# IDI / CMI capability in Gemini

**Disclaimer**: This chapter provides preliminary information for IDI and CMI protocol support of CFG NOC. All features listed here are still under development and it may change without prior notice. Please contact CFG support for further details.

Terminology:

* Master bridge, is a module sitting between the NOC and an external host master (e.g. CPU).
* Slave bridge, is a module sitting between the NOC and an external host slave (e.g. DDR controller)
* NOC agent, developed by CFG, is a module sitting outside the NOC functionally but instantiated as part of NocStudio generated RTL (e.g. CCC, IDI2CMI, etc.).

## System view of a IDI/CMI based design

The figure below captured a simplified block diagram to illustrate the NOC bridges, agents and their connectivity of an IDI/CMI/CCC based design. In additions to master / slave bridges, the NOC agents are CCC (Cache Coherency Controller) and an IDI2CMI converter.

NOC

CCF

CMI  
slave

IDI  
master

idim

idis

CCC

cmim

cmis

idis

idim

cmim

Idi2  
Cmi

idis

cmim

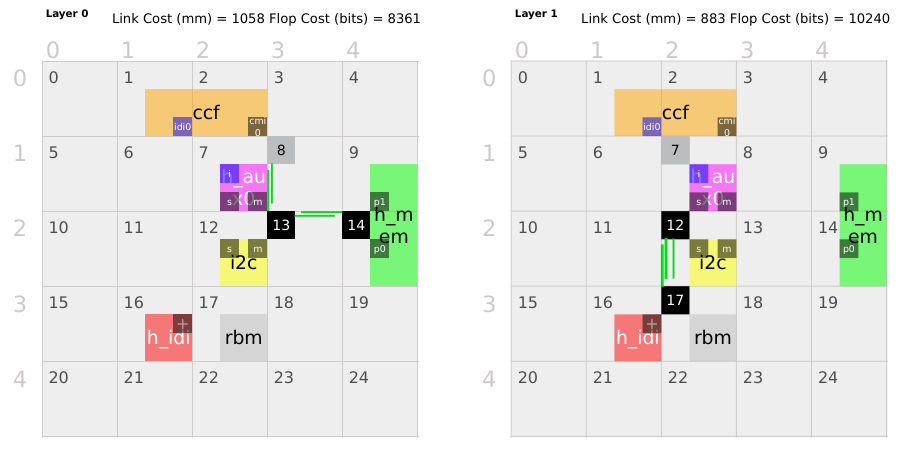
Figure 1 Illustration of IDI/CMI based system

The IDI (or INI) master connects to the NOC via an idim – IDI master bridge. The cmis (CMI Slave Bridge) sits between the NOC and the CMI slave or memory controller. The CCF, as an example, connects with the NOC in two ways: (p1) an idis (IDI slave bridge) communicating with CCC and (p2) a cmim (CMI master bridge) for meory access. It uses p1 to receive IDI messages from CCC and issues memory access through p2. The IDI2CMI conversion block manages the non-coherent communication between IDI/INI and CMI fabric.

SOC designer can use add\_traffic\_b with abstracted command “rd” or “wr” to define the traffic profile and bandwidth requirement between a CMI master and a CMI slave. To specify the traffic between IDI and CMI using the idi2cmi\_converter, a new option “midpoint” has been added to add\_traffic\_b support.

The CMI slave region or at least part of it should be defined as coherent region by including the ccc\_group. Please refer to add\_ccc\_group in NocStudio help manual if you are not familiar with it. The IDI slave (connected with CCF) also needs to be mapped as coherent region and setup\_coherency command should take care of the rest.

Due to NOC cell size differences, currently NocStudio will map IDI and CMI traffic flows on differnet layers. Illustrated below, IDI traffic are mapped on Layer 1 and CMI are on Layer 0. Optimization will be done in future releases. The CCC, shown in color magenta, has a direct connection to the CCF, hence NocStudio optimizes thes router (shown in gray) and bridges away (shonw in blue) to minimize latency and cost.



## IDI Bridge Microarchitecture

### IDI Master Bridge

Figure 96 shows the block diagram of the IDI master bridge. It consists of IDI protocol converter and the switch module that communicates between the converter and routers, up to 16 layers.



Figure 2 IDI Master Bridge

The IDI master bridge has been developed based on IDI specification version 2.02 draft A published by Eric Hallnor on Oct 10th, 2018 which can be found on SharePoint: <https://sharepoint.amr.ith.intel.com/sites/mdgarchtech/techforums/idi/default.aspx>

#### IDI Channels and NoC

The master bridge provides an IDI host (coherent master) connectivity to the CFG NOC. All 5 channels are supported on both directions (c2u and u2c). Three main channels are request, response and data, two side channels are credit return and hardware straps.

The channels have the naming prefix as shown below where <host> and <bridge> are user defined names in NocStudio design script file.

IDI host 🡺 NOC

<host>\_<bridge>\_*c2u*\_<channel> for the data path direction

<host>\_<bridge>\_*u2c*\_credit\_ret\_<channel> for the credit return path

NOC 🡺 IDI host

<host>\_<bridge>\_*u2c*\_<channel> for the data path direction

<host>\_<bridge>\_*c2u*\_credit\_ret\_<channel> for the credit return path

Up to 16 NoC layers with 4 virtual channels each can be configured based on system requirement and performance goal. The mapping of these resource are determined by NocStudio or can be manually override by the user.

#### Decoding and Routing

The C2U request address is used to decode the destined slave port. The NoC traffic class and address hashing can also be used to determine the physical route, virtual channel, and NoC layer for the transport. The C2U data and response channels use UQId to decide and route transactions to the destined port.

A slave device is identified by address ranges specified in either base/mask or lo-hi format in NocStudio. The number of address ranges supported by IDI master bridge is the same across all CFG product families and it’s 256. The same programmability is supported and regbus layer must be enabled to support such capability. Multiple address ranges can be specified for a single slave and these can have different access privileges. Please note that increasing address ranges supported by a given master bridge reduces its highest achievable target frequency and it is a system level design tradeoff.

In typical system most address ranges are enabled by default while some may be programmable and can be enabled/disabled at runtime. During address decode phase, the *opcode* is used to check against its coherency type. A coherent transaction access to non-coherent address range will result in a decode error response.

Each address range can also be associated with hash functions which are used in the destination/route lookup process. Address ranges can also be defined with foreground (high priority) or background (low priority), this allows NOC designer to define overlapping address ranges and reduce total number of address ranges in address decoder table entry. In addition, address relocation is supported for an IDI master bridge to alter the system address in the packet before injecting it into the NOC. Please refer to Address Relocation and Hash Function section for details.

#### Flow control, Error and Stall

Both credit-based and stall-based flow control (indicated by strap) are supported based on IDI specification. On NOC router links, only credit based flow control is used. An unknown address or unsupported opcode on the C2U channel will trigger a decode error. Temporary command channel stall may happen if u2c response credit is not available. As mentioned earlier, coherent transactions to a non-coherent address range can trigger a decode error as well.

### IDI Slave Bridge

Figure 97 shows a block diagram of the IDI slave bridge. Similar to the master bridge, it consists of the protocol converter and the switch module for the bridge to communicate with the NOC layers / routers.



Figure 3 IDI Slave Bridge

The main features of the IDI slave bridge are listed in the following sections.

#### IDI Channels and NoC

The slave bridge provides an IDI connectivity for NOC agent (i.e. CCC, Cache Coherency Controller) and CCF in multi-level coherency applications. All 5 channels are supported on both directions (c2u and u2c). Three main channels are request, response and data; two side channels are credit return and hardware straps. Similar to all CFG protocol bridges, up to 16 NoC layers with 4 virtual channels each can be configured based on system requirement and performance goal. The mapping of these resource are determined by NocStudio or can be manually override by the user.

#### Decoding, Routing, and Flow Control

The IDI slave bridge requires additional signaling shown below for communication inside the NOC.

|  |  |  |  |
| --- | --- | --- | --- |
| Channel | Data Width | Signal Name | Description |
| C2U Request | P\_AGNID\_WIDTH | Srcid | Indicates the source of the request going to the IDI slave. The IDI protocol doesn’t send the CPU node ID with the request, so the NoC will attach it before sending it on. |
| U2C Request | P\_AGNID\_WIDTH | Tgtid | This is the target nodeID of the master that IDI slave is sending the request to. For example, a snoop would be targeted at a specific CPU. |
| U2C Response | P\_AGNID\_WIDTH | Tgtid | This is the target nodeID of the master that IDI slave is sending the request to. |
| U2C Response | 13 | UQID | This field is to attach an UQid value to a GO message, so the NOC can return a packet when the GO message has reached an ordering point where snoops to that address can be sent again. While RspData field carries UQid for some requests, it doesn’t for GO messages. In some packets this field will be redundant with RspData. |
| U2C Data | P\_AGNID\_WIDTH | Tgtid | This is the target nodeID of the master that IDI slave is sending the request to. |

Table 1 IDI Slave Bridge Additional Requirements

### NOC implementation on GO (Global Observation) message

The IDI specification defines interconnect ordering rule where go and snoop messages with the same address must be ordered in the fabric. Internally the NOC has implemented a special GO-ACK message between the master bridge and the CCC by borrowing an unused C2U response opcode (5’b10000).

Upon sending the GO message (initiated by the CCC) to the IDI master, the master bridge sets the C2U response opcode to 5’b10000 to acknowledge the GO message. The CCC can then issue subsequent snoop message as the GO has reached its serialization point in the system.

## CMI Bridge Microarchitecture

Key features:

* CMI specification version 1.1 compliance.
* Opcode supported: MRd, MWr, MWrPtl, MPCmt & NDTC.
  + Drop error type and hence read retry is not supported.
* Number of Virtual Channels in NocStudio: 1, 2-4
* Configurable credit FIFO depths.
* Transfer Size supported: 32/64 Bytes
* Interface widths supported: 8 / 16 / 32 / 64 Bytes
* Configurable pipeline delay between the header and the write data (on CMI interface, fixed delays only)
* Parity generation on transmitting CMI interface.
* Parity check on receiving req\_address field on CMI Master bridge.
* Read data de-interleaving.

### CMI Master Bridge

Figure 98 shows a block diagram of the CMI master bridge.



Figure 4 CMI Master Bridge

The main features of the CMI master bridge are listed in the following sections.

#### CMI Channels and NoC

All CMI channels are supported. The read and write requests which share the same CMI virtual channel may be sent on separate NoC layer or virtual channels on the same layer. The write requests share the same NoC channel as *reqdata* (Write Data). Our CMI fabric is un-ordered.

#### Decoding and Routing

NocStudio configures CMI master bridge to use either address- or id- based lookup as its primary look-up. It is then followed by address based hashing, where the hash function is defined by the user in NocStudio. The read and write requests can take a different route path within the NoC. Requests may also take a different route based on different CMI virtual channel.

#### Flow control

The CMI master bridge implementation is based on CMI specification version 1.1. The PDF can be downloaded from SharePoint link shown below: <https://sharepoint.amr.ith.intel.com/sites/Arch_Convergence/Interfaces/CMI/CMI/Meeting%20Docs/CMI%20Spec%20rev%201.1.pdf>. Please refer to section 4.2 for various timing diagrams and description. The bridge doesn’t publish any fabric credits for request but does honor fabric credits for the response / read completion channel, originated by the CMI master host.

The bridge property cmi\_bypass\_register\_config can be set to yes to bypass credit initialization after reset de-assertion. This allows the bridge to be operational immediately coming out of reset. The CMI master bridge has bridge properties to configure VC credit FIFO depths for read and write requests. There are programmable registers by which user can control the usable VC credit FIFO depth within the value defined by the bridge property.

#### Errors and Stalls

CMI Master bridge can raise an interrupt if

the user tries to program the fifo depth register to a value higher than that of available hardware.

the incoming transaction has an address decode error or address parity error (CMI spec doesn’t allow the Master bridge to inform the Master agent of such errors on a CMI interface)

Incoming transaction uses an unsupported opcode.

The *rsp\_rd\_cpl\_stall* triggered by CMI master is honoured by the master bridge. The bridge would acknowledge it by asserting rsp\_rd\_cpl\_stall\_ack after all pending data phases are complete.

#### Read data De-interleaving

CMI Master bridge has props to indicate whether it can accept interleaved data or not for each VC. This indication is used by CMI Slave bridge to perform de-interleaving for the transactions for which the CMI Master bridge cannot accept interleaved data.

### CMI Slave Bridge

Figure 99 shows a block diagram of the CMI slave bridge.



Figure 5 CMI Slave Bridge

#### CMI Channels and NoC

All CMI channels are supported. The read completions the same NoC channel as read completion data.

#### Decoding, Routing, and Flow Control

CMI Slave Bridge uses CMI source ID of response and read completion channels for route lookup. Responses and read completions may take a different route based on CMI virtual channel. As mentioned earlier on the master bridge section, a bridge property cmi\_bypass\_register\_config can be set to yes to bypass credit initialization after reset de-assertion. The CMI Slave bridge has bridge properties to configure VC credit FIFO depths for responses and read completions. There are programmable registers by which user can control the usable VC credit FIFO depth within the value defined by the bridge property.

#### Errors and Stalls

CMI Slave bridge can raise an interrupt if

the user tries to program the fifo depth register to a value higher than that of available hardware.

Route cannot be found for the incoming CMI source ID

Incoming read completions have ‘Drop’ error type which is unsupported.

Parity on incoming response and read completion are not checked by CMI slave bridge. Similar to response stall on the master side, the request stall handshake is supported by the slave bridge.

#### Read data De-interleaving

As mentioned in CMI Master bridge section, the CMI Slave bridge performs de-interleaving for the transactions for which the CMI Master bridge cannot accept interleaved data. The CMI slave bridges MSB bit of lbinfo field to indicate whether a transaction is a 32B/64B, and the MSB-1 bit to indicate whether the CMI Master bridge can accept interleaved read data for this transaction or not.

## IDI2CMI Converter Microarchitecture

### Introduction

The IDI2CMI converter is designed to handle all non-coherent traffic from IDI agents to CMI slave memories. The entire IDI2CMI converter works in a single clock domain and single power domain. The non-coherent IDI opcodes supported are WiL, WCiL and WCiL\_NS opcodes for write and PRd, CRd\_UC, DRd, DRd\_OPT opcodes for reads. On the CMI interface the converter generates MRd, MWr and MWrPtl opcodes. The Converter follows IDI spec2.02a and CMI spec1.0. As the converter handles only non-coherent traffic, there is no snoop requests or responses. Hence the converter does not have U2C\_REQ and C2U\_RSP channels.

### Block Diagram



Figure 6 IDI2CMI Converter Block Diagram

The various components of the IDI2CMI Converter block are described as follows:

#### Input FIFO

Input FIFO is 2 Deep FIFO whose write clock is running on ungated clock. This will be the only FIFO (apart from IDI/CMI credit manager) running on un-gated clock when the bridge is clock-gated. The empty signal of this FIFO will be used to automatically wake-up the bridge when any new c2u\_request is received.

#### Command Decode

Command Decode logic pushes the incoming to write/read command fifo based on the opcode/opgroup valus and re-map the command context to CMI interface. It is purely combinational logic.

#### Read Command FIFO

Depth of FIFO controlled by P\_IDI\_C2U\_REQ\_IF\_CREDITS parameter. The write to this FIFO is controlled by incoming c2u\_req\_opcode and read is controlled by availability of read\_credits, availability of free\_tids.

#### Write Command FIFO

Depth of FIFO controlled by P\_IDI\_C2U\_REQ\_IF\_CREDITS parameter. The write to this FIFO is controlled by incoming c2u\_req\_opcode and read is controlled by availability of free entries in the write\_pull tracker.

#### Write\_Pull Tracker

The Write\_Pull tracker generates unique UQIDs for write\_pull request to be sent. The Implementation is done as a LUT where each entry stores the incoming address, cqid opcode, parity information etc. and the unique index of the entry is provided as UQID. The number of entries is controlled by P\_IDI\_NUM\_OUTSTANDING parameter which in-turn is controlled from NocStudio using converter prop “conv\_write\_pull\_max\_oustanding”.

