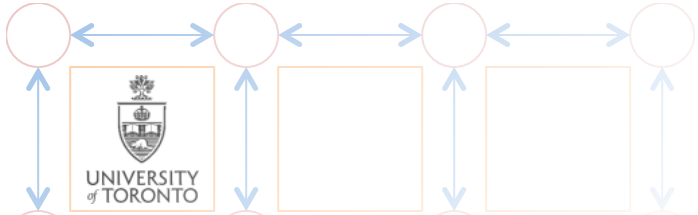


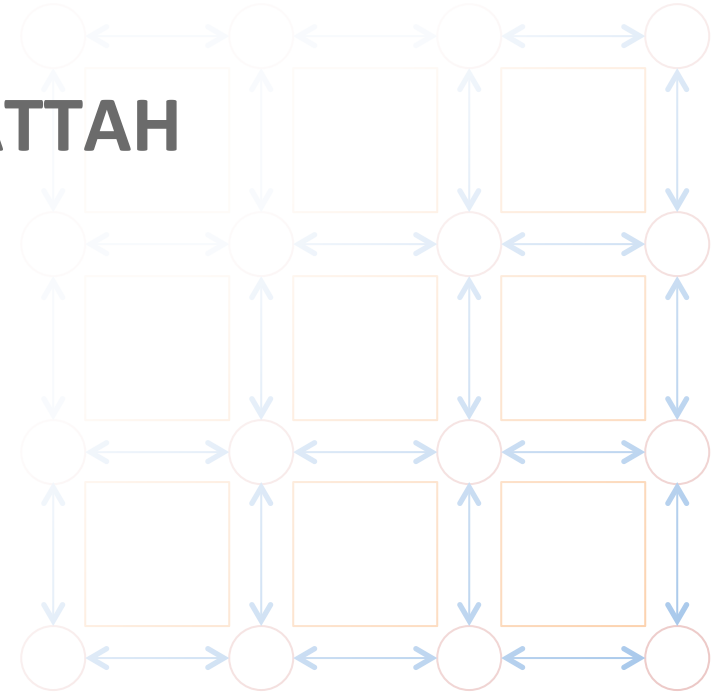
Take the Highway:

Design for Embedded NoCs on FPGAs

Mohamed **ABDELFATTAH**

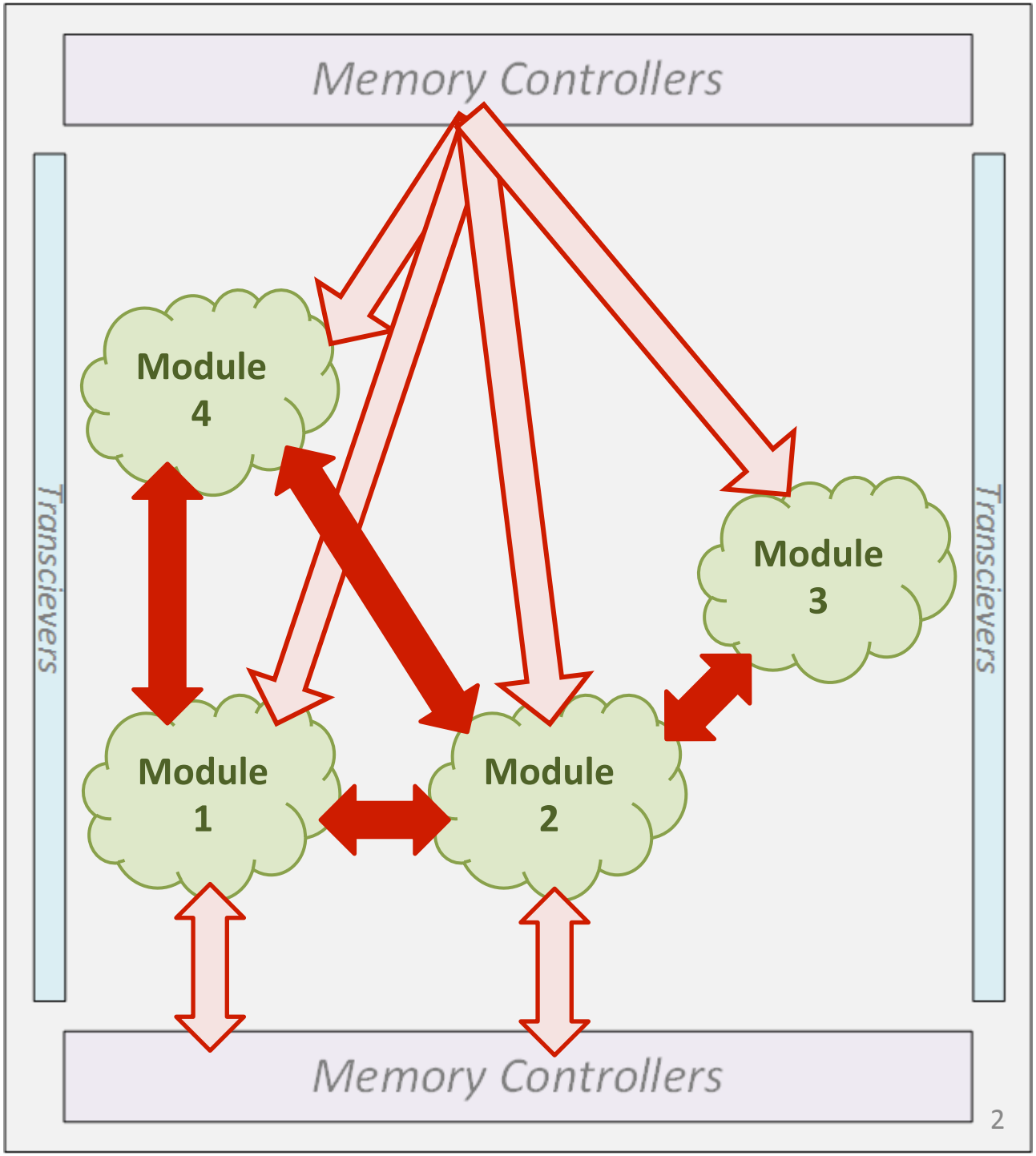
Andrew **BITAR**

Vaughn **BETZ**



Motivation

- FPGAs are big!
- Design big systems
- High on-chip communication



Motivation

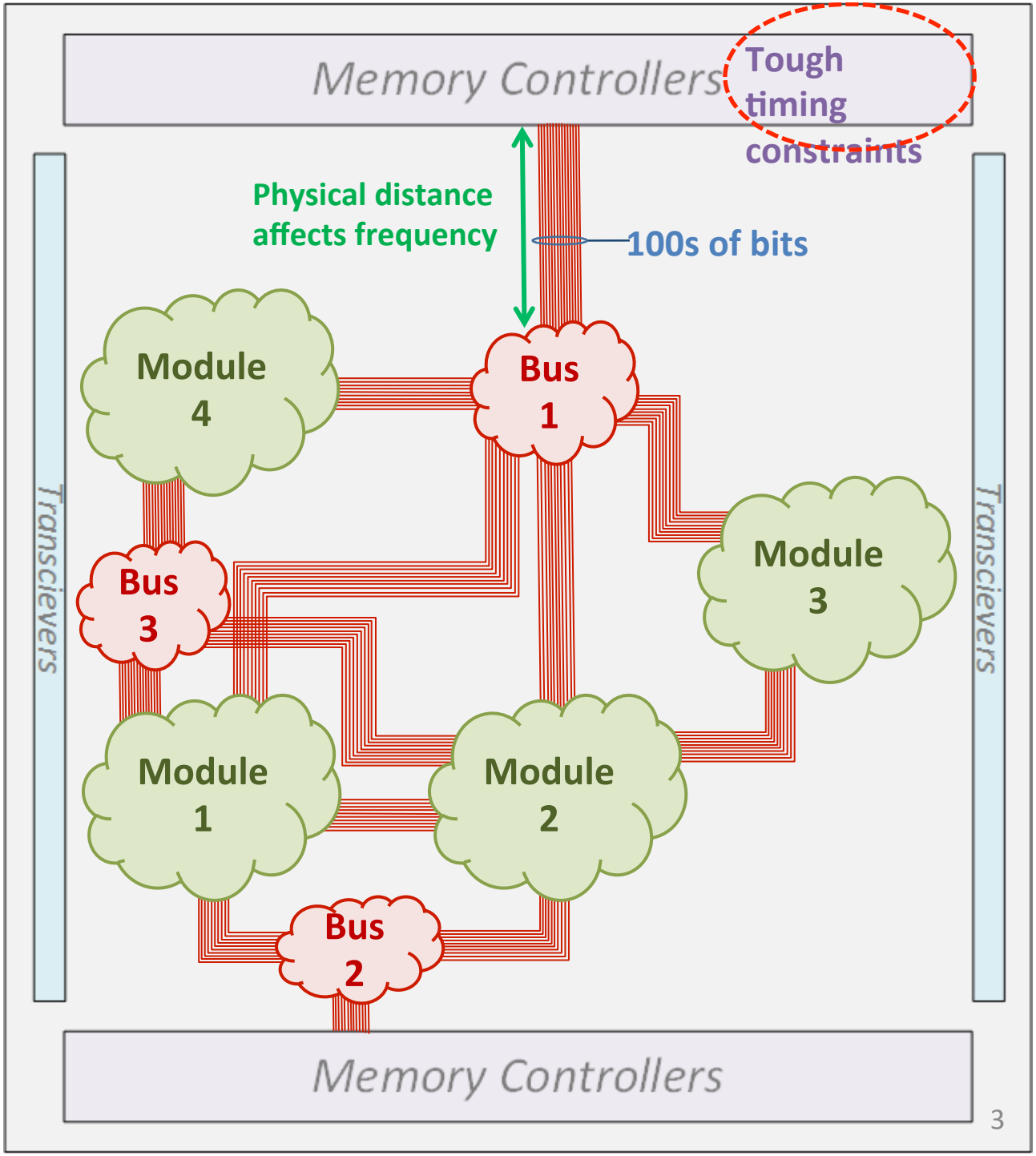
FPGAs are big!

Design big systems

High on-chip communication

Design a bus for each set of connections

- Wide links from single bit-programmable interconnect
- *Somewhat* unpredictable frequency → re-pipeline
- Buses are *unique* to the application, and.. inefficient



Motivation

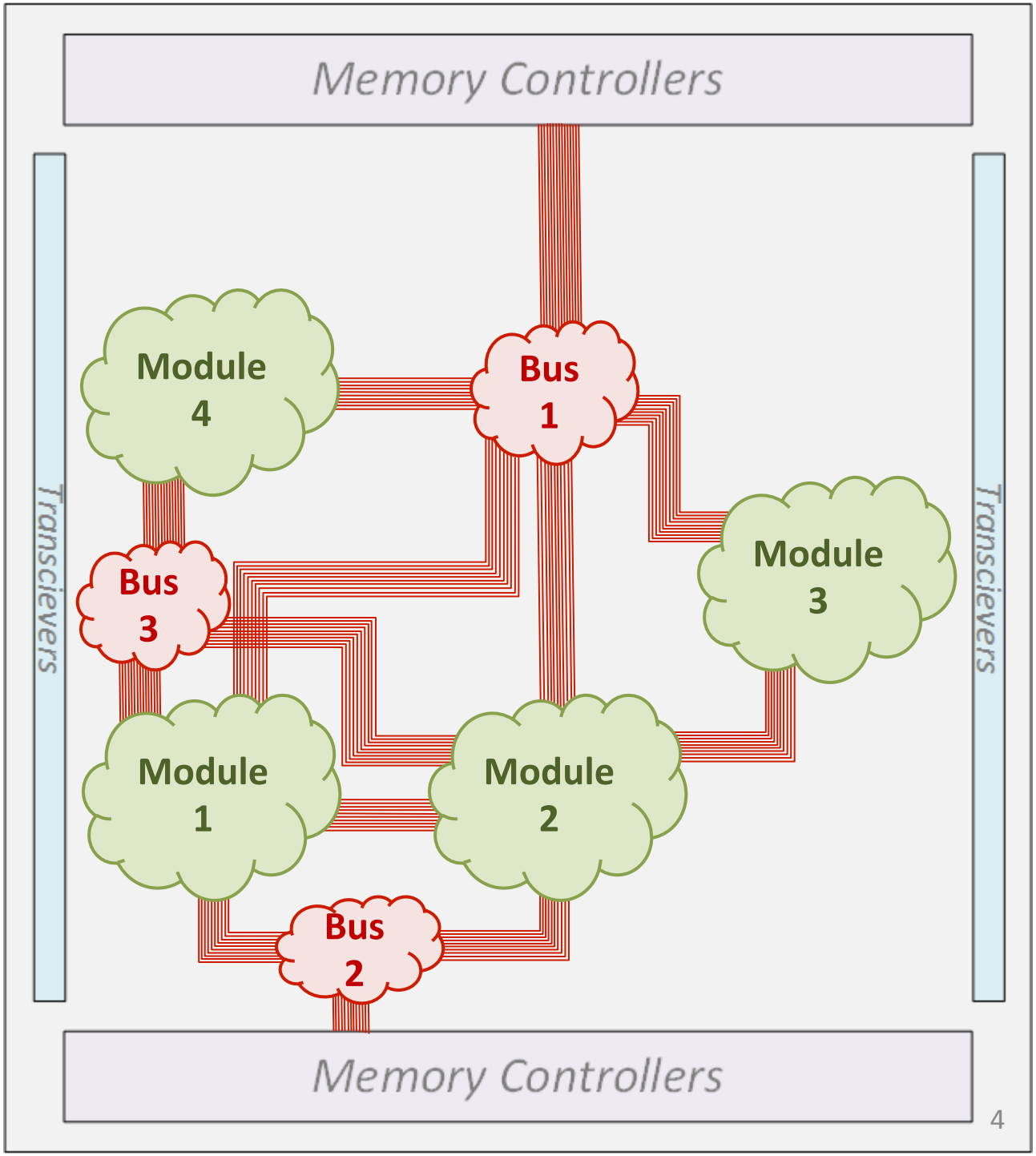
FPGAs are big!

Design big systems

High on-chip communication

Design a bus for each set of connections

- Wide links from single bit-programmable interconnect
- *Somewhat* unpredictable frequency → re-pipeline
- Buses are *unique* to the application, and.. inefficient



Motivation

FPGAs are big!

