Page 1

Loongson 3A2000 / 3B2000 processor

User Manual

volume One

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

V1.7

2017 Nian ***02*** Yue

Loongson Zhongke Technology Co., Ltd.

Page 2

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 3A2000 / 3B2000 Processor User Manual" is divided into the first and second volumes.

"Loongson 3A2000 / 3B2000 Processor User Manual" is divided into two parts, the first part introduces Loongson 3A2000 / 3B2000

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.

"Loongson 3A2000 / 3B2000 Processor User Manual" volume II, detailed introduction of Loongson from the perspective of system software developers

GS464e high-performance processor core used in 3A2000 / 3B2000.

Page 4

revise history

Document update record

Document name: Godson 3A2000 / 3B2000 Processor User Manual --volume One

version number V1.7

founder: Chip R & D Department

Creation Date: 2017-02-14

Update history

Serial number	Updated	version number	update content
1	2015-03-31	V1.0	initial version
2	2015-05-27	V1.1	Add a section of software and hardware changes
3	2015-07-17	V1.2	Supplemental register description
4	2016-02-23	V1.3	Fix some register description errors, modify "software and hardware modification instructions"
5	2016-04-13	V1.4	Revise the description of HT receiving window, modify the "software and hardware modification instructions"
6	2016-06-21	V1.5	Correct CLKSEL [15] description
7	2016-10-10	V1.6	Added software configuration HT frequency description
8	2017-02-14	V1.7	Modify the "Software and Hardware Changes Instructions" to add SPI address space instructions
9			

Manual information feedback: service@loongson.cn

You can also use the problem feedback website http://bugs.loongnix.org/Submit the use process of chip products to our company Problems and obtain technical support.

Page 5

Godson **3A2000 / 3B2000** processor user manual directory

table of Contents

Page 6

Page 7

Figure catalog

v

Page 10

Table directory

Page 13

X

Page 15

Godson **3A2000 / 3B2000** Processor User Manual Part 1

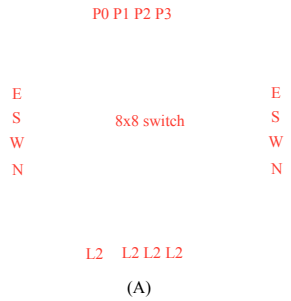
1 Overview

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 Yes, some low-end Loongson 3 can also be used for some desktop applications. The above three series will be developed in parallel.

Loongson No. 3 multi-core series processor is based on a scalable multi-core interconnect architecture design, integrating multiple high-end on a single chip Performance processor core and a large number of level 2 caches, and also realize the interconnection of multiple chips through high-speed I / O interface to form a larger Modular system.

The scalable interconnection structure adopted by Loongson 3 is as follows Picture 1-1 As shown. 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.



(A) (B) (C)

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.

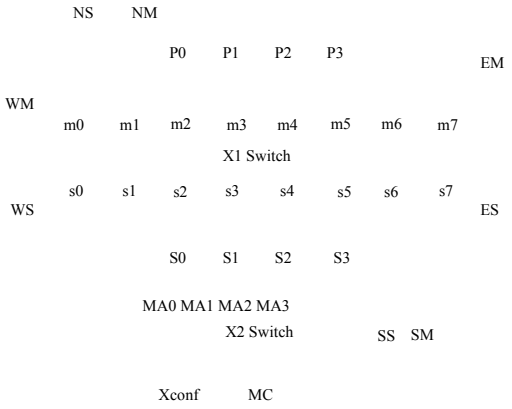


Figure 1-2 Loongson No. 3 node structure

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 3A2000 / 3B2000

Loongson 3A2000 / 3B2000 is a micro-structured upgraded version of Loongson 3A1000 quad-core processor. The package pins and Loongson

3A1000 compatible. Loongson 3A2000 / 3B2000 is a single-node 4-core processor, using 40nm process technology

The main frequency is 800MHz-1GHz, and the main technical characteristics are as follows:

- Four 64-bit super-scalar GS464e high-performance processor cores are integrated on-chip;
- On-chip integrated 4 MB split shared three-level cache (composed of 4 individual modules, each module has a capacity of 1MB);

- 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;
- 3B2000 integrates two 16-bit 1.6GHz HyperTransport controllers (hereinafter referred to as HT);
- 3A2000 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 3A1000, the main improvements are as follows:

- Comprehensive upgrade of the processor core microstructure;
- The memory controller structure and frequency are fully upgraded;
- HT controller structure and frequency are fully upgraded;
- The internal interconnection structure is fully upgraded;
- The external expansion interconnection structure is fully upgraded;
- Support SPI start function;
- Support full chip software frequency configuration;
- The performance of the whole chip is optimized and improved.

Godson 3A2000 Chip architecture is based on two integrally interconnected to achieve the following structure in FIG 1-3 FIG.

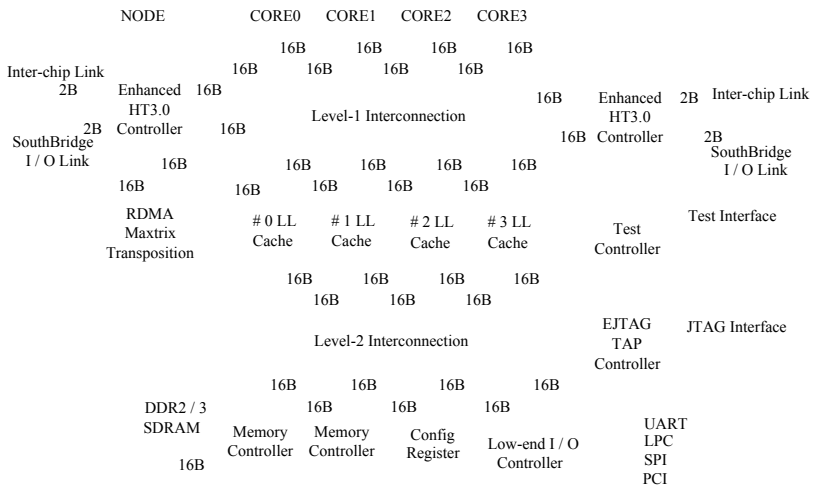


Figure 1-3 Godson 3A2000 chip structure

13

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).

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, connecting 4 shared Cache modules (as the main 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.

The difference between 3B2000 and 3A2000 processors is that 3B2000 supports the use of HT0 as a consistent interconnect interface. Based on Loongson No. 3 scalable interconnect architecture, 4 quad-core Loongson 3B2000 can be connected via HT port to form 4 chip 16 The NUMA structure of the core. The 3A2000 processor only supports the IO use of the HT1 controller.

In the following, there will be no difference between 3B2000 and 3A2000, referred to as Loongson 3A2000.

1.3 Description of Loongson 3A2000 Commercial and Industrial Chips

Loongson 3A2000 chips are available in both industrial and commercial grades. Their main features are as follows:

Configuration	Commercial grade	Industrial grade
Operating temperature	0 ℃ ～ 70 ℃	-40 ℃ ～ 85 ℃
Whether to filter	—	√
Whether the quality consistency test	—	√
Quality consistency test standard	—	GB 4937-1995

The Loongson 3A chip, like most semiconductor devices, has a failure rate that conforms to the bathtub curve model. Loongson 3A industrial grade chip In order to ensure longer-term, stable, and reliable operation, and to be able to adapt to more demanding environmental temperature requirements, the chip Reliability screening was conducted to eliminate early failure chips. This reliability screening is a 100% test, passed the screening To meet the requirements of industrial grade chips.

The operating voltage of 3A2000 commercial grade and industrial grade chips is slightly different. Industrial-grade chips require an operating voltage of 1.15V, The power supply jitter is less than 50mV; the commercial-grade chip requires an operating voltage of 1.25V, and the power supply jitter is less than 50mV.

