



Loongson 3A3000/3B3000 processor user manual

Part i

# Multi - core processor architecture, register description and System Software programming guide V1.3

***In April 2017***

***Loongson Zhongke Technology Co. LTD***



**Copyright statement**

The copyright of this document belongs to Loongson Technology Co., LTD., and all rights are reserved. No company or individual may publish, reprint or otherwise distribute any part of this document to third parties without written permission. Otherwise, it will be held legally liable.

**disclaimer**

This document only provides phased information. The contents can be updated according to the actual situation of the product at any time without prior notice. The Company shall not be liable for any direct or indirect losses caused by improper use of documents.

**Loongson Zhongke Technology Co. LTD**

Loongson Technology Corporation Limited

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

Building No.2, Loongson Industrial Park,

Zhongguancun Environmental Protection Park, Haidian District, Beijing

Tel: 010-62546668 Fax: 010-62600826

### Reading guide

The User manual of Loongson 3A3000/3B3000 processor is divided into volumes 1 and 2.

The first volume of The User's Manual of Longson 3A300/3B3000 Processor is divided into two parts. The first part introduces the architecture and register description of longson 3A300/3B3000 multi-core processor, and gives a detailed description of the chip system architecture, functions and configuration of main modules, register list and bit field.

The second volume of User's Manual of Loongson 3A300/3B3000 processor introduces GS464e high-performance processor core used by Loongson 3A300/3B3000 in detail from the perspective of system software developers.

# Revision history

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Document update record** | | | The document name: | | Loongson 3A3000/3B3000 processor user manual  - the days |
| The version number | | V1.3 |
| The founders: | | Chip R&d Department |
| Date created: | | 2017-04-13 |
| **Update history** | | | | | |
| **The serial number** | **Updated date** | **The version number** | | **Update the content** | |
| 1 | 2016-06-14 | V1.0 | | The initial release | |
| 2 | 2016-09-14 | V1.1 | | Update part of PLL configuration register description, update software and hardware change description | |
| 3 | 2016-11-25 | V1.2 | | Add register description for BBGEN section | |
| 4 | 2017-04-13 | V1.3 | | Add SPI address space description | |

Feedback of manual information: 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.

# directory

目录

[Multi - core processor architecture, register description and System Software programming guide V1.3 1](#_Toc44428390)

[Reading guide 3](#_Toc44428391)

[Revision history 4](#_Toc44428392)

[directory 1](#_Toc44428393)

[Figure orders to record 4](#_Toc44428394)

[Table item record 6](#_Toc44428395)

[1 An overview of the 15](#_Toc44428396)

[2 System configuration and control 18](#_Toc44428397)

[3 GS464e processor core 41](#_Toc44428398)

[4 Shared Cache (SCache) 43](#_Toc44428399)

[5 Matrix processing accelerator 46](#_Toc44428400)

[6 Interrupt communication between processor cores 50](#_Toc44428401)

[7 I/O interrupt 51](#_Toc44428402)

[8 Temperature sensor 55](#_Toc44428403)

[9 Ddr2/3 SDRAM controller configuration 59](#_Toc44428404)

[9.3 Ddr2/3 SDRAM write operation protocol 61](#_Toc44428405)

[10 HyperTransport controller 60](#_Toc44428406)

[10.5.1 Bridge Control 72](#_Toc44428407)

[10.5.2 Capability Registers 72](#_Toc44428408)

[10.5.3 Custom register 75](#_Toc44428409)

[10.5.4 Receiving diagnostic register 78](#_Toc44428410)

[10.5.5 Interrupt routing mode selects registers 78](#_Toc44428411)

[10.5.6 Receive buffer initial register 78](#_Toc44428412)

[10.5.7 The receive address window configures registers 79](#_Toc44428413)

[10.5.8 Interrupt vector register 82](#_Toc44428414)

[10.5.9 Interrupt enable register 85](#_Toc44428415)

[10.5.10 Interrupt Discovery & Configuration 89](#_Toc44428416)

[10.5.11 The POST address window configures registers 90](#_Toc44428417)

[10.5.12 The prefetch address window configures registers 91](#_Toc44428418)

[10.5.13 The UNCACHE Address window configures registers 93](#_Toc44428419)

[10.5.14 The P2P address window configures registers 95](#_Toc44428420)

[10.5.15 The command sends the cache size register 97](#_Toc44428421)

[10.5.16 Data sent cache size register 97](#_Toc44428422)

[10.5.17 Send the cache debug register 98](#_Toc44428423)

[10.5.18 PHY impedance matching control register 99](#_Toc44428424)

[10.5.19 Revision ID register 99](#_Toc44428425)

[10.5.20 Error Retry controls the register 99](#_Toc44428426)

[10.5.21 The Retry Count register 100](#_Toc44428427)

[10.5.22 Link Train register 100](#_Toc44428428)

[10.5.23 Training 0 short timeout register 101](#_Toc44428429)

[10.5.24 Training 0 timeout long timing register 103](#_Toc44428430)

[10.5.25 Training 1 counting register 103](#_Toc44428431)

[10.5.26 Training 2 counting register 103](#_Toc44428432)

[10.5.27 Training 3 counting register 103](#_Toc44428433)

[10.5.28 Software frequency configuration registers 104](#_Toc44428434)

[10.5.29 PHY configuration register 105](#_Toc44428435)

[10.5.30 Link initializes the debug register 106](#_Toc44428436)

[10.5.31 LDT debug register 106](#_Toc44428437)

[11 Low speed IO controller configuration 112](#_Toc44428438)

[11.3.2 Interrupt enablement register (IER) 121](#_Toc44428439)

[11.3.3 Interrupt Identification Register (IIR) 121](#_Toc44428440)

[11.3.4 FIFO control Register (FCR) 122](#_Toc44428441)

[11.3.5 Line Control Register (LCR) 123](#_Toc44428442)

[11.3.6 MODEM Control Register (MCR) 125](#_Toc44428443)

[11.3.7 Line Status Register (LSR) 125](#_Toc44428444)

[11.3.8 MODEM Status Register (MSR) 127](#_Toc44428445)

[11.3.9 Frequency division latch 127](#_Toc44428446)

[11.4.1 Control register (SPCR) 128](#_Toc44428447)

[11.4.2 Status Register (SPSR) 129](#_Toc44428448)

[11.4.3 Data Register (TxFIFO) 129](#_Toc44428449)

[11.4.4 External register (SPER) 129](#_Toc44428450)

[12 Chip configuration register list 138](#_Toc44428451)

[13 Software and Hardware Design Guide 172](#_Toc44428452)

## Figure orders to record

[Figure 1-1 System structure 11 of Loongson 3](#_bookmark2)

[FIG. 1-2 Structure of Nodal 3 of the Loong Chip 12](#_bookmark4)

[Figure 1-3 Structure of Longson 3A3000/3B3000 chip 13](#_bookmark5)

[Figure 3-1 GS464e Structure Figure 31](#_bookmark26)

[FIG. 7-1 Schematic diagram of interrupt routing for longshon 3A3000/3B3000 processor](#_bookmark39)

[Figure 9-1 DDR2 SDRAM read operation protocol 47](#_bookmark54)

[Figure 9-2 DDR2 SDRAM write protocol 47](#_bookmark54)

[Figure 10-1 Configuration access of HT protocol in Longson 3A300/3B3000](#_bookmark142)

[FIG. 10-2 Four-chip Longshon No.3 Interconnection structure 99](#_bookmark144)

[Figure 10-3 Interconnection structure of two pieces of Loong chip no.3 and 8-bit 100](#_bookmark145)

[FIG. 10-4 16-bit interconnection structure of two pieces of Loong chip No.3](#_bookmark146)

[Figure 11-1. Configure the read-write bus address generation 105](#_bookmark151)

## Table item record

[Table 2-1 Control pin description 15](#_bookmark8)

[Table 2-2 Global address distribution of the system at node level 17](#_bookmark10)

[Table 2-3 Address distribution in nodes 18](#_bookmark11)

[Table 2-4 Address distribution in nodes 18](#_bookmark11)

[Table 2-5 MMAP field corresponding to the space access property 19](#_bookmark13)

[Table 2-6 Table 19 of address window register of first level cross switch](#_bookmark14)

[Table 2-7 At level 2 XBAR, the corresponding relationship between the device number and the module is 22](#_bookmark15)

[Table 2-8 MMAP field corresponding to the space access property 22](#_bookmark16)

[Table 2-9 Secondary XBAR Address window conversion Register Table 22](#_bookmark17)

[Table 2-10. Level 2 XBAR default address configuration 25](#_bookmark19)

[Table 2-11 Chip configuration register (physical address 0x1fe00180) 25](#_bookmark20)

[Table 2-12 Chip Sampling register (physical address 0x1fe00190](#_bookmark21)

[Table 2-13 Chip node and processor core software frequency doubling setting register (physical address 0x1fe001b0) 27](#_bookmark22)

[Table 2-14 Chip memory and HT clock software frequency multiplier setting register (physical address 0x1fe001c0) 28](#_bookmark23)

[Table 2-15 Register for frequency division setting of chip processor core software (physical address 0x1fe001d0](#_bookmark24)

[Table 4-1 Shared Cache lock window register configuration 33](#_bookmark28)

[Table 5-1 Matrix processing programming interface description 34](#_bookmark30)

[Table 5-2 Matrix processing register address description 35](#_bookmark31)

[Table 5-3 The trans\_CTRL register indicates 35](#_bookmark32)

[Table 5-4 The trans\_status register shows 36](#_bookmark33)

[Table 6-1. Registers associated with interrupt between processor cores and their functions](#_bookmark35)

[Table 6-2 Intercore interrupt and communication registers for Processor No. 0](#_bookmark36)

[Table 6-3 Intercore interrupt and communication registers for Processor core No. 1](#_bookmark37)

[Table 6-4 Intercore interrupt and communication registers for No. 2 processor core](#_bookmark37)

[Table 6-5 Intercore interrupt and communication registers for No. 3 processor core](#_bookmark37)

[Table 7-1 Interrupt control register 41](#_bookmark40)

[Table 7-2 IO control register address 41](#_bookmark41)

[Table 7-3 Description of interrupt routing register](#_bookmark42)

[Table 7-4 Interrupt routing register address 42](#_bookmark43)

[Table 8-1 Temperature sampling register description 43](#_bookmark46)

[Table 8-2 High and low temperature interrupt register description 44](#_bookmark48)

[Table 8-3 High temperature and frequency drop control register description 45](#_bookmark49)

[Table 10-1 Related pin signals of HyperTransport Bus](#_bookmark65)

[Table 10-2 Command received by the HyperTransport receiver 61](#_bookmark67)

[Table 10-3 Outgoing commands in two modes 61](#_bookmark68)

[Table 10-4 Shows the distribution of address Windows for the default four HyperTransport interfaces](#_bookmark71)

[Table 10-5 Distribution of address Windows within the HyperTransport interface of Loongson 3 processor](#_bookmark73)

[Table 10-6 Address window 63 provided by Longson 3A300/3B3000 processor HyperTransport interface](#_bookmark74)

[Table 10-7 Software visible registers List 64](#_bookmark76)

[Table 10-8 Bus Reset Control register definition 66](#_bookmark78)

[Table 10-9 Command, Capabilities Pointer, Capability ID register definition 66](#_bookmark79)

[Table 10-10 Link Config, Link Control register definition 67](#_bookmark80)

[Table 10-11 Revision ID, Link Freq, Link Error, and Link Freq Cap register definition 68](#_bookmark81)

[Table 10-12 Feature Capability Register definition 69](#_bookmark83)

[Table 10-13 MISC register definition 69](#_bookmark84)

[Table 10-14 receives diagnostic register 71](#_bookmark86)

[Table 10-15 Interrupt routing mode select register 71](#_bookmark87)

[Table 10-16 Receive buffer initial register 71](#_bookmark88)

[Table 10-17 HT bus receive address window 0 enable (external access) register definition 72](#_bookmark90)

[Table 10-18 HT bus receive address window 0 base address (external access) register definition 72](#_bookmark90)

[Table 10-19 HT bus receive address window 1 enable (external access) register definition 73](#_bookmark91)

[Table 10-20 HT bus receive address window 1 Base address (external access) register definition 73](#_bookmark91)

[Table 10-21 HT bus receive address window 2 enable (external access) register definition 73](#_bookmark91)

[Table 10-22 HT bus receiver address window 2 Base address (external access) register definition 74](#_bookmark92)

[Table 10-23 HT bus receive address window 3 enable (external access) register definition 74](#_bookmark92)

[Table 10-24 HT bus receiver address window 3 Base address (external access) register definition 74](#_bookmark92)

[Table 10-25 HT bus receive address window 4 enable (external access) register definition 75](#_bookmark94)

[Table 10-26 HT bus receiver address window 4 Base address (external access) register definition 75](#_bookmark94)

[Table 10-27 HT bus interrupt vector register definition (1) 76](#_bookmark95)

[Table 10-28 HT bus interrupt vector register definition (2) 76](#_bookmark95)

[Table 10-29 HT bus interrupt vector register definition (3) 77](#_bookmark96)

[Table 10-30 HT bus interrupt vector register definition (4) 77](#_bookmark96)

[Table 10-31 HT bus interrupt vector register definition (6) 77](#_bookmark96)

[Table 10-32 HT bus interrupt vector register definition (7) 77](#_bookmark96)

[Table 10-33 HT bus interrupt vector register definition (8) 78](#_bookmark98)

[Table 10-34 HT bus interrupt enable register definition (1) 79](#_bookmark99)

[Table 10-35 HT bus interrupt enable register definition (2) 79](#_bookmark99)

[Table 10-36 HT bus interrupt enable register definition (3) 79](#_bookmark99)

[Table 10-37 HT bus interrupt enable register definition (4) 79](#_bookmark99)

[Table 10-38 HT bus interrupt enable register definition (5) 80](#_bookmark100)

[Table 10-39 HT bus interrupt enabled register definition (6) 80](#_bookmark100)

[Table 10-40 HT bus interrupt enable register definition (7) 80](#_bookmark100)

[Table 10-41 HT bus interrupt enable register definition (8) 80](#_bookmark100)

[Table 10-42 Interrupt Register definition 81](#_bookmark102)

[Table 10-43 The Dataport register definition 81](#_bookmark103)

[Table 10-44 IntrInfo register definition (1) 81](#_bookmark104)

[Table 10- 45 IntrInfo register definition (2) 81](#_bookmark104)

[Table 10-46 HT bus POST address window 0 enable (internal access) 82](#_bookmark106)

[Table 10-47 HT bus POST Address window 0 base address (internal access) 82](#_bookmark106)

[Table 10-48 HT bus POST address window 1 enable (internal access) 83](#_bookmark108)

[Table 10-49 HT bus POST address window 1 Base address (internal access) 83](#_bookmark108)

[Table 10-50 HT bus prefetch address window 0 enabling (internal access) 83](#_bookmark109)

[Table 10-51 HT bus prefetchable address window 0 base address (internal access) 84](#_bookmark111)

[Table 10-52 HT bus can prefetch address window 1 enabling (internal access) 84](#_bookmark111)

[Table 10-53 HT bus prefetchable address window 1 Base address (internal access) 84](#_bookmark111)

[Table 10-54 HT bus Uncache Address window 0 enables (internal access) 85](#_bookmark112)

[Table 10-55 HT Bus Uncache Address window 0 base address (internal access) 85](#_bookmark112)

[Table 10-56 HT bus Uncache Address window 1 enables (internal access) 85](#_bookmark112)

[Table 10-57 HT Bus Uncache Address window 1 Base address (internal access) 86](#_bookmark113)

[Table 10-58 HT bus Uncache Address window 2 enabling (internal access) 86](#_bookmark113)

[Table 10-59 HT Bus Uncache Address window 2 Base address (internal access) 86](#_bookmark113)

[Table 10-60 HT bus Uncache Address window 3 enabled (internal access) 87](#_bookmark116)

[Table 10-61 HT Bus Uncache Address window 3 Base address (internal access) 87](#_bookmark116)

[Table 10-62 HT bus P2P address window 0 enable (external access) register definition 87](#_bookmark115)

[Table 10-63 HT bus P2P address window 0 base address (external access) register definition 88](#_bookmark117)

[Table 10-64 HT bus P2P address window 1 enabling (external access) register definition 88](#_bookmark117)

[Table 10-65 HT bus P2P address window 1 base address (external access) register definition 88](#_bookmark117)

[Table 10-66 command sends cache size register 89](#_bookmark119)

[Table 10-67 Data send cache size register 89](#_bookmark119)

[Table 10-68 sends cache debug register 90](#_bookmark121)

[Table 10-69 Impedance matching control register 91](#_bookmark123)

[Table 10-70 Revision ID register 91](#_bookmark124)

[Table 10-71 Error Retry control register 91](#_bookmark125)

[Table 10-72 Retry Count register 92](#_bookmark127)

[Table 10-73 Link Train register 92](#_bookmark128)

[Table 10-74 Short timeout register 93 of Training 0](#_bookmark130)

[Table 10-75 Training 0 timeout long count register 94](#_bookmark132)

[Table 10-76 Training 1 counting register 94](#_bookmark133)

[Table 10-77 Training 2 counting register 94](#_bookmark133)

[Table 10-78 Training 3 counting register 95](#_bookmark135)

[Table 10-79 Software frequency configuration register 95](#_bookmark136)

[Table 10-80 PHY configuration register 96](#_bookmark138)

[Table 10-81 Link Initialization debug register 97](#_bookmark140)

[Table 10-82 LDT debug register 98](#_bookmark143)

[Table 11-1 PCI controller configuration header 101](#_bookmark149)

[Table 11-2 PCI control register 102](#_bookmark150)

[Table 11-3 PCI/PCIX Bus request and reply line allocation 105](#_bookmark152)

[Table 11-4 LPC controller address space distribution 106](#_bookmark154)

[Table 11-5 Meaning of LPC configuration register](#_bookmark155)

[Table 11-6 SPI controller address space distribution 114](#_bookmark163)

[Table 11-7 IO control register 118](#_bookmark168)

[Table 11-8 registers describe 119 in detail](#_bookmark169)

## An overview of the

#### Introduction to loong chip series processor

The godson processor mainly consists of three series. Loongson No. 1 processor and its IP series are mainly for embedded applications, Loongson No. 2 superstandard processor and its IP series are mainly for desktop applications, and Loongson No. 3 multi-core processor series are mainly for server and high-performance computer applications. According to application needs, part of the Loongson 2 can also be oriented to some high-end embedded applications.

Part of the low end loongson 3 can also be used for some desktop applications. The three series will evolve in parallel.

Based on the scalable multi-core interconnection architecture, the Loongson 3 multi-core series processor integrates multiple high-performance processor cores and a large number of 2-level Caches on a single chip, and realizes multi-chip interconnection through high-speed I/O interface to form a larger scale system.

The telescopic interconnection structure adopted by Loongson no. 3 is shown in Figure 1-1 below. Longson no. 3 chip and multiple chip system are two

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



**L2**

**L2 L2**

(a)

**L2**

**P0, P1 P2, P3**

**8 by 8 switch**

Node 3 and 2-D interconnection structure, (a) node structure, (b) mesh network of 2\*2 connected 16 processors, (c)

The 4\*4 mesh network connects 64 processors.

(c)

(b)

**E S W**

**N**

**E S W**

**N**

Figure 1-1 System structure of No. 3 Loong Chip

The node structure of No.3 is shown in Figure 1-2 below. Each node has two levels of AXI switch to connect processor and share

Cache, memory controller, and IO controller.  The first level of AXI cross-switches (called the X1 Switch, or X1 for short) connects the processor to the Shared Cache. The second level cross Switch (called the X2 Switch, or X2 for short) connects the Shared Cache to the memory controller.



**The MC**

**Xconf**

SS SM

MA0MA1MA2MA3

**X2 Switch**

WM

m0m1m2m3m4m5m6m7

**The X1 Switch**

s0s1s2s3s4s5s6s7

WSES

**S0S1S2S3**

EM

**P0P1P2P3**

NSNM

FIG. 1-2 Structure of Nodal 3 of the Loong Chip

In each node, the X1 cross switch of up to 8\*8 connects four GS464 processor cores through four Master ports

(P0, P1, P2, P3), connect four interleave Shared Cache blocks uniformly addressed through four Slave ports (S0, S1, S2, S3), and connect four Master/Slave ports to other nodes or IO nodes in the east, south, west, and north directions (EM/ES, SM/SS, WM/WS, NM/NS).

X2 cross-switch connects four Shared caches through four Master ports, at least one Slave port connects to a memory controller, at least one cross-switch configuration module (Xconf) is used to configure the address window of X1 and X2 of this node, etc. You can also connect more memory controllers and IO ports as needed.

#### 1.2 Introduction to Loongson 3A3000/3B3000

The 3A3000/3B3000 is the process upgrade version of the 4core processor of the 3A2000/3B2000. Compared with the 3A1000, the packaging pin of the PLL\_AVDD is changed from 2.5V to 1.8V. Compared with the 3A2000/3B2000, more PLL\_AVDD is used. Longshon 3A3000/3B3000 is a processor configured with a single node and 4 cores. It is manufactured by a 28nm process with a working main frequency of 1.2ghz to 1.5ghz. The main technical features are as follows:

* Four 64-bit quad-emission superscalar GS464e processor cores are integrated on the chip.
* Integrated 8MB split Shared three-level Cache(composed of 4 individual modules, each with a capacity of 2MB);
* Maintain Cache consistency of multi-core and I/O DMA access through directory protocol;
* On - chip integrated two 64 - bit ECC, 667MHz DDR2/3 controller;
* 3B3000 chips with 2 16-bit 1.6ghz HyperTransport controllers (HT);
* HT1 in 3A3000 chips is a 16-bit 1.6ghz HT controller, and HT0 is not available.
* Each 16-bit HT port is split into two 8-way HT ports for use.
* Integrated 32-bit 33MHz PCI on chip;
* Integrated one LPC, two UART, one SPI and 16-channel GPIO interface on the chip. Compared with Loongson 3A2000/3B2000, its main improvements are as follows:
* Processor core microstructure upgrade;
* Memory controller structure, frequency upgrade;
* HT controller structure and frequency upgrading;
* The performance of the whole chip is optimized and improved.

The overall architecture of Loongson 3A300/3B3000 chip is realized based on two-stage interconnection. The structure is shown in Figure 1-3 below.

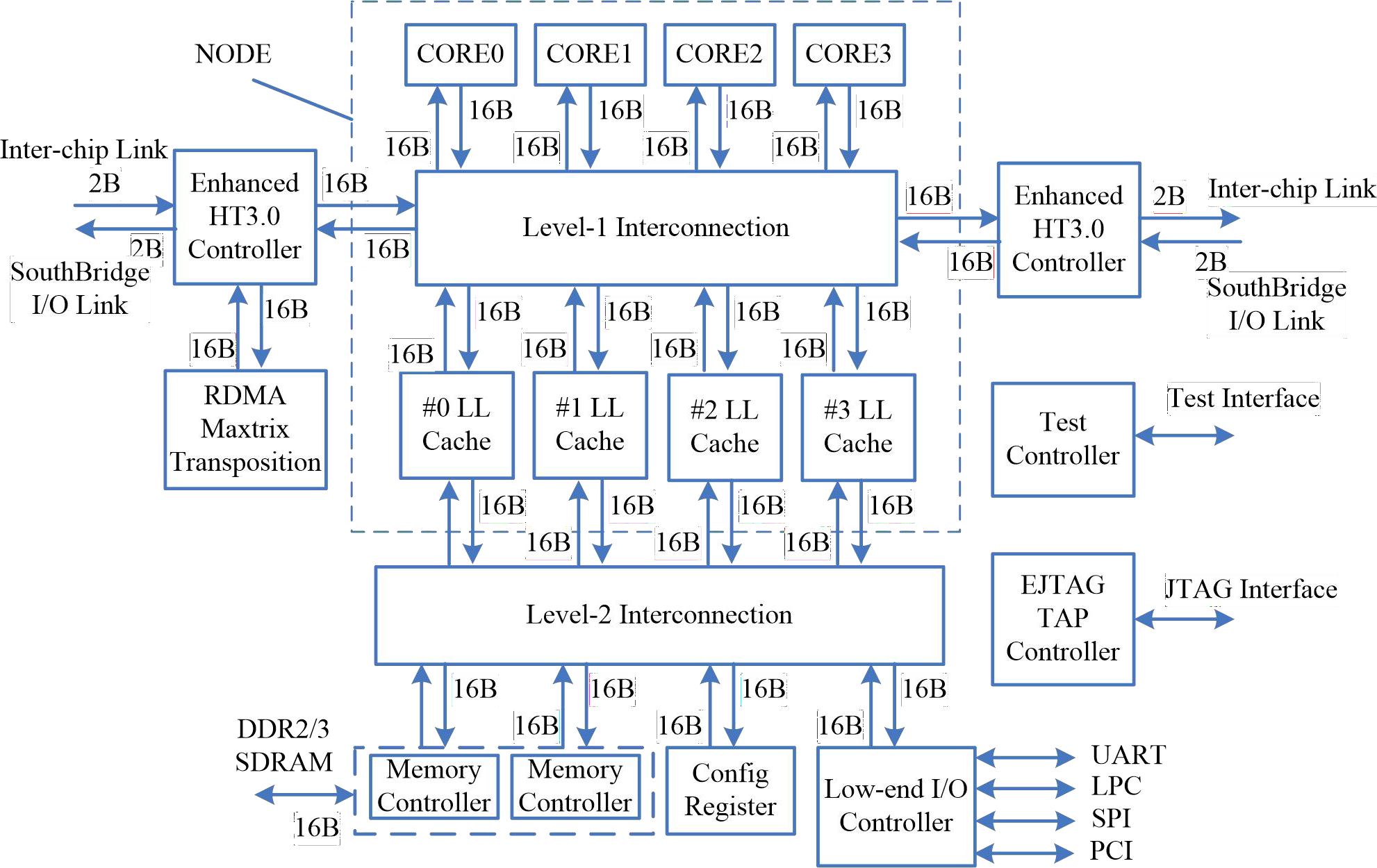


Figure 1-3 Structure of longson 3A3000/3B3000 chip

The first interconnection USES a 6x6 cross-switch for connecting four GS464e cores (as the primary device), four Shared Cache modules (as the Slave device), and two IO ports (each using a Master and a Slave).