Design big systems

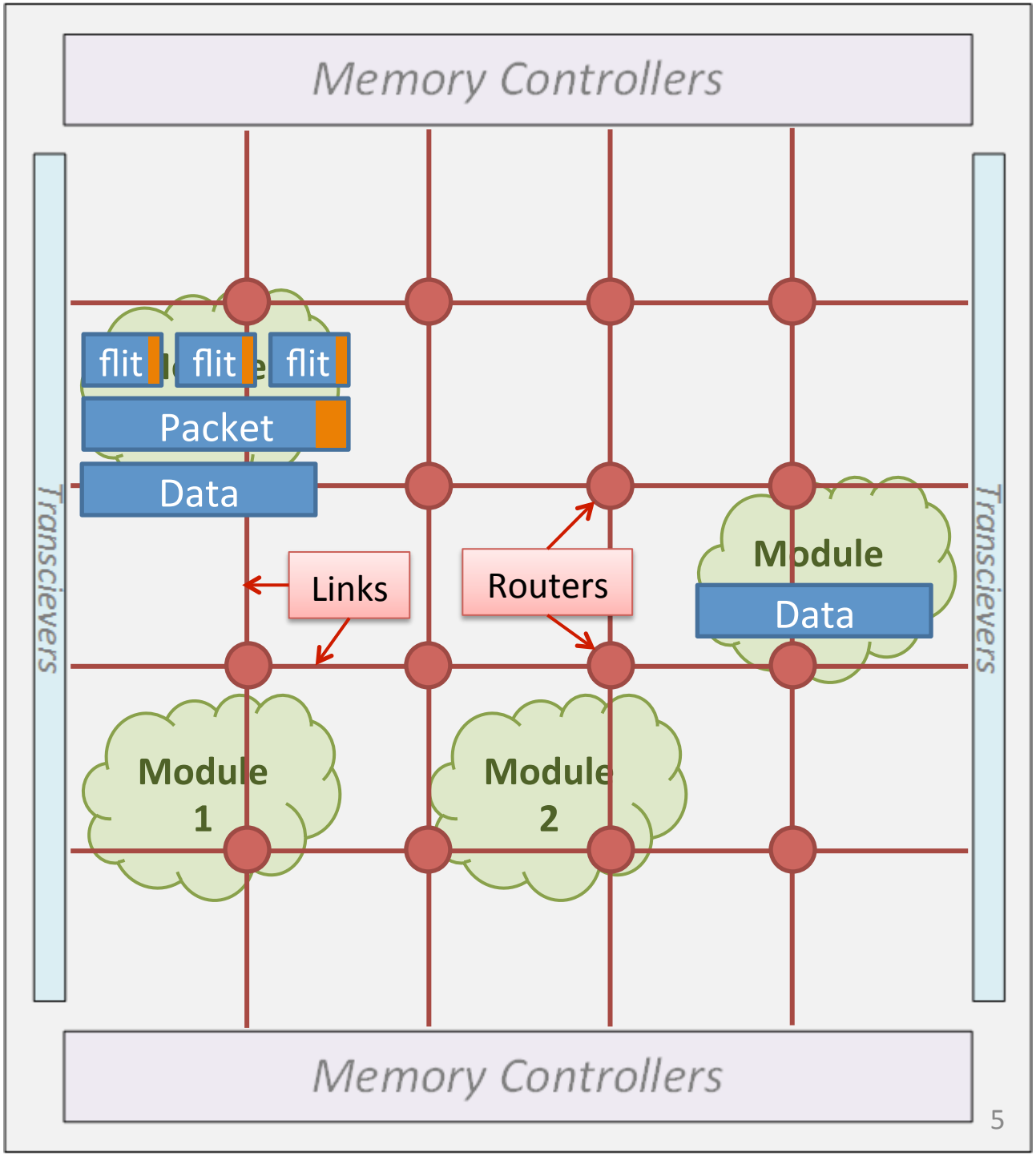
High on-chip communication

Embedded NoC

Sys-level interconnect used in most designs →**harden!**

More efficient than soft buses and huge bandwidth

NoC=*complete* interconnect
- Data transport
- Switching
- Buffering

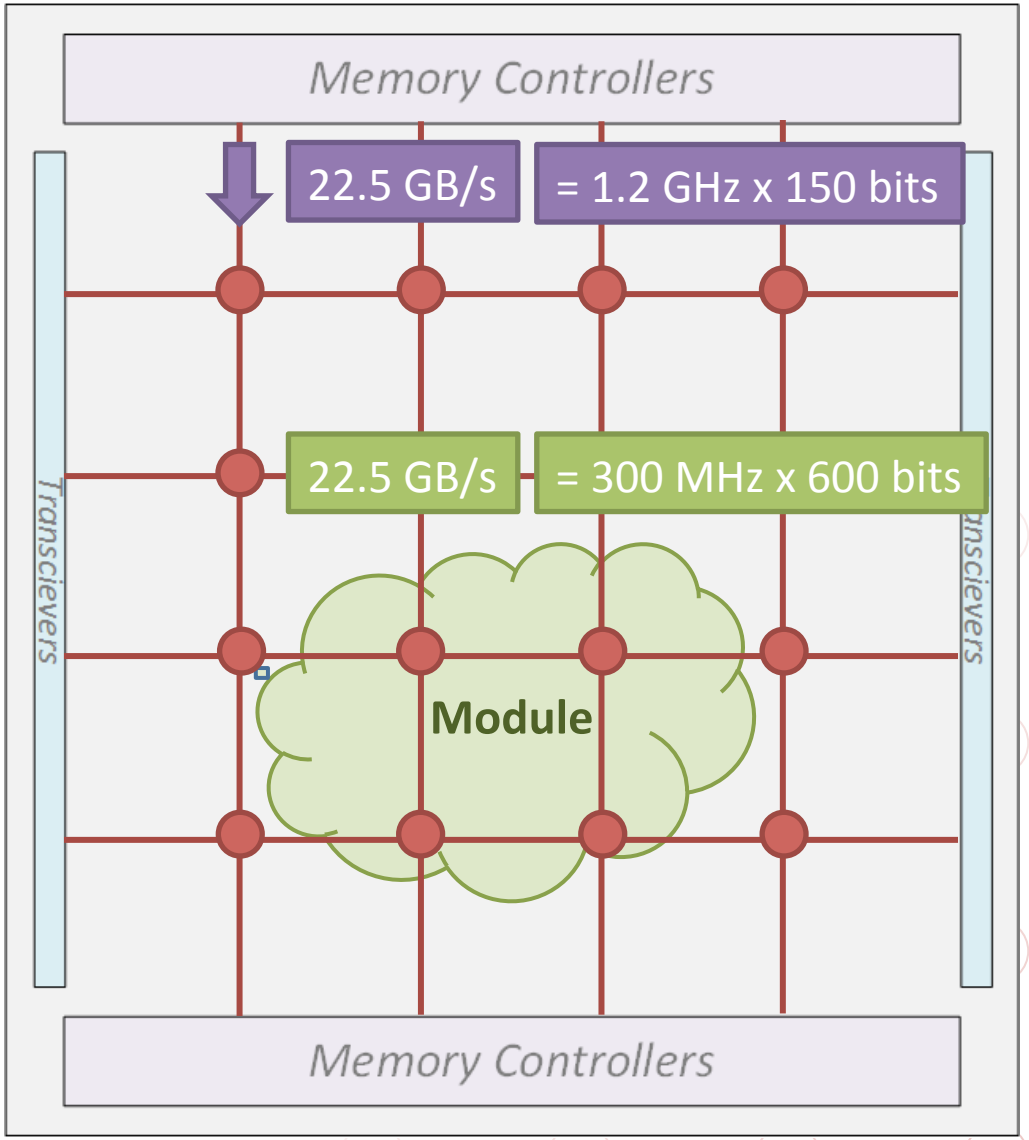


NoC Architecture @ 28nm

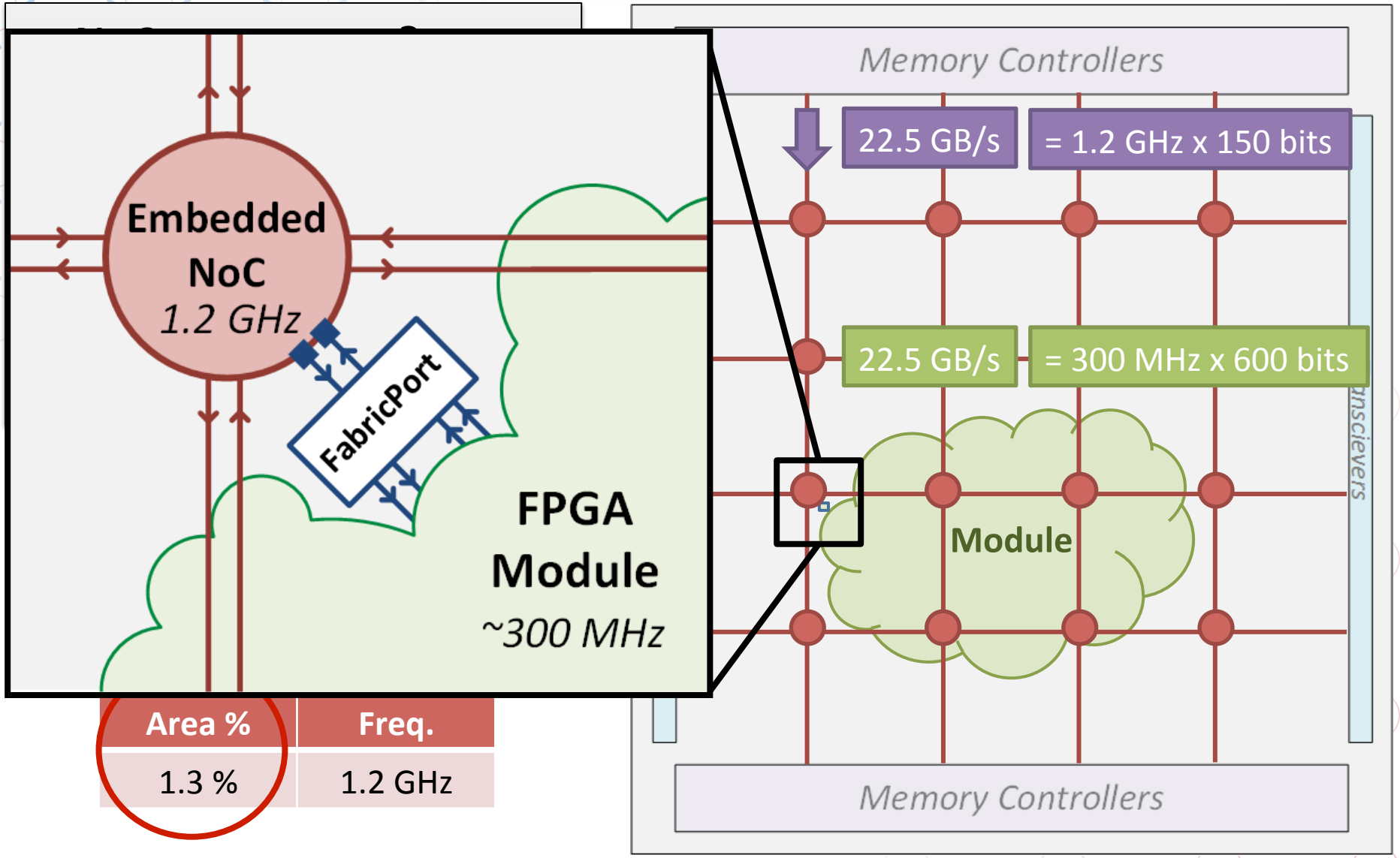
- **NoC parameters?**
- No segmentation of data
  - DDR3 max = 17 GB/s
- Run NoC at ***fixed***, ***high*** frequency → 22.5 GB/s

Link Width	# Nodes
150 bits	16

Area %	Freq.
1.3 %	1.2 GHz



NoC Architecture @ 28nm



Area %	Freq.
1.3 %	1.2 GHz

Outline

How do we ***adapt*** Embedded NoCs to FPGAs?

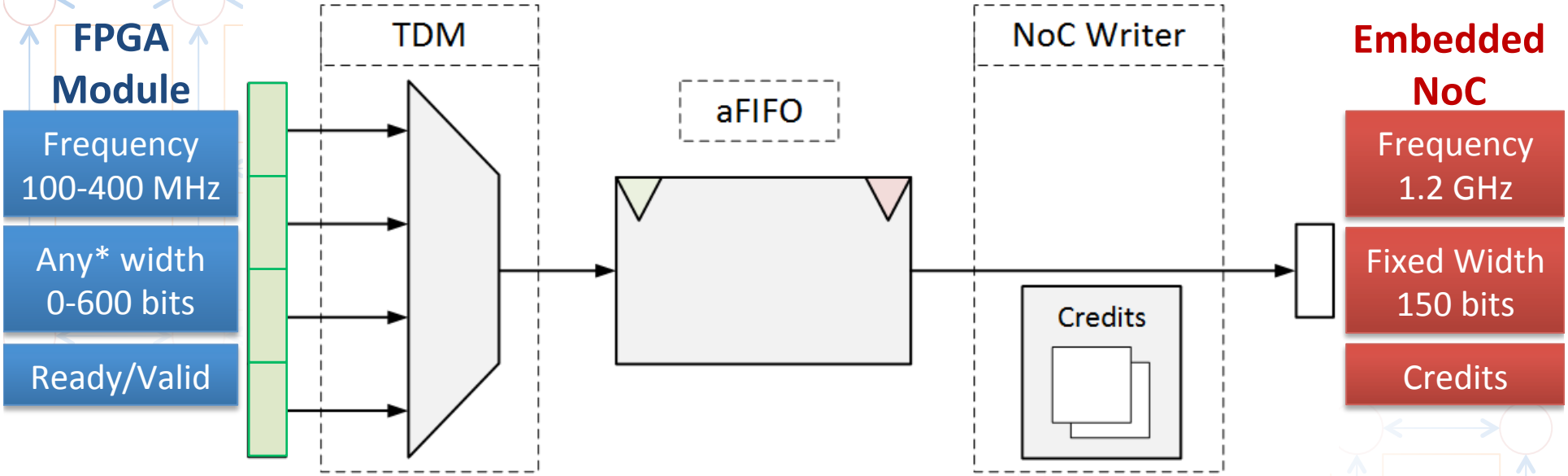
1. **FabricPort**: NoC-to-FPGA interface

2. NoC with FPGA **design styles**

3. **Case studies**: JPEG & Ethernet switch

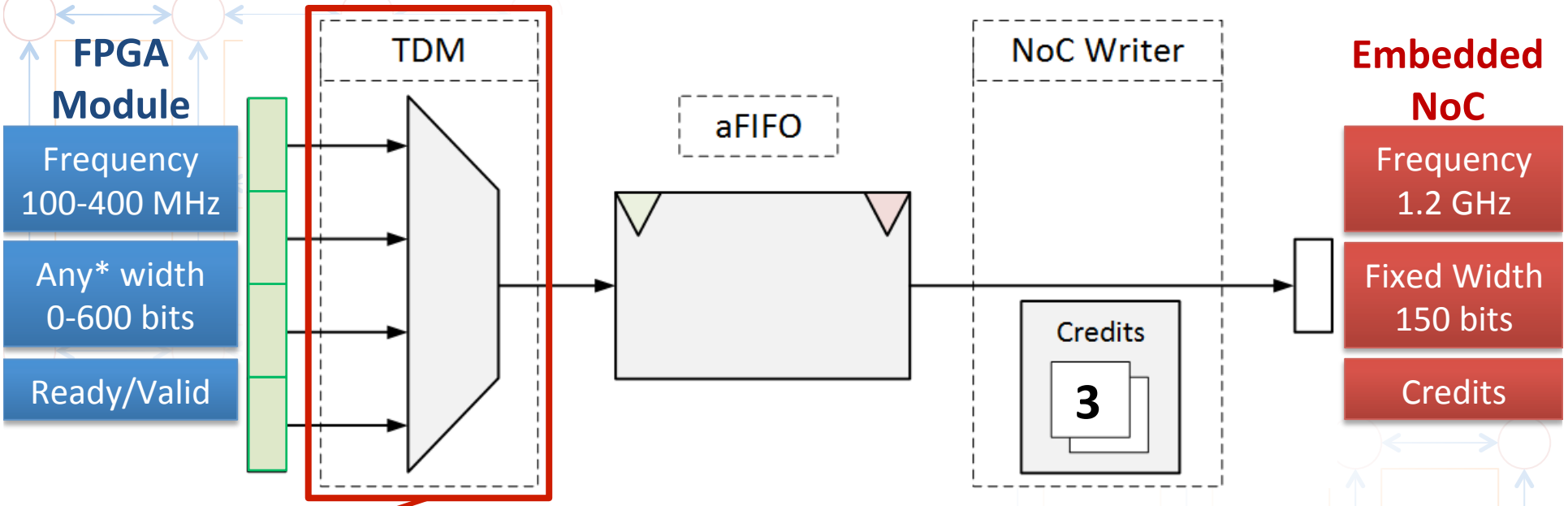

FREEWAY
ENTRANCE
NoC
Speed
Limit
∞
FabricPort = On-ramp
Photo from: http://www.aaroads.com

FabricPort In



Width	# flits
0-150 bits	1
150-300 bits	2
300-450 bits	3
450-600 bits	4

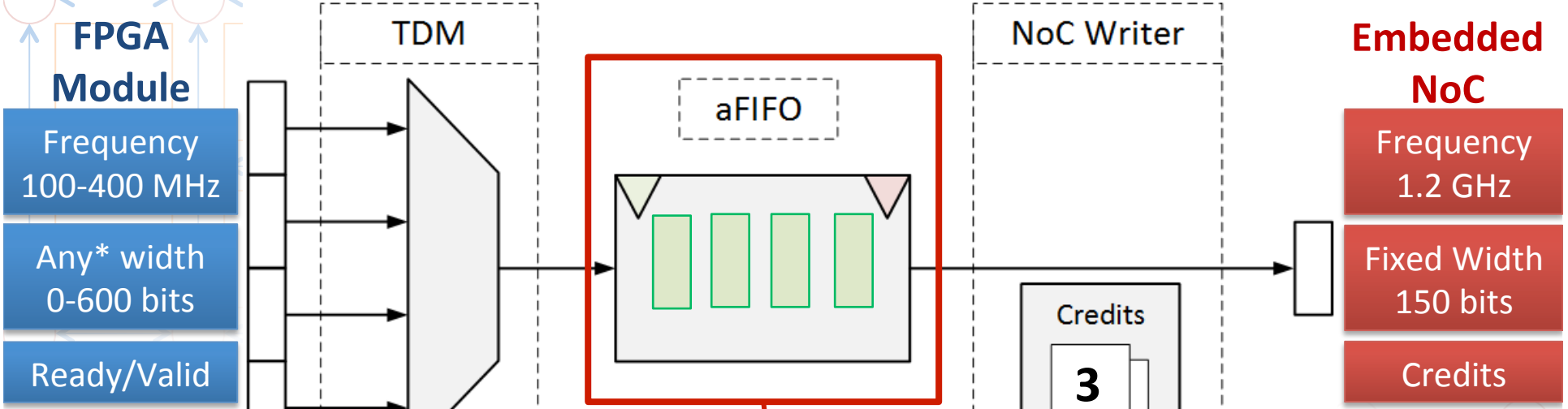
FabricPort In



Time-domain multiplexing:

- Divide width by 4
- Multiply frequency by 4

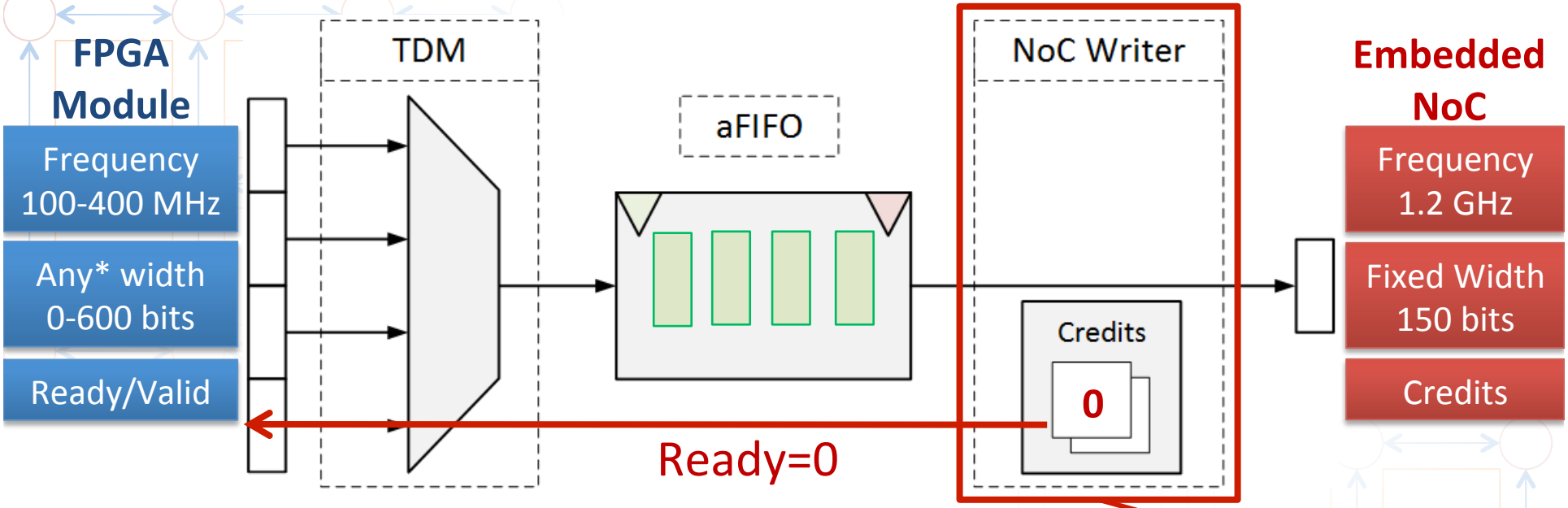
FabricPort In



Time-domain multiplexing:
- Divide width by 4
- Multiply frequency by 4

Asynchronous FIFO:
- Cross into NoC clock
- No restriction on module frequency

FabricPort In

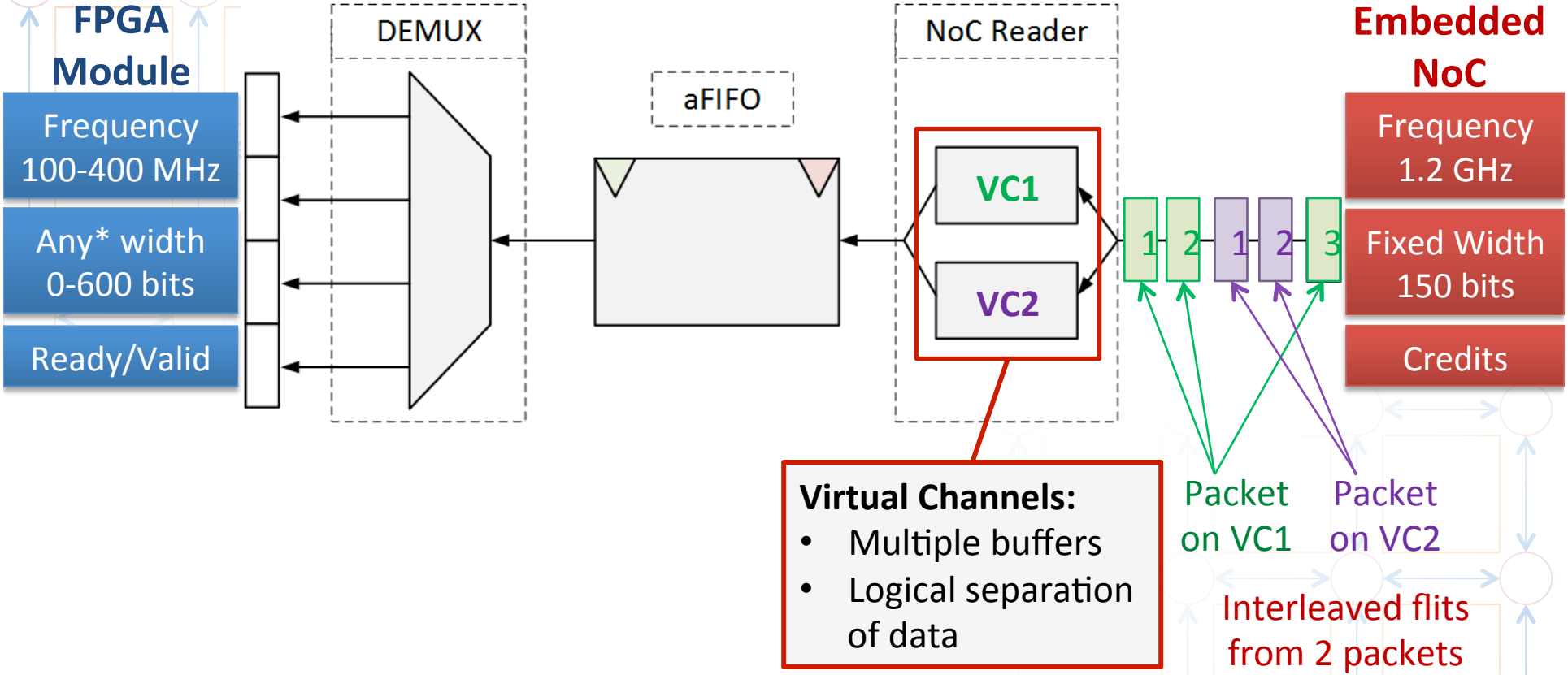


Time-domain multiplexing:
- Divide width by 4
- Multiply frequency by 4

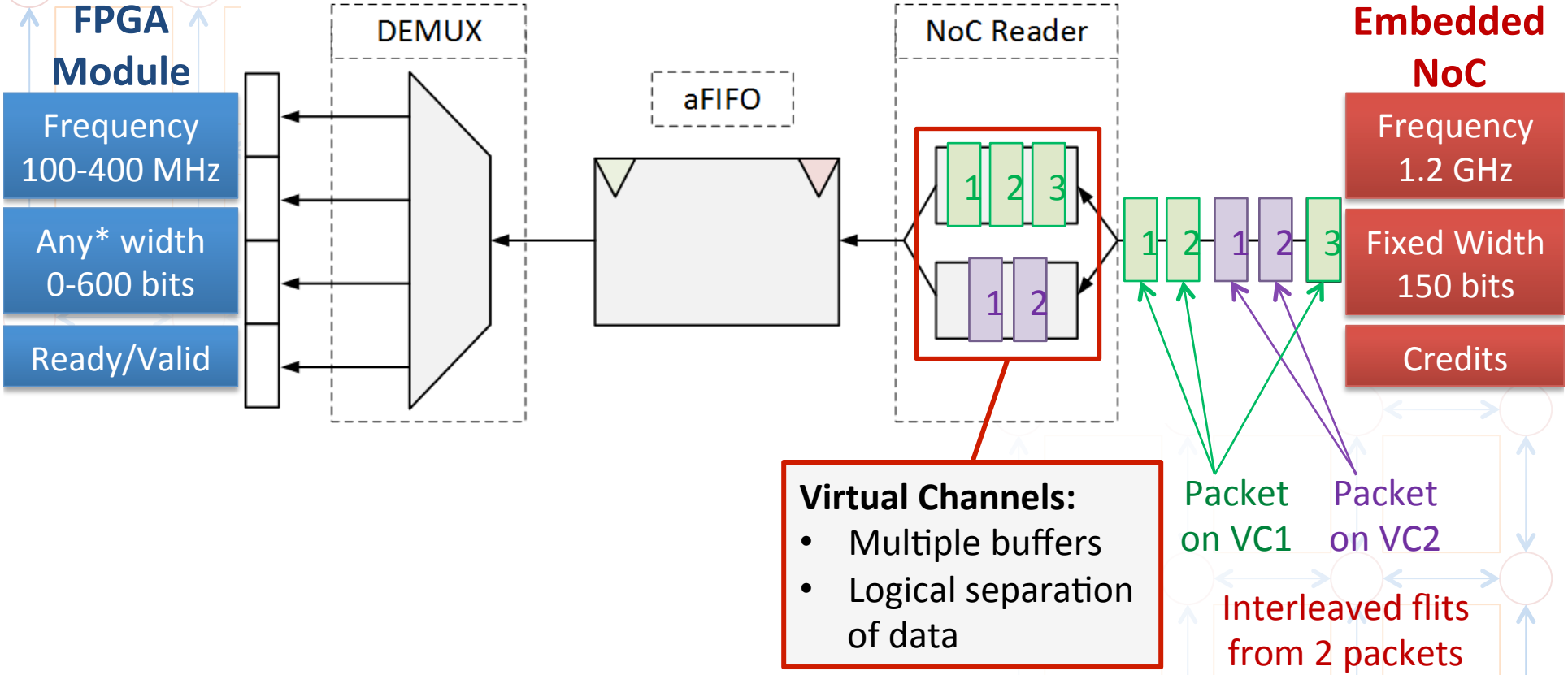
Asynchronous FIFO:
- Cross into NoC clock
- No restriction on module frequency

NoC Writer:
- Track available buffers in NoC Router
- Forward flits to NoC
- Backpressure

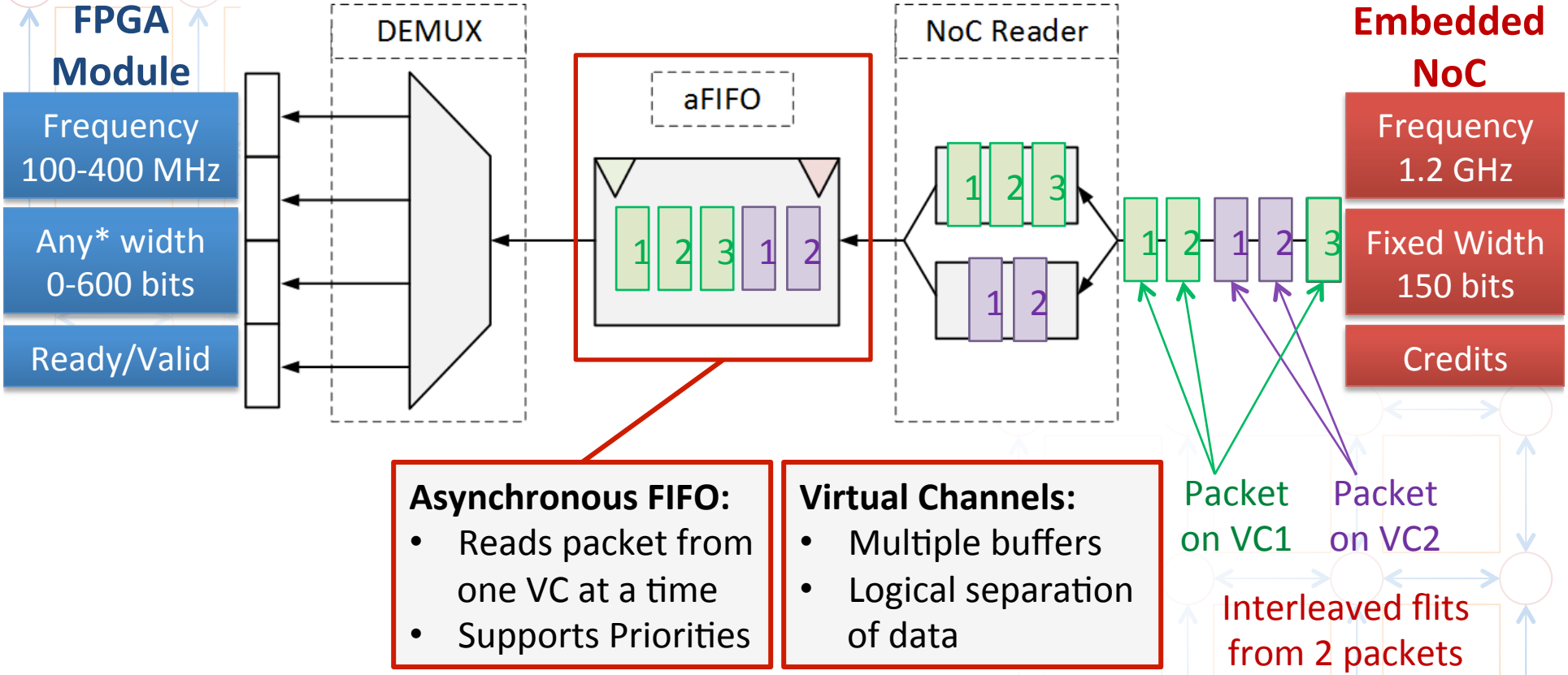
⇦ FabricPort Out ⇦



⇦ FabricPort Out ⇦



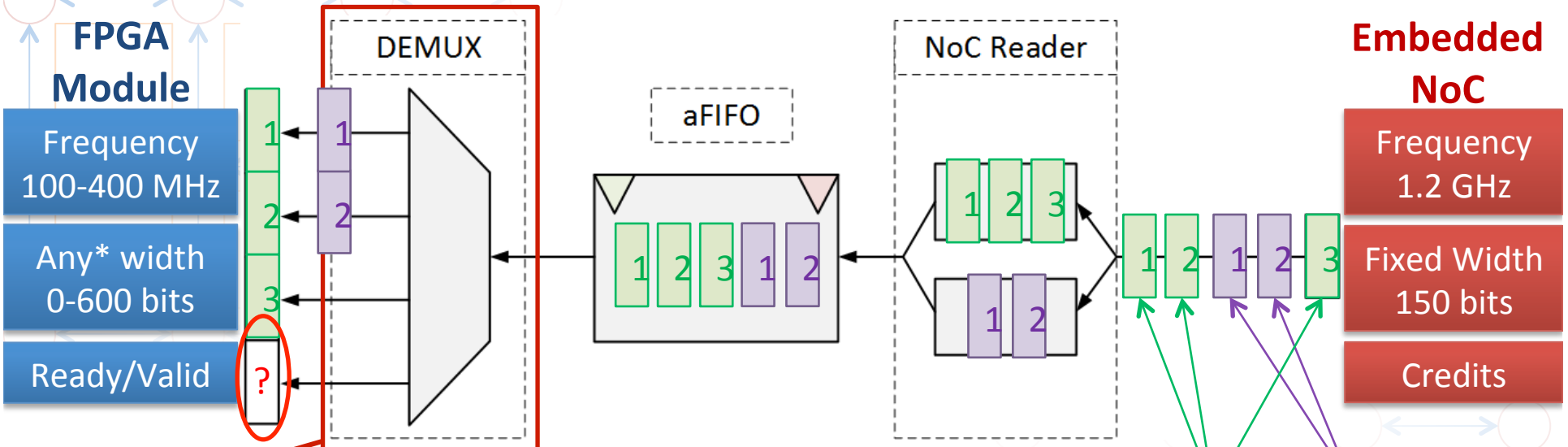
⇦ FabricPort Out ⇦



Asynchronous FIFO:
- Reads packet from one VC at a time
- Supports Priorities

Virtual Channels:
- Multiple buffers
- Logical separation of data

⇦ FabricPort Out ⇦



TD Demultiplexing:
- Output 1 packet at a time → *easy* for designers

Asynchronous FIFO:
- Reads packet from one VC at a time
- Supports Priorities

Virtual Channels:
- Multiple buffers
- Logical separation of data

Outline

How do we ***adapt*** Embedded NoCs to FPGAs?

1. **FabricPort**: NoC-to-FPGA interface

2. NoC with FPGA **design styles**

3. **Case studies**: JPEG & Ethernet switch

Rules for using an NoC with FPGA design styles

Photo from: http://www.dailymail.co.uk

Packet Ordering

Multiprocessors

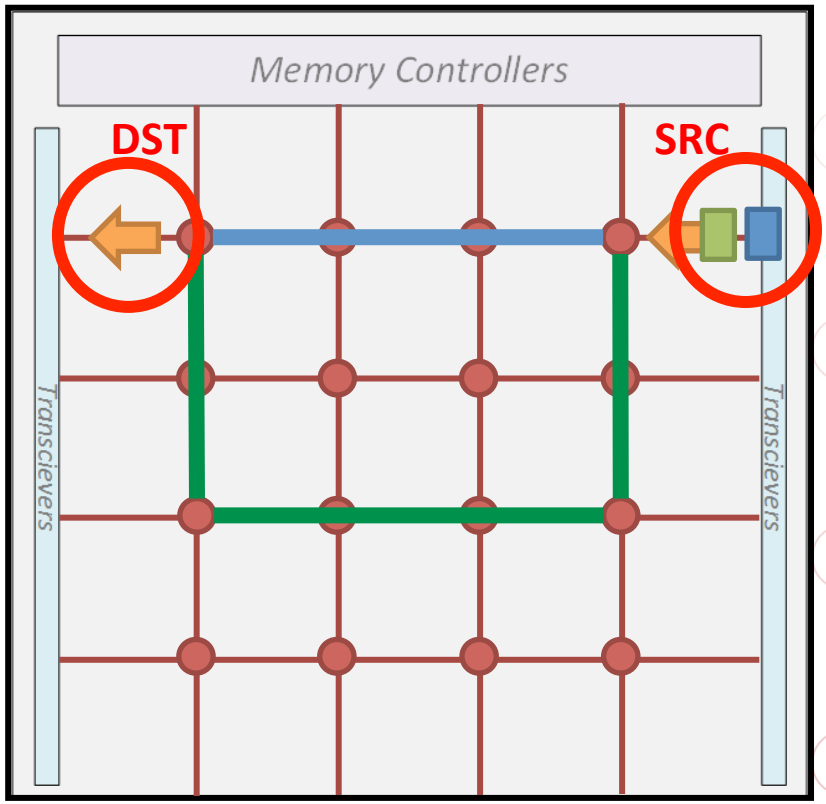
- Memory mapped
- Packets arrive out-of-order
  - Fine for cache lines
  - Processors have re-order buffers

FPGA Designs

- Mostly streaming
- Cannot tolerate reordering
  - Hardware expensive and difficult

RULE

All packets with same src/dst must take **same NoC path**



Packet Ordering

Multiprocessors

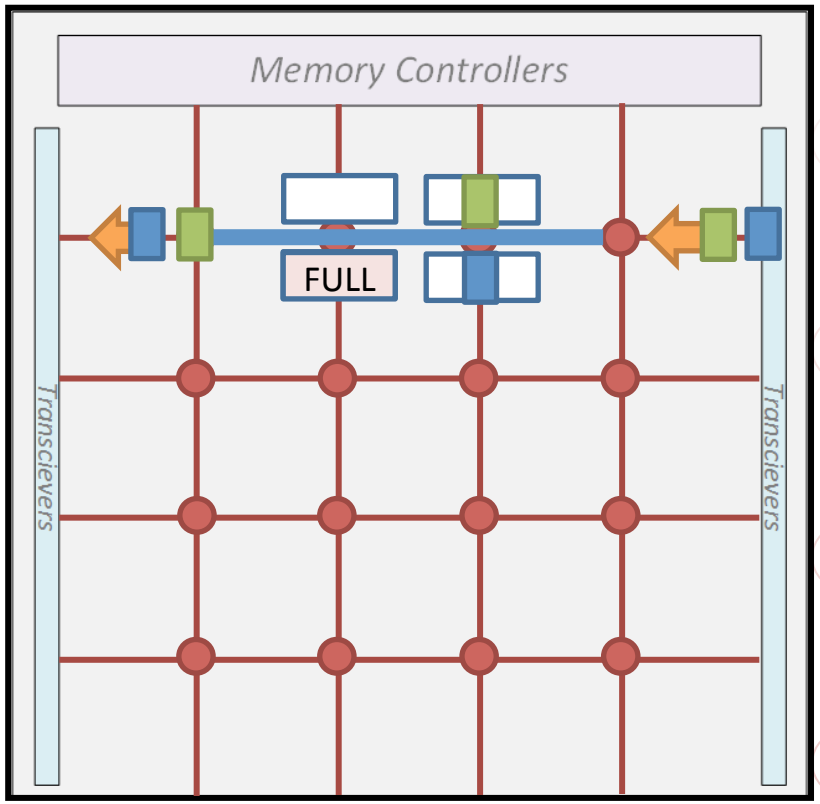
- Memory mapped
- Packets arrive out-of-order
  - Fine for cache lines
  - Processors have re-order buffers

