

?

Loongson 3a2000/3b2000 processor user manual

Part i

# Multicore processor architecture, register description and system software programming guide

**V1.7**

***02, 2017***

***Loongson technology co. LTD***



**Copyright statement**

The copyright of this document belongs to loongson technology co., LTD. All rights reserved. No company or individual may make public, reprint or otherwise distribute any portion of this document to a third party without written permission. Otherwise, will pursue its legal responsibility certainly.

**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 prior notice. The company shall not be liable for any direct or indirect loss caused by improper use of the documents.

**Loongson technology co. LTD**

Loongson Technology Corporation Limited

Address: Loongson Industrial Park, Building No. 2, longson Industrial Park, zhongguancun environmental science and technology demonstration Park, haidian district, Beijing

Zhongguancun Environmental Protection Park, Haidian District, Beijing

Tel: 010-62546668 Fax: 010-62600826

### Reading guide

Loongson 3a2000/3b2000 processor user manual is divided into the first volume and the second volume.

The first part introduces the architecture and register description of the longson 3a2000/3b2000 multi-core processor, and gives a detailed description of the chip system architecture, the functions and configuration of the main modules, the register list and the bit domain.

Loongson 3a2000/3b2000 processor user manual, the second volume, introduces loongson in detail from the perspective of system software developers

3a2000/3b2000 USES GS464e high performance processor core.

# Revision history

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Document update record** | | | The document name: | | Loongson 3a2000/3b2000 processor user manual  - the days |
| The version number | | V1.7 |
| The founders: | | Chip r&d department |
| Date of creation: | | 2017-02-14 |
| **Update history** | | | | | |
| **The serial number** | **Updated date** | **The version number** | | **Update the content** | |
| 1 | 2015-03-31 | V1.0 | | The initial release | |
| 2 | 2015-05-27 | V1.1 | | Add a description of hardware and software changes | |
| 3 | 2015-07-17 | V1.2 | | Supplementary register description | |
| 4 | 2016-02-23 | V1.3 | | Fixed some register description errors, modified "software and hardware modification instructions" | |
| 5 | 2016-04-13 | V1.4 | | Fix the HT receive window description, modify "software and hardware change description" | |
| 6 | 2016-06-21 | V1.5 | | Fix CLKSEL[15] description | |
| 7 | 2016-10-10 | V1.6 | | Added software configuration HT frequency description | |
| 8 | 2017-02-14 | V1.7 | | Modify the "hardware and software change specification" and add the SPI address space specification | |
| 9 |  |  | |  | |

Manual information feedback: service@loongson.cn

Problem feedback web site, http://bugs.loongnix.org/, also can be submitted to our chip problem in the process of product use, and obtain technical support.

目录

[Multicore processor architecture, register description and system software programming guide 1](#_Toc42179697)

[Reading guide 3](#_Toc42179698)

[Revision history 4](#_Toc42179699)

[1 An overview of the 11](#_Toc42179700)

[2 System configuration and control 12](#_Toc42179701)

[3 GS464e processor core 12](#_Toc42179702)

[4 Shared Cache (SCache) 12](#_Toc42179703)

[5 Matrix processing accelerator 12](#_Toc42179704)

[6 Interrupt and communication between processor cores 12](#_Toc42179705)

[7 I/O interrupt 12](#_Toc42179706)

[8 Temperature sensor 12](#_Toc42179707)

[9 Ddr2/3 SDRAM controller configuration 12](#_Toc42179708)

[9.3 Ddr2/3 SDRAM write operation protocol 12](#_Toc42179709)

[10 HyperTransport controller 12](#_Toc42179710)

[10.5.1 Bridge Control 12](#_Toc42179711)

[10.5.2 Capability Registers 12](#_Toc42179712)

[10.5.3 Custom register 12](#_Toc42179713)

[10.5.4 Receive diagnostic register 12](#_Toc42179714)

[10.5.5 Interrupt routing mode selection register 12](#_Toc42179715)

[10.5.6 Receive buffer initial register 12](#_Toc42179716)

[10.5.7 Receive address window configuration register 12](#_Toc42179717)

[10.5.8 Interrupt vector register 12](#_Toc42179718)

[10.5.9 Interrupt enable register 12](#_Toc42179719)

[10.5.10 Interrupt Discovery & Configuration 12](#_Toc42179720)

[10.5.11 Configure the register in the POST address window 12](#_Toc42179721)

[10.5.12 Pre-fetch address window configuration register 12](#_Toc42179722)

[10.5.13 UNCACHE address window configuration register 12](#_Toc42179723)

[10.5.14 P2P address window configuration register 12](#_Toc42179724)

[10.5.15 The command sends the cache size register 12](#_Toc42179725)

[10.5.16 Data send cache size register 12](#_Toc42179726)

[10.5.17 Send the cache debug register 12](#_Toc42179727)

[10.5.18 PHY impedance matching control register 12](#_Toc42179728)

[10.5.19 Revision ID register 12](#_Toc42179729)

[10.5.20 Error Retry controls the register 12](#_Toc42179730)

[10.5.21 The Retry Count register 12](#_Toc42179731)

[10.5.22 Link Train register 12](#_Toc42179732)

[10.5.23 Training 0 timeout short timing register 12](#_Toc42179733)

[10.5.24 Training 0 timeout long timing register 12](#_Toc42179734)

[10.5.25 Training 1 counting register 12](#_Toc42179735)

[10.5.26 Training 2 counting register 12](#_Toc42179736)

[10.5.27 Training 3 counting register 12](#_Toc42179737)

[10.5.28 Software frequency configuration register 12](#_Toc42179738)

[10.5.29 PHY configuration register 12](#_Toc42179739)

[10.5.30 The link initializes the debug register 12](#_Toc42179740)

[10.5.31 LDT debug register 12](#_Toc42179741)

[11 Low speed IO controller configuration 12](#_Toc42179742)

[11.3.2 Interrupt enabled register (IER) 12](#_Toc42179743)

[11.3.3 Interrupt identification register (IIR) 12](#_Toc42179744)

[11.3.4 FIFO control register (FCR) 12](#_Toc42179745)

[11.3.5 Line control register (LCR) 12](#_Toc42179746)

[11.3.6 MODEM control register (MCR) 12](#_Toc42179747)

[11.3.7 Line status register (LSR) 12](#_Toc42179748)

[11.3.8 MODEM status register (MSR) 12](#_Toc42179749)

[11.3.9 Frequency divider latch 12](#_Toc42179750)

[11.4.1 Control register (SPCR) 12](#_Toc42179751)

[11.4.2 Status register (SPSR) 12](#_Toc42179752)

[11.4.3 Data register (TxFIFO) 12](#_Toc42179753)

[11.4.4 External register (SPER) 12](#_Toc42179754)

[11.4.5 Parameter control register (SFC\_PARAM) 12](#_Toc42179755)

[11.4.6 Chip selection control register (SFC\_SOFTCS) 12](#_Toc42179756)

[11.4.7 Timing control register (SFC\_TIMING) 12](#_Toc42179757)

[12 List of chip configuration registers 12](#_Toc42179758)

[13 Hardware and software design guide 12](#_Toc42179759)

Figure orders to record

[Figure 1-1 longson 3 system structure](#_bookmark4) 11

[Figure 1-2 structure of longson no3 node](#_bookmark5) 12

[Figure 1-3 loong chip 3A2000 chip structure](#_bookmark7) 13

[Figure 3-1 GS464e structure diagram](#_bookmark32) 31

[Figure 7-1 loongson 3A2000 processor interrupt routing schematic diagram 40](#_bookmark47)

[Figure 9-1 DDR2 SDRAM read operation protocol](#_bookmark62) 47

[Figure 9-2 DDR2 SDRAM write operation protocol](#_bookmark64) 47

[Figure 10-1 configuration access 97 of HT protocol in loongson 3A2000](#_bookmark198)

[FIG. 10-2 interconnection structure of four-piece longson no. 3 98](#_bookmark201)

[Figure 10-3 8-bit interconnect structure of two pieces of loong core no. 3 99](#_bookmark202)

[Figure 10-4 16-bit interconnect structure of two pieces of loong core no. 3 99](#_bookmark203)

[Figure 11-1 configures read and write bus address generation 104](#_bookmark208)

V

Table item record

[Table 2-1 control pin](#_bookmark12) 16

[Table 2-2 node-level system global address distribution](#_bookmark15) 18

[Table 2-3 address distribution within nodes](#_bookmark16) 19

[Table 2-4 address distribution within nodes](#_bookmark17) 19

[Table 2-5 the MMAP field corresponds to the spatial access property](#_bookmark19) 20

[Table 2-6 level 1 cross switch address window register table](#_bookmark20) 20

[Table 2-7 at level 2 XBAR, from the corresponding relationship between the device number and the module](#_bookmark21) 23

[Table 2-8 the MMAP field corresponds to the spatial access property 23](#_bookmark22)

[Table 2-9 XBAR address window translation register table](#_bookmark23) 23

[Table 2-10 XBAR level 2 default address configuration](#_bookmark24) 25

[Table 2-11 chip configuration register (physical address 0x1fe00180)](#_bookmark26) 26

[Table 2-12 chip sampling register (physical address 0x1fe00190)](#_bookmark27) 26

[Table 2-13 chip node and processor core software frequency doubling setting register (physical address 0x1fe001b0)](#_bookmark28) 27

[Table 2-14 chip memory and HT clock software frequency doubling setting register (physical address 0x1fe001c0)](#_bookmark29) 28

[Table 2-15 chip processor core software frequency division setting register (physical address 0x1fe001d0)](#_bookmark30) 29

[Table 4-1 Shared Cache lock window register configuration](#_bookmark34) 33

[Table 5-1 matrix processing programming interface notes](#_bookmark36) 34

[Table 5-2 matrix processing register address](#_bookmark37) 35

[Table 5-3 trans\_ctrl register description](#_bookmark38) 35

[Table 5-4 shows the trans\_status register](#_bookmark39) 36

[Table 6-1 registers associated with interrupts between processor cores and their functional descriptions](#_bookmark41) 37

[Table 6-20 intercore interrupt and communication registers for processor no. 2](#_bookmark42)

[Table 6-3 intercore interrupt and communication registers for no. 1 processor core listing](#_bookmark43) 38

[Table 6-4 intercore interrupt and communication registers for processor core no. 2, listing](#_bookmark44) 38

[Table 6-5 intercore interrupt and communication registers for no. 3 processor core listing](#_bookmark45) 38

[Table 7-1 interrupt control register](#_bookmark48) 41

[Table 7-2 IO control register address](#_bookmark49) 41

[Table 7-3 notes to the interrupt routing register](#_bookmark50) 42

[Table 7-4 interrupt routing register address](#_bookmark51) 42

[Table 8-1 temperature sampling register description](#_bookmark54) 43

[Table 8-2 high and low temperature interrupt registers](#_bookmark56) 44

[Table 8-3 high temperature frequency down control register](#_bookmark58) 45

[Table 10-1 HyperTransport bus-related pin signals](#_bookmark78) 58

[Table 10-2 HyperTransport receiver can receive commands](#_bookmark80) 61

[Table 10-3 commands sent out in both modes](#_bookmark81) 61

[Table 10-4 the default address window distribution of the four HyperTransport interfaces is](#_bookmark85) 62

[Table 10-5 address window distribution inside the HyperTransport interface of loongson 3 processor](#_bookmark86) 63

[Table 10-6 address window 63 provided in the loongson 3A2000 processor HyperTransport interface](#_bookmark88)

[Table 10-7 software visible register list](#_bookmark90) 64

[Table 10-8 Bus Reset Control register definition](#_bookmark92) 66

[Table 10-9 Command, Capabilities Pointer, Capability ID register definition 66](#_bookmark94)

[Table 10-10 Link Config, Link Control register definition](#_bookmark95) 67

[Table 10-11 Revision ID, Link Freq, Link Error, Link Freq Cap register definition 68](#_bookmark96)

[Table 10-12 Feature Capability register definition](#_bookmark97) 69

[Table 10-13 MISC register definition](#_bookmark99) 69

[Table 10-14 receive diagnostic register](#_bookmark101) 70

[Table 10-15 interrupt routing selection register](#_bookmark103) 71

[Table 10-16 receive buffer initial register](#_bookmark105) 71

[Table 10-17 HT bus receive address window 0 enable (external access) register definition](#_bookmark107) 72

[Table 10-18 HT bus receive address window 0 base address (external access) register definition](#_bookmark108) 72

[Table 10-19 HT bus receive address window 1 enable (external access) register definition](#_bookmark109) 72

[Table 10-20 HT bus receive address window 1 base address (external access) register definition](#_bookmark110) 73

[Table 10-21 HT bus receive address window 2 enable (external access) register definition](#_bookmark111) 73

[Table 10-22 HT bus receive address window 2 base address (external access) register definition](#_bookmark112) 73

[Table 10-23 HT bus receive address window 3 enable (external access) register definition](#_bookmark113) 73

[Table 10-24 HT bus receive address window 3 base address (external access) register definition](#_bookmark114) 74

[Table 10-25 HT bus receive address window 4 enable (external access) register definition](#_bookmark115) 74

[Table 10-26 HT bus receive address window 4 base address (external access) register definition](#_bookmark116) 74

[Table 10-27 HT bus interrupt vector register definition (1)](#_bookmark118) 75

[Table 10-28 HT bus interrupt vector register definition (2)](#_bookmark119) 76

[Table 10-29 HT bus interrupt vector register definition (3)](#_bookmark120) 76

[Table 10-30 HT bus interrupt vector register definition (4)](#_bookmark121) 76

[Table 10-31 HT bus interrupt vector register definition (6)](#_bookmark122) 77

[Table 10-32 HT bus interrupt vector register definition (7)](#_bookmark123) 77

[Table 10-33 HT bus interrupt vector register definition (8)](#_bookmark124) 77

[Table 10-34 HT bus interrupt enable register definition (1)](#_bookmark126) 78

[Table 10-35 HT bus interrupt enable register definition (2)](#_bookmark127) 78

[Table 10-36 HT bus interrupt enable register definition (3)](#_bookmark128) 79

[Table 10-37 HT bus interrupt enable register definition (4)](#_bookmark129) 79

[Table 10-38 HT bus interrupt enable register definition (5)](#_bookmark130) 79

[Table 10-39 HT bus interrupt enable register definition (6)](#_bookmark131) 79

[Table 10-40 HT bus interrupt enable register definition (7)](#_bookmark132) 79

[Table 10-41 HT bus interrupt enable register definition (8)](#_bookmark133) 80

[Table 10-42 definition 80 of the Interrupt Capability register](#_bookmark135)

[Table 10-43 Dataport register definition](#_bookmark136) 80

[Table 10-44 IntrInfo register definition (1)](#_bookmark137) 80

[Table 10-45 IntrInfo register definitions (2)](#_bookmark138) 81

[Table 10-46 HT bus POST address window 0 enable (internal access) 81](#_bookmark140)

[Table 10-47 HT bus POST address window 0 base address (internal access) 82](#_bookmark141)

[Table 10-48 HT bus POST address window 1 enable (internal access) 82](#_bookmark142)

[Table 10-49 HT bus POST address window 1 base address (internal access) 82](#_bookmark143)

[Table 10-50 HT bus prefetch address window 0 enable (internal access) 83](#_bookmark145)

[Table 10-51 HT bus can prefetch address window 0 base address (internal access) 83](#_bookmark146)

[Table 10-52 HT bus prefetch address window 1 enable (internal access) 83](#_bookmark147)

[Table 10-53 HT bus can prefetch address window 1 base address (internal access) 83](#_bookmark148)

[Table 10-54 HT bus Uncache address window 0 enable (internal access) 84](#_bookmark150)

[Table 10-55 HT bus Uncache address window 0 base address (internal access) 84](#_bookmark151)

[Table 10-56 HT bus Uncache address window 1 enable (internal access) 85](#_bookmark152)

[Table 10-57 HT bus Uncache address window 1 base address (internal access) 85](#_bookmark153)

[Table 10-58 HT bus Uncache address window 2 enable (internal access) 85](#_bookmark154)

[Table 10-59 HT bus Uncache address window 2 base address (internal access) 85](#_bookmark155)

[Table 10-60 HT bus Uncache address window 3 enable (internal access) 86](#_bookmark156)

[Table 10-61 HT bus Uncache address window 3 base address (internal access) 86](#_bookmark157)

[Table 10-62 HT bus P2P address window 0 enable (external access) register definition](#_bookmark159) 87

[Table 10-63 HT bus P2P address window 0 base address (external access) register definition](#_bookmark160) 87

[Table 10-64 HT bus P2P address window 1 enable (external access) register definition](#_bookmark161) 87

[Table 10-65 HT bus P2P address window 1 base address (external access) register definition](#_bookmark162) 87

[Table 10-66 sends the cache size register](#_bookmark164) 88

[Table 10-67 data sending cache size register](#_bookmark166) 88

[Table 10-68 sends cache debug register](#_bookmark168) 89

[Table 10-69 impedance matching control register](#_bookmark170) 89

[Table 10-70 Revision ID register](#_bookmark172) 90

[Table 10-71 Error Retry control register](#_bookmark174) 90

[Table 10-72 Retry Count register](#_bookmark176) 91

[Table 10-73 Link Train register](#_bookmark178) 91

[Table 10-74 Training 0 timeout short timing register](#_bookmark180) 92

[Table 10-75 Training 0 timeout long count register](#_bookmark182) 92

[Table 10-76 Training 1 counting register](#_bookmark184) 93

[Table 10-77 Training 2 counting register](#_bookmark186) 93

[Table 10-78 Training 3 counting register](#_bookmark188) 93

[Table 10-79 software frequency configuration register](#_bookmark190) 94

[Table 10-80 PHY configuration register](#_bookmark192) 95

[Table 10-81 link initialization debug register](#_bookmark194) 96

[Table 10-82 LDT debug register](#_bookmark196) 96

[Table 11-1 PCI controller configuration header](#_bookmark206) 100

[Table 11-2 PCI control register](#_bookmark207) 101

[Table 11-3 PCI/PCIX bus request and reply line allocation](#_bookmark209) 104

[Table 11-4 address space distribution of LPC controller](#_bookmark211) 105

[Table 11-5 LPC configuration register meaning](#_bookmark212) 105

[Table 11-6 SPI controller address space distribution](#_bookmark224) 113

[Table 11-6 IO control register](#_bookmark233) 117

## An overview of the

#### Introduction to loongson series processors

The loongson processor mainly includes three series. Loongson 1 processor and its IP series are mainly for embedded applications, loongson 2 superstandard processor and its IP series are mainly for desktop applications, and loongson 3 multi-core processor series are mainly for server and high-performance machine applications. According to the application needs, part of the loongson 2 can also be oriented to part of the high-end embedded should With, part of low - end loongson 3 can also face part of the desktop application. The three series will develop in parallel.

Based on the scalable multi-core interconnect architecture, the loong chip 3 multi-core series integrates multiple high-performance processor cores and a large number of level-2 caches on a single chip, and interconnects multiple chips through high-speed I/O interfaces to form a larger system.

The retractable interconnection structure adopted by loongson 3 is shown in figure 1-1 below.Loongson 3 and multi - chip system are adopted 2

D mesh interconnection structure, in which each node is composed of 8\*8 cross switches, each cross switch connects four processor cores and four Shared Cache, and interconnects with other nodes in the four directions of east (E), south (N), west (W) and north (N). Therefore, a 2\*2 mesh can connect 16 processor cores, and a 4\*4 mesh can connect 64 processor cores.



Loongson no. 3 node and 2d interconnect structure, (a) node structure, (b) 2\*2 mesh network connecting 16 processors, (c)

A 4\*4 mesh network connects 64 processors.

(c)

(b)

**L2 L2 L2 L2**

(a)

**E S W**

**N**

**8 by 8 switch**

**E S W**

**N**

**P0, P1 P2, P3**

Figure 1-1 longson 3 system structure

The node structure of loongson 3 is shown in figure 1-2 below. Each node has two AXI cross-switches connected to the processor and Shared

Cache, memory controller, and IO controller. The first level is a AXI cross Switch (called the X1 Switch, or X1) that connects the processor to the Shared Cache. The second level cross Switch (called the X2 Switch) connects the Shared Cache to the memory controller.



**The MC**

SS SM

MA0 MA1 MA2 MA3

**X2 Switch**

ES

WS

s5s6s7

s3s4

s0s1s2

**The X1 Switch**

m7

m4m5m6

The m3

m1m2

m0

WM

EM

**P0 P1 P2 P3**

NSNM

**Xconf**

**S3**

**S2**

**S1**

**S0**

Figure 1-2 structure of longson node3

At most 8\*8 X1 crossovers are connected to four GS464 processor cores via four Master ports at each node(P0, P1, P2, P3 in the figure), connect the four interleave Shared Cache blocks (S0, S1, S2, S3 in the figure) with four Slave ports, and connect other nodes or IO nodes (EM/ES, SM/SS, WM/WS, NM/NS in the figure) with four Master/Slave ports in the east, south, west, and north directions.

The X2 cross-switch connects four Shared caches through four Master ports, at least one Slave port to a memory controller, and at least one Slave port to a cross-switch configuration module (Xconf) to configure the local node's X1 and X2 address Windows. You can also connect more memory controllers and IO ports as needed.

#### 1.2 introduction of loongson 3a2000/3b2000

Loongson 3a2000/3b2000 is an upgraded version of the loongson 3A1000 quad-core processor. The package pin is compatible with loongson 3A1000. The godson 3a2000/3b2000 is a processor configured as a single node with 4 cores. It is manufactured by the 40nm process and its main operating frequency is 800MHz-1GHz. The main technical features are as follows:

* 4 64-bit quad-emission superscalar GS464e high-performance processor cores are integrated on the chip.
* 4 MB split Shared three-level Cache(composed of 4 individual modules, each with a capacity of 1MB);
* Maintain Cache consistency for multi-core and I/O DMA access through directory protocols;
* Two 64-bit ddr2/3 controllers with ECC and 667MHz are integrated on the chip.
* 3B2000 chips are integrated with 2 16-bit 1.6ghz HyperTransport controllers (HT);
* HT1 in 3A2000 is a 16-bit 1.6ghz HT controller, HT0 is not available;
* Each 16-bit HT port is split into two 8-way HT ports for use.
* Integrated 32-bit 33MHz PCI;
* In - chip integration 1 LPC, 2 UART, 1 SPI, 16 GPIO interface. Compared with loongson 3A1000, the main improvements are as follows:
* Processor core microstructure upgrade;
* Memory controller structure, frequency upgrade;
* The structure and frequency of HT controller were upgraded comprehensively.
* Comprehensive upgrade of internal interconnection structure;
* Comprehensive upgrade of external extended interconnection structure;
* Support SPI startup function;
* Support full chip software frequency configuration;
* Full - chip performance optimization.

The overall architecture of loongson 3A2000 is based on two-level interconnection, as shown in figure 1-3 below.

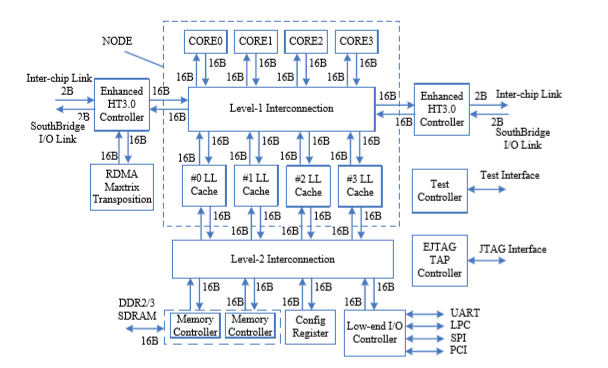


Figure 1-3 structure of loongson 3A2000 chip

Level 1 interconnect USES 6x6 cross-switches to connect four GS464e cores (as Master devices), four Shared Cache modules (as Slave devices), and two IO ports (one Master and one Slave for each port). Each IO port connected by the level 1 interconnect switch is connected to a 16-bit HT controller, and each 16-bit HT port can also be used as two 8-bit HT ports. The HT controller is connected to a level 1 interconnect switch by a DMA controller, which is responsible for the DMA control of IO and the maintenance of inter-chip consistency. The loongson 3's DMA controller can also be configured to prefetch and matrix transpose or shift.

The second level interconnection USES a 5x4 cross-switch to connect four Shared Cache modules (as the main device) and two ddr2/3

Memory controller, low speed and high speed I/O (including PCI, LPC, SPI, etc.) and the configuration register module inside the chip. The above two level interconnect switches all adopt the data channel separated by read and write. The data channel width is 128bit and works at the same place

The same frequency is used to provide high-speed on-chip data transmission.

The difference between 3B2000 and 3A2000 processors is that 3B2000 supports HT0 as the consistent interconnect interface. Based on the loongson 3 extensible interconnect architecture, 4 pieces of four-core loongson 3B2000 can be connected through HT port to form a NUMA structure with 16 cores on 4 chips. The 3A2000 processor only supports the IO use of the HT1 controller.

The following will not make a distinction between 3B2000 and 3A2000, referred to as longson 3A2000.

#### 1.3 loongson 3A2000 commercial and industrial grade chip description

Loongson 3A2000 chip has two kinds of industrial grade and commercial grade. Its main features are as follows:

|  |  |  |
| --- | --- | --- |
| configuration | Commercial grade | industrial-grade |
| Working temperature | 0 ℃ ~ 70 ℃ | - 40 ℃ ~ 85 ℃ |
| Whether screening | - | Square root |
| Quality consistency test | - | Square root |
| Quality consistency test standard | - | GB 4937-1995 |

As with most semiconductor devices, the failure rate of the loongson 3A chip conforms to the bathtub curve model. In order to ensure a more long-term, stable and reliable operation, and to adapt to the more stringent environmental temperature requirements, loongson 3A industrial-grade chip was selected for reliability, so as to eliminate the early failure of the chip. This reliability screening is 100% trial, through which the chips are screened for industrial-grade applications.

The operating voltage of 3A2000 commercial - and industrial-grade chips is slightly different. The working voltage of the industrial-grade chip is required to be 1.15v, and the jitter of the power supply is less than 50mV. Commercial - grade chips require working voltage of 1.25V, power jitter less than 50mV.

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

|  |  |  |
| --- | --- | --- |
| 2. Stable baking | 125 ℃ for 24 h | 100% |
| 3. The temperature changes rapidly | 10 cycles at maximum and minimum storage temperatures | 100% |
| 4. Serial number |  | 100% |
| 5, intermediate (before the old) electrical test | The normal temperature | 100% |
| 6, aging | TC = 85 ℃, 160 h | 100% |
| 7. Intermediate (after aging) electrical test | The normal temperature | 100% |
| 8. Percentage of defective products allowed (PDA)  To calculate | PDA≤5%, normal temperature, when 5% < PDA≤10%, can be heavy  New submissions, but only once | All batch |
| 9. Terminal electrical test | Three temperature, record all test data | 100% |
| 10. External visual inspection | Clear identification, no contamination, no oxidation of welding ball, chip intact | 100% |

|  |  |  |
| --- | --- | --- |
| Screening program | Methods and conditions (summary) | requirements |
| 1, visual inspection | Clear identification, no contamination, no oxidation of welding ball, chip intact | 100% |

## System configuration and control

#### Chip working mode

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

* + - Single chip mode. The system consists of only one loongson 3A2000, which is a symmetric multiprocessor system (SMP).
    - Multi-chip interconnection model. The system contains 2 or 4 pieces of loongson 3A2000, which are interlinked through the HT port of loongson 3A2000. It is a non-uniform memory multi-processor system (cc-numa).
    - Large-scale interconnection model. A large scale non-uniform memory access multiprocessor system (cc-numa) is constructed by means of a special extension bridge.

#### Control pin instructions

The main control pins include DO\_TEST, ICCC\_EN, NODE\_ID[1:0], CLKSEL[15:0], and PCI\_CONFIG.

Table 2-1 control pin description

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| signal | Pull up and down | role | | | | |
| DO\_TEST | On the pull | 1 'b1 represents the functional mode  1 'b0 represents the test mode | | | | |
| ICCC\_EN | The drop-down | 1 'b1 represents the consistent interconnection mode of multiple chips  1 'b0 stands for single chip mode | | | | |
| NODE\_ID [1:0] |  | Represents the processor number in the multichip consistent interconnection mode | | | | |
| CLKSEL [15:0] |  | HT clock control | | | | |
|  | signal | role | |  |
| CLKSEL [15] | 1 'b1 means that HT PLL frequency is only set by hardware  1 'b0 means HT PLL frequency can be set by software | |  |
| CLKSEL [14] | 1 'b1 represents HT PLL with normal clock input  1 'b0 means HT PLL USES differential clock input | |  |
| CLKSEL [12] | 2 'b00 represents the PHY clock of 1.6GHZ  2 'b01 represents the PHY clock of 3.2GHZ  2 'b10 represents the PHY clock of 1.2GHz  2 'b11 represents the PHY clock of 2.4GHz | |  |
| |  |  |  | | --- | --- | --- | |  |  |  | |  |  |  | |  |  | CLKSEL [10] | 2 'b00 means that the HT controller clock is the PHY clock 8 frequency division  2 'b01 means that the HT controller clock is the PHY clock 4 frequency division  2 'b10 means that the HT controller clock is the PHY clock 2 frequency division  2 'b11 means that HT controller clock is SYSCLOCK | |  |
| Note: when CLKSEL[13:10] == 4 'b1111, the HT controller clock is bypass mode, and the external input 100MHz reference clock is used directly  MEM clock control  The signal function  5 'b11111 means MEM clock directly adopts memclk  5 'b01111 means the MEM clock is set by software. See the setting method  [Section 2.6 shows](#_bookmark25)  In other cases, the MEM clock is  CLKSEL [will]  Memclk \* (clksel [and] + 30)/(clksel [9] + 3)  Note:  Memclk \*(clksel[8:5]+30) must be 1.2ghz ~ 3.2ghz memclk as the input reference clock, must be 20~40MHz  CORE clock control | | | | |
|  | signal | | role |  |
| CLKSEL [Wednesday] | | 5 'b11111 means that the CORE clock directly adopts sysclk  5 'b011xx means CORE clock is set by software. See the setting method  [Section 2.6 description.](#_bookmark25)  5 'b01111 is in normal working mode, otherwise it is in debugging mode  5 'b0110x means that the FIFO depth is set to 2  5 'b011x0 represents the DCDL control mode. In other cases, the CORE clock is sysclk\*(clksel[3:0]+30)/(clksel[4]+1). Note:  Sysclk \*(clksel[3:0]+30) must be 1.2ghz ~ 3.2ghz  Sysclk is the input reference clock and must be 20~40MHz |  |
| PCI\_CONFIG [away] |  | IO configuration control  The 7HT bus is forced to start in 1.0 mode  6:4 needs to be set to 000  3PCI main device mode | | | | |
|  |  | 2 needs to be set to 0  1 use external PCI mediation  0 use SPI to start the function | | | | |

* 1. The Cache consistency

The loongson 3A2000 is maintained by the hardware for Cache consistency between the processor and the I/O accessed through the HT port, but the hardware does not maintain the Cache consistency of the I/O devices accessed through PCI into the system. At the time of driver development, the software is required to maintain Cache consistency for DMA (Direct Memory Access) transfers to devices accessed through PCI.

#### Physical address space distribution at the system node level

The system physical address distribution of the loongson 3 series processor adopts globally accessible hierarchical addressing design to ensure

System development extension compatibility. The overall system physical address width is 48 bits. According to the height of the address, the whole address is empty

It is evenly distributed among 16 nodes, that is, each node is allocated with a 44-bit address space.

Logodson 3A2000 processor can directly use 4 chips to build cc-numa system. The processor number of each chip is determined by the pin NODEID. The address space distribution of each chip is as follows:

Table 2-2 node-level system global address distribution

|  |  |  |  |
| --- | --- | --- | --- |
| Chip node number (NODEID) | Address [47:44] | The starting address | End address |
| 0 | 0 | 0 x0000\_0000\_0000 | 0 x0fff\_ffff\_ffff |
| 1 | 1 | 0 x1000\_0000\_0000 | 0 x1fff\_ffff\_ffff |
| 2 | 2 | 0 x2000\_0000\_0000 | 0 x2fff\_ffff\_ffff |
| 3 | 3 | 0 x3000\_0000\_0000 | 0 x3fff\_ffff\_ffff |

The godson 3A2000 adopts a single-node 4-core configuration, so the corresponding addresses of the DDR memory controller, HT bus and PCI bus integrated by the godson 3A2000 chip are all contained in the 44-bit position from 0x0 (inclusive) to 0x1000\_0000\_0000 (not inclusive) inside each node, and the 44-bit address space is further uniformly distributed to up to 8 devices connected within the node. The lower 43-bit address is owned by four Shared Cache modules, and the higher 43-bit address is further determined by the [43:42] bit of the address Distributed to devices connected to four directional ports. Depending on the chip and system configuration, if there is no slave device connected to a port, the corresponding address space is reserved address space and access is not allowed.

The address space of the internal first-level cross-switch of loongson 3A2000 chip is as follows:

Table 2-3 address distribution within nodes

|  |  |  |  |
| --- | --- | --- | --- |
| equipment | Address [43:41] | The initial address within the node | End of node address |
| The Shared Cache | 0,1,2,3 | 0 x000\_0000\_0000 | 0 x7ff\_ffff\_ffff |
| HT0 controller | 6 | 0 xc00\_0000\_0000 | 0 xdff\_ffff\_ffff |
| HT1 controller | 7 | 0 xe00\_0000\_0000 | 0 xfff\_ffff\_ffff |

Unlike directional port mapping, loongson 3A2000 can determine the cross-addressing method of Shared Cache based on the actual access behavior of the application. The four Shared Cache modules in the node correspond to a total of 43 bit address Spaces, and the address space corresponding to each module is determined according to a certain two selection bits of the address bit, which can be dynamically configured and modified by the software. A configuration register named SCID\_SEL is set in the system to determine the address selection bit, as shown in the following table. By default, it is distributed in a [7:6] status hash, where the address [7:6] determines the corresponding Shared Cache number. The register address is 0x3FF00400.

Table 2-4 address distribution within nodes

|  |  |  |  |
| --- | --- | --- | --- |
| SCID\_SEL | Address bit selection | SCID\_SEL | Address bit selection |
| 4 'h0 | 7:6 | 4 'h8 | " |
| 4 'h1 | 9:8 | 4 'h9 | Thus for |
| 4 'h2 | " | 4 'ha | But after |
| 4 'h3 | She answered | 4 'hb | then |
| 4 'h4 | The lowest | 4 'hc | charm |
| 4 'h5 | " | 4 'hd | 33:32 |
| 4 'h6 | 7 | 4 'he | " |
| 4 'h7 | mark | 4 'hf | meanwhile |

#### Address routing distribution and configuration

The routing of loongson 3A2000 is mainly realized through the two-stage crossover switch of the system. A level 1 cross switch can configure the routing of requests received by each Master port. Each Master port has 8 address Windows, which can complete the target routing of 8 address Windows. Each address window consists of three 64-bit registers, BASE, MASK and MMAP. BASE is aligned with K bytes. The MASK used a format similar to the network MASK with a high position of 1.The lower three bits of MMAP represent the number of the corresponding target Slave port, MMAP[4] means the allowed reference, MMAP[5] means the allowed block read, MMAP[6] means the allowed interleaved access enable to Scache, and MMAP[7] means the window enable.

Table 2-5 MMAP field corresponding to this spatial access property

|  |  |  |  |
| --- | --- | --- | --- |
| [7] | [6] | [5] | [4] |
| The window can make | Allows interlaced access to SCACHE, valid with Slave zero, and routes requests to hit window addresses as configured in the previous section SCID\_SEL | Allow the block read | Allowed to take to |

Window hit formula :(IN\_ADDR & MASK) == BASE

Because the loongson 3 adopts fixed route by default, the configuration window is closed when it is powered on and started, so the system software is required to enable it to be configured when it is in use.