Each IO port connected by the primary interconnect switch is connected to a 16-bit HT controller. Each 16-bit HT port can also be used as two 8-bit HT ports. The HT controller is connected to the first-level interconnection switch through a DMA controller, which is responsible for DMA control of IO and maintenance of inter-chip consistency. The DMA controller of The Loongson 3 can also be configured for prefetching and matrix transpose or move.

The second interconnection USES a 5x4 cross-switch to connect four Shared Cache modules (as the primary device), two DDR2/3 memory controllers, low speed and high speed I/O (including PCI, LPC, SPI, etc.), and a configuration register module within the chip.

The two interconnect switches use a read-write separated data channel with a width of 128 bits and operate at the same frequency as the processor core to provide high-speed on-chip data transmission.

## System configuration and control

#### Chip operation mode

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

* + - Single chip mode. The system contains only one piece of longson 3A300/3B3000, which is a symmetric multiprocessor system (SMP).
    - Multi-chip interconnection mode. The system contains 2 or 4 pieces of longshank 3A300/3B3000, which is interconnect through HT port of longshank 3A300/3B3000. It is a cc-NUMa.
    - Mass interconnection model. Multi-chip large-scale extension interconnection through special extension bridge constitutes a large-scale non-uniform access multiprocessor system (CC-NUMA).

#### Control pin description

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

Table 2-1 Description of control pins

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 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 multi-chip consistent interconnection mode  1 'b0 represents the single-chip mode | | | | |
| NODE\_ID [1:0] |  | Represents the processor number in multi-chip consistent interconnection mode | | | | |
| CLKSEL [15:0] |  | HT clock control | | | | |
|  | signal | role | |  |
| CLKSEL [15] | 1 'b1 means HT controller frequency only adopts hardware setting  1 'b0 means HT controller frequency can be set by software | |  |
| CLKSEL [14] | 1 'b1 means HT PLL adopts ordinary clock input  1 'b0 means HT PLL adopts differential clock input | |  |
| CLKSEL [12] | 2 'b00 represents a PHY clock of 1.6ghz  2 'B01 represents a PHY clock of 3.2ghz  2 'b10 represents a PHY clock of 1.2ghz  2 'b11 represents a PHY clock of 2.4ghz | |  |
|  |  |  | CLKSEL [10] | 2 'b00 means HT controller clock is PHY clock 8 frequency division  2 'B01 means that HT controller clock is PHY clock 4 frequency division  2 'B10 means HT controller clock is PHY clock 2 frequency division  2 'b11 means that HT controller clock is SYSCLOCK | |  |
| Note: CLKSEL[13:10] == 4 'b1111, HT controller clock is in bypass mode, use external input 100MHz reference clock directly  MEM clock control   |  |  | | --- | --- | | signal | role | | CLKSEL [will] | 5 'B11111 means that MEM clock adopts MEMCLK directly  5 'b01111 means that MEM clock is set by software. See setting method  Section 2.6 shows  In other cases, MEM clock is MEMCLK \*(Clksel [8:5]+30)/(Clksel [9]+3) note:  Memclk \*(Clksel [8:5]+30) must be 1.2ghz ~ 3.2ghz  Memclk is the input reference clock, which must be 20~40MHz |   CORE clock control | | | | |
|  | signal | | role |  |
| CLKSEL [Wednesday] | | 5 'b11111 means the CORE clock USES sySCLk directly  5 'b011xx means that the CORE clock is set by software. See the setting method  Section 2.6 explains.  5 'b01111 is in normal working mode, other cases are in debug mode  5 'b0110x indicates that the processor interface is in asynchronous mode  5 'B011x0 represents delayed debugging 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  7HT bus cold start force set to 1.0 mode  6:4 needs to be set to 000  3PCI master device mode | | | | |
|  |  | Set 2 to 0  Use external PCI arbitrators  0 USES SPI to boot | | | | |

* 1. The Cache consistency

The longson 3A300/3B3000 is maintained by the hardware for Cache consistency between the processor and I/O accessed through HT port, but the hardware does not maintain Cache consistency for I/O devices accessed through PCI. During driver development, the software needs to maintain Cache consistency during Direct Memory Access (DMA) transfer to a device accessed through PCI.

#### The physical address space distribution at the node level of the system

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

Development of the extension compatible. The physical address width of the entire system is 48 bits. According to the address of the high 4 bits, the entire address space

Is uniformly distributed to 16 nodes, that is, each node allocated 44 bit address space.

Longson 3A3000/3B3000 processor can directly use 4 chips to build CC-NUMa system. The processor number of each chip is determined by pin NODEID. The address space distribution of each chip is as follows:

Table 2-2 Global address distribution of the system at node level

|  |  |  |  |
| --- | --- | --- | --- |
| 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 single node 4-core configuration is adopted for the 3A300/3B3000 chip, so the corresponding addresses of DDR memory controller, HT bus and PCI bus integrated with the 3A300/3B3000 chip are contained in the interior of each node from 0x0 (including) to 0x1000\_0000\_0000 (excluding), and the 44-bit address space is further distributed evenly to up to 8 devices connected within the node. The low 43-bit address is owned by the four Shared Cache modules, and the high 43-bit address is further addressed

The [43:42] bits are distributed to devices connected to four directional ports. Depending on the configuration of the chip and system structure, if there is no slave device connected on a port, the corresponding address space is reserved address space and is not allowed to access.

The address space corresponding to each slave end of the first stage cross switch in longson 3A300/3B3000 chip is as follows

|  |  |  |  |
| --- | --- | --- | --- |
| equipment | Address [43:41] | Start address within the node | Node end 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 |

Different from the mapping relationship of directional ports, longson 3A300/3B3000 can determine the cross-addressing mode of Shared Cache according to the practical application of access behavior. The four Shared Cache modules in the node correspond to a total 43-bit address space, and the address space corresponding to each module is determined according to some two-bit selection bit of the address bit, and can be dynamically configured by software. A configuration register called SCID\_SEL is set up to determine the address selection bit, as shown in the table below. By default, the distribution takes the form of a [7:6] status hash, in which the [7:6] two digits of the address determine the corresponding Shared Cache number. The register address is 0x3FF00400.

Table 2-4 Address distribution in nodes

|  |  |  |  |
| --- | --- | --- | --- |
| SCID\_SEL | Address bit selection | SCID\_SEL | Address bit selection |
| 4 'h0 | 7:6 | 4 'h8 | " |
| 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 |

#### The physical address space distribution within a node

The default distribution of 44-bit physical addresses in each node of Longson 3A300/3B3000 processor is shown in table 2-2

|  |  |  |  |
| --- | --- | --- | --- |
| The starting address | End address | The name of the | instructions |
|  |  |  |  |
| 0 x0000\_1000\_0000 | 0 x0000\_1fff\_ffff | Low speed IO | Mapping needs to be done using a secondary cross switch |

|  |  |  |  |
| --- | --- | --- | --- |
| 0 x2000\_0000\_0000 | 0 x2fff\_ffff\_ffff | 2 | 2 |
| 0 x3000\_0000\_0000 | 0 x3fff\_ffff\_ffff | 3 | 3 |

Longson 3A3000/3B3000 adopts single node 4-core configuration, so longson

#### Address routing distribution and configuration

The routing of Loongson 3A300/3B3000 is mainly realized through the two-stage cross switch of the system. The first-level cross switch can conduct routing configuration for the requests received by each Master port. Each Master port has 8 address Windows, which can complete the target routing selection of 8 address Windows. Each address window is composed of three 64-bit registers, BASE, MASK and MMAP. The BASE is aligned with K bytes. MASK adopts a format similar to network MASK in which the high digit is 1. The low three-digit MMAP represents the number of the corresponding target Slave port, MMAP[4] represents the point allowed, MMAP[5] represents the block read allowed, MMAP[6] represents the interleaving enable of Scache, and MMAP[7] represents the window enable.

Table 2-5 MMAP field corresponding to the space access properties

|  |  |  |  |
| --- | --- | --- | --- |
| [7] | [6] | [5] | [4] |
| The window can make | Allows interleaving access to SCACHE, valid when Slave number is 0, and routes the hit window address request as configured in the SCID\_SEL section above | Allow the block read | Allowed to take to |

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

As the default route is fixed, the configuration window is closed when starting power on, and the system software is required to enable configuration of the loongson no. 3.

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

Table 2-6 Register table of address window of first level cross switch

|  |  |  |  |
| --- | --- | --- | --- |
| 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 |
| 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 |

The configuration register address space, DDR2 address space and PCI address space are three IP related address Spaces in the secondary XBAR of Loongson 3.  The address window is set for routing and address translation by THE IP with CPU and PCI-DMA as the master device. Both 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 byte, while MASK adopts a format similar to network MASK with the high order of 1. MMAP contains the converted address, route selection and enabling control alleles, as shown in the following table:

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

Where, the equipment corresponding to the device number is shown in the following table:

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

|  |  |
| --- | --- |
| From the device number | The default value |
| 0 | 0 DDR2/3 controller |
| 1 | No. 1 DDR2/3 controller |
| 2 | Low speed I/O (PCI, LPC, etc.) |
| 3 | Configuration register |

The window enabling bits have the following meanings:

Table 2-8 MMAP field corresponding to the space access properties

|  |  |  |  |
| --- | --- | --- | --- |
| [7] | [6] | [5] | [4] |
| The window can make | Allows interleaving access to DDR, valid when the slave device number is 0, and routes requests to hit window addresses in a "interleaving select bit" configuration. You want to interleave the energy bit greater than 10 | Allow the block read | Allowed to take to |

It should be noted that the window configuration of the first level XBAR cannot translate the address of the Cache consistency request, otherwise the address at SCache will be inconsistent with the address at the processor's first level Cache, resulting in a Cache consistency maintenance error.

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

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

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

Table 2-9 The second-level XBAR address window conversion register table

|  |  |  |  |
| --- | --- | --- | --- |
| address | register | describe | The default value |

|  |  |  |  |
| --- | --- | --- | --- |
| 3 ff0 0000 | CPU\_WIN0\_BASE | The base address of CPU window 0 | 0 x0 |
| 3 ff0 0008 | CPU\_WIN1\_BASE | The base address of CPU window 1 | 0 x1000\_0000 |
| 3 ff0 0010 | CPU\_WIN2\_BASE | The base address of CPU window 2 | 0 x0 |
| 3 ff0 0018 | CPU\_WIN3\_BASE | The base address of CPU window 3 | 0 x0 |
| 3 ff0 0020 | CPU\_WIN4\_BASE | The base address of CPU window 4 | 0 x0 |
| 3 ff0 0028 | CPU\_WIN5\_BASE | The base address of CPU window 5 | 0 x0 |
| 3 ff0 0030 | CPU\_WIN6\_BASE | The 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 | 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 | Mask for CPU window 2 | 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 | Mask for CPU window 7 | 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 | The base address of PCI window 0 | 0 x8000\_0000 |
| 3 ff0 0108 | PCI\_WIN1\_BASE | The base address of PCI window 1 | 0 x0 |
| 3 ff0 0110 | PCI\_WIN2\_BASE | The base address of PCI window 2 | 0 x0 |

|  |  |  |  |
| --- | --- | --- | --- |
| 3 ff0 0118 | PCI\_WIN3\_BASE | The base address of PCI window 3 | 0 x0 |
| 3 ff0 0120 | PCI\_WIN4\_BASE | The base address of PCI window 4 | 0 x0 |
| 3 ff0 0128 | PCI\_WIN5\_BASE | The base address of PCI window 5 | 0 x0 |
| 3 ff0 0130 | PCI\_WIN6\_BASE | The base address of PCI window 6 | 0 x0 |
| 3 ff0 0138 | PCI\_WIN7\_BASE | The base address of 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 interval after chip startup

(256M) maps to the address interval of 0x00000000-0x0fffffff of DDR2, the CPU's 0x10000000-0x1FFfffff (256M) maps to PCI's 0x10000000-0x1FFfffff, and PCIDMA's 0x80000000

The address interval of -0x8fffffff (256M) maps to the address interval of 0x00000000-0x0fffffff of DDR2. Software can modify the corresponding configuration registers to achieve the new address space routing and transformation.

In addition, when read access to illegal address occurs due to CPU guessing execution, all 8 address Windows are not hit, and all 0 data is returned to THE CPU by the configuration register module to prevent CPU from dying.

Table 2-10 Secondary XBAR default address configuration

|  |  |  |
| --- | --- | --- |
| Base address | high | The owner of the |
| 0 x0000\_0000\_0000\_0000 | 0 x0000\_0000\_0fff\_ffff | DDR controller 0 |
| 0 x0000\_0000\_1000\_0000 | 0 x0000\_0000\_1fff\_ffff | Low speed I/O (PCI, etc.) |

#### Chip configuration and sampling registers

The Chip\_config and Chip\_sample of longcore 3A3000/3B3000 provide the Chip\_sample reading and writing mechanism for 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 | 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 | 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 |
| they | HT0\_freq\_scale\_ctrl | RW | 3 'b111 | HT controller 0 frequency division |
| 27 | HT0\_clken | RW | 1 'b1 | Whether or not I enabled HT0 |
| he | HT1\_freq\_scale\_ctrl | RW | 3 'b111 | HT controller 1 frequency division |
| 31 | HT1\_clken | RW | 1 'b1 | Whether I enabled HT1 |
| 42:40 | Node0\_freq\_ CTRL | RW | 3 'b111 | Node 0 frequency division |
| 43 | - | RW | 1 'b1 |  |
| 46:44 | Node1\_freq\_ CTRL | RW | 3 'b111 | Node 1 frequency division |
| 47 | - | RW | 1 'b1 |  |
| 63:56 | Cpu\_version | R | 2 'h39 | The CPU version |
| A 95-64 |  |  |  | (empty) |
| A 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 multiplication Settings |
| 55:48 | Bad\_ip\_core | R |  | Core7 - core0 is bad |
| 57:56 | Bad\_ip\_ddr | R |  | Are 2 DDR controllers broken |
| 61:60 | Bad\_ip\_ht | R |  | Is 2 HT controllers broken |
| A 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℃) |
| A 111-96 | Thsens0\_out | R |  | Temperature sensor 0 ℃  Node temperature =Thens0\_out  \* 731/0 x4000-273  Temperature range -40-125 degrees |
| A 127-112 | Thsens1\_out | R |  | Temperature sensor 1 ℃  Node temperature =Thens1\_out  \* 731/0 x4000-273  Temperature range -40-125 degrees |
| other |  | R |  | reserve |

The following sets of software frequency doubling setting registers are used to set the operating frequency of each clock under the CLKSEL configuration as software control mode (refer to the CLKSEL setting method in Section 2.2). Where, MEM CLOCK is configured to correspond to memory controller and bus CLOCK frequency; CORE CLOCK corresponds to the CLOCK frequency of the processor CORE, on-chip network, and high-speed Shared cache; HT CLOCK corresponds to the CLOCK frequency of HT controller.

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

For HT CLOCK configuration method is quite special, please refer to the specific configuration method in Section 10.5.28.

In software control mode, the default corresponding CLOCK frequency is the frequency of external reference CLOCK (for CORE CLOCK, it is the corresponding frequency of pin SYS\_CLK; For MEM CLOCK, it is the corresponding frequency of pin MEM\_CLK), and it is necessary to set the CLOCK software during processor startup. Each clock should be set in the following manner:

1. Set registers other than SEL\_PLL\_\* and SOFT\_SET\_PLL, which are written to 0 during the set.
2. Set SOFT\_SET\_PLL to 1 with the other register values unchanged.
3. Wait register lock signal LOCKED\_\* for 1;
4. Set SEL\_PLL\_\* to 1 and the corresponding clock frequency will switch to the frequency set by the software. 3 a3000/3 b3000, can use two different PLLL1 L2 se ri al\_m yao dePLL

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A domain** | **The field name** | **access** | **Reset value** | **describe** |
| 0 | SEL\_PLL\_NODE | RW | 0 x0 | Node clock is not software bypass the whole  PLL |
| 1 | SEL\_PLL\_NODE | RW | 0 x0 | Core clock non-software bypass the entire PLL |
| 2 | SOFT\_SET\_PLL | RW | 0 x0 | Allows software to set PLL |
| 3 | BYPASS\_L1 | RW | 0 x0 | Bypass L1 PLL |
| Indeed, | - | RW | 0 x0 | - |
| 16 | LOCKED\_L1 | R | 0 x0 | Is L1 PLL locked |
| 17 | LOCKED\_L2 | R | 0 x0 | Whether L2 PLL is locked |
| That is | - | R | 0 x0 | - |
| 19 | PD\_L1 | RW | 0 x0 | Close the L1 PLL |
| 20 | PD\_L2 | RW | 0 x0 | Close the L2 PLL |
| 21 |  |  |  |  |
| 22 | Serial\_mode | RW | 0 x0 | 0: L1 PLL is selected as the main clock  1: L2 PLL is selected as the main clock |
| 23 | Serial\_mode3 | RW | 0 x0 | 0: Select NODE clock as the core clock  1. CORE clock is selected as the CORE clock |
| Thus for | - | RW |  | - |
| Upon this | L1\_DIV\_REFC | RW | 0 x1 | L1 PLL input parameter |
| 40:32 | L1\_DIV\_LOOPC | RW | 0 x1 | L1 PLL input parameter |
| 41 |  |  |  |  |
| 47:42 | L1\_DIV\_OUT | RW | 0 x1 | L1 PLL input parameter |
| 50:48 | L2\_DIV\_REFC | RW | 0 x1 | L2 PLL input parameters |
| 53:51 |  |  |  |  |
| 63:54 | L2\_DIV\_LOOPC | RW | 0 x1 | L2 PLL input parameters |
| A 69-64 | L2\_DIV\_OUT | RW | 0 x1 | The L2 PLL input parameter must have one and only one bit of 1 |
|  |  |  |  |  |
| 96 | BBGEN\_enable | RW | 0 x0 | Bias can make |
| 97 | BBMUX\_first | RW | 0 x0 | Set to switch voltage mode first |
| A 99-98 | BBMUX\_SEL\_0 | RW | 0 x0 | The setting of the BBMUX\_SEL\_0 |
| A 101-100 | BBGEN\_feedback | RW | 0 x0 | Disable BBGEN feedback signals |
|  |  |  |  |  |
| A 107-104 | BBGEN\_vbbp\_val | send | 0 x0 | Settings for Vbbp |
| A 111-108 | BBGEN\_vbbn\_val | send | 0 x0 | Set value for Vbbn |
|  |  |  |  |  |
| A 123-122 | BBMUX\_SEL\_1 | RW | 0 x0 | The setting of the BBMUX\_SEL\_1 |
| A 125-124 | BBMUX\_SEL\_2 | RW | 0 x0 | The setting of the BBMUX\_SEL\_2 |
| 127-126 | BBMUX\_SEL\_3 | RW | 0 x0 | The setting of the BBMUX\_SEL\_3 |
| other | - | RW |  | reserve |

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

The VCO frequency of L1 PLL (the part in brackets above) must be within the range of 1.2ghz -- 3.2ghz. This requirement also applies to MEM PLL and HT PLL. The VCO frequency of L2 PLL must be within the range of 3.2ghz -- 6.4ghz.

Table 2-14 Chip memory and HT clock software frequency multiplier Setting register (physical address 0x1fe001C0)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A domain** | **The field name** | **access** | **Reset value** | **describe** |
| 0 | SEL \_MEM\_PLL | RW | 0 x0 | MEM clock is not software bypass the whole PLL |
| 1 | SOFT\_SET\_MEM\_PLL | RW | 0 x0 | Allows software to set MEM PLL |
| 2 | BYPASS\_MEM\_PLL | RW | 0 x0 | Bypass MEM\_PLL |
| o |  |  |  |  |
| 6 | LOCKED\_MEM\_PLL | R | 0 x0 | Is MEM\_PLL locked |
| 7 | PD\_MEM\_PLL | RW | 0 x0 | Close the MEM PLL |
| Will you | MEM\_PLL\_DIV\_REFC | RW | 0 x1 | MEM PLL input parameter  When NODE clock is selected (NODE\_CLOCK\_SEL is 1), it is used as a frequency divider input |
| brake | MEM\_PLL\_DIV\_LOOPC | RW | 0 x41 | MEM PLL input parameter |
| A partner | MEM\_PLL\_DIV\_OUT | RW | 0 x0 | MEM PLL input parameter |
| 30 | NODE\_CLOCK\_SEL | RW | 0 x0 | 0: Use MEM\_PLL as the MEM clock  1: Use NODE\_CLOCK as the divider input |
| 32 | SEL\_HT0\_PLL | RW | 0 x0 | HT0 non-software 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 of HT0 PLL |
| meanwhile | LOCKC\_HT0\_PLL | RW | 0 x0 | The phase accuracy used to determine whether HT0 PLL is locked or not |
| 38 | LOCKED\_HT0\_PLL | R | 0 x0 | Whether HT0\_PLL is locked |
| 45:40:15 | HT0\_DIV\_HTCORE | RW | 0 x1 | HT0 Core PLL input parameters |
| 48 | SEL\_HT1\_PLL | RW | 0 x0 | HT1 non-software 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 | The phase accuracy used to determine whether HT1 PLL is locked or not |
| 54 | LOCKED\_HT1\_PLL | R | 0 x0 | Whether HT1\_PLL is locked |
| 61:56 | HT1\_DIV\_HTCORE | RW | 0 x1 | HT1 Core PLL input parameters |
| other |  | RW |  | reserve |

Table 2-15 Register for frequency division setting of chip processor core software (physical address 0x1fe001D0)

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

## GS464e processor core

The GS464e is a quad-emitting 64-bit high-performance processor core. The processor core can be used as a single core for high-end embedded applications and desktop applications, or as a basic processor core to form an on-chip multi-core system for servers and high-performance machines. The GS464 cores in the loongson 3A300/3B3000 and the Shared Cache module form a multi-core structure of the last-level Cache on the distributed Shared chip through the AXI network. The main features of GS464 are as follows:

* MIPS64 compatible, support godson expansion instruction set;
* Four transmit superscalar structure, two fixed point, two floating point, two accessors;
* Each floating point unit supports full stream 64 bit/double 32 bit floating point multiplication and addition operation;
* The accessor supports 128-bit storage access, the virtual address is 64-bit, and the physical address is 48-bit.
* Support register renaming, dynamic scheduling, transfer prediction and other out-of-order execution techniques;
* 64 items fully connected plus 8 groups connected 1024 items, a total of 1088 TLB, 64 instruction TLB, variable page size;
* The size of the first-level instruction Cache and the data Cache is 64KB each, and the 4-way group is linked.
* The Victim Cache is a private secondary Cache, 256KB in size, connected to a 16-way block.
* Support non-blocking access, load-Speculation and other access optimization technologies;
* Supports Cache consistency protocol, which can be used for on-chip multi-core processors.
* Instruction Cache to achieve parity, data Cache to achieve ECC check;
* Support EJTAG debugging standard, convenient for software and hardware debugging;
* Standard 128-bit AXI interface.

The structure of GS464e is shown in the figure below. Refer to the GS464e User manual and the MIPS64 User manual for more details.

