|  |
| --- |
| NetSpeed Register Bus Architecture Specification |

NetSpeed Register Bus Architecture Specification

About This Document

This document describes the Architecture specification of Register Bus. This document includes feature set, block diagram, micro-architecture description, pinout and parameters used in Streaming Bridge design

Audience

This document is intended for users of NetSpeed’s IP:

* NoC Architects
* NoC Designers
* NoC Verification Engineers

Prerequisite

Before proceeding, you should generally understand:

* Basics of Network on Chip technology
* NetSpeed Streaming Interface Specification

Related Documents

The following documents can be used as a reference to this document.

* NocStudio User Manual
* NetSpeed Streaming Interface Specification

Customer Support

For technical support about this product, please contact your local NetSpeed sales office, representative, or distributor.

For general information about NetSpeed products refer to: [www.netspeedsystems.com](http://www.netspeedsystems.com)

Contents

[About This Document 2](#_Toc373931278)

[Audience 2](#_Toc373931279)

[Prerequisite 2](#_Toc373931280)

[Related Documents 2](#_Toc373931281)

[Customer Support 2](#_Toc373931282)

[Acronyms 6](#_Toc373931283)

[1 Overview 9](#_Toc373931284)

[2 Description 10](#_Toc373931285)

[3 Register Bus Interfaces 12](#_Toc373931286)

[4 Register List and Summary 14](#_Toc373931287)

[4.1 Configuration Registers 17](#_Toc373931288)

[4.1.1 axi4\_slave\_bridge\_base\_addr (B) 17](#_Toc373931289)

[4.1.2 qos\_profile\_data (P) 18](#_Toc373931290)

[4.2 Debug and Performance Monitor 19](#_Toc373931291)

[4.2.1 noc\_injection\_flit\_count (I), noc\_ejection\_flit\_count (E), bridge\_transmit\_beat\_count (T), and bridge\_receive\_beat\_count (R) 19](#_Toc373931292)

[4.2.2 bridge\_transmit\_fifo\_status (BTS) and bridge\_receive\_fifo\_status (BRS) 20](#_Toc373931293)

[4.2.3 router\_input\_vc\_status (RIVCS) and router\_output\_vc\_status (ROVCS) 21](#_Toc373931294)

[4.3 Error Log Registers 22](#_Toc373931295)

[4.3.1 bridge\_transmit\_fifo\_overflow (BTOE) and bridge\_receive\_fifo\_overflow (BROE) 22](#_Toc373931296)

[4.3.2 Router\_error (RERRS) 22](#_Toc373931297)

[4.3.3 Transaction errors (XE[0], XE[1], XE[2]): transaction\_valid\_without\_sop, transaction\_sop\_after\_sop, transaction\_illegal\_dest\_qos 22](#_Toc373931298)

[4.4 Address Map of Registers 23](#_Toc373931299)

[4.5 Ring bus Protocol 24](#_Toc373931300)

[4.5.1 Ring bus Commands 24](#_Toc373931301)

[4.5.2 Ring bus Protocol 24](#_Toc373931302)

List of Figures

[Figure 1: Register Bus block diagram 10](#_Toc373931303)

[Figure 2: Address decoding and register address space for various NoC nodes and modules within the nodes 23](#_Toc373931304)

[Figure 3: Ring bus Protocol 24](#_Toc373931305)

List of Tables

[Table 1: Interface Signals 13](#_Toc373931306)

[Table 2: NoC Register List and Summary 17](#_Toc373931307)

[Table 3: Ring bus commands 24](#_Toc373931308)