The main contents of the Godson 3A screening test are as follows:

Filter items	Methods and conditions (Summary)	Claim
1. Visual inspection	The logo is clear, no contamination, no solder ball oxidation, and the chip is intact	100%

2. Stability baking	125 ℃, 24h	100%
3. Rapid temperature changes	10 cycles at maximum and minimum storage temperature	100%
4. Serial number		100%
5. Intermediate (before aging) electrical testing		100%
6, veteran	TC = 85 ℃, 160h	100%
7. Intermediate (after aging) electrical test at room temperature		100%
8. Permitted non-conforming product rate (PDA) Calculation	PDA≤5%, normal temperature, when 5% ＜ PDA≤10%, it can be Newly submitted and refined, but only allowed once	All batches
9. End point electrical test	Three temperature, record all test data	100%
10. External visual inspection	The logo is clear, no contamination, no solder ball oxidation, and the chip is intact	100%

15

2 System configuration and control

2.1 Chip working mode

According to the structure of the system, Loongson 3A2000 mainly includes three working modes:

- Single chip mode. The system only contains one Loongson 3A2000, which is a symmetric multiprocessor system (SMP);
- Multi-chip interconnect mode. The system contains 2 pieces or 4 pieces Godson 3A2000, through the HT end of Godson 3A2000 It is a non-uniform memory access 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

HT clock control

signal	effect
CLKSEL [15]	1'b1 means the HT PLL frequency is only set by hardware 1'b0 means HT PLL frequency can be set by software 1'b1 means HT PLL uses normal clock input

CLKSEL [15: 0]	CLKSEL [14]	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

	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 2.6 Description 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 means FIFO depth is set to 2 5'b011x0 means DCDL 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

IO configuration control

PCI_CONFIG [7: 0]	7 HT bus cold start is forced to 1.0 mode 6: 4 needs to be set to 000 3 PCI master mode

Page 22

2 needs to be set to 0

1 Use external PCI arbitration

0 Use SPI boot function

2.3 Cache consistency

Loongson 3A2000 maintains the cache consistency between the processor and the I / O accessed through the HT port by hardware, but The hardware does not maintain the cache consistency of I / O devices connected to the system through PCI. During driver development, When PCI access devices perform DMA (Direct Memory Access) transmission, the software needs to perform Cache consistency maintain.

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 to System development is compatible with expansion. The physical address width of the entire system is 48 bits. According to the upper 4 bits of the address, the entire address is empt Time is evenly distributed to 16 nodes, that is, each node is allocated 44-bit address space.

Loongson 3A2000 processor can directly use 4-chip direct connection to build CC-NUMA system, the processor number of each chip Determined by the pin NODEID, 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 3A2000 uses a single node 4-core configuration, so Loongson 3A2000 chip integrated DDR memory controller, HT The corresponding addresses of the bus and PCI bus are contained in the 44-bit field from 0x0 (inclusive) to 0x1000_0000_0000 (not included) Within each node, the 44-bit address space is further evenly distributed to a maximum of 8 devices that may be connected within the node. Among them, the lower 43 bits of addresses are owned by 4 shared cache modules, and the higher 43 bits of addresses are further according to the address [43:42] bits Distribute to devices connected to 4 directional ports. According to the different configuration of the chip and system structure, if there is no If there is a connected slave device, the corresponding address space is reserved address space, and access is not allowed.

Page 23

The slave devices corresponding to the address space of the first-level crossbar in the Loongson 3A2000 chip 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 3A2000 can decide to share based on the actual application access behavior
Cache cross-addressing mode. The 4 shared Cache modules in the node correspond to a total of 43 bits of address space, and each module
The address space corresponding to the block is determined according to one of the two selection bits of the address bit, and can be dynamically configured and repaired by software
change. The configuration register named SCID_SEL is set in the system to determine the address selection bits, as shown in the following table. By default
In the case of [7: 6] status hash distribution, that is, the two addresses [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 Address Routing Distribution and Configuration

The routing of Loongson 3A2000 is mainly realized through the two-stage crossbar of the system. One-level crossbar can
The master port receives requests for routing configuration. Each master port has 8 address windows, which can be completed
Target routing in 8 address windows. Each address window consists of three 64-bit registers BASE, MASK and MMAP,
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 corresponding target
Slave port number, MMAP [4] means to allow instruction fetch, MMAP [5] means to allow block read, MMAP [6] means to allow pair
Scache's interleaved access is enabled, MMAP [7] indicates that the window is enabled.

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

[7]	[6]	[5]	[4]
Window enable allows interleaved access to SCACHE. It is valid when the slave number is 0, according to the address		Allow block read	Allow instruction fetch

A section of SCID_SEL configuration routes requests that hit window addresses

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 First-level crossbar address window register table

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

Page 25

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:

Page 27

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

Slave device number default	
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]
The window enable allows interleaved access to DDR. It is valid when the slave device number is 0		Allow block read, allow instruction fetch	

Select bit configuration to route requests that hit the window address. The interleaving enable bit is required

Greater than 10

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

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
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

twenty three

Page 28

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

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-

The 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 Secondary 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.)

2.6 Chip configuration and sampling register

The chip configuration register (Chip_config) and chip sampling register (Chip_sample) in Loongson 3A2000 provide 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 divider
47-		RW	1'b1	
63:56	Cpu_version	R	2'h37	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 ℃)
103: 96	Thsens0_out	R	Temperature sensor 0 Celsius Junction temperature = Thens0_out -100 Temperature range -40 degrees – 125 degrees

111: 104	Thsens1_out	R	Temperature sensor 1 Celsius Junction temperature = Thens1_out -100 Temperature range -40 degrees – 125 degrees
other		R	Keep

The following groups 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 configuration method of HT CLOCK is special, please refer to the specific configuration method in Section 10.5.28 .

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) Other registers in the setting register except SEL_PLL_ * and SOFT_SET_PLL, that is, these two registers
The register is written as 0 during the setting process;

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

3) The lock signal LOCKED_ * in the waiting register is 1;

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

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	-	RW	0x0	-
2	SOFT_SET_PLL	RW	0x0	Allow software to set PLL
3	BYPASS_L1	RW	0x0	Bypass L1 PLL
6: 4	-	RW	0x0	-
7	LOCKEN_L1	RW	0x0	Allow lock L1 PLL
9: 8	-	RW	0x0	-
11:10	LOCKC_L1	RW	0x0	Determine whether the L1 PLL locks the used phase Accuracy
15:12	-	RW	0x0	-
16	LOCKED_L1	R	0x0	Whether L1 PLL is locked
18:17	-	R	0x0	-
19	PD_L1	R / W	0x0	Turn off L1 PLL
31:20	-	RW	0x1	-

Page 32

38:32	L1_DIV_LOOPC	RW	0x1	L1 PLL input parameters
41:39	-			-
47:42	L1_DIV_OUT	RW	0x1	L1 PLL input parameters
Other	-	RW		Keep

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

The VCO frequency of the PLL (in parentheses in the above formula) must be within the range of 1.2GHz-3.2GHz. The requirement The same applies to MEM PLL and HT PLL.

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
3	LOCKEN_MEM_PLL	RW	0x0	Allow to lock MEM_PLL
5: 4	LOCKC_ MEM_PLL	RW	0x0	Determine whether the MEM PLL locks the phase used Bit precision
6	LOCKED_MEM_PLL	R	0x0	Whether MEM_PLL is locked
7	PD_MEM_PLL	RW	0x0	Turn off MEM PLL
13: 8	-	RW	0x1	-
23:14	MEM_PLL_DIV_LOOPC	RW	0x41	MEM PLL input parameters
29:24	MEM_PLL_DIV_OUT	RW	0x0	MEM PLL input parameters
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 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 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

28

Page 33

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

29

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 and shared Cache modules in Loongson 3A2000 form one through AXI interconnection network Multi-core structure of 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. For more detailed introduction, please refer to the GS464e user manual and MIPS64 user manual.

30

Page 35

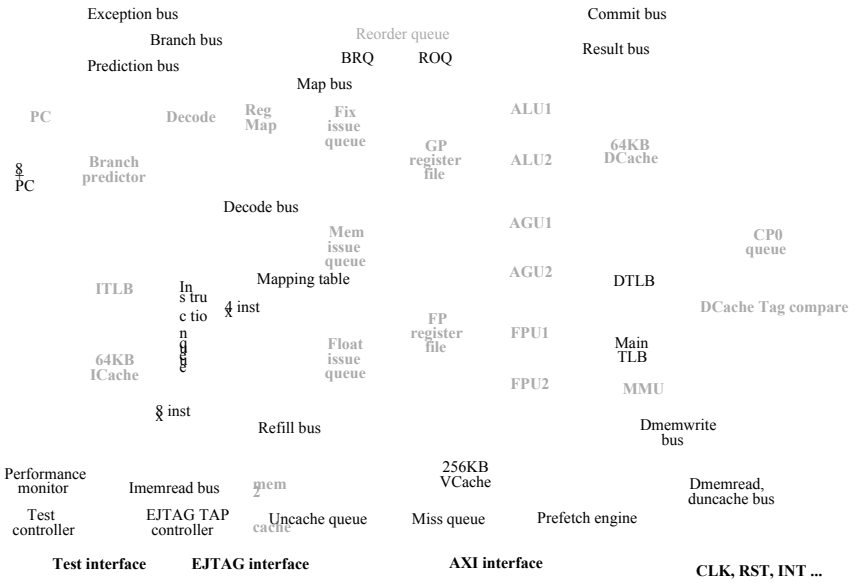


Figure 3-1 GS464e structure diagram

31

Page 36

4 Shared Cache (SCache)

The SCache module is a three-level cache shared by all processor cores within the Loongson 3A2000 processor. SCache module

The main features 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 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

Piece). Through the chip configuration register space, four groups of lock window registers in the shared Cache module can be dynamically configured

However, it must be ensured that one of the 16 shared caches is not locked. The size of each group of windows can be based on

Make adjustments, but not more than 3/4 of the entire shared cache size. In addition, when the shared cache receives the DMA write request,

If the written area hits and is locked in the shared cache, the DMA write will be written directly to the shared cache instead of

32

Page 37

RAM.