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

Table 2-6 level 1 cross switch address window register table

|  |  |  |  |
| --- | --- | --- | --- |
| address | register | address | register |
| 0 x3ff0\_2000 | CORE0\_WIN0\_BASE | 0 x3ff0\_2100 | CORE1\_WIN0\_BASE |
| 0 x3ff0\_2008 | CORE0\_WIN1\_BASE | 0 x3ff0\_2108 | CORE1\_WIN1\_BASE |
| 0 x3ff0\_2010 | CORE0\_WIN2\_BASE | 0 x3ff0\_2110 | CORE1\_WIN2\_BASE |
| 0 x3ff0\_2018 | CORE0\_WIN3\_BASE | 0 x3ff0\_2118 | CORE1\_WIN3\_BASE |
| 0 x3ff0\_2020 | CORE0\_WIN4\_BASE | 0 x3ff0\_2120 | CORE1\_WIN4\_BASE |
| 0 x3ff0\_2028 | CORE0\_WIN5\_BASE | 0 x3ff0\_2128 | CORE1\_WIN5\_BASE |
| 0 x3ff0\_2030 | CORE0\_WIN6\_BASE | 0 x3ff0\_2130 | CORE1\_WIN6\_BASE |
| 0 x3ff0\_2038 | CORE0\_WIN7\_BASE | 0 x3ff0\_2138 | CORE1\_WIN7\_BASE |
| 0 x3ff0\_2040 | CORE0\_WIN0\_MASK | 0 x3ff0\_2140 | CORE1\_WIN0\_MASK |
| 0 x3ff0\_2048 | CORE0\_WIN1\_MASK | 0 x3ff0\_2148 | CORE1\_WIN1\_MASK |
| 0 x3ff0\_2050 | CORE0\_WIN2\_MASK | 0 x3ff0\_2150 | CORE1\_WIN2\_MASK |
| 0 x3ff0\_2058 | CORE0\_WIN3\_MASK | 0 x3ff0\_2158 | CORE1\_WIN3\_MASK |
| 0 x3ff0\_2060 | CORE0\_WIN4\_MASK | 0 x3ff0\_2160 | CORE1\_WIN4\_MASK |
| 0 x3ff0\_2068 | CORE0\_WIN5\_MASK | 0 x3ff0\_2168 | CORE1\_WIN5\_MASK |
| 0 x3ff0\_2070 | CORE0\_WIN6\_MASK | 0 x3ff0\_2170 | CORE1\_WIN6\_MASK |
| 0 x3ff0\_2078 | CORE0\_WIN7\_MASK | 0 x3ff0\_2178 | CORE1\_WIN7\_MASK |
| 0 x3ff0\_2080 | CORE0\_WIN0\_MMAP | 0 x3ff0\_2180 | CORE1\_WIN0\_MMAP |
| 0 x3ff0\_2088 | CORE0\_WIN1\_MMAP | 0 x3ff0\_2188 | CORE1\_WIN1\_MMAP |
| 0 x3ff0\_2090 | CORE0\_WIN2\_MMAP | 0 x3ff0\_2190 | CORE1\_WIN2\_MMAP |
| 0 x3ff0\_2098 | CORE0\_WIN3\_MMAP | 0 x3ff0\_2198 | CORE1\_WIN3\_MMAP |
| 0 x3ff0\_20a0 | CORE0\_WIN4\_MMAP | 0 x3ff0\_21a0 | CORE1\_WIN4\_MMAP |
| 0 x3ff0\_20a8 | CORE0\_WIN5\_MMAP | 0 x3ff0\_21a8 | CORE1\_WIN5\_MMAP |
| 0 x3ff0\_20b0 | CORE0\_WIN6\_MMAP | 0 x3ff0\_21b0 | CORE1\_WIN6\_MMAP |
| 0 x3ff0\_20b8 | CORE0\_WIN7\_MMAP | 0 x3ff0\_21b8 | CORE1\_WIN7\_MMAP |
|  | | | |
| 0 x3ff0\_2200 | CORE2\_WIN0\_BASE | 0 x3ff0\_2300 | CORE3\_WIN0\_BASE |
| 0 x3ff0\_2208 | CORE2\_WIN1\_BASE | 0 x3ff0\_2308 | CORE3\_WIN1\_BASE |
| 0 x3ff0\_2210 | CORE2\_WIN2\_BASE | 0 x3ff0\_2310 | CORE3\_WIN2\_BASE |
| 0 x3ff0\_2218 | CORE2\_WIN3\_BASE | 0 x3ff0\_2318 | CORE3\_WIN3\_BASE |
| 0 x3ff0\_2220 | CORE2\_WIN4\_BASE | 0 x3ff0\_2320 | CORE3\_WIN4\_BASE |
| 0 x3ff0\_2228 | CORE2\_WIN5\_BASE | 0 x3ff0\_2328 | CORE3\_WIN5\_BASE |
| 0 x3ff0\_2230 | CORE2\_WIN6\_BASE | 0 x3ff0\_2330 | CORE3\_WIN6\_BASE |
| 0 x3ff0\_2238 | CORE2\_WIN7\_BASE | 0 x3ff0\_2338 | CORE3\_WIN7\_BASE |
| 0 x3ff0\_2240 | CORE2\_WIN0\_MASK | 0 x3ff0\_2340 | CORE3\_WIN0\_MASK |
| 0 x3ff0\_2248 | CORE2\_WIN1\_MASK | 0 x3ff0\_2348 | CORE3\_WIN1\_MASK |
| 0 x3ff0\_2250 | CORE2\_WIN2\_MASK | 0 x3ff0\_2350 | CORE3\_WIN2\_MASK |
| 0 x3ff0\_2258 | CORE2\_WIN3\_MASK | 0 x3ff0\_2358 | CORE3\_WIN3\_MASK |
| 0 x3ff0\_2260 | CORE2\_WIN4\_MASK | 0 x3ff0\_2360 | CORE3\_WIN4\_MASK |
| 0 x3ff0\_2268 | CORE2\_WIN5\_MASK | 0 x3ff0\_2368 | CORE3\_WIN5\_MASK |
| 0 x3ff0\_2270 | CORE2\_WIN6\_MASK | 0 x3ff0\_2370 | CORE3\_WIN6\_MASK |
| 0 x3ff0\_2278 | CORE2\_WIN7\_MASK | 0 x3ff0\_2378 | CORE3\_WIN7\_MASK |
| 0 x3ff0\_2280 | CORE2\_WIN0\_MMAP | 0 x3ff0\_2380 | CORE3\_WIN0\_MMAP |
| 0 x3ff0\_2288 | CORE2\_WIN1\_MMAP | 0 x3ff0\_2388 | CORE3\_WIN1\_MMAP |
| 0 x3ff0\_2290 | CORE2\_WIN2\_MMAP | 0 x3ff0\_2390 | CORE3\_WIN2\_MMAP |
| 0 x3ff0\_2298 | CORE2\_WIN3\_MMAP | 0 x3ff0\_2398 | CORE3\_WIN3\_MMAP |
| 0 x3ff0\_22a0 | CORE2\_WIN4\_MMAP | 0 x3ff0\_23a0 | CORE3\_WIN4\_MMAP |
| 0 x3ff0\_22a8 | CORE2\_WIN5\_MMAP | 0 x3ff0\_23a8 | CORE3\_WIN5\_MMAP |
| 0 x3ff0\_22b0 | CORE2\_WIN6\_MMAP | 0 x3ff0\_23b0 | CORE3\_WIN6\_MMAP |
| 0 x3ff0\_22b8 | CORE2\_WIN7\_MMAP | 0 x3ff0\_23b8 | CORE3\_WIN7\_MMAP |
|  | | | |
| 0 x3ff0\_2600 | HT0\_WIN0\_BASE | 0 x3ff0\_2700 | HT1\_WIN0\_BASE |
| 0 x3ff0\_2608 | HT0\_WIN1\_BASE | 0 x3ff0\_2708 | HT1\_WIN1\_BASE |
| 0 x3ff0\_2610 | HT0\_WIN2\_BASE | 0 x3ff0\_2710 | HT1\_WIN2\_BASE |
| 0 x3ff0\_2618 | HT0\_WIN3\_BASE | 0 x3ff0\_2718 | HT1\_WIN3\_BASE |

|  |  |  |  |
| --- | --- | --- | --- |
| 0 x3ff0\_2620 | HT0\_WIN4\_BASE | 0 x3ff0\_2720 | HT1\_WIN4\_BASE |
| 0 x3ff0\_2628 | HT0\_WIN5\_BASE | 0 x3ff0\_2728 | HT1\_WIN5\_BASE |
| 0 x3ff0\_2630 | HT0\_WIN6\_BASE | 0 x3ff0\_2730 | HT1\_WIN6\_BASE |
| 0 x3ff0\_2638 | HT0\_WIN7\_BASE | 0 x3ff0\_2738 | HT1\_WIN7\_BASE |
| 0 x3ff0\_2640 | HT0\_WIN0\_MASK | 0 x3ff0\_2740 | HT1\_WIN0\_MASK |
| 0 x3ff0\_2648 | HT0\_WIN1\_MASK | 0 x3ff0\_2748 | HT1\_WIN1\_MASK |
| 0 x3ff0\_2650 | HT0\_WIN2\_MASK | 0 x3ff0\_2750 | HT1\_WIN2\_MASK |
| 0 x3ff0\_2658 | HT0\_WIN3\_MASK | 0 x3ff0\_2758 | HT1\_WIN3\_MASK |
| 0 x3ff0\_2660 | HT0\_WIN4\_MASK | 0 x3ff0\_2760 | HT1\_WIN4\_MASK |
| 0 x3ff0\_2668 | HT0\_WIN5\_MASK | 0 x3ff0\_2768 | HT1\_WIN5\_MASK |
| 0 x3ff0\_2670 | HT0\_WIN6\_MASK | 0 x3ff0\_2770 | HT1\_WIN6\_MASK |
| 0 x3ff0\_2678 | HT0\_WIN7\_MASK | 0 x3ff0\_2778 | HT1\_WIN7\_MASK |
| 0 x3ff0\_2680 | HT0\_WIN0\_MMAP | 0 x3ff0\_2780 | HT1\_WIN0\_MMAP |
| 0 x3ff0\_2688 | HT0\_WIN1\_MMAP | 0 x3ff0\_2788 | HT1\_WIN1\_MMAP |
| 0 x3ff0\_2690 | HT0\_WIN2\_MMAP | 0 x3ff0\_2790 | HT1\_WIN2\_MMAP |
| 0 x3ff0\_2698 | HT0\_WIN3\_MMAP | 0 x3ff0\_2798 | HT1\_WIN3\_MMAP |
| 0 x3ff0\_26a0 | HT0\_WIN4\_MMAP | 0 x3ff0\_27a0 | HT1\_WIN4\_MMAP |
| 0 x3ff0\_26a8 | HT0\_WIN5\_MMAP | 0 x3ff0\_27a8 | HT1\_WIN5\_MMAP |
| 0 x3ff0\_26b0 | HT0\_WIN6\_MMAP | 0 x3ff0\_27b0 | HT1\_WIN6\_MMAP |
| 0 x3ff0\_26b8 | HT0\_WIN7\_MMAP | 0 x3ff0\_27b8 | HT1\_WIN7\_MMAP |

There are three ip-related address Spaces in the secondary XBAR of loongson 3, namely, the configuration register address space, the DDR2 address space, and the PCI address space. The address window is set for routing and address translation between the CPU and pci-dma IP with the main device function. Both the CPU and pci-dma have eight address Windows that enable the selection of the target address space and the conversion from the source address space to the target address space.

Each address window is composed of three 64-bit registers, BASE, MASK and MMAP. BASE is aligned with K bytes, and MASK USES a format similar to the network MASK with the high digit of 1. MMAP contains the transformed address, routing and enabling alleles, as shown in the following table:

|  |  |  |  |
| --- | --- | --- | --- |
| 48] [63: | [47:10] | [17] | [3-0] |
| Alternating selective bit | Converted address | The window can make | From the device number |

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

Table 2-7 at level 2 XBAR, the corresponding relationship between the device number and the module

|  |  |
| --- | --- |
| From the device number | The default value |
| 0 | No. 0 ddr2/3 controller |
| 1 | No. 1 ddr2/3 controller |
| 2 | Low speed I/O (PCI, LPC, etc.) |
| 3 | Configuration register |

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

Table 2-8 MMAP field corresponding to this spatial access property

|  |  |  |  |
| --- | --- | --- | --- |
| [7] | [6] | [5] | [4] |
| The window can make | Allows interleaved access to the DDR, which is valid when the device number is 0, and routes requests to the address of the hit window according to the configuration of "interleaved selection bits". Requires a staggered enable bit  More than 10 | Allow the block read | Allowed to take to |

It is important to note that the window configuration of level 1 XBAR cannot translate the address of the Cache consistent request, otherwise the address at SCache will be different from that at the processor level Cache, resulting in the maintenance error of Cache consistency.

Window hit formula :(IN\_ADDR & MASK) == BASE

New address conversion formula :OUT\_ADDR = (IN\_ADDR & ~MASK) | {MMAP[63:10],10 'h0}

Address window conversion register is shown in the following table:

Table 2-9 the register table is converted from the level 2 XBAR address window

|  |  |  |  |
| --- | --- | --- | --- |
| address | register | describe | The default value |
| 3 ff0 0000 | CPU\_WIN0\_BASE | Base address of CPU window 0 | 0 x0 |
| 3 ff0 0008 | CPU\_WIN1\_BASE | Base address of CPU window 1 | 0 x1000\_0000 |
| 3 ff0 0010 | CPU\_WIN2\_BASE | Base address of CPU window 2 | 0 x0 |
| 3 ff0 0018 | CPU\_WIN3\_BASE | Base address of CPU window 3 | 0 x0 |
| 3 ff0 0020 | CPU\_WIN4\_BASE | Base address of CPU window 4 | 0 x0 |
| 3 ff0 0028 | CPU\_WIN5\_BASE | Base address of CPU window 5 | 0 x0 |
| 3 ff0 0030 | CPU\_WIN6\_BASE | Base address of CPU window 6 | 0 x0 |
| 3 ff0 0038 | CPU\_WIN7\_BASE | The base address of CPU window 7 | 0 x0 |
| 3 ff0 0040 | CPU\_WIN0\_MASK | The mask for CPU window 0 | 0 xffff\_ffff\_f000\_0000 |
| 3 ff0 0048 | CPU\_WIN1\_MASK | Mask for CPU window 1 | 0 xffff\_ffff\_f000\_0000 |
| 3 ff0 0050 | CPU\_WIN2\_MASK | CPU window 2 mask | 0 x0 |
| 3 ff0 0058 | CPU\_WIN3\_MASK | Mask for CPU window 3 | 0 x0 |
| 3 ff0 0060 | CPU\_WIN4\_MASK | Mask for CPU window 4 | 0 x0 |
| 3 ff0 0068 | CPU\_WIN5\_MASK | Mask for CPU window 5 | 0 x0 |
| 3 ff0 0070 | CPU\_WIN6\_MASK | Mask for CPU window 6 | 0 x0 |
| 3 ff0 0078 | CPU\_WIN7\_MASK | CPU window 7 mask | 0 x0 |
| 3 ff0 0080 | CPU\_WIN0\_MMAP | New base address for CPU window 0 | 0 xf0 |
| 3 ff0 0088 | CPU\_WIN1\_MMAP | New base address for CPU window 1 | 0 x1000\_00f2 |
| 3 ff0 0090 | CPU\_WIN2\_MMAP | New base address for CPU window 2 | 0 |
| 3 ff0 0098 | CPU\_WIN3\_MMAP | New base address for CPU window 3 | 0 |
| 3 ff0 00 a nought | CPU\_WIN4\_MMAP | New base address for CPU window 4 | 0 x0 |
| 3 ff0 00 a8 | CPU\_WIN5\_MMAP | New base address for CPU window 5 | 0 x0 |
| 3 ff0 00 b0 | CPU\_WIN6\_MMAP | New base address for CPU window 6 | 0 |
| 3 ff0 00 b8 | CPU\_WIN7\_MMAP | New base address for CPU window 7 | 0 |
| 3 ff0 0100 | PCI\_WIN0\_BASE | Base address for PCI window 0 | 0 x8000\_0000 |
| 3 ff0 0108 | PCI\_WIN1\_BASE | Base address for PCI window 1 | 0 x0 |
| 3 ff0 0110 | PCI\_WIN2\_BASE | PCI window 2 base address | 0 x0 |
| 3 ff0 0118 | PCI\_WIN3\_BASE | Base address for PCI window 3 | 0 x0 |
| 3 ff0 0120 | PCI\_WIN4\_BASE | Base address for PCI window 4 | 0 x0 |
| 3 ff0 0128 | PCI\_WIN5\_BASE | Base address for PCI window 5 | 0 x0 |
| 3 ff0 0130 | PCI\_WIN6\_BASE | Base address for PCI window 6 | 0 x0 |
| 3 ff0 0138 | PCI\_WIN7\_BASE | Base address for PCI window 7 | 0 x0 |
| 3 ff0 0140 | PCI\_WIN0\_MASK | Mask for PCI window 0 | 0 xffff\_ffff\_8000\_0000 |
| 3 ff0 0148 | PCI\_WIN1\_MASK | Mask for PCI window 1 | 0 x0 |
| 3 ff0 0150 | PCI\_WIN2\_MASK | Mask for PCI window 2 | 0 x0 |
| 3 ff0 0158 | PCI\_WIN3\_MASK | Mask for PCI window 3 | 0 x0 |
| 3 ff0 0160 | PCI\_WIN4\_MASK | Mask for PCI window 4 | 0 x0 |
| 3 ff0 0168 | PCI\_WIN5\_MASK | Mask for PCI window 5 | 0 x0 |
| 3 ff0 0170 | PCI\_WIN6\_MASK | Mask for PCI window 6 | 0 x0 |
| 3 ff0 0178 | PCI\_WIN7\_MASK | Mask for PCI window 7 | 0 x0 |
| 3 ff0 0180 | PCI\_WIN0\_MMAP | New base address for PCI window 0 | 0 xf0 |
| 3 ff0 0188 | PCI\_WIN1\_MMAP | New base address for PCI window 1 | 0 x0 |
| 3 ff0 0190 | PCI\_WIN2\_MMAP | New base address for PCI window 2 | 0 |
| 3 ff0 0198 | PCI\_WIN3\_MMAP | New base address for PCI window 3 | 0 |
| 3 ff0 a0 01 | PCI\_WIN4\_MMAP | New base address for PCI window 4 | 0 x0 |
| 3 ff0 01 a8 | PCI\_WIN5\_MMAP | New base address for PCI window 5 | 0 x0 |
| 3 ff0 b0 01 | PCI\_WIN6\_MMAP | New base address for PCI window 6 | 0 |
| 3 ff0 b8 01 | PCI\_WIN7\_MMAP | New base address for PCI window 7 | 0 |

According to the default register configuration, CPU 0x00000000-0x0fffffff address range after chip startup

(256M) is mapped to the address range of 0x00000000-0x0fffffff of DDR2, 0x10000000-0x1fffffff of CPU (256M) is mapped to 0x10000000-0x1fffffff of PCI, and 0x80000000 of PCIDMA

The address interval of -0x8fffffff (256M) maps to the address interval of 0x00000000-0x0fffffff of DDR2. The software can realize the new address space routing and transformation by modifying the corresponding configuration registers.

In addition, when there is read access to illegal address caused by CPU guess execution, the 8 address Windows are not hit, and the configuration register module returns all 0 data to CPU to prevent CPU dead and so on.

Table 2-10 XBAR level 2 default address configuration

|  |  |  |
| --- | --- | --- |
| Base address | high | The owner of the |
| 0 x0000\_0000\_0000\_0000 | 0 x0000\_0000\_0fff\_ffff | No. 0 DDR controller |
| 0 x0000\_0000\_1000\_0000 | 0 x0000\_0000\_1fff\_ffff | Low speed I/O (PCI, etc.) |

#### Chip configuration and sampling register

The chip configuration register (Chip\_config) and the chip sampling register (Chip\_sample) in the loong chip 3A2000 provide the mechanism for reading and writing the chip configuration.

Table 2-11 chip configuration register (physical address 0x1fe00180)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A domain** | **The field name** | **access** | **Reset value** | **describe** |
| 3-0 | - | RW | 4 'b7 | reserve |
| 4 | MC0\_disable\_ddr2\_confspace | RW | 1 'b0 | Whether to disable the MC0 DDR configuration space |
| 5 | - | RW | 1 'b0 | reserve |
| 6 | - | RW | 1 'b0 | reserve |
| 7 | MC0\_ddr2\_resetn | RW | 1 'b1 | MC0 software reset (low efficiency) |
| 8 | MC0\_clken | RW | 1 'b1 | Whether to enable MC0 |
| 9 | MC1\_disable\_ddr2\_confspace | RW | 1 'b0 | Whether to disable the MC1 DDR configuration space |
| 10 | - | RW | 1 'b0 | reserve |
| 11 | - | RW | 1 'b0 | reserve |
| 12 | MC1\_ddr2\_resetn | RW | 1 'b1 | MC1 software reset (low efficiency) |
| 13 | MC1\_clken | RW | 1 'b1 | Whether to enable MC1 |
| 26:24 | HT0\_freq\_scale\_ctrl | RW | 3 'b111 | HT controller 0 frequency division |
| 27 | HT0\_clken | RW | 1 'b1 | Whether to enable HT0 |
| 30:28 | HT1\_freq\_scale\_ctrl | RW | 3 'b111 | HT controller 1 frequency division |
| 31 | HT1\_clken | RW | 1 'b1 | Whether to enable HT1 |
| 42:40 | Node0\_freq\_ CTRL | RW | 3 'b111 | Node 0 frequency division |
| 43 | - | RW | 1 'b1 |  |
| 46:44 | Node1\_freq\_ CTRL | RW | 3 'b111 | Node 1 frequency division |
| 47 | - | RW | 1 'b1 |  |
| 63:56 | Cpu\_version | R | 2 'h37 | The CPU version |
| 95:64 |  |  |  | (empty) |
| 127:96 | Pad1v8\_ctrl | RW | 6 'h780 | 1 v8 control pad |
| other |  | R |  | reserve |

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A domain** | **The field name** | **access** | **Reset value** | **describe** |
| 31:0 | Compcode\_core | R |  |  |
| 47:32 | Sys\_clkseli | R |  | On board frequency doubling 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 |  | Is the 2 HT controllers bad |
| 83:80 | Compcode\_ok | R |  |  |
| 88 | Thsens0\_overflow | R |  | Overflow of temperature sensor 0 (over 125℃) |
| 89 | Thsens1\_overflow | R |  | Overflow of temperature sensor 1 (over 125℃) |
| 103:96 | Thsens0\_out | R |  | Temperature sensor: 0 ℃  Node temperature = thens0\_out-100 temperature range -40 ° -125 ° |
| 111:104 | Thsens1\_out | R |  | Temperature sensor 1 ℃ temperature  Node temperature = thens1\_out-100 temperature range -40 ° -125 ° |
| other |  | R |  | reserve |

The following sets of software frequency doubling Settings registers are used to set the working frequencies of each clock when CLKSEL is configured in software control mode (refer to the CLKSEL setting method in section 2.2). Among them, the MEM CLOCK configuration corresponds to the memory controller and the bus CLOCK frequency;The CORE CLOCK corresponds to the CLOCK frequency of processor CORE, on-chip network and high-speed Shared cache.HT CLOCK corresponds to HT controller CLOCK frequency.

Each clock configuration typically has two parameters, DIV\_LOOPC and DIV\_OUT. The final clock frequency is (reference clock \* DIV\_LOOPC)/DIV\_OUT.

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

In the software control mode, the default corresponding CLOCK frequency is the frequency of the external reference CLOCK (for CORE CLOCK, is the corresponding frequency of pin SYS\_CLK; For MEM CLOCK, is the corresponding frequency of pin MEM\_CLK), you need to set the software for the CLOCK during processor startup. The process of setting each clock should be as follows:

1. Other registers in the setting register except SEL\_PLL\_\* and SOFT\_SET\_PLL, that is, these two registers are written as 0 in the setting process;
2. Set SOFT\_SET\_PLL to 1 with other register values unchanged;
3. The lock signal LOCKED\_\* in the wait register is 1;
4. Set SEL\_PLL\_\* to 1, and the corresponding clock frequency will be switched to the frequency set by the software.

Table 2-13 chip node and processor core software frequency doubling setting

register (physical address 0x1fe00180)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A domain** | **The field name** | **access** | **Reset value** | **describe** |
| 0 | SEL\_PLL\_NODE | RW | 0 x0 | Bypass the entire Node clock  PLL |
| 1 | - | RW | 0 x0 | - |
| 2 | SOFT\_SET\_PLL | RW | 0 x0 | Allows software to set up PLL |
| 3 | BYPASS\_L1 | RW | 0 x0 | Bypass L1 PLL |
| 6:4 | - | RW | 0 x0 | - |
| 7 | LOCKEN\_L1 | RW | 0 x0 | Allows you to lock L1 PLL |
| 9:8 | - | RW | 0 x0 | - |
| 11:10 | LOCKC\_L1 | RW | 0 x0 | Determines whether L1 PLL locks the phase used  The accuracy of the |
| 15:12 | - | RW | 0 x0 | - |
| 16 | LOCKED\_L1 | R | 0 x0 | Whether L1 PLL is locked |
| 18:17 | - | R | 0 x0 | - |
| 19 | PD\_L1 | R/W | 0 x0 | Close the L1 PLL |
| 31:20 | - | RW | 0 x1 | - |
| 38:32 | L1\_DIV\_LOOPC | RW | 0 x1 | L1 PLL input parameter |
| 41:39 | - |  |  | - |
| 47:42 | L1\_DIV\_OUT | RW | 0 x1 | L1 PLL input parameter |
| other | - | RW |  | reserve |

Note: PLL ouput = (clk\_ref \* div\_loopc)/div\_out.

The PLL VCO frequency (in brackets above) must be in the range of 1.2ghz to 3.2ghz. This requirement applies equally to MEM PLL and HT PLL.

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A domain** | **The field name** | **access** | **Reset value** | **describe** |
| 0 | SEL \_MEM\_PLL | RW | 0 x0 | MEM clock non-software bypass entire PLL |
| 1 | SOFT\_SET\_MEM\_PLL | RW | 0 x0 | Allows software to set up MEM PLL |
| 2 | BYPASS\_MEM\_PLL | RW | 0 x0 | Bypass MEM\_PLL |
| 3 | LOCKEN\_MEM\_PLL | RW | 0 x0 | Allows you to lock MEM\_PLL |
| when | LOCKC\_ MEM\_PLL | RW | 0 x0 | Determines whether MEM PLL is locked to the accuracy of the phase used |
| 6 | LOCKED\_MEM\_PLL | R | 0 x0 | Whether MEM\_PLL is locked or not |
| 7 | PD\_MEM\_PLL | RW | 0 x0 | Close the MEM PLL |
| Will you | - | RW | 0 x1 | - |
| brake | MEM\_PLL\_DIV\_LOOPC | RW | 0 x41 | MEM PLL input parameters |
| A partner | MEM\_PLL\_DIV\_OUT | RW | 0 x0 | MEM PLL input parameters |
| 32 | SEL\_HT0\_PLL | RW | 0 x0 | HT0 bypass PLL |
| 33 | SOFT\_SET\_HT0\_PLL | RW | 0 x0 | Allows software to set HT0 PLL |
| 34 | BYPASS\_HT0\_PLL | RW | 0 x0 | Bypass HT0\_PLL |
| 35 | LOCKEN\_HT0\_PLL | RW | 0 x0 | Allows locking HT0 PLL |
| meanwhile | LOCKC\_HT0\_PLL | RW | 0 x0 | Determine whether the HT0 PLL is locked  Precision of phase |
| 38 | LOCKED\_HT0\_PLL | R | 0 x0 | Whether HT0\_PLL is locked or not |
| 45:40:15 | HT0\_DIV\_HTCORE | RW | 0 x1 | HT0 Core PLL input parameter |
| 48 | SEL\_HT1\_PLL | RW | 0 x0 | HT1 bypass PLL |
| 49 | SOFT\_SET\_HT1\_PLL | RW | 0 x0 | Allows software to set HT1 PLL |
| 50 | BYPASS\_HT1\_PLL | RW | 0 x0 | Bypass HT1\_PLL |
| 51 | LOCKEN\_HT1\_PLL | RW | 0 x0 | Allows locking of HT1 PLL |
| 53:52 | LOCKC\_HT1\_PLL | RW | 0 x0 | Determine whether the HT1 PLL is locked using the phase accuracy |
| 54 | LOCKED\_HT1\_PLL | R | 0 x0 | Whether HT1\_PLL is locked or not |
| 61:56 | HT1\_DIV\_HTCORE | RW | 0 x1 | HT1 Core PLL input parameter |
| other |  | RW |  | reserve |

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A domain** | **The field name** | **access** | **Reset value** | **describe** |
| The 2-0 | core0\_freqctrl | RW | 0 x7 | Kernel 0 frequency division control value |
| 3 | core0\_en | RW | 0 x1 | Nuclear 0 clock enabled |
| 6:4 | core1\_freqctrl | RW | 0 x7 | Kernel 1 frequency division control value |
| 7 | core1\_en | RW | 0 x1 | Nuclear 1 clock enabled |
| 10:8 | core2\_freqctrl | RW | 0 x7 | Kernel 2 frequency division control value |
| 11 | core2\_en | RW | 0 x1 | Nuclear 2 clock enabled |
| then | core3\_freqctrl | RW | 0 x7 | Kernel 3 frequency division control value |
| 15 | core3\_en | RW | 0 x1 | Nuclear 3 clock enabled |
|  |  |  | Note: | The clock frequency value after software frequency division is equal to the original (frequency division control value +1) /8 |

GS464e processor core

The GS464e is a four-emission 64-bit high-performance processor core. The processor core can be used as a single core for high-end embedded applications and desktop applications, and can also be used as a basic processor core for in-chip multi-core systems for server and high-performance machine applications. Multiple GS464 cores in loongson 3A2000 form a multicore structure of a last-level Cache on a distributed Shared chip through a Shared Cache module and a AXI interconnection network. The main features of GS464 are as follows:

* MIPS64 compatible, support loongson extended instruction set;
* Four emission superscalar structure, two fixed points, two floating point, two memory access parts;
* Each floating point component supports full-flow 64-bit/double-32-bit floating point multiplication and addition;
* Memory access parts support 128 bit storage access, virtual address is 64 bit, physical address is 48 bit;
* Support register renaming, dynamic scheduling, transfer prediction and other out-of-order execution techniques;
* 64 items are fully linked, and 1024 items are connected to 8-channel group, a total of 1088 TLB items, 64 instructions TLB, variable page size;
* The size of the first-level instruction Cache and the data Cache is 64KB respectively, and the four-way groups are connected.
* Victim Cache is a private second-level Cache with a size of 256KB and a 16-way connection.
* Support the optimization technology of non-blocking visit and load-speculation;
* Support Cache consistency protocol, which can be used in chip multi-core processors;
* The instruction Cache implements parity check, and the data Cache implements ECC check.
* Standard EJTAG debugging standard is supported to facilitate debugging of software and hardware.
* Standard 128 - bit AXI interface.

The structure of GS464e is shown in the following figure. For more details on GS464e, please refer to the GS464e user manual as well

MIPS64 user's manual.

PC + 8

By 8 inst

Decode bus

X4 inst

The Mapping table

**DCache Tag compare**

Figure 3-1 GS464e structure diagram



Commit bus

The Reorder the queue

Prediction bus

BRQ

The Map bus

ROQ

The Result bus

**The PC**

**Fix issu que**

**64 KB**

**DCache**

**CP0**

**The queue**

Refill bus

Dmemwrite bus

Imemread bus

Dmemread, duncache bus

**The Test interface**

**EJTAG interface**

**AXI interface**

**CLK, RST, INT...**

Miss queue

Uncache queue

**Decode**

**Reg the Map**

The Exception bus

Branch bus

**FP**

**The register file**

**GP**

**The register file**

**E ue**

EJTAG TAP

The controller

The Test controller

The Performance monitor

Prefetch engine

256 KB

VCache

**FPU2**

**64 KB**

**ICache**

**FPU1**

**Float issue queue**

**ITLB**

**AGU2**

**MMU**

**AGU1**

**Mem issue queue**

**ALU2**

**Branch predictor**

**ALU1**

The Main TLB

DTLB

**cache2mem**

## Shared Cache (SCache)

The SCache module is a three-level Cache Shared by all processor cores in the loongson 3A2000 processor.The main features of the SCache module include:

* Use 128 - bit AXI interface.
* 16 Cache items access the queue.
* Keyword first.
* The fastest time to receive a read failure request is 12 beats.
* Cache consistency protocol is supported through directories.
* It can be used for multi-core structure on chip, and also can be directly connected with IP of single processor.
* Adopt 16 - way group - linked structure.
* Support ECC validation.
* Support DMA consistent read-write and prefetch reads.
* Support 16 Shared Cache hashes.
* Support for sharing Cache by window lock.
* Ensure that the 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 the processor's access requests from the processor and DMA, while information such as tags, directories, and data that share the Cache is stored in the scacheaccess module. To reduce power consumption, tags, directories, and DATA in a Shared Cache can be accessed separately. The Shared Cache status bit and w bit are stored with tags, which are stored in TAG RAM, DIR RAM, and DATA RAM. Invalid requests to access the Shared Cache, read all the tags and directories, select the directory based on the TAG, and read the data based on the hit. Replace requests, refill requests, and write back requests operate only on tags, directories, and data along the way.

To improve the performance of certain computing tasks, a locking mechanism has been added to the Shared Cache. A Shared Cache block in a locked area is locked and will not be replaced from the Shared Cache (unless the 16-way Shared Cache is full of locked blocks). The four groups of lock window registers inside the Shared Cache module can be dynamically configured through the chip configuration register space, but one of the 16 Shared caches must not be locked. The size of each group of Windows can be adjusted according to the mask, but not more than 3/4 of the overall Shared Cache size. In addition, when the Shared Cache receives a DMA write request, if the region being written is hit in the Shared Cache and locked, then the DMA write will be written directly to the Shared Cache instead of to the Shared Cach Memory.

Table 4-1 Shared Cache lock window register configuration

|  |  |  |  |
| --- | --- | --- | --- |
| The name of the | address | A domain | describe |
| Slock0\_valid | 0 x3ff00200 | [63-63] | Zero lock window valid bit |
| Slock0\_addr | 0 x3ff00200 | [47:0] | Lock window 0 lock address |
| Slock0\_mask | 0 x3ff00240 | [47:0] | Lock window mask no. 0 |
| Slock1\_valid | 0 x3ff00208 | [63-63] | No. 1 lock window valid bit |
| Slock1\_addr | 0 x3ff00208 | [47:0] | No. 1 lock window lock address |
| Slock1\_mask | 0 x3ff00248 | [47:0] | No. 1 lock window mask |
| Slock2\_valid | 0 x3ff00210 | [63-63] | No. 2 lock window valid bit |
| Slock2\_addr | 0 x3ff00210 | [47:0] | No. 2 lock window lock address |
| Slock2\_mask | 0 x3ff00250 | [47:0] | No. 2 lock window mask |
| Slock3\_valid | 0 x3ff00218 | [63-63] | No. 3 lock window valid bit |
| Slock3\_addr | 0 x3ff00218 | [47:0] | No. 3 lock window lock address |
| Slock3\_mask | 0 x3ff00258 | [47:0] | No. 3 lock window mask |

For example, when an address addr makes slock0\_valid && ((addr & slock0\_mask) == (slock0\_addr & slock0\_mask)) equal to 1, the address is locked by lock window 0.

## Matrix processing accelerator

Logodson 3A2000 is built with two matrix processing accelerators independent of the processor core. Its basic function is to realize the function of transposing or moving the matrix stored in memory from the source matrix to the target matrix through the configuration of software. The two accelerators are respectively integrated in the two HyperTransport controllers of loongson 3A2000, and the read-write of SCache and memory can be realized through a cross switch.

Before due to transpose the same Cache line element order after the transposed matrix is distributed, in order to improve the efficiency of reading and writing, need read many rows of data, makes the data can be in after the transposed matrix unit to write to the Cache behavior, thus set up a size 32 in the module the buffer zone, realize transverse writing (matrix into the buffer from the source), longitudinal read (matrix) by the buffer is written to the target.

The working process of matrix processing is to read in the 32 rows of source matrix data, and then write the 32 rows of data to the target matrix, and so on, until the entire matrix is transposed or moved. The matrix processing accelerator can also prefetch the source matrix without writing the target matrix as required, so as to achieve the operation of prefetching the data to SCache.

The source matrix involved in transpose or shift may be a small matrix located in a large matrix, so its matrix address may not be completely contiguous, and the addresses between adjacent rows will be spaced, requiring more programming control interfaces to be implemented. The following tables 5-1 through 5-4 illustrate the programming interfaces involved in matrix processing.

Table 5-1 matrix processing programming interface description

|  |  |  |  |
| --- | --- | --- | --- |
| address | The name of the | attribute | instructions |
| 0 x3ff00600 | src\_start\_addr | RW | The starting address of the source matrix |
| 0 x3ff00608 | dst\_start\_addr | RW | The starting address of the destination matrix |
| 0 x3ff00610 | The row | RW | The number of elements in a row in the source matrix |
| 0 x3ff00618 | Col. | RW | The number of elements in a column of the source matrix |
| 0 x3ff00620 | length | RW | The row span (bytes) of the large matrix in which the source matrix is located |
| 0 x3ff00628 | width | RW | The row span (bytes) of the large matrix in which the target matrix is located |
| 0 x3ff00630 | trans\_ctrl | RW | Transpose the control register |
| 0 x3ff00638 | trans\_status | RO | Transpose the status register |

Table 5-2 matrix processing register address description

|  |  |
| --- | --- |
| address | The name of the |
| 0 x3ff00600 | Src\_start\_addr of the zero transpose module |
| 0 x3ff00608 | Dst\_start\_addr of the zero transpose module |
| 0 x3ff00610 | Row of the zero transpose module |
| 0 x3ff00618 | Col of the 0 transposed module |
| 0 x3ff00620 | Length of the zero transpose module |
| 0 x3ff00628 | Width of the zero transpose module |
| 0 x3ff00630 | Trans\_ctrl for the zero transpose module |
| 0 x3ff00638 | Trans\_status of the zero transpose module |
| 0 x3ff00700 | Src\_start\_addr of the # 1 transpose module |
| 0 x3ff00708 | Dst\_start\_addr of the # 1 transpose module |
| 0 x3ff00710 | Src\_row\_stride of no. 1 transpose module |
| 0 x3ff00718 | Src\_last\_row\_addr of the # 1 transpose module |
| 0 x3ff00720 | Length of the number 1 transpose module |
| 0 x3ff00728 | Width of no. 1 transpose module |
| 0 x3ff00730 | Trans\_ctrl for the # 1 transpose module |
| 0 x3ff00738 | Trans\_status for the # 1 transpose module |

Table 5-3 trans\_ctrl register description

|  |  |
| --- | --- |
| field | instructions |
| 0 | Can make a |
| 1 | Whether writing the target matrix is allowed. When the value is 0, the transpose process only prefetches the source matrix, but does not write the target matrix. |
| 2 | After the source matrix is read, whether it is valid or not is interrupted. |
| 3 | After the completion of writing the target matrix, whether it is valid or not, |
| 7.. 4 | Arcmd, read command internal control bit. When arcache is 4 'hf, it must be set to 4' hc. When arcache is another value, it is meaningless. |
| 11.. 8 | Arcache, read command internal control bit. The cache path is used when 4 'hf, and the uncache path is used when 4' h0. other  The value is meaningless. |
| 15..12 | Awcmd, write command internal control bit. When awcache is 4 'hf, it must be set to 4' hb. It is meaningless when awcache is other values. |
| 19..16 | Awcache, write command internal control bit. The cache path is used when 4 'hf, and the uncache path is used when 4' h0. other |
|  | The value is meaningless. |
| 21..20 | The element size of the matrix, 00 for 1 byte, 01 for 2 bytes, 10 for 4 bytes, and 11 for 8 bytes |
| 22 | Trans\_yes, 1 means transpose; Zero means no transpose |

Table 5-4 trans\_status register description

|  |  |
| --- | --- |
| field | instructions |
| 0 | The source matrix is read |
| 1 | The target matrix is written |

# Interrupt and communication between processor cores

Loongson 3A2000 implements 8 intercore interrupt registers (IPI) for each processor core to support interrupt and communication between processor cores during multi-core BIOS startup and operating system runtime. The instructions and addresses are shown in tables 6-1 through 6-5.