# Acronyms

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | NoC | Network on Chip | | SoC | System on Chip | | Host | An IP core, component, or device sitting in an SoC | | Hostport | A port of a host that connects to NoC router’s injection and ejection port via a bridge to be able to inject traffic into NoC or eject traffic from NoC | | Interface | Sets of signals to receive or transmit transaction messages of a hostport; a hostport may contain multiple interfaces; each interface may be uni-directional i.e. it sends or receives transaction messages or bidirectional i.e. it both receives and transmits transaction messages | | Router | A hardware switch at the cross point of a mesh connecting to up to 4 of its neighboring routers and one or more bridges to connect to one or more hostports | | Bridge | Sits between a router’s port (often injection/ejection) and a hostport of a host to convert the hostports signal protocol (such as AMBA AXI-4) to NoC packet format and vice-versa and additional operations needed by the signaling protocol such as width conversion, etc. | | NetSpeed Streaming Interface/Protocol | Signaling protocol provided by NetSpeed NoC streaming bridges; it is a simple credit based bidirectional interface for hosts to inject and eject messages into/from NoC | | NetSpeed Streaming Bridge | Sits between a router’s port (often injection/ejection) and a NetSpeed streaming hostport of a host to convert the hostports signal to NoC packet format and vice-versa and additional operations needed by the protocol such as width conversion, etc.; there are 4 physical bidirectional NetSpeed streaming interfaces available per bridge | | Streaming Interfaces | Streaming bridge has four bidirectional interfaces named a, b, c and d | | Link/Port | Physical channel between two routers or between a router and a bridge | | Channel | Physical or virtual channel between two routers or between a router and a bridge | | Virtual Channel (VC) | Virtual channel between two routers or between a router and a bridge | | Injection channel | Incoming (with respect to NoC) virtual or physical channel at a router at which a hostport is connected to via a bridge and at which the router receives traffic from the hostport | | Ejection channel | Outgoing (with respect to NoC) virtual or physical channel at a router at which a hostport is connected to via a bridge and at which the router sends traffic to the hostport | | Cell | A node in a 2D grid or mesh; A NoC router is associated with every cell | | Node | A router or cell of the 2D NoC grid | | Virtual node | A boundary router or cell of the 2D NoC grid; in nxm mesh, there are n virtual nodes each at the top boundary and the bottom boundary in the mesh, and m at the right and the left boundary of the mesh | | Virtual router | A router at a virtual node; virtual router is connected to an adjacent internal (non-boundary) router; there is no real hardware associated with a virtual router; the hostport bridge connected to a virtual router is directly connected to the internal router’s port at which the virtual router is connected to the internal router | | Multi-layer NoC | Multiple parallel mesh NoCs each forming a layer; routers in each layer operate independently of each other; two NoC layers have no connection between their routers; a bridge at a cell connects the injection/ejection port to a router in each layer and transmits each hostport transaction message to one of the layer’s routers, and receives transaction messages from all layer’s routers delivering them to the hostport | | NoC layer | An independent NoC layer in a multi-layer NoC | | Packetization | Encoding of hostport signals into NoC packet format before they are delivered to the NoC; bridges perform packetization of hostport transaction messages into NoC packets and de-packetization of NoC packets into hostport transaction messages | | Beat | A single cycle of data part of a transaction message at an AXI-4 or streaming hostport interface | | Flit | Part of a packet that is transmitted or received at a router’s port in a single cycle | | Packet | A transaction message packetized into NoC message; a packet may contain one or many flits | | Transaction | A sequence of inter-dependent messages between various source and destination hosts/ports/interfaces. | | Hostport id | The globally unique numerical id of a hostport bridge; this is assigned by NocStudio or can be assigned by user during hostport addition; injecting transaction messages must have an associated destination in form of hostport id | | hostport name | Name of the hostport in form of hostname/portname | | interface id | This is a, b, c or d if used in context of name in streaming bridge or ld, st, st\_resp and ld\_data in context of name of AXI bridge; In streaming, they corresponds to 0, 1, 2 or 3 values respectively; in streaming bridge, injecting transaction messages must have an associated destination in form of interface id | | QoS id | 4-bit QoS id; each transaction has a single QoS id | | priority | 2-bit priority of a transaction; there is a 1-1 mapping between QoS id and priority which can be set in NocStudio | | weight | weight of a QoS id; this 8-bit value | |

# Overview

NoC Elements like Streaming Bridge and Router support configuration and status registers. These NoC registers can be accessed using AXI4-lite interface. The internal bus that enables this access is called Register Bus. Register bus can be enabled or disabled for a particular configuration.

NoC implements the Register bus using one NoC layer. On this layer at a given time there can only be one outstanding Read or Write transaction.

# Description

Figure below shows simplified schematic block diagram of Register Bus.