#### Write\_Pull Request/Error FIFO

Depth of FIFO controlled by P\_IDI\_C2U\_REQ\_IF\_CREDITS parameter. The write to this FIFO is controlled by availability of free entries in the write\_pull tracker and read is controlled by U2C Response arbiter. In case of an error response that needs to be sent due to unsupported opcodes. The GO-Err response is also pushed into this FIFO.

#### U2C Response Arbiter

U2C response Arbiter is purely combinational logic does the priority arbitration between three path trying to access the IDI U2C\_RSP channel that is listed below.

1) The Write\_pull request FIFO triying to send write\_pull request.

2) The CMI rsp\_\* channel trying to send GO-I/GO-Err for write responses.

3) The CMI rd\_cpl\_\* channel trying to send GO-I/GO\_Err for read completions.

The priority is set as 3>2>1 and this is fixed.

#### CMI Arbiter

CMI Arbiter is a purely combinational implementation that arbitrates between the IDI read and write request. The read has priority over writes and this priority is not programmable. Writes will be forwarded when there are no pending reads.

#### CMI TID Tracker

The CMI TID tracker generates unique TIDs for read and write request to be posted on CMI interface. The Implementation is done as a LUT where each entry stores the IDI command context that is needed to send the response back on U2C\_RSP channel (cqid srcid,opcode etc. and the unique index of the entry is provided as TID. The number of entries is controlled by P\_NUM\_TIDS parameter which in-turn is controlled from NocStudio using converter prop “conv\_req\_max\_outstanding”.

### Data Flow

#### READ Request Data Flow

1. IDI read request is decoded and pushed into the read command FIFO and then posted on CMI interface based on TID and read credit availability.
2. On receiving 2 read completions (one for each 32B), a Go-I is sent on the U2C\_RSP channel.
3. The read completion data received on CMI interface is simply forwarded along with error and poison information on U2C\_DATA interface.

Step 2 and 3 can go in parallel

#### WRITE Request Data FlowC2U Response

1. IDI Write request is decoded and pushed into the write command FIFO and then posted on U2C\_RSP interface based on UQID and U2C\_RSP credit availability.
2. On receiving write pull response on C2U\_DATA the data is re-ordered and 64B data is pushed to the write data buffer. The arbiter posts it on the CMI interface if there is no reads available and write credits and TID are available.
3. On receiving the write response from CMI rsp channel, a GO-I or GO-Err response is sent on U2C\_RSP channel based on the rsp\_type.

### Clock Gating

There are two types clock gating implemented in IDI2CMI converter. One where an external agent like a SoC clock controller wants to gate the clock. This is done by driving a logic ‘1’ on system\_clk\_en port of the converter. In this mode the clock to entire converter is gated. The second mode is where the converter monitors all the request fifos, response fifos and trackers for any pending responses. If all are empty and remain empty for P\_CG\_WATCHDOG\_TIMER number of cycles then the bridge bridge would gate the clock by itself. In this state only the input FIFO would get ungated clock. When a new request is received and the input FIFO becomes non-empty, the bridge wakes up automatically. Also the IDI/CMI credit managers run on un-gated clock in order to respond to master/slave ISMs.

## Programming Model

### IDI master bridge registers

#### IDIM\_BRIDGE\_ID

Unique identifier assigned to the master bridge.

Attribute: R

Security: Non-secure

Bit field description:

* **ID**[15:0] - Unique bridge ID

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | | | | | | | | | | | | | | ID | | | | | | | | | | | | | | | |

Table 2 IDIM\_BRIDGE\_ID register.

#### IDIM\_CFG\_ADDRS\_BASE

Attribute: RW

Security: Secure access only

Bit field description:

* **BASE\_ADDRESS**[63:6] - Base address
* **LLC**[5] - LLC disable
* **DI**[4] -   
  1'b1: Address range disabled
* **R\_Wn**[3] -   
  1'b1: Read enabled to range  
  1'b0: Write enabled to range
* **I**[2] -   
  1'b1: Instruction
* **NS**[1] -   
  1'b1: Non-secure
* **P**[0] -   
  1'b1: Privileged

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| BASE\_ADDRESS | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| BASE\_ADDRESS | | | | | | | | | | | | | | | | | | | | | | | | | | LLC | DI | R\_Wn | I | NS | P |

Table 3 IDIM\_CFG\_ADDRS\_BASE register.

#### IDIM\_CFG\_ADDRS\_MASK

See IDIM\_ADDR\_BASE.

Attribute: RW

Security: Secure access only

Bit field description:

* **MASK**[63:6] - Mask
* **KO**[5] - Keep out/reject read and/or write accesses
* **TM**[4] -   
  1'b1: Enable Trusted Master behavior for secure transactions
* **VAL**[3] -   
  1'b1: R\_Wn field is valid
* **I**[2] -   
  1'b1: Instruction field is valid
* **NS**[1] -   
  1'b1: Non-secure field is valid
* **P**[0] -   
  1'b1: Privileged field is valid

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| MASK | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| MASK | | | | | | | | | | | | | | | | | | | | | | | | | | KO | TM | VAL | I | NS | P |

Table 4 IDIM\_CFG\_ADDRS\_MASK register.

#### IDIM\_EXT\_CFG\_READ\_ALLOWED\_

Register used to determine read access control.

Attribute: RW

Security: Secure access only

Bit field description:

* **read\_allowed**[0] - Read allowed

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | read\_allowed |

Table 5 IDIM\_EXT\_CFG\_READ\_ALLOWED\_ register.

#### IDIM\_INTERRUPT\_FATAL\_MASK

Fatal interrupt Mask

Attribute: RW

Security: Non-secure

Bit field description:

* **FM28**[28] - Fatal Mask for U2C Data channel Data Header Parity Error Interrupts
* **FM27**[27] - Fatal Mask for U2C Data channel Data Parity Error Interrupts
* **FM26**[26] - Fatal Mask for U2C Response channel Response Parity Error Interrupts
* **FM25**[25] - Fatal Mask for U2C Request channel Address Parity Error Interrupts
* **FM24**[24] - Fatal Mask for U2C Request channel Request Parity Error Interrupts
* **FM21**[21] - Fatal Mask for C2U Data channel BE Parity Error Interrupts
* **FM20**[20] - Fatal Mask for C2U Data channel Data Parity Error Interrupts
* **FM19**[19] - Fatal Mask for C2U Data channel Header Parity Error Interrupts
* **FM18**[18] - Fatal Mask for C2U Response channel Response Parity Error Interrupts
* **FM17**[17] - Fatal Mask for C2U Request channel Request Parity Error Interrupts
* **FM16**[16] - Fatal Mask for C2U Request channel Address Parity Error Interrupts
* **FM10**[10] - Fatal Mask for C2U Data channel Route Table Error Interrupts
* **FM9**[9] - Fatal Mask for C2U Rsp channel Route Table Error Interrupts
* **FM8**[8] - Fatal Mask for C2U Req channel Route Table Error Interrupts
* **FM2**[2] - Fatal Mask for C2U Data channel FIFO Overflow
* **FM1**[1] - Fatal Mask for C2U Rsp channel FIFO Overflow
* **FM0**[0] - Fatal Mask for C2U Req channel FIFO Overflow

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | FM28 | FM27 | FM26 | FM25 | FM24 | u | | FM21 | FM20 | FM19 | FM18 | FM17 | FM16 | u | | | | | FM10 | FM9 | FM8 | u | | | | | FM2 | FM1 | FM0 |

Table 6 IDIM\_INTERRUPT\_FATAL\_MASK register.

#### IDIM\_INTERRUPT\_MASK

Interrupt mask register. Individual bit positions match the error bit positions in IDIM\_INTERRPUT\_STATUS. When a mask bit is set, occurrence of the corresponding error event will not cause an interrupt to be raised. When 1'b0, error event will cause interrupt to be raised.

Attribute: RW

Security: Non-secure

Bit field description:

* **M28**[28] - Mask for U2C Data channel Data Header Parity Error Interrupts
* **M27**[27] - Mask for U2C Data channel Data Parity Error Interrupts
* **M26**[26] - Mask for U2C Response channel Response Parity Error Interrupts
* **M25**[25] - Mask for U2C Request channel Address Parity Error Interrupts
* **M24**[24] - Mask for U2C Request channel Request Parity Error Interrupts
* **M21**[21] - Mask for C2U Data channel BE Parity Error Interrupts
* **M20**[20] - Mask for C2U Data channel Data Parity Error Interrupts
* **M19**[19] - Mask for C2U Data channel Header Parity Error Interrupts
* **M18**[18] - Mask for C2U Response channel Response Parity Error Interrupts
* **M17**[17] - Mask for C2U Request channel Request Parity Error Interrupts
* **M16**[16] - Mask for C2U Request channel Address Parity Error Interrupts
* **M10**[10] - Mask for C2U Data channel Route Table Error Interrupts
* **M9**[9] - Mask for C2U Rsp channel Route Table Error Interrupts
* **M8**[8] - Mask for C2U Req channel Route Table Error Interrupts
* **M2**[2] - Mask for C2U Data channel FIFO Overflow
* **M1**[1] - Mask for C2U Rsp channel FIFO Overflow
* **M0**[0] - Mask for C2U Req channel FIFO Overflow

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | M28 | M27 | M26 | M25 | M24 | u | | M21 | M20 | M19 | M18 | M17 | M16 | u | | | | | M10 | M9 | M8 | u | | | | | M2 | M1 | M0 |

Table 7 IDIM\_INTERRUPT\_MASK register.

#### IDIM\_INTERRUPT\_STATUS

These error status bits record the first error event and have to be cleared by writing a 1'b0 before new errors are recorded. For flop structure parity errors, the IDIM\_LOG\_FLOPPARITY\_ERROR register should be cleared by writing 0s to it, before the flop structure parity bit is cleared in this register.

Attribute: WZC

Security: Non-secure

Bit field description:

* **E28**[28] -   
  1'b1: U2C Data Header Parity Error : U2C Data channel
* **E27**[27] -   
  1'b1: U2C Data Parity Error : U2C Data channel
* **E26**[26] -   
  1'b1: U2C Response Parity Error : U2C Rsp channel
* **E25**[25] -   
  1'b1: U2C Req Addr Parity Error : U2C Req channel
* **E24**[24] -   
  1'b1: U2C Req Parity Error : U2C Req channel
* **E21**[21] -   
  1'b1: C2U Data BE Parity Error : C2U data channel
* **E20**[20] -   
  1'b1: C2U Data Parity Error : C2U data channel
* **E19**[19] -   
  1'b1: C2U Data Header Parity Error : C2U data channel
* **E18**[18] -   
  1'b1: C2U Response Parity Error : C2U rsp channel
* **E17**[17] -   
  1'b1: C2U Request Parity Error : C2U req channel
* **E16**[16] -   
  1'b1: C2U Request Address Parity Error : C2U req channel
* **E10**[10] -   
  1'b1: Route Table Error : C2U data channel
* **E9**[9] -   
  1'b1: Route Table Error : C2U rsp channel
* **E8**[8] -   
  1'b1: Route Table Error : C2U req channel
* **E2**[2] -   
  1'b1: FIFO Overflow error : C2U data channel
* **E1**[1] -   
  1'b1: FIFO Overflow error : C2U rsp channel
* **E0**[0] -   
  1'b1: FIFO Overflow error : C2U req channel

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | E28 | E27 | E26 | E25 | E24 | u | | E21 | E20 | E19 | E18 | E17 | E16 | u | | | | | E10 | E9 | E8 | u | | | | | E2 | E1 | E0 |

Table 8 IDIM\_INTERRUPT\_STATUS register.

### IDI Slave Bridge registers

#### IDIS\_BRIDGE\_ID

Unique identifier assigned to the slave bridge.

Attribute: R

Security: Non-secure

Bit field description:

* **ID**[15:0] - Unique bridge ID

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | | | | | | | | | | | | | | ID | | | | | | | | | | | | | | | |

Table 9 IDIS\_BRIDGE\_ID register.

#### IDIS\_INTERRUPT\_FATAL\_MASK

Fatal interrupt Mask

Attribute: RW

Security: Non-secure

Bit field description:

* **FM28**[28] - Fatal Mask for U2C Data channel Data Header Parity Error Interrupts
* **FM27**[27] - Fatal Mask for U2C Data channel Data Parity Error Interrupts
* **FM26**[26] - Fatal Mask for U2C Response channel Response Parity Error Interrupts
* **FM25**[25] - Fatal Mask for U2C Request channel Address Parity Error Interrupts
* **FM24**[24] - Fatal Mask for U2C Request channel Request Parity Error Interrupts
* **FM21**[21] - Fatal Mask for C2U Data channel BE Parity Error Interrupts
* **FM20**[20] - Fatal Mask for C2U Data channel Data Parity Error Interrupts
* **FM19**[19] - Fatal Mask for C2U Data channel Header Parity Error Interrupts
* **FM18**[18] - Fatal Mask for C2U Response channel Response Parity Error Interrupts
* **FM17**[17] - Fatal Mask for C2U Request channel Request Parity Error Interrupts
* **FM16**[16] - Fatal Mask for C2U Request channel Address Parity Error Interrupts
* **FM10**[10] - Fatal Mask for C2U Data channel Route Table Error Interrupts
* **FM9**[9] - Fatal Mask for C2U Rsp channel Route Table Error Interrupts
* **FM8**[8] - Fatal Mask for C2U Req channel Route Table Error Interrupts
* **FM2**[2] - Fatal Mask for C2U Data channel FIFO Overflow
* **FM1**[1] - Fatal Mask for C2U Rsp channel FIFO Overflow
* **FM0**[0] - Fatal Mask for C2U Req channel FIFO Overflow

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | FM28 | FM27 | FM26 | FM25 | FM24 | u | | FM21 | FM20 | FM19 | FM18 | FM17 | FM16 | u | | | | | FM10 | FM9 | FM8 | u | | | | | FM2 | FM1 | FM0 |

Table 10 IDIS\_INTERRUPT\_FATAL\_MASK register.

#### IDIS\_INTERRUPT\_MASK

Interrupt mask register. Individual bit positions match the error bit positions in IDIS\_INTERRUPT\_STATUS. When a mask bit is set, occurrence of the corresponding error event will not cause an interrupt to be raised. When 1'b0, error event will cause interrupt to be raised.

Attribute: RW

Security: Non-secure

Bit field description:

* **M28**[28] - Mask for U2C Data channel Data Header Parity Error Interrupts
* **M27**[27] - Mask for U2C Data channel Data Parity Error Interrupts
* **M26**[26] - Mask for U2C Response channel Response Parity Error Interrupts
* **M25**[25] - Mask for U2C Request channel Address Parity Error Interrupts
* **M24**[24] - Mask for U2C Request channel Request Parity Error Interrupts
* **M21**[21] - Mask for C2U Data channel BE Parity Error Interrupts
* **M20**[20] - Mask for C2U Data channel Data Parity Error Interrupts
* **M19**[19] - Mask for C2U Data channel Header Parity Error Interrupts
* **M18**[18] - Mask for C2U Response channel Response Parity Error Interrupts
* **M17**[17] - Mask for C2U Request channel Request Parity Error Interrupts
* **M16**[16] - Mask for C2U Request channel Address Parity Error Interrupts
* **M10**[10] - Mask interrupts for U2C Data channel Route Table Error
* **M9**[9] - Mask interrupts for U2C Rsp channel Route Table Error
* **M8**[8] - Mask interrupts for U2C Req channel Route Table Error
* **M2**[2] - Mask interrupts for U2C Data channel FIFO Overflow
* **M1**[1] - Mask interrupts for U2C Rsp channel FIFO Overflow
* **M0**[0] - Mask interrupts for U2C Req channel FIFO Overflow

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | M28 | M27 | M26 | M25 | M24 | u | | M21 | M20 | M19 | M18 | M17 | M16 | u | | | | | M10 | M9 | M8 | u | | | | | M2 | M1 | M0 |

Table 11 IDIS\_INTERRUPT\_MASK register.

#### IDIS\_INTERRUPT\_STATUS

These error status bits record the first error event and have to be cleared by writing a 1'b0 before new errors are recorded. For flop structure parity errors, the IDISB\_LOG\_FLOPPARITY register should be cleared by writing 0s to it, before the flop structure parity bit is cleared in this register.