Table 6-1 registers associated with interrupts between processor cores and their functional descriptions

|  |  |  |
| --- | --- | --- |
| The name of the | Read and write access | describe |
| IPI\_Status | R | The 32-bit status register, where any bit is set to 1 and the corresponding bit is enabled, is set to the processor core INT4 interrupts. |
| IPI\_Enable | RW | The 32-bit enable register controls whether the corresponding interrupt bit is valid |
| IPI\_Set | W. | 32 position bit register, write 1 to the corresponding bit, the corresponding STATUS register bit is set 1 |
| IPI\_Clear | W. | 32 bit clear register, write 1 to the corresponding bit, the corresponding STATUS register bit is clear 0 |
| MailBox0 | RW | Cache register, used to pass parameters at startup, 64 or 32 bits  Uncache method for access. |
| MailBox01 | RW | Cache register, used to pass parameters at startup, 64 or 32 bits  Uncache method for access. |
| MailBox02 | RW | Cache register, used to pass parameters at startup, 64 or 32 bits  Uncache method for access. |
| MailBox03 | RW | Cache register, used to pass parameters at startup, 64 or 32 bits  Uncache method for access. |

The registers related to intercore interrupts in loongson 3A2000 and their functions are described as follows: table 6-20 processor core interrupts and communication registers list

|  |  |  |  |
| --- | --- | --- | --- |
| The name of the | address | permissions | describe |
| Core0\_IPI\_Status |  | R | The IPI\_Status register for the number 0 processor core |
| Core0\_IPI\_Enalbe | 0 x3ff01004 | RW | The IPI\_Enalbe register of the number 0 processor core |
| Core0\_IPI\_Set | 0 x3ff01008 | W. | The IPI\_Set register of the number 0 processor core |
| Core0 \_IPI\_Clear | 0 x3ff0100c | W. | The IPI\_Clear register for the number 0 processor core |
| Core0\_MailBox0 | 0 x3ff01020 | RW | Register IPI\_MailBox0 of the number 0 processor core |
| Core0\_ MailBox1 | 0 x3ff01028 | RW | Register IPI\_MailBox1 of processor core 0 |
| Core0\_ MailBox2 | 0 x3ff01030 | RW | Register IPI\_MailBox2 for the number 0 processor core |
| Core0\_ MailBox3 | 0 x3ff01038 | RW | Register IPI\_MailBox3 of processor core 0 |

Table 6-3 intercore interrupt and communication registers list for processor core no. 1

|  |  |  |  |
| --- | --- | --- | --- |
| The name of the | address | permissions | describe |
| Core1\_IPI\_Status | 0 x3ff01100 | R | The IPI\_Status register for the number 1 processor core |
| Core1\_IPI\_Enalbe | 0 x3ff01104 | RW | The IPI\_Enalbe register of the number 1 processor core |
| Core1\_IPI\_Set | 0 x3ff01108 | W. | The IPI\_Set register of the number 1 processor core |
| Core1 \_IPI\_Clear | 0 x3ff0110c | W. | The IPI\_Clear register of the number 1 processor core |
| Core1\_MailBox0 | 0 x3ff01120 | R | Register IPI\_MailBox0 for the number 1 processor core |
| Core1\_ MailBox1 | 0 x3ff01128 | RW | Register IPI\_MailBox1 of the number 1 processor core |
| Core1\_ MailBox2 | 0 x3ff01130 | W. | The IPI\_MailBox2 register of the number 1 processor core |
| Core1\_ MailBox3 | 0 x3ff01138 | W. | Register IPI\_MailBox3 for the number 1 processor core |

Table 6-4 intercore interrupt and communication registers list for no. 2 processor core

|  |  |  |  |
| --- | --- | --- | --- |
| The name of the | address | permissions | describe |
| Core2\_IPI\_Status | 0 x3ff01200 | R | The IPI\_Status register of the number 2 processor core |
| Core2\_IPI\_Enalbe | 0 x3ff01204 | RW | The IPI\_Enalbe register of the number 2 processor core |
| Core2\_IPI\_Set | 0 x3ff01208 | W. | The IPI\_Set register of the number 2 processor core |
| Core2 \_IPI\_Clear | 0 x3ff0120c | W. | The IPI\_Clear register of the number 2 processor core |
| Core2\_MailBox0 | 0 x3ff01220 | R | Register IPI\_MailBox0 of the number 2 processor core |
| Core2\_ MailBox1 | 0 x3ff01228 | RW | Register IPI\_MailBox1 in the number 2 processor core |
| Core2\_ MailBox2 | 0 x3ff01230 | W. | The IPI\_MailBox2 register of the number 2 processor core |
| Core2\_ MailBox3 | 0 x3ff01238 | W. | The IPI\_MailBox3 register of the number 2 processor core |

Table 6-5 intercore interrupt and communication registers list for no. 3 processor core

|  |  |  |  |
| --- | --- | --- | --- |
| The name of the | address | permissions | describe |
| Core3\_IPI\_Status | 0 x3ff01300 | R | The IPI\_Status register of the number 3 processor core |
| Core3\_IPI\_Enalbe | 0 x3ff01304 | RW | The IPI\_Enalbe register of the number 3 processor core |
| Core3\_IPI\_Set | 0 x3ff01308 | W. | The IPI\_Set register of the number 3 processor core |
| Core3 \_IPI\_Clear | 0 x3ff0130c | W. | The IPI\_Clear register of the number 3 processor core |
| Core3\_MailBox0 | 0 x3ff01320 | R | Register IPI\_MailBox0 for the number 3 processor core |
| Core3\_ MailBox1 | 0 x3ff01328 | RW | Register IPI\_MailBox1 for the number 3 processor core |
| Core3\_ MailBox2 | 0 x3ff01330 | W. | The IPI\_MailBox2 register of the number 3 processor core |
| Core3\_ MailBox3 | 0 x3ff01338 | W. | The IPI\_MailBox3 register of the number 3 processor core |

The above is a list of intercore interrupt related registers for a single-node multiprocessor system consisting of a single loongson 3A2000 chip. When a multi-node cc-numa system is constructed by multi-chip godson 3A2000 interconnection, each node in the chip corresponds to a system global node number, and the IPI register address of the processor core in the node is according to the above table and the base of its node

Address into a fixed offset relationship. For example, the IPI\_Status address of node 0 processor core 0 is 0x3ff01000 and 1

The address of processor no. 0 of node no. 1 is 0x10003ff01000, and so on.

## I/O interrupt

As shown in figure 7-1 below, any IO interrupt source can be configured to enable or disable, trigger mode, and target processor core interrupt pin to be outed.

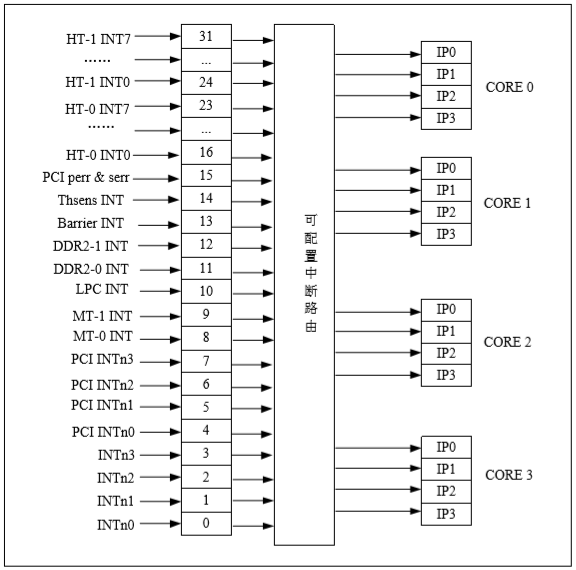


Figure 7-1 loongson 3A2000 processor interrupt routing diagram

The interruption-related configuration registers control the corresponding interrupts in the form of bits. See table 7-1 below for the connection and property configuration of interrupt control bits.The interrupt Enable configuration has three registers: Intenset, Intenclr, and Inten. The Intenset sets the interrupt enable, and the interrupt corresponding to the bit write 1 in the Intenset register is enabled. The Intenclr clears interrupts to enable, and the interrupt corresponding to the Intenclr register write 1 is cleared. The Inten register reads the current status of each interrupt enable.Interrupt signals in the form of pulses (such as PCI\_SERR) are selected by the Intedge configuration register, with write 1 for pulse trigger and write 0 for level trigger. The interrupt handler can clear the pulse record by the corresponding bit of the Intenclr.

Table 7-1 interrupt control registers

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | Access properties/default values | | | | |
| Intedge | Inten | Intenset | Intenclr | The interrupt source |
| 3:0 | RW / 0 | R / 0 | W / 0 | W / 0 | Sys\_int0-3 |
| 7:4 | RO / 0 | R / 0 | RW / 0 | RW / 0 | PCI\_INTn |
| 8 | RO / 0 | R / 0 | RW / 0 | RW / 0 | Matrix\_int0 |
| 9 | RO / 1 | R / 0 | RW / 0 | RW / 0 | Matrix\_int1 |
| 10 | RO / 1 | R / 0 | RW / 0 | RW / 0 | Lpc |
| 12:11 | RW / 0 | reserve | reserve | reserve | Mc0-1 |
| 13 | RW / 0 | R / 0 | RW / 0 | RW / 0 | The Barrier |
| 14 | RW / 0 | R / 0 | RW / 0 | RW / 0 | Thsens int |
| 15 | RW / 0 | R / 0 | RW / 0 | RW / 0 | Pci\_perr |
| 23:16 | RW / 0 | R / 0 | RW / 0 | RW / 0 | HT0 int0-7 |
| 31:24 | RW / 0 | R / 0 | RW / 0 | RW / 0 | HT1 int0-7 |

Table 7-2 IO control register addresses

|  |  |  |
| --- | --- | --- |
| The name of the | Address offset | describe |
| Intisr | 0 x3ff01420 | 32-bit interrupt status register |
| Inten | 0 x3ff01424 | 32-bit interrupt enabled status register |
| Intenset | 0 x3ff01428 | The 32-bit setting enables the register |
| Intenclr | 0 x3ff0142c | 32-bit clear enable register |
| Intedge | 0 x3ff01438 | 32-bit trigger mode register |
| CORE0\_INTISR | 0 x3ff01440 | Routing to the 32-bit interrupt state of CORE0 |
| CORE1\_INTISR | 0 x3ff01448 | Routing a 32-bit interrupt state to CORE1 |
| CORE2\_INTISR | 0 x3ff01450 | Routing a 32-bit interrupt state to CORE2 |
| CORE3\_INTISR | 0 x3ff01458 | Routing a 32-bit interrupt state to CORE3 |

With four processor cores integrated into the loson 3A2000, the 32-bit interrupt source can be configured to select the target processor core to interrupt. Further, the interrupt source can choose to route to any of the processor core interrupts INT0 through INT3, IP2 through IP5 corresponding to CP0\_Status.Each of the 32 I/O interrupt sources corresponds to an 8-bit routing controller, whose format and address are shown in tables 7-3 and 7-4 below. The routing register USES a vector approach for routing, such as 0x48 to indicate routing to INT2 of processor no. 3.

Table 7-3 description of interrupt routing registers

|  |  |
| --- | --- |
| A domain | Said Ming |
| 3-0 | Routing the processor core vector number |
| The log | Routing the processor core interrupt pin vector number |

Table 7-4 interrupt routing register addresses

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| The name of the | Address offset | describe | The name of the | Address offset | describe |
| Entry0 | 0 x3ff01400 | Sys\_int0 | Entry16 | 0 x3ff01410 | HT0 - int0 |
| Entry1 | 0 x3ff01401 | Sys\_int1 | Entry17 | 0 x3ff01411 | HT0 - int1 |
| Entry2 | 0 x3ff01402 | Sys\_int2 | Entry18 | 0 x3ff01412 | HT0 - int2 |
| Entry3 | 0 x3ff01403 | Sys\_int3 | Entry19 | 0 x3ff01413 | HT0 - int3 |
| Entry4 | 0 x3ff01404 | Pci\_int0 | Entry20 | 0 x3ff01414 | HT0 - int4 |
| Entry5 | 0 x3ff01405 | Pci\_int1 | Entry21 | 0 x3ff01415 | HT0 - int5 |
| Entry6 | 0 x3ff01406 | Pci\_int2 | Entry22 | 0 x3ff01416 | HT0 - int6 |
| Entry7 | 0 x3ff01407 | Pci\_int3 | Entry23 | 0 x3ff01417 | HT0 - int7 |
| Entry8 | 0 x3ff01408 | Matrix int0 | Entry24 | 0 x3ff01418 | HT1 - int0 |
| Entry9 | 0 x3ff01409 | Matrix int1 | Entry25 | 0 x3ff01419 | HT1 - int1 |
| Entry10 | 0 x3ff0140a | Lpc int | Entry26 | 0 x3ff0141a | HT1 - int2 |
| Entry11 | 0 x3ff0140b | Mc0 | Entry27 | 0 x3ff0141b | HT1 - int3 |
| Entry12 | 0 x3ff0140c | Mc1 | Entry28 | 0 x3ff0141c | HT1 - int4 |
| Entry13 | 0 x3ff0140d | The Barrier | Entry29 | 0 x3ff0141d | HT1 - int5 |
| Entry14 | 0 x3ff0140e | Thsens int | Entry30 | 0 x3ff0141e | HT1 - int6 |
| Entry15 | 0 x3ff0140f | Pci\_perr/serr | Entry31 | 0 x3ff0141f | HT1 - int7 |

## Temperature sensor

#### Real-time temperature sampling

The godson 3A2000 is internally integrated with two temperature sensors, which can be observed through the sampling register starting from 0x1FE00198. At the same time, it can be controlled by flexible high-low temperature interrupt alarm or automatic FM function. The corresponding bit of the temperature sensor in the sampling register is as follows (base address 0x1FE00198) :

Table 8-1 temperature sampling register description

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **The field name** | | **access** | **Reset value** | **describe** |
| 24 | | Thsens0\_overflow | R |  | Overflow of temperature sensor 0 (over 125℃) |
| 25 | | Thsens1\_overflow | R |  | Overflow of temperature sensor 1 (over 125℃) |
| 39:32 | | Thsens0\_out | R |  | Temperature sensor: 0 ℃  Node temperature = thens0\_out-100 temperature range -40 ° -125 ° |
| 47:40 | | Thsens1\_out | R |  | Temperature sensor 1 ℃ temperature  Node temperature = thens1\_out-100 temperature range -40 ° -125 ° |
| other | |  | R |  | reserve |

By setting the control register, it can realize the functions of interrupting above the preset temperature, interrupting below the preset temperature and automatically lowering the frequency at high temperature.

#### High and low temperature interrupt triggered

For the high and low temperature interrupt alarm function, there are four groups of control registers to set the threshold value. Each set of registers contains the following three control bits:

GATE: sets the threshold of high or low temperature. When the input temperature is higher than the high temperature threshold or lower than the low temperature threshold, an interrupt will occur.

EN: interrupt enable control. The set of registers is only valid after setting 1;

SEL: input temperature selection. The register is used to configure which sensor's temperature is selected as the input. You can use either a 0 or a 1.

The high temperature interrupt control register contains four sets of Settings for controlling the high temperature interrupt trigger. Low temperature interrupt control register

It contains four sets of Settings to control the cold interrupt triggering. There is also a set of registers to display the interrupt status, corresponding to high temperature interrupts and low temperature interrupts, and any write to this register will clear the interrupt status.

The specific description of these registers is as follows:

Table 8-2 high and low temperature interrupt registers

|  |  |  |  |
| --- | --- | --- | --- |
| register | address | control | instructions |
| High temperature interrupt control register  Thsens\_int\_ctrl\_Hi | 0 x3ff01460 | RW | [7:0] : Hi\_gate0: high temperature threshold 0 above which an interrupt will occur [8:8] : Hi\_en0: high temperature interrupt enable 0  [11:10] : Hi\_Sel0: select high temperature interrupt 0 as the temperature sensor input source [23:16] : Hi\_gate1: high temperature threshold 1 above which interruption will occur [24:24] : Hi\_en1: high temperature interrupt enable 1  [27:26] : Hi\_Sel1: select high temperature interrupt 1 for the temperature sensor input source [39:32] : Hi\_gate2: high temperature threshold 2, beyond which there will be an interrupt [40:40] : Hi\_en2: high temperature interrupt enable 2  [43:42] : Hi\_Sel2: select high temperature interrupt 2 as the temperature sensor input source [55:48] : Hi\_gate3: high temperature threshold 3, beyond which there will be an interrupt [56:56] : Hi\_en3: high temperature interrupt enable 3  [59:58] : Hi\_Sel3: select the temperature sensor input source of high temperature interrupt 3 |
| Low temperature interrupt control register  Thsens\_int\_ctrl\_Lo | 0 x3ff01468 | RW | [7:0] : Lo\_gate0: low temperature threshold 0 below which there will be an interrupt [8:8] : Lo\_en0: low temperature interrupt enable 0  [11:10] : Lo\_Sel0: select the temperature sensor input source of cryogenic interrupt 0 [23:16] : Lo\_gate1: cryogenic threshold 1 below which there will be an interrupt [24:24] : Lo\_en1: cryogenic interrupt enable 1  [27:26] : Lo\_Sel1: select the temperature sensor input source of cryogenic interrupt 1 [39:32] : Lo\_gate2: cryogenic threshold 2, below which there will be an interrupt [40:40] : Lo\_en2: cryogenic interrupt enable 2  [43:42] : Lo\_Sel2: select the temperature sensor input source for cold interrupt 2 [55:48] : Lo\_gate3: cold threshold 3, below which an interrupt [56:56] will occur: Lo\_en3: cold interrupt enable 3  [59:58] : Lo\_Sel3: select the temperature sensor input source of cryogenic interrupt 3 |
| Interrupt status register  Thsens\_int\_status/CLR | 0 x3ff01470 | RW | Interrupt status register, write any value to clear the interrupt  [0] : high temperature interrupt trigger [1] : low temperature interrupt trigger |

#### High temperature automatic frequency down setting

In order to ensure the operation of the chip in high temperature environment, the chip can be set to automatically lower the frequency at high temperature, so that the chip can actively carry out clock frequency division when it exceeds the preset range, so as to reduce the chip turnover rate.

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

GATE: sets the threshold of high or low temperature. It is triggered when the input temperature is above or below the high temperature threshold

Frequency division operation;

EN: interrupt enable control. The set of registers is only valid after setting 1;

SEL: input temperature selection. The register is used to configure which sensor's temperature is selected as the input. You can use either a 0 or a 1.

FREQ: frequency division. When triggering a frequency division operation, adjust the frequency to FREQ/8 times the current clock frequency.  Table 8-3 high

|  |  |  |  |
| --- | --- | --- | --- |
| register | address | control | instructions |
| High temperature frequency down control register  Thsens\_freq\_scale | 0 x3ff01480 | RW | Four groups set priority from high to low  [7:0] : Scale\_gate0: high temperature threshold 0, above which the frequency will be reduced [8:8] : Scale\_en0: high temperature frequency reduction enables 0  Scale\_Sel0: select the temperature sensor input source of high temperature frequency reduction 0 [14:12] : Scale\_freq0: frequency separation value when frequency reduction [23:16] : Scale\_gate1: high temperature threshold 1, beyond which the frequency will be reduced [24:24] : Scale\_en1: high temperature frequency reduction enable 1  Scale\_Sel1: select the temperature sensor input source of high temperature frequency reduction 1 [30:28] : Scale\_freq1: frequency separation value when frequency reduction [39:32] : Scale\_gate2: high temperature threshold 2, above which the frequency will be reduced [40:40] : Scale\_en2: high temperature frequency reduction enable 2  Scale\_Sel2: select the temperature sensor input source of high temperature frequency reduction 2 [46:44] : Scale\_freq2: frequency separation value when frequency reduction [55:48] : Scale\_gate3: high temperature threshold 3, above which frequency reduction [56:56] : Scale\_en3: high temperature frequency reduction enable 3  [59:58] : Scale\_Sel3: select the temperature sensor input source with high temperature down frequency 3  [62:60] : Scale\_freq3: frequency division value when frequency is down |

## Ddr2/3 SDRAM controller configuration

The integrated memory controller inside the loongson 3 processor is designed to comply with the ddr2/3 SDRAM industry standard (jesd79-2 and jesd79-3). In the loongson 3 processor, all memory read/write operations are implemented in accordance with jesd79-2b and jesd79-3.

#### Ddr2/3 SDRAM controller features overview

The loongson 3 processor supports up to 4 CS (implemented by 4 DDR2 SDRAM chip selectors, namely two double-sided memory strips) and contains a total of 19 bit address bus (i.e., 16 bit line and column address bus and 3 bit logical Bank bus).

Loongson 3 processor can adjust the parameter setting of ddr2/3 controller to support the use of different memory chip types. Among them, the maximum number of supported block selections (CS\_n) is 4, the number of row addresses (RAS\_n) is 16, and the number of column addresses

The number (CAS\_n) is 15, and the logical body choice (BANK\_n) is 3.

The physical address of the memory request sent by the CPU can be mapped to a variety of different addresses according to different configurations within the controller.

The memory control circuit integrated by the loongson 3 processor only accepts memory read/write requests from the processor or external devices. In all memory read/write operations, the memory control circuit is in Slave State.

The memory controller in loongson 3 processor has the following characteristics:

* + - Interface command, read and write data full flow operation
    - Memory command merge, sort to improve the overall bandwidth
    - Configure the register read-write port to modify the basic parameters of the memory device
    - Built - in dynamic delay compensation circuit (DCC) for reliable data sending and receiving
    - The ECC function can detect 1 - and 2-bit errors in the data path and automatically correct 1 - bit errors
    - Support 133-667mhz operating frequency

#### Ddr2/3 SDRAM read operation protocol

The protocol for ddr2/3 SDRAM read operations is shown in figure 11-2. In the figure, the Command (CMD) consists of RAS\_n, CAS\_n and WE\_n. For read operations, RAS\_n=1, CAS\_n = 0, and WE\_n =1.

‘

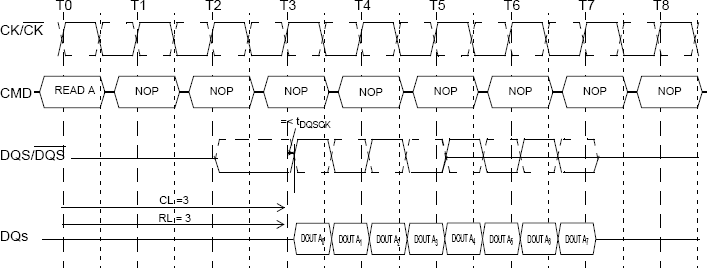


Figure 9-1 DDR2 SDRAM read operation protocol

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

## Ddr2/3 SDRAM write operation protocol

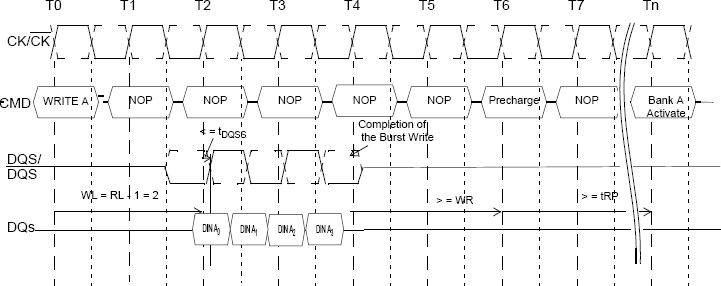
The protocol for ddr2/3 SDRAM write operations is shown in figure 11-3. In the figure, the command CMD is composed of RAS\_n, CAS\_n, and WE\_n. For write operations, RAS\_n=1, CAS\_n = 0, and WE\_n = 0. Also, unlike read operations, write operations require DQM to identify the mask of the write operation, that is, the number of bytes to write.DQM synchronizes with DQs signal in the figure.

Figure 9-2 DDR2 SDRAM write operation protocol

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

#### Ddr2/3 SDRAM parameter configuration format

The visible parameter list and description of the memory controller software are shown in the following table:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 63:56 | | 55:48 | | 47:40 | 39:32 | | came | | Ephron; | | | " | away | |
| x000 | Dll\_value\_0 / Dll\_adj\_cnt (RD) | | | | Dll\_value\_ck (RD) | | | Dll\_init\_done (RD) | | | | | Version (RD) | | |
| x008 | Dll\_value\_4 (RD) | | | | Dll\_value\_3 (RD) | | | Dll\_value\_2 (RD) | | | | | Dll\_value\_1 (RD) | | |
| x010 | Dll\_value\_8 (RD) | | | | Dll\_value\_7 (RD) | | | Dll\_value\_6 (RD) | | | | | Dll\_value\_5 (RD) | | |
| x018 | Dll\_ck\_3 | | Dll\_ck\_2 | | Dll\_ck\_1 | Dll\_ck\_0 | | Dll\_increment | | Dll\_start\_point | | | Dll\_bypass | Init\_start | |
| x020 | Dq\_oe\_end\_0 | | Dq\_oe\_begin\_0 | | Dq\_stop\_edge\_0 | Dq\_start\_edge\_0 | | Rddata\_delay\_0 | | Rddqs\_lt\_half\_0 | | | Wrdqs\_lt\_half\_0 | Wrdq\_lt\_half\_0 | |
| x028 | Rd\_oe\_end\_0 | | Rd\_oe\_begin\_0 | | Rd\_stop\_edge\_0 | Rd\_start\_edge\_0 | | Dqs\_oe\_end\_0 | | Dqs\_oe\_begin\_0 | | | Dqs\_stop\_edge\_0 | Dqs\_start\_edge\_0 | |
| x030 | Enzi\_end\_0 | | Enzi\_begin\_0 | | Wrclk\_sel\_0 | Wrdq\_clkdelay\_0 | | Odt\_oe\_end\_0 | | Odt\_oe\_begin\_0 | | | Odt\_stop\_edge\_0 | Odt\_start\_edge\_0 | |
| x038 | Enzi\_stop\_0 | | Enzi\_start\_0 | |  | Dll\_rddqs\_n\_0 | | Dll\_rddqs\_p\_0 | | Dll\_wrdqs\_0 | | | Dll\_wrdata\_0 | Dll\_gate\_0 | |
| x040 | Dq\_oe\_end\_1 | | Dq\_oe\_begin\_1 | | Dq\_stop\_edge\_1 | Dq\_start\_edge\_1 | | Rddata\_delay\_1 | | Rddqs\_lt\_half\_1 | | | Wrdqs\_lt\_half\_1 | Wrdq\_lt\_half\_1 | |
| x048 | Rd\_oe\_end\_1 | | Rd\_oe\_begin\_1 | | Rd\_stop\_edge\_1 | Rd\_start\_edge\_1 | | Dqs\_oe\_end\_1 | | Dqs\_oe\_begin\_1 | | | Dqs\_stop\_edge\_1 | Dqs\_start\_edge\_1 | |
| x050 | Enzi\_end\_1 | | Enzi\_begin\_1 | | Wrclk\_sel\_1 | Wrdq\_clkdelay\_1 | | Odt\_oe\_end\_1 | | Odt\_oe\_begin\_1 | | | Odt\_stop\_edge\_1 | Odt\_start\_edge\_1 | |
| x058 | Enzi\_stop\_1 | | Enzi\_start\_1 | |  | Dll\_rddqs\_n\_1 | | Dll\_rddqs\_p\_1 | | Dll\_wrdqs\_1 | | | Dll\_wrdata\_1 | Dll\_gate\_1 | |
| x060 | Dq\_oe\_end\_2 | | Dq\_oe\_begin\_2 | | Dq\_stop\_edge\_2 | Dq\_start\_edge\_2 | | Rddata\_delay\_2 | | Rddqs\_lt\_half\_2 | | | Wrdqs\_lt\_half\_2 | Wrdq\_lt\_half\_2 | |
| x068 | Rd\_oe\_end\_2 | | Rd\_oe\_begin\_2 | | Rd\_stop\_edge\_2 | Rd\_start\_edge\_2 | | Dqs\_oe\_end\_2 | | Dqs\_oe\_begin\_2 | | | Dqs\_stop\_edge\_2 | Dqs\_start\_edge\_2 | |
| x070 | Enzi\_end\_2 | | Enzi\_begin\_2 | | Wrclk\_sel\_2 | Wrdq\_clkdelay\_2 | | Odt\_oe\_end\_2 | | Odt\_oe\_begin\_2 | | | Odt\_stop\_edge\_2 | Odt\_start\_edge\_2 | |
| x078 | Enzi\_stop\_2 | | Enzi\_start\_2 | |  | Dll\_rddqs\_n\_2 | | Dll\_rddqs\_p\_2 | | Dll\_wrdqs\_2 | | | Dll\_wrdata\_2 | Dll\_gate\_2 | |
| x080 | Dq\_oe\_end\_3 | | Dq\_oe\_begin\_3 | | Dq\_stop\_edge\_3 | Dq\_start\_edge\_3 | | Rddata\_delay\_3 | | Rddqs\_lt\_half\_3 | | | Wrdqs\_lt\_half\_3 | Wrdq\_lt\_half\_3 | |
| x088 | Rd\_oe\_end\_3 | | Rd\_oe\_begin\_3 | | Rd\_stop\_edge\_3 | Rd\_start\_edge\_3 | | Dqs\_oe\_end\_3 | | Dqs\_oe\_begin\_3 | | | Dqs\_stop\_edge\_3 | Dqs\_start\_edge\_3 | |
| x090 | Enzi\_end\_3 | | Enzi\_begin\_3 | | Wrclk\_sel\_3 | Wrdq\_clkdelay\_3 | | Odt\_oe\_end\_3 | | Odt\_oe\_begin\_3 | | | Odt\_stop\_edge\_3 | Odt\_start\_edge\_3 | |
| x098 | Enzi\_stop\_3 | | Enzi\_start\_3 | |  | Dll\_rddqs\_n\_3 | | Dll\_rddqs\_p\_3 | | Dll\_wrdqs\_3 | | | Dll\_wrdata\_3 | Dll\_gate\_3 | |
| x0A0 | Dq\_oe\_end\_4 | | Dq\_oe\_begin\_4 | | Dq\_stop\_edge\_4 | Dq\_start\_edge\_4 | | Rddata\_delay\_4 | | Rddqs\_lt\_half\_4 | | | Wrdqs\_lt\_half\_4 | Wrdq\_lt\_half\_4 | |
| X0A8. | Rd\_oe\_end\_4 | | Rd\_oe\_begin\_4 | | Rd\_stop\_edge\_4 | Rd\_start\_edge\_4 | | Dqs\_oe\_end\_4 | | Dqs\_oe\_begin\_4 | | | Dqs\_stop\_edge\_4 | Dqs\_start\_edge\_4 | |
| x0B0 | Enzi\_end\_4 | | Enzi\_begin\_4 | | Wrclk\_sel\_4 | Wrdq\_clkdelay\_4 | | Odt\_oe\_end\_4 | | Odt\_oe\_begin\_4 | | | Odt\_stop\_edge\_4 | Odt\_start\_edge\_4 | |
| x0B8 | Enzi\_stop\_4 | | Enzi\_start\_4 | |  | Dll\_rddqs\_n\_4 | | Dll\_rddqs\_p\_4 | | Dll\_wrdqs\_4 | | | Dll\_wrdata\_4 | Dll\_gate\_4 | |
| x0C0 | Dq\_oe\_end\_5 | | Dq\_oe\_begin\_5 | | Dq\_stop\_edge\_5 | Dq\_start\_edge\_5 | | Rddata\_delay\_5 | | Rddqs\_lt\_half\_5 | | | Wrdqs\_lt\_half\_5 | Wrdq\_lt\_half\_5 | |
| x0C8 | Rd\_oe\_end\_5 | | Rd\_oe\_begin\_5 | | Rd\_stop\_edge\_5 | Rd\_start\_edge\_5 | | Dqs\_oe\_end\_5 | | Dqs\_oe\_begin\_5 | | | Dqs\_stop\_edge\_5 | Dqs\_start\_edge\_5 | |
| x0D0 | Enzi\_end\_5 | | Enzi\_begin\_5 | | Wrclk\_sel\_5 | Wrdq\_clkdelay\_5 | | Odt\_oe\_end\_5 | | Odt\_oe\_begin\_5 | | | Odt\_stop\_edge\_5 | Odt\_start\_edge\_5 | |
| x0D8 | Enzi\_stop\_5 | | Enzi\_start\_5 | |  | Dll\_rddqs\_n\_5 | | Dll\_rddqs\_p\_5 | | Dll\_wrdqs\_5 | | | Dll\_wrdata\_5 | Dll\_gate\_5 | |
| x0E0 | Dq\_oe\_end\_6 | | Dq\_oe\_begin\_6 | | Dq\_stop\_edge\_6 | Dq\_start\_edge\_6 | | Rddata\_delay\_6 | | Rddqs\_lt\_half\_6 | | | Wrdqs\_lt\_half\_6 | Wrdq\_lt\_half\_6 | |
| x0E8 | Rd\_oe\_end\_6 | | Rd\_oe\_begin\_6 | | Rd\_stop\_edge\_6 | Rd\_start\_edge\_6 | | Dqs\_oe\_end\_6 | | Dqs\_oe\_begin\_6 | | | Dqs\_stop\_edge\_6 | Dqs\_start\_edge\_6 | |
| x0F0 | Enzi\_end\_6 | | Enzi\_begin\_6 | | Wrclk\_sel\_6 | Wrdq\_clkdelay\_6 | | Odt\_oe\_end\_6 | | Odt\_oe\_begin\_6 | | | Odt\_stop\_edge\_6 | Odt\_start\_edge\_6 | |
| x0F8 | Enzi\_stop\_6 | | Enzi\_start\_6 | |  | Dll\_rddqs\_n\_6 | | Dll\_rddqs\_p\_6 | | Dll\_wrdqs\_6 | | | Dll\_wrdata\_6 | Dll\_gate\_6 | |
| x100 | Dq\_oe\_end\_7 | | Dq\_oe\_begin\_7 | | Dq\_stop\_edge\_7 | Dq\_start\_edge\_7 | | Rddata\_delay\_7 | | Rddqs\_lt\_half\_7 | | | Wrdqs\_lt\_half\_7 | Wrdq\_lt\_half\_7 | |
| x108 | Rd\_oe\_end\_7 | | Rd\_oe\_begin\_7 | | Rd\_stop\_edge\_7 | Rd\_start\_edge\_7 | | Dqs\_oe\_end\_7 | | Dqs\_oe\_begin\_7 | | | Dqs\_stop\_edge\_7 | Dqs\_start\_edge\_7 | |
| x110 | Enzi\_end\_7 | | Enzi\_begin\_7 | | Wrclk\_sel\_7 | Wrdq\_clkdelay\_7 | | Odt\_oe\_end\_7 | | Odt\_oe\_begin\_7 | | | Odt\_stop\_edge\_7 | Odt\_start\_edge\_7 | |
| x118 | Enzi\_stop\_7 | | Enzi\_start\_7 | |  | Dll\_rddqs\_n\_7 | | Dll\_rddqs\_p\_7 | | Dll\_wrdqs\_7 | | | Dll\_wrdata\_7 | Dll\_gate\_7 | |
|  | 63:56 | | 55:48 | | 47:40 | 39:32 | | came | | Ephron; | | | " | away | | |
| X120 measures how | Dq\_oe\_end\_8 | | Dq\_oe\_begin\_8 | | Dq\_stop\_edge\_8 | Dq\_start\_edge\_8 | | Rddata\_delay\_8 | | Rddqs\_lt\_half\_8 | | | Wrdqs\_lt\_half\_8 | Wrdq\_lt\_half\_8 | | |
| x128 | Rd\_oe\_end\_8 | | Rd\_oe\_begin\_8 | | Rd\_stop\_edge\_8 | Rd\_start\_edge\_8 | | Dqs\_oe\_end\_8 | | Dqs\_oe\_begin\_8 | | | Dqs\_stop\_edge\_8 | Dqs\_start\_edge\_8 | | |
| x130 | Enzi\_end\_8 | | Enzi\_begin\_8 | | Wrclk\_sel\_8 | Wrdq\_clkdelay\_8 | | Odt\_oe\_end\_8 | | Odt\_oe\_begin\_8 | | | Odt\_stop\_edge\_8 | Odt\_start\_edge\_8 | | |
| x138 | Enzi\_stop\_8 | | Enzi\_start\_8 | |  | Dll\_rddqs\_n\_8 | | Dll\_rddqs\_p\_8 | | Dll\_wrdqs\_8 | | | Dll\_wrdata\_8 | Dll\_gate\_8 | | |
| x140 | Pad\_ocd\_clk | | Pad\_ocd\_ctl | | Pad\_ocd\_dqs | Pad\_ocd\_dq | | Pad\_enzi | | | | | Pad\_en\_ctl | Pad\_en\_clk | | |
| x148 | Pad\_adj\_code\_dqs | | Pad\_code\_dqs | | Pad\_adj\_code\_dq | Pad\_code\_dq | |  | | Pad\_vref\_internal | | | Pad\_odt\_se | Pad\_modezi1v8 | | |
| x150 |  | | Pad\_reset\_po | | Pad\_adj\_code\_clk | Pad\_code\_lk | | Pad\_adj\_code\_cmd | | Pad\_code\_cmd | | | Pad\_adj\_code\_addr | Pad\_code\_addr | | |
| x158 |  | | Pad\_comp\_code\_o | | Pad\_comp\_okn | Pad\_comp\_code\_i | | | | Pad\_comp\_mode | | | Pad\_comp\_tm | Pad\_comp\_pd | | |
| x160 | Rdfifo\_empty (RD) | | | | Overflow (RD) | | | Dram\_init (RD) | | Rdfifo\_valid | | | Cmd\_timming | Ddr3\_mode | | |
| x168 | Ba\_xor\_row\_offset | | Addr\_mirror | | Cmd\_delay | Burst\_length | | Bank | | Cs\_zq | | | Cs\_mrs | Cs\_enable | | |
| x170 | Odt\_wr\_cs\_map | | | | Odt\_wr\_length | Odt\_wr\_delay | | Odt\_rd\_cs\_map | | | | | Odt\_rd\_length | Odt\_rd\_delay | | |
| x178 |  | |  | |  |  | |  | |  | | |  |  | | |
| x180 | Lvl\_resp\_0 (RD) | | Lvl\_done (RD) | | Lvl\_ready (RD) |  | | Lvl\_cs | | tLVL\_DELAY | | | Lvl\_req (WR) | Lvl\_mode | | |
| x188 | Lvl\_resp\_8 (RD) | | Lvl\_resp\_7 (RD) | | Lvl\_resp\_6 (RD) | Lvl\_resp\_5 (RD) | | Lvl\_resp\_4 (RD) | | Lvl\_resp\_3 (RD) | | | Lvl\_resp\_2 (RD) | Lvl\_resp\_1 (RD) | | |
| x190 | Cmd\_a | | | | Cmd\_ba | Cmd\_cmd | | Cmd\_cs | | Status\_cmd (RD) | | | Cmd\_req (WR) | Command\_mode | | |
| x198 |  | |  | | Status\_sref (RD) | Srefresh\_req | | Pre\_all\_done (RD) | | Pre\_all\_req (RD) | | | Mrs\_done (RD) | Mrs\_req (WR) | | |
| x1A0 | Mr\_3\_cs\_0 | | | | Mr\_2\_cs\_0 | | | Mr\_1\_cs\_0 | | | | | Mr\_0\_cs\_0 | | | |
| x1A8 | Mr\_3\_cs\_1 | | | | Mr\_2\_cs\_1 | | | Mr\_1\_cs\_1 | | | | | Mr\_0\_cs\_1 | | | |
| x1B0 | Mr\_3\_cs\_2 | | | | Mr\_2\_cs\_2 | | | Mr\_1\_cs\_2 | | | | | Mr\_0\_cs\_2 | | | |
| x1B8 | Mr\_3\_cs\_3 | | | | Mr\_2\_cs\_3 | | | Mr\_1\_cs\_3 | | | | | Mr\_0\_cs\_3 | | | |
| x1C0 | tRESET | | tCKE | | tXPR | tMOD | | tZQCL | | tZQ\_CMD | | | tWLDQSEN | | tRDDATA | |
| x1C8 | tFAW | | tRRD | | tRCD | tRP | | tREF | | tRFC | | | tZQCS | | tZQperiod | |
| x1D0 | tODTL | | tXSRD | | tPHY\_RDLAT | tPHY\_WRLAT | | tRAS\_max | | | | | | | tRAS\_min | |
| x1D8 | tXPDLL | | tXP | | tWR | tRTP | | tRL | | these | | tCCD | | | tWTR | |
| x1E0 | tW2R\_diffCS | | tW2W\_diffCS | | tR2P\_sameBA | tW2P\_sameBA | | tR2R\_sameBA | | tR2W\_sameBA | | tW2R\_sameBA | | | tW2W\_sameBA | |
| x1E8 | tR2R\_diffCS | | tR2W\_diffCS | | tR2P\_sameCS | tW2P\_sameCS | | tR2R\_sameCS | | tR2W\_sameCS | | tW2R\_sameCS | | | tW2W\_sameCS | |
| x1F0 | Power\_up | | Age\_step | | tCPDED | Cs\_map | | Bs\_config | | Nc | | Pr\_r2w | | | Placement\_en | |
| x1F8 | Hw\_pd\_3 | | Hw\_pd\_2 | | Hw\_pd\_1 | Hw\_pd\_0 | | Credit\_16 | | Credit\_32 | | Credit\_64 | | | Selection\_en | |
| x200 | Cmdq\_age\_16 | | | | Cmdq\_age\_32 | | | Cmdq\_age\_64 | | | | tCKESR | | | tRDPDEN | |
| x208 | Wfifo\_age | | | | Rfifo\_age | | | Power\_stat3 | | Power\_stat2 | | Power\_stat1 | | | Power\_stat0 | |
| x210 | Active\_age | | | | Cs\_place\_0 | Addr\_win\_0 | | Cs\_diff\_0 | | Row\_diff\_0 | | Ba\_diff\_0 | | | Col\_diff\_0 | |
| x218 | Fastpd\_age | | | | Cs\_place\_1 | Addr\_win\_1 | | Cs\_diff\_1 | | Row\_diff\_1 | | Ba\_diff\_1 | | | Col\_diff\_1 | |
| x220 | Slowpd\_age | | | | Cs\_place\_2 | Addr\_win\_2 | | Cs\_diff\_2 | | Row\_diff\_2 | | Ba\_diff\_2 | | | Col\_diff\_2 | |
| x228 | Selfref\_age | | | | Cs\_place\_3 | Addr\_win\_3 | | Cs\_diff\_3 | | Row\_diff\_3 | | Ba\_diff\_3 | | | Col\_diff\_3 | |
| x230 | Win\_mask\_0 | | | | | | | Win\_base\_0 | | | | | | | | |
| x238 | Win\_mask\_1 | | | | | | | Win\_base\_1 | | | | | | | | |
| x240 | Win\_mask\_2 | | | | | | | Win\_base\_2 | | | | | | | | |
| x248 | Win\_mask\_3 | | | | | | | Win\_base\_3 | | | | | | | | |
| x250 |  | | Cmd\_monitor | Axi\_monitor | | | | Ecc\_code (RD) | | Ecc\_enable | Int\_vector | | | Int\_enable | | |
|  | 63:56 | | 55:48 | | 47:40 | 39:32 | | came | | Ephron; | | | " | away | |
| x258 |  | |  |  | |  | |  | |  |  | | |  | |
| x260 | Ecc\_addr (RD) | | | | | | | | | | | | | | |
| x268 | Ecc\_data (RD) | | | | | | | | | | | | | | |
| x270 | Lpbk\_ecc\_mask (RD) | | Prbs\_init | | | | Lpbk\_error (RD) | | | Prbs\_23 | Lpbk\_start | | | Lpbk\_en | |
| x278 | Lpbk\_ecc (RD) | | | Lpbk\_data\_mask (RD) | | | Lpbk\_correct (RD) | | | | Lpbk\_counter (RD) | | | | |
| x280 | Lpbk\_data\_r (RD) | | | | | | | | | | | | | | |
| x288 | Lpbk\_data\_f (RD) | | | | | | | | | | | | | | |
| x290 | Axi0\_bandwidth\_w | | | | | | Axi0\_bandwidth\_r | | | | | | | | |
| x298 | Axi0\_latency\_w | | | | | | Axi0\_latency\_r | | | | | | | | |
| x2A0 | Axi1\_bandwidth\_w | | | | | | Axi1\_bandwidth\_r | | | | | | | | |
| x2A8 | Axi1\_latency\_w | | | | | | Axi1\_latency\_r | | | | | | | | |
| x2B0 | Axi2\_bandwidth\_w | | | | | | Axi2\_bandwidth\_r | | | | | | | | |
| x2B8 | Axi2\_latency\_w | | | | | | Axi2\_latency\_r | | | | | | | | |
| x2C0 | Axi3\_bandwidth\_w | | | | | | Axi3\_bandwidth\_r | | | | | | | | |
| x2C8 | Axi3\_latency\_w | | | | | | Axi3\_latency\_r | | | | | | | | |
| x2D0 | Axi4\_bandwidth\_w | | | | | | Axi4\_bandwidth\_r | | | | | | | | |
| x2D8 | Axi4\_latency\_w | | | | | | Axi4\_latency\_r | | | | | | | | |
| x2E0 | Cmdq0\_bandwidth\_w | | | | | | Cmdq0\_bandwidth\_r | | | | | | | | |
| x2E8 | Cmdq0\_latency\_w | | | | | | Cmdq0\_latency\_r | | | | | | | | |
| x2F0 | Cmdq1\_bandwidth\_w | | | | | | Cmdq1\_bandwidth\_r | | | | | | | | |
| x2F8 | Cmdq1\_latency\_w | | | | | | Cmdq1\_latency\_r | | | | | | | | |
| Adaptive: | Cmdq2\_bandwidth\_w | | | | | | Cmdq2\_bandwidth\_r | | | | | | | | |
| x308 | Cmdq2\_latency\_w | | | | | | Cmdq2\_latency\_r | | | | | | | | |
| x310 | Cmdq3\_bandwidth\_w | | | | | | Cmdq3\_bandwidth\_r | | | | | | | | |
| x318 | Cmdq3\_latency\_w | | | | | | Cmdq3\_latency\_r | | | | | | | | |
| x320 |  |  | |  | |  |  | |  | |  | | | tREF\_low | |
| x328 |  |  | |  | |  |  | |  | |  | | |  | |
| x330 | Stat\_en | Rdbuffer\_max | | Retry | | Wr\_pkg\_num | Rwq\_rb | | Stb\_en | | Addr\_new | | | tRDQidle | |
| x338 |  |  | |  | | Rd\_fifo\_depth | Retry\_cnt | | | | | | | | |
| x340 | tREFretention | | | | | |  | | Ref\_num | | tREF\_IDLE | | | Ref\_sch\_en | |
| x348 |  |  | |  | |  |  | |  | |  | | |  | |
| x350 | Lpbk\_data\_en | | | | | | | | | | | | | | |
| x358 |  |  | |  | |  |  | | Lpbk\_ecc\_mask\_en | | Lpbk\_ecc\_en | | | Lpbk\_data\_mask\_en | |
| x360 |  |  | | Int\_ecc\_cnt\_fatal | | Int\_ecc\_cnt\_err  The or | Ecc\_cnt\_cs\_3 | | Ecc\_cnt\_cs\_2 | | Ecc\_cnt\_cs\_1 | | | Ecc\_cnt\_cs\_0 | |
| x368 |  |  | |  | |  |  | |  | |  | | |  | |

