# Appendix A: NetSpeed Streaming Interface Protocol

This section describes NetSpeed Streaming Interface Protocol (NSIP) and streaming bridge that connects a streaming host port device to the NoC via streaming bridge. NSIP provides a flexible and simple interface to the SoC IPs by allowing up to four bi-directional interfaces at each host port. Each interface uses separate data bus and credit-based flow control signals and can receive and transmit messages simultaneously. Each message may be composed of multiple data beats. In each cycle, a single data beat is transmitted or received at an interface, containing a fixed number of data bits equal to the interface data width. The data width of the transmitting and receiving interfaces are independently configurable and must be power of two multiples of their cell size.

Cell size can be a system-wide constant indicating a common denominator for data sizes. Upsizing and downsizing of data happens on power-of-2 multiples of the cell size. For a system with more than one cell size, where interface data width is specified in terms of one of the cell sizes, and communication only happens between interfaces that share a cell size, the cell size property can be set to -1 to indicate that NocStudio should allow multiple cell sizes. Each physical network (a.k.a. layer) can only support one cell size.

Outgoing message data from each host port’s TX interface are packetized into NoC packets for injection into the NoC and ejected packets from the NoC are de-packetized into message data beats delivered to the destination RX interface. A single NSIP host port may have both transmitting and receiving interfaces, in the following sections we describe the transmitter and receiver interface behaviors.

## NSIP Transmitter

A host port with transmitting NSIP interfaces can use up to four independently flow controlled interfaces, named a, b, c, and d. Each TX interface has its own specified data width which is a power of 2 multiple of the system wide specified cell size. Each TX interface may transmit messages that are composed of one or more data beats – each data beat is transmitted in a single cycle. Data beats are always equal to the data width of the TX interface and are marked with SOP and EOP bits indicating the start of a message and end of the message, respectively. Multiple messages may not interleave on a TX interface, however there may be any number of idle cycles on a TX interface between successive data beats of same of different messages. Furthermore, any combination of the RX interfaces may receive data beats simultaneously.

NSIP protocol supports a 4-bit QoS that can be transmitted along with each transmitted message. The QoS indicates the Quality-of-Service level of the message and must stay the same for every data beat of the message. In CFG IP, each QoS levels are isolated from each other and has a 2-bit associated priority. Furthermore, each QoS level may have an associated weight at every host port. In CFG NoC IP, the weights are programmable.

A transmitting host port’s interface is allowed to transmit to any other host port’s interface. Therefore, each transmitted message must also include an 8-bit destination id, indicating a system wide identifier for each destination host port in the system, and 2-bits indicating the destination interface. These 10-bits must stay the same for all data beats of a message. Host port identifiers are system wide unique values (it can be user assigned for each host port in NocStudio environment; if unassigned for a host port then NocStudio automatically assigns unique identifiers).

For flow control, each TX interface uses a separate flow control. The flow control is credit based and occurs at each interface between the transmitting host port and the streaming bridge. For each interface, streaming bridge contains a flow control FIFO for storing the transmitted data beats at the TX interface. The TX host port is expected to maintain a credit counter per interface and is allowed to transmit data beats only when it has credits available for the respective interface. Upon transmitting a data beat, the host port is expected to decrement its credit counter value. When streaming bridge reads data beats from an interface FIFO, it returns the credit back to the TX host port for the respective interface, which is expected to increment the interface’s credit counter by one. This is illustrated in Figure 38.

NSIP protocol does not allow interleaving of data beats of different packets at a TX interface.

### NSIP TX Signals

The following tables describe the interface pins on a streaming bridge that connects to a streaming host port. Configuration, clocks and reset signals are separately described in the next section.

<Bridge name> is a unique bridge name, which connects a specific host port on a specific host to the NoC. <N> identifies the interface on a streaming bridge; it is one of {a, b, c, d}.

