RFNoC Crossbar Architectures

Background

Networking 101

Main Components of a Network

1. **Network Topology**
   - The arrangement of network components (terminals, routers, links, etc)
2. **Routing Algorithm**
   - The path selection for packets traversing the network
3. **Flow Control**
   - The process of managing the rate of transmission between components
4. **Microarchitecture**
   - The organization and implementation of router components

Definitions

- The ***throughput*** of a network is the data rate (bps) that the network accepts per input port
- The ***injection bandwidth*** is the max throughput for a given channel
- The ***channel load*** is the ratio between the bandwidth on a channel to the injection bandwidth
- The ***latency*** is the amount of time it takes to traverse the network

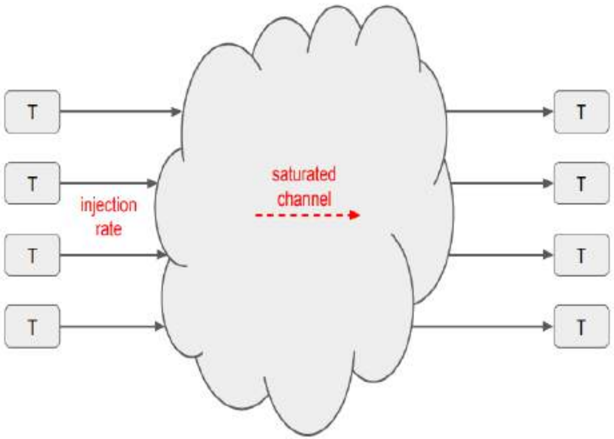
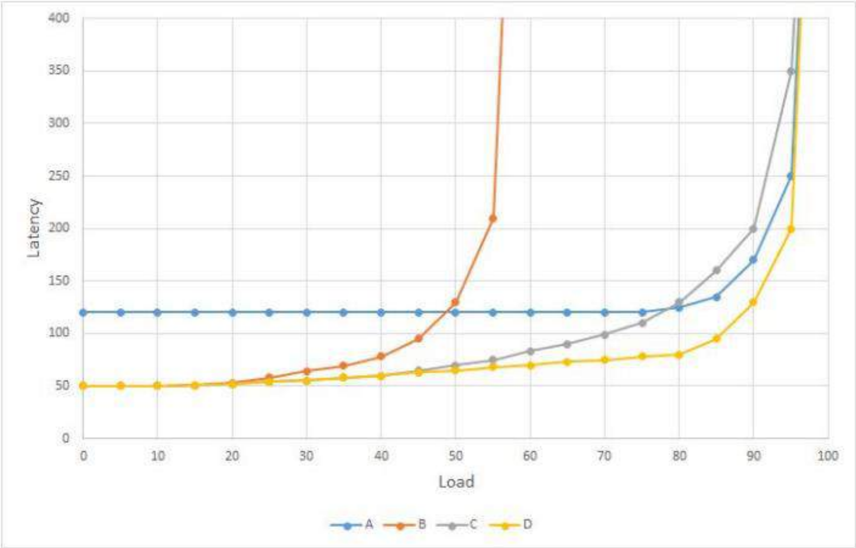


Image Credits: http://web.stanford.edu/class/ee382c/lectures/01_intro.pdf

Load vs Latency Graph

Used to evaluate network performance under various load conditions. Traces represent percentile latency for various implementations. Conveys:

- Statistical spread of latency
- Maximum load tolerated by network



Examples:

1. Blue: High latency network with high load capacity
2. Orange: Low latency network with low capacity

Traffic Patterns

Network performance depends on the traffic pattern i.e. the pattern of the source to destination paths.

Examples:

- **Uniform**: A node sends to all nodes with equal probability
- **Uniform Others**: A node sends to all other nodes with equal probability
- **Neighbor**: A node only sends traffic to their neighbor
- **Bit Complement**: A node only sends traffic to the diametrically opposite node
- **Sequential**: A node sends to all nodes sequentially
- **Loopback**: A nodes sends to itself

Uniform Random (with self)

0.25	0.25	0.25	0.25
0.25	0.25	0.25	0.25
0.25	0.25	0.25	0.25
0.25	0.25	0.25	0.25

Uniform Random (without self)

0.00	0.33	0.33	0.33
0.33	0.00	0.33	0.33
0.33	0.33	0.00	0.33
0.33	0.33	0.33	0.00

Neighbor `dest=(src+1)%N`

0.00	1.00	0.00	0.00
0.00	0.00	1.00	0.00
0.00	0.00	0.00	1.00
1.00	0.00	0.00	0.00

Bit Complement `dest=(~src)%N`

0.00	0.00	0.00	1.00
0.00	0.00	1.00	0.00
0.00	1.00	0.00	0.00
1.00	0.00	0.00	0.00

Random Permutation

1.00	0.00	0.00	0.00
0.00	0.00	0.00	1.00
0.00	1.00	0.00	0.00
0.00	0.00	1.00	0.00

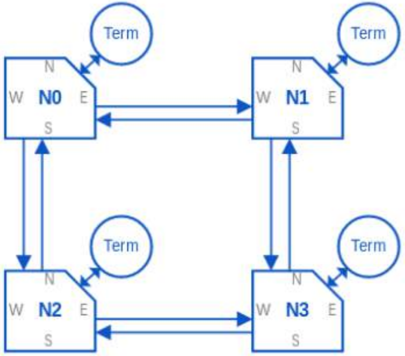
Other

0.20	0.40	0.30	0.10
1.00	0.00	0.00	0.00
0.05	0.70	0.15	0.10
0.00	0.50	0.50	0.00

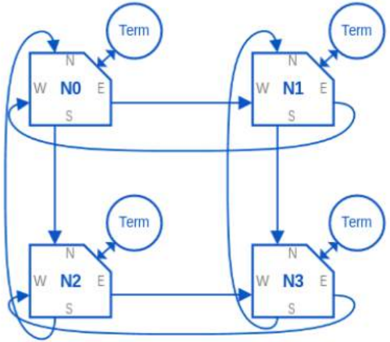
RFNoC Crossbar Implementations

axis_ctrl_crossbar_2d_mesh

- **Topology**: Bidirectional Mesh or Unidirectional Torus
- **Routing**: Wormhole or Store-and-Fwd
  - Wormhole: A packet can be in several routers at a time.
  - Store-and-Fwd: A packet is completely buffered in one router before moving to the next one.
- **Flow Control**: Packet buffer with cut-through
  - Packet buffer: Entire packet is buffered in router
  - Cut-through: Only flits (words) are backpressured



2x2 Mesh

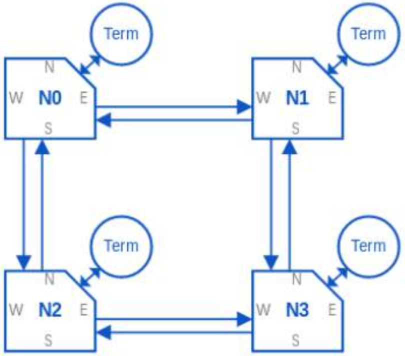


2x2 Torus

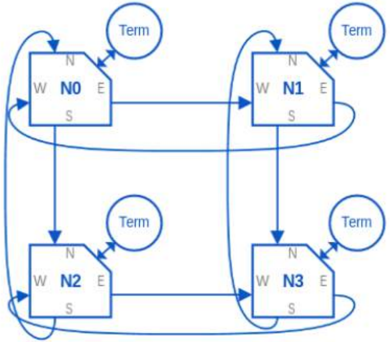
axis_ctrl_crossbar_2d_mesh

Microarchitecture

This crossbar has been optimized for low-throughput control and can scale to a large number of ports. The underlying implementation uses a 2-dimensional mesh topology which can be configured either as a bidirectional mesh or a unidirectional torus. The crossbar is not deadlock free by design but there are various features implemented that reduce the possibility of deadlock. In the event of a deadlock the crossbar will self-recover by dropping all the packets in flight. This behaviour makes the crossbar lossy. The underlying mesh topology can only be a square so the number of ports supported are N^2 for N > 1. All unused ports need to be terminated using the axis_port_terminator module.



