Brief

HyNoC (High-performance NoC) is a Network-On-a-Chip dedicated to High Performance Computing with static and dynamic routing capabilities. It can manage any topologies by assembling routers with a variable number of ports. Each router implements a distributed arbitration schemes within each port.

Features

The HyNoC router is built upon following characteristics:

- Wormhole switching,
- Buffered (FIFO) flow control,
- Distributed arbitration,
- Fully parallel round robin in each distributed arbiter,
- Dedicated clock domain to each port.

A router can be delivered in a fully synchronous way, ie the router and the port use the same clock domain. For more complex designs, the router can also be delivered with dedicated clock domains for the router itself (core clock) and for the interfaces (ifce clocks).
A router can also be delivered with 3 to 9 interfaces, ask us for the exact topology you need. Each interface embeds a synchonous or asynchronous FIFO with a depth configurable between 2 to 64 elements.

Short description table

The following table presents some place and route results. The figures given here are obtained with altera tools tuned for maximum performances in terms of frequency. The router considered uses 32-bit payload words with 16-element dual-clocked FIFO for each router interface. Here, the number of memory bit used is directly proportional to the size of the 16-element FIFOs used: for a 3-port 32-bit router it represents $3 \times 16 \times 32 = 1536$ bits.

Core Information					
FPGA family	Altera, Xilinx				
Protocol	Stream				
Resources example for Altera Cyclone IV E					
Router Type	**LEs**	**Mem (bits)**	**FFs**	**Core** f_{max}	**Ifce** f_{max}
32-bit 3-port	730	1600	450	280	315
32-bit 5-port	1600	2650	750	245	315
32-bit 7-port	2860	3700	1113	245	315
32-bit 9-port	4250	4800	1500	230	315

Summary

1 Revisions **2**

2 Description **3**
2.1 Architecture 3

3 First Layer Protocol **4**
3.1 Packet routing 4
3.2 Packet structure 5
3.3 Routing protocols 5
3.3.1 Unicast Circuit Switch routing 6
3.3.2 Multicast Circuit Switch routing 7
3.3.3 Combining multiple routing policies 8

4 Router example: 32-bit 3-port **8**
4.1 introduction 8
4.2 Packet structure 8
4.3 Parameters 9
4.4 Ports 9
4.5 Local Interface 9
4.6 Test environment 10

References **13**

List of Figures

1 Router interconnections overview 3
2 Internal data path of a 3-port router 4
3 Internal control path of a 3-port router 4
4 Unicast Circuit Switch hops encoding 6
5 Example of a unicast circuit switch packet of a 64-bit NoC 7
6 Example of a unicast circuit switch packet of a 16-bit NoC 7
7 Unicast packet structure for 3-port 32-bit NoC 8
8 Multicast packet structure for 3-port 32-bit NoC 9
9 Unicast test for 3-port 32-bit routers 10

List of Tables

1 Revisions table 2
2 Packet definition 5
3 Smallest packet 5
4 Supported routing protocols 5
5 Unicast Circuit Switch Field Description 6
6 Multicast Circuit Switch Field Description 8
7 Parameters of the 3-port 32-bit router 9
8 Ports of the 3-port 32-bit router 10
9 Local interface port description - Router side 11
10 Local interface port description - Client side 11
11 Unicast routing table of 3-port 32-bit router 12
12 Multicast routing table of 3-port 32-bit router 12

1 Revisions

Date	Version	Modification
2020-03-01	1.0.1	Update routing protocol definitions.
2020-02-29	1.0.0	Initial version.

Table 1: Revisions table

2 Description

2.1 Architecture

A router is made of full-duplex ports which rely on ingress and egress interfaces and that are respectively plugged to egress and ingress of another router ports. Each router has its own clock domains and the crossing rely on dual-clocked FIFO at the input of each ingress interface. A node is attached to a router using a local interface, this interface instantiates an extra FIFO to the egress output to support the node's clock domain.

Figure 1 shows both internal organization of 5-port routers and how they are connected to their neighbors.