#### Software programming guide

#### Initialization operation

Initialization begins when the software writes 1 to register Init\_start (0x018), setting Init\_start

All other registers must be set to the correct value before signaling.The DRAM initialization process of software/hardware collaboration is as follows:

* + - 1. The software writes the correct configuration values to all registers, but Init\_start (0x018) must remain at 0 during this process;
      2. The software sets Init\_start (0x018) to 1, which results in the start of hardware initialization.
      3. The initialization begins internally at PHY, and the DLL will attempt to lock.If the lock is successful, the corresponding state can be read from Dll\_init\_done (0x000) and the number of current lock delay lines can be read and written from Dll\_value\_ck (0x000).If the lock is not successful, the initialization will not continue (this can be done by setting Dll\_bypass (0x018));
      4. After DLL lock (or bypass setting), the controller will be directed to the DRAM according to the initialization requirements of the corresponding DRAM

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

* + - 1. The software can determine if the memory initialization operation is complete by sampling the Dram\_init (0x160) register.

#### Reset pin control

In order to control the reset pin more easily in the state of STR and so on, special reset pin (DDR\_RESETn) control can be carried out through the reset\_ctrl (0x150) register. There are two main control modes:

In general mode, reset\_ctrl[1:0] == 2 'b00.In this mode, the reset signal pin behavior is compatible with the general control mode. DDR\_RESETn is directly connected to the corresponding pin on the memory slot on the motherboard. The behavior of the pin is:

* When the power is not on: the pin state is low;
* When power on: the pin state is low;
* When the controller starts to initialize, the pin state is high;
* When working normally, the pin state is high.The sequence is shown in the following figure:

The POWER

The internal reset

Software can make

DLL locking

Sys\_reset DDR\_RESETn

Particles RESETn

##### 

Reverse mode, reset\_ctrl[1:0] == 2 'b10.In this mode, the active level of reset pin is contrary to the normal control mode.So DDR\_RESETn needs to be connected to the corresponding pin on the memory slot on the motherboard through the inverter.The behavior of the pin is:

* When the power is not on: the pin state is low;
* When power on: the pin state is low;
* When the controller is configured: the pin state is high;
* When the controller starts to initialize: the pin state is low;
* Normal operation: the pin state is low.The sequence is shown in the following figure:

The POWER

The internal reset

Software can make

DLL locking

Sys\_reset DDR\_RESETn

Particles RESETn

Reset disable mode, pm\_reset\_ctrl[1:0] == 2 'b01.In this mode, the reset signal pin remains low throughout the memory operation. So DDR\_RESETn needs to be connected to the corresponding pin on the memory slot on the motherboard through the inverter. The behavior of the pin is:

* Always low; The sequence is shown in the following figure:

The POWER

The internal reset

Software can make

DLL locking

Sys\_reset DDR\_RESETn

Particles RESETn

With the combination of the latter two reset modes, STR control can be implemented directly using the reset signal of the memory controller. When the whole system starts from the closed state, use the method in (2) to reset the memory bar normally and start working. When the system recovers from STR, the method in (3) is used to reconfigure the memory bar so that it can resume normal operation without destroying the original state of the memory bar.

#### Leveling

It was used in DDR3 to intelligently configure the phase relationship between the various signals in the read and write operations of the memory controller. It's often the case that it was written like that, like Read like that, like Gate like that. In this controller, there was as much Write as Gate and as little Read as possible. The software needed to be able to Read as well as Read. In addition to the DQS phase and GATE phase operated in the process, the configuration method of writing DQ phase and reading DQ phase can also be calculated based on these final confirmed phases.

#### The Write Leveling

* + - * 1. Write ever was used to configure the phase relationship between Write DQS and the clock. Software programming required the following steps.
        2. Complete the initialization of the controller, as described in the previous section.
        3. Dll\_wrdqs\_x (x = 0... 8) set to 0;
        4. Set Lvl\_mode (0x180) to 2 'b01;
        5. Sample the Lvl\_ready (0x180) register. If it is 1, you can begin to Write a request.
        6. Set Lvl\_req (0x180) to 1;
        7. Sample the Lvl\_done (0x180) register. If it is 1, it shows that a Write request was done.
        8. Sample Lvl\_resp\_ (x 0x180, 0x188) registers, if 0, the corresponding Dll\_wrdqs\_x[6:0]

Add 1 and repeat 5-7; If it was 1, it must have been successful.

* + - * 1. At this point, the value of Dll\_wrdqs\_x should be the correct setting value.
        2. Now it was written. If Lvl\_resp\_x is found to be 1 on the first sampling, this result is problematic and other registers should be checked for incorrect Settings, such as Wrdqs\_lt\_half, Dqs\_start\_edge, Dqs\_stop\_edge, Dqs\_oe\_begin, and Dqs\_oe\_end.
        3. Then set Wrdqs\_lt\_half\_x based on whether the value of Dll\_wrdqs\_x is less than 0x40;
        4. Dll\_wrdqs\_x is set based on whether the value of Dll\_wrdqs\_x is less than 0x20. If Dll\_wrdqs\_x > 0x20, Dll\_wrdata\_x = Dll\_wrdqs\_x -- 0x20, otherwise Dll\_wrdata\_x = Dll\_wrdqs\_x

+ 0 x60;

* + - * 1. Set Wrdata\_lt\_half\_x based on whether Dll\_wrdata\_x is less than 0x40;
        2. Determine if the following conditions exist: different Dll\_wrdata\_x values are near 0x40, and there are cases that cross the 0x40 boundary (meaning that some Dll\_wrdata\_x is slightly less than 0x40, and some Dll\_wrdata\_x is slightly greater than 0x40). If this happens, set Write\_clk\_delay\_x to 1 for the Wrdata\_lt\_half\_x == 0 data group. Then subtract 1 from the values of tPHY\_WRDATA and tRDDATA.
        3. Set Lvl\_mode (0x180) to 2 'b00 to exit from Write like this.

##### 

#### Gate Leveling

Gate was used to configure the timing of the DQS window in the controller. The following steps were used in software programming.

* + - * 1. Complete the initialization of the controller, as described in the previous section.
        2. See the last measure.
        3. Dll\_gate\_x (x = 0... 8) set to 0;
        4. Set Lvl\_mode (0x180) to 2 'b10;
        5. Sample Lvl\_ready (0x180) register. If it's 1, you can start Gate like this.
        6. Set Lvl\_req (0x180) to 1;
        7. Sample the Lvl\_done (0x180) register. If it is 1, it means a Gate like request was done.
        8. Sample the Lvl\_resp\_x[0] (0x180, 0x188) registers.If Lvl\_resp\_x[0] is found to be 1 on the first sampling, increment the corresponding Dll\_gate\_x[6:0] by 1 and repeat 6-8 until the sampling result is 0, otherwise proceed to the next step.
        9. If the sample result is 0, increment the corresponding Dll\_gate\_x[6:0] by 1 and repeat 6-9;If it is
        10. Gate was a success.
        11. So this was the end of Gate. Now the sum between Dll\_gate\_x[6:0] and Dll\_wrdata\_x[6:0] was in effect a phase relationship between DQS and the internal clock over the PHY. Now let's adjust the parameters accordingly.
        12. Dll\_rddqs\_lt\_halt is set to 1 if the sum of Dll\_gate\_x[6:0] and Dll\_wrdata\_x[6:0] is less than or greater than 0x20 or 0x60. Because the phase relation of RDDQS is actually equal to a quarter of the input read DQS.
        13. At this point, if the value of Dll\_gate\_x is greater than 0x40, subtract 0x40 from the value of Dll\_gate\_x; Otherwise, set it to 0.
        14. Once the adjustment is completed, Lvl\_req is performed for two more times respectively to observe the change in the values of Lvl\_resp\_x[7:5] and Lvl\_resp\_x[4:2]. If the values of Lvl\_resp\_x[7:5] and Lvl\_resp\_x[4:2] are increased to Burst\_length/2, the 13th operation is continued. If it is not 4, you may need to add or subtract one from Rd\_oe\_begin\_x, and if it is greater than Burst\_length/2, you will most likely need to tweak the value of Dll\_gate\_x
        15. Set Lvl\_mode (0x180) to 2 'b00 to exit Gate.

#### Issue the MRS command separately

The order of MRS commands sent to memory by the memory controller is: MR2\_CS0, MR2\_CS1, MR2\_CS2, MR2\_CS3, MR3\_CS0, MR3\_CS2, MR1\_CS3, MR1\_CS3, MR1\_CS0, MR1\_CS1, MR1\_CS2, MR1\_CS3, MR1\_CS3, MR1\_CS1, MR1\_CS2, MR1\_CS3, MR1\_CS3.

Among them, whether the MRS command of corresponding CS is valid or not is determined by Cs\_mrs. Only if the bits of each slice selected on Cs\_mrs are valid can this MRS command be issued to DRAM. The value of each MR is determined by the register MR \*\_cs\*. These values are also used for the MRS command when initializing memory.

The specific operation is as follows:

1. Set registers Cs\_mrs (0x168), Mr\*\_cs\* (0x190 -- 0x1B8) to the correct values;
2. Set Command\_mode (0x190) to 1 to enable the controller to enter command sending mode;
3. Sample Status\_cmd (0x190). If it is 1, it means that the controller has entered command sending mode and can proceed to the next operation. If it is 0, it needs to continue to wait.
4. Write Mrs\_req (0x198) to 1 and send MRS command to DRAM;
5. Sample Mrs\_done (0x198). If it is 1, it means that the MRS command has been sent and can exit. If it is 0, it needs to continue to wait.
6. Set Command\_mode (0x190) to 0 to enable the controller to exit command send mode.

#### Any operation controls the bus

The memory controller can issue any combination of commands to the DRAM in command send mode, which can be set by the software

Cmd\_cs, Cmd\_cmd, Cmd\_ba, Cmd\_a (0x168), issued to DRAM in command send mode.The specific operation is as follows:

1. Set registers Cmd\_cs, Cmd\_cmd, Cmd\_ba, Cmd\_a (0x190) to the correct values;
2. Set Command\_mode (0x190) to 1 to enable the controller to enter command sending mode;
3. Sample Status\_cmd (0x190). If it is 1, it means that the controller has entered command sending mode and can proceed to the next operation. If it is 0, it needs to continue to wait.
4. Write Cmd\_req (0x190) to 1 and send the command to DRAM;
5. Set Command\_mode (0x190) to 0 to enable the controller to exit command send mode.

##### 

#### Self-looping test mode control

Since the cycle test pattern can be respectively in test mode or normal mode using the function, therefore, the memory controller, the two sets of independent control interface is implemented, a set of used in the test mode directly controlled by the test port, another set of used in normal function mode by the register allocation module configuration can make the test.

The multiplexing of these two interfaces is controlled by port test\_phy. When test\_phy is effective, the test\_\* port of the controller is used for control. At this time, the self-test is completely controlled by hardware. When test\_phy is invalid, the parameters of pm\_\* programmed by the software are used for control. The specific signal meaning for using the test port can be referred to the register parameter with the same name.

The two sets of interface from the control parameters are basically the same, only different access points, in this article introduces the control method of software programming.The specific operation is as follows:

1. Set all parameters of the memory controller correctly;
2. Set register Lpbk\_en (0x270) to 1;
3. Set register Init\_start (0x018) to 1;
4. The sample register Dll\_init\_done (0x000), if this value is 1, means that the DLL is locked, yes

Carry out the next operation; If this value is 0, you need to wait; (when using the test port for control, since the output of this register is not visible, it is not necessary to sample this register, but only need to wait here for a certain amount of time to ensure that the DLL lock is completed before proceeding to the next operation);

1. Set register Lpbk\_start (0x270) to 1; This is when the loop test begins.

Since the cycle test has started, the software needs to check whether there are any errors frequently. The specific operation is as follows:

1. Sampling register Lpbk\_error (0x270), if this value is 1, it means that an error has occurred. At this time, registers (0x270, 0x278, 0x280, 0x288) can be used to observe the error data and the correct data in the first error through Lpbk\_\* and other observations. If this value is 0, no data errors have occurred.

##### 

#### ECC functions use controls

ECC functionality is only available in 64-bit mode.

Ecc\_enable includes the following four control bits:

Ecc\_enable[0] controls whether the ECC function is enabled or not. ECC function will only be enabled if this significant bit is set.

Ecc\_enable[1] controls whether or not an error is reported through the read response path inside the processor so that a read access with ECC two-digit errors can immediately result in an exception in the processor core.

Ecc\_enable[2] controls whether an error is reported through the write response path within the processor so that a write access (read after write) with ECC two-digit errors can immediately cause an exception to occur in the processor core.

Ecc\_enable[3] controls when the error message is triggered in the register.These error messages will not be triggered continuously without software to process them, only the first error will be recorded. This information includes Ecc\_code, Ecc\_addr, Ecc\_data. When Ecc\_enable[3] is 0, as long as there is an ECC error (including 1 dislocation and 2 dislocation), the record will be triggered. When Ecc\_enable[3] is 1, the record will be triggered only if there is an ECC two-digit error. This "first time" refers to the setting of the corresponding bit of the interrupt vector register. That is, the access that caused the interruption is recorded.

In addition, ECC errors can be notified to the processor core by means of an interrupt. This interrupt is controlled with Int\_enable. The interrupt consists of two vectors, Int\_vector[0], indicating ECC error (including 1 dislocation and 2 dislocation), and Int\_vecotr[1], indicating ECC two-bit error.The clearing of Int\_vector is achieved by writing 1 to the corresponding bit.

## HyperTransport controller

In loongson 3A2000, the HyperTransport bus is used for external device connection and multi-chip interconnection. When used for connecting peripherals, but by the user program free to choose whether to support the IO Cache consistency (through the address window Uncache Settings, see section 10.5.13) : when configured to support the Cache consistency model, IO device internal DMA access for transparent Cache levels, namely by the hardware, automatically maintain consistency without software Cache instructions for maintenance by the program; When HyperTransport bus is used for multi-chip interconnection, HT0 controller (initial address 0x0C00\_0000\_0000)

0x0DFF\_FFFF\_FFFF) can be configured to support inter-chip Cache consistency transfer, while the HT1 controller (initial address 0x0E00\_0000\_0000 -- 0x0FFF\_FFFF\_FFFF) can be configured to support inter-chip Cache consistency maintenance, as shown in section 10.8.

The HyperTransport controller supports up to a two-way 16-bit width and a 2.0-ghz operating frequency. After the automatic initialization of the system to establish a connection, the user program can change the width and running frequency by modifying the corresponding configuration register in the protocol, and then re-initialize, as detailed in section 10.1.

The main features of loongson 3A2000 HyperTransport controller are as follows:

* + Support for the HT1.0/HT3.0 protocol
  + Support 200/400/800/1600/2000mhz operating frequency
  + HT1.0 supports 8-bit widths
  + HT3.0 supports 8/16 bit widths
  + Each HT controller (HT0/HT1) can be configured as two 8-bit HT controllers
  + The bus control signal (including PowerOK, Rstn, LDT\_Stopn) direction is configurable
  + Peripheral DMA space Cache/Uncache is configurable
  + Cache consistency mode can be configured for multi-slice interconnection

#### HyperTransport hardware setup and initialization

The HyperTransport bus consists of a transmission signal bus and a control signal pin, as shown in the following table

HyperTransport bus-related pins and their functional descriptions.

Table 10-1 HyperTransport bus-related pin signals

|  |  |  |
| --- | --- | --- |
| **pin** | **The name of the** | **describe** |
| HT0\_8x2 | Bus width configuration | 1: configure the 16-bit HyperTransport bus as two independent 8-bit buses, |
|  |  | It is controlled by two independent controllers, and the address space is distinguished as 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, address space is the address of HT0\_Lo, that is, address[40]  = 0; All signals of HT0\_Hi are invalid. |
| HT0\_Lo\_mode | Master device mode | 1: set HT0\_Lo as the main device mode. In this mode, bus control signals are driven by HT0\_Lo, including HT0\_Lo\_Powerok, HT0\_Lo\_Rstn, HT0\_Lo\_Ldt\_Stopn.In this mode, these control signals can also be bidirectional.At the same time, this pin determines the initial value of the register "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 request address on the HyperTransport bus is not hit by the controller's receive window, it will be sent back to the bus as a P2P request, and if this register is 1, if it is not hit, it will be responded as an error request.  0: set HT0\_Lo to slave device mode. In this mode, bus control signals are driven by other devices, including HT0\_Lo\_Powerok, HT0\_Lo\_Rstn, HT0\_Lo\_Ldt\_Stopn.In this mode, these control signals are driven by each other's devices, or if they are not driven correctly, the HT bus  Not working properly. |
| HT0\_Lo\_Powerok | Bus Powerok | When the HyperTransport master line Powerok number, HT0\_Lo\_Mode is 1, it is controlled by HT0\_Lo; When HT0\_Lo\_Mode is 0, it is controlled by the other device. |
| HT0\_Lo\_Rstn | Bus Rstn | When the Rstn signal number of HyperTransport main line, HT0\_Lo\_Mode is 1, it is controlled by HT0\_Lo;  When HT0\_Lo\_Mode is 0, it is controlled by the other device. |
| HT0\_Lo\_Ldt\_Stopn | Bus Ldt\_Stopn | HyperTransport main line Ldt\_Stopn, HT0\_Lo\_Mode = 1, is controlled by HT0\_Lo;  When HT0\_Lo\_Mode is 0, it is controlled by the other device. |
| HT0\_Lo\_Ldt\_Reqn | Bus Ldt\_Reqn | HyperTransport bus Ldt\_Reqn signal, |
| HT0\_Hi\_mode | Master device mode | 1: set HT0\_Hi to the main device mode. In this mode, bus control signals are driven by HT0\_Hi. These control signals include HT0\_Hi\_Powerok, HT0\_Hi\_Rstn, HT0\_Hi\_Ldt\_Stopn.In this mode, these control signals can also be bidirectional.At the same time, this pin determines the initial value of the register "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 request address on the HyperTransport bus is not hit by the controller's receive window, it will be sent back to the bus as a P2P request, and if this register is 1, if it is not hit, it will be responded as an error request.  0: set HT0\_Hi to slave device mode, in which bus control signals, etc  Driven by the other device, these control signals include HT0\_Hi\_Powerok, |

|  |  |  |
| --- | --- | --- |
|  |  | HT0\_Hi\_Rstn HT0\_Hi\_Ldt\_Stopn.In this mode, these control signals are driven by each other's devices, or if they are not driven correctly, the HT bus  Not working properly. |
| HT0\_Hi\_Powerok | Bus Powerok | When the HyperTransport main line Powerok number, HT0\_Lo\_Mode is 1, it is controlled by HT0\_Hi; When HT0\_Lo\_Mode is 0, it is controlled by the other device. When HT0\_8x2 is 1, control the high 8-bit bus;  Invalid when HT0\_8x2 is 0. |
| HT0\_Hi\_Rstn | Bus Rstn | When Rstn number of HyperTransport main line, HT0\_Lo\_Mode is 1, it is controlled by HT0\_Hi; When HT0\_Lo\_Mode is 0, it is controlled by the other device. When HT0\_8x2 is 1, control the high 8-bit bus; Invalid when HT0\_8x2 is 0. |
| HT0\_Hi\_Ldt\_Stopn | Bus Ldt\_Stopn | HyperTransport main line Ldt\_Stopn, HT0\_Lo\_Mode = 1, is controlled by HT0\_Hi; When HT0\_Lo\_Mode is 0, it is controlled by the other device. When HT0\_8x2 is 1, control the high 8-bit bus;  Invalid when HT0\_8x2 is 0. |
| HT0\_Hi\_Ldt\_Reqn | Bus Ldt\_Reqn | HyperTransport bus Ldt\_Reqn signal, HT0\_8x2 is 1, control the high 8-bit bus;  Invalid when HT0\_8x2 is 0. |
| HT0\_Rx\_CLKp HT0\_Rx\_CLKn [1:0] [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 HT0\_Rx\_CTLn [1:0] [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  When CTL[0] is controlled by HT0\_Lo and HT0\_8x2 is 0, CTL[1] is invalid  The CTL[0] is controlled by HT0\_Lo |
| HT0\_Rx\_CADp HT0\_Rx\_CADn [15:0] [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. After cold startup, the HyperTransport bus will automatically work at the lowest frequency (200MHz) and minimum width (8bit), and attempt to perform the bus initialization handshake.Whether the initialization is Complete or not can be read by the register "Init Complete" (see section 10.5.2).After initialization, the Width of the bus can be read from the registers "Link Width Out" and "Link Width In" (see section 10.5.2).After the initialization is completed, the user can rewrite the registers "Link Width Out", "Link Width In" and "Link Freq". At the same time, the user also needs to configure the corresponding register of the other device. After the configuration is completed, the user needs to hot-reset the bus or pass through

The "HT\_Ldt\_Stopn" signal is reinitialized to give effect to the overwritten value of the register. The HyperTransport bus will work at the new frequency and width after the reinitialization is complete. It is important to note that the configuration of devices at both ends of HyperTransport needs to be one-to-one, otherwise the HyperTransport interface will not work properly.

#### HyperTransport protocol support

Loongson 3A2000's HyperTransport bus supports most of the commands in the 1.03/3.0 protocol, and includes some extended instructions in the extended conformance protocol that supports multi-chip interconnection. In both modes, the HyperTransport receiver can receive commands as shown in the following table. It is important to note that the atomic operation command of the HyperTransport bus is not supported.

Table 10-2 commands that the HyperTransport receiver can receive

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **coding** | **channel** | **The command** | **The standard model** | **Extension (consistency)** |
| 000000 | - | The NOP | Empty packet or flow control |  |
| 000001 | NPC | FLUSH | No operation |  |
| x01xxx | NPC  The or PC | The 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 |
| 01 XXXX | NPC | The 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 | RdRespons e | Read operation return |  |
| 110011 | R | TgtDone | Write operation return |  |
| 110100 | The PC | WrCoherent | ---- | Write command extension |
| 110101 | The PC | WrAddr | ---- | Write address extension |
| 111000 | R | RespCohere  nt | ---- | Read response extension |
| 111001 | NPC | RdCoherent | ---- | Read command extension |
| 111010 | The PC | Broadcast | No operation |  |
| 111011 | NPC | RdAddr | ---- | Read address extension |
| 111100 | The PC | A FENCE | Guarantee order relation |  |
| 111111 | - | The Sync/Error | The Sync/Error |  |

For the sending side, the commands sent out in both modes are shown in the following table.

Table 10-3 commands sent out in both modes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **coding** | **channel** | **The command** | **The standard model** | **Extension (consistency)** |
| 000000 | - | The NOP | Empty packet or flow control |  |
| x01x0x | NPC  The or PC | The Write | Bit 5:0 - Nonposted  1 - Posted bit 2:0 - Byte  1 - Doubleword | Bit 5: must be 1, POSTED    Bit 2:0 -- Byte  1 - Doubleword |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | Bit 0: must be 0 | Bit 0: must be 1 |
| 010 x0x | NPC | The 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 return |  |
| 110011 | R | TgtDone | Write operation return |  |
| 110100 | The PC | WrCoherent | ---- | Write command extension |
| 110101 | The PC | WrAddr | ---- | Write address extension |
| 111000 | R | RespCoherent | ---- | Read response extension |
| 111001 | NPC | RdCoherent | ---- | Read command extension |
| 111011 | NPC | RdAddr | ---- | Read address extension |
| 111111 | - | The Sync/Error | Will only forward |  |

#### HyperTransport interrupt support

The HyperTransport controller provides 256 interrupt vectors and supports Fix, Arbiter, and other interrupt types, but no hardware automatic EOI support. For the above two supported types of interrupts, the controller will automatically write to the interrupt register after receiving and notify the system interrupt controller according to the Settings of the interrupt mask register. For the specific interrupt control, see the interrupt control register group in section 10.5.8.

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

A typical PIC interrupt is completed by the following steps: (1) the PIC controller sends a PIC interrupt request to the system; The system sends interrupt vector query to PIC controller; (3) PIC controller sends interrupt vector number to the system; The system clears the corresponding interrupt on the PIC controller. The PIC controller will not issue the next interrupt to the system until all 4 steps above have been completed. For loongson 3A2000 HyperTransport controller, the first 3 steps will be automatically processed and the PIC interrupt vector will be written to the corresponding position in the 256 interrupt vectors. After the interrupt is processed by the software system, the fourth step is required, which is to send a clear interrupt to the PIC controller. Then the processing of the next interrupt begins.

#### HyperTransport address window

* + 1. **HyperTransport space**

In loongson 3A2000 processor, the address window distribution of the default 4 HyperTransport interfaces is as follows: table 10-4 the address window

|  |  |  |  |
| --- | --- | --- | --- |
| **Base address** | **End** address | **The size of the** | **define** |
| 0 x0c00\_0000\_0000 | 0 x0cff\_ffff\_ffff | One Tbytes | HT0\_LO window |
| 0 x0d00\_0000\_0000 | 0 x0dff\_ffff\_ffff | One Tbytes | HT0\_HI window |

distribution of the default 4 HyperTransport interfaces

|  |  |  |  |
| --- | --- | --- | --- |
| 0 x0e00\_0000\_0000 | 0 x0eff\_ffff\_ffff | One Tbytes | HT1\_LO window |
| 0 x0f00\_0000\_0000 | 0 x0fff\_ffff\_ffff | One Tbytes | HT1\_HI window |

By default (the system address window is not configured separately), the software accesses each HyperTransport interface according to the above address space. In addition, the software can configure the address window on the crossover switch to access it with other address Spaces (see section 2.5). The internal 40-bit address space of each HyperTransport interface has its address window distribution as shown in the following table.

Table 10-5 address window distribution inside the HyperTransport interface of loongson 3 processor

|  |  |  |  |
| --- | --- | --- | --- |
| **Base address** | **End address** | **The size of the** | **define** |
| 0 x00\_0000\_0000 | 0 xfc\_ffff\_ffff | 1012 Gbytes | MEM space |
| 0 xfd\_0000\_0000 | 0 xfd\_f7ff\_ffff | 3968 Mbytes | reserve |
| 0 xfd\_f800\_0000 | 0 xfd\_f8ff\_ffff | 16 Mbytes | interrupt |
| 0 xfd\_f900\_0000 | 0 xfd\_f90f\_ffff | 1 Mbyte | PIC interrupt response |
| 0 xfd\_f910\_0000 | 0 xfd\_f91f\_ffff | 1 Mbyte | System information |
| 0 xfd\_f920\_0000 | 0 xfd\_faff\_ffff | 30 Mbytes | reserve |
| 0 xfd\_fb00\_0000 | 0 xfd\_fbff\_ffff | 16 Mbytes | HT controller configuration space |
| 0 xfd\_fc00\_0000 | 0 xfd\_fdff\_ffff | 32 Mbytes | I/O space |
| 0 xfd\_fe00\_0000 | 0 xfd\_ffff\_ffff | 32 Mbytes | HT bus configuration space |
| 0 xfe\_0000\_0000 | 0 xff\_ffff\_ffff | 8 Gbytes | reserve |

* + 1. **HyperTransport controller internal window configuration**

The HyperTransport interface of loongson 3A2000 processor provides a variety of rich address Windows for users to use. The functions and functions of these address Windows are described in the following table.

Table 10-6 address window provided in loongson 3A2000 processor HyperTransport interface

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **The address window** | **Window number** | **Accept the bus** | **role** | **note** |
| Receiving window  See window configuration  [10.5.7)](#_bookmark106) | 3 | HyperTransport | Judge whether to accept  An access emitted on the HyperTransport bus. | In the main bridge mode (i.e., act\_as\_slave is 0 in the configuration register), only the access that falls into these address Windows will be responded to by the internal bus. Other access will be considered as P2P access and sent back to the HyperTransport bus.When in device mode (that is, act\_as\_slave is 1 in the configuration register), only the accesses that fall into these address Windows will be received and processed by the internal bus, and other accesses will be returned with protocol errors. |
| The Post window  See window configuration  [10.5.11)](#_bookmark139) | 2 | Inside the bus | Determines whether Write access from the internal bus to the HyperTransport bus is treated as Post Write | Outgoing Write access that falls into these address Spaces will be treated as Post Write.  In the Post Write: HyperTransport protocol, this Write access does not have to wait for the Write response, that is, after the controller issues the Write access to the bus, the Write access to the processor completes the response. |
| Prefetch window  See window configuration  10.5.12) | 2 | Inside the bus | To determine whether to receive internal Cache access, fetch access. | When the processor core is out of order, some guess reads or fetches are given to the bus, which is wrong for some IO Spaces.By default, this access HT controller will return directly without access to the HyperTransport bus.These Windows enable such access to the HyperTransport bus. |
| Uncache window  See window configuration  10.5.13) | 2 | HyperTransport | Determine if the access on the HyperTransport bus is to be accessed as Uncache on the inside | The IO DMA access inside the loongson 3A2000 processor will be judged as a hit by SCache in the case of Cache access, so as to maintain its IO consistency information. Through the configuration of these Windows, it is possible to make the access hit in these Windows access memory directly in the manner of Uncache, without maintaining its IO consistency information through hardware. |

#### Configuration register

The configuration register module is mainly used to control the access requests to the configuration register from the AXI SLAVE or HT RECEIVER, perform external interrupt processing, and save a large number of configuration registers visible to the software to control the various working modes of the system.