Attribute: WZC

Security: Non-secure

Bit field description:

* **E28**[28] -   
  1'b1: U2C Data Header Parity Error : U2C Data channel
* **E27**[27] -   
  1'b1: U2C Data Parity Error : U2C Data channel
* **E26**[26] -   
  1'b1: U2C Response Parity Error : U2C Rsp channel
* **E25**[25] -   
  1'b1: U2C Req Addr Parity Error : U2C Req channel
* **E24**[24] -   
  1'b1: U2C Req Parity Error : U2C Req channel
* **E21**[21] -   
  1'b1: C2U Data BE Parity Error : C2U data channel
* **E20**[20] -   
  1'b1: C2U Data Parity Error : C2U data channel
* **E19**[19] -   
  1'b1: C2U Data Header Parity Error : C2U data channel
* **E18**[18] -   
  1'b1: C2U Response Parity Error : C2U rsp channel
* **E17**[17] -   
  1'b1: C2U Request Parity Error : C2U req channel
* **E16**[16] -   
  1'b1: C2U Request Address Parity Error : C2U req channel
* **E10**[10] -   
  1'b1: Route Table Error : U2C data channel
* **E9**[9] -   
  1'b1: Route Table Error : U2C rsp channel
* **E8**[8] -   
  1'b1: Route Table Error : U2C req channel
* **E2**[2] -   
  1'b1: FIFO Overflow error : U2C data channel
* **E1**[1] -   
  1'b1: FIFO Overflow error : U2C rsp channel
* **E0**[0] -   
  1'b1: FIFO Overflow error : U2C req channel

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | E28 | E27 | E26 | E25 | E24 | u | | E21 | E20 | E19 | E18 | E17 | E16 | u | | | | | E10 | E9 | E8 | u | | | | | E2 | E1 | E0 |

Table 12 IDIS\_INTERRUPT\_STATUS register.

### CMI master bridge

#### CMIM\_CFG\_ADDRS\_BASE

Attribute: RW

Security: Secure access only

Bit field description:

* **BASE\_ADDRESS**[63:6] - Base address
* **LLC**[5] - LLC disable
* **DI**[4] -   
  1'b1: Address range disabled
* **R\_Wn**[3] -   
  1'b1: Read enabled to range  
  1'b0: Write enabled to range
* **I**[2] -   
  1'b1: Instruction
* **NS**[1] -   
  1'b1: Non-secure
* **P**[0] -   
  1'b1: Privileged

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| BASE\_ADDRESS | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| BASE\_ADDRESS | | | | | | | | | | | | | | | | | | | | | | | | | | LLC | DI | R\_Wn | I | NS | P |

Table 13 CMIM\_CFG\_ADDRS\_BASE register.

#### CMIM\_CFG\_ADDRS\_MASK

See CMIM\_ADDR\_BASE.

Attribute: RW

Security: Secure access only

Bit field description:

* **MASK**[63:6] - Mask
* **KO**[5] - Keep out/reject read and/or write accesses
* **TM**[4] -   
  1'b1: Enable Trusted Master behavior for secure transactions
* **VAL**[3] -   
  1'b1: R\_Wn field is valid
* **I**[2] -   
  1'b1: Instruction field is valid
* **NS**[1] -   
  1'b1: Non-secure field is valid
* **P**[0] -   
  1'b1: Privileged field is valid

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| MASK | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| MASK | | | | | | | | | | | | | | | | | | | | | | | | | | KO | TM | VAL | I | NS | P |

Table 14 CMIM\_CFG\_ADDRS\_MASK register.

#### CMIM\_BRIDGE\_ID

Unique identifier assigned to the master bridge.

Attribute: R

Security: Non-secure

Bit field description:

* **ID**[15:0] - Unique bridge ID

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | | | | | | | | | | | | | | ID | | | | | | | | | | | | | | | |

Table 15 CMIM\_BRIDGE\_ID register.

#### CMIM\_ERROR\_INTERRUPT\_MASK

Interrupt mask register. Individual bit positions match the error bit positions in CMIM\_ERROR\_INTERRUPT\_STATUS. When a mask bit is set, occurrence of the corresponding error event will not cause an interrupt to be raised. When 1'b0, error event will cause interrupt to be raised.

Attribute: RW

Security: Non-secure

Bit field description:

* **M1**[1] - Mask interrupts for Read Req channel Route Table Error
* **M0**[0] - Mask interrupts for Write Req channel Route Table Error

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | M1 | M0 |

Table 16 CMIM\_ERROR\_INTERRUPT\_MASK register.

#### CMIM\_ERROR\_INTERRUPT\_STATUS

These error status bits record the first error event and have to be cleared by writing a 1'b0 before new errors are recorded. For flop structure parity errors, the CMIM\_LOG\_FLOPPARITY\_ERROR register should be cleared by writing 0s to it, before the flop structure parity bit is cleared in this register.

Attribute: WZC

Security: Non-secure

Bit field description:

* **E1**[1] -   
  1'b1: FIFO Overflow error : Read Req channel Route Table Error
* **E0**[0] -   
  1'b1: FIFO Overflow error : Write Req channel Route Table Error

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | E1 | E0 |

Table 17 CMIM\_ERROR\_INTERRUPT\_STATUS register.

#### CMIM\_EXT\_CFG\_READ\_ALLOWED\_

Register used to determine read access control.

Attribute: RW

Security: Secure access only

Bit field description:

* **read\_allowed**[0] - Read allowed

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | read\_allowed |

Table 18 CMIM\_EXT\_CFG\_READ\_ALLOWED\_ register.

#### CMIM\_SCRATCH

This register is used as scratch pad test CSR access.

Attribute: RW

Security: Non-secure

Bit field description:

* **SCRATCH\_PAD**[63:0] - Scratch pad register to test access for CSR space.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| SCRATCH\_PAD | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SCRATCH\_PAD | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Table 19 CMIM\_SCRATCH register.

#### CMIM\_CFG\_ADDRS\_RELOC\_SLV

Register used to relocate a master address to slave address.

Attribute: R

Security: Secure access only

Bit field description:

* **SLV\_RELOC**[59:6] - Slave Relocated Address

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | SLV\_RELOC | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SLV\_RELOC | | | | | | | | | | | | | | | | | | | | | | | | | | u | | | | | |

Table 20 CMIM\_CFG\_ADDRS\_RELOC\_SLV register.

#### CMIM\_RD\_CPL\_RSP\_MAX\_CREDIT

Read completion and response credits and config done

Attribute: RW

Security: Non-secure

Bit field description:

* **credits\_config\_done**[16] - Flag indicating that credit configurations are done and ISM can start initialization
* **rsp\_max\_credit**[15:8] - Maximum credit sent for response
* **rd\_cpl\_max\_credit**[7:0] - Maximum credit sent for read completion

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | | | | | | | | | | | | | credits\_config\_done | rsp\_max\_credit | | | | | | | | rd\_cpl\_max\_credit | | | | | | | |

Table 21 CMIMB\_RD\_CPL\_RSP\_MAX\_CREDIT register.

#### CMIM\_RD\_WR\_REQ\_MAX\_CREDIT

Read and write request credits

Attribute: RW

Security: Non-secure

Bit field description:

* **wr\_req\_max\_credits\_vc3**[63:56] - Maximum credit sent for write request virtual channel 3
* **wr\_req\_max\_credits\_vc2**[55:48] - Maximum credit sent for write request virtual channel 2
* **wr\_req\_max\_credits\_vc1**[47:40] - Maximum credit sent for write request virtual channel 1
* **wr\_req\_max\_credits\_vc0**[39:32] - Maximum credit sent for write request virtual channel 0
* **rd\_req\_max\_credits\_vc3**[31:24] - Maximum credit sent for read request virtual channel 3
* **rd\_req\_max\_credits\_vc2**[23:16] - Maximum credit sent for read request virtual channel 2
* **rd\_req\_max\_credits\_vc1**[15:8] - Maximum credit sent for read request virtual channel 1
* **rd\_req\_max\_credits\_vc0**[7:0] - Maximum credit sent for read request virtual channel 0

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| wr\_req\_max\_credits\_vc3 | | | | | | | | wr\_req\_max\_credits\_vc2 | | | | | | | | wr\_req\_max\_credits\_vc1 | | | | | | | | wr\_req\_max\_credits\_vc0 | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| rd\_req\_max\_credits\_vc3 | | | | | | | | rd\_req\_max\_credits\_vc2 | | | | | | | | rd\_req\_max\_credits\_vc1 | | | | | | | | rd\_req\_max\_credits\_vc0 | | | | | | | |

Table 22 CMIMB\_RD\_WR\_REQ\_MAX\_CREDIT register.

### CMI Slave Bridge registers

#### CMISB\_BRIDGE\_ID

Unique identifier assigned to the slave bridge.

Attribute: R

Security: Non-secure

Bit field description:

* **ID**[15:0] - Unique bridge ID

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | | | | | | | | | | | | | | ID | | | | | | | | | | | | | | | |

Table 23 CMISB\_BRIDGE\_ID register.

#### CMISB\_CG\_OVERRIDE

This register can be used to disable coarse grained clock gating in this module.

Attribute: RW

Security: Non-secure

Bit field description:

* **CG\_OR**[0] -   
  1'b1: Coarse clock gating is locally disabled.  
  1'b0: Coarse clock gating is locally enabled.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | CG\_OR |

Table 24 CMISB\_CG\_OVERRIDE register.

#### CMISB\_ERR\_INJ

Error injection: bit positions match the error bit positions in ERR\_STATUS. When an ERR\_INJ bit is written to 1, the corresponding ERR\_STATUS bit will be set, and an error message will be sent at the severity indicated by ERR\_SVRTY if the corresponding ERR\_MASK bit is clear. All bits of this register are self-clearing (return immediately to 0).

Attribute: RW

Security: Non-secure

Bit field description:

* **I8**[8] - Injects VC1 read completion TID parity error.
* **I7**[7] - Injects VC0 read completion TID parity error.
* **I6**[6] - Injects VC0 write request credit overflow error.
* **I5**[5] - Injects VC1 read request credit overflow error.
* **I4**[4] - Injects VC0 read request credit overflow error.
* **I3**[3] - Injects VC1 read completion data buffer overflow error.
* **I2**[2] - Injects VC1 read completion buffer overflow error.
* **I1**[1] - Injects VC0 read completion data buffer overflow error.
* **I0**[0] - Injects VC0 read completion buffer overflow error.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | | | | | | | | | | | | | | | | | | | | | I8 | I7 | I6 | I5 | I4 | I3 | I2 | I1 | I0 |

Table 25 CMISB\_ERR\_INJ register.

#### CMISB\_ERR\_MASK

Error mask: bit positions match the error bit positions in ERR\_STATUS. When an ERR\_MASK bit is set to 1, occurrence of the corresponding error event will not cause an error message to be sent. When cleared to 0, error event will cause an error message to be sent.

Attribute: RW

Security: Non-secure

Bit field description:

* **M8**[8] -   
  1'b1: Disable error message for VC0 read completion TID parity error.  
  1'b0: Enable error message for VC0 read completion TID parity error.
* **M7**[7] -   
  1'b1: Disable error message for VC0 read completion TID parity error.  
  1'b0: Enable error message for VC0 read completion TID parity error.
* **M6**[6] -   
  1'b1: Disable error message for VC0 write request credit overflow error.  
  1'b0: Enable error message for VC0 write request credit overflow error.
* **M5**[5] -   
  1'b1: Disable error message for VC1 read request credit overflow error.  
  1'b0: Enable error message for VC1 read request credit overflow error.
* **M4**[4] -   
  1'b1: Disable error message for VC0 read request credit overflow error.  
  1'b0: Enable error message for VC0 read request credit overflow error.
* **M3**[3] -   
  1'b1: Disable error message for VC1 read completion data buffer overflow error.  
  1'b0: Enable error message for VC1 read completion data buffer overflow error.
* **M2**[2] -   
  1'b1: Disable error message for VC1 read completion buffer overflow error.  
  1'b0: Enable error message for VC1 read completion buffer overflow error.
* **M1**[1] -   
  1'b1: Disable error message for VC0 read completion data buffer overflow error.  
  1'b0: Enable error message for VC0 read completion data buffer overflow error.
* **M0**[0] -   
  1'b1: Disable error message for VC0 read completion buffer overflow error.  
  1'b0: Enable error message for VC0 read completion buffer overflow error.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | | | | | | | | | | | | | | | | | | | | | M8 | M7 | M6 | M5 | M4 | M3 | M2 | M1 | M0 |

Table 26 CMISB\_ERR\_MASK register.

#### CMISB\_ERR\_STATUS

These error status bits record the first error event and have to be cleared by writing a 1'b0 before new errors are recorded.

Attribute: WZC

Security: Non-secure

Bit field description:

* **E8**[8] - 1'b1: VC0 read completion TID parity error detected.
* **E7**[7] - 1'b1: VC0 read completion TID parity error detected.
* **E6**[6] - 1'b1: VC0 write request credit overflow error detected.
* **E5**[5] - 1'b1: VC1 read request credit overflow error detected.
* **E4**[4] - 1'b1: VC0 read request credit overflow error detected.
* **E3**[3] - 1'b1: VC1 read completion data buffer overflow error detected.
* **E2**[2] - 1'b1: VC1 read completion buffer overflow error detected.
* **E1**[1] - 1'b1: VC0 read completion data buffer overflow error detected.
* **E0**[0] - 1'b1: VC0 read completion buffer overflow error detected.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | | | | | | | | | | | | | | | | | | | | | E8 | E7 | E6 | E5 | E4 | E3 | E2 | E1 | E0 |

Table 27 CMISB\_ERR\_STATUS register.

#### CMISB\_ERR\_SVRTY

Error severity: bit positions match the error bit positions in ERR\_STATUS, each determining the severity level of the associated error. When an ERR\_SVRTY bit is set to 1, the error is considered fatal. When cleared to 0, the error is considered non-fatal.

Attribute: RW

Security: Non-secure

Bit field description:

* **S8**[8] - Severity of VC0 read completion TID parity error.
* **S7**[7] - Severity of VC0 read completion TID parity error.
* **S6**[6] - Severity of VC0 write request credit overflow error.
* **S5**[5] - Severity of VC1 read request credit overflow error.
* **S4**[4] - Severity of VC0 read request credit overflow error.
* **S3**[3] - Severity of VC1 read completion data buffer overflow error.
* **S2**[2] - Severity of VC1 read completion buffer overflow error.
* **S1**[1] - Severity of VC0 read completion data buffer overflow error.
* **S0**[0] - Severity of VC0 read completion buffer overflow error.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | | | | | | | | | | | | | | | | | | | | | S8 | S7 | S6 | S5 | S4 | S3 | S2 | S1 | S0 |

Table 28 CMISB\_ERR\_SVRTY register.

#### CMISB\_IDLE\_STATUS

This register observes the idle status of this module. Note that in order to read this register, Regbus layer logic must be active.

Attribute: R

Security: Non-secure

Bit field description:

* **RX**[4] - RX NoC interface idle.
* **TX**[3] - TX NoC interface idle.
* **REQ**[2] - CMI Req channel (reqch) idle.
* **CPL**[1] - CMI Read Completion channel (rdcplch) idle.
* **RSP**[0] - CMI Resp channel (rspch) idle.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | RX | TX | REQ | CPL | RSP |

Table 29 CMISB\_IDLE\_STATUS register.

#### CMISB\_ISM\_IDLE\_THRESHOLD

Threshold for IDLE transition on CMI ISM

Attribute: RW

Security: Non-secure

Bit field description:

* **ism\_idle\_threshold**[7:0] - Number of consecutive idle cycles on CMI interface for F2MEM CMI requestor ISM to transition from ACTIVE to IDLE\_REQ

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | | | | | | | | | | | | | | | | | | | | | | ism\_idle\_threshold | | | | | | | |

Table 30 CMISB\_ISM\_IDLE\_THRESHOLD register.