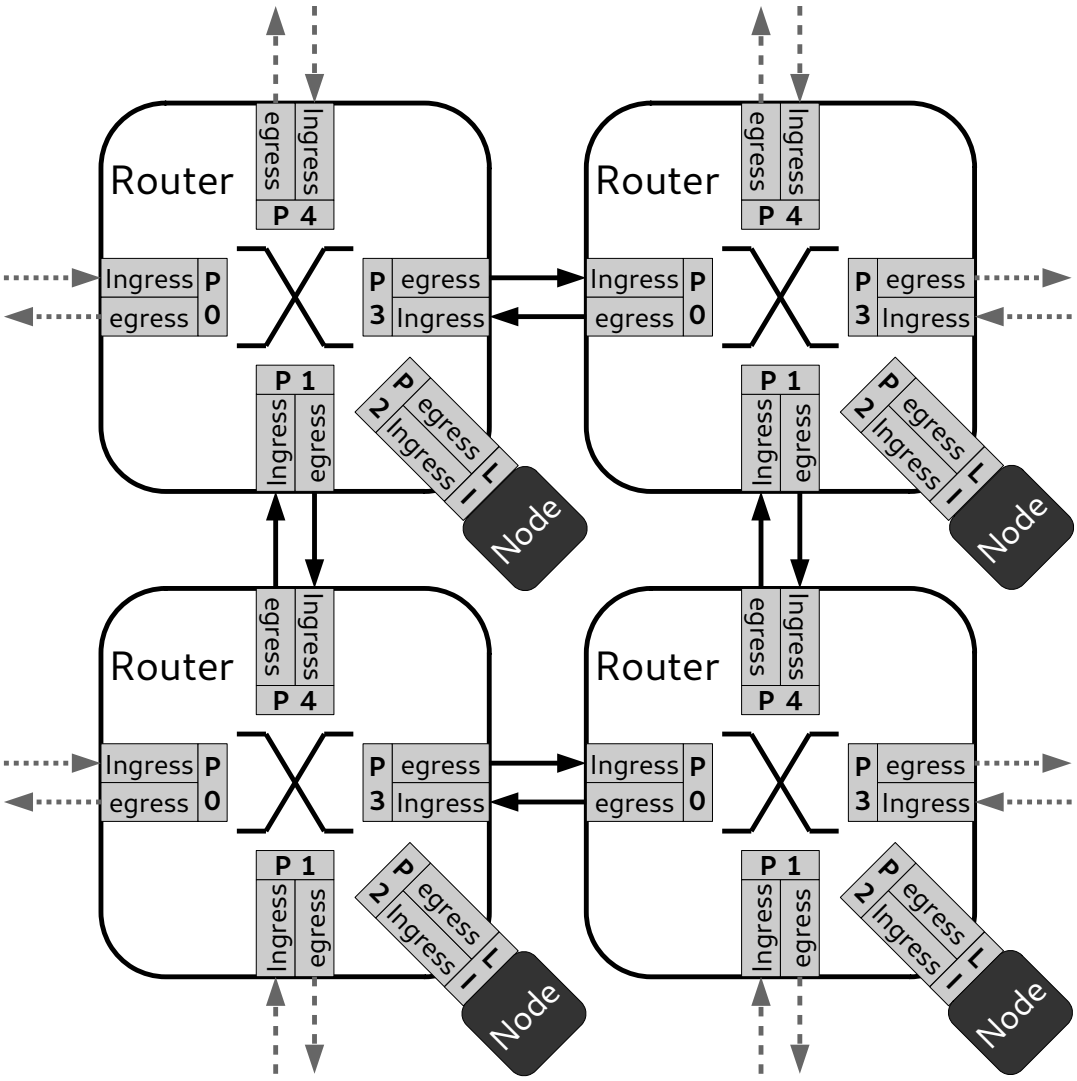

Figure 1: Router interconnections overview

The router is a full crossbar, meaning that each ports can establish a communication to all ports except with itself. Multiple paths inside the router can be opened at the same time thanks to the distributed arbitration scheme. Thus, each egress port embeds an output multiplexer and an arbiter to route, without starvation, data from the ingress ports that has requested a transfer.

Figure 2 presents the data paths between ingress and egress. Each ingress broadcasts its data to all egress ports except to the one which is grouped in the same router port. Then, the egress will forward data to the next router depending on the request asserted by the ingress ports and also depending on the state of the arbiter.

