOCK must use L2 PLL as the main clock, DDR CLOCK The NODE clock must be used for frequency division;

13.3 PMON Change Guide

The following PMON changes are basically consistent with the 3A2000 / 3B2000 processor.

Because from the processor core, memory controller, HT controller to all levels of crossbar switches have been upgraded to varying degrees,

Therefore, compared with Loongson 3A1000, PMON needs to make some changes, mainly including the following necessary parts:

1. Remove the initialization operations of L1 Dcache, L1 Icache, Vcache, and L2 Cache after power-on (hardware completion);

2. After the CPU is powered on, close the Store Fill Buffer of all cores;

3. Immediately after the CPU is powered on, turn off the word write merge function of all cores;

4. If you need to maintain compatibility with 3A5, set the PRID hidden bit in the CP0 Diag register of all cores;

5. Modify the statements of jr rx and rx which are not register 31 in all assembly codes to jr $ 31;

6. Use code similar to 3B1500 to configure processor core, memory and node PLL;

7. Use the memory controller configuration and parameter training code similar to 3B1500;

8. If HT works in 1.0 mode, HT can only work in 8-bit mode;

9. If an SPI controller is used, the base address is changed from 0xBFE001F0 to 0xBFE00220;

In addition to these necessary changes, the following changes can be made to enhance the PMON function:

1. Modify the delay delay of the buzzer to ensure that the user can hear the buzzer;

2. Add support to shut down the defective core clock;

3. Remove part of the workaround of the 3A5 to 2h bridge HT controller in the code (still retain some workaround);

13.4 Guidelines for kernel changes

The following kernel changes are basically the same as 3A2000 / 3B2000, but need to be added in the corresponding part of the kernel

3A3000 / 3B3000 processor support.

1. Modify the Cache description structure in the kernel. Both VCache and SCache are connected by a 16-way group; (same as 3A2000)

2. Modify the calculation method of the temperature sensor, the algorithm is: junction temperature = Thens_out * 731 / 0x4000-273;

3. Modify the configuration register address when shutting down the core; (same as 3A2000)

4. Change the operation of flashing ICache / DCache to flashing ICache / DCache / VCache; (same as 3A2000)

5. If an SPI controller is used, the base address is changed from 0xBFE001F0 to 0xBFE00220; (same as 3A2000)

6. Uncache DMA must be used, and the data consistency of Cache must be maintained by software; (consistent with 3A2000)

7. Add store fill buffer support: One is to add a SYNC before all Uncache requests to ensure

When the Uncache request occurs, the contents of the store fill buffer have been written back to the Cache; the second is that all

The unlock operation in the synchronous operation shared among different cores is implemented using LL / SC instructions. (Consistent with 3A2000)

8. Do not use the MSI function of the device. When you must use the MSI function, you need to transfer the data of the POST channel of the HT controller

Set the number of receive buffers to 1 and reconnect to the HT bus; (same as 3A2000)

9. Lock Cache operations cannot be used for DMA areas where hardware automatically maintains consistency. (Consistent with 3A2000)

Modifications that can also be used to improve performance are:

1. Increase support for FTLB; (consistent with 3A2000)

2. Add support for TLB fast refill; (consistent with 3A2000)

3. Add support for wait command; (consistent with 3A2000)

4. Add support for prefetch instructions; (consistent with 3A2000)

5. Use DI / EI to implement interrupt return. But it should be noted that the [31: 4] returned by the EI instruction is a random value, which is different from the MIPS

Differences. (Consistent with 3A2000)

166