Table 4-1 Shared Cache Lock Window Register Configuration

name	address	Bit field	description
Slock0_valid	0x3ff00200	[63:63]	Lock window 0 valid bits
Slock0_addr	0x3ff00200	[47: 0]	No. 0 lock window lock address
Slock0_mask	0x3ff00240	[47: 0]	No. 0 lock window mask
Slock1_valid	0x3ff00208	[63:63]	Lock window 1 valid bit
Slock1_addr	0x3ff00208	[47: 0]	No. 1 lock window lock address
Slock1_mask	0x3ff00248	[47: 0]	No. 1 lock window mask
Slock2_valid	0x3ff00210	[63:63]	Lock window 2 valid bits
Slock2_addr	0x3ff00210	[47: 0]	No. 2 lock window lock address
Slock2_mask	0x3ff00250	[47: 0]	No. 2 lock window mask
Slock3_valid	0x3ff00218	[63:63]	Lock window 3 valid bits
Slock3_addr	0x3ff00218	[47: 0]	No. 3 lock window lock address
Slock3_mask	0x3ff00258	[47: 0]	No. 3 lock window mask

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.

Page 38

5 Matrix processing accelerator

Godson 3A2000 has built-in two matrix processing accelerators independent of the processor core. Its basic function is through software
The configuration of the matrix realizes the function of transposing or moving the matrix stored in the memory from the source matrix to the target matrix. Two
The accelerators are integrated in the two HyperTransport controllers of Loongson 3A2000, which are realized by a first-level cross switch
Read and write to 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 the read and write efficiency
Rate, you need to read in multiple rows of data, so that these data can be written in Cache rows in the transposed matrix
Input, 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
(Buffer), vertical readout (written from the buffer to the target matrix).

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

The source matrix involved in transposing or moving may be a small matrix located in a large matrix, so its matrix address
It may not be completely continuous. There will be gaps between the addresses of adjacent rows, and more programming control interfaces need to be implemented. The table below
5-1 to 5-4 illustrate the programming interfaces involved in matrix processing.

Table 5-1 Matrix processing programming interface description

address	name	Attributes	Explanation
0x3ff00600	src_start_addr	RW	Source matrix start address
0x3ff00608	dst_start_addr	RW	Target matrix start address
0x3ff00610	row	RW	Number of elements in a row of the source matrix
0x3ff00618	col	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	width	RW	Row span of the large matrix where the target matrix is located (bytes)
0x3ff00630	trans_ctrl	RW	Transpose control register
0x3ff00638	trans_status	RO	Transpose Status Register

Table 5-2 Matrix processing register address description

address	name
0x3ff00600	Src_start_addr of transpose module 0
0x3ff00608	Dst_start_addr of transpose module 0
0x3ff00610	Row 0 transpose module
0x3ff00618	Col of transpose module 0
0x3ff00620	Length of transposed module 0
0x3ff00628	Width of transposed module 0
0x3ff00630	Trans_ctrl of transpose module 0
0x3ff00638	Trans_status of transpose module 0
0x3ff00700	Src_start_addr of transpose module 1
0x3ff00708	Dst_start_addr of transpose module 1

0x3ff00710	Src_row_stride of transpose module 1
0x3ff00718	Src_last_row_addr of transpose module 1
0x3ff00720	The length of transpose module 1
0x3ff00728	Width of transpose module 1
0x3ff00730	Trans_ctrl of transpose module 1
0x3ff00738	Trans_status of transpose module 1

Table 5-3 Trans_ctrl 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	Arcmd, read command internal control bit. When arcache is 4'hf, it must be set to 4'hc. It is meaningless when arcache is other value.
11..8	Arcache, 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	trans_yes, 1 means transpose; 0 means no transpose

Table 5-4 Trans_status register description

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

36

Page 41

Godson **3A2000 / 3B2000** Processor User Manual Part 1

6 Inter-processor interrupt and communication

Godson 3A2000 implements 8 inter-core interrupt registers (IPI) for each processor core to support oot

Interrupt and communication between the processor cores when the mobile and operating system are runr ıs and addresses, see 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_Clear	W	32-bit clear register, write 1 to the corresponding bit, the corresponding STATUS register Bit cleared 0
MailBox0	RW	Cache register, used to transfer parameters at startup, according to 64 or 32 bit Uncache access.
MailBox01	RW	Cache register, used to transfer parameters at startup, according to 64 or 32 bit Uncache access.
MailBox02	RW	Cache register, used to transfer parameters at startup, according to 64 or 32 bit Uncache access.
MailBox03	RW	Cache register, used to transfer parameters at startup, according to 64 or 32 bit Uncache access.

The registers and functions of the interrupts between Loongson 3A2000 and processor cores are described as follows:

Table 6-2 Interrupt and communication register list of processor core 0

name	address	Authority	description
Core0_IPI_Status	0x3ff01000	R	IPI_Status register of processor core 0
Core0_IPI_Enalbe	0x3ff01004	RW	IPI_Enalbe register of processor core 0
Core0_IPI_Set	0x3ff01008	W	IPI_Set register of processor core 0
Core0 _IPI_Clear	0x3ff0100c	W	IPI_Clear register of processor core 0
Core0_MailBox0	0x3ff01020	RW	IPI_MailBox0 register of processor core 0
Core0_ MailBox1	0x3ff01028	RW	IPI_MailBox1 register of processor core 0
Core0_ MailBox2	0x3ff01030	RW	IPI_MailBox2 register of processor core 0
Core0_ MailBox3	0x3ff01038	RW	IPI_MailBox3 register of processor core 0

37

Page 42

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

name	address	Authority	description
Core1_IPI_Status	0x3ff01100	R	IPI_Status register of processor core 1
Core1_IPI_Enalbe	0x3ff01104	RW	IPI_Enalbe register of processor core 1
Core1_IPI_Set	0x3ff01108	W	IPI_Set register of processor core 1
Core1 _IPI_Clear	0x3ff0110c	W	IPI_Clear register of processor core 1
Core1_MailBox0	0x3ff01120	R	IPI_MailBox0 register of processor core 1
Core1_ MailBox1	0x3ff01128	RW	IPI_MailBox1 register of processor core 1
Core1_ MailBox2	0x3ff01130	W	IPI_MailBox2 register of processor core 1
Core1_ MailBox3	0x3ff01138	W	IPI_MailBox3 register of processor core 1

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

name	address	Authority	description
Core2_IPI_Status	0x3ff01200	R	IPI_Status register of processor core 2
Core2_IPI_Enalbe	0x3ff01204	RW	IPI_Enalbe register of processor core 2
Core2_IPI_Set	0x3ff01208	W	IPI_Set register of processor core 2
Core2 _IPI_Clear	0x3ff0120c	W	IPI_Clear register of processor core 2
Core2_MailBox0	0x3ff01220	R	IPI_MailBox0 register of processor core 2
Core2_ MailBox1	0x3ff01228	RW	IPI_MailBox1 register of processor core 2
Core2_ MailBox2	0x3ff01230	W	IPI_MailBox2 register of processor core 2
Core2_ MailBox3	0x3ff01238	W	IPI_MailBox3 register of processor core 2

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

name	address	Authority	description
Core3_IPI_Status	0x3ff01300	R	IPI_Status register of processor core 3
Core3_IPI_Enalbe	0x3ff01304	RW	IPI_Enalbe register of processor core 3
Core3_IPI_Set	0x3ff01308	W	IPI_Set register of processor core 3
Core3 _IPI_Clear	0x3ff0130c	W	IPI_Clear register of processor core 3
Core3_MailBox0	0x3ff01320	R	IPI_MailBox0 register of processor core 3
Core3_ MailBox1	0x3ff01328	RW	IPI_MailBox1 register of processor core 3
Core3_ MailBox2	0x3ff01330	W	IPI_MailBox2 register of processor core 3
Core3_ MailBox3	0x3ff01338	W	IPI_MailBox3 register of processor core 3

Listed above are the inter-core interrupt related messages for a single-node multiprocessor system composed of a single Loongson 3A2000 chip Memory list. When using multiple Loongson 3A2000 interconnects to form a multi-node CC-NUMA system, the node pairs in each chip Should be a system global node number, the IPI register address of the processor core in the node is based on the above table and the base of the node

38

Page 43

The addresses are in a fixed offset relationship. For example, the IPI_Status address of processor core 0 in node 0 is 0x3ff01000, and 1

The address of the No. 0 processor of the No. node is 0x10003ff01000, and so on.