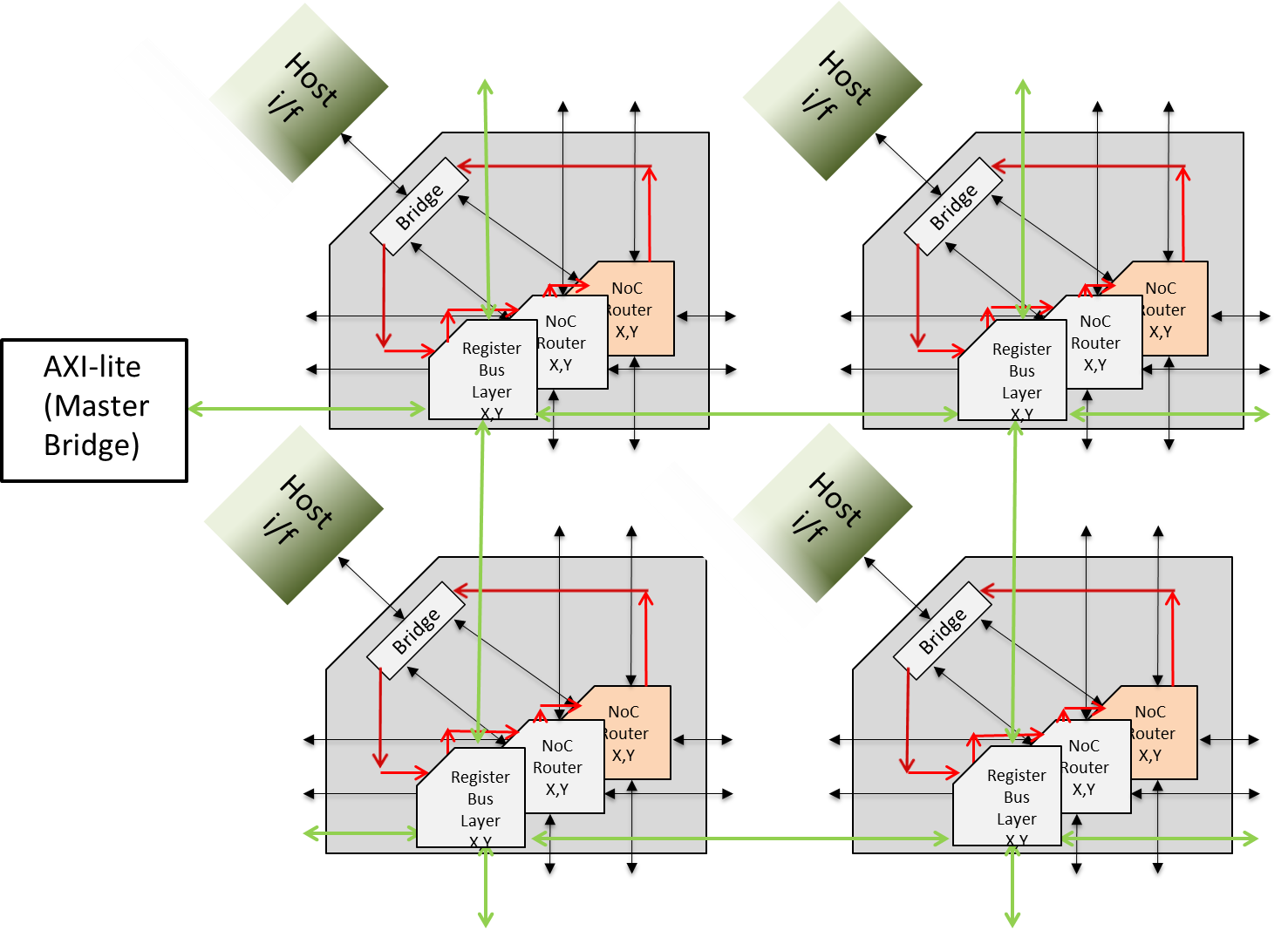


Figure : Register Bus block diagram

Register bus has 3 major components

* Register Bus Master Bridge
* Register Bus Slave Bridge
* Register Bus Slave.

Register Bus master bridge converts the AXI4-lite transaction into internal Router (Layer) protocol transaction. The Register Bus master bridge can be placed on any Node, and this Node can be defined using NocStudio’s configuration file. According to the Register address the transaction is routed to the appropriate Node. XY routing is shown in the block diagram using Green multidirectional arrows.

Register Bus Slave Bridge is instantiated at each node. This block converts Layer transaction to a Ring protocol transaction. The Red unidirectional bus is the Ring bus which is the register access bus for all routers and bridges in that particular node.

Register Bus Slave is instantiated in each NoC element like a Streaming Bridge and a Router. This block converts a Ring bus transaction into a register Read/Write access of the NoC element.