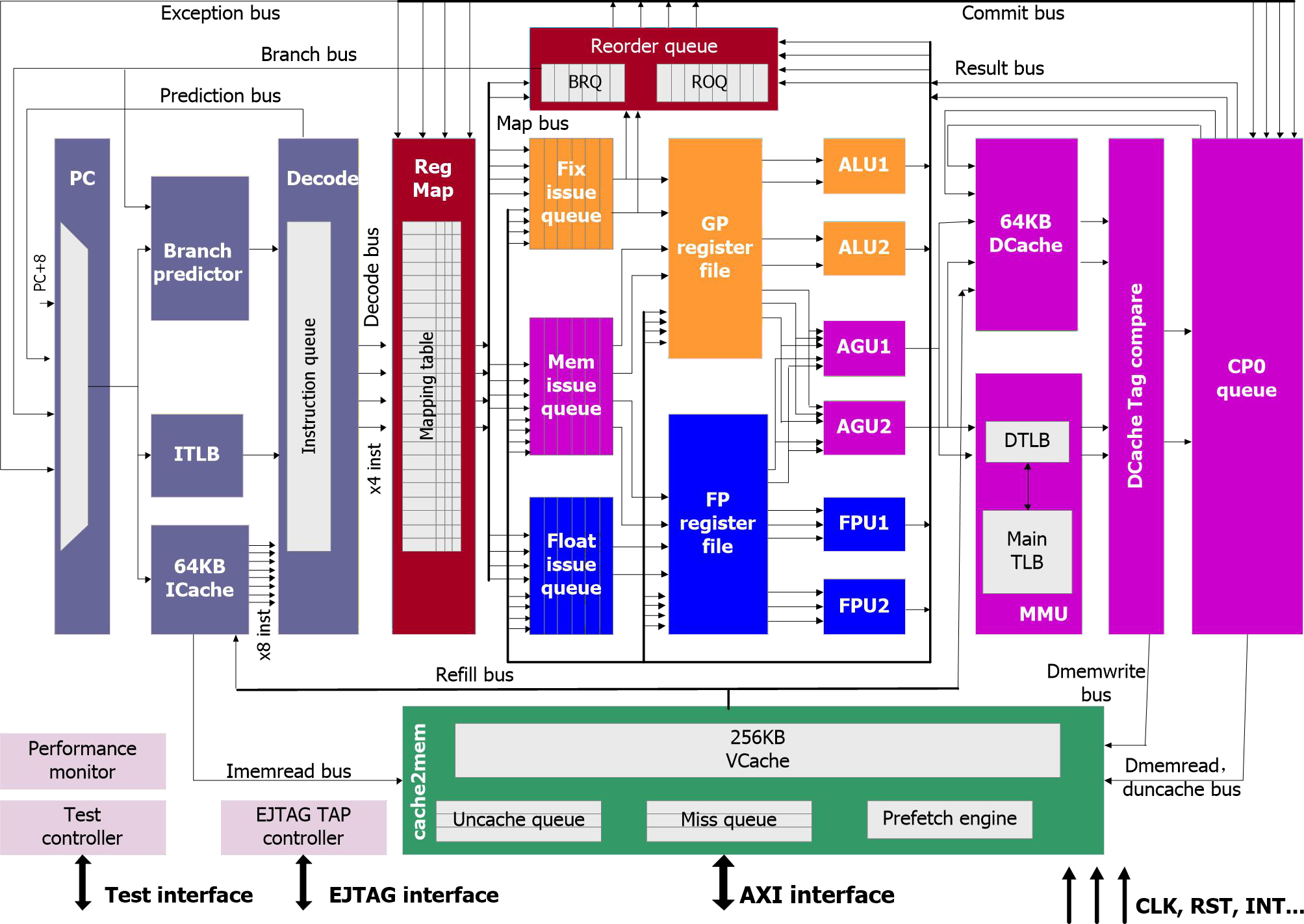


Figure 3-1 GS464e structure diagram

## Shared Cache (SCache)

SCache module is the three-level Cache Shared by all processor cores in the longson 3A300/3B3000 processor. The main features of SCache module include:

* Adopt 128-bit AXI interface.
* The 16-item Cache access queue.
* Keywords first.
* The fastest time to receive a failed read request to return data is 12 beats.
* Support for Cache consistency protocol through directories.
* It can be used for on-chip multi-core structure, and can also be directly connected to a single processor IP.
* The 16-channel group linkage structure is adopted.
* Support for ECC validation.
* Support DMA consistent reads and writes and prefetch reads.
* Supports 16 Shared Cache hashes.
* Support sharing Cache by window lock.
* Ensures that read data returns atomicity.

Shared Cache module includes Shared Cache management module Scachemanage and Shared Cache access module ScacheAccess. The Scachemanage module is responsible for the processor's access requests from the processor and DMA, while information such as tags, directories, and data from the Shared Cache is stored in the ScacheAccess module. To reduce power consumption, the tags, directories and DATA of the Shared Cache can be accessed separately. The Shared Cache status bits and w bits are stored together with the TAG, which is stored in TAG RAM, the directory in DIR RAM, and the DATA in DATA RAM. The invalidated request accesses the Shared Cache, reads out the TAG and directory of all channels at the same time, selects the directory according to the TAG, and reads the data according to the hit situation. Replace requests, refill requests, and write back requests only work with tags, directories, and data along the way.

To improve performance for certain computing tasks, Shared Cache adds a locking mechanism. A Shared Cache block in a locked area is locked and will not be replaced (unless the 16-way Shared Cache is full of locked blocks). The chip configuration register space can be used to dynamically configure four sets of lock window registers within the Shared Cache module, but it must ensure that one of the 16 Shared Cache lines is not locked. The size of each set of Windows can be adjusted according to mask, but cannot exceed 3/4 of the total Shared Cache size. In addition, when a Shared Cache receives a DMA write request, if the region being written is hit and locked in the Shared Cache, DMA writes are written directly to the Shared Cache rather than to memory.

Table 4-1 Configuration of the Shared Cache lock window register

|  |  |  |  |
| --- | --- | --- | --- |
| The name of the | address | A domain | describe |
| Sl yao ck0\_val d | 0 x3ff00200 | [63-63] | Lock window 0 valid bit |
| Sl yao ck0\_add ri | 0 x3ff00200 | [47:0] | Lock window lock address |
| Sl yao ck0\_mask | 0 x3ff00240 | [47:0] | Lock window mask 0 |
| Sl yao ck1\_val d | 0 x3ff00208 | [63-63] | Lock window 1 valid bit |
| Sl yao ck1\_add ri | 0 x3ff00208 | [47:0] | Lock address of lock window 1 |
| Sl yao ck1\_mask | 0 x3ff00248 | [47:0] | Lock 1 window mask |
| Sl yao ck2\_val d | 0 x3ff00210 | [63-63] | Lock window # 2 valid bit |
| Sl yao ck2\_add ri | 0 x3ff00210 | [47:0] | Lock 2 window lock address |
| Sl yao ck2\_mask | 0 x3ff00250 | [47:0] | Lock 2 window mask |
| Sl yao ck3\_val d | 0 x3ff00218 | [63-63] | Lock window 3 valid bit |
| Sl yao ck3\_add ri | 0 x3ff00218 | [47:0] | Lock address for lock window no.3 |
| Sl yao ck3\_mask | 0 x3ff00258 | [47:0] | Lock window mask number 3 |

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

## Matrix processing accelerator

Longson 3A300/3B3000 is built with two matrix processing accelerators independent of the processor core. Its basic function is to realize the function of column and column translocation or transfer of the matrix stored in memory from the source matrix to the target matrix through software configuration. The two accelerators are respectively integrated in two HyperTransport controllers of Longmancore 3A300/3B3000, and they can read and write SCache and memory by means of 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 32 rows of source matrix data first, then write the 32 rows of data to the target matrix, and continue in sequence until the complete transpose or move of the entire matrix. Matrix processing accelerator can also perform prefetching of data to SCache in this way, as required, by only performing the source matrix without writing the target matrix.

The source matrix involved in the transpose or move may be a small matrix located in a large matrix, so its matrix address may not be completely continuous, and there will be gaps between the addresses of adjacent rows, requiring more programming control interfaces to be implemented. The following table 5-1

To 5-4 illustrates 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 | Source matrix starting address |
| 0 x3ff00608 | Dst\_start\_addr | RW | Target matrix starting address |
| 0 x3ff00610 | Row | RW | The number of elements in a row of the source matrix |
| 0 x3ff00618 | col | RW | The number of elements in a column of the source matrix |
| 0 x3ff00620 | length | RW | Row span (in bytes) of the large matrix where the source matrix is located |
| 0 x3ff00628 | width | RW | Row span (in bytes) of the large matrix where the target matrix is located |
| 0 x3ff00630 | Trans\_ctrl | RW | Transpose control register |
| 0 x3ff00638 | Trans\_status | RO | Transpose status register |

Table 5-2 Matrix processing register address description

|  |  |
| --- | --- |
| address | The name of the |
| 0 x3ff00600 | 0 transposed module s src\_start\_addr |
| 0 x3ff00608 | 0 transposed module dst\_start\_addr |
| 0 x3ff00610 | 0 transposed module row |
| 0 x3ff00618 | col of no. 0 transpose module |
| 0 x3ff00620 | The length of the 0 transpose module |
| 0 x3ff00628 | 0 transpose w DTH of the module |
| 0 x3ff00630 | 0 t ri transposed module trans\_ctrll |
| 0 x3ff00638 | 0 transposed module trans\_status |
| 0 x3ff00700 | Transpose module 1 src\_start\_addr |
| 0 x3ff00708 | Transpose module 1 dst\_start\_addr |
| 0 x3ff00710 | Transpose module 1 src\_row\_stride |
| 0 x3ff00718 | Transpose module 1 src\_last\_row\_addr |
| 0 x3ff00720 | Length of one transpose module |
| 0 x3ff00728 | width of no. 1 transpose module |
| 0 x3ff00730 | Transpose module 1 trans\_ctrl |
| 0 x3ff00738 | Transpose module 1 trans\_status |

Table 5-3 t ri ans\_ct ri l register

|  |  |
| --- | --- |
| field | instructions |
| 0 | Can make a |
| 1 | Is it allowed to write the target matrix? When is 0, the transpose process prefetuses only the source matrix, but does not write the target matrix. |
| 2 | After the source matrix is read, whether the interrupt is valid. |
| 3 | After the target matrix is written, is it effective to interrupt? |
| 7.. 4 | arcmd, read command internal control. When a arcache is 4 'hf, must be set to 4' hc. When a arcache to other meaningless values |
| 11.. 8 | A arcache, the read command internal control bits. When is 4 'HF, use cache path; when is 4' h0, use uncache path. Other values are meaningless. |
| 15.. 12 | Awcmd, write command internal control bit. When awcache is 4 'hf, it must be set to 4' Hb. Awcache is meaningless if it is another value. |
| 19.. 16 | Awcache, write command internal control bit. When is 4 'HF, use cache path; when is 4' h0, use uncache path. 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, transpose for 1; Zero means no transpose |

Table 5-4 t trans\_status register

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

# Interrupt communication between processor cores

Longson 3A3000/3B3000 implements 8 inter-core interrupt registers (IPI) for each processor core to support interrupt and communication between processor cores when the multi-core BIOS is started and the operating system is running. The instructions and addresses are shown in Table 6-1

6-5.

Table 6-1. Registers and their functions related to interrupt between processor cores

Sorry… there are some tables too complex to translate it,please read Chinese loongson book

The list above is a list of intercore interruption-related registers for a single-node multiprocessor system consisting of a single godson 3A300/3B3000 chip. When the multi-chip longshon 3A30000/3B3000 interconnection is used to constitute the multi-node CC-NUMA system, each chip is

The node in the chip corresponds to a system global node number, and the IPI register address of the processor core in the node is fixed offset according to the base address of the node in the table above. For example, node no. 0 processor core has IPI\_Status address 0x3FF01000, node No. 1 processor address 0x10003FF01000, and so on.

## I/O interrupt

The Longson 3A300/3B3000 chip supports up to 32 interrupt sources, which are managed in a unified manner. As shown in Figure 7-1 below, any IO interrupt source can be configured to enable, trigger, and route the target processor core interrupt pin.

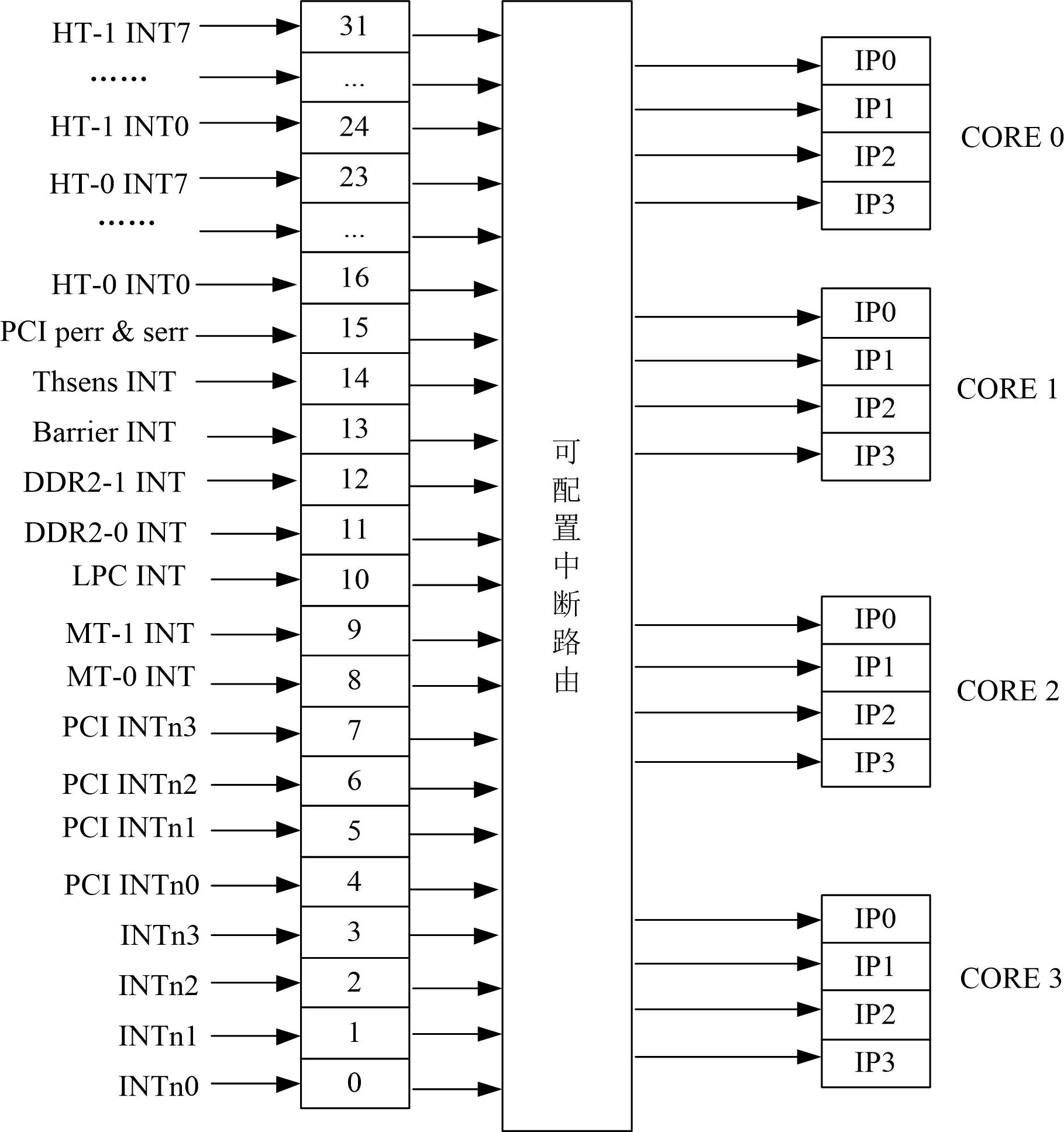


FIG. 7-1 Schematic diagram of interrupt routing for longshon 3A3000/3B3000 processor

Interrupt-related configuration registers control the corresponding broken wires in the form of bits. Interrupt control bit connection and property configuration are shown in Table 7-1 below. The interrupt Enable configuration has three registers: Intenset, Intenclr, and Inten. The Intenset sets the interrupt enabled, and the Intenset register writes 1 to the interrupt enabled. The Intenclr clears the interrupt enable, and the interrupt corresponding to the write 1 bit in the Intenclr register is cleared. The Inten register reads the current state of each interrupt enable

Conditions. Interrupt signals in the form of pulses (such as PCI\_SERR) are selected by Intedge configuration register, with write 1 for pulse trigger and write 0 for level trigger.  The interrupt handler can clear the pulse record with the corresponding bit in Intenclr.

Table 7-1 Interrupt control register

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | Access properties/default values | | | | |
| Intedge | Inten | Intenset | Intencl ri | The interrupt source |
| 3:0 | RW / 0 | R / 0 | W / 0 | W / 0 | Sys\_ nt0-3 |
| 7:4 | RO / 0 | R / 0 | RW / 0 | RW / 0 | PCI\_INTn |
| 8 | RO / 0 | R / 0 | RW / 0 | RW / 0 | Mat ri x\_ nt0 |
| 9 | RO / 1 | R / 0 | RW / 0 | RW / 0 | Mat ri x\_ nt1 |
| 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 | Ba ri ri e ri |
| 14 | RW / 0 | R / 0 | RW / 0 | RW / 0 | Thsens nt |
| 15 | RW / 0 | R / 0 | RW / 0 | RW / 0 | Pc \_pe ri ri |
| 23:16 | RW / 0 | R / 0 | RW / 0 | RW / 0 | HT0 nt0-7 |
| 31:24 | RW / 0 | R / 0 | RW / 0 | RW / 0 | HT1 nt0-7 |

Table 7-2 IO control register address

|  |  |  |
| --- | --- | --- |
| The name of the | Address offset | describe |
| Int s ri | 0 x3ff01420 | 32-bit interrupt status register |
| Inten | 0 x3ff01424 | The 32-bit interrupt enabled status register |
| Intenset | 0 x3ff01428 | The 32-bit setup enable register |
| Intencl ri | 0 x3ff0142c | The 32-bit clear enable register |
| Intedge | 0 x3ff01438 | 32 bit trigger mode register |
| CORE0\_INTISR | 0 x3ff01440 | 32-bit interrupt status routed to CORE0 |
| CORE1\_INTISR | 0 x3ff01448 | 32-bit interrupt status routed to CORE1 |
| CORE2\_INTISR | 0 x3ff01450 | 32-bit interrupt status routed to CORE2 |
| CORE3\_INTISR | 0 x3ff01458 | 32-bit interrupt status routed to CORE3 |

With four processor cores integrated in the Loongson 3A300/3B3000, the 32-bit interrupt source above can be configured to select the desired interrupt target processor core. Further, the interrupt source can optionally route to either of the processor core interrupts INT0 to INT3, IP2 to 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 table 7-3 and 7-4 below. The routing register USES vector routing, such as 0x48, to route to the INT2 of processor 3.

Table 7-3 Description of interrupt routing register

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

Table 7-4 Interrupt routing register address

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name address offset description Name address offset description | | | | | |
| Ent ri y0 | 0 x3ff01400 | Sys\_ nt0 | Ent ri y16 | 0 x3ff01410 | HT0 - nt0 |
| Ent ri y1 | 0 x3ff01401 | Sys\_ nt1 | Ent ri y17 | 0 x3ff01411 | HT0 - nt1 |
| Ent ri y2 | 0 x3ff01402 | Sys\_ nt2 | Ent ri y18 | 0 x3ff01412 | HT0 - nt2 |
| Ent ri y3 | 0 x3ff01403 | Sys\_ nt3 | Ent ri y19 | 0 x3ff01413 | HT0 nt3 - |
| Ent ri y4 | 0 x3ff01404 | Pc \_ nt0 | Ent ri y20 | 0 x3ff01414 | HT0 - nt4 |
| Ent ri y5 | 0 x3ff01405 | Pc \_ nt1 | Ent ri y21 | 0 x3ff01415 | HT0 - nt5 |
| Ent ri y6 | 0 x3ff01406 | Pc \_ nt2 | Ent ri y22 | 0 x3ff01416 | HT0 - nt6 |
| Ent ri y7 | 0 x3ff01407 | Pc \_ nt3 | Ent ri y23 | 0 x3ff01417 | HT0 - nt7 |
| Ent ri y8 | 0 x3ff01408 | Mat ri nt0 x | Ent ri y24 | 0 x3ff01418 | HT1 - nt0 |
| Ent ri y9 | 0 x3ff01409 | Mat ri x nt1 | Ent ri y25 | 0 x3ff01419 | HT1 - nt1 |
| Ent ri y10 | 0 x3ff0140a | Lpc nt | Ent ri y26 | 0 x3ff0141a | HT1 - nt2 |
| Ent ri y11 | 0 x3ff0140b | Mc0 | Ent ri y27 | 0 x3ff0141b | HT1 nt3 - |
| Ent ri y12 | 0 x3ff0140c | Mc1 | Ent ri y28 | 0 x3ff0141c | HT1 - nt4 |
| Ent ri y13 | 0 x3ff0140d | Ba ri ri e ri | Ent ri y29 | 0 x3ff0141d | HT1 - nt5 |
| Ent ri y14 | 0 x3ff0140e | Thsens nt | Ent ri y30 | 0 x3ff0141e | HT1 - nt6 |
| Ent ri y15 | 0 x3ff0140f | Pc \_pe ri ri / se ri ri | Ent ri y31 | 0 x3ff0141f | HT1 - nt7 |

## Temperature sensor

#### Real-time temperature sampling

The longson 3A300/3B3000 is internally integrated with two temperature sensors, which can be observed through the sampling register at 0x1FE00198. At the same time, it can be controlled by using flexible high-low temperature interrupt alarm or automatic frequency modulation function. The corresponding bits of the temperature sensor in the sampling register are as follows (base address 0x1FE00198) :

Table 8-1 Description of temperature sampling registers

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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℃) |
| 47:32 | | Thsens0\_out | R |  | Temperature sensor 0 ℃  Node temperature =Thens0\_out  \* 731/0 x4000-273  Temperature range -40-125 degrees |
| 65:48 | | Thsens1\_out | R |  | Temperature sensor 1 ℃  Node temperature =Thens1\_out  - \* 731/0 x4000-273  Temperature range -40-125 degrees |

By setting the control register, the function of interruption above the preset temperature, interruption below the preset temperature and automatic frequency reduction at high temperature can be realized.

#### High and low temperature interrupt triggers

For high and low temperature interrupt alarm function, there are 4 sets of control registers to set its threshold. Each set of registers contains the following three control bits:

GATE: Sets the threshold for high or low temperature. When the input temperature is higher than the high temperature threshold or lower than the low temperature threshold, an interruption will occur.

EN: Interrupt enable control. The set of registers is only effective after setting 1.

SEL: Input temperature selection. The current 3A300/3B3000 has two temperature sensors integrated into it, and this register is used to configure which sensor's temperature to select as input. You could use a 0 or a 1.

The high temperature interrupt control register contains 4 sets of setting bits used to control the high temperature interrupt trigger. Low temperature interrupt control register

The device contains four sets of Settings for controlling the low temperature interrupt trigger. There is also a set of registers for displaying interrupt status

Any write to the register will clear the interrupt state. The specific description of these registers is as follows:

Table 8-2 Description of high and low temperature interrupt registers

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

#### High temperature automatic frequency reduction setting

In order to ensure the operation of the chip in the high temperature environment, the high temperature can be set to automatically reduce the frequency, so that the chip actively performs clock frequency division when the preset range is exceeded, so as to reduce the chip turnover rate.

There are four sets of control registers to set the behavior for the high temperature frequency reduction function. Each set of registers contains the following four control bits:

GATE: Sets the threshold for high or low temperature. When the input temperature is higher than the high temperature threshold or lower than the low temperature threshold, the frequency division operation will be triggered.

EN: Interrupt enable control. The set of registers is only effective after setting 1.

SEL: Input temperature selection. The current 3A300/3B3000 has two temperature sensors integrated into it, and this register is used to configure which sensor's temperature to select as input. You could use a 0 or a 1.

## Ddr2/3 SDRAM controller configuration

The internal integrated memory controller of the Godson 3 processor is designed in accordance with the INDUSTRY standard of DDR2/3 SDRAM (JESD79-2 and JESD79-3). In the Godson 3 processor, all memory read/write operations implemented comply with the provisions of JESD79-2B and JESD79-3.

#### Ddr2/3 SDRAM controller features overview

The Loongson no. 3 processor supports up to four CS (realized by four DDR2 SDRAM chip selection signals, namely two double-sided memory strips) and a total of 19 bit address bus (i.e., 16 bit column address bus and 3 bit logical Bank bus).

The parameter setting of DDR2/3 controller can be adjusted to support the specific choice of using different memory chip types for longson no. 3 processor. Where, the supported maximum block selection (CS\_n) number is 4, the row address (RAS\_n) number is 16, the column address (CAS\_n) number is 15, and the logical body selection (BANK\_n) number is 3.

The physical address of the memory request sent by the CPU can be mapped to many different addresses, depending on the configuration within the controller

Shoot.

The integrated memory control circuit of the Loongson 3 processor only accepts 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 the Godson no. 3 processor has the following characteristics:

* + - Interface command, read and write data full flow operation
    - Memory commands merge and sort to improve overall bandwidth
    - Configure register read/write ports to modify the basic parameters of memory devices
    - Built-in dynamic delay compensation circuit (DCC) for reliable sending and receiving of data
    - ECC can detect 1 - and 2-bit errors on 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 diagram, 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.

