Page 1

Loongson 3A4000 processor

Register User Manual

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

V1.5

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

"Loongson 3A4000 Processor Register User Manual" introduces Loongson 3A4000 multi-core processor architecture and register description.

The chip system architecture, function and configuration of main modules, register list and bit fields are described in detail.

Page 4

revise history

Document update record		
	Document name:	Loongson 3A4000 processor register user manual
	version number	V1.5
	founder	Chip R & D Department
	Creation date	2019-12-20

Update history

Serial number	Updated	version number	update content
1	2018-05-08	V0.1	initial version
2	2018-05-28	V0.2	Update various chip configuration registers
3	2018-06-02	V0.3	Add frequency division control chapter, update GPIO, UART, I2C, SPI content
4	2018-06-04	V0.4	Modify route
5	2018-06-05	V0.5	Modify the memory controller section, add software clock system
6	2018-06-13	V0.6	Add clock description; add GPIO interrupt description; add temperature status detection; Add HT interrupt description; add 3A3000 compatible description; modify EXTIoi, Support 8-node interconnection; modify the processor core description.
7	2018-09-11	V0.7	Update the configuration register list and add some register field descriptions.
8	2018-09-13	V0.8	Updated stable clock structure description.

9	2018-10-26	V0.9	Update DDR section
10	2019-02-19	V1.0	Internal debug version
11	2019-05-29	V1.1	Update configuration register, temperature sensor, stable counter, expansion Interrupt, Scache interrupt, processor CPUCFG part description
12	2019-07-01	V1.2	Modified some statement errors
13	2019-09-11	V1.3	Chapter 4 adds some feature control instructions 11.1.1 Fix int_edge address offset
14	2019-10-11	V1.4	12.5 Correction of temperature sensor register description
15	2019-12-17	V1.5	Fix some format errors

Manual feedback: service@loongson.cn

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

Page 5

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

Page 6

table of Contents

Page 7

Figure catalog

Table directory

Page 17

Page 18

Page 20

Godson **3A4000** processor register user manual table list

XV

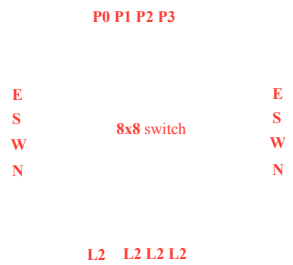
1 Overview

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 are 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 shown in Figure 1-1 below. Loongson No. 3 on-chip and multi-chip systems pass similar Of the interconnection ports are implemented in units of nodes, where each node consists of 8 * 8 crossbar switches, each The switch connects four processor cores and four shared caches, and connects the four directions of east (E) south (N) west (W) north (N) The other nodes are interconnected.



(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

1

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.

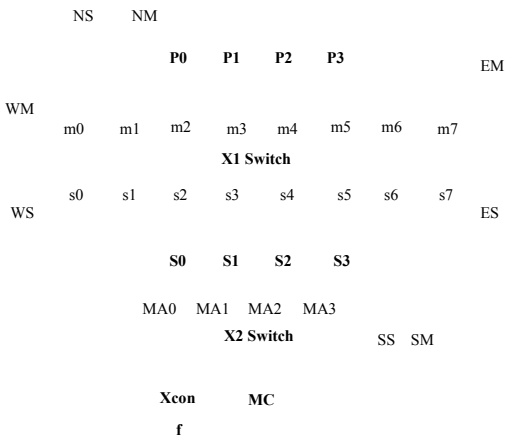


Figure 1-2 Loongson No. 3 node structure

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 is connected to a crossbar configuration module (Xconf), which is used for
Configure the X1 and X2 address windows of this node. You can also connect more memory controllers and IO ports as needed
Wait.

1.2 Introduction to Godson 3A4000

Loongson 3A4000 is a quad-core Loongson processor, which is manufactured using 28nm process and has a stable operating frequency of
1.5-2.0GHz, the main technical features are as follows:

- Four 64-bit super-scalar GS464V high-performance processor cores are integrated on-chip;

- On-chip integrated 8MB split shared three-level cache (composed of 4 individual modules, each module has a capacity of 2MB);
- Maintain the cache consistency of multi-core and I / O DMA access through the directory protocol;
- Two 64-bit DDR3 / 4 controllers with ECC and 800MHz are integrated on-chip;
- Two 16-bit HyperTransport controllers (hereinafter referred to as HT) are integrated on-chip;
- Each 16-bit HT port is split into two 8-way HT ports for use;
- Integrate 2 I2C, 1 UART, 1 SPI, 16 GPIO interfaces on-chip.

The top structure design of Loongson 3A4000 is optimized on a large scale based on 3A2000 / 3A3000.

Go as follows:

- Adjusted the on-chip interconnect structure, simplified the address routing, the interconnection between IO modules adopts RING structure;
- Optimized the bandwidth utilization and cross-chip delay of the HT controller;
- Optimized the structure of the memory controller, added support for the memory controller DDR4, and supported the memory slot to connect the accelerator card
- Standardize the configuration register space and access method, and introduce the CSR configuration register access mechanism;
- Optimized the structure of the interrupt controller and supported the vector interrupt hardware distribution mechanism
- Added 8-way interconnection support.

The overall architecture of Loongson 3A4000 chip is based on multi-level interconnection. The structure is shown in Figures 1-3 below.

Figure 1-3 Loongson 3A4000 chip structure

The first level interconnection uses a 5x5 crossbar switch to connect four GS464v cores (as the master device) and four shares

The Cache module (as a slave device) and an IO port are connected to the IO-RING. (IO port uses a Master and

A Slave).

3

The second level interconnection uses a 5x3 crossbar switch to connect 4 shared Cache modules (as the main device), two DDR3 / 4

The memory controller and an IO port are connected to IO-RING.

IO-RING contains 8 ports, the connection includes 4 HT controllers, MISC modules, SE modules and two stages are cross-connected

turn off. The two HT controllers (lo / hi) share the 16-bit HT bus, which is used as two 8-bit HT buses.

lo monopolizes the 16-bit HT bus. A DMA controller is integrated in the HT controller, and the DMA controller is responsible for the DMA control of the IO and

Responsible for maintaining consistency between slices.

The above interconnection structures all use separate data channels for reading and writing. The width of the data channel is 128 bits.

The same frequency of the core is used to provide high-speed on-chip data transmission. In addition, the first-level crossbar connects 4 processor cores with

The read data channel of scache is 256 bits to improve the read bandwidth of the on-chip processor core to access scache.

4

2 System configuration and control

2.1 Chip working mode

According to the structure of the system, Loongson 3A4000 mainly includes two working modes:

- Single chip mode. The system only contains one piece of Godson 3A4000, which is a symmetric multiprocessor system (SMP);
- Multi-chip interconnect mode. The system contains 2, 4, or 8 Loongson 3A4000, which communicate with each other through the HT port Connected to form a non-uniform memory multiprocessor system (CC-NUMA)

2.2 Description of control pins

The main control pins include DO_TEST, ICCC_EN, NODE_ID [2: 0], CLKSEL [9: 0], CHIP_CONFIG [5: 0].

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 [2: 0]		Indicates the processor number in multi-chip consistent interconnect mode

HT clock control

signal	effect
CLKSEL [9]	1'b1: indicates that the HT PLL clock is controlled by CLKSEL [9: 4] 1'b0: The initial frequency multiplication is 1 frequency multiplication, which can be reconfigured by software Set 1'b1: indicates that the HT PLL uses the SYSCLK clock input

CLKSEL [9: 0]	CLKSEL [8]	1'b0: indicates that the HT PLL uses a differential clock input
	CLKSEL [7: 6]	2'b00 means the PHY clock frequency is 1.6GHz 2'b01 indicates that the PHY clock frequency is 3.2GHz (the reference clock is 1.6GHz at 25MHz) 2'b10 means the PHY clock frequency is 1.2GHz 2'b11 means the PHY clock frequency is 2.4GHz
	CLKSEL [5]	1'b1: indicates that the HT PLL clock is in bypass mode, directly Use external input 100MHz / 25MHz reference clock

5

CLKSEL [4]	1-Reference clock is 25MHz, 0-Reference clock is 100MHz

MEM clock control (clock frequency should be 1/2 of interface clock)

CLKSEL [3: 2]	Output frequency
2'b00	466MHz
2'b01	600MHz
2'b10	Software configuration (PLL clock multiplier 1.6-3.2GHz)
2'b11	SYSCLK (100MHz / 25MHz)

Main clock control (highest frequency of network and processor core)

CLKSEL [1: 0]	Output frequency
2'b00	1GHz
2'b01	2GHz
2'b10	Software configuration (PLL clock multiplier 1.6-3.2GHz)
2'b11	SYSCLK (100MHz / 25MHz)

Chip configuration control

CHIP_CONFIG [5: 0]	CHIP_CONFIG [0] SE function enable CHIP_CONFIG [1] Default HT Gen1 mode CHIP_CONFIG [2] reserved CHIP_CONFIG [3] HT0 / 1-hi enters the consistency mode by default, which is used to support 8-way mutual even CHIP_CONFIG [4] HT logic function interchange, HT0 / HT1 exchange CHIP_CONFIG [5] On-chip clock debug enable (DCDL)

6

Page 27

3 Physical address space distribution

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.

3.1 Distribution of physical address space between nodes

Loongson 3A4000 processor can directly use 2/4/8 3A4000 chips to connect directly to build CC-NUMA system, each core The processor number of the chip is determined by the pin NODEID. The address space of each chip is distributed as follows:

Table 3-1 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
4	4	0x4000_0000_0000	0x4FFF_FFFF_FFFF
5	5	0x5000_0000_0000	0x5FFF_FFFF_FFFF
6	6	0x6000_0000_0000	0x6FFF_FFFF_FFFF
7	7	0x7000_0000_0000	0x7FFF_FFFF_FFFF

When the number of system nodes is less than 8 nodes, the nodemask field of the routing setting register (0x1fe00400) should be set, When a guess visit occurs, it is guaranteed that even if there is no physical node address, a response can be obtained. (2 channels: 0x1; 4 channels: 0x3)

3.2 Physical address space distribution within the node

Godson 3A4000 uses a single-node 4-core configuration, so Godson 3A4000 chip integrated DDR memory controller, HT The corresponding addresses of the bus are contained in the 44-bit address space from 0x0 (inclusive) to 0x1000_0000_0000 (not included). Within each node, the 44-bit address space is further divided among all devices connected within the node, only if the access type When it is cached, the request will be routed to 4 shared cache modules. According to the different configuration of chip and system structure, such as If a slave device is not connected to a port, the corresponding address space is reserved address space and access is not allowed.

7

Page 28

The slave devices corresponding to the address space of the Loongson 3A4000 chip are as follows:

Table 3-2 Address distribution within the node

device	Address [43:40] bits	Start address in the node	Node end address
MC0	4	0x400_0000_0000	0x4FF_FFFF_FFFF

MC1	5	0x500_0000_0000	0x5FF_FFFF_FFFF
SE	c	0xC00_0000_0000	0xCFF_FFFF_FFFF
HT0 Lo controller	a	0xA00_0000_0000	0xAFF_FFFF_FFFF
HT0 Hi controller	b	0xB00_0000_0000	0xBFF_FFFF_FFFF
HT1 Lo controller	e	0xE00_0000_0000	0xEFF_FFFF_FFFF
HT1 Hi controller	f	0xF00_0000_0000	0xFFF_FFFF_FFFF

Unlike the mapping relationship of direction ports, Loongson 3A4000 can decide to share based on the actual application access behavior Cache cross-addressing mode. The address space corresponding to the 4 shared Cache modules in the node is based on two of the address bits The bit selection bit is determined and can be dynamically modified by software. A configuration mail named SCID_SEL is set in the system Register to determine the address selection bits, as shown in the table below. By default, [7: 6] address hashing is used for distribution, That is, two bits of address [7: 6] determine the corresponding shared cache number. The register address is 0x3FF00400 or 0x1fe00400.

Table 3- 3 SCID_SEL address bit setting

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

The default distribution of the internal 44-bit physical address of each node of Loongson 3A4000 processor is shown in the following table:

Table 3-4 44-bit physical address distribution in the node

Address range	Access attribute	destination
addr [43:40] == 4'ha	Local node, uncache	HT0_LO
addr [43:40] == 4'hb	Local node, uncache	HT0_HI
addr [43:40] == 4'hc	Local node, uncache	SE
addr [43:40] == 4'he	Local node, uncache	HT1_LO
addr [43:40] == 4'hf	Local node, uncache	HT1_HI
0x10000000-0x1fffffff, 0x3ff00000-0x3ff0ffff (can be closed)	Local node, uncache	MISC
Mc interleave is 0, and not the above address	Local node, uncache	MC0
Mc interleave is 1, and not the above address	Local node, uncache	MC1
Scache interleave is 0 (address bit selection determined by scid_sel)	Local node, cache	Scache0
Scache interleave is 1 (selection of address bits determined by scid_sel)	Local node, cache	Scache1
Scache interleave is 2 (address bit selection determined by scid_sel)	Local node, cache	Scache2
Scache interleave is 3 (selection of address bits determined by scid_sel)	Local node, cache	Scache3

3.3 Address routing distribution and configuration

The routing of Loongson 3A4000 is mainly realized through the system's two-level crossbar switch and IO-RING. The software 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 is composed 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 four bits of MMAP indicate the corresponding target Slave port number, MMAP [4] means to allow instruction fetch, MMAP [5] means to allow block reading, MMAP [6] means to allow interleaving Access is enabled, MMAP [7] means the window is enabled.

Table 3-5 The space access attributes corresponding to the MMAP field

[7]	[6]	[5]	[4]
Window enable	Allow interleaved access to SCACHE / memory	Block read	Allow fetching

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

Since Loongson 3 uses fixed routing by default, the configuration window is closed when the power is turned on. System software is required to enable and configure it.

After the SCACHE / memory interleaving access configuration is enabled, the slave number is only valid when it is 0 or 4. 0 means route to SCACHE, and SCID_SEL decides how to interleave access in 4 SCACHEs. 4 means routing to memory, by interleave_bit determines how to interleave access between 2 MCs.

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

Table 3- 6 Address window register table

9

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

0x3ff0_2220	CORE2_WIN4_BASE	0x3ff0_2320	CORE3_WIN4_BASE
0x3ff0_2228	CORE2_WIN5_BASE	0x3ff0_2328	CORE3_WIN5_BASE
0x3ff0_2230	CORE2_WIN6_BASE	0x3ff0_2330	CORE3_WIN6_BASE
0x3ff0_2238	CORE2_WIN7_BASE	0x3ff0_2338	CORE3_WIN7_BASE
0x3ff0_2240	CORE2_WIN0_MASK	0x3ff0_2340	CORE3_WIN0_MASK
0x3ff0_2248	CORE2_WIN1_MASK	0x3ff0_2348	CORE3_WIN1_MASK
0x3ff0_2250	CORE2_WIN2_MASK	0x3ff0_2350	CORE3_WIN2_MASK
0x3ff0_2258	CORE2_WIN3_MASK	0x3ff0_2358	CORE3_WIN3_MASK
0x3ff0_2260	CORE2_WIN4_MASK	0x3ff0_2360	CORE3_WIN4_MASK

Page 31

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_2400	SCACHE0_WIN0_BASE	0x3ff0_2500	SCACHE1_WIN0_BASE
0x3ff0_2408	SCACHE0_WIN1_BASE	0x3ff0_2508	SCACHE1_WIN1_BASE
0x3ff0_2410	SCACHE0_WIN2_BASE	0x3ff0_2510	SCACHE1_WIN2_BASE
0x3ff0_2418	SCACHE0_WIN3_BASE	0x3ff0_2518	SCACHE1_WIN3_BASE
0x3ff0_2420	SCACHE0_WIN4_BASE	0x3ff0_2520	SCACHE1_WIN4_BASE
0x3ff0_2428	SCACHE0_WIN5_BASE	0x3ff0_2528	SCACHE1_WIN5_BASE
0x3ff0_2430	SCACHE0_WIN6_BASE	0x3ff0_2530	SCACHE1_WIN6_BASE
0x3ff0_2438	SCACHE0_WIN7_BASE	0x3ff0_2538	SCACHE1_WIN7_BASE
0x3ff0_2440	SCACHE0_WIN0_MASK	0x3ff0_2540	SCACHE1_WIN0_MASK
0x3ff0_2448	SCACHE0_WIN1_MASK	0x3ff0_2548	SCACHE1_WIN1_MASK
0x3ff0_2450	SCACHE0_WIN2_MASK	0x3ff0_2550	SCACHE1_WIN2_MASK
0x3ff0_2458	SCACHE0_WIN3_MASK	0x3ff0_2558	SCACHE1_WIN3_MASK
0x3ff0_2460	SCACHE0_WIN4_MASK	0x3ff0_2560	SCACHE1_WIN4_MASK
0x3ff0_2468	SCACHE0_WIN5_MASK	0x3ff0_2568	SCACHE1_WIN5_MASK
0x3ff0_2470	SCACHE0_WIN6_MASK	0x3ff0_2570	SCACHE1_WIN6_MASK
0x3ff0_2478	SCACHE0_WIN7_MASK	0x3ff0_2578	SCACHE1_WIN7_MASK
0x3ff0_2480	SCACHE0_WIN0_MMAP	0x3ff0_2580	SCACHE1_WIN0_MMAP
0x3ff0_2488	SCACHE0_WIN1_MMAP	0x3ff0_2588	SCACHE1_WIN1_MMAP
0x3ff0_2490	SCACHE0_WIN2_MMAP	0x3ff0_2590	SCACHE1_WIN2_MMAP
0x3ff0_2498	SCACHE0_WIN3_MMAP	0x3ff0_2598	SCACHE1_WIN3_MMAP
0x3ff0_24a0	SCACHE0_WIN4_MMAP	0x3ff0_25a0	SCACHE1_WIN4_MMAP
0x3ff0_24a8	SCACHE0_WIN5_MMAP	0x3ff0_25a8	SCACHE1_WIN5_MMAP
0x3ff0_24b0	SCACHE0_WIN6_MMAP	0x3ff0_25b0	SCACHE1_WIN6_MMAP
0x3ff0_24b8	SCACHE0_WIN7_MMAP	0x3ff0_25b8	SCACHE1_WIN7_MMAP

0x3ff0_2600	SCACHE2_WIN0_BASE	0x3ff0_2700	SCACHE3_WIN0_BASE
0x3ff0_2608	SCACHE2_WIN1_BASE	0x3ff0_2708	SCACHE3_WIN1_BASE

Page 32

0x3ff0_2610	SCACHE2_WIN2_BASE	0x3ff0_2710	SCACHE3_WIN2_BASE
0x3ff0_2618	SCACHE2_WIN3_BASE	0x3ff0_2718	SCACHE3_WIN3_BASE
0x3ff0_2620	SCACHE2_WIN4_BASE	0x3ff0_2720	SCACHE3_WIN4_BASE
0x3ff0_2628	SCACHE2_WIN5_BASE	0x3ff0_2728	SCACHE3_WIN5_BASE
0x3ff0_2630	SCACHE2_WIN6_BASE	0x3ff0_2730	SCACHE3_WIN6_BASE
0x3ff0_2638	SCACHE2_WIN7_BASE	0x3ff0_2738	SCACHE3_WIN7_BASE
0x3ff0_2640	SCACHE2_WIN0_MASK	0x3ff0_2740	SCACHE3_WIN0_MASK
0x3ff0_2648	SCACHE2_WIN1_MASK	0x3ff0_2748	SCACHE3_WIN1_MASK
0x3ff0_2650	SCACHE2_WIN2_MASK	0x3ff0_2750	SCACHE3_WIN2_MASK
0x3ff0_2658	SCACHE2_WIN3_MASK	0x3ff0_2758	SCACHE3_WIN3_MASK
0x3ff0_2660	SCACHE2_WIN4_MASK	0x3ff0_2760	SCACHE3_WIN4_MASK
0x3ff0_2668	SCACHE2_WIN5_MASK	0x3ff0_2768	SCACHE3_WIN5_MASK
0x3ff0_2670	SCACHE2_WIN6_MASK	0x3ff0_2770	SCACHE3_WIN6_MASK
0x3ff0_2678	SCACHE2_WIN7_MASK	0x3ff0_2778	SCACHE3_WIN7_MASK
0x3ff0_2680	SCACHE2_WIN0_MMAP	0x3ff0_2780	SCACHE3_WIN0_MMAP
0x3ff0_2688	SCACHE2_WIN1_MMAP	0x3ff0_2788	SCACHE3_WIN1_MMAP
0x3ff0_2690	SCACHE2_WIN2_MMAP	0x3ff0_2790	SCACHE3_WIN2_MMAP
0x3ff0_2698	SCACHE2_WIN3_MMAP	0x3ff0_2798	SCACHE3_WIN3_MMAP
0x3ff0_26a0	SCACHE2_WIN4_MMAP	0x3ff0_27a0	SCACHE3_WIN4_MMAP
0x3ff0_26a8	SCACHE2_WIN5_MMAP	0x3ff0_27a8	SCACHE3_WIN5_MMAP
0x3ff0_26b0	SCACHE2_WIN6_MMAP	0x3ff0_27b0	SCACHE3_WIN6_MMAP
0x3ff0_26b8	SCACHE2_WIN7_MMAP	0x3ff0_27b8	SCACHE3_WIN7_MMAP
-	-	0x3ff0_2900	IO_L2X_WIN0_BASE
-	-	0x3ff0_2908	IO_L2X_WIN1_BASE
-	-	0x3ff0_2910	IO_L2X_WIN2_BASE
-	-	0x3ff0_2918	IO_L2X_WIN3_BASE
-	-	0x3ff0_2920	IO_L2X_WIN4_BASE
-	-	0x3ff0_2928	IO_L2X_WIN5_BASE
-	-	0x3ff0_2930	IO_L2X_WIN6_BASE
-	-	0x3ff0_2938	IO_L2X_WIN7_BASE
-	-	0x3ff0_2940	IO_L2X_WIN0_MASK
-	-	0x3ff0_2948	IO_L2X_WIN1_MASK
-	-	0x3ff0_2950	IO_L2X_WIN2_MASK
-	-	0x3ff0_2958	IO_L2X_WIN3_MASK
-	-	0x3ff0_2960	IO_L2X_WIN4_MASK
-	-	0x3ff0_2968	IO_L2X_WIN5_MASK
-	-	0x3ff0_2970	IO_L2X_WIN6_MASK
-	-	0x3ff0_2978	IO_L2X_WIN7_MASK

12

Page 33

-	-	0x3ff0_2980	IO_L2X_WIN0_MMAP
-	-	0x3ff0_2988	IO_L2X_WIN1_MMAP
-	-	0x3ff0_2990	IO_L2X_WIN2_MMAP
-	-	0x3ff0_2998	IO_L2X_WIN3_MMAP

-	-	0x3ff0_29a0	IO_L2X_WIN4_MMAP
-	-	0x3ff0_29a8	IO_L2X_WIN5_MMAP
-	-	0x3ff0_29b0	IO_L2X_WIN6_MMAP
-	-	0x3ff0_29b8	IO_L2X_WIN7_MMAP

0x3ff0_2a00	HT0_LO_WIN0_BASE	0x3ff0_2b00	HT0_HI_WIN0_BASE
0x3ff0_2a08	HT0_LO_WIN1_BASE	0x3ff0_2b08	HT0_HI_WIN1_BASE
0x3ff0_2a10	HT0_LO_WIN2_BASE	0x3ff0_2b10	HT0_HI_WIN2_BASE
0x3ff0_2a18	HT0_LO_WIN3_BASE	0x3ff0_2b18	HT0_HI_WIN3_BASE
0x3ff0_2a20	HT0_LO_WIN4_BASE	0x3ff0_2b20	HT0_HI_WIN4_BASE
0x3ff0_2a28	HT0_LO_WIN5_BASE	0x3ff0_2b28	HT0_HI_WIN5_BASE
0x3ff0_2a30	HT0_LO_WIN6_BASE	0x3ff0_2b30	HT0_HI_WIN6_BASE
0x3ff0_2a38	HT0_LO_WIN7_BASE	0x3ff0_2b38	HT0_HI_WIN7_BASE
0x3ff0_2a40	HT0_LO_WIN0_MASK	0x3ff0_2b40	HT0_HI_WIN0_MASK
0x3ff0_2a48	HT0_LO_WIN1_MASK	0x3ff0_2b48	HT0_HI_WIN1_MASK
0x3ff0_2a50	HT0_LO_WIN2_MASK	0x3ff0_2b50	HT0_HI_WIN2_MASK
0x3ff0_2a58	HT0_LO_WIN3_MASK	0x3ff0_2b58	HT0_HI_WIN3_MASK
0x3ff0_2a60	HT0_LO_WIN4_MASK	0x3ff0_2b60	HT0_HI_WIN4_MASK
0x3ff0_2a68	HT0_LO_WIN5_MASK	0x3ff0_2b68	HT0_HI_WIN5_MASK
0x3ff0_2a70	HT0_LO_WIN6_MASK	0x3ff0_2b70	HT0_HI_WIN6_MASK
0x3ff0_2a78	HT0_LO_WIN7_MASK	0x3ff0_2b78	HT0_HI_WIN7_MASK
0x3ff0_2a80	HT0_LO_WIN0_MMAP	0x3ff0_2b80	HT0_HI_WIN0_MMAP
0x3ff0_2a88	HT0_LO_WIN1_MMAP	0x3ff0_2b88	HT0_HI_WIN1_MMAP
0x3ff0_2a90	HT0_LO_WIN2_MMAP	0x3ff0_2b90	HT0_HI_WIN2_MMAP
0x3ff0_2a98	HT0_LO_WIN3_MMAP	0x3ff0_2b98	HT0_HI_WIN3_MMAP
0x3ff0_2aa0	HT0_LO_WIN4_MMAP	0x3ff0_2ba0	HT0_HI_WIN4_MMAP
0x3ff0_2aa8	HT0_LO_WIN5_MMAP	0x3ff0_2ba8	HT0_HI_WIN5_MMAP
0x3ff0_2ab0	HT0_LO_WIN6_MMAP	0x3ff0_2bb0	HT0_HI_WIN6_MMAP
0x3ff0_2ab8	HT0_LO_WIN7_MMAP	0x3ff0_2bb8	HT0_HI_WIN7_MMAP

0x3ff0_2c00	SE_WIN0_BASE	0x3ff0_2d00	MISC_WIN0_BASE
0x3ff0_2c08	SE_WIN1_BASE	0x3ff0_2d08	MISC_WIN1_BASE
0x3ff0_2c10	SE_WIN2_BASE	0x3ff0_2d10	MISC_WIN2_BASE
0x3ff0_2c18	SE_WIN3_BASE	0x3ff0_2d18	MISC_WIN3_BASE
0x3ff0_2c20	SE_WIN4_BASE	0x3ff0_2d20	MISC_WIN4_BASE

0x3ff0_2c28	SE_WIN5_BASE	0x3ff0_2d28	MISC_WIN5_BASE
0x3ff0_2c30	SE_WIN6_BASE	0x3ff0_2d30	MISC_WIN6_BASE
0x3ff0_2c38	SE_WIN7_BASE	0x3ff0_2d38	MISC_WIN7_BASE
0x3ff0_2c40	SE_WIN0_MASK	0x3ff0_2d40	MISC_WIN0_MASK
0x3ff0_2c48	SE_WIN1_MASK	0x3ff0_2d48	MISC_WIN1_MASK
0x3ff0_2c50	SE_WIN2_MASK	0x3ff0_2d50	MISC_WIN2_MASK
0x3ff0_2c58	SE_WIN3_MASK	0x3ff0_2d58	MISC_WIN3_MASK
0x3ff0_2c60	SE_WIN4_MASK	0x3ff0_2d60	MISC_WIN4_MASK
0x3ff0_2c68	SE_WIN5_MASK	0x3ff0_2d68	MISC_WIN5_MASK
0x3ff0_2c70	SE_WIN6_MASK	0x3ff0_2d70	MISC_WIN6_MASK
0x3ff0_2c78	SE_WIN7_MASK	0x3ff0_2d78	MISC_WIN7_MASK
0x3ff0_2c80	SE_WIN0_MMAP	0x3ff0_2d80	MISC_WIN0_MMAP
0x3ff0_2c88	SE_WIN1_MMAP	0x3ff0_2d88	MISC_WIN1_MMAP
0x3ff0_2c90	SE_WIN2_MMAP	0x3ff0_2d90	MISC_WIN2_MMAP
0x3ff0_2c98	SE_WIN3_MMAP	0x3ff0_2d98	MISC_WIN3_MMAP
0x3ff0_2ca0	SE_WIN4_MMAP	0x3ff0_2da0	MISC_WIN4_MMAP

0x3ff0_2ca8	SE_WIN5_MMAP	0x3ff0_2da8	MISC_WIN5_MMAP
0x3ff0_2cb0	SE_WIN6_MMAP	0x3ff0_2db0	MISC_WIN6_MMAP
0x3ff0_2cb8	SE_WIN7_MMAP	0x3ff0_2db8	MISC_WIN7_MMAP

0x3ff0_2e00	HT1_LO_WIN0_BASE	0x3ff0_2f00	HT1_HI_WIN0_BASE
0x3ff0_2e08	HT1_LO_WIN1_BASE	0x3ff0_2f08	HT1_HI_WIN1_BASE
0x3ff0_2e10	HT1_LO_WIN2_BASE	0x3ff0_2f10	HT1_HI_WIN2_BASE
0x3ff0_2e18	HT1_LO_WIN3_BASE	0x3ff0_2f18	HT1_HI_WIN3_BASE
0x3ff0_2e20	HT1_LO_WIN4_BASE	0x3ff0_2f20	HT1_HI_WIN4_BASE
0x3ff0_2e28	HT1_LO_WIN5_BASE	0x3ff0_2f28	HT1_HI_WIN5_BASE
0x3ff0_2e30	HT1_LO_WIN6_BASE	0x3ff0_2f30	HT1_HI_WIN6_BASE
0x3ff0_2e38	HT1_LO_WIN7_BASE	0x3ff0_2f38	HT1_HI_WIN7_BASE
0x3ff0_2e40	HT1_LO_WIN0_MASK	0x3ff0_2f40	HT1_HI_WIN0_MASK
0x3ff0_2e48	HT1_LO_WIN1_MASK	0x3ff0_2f48	HT1_HI_WIN1_MASK
0x3ff0_2e50	HT1_LO_WIN2_MASK	0x3ff0_2f50	HT1_HI_WIN2_MASK
0x3ff0_2e58	HT1_LO_WIN3_MASK	0x3ff0_2f58	HT1_HI_WIN3_MASK
0x3ff0_2e60	HT1_LO_WIN4_MASK	0x3ff0_2f60	HT1_HI_WIN4_MASK
0x3ff0_2e68	HT1_LO_WIN5_MASK	0x3ff0_2f68	HT1_HI_WIN5_MASK
0x3ff0_2e70	HT1_LO_WIN6_MASK	0x3ff0_2f70	HT1_HI_WIN6_MASK
0x3ff0_2e78	HT1_LO_WIN7_MASK	0x3ff0_2f78	HT1_HI_WIN7_MASK
0x3ff0_2e80	HT1_LO_WIN0_MMAP	0x3ff0_2f80	HT1_HI_WIN0_MMAP
0x3ff0_2e88	HT1_LO_WIN1_MMAP	0x3ff0_2f88	HT1_HI_WIN1_MMAP
0x3ff0_2e90	HT1_LO_WIN2_MMAP	0x3ff0_2f90	HT1_HI_WIN2_MMAP

0x3ff0_2e98	HT1_LO_WIN3_MMAP	0x3ff0_2f98	HT1_HI_WIN3_MMAP
0x3ff0_2ea0	HT1_LO_WIN4_MMAP	0x3ff0_2fa0	HT1_HI_WIN4_MMAP
0x3ff0_2ea8	HT1_LO_WIN5_MMAP	0x3ff0_2fa8	HT1_HI_WIN5_MMAP
0x3ff0_2eb0	HT1_LO_WIN6_MMAP	0x3ff0_2fb0	HT1_HI_WIN6_MMAP
0x3ff0_2eb8	HT1_LO_WIN7_MMAP	0x3ff0_2fb8	HT1_HI_WIN7_MMAP

The second level xbar is mainly connected to 2 memory controllers and IO-RING as slave devices, and is composed of 4 Scache (4, representing 0x3ff0_4xxx, the same below, 5, 6, 7) and IO-RING (9) as the main device for window mapping, these

The window configuration register (4, 5, 6, 7, 9) performs memory window configuration and address conversion.