FPGA Designs

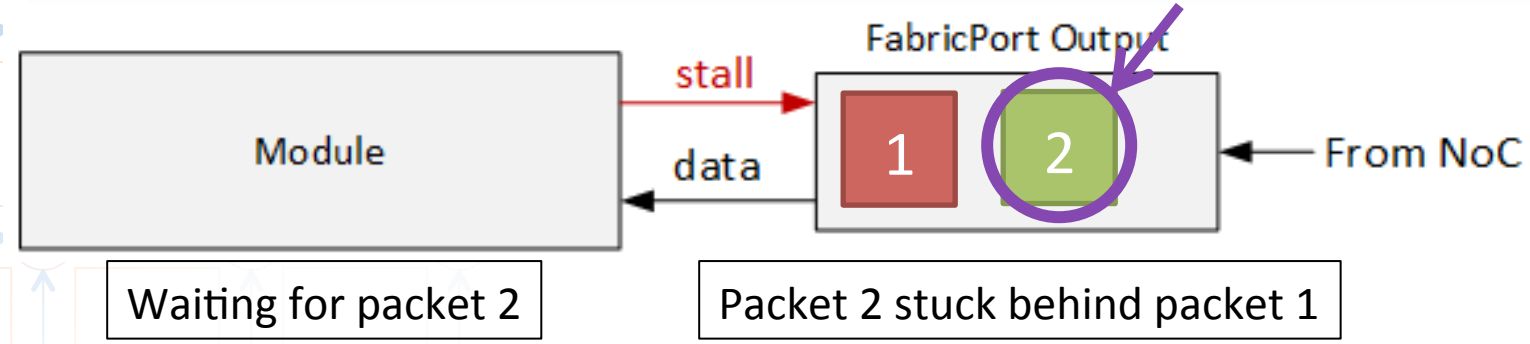
- Mostly streaming
- Cannot tolerate reordering
  - Hardware expensive and difficult

RULE All packets with same src/dst must take **same NoC path**

RULE All packets with same src/dst must take **same VC**

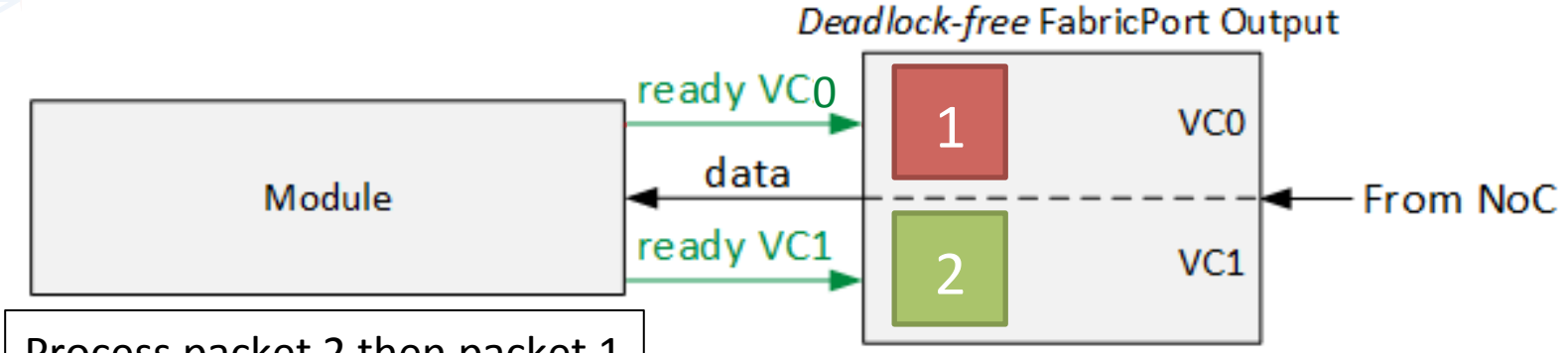


Dependencies & Deadlock



Waiting for packet 2

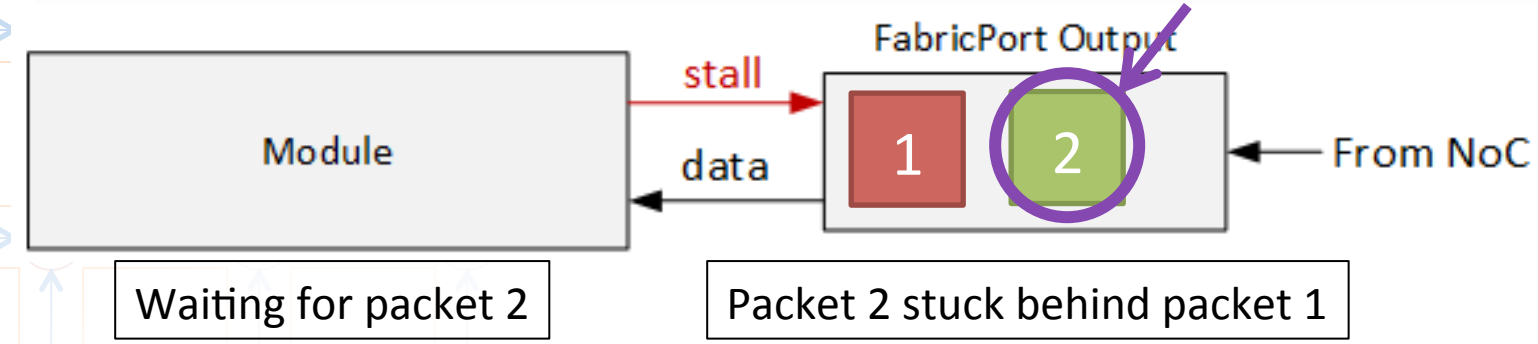
Packet 2 stuck behind packet 1



Process packet 2 then packet 1

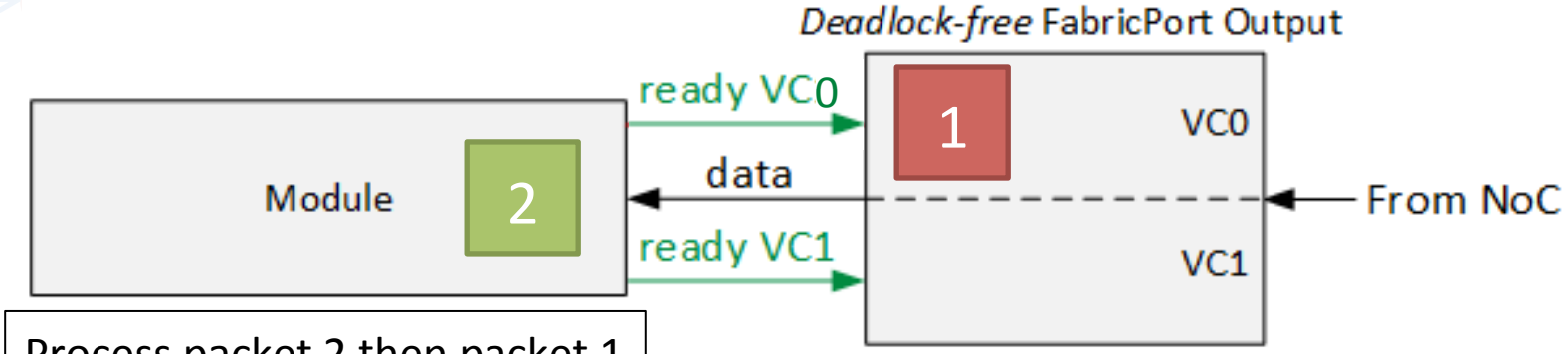
Each Packet in different VC

Dependencies & Deadlock



Waiting for packet 2

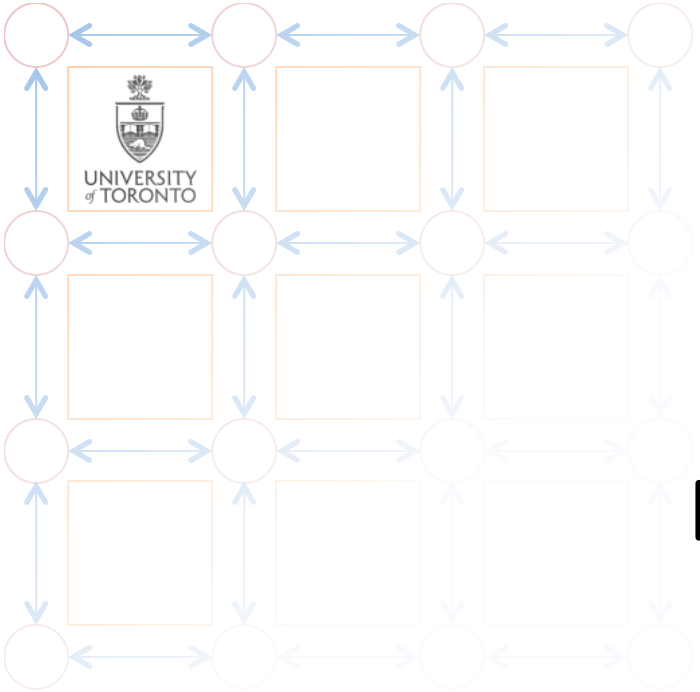
Packet 2 stuck behind packet 1



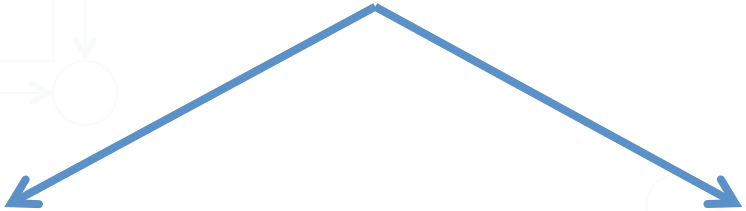
Process packet 2 then packet 1

Each Packet in different VC

RULE Each message type must take a different VC
→ FabricPort output must support that

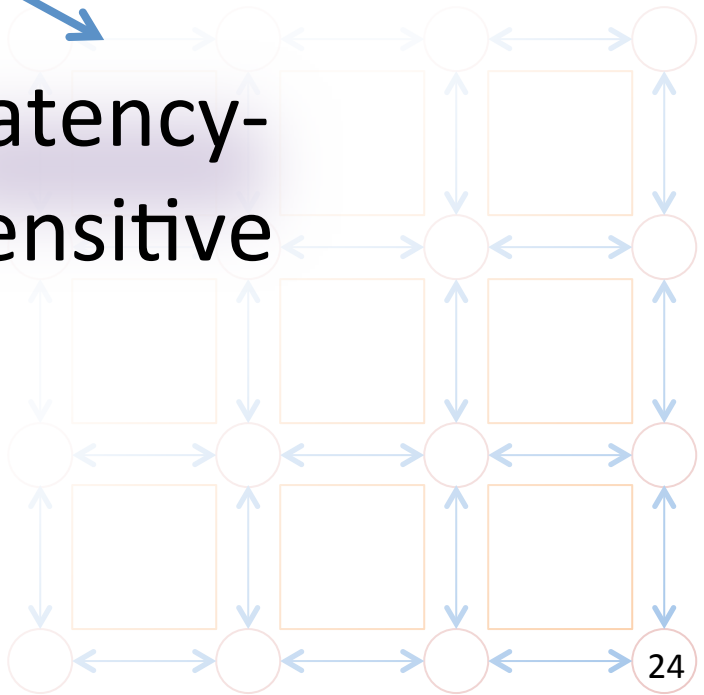


Design Styles



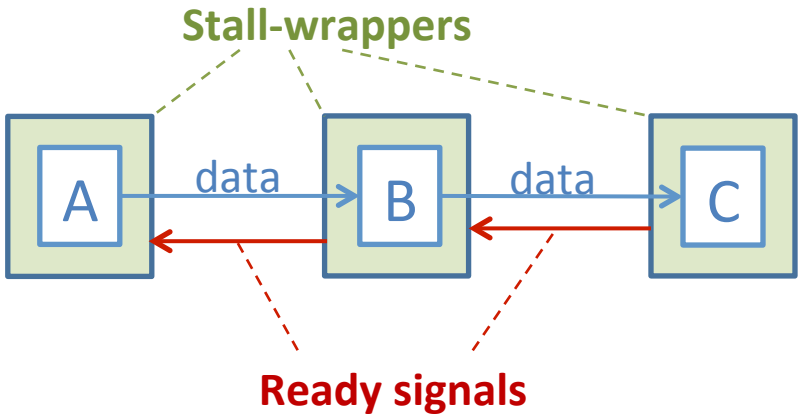
Latency-insensitive

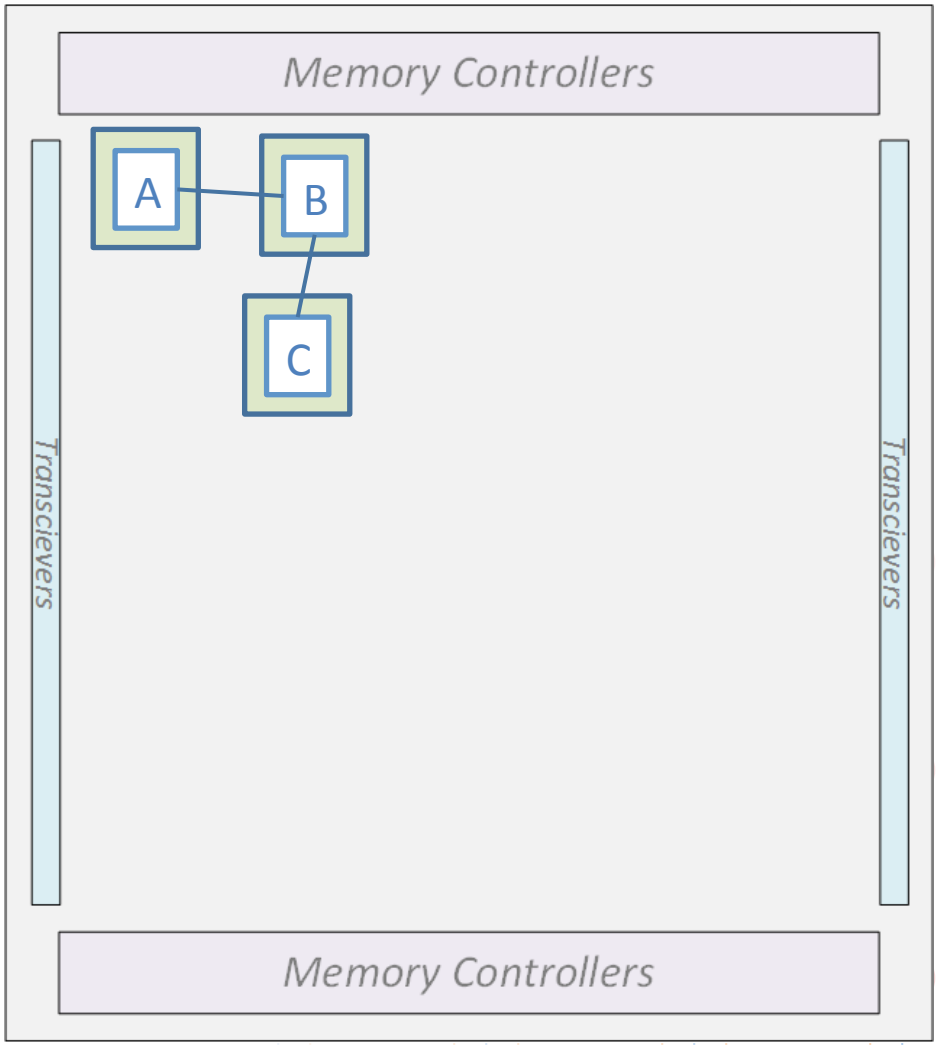
Latency-sensitive



1. Latency-Insensitive Design

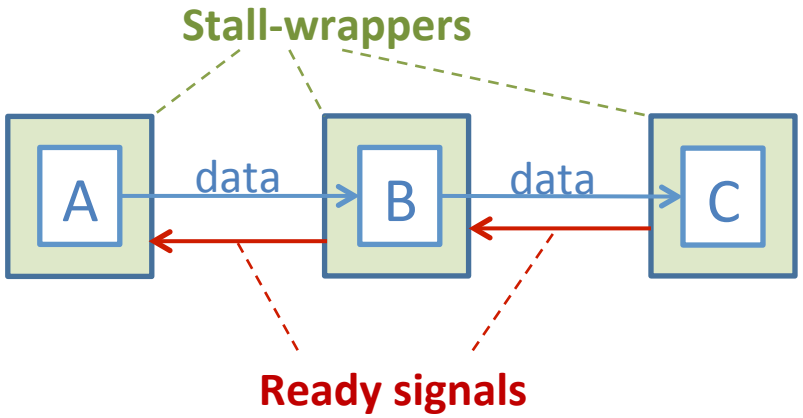
- Latency-Insensitive
- Modules are stallable
- Ready/valid signals
- Tolerate latency on links