39

7 I / O interrupt

Loongson 3A2000 chip supports up to 32 interrupt sources, which are managed in a unified manner, as shown in Figure 7-1 below, any An IO interrupt source can be configured as enabled, triggered, and routed to the processor core interrupt pin.

HT-1 INT7	31		
...	...		
HT-1 INT0	twenty four	IP0	
HT-0 INT7	twenty three	IP1 IP2 IP3	CORE 0
...	...		
HT-0 INT0	16		
		IP0	

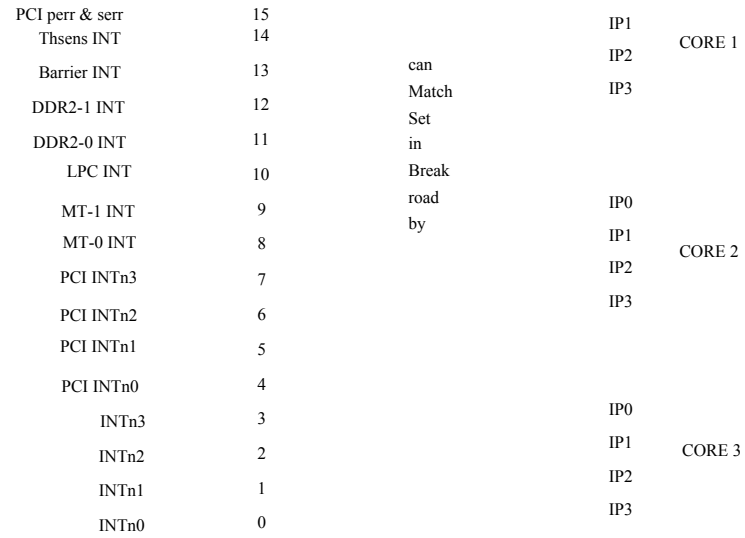


Figure 7-1 Loongson 3A2000 processor interrupt routing diagram

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

The clear interrupt is enabled, and the interrupt corresponding to the bit written in the Intenclr register is cleared. Inten register reads the current interrupt

Enabled situation. The interrupt signal in the form of pulse (such as PCI_SERR) is selected by the Intedge configuration register, write 1

Display pulse trigger, write 0 to indicate level trigger. The interrupt handler can clear the pulse record through the corresponding bit of Intenclr

record.

Table 7-1 Interrupt Control Register

Bit field	Access properties / default				
	Intedge	Inten	Intenset	Intenclr	Interrupt source
3: 0	RW / 0	R / 0	W / 0	W / 0	Sys_int0-3
7: 4	RO / 0	R / 0	RW / 0	RW / 0	PCI_INTn
8	RO / 0	R / 0	RW / 0	RW / 0	Matrix_int0
9	RO / 1	R / 0	RW / 0	RW / 0	Matrix_int1
10	RO / 1	R / 0	RW / 0	RW / 0	Lpc
12: 11	RW / 0	Keep	Keep	Keep	Mc0-1
13	RW / 0	R / 0	RW / 0	RW / 0	Barrier
14	RW / 0	R / 0	RW / 0	RW / 0	Thsens int
15	RW / 0	R / 0	RW / 0	RW / 0	Pci_perr
23: 16	RW / 0	R / 0	RW / 0	RW / 0	HT0 int0-7
31: 24	RW / 0	R / 0	RW / 0	RW / 0	HT1 int0-7

Table 7-2 IO Control Register Address

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

Intenset	0x3ff01428	32-bit setting enable register
Intenclr	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 3A2000. The above 32-bit interrupt sources can be selected through software configuration. The target processor core is expected to be interrupted. Further, the interrupt source can be selected to route to any of the processor core interrupts Meaning one, that is, IP2 to IP5 corresponding to CP0_Status. Each of the 32 I / O interrupt sources corresponds to an 8-bit path By the controller, its format and address are shown in Tables 7-3 and 7-4 below. The routing register is routed in a vector way Select, such as 0x48 to route to INT2 of processor 3.

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

42

8 Temperature sensor

8.1 Real-time temperature sampling

Loongson 3A2000 integrates two temperature sensors internally, which can be performed through the sampling register starting at 0x1FE00198
Observation, at the same time, can use the flexible high and low temperature interrupt alarm or automatic frequency modulation function to control. Temperature sensor in
The corresponding bits of the sampling register are as follows (base address is 0x1FE00198):

Table 8-1 Temperature sampling register description

Bit field	Field name	access	Reset value	description
24	Thsens0_overflow	R		Temperature sensor 0 overflow (over 125 ℃)
25	Thsens1_overflow	R		Temperature sensor 1 overflow (over 125 ℃)
39:32	Thsens0_out	R		Temperature sensor 0 Celsius Junction temperature = Thens0_out -100 Temperature range -40 degrees – 125 degrees
47:40	Thsens1_out	R		Temperature sensor 1 Celsius Junction temperature = Thens1_out -100 Temperature range -40 degrees – 125 degrees
other		R		Keep

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. Currently 3A2000 integrates two temperature sensors, this register is used for configuration selection
The temperature of which sensor is used 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.

43

The specific descriptions of these registers are as follows:

Table 8-2 High and low temperature interrupt register description

register	address	Control instructions	
			[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. Currently 3A2000 integrates two temperature sensors, this register is used for configuration selection The temperature of which sensor is used 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 Description of high-temperature down-frequency control register

register	address	Control instructions	
			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
High temperature down frequency control register			[59:58]: Scale_Sel3: Select the temperature sensor input source for high temperature down-conversion 3
Thsens_freq_scale	0x3ff01480	RW	[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 The number of (CAS_n) is 15, and the number of logical body 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 commands are combined and sorted to 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 automatically detect 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, CMD for short) consists 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 is composed of three signals. For write operations, RAS_n = 1, CAS_n = 0, and WE_n = 0. In addition, with the read operation Differently, write operations require DQM to identify the mask of the write operation, that is, the number of bytes to be written. DQM is the same as the DQs signal in the figure step.

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.

47

Page 52

Godson **3A2000 / 3B2000** Processor User Manual Part 1

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_value_0 (RD) /	Dll_adj_cnt	Dll_value_ck (RD)		Dll_init_done (RD)		Version (RD)	
0x008 Dll_value_4 (RD)		Dll_value_3 (RD)		Dll_value_2 (RD)		Dll_value_1 (RD)	
0x010 Dll_value_8 (RD)		Dll_value_7 (RD)		Dll_value_6 (RD)		Dll_value_5 (RD)	
0x018 Dll_ck_3	Dll_ck_2	Dll_ck_1	Dll_ck_0	Dll_increment	Dll_start_point	Dll_bypass	Init_start
0x020 Dq_oe_end_0	Dq_oe_begin_0	Dq_stop_edge_0	Dq_start_edge_0	Rddata_delay_0	Rddqs_lt_half_0	Wrdqs_lt_half_0	Wrdq_lt_half_0
0x028 Rd_oe_end_0	Rd_oe_begin_0	Rd_stop_edge_0	Rd_start_edge_0	Dqs_oe_end_0	Dqs_oe_begin_0	Dqs_stop_edge_0	Dqs_start_edge_0
0x030 Enzi_end_0	Enzi_begin_0	Wrclk_sel_0	Wrdq_clkdelay_0	Odt_oe_end_0	Odt_oe_begin_0	Odt_stop_edge_0	Odt_start_edge_0
0x038 Enzi_stop_0	Enzi_start_0		Dll_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_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_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_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_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_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_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_rddqs_n_7	Dll_rddqs_p_7	Dll_wrdqs_7	Dll_wrdata_7	Dll_gate_7

48

Page 53

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

Page 54

	63:56	55:48	47:40	39:32	31:24	23:16	15: 8	7: 0
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							tREF_low
0x328							
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_err or	Ecc_cnt_cs_3	Ecc_cnt_cs_2	Ecc_cnt_cs_1	Ecc_cnt_cs_0
0x368							

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:

51

Page 56

Internal reset Software enableDLL lock

POWER

Sys_reset

DDR_RESETn

Particle RESETn

(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 on the motherboard

DDR_RESETn is connected to the corresponding pin on the memory slot through an 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 status is low.

The timing is shown below:

Internal reset Software enableDLL lock

POWER

Sys_reset

DDR_RESETn

Particle RESETn

(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:

52

Internal reset Software enable DLL lock

POWER

Sys_reset

DDR_RESETn

Particle RESETn

By the combination of the latter two reset modes, it can be realized directly using the reset signal of the memory controller STR control. When the entire system is started from the shutdown state, use the method in (2) to use the memory module to reset normally and start working. When the system recovers from the STR, use the method in (3) to reconfigure the memory module so that Under the condition of destroying the original state of the memory module, it restarts to work normally.