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:

[47:10]	[7: 4]	[3: 0]
Address after conversion	Window enable	Slave number

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

Table 3-7 Correspondence between the slave device number and the module

Slave number	Destination device
0-3	Scache0-3
4-5	MC0-1
a	HT0_lo
b	HT0_hi
c	SE
d	MISC
e	HT1_lo
f	HT1_hi

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

Table 3-8 The space access attributes corresponding to the MMAP field

[7]	[6]	[5]	[4]
Window enable	Allow interleaved access to DDR, valid when the slave device number is 0, according to the address	Block or not interleaved	Allow fetching
	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 cannot perform address translation on Cache consistency requests, otherwise it is in the SCache Will be inconsistent with the address of the first-level cache of the processor, resulting in incorrect maintenance of Cache consistency.

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

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

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 DDR, 0x10000000-0x17ffffff mapping Shot into the PCI_MEM space of the bridge, 0x18000000-0x19ffffff is mapped to the PCI_IO space of the bridge, 0x1a000000-0x1affffff is mapped to the PCI configuration space (Type0) of the bridge, and 0x1b000000-0x1bffffff The PCI configuration space (Type1) shot to the bridge, 0x40000000-0x7fffffff is mapped to the PCI_MEM space of the bridge. 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. Random data will be returned to prevent the CPU from dying, etc.

Page 37

4 Chip Configuration Register

The chip configuration register in Loongson 3A4000 provides a mechanism to read and write various functions of the chip. below Details each configuration register.

The base address of each chip configuration register in this chapter is 0x1fe00000.

4.1 Version register (0x0000)

The base address is 0x1fe00000 and the offset address is 0x0000.

Table 4- 1 Version Register

Bit field	Field name	access	Reset value	description
7: 0	Version	R	8'h10	Configuration register version number

4.2 Chip feature register (0x0008)

This register identifies some software-related processor features for software to view before enabling specific functions. Registered The base address is 0x1fe00000 and the offset address is 0x0008.

Table 4- 2 Chip feature register

Bit field	Field name	access	Reset value	description
0	Centigrade	R	1'b1	When it is 1, it means that CSR [0x428] is valid
1	Node counter	R	1'b1	When it is 1, it means that CSR [0x408] is valid
2	MSI	R	1'b1	When 1, it means MSI is available
3	EXT_IOI	R	1'b1	When 1, it means EXT_IOI is available
4	IPI_percore	R	1'b1	When it is 1, it means to enter through the CSR private address Send by IPI
5	Freq_percore	R	1'b1	When 1, it means that the CSR private address Whole frequency
6	Freq_scale	R	1'b0	When 1, it means that the dynamic crossover function is available
7	DVFS_v1	R	1'b0	When 1, it means that dynamic frequency modulation v1 is available
8	Tsensor	R	1'b0	When 1, it means the temperature sensor is available

4.3 Vendor name (0x0010)

This register is used to identify the name of the manufacturer. The base address is 0x1fe00000 and the offset address is 0x0010.

17

Page 38

Table 4- 3 Manufacturer Name Register

Bit field	Field name	access	Reset value	description

63: 0 Vendor	R	0x6e6f7367_6e6f6f4c the string "Loongson"	

4.4 Chip name (0x0020)

This register is used to identify the chip name. The base address is 0x1fe00000 and the offset address is 0x0020.

Table 4- 4 Chip name register

Bit field	Field name	access	Reset value	description
63: 0	ID	R	0x00003030_30344133	character string "3A4000"

4.5 Function setting register (0x0180)

The base address is 0x1fe00000 and the offset address is 0x0180.

Table 4- 5 Function Setting Register

Bit field	Field name	access	Reset value	description
0		RW	1'b0	
1		RW	1'b0	
3: 2		RW	2'b0	Keep
4	MC0_disable_confspace	RW	1'b0	Whether to disable MC0 DDR configuration space
5	MC0_defult_confspace	RW	1'b1	Route all memory access to configuration space
6	MCA0 clock en	RW	1'b1	MCA0 clock enable
7	MC0_resetn	RW	1'b1	MC0 software reset (active low)
8	MC0_clken	RW	1'b1	Whether to enable MC0
9	MC1_disable_confspace	RW	1'b0	Whether to disable MC1 DDR configuration space
10	MC1_defult_confspace	RW	1'b1	Route all memory access to configuration space
11	MCA1 clock en	RW	1'b1	MCA1 clock enable
12	MC1_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'b011	HT controller divide by 0
27	HT0_clken	RW	1'b1	Whether to enable HT0
30:28	HT1_freq_scale_ctrl	RW	3'b011	HT controller divided by 1
31	HT1_clken	RW	1'b1	Whether to enable HT1
42:40	Node_freq_ ctrl	RW	3'b111	Node crossover
43	-	RW	1'b1	
63:56	Cpu_version	R	2'h3B	CPU version

4.6 Pin drive setting register (0x0188)

The base address is 0x1fe00000 and the offset address is 0x0188.

Table 4- 6-pin drive setting register

Bit field	Field name	access	Reset value	description
31: 0				(air)
63:32	Pad1v8_ctrl	RW	32'h4f0000	1v8 pad control

4.7 Function sampling register (0x0190)

The base address is 0x1fe00000 and the offset address is 0x0190.

Table 4-7 Function Sampling Register

Bit field	Field name	access	Reset value	description
31: 0	Compcode_core	R		
37:32	Chip_config	R		Motherboard configuration control
47:38	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

4.8 Temperature sampling register (0x0198)

The base address is 0x1fe00000 and the offset address is 0x0198.

Table 4- 8 Temperature sampling register

Bit field	Field name	access	Reset value	description
15: 0		R		
19:16	Compcode_ok	R		
20	dotest	R		
twenty one	tsc_en	R		
23:22		R		
twenty four	Thsens0_overflow	R		Temperature sensor 0 overflow
25	Thsens1_overflow	R		Temperature sensor 1 overflow
31:26				
47:32	Thsens0_out	R		Temperature sensor 0 Celsius Knot point temperature = Thens0_out * 731 / 0x4000-273 Temperature range -40 degrees – 125 degrees
63:48	Thsens1_out	R		Temperature sensor 1 Celsius Knot point temperature = Thens1_out * 731 / 0x4000-273 Temperature range -40 degrees – 125 degrees

19

4.9 Bias Configuration Register (0x01A0)

The 3A4000 integrates a bias voltage generation module. The following registers are used to control these bias voltage modules. The base address is 0x1fe00000, the offset address is 0x1a0.

Table 4- 9 Bias Setting Register

Bit field	Field name	access	Reset value	description
0	BBGEN_enable	RW	0x0	Bias enable
1	BBMUX_first	RW	0x0	Set to switch voltage mode first
3: 2		RW	0x0	
7: 4	BBGEN_feedback	RW	0x0	Disable BBGEN feedback signal
11: 8	BBGEN_vbbp_val	RW	0x0	Setting value of Vbbp
15:12	BBGEN_vbbn_val	RW	0x0	Setting value of Vbbn
17:16	BBMUX_SEL_0	RW	0x0	BBMUX_SEL_0 setting value
19:18	BBMUX_SEL_1	RW	0x0	BBMUX_SEL_1 setting value
21:20	BBMUX_SEL_2	RW	0x0	BBMUX_SEL_2 setting value
23:22	BBMUX_SEL_3	RW	0x0	BBMUX_SEL_3 setting value
31:24		RW	0x0	Keep

40:32	BBGEN_sm	RO	0x0	The current state of the BBGEN state machine
other	-	RW		Keep

4.10 Frequency configuration register (0x01B0)

The following sets of software frequency multiplication setting registers are used to set the CLKSEL to software control mode (refer to section 2.2 CLKSEL setting method), the working frequency of the chip's main clock and the memory controller clock. Among them, MEM CLOCK configuration pair According to the memory controller clock frequency, the bus operating frequency is 2 times the clock, and the bus operating rate is 4 times the clock; NODE CLOCK corresponds to the clock frequency of the processor core, on-chip network, and high-speed shared cache.

Each clock configuration generally has three parameters, DIV_REFC, DIV_LOOPC, DIV_OUT. The final clock frequency is

(Reference clock / DIV_REFC * DIV_LOOPC) / DIV_OUT.

In software control mode, the default clock frequency is the frequency of the external reference clock (100MHz or 25MHz) To set the clock software during the processor startup. The process of setting each clock should follow the following methods:

1) The other registers in the setting register except SEL_PLL_ * and SOFT_SET_PLL, that is, these two The register is written as 0 during the setting process;
2) Other register values remain unchanged, set SOFT_SET_PLL to 1;
3) Wait for the lock signal LOCKED_ * in the register to be 1;
4) Set SEL_PLL_ * to 1, and the corresponding clock frequency will be switched to the frequency set by the software.

The following registers are the configuration registers of Main CLOCK. Main Clock is used to generate node clock and core clock The highest operating frequency. The base address is 0x1fe00000 and the offset address is 0x1b0:

Table 4-10 Node clock software frequency multiplication setting register

Bit field	Field name	access	Reset value	description
0	SEL_PLL_NODE	RW	0x0	Clock output selection 1: Node clock selects PLL output 0: Node clock selection SYS CLOCK
1		RW	0x0	
2	SOFT_SET_PLL	RW	0x0	Allow software to set PLL
3	BYPASS_L1	RW	0x0	Bypass L1 PLL
15: 4	-	RW	0x0	-
16	LOCKED_L1	R	0x0	Whether L1 PLL is locked
18:17	-	R	0x0	-
19	PD_L1	RW	0x0	Turn off L1 PLL
25:20		RW	0x0	
31:26	L1_DIV_REFC	RW	0x1	L1 PLL input parameters
40:32	L1_DIV_LOOPC	RW	0x1	L1 PLL input parameters
41				
47:42	L1_DIV_OUT	RW	0x1	L1 PLL input parameters
other	-	RW		Keep

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

The result of clk_ref / div_refc of PLL should be 25-50MHz, and the VCO frequency (inside the brackets in the above formula Points) must be in the range 1.2GHz-3.2GHz. This requirement also applies to memory PLLs.

The following registers are the configuration registers of MEM CLOCK. The MEM CLOCK clock frequency should be configured to the final DDR total 1/2 of the line clock frequency. The base address is 0x1fe00000 and the offset address is 0x1c0:

Table 4-11 Memory clock software frequency multiplication setting register

twenty one

Bit field	Field name	access	Reset value	description
0	SEL _MEM_PLL	RW	0x0	Clock output selection 1: MEM clock selects PLL output 0: MEM clock selection SYS CLOCK
1	SOFT_SET_MEM_PLL	RW	0x0	Allow software to set MEM PLL
2	BYPASS_MEM_PLL	RW	0x0	Bypass MEM_PLL
5: 3				
6	LOCKED_MEM_PLL	R	0x0	Whether MEM_PLL is locked
7	PD_MEM_PLL	RW	0x0	Turn off MEM PLL
13: 8	MEM_PLL_DIV_REFC	RW	0x1	MEM PLL input parameters When selecting NODE clock (NODE_CLOCK_SEL When 1), it is used as frequency division input
23:14	MEM_PLL_DIV_LOOPC	RW	0x41	MEM PLL input parameters
29:24	MEM_PLL_DIV_OUT	RW	0x0	MEM PLL input parameters
30	NODE_CLOCK_SEL	RW	0x0	0: Use MEM_PLL as the MEM clock 1: Use NODE_CLOCK as the crossover input
other		RW		Keep

4.11 Processor core frequency division setting register (0x01D0)

The following register is used for dynamic frequency division of the processor core. Use this register to set the frequency modulation of the processor core In order to complete the frequency conversion operation within 100ns, there is no other overhead. Base address is 0x1fe00000, offset address 0x01d0.

Table 4-12 Processor core software frequency division setting register

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

twenty two

4.12 Processor core reset control register (0x01D8)

The following registers are used to reset the processor core software. When reset is needed, first resetn the corresponding core to 0, Set resetn_pre to 0, wait 500 microseconds, set resetn_pre to 1, then set resetn to 1 to complete the adjustment Reset process. The base address of this register is 0x1fe00000 and the offset address is 0x01d8.

Table 4-13 Processor core software frequency division setting register

Bit field	Field name	access	Reset value	description
0	Core0_resetn_pre	RW	0x1	Core 0 reset auxiliary control
1	Core0_resetn	RW	0x1	Core 0 reset
2	Core1_resetn_pre	RW	0x1	Core 1 reset auxiliary control
3	Core1_resetn	RW	0x1	Core 1 reset
4	Core2_resetn_pre	RW	0x1	Core 2 reset auxiliary control
5	Core2_resetn	RW	0x1	Core 2 reset
6	Core3_resetn_pre	RW	0x1	Core 3 reset auxiliary control
7	Core3_resetn	RW	0x1	Core 3 reset

4.13 Routing setting register (0x0400)

The following registers are used to control some routing settings in the chip. Base address is 0x1fe00000, offset address 0x0400.

Table 4-14 Chip Routing Setting Register

Bit field	Field name	access	Reset value	description
3: 0	scid_sel	RW	0x0	Shared cache hash bit control
6: 4	Node_mask	RW	0x7	Node mask to avoid guessing unused nodes Address is not responding
7		RW	0x0	Keep
8	xrouter_en	RW	0x0	HT1 inter-chip routing enable control
9	disable_0x3ff0	RW	0x0	Prohibit matching through the base address 0x3ff0_0000 Register space routing
12	mcc_en	RW	0x0	MCC mode enable
19:16	ccsd_id	RW	0x0	
twenty four	ccsd_en	RW	0x0	
31:30	mc_en	RW	0x3	Enable routing control of two MCs
37:32	interleave_bit	RW	0x0	Memory hash bit control
39	interleave_en	RW	0x0	Memory hash enable
43:40	ht_control	R		Ht related configuration pins

47:44	ht_reg_disable	RW	0x0	Close ht space, used in consistency mode, Avoid routing HT space addresses to HT

4.14 Other function setting register (0x0420)

The following registers are used to control the on-chip partial function enable. The base address is 0x1fe00000 and the offset address is 0x0420.

Table 4-15 Other function setting registers

Bit field	Field name	access	Reset value	description
0	disable_jtag	RW	0x0	Completely disable the JTAG interface
1	disable_ejtag	RW	0x0	Disable EJTAG interface completely
2	disable_gs132	RW	0x0	Disable GS132 completely

3	disable_ejtag132	RW	0x0	Completely disable the GS132 EJTAG interface
4	Disable_antifuse0	RW	0x0	
5	Disable_antifuse1	RW	0x0	
6	Disable_ID	RW	0x0	
8	resetn_gs132	RW	0x0	GS132 reset control
9	sleeping_gs132	R	0x0	GS132 goes to sleep
10	soft_int_gs132	RW	0x0	GS132 Intercore Interrupt Register
15:12	core_int_en_gs132	RW	0x0	GS132 corresponds to the IO interrupt enable of each core
18:16	freqscale_gs132	RW	0x0	GS132 crossover control
19	clken_gs132	RW	0x0	GS132 clock enable
twenty	stable_resetn	RW	0x0	Stable clock reset control
twenty one	freqscale_percore	RW	0x0	Enable private FM register for each core
twenty two	clken_percore	RW	0x0	Enable clock enable for each core
27:24	confbus_timeout	RW	0x8	Configure bus timeout setting, actual time To the power of 2
29:28	HT_softresetn	RW	0x3	HT controller software reset control
35:32	freqscale_mode_core	RW	0x0	FM mode selection for each core 0: (n + 1) / 8 1: 1 / (n + 1)
36	freqscale_mode_node	RW	0x0	Node FM mode selection
37	freqscale_mode_gs132	RW	0x0	GS132 FM mode selection
39:38	freqscale_mode_HT	RW	0x0	FM mode selection for each HT
40	freqscale_mode_stable	RW	0x0	Stable clock FM mode selection
46:44	freqscale_stable	RW	0x0	Stable clock FM register
47	clken_stable	RW	0x0	Stable clock clock enable
48	EXT_INT_en	RW	0x0	Extended IO interrupt enable

twenty four

Page 45

57:56	thsensor_sel	RW	0x0	Temperature sensor selection
62:60	Auto_scale	R	0x0	Automatic frequency modulation current value
63	Auto_scale_doing	R	0x0	Auto tuning is in effect flag

4.15 Celsius temperature register (0x0428)

The following registers are used to observe the value of the temperature sensor inside the chip. Base address is 0x1fe00000, offset address 0x0428. This register is only available when CSR [0x0008] [0] is valid.

Table 4- 16 Temperature observation register

Bit field	Field name	access	Reset value	description
7: 0	Centigrade temperature	RO	0x0	Celsius
63: 8		RW	0x0	

4.16 SRAM adjustment register (0x0430)

The following registers are used to adjust the operating frequency of Sram inside the processor core. Base address is 0x1fe00000, offset address 0x0430.

Table 4- 17 Processor core SRAM adjustment registers

Bit field	Field name	access	Reset value	description
31: 0	sram_ctrl	RW	0x0	Sram configuration register in core
63:32		RW	0x0	

4.17 FUSE0 observation register (0x0460)

The following registers are used to observe the Fuse0 value visible in some software. Base address is 0x1fe00000, offset address 0x0460.

Table 4- 18 FUSE Observation Register

Bit field	Field name	access	Reset value	description
127: 0	Fuse_0	RW	0x0	

25

4.18 FUSE1 observation register (0x0470)

The following registers are used to observe the Fuse1 value visible in some software. Base address is 0x1fe00000, offset address 0x0470.

Table 4- 19 FUSE observation register

Bit field	Field name	access	Reset value	description
127: 0	Fuse_1	RW	0x0	

26

Page 47

5 Chip clock frequency division and enable control

Loongson 3A4000 can use a single external reference clock SYS_CLOCK. The generation of each clock can depend on SYS_CLOCK, the following sections introduce these clocks separately.

Loongson 3A4000 has separate points for processor core, on-chip network and shared cache, HT controller and GS132 core Frequency mechanism. Compared with the original frequency division mechanism of 3A3000, the version implemented in 3A4000 adds a new frequency division mode, which can Supports 1 / n frequency division value.

The base address of each chip configuration register in this chapter is 0x1fe00000.

5.1 Introduction to chip module clock

The chip reference clock SYS_CLOCK usually uses a 100MHz crystal input, or a 25MHz crystal input can also be used. Do not The same crystal frequency needs to be selected by CLKSEL [4].

In addition to using SYS CLOCK, the reference clock of the HT PHY can also use the 200MHz differential reference input of each PHY. Use CLKSEL [8] to select. When SYS CLOCK is selected as the reference clock and a 25MHz crystal oscillator input is used, HT The PHY cannot work at 3.2GHz.

The clock used in the Godson 3A4000 chip and its control method are shown in the following table.

clock	Clock source	Frequency doubling switch control	Clock switch control	Enable control	Clock description
Boot Clock	SYS_CLOCK	*1	not support	not support	SPI, UART, I2C controller clock
Main Clock	SYS PLL	PLL configuration	not support	not support	SYS PLL output. Node Clock, Core Clock, HTcore Clock, GS132 Clock clock source Mem Clock, Stable Clock optional Zhong Yuan
Node Clock	Main Clock	*1	stand by	not support	On-chip network, shared cache, node clock, HT controller clock source
Core0 Clock	Main Clock	*1	stand by	stand by	Core0 clock
Core1 Clock	Main Clock	*1	stand by	stand by	Core1 clock
Core2 Clock	Main Clock	*1	stand by	stand by	Core2 clock
Core3 Clock	Main Clock	*1	stand by	stand by	Core3 clock
HTcore0 Clock	Node Clock	*1	stand by	stand by	HT0 controller clock, software needs to ensure After the frequency is lower than 1GHz
HTcore1 Clock	Node Clock	*1	stand by	stand by	HT1 controller clock, software needs to ensure After the frequency is lower than 1GHz
GS132 Clock	Main Clock	*1	stand by	stand by	GS132 clock, the software needs to ensure the frequency division Below 1GHz

27

Page 48

Stable Clock	Main Clock		*1	stand by	stand by	Processor core constant counter clock
Mem Clock	MEM PLL		PLL configuration not support	stand by	stand by	Memory controller clock
	Main Clock		/ 2, / 4, / 8 are not supported	stand by	stand by	Memory controller alternative clock

5.2 Frequency division and enable control of processor core

There are multiple modes of processor core frequency division, one is access mode by address, and the other is processor configuration instruction access mode, the following Introduce separately. Each processor core can be controlled separately.

5.2.1 Access by address

Access mode by address is compatible with the 3A3000 processor, using the processor core software frequency division setting register, using the same Address.

Use this register to set the frequency modulation of the processor core, you can complete the frequency conversion operation within 100ns, there is no other amount External expenses. The base address is 0x1fe00000 and the offset address is 0x01d0.

Table 5- 1 Processor core software frequency division setting register

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

In addition to the frequency division configuration compatible with the 3A3000 processor, the 3A4000 can also divide the The frequency of the clock after the frequency is adjusted from the original "(frequency division control value +1) / 8" to "1 / (frequency division control value +1)" Send this The register is located in "Other Function Setting Register". The base address is 0x1fe00000 and the offset address is 0x0420.

Table 5- 2 Other function setting registers

Bit field	Field name	access	Reset value	description
35:32	freqscale_mode_core	RW	0x0	FM mode selection for each core 0: (n + 1) / 8 1: 1 / (n + 1)

5.2.2 Configuration register instruction access

In addition to the traditional access mode by address, 3A4000 also supports the use of configuration register instructions to configure private frequency division Register access.

It should be noted that the private divider configuration register control and the original processor core software divider setting register control They are mutually exclusive, and only one of them can be used. The method of selection is to enter the corresponding bit 行 控制。 Line control. The base address of this register is 0x1fe00000 and the offset address is 0x0420.

Table 5- 3 Other function setting registers

Bit field	Field name	access	Reset value	description
twenty two	freqscale_percore	RW	0x0	Enable private FM register for each core
twenty one	clken_percore	RW	0x0	Enable clock enable for each core

When freqscale_percore is set to 1, the freqscale bit pair in the private divider configuration register is used Set the frequency division of your own clock (including freqscale_mode); when clken_percore is set to 1, make Use the clken bit in the private divider configuration register to control the clock enable.

The configurator is defined as follows. The offset address is 0x1050.

Table 5- 4 Private Division Register of Processor Core

Bit field	Field name	access	Reset value	description
4	freqscale_mode	RW	0x0	Current processor core frequency division mode selection
3	clken	RW	0x0	Clock enable of current processor core
2: 0	freqscale	RW	0x0	Current processor core frequency setting

5.3 Node clock frequency division and enable control

The node clock is the clock used by the on-chip network and shared cache. There are two different control modes. One is the software setting. The second mode is the hardware automatic crossover setting.

The node clock does not support the complete shutdown function, so there is no corresponding clken control bit.

5.3.1 Software settings

The software setting method is compatible with the 3A3000 processor, use the function to set the node frequency division bit in the register, use the same

Address.

The base address of this register is 0x1fe00000, and the offset address is 0x0180.

Table 5- 5 Function Setting Register

Bit field	Field name	access	Reset value	description
42:40	Node0_freq_ ctrl	RW	3'b111	Node 0 frequency division

Consistent with the frequency division control of the processor core, the node clock can also be set by the register to divide the clock frequency after the frequency division The rate is adjusted from the original "(frequency division control value + 1) / 8" to "1 / (frequency division control value + 1)" This register is located in "Other Function setting register ". The base address is 0x1fe00000 and the offset address is 0x0420.

Table 5- 6 Other function setting registers

Bit field	Field name	access	Reset value	description
36	freqscale_mode_node	RW	0x0	Node FM mode selection

5.3.2 Automatic hardware settings

In addition to the active setting by the software, the node clock also supports automatic frequency setting triggered by the temperature sensor. Automatic crossover setting is set by the software for different temperatures in advance, when the temperature of the temperature sensor reaches the corresponding preset Value, it will trigger the corresponding automatic crossover 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: enable control. The setting of this group of registers is valid after being set to 1;

SEL: Input temperature selection. Currently, there are four temperature sensors integrated in the 3A4000. This register is used for configuration selection. The temperature of which sensor is used as input.

FREQ: frequency division number. When the frequency division operation is triggered, this frequency division number is also affected by freqscale_mode_node Sound, when it is 0, adjust the frequency to (FREQ + 1) / 8 times the current clock frequency; when it is 1, adjust the frequency to 1 / (FREQ + 1) times the previous clock frequency.

The base address is 0x1fe00000 or 0x3ff00000.

Table 5- 7 High-temperature frequency-reduction control register description

register	address	control	Explanation
			Four sets of setting priority from high to low [7: 0]: Scale_gate0: High temperature threshold 0, frequency will be reduced if this temperature is exceeded [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	0x1480	RW	[62:60]: Scale_freq3: Frequency division value when frequency is reduced
			High bit of temperature sensor control register [7: 0] Scale_Hi_gate0 high 8 bits [15: 8] Scale_Hi_gate1 high 8 bits [23:16] Scale_Hi_gate2 high 8 bits [31:24] Scale_Hi_gate3 high 8 bits [39:32] Scale_Lo_gate0 high 8 bits [47:40] Scale_Lo_gate1 high 8 bits [55:48] Scale_Lo_gate2 high 8 bits
Thsens_freq_scale_up	0x1490	RW	[63:56] Scale_Lo_gate3 high 8 bits

5.4 HT controller frequency division and enable control

The frequency division mechanism of the HT controller is similar to others. Two HT controllers can be controlled separately. Use function to set up hosting Set the corresponding bit in the device. The base address is 0x1fe00000 and the offset address is 0x0180.

Table 5- 8 Function Setting Register

Bit field	Field name	access	Reset value	description
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

Consistent with other frequency division control, the HT controller clock can also be set by the register to divide the clock frequency after frequency division The rate is adjusted from the original "(frequency division control value + 1) / 8" to "1 / (frequency division control value + 1)" This register is located in "Other Function setting register ". The base address is 0x1fe00000 and the offset address is 0x0420.

It should be noted that because the HT core clock is derived from the Node clock, it is also divided by the Node clock Impact.

Table 5- 9 Other function setting registers

Bit field	Field name	access	Reset value	description
39:38	freqscale_mode_HT	RW	0x0	FM controller FM mode selection

5.5 Stable Counter Frequency Division and Enable Control

The frequency division mechanism of Stable Counter is similar to others. Use other functions to set the corresponding bits in the register Settings. The base address is 0x1fe00000 and the offset address is 0x0420.

Table 5-10 Other function setting registers

Bit field	Field name	access	Reset value	description
twenty one	stable_reset	RW	0x0	Stable clock reset control 1: Set to reset state 0: release software reset
40	freqscale_mode_stable	RW	0x0	Stable clock FM mode selection
46:44	freqscale_stable	RW	0x0	Stable clock FM register
47	clken_stable	RW	0x0	Stable clock clock enable

It should be noted that after stable_reset is set to 0, the software reset is only released. At this time, if When GPIO_FUNC_en [13] is 1, the reset of the stable counter is also controlled by GPIO [13] (active low).

The base address of the GPIO output enable register is 0x1fe00000, and the offset address is 0x0500.

Table 5- 11 GPIO output enable register

Bit field	Field name	access	Reset value	description
31: 0	GPIO_OEn	RW	32'hffffffff	GPIO output enable (active low)
63:32	GPIO_FUNC_En	RW	32'hffff0000	GPIO function enable (active low)

6 Software clock system

The Loongson 3A4000 processor defines a number of different levels of use for the clock used by the system software. processor
The core has traditional counter / compare registers, a new stable counter register, and chip-level
node counter register.

The following describes the stable counter and node counter.

6.1 Stable Counter

A new constant clock source is introduced in Loongson 3A4000, which is called stable counter. Stable counter
The clock is different from the clock of the processor core itself, and it is also different from the node clock. It is an independent master clock.

Both the processor core clock and the node clock are derived from the main clock, but both can freely control the frequency division number (see the introduction in the previc
Shao), and the clock of the stable counter is also derived from the main clock, and can also be divided independently, not with other clocks
Frequency changes.

According to this clock source, a timer and a timer are implemented. Please refer to the timer and how to use the timer
Examine the relevant content of Chapter 13 (Timekeeping Equipment) of the Loongson 3A4000 Command System Manual. This chapter mainly introduces Stable couter
Related registers.

6.1.1 Stable Timer configuration address

Use the Stable counter clock source to implement a monotonically increasing timer counter and a slave setting
Timer whose value decreases downwards; each processor core has its own independent Stable counter and Stable timer.
When the processor accesses the timer, it can only be accessed through special instructions such as rdhwr, DRDTIME, etc.
It can be accessed by load / store by address or by CSR configuration register instruction.

Table 6-1 Address access method

name	Offset address	Authority	description
Core0_timer_config	0x1060	RW	Processor core 0 timer configuration register
Core0_timer_ticks	0x1070	R	Remaining timer value of processor core 0
Core1_timer_config	0x1160	RW	Timer configuration register for processor core 1
Core1_timer_ticks	0x1170	R	Remaining timer value of processor core 1
Core2_timer_config	0x1260	RW	Processor core 2 timer configuration register
Core2_timer_ticks	0x1270	R	Remaining timer value of processor core 2
Core3_timer_config	0x1360	RW	Processor core 3 timer configuration register
Core3_timer_ticks	0x1370	R	Remaining timer value of processor core 3

33

Table 6-2 Configuration register instruction access method

name	Offset address	Authority	description
percore_timer_config	0x1060	RW	Timer configuration register of the current processor core
percore_timer_ticks	0x1070	R	Current timer value of the processor core

Table 6-3 Register meaning

Bit field	Field name	Access	reset value	description
63	timer_config	RW	0x1	Reset to 1, should be written as 1

62	Periodic	RW	0x0	Cycle counting is enabled. When this bit is 1, after the timer is reduced to 0, it is automatically reset to The value of the InitVal field in timer_config.
61	Enable	RW	0x0	Always enable. When this bit is 1, the timer will take effect.
47: 0	InitVal	RW	0x0	Initial value for countdown
timer_ticks				
63:48	0	R	0x0	0 value
47: 0	Ticks	R	0x0	Countdown remaining value. When in acyclic counting, the value will be Stay at 48'hffff_ffff_ffff.

6.1.2 Clock Control of Stable Counter

Stable counter uses the main clock, and can be divided by software frequency division control. .

The following is the clock control register of Stable counter. This register is located in other function setting register of the control chip Device. The base address is 0x1fe00000 and the offset address is 0x0420.

Table 6-4 Other function setting registers

Bit field	Field name	access	Reset value	description
twenty one	stable_reset	RW	0x0	Stable clock reset control 1: Set to reset state 0: release software reset
40	freqscale_mode_stable	RW	0x0	Stable clock FM mode selection
46:44	freqscale_stable	RW	0x0	Stable clock FM register
47	clken_stable	RW	0x0	Stable clock clock enable

After the BIOS configures the Stable counter clock source, the MCSR part in each processor core needs to be updated Used to control the values of CPUCFG.0x4 and CPUCFG.0x5. According to the description in Section 8.1, CPUCFG.0x4 should be filled with The frequency of the crystal oscillator in Hz; CPUCFG.0x5 [31:16] should be filled with frequency division factor; CPUCFG.0x5 [15: 0] should be filled Multiplication factor. Filling in the latter two requires BIOS help to calculate, so that the result of CCFreq * CFM / CFD is equal to The actual frequency of the Stable Counter.

6.1.3 Stable Counter calibration

In the case of a single chip, the counter gap of each core is within 2 cycles, and no special calibration is required. In multi-chip love Under the circumstances, there will be large differences between different chips. A special software and hardware calibration mechanism is required to The counter difference is kept below 100ns.

First of all, in order to ensure that the main clock of each chip does not produce deviations during use, use the same crystal drive driver There is chip SYS_CLK.

Secondly, in order to ensure that the Stable counter of each chip starts timing at the same time, two hardware The multiplexing function of one GPIO pin. Node 0 uses GPIO12 to output the reset signal, all other nodes (including node 0) Use GPIO13 to input the reset signal (need to be configured as Stable counter function). Need to use buffer on the motherboard The device ensures the reset timing (mainly the signal slope). The better the reset timing, the smaller the clock difference between different chips.