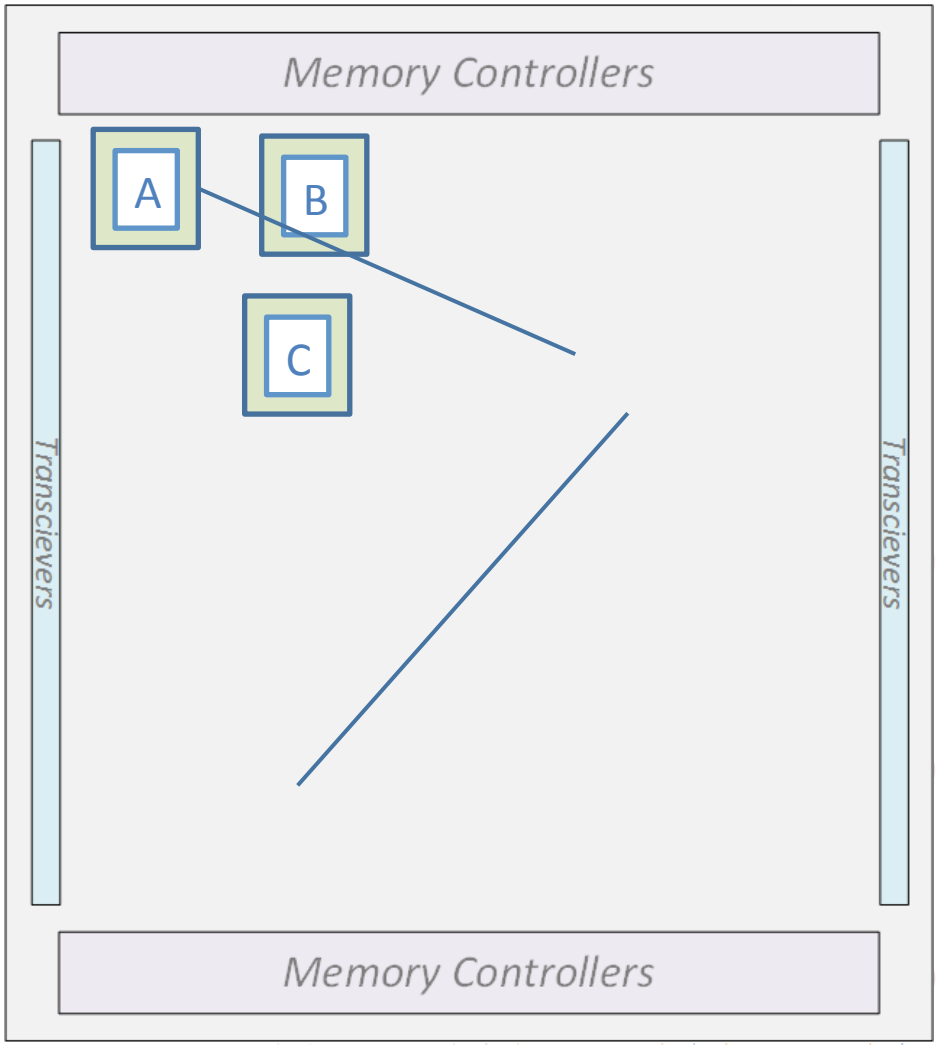




1. Latency-Insensitive Design

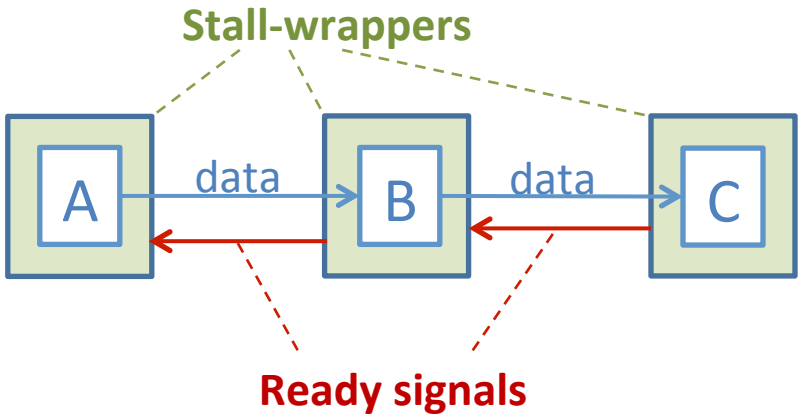
- Latency-Insensitive
- Modules are stallable
- Ready/valid signals
- Tolerate latency on links

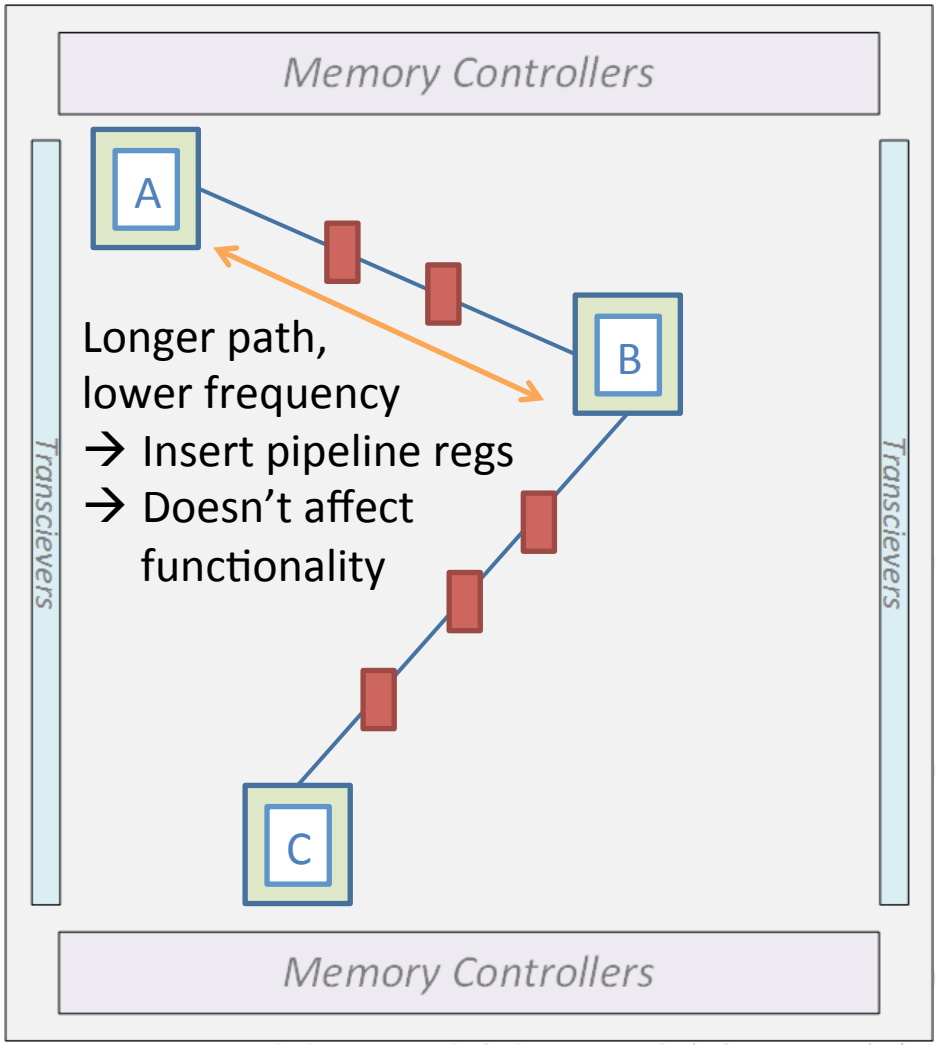




1. Latency-Insensitive Design

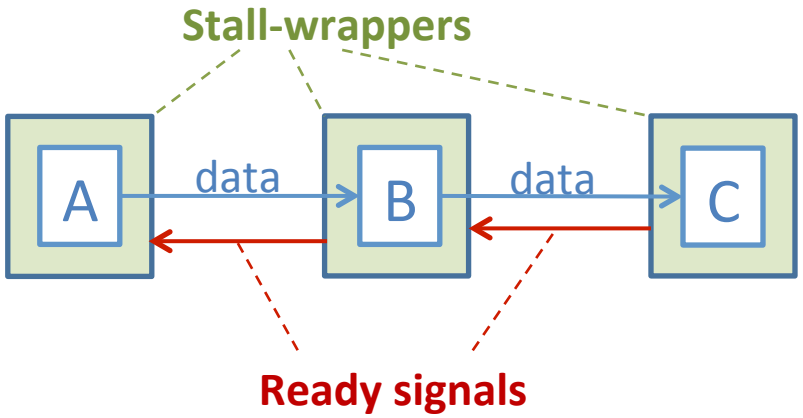
- Latency-Insensitive
- Modules are stallable
- Ready/valid signals
- Tolerate latency on links

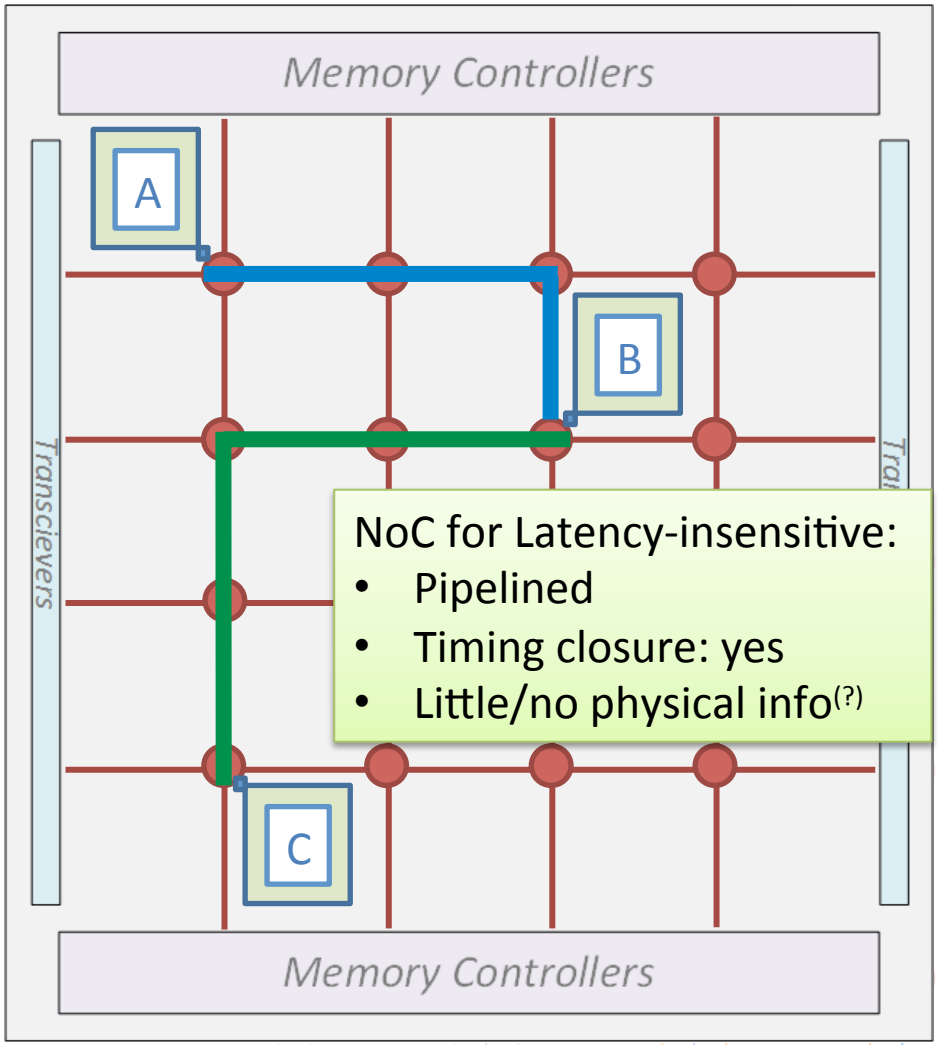




1. Latency-Insensitive Design

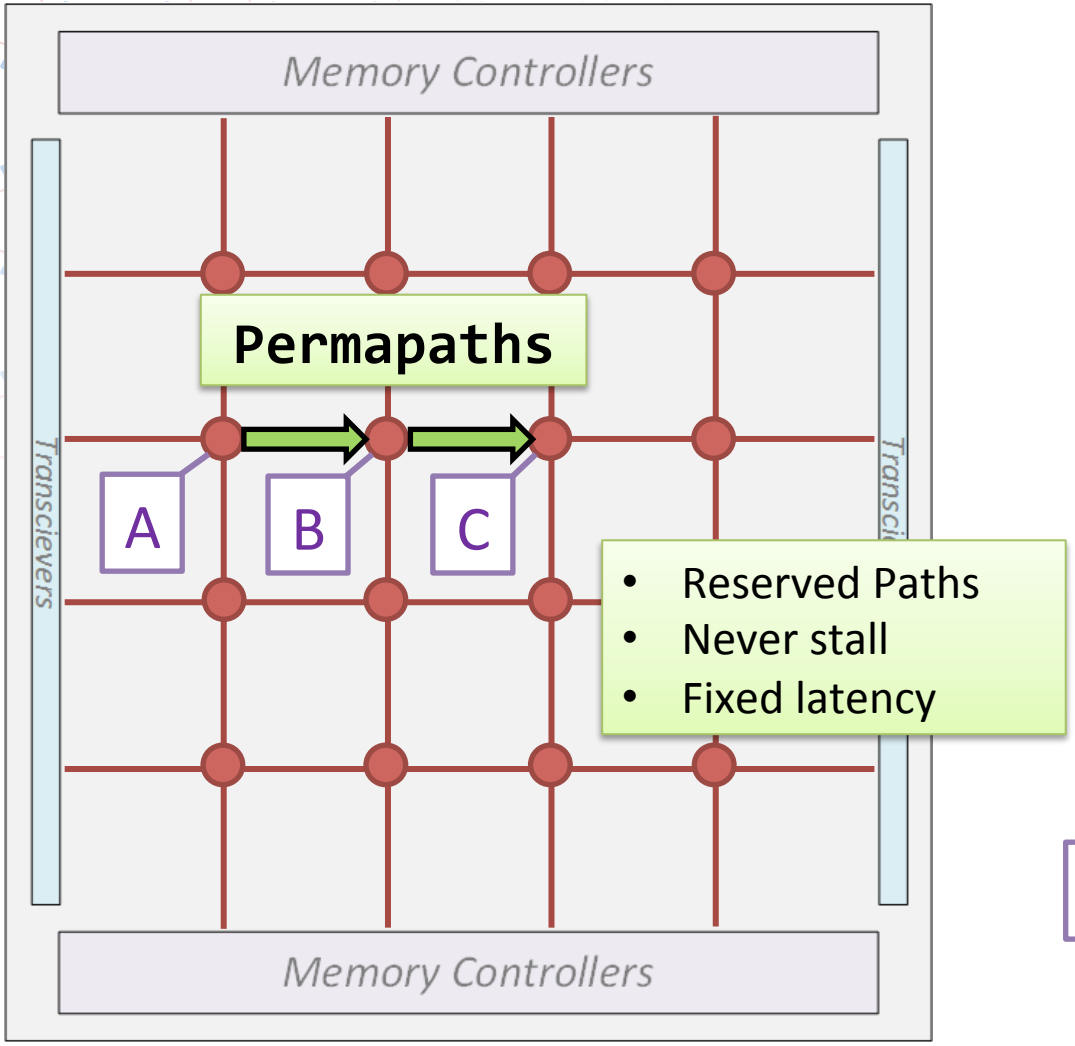
- Latency-Insensitive
- Modules are stallable
- Ready/valid signals
- Tolerate latency on links





NoC for Latency-insensitive:
- Pipelined
- Timing closure: yes
- Little/no physical info(?)

2. Latency-Sensitive Design



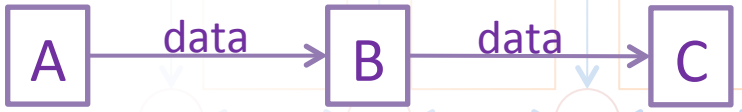
- Reserved Paths
- Never stall
- Fixed latency

Latency-Sensitive

Not stallable

No Ready/valid signals

Latency must remain constant to be functional



Outline

How do we ***adapt*** Embedded NoCs to FPGAs?

1. **FabricPort**: NoC-to-FPGA interface

2. NoC with FPGA **design styles**

3. **Case studies**: JPEG & Ethernet switch

RTL2Booksim

RTL (Verilog)

- Hardware designs
- FabricPort
- NoC looks exactly like a hardware module

SV DPI

RTL2Booksim

Sockets

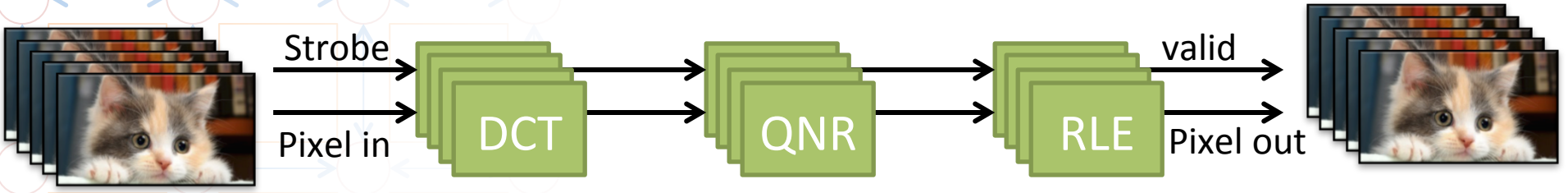
Booksim

- NoC simulator
- Cycle-accurate
- Widely-used

Simulate **any RTL** design with a **flexible** NoC Simulator

www.eecg.utoronto.ca/~mohamed/rtl2booksim

Parallel **JPEG** Compression



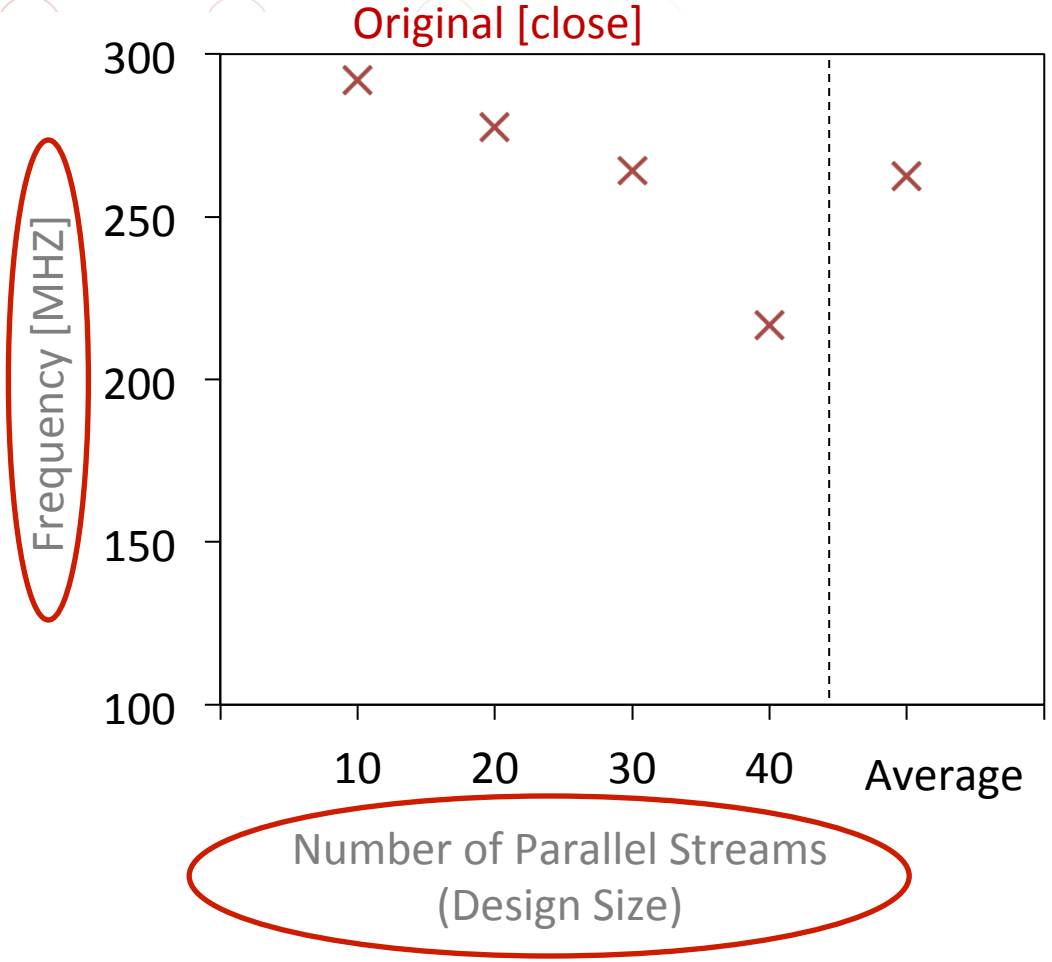
- Latency-sensitive [Permapaths]