Table 49 NoC Streaming Bridge TX signals from Streaming Host port to NoC

|  |  |  |
| --- | --- | --- |
| **Signal Name** | **Direction** | **Width** |

|  |  |  |
| --- | --- | --- |
| **Host port egress interface \* to NoC** | | |
| host\_dest\_hp*\_<Bridge name>\_<N>* | IN | 8-bit (destination host port id) |
| host\_dest\_int*\_<Bridge name>\_<N>* | IN | 2-bit (destination interface id) |
| host\_xfer\_qos*\_<Bridge name>\_<N>* | IN | 4-bit |
| host\_beat\_valid*\_<Bridge name>\_<N>* | IN | 1-bit |
| host\_beat\_sop*\_<Bridge name>\_<N>* | IN | 1-bit |
| host\_beat\_eop*\_<Bridge name>\_<N>* | IN | 1-bit |
| host\_beat\_mcpkt*\_<Bridge name>\_<N*> | IN | 1-bit (when multicast enabled) |
| host\_beat\_mcdst*\_<Bridge name>\_<N>* | IN | Number of mcast destinations (when multicast enabled) |
| host\_beat\_data*\_<Bridge name>\_<N>* | IN | Power of 2 x cell\_size based on NocStudio specification. |
| noc\_host\_credit\_inc*\_<Bridge name>\_<N>* | OUT | 1-bit |

### NSIP TX Timing Diagram

Figure given below shows an example timing diagram of three single-beat transactions at TX Bridge



Figure 34: Timing of Single beat transactions at TX Bridge

Figure given below shows an example timing diagram of multi-beat transactions at TX Bridge



Figure 35: Timing of Multi beat transactions at TX Bridge

## NSIP Receiver

A host port with receiving NSIP interfaces can use up to four independently flow controlled interfaces, named a, b, c, and d. Just like TX interfaces, each RX interface has its own specified data width which is a power of 2 multiple of its cell size. Each RX interface may receive messages that are composed of one or more data beats – each data beat is received in a single cycle. Data beats are always equal to the data width of the RX interface and are marked with SOP and EOP bits indicating the start of a message and end of the message, respectively. Multiple messages may not interleave on a RX interface, however there may be any number of idle cycles on a RX interface between successive data beats of same of different messages. Furthermore, any combination of the RX interfaces may receive data beats simultaneously.

The 4-bit QoS value is not available at the RX interface; thus, the QoS at TX interfaces is intended only for the NoC consumption. Furthermore, the 8-bit destination identifier and 2-bit interface is not available at the RX interfaces, as these are implicit at each RX interface. Just like TX interfaces, each RX interface uses a separate credit-based flow control between the receiving host port and the streaming bridge. However, unlike TX interfaces, the flow control FIFO for each RX interface is present at host side within the host port, in which the data beats received by the host port are stored. At the NoC side, the streaming bridge maintains a credit counter per RX interface and is allowed to send data beats to a host port RX interface only when it has credits available for the respective interface; credit counter is decremented by one every time a data beat is sent. When host port reads data beats from one of its RX interface FIFOs, it is expected to return the credit back to the streaming bridge for the respective interface, which increments the interface’s credit counter by one. This is illustrated in Figure 38.

NSIP protocol does not allow interleaving of data beats of different messages at a RX interface.

### NSIP RX Signals

Following table describes the signals at the host port side of a rx streaming bridge.

Table 50 NoC Streaming Bridge RX signals from NoC to Streaming Host port

|  |  |  |
| --- | --- | --- |
| **Signal Name** | **Direction** | **Width** |

|  |  |  |
| --- | --- | --- |
| **Host port ingress interface \* from NoC** | | |
| noc\_beat\_valid*\_<Bridge name>\_<N>* | OUT | 1-bit |
| noc\_beat\_sop*\_<Bridge name>\_<N>* | OUT | 1-bit |
| noc\_beat\_eop*\_<Bridge name>\_<N>* | OUT | 1-bit |
| noc\_beat\_data*\_<Bridge name>\_<N>* | OUT | Power of 2 x cell\_size based on NocStudio specification. |
| noc\_bridge\_credit\_inc*\_<Bridge name>\_<N>* | IN | 1-bit |

### NSIP RX Timing Diagram

Figure given below shows an example timing diagram of single-beat transactions at RX bridge



Figure 36: Timing of Single beat transactions at RX Bridge

Figure given below shows an example timing diagram of multi-beat transactions at RX Bridge



Figure 37: Timing of Multi beat transactions at RX Bridge

## Credit Based Flow Control at TX and RX Interfaces

In this section we illustrate the credit-based flow control logic at TX and RX interfaces of NSIP host port in Figure 38. While the overall flow control logic is identical at TX and RX interfaces, the logic is partitioned between credit control logic and flow control FIFO differently at the RX and TX interfaces. At TX interface host port is responsible for containing and managing the credit counters for each interface and NoC contains and manages the flow control FIFOs. At RX interfaces roles are reversed.