Before using the Stable counter, the software must reset the global Stable counter through GPIO12 Before resetting, ensure that the clock selection of each chip is the same, and the reset of each chip has been released. This job is usually It is done by the BIOS. The connection scheme of the system is shown in the figure below.

Figure 6-1 Stable reset control when multiple chips are interconnected

6.2 Node Counter

The address of Node counter in Loongson 3A4000 is the same as that of 3A3000 and before, but avoids the original need To correct the problem by software, you can also use the configuration register instruction to access. It is also important to note that with

3A3000 and the previous chip are the same, the counting frequency of Node counter is exactly the same as Node clock, if you want to make Using Node counter as the clock calculation basis, it is necessary to avoid frequency conversion of Node clock.

6.2.1 Access by address

The access by address mode is compatible with the 3A3000 processor and uses the same address for setting.

The base address of the configuration register is 0x1fe00000 or 0x3ff00000, as shown in the following table.

Table 6-5 Node counter register

name	Offset address	Authority	description
Node counter	0x0408	R	64-bit node clock count

6.2.2 Configuration register instruction access

Node counters are slightly different from other configuration registers when accessed using configuration register instructions. Node The use of counter requires that all processor cores access the same counter, rather than their respective on-chip counters (more At the time), each core is required to access the node counter of the same chip. Therefore, even in a multiplex system, each The chip accesses CSR [0x408] through the configuration register instruction, and all access the node counter on NODE 0.

Please refer to the processor core manual for the specific access address and register definition.

6.3 Summary of Clock System

Stable counter added in Loongson 3A4000 is more stable than node counter and CP0 counter It has more advantages and will not change with the division of other clocks (node clock and core clock).

In terms of ease of use, the Stable counter is also more convenient to access, whether it is user mode or using the rdhwr command Guest state can be obtained directly. Stable counter is the first choice of software reference clock system.

Node clock is more of a design that considers traditional compatibility and is a backup solution for a clock system.

36

Page 57

Loongson **3A4000** processor register user manual

7 GPIO control

Godson 3A4000 provides up to 32 GPIOs for system use, most of which are multiplexed with other functions. Send by Memory settings, you can also configure GPIO as an interrupt input function, and you can set its interrupt level.

The base address of each chip configuration register in this chapter is 0x1fe00000.

7.1 Output enable register (0x0500)

The base address is 0x1fe00000 and the offset address is 0x0500.

Table 7- 1 Output enable register

Bit field	Field name	access	Reset value	description
31: 0	GPIO_OEn	RW	32'hffffffff	GPIO output enable (active low)
63:32	GPIO_FUNC_En	RW	32'hffff0000	GPIO function enable (active low)

7.2 Input Output Register (0x0508)

The base address is 0x1fe00000 and the offset address is 0x0508.

Table 7- 2 Input Output Register

Bit field	Field name	access	Reset value	description
31: 0	GPIO_O	RW	32'h0	GPIO output settings
63:32	GPIO_I	RO	32'h0	GPIO input status

7.3 Interrupt control register (0x0510)

The base address is 0x1fe00000 and the offset address is 0x0510.

Table 7- 3 Interrupt Control Register

Bit field	Field name	access	Reset value	description
31: 0	GPIO_INT_Pol	RW	32'h0	GPIO interrupt effective level setting 0-active low 1-Active high
63:32	GPIO_INT_en	RW	32'h0	GPIO interrupt enable control, high effective

37

7.4 GPIO pin function multiplexing table

The GPIO pins in 3A4000 are heavily multiplexed with other functions. The following list shows the pin functions of the chip function pins select.

It should be pointed out that GPIO00 – GPIO15 are GPIO functions when the chip is reset, and the default is the input state. Does not drive IO.

And GPIO16 – GPIO31 are multiplexed control pins of HT, which is HT function at reset, in order to prevent internal logic To drive the corresponding IO, you can pull down the corresponding HT0 / 1_Hi / Lo_Hostmode pin. Although reset by default at this time It is an HT function, but it will not drive IO pins and will not affect external devices. It only needs to use GPIO function in software Before you can set the function to GPIO mode.

Table 7- 4 GPIO function multiplexing table

GPIO register	Pin name	Reuse function	Default function
0	GPIO00	SPI_CSn1	GPIO
1	GPIO01	SPI_CSn2	GPIO
2	GPIO02	UART1_RXD	GPIO
3	GPIO03	UART1_TXD	GPIO
4	GPIO04	UART1_RTS	GPIO
5	GPIO05	UART1_CTS	GPIO
6	GPIO06	UART1_DTR	GPIO
7	GPIO07	UART1_DSR	GPIO
8	GPIO08	UART1_DCD	GPIO
9	GPIO09	UART1_RI	GPIO
10	GPIO10	-	GPIO
11	GPIO11	-	GPIO
12	GPIO12	-	GPIO
13	GPIO13	SCNT_RSTn	GPIO
14	GPIO14	PROCHOTn	GPIO
15	GPIO15	THERMTRIPn	GPIO
16	HT0_LO_POWEROK	GPIO16	HT0_LO_POWEROK
17	HT0_LO_RSTn	GPIO17	HT0_LO_RSTn
18	HT0_LO_LDT_REQn	GPIO18	HT0_LO_LDT_REQn
19	HT0_LO_LDT_STOPn	GPIO19	HT0_LO_LDT_STOPn
38			

20	HT0_HI_POWEROK	GPIO20	HT0_HI_POWEROK
twenty one	HT0_HI_RSTn	GPIO21	HT0_HI_RSTn
twenty two	HT0_HI_LDT_REQn	GPIO22	HT0_HI_LDT_REQn
twenty three	HT0_HI_LDT_STOPn	GPIO23	HT0_HI_LDT_STOPn
twenty four	HT1_LO_POWEROK	GPIO24	HT1_LO_POWEROK
25	HT1_LO_RSTn	GPIO25	HT1_LO_RSTn
26	HT1_LO_LDT_REQn	GPIO26	HT1_LO_LDT_REQn

27	HT1_LO_LDT_STOPn	GPIO27	HT1_LO_LDT_STOPn
28	HT1_HI_POWEROK	GPIO28	HT1_HI_POWEROK
29	HT1_HI_RSTn	GPIO29	HT1_HI_RSTn
30	HT1_HI_LDT_REQn	GPIO30	HT1_HI_LDT_REQn
31	HT1_HI_LDT_STOPn	GPIO31	HT1_HI_LDT_STOPn

7.5 GPIO interrupt control

All GPIO pins in 3A4000 can be used as interrupt inputs.

GPIO00, GPIO08, GPIO16, GPIO24 share interrupt line 0 of the interrupt controller.

GPIO01, GPIO09, GPIO17, GPIO25 share interrupt line 1 of the interrupt controller.

GPIO02, GPIO10, GPIO18, GPIO26 share interrupt line 2 of the interrupt controller.

GPIO03, GPIO11, GPIO19, GPIO27 share interrupt line 3 of the interrupt controller.

GPIO04, GPIO12, GPIO20, GPIO28 share interrupt line 4 of the interrupt controller.

GPIO05, GPIO13, GPIO21, GPIO29 share interrupt line 5 of the interrupt controller.

GPIO06, GPIO14, GPIO22, GPIO30 share interrupt line 6 of the interrupt controller.

GPIO07, GPIO15, GPIO23, GPIO31 share interrupt line 7 of the interrupt controller.

The interrupt enable of each GPIO is controlled by the configuration register GPIO_INT_en, the interrupt level is controlled by GPIO_INT_POL, The registers are as follows:

The base address is 0x1fe00000 and the offset address is 0x0510.

Table 7- 5 Interrupt Control Register

Bit field	Field name	access	Reset value	description
31: 0	GPIO_INT_Pol	RW	32'h0	GPIO interrupt effective level setting 0-active low 1-Active high
63:32	GPIO_INT_en	RW	32'h0	GPIO interrupt enable control, high effective

When each interrupt line on the interrupt controller only enables one of the GPIO, you can use the edge trigger mode An interrupt (falling edge when POL is set to 0 and rising edge when it is 1) triggers an interrupt and records it in the interrupt controller.

40

8 GS464V processor core

GS464V 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 GS464V cores in Loongson 3A4000 and shared Cache modules form one through AXI interconnection network Multi-core structure of distributed shared on-chip last-level cache. The main features of GS464V are as follows:

- MIPS64 compatible, support Godson extended instruction set;
- Four-shot superscalar structure, four fixed points, two vectors, and two memory access components
- Each vector component has a width of 256bit, and each component supports up to 8 double 32-bit floating point multiply-add operations;
- The memory access unit supports 256-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 fully connected items plus 2048 items connected by 8-way group, a total of 2112 TLB, 64 instruction 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;
- The first-level cache implements parity check, and the second-level and on-chip last-level cache implements ECC check;
- Support the standard EJTAG debugging standard, which is convenient for hardware and software debugging;

The structure of GS464V is shown in the figure below.

41

Figure 8- 1 GS464V structure diagram

8.1 Instruction set features implemented by 3A4000

Loongson 3A4000 implements Loongson instruction set functions and features, in addition to the methods defined in the MIPS specification. Recognition can also be dynamically confirmed through the Godson instruction set attribute recognition mechanism.

Loongson 3A4000 recommended software uses Loongson's custom CPUCFG instruction to identify Loongson instruction set attributes (through The way of executing RDCSR to read related CSR can also obtain general information, but RDCSR can only be in the system state carried out).

The CPUCFG instruction is a user mode instruction, and its usage mode is CPUCFG rd, rs, in which the source operand rs register Register number of the configuration information word to be accessed, the returned configuration word information is written to the rd register, each configuration letter The information word contains up to 32 bits of configuration information. For example, the configuration word No. 1 contains the Loongson instruction set involving MIPS compati Related information, where bit 0 indicates whether the hardware floating-point coprocessor is supported, then this configuration information is expressed as CPUCFG.0x1.FP [bit0], where 0x1 indicates that the font size of the configuration information word is 1, and FP indicates this configuration information field

42

The mnemonic name is called FP, and bit0 means that MSA1 is in the 0th bit of the configuration word. If you need more configuration information Bit expression, then its position information will be written in the form of bitAA: BB, indicating from the AA bit to the BB bit Consecutive (AA-BB + 1) bits.

The following table gives a list of configuration information of the instruction set functions implemented by 3A4000. The last column "possible value" means that from this It is possible to read the value in this register, but it does not mean that this value is read from the 3A4000 processor. Specific reading Value, please refer to the actual hardware execution result of the instruction readout, and follow the actual readout value for subsequent software Judgment, try not to directly determine whether a 3A4000 chip supports or does not support a certain function according to the content of the last column of this table can.

Table 8- 1 List of Configuration Information of Instruction Set Functions Realized by 3A4000

Deposit Device number	Bit field	Field name	description	Possible values
0x0	31: 0	PRId	CP0.PRId	32'h14_8001
	0	FP	Equivalent to CP0.Config1.FP [bit0]	1'b1
	3: 1	FPRev	Loongson FPU floating point operation follows the specification version number	3'b2
	4	MMI	1 means the Loongson multimedia instruction extension is realized	1'b1
	5			
	6			
	7			
	8			
	9	LSX1	1 means support Loongson SIMD expansion I	1'b1
	10	LSX2	1 means support Loongson SIMD extension II	1'b1
	11	LASX	1 means support Loongson advanced SIMD extension	1'b1
0x1	12			
	13			
	14			
	15	CNT64	1 means CP0.Count is 64 bits	1'b1
	16	LSLDR0	1 means load to R0 is equivalent to prefetch function	1'b1
	17	LSPREF	1 means PREF instruction has prefetch effect	1'b1
	18	LSPREFX	1 means PREF instruction has prefetch effect	1'b1
	19	LSSYNCI	1 means the SYNCI instruction is implemented as a serialized instruction	1'b1
	20	LSUCA	1 means support partial CACHE in user mode instruction	1'b1
	twenty one	LLSYNC	1 means add SYNC 0 instruction before LL	1'b0

twenty two	TGTSYNC	1 means that the branch between LL and SC needs to Jump target Add SYNC 0 instruction	1'b0
twenty three	LLEXC	1 means it supports the function of LL instruction to initiate an exclusive request can	1'b1
twenty four	SCRAND	1 indicates that the support directory is LL / SC exclusive request increase Random delay	1'b1

			Late function	
	25	MUALP	1 means support unaligned memory access function	1'b1
	26	KMUALEn	1 means that the non-aligned memory access function is enabled in the non-user state Opened	1'b0
	27	ITLBT	1 means ITLB is software transparent	1'b1
	28	LSUPERF	1 means to allow access with (D) MFC0 in user mode ask	1'b1
			Performance Counter	
	29	SFBP	1 means support Store Fill Buffer function	1'b1
	30	CDMAP	1 means support Cache DMA function	1'b1
	0	LEXT1	A value of 1 indicates that Loongson Universal Extension I is implemented	1'b1
	1	LEXT2	A value of 1 indicates that Loongson Universal Extension II is implemented	1'b1
	2	LEXT3	1 means Loongson Universal Extension III	1'b1
	3	LSPW	1 indicates that the Loongson page table traversal instruction extension is implemented	1'b1
	4	LBT1	A value of 1 indicates that the Godson binary translation acceleration extension I is implemented version	1'b1
	5	LBT2	A value of 1 means that the accelerated translation of Godson's binary translation is realized II version	1'b1
	6	LBT3	A value of 1 means that the accelerated translation of Godson's binary translation is realized III version	1'b1
0x2	7	LBTMMU	A value of 1 indicates that Godson's binary translation address translation is implemented Acceleration mechanism	1'b1
	8	LPMP	1 indicates that the Loongson performance counter is implemented, at this time CP0.config1.PC [bit4] must be 1	1'b1
	11: 9	LPMRev	Loongson performance counter implementation version number	3'h2
	13	LPIXU	1 means support to enable Longxin position-independent in user mode Expand	1'b1
	14	LPIXNU	1 means support to enable Godson position in non-user mode Off expansion	1'b1
	15	LVZP	A value of 1 indicates that Loongson's virtualization extension is implemented	1'b1
	18:16	LVZRev	Version Number of Godson Virtualization Specification	3'h2
	19	LGFTP	A value of 1 means that a global constant frequency timing device is implemented	1'b1

	22:20	LGFTPRev	Version number of the global constant frequency timing device	3'h2
	twenty three	LLFTP	1 means local constant frequency timing equipment is implemented	1'b1
	26:24	LLFTPRev	The version number of the local constant frequency timing device	3'h2
	27	LCSRP	1 means the Loongson control status register is supported	1'b1
	28	LDISBLIKELY	1 means to support the function of disabling like branch instruction	1'b1
	0	LCAMP	1 means the hardware lookup table function is implemented	1'b1
	3: 1	LCAMRev	The version number of the hardware lookup table function	3'h2
0x3	11: 4	LCAMNUM	Number of hardware lookup table entries -1	8'h3f
	19:12	LCAMKW	Hardware lookup table Key field bit width -1	8'h2f
	27:20	LCAMVW	Hardware lookup table Data field bit width -1	8'h3f
0x4	31: 0	CCFreq	Processor core crystal frequency in Hz	N / A
0x5	15: 0	CFM	Processor core multiplication factor	N / A
	31:16	CFD	Processor core frequency division factor	N / A
0x6	31: 0	Safe	Godson safety extended parameters	N / A
0x7	0	GCCAEQRP	A value of 1 means that Guest CCA is supported and only Root decides Function of	1'b1
	1	UCAWINP	1 means support non-cache acceleration attribute by address window Configuration function	1'b1

8.2 3A4000 Configuration Status Register Access

3A4000 supports configuration status register space access, CSR uses a new independent addressing space for access, Called CSR space, it does not overlap with the existing register space, memory space and EJTAG dseg space.

CSR uses the customized RDCSR and WRCSR instructions for read and write access. The use of RDCSR is RDCSR rd, rs, where the source operand rs register stores the address of the accessed CSR, the contents of the CSR read back are written Into the rd register. WRCSR is used in WRCSR rd, rs, where the source operand rs register is stored with access Ask the address of the CSR, and store the value of the CSR in the source operand rd register. RDCSR and WRCSR are only allowed in Run in core mode.

The RDCSR / WRCSR instruction can replace the original address mapping configuration register, that is, 0x1fe00000 and 0x3ff00000 space. For the specific access method, please refer to the relevant chapters.

In addition, the core supports a set of CSR registers, which are unique to each processor core. under. The following registers cannot be accessed using 0x3ff00000 and 0x1fe00000 spaces.

Table 8-2 List of Configuration Status Registers in the Core

name	address	description
GFTOffset	0xffffffffffffffff8	Offset of fixed frequency timer in Guest mode
TimerID	0xffffffffffffffff0	ID number of local fixed frequency timer
CSRffe8	0xffffffffffffffe8	Adjust the parameters, see "Godson 3A4000 Command System" for details manual"
ucacc_win0_lo	0xffffffffffffef8	Low bit of non-cache acceleration window 0
ucacc_win1_lo	0xffffffffffffef0	Low bit of non-cache acceleration window 1
ucacc_win2_lo	0xffffffffffffee8	Low bit of non-cache acceleration window 2
ucacc_win3_lo	0xffffffffffffee0	Low bit of non-cache acceleration window 3
ucacc_win0_hi	0xffffffffffffeb8	High bit of non-cache acceleration window 0
ucacc_win1_hi	0xffffffffffffeb0	High bit of non-cache acceleration window 1
ucacc_win2_hi	0xffffffffffffea8	High bit of non-cache acceleration window 2
ucacc_win3_hi	0xffffffffffffea0	High bit of non-cache acceleration window 3
MCSRWG	0xffffffffffff0000	MCSR write control

46

Page 67

9 Shared Cache (SCache)

The SCache module is a three-level cache shared by all processor cores within the Loongson 3A4000 processor. SCache module

The main features include:

- Using 128-bit AXI interface.
- 16 items Cache access queue.
- Keywords first.
- 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. In addition, when the shared cache receives the DMA write request

When the time is required, if the written area is hit and locked in the shared cache, the DMA write will be directly written to the shared cache

Instead of memory.

47

Page 68

Table 9-1 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]	Lock window mask 0
Slock1_valid	0x3ff00208	[63:63]	Lock window 1 valid bit
Slock1_addr	0x3ff00208	[47: 0]	Lock address of No. 1 lock window
Slock1_mask	0x3ff00248	[47: 0]	Lock window mask number 1
Slock2_valid	0x3ff00210	[63:63]	Lock window 2 valid bits
Slock2_addr	0x3ff00210	[47: 0]	Lock address of No. 2 lock window
Slock2_mask	0x3ff00250	[47: 0]	Lock window mask number 2
Slock3_valid	0x3ff00218	[63:63]	Lock window 3 valid bits
Slock3_addr	0x3ff00218	[47: 0]	Lock address of No. 3 lock window
Slock3_mask	0x3ff00258	[47: 0]	Lock window mask number 3

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

The four scache use the same configuration register, the base address is 0x1fe00000, and the offset address is 0x0280.

Table 9- 2 Shared Cache Configuration Register (SC_CONFIG)

Bit field	Field name	access	Reset value	description
0	LRU en	RW	1'b1	Scache LRU replacement algorithm is enabled
16	Prefetch En	RW	1'b1	Scache prefetch function is enabled
22:20	Prefetch config	RW	3'h1	When scache prefetch crosses the configured size Stop prefetching when address range 0 – 4KB 1 – 16KB 2 – 64KB 3 – 1MB 7-Unrestricted (Note: Valid when SCID_SEL == 0)
26:24	Prefetch lookahead	RW	3'h2	scache prefetch step 0-reserved 1 – 0x100 2 – 0x200 3 – 0x300 4 – 0x400 5 – 0x500 6 – 0x600

48

Page 69

30:28	Sc stall dirq cycle	RW	3'h2	7 – 0x700 (Note: Valid when SCID_SEL == 0) SC instruction blocks the number of clock cycles of dirq 0 – 1 cycle (nonstall) 1 – 16-31 cycle random 2 – 32-63 cycle random

				3- 64-127 cycle random 4 – 128-255 cycle random Other-invalid value
31	MCC storefill en	RW	1'b0	MCC storefill function is enabled

10 Inter-processor interrupt and communication

Godson 3A4000 implements 8 inter-core interrupt registers (IPI) for each processor core to support multi-core BIOS boot Interrupt and communication between processor cores when the mobile and operating system are running.

Godson 3A4000 supports two different access methods, one is the address access module compatible with 3A3000 and other processors The other is to support direct private access to the processor register space. The following chapters explain separately.

10.1 Access by address mode

For Loongson 3A4000, the following registers can be accessed using the base address 0x3ff0_0000 or 0x1fe0_0000 ask. Among them, the base address 0x3ff0_0000 can be carried out through the disable_0x3ff0 control bit in the routing setting register shut down. For detailed register descriptions and addresses, see Table 10-1 to Table 10-5.

Table 10- 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 interruption between Loongson 3A4000 and the processor core are described as follows:

Table 10- List of Internuclear Interrupt and Communication Registers of No. 2 Processor Core

name	Offset address	Authority	description
Core0_IPI_Status	0x1000	R	IPI_Status register of processor core 0

Core0_IPI_Enalbe	0x1004	RW	IPI_Enalbe register of processor core 0
Core0_IPI_Set	0x1008	W	IPI_Set register of processor core 0
Core0_IPI_Clear	0x100c	W	IPI_Clear register of processor core 0
Core0_MailBox0	0x1020	RW	IPI_MailBox0 register of processor core 0
Core0_ MailBox1	0x1028	RW	IPI_MailBox1 register of processor core 0
Core0_ MailBox2	0x1030	RW	IPI_MailBox2 register of processor core 0
Core0_ MailBox3	0x1038	RW	IPI_MailBox3 register of processor core 0

Table 10-3 List of Inter-Core Interrupts and Communication Registers for Processor No. 1

name	Offset address	Authority	description
Core1_IPI_Status	0x1100	R	IPI_Status register of processor core 1
Core1_IPI_Enalbe	0x1104	RW	IPI_Enalbe register of processor core 1
Core1_IPI_Set	0x1108	W	IPI_Set register of processor core 1
Core1_IPI_Clear	0x110c	W	IPI_Clear register of processor core 1
Core1_MailBox0	0x1120	R	IPI_MailBox0 register of processor core 1
Core1_ MailBox1	0x1128	RW	IPI_MailBox1 register of processor core 1
Core1_ MailBox2	0x1130	W	IPI_MailBox2 register of processor core 1
Core1_ MailBox3	0x1138	W	IPI_MailBox3 register of processor core 1

Table 10- 4 List of Internuclear Interrupts and Communication Registers of Processor Core 2

name	Offset address	Authority	description
Core2_IPI_Status	0x1200	R	IPI_Status register of processor core 2
Core2_IPI_Enalbe	0x1204	RW	IPI_Enalbe register of processor core 2
Core2_IPI_Set	0x1208	W	IPI_Set register of processor core 2
Core2_IPI_Clear	0x120c	W	IPI_Clear register of processor core 2
Core2_MailBox0	0x1220	R	IPI_MailBox0 register of processor core 2
Core2_ MailBox1	0x1228	RW	IPI_MailBox1 register of processor core 2
Core2_ MailBox2	0x1230	W	IPI_MailBox2 register of processor core 2

Core2_ MailBox3	0x1238	W	IPI_MailBox3 register of processor core 2

Table 10- 5 List of Inter-Core Interrupts and Communication Registers of Processor Core 3

name	Offset address	Authority	description
Core3_IPI_Status	0x1300	R	IPI_Status register of processor core 3
Core3_IPI_Enalbe	0x1304	RW	IPI_Enalbe register of processor core 3
Core3_IPI_Set	0x1308	W	IPI_Set register of processor core 3
Core3_IPI_Clear	0x130c	W	IPI_Clear register of processor core 3

Core3_MailBox0	0x1320	R	IPI_MailBox0 register of processor core 3
Core3_ MailBox1	0x1328	RW	IPI_MailBox1 register of processor core 3
Core3_ MailBox2	0x1330	W	IPI_MailBox2 register of processor core 3
Core3_ MailBox3	0x1338	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 3A4000 chip Memory list. When using multiple Loongson 3A4000 interconnects to form a multi-node CC-NUMA system, the pairs of nodes in each chip It 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 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 processor 0 at the node is 0x10003ff01000, and so on.

10.2 Configuration register instruction access

In Loongson 3A4000, there is a new register access instruction for processor core, which can be configured through private space Register access. In order to use the inter-core interrupt register more conveniently, in this mode Yi made some adjustments.

Table 10-6 List of interrupts and communication registers between current processor cores

name	Offset address	Authority	description
perCore_IPI_Status	0x1000	R	IPI_Status register of the current processor core
perCore_IPI_Enalbe 0x1004		RW	IPI_Enalbe register of the current processor core
perCore_IPI_Set	0x1008	W	IPI_Set register of the current processor core
perCore_IPI_Clear	0x100c	W	IPI_Clear register of the current processor core
perCore_MailBox0	0x1020	RW	IPI_MailBox0 register of the current processor core
perCore_ MailBox1	0x1028	RW	IPI_MailBox1 register of the current processor core
perCore_ MailBox2	0x1030	RW	IPI_MailBox2 register of the current processor core
perCore_ MailBox3	0x1038	RW	IPI_MailBox3 register of the current processor core

In order to send inter-core interrupt requests and MailBox communication to other cores, access is made through the following registers.

Table 10- 7 Communication Register between Processor Cores

name	Offset address	Authority	description
IPI_Send	0x1040	WO	32-bit interrupt distribution register [31] Wait for completion flag, set to 1 to wait for interrupt to take effect [30:26] Reserved [25:16] processor core number [15: 5] reserved [4: 0] Interrupt vector number, corresponding to the vector in IPI_Status

Page 73

Mail_Send	0x1048	WO	64-bit MailBox cache register [63:32] MailBox data [31] Wait for completion flag, when set to 1, it will wait for write to take effect [30:26] Reserved [25:16] processor core number [15: 5] reserved [4: 2] MailBox number 0-MailBox0 lower 32 bits 1-MailBox0 high 32 bits 2-MailBox1 lower 32 bits 3-MailBox1 high 32 bits 4-MailBox2 lower 32 bits 5-MailBox2 high 32 bits 6-MailBox3 lower 32 bits 7-MailBox4 high 32 bits [1: 0] Reserved
FREQ_Send	0x1058	WO	32-bit frequency enable register [31] Wait for completion flag, when set to 1, it will wait for the setting to take effect [30:26] Reserved [25:16] processor core number [15: 5] reserved [4: 0] Write to the corresponding processor core private frequency configuration register. CSR [0x1050]

It should be noted that since the Mail_Send register can only send 32-bit data at a time, when sending 64-bit data Must be split into two transmissions. Therefore, the target core needs to pass other software means while waiting for the content of Mail_Box To ensure the integrity of the transmission. For example, after sending Mail_Box data, an inter-core interrupt indicates that it has been sent to make.

Page 74

11 I / O interrupt

Loongson 3A4000 chip supports two different interrupt methods. The first is the traditional interrupt method, which is handled with 3A3000, etc.

The controller is compatible; the second is the newly added extended IO interrupt mode, which is used to support the interrupt cross-chip and dynamic distribution functions of the H

The two interrupt methods are introduced below.

11.1 Traditional I / O interrupt

The traditional interrupt of Loongson 3A4000 chip supports 32 interrupt sources and is managed in a unified manner, as shown in Figure 7-1 below. Any IO interrupt source can be configured to enable, trigger, and be routed to the target processor core interrupt foot. Traditional interrupts do not support cross-chip distribution of interrupts, and can only interrupt processor cores within the same processor chip.

Figure 11-1 Loongson 3A4000 processor interrupt routing diagram

Interrupt related configuration registers are used to control the corresponding interrupt lines in the form of bits. See the table below for sexual configuration.

The interrupt enable (Enable) configuration has three registers: Intenset, Intenclr and Inten. Intenset Set the interrupt enable, and the interrupt corresponding to the write 1 bit in the Intenset register is enabled. Intenclr clears interrupt enable, The interrupt corresponding to the bit written to 1 in the Intenclr register is cleared. The Inten register reads the current status of each interrupt enable.

The edge-triggered interrupt signal is selected by the Intedge configuration register. Writing 1 means edge triggering, and writing 0 means power. Flat trigger. The interrupt handler can clear the interrupt record through the corresponding bit of Intenclr. Clear interrupt enable.

Table 11- 1 Interrupt Control Register

Bit field	Access properties / default				
	Intedge	Inten	Intenset	Intenclr	Interrupt source
0	RW / 0	R / 0	RW / 0	RW / 0	GPIO24 / 16/8/0 / SC0
1	RW / 0	R / 0	RW / 0	RW / 0	GPIO25 / 17/9/1 / SC1

2	RW / 0	R / 0	RW / 0	RW / 0	GPIO26 / 18/10/2 / SC 2
3	RW / 0	R / 0	RW / 0	RW / 0	GPIO27 / 19/11/3 / SC 3
4	RW / 0	R / 0	RW / 0	RW / 0	GPIO28 / 20/12/4
5	RW / 0	R / 0	RW / 0	RW / 0	GPIO29 / 21/13/5
6	RW / 0	R / 0	RW / 0	RW / 0	GPIO30 / 22/14/6
7	RW / 0	R / 0	RW / 0	RW / 0	GPIO31 / 23/15/7
8	RW / 0	R / 0	RW / 0	RW / 0	I2C0
9	RW / 0	R / 0	RW / 0	RW / 0	I2C1
10	RW / 0	R / 0	RW / 0	RW / 0	UART0
11	RW / 0	R / 0	RW / 0	RW / 0	MC0
12	RW / 0	R / 0	RW / 0	RW / 0	MC1
13	RW / 0	R / 0	RW / 0	RW / 0	SPI
14	RW / 0	R / 0	RW / 0	RW / 0	Thsens
15	RW / 0	R / 0	RW / 0	RW / 0	UART1
23: 16	RW / 0	R / 0	RW / 0	RW / 0	HT0 [7: 0]
31: 24	RW / 0	R / 0	RW / 0	RW / 0	HT1 [7: 0]

Similar to the inter-core interrupt, the base address of the IO interrupt can also be accessed using 0x1fe00000 or 0x3ff00000, It can also be accessed through special register configuration instructions of the processor core.

11.1.1 Access by address

This access method is compatible with the access method of 3A3000 and other processors. The base address can be 0x1fe00000 or 0x3ff00000. The base address of 0x3ff00000 can be entered through the disable_0x3ff0 control bit in the routing configuration register Line is disabled.

Table 11- 2 IO Control Register Address

name	Offset address	description
Intisr	0x1420	32-bit interrupt status register
Inten	0x1424	32-bit interrupt enable status register
Intenset	0x1428	32-bit setting enable register
Intenclr	0x142c	32-bit clear enable register
Intedge	0x1434	32-bit trigger mode register
CORE0_INTISR	0x1440	32-bit interrupt status routed to CORE0
CORE1_INTISR	0x1448	32-bit interrupt status routed to CORE1
CORE2_INTISR	0x1450	32-bit interrupt status routed to CORE2
CORE3_INTISR	0x1458	32-bit interrupt status routed to CORE3

Four cores are integrated in Loongson 3A4000. The above 32-bit interrupt sources can be selected through software configuration. The interrupted target processor core. Further, the interrupt source can be routed to any of the processor core interrupts INT0 to INT3 One is IP2 to IP5 corresponding to CP0_Status. Each of the 32 I / O interrupt sources corresponds to an 8-bit route Controller, its format and address are shown in Table 11-3 and Table 11-4. The routing register is routed in a vector way Select, such as 0x48 to route to INT2 of processor 3.

Table 11- 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 11-4 Interrupt Routing Register Address

name	Offset address	description	name	Offset address	description
Entry0	0x1400	GPIO24 / 16/8/0	Entry16	0x1410	HT0-int0
Entry1	0x1401	GPIO25 / 17/9/1	Entry17	0x1411	HT0-int1
Entry2	0x1402	GPIO26 / 18/10/2	Entry18	0x1412	HT0-int2
Entry3	0x1403	GPIO27 / 19/11/3	Entry19	0x1413	HT0-int3

Entry4	0x1404	GPIO28 / 20/12/4	Entry20	0x1414	HT0-int4
Entry5	0x1405	GPIO29 / 21/13/5	Entry21	0x1415	HT0-int5
Entry6	0x1406	GPIO30 / 22/14/6	Entry22	0x1416	HT0-int6
Entry7	0x1407	GPIO31 / 23/15/7	Entry23	0x1417	HT0-int7
Entry8	0x1408	I2C0	Entry24	0x1418	HT1-int0
Entry9	0x1409	I2C1	Entry25	0x1419	HT1-int1
Entry10	0x140a	UART0	Entry26	0x141a	HT1-int2
Entry11	0x140b	MC0	Entry27	0x141b	HT1-int3
Entry12	0x140c	MC1	Entry28	0x141c	HT1-int4
Entry13	0x140d	SPI	Entry29	0x141d	HT1-int5
Entry14	0x140e	Thsens	Entry30	0x141e	HT1-int6
Entry15	0x140f	UART1	Entry31	0x141f	HT1-int7

11.1.2 Configuration register instruction access

In Godson 3A4000, the configuration register instruction can also be used to access configuration Memory for access. The offset address used by the instruction is the same as that accessed by address. In addition, for the convenience of users In use, a dedicated private interrupt status register is set for each core's different current interrupt status, as shown in the following table.