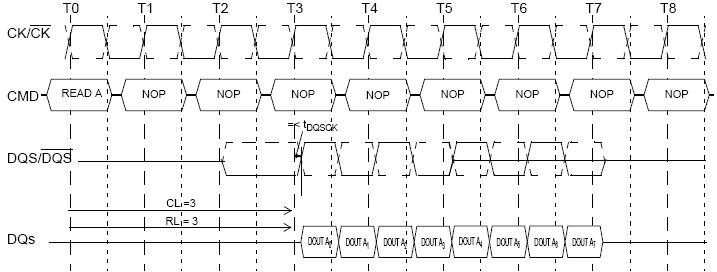


Figure 9-1 DDR2 SDRAM read operation protocol

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

## Ddr2/3 SDRAM write operation protocol

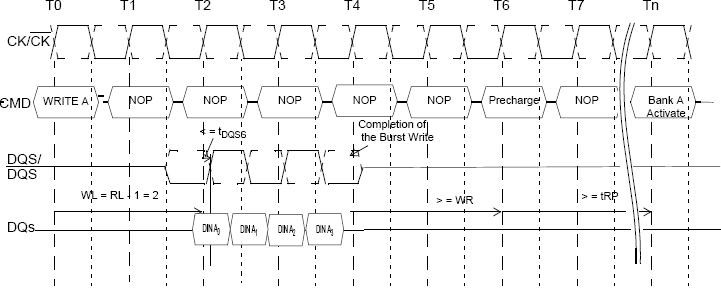
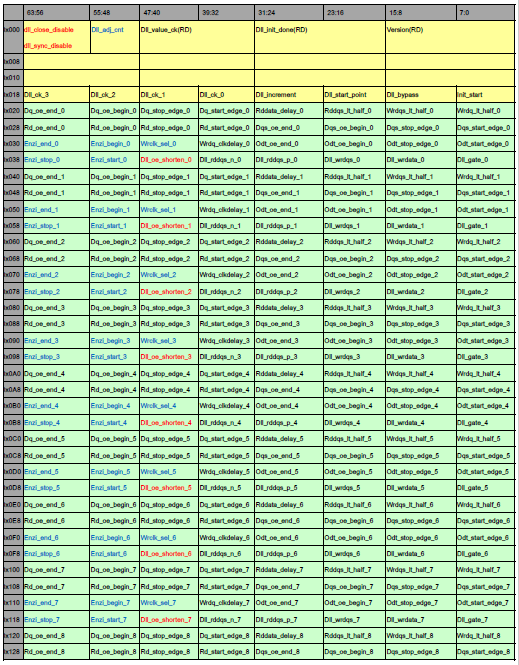
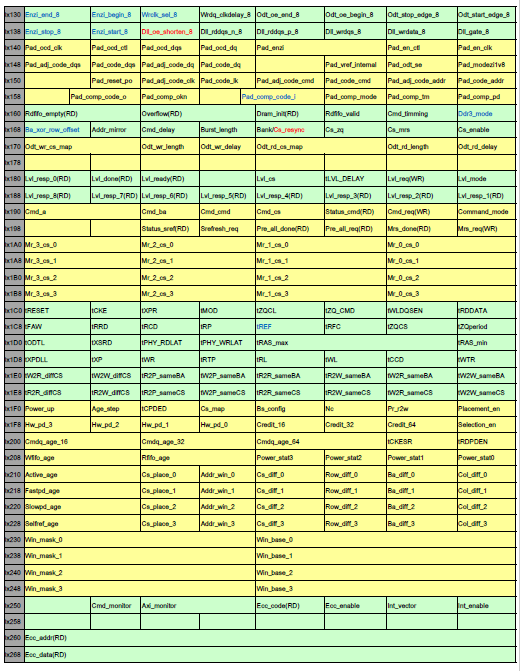
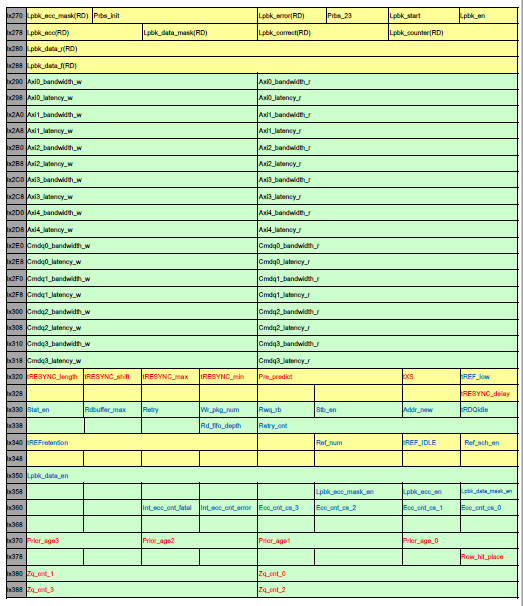
 The protocol for DDR2/3 SDRAM write operations is shown in Figure 11-3. In the figure, the command CMD is composed of RAS\_n, CAS\_n, and WE\_n. For write operations, RAS\_n=1, CAS\_n = 0, WE\_n = 0. Also, unlike a read operation, a write operation requires DQM to identify the mask of the write operation, that is, the number of bytes to write. DQM is synchronized with DQs signals 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 memory controller software are shown in the following table:   

#### Software Programming Guide

#### Initialization operation

Initialization begins when the software writes 1 to register Init\_start (0x018). All other registers must be set to the correct value before the Init\_start signal is set.

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

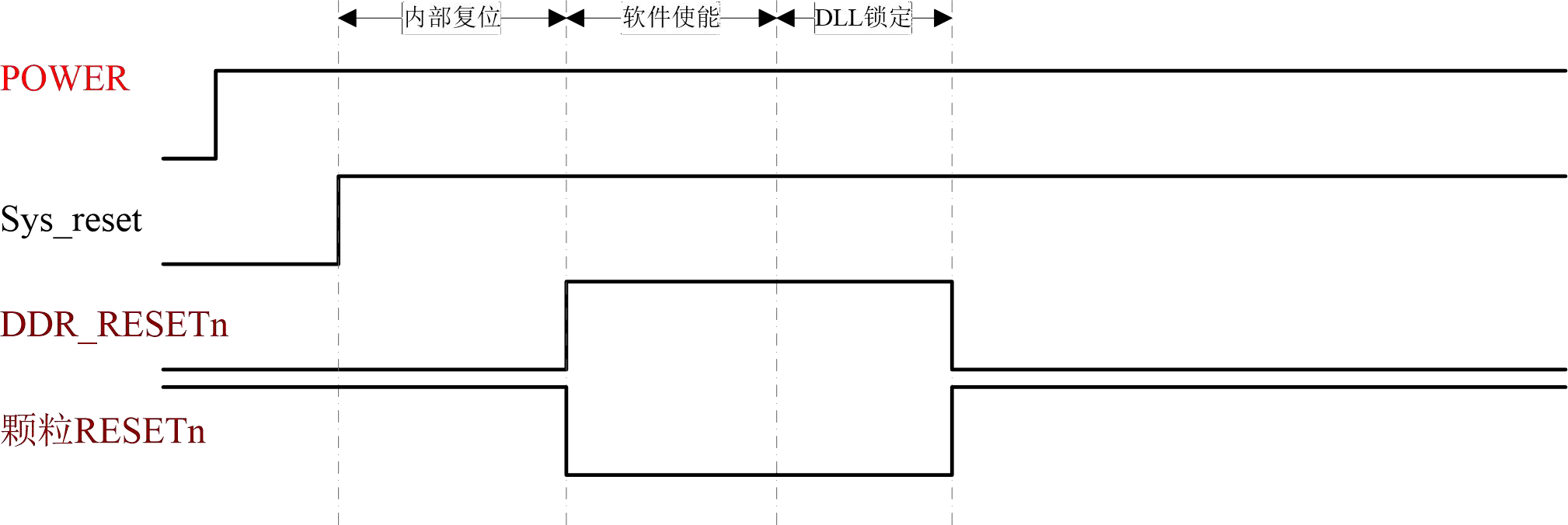
* + - 1. The software writes the correct configuration values to all registers, but Init\_start (0x018) must remain at 0 during this process;
      2. The software sets Init\_start (0x018) to 1, which causes the start of hardware initialization;
      3. Initialization begins internally in the PHY and the DLL will attempt to do a lock operation. If the lock is successful, the corresponding state can be read from Dll\_init\_done (0x000) and the current number of lock delay lines can be read and written from Dll\_value\_ck (0x000); If the lock is unsuccessful, initialization will not continue (at this point you can continue by setting Dll\_bypass (0x018));
      4. After DLL lock (or bypass setting), the controller will issue corresponding initialization sequences to DRAM according to the initialization requirements of corresponding DRAM, such as the corresponding MRS command, ZQCL command, etc.
      5. The Dram\_init (0x160) register is sampled to determine if the memory initialization operation has completed.

#### Control of the reset pin

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

* + - 1. In general mode, reset\_ctrl[1:0] == 2 'b00. In this mode, the behavior of the reset signal pin 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 not powered on: pin state is low;
* When power on: pin state is low;
* When the controller is initialized, the pin state is high;
* In normal operation, the pin state is high. The timing sequence is shown in the figure below:

In reverse mode, reset\_ctrl[1:0] == 2 'b10. In this mode, the effective level of the reset signal pin is reversed when the actual memory control is carried out. So DDR\_RESETn needs to be connected to the corresponding pin on the memory slot via the reverter on the motherboard. The behavior of the pin is:

* When not powered on: pin state is low;
* When power on: pin state is low;
* When the controller is configured: the pin state is high;
* When the controller is initialized: the pin state is low;
*  Normal operation: pin condition is low. The timing sequence is shown in the figure below:

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 via the reverter on the motherboard. The behavior of the pin is:

* It's always low; The timing sequence is shown in the figure below:

Combined with the latter two reset modes, STR control can be implemented directly using the reset signal of the memory controller. When the entire system is started from the closed state, use the method in (2) to reset normally and start working with the memory stick. When the system recovers from STR, use the method in (3) to reconfigure the memory bar so that it can start working properly again without breaking the original state of the memory bar.

#### Leveling

The operation came with Leveling in DDR3, which was used to intelligatively configure the phase relationship between various signals during the read and write operation of the memory controller. Usually it includes the Write Leveling, Read Leveling and Gate Leveling. In this controller, only the Write Leveling and Gate Leveling were made, and the Read Leveling was not made. The software needed to make the function of the Read Leveling come to pass by judging the correctness of the Read Leveling. In addition to the DQS phase and GATE phase that were handled during the Leveling process, the configuration methods of writing DQ phase and reading DQ phase could also be calculated according to the final phase Leveling.

#### The Write Leveling

* + - * 1. To configure the phase relationship between Write DQS and the clock, the software program needs to refer to the following steps.
        2. Complete the controller initialization, see the previous section;
        3. Put Dll\_wrdqs\_x (x = 0... 8) Set to 0;
        4. Set Lvl\_mode (0x180) to 2 'b01;
        5. The Lvl\_ready (0x180) register, if it is 1, means that the Write Leveling request can be started.
        6. Set Lvl\_req (0x180) to 1;
        7. The Lvl\_done (0x180) register, if it is 1, represents the completion of the Write Leveling request.
        8. Sample Lvl\_resp\_x (0x180, 0x188) register, if 0, increase the corresponding Dll\_wrdqs\_x[6:0] by 1, and repeat 5-7; If it is 1, the Write Leveling operation has been successful.
        9. The value of Dll\_wrdqs\_x should be the correct setting.
        10. The Write Leveling operation ends. If, during this process, Lvl\_resp\_x is 1 on the first sampling, the result is problematic and you should check for incorrect Settings in other registers, which might include Wrdqs\_lt\_half, Dqs\_start\_edge, Dqs\_stop\_edge, Dqs\_oe\_begin, Dqs\_oe\_end.
        11. Then set Wrdqs\_lt\_half\_x according to whether the value of Dll\_wrdqs\_x is less than 0x40;
        12. Set Dll\_wrdata\_x 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 according to whether the value of Dll\_wrdata\_x is less than 0x40;
        2. Determine if different Dll\_wrdata\_x values are around 0x40 and cross the boundary of 0x40 (meaning 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 for the Wrdata\_lt\_half\_x == 0 data set to 1. Then subtract 1 from the values of tPHY\_WRDATA and tRDDATA.
        3. Set Lvl\_mode (0x180) to 2 'b00 and quit the Write Leveling mode.

#### Gate Leveling

* + - * 1. The software programming was made with reference to the following steps.
        2. Complete the controller initialization, see the previous section;
        3. Complete the Write Leveling, see the previous section;
        4. Will Dll\_gate\_x (x = 0... 8) Set to 0;
        5. Set Lvl\_mode (0x180) to 2 'b10;
        6. The Lvl\_ready (0x180) register, if it is 1, means that the Gate Leveling request can be started.
        7. Set Lvl\_req (0x180) to 1;
        8. The Lvl\_done (0x180) register, if it is 1, represents the completion of a Gate Leveling request.
        9. Sample Lvl\_resp\_x[0] (0x180, 0x188) registers. If the first sampling finds Lvl\_resp\_x[0] to be 1, 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.
        10. If the sample result is 0, increment the corresponding Dll\_gate\_x[6:0] by 1 and repeat 6-9; If it is 1, then the Gate Leveling operation has been successful.
        11. At the end of the Gate Leveling operation, the sum of Dll\_gate\_x[6:0] and Dll\_wrdata\_x[6:0] was actually the phase relationship between DQS and the INTERNAL clock of a PHY. The parameters were adjusted according to the Leveling result.
        12. If the sum of Dll\_gate\_x[6:0] and Dll\_wrdata\_x[6:0] is less than 0x20 or greater than 0x60, then Dll\_rddqs\_lt\_halt is set to 1. Because RDDQS phase relationship is actually equal to the input read DQS on the basis of the delay of another quarter.
        13. If the value of Dll\_gate\_x is greater than 0x40, subtract 0x40 from the value of Dll\_gate\_x; Otherwise, just set it to 0.
        14. After adjusting, do two more Lvl\_req operations. Watch for changes in the values of Lvl\_resp\_x[7:5] and Lvl\_resp\_x[4:2]. If it's not 4, you might need to add or subtract one to Rd\_oe\_begin\_x, and if it's greater than Burst\_length/2, you'll probably need to do some fine-tuning of the value of Dll\_gate\_x
        15. Set Lvl\_mode (0x180) to 2 'b00 and quit the Gate Leveling mode.

#### Initiate MRS command alone

The MRS commands issued by the memory controller to the memory are: MR2\_CS0, MR2\_CS1, MR2\_CS2, MR2\_CS3, MR3\_CS1, MR3\_CS2, MR3\_CS3, MR1\_CS0, MR1\_CS2, MR1\_CS3, MR0\_CS0, MR1\_CS1, MR1\_CS2, MR1\_CS2, MR1\_CS3.

Where, the validity of MRS command corresponding to CS is determined by Cs\_mrs. Only when the selected bit of Cs\_mrs corresponding to each slice is valid, the MRS command will be issued to DRAM. The value of each corresponding MR is determined by the register MR \*\_cs\*. These values are also used for the MRS command when initializing memory.

Specific operations are as follows:

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

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

* + - 1. Set Command\_mode (0x190) to 0 to cause the controller to exit command send mode.

#### Any operation controls the bus

The memory controller can issue any combination of commands to the DRAM through command sending mode, and the software can set Cmd\_cs, Cmd\_cmd, Cmd\_ba, Cmd\_a (0x168) to issue to the DRAM in command sending mode.

Specific operations are 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 make the controller enter command send mode;
      3. Sample Status\_cmd (0x190). If it is 1, the controller has entered the command sending mode and can proceed to the next step. If it is 0, it needs to continue to wait.
      4. Write Cmd\_req (0x190) as 1 and send the command to DRAM;
      5. Set Command\_mode (0x190) to 0 to cause the controller to exit command send mode.

#### Self-circulating 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 reuse of these two sets of interfaces is controlled by port test\_PHY. When test\_PHY is effective, the controller's test\_\* port is used for control. At this time, the self-test is completely controlled by hardware. When test\_PHY does not work, the pm\_\* parameter of the software program is used for control. The specific signal meaning of using the test port can be referred to in the register parameters with the same name.

These two sets of interfaces are basically the same in terms of control parameters, only different access points. This paper introduces the control method of software programming. Specific operations are 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. Sampling register Dll\_init\_done (0x000). If this value is 1, the DLL is locked, and you can proceed to the next step. If the value is 0, you need to wait; When using the test port for control, because you can't see the output of the register, you don't need to sample the register

Wait for a certain amount of time here to ensure DLL locking is complete before proceeding to the next step);

* + - 1. Set register Lpbk\_start (0x270) to 1; At this point, loop testing begins.

Since the cycle test has started, the software needs to frequently detect whether there is an error. The specific operation is as follows:

* + - 1. Sample register Lpbk\_error (0x270). If the value is 1, it indicates that an error has occurred. At this time, the error data and correct data of the first error can be observed by using registers (0x270, 0x278, 0x280, 0x288) through Lpbk\_\* and other observations. If this value is 0, no data error has occurred.

#### ECC function usage control

ECC functionality is available only in 64-bit mode. Ecc\_enable includes the following four control bits:

Ecc\_enable[0] controls whether to enable the ECC function, which will only be enabled if the effective bit is set. Ecc\_enable[1] controls whether an error is reported through the read response path inside the processor to enable the ECC two digits to occur

A wrong read access can cause an immediate exception to the processor core.

Ecc\_enable[2] controls whether error is reported through the write response path within the processor so that the occurrence of ECC two-digit write access (read and write) can immediately cause processor core exceptions.

Ecc\_enable[3] controls the trigger time of recording error information in the register. These error messages will not be triggered continuously without software processing, but will only record the information of the first error. This information includes Ecc\_code, Ecc\_addr, and Ecc\_data. When Ecc\_enable[3] is 0, as long as there is an ECC error (including 1 and 2 bits), the record will be triggered; when Ecc\_enable[3] is 1, the record will only be triggered if there is an ECC two-bit error. This "first time" means that the corresponding bit of the interrupt vector register is set. In other words, it records the visit that caused the interrupt to occur.

In addition, AN ECC error can notify the processor core by means of an interrupt. This interrupt is controlled by Int\_enable. The interrupt consists of two vectors. Int\_vector[0] indicates an ECC error (including 1 and 2 bits), and Int\_vecotr[1] indicates an ECC two-bit error. The purge of Int\_vector is achieved by writing 1 to the corresponding bit.

## HyperTransport controller

In longson 3A300/3B3000, HyperTransport bus is used to realize 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 the HyperTransport master line is used for multi-core interlinking, the HT0 controller (with initial address 0x0C00\_0000\_0000 -- 0x0DFFFF\_FFFF) can be configured to support inter-chip Cache consistency, while the HT1 controller (with initial address 0x0E00\_0000\_0000 -- 0x0FFF\_FFFF\_FFFF) can be configured to support inter-chip Cache consistency.See Section 10.7 for details.

The HyperTransport controller supports a maximum two-way 16 bit width and 2.0GHz operation frequency. After the connection is automatically initialized by the system, the user program can modify the corresponding configuration registers in the protocol to change the width and running frequency and reinitialize them, as detailed in Section 10.1.

The main features of the Longcore 3A300/3B3000 HyperTransport controller are as follows:

* Support HT1.0/HT3.0 protocol
* Support 200/400/800/1600/2000mhz operating frequency
* HT1.0 supports 8-bit 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 can be configured
* It can be configured in Cache consistency mode for multi-chip interconnections

#### HyperTransport hardware setup and initialization

HyperTransport bus is composed of transmission signal bus and control signal pin, etc. The following table gives the pins related to HyperTransport bus and its function description.

Table 10-1 Hype ri T ri ansp yao ri T bus signal correlation pin

|  |  |  |
| --- | --- | --- |
| **pin** | **The name of the** | **describe** |
| HT0\_8x2 | Bus width configuration | 1. Configure the 16-bit HyperTransport bus as two independent 8-bit buses. |
|  |  | Respectively controlled by two independent controllers, 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, namely address[40]  = 0; All HT0\_Hi signals 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, the 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 package 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 does not hit the receive window of the controller, it will be sent back to the bus as a P2P request; if this register is 1 and it does not hit, it will be responded as an error request.  0: Set HT0\_Lo to slave device mode, in which bus control signals such as HT0\_Lo\_Powerok, HT0\_Lo\_Rstn, HT0\_Lo\_Ldt\_Stopn are driven by the other device. In this mode, these control signals are driven by the other device, if not properly driven, then HT bus  Not working correctly. |
| HT0\_Lo\_Powerok | Bus Powerok | When 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 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 bus Ldt\_Stopn signal,  When HT0\_Lo\_Mode is 1, it is controlled by HT0\_Lo.  When HT0\_Lo\_Mode is 0, it is controlled by the other device. |
| HT0\_Lo\_Ldt\_Reqn | Bus Ldt\_Reqn | HyperTransport bus Ldt\_Reqn signal, |
| HT0\_Hi\_mode | Master device mode | 1: Set HT0\_Hi as the main device mode. In this mode, bus control signals are driven by HT0\_Hi, including HT0\_Hi\_Powerok, HT0\_Hi\_Rstn, HT0\_Hi\_Ldt\_Stopn. In this mode, the 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 package 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 does not hit the receive window of the controller, it will be sent back to the bus as a P2P request; if this register is 1 and it does not hit, it will be responded as an error request.  0: Set HT0\_Hi to slave mode, bus control signal, 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 the other device, if not driven correctly, the HT bus will not work correctly. |
| HT0\_Hi\_Powerok | Bus Powerok | When 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, the high 8-bit bus is controlled.  If HT0\_8x2 is 0, it is invalid. |
| HT0\_Hi\_Rstn | Bus Rstn | When 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, the high 8-bit bus is controlled. If HT0\_8x2 is 0, it is invalid. |
| HT0\_Hi\_Ldt\_Stopn | Bus Ldt\_Stopn | When 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, the high 8-bit bus is controlled. If HT0\_8x2 is 0, it is invalid. |
| HT0\_Hi\_Ldt\_Reqn | Bus Ldt\_Reqn | HyperTransport bus Ldt\_Reqn signal,  When HT0\_8x2 is 1, the high 8-bit bus is controlled.  If HT0\_8x2 is 0, it is invalid. |
| 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  CTL[0] is controlled by HT0\_Lo  When HT0\_8x2 is 0, CTL[1] is invalid  CTL[0] is controlled by HT0\_Lo |
| HT0\_Rx\_CADp HT0\_Rx\_CADn [15:0] [15:0] HT0\_Tx\_CADp HT0\_Tx\_CADn [15:0] [15:0] | CAD [15:0] | HyperTransport Bus CAD signals  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. After cold start, the HyperTransport bus will automatically work at the lowest frequency (200MHz) and the minimum width (8BIT), and try to do the bus initialization handshake. Whether the initialization is Complete can be read out by the register "Init Complete" (see section 10.5.2). After initialization, the bus Width can be read Out from registers "Link Width Out" and "Link Width In" (see section 10.5.2). After initialization, the user can rewrite registers "Link Width Out", "Link Width In" and "Link Freq", and configure corresponding registers of the other device. After configuration, the user needs to hot-reset the bus or pass through

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

#### HyperTransport protocol support

The HyperTransport bus of the Loongson 3A300/3B3000 supports most of the commands in the 1.03/3.0 protocol, and some extension instructions have been added to the extended Consistency protocol that supports multi-chip interconnection. In both modes, the commands that the HyperTransport receiver can receive are shown in the following table. It is important to note that atomic operation commands are not supported for the HyperTransport bus.

Table 10-2 Hype ri T ri ansp yao ri T the receiver can receive commands

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **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 | Guaranteed order relation |  |
| 111111 | - | The Sync/Error | The Sync/Error |  |

For the sender, the commands that are sent out in both modes are shown in the table below.

Table 10-3 Commands that will be sent out in two modes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **coding** | **channel** | **The command** | **The standard model** | **Extension (consistency)** |
| 000000 | - | The NOP | Empty packet or flow control |  |
| x01x0x | NPC  The or | The Write | Bit 5:0 - Nonposted  1 - the Posted | Bit 5: Must be 1, POSTED |
|  | The PC |  | Bit 2:0 - Byte  1 - Doubleword  Bit 0: Must be 0 | Bit 2:0 - Byte  1 - Doubleword  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 that can support types of interrupts like Fix, Arbiter, etc., but has no support for hardware automatic EOI. For the above two types of interrupts, the controller will automatically write to the interrupt register after receiving, and the system interrupt controller will be informed of the interrupt according to the setting of the interrupt mask register. For 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 this type of interrupt handling.

A typical PIC interrupt is accomplished by the following steps: (1) PIC controller sends PIC interrupt request to the system; The system sends interrupt vector query to PIC controller; PIC controller sends interrupt vector number to the system; The system clears the corresponding interrupt on PIC controller. PIC controller will issue the next interrupt to the system only after the above 4 steps are completed. For the Longson 3A300/3B3000 HyperTransport controller, the first three steps will be automatically processed, and PIC interrupt vector will be written into the corresponding position in 256 interrupt vectors. After the software system has processed the interrupt, it needs to carry out the fourth step, that is, send the clear interrupt to PIC controller. Then the processing of the next interrupt begins.

#### HyperTransport address window

* + 1. **HyperTransport space**

The godson 3 a3000/3 b3000 processors, the default four HyperTransport distribution is as follows: the address of the interface window table 10-4 default 4 Hype ri T ri ansp yao ri T distribution in the address of the interface 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 |
| 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 also access the address space by configuring the address window on the cross-switch (see section 2.5 for details). The distribution of address Windows in the internal 40-bit address space of each HyperTransport interface is shown in the table below.

Table 10-5 the godson 3 processor Hype ri T ri ansp yao ri T the address window distribution within the interface

|  |  |  |  |
| --- | --- | --- | --- |
| **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 3A300/3B3000 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 the godson 3 a3000/3 b3000 processor Hype ri T ri ansp yao ri T provide the address of the window in the interface

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **The address window** | **Window number** | **Accept the bus** | **role** | **note** |
| Receiving window  See window configuration  10.5.7) | 3 | HyperTransport | Decide whether to accept an access issued on a HyperTransport bus. | When in master bridge mode (that is, act\_as\_slave in the configuration register is 0), only those accesses falling into these address Windows will be responded to by the internal bus, and other accesses will be considered as P2P accesses re-sent back to the HyperTransport bus; When in device mode (that is, act\_AS\_slave is 1 in the configuration register), only accesses falling into these address Windows are received and processed by the internal bus, and other accesses are given error returns according to the protocol. |
| The Post window  See window configuration  10.5.11) | 2 | Inside the bus | Determines whether Write access from the internal bus to the HyperTransport bus should be treated as Post Write | Outgoing calls that fall into these address Spaces will be treated as Post writes.  Post Write: In the HyperTransport protocol, this Write access does not need to wait for the Write response to be completed, that is, issued to the bus by the controller |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  | This write access is followed by a write access to the processor to complete the response. |
| Prefetch window is available  See window configuration  10.5.12) | 2 | Inside the bus | Determines whether to receive internal Cache access fetcher. | When the processor core is executing out of order, some guess read or point access is made to the bus, which is wrong for some IO Spaces. By default, this  , access to the HT controller is returned 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 | Decide whether to treat an access on a HyperTransport bus as an Uncache access to an inner part | IO DMA access within the Loongson 3A300/3B3000 processor, in case of Cache access, is determined as a hit via SCache to maintain IO consistency information. Through the configuration of these Windows, access hit in these Windows can be accessed directly to memory 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 configuration register access request arriving from AXI SLAVE or HT RECEIVER, carry out external interrupt processing, and save a large number of configuration locators visible to the software for controlling various working modes of the system.