CCa

data

(If Cca > 0)

Cca - -

Cca ++

CCb

CCc

CCd

Credit counter at 4

interfaces a,b,c,d

Data FIFOs at 4

interfaces a,b,c,d

credit

…

…

…

Host port

NoC

CCa

data

(If Cca > 0)

Cca - -

Cca ++

CCb

CCc

CCd

Credit counter at 4

interfaces a,b,c,d

Data FIFOs at 4

interfaces a,b,c,d

credit

…

…

…

NoC

Host port

Credit based flow control at TX interfaces

Credit based flow control at RX interfaces

Figure 38: Credit based flow control at TX and RX interfaces of NSIP agents.

Note that the at TX interfaces, the credit control logic is present at the host port while at RX interfaces, the flow control FIFO is present at the host port

Figure given below shows the timing diagram of multi-beat transactions with Credit flow control at TX Bridge



Figure 39: Timing of Multi beat transaction with credit flow control at TX Bridge

Figure given below shows the timing diagram of multi-beat transactions with Credit flow control at RX Bridge



Figure 40: Timing of Multi beat transaction with credit flow control at RX Bridge

## Width Ratios and Conversion Summary

Following list summarizes the streaming interface data width restrictions and constraints on the RX and TX communicating interfaces.

* Each host port interface’s data width must be power of two multiples of cell\_size.
* The minimum and maximum width of interfaces is 1x cell\_size and 64x cell\_size, respectively.
* Interfaces may be marked as single beat or multi beat.
* A single beat interface must always receive or transmit single beat messages.
* A single beat interface must only communicate with single beat interfaces.
* A multi beat interface must only communicate with multibeat interfaces.
* Any two host port interfaces marked as single beat may communicate with each other only if they have same data width.
* Any two host port interfaces marked as multibeat may communicate with each other irrespective of their data width.
* A host port may have any combination of up to four TX and up to four RX active interfaces, and each interface may have any allowed width and single beat property.

The allowed communication between different interfaces marked as single beat or multibeat and having different widths are illustrated between in Figure 41.

Single beat, w1

Single beat, w2

Multi beat, w1

Multi beat, w3

Single beat, w1

Single beat, w2

Multi beat, w1

Multi beat, w3

Allowed communication between TX and Rx interfaces

with the highlighted single/multibeat prop and widths

w1 < w3 < w2

Single beat, w1

Single beat, w2

Multi beat, w1

Multi beat, w3

Single beat, w1

Single beat, w2

Multi beat, w1

Multi beat, w3

Disallowed communication between TX and Rx interfaces

with the highlighted single/multibeat prop and widths

w1 < w3 < w2

Figure 41 Four TX interfaces (left) and four RX interface (right) that may and may not communicate with each other based on interface widths and single beat prop

When a TX interface communicates with a RX interface with different width (none of them can be single beat), TX message data beats undergo width conversion before being delivered at the RX interfaces. If TX interface is wider than the RX interface then, each TX data beat is split into r data beats, where r is the interface width ratio, width (TX)/width (RX). This number is always a power of 2 value as interface widths are always a power of 2 value. When TX interface is narrower than the RX interface then beginning from the SOP data beat, each successive data beats of up to *t* beats of a message at the TX interface are combined into a single data beat delivered at the RX interface, where *t* is the interface width ratio width (RX)/width (TX). If the number of TX data beats modulo t is not zero, then the last succession of combined TX data beats would contain less than t TX data beats. In this case, valid TX beats are combined and the invalid beat locations have random content.

Below is an example to illustrate the width conversion of a message from a narrow TX interface to a wide RX interface, assuming that the TX message is composed of seven data beats and width ratio is four.

0

1

2

3

4

5

6

Data beats at TX interface

SOP

EOP

0

1

2

3

4

5

6

Data beats at RX interface

SOP

EOP

X

LSB

MSB

LSB

MSB

Figure 42 Illustration of width conversion of data beats of a message in NSIP protocol from a narrow TX interface to a wide RX interface

Wide TX to narrow RX width conversion is identical.

## Ordering Requirements

All message with the same QoS value transmitted from a host port’s TX interface to another host port’s RX interface must be delivered in the transmitted order at the RX interface.

Messages with different QoS values between same source and destination host ports interfaces are not required to be ordered.