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 Among them, only Write Leveling and Gate Leveling are implemented, Read Leveling is not implemented, the software needs to pass Judging the correctness of reading and writing to achieve the functions completed by Read Leveling. In addition to DQS operating during Leveling In addition to the phase and GATE phase, you can also calculate the write DQ phase and read DQ phase based on these last confirmed phases. Configuration 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 following 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;

53

(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, these

Registers 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 the sampling and reading DQS window in the controller. For software programming, refer to the following steps Step.

(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 is completed, perform two Lvl_req operations, 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 * set. 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 the command sending mode, you can

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;

55

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

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 be 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 the command sending mode, you can

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 respectively.

The device implements two independent control interfaces, one for direct control by the test port in the test mode, and the other

Used for configuration enable test by register configuration module in normal function mode.

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 same name part in the register parameter

Minute.

The two sets of interfaces are basically the same in terms of control parameters, only the access point is different. Here is the introduction of software programming

Control Method. 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 amount 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, you can

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

Error data and correct data at the time; 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. and also

That is, the access that caused the interruption is 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 3A2000, the HyperTransport bus is used to connect external devices and interconnect multiple chips. Used outside When setting up the connection, the user program can freely choose whether to support IO Cache consistency (through the address window Uncache Settings, see Section 10.5.13 for details): When configured to support Cache consistency mode, the IO device accesses the internal DMA Transparent at the Cache level, that is, the consistency is automatically maintained by the hardware, without the need for software to maintain through the program Cache instruction Protection; 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 inter-chip cache coherent transmission through pin configuration, while the HT1 controller (initial The address is 0x0E00_0000_0000 – 0x0FFF_FFFF_FFFF) can be configured by software to support inter-chip cache consistency For maintenance, see section 10.8 .

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. Change the rate and re-initialize, see section 10.1 for the specific method.

The main characteristics of Loongson 3A2000 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 HyperTransport 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,

By the width of the bus may register "Link Width Out" and "Link Width In" (see 10.5. Section 2) is read out.

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. Reinitialize

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

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

10.2 HyperTransport protocol support

Godson 3A2000's HyperTransport bus supports most commands in version 1.03 / 3.0 protocol, and is

Some extended instructions have been 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 Commands that the HyperTransport receiver can receive

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	1 – Doubleword
			bit 1: Don't Care	bit 1: Don't Care
			bit 0: Don't Care	bit 0: must be 1
01xxxx	NPC	Read	bit 3: Don't Care	bit 3: Don't Care
			bit 2: 0 – Byte	bit 2: 0 – Byte
			1 – Doubleword	1 – Doubleword
			bit 1: Don't Care	bit 1: Don't Care
			bit 0: 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 5: Must be 1, POSTED
			bit 2: 0 – Byte 1 – Doubleword	bit 2: 0 – Byte 1 – Doubleword

61

Page 66

		bit 0: must be 0	bit 0: must be 1
		bit 2: 0 – Byte	bit 2: 0 – Byte
010x0x	NPC Read	1 – Doubleword	1 – Doubleword
		bit 0: 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
111011	NPC RdAddr	----	Read address extension
111111	- Sync / Error	Will only forward	

10.3 HyperTransport interrupt support

HyperTransport controller provides 256 interrupt vectors, which can support Fix, Arbiter, etc.

However, there is no support for hardware automatic EOI. For the above two supported types of interrupts, the controller is receiving

After that, it will be automatically written into the interrupt register, and the system interrupt controller will be interrupted according to the setting of the interrupt mask register.

Notice. See specific interrupt control 10 .5 Section 8 of the interrupt control register group.

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 3A2000 HyperTransport controller will automatically process the first 3 steps and write the PIC interrupt vector

Corresponding position in 256 interrupt vectors. After processing the interrupt, the software system needs to perform step 4 processing, namely

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 3A2000 processor, the default address windows of the four HyperTransport interfaces are distributed as follows:

Table 10-4 Default address window distribution of the four HyperTransport 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

62

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 HyperTransport interface of Godson 3 processor

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

The HyperTransport interface of Loongson 3A2000 processor provides a variety of rich address windows for users to use.

The functions and functions of these address windows are described in the following table.

Table 10-6 Address window provided in the HyperTransport interface of Loongson 3A2000 processor

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 on Access in these address windows will be internal Received and processed by the bus, other access will be Will return an error according to the agreement.
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 Operation 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 After this write access will enter the processor Row write access complete response.

63

Page 68

Prefetch window (See window configuration 10.5.12)	2	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 configuration 10.5.13)	2	HyperTransport	Determine whether to HyperTransport Access operations on the bus For internal Uncache access	IO inside Loongson 3A2000 processor DMA access, in case will be Cache access is judged by SCache Break is a hit, thus maintaining its IO consistency information. And through the configuration of these windows, You can make access hits in these windows to Uncache way to directly access memory, Without maintaining its IO consistency letter through hardware interest.

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 software-visible configurations for controlling various working modes of the system register.

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	The interrupt routing method selection register corresponds 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)

Page 69

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)
0x154	HT bus receive address window 4 base address (external access)
0x80	HT bus interrupt vector register [31: 0]

0x84	Interrupt vector register	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	The PHY impedance matching configuration register is used to configure the impedance matching configuration of the PHY transmitter and receiver	
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 Control 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 Command, Capabilities Pointer, Capability 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 three	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 Link Config, Link Control 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 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	When using the 16-bit HT bus to run the 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 Revision ID, Link Freq, Link Error, Link Freq 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
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

Page 73

Reset value: 0x00000002

Name: Feature Capability

Table 10-12 Feature Capability register definition

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	Put the bus into LDT DISCONNECT mode The correct method is: 0-> 1
29	Ldt Req Gen	1	0x0	R / W	Wake up HT bus from 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
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

Page 74

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) Whether it is processor consistency mode

twenty two	Coherent Mode	1	0x0	R	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	rx_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	rx_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	Corresponding to 3 interrupt routing methods, see 0 interrupt direction for specific description 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

71

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 is enabled (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
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

Page 78

Bit field	Bit field name	Bit width	reset value	Visit description
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

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

74

Page 79

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 mapping sequence is {INTD,

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

Internal vector [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 HT bus interrupt vector register definition (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 Definition of HT bus interrupt vector register (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 HT bus interrupt vector register definition (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 configuration of the interrupt routing mode selection, with

The body 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

77

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 HT Bus Interrupt Enable Register Definition (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]

78

Page 83

Table 10-36 Definition of HT Bus Interrupt Enable Register (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 Definition of HT Bus Interrupt Enable Register (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 HT Bus Interrupt Enable Register Definition (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

79

Page 84

Offset: 0xbc

Reset value: 0x00000000

Name: HT Bus Interrupt Enable Register [255: 224]

Table 10-41 Definition of HT Bus Interrupt Enable Register (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 Interrupt Capability 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 Dataport 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 IntrInfo register definition (1)

Bit field	Bit field name	Bit width	reset value	Visit	description
31:24	IntrInfo [31:24]	32	0xF8	R	Keep

Bit field	Bit field name	Bit width	reset value	Visit	description
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 IntrInfo 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 the external 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)

81

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 the external 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 prefetch address window 0 enabled (internal access)

Table 10-50 HT Bus Prefetchable Address Window 0 Enable (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 prefetch the address window 1, the translated address [39:24]

Offset: 0xec

Reset value: 0x00000000

Name: HT bus prefetchable 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

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 invalidate the first-level CACHE, but will be sent directly to memory or other address space.

That is, the read and write commands in the address window will not maintain the CACHE consistency of IO. This window is mainly aimed at some CACHE hits 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]
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_ base [39:24]	16	0x0	R / W HT bus uncache address window 3, address base address
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_p2p_image0_en 1		0x0	R / W HT bus P2P address window 0, enable signal
30	ht_p2p_image0_ trans_en	1	0x0	R / W HT bus P2P address window 0, mapping enable signal
29: 0	ht_p2p_image0_ 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_p2p_image0_ base [39:24]	16	0x0	R / W HT bus P2P address window 0, address base address [39:24]
15: 0	ht_p2p_image0_ mask [39:24]	16	0x0	R / W HT bus P2P address window 0, 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_p2p_image1_en	1	0x0	R / W	HT bus P2P address window 1, enable signal
30	ht_p2p_image1_trans_en	1	0x0	R / W	HT bus P2P address window 1, mapping enable signal
29: 0	ht_p2p_image1_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
87					

Bit field	Bit field name	Bit width	reset value	Visit	description
31:16	ht_p2p_image1_base [39:24]	16	0x0	R / W	HT bus P2P address window 1, address base address [39:24]
15: 0	ht_p2p_image1_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.

88

Adjust the number of different send buffers.

