NetSpeed SYSTEMS

NetSpeed Register Bus Architecture Specification

Customer Document

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

Contents

List of Figures

List of Tables

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

1 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.

2 Description

Figure below shows simplified schematic block diagram of Register Bus.

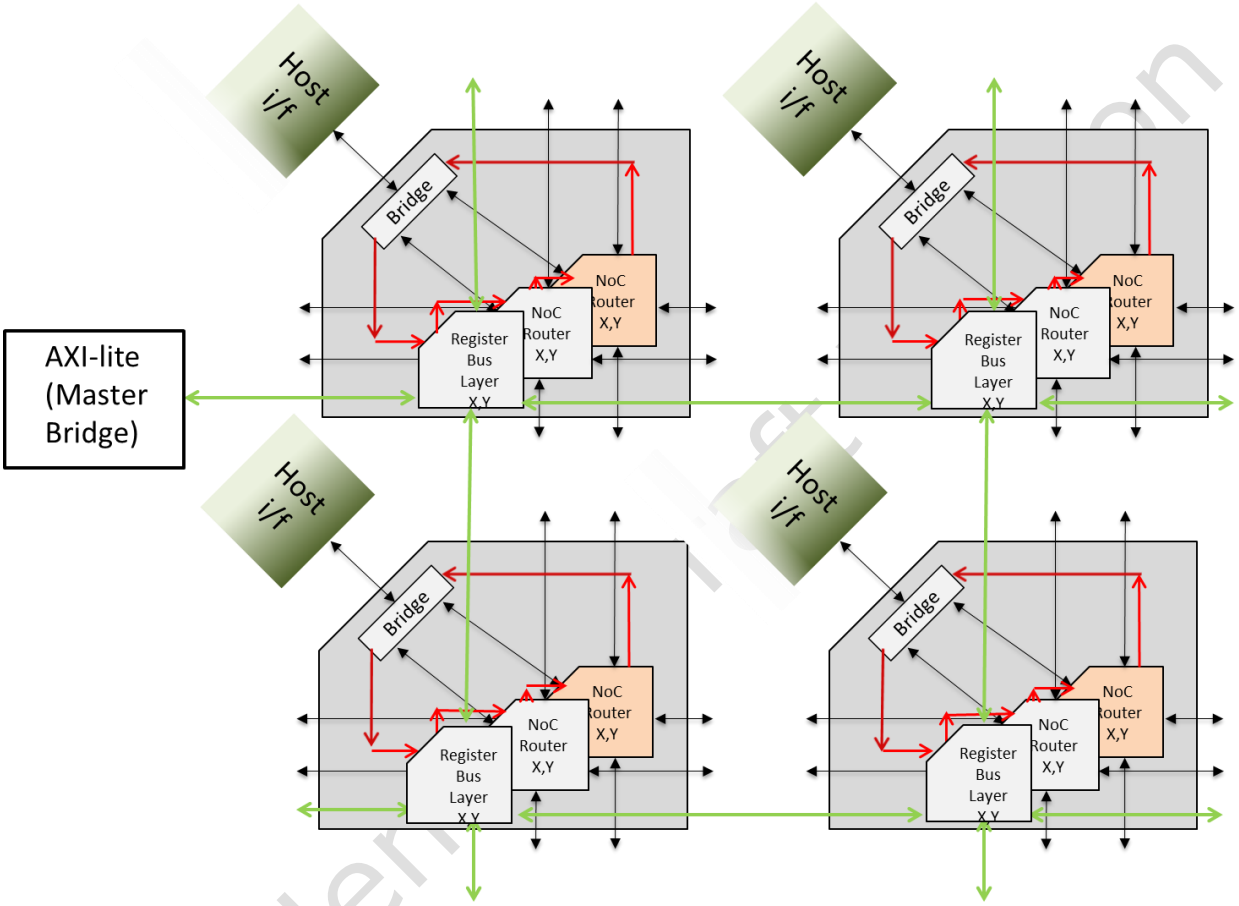

Figure 1: Register Bus block diagram

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.

3 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 1: Interface Signals

4 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	per transmitting bridge	Describes the weight value of each QoS profile supported at the bridge. Each byte of this register

Count: 1 Reset value: 0x07070707		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	per receiving bridge	Can count # of transaction beats from bridge to bridge at every hostport. The counting can be filtered with a mask.

Reset value: 0x00000000		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	per transmitting bridge	Indicates that one of the per-interface FIFOs at the transmitting bridge to NoC is overflown.