# Register Bus Interfaces

Following are the interface signals of Register Bus

|  |  |  |  |
| --- | --- | --- | --- |
| **Signal name** | **Width** | **Input/**  **Output** | **Description** |
| clk | 1 | Input | Clock |
| reset | 1 | Input | Reset |
| **AXI-Lite Interface** |  |  |  |
| axi\_clk | 1 | Input | Clock |
| axi\_reset | 1 | Input | Reset |
| axi\_awaddr | 32 | Input | Write address channel - Address |
| axi\_awvalid | 1 | Input | Write address channel - Valid |
| amb\_awready | 1 | Output | Write address channel - Ready |
| axi\_wdata | 32 | Input | Write data channel - Data |
| axi\_wstrb | 4 | Input | Write data channel – Data Strobe |
| axi\_wvalid | 1 | Input | Write data channel – Data Valid |
| amb\_wready | 1 | Output | Write data channel – Data Ready |
| axi\_bready | 1 | Input | Write Response channel – Ready |
| amb\_bresp | 2 | Output | Write Response channel – Response |
| amb\_bvalid | 1 | Output | Write Response channel – Valid |
| axi\_araddr | 32 | Input | Read address channel - Address |
| axi\_arprot | 3 | Input | Read address channel - Protection |
| axi\_arvalid | 1 | Input | Read address channel - Valid |
| amb\_arready | 1 | Output | Read address channel - Ready |
| axi\_rready | 1 | Input | Read data channel - Data |
| amb\_rdata | 32 | Output | Read Response channel – Ready |
| amb\_rresp | 2 | Output | Read Response channel – Response |
| amb\_rvalid | 1 | Output | Read Response channel – Valid |
| **Router Host port to RBUS bridge Signals** |  |  |  |
| rt\_rbusbrdg\_flit\_sop | 1 | Input | Start of Packet |
| rt\_rbusbrdg\_flit\_eop | 1 | Input | End of Packet |
| rt\_rbusbrdg\_flit\_bv | ?? | Input | Byte Valid in the flit with eop |
| rt\_rbusbrdg\_flit\_valid | 1 | Input | Flit valid (1 bit since we use 1 VC |
| rt\_rbusbrdg\_flit\_data | 32 | Input | Flit data |
| rt\_rbusbrdg\_flit\_sb | 18 | Input | Side band signals |
| rt\_rbusbrdg\_flit\_outp | ?? | Input | Router out port (NOT USED) |
| rbusbrdg\_rt\_credit\_inc | 3 | Output | Credit info |
| **RBUS bridge to Router Host port Signals** |  |  |  |
| rt\_strtxbrdg\_credit\_inc | 3 | Input | Credit info |
| strtxbrdg\_rt\_flit\_sop | 1 | Output | Start of Packet |
| strtxbrdg\_rt\_flit\_eop | 1 | Output | End of Packet |
| strtxbrdg\_rt\_flit\_bv | ?? | Output | Byte Valid in the flit with eop |
| strtxbrdg\_rt\_flit\_valid | 1 | Output | Flit valid |
| strtxbrdg\_rt\_flit\_data | 32 | Output | Flit data |
| strtxbrdg\_rt\_flit\_type | ?? | Output | Flit type 0: Normal; 1:barrier |
| strtxbrdg\_rt\_flit\_sb | 18 | Output | Side band signals |
| strtxbrdg\_rt\_flit\_outp | 3 | Output | Router out port |
| **Router to/from Streaming RX bridge Signals** |  |  |  |
| cmd\_in | 3 | Input | Command In |
| data\_in | 32 | Input | Data In |
| cmd\_out | 3 | Output | Command Out |
| data\_out | 32 | Output | Data Out |

Table : Interface Signals

# Register List and Summary

NoC IP includes a number of registers, which are divided into the following three categories.

* **Configuration registers – must be configured for the correct operation of NoC.**
* **Debug and performance monitor registers – provide debug-ability, observe-ability of the correct operation and basic performance stat.**
* **Error log registers – log basic protocol errors for post-silicon debug.**

Registers can be R (read only), or RW (can be read from or written to), or RC (can be read or cleared: writing 1 to a register bit will have no effect on it, while writing 0 to a register bit will clear it). The following table summarizes all registers, and subsequent sections describe them in greater detail.

Register bits are valid, or reserved, or unavailable. If reserved, the bits are present, can be written into or read from, but are not used. If unavailable, register bits are not present. Reset values of valid register bits are marked with 0 or 1, reserved register bits are marked “r”, and unavailable register bits are marked “u”.