Firstly, the access and storage of configuration registers used to control various behaviors of HT controller are in this module. The access offset address of this module is 0xFD\_FB00\_0000 to 0xFD\_FBFF\_FFFF at the HT controller side. All visible registers of 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](#_bookmark77) | Bus Reset Control |
| 0 x40 | [Capability Registers](#_bookmark77) | 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](#_bookmark82) | MISC |
| 0 x54 | [Receiving diagnostic register](#_bookmark85) | Used to diagnose the signal sampled at the receiving end |
| 0 x58 | [Interrupt routing mode selects registers](#_bookmark85) | Correspond to three interrupt routing modes |
| 0 x5c | [Receive cache register](#_bookmark85) |  |
| 0 x60 | [The receive address window configures registers](#_bookmark89) | 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 enabling (external access) |
| 0 x6c | HT Bus Receiving Address Window 1 Base Address (external access) |
| 0 x70 | HT Bus Receive Address Window 2 enabling (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 Receiving Address Window 3 Base Address (external access) |
| 0 x150 | HT Bus Receive Address Window 4 enable (external access) |
| 0 x154 | HT Bus Receiving Address Window 4 Base Address (external access) |
| 0 x80 | [Interrupt vector register](#_bookmark93) | 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](#_bookmark97) | HT Bus interrupt enable register [31:0] |
| 0 xa4 | HT bus interrupt enable register [63:32] |
| 0 xa8 | HT bus interrupt enable register [95:64] |
| 0 xac | HT bus interrupt enable register [127:96] |
| 0 xb0 | HT bus Interrupt enable register [159:128] |
| 0 xb4 | HT bus interrupt enable register [191:160] |
| 0 xb8 | HT bus interrupt enable register [223:192] |
| 0 XBC | HT bus interrupt enable register [255:224] |
| 0 xc0 | [Interrupt Discovery & Configuration](#_bookmark101) | Interrupt Capability |
| 0 xc4 | DataPort |
| 0 xc8 | IntrInfo [31:0] |
| 0 XCC | IntrInfo [63:32] |
| 0 xd0 | [The POST address window configures registers](#_bookmark105) | HT bus POST Address window 0 enabled (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 | [The prefetch address window configures registers](#_bookmark107) | HT Bus Prefetchable Address window 0 enable (internal access) |
| 0 xe4 | HT Bus preaddressable window 0 base address (internal access) |
| 0 xe8 | HT Bus Preaddressable window 1 enabling (internal access) |
| 0 xec | Ht Bus Prefetchable Address Window 1 Base address (internal access) |
| 0 xf0 | [The Uncache Address window configures registers](#_bookmark110) | HT Bus Uncache Address Window 0 enabled (external access) |
| 0 xf4 | HT Bus Uncache Address Window 0 Base address (external access) |
| 0 xf8 | HT Bus Uncache Address Window 1 enabling (external access) |
| 0 XFC | HT Bus Uncache Address Window 1 Base address (external access) |
| 0 x168 | HT Bus Uncache Address Window 2 enabling (external access) |
| 0 x16c | HT Bus Uncache Address Window 2 Base Address (external access) |
| 0 x170 | HT Bus Uncache Address Window 3 enabling (external access) |
| 0 x174 | HT Bus Uncache Address Window 3 Base Address (external access) |
| 0 x158 | [The P2P address window configures registers](#_bookmark114) | HT bus P2P address window 0 enabled (external access) |
| 0 x15c | HT Bus P2P Address Window 0 Base address (external access) |
| 0 x160 | HT bus P2P address window 1 enabling (external access) |
| 0 x164 | HT Bus P2P Address Window 1 Base Address (external access) |
| 0 x100 | [Sender cache size register](#_bookmark118) | The sender command cache size register |

|  |  |  |
| --- | --- | --- |
| 0 x104 |  | Sender data cache size register |
| 0 x108 | [The sender caches the debug register](#_bookmark118) | Used to manually set the size of the sender cache (for debugging) |
| 0 x10c | [PHY impedance matching configuration registers](#_bookmark120) | Used to configure THE PHY sender and receiver impedance matching configuration |
| 0 x110 | [Revision ID register](#_bookmark122) | Used to configure the controller version |
| 0 x118 | [Error Retry controls the register](#_bookmark122) | Retry Count Rollover,Short Retry Attempts |
| 0 x11c | [The Retry Count register](#_bookmark126) | Used for error retransmission counting in HyerTransport 3.0 mode |
| 0 x130 | [Link Train register](#_bookmark126) | HyperTransport 3.0 Link initialization and Link training control |
| 0 x134 | [Training 0 timeout short count register](#_bookmark129) | Used for Training 0 short timed timeout threshold configuration |
| 0 x138 | [Training 0 timeout long count register](#_bookmark131) | Used for Training 0 long count timeout threshold configuration |
| 0 x13c | [Training 1 counting register](#_bookmark131) | Used for Training 1 counting threshold configuration |
| 0 x140 | [Training 2 counting register](#_bookmark131) | Used for Training 2 counting threshold configuration |
| 0 x144 | [Training 3 counting register](#_bookmark131) | Used for Training 3 counting threshold configuration |
| 0 x178 | [Software frequency configuration registers](#_bookmark134) | Realize the frequency switch of the controller in the working process |
| 0 x17c | [PHY configuration register](#_bookmark137) | Used to configure phY-related physical parameters |
| 0 x180 | [Link initializes the debug register](#_bookmark139) | Used to ignore PHY CDR Lock signals and customize wait times |
| 0 x184 | [LDT debug register](#_bookmark139) | Used to configure the time when the LDT signal is invalid to the start of link initialization |

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

### Bridge Control

Offset: 0x3C

Reset value: 0x00000000

Name: Bus Reset Control

Table 10-8 Bus Reset C yao nt ri yao l register definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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 to 9 C yao mmand, Capab l t es P yao nte ri, Capab l ty ID register definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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 the 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 ids used  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 | Bridge Control reset adopts hot 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 L nk C yao nf g, L nk C yao nt ri yao L register definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| 31 | ht\_phase\_select  \_disable | 1 | 0 x0 |  | Phase selection enablement  0: Enable phase selection  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, and the value written to this register will take effect after the next hot copy 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, and the value written to this register will take effect after the next hot copy or HT Disconnect |
| 23 | Dw Fc out | 1 | 0 x0 | R | The sender does not support two-word streaming |
| Lift up | Max Link Width out | 3 | 0 x1 | R | Maximum width of HT bus sender: 16bits |
| 19 | Dw Fc In | 1 | 0 x0 | R | Two-word streaming is not supported on the receiving end |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| thou | Max Link Width In | 3 | 0 x1 | R | Maximum width of HT bus receiver: 16bits |
| The lowest | Reserved | 2 | 0 x0 |  | reserve |
| 13 | LDTSTOP#  Tristate Enable | 1 | 0 x1 | R/W | When the HT bus enters THE HT Disconnect state, does it close the HT PHY  1: closed  0: Not closed |
| 12:10 | Reserved | 3 | 0 x0 |  | reserve |
| 9 | CRC Error (hi) | 1 | 0 x0 | R/W | CRC errors occurred in high 8 bits |
| 8 | CRC Error (lo) | 1 | 0 x0 | R/W | CRC errors occurred in low 8 bits |
| 7 | Trans off | 1 | 0 x0 | R/W | HT PHY shutdown control  When in 16-bit bus mode  1: Closed HIGH/low 8-bit HT PHY  0: The lower 8-bit HT PHY and the higher 8-bit HT PHY are controlled by BIT 0 |
| 6 | The End of the Chain | 0 | 0 x0 | R | HT bus terminal |
| 5 | Init Complete | 1 | 0 x0 | R | HT bus initialization is complete |
| 4 | The Link Fail | 1 | 0 x0 | R | Indicates connection failure |
| 3:2 | Reserved | 2 | 0 x0 |  | reserve |
| 1 | CRC Flood Enable | 1 | 0 x0 | R/W | When CRC error occurs, whether flood HT bus |
| 0 | Trans off (hi) | 1 | 0 x0 | R/W | When the 16-bit HT bus is used to run the 8-bit protocol, the high-8-bit PHY is switched off  1: High 8-bit HT PHY was closed  0: Elevating 8-bit HT PHY |

Offset: 0x48

Reset value: 0x80250023

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

Table 10-11 Rev s yao n ID, nk F ri eq, L L nk E ri ri yao ri, L nk F ri eq 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 produces different values depending on the setting of the external PLL |
| The lowest | Reserved | 2 | 0 x0 |  | reserve |
| 13 | Over Flow Error | 1 | 0 x0 | R | HT bus package 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  Writing the value of this register takes 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 Featu ri Capab e l ty register definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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 | Allow the bus to enter THE LDT DISCONNECT mode  The correct way to do it is: 0 minus > 1 |
| 29 | Ldt the Req Gen | 1 | 0 x0 | R/W | Wake HT bus from LDT DISCONNECT, set  LDT\_REQ\_n  The right way to do it is to put 0 first and then 1:0 -> 1  In addition, a direct read/write request 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, and this register represents the interrupt direction  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 | Is it processor consistent  This is determined by the pin ICCC\_EN |
| 21 | Not Care Seqid | 1 | 0 x0 | R/W | I don't care about the HT order relationship |
| 20 | The Not Axi2Seqid | 1 | 0 x1 | R | Are commands on the Axi bus converted to different SEqids? If not, all read and write commands will take the Fixed ID number in the Fixed SeqID  1: Do not convert  0: conversion |
| He hath | Fixed Seqid | 4 | 0 x0 | R/W | When Not Axi2Seqid is valid, configure the HT bus emitted  Seqid |
| " | Priority Nop | 4 | 0 x4 | R/W | HT bus Nop stream packet priority |
| and | Specify the NPC | 4 | 0 x3 | R/W | Non Post channel read/write priority |
| The log | Priority RC | 4 | 0 x2 | R/W | The Response channel reads and writes the first stage |
| 3-0 | Priority PC | 4 | 0 x1 | R/W | Post channel read/write priority  0x0: Highest priority  0xF: Lowest priority  Priority strategy is adopted for each channel according to time change, and this storage is used to configure each channel  Initial priority |

### Receiving diagnostic register

Offset: 0x54

Reset value: 0x00000000

Name: Receive diagnostic register

Table 10-14 receives 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 |
| " | ri x\_ctl\_catch | 24 | 0 x0 | R/W | Save the sampled input CTL  (0, 2, 4, 6) correspond to four phases of CTL0 sampling  (1, 3, 5, 7) correspond to four phases of CTL1 sampling |
| Caused the | ri x\_cad\_phase\_0 | 24 | 0 x0 | R/W | Save the value of the sampled input CAD[15:0] |

### Interrupt routing mode selects registers

Offset: 0x58

Reset value: 0x00000000

Name: Interrupt routing mode select register

Table 10-15 Interrupt routing mode select register

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 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, detailed description is shown in the 0 interrupt vector register  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 initializes the configuration register

Table 10-16 receive buffer initial register

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | Reset value | access | describe |
| he | rx\_buffer\_r\_data | 4 | 0 x0 | R/W | Read Buffer initialization information for the receive buffer |
| Behold, | rx\_buffer\_npc\_data | 4 | 0 x0 | R/W | The NPC data Buffer that receives the buffer initializes the information |
| He hath | rx\_buffer\_pc\_data | 4 | 0 x0 | R/W | Receive PC data Buffer initialization information for the buffer |
| " | rx\_buffer\_b\_cmd | 4 | 0 x0 | R/W | Receive bResponse buffer initialization information for the buffer |
| and | rx\_buffer\_r\_cmd | 4 | 0 x0 | R/W | Receives read buffer initialization information for the buffer |
| The log | rx\_buffer\_npc\_cmd | 4 | 0 x0 | R/W | The NPC command Buffer that receives the buffer initializes the information |
| 3-0 | rx\_buffer\_pc\_cmd | 4 | 0 x0 | R/W | Receive THE PC command Buffer initialization information for the buffer |

### The receive address window configures registers

The hitting formula of address window in HT controller is as follows:

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

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

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

The address of the receiving address window is the address received on the HT bus. HT address in the P2P window will be forwarded back to THE HT bus as a P2P command, HT address in the normal receiving window and not in the P2P window will be sent to the CPU, and commands at other addresses will be forwarded back to the HT bus as a P2P command.

Offset: 0x60

Reset value: 0x00000000

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 enabled signal |
| 29:0 | ht\_rx\_image0\_  Trans [53:24] | 30 | 0 x0 | R/W | HT bus receives address window 0, mapped address [53:24] |

Offset: 0x64

Reset value: 0x00000000

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** |
| **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 receives 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

HT Bus Receive Address Window 1 enabling (external access)

Table 10-19 HT bus receiver address window 1 enables (externally accessible) 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 enabled signal |
| 29:0 | ht\_rx\_image1\_  Trans [53:24] | 30 | 0 x0 | R/W | HT bus receives address window 1, mapped address [53:24] |

Offset: 0x6c

Reset value: 0x00000000

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 receives address window 1, address base address [39:24] |
| 15:0 | ht\_rx\_image1\_  Mask [they] | 16 | 0 x0 | R/W | HT bus receiving address window 1, address shielded [39:24] |

Offset: 0x70

Reset value: 0x00000000

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 | HT bus receives address window 2, mapping enabled signal |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| 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

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 receives address window 2, address base address [39:24] |
| 15:0 | ht\_rx\_image2\_  Mask [they] | 16 | 0 x0 | R/W | HT bus receiving address window 2, address shielded [39:24] |

Offset: 0x148

Reset value: 0x00000000

HT Bus Receive Address Window 3 enable (external access)

Table 10-23 HT bus receiver address window 3 enables (externally accessible) 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 enabled signal |
| 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

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 receiving address window 3, address shielded [39:24] |

Offset: 0x150

Reset value: 0x00000000

HT Bus Receive Address Window 4 enable (external access)

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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 enabled 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

HT Bus Receiving Address Window 4 Base Address (external access)

Table 10-26 HT bus receiving 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 receives address window 4, address base address [39:24] |
| 15:0 | ht\_rx\_image4\_  Mask [they] | 16 | 0 x0 | R/W | HT bus receiving address window 4, address shielded [39:24] |

### Interrupt vector register

Interrupt vector register a total of 256, including removal of HT bus Fix, interrupt Arbiter and PIC 256 map directly to the interrupt vector, other plants, such as SMI, NMI, INIT, INTA, intb.br deal, a steady, 0 x50 [doth INTD can register mapped to any one of the eight interrupt vector, the order of the map for {INTD, steady, intb.br deal, INTA, 1 'b0, INIT, NMI, SMI}. At this point, the corresponding value of Interrupt vector is {Interrupt Index, internal vector [2:0]}.

LS3A1000E and above, 256 interrupt vectors are mapped to different interrupt lines based on different register configurations according to the interrupt routing mode. The specific mapping mode is as follows:

Ht\_int\_stripe\_1:

[0,1,2,3... 63] corresponds to neutral 0 /HT HI corresponds to neutral 4

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

[128,129,130,131... 191] corresponds to median 2 /HT HI corresponds to median 6

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

[0,2,4,6... 126] corresponds to neutral 0 /HT HI corresponds to neutral 4

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

[128,130,132,134...... 254] corresponds to medium break 2 /HT HI corresponds to medium break 6

[129,131,133,135... 255] The median break 3 /HT HI the median break 7 ht\_int\_stripe\_4:

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

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

[2,6,10,14... 254] corresponding medium break line 2 /HT HI corresponding medium break line 6

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

The following interrupt vector description corresponds to HT\_int\_stripe\_1, and the other two modes can be obtained from the above description. For LS3A1000D and below, only HT\_int\_stripe\_1 can be used.

Offset: 0x80

Reset value: 0x00000000

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** |
| 31:0 | Interrupt\_case  [31:0] | 32 | 0 x0 | R/W | HT bus interrupt vector register [31:0],  So this is 0 over HT HI and this is 4 |

Offset: 0x84

Reset value: 0x00000000

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],  So this is 0 over HT HI and this is 4 |

Offset: 0x88

Reset value: 0x00000000

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],  It's 1 over HT HI, it's 5 |

Offset: 0x8c

Reset value: 0x00000000

HT Bus Interrupt Vector Register [127:96]

Table 10-30 HT Bus Interrupt Vector register Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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],  It's 1 over HT HI, it's 5 |

Offset: 0x90

Reset value: 0x00000000

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 is 2 over HT HI and this is 6 |

Offset: 0x94

Reset value: 0x00000000

HT Bus Interrupt Vector Register [191:160]

Table 10-31 HT Bus Interrupt Vector Register Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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 is 2 over HT HI and this is 6 |

Offset: 0x98

Reset value: 0x00000000

HT Bus Interrupt Vector Register [223:192]

Table 10-32 HT Bus Interrupt Vector Register Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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  [223-192] | 32 | 0 x0 | R/W | HT bus interrupt vector register [223:192],  It's 3 over HT HI, it's 7 |

Offset: 0x9c

Reset value: 0x00000000

HT Bus Interrupt Vector Register [255:224]

Table 10-33 HT Bus Interrupt Vector Register Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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],  It's 3 over HT HI, it's 7 |

### Interrupt enable register

There are 256 interrupt enabled registers, corresponding to interrupt vector registers. Set 1 is open for the corresponding interrupt, and set 0 is interrupt shielding.

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

Ht\_int\_stripe\_1:

[0,1,2,3... 63] corresponds to neutral 0 /HT HI corresponds to neutral 4

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

[128,129,130,131... 191] corresponds to median 2 /HT HI corresponds to median 6

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

[0,2,4,6... 126] corresponds to neutral 0 /HT HI corresponds to neutral 4

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

[128,130,132,134...... 254] corresponds to medium break 2 /HT HI corresponds to medium break 6

[129,131,133,135... 255] The median break 3 /HT HI the median break 7 ht\_int\_stripe\_4:

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

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

[2,6,10,14... 254] corresponding medium break line 2 /HT HI corresponding medium break line 6

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

The following interrupt vector description corresponds to HT\_int\_stripe\_1, and the other two modes can be obtained from the above description.

Offset: 0xa0

Reset value: 0x00000000

HT Bus Interrupt Enable 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 interrupt enable register [31:0],  So this is 0 over HT HI and this is 4 |

Offset: 0xA4

Reset value: 0x00000000

HT Bus Interrupt Enable 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 interrupt enable register [63:32],  So this is 0 over HT HI and this is 4 |

Offset: 0xa8

Reset value: 0x00000000

HT Bus Interrupt Enable 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 interrupt enable register [95:64],  It's 1 over HT HI, it's 5 |

Offset: 0xAC

Reset value: 0x00000000

HT Bus Interrupt Enable 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 interrupt enable register [127:96],  It's 1 over HT HI, it's 5 |

Offset: 0xb0

Reset value: 0x00000000

HT Bus Interrupt Enable 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 interrupt enable register [159:128],  So this is 2 over HT HI and this is 6 |

Offset: 0xb4

Reset value: 0x00000000

HT Bus Interrupt Enable Register [191:160]

Table 10-39 HT Bus Interrupt Enabled Register Definition

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

Offset: 0xb8

Reset value: 0x00000000

Name: HT Bus Interrupt Enable 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 interrupt enable register [223:192],  It's 3 over HT HI, it's 7 |

Offset: 0xBC

Reset value: 0x00000000

HT Bus Interrupt Enable 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 interrupt enable register [255:224],  It's 3 over HT HI, it's 7 |

### Interrupt Discovery & Configuration

Offset: 0xc0

Reset value: 0x80000008

Name: Interrupt Capability

Table 10-42 Inte ri ri while upt Capab l ty register definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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 the 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 Datap yao ri t register definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| 31:0 | Dataport | 32 | 0 x0 | R/W | When the previous register Index is 0x10, the read-write result of this register is 0xA8, otherwise it is 0xAC |

Offset: 0xc8

Reset value: 0xF8000000

Name: IntrInfo [31:0]

Table 10-44 Int ri Inf yao register definition (1)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| came | IntrInfo [came] | 32 | 0 xf8 | R | reserve |
| Isle; | IntrInfo [great] | 22 | 0 x0 | R/W | IntrInfo[23:2], when PIC interrupt is emitted, value of IntrInfo  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 Int ri Inf yao register definition (2)

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

### The POST address window configures registers

See Section 10.5.7 for the hit formula of address window.

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

Offset: 0xd0

Reset value: 0x00000000

HT Bus POST Address window 0 enabled (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, enabling signal |
| 30 | ht\_depart0\_en | 1 | 0 x0 | R/W | HT access unpacking 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

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 shielded [39:24] |

Offset: 0xd8

Reset value: 0x00000000

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, enabling signal |
| 30 | ht\_depart1\_en | 1 | 0 x0 | R/W | HT access unpacking 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

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 shielded [39:24] |

### The prefetch address window configures registers

See Section 10.5.7 for the hit formula of address window.

The address of this window is the address received on a AXI bus. The fetch instruction and CACHE access in this window will be sent to THE HT bus. Other fetch instruction or CACHE access will not be sent to the HT bus, but will be returned immediately. If it is a read command, the corresponding number of invalid read data will be returned.

Offset: 0xe0

Reset value: 0x00000000

HT Bus Preaddressable window 0 enabling (internal access)

Table 10-50 HT Bus Prefetch 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, enabling signal |
| then | Reserved | 15 | 0 x0 |  | reserve |
| 15:0 | ht\_prefetch0\_trans  [they] | 16 | 0 x0 | R/W | HT bus prefetchable address window 0, translated address [39:24] |

Offset: 0xe4

Reset value: 0x00000000

HT Bus Preaddressable 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\_  The base [they] | 16 | 0 x0 | R/W | HT bus prefetchable address window 0, address base address of [39:24]  An address |
| 15:0 | ht\_prefetch0\_  Mask [they] | 16 | 0 x0 | R/W | HT bus prefetchable address window 0, address shielded [39:24] |

Offset: 0xe8

Reset value: 0x00000000

HT Bus Preaddressable Window 1 enabling (internal access)

Table 10-52 HT Bus Prefetch Address Window 1 Enabling (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, enabling signal |
| then | Reserved | 15 | 0 x0 |  | reserve |
| 15:0 | ht\_prefetch1\_  Trans. [they] | 16 | 0 x0 | R/W | HT bus prefetchable address window 1, translated address [39:24] |

Offset: 0xEC

Reset value: 0x00000000

HT Bus Preaddressable 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** |
| Caused the | ht\_prefetch1\_  The base [they] | 16 | 0 x0 | R/W | HT bus prefetchable address window 1, address base address [39:24] |
| 15:0 | ht\_prefetch1\_  Mask [they] | 16 | 0 x0 | R/W | HT bus prefetchable address window 1, address shielded [39:24] |

### The UNCACHE Address window configures registers

See Section 10.5.7 for the hit formula of address window.

The address of this window is the address received on the HT bus. Read and write commands that fall into this window address will not be sent to SCACHE, nor will they invalidate a primary CACHE, but will be sent directly to memory or other address space, meaning that the read and write commands in this window will not maintain IO CACHE consistency. This window is mainly for some operations that will not hit in the CACHE so that it can improve the efficiency of memory, such as video memory access.

Offset: 0xf0

Reset value: 0x00000000

Name: HT Bus Uncache Address window 0 enabled (internal access)

Table 10-54 HT Bus Uncache Address Window 0 enabled (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, enabling signal |
| 30 | ht\_uncache0\_  trans\_en | 1 | 0 x0 | R/W | HT bus UNCache address window 1, map enabling signal |
| 29:0 | ht\_uncache0\_  Trans [53:24] | 16 | 0 x0 | R/W | HT bus UNCache address window 0, after translation 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 shielded [39:24] |

Offset: 0xf8

Reset value: 0x00000000

Name: HT Bus Uncache Address Window 1 enabled (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, enabling signal |
| 30 | ht\_uncache1\_  trans\_en | 1 | 0 x0 | R/W | HT bus UNCache address window 1, map enabling 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 shielded [39:24] |

Offset: 0x168

Reset value: 0x00000000

Name: HT Bus Uncache Address Window 2 enabled (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, enabling signal |
| 30 | ht\_uncache1\_  trans\_en | 1 | 0 x0 | R/W | HT bus UNCache address window 2, map enabling signal |
| 29:0 | ht\_uncache1\_  Trans [53:24] | 16 | 0 x0 | R/W | HT Bus Uncache Address window 2, translated address [53:24] |

Offset: 0x16c

Reset value: 0x00000000

Name: HT Bus Uncache Address Window 2 Base address (internal access)

Table 10-59 HT Bus Uncache Address Window 2 Base address (internal access)

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

Offset: 0x170

Reset value: 0x00000000

Name: HT Bus Uncache Address Window 3 enabled (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, enabling signal |
| 30 | ht\_uncache1\_  trans\_en | 1 | 0 x0 | R/W | HT bus UNCache address window 3, map enabling signal |
| 29:0 | ht\_uncache1\_  Trans [53:24] | 16 | 0 x0 | R/W | HT Bus Uncache Address window 3, translated address [53:24] |

Offset: 0x174

Reset value: 0x00000000

Name: HT Bus Uncache Address Window 3 Base address (internal access)

Table 10-61 HT Bus Uncache Address Window 3 Base address (internal access)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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 shielded [39:24] |

### The P2P address window configures registers

See Section 10.5.7 for the hit formula of address window.

The address of this window is the address received on the HT bus. The read and write command falling on the address of this window is directly forwarded back to the bus as a P2P command, which has the highest priority compared to the normal receive window and Uncache window.