Count: 1 Reset value: 0x00000000		
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 2: NoC Register List and Summary

4.1 Configuration Registers

4.1.1 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.

B		256	axi4_slave_hostport_base_addr																													
	1		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
			Unused																				ADDRESS_BASE										
	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
			ADDRESS_BASE											Unused																				

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

4.1.2 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.

P		1	qos_profile_data																											
	3		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
			weight of QoS profile 15								weight of QoS profile 14								weight of QoS profile 13								weight of QoS profile 12							
	2		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
			weight of QoS profile 11								weight of QoS profile 10								weight of QoS profile 9								weight of QoS profile 8							
	1		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
			weight of QoS profile 7								weight of QoS profile 6								weight of QoS profile 5								weight of QoS profile 4							
	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
			weight of QoS profile 3								weight of QoS profile 2								weight of QoS profile 1								weight of QoS profile 0							

4.2 Debug and Performance Monitor

4.2.1 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.

I, E, T, R	3		counter value (MSB)																															
			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
			Unused																counter[47:32]															
I, E, T, R	2		counter value (LSB)																															
			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
			counter[31:0]																															
I, E, T, R	1		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																															

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.

I,E	0		All protocol bridges NoC side packet counting field values																												
			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
			Reserved											Route info										Reserved				NoC id			VC	EOP	SOP

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.

T	0		Stream hostport transmit side transaction counting field values																												
			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
			Reserved								dest port id								res	QoS				res	src if id				res		dest if id		EOP	SOP
																									src if id				this is one hot					
R	0		Stream hostport receive side transaction counting field values																												
			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
			Reserved																										if id				EOP	SOP

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.

T	0		AXI master hostport transmit side and slave hostport receive side transaction counting field values																												
			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
			Unused								address[min(ADDRS_WIDTH-1, 43) : 12]																							
T	0		AXI slave hostport transmit transaction counting field values																												
			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
			Unused																master hostport id								aid[AID_WIDTH-1:0]							
R	0		AXI master hostport receive side transaction counting field values																												
			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
			Unused																								aid[AID_WIDTH-1:0]							

4.2.2 bridge_transmit_fifo_status (BTS) and bridge_receive_fifo_status (BRS)

The format of the register is shown below.

BTS	1	State of input FIFOs of Tx streaming																													
		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
		Unused																int d				int c				int b				int a			
																														un	F	S	V
														1-bit	F			Buffer Full															
														1-bit	S			Head flit sop															
														1-bit	V			Head flit valid (buffer ready)															

BRS	8	State of input FIFOs of Rx streaming (8 such registers)																													
		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
		vc3, NoC layer i								vc2, NoC layer i								vc1, NoC layer i								vc0, NoC layer i							
																										un	F	B	S	V	Outi		
														1-bit	F			Buffer Full															
														1-bit	B			Head flit Barrier state															
														1-bit	S			Head flit sop															
														1-bit	V			Head flit valid (buffer ready)															
														2-bits	Outi			Head flit Output interface															

4.2.3 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.

RIVCS	8	State of input port p (8 such registers, one for each port)																													
		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
		ivc3 state								ivc2 state								ivc1 state								ivc0 state							
														1-bit	V			Head flit valid (buffer rea								V	F	B	S	un	outp		
														1-bit	F			Buffer Full															
														1-bit	B			Head flit Barrier state															
														1-bit	S			Head flit sop															
														3-bits	Outp			Head flit Output port															

ROVCS	8	State of output port p (8 such registers, one for each port)																													
		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
		ovc3 state								ovc2 state								ovc1 state								ovc0 state							
														1-bit	B			VC barrier state								un					B	VB	CE	CF
														1-bit	VB			VC occupied															
														1-bit	CE			VC empty credit (cntr == 0)															
														1-bit	CF			VC full credit (cntr == Max)															

4.3 Error Log Registers

4.3.1 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.

BTOE		1	Buffer overflow state of input FIFOs of Tx streaming																												
			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
			Unused																										d	c	b	a
BROE		1	Buffer overflow state of input VC FIFOs from NoC Routers of Rx streaming																												
			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
			NoC 7				NoC 6				NoC 5				NoC 4				NoC 3				NoC 2				NoC 1				NoC 0			
			If a NoC layer is not connected to the bridge, then the corresponding bits are unused																												
			For the 4 VCs																												3	2	1	0

4.3.2 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.