#### CMISB\_RD\_WR\_REQ\_MAX\_CREDIT

Read and write request credits

Attribute: RW

Security: Non-secure

Bit field description:

* **credits\_config\_done**[16] - Flag indicating that credit configurations are done and ISM can start initialization
* **wr\_req\_max\_credit**[15:8] - Maximum credit sent for write request
* **rd\_req\_max\_credit**[7:0] - Maximum credit sent for read request

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | | | | | | | | | | | | | credits\_config\_done | wr\_req\_max\_credit | | | | | | | | rd\_req\_max\_credit | | | | | | | |

Table 31 CMISB\_RD\_WR\_REQ\_MAX\_CREDIT register.

#### CMISB\_RDCPL\_RSP\_VC\_MAX\_CREDIT

Read completion and response credits

Attribute: RW

Security: Non-secure

Bit field description:

* **rsp\_max\_credits\_vc3**[63:56] - Maximum credit sent for response virtual channel 3
* **rsp\_max\_credits\_vc2**[55:48] - Maximum credit sent for response virtual channel 2
* **rsp\_max\_credits\_vc1**[47:40] - Maximum credit sent for response virtual channel 1
* **rsp\_max\_credits\_vc0**[39:32] - Maximum credit sent for response virtual channel 0
* **rd\_cpl\_max\_credits\_vc3**[31:24] - Maximum credit sent for read completion virtual channel 3
* **rd\_cpl\_max\_credits\_vc2**[23:16] - Maximum credit sent for read completion virtual channel 2
* **rd\_cpl\_max\_credits\_vc1**[15:8] - Maximum credit sent for read completion virtual channel 1
* **rd\_cpl\_max\_credits\_vc0**[7:0] - Maximum credit sent for read completion virtual channel 0

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| rsp\_max\_credits\_vc3 | | | | | | | | rsp\_max\_credits\_vc2 | | | | | | | | rsp\_max\_credits\_vc1 | | | | | | | | rsp\_max\_credits\_vc0 | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| rd\_cpl\_max\_credits\_vc3 | | | | | | | | rd\_cpl\_max\_credits\_vc2 | | | | | | | | rd\_cpl\_max\_credits\_vc1 | | | | | | | | rd\_cpl\_max\_credits\_vc0 | | | | | | | |

Table 32 CMISB\_RDCPL\_RSP\_VC\_MAX\_CREDIT register.

### RSSB Registers

#### RSSB\_ERROR\_INTERRUPT\_INJ

Error injection: bit positions match the error bit positions in rssb\_error\_interrupt\_status. When an ERR\_INJ bit is written to 1, the corrsponding ERR\_STATUS bit will be set, and an error message will be sent at the severity indicated by ERR\_SVRTY if the corresponding ERR\_MASK bit is clear. All bits of this register are self-clearing (return immediately to 0).

Attribute: RW

Security: Non-secure

Bit field description:

* **PGE\_I**[33] - 1'b1: Inject PGE error interrupt
* **OPRT\_UNAVL\_I**[32] - 1'b1: Inject OPRT\_UNAVL error interrupt
* **CRD\_OFLW\_VC\_7\_I**[31] - 1'b1: Inject CRD\_OFLW\_VC\_7 error interrupt
* **FIFO\_OVERFLOW\_VC\_7\_I**[30] - 1'b1: Inject FIFO\_OVERFLOW\_VC\_7 error interrupt
* **SOP\_AFTER\_SOP\_VC\_7\_I**[29] - 1'b1: Inject SOP\_AFTER\_SOP\_VC\_7 error interrupt
* **TRANS\_WITHOUT\_SOP\_VC\_7\_I**[28] - 1'b1: Inject TRANS\_WITHOUT\_SOP\_VC\_7 error interrupt
* **CRD\_OFLW\_VC\_6\_I**[27] - 1'b1: Inject CRD\_OFLW\_VC\_6 error interrupt
* **FIFO\_OVERFLOW\_VC\_6\_I**[26] - 1'b1: Inject FIFO\_OVERFLOW\_VC\_6 error interrupt
* **SOP\_AFTER\_SOP\_VC\_6\_I**[25] - 1'b1: Inject SOP\_AFTER\_SOP\_VC\_6 error interrupt
* **TRANS\_WITHOUT\_SOP\_VC\_6\_I**[24] - 1'b1: Inject TRANS\_WITHOUT\_SOP\_VC\_6 error interrupt
* **CRD\_OFLW\_VC\_5\_I**[23] - 1'b1: Inject CRD\_OFLW\_VC\_5 error interrupt
* **FIFO\_OVERFLOW\_VC\_5\_I**[22] - 1'b1: Inject FIFO\_OVERFLOW\_VC\_5 error interrupt
* **SOP\_AFTER\_SOP\_VC\_5\_I**[21] - 1'b1: Inject SOP\_AFTER\_SOP\_VC\_5 error interrupt
* **TRANS\_WITHOUT\_SOP\_VC\_5\_I**[20] - 1'b1: Inject TRANS\_WITHOUT\_SOP\_VC\_5 error interrupt
* **CRD\_OFLW\_VC\_4\_I**[19] - 1'b1: Inject CRD\_OFLW\_VC\_4 error interrupt
* **FIFO\_OVERFLOW\_VC\_4\_I**[18] - 1'b1: Inject FIFO\_OVERFLOW\_VC\_4 error interrupt
* **SOP\_AFTER\_SOP\_VC\_4\_I**[17] - 1'b1: Inject SOP\_AFTER\_SOP\_VC\_4 error interrupt
* **TRANS\_WITHOUT\_SOP\_VC\_4\_I**[16] - 1'b1: Inject TRANS\_WITHOUT\_SOP\_VC\_4 error interrupt
* **CRD\_OFLW\_VC\_3\_I**[15] - 1'b1: Inject CRD\_OFLW\_VC\_3 error interrupt
* **FIFO\_OVERFLOW\_VC\_3\_I**[14] - 1'b1: Inject FIFO\_OVERFLOW\_VC\_3 error interrupt
* **SOP\_AFTER\_SOP\_VC\_3\_I**[13] - 1'b1: Inject SOP\_AFTER\_SOP\_VC\_3 error interrupt
* **TRANS\_WITHOUT\_SOP\_VC\_3\_I**[12] - 1'b1: Inject TRANS\_WITHOUT\_SOP\_VC\_3 error interrupt
* **CRD\_OFLW\_VC\_2\_I**[11] - 1'b1: Inject CRD\_OFLW\_VC\_2 error interrupt
* **FIFO\_OVERFLOW\_VC\_2\_I**[10] - 1'b1: Inject FIFO\_OVERFLOW\_VC\_2 error interrupt
* **SOP\_AFTER\_SOP\_VC\_2\_I**[9] - 1'b1: Inject SOP\_AFTER\_SOP\_VC\_2 error interrupt
* **TRANS\_WITHOUT\_SOP\_VC\_2\_I**[8] - 1'b1: Inject TRANS\_WITHOUT\_SOP\_VC\_2 error interrupt
* **CRD\_OFLW\_VC\_1\_I**[7] - 1'b1: Inject CRD\_OFLW\_VC\_1 error interrupt
* **FIFO\_OVERFLOW\_VC\_1\_I**[6] - 1'b1: Inject FIFO\_OVERFLOW\_VC\_1 error interrupt
* **SOP\_AFTER\_SOP\_VC\_1\_I**[5] - 1'b1: Inject SOP\_AFTER\_SOP\_VC\_1 error interrupt
* **TRANS\_WITHOUT\_SOP\_VC\_1\_I**[4] - 1'b1: Inject TRANS\_WITHOUT\_SOP\_VC\_1 error interrupt
* **CRD\_OFLW\_VC\_0\_I**[3] - 1'b1: Inject CRD\_OFLW\_VC\_0 error interrupt
* **FIFO\_OVERFLOW\_VC\_0\_I**[2] - 1'b1: Inject FIFO\_OVERFLOW\_VC\_0 error interrupt
* **SOP\_AFTER\_SOP\_VC\_0\_I**[1] - 1'b1: Inject SOP\_AFTER\_SOP\_VC\_0 error interrupt
* **TRANS\_WITHOUT\_SOP\_VC\_0\_I**[0] - 1'b1: Inject TRANS\_WITHOUT\_SOP\_VC\_0 error interrupt

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | PGE\_I | OPRT\_UNAVL\_I |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CRD\_OFLW\_VC\_7\_I | FIFO\_OVERFLOW\_VC\_7\_I | SOP\_AFTER\_SOP\_VC\_7\_I | TRANS\_WITHOUT\_SOP\_VC\_7\_I | CRD\_OFLW\_VC\_6\_I | FIFO\_OVERFLOW\_VC\_6\_I | SOP\_AFTER\_SOP\_VC\_6\_I | TRANS\_WITHOUT\_SOP\_VC\_6\_I | CRD\_OFLW\_VC\_5\_I | FIFO\_OVERFLOW\_VC\_5\_I | SOP\_AFTER\_SOP\_VC\_5\_I | TRANS\_WITHOUT\_SOP\_VC\_5\_I | CRD\_OFLW\_VC\_4\_I | FIFO\_OVERFLOW\_VC\_4\_I | SOP\_AFTER\_SOP\_VC\_4\_I | TRANS\_WITHOUT\_SOP\_VC\_4\_I | CRD\_OFLW\_VC\_3\_I | FIFO\_OVERFLOW\_VC\_3\_I | SOP\_AFTER\_SOP\_VC\_3\_I | TRANS\_WITHOUT\_SOP\_VC\_3\_I | CRD\_OFLW\_VC\_2\_I | FIFO\_OVERFLOW\_VC\_2\_I | SOP\_AFTER\_SOP\_VC\_2\_I | TRANS\_WITHOUT\_SOP\_VC\_2\_I | CRD\_OFLW\_VC\_1\_I | FIFO\_OVERFLOW\_VC\_1\_I | SOP\_AFTER\_SOP\_VC\_1\_I | TRANS\_WITHOUT\_SOP\_VC\_1\_I | CRD\_OFLW\_VC\_0\_I | FIFO\_OVERFLOW\_VC\_0\_I | SOP\_AFTER\_SOP\_VC\_0\_I | TRANS\_WITHOUT\_SOP\_VC\_0\_I |

Table 33 RSSB\_ERROR\_INTERRUPT\_INJ register.

#### RSSB\_ERROR\_INTERRUPT\_MASK

Interrupt mask register. Individual bit position matches the error bit positions in rssb\_error\_interrupt\_status. When an INTM bit is set, occurrence of the corresponding error event will not cause an interrupt to be raised. When 1'b0, error event will cause interrupt to be raised.

Attribute: RW

Security: Non-secure

Bit field description:

* **PGE\_M**[33] - 1'b1: Mask PGE error interrupt
* **OPRT\_UNAVL\_M**[32] - 1'b1: Mask OPRT\_UNAVL error interrupt
* **CRD\_OFLW\_VC\_7\_M**[31] - 1'b1: Mask CRD\_OFLW\_VC\_7 error interrupt
* **FIFO\_OVERFLOW\_VC\_7\_M**[30] - 1'b1: Mask FIFO\_OVERFLOW\_VC\_7 error interrupt
* **SOP\_AFTER\_SOP\_VC\_7\_M**[29] - 1'b1: Mask SOP\_AFTER\_SOP\_VC\_7 error interrupt
* **TRANS\_WITHOUT\_SOP\_VC\_7\_M**[28] - 1'b1: Mask TRANS\_WITHOUT\_SOP\_VC\_7 error interrupt
* **CRD\_OFLW\_VC\_6\_M**[27] - 1'b1: Mask CRD\_OFLW\_VC\_6 error interrupt
* **FIFO\_OVERFLOW\_VC\_6\_M**[26] - 1'b1: Mask FIFO\_OVERFLOW\_VC\_6 error interrupt
* **SOP\_AFTER\_SOP\_VC\_6\_M**[25] - 1'b1: Mask SOP\_AFTER\_SOP\_VC\_6 error interrupt
* **TRANS\_WITHOUT\_SOP\_VC\_6\_M**[24] - 1'b1: Mask TRANS\_WITHOUT\_SOP\_VC\_6 error interrupt
* **CRD\_OFLW\_VC\_5\_M**[23] - 1'b1: Mask CRD\_OFLW\_VC\_5 error interrupt
* **FIFO\_OVERFLOW\_VC\_5\_M**[22] - 1'b1: Mask FIFO\_OVERFLOW\_VC\_5 error interrupt
* **SOP\_AFTER\_SOP\_VC\_5\_M**[21] - 1'b1: Mask SOP\_AFTER\_SOP\_VC\_5 error interrupt
* **TRANS\_WITHOUT\_SOP\_VC\_5\_M**[20] - 1'b1: Mask TRANS\_WITHOUT\_SOP\_VC\_5 error interrupt
* **CRD\_OFLW\_VC\_4\_M**[19] - 1'b1: Mask CRD\_OFLW\_VC\_4 error interrupt
* **FIFO\_OVERFLOW\_VC\_4\_M**[18] - 1'b1: Mask FIFO\_OVERFLOW\_VC\_4 error interrupt
* **SOP\_AFTER\_SOP\_VC\_4\_M**[17] - 1'b1: Mask SOP\_AFTER\_SOP\_VC\_4 error interrupt
* **TRANS\_WITHOUT\_SOP\_VC\_4\_M**[16] - 1'b1: Mask TRANS\_WITHOUT\_SOP\_VC\_4 error interrupt
* **CRD\_OFLW\_VC\_3\_M**[15] - 1'b1: Mask CRD\_OFLW\_VC\_3 error interrupt
* **FIFO\_OVERFLOW\_VC\_3\_M**[14] - 1'b1: Mask FIFO\_OVERFLOW\_VC\_3 error interrupt
* **SOP\_AFTER\_SOP\_VC\_3\_M**[13] - 1'b1: Mask SOP\_AFTER\_SOP\_VC\_3 error interrupt
* **TRANS\_WITHOUT\_SOP\_VC\_3\_M**[12] - 1'b1: Mask TRANS\_WITHOUT\_SOP\_VC\_3 error interrupt
* **CRD\_OFLW\_VC\_2\_M**[11] - 1'b1: Mask CRD\_OFLW\_VC\_2 error interrupt
* **FIFO\_OVERFLOW\_VC\_2\_M**[10] - 1'b1: Mask FIFO\_OVERFLOW\_VC\_2 error interrupt
* **SOP\_AFTER\_SOP\_VC\_2\_M**[9] - 1'b1: Mask SOP\_AFTER\_SOP\_VC\_2 error interrupt
* **TRANS\_WITHOUT\_SOP\_VC\_2\_M**[8] - 1'b1: Mask TRANS\_WITHOUT\_SOP\_VC\_2 error interrupt
* **CRD\_OFLW\_VC\_1\_M**[7] - 1'b1: Mask CRD\_OFLW\_VC\_1 error interrupt
* **FIFO\_OVERFLOW\_VC\_1\_M**[6] - 1'b1: Mask FIFO\_OVERFLOW\_VC\_1 error interrupt
* **SOP\_AFTER\_SOP\_VC\_1\_M**[5] - 1'b1: Mask SOP\_AFTER\_SOP\_VC\_1 error interrupt
* **TRANS\_WITHOUT\_SOP\_VC\_1\_M**[4] - 1'b1: Mask TRANS\_WITHOUT\_SOP\_VC\_1 error interrupt
* **CRD\_OFLW\_VC\_0\_M**[3] - 1'b1: Mask CRD\_OFLW\_VC\_0 error interrupt
* **FIFO\_OVERFLOW\_VC\_0\_M**[2] - 1'b1: Mask FIFO\_OVERFLOW\_VC\_0 error interrupt
* **SOP\_AFTER\_SOP\_VC\_0\_M**[1] - 1'b1: Mask SOP\_AFTER\_SOP\_VC\_0 error interrupt
* **TRANS\_WITHOUT\_SOP\_VC\_0\_M**[0] - 1'b1: Mask TRANS\_WITHOUT\_SOP\_VC\_0 error interrupt

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | PGE\_M | OPRT\_UNAVL\_M |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CRD\_OFLW\_VC\_7\_M | FIFO\_OVERFLOW\_VC\_7\_M | SOP\_AFTER\_SOP\_VC\_7\_M | TRANS\_WITHOUT\_SOP\_VC\_7\_M | CRD\_OFLW\_VC\_6\_M | FIFO\_OVERFLOW\_VC\_6\_M | SOP\_AFTER\_SOP\_VC\_6\_M | TRANS\_WITHOUT\_SOP\_VC\_6\_M | CRD\_OFLW\_VC\_5\_M | FIFO\_OVERFLOW\_VC\_5\_M | SOP\_AFTER\_SOP\_VC\_5\_M | TRANS\_WITHOUT\_SOP\_VC\_5\_M | CRD\_OFLW\_VC\_4\_M | FIFO\_OVERFLOW\_VC\_4\_M | SOP\_AFTER\_SOP\_VC\_4\_M | TRANS\_WITHOUT\_SOP\_VC\_4\_M | CRD\_OFLW\_VC\_3\_M | FIFO\_OVERFLOW\_VC\_3\_M | SOP\_AFTER\_SOP\_VC\_3\_M | TRANS\_WITHOUT\_SOP\_VC\_3\_M | CRD\_OFLW\_VC\_2\_M | FIFO\_OVERFLOW\_VC\_2\_M | SOP\_AFTER\_SOP\_VC\_2\_M | TRANS\_WITHOUT\_SOP\_VC\_2\_M | CRD\_OFLW\_VC\_1\_M | FIFO\_OVERFLOW\_VC\_1\_M | SOP\_AFTER\_SOP\_VC\_1\_M | TRANS\_WITHOUT\_SOP\_VC\_1\_M | CRD\_OFLW\_VC\_0\_M | FIFO\_OVERFLOW\_VC\_0\_M | SOP\_AFTER\_SOP\_VC\_0\_M | TRANS\_WITHOUT\_SOP\_VC\_0\_M |

Table 34 RSSB\_ERROR\_INTERRUPT\_MASK register.