Offset: 0x108

Reset value: 0x00000000

Name: Send cache 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	Increase and decrease the number of R channel data buffers at the sending end 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	Number of data buffers at the sender's NPC channel 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	Increase or decrease the number of PC channel data buffers at the sending end 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	Number of increase and decrease of the command buffer of the B channel of the sending end 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	Increase and decrease the number of R channel command buffers at the sending end 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	Number of increase / decrease of NPC channel command / data buffer at the sending end 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	Increase or decrease the number of PC channel command buffers at the sending end 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

89

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 Revision 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 Error Retry 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	R / W	The maximum number of Short Retry allowed by

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 Retry Count 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 Link Train 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 sender is in the Disconnected or Inactive state Link status: 2'b00 LS1 2'b01 LS0 2'b10 LS2 2'b11 LS3
14	Dsiable Cmd Throttling	1	0x0	R / W	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	R / W	The receiver is in Disconnected or Inactive state 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	R / W	Whether to enable Scramble 0: Disable Scramble 1: enable Scramble
2	8B10B Enable	1	0x0	R / W	Whether to enable 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 Training 0 Timeout 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 Training 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 counting register

Table 10-76 Training 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 counting register

Table 10-77 Training 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 counting register

Table 10-78 Training 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

When CLKSEL [15] is pulled low, it is used to switch the controller to any protocol and PLL support during operation

The link frequency and the controller frequency are supported; when CLKSEL [15] is pulled high, it has no effect.

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

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

Frequency division 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 When set to counter select, the upper limit of the counter is {PLL_relock_counter, 5'h1f} Otherwise, the upper limit of the count is 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 high-order

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

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 mode Under the formula, it is automatically selected as CLKPAD: 1'b0 external clock source 1'b1 PLL clock
28	Rx_ckdll_sell	1	0x0	R / W	Clock selection for locking DLL: 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	Visit	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 completed by the controller clock

10.5.31 LDT debug register

After the software changes the controller frequency, it will result in inaccurate timing of the LDT reconnect phase.

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

Controller clock.

Offset: 0x184

Reset value: 0x00000000

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

It is 0xFD_FE00_0000 to 0xFD_FFFF_FFFF. For configuration access in HT protocol, it is adopted in Godson 3A2000

Implemented in the following format:

Type 0:

Type 1:

Figure 10-1 HT protocol configuration access in Loongson 3A2000

10.7 HyperTransport bus frequency software configuration method

The frequency of the HyperTransport interface bus can be controlled by two methods, the first is to use the CLKSEL pin

PLL frequency configuration, then the configuration register Link Freq (offset 0x48, see 10. 5.2) bus is provided actual frequency

Rate method; the other is to use the software frequency configuration register (see 10.0.52) inside the controller to set the PLL and total

The actual frequency of the line can be obtained in a richer frequency combination than the pin setting method. Both methods are ultimately needed

Only one software reset or LDT RECONNECT will take effect.

The method of using the CLKSEL pin configuration is relatively simple and will not be described here. Use software register to set

Method 10.5 .28 section specifically noted, this in addition to some of the places that require special attention be specified.

When splitting a 16-bit HT into two 8-bit uses, only the software frequency configuration register of HT LO can

PLL and bus frequency division control, including HT HI bus frequency division value. In other words, if no treatment is taken, when HT

Resetting the frequency by LO will also cause the frequency of HT HI to change. At this time, if HT HI is in normal work

In the enabled state, the bus may be unstable.

To avoid this situation, there are two methods that can be used.

The first is to connect the reset signals of all HT together, so that the software frequency of all HT controllers is configured

After the register configuration is completed, pull the HT reset signal low, and then pull high to re-shake. This will make HT LO

Switch the clock at the same time as HT HI to ensure the normal operation of the system. This method is suitable for four-way interconnection

HT0 connection in the system.

The other is when the reset signal of HT cannot be connected together. At this time, it needs to be prevented by software control

HT LO affects HT HI when switching the PLL frequency. The most direct way is to first set HT HI to reset state,

97

Until HT LO finishes switching the PLL frequency, then reset the HT HI bus. This method is suitable for HT1 LO

Connect the bridge, HT1 HI cross interconnection.

10.8 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

Connected, where HTx_LO is the master device and HTx_HI is the slave device, and the interconnection structure as shown below is obtained

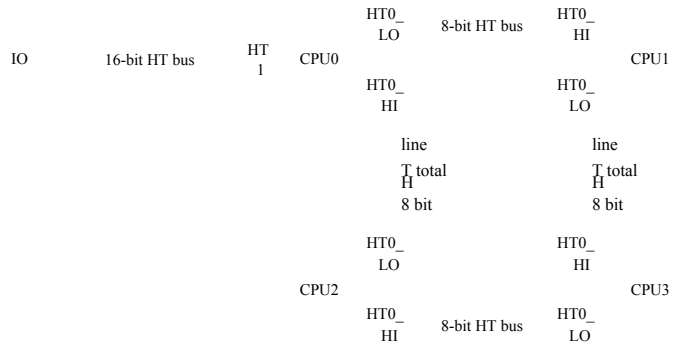


Figure 10-2 Four-piece Loongson No. 3 interconnection structure

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.

98

Page 103

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. first of all

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

The numbers are 00 and 01 respectively. From the routing algorithm, we can know that when two chips access each other, they are connected to

8-bit HT bus at the same time. As follows:

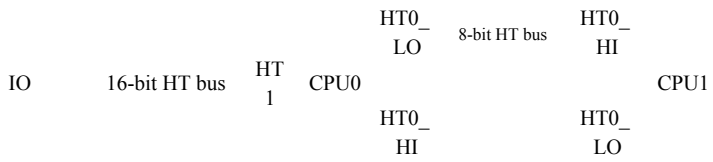


Figure 10-3 Two-chip Loongson No. 3 8-bit interconnection structure

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

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 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

99

Page 104

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

100

Page 105

...	
PCIX Command Register	E0
PCIX Status Register	E4

The PCIX controller of Loongson 3A2000 supports three 64-bit windows, composed of {BAR1, BAR0}, {BAR3, BAR2}, {BAR5, BAR4} 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	tar_read_io	Read and write (Write 1 clear)	0	Target end receives access to IO or non-prefetchable area
30	tar_read_discard	Read and write (Write 1 clear)	0	The delay request on the target side is discarded
29	tar_resp_delay	Read and write	0	When target access is given delay / split 0: After timeout 1: right away
28	tar_delay_retry	Read and write	0	target access retry strategy 0: According to internal logic (see bit 29) 1: Retry now
27	tar_read_abort_en	Read and write	0	If the target times out for internal read requests, whether to let target-abort respond
26:25	Reserved	-	0	
24	tar_write_abort_en	Read and write	0	If the target's internal write request times out, whether to respond with target-abort
23	tar_master_abort	Read and write	0	Whether to allow master-abort
22:20	tar_subseq_timeout	Read and write	000	target subsequent delay timeout 000: 8 cycles Other: Not supported
19:16	tar_init_timeout	Read and write	0000	target 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.
15: 4	tar_pref_boundary	Read and write	000h	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 Prefetchable boundary configuration (in units of 16 bytes) FFF: 64KB to 16byte FFE: 64KB to 32byte FF8: 64KB to 128byte
3	tar_pref_bound_en	Read and write	0	Configuration using tar_pref_boundary 0: prefetch to device boundary 1: Use tar_pref_boundary
2	Reserved	-	0	
1	tar_splitw_ctrl	Read and write	0	target split write control 0: Block access other than Posted Memory Write 1: Block all access until the split is completed
0	mas_lat_timeout	Read and write	0	Disable mater access timeout 0: Allow master access timeout 1: not allowed
REG_44				
31: 0	Reserved	-	-	

REG_48			
31: 0 tar_pending_seq	Read and write	0	target unprocessed request number bit vector The corresponding bit can be cleared by writing 1
REG_4C			
31:30 Reserved	-	-	
29 mas_write_defer	Read and write	0	Allow subsequent reads to skip past unfinished writes (Only valid for PCI)
28 mas_read_defer	Read and write	0	Allow subsequent reads and writes to bypass previous unfinished reads (Only valid for PCI)
27 mas_io_defer_cnt	Read and write	0	Maximum number of IO requests out 0: controlled by 1: 1
26:24 mas_read_defer_cnt	Read and write	0,1 0	The maximum number of master supports reading outside (only valid for PCI) 0: 8 1-7: 1-7 Note: A dual address cycle access accounts for two
23:16 err_seq_id	Read only	00h	target / master error number
15 err_type	Read only	0	Command type of target / master error 0:
14 err_module	Read only	0	The wrong module

			0: target 1: master
13 system_error	Read and write	0	Target / master system error (write 1 clear)
12 data_parity_error	Read and write	0	Target / master data parity error (write 1 clear)
11 ctrl_parity_error	Read and write	0	Target / master address parity error (write 1 clear)
10: 0 Reserved	-	-	
REG_50			
31: 0 mas_pending_seq	Read and write	0	Vector of unprocessed request number of master The corresponding bit can be cleared by writing 1
REG_54			
31: 0 mas_split_err	Read and write	0	split returns the wrong request number vector
REG_58			
31:30 Reserved	-	-	
29:28 tar_split_priority	Read and write	0	target split returns priority 0 highest, 3 lowest
27:26 mas_req_priority	Read and write	0	master external priority 0 highest, 3 lowest
25 Priority_en	Read and write	0	Arbitration algorithm (arbitration between master's access and target's split return) 0: fixed priority 1: rotation
24:18 Reserved	-	-	
17 mas_retry_aborted	Read and write	0	master retry cancellation (write 1 to clear)
16 mas_trdy_timeout	Read and write	0	master TRDY timeout count
15: 8 mas_retry_value	Read and write	00h	master retries 0: unlimited retry 1-255: 1-255 times
7: 0 mas_trdy_count	Read and write	00h	master 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 .