RERRS		1	VC Error register																												
			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
			ovc3 Err				ivc3 Err				ovc2 Err				ivc2 Err				ovc1 Err				ivc1 Err				ov0 Err				ivc0 Err			
																											un		CU	CO	un			O
			1-bit								O				Input VC buffer overflow occurred																			
			1-bit								CO				Output VC credit overflow occurred																			
			1-bit								CU				Output VC credit underflow occurred																			

4.3.3 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.

XE	1	transaction_valid_without_sop, transaction_sop_after_sop, transaction_illegal_dest_qos																												
		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
		Unused																													c	b	a
																															a = transaction_valid_without_sop		
																															b = transaction_sop_after_sop		
																															c = transaction_illegal_dest_qos		

4.4 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.

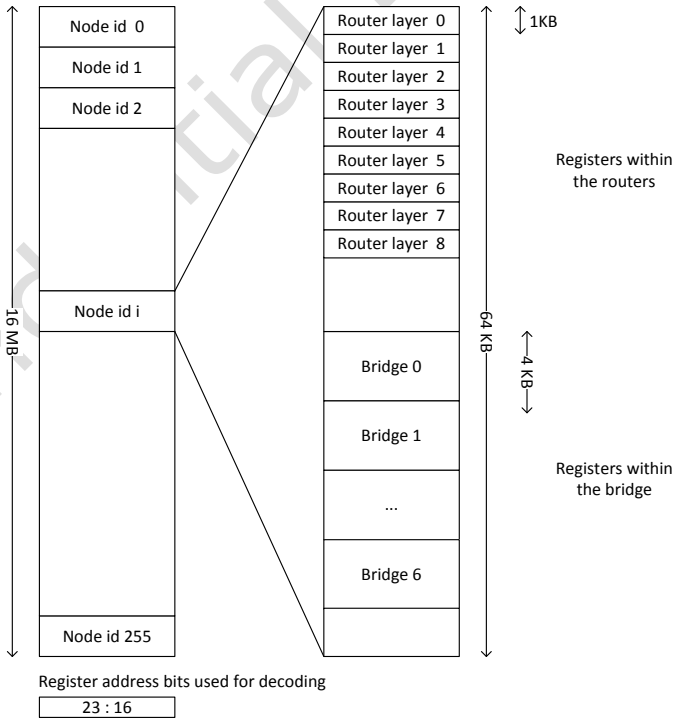

Figure 2: Address decoding and register address space for various NoC nodes and modules within the nodes

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

4.5 Ring bus Protocol

4.5.1Ring 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 3: Ring bus commands

4.5.2Ring bus Protocol

Shown below is the Ring bus protocol

Single cycle Reads	cmd_in	Read	Nop	Nop	Nop	Nop	Nop	Nop	Nop	Nop
	data_in	XXX	RdAddr	XXX	XXX	XXX	XXX	XXX	XXX	XXX
	cmd_out	Nop	Nop	Read	Nop	Nop	Nop	Ack	Nop	Nop
	data_out	XXX	XXX	XXX	RdAddr	XXX	XXX	XXX	RdData	XXX
Multi cycle Reads	cmd_in	Read	Nop	Nop	Nop	Nop	Nop	Nop	Nop	Nop
	data_in	XXX	RdAddr	XXX	XXX	XXX	XXX	XXX	XXX	XXX
	cmd_out	Nop	Nop	Read	Nop	Nop	AckClaim	AckClaim	Ack	Nop
	data_out	XXX	XXX	XXX	RdAddr	XXX	XXX	XXX	XXX	RdData
Single cycle Writes	cmd_in	Write	WrD	Nop	Nop	Nop	Nop	Nop	Nop	Nop
	data_in	XXX	WrAddr	WrData	XXX	XXX	XXX	XXX	XXX	XXX
	cmd_out	Nop	Nop	Write	WrD	Ack	Nop	Nop	Nop	Nop
	data_out	XXX	XXX	XXX	WrAddr	WrData	XXX	XXX	XXX	XXX
Multi cycle Writes	cmd_in	Write	WrD	Nop	Nop	Nop	Nop	Nop	Nop	Nop
	data_in	XXX	WrAddr	WrData	XXX	XXX	XXX	XXX	XXX	XXX
	cmd_out	Nop	Nop	Write	WrD	AckClaim	AckClaim	AckClaim	AckClaim	Ack
	data_out	XXX	XXX	XXX	WrAddr	WrData	XXX	XXX	XXX	XXX

Figure 3: 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