## NetSpeed Streaming Bridge Overview

This section describes the NetSpeed Streaming Bridge functionality and how it is used in NocStudio to interface with streaming host port devices and how communication between various streaming interfaces is specified.

At TX side, streaming bridge maps the transmitted messages at TX interfaces by host ports to NoC layers and VCs by looking up a VC mapping table (a VC mapping tables exists at each transmitting streaming bridge). The bridge also performs the route computation and provides the needed infrastructure for the end-to-end QoS (fixed priority as well as the weights) enforcement. At RX side, streaming bridge simply delivered the NoC messages to the host port interfaces. A high-level diagram of streaming bridge is shown in Figure 43. There are up to four TX and up to four RX interfaces to the host port. These interfaces are named a, b, c, and d. On the NoC side, a streaming bridge may connect to up to 8 NoC layers, *i.e.* 8 routers ports, a router from each layer. Each router port may have up to 4 VCs indicated by vc0, vc1, vc2, and vc3. TX part of the bridge sends data from host port to NoC, and RX part does the opposite.



Figure 43 NetSpeed streaming bridge Interfaces to host port and to NoC

### Functions of Transmitting (TX) Streaming Bridge

A transmitting streaming bridge performs the following functions.

1. Maps outgoing messages from host port TX interface to NoC layer and VC.
2. Determine route for the message.
3. Packetizes the message into NoC packets.
4. Implements QoS functions, both strict priority and weighted bandwidth.
5. Implements the credit-based flow control at the host port TX interfaces.
6. Implements the flow control at the router side.
7. Performs arbitration between the interfaces if they contend for the same VC and/or NoC layer.

### Functions of Receiving (RX) Streaming Bridge

A receiving streaming bridge performs the following functions.

1. De-packetizes NoC packets into messages and deliver at the host port RX interfaces.
2. Implements the flow control at the router side.
3. Implements the credit-based flow control at the RX host port interfaces.
4. Performs arbitration between VC and/or NoC layer if they contend for the same interface.

## Adding Streaming Bridge and Traffic in NocStudio

Below is an illustration of an example of adding hosts with streaming host ports with different number of interfaces and using *add\_traffic* to add a request and response messages between the host port’s interfaces.