|  |  |  |
| --- | --- | --- |
| **Register Group Name** | **Where** | **Description** |
| **Configuration** | | |
| axi4\_slave\_bridge\_base\_addr  acronym: B  Type: RW  Size: 2x32  Count: 256  Reset value: | per AXI4 master bridge | Describes the base address of the AXI4 slave bridges. A master bridge includes one such register for every slave host port with which it communicates with. |
| qos\_profile\_data  acronym: P  Type: RW  Size: 4x32  Count: 1  Reset value: 0x07070707 | per transmitting bridge | Describes the weight value of each QoS profile supported at the bridge.  Each byte of this register must be greater than or equal to 3. |
| **Debug and Performance Monitor** | | |
| noc\_injection\_flit\_count  Acronym: I  Type: RW  Size: 4x32  Count: 1  Reset value: 0x00000000 | per transmitting bridge | Can count # of injected flits into NoC from every bridge. The counting can be filtered with a mask (mask bits are SOP, EOP, route info, NoC id, VC id).  Saturating counter (does not rollover). |
| bridge\_transmit\_beat\_count  Acronym: T  Type: RW  Size: 4x32  Count: 1  Reset value: 0x00000000 | per transmitting bridge | Can count # of transaction beats to bridge from bridge at every hostport. The counting can be filtered with a mask. Supported mask bits are:  1. SOP, EOP, QoS, src interface id, dest hostport id, dest interface id for stream  2. address[43:12] for axi4m  3. AID for axi4s  Saturating counter (does not rollover). |
| noc\_ejection\_flit\_count  Acronym: E  Type: RW  Size: 4x32  Count: 1  Reset value: 0x00000000 | per receiving bridge | Can count # of ejected flits from NoC to every bridge. The counting can be filtered with a mask (mask bits are SOP, EOP, route info, NoC id, VC id).  Saturating counter (does not rollover). |
| bridge\_receive\_beat\_count  Acronym: R  Type: RW  Size: 4x32  Count: 1  Reset value: 0x00000000 | per receiving bridge | Can count # of transaction beats from bridge to bridge at every hostport. The counting can be filtered with a mask.  Supported mask bits are:  1. SOP, EOP, interface id for stream  2. address[43:12] for axi4s  3. AID for axi4m  Saturating counter (does not rollover). |
| bridge\_transmit\_fifo\_status  Acronym: BTS  Type: R  Size: 1x32  Count: 1  Reset value: 0x00000000 | per transmitting bridge | Indicates the status of one of the per-interface FIFOs at the transmitting bridge to NoC. |
| bridge\_receive\_fifo\_status  Acronym: BRS  Type: R  Size: 1x32  Count: 8  Reset value: 0x00000000 | per receiving bridge | Indicates the status of one of the per-NoC and per-VC FIFOs at the receiving bridge from NoC. |
| router\_input\_vc\_status  Acronym: RIVCS  Type: R  Size: 1x32  Count: 8  Reset value: 0x00000000 | per router port | Indicates the status of an input port of a router. |
| router\_output\_vc\_status  Acronym: ROVCS  Type: R  Size: 1x32  Count: 8  Reset value: 0x01010101 | per router port | Indicates the status of one of the output ports of a router. |
| **Error log registers (protocol checkers include assertions for these)** | | |
| bridge\_transmit\_fifo\_overflow  Acronym: BTOE  Type: RC  Size: 1x32  Count: 1  Reset value: 0x00000000 | per transmitting bridge | Indicates that one of the per-interface FIFOs at the transmitting bridge to NoC is overflown. |
| bridge\_receive\_fifo\_overflow  Acronym: BROE  Type: RC  Size: 1x32  Count: 1  Reset value: 0x00000000 | per receiving bridge | Indicates that one of the per-NoC and per-VC FIFOs at the receiving bridge from NoC is overflown. |
| transaction\_valid\_without\_sop (0)  transaction\_sop\_after\_sop (1)  transaction\_illegal\_dest\_qos (2)  Acronym: XE  Type: RC  Size: 1x32  Count: 1  Reset value: 0x00000000 | per bridge transmitting interface | Bit 0: For streaming bridge interface, sets if a transaction is initiated w/o SOP.  Bit 1: For streaming bridge interface, sets if a SOP is received after SOP.  Bit 2: Sets if a transaction is received from bridge for which there is no entry present in the vcmap, i.e. the destination and/or QoS is not supported  In AXI, this gets set if the provided address and/or QoS are illegal. |
| router\_error  Acronym: RERRS  Type: RC  Size: 1x32  Count: 1  Reset value: 0x00000000 | per router | Indicates that one of the FIFOs at the router has overflown or has credit underrun. |

Table : NoC Register List and Summary

## Configuration Registers

### axi4\_slave\_bridge\_base\_addr (B)

The device address range of an AXI4 slave device must be power of two and in multiple of 4KB. Additionally, if the address range of a slave device is x bytes, then the base address (starting address) of the slave must be multiple of x. The address range of all slave devices is set in NocStudio using parameters, thus the address range is not re-programmable, it gets set at the NoC design time. The base address is programmable via these registers. These registers are kept at every AXI4 master bridge to store the address of AXI4 slave devices. At a master bridge, one register is present for every slave device the master communicates with.

*ADDRS\_WIDTH:* Up to 64

*BLOCK\_SIZE\_WIDTH:* Number of lower order addresses bits designating the granularity of address blocks used to specify the address range; default is 4KB (i.e. 12 bits)

*SLAVE\_ADDR\_RANGE:* The address range of slave in units of 2^^*BLOCK\_SIZE\_WIDTH* bytes.