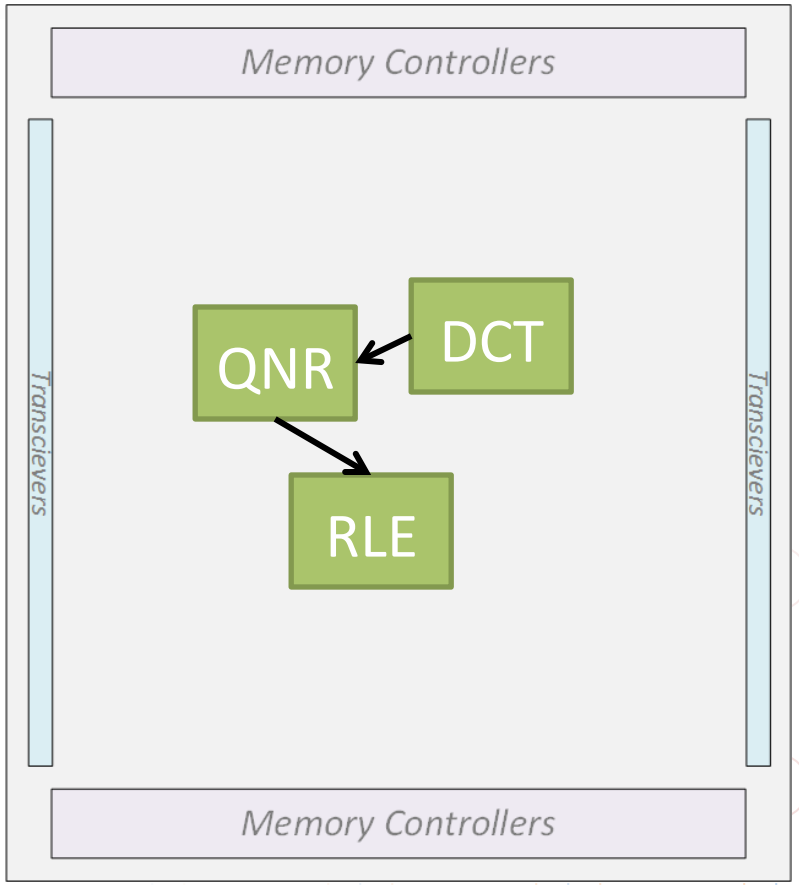
Strobe	0
Pixel	X

- Parallelize **10 – 40** times
  - Data width **130 – 520** bits
- Important for data center applications
- Compare NoC version to non-NoC version