#### RSSB\_ERROR\_INTERRUPT\_STATUS

This register tracks the interrupt or error events that can occur in the rssb. This register is readable, and can be cleared by performing a write with the write data bits set to 0 for the bits that should be cleared.

Attribute: WZC

Security: Non-secure

Bit field description:

* **PGE**[33] - 1'b1: [FATAL] Power gating error, traffic received for port p0 after rssb committed to power down, and then a valid pkt comes in. Per port.
* **OPRT\_UNAVL**[32] - 1'b1: [FATAL] Traffic destined for output port p0 which is unavailable
* **CRD\_OFLW\_VC\_7**[31] - Credit overflow for output port p0 vc7
* **FIFO\_OVERFLOW\_VC\_7**[30] - Fifo overlow for input port p0 vc7
* **SOP\_AFTER\_SOP\_VC\_7**[29] - 1'b1: Sets if a SOP is received after SOP for input port p0 vc7. This event will always trigger an interrupt and cannot be masked.
* **TRANS\_WITHOUT\_SOP\_VC\_7**[28] - 1'b1: Sets if a transaction is initiated w/o SOP for input port p0 vc7. This event will always trigger an interrupt and cannot be masked.
* **CRD\_OFLW\_VC\_6**[27] - Credit overflow for output port p0 vc7
* **FIFO\_OVERFLOW\_VC\_6**[26] - Fifo overlow for input port p0 vc7
* **SOP\_AFTER\_SOP\_VC\_6**[25] - 1'b1: Sets if a SOP is received after SOP for input port p0 vc7. This event will always trigger an interrupt and cannot be masked.
* **TRANS\_WITHOUT\_SOP\_VC\_6**[24] - 1'b1: Sets if a transaction is initiated w/o SOP for input port p0 vc7. This event will always trigger an interrupt and cannot be masked.
* **CRD\_OFLW\_VC\_5**[23] - Credit overflow for output port p0 vc7
* **FIFO\_OVERFLOW\_VC\_5**[22] - Fifo overlow for input port p0 vc7
* **SOP\_AFTER\_SOP\_VC\_5**[21] - 1'b1: Sets if a SOP is received after SOP for input port p0 vc7. This event will always trigger an interrupt and cannot be masked.
* **TRANS\_WITHOUT\_SOP\_VC\_5**[20] - 1'b1: Sets if a transaction is initiated w/o SOP for input port p0 vc7. This event will always trigger an interrupt and cannot be masked.
* **CRD\_OFLW\_VC\_4**[19] - Credit overflow for output port p0 vc7
* **FIFO\_OVERFLOW\_VC\_4**[18] - Fifo overlow for input port p0 vc7
* **SOP\_AFTER\_SOP\_VC\_4**[17] - 1'b1: Sets if a SOP is received after SOP for input port p0 vc7. This event will always trigger an interrupt and cannot be masked.
* **TRANS\_WITHOUT\_SOP\_VC\_4**[16] - 1'b1: Sets if a transaction is initiated w/o SOP for input port p0 vc7. This event will always trigger an interrupt and cannot be masked.
* **CRD\_OFLW\_VC\_3**[15] - Credit overflow for output port p0 vc7
* **FIFO\_OVERFLOW\_VC\_3**[14] - Fifo overlow for input port p0 vc7
* **SOP\_AFTER\_SOP\_VC\_3**[13] - 1'b1: Sets if a SOP is received after SOP for input port p0 vc7. This event will always trigger an interrupt and cannot be masked.
* **TRANS\_WITHOUT\_SOP\_VC\_3**[12] - 1'b1: Sets if a transaction is initiated w/o SOP for input port p0 vc7. This event will always trigger an interrupt and cannot be masked.
* **CRD\_OFLW\_VC\_2**[11] - Credit overflow for output port p0 vc7
* **FIFO\_OVERFLOW\_VC\_2**[10] - Fifo overlow for input port p0 vc7
* **SOP\_AFTER\_SOP\_VC\_2**[9] - 1'b1: Sets if a SOP is received after SOP for input port p0 vc7. This event will always trigger an interrupt and cannot be masked.
* **TRANS\_WITHOUT\_SOP\_VC\_2**[8] - 1'b1: Sets if a transaction is initiated w/o SOP for input port p0 vc7. This event will always trigger an interrupt and cannot be masked.
* **CRD\_OFLW\_VC\_1**[7] - Credit overflow for output port p0 vc7
* **FIFO\_OVERFLOW\_VC\_1**[6] - Fifo overlow for input port p0 vc7
* **SOP\_AFTER\_SOP\_VC\_1**[5] - 1'b1: Sets if a SOP is received after SOP for input port p0 vc7. This event will always trigger an interrupt and cannot be masked.
* **TRANS\_WITHOUT\_SOP\_VC\_1**[4] - 1'b1: Sets if a transaction is initiated w/o SOP for input port p0 vc7. This event will always trigger an interrupt and cannot be masked.
* **CRD\_OFLW\_VC\_0**[3] - Credit overflow for output port p0 vc7
* **FIFO\_OVERFLOW\_VC\_0**[2] - Fifo overlow for input port p0 vc7
* **SOP\_AFTER\_SOP\_VC\_0**[1] - 1'b1: Sets if a SOP is received after SOP for input port p0 vc7. This event will always trigger an interrupt and cannot be masked.
* **TRANS\_WITHOUT\_SOP\_VC\_0**[0] - 1'b1: Sets if a transaction is initiated w/o SOP for input port p0 vc7. This event will always trigger an interrupt and cannot be masked.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | PGE | OPRT\_UNAVL |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CRD\_OFLW\_VC\_7 | FIFO\_OVERFLOW\_VC\_7 | SOP\_AFTER\_SOP\_VC\_7 | TRANS\_WITHOUT\_SOP\_VC\_7 | CRD\_OFLW\_VC\_6 | FIFO\_OVERFLOW\_VC\_6 | SOP\_AFTER\_SOP\_VC\_6 | TRANS\_WITHOUT\_SOP\_VC\_6 | CRD\_OFLW\_VC\_5 | FIFO\_OVERFLOW\_VC\_5 | SOP\_AFTER\_SOP\_VC\_5 | TRANS\_WITHOUT\_SOP\_VC\_5 | CRD\_OFLW\_VC\_4 | FIFO\_OVERFLOW\_VC\_4 | SOP\_AFTER\_SOP\_VC\_4 | TRANS\_WITHOUT\_SOP\_VC\_4 | CRD\_OFLW\_VC\_3 | FIFO\_OVERFLOW\_VC\_3 | SOP\_AFTER\_SOP\_VC\_3 | TRANS\_WITHOUT\_SOP\_VC\_3 | CRD\_OFLW\_VC\_2 | FIFO\_OVERFLOW\_VC\_2 | SOP\_AFTER\_SOP\_VC\_2 | TRANS\_WITHOUT\_SOP\_VC\_2 | CRD\_OFLW\_VC\_1 | FIFO\_OVERFLOW\_VC\_1 | SOP\_AFTER\_SOP\_VC\_1 | TRANS\_WITHOUT\_SOP\_VC\_1 | CRD\_OFLW\_VC\_0 | FIFO\_OVERFLOW\_VC\_0 | SOP\_AFTER\_SOP\_VC\_0 | TRANS\_WITHOUT\_SOP\_VC\_0 |

Table 35 RSSB\_ERROR\_INTERRUPT\_STATUS register.

#### RSSB\_ERROR\_INTERRUPT\_SVRTY

Interrupt Severity Register. Individual bit position matches the error bit positions in rssb\_error\_interrupt\_status. When an INT severity bit is set, occurrence of the corresponding error event will cause a fatal interrupt to be raised. When not set (1'b0), error event will cause non-fatal interrupt to be raised.

Attribute: RW

Security: Non-secure

Bit field description:

* **PGE\_S**[33] -   
  1'b1: PGE error interrupt is fatal.  
  1'b0: PGE error interrupt is non-fatal.
* **OPRT\_UNAVL\_S**[32] -   
  1'b1: OPRT\_UNAVL error interrupt is fatal.  
  1'b0: OPRT\_UNAVL error interrupt is non-fatal.
* **CRD\_OFLW\_VC\_7\_S**[31] -   
  1'b1: CRD\_OFLW\_VC\_7 error interrupt is fatal.  
  1'b0: CRD\_OFLW\_VC\_7 error interrupt is non-fatal.
* **FIFO\_OVERFLOW\_VC\_7\_S**[30] -   
  1'b1: FIFO\_OVERFLOW\_VC\_7 error interrupt is fatal.  
  1'b0: FIFO\_OVERFLOW\_VC\_7 error interrupt is non-fatal.
* **SOP\_AFTER\_SOP\_VC\_7\_S**[29] -   
  1'b1: SOP\_AFTER\_SOP\_VC\_7 error interrupt is fatal.  
  1'b0: SOP\_AFTER\_SOP\_VC\_7 error interrupt is non-fatal.
* **TRANS\_WITHOUT\_SOP\_VC\_7\_S**[28] -   
  1'b1: TRANS\_WITHOUT\_SOP\_VC\_7 error interrupt is fatal.  
  1'b0: TRANS\_WITHOUT\_SOP\_VC\_7 error interrupt is non-fatal.
* **CRD\_OFLW\_VC\_6\_S**[27] -   
  1'b1: CRD\_OFLW\_VC\_6 error interrupt is fatal.  
  1'b0: CRD\_OFLW\_VC\_6 error interrupt is non-fatal.
* **FIFO\_OVERFLOW\_VC\_6\_S**[26] -   
  1'b1: FIFO\_OVERFLOW\_VC\_6 error interrupt is fatal.  
  1'b0: FIFO\_OVERFLOW\_VC\_6 error interrupt is non-fatal.
* **SOP\_AFTER\_SOP\_VC\_6\_S**[25] -   
  1'b1: SOP\_AFTER\_SOP\_VC\_6 error interrupt is fatal.  
  1'b0: SOP\_AFTER\_SOP\_VC\_6 error interrupt is non-fatal.
* **TRANS\_WITHOUT\_SOP\_VC\_6\_S**[24] -   
  1'b1: TRANS\_WITHOUT\_SOP\_VC\_6 error interrupt is fatal.  
  1'b0: TRANS\_WITHOUT\_SOP\_VC\_6 error interrupt is non-fatal.
* **CRD\_OFLW\_VC\_5\_S**[23] -   
  1'b1: CRD\_OFLW\_VC\_5 error interrupt is fatal.  
  1'b0: CRD\_OFLW\_VC\_5 error interrupt is non-fatal.
* **FIFO\_OVERFLOW\_VC\_5\_S**[22] -   
  1'b1: FIFO\_OVERFLOW\_VC\_5 error interrupt is fatal.  
  1'b0: FIFO\_OVERFLOW\_VC\_5 error interrupt is non-fatal.
* **SOP\_AFTER\_SOP\_VC\_5\_S**[21] -   
  1'b1: SOP\_AFTER\_SOP\_VC\_5 error interrupt is fatal.  
  1'b0: SOP\_AFTER\_SOP\_VC\_5 error interrupt is non-fatal.
* **TRANS\_WITHOUT\_SOP\_VC\_5\_S**[20] -   
  1'b1: TRANS\_WITHOUT\_SOP\_VC\_5 error interrupt is fatal.  
  1'b0: TRANS\_WITHOUT\_SOP\_VC\_5 error interrupt is non-fatal.
* **CRD\_OFLW\_VC\_4\_S**[19] -   
  1'b1: CRD\_OFLW\_VC\_4 error interrupt is fatal.  
  1'b0: CRD\_OFLW\_VC\_4 error interrupt is non-fatal.
* **FIFO\_OVERFLOW\_VC\_4\_S**[18] -   
  1'b1: FIFO\_OVERFLOW\_VC\_4 error interrupt is fatal.  
  1'b0: FIFO\_OVERFLOW\_VC\_4 error interrupt is non-fatal.
* **SOP\_AFTER\_SOP\_VC\_4\_S**[17] -   
  1'b1: SOP\_AFTER\_SOP\_VC\_4 error interrupt is fatal.  
  1'b0: SOP\_AFTER\_SOP\_VC\_4 error interrupt is non-fatal.
* **TRANS\_WITHOUT\_SOP\_VC\_4\_S**[16] -   
  1'b1: TRANS\_WITHOUT\_SOP\_VC\_4 error interrupt is fatal.  
  1'b0: TRANS\_WITHOUT\_SOP\_VC\_4 error interrupt is non-fatal.
* **CRD\_OFLW\_VC\_3\_S**[15] -   
  1'b1: CRD\_OFLW\_VC\_3 error interrupt is fatal.  
  1'b0: CRD\_OFLW\_VC\_3 error interrupt is non-fatal.
* **FIFO\_OVERFLOW\_VC\_3\_S**[14] -   
  1'b1: FIFO\_OVERFLOW\_VC\_3 error interrupt is fatal.  
  1'b0: FIFO\_OVERFLOW\_VC\_3 error interrupt is non-fatal.
* **SOP\_AFTER\_SOP\_VC\_3\_S**[13] -   
  1'b1: SOP\_AFTER\_SOP\_VC\_3 error interrupt is fatal.  
  1'b0: SOP\_AFTER\_SOP\_VC\_3 error interrupt is non-fatal.
* **TRANS\_WITHOUT\_SOP\_VC\_3\_S**[12] -   
  1'b1: TRANS\_WITHOUT\_SOP\_VC\_3 error interrupt is fatal.  
  1'b0: TRANS\_WITHOUT\_SOP\_VC\_3 error interrupt is non-fatal.
* **CRD\_OFLW\_VC\_2\_S**[11] -   
  1'b1: CRD\_OFLW\_VC\_2 error interrupt is fatal.  
  1'b0: CRD\_OFLW\_VC\_2 error interrupt is non-fatal.
* **FIFO\_OVERFLOW\_VC\_2\_S**[10] -   
  1'b1: FIFO\_OVERFLOW\_VC\_2 error interrupt is fatal.  
  1'b0: FIFO\_OVERFLOW\_VC\_2 error interrupt is non-fatal.
* **SOP\_AFTER\_SOP\_VC\_2\_S**[9] -   
  1'b1: SOP\_AFTER\_SOP\_VC\_2 error interrupt is fatal.  
  1'b0: SOP\_AFTER\_SOP\_VC\_2 error interrupt is non-fatal.
* **TRANS\_WITHOUT\_SOP\_VC\_2\_S**[8] -   
  1'b1: TRANS\_WITHOUT\_SOP\_VC\_2 error interrupt is fatal.  
  1'b0: TRANS\_WITHOUT\_SOP\_VC\_2 error interrupt is non-fatal.
* **CRD\_OFLW\_VC\_1\_S**[7] -   
  1'b1: CRD\_OFLW\_VC\_1 error interrupt is fatal.  
  1'b0: CRD\_OFLW\_VC\_1 error interrupt is non-fatal.
* **FIFO\_OVERFLOW\_VC\_1\_S**[6] -   
  1'b1: FIFO\_OVERFLOW\_VC\_1 error interrupt is fatal.  
  1'b0: FIFO\_OVERFLOW\_VC\_1 error interrupt is non-fatal.
* **SOP\_AFTER\_SOP\_VC\_1\_S**[5] -   
  1'b1: SOP\_AFTER\_SOP\_VC\_1 error interrupt is fatal.  
  1'b0: SOP\_AFTER\_SOP\_VC\_1 error interrupt is non-fatal.
* **TRANS\_WITHOUT\_SOP\_VC\_1\_S**[4] -   
  1'b1: TRANS\_WITHOUT\_SOP\_VC\_1 error interrupt is fatal.  
  1'b0: TRANS\_WITHOUT\_SOP\_VC\_1 error interrupt is non-fatal.
* **CRD\_OFLW\_VC\_0\_S**[3] -   
  1'b1: CRD\_OFLW\_VC\_0 error interrupt is fatal.  
  1'b0: CRD\_OFLW\_VC\_0 error interrupt is non-fatal.
* **FIFO\_OVERFLOW\_VC\_0\_S**[2] -   
  1'b1: FIFO\_OVERFLOW\_VC\_0 error interrupt is fatal.  
  1'b0: FIFO\_OVERFLOW\_VC\_0 error interrupt is non-fatal.
* **SOP\_AFTER\_SOP\_VC\_0\_S**[1] -   
  1'b1: SOP\_AFTER\_SOP\_VC\_0 error interrupt is fatal.  
  1'b0: SOP\_AFTER\_SOP\_VC\_0 error interrupt is non-fatal.
* **TRANS\_WITHOUT\_SOP\_VC\_0\_S**[0] -   
  1'b1: TRANS\_WITHOUT\_SOP\_VC\_0 error interrupt is fatal.  
  1'b0: TRANS\_WITHOUT\_SOP\_VC\_0 error interrupt is non-fatal.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | PGE\_S | OPRT\_UNAVL\_S |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CRD\_OFLW\_VC\_7\_S | FIFO\_OVERFLOW\_VC\_7\_S | SOP\_AFTER\_SOP\_VC\_7\_S | TRANS\_WITHOUT\_SOP\_VC\_7\_S | CRD\_OFLW\_VC\_6\_S | FIFO\_OVERFLOW\_VC\_6\_S | SOP\_AFTER\_SOP\_VC\_6\_S | TRANS\_WITHOUT\_SOP\_VC\_6\_S | CRD\_OFLW\_VC\_5\_S | FIFO\_OVERFLOW\_VC\_5\_S | SOP\_AFTER\_SOP\_VC\_5\_S | TRANS\_WITHOUT\_SOP\_VC\_5\_S | CRD\_OFLW\_VC\_4\_S | FIFO\_OVERFLOW\_VC\_4\_S | SOP\_AFTER\_SOP\_VC\_4\_S | TRANS\_WITHOUT\_SOP\_VC\_4\_S | CRD\_OFLW\_VC\_3\_S | FIFO\_OVERFLOW\_VC\_3\_S | SOP\_AFTER\_SOP\_VC\_3\_S | TRANS\_WITHOUT\_SOP\_VC\_3\_S | CRD\_OFLW\_VC\_2\_S | FIFO\_OVERFLOW\_VC\_2\_S | SOP\_AFTER\_SOP\_VC\_2\_S | TRANS\_WITHOUT\_SOP\_VC\_2\_S | CRD\_OFLW\_VC\_1\_S | FIFO\_OVERFLOW\_VC\_1\_S | SOP\_AFTER\_SOP\_VC\_1\_S | TRANS\_WITHOUT\_SOP\_VC\_1\_S | CRD\_OFLW\_VC\_0\_S | FIFO\_OVERFLOW\_VC\_0\_S | SOP\_AFTER\_SOP\_VC\_0\_S | TRANS\_WITHOUT\_SOP\_VC\_0\_S |