First, the access and storage of the configuration registers used to control the HT controller's various behaviors are in this module, whose access offset address is 0xFD\_FB00\_0000 to 0xFD\_FBFF\_FFFF at the HT controller end. The visible registers of all software in HT controller are shown in the following table:

Table 10-7 software visible register list

|  |  |  |
| --- | --- | --- |
| **offset** | **The name of the** | **describe** |
| 0 x30 |  |  |
| 0 x34 |  |  |
| 0 x38 |  |  |
| 0 x3c | Bridge Control | Bus Reset Control |
| 0 x40 | Capability Registers | Command, Capabilities Pointer, Capability ID |
| 0 x44 | Link Config, Link Control |
| 0 x48 | Revision ID, Link Freq, Link Error, Link Freq Cap |
| 0 x4c | Feature Capability |
| 0 x50 | Custom register | MISC |
| 0 x54 | Receive diagnostic register | Used to diagnose a sampled signal at the receiving end |
| 0 x58 | Interrupt routing mode selection register | Corresponds to three interrupt routing modes |
| 0 x5c | Receive cache register |  |
| 0 x60 | Receive address window configuration register | HT bus receive address window 0 enable (external access) |
| 0 x64 | HT bus receiving address window 0 base address (external access) |
| 0 x68 | HT bus receive address window 1 enable (external access) |
| 0 x6c | HT bus receiving address window 1 base address (external access) |
| 0 x70 |  | HT bus receive address window 2 enable (external access) |
| 0 x74 | HT bus receiving address window 2 base address (external access) |
| 0 x148 | HT bus receive address window 3 enable (external access) |
| 0 x14c | HT bus receive address window 3 base address (external access) |
| 0 x150 | HT bus receive address window 4 enable (external access) |
| 0 x154 | HT bus receive address window 4 base address (external access) |
| 0 x80 | Interrupt vector register | HT bus interrupt vector register [31:0] |
| 0 x84 | HT bus interrupt vector register [63:32] |
| 0 x88 | HT bus interrupt vector register [95:64] |
| 0 x8c | HT bus interrupt vector register [127:96] |
| 0 x90 | HT bus interrupt vector register [159:128] |
| 0 x94 | HT bus interrupt vector register [191:160] |
| 0 x98 | HT bus interrupt vector register [223:192] |
| 0 x9c | HT bus interrupt vector register [255:224] |
| 0 xa0 | Interrupt enable register | HT bus interrupts enable register [31:0] |
| 0 xa4 | HT bus interrupt enabled register [63:32] |
| 0 xa8 | HT bus interrupt enabled register [95:64] |
| 0 xac | HT bus interrupt enabled register [127:96] |
| 0 xb0 | HT bus interrupt enabled register [159:128] |
| 0 xb4 | HT bus interrupt enabled register [191:160] |
| 0 xb8 | HT bus interrupt enabled register [223:192] |
| 0 XBC | HT bus interrupt enabled register [255:224] |
| 0 xc0 | Interrupt Discovery & Configuration | Interrupt Capability |
| 0 xc4 | DataPort |
| 0 xc8 | IntrInfo [31:0] |
| 0 XCC | IntrInfo [63:32] |
| 0 xd0 | Configure the register in the POST address window | HT bus POST address window 0 enable (internal access) |
| 0 xd4 | HT bus POST address window 0 base address (internal access) |
| 0 xd8 | HT bus POST address window 1 enable (internal access) |
| 0 XDC | HT bus POST address window 1 base address (internal access) |
| 0 xe0-0xfc | Pre-fetch address window configuration register | HT bus prefetching address window 0 enable (internal access) |
| 0 xe4 | HT bus can prefetch address window 0 base address (internal access) |
| 0 xe8 | HT bus prefetching address window 1 enable (internal access) |
| 0 xec | Ht bus prefetch address window 1 base address (internal access) |
| 0 xf0 | Uncache address window configuration register | HT bus Uncache address window 0 enable (external access) |
| 0 xf4 | HT bus Uncache address window 0 base address (external access) |
| 0 xf8 | HT bus Uncache address window 1 enable (external access) |
| 0 XFC | HT bus Uncache address window 1 base address (external access) |
| 0 x168 | HT bus Uncache address window 2 enable (external access) |
| 0 x16c | HT bus Uncache address window 2 base address (external access) |
| 0 x170 | HT bus Uncache address window 3 enable (external access) |
| 0 x174 | HT bus Uncache address window 3 base address (external access) |
| 0 x158 | P2P address window configuration register | HT bus P2P address window 0 enable (external access) |
| 0 x15c | HT bus P2P address window 0 base address (external access) |
| 0 x160 | HT bus P2P address window 1 enable (external access) |
| 0 x164 | HT bus P2P address window 1 base address (external access) |
| 0 x100 | The sending end cache size register | The sending command caches the size register |
| 0 x104 | The sending end data cache size register |
| 0 x108 | The sending end cache debug register | Used to manually set the size of the sender cache (for debugging) |

|  |  |  |
| --- | --- | --- |
| 0 x10c | PHY impedance matching configuration register | Used to configure the impedance matching configuration for the sending and receiving ends of the PHY |
| 0 x110 | Revision ID register | Used to configure the controller version |
| 0 x118 | Error Retry controls the register | Retry Count Rollover,Short Retry Attempts |
| 0 x11c | The Retry Count register | Used for error retransmission counting in HyerTransport 3.0 mode |
| 0 x130 | Link Train register | HyperTransport 3.0 link initialization and link training control |
| 0 x134 | Training 0 short count sending over time  store | Used for Training 0 short timeout threshold configuration |
| 0 x138 | Training 0 timeout long count register | Used for Training 0 long count timeout threshold configuration |
| 0 x13c | Training 1 counting register | Used for Training 1 counting threshold configuration |
| 0 x140 | Training 2 counting register | Used for Training 2 counting threshold configuration |
| 0 x144 | Training 3 counting register | Used for Training 3 counting threshold configuration |
| 0 x178 | Software frequency configuration register | Realize the frequency switch of the controller in the working process |
| 0 x17c | PHY configuration register | Used to configure phy-related physical parameters |
| 0 x180 | The link initializes the debug register | For ignoring the PHY CDR lock signal and customizing the wait time |
| 0 x184 | LDT debug register | Used to configure the time from the invalidation of the LDT signal to the start of the link initialization |

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

### Bridge Control

Offset: 0x3C

Reset value: 0x00000000

Name: Bus Reset Control

Table 10-8 Bus Reset Control register definitions

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| For calamity | Reserved | 4 | 0 x0 |  | reserve |
| 22 | The Reset | 12 | 0 x0 | R/W | Bus reset control:   1. >1: HT\_RSTn set 0, bus reset 2. >0: HT\_RSTn set 1, bus unreset |
| 21:0 | Reserved | 5 | 0 x0 |  | reserve |

### Capability Registers

Offset: 0x40

Reset value: 0x20010008

Name: Command, Capabilities Pointer, Capability ID

Table 10-9 Command, Capabilities Pointer, Capability ID register definitions

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| take | The HOST/Sec | 3 | 0 x1 | R | The Command format is HOST/Sec |
| Forepart thereof | Reserved | 2 | 0 x0 | R | reserve |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 26 | Act as a Slave | 1 | 0 x0  / 0 x1 | R/W | The HOST/SLAVE mode  The initial value is determined by the pin HOSTMODE  HOSTMODE pull up: 0 HOSTMODE pull down: 1 |
| 25 | Reserved | 1 | 0 x0 |  | reserve |
| 24 | The Host Hide | 1 | 0 x0 | R/W | Whether to disable register access from HT bus |
| 23 | Reserved | 1 | 0 x0 |  | reserve |
| " | The Unit ID | 5 | 0 x0 | R/W | HOST mode: can be used to record the number of use ids  SLAVE mode: record self Unit ID |
| 17 | A Double Ended | 1 | 0 x0 | R | The dual HOST mode is not used |
| 16 | A Warm Reset | 1 | 0 x1 | R | The reset in Bridge Control adopts the thermal reset mode |
| " | "Capabilities Pointer | 8 | 0 xa0 | R | The next Cap register offset address |
| away | Capability ID | 8 | 0 x08 | R | HyperTransport capability ID |

Offset: 0x44

Reset value: 0x00112000

Name: Link Config, Link Control

Table 10-10 Link Config, Link Control register definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31 | ht\_phase\_select  \_disable | 1 | 0 x0 |  | Phase selection enable 0: enable phase selection function  1: disable the phase selection function |
| he | The Link Width Out | 3 | 0 x0 | R/W | Sending end width  The value after cold reset is the maximum width of the current connection. The value written to this register will take effect after the next hot reset or HT Disconnect  000:8 bit mode  001:16 bit mode |
| 27 | Reserved | 1 | 0 x0 |  | reserve |
| they | The Link Width In | 3 | 0 x0 | R/W | Receiver width  The value after cold reset is the maximum width of the current connection, write to this post  The save value will take effect after the next hot reset or HT Disconnect |
| 23 | Dw Fc out | 1 | 0 x0 | R | Sending terminal does not support double word flow control |
| Lift up | Max Link Width out | 3 | 0 x1 | R | HT bus sending end maximum width: 16bits |
| 19 | Dw Fc In | 1 | 0 x0 | R | The receiver does not support double word flow control |
| thou | Max Link Width In | 3 | 0 x1 | R | HT bus receiver maximum width: 16bits |
| The lowest | Reserved | 2 | 0 x0 |  | reserve |
| 13 | LDTSTOP#  Tristate Enable | 1 | 0 x1 | R/W | Whether to close HT PHY when HT bus enters HT Disconnect state  1: closed  0: not closed |
| 12:10 | Reserved | 3 | 0 x0 |  | reserve |
| 9 | CRC Error (hi) | 1 | 0 x0 | R/W | CRC errors occur in the high 8 bit |
| 8 | CRC Error (lo) | 1 | 0 x0 | R/W | CRC errors occur in the lower 8 bits |
| 7 | Trans off | 1 | 0 x0 | R/W | HT PHY close control  In 16-bit bus working mode 1: close high/low 8-bit HT PHY  0: enable low 8 HT phys,  The high 8-bit HT PHY is controlled by bit 0 |
| 6 | The End of the Chain | 0 | 0 x0 | R | HT bus terminal |
| 5 | Init Complete | 1 | 0 x0 | R | Whether the HT bus initialization is completed |
| 4 | The Link Fail | 1 | 0 x0 | R | Indicating connection failure |
| 3:2 | Reserved | 2 | 0 x0 |  | reserve |
| 1 | CRC Flood Enable | 1 | 0 x0 | R/W | Whether flood HT bus occurs when CRC error occurs |
| 0 | Trans off (hi) | 1 | 0 x0 | R/W | High 8-bit PHY closes control when running 8-bit protocol using 16-bit HT bus  1: close high 8-bit HT PHY  0: enable high 8 bit HT PHY |

Offset: 0x48

Reset value: 0x80250023

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

Table 10-11 Revision ID, Link Freq, Link Error, Link Freq Cap register definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| Caused the | The Link Freq Cap | 16 | 0 x0025 | R | The supported HT bus frequency is generated according to the external PLL Settings  Different values |
| The lowest | Reserved | 2 | 0 x0 |  | reserve |
| 13 | Over Flow Error | 1 | 0 x0 | R | HT bus packet overflow |
| 12 | Protocol Error | 1 | 0 x0 | R/W | Protocol error,  An unrecognized command received on the HT bus |
| and | The Link Freq | 4 | 0 x0 | R/W | HT bus operating frequency  The value of this register will take effect after the next hot reset or HT Disconnect  0000-200 m  0010-400 m  0101-800 m |
| away | Revision ID | 8 | 0 x23 | R/W | Version number: 1.03 |

Offset: 0x4C

Reset value: 0x00000002

Name: Feature Capability

Table 10-12 Feature Capability register definitions

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| Is wasted | Reserved | 25 | 0 x0 |  | reserve |
| 8 | Extended the Register | 1 | 0 x0 | R | There is no |
| The log | Reserved | 3 | 0 x0 |  | reserve |
| 3 | Extended CTL Time | 1 | 0 x0 | R | Don't need |
| 2 | CRC Test Mode | 1 | 0 x0 | R | Does not support |
| 1 | LDTSTOP# | 1 | 0 x1 | R | Support LDTSTOP# |
| 0 | Isochronous Mode | 1 | 0 x0 | R | Does not support |

### Custom register

Offset: 0x50

Reset value: 0x00904321

Name: MISC

Table 10-13 MISC register definitions

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31 | Reserved | 1 | 0 x0 |  | reserve |
| 30 | Ldt Stop Gen | 1 | 0 x0 | R/W | Take the bus into LDT DISCONNECT mode  The correct way is: 0 minus >, 1 |
| 29 | Ldt the Req Gen | 1 | 0 x0 | R/W | Waken HT bus from LDT DISCONNECT, set  LDT\_REQ\_n  The right way to do it is to put a 0 and then a 1:0 minus >, 1  In addition, direct read and write requests to the bus can also automatically wake up the bus |
| throughout | Interrupt the Index | 5 | 0 x0 | R/W | Redirect interrupts other than standard interrupts to which interrupt vector (including SMI, NMI, INIT, INTA, INTB, INTC, INTD)  There are 256 interrupt vectors in total. This register represents the interrupt direction  The high of the quantity is 5 bits, the internal interrupt vector is as follows: 000: SMI  001: NMI  010: INIT  011: Reservered 100: INTA  101: intb.br deal  110: INTC  111: INTD |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 23 | Dword Write | 1 | 0 x1 | R/W | For 32/64/128/256 bit write access, whether to use  Dword Write command format 1: use Dword Write  0: use Byte Write (with MASK) |
| 22 | Coherent Mode | 1 | 0 x0 | R | Processor consistency mode  Determined by pin ICCC\_EN |
| 21 | Not Care Seqid | 1 | 0 x0 | R/W | Whether you don't care about HT |
| 20 | The Not Axi2Seqid | 1 | 0 x1 | R | Whether to convert the commands on the Axi bus to a different SeqID or not, if not, all read and write commands will take the Fixed ID number in the Fixed SeqID  1: no conversion  0: conversion |
| He hath | Fixed Seqid | 4 | 0 x0 | R/W | Configure the HT bus emitted when Not Axi2Seqid is valid  Seqid |
| " | Priority Nop | 4 | 0 x4 | R/W | HT bus Nop flow control packet priority |
| and | Specify the NPC | 4 | 0 x3 | R/W | Non Post channel read and write priority |
| The log | Priority RC | 4 | 0 x2 | R/W | The Response channel reads and writes first |
| 3-0 | Priority PC | 4 | 0 x1 | R/W | Post channel read and write priority 0x0: highest priority  0xF: lowest priority  The priority strategy of increasing with time is adopted for each channel, and this memory set is used to configure the initial priority of each channel |

### Receive diagnostic register

Offset: 0x54

Reset value: 0x00000000

Name: receive diagnostic register

Table 10-14 receive diagnostic registers

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 0 | Sample\_en | 1 | 0 x0 | R/W | Enable sampling of input cad and CTL  0 x0: ban  0 x1: can make |
| " | rx\_ctl\_catch | 24 | 0 x0 | R/W | Save the input CTL from the sample  (0, 2, 4, 6) corresponds to the four phases of CTL0 sampling  (1, 3, 5, 7) corresponds to the four phases of CTL1 sampling |
| Caused the | rx\_cad\_phase\_0 | 24 | 0 x0 | R/W | Save the values of the input CAD[15:0] from the sample |

### Interrupt routing mode selection register

Offset: 0x58

Reset value: 0x00000000

Name: interrupt routing selection register

Table 10-15 interrupt routing selection registers

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| o | ht\_int\_stripe | 2 | 0 x0 | R/W | Corresponding to three interrupt routing methods, see 0 interrupt vector register for details  0x0: ht\_int\_stripe\_1 0x1: ht\_int\_stripe\_2  0 x2: ht\_int\_stripe\_4 |

### Receive buffer initial register

Offset: 0x5c

Reset value: 0x07778888

Name: receive buffer initialization configuration register

Table 10-16 receive buffer initial registers

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| he | rx\_buffer\_r\_data | 4 | 0 x0 | R/W | Receive buffer's read data buffer initialization information |
| Behold, | rx\_buffer\_npc\_data | 4 | 0 x0 | R/W | Receive the NPC data buffer initialization information for the buffer |
| He hath | rx\_buffer\_pc\_data | 4 | 0 x0 | R/W | Receive the buffer's PC data buffer initialization information |
| " | rx\_buffer\_b\_cmd | 4 | 0 x0 | R/W | Receive the bresponse command buffer initialization information for the buffer |
| and | rx\_buffer\_r\_cmd | 4 | 0 x0 | R/W | Receive buffer's read command buffer initialization information |
| The log | rx\_buffer\_npc\_cmd | 4 | 0 x0 | R/W | Receive the NPC command buffer initialization information for the buffer |
| 3-0 | rx\_buffer\_pc\_cmd | 4 | 0 x0 | R/W | Receive the buffer's PC command buffer initialization information |

### Receive address window configuration register

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

Hit = (BASE & MASK) == (ADDR & MASK)

Addr\_out = TRANS\_EN? TRANS b0 ADDR & ~MASK: ADDR

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

Is the size of the address window.

The address of the receive address window is the address received on the HT bus. The HT address dropped in the P2P window will act as P2P

Commands are forwarded back to the HT bus. The HT addresses that fall in the normal receiving window and are not in the P2P window will be sent to the CPU, and commands from other addresses will be forwarded back to the HT bus as P2P commands.

Offset: 0x60

Reset value: 0x00000000

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

Table 10-17 HT bus receive address window 0 enable (external access) register definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| 31 | ht\_rx\_image0\_en | 1 | 0 x0 | R/W | HT bus receives address window 0, enabling signal |
| 30 | ht\_rx\_image0\_  trans\_en | 1 | 0 x0 | R/W | HT bus receives address window 0, mapping enable signal |
| 29:0 | ht\_rx\_image0\_  Trans [53:24] | 30 | 0 x0 | R/W | HT bus receiving address window 0, the mapped address [53:24] |

Offset: 0x64

Reset value: 0x00000000

Name: HT bus receiving address window 0 base address (external access)

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| Caused the | ht\_rx\_image0\_  The base [they] | 16 | 0 x0 | R/W | HT bus receive address window 0, address base address [39:24] |
| 15:0 | ht\_rx\_image0\_  Mask [they] | 16 | 0 x0 | R/W | HT bus receiving address window 0, address shielded [39:24] |

Offset: 0x68

Reset value: 0x00000000

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

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| 31 | ht\_rx\_image1\_en | 1 | 0 x0 | R/W | HT bus receives address window 1, enabling signal |
| 30 | ht\_rx\_image1\_  trans\_en | 1 | 0 x0 | R/W | HT bus receives address window 1, mapping the enable signal |
| 29:0 | ht\_rx\_image1\_  Trans [53:24] | 30 | 0 x0 | R/W | HT bus receives address window 1, the mapped address [53:24] |

Offset: 0x6c

Reset value: 0x00000000

Name: HT bus receiving address window 1 base address (external access)

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| Caused the | ht\_rx\_image1\_  The base [they] | 16 | 0 x0 | R/W | HT bus receive address window 1, address base address [39:24] |
| 15:0 | ht\_rx\_image1\_  Mask [they] | 16 | 0 x0 | R/W | HT bus receive address window 1, address mask [39:24] |

Offset: 0x70

Reset value: 0x00000000

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

Table 10-21 HT bus receive address window 2 enable (external access) register definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| 31 | ht\_rx\_image2\_en | 1 | 0 x0 | R/W | HT bus receives address window 2, enabling signal |
| 30 | ht\_rx\_image2\_  trans\_en | 1 | 0 x0 | R/W | The HT bus receives the address window 2, mapping the enable signal |
| 29:0 | ht\_rx\_image2\_  Trans [53:24] | 16 | 0 x0 | R/W | HT bus receiving address window 2, translated address [53:24] |

Offset: 0x74

Reset value: 0x00000000

Name: HT bus receiving address window 2 base address (external access)

Table 10-22 HT bus receive address window 2 base address (external access) register definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| Caused the | ht\_rx\_image2\_  The base [they] | 16 | 0 x0 | R/W | HT bus receive address window 2, address base address [39:24] |
| 15:0 | ht\_rx\_image2\_  Mask [they] | 16 | 0 x0 | R/W | HT bus receive address window 2, address mask [39:24] |

Offset: 0x148

Reset value: 0x00000000

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

Table 10-23 HT bus receive address window 3 enable (external access) register definitions

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| 31 | ht\_rx\_image3\_en | 1 | 0 x0 | R/W | HT bus receives address window 3, enabling signal |
| 30 | ht\_rx\_image3\_  trans\_en | 1 | 0 x0 | R/W | HT bus receives address window 3, mapping the enable signal |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| 29:0 | ht\_rx\_image3\_  Trans [53:24] | 16 | 0 x0 | R/W | HT bus receiving address window 3, translated address [53:24] |

Offset: 0x14C

Reset value: 0x00000000

Name: HT bus receiving address window 3 base address (external access)

Table 10-24 HT bus receive address window 3 base address (external access) register definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| Caused the | ht\_rx\_image3\_  The base [they] | 16 | 0 x0 | R/W | HT bus receive address window 3, address base address [39:24] |
| 15:0 | ht\_rx\_image3\_  Mask [they] | 16 | 0 x0 | R/W | HT bus receive address window 3, address mask [39:24] |

Offset: 0x150

Reset value: 0x00000000

Name: HT bus receive address window 4 enable (external access)

Table 10-25 HT bus receive address window 4 enable (external access) register definitions

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| 31 | ht\_rx\_image4\_en | 1 | 0 x0 | R/W | HT bus receives address window 4, enabling signal |
| 30 | ht\_rx\_image4\_  trans\_en | 1 | 0 x0 | R/W | HT bus receives address window 4, mapping enable signal |
| 29:0 | ht\_rx\_image4\_  Trans [53:24] | 16 | 0 x0 | R/W | HT bus receiving address window 4, translated address [53:24] |

Offset: 0x154

Reset value: 0x00000000

Name: HT bus receiving address window 4 base address (external access)

Table 10-26 HT bus receive address window 4 base address (external access) register definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| Caused the | ht\_rx\_image4\_  The base [they] | 16 | 0 x0 | R/W | HT bus receive address window 4, address base address [39:24] |
| 15:0 | ht\_rx\_image4\_  Mask [they] | 16 | 0 x0 | R/W | HT bus receive address window 4, address mask [39:24] |

### Interrupt vector register

Interrupt vector registers 256 in all, except for Fix, Arbiter, and PIC interrupt direct mappings on HT bus

To these 256 interrupt vectors, other interrupts, such as SMI, NMI, INIT, INTA, INTB, INTC, INTD, can be mapped to any 8-bit interrupt vector by register 0x50 [28:24] in the order of {INTD, INTC, INTB, INTA, 1 'b0, INIT, NMI, SMI}. The corresponding value of Interrupt vector is {Interrupt Index, internal vector [2:0]}.

LS3A1000E and above versions, 256 interrupt vectors are mapped to different interrupt lines according to different interrupt routing mode selection register configuration. The specific mapping mode is as follows:

Ht\_int\_stripe\_1:

[0,1,2,3... 63] corresponds to the interrupted line 0 /HT HI corresponds to the interrupted line 4

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

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

[192,193,194,195... 255] corresponding to interrupted line 3 /HT HI corresponding to interrupted line 7 ht\_int\_stripe\_2:

[0,2,4,6... 126] corresponds to the interrupted line 0 /HT HI corresponds to the interrupted line 4

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

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

[129,131,133,135... 255] corresponding to discontinuous line 3 /HT HI corresponding to discontinuous line 7 ht\_int\_stripe\_4:

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

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

[2,6,10,14... 254] corresponding to discontinuous line 2 /HT HI corresponding to discontinuous line 6

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

The following interrupt vector description corresponds to ht\_int\_stripe\_1, and two other ways can be obtained as described above. For LS3A1000D and below, you can only use ht\_int\_stripe\_1.

Offset: 0x80

Reset value: 0x00000000

Name: HT bus interrupt vector register [31:0]

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| 31:0 | Interrupt\_case  [31:0] | 32 | 0 x0 | R/W | HT bus interrupt vector register [31:0],  It's 0 over HT HI and it's 4 |

Offset: 0x84

Reset value: 0x00000000

Name: HT bus interrupt vector register [63:32]

Table 10-28 HT bus interrupt vector register definition (2)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| 31:0 | Interrupt\_case  [63:32] | 32 | 0 x0 | R/W | HT bus interrupt vector register [63:32],  It's 0 over HT HI and it's 4 |

Offset: 0x88

Reset value: 0x00000000

Name: HT bus interrupt vector register [95:64]

Table 10-29 HT bus interrupt vector register definition (3)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31:0 | Interrupt\_case  [95-64] | 32 | 0 x0 | R/W | HT bus interrupt vector register [95:64],  This corresponds to 1 /HT HI and this corresponds to 5 |

Offset: 0x8c

Reset value: 0x00000000

Name: HT bus interrupt vector register [127:96]

Table 10-30 HT bus interrupt vector register definition (4)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31:0 | Interrupt\_case  [127-96] | 32 | 0 x0 | R/W | HT bus interrupt vector register [127:96],  This corresponds to 1 /HT HI and this corresponds to 5 |

Offset: 0x90

Reset value: 0x00000000

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

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31:0 | Interrupt\_case  [159-128] | 32 | 0 x0 | R/W | HT bus interrupt vector register [159:128],  So this corresponds to 2 /HT HI and this corresponds to 6 |

Offset: 0x94

Reset value: 0x00000000

Name: HT bus interrupt vector register [191:160]

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31:0 | Interrupt\_case  [191-160] | 32 | 0 x0 | R/W | HT bus interrupt vector register [191:160],  So this corresponds to 2 /HT HI and this corresponds to 6 |

Offset: 0x98

Reset value: 0x00000000

Name: HT bus interrupt vector register [223:192]

Table 10-32 HT bus interrupt vector register definition (7)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31:0 | Interrupt\_case  [223-192] | 32 | 0 x0 | R/W | HT bus interrupt vector register [223:192],  So this corresponds to 3 /HT HI and this corresponds to 7 |

Offset: 0x9c

Reset value: 0x00000000

Name: HT bus interrupt vector register [255:224]

Table 10-33 HT bus interrupt vector register definition (8)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31:0 | Interrupt\_case  [255-224] | 32 | 0 x0 | R/W | HT bus interrupt vector register [255:224],  So this corresponds to 3 /HT HI and this corresponds to 7 |

### Interrupt enable register

There are 256 interrupt enabled registers, corresponding to the interrupt vector register. Set 1 to open the corresponding interrupt, set 0

Then it is interrupt shielding.

The 256 interrupt vectors are mapped to different interrupt lines according to the interrupt routing mode and register configuration. The specific mapping mode is as follows:

Ht\_int\_stripe\_1:

[0,1,2,3... 63] corresponds to the interrupted line 0 /HT HI corresponds to the interrupted line 4

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

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

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

Ht\_int\_stripe\_2:

[0,2,4,6... 126] corresponds to the interrupted line 0 /HT HI corresponds to the interrupted line 4

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

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

[129,131,133,135... 255] corresponding to discontinuous line 3 /HT HI corresponding to discontinuous line 7 ht\_int\_stripe\_4:

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

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

[2,6,10,14... 254] corresponding to discontinuous line 2 /HT HI corresponding to discontinuous line 6

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

The following interrupt vector description corresponds to ht\_int\_stripe\_1, and two other ways can be obtained as described above.

Offset: 0xa0

Reset value: 0x00000000

Name: HT bus interrupt enabled register [31:0]

Table 10-34 HT bus interrupt enabled register definition (1)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31:0 | Interrupt\_mask  [31:0] | 32 | 0 x0 | R/W | HT bus interrupts enable register [31:0],  It's 0 over HT HI and it's 4 |

Offset: 0xa4

Reset value: 0x00000000

Name: HT bus interrupt enabled register [63:32]

Table 10-35 HT bus interrupt enabled register definition (2)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31:0 | Interrupt\_mask  [63:32] | 32 | 0 x0 | R/W | HT bus interrupts enable register [63:32],  It's 0 over HT HI and it's 4 |

Offset: 0xa8

Reset value: 0x00000000

Name: HT bus interrupt enabled register [95:64]

Table 10-36 HT bus interrupt enabled register definition (3)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31:0 | Interrupt\_mask  [95-64] | 32 | 0 x0 | R/W | HT bus interrupts enable register [95:64],  This corresponds to 1 /HT HI and this corresponds to 5 |

Offset: 0xac

Reset value: 0x00000000

Name: HT bus interrupt enabled register [127:96]

Table 10-37 HT bus interrupt enabled register definition (4)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31:0 | Interrupt\_mask  [127-96] | 32 | 0 x0 | R/W | HT bus interrupts enable register [127:96],  This corresponds to 1 /HT HI and this corresponds to 5 |

Offset: 0xb0

Reset value: 0x00000000

Name: HT bus interrupt enabled register [159:128]

Table 10-38 HT bus interrupt enabled register definition (5)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31:0 | Interrupt\_mask  [159-128] | 32 | 0 x0 | R/W | HT bus interrupts enable register [159:128],  So this corresponds to 2 /HT HI and this corresponds to 6 |

Offset: 0xb4

Reset value: 0x00000000

Name: HT bus interrupt enabled register [191:160]

Table 10-39 HT bus interrupt enabled register definition (6)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31:0 | Interrupt\_mask  [191-160] | 32 | 0 x0 | R/W | HT bus interrupts enable register [191:160],  So this corresponds to 2 /HT HI and this corresponds to 6 |

Offset: 0xb8

Reset value: 0x00000000

Name: HT bus interrupt enabled register [223:192]

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31:0 | Interrupt\_mask  [223-192] | 32 | 0 x0 | R/W | HT bus interrupts enable register [223:192],  So this corresponds to 3 /HT HI and this corresponds to 7 |

Offset: 0xbc

Reset value: 0x00000000

Name: HT bus interrupt enabled register [255:224]

Table 10-41 HT bus interrupt enabled register definition (8)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31:0 | Interrupt\_mask  [255-224] | 32 | 0 x0 | R/W | HT bus interrupts enable register [255:224],  So this corresponds to 3 /HT HI and this corresponds to 7 |

### Interrupt Discovery & Configuration

Offset: 0xc0

Reset value: 0x80000008

Name: Interrupt Capability

Table 10-42 definitions of the Interrupt Capability register

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| came | "Capabilities Pointer | 8 | 0 x80 | R | Interrupt discovery and configuration block |
| Ephron; | The Index | 8 | 0 x0 | R/W | Read register offset address |
| " | "Capabilities Pointer | 8 | 0 x0 | R | "Capabilities Pointer |
| away | Capability ID | 8 | 0 x08 | R | Hypertransport Capablity ID |

Offset: 0xc4

Reset value: 0x00000000

Name: Dataport

Table 10-43 Dataport register definitions

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31:0 | Dataport | 32 | 0 x0 | R/W | When the Index of the previous register is 0x10, the read-write result of this register is 0xa8 register, otherwise it is 0xac |

Offset: 0xc8

Reset value: 0xF8000000

Name: IntrInfo [31:0]

Table 10-44 IntrInfo register definitions (1)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| came | IntrInfo [came] | 32 | 0 xf8 | R | reserve |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| Isle; | IntrInfo [great] | 22 | 0 x0 | R/W | IntrInfo[23:2], the value of IntrInfo when sending PIC interrupt  It's used to represent interrupt vectors |
| 1-0 | Reserved | 2 | 0 x0 | R | reserve |

Offset: 0xcc

Reset value: 0x00000000

Name: IntrInfo [63:32]

Table 10-45 IntrInfo register definitions (2)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31:0 | IntrInfo [63:32] | 32 | 0 x0 | R | reserve |

### Configure the register in the POST address window

The address window hit formula is shown in section 10.5.7.

The address of this window is the one received on the AXI bus. All WRITE accesses that fall into this window will immediately be returned in the AXI B channel and sent to the HT bus in the command format of POST WRITE. WRITE requests that are not in this window are sent to the HT bus as NONPOST WRITE and wait for the HT bus to respond before returning to the AXI bus.

Offset: 0xd0

Reset value: 0x00000000

Name: HT bus POST address window 0 enable (internal access)

Table 10-46 HT bus POST address window 0 enable (internal access)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31 | ht\_post0\_en | 1 | 0 x0 | R/W | HT bus POST address window 0, enable signal |
| 30 | ht\_depart0\_en | 1 | 0 x0 | R/W | HT access unpack enable (corresponding to the external of the CPU core  Uncache ACC operation window) |
| throne | Reserved | 14 | 0 x0 |  | reserve |
| 15:0 | ht\_post0\_trans  [they] | 16 | 0 x0 | R/W | HT bus POST address window 0, translated address [39:24] |

Offset: 0xd4

Reset value: 0x00000000

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

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| Caused the | ht\_post0\_base  [they] | 16 | 0 x0 | R/W | HT bus POST address window 0, address base address [39:24] |
| 15:0 | ht\_post0\_mask  [they] | 16 | 0 x0 | R/W | HT bus POST address window 0, address masked [39:24] |

Offset: 0xd8

Reset value: 0x00000000

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

Table 10-48 HT bus POST address window 1 enable (internal access)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31 | ht\_post1\_en | 1 | 0 x0 | R/W | HT bus POST address window 1, enable signal |
| 30 | ht\_depart1\_en | 1 | 0 x0 | R/W | HT access unpack enable (corresponding to the external of the CPU core  Uncache ACC operation window) |
| Was a | Reserved | 14 | 0 x0 |  | reserve |
| 15:0 | ht\_post1\_trans  [they] | 16 | 0 x0 | R/W | HT bus POST address window 1, translated address [39:24] |

Offset: 0xdc

Reset value: 0x00000000

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

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| Caused the | ht\_post1\_base  [they] | 16 | 0 x0 | R/W | HT bus POST address window 1, address base address [39:24] |
| 15:0 | ht\_post1\_mask  [they] | 16 | 0 x0 | R/W | HT bus POST address window 1, address masked [39:24] |

### Pre-fetch address window configuration register

The address window hit formula is shown in section 10.5.7.

The address of this window is the one received on the AXI bus. Other fetches or CACHE accesses will not be sent to the HT bus, but will be returned immediately. If it is a read command, invalid read data of the corresponding number will be returned.

Offset: 0xe0

Reset value: 0x00000000

Name: HT bus prefetching address window 0 enable (internal access)

Table 10-50 HT bus prefetching address window 0 enable (internal access)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31 | ht\_prefetch0\_en | 1 | 0 x0 | R/W | HT bus can prefetch address window 0, enable signal |
| then | Reserved | 15 | 0 x0 |  | reserve |
| 15:0 | ht\_prefetch0\_trans  [they] | 16 | 0 x0 | R/W | HT bus prefetch address window 0, translated address [39:24] |

Offset: 0xe4

Reset value: 0x00000000

Name: HT bus prefetch address window 0 base address (internal access)

Table 10-51 HT bus prefetch address window 0 base address (internal access)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| Caused the | Ht\_prefetch0\_ base [they] | 16 | 0 x0 | R/W | HT bus can prefetch address window 0, address base address [39:24]  An address |
| 15:0 | ht\_prefetch0\_  Mask [they] | 16 | 0 x0 | R/W | HT bus can prefetch address window 0, address mask [39:24] |

Offset: 0xe8

Reset value: 0x00000000

Name: HT bus prefetching address window 1 enable (internal access)

Table 10-52 HT bus prefetching address window 1 enable (internal access)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31 | ht\_prefetch1\_en | 1 | 0 x0 | R/W | HT bus can prefetch address window 1 to enable the signal |
| then | Reserved | 15 | 0 x0 |  | reserve |
| 15:0 | ht\_prefetch1\_  Trans. [they] | 16 | 0 x0 | R/W | HT bus prefetch address window 1, translated address [39:24] |

Offset: 0xec

Reset value: 0x00000000

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

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| A domain | A domain name | A wide | Reset value | access | describe |
| Caused the | ht\_prefetch1\_  The base [they] | 16 | 0 x0 | R/W | HT bus can prefetch address window 1, address base address [39:24] |
| 15:0 | ht\_prefetch1\_  Mask [they] | 16 | 0 x0 | R/W | HT bus can prefetch address window 1, address mask [39:24] |

### UNCACHE address window configuration register

The address window hit formula is shown in section 10.5.7.

The address of this window is the address received on the HT bus. The read/write command that falls at the address of this window will not be sent to SCACHE, nor will it invalidate the level 1 CACHE, but will be sent directly to memory or other address space, that is, the read/write command in this address window will not maintain the CACHE consistency of IO. This window focuses on operations that are not hit in the CACHE and therefore can improve memory efficiency, such as access to video memory.

Offset: 0xf0

Reset value: 0x00000000

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

Table 10-54 HT bus Uncache address window 0 enable (internal access)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31 | ht\_uncache0\_en | 1 | 0 x0 | R/W | HT bus uncache address window 0, enable signal |
| 30 | ht\_uncache0\_  trans\_en | 1 | 0 x0 | R/W | HT bus uncache address window 1, mapping enable signal |
| 29:0 | Ht\_uncache0\_ trans [53:24] | 16 | 0 x0 | R/W | HT bus uncache address window 0, translated address  [53:24] |

Offset: 0xf4

Reset value: 0x00000000

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

Table 10-55 HT bus Uncache address window 0 base address (internal access)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| Caused the | ht\_uncache0\_  The base [they] | 16 | 0 x0 | R/W | HT bus uncache address window 0, address base address [39:24] |
| 15:0 | ht\_uncache0\_  Mask [they] | 16 | 0 x0 | R/W | HT bus uncache address window 0, address mask [39:24] |

Offset: 0xf8

Reset value: 0x00000000

Name: HT bus Uncache address window 1 enable (internal access)

Table 10-56 HT bus Uncache address window 1 enable (internal access)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31 | ht\_uncache1\_en | 1 | 0 x0 | R/W | HT bus uncache address window 1, enable the signal |
| 30 | ht\_uncache1\_  trans\_en | 1 | 0 x0 | R/W | HT bus uncache address window 1, mapping enable signal |
| 29:0 | Ht\_uncache1\_ trans [53:24] | 16 | 0 x0 | R/W | HT bus uncache address window 1, translated address [53:24] |

Offset: 0xfc

Reset value: 0x00000000

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

Table 10-57 HT bus Uncache address window 1 base address (internal access)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| Caused the | ht\_uncache1\_  The base [they] | 16 | 0 x0 | R/W | HT bus uncache address window 1, address base address [39:24] |
| 15:0 | ht\_uncache1\_  Mask [they] | 16 | 0 x0 | R/W | HT bus uncache address window 1, address mask [39:24] |

Offset: 0x168

Reset value: 0x00000000

Name: HT bus Uncache address window 2 enable (internal access)

Table 10-58 HT bus Uncache address window 2 enable (internal access)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31 | ht\_uncache1\_en | 1 | 0 x0 | R/W | HT bus uncache address window 2, enable the signal |
| 30 | Ht\_uncache1\_ trans\_en | 1 | 0 x0 | R/W | HT bus uncache address window 2, mapping enable signal |
| 29:0 | ht\_uncache1\_  Trans [53:24] | 16 | 0 x0 | R/W | HT bus uncache address window 2, after translation of the address  [53:24] |

Offset: 0x16c

Reset value: 0x00000000

Name: HT bus Uncache address window 2 base address (internal access)

Table 10-59 HT bus Uncache address window 2 base address (internal access)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| Caused the | ht\_uncache1\_  The base [they] | 16 | 0 x0 | R/W | HT bus uncache address window 2, address base address [39:24] |
| 15:0 | ht\_uncache1\_  Mask [they] | 16 | 0 x0 | R/W | HT bus uncache address window 2, address mask [39:24] |

Offset: 0x170

Reset value: 0x00000000

Name: HT bus Uncache address window 3 enable (internal access)

Table 10-60 HT bus Uncache address window 3 enable (internal access)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31 | ht\_uncache1\_en | 1 | 0 x0 | R/W | HT bus uncache address window 3, enable the signal |
| 30 | ht\_uncache1\_  trans\_en | 1 | 0 x0 | R/W | HT bus uncache address window 3, mapping enable signal |
| 29:0 | Ht\_uncache1\_ trans [53:24] | 16 | 0 x0 | R/W | HT bus uncache address window 3, after translation of the address  [53:24] |

Offset: 0x174

Reset value: 0x00000000

Name: HT bus Uncache address window 3 base address (internal access)

Table 10-61 HT bus Uncache address window 3 base addresses (internal access)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| Caused the | ht\_uncache1\_  The base [they] | 16 | 0 x0 | R/W | HT bus uncache address window 3, address base address [39:24] |
| 15:0 | ht\_uncache1\_  Mask [they] | 16 | 0 x0 | R/W | HT bus uncache address window 3, address mask [39:24] |

### P2P address window configuration register

The address window hit formula is shown in section 10.5.7.