Table 11-5 Processor core private interrupt status register

name	Offset address	description
perCore_INTISR	0x1010	32-bit interrupt status routed to the current processor core

11.2 Extended I / O interrupt

In addition to being compatible with the original traditional IO interrupt method, 3A4000 began to support extended I / O interrupts, which were used to connect the HT bus The 256-bit interrupt is directly distributed to each processor core, instead of forwarding through the HT interrupt line, and the IO interrupt is improved Use flexibility.

Before the kernel uses the extended IO interrupt, it is necessary to enable the corresponding bit in the "other function setting register". The register The base address is 0x1fe00000 and the offset address is 0x0420.

Table 11-6 Other function setting registers

Page 78

Bit field	Field name	access	Reset value	description
48	EXT_INT_en	RW	0x0	Extended IO interrupt enable

In the extended IO interrupt mode, the HT interrupt can directly perform cross-chip forwarding and rotation distribution operations. Current version This can support up to 256 extended interrupt vectors.

11.2.1 Access by address

The following are the related extended IO interrupt registers. Like other configuration registers, the base address can be used 0x1fe00000 or 0x3ff00000 can also be accessed through the special register configuration instructions of the processor core.

Table 11- 7 Extended IO interrupt enable register

name	Offset address	description
EXT_IOIen [63: 0]	0x1600	Extended IO interrupt [63: 0] interrupt enable configuration
EXT_IOIen [127: 64]	0x1608	Extended IO interrupt [127: 64] interrupt enable configuration
EXT_IOIen [191: 128]	0x1610	Extended IO interrupt [191: 128] interrupt enable configuration
EXT_IOIen [255: 192]	0x1618	Extended IO interrupt [255: 192] interrupt enable configuration

Table 11- 8 Extended IO interrupt auto-rotation enable register

name	Offset address	description
EXT_IOIbounce [63: 0]	0x1680	Automatic rotation enable configuration of extended IO interrupt [63: 0]
EXT_IOIbounce [127: 64]	0x1688	Automatic rotation enable configuration of extended IO interrupt [127: 64]
EXT_IOIbounce [191: 128]	0x1690	Automatic rotation enable configuration of extended IO interrupt [191: 128]
EXT_IOIbounce [255: 192]	0x1698	Automatic rotation enable configuration of extended IO interrupt [255: 192]

Table 11- 9 Extended IO Interrupt Status Register

name	Offset address	description
EXT_IOIsr [63: 0]	0x1700	Interrupt status of extended IO interrupt [63: 0]
EXT_IOIsr [127: 64]	0x1708	Interrupt status of extended IO interrupt [127: 64]
EXT_IOIsr [191: 128]	0x1710	Interrupt status of extended IO interrupt [191: 128]
EXT_IOIsr [255: 192]	0x1718	Interrupt status of extended IO interrupt [255: 192]

Table 11-10 Extended IO interrupt status registers for each processor core

name	Offset address	description

58

Page 79

CORE0_EXT_IOIsr [63: 0]	0x1800	Interrupt status of the extended IO interrupt [63: 0] routed to processor core 0
CORE0_EXT_IOIsr [127: 64]	0x1808	Interrupt status of the extended IO interrupt [127: 64] routed to processor core 0

CORE0_EXT_IOIsr [191: 128]	0x1810	Interrupt status of the extended IO interrupt [191: 128] routed to processor core 0
CORE0_EXT_IOIsr [255: 192]	0x1818	Interrupt status of the extended IO interrupt [255: 192] routed to processor core 0
CORE1_EXT_IOIsr [63: 0]	0x1900	Interrupt status of the extended IO interrupt [63: 0] routed to processor core 1
CORE1_EXT_IOIsr [127: 64]	0x1908	Interrupt status routed to processor core 1's extended IO interrupt [127: 64]
CORE1_EXT_IOIsr [191: 128]	0x1910	Interrupt status routed to processor core 1's extended IO interrupt [191: 128]
CORE1_EXT_IOIsr [255: 192]	0x1918	Interrupt status of the extended IO interrupt [255: 192] routed to processor core 1
CORE2_EXT_IOIsr [63: 0]	0x1A00	Interrupt status of the extended IO interrupt [63: 0] routed to processor core 2
CORE2_EXT_IOIsr [127: 64]	0x1A08	Interrupt status routed to processor core 2's extended IO interrupt [127: 64]
CORE2_EXT_IOIsr [191: 128]	0x1A10	Interrupt status routed to processor core 2's extended IO interrupt [191: 128]
CORE2_EXT_IOIsr [255: 192]	0x1A18	Interrupt status routed to processor core 2's extended IO interrupt [255: 192]
CORE3_EXT_IOIsr [63: 0]	0x1B00	Interrupt status routed to processor core 3's extended IO interrupt [63: 0]
CORE3_EXT_IOIsr [127: 64]	0x1B08	Interrupt status of the extended IO interrupt [127: 64] routed to processor core 3
CORE3_EXT_IOIsr [191: 128]	0x1B10	Interrupt status routed to processor core 3's extended IO interrupt [191: 128]
CORE3_EXT_IOIsr [255: 192]	0x1B18	Interrupt status routed to processor core 3's extended IO interrupt [255: 192]

Similar to the traditional IO interrupt, the 256-bit interrupt source of the extended IO interrupt can also be selected by software configuration for the desired interrupt Target processor core.

However, the interrupt source cannot be individually routed to any one of the processor core interrupts INT0 to INT3, but instead The INT interrupt routing is performed in groups to interrupt the IP2 to IP5 corresponding to CP0_Status. The following is to match by group Set interrupt pin routing register.

Table 11- 11 Interrupt pin routing register description

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

Table 11- 12 Interrupt Routing Register Address

name	Offset address	description
EXT_IOImap0	0x14C0	EXT_IOI [31: 0] pin routing
EXT_IOImap1	0x14C1	EXT_IOI [63:32] pin routing
EXT_IOImap2	0x14C2	EXT_IOI [95:64] pin routing
EXT_IOImap3	0x14C3	EXT_IOI [127: 96] pin routing

EXT_IOImap4	0x14C4	EXT_IOI [159: 128] pin routing
EXT_IOImap5	0x14C5	EXT_IOI [191: 160] pin routing
EXT_IOImap6	0x14C6	EXT_IOI [223: 192] pin routing
EXT_IOImap7	0x14C7	EXT_IOI [255: 224] pin routing

Each interrupt source also corresponds to an 8-bit routing controller. Its format and address are shown in Table 11-13 and Table below. 11-14. Among them, [7: 4] is used to select the true node routing vector in Table 11-5. Routing register vector Route selection, such as 0x48 to mark the route to the No. 3 processor core of the node pointed to by EXT_IOI_node_type4.

Table 11- 13 Interrupt target processor core routing register description

Bit field	Explanation
3: 0	Routed processor core vector number
7: 4	Route node mapping method selection (such as the configuration method in Table 11-15)

It should be noted that when the round-robin distribution mode is used (the corresponding EXT_IOIbounce is 1), the

Rotate on the full mapping mode of the processor core number. The setting of EXT_IOIbounce should be after the related route mapping configuration.

When the fixed distribution mode is used (the corresponding EXT_IOIbounce is 0), only one bitmap node is allowed

Bit is 1, or all 0s, corresponding to local trigger.

Table 11- 14 Interrupt Target Processor Core Routing Register Address

name	Offset address	description
EXT_IOImap_Core0	0x1C00	EXT_IOI [0] processor core routing
EXT_IOImap_Core1	0x1C01	EXT_IOI [1] processor core routing
EXT_IOImap_Core2	0x1C02	EXT_IOI [2] processor core routing
...		
EXT_IOImap_Core254	0x1CFE	EXT_IOI [254] processor core routing
EXT_IOImap_Core255	0x1CFF	EXT_IOI [255] processor core routing

Table 11- 15 Interrupt target node mapping mode configuration

name	Offset address	description
EXT_IOI_node_type0	0x14A0	16 nodes mapping vector 0 (software configuration)
EXT_IOI_node_type1	0x14A2	16 nodes mapping vector 1 (software configuration)
EXT_IOI_node_type2	0x14A4	16 nodes mapping vector 2 (software configuration)
...		
EXT_IOI_node_type15	0x14BE	16 nodes mapping vector 15 (software configuration)

11.2.2 Configuration register instruction access

When using the configuration register instructions of the processor core to access, the biggest difference is the interrupt status

Register access becomes private access, each core only needs to send a query request to the same address to get

The interrupted state of the core.

Table 11-16 Current processor core's extended IO interrupt status register

name	Offset address	description
perCore_EXT_IOIsr [63: 0]	0x1800	Interrupt status of the extended IO interrupt [63: 0] routed to the current processor core
perCore_EXT_IOIsr [127: 64]	0x1808	Interrupt status of the extended IO interrupt [127: 64] routed to the current processor core
perCore_EXT_IOIsr [191: 128]	0x1810	Interrupt status of the extended IO interrupt [191: 128] routed to the current processor core
perCore_EXT_IOIsr [255: 192]	0x1818	Interrupt status of the extended IO interrupt [255: 192] routed to the current processor core

11.2.3 Extended IO interrupt trigger register

In order to support the dynamic distribution of extended IO interrupts, an extended IO interrupt trigger register is added to the configuration register

It is used to set the corresponding IO interrupt. You can usually use this register to debug or test interrupts.

The description of this register is as follows:

Table 11- 17 Extended IO interrupt trigger register

name	Offset address	Authority	description
EXT_IOI_send	0x1140	WO	Extended IO interrupt setting register [7: 0] Interrupt vector set as desired

11.2.4 Differences between extended IO interrupt processing and traditional HT interrupt processing

In the traditional HT interrupt processing method, the HT interrupt is processed internally by the HT controller and directly mapped to the HT configuration register. 256 interrupt vectors in the memory, and then 4 or 8 interrupts are generated by grouping 256 interrupt vectors, and then routed to each The same processor core. Because the traditional interrupt line connection is used, the cross-chip interrupt cannot be directly generated, so all HT IO Interrupts can only be handled directly by a single chip. On the other hand, the interruption of hardware distribution on chip is only the final 4 One or eight interrupts are used as a unit and cannot be processed bit by bit, which leads to the problem that hardware interrupt distribution is not useful.

Extended IO interrupt mode, HT interrupt is directly processed by the HT controller to the interrupt controller of the chip, and the interrupt control The controller can directly get the 256-bit interrupt instead of the previous 4 or 8 interrupts. Each of these 256-bit interrupts can be independently

Independent routing, independent distribution, and can achieve cross-chip distribution and rotation.

After using the extended IO interrupt, the software processing is slightly different from using the traditional HT interrupt.

In traditional HT interrupt processing, the core directly goes to the interrupt vector of the HT controller (generally 0x90000efdfb000080) Search on the computer, and then process it bit by bit, no matter how the routing mode is configured, it is directly read to the HT controller All interruptions.

After using the extended IO interrupt, the kernel directly reads the extended IO status register (configuration space 0x1800) Processing in the off state, each core will only read and process its own interrupt state, and there will be no interference between different cores Disturb.

62

Page 83

12 Temperature sensor

12.1 Real-time temperature sampling

Loongson 3A4000 internally integrates two temperature sensors, 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 0x1FE00000, offset address is 0x0198):

Table 12- 1 Temperature sampling register description

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

Through the setting of the control register, it is possible to achieve interruptions above the preset temperature, interruptions below the preset temperature and high temperature
Automatic frequency reduction function.

In addition, you can also use the newly added Celsius temperature register to directly read the current Celsius temperature. This register is the same
You can use 0x1FE00000 or 0x3FF00000 as the base address of the read operation to access, you can also use the configuration register
The instruction performs direct access and the offset address is 0x0428. The register is described as follows:

Table 12- 2 Extended IO interrupt trigger register

name	Offset address	Authority	description
Thsens_Temperature	0x0428	R	Temperature sensor Celsius

12.2 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

63

Page 84

Interruption

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

SEL: Input temperature selection. Currently 3A4000 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.

The specific description of these registers is as follows, and the base address is 0x1fe00000 or 0x3ff00000:

Table 12-3 High and low temperature interrupt register description

register	address	control	Explanation
			[7: 0]: Hi_gate0: high temperature threshold 0, an interrupt will be generated if this temperature is exceeded
			[8: 8]: Hi_en0: High temperature interrupt enable 0
			[11:10]: Hi_Sel0: Select the temperature sensor input source of high temperature interrupt
			[23:16]: Hi_gate1: high temperature threshold 1, exceeding this temperature will generate an interrupt
			[24:24]: Hi_en1: High temperature interrupt enable 1
			[27:26]: Hi_Sel1: Select the temperature sensor input source for high temperature interrupt 1
			[39:32]: Hi_gate2: High temperature threshold 2, above this temperature will generate an interrupt
			[40:40]: Hi_en2: High temperature interrupt enable 2
			[43:42]: Hi_Sel2: Select the temperature sensor input source for high temperature interrupt 2
			[55:48]: Hi_gate3: High temperature threshold 3, exceeding this temperature will generate interrupt
High temperature interrupt control register			[56:56]: Hi_en3: High temperature interrupt enable 3
Thsens_int_ctrl_Hi	0x1460	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	0x1468	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	0x1470	RW	[1]: Low temperature interrupt trigger

64

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

SEL: Input temperature selection. Currently, there are four temperature sensors integrated in the 3A4000. This register is used for configuration selection.

The temperature of which sensor is used as input.

FREQ: frequency division number. When the frequency division operation is triggered, the preset FREQ is used to divide the clock. The frequency division mode is affected l

Control of freqscale_mode_node.

The base address is 0x1fe00000 or 0x3ff00000.

Table 12- 4 Description of the high-temperature down-frequency control register

register	address	control	Explanation
			Four sets of setting priority from high to low
			[7: 0]: Scale_gate0: High temperature threshold 0, frequency will be reduced if this temperature is exceeded
			[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	0x1480	RW	[62:60]: Scale_freq3: Frequency division value when frequency is reduced

			High bit of temperature sensor control register
			[7: 0] Scale_Hi_gate0 high 8 bits
			[15: 8] Scale_Hi_gate1 high 8 bits
			[23:16] Scale_Hi_gate2 high 8 bits
			[31:24] Scale_Hi_gate3 high 8 bits
			[39:32] Scale_Lo_gate0 high 8 bits
			[47:40] Scale_Lo_gate1 high 8 bits
			[55:48] Scale_Lo_gate2 high 8 bits
Thsens_freq_scale_up	0x1490	RW	[63:56] Scale_Lo_gate3 high 8 bits

12.4 Temperature state detection and control

The pins PROCHOTn and THERMTRIPn are used for temperature status detection and control. These two signals are respectively connected to GPIO14 and GPIO15 multiplexing. PROCHOTn can be used as both input and output, THERMTRIPn only has the output function.

When PROCHOTn is used as an input, the chip is controlled by an external temperature detection circuit, and the external temperature detection circuit needs to be reduced You can set PROCHOTn to 0 when the chip temperature is reached. After the chip receives this low level, it will take frequency reduction measures and divide the frequency The value is set by the register prochotn_freq_scale. When PROCHOTn is used as output, the chip can output high temperature interruption, Select one of the four interrupts set in the high-temperature interrupt control register as the external through the prochotn_o_sel register High temperature interruptions issued.

THERMTRIPn as the output, which is controlled by the chip from the high temperature interrupt control register through the thermotripn_o_sel register Select one of the 4 set interrupts as the external high temperature interrupt.

Although THERMTRIPn and PROCHOTn are externally interrupted by high temperature, THERMTRIPn is more urgent PROCHOTn is higher. When PROCHOTn is set, the external temperature control circuit can also take certain measures, such as increasing the wind

Fan speed. When THERMTRIPn is set, the external power supply control circuit should directly take emergency power-off measures.

The specific control registers are as follows:

Table 12-5 Temperature status detection and control register description

register	address	control	Explanation
Temperature status detection and control Register			[0: 0]: prochotn_oe PROCHOTn pin output enable control, 0 is output, 1 is input [5: 4]: prochotn_o_sel PROCHOTn high temperature interrupt output selection [10: 8]: prochotn_freq_scale: frequency division when PROCHOTn input is valid value
Thsens_hi_ctrl	0x1498	RW	[17:16]: thermotripn_o_sel THERMTRIPn high temperature interrupt output selection

12.5 Control of temperature sensors

There are 4 temperature sensors integrated in the 3A4000, and temperature / voltage monitoring can be adjusted through register configuration. It can also directly observe the output content of each temperature sensor for debugging. (Base address is 0x1FE00000, the offset address of the temperature sensor configuration register is 0x01580 + vtsensor_id << 4, temperature sensor The offset address of the data register is 0x01588 + vtsensor_id << 4)

Table 12- 6 Temperature sensor configuration register description

Bit field	Field name	access	Reset value	description
0	Thsens_trigger	RW	0	Enable temperature sensor configuration, if set, can By thsens_mode and thsens_cluster Select monitoring mode and monitoring point; 0 is silent
2	Thsens_mode	RW	0	Recognize the temperature monitoring mode, and the monitoring point by temp_cluster configuration. 0: temperature mode; 1: voltage mode
3	Thsens_datarate	RW	0	Monitoring frequency: 0 – 10 ~ 20Hz 1 – 325 ~ 650Hz
6: 4	Thsens_cluster	RW	0	Sensor monitoring point configuration: 0 is local monitoring Point, 1 ~ 7 is the remote monitoring point
8	Temp_valid	RW	0	Enable temperature sensor output, replace Thsns0_out and CSR [0x198] Thsens0_overflow is the temperature Sensor temperature monitoring value.
11: 9	Temp_cluster	RW	0	Temperature sensor output monitoring point selection, Thsens_trigger is invalid when enabled

Table 12- 7 Temperature sensor data register description

Bit field	Field name	access	Reset value	description
3	Out_mode	R	0	Sensor configuration monitoring mode 0: temperature mode; 1: voltage mode
6: 4	Out_cluster	R	0	Sensor configuration monitoring point
7	Overflow	R	0	Sensor monitoring value overflow
29:16	Data	R	0	Monitoring value read by sensor

Calculation method of readout value:

Junction temperature = data * 731 / 0x4000-273 (temperature range -40 degrees ~ 125 degrees)

Voltage = data * 1.226 / 0x1000

The configuration of the monitoring point is as follows

Table 12-8 Description of temperature sensor monitoring points

sensor	Cluster	Monitoring points	Sensor Cluster	Monitoring points
0	0	Reserved	2	0 Reserved
	1	Core0 monitoring point 0		1 Core2 monitoring point 0
	2	Core0 monitoring point 1		2 Core2 monitoring point 1
	3	Core0 monitoring point 2		3 Scache2
	4	Core0 monitoring point 3		4 Mc1-phy monitoring point 0
	5	SCache0		5 Mc0-phy monitoring point 0
	6	HT0		6 Mc0-ctrl
	7	Reserved		7 Reserved
1	0	Reserved	3	0 Reserved
	1	Core1 monitoring point 0		1 Core3 monitoring point 2
	2	Core1 monitoring point 1		2 Core3 monitoring point 3
	3	Core1 monitoring point 2		3 Scache3
	4	SCache1		4 Mc0-phy monitoring point 1
	5	L1X		5 Mc1-phy monitoring point 1
	6	HT1		6 Mc1-ctrl
	7	NOC-VERT		7 L2X

13 DDR3 / 4 SDRAM controller configuration

The design of the integrated memory controller inside Loongson 3A4000 processor complies with the DDR3 / 4 SDRAM industry standard (JESD79-3 and JESD79-4). In the Godson 3A4000 processor, all memory read / write operations implemented are observed The provisions of JESD79-3 and JESD79-4.

13.1 Overview of DDR3 / 4 SDRAM controller functions

Loongson 3A4000 processor supports DDP and 3DS packaging modes. DDP supports a maximum of 8 CS (by 8 DDR3 / DDR4 SDRAM chip select signal, that is, 4 double-sided memory modules Chip select signal implementation, that is, 32 logical RANK). A total of 22-bit address bus (ie: 18-bit row and column address bus, 2-bit logical bank bus and 2-bit logical bank group bus, in which the row and column address bus and RASn, CASn and Wen Reuse).

Loongson 3A4000 processor can adjust DDR3 / 4 controller parameters when choosing different memory chip types Set up for support. Among them, the maximum supported chip select (CS_n) is 8, the number of logical RANK (CHIP ID) is 8, and the line The number of addresses (ROW) is 18, the number of column addresses (COL) is 12, the number of logical body selection (BANK) is 2 (DDR4) or 3 (DDR3) The number of BANK groups is 2 (DDR4 only). Among them, the pins of DDR3 and DDR4 have multiplexing relationship, see The following table. Further multiplexing relationship CS_n can be equipped with the Chip ID, refer to the specific 13 is *A* bar.

Table 13- 1 DDR3 / 4 Address Control Signal Multiplexing

PAD name	DDR3	DDR4
DDR_ACTn	DDR_A15	DDR_ACTn
DDR_RASn	DDR_RASn	DDR_RASn / DDR_A16
DDR_CASn	DDR_CASn	DDR_CASn / DDR_A15
DDR_WEn	DDR_WEn	DDR_WEn / DDR_A14
DDR_BG [1]	DDR_A14	DDR_BG1
DDR_BG [0]	DDR_BA [2]	DDR_BG0

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 3A4000 processor only accepts memory reads from the processor or external devices. Write request, in all memory read / write operations, the memory control circuit is in slave state.

The memory controller in Loongson 3A4000 processor has the following characteristics:

- Full pipeline operation of commands and read and write data on the interface;
- Memory commands are merged and sorted to improve overall bandwidth;

- The configuration register read-write port can modify the basic parameters of the memory device;
- Built-in dynamic delay compensation circuit (DCC) for reliable data transmission and reception
- ECC function can detect 1-bit and 2-bit errors on the data path, and can detect 1-bit errors

Carry out automatic error correction;

- Support DDR3 / 4 SDRAM, and parameter configuration support x4, x8, x16 particles;
- Controller and PHY frequency ratio 1/2;
- Support data transmission rate range of 800Mbps-3200Mbps.

13.2 DDR3 / 4 SDRAM read operation protocol

The protocol of DDR3 SDRAM read operation is shown in Figure 13-1. In the figure, the command (Command, abbreviated as CMD) is composed of CAS_n and WE_n are composed of 3 signals. For read operations, RAS_n = 1, CAS_n = 0, WE_n = 1.

Figure 13- 1 DDR3 SDRAM read operation protocol

In the figure above, Cas Latency (CL) = 5, Read Latency (RL) = 5, Burst Length = 8.

The DDR4 SDRAM read operation protocol is similar. In the figure, the command CMD is composed of 4 ACT_n, RAS_n, CAS_n and WE_n Signal composition. For read operations, ACT_n = 1, RAS_n = 1, CAS_n = 0, WE_n = 1.

13.3 DDR3 / 4 SDRAM write operation protocol

The protocol of DDR3 SDRAM write operation is shown in Figure 13-2. In the figure, the command CMD consists of RAS_n, CAS_n and WE_n It consists of 3 signals. For write operations, RAS_n = 1, CAS_n = 0, WE_n = 0. In addition, unlike read operations, write operations DQM can be used to identify the data mask of the write operation, that is, the number of bytes to be written. DQM is synchronized with the DQS signal in the figure.

Figure 13- 2 DDR3 SDRAM write operation protocol

In the figure above, Cas Latency (CL) = 5, Wead Latency (WL) = 5, Burst Length = 8.

The DDR4 SDRAM write operation protocol is similar. In the figure, the command CMD is composed of 4 ACT_n, RAS_n, CAS_n and WE_n Signal composition. For read operations, ACT_n = 1, RAS_n = 1, CAS_n = 0, WE_n = 0.

13.4 DDR3 / 4 SDRAM parameter configuration format

13.4.1 Parameter list of the memory controller

Table 13- 2 List of visible parameters of the memory controller software

Offset	63:55	55:48	47:40	39:32	31:24	23:16	15: 8	7: 0
PHY								
0x0000							version (RD)	

0x0008			x4_mode	ddr3_mode			capability (RD)	
0x0010							dram_init (RD)	init_start
0x0018								
0x0020							preamble2	rdfifo_valid
0x0028		rdfifo_empty (RD)				Overflow (RD)		
0x0030		dll_value (RD)	dll_init_done (RD)	dll_lock_mode	dll_bypass	dll_adjj_cnt	dll_increment	dll_start_point
0x0038				dll_dbl_fix			dll_close_disable	dll_ck
0x0040				dbl_ctrl_ckca				dll_dbl_ckca
0x0048	pll_ctrl_ckca				pll_lock_ckca (RD)	dll_lock_ckca (RD)	clken_ckca	clksel_ckca
0x0050				dbl_ctrl_ds_0				dll_dbl_ds_0
0x0058	pll_ctrl_ds_0				pll_lock_ds_0 (RD)	dll_lock_ds_0 (RD)	clken_ds_0	clksel_ds_0
0x0060				dbl_ctrl_ds_1				dll_dbl_ds_1
0x0068	pll_ctrl_ds_1				pll_lock_ds_1 (RD)	dll_lock_ds_1 (RD)	clken_ds_1	clksel_ds_1
0x0070				dbl_ctrl_ds_2				dll_dbl_ds_2
0x0078	pll_ctrl_ds_2				pll_lock_ds_2 (RD)	dll_lock_ds_2 (RD)	clken_ds_2	clksel_ds_2
0x0080				dbl_ctrl_ds_3				dll_dbl_ds_3

0x0088	pll_ctrl_ds_3				pll_lock_ds_3 (RD)	dll_lock_ds_3 (RD)	clken_ds_3	clksel_ds_3
0x0090				dbl_ctrl_ds_4				dll_dbl_ds_4
0x0098	pll_ctrl_ds_4				pll_lock_ds_4 (RD)	dll_lock_ds_4 (RD)	clken_ds_4	clksel_ds_4
0x00a0				dbl_ctrl_ds_5				dll_dbl_ds_5
0x00a8	pll_ctrl_ds_5				pll_lock_ds_5 (RD)	dll_lock_ds_5 (RD)	clken_ds_5	clksel_ds_5
0x00b0				dbl_ctrl_ds_6				dll_dbl_ds_6
0x00b8	pll_ctrl_ds_6				pll_lock_ds_6 (RD)	dll_lock_ds_6 (RD)	clken_ds_6	clksel_ds_6
0x00c0				dbl_ctrl_ds_7				dll_dbl_ds_7
0x00c8	pll_ctrl_ds_7				pll_lock_ds_7 (RD)	dll_lock_ds_7 (RD)	clken_ds_7	clksel_ds_7
0x00d0				dbl_ctrl_ds_8				dll_dbl_ds_8
0x00d8	pll_ctrl_ds_8				pll_lock_ds_8 (RD)	dll_lock_ds_8 (RD)	clken_ds_8	clksel_ds_8
0x00e0			vrefclk_inv	vref_sample		vref_num	vref_dly	dll_vref
...								
0x0100					dll_1xdly_0	dll_1xgen_0	dll_wrdqs_0	dll_wrdq_0
0x0108						dll_gate_0	dll_rddqs1_0	dll_rddqs0_0
0x0110	rdodt_ctrl_0	rdgate_len_0	rdgate_mode_0	rdgate_ctrl_0			dqs_oe_ctrl_0	dq_oe_ctrl_0
0x0118						dly_2x_0	redge_sel_0	rddqs_phase_0 (RD)
0x0120	w_bdly0_0 [31:28]	w_bdly0_0 [27:24]	w_bdly0_0 [23:20]	w_bdly0_0 [19:16]	w_bdly0_0 [15:12]	w_bdly0_0 [11: 8]	w_bdly0_0 [7: 4]	w_bdly0_0 [3: 0]
0x0128		w_bdly0_0 [59:56]	w_bdly0_0 [55:52]	w_bdly0_0 [51:48]	w_bdly0_0 [47:44]	w_bdly0_0 [43:40]	w_bdly0_0 [39:36]	w_bdly0_0 [35:32]
0x0130	w_bdly1_0 [24:21]	w_bdly1_0 [20:18]	w_bdly1_0 [17:15]	w_bdly1_0 [14:12]	w_bdly1_0 [11: 9]	w_bdly1_0 [8: 6]	w_bdly1_0 [5: 3]	w_bdly1_0 [2: 0]
0x0138								w_bdly1_0 [27:26]
0x0140							rg_bdly_0 [7: 4]	rg_bdly_0 [3: 0]
0x0148								
0x0150	rdqsp_bdly_0 [31:28]	rdqsp_bdly_0 [27: twenty four]	rdqsp_bdly_0 [23: 20]	rdqsp_bdly_0 [19:16]	rdqsp_bdly_0 [15:12]	rdqsp_bdly_0 [11: 8]	rdqsp_bdly_0 [7: 4]	rdqsp_bdly_0 [3: 0]
0x0158								rdqsp_bdly_0 [35:32]
0x0160	rdqsn_bdly_0 [31:28]	rdqsn_bdly_0 [27: twenty four]	rdqsn_bdly_0 [23: 20]	rdqsn_bdly_0 [19:16]	rdqsn_bdly_0 [15:12]	rdqsn_bdly_0 [11: 8]	rdqsn_bdly_0 [7: 4]	rdqsn_bdly_0 [3: 0]
0x0168								rdqsn_bdly_0 [35:32]
0x0170	rdq_bdly_0 [24:21]	rdq_bdly_0 [20:18]	rdq_bdly_0 [17:15]	rdq_bdly_0 [14:12]	rdq_bdly_0 [11: 9]	rdq_bdly_0 [8: 6]	rdq_bdly_0 [5: 3]	rdq_bdly_0 [2: 0]
0x0178								rdq_bdly_0 [27:26]
0x0180					dll_1xdly_1	dll_1xgen_1	dll_wrdqs_1	dll_wrdq_1
0x0188						dll_gate_1	dll_rddqs1_1	dll_rddqs0_1
0x0190	rdodt_ctrl_1	rdgate_len_1	rdgate_mode_1	rdgate_ctrl_1			dqs_oe_ctrl_1	dq_oe_ctrl_1
0x0198						dly_2x_1	redge_sel_1	rddqs_phase_1 (RD)
0x01a0	w_bdly0_1 [31:28]	w_bdly0_1 [27:24]	w_bdly0_1 [23:20]	w_bdly0_1 [19:16]	w_bdly0_1 [15:12]	w_bdly0_1 [11: 8]	w_bdly0_1 [7: 4]	w_bdly0_1 [3: 0]
0x01a8		w_bdly0_1 [59:56]	w_bdly0_1 [55:52]	w_bdly0_1 [51:48]	w_bdly0_1 [47:44]	w_bdly0_1 [43:40]	w_bdly0_1 [39:36]	w_bdly0_1 [35:32]

0x01b0	w_bdly1_1 [24:21]	w_bdly1_1 [20:18]	w_bdly1_1 [17:15]	w_bdly1_1 [14:12]	w_bdly1_1 [11: 9]	w_bdly1_1 [8: 6]	w_bdly1_1 [5: 3]	w_bdly1_1 [2: 0]
0x01b8								w_bdly1_1 [27:26]
0x01c0							rg_bdly_1 [7: 4]	rg_bdly_1 [3: 0]