Table 36 RSSB\_ERROR\_INTERRUPT\_SVRTY register.

#### RSSB\_ID

This register holds layer and position information for the RSSB. It is a read-only register. It can be used for debugging software access to the NoC elements by confirming that a read has successfully targeted the correct NoC element.

Attribute: R

Security: Non-secure

Bit field description:

* **ONE**[24] - One
* **ZERO**[23:21] - Zeroes
* **POS**[20:5] - 16-bit position ID of this RSSB in the NoC
* **LAYER**[4:0] - 5-bit identifier of the NoC layer on which this router is located

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| u | | | | | | | ONE | ZERO | | | POS | | | | | | | | | | | | | | | | LAYER | | | | |

Table 37 RSSB\_ID register.

#### RSSB\_IVC\_PORT

This register indicates the output port for each vc of an input port, vc 0 to 7.

Attribute: R

Security: Non-secure

Bit field description:

* **OUTP\_VC\_7**[39:35] - Output port for VC 7
* **OUTP\_VC\_6**[34:30] - Output port for VC 6
* **OUTP\_VC\_5**[29:25] - Output port for VC 5
* **OUTP\_VC\_4**[24:20] - Output port for VC 4
* **OUTP\_VC\_3**[19:15] - Output port for VC\_3
* **OUTP\_VC\_2**[14:10] - Output port for VC\_2
* **OUTP\_VC\_1**[9:5] - Output port for VC\_1
* **OUTP\_VC\_0**[4:0] - Output port for VC\_0

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| u | | | | | | | | | | | | | | | | | | | | | | | | OUTP\_VC\_7 | | | | | OUTP\_VC\_6 | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| OUTP\_VC\_6 | | OUTP\_VC\_5 | | | | | OUTP\_VC\_4 | | | | | OUTP\_VC\_3 | | | | | OUTP\_VC\_2 | | | | | OUTP\_VC\_1 | | | | | OUTP\_VC\_0 | | | | |

Table 38 RSSB\_IVC\_PORT register.

#### RSSB\_IVC\_STATUS

This register indicates the current status of a single input port of an rssb. Each register tracks the status of up to 8 virtual channels for the input port. There are 64 rssb\_ivc\_status per router, one for each rssb input port, for vc's 0-7.

Attribute: R

Security: Non-secure

Bit field description:

* **V\_VC\_7**[63] - 1'b1: Head flit valid (buffer ready)
* **F\_VC\_7**[62] - 1'b1: Buffer full
* **B\_VC\_7**[61] -   
  1'b1: Indicates that the head flit of the VC is of the 'QoS Barrier' type  
  1'b0: Indicates that the head flit of the VC is of the 'QoS Normal' type
* **S\_VC\_7**[60] -   
  1'b1: Indicates that the head flit is a start of packet. This also indicates that this input VC has not yet acquired its corresponding output VC  
  1'b0: Indicates that the head flit is not a start of packet. Also indicates that this input VC has already acquired the VC on the output port
* **UPSZ\_VC\_7**[59] -   
  1'b1: Indicates that the flit accumulator on this VC for upsizing to an output port is currently holding a flit  
  1'b0: Indicates that either the upsizing accumulator is empty or there is no upsizing from the VC
* **V\_VC\_6**[55] - 1'b1: Head flit valid (buffer ready)
* **F\_VC\_6**[54] - 1'b1: Buffer full
* **B\_VC\_6**[53] -   
  1'b1: Indicates that the head flit of the VC is of the 'QoS Barrier' type  
  1'b0: Indicates that the head flit of the VC is of the 'QoS Normal' type
* **S\_VC\_6**[52] -   
  1'b1: Indicates that the head flit is a start of packet. This also indicates that this input VC has not yet acquired its corresponding output VC  
  1'b0: Indicates that the head flit is not a start of packet. Also indicates that this input VC has already acquired the VC on the output port
* **UPSZ\_VC\_6**[51] -   
  1'b1: Indicates that the flit accumulator on this VC for upsizing to an output port is currently holding a flit  
  1'b0: Indicates that either the upsizing accumulator is empty or there is no upsizing from the VC
* **V\_VC\_5**[47] - 1'b1: Head flit valid (buffer ready)
* **F\_VC\_5**[46] - 1'b1: Buffer full
* **B\_VC\_5**[45] -   
  1'b1: Indicates that the head flit of the VC is of the 'QoS Barrier' type  
  1'b0: Indicates that the head flit of the VC is of the 'QoS Normal' type
* **S\_VC\_5**[44] -   
  1'b1: Indicates that the head flit is a start of packet. This also indicates that this input VC has not yet acquired its corresponding output VC  
  1'b0: Indicates that the head flit is not a start of packet. Also indicates that this input VC has already acquired the VC on the output port
* **UPSZ\_VC\_5**[43] -   
  1'b1: Indicates that the flit accumulator on this VC for upsizing to an output port is currently holding a flit  
  1'b0: Indicates that either the upsizing accumulator is empty or there is no upsizing from the VC
* **V\_VC\_4**[39] - 1'b1: Head flit valid (buffer ready)
* **F\_VC\_4**[38] - 1'b1: Buffer full
* **B\_VC\_4**[37] -   
  1'b1: Indicates that the head flit of the VC is of the 'QoS Barrier' type  
  1'b0: Indicates that the head flit of the VC is of the 'QoS Normal' type
* **S\_VC\_4**[36] -   
  1'b1: Indicates that the head flit is a start of packet. This also indicates that this input VC has not yet acquired its corresponding output VC  
  1'b0: Indicates that the head flit is not a start of packet. Also indicates that this input VC has already acquired the VC on the output port
* **UPSZ\_VC\_4**[35] -   
  1'b1: Indicates that the flit accumulator on this VC for upsizing to an output port is currently holding a flit  
  1'b0: Indicates that either the upsizing accumulator is empty or there is no upsizing from the VC
* **V\_VC\_3**[31] - 1'b1: Head flit valid (buffer ready)
* **F\_VC\_3**[30] - 1'b1: Buffer full
* **B\_VC\_3**[29] -   
  1'b1: Indicates that the head flit of the VC is of the 'QoS Barrier' type  
  1'b0: Indicates that the head flit of the VC is of the 'QoS Normal' type
* **S\_VC\_3**[28] -   
  1'b1: Indicates that the head flit is a start of packet. This also indicates that this input VC has not yet acquired its corresponding output VC  
  1'b0: Indicates that the head flit is not a start of packet. Also indicates that this input VC has already acquired the VC on the output port
* **UPSZ\_VC\_3**[27] -   
  1'b1: Indicates that the flit accumulator on this VC for upsizing to an output port is currently holding a flit  
  1'b0: Indicates that either the upsizing accumulator is empty or there is no upsizing from the VC
* **V\_VC\_2**[23] - 1'b1: Head flit valid (buffer ready)
* **F\_VC\_2**[22] - 1'b1: Buffer full
* **B\_VC\_2**[21] -   
  1'b1: Indicates that the head flit of the VC is of the 'QoS Barrier' type  
  1'b0: Indicates that the head flit of the VC is of the 'QoS Normal' type
* **S\_VC\_2**[20] -   
  1'b1: Indicates that the head flit is a start of packet. This also indicates that this input VC has not yet acquired its corresponding output VC  
  1'b0: Indicates that the head flit is not a start of packet. Also indicates that this input VC has already acquired the VC on the output port
* **UPSZ\_VC\_2**[19] -   
  1'b1: Indicates that the flit accumulator on this VC for upsizing to an output port is currently holding a flit  
  1'b0: Indicates that either the upsizing accumulator is empty or there is no upsizing from the VC
* **V\_VC\_1**[15] - 1'b1: Head flit valid (buffer ready)
* **F\_VC\_1**[14] - 1'b1: Buffer full
* **B\_VC\_1**[13] -   
  1'b1: Indicates that the head flit of the VC is of the 'QoS Barrier' type  
  1'b0: Indicates that the head flit of the VC is of the 'QoS Normal' type
* **S\_VC\_1**[12] -   
  1'b1: Indicates that the head flit is a start of packet. This also indicates that this input VC has not yet acquired its corresponding output VC  
  1'b0: Indicates that the head flit is not a start of packet. Also indicates that this input VC has already acquired the VC on the output port
* **UPSZ\_VC\_1**[11] -   
  1'b1: Indicates that the flit accumulator on this VC for upsizing to an output port is currently holding a flit  
  1'b0: Indicates that either the upsizing accumulator is empty or there is no upsizing from the VC
* **V\_VC\_0**[7] - 1'b1: Head flit valid (buffer ready)
* **F\_VC\_0**[6] - 1'b1: Buffer full
* **B\_VC\_0**[5] -   
  1'b1: Indicates that the head flit of the VC is of the 'QoS Barrier' type  
  1'b0: Indicates that the head flit of the VC is of the 'QoS Normal' type
* **S\_VC\_0**[4] -   
  1'b1: Indicates that the head flit is a start of packet. This also indicates that this input VC has not yet acquired its corresponding output VC  
  1'b0: Indicates that the head flit is not a start of packet. Also indicates that this input VC has already acquired the VC on the output port
* **UPSZ\_VC\_0**[3] -   
  1'b1: Indicates that the flit accumulator on this VC for upsizing to an output port is currently holding a flit  
  1'b0: Indicates that either the upsizing accumulator is empty or there is no upsizing from the VC
* **DWNSZ\_0**[0] -   
  1'b1: Indicates that the downsizer is busy.  
  1'b0: Indicates that the downsizer is not busy.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| V\_VC\_7 | F\_VC\_7 | B\_VC\_7 | S\_VC\_7 | UPSZ\_VC\_7 | u | | | V\_VC\_6 | F\_VC\_6 | B\_VC\_6 | S\_VC\_6 | UPSZ\_VC\_6 | u | | | V\_VC\_5 | F\_VC\_5 | B\_VC\_5 | S\_VC\_5 | UPSZ\_VC\_5 | u | | | V\_VC\_4 | F\_VC\_4 | B\_VC\_4 | S\_VC\_4 | UPSZ\_VC\_4 | u | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| V\_VC\_3 | F\_VC\_3 | B\_VC\_3 | S\_VC\_3 | UPSZ\_VC\_3 | u | | | V\_VC\_2 | F\_VC\_2 | B\_VC\_2 | S\_VC\_2 | UPSZ\_VC\_2 | u | | | V\_VC\_1 | F\_VC\_1 | B\_VC\_1 | S\_VC\_1 | UPSZ\_VC\_1 | u | | | V\_VC\_0 | F\_VC\_0 | B\_VC\_0 | S\_VC\_0 | UPSZ\_VC\_0 | u | | DWNSZ\_0 |

Table 39 RSSB\_IVC\_STATUS register.

#### RSSB\_OVC\_STATUS

This register indicates the current status of a single output port of an rssb. Each register tracks the status of up to 8 virtual channels for the output port. There are 32 regs rssb\_ovc\_status per router, one for each rssb output port, for vc's 0-7.

Attribute: R

Security: Non-secure

Bit field description:

* **VCO\_VC\_7**[31] - 1'b1: OVC occupied
* **NC\_VC\_7**[30] - 1'b1: No credits
* **FC\_VC\_7**[29] - 1'b1: Full credit
* **VCO\_VC\_6**[27] - 1'b1: OVC occupied
* **NC\_VC\_6**[26] - 1'b1: No credits
* **FC\_VC\_6**[25] - 1'b1: Full credit
* **VCO\_VC\_5**[23] - 1'b1: OVC occupied
* **NC\_VC\_5**[22] - 1'b1: No credits
* **FC\_VC\_5**[21] - 1'b1: Full credit
* **VCO\_VC\_4**[19] - 1'b1: OVC occupied
* **NC\_VC\_4**[18] - 1'b1: No credits
* **FC\_VC\_4**[17] - 1'b1: Full credit
* **VCO\_VC\_3**[15] - 1'b1: OVC occupied
* **NC\_VC\_3**[14] - 1'b1: No credits
* **FC\_VC\_3**[13] - 1'b1: Full credit
* **VCO\_VC\_2**[11] - 1'b1: OVC occupied
* **NC\_VC\_2**[10] - 1'b1: No credits
* **FC\_VC\_2**[9] - 1'b1: Full credit
* **VCO\_VC\_1**[7] - 1'b1: OVC occupied
* **NC\_VC\_1**[6] - 1'b1: No credits
* **FC\_VC\_1**[5] - 1'b1: Full credit
* **VCO\_VC\_0**[3] - 1'b1: OVC occupied
* **NC\_VC\_0**[2] - 1'b1: No credits
* **FC\_VC\_0**[1] - 1'b1: Full credit

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| VCO\_VC\_7 | NC\_VC\_7 | FC\_VC\_7 | u | VCO\_VC\_6 | NC\_VC\_6 | FC\_VC\_6 | u | VCO\_VC\_5 | NC\_VC\_5 | FC\_VC\_5 | u | VCO\_VC\_4 | NC\_VC\_4 | FC\_VC\_4 | u | VCO\_VC\_3 | NC\_VC\_3 | FC\_VC\_3 | u | VCO\_VC\_2 | NC\_VC\_2 | FC\_VC\_2 | u | VCO\_VC\_1 | NC\_VC\_1 | FC\_VC\_1 | u | VCO\_VC\_0 | NC\_VC\_0 | FC\_VC\_0 | u |

Table 40 RSSB\_OVC\_STATUS register.