The address of this window is the address received on the HT bus. Read and write commands that fall on the address of this window, directly as P2P

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

Offset: 0x158

Reset value: 0x00000000

Name: HT bus P2P address window 0 enable (external access)

Table 10-62 HT bus P2P address window 0 enable (external access) register definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31 | ht\_p2p\_image0\_en | 1 | 0 x0 | R/W | HT bus P2P address window 0, enable signal |
| 30 | ht\_p2p\_image0\_  trans\_en | 1 | 0 x0 | R/W | HT bus P2P address window 0, mapping enable signal |
| 29:0 | ht\_p2p\_image0\_  Trans [53:24] | 16 | 0 x0 | R/W | HT bus P2P address window 0, the translated address [53:24] |

Offset: 0x15c

Reset value: 0x00000000

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

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| Caused the | Ht\_p2p\_image0\_ base [they] | 16 | 0 x0 | R/W | HT bus P2P address window 0, address base address [39:24] |
| 15:0 | ht\_p2p\_image0\_  Mask [they] | 16 | 0 x0 | R/W | HT bus P2P address window 0, address shielding [39:24] |

Offset: 0x160

Reset value: 0x00000000

Name: HT bus P2P address window 1 enable (external access)

Table 10-64 HT bus P2P address window 1 enable (external access) register definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31 | ht\_p2p\_image1\_en | 1 | 0 x0 | R/W | HT bus P2P address window 1, enable signal |
| 30 | Ht\_p2p\_image1\_ trans\_en | 1 | 0 x0 | R/W | HT bus P2P address window 1, mapping enable signal |
| 29:0 | ht\_p2p\_image1\_  Trans [53:24] | 16 | 0 x0 | R/W | HT bus P2P address window 1, the translated address [53:24] |

Offset: 0x164

Reset value: 0x00000000

Name: HT bus P2P address window 1 base address (external access)

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| A domain | A domain name | A wide | Reset value | access | describe |
| Caused the | ht\_p2p\_image1\_  The base [they] | 16 | 0 x0 | R/W | HT bus P2P address window 1, address base address [39:24] |
| 15:0 | Ht\_p2p\_image1\_ mask [they] | 16 | 0 x0 | R/W | HT bus P2P address window 1, address shielded [39:24] |

### The command sends the cache size register

The command send cache size register is used to measure the number of caches available for each command channel at the sending end.

Offset: 0x100

Reset value: 0x00000000

Name: command sends cache size register

Table 10-66 sends the cache size register

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| came | B\_CMD\_txbuffer | 8 | 0 x0 | R | Number of B channel command caches at sending end |
| 2316 | R\_CMD\_txbuffer | 8 | 0 x0 | R | Number of R channel command caches at sending end |
| " | NPC\_CMD\_txbuffer | 8 | 0 x0 | R | Number of sending end NPC channel command cache |
| away | PC\_CMD\_txbuffer | 8 | 0 x0 | R | Number of PC channel command caches at sending end |

### Data send cache size register

The data send cache size register is used to measure the number of caches available for each data channel at the sending end.

Offset: 0x104

Reset value: 0x00000000

Name: data send cache size register

Table 10-67 data sending cache size registers

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| came | Reserved | 8 | 0 x0 | R | reserve |
| 2316 | R\_DATA\_txbuffer | 8 | 0 x0 | R | The number of R channel data caches at the sending end |
| " | NPC\_DATA\_txbuffer | 8 | 0 x0 | R | The number of data caches of sending end NPC channel |
| away | PC\_DATA\_txbuffer | 8 | 0 x0 | R | The number of data caches of sending PC channel |

### Send the cache debug register

The send cache debug register is used to manually set the number of buffers at the sending end of the HT controller by increasing or decreasing

Adjust the number of different send caches.

Offset: 0x108

Reset value: 0x00000000

Name: send cache debug register

Tables 10-68 send cache debug registers

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| charm | Reserved | 2 | 0 x0 | R | reserve |
| 29 | Tx\_neg | 1 | 0 x0 | R/W | Send side cache debug symbol 0: increase the number  1: reduce (the number of corresponding registers +1) |
| 28 | Tx\_buff\_adj\_en | 1 | 0 x0 | R/W | The sending side cache debug enable register  0->1: causes the value of this register to increase or decrease |
| he | R\_DATA\_txadj | 4 | 0 x0 | R/W | The number of increase or decrease of R channel data cache at the sending end  When tx\_neg is 0, increase R\_DATA\_txadj; When tx\_neg is 1, reduce R\_DATA\_txadj+1 |
| Behold, | NPC\_DATA\_txadj | 4 | 0 x0 | R/W | Sending end NPC channel data cache increase or decrease  When tx\_neg is 0, increase NPC\_DATA\_txadj; When tx\_neg is 1, reduce NPC\_DATA\_txadj+1 |
| He hath | PC\_DATA\_txadj | 4 | 0 x0 | R/W | The number of data cache increase or decrease of PC channel at the sending end  When tx\_neg is 0, increase PC\_DATA\_txadj; When tx\_neg is 1, reduce PC\_DATA\_txadj+1 |
| " | B\_CMD\_txadj | 4 | 0 x0 | R/W | Send end B channel command cache increase or decrease  When tx\_neg is 0, add B\_CMD\_txadj; When tx\_neg is 1, reduce B\_CMD\_txadj+1 |
| and | R\_CMD\_txadj | 4 | 0 x0 | R/W | Send R channel command cache increase or decrease  When tx\_neg is 0, add R\_CMD\_txadj; When tx\_neg is 1, reduce R\_CMD\_txadj+1 |
| The log | NPC\_CMD\_txadj | 4 | 0 x0 | R/W | Sending end NPC channel command/data cache increase or decrease  When tx\_neg is 0, increase NPC\_CMD\_txadj; When tx\_neg is 1, reduce NPC\_CMD\_txadj+1 |
| 3-0 | PC\_CMD\_txadj | 4 | 0 x0 | R/W | Send PC channel command cache increase or decrease  When tx\_neg is 0, increase PC\_CMD\_txadj; When tx\_neg is 1, reduce PC\_CMD\_txadj+1 |

### PHY impedance matching control register

Used to control the impedance matching enabling of PHY, the sending and receiving impedance matching parameters are set to offset: 0x10C

Reset value: 0x00000000

Name: PHY impedance matching control register

Table 10-69 impedance matching control registers

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| 31 | Tx\_scanin\_en | 1 | 0 x0 | R/W | TX impedance matching enabled |
| 30 | Rx\_scanin\_en | 1 | 0 x0 | R/W | RX impedance matching enabled |
| he | Tx\_scanin\_ncode | 4 | 0 x0 | R/W | TX impedance matching scan input ncode |
| Behold, | Tx\_scanin\_pcode | 4 | 0 x0 | R/W | TX impedance matching scan input pcode |
| then | Rx\_scanin\_code | 8 | 0 x0 | R/W | RX impedance matching scan input |

### Revision ID register

Used to configure the controller version, configured to a new version number, with a Warm Reset effect.

Offset: 0x110

Reset value: 0x00200000

Name: RevisionID register

Table 10-70 Revision ID register

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| came | Reserved | 8 | 0 x0 | R | reserve |
| Ephron; | Revision ID | 8 | 0 x20 | R/W | Revision ID control register 0x20: HyperTransport 1.00  0 x60: HyperTransport 3.00 |
| 15:0 | Reserved | 16 | 0 x0 | R | reserve |

### Error Retry controls the register

For error retransmission enablers in HyerTransport 3.0 mode, configure the maximum number of Short Retry, display

Whether the Retry counter is flipped.Offset: 0x118

Reset value: 0x00000000

Name: Error Retry control register

Table 10-71 Error Retry control register

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| for | Reserved | 22 | 0 x0 | R | reserve |
| 9 | Retry Count Rollover | 1 | 0 x0 | R | Retry counter count flip |
| 8 | Reserved | 1 | 0 x0 | R | reserve |
| but | Short Retry Attempts | 2 | 0 x0 | R/W | The maximum number of Short Retry allowed |

### The Retry Count register

Used for error retransmission counting in HyerTransport 3.0 mode.Offset: 0x11C

Reset value: 0x00000000

Name: Retry Count register

Table 10-72 Retry Count register

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| conspiracies | Reserved | 12 | 0 x0 | R | reserve |
| He hath | Rrequest delay | 4 | 0 x0 | R/W | Used to control the random delay range of Rrequest transmissions in consistent mode  000:0 delay  001: random delay 0-8  010: random delay of 8-15  011: random delay 16-31  100: random delay 32-63  101: random delay 64-127  110: random delay 128-255  111:0 delay |
| 15:0 | Retry Count | 16 | 0 x0 | R | Retry count |

### Link Train register

HyperTransport 3.0 link initialization and link training control registers.Offset: 0x130

Reset value: 0x00000070

Name: Link Train register

Table 10-73 Link Train register

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| For calamity | Reserved | 9 | 0 x0 | R | reserve |
| "You | Transmitter LS the select | 2 | 0 x0 | R/W | Link state for the sender in Disconnected or Inactive state:  2 'b00 LS1  2 'b01 LS0  2 'b10 LS2  2 'bl1 you |
| 14 | DsiableCmd  Throttling | 1 | 0 x0 | R/W | In HyperTransport 3.0 mode, only one non-info CMD can appear in any 4 consecutive DWS by default.  1 'b0 enables Cmd Throttling  1 'b1 Cmd Throttling is prohibited |
| " | Reserved | 4 | 0 x0 | R | reserve |
| " | Receiver LS the select | 2 | 0 x0 | R/W | Link state for the receiver in Disconnected or Inactive state:  2 'b00 LS1 |
|  |  |  |  |  | 2 'b01 LS0  2 'b10 LS2  2 'bl1 you |
| 6:4 | Long Retry Count | 3 | 0 x7 | R/W | The maximum number of Long Retry |
| 3 | Scrambling the Enable | 1 | 0 x0 | R/W | Do you make it possible to Scramble  1: can Scramble |
| 2 | 8 b10b Enable | 1 | 0 x0 | R/W | Enable or disable 8B10B 0: disable 8B10B  1: can make 8 b10b |
| 1 | AC | 1 | 0 x0 | R | Is AC mode detected  0: no AC mode detected  1: AC mode has been detected |
| 0 | Reserved | 1 | 0 x0 | R | reserve |

### Training 0 timeout short timing register

Used to configure Training 0 short timeouts timeout threshold in HyerTransport 3.0 mode, and the counter clock frequency is

HyperTransport3.0 link bus clock frequency 1/4.Offset: 0x134

Reset value: 0x00000080

Name: Training 0 timeout short count register

Table 10-74 Training 0 timeout short timing register

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| 31:0 | T0 time | 32 | By 8 0 | R/W | Training 0 timeout short timing register |

### Training 0 timeout long timing register

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

HyperTransport3.0 link bus clock frequency 1/4.Offset: 0x138

Reset value: 0x000fffff

Name: Training 0 timeout long count register

Table 10-75 Training 0 timeout long count register

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| 31:0 | T0 time | 32 | 0 XFFFFF | R/W | Training 0 timeout long count register |

### Training 1 counting register

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

HyperTransport3.0 link bus clock frequency 1/4.

Offset: 0x13C

Reset value: 0x0004fffff

Name: Training 1 counting register

Table 10-76 Training 1 counting register

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| 31:0 | T1 time | 32 | 0 x4fffff | R/W | Training 1 counting register |

### Training 2 counting register

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

HyperTransport3.0 link bus clock frequency 1/4.Offset: 0x144

Reset value: 0x0007fffff

Name: Training 2 counting register

Table 10-77 Training 2 counting registers

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| 31:0 | T2 time | 32 | 0 x7fffff | R/W | Training 2 counting register |

### Training 3 counting register

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

HyperTransport3.0 link bus clock frequency 1/4.Offset: 0x13C

Name: Training 3 counting register

Table 10-78 Training 3 counting registers

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| 31:0 | T3 time | 32 | 0 x7fffff | R/W | Training 3 counting register |

### Software frequency configuration register

In the case of CLKSEL[15] pulling down, it is used to realize the controller switching to any protocol and PLL supported link frequency and controller frequency during the working process. In the case of CLKSEL[15] pulling high, it has no effect.

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

And write the new clock-related parameters, including div\_refc and div\_loop that determine the PLL output frequency, the frequency division coefficients of phy\_hi\_div and phy\_lo\_div on the link, and the frequency division coefficient of the controller, core\_div. After entering warm reset or LDT disconnect, the controller will automatically reset PLL and configure the new clock parameters.

The calculation formula of clock frequency is:

HyperTransport 1.0:

PHY\_LINK\_CLK = 50MHz×div\_loop /div\_refc /phy\_div HT\_CORE\_CLK=100MHz×div\_loop /div\_refc /core\_div

HyperTransport 3.0:

PHY\_LINK\_CLK =100MHz×div\_loop /div\_refc ht\_clk =100MHz×div\_loop /div\_refc /core\_div

The time to wait for PLL relock is about 30us when system CLK is 33M by default. You can also write custom wait count caps in registers.

Offset: 0x178

Reset value: 0x00000000

Name: software frequency configuration register

Table 10-79 software frequency configuration registers

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| behold | PLL relock counter | 5 | 0 x0 | R/W | Counter upper limit configuration register  When set to counter select, the upper limit of the counter count is  {PLL\_relock\_counter, 5 'h1f}  Otherwise the count is capped at 10 '3ff |
| 26 | Counter the select | 1 | 0 x0 | R/W | Lock timer custom enable: 1 'b0 USES the default count upper limit;  1 'b1 is calculated from PLL\_relock\_counter |
| Struggled together | Soft\_phy\_lo\_div | 4 | 0 x0 | R/W | High PHY frequency division coefficient |
| Lift up | Soft\_phy\_hi\_div | 4 | 0 x0 | R/W | Low frequency division coefficient |
| " | Soft\_div\_refc | 2 | 0 x0 | R/W | PLL internal frequency division coefficient |
| Put no | Soft\_div\_loop | 7 | 0 x0 | R/W | PLL internal frequency multiplication coefficient |
| then | Soft\_core\_div | 4 | 0 x0 | R/W | Controller clock frequency division coefficient |
| 4-2 | Reserved | 3 | 0 x0 | R | reserve |
| 1 | Soft cofig enable | 1 | 0 x0 | R/W | Software configuration enable bit  1 'b0 disables software frequency configuration  1 'b1 enables software frequency configuration |
| 0 | Reserved | 1 | 0 x0 | R | reserve |

### PHY configuration register

For the configuration of phy-related physical parameters, when the controller ACTS as two independent 8bit controllers, the higher

The PHY and the low-lying PHY are controlled by two controllers independently. When the controller ACTS as a 16bit controller, the configuration parameters of high-bit and low-bit PHY are uniformly controlled by the low-bit controller.

Offset: 0x17C

Reset value: 0x83308000

Name: PHY configuration register

Table 10-80 PHY configuration registers

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| 31 | Rx\_ckpll\_term | 1 | 0 x1 | R/W | The terminal impedance of the transmission line from PLL to RX |
| 30 | Tx\_ckpll\_term | 1 | 0 x0 | R/W | The terminal impedance of the transmission line from PLL to TX |
| 29 | Rx\_clk\_in\_sel\_ | 1 | 0 x0 | R/W | The clock PAD supplies the clock selection of the data PAD, which is automatically selected as CLKPAD in HT1 mode:  1 'b0 external clock source  1 'b1 PLL clock |
| 28 | Rx\_ckdll\_sell | 1 | 0 x0 | R/W | Clock selection for locking DLL: 1 'b0 PLL clock  1 'b1 external clock source |
| But after | Rx\_ctle\_bitc | 2 | 0 x0 | R/W | PAD EQD high frequency gain |
| Thus for | Rx\_ctle\_bitr | 2 | 0 x3 | R/W | PAD EQD low frequency gain |
| " | Rx\_ctle\_bitlim | 2 | 0 x0 | R/W | PAD EQD compensation limitation |
| 21 | Rx\_en\_ldo | 1 | 0 x1 | R/W | They control  1 'b0 "disabled  1 'b1 can they make |
| 20 | Rx\_en\_by | 1 | 0 x1 | R/W | BandGap control  1 'b0 BandGap disabled 1' b1 BandGap enabled |
| michal | Reserved | 3 | 0 x0 | R | reserve |
| then | Tx\_preenmp | 5 | 0 x08 | R/W | PAD preload control signal |
| 11:0 | Reserved | 12 | 0 x0 | R | reserve |

### The link initializes the debug register

It is used to configure whether to use the CDR lock signal provided by PHY as the symbol to complete the link CDR during the link initialization in HyperTransport 3.0 mode. If the lock signal is ignored, the controller counts and waits for a certain amount of time before the default CDR completes.

Offset: 0x180

Reset value: 0x00000000

Name: link initialization debug register

Table 10-81 link initializes debug registers

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| 15 | Cdr\_ignore\_enable | 1 | 0 x0 | R/W | Whether CRC lock is ignored when the link is initialized, wait for completion through counter counting:  1 'b0 waits for CDR lock  1 'b1 ignores the CDR lock signal and accumulates and waits through the counter |
| 14:0 | Cdr\_wait\_counter | 15 | 0 x0 | R/W | Wait for the counter to count the upper limit, based on the controller clock to complete the technique |

### LDT debug register

When the software changes the frequency of the controller, the timing of the LDT reconnect phase will not be accurate, so the count needs to be configured

The time between the LDT signal being invalid as a software configuration frequency and the controller starting the link initialization is based on the controller clock.

Offset: 0x184

Reset value: 0x00000000

Name: LDT debug register

Table 10-82 LDT debug registers

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| Caused the | Rx\_wait\_time | 16 | 0 x0 | R/W | The RX terminal waits for the initial value of the counter |
| 15:0 | Tx\_wait\_time | 16 | 0 x0 | R/W | The TX end waits for the initial value of the counter |

#### The HyperTransport bus conpoints the access methods to the space

The protocol of the HyperTransport interface software layer is basically the same as that of PCI. Since the access of the configuration space is directly related to the underlying protocol, the specific access details are slightly different.[As listed in table 10-5, the address range of the HT bus configuration space is 0xFD\_FE00\_0000 ~ 0xFD\_FFFF\_FFFF.](#_bookmark86) For configuration access in HT protocol, the following format is adopted in loongson 3A2000:



Type 0:

Type 1:

? Figure 10-1 configuration access of HT protocol in loongson 3A2000

#### HyperTransport bus frequency software configuration method

The frequency of the HyperTransport interface bus can be controlled by two methods. The first is to configure the PLL frequency with the CLKSEL pin, and the second is to set the actual bus frequency with the configuration register Link Freq (offset 0x48, see 10.5.2).The other is to use the software frequency configuration register inside the controller (see 10.5.28) to set the PLL and the actual bus frequency. Compared with the pin setting method, a more abundant frequency combination can be obtained.Both methods eventually require a software reset via a one-time bus or LDT RECONNECT to take effect.

The method of using CLKSEL pin configuration is simple and will not be covered here. The method of setting using software registers is described in section 10.5.28, and some special notes are given here.

When a 16-bit HT is split into two 8-bit HTS, only the software frequency configuration register of HT LO can control PLL and bus frequency division, including the bus frequency division value of HT HI. That is to say, if left untreated, when HT LO resets the frequency, the HT HI frequency will also change. At this point, if HT HI is in a normal functional state, the bus may become unstable.

To avoid this, there are two ways to do it.

The first is to connect the reset signals of all HT together, so that after the software frequency configuration register of all HT controllers is configured, the reset signals of HT will be uniformly pulled down, and then pulled up to shake hands again. This enables HT LO and HT HI to switch clocks at the same time, ensuring the normal operation of the system. This method is suitable for HT0 connections in four-way interconnected systems.

The other is when the HT reset signal cannot be connected together. This needs to be prevented by means of software control

HT HI was affected when HT LO switched PLL frequency. The most straightforward approach is to set HT HI to the reset state,

Until HT LO completes the switch of PLL frequency, then HT HI's bus decompression is reset. This method is suitable for HT1 LO

The condition of the cross interconnect of the bridge piece, HT1 HI.

#### HyperTransport multi-processor support

The loongson 3 processor USES the HyperTransport interface to interconnect multiple processors and is capable of hardware automatic maintenance

Consistency requests between 4 chips. Here are two ways to interconnect multiple processors:

###### Four pieces of loong core no. 3 interconnection structure

Four pieces of CPU are connected to form a ring structure. Each CPU USES two 8-bit controllers of HT0 to connect with two adjacent ones, among which HTx\_LO is the main device and HTx\_HI is the slave device. Thus, the following interconnection structure can be obtained:

8 bit HT bus

8 bit HT bus

FIG. 10-2 interconnection structure of four-piece longson no. 3

HT0\_ LO

HT0\_

Hi8-bit HT bus

CPU2CPU3

HT0\_ HI

HT0\_ LO

HT0\_HT0\_

HILO

CPU1

CPU0

HT0\_ HI

HT 1

The 16-bit HT bus

IO

HT0\_

Lo8-bit HT bus

###### Loongson 3 interconnection routing

Loongson 3 interconnection routing adopts simple x-y routing method. That is, when routing, first X, then Y, take four chips as an example, ID

The Numbers are 0, 01, 10, 11. If a request is made from 11 to 00, it is routed from 11 to 00, first in the X direction,

Go from 11 to 10, then go in the Y direction, go from 10 to 00. When the request response returns from 00 to 11, the route goes first in the X direction, from 00 to 01, and then in the Y direction, from 01 to 11. As you can see, these are two different routing lines. Because of the characteristics of this algorithm, we will take a different approach when building the interconnection between two chips.

###### Two - piece loong core 3 interconnection structure

Because of the nature of the fixed routing algorithm, there are two different approaches to building a two-chip interconnection. The first is the use of 8-bit HT bus interconnection. In this mode of interconnection, only 8-bit HT interconnection can be used between two processors. The Numbers of the two chips are 00 and 01 respectively. According to the routing algorithm, we can know that when the two chips visit each other, they all pass through the 8-bit HT bus consistent with the four-chip interconnection. As follows:

8 bit HT bus

IO

The 16-bit HT bus

HT0\_ HI

HT0\_ LO

CPU0

HT 1

HT0\_ HI

CPU1

HT0\_ LO

Figure 10-3 8-bit interconnection structure of two pieces of loong core no. 3

However, our HT bus can adopt 16-bit mode at its widest, so the connection mode to maximize bandwidth should be 16-bit interconnection structure. In loongson iii, as long as the HT0 controller is set to 16-bit mode, all commands sent to the HT0 controller will be sent to HT0\_LO instead of to HT0\_HI or HT0\_LO according to the routing table as before, so that we can use the 16-bit bus when interconnecting. Therefore, we only need to properly configure the 16-bit mode of CPU0 and CPU1 and properly connect the high-low bit bus to interconnect using the 16-bit HT bus. This interconnection structure can also use the 8-bit HT bus protocol to access each other. The resulting interconnection structure is as follows:

IO

The 16-bit HT bus

The 16-bit HT bus

CPU1

HT0

|  |  |  |
| --- | --- | --- |
| HT 1 | CPU0 | HT0 |

Figure 10-4 16-bit interconnection structure of two pieces of loong core 3

## Low speed IO controller configuration

Loongson 3 I/O controller includes PCI controller, LPC controller, UART controller, SPI controller, GPIO and configuration register. These I/O controllers share a AXI port, and CPU requests are decoded to the corresponding device.

#### The PCI controller

Loongson 3's PCI controller can control the entire system either as a main bridge or as a normal PC device

On the PCI bus. Its implementation conforms to the PCI 2.3 specification. The loongson 3's PCI controller also has a PCI arbitrator built in.

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

Table 11-1 PCI controller configuration header

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| The byte 3 | 2 bytes | 1 byte | Byte 0 | address |
| The Device ID | | Vendor ID | | 00 |
| The Status | | The Command | | 04 |
| The Class Code | | | Revision ID | 08 |
| BIST | The Header Type | Latency Timer | CacheLine Size | 0 c |
| The Base Address Register 0 | | | | 10 |
| The Base Address Register 1 | | | | 14 |
| The Base Address Register 2 | | | | 18 |
| The Base Address Register 3 | | | | 1 c |
| The Base Address Register 4 | | | | 20 |
| The Base Address Register 5 | | | | 24 |
|  | | | | 28 |
| Subsystem ID | | Subsystem Vendor ID | | 2 c |
|  | | | | 30 |
|  | | | "Capabilities Pointer | 34 |
|  | | | | 38 |
| Maximum Latency | A Minimum Grant | Interrupt Pin | Interrupt Line | 3 c |
| Implementation Specific Register (ISR40) | | | | 40 |
| Implementation Specific Register (ISR44) | | | | 44 |
| Implementation Specific Register (ISR48) | | | | 48 |
| Implementation Specific Register (ISR4C) | | | | 4 c |
| Implementation Specific Register (ISR50) | | | | 50 |
| Implementation Specific Register (ISR54) | | | | 54 |
| Implementation Specific Register (ISR58) | | | | 58 |
|  | | | | . |
| PCIX Command Register | | | | E0 |
| PCIX Status Register | | | | E4 |

Loongson 3A2000's PCIX controller supports three 64-bit Windows, including {BAR1, BAR0}, {BAR3, BAR2},

{BAR5, BAR4} three pairs of register configuration Windows 0, 1, 2 base addresses. The size, enabling, and other details of the window are controlled by the other three corresponding registers, PCI\_Hit0\_Sel, PCI\_Hit1\_Sel, and PCI\_Hit2\_Sel. See table 2 for the specific bit fields.

Table 11-2 PCI control registers

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A domain** | **The field name** | **access** | **Reset value** | **instructions** |
| REG\_40 | | | | |
| 31 | tar\_read\_io | Read and write  (write 1 clear) | 0 | The target side receives access to IO or the unprefetched region |
| 30 | tar\_read\_discard | Read and write  (write 1 clear) | 0 | The delay request on the target side is discarded |
| 29 | tar\_resp\_delay | Read and write | 0 | Target accesses when to give delay/split 0: after timeout  1: immediately |
| 28 | tar\_delay\_retry | Read and write | 0 | Target accesses the retry policy  0: according to internal logic (see 29 bits)  1: try again immediately |
| 27 | tar\_read\_abort\_en | Read and write | 0 | If target times out for an internal read request, do you want to respond with target-abort |
| " | Reserved | - | 0 |  |
| 24 | tar\_write\_abort\_en | Read and write | 0 | If target times out for an internal write request, do you want to respond with target-abort |
| 23 | tar\_master\_abort | Read and write | 0 | Whether master-abort is allowed |
| Lift up | tar\_subseq\_timeout | Read and write | 000 | Target subsequent delay timeout  000:8 cycles  Others: not supported |
| He hath | tar\_init\_timeout | Read and write | 0000 | Target initial delay timeout  PCI mode  0:16 cycles  1-7: disable the counter  8-15:8 to 15 cycles  The timeout count is fixed at 8 cycles in PCIX mode, where the configuration has the greatest impact  Delay access number  0: 8 delay access  8: 1 delay access  9: 2 delay access  10: 3 delay access  11: 4 delay access |
|  |  |  |  | 12: 5 delay access  13: 6 delay access  14: 7 delay access  15: 8 delay access |
| Indeed, | tar\_pref\_boundary | Read and write | 000 h. | Prefetching boundary configuration (in 16 bytes)  FFF: 64KB to 16byte FFE: 64KB to 32byte  FF8:64 KB to 128 byte |
| 3 | tar\_pref\_bound\_en | Read and write | 0 | Configuration using tar\_pref\_boundary  0: prefetch to device boundary  1: using tar\_pref\_boundary |
| 2 | Reserved | - | 0 |  |
| 1 | tar\_splitw\_ctrl | Read and write | 0 | Target split write control  0: blocks access other than Posted Memory Write  1: block all access until split is completed |
| 0 | mas\_lat\_timeout | Read and write | 0 | Disable mater access timeout  0: allows master access timeout  1: not allowed |
| REG\_44 | | | | |
| 31:0 | Reserved | - | - |  |
| REG\_48 | | | | |
| 31:0 | tar\_pending\_seq | Read and write | 0 | The request number vector that target has not processed can be marked with 1 |
| REG\_4C | | | | |
| charm | Reserved | - | - |  |
| 29 | mas\_write\_defer | Read and write | 0 | Allow subsequent reads to override previous incomplete writes  (for PCI only) |
| 28 | mas\_read\_defer | Read and write | 0 | Allow subsequent reads and writes to override previous incomplete reads  (for PCI only) |
| 27 | mas\_io\_defer\_cnt | Read and write | 0 | The maximum number of external IO requests  Zero: by the control  1:1. |
| they | mas\_read\_defer\_cnt | Read and write | 010 | Master supports maximum number of external reads (for PCI only)  Zero: 8  1-7:1-7  Note: a dual address cycle access accounts for two items |
| Ephron; | err\_seq\_id | read-only | 00 h | The target/master error number |
| 15 | err\_type | read-only | 0 | Target /master error command type  Zero: |
| 14 | err\_module | read-only | 0 | Wrong module |
|  |  |  |  | 0: target  1: master |
| 13 | system\_error | Read and write | 0 | Target /master system error (write 1 clear) |
| 12 | data\_parity\_error | Read and write | 0 | Target /master data parity error (write 1 clear) |
| 11 | ctrl\_parity\_error | Read and write | 0 | Target /master address odd and even wrong (write 1 clear) |
| 10:0 | Reserved | - | - |  |
| REG\_50 | | | | |
| 31:0 | mas\_pending\_seq | Read and write | 0 | The request number vector that the master hasn't finished processing is going to be 1 |
| REG\_54 | | | | |
| 31:0 | mas\_split\_err | Read and write | 0 | Split returns the error request bit vector |
| REG\_58 | | | | |
| charm | Reserved | - | - |  |
| then | tar\_split\_priority | Read and write | 0 | Target split returns the priority  Zero is the highest, three is the lowest |
| But after | mas\_req\_priority | Read and write | 0 | The external priority of master  Zero is the highest, three is the lowest |
| 25 | Priority\_en | Read and write | 0 | Arbitrate algorithm (arbitrate between master's access and target's split return)  0: fixed priority  1: rotary |
| Whereupon certain | reserve | - | - |  |
| 17 | mas\_retry\_aborted | Read and write | 0 | Master retry cancel (write 1 clear) |
| 16 | mas\_trdy\_timeout | Read and write | 0 | Master TRDY timeout count |
| " | mas\_retry\_value | Read and write | 00 h | Number of master retries  0: infinite retry  1-255:1-255 |
| away | mas\_trdy\_count | Read and write | 00 h | Master TRDY timeout counter  0: disable  1-255:1-255 |

Before initiating configuration space reads and writes, the application should configure the PCIMap\_Cfg register to tell the controller the type of configuration operation to initiate and the value on the high 16-bit address line. The configuration header for the corresponding device can then be accessed by reading and writing to the 2K space starting at 0x1fe80000. The device number is obtained by encoding from low to high priority according to PCIMap\_Cfg[15:0].

The configuration action address generation is shown in figure 11-1.

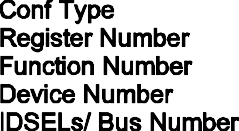


Figure 11-1 configure read and write bus address generation

PCI arbitrator realizes two-stage rotation arbitration, bus docking and isolation of damaged master devices.See its configuration and status

The PXArb\_Config and PXArb\_Status registers. PCI bus request and reply line assignment is shown in table 11-3.

Table 11-3 PCI/PCIX bus request and reply line allocation

|  |  |
| --- | --- |
| Request and reply lines | describe |
| 0 | Internal integrated PCI/PCIX controller |
| 7:1 | External requests 6~0 |

The route-based arbitration algorithm provides two levels, the second level as a whole as a member of the first level scheduling together. When multiple devices apply for the bus at the same time, the first level device is transferred once per cycle, and the highest priority device in the second level can get the bus.

The mediators are designed to be switched whenever conditions permit, which can make them abnormal for some PCI devices that do not conform to the protocol. Using a forced priority allows these devices to occupy the bus through continuous requests.

Bus docking refers to whether one of the devices is selected to give a permit signal when no device requests to use the bus. Directly initiating bus operations can improve efficiency for devices that are already permitted. Internal PCI mediators provide two docking modes: the last master and the default master. If you can't dock on special occasions, you can set the mediator to dock to the default master device 0 (internal controller) with a dependency delay of 0.

#### LPC controller

LPC controller has the following characteristics:

* Comply with LPC1.1 specification
* Support for LPC access timeout counters
* Memory Read, Memory write access types are supported
* Firmware Memory Read, Firmware Memory Write access type (single byte)
* I/O read and I/O write access types are supported
* Support for Memory access type address translation
* Support for the Serizlized IRQ specification, providing 17 interrupt sources

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

Table 11-4 address space distribution of LPC controller

|  |  |  |
| --- | --- | --- |
| **Address name** | **Address range** | **The size of the** |
| LPC Boot | 0 x1fc0\_0000 x1fd0\_0000 0 | 1 mbyte |
| LPC Memory | 0 x1c00\_0000 x1d00\_0000 0 | 16 mbyte |
| LPC I/O | 0 x1ff0\_0000 x1ff1\_0000 0 | 64 kbyte |
| LPC Register | 0 x1fe0\_0200 x1fe0\_0300 0 | 256 byte |

The LPC Boot address space is the address space first accessed by the processor when the system is started. When PCI\_CONFIG[0] pin is pulled down, the address of 0xBFC00000 is automatically routed to the LPC. This address space supports LPC Memory or Firmware Memory access types. The type of access emitted at system startup is controlled by LPC\_ROM\_INTEL pins. LPC\_ROM\_INTEL pins send out LPC Firmware Memory access when pulled up, and LPC\_ROM\_INTEL pins send out LPC Memory access type when pulled down.

The LPC Memory address space is the address space that the system accesses with Memory/Firmware Memory.The type of Memory access issued by an LPC controller is determined by the LPC controller's configuration register, LPC\_MEM\_IS\_FWH.The address sent by the processor to this address space can be translated. The converted address is set by the LPC controller's configuration register LPC\_MEM\_TRANS.

The access that the processor sends to the LPC I/O address space is sent to the LPC bus according to the LPC I/O access type. The address is 16 bits below the address space.

The LPC controller configuration registers have a total of 3 32-bit registers. The meaning of the configuration register is shown in table 11-5:

Table 11-5 LPC configuration register meanings

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A domain** | **The field name** | **access** | **Reset value** | **instructions** |
| REG0 | | | | |
| REG0 [hands] | SIRQ\_EN | Read and write | 0 | SIRQ enables control |
| REG0 [take] | LPC\_MEM\_TRANS | Read and write | 0 | LPC Memory space address translation control |
| REG0 [15:0] | LPC\_SYNC\_TIMEOUT | Read and write | 0 | LPC access timeout counter |
| REG1 | | | | |
| REG1 [hands] | LPC\_MEM\_IS\_FWH | Read and write | 0 | LPC Memory space Firmware  Memory access type Settings |
| REG1 [17:0] | LPC\_INT\_EN | Read and write | 0 | LPC SIRQ interrupt enablement |
| REG2 | | | | |
| REG2 [17:0] | LPC\_INT\_SRC | Read and write | 0 | LPC SIRQ interrupts source indication |
| REG3 | | | | |
| REG3 [17:0] | LPC\_INT\_CLEAR | write | 0 | LPC SIRQ interrupt clear |

#### UART controller

UART controllers have the following characteristics

* Full duplex asynchronous data receive/send
* Programmable data format
* 16 bit programmable clock counter
* Receive timeout detection is supported
* A multi-interrupt system with arbitration
* Work in FIFO mode only
* Register and function compatible NS16550A

Chip internal integration of two UART interfaces, function registers are exactly the same, but access to different base address.

The physical base address of the UART0 register is 0x1FE001E0. The physical base address of UART1 register is 0x1FE001E8.