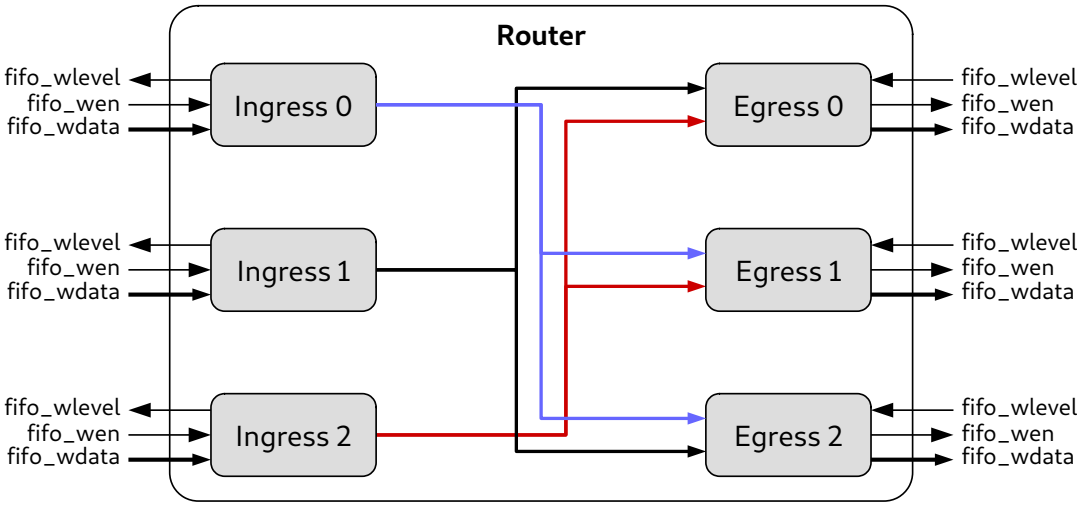

Figure 2: Internal data path of a 3-port router

The control path is shown in the figure 3. Contrary to the data path structure, there is no broadcast of control signals. Each ingress has dedicated links with each egress to assert transmit requests and to receive the grant from an egress. Once the grand is received, the ingress can push data.

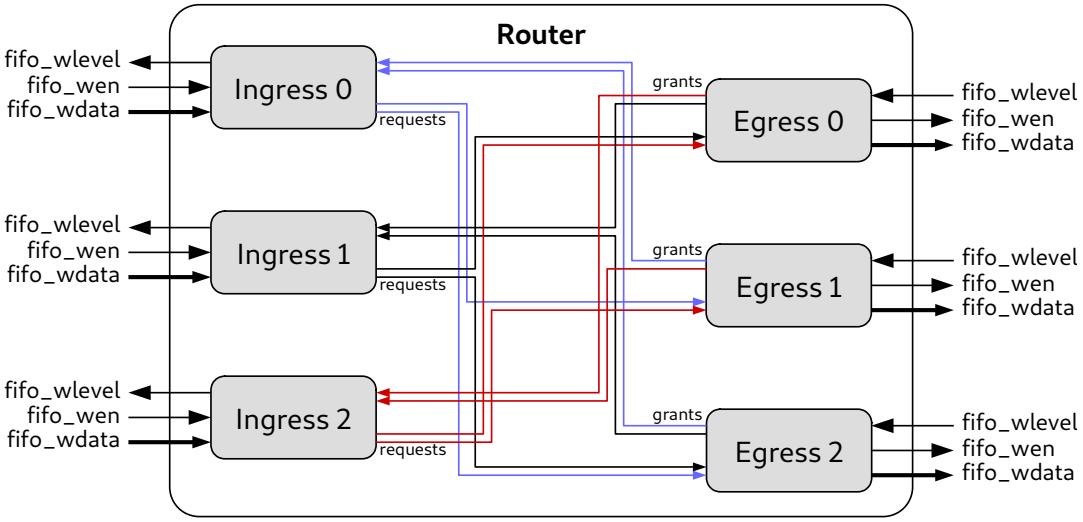

Figure 3: Internal control path of a 3-port router

The egress arbiter uses a parallel round-robin arbiter which allows to schedule all ingress requests without starvation in a fix latency.

3 First Layer Protocol

3.1 Packet routing

HyNoC uses source routing techniques to send data through routers. Instead of addressing node with coordinates, for instance (x,y) for a network with a mesh topology, we define in the packet's header the route to take through the network. This means that the header contains a list of output ports, called hops, of routers to cross. This technique is used by [MS07] and [MK10] to avoid congestion with a low packet header overhead using a simple encoding hop scheme.

A routing algorithm can be defined upon Source Routing network depending on topology. [MK10] survey presents such a technique. The routing algorithm generates for each communication the list of hops either dynamically (in hardware in the local interface or in software using the node's processor) or statically at compilation time.