2x2 Mesh



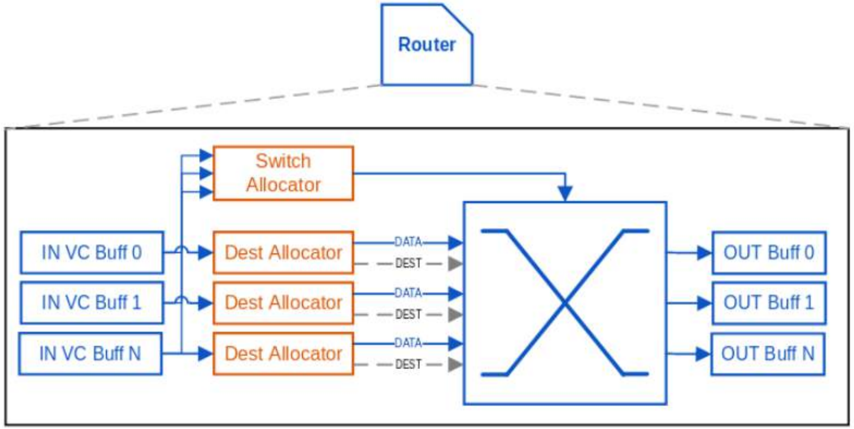
2x2 Torus

axis_ctrl_crossbar_2d_mesh

Router Architecture

Each node in the crossbar is comprised of a terminal and a router. A terminal is the interface to client logic and the network of routers performs packet switching. It consists of:

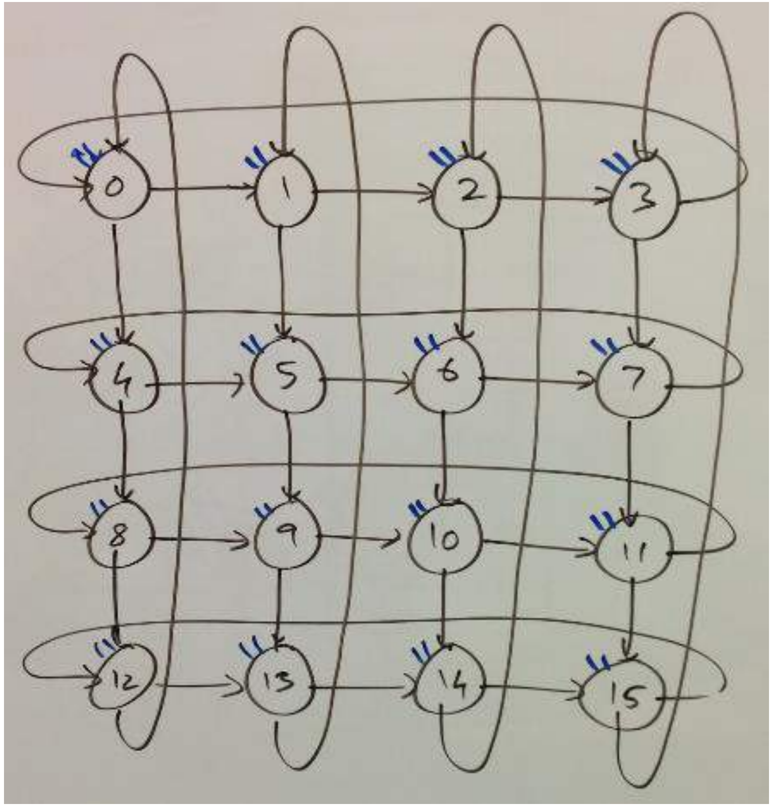
- A switch to do the routing
- Ingress buffers with one or more virtual channels
- A switch allocator to choose an input port to drive the switch
- A destination selector to choose the destination port
- Egress buffers



axis_ctrl_crossbar_2d_mesh

Scaling

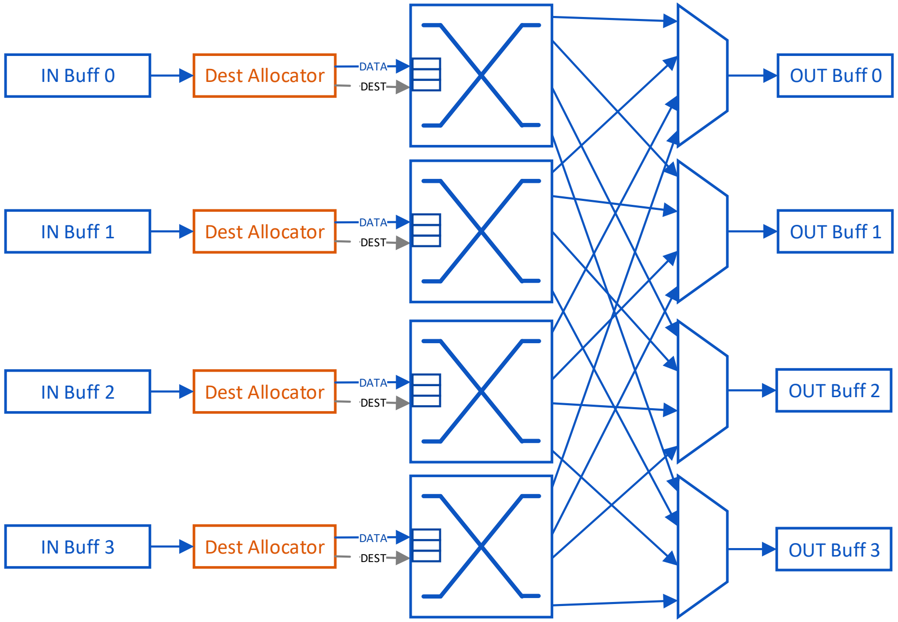
The mesh can scale to an arbitrary number of ports.