Page 93

0x01c8								
0x01d0	rdqsp_bdly_1 [31:28]	rdqsp_bdly_1 [27: twenty four]	rdqsp_bdly_1 [23: 20]	rdqsp_bdly_1 [19:16]	rdqsp_bdly_1 [15:12]	rdqsp_bdly_1 [11: 8]	rdqsp_bdly_1 [7: 4]	rdqsp_bdly_1 [3: 0]
0x01d8								rdqsp_bdly_1 [35:32]
0x01e0	rdqsn_bdly_1 [31:28]	rdqsn_bdly_1 [27: twenty four]	rdqsn_bdly_1 [23: 20]	rdqsn_bdly_1 [19:16]	rdqsn_bdly_1 [15:12]	rdqsn_bdly_1 [11: 8]	rdqsn_bdly_1 [7: 4]	rdqsn_bdly_1 [3: 0]
0x01e8								rdqsn_bdly_1 [35:32]
0x01f0	rdq_bdly_1 [24:21]	rdq_bdly_1 [20:18]	rdq_bdly_1 [17:15]	rdq_bdly_1 [14:12]	rdq_bdly_1 [11: 9]	rdq_bdly_1 [8: 6]	rdq_bdly_1 [5: 3]	rdq_bdly_1 [2: 0]
0x01f8								rdq_bdly_1 [27:26]
0x0200					dll_1xdly_2	dll_1xgen_2	dll_wrdqs_2	dll_wrdq_2
0x0208						dll_gate_2	dll_rddqs1_2	dll_rddqs0_2
0x0210	rdodt_ctrl_2	rdgate_len_2	rdgate_mode_2	rdgate_ctrl_2			dqs_oe_ctrl_2	dq_oe_ctrl_2
0x0218						dly_2x_2	redge_sel_2	rddqs_phase_2 (RD)
0x0220	w_bdly0_2 [31:28]	w_bdly0_2 [27:24]	w_bdly0_2 [23:20]	w_bdly0_2 [19:16]	w_bdly0_2 [15:12]	w_bdly0_2 [11: 8]	w_bdly0_2 [7: 4]	w_bdly0_2 [3: 0]
0x0228		w_bdly0_2 [59:56]	w_bdly0_2 [55:52]	w_bdly0_2 [51:48]	w_bdly0_2 [47:44]	w_bdly0_2 [43:40]	w_bdly0_2 [39:36]	w_bdly0_2 [35:32]
0x0230	w_bdly1_2 [24:21]	w_bdly1_2 [20:18]	w_bdly1_2 [17:15]	w_bdly1_2 [14:12]	w_bdly1_2 [11: 9]	w_bdly1_2 [8: 6]	w_bdly1_2 [5: 3]	w_bdly1_2 [2: 0]
0x0238								w_bdly1_2 [27:26]
0x0240							rg_bdly_2 [7: 4]	rg_bdly_2 [3: 0]
0x0248								
0x0250	rdqsp_bdly_2 [31:28]	rdqsp_bdly_2 [27: twenty four]	rdqsp_bdly_2 [23: 20]	rdqsp_bdly_2 [19:16]	rdqsp_bdly_2 [15:12]	rdqsp_bdly_2 [11: 8]	rdqsp_bdly_2 [7: 4]	rdqsp_bdly_2 [3: 0]
0x0258								rdqsp_bdly_2 [35:32]
0x0260	rdqsn_bdly_2 [31:28]	rdqsn_bdly_2 [27: twenty four]	rdqsn_bdly_2 [23: 20]	rdqsn_bdly_2 [19:16]	rdqsn_bdly_2 [15:12]	rdqsn_bdly_2 [11: 8]	rdqsn_bdly_2 [7: 4]	rdqsn_bdly_2 [3: 0]
0x0268								rdqsn_bdly_2 [35:32]
0x0270	rdq_bdly_2 [24:21]	rdq_bdly_2 [20:18]	rdq_bdly_2 [17:15]	rdq_bdly_2 [14:12]	rdq_bdly_2 [11: 9]	rdq_bdly_2 [8: 6]	rdq_bdly_2 [5: 3]	rdq_bdly_2 [2: 0]
0x0278								rdq_bdly_2 [27:26]
0x0280					dll_1xdly_3	dll_1xgen_3	dll_wrdqs_3	dll_wrdq_3
0x0288						dll_gate_3	dll_rddqs1_3	dll_rddqs0_3
0x0290	rdodt_ctrl_3	rdgate_len_3	rdgate_mode_3	rdgate_ctrl_3			dqs_oe_ctrl_3	dq_oe_ctrl_3
0x0298						dly_2x_3	redge_sel_3	rddqs_phase_3 (RD)
0x02a0	w_bdly0_3 [31:28]	w_bdly0_3 [27:24]	w_bdly0_3 [23:20]	w_bdly0_3 [19:16]	w_bdly0_3 [15:12]	w_bdly0_3 [11: 8]	w_bdly0_3 [7: 4]	w_bdly0_3 [3: 0]
0x02a8		w_bdly0_3 [59:56]	w_bdly0_3 [55:52]	w_bdly0_3 [51:48]	w_bdly0_3 [47:44]	w_bdly0_3 [43:40]	w_bdly0_3 [39:36]	w_bdly0_3 [35:32]
0x02b0	w_bdly1_3 [24:21]	w_bdly1_3 [20:18]	w_bdly1_3 [17:15]	w_bdly1_3 [14:12]	w_bdly1_3 [11: 9]	w_bdly1_3 [8: 6]	w_bdly1_3 [5: 3]	w_bdly1_3 [2: 0]
0x02b8								w_bdly1_3 [27:26]
0x02c0							rg_bdly_3 [7: 4]	rg_bdly_3 [3: 0]
0x02c8								
0x02d0	rdqsp_bdly_3 [31:28]	rdqsp_bdly_3 [27: twenty four]	rdqsp_bdly_3 [23: 20]	rdqsp_bdly_3 [19:16]	rdqsp_bdly_3 [15:12]	rdqsp_bdly_3 [11: 8]	rdqsp_bdly_3 [7: 4]	rdqsp_bdly_3 [3: 0]
0x02d8								rdqsp_bdly_3 [35:32]

Page 94

0x02e0	rdqsn_bdly_3 [31:28]	rdqsn_bdly_3 [27: twenty four]	rdqsn_bdly_3 [23: 20]	rdqsn_bdly_3 [19:16]	rdqsn_bdly_3 [15:12]	rdqsn_bdly_3 [11: 8]	rdqsn_bdly_3 [7: 4]	rdqsn_bdly_3 [3: 0]
0x02e8								rdqsn_bdly_3 [35:32]
0x02f0	rdq_bdly_3 [24:21]	rdq_bdly_3 [20:18]	rdq_bdly_3 [17:15]	rdq_bdly_3 [14:12]	rdq_bdly_3 [11: 9]	rdq_bdly_3 [8: 6]	rdq_bdly_3 [5: 3]	rdq_bdly_3 [2: 0]
0x02f8								rdq_bdly_3 [27:26]
0x0300					dll_1xdly_4	dll_1xgen_4	dll_wrdqs_4	dll_wrdq_4
0x0308						dll_gate_4	dll_rddqs1_4	dll_rddqs0_4
0x0310	rdodt_ctrl_4	rdgate_len_4	rdgate_mode_4	rdgate_ctrl_4			dqs_oe_ctrl_4	dq_oe_ctrl_4
0x0318						dly_2x_4	redge_sel_4	rddqs_phase_4 (RD)
0x0320	w_bdly0_4 [31:28]	w_bdly0_4 [27:24]	w_bdly0_4 [23:20]	w_bdly0_4 [19:16]	w_bdly0_4 [15:12]	w_bdly0_4 [11: 8]	w_bdly0_4 [7: 4]	w_bdly0_4 [3: 0]
0x0328		w_bdly0_4 [59:56]	w_bdly0_4 [55:52]	w_bdly0_4 [51:48]	w_bdly0_4 [47:44]	w_bdly0_4 [43:40]	w_bdly0_4 [39:36]	w_bdly0_4 [35:32]
0x0330	w_bdly1_4 [24:21]	w_bdly1_4 [20:18]	w_bdly1_4 [17:15]	w_bdly1_4 [14:12]	w_bdly1_4 [11: 9]	w_bdly1_4 [8: 6]	w_bdly1_4 [5: 3]	w_bdly1_4 [2: 0]
0x0338								w_bdly1_4 [27:26]
0x0340							rg_bdly_4 [7: 4]	rg_bdly_4 [3: 0]
0x0348								
0x0350	rdqsp_bdly_4 [31:28]	rdqsp_bdly_4 [27: twenty four]	rdqsp_bdly_4 [23: 20]	rdqsp_bdly_4 [19:16]	rdqsp_bdly_4 [15:12]	rdqsp_bdly_4 [11: 8]	rdqsp_bdly_4 [7: 4]	rdqsp_bdly_4 [3: 0]
0x0358								rdqsp_bdly_4 [35:32]
0x0360	rdqsn_bdly_4 [31:28]	rdqsn_bdly_4 [27: twenty four]	rdqsn_bdly_4 [23: 20]	rdqsn_bdly_4 [19:16]	rdqsn_bdly_4 [15:12]	rdqsn_bdly_4 [11: 8]	rdqsn_bdly_4 [7: 4]	rdqsn_bdly_4 [3: 0]
0x0368								rdqsn_bdly_4 [35:32]
0x0370	rdq_bdly_4 [24:21]	rdq_bdly_4 [20:18]	rdq_bdly_4 [17:15]	rdq_bdly_4 [14:12]	rdq_bdly_4 [11: 9]	rdq_bdly_4 [8: 6]	rdq_bdly_4 [5: 3]	rdq_bdly_4 [2: 0]
0x0378								rdq_bdly_4 [27:26]
0x0380					dll_1xdly_5	dll_1xgen_5	dll_wrdqs_5	dll_wrdq_5
0x0388						dll_gate_5	dll_rddqs1_5	dll_rddqs0_5
0x0390	rdodt_ctrl_5	rdgate_len_5	rdgate_mode_5	rdgate_ctrl_5			dqs_oe_ctrl_5	dq_oe_ctrl_5
0x0398						dly_2x_5	redge_sel_5	rddqs_phase_5 (RD)
0x03a0	w_bdly0_5 [31:28]	w_bdly0_5 [27:24]	w_bdly0_5 [23:20]	w_bdly0_5 [19:16]	w_bdly0_5 [15:12]	w_bdly0_5 [11: 8]	w_bdly0_5 [7: 4]	w_bdly0_5 [3: 0]
0x03a8		w_bdly0_5 [59:56]	w_bdly0_5 [55:52]	w_bdly0_5 [51:48]	w_bdly0_5 [47:44]	w_bdly0_5 [43:40]	w_bdly0_5 [39:36]	w_bdly0_5 [35:32]
0x03b0	w_bdly1_5 [24:21]	w_bdly1_5 [20:18]	w_bdly1_5 [17:15]	w_bdly1_5 [14:12]	w_bdly1_5 [11: 9]	w_bdly1_5 [8: 6]	w_bdly1_5 [5: 3]	w_bdly1_5 [2: 0]
0x03b8								w_bdly1_5 [27:26]
0x03c0							rg_bdly_5 [7: 4]	rg_bdly_5 [3: 0]
0x03c8								
0x03d0	rdqsp_bdly_5 [31:28]	rdqsp_bdly_5 [27: twenty four]	rdqsp_bdly_5 [23: 20]	rdqsp_bdly_5 [19:16]	rdqsp_bdly_5 [15:12]	rdqsp_bdly_5 [11: 8]	rdqsp_bdly_5 [7: 4]	rdqsp_bdly_5 [3: 0]
0x03d8								rdqsp_bdly_5 [35:32]
0x03e0	rdqsn_bdly_5 [31:28]	rdqsn_bdly_5 [27: twenty four]	rdqsn_bdly_5 [23: 20]	rdqsn_bdly_5 [19:16]	rdqsn_bdly_5 [15:12]	rdqsn_bdly_5 [11: 8]	rdqsn_bdly_5 [7: 4]	rdqsn_bdly_5 [3: 0]
0x03e8								rdqsn_bdly_5 [35:32]
0x03f0	rdq_bdly_5 [24:21]	rdq_bdly_5 [20:18]	rdq_bdly_5 [17:15]	rdq_bdly_5 [14:12]	rdq_bdly_5 [11: 9]	rdq_bdly_5 [8: 6]	rdq_bdly_5 [5: 3]	rdq_bdly_5 [2: 0]

0x03f8								rdq_bdly_5 [27:26]
0x0400					dll_1xdly_6	dll_1xgen_6	dll_wrdqs_6	dll_wrdq_6
0x0408						dll_gate_6	dll_rddqs1_6	dll_rddqs0_6
0x0410	rdodt_ctrl_6	rdgate_len_6	rdgate_mode_6	rdgate_ctrl_6			dqs_oe_ctrl_6	dq_oe_ctrl_6
0x0418						dly_2x_6	redge_sel_6	rddqs_phase_6 (RD)
0x0420	w_bdly0_6 [31:28]	w_bdly0_6 [27:24]	w_bdly0_6 [23:20]	w_bdly0_6 [19:16]	w_bdly0_6 [15:12]	w_bdly0_6 [11: 8]	w_bdly0_6 [7: 4]	w_bdly0_6 [3: 0]
0x0428		w_bdly0_6 [59:56]	w_bdly0_6 [55:52]	w_bdly0_6 [51:48]	w_bdly0_6 [47:44]	w_bdly0_6 [43:40]	w_bdly0_6 [39:36]	w_bdly0_6 [35:32]
0x0430	w_bdly1_6 [24:21]	w_bdly1_6 [20:18]	w_bdly1_6 [17:15]	w_bdly1_6 [14:12]	w_bdly1_6 [11: 9]	w_bdly1_6 [8: 6]	w_bdly1_6 [5: 3]	w_bdly1_6 [2: 0]
0x0438								w_bdly1_6 [27:26]
0x0440							rg_bdly_6 [7: 4]	rg_bdly_6 [3: 0]

0x0448								
0x0450	rdqsp_bdly_6 [31:28]	rdqsp_bdly_6 [27: twenty four]	rdqsp_bdly_6 [23: 20]	rdqsp_bdly_6 [19:16]	rdqsp_bdly_6 [15:12]	rdqsp_bdly_6 [11: 8]	rdqsp_bdly_6 [7: 4]	rdqsp_bdly_6 [3: 0]
0x0458								rdqsp_bdly_6 [35:32]
0x0460	rdqsn_bdly_6 [31:28]	rdqsn_bdly_6 [27: twenty four]	rdqsn_bdly_6 [23: 20]	rdqsn_bdly_6 [19:16]	rdqsn_bdly_6 [15:12]	rdqsn_bdly_6 [11: 8]	rdqsn_bdly_6 [7: 4]	rdqsn_bdly_6 [3: 0]
0x0468								rdqsn_bdly_6 [35:32]
0x0470	rdq_bdly_6 [24:21]	rdq_bdly_6 [20:18]	rdq_bdly_6 [17:15]	rdq_bdly_6 [14:12]	rdq_bdly_6 [11: 9]	rdq_bdly_6 [8: 6]	rdq_bdly_6 [5: 3]	rdq_bdly_6 [2: 0]
0x0478								rdq_bdly_6 [27:26]
0x0480					dll_1xdly_7	dll_1xgen_7	dll_wrdqs_7	dll_wrdq_7
0x0488						dll_gate_7	dll_rddqs1_7	dll_rddqs0_7
0x0490	rdodt_ctrl_7	rdgate_len_7	rdgate_mode_7	rdgate_ctrl_7			dqs_oe_ctrl_7	dq_oe_ctrl_7
0x0498						dly_2x_7	redge_sel_7	rddqs_phase_7 (RD)
0x04a0	w_bdly0_7 [31:28]	w_bdly0_7 [27:24]	w_bdly0_7 [23:20]	w_bdly0_7 [19:16]	w_bdly0_7 [15:12]	w_bdly0_7 [11: 8]	w_bdly0_7 [7: 4]	w_bdly0_7 [3: 0]
0x04a8		w_bdly0_7 [59:56]	w_bdly0_7 [55:52]	w_bdly0_7 [51:48]	w_bdly0_7 [47:44]	w_bdly0_7 [43:40]	w_bdly0_7 [39:36]	w_bdly0_7 [35:32]
0x04b0	w_bdly1_7 [24:21]	w_bdly1_7 [20:18]	w_bdly1_7 [17:15]	w_bdly1_7 [14:12]	w_bdly1_7 [11: 9]	w_bdly1_7 [8: 6]	w_bdly1_7 [5: 3]	w_bdly1_7 [2: 0]
0x04b8								w_bdly1_7 [27:26]
0x04c0							rg_bdly_7 [7: 4]	rg_bdly_7 [3: 0]
0x04c8								
0x04d0	rdqsp_bdly_7 [31:28]	rdqsp_bdly_7 [27: twenty four]	rdqsp_bdly_7 [23: 20]	rdqsp_bdly_7 [19:16]	rdqsp_bdly_7 [15:12]	rdqsp_bdly_7 [11: 8]	rdqsp_bdly_7 [7: 4]	rdqsp_bdly_7 [3: 0]
0x04d8								rdqsp_bdly_7 [35:32]
0x04e0	rdqsn_bdly_7 [31:28]	rdqsn_bdly_7 [27: twenty four]	rdqsn_bdly_7 [23: 20]	rdqsn_bdly_7 [19:16]	rdqsn_bdly_7 [15:12]	rdqsn_bdly_7 [11: 8]	rdqsn_bdly_7 [7: 4]	rdqsn_bdly_7 [3: 0]
0x04e8								rdqsn_bdly_7 [35:32]
0x04f0	rdq_bdly_7 [24:21]	rdq_bdly_7 [20:18]	rdq_bdly_7 [17:15]	rdq_bdly_7 [14:12]	rdq_bdly_7 [11: 9]	rdq_bdly_7 [8: 6]	rdq_bdly_7 [5: 3]	rdq_bdly_7 [2: 0]
0x04f8								rdq_bdly_7 [27:26]
0x0500					dll_1xdly_8	dll_1xgen_8	dll_wrdqs_8	dll_wrdq_8
0x0508						dll_gate_8	dll_rddqs1_8	dll_rddqs0_8
0x0510	rdodt_ctrl_8	rdgate_len_8	rdgate_mode_8	rdgate_ctrl_8			dqs_oe_ctrl_8	dq_oe_ctrl_8

Page 96

0x0518						dly_2x_8	redge_sel_8	rddqs_phase_8 (RD)
0x0520	w_bdly0_8 [31:28]	w_bdly0_8 [27:24]	w_bdly0_8 [23:20]	w_bdly0_8 [19:16]	w_bdly0_8 [15:12]	w_bdly0_8 [11: 8]	w_bdly0_8 [7: 4]	w_bdly0_8 [3: 0]
0x0528		w_bdly0_8 [59:56]	w_bdly0_8 [55:52]	w_bdly0_8 [51:48]	w_bdly0_8 [47:44]	w_bdly0_8 [43:40]	w_bdly0_8 [39:36]	w_bdly0_8 [35:32]
0x0530	w_bdly1_8 [24:21]	w_bdly1_8 [20:18]	w_bdly1_8 [17:15]	w_bdly1_8 [14:12]	w_bdly1_8 [11: 9]	w_bdly1_8 [8: 6]	w_bdly1_8 [5: 3]	w_bdly1_8 [2: 0]
0x0538								w_bdly1_8 [27:26]
0x0540							rg_bdly_8 [7: 4]	rg_bdly_8 [3: 0]
0x0548								
0x0550	rdqsp_bdly_8 [31:28]	rdqsp_bdly_8 [27: twenty four]	rdqsp_bdly_8 [23: 20]	rdqsp_bdly_8 [19:16]	rdqsp_bdly_8 [15:12]	rdqsp_bdly_8 [11: 8]	rdqsp_bdly_8 [7: 4]	rdqsp_bdly_8 [3: 0]
0x0558								rdqsp_bdly_8 [35:32]
0x0560	rdqsn_bdly_8 [31:28]	rdqsn_bdly_8 [27: twenty four]	rdqsn_bdly_8 [23: 20]	rdqsn_bdly_8 [19:16]	rdqsn_bdly_8 [15:12]	rdqsn_bdly_8 [11: 8]	rdqsn_bdly_8 [7: 4]	rdqsn_bdly_8 [3: 0]
0x0568								rdqsn_bdly_8 [35:32]
0x0570	rdq_bdly_8 [24:21]	rdq_bdly_8 [20:18]	rdq_bdly_8 [17:15]	rdq_bdly_8 [14:12]	rdq_bdly_8 [11: 9]	rdq_bdly_8 [8: 6]	rdq_bdly_8 [5: 3]	rdq_bdly_8 [2: 0]
0x0578								rdq_bdly_8 [27:26]
...								
0x0700					leveling_cs	tLVL_DELAY	leveling_req (WR)	leveling_mode
0x0708							leveling_done (RD)	leveling_ready (RD)
0x0710	leveling_resp_7	leveling_resp_6	leveling_resp_5	leveling_resp_4	leveling_resp_3	leveling_resp_2	leveling_resp_1	leveling_resp_0
0x0718								leveling_resp_8
0x0720								
...								

0x0800	dfe_ctrl_ds	pad_ctrl_ds				pad_ctrl_ck		
0x0808		pad_reset_po	pad_oplen_ca	pad_opdly_ca		pad_ctrl_ca		
0x0810	vref_ctrl_ds_3		vref_ctrl_ds_2		vref_ctrl_ds_1		vref_ctrl_ds_0	
0x0818	vref_ctrl_ds_7		vref_ctrl_ds_6		vref_ctrl_ds_5		vref_ctrl_ds_4	
0x0820							vref_ctrl_ds_8	
0x0828								
0x0830			pad_comp_o (RD)				pad_comp_i	
0x0838								
CTL								
0x1000		tRP	tWLDQSEN	tMOD	tXPR		tCKE	tRESET
0x1008								tODTL
0x1010	tREFretention				tRFC		tREF	
0x1018	tCKESR	tXSRD	tXS		tRFC_dlr			tREF_IDLE
0x1020					tRDPDEN	tCPDED	tXPDLL	tXP
0x1028					tZQperiod	tZQCL	tZQCS	tZQ_CMD
...								
0x1040	tRCD	tRRD_S_slr	tRRD_L_slr	tRRD_dlr				tRAS_min

0x1048				tRTP	tWR_CRC_DM	tWR	tFAW_slr	tFAW
0x1050	tWTR_S_CRC_DM	tWTR_L_CRC_DM	tWTR_S	tWTR		tCCD_dlr	tCCD_S_slr	tCCD_L_slr
0x1058								
0x1060			tPHY_WRLAT	tWL		tRDDATA	tPHY_RDLAT	tRL
0x1068				tCAL				tPL
0x1070			tW2P_sameba	tW2W_sameba	tW2R_sameba	tR2P_sameba	tR2W_sameba	tR2R_sameba
0x1078			tW2P_samebg	tW2W_samebg	tW2R_samebg	tR2P_samebg	tR2W_samebg	tR2R_samebg
0x1080			tW2P_samec	tW2W_samec	tW2R_samec	tR2P_samec	tR2W_samec	tR2R_samec
0x1088								
0x1090			tW2P_samecs	tW2W_samecs	tW2R_samecs	tR2P_samecs	tR2W_samecs	tR2R_samecs
0x1098				tW2W_diffcs	tW2R_diffcs		tR2W_diffcs	tR2R_diffcs
...								
0x1100			cs_ref	cs_resync	cs_zqcl	cs_zq	cs_mrs	cs_enable
0x1108	cke_map				cs_map			
0x1110				cs2cid				cid_map
0x1118								
0x1120	mrs_done (RD)	mrs_req (WR)	pre_all_done (RD)	pre_all_req (WR)	cmd_cmd	status_cmd (RD)	cmd_req (WR)	command_mode
0x1128	cmd_cke	cmd_a			cmd_ba	cmd_bg	cmd_c	cmd_cs
0x1130								cmd_pda
0x1138						cmd_dq0		
0x1140	mr_3_cs_0		mr_2_cs_0		mr_1_cs_0		mr_0_cs_0	
0x1148	mr_3_cs_1		mr_2_cs_1		mr_1_cs_1		mr_0_cs_1	
0x1150	mr_3_cs_2		mr_2_cs_2		mr_1_cs_2		mr_0_cs_2	
0x1158	mr_3_cs_3		mr_2_cs_3		mr_1_cs_3		mr_0_cs_3	
0x1160	mr_3_cs_4		mr_2_cs_4		mr_1_cs_4		mr_0_cs_4	
0x1168	mr_3_cs_5		mr_2_cs_5		mr_1_cs_5		mr_0_cs_5	
0x1170	mr_3_cs_6		mr_2_cs_6		mr_1_cs_6		mr_0_cs_6	
0x1178	mr_3_cs_7		mr_2_cs_7		mr_1_cs_7		mr_0_cs_7	
0x1180	mr_3_cs_0_ddr4		mr_2_cs_0_ddr4		mr_1_cs_0_ddr4		mr_0_cs_0_ddr4	
0x1188			mr_6_cs_0_ddr4		mr_5_cs_0_ddr4		mr_4_cs_0_ddr4	
0x1190	mr_3_cs_1_ddr4		mr_2_cs_1_ddr4		mr_1_cs_1_ddr4		mr_0_cs_1_ddr4	
0x1198			mr_6_cs_1_ddr4		mr_5_cs_1_ddr4		mr_4_cs_1_ddr4	
0x11a0	mr_3_cs_2_ddr4		mr_2_cs_2_ddr4		mr_1_cs_2_ddr4		mr_0_cs_2_ddr4	
0x11a8			mr_6_cs_2_ddr4		mr_5_cs_2_ddr4		mr_4_cs_2_ddr4	
0x11b0	mr_3_cs_3_ddr4		mr_2_cs_3_ddr4		mr_1_cs_3_ddr4		mr_0_cs_3_ddr4	

0x11b8			mr_6_cs_3_ddr4		mr_5_cs_3_ddr4		mr_4_cs_3_ddr4
0x11c0	mr_3_cs_4_ddr4		mr_2_cs_4_ddr4		mr_1_cs_4_ddr4		mr_0_cs_4_ddr4
0x11c8			mr_6_cs_4_ddr4		mr_5_cs_4_ddr4		mr_4_cs_4_ddr4
0x11d0	mr_3_cs_5_ddr4		mr_2_cs_5_ddr4		mr_1_cs_5_ddr4		mr_0_cs_5_ddr4

0x11d8			mr_6_cs_5_ddr4		mr_5_cs_5_ddr4		mr_4_cs_5_ddr4	
0x11e0	mr_3_cs_6_ddr4		mr_2_cs_6_ddr4		mr_1_cs_6_ddr4		mr_0_cs_6_ddr4	
0x11e8			mr_6_cs_6_ddr4		mr_5_cs_6_ddr4		mr_4_cs_6_ddr4	
0x11f0	mr_3_cs_7_ddr4		mr_2_cs_7_ddr4		mr_1_cs_7_ddr4		mr_0_cs_7_ddr4	
0x11f8			mr_6_cs_7_ddr4		mr_5_cs_7_ddr4		mr_4_cs_7_ddr4	
0x1200			nc16_map	nc	channel_width	ba_xor_row_offset	addr_new	cs_place
0x1208						bg_xor_row_offset		addr_mirror
0x1210	addr_base_1				addr_base_0			
0x1218								
0x1220	addr_mask_1				addr_mask_0			
0x1228								
0x1230			cs_diff	c_diff	bg_diff	ba_diff	row_diff	col_diff
0x1238				CF_confbus_timeout				
0x1240	WRQthreshold	tRDQidle	wr_pkc_num	rwq_rb	retry	no_dead_inorder	placement_en	stb_en / pbuf
0x1248								tRWGNTidle
0x1250							rfifo_age	
0x1258	prior_age3		prior_age2		prior_age1		prior_age0	
0x1260	retry_cnt (RD)					rbuffer_max (RD)	rdfifo_depth	stat_en
0x1268								
...								
0x1280	aw_512_align		rd_before_wr	ecc_enable		int_vector (RD)	int_trigger (RD)	int_enable
0x1288								
0x1290						int_cnt_fatal (RD)	int_cnt_err (RD)	int_cnt
0x1298	ecc_cnt_cs_7 (RD)	ecc_cnt_cs_6 (RD)	ecc_cnt_cs_5 (RD)	ecc_cnt_cs_4 (RD)	ecc_cnt_cs_3 (RD)	ecc_cnt_cs_2 (RD)	ecc_cnt_cs_1 (RD)	ecc_cnt_cs_0 (RD)
0x12a0	ecc_data_dir (RD)	ecc_code_dir (RD)	ecc_code_256 (RD)					ecc_code_64 (RD)
0x12a8	ecc_addr (RD)							
0x12b0	ecc_data [63: 0] (RD)							
0x12b8	ecc_data [127: 64] (RD)							
0x12c0	ecc_data [191: 128] (RD)							
0x12c8	ecc_data [255: 192] (RD)							
...								
0x1300							ref_num	ref_sch_en
0x1308							Status_sref (RD)	srefresh_req
...								
0x1340	hardware_pd_7	hardware_pd_6	hardware_pd_5	hardware_pd_4	hardware_pd_3	hardware_pd_2	hardware_pd_1	hardware_pd_0
0x1348	power_sta_7 (RD)	power_sta_6 (RD)	power_sta_5 (RD)	power_sta_4 (RD)	power_sta_3 (RD)	power_sta_2 (RD)	power_sta_1 (RD)	power_sta_0 (RD)
0x1350	selfref_age		slowpd_age		fastpd_age		active_age	

Page 99

0x1358				power_up				Age_step
0x1360	tCONF_IDLE				tLPMC_IDLE			
...								
0x1380								zq_overlap
0x1388								zq_stat_en
0x1390	zq_cnt_1 (RD)				zq_cnt_0 (RD)			
0x1398	zq_cnt_3 (RD)				zq_cnt_2 (RD)			
0x13a0	zq_cnt_5 (RD)				zq_cnt_4 (RD)			
0x13a8	zq_cnt_6 (RD)				zq_cnt_6 (RD)			
...								
0x13c0					odt_wr_cs_map			
0x13c8							odt_wr_length	odt_wr_delay
0x13d0					odt_rd_cs_map			
0x13d8							odt_rd_length	odt_rd_delay
...								
0x1400				tRESYNC_length	tRESYNC_delay	tRESYNC_shift	tRESYNC_max	tRESYNC_min
...								
0x1440					pre_predict		tm_cmdq_num	burst_length
0x1448								ca_timing
0x1450						wr / rd_dbi_en	ca_par_en	crc_en
0x1458							tCA_PAR	tWR_CRC
0x1460	bit_map_7	bit_map_6	bit_map_5	bit_map_6	bit_map_3	bit_map_2	bit_map_1	bit_map_0
0x1468	bit_map_15	bit_map_14	bit_map_13	bit_map_12	bit_map_11	bit_map_10	bit_map_9	bit_map_8
0x1470							bit_map_17	bit_map_16
0x1478								bitmap_mirror
0x1480				alertn_misc (RD)			alertn_cnt	alertn_clr
0x1488	alertn_addr (RD)							
...								
0x1500	win0_base							
0x1508	win1_base							
0x1510	win2_base							
0x1518	win3_base							
0x1520	win4_base							
0x1528	win5_base							
0x1530	win6_base							
0x1538	win7_base							
...								
0x1580	win0_mask							
0x1588	win1_mask							
0x1590	win2_mask							

Page 100

0x1598	win3_mask
0x15a0	win4_mask
0x15a8	win5_mask
0x15b0	win6_mask
0x15b8	win7_mask
...	
0x1600	win0_mmap
0x1608	win1_mmap
0x1610	win2_mmap

0x1618	win3_mmap				
0x1620	win4_mmap				
0x1628	win5_mmap				
0x1630	win6_mmap				
0x1638	win7_mmap				
...					
0x1700				acc_hp	acc_en
0x1708	acc_fake_b		acc_fake_a		
0x1710					
0x1718					
0x1720	addr_base_acc_1		addr_base_acc_0		
0x1728					
0x1730	addr_mask_acc_1		addr_mask_acc_0		
0x1738					
MON					
0x2000					cmd_monitor
0x2008					
0x2010	cmd_fbck [63: 0] (RD)				
0x2018	cmd_fbck [127: 64] (RD)				
0x2020			rw_switch_cnt (RD)		
...					
0x2100					scheduler_mon
0x2108					
0x2110	sch_cmd_num (RD)				
0x2118	ba_conflict_all (RD)				
0x2120	ba_conflict_last1 (RD)				
0x2128	ba_conflict_last2 (RD)				
0x2130	ba_conflict_last3 (RD)				
0x2138	ba_conflict_last4 (RD)				
0x2140	ba_conflict_last5 (RD)				
0x2148	ba_conflict_last6 (RD)				

0x2150	ba_conflict_last7 (RD)				
0x2158	ba_conflict_last8 (RD)				
0x2160	rd_conflict (RD)				
0x2168	wr_conflict (RD)				
0x2170	rtw_conflict (RD)				
0x2178	wtr_conflict (RD)				
0x2180	rd_conflict_last1 (RD)				
0x2188	wr_conflict_last1 (RD)				
0x2190	rtw_conflict_last1 (RD)				
0x2198	wtr_conflict_last1 (RD)				
0x21a0	wr_rd_turnaround (RD)				
0x21a8	cs_turnaround (RD)				
0x21b0	bg_conflict (RD)				
...					
0x2300				sm_leveling	sm_init
0x2308					
0x2310	sm_rank_03	sm_rank_02		sm_rank_01	sm_rank_00
0x2318	sm_rank_07	sm_rank_06		sm_rank_05	sm_rank_04
0x2320	sm_rank_11	sm_rank_10		sm_rank_09	sm_rank_08
0x2328	sm_rank_15	sm_rank_14		sm_rank_13	sm_rank_12
0x2330	sm_rank_19	sm_rank_18		sm_rank_17	sm_rank_16

0x2338		sm_rank_23		sm_rank_22		sm_rank_21	sm_rank_20
0x2340		sm_rank_27		sm_rank_26		sm_rank_25	sm_rank_24
0x2348		sm_rank_31		sm_rank_30		sm_rank_29	sm_rank_28
...							
TST							
0x3000					lpbk_mode	lpbk_start	lpbk_en
0x3008	lpbk_correct (RD)			lpbk_counter (RD)			lpbk_error (RD)
0x3010	lpbk_data_en [63: 0]						
0x3018							lpbk_data_en [71:64]
0x3020						lpbk_data_mask_en	
0x3028							
0x3030	Lpbk_dat_w0 [63: 0]						
0x3038	Lpbk_dat_w0 [127: 64]						
0x3040	Lpbk_dat_w1 [63: 0]						
0x3048	Lpbk_dat_w1 [127: 64]						
0x3050		lpbk_ecc_mask_w0	lpbk_dat_mask_w0			lpbk_ecc_w0	
0x3058		lpbk_ecc_mask_w1	lpbk_dat_mask_w1			lpbk_ecc_w1	

0x3060							prbs_23
0x3068					prbs_init		
...							
0x3100				fix_data_pattern_index	bus_width	page_size	test_engine_en
0x3108		cs_diff_tst	c_diff_tst	bg_diff_tst	ba_diff_tst	row_diff_tst	col_diff_tst
0x3120	addr_base_tst						
0x3128							
0x3130	user_data_pattern						
0x3138							
0x3140	valid_bits [63: 0]						
0x3148							valid_bits [71:64]
0x3150	ctrl [63: 0]						
0x3158	ctrl [127: 64]						
0x3160	obs [63: 0] (RD)						
0x3168	obs [127: 64] (RD)						
0x3170	obs [191: 128] (RD)						
0x3178	obs [255: 192] (RD)						
0x3180	obs [319: 256] (RD)						
0x3188	obs [383: 320] (RD)						
0x3190	obs [447: 384] (RD)						
0x3198	obs [511: 448] (RD)						
0x31a0	obs [575: 512] (RD)						
0x31a8	obs [639: 576] (RD)						
0x31b0				obs [671: 640] (RD)			
...							
0x3200							
0x3208							
0x3220	tud_i0						
0x3228	tud_i1						
0x3230	tud_o (RD)						
...							
0x3300	tst_300						
0x3308	tst_308						

0x3310 tst_310
0x3318 tst_318
0x3320 tst_320
0x3328 tst_328
0x3330 tst_330
0x3338 tst_338

0x3340 tst_340
0x3348 tst_348
0x3350 tst_350
0x3358 tst_358
0x3360 tst_360
0x3368 tst_368
0x3370 tst_370
0x3378 tst_378

13.5 Software Programming Guide

13.5.1 Initial operation

The initialization operation is started when the software writes 0x2 to the register Init_start (0x010). 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) Set pm_clk_sel_ckca and pm_clk_sel_ds

(2) Set pm_phy_init_start to 1 to start initializing PHY

(3) Wait for the DLL main control module to lock, that is, pm_dll_init_done is 1

(4) Wait for pm_dll_lock_ * or pm_pll_lock_ * of all clock generation modules to become 1

(5) Enable all pm_clken_ *

(6) Set pm_init_start to 1, the memory controller starts to initialize

(7) Wait for the memory controller to initialize, that is, the value of pm_dram_init is the same as pm_cs_enable.