Routes are established in a distributed way in each router, this technique is presented by [PMMC11]. Each egress interface manage itself and it is in charge of selecting the right ingress interface using a simple request/acknowledge protocol. Moreover, the egress interface embeds a LUT-based parallel round robin arbiter to respond in fix amount of time without creating any starvation.

3.2 Packet structure

A packet is composed of multiple flits, a flit is the smallest unit transmitted over the network and it is composed of $K + 1$ bits. The most significant bit is the last bit and it indicates the last flit of a packet.

The table 2 shows the packet structure. A packet is built with at least one Routing flit followed by at least one Payload flit. If multiple header flits are used to encode the path through the network, the router will consider only the first flit as a header and the remaining flits as payload. When all hops of the header are marked used, that header flit will not be forwarded to the next router. Thus, the next router will use the next flit as a header flit and so on.

Last bit	**payload (K bits)**	
0	4-bit Proto	Routing flit
0	...	...
0	Payload flit	
0	...	
1	last Payload flit	

Table 2: Packet definition

The table 3 presents the smallest packet that can be sent over the network. The number of router that it can pass through depends on the protocol, the payload width and the number of ports within a router.

Last flit bit	**payload (K bits)**	
0	4-bit Proto	Routing flit
1	last Payload flit	

Table 3: Smallest packet

3.3 Routing protocols

Multiple protocols can be supported in a routing flit depending on the 4-bit Proto value. The table 4 shows supported protocols and the following sub-sections describe how they work.

Proto value	**Routing method**
4'b0000	Unicast Circuit Switch.
4'b0001	Multicast Circuit Switch.
4'b1000	XY routing (not yet implemented).
4'b1111	Forbidden value.

Table 4: Supported routing protocols

3.3.1 Unicast Circuit Switch routing

In this routing policy, the routing flit is a list of router egress ID to cross. The table 5 describes the unicast circuit switch protocol field. For a router with P ports, the Hop field is encoded using $W = \lceil \log_2(P-1) \rceil$ bits. The index field points the correct hop to be used by the router ingress port. The Gap width can be in $[0, W-1]$.

Proto	**Hops list (K-4 bits)**				
4'b0000	Gap	Hop H-1	...	Hop 0	Index

Table 5: Unicast Circuit Switch Field Description

The index is numbered between $[0, H-1]$ for a flit with H hops and is initialized to $H-1$. The index is decremented once the pointed hop is used to open the path inside the router. If the index is null before path opening, the related Network Hops flit will not be transmitted to the router's egress port, else the index is decremented and the updated Routing flit is transmitted. A unicast flit can embed less than the maximum allowed number of hop fields by just initializing the index accordingly.

The hop encoding is based upon the fact that a same packet can not use the same port for incoming and outgoing. The number of accessible egress port are $P-1$ for a router with P ports. This technique reduces the internal router's crossbar size.

The figure 4 shows the hops list $0 \rightarrow 2 \rightarrow 0 \rightarrow 1 \rightarrow 2$ to establish a communication channel from node $(0,0)$ to node $(2,2)$ in a 3×3 mesh network. This NoC is built upon 5-port routers and only two bits are needed to encode each hop.