chdr_crossbar_nxn

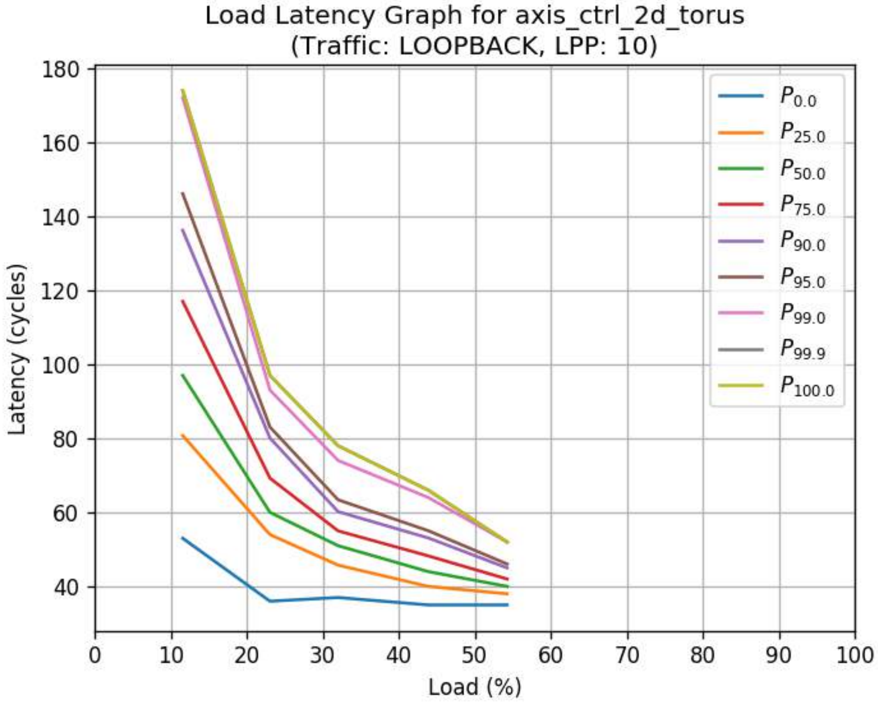
- **Topology**: Trivial. Single router.
- **Routing**: Store-and-Fwd
  - Store-and-Fwd: A packet is completely buffered in one router before moving to the next one.
- **Flow Control**: Packet buffer with cut-through
  - Packet buffer: Entire packet is buffered in router
  - Cut-through: Only flits (words) are backpressured

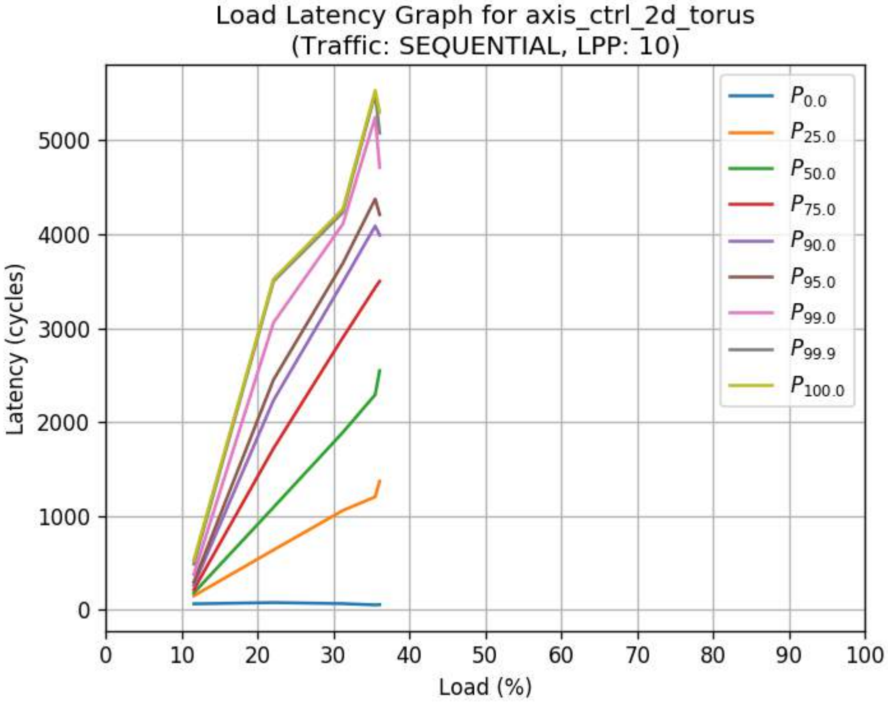
Alternative for axi_crossbar



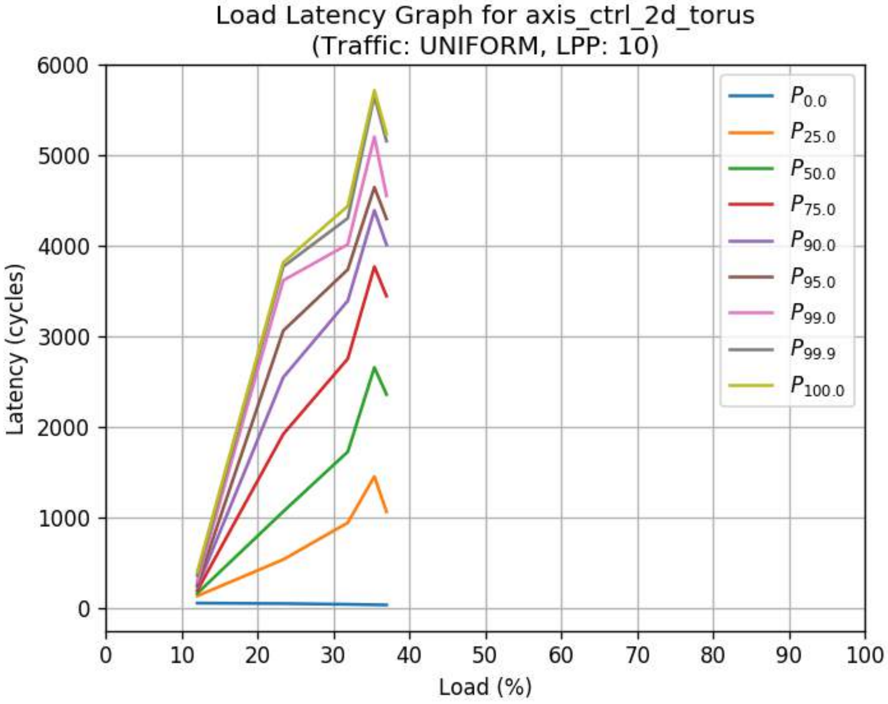
Simulation Data

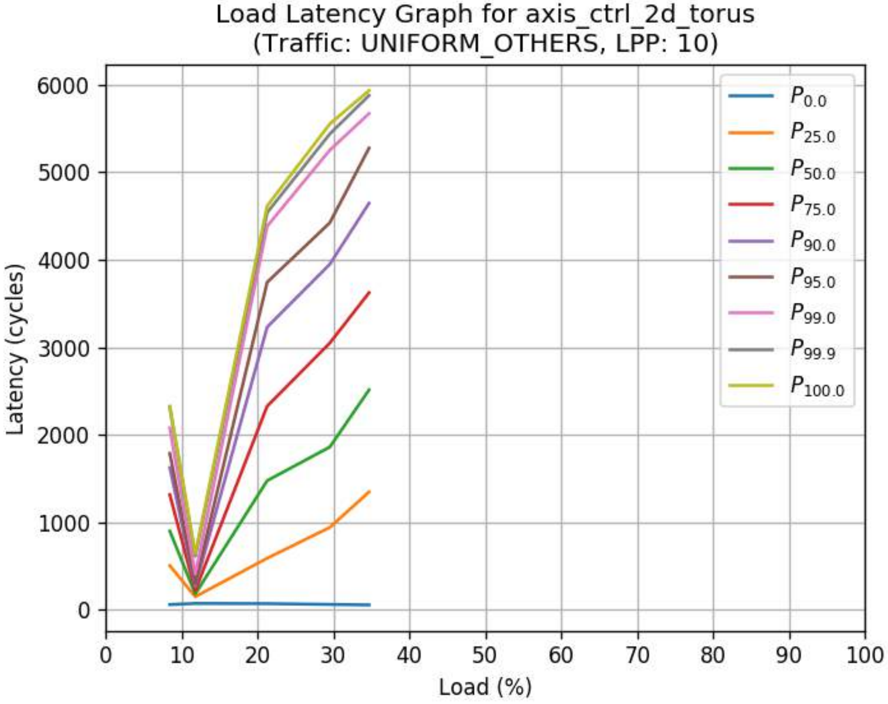
Load vs Latency: **axis_ctrl_crossbar_2d_mesh**
(TORUS, 25 nodes, 25 traffic generators)