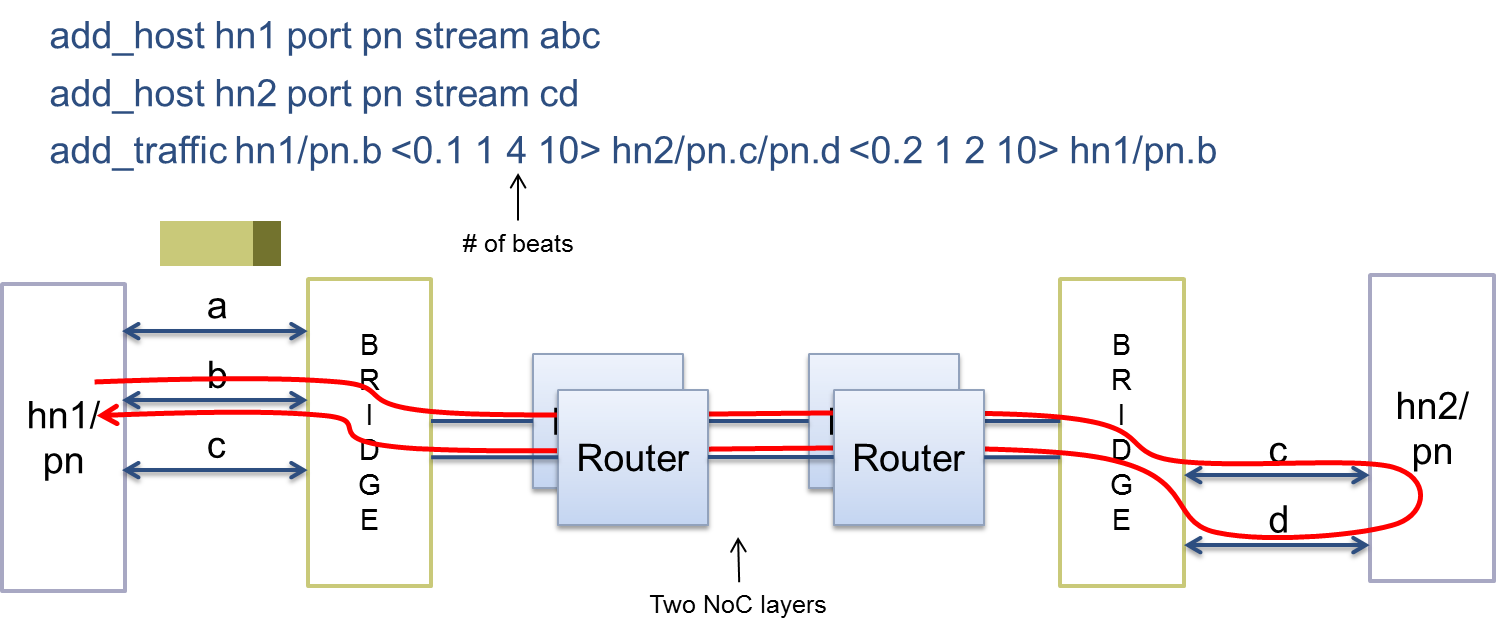


Figure 44 Illustration of a NoC with streaming host ports and traffic between them

Here two hosts named hn1 and hn2 are added. Each has a single host port named *pn* of protocol type stream, and they have a,b,c and c,d interfaces, respectively. Traffic is added which consists of a request and response. hn1/pn makes a request message at interface b to destination hn2/pn’s interface c. hn2/pn responds with a response message at interface d which is sent to the interface b of hn1/pn. Notice that the response message may be sent by and destined to a different interface than the interface from where the request message has arrived from. Host hn2 in this case must be able to determine the correct destination and put this information in the response message so that NoC can deliver it correctly. Alternatively, such traffic may be added as two separate add\_traffic commands, one for request and one for response, as shown below:

add\_traffic hn1/pn.b 0.1 1 4 10 hn2/pn.c

add\_traffic hn2/pn.d 0.2 1 2 10 hn1/pn.b

add\_dep hn2/pn.c hn2/pn.d

Notice that here an additional ***add\_dep*** command is used to indicate to NocStudio that the second message is a response of the first message, and that the second message may backpressure the first message. add\_dep command must be used to correctly specify the dependencies between various interfaces of streaming bridges to ensure that the resulting NoC constructed by NocStudio is deadlock free.

## Streaming Bridge Specification

This section describes the Streaming bridge interfaces in greater details, and the functional specification of various streaming bridge components.

### Streaming Host Port Requirements

Following are requirements, and expected behavior of the host port interfaces connecting to a streaming bridge:

* Streaming host ports using the streaming bridges to communicate with each other can use up to four separate physical interfaces supported by the bridge. The interfaces are named a, b, c and d, and the list of active interfaces of a host port may be provided with the **add\_host** command. Each interface can be uni- or bi-directional. Whether an interface is bidirectional or not depends on the traffic specification, *i.e.* if traffic transaction hops only leave (arrive at) an interface then it becomes an output (input) only interface, else it remains bi-directional.
* The width of each direction of an interface can be set to any power-of-two multiple of the cell size and can be viewed or modified using the **bridge\_prop** command. If a NoC supports multiple cell sizes (**cell\_size** = -1), the interface must be a power-of-two multiple of its cell size. An interface may be marked as single beat using **ifce\_prop** is\_single\_beat yes command; if an interface is marked as single beat then it may not send or receive a multi-beat message and may not communicate with multibeat interfaces.
* Two communicating interfaces must have same is\_single\_beat property.
* If two communicating interfaces are single beat, then their width must be the same. They may have different width only if they are not marked as single beat.
* When two communicating non-single-beat interfaces have identical width, the transmitted data beats at a TX interfaces are delivered in order at the destination interface and the numbers of transmitted and received data beats are equal.
* When two communicating non-single beat interfaces have different widths, the ratio of their widths, r, a power of two value, determines the number of beats in received message. Going from a wider interface to a narrower interface, each wider data beat is divided into r narrower data beats during delivery at the destination interface; the LSBs of the wider flit form the first narrow flits which are delivered before the MSBs. Going from a narrow to wide interface, up to r transmitted data beats are combined into a single data beat delivered at the RX interface; the first narrow beat becomes the LSBs of the first wider data beat delivered at the RX interface. During combining, if an EOP arrives and there are not r data beats to combine then the combined data beat at RX interface is padded.
* Any source streaming host port’s interface may communicate with any destination streaming host port’s interface, as long as they have identical is\_single\_beat property.
* Each injected message must have the destination bridge id and interface id and a 4-bit QoS, each of which must stay constant for all data beats of the message at the TX interface. A TX interface can only send messages with those destination bridges and interface ids and QoS values that are specified in the **add\_traffic**. Any other messages will be dropped.
* For streaming host ports, each transaction hop or message must be explicitly specified using **add\_traffic** and there is no response automatically associated with any request.
* An interface may transmit or receive traffic of multiple priorities and weights, which is indicated in the message using the 4-bit QoS signal at the TX interface. Refer to QoS section for more details on this signal. At the bridge, each 4-bit QoS value maps to a 2-bit priority and 8-bit weight.

