Interconnection Networks

Computer Architecture: A Quantitative Approach 4th Edition, Appendix E

Timothy Mark Pinkston
University of Southern California
http://ceng.usc.edu/smart/slides/appendixE.html

José Duato
Universidad Politécnica de Valencia
http://www.gap.upv.es/slides/appendixE.html

*…with major presentation contribution from José Flich, UPV
(and Cell BE EIB slides by Tom Ainsworth, USC)*

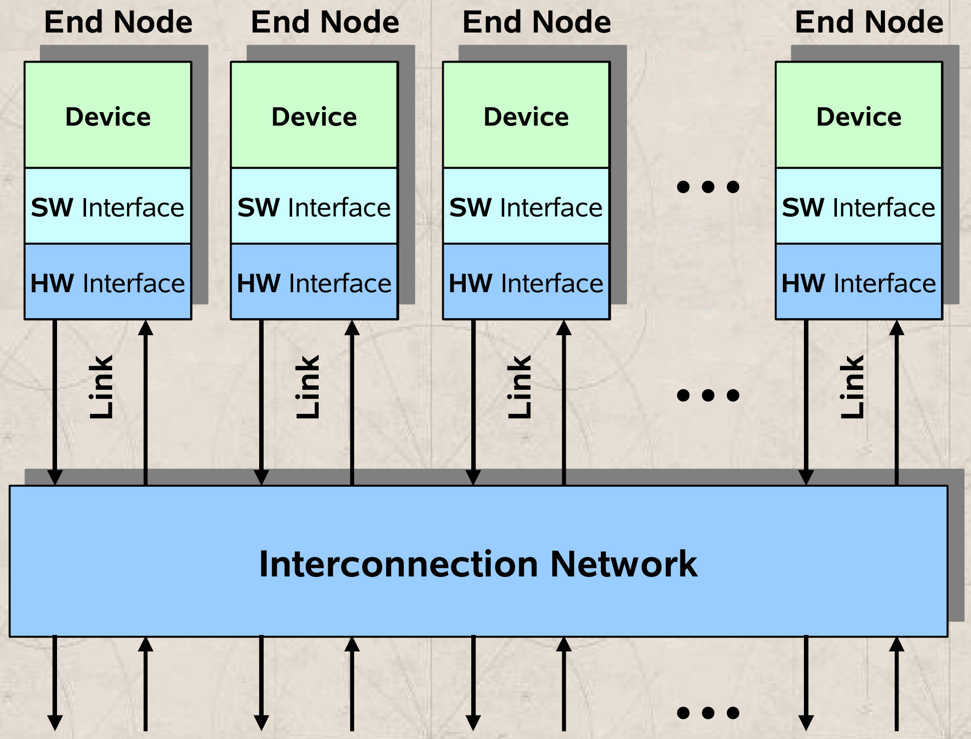
Outline

- **E.1 Introduction *(Lecture 1)***
- E.2 Interconnecting Two Devices *(Lecture 1)*
- E.3 Interconnecting Many Devices *(Lecture 2)*
- E.4 Network Topology *(Lecture 2)*
- E.5 Network Routing, Arbitration, and Switching *(Lecture 3)*
- E.6 Switch Microarchitecture *(Lecture 4)*
- E.7 Practical Issues for Commercial Interconnection Networks *(Lec 4)*
- E.8 Examples of Interconnection Networks *(Lecture 5)*
- E.9 Internetworking (skipped)
- E.10 Crosscutting Issues for Interconnection Networks (skipped)
- E.11 Fallacies and Pitfalls *(Lecture 5)*
- E.12 Concluding Remarks and References *(Lecture 5)*

Introduction

How to connect individual devices together into a community of communicating devices?

- Device (definition):
  - Component within a computer
  - Single computer
  - System of computers
- Types of elements:
  - end nodes (device + interface)
  - links
  - interconnection network



- Internetworking: interconnection of multiple networks

- ***Interconnection networks*** *should be designed to transfer the* *__maximum amount of information__* *within the* *__least amount of time__* *(and cost, power constraints) so as not to bottleneck the system*

Introduction

Reasons to devote attention to interconnection networks

- They provide external connectivity from system to outside world
- Also, connectivity within a single computer system at many levels
  - I/O units, boards, chips, modules and blocks inside chips
- *Trends:* high demand on communication bandwidth
  - increased computing power and storage capacity
  - switched networks are replacing buses
- *Computer architects must understand interconnect problems and solutions in order to more effectively design and evaluate systems*

Application domains

- Networks implemented within processor chips and systems
- Networks implemented across systems

Goal

- *To provide an overview of network problems and solutions*
- *Examine a few case studies and examples*

Introduction

Interconnection Network Domains

Interconnection networks can be grouped into *four major networking domains*, depending on the *number* and *proximity* of devices to be interconnected: *OCNs, SANs, LANs,* and *WANs*

- *On-chip networks* (*OCNs*), a.k.a., network-on-chip (NoC)
  - – Interconnect microarchitecture functional units, register files, caches, compute tiles, processor and IP cores
  - – Chip or multichip modules
  - – Tens (in future, possibly 100s) of devices interconnected
  - – Maximum interconnect distance on the order of centimeters
  - – Examples (custom designed)
    - › Element Interconnect Bus (Cell Broadband Engine processor chip)
      - » 2,400 Gbps (3.2 Ghz processor clock), 12 elements on the chip
  - – Examples (proprietary designs)
    - › CoreConnect (IBM), AMBA (ARM), Smart Interconnect (Sonic)

Introduction

Interconnection Network Domains

- *System/storage area networks* (*SANs*)
  - Multiprocessor and multicomputer systems
    - Interprocessor and processor-memory interconnections
  - Server and data center environments
    - Storage and I/O components
  - Hundreds to thousands of devices interconnected
    - IBM Blue Gene/L supercomputer (64K nodes, each with 2 processors)
  - Maximum interconnect distance typically on the order of tens of meters, but some with as high as a few hundred meters
    - InfiniBand: 120 Gbps over a distance of 300 m
  - Examples (standards and proprietary)
    - InfiniBand, Myrinet, Quadrics, Advanced Switching Interconnect

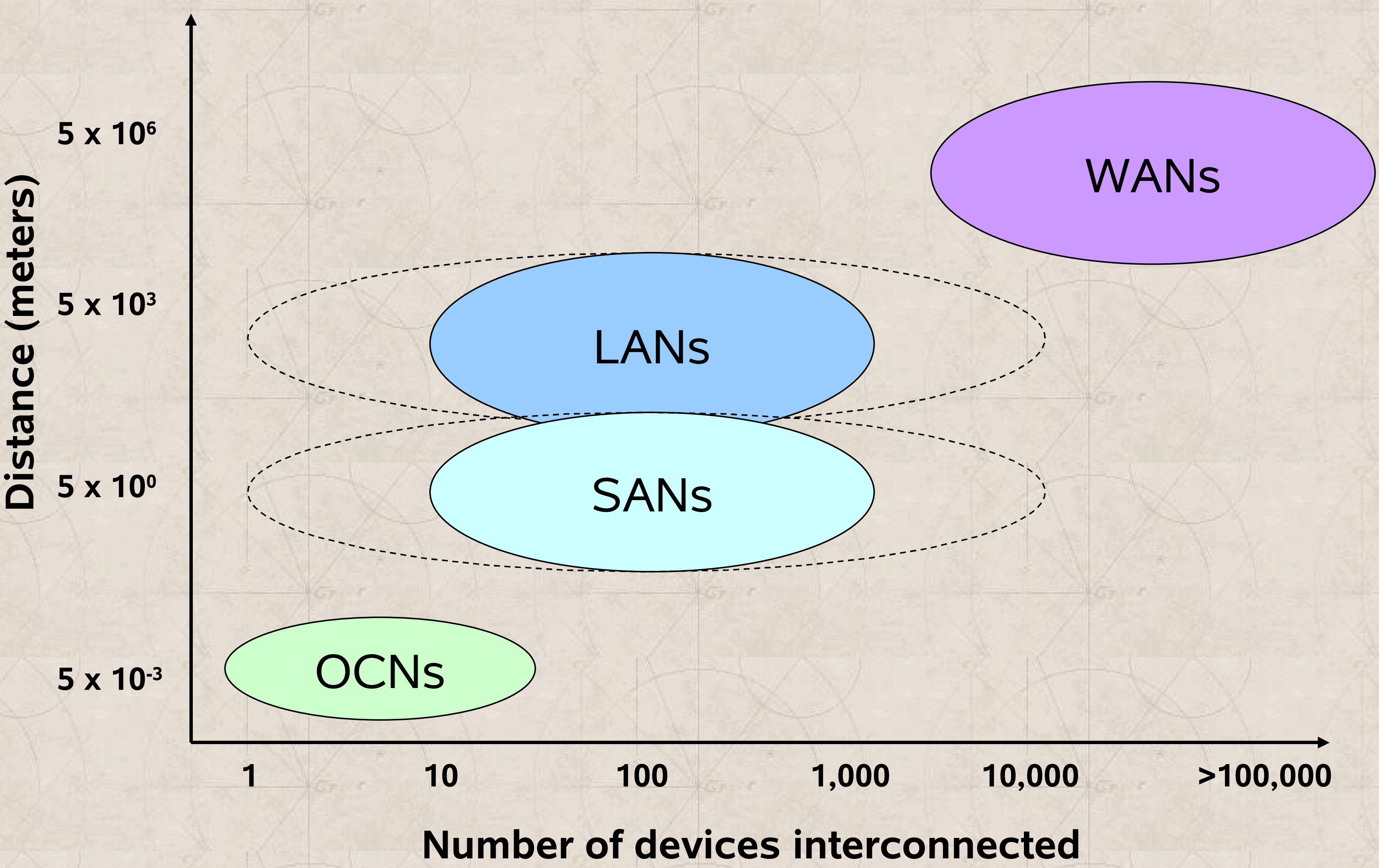
Introduction

Interconnection Network Domains

- *Local area networks* (*LANs*)
  - – Interconnect autonomous computer systems
  - – Machine room or throughout a building or campus
  - – Hundreds of devices interconnected (1,000s with bridging)
  - – Maximum interconnect distance on the order of few kilometers, but some with distance spans of a few tens of kilometers
  - – Hundred devices (thousands with bridging)
  - – Example (most popular): Ethernet, with 10 Gbps over 40Km
- *Wide area networks* (*WANs*)
  - – Interconnect systems distributed across the globe
  - – Internetworking support is required
  - – Many millions of devices interconnected
  - – Maximum interconnect distance of many thousands of kilometers
  - – Example: ATM

Introduction

Interconnection Network Domains



Introduction

Organization

- Top-down Approach
  - We unveil concepts and complexities involved in designing interconnection networks by first viewing the network as an ideal “black box” and then systematically removing various layers of the black box, exposing non-ideal behavior and complexities.
  - We first consider interconnecting only two devices (E.2)
  - We then consider interconnecting many devices (E.3)
  - Other layers of the black box are peeled away, exposing the network topology, routing, arbitration, and switching (E.4, E.5)
  - We then zoom in on the switch microarchitecture (E.6)
  - Next, we consider Practical Issues for Interconnection Networks (E.7)
  - Finally, we look at some examples: OCNs and SANs (E.8)
  - Internetworking (E.9) (skipped)
  - Additional Crosscutting Issues (E.10) (skipped)
  - Fallacies and Pitfalls (E.11)
  - Concluding Remarks (E.12) and References

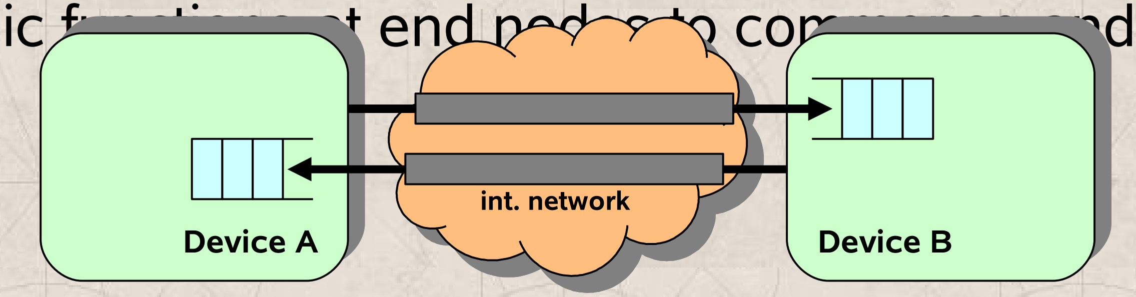
Outline

- E.1 Introduction *(Lecture 1)*
- **E.2 Interconnecting Two Devices** *(Lecture 1)*
    - **An Ideal Network**
    - **Network Interface Functions**
    - **Communication Protocol**
    - **Basic Network Structure and Functions**
    - **Characterizing Performance: Latency & Effective Bandwidth**
    - **Basic Network Characteristics in Commercial Machines**
- E.3 Interconnecting Many Devices *(Lecture 2)*
- E.4 Network Topology *(Lecture 2)*
- E.5 Network Routing, Arbitration, and Switching *(Lecture 3)*
- E.6 Switch Microarchitecture *(Lecture 4)*
- E.7 Practical Issues for Commercial Interconnection Networks *(Lecture 4)*
- E.8 Examples of Interconnection Networks *(Lecture 5)*
- E.9 Internetworking (skipped)
- E.10 Crosscutting Issues for Interconnection Networks (skipped)
- E.11 Fallacies and Pitfalls *(Lecture 5)*
- E.12 Concluding Remarks and References *(Lecture 5)*

Interconnecting Two Devices

An Ideal Network

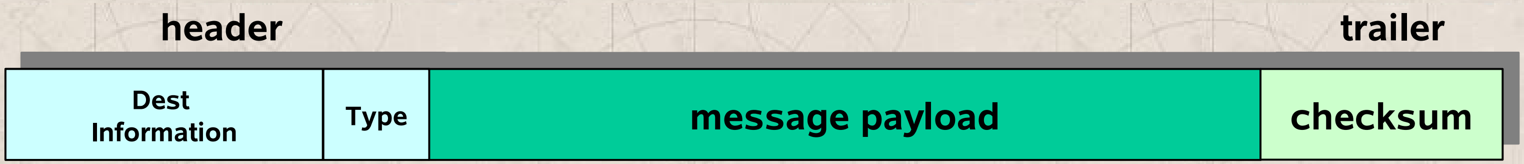
- Two end node devices (A and B)
- Device A (or B) may read an address in device B (or A)
- Interconnection network behaves as *dedicated links* between A, B
  - Unidirectional wires each way dedicated to each device
- Receiving buffers used for staging the transfers at the end nodes
- Communication protocol: *request, reply*
  - basic functions at end nodes to commence and complete comm.



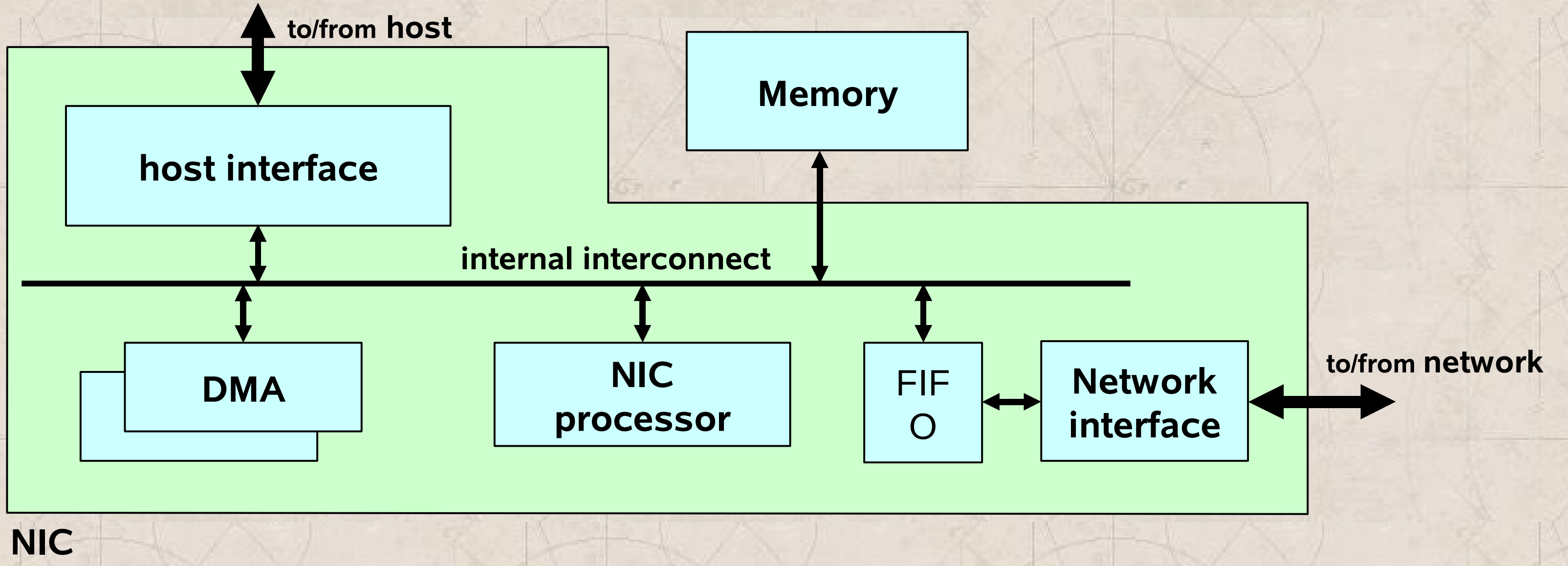
Interconnecting Two Devices

Network Interface Functions

- A *message* is the unit of information: header, payload, trailer



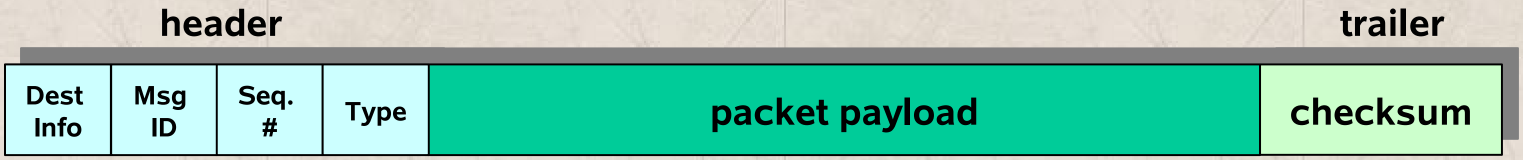
- Interfacing to the network (hardware)
  - Communication device itself (OCNs and some SANs)
  - Additional network interface card or *NIC* (SANs, LANs, WANs)
    - › Embedded processor(s), DMA engine(s), RAM memory



Interconnecting Two Devices

Network Interface Functions

- Interfacing to the network (software, firmware)
  - Translate *requests* and *replies* into messages
  - Tight integration with operating system (OS)
    - › Process isolation, protection mechanisms, etc.
    - › Port (binds sender process with intended receiver process)
  - Packetization
    - › *Maximum transfer unit* (MTU) used for dimensioning resources
    - › Messages larger than MTU divided into *packets* with *message id*
    - › Packet reordering at destination (for message reassembly) is done using *sequence numbers*



00 = request
01 = reply
10 = request acknowledge
11 = reply acknowledge

Interconnecting Two Devices

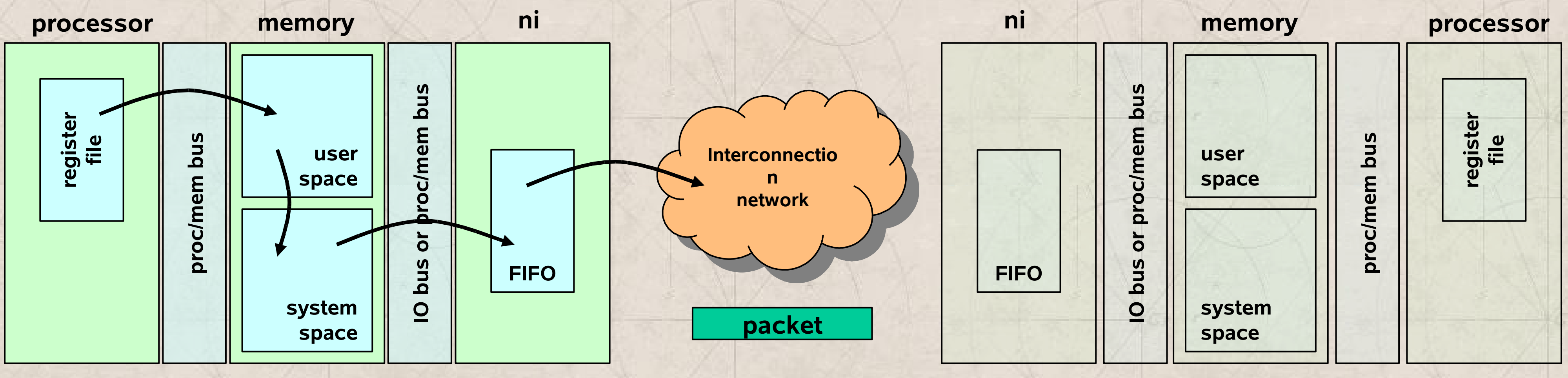
Communication Protocol

- Defines the sequence of steps followed by sender and receiver
- Implemented by a combination of software and hardware
  - Hardware timers, CRC, packet processing, TLBs, ...
  - TCP off-load engines for LANs, WANs
  - OS-bypassing (zero-copy protocols)
    - › Direct allocation of buffers at the network interface memory
    - › Applications directly read/write from/to these buffers
    - › Memory-to-memory copies avoided
    - › Protection guaranteed by OS

Interconnecting Two Devices

Communication Protocol

- Typical steps followed by the *sender*:
  1. System call by application
     - › Copies the data into OS and/or network interface memory
     - › Packetizes the message (if needed)
     - › Prepares headers and trailers of packets
  2. Checksum is computed and added to header/trailer
  3. Timer is started and the network interface sends the packets



user writes data in memory

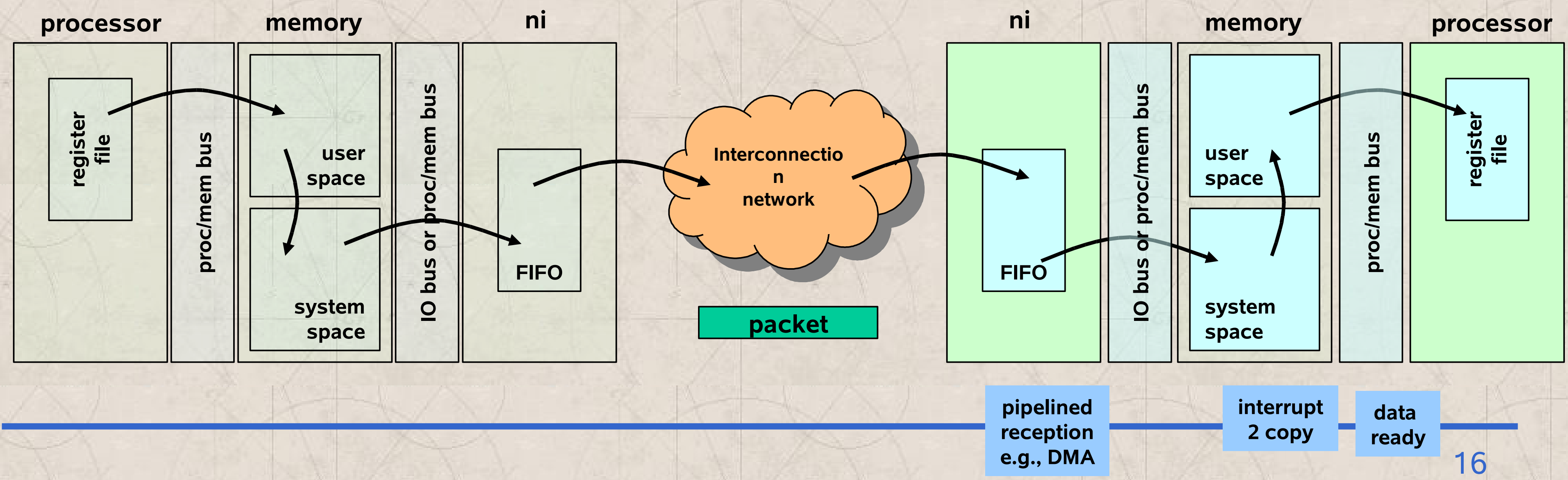
system call sends 1 copy

pipelined transfer e.g., DMA

Interconnecting Two Devices

Communication Protocol

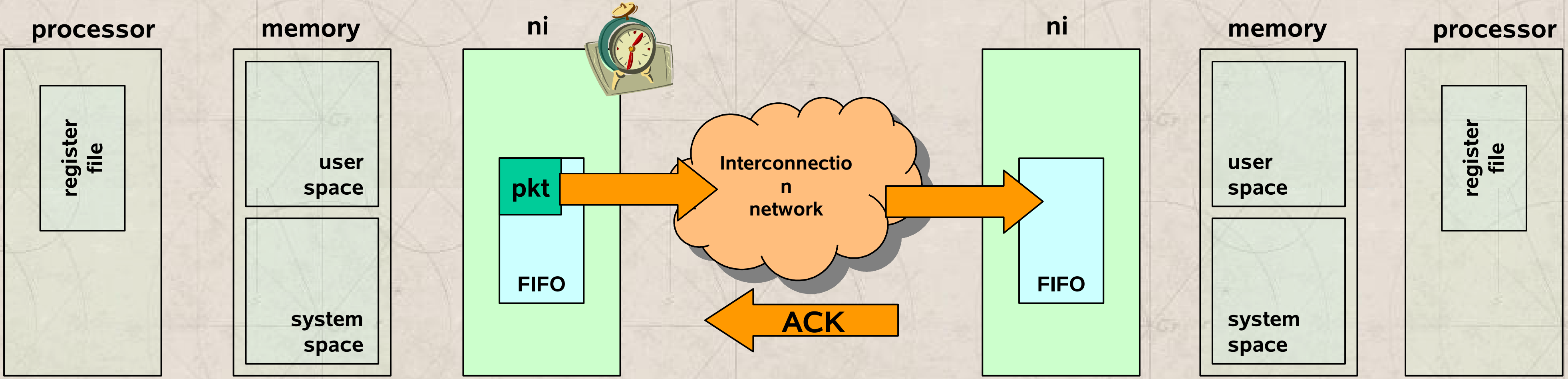
- Typical steps followed by the *receiver*:
  1. NI allocates received packets into its memory or OS memory
  2. Checksum is computed and compared for each packet
     - If checksum matches, NI sends back an ACK packet
  3. Once all packets are correctly received
     - The message is reassembled and copied to user's address space
     - The corresponding application is signalled (via polling or interrupt)



Interconnecting Two Devices

Communication Protocol

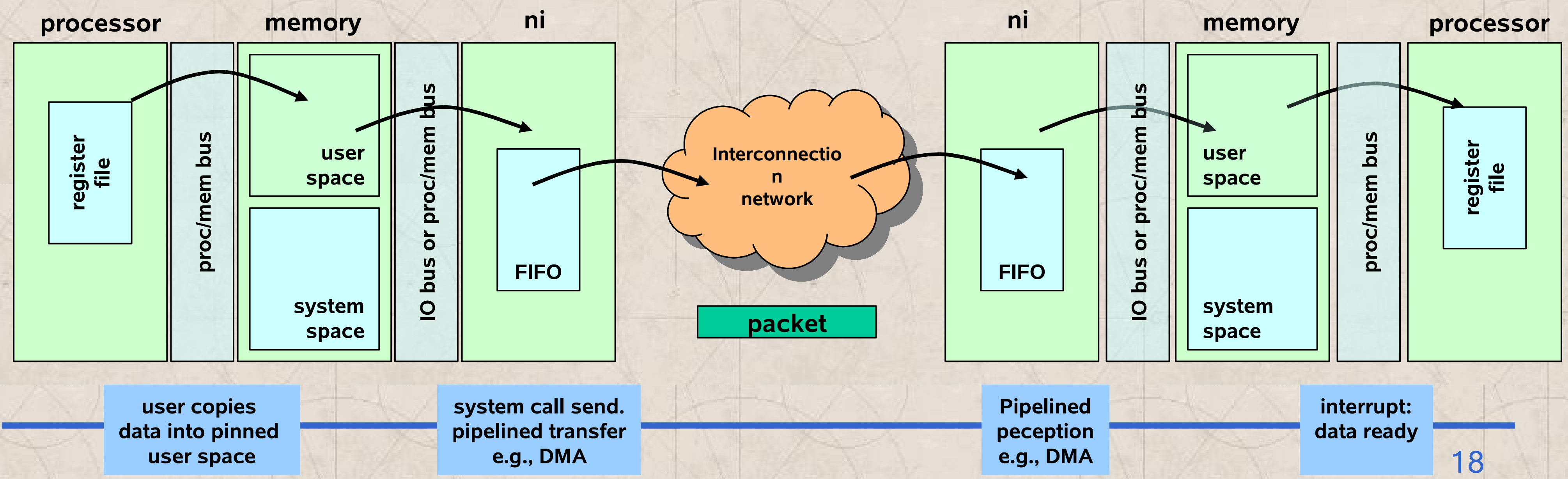
- Additional steps at the sender side:
  - ACK received: the copy is released and timer is cancelled
  - Timer expires before receiving an ACK: packet is resent and the timer is started



Interconnecting Two Devices

Communication Protocol

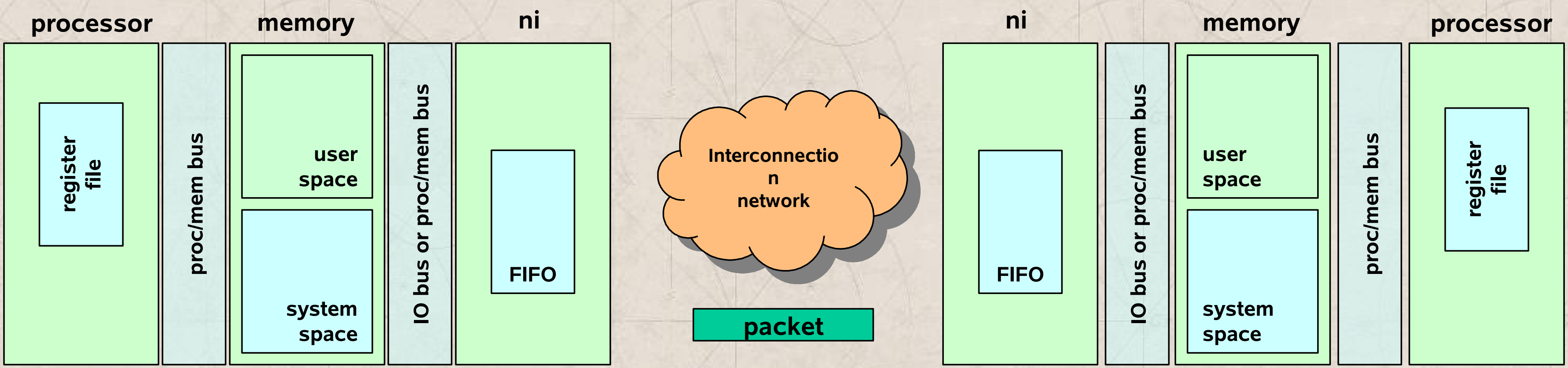
- OS-bypassing (*zero-copy* or user-level protocols)
  - – Direct allocation for DMA transfer to/from memory/NI buffers
  - – Application directly reads/writes from/to these locations
  - – Memory-to-memory copies are avoided
  - – Protection is guaranteed by OS



Interconnecting Two Devices

Communication Protocol

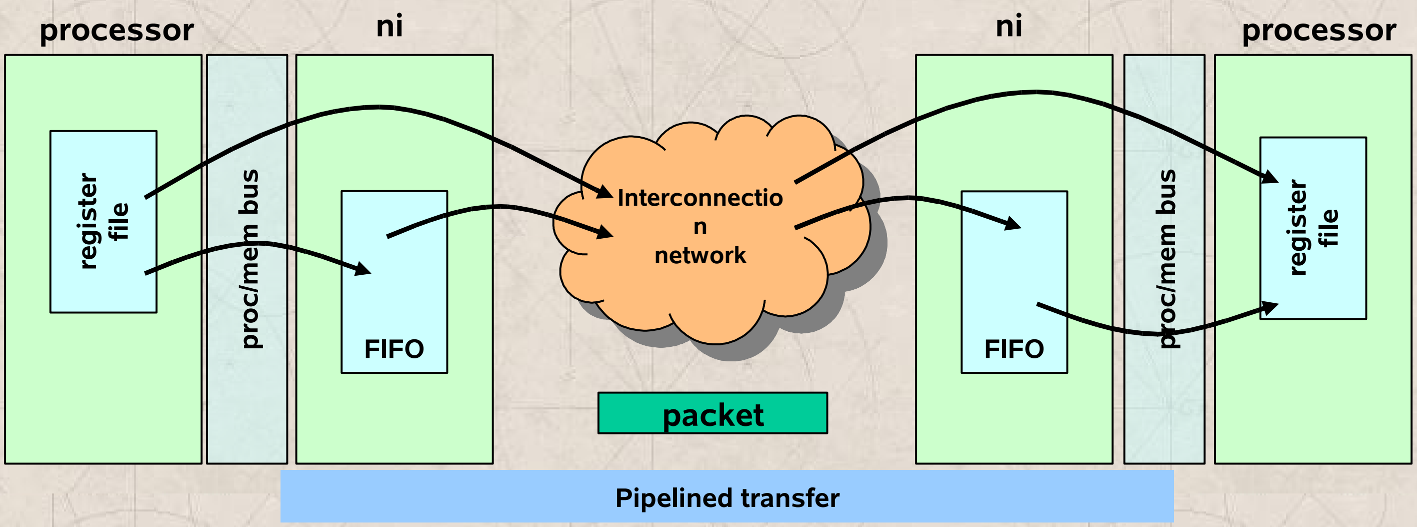
- OS-bypassing (zero-copy or user-level protocols)
  - Direct allocation for DMA transfer to/from memory/NI buffers
  - Application directly reads/writes from/to these locations
  - Memory-to-memory copies are avoided
  - Protection is guaranteed by OS
- *Is it possible to take out register to memory/buffer copy as well?*



Interconnecting Two Devices

Communication Protocol

- OS-bypassing (zero-copy or user-level protocols)
  - Direct allocation of system or network interface memory/buffers
  - Application directly reads/writes from/to these locations
  - Memory-to-memory copies are avoided
  - Protection is guaranteed by OS
- Is it possible to take out register to memory/buffer copy as well?
  - *NI buffer is associated with (or replaced by) register mapping*



Interconnecting Two Devices

Raw’s 5-tuple Model (Taylor, et al. TPDS 2004):

- Defines a figure of merit for *operation-operand matching*
- *End-to-end model*: follows timing from sender to receiver
- 5-tuple: <SO, SL, NHL, RL, RO>
  - SO: Send Occupancy
  - SL: Send Latency
  - NHL: Network Hop Latency
  - RL: Receive Latency
  - RO: Receive Occupancy

• **Conventional distr. SM MP:**	**<10, 30, 5, 30, 40>**
• **Raw / msg passing:**	**< 3, 2, 1, 1, 7>**
• **Raw / scalar:**	**< 0, 1, 1, 1, 0>**
• **ILDP:**	**<0, n, 0, 1, 0>, (n = 0, 2)**
• **Grid:**	**<0, 0, n/8, 0, 0>, (n = 0..8)**
• **Superscalar:**	**< 0, 0, 0, 0, 0>**

“Scalar Operand Networks,” M. B. Taylor, W. Lee, S. P. Amarasinghe, and A. Agarwal, *IEEE Trans. on Parallel and Distributed Systems* , Vol. 16, No. 2, pp. 145–162, February, 2005.

Interconnecting Two Devices

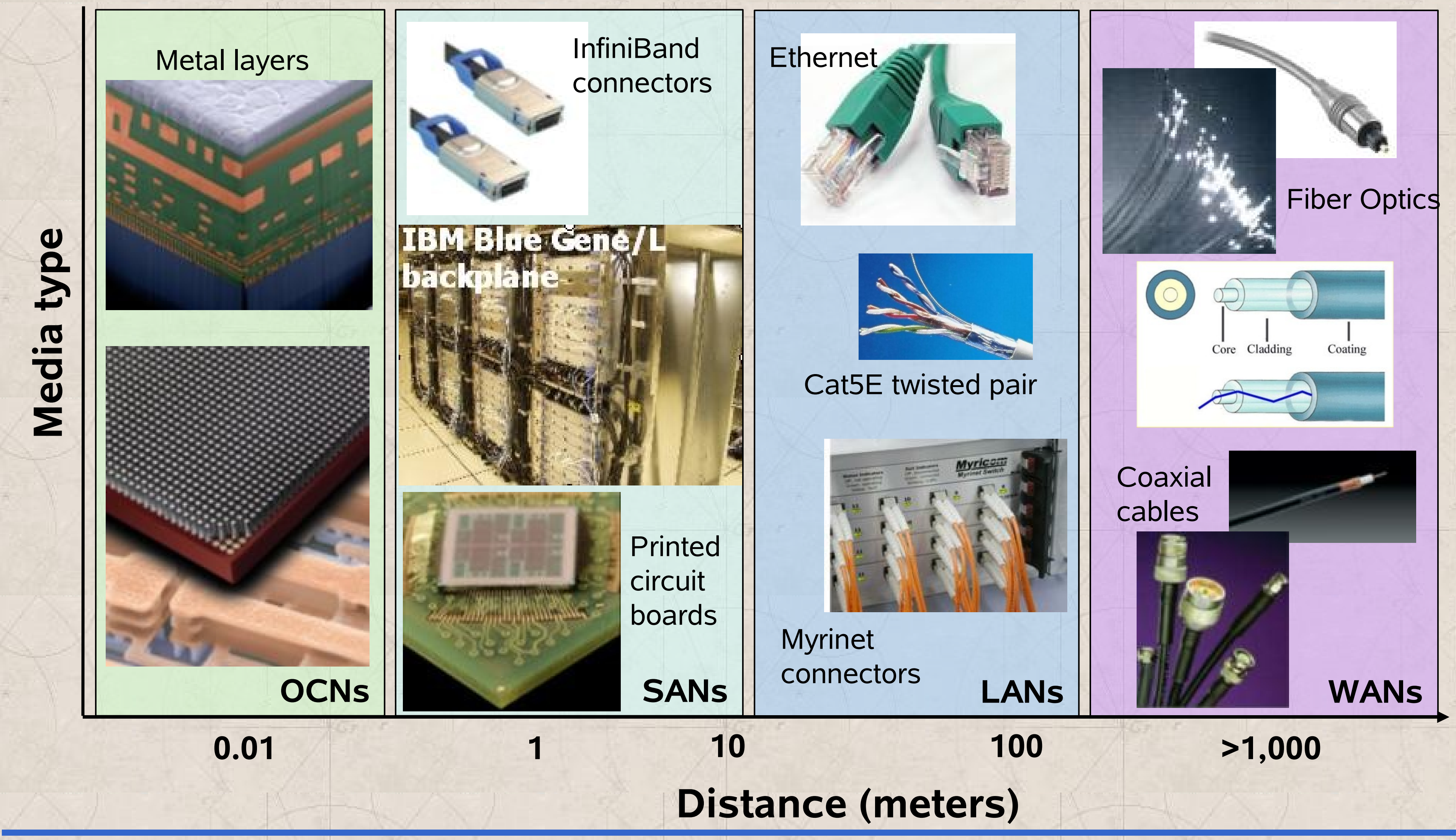
Basic Network Structure and Functions

- *Media* and *Form Factor*
  - largely depends on distance and signaling rate
  - up to centimeters (OCNs)
    - middle and upper copper metal layers at multi-gigabit rates
  - up to meters (SANs)
    - several layers of copper traces or tracks imprinted on circuit boards, midplanes, and backplanes at gigabit rates (differential-pair signaling); Cat 5 unshielded twisted-pair copper wiring
  - 100 meter distances (LANs)
    - Cat 5E unshielded twisted-pair copper wiring at 0.25 Gbps
  - Kilometer distances and beyond (WANs)
    - Coaxial copper cables at 10 Mbps
    - Optical media allows faster transmission speeds
      - Multimode and WDM techniques allow 100s to 1,000s Mbps

Interconnecting Two Devices

Basic Network Structure and Functions

- *Media* and *Form Factor*



Media type

Metal layers

OCNs

InfiniBand connectors

Printed circuit boards

SANs

Ethernet

Cat5E twisted pair

Myrinet connectors

LANs

Fiber Optics

Coaxial cables

WANs

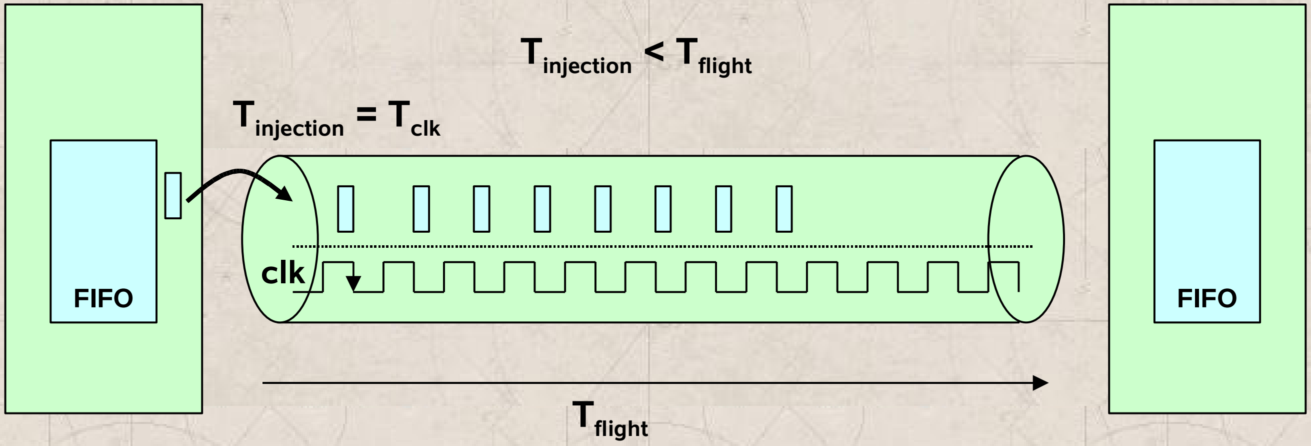
0.01 1 10 100 >1,000

Distance (meters)

Interconnecting Two Devices

Basic Network Structure and Functions

- *Packet Transport*
  - Hardware circuitry needed to drive network links
  - Encoding at the sender and decoding at the receiver
    - Multiple voltage levels, redundancy, data & control rotation (4b/5b)
    - Encoding—along with packet headers and trailers—adds some overhead, which reduces link efficiency
  - Physical layer abstraction:
    - viewed as a long linear pipeline without staging
    - signals propagate as waves through the transmission medium



pipelined transfer

Interconnecting Two Devices

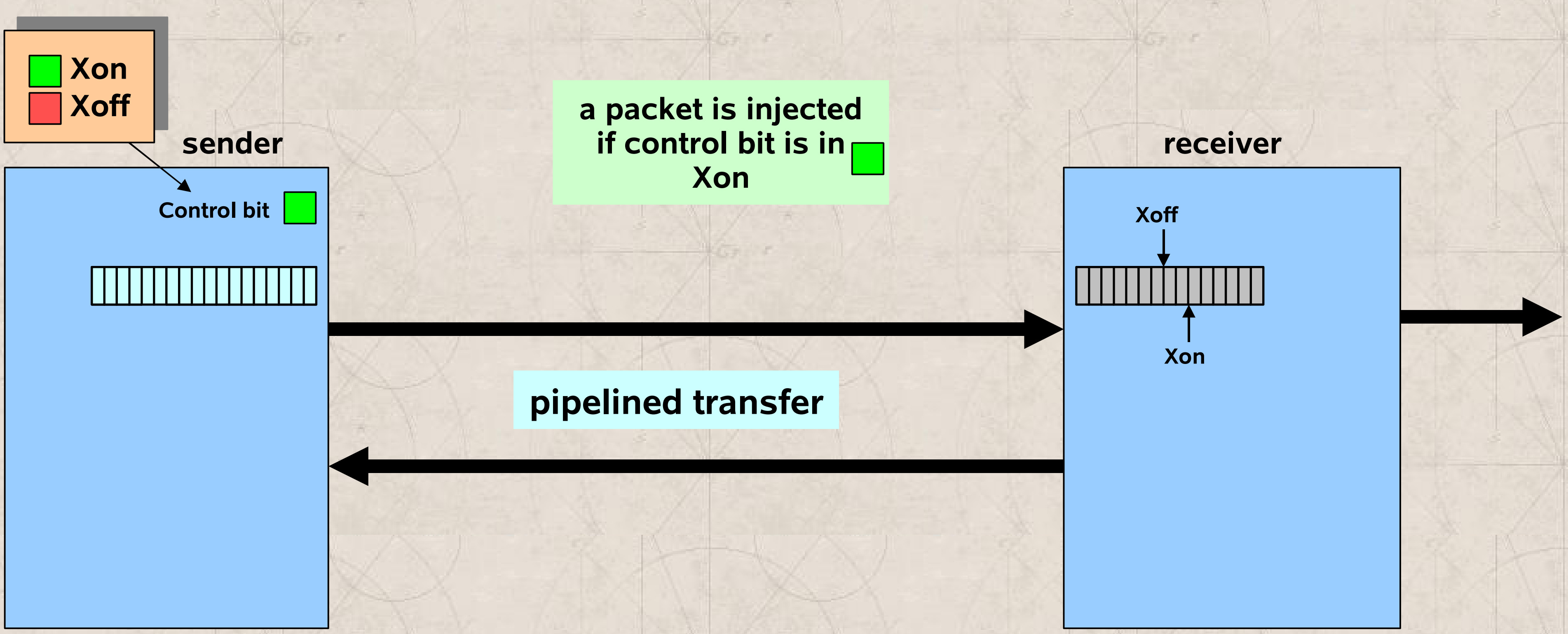
Basic Network Structure and Functions

- *Reliable delivery* of packets: *lossy* versus *lossless* networks
  - Must prevent the sender from sending packets at a faster rate than can be received and processed by the receiver
  - Lossy networks
    - Packets are dropped (discarded) at receiver when buffers fill up
    - Sender is notified to retransmit packets (via time-out or NACK)
  - Lossless networks (flow controlled)
    - Before buffers fill up, sender is notified to stop packet injection
      - *Xon/Xoff* (Stop & Go) flow control
      - *Credit-based* flow control (token or batched modes)
  - Implications of network type (lossless vs. lossy)
    - Constrains the solutions for packet routing, congestion, & deadlock
    - Affects network performance
    - The interconnection network domain dictates which is used
      - OCN, SAN: typically lossless; LAN, WAN: typically lossy

Interconnecting Two Devices

Basic Network Structure and Functions

- Xon/Xoff flow control



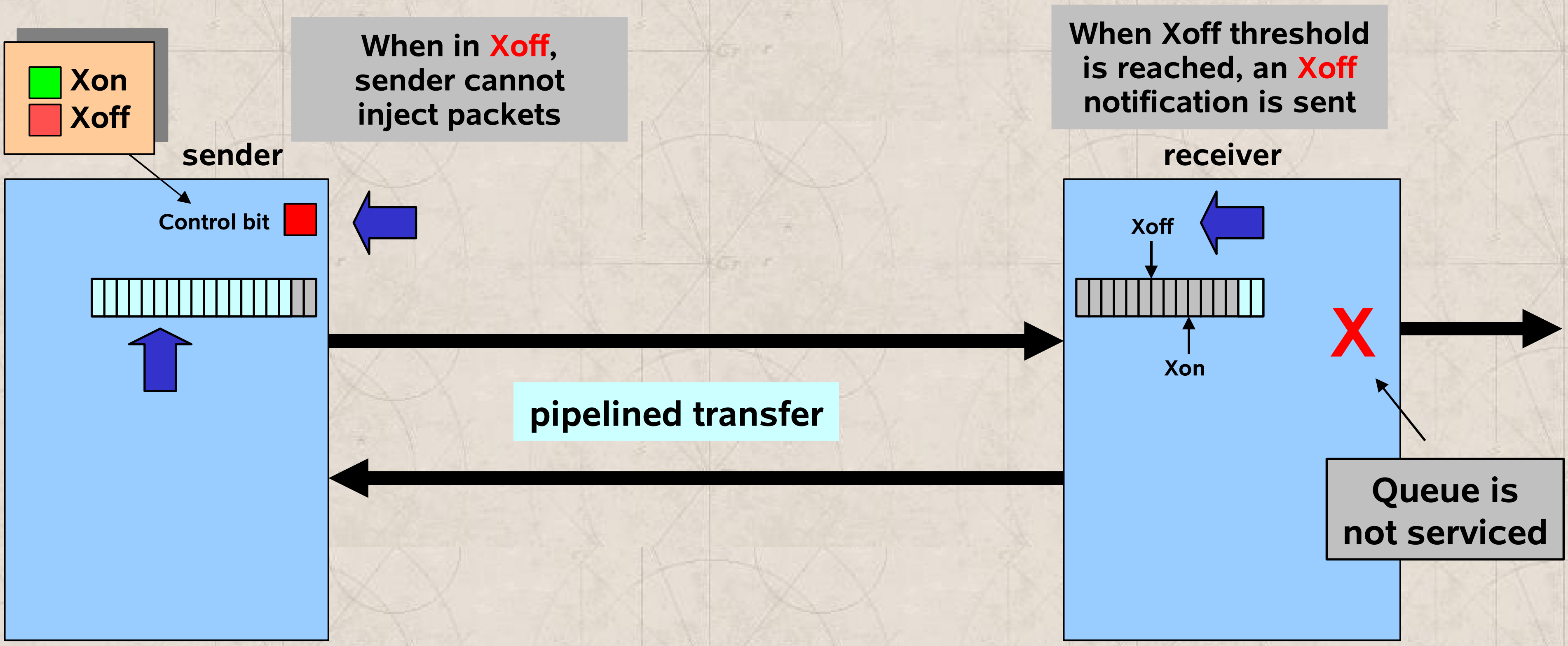
a packet is injected if control bit is in 🟩 Xon

pipelined transfer

Interconnecting Two Devices

Basic Network Structure and Functions

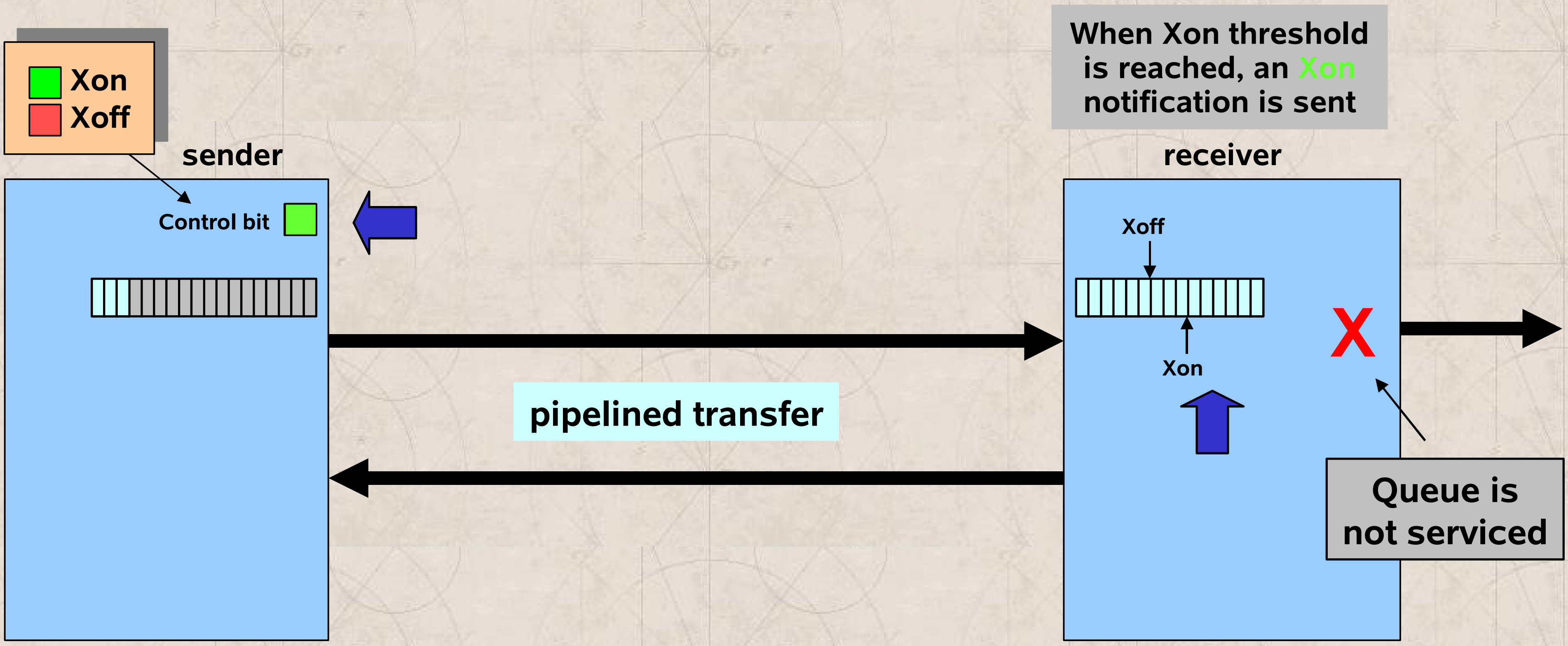
- Xon/Xoff flow control


🟩 Xon
🟥 Xoff
When in Xoff, sender cannot inject packets
When Xoff threshold is reached, an Xoff notification is sent
sender
receiver
Control bit
Xoff
Xon
X
pipelined transfer
Queue is not serviced

Interconnecting Two Devices

Basic Network Structure and Functions

- Xon/Xoff flow control



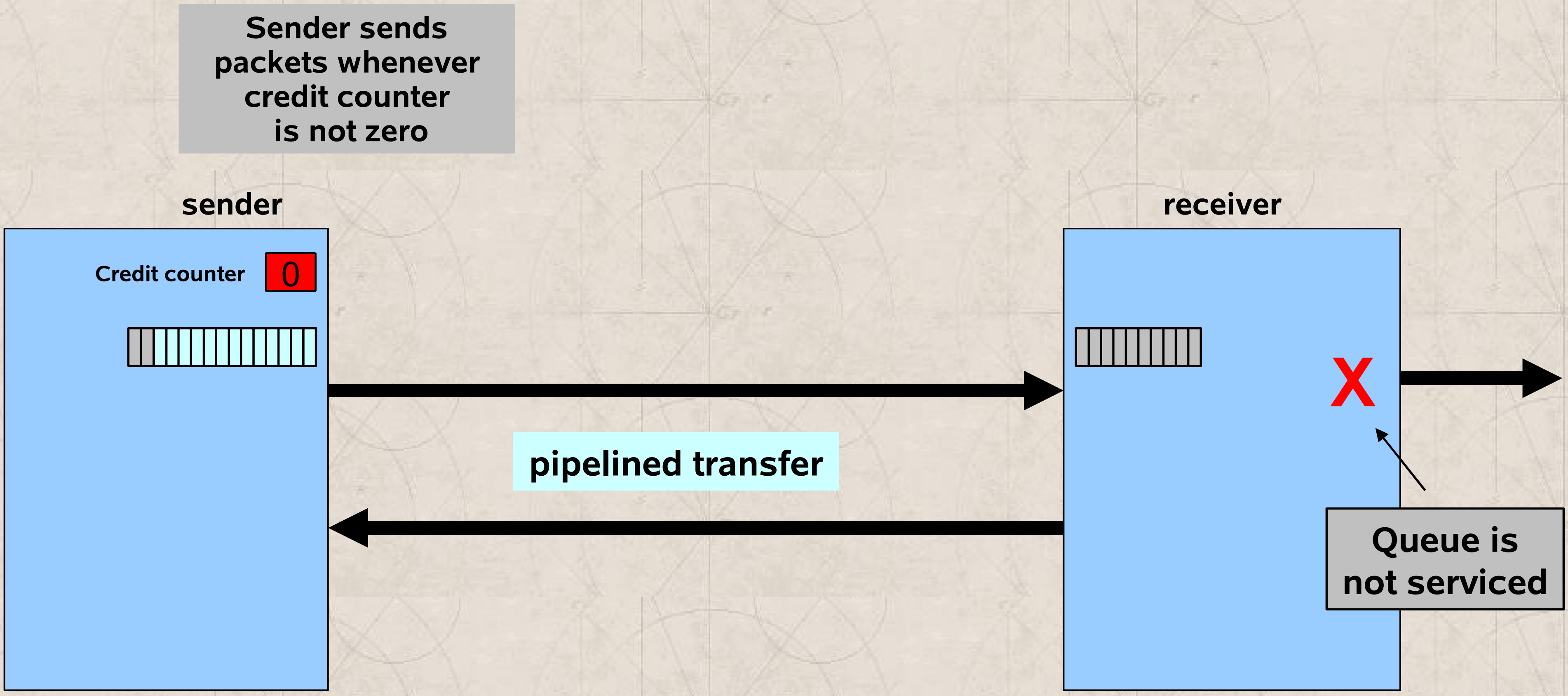
When Xon threshold is reached, an Xon notification is sent

Queue is not serviced

Interconnecting Two Devices

Basic Network Structure and Functions

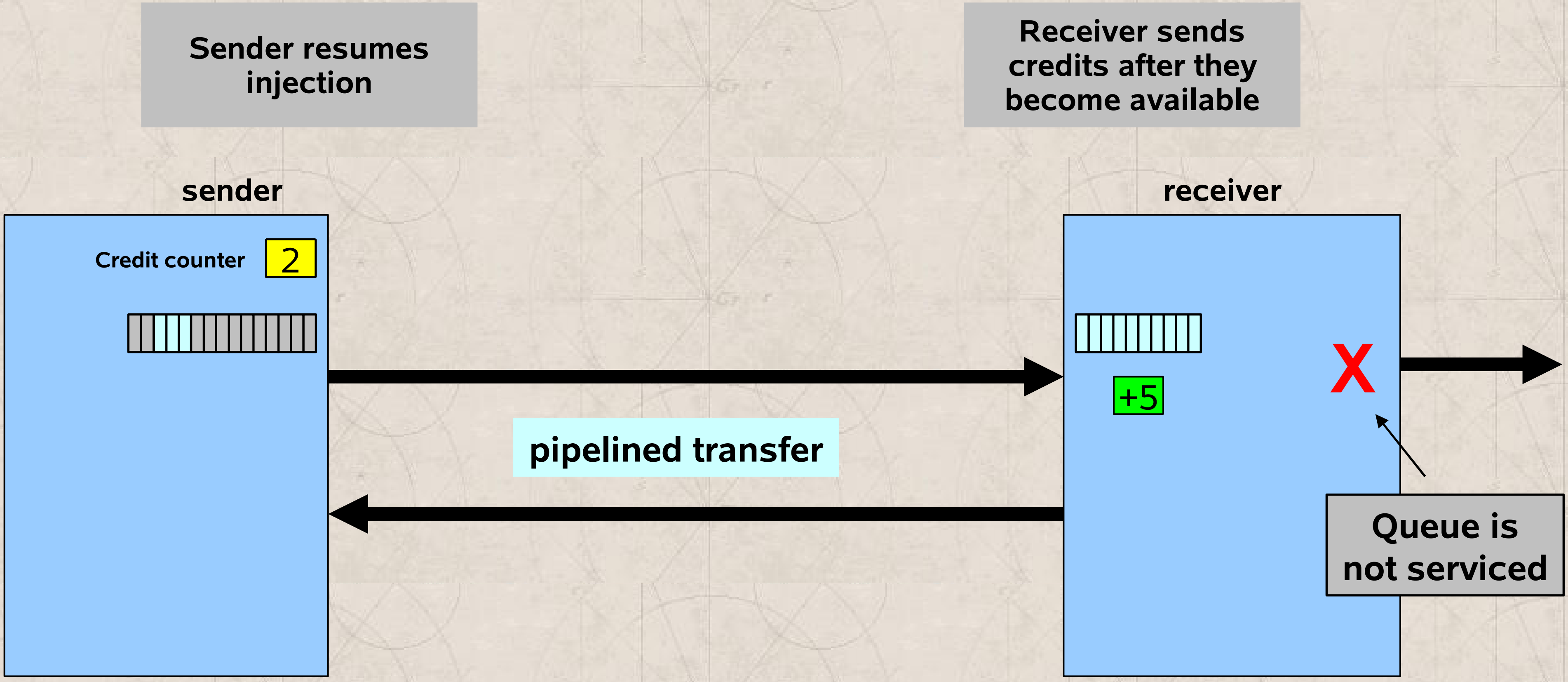
- Credit-based flow control



Interconnecting Two Devices

Basic Network Structure and Functions

- Credit-based flow control



Sender resumes injection

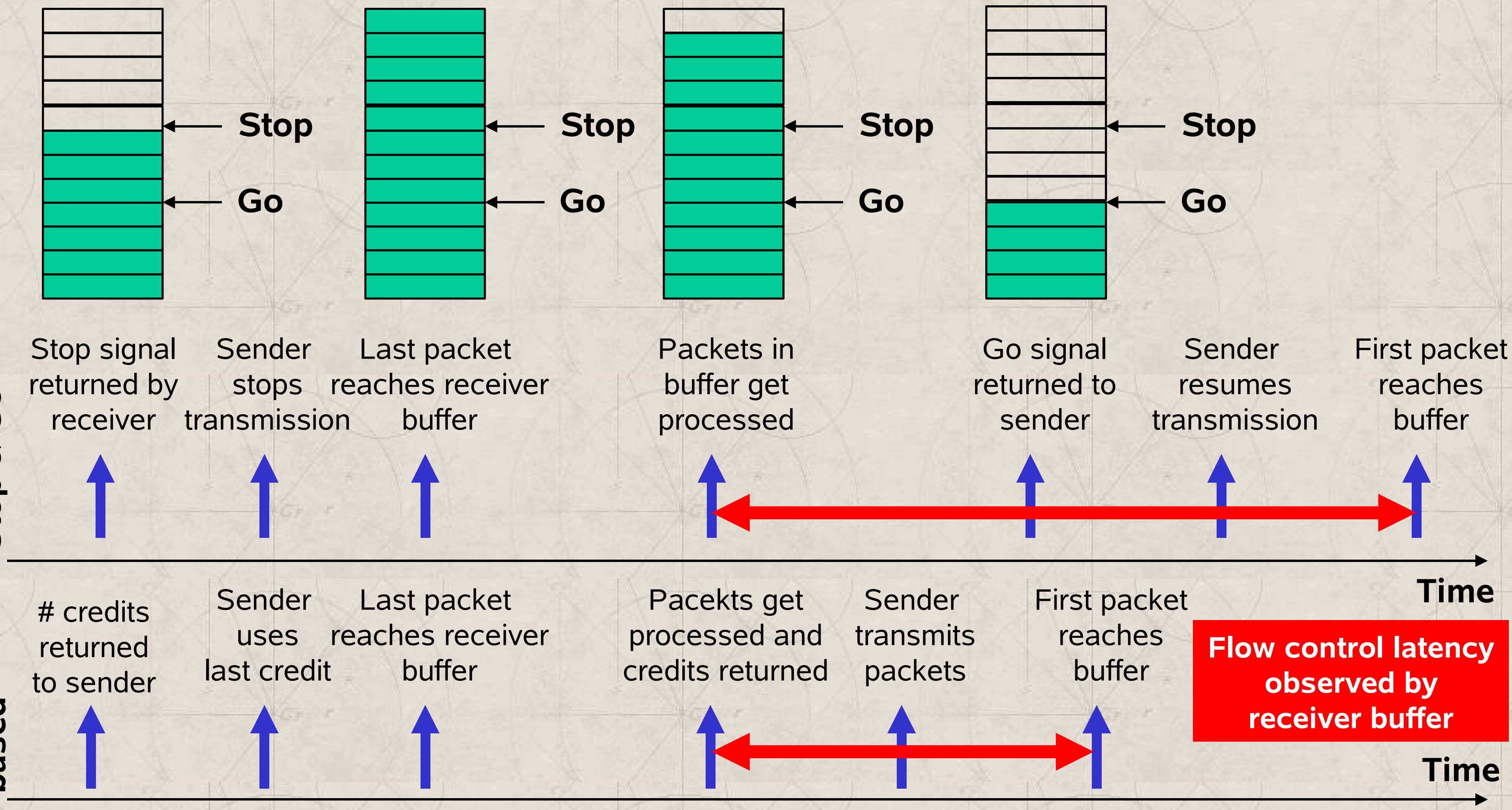
Receiver sends credits after they become available

Queue is not serviced

Interconnecting Two Devices

Basic Network Structure and Functions

- Comparison of Xon/Xoff vs credit-based flow control



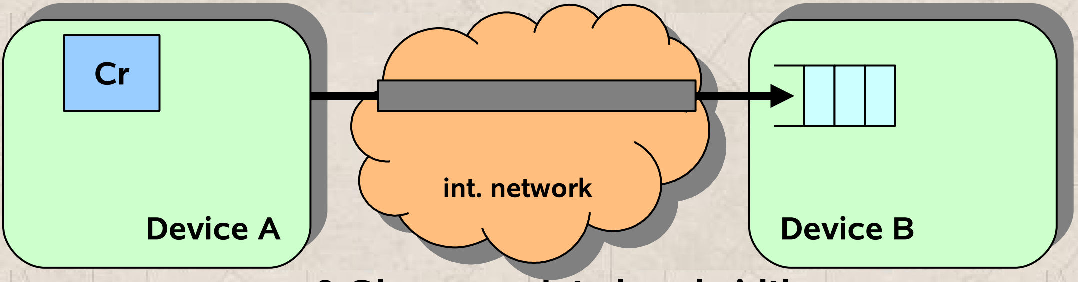
Stop
Go
Stop
Go
Stop
Go
Stop
Go
Stop & Go
Stop signal returned by receiver
Sender stops transmission
Last packet reaches receiver buffer
Packets in buffer get processed
Go signal returned to sender
Sender resumes transmission
First packet reaches buffer
Time
Credit based
# credits returned to sender
Sender uses last credit
Last packet reaches receiver buffer
Pacekts get processed and credits returned
Sender transmits packets
First packet reaches buffer
Flow control latency observed by receiver buffer
Time

➔ ***Poor flow control can reduce <u>link efficiency</u>***

Interconnecting Two Devices

Basic Network Structure and Functions

- Example: comparison of flow control techniques
  - – Calculate the minimum amount of credits and buffer space for interconnect distances of *1 cm*, *1 m*, *100 m*, and *10 km*
  - – Assume a dedicated-link network with
    - › 8 Gbps (raw) data bandwidth per link (each direction)
    - › Credit-based flow control
  - – Device A continuously sends 100-byte packets (header included)
  - – Consider only the link propagation delay (no other delays or overheads)



8 Gbps raw data bandwidth

Interconnecting Two Devices

Basic Network Structure and Functions

- Example: comparison of flow control techniques

$$\textit{Packet propagation delay} + \textit{Credit propagation delay} \le \frac{\textit{Packet size}}{\textit{Bandwidth}} \times \textit{Credit count}$$

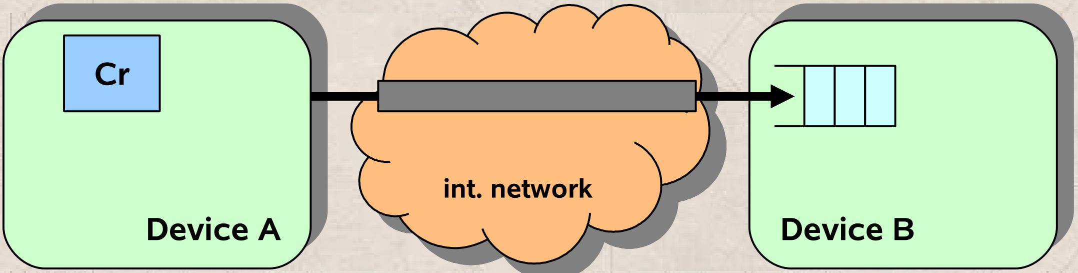
$$\left(\frac{\textit{distance}}{2/3 \times 300{,}000 \text{ km/sec}}\right) \times 2 \le \frac{100 \text{ bytes}}{8 \text{ Gbits/sec}} \times \textit{Credit count}$$

1 cm → 1 credit

1 m → 1 credit

100 m → 10 credits

10 km → 1,000 credits



8 Gbps raw data bandwidth

Interconnecting Two Devices

Basic Network Structure and Functions

- *Error handling*: must detect and recover from transport errors
  - *Checksum* added to packets
  - *Timer* and *ACK* per packet sent
- Additional basic functionality needed by the protocol
  - Resolve duplicates among packets
  - Byte ordering (Little Endian or Big Endian)
  - Pipelining across operations

Interconnecting Two Devices

Characterizing Performance: Latency & Effective Bandwidth

Terms and Definitions:

- *Bandwidth*:
  - Maximum rate at which information can be transferred (including packet header, payload and trailer)
  - Unit: bits per second (bps) or bytes per second (Bps)
  - *Aggregate bandwidth*: Total data bandwidth supplied by network
  - *Effective bandwidth* (*throughput*): fraction of aggregate bandwidth that gets delivered to the application
- *Time of flight*: Time for first bit of a packet to arrive at the receiver
  - Includes the time for a packet to pass through the network, not including the transmission time (defined next)
  - Picoseconds (OCNs), nanoseconds (SANs), microseconds (LANs), milliseconds (WANs)

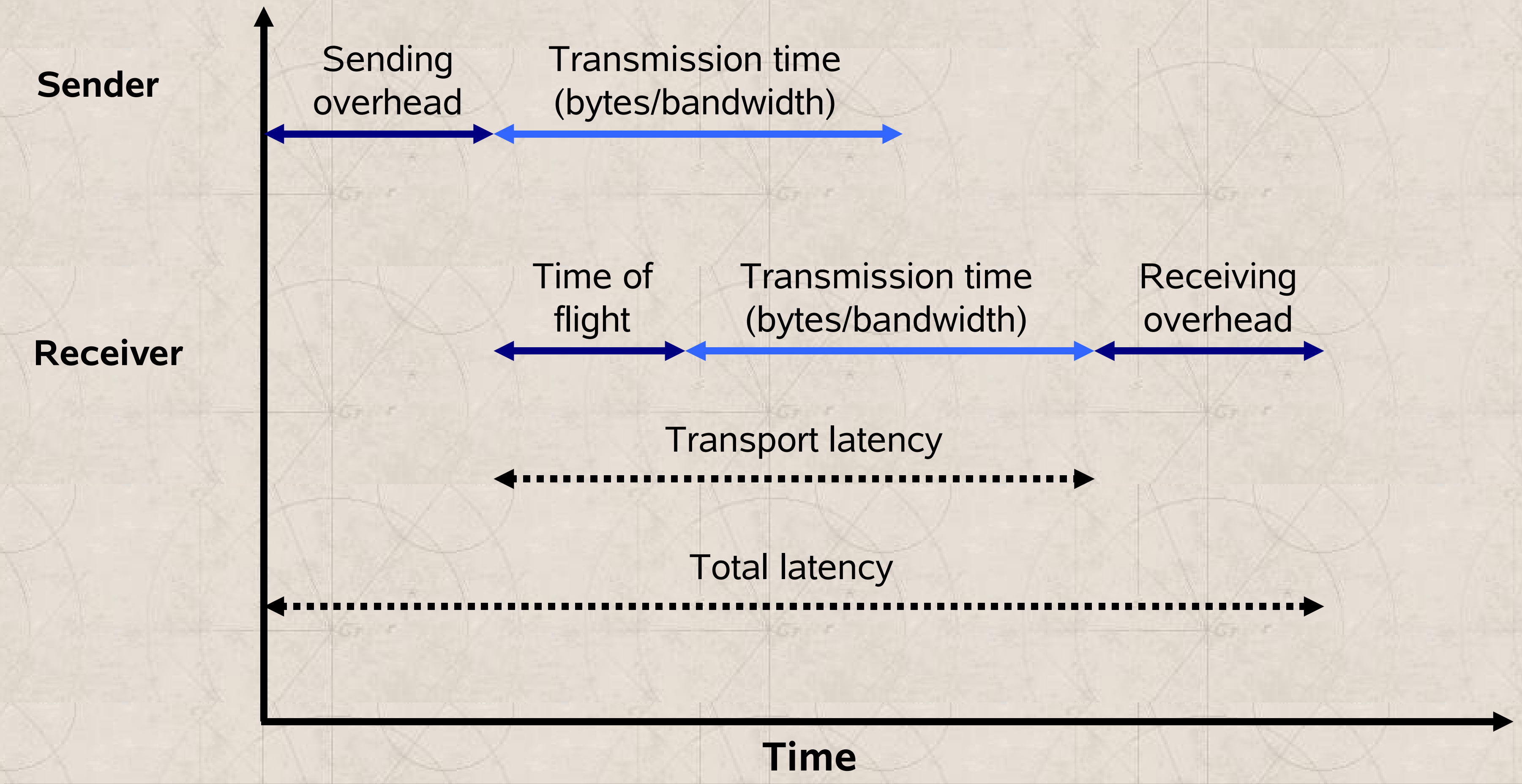
Interconnecting Two Devices

Characterizing Performance: Latency & Effective Bandwidth

- *Transmission time*:
  - The time for a packet to pass through the network, not including the time of flight
  - Equal to the packet size divided by the data bandwidth of the link
- *Transport latency*:
  - Sum of the time of flight and the transmission time
  - Measures the time that a packet spends in the network
- *Sending overhead (latency)*:
  - Time to prepare a packet for injection, including hardware/software
  - A constant term (packet size) plus a variable term (buffer copies)
- *Receiving overhead (latency)*:
  - Time to process an incoming packet at the end node
  - A constant term plus a variable term
  - Includes cost of interrupt, packet reorder and message reassembly

Interconnecting Two Devices

Characterizing Performance: Latency & Effective Bandwidth

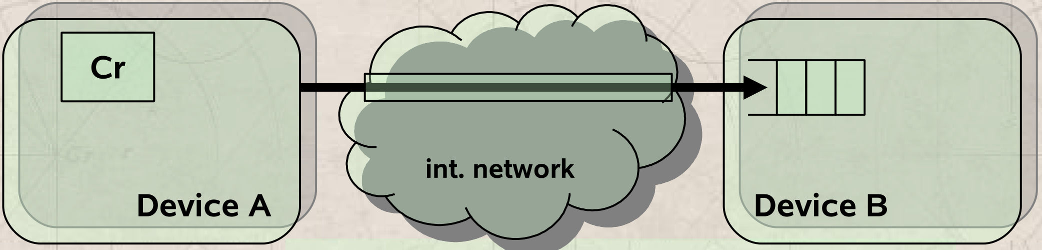


$$\textbf{Latency} = \textit{Sending Overhead} + \textit{Time of flight} + \frac{\textit{packet size}}{\textit{Bandwidth}} + \textit{Receiving Overhead}$$

Interconnecting Two Devices

Characterizing Performance: Latency & Effective Bandwidth

- Example (latency): calculate the total packet latency for interconnect distances of *0.5 cm*, *5 m*, *5,000 m*, and *5,000 km*
  – Assume a dedicated-link network with
    › 8 Gbps (raw) data bandwidth per link, credit-based flow control
  – Device A sends 100-byte packets (header included)
  – Overheads
    › Sending overhead: *x* + 0.05 ns/byte
    › Receiving overhead: 4/3(*x*) + 0.05 ns/byte
    › *x* is 0 µs for OCN, 0.3 µs for SAN, 3 µs for LAN, 30 µs for WAN
  – Assume time of flight consists only of link propagation delay (no other sources of delay)



8 Gbps raw data bandwidth per link

Interconnecting Two Devices

Characterizing Performance: Latency & Effective Bandwidth

- Example (latency):

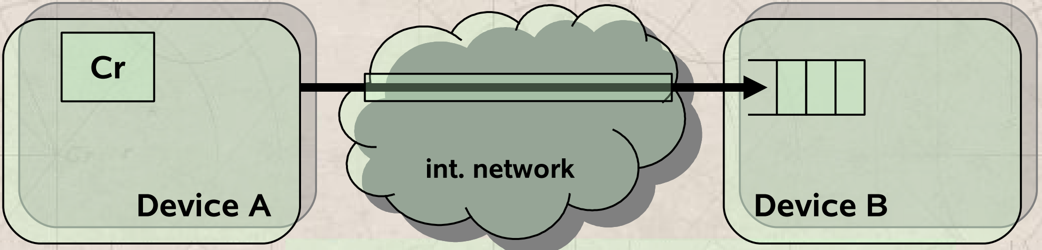
$$\textit{Latency} = \textit{Sending overhead} + \textit{Time of flight} + \frac{\textit{Packet size}}{\textit{Bandwidth}} + \textit{Receiving overhead}$$

$$\textit{Latency}_{OCN} = 5\ ns + 0.025\ ns + 100\ ns + 5\ ns = 110.025\ ns$$

$$\textit{Latency}_{SAN} = 0.305\ \mu s + 0.025\ ns + 0.1\ \mu s + 0.405\ \mu s = 0.835\ \mu s$$

$$\textit{Latency}_{LAN} = 3.005\ \mu s + 25\ \mu s + 0.1\ \mu s + 4.005\ \mu s = 32.11\ \mu s$$

$$\textit{Latency}_{WAN} = 20.05\ \mu s + 25\ \mu s + 0.1\ \mu s + 40.05\ \mu s = 25.07\ ms$$

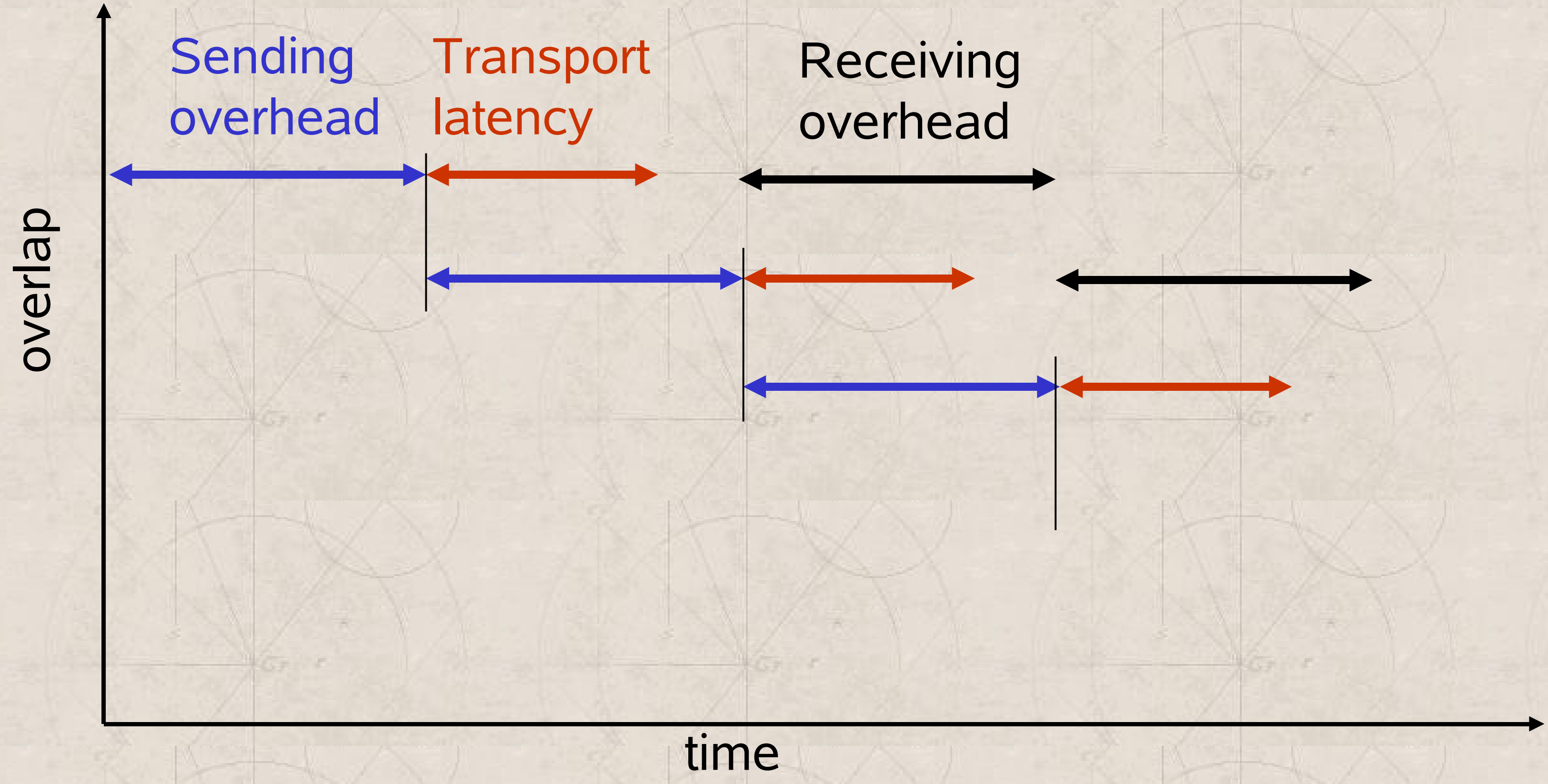


8 Gbps raw data bandwidth per link

Interconnecting Two Devices

Characterizing Performance: Latency & Effective Bandwidth

- *Effective bandwidth* with *link pipelining*
  - – Pipeline the flight and transmission of packets over the links
  - – Overlap the sending overhead with the transport latency and receiving overhead of prior packets



"The Impact of Pipelined Channels on *k*-ary *n*-cube Networks ," S. L. Scott and J. Goodman, *IEEE Trans. on Parallel and Distributed Systems*, Vol. 5, No. 1, pp. 1–16, January, 1994.

Interconnecting Two Devices

Characterizing Performance: Latency & Effective Bandwidth

- *Effective bandwidth* with *link pipelining*
  - – Pipeline the flight and transmission of packets over the links
  - – Overlap the sending overhead with the transport latency and receiving overhead of prior packets

$$BW_{LinkInjection} = \frac{\text{Packet size}}{\max(\text{sending overhead, transmission time})}$$

$$BW_{LinkReception} = \frac{\text{Packet size}}{\max(\text{receiving overhead, transmission time})}$$

$$\text{Eff. bandwidth} = \min(2xBW_{LinkInjection}, 2xBW_{LinkReception}) = \frac{2 \text{ x Packet size}}{\max(\text{overhead, transmission time})}$$

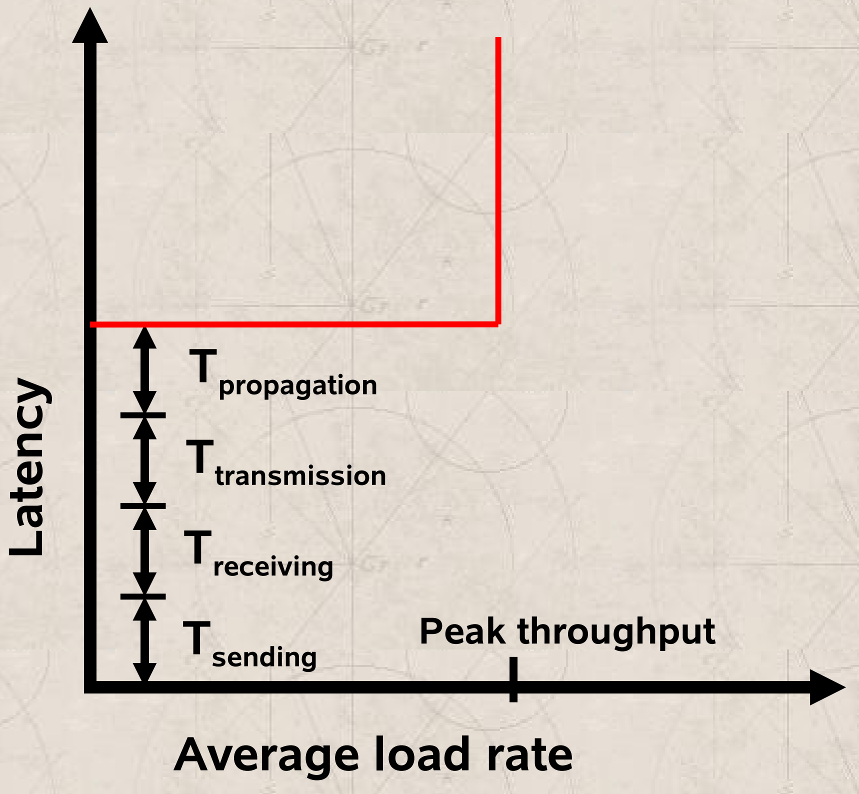
(only two devices)

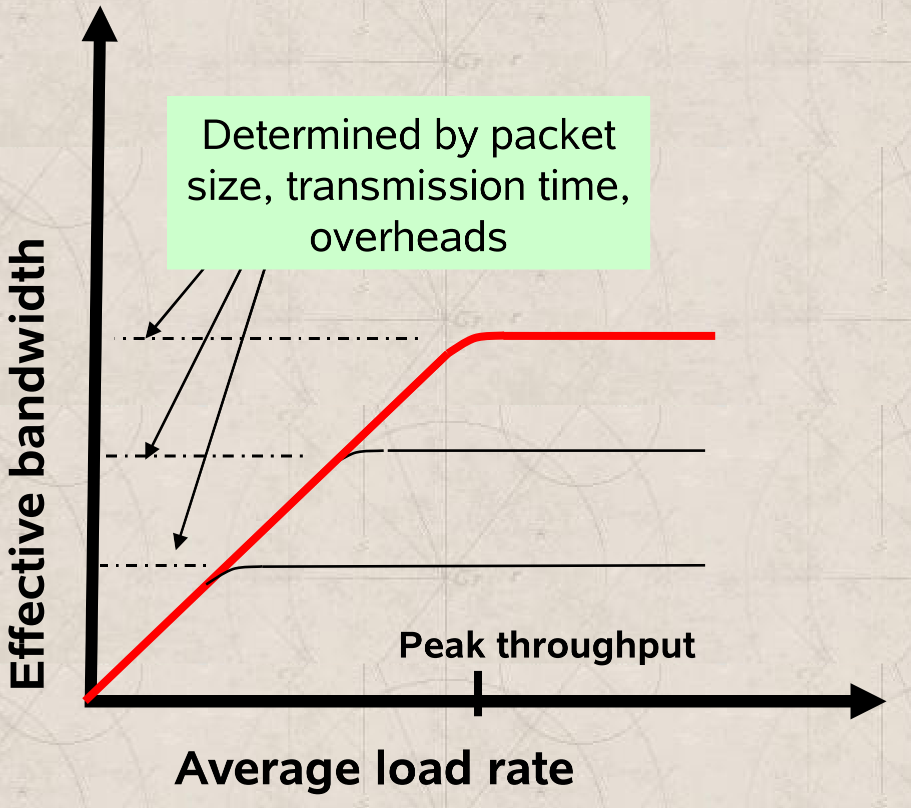
$$\text{overhead} = \max(\text{sending overhead, receiving overhead})$$

Interconnecting Two Devices

Characterizing Performance: Latency & Effective Bandwidth

- *Characteristic performance plots*: latency vs. average load rate; throughput (effective bandwidth) vs. average load rate





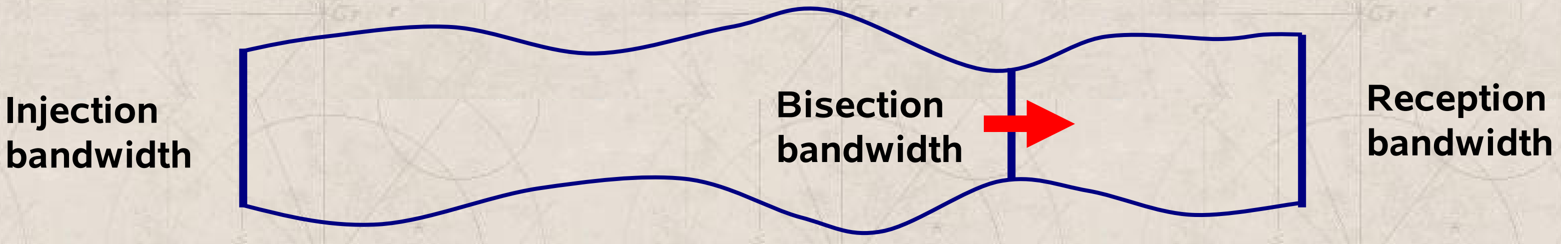
Determined by packet size, transmission time, overheads

Interconnecting Two Devices

Characterizing Performance: Latency & Effective Bandwidth

A Simple (General) Throughput Performance Model:

- The network can be considered as a "*pipe*" of variable width



- There are three points of interest *end-to-end*:
  - Injection into the pipe
  - Narrowest section within pipe (i.e., minimum network bisection that has traffic crossing it)
  - Reception from the pipe
- *The bandwidth at the narrowest point **and utilization of that bandwidth** determines the throughput!!!*

Interconnecting Two Devices

Characterizing Performance: Latency & Effective Bandwidth

A Simple (General) Throughput Performance Model:

$$\begin{aligned} \text{Effective bandwidth} &= \min(BW_{NetworkInjection}, BW_{Network}, \sigma \times BW_{NetworkReception}) \\ &= \min(N \times BW_{LinkInjection}, BW_{Network}, \sigma \times N \times BW_{LinkReception}) \end{aligned}$$

$$BW_{Network} = \rho \times \frac{BW_{Bisection}}{\gamma}$$

σ is the *ave. fraction of traffic to reception links that can be accepted* (captures contention at reception links due to application behavior)

γ is the *ave. fraction of traffic that <u>must</u> cross the network bisection*

ρ is the *network efficiency*, which mostly depends on other factors: *link efficiency, routing efficiency, arb. efficiency, switching eff., etc.*

$BW_{Bisection}$ *is minimum network bisection that has traffic crossing it*

Interconnecting Two Devices

Characterizing Performance: Latency & Effective Bandwidth

Simple (General) Model Applied to Interconnecting ***Two Devices***:

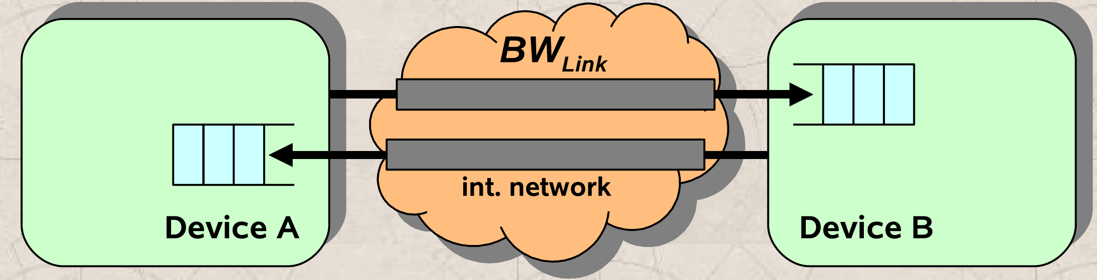
$$\text{Effective bandwidth} = \min(BW_{NetworkInjection}, BW_{Network}, \sigma \times BW_{NetworkReception})$$

$$= \min(2 \times BW_{LinkInjection}, BW_{Network}, 1 \times (2 \times BW_{LinkReception}))$$

$$BW_{Network} = \rho_L \times \frac{2 \times BW_{Link}}{1}$$

ρ_L = ***link efficiency*** resulting from *flow control, encoding, packet header and trailer overheads*

Dedicated-link network

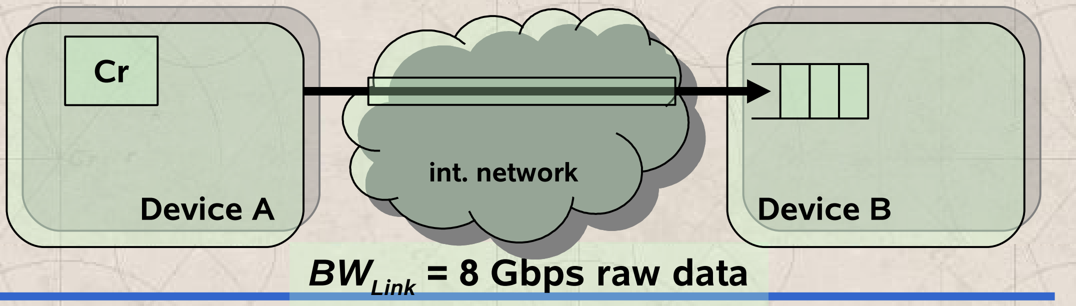


Interconnecting Two Devices

Characterizing Performance: Latency & Effective Bandwidth

- Example: plot the effective bandwidth versus packet size
  - Assume a dedicated-link network with
    - 8 Gbps (raw) data bandwidth per link, credit-based flow control
  - Packets range in size from 4 bytes to 1500 bytes
  - Overheads
    - Sending overhead: *x* + 0.05 ns/byte
    - Receiving overhead: 4/3(*x*) + 0.05 ns/byte
    - *x* is 0 μs for OCN, 0.3 μs for SAN, 3 μs for LAN, 30 μs for WAN
  - What limits the effective bandwidth?
  - For what packet sizes is 90% of the aggregate bandwidth utilized?

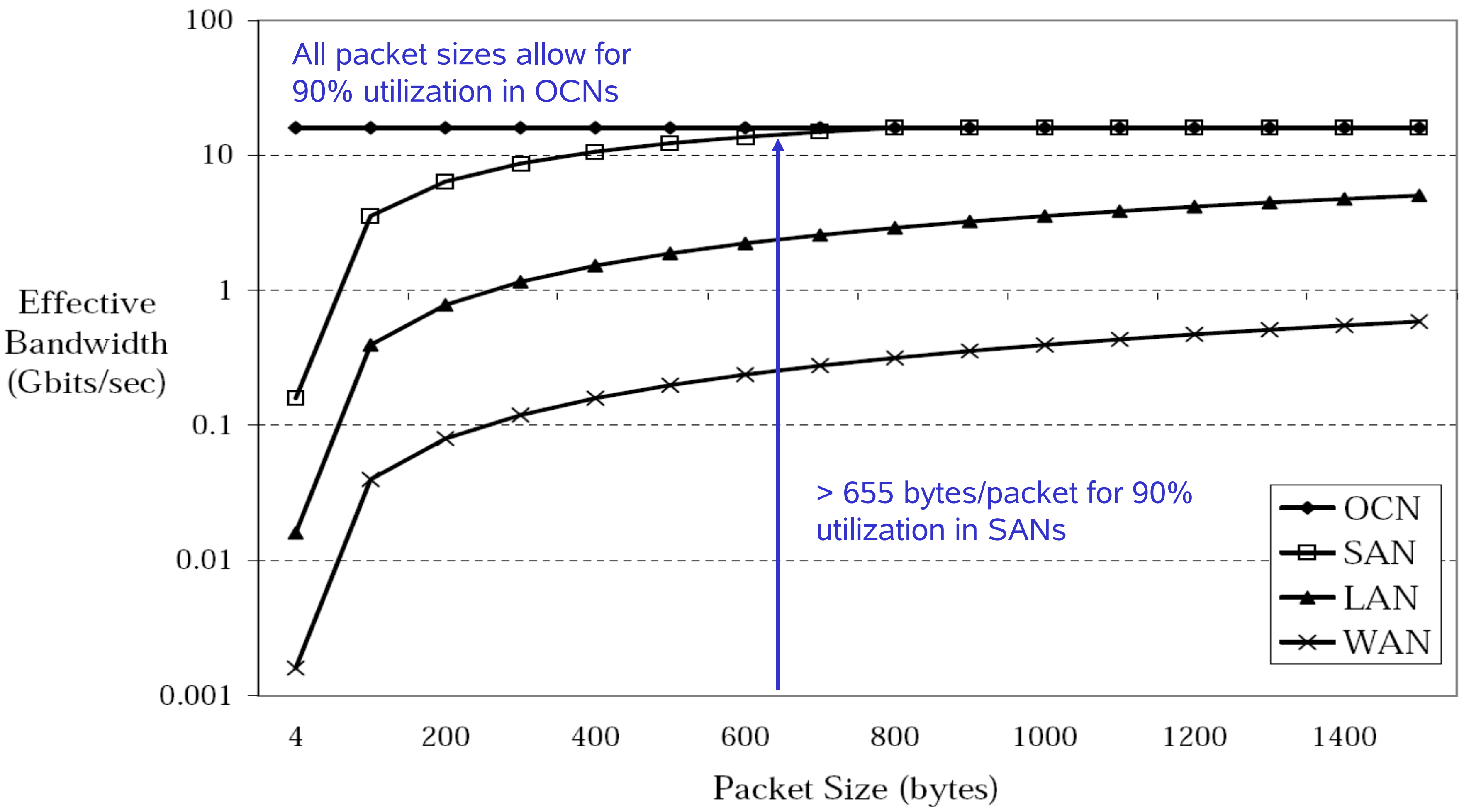
Dedicated-link network



Interconnecting Two Devices

Characterizing Performance: Latency & Effective Bandwidth

- Example: plot the effective bandwidth versus packet size



All packet sizes allow for 90% utilization in OCNs

> 655 bytes/packet for 90% utilization in SANs

Transmission time *is the limiter for OCNs;* ***overhead*** *limits SANs for packets sizes < 800 bytes*

Interconnecting Two Devices

Basic Network Characteristics of Commercial Machines

Company	System [Network] Name	Intro year	Max. compute nodes [x # CPUs]	System footprint for max. configuration	Packet [header] max. size	Injection [Recept'n] node BW in Mbytes/s	Minimum send/rec overhead	Maximum copper link length; flow control; error
Intel	ASCI Red Paragon	2001	4,510 [x 2]	2,500 sq. feet	1984 B [4 B]	400 [400]	few µsec	handshaking; CRC+parity
IBM	ASCI White SP Power3 [Colony]	2001	512 [x 16]	10,000 sq. feet	1024 B [6 B]	500 [500]	~3 µsec	25 m; credit-based; CRC
Intel	Thunter Itanium2 Tiger4 [QsNet^{II}]	2004	1,024 [x 4]	120 m^2	2048 B [14 B]	928 [928]	0.240 µsec	13 m;credit-based; CRC for link, dest
Cray	XT3 [SeaStar]	2004	30,508 [x 1]	263.8 m^2	80 B [16 B]	3,200 [3,200]	few µsec	7 m; credit-based; CRC
Cray	X1E	2004	1,024 [x 1]	27 m^2	32 B [16 B]	1,600 [1,600]	~0 (direct LD/ST acc.)	5 m; credit-based; CRC
IBM	ASC Purple pSeries 575 [Federation]	2005	>1,280 [x 8]	6,720 sq. feet	2048 B [7 B]	2,000 [2,000]	~ 1 µsec *	25 m; credit-based; CRC
IBM	Blue Gene/L eServer Sol. [Torus Net]	2005	65,536 [x 2]	2,500 sq. feet (.9x.9x1.9 m^3/1K node rack)	256 B [8 B]	612,5 [1,050]	~ 3 µsec (2,300 cycles)	8.6 m; credit-based; CRC (header/pkt)

*with up to 4 packets processed in parallel

Outline

- E.1 Introduction *(Lecture 1)*
- E.2 Interconnecting Two Devices *(Lecture 1)*
- **E.3 Interconnecting Many Devices** *(Lecture 2)*
  - **Additional Network Structure and Functions**
  - **Shared-Media Networks**
  - **Switched-Media Networks**
  - **Comparison of Shared- versus Switched-Media Networks**
  - **Characterizing Performance: Latency & Effective Bandwidth**
- E.4 Network Topology *(Lecture 2)*
- E.5 Network Routing, Arbitration, and Switching *(Lecture 3)*
- E.6 Switch Microarchitecture *(Lecture 4)*
- E.7 Practical Issues for Commercial Interconnection Networks *(Lec 4)*
- E.8 Examples of Interconnection Networks *(Lecture 5)*
- E.9 Internetworking (skipped)
- E.10 Crosscutting Issues for Interconnection Networks (skipped)
- E.11 Fallacies and Pitfalls *(Lecture 5)*
- E.12 Concluding Remarks and References *(Lecture 5)*

Interconnecting Many Devices

Additional Network Structure and Functions

- Basic network structure and functions
  - Composing and processing messages, packets
  - Media and form factor
  - Packet transport
  - Reliable delivery (e.g., flow control) and error handling
- Additional structure
  - *Topology*
    - *What paths are possible for packets?*
    - Networks usually share paths among different pairs of devices
- Additional functions (*routing*, *arbitration*, *switching*)
  - Required in every network connecting more than two devices
  - Required to establish a valid path from source to destination
  - Complexity and applied order depends on the category of the topology: *shared-media* or *switched media*

Interconnecting Many Devices

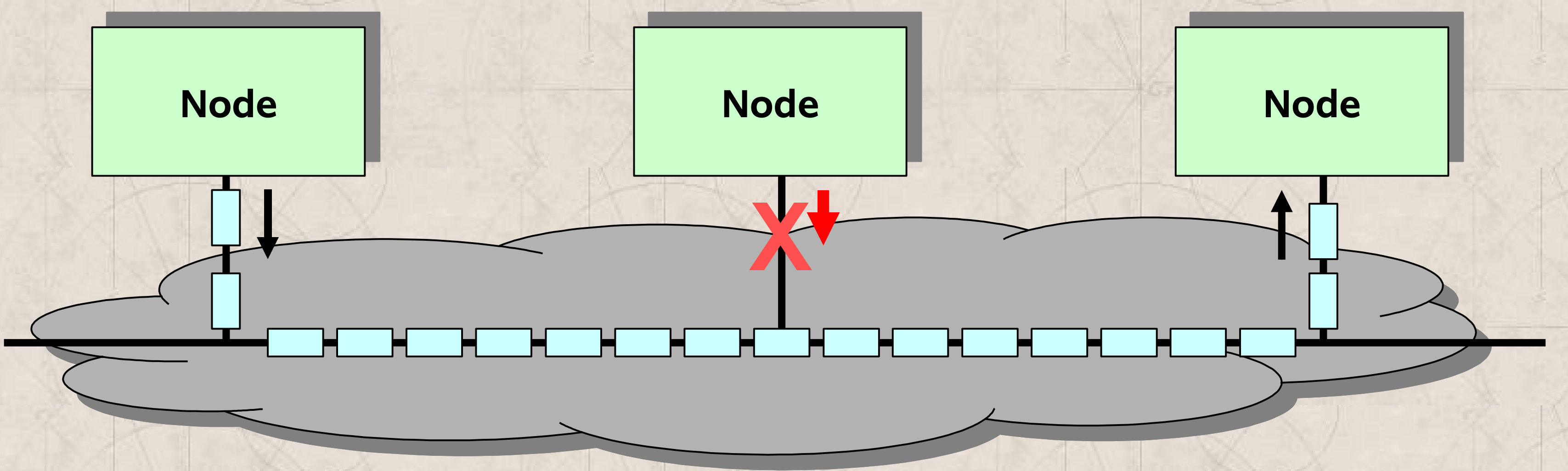
Additional Network Structure and Functions

- Additional functions (routing, arbitration, switching)
  - *Routing*
    - *Which of the possible paths are allowable (valid) for packets?*
    - Provides the set of operations needed to compute a valid path
    - Executed at source, intermediate, or even at destination nodes
  - *Arbitration*
    - *When are paths available for packets?* (along with flow control)
    - Resolves packets requesting the same resources at the same time
    - For every arbitration, there is a winner and possibly many losers
      - Losers are buffered (lossless) or dropped on overflow (lossy)
  - *Switching*
    - *How are paths allocated to packets?*
    - The winning packet (from arbitration) proceeds towards destination
    - Paths can be established one fragment at a time or in their entirety

Interconnecting Many Devices

Shared-media Networks

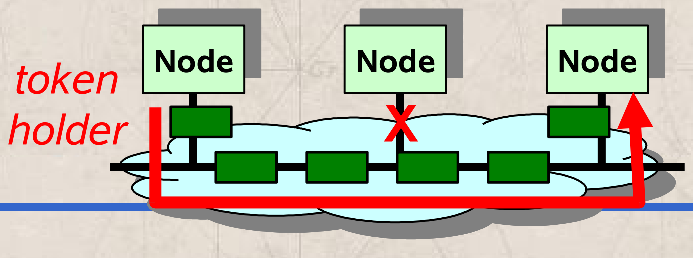
- The network media is shared by all the devices
- Operation: half-duplex or full-duplex



Interconnecting Many Devices

Shared-media Networks

- *Arbitration*
  - *Centralized* arbiter for smaller distances between devices
    - Dedicated control lines
  - *Distributed* forms of arbiters
    - CSMA/CD
      - The device first checks the network (carrier sensing)
      - Then checks if the data sent was garbled (collision detection)
      - If collision, device must send data again (retransmission): wait an increasing exponential random amount of time beforehand
      - Fairness is not guaranteed
    - Token ring—provides fairness
      - Owning the token provides permission to use network media



Interconnecting Many Devices

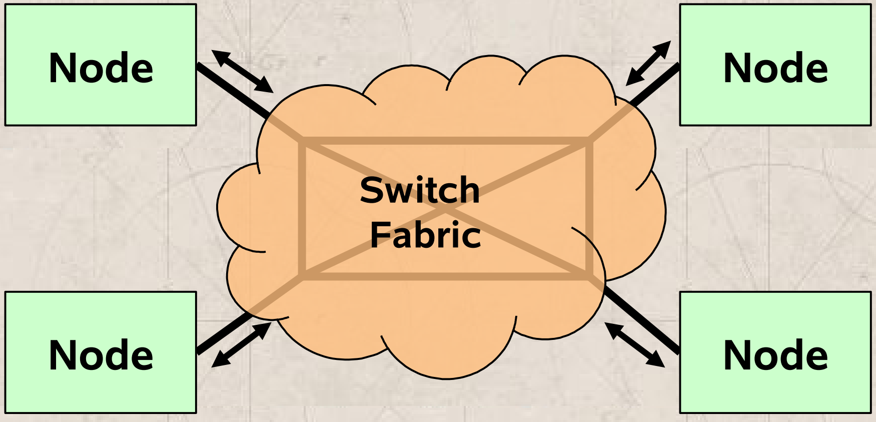
Shared-media Networks

- *Switching*
  - – Switching is straightforward
  - – The granted device connects to the shared media
- *Routing*
  - – Routing is straightforward
  - – Performed at all the potential destinations
    - › Each end node device checks whether it is the target of the packet
  - – Broadcast and multicast is easy to implement
    - › Every end node devices sees the data sent on shared link anyway
- Established order: arbitration, switching, and *then* routing

Interconnecting Many Devices

Switched-media Networks

- *Disjoint portions of the media are shared via switching*
- Switch fabric components
  - Passive *point-to-point links*
  - Active *switches*
    - Dynamically establish communication between sets of source-destination pairs
- Aggregate bandwidth can be many times higher than that of shared-media networks



Interconnecting Many Devices

Switched-media Networks

- *Routing*
  - Every time a packet enters the network, it is routed
- *Arbitration*
  - Centralized or distributed
  - Resolves conflicts among concurrent requests
- *Switching*
  - Once conflicts are resolved, the network "*switches in*" the required connections
- Established order: routing, arbitration, and ***then*** switching

Interconnecting Many Devices

Comparison of Shared- versus Switched-media Networks

- Shared-media networks
  - Low cost
  - Aggregate network bandwidth does not scale with # of devices
  - Global arbitration scheme required (a possible bottleneck)
  - Time of flight increases with the number of end nodes
- Switched-media networks
  - Aggregate network bandwidth scales with number of devices
  - Concurrent communication
    - Potentially much higher network effective bandwidth
  - *Beware:* inefficient designs are quite possible
    - Superlinear network cost but sublinear network effective bandwidth

Interconnecting Many Devices

Characterizing Performance: Latency & Effective Bandwidth

$$\text{Latency} = \text{Sending overhead} + (T_{TotalProp} + T_R + T_A + T_S) + \frac{\text{Packet size}}{\text{Bandwidth}} + \text{Receiving overhead}$$

T_R = routing delay
T_A = arbitration delay
T_S = switching delay
$T_{TotalProp}$ = propagation delay

lower bound (contention delay not included)

upper bound (contention effects not *fully* included)

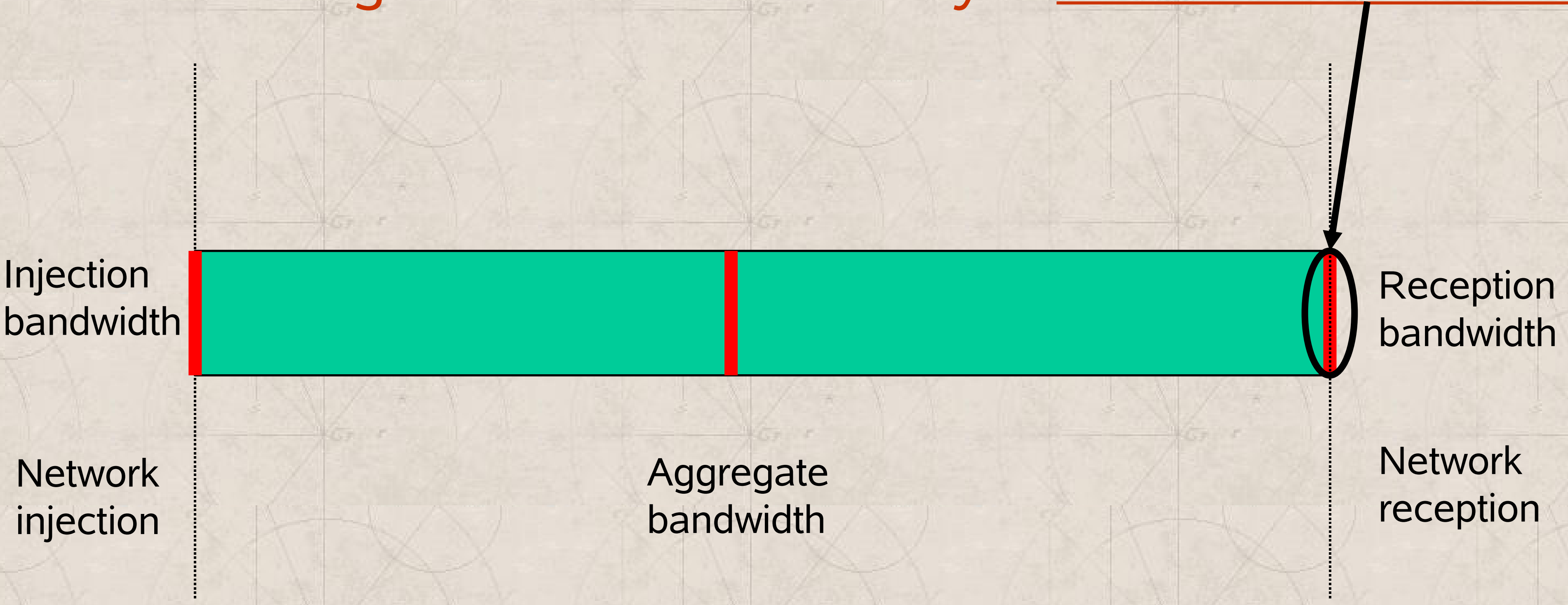
$$\text{Effective bandwidth} = \min(BW_{NetworkInjection}, BW_{Network}, \sigma \times BW_{NetworkReception})$$

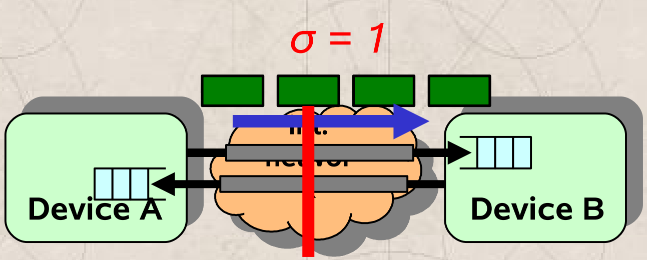
$$= \min(N \times BW_{LinkInjection}, BW_{Network}, \sigma \times N \times BW_{LinkReception})$$

σ = average reception factor (contention at reception links due to application behavior)

Interconnecting Many Devices

Characterizing Performance: Latency & Effective Bandwidth

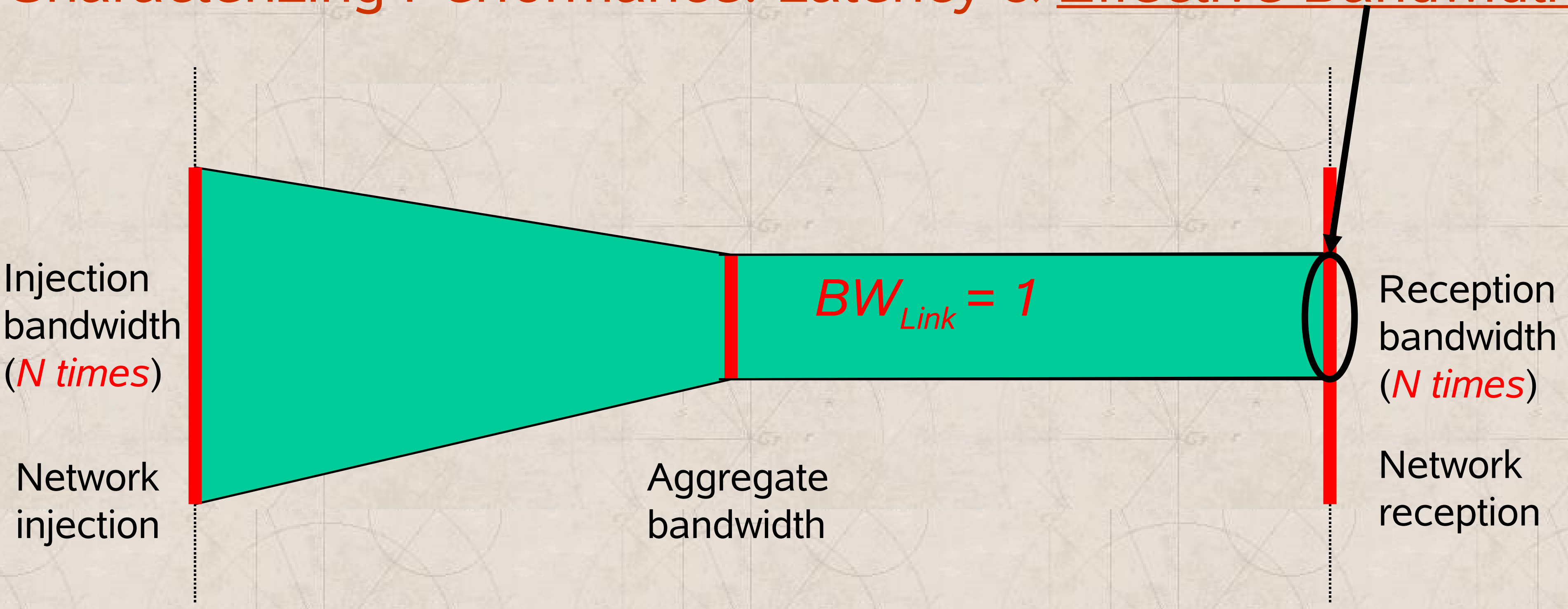




Dedicated-link network

Interconnecting Many Devices

Characterizing Performance: Latency & Effective Bandwidth



Injection bandwidth (*N times*)

Network injection

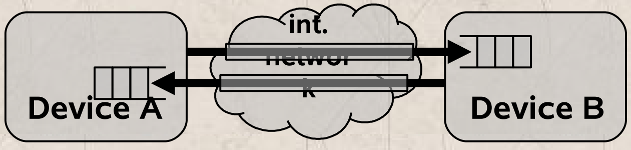
$BW_{Link} = 1$

Aggregate bandwidth

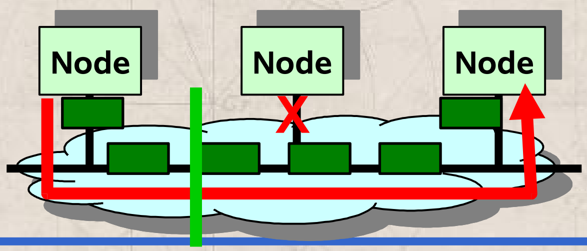
Reception bandwidth (*N times*)

Network reception

$BW_{Network} = (BW_{NetworkInjection})/N$



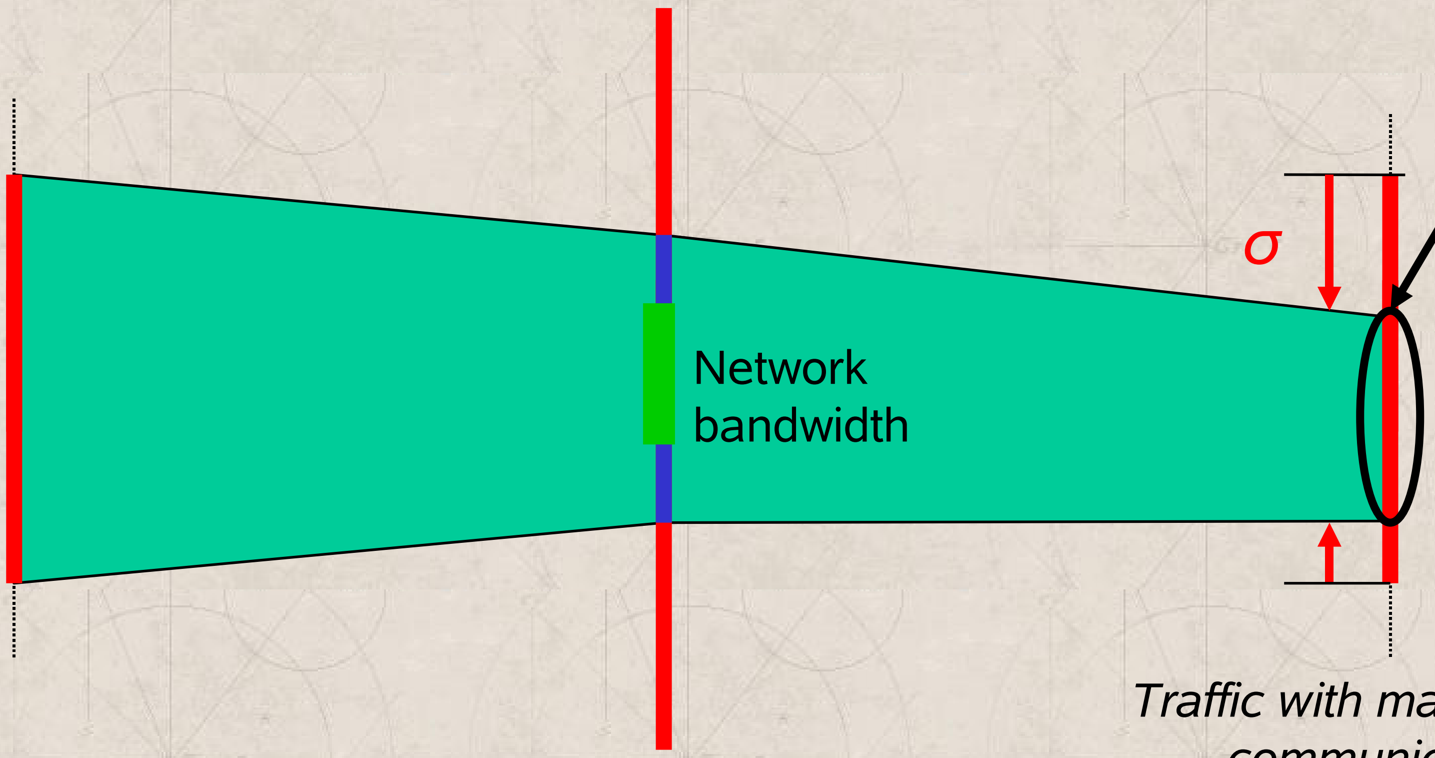
Dedicated-link network



Shared-media network

Interconnecting Many Devices

Characterizing Performance: Latency & Effective Bandwidth



Injection bandwidth (*N times*)

Network injection

Network bandwidth

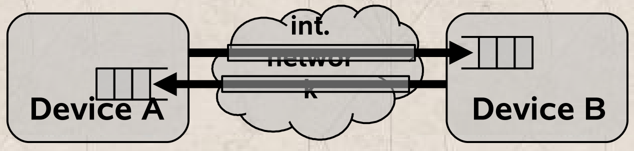
Reception bandwidth (*N times*)

Network reception

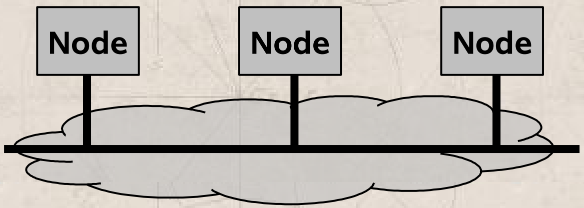
Aggregate bandwidth (≥ *N times*)

Traffic with many-to-one communication

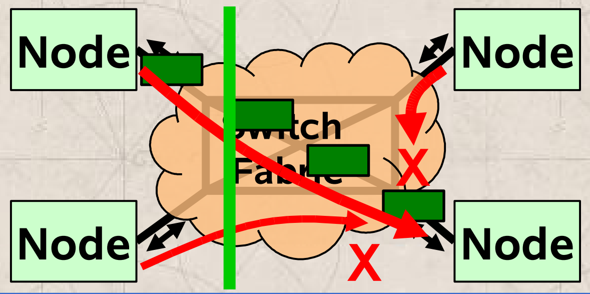
σ < 1



Dedicated-link network



Shared-media network

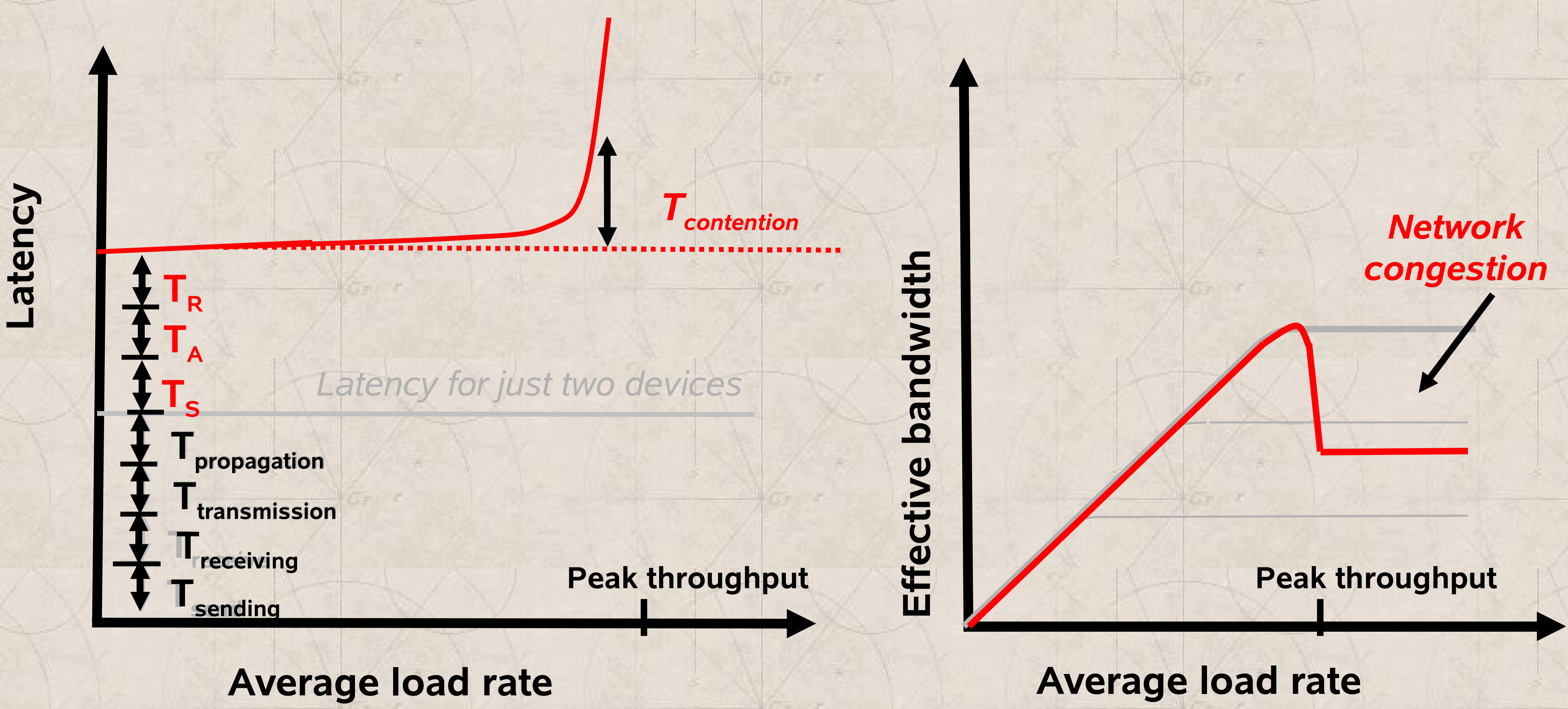


Switched-media network

Interconnecting Many Devices

Characterizing Performance: Latency & Effective Bandwidth

- *Characteristic performance plots*: latency vs. average load rate; throughput (effective bandwidth) vs. average load rate



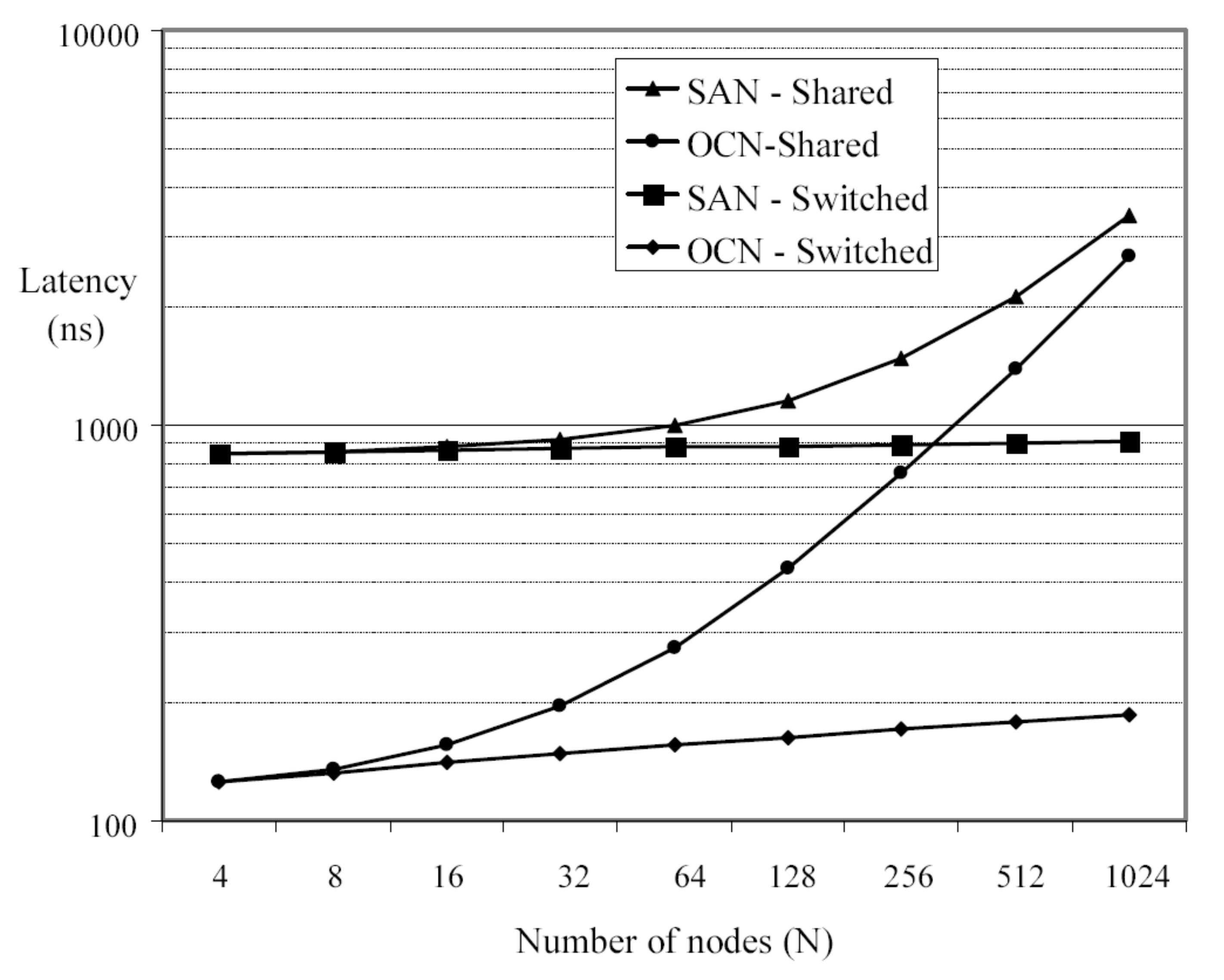
Interconnecting Many Devices

Characterizing Performance: Latency & Effective Bandwidth

- Example: plot the total packet latency and effective bandwidth
  - Assume *N* scales from 4 to 1024 end nodes and the following:
  - Shared media (one link) and switched media (*N* links)
  - All network links have data bandwidth of 8 Gbps
  - Unicast transmission of packets of size 100 bytes
  - Overheads
    - sending: *x* + 0.05 ns/byte; receiving: 4/3(*x*) + 0.05 ns/byte
    - *x* is 0 for OCNs, 0.3 μs for SANs, 3 μs for LANs, 30 μs for WANs
  - Distances: 0.5 cm, 5 m, 5,000 m, and 5,000 km
  - Routing, Arbitration, and Switching
    - shared media: T_R = 2.5 ns, T_A = 2.5(N) ns, T_S = 2.5 ns
    - switched media: $T_R = T_A = T_S = 2.5 (\log_2 N)$ ns
  - $\sigma = N^{-1}$ (shared), $\sigma = (\log_2 N)^{-1/4}$ (switched)

Interconnecting Many Devices

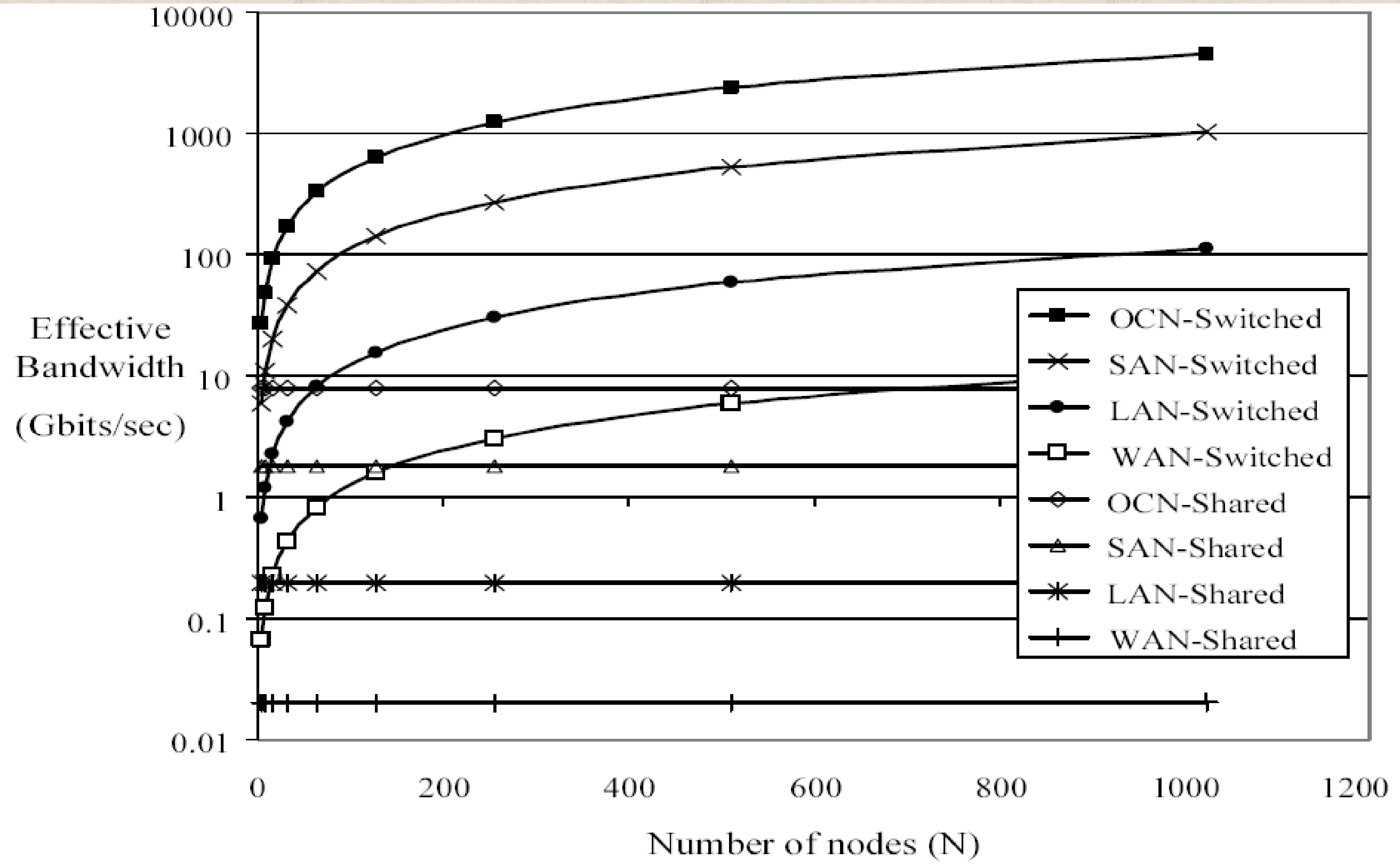
Characterizing Performance: Latency & Effective Bandwidth



- For OCNs, T_R, T_A, and T_S combine to dominate time of flight delay and are >> than other latency components for all network sizes.
- For SANs, T_R, T_A, and T_S combine to dominate time of flight delay but are less than other latency components for switched-media (but not negligibly so)
- For LANs and WANs, latency is dominated by propagation delay, T_{Prop}

Interconnecting Many Devices

Characterizing Performance: Latency & Effective Bandwidth



Eff. BW** constant through scaling for <u>shared;</u> **Eff. BW** increases for <u>switched,</u> but scaled down by **σ

***overhead** limits Eff. BW in switched for all but the OCN*

Outline

- E.1 Introduction *(Lecture 1)*
- E.2 Interconnecting Two Devices *(Lecture 1)*
- E.3 Interconnecting Many Devices *(Lecture 2)*
- **E.4 Network Topology** *(Lecture 2)*
  - **Preliminaries and Evolution**
  - **Centralized Switched (Indirect) Networks**
  - **Distributed Switched (Direct) Networks**
  - **Comparison of Indirect and Direct Networks**
  - **Characterizing Performance: Latency & Effective Bandwidth**
- E.5 Network Routing, Arbitration, and Switching *(Lecture 3)*
- E.6 Switch Microarchitecture *(Lecture 4)*
- E.7 Practical Issues for Commercial Interconnection Networks *(Lecture 4)*
- E.8 Examples of Interconnection Networks *(Lecture 5)*
- E.9 Internetworking (skipped)
- E.10 Crosscutting Issues for Interconnection Networks (skipped)
- E.11 Fallacies and Pitfalls *(Lecture 5)*
- E.12 Concluding Remarks and References *(Lecture 5)*

Network Topology

Preliminaries and Evolution

- One switch suffices to connect a small number of devices
  - Number of switch ports limited by VLSI technology, power consumption, packaging, and other such *cost* constraints
- A *fabric* of interconnected switches (i.e., *switch fabric* or *network fabric*) is needed when the number of devices is much larger
  - The *topology* must make a path(s) available for every pair of devices—property of *connectedness* or *full access* (*What paths?)*
- Topology defines the connection structure across all components
  - *Bisection bandwidth*: the *minimum* bandwidth of all links crossing a network split into two roughly equal halves
  - *Full bisection bandwidth*:
    - Network $BW_{Bisection}$ = Injection (or Reception) $BW_{Bisection}$ = $N/2$
  - Bisection bandwidth mainly affects *performance*
- Topology is constrained primarily by local chip/board *pin-outs*; secondarily, (if at all) by global bisection bandwidth

Network Topology

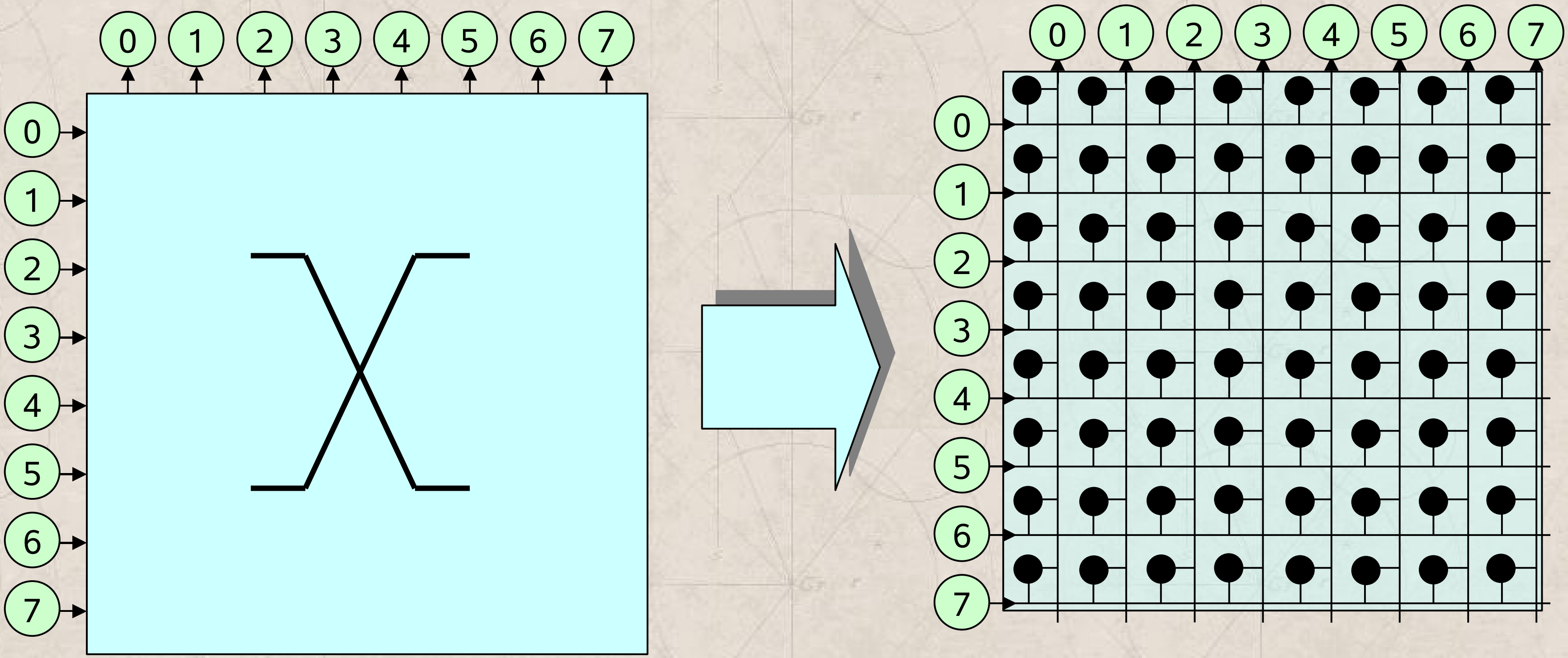
Preliminaries and Evolution

- Several tens of topologies proposed, but less than a dozen used
- 1970s and 1980s
  - Topologies were proposed to reduce *hop count*
- 1990s
  - Pipelined transmission and switching techniques
  - Packet latency became decoupled from hop count
- 2000s
  - Topology still important (especially OCNs, SANs) when *N* is high
  - Topology impacts performance and has a major impact on cost

Network Topology

Centralized Switched (Indirect) Networks

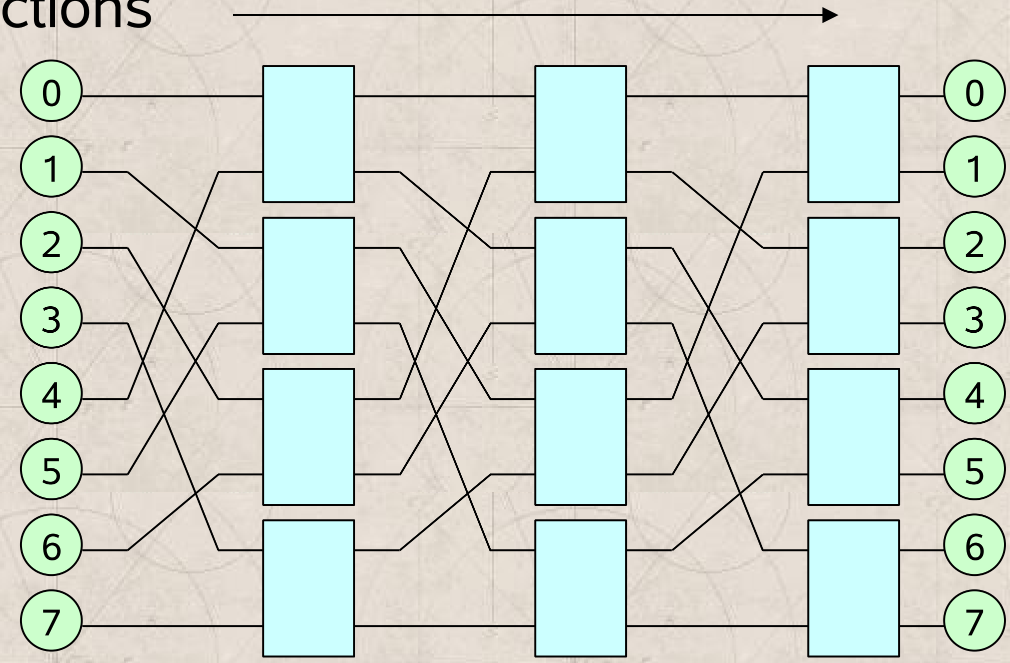
- ***Crossbar network***
  - Crosspoint switch complexity increases quadratically with the number of crossbar input/output ports, *N*, i.e., grows as O(N^2)
  - Has the property of being *non-blocking*



Network Topology

Centralized Switched (Indirect) Networks

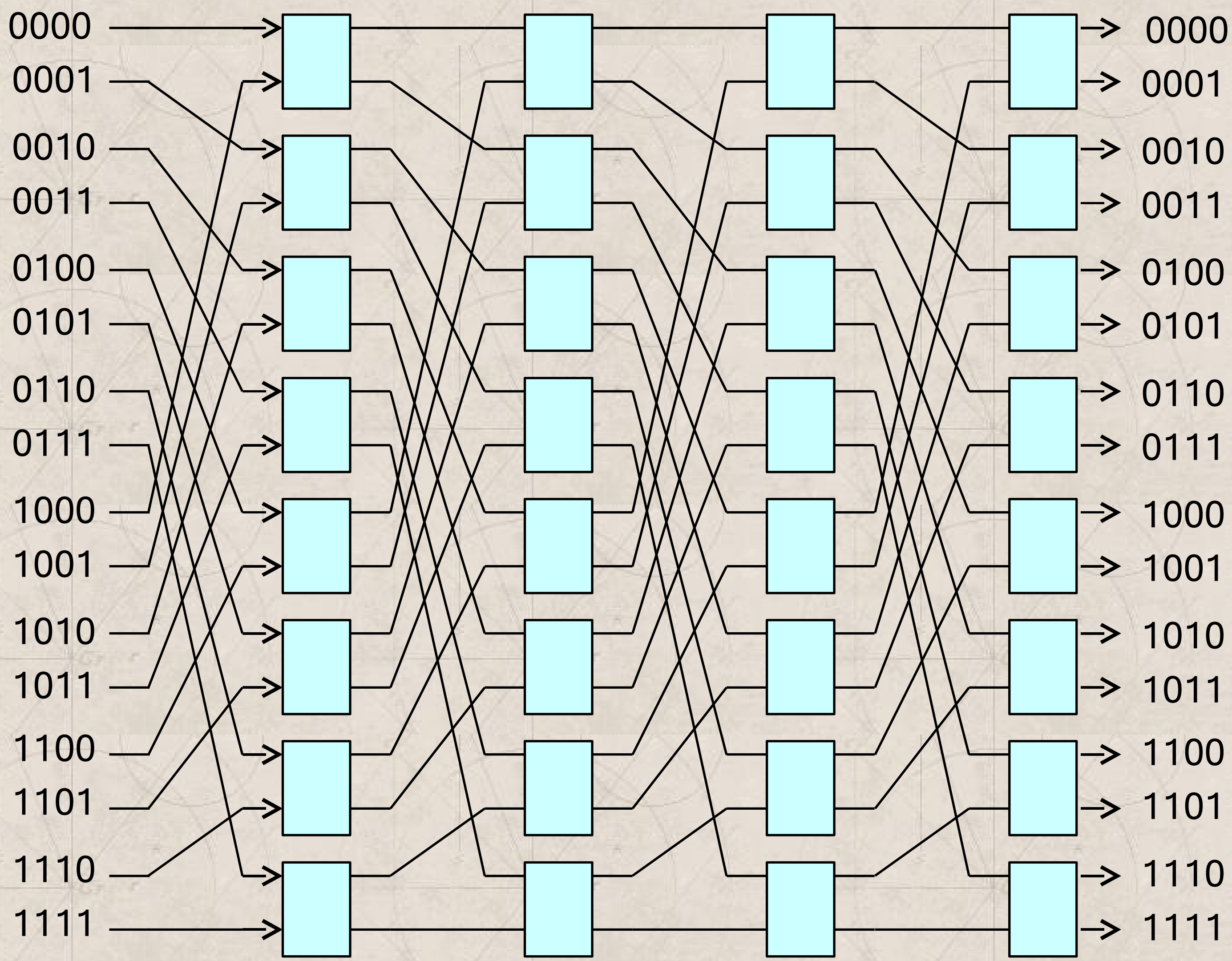
- ***Multistage interconnection networks (MINs)***
  - – Crossbar split into several stages consisting of smaller crossbars
  - – Complexity grows as O(*N* × log *N*), where *N* is # of end nodes
  - – Inter-stage connections represented by a set of permutation functions



Omega topology, perfect-shuffle exchange

Network Topology

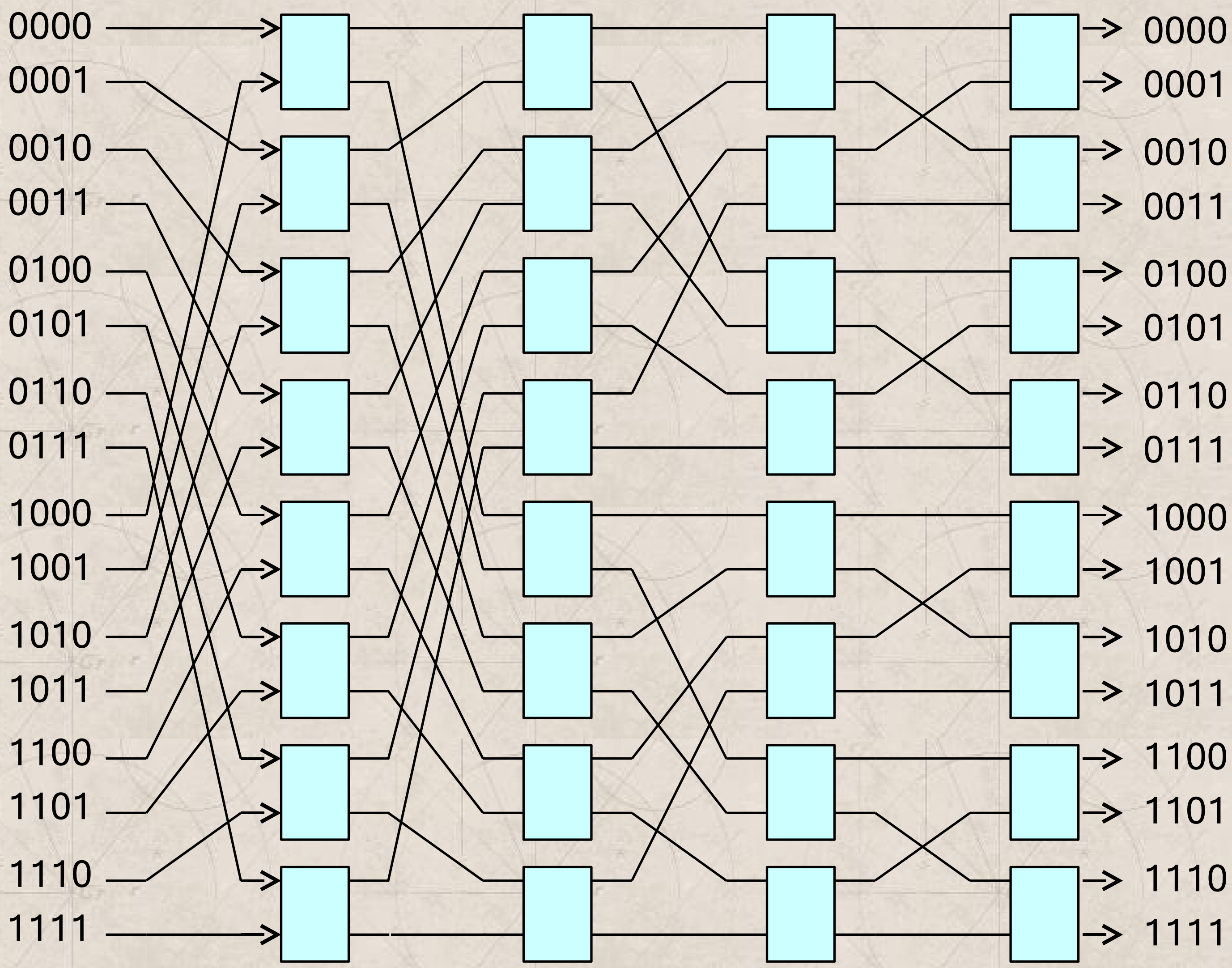
Centralized Switched (Indirect) Networks



16 port, *4 stage* ***Omega*** *network*

Network Topology

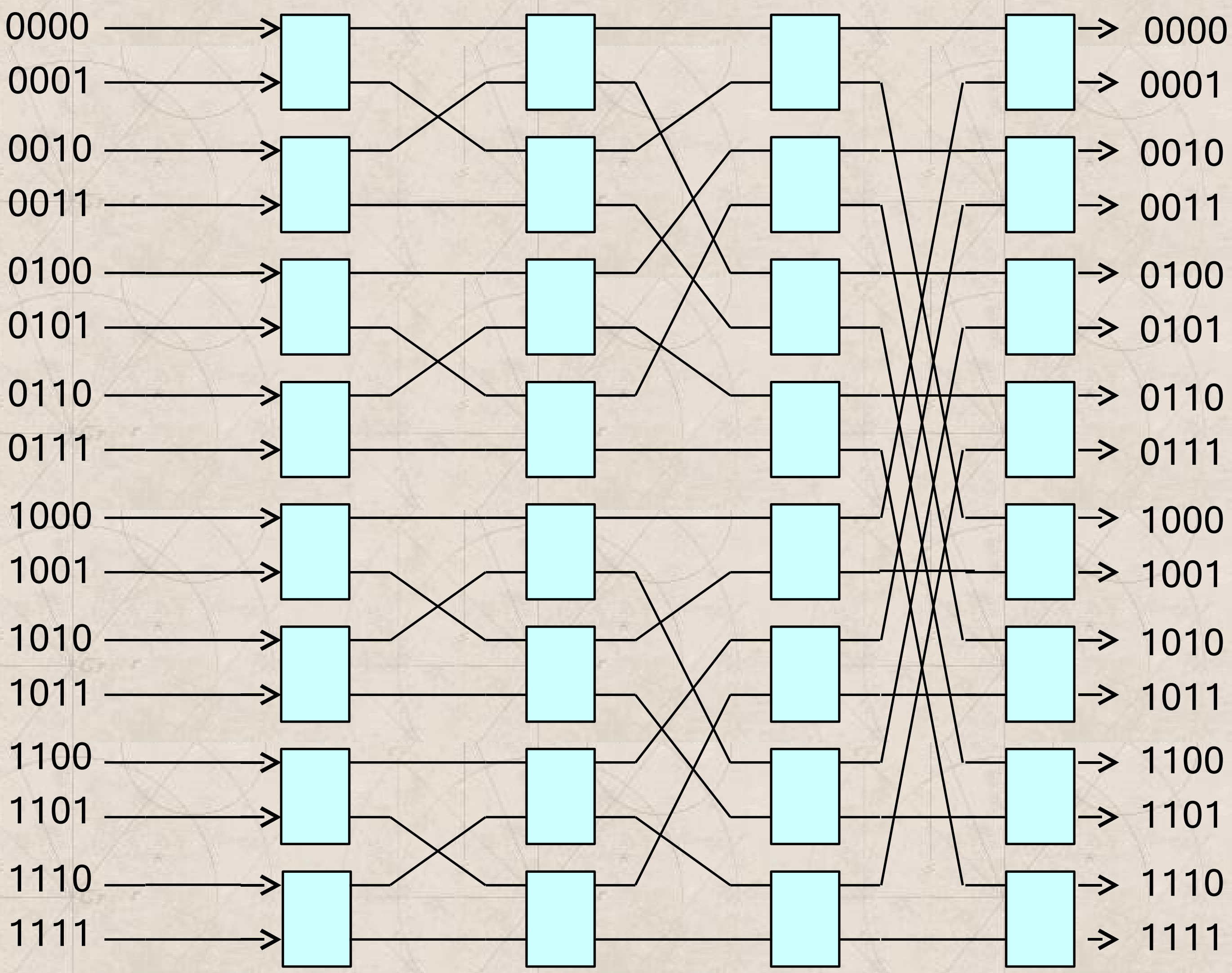
Centralized Switched (Indirect) Networks



16 port, *4 stage **Baseline** network*

Network Topology

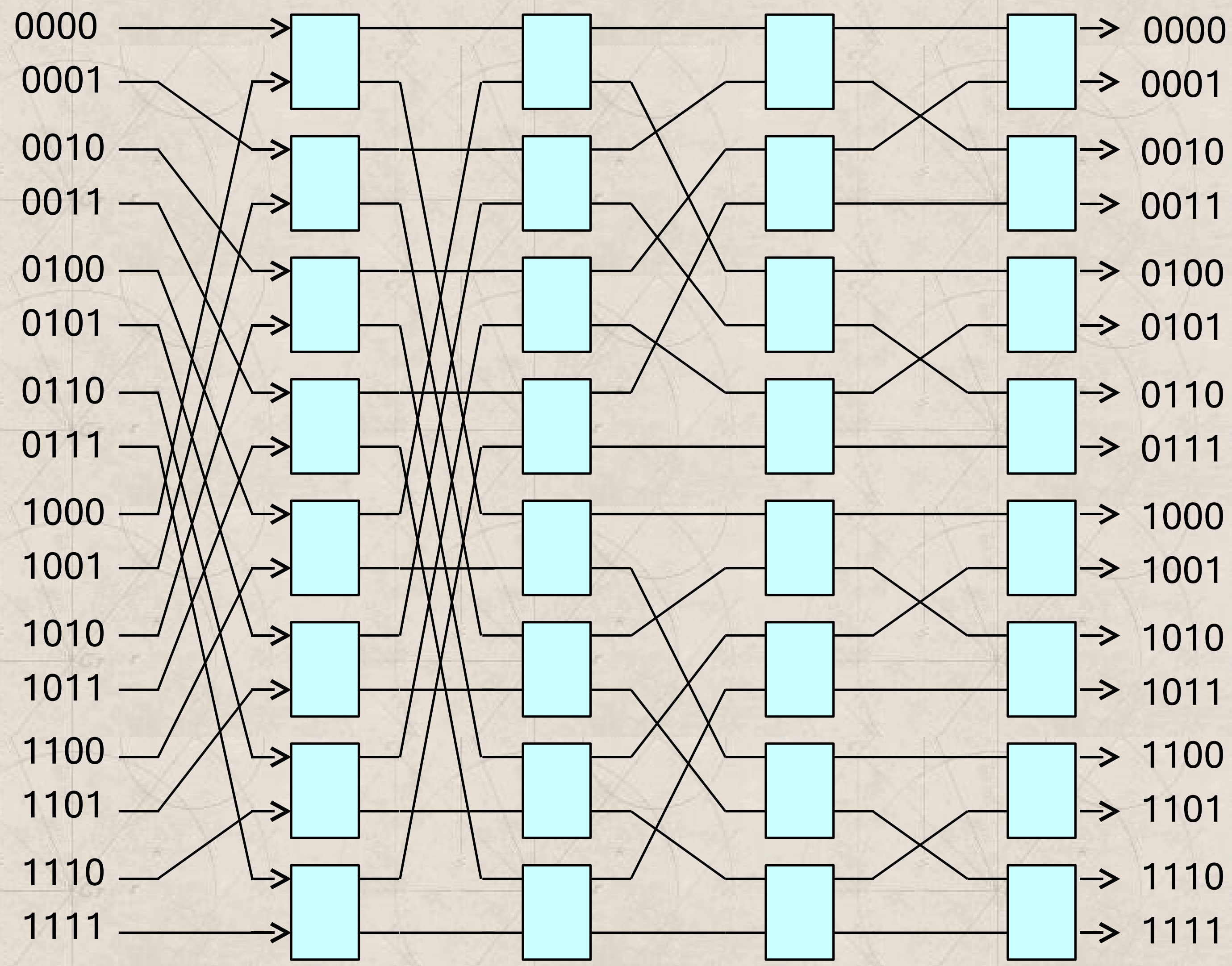
Centralized Switched (Indirect) Networks



16 port, *4 stage **Butterfly** network*

Network Topology

Centralized Switched (Indirect) Networks



16 port, *4 stage* ***Cube*** *network*

Network Topology

Centralized Switched (Indirect) Networks

- Multistage interconnection networks (MINs)
  - MINs interconnect N input/output ports using k x k switches
    - › $\log_k N$ switch stages, each with N/k switches
    - › $N/k(\log_k N)$ total number of switches
  - *Example:* Compute the switch and link costs of interconnecting 4096 nodes using a crossbar relative to a MIN, assuming that switch cost grows quadratically with the number of input/output ports (k). Consider the following values of k:
    - › MIN with 2 x 2 switches
    - › MIN with 4 x 4 switches
    - › MIN with 16 x 16 switches

Network Topology

Centralized Switched (Indirect) Networks

- Example: compute the relative switch and link costs, *N* = 4096

$\text{cost(crossbar)}_{\textit{switches}} = 4096^2$

$\text{cost(crossbar)}_{\textit{links}} = 8192$

$\text{relative\_cost}(2 \times 2)_{\textit{switches}} = 4096^2 / (2^2 \times 4096/2 \times \log_2 4096) = \mathbf{\mathit{170}}$

$\text{relative\_cost}(2 \times 2)_{\textit{links}} = 8192 / (4096 \times (\log_2 4096 + 1)) = 2/13 = \mathbf{\mathit{0.1538}}$

$\text{relative\_cost}(4 \times 4)_{\textit{switches}} = 4096^2 / (4^2 \times 4096/4 \times \log_4 4096) = \mathbf{\mathit{170}}$

$\text{relative\_cost}(4 \times 4)_{\textit{links}} = 8192 / (4096 \times (\log_4 4096 + 1)) = 2/7 = \mathbf{\mathit{0.2857}}$

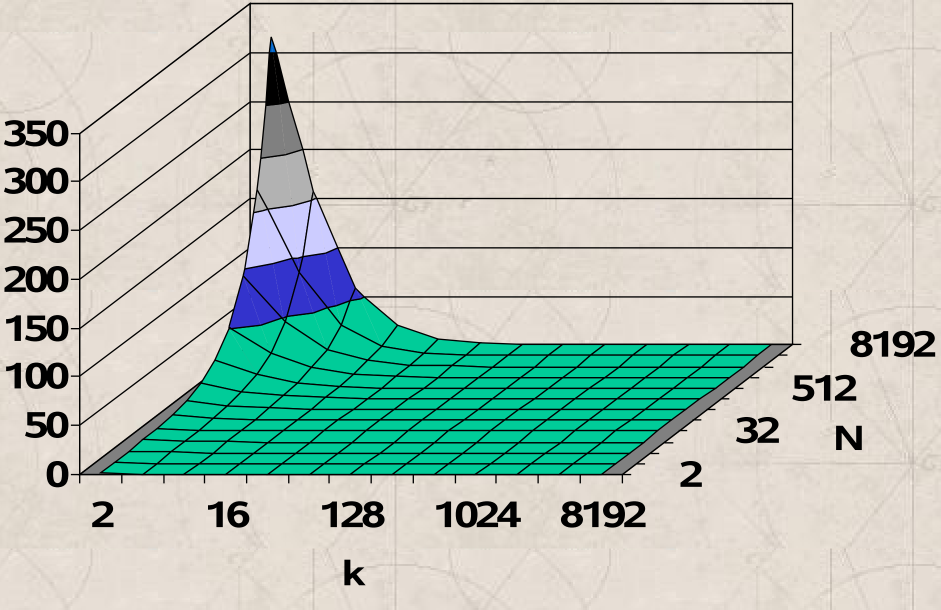
$\text{relative\_cost}(16 \times 16)_{\textit{switches}} = 4096^2 / (16^2 \times 4096/16 \times \log_{16} 4096) = \mathbf{\mathit{85}}$

$\text{relative\_cost}(16 \times 16)_{\textit{links}} = 8192 / (4096 \times (\log_{16} 4096 + 1)) = 2/4 = \mathbf{\mathit{0.5}}$

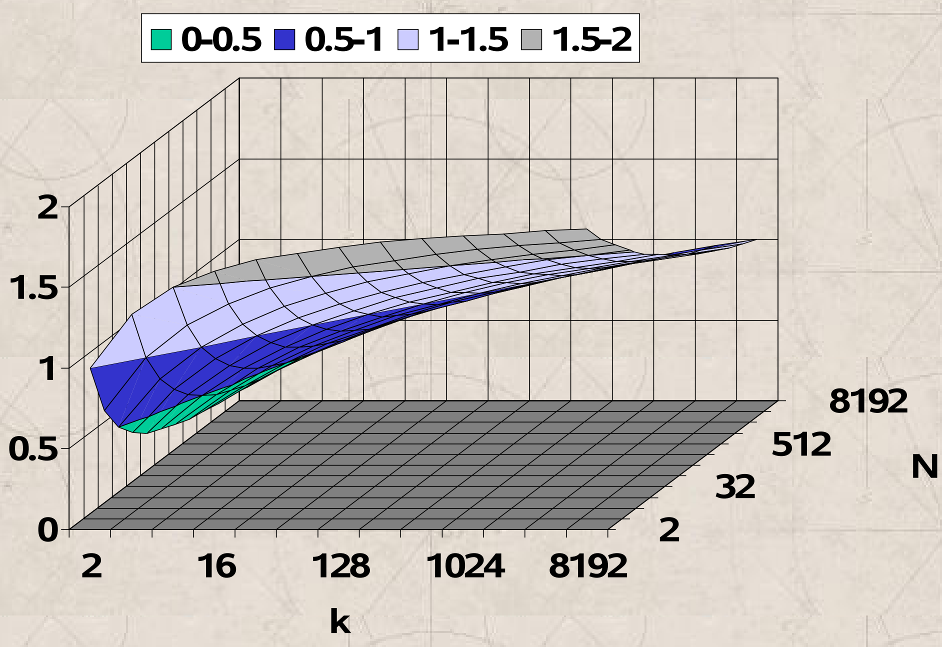
Network Topology

Centralized Switched (Indirect) Networks

- Relative switch and link costs for various values of *k* and *N* (crossbar relative to a MIN)



Relative switch cost

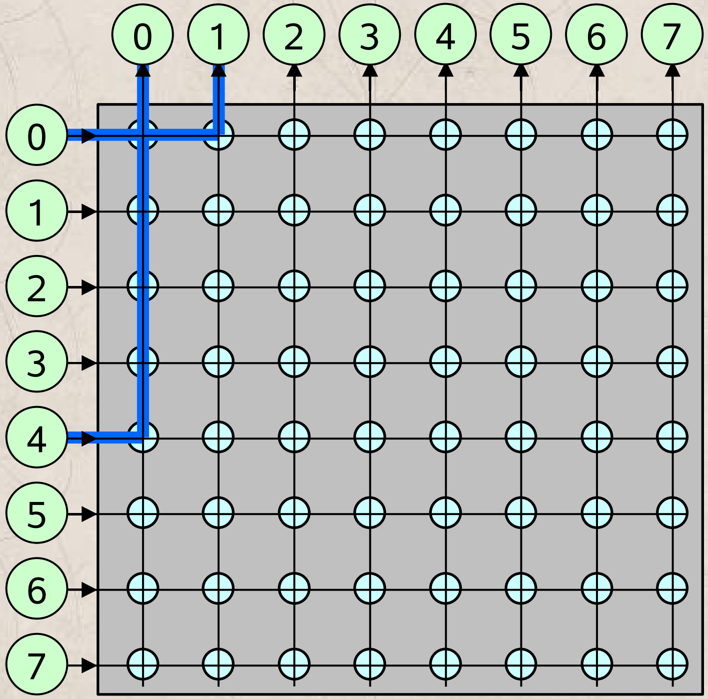


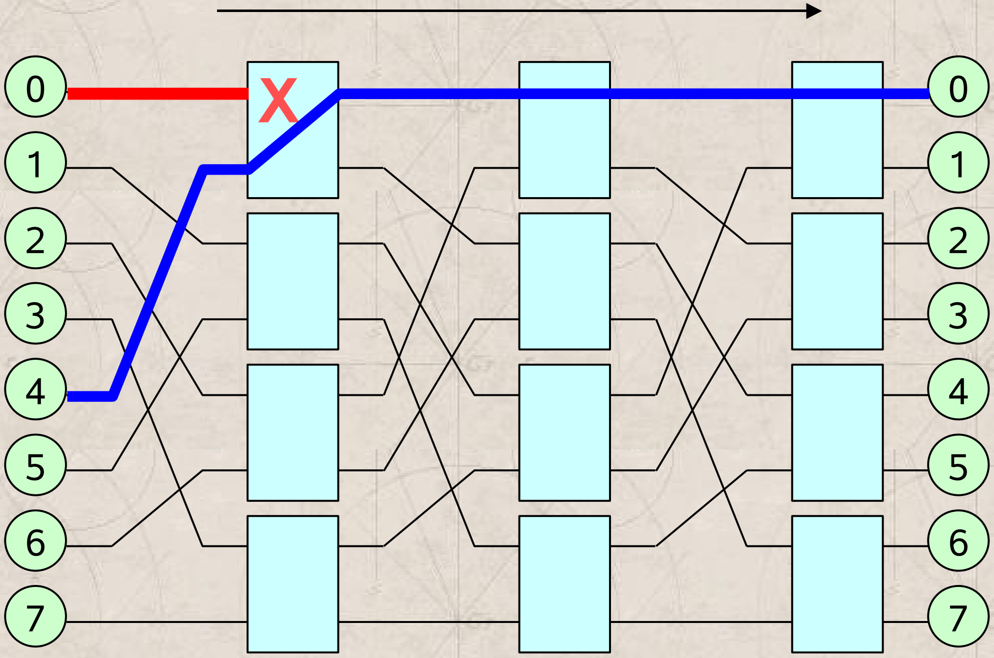
Relative link cost

Network Topology

Centralized Switched (Indirect) Networks

- Reduction in MIN switch *cost* comes at the price of *performance*
  - Network has the property of being *blocking*
  - *Contention* is more likely to occur on network links
    - › Paths from different sources to different destinations share one or more links





non-blocking topology

blocking topology

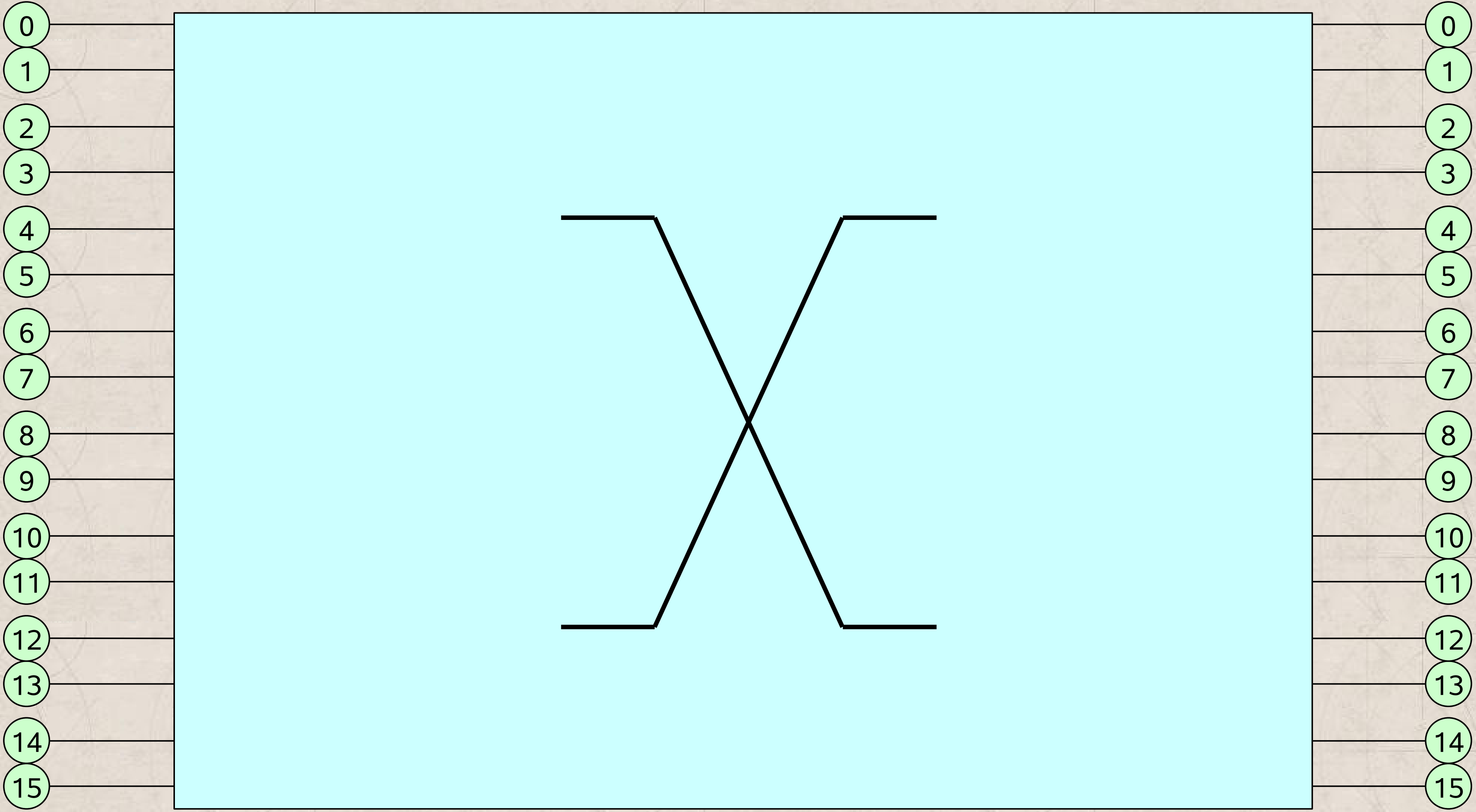
Network Topology

Centralized Switched (Indirect) Networks

- How to reduce blocking in MINs? *Provide alternative paths!*
  - – Use larger switches (can equate to using more switches)
    - › ***Clos network***: minimally three stages (non-blocking)
      - » A larger switch in the middle of two other switch stages provides enough alternative paths to avoid all conflicts
  - – Use more switches
    - › Add $\log_k N$ - 1 stages, mirroring the original topology
      - » *Rearrangeably non-blocking*
      - » Allows for non-conflicting paths
      - » Doubles network *hop count (distance), d*
      - » Centralized control can rearrange established paths
    - › ***Benes topology***: $2(\log_2 N)$ - 1 stages (rearrangeably non-blocking)
      - » Recursively applies the three-stage Clos network concept to the middle-stage set of switches to reduce all switches to 2 x 2

Network Topology

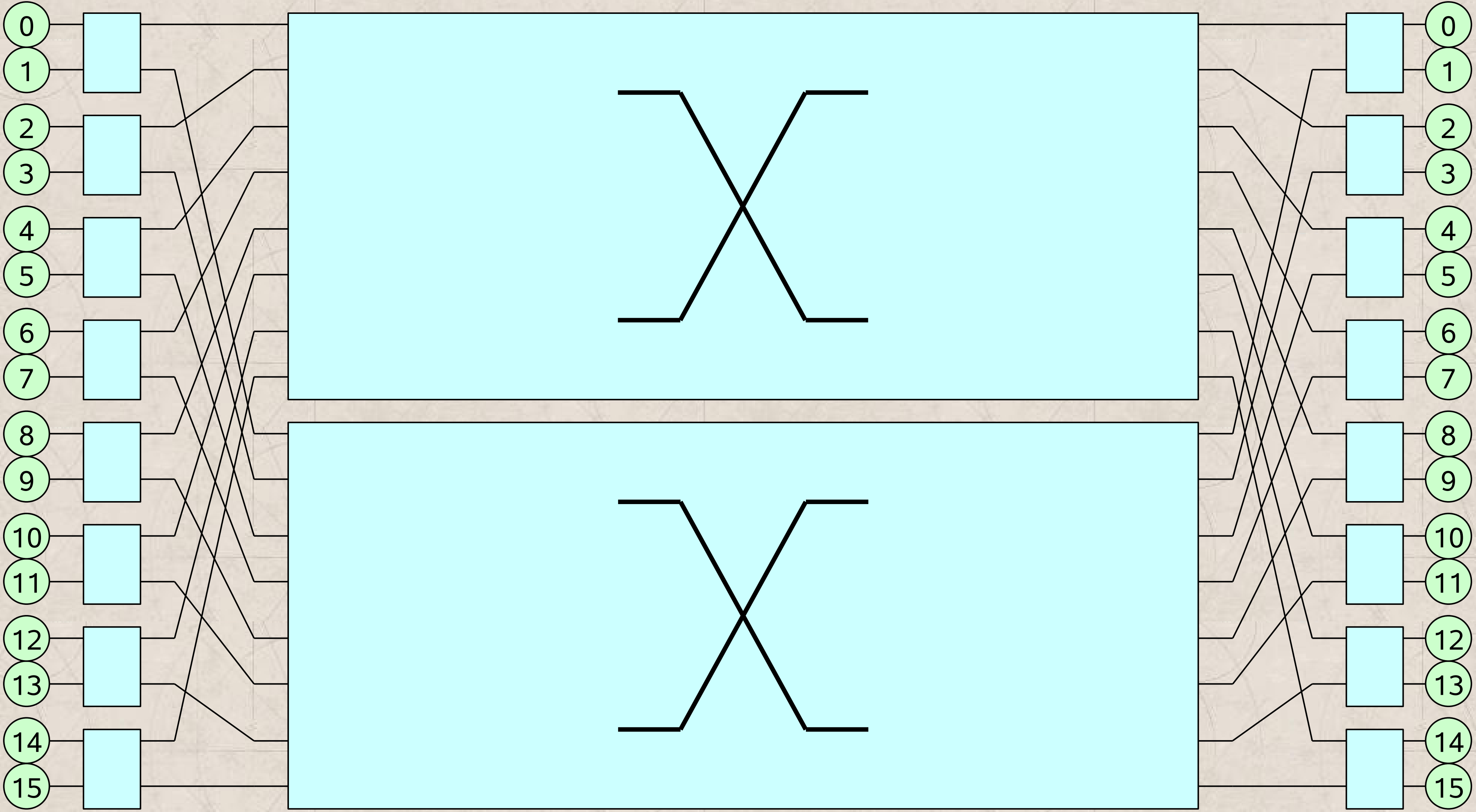
Centralized Switched (Indirect) Networks



16 port ***Crossbar*** network

Network Topology

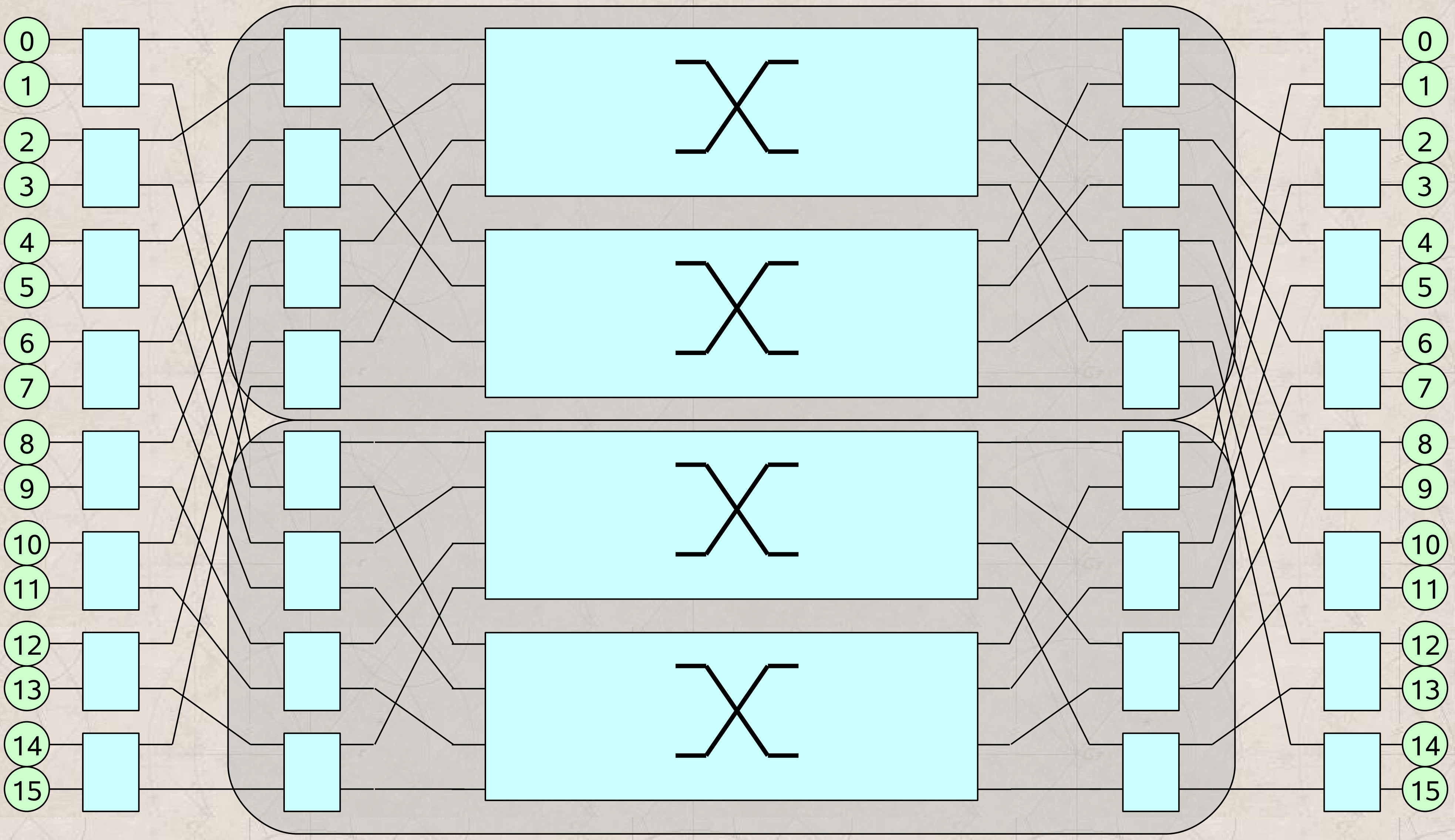
Centralized Switched (Indirect) Networks



16 port, ***3-stage Clos*** network

Network Topology

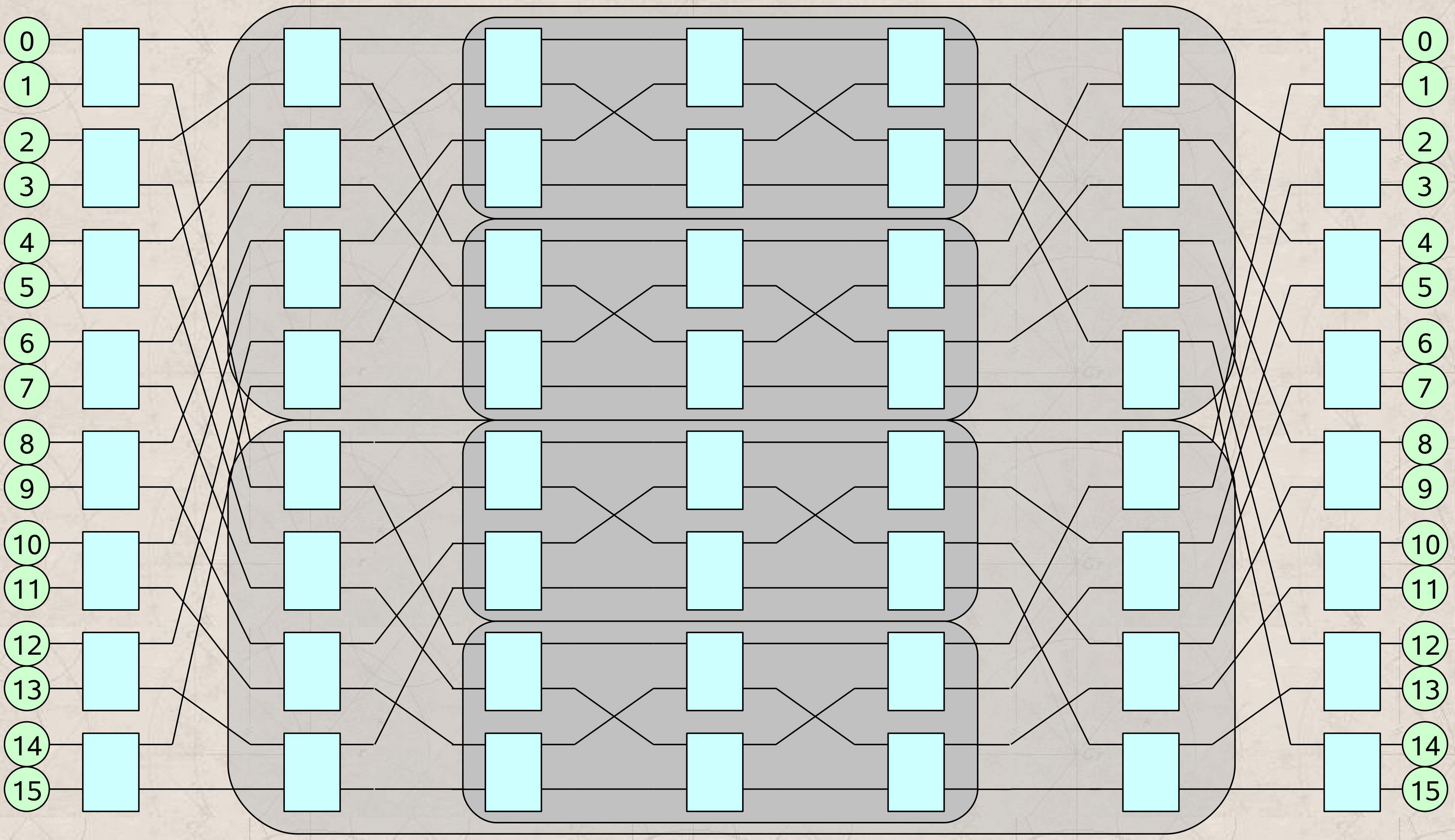
Centralized Switched (Indirect) Networks



16 port, ***5-stage Clos*** network

Network Topology

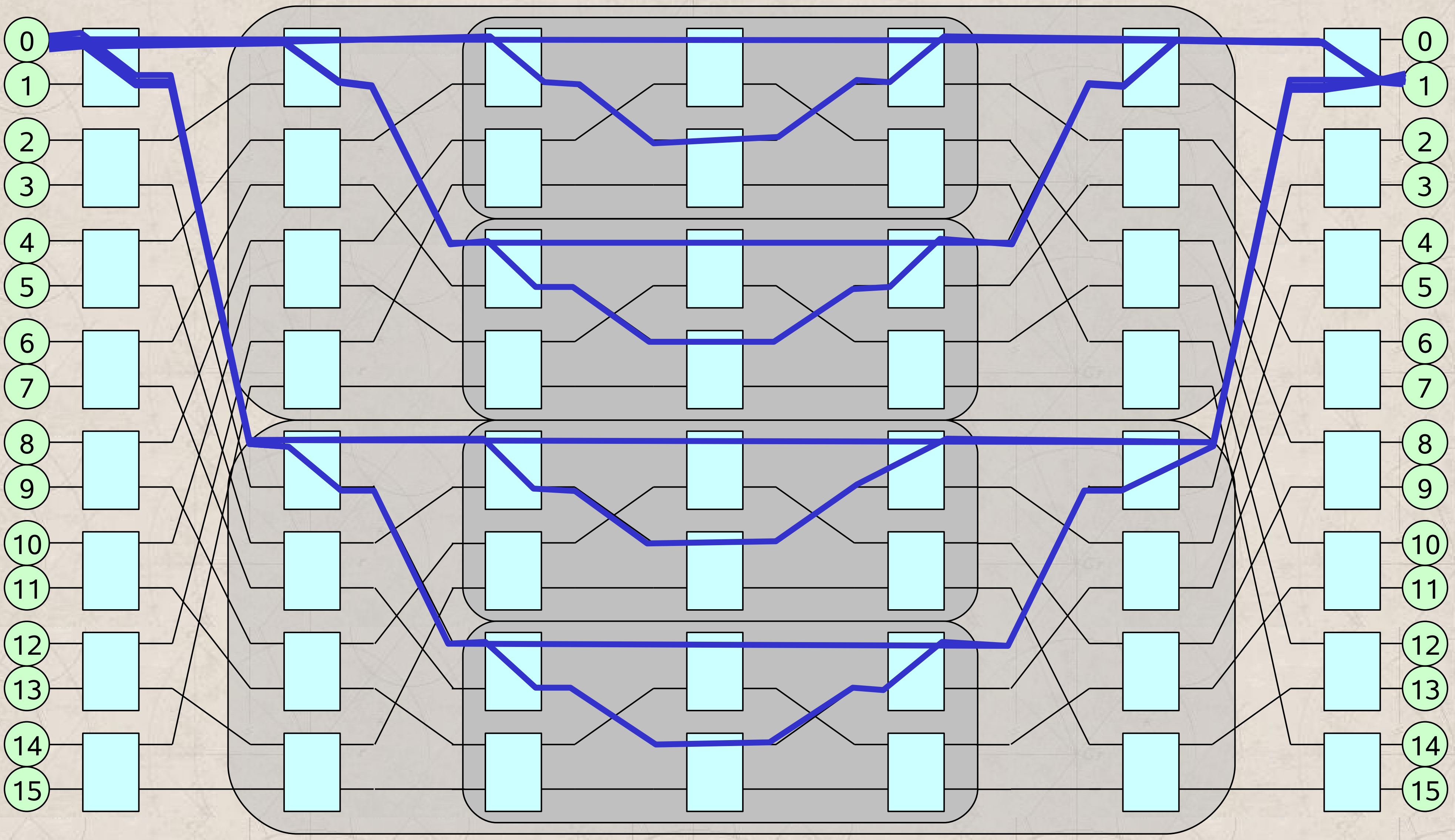
Centralized Switched (Indirect) Networks



16 port, ***7 stage Clos*** network = ***Benes topology***

Network Topology

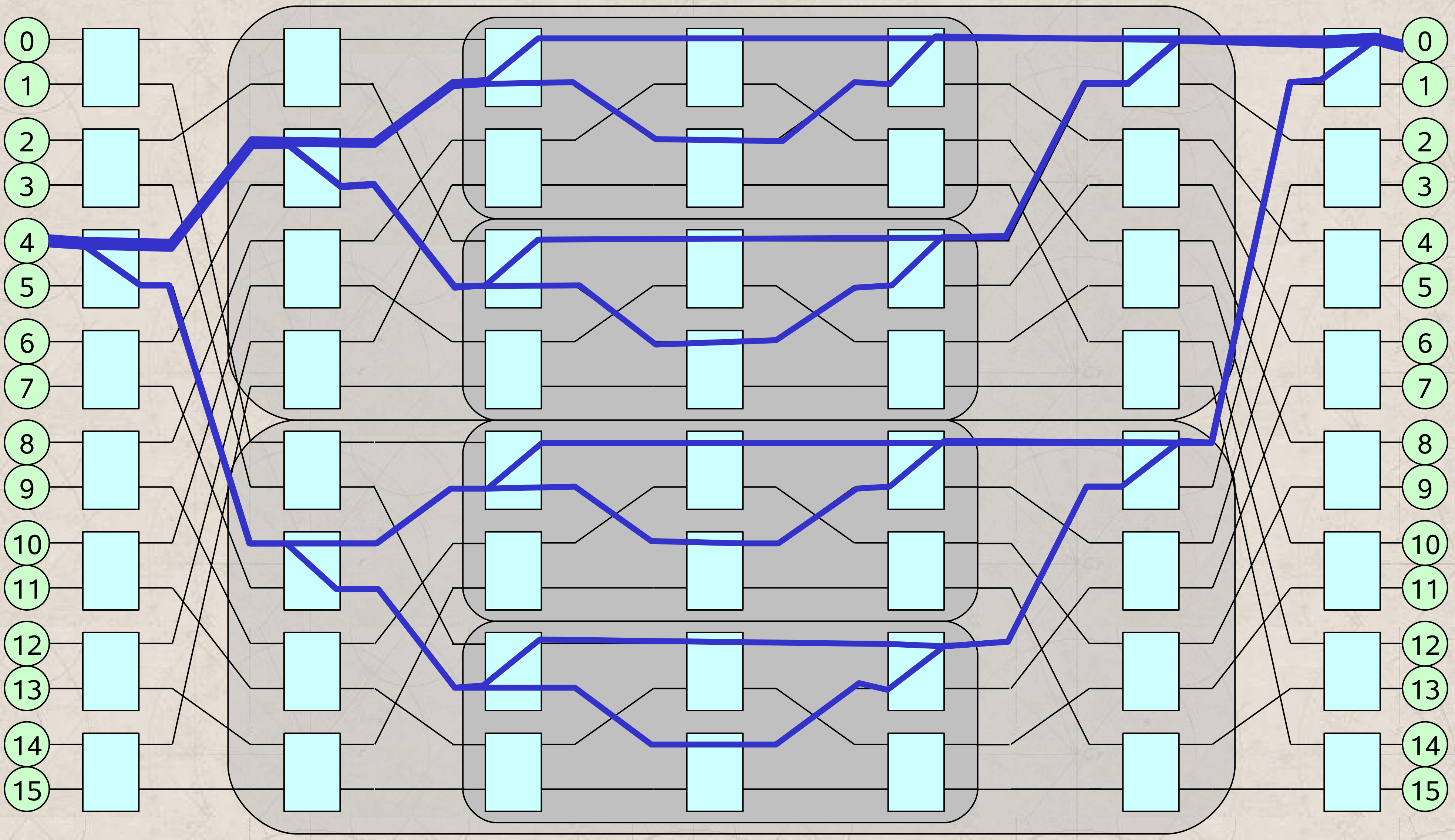
Centralized Switched (Indirect) Networks



Alternative paths from 0 to 1. 16 port, *7 stage Clos* network = *Benes topology*

Network Topology

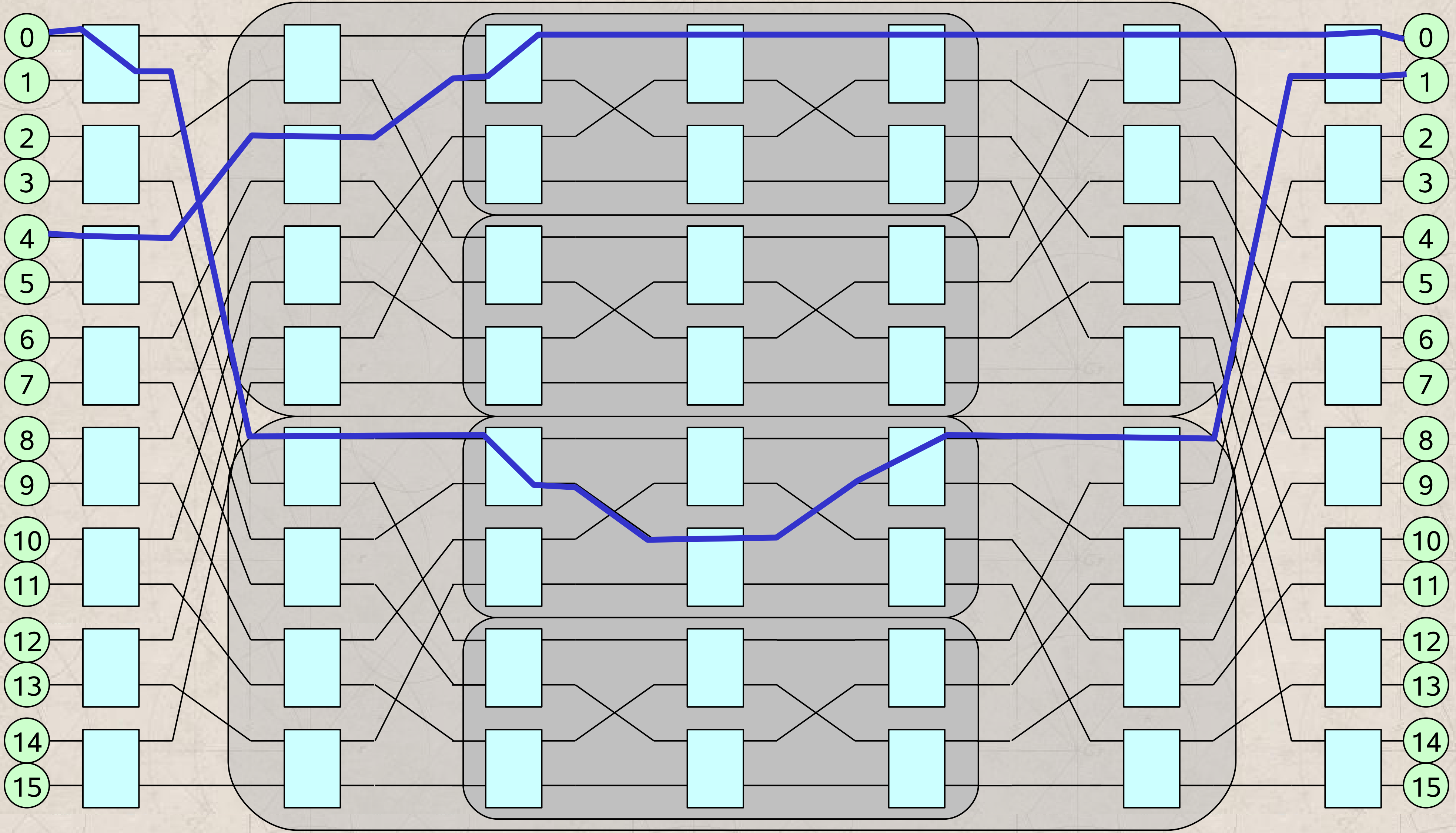
Centralized Switched (Indirect) Networks



Alternative paths from 4 to 0. 16 port, *7 stage Clos* network = *Benes topology*

Network Topology

Centralized Switched (Indirect) Networks



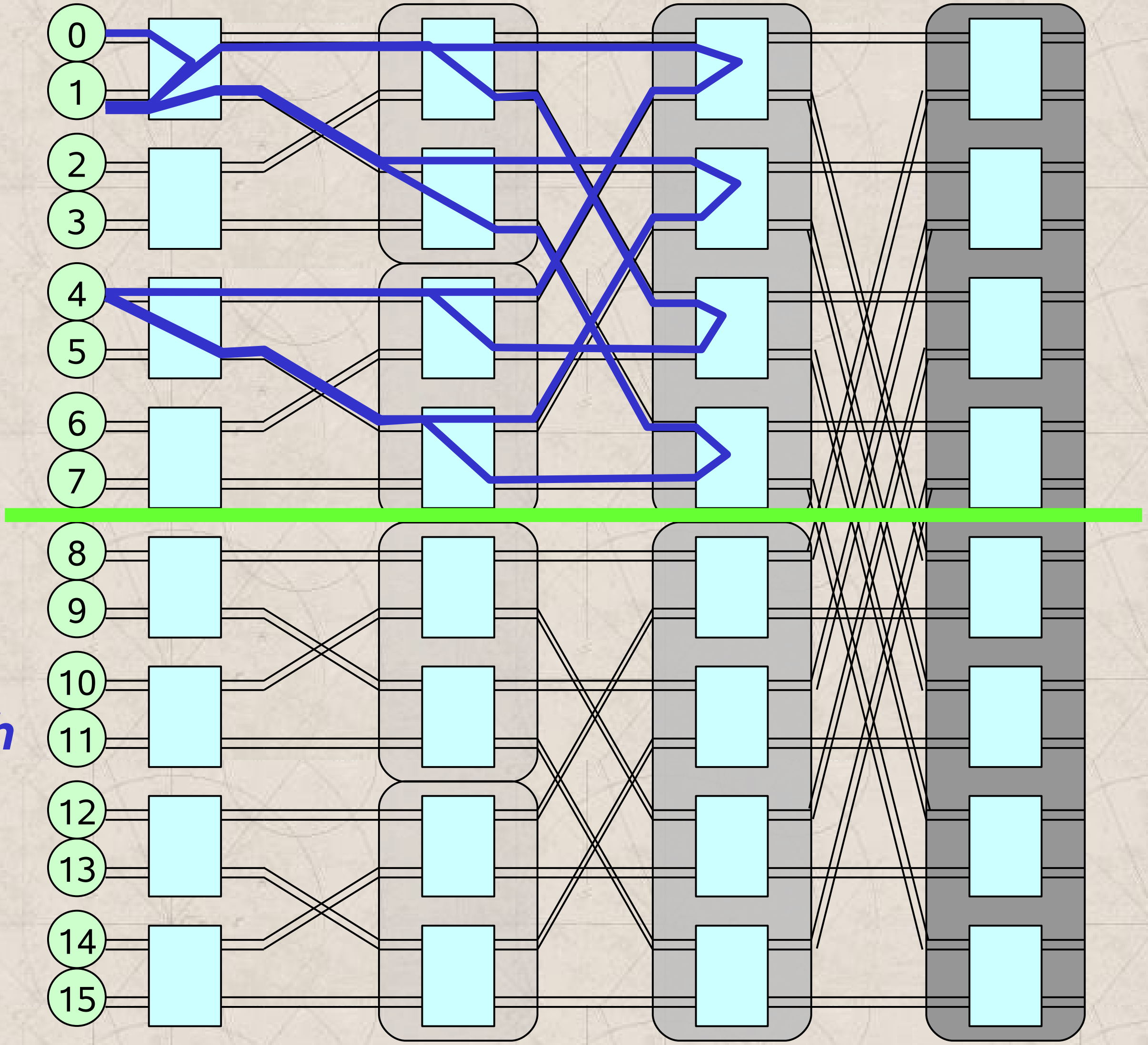
Contention free, paths 0 to 1 and 4 to 1. 16 port, *7 stage Clos* network = *Benes topology*

Network Topology

Centralized Switched (Indirect) Networks

- *Bidirectional MINs*
- Increase modularity
- Reduce hop count, *d*
- ***Fat tree network***
  - Nodes at tree leaves
  - Switches at tree vertices
  - Total link bandwidth is constant across all tree levels, with ***full bisection bandwidth***
  - Equivalent to folded Benes topology
  - Preferred topology in many SANs

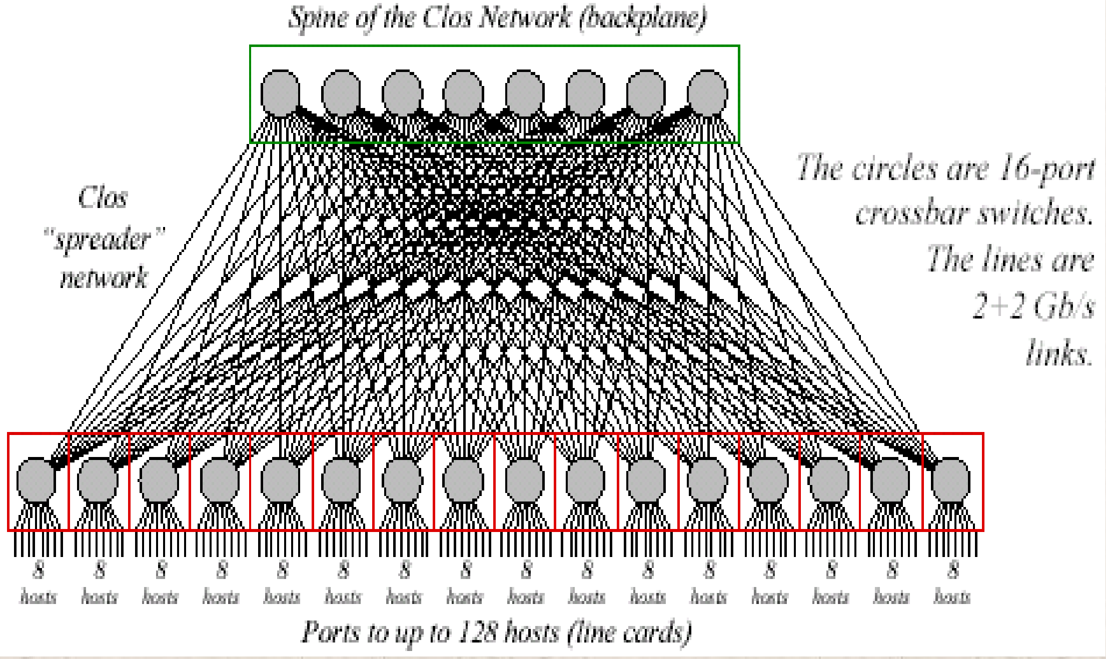
Network Bisection

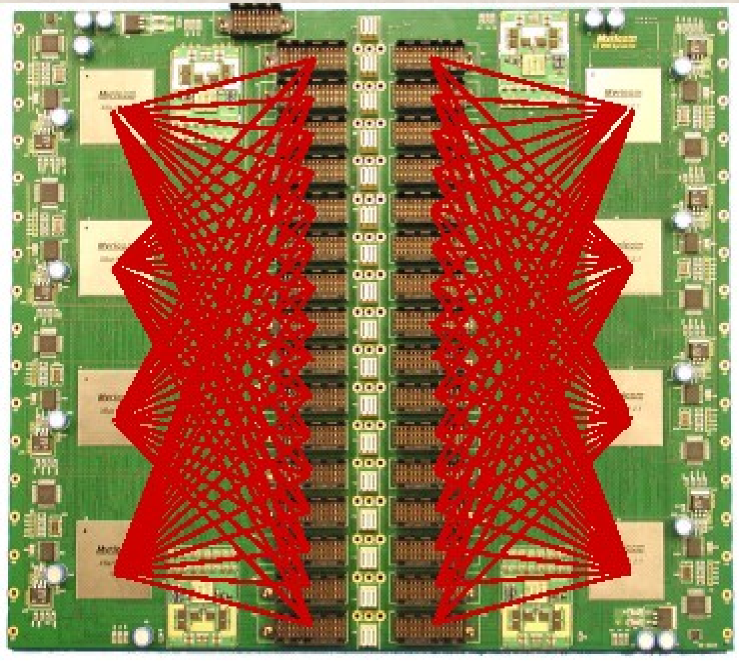


***Folded Clos* = *Folded Benes* = *Fat tree* network**

Network Topology

Myrinet-2000 Clos Network for 128 Hosts


Spine of the Clos Network (backplane)
Clos "spreader" network
The circles are 16-port crossbar switches.
The lines are 2+2 Gb/s links.
8 hosts
8 hosts
8 hosts
8 hosts
8 hosts
8 hosts
8 hosts
8 hosts
8 hosts
8 hosts
8 hosts
8 hosts
8 hosts
8 hosts
8 hosts
8 hosts
Ports to up to 128 hosts (line cards)

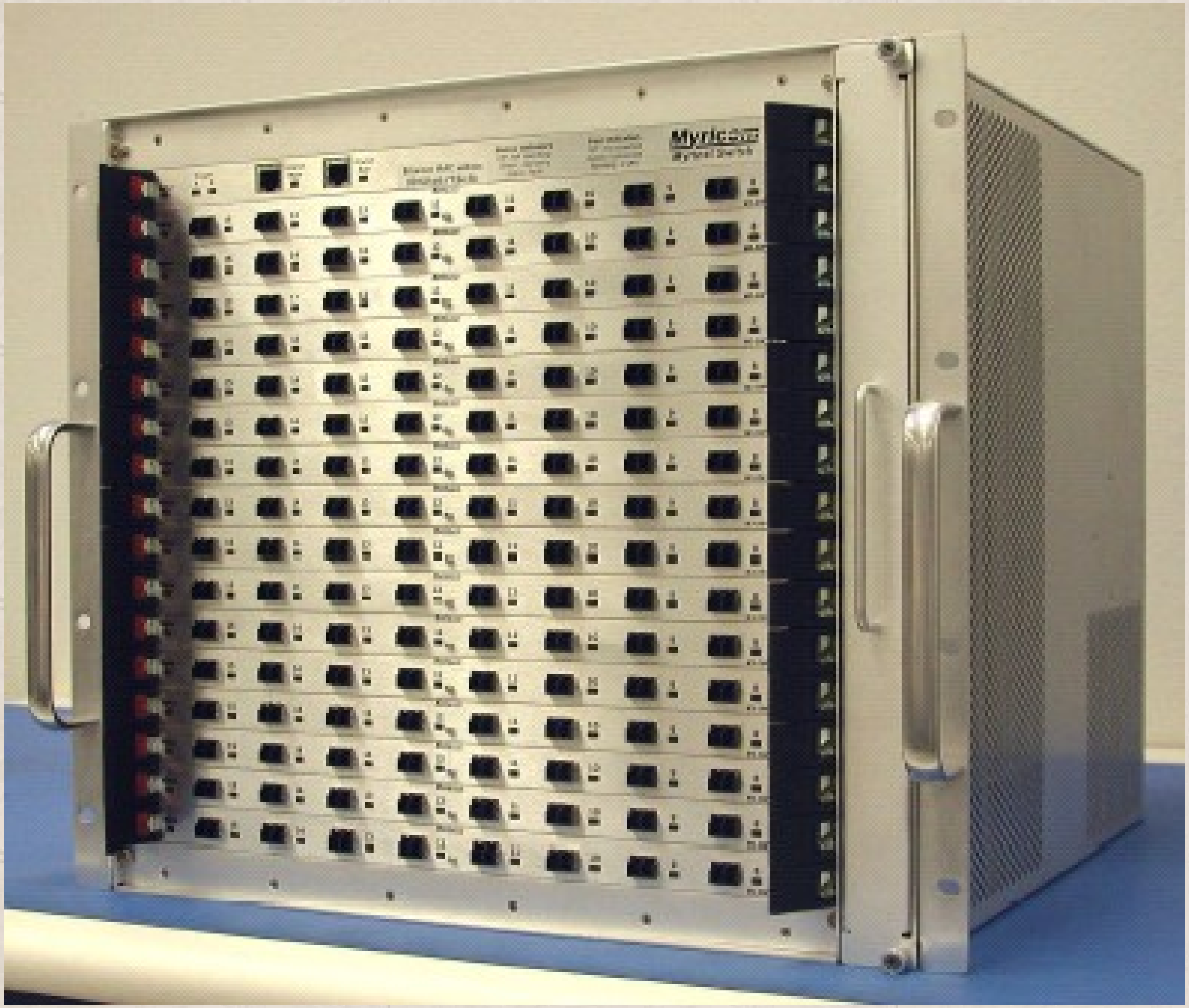


- Backplane of the M3-E128 Switch
- M3-SW16-8F fiber line card (8 ports)

http://myri.com

Network Topology

Myrinet-2000 Clos Network for 128 Hosts

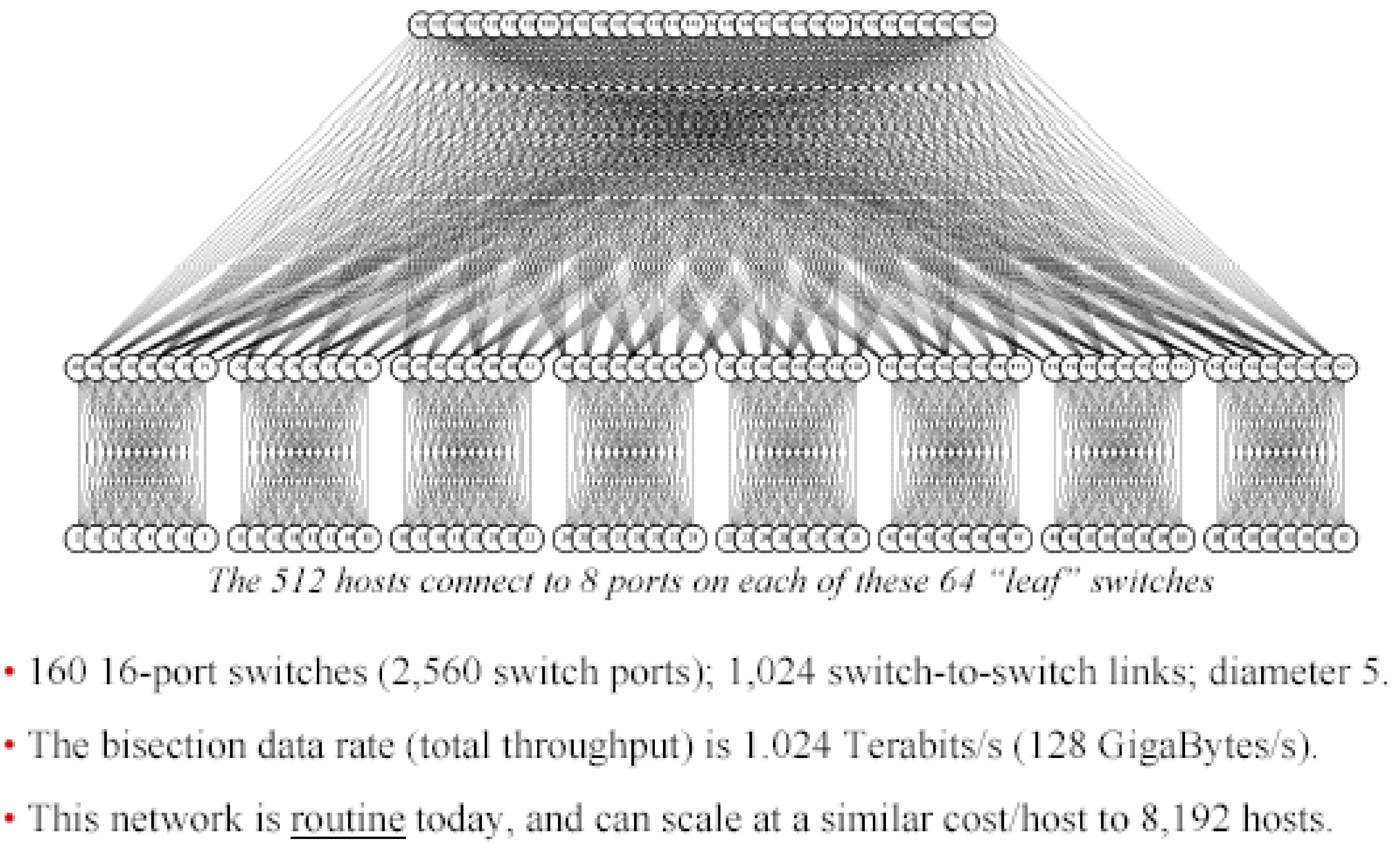


- “Network in a Box”
- 16 fiber line cards connected to the M3-E128 Switch backplane

http://myri.com

Network Topology

Myrinet-2000 Clos Network Extended to 512 Hosts



The 512 hosts connect to 8 ports on each of these 64 "leaf" switches

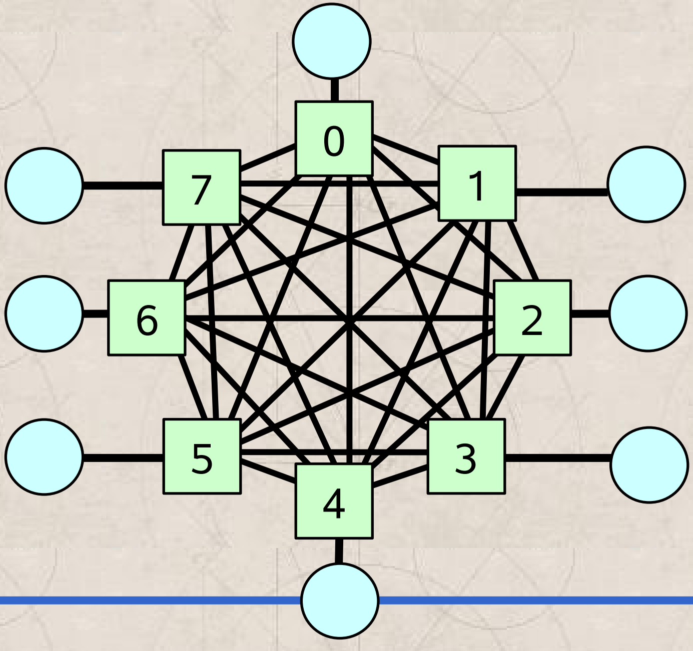
- 160 16-port switches (2,560 switch ports); 1,024 switch-to-switch links; diameter 5.
- The bisection data rate (total throughput) is 1.024 Terabits/s (128 GigaBytes/s).
- This network is routine today, and can scale at a similar cost/host to 8,192 hosts.

http://myri.com

Network Topology

Distributed Switched (Direct) Networks

- Tight integration of end node devices with network resources
  - Network switches distributed among end nodes
  - A “node” now consists of a network switch with *one or more end node devices directly connected* to it
  - Nodes are directly connected to other nodes
- ***Fully-connected network***: all nodes are directly connected to all other nodes using bidirectional dedicated links



Network Topology

Distributed Switched (Direct) Networks

- *Example:* Compute the switch and link costs of interconnecting *N* nodes using a fully connected topology relative to a crossbar, assuming
  - Cost of a *k* x *k* switch grows quadraticaly with the number of unidirectional ports
  - Cost of a 1 x *k* switch grows only linearly

$$\text{Relative cost}_{\text{switches}} = \frac{2N(N-1)}{N^2} = 2\left(1 - \frac{1}{N}\right)$$

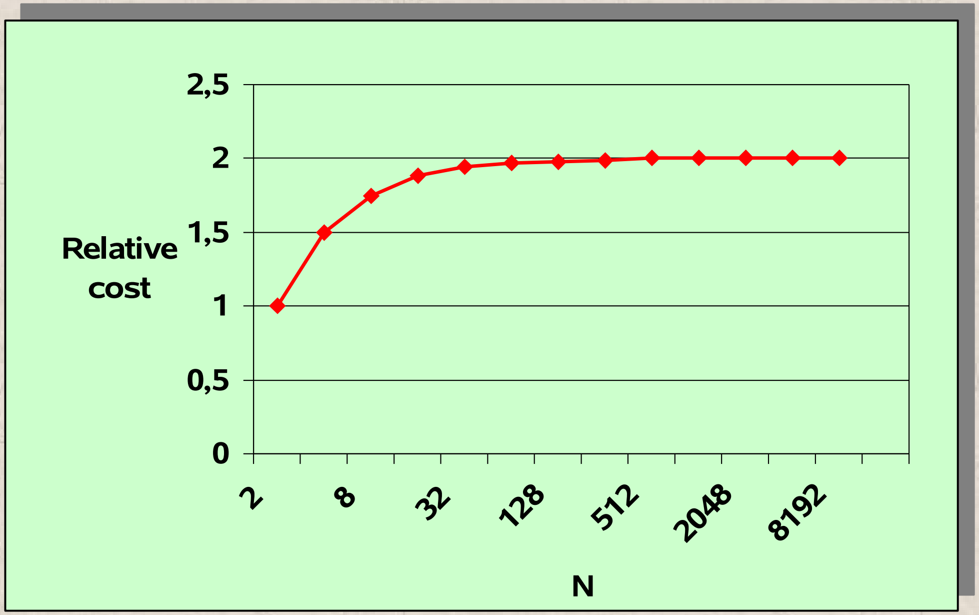
$$\text{Relative cost}_{\text{links}} = \frac{N(N+1)}{2N} = \frac{N+1}{2}$$

- As *N* increases, the switch cost nearly doubles the crossbar's
- Link cost is always higher than a crossbar's
- ***No extra benefits*** *of a fully connected network over a crossbar!*

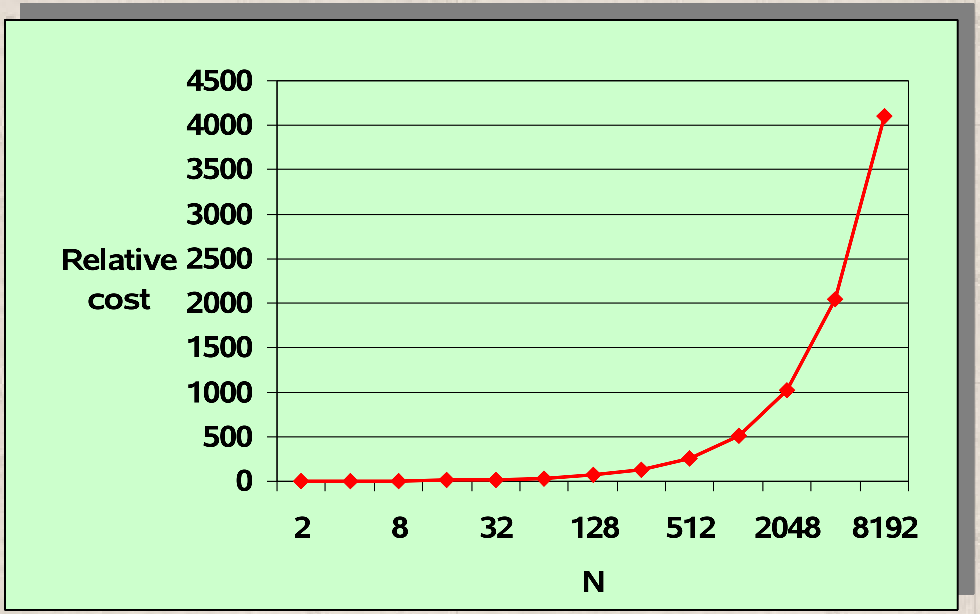
Network Topology

Distributed Switched (Direct) Networks

– *Example:* Compute the switch and link costs of interconnecting *N* nodes using a fully connected topology relative to a crossbar



Relative switch cost

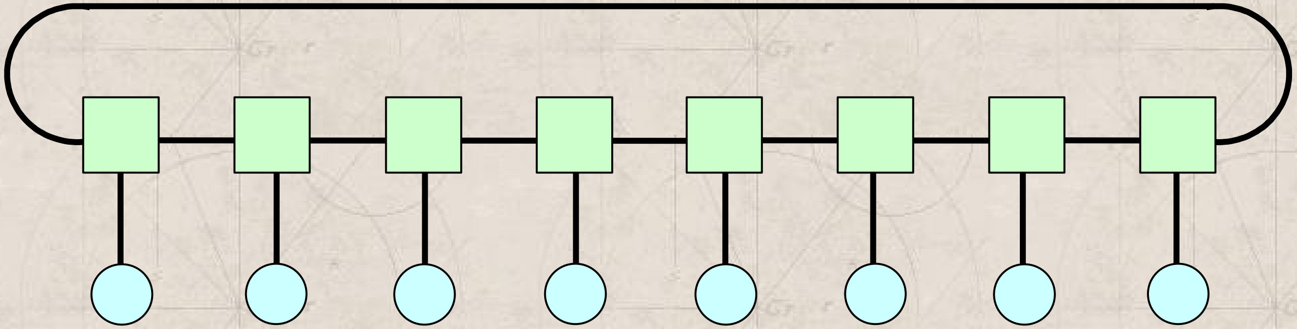


Relative link cost

Network Topology

Distributed Switched (Direct) Networks

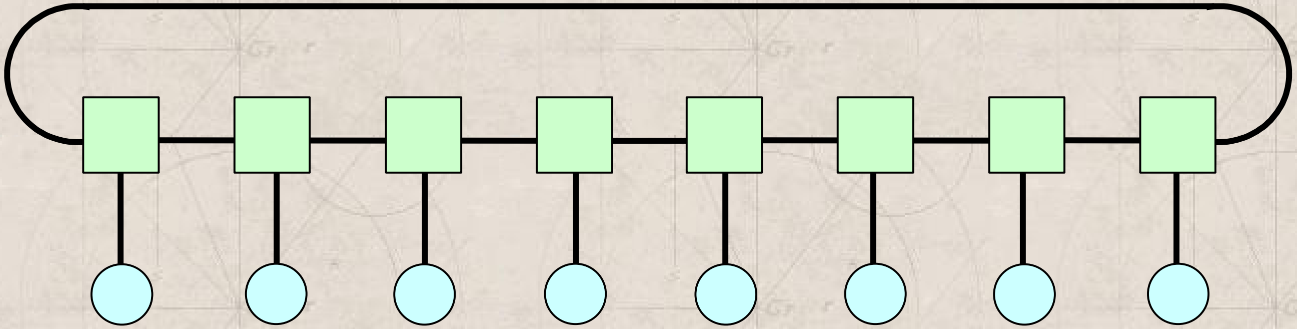
- ***Bidirectional Ring networks***
  - – *N* switches (3 × 3) and *N* bidirectional network links
  - – Simultaneous packet transport over disjoint paths
  - – Packets must hop across intermediate nodes
  - – Shortest direction usually selected (*N*/4 hops, on average)

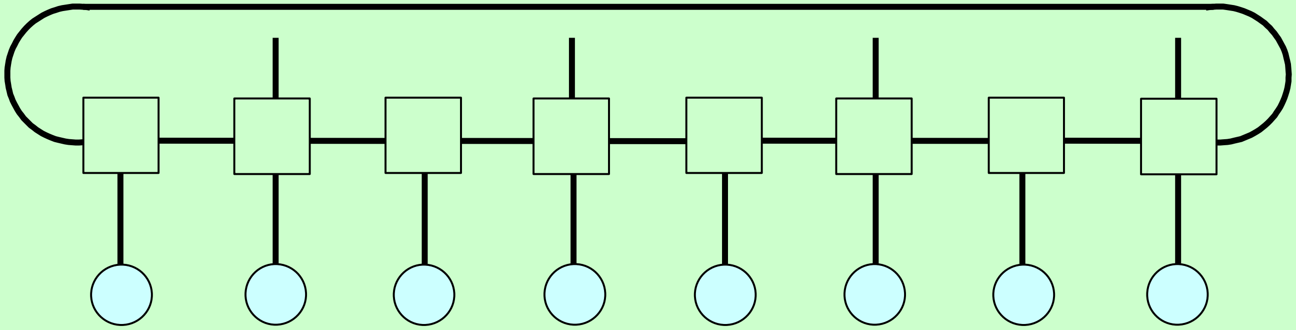


Network Topology

Distributed Switched (Direct) Networks

- ***Bidirectional Ring networks (folded)***
  - – *N* switches (3 × 3) and *N* bidirectional network links
  - – Simultaneous packet transport over disjoint paths
  - – Packets must hop across intermediate nodes
  - – Shortest direction usually selected (*N*/4 hops, on average)



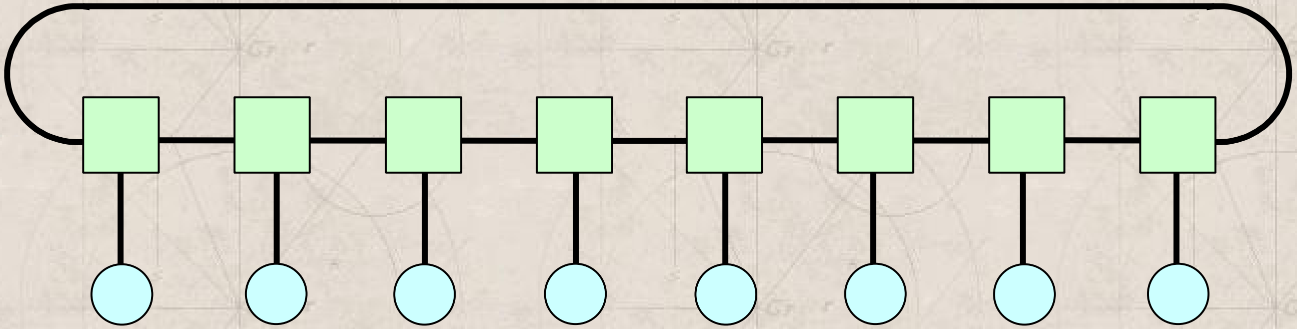


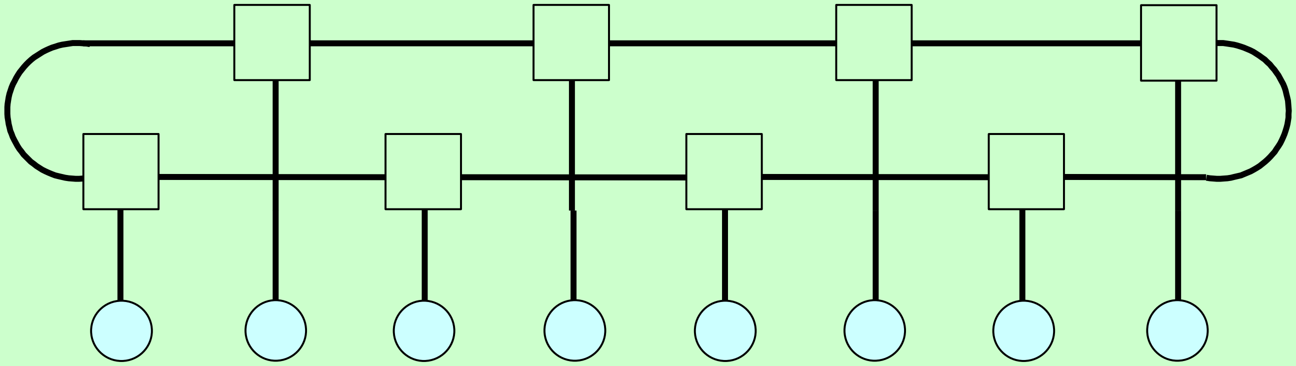
***Folded ring*: Lower maximum physical link length**

Network Topology

Distributed Switched (Direct) Networks

- ***Bidirectional Ring networks (folded)***
  - – *N* switches (3 × 3) and *N* bidirectional network links
  - – Simultaneous packet transport over disjoint paths
  - – Packets must hop across intermediate nodes
  - – Shortest direction usually selected (*N*/4 hops, on average)





***Folded ring*: Lower maximum physical link length**

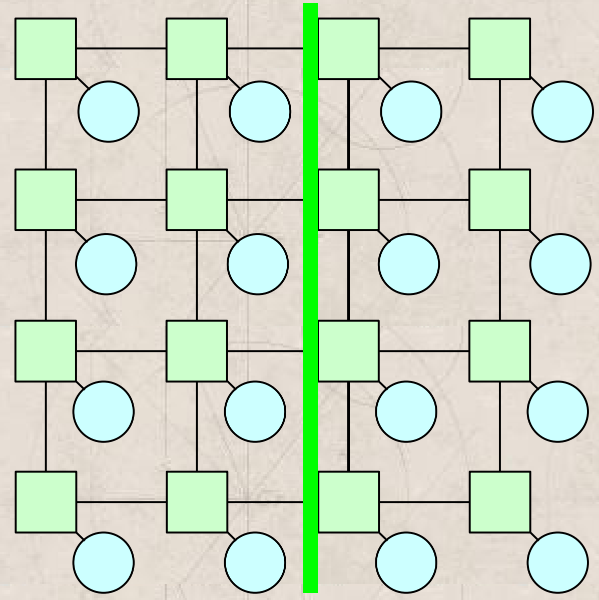
Network Topology

Distributed Switched (Direct) Networks:

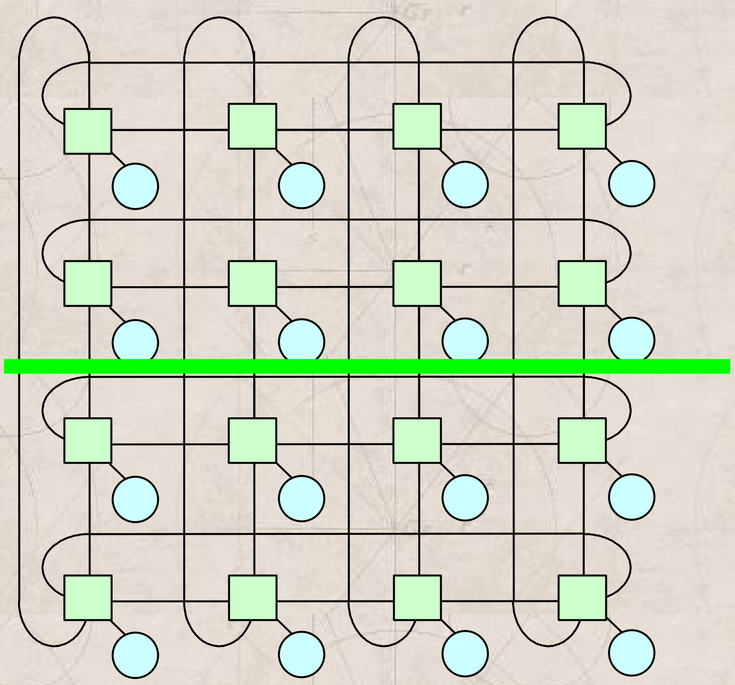
- *Fully connected and ring topologies delimit the two extremes*
- ***The ideal topology:***
  - Cost approaching a ring
  - Performance approaching a fully connected (crossbar) topology
- More practical topologies:
  - ***k-ary n-cubes*** (*meshes*, *tori*, *hypercubes*)
    - ***k*** nodes connected in each dimension, with ***n*** total dimensions
    - *Symmetry* and *regularity*
      - network implementation is simplified
      - routing is simplified

Network Topology

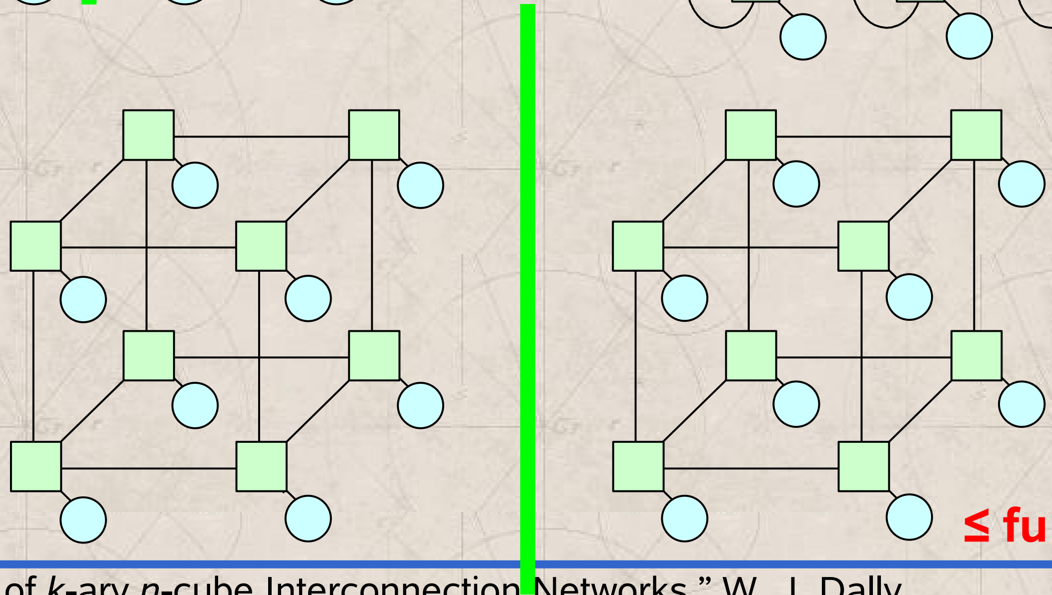
Distributed Switched (Direct) Networks



2D ***mesh*** or grid of 16 nodes



2D ***torus*** of 16 nodes



hypercube of 16 nodes (16 = 2^4, so n = 4)

Network Bisection

≤ full bisection bandwidth!

"Performance Analysis of *k*-ary *n*-cube Interconnection Networks," W. J. Dally, *IEEE Trans. on Computers*, Vol. 39, No. 6, pp. 775–785, June, 1990.

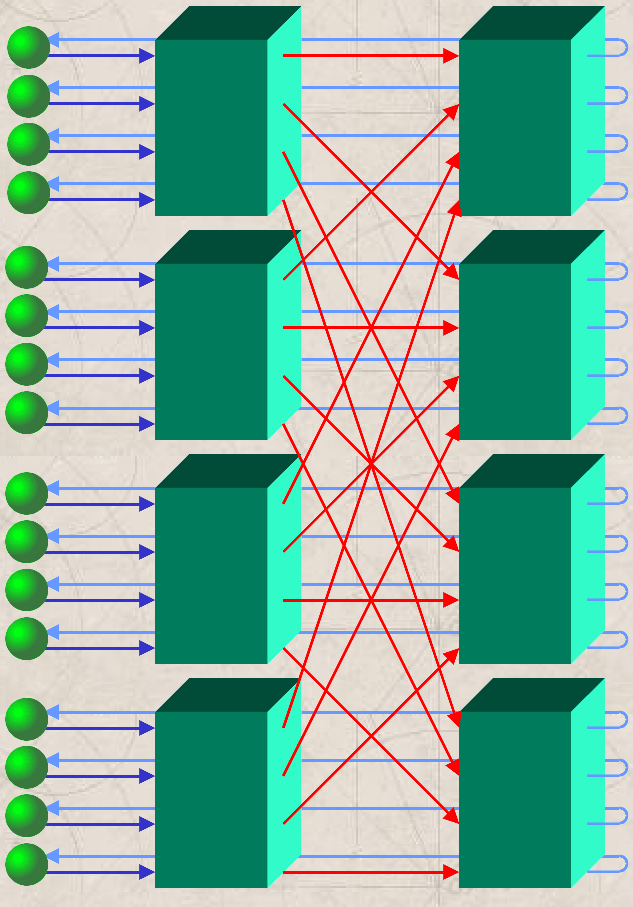
Network Topology

Comparison of Indirect and Direct Networks

- *Indirect networks have end nodes connected at network periphery*

N = 16, *k* = 4
fat tree-like MIN

Switch Cost = 128
Link Cost = 48



Network Topology

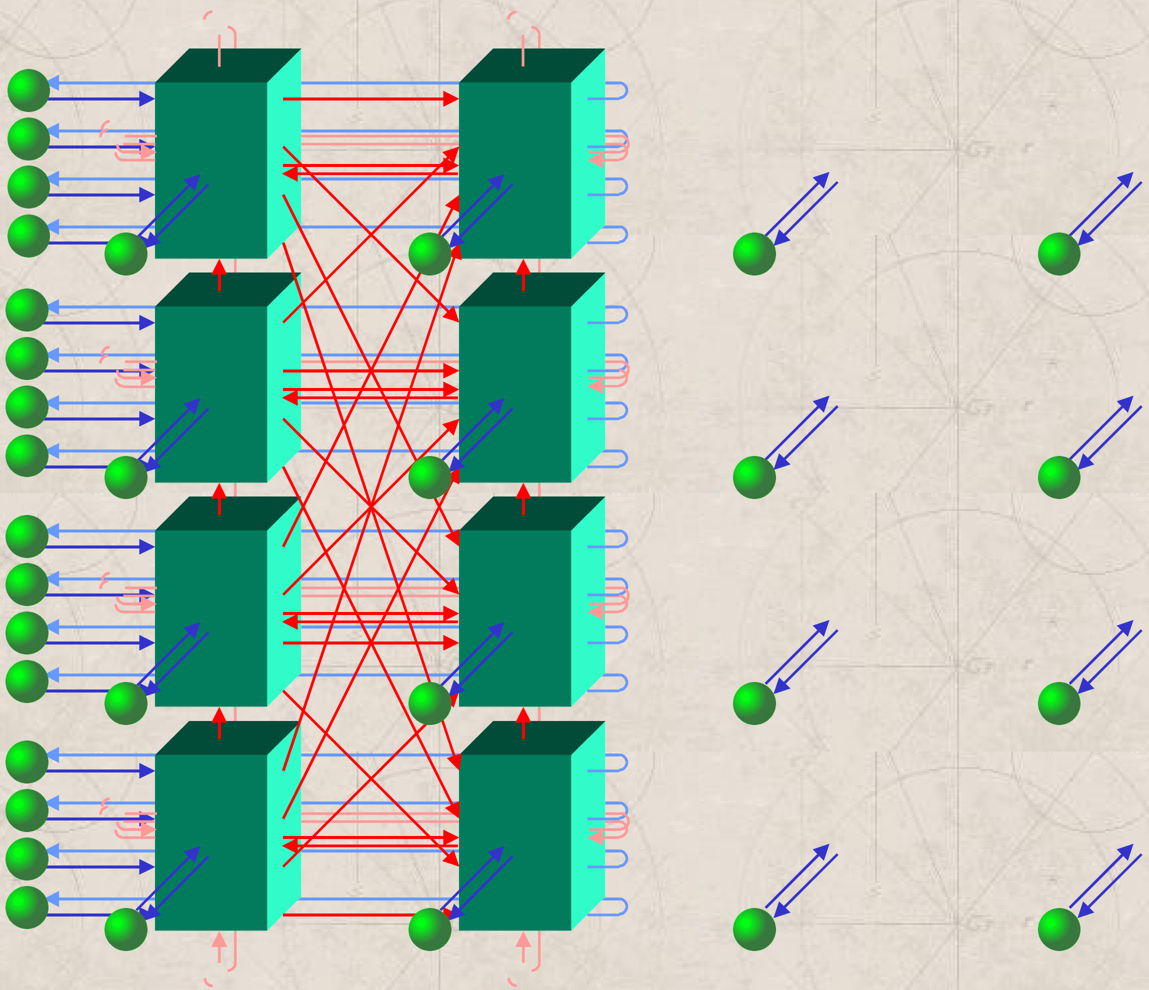
Comparison of Indirect and Direct Networks

- *Direct networks have end nodes connect in network area/volume*

N = **8**, *k* = 4
2D torus

Switch Cost = 128
Link Cost = 48

Switch Cost = 128
Link Cost = 48 → ***40***



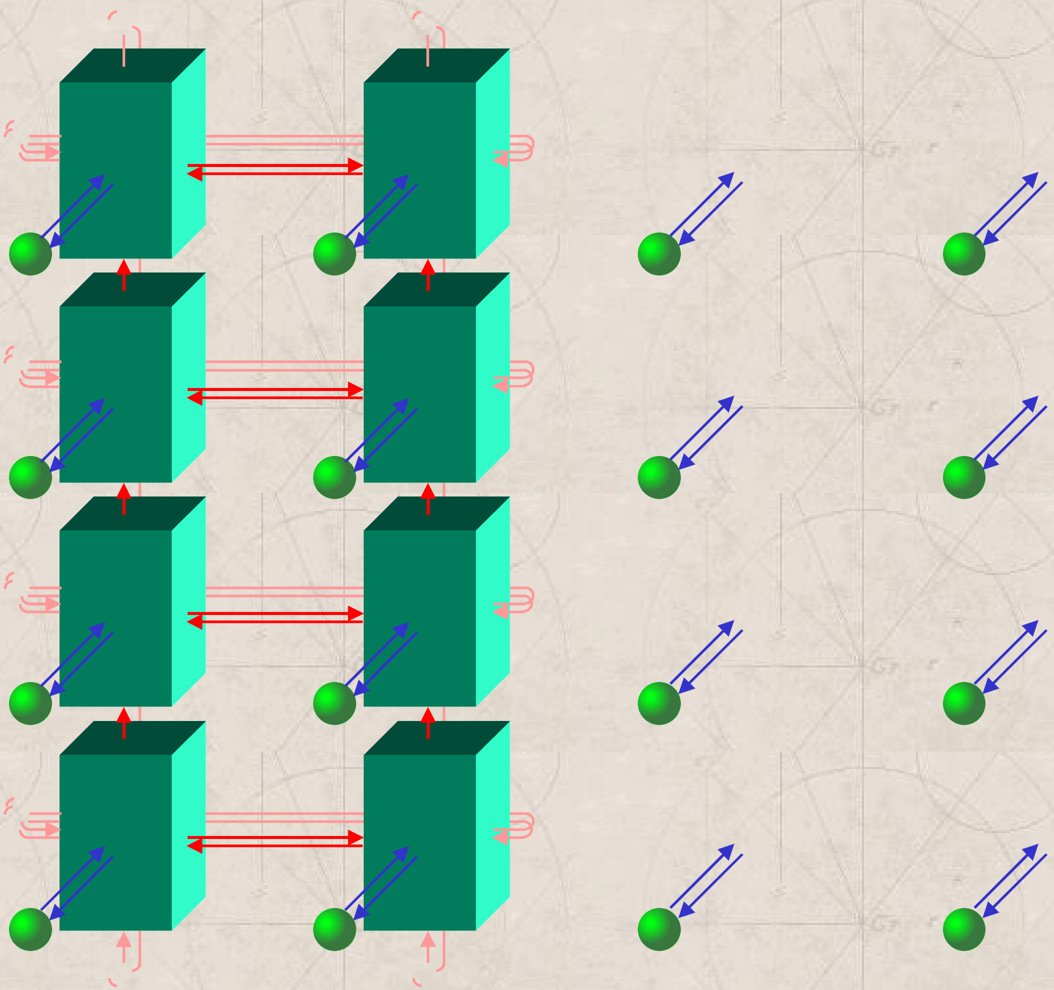
Network Topology

Comparison of Indirect and Direct Networks

- *Direct networks have end nodes connect in network area/volume*

N = **8**, *k* = 4
2D torus

Switch Cost = 128
Link Cost = 48 → ***40***



Network Topology

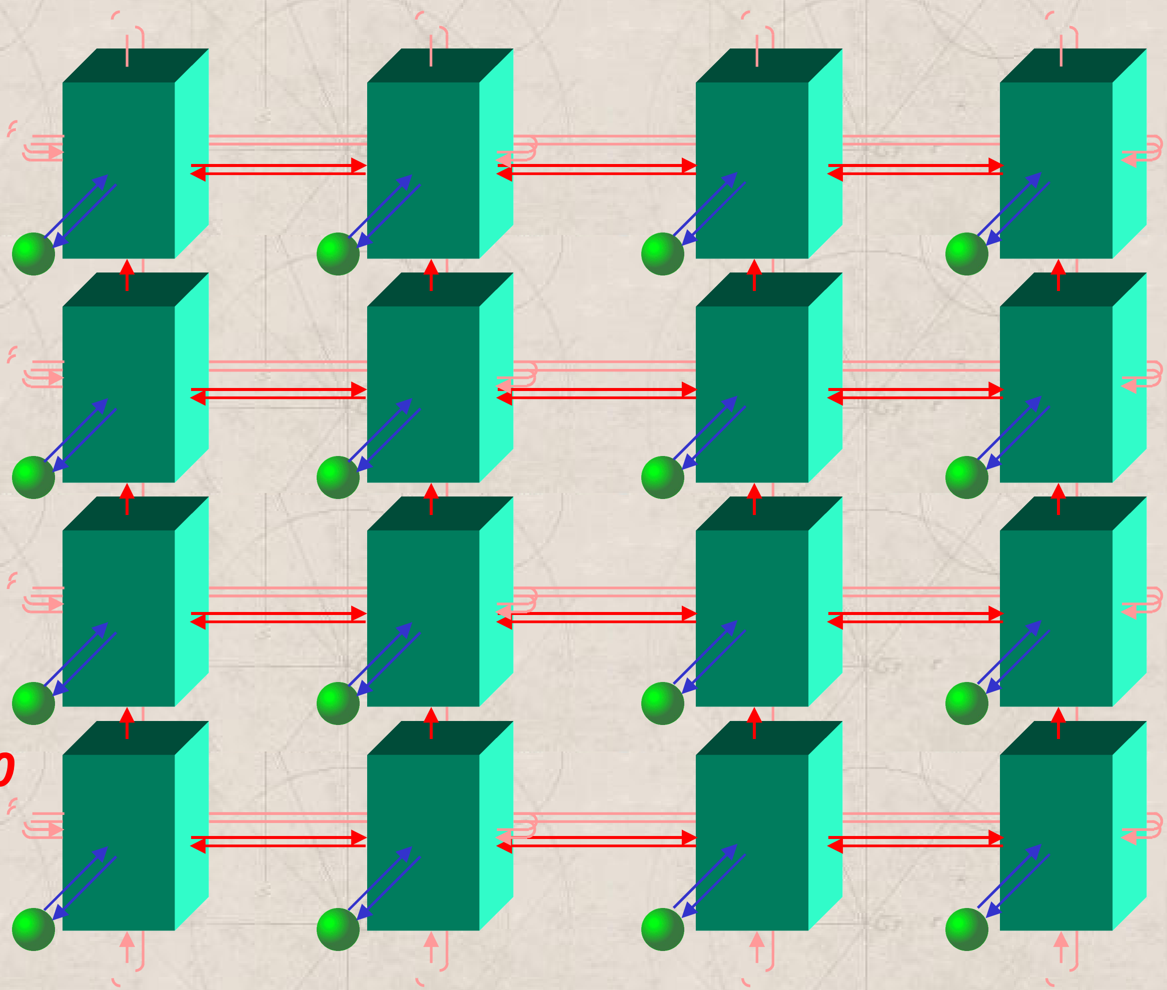
Comparison of Indirect and Direct Networks

- *Direct networks have end nodes connect in network area/volume*

N = **16**, *k* = 4
2D torus

Switch Cost = 128
Link Cost = 48 → ***40***

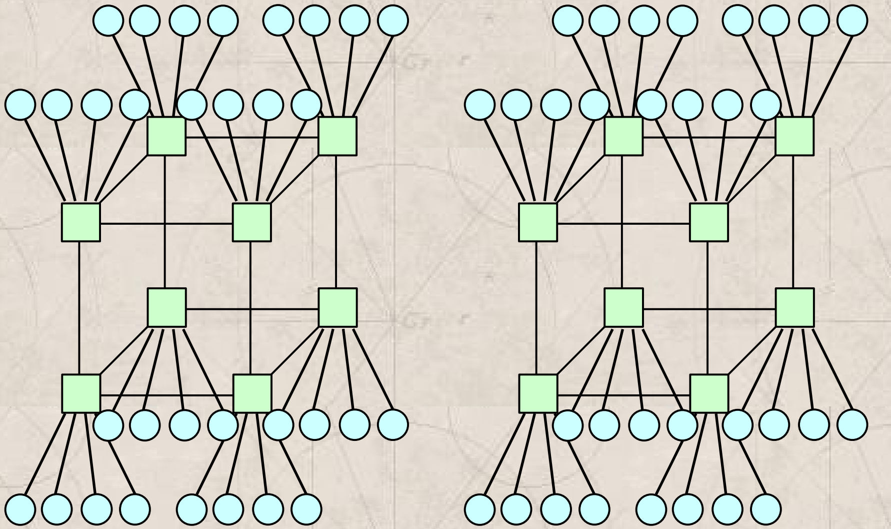
Switch Cost = 128 → ***256***
Link Cost = 48 → 40 → ***80***



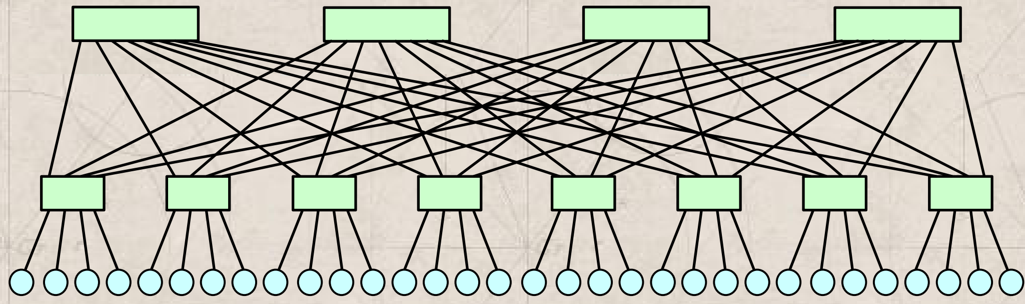
Network Topology

Comparison of Indirect and Direct Networks

- *Bristling* can be used to reduce direct network switch & link costs
  - “*b*” end nodes connect to each switch, where *b* is bristling factor
  - Allows larger systems to be built from fewer switches and links
  - Requires larger *switch degree*
  - For *N* = 32 and *k* = 8, fewer switches and links than fat tree



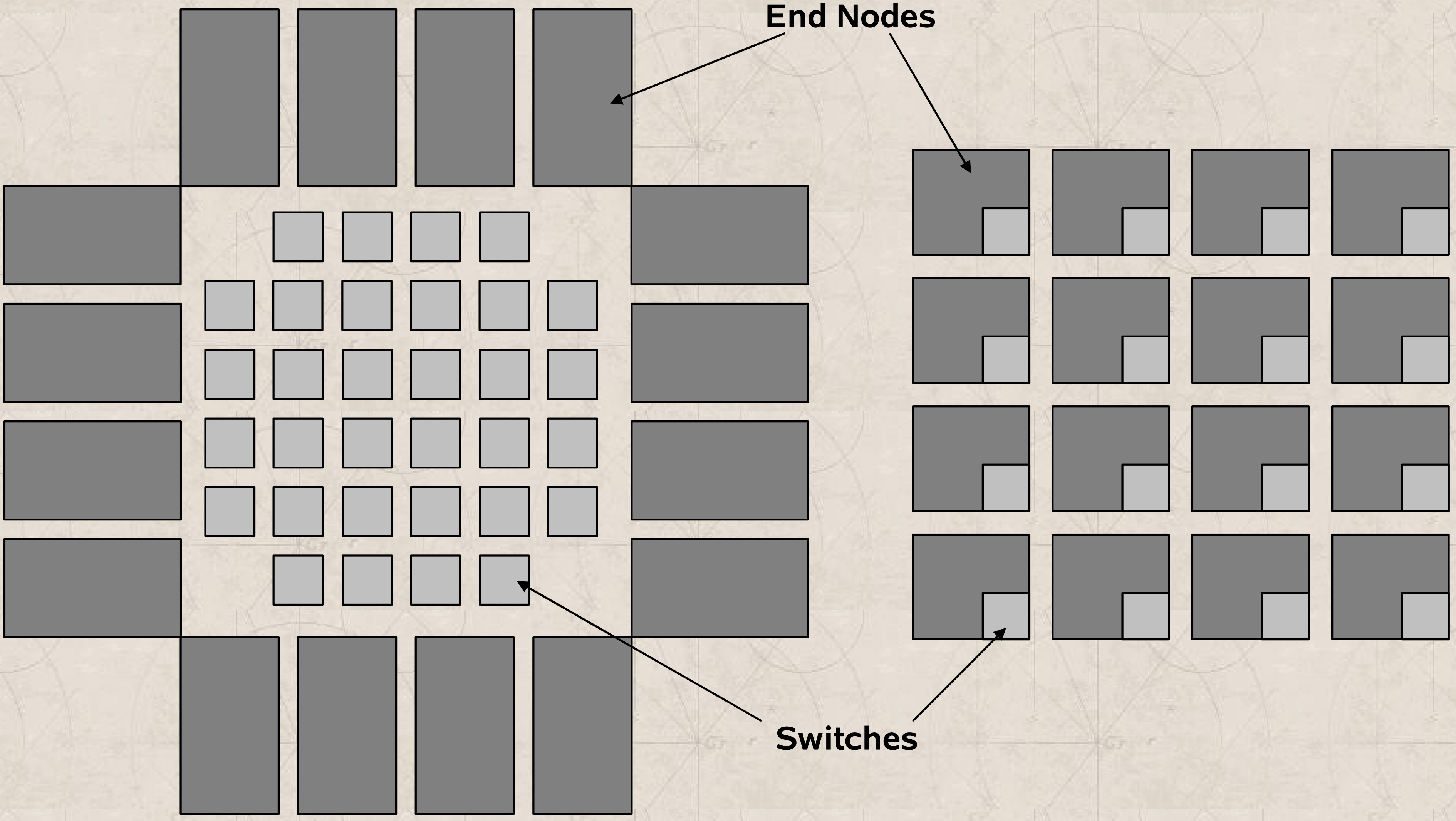
***64-node* system with *8-port* switches, *b* = 4**



***32-node* system with *8-port* switches**

Network Topology

Comparison of Indirect and Direct Networks



Distance scaling problems may be exacerbated in on-chip MINs

Network Topology

Comparison of Indirect and Direct Networks

- *Example:* Compute the switch and link costs of interconnecting *N* nodes using a torus (*without bristling*) relative to a fat tree, assuming
  - *k* x *k* switches (fat tree), *n* dimensions (torus)

$$\text{relative\_cost}_{\text{switches}} = \frac{(2\text{n}+1)^2 N}{2\text{kNlog}_{\text{k}/2} N} = \frac{(2\text{n}+1)^2}{2\text{klog}_{\text{k}/2} N} = \frac{\overset{2\text{n}+1 \simeq k}{\overset{\downarrow}{k}}}{2\text{log}_{\text{k}/2} N}$$

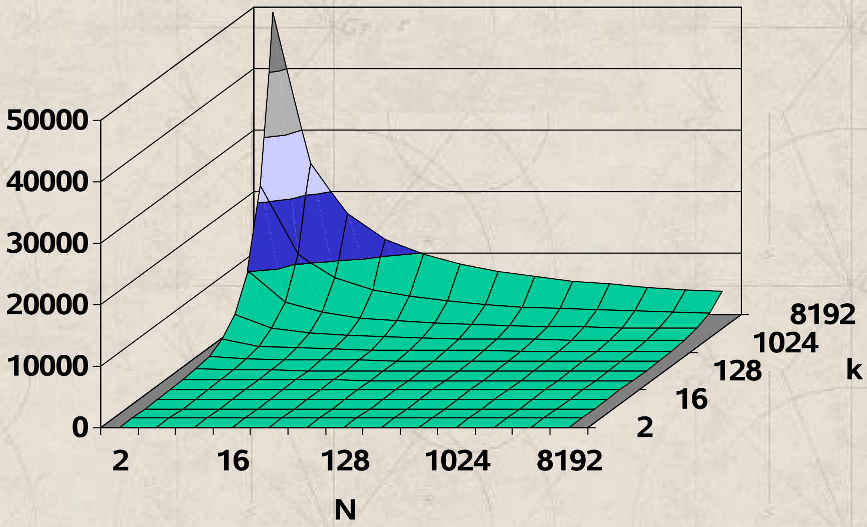
$$\text{relative\_cost}_{\text{links}} = \frac{(n+1)N}{\text{Nlog}_{\text{k}/2} N} = \frac{n+1}{\text{log}_{\text{k}/2} N}$$

- If switch degree (*k*) is low relative to *N*, tori have lower cost
- If switch degree (*k*) is high relative to *N*, fat trees have lower cost
- For *N* = 256 and *k* = 4, fat tree is *four times more expensive!!*
- For *N* = 256 and *k* = 8, fat tree is comparable in cost to torus (3D)

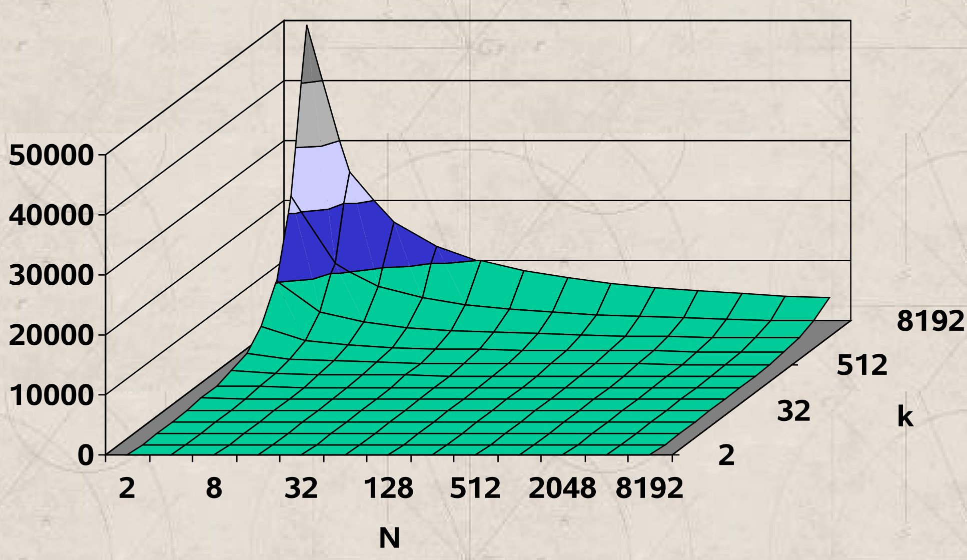
Network Topology

Comparison of Indirect and Direct Networks

- *Example:* Compute the switch and link costs of interconnecting *N* nodes using a torus (*without bristling*) relative to using a fat tree



Relative switch cost



Relative link cost

Network Topology

Comparison of Indirect and Direct Networks

- Blocking reduced by maximizing dimensions (switch degree)
  - Can increase bisection bandwidth, but
    - › Additional dimensions may increase wire length (must observe 3D packaging constraints)
    - › Flow control issues (buffer size increases with link length)
    - › Pin-out constraints (limit the number of dimensions achievable)

	Evaluation category	**Bus**	**Ring**	**2D mesh**	**2D torus**	**Hypercube**	**Fat tree**	**Fully connected**
Perf.	**$BW_{Bisection}$ in # links**	**1**	**2**	**8**	**16**	**32**	**32**	**1024**
	Max (ave.) hop count	**1 (1)**	**32 (16)**	**14 (7)**	**8 (4)**	**6 (3)**	**11 (9)**	**1 (1)**
Cost	**I/O ports per switch**	**NA**	**3**	**5**	**5**	**7**	**4**	**64**
	Number of switches	**NA**	**64**	**64**	**64**	**64**	**192**	**64**
	Number of net. links	**1**	**64**	**112**	**128**	**192**	**320**	**2016**
	Total number of links	**1**	**128**	**176**	**192**	**256**	**384**	**2080**

Performance and cost of several network topologies for 64 nodes. Values are given in terms of bidirectional links & ports. Hop count includes a switch and its output link (in the above, end node links are not counted for the bus topology).

Network Topology

Characterizing Performance: Latency & Effective Bandwidth

- Topology affects the number of hops, **d**, experienced by packets
  - Transport functions (propagation, routing, switching, arbitration, and transmission) are performed on each hop through switches
- Topology affects $BW_{Bisection}$; affects γ only for bus & dedicated-link
- Network traffic pattern determines γ

$$Latency = Sending\ overhead + T_{LinkProp} \times (d + 1) + (T_r + T_s + T_a) \times d + \frac{Packet\ size}{Bandwidth} \times (d + 1) + Receiving\ overhead$$

lower bound (contention delay not included)

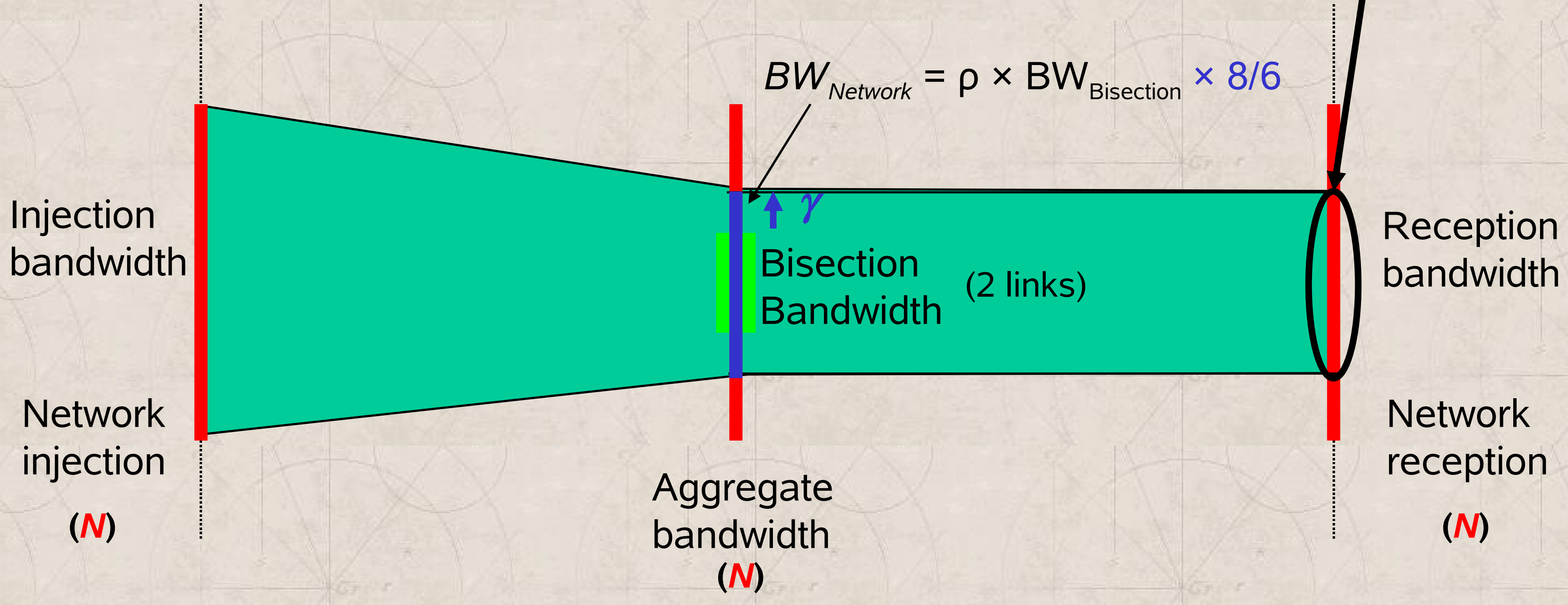
T_r = per switch routing delay
T_a = per switch arbitration delay
T_s = per switch switching delay
$T_{LinkProp}$ = per link propagation delay
d = hop count (distance)

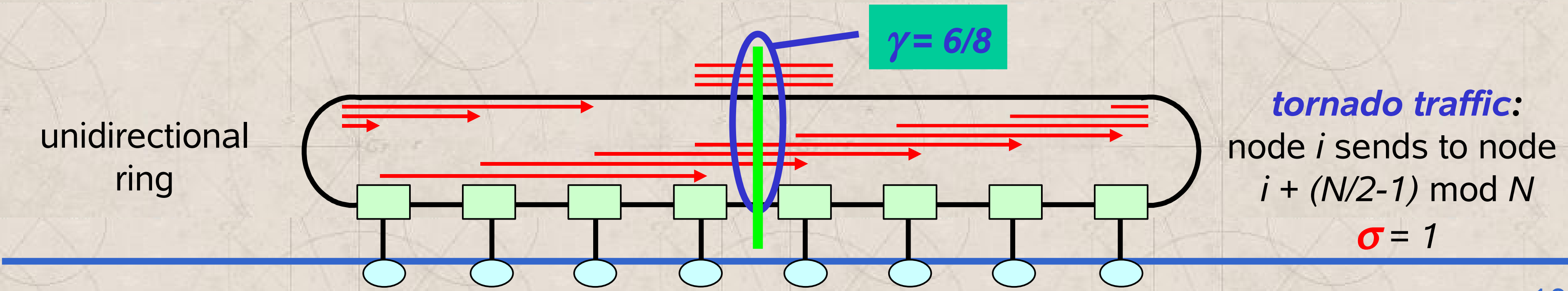
upper bound (contention effects not *fully* included)

$$Effective\ bandwidth = min(N \times BW_{LinkInjection},\ \underbrace{\frac{\rho \times BW_{Bisection}}{\gamma}}_{BW_{Network}},\ \sigma \times N \times BW_{LinkReception})$$

Network Topology

Characterizing Performance: Latency & Effective Bandwidth

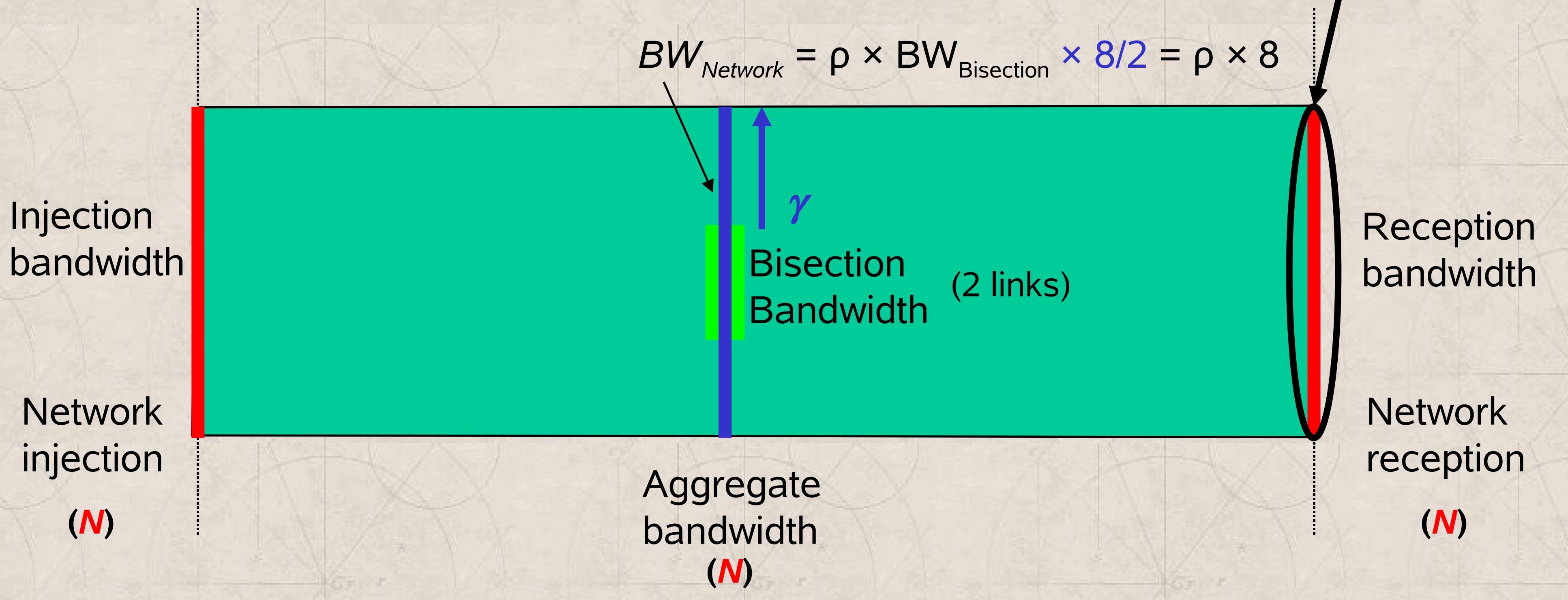


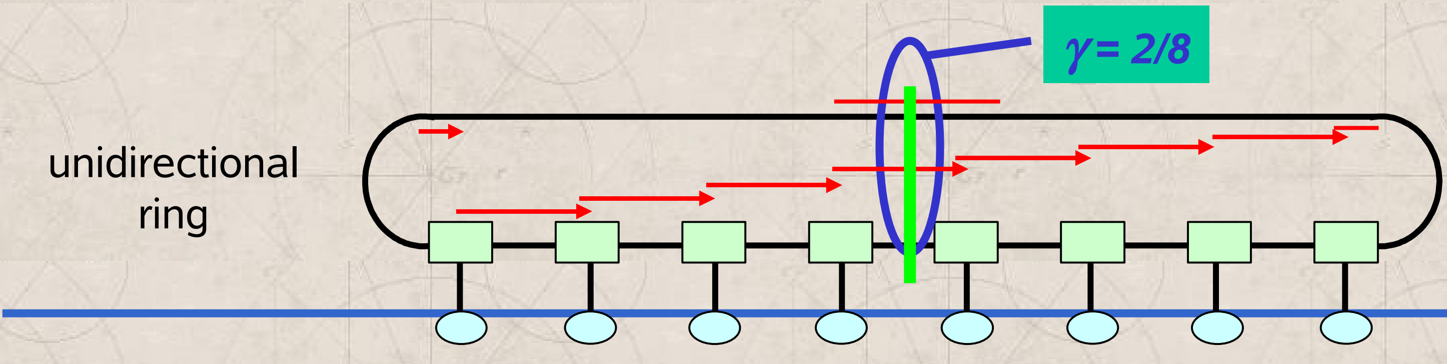


***tornado traffic*:**
node *i* sends to node
i + (N/2-1) mod *N*
$\sigma = 1$

Network Topology

Characterizing Performance: Latency & Effective Bandwidth





***near-neighbour traffic*:**
node *i* sends to node
(i + 1) mod *N*
$\sigma = 1$

Network Topology

Topological Characteristics of Commercial Machines

Company	System [Network] Name	Max. number of nodes [x # CPUs]	Basic network topology	Injection [Recept'n] node BW in MBytes/s	# of data bits per link per direction	Raw network link BW per direction in Mbytes/sec	Raw network bisection BW (bidir) in Gbytes/s
Intel	ASCI Red Paragon	4,510 [x 2]	2-D mesh 64 x 64	400 [400]	16 bits	400	51.2
IBM	ASCI White SP Power3 [Colony]	512 [x 16]	BMIN w/8-port bidirect. switches (fat-tree or Omega)	500 [500]	8 bits (+1 bit of control)	500	256
Intel	Thunter Itanium2 Tiger4 [QsNet^{II}]	1,024 [x 4]	fat tree w/8-port bidirectional switches	928 [928]	8 bits (+2 control for 4b/5b enc)	1,333	1,365
Cray	XT3 [SeaStar]	30,508 [x 1]	3-D torus 40 x 32 x 24	3,200 [3,200]	12 bits	3,800	5,836.8
Cray	X1E	1,024 [x 1]	4-way bristled 2-D torus (~ 23 x 11) with express links	1,600 [1,600]	16 bits	1,600	51.2
IBM	ASC Purple pSeries 575 [Federation]	>1,280 [x 8]	BMIN w/8-port bidirect. switches (fat-tree or Omega)	2,000 [2,000]	8 bits (+2 bits of control)	2,000	2,560
IBM	Blue Gene/L eServer Sol. [Torus Net]	65,536 [x 2]	3-D torus 32 x 32 x 64	612,5 [1,050]	1 bit (bit serial)	175	358.4

Outline

- E.1 Introduction *(Lecture 1)*
- E.2 Interconnecting Two Devices *(Lecture 1)*
- E.3 Interconnecting Many Devices *(Lecture 2)*
- E.4 Network Topology *(Lecture 2)*
- **E.5 Network Routing, Arbitration, and Switching** *(Lecture 3)*
  - **Routing**
  - **Arbitration**
  - **Switching**
  - **Characterizing Performance: Latency & Effective Bandwidth**
- E.6 Switch Microarchitecture *(Lecture 4)*
- E.7 Practical Issues for Commercial Interconnection Networks *(Lec 4)*
- E.8 Examples of Interconnection Networks *(Lecture 5)*
- E.9 Internetworking (skipped)
- E.10 Crosscutting Issues for Interconnection Networks (skipped)
- E.11 Fallacies and Pitfalls *(Lecture 5)*
- E.12 Concluding Remarks and References *(Lecture 5)*

Routing, Arbitration, and Switching

Routing

- Performed at each switch, regardless of topology
- Defines the "allowed" path(s) for each packet (*Which paths?*)
- Needed to *direct packets through network* to intended destinations
- ***Ideally:***
  - *Supply as many routing options to packets as there are paths provided by the topology, and evenly distribute network traffic among network links using those paths, minimizing contention*
- *Problems:* situations that cause packets never to reach their dest.
  - ***Livelock***
    - › Arises from an unbounded number of allowed non-minimal hops
    - › *Solution:* restrict the number of non-minimal (mis)hops allowed
  - ***Deadlock***
    - › Arises from a set of packets being blocked waiting only for network resources (i.e., links, buffers) held by other packets in the set
    - › Probability increases with increased traffic & decreased availability

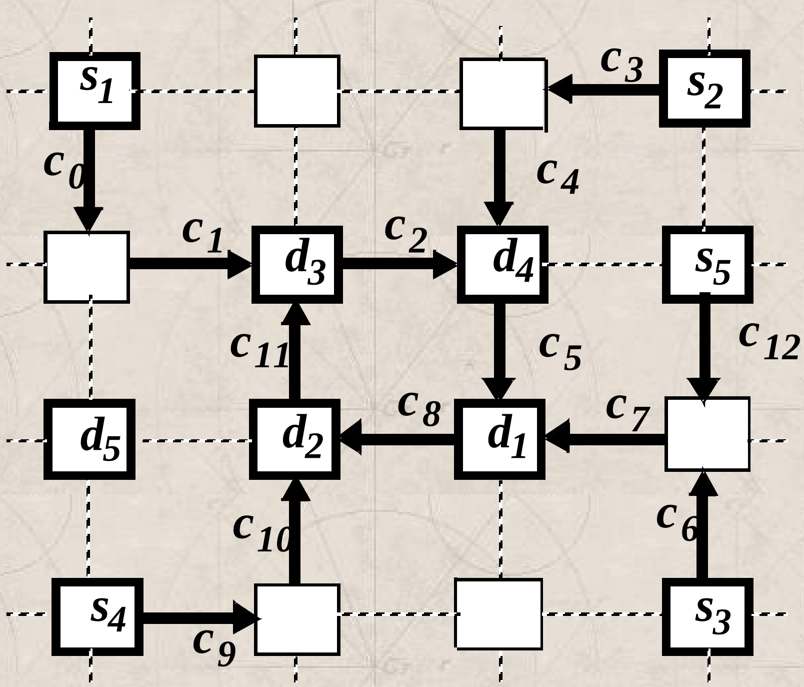
Routing, Arbitration, and Switching

Routing

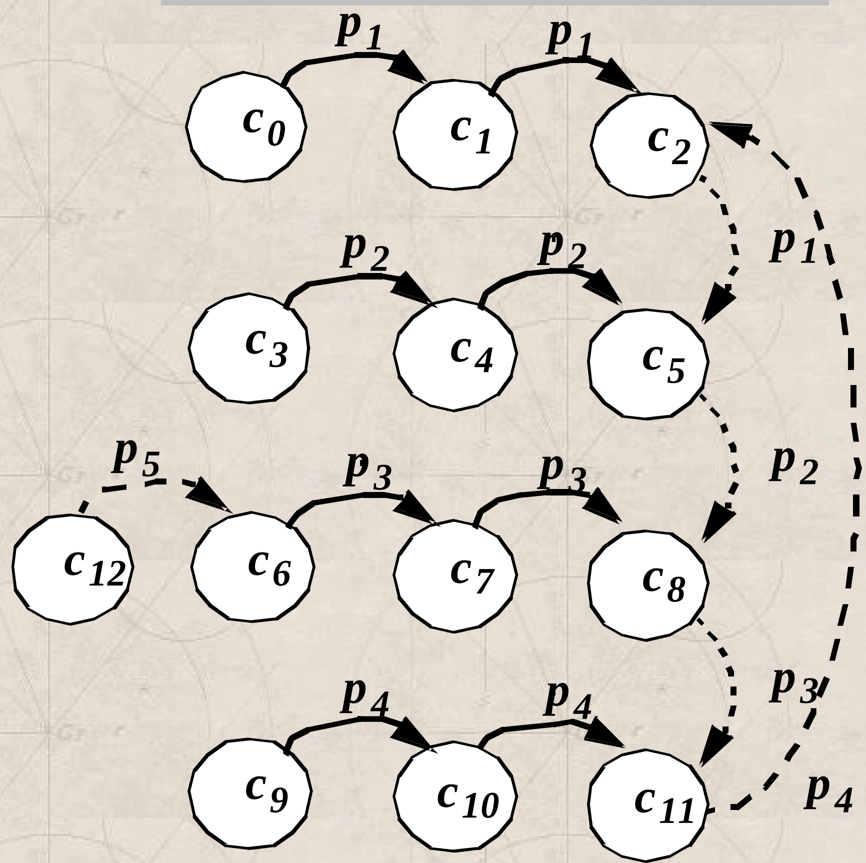
- Common forms of deadlock:
  - *Routing-induced deadlock*

c_i = channel i
s_i = source node i
d_i = destination node i
p_i = packet i

Routing of packets in a 2D mesh



Channel dependency graph



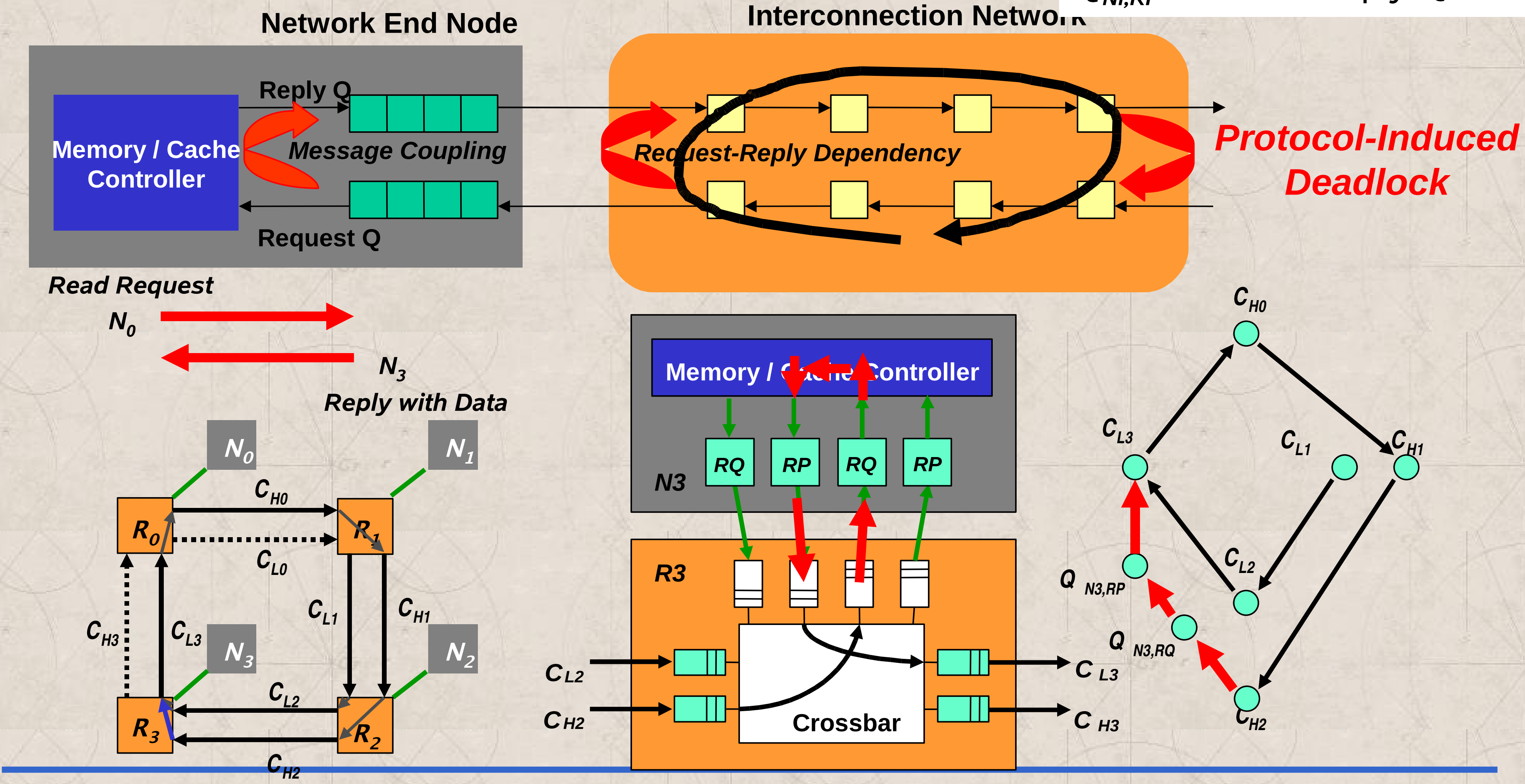
"A Formal Model of Message Blocking and Deadlock Resolution in Interconnection Networks," S. Warnakulasuriya and T. Pinkston, *IEEE Trans. on Parallel and Distributed Systems* , Vol. 11, No. 3, pp. 212–229, March, 2000.

Routing, Arbitration, and Switching

Routing

- Common forms of deadlock:
  - ***Protocol (Message)-induced deadlock***

C_{Hi} = high-ordered channel i
C_{Li} = low-ordered channel i
$Q_{Ni,RQ}$ = node i Request Q
$Q_{Ni,RP}$ = node i Reply Q

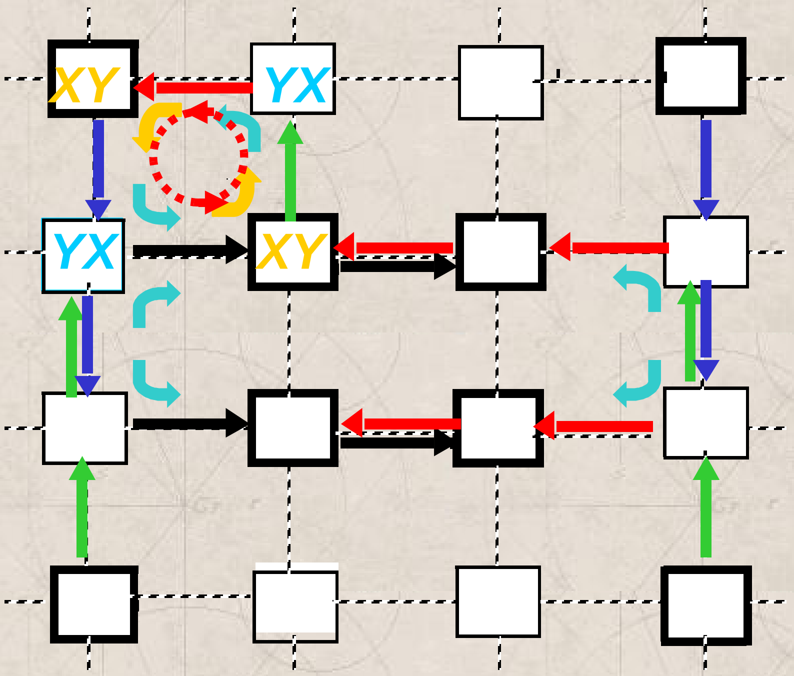


"A Progressive Approach to Handling Message-Dependent Deadlocks in Parallel Computer Systems," Y. Song and T. Pinkston, *IEEE Trans. on Parallel and Distributed Systems* , Vol. 14, No. 3, pp. 259–275, March, 2003.

Routing, Arbitration, and Switching

Routing

- Common forms of deadlock:
  - *Fault (Reconfiguration)-induced deadlock*



- The transition from one routing function (***YX*** routing) to another routing function (***XY*** routing) in order to circumvent faults can create cyclic dependencies on resources that are not present in either routing function alone!

"Part I: A Theory for Deadlock-free Dynamic Reconfiguration of Interconnection Networks," J. Duato, O. Lysne, R. Pang, and T. Pinkston, *IEEE Trans. on Parallel and Distributed Systems* , Vol. 16, No. 5, pp. 412–427, May, 2005.

Routing, Arbitration, and Switching

Routing

- Common strategies to deal with all forms of deadlock
  - *Deadlock avoidance:* restrict allowed paths only to those that keep the global state deadlock-free
    - *Duato's Protocol*: always guarantee an *escape path* from deadlock
      - Establish ordering only on a minimal (*escape*) set of resources
      - Grant escape resources in a partial or total order
      - Cyclic dependencies cannot form on escape resources, although cycles may form on larger set of network resources
    - *DOR* (*dimension-order routing*) on meshes and hypercubes
      - Establish ordering on ***all*** resources based on network dimension
    - *DOR* on rings and tori (*k*-ary *n*-cubes with wrap-around links)
      - Ordering on all resources between *and within* each dimension
      - Apply to multiple *virtual channels* (*VCs*) per physical channel
      - Alternatively, keep resources along each dimension from reaching full capacity by ensuring the existence of a *bubble(s)*

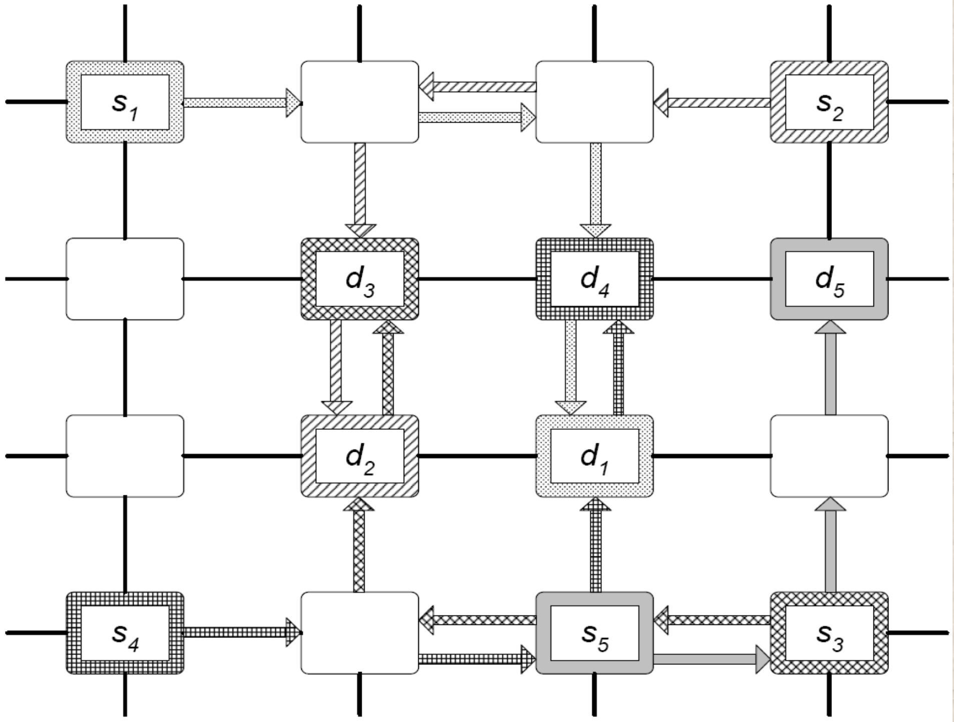
Routing, Arbitration, and Switching

Routing

- Common strategies to deal with deadlock
  - *Deadlock avoidance:*

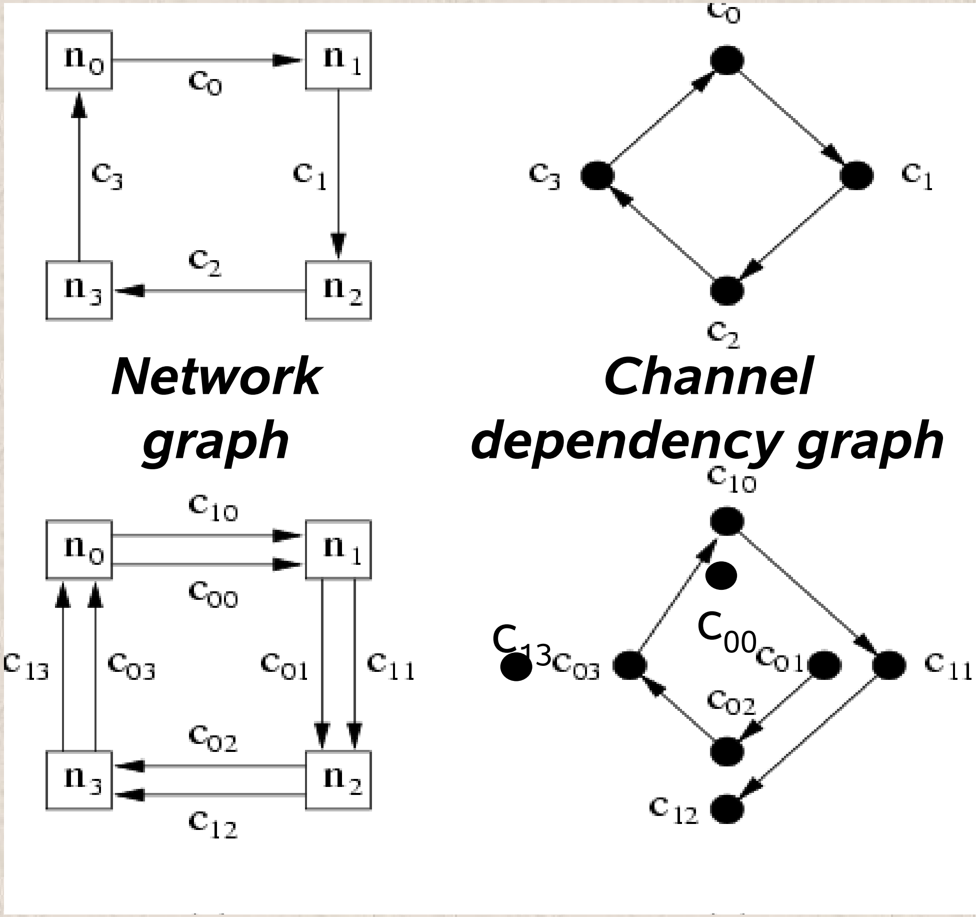
n_i = node ***i***
c_i = physical channel ***i***
c_{1i} = high-ordered VC ***i***
c_{0i} = low-ordered VC ***i***

Deadlock avoidance in 2D mesh using *DOR*



2 VCs per physical channel

Deadlock avoidance in ring using *VCs*



Network graph

Channel dependency graph

"A General Theory for Deadlock-free Adaptive Routing Using a Mixed Set of Resources," J. Duato and T. Pinkston, *IEEE Trans. on Parallel and Distributed Systems* , Vol. 12, No. 12, pp. 1219–1235, December, 2001.

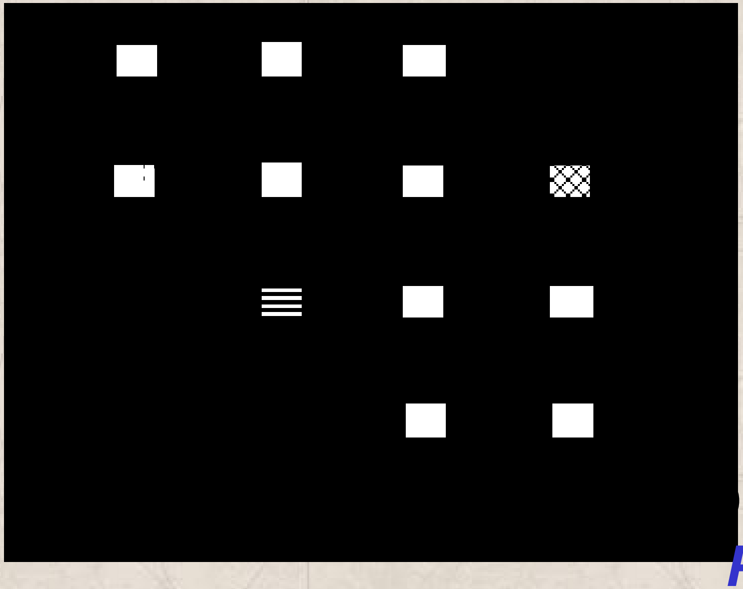
Routing, Arbitration, and Switching

Routing

- Common strategies to deal with all forms of deadlock
  - *Deadlock recovery:* allow deadlock to occur, but once a potential deadlock situation is detected, break at least one of the cyclic dependencies to gracefully recover
    - A mechanism to detect potential deadlock is needed
    - *Regressive recovery* (*abort-and-retry*): remove packet(s) from a dependency cycle by killing (aborting) and later re-injecting (retry) the packet(s) into the network after some delay
    - *Progressive recovery* (preemptive): remove packet(s) from a dependency cycle by rerouting the packet(s) onto a deadlock-free lane
- *Deterministic routing*: routing function always supplies the same path for a given source-destination pair (e.g., *DOR*)
- *Adaptive routing*: routing function allows alternative paths for a given source-destination pair (e.g., *Duato's Protocol, Bubble Adaptive Routing, Disha Routing*)
  - Increases routing freedom to improve network efficiency, ***ρ***

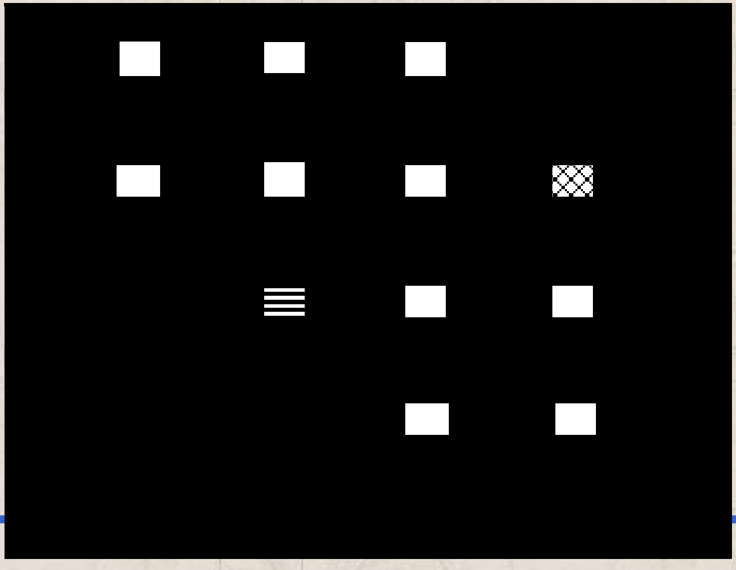
Routing, Arbitration, and Switching

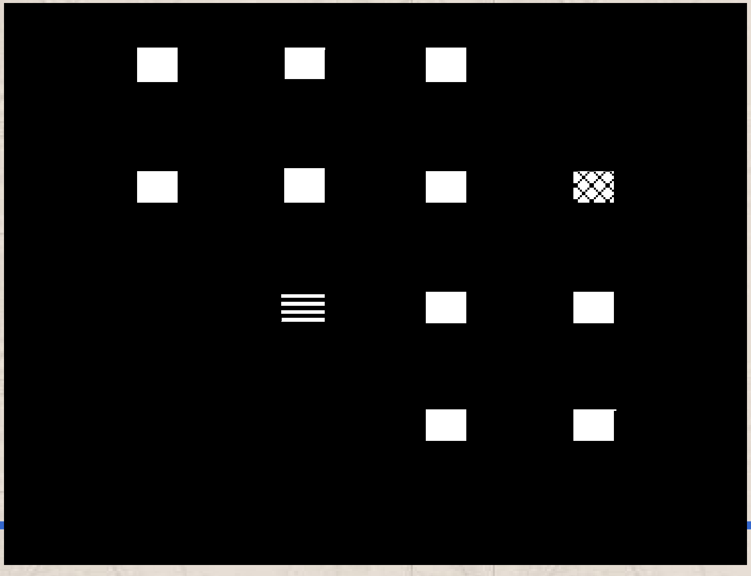
Routing



Comparison of Routing Freedom

Routing freedom can increase ρ *(i.e., ρ_R)*





Routing, Arbitration, and Switching

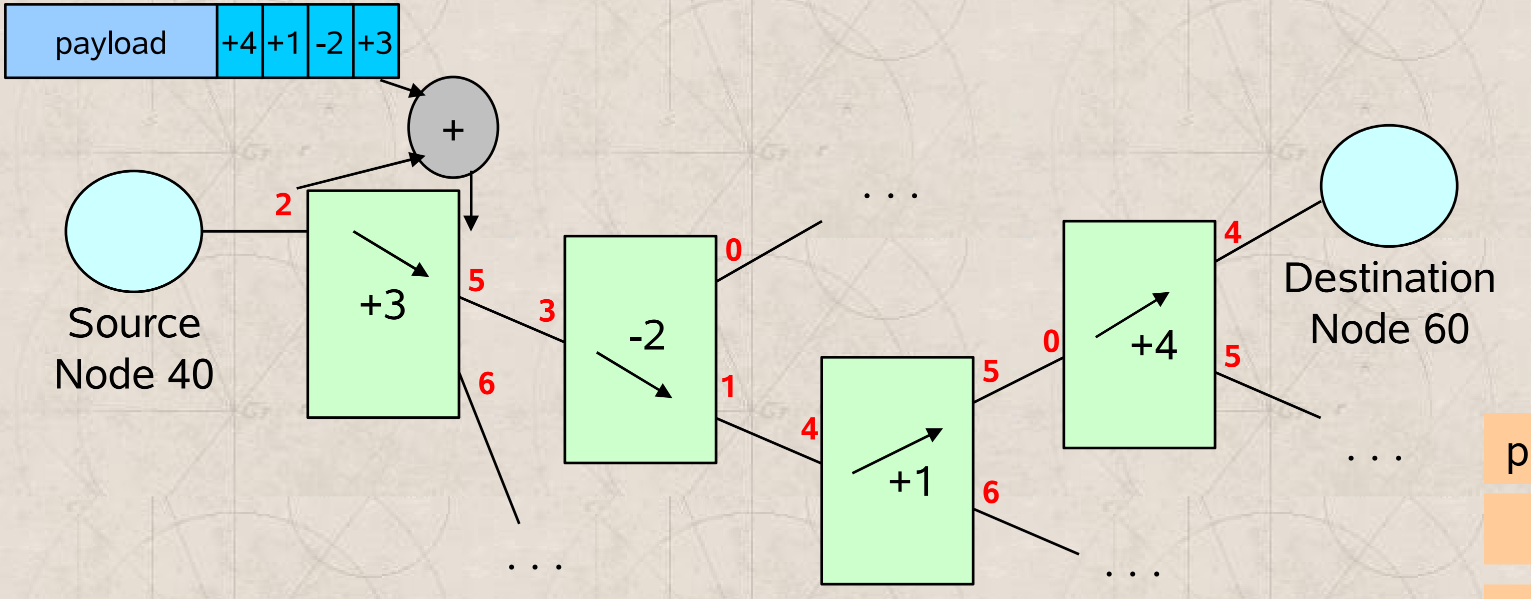
Routing

- Routing in centralized switched (indirect) networks
  - *Least common ancestor* (*LCA*) *routing*
    - Applicable to fat tree and other bidirectional MINs
    - Use resources in some partial order to avoid cycles, deadlock
    - Reach any LCA switch through any one of multiple paths
    - Traverse down the tree to destination through a deterministic path
    - *Self routing property*: switch output port at each hop is given by shifts of the destination node address (least significant bit/digit)
  - *Up*/down* routing*:
    - Universally applicable to *any* topology: map a tree graph onto it
    - Assign "up" and "down" directions to network links (or VCs)
    - Allowed paths to destination consist of zero or more "up" traversals followed by zero or more "down" traversals
    - Up-down traversals impose partial order to avoid cycles, deadlocks

Routing, Arbitration, and Switching

Routing

- Implementing the routing: source routing vs distributed routing
  - *Source routing* (offset-based or could use absolute output port #)
    - › Routing control unit in switches is simplified; computed at source
    - › Headers containing the route tend to be larger → increase overhead



Source routing

port identifiers

offsets

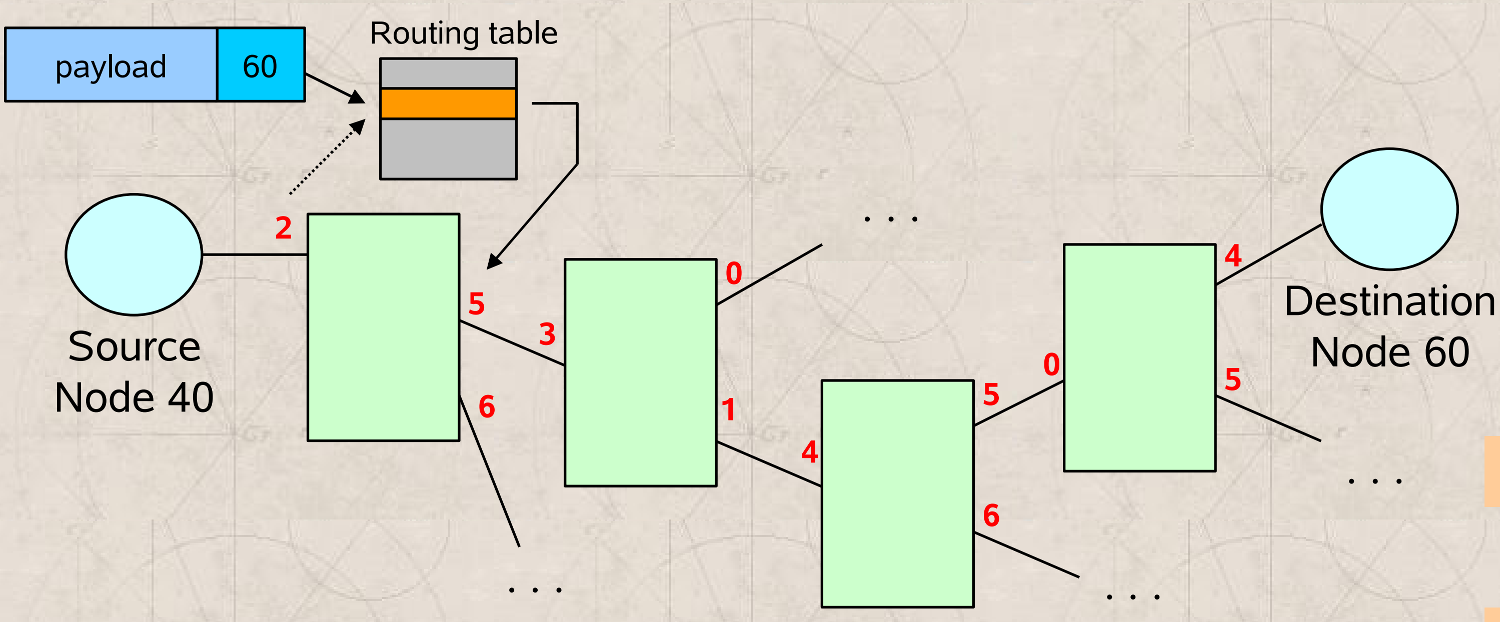
packet header

routing unit

Routing, Arbitration, and Switching

Routing

- Implementing the routing: source routing vs distributed routing
  - *Distributed routing*
    - Next route computed by *finite-state machine