* + 1. **Data register (DAT)**

Chinese name: data transfer register register bit width: [7:0]

Offset: 0x00

Reset value: 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe |
| away | Tx FIFO | 8 | W. | Data transfer register |

### Interrupt enabled register (IER)

Chinese name: interrupt enabled register register bit width: [7:0]

Offset: 0x01

Reset value: 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe |
| The log | Reserved | 4 | RW | reserve |
| 3 | IME | 1 | RW | Modem status interrupted to enable '0' -- off '1' -- on |
| 2 | ILE | 1 | RW | Receiver line status interruption enables' 0 '-- off' 1 '-- on |
| 1 | ITxE | 1 | RW | Transfer save register for air break enable '0' -- close '1' -- open |
| 0 | IRxE | 1 | RW | Receive valid data interrupt enables' 0 '-- close' 1 '-- open |

### Interrupt identification register (IIR)

Chinese name: interrupt source register register bit width: [7:0]

Offset: 0x02

Reset value: 0xc1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe |
| The log | Reserved | 4 | R | reserve |
| 3:1 | II | 3 | R | Interrupt source representation bits, as shown in the table below |
| 0 | INTp | 1 | R | Interrupt representation bit |

Interrupt control menu

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Bit 3 | 2 - | Bit 1 | priority | Interrupt type | The interrupt source | Interrupt reset control |
| 0 | 1 | 1 | 1 st | Receiving line status | Odd or even, overflow or frame error, or hit  Break the interrupt | Read the LSR |
| 0 | 1 | 0 | 2 nd | A significant number is received  According to the | The number of characters of FIFO is reached  The level of the trigger | FIFO has a low number of characters  In the trigger value |
| 1 | 1 | 0 | 2 nd | Receive a timeout | There is at least one character in FIFO,  But there are no operations, including read and write operations, in 4 character time | Read receive FIFO |
| 0 | 0 | 1 | 3 rd | Transfer save hosting  Device is empty | The transfer save register is empty | Write data to THR or  More IIR |
| 0 | 0 | 0 | 4 th | Modem state | CTS, DSR, RI or DCD. | Read MSR |

### FIFO control register (FCR)

Chinese name: FIFO control register register bit width: [7:0]

Offset: 0x02

Reset value: 0xc0

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe |
| but | TL | 2 | W. | Receive the trigger value of FIFO's interrupt application  '00' -- 1 byte '01' -- 4 bytes  '10' -- 8 bytes' 11 '-- 14 bytes |
| o | Reserved | 3 | W. | reserve |
| 2 | Txset | 1 | W. | '1' clears the content of sending FIFO and reset its logic |
| 1 | Rxset | 1 | W. | '1' clears the contents of the received FIFO and reset its logic |
| 0 | Reserved | 1 | W. | reserve |

### Line control register (LCR)

Chinese name: circuit control register register bit width: [7:0]

Offset: 0x03

Reset value: 0x03

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe | | | |
| 7 | dlab | 1 | RW | Frequency divider latch access bit  '1' - access operation frequency divider latch  '0' - access operation normal register | | | |
| 6 | BCB | 1 | RW | Interrupt control bit  '1' - the output of the serial port is set to 0(interrupted state). '0' - normal operation | | | |
| 5 | .spb | 1 | RW | Specifies parity bits  '0' - do not specify parity bits  '1' - if the LCR[4] bit is 1, the transmission and check parity bits are 0. If the LCR[4] bit is 0, the transmission and check parity bit is 1. | | | |
| 4 | eps | 1 | RW | Parity bit selection  '0' - has an odd number of 1's in each character (including data and parity bits)  '1' - there are an even number of 1's in each character | | | |
| 3 | PE | 1 | RW | Parity bit enabled  '0' - no parity bits  '1' - the parity bit is generated on the output, and the parity bit is judged on the input | | | |
| 2 | sb | 1 | RW | Defines the number of bits that generate the stop bit  '0' - 1 stop bit  '1' - 1.5 stop bits at 5 character length, others  The length is 2 stop bits | | | |
| 1-0 | bec | 2 | RW | Sets the number of bits per character  '00' -- 5 '01' -- 6 | | | |
|  |  |  |  | "10" - 7 | position | "11" - 8 | position |

### MODEM control register (MCR)

Chinese name: Modem control register register bit width: [7:0]

Offset: 0x04

Reset value: 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe |
| 7:5 | Reserved | 3 | W. | reserve |
| 4 | Loop | 1 | W. | Loop mode control bit  '0' - normal operation  '1' - loop mode.In loopback mode, TXD output is always 1, and the output shift register is connected directly to the input shift register.The other links are as follows.  DTR  DSR RTS  CTS  Out1  RI  Out2  DCD |
| 3 | OUT2 | 1 | W. | Connect to DCD input in loopback mode |
| 2 | The OUT1 | 1 | W. | Connect to RI input in loop mode |
| 1 | RTSC | 1 | W. | RTS signal control bit |
| 0 | DTRC | 1 | W. | DTR signal control bit |

### Line status register (LSR)

Chinese name: line status register register bit width: [7:0]

Offset: 0x05

Reset value: 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe |
| 7 | The ERROR | 1 | R | Error representation bit  '1' - at least one with a parity bit error, frame error, or interrupt.  '0' - no errors |
| 6 | TE | 1 | R | The null transport represents the bit  '1' - transfer FIFO and transfer shift registers are empty. Zero when writing data to the transmitted FIFO  '0' -- has data |
| 5 | TFE | 1 | R | Transmission of FIFO bit empty represents bit  '1' -- the current transmission FIFO is empty, and when data is written to the transmission FIFO, it will be zero  '0' -- has data |
| 4 | BI | 1 | R | Interrupts interrupt bits  '1' - received start bit + data + odd and even bit + stop bit are all 0, that is, there is interrupt interrupt  '0' -- no interruptions |
| 3 | FE | 1 | R | Frame errors represent bits  '1' - received data without stop bits  '0' - no errors |
| 2 | PE | 1 | R | A parity bit error represents a bit  '1' - there is a parity error in the received data  '0' - no parity errors |
| 1 | OE | 1 | R | Data overflow represents bits  '1' -- data overflow  '0' - no overflow |
| 0 | Dr. | 1 | R | Receiving data effectively represents bits  '0' - no data in a FIFO |
|  |  |  |  | '1' - data in FIFO |

When this register is read, LSR[4:1] and LSR[7] are cleared, LSR[6:5] is cleared when writing data to transmit FIFO, and LSR[0] judges the receiving FIFO.

### MODEM status register (MSR)

Chinese name: Modem status register register bit width: [7:0]

Offset: 0x06

Reset value: 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe |
| 7 | CDCD | 1 | R | DCD input the inverse of the value, or in loop mode to Out2 |
| 6 | CRI | 1 | R | The inverse of the RI input value, or connected to OUT1 in loopback mode |
| 5 | CDSR | 1 | R | The inverse of the DSR input value, or in loopback mode connected to DTR |
| 4 | CCTS | 1 | R | Invert the input value of the CTS, or connect to the RTS in loopback mode |
| 3 | DDCD | 1 | R | DDCD indicating a |
| 2 | TERI | 1 | R | RI edge detection. RI state changes from low to high |
| 1 | DDSR | 1 | R | DDSR indicating a |
| 0 | DCTS | 1 | R | DCTS indicating a |

### Frequency divider latch

Chinese name: frequency division latch 1

Register bit width: [7:0]

Offset: 0x00

Reset value: 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe |
| away | LSB | 8 | RW | Store the lower 8 bits of the frequency divider latch |

Chinese name: frequency division latch 2

Register bit width: [7:0]

Offset: 0x01

Reset value: 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe |
| away | The MSB | 8 | RW | Store the high 8 bits of the frequency divider latch |

#### SPI controller

SPI controller has the following characteristics:

* Full duplex synchronous serial port data transmission
* Support for variable-length byte transfers up to 4
* Master mode support
* A mode failure generates an error flag and issues an interrupt request
* Double buffer receiver
* Polarity and phase programmable serial clock
* SPI can be controlled in wait mode
* Support for starting from SPI

The SPI controller register physical address is 0x1FE00220.

Table 11-6 SPI controller address space distribution

|  |  |  |
| --- | --- | --- |
| **Address name** | **Address range** | **The size of the** |
| SPI Boot | 0 x1fc0\_0000 x1fd0\_0000 0 | 1 mbyte |
| SPI Memory | 0 x1d00\_0000 x1e00\_0000 0 | 16 mbyte |
| SPI Register | 0 x1fe0\_0220 x1fe0\_0230 0 | 16 byte |

SPI Boot address space is the address space first accessed by the processor when the system is started. When PCI\_CONFIG[0] pin is pulled up, the address of 0xBFC00000 is automatically routed to SPI.

The SPI Memory space can also be accessed directly through CPU read requests, with a minimum of 1M bytes overlapping the SPI BOOT space.

### Control register (SPCR)

Chinese name: control register register bit width: [7:0]

Offset: 0x00

Reset value: 0x10

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe |
| 7 | Spie | 1 | RW | Interrupt output to make the signal highly efficient |
| 6 | The spe | 1 | RW | The system works to make the signal highly efficient |
| 5 | Reserved | 1 | RW | reserve |
| 4 | MSTR | 1 | RW | Select bit in master mode, this bit is always 1 |
| 3 | cpol | 1 | RW | Clock polarity |
| 2 | cpha | 1 | RW | Clock phase 1 is in opposite phase and 0 is the same |
| 1-0 | SPR | 2 | RW | Sclk\_o frequency divider setting, to be used with sper's spre |

### Status register (SPSR)

Chinese name: status register register bit width: [7:0]

Offset: 0x01

Reset value: 0x05

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe |
| 7 | spif | 1 | RW | The interrupt sign bit 1 indicates that there is an interrupt request, and if you write 1, it will clear zero |
| 6 | wcol | 1 | RW | A write register overflow bit of 1 indicates overflow, and a write of 1 clears |
| when | Reserved | 2 | RW | reserve |
| 3 | wffull | 1 | RW | Write register full flag 1 to indicate full |
| 2 | wfempty | 1 | RW | Write register empty flag 1 to indicate empty |
| 1 | rffull | 1 | RW | Read register full flag 1 to indicate full |
| 0 | rfempty | 1 | RW | Read register empty flag 1 to indicate empty |

### Data register (TxFIFO)

Chinese name: data transfer register register bit width: [7:0]

Offset: 0x02

Reset value: 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe |
| away | Tx FIFO | 8 | W. | Data transfer register |

### External register (SPER)

Chinese name: external register register bit width: [7:0]

Offset: 0x03

Reset value: 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe |
| but | icnt | 2 | RW | How many bytes have been transmitted before the interrupt request signal is sent  00 -- 1 byte 01 -- 2 bytes  10-3 bytes 11-3 bytes |
| 5-2 | Reserved | 4 | RW | reserve |
| 1-0 | spre | 2 | RW | Set the frequency division ratio with the Spr |

Frequency division coefficient:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| spre  SPR | 00  00 | 00  01 | 00  10 | 00  11 | 01  00 | 01  01 | 01  10 | 01  11 | 10  00 | 10  01 | 10  10 | 10  11 |
| Frequency division coefficient | 2 | 4 | 16 | 32 | 8 | 64 | 128 | 256 | 512 | 1024 | 2048 | 4096 |

### Parameter control register (SFC\_PARAM)

SPI Flash parameter control register register bit width: [7:0]

Offset: 0x04

Reset value: 0x21

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **access** | **describe** |
| The log | clk\_div | 4 | RW | Clock frequency selection (frequency division coefficient is the same as {spre, SPR} combination) |
| 3 | dual\_io | 1 | RW | With dual I/O mode, priority is given to fast read mode |
| 2 | fast\_read | 1 | RW | Use quick read mode |
| 1 | burst\_en | 1 | RW | Spi flash supports sequential address reading mode |
| 0 | memory\_en | 1 | RW | Spi flash read enable, invalid CSN [0] can be controlled by the software. |

### Chip selection control register (SFC\_SOFTCS)

SPI Flash chip selection control register register bit width: [7:0]

Offset: 0x05

Reset value: 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **access** | **describe** |
| The log | CSN | 4 | RW | CSN pin output value |
| 3-0 | csen | 4 | RW | When is 1, the corresponding bit cs line is controlled by 7:4 bit |

|  |  |  |
| --- | --- | --- |
| **address** | **register** | **instructions** |
| 00 | PonCfg | The electric configuration |
| 04 | GenCfg | General configuration |
| 08 | reserve |  |
| 0 c | reserve |  |
| 10 | PCIMap | PCI mapping |
| 14 | PCIX\_Bridge\_Cfg | PCI/X bridge related configuration |
| 18 | PCIMap\_Cfg | PCI concatenates the read and write device address |
| 1 c | GPIO\_Data | GPIO data |
| 20 | GPIO\_EN | GPIO direction |
| 24 | reserve |  |
| 28 | reserve |  |
| 2 c | reserve |  |
| 30 | reserve |  |
| 34 | reserve |  |
| 38 | reserve |  |
| 3 c | reserve |  |
| 40 | Mem\_Win\_Base\_L | Prefetch window base address low 32 bits |
| 44 | Mem\_Win\_Base\_H | Prefetch window base address height 32 bits |
| 48 | Mem\_Win\_Mask\_L | Prefetch window mask lower 32 bits |
| 4 c | Mem\_Win\_Mask\_H | Prefetch window mask height 32 bits |
| 50 | PCI\_Hit0\_Sel\_L | PCI window 0 controls low 32 bits |

### Timing control register (SFC\_TIMING)

SPI Flash timing control register register bit width: [7:0]

Offset: 0x06

Reset value: 0x03

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **access** | **describe** |
| Booths, | Reserved | 6 | RW | reserve |
| 1-0 | it | 2 | RW | The minimum invalid time of SPI Flash is calculated by clock period T after frequency division  00:1 t  01:2 t  10:4 t  11:8 t |

#### IO controller configuration

The configuration register is used to configure the address window of the PCI controller, the arbitrator, and the GPIO controller.[Table 11-6](#_bookmark233)

These registers are listed and detailed descriptions of them are given in tables 11-7. This part of the register is base address 0x1FE00100.  Table 11-7 IO control registers

|  |  |  |
| --- | --- | --- |
| 54 | PCI\_Hit0\_Sel\_H | PCI window 0 controls the height of 32 bits |
| 58 | PCI\_Hit1\_Sel\_L | PCI window 1 controls the low 32-bit |
| 5 c | PCI\_Hit1\_Sel\_H | PCI window 1 controls the high 32-bit |
| 60 | PCI\_Hit2\_Sel\_L | PCI window 2 controls low 32 bits |
| 64 | PCI\_Hit2\_Sel\_H | PCI window 2 controls the high 32-bit |
| 68 | PXArb\_Config | PCIX arbitrator configuration |
| 6 c | PXArb\_Status | PCIX arbitrator state |
| 70 |  |  |
| 74 |  |  |
| 78 |  |  |
| 7 c |  |  |
| 80 | Chip Config | Chip configuration register |
| 84 |  |  |
| 88 |  |  |
| 8 c |  |  |
| 90 | Chip Sample | Chip sampling register |

Table 11-8 registers are described in detail

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **A domain** | **The field name** | **access** | | **Reset value** | **instructions** | |
| CR00: PonCfg | | | | | | |
| 15:0 | pcix\_bus\_dev | read-only | | Lio\_ad [away] | The total amount used by the CPU in the PCIX Agent mode  Line and equipment number | |
| " | reserve | read-only | | Lio\_ad [or] |  | |
| Ephron; | pon\_pci\_configi | read-only | | pci\_configi | PCI\_Configi pin value | |
| came | reserve | read-only | |  |  | |
| CR04: reserve | | | | | | |
| 31:0 | reserve | read-only | | 0 |  | |
| CR08: reserve | | | | | | |
| 31:0 | reserve | read-only | | 0 |  | |
| CR10: PCIMap | | | | | | |
| 5-0 | trans\_lo0 | Read and write | | 0 | The PCI\_Mem\_Lo0 window maps the address six bits higher | |
| but | trans\_lo1 | Read and write | | 0 | The PCI\_Mem\_Lo1 window maps the address six bits higher | |
| " | trans\_lo2 | Read and write | 0 | | | The PCI\_Mem\_Lo2 window maps the address to a height of 6 bits |
| all | reserve | read-only | 0 | | |  |
| CR14: PCIX\_Bridge\_Cfg | | | | | | |
| 5-0 | pcix\_rgate | Read and write | 6 'h18 | | | PCIX mode to DDR2 read number threshold |
| 6 | pcix\_ro\_en | Read and write | 0 | | | The PCIX bridge allows writing over reading |
| all | reserve | read-only | 0 | | |  |
| CR18: PCIMap\_Cfg | | | | | | |
| 15:0 | dev\_addr | Read and write | 0 | | | PCI configuration reads and writes when the AD line is 16 bits high |
| 16 | conf\_type | Read and write | 0 | | | Configure the types of reads and writes |
| 31:17 | reserve | read-only | 0 | | |  |
| CR1C: GPIO\_Data | | | | | | |
| 15:0 | gpio\_out | Read and write | 0 | | | GPIO outputs data |
| Caused the | gpio\_in | Read and write | 0 | | | GPIO enters data |
| CR20: GPIO\_EN | | | | | | |
| 15:0 | gpio\_en | Read and write | FFFF | | | High is the input and low is the output |
| Caused the | reserve | read-only | 0 | | |  |
| CR3C: reserve | | | | | | |
| 31:0 | reserve | read-only | 0 | | | reserve |
| CR24, 2 c, 30,34,38: reservations | | | | | | |
| As shown in table 11 | | | | | | |
| CR50, 60 (54/58, 5 C / : \_Sel\_ PCI\_Hit \* \* | | | | | | |
| 0 | reserve | read-only | 0 | | |  |
| 2:1 | pci\_img\_size | Read and write | 2 'bl1 | | | 00:32; 10:64; Others: invalid |
| 3 | pref\_en | Read and write | 0 | | | Prefetching can make |
| 4 | reserve | read-only | 0 | | |  |
| 62:12 | bar\_mask | Read and write | 0 | | | Window size mask (high 1, low 0) |
| 63 | burst\_cap | Read and write | 1 | | | Whether to allow burst transmission |
| CR68: PXArb\_Config | | | | | | |
| 0 | device\_en | Read and write | 1 | | | External device allowed |
| 1 | disable\_broken | Read and write | 0 | | | Disable damaged master devices |
| 2 | default\_mas\_en | Read and write | 1 | | | The bus is docked to the default main device  0: dock to the last main device  1: dock to the default main device |
| o | default\_master | Read and write | 0 | | | The bus is docked with the default main device number |
| but | park\_delay | Read and write | 2 'bl1 | | | The delay from the beginning of no device request bus to the triggering of docked default device behavior  00:0 cycle |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  | 01:8 cycles  10:32 cycle  11:128 cycles |
| " | level | Read and write | 8 'h01-2 | A device in the first stage |
| Ephron; | rude\_dev | Read and write | 0 | Force priority devices  The PCI device corresponding to the bit of 1 can occupy the bus by continuous request after obtaining the bus |
| away | reserve | read-only | 0 |  |
| CR6C: PXArb\_Status | | | | |
| away | broken\_master | read-only | 0 | Damaged master device (reset when changing disabled policy) |
| 10:8 | Last\_master | read-only | 0 | Finally, the main device of the bus is used |
| lustful | reserve | read-only | 0 |  |
| CR80: Chip config (see section 2.6) | | | | |
| CR90: Chip Sample (see section 2.6) | | | | |
| CRA0: Chip Sample (see section 2.6) | | | | |
| CRB0: PLL config (see section 2.6) | | | | |
| CRC0: PLL config (see section 2.6) | | | | |
| CRD0: Core config (see section 2.6) | | | | |