13.5.2 Control of reset pin

In order to control the reset pin more easily in the state of STR, etc., you can register through pad_reset_po (0x808)

The device performs special reset pin (DDR_RESETn) control. There are two main control modes:

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

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

The behavior of the feet is:

- When not powered: the pin status is low;
- At power-on: the pin status is low;

- When the controller starts to initialize, the pin state is high;
- During normal operation, the pin status is high.

The timing is shown below:

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

In actual control, the effective level is opposite to the general control mode. So 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 state is low.

The timing is shown below:

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

During the whole memory working period, keep low level. Therefore, the motherboard needs to pass DDR_RESETn through the inverter and internal

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

- Always low

The timing is shown below:

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.

13.5.3 Leveling

Leveling operation is in DDR3 / 4, which is used to intelligently configure the phase relationship between various signals in read and write operations of the memory controlle Department of 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. In addition, this design also supports the bit-deskew function, which is used to compensate for different bits in a dataslice Time delay difference.

13.5.3.1 Write Leveling

Write Leveling is used to configure the phase relationship between writing DQS and clock. Software programming needs to refer to the following steps.

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

(2) Set Dll_wrdqs_x (x = 0… 8) to 0x20;

(3) Set Dll_wrdq_x (x = 0… 8) to 0x0;

(4) Set Lvl_mode to 2'b01;

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

(6) Set Lvl_req to 1;

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

(8) Sampling the Lvl_resp_x register, if it is 0, the corresponding Dll_wrdq_x [6: 0] and dll_1xdly [6: 0] increases by 1 and repeats 5-7 until Lvl_resp_x is 1, then turns to 9; If it is 1, increase the corresponding Dll_wrdq_x [6: 0] and dll_1xdly [6: 0] by 1 and repeat 5-7 Until Lvl_resp_x is 0, and then continue to the corresponding Dll_wrdq_x [6: 0] and dll_1xdly [6: 0] Increase by 1 and repeat 5-7 until Lvl_resp_x is 1, then turn to 9.

(9) Decrease the value of Dll_wrdq_x and dll_1xdly by 0x40, then the value of Dll_wrdq_x and dll_1xdly will be It should be the correct setting value.

(10) Set pm_dly_2x according to the DIMM type, and the value of pm_dly_2x corresponding to the particle on the right side of the 0x0 boundary increases Add 0x010101.

(11) Set Lvl_mode (0x700) to 2'b00 to exit Write Leveling mode.

13.5.3.2 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 to 2'b10;

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

(6) Set Lvl_req to 1;

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

(8) Sampling Lvl_resp_x [0] register. If the first sampling finds that Lvl_resp_x [0] is 1, it will correspond Dll_gate_x [6: 0] increases by 1 and repeats 6-8 until the sampling result is 0, otherwise 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) Set pm_rdedge_sel according to the value of pm_rddqs_phase

(11) Decrease Dll_gate_x (x = 0 ... 8) by 0x20;

(12) After the adjustment, 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, then proceed to step 13; If it is not 4, you may need to add or subtract one to Rd_oe_begin_x, if it is greater than Burst_length / 2, it may be necessary to fine-tune the value of Dll_gate_x;

(13) Set Lvl_mode (0x700) to 2'b00 to exit Gate Leveling mode;

(14) At this point, the Gate Leveling operation ends.

13.5.4 Power control configuration process

First you need to set pm_pad_ctrl_ca [0] to 1 and wait for the memory initialization to complete before setting

pm_pad_ctrl_ca [0] is 0. This function can only be used when CAL Mode is enabled in DDR4 mode.

13.5.5 Initiating MRS commands separately

In DDR3 mode, the order of MRS commands issued by the memory controller to the memory are:

MR2_CS0, MR2_CS1, MR2_CS2, MR2_CS3, MR2_CS4, MR2_CS5, MR2_CS6, MR2_CS7,

MR3_CS0, MR3_CS1, MR3_CS2, MR3_CS3, MR3_CS4, MR3_CS5, MR3_CS6, MR3_CS7,

MR1_CS0, MR1_CS1, MR1_CS2, MR1_CS3, MR1_CS4, MR1_CS5, MR1_CS6, MR1_CS7,

MR0_CS0, MR1_CS1, MR1_CS2, MR1_CS3, MR0_CS4, MR0_CS5, MR0_CS6, MR0_CS7.

In addition, for DDR4 mode, the order of MRS commands issued by the memory controller to the memory are:

MR3_CS0, MR3_CS1, MR3_CS2, MR3_CS3, MR3_CS4, MR3_CS5, MR3_CS6, MR3_CS7,

MR6_CS0, MR6_CS1, MR6_CS2, MR6_CS3, MR6_CS4, MR6_CS5, MR6_CS6, MR6_CS7

MR5_CS0, MR5_CS1, MR5_CS2, MR5_CS3, MR5_CS4, MR5_CS5, MR5_CS6, MR5_CS7

MR4_CS0, MR1_CS1, MR1_CS2, MR1_CS3, MR4_CS4, MR4_CS5, MR4_CS6, MR4_CS7

MR2_CS0, MR2_CS1, MR2_CS2, MR2_CS3, MR2_CS4, MR2_CS5, MR2_CS6, MR2_CS7,

MR1_CS0, MR1_CS1, MR1_CS2, MR1_CS3, MR1_CS4, MR1_CS5, MR1_CS6, MR1_CS7,

MR0_CS0, MR1_CS1, MR1_CS2, MR1_CS3, MR0_CS4, MR0_CS5, MR0_CS6, MR0_CS7.

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 (0x1101) and Mr * _cs * (0x1140 – 0x11f8) to the correct values;

(2) Set Command_mode (0x0x1120) to 1 to make the controller enter the command sending mode;

(3) Sampling Status_cmd (0x1122), if it is 1, it means that 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 (0x1126) to 1 and send MRS command to DRAM;

(5) Sampling Mrs_done (0x1127), 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 (0x1120) to 0 to make the controller exit the command sending mode.

13.5.6 Any operation control bus

The memory controller can send any command combination to the DRAM through the command sending mode, and the software can set Cmd_cs, Cmd_cmd, Cmd_ba, Cmd_a (0x1128), issued to the DRAM in the command transmission mode.

The specific operations are as follows:

(1) Set the registers Cmd_cs, Cmd_cmd, Cmd_ba, Cmd_a (0x1128) to the correct values;

(2) Set Command_mode (0x1120) to 1 to make the controller enter the command sending mode;

(3) Sampling Status_cmd (0x1122), if it is 1, it means that 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 (0x1121) to 1 to send commands to DRAM;

(5) Set Command_mode (0x1120) to 0 to make the controller exit the command sending mode.

13.5.7 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) Follow the initialization process to wait for the clock to reset and stabilize;

(3) Set the register Lpbk_en to 1;

(4) Set the register Lpbk_start to 1; this time the self-loop test officially begins.

(5) So far, since the loop test has started, the software needs to constantly check whether there is an error as follows:

(6) Sampling register Lpbk_error, if this value is 1, it means that an error occurred Lpbk_ * and other observation registers are used to observe the error data and correct data of the first error;

Page 109

A value of 0 means that no data errors have occurred.

13.5.8 ECC function usage control

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

Ecc_enable includes the following 2 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.

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.

13.5.9 Error status observation

After an error occurs in the memory controller, you can access the corresponding system configuration register to obtain the corresponding error information and Simple debugging operation. The register base address is 0x1fe00000 or 0x3ff00000, the configuration register can also be used The instruction accesses the register and its corresponding bits as follows.

Table 13-3 No. 0 Memory Controller Error Status Observation Register

register	Offset address	control	Explanation
ECC setting of memory controller 0 register	0x0600	RW	No. 0 memory controller ECC setting register [5: 0]: MC0 int_enable, interrupt enable [8]: MC0 int_trigger, interrupt trigger configuration [21:16]: MC0 int_vector (RO), interrupt vector (read only) [33:32]: MC0 ecc_enable, ECC related functions are enabled [40]: MC0 rd_before_wr, enable after read and write function
Mc0_ecc_set	0x0608	RW	Keep
Memory controller 0 ECC count register	0x0610	RW	No. 0 memory controller ECC count register [7: 0]: MC0 int_cnt, configure the threshold for the number of interrupts triggered by ECC check [15: 8]: MC0 int_cnt_err (RO), statistics of the number of errors in one bit of ECC check (Read only) [23:16]: MC0 int_cnt_fatal (RO), ECC check two-digit error count system
Mc0_ecc_cnt			Meter (read only)

Page 110

No. 0 memory controller ECC error count register

[7: 0]: MC0 ecc_cnt_cs_0, CS0 ECC check error count

[15: 8]: MC0 ecc_cnt_cs_1, CS1 ECC check error count

[23:16]: MC0 ecc_cnt_cs_2, CS2 ECC check error count

[31:24]: MC0 ecc_cnt_cs_3, statistics on the number of ECC check errors in CS3

[39:32]: MC0 ecc_cnt_cs_4, statistics on the number of ECC check errors in CS4

No. 0 memory controller ECC error			[47:40]: MC0 ecc_cnt_cs_5, CS5 ECC check error count
Statistics register			[55:48]: MC0 ecc_cnt_cs_6, statistics of the number of ECC check errors in CS6
Mc0_ecc_cs_cnt	0x0618	RO	[63:56]: MC0 ecc_cnt_cs_7, CS7 count of ECC check errors
			No. 0 memory controller ECC check code register
			[7: 0]: MC0 ecc_code_64, ECC check code for 64-bit ECC check,
			Disabled when the memory directory function is enabled
			[41:32]: MC0 ecc_code_256, ECC check during 256-bit ECC check
			Code, valid when the memory directory function is enabled
			[52:48]: MC0 ecc_code_dir, ECC check code of memory directory, only use
No. 0 memory controller ECC check			Effective when memory directory function is available
Code register			[60:56]: MC0 ecc_data_dir, memory directory ECC data, only enabled
Mc0_ecc_code	0x0620	RO	Effective when the memory directory function
No. 0 memory controller ECC error			
Address register			No. 0 memory controller ECC error address register
Mc0_ecc_addr	0x0628	RO	[63: 0]: MC0 ecc_addr, ECC check error address information
No. 0 memory controller ECC error			No. 0 memory controller ECC error data register 0
Data register 0			[63: 0]: Mc0_ecc_data0, data information when ECC check error occurs, 64 bits
Mc0_ecc_data0	0x0630	RO	Data in ECC mode, data in 256-bit ECC mode [63: 0]
No. 0 memory controller ECC error			No. 0 memory controller ECC error data register 1
Data register 1			[63: 0]: Mc0_ecc_data1, data information when ECC check error occurs, 256 bits
Mc0_ecc_data1	0x0638	RO	Data in ECC mode [127: 64]
No. 0 memory controller ECC error			No. 0 memory controller ECC error data register 2
Data register 2			[63: 0]: Mc0_ecc_data2, data information when ECC check error occurs, 256 bits
Mc0_ecc_data2	0x0640	RO	Data in ECC mode [191: 128]
No. 0 memory controller ECC error			No. 0 memory controller ECC error data register 3
Data register 3			[63: 0]: Mc0_ecc_data3, data information when ECC check error occurs, 256 bits
Mc0_ecc_data3	0x0648	RO	Data in ECC mode [255: 192]

Table 13-4 No. 1 memory controller error status observation register

register	address	control	Explanation

			Memory controller 1 ECC setting register
			[5: 0]: MC1 int_enable, interrupt enable
			[8]: MC1 int_trigger, interrupt trigger configuration
Memory controller 1 ECC settings			[21:16]: MC1 int_vector (RO), interrupt vector (read only)
register			[33:32]: MC1 ecc_enable, ECC related function enable
Mc1_ecc_set	0x0700	RW	[40]: MC1 rd_before_wr, enable after read and write function
	0x0708	RW	Keep
			Memory controller 1 ECC count register
			[7: 0]: MC1 int_cnt, configure the threshold for the number of interrupts triggered by ECC check
			[15: 8]: MC1 int_cnt_err (RO), statistics of the number of errors in one bit of ECC check
Memory controller 1 ECC count			(Read only)
register			[23:16]: MC1 int_cnt_fatal (RO), ECC check two-digit error count system
Mc1_ecc_cnt	0x0710	RW	Meter (read only)
			No. 1 memory controller ECC error count register
			[7: 0]: MC1 ecc_cnt_cs_0, CS0 ECC check error count
			[15: 8]: MC1 ecc_cnt_cs_1, statistics on the number of ECC check errors in CS1
			[23:16]: MC1 ecc_cnt_cs_2, CS2 ECC check error count
			[31:24]: MC1 ecc_cnt_cs_3, CS3 ECC check error count
			[39:32]: MC1 ecc_cnt_cs_4, CS4 ECC check error count

No. 1 memory controller ECC error			[47:40]: MC1 ecc_cnt_cs_5, CS5 ECC check error count
Statistics register			[55:48]: MC1 ecc_cnt_cs_6, statistics of the number of ECC check errors in CS6
Mc1_ecc_cs_cnt	0x0718	RO	[63:56]: MC1 ecc_cnt_cs_7, CS7 ECC check error count
No. 1 memory controller ECC check code register			No. 1 memory controller ECC check code register [7: 0]: MC1 ecc_code_64, ECC check code for 64-bit ECC check, Disabled when the memory directory function is enabled [41:32]: MC1 ecc_code_256, ECC check when 256-bit ECC check Code, valid when the memory directory function is enabled [52:48]: MC1 ecc_code_dir, ECC check code of memory directory, only use Effective when memory directory function is available [60:56]: MC1 ecc_data_dir, memory directory ECC data, only enabled Effective when the memory directory function
Mc1_ecc_code	0x0720	RO	
No. 1 memory controller ECC error Address register			No. 1 memory controller ECC error address register
Mc1_ecc_addr	0x0728	RO	[63: 0]: MC1 ecc_addr, ECC check error address information
No. 1 memory controller ECC error Data register 0			No. 1 memory controller ECC error data register 0 [63: 0]: Mc1_ecc_data0, data information when ECC check error occurs, 64 bits
Mc1_ecc_data0	0x0730	RO	Data in ECC mode, data in 256-bit ECC mode [63: 0]
No. 1 memory controller ECC error Data register 1			No. 1 memory controller ECC error data register 1 [63: 0]: Mc1_ecc_data1, data information when ECC check error occurs, 256 bits
Mc1_ecc_data1	0x0738	RO	Data in ECC mode [127: 64]

No. 1 memory controller ECC error Data register 2			No. 1 memory controller ECC error data register 2 [63: 0]: Mc1_ecc_data2, data information when ECC check error occurs, 256 bits
Mc1_ecc_data2	0x0740	RO	Data in ECC mode [191: 128]
No. 1 memory controller ECC error Data register 3			No. 1 memory controller ECC error data register 3 [63: 0]: Mc1_ecc_data3, data information when ECC check error occurs, 256 bits
Mc1_ecc_data3	0x0748	RO	Data in ECC mode [255: 192]

92

14 HyperTransport controller

In Loongson 3A4000, 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 14.5.14 for details): When configured to support Cache consistency mode, IO device access to internal DMA is Cache layer is transparent, that is, the consistency is automatically maintained by the hardware, without the need for software to maintain through the program Cache instruction; When the HyperTransport bus is used for multi-chip interconnection, the HT0 controller (the initial address is 0x0C00_0000_0000 – 0x0DFF_FFFF_FFFF) can support the consistent transmission of Cache between chips through pin configuration, and the HT1 controller (initial address 0x0E00_0000_0000-0x0FFF_FFFF_FFFF) can be configured to support the maintenance of Cache consistency between slices through software configuration, See Section 14.7 for details. In the 8-chip interconnect structure, the consistency mode of the HT1_HI controller is passed through the pins in CHIP_CONFIG To configure.

The HyperTransport controller supports up to two-way 16-bit width and 2.4GHz 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 14.1 for the specific method.

The main features of Loongson 3A4000 HyperTransport controller are as follows:

- Support HT1.0 / HT3.0 protocol
- Support 200/400/800/1600/2000 / 2400MHz operating frequency
- The controller frequency is up to 1GHz
- 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

14.1 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.

93

Page 114

Table 14-1 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, 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

Page 115

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,

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 14.5.2). After initialization,

The width of the bus can be read from the registers "Link Width Out" and "Link Width In" (see Section 14.5.2).

95

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.

14.2 HyperTransport protocol support

Godson 3A4000's HyperTransport bus supports most commands in the 1.03 / 3.0 protocol, and is

Some extended instructions have been added to the extended consistency protocol that supports multi-chip interconnects. 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 14- 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 1 – Doubleword bit 1: Don't Care bit 0: Don't Care	bit 5: Must be 1, POSTED bit 2: 0 – Byte 1 – Doubleword bit 1: Don't Care bit 0: must be 1
01xxxx	NPC	Read	bit 3: Don't Care bit 2: 0 – Byte 1 – Doubleword bit 1: Don't Care bit 0: Don't Care	bit 3: Don't Care bit 2: 0 – Byte 1 – Doubleword bit 1: Don't Care bit 0: must be 1
110000	R	RdResponse	Read operation returns	
110011	R	TgtDone	Write operation returns	
110100	PC	WrCoherent	----	Write command extension
110101	PC	WrAddr	----	Write address extension
111000	R	RespCoherent	----	Read response extension
111001	NPC	RdCoherent	----	Read command extension
111010	PC	Broadcast	No operation	
111011	NPC	RdAddr	----	Read address extension
111100	PC	FENCE	Guaranteed order relationship	
111111	-	Sync / Error	Sync / Error	

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

Table 14- 3 Commands to be sent out in two modes

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

14.3 HyperTransport interrupt support

The HyperTransport controller provides 256 interrupt vectors, which can support Fix, Arbiter and other types of interrupts. However, there is no support for hardware automatic EOI. For the above two supported types of interrupts, the controller Will be automatically written into the interrupt register, and interrupt the system interrupt controller according to the setting of the interrupt mask register know. For the specific interrupt control, please refer to the description of the interrupt control register in Section 14.5.7.

14.3.1 PIC Interrupt

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 3A4000 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.

14.3.2 Local interrupt handling

In the traditional interrupt processing mode, all interrupts are stored by the interrupt vector inside the HT controller, and then passed

The interrupt line of the HT controller is connected to the interrupt router on the chip for distribution. In this case, the HT interrupt is only limited by

There are several connection methods to interrupt the CPU core, and can not be distributed across slices. The usage scenarios are relatively limited.

In this HT interrupt mode, when performing interrupt processing, the interrupt router on the chip is transparent to the software, and the kernel directly

Go to the interrupt vector of the HT controller (generally 0x90000efdfb000080), and then process it bit by bit,

At this time, no matter how the routing mode is configured, all interrupts on the HT controller are directly read.

14.3.3 Extended interrupt handling

The extended IO interrupt implemented in 3A4000 can greatly increase the flexibility of interrupt distribution and interrupt processing.

In the interrupt extension mode of HT, other interrupts than PIC interrupts are directly written to the chip interrupt router.

On the added extended interrupt register, routing or distribution is performed according to the related configuration of the extended interrupt register.

After using the extended IO interrupt, when performing interrupt processing, the HT controller is transparent to the software, and the core goes directly to the extended IO

The status register (configuration space 0x1800) reads the interrupt status for processing.

When the power is off and processed, no interference will occur between different cores.

The interrupt forwarding is performed on the HT controller by enabling the external interrupt conversion configuration register. As described in 14.5.34,

Software needs to set HT_int_trans to the target address of the extended IO interrupt trigger register. This register in 3A4000

The address is 0x1fe01140, or 0x10000_00001140.

Before the kernel uses extended interrupt processing, it is necessary to enable the corresponding bit in the "other function setting register". The register

The base address is 0x1fe00000 and the offset address is 0x0420.

Table 14- 4 Other function setting registers

Bit field	Field name	access	Reset value	description
51:48	EXT_INT_en	RW	0x0	Extended IO interrupt enable

14.4 HyperTransport address window

14.4.1 HyperTransport space

In the Loongson 3A4000 processor, the default address windows of the four HyperTransport interfaces are as follows:

Table 14- 5 Address window distribution of the default 4 HyperTransport interfaces

Base address	End address	size	definition
0x0A00_0000_0000	0x0AFF_FFFF_FFFF	1 Tbytes	HT0_LO window
0x0B00_0000_0000	0x0BFF_FFFF_FFFF	1 Tbytes	HT0_HI window
0x0E00_0000_0000	0x0EFF_FFFF_FFFF	1 Tbytes	HT1_LO window
0x0F00_0000_0000	0x0FFF_FFFF_FFFF	1 Tbytes	HT1_HI window

By default (not configured separately for the system address window), the software HyperTransport interface to access, in addition, the software can also configure the address window on the crossbar Implement access to it with other address spaces (see section 3.3 for details). 40 inside each HyperTransport interface The address window distribution of the bit address space is shown in the following table.

Table 14- 6 Address window distribution inside the HyperTransport interface of Loongson 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

14.4.2 Internal window configuration of HyperTransport controller

The HyperTransport interface of Loongson 3A4000 processor provides a variety of rich address windows for users. The functions and functions of these address windows are described in the following table.

Table 14- 7 Address window provided in HyperTransport interface of Loongson 3A4000 processor

Address window	Number of windows	Accept bus	effect	Remarks
Receive window (See window configuration 14.5.10)	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 Section 14.5.12)	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.

99

Prefetch window			Determine whether to receive	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

(See window configuration 14.5.13)	2 Internal bus	Department 's Cache access. Fetch access.	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 14.5.14)	2 HyperTransport	Determine whether to HyperTransport Access operations on the bus. For internal Uncache access	IO inside Loongson 3A4000 processor DMA access, by default will be used as Cache access is judged by SCache Whether the break hits, thereby maintaining its IO consistency Signal 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 compliance letter through hardware interest.

14.5 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:

Enable	0x00	Device ID			Vendor ID
	0x04	Status			Command
	0x08	Class Code			Revision ID
	0x0c	BIST	Header Type	Latency Timer	Cache Line Size
	0x10				
	0x14				
	0x18				
	0x1c				
	0x20				
	0x24				
	0x28	Cardbus CIS Pointer			
	0x2c	Subsystem ID			Subsystem Vendor ID
	0x30	Expansion ROM Enable Address			
	0x34	Reserved			Capabilities Pointer

	0x38	Reserved			
	0x3c	Bridge Control		Interrupt Pin	Interrupt Line
	0x40	Command		Capabilities Pointer	Capability ID
	0x44	Link Config 0		Link Control 0	
	0x48	Link Config 1		Link Control 1	
Cap 0 PRI	0x4C	LinkFreqCap0		Link Error0 / Link Freq 0	Revision ID
	0x50	LinkFreqCap1		Link Error1 / Link Freq 1	Feature
	0x54	Error Handling		Enumeration Scratchpad	
	0x58	Reserved		Mem Limit Upper	Mem Enable Upper
Cap 1	0x60	Capability Type	Reserved	Capability Pointer	Capabiliter ID
	0x64	Status 1	Control 1	Status 0	Control 0

Retry	0x68	Retry Count 1		Retry Count 0	
CAP 3	0x6C	Capability Type	Revision ID	Capability Pointer	Capabiliter ID
	0x70	Capability Type	Index	Capability Pointer	Capabiliter ID
CAP 4	0x74	Dataport			
Interrupt	0x78	IntrInfo [31: 0]			
	0x7C	IntrInfo [63:32]			
	0x80	INT Vector [31: 0]			
	0x84	INT Vector [63:32]			
	0x88	INT Vector [95:64]			
	0x8C	INT Vector [127: 96]			
	0x90	INT Vector [159: 128]			
	0x94	INT Vector [191: 160]			
	0x98	INT Vector [223: 192]			
	0x9C	INT Vector [255: 224]			
Int Vector	0xA0	INT Enable [31: 0]			
	0xA4	INT Enable [63:32]			
	0xA8	INT Enable [95:64]			
	0xAC	INT Enable [127: 96]			
	0xB0	INT Enable [159: 128]			
	0xB4	INT Enable [191: 160]			
	0xB8	INT Enable [223: 192]			
	0xBC	INT Enable [255: 224]			
CAP 5	0xC0	Capability Type	Cap Enum / Index	Capability Pointer	Capabiliter ID

Gen3	0xC4	Global Link Training			
	0xC8	Transmitter Configuration 0			
	0xCC	Receiver Configuration 0			
	0xD0	Link Training 0			
	0xD4	Frequency Extension			
	0xD8	Transmitter Configuration 1			
	0xDC	Receiver Configuration 1			
	0xE0	Link Training 1			
	0xE4	BIST Control			

	0x100	Device ID		Vendor ID	
	0x104	Status		Command	
	0x108	Class Code			Revision ID
	0x10c	BIST	Header Type	Latency Timer	Cache Line Size
	0x110				
	0x114				
	0x118				
Enable	0x11c				
	0x120				
	0x124				
	0x128	Cardbus CIS Pointer			
	0x12c	Subsystem ID		Subsystem Vendor ID	
	0x130	Expansion ROM Enable Address			
	0x134	Reserved		Capabilities Pointer	
	0x138	Reserved			
	0x13c	Bridge Control	Interrupt Pin	Interrupt Line	

	0x140	HT RX Enable 0
	0x144	HT RX Mask 0
	0x148	HT RX Enable 1
	0x14C	HT RX Mask 1
Receive	0x150	HT RX Enable 2
Windows	0x154	HT RX Mask 2
	0x158	HT RX Enable 3
	0x15C	HT RX Mask 3
	0x160	HT RX Enable 4
	0x164	HT RX Mask 4
Header Trans	0x168	HT RX Header Trans

102

	0x16C	HT RX EXT Header Trans
	0x170	HT TX Post Enable 0
Post	0x174	HT TX Post Mask 0
Windows	0x178	HT TX Post Enable 1
	0x17C	HT TX Post Mask 1
	0x180	HT TX Prefetchable Enable 0
Prefetchable	0x184	HT TX Prefetchable Mask 0
Windows	0x188	HT TX Prefetchable Enable 1
	0x18C	HT TX Prefetchable Mask 1
	0x190	HT RX Uncache Enable 0
	0x194	HT RX Uncache Mask 0
	0x198	HT RX Uncache Enable 1
Uncache	0x19C	HT RX Uncache Mask 1
Windows	0x1A0	HT RX Uncache Enable 2
	0x1A4	HT RX Uncache Mask 2
	0x1A8	HT RX Uncache Enable 3
	0x1AC	HT RX Uncache Mask 3
	0x1B0	HT RX P2P Enable 0
P2P	0x1B4	HT RX P2P Mask 0
Windows	0x1B8	HT RX P2P Enable 1
	0x1BC	HT RX P2P Mask 1
	0x1C0	APP Configuration 0
APP	0x1C4	APP Configuration 1
Config	0x1C8	RX Bus Value
	0x1CC	PHY status
	0x1D0	TX Buffer 0
Buffer	0x1D4	TX Buffer 1 / Rx buffer hi
	0x1D8	TX Buffer turning
	0x1DC	RX Buffer lo
	0x1E0	Training 0 Counter Short
	0x1E4	Training 0 Counter Long
Training	0x1E8	Training 1 Counter
	0x1EC	Training 2 Counter
	0x1F0	Training 3 Counter
PLL	0x1F4	PLL Configuration
PHY	0x1F8	IO Configuration
	0x1FC	PHY Configuration

103

Page 124

DEBUG	0x240	HT3 DEBUG 0
	0x244	HT3 DEBUG 1
	0x248	HT3 DEBUG 2
	0x24C	HT3 DEBUG 3
	0x250	HT3 DEBUG 4
	0x254	HT3 DEBUG 5
	0x258	HT3 DEBUG 6
POST ID WINDOWS	0x260	HT TX POST ID WIN0
	0x264	HT TX POST ID WIN1
	0x268	HT TX POST ID WIN2
	0x26C	HT TX POST ID WIN3
POST ID WINDOWS	0x270	INT TRANS WIN lo
	0x274	INT TRANS WIN hi

The specific meaning of each register is shown in the following section:

14.5.1 Bridge Control

Offset: 0x3C
Reset value: 0x00000000
name: Bus Reset Control

Table 14- 8 Bus Reset Control Register Definition

Bit field	Bit field name	Bit width	reset value	Visit	description
31:23	Reserved	9	0x0		Keep
twenty two	Reset	1	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	twenty two	0x0		Keep

14.5.2 Capability Registers

Offset: 0x40
Reset value: 0x20010008
name: Command, Capabilities Pointer, Capability ID

Table 14- 9 Command, Capabilities Pointer, Capability ID register definition

104

Page 125

Bit field	Bit field name	Bit width	Reset value	access	description

31:29	Slave / Pri	3	0x0	R	Command format is HOST / Sec
28:26	Reserved	2	0x0	R	Keep
25:21	Unit Count	5	0x0	R / W	Provided to the software for recording the current number of units
20:16	Unit ID	5	0x0		In HOST mode: can be used to record the number of IDs used In SLAVE mode: record your own Unit ID HOST / SLAVE mode is sent by act_as_slave Register control
15:08	Capabilities Pointer	8	0x60	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 14-10 Link Config, Link Control register definition

Bit field	Bit field name	Bit width	Reset value	access	description
30:28	Link Width Out	3	0x0	R / W	Sender width The value after cold reset is the maximum width of the current connection Degrees, the value written to this register will be Effective after reset or HT 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 Degrees, the value written to this register will be Effective after reset or HT 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 , Whether to turn off HT PHY 1: Close 0: do not close
12:10	Reserved	3	0x0		Keep

105

9	CRC Error (hi)	1	0x0	R / W	CRC error in the upper 8 bits
8	CRC Error (lo)	1	0x0	R / W	CRC error in the lower 8 bits
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 low 8-bit HT PHY, high 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 When using the 16-bit HT bus to run the 8-bit protocol, High 8-bit PHY shutdown control

0	Trans off (hi)	1	0x0	R / W	1: Turn off the upper 8-bit HT PHY 0: enable high 8-bit HT PHY

Offset: 0x4C
Reset value: 0x80250023
name: Revision ID, Link Freq, Link Error, Link Freq Cap

Table 14-11 Definition of Revision ID, Link Freq, Link Error, Link Freq Cap register

Bit field	Bit field name	Bit width	Reset value	access	description
31:16	Link Freq Cap	16	0x0000	R	Supported HT bus frequency, according to the design of external PLL Settings produce different values (when using software to configure the PLL (0x1F4), this bit is meaningless) {3.2G, 2.6G, 2.4G, 2.2G, 2.0G, 1.8G, 1.6G, 1.4G, 1.2G, 1.0G, 800M, 600M, 500M, 400 M, 300M, 200M}
15:14	Reserved	2	0x0		Keep
13	Over Flow Error	1	0x0	R	HT bus packet overflow
12	Protocol Error	1	0x0	R / W	Protocol error, refers to unrecognized received on the HT bus command
11: 8	Link Freq	4	0x0	R / W	HT bus operating frequency, after writing the value of this register Will be the next warm reset or HT Disconnect After the entry into force, the set value and Link Freq Cap bit Corresponding (When using software to configure the PLL (0x1F4), the Bit meaningless)
7: 0	Revision ID	8	0x60	R / W	Version number: 3.0

Offset: 0x50
Reset value: 0x00000002
name: Feature Capability

Table 14-12 Feature Capability register definition

Bit field	Bit field name	Bit width	reset value	Visit	description
31: 9	Reserved	twenty three	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

14.5.3 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: 0x64
Reset value: 0x00000000
name: Error Retry Control Register

Table 14-13 Error Retry Control Register

Bit field	Bit field name	Bit width	Reset value	access	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	Maximum number of Short Retry allowed
5: 1	Reserved	5	0x0	R	
0	Link Retry Enable	1	0x0	R / W	Error reconnect function enable control

14.5.4 Retry Count Register

Used for error retransmission count in HyerTransport 3.0 mode.