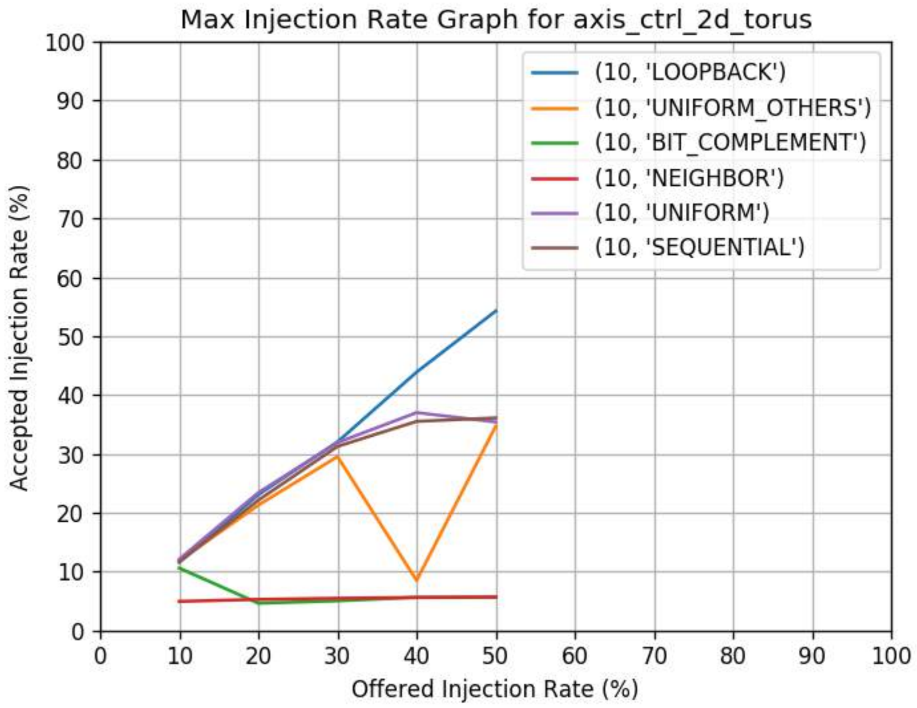


Load vs Latency: **axis_ctrl_crossbar_2d_mesh** (TORUS, 25 nodes, 25 traffic generators)

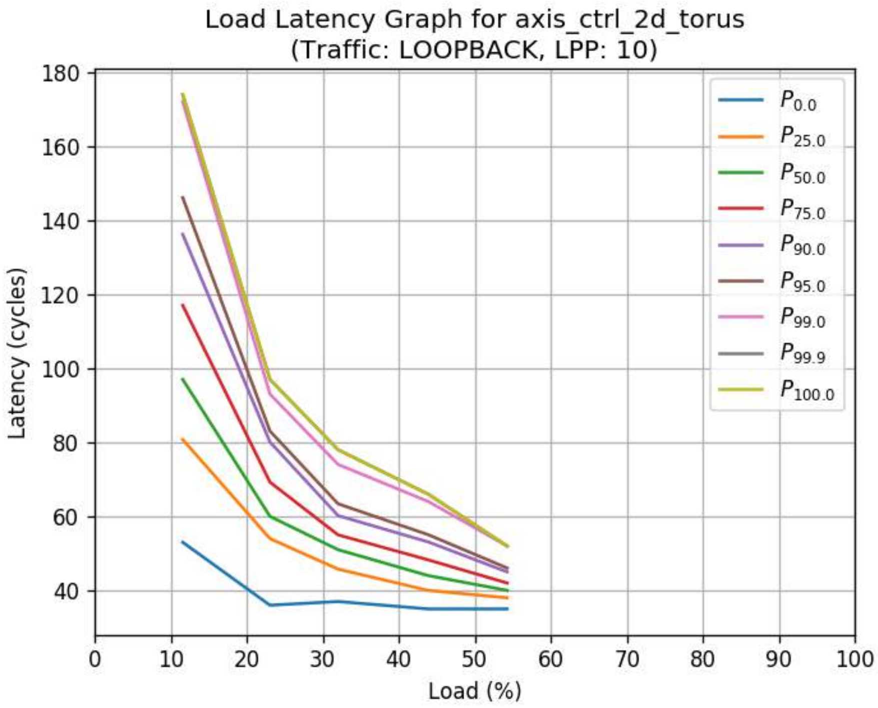


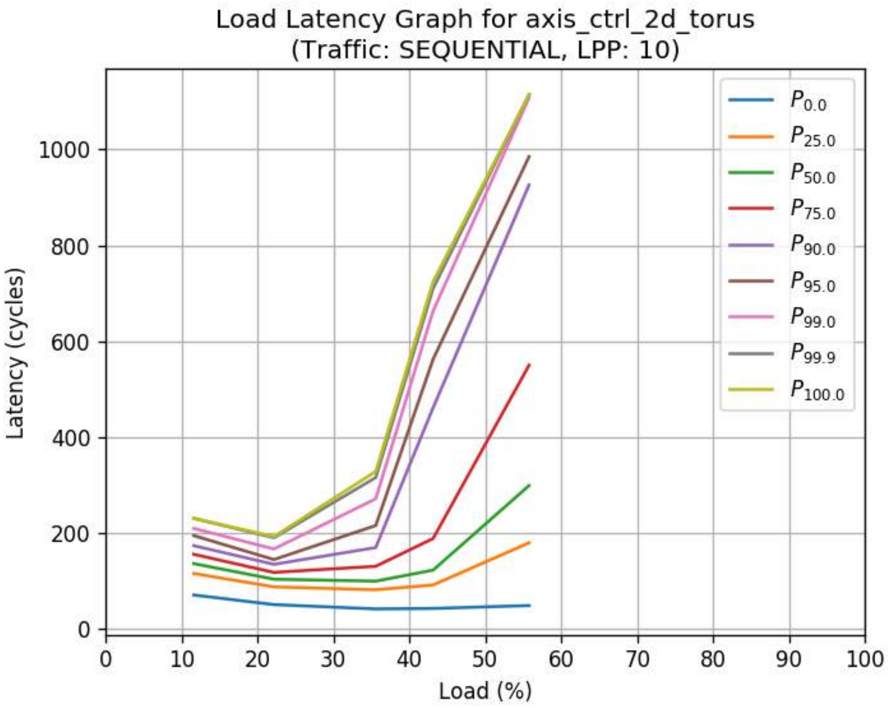


Load vs Latency: **axis_ctrl_crossbar_2d_mesh**
(TORUS, 25 nodes, 25 traffic generators)