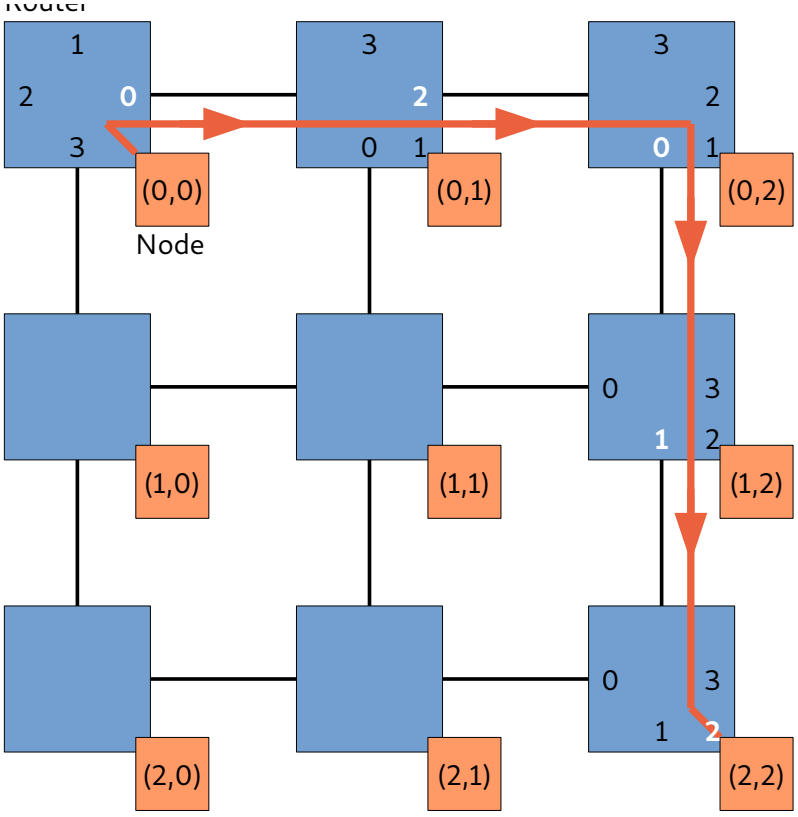

Figure 4: Unicast Circuit Switch hops encoding

Each ports encodes the next counterclockwise egress with the ID zero. In other words, the selected egress ID must be calculated by taking into account the ingress id, this is a relative egress addressing.

An example of an unicast packet is given in figure 5. It corresponds to the path described in the figure 4, the NoC payload width is 64-bit.

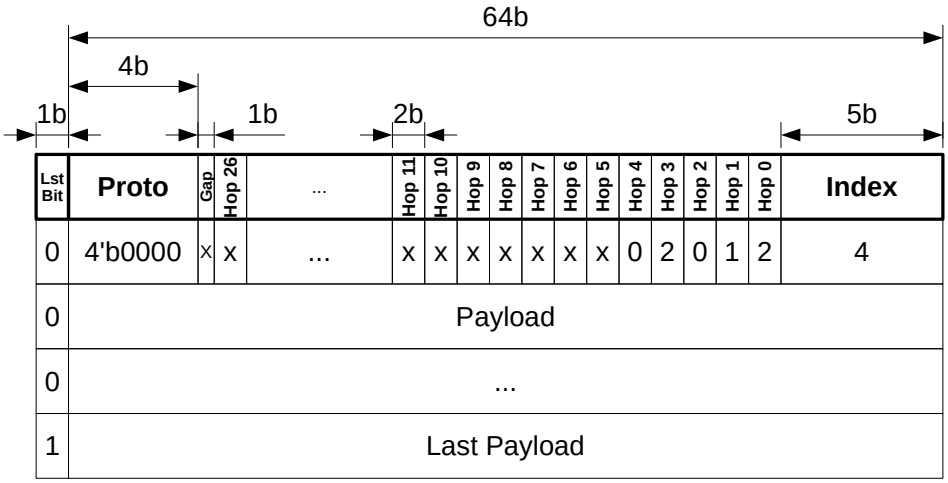

Figure 5: Example of a unicast circuit switch packet of a 64-bit NoC

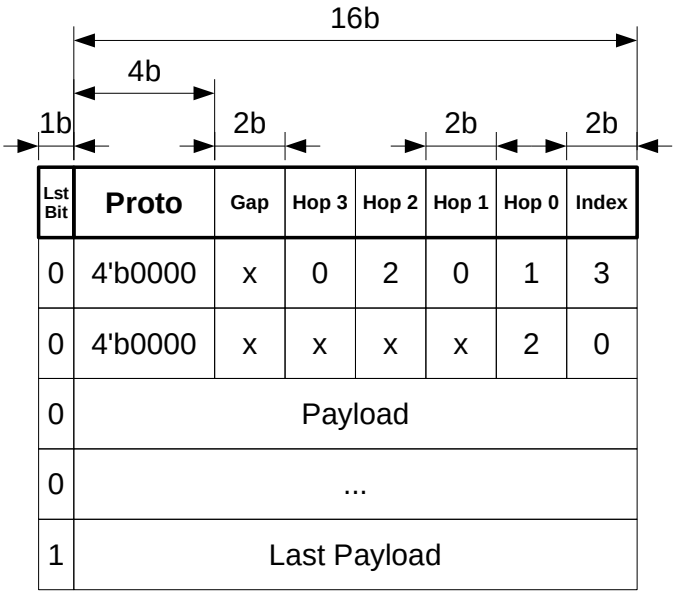

Figure 6: Example of a unicast circuit switch packet of a 16-bit NoC

3.3.2 Multicast Circuit Switch routing

The multicast routing policy permits to target multiple egress ports at a time from a unique ingress port. The table 6 describes the multicast circuit switch protcol field. For a router with P ports, the Hop field width is $W = P - 1$ bits. The index field points the correct hop to be used by the router ingress port. The Gap width can be in $[0, W - 1]$.