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

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 Note that doing so may make it abnormal. Using mandatory priority allows these devices to occupy a line.

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. Internal PCI arbiter provides two docking modes Type: The last master device and the default master device. If you cannot dock in special occasions, you can set the arbiter to Docking to the default master device 0 (internal controller) and relying on delay 0.

Page 109

11.2 LPC controller

The LPC controller has the following characteristics:

- Conform to LPC1.1 specification
- Support LPC access timeout counter
- Supports Memory Read and Memory write access types
- Support Firmware Memory Read, Firmware Memory Write access type (single byte)
- Supports I / O read and I / O write access types
- Support memory access type address conversion
- Support Serizlized 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 Boot	0X1FC0_0000-0X1FD0_0000	1MByte
LPC Memory	0X1C00_0000-0X1D00_0000	16MByte
LPC I / O	0X1FF0_0000-0X1FF1_0000	64KByte
LPC Register	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 Firmware Memory access is issued when the LPC_ROM_INTEL pin is pulled up, and issued when the LPC_ROM_INTEL pin is pulled down The LPC Memory access type is displayed.

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

Page 110

Bit field	Field name	Access reset value description	
		REG0	
REG0 [31:31]	SIRQ_EN	Read-write 0	SIRQ enable control
REG0 [23:16]	LPC_MEM_TRANS	Read-write 0	LPC Memory Space Address Translation Control

REG0 [15: 0]	LPC_SYNC_TIMEOUT	Read-write 0	LPC access timeout counter
		REG1	
REG1 [31:31]	LPC_MEM_IS_FWH	Read-write 0	LPC Memory Space Firmware
			Memory access type settings
REG1 [17: 0]	LPC_INT_EN	Read-write 0	LPC SIRQ interrupt enable
		REG2	
REG2 [17: 0]	LPC_INT_SRC	Read-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 receiving / sending
- Programmable data format
- 16-bit programmable clock counter
- Support receiving timeout detection
- Multi-interrupt system with arbitration
- Only work in FIFO mode
- Compatible with NS16550A in register and function

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	Reserved	4	RW	Keep
3	IME	1	RW	Modem status interrupt enable '0' – close '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	Reserved	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 trigger level	Less number of characters in FIFO Value for trigger
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	3rd	Transfer, save, deposit The device is empty	Transfer save register is empty	Write data to THR or Multi IIR
0	0	0	4th	Modem status	CTS, DSR, RI or 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 FIFO trigger value for interrupt request '00' – 1 byte '01' – 4 bytes '10' – 8 bytes '11' – 14 bytes
5: 3	Reserved	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	Reserved	1	W	Keep

11.3.5 Line Control Register (**LCR**)

Chinese name: Line Control Register

Register bit width: [7: 0]

108

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

109

Page 114

'10' – 7 digits '11' – 8 digits

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

Chinese name: Modem control register

Register bit width: [7: 0]

Offset: 0x04

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7: 5	Reserved	3	W	Keep
4	Loop	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
110				

Page 115

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

111

'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: Modem 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: Frequency Division Latch 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: Frequency Division Latch 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 transmission
- Main mode support
- Mode failure generates an error flag and issues an interrupt request
- Double buffer receiver
- Serial clock with programmable polarity and phase
- Can control SPI 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 Boot	0X1FC0_0000-0X1FD0_0000	1MByte
SPI Memory	0X1D00_0000-0X1E00_0000	16MByte
SPI Register	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 Memory space can also be accessed directly 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	Spie	1	RW	Interrupt output enable signal is high and effective
6	spe	1	RW	System work enable signal is highly effective

5	Reserved	1	RW	Keep
4	mstr	1	RW	master mode selection bit, this bit keeps 1
3	cpol	1	RW	Clock polarity bit
2	cpha	1	RW	Clock phase bit 1 is the opposite phase, and 0 is the same
1: 0	spr	2	RW	sclk_o crossover setting, need to be used with sper spre

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	spif	1	RW	Interrupt flag bit 1 indicates that there is an interrupt request, write 1 to clear
6	wcol	1	RW	Write register overflow flag bit is 1 indicates that it has overflowed, write 1 to clear
5: 4	Reserved	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	rffull	1	RW	Read register full flag 1 means full
0	rfempty	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	icnt	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	Reserved	4	RW	Keep
1: 0	spre	2	RW	Set the frequency division ratio with Spr

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	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

115

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

116

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 11-6 These registers are listed, and Table 11-7 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	PonCfg	Power-on configuration
04	GenCfg	General configuration
08	Keep	
0C	Keep	

10	PCIMap	PCI mapping
14	PCIX_Bridge_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_Win_Base_L	Prefetch the lower 32 bits of the base address of the window
44	Mem_Win_Base_H	Pre-fetch window base 32 higher bits
48	Mem_Win_Mask_L	Prefetchable window mask lower 32 bits
4C	Mem_Win_Mask_H	Pre-fetch window mask high 32 bits
50	PCI_Hit0_Sel_L	PCI window 0 controls the lower 32 bits

54	PCI_Hit0_Sel_H	PCI window 0 controls the upper 32 bits
58	PCI_Hit1_Sel_L	PCI Window 1 controls the lower 32 bits
5C	PCI_Hit1_Sel_H	PCI Window 1 controls the upper 32 bits
60	PCI_Hit2_Sel_L	PCI Window 2 controls the lower 32 bits
64	PCI_Hit2_Sel_H	PCI Window 2 controls the upper 32 bits
68	PXArb_Config	PCIX arbiter configuration
6C	PXArb_Status	PCIX arbiter status
70		
74		
78		
7C		
80	Chip Config	Chip configuration register
84		
88		
8C		
90	Chip Sample	Chip sampling register

Table 11-8 Register detailed description

Bit field	Field name	access	Reset value	Explanation
CR00: PonCfg				
15: 0 pcix_bus_dev		Read-only	lio_ad [7: 0]	In PCIX Agent mode, the total CPU usage

			Line, equipment number
15: 8	Keep	Read-only lio_ad [15: 8]	
23:16	pon_pci_configi	Read-only pci_configi	PCI_Configi pin value
31:24	Reserved	Read only	
CR04: reserved			
31: 0	Keep	Read only 0	
CR08: reserved			
31: 0	Keep	Read only 0	
CR10: PCIMap			
5: 0	trans_lo0	Read-write 0	PCI_Mem_Lo0 window map address high 6 bits
11: 6	trans_lo1	Read-write 0	PCI_Mem_Lo1 window map address high 6 bits

17:12	trans_lo2	Read-write 0	PCI_Mem_Lo2 window map address high 6 bits
31:18	Reserved	Read only 0	
CR14: PCIX_Bridge_Cfg			
5: 0	pcix_rgate	Read and write 6'h18	Threshold for sending data to DDR2 in PCIX mode
6	pcix_ro_en	Read-write 0	Does the PCIX bridge allow write over read
31:18	Reserved	Read only 0	
CR18: PCIMap_Cfg			
15: 0	dev_addr	Read-write 0	The upper 16 bits of the AD line in PCI configuration
16	conf_type	Read-write 0	Configure the type of read and write
31:17	Reserved	Read only 0	
CR1C: GPIO_Data			
15: 0	gpio_out	Read-write 0	GPIO output data
31:16	gpio_in	Read-write 0	GPIO input data
CR20: GPIO_EN			
15: 0	gpio_en	Read and write FFFF	High is input, low output
31:16	Reserved	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_Hit * _Sel_ *			
0	Keep	Read only 0	
2: 1	pci_img_size	Read and write 2'b11	00: 32 bits; 10: 64 bits; others: invalid
3	pref_en	Read-write 0	Prefetch enable
11: 4	Keep	Read only 0	
62:12	bar_mask	Read-write 0	Window size mask (high order 1, low order 0)
63	burst_cap	Read and write 1	Whether to allow burst transfer
CR68: PXArb_Config			
0	device_en	Read and write 1	Permitted by external equipment
1	disable_broken	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_master	Read-write 0	Bus docking default master device number Starting from no device requesting the bus to triggering the docking default
7: 6	park_delay	Read and write 2'b11	Delay in device behavior 00: 0 cycles