##### 



## List of chip configuration registers

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| The Name | ADDR | R/W | Description(NULL means no effect) | The default value |
| CPU\_WIN0\_BASE | 0 x3ff00000 | RW | Base address of CPU window 0 | 0 x0 |
| CPU\_WIN1\_BASE | 0 x3ff00008 | RW | Base address of CPU window 1 | 0 x1000\_0000 |
| CPU\_WIN2\_BASE | 0 x3ff00010 | RW | Base address of CPU window 2 | 0 x1000\_8000\_0000 |
| CPU\_WIN3\_BASE | 0 x3ff00018 | RW | Base address of CPU window 3 | 0 x0 |
| CPU\_WIN4\_BASE | 0 x3ff00020 | RW | Base address of CPU window 4 | 0 x0 |
| CPU\_WIN5\_BASE | 0 x3ff00028 | RW | Base address of CPU window 5 | 0 x0 |
| CPU\_WIN6\_BASE | 0 x3ff00030 | RW | Base address of CPU window 6 | 0 x0 |
| CPU\_WIN7\_BASE | 0 x3ff00038 | RW | The base address of CPU window 7 | 0 x0 |
| CPU\_WIN0\_MASK | 0 x3ff00040 | RW | The mask for CPU window 0 | 0 xffff\_ffff\_f000\_0000 |
| CPU\_WIN1\_MASK | 0 x3ff00048 | RW | Mask for CPU window 1 | 0 xffff\_ffff\_f000\_0000 |
| CPU\_WIN2\_MASK | 0 x3ff00050 | RW | CPU window 2 mask | 0 xffff\_ffff\_f000\_0000 |
| CPU\_WIN3\_MASK | 0 x3ff00058 | RW | Mask for CPU window 3 | 0 x0 |
| CPU\_WIN4\_MASK | 0 x3ff00060 | RW | Mask for CPU window 4 | 0 x0 |
| CPU\_WIN5\_MASK | 0 x3ff00068 | RW | Mask for CPU window 5 | 0 x0 |
| CPU\_WIN6\_MASK | 0 x3ff00070 | RW | Mask for CPU window 6 | 0 x0 |
| CPU\_WIN7\_MASK | 0 x3ff00078 | RW | CPU window 7 mask | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CPU\_WIN0\_MMAP | 0 x3ff00080 | RW | New base address for CPU window 0 | 0 xf0 |
| CPU\_WIN1\_MMAP | 0 x3ff00088 | RW | New base address for CPU window 1 | 0 x1000\_00f2 |
| CPU\_WIN2\_MMAP | 0 x3ff00090 | RW | New base address for CPU window 2 | 0 xf0 |
| CPU\_WIN3\_MMAP | 0 x3ff00098 | RW | New base address for CPU window 3 | 0 x0 |
| CPU\_WIN4\_MMAP | 0 x3ff000a0 | RW | New base address for CPU window 4 | 0 x0 |
| CPU\_WIN5\_MMAP | 0 x3ff000a8 | RW | New base address for CPU window 5 | 0 x0 |
| CPU\_WIN6\_MMAP | 0 x3ff000b0 | RW | New base address for CPU window 6 | 0 x0 |
| CPU\_WIN7\_MMAP | 0 x3ff000b8 | RW | New base address for CPU window 7 | 0 x0 |
| PCI\_WIN0\_BASE | 0 x3ff00100 | RW | Base address for PCI window 0 | 0 x8000\_0000 |
| PCI\_WIN1\_BASE | 0 x3ff00108 | RW | Base address for PCI window 1 | 0 x0 |
| PCI\_WIN2\_BASE | 0 x3ff00110 | RW | PCI window 2 base address | 0 x0 |
| PCI\_WIN3\_BASE | 0 x3ff00118 | RW | Base address for PCI window 3 | 0 x0 |
| PCI\_WIN4\_BASE | 0 x3ff00120 | RW | Base address for PCI window 4 | 0 x0 |
| PCI\_WIN5\_BASE | 0 x3ff00128 | RW | Base address for PCI window 5 | 0 x0 |
| PCI\_WIN6\_BASE | 0 x3ff00130 | RW | Base address for PCI window 6 | 0 x0 |
| PCI\_WIN7\_BASE | 0 x3ff00138 | RW | Base address for PCI window 7 | 0 x0 |
| PCI\_WIN0\_MASK | 0 x3ff00140 | RW | Mask for PCI window 0 | 0 xffff\_ffff\_8000\_0000 |
| PCI\_WIN1\_MASK | 0 x3ff00148 | RW | Mask for PCI window 1 | 0 x0 |
| PCI\_WIN2\_MASK | 0 x3ff00150 | RW | Mask for PCI window 2 | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PCI\_WIN3\_MASK | 0 x3ff00158 | RW | Mask for PCI window 3 | 0 x0 |
| PCI\_WIN4\_MASK | 0 x3ff00160 | RW | Mask for PCI window 4 | 0 x0 |
| PCI\_WIN5\_MASK | 0 x3ff00168 | RW | Mask for PCI window 5 | 0 x0 |
| PCI\_WIN6\_MASK | 0 x3ff00170 | RW | Mask for PCI window 6 | 0 x0 |
| PCI\_WIN7\_MASK | 0 x3ff00178 | RW | Mask for PCI window 7 | 0 x0 |
| PCI\_WIN0\_MMAP | 0 x3ff00180 | RW | New base address for PCI window 0 | 0 xf0 |
| PCI\_WIN1\_MMAP | 0 x3ff00188 | RW | New base address for PCI window 1 | 0 x0 |
| PCI\_WIN2\_MMAP | 0 x3ff00190 | RW | New base address for PCI window 2 | 0 x0 |
| PCI\_WIN3\_MMAP | 0 x3ff00198 | RW | New base address for PCI window 3 | 0 x0 |
| PCI\_WIN4\_MMAP | 0 x3ff001a0 | RW | New base address for PCI window 4 | 0 x0 |
| PCI\_WIN5\_MMAP | 0 x3ff001a8 | RW | New base address for PCI window 5 | 0 x0 |
| PCI\_WIN6\_MMAP | 0 x3ff001b0 | RW | New base address for PCI window 6 | 0 x0 |
| PCI\_WIN7\_MMAP | 0 x3ff001b8 | RW | New base address for PCI window 7 | 0 x0 |
| Slock0\_addr | 0 x3ff00200 | RW | Lock address of lock window no. 0 ([63]: valid, [47:0]: addr) | 0 x0 |
| Slock1\_addr | 0 x3ff00208 | RW | Lock address of no. 1 lock window ([63]: valid, [47:0]: addr) | 0 x0 |
| Slock2\_addr | 0 x3ff00210 | RW | Lock address of no. 2 lock window ([63]: valid, [47:0]: addr) | 0 x0 |
| Slock3\_addr | 0 x3ff00218 | RW | Lock address of no. 3 lock window ([63]: valid, [47:0]: addr) | 0 x0 |
| Slock0\_mask | 0 x3ff00240 | RW | No. 0 lock window mask ([47:0]: mask) | 0 x0 |
| Slock1\_mask | 0 x3ff00248 | RW | No. 1 lock window mask ([47:0]: mask) | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Slock2\_mask | 0 x3ff00250 | RW | No. 2 lock window mask ([47:0]: mask) | 0 x0 |
| Slock3\_mask | 0 x3ff00258 | RW | No. 3 lock window mask ([47:0]: mask) | 0 x0 |
| BARRIER\_SET | 0 x3ff00300 | send | The barrier value plus one |  |
| BARRIER\_CLR | 0 x3ff00308 | send | The barrier value minus 1 |  |
| BARRIER\_REF | 0 x3ff00310 | RW | The barrier threshold | 0 x0 |
| BARRIER\_CTRL | 0 x3ff00318 | RW | Bit [0]: add or subtract enablement /barrier interrupt enablement | 0 x0 |
| BARRIER\_VEC | 0 x3ff00320 | RO | The current values of the barrier |  |
| CONFSIGNAL\_CR | 0 x3ff00400 | RW | 24: ccsd\_en 19:16: ccsd\_id 8: xrouter\_en 5: x2\_pci\_rdinterleave 4: x2\_cpu\_rdinterleave  3-0: scid\_sel | 0 xffff\_0000 |
| gs3\_HPT | 0 x3ff00408 | RO | A counter that adds 1 to each clock cycle |  |
| MTX0\_SRC\_START\_ADDR | 0 x3ff00600 | RW |  | 0 x0 |
| MTX0\_DST\_START\_ADDR | 0 x3ff00608 | RW |  | 0 x0 |
| MTX0\_ORI\_LENTH | 0 x3ff00610 | RW |  | 0 x0 |
| MTX0\_ORI\_WIDTH | 0 x3ff00618 | RW |  | 0 x0 |
| MTX0\_SRC\_ROW\_STRIDE | 0 x3ff00620 | RW |  | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MTX0\_DST\_ROW\_STRIDE | 0 x3ff00628 | RW |  | 0 x0 |
| MTX0\_TRANS\_CTRL | 0 x3ff00630 | RW |  | 0 x0 |
| MTX1\_SRC\_START\_ADDR | 0 x3ff00700 | RW |  | 0 x0 |
| MTX1\_DST\_START\_ADDR | 0 x3ff00708 | RW |  | 0 x0 |
| MTX1\_ORI\_LENTH | 0 x3ff00710 | RW |  | 0 x0 |
| MTX1\_ORI\_WIDTH | 0 x3ff00718 | RW |  | 0 x0 |
| MTX1\_SRC\_ROW\_STRIDE | 0 x3ff00720 | RW |  | 0 x0 |
| MTX1\_DST\_ROW\_STRIDE | 0 x3ff00728 | RW |  | 0 x0 |
| MTX1\_TRANS\_CTRL | 0 x3ff00730 | RW |  | 0 x0 |
| SCache0\_perfctrl0 | 0 x3ff00800 | RW |  |  |
| SCache0\_perfcnt0 | 0 x3ff00808 | RO |  |  |
| SCache0\_perfctrl1 | 0 x3ff00810 | RW |  |  |
| SCache0\_perfcnt1 | 0 x3ff00818 | RO |  |  |
| SCache0\_perfctrl2 | 0 x3ff00820 | RW |  |  |
| SCache0\_perfcnt2 | 0 x3ff00828 | RO |  |  |
| SCache0\_perfctrl3 | 0 x3ff00830 | RW |  |  |
| SCache0\_perfcnt3 | 0 x3ff00838 | RO |  |  |
| SCache1\_perfctrl0 | 0 x3ff00900 | RW |  |  |
| SCache1\_perfcnt0 | 0 x3ff00908 | RO |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SCache1\_perfctrl1 | 0 x3ff00910 | RW |  |  |
| SCache1\_perfcnt1 | 0 x3ff00918 | RO |  |  |
| SCache1\_perfctrl2 | 0 x3ff00920 | RW |  |  |
| SCache1\_perfcnt2 | 0 x3ff00928 | RO |  |  |
| SCache1\_perfctrl3 | 0 x3ff00930 | RW |  |  |
| SCache1\_perfcnt3 | 0 x3ff00938 | RO |  |  |
| SCache2\_perfctrl0 | 0 x3ff00a00 | RW |  |  |
| SCache2\_perfcnt0 | 0 x3ff00a08 | RO |  |  |
| SCache2\_perfctrl1 | 0 x3ff00a10 | RW |  |  |
| SCache2\_perfcnt1 | 0 x3ff00a18 | RO |  |  |
| SCache2\_perfctrl2 | 0 x3ff00a20 | RW |  |  |
| SCache2\_perfcnt2 | 0 x3ff00a28 | RO |  |  |
| SCache2\_perfctrl3 | 0 x3ff00a30 | RW |  |  |
| SCache2\_perfcnt3 | 0 x3ff00a38 | RO |  |  |
| SCache3\_perfctrl0 | 0 x3ff00b00 | RW |  |  |
| SCache3\_perfcnt0 | 0 x3ff00b08 | RO |  |  |
| SCache3\_perfctrl1 | 0 x3ff00b10 | RW |  |  |
| SCache3\_perfcnt1 | 0 x3ff00b18 | RO |  |  |
| SCache3\_perfctrl2 | 0 x3ff00b20 | RW |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SCache3\_perfcnt2 | 0 x3ff00b28 | RO |  |  |
| SCache3\_perfctrl3 | 0 x3ff00b30 | RW |  |  |
| SCache3\_perfcnt3 | 0 x3ff00b38 | RO |  |  |
| Core0\_IPI\_Status | 0 x3ff01000 | RO | The IPI\_Status register for the number 0 processor core |  |
| Core0\_IPI\_Enalbe | 0 x3ff01004 | RW | The IPI\_Enalbe register of the number 0 processor core | 0 x0 |
| Core0\_IPI\_Set | 0 x3ff01008 | send | The IPI\_Set register of the number 0 processor core |  |
| Core0\_IPI\_Clear | 0 x3ff0100c | send | The IPI\_Clear register for the number 0 processor core |  |
| Core0\_MailBox0 | 0 x3ff01020 | RW | Register IPI\_MailBox0 of the number 0 processor core | 0 x0 |
| Core0\_MailBox1 | 0 x3ff01028 | RW | Register IPI\_MailBox1 of processor core 0 | 0 x0 |
| Core0\_MailBox2 | 0 x3ff01030 | RW | Register IPI\_MailBox2 for the number 0 processor core | 0 x0 |
| Core0\_MailBox3 | 0 x3ff01038 | RW | Register IPI\_MailBox3 of processor core 0 | 0 x0 |
| Core0\_int\_interval | 0 x3ff01060 | RW |  |  |
| Core0\_int\_compare | 0 x3ff01068 | RW |  |  |
| Core1\_IPI\_Status | 0 x3ff01100 | RO | The IPI\_Status register for the number 1 processor core |  |
| Core1\_IPI\_Enalbe | 0 x3ff01104 | RW | The IPI\_Enalbe register of the number 1 processor core | 0 x0 |
| Core1\_IPI\_Set | 0 x3ff01108 | send | The IPI\_Set register of the number 1 processor core |  |
| Core1\_IPI\_Clear | 0 x3ff0110c | send | The IPI\_Clear register of the number 1 processor core |  |
| Core1\_MailBox0 | 0 x3ff01120 | RW | Register IPI\_MailBox0 for the number 1 processor core | 0 x0 |
| Core1\_MailBox1 | 0 x3ff01128 | RW | Register IPI\_MailBox1 of the number 1 processor core | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Core1\_MailBox2 | 0 x3ff01130 | RW | The IPI\_MailBox2 register of the number 1 processor core | 0 x0 |
| Core1\_MailBox3 | 0 x3ff01138 | RW | Register IPI\_MailBox3 for the number 1 processor core | 0 x0 |
| Core1\_int\_interval | 0 x3ff01160 | RW |  |  |
| Core1\_int\_compare | 0 x3ff01168 | RW |  |  |
| Core2\_IPI\_Status | 0 x3ff01200 | RO | The IPI\_Status register of the number 2 processor core |  |
| Core2\_IPI\_Enalbe | 0 x3ff01204 | RW | The IPI\_Enalbe register of the number 2 processor core | 0 x0 |
| Core2\_IPI\_Set | 0 x3ff01208 | send | The IPI\_Set register of the number 2 processor core |  |
| Core2\_IPI\_Clear | 0 x3ff0120c | send | The IPI\_Clear register of the number 2 processor core |  |
| Core2\_MailBox0 | 0 x3ff01220 | RW | Register IPI\_MailBox0 of the number 2 processor core | 0 x0 |
| Core2\_MailBox1 | 0 x3ff01228 | RW | Register IPI\_MailBox1 in the number 2 processor core | 0 x0 |
| Core2\_MailBox2 | 0 x3ff01230 | RW | The IPI\_MailBox2 register of the number 2 processor core | 0 x0 |
| Core2\_MailBox3 | 0 x3ff01238 | RW | The IPI\_MailBox3 register of the number 2 processor core | 0 x0 |
| Core2\_int\_interval | 0 x3ff01260 | RW |  |  |
| Core2\_int\_compare | 0 x3ff01268 | RW |  |  |
| Core3\_IPI\_Status | 0 x3ff01300 | RO | The IPI\_Status register of the number 3 processor core |  |
| Core3\_IPI\_Enalbe | 0 x3ff01304 | RW | The IPI\_Enalbe register of the number 3 processor core | 0 x0 |
| Core3\_IPI\_Set | 0 x3ff01308 | send | The IPI\_Set register of the number 3 processor core |  |
| Core3\_IPI\_Clear | 0 x3ff0130c | send | The IPI\_Clear register of the number 3 processor core |  |
| Core3\_MailBox0 | 0 x3ff01320 | RW | Register IPI\_MailBox0 for the number 3 processor core | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Core3\_MailBox1 | 0 x3ff01328 | RW | Register IPI\_MailBox1 for the number 3 processor core | 0 x0 |
| Core3\_MailBox2 | 0 x3ff01330 | RW | The IPI\_MailBox2 register of the number 3 processor core | 0 x0 |
| Core3\_MailBox3 | 0 x3ff01338 | RW | The IPI\_MailBox3 register of the number 3 processor core | 0 x0 |
| Core3\_int\_interval | 0 x3ff01360 | RW |  |  |
| Core3\_int\_compare | 0 x3ff01368 | RW |  |  |
| Int Entry [0 -- 31] | 0 x3ff01400 | RW | 32 8-bit interrupt routing registers | 0 x0 |
| Intisr | 0 x3ff01420 | RO | 32-bit interrupt status register |  |
| Inten | 0 x3ff01424 | RO | 32-bit interrupt enabled status register |  |
| Intenset | 0 x3ff01428 | send | The 32-bit setting enables the register |  |
| Intenclr | 0 x3ff0142c | send | 32-bit clear enable registers and pulse-triggered interrupts |  |
| Intpol | 0 x3ff01430 | send | useless | 0 x0 |
| Intedge | 0 x3ff01434 | send | 32-bit trigger mode register (1: pulse trigger; 0: level triggered) | 0 x0 |
| CORE0\_INTISR | 0 x3ff01440 | RO | Routing to the 32-bit interrupt state of CORE0 |  |
| CORE1\_INTISR | 0 x3ff01448 | RO | Routing a 32-bit interrupt state to CORE1 |  |
| CORE2\_INTISR | 0 x3ff01450 | RO | Routing a 32-bit interrupt state to CORE2 |  |
| CORE3\_INTISR | 0 x3ff01458 | RO | Routing a 32-bit interrupt state to CORE3 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Thsens\_int\_ctrl\_Hi | 0 x3ff01460 | RW | Hi\_gate0: high temperature threshold 0, above which interruption will occur [8:8] : Hi\_en0: high temperature interrupt enable 0  [11:10] : Hi\_Sel0: select high temperature interrupt 0 as the temperature sensor input source [23:16] : Hi\_gate1: high temperature threshold 1 above which interruption will occur [24:24] : Hi\_en1: high temperature interrupt enable 1  [27:26] : Hi\_Sel1: select high temperature interrupt 1 for the temperature sensor input source [39:32] : Hi\_gate2: high temperature threshold 2, beyond which there will be an interrupt [40:40] : Hi\_en2: high temperature interrupt enable 2  [43:42] : Hi\_Sel2: select high temperature interrupt 2 as the temperature sensor input source [55:48] : Hi\_gate3: high temperature threshold 3, beyond which there will be an interrupt [56:56] : Hi\_en3: high temperature interrupt enable 3  [59:58] : Hi\_Sel3: select the temperature sensor input source of high temperature interrupt 3 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Thsens\_int\_ctrl\_Lo | 0 x3ff01468 | RW | Lo\_gate0: low temperature threshold 0, below which an interrupt will occur [8:8] : Lo\_en0: low temperature interrupt enable 0  [11:10] : Lo\_Sel0: select the temperature sensor input source of cryogenic interrupt 0 [23:16] : Lo\_gate1: cryogenic threshold 1 below which there will be an interrupt [24:24] : Lo\_en1: cryogenic interrupt enable 1  [27:26] : Lo\_Sel1: select the temperature sensor input source of cryogenic interrupt 1 [39:32] : Lo\_gate2: cryogenic threshold 2, below which there will be an interrupt [40:40] : Lo\_en2: cryogenic interrupt enable 2  [43:42] : Lo\_Sel2: select the temperature sensor input source for cold interrupt 2 [55:48] : Lo\_gate3: cold threshold 3, below which an interrupt [56:56] will occur: Lo\_en3: cold interrupt enable 3  [59:58] : Lo\_Sel3: select the temperature sensor input source of cryogenic interrupt 3 |  |
| Thsens\_int\_status/CLR | 0 x3ff01470 | RW | Interrupt status register, write any value to clear the interrupt  [0] : high temperature interrupt trigger [1] : low temperature interrupt trigger |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Thsens\_freq\_scale | 0 x3ff01480 | RW | Temperature sensor high temperature frequency down control register, four sets the priority from high to low [7:0] : Scale\_gate0: high temperature threshold 0, beyond which the frequency will be down [8:8] : Scale\_en0: high temperature frequency down enable 0  Scale\_Sel0: select the temperature sensor input source of high temperature frequency reduction 0 [14:12] : Scale\_freq0: frequency separation value when frequency reduction [23:16] : Scale\_gate1: high temperature threshold 1, beyond which the frequency will be reduced [24:24] : Scale\_en1: high temperature frequency reduction enable 1  Scale\_Sel1: select the temperature sensor input source of high temperature frequency reduction 1 [30:28] : Scale\_freq1: frequency separation value when frequency reduction [39:32] : Scale\_gate2: high temperature threshold 2, above which the frequency will be reduced [40:40] : Scale\_en2: high temperature frequency reduction enable 2  Scale\_Sel2: select the temperature sensor input source of high temperature frequency reduction 2 [46:44] : Scale\_freq2: frequency separation value when frequency reduction [55:48] : Scale\_gate3: high temperature threshold 3, above which frequency reduction [56:56] : Scale\_en3: high temperature frequency reduction enable 3  [59:58] : Scale\_Sel3: select the temperature sensor input source with high temperature down frequency 3  [62:60] : Scale\_freq3: frequency division value when frequency is down |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| DFD\_PARAM | 0 x3ff01500 | RW | Debug trigger condition enabled  [7:0] : timer, trigger delay, set to 1 to indicate immediate trigger when the condition is met, set to 0 to indicate no trigger, and set to other values to indicate the number of beats delayed triggered after the condition is met +1 [15:8] : trigger\_en, trigger condition enabled, corresponding to the enable control of the eight external trigger events |  |
| DFD\_TRIGGER | 0 x3ff01508 | send | A software trigger, which sends a write to this address, will cause a software trigger condition that causes a timer-1  Trigger after beat |  |
| CORE0\_AWCOND0 | 0 x3ff01800 | RW | CORE0's AXI interface AW triggers the condition 0 setting  [15:0] : awid [19:16] : awlen [22:20] : awburst [26:25] : awlock [30:27] : awcache [33:31] : awprot [37:34] : awcmd [41:38] : awdirqid [43:42] : awstate [47:44] : swscseti [48] : awvalid  [49] : awready |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CORE0\_AWMASK0 | 0 x3ff01808 | RW | CORE0's AXI interface AW trigger enable 0 is set, with the highest bit being AW channel trigger enable  [49:0] : awmask   1. : awdata\_en: trigger is allowed only when the wdata trigger condition of wid is met at the same time 2. : awchannel\_en: trigger condition enables the trigger condition to be   (AW\_IN & AWMASK) == (AWCOND & AWMASK) |  |
| CORE0\_AWCOND1 | 0 x3ff01810 | RW | The trigger condition for AW must be COND0 and COND1 at the same time  [47:0] : awaddr |  |
| CORE0\_AWMASK1 | 0 x3ff01818 | RW |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CORE0\_ARCOND0 | 0 x3ff01820 | RW | CORE0 has a AXI interface with AR trigger conditions similar to AW  [15:0] : arid [19:16] : Arlen [22:20] : arsize [24:23] : arburst [26:25] : arlock [30:27] : arcache [33:31] : arprot [37:34] : arcpuno [48] : arvalid  [49] : arready |  |
| CORE0\_ARMASK0 | 0 x3ff01828 | RW | CORE0's AXI interface AR trigger enable 0 is set, with the highest bit being the AR channel trigger enable  [49:0] : armask   1. : ardata\_en: only when the rdata triggering condition of rid is met 2. : archannel\_en: enable the trigger condition |  |
| CORE0\_ARCOND1 | 0 x3ff01830 | RW | [47:0] : araddr |  |
| CORE0\_ARMASK1 | 0 x3ff01838 | RW |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CORE0\_WCOND0 | 0 x3ff01840 | RW | CORE0 has a AXI interface W trigger condition similar to AW  [15:0] : wid [31:16] : WSTRB [32] : wlast [33] : wvalid  [34] : wready |  |
| CORE0\_WMASK0 | 0 x3ff01848 | RW | CORE0's AXI interface W trigger enable 0 is set, with the highest bit being the W channel trigger enable  [49:0] : wmask  [63] : wchannel\_en: trigger condition enabled, not set when awdata\_en is valid |  |
| CORE0\_WCOND1 | 0 x3ff01850 | RW |  |  |
| CORE0\_WMASK1 | 0 x3ff01858 | RW |  |  |
| CORE0\_WCOND2 | 0 x3ff01860 | RW |  |  |
| CORE0\_WMASK2 | 0 x3ff01868 | RW |  |  |
| CORE0\_BCOND0 | 0 x3ff01870 | RW | CORE0 has a AXI interface B trigger condition similar to AW  [15:0] : bid [17:16] : bresp [18] : bvalid  [19] : bready |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CORE0\_BMASK0 | 0 x3ff01878 | RW | CORE0's AXI interface B trigger enable 0 is set, with the highest bit being the B channel trigger enable  [19:0] : bmask [63] : bchannel\_en |  |
| CORE0\_RCOND0 | 0 x3ff01880 | RW | CORE0 has a AXI interface R trigger condition similar to AW  [15:0] : rid [17:16] : rresp [18] : rlast [19] : rrequest [21:20] : rstate [25:22] : rscseti [26] : rvalid  [27] : rready |  |
| CORE0\_RMASK0 | 0 x3ff01888 | RW | CORE0's AXI interface R trigger enable 0 is set, with the highest bit being the R channel trigger enable  [27:0] : rmask [63] : rchannel\_en |  |
| CORE0\_RCOND1 | 0 x3ff01890 | RW |  |  |
| CORE0\_RMASK1 | 0 x3ff01898 | RW |  |  |
| CORE0\_RCOND2 | 0 x3ff018a0 | RW |  |  |
| CORE0\_RMASK2 | 0 x3ff018a8 | RW |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TUD0\_CONF0 | 0 x3ff018e0 | RW | TUD0 configuration register 0  [47:0] : count\_target [55:48] : monitor\_enable |  |
| TUD0\_CONF1 | 0 x3ff018e8 | RW | TUD0 with register 1 [2:0] : DCDL\_sel\_signal [5:3] : DCDL\_sel\_clock [9:6] : signal\_sel [13:10] : clok\_sel [20:14] : reading\_sel [21] : counter\_clock\_sel [22] : sticky [23] : reset\_g [24] : stop   1. : start 2. : cg\_en |  |
| TUD0\_RESULT | 0 x3ff018f0 | R | TUD0 result register |  |
|  |  |  |  |  |
| CORE1\_AWCOND0 | 0 x3ff01900 | RW | CORE1's AXI interface AW triggers the condition 0 setting |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CORE1\_AWMASK0 | 0 x3ff01908 | RW | CORE1's AXI interface AW trigger enable 0 is set, with the highest bit being AW channel trigger enable  (AW\_IN & AWMASK) == (AWCOND & AWMASK) |  |
| CORE1\_AWCOND1 | 0 x3ff01910 | RW | The trigger condition for AW must be COND0 and COND1 at the same time |  |
| CORE1\_AWMASK1 | 0 x3ff01918 | RW |  |  |
| CORE1\_ARCOND0 | 0 x3ff01920 | RW | CORE1 has a AXI interface with AR trigger conditions similar to AW |  |
| CORE1\_ARMASK0 | 0 x3ff01928 | RW |  |  |
| CORE1\_ARCOND1 | 0 x3ff01930 | RW |  |  |
| CORE1\_ARMASK1 | 0 x3ff01938 | RW |  |  |
| CORE1\_WCOND0 | 0 x3ff01940 | RW | CORE1 has a AXI interface W trigger condition similar to AW |  |
| CORE1\_WMASK0 | 0 x3ff01948 | RW |  |  |
| CORE1\_WCOND1 | 0 x3ff01950 | RW |  |  |
| CORE1\_WMASK1 | 0 x3ff01958 | RW |  |  |
| CORE1\_WCOND2 | 0 x3ff01960 | RW |  |  |
| CORE1\_WMASK2 | 0 x3ff01968 | RW |  |  |
| CORE1\_BCOND0 | 0 x3ff01970 | RW | CORE1 has a AXI interface B trigger condition similar to AW |  |
| CORE1\_BMASK0 | 0 x3ff01978 | RW |  |  |
| CORE1\_RCOND0 | 0 x3ff01980 | RW | CORE1 has a AXI interface R trigger condition similar to AW |  |
| CORE1\_RMASK0 | 0 x3ff01988 | RW |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CORE1\_RCOND1 | 0 x3ff01990 | RW |  |  |
| CORE1\_RMASK1 | 0 x3ff01998 | RW |  |  |
| CORE1\_RCOND2 | 0 x3ff019a0 | RW |  |  |
| CORE1\_RMASK2 | 0 x3ff019a8 | RW |  |  |
|  |  |  |  |  |
| TUD1\_CONF0 | 0 x3ff019e0 | RW | TUD1 configuration register 0 [47:0] : count\_target  [55:48] : monitor\_enable |  |
| TUD1\_CONF1 | 0 x3ff019e8 | RW | TUD0 with register 1 [2:0] : DCDL\_sel\_signal [5:3] : DCDL\_sel\_clock [9:6] : signal\_sel [13:10] : clok\_sel [20:14] : reading\_sel [21] : counter\_clock\_sel [22] : sticky [23] : reset\_g [24] : stop   1. : start 2. : cg\_en |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TUD1\_RESULT | 0 x3ff019f0 | R | TUD1 result register |  |
|  |  |  |  |  |
| CORE2\_AWCOND0 | 0 x3ff01a00 | RW | CORE2's AXI interface AW triggers the condition 0 setting |  |
| CORE2\_AWMASK0 | 0 x3ff01a08 | RW | CORE2's AXI interface AW trigger enable 0 is set, with the highest bit being AW channel trigger enable  (AW\_IN & AWMASK) == (AWCOND & AWMASK) |  |
| CORE2\_AWCOND1 | 0 x3ff01a10 | RW | The trigger condition for AW must be COND0 and COND1 at the same time |  |
| CORE2\_AWMASK1 | 0 x3ff01a18 | RW |  |  |
| CORE2\_ARCOND0 | 0 x3ff01a20 | RW | CORE2 has a AXI interface with AR trigger conditions similar to AW |  |
| CORE2\_ARMASK0 | 0 x3ff01a28 | RW |  |  |
| CORE2\_ARCOND1 | 0 x3ff01a30 | RW |  |  |
| CORE2\_ARMASK1 | 0 x3ff01a38 | RW |  |  |
| CORE2\_WCOND0 | 0 x3ff01a40 | RW | CORE2 has a AXI interface W trigger condition similar to AW |  |
| CORE2\_WMASK0 | 0 x3ff01a48 | RW |  |  |
| CORE2\_WCOND1 | 0 x3ff01a50 | RW |  |  |
| CORE2\_WMASK1 | 0 x3ff01a58 | RW |  |  |
| CORE2\_WCOND2 | 0 x3ff01a60 | RW |  |  |
| CORE2\_WMASK2 | 0 x3ff01a68 | RW |  |  |
| CORE2\_BCOND0 | 0 x3ff01a70 | RW | CORE2 has a AXI interface B trigger condition similar to AW |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CORE2\_BMASK0 | 0 x3ff01a78 | RW |  |  |
| CORE2\_RCOND0 | 0 x3ff01a80 | RW | CORE2 has a AXI interface R trigger condition similar to AW |  |
| CORE2\_RMASK0 | 0 x3ff01a88 | RW |  |  |
| CORE2\_RCOND1 | 0 x3ff01a90 | RW |  |  |
| CORE2\_RMASK1 | 0 x3ff01a98 | RW |  |  |
| CORE2\_RCOND2 | 0 x3ff01aa0 | RW |  |  |
| CORE2\_RMASK2 | 0 x3ff01aa8 | RW |  |  |
|  |  |  |  |  |
| TUD2\_CONF0 | 0 x3ff01ae0 | RW | TUD2 configuration register 0 [47:0] : count\_target  [55:48] : monitor\_enable |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TUD2\_CONF1 | 0 x3ff01ae8 | RW | TUD0 with register 1 [2:0] : DCDL\_sel\_signal [5:3] : DCDL\_sel\_clock [9:6] : signal\_sel [13:10] : clok\_sel [20:14] : reading\_sel [21] : counter\_clock\_sel [22] : sticky [23] : reset\_g [24] : stop   1. : start 2. : cg\_en |  |
| TUD2\_RESULT | 0 x3ff01af0 | R | TUD2 result register |  |
|  |  |  |  |  |
| CORE3\_AWCOND0 | 0 x3ff01b00 | RW | CORE3's AXI interface AW triggers the condition 0 setting |  |
| CORE3\_AWMASK0 | 0 x3ff01b08 | RW | CORE3's AXI interface AW trigger enable 0 is set, with the highest bit being AW channel trigger enable  (AW\_IN & AWMASK) == (AWCOND & AWMASK) |  |
| CORE3\_AWCOND1 | 0 x3ff01b10 | RW | The trigger condition for AW must be COND0 and COND1 at the same time |  |
| CORE3\_AWMASK1 | 0 x3ff01b18 | RW |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CORE3\_ARCOND0 | 0 x3ff01b20 | RW | CORE3 has a AXI interface with AR trigger conditions similar to AW |  |
| CORE3\_ARMASK0 | 0 x3ff01b28 | RW |  |  |
| CORE3\_ARCOND1 | 0 x3ff01b30 | RW |  |  |
| CORE3\_ARMASK1 | 0 x3ff01b38 | RW |  |  |
| CORE3\_WCOND0 | 0 x3ff01b40 | RW | CORE3 has a AXI interface W trigger condition similar to AW |  |
| CORE3\_WMASK0 | 0 x3ff01b48 | RW |  |  |
| CORE3\_WCOND1 | 0 x3ff01b50 | RW |  |  |
| CORE3\_WMASK1 | 0 x3ff01b58 | RW |  |  |
| CORE3\_WCOND2 | 0 x3ff01b60 | RW |  |  |
| CORE3\_WMASK2 | 0 x3ff01b68 | RW |  |  |
| CORE3\_BCOND0 | 0 x3ff01b70 | RW | CORE3 has a AXI interface B trigger condition similar to AW |  |
| CORE3\_BMASK0 | 0 x3ff01b78 | RW |  |  |
| CORE3\_RCOND0 | 0 x3ff01b80 | RW | CORE3 has a AXI interface R trigger condition similar to AW |  |
| CORE3\_RMASK0 | 0 x3ff01b88 | RW |  |  |
| CORE3\_RCOND1 | 0 x3ff01b90 | RW |  |  |
| CORE3\_RMASK1 | 0 x3ff01b98 | RW |  |  |
| CORE3\_RCOND2 | 0 x3ff01ba0 | RW |  |  |
| CORE3\_RMASK2 | 0 x3ff01ba8 | RW |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TUD3\_CONF0 | 0 x3ff01be0 | RW | TUD3 configuration register 0  [47:0] : count\_target [55:48] : monitor\_enable |  |
| TUD3\_CONF1 | 0 x3ff01be8 | RW | TUD0 with register 1 [2:0] : DCDL\_sel\_signal [5:3] : DCDL\_sel\_clock [9:6] : signal\_sel [13:10] : clok\_sel [20:14] : reading\_sel [21] : counter\_clock\_sel [22] : sticky [23] : reset\_g [24] : stop   1. : start 2. : cg\_en |  |
| TUD3\_RESULT | 0 x3ff01bf0 | R | TUD3 result register |  |
|  |  |  |  |  |
| TUD4\_CONF0 | 0 x3ff01ce0 | RW | TUD4 configuration register 0 [47:0] : count\_target  [55:48] : monitor\_enable |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TUD4\_CONF1 | 0 x3ff01ce8 | RW | TUD4 with a register 1 [2:0] : DCDL\_sel\_signal [5:3] : DCDL\_sel\_clock [8:6] : signal\_sel [11:12] : clock\_sel [18:12] : reading\_sel [19] : counter\_clock\_sel [20] : sticky [21] : reset\_g [22] : stop   1. : start 2. : cg\_en |  |
| TUD4\_RESULT | 0 x3ff01cf0 | R | TUD4 result register |  |
|  |  |  |  |  |
| TUD5\_CONF0 | 0 x3ff01de0 | RW | TUD5 configuration register 0 [47:0] : count\_target  [55:48] : monitor\_enable |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TUD5\_CONF1 | 0 x3ff01de8 | RW | TUD5 with depositor 1 [2:0] : DCDL\_sel\_signal [5:3] : DCDL\_sel\_clock [8:6] : signal\_sel [11:12] : clock\_sel [18:12] : reading\_sel [19] : counter\_clock\_sel [20] : sticky [21] : reset\_g [22] : stop   1. : start 2. : cg\_en |  |
| TUD5\_RESULT | 0 x3ff01df0 | R | TUD5 result register |  |
|  |  |  |  |  |
| HT0\_AWCOND0 | 0 x3ff01e00 | RW | The AXI interface AW triggers the condition 0 setting for HT0 |  |
| HT0\_AWMASK0 | 0 x3ff01e08 | RW | HT0's AXI interface AW trigger enable 0 is set to the highest bit of AW channel trigger enable condition  (AW\_IN & AWMASK) == (AWCOND & AWMASK) |  |
| HT0\_AWCOND1 | 0 x3ff01e10 | RW | The trigger condition for AW must be COND0 and COND1 at the same time |  |
| HT0\_AWMASK1 | 0 x3ff01e18 | RW |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| HT0\_ARCOND0 | 0 x3ff01e20 | RW | HT0 has a AXI interface with AR trigger conditions similar to AW |  |
| HT0\_ARMASK0 | 0 x3ff01e28 | RW |  |  |
| HT0\_ARCOND1 | 0 x3ff01e30 | RW |  |  |
| HT0\_ARMASK1 | 0 x3ff01e38 | RW |  |  |
| HT0\_WCOND0 | 0 x3ff01e40 | RW | The AXI interface of HT0 has AW trigger condition similar to AW |  |
| HT0\_WMASK0 | 0 x3ff01e48 | RW |  |  |
| HT0\_WCOND1 | 0 x3ff01e50 | RW |  |  |
| HT0\_WMASK1 | 0 x3ff01e58 | RW |  |  |
| HT0\_WCOND2 | 0 x3ff01e60 | RW |  |  |
| HT0\_WMASK2 | 0 x3ff01e68 | RW |  |  |
| HT0\_BCOND0 | 0 x3ff01e70 | RW | HT0 has a AXI interface B trigger condition similar to AW |  |
| HT0\_BMASK0 | 0 x3ff01e78 | RW |  |  |
| HT0\_RCOND0 | 0 x3ff01e80 | RW | HT0 has a AXI interface R trigger condition similar to AW |  |
| HT0\_RMASK0 | 0 x3ff01e88 | RW |  |  |
| HT0\_RCOND1 | 0 x3ff01e90 | RW |  |  |
| HT0\_RMASK1 | 0 x3ff01e98 | RW |  |  |
| HT0\_RCOND2 | 0 x3ff01ea0 | RW |  |  |
| HT0\_RMASK2 | 0 x3ff01ea8 | RW |  |  |
| HT1\_AWCOND0 | 0 x3ff01f00 | RW | HT1's AXI interface AW trigger condition is set to 0 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| HT1\_AWMASK0 | 0 x3ff01f08 | RW | HT1's AXI interface AW trigger enable is set to 0, with the highest bit being AW channel trigger enable  (AW\_IN & AWMASK) == (AWCOND & AWMASK) |  |
| HT1\_AWCOND1 | 0 x3ff01f10 | RW | The trigger condition for AW must be COND0 and COND1 at the same time |  |
| HT1\_AWMASK1 | 0 x3ff01f18 | RW |  |  |
| HT1\_ARCOND0 | 0 x3ff01f20 | RW | HT1 has a AXI interface with AR trigger conditions similar to AW |  |
| HT1\_ARMASK0 | 0 x3ff01f28 | RW |  |  |
| HT1\_ARCOND1 | 0 x3ff01f30 | RW |  |  |
| HT1\_ARMASK1 | 0 x3ff01f38 | RW |  |  |
| HT1\_WCOND0 | 0 x3ff01f40 | RW | The AXI interface of HT1 has AW trigger condition similar to AW |  |
| HT1\_WMASK0 | 0 x3ff01f48 | RW |  |  |
| HT1\_WCOND1 | 0 x3ff01f50 | RW |  |  |
| HT1\_WMASK1 | 0 x3ff01f58 | RW |  |  |
| HT1\_WCOND2 | 0 x3ff01f60 | RW |  |  |
| HT1\_WMASK2 | 0 x3ff01f68 | RW |  |  |
| HT1\_BCOND0 | 0 x3ff01f70 | RW | HT1 has a AXI interface B trigger condition similar to AW |  |
| HT1\_BMASK0 | 0 x3ff01f78 | RW |  |  |
| HT1\_RCOND0 | 0 x3ff01f80 | RW | HT1 has a AXI interface R trigger condition similar to AW |  |
| HT1\_RMASK0 | 0 x3ff01f88 | RW |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| HT1\_RCOND1 | 0 x3ff01f90 | RW |  |  |
| HT1\_RMASK1 | 0 x3ff01f98 | RW |  |  |
| HT1\_RCOND2 | 0 x3ff01fa0 | RW |  |  |
| HT1\_RMASK2 | 0 x3ff01fa8 | RW |  |  |
| CORE0\_WIN0\_BASE | 0 x3ff02000 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN1\_BASE | 0 x3ff02008 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN2\_BASE | 0 x3ff02010 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN3\_BASE | 0 x3ff02018 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN4\_BASE | 0 x3ff02020 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN5\_BASE | 0 x3ff02028 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN6\_BASE | 0 x3ff02030 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN7\_BASE | 0 x3ff02038 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN0\_MASK | 0 x3ff02040 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN1\_MASK | 0 x3ff02048 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN2\_MASK | 0 x3ff02050 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN3\_MASK | 0 x3ff02058 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN4\_MASK | 0 x3ff02060 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN5\_MASK | 0 x3ff02068 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN6\_MASK | 0 x3ff02070 | RW | Level 1 cross switch address window | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CORE0\_WIN7\_MASK | 0 x3ff02078 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN0\_MMAP | 0 x3ff02080 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN1\_MMAP | 0 x3ff02088 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN2\_MMAP | 0 x3ff02090 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN3\_MMAP | 0 x3ff02098 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN4\_MMAP | 0 x3ff020a0 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN5\_MMAP | 0 x3ff020a8 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN6\_MMAP | 0 x3ff020b0 | RW | Level 1 cross switch address window | 0 x0 |
| CORE0\_WIN7\_MMAP | 0 x3ff020b8 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN0\_BASE | 0 x3ff02100 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN1\_BASE | 0 x3ff02108 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN2\_BASE | 0 x3ff02110 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN3\_BASE | 0 x3ff02118 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN4\_BASE | 0 x3ff02120 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN5\_BASE | 0 x3ff02128 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN6\_BASE | 0 x3ff02130 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN7\_BASE | 0 x3ff02138 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN0\_MASK | 0 x3ff02140 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN1\_MASK | 0 x3ff02148 | RW | Level 1 cross switch address window | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CORE1\_WIN2\_MASK | 0 x3ff02150 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN3\_MASK | 0 x3ff02158 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN4\_MASK | 0 x3ff02160 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN5\_MASK | 0 x3ff02168 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN6\_MASK | 0 x3ff02170 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN7\_MASK | 0 x3ff02178 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN0\_MMAP | 0 x3ff02180 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN1\_MMAP | 0 x3ff02188 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN2\_MMAP | 0 x3ff02190 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN3\_MMAP | 0 x3ff02198 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN4\_MMAP | 0 x3ff021a0 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN5\_MMAP | 0 x3ff021a8 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN6\_MMAP | 0 x3ff021b0 | RW | Level 1 cross switch address window | 0 x0 |
| CORE1\_WIN7\_MMAP | 0 x3ff021b8 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN0\_BASE | 0 x3ff02200 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN1\_BASE | 0 x3ff02208 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN2\_BASE | 0 x3ff02210 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN3\_BASE | 0 x3ff02218 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN4\_BASE | 0 x3ff02220 | RW | Level 1 cross switch address window | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CORE2\_WIN5\_BASE | 0 x3ff02228 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN6\_BASE | 0 x3ff02230 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN7\_BASE | 0 x3ff02238 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN0\_MASK | 0 x3ff02240 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN1\_MASK | 0 x3ff02248 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN2\_MASK | 0 x3ff02250 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN3\_MASK | 0 x3ff02258 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN4\_MASK | 0 x3ff02260 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN5\_MASK | 0 x3ff02268 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN6\_MASK | 0 x3ff02270 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN7\_MASK | 0 x3ff02278 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN0\_MMAP | 0 x3ff02280 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN1\_MMAP | 0 x3ff02288 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN2\_MMAP | 0 x3ff02290 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN3\_MMAP | 0 x3ff02298 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN4\_MMAP | 0 x3ff022a0 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN5\_MMAP | 0 x3ff022a8 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN6\_MMAP | 0 x3ff022b0 | RW | Level 1 cross switch address window | 0 x0 |
| CORE2\_WIN7\_MMAP | 0 x3ff022b8 | RW | Level 1 cross switch address window | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CORE3\_WIN0\_BASE | 0 x3ff02300 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN1\_BASE | 0 x3ff02308 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN2\_BASE | 0 x3ff02310 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN3\_BASE | 0 x3ff02318 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN4\_BASE | 0 x3ff02320 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN5\_BASE | 0 x3ff02328 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN6\_BASE | 0 x3ff02330 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN7\_BASE | 0 x3ff02338 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN0\_MASK | 0 x3ff02340 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN1\_MASK | 0 x3ff02348 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN2\_MASK | 0 x3ff02350 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN3\_MASK | 0 x3ff02358 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN4\_MASK | 0 x3ff02360 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN5\_MASK | 0 x3ff02368 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN6\_MASK | 0 x3ff02370 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN7\_MASK | 0 x3ff02378 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN0\_MMAP | 0 x3ff02380 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN1\_MMAP | 0 x3ff02388 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN2\_MMAP | 0 x3ff02390 | RW | Level 1 cross switch address window | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CORE3\_WIN3\_MMAP | 0 x3ff02398 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN4\_MMAP | 0 x3ff023a0 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN5\_MMAP | 0 x3ff023a8 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN6\_MMAP | 0 x3ff023b0 | RW | Level 1 cross switch address window | 0 x0 |
| CORE3\_WIN7\_MMAP | 0 x3ff023b8 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN0\_BASE | 0 x3ff02400 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN1\_BASE | 0 x3ff02408 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN2\_BASE | 0 x3ff02410 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN3\_BASE | 0 x3ff02418 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN4\_BASE | 0 x3ff02420 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN5\_BASE | 0 x3ff02428 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN6\_BASE | 0 x3ff02430 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN7\_BASE | 0 x3ff02438 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN0\_MASK | 0 x3ff02440 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN1\_MASK | 0 x3ff02448 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN2\_MASK | 0 x3ff02450 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN3\_MASK | 0 x3ff02458 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN4\_MASK | 0 x3ff02460 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN5\_MASK | 0 x3ff02468 | RW | Level 1 cross switch address window | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| EAST\_WIN6\_MASK | 0 x3ff02470 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN7\_MASK | 0 x3ff02478 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN0\_MMAP | 0 x3ff02480 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN1\_MMAP | 0 x3ff02488 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN2\_MMAP | 0 x3ff02490 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN3\_MMAP | 0 x3ff02498 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN4\_MMAP | 0 x3ff024a0 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN5\_MMAP | 0 x3ff024a8 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN6\_MMAP | 0 x3ff024b0 | RW | Level 1 cross switch address window | 0 x0 |
| EAST\_WIN7\_MMAP | 0 x3ff024b8 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN0\_BASE | 0 x3ff02500 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN1\_BASE | 0 x3ff02508 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN2\_BASE | 0 x3ff02510 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN3\_BASE | 0 x3ff02518 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN4\_BASE | 0 x3ff02520 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN5\_BASE | 0 x3ff02528 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN6\_BASE | 0 x3ff02530 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN7\_BASE | 0 x3ff02538 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN0\_MASK | 0 x3ff02540 | RW | Level 1 cross switch address window | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SOUTH\_WIN1\_MASK | 0 x3ff02548 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN2\_MASK | 0 x3ff02550 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN3\_MASK | 0 x3ff02558 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN4\_MASK | 0 x3ff02560 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN5\_MASK | 0 x3ff02568 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN6\_MASK | 0 x3ff02570 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN7\_MASK | 0 x3ff02578 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN0\_MMAP | 0 x3ff02580 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN1\_MMAP | 0 x3ff02588 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN2\_MMAP | 0 x3ff02590 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN3\_MMAP | 0 x3ff02598 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN4\_MMAP | 0 x3ff025a0 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN5\_MMAP | 0 x3ff025a8 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN6\_MMAP | 0 x3ff025b0 | RW | Level 1 cross switch address window | 0 x0 |
| SOUTH\_WIN7\_MMAP | 0 x3ff025b8 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN0\_BASE | 0 x3ff02600 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN1\_BASE | 0 x3ff02608 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN2\_BASE | 0 x3ff02610 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN3\_BASE | 0 x3ff02618 | RW | Level 1 cross switch address window | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| WEST\_WIN4\_BASE | 0 x3ff02620 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN5\_BASE | 0 x3ff02628 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN6\_BASE | 0 x3ff02630 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN7\_BASE | 0 x3ff02638 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN0\_MASK | 0 x3ff02640 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN1\_MASK | 0 x3ff02648 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN2\_MASK | 0 x3ff02650 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN3\_MASK | 0 x3ff02658 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN4\_MASK | 0 x3ff02660 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN5\_MASK | 0 x3ff02668 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN6\_MASK | 0 x3ff02670 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN7\_MASK | 0 x3ff02678 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN0\_MMAP | 0 x3ff02680 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN1\_MMAP | 0 x3ff02688 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN2\_MMAP | 0 x3ff02690 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN3\_MMAP | 0 x3ff02698 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN4\_MMAP | 0 x3ff026a0 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN5\_MMAP | 0 x3ff026a8 | RW | Level 1 cross switch address window | 0 x0 |
| WEST\_WIN6\_MMAP | 0 x3ff026b0 | RW | Level 1 cross switch address window | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| WEST\_WIN7\_MMAP | 0 x3ff026b8 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN0\_BASE | 0 x3ff02700 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN1\_BASE | 0 x3ff02708 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN2\_BASE | 0 x3ff02710 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN3\_BASE | 0 x3ff02718 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN4\_BASE | 0 x3ff02720 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN5\_BASE | 0 x3ff02728 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN6\_BASE | 0 x3ff02730 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN7\_BASE | 0 x3ff02738 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN0\_MASK | 0 x3ff02740 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN1\_MASK | 0 x3ff02748 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN2\_MASK | 0 x3ff02750 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN3\_MASK | 0 x3ff02758 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN4\_MASK | 0 x3ff02760 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN5\_MASK | 0 x3ff02768 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN6\_MASK | 0 x3ff02770 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN7\_MASK | 0 x3ff02778 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN0\_MMAP | 0 x3ff02780 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN1\_MMAP | 0 x3ff02788 | RW | Level 1 cross switch address window | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| NORTH\_WIN2\_MMAP | 0 x3ff02790 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN3\_MMAP | 0 x3ff02798 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN4\_MMAP | 0 x3ff027a0 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN5\_MMAP | 0 x3ff027a8 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN6\_MMAP | 0 x3ff027b0 | RW | Level 1 cross switch address window | 0 x0 |
| NORTH\_WIN7\_MMAP | 0 x3ff027b8 | RW | Level 1 cross switch address window | 0 x0 |

## Hardware and software design guide

Loongson 3A2000 processor pin downward compatible with loongson 3A1000 processor, but the corresponding software and hardware need to make some configuration changes to enable the original compatibility mode, or to open some new features of loongson 3A2000. This chapter focuses on the difference between the software and hardware Settings of loongson 3A2000 processor and loongson 3A1000 processor.

#### Hardware change guide

1. The original CORE\_PLL\_AVDD, DDR\_PLL\_AVDD, ht0/1\_pll\_avdd are now NC pin. If the original 3A motherboard is used, no modification can be made. However, if compatibility with future 3A3000 is considered, these supply voltages can be modified to 1.8v, or a 1.8v/2.5v configurable design can be adopted.
2. The original Mc0/1\_comp\_ref\_res was changed to NC pin. If the original 3A motherboard is used, no modification can be made;
3. The original ht0/1\_pll\_ref was changed to NC pin. If the original 3A motherboard is used, no modification can be made;
4. The original Mc0/1\_comp\_ref\_gnd was changed to Mc0/1\_a15. If the original 3A motherboard is used, no modification can be made; But if connected to a memory stick, you can support a larger amount of memory;
5. The function of PCI\_CONFIG[0] control is changed to SPI startup enable, which can be started from SPI FLASH after setting to 1.If you use the original 3A motherboard, you need to set it to 0 and start from LPC FLASH. If the mainboard already has SPI FLASH, you can connect GPIO[0] as SPI\_CS, and set PCI\_CONFIG[0] to 1 to start from SPI FLASH.
6. The control function of PCI\_CONFIG[7] was changed to force HT1.0 mode, and HT was directly started in 1.0 mode after it was set to 1. If you use the 3A780E motherboard, you need to set it to 1 at present; If using 3A2H motherboard, no special Settings are required;
7. CLKSEL[15:10] needs to be set to 6 'b100001; If you need to use HT3.0 mode, you need to use CLKSEL[15:10]
8. Set it to 6 'b100101;
9. CLKSEL[9:5] needs to be set to 5 'b01111; Use PMON to set the memory frequency;
10. CLKSEL[4:0] needs to be set to 5 'b01111; PMON is used to set the processor core frequency.
11. For the 3A2H motherboard, the pull up resistance on ht0/1\_powerok and ht0/1\_resetn should be removed.(the original pull-up
12. The resistance of 300 ohms is not suitable for 3A, it can also be removed.)

#### Frequency setting instructions

1. In order to be basically compatible with the frequency configuration of loongson 3A1000, the hardware frequency configuration range of loongson 3A2000 is relatively narrow. In order to obtain a wider frequency range and better clock quality, the software configuration method in PMON is mainly used in loongson 3A2000. The configuration method is the same as that of loongson 3B1500. Refer to the PMON source code for the specific configuration method.
2. The frequency setting is completely set by the software, and the CLKSEL need not be modified when the frequency is changed.
3. 1.15v stable operating frequency of core voltage: the processor core frequency is set to 800MHz, the memory frequency is set to 500MHz, the HT controller is set to 400MHz, and the HT bus is 800MHz/1600MHz.

#### PMON change guide

1. Compared with the loong chip 3A1000, it has been upgraded to different degrees from processor core, memory controller, HT controller to cross switches at all levels, so PMON needs to make some changes, mainly including the following parts:
2. Initialization of L1 Dcache, L1 Icache, Vcache and L2 Cache after power is removed (hardware completed);
3. After the CPU is energized, close all the core Store Fill Buffer;
4. After the CPU is powered on, turn off all the core write merge functions;
5. If you need to maintain compatibility with 3A5, set the PRID hidden bit in CP0 Diag register for all cores.
6. Change all statements in assembly code that are jr rx and rx is not register 31 to jr $31;
7. Use the code to configure the processor core, memory and node PLL similar to 3B1500.
8. Use the memory controller configuration and parameter training code similar to 3B1500;
9. If HT works in mode 1.0, HT can only work in 8-bit mode.
10. If the SPI controller is used, the base address is changed from 0xBFE001F0 to 0xBFE00220;
11. In addition to these required changes, you can make the following changes to enhance PMON functionality:
12. Modify the delay of the buzzer to ensure that the user can hear the buzzer;
13. Added support for turning off defective nuclear clocks;

#### Kernel change guide

1. Changes that need to be made in the kernel include:
2. Modify the Cache description structure in the kernel, VCache and SCache are connected by 16-way groups;
3. Modify the calculation method of the temperature sensor to read -100 as the same as 3B1500. At present, the sample has not been tested and calibrated, and there may be a large deviation between the readout value of some chips and the actual temperature. Therefore, it is recommended not to use the temperature indicator of the temperature sensor inside theprocessor in the current kernel.
4. Modify the configuration register address when closing the core;
5. Change the operation of brush ICache/DCache to brush ICache/DCache/VCache;
6. If the SPI controller is used, the base address is changed from 0xBFE001F0 to 0xBFE00220;
7. Uncache DMA must be used to maintain the consistency of Cache data.
8. Add store fill buffer support: first, SYNC needs to be added before all Uncache requests to ensure
9. When the Uncache request occurs, all contents in the store fill buffer are written back to the Cache. Second, it is necessary to use LL/SC instruction to unlock all synchronous operations Shared among different cores.
10. Do not use the device's MSI functionality. When the MSI function must be used, the number of data receiving buffers of the HT controller's POST channel should be set to 1, and the HT bus should be reconnected.
11. Lock Cache operations cannot be used for DMA regions where hardware automatically maintains consistency.
12. Other modifications that can be used to improve performance are:
13. Increased support for FTLB;
14. Added support for TLB rapid refill;
15. Add wait command support;
16. Added prefetch instruction support;
17. Interrupt return is implemented using DI/EI. However, it should be noted that the [31:4] returned by EI instruction is a random value, which is different from the MIPS regulation.

#### Other change description

1. Performance counter overflow interrupts cannot achieve precise interrupts, resulting in limited use of current perf tools. If necessary, frequent mfc0 perfcnt instructions (user-mode available) must be added. For example, if the instruction is inserted into the processing function of high-frequency clock interrupt, it will still cause the interrupt to fail to be generated in time, resulting in a large error in event statistics.