The register is composed of two 32-bit registers B0 and B1, however based on the address parameters set in NocStudio only certain bits of these registers are available for user programming. The format of this register is shown below for a slave device with address range 512, block size of 4KB (*BLOCK\_SIZE\_WIDTH* = 12 bits) and 64-bit address. The fields marked reserved are not available.



In each AXI4 master bridge, there is one such register for every slave it communicates with, thereby up to 256 such registers.

Note: AXI registers are preliminary specification

### qos\_profile\_data (P)

Each transmitting bridge supports up to 16 QoS profiles. Each QoS profile is composed of pri and weight, however only the weight is programmable, therefore is part of the registers. QoS profile data is composed of four registers, P0, P1, P2 and P3, 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. The registers in a hostport with 16 QoS profiles is shown below.



## Debug and Performance Monitor

### noc\_injection\_flit\_count (I), noc\_ejection\_flit\_count (E), bridge\_transmit\_beat\_count (T), and bridge\_receive\_beat\_count (R)

Each of these registers is composed of four 32-bit registers: 3, 2, 1, and 0. The first two registers store the 48-bit counter, while the next two registers store the value of various fields against which the fields of transaction/packet are matched, and the mask. If a mask bit is set, then the corresponding bit in the field is ignored during match, otherwise it is matched. For example, if the mask bits of QoS are all 1, then all flits/beats with of all QoS values will be counted; if it is “1100” then the QoS value of a beat is compared with the two LSB bits of the QoS field value and counter is incremented upon a match.

The first three registers 3, 2, and 1 are shown below.



The last 0th register interpretation depends upon the register, and upon which values the counter is counting. For noc\_injection\_flit\_count, noc\_ejection\_flit\_count registers, the 0th register has following format.



For the bridge side transaction counters, the 0th register format for the streaming bridge is shown below. The format is different for the receive and transmit side, as shown below.



The 0th register format for the AXI bridge is different for master and slave and for transmit and receive transaction. The four resulting formats are shown below.



### bridge\_transmit\_fifo\_status (BTS) and bridge\_receive\_fifo\_status (BRS)

The format of the register is shown below.



### router\_input\_vc\_status (RIVCS) and router\_output\_vc\_status (ROVCS)

There are 8 RIVCS and 8 ROVCS per router, one for each router’s port (note that only 5 are active registers other 3 are reserved). The format of the register is shown below.



## Error Log Registers

### bridge\_transmit\_fifo\_overflow (BTOE) and bridge\_receive\_fifo\_overflow (BROE)

FIFO overflow flag stores a FIFO overflow event and all such flags are read write registers. FIFO overflow flag is 1-bit wide, and a single port may have up to 8 VCs or interfaces, therefore a single 8-bit register can store all FIFO overflow flags of a single port. The format of the register is shown below.



### Router\_error (RERRS)

FIFO overflow flag stores a FIFO overflow event; credit overflow and credit underflow flags capture these events for the credit interface per VC and per port of the router. All such flags are read write (RW) registers. There is one error register per router port.



### Transaction errors (XE[0], XE[1], XE[2]): transaction\_valid\_without\_sop, transaction\_sop\_after\_sop, transaction\_illegal\_dest\_qos

This register logs error events. There is a single register per node id. The format of the registers is shown below.



## Address Map of Registers

Entire NoC has a total 16MB of register address space. The base address of NoC register address space must be aligned to 16MB blocks.

NoC configuration bus only operates on the 24 LSB bits of the provided address. A NoC system may have up to 256 addressable nodes. Each node has a unique numerical identifier assigned by NocStudio. 8-bits of the register address (16 through 23) are directly used as the node id.

A node has 64KB register address space. Within each node there may be up to 15 modules – 9 routers for each layer and 7 bridges. The routers have first 32KB reserved of which 9 1 KB regions are used. Next 32KB is for bridges of which 7 4KB regions are used.

The address mapping, the resulting decoding scheme, and address space available to each module is shown in the figure below.



Figure : Address decoding and register address space for various NoC nodes and modules within the nodes

Within each router and the bridge modules the registers are assigned addresses to further simplify the decoding of the address within the module.