Offset: 0x158

Reset value: 0x00000000

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\_rx\_image2\_en | 1 | 0 x0 | R/W | HT bus P2P address window 0, enabling signal |
| 30 | Ht\_rx\_image2\_ trans\_en | 1 | 0 x0 | R/W | HT bus P2P address window 0, mapping enabled signal |
| 29:0 | ht\_rx\_image2\_  Trans [53:24] | 16 | 0 x0 | R/W | HT bus P2P address window 0, translated address [53:24] |

Offset: 0x15C

Reset value: 0x00000000

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\_rx\_image2\_ base [they] | 16 | 0 x0 | R/W | HT bus P2P address window 1, address base address [39:24] |
| 15:0 | ht\_rx\_image2\_  Mask [they] | 16 | 0 x0 | R/W | HT bus P2P address window 1, address shielded [39:24] |

Offset: 0x160

Reset value: 0x00000000

HT bus P2P address window 1 enabling (external access)

Table 10-64 HT bus P2P address window 1 enables (externally accessible) register definitions

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A domain** | **A domain name** | **A wide** | **Reset value** | **access** | **describe** |
| 31 | ht\_rx\_image2\_en | 1 | 0 x0 | R/W | HT bus P2P address window 1, enabling signal |
| 30 | Ht\_rx\_image2\_ trans\_en | 1 | 0 x0 | R/W | HT bus P2P address window 1, mapping enabled signal |
| 29:0 | ht\_rx\_image2\_  Trans [53:24] | 16 | 0 x0 | R/W | HT bus P2P address window 1, translated address [53:24] |

Offset: 0x164

Reset value: 0x00000000

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\_rx\_image2\_  The base [they] | 16 | 0 x0 | R/W | HT bus P2P address window 1, address base address [39:24] |
| 15:0 | ht\_rx\_image2\_  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 observe the number of caches available for each command channel at the sender. Offset: 0x100

Reset value: 0x00000000

Name: Command sends cache size register

Table 10-66 commands send 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 on the sending side |
| " | NPC\_CMD\_txbuffer | 8 | 0 x0 | R | Number of NPC channel command caches on the sending side |
| away | PC\_CMD\_txbuffer | 8 | 0 x0 | R | Number of caches of sending PC channel commands |

### Data sent cache size register

The data send cache size register is used to observe 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 send cache size register

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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 | Number of R channel data caches at the sending end |
| " | NPC\_DATA\_txbuffer | 8 | 0 x0 | R | Number of NPC channel data caches on the sending side |
| away | PC\_DATA\_txbuffer | 8 | 0 x0 | R | Number of PC channel data caches at the sending end |

### 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

Table 10-68 sends the cache debug register

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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 | The sender cache debug symbol  0: Increase the corresponding number  1: Reduce (corresponding register number +1) |
| 28 | Tx\_buff\_adj\_en | 1 | 0 x0 | R/W | The sender cache debugging enablement register  0->1: causes the value of this register to increase or decrease once |
| he | R\_DATA\_txadj | 4 | 0 x0 | R/W | Number of increase or decrease of R channel data cache at 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 | The number of NPC channel data cache increases or decreases on the sending side  When tx\_NEg is 0, add NPC\_DATA\_txadj; When tx\_NEg is 1, reduce NPC\_DATA\_txadj+1 |
| He hath | PC\_DATA\_txadj | 4 | 0 x0 | R/W | Number of increase or decrease of data cache on PC channel at 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 | Number of increase or decrease in the command cache of the sending side B channel  When tx\_NEg is 0, increase B\_CMD\_txadj; When tx\_NEg is 1, reduce B\_CMD\_txadj+1 |
| and | R\_CMD\_txadj | 4 | 0 x0 | R/W | Number of increase or decrease of R channel command cache on sending side  When tx\_NEg is 0, increase R\_CMD\_txadj; When tx\_NEg is 1, reduce R\_CMD\_txadj+1 |
| The log | NPC\_CMD\_txadj | 4 | 0 x0 | R/W | Number of NPC channel commands/data cache increases or decreases on the sending side  When tx\_NEg is 0, add NPC\_CMD\_txadj; When tx\_NEg is 1, reduce NPC\_CMD\_txadj+1 |
| 3-0 | PC\_CMD\_txadj | 4 | 0 x0 | R/W | The number of increase or decrease of command cache on PC channel on sending side  When tx\_NEg is 0, increase PC\_CMD\_txadj; When tx\_NEg is 1, reduce PC\_CMD\_txadj+1 |

### PHY impedance matching control register

Impedance matching enablement for controlling THE PHY and impedance matching parameter setting for the transmitter and the receiver

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 enablement |
| 30 | Rx\_scanin\_en | 1 | 0 x0 | R/W | RX impedance matching enablement |
| 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 to a new version number that takes effect via Warm Reset. Offset: 0x110

Reset value: 0x00200000

Name: RevisionID register

Table 10-70 Rev S yao N 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  0 x20: HyperTransport 1.00  0 x60: HyperTransport 3.00 |
| 15:0 | Reserved | 16 | 0 x0 | R | reserve |

### Error Retry controls the register

For error retransmission enabled in HyerTransport 3.0 mode, the maximum number of times Short Retry is configured, displayed

Whether the Retry counter is flipped. Offset: 0x118

Reset value: 0x00000000

Name: Error Retry control register

Table 10-71 - E ri ri yao ri Ret ri y 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 | The Retry counter flips its count |
| 8 | Reserved | 1 | 0 x0 | R | reserve |
| but | Short Retry Attempts | 2 | 0 x0 | R/W | Maximum number of Short retries 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 Ret ri y C yao without 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 transmission in consistent mode  000:0 delay  001: Random delay 0-8  010: Random delay 8-15  011: Random delay 16-31  100: Random delay 32-63  101: Random delay 64-127  110: Random delay 128-255  111:0 delay |
| 15:0 | Retry Count | 16 | 0 x0 | R | Retry count |

### Link Train register

HyperTransport 3.0 Link initialization and Link Training Control Register.

Offset: 0x130

Reset value: 0x00000070

Name: Link Train register

Table 10-73 - L nk ri T a n registers

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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 on the sending side 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 four consecutive DWS by default.  1 'b0 enabled Cmd Throttling  1 'B1 prohibits the use of Cmd Throttling |
| " | Reserved | 4 | 0 x0 | R | reserve |
| " | Receiver LS the select | 2 | 0 x0 | R/W | Link state on the receiving end in a 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 | Maximum number of times Long Retry |
| 3 | Scrambling the Enable | 1 | 0 x0 | R/W | (3) : to misuse or deprive of health care  1: can Scramble |
| 2 | 8 b10b Enable | 1 | 0 x0 | R/W | Whether to enable 8B10B 0: Ban 8B10B  1: can make 8 b10b |
| 1 | AC | 1 | 0 x0 | R | Is AC Mode detected  AC Mode 1: AC Mode was detected |
| 0 | Reserved | 1 | 0 x0 | R | reserve |

### Training 0 short timeout register

It is used to configure the short time timeout threshold of Training 0 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: Short timeout count register for Training 0

Table 10-74 - T ri a short n ng 0 timeout timing registers

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

### Training 0 timeout long timing register

It is used for Training 0 long count timeout threshold in HyerTransport 3.0 mode, and 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 - T ri n ng zero overtime long count register a

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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

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

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

Reset value: 0x0004fffff

Name: Training 1 counting register

Table 10-76 - T ri n ng 1 counter register a

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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 - T ri n ng 2 counter register a

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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

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

HyperTransport3.0 Link bus clock frequency 1/4.

Offset: 0x13C

Name: Training 3 counting register

Table 10-78 - T ri n ng 3 counter register a

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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 registers

It is used to switch to the link frequency and controller frequency supported by any protocol and PLL during the operation. The specific switching method is as follows: under 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 to determine the PLL output frequency, phy\_hi\_div and phy\_lo\_div on the link, and core\_div for the controller. After entering Warm Reset or LDT Disconnect, the controller will automatically reset the 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\_CORE\_CLK = 100MHz×div\_loop/div\_refc /core\_div

The waiting time for PLL re-lock is about 30US when the system CLK is 33M by default. You can also write a custom wait count upper limit in the register;

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 | The upper limit of the counter configures the register when setting the counter SELECT  , the upper limit of the counter count is |
|  |  |  |  |  | {PLL\_relock\_counter,5 'h1f}, otherwise count upper limit  For 10 '3 ff |
| 26 | Counter the select | 1 | 0 x0 | R/W | Lock timer custom enablement:  1 'b0 USES the default upper limit;  1 'b1 is calculated by PLL\_relock\_counter |
| Struggled together | Soft\_phy\_lo\_div | 4 | 0 x0 | R/W | High LEVEL PHY frequency division coefficient |
| Lift up | Soft\_phy\_hi\_div | 4 | 0 x0 | R/W | Low LEVEL PHY frequency division coefficient |
| " | Soft\_div\_refc | 2 | 0 x0 | R/W | PLL frequency division coefficient |
| Put no | Soft\_div\_loop | 7 | 0 x0 | R/W | PLL internal frequency multiplication factor |
| then | Soft\_core\_div | 4 | 0 x0 | R/W | Frequency division coefficient of controller clock |
| 4-2 | Reserved | 3 | 0 x0 | R | reserve |
| 1 | Soft cofig enable | 1 | 0 x0 | R/W | Software configuration enablement bit  1 'b0 disables software frequency configuration  1 'b1 enabled software frequency configuration |
| 0 | Reserved | 1 | 0 x0 | R | reserve |

### PHY configuration register

PHY are controlled independently by the two controllers respectively. When the controller is a 16bit controller, the high order sum It is used to configure phY-related physical parameters. When the controller is two independent 8BIT controllers, the higher level PHY and the lower level

The configuration parameters of the low level PHY are controlled by the low level 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 | Terminal impedance of PLL to RX terminal |
| 30 | Tx\_ckpll\_term | 1 | 0 x0 | R/W | Terminal impedance of PLL to TX terminal |
| 29 | Rx\_clk\_in\_sel\_ | 1 | 0 x0 | R/W | The clock PAD is provided with 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 DLLS:  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 restrictions |
| 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 is disabled  1 'b1 BandGap can make |
| michal | Reserved | 3 | 0 x0 | R | reserve |
| then | Tx\_preenmp | 5 | 0 x08 | R/W | PAD pre-weighted control signal |
| 11:0 | Reserved | 12 | 0 x0 | R | reserve |

### Link initializes the debug register

In HyperTransport 3.0 mode, it is used to configure whether THE CDR lock signal provided by the PHY is used as the symbol of completion of link CDR during link initialization. If the lock signal is ignored, the default CDR is completed after the controller counts for a certain amount of time.

Offset: 0x180

Reset value: 0x00000000

Name: Link initialization debug register

Table 10-81 Link Initialization debug register

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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 during link initialization and wait is completed through counter counting:  1 'b0 wait for CDR lock  1 'b1 ignores the CDR lock signal and waits by accumulating through the counter |
| 14:0 | Cdr\_wait\_counter | 15 | 0 x0 | R/W | Wait for the upper limit of the counter count, based on the controller clock completion technique |

### LDT debug register

When the software changes the frequency of the controller, the timing of LDT REconnect stage will be inaccurate. The counter needs to be configured.

After the software is configured as a frequency, the time between the LDT signal being invalid and the initialization of the controller start link is based on the controller clock.

Offset: 0x184

Reset value: 0x00000000

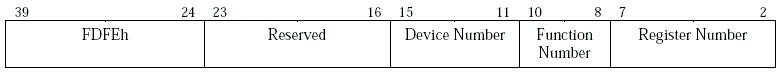
Name: LDT debug register

Table 10-82 LDT debug register

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

#### The HyperTransport bus concodes access methods for Spaces

The protocol of the HyperTransport interface software layer is basically the same as PCI protocol, but the details of the access are slightly different because the access to the configuration space is directly related to the underlying protocol. As listed in Table 10-5, the address range of the HT bus configuration space is 0xFD\_FE00\_0000 ~ 0xFD\_FFFF\_FFFF. For configuration access in HT protocol, the following format is adopted in longson 3A300/3B3000:



Type 0:

Type 1:

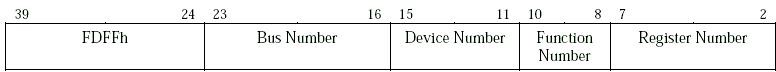


Figure 10-1 Configuration access of HT protocol in Longson 3A3000/3B3000

#### HyperTransport multiprocessor support

The Loongson 3 processor USES the HyperTransport interface for multiprocessor interconnection, and the hardware can automatically maintain consistency requests between the four chips. Two methods of multiprocessor interconnection are provided below:

##### Four - chip Longshon no. 3 interconnection structure

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

8 bit HT bus

8 bit HT bus

Where HTx\_LO serves as the primary device and HTx\_HI serves as the connection from the secondary device. Thus, the following interconnection structure can be obtained:

IO

8 bit HT bus

|  |  |  |
| --- | --- | --- |
| HT1  \_LO | CPU0 | HT0\_ LO |
| HT1  \_HI | HT0\_ HI |

|  |  |  |
| --- | --- | --- |
| HT1  \_HI |  | HT0\_ LO |
| CPU2 | | HT0\_ HI |

|  |  |  |
| --- | --- | --- |
| HT0\_ HI |  | HT1  \_HI |
| HT0\_ LO | CPU3 | |

FIG. 10-2 Interconnection structure of No. 3 with four pieces of Longshon

8 bit HT bus

CPU1

HT1

\_HI

8 bit HT bus

HT0\_ LO

HT0\_ HI

##### Loongson no.3 Interconnection route

The simple X-Y route is adopted for The Longshon no. 3 Interconnection route. In other words, when routing, X is followed by Y. Take four chips for example, ID number is 00,01,10,11 respectively. If a request is sent from 11 to 00, then 11 is routed to 00, first in the X direction,

I go 11 to 10, and then I go Y, and Then I go 10 to 00. When the response to the request returns 11 from 00, 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 routes. Due to the characteristics of this algorithm, we will take a different approach when building the interconnection between two chips.

##### Two - piece loong chip no. 3 interconnection structure

Due to the nature of the fixed routing algorithm, we have two different approaches to constructing the interconnection between two chips. The first is to use 8 bit HT bus interconnection. In this interconnection mode, only 8-bit HT interconnection can be used between two processors. The two chip Numbers are 00 and 01 respectively. From the routing algorithm, we can know that both chips access each other through the 8-bit HT bus consistent with the four-chip interconnection. As shown below:

8 bit HT bus

IO

16-bit HT bus

HT0\_ HI

HT0\_ LO

CPU0

HT 1

HT0\_ HI

CPU1

HT0\_ LO

Figure 10-3 Interconnection structure of two pieces of Loong chip no.3 and 8-bit

However, the widest HT bus can adopt the 16-bit mode, so the connection mode to maximize the bandwidth should adopt the 16-bit interconnection structure. In Loongson 3, 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 before. In this way, we can use the 16-bit bus when connecting. Therefore, we only need to correctly configure the 16-bit mode of CPU0 and CPU1 and properly connect the high-low bit bus to use the 16-bit HT bus to interconnect. The interconnection structure can also be accessed using 8-bit HT bus protocol. The resulting interconnection structure is as follows:

16-bit HT bus

CPU1

HT0

IO

16-bit HT bus

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

FIG. 10-4 16-bit interconnection structure of two pieces of Loong chip No.3

## Low speed IO controller configuration

The Loongson 3 I/O controller includes PCI controller, LPC controller, UART controller, SPI controller, GPIO and configuration register. These I/O controllers share a port where CPU requests are addressed and sent to the appropriate device.

#### The PCI controller

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

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

Table 11-1 PCI controller configuration header

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| The byte 3 | 2 bytes | 1 byte | Byte 0 | address |
| Dev ce ID | | Vend yao ri ID | | 00 |
| The Status | | C yao mmand | | 04 |
| Class C yao DE | | | Rev S yao N ID | 08 |
| BIST | Heade ri Type | Latency ri T me | CacheL ne S ze | 0 c |
| Base the Add ri ess Reg ste ri 0 | | | | 10 |
| Base the Add ri ess Reg ste ri 1 | | | | 14 |
| Base the Add ri ess Reg ste ri 2 | | | | 18 |
| Base the Add ri ess Reg ste ri 3 | | | | 1 c |
| Base the Add ri ess Reg ste ri 4 | | | | 20 |
| Base the Add ri ess Reg ste ri 5 | | | | 24 |
|  | | | | 28 |
| Subsystem ID | | Subsystem Vend yao ri ID | | 2 c |
|  | | | | 30 |
|  | | | Capab l t es P yao nte ri | 34 |
|  | | | | 38 |
| Max mum Latency | M n mum G ri ant | Inte ri ri while upt P n | Inte ri ri while upt ne L | 3 c |
| Implementat yao n Spec f c Reg ste ri (ISR40) | | | | 40 |
| Implementat yao n Spec f c Reg ste ri (ISR44) | | | | 44 |
| Implementat yao n Spec f c Reg ste ri (ISR48) | | | | 48 |
| Implementat yao n Spec f c Reg ste ri (ISR4C) | | | | 4 c |
| Implementat yao n Spec f c Reg ste ri (ISR50) | | | | 50 |
| Implementat yao n Spec f c Reg ste ri (ISR54) | | | | 54 |
| Implementat yao n Spec f c Reg ste ri (ISR58) | | | | 58 |
|  | | | | . |
| PCIX C yao mmand Reg ste ri | | | | E0 |
| PCIX Status Reg ste ri | | | | E4 |

The PCIX controller of Longson 3A300/3B3000 supports three 64-bit Windows, including {BAR1, BAR0}, {BAR3, BAR2},

Three pairs of registers configure the base addresses of Windows 0, 1, and 2. The window size, enablement, and other details 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 | Ta ri \_ ri ead\_ yao | Read and write  (Write 1 clear) | 0 | Ta ri get receive access to IO or area is not prefetch |
| 30 | Ta ri \_ ri ead\_d sca ri d | Read and write  (Write 1 clear) | 0 | Ta ri get the delay request to be discarded |
| 29 | Ta ri \_ ri esp\_delay | Read and write | 0 | When ta ri get access delay/SPL is given t 0: after a timeout  1: immediately |
| 28 | Ta ri \_delay\_ ri et ri y | Read and write | 0 | Ta ri get access to retry strategy  0: According to internal logic (see 29 bits)  1: Try again right away |
| 27 | Ta ri \_ ri ead\_ab yao ri t\_en | Read and write | 0 | If ta ri get read request to internal timeout, whether to ta ri get - ab yao ri t respond |
| " | Rese ri ved | - | 0 |  |
| 24 | Ta ri \_w ri te\_ab yao ri t\_en | Read and write | 0 | If ta ri get to write requests within the timeout, whether to ta ri get - ab yao ri t respond |
| 23 | Ta ri \_maste ri \_ab yao ri t | Read and write | 0 | Whether to allow maste ri - ab yao ri t |
| Lift up | Ta ri \_subseq\_t me yao ut | Read and write | 000 | Ta ri get further delayed timeouts  000:8 cycles  Others: not supported |
| He hath | Ta ri \_ n t\_t me yao ut | Read and write | 0000 | Ta ri get initial delay timeout  PCI mode  0:16 cycles  1-7: Disable the counter  8-15:8 to 15 cycles  The timeout count is fixed to 8 cycles in PCIX mode, where 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 visit |
|  |  |  |  | 12:5 delay visit  13:6 delay visit  14:7 delay visit  15: 8 delay visit |
| Indeed, | Ta ri \_p ri ef\_b yao unda ri y | Read and write | 000 h. | Prefetching boundary configuration (in 16 bytes)  FFF: 64KB to 16byte FFE: 64KB to 32byte FF8: 64KB to 128byte |
| 3 | Ta ri \_p ri ef\_b yao und\_en | Read and write | 0 | Using ta ri \_p ri ef\_b yao unda ri y configuration  0: Prefetch to device boundary  1: using ta ri \_p ri ef\_b yao unda ri y |
| 2 | Rese ri ved | - | 0 |  |
| 1 | Ta ri \_spl tw\_ct ri l | Read and write | 0 | Ta ri get SPL t write control  0: stop except P yao sted Mem ri te yiu ri y W outside access  1: Block all access until SPL T is complete |
| 0 | Mas\_lat\_t me yao ut | Read and write | 0 | Disable mate ri access timeout  0: allow maste ri access timeout  1: not allowed |
| REG\_44 | | | | |
| 31:0 | Rese ri ved | - | - |  |
| REG\_48 | | | | |
| 31:0 | Ta ri \_pend ng\_seq | Read and write | 0 | Ta ri a get request to the pending bit vector  Write 1 in the corresponding bit to make it clear |
| REG\_4C | | | | |
| charm | Rese ri ved | - | - |  |
| 29 | Mas\_w ri te\_defe ri | Read and write | 0 | Allows subsequent reads to override previous, unfinished writes  (valid for PCI only) |
| 28 | Mas\_ ri ead\_defe ri | Read and write | 0 | Allows subsequent reads and writes to override previous incomplete reads  (valid for PCI only) |
| 27 | Mas\_ yao \_defe ri \_cnt | Read and write | 0 | Maximum number of IO requests outside  Zero: by the control  1:1. |
| they | Mas\_ ri ead\_defe ri \_cnt | Read and write | 010 | Maste ri support outside the maximum number of read (applies only to PCI)  Zero: 8  1-7:1-7  Note: A dual address cycle access account for two entries |
| Ephron; | E ri ri \_seq\_ d | read-only | 00 h | Ta ri get/maste ri error number |
| 15 | E ri ri \_type | read-only | 0 | Ta ri get/maste ri error command type  Zero: |
| 14 | E ri ri \_m yao dule | read-only | 0 | Faulty module |

)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  | 0: ta ri get  1: maste ri |
| 13 | System\_e ri ri yao ri | Read and write | 0 | Ta ri get/maste ri system fault (write 1 qing) |
| 12 | Data\_pa ri ty\_e ri ri yao ri | Read and write | 0 | Ta ri get/maste ri data parity (write 1 qing) wrong |
| 11 | Ct ri l\_pa ri ty\_e ri ri yao ri | Read and write | 0 | Ta ri get/maste ri address parity error (write 1 qing) |
| 10:0 | Rese ri ved | - | - |  |
| REG\_50 | | | | |
| 31:0 | Mas\_pend ng\_seq | Read and write | 0 | Maste ri request number the pending a vector corresponding to a write 1 but clear |
| REG\_54 | | | | |
| 31:0 | Mas\_spl t\_e ri ri | Read and write | 0 | SPL T returns the wrong request bit vector |
| REG\_58 | | | | |
| charm | Rese ri ved | - | - |  |
| then | Ta ri \_spl t\_p ri yao ri ty | Read and write | 0 | Ta ri get SPL t return to priority  0 is the highest, 3 is the lowest |
| But after | Mas\_ ri eq\_p ri yao ri ty | Read and write | 0 | Maste ri foreign priority  0 is the highest, 3 is the lowest |
| 25 | P ri yao ri ty\_en | Read and write | 0 | Arbitration algorithm (the maste ri access and ta ri get between SPL t return for arbitration  0: Fixed priority  1: rotary |
| Whereupon certain | reserve | - | - |  |
| 17 | Mas\_ ri et ri y\_ab yao ri ted | Read and write | 0 | Maste ri retry to cancel (write 1 qing) |
| 16 | Mas\_t ri dy\_t me yao ut | Read and write | 0 | Maste ri TRDY count overtime |
| " | Mas\_ ri et ri y\_value | Read and write | 00 h | Maste ri retries  0: Infinite retry  1-255:1-255 |
| away | Mas\_t ri dy\_c yao without | Read and write | 00 h | Maste ri 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. Then read and write to the 2K space at the beginning of 0x1fe80000 to access the configuration header of the corresponding device. The device number is obtained from low to high priority coding according to PCIMap\_Cfg[15:0].

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

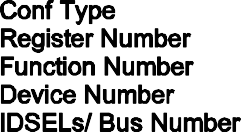


Figure 11-1. Configure read-write bus address generation

PCI arbitrator implements two - stage wheel arbitration, bus docking and isolation of damaged master devices. The configuration and status are shown in the PXArb\_Config and PXArb\_Status registers. PCI bus request and reply line allocation are shown in Table 11-3.

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

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

The rotation-based arbitration algorithm provides two levels, with the second level as a whole being scheduled together as a member of the first level. When multiple devices apply for the bus at the same time, after each round of the first device, the device with the highest priority in the second level can get the bus.

Mediators are designed to be switched whenever conditions permit, and for some PCI devices that do not conform to the protocol, this may make them abnormal. Using mandatory priority allows these devices to take possession of the bus through persistent requests.

Bus docking is whether to select one to give permission when no device requests to use the bus. For devices that are already permitted, initiating the bus operation directly improves efficiency. The internal PCI mediator provides two docking modes: the last master and the default master. If it is not possible to dock in special circumstances, the arbitrator can be set to dock to the default no. 0 primary device (internal controller) with a latency of 0.

#### LPC controller

LPC controller has the following characteristics:

* Comply with LPC1.1 specification
* LPC access timeout counters are supported
* Support Mem yao ri y Read, Mem yaojiang ri y w ri te access types
* Support F ri mwa ri e Mem yao ri Read, F y ri mwa ri e Mem yao ri y W ri te access type (single-byte)
* Support the I/O ri eads, the I/O w ri te access types
* Address translation support Mem yao ri y access type
* According to specifications, support Se ri zl zed IRQ 17 interrupt source address space distribution of the LPC controller are shown in table 11-4:

Table 11-4 Address space distribution of LPC controller

|  |  |  |
| --- | --- | --- |
| **Address name** | **Address range** | **The size of the** |
| LPC B shines t | 0 x1fc0\_0000 x1fd0\_0000 0 | 1 mbyte |
| LPC Mem yao ri y | 0 x1c00\_0000 x1d00\_0000 0 | 16 mbyte |
| LPC I/O | 0 x1ff0\_0000 x1ff1\_0000 0 | 64 kbyte |
| LPC Reg ste ri | 0 x1fe0\_0200 x1fe0\_0300 0 | 256 byte |

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

The LPC Memory address space is the address space accessible by the system for Memory/Firmware Memory. The type of Memory access issued by the LPC controller is determined by the LPC controller's configuration register LPC\_MEM\_IS\_FWH. Addresses sent by the processor to this address space can be converted. The converted address is set by the LPC controller's configuration register LPC\_MEM\_TRANS.