As a reminder, the index must be initialized to $H - 1$, because the ingress port forwards the flit by decrementing the index. A multicast flit can embed less than the maximum allowed number of hop fields by just initializing the index accordingly.

Proto	Hops list (K-4 bits)				
4'b0001	Gap	Hop H-1	...	Hop 0	Index

Table 6: Multicast Circuit Switch Field Description

3.3.3 Combining multiple routing policies

Multiple types of routing flits can be mixed at the beginning of a packet.

4 Router example: 32-bit 3-port

4.1 introduction

The 32-bit 3-port router is delivered with a testbench that demonstrates the functionality and the behavior of the router. The following subsections describe the packet structure used, the module parameters and ports description and an overview of the testing environment proposed.

4.2 Packet structure

The 32-bit 3-port NoC can hold up to 23 hops for unicast packets and up to 11 hops for multicast packets. The figures 7 and 8 describe the packets structures for both unicast and multicast protocol.

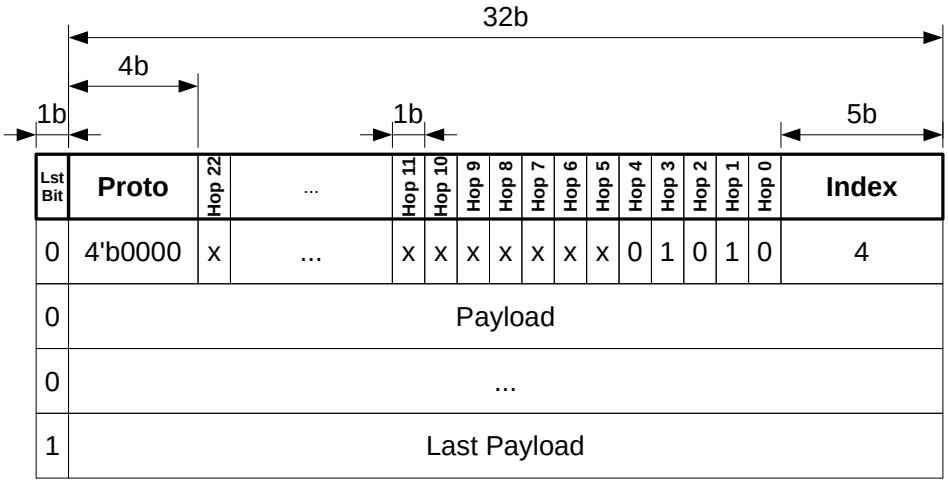

Figure 7: Unicast packet structure for 3-port 32-bit NoC

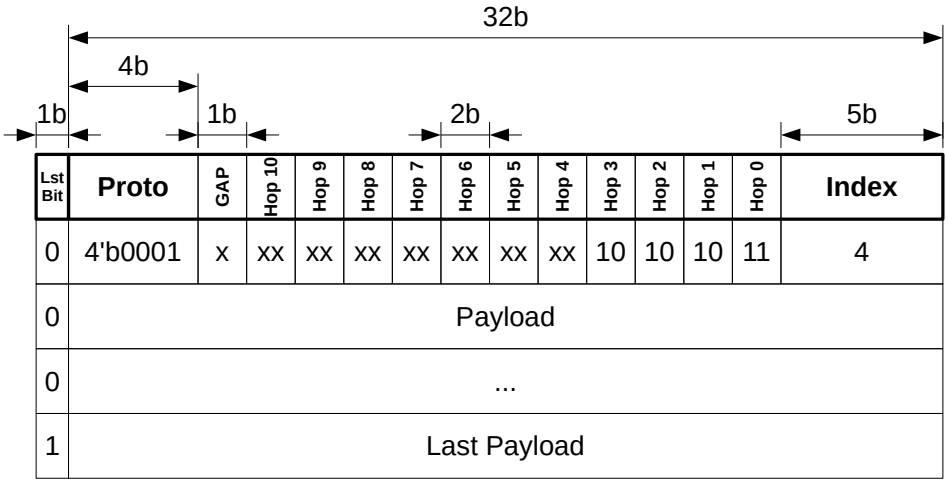

Figure 8: Multicast packet structure for 3-port 32-bit NoC

4.3 Parameters

The following module parameters declared in the 3-port router module are presented in the following table 7. Please note that some can be directly inlined, depending on the delivery content.