#### RSSB\_XFC\_IN\_IDLE\_STATUS

This register observes the idle status of the input interface logic of this rssb. Note to read this register CSR/regbus logic must be active.

Attribute: R

Security: Non-secure

Bit field description:

* **P31I**[31] - P31 input idle.
* **P30I**[30] - P30 input idle.
* **P29I**[29] - P29 input idle.
* **P28I**[28] - P28 input idle.
* **P27I**[27] - P27 input idle.
* **P26I**[26] - P26 input idle.
* **P25I**[25] - P25 input idle.
* **P24I**[24] - P24 input idle.
* **P23I**[23] - P23 input idle.
* **P22I**[22] - P22 input idle.
* **P21I**[21] - P21 input idle.
* **P20I**[20] - P20 input idle.
* **P19I**[19] - P19 input idle.
* **P18I**[18] - P18 input idle.
* **P17I**[17] - P17 input idle.
* **P16I**[16] - P16 input idle.
* **P15I**[15] - P15 input idle.
* **P14I**[14] - P14 input idle.
* **P13I**[13] - P13 input idle.
* **P12I**[12] - P12 input idle.
* **P11I**[11] - P11 input idle.
* **P10I**[10] - P10 input idle.
* **P9I**[9] - P9 input idle.
* **P8I**[8] - P8 input idle.
* **P7I**[7] - P7 input idle.
* **P6I**[6] - P6 input idle.
* **P5I**[5] - P5 input idle.
* **P4I**[4] - P4 input idle.
* **P3I**[3] - P3 input idle.
* **P2I**[2] - P2 input idle.
* **P1I**[1] - P1 input idle.
* **P0I**[0] - P0 input idle.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| P31I | P30I | P29I | P28I | P27I | P26I | P25I | P24I | P23I | P22I | P21I | P20I | P19I | P18I | P17I | P16I | P15I | P14I | P13I | P12I | P11I | P10I | P9I | P8I | P7I | P6I | P5I | P4I | P3I | P2I | P1I | P0I |

Table 41 RSSB\_XFC\_IN\_IDLE\_STATUS register.

#### RSSB\_XFC\_OUT\_IDLE\_STATUS

This register observes the idle status of the output interface logic of this rssb. Note to read this register CSR/regbus logic must be active.

Attribute: R

Security: Non-secure

Bit field description:

* **P31O**[31] - P31 output idle.
* **P30O**[30] - P30 output idle.
* **P29O**[29] - P29 output idle.
* **P28O**[28] - P28 output idle.
* **P27O**[27] - P27 output idle.
* **P26O**[26] - P26 output idle.
* **P25O**[25] - P25 output idle.
* **P24O**[24] - P24 output idle.
* **P23O**[23] - P23 output idle.
* **P22O**[22] - P22 output idle.
* **P21O**[21] - P21 output idle.
* **P20O**[20] - P20 output idle.
* **P19O**[19] - P19 output idle.
* **P18O**[18] - P18 output idle.
* **P17O**[17] - P17 output idle.
* **P16O**[16] - P16 output idle.
* **P15O**[15] - P15 output idle.
* **P14O**[14] - P14 output idle.
* **P13O**[13] - P13 output idle.
* **P12O**[12] - P12 output idle.
* **P11O**[11] - P11 output idle.
* **P10O**[10] - P10 output idle.
* **P9O**[9] - P9 output idle.
* **P8O**[8] - P8 output idle.
* **P7O**[7] - P7 output idle.
* **P6O**[6] - P6 output idle.
* **P5O**[5] - P5 output idle.
* **P4O**[4] - P4 output idle.
* **P3O**[3] - P3 output idle.
* **P2O**[2] - P2 output idle.
* **P1O**[1] - P1 output idle.
* **P0O**[0] - P0 output idle.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| P31O | P30O | P29O | P28O | P27O | P26O | P25O | P24O | P23O | P22O | P21O | P20O | P19O | P18O | P17O | P16O | P15O | P14O | P13O | P12O | P11O | P10O | P9O | P8O | P7O | P6O | P5O | P4O | P3O | P2O | P1O | P0O |

Table 42 RSSB\_XFC\_OUT\_IDLE\_STATUS register.

#### RSSB\_QOS\_WEIGHT

This register describes the weight value of each QoS supported at the rssb. Each byte of this register must be greater than or equal to 3. Each transmitting rssb supports up to 16 QoS profiles. Each QoS is composed of pri and weight, however only the weight is programmable, therefore is part of the registers.QoS data is composed of four registers, rssb\_qos\_weight\_0, rssb\_qos\_weight\_1, rssb\_qos\_weight\_2 and rssb\_qos\_weight\_3, each of which contains the weight of four profiles. Depending upon how many QoS profiles are enabled, the appropriate bits in the following registers are available.

Attribute: RW

Security: Non-secure

Bit field description:

* **WT\_QOS\_3**[31:24] - OQS weight 3
* **WT\_QOS\_2**[23:16] - OQS weight 2
* **WT\_QOS\_1**[15:8] - OQS weight 1
* **WT\_QOS\_0**[7:0] - OQS weight 0

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| WT\_QOS\_3 | | | | | | | | WT\_QOS\_2 | | | | | | | | WT\_QOS\_1 | | | | | | | | WT\_QOS\_0 | | | | | | | |

Table 43 RSSB\_QOS\_WEIGHT register.

## Appendix C: IDI/CMI Protocol Support

### IDI Feature Adoption

The table below provides list of supported IDI opcodes and address range type by CFG NOC. Upon detection of un-supported opcode, the bridge will trigger decode error and terminate the transaction gracefully.

|  |  |  |  |
| --- | --- | --- | --- |
| C2U Opcode | Coherent Address Range | Mbox | Non-coherent Address Range |
| AtomicWr |  |  |  |
| AtomicRdWr |  |  |  |
| AtomicRd |  |  |  |
| CLDemote | X | X |  |
| CLFlush | X | X |  |
| CLFlushOPT | X | X |  |
| ClrMonitor | \* | \* | \* |
| CLWB | X | X |  |
| CRd | X | X |  |
| CRD\_PREF | X | X |  |
| CRD\_UC | X | X | X |
| DRd | X | X | X |
| DRD\_NS | X | X |  |
| DRD\_OPT | X | X | X |
| DRD\_OPT\_PREF | X | X |  |
| DRD\_PREF | X | X |  |
| DRdPTE | X | X |  |
| Enqueue |  |  |  |
| EOI |  | X |  |
| IntA |  | X |  |
| IntLog |  | X |  |
| IntPhy |  | X |  |
| IntPriUp |  | X |  |
| ItoM | X | X |  |
| ItoMWr | X | X |  |
| ItoMWr\_NS | X | X |  |
| ItoMWr\_Wt | X | X |  |
| ItoMWR\_WT\_NS | X | X |  |
| LLCInv |  |  |  |
| LlcPrefCode |  |  |  |
| LlcPrefData |  |  |  |
| LlcPrefRFO |  |  |  |
| LLCWB |  |  |  |
| LLCWBInv |  |  |  |
| Lock |  | X |  |
| MemPushWr | X | X |  |
| MemPushWr\_NS | X | X |  |
| NOP |  | X |  |
| Pcommit |  |  |  |
| PortIn |  | X |  |
| PortOut |  | X |  |
| PRd | X | X | X |
| RdCurr | X | X |  |
| RFO | X | X |  |
| RFO\_PREF | X | X |  |
| RFOWr |  |  |  |
| SetMonitor | X | X |  |
| SpCyc |  | X |  |
| SpecItoM | X | X |  |
| SplitLock |  | X |  |
| UcRdF | X | X |  |
| Unlock |  | X |  |
| WbEFtoE | X | X |  |
| WbEFtoI | X | X |  |
| WbMtoE | X | X |  |
| WbMtoI | X | X |  |
| WbOtoE |  |  |  |
| WbOtoI |  |  |  |
| WbStoI | X | X |  |
| WCiL | X | X |  |
| WCIL\_NS | X | X |  |
| WcilF | X | X |  |
| WCILF\_NS | X | X |  |
| WiL | X | X | X |

\* - Treated as NOP instruction

The table below provides implementation of C2U and U2C straps.

|  |  |
| --- | --- |
| C2U Strap | CFG IDI NoC Support |
| Hash | Not supported |
| Parity | Not transported over NoC, regenerated at destination |
| EnhParity | Not transported over NoC, regenerated at destination |
| ECC | Not supported |
| Owned | Not supported |
| DataHdrSep | Fully Supported |
| SupportStall | Fully Supported |

|  |  |
| --- | --- |
| U2C Strap | CFG IDI NoC Support |
| CleanEvict | Not Supported |
| Parity | Not transported over NoC, regenerated at destination |
| EnhParity | Not transported over NoC, regenerated at destination |
| ECC | Not Supported |
| FLSupport | Fully Supported, if slave agents does not support Full Line functionality, slave bridge can do regeneration of byte enable and data |
| SupportStall | Fully Supported |

### CMI Feature Adoption

The table below summarizes CMI features (spec 1.1) supported by CFG NOC.

| CMI parameter | CFG CMI NoC Support |
| --- | --- |
| ADDR\_MSB | Supported range 18-59 |
| ADDR\_LSB | The IP supports 0-6 |
| ADDR\_WIDTH | 14-60 |
| CHANCT | The IP supports 1-4 VCs. |
| CHID\_MSB | Supports 0-1 |
| NUM\_TIDS | The IP supports 64-2047. Num\_TIDs is automatically computed by Nocstudio. |
| TID\_MSB | The IP supports from 1 bit to 63 bit TID (TID\_MSB=0-62). |
| LINE\_SIZE | The IP supports line size of 64B |
| WR\_DWIDTH | The agent can take 8B, 16B, 32B and 64B interface widths |
| RD\_DWIDTH | The agent can take 8B, 16B, 32B and 64B interface widths |
| WR\_DCREDIT | IP supports for 64 |
| RD\_DCREDIT | The IP supports for 32 (non-64B interface) and 64 (64B interface) |
| WDATA\_DELAY | The IP supports fixed delay mode with any WDATA\_DELAY between 0 & 10 |
| RDATA\_DELAY | The IP supports fixed delay mode with any RDATA\_DELAY between 0 & 10 |
| AGNTID\_MSB | The IP supports 2-11 |
| LBINFO\_MSB | The IP supports 0-30 (Master bridge supports 0-30, Slave bridge supports 1-32) |
| FDATA\_ID\_MSB | Data forwarding is not supported. |
| PSC\_MSB | The IP supports 0,9 |
| PSR\_MSB | The IP supports 0,9 |
| CLOS\_MSB | The IP supports 0,2 |
| MAX\_INTLV\_CPL | The IP supports a value of 0-16 |
| REQ\_METADATA\_WIDTH | The IP supports value of only 1. But Meta data is not transported. |
| RD\_METADATA\_WIDTH | The IP supports value of only 1. But Meta data is not transported. |
| WR\_METADATA\_WIDTH | The IP supports value of only 1. But Meta data is not transported. |
| RSP\_METADATA\_WIDTH | The IP supports value of only 1. But Meta data is not transported. |
| req\_length | Supports both 32B and 64B transactions |
| dst\_id | Supports both ID based lookup and Address based look up followed by static hashing |
| Ordering | Un ordered CMI NOC. |
| Clock crossing | Clock crossing supported between Bridge and Router but not at protocol Interface |
| Credit Initialization | The IP supports only long Init. |
| Opcodes | MRd, MWr and MWrPtl, MPCmt and NDTC. Write0 is supported. |