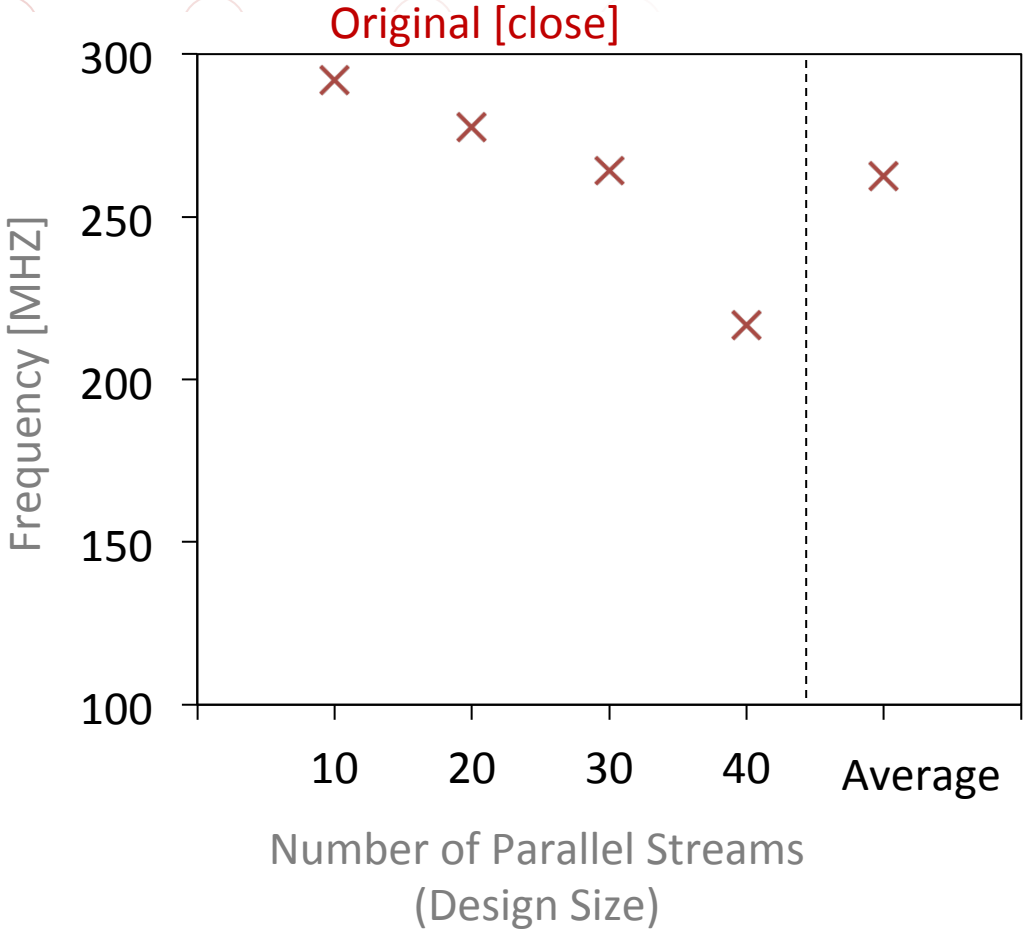
Photo from: http://www.fbpapa.com

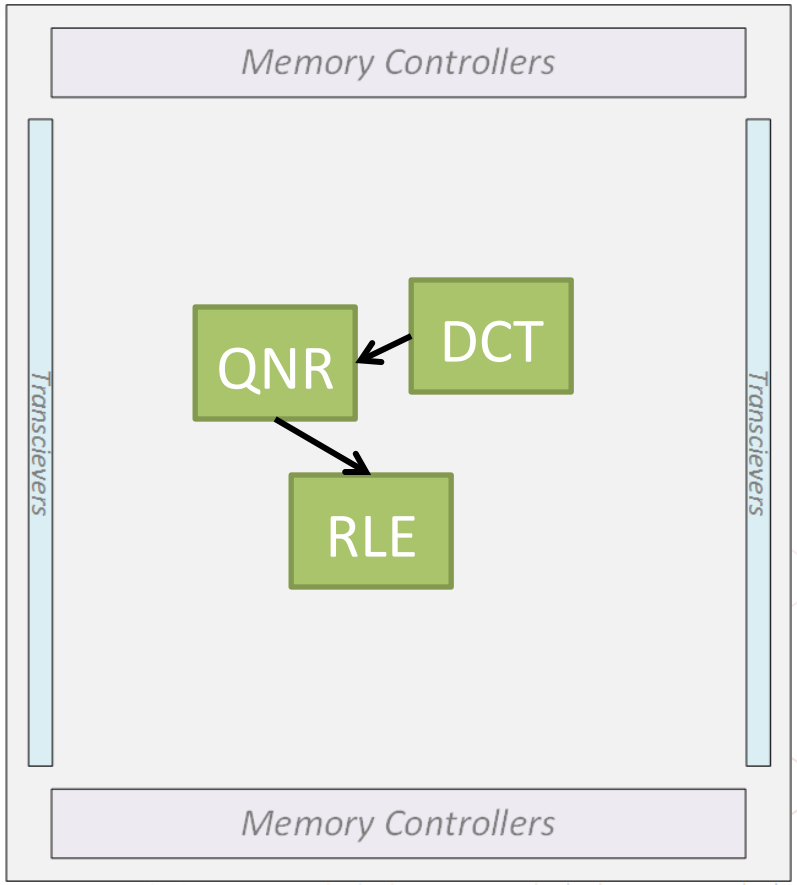
Parallel JPEG



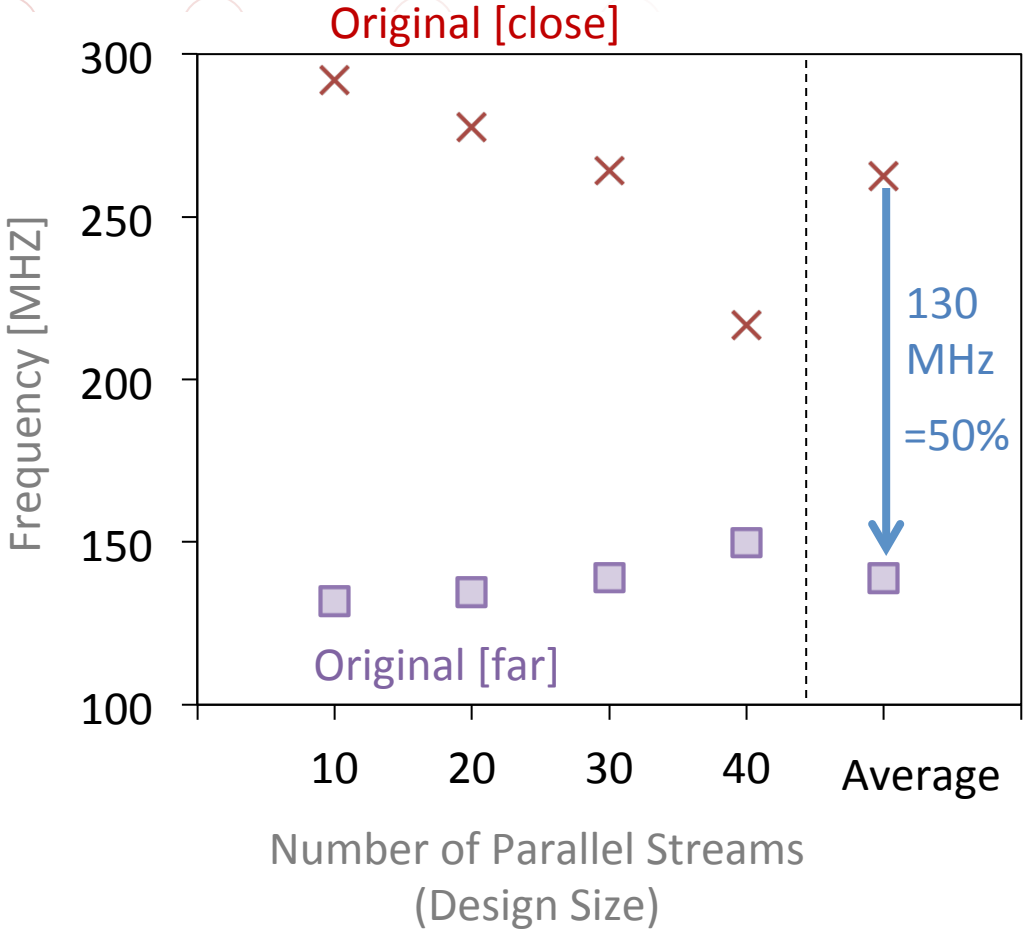


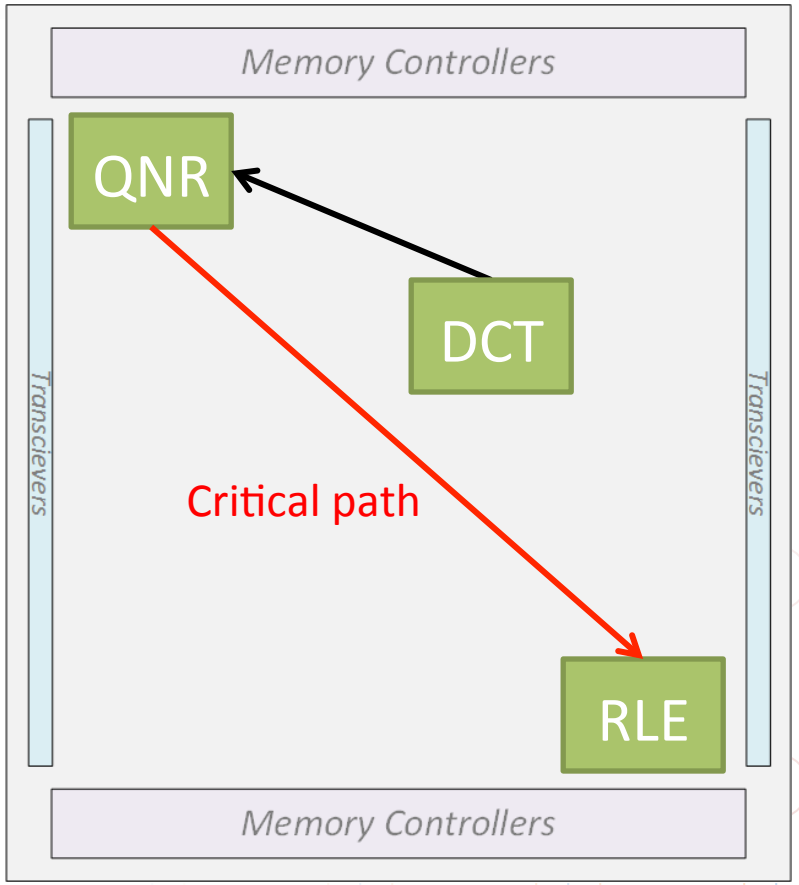
Parallel JPEG



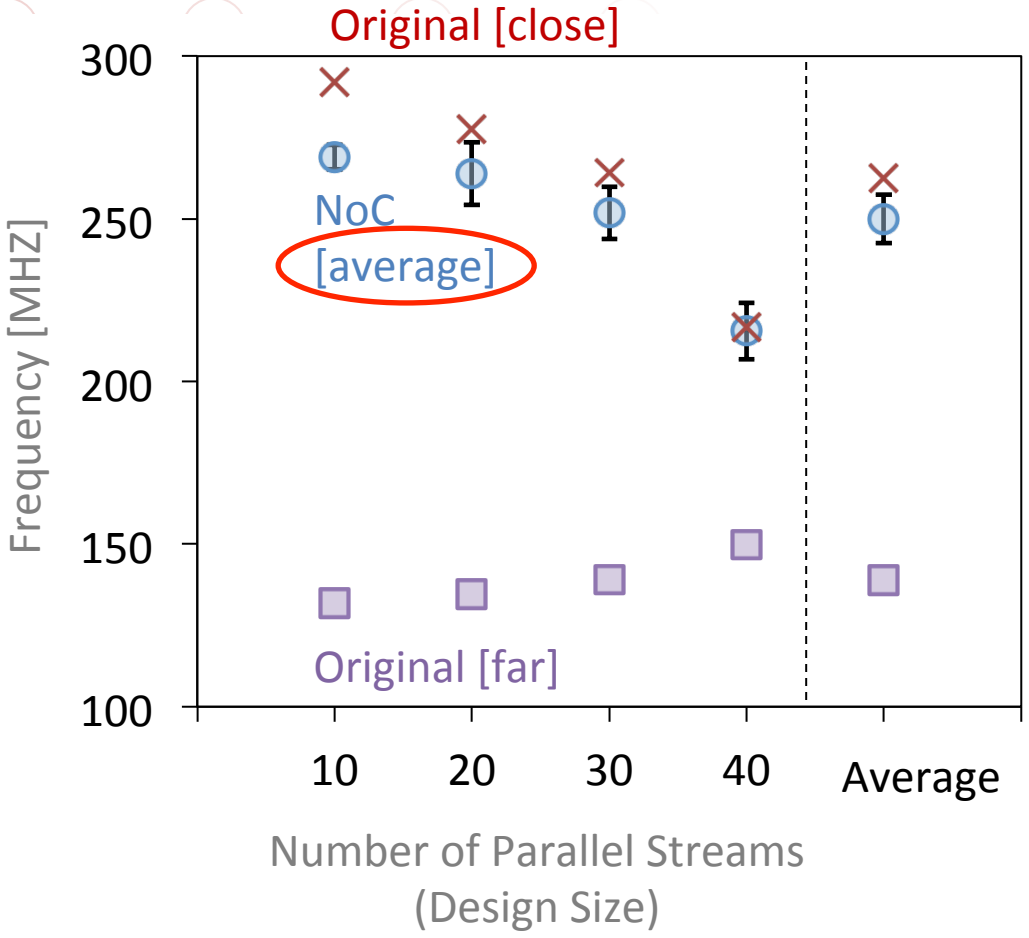


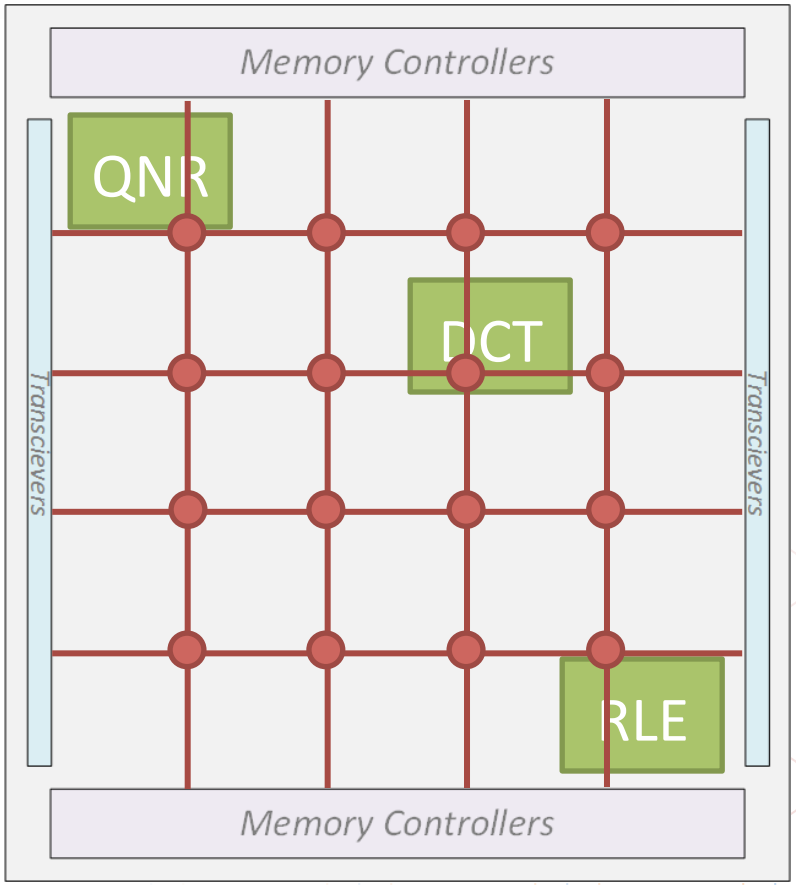
Parallel JPEG



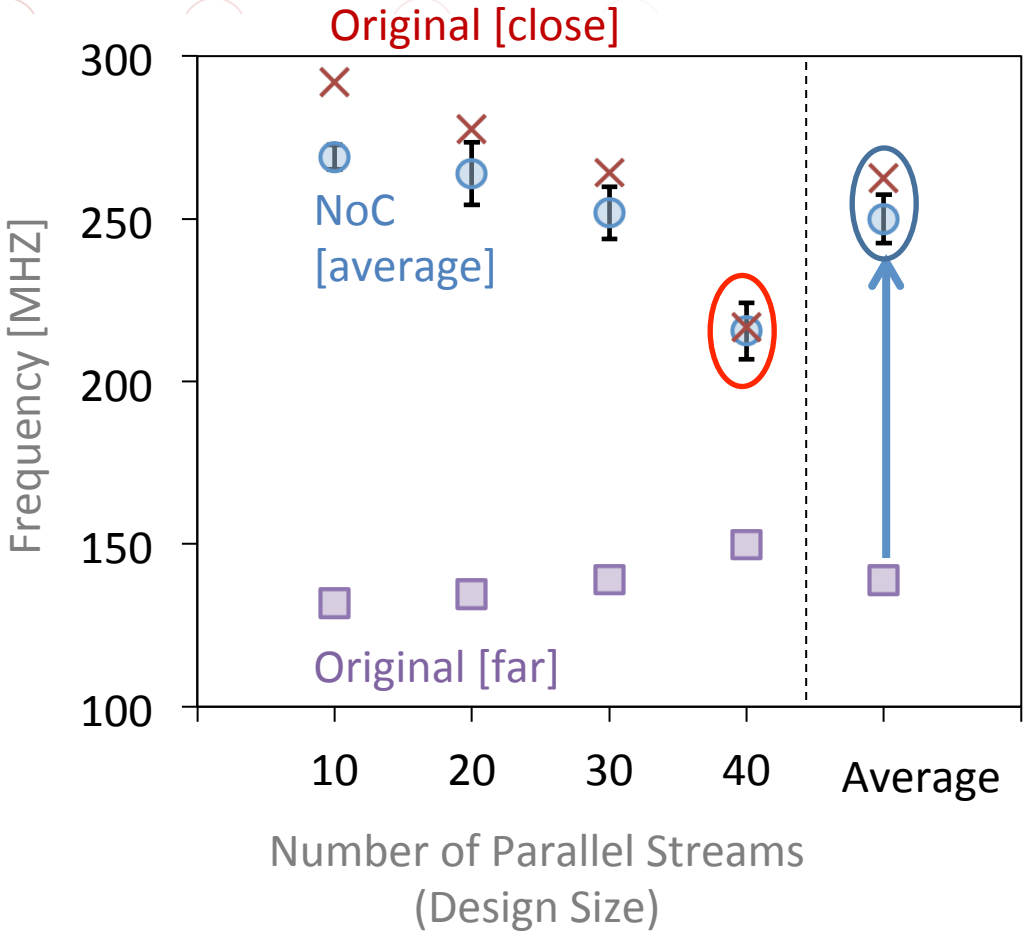


Parallel JPEG

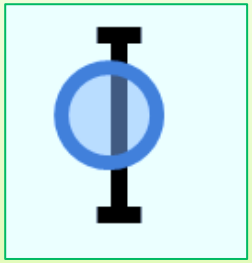




Parallel JPEG

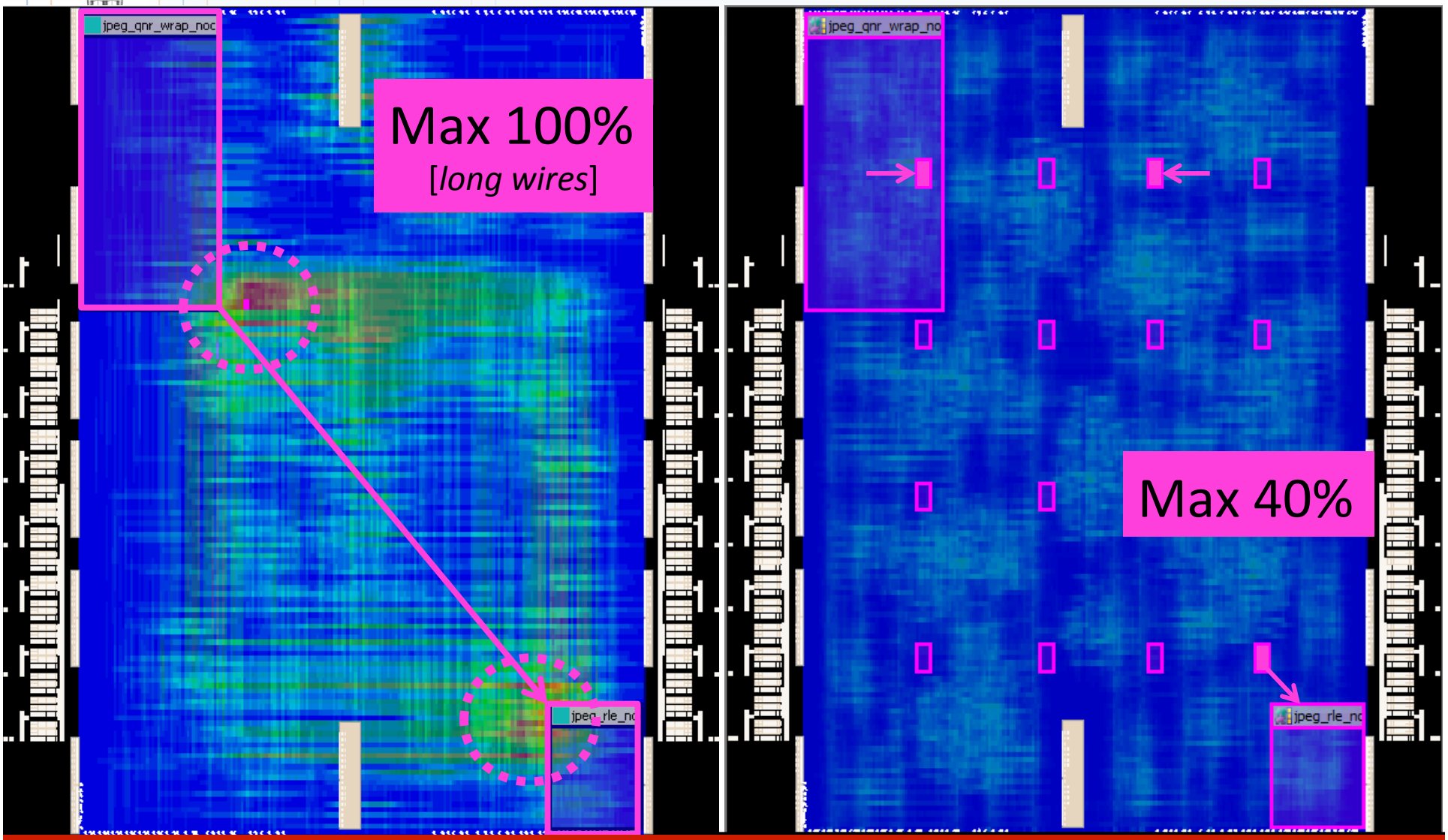


- **5%** drop in frequency from best case
- No drop in frequency for largest design
- **80%** improvement from worst case
- **~6%** variability regardless of module placement



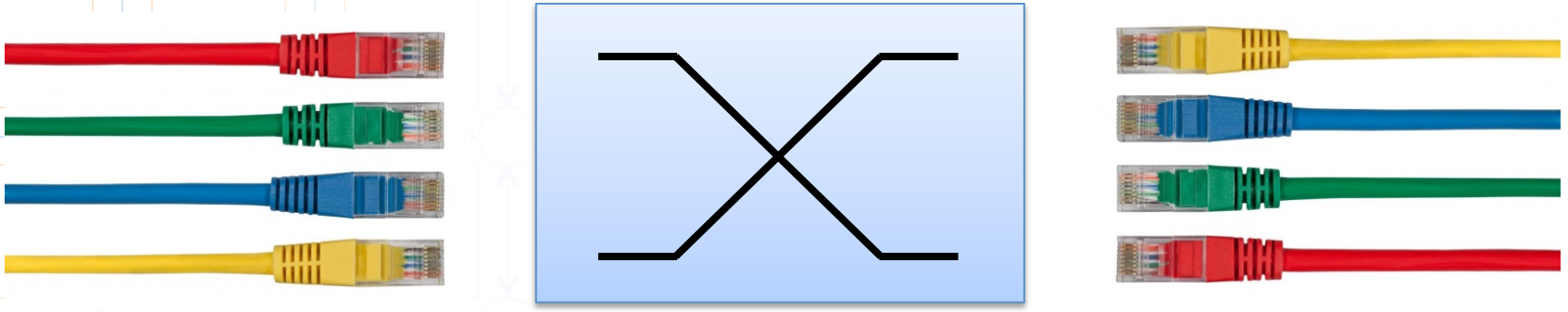
Frequency is **good**, and (much) more **predictable** with NoC

Parallel JPEG



- NoC doesn't produce routing hotspots – it reduces them
- NoC reduces utilization of scarce long FPGA wires

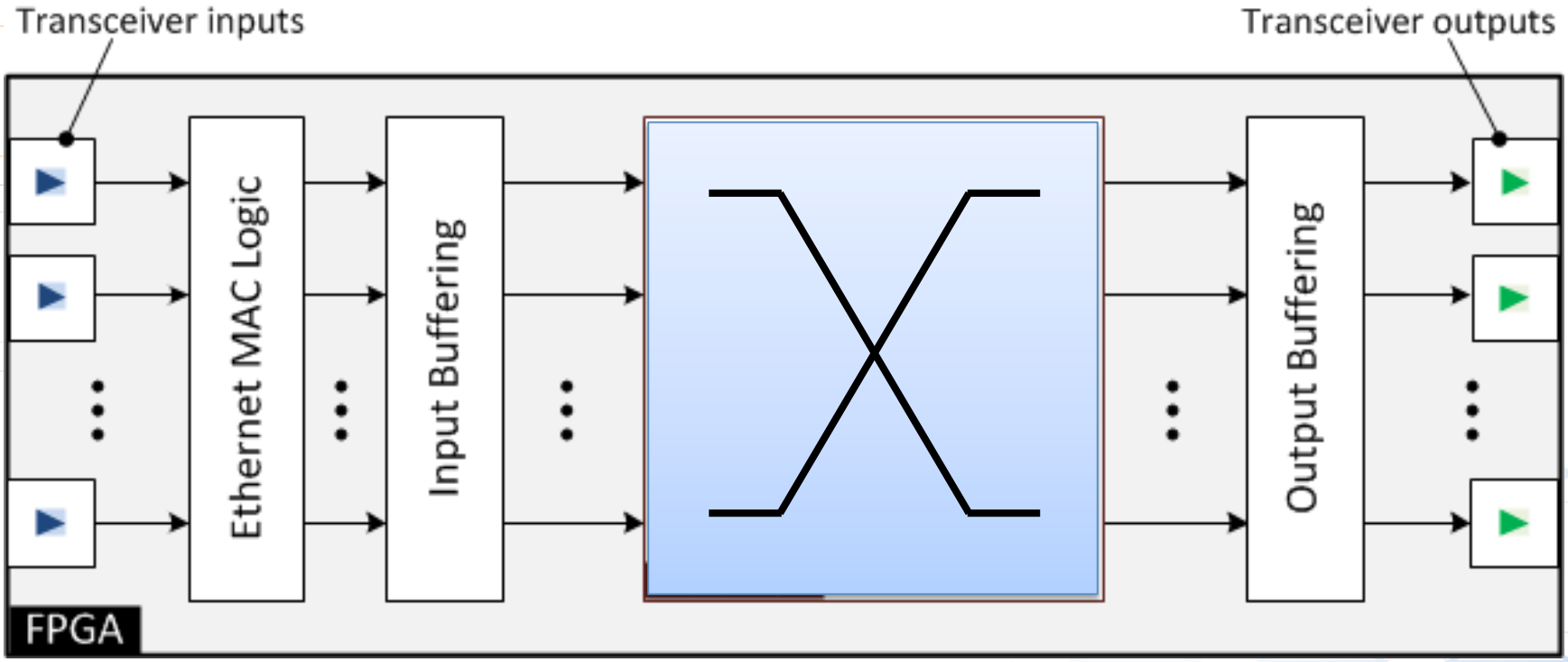
Ethernet Switch



- High Bandwidth Ethernet switches → **ASIC** land
  - **10/25/40/100 Gb/s** per port switches, e.g. **32x32**
- Best previous FPGA switch:
  - **16x16**, 10 Gb/s per port, total **160 Gb/s**
  - FPGAs have transceiver BW more than 1000 Gb/s