Accesses sent by the processor to the LPC I/O address space are sent to the LPC bus according to the LPC I/O access type. The address is 16 bits lower in the address space.

The LPC controller configuration registers have three 32-bit registers. The meanings of configuring registers are shown in Table 11-5:

Table 11-5 Meanings of LPC configuration registers

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A domain** | **The field name** | **access** | **Reset value** | **instructions** |
| REG0 | | | | |
| REG0 [hands] | SIRQ\_EN | Read and write | 0 | SIRQ enabled control |
| REG0 [take] | LPC\_MEM\_TRANS | Read and write | 0 | LPC Mem yao ri address translation control y space |
| 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 Mem yao ri space F y ri mwa ri e  Mem yao ri y access type |
| 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 interrupt source indication |
| REG3 | | | | |
| REG3 [17:0] | LPC\_INT\_CLEAR | write | 0 | LPC SIRQ interrupt cleanup |

#### UART controller

The UART controller has the following characteristics

* Full duplex asynchronous data receiving/sending
* A programmable data format
* 16-bit programmable clock counter
* Support for receiving timeout detection
* Multiinterrupt system with arbitration
* Work only in FIFO mode
* NS16550A compatible register and function

Two UART interfaces are integrated inside the chip, with exactly the same functional registers, but with different access addresses. The physical address base of the UART0 register is 0x1FE001E0.

The physical address of the UART1 register is 0x1FE001E8.

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

Chinese name: Data Transmission Register Register 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 enablement register (IER)

Interrupt enabled register Register width: [7:0]

Offset: 0x01

Reset value: 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe |
| The log | Rese ri ved | 4 | RW | reserve |
| 3 | IME | 1 | RW | M yao DEM status interruption enables' 0 '-' close '-' open ' |
| 2 | ILE | 1 | RW | Receiver line state interrupts to enable '0' - close '1' - open |
| 1 | ITxE | 1 | RW | The transfer save register enables' 0 '- close' 1 '- open for air break |
| 0 | IRxE | 1 | RW | Receive valid data interrupts enable '0' - close '1' - open |

### Interrupt Identification Register (IIR)

Interrupt source Register Register width: [7:0]

Offset: 0x02

Reset value: 0xc1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe |
| The log | Rese ri ved | 4 | R | reserve |
| 3:1 | II | 3 | R | Interrupt sources represent bits, as shown in the following table |
| 0 | INTp | 1 | R | Interrupt 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 | Parity, overflow, or frame error, or dozen  Break the interrupt | Read the LSR |
| 0 | 1 | 0 | 2 nd | A significant number is received  According to the | The number of characters in FIFO is up to  The level of t ri gge ri | FIFO has a low number of characters  In t ri gge ri values |
| 1 | 1 | 0 | 2 nd | Receive a timeout | At least one character in FIFO,  But there are no operations, including read and write operations, within four characters | Read receive FIFO |
| 0 | 0 | 1 | ri 3 d | Transfer save hosting  Device is empty | The transfer save register is empty | Write data to THR or  More IIR |
| 0 | 0 | 0 | 4 th | M yao DEM state | CTS, DSR, RI yao ri DCD. | Read MSR |

### FIFO control Register (FCR)

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

Offset: 0x02

Reset value: 0xc0

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe |
| but | TL | 2 | W. | Receive the FIFO interrupt request t ri gge ri values  '00' - 1 byte '01' - 4 bytes  '10' - 8 bytes' 11 '- 14 bytes |
| o | Rese ri ved | 3 | W. | reserve |
| 2 | Txset | 1 | W. | '1' clears the contents of sending FIFO and resets its logic |
| 1 | Rxset | 1 | W. | '1' clears the contents of the RECEIVED FIFO and resets its logic |
| 0 | Rese ri ved | 1 | W. | reserve |

### Line Control Register (LCR)

Chinese name: Line Control register Register 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 the operation divider latch  '0' - Access the normal register | | | |
| 6 | BCB | 1 | RW | Interrupt control bit  '1' - The output of the serial port is set to 0(interrupt state). '0' - Normal operation | | | |
| 5 | .spb | 1 | RW | Specifies the parity bit  '0' - Do not specify parity bits  '1' - Transfer and check parity bit 0 if LCR[4] bit is 1. If the LCR[4] bit is 0, the transmission and check parity bit is 1. | | | |
| 4 | eps | 1 | RW | Parity bit selection  '0' - An odd number of 1s in each character (including data and parity bits)  '1' - An even number of 1s in each character | | | |
| 3 | PE | 1 | RW | Parity bit enable  '0' - No parity bits  '1' - Generates parity bits on output, and determines parity bits on input | | | |
| 2 | sb | 1 | RW | Defines the number of bits that generate the stop bits  '0' - 1 stop bit  '1' - 1.5 stop bits in 5 character length, others  The length is 2 stop bits | | | |
| 1-0 | bec | 2 | RW | Sets the number of digits per character  '00' - 5 '01' - 6 bits | | | |
|  |  |  |  | "10" - 7 | position | "11" - 8 | position |

### MODEM Control Register (MCR)

Chinese name: M Yao DEM control register Register width: [7:0]

Offset: 0x04

Reset value: 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe |
| 7:5 | Rese ri ved | 3 | W. | reserve |
| 4 | Yao yao L p | 1 | W. | Loop mode control bit  '0' - Normal operation  '1' - loop mode. In loopback mode, TXD output is always 1, output shift register directly connected 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 loop mode |
| 2 | The OUT1 | 1 | W. | Connect to the 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' - one with at least a parity bit error, frame error, or interrupt.  '0' - No errors |
| 6 | TE | 1 | R | The transfer is empty to represent a bit  '1' - Transfer FIFO and transfer shift register are null. Zero out when writing data to transmit FIFO  '0' - there is data |
| 5 | TFE | 1 | R | Transmit FIFO bit null for bit representation  '1' - The current transmission OF FIFO is null, and the data written to the transmission of FIFO is cleared  '0' - there is data |
| 4 | BI | 1 | R | Interrupts indicate bits  '1' - the start bit received + data + parity + stop bits are 0, that is, there is an interrupt  '0' - No interruptions |
| 3 | FE | 1 | R | A frame error represents a bit  '1' - Received data with no stop bits  '0' - No errors |
| 2 | PE | 1 | R | Parity bit errors represent bits  '1' - An even or odd error is currently receiving data  '0' - No parity errors |
| 1 | OE | 1 | R | Data overflow represents a bit  '1' -- Data overflow  '0' - No overflow |
| 0 | Dr. | 1 | R | The received data effectively represents a bit  '0' - No data in FIFO |
|  |  |  |  | '1' - Data in FIFO |

When reading this register, LSR[4:1] and LSR[7] are cleared, and LSR[6:5] is cleared when writing data to TRANSMIT FIFO, while LSR[0] makes judgment on receiving FIFO.

### MODEM Status Register (MSR)

Chinese name: M yao DEM 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 value, or connect to Out2 in loopback mode |
| 6 | CRI | 1 | R | RI input value inverse, or connected to OUT1 in loopback mode |
| 5 | CDSR | 1 | R | The inverse of the DSR input value, or is connected to the DTR in loopback mode |
| 4 | CCTS | 1 | R | The inverse of the CTS input value, or is connected to the RTS in loopback mode |
| 3 | DDCD | 1 | R | DDCD indicating a |
| 2 | TERI | 1 | R | RI edge detection. RI goes from low to high |
| 1 | DDSR | 1 | R | DDSR indicating a |
| 0 | DCTS | 1 | R | DCTS indicating a |

### Frequency division 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 | The lower 8 bits of the 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 divider latch |

#### SPI controller

The SPI controller has the following characteristics:

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

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

Table 11-6 SPI controller address space distribution

|  |  |  |
| --- | --- | --- |
| **Address name** | **Address range** | **The size of the** |
| SPI B shines t | 0 x1fc0\_0000 x1fd0\_0000 0 | 1 mbyte |
| SPI Mem yao ri y | 0 x1d00\_0000 x1e00\_0000 0 | 16 mbyte |
| SPI Reg ste ri | 0 x1fe0\_0220 x1fe0\_0230 0 | 16 byte |

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

SPI Mem yao ri y space also can direct access to the read request by the CPU, the minimum 1 m bytes and SPI BOOT space overlap.

### Control register (SPCR)

Chinese name: Control register Register width: [7:0]

Offset: 0x00

Reset value: 0x10

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe |
| 7 | Sp e | 1 | RW | Interrupt output to make the energy signal highly efficient |
| 6 | The spe | 1 | RW | The system works to make the energy signal highly efficient |
| 5 | Rese ri ved | 1 | RW | reserve |
| 4 | The MST ri | 1 | RW | Maste ri mode selection, the reserve 1 |
| 3 | Cp group l | 1 | RW | Clock polarity |
| 2 | cpha | 1 | RW | Clock phase 1 is opposite and 0 is the same |
| 1-0 | Sp ri | 2 | RW | Sclk\_ yao divider setting, it is necessary to use with spe ri sp ri e |

### Status Register (SPSR)

Status register Register width: [7:0]

Offset: 0x01

Reset value: 0x05

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe |
| 7 | Sp f | 1 | RW | Interrupt flag bit 1 indicates an interrupt request, write 1 to clear |
| 6 | Wc yao l | 1 | RW | Write register overflow flag bit 1 indicates overflow, write 1 is cleared |
| when | Rese ri ved | 2 | RW | reserve |
| 3 | wffull | 1 | RW | Write register full flag 1 indicates full |
| 2 | wfempty | 1 | RW | Write register null flag 1 indicates null |
| 1 | ri ffull | 1 | RW | Read register full flag 1 indicates full |
| 0 | ri fempty | 1 | RW | Read register null flag 1 indicates null |

### Data Register (TxFIFO)

Chinese name: Data Transmission Register Register 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 width: [7:0]

Offset: 0x03

Reset value: 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A domain | A domain name | A wide | access | describe |
| but | CNT | 2 | RW | Sends an interrupt request signal after how many bytes have been transmitted  00 -- 1 byte 01-2 bytes  10-3 bytes 11-3 bytes |
| 5-2 | Rese ri ved | 4 | RW | reserve |
| 1-0 | Sp ri e | 2 | RW | With Sp ri set the ratio of frequency division |

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 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 division selection (the division coefficient is the same as {SPre, SPR} combination) |
| 3 | dual\_io | 1 | RW | Use dual I/O mode, with priority over 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, when invalid CSN [0] can be controlled by software. |

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

SPI Flash Chip Selection control Register Register 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 cs line is controlled by 7:4 |

#### Timing Control Register (SFC\_TIMING)

SPI Flash Timing Control Register Register 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 shortest invalid time of chip selection signal of SPI Flash is calculated with clock cycle 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 PCI controller address window, the mediator, and the GPIO controller.  These registers are listed in Table 11-7, and the register details are given in Table 11-8. The base address of this register is 0x1FE00100.

Table 11-7 IO control registers

|  |  |  |
| --- | --- | --- |
| **address** | **register** | **instructions** |
| 00 | P yao nCfg | The electric configuration |
| 04 | GenCfg | General configuration |
| 08 | reserve |  |
| 0 c | reserve |  |
| 10 | PCIMap | PCI mapping |
| 14 | PCIX\_B ri dge\_Cfg | PCI/X bridge related configuration |
| 18 | PCIMap\_Cfg | PCI configures read and write device addresses |
| 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\_W n\_Base\_L | Can prefetch window base address low 32 bits |
| 44 | Mem\_W n\_Base\_H | The height of the prefetch window base address is 32 bits |
| 48 | Mem\_W n\_Mask\_L | Prefetchable window mask low 32 bits |
| 4 c | Mem\_W n\_Mask\_H | Can prefetch window mask height 32 bits |
| 50 | PCI\_H t0\_Sel\_L | PCI window 0 controls low 32 bits |

|  |  |  |
| --- | --- | --- |
| 54 | PCI\_H t0\_Sel\_H | PCI window 0 controls high 32 bits |
| 58 | PCI\_H t1\_Sel\_L | PCI window 1 controls low 32 bits |
| 5 c | PCI\_H t1\_Sel\_H | PCI window 1 controls high 32 bits |
| 60 | PCI\_H t2\_Sel\_L | PCI window 2 controls low 32 bits |
| 64 | PCI\_H t2\_Sel\_H | PCI window 2 controls high 32 bits |
| 68 | PXA ri b\_C yao nf g | PCIX mediator configuration |
| 6 c | PXA ri b\_Status | PCIX arbiter status |
| 70 |  |  |
| 74 |  |  |
| 78 |  |  |
| 7 c |  |  |
| 80 | Ch p, C, NF G | Chip configuration register |
| 84 |  |  |
| 88 |  |  |
| 8 c |  |  |
| 90 | Ch p Sample | Chip sampling register |

Table 11- 8 registers are described in detail

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A domain** | **The field name** | **access** | **Reset value** | **instructions** |
| CR00: P yiu nCfg | | | | |
| 15:0 | PC x\_bus\_dev | read-only | L yao \_ad [away] | In PCIX Agent mode, the CPU refers to the total amount used  Line, equipment number |
| " | reserve | read-only | L yao \_ad [or] |  |
| Ephron; | P \_ PC \_c \_ nf g | read-only | PC \_c \_ nf g | PCI\_C - nf G 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 | T ri ans\_l boast is 0 | Read and write | 0 | PCI\_Mem\_L yao 0 window maps address 6 bits higher |
| but | T ri ans\_l group 1 | Read and write | 0 | PCI\_Mem\_L yao 1 window maps address 6 bits higher |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| " | T ri ans\_l group 2 | Read and write | 0 | PCI\_Mem\_L yao 2 window map address 6 bits higher |
| all | reserve | read-only | 0 |  |
| CR14: PCIX\_B ri dge\_Cfg | | | | |
| 5-0 | PC x\_ ri gate | Read and write | 6 'h18 | Number of reads to DDR2 in PCIX mode |
| 6 | PC x\_ ri yao \_en | Read and write | 0 | Whether the PCIX bridge allows write to cross read |
| all | reserve | read-only | 0 |  |
| CR18: PCIMap\_Cfg | | | | |
| 15:0 | Dev\_add ri | Read and write | 0 | PCI configuration reads and writes 16 bits higher than the AD line |
| 16 | C yao nf\_type | Read and write | 0 | Configure the type of reads and writes |
| 31:17 | reserve | read-only | 0 |  |
| CR1C: GPIO\_Data | | | | |
| 15:0 | Gp Yao \_ Yao UT | Read and write | 0 | GPIO outputs data |
| Caused the | Gp yew \_ n | Read and write | 0 | GPIO input data |
| CR20: GPIO\_EN | | | | |
| 15:0 | Gp yew \_en | Read and write | FFFF | High is input, low is 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\_H t \* \* | | | | |
| 0 | reserve | read-only | 0 |  |
| 2:1 | PC \_ mg\_s ze | Read and write | 2 'bl1 | 00:32; 10:64; Others: Invalid |
| 3 | P ri ef\_en | Read and write | 0 | Prefetching can make |
| 4 | reserve | read-only | 0 |  |
| 62:12 | Ba ri \_mask | Read and write | 0 | Window size mask (high 1, low 0) |
| 63 | Bu ri st\_cap | Read and write | 1 | Whether to allow burst transmissions |
| CR68: PXA ri b\_C yao nf g | | | | |
| 0 | Dev ce\_en | Read and write | 1 | External devices allow |
| 1 | D sable\_b ri yao, Ken | Read and write | 0 | Disable damaged master devices |
| 2 | default\_mas\_en | Read and write | 1 | The bus is docked to the default master device  0: Dock to the last master  1: Dock to the default master device |
| o | Default\_maste ri | Read and write | 0 | Bus docking default major number |
| but | Pa ri k\_delay | Read and write | 2 'bl1 | Delay from no device request bus to triggering the docking of default device behavior  00:0 cycle |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  | 01:8 cycles  10:32 cycle  11:128 cycles |
| " | level | Read and write | 8 'h01-2 | Level one equipment |
| Ephron; | ri ude\_dev | Read and write | 0 | Force priority devices  A PCI device corresponding to a bit of 1 can occupy the bus with persistent requests after it has acquired it |
| away | reserve | read-only | 0 |  |
| CR6C: PXA ri b\_Status | | | | |
| away | B ri yao ken\_maste ri | read-only | 0 | Damaged master device (reset when changing disable policy) |
| 10:8 | Last\_maste ri | read-only | 0 | The main device that USES the bus last |
| lustful | reserve | read-only | 0 |  |
| CR80: Ch p C ray NF G (see Section 2.6) | | | | |
| CR90: Ch P Sample (see Section 2.6) | | | | |
| CRA0: Ch P Sample (see section 2.6) | | | | |
| CRB0: PLL C flare NF G (see Section 2.6) | | | | |
| CRC0: PLL C flare NF G (see Section 2.6) | | | | |
| CRD0: yao ri C e C yao nf g (see section 2.6) | | | | |