RIVCS

|  |  |
| --- | --- |
| Fields | Description |
| Head flit valid | 1’b0 : indicates that the VC buffer is empty  1’b1 : Indicates that the VC buffer has a flit ready |
| Buffer full | 1’b1 : Indicates that the VC buffer is full  1’b0 : Indicates that the VC buffer has space to receive flits from the link |
| Head flit barrier state | 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 |
| Head flit sop | 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 |
| VC upsizer occupied | 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 |
| Head flit output port | Value indicates the router output port to which the packet at the head of the VC is destined to  3’d0 : N, 3’d1 : E, 3’d2 : W, 3’d3 : S, 3’d4 : H |

ROVCS

|  |  |
| --- | --- |
| Fields | Description |
| VC barrier state | 1’b1 : Indicates that the output VC is currently in ‘QoS Barrier’ state and will only allow flits of ‘barrier’ type from the corresponding VCs on the input ports  1’b1 : Indicates that the output VC is currently in ‘QoS Normal’ state and only allow flits of ‘normal’ type from the corresponding VCs on the input ports |
| VC occupied | 1’b1 : Indicates that this output VC is currently locked to the corresponding VC on one of the input ports.  1’b0 : Indicates that this output VC is free and can be acquired by the corresponding VC on one of the input port for the transmission of a packet |
| VC empty credit | 1’b1 : Indicates that this output VC has no credit for transmission of flits to the downstream link  1’b0 : Indicates that credit are available for transmission to downstream link |
| VC full credit | 1’b1 : Indicates that the credit level with this VC is at the maximum provisioned value |

RE

|  |  |
| --- | --- |
| Fields | Description |
| OVF | 1’b1 on this status bit indicates that the router event counter has overflowed (32’hFFFFFFFF -> 32’dh0). This is a sticky status bit which must be written with 1’b0 to clear. |

REM

|  |  |
| --- | --- |
| Fields | Description |
| IM | 1’b1 : Masks or disables an interrupt from being generated by the event count overflow status bit (RE)  1’b0 : Enables an interrupt to be generated when event counter status bit is set. |

RECC

|  |  |
| --- | --- |
| Fields | Description |
| EVT | Selects an event for the counter  2’b00 : Counter disabled (default)  2’b01 : Generates count event on every EOP received for the selected input port and selected input VCs. This can be used to count packets received on a router input port  2’b10 : Generates count event on every flit received for the selected input port and selected input VCs. This can be used to count total flits received on a router input port  2’b11 : Generates count event when VC buffer is full for any selected VC on the selected input port |
| INP | Specified the input router port on which the events are captured  3’d0 : N, 3’d1 : E, 3’d2 : W, 3’d3 : S, 3’d4 : H |
| IVC | 4’bxxxx : any bit position set to 1’b1 selects that VC for event capture. Bit position at 1’b0 means that VC is not selected for event capture. |

REC

|  |  |
| --- | --- |
| Fields | Description |
| Event Counter | 32’bit event incrementing counter. Rollover from 32’hFFFFFFFF -> 32’d0 sets the rollover status bit RE |

RID

|  |  |
| --- | --- |
| Fields | Description |
| Layer | 4-bit identifier of the NoC layer on which this router is located |
| Position | 8-bit position ID of this router in the NoC |

Regbus master bridge

E

|  |  |
| --- | --- |
| Fields | Description |
| E0 | 1’b1 indicates that a read address AR decode error was detected on the regbus master bridge. This is a sticky status bit which must be cleared by writing 1’b0 |
| E1 | 1’b1 indicates that a read address decode error was received from a slave |
| E2 | 1’b1 indicates that a write address AW decode error was detected on the regbus master bridge. This is a sticky status bit which must be cleared by writing 1’b0 |
| E3 | 1’b1 indicates that a write address decode error was received from a slave |

EIM

|  |  |
| --- | --- |
| Fields | Description |
| M0 | 1’b1 : Masks or disables an interrupt from being generated by error status bit E0  1’b0 : Enables an interrupt to be generated when errors status bit E0 is set |
| M1 | 1’b1 : Masks or disables an interrupt from being generated by error status bit E1  1’b0 : Enables an interrupt to be generated when errors status bit E1 is set |
| M2 | 1’b1 : Masks or disables an interrupt from being generated by error status bit E2  1’b0 : Enables an interrupt to be generated when errors status bit E2 is set |
| M3 | 1’b1 : Masks or disables an interrupt from being generated by error status bit E3  1’b0 : Enables an interrupt to be generated when errors status bit E3 is set |

ERA

|  |  |
| --- | --- |
| Fields | Description |
| Read decode error address | This is the address on AR channel for which a decode error was detected on the regbus master. This corresponds to the status register bit E0 |

EWA

|  |  |
| --- | --- |
| Fields | Description |
| Write decode error address | This is the address on AW channel for which a decode error was detected on the regbus master. This corresponds to the status register bit E2 |

RBMID

|  |  |
| --- | --- |
| Fields | Description |
| Unique bridge ID | A unique 8-bit identifier assigned to the regbus master to uniquely identify it on the NoC |