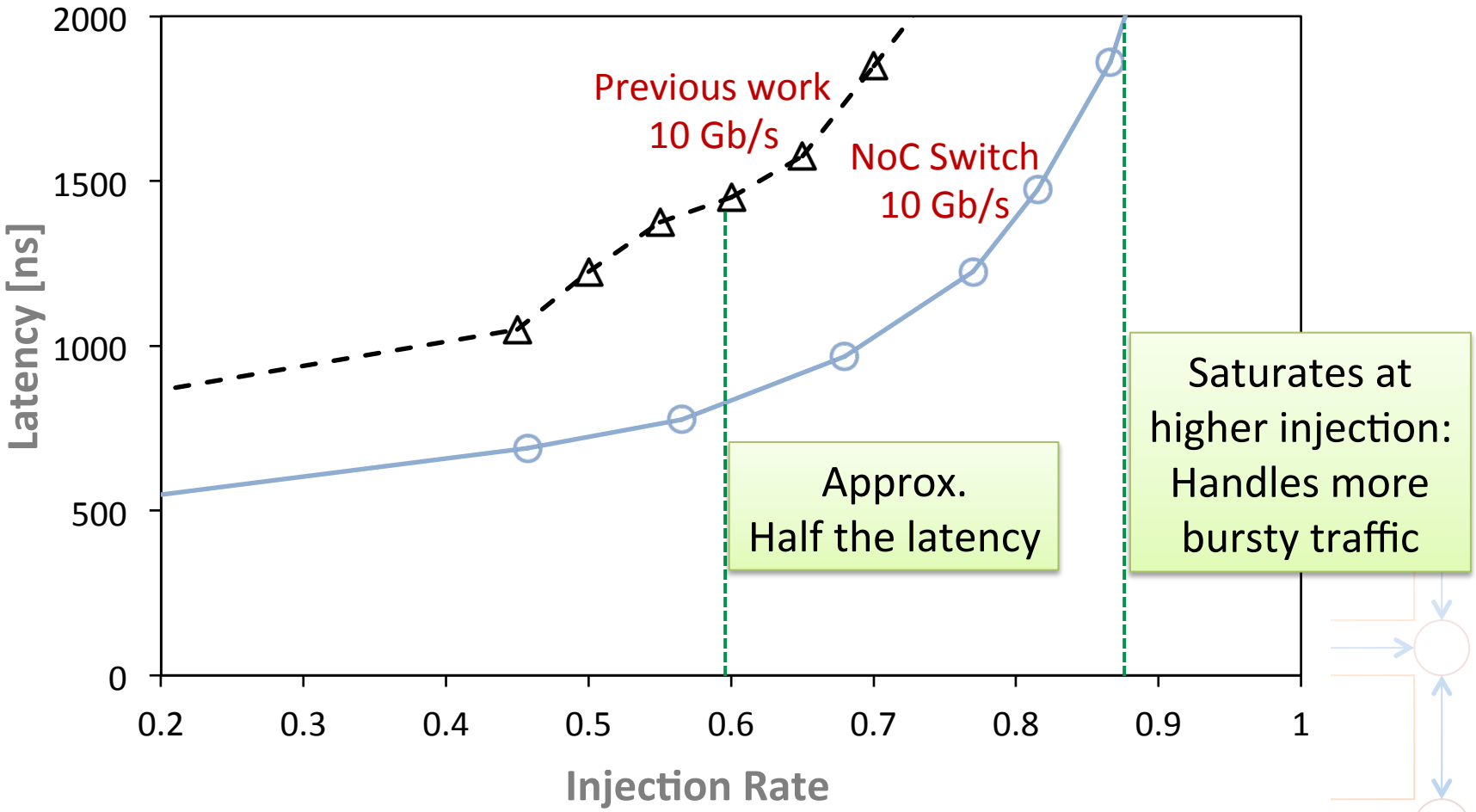
Photo from: http://wisegeek.org

Ethernet Switch

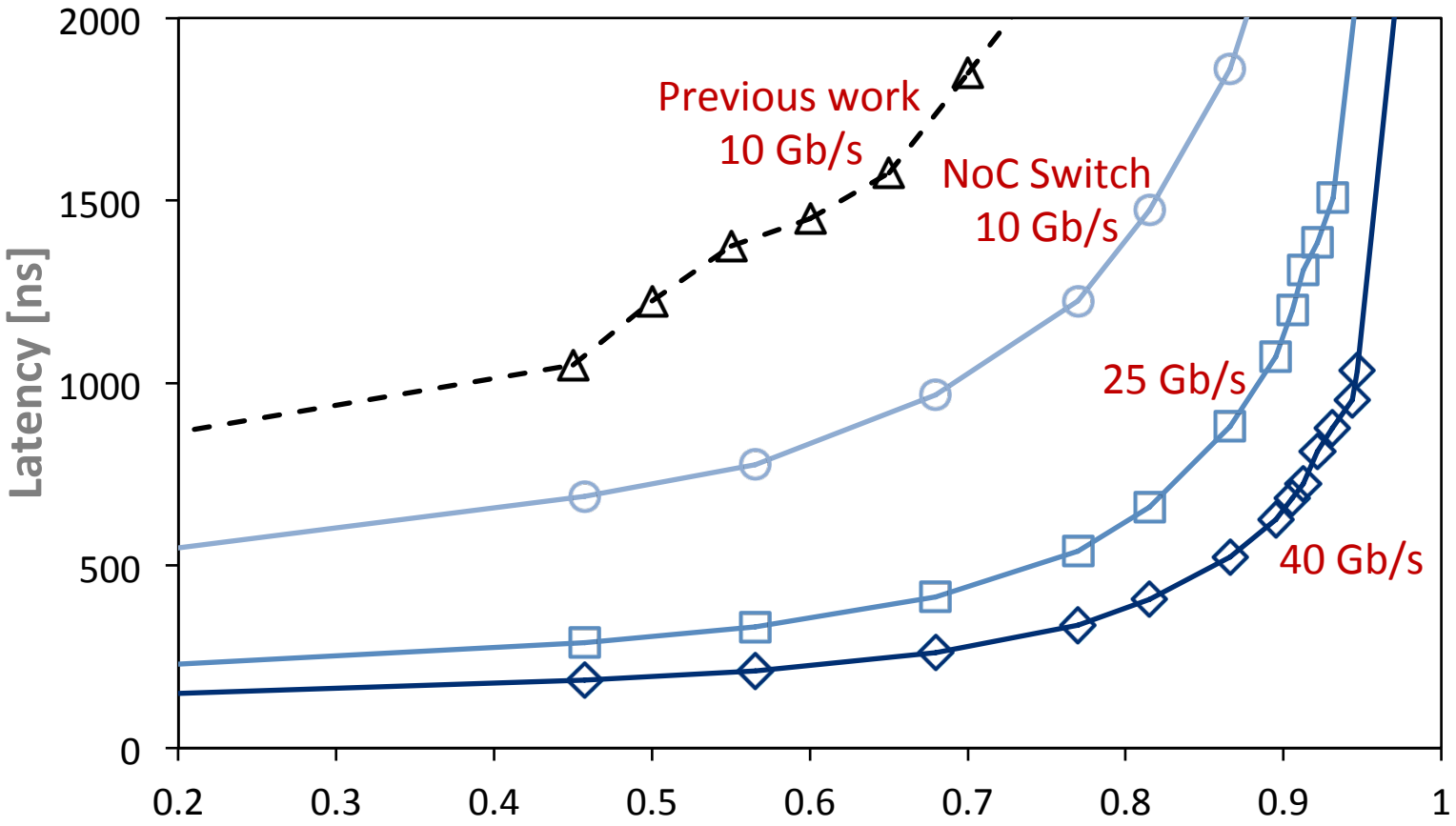


- Embedded NoC is the switch **(16X16 switch)**
- Use `RTL2Booksim` to measure latency/throughput

Ethernet Switch (16x16)



Ethernet Switch (16x16)



- **5X** more switching than best: 819 Gb/s vs 160 Gb/s
- ~ **Half** the latency
- **3X** smaller area

Reconfigurable speed, radix, protocol – augment pkt processing

How do we ***adapt*** Embedded NoCs to FPGAs?

1. **FabricPort**: NoC-to-FPGA interface

- Module (*any* freq./width) → Embedded NoC (1.2 GHz, 150 bits)
- **Designer-friendly**
- Leverage VCs, credits

2. NoC with FPGA **design styles**

- Rules for NoC design on FPGAs: packet **ordering** and **dependencies**
- Support FPGA Design Styles: Latency-insensitive design
  Latency-sensitive design

3. **Case studies**: JPEG & Ethernet switch

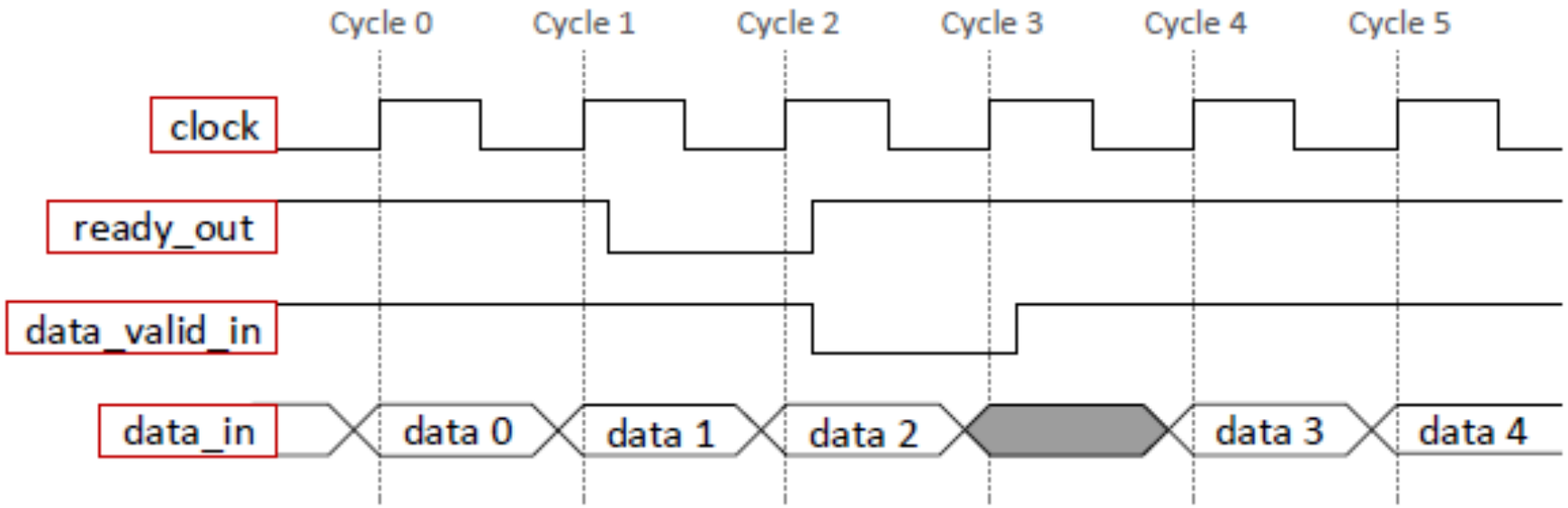
- Parallel JPEG: Frequency **predictable** and **good**,
  Reduces routing hotspots and long wire utilization
- Ethernet Switch: **5X** more switching, **3X** less area, reconfigurable

Compelling case for Embedded NoCs

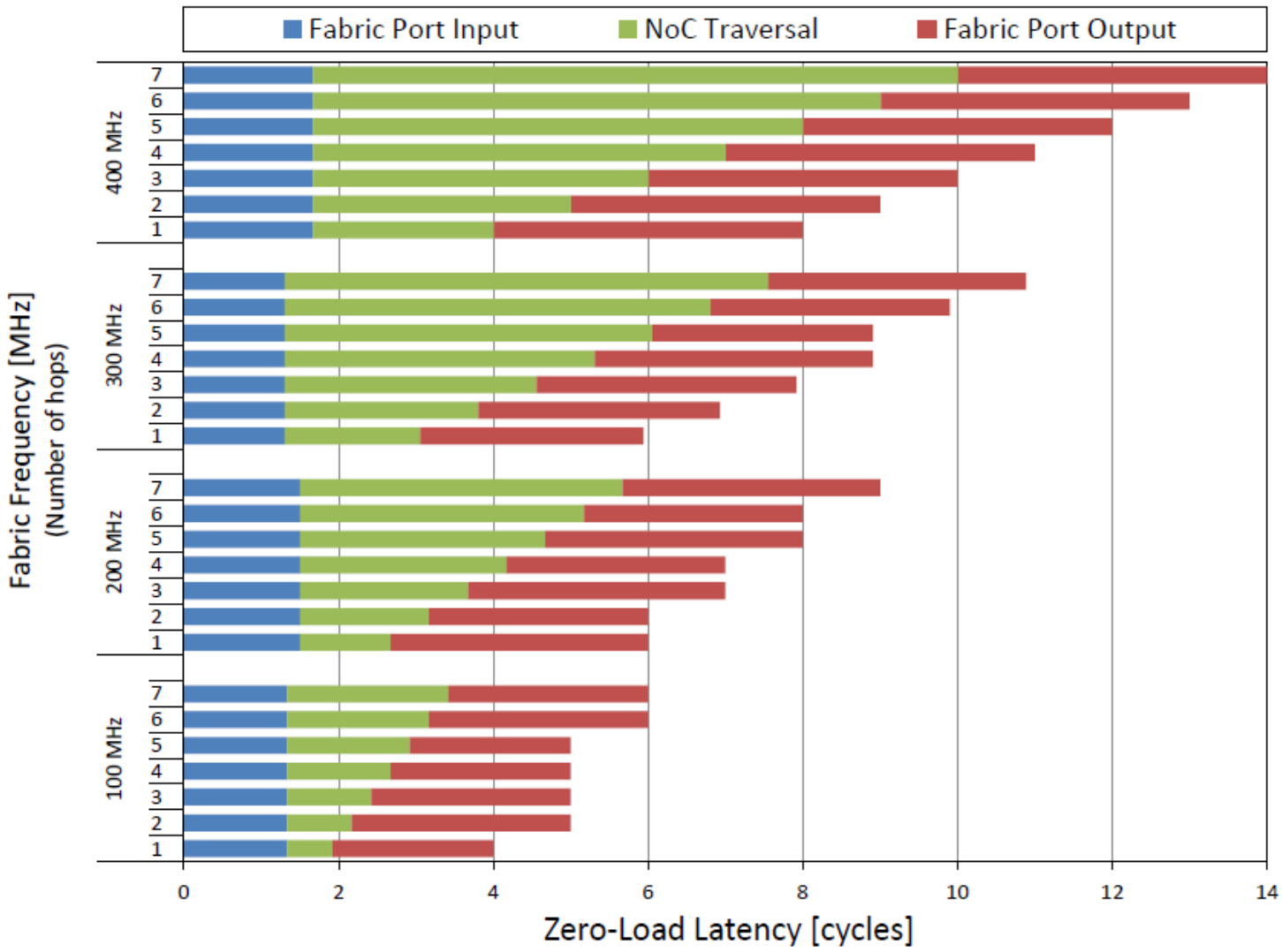
Thank You!

www.eecg.utoronto.ca/~mohamed

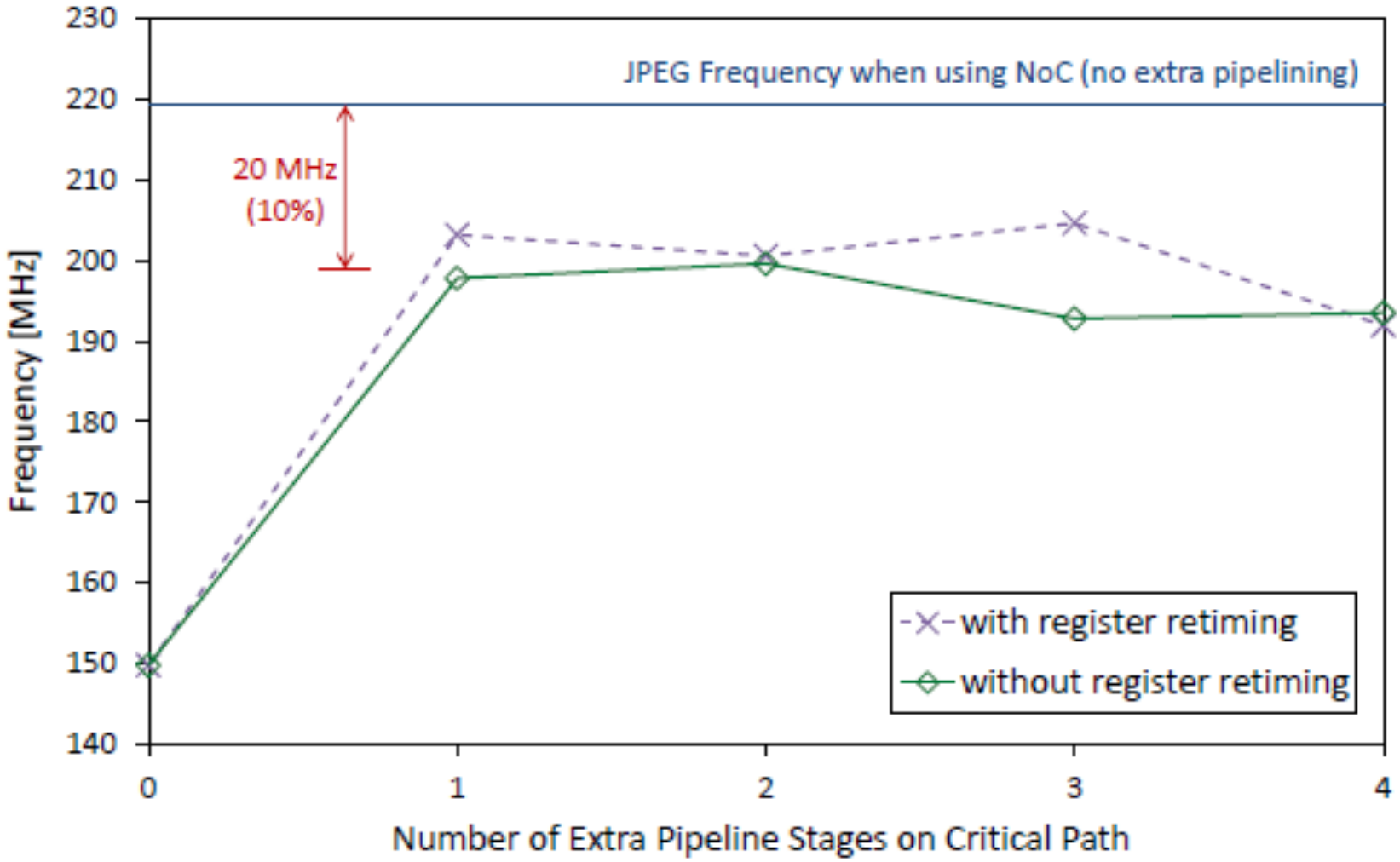
NoC Interface



Permapath Latency



JPEG with pipelining

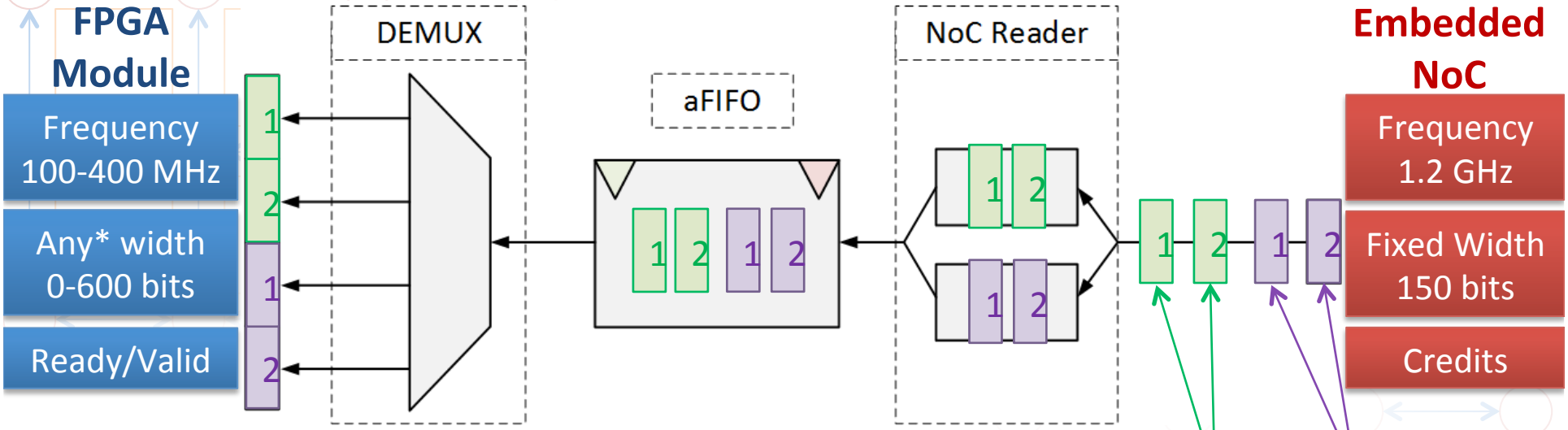


JPEG Interconnect Utilization

Interconnect Resource		Difference	Geomean
Short	Vertical (C4)	+13.2%	+10.2%
	Horizontal (R3,R6)	+7.8%	
Long	Vertical (C14)	-47.2%	-38.6%
	Horizontal (R24)	-31.6%	

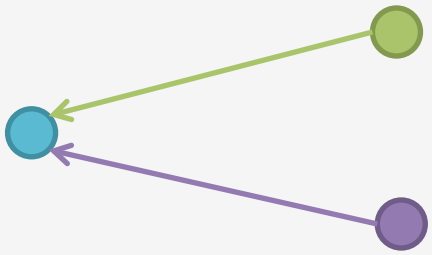
Wire naming convention: C=column, R=row, followed by number of logic clusters of wire length.

⇦ FabricPort Out ⇦



***combine_data* mode:**

- Packets from VC1 → upper bits
- Packets from VC2 → lower bits



UNIVERSITY
OF TORONTO