## Chip configuration register list

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| The Name | ADDR | R/W | Description(NULL means no effect) | The default value |
| CPU\_WIN0\_BASE | 0 x3ff00000 | RW | The base address of CPU window 0 | 0 x0 |
| CPU\_WIN1\_BASE | 0 x3ff00008 | RW | The base address of CPU window 1 | 0 x1000\_0000 |
| CPU\_WIN2\_BASE | 0 x3ff00010 | RW | The base address of CPU window 2 | 0 x1000\_8000\_0000 |
| CPU\_WIN3\_BASE | 0 x3ff00018 | RW | The base address of CPU window 3 | 0 x0 |
| CPU\_WIN4\_BASE | 0 x3ff00020 | RW | The base address of CPU window 4 | 0 x0 |
| CPU\_WIN5\_BASE | 0 x3ff00028 | RW | The base address of CPU window 5 | 0 x0 |
| CPU\_WIN6\_BASE | 0 x3ff00030 | RW | The 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 | 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 | Mask for CPU window 2 | 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 | Mask for CPU window 7 | 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 | The base address of PCI window 0 | 0 x8000\_0000 |
| PCI\_WIN1\_BASE | 0 x3ff00108 | RW | The base address of PCI window 1 | 0 x0 |
| PCI\_WIN2\_BASE | 0 x3ff00110 | RW | The base address of PCI window 2 | 0 x0 |
| PCI\_WIN3\_BASE | 0 x3ff00118 | RW | The base address of PCI window 3 | 0 x0 |
| PCI\_WIN4\_BASE | 0 x3ff00120 | RW | The base address of PCI window 4 | 0 x0 |
| PCI\_WIN5\_BASE | 0 x3ff00128 | RW | The base address of PCI window 5 | 0 x0 |
| PCI\_WIN6\_BASE | 0 x3ff00130 | RW | The base address of PCI window 6 | 0 x0 |
| PCI\_WIN7\_BASE | 0 x3ff00138 | RW | The base address of 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 | Valid, [47:0]: addr | 0 x0 |
| Slock1\_addr | 0 x3ff00208 | RW | Lock address of Lock window No. 1 ([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 | Locking address of Lock window No.3 ([63]: Valid, [47:0]: addr) | 0 x0 |
| Slock0\_mask | 0 x3ff00240 | RW | Lock 0 Window mask ([47:0]: Mask) | 0 x0 |
| Slock1\_mask | 0 x3ff00248 | RW | Lock 1 Window mask ([47:0]: Mask) | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Slock2\_mask | 0 x3ff00250 | RW | Lock 2 Window mask ([47:0]: Mask) | 0 x0 |
| Slock3\_mask | 0 x3ff00258 | RW | Lock 3 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]: barrier enable /barrier break enable | 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 | Counters that are incremented by 1 per 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 no. 0 processor core |  |
| Core0\_IPI\_Enalbe | 0 x3ff01004 | RW | The IPI\_Enalbe register for processor core no. 0 | 0 x0 |
| Core0\_IPI\_Set | 0 x3ff01008 | send | The IPI\_Set register for the no. 0 processor core |  |
| Core0\_IPI\_Clear | 0 x3ff0100c | send | The IPI\_Clear register for the no. 0 processor core |  |
| Core0\_MailBox0 | 0 x3ff01020 | RW | The IPI\_MailBox0 register for the no. 0 processor core | 0 x0 |
| Core0\_MailBox1 | 0 x3ff01028 | RW | The IPI\_MailBox1 register for the no. 0 processor core | 0 x0 |
| Core0\_MailBox2 | 0 x3ff01030 | RW | The IPI\_MailBox2 register for the no. 0 processor core | 0 x0 |
| Core0\_MailBox3 | 0 x3ff01038 | RW | The IPI\_MailBox3 register for processor core no. 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 processor core No. 1 |  |
| Core1\_IPI\_Enalbe | 0 x3ff01104 | RW | The IPI\_Enalbe register for processor core 1 | 0 x0 |
| Core1\_IPI\_Set | 0 x3ff01108 | send | The IPI\_Set register of processor core 1 |  |
| Core1\_IPI\_Clear | 0 x3ff0110c | send | The IPI\_Clear register of processor core no. 1 |  |
| Core1\_MailBox0 | 0 x3ff01120 | RW | The IPI\_MailBox0 register for processor core 1 | 0 x0 |
| Core1\_MailBox1 | 0 x3ff01128 | RW | The IPI\_MailBox1 register of processor core 1 | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Core1\_MailBox2 | 0 x3ff01130 | RW | The IPI\_MailBox2 register of processor core 1 | 0 x0 |
| Core1\_MailBox3 | 0 x3ff01138 | RW | The IPI\_MailBox3 register of processor core 1 | 0 x0 |
| Core1\_int\_interval | 0 x3ff01160 | RW |  |  |
| Core1\_int\_compare | 0 x3ff01168 | RW |  |  |
| Core2\_IPI\_Status | 0 x3ff01200 | RO | IPI\_Status register for no. 2 processor core |  |
| Core2\_IPI\_Enalbe | 0 x3ff01204 | RW | The IPI\_Enalbe register for processor core 2 | 0 x0 |
| Core2\_IPI\_Set | 0 x3ff01208 | send | The IPI\_Set register for processor core 2 |  |
| Core2\_IPI\_Clear | 0 x3ff0120c | send | IPI\_Clear register for no. 2 processor core |  |
| Core2\_MailBox0 | 0 x3ff01220 | RW | The IPI\_MailBox0 register for processor no.2 core | 0 x0 |
| Core2\_MailBox1 | 0 x3ff01228 | RW | The IPI\_MailBox1 register for processor no.2 core | 0 x0 |
| Core2\_MailBox2 | 0 x3ff01230 | RW | The IPI\_MailBox2 register for processor no.2 core | 0 x0 |
| Core2\_MailBox3 | 0 x3ff01238 | RW | The IPI\_MailBox3 register for processor no.2 core | 0 x0 |
| Core2\_int\_interval | 0 x3ff01260 | RW |  |  |
| Core2\_int\_compare | 0 x3ff01268 | RW |  |  |
| Core3\_IPI\_Status | 0 x3ff01300 | RO | IPI\_Status register for no. 3 processor core |  |
| Core3\_IPI\_Enalbe | 0 x3ff01304 | RW | The IPI\_Enalbe register for processor core 3 | 0 x0 |
| Core3\_IPI\_Set | 0 x3ff01308 | send | The IPI\_Set register for processor core 3 |  |
| Core3\_IPI\_Clear | 0 x3ff0130c | send | The IPI\_Clear register of processor core 3 |  |
| Core3\_MailBox0 | 0 x3ff01320 | RW | The IPI\_MailBox0 register for processor core 3 | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Core3\_MailBox1 | 0 x3ff01328 | RW | The IPI\_MailBox1 register for processor core 3 | 0 x0 |
| Core3\_MailBox2 | 0 x3ff01330 | RW | The IPI\_MailBox2 register for processor core 3 | 0 x0 |
| Core3\_MailBox3 | 0 x3ff01338 | RW | The IPI\_MailBox3 register for processor core 3 | 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 | The 32-bit interrupt enabled status register |  |
| Intenset | 0 x3ff01428 | send | The 32-bit setup enable 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 trigger) | 0 x0 |
| CORE0\_INTISR | 0 x3ff01440 | RO | 32-bit interrupt status routed to CORE0 |  |
| CORE1\_INTISR | 0 x3ff01448 | RO | 32-bit interrupt status routed to CORE1 |  |
| CORE2\_INTISR | 0 x3ff01450 | RO | 32-bit interrupt status routed to CORE2 |  |
| CORE3\_INTISR | 0 x3ff01458 | RO | 32-bit interrupt status routed to CORE3 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Thsens\_int\_ctrl\_Hi | 0 x3ff01460 | RW | Temperature sensor high temperature interrupt control register  [7:0] : Hi\_gate0: High temperature threshold, above which interrupt will be generated [8:8] : Hi\_en0: high temperature interrupt enable 0  [11:10] : Hi\_Sel0: Select high temperature interrupt 0 input source of temperature sensor [23:16] : Hi\_gate1: high temperature threshold 1, exceeding which will generate interrupt [24:24] : Hi\_en1: high temperature interrupt enable 1  [27:26] : Hi\_Sel1: select the input source of temperature sensor for HTS interrupt 1 [39:32] : Hi\_gate2: HTS threshold 2, exceeding which will generate interrupt [40:40] : Hi\_en2: HTS interrupt enable 2  [43:42] : Hi\_Sel2: Select the temperature sensor input source of HTS interrupt 2 [55:48] : Hi\_gate3: HTS threshold 3, exceeding which will generate interrupt [56:56] : Hi\_en3: HTS interrupt enable 3  [59:58] : Hi\_Sel3: Select the input source of temperature sensor for high-temperature interrupt 3 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Thsens\_int\_ctrl\_Lo | 0 x3ff01468 | RW | Temperature sensor low temperature interrupt control register  [7:0] : Lo\_gate0: low temperature threshold below which interrupts will be generated [8:8] : Lo\_en0: low temperature interrupt enabled 0  [11:10] : Lo\_Sel0: Select the temperature sensor input source of low temperature interrupt 0 [23:16] : Lo\_gate1: low temperature threshold 1, below which an interrupt will occur [24:24] : Lo\_en1: low temperature interrupt enabling 1  [27:26] : Lo\_Sel1: select the temperature sensor input source of low temperature interrupt 1 [39:32] : Lo\_gate2: low temperature threshold 2, below which an interrupt [40:40] : Lo\_en2: low temperature interrupt enabling 2  [43:42] : Lo\_Sel2: Select the temperature sensor input source of low temperature interrupt 2 [55:48] : Lo\_gate3: low temperature threshold 3, below which an interrupt will occur [56:56] : Lo\_en3: low temperature interrupt enabling 3  [59:58] : Lo\_Sel3: Select the temperature sensor input source of low temperature interrupt 3 |  |
| Thsens\_int\_status/CLR | 0 x3ff01470 | RW | Interrupt status register, write any value clear interrupt [0] : high temperature interrupt trigger  [1] : Low temperature interrupt trigger |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Thsens\_freq\_scale | 0 x3ff01480 | RW | Temperature sensor high temperature frequency drop control register, four sets of setting priority from high to low [7:0] : Scale\_gate0: high temperature threshold value 0, beyond this temperature will reduce frequency [8:8] : Scale\_en0: high temperature drop frequency enable 0  [11:10] : Scale\_Sel0: Temperature sensor input source [14:12] : Scale\_freq0: Frequency dividing value when reducing frequency [23:16] : Scale\_gate1: high temperature threshold value 1, exceeding which will reduce frequency [24:24] : Scale\_en1: high temperature reducing frequency enable 1  [27:26] : Scale\_Sel1: Temperature sensor input source [30:28] : Scale\_freq1: Frequency partition value for frequency reduction [399:32] : Scale\_gate2: high temperature threshold value 2, above which the frequency will be reduced [40:40] : Scale\_en2: high temperature frequency reduction enable 2  [43:42] : Scale\_Sel2: Temperature sensor input source [46:44] : Scale\_freq2: Frequency partition value [55:48] : Scale\_gate3: high temperature threshold value 3, above which the high temperature will be reduced [56:56] : Scale\_en3: High temperature reduced frequency enable 3  [59:58] : Scale\_Sel3: Select the input source of temperature sensor with high temperature drop frequency 3  [62:60] : Scale\_freq3: The frequency division value when the frequency is reduced |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| DFD\_PARAM | 0 x3ff01500 | RW | Debug trigger condition enablement  [7:0] : Timer, trigger delay. Set as 1 means that it will be triggered immediately when the condition is satisfied, set as 0 means that it will be forbidden to trigger, and set as other values means that the number of beats delayed to be triggered after the condition is satisfied +1  [15:8] : Trigger\_en, trigger condition enablement, corresponding to the external enablement of the eight trigger events |  |
| DFD\_TRIGGER | 0 x3ff01508 | send | Software trigger, send a write operation to this address, will cause a software trigger condition, make in timer-1  Trigger after beat |  |
| CORE0\_AWCOND0 | 0 x3ff01800 | RW | CORE0's AW trigger condition 0 setting [15:0] : AWId  [19:16] awlen [22:20] : Awsize [24:23] : Awburst [26:25] : 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 is set to 0, with the highest bit being AW channel trigger enable [49:0] : awmask  [62] : AWDATa\_en: When both the wDATA trigger condition of WID are met, it is allowed to trigger [63] : awchannel\_en: trigger condition enabled  Trigger condition is  (AW\_IN & AWMASK) = (AWCOND & AWMASK) |  |
| CORE0\_AWCOND1 | 0 x3ff01810 | RW | The trigger condition of AW must be COND0 and COND1 simultaneously  [47:0] : awaddr |  |
| CORE0\_AWMASK1 | 0 x3ff01818 | RW |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CORE0\_ARCOND0 | 0 x3ff01820 | RW | The AXI interface AR trigger condition of CORE0 is similar to AW [15:0] : Arid  [19:16] : Arlen [22:20] : Arburst [26:25] : Arlock [30:27] : Arcache [33:31] : Arprot [37:34] : ArcMD [47:38] : Arcpuno [48] : Arvalid  [49] : arready |  |
| CORE0\_ARMASK0 | 0 x3ff01828 | RW | CORE0's AXI interface AR trigger enable is set to 0, and the highest bit is AR channel trigger enable [49:0] : Armask   1. Ardata\_en: The trigger is allowed only when the trigger condition is met with the Rdata trigger condition of rid 2. : Archannel\_en: Trigger condition enablement |  |
| CORE0\_ARCOND1 | 0 x3ff01830 | RW | [47:0] : araddr |  |
| CORE0\_ARMASK1 | 0 x3ff01838 | RW |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CORE0\_WCOND0 | 0 x3ff01840 | RW | CORE0's AXI interface W trigger condition is 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 is set to 0, with the highest bit being W channel trigger enable [49:0] : Wmask  [63] : Wchannel\_en: Trigger condition enable, no need to 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's AXI interface B trigger condition, similar to AW [15:0] : BID  [was] : bresp  [18] : Bvalid [19] : bready |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CORE0\_BMASK0 | 0 x3ff01878 | RW | CORE0's AXI interface B trigger enable is set to 0, with the highest bit being the B-channel trigger enable [19:0] : bmask  [63] : bchannel\_en |  |
| CORE0\_RCOND0 | 0 x3ff01880 | RW | CORE0's 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 is set to 0, 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 configures register 0 [47:0] : count\_target  [55:48] : monitor\_enable |  |
| TUD0\_CONF1 | 0 x3ff018e8 | RW | TUD0 collocation 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 is set to 0, and the highest bit is AW channel trigger enable trigger condition is  (AW\_IN & AWMASK) = (AWCOND & AWMASK) |  |
| CORE1\_AWCOND1 | 0 x3ff01910 | RW | The trigger condition of AW must be COND0 and COND1 simultaneously |  |
| CORE1\_AWMASK1 | 0 x3ff01918 | RW |  |  |
| CORE1\_ARCOND0 | 0 x3ff01920 | RW | The CORE1 interface AR trigger condition is similar to AW |  |
| CORE1\_ARMASK0 | 0 x3ff01928 | RW |  |  |
| CORE1\_ARCOND1 | 0 x3ff01930 | RW |  |  |
| CORE1\_ARMASK1 | 0 x3ff01938 | RW |  |  |
| CORE1\_WCOND0 | 0 x3ff01940 | RW | The CORE1 interface W trigger condition is 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's interface B trigger condition is similar to AW |  |
| CORE1\_BMASK0 | 0 x3ff01978 | RW |  |  |
| CORE1\_RCOND0 | 0 x3ff01980 | RW | CORE1's interface R trigger condition is 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 configured register 0  [47:0] : count\_target [55:48] : monitor\_enable |  |
| TUD1\_CONF1 | 0 x3ff019e8 | RW | TUD0 collocation 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 is set to 0, and the highest bit is AW channel trigger enable trigger condition is  (AW\_IN & AWMASK) = (AWCOND & AWMASK) |  |
| CORE2\_AWCOND1 | 0 x3ff01a10 | RW | The trigger condition of AW must be COND0 and COND1 simultaneously |  |
| CORE2\_AWMASK1 | 0 x3ff01a18 | RW |  |  |
| CORE2\_ARCOND0 | 0 x3ff01a20 | RW | The CORE2 interface AR trigger condition is similar to AW |  |
| CORE2\_ARMASK0 | 0 x3ff01a28 | RW |  |  |
| CORE2\_ARCOND1 | 0 x3ff01a30 | RW |  |  |
| CORE2\_ARMASK1 | 0 x3ff01a38 | RW |  |  |
| CORE2\_WCOND0 | 0 x3ff01a40 | RW | The CORE2 interface W trigger condition is 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's interface B trigger condition is similar to AW |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CORE2\_BMASK0 | 0 x3ff01a78 | RW |  |  |
| CORE2\_RCOND0 | 0 x3ff01a80 | RW | CORE2's interface R trigger condition is 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 consigns register 0  [47:0] : count\_target [55:48] : monitor\_enable |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TUD2\_CONF1 | 0 x3ff01ae8 | RW | TUD0 collocation 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 is set to 0, and the highest bit is AW channel trigger enable trigger condition is  (AW\_IN & AWMASK) = (AWCOND & AWMASK) |  |
| CORE3\_AWCOND1 | 0 x3ff01b10 | RW | The trigger condition of AW must be COND0 and COND1 simultaneously |  |
| CORE3\_AWMASK1 | 0 x3ff01b18 | RW |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CORE3\_ARCOND0 | 0 x3ff01b20 | RW | The CORE3 interface AR trigger condition is similar to AW |  |
| CORE3\_ARMASK0 | 0 x3ff01b28 | RW |  |  |
| CORE3\_ARCOND1 | 0 x3ff01b30 | RW |  |  |
| CORE3\_ARMASK1 | 0 x3ff01b38 | RW |  |  |
| CORE3\_WCOND0 | 0 x3ff01b40 | RW | The CORE3 interface W trigger condition is 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's interface B trigger condition is similar to AW |  |
| CORE3\_BMASK0 | 0 x3ff01b78 | RW |  |  |
| CORE3\_RCOND0 | 0 x3ff01b80 | RW | CORE3's interface R trigger condition is 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 configures register 0 [47:0] : count\_target  [55:48] : monitor\_enable |  |
| TUD3\_CONF1 | 0 x3ff01be8 | RW | TUD0 collocation 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 configures register 0 [47:0] : count\_target  [55:48] : monitor\_enable |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TUD4\_CONF1 | 0 x3ff01ce8 | RW | TUD4 equipped register 1 [2:0] : DCDL\_sel\_signal [5:3] : DCDL\_sel\_clock [8:6] : Signal\_sel [11:9] : clock\_sel [18:12] : Reading\_sel [19] : counter\_clock\_sel [20] : sticky [21] : reset\_g [22] : stop   1. : start 2. : cg\_en |  |
| TUD4\_RESULT | 0 x3ff01cf0 | R | TUD4 result register |  |
|  |  |  |  |  |
| TUD5\_CONF0 | 0 x3ff01de0 | RW | TUD5 configures register 0 [47:0] : count\_target  [55:48] : monitor\_enable |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TUD5\_CONF1 | 0 x3ff01de8 | RW | TUD5 equipped register 1 [2:0] : DCDL\_sel\_signal [5:3] : DCDL\_sel\_clock [8:6] : Signal\_sel [11:9] : clock\_sel [18:12] : Reading\_sel [19] : counter\_clock\_sel [20] : sticky [21] : reset\_g [22] : stop   1. : start 2. : cg\_en |  |
| TUD5\_RESULT | 0 x3ff01df0 | R | TUD5 result register |  |
|  |  |  |  |  |
| HT0\_AWCOND0 | 0 x3ff01e00 | RW | HT0's AXI interface AW triggers the condition 0 setting |  |
| HT0\_AWMASK0 | 0 x3ff01e08 | RW | The AXI interface AW trigger enable of HT0 is set to 0, and the highest bit is AW channel trigger enable condition is  (AW\_IN & AWMASK) = (AWCOND & AWMASK) |  |
| HT0\_AWCOND1 | 0 x3ff01e10 | RW | The trigger condition of AW must be COND0 and COND1 simultaneously |  |
| HT0\_AWMASK1 | 0 x3ff01e18 | RW |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| HT0\_ARCOND0 | 0 x3ff01e20 | RW | HT0's AXI interface AR trigger condition is similar to AW |  |
| HT0\_ARMASK0 | 0 x3ff01e28 | RW |  |  |
| HT0\_ARCOND1 | 0 x3ff01e30 | RW |  |  |
| HT0\_ARMASK1 | 0 x3ff01e38 | RW |  |  |
| HT0\_WCOND0 | 0 x3ff01e40 | RW | HT0's AXI interface W trigger condition is 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's AXI interface B trigger condition is 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 | The AXI interface for HT1 triggers the condition 0 setting |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| HT1\_AWMASK0 | 0 x3ff01f08 | RW | The AXI interface AW trigger enable of HT1 is set to 0, and the highest bit is AW channel trigger enable condition is  (AW\_IN & AWMASK) = (AWCOND & AWMASK) |  |
| HT1\_AWCOND1 | 0 x3ff01f10 | RW | The trigger condition of AW must be COND0 and COND1 simultaneously |  |
| HT1\_AWMASK1 | 0 x3ff01f18 | RW |  |  |
| HT1\_ARCOND0 | 0 x3ff01f20 | RW | HT1's AXI interface AR trigger condition is similar to AW |  |
| HT1\_ARMASK0 | 0 x3ff01f28 | RW |  |  |
| HT1\_ARCOND1 | 0 x3ff01f30 | RW |  |  |
| HT1\_ARMASK1 | 0 x3ff01f38 | RW |  |  |
| HT1\_WCOND0 | 0 x3ff01f40 | RW | HT1's AXI interface W trigger condition is 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's AXI interface B trigger condition is 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 | One level cross-switch address window | 0 x0 |
| CORE0\_WIN1\_BASE | 0 x3ff02008 | RW | One level cross-switch address window | 0 x0 |
| CORE0\_WIN2\_BASE | 0 x3ff02010 | RW | One level cross-switch address window | 0 x0 |
| CORE0\_WIN3\_BASE | 0 x3ff02018 | RW | One level cross-switch address window | 0 x0 |
| CORE0\_WIN4\_BASE | 0 x3ff02020 | RW | One level cross-switch address window | 0 x0 |
| CORE0\_WIN5\_BASE | 0 x3ff02028 | RW | One level cross-switch address window | 0 x0 |
| CORE0\_WIN6\_BASE | 0 x3ff02030 | RW | One level cross-switch address window | 0 x0 |
| CORE0\_WIN7\_BASE | 0 x3ff02038 | RW | One level cross-switch address window | 0 x0 |
| CORE0\_WIN0\_MASK | 0 x3ff02040 | RW | One level cross-switch address window | 0 x0 |
| CORE0\_WIN1\_MASK | 0 x3ff02048 | RW | One level cross-switch address window | 0 x0 |
| CORE0\_WIN2\_MASK | 0 x3ff02050 | RW | One level cross-switch address window | 0 x0 |
| CORE0\_WIN3\_MASK | 0 x3ff02058 | RW | One level cross-switch address window | 0 x0 |
| CORE0\_WIN4\_MASK | 0 x3ff02060 | RW | One level cross-switch address window | 0 x0 |
| CORE0\_WIN5\_MASK | 0 x3ff02068 | RW | One level cross-switch address window | 0 x0 |
| CORE0\_WIN6\_MASK | 0 x3ff02070 | RW | One level cross-switch address window | 0 x0 |

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

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

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

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

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

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SOUTH\_WIN1\_MASK | 0 x3ff02548 | RW | One level cross-switch address window | 0 x0 |
| SOUTH\_WIN2\_MASK | 0 x3ff02550 | RW | One level cross-switch address window | 0 x0 |
| SOUTH\_WIN3\_MASK | 0 x3ff02558 | RW | One level cross-switch address window | 0 x0 |
| SOUTH\_WIN4\_MASK | 0 x3ff02560 | RW | One level cross-switch address window | 0 x0 |
| SOUTH\_WIN5\_MASK | 0 x3ff02568 | RW | One level cross-switch address window | 0 x0 |
| SOUTH\_WIN6\_MASK | 0 x3ff02570 | RW | One level cross-switch address window | 0 x0 |
| SOUTH\_WIN7\_MASK | 0 x3ff02578 | RW | One level cross-switch address window | 0 x0 |
| SOUTH\_WIN0\_MMAP | 0 x3ff02580 | RW | One level cross-switch address window | 0 x0 |
| SOUTH\_WIN1\_MMAP | 0 x3ff02588 | RW | One level cross-switch address window | 0 x0 |
| SOUTH\_WIN2\_MMAP | 0 x3ff02590 | RW | One level cross-switch address window | 0 x0 |
| SOUTH\_WIN3\_MMAP | 0 x3ff02598 | RW | One level cross-switch address window | 0 x0 |
| SOUTH\_WIN4\_MMAP | 0 x3ff025a0 | RW | One level cross-switch address window | 0 x0 |
| SOUTH\_WIN5\_MMAP | 0 x3ff025a8 | RW | One level cross-switch address window | 0 x0 |
| SOUTH\_WIN6\_MMAP | 0 x3ff025b0 | RW | One level cross-switch address window | 0 x0 |
| SOUTH\_WIN7\_MMAP | 0 x3ff025b8 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN0\_BASE | 0 x3ff02600 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN1\_BASE | 0 x3ff02608 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN2\_BASE | 0 x3ff02610 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN3\_BASE | 0 x3ff02618 | RW | One level cross-switch address window | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| WEST\_WIN4\_BASE | 0 x3ff02620 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN5\_BASE | 0 x3ff02628 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN6\_BASE | 0 x3ff02630 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN7\_BASE | 0 x3ff02638 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN0\_MASK | 0 x3ff02640 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN1\_MASK | 0 x3ff02648 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN2\_MASK | 0 x3ff02650 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN3\_MASK | 0 x3ff02658 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN4\_MASK | 0 x3ff02660 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN5\_MASK | 0 x3ff02668 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN6\_MASK | 0 x3ff02670 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN7\_MASK | 0 x3ff02678 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN0\_MMAP | 0 x3ff02680 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN1\_MMAP | 0 x3ff02688 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN2\_MMAP | 0 x3ff02690 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN3\_MMAP | 0 x3ff02698 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN4\_MMAP | 0 x3ff026a0 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN5\_MMAP | 0 x3ff026a8 | RW | One level cross-switch address window | 0 x0 |
| WEST\_WIN6\_MMAP | 0 x3ff026b0 | RW | One level cross-switch address window | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| WEST\_WIN7\_MMAP | 0 x3ff026b8 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN0\_BASE | 0 x3ff02700 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN1\_BASE | 0 x3ff02708 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN2\_BASE | 0 x3ff02710 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN3\_BASE | 0 x3ff02718 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN4\_BASE | 0 x3ff02720 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN5\_BASE | 0 x3ff02728 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN6\_BASE | 0 x3ff02730 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN7\_BASE | 0 x3ff02738 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN0\_MASK | 0 x3ff02740 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN1\_MASK | 0 x3ff02748 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN2\_MASK | 0 x3ff02750 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN3\_MASK | 0 x3ff02758 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN4\_MASK | 0 x3ff02760 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN5\_MASK | 0 x3ff02768 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN6\_MASK | 0 x3ff02770 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN7\_MASK | 0 x3ff02778 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN0\_MMAP | 0 x3ff02780 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN1\_MMAP | 0 x3ff02788 | RW | One level cross-switch address window | 0 x0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| NORTH\_WIN2\_MMAP | 0 x3ff02790 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN3\_MMAP | 0 x3ff02798 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN4\_MMAP | 0 x3ff027a0 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN5\_MMAP | 0 x3ff027a8 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN6\_MMAP | 0 x3ff027b0 | RW | One level cross-switch address window | 0 x0 |
| NORTH\_WIN7\_MMAP | 0 x3ff027b8 | RW | One level cross-switch address window | 0 x0 |



## Software and Hardware Design Guide

The pin of the 3A300/3B3000 processor is compatible with the 3A1000 processor, but the corresponding hardware and software should be changed to enable the original compatibility mode, or some new features of the 3A300/3B3000 processor should be opened. This chapter focuses on the difference of hardware and software Settings in the use of the 3A300/3B3000 processor compared with the 3A1000/2000 processor.

#### Hardware Change Guide

1. The original CORE\_PLL\_AVDD and DDR\_PLL\_AVDD (2.5V) are now 1.8V. If you use the original 3A1000 motherboard, you need to change the two power sources from 2.5V to 1.8V. These pins (including HT0/1\_PLL\_AVDD) on the 3A2000/3B2000 are NC. For compatibility with the 3A2000/3B2000, these power voltages can be modified to 1.8V or a 1.8V / 2.5V configurable design.
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;

(Consistent with 3A2000)

1. The original HT0/1\_pll\_ref has been changed to NC PIN. If the original 3A motherboard is used, no modification can be made; (with 3 a2000

Consistent)

1. Change Mc0/1\_comp\_ref\_gnd to Mc0/1\_a15. If the original 3A motherboard is used, no modification can be made; But if you connect to a memory stick, you can support more memory; (Consistent with 3A2000)
2. The function controlled by PCI\_CONFIG[0] has been changed to SPI startup enable, which can be started from SPI FLASH after setting to 1. If the original 3A motherboard is used, it needs to be set to 0 and start from LPC FLASH; If the motherboard already has SPI FLASH, you can connect GPIO[0] as SPI\_CS, and set PCI\_CONFIG[0] to 1 to start from SPI FLASH. (Consistent with 3A2000)
3. The function controlled by PCI\_CONFIG[7] was changed to force HT1.0 mode, and HT started directly in 1.0 mode after setting to 1. If you use 3A780E motherboard, you need to set it to 1 at present. If using 3A2H motherboard, no special setting is required; (Consistent with 3A2000)

Reconfigure the frequency using software in PMON;

1. CLKSEL[9:5] needs to be set to 5 'b01111; Use PMON for memory frequency Settings. For longson 3A3000/3B3000A/B/C, the NODE clock must be used in the frequency configuration of PMON. (Consistent with 3A2000)
2. CLKSEL[4:0] needs to be set to 5 'b01111; Use PMON for processor core frequency Settings. For the loong chip 3A300/3B3000A /B/C, L2 PLL must be used as the master clock in the frequency configuration of PMON. (Consistent with 3A2000)
3. For 3A2H motherboard, the pull up resistance on HT0/1\_POWEROK and HT0/1\_resetn should be removed. (The original pull-up resistance of 300 ohms is not suitable for 3A, which can also be removed) (consistent with 3A2000)

#### Description of frequency setting

In order to be basically compatible with the frequency configuration of the godson 3A1000, the hardware frequency configuration range of the Godson 3A300/3B3000 is relatively narrow. In order to obtain a wider frequency range and better clock quality, the software configuration method in the Godson 3A300/3B3000 is mainly used in the PMON, and the configuration method is the same as the godson 3B1500. Please refer to the PMON source code for specific configuration methods.

1. The frequency setting is completely set by software, and CLKSEL is not modified when changing the frequency.
2. Stable operating frequency of 1.25V core voltage: processor core frequency is set at 1400MHz, memory frequency is set at 700MHz, HT controller is set at 800MHz, HT bus is set at 800MHz/1600MHz.
3. For loongson 3A3000/3b3000A /B/C, the NODE CLOCK must use L2 PLL as the primary CLOCK and DDR CLOCK

The NODE clock must be used for frequency division;

#### PMON change guide

The following PMON changes are basically the same as the 3A2000/3B2000 processor.

Since the processor core, memory controller and HT controller have been upgraded to different levels of cross-switches, PMON needs to make some changes compared with the godson 3A1000, mainly including the following necessary parts:

1. Initialize L1 Dcache, L1 Icache, Vcache and L2 Cache after power is removed (hardware is completed);
2. After the CPU is powered on, close the Store Fill Buffer of all cores.
3. After the CPU is powered on, turn off the word write merging function of all cores;
4. If you want to maintain compatibility with 3A5, set the PRID hidden bit in the CP0 Diag register for all cores;
5. In all the assembly code jr RX, RX is not register 31 statement modified to JR $31;
6. Use code similar to 3B1500 to configure processor core, memory, and node PLL;
7. Code using memory controller configuration and parameter training similar to 3B1500;
8. If HT works in 1.0 mode, HT can only work in 8-bit mode.
9. If the SPI controller is used, the base address is changed from 0xBFE001F0 to 0xBFE00220;

In addition to these required changes, you can make the following changes to enhance PMON functionality:

1. Modify the delay of the buzzer to ensure that the user can hear the buzzer;
2. Add support for turning off defective nuclear clocks;
3. Remove part of the workaround of 3A5 to the 2H bridge HT controller from the code (leaving part of the workaround);

#### Kernel modification guide

The following kernel changes are basically the same as 3A2000/3B2000, but you need to add pairs to the corresponding parts of the kernel

3A3000/3B3000 processor support.

1. Modify the Cache description structure in the kernel. VCache and SCache are connected by a 16-way group. (Consistent with 3A2000)
2. Modify the calculation method of temperature sensor, the algorithm is: node temperature =Thens\_out \*731/ 0x4000-273;
3. Modify the configuration register address when the core; (Consistent with 3A2000)
4. Change "brush ICache/DCache" to "brush ICache/DCache/VCache". (Consistent with 3A2000)
5. If the SPI controller is used, the base address is changed from 0xBFE001F0 to 0xBFE00220; (Consistent with 3A2000)
6. Must use Uncache DMA, use software to maintain the consistency of the Cache data; (Consistent with 3A2000)
7. Add support for Store Fill Buffer: First, a SYNC should be added before all Uncache requests to ensure that all contents in Store Fill Buffer have been written back into Cache when Uncache requests occur; Second, all unlocking operations Shared among different cores need to be implemented using LL/SC instructions. (Consistent with 3A2000)
8. The MSI functionality of the device is not used. When MSI function must be used, the number of data receiving buffers of HT controller's POST channel should be set to 1, and HT bus should be reconnected. (Consistent with 3A2000)
9. Lock Cache operations cannot be used for DMA areas where hardware automatically maintains consistency. (Consistent with 3A2000)

Other modifications that can be made to improve performance are:

1. Increased support for FTLB; (Consistent with 3A2000)
2. Increased support for TLB rapid refill; (Consistent with 3A2000)
3. Add support for wait instruction; (Consistent with 3A2000)
4. Increase prefetch instruction support; (Consistent with 3A2000)
5. Use DI/EI for interrupt return. However, it should be noted that the [31:4] returned by EI instruction is a random value, which is different from the MIPS requirement. (Consistent with 3A2000)