Offset: 0x68
Reset value: 0x00000000
name: Retry Count register

Table 14-14 Retry Count Register

Bit field	Bit field name	Bit width	reset value	Visit	description
31:16	Reserved	16	0x0	R	Keep
15: 0	Retry Count	16	0x0	R	Retry count

14.5.5 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: 0x6C
Reset value: 0x00200000
name: RevisionID register

Table 14- 15 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

14.5.6 Interrupt Discovery & Configuration

Offset: 0x70
Reset value: 0x80000008
name: Interrupt Capability

Table 14- 16 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

108

Offset: 0x74
Reset value: 0x00000000
name: Dataport

Table 14- 17 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: 0x78
Reset value: 0xF8000000
name: IntrInfo [31: 0]

Table 14- 18 IntrInfo register definition (1)

Bit field	Bit field name	Bit width	reset value	Visit	description
31:24	IntrInfo [31:24]	8	0xF8	R	Keep
23: 2	IntrInfo [23: 2]	twenty two	0x0	R / W	IntrInfo [23: 2], when the PIC interrupt is issued, the value of IntrInfo Used to represent interrupt vector
1: 0	Reserved	2	0x0	R	Keep

Offset: 0x7c
Reset value: 0x00000000
name: IntrInfo [63:32]

Table 14- 19 IntrInfo register definition (2)

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

14.5.7 Interrupt Vector Register

A total of 256 interrupt vector registers, including the direct mapping of Fix, Arbiter and PIC interrupts on the HT bus Up to this 256 interrupt vectors, other interrupts such as SMI, NMI, INIT, INTA, INTB, INTC, INTD can To map to any 8-bit interrupt vector through [28:24] of register 0x50, the 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]}.

By default, 256-bit interrupts can be distributed to 4-bit interrupt lines. When not using high 8-bit HT controller interrupt

109

, You can also distribute 256-bit interrupt to 8-bit interrupt line by setting ht_int_8bit.

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:

Number of interrupts	Stripe	0	1	2	3	4	5	6	7
4	1	[X]	[X + 64]	[X + 128]	[X + 192]	-	-	-	-
X = [63: 0]	2	[2X]	[2X + 1]	[2X + 128]	[2X + 129]	-	-	-	-
	4	[4X]	[4X + 1]	[4X + 2]	[4X + 3]	-	-	-	-
8	1	[X]	[Y]	[X + 64]	[Y + 64]	[X + 128]	[Y + 128]	[X + 192]	[Y + 192]
X = [31: 0]	2	[2X]	[2Y]	[2X + 1]	[2Y + 1]	[2X + 128]	[2Y + 128]	[2X + 129]	[2Y + 129]
Y = [63:32]	4	[4X]	[4X + 32]	[4X + 1]	[4X + 33]	[4X + 2]	[4X + 34]	[4X + 3]	[4X + 35]

Taking the 4-bit interrupt line as an example, the different mapping methods are as follows.

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

110

Page 131

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

Table 14- 20 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 14- 21 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 14- 22 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 14- 23 HT Bus Interrupt Vector Register Definition (4)

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

Offset: 0x90

Reset value: 0x00000000

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

Table 14-31 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 14- 24 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 14- 25 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 14- 26 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

14.5.8 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

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 14- 27 Definition of HT Bus Interrupt Enable Register (1)

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

Offset: 0xa4

Reset value: 0x00000000

name: HT bus interrupt enable register [63:32]

113

Table 14- 28 Definition of HT Bus Interrupt Enable Register (2)

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

Offset: 0xa8

Reset value: 0x00000000

name: HT bus interrupt enable register [95:64]

Table 14- 29 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 14- 30 HT Bus Interrupt Enable Register Definition (4)

Bit field	Bit field name	Bit width	Reset value	access	description
31: 0	Interrupt_mask [127: 96]	32	0x0	R / W	HT bus interrupt enable register [127: 96], Corresponding to interrupt line 1 / HT HI Corresponding to interrupt line 5

Offset: 0xb0

Reset value: 0x00000000

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

Table 14- 31 Definition of HT bus interrupt enable register (5)

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

Offset: 0xb4

Reset value: 0x00000000

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

Table 14- 32 Definition of HT Bus Interrupt Enable Register (6)

Bit field	Bit field name	Bit width	reset value	Visit	description

114

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 14- 33 HT bus interrupt enable register definition (7)

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

Offset: 0xbc

Reset value: 0x00000000

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

Table 14- 34 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

14.5.9 Link Train Register

HyperTransport 3.0 link initialization and link training control register.

Offset: 0xD0

Reset value: 0x00000070

name: Link Train Register

Table 14- 35 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

115

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	AC	1	0x0	R	1: enable 8B10B Whether AC mode is detected 0: AC mode is not detected 1: AC mode detected
0	Reserved	1	0x0	R	Keep

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

addr_out = Multi_node_en?

addr_out_trans [39:37], addr_out_trans [43:40], 3'b0, addr_out [36: 0]:

addr_out_trans;

It should be noted that when configuring the address window register, the high bit of MASK should be all 1, and the low bit should be all 0. 0 in MASK The actual number of bits indicates the size of the address window.

The address in the receive address window is the address received on the HT bus. The HT address falling within the P2P window will be regarded as P2P The command is forwarded back to the HT bus, and the HT address that falls within the normal receive window and is not in the P2P window will be sent to the CPU. The command at its address will be forwarded back to the HT bus as a P2P command.

Offset: 0x140
Reset value: 0x00000000
name: HT bus receive address window 0 enable (external access)

Table 14- 36 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	ht_rx_image0_multi_node_en 1		0x0	R / W HT bus receive address window 0, multi-node address mapping is enabled Convert [39:37] of the address to [46:44]
28	ht_rx_image0_conf_hit_en	1	0x0	R / W HT bus receive address window 0, protocol address hit enable Must be set to 0
25: 0	ht_rx_image0_trans [49:24]	26	0x0	R / W HT bus receive address window 0, the mapped address [49:24]

Offset: 0x144
Reset value: 0x00000000
name: HT bus receive address window 0 base address (external access)

Table 14- 37 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: 0x148
Reset value: 0x00000000
name: HT bus receive address window 1 enable (external access)

Table 14- 38 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	ht_rx_image1_multi_node_en	1	0x0	R / W HT bus receive address window 1, multi-node address mapping is enabled Convert [39:37] of the address to [46:44]
28	ht_rx_image1_conf_hit_en	1	0x0	R / W HT bus receive address window 1, protocol address hit enable Must be set to 0
25: 0	ht_rx_image1_trans [49:24]	26	0x0	R / W HT bus receive address window 1, the mapped address [49:24]

Offset: 0x14c
Reset value: 0x00000000
name: HT bus receive address window 1 base address (external access)

Table 14- 39 HT Bus Receive Address Window 1 Base Address (External Access) Register Definition

117

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: 0x150
Reset value: 0x00000000
name: HT bus receive address window 2 enable (external access)

Table 14- 40 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	ht_rx_image2_multi_node_en 1		0x0	R / W HT bus receive address window 2, multi-node address mapping is enabled Convert [39:37] of the address to [46:44]
28	ht_rx_image2_conf_hit_en	1	0x0	R / W HT bus receive address window 2, protocol address hit enable Must be set to 0
25: 0	ht_rx_image2_trans [49:24]	26	0x0	R / W HT bus receive address window 2, the mapped address [49:24]

Offset: 0x154
Reset value: 0x00000000
name: HT bus receive address window 2 base address (external access)

Table 14- 41 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: 0x158
Reset value: 0x00000000
name: HT bus receive address window 3 enable (external access)

Table 14- 42 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

Bit field	Bit field name	Bit width	reset value	Visit description
29	ht_rx_image3_multi_node_en	1	0x0	R / W HT bus receive address window 3, multi-node address mapping is enabled Convert [39:37] of the address to [46:44]
28	ht_rx_image3_conf_hit_en	1	0x0	R / W HT bus receive address window 3, protocol address hit enable Must be set to 0
25: 0	ht_rx_image3_trans [49:24]	26	0x0	R / W HT bus receive address window 3, the mapped address [49:24]

Offset: 0x15C
Reset value: 0x00000000
name: HT bus receive address window 3 base address (external access)

Table 14- 43 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: 0x160
Reset value: 0x00000000
name: HT bus receive address window 4 is enabled (external access)

Table 14- 44 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	ht_rx_image4_multi_node_en	1	0x0	R / W HT bus receive address window 4, multi-node address mapping is enabled Convert [39:37] of the address to [46:44]
28	ht_rx_image4_conf_hit_en	1	0x0	R / W HT bus receive address window 4, protocol address hit enable Must be set to 0
25: 0	ht_rx_image4_trans [49:24]	26	0x0	R / W HT bus receive address window 4, the mapped address [49:24]

Offset: 0x164
Reset value: 0x00000000
name: HT bus receive address window 4 base address (external access)

Table 14- 45 HT Bus Receive Address Window 4 Base Address (External Access) Register Definition

Bit field	Bit field name	Bit width	reset value	Visit description

Bit field	Bit field name	Bit width reset value	Visit description
31:16	ht_rx_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]

14.5.11 Configuration Space Conversion Register

Used to perform various conversions on the HT configuration space.

Offset: 0x168
Reset value: 0x00000000
name: Configuration space extended address translation

Table 14-46 Definition of Extended Address Translation Register in Configuration Space

Bit field	Bit field name	Bit width	Reset value	access	description
31	ht_rx_header_trans_ext	1	0x1	R / W	Convert the configuration space (0xFD_FE000000) After the address1 flag bit is adjusted by 24 bits Up to 28 bits for space with EXT HEADER Unite
30	ht_rx_header_trans_en	1	0x1	R / W	Enable configuration space (0xFD_FE000000) High-order address ([39:24]) conversion
29: 0	ht_rx_header_trans [53:24]	30	0xFE00	R / W	High address after configuration space conversion [53:24] (Only [53:25] is available)

Offset: 0x16C
Reset value: 0x00000000
name: Extended Address Translation

Table 14-47 Definition of Extended Address Translation Register

Bit field	Bit field name	Bit width	Reset value	access	description
30	ht_rx_ext_header_trans_en	1	0x0	R / W	Enable expansion configuration space (0xFE_00000000) high address ([39:28]) Conversion
29: 0	ht_rx_ext_header_trans [53:24]	30	0x0	R / W	High address after conversion of extended configuration space [53:24] (actually only [53:29] is available)

14.5.12 POST address window configuration register

For the address window hit formula, see section 14.5.10.

120

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

Table 14-48 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_split0_en	1	0x0	R / W HT access unpacking enable (corresponding to external CPU core uncache ACC operation window)
29:23	Reserved	14	0x0	Keep
15: 0	ht_post0_trans [39:24]	16	0x0	R / W HT bus POST address window 0, the translated address [39:24]

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

Table 14-49 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: 0x178
Reset value: 0x00000000
name: HT bus POST address window 1 enable (internal access)

Table 14-50 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_split1_en	1	0x0	R / W HT access unpacking enable (corresponding to external CPU core uncache ACC operation window)
29:16	Reserved	14	0x0	Keep
15: 0	ht_post1_trans [39:24]	16	0x0	R / W HT bus POST address window 1, the translated address [39:24]

Offset: 0x17c
Reset value: 0x00000000
name: HT bus POST address window 1 base address (internal access)

Table 14-51 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]

14.5.13 Prefetchable address window configuration register

For the address window hit formula, see section 14.5.10.

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: 0x180
Reset value: 0x00000000

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

Table 14- 52 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
30:16	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: 0x184

Reset value: 0x00000000

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

Table 14- 53 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: 0x188

Reset value: 0x00000000

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

Table 14- 54 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
30:16	Reserved	15	0x0	Keep
15: 0	ht_prefetch1_trans [39:24]	16	0x0	R / W HT bus can pre-fetch the address window 1, the translated address [39:24]

Offset: 0x18c

Reset value: 0x00000000

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

Table 14- 55 HT Bus Prefetchable Address Window 1 Base Address (Internal Access)

Bit field	Bit field name	Bit width	reset value	Visit description
31:16	ht_prefetch1_base [39:24]	16	0x0	R / W HT bus can prefetch address window 1, address base address [39:24]
15: 0	ht_prefetch1_mask [39:24]	16	0x0	R / W HT bus can prefetch address window 1, address masked [39:24]

14.5.14 UNCACHE address window configuration register

For the address window hit formula, see section 14.5.10.

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 improve memory access efficiency, such as video memory access.

Offset: 0x190

Reset value: 0x00000000

name: HT bus Uncache address window 0 enable (internal access)

Table 14- 56 HT Bus Uncache Address Window 0 Enable (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	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 0, mapping enable signal
29	ht_uncache0_multi_node_en	1	0x0	R / W HT bus uncache receive address window 0, multi-node ground Address mapping enable
28	ht_uncache0_conf_hit_en	1	0x0	R / W HT bus uncache receiving address window 0, protocol address Hit enable
25: 0	ht_uncache0_trans [49:24]	26	0x0	R / W HT bus uncache address window 0, the translated address [49:24]

Offset: 0x194

Reset value: 0x00000000

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

Table 14-57 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: 0x198

Reset value: 0x00000000

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

Table 14-58 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	ht_uncache1_multi_node_en	1	0x0	R / W HT bus uncache receive address window 1, multi-node ground Address mapping enable
28	ht_uncache1_conf_hit_en	1	0x0	R / W HT bus uncache receive address window 1, protocol address Hit enable
25: 0	ht_uncache1_trans [49:24]	26	0x0	R / W HT bus uncache address window 1, the translated address [49:24]

Offset: 0x19c

Reset value: 0x00000000

Page 145

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

Table 14- 59 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: 0x1A0
Reset value: 0x00000000
name: HT bus Uncache address window 2 enable (internal access)

Table 14- 60 HT Bus Uncache Address Window 2 Enable (Internal Access)

Bit field	Bit field name	Bit width	reset value	Visit description
31	ht_uncache2_en	1	0x0	R / W HT bus uncache address window 2, enable signal
30	ht_uncache2_trans_en	1	0x0	R / W HT bus uncache address window 2, mapping enable signal
29	ht_uncache2_multi_node_en	1	0x0	R / W HT bus uncache receive address window 2, multi-node address Mapping enable
28	ht_uncache2_conf_hit_en	1	0x0	R / W HT bus uncache receive address window 2, protocol address command Enable
25: 0	ht_uncache2_trans [49:24]	26	0x0	R / W HT bus uncache address window 2, the translated address [49:24]

Offset: 0x1A4
Reset value: 0x00000000
name: HT bus Uncache address window 2 base address (internal access)

Table 14- 61 HT Bus Uncache Address Window 2 Base Address (Internal Access)

Bit field	Bit field name	Bit width	reset value	Visit description
31:16	ht_uncache2_base [39:24]	16	0x0	R / W HT bus uncache address window 2, of the base address [39:24]
15: 0	ht_uncache2_mask [39:24]	16	0x0	R / W HT bus uncache address window 2, address masked [39:24]

Offset: 0x1A8
Reset value: 0x00000000
name: HT bus Uncache address window 3 enable (internal access)

Table 14- 62 HT Bus Uncache Address Window 3 Enable (Internal Access)

Bit field	Bit field name	Bit width	reset value	Visit description

Page 146

31	ht_uncache3_en	1	0x0	R / W HT bus uncache address window 3, enable signal
30	ht_uncache3_trans_en	1	0x0	R / W HT bus uncache address window 3, mapping enable signal
29	ht_uncache3_multi_node_en	1	0x0	R / W HT bus uncache receive address window 3, multi-node address Mapping enable
28	ht_uncache3_conf_hit_en	1	0x0	R / W HT bus uncache receive address window 3, protocol address command Enable

25: 0	ht_uncache3_trans [49:24]	26	0x0	R / W HT bus uncache address window 3, the translated address [49:24]

Offset: 0x1AC
Reset value: 0x00000000
name: HT bus Uncache address window 3 base address (internal access)

Table 14- 63 HT Bus Uncache Address Window 3 Base Address (Internal Access)

Bit field	Bit field name	Bit width	reset value	Visit description
31:16	ht_uncache3_base [39:24]	16	0x0	R / W HT bus uncache address window 3, address base address [39:24]
15: 0	ht_uncache3_mask [39:24]	16	0x0	R / W HT bus uncache address window 3, address masked [39:24]

14.5.15 P2P Address Window Configuration Register

For the address window hit formula, see section 14.5.10.

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: 0x1B0
Reset value: 0x00000000
name: HT bus P2P address window 0 enable (external access)

Table 14-64 HT Bus P2P Address Window 0 Enable (External Access) Register Definition

Bit field	Bit field name	Bit width	reset value	Visit description
31	ht_rx_p2p0_en	1	0x0	R / W HT bus P2P address window 0, enable signal
29: 0	ht_rx_p2p0_trans [53:24]	30	0x0	R / W HT bus P2P address window 0, translated address [53:24]

Offset: 0x1B4
Reset value: 0x00000000

126

name: HT bus P2P address window 0 base address (external access)

Table 14- 65 HT bus P2P address window 0 base address (external access) register definition

Bit field	Bit field name	Bit width	reset value	Visit description
31:16	ht_rx_p2p0_base [39:24]	16	0x0	R / W HT bus P2P address window 1, address base address [39:24]
15: 0	ht_rx_p2p0_mask [39:24]	16	0x0	R / W HT bus P2P address window 1, address masked [39:24]

Offset: 0x1B8
Reset value: 0x00000000
name: HT bus P2P address window 1 enable (external access)

Table 14- 66 HT bus P2P address window 1 enable (external access) register definition

Bit field	Bit field name	Bit width	reset value	Visit description
31	ht_rx_p2p1_en	1	0x0	R / W HT bus P2P address window 1, enable signal
29: 0	ht_rx_p2p1_trans [53:24]	30	0x0	R / W HT bus P2P address window 1, the translated address [53:24]

Offset: 0x1BC
Reset value: 0x00000000
name: HT bus P2P address window 1 base address (external access)

Table 14- 67 HT bus P2P address window 1 base address (external access) register definition

Bit field	Bit field name	Bit width	reset value	Visit description
31:16	ht_rx_p2p1_base [39:24]	16	0x0	R / W HT bus P2P address window 1, address base address [39:24]
15: 0	ht_rx_p2p1_mask [39:24]	16	0x0	R / W HT bus P2P address window 1, address masked [39:24]

14.5.16 Controller parameter configuration register

Offset: 0x1C0
Reset value: 0x00904321
name: APP CONFIG 0

Table 14- 68 Definition of Controller Parameter Configuration Register 0

Bit field	Bit field name	Bit width	Reset value	access	description
31:30	Reserved	1	0x0		Keep
29	Ldt Stop Gen	1	0x0	R / W	Put the bus into LDT DISCONNECT mode The correct method is: 0-> 1
28	Ldt Req Gen	1	0x0	R / W	Wake up the HT bus from LDT DISCONNECT, set Set LDT_REQ_n

127

					The correct way is to set 0 first and then set 0: 0-> 1 In addition, it is also possible to issue read and write requests directly to the bus Automatic wake-up bus
27	rx sample en	1	0x0	R / W	Enable the cad and ctl of sampling input, send in Display in memory for debugging
26	Dword Write	1	0x1	R / W	For 32/64/128/256 bit MEM write access, whether to use Use the Dword Write command format (in Byte Write mode) Writes are converted to 128-bit writes with MASK when received)
25	Dword Write cfg	1	0x1	R / W	Whether to use Dword for write access to configuration space Write command format (write in the Byte Write mode is received Will be converted to 128-bit write with MASK)
twenty four	Dword Write IO	1	0x1	R / W	For write access to IO space, whether to use Dword Write Command format (write in Byte Write mode will be transferred when received Change to 128-bit write with MASK)
twenty three	axi aw resize	1	0x0	RW	Whether to write to 128 bit with MASK and press Mask for size Reset
twenty two	Coherent Mode	1	0x0	RW	Whether it is the processor consistency mode, the initial value is ICCC_EN pin decision, effective after reset
twenty one	Coherent_split	1	0x0	RW	In consistency mode, all packets are split into 32 bytes for processing
20	Not Care Seqid	1	0x0	R / W	Whether the receiving end does not care about the HT order relationship
19:16	Fixed Seqid	4	0x0	R / W	When Not Axi2Seqid is valid, configure the HT bus to issue 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 Increased priority strategy, the set of registers is used to configure each Channel's initial priority

Offset: 0x1C4

Reset value: 0x00904321

name: APP CONFIG1

Table 14- 69 Definition of Controller Parameter Configuration Register 1

Bit field	Bit field name	Bit width	Reset value	access	description
31	tx post split en	1	0x0	R / W	Enable the write and unpack function when the tx post ID window hits (All write requests that cross the 32-byte boundary will be split into

128

Page 149

					Two consecutive write requests (byte write)
30	tx wr passPW pc	1	0x0	R / W	Write the passPW of all issued Post channel requests Bit set to 1
29	tx wr passPW npc	1	0x0	R / W	Write all nonpost channel write requests passPW bit is set to 1
28	tx rd passPW	1	0x0	R / W	Set the passPW bit of all issued read requests to 1
27	stop same id wr	1	0x0	R / W	The sender stops when it encounters a write request with the same AXI ID Send until the previous request with the same ID returns
26	stop same id rd	1	0x0	R / W	The sender stops when it encounters a read request with the same AXI ID Send until the previous request with the same ID returns
25	Not axi2seqid wr	1	0x0	R / W	Prohibit write request AXI ID to seqid conversion, directly Use fixed seqid
twenty four	Not axi2seqid rd	1	0x0	R / W	Prohibit the read request AXI ID to seqid conversion, directly Use fixed seqid
23:22	Reserved	2	0x0	R / W	Keep
twenty one	act as slave	1	0x1	R / W	Set SLAVE mode
20	Host hide	1	0x0	R / W	Forbid the receiver to access the configuration register space
19:16	Rrequest delay	4	0x3	R / W	Used to control Rrequest transmission in consistency mode Random 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	Crc Int en	1	0x0	R / W	Enable interrupt transmission when CRC error
14:12	Crc Int route	3	0x0	R / W	Interrupt pin selection during CRC interrupt
11	Reserved				
10	ht int 8 bit	1	0x0	R / W	Use 8 interrupt lines
9: 8	ht_int_stripe	2	0x0	R / W	Corresponding to 3 interrupt routing methods, see the detailed description in Break vector register 0x0: ht_int_stripe_1 0x1: ht_int_stripe_2 0x2: ht_int_stripe_4
4: 0	Interrupt Index	5	0x0	R / W	Redirect all interrupts other than the standard interrupt to Which interrupt vector (including SMI, NMI, INIT, INTA, INTB, INTC, INTD) A total of 256 interrupt vectors, this register represents

129

Page 150

The upper 5 bits of the interrupt vector, the internal interrupt vector is as follows:

000: SMI

001: NMI

010: INIT

011: Reserved

100: INTA

101: INTB

110: INTC

111: INTD

14.5.17 Receive Diagnostic Register

Offset: 0x1C8

Reset value: 0x00000000

name: Receive Diagnostic Register

Table 14-70 Receive Diagnostic Register

Bit field	Bit field name	Bit width	reset value	Visit	description
31:16	rx_cad_phase_0	16	0x0	R / W	save the input CAD [15: 0] value obtained by sampling
15: 8	rx_ctl_catch	8	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
7: 0					

14.5.18 PHY Status Register

Used to observe the PHY related status, debug use

Offset: 0x1CC

Reset value: 0x83308000

name: PHY status register

Table 14- 71 PHY Status Register

Bit field	Bit field name	Bit width	Reset value	access	description
31:29	Reserved	3	0x0	R	Keep
28	dll locked hi	1	0x0	R	
27	dll locked lo	1	0x0	R	
26	cdr locked hi	1	0x0	R	
25	cdr locked lo	1	0x0	R	
twenty four	phase locked	1	0x0	R	

130

Page 151

23:20	phy state	4	0x0	R
19:17	tx training status	3	0x0	R
16:14	rx training status	3	0x0	R
13: 8	Init done	6	0x0	R
7: 0	Reserved	8		R

14.5.19 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: 0x1D0

Reset value: 0x00000000

name: Command send buffer size register

Table 14-72 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
23:16	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

14.5.20 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: 0x1D4

Reset value: 0x00000000

name: Data transmission buffer size register

Table 14- 73 Data transmission buffer size register

Bit field	Bit field name	Bit width	Reset value	access	description
31	Reserved	1	0x0	R	Keep
30	rx_buffer_r_data [4]	1	0x0	R / W	Receive buffer read data initialization Bit [4] of information
29	rx_buffer_npc_data [4]	1	0x0	R / W	Receive buffer npc data buffer initial Bit of information [4]
28	rx_buffer_pc_data [4]	1	0x0	R / W	Initialize the pc data buffer of the receive buffer Bit [4] of information
27	rx_buffer_b_cmd [4]	1	0x0	R / W	Receive buffer bresponse command

131

					buffer initialization information bit [4]
26	rx_buffer_r_cmd [4]	1	0x0	R / W	Receive buffer read command initialization Bit [4] of information
25	rx_buffer_npc_cmd [4]	1	0x0	R / W	Receive buffer npc command buffer initial Bit of information [4]
twenty four	rx_buffer_pc_cmd [4]	1	0x0	R / W	Receive buffer pc command buffer initialization Bit [4] of information
23:16	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

14.5.21 Send buffer debug register

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

Adjust the number of different send buffers.

Offset: 0x1D8

Reset value: 0x00000000
name: Send buffer debug register

Table 14-74 Send Buffer Debug Register

Bit field	Bit field name	Bit width	Reset value	access	description
31	b_interleave	1	0x0	R / W	In consistency mode, enable interleaving of B channel with other channels transmission
30	nop_interleave	1	0x0	R / W	Enable interleaved transmission of flow control packets and other virtual channels
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 A When tx_neg is 1, reduce NPC_DATA_txadj + 1 Pc
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 Pc
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 Pc
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 Pc

14.5.22 Receive buffer initial register

Offset: 0x1DC
Reset value: 0x07778888
name: Receive buffer initialization configuration register

Table 14-75 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

14.5.23 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: 0x1E0
Reset value: 0x00000080
name: Training 0 timeout short count register

Table 14- 76 Training 0 Timeout Short Timer Register

Bit field	Bit field name	Bit width	reset value	Visit description
31	Gen3_timing_soft	1	0x0	R / W
30:23	Retry_nop_num	8	0x0	R / W
22: 0	T0 time	twenty three	0x80	R / W Training 0 Timeout short timer register

14.5.24 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: 0x1E4
Reset value: 0x000fffff
name: Training 0 timeout long count register

Table 14- 77 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

14.5.25 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: 0x1E8
Reset value: 0x0004fffff
name: Training 1 count register

Table 14- 78 Training 1 Count Register

Page 155

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

14.5.26 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: 0x1EC
Reset value: 0x0007fffff
name: Training 2 count register

Table 14- 79 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

14.5.27 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: 0x1F0
name: Training 3 count register

Table 14- 80 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

14.5.28 Software Frequency Configuration Register

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

The specific switching method is: on the premise of enabling the software configuration mode, set bit 1 of the software frequency configuration register, and Write parameters related to the new clock, including div_refc and div_loop that determine the output frequency of the PLL Frequency coefficients phy_hi_div and phy_lo_div, and the frequency division coefficient core_div of the controller. Then enter the warm reset or LDT disconnect, the controller will automatically reset the PLL and configure new clock parameters.

135

Page 156

PHY_LINK_CLK is the HT bus frequency.

The calculation formula of the clock frequency is:

When using SYS_CLOCK as the reference clock input and SYS_CLOCK is 25MHz (CLKSEL [8] is 1 and CLKSEL [5]

Is 1), the frequency calculation method is:

HyperTransport 1.0:

PHY_LINK_CLK = 12.5MHz × div_loop / div_refc / phy_div

HyperTransport 3.0:

PHY_LINK_CLK = 25MHz × div_loop / div_refc / phy_div

In other cases, the frequency calculation method is:

HyperTransport 1.0:

PHY_LINK_CLK = 50MHz × div_loop / div_refc / phy_div

HyperTransport 3.0:

PHY_LINK_CLK = 100MHz × div_loop / div_refc / phy_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.

It should be noted that in 3A4000, HT_CORE_CLK is no longer controlled by this configuration, but is divided by the NODE clock

Frequency control.