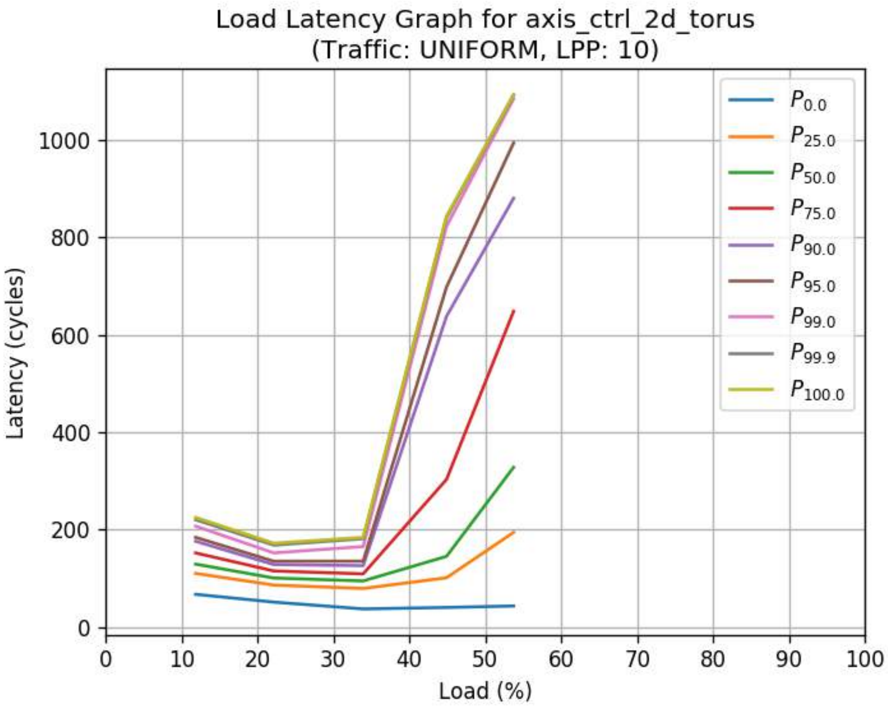


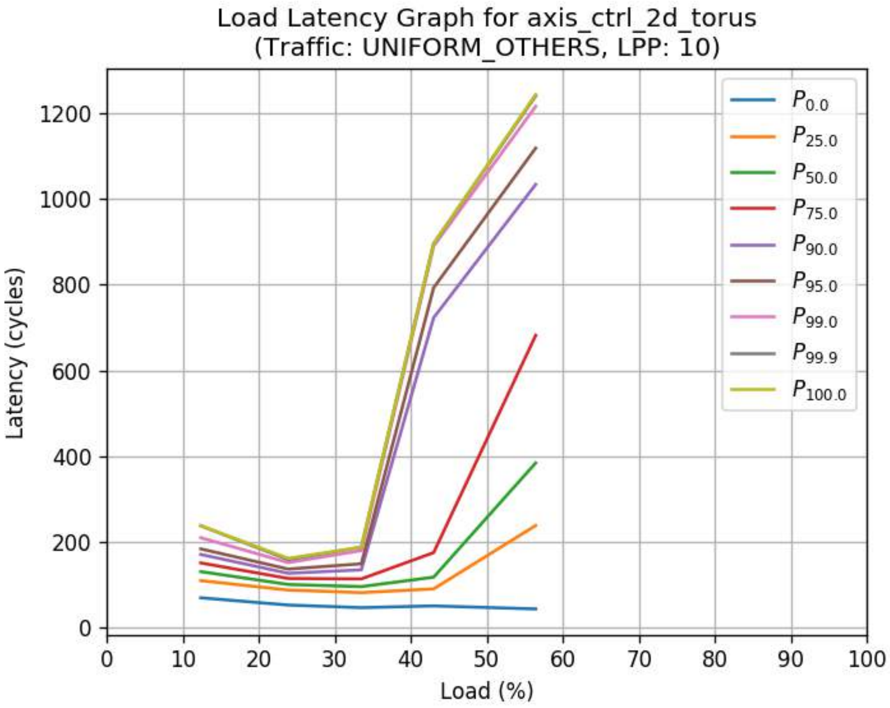
Load vs Latency: **axis_ctrl_crossbar_2d_mesh**
(TORUS, 25 nodes, 4 traffic generators)





Load vs Latency: **axis_ctrl_crossbar_2d_mesh**
(TORUS, 25 nodes, 4 traffic generators)



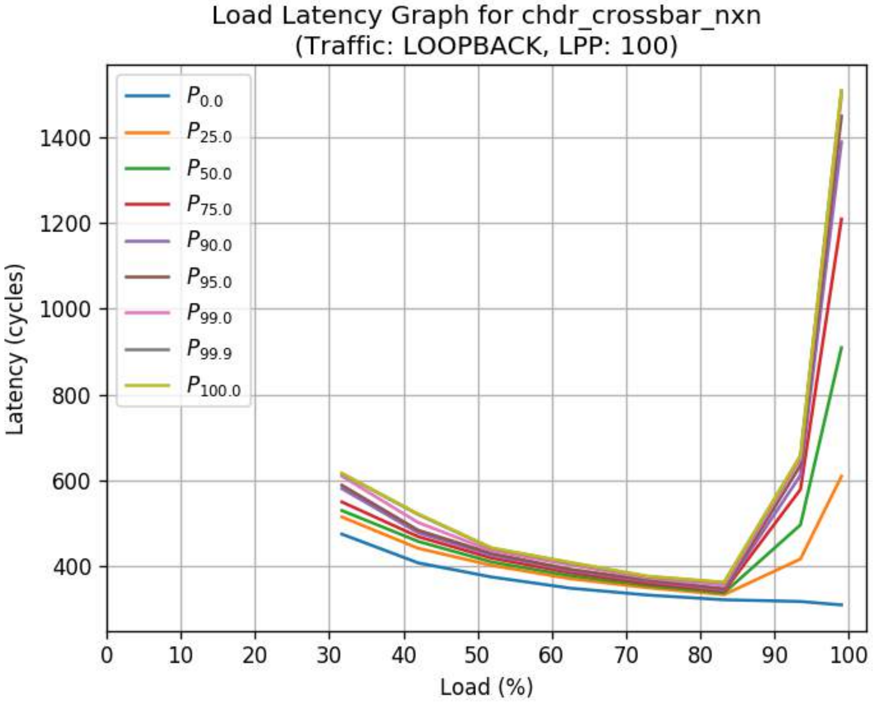


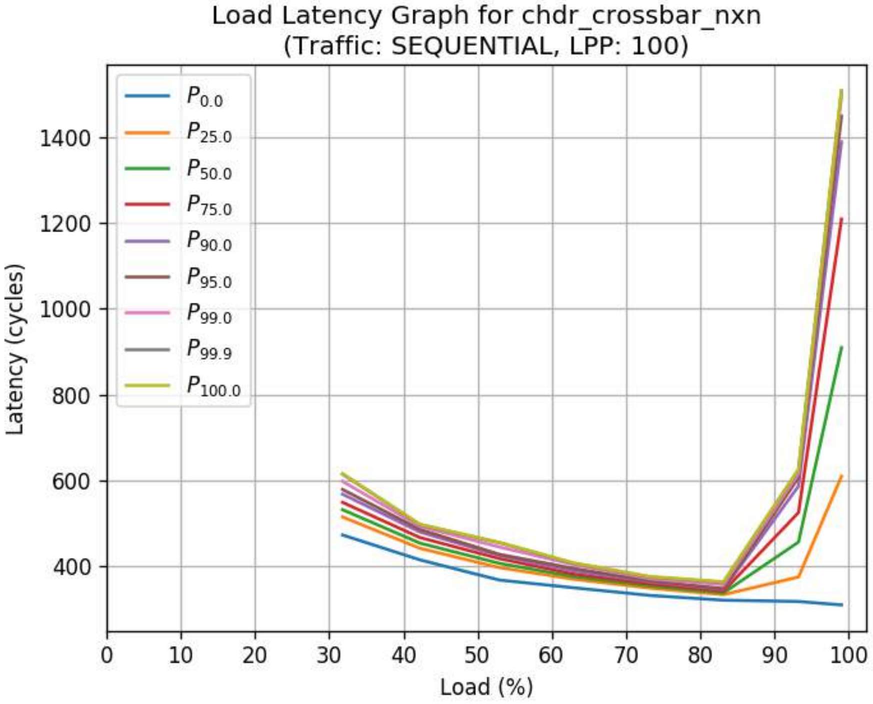
Load vs Latency: **axis_ctrl_crossbar_2d_mesh** (TORUS, 25 nodes, 4 traffic generators)