Name	Default Value	Description
INDEX_WIDTH	5	Bit width of the index in an address flit.
LOG2_FIFO_DEPTH	5	Size of the FIFO inserted in each port's ingress expressed in log_2 basis.
PAYLOAD_WIDTH	32	Bit width of the payload.
FLIT_WIDTH	PAYLOAD_WIDTH + 1	Bit width of the flit.
PRRA_PIPELINE	0	2-cycle parallel round-robin arbiter response when set to 0 else 3-cycle.
SINGLE_CLOCK_ROUTER	0	When set to 1, each port uses the router clock instead of its own clock to reduce the traversal latency.
ENABLE_MCAST_ROUTING	1	When set to 1, enable the multicast routing protocol.

Table 7: Parameters of the 3-port 32-bit router

4.4 Ports

The router ports description is presented in 8. The port is composed of two parts, an ingress side and an egress side. The ingress side receives data and the egress side sends data. When connecting two ports, the egress port X must be connected to ingress port Y and vice-versa.

4.5 Local Interface

A local interface can be used in order to connect a client to a router port. The local interface instantiates an extra FIFO to the egress router port in order to let the client sends an receives data at its own pace.

The local interface exposes a router side that connects to the egress/ingress router port and a local side that exposes FIFO interfaces to the client in order to send and receive data. The tables 9 and 10 presents the local interface ports.

Name	Dir	Width	Description
router_clk	In	1	Internal router clock.
router_srst	In	1	Internal router synchronous reset.
portX_ingress_clk	In	1	Clock of ingress port X, not used if SINGLE_CLOCK_ROUTER is set to 1.
portX_ingress_srst	In	1	Synchronous reset of the ingress port X, not used if SINGLE_CLOCK_ROUTER is set to 1.
portX_ingress_write	In	1	Send the word in the ingress port X. Must be set only one cycle.
portX_ingress_data	In	FLIT_WIDTH	Data to send to the ingress port X.
portX_ingress_full	Out	1	Full signal of the ingress port X. If full, the write request is ignored.
portX_ingress_fifo_level	Out	LOG2_FIFO_DEPTH+1	FIFO level of the ingress port X.
portX_egress_clk	In	1	Clock of egress port X, not used if SINGLE_CLOCK_ROUTER is set to 1.
portX_egress_srst	In	1	Synchronous reset of the egress port X, not used if SINGLE_CLOCK_ROUTER is set to 1.
portX_egress_write	Out	1	A data is ready on egress port X, set one cycle per data to read.
portX_egress_data	Out	FLIT_WIDTH	Data to read on egress port X.
portX_egress_fifo_level	In	LOG2_FIFO_DEPTH+1	Input FIFO level on egress port X, it must be connected to another router.

Table 8: Ports of the 3-port 32-bit router

4.6 Test environment

Two tests are delivered with the 3-port 32-bit NoC to illustrate the two possible communication modes: unicast and multicast. Both tests use the same network topology as depicted in the figure 9. The design proposed here consist in 2 routers connected with 4 nodes and each node is connected to the router using a local interface (LI).

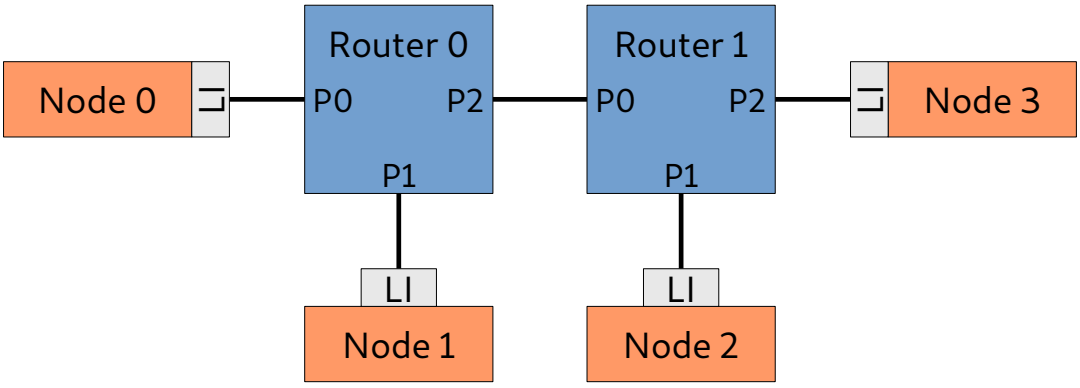

Figure 9: Unicast test for 3-port 32-bit routers

Unicast The first test is related to unicast transfers, where each node send packets to all other nodes. The hops encoded in the routing flits can be established using the table 11 and all node dialog combinations are