### IDI2CMI Feature Adoption

#### IDI2CMI Converter Signal List

|  |  |  |  |
| --- | --- | --- | --- |
| Signal | Direction | Option (Strap) | Width |
| idi2cmi\_clk | Input |  | 1 |
| idi2cmi\_reset\_n | Input |  | 1 |
| tst\_rst | Input |  | 1 |
| tst\_rst\_bypass | input |  | 1 |
| c2u\_req\_valid | Input |  | 1 |
| c2u\_req\_validmirror | Input | Enhparity  P\_IDI\_STRAP\_ENH\_PARITY\_EN | 0 or 1 |
| c2u\_req\_reqparity | Input | Enhparity  P\_IDI\_STRAP\_ENH\_PARITY\_EN | 0 or 1 |
| c2u\_req\_opgroup | Input |  | 2 |
| c2u\_req\_opcode | Input |  | 6 |
| c2u\_req\_address | Input |  | P\_MST\_ADDR\_WIDTH |
| c2u\_req\_addrparity | Input | Parity  P\_IDI\_STRAP\_PARITY\_EN | 0 or 1 |
| c2u\_req\_slfsnp | input |  | 1 |
| c2u\_req\_cqid | input |  | 12 |
| c2u\_req\_length | input |  | 6 |
| c2u\_req\_lpid | input |  | 3 |
| c2u\_req\_clos | input |  | 5 |
| c2u\_req\_rmid | input |  | 10 |
| c2u\_req\_sai | input |  | 8 |
| c2u\_req\_secure | input |  | 1 |
| c2u\_req\_nontemporal | input |  | 2 |
| c2u\_req\_cachenear | input |  | 1 |
| c2u\_req\_cachefar | input |  | 1 |
| c2u\_req\_hash | input | Hash  P\_IDI\_STRAP\_HASH\_EN | 0 or 6 |
| c2u\_req\_topology | input | Hash  P\_IDI\_STRAP\_HASH\_EN | 0 or 4 |
| c2u\_req\_trivialdata | input |  | 1 |
| c2u\_req\_spare | input |  | 8 |
| c2u\_req\_srcid | input |  | 2 |
| c2u\_data\_valid | input |  | 1 |
| c2u\_data\_validmirror | input | Enhparity  P\_IDI\_STRAP\_ENH\_PARITY\_EN | 0 or 1 |
| c2u\_data\_datahdrparity | input | Enhparity  P\_IDI\_STRAP\_ENH\_PARITY\_EN | 0 or 1 |
| c2u\_data\_uqid | input |  | 13 |
| c2u\_data\_chunkvalid | input |  | 4 |
| c2u\_data\_data | input |  | P\_IDI\_DATA\_WIDTH |
| c2u\_data\_byteenable | input |  | P\_IDI\_DATA\_WIDTH/8 |
| c2u\_data\_bogus | input |  | 1 |
| c2u\_data\_poison | input |  | 1 |
| c2u\_data\_dataparity | input | Parity  P\_IDI\_STRAP\_PARITY\_EN | P\_IDI\_DATA\_WIDTH/64 |
| c2u\_data\_byteenableparity | input | Parity  P\_IDI\_STRAP\_PARITY\_EN | 0 or 1 |
| c2u\_data\_ecc | input | ecc  P\_IDI\_STRAP\_ECC\_EN | 5\*(P\_IDI\_DATA\_WIDTH/128) |
| c2u\_data\_eccvalid | input | ecc  P\_IDI\_STRAP\_ECC\_EN | 0 or 1 |
| c2u\_data\_fullline | input |  | 1 |
| c2u\_data\_spare | input |  | 2 |
| c2u\_strap\_version | input |  | 5 |
| c2u\_strap\_hash | input | P\_MUL\_LVL\_COH\_PORT | 1 |
| c2u\_strap\_parity | input | Parity  P\_IDI\_STRAP\_PARITY\_EN | 1 |
| c2u\_strap\_enhparity | input | Enhparity  P\_IDI\_STRAP\_ENH\_PARITY\_EN | 1 |
| c2u\_strap\_ecc | input |  | 1 |
| c2u\_strap\_owned | input |  | 1 |
| c2u\_strap\_datahdrsep | input |  | 1 |
| c2u\_strap\_supportstall | input |  | 1 |
| c2u\_strap\_spare | input |  | 4 |
| c2u\_credit\_ret\_rsp | input |  | 1 |
| c2u\_credit\_ret\_data | input |  | 1 |
| rsp\_valid | input |  | 1 |
| rsp\_chid | input |  | 1 |
| rsp\_type | input |  | 3 |
| rsp\_srcid | input |  | 3 |
| rsp\_tid | input |  | 7 |
| rsp\_lbinfo | input |  | 1 |
| rsp\_valid\_early | input |  | 1 |
| rsp\_fdata | input |  | 1 |
| rsp\_psr | input |  | 1 |
| rsp\_error\_type | input |  | 2 |
| rsp\_tid\_parity | input |  | 1 |
| rsp\_tid\_parity\_valid | input |  | 1 |
| rsp\_metadata | input |  | 1 |
| rd\_cpl\_valid | input |  | 1 |
| rd\_cpl\_chid | input |  | 1 |
| rd\_cpl\_srcid | input |  | 3 |
| rd\_cpl\_tid | input |  | 5 |
| rd\_cpl\_lbinfo | input |  | 1 |
| rd\_cpl\_ofst | input |  | 1 |
| rd\_cpl\_valid\_early | Input |  | 1 |
| rd\_cpl\_tid\_parity | Input |  | 1 |
| rd\_cpl\_tid\_parity\_valid | Input |  | 1 |
| rd\_cpl\_data\_valid | input |  | 1 |
| rd\_cpl\_data | input | 8\*P\_CMI\_RD\_DWIDTH | 8\*32 |
| rd\_cpl\_data\_valid\_early | Input |  | 1 |
| rd\_cpl\_dir\_state | Input |  | 2 |
| rd\_cpl\_metadata | input |  | 1 |
| rd\_cpl\_error | input |  | 1 |
| rd\_cpl\_data\_error\_type | input |  | 2 |
| rd\_cpl\_data\_poison | input |  | 1 |
| rd\_cpl\_parity | input | P\_CMI\_RD\_DWIDTH/8 | 32/8 |
| rd\_cpl\_parity\_valid | input |  | 1 |
| rd\_cpl\_ecc | input |  | 10 |
| rd\_cpl\_ecc\_valid | input |  | 1 |
| rd\_credit\_put | input |  | 1 |
| rd\_credit\_chid | input |  | 1 |
| wr\_credit\_put | input |  | 1 |
| wr\_credit\_chid | input |  | 1 |
| req\_fab\_credit\_put | input |  | 1 |
| req\_stall | input |  | 1 |
| rsp\_rd\_cpl\_stall\_ack | input |  | 1 |
| ism\_responder | input |  | 3 |
| system\_clk\_en | input |  | 1 |
| u2c\_credit\_ret\_req | output |  | 1 |
| u2c\_credit\_ret\_data | output |  | 1 |
| u2c\_rsp\_valid | output |  | 1 |
| u2c\_rsp\_validmirror | output | Enhparity  P\_IDI\_STRAP\_ENH\_PARITY\_EN | 0 or 1 |
| u2c\_rsp\_rspparity | output | Enhparity  P\_IDI\_STRAP\_ENH\_PARITY\_EN | 0 or 1 |
| u2c\_rsp\_opcode | output |  | 4 |
| u2c\_rsp\_rspdata | output |  | 13 |
| u2c\_rsp\_pre | output |  | 3 |
| u2c\_rsp\_cqid | output |  | 12 |
| u2c\_rsp\_spare | output |  | 2 |
| u2c\_rsp\_tgtid | output | P\_MUL\_LVL\_COH\_PORT | 2 |
| u2c\_data\_valid | output |  | 1 |
| u2c\_data\_validmirror | output | Enhparity  P\_IDI\_STRAP\_ENH\_PARITY\_EN | 0 or 1 |
| u2c\_data\_datahdrparity | output | Enhparity  P\_IDI\_STRAP\_ENH\_PARITY\_EN | 0 or 1 |
| u2c\_data\_cqid | output |  | 12 |
| u2c\_data\_tgtid | output |  | 2 |
| u2c\_data\_chunkvalid | output |  | 4 |
| u2c\_data\_data | output |  | 256 |
| u2c\_data\_poison | output |  | 1 |
| u2c\_data\_dataerr | output |  | 1 |
| u2c\_data\_pre | output |  | 7 |
| u2c\_data\_dataparity | output | Parity  P\_IDI\_STRAP\_PARITY\_EN | 4 |
| u2c\_data\_ecc | output | ecc  P\_IDI\_STRAP\_ECC\_EN | 10 |
| u2c\_data\_eccvalid | output | ecc  P\_IDI\_STRAP\_ECC\_EN | 1 |
| u2c\_data\_spare | output |  | 1 |
| u2c\_strap\_version | output |  | 5 |
| u2c\_strap\_cleanevict | output |  | 1 |
| u2c\_strap\_parity | output | Parity  P\_IDI\_STRAP\_PARITY\_EN | 1 |
| u2c\_strap\_enhparity | output | Enhparity  P\_IDI\_STRAP\_ENH\_PARITY\_EN | 1 |
| u2c\_strap\_ecc | output | ecc  P\_IDI\_STRAP\_ECC\_EN | 1 |
| u2c\_strap\_flsupport | output |  | 1 |
| u2c\_strap\_supportstall | output |  | 1 |
| u2c\_strap\_spare | output |  | 5 |
| req\_valid | output |  | 1 |
| req\_chid | output |  | 1 |
| req\_priority | output |  | 2 |
| req\_opcode | output |  | 3 |
| req\_address | output |  | 60 |
| req\_srcid | output |  | 3 |
| req\_tid | output |  | 5 |
| req\_dstid | output |  | 3 |
| req\_lbinfo | output |  | 1 |
| req\_pmem\_region | output |  | 1 |
| req\_secure\_region | output |  | 2 |
| req\_srsp | output |  | 1 |
| req\_uncacheable | output |  | 1 |
| req\_valid\_early | output |  | 1 |
| req\_tunnel | output |  | 1 |
| req\_no\_addr | output |  | 1 |
| req\_psc | output |  | 1 |
| req\_fdata | output |  | 1 |
| req\_fdata\_type | output |  | 2 |
| req\_fdata\_id | output |  | 1 |
| req\_chain | output |  | 1 |
| req\_no\_allocate\_nm | output |  | 1 |
| req\_no\_fetch\_fm | output |  | 1 |
| req\_specrd | output |  | 1 |
| req\_length | output |  | 1 |
| req\_clos | output |  | 1 |
| req\_no\_lookup | output |  | 1 |
| req\_dir\_only\_hint | output |  | 1 |
| req\_directory\_update | output |  | 2 |
| req\_GT | output |  | 1 |
| req\_mirror | output |  | 1 |
| req\_primary | output |  | 1 |
| req\_mirror\_failover | output |  | 1 |
| req\_address\_parity | output |  | 1 |
| req\_address\_parity\_valid | output |  | 1 |
| req\_tid\_parity | output |  | 1 |
| req\_tid\_parity\_valid | output |  | 1 |
| req\_metadata | output |  | 1 |
| req\_wdata\_valid | output |  | 1 |
| req\_wdata | output | 8\*P\_CMI\_WR\_DWIDTH | 8\*32 |
| req\_wbe | output |  | 32 |
| req\_wdata\_valid\_early | output |  | 1 |
| req\_wdata\_dir | output |  | 2 |
| req\_wdata\_metadata | output |  | 1 |
| req\_wdata\_poison | output |  | 1 |
| req\_wdata\_parity | output | P\_CMI\_WR\_DWIDTH/8 | 4 |
| req\_wdata\_parity\_valid | output |  | 1 |
| req\_wdata\_ecc | output |  | 10 |
| req\_wdata\_ecc\_valid | output |  | 1 |
| req\_wbe\_parity | output |  | 1 |
| req\_wbe\_parity\_valid | output |  | 1 |
| rd\_cpl\_fab\_credit\_put | output |  | 1 |
| rsp\_fab\_credit\_put | output |  | 1 |
| rd\_cpl\_credit\_put | output |  | 1 |
| rd\_cpl\_credit\_chid | output |  | 1 |
| rsp\_credit\_put | output |  | 1 |
| rsp\_credit\_chid | output |  | 1 |
| req\_stall\_ack | output |  | 1 |
| rsp\_rd\_cpl\_stall | output |  | 1 |
| ism\_requester | output |  | 3 |

#### IDI2CMI Converter Parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Legal values | Description |
| P\_CG\_WATCHDOG\_TIMER | 16 | Hysteresis count indicates the no of cycles for which the bridge has to be idle after which the clk gets gated. |
| P\_CLK\_SYNCHRONIZER\_DEPTH | 2 | 2 stage flop synchronizer |
| P\_CMI\_ADDR\_LSB | 2-6 | Refer CMI Spec 1.1 |
| P\_CMI\_ADDR\_MSB | 30-51 | Refer CMI Spec 1.1 |
| P\_CMI\_ADDR\_WIDTH | 25-50 | Refer CMI Spec 1.1 |
| P\_CMI\_AGNTID\_WIDTH | 2-5 | Refer CMI Spec 1.1 |
| P\_CMI\_CHANCT | 1-4 | Refer CMI Spec 1.1 |
| P\_CMI\_CHID\_WIDTH | 1,0 | Refer CMI Spec 1.1 |
| P\_CMI\_CLOS\_WIDTH | 0,2 | Refer CMI Spec 1.1 |
| P\_CMI\_CREDIT\_CNTR\_WIDTH | 8 | Width of CMI credit counter, max CMI credits to 256. |
| P\_CMI\_DSTID | 0 | For VAL purpose, not used in RTL |
| P\_CMI\_FDATA\_ID\_WIDTH | 20 | Refer CMI Spec 1.1 |
| P\_CMI\_FLOW\_CTL\_CON\_SPARES\_WIDTH | 4 | Spares are to be used for future feature growth and bug fixes |
| P\_CMI\_FLOW\_CTL\_PRO\_SPARES\_WIDTH | 4 | Spares are to be used for future feature growth and bug fixes |
| P\_CMI\_LBINFO\_WIDTH | 0-32 | Refer CMI Spec 1.1 |
| P\_CMI\_LINE\_SIZE | 64 | Refer CMI Spec 1.1 |
| P\_CMI\_MASTER\_ID | 3 | For VAL purpose, not used in RTL |
| P\_CMI\_MAX\_INTRLV\_CPL | 0,8-16 | Refer CMI Spec 1.1 |
| P\_CMI\_NUM\_TIDS | 64-2048 | Refer CMI Spec 1.1 |
| P\_CMI\_PSC\_WIDTH | 0,9 | Refer CMI Spec 1.1 |
| P\_CMI\_PSR\_WIDTH | 0,9 | Refer CMI Spec 1.1 |
| P\_CMI\_RDATA\_DELAY | 0,8 | Refer CMI Spec 1.1 |
| P\_CMI\_RD\_CPL\_CMD\_SPARES\_WIDTH | 4 | Spares are to be used for future feature growth and bug fixes |
| P\_CMI\_RD\_CPL\_CREDIT | 7 | No of CMI Rd cpl channel credits |
| P\_CMI\_RD\_CPL\_DATA\_SPARES\_WIDTH | 2 | Spares are to be used for future feature growth and bug fixes |
| P\_CMI\_RD\_DCREDIT | 32,64 | Refer CMI Spec 1.1 |
| P\_CMI\_RD\_DWIDTH | 8,16,32,64 | Refer CMI Spec 1.1 |
| P\_CMI\_RD\_METADATA\_WIDTH | 1 | Refer CMI Spec 1.1 |
| P\_CMI\_REQ\_CMD\_SPARES\_WIDTH | 4 | Spares are to be used for future feature growth and bug fixes |
| P\_CMI\_REQ\_METADATA\_WIDTH | 1 | Refer CMI Spec 1.1 |
| P\_CMI\_RSP\_CREDIT | 16 | No of CMI rsp channel credits |
| P\_CMI\_RSP\_METADATA\_WIDTH | 1 | Refer CMI Spec 1.1 |
| P\_CMI\_RSP\_SPARES\_WIDTH | 4 | Spares are to be used for future feature growth and bug fixes |
| P\_CMI\_SRC\_ID | 0 | The value that needs to be sent on req\_srcid signal on CMI request channel. |
| P\_CMI\_TID\_WIDTH | 2-11 | Refer CMI Spec 1.1 (only 6-11 is allowed as per spec) |
| P\_CMI\_WDATA\_DELAY | 2 | Refer CMI Spec 1.1 |
| P\_CMI\_WDATA\_SPARES\_WIDTH | 2 | Spares are to be used for future feature growth and bug fixes |
| P\_CMI\_WR\_DCREDIT | 64 | Refer CMI Spec 1.1 |
| P\_CMI\_WR\_DWIDTH | 32 | Refer CMI Spec 1.1 |
| P\_CMI\_WR\_METADATA\_WIDTH | 1 | Refer CMI Spec 1.1 |
| P\_COARSE\_CLOCK\_GATE | 0 | Course clock gate enable parameter default value as ‘0’ |
| P\_E2E\_EXTEND\_FROM\_PREV\_STAGE | 1 | For VAL purpose, not used in RTL |
| P\_E2E\_EXTEND\_TO\_NEXT\_STAGE | 1 | For VAL purpose, not used in RTL |
| P\_EARLY\_VALID\_ENABLE | 1 | Early valid feature enable parameter, default value is ‘1’ |
| P\_IDIS\_UQID\_MASK | 2048 | Mask provided for UQID generating with IDI slave bridge ID. |
| P\_IDI\_BYTE\_ENABLE | 16/32/64 | C2U data byte enable parameter |
| P\_IDI\_C2U\_DATA\_IF\_CREDITS | 4 | No of C2U data interface credits |
| P\_IDI\_C2U\_REQ\_ADDR\_WIDTH | 36-52 | C2U req address width parameter |
| P\_IDI\_C2U\_REQ\_IF\_CREDITS | 4 | No of C2U req interface credits |
| P\_IDI\_C2U\_REQ\_SPARE\_WIDTH | 8 | Spares are to be used for future feature growth and bug fixes |
| P\_IDI\_C2U\_RSP\_IF\_CREDITS | 4 | No of C2U rsp interface credits |
| P\_IDI\_CQID\_WIDTH | 12 | IDI CQID width. |
| P\_IDI\_DATA\_PARITY\_WIDTH | 0/2/4/8 | C2U data parity width parameter |
| P\_IDI\_DATA\_WIDTH | 128/256/512 | C2U data width parameter |
| P\_IDI\_ECC\_WIDTH | 0/5/10/20 | ECC width parameter |
| P\_IDI\_MAX\_U2C\_CREDITS | 16 | Max no of U2C credits |
| P\_IDI\_NUM\_OUTSTANDING | 128 | No of outstanding Write pull request that can be tracked by the converter. |
| P\_IDI\_RD\_DATA\_PARITY\_WIDTH | 0/2/4/8 | U2C data parity width |
| P\_IDI\_RD\_DATA\_WIDTH | 128/256/512 | U2C data width |
| P\_IDI\_RD\_ECC\_WIDTH | 0/5/10/20 | U2C ECC width |
| P\_IDI\_SLAVE\_ID | 8 | For VAL purpose, not used in RTL |
| P\_IDI\_SRCID | 2 | Used as target id in U2C\_RSP channel |
| P\_IDI\_U2C\_REQ\_ADDR\_WIDTH | 33-49 | U2C req address width |
| P\_PARITY\_CHECK\_EN | 0 | Parameter to enable/disable parity gen and parity err gen inside module |