FPGA Utilization: **axis_ctrl_crossbar_2d_mesh**

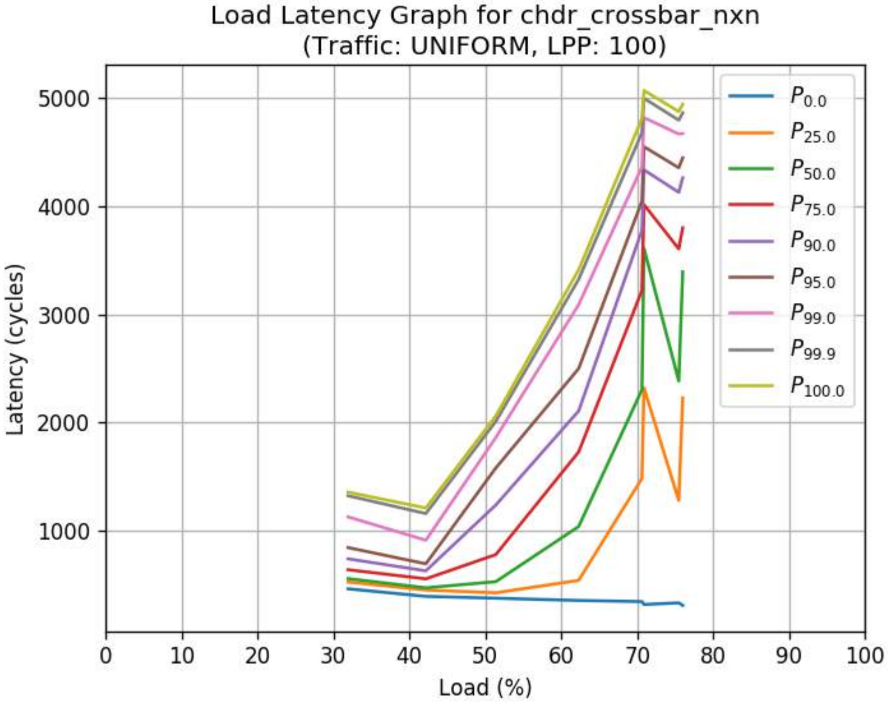
Topology	Ports	Width	LUTs (% K7 410T)	FFs (% K7 410T)	BRAM (% K7 410T)	Fmax (MHz)	MTU (Bytes)
Torus	16	32	3.6%	1.7%	0.0%	375	128
Torus	25	32	5.6%	2.7%	0.0%	375	128
Torus	36	32	9.1%	3.9%	0.0%	250	128
Mesh	16	32	4.5%	2.2%	0.0%	305	128
Mesh	25	32	7.4%	3.6%	0.0%	275	128
Mesh	36	32	12.8%	5.3%	0.0%	210	128

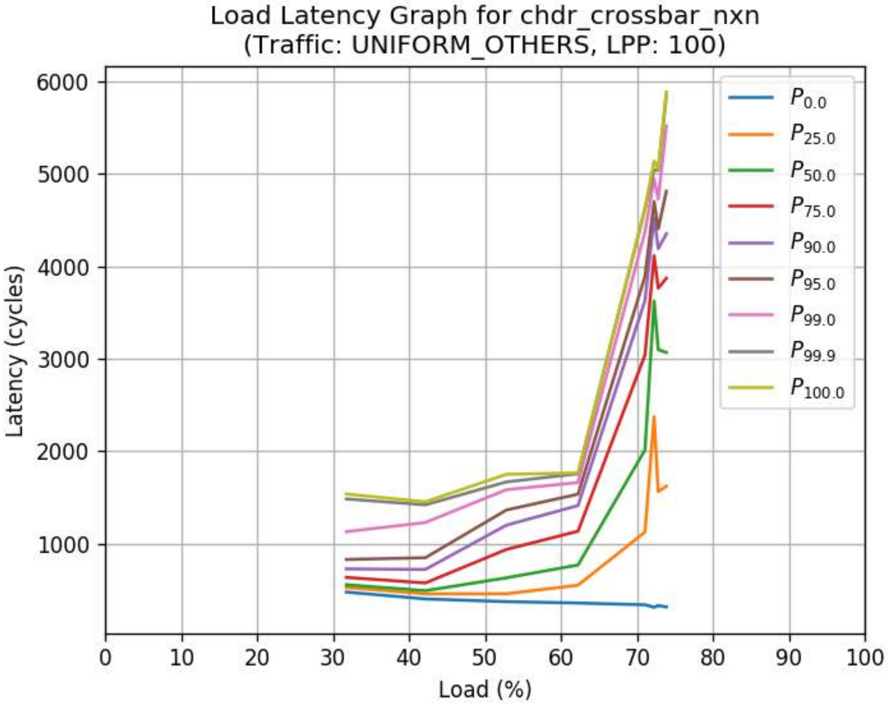
Load vs Latency: **chdr_crossbar_nxn**
(12 nodes, 12 traffic generators)



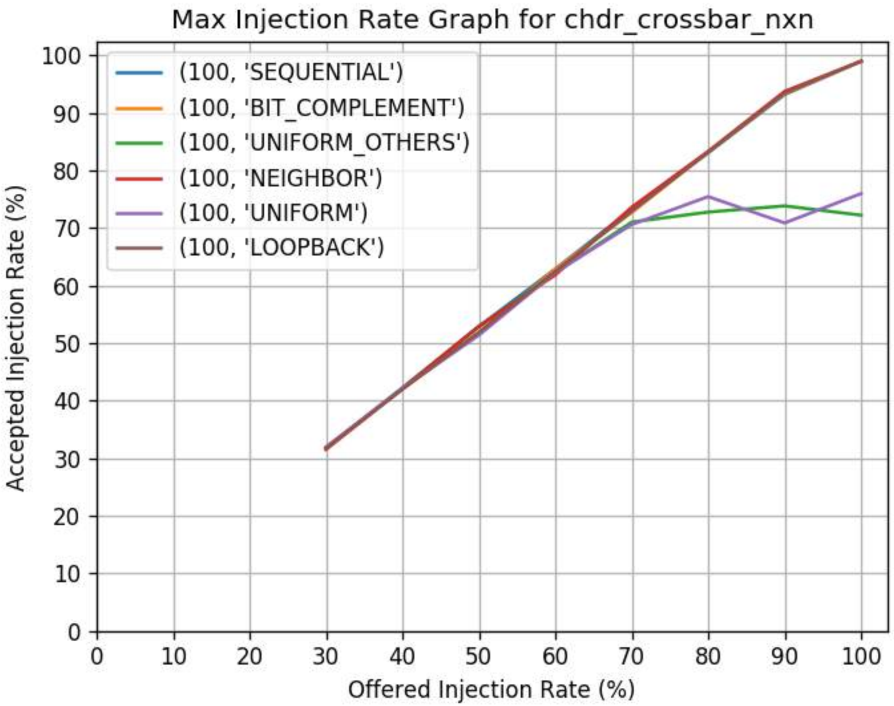


Load vs Latency: **chdr_crossbar_nxn**
(12 nodes, 12 traffic generators)