Router side			
Name	**Dir**	**Width**	**Description**
port_ingress_srst	Out	1	
port_ingress_clk	Out	1	
port_ingress_write	Out	1	
port_ingress_data	Out	FLIT_WIDTH	
port_ingress_full	In	1	
port_ingress_fifo_level	In	LOG2_FIFO_DEPTH + 1	
port_egress_srst	In	1	
port_egress_clk	In	1	
port_egress_write	In	1	
port_egress_data	In	FLIT_WIDTH	
port_egress_fifo_level	Out	LOG2_FIFO_DEPTH+1	

Table 9: Local interface port description - Router side

Client side			
Name	**Dir**	**Width**	**Description**
local_clk	In	1	
local_srst	In	1	
local_ingress_write	In	1	
local_ingress_data	In	FLIT_WIDTH	
local_ingress_full	Out	1	
local_ingress_fifo_level	Out	LOG2_FIFO_DEPTH + 1	
local_egress_read	In	1	
local_egress_data	Out	FLIT_WIDTH	
local_egress_empty	Out	1	
local_egress_fifo_level	Out	LOG2_FIFO_DEPTH + 1	

Table 10: Local interface port description - Client side

listed hereafter:

- Node 0 $\rightarrow$ Node 1: hops $[0]$
- Node 0 $\rightarrow$ Node 2: hops $[1 \rightarrow 0]$
- Node 0 $\rightarrow$ Node 3: hops $[1 \rightarrow 1]$
- Node 1 $\rightarrow$ Node 0: hops $[1]$
- Node 1 $\rightarrow$ Node 2: hops $[0 \rightarrow 0]$
- Node 1 $\rightarrow$ Node 3: hops $[0 \rightarrow 1]$
- Node 2 $\rightarrow$ Node 0: hops $[1 \rightarrow 0]$
- Node 2 $\rightarrow$ Node 1: hops $[1 \rightarrow 1]$
- Node 2 $\rightarrow$ Node 3: hops $[0]$
- Node 3 $\rightarrow$ Node 0: hops $[0 \rightarrow 0]$

- Node 3 $\rightarrow$ Node 1: hops $[0 \rightarrow 1]$
- Node 3 $\rightarrow$ Node 2: hops $[1]$

Direction	Unicast
P0 $\rightarrow$ P1	1'b0
P0 $\rightarrow$ P2	1'b1
P1 $\rightarrow$ P2	1'b0
P1 $\rightarrow$ P0	1'b1
P2 $\rightarrow$ P0	1'b0
P2 $\rightarrow$ P1	1'b1

Table 11: Unicast routing table of 3-port 32-bit router

Multicast The second test is a multicast transfer from node 0 to node 2 and node 3 and simultaneously from node 3 to node 0 and 1. The address flits can be deduced using the routing table 12 and are presented hereafter:

- Node 0 $\rightarrow$ (Node 2, Node 3): hops $[b10 \rightarrow b11]$
- Node 3 $\rightarrow$ (Node 0, Node 1): hops $[b01 \rightarrow b11]$

Direction	Multicast
P0 $\rightarrow$ P1	2'bx1
P0 $\rightarrow$ P2	2'b1x
P1 $\rightarrow$ P2	2'bx1
P1 $\rightarrow$ P0	2'b1x
P2 $\rightarrow$ P0	2'bx1
P2 $\rightarrow$ P1	2'b1x

Table 12: Multicast routing table of 3-port 32-bit router

References

[MK10] S. Mubeen and S. Kumar. Designing efficient source routing for mesh topology network on chip platforms. In *Digital System Design: Architectures, Methods and Tools (DSD), 2010 13th Euromicro Conference on*, pages 181–188, Sept 2010. *(cited on pages 4 and 5)*

[MS07] L. Ma and Y. Sun. On-chip network design automation with source routing switches. *Tsinghua Science and Technology*, 12(1):77–85, Feb 2007. *(cited on page 4)*

[PMMC11] Julian Pontes, Matheus Moreira, Fernando Moraes, and Ney Calazans. *Hermes-A – An Asynchronous NoC Router with Distributed Routing*, pages 150–159. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. *(cited on page 5)*