119

Page 124

Godson **3A2000 / 3B2000** Processor User Manual Part 1

		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 rude_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 Reserved	Read only 0	
CR6C: PXArb_Status		
7: 0 broken_master	Read only 0	Damaged master device (cleared when changing the disable policy)
10: 8 Last_master	Read only 0	Last master device using the bus
31:11 Keep	Read only 0	
CR80: Chip config (see Section 2. 6)		
CR90: Chip Sample (see section 2.6)		
CRA0: Chip Sample (see Section 2. 6)		
CRB0: PLL config (see section 2. 6)		
CRC0: PLL config (see section 2. 6)		
CRD0: Core config (see section 2. 6)		

120

Page 125

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

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 127

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

123

Page 128

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 129

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 130

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 131

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 132

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

128

Page 133

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	

129

Page 134

Thsens_int_ctrl_Hi	0x3ff01460	RW	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 [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

Page 135

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

131

Page 136

			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
Thsens_freq_scale	0x3ff01480	RW	[62:60]: Scale_freq3: Frequency division value when frequency is reduced

132

Page 137

			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
DFD_PARAM	0x3ff01500	RW	[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 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
CORE0_AWCOND0	0x3ff01800	RW	[49]: awready

			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
CORE0_AWMASK0	0x3ff01808	RW	(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	

			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
CORE0_ARCOND0	0x3ff01820	RW	[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	

135

Page 140

			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

136

Page 141

			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
			[17:16]: rresp
			[18]: rlast
			[19]: rrequest
			[21:20]: rstate
			[25:22]: rscseti
			[26]: rvalid
CORE0_RCOND0	0x3ff01880	RW	[27]: rready
			CORE0's AXI interface R trigger enable 0 setting, the highest bit is the R channel trigger enable
			[27: 0]: rmask
CORE0_RMASK0	0x3ff01888	RW	[63]: rchannel_en
CORE0_RCOND1	0x3ff01890	RW	
CORE0_RMASK1	0x3ff01898	RW	
CORE0_RCOND2	0x3ff018a0	RW	
CORE0_RMASK2	0x3ff018a8	RW	

Page 142

			TUD0 configuration register 0
			[47: 0]: count_target
TUD0_CONF0	0x3ff018e0	RW	[55:48]: monitor_enable
			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
TUD0_CONF1	0x3ff018e8	RW	[26]: cg_en
TUD0_RESULT	0x3ff018f0	R	TUD0 result register
CORE1_AWCOND0	0x3ff01900	RW	CORE1 AXI interface AW trigger condition 0 setting

138

Page 143

			CORE1 AXI interface AW trigger enable 0 is set, the highest bit is AW channel trigger enable
			The trigger condition is
CORE1_AWMASK0	0x3ff01908	RW	(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	

139

Page 144

CORE1_RCOND1	0x3ff01990	RW	
CORE1_RMASK1	0x3ff01998	RW	
CORE1_RCOND2	0x3ff019a0	RW	
CORE1_RMASK2	0x3ff019a8	RW	
			TUD1 configuration register 0
			[47: 0]: count_target
TUD1_CONF0	0x3ff019e0	RW	[55:48]: monitor_enable
			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
TUD1_CONF1	0x3ff019e8	RW	[26]: cg_en

140

Page 145

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

141

Page 146

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

142

Page 147

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	

143

Page 148

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	

Page 149

			TUD3 configuration register 0 [47: 0]: count_target
TUD3_CONF0	0x3ff01be0	RW	[55:48]: monitor_enable 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
TUD3_CONF1	0x3ff01be8	RW	[26]: cg_en
TUD3_RESULT	0x3ff01bf0	R	TUD3 result register
			TUD4 configuration register 0 [47: 0]: count_target

TUD4_CONF0	0x3ff01ce0	RW	[55:48]: monitor_enable

Page 150

			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
TUD4_CONF1	0x3ff01ce8	RW	[24]: cg_en
TUD4_RESULT	0x3ff01cf0	R	TUD4 result register
			TUD5 configuration register 0
			[47: 0]: count_target
TUD5_CONF0	0x3ff01de0	RW	[55:48]: monitor_enable

Page 151

			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

TUD5_CONF1	0x3ff01de8	RW	[24]: cg_en
TUD5_RESULT	0x3ff01df0	R	TUD5 result register
HT0_AWCOND0	0x3ff01e00	RW	HT0 AXI interface AW trigger condition 0 setting
			HT0's AXI interface AW trigger enable 0 setting, the highest bit is AW channel trigger enable
			The trigger condition is
HT0_AWMASK0	0x3ff01e08	RW	(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	

147

Page 152

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

148

Page 153

			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	

149

Page 154

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

150

Page 155

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

151

Page 156

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

152

Page 157

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

153

Page 158

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 159

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 160

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

159

Page 164

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

160

Page 165

13 Software and Hardware Design Guidelines

Loongson 3A2000 processor pins are downward compatible with Loongson 3A1000 processor, but the corresponding software and hardware need to be carried out Configuration changes to enable the original compatibility mode, or open some new features of Godson 3A2000, this chapter focuses on Compared with the Godson 3A1000, the software and hardware settings of the Godson 3A2000 processor are different.

13.1 Hardware modification guide

1. The original CORE_PLL_AVDD, DDR_PLL_AVDD, HT0 / 1_PLL_AVDD are now NC pins. If you use the original The 3A motherboard can be left unchanged. However, if you consider the compatibility with the future 3A3000, you can The pressure is modified to 1.8v, or a configurable design of 1.8v / 2.5v;

2. The original MC0 / 1_COMP_REF_RES is changed to NC pin. If the original 3A motherboard is used, no modification is required;

3. The original HT0 / 1_PLL_REF is changed to NC pin. If the original 3A motherboard is used, no modification is required;

4. The original MC0 / 1_COMP_REF_GND is changed to MC0 / 1_A15. If you use the original 3A motherboard, you don't need to repair it
Change; but if connected to a memory module, it can support a larger capacity

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. in case
Use the original 3A motherboard, you need to set it to 0 to start from LPC FLASH; if the motherboard already has SPI FLASH, you can
To connect GPIO [0] as SPI_CS and set PCI_CONFIG [0] to 1, start from SPI FLASH;

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 the 3A780E motherboard, you need to set it to 1 at present; if you use the 3A2H motherboard, no special settings are required;

7. CLKSEL [15:10] needs to be set to 6'b100001; if you need to use HT3.0 mode, you need to set CLKSEL [15:10]
Set to 6'b100101;

8. CLKSEL [9: 5] needs to be set to 5'b01111; use PMON to set the memory frequency;

9. CLKSEL [4: 0] needs to be set to 5'b01111; use PMON to set the processor core frequency;

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)

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 3A2000 is narrow.
To obtain a wider frequency range and better clock quality, the software configuration in PMON is mainly used in Godson 3A2000
The configuration method is the same as Loongson 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 1.15V core voltage: processor core frequency is set to 800MHz, memory frequency is set to 500MHz,
HT controller is set to 400MHz, HT bus 800MHz / 1600MHz;

13.3 PMON Change Guide

Compared with Loongson 3A1000, from the processor core, memory controller, HT controller to all levels of crossbar switches are different
Upgrade, so 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;

13.4 Guidelines for kernel changes

The modifications required in the kernel include:

1. Modify the Cache description structure in the kernel. Both VCache and SCache are connected using 16-way groups;

2. Modify the calculation method of the temperature sensor, which is the same as 3B1500 with the readout value -100. At present, the samples have not been tested and calibrated,
There may be a large deviation between the read value of some chips and the actual temperature, so it is recommended that in the current kernel, temporarily
Do not use the temperature indication of the internal temperature sensor of the processor;

3. Modify the configuration register address when shutting down the core;

4. Change the operation of flashing ICache / DCache to flashing ICache / DCache / VCache;

5. If an SPI controller is used, the base address is changed from 0xBFE001F0 to 0xBFE00220;

6. Uncache DMA must be used, and data consistency of Cache must be maintained by software;

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 between different cores is implemented using LL / SC instructions;

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;

9. Lock Cache operations cannot be used for DMA areas where hardware automatically maintains consistency.

Modifications that can also be used to improve performance are:

1. Increase support for FTLB;

2. Add support for TLB fast refill;

3. Add wait instruction support;

4. Add prefetch instruction support;

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.

13.5 Other changes

1. The performance counter overflow interrupt cannot achieve precise interrupts, resulting in restrictions on the current perf tools. If needed

To do so, frequent mfc0 perfcnt instructions must be added (user mode is available), for example, in the processing function of high-frequency clock interrupt

Insert the instruction in the data, but it will still cause the interruption to not be generated in time, and the event statistical error will be larger;