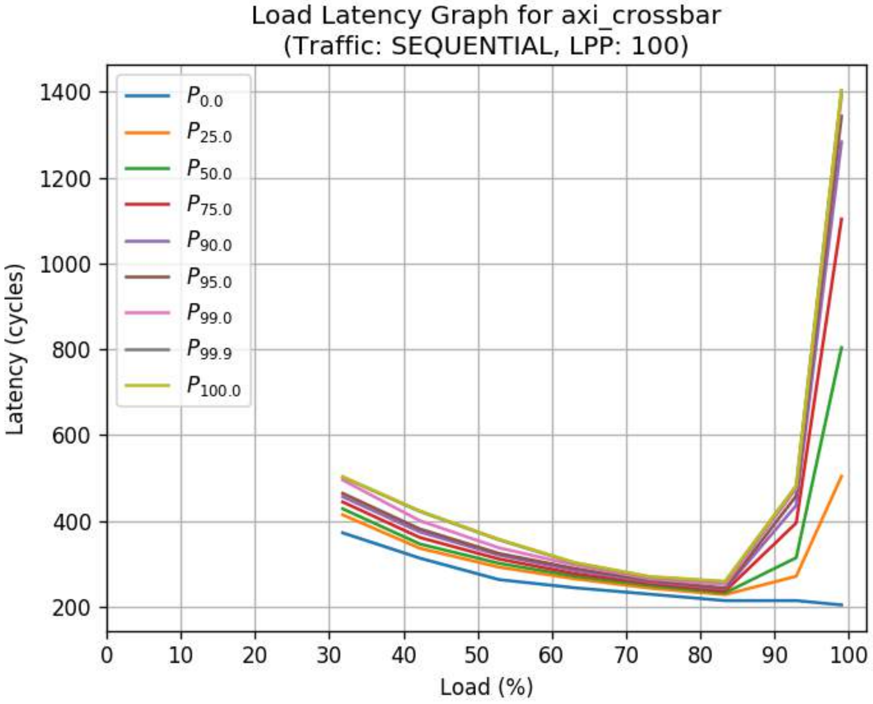
Load vs Latency: **chdr_crossbar_nxn**
(12 nodes, 12 traffic generators)

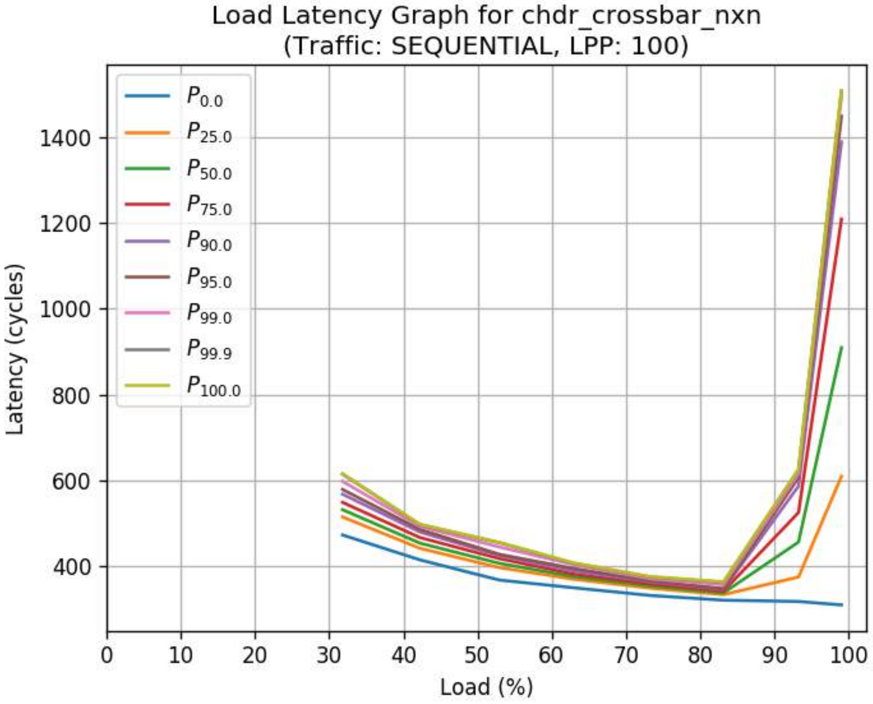


FPGA Utilization: **chdr_crossbar_nxn**

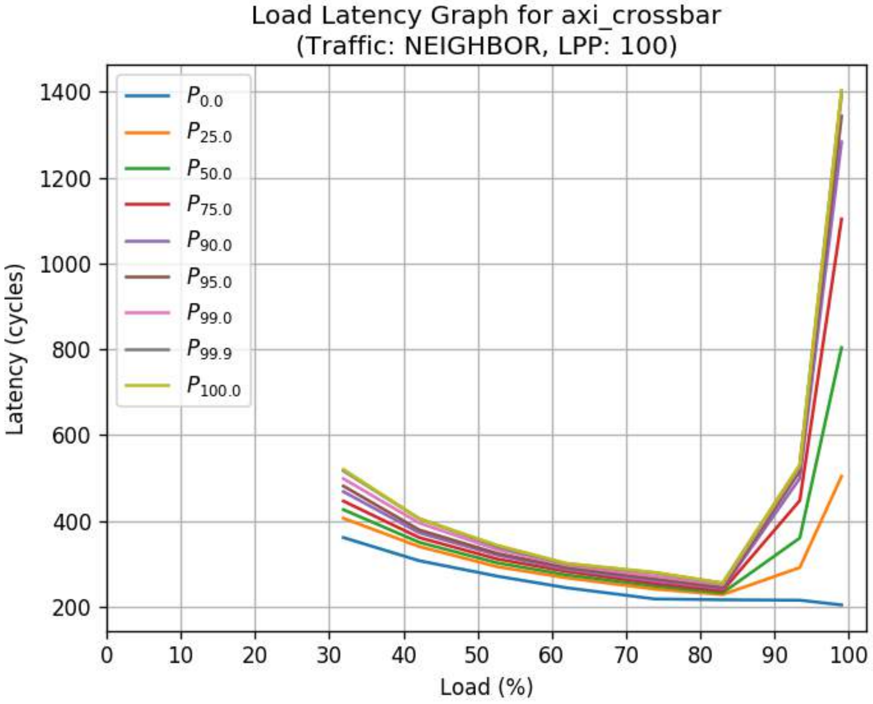
Optimize	Ports	Width	LUTs (% K7 410T)	FFs (% K7 410T)	BRAM (% K7 410T)	Fmax (MHz)	MTU (Bytes)
Area	6	64	2.2%	2.1%	1.5%	370	8192
Area	8	64	3.0%	3.2%	2.0%	370	8192
Area	10	64	4.6%	4.5%	2.5%	340	8192
Area	12	64	6.0%	6.1%	3.0%	340	8192
Area	6	256	6.7%	7.8%	3.0%	360	16384
Area	8	256	9.8%	12.0%	4.0%	360	16384
Area	10	256	15.0%	17.0%	5.0%	330	16384
Area	12	256	19.6%	22.9%	6.0%	340	16384
Performance	6	64	2.2%	2.0%	1.5%	360	8192
Performance	8	64	3.1%	3.0%	2.0%	330	8192
Performance	10	64	4.6%	4.3%	2.5%	280	8192
Performance	12	64	6.2%	5.8%	3.0%	260	8192
Performance	6	256	6.5%	7.2%	3.0%	360	16384
Performance	8	256	9.6%	11.2%	4.0%	300	16384
Performance	10	256	14.3%	16.0%	5.0%	240	16384
Performance	12	256	19.4%	22.7%	6.0%	240	16384