## Ring bus Protocol

### Ring bus Commands

Shown below are ring bus commands

|  |  |
| --- | --- |
| cmd\_\* <2:0> | Command Description |
| 000 | Nop |
| 001 | Read |
| 010 | Write |
| 011 | WrD (Write Data) |
| 100 | Ack |
| 101 | AckClaim (To be implemented) |
| 110 | SlaveReq (To be implemented) |
| 111 | SlaveAck (To be implemented) |

Table : Ring bus commands

### Ring bus Protocol

Shown below is the Ring bus protocol

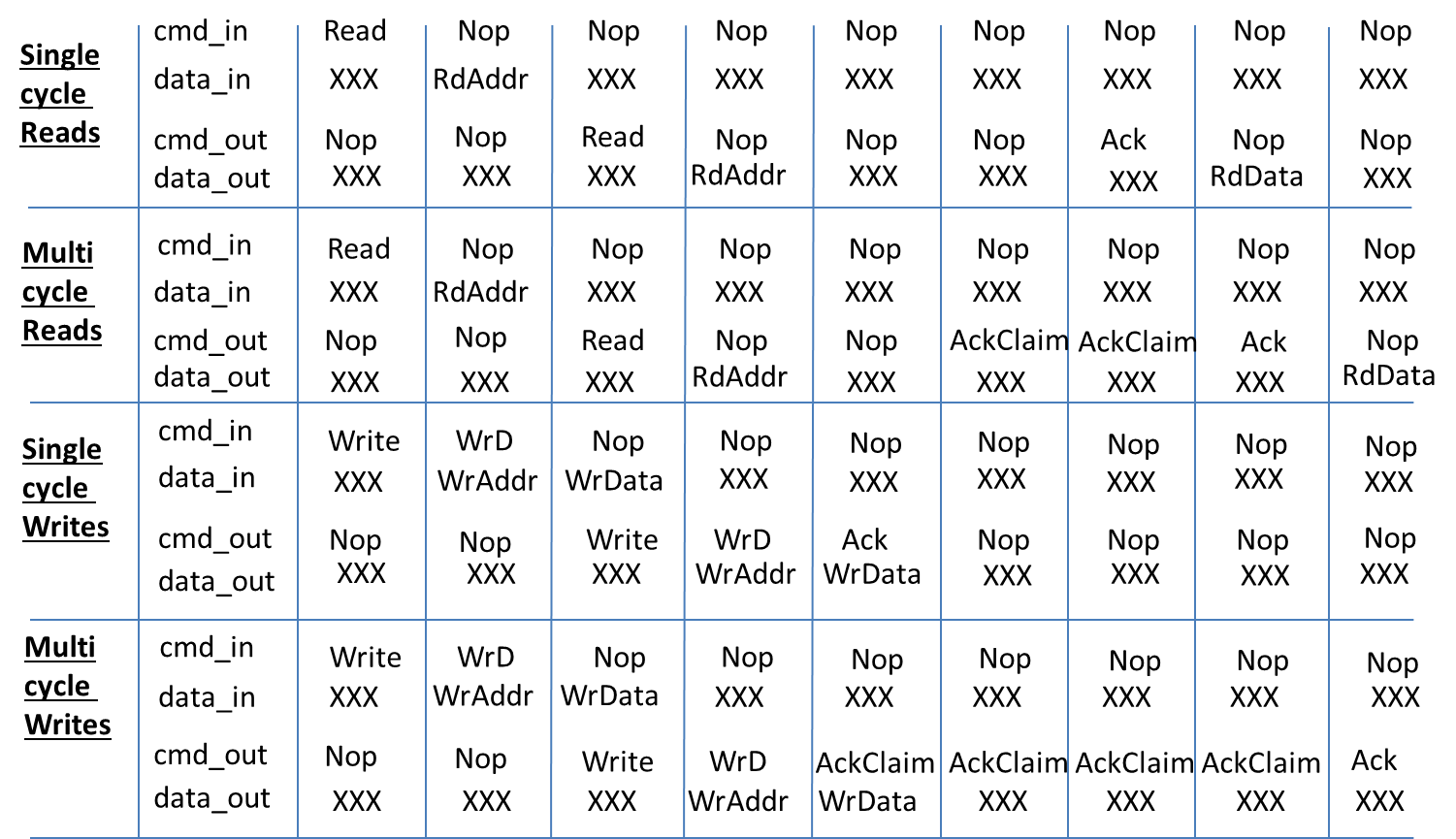


Figure : Ring bus Protocol

Document Changes/Revisions

*Documentation Changes* include additions, deletions, and modifications made to this document. This section identifies the changes made in each release of the document.

Document Revision A

|  |
| --- |
| 2670 Seely Ave  Building 11  San Jose, CA 95134  (408) 914-6962  [www.netspeedsystems.com](http://www.exar.com) |