Offset: 0x1F4
Reset value: 0x00000000
name: Software frequency configuration register

Table 14- 81 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 counter When select, the upper limit of counter count is {PLL_relock_counter, 5'h1f}, otherwise count The upper limit 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	Lower PHY Divider

21: 18	Soft_phy_hi_div	4	0x0	R / W	High 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	Frequency multiplication factor in PLL
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

14.5.29 PHY Impedance Match Control Register

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

Offset: 0x1F8

Reset value: 0x00000000
name: PHY impedance matching control register

Table 14- 82 Impedance Matching Control Register

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

14.5.30 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 low-order PHY are independently controlled by two controllers; when the controller acts as a 16-bit controller, the high-order

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

Offset: 0x1FC
Reset value: 0x83308000
name: PHY configuration register

Table 14- 83 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

14.5.31 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: 0x240
Reset value: 0x00000000
name: Link initialization debug register

Table 14-84 Link Initialization Debug Register

Bit field	Bit field name	Bit width	reset value	access	description
15	Cdr_ignore_enable 1		0x0	R / W	Whether to ignore CRC lock during link initialization and count through counter Wait for completion: 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	Wait for the upper limit of the counter to count and complete the count based on the controller clock

14.5.32 LDT debug register

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

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

Clock.

Offset: 0x244
Reset value: 0x00000000
name: LDT debug register 1

Table 14- 85 LDT debug register 1

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

Offset: 0x248
Reset value: 0x00000000
name: LDT debug register 2

Table 14- 86 LDT debug register 2

Bit field	Bit field name	Bit width	Reset value	access	description
31:30	Reserved	16	0x0	R / W	
29: 0	rx lane ts 0	16	0x0	R / W	

Offset: 0x24C
Reset value: 0x00000000
name: LDT debug register 3

Table 14- 87 LDT debug register 3

Bit field	Bit field name	Bit width	Reset value	access	description

31:30	Reserved	16	0x0	R / W
29: 0	rx lane ts 1	16	0x0	R / W

Offset: 0x250
Reset value: 0x00000000
name: LDT debug register 4

Table 14- 88 LDT debug register 4

Bit field	Bit field name	Bit width	Reset value	access	description
31:30	Reserved	16	0x0	R / W	
29: 0	rx lane ts 2	16	0x0	R / W	

Offset: 0x254
Reset value: 0x00000000
name: LDT debug register 5

Table 14- 89 LDT debug register 5

Bit field	Bit field name	Bit width	Reset value	access	description
31:22	Reserved	10	0x0	R / W	
21:18	wait ctl	4	0x0	R / W	
17: 0	phase lock	18	0x0	R / W	

Offset: 0x258
Reset value: 0x00000000
name: LDT debug register 5

Table 14- 90 LDT debug register 5

Bit field	Bit field name	Bit width	Reset value	access	description
31: 0	wait cad	32	0x0	R / W	

14.5.33 HT TX POST ID window configuration register

This window compares the ID of the internal write request with the preset window and passes the hit request through the HT POST channel

Outgoing.

Offset: 0x260
Reset value: 0x00000000
name: HT TX POST ID WIN0

140

Table 14- 91 HT TX POST ID WIN0

Bit field	Bit field name	Bit width	Reset value	access	description

31:16	HT TX POST ID0 MASK	16	0x0	R / W	AXI ID hit requests use POST Window transmission, MASK bit of ID
15: 0	HT TX POST ID0 BASE	16	0x0	R / W	AXI ID hit requests use POST Window transmission, BASE bit of ID

Offset: 0x264
Reset value: 0x00000000
name: HT TX POST ID WIN1

Table 14- 92 HT TX POST ID WIN1

Bit field	Bit field name	Bit width	Reset value	access	description
31:16	HT TX POST ID1 MASK	16	0x0	R / W	AXI ID hit requests use POST Window transmission, MASK bit of ID
15: 0	HT TX POST ID1 BASE	16	0x0	R / W	AXI ID hit requests use POST Window transmission, BASE bit of ID

Offset: 0x268
Reset value: 0x00000000
name: HT TX POST ID WIN2

Table 14- 93 HT TX POST ID WIN2

Bit field	Bit field name	Bit width	Reset value	access	description
31:16	HT TX POST ID2 MASK	16	0x0	R / W	AXI ID hit requests use POST Window transmission, MASK bit of ID
15: 0	HT TX POST ID2 BASE	16	0x0	R / W	AXI ID hit requests use POST Window transmission, BASE bit of ID

Offset: 0x26C
Reset value: 0x00000000
name: HT TX POST ID WIN3

Table 14- 94 HT TX POST ID WIN3

Bit field	Bit field name	Bit width	Reset value	access	description
31:16	HT TX POST ID3 MASK	16	0x0	R / W	AXI ID hit requests use POST Window transmission, MASK bit of ID
15: 0	HT TX POST ID3 BASE	16	0x0	R / W	AXI ID hit requests use POST Window transmission, BASE bit of ID

141

14.5.34 External Interrupt Conversion Configuration

This setting converts the interrupt received by HT into a write operation to a specific address and directly writes to the expansion IO inside the chip Interrupt vector, instead of generating an interrupt within the HT controller. In this way, you can use the IO interrupt directly across the slice Advanced features such as distribution.

Offset: 0x270
Reset value: 0x00000000
name: HT RX INT TRANS Lo

Table 14- 95 HT RX INT TRANS LO

Bit field	Bit field name	Bit width	Reset value	access	description
31: 4	INT_trans_addr [31: 4]	28	0x0	R / W	Interrupt conversion address low

3: 0	Reserved	4	0x0	R	Keep

Offset: 0x274
Reset value: 0x00000000
name: HT RX INT TRANS Hi

Table 14- 96 HT RX INT TRANS Hi

Bit field	Bit field name	Bit width	Reset value	access	description
31	INT_trans_en	1	0x0	R / W	Interrupt conversion enable
30	INT_trans_allow	1	0x0	R / W	Interrupt conversion enable After setting this bit, INT_trans_en or Only the EXT_INT_en of the chip can be born effect.
29:26	INT_trans_cache	4	0x0	R / W	Interrupt conversion of Cache field
25: 0	INT_trans_addr [57:32]	26	0x0	R / W	Interrupt conversion address high

14.6 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 14-6, the address range of the HT bus configuration space The range is 0xFD_FE00_0000 to 0xFD_FFFF_FFFF. For configuration access in the HT protocol, it is adopted in Godson 3A4000 Use the following format:

Type 0:

Type 1:

Figure 14- 1 Configuration access of HT protocol in Loongson 3A4000

14.7 HyperTransport multiprocessor support

Loongson 3 processor uses HyperTransport interface for multi-processor interconnection, and can be automatically maintained by hardware Consistency request between 2-8 chips.

Loongson No. 3 interconnection routing

There are two methods for Loongson No. 3 interconnection routing. When routing, X followed by Y, with four Take a chip as an example, the ID numbers are 00, 01, 10, and 11, respectively. If a request is sent from 11 to 00, it is 11 to 00 routing, First go in the X direction, from 11 to 10, then in the Y direction, from 10 to 00. When the response returns 11 from 00, the route First go in the X direction, from 00 to 01, then go in the Y direction, from 01 to 11. The other is direct access to the diagonal through hard The device connects two diagonal chips to achieve direct access, which greatly reduces the access delay. This access method needs to be individually enabled by software.

can. Due to the characteristics of this algorithm, we can use many different methods when constructing multi-chip interconnects.

Four piece Loongson No. 3 interconnection structure

The four CPUs are connected in pairs to form a ring structure. Each CPU uses two 8-bit controllers of HT0 to connect with two adjacent chips, Using HT1 HI to connect with the diagonal chip, the interconnect structure as shown below is obtained:

Figure 14- 2 Four-chip Loongson No. 3 interconnection structure

Eight piece Loongson No. 3 interconnection structure

Eight CPUs form a cube structure. Each CPU uses two 8-bit controllers of HT0 to connect with two adjacent chips. HT1 thus obtains the interconnection structure shown below:

Figure 14- 3 Eight piece Loongson No. 3 interconnection structure

144

Page 165

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 interconnected by four chips. 8-bit HT bus at the same time. As follows:

Figure 14- 4 Two-chip Loongson No. 3 8-bit interconnection structure

However, the 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 commands sent to the The order will be sent to HT0_LO instead of the previous routing table to HT0_HI or HT0_LO, so that we You can use a 16-bit bus when interconnecting. Therefore, we only need to correctly configure the 16-bit mode of CPU0 and CPU1 and The 16-bit HT bus can be used to interconnect the high and low bus correctly. And this interconnect structure can also use 8-bit HT bus protocol for mutual access. The resulting interconnect structure is as follows:

145

Page 166

Figure 14- 5 Two-chip Loongson No. 3 16-bit interconnection structure

15 Low-speed IO controller configuration

Loongson No. 3 I / O controller includes UART controller, SPI controller, I2C and GPIO registers. These I / O controls

The controller shares an AXI port, and the CPU request is sent to the corresponding device after address decoding.

15.1 UART controller

The UART controller has the following features

- Full-duplex asynchronous data reception / transmission

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

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

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

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

For these two UARTs, a physical address is also provided, which is 0x1FE00100 (UART0) and 0x1FE00110 (UART1). This group of addresses can access the two newly added registers RFC and TFC.

15.1.1 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

15.1.2 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'-off '1'-open
2	ILE	1	RW	Receiver line status interrupt enable '0' – close '1' – open
1	ITxE	1	RW	Transfer save register is empty Interrupt enable '0' – close '1' – open
0	IRxE	1	RW	Receive valid data interrupt enable '0' – close '1' – open

15.1.3 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	Read 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

15.1.4 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

149

0	Reserved	1	W	Keep

15.1.5 Line Control Register (LCR)

Chinese name: Line control register

Register bit width: [7: 0]

Offset: 0x03

Reset value: 0x03

Bit field	Bit field name	Bit width	access	description
7	dlab	1	RW	Divider latch access bit '1'-access to the operation divider latch '0'-access to normal operation register
6	bcb	1	RW	Interrupt control bit '1'-At this time the output of the serial port is set to 0 (interrupted state). '0'-normal operation
5	spb	1	RW	Specify parity '0' – no parity bit specified '1' – transmission and check parity if LCR [4] bit is 1 The bit is 0. If the LCR [4] bit is 0, transmit and check the parity The checkpoint is 1.
4	eps	1	RW	Parity bit selection '0' – There are an odd number of 1s in each character (including data and odd Even parity bit) '1' – there are an even number of 1s in each character

150

3	pe	1	RW	Parity bit enable '0' – no parity bit '1'-generate parity bit on output, judge odd on input Even parity
2	sb	1	RW	Define the number of generated stop bits '0' – 1 stop bit '1' – 1.5 stop bits when 5 characters long, others The length is 2 stop bits
1: 0	bec	2	RW	Set the number of digits for each character '00' – 5 digits '01' – 6 digits '10' – 7 digits '11' – 8 digits

15.1.6 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

Bit field	Bit field name	Bit width	access	description
				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

15.1.7 Line Status Register (LSR)

Chinese name: Line status register

Register bit width: [7: 0]

Offset: 0x05

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7	ERROR	1	R	Error indication bit '1'-at least parity error, framing error or interruption The broken one. '0' – no errors
6	TE	1	R	Transmission is empty '1' – Both the transmission FIFO and the transmission shift register are empty. give

				Clear when the transmit FIFO writes data '0' – with data
5	TFE	1	R	Transmit FIFO bit empty representation bit '1' – The current transmit FIFO is empty, write data to the transmit FIFO Time zero '0' – with data
4	BI	1	R	Interrupt interruption bit '1'-Start bit + data + parity bit + stop bit received Is 0, that is interrupted '0'-no interruption
3	FE	1	R	Frame error indication bit '1' – received data has no stop bit '0' – no errors
2	PE	1	R	Parity bit error indicates bit

				'1'-The current received data has a parity error
				'0' – no parity error
1	OE	1	R	Data overflow indication bit
				'1'-There is data overflow
				'0' – no overflow
0	DR	1	R	Receive data valid representation bit
				'0' – No data in FIFO
				'1' – There is data in the FIFO

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

153

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

15.1.8 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

15.1.9 Receive FIFO count value (RFC)

Chinese name: Receive FIFO count value

Register bit width: [7: 0]

Offset: 0x08

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description

154

7: 0	RFC	8	R	Reflect the number of valid data in the current receive FIFO

15.1.10 Transmit FIFO count value (TFC)

Chinese name: send FIFO count value

Register bit width: [7: 0]

Offset: 0x09

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7: 0	TFC	8	R	Reflect the number of valid data in the current transmit FIFO

15.1.11 Frequency division latch

Chinese name: Divider 1

Register bit width: [7: 0]

Offset: 0x00

Reset value: 0x00

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

Chinese name: Divider 2

Register bit width: [7: 0]

Offset: 0x01

Reset value: 0x00

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

Chinese name: Frequency Division Latch 3

Register bit width: [7: 0]

Offset: 0x02

155

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7: 0	D_DIV	8	RW	Stores the fractional divider value of the divider latch

15.1.12 Use of new registers

The new receive FIFO counter (RFC) allows the CPU to detect the number of valid data in the receive FIFO. According to this, the CPU Multiple data can be read continuously after receiving an interrupt to improve CPU's ability to process UART received data;

The transmit FIFO counter (TFC) allows the CPU to detect the number of valid data in the transmit FIFO, according to which the CPU can keep Proof that the sending FIFO does not overflow, and continuously send multiple data to improve the CPU's ability to process UART sending data

Frequency-dividing latch 3 (that is, decimal frequency-dividing register) is used to solve the problem that the required baud rate cannot be accurately obtained by integer divis problem. Divide the reference clock 100MHz by 16, and then divide by the baud rate.

For MSB and LSB, the fractional part is multiplied by 256 and assigned to the divider latch D_DIV.

15.2 SPI controller

The SPI controller has the following features:

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

The base address of the physical address of the SPI controller register is 0x1FE001F0.

Table 15- 1 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_01F0-0X1FE0_01FF	16Byte

The SPI Boot address space is the address space that the processor first accesses when the system starts. The address of 0xBFC00000 is selected from Automatic routing to SPI.

The SPI Memory space can also be directly accessed through the CPU's read request, its minimum 1M bytes and SPI BOOT space overlapping.

15.2.1 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

15.2.2 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

15.2.3 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

15.2.4 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	3	4	16	32	8	64	128	256	512	1024	2048	4096

15.2.5 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.

15.2.6 Chip Select Control Register (SFC_SOFTCS)

Chinese name: SPI Flash Chip Select Control Register

Register bit width: [7: 0]

Offset: 0x05

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7: 4	csn	4	RW	csn pin output value
3: 0	csen	4	RW	When it is 1, the corresponding cs line is controlled by 7: 4 bits

15.2.7 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: 4	Reserved	4	RW	Keep
3	quad_io	1	RW	4-wire mode enabled, 1 valid
2	tFast	1	RW	The shortest invalid time of the chip select signal of SPI Flash, divided by frequency Clock period T calculation
1: 0	tCSH	2	RW	00: 1T 01: 2T 10: 4T 11: 8T

15.2.8 Custom Control Register (CTRL)

Chinese name: SPI Flash custom control register

Register bit width: [7: 0]

Offset: 0x08

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7: 4	nbyte	4	RW	Number of bytes transferred at a time
3: 2	reserve	2	RW	Keep
1	nbmode	1	RW	Multi-byte transfer mode
0	start	1	RW	Start multi-byte transfer, clear automatically after completion

15.2.9 Custom Command Register (CMD)

Chinese name: SPI Flash custom command register

Register bit width: [7: 0]

Offset: 0x09

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7: 0	cmd	8	RW	Set the command sent to spi flash

15.2.10 User -defined data register 0 (BUF0)

Chinese name: SPI Flash custom data register 0

Register bit width: [7: 0]

Offset: 0x0a

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7: 0	buf0	8	RW	When sending a write command to SPI, this register configures the first Bytes of data; this register is sent when a read command is sent to the SPI Store the first data read back.

15.2.11 User -defined data register 1 (BUF1)

Chinese name: SPI Flash custom data register 1

Register bit width: [7: 0]

Offset: 0x0b

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
				When sending a write command to SPI, this register configures the second

7: 0	buf1	8	RW	Bytes of data; this register is sent when a read command is sent to the SPI Store the second read data.

15.2.12 Custom Timing Register 0 (TIMER0)

Chinese name: SPI Flash custom timing register 0

Register bit width: [7: 0]

Offset: 0x0c

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7: 0	time0	8	RW	Lower 8 bits of time value required for custom command

15.2.13 Custom Timing Register 1 (TIMER1)

Chinese name: SPI Flash custom timing register 1

Register bit width: [7: 0]

Offset: 0x0d

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7: 0	time1	8	RW	The middle 8 bits of the time value of the custom command

15.2.14 Custom Timing Register 2 (TIMER2)

Chinese name: SPI Flash custom timing register 2

Register bit width: [7: 0]

Offset: 0x0e

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7: 0	time2	8	RW	The upper 8 bits of the time value required by the custom command

15.2.15 SPI two-wire four-wire user guide

In addition to the traditional single-wire mode, the SPI controller also supports two modes (dual mode) and quad mode (quad mode)

The working mode is started from SPI flash. By setting the dual_io register, the SPI controller can enter the two-wire mode, set

Setting the quad_io register allows the SPI controller to enter four-wire mode. Can be added in the first few instructions of the BIOS code

For the configuration codes of these two registers, after the configuration is completed, the controller fetches instructions according to the corresponding working mode of the config

Can increase the boot speed.

It should be noted that some SPI FLASH does not enable four-wire mode by default, or need to be configured in four-wire mode

Parameters related to timing (such as dummy clocks). In order to increase the applicability of SPI controller to various FLASH, this control

The device adds a custom register (0x8-0xe). The specific method of use is:

1. Set the custom command register (CMD) (0x9), which is a command sent to SPI FLASH;

2. If SPI FLASH requires the command sent this time to complete after a period of time, then configure the waiting time

Go to the custom timing register TIMER0-TIMER2 (0xc-0xe), otherwise these registers keep the default value of 0;

3. If you write configuration information to SPI FLASH, you need to write the configuration information to the custom data register

BUF0-BUF1 (0xa-0xb); if configuration information is read to SPI FLASH, the two registers are stored and read back

Value of

4. Configure the custom control register CTRL [7: 1] where CTRL [1] (nbmode) represents the multi-byte transmission mode

The number of bytes transferred this time is given by CTRL [7: 4] (nbyte);

5. Configure the custom control register CTRL [0] to start this transfer.

Generally speaking, the register to be configured is located in the non-volatile storage area of FLASH, so the above configuration only needs to be configured

Set once.

15.3 I2C controller

This chapter gives a detailed description and configuration of I2C. This system chip integrates I2C interface, mainly used to realize two

Data exchange between devices. I2C bus is a serial bus composed of data line SDA and clock SCL, which can send and receive

data. Two-way transmission between devices, the maximum transmission rate of 400kbps.

The I2C controller integrated in Loongson 3A4000 can be used as a master device or a slave device, these two modes

Switch between by configuring internal registers. When used as a slave device, it is only used to read the internal temperature of the chip.

The address is specified by the register SLV_CTRL [6: 0].

The base address of the physical address of the I2C0 controller register is 0x1FE00120.

The base address of the physical address of the I2C1 controller register is 0x1FE00130.

The specific internal registers are described below.

15.3.1 Divider Latch Low Byte Register (PRERlo)

Chinese name: Low-frequency register of frequency-dividing latch

Register bit width: [7: 0]

Offset: 0x00

Reset value: 0xff

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

15.3.2 divisor latch high byte register (PRERhi)

Chinese name: high-frequency register of frequency division latch

Register bit width: [7: 0]

164

Offset: 0x01

Reset value: 0xff

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

Assume that the value of the frequency division latch is prescale The frequency of the clock input from the LPB bus PCLK is clock_a SCL total

If the output frequency of the line is clock_s, the following relationship should be satisfied

Prcescale = clock_a / (4 * clock_s) -1

15.3.3 Control Register (CTR)

Chinese name: control register

Register bit width: [7: 0]

Offset: 0x02

Reset value: 0x20

Bit field	Bit field name	Bit width	access	description
7	EN	1	RW	Module work enable bit Is 1 normal working mode, 0 Operate the divider register
6	IEN	1	RW	Interrupt enable bit is 1 to open the interrupt
5	MST_EN	1	RW	Module master-slave selection 0: slave mode 1: master mode
4: 0	Reserved	5	RW	Keep

15.3.4 Transmit Data Register (TXR)

Chinese name: send register

Register bit width: [7: 0]

Offset: 0x03

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7: 1	DATA	7	W	Store the next byte to be sent
0	DRW	1	W	When data is transferred, this bit saves the lowest bit of the data When the address is transferred, this bit indicates the read and write status

165

15.3.5 Receive Data Register (RXR)

Chinese name: Receive Register

Register bit width: [7: 0]

Offset: 0x03

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7: 0	RXR	8	R	Store the last byte received

15.3.6 Command Control Register (CR)

Chinese name: command register

Register bit width: [7: 0]

Offset: 0x04

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7	STA	1	W	Generate START signal
6	STO	1	W	Generate STOP signal
5	RD	1	W	Generate read signal
4	WR	1	W	Write signal
3	ACK	1	W	Response signal
2: 1	Reserved	2	W	Keep
0	IACK	1	W	Generate interrupt acknowledge signal

The hardware is automatically cleared after I2C sends data. Always read '0' bit 3 as 1 when reading these bits

The time indicates that the controller will not send ack at the end of this transmission, otherwise it will send ack at the end.

15.3.7 Status Register (SR)

Chinese name: Status Register

Register bit width: [7: 0]

Offset: 0x04

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7	RxACK	1	R	Response bit received 1 No response bit received

166

				0 response bit received
6	Busy	1	R	I2c bus busy flag 1 Bus is busy 0 Bus is idle
5	AL	1	R	This bit is set when the I2C core loses control of the I2C bus
4: 2	Reserved	3	R	Keep
1	TIP	1	R	Indicate the transmission process 1 means data is being transferred 0 means the data transmission is completed
0	IF	1	R	Interrupt flag, one data transfer is completed, or another device

Initiate data transfer, position 1

15.3.8 Slave Control Register (SLV_CTRL)

Chinese name: slave device control register

Register bit width: [7: 0]

Offset: 0x07

Reset value: 0x00

Bit field	Bit field name	Bit width	access	description
7	SLV_EN	1	WR	Slave mode enable, effective when MST_EN is 0, can be used for Reset slave internal logic
6: 0	SLV_ADDR	7	WR	Slave mode I2C address, configurable through software

16 3A3000 kernel compatibility

In order to achieve backward compatibility of the Linux kernel starting from 3A3000, the existing kernel must be implemented according to the chip's implementation regulati

Fan made some modifications.

In order to achieve a 3A3000 compatible core, the 3A4000 chip, in addition to implementing a set of configuration methods according to the new specification,

It also needs to support the mechanisms that are widely used in the current kernel.

The following introduces the 3A4000 core from two aspects: kernel compatibility and new feature support.

16.1 Compatible with 3A3000 core

In order to be compatible with the 3A3000 core, the following parts in the core must be modified.

16.1.1 Processor characteristic recognition method

For MIPS processors, there is no common way to identify the different characteristics of the processor in the kernel,

Instead, the processor model is distinguished by PRID, and then different processing is performed in different situations according to the processor model.

Because the current kernel only judges and handles the existing processor models, there is no new processor that has not been implemented yet.

There is a default processing method, which results in a lot of low-level code that does not have a corresponding implementation when running on a new processor.

In order to solve this problem, starting from 3A4000, a set of processor configuration instructions and processor feature recognition are implemented

Instructions to standardize software and hardware interfaces. Can be accessed through processor configuration instructions, or through 0x3ff00000

To access. The register is written as CSR [Offset Address] [Bit].

This register identifies some software-related processor features for software to view before enabling specific functions. Registered

Offset address 0x0008. Record as CSR [0x08].

Table 16- 1 Chip feature register

Bit field	Field name	access	Reset value	description
0	Centigrade	R	1'b1	CSR [0x428] valid
1	Node counter	R	1'b1	CSR [0x408] valid
2	MSI	R	1'b1	MSI available
3	EXT_IOI	R	1'b1	EXT_IOI available
4	IPI_percore	R	1'b1	IPI sending via CSR private address

5	Freq_percore	R	1'b1	Adjust frequency by CSR private address
6	Freq_scale	R	1'b0	Dynamic crossover function is available
7	DVFS_v1	R	1'b0	Dynamic FM v1 is available
8	Tsensor	R	1'b0	Temperature sensor available

16.1.2 Current kernel modification method

In the current 3.10 kernel, there are six codes that use PRID for functional feature identification.

Support, you need to modify five of them.

The five functions are as follows:

function	path	description
cpu_probe_loongson	arch / mips / kernel / cpu-probe.c	Identify the chip model
loonson_cpu_temp	driver / platform / mips / cpu_hwmon.c	Read on-chip temperature sensor
play_dead	arch / mips / loongson / loongson-3 / smp.c	Dynamic switch core support
init_node_counter_clocksource	arch / mips / loongson / loongson-3 / node_counter.c	enable on-chip clock source
ls7a_init_irq	arch / mips / loongson / loongson-3 / ls7a-irq.c	Enable MSI interrupt

(1) cpu_probe_loongson

This function is used to identify the chip model. It is necessary to add the processor configuration instruction identification in the original default condition.

The code of the name of the other manufacturer, the name of the chip and the assignment of the corresponding data structure The corresponding registers are as follows.

Manufacturer name register. CSR [0x0010].

Table 16- 2 Manufacturer Name Register

Bit field	Field name	access	Reset value	description
63: 0	Vendor	R	0x6e6f7367_6e6f6f4c	the string "Loongson"

Chip name register. CSR [0x0020].

Table 16- 3 Chip Name Register

Bit field	Field name	access	Reset value	description
63: 0	ID	R	0x00003030_30344133	character string "3A4000"

(2) loongson_cpu_temp

This function is used to read the on-chip temperature sensor, and new processing needs to be added to the original default condition. First root
According to CSR [0x8] [0], determine whether there is an on-chip temperature sensor and decide whether to use the processor configuration instruction to read the on-chip tempera
Register CSR [0x428].

169

(3) play_dead

This function is used to dynamically switch cores. For future processors, this function needs to be significantly changed. original
The function calls different functions according to the PRID to perform the targeted Cache operation and check the core. It needs to be changed to MIPS
The specification of the second volume of the manual, the number and size of ICACHE / DCACHE / VCACHE read from the relevant CP0 register
Configuration, perform the corresponding flash cache operation, and then close the core. It should be noted that in all Loongson 2 and Loongson 3 series
In the column processor, when reading the cache configuration information through the CP0 register, the Secondary Cache in CP0.config2
Refers to the on-chip last-level cache or scache; when there is an on-chip private second-level cache, through CP0.config2
Tertiary Cache said. When closing the core, you need to decide whether to use CSR [0x1050] [3] based on whether CSR [0x420] [23] is 1
Whether to close the core or use the corresponding bit of 0x3ff001d0 to close the core.

(4) init_node_counter_clocksource

This function is used to initialize the on-chip clock source. A default condition needs to be added. According to CSR [0x8] [1], determine
Is there a node_counter. At the same time, a parameter needs to be added, and certain specific values are no longer corrected.

(5) ls7a_init_irq

This function is used to decide whether to use MSI interrupt. A default condition needs to be added.
In addition, according to CSR [0x8] [2], determine whether there is MSI support.

16.2 New feature support

In order to use the new features provided by the 3A4000 processor in the core, it can be identified or enabled according to the following method.
Only the parts that can improve system performance are introduced here, and new mechanisms such as sending inter-core interrupts through CSR instructions,
Because it is compatible with the requirements of the 3A3000 processor, it must actually use the existing specific address access method, there is no need to special
It is intended to support CSR, but increase software overhead.

16.2.1 Identification of processor characteristics

New features are identified through CSR register instructions, in order to support the new features, you need to consider not supporting the processor

170

Page 191

The processing method of the old processor with configuration instructions. One is to increase the judgment of PRID at each position that needs to be characterized,

All existing PRIDs are processed. The second is to add exception instruction processing. When the reserved instruction is recognized as a processor configuration instruction

When, according to the instruction content, according to the PRID, construct the correct return value.

16.2.2 Extended interrupt mode

In order to enable the extended interrupt mode in the kernel, it needs to be set in the following order.

1) Extended interrupt mode supports identification by CSR [0x8] [3].

2) PMON needs to configure the external interrupt conversion register of the HT controller that expects to support the extended interrupt mode to be positive

Exact value. The register is defined as follows, set to the following values:

INT_trans_en = 0 // Enable control using CSR register, CSR [0x420] [48] and this register can enable

The interrupt mode can be extended. This mode is not enabled by default in PMON, and is enabled by the kernel configuration CSR [0x420] [48]

INT_trans_allow = 1 // Allow to enable interrupt conversion function externally

INT_trans_addr = 0x1000000001140 // Extended interrupt register address, see 14.3.3.

INT_trans_cache = 0 // Uncache way

Offset: 0x270

Reset value: 0x00000000

name: HT RX INT TRANS Lo

Table 16- 4 HT RX INT TRANS LO

Bit field	Bit field name	Bit width	Reset value	access	description
31: 4	INT_trans_addr [31: 4]	28	0x0	R / W	Interrupt conversion address low
3: 0	Reserved	4	0x0	R	Keep

Offset: 0x274

Reset value: 0x00000000

name: HT RX INT TRANS Hi

Table 16- 5 HT RX INT TRANS Hi

Bit field	Bit field name	Bit width	Reset value	access	description

171

Page 192

31	INT_trans_en	1	0x0	R / W	Interrupt conversion enable
30	INT_trans_allow	1	0x0	R / W	Interrupt conversion is allowed
29:26	INT_trans_cache	4	0x0	R / W	Interrupt conversion of Cache field
25: 0	INT_trans_addr [58:32]	26	0x0	R / W	Interrupt conversion address high

3) The kernel first recognizes extended interrupt mode support through CSR [0x8] [3], and then through register CSR [0x420] [48]

Enable extended interrupt mode. The base address is 0x1fe00000 and the offset address is 0x0420.

Table 16-6 Other function setting registers

Bit field	Field name	access	Reset value	description
48	EXT_INT_en	RW	0x0	Extended IO interrupt enable

4) Set the corresponding routing and internal control of the extended interrupt mode.

16.3 Configuration register instruction debugging support

In principle, the configuration register instruction is not accessed across the chip when it is used, but in order to meet the needs for debugging and other functions, we use here Supports cross-chip access with multiple register addresses. It is worth noting that such registers can only be written, not read.

In addition to the original inter-core interrupts and other registers that can be accessed across the chip, all such registers and addresses are as follows.

Table 16- 7 Communication Register between Processor Cores

name	Offset address	Authority	description
IPI_Send	0x1040	WO	32-bit interrupt distribution register [31] Wait for completion flag, set to 1 to wait for interrupt to take effect [30:26] Reserved [25:16] processor core number [15: 5] reserved [4: 0] Interrupt vector number, corresponding to the vector in IPI_Status
Mail_Send	0x1048	WO	64-bit MailBox cache register [63:32] MailBox data [31] Wait for completion flag, when set to 1, it will wait for write to take effect [30:27] Mask for writing data, each bit represents 32-bit write data The corresponding byte will not be written to the target address, such as 1000b Write 0-2 bytes, 0000b then write all 0-3 bytes [26] Reserved [25:16] processor core number

			[15: 5] reserved [4: 2] MailBox number 0-MailBox0 lower 32 bits 1-MailBox0 high 32 bits 2-MailBox1 lower 32 bits 3-MailBox1 high 32 bits 4-MailBox2 lower 32 bits 5-MailBox2 high 32 bits 6-MailBox3 lower 32 bits 7-MailBox4 high 32 bits [1: 0] Reserved
FREQ_Send	0x1058	WO	32-bit frequency enable register [31] Wait for completion flag, set to 1 to wait for interrupt to take effect [30:26] Reserved [25:16] processor core number [15: 5] reserved [4: 0] Write to the corresponding processor core private frequency configuration register. CSR [0x1050]
ANY_Send	0x1158	WO	64-bit register access register

[63:32] Write data

[31] Wait for completion flag, set to 1 to wait for interrupt to take effect

[30:27] Mask for writing data, each bit represents 32-bit write data

The corresponding byte will not be written to the target address, such as 1000b

Write 0-2 bytes, 0000b then write all 0-3 bytes

[26] Reserved [25:16] Target processor core number

[15: 0] Register offset address written

173