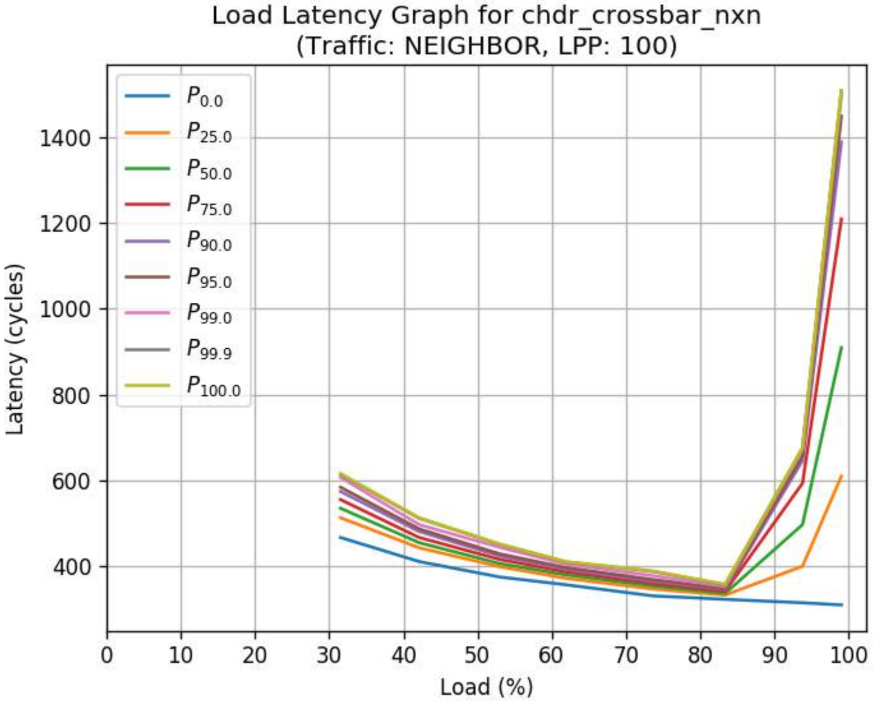
Load vs Latency: **axi_crossbar** vs **chdr_crossbar_nxn** (12 nodes, 12 traffic generators)



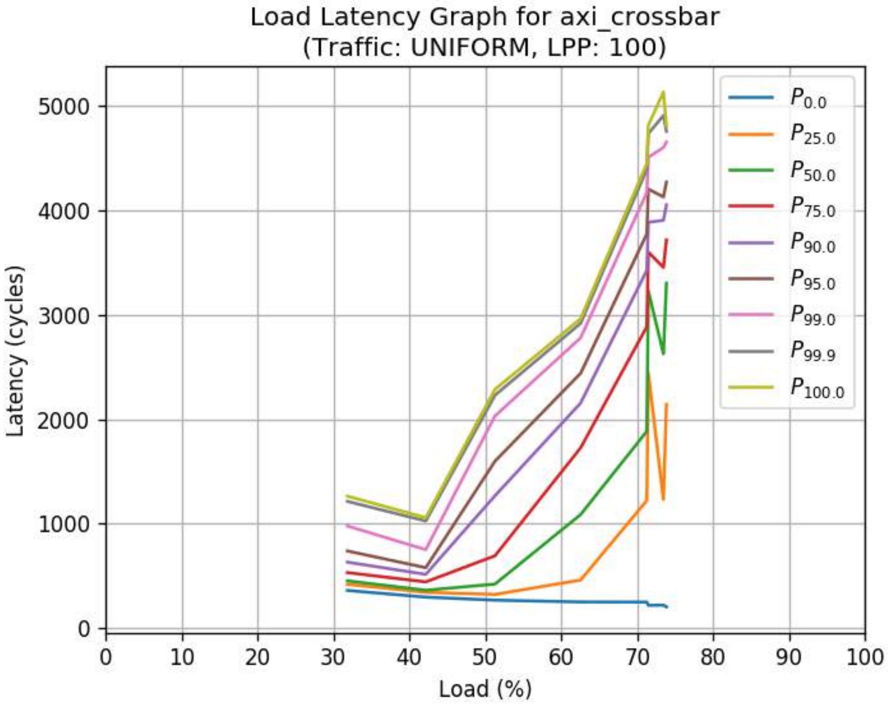


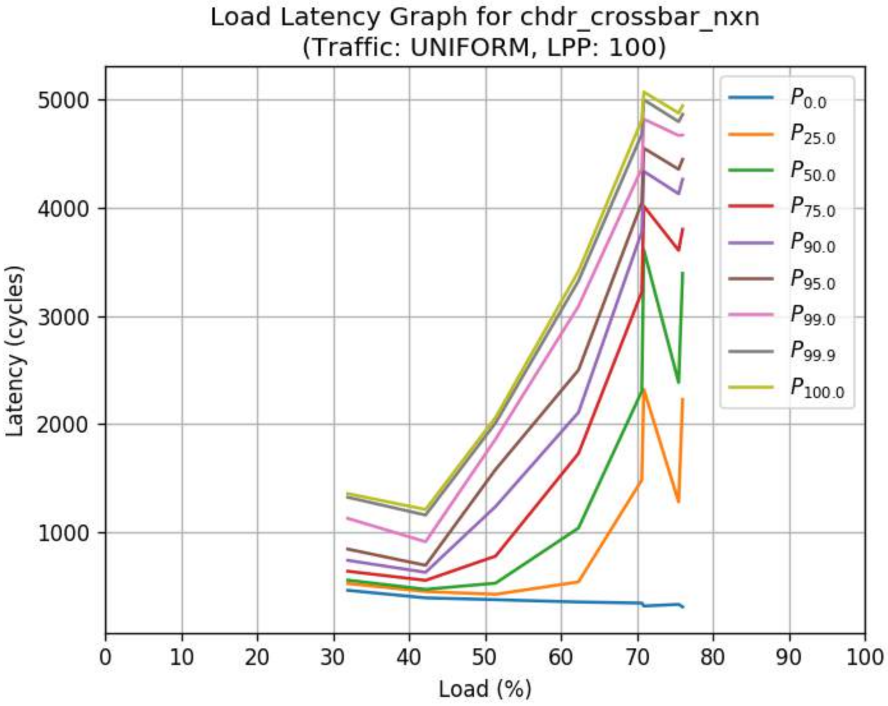
Load vs Latency: **axi_crossbar** vs **chdr_crossbar_nxn** (12 nodes, 12 traffic generators)





Load vs Latency: **axi_crossbar** vs **chdr_crossbar_nxn** (12 nodes, 12 traffic generators)





Load vs Latency: **axi_crossbar** vs **chdr_crossbar_nxn** (12 nodes, 12 traffic generators)