### Injection and Ejection Port VC Width Computation

Following rules are generally (but not always) used by NocStudio to determine the width of the VCs at the injection and ejection ports to which the streaming bridge is connected.

* The width of injection/ejection port VCs of a router to which Streaming interfaces are connected are determined based on the Streaming interface data widths and how VCs and interfaces are connected.
* At RX side, if a router’ ejection port sends data to a single interface then the interface width and the port and VC width are the same.
* At RX side, if a router’ ejection port sends data to multiple interfaces then the width a VC on the port will be equal to the width of the widest interface to which it sends data.
* At TX side, if a single interface sends data to a router’ injection port then the interface width and the port and VC width are the same.
* At TX side, if multiple interfaces send data to a router’ injection port then the width of a VCs on the port will be equal to the width of the widest interface from which it received data.

## QoS

There are two types of QoS attributes that can be assigned to the messages; the 4-bit QoS signal along with a message is used to describe these two for the message.

1. strict priority
2. weighted bandwidth allocation

There is a fixed mapping between 4-bit QoS and 2-bit priority at every host port; by default, the 2-LSB bits of QoS forms the priority of the message. This mapping can be modified in NocStudio using **bridge\_prop** command.

The QoS is used to determine the VC and NoC layer, therefore it is used as one of the indices into the VC mapping table at TX bridges. VCs and routes are allocated only for those QoS values that were specified in the traffic profile using **add\_traffic**. A message that was not specified using **add\_traffic** will cause a protocol error at the bridge.

|  |  |
| --- | --- |
| priority (2-bit) | weight (8-bit) |
| QoS (4-bit) |  |
|  |  |
|  |  |

With each QoS, there is an associated weight value too. The reset time weight value at each streaming bridge is configurable via NocStudio using **bridge\_prop** command. Later the weight values must be re-configured via regbus. Alternatively, one may modify the weight of by using different QoS values for the messages, assuming that they are added via **add\_traffic** and they have different configured weights.

### Programmable Configurations

The weight for each QoS level is programmable via regbus. Initial weight values at reset time are configured via NocStudio **bridge\_prop** command.

### VC Mapping table and QoS priority

VC Mapping table at TX wide of streaming bridge is configured based on the destination host port and interface and QoS value of various messages at the source host port and interfaces. Only the messages that are added via **add\_traffic** command are supported. Any other message will cause a protocol error. The VC mapping table is designed such that the resulting logic is optimized based on the VC and NoC layers of various messages, QoS values and the destination bridge and interface id.

VC mapping table is not configurable; it is set during the NoC design time.

## Multicast

Multicast allows for a single packet transmitted from a source bridge to be replicated within the NoC and delivered to multiple destinations. The multicast feature can be enabled or disabled per configuration. When enabled, both unicast and multicast traffic are supported. For multicast, the source host supplies two addition signals per packet: host\_beat\_mcpkt and host\_beat\_mcdst. The host\_beat\_mcpkt input is a single bit signal that when set to 1 identifies the packet as a multicast packet. When 0, the packet is a standard unicast packet. For multicast packets, the host\_beat\_mcdst input specifies the destinations for the multicast packet. Each bit in the host\_beat\_mcdst vector corresponds to a multicast destination or destination group.

### Restrictions and Assumptions

* NocStudio limits multicast to a single NoC layer. This layer supports both unicast and multicast packets.
* A single bridge interface generates and receives multicast traffic.
* There is no multicast from a NoC layer to multiple host interfaces at the destination.
* No upsizing/downsizing is allowed for multicast packets.
* No clock crossing is supported in the VC buffers for multicast layers.
* Only single-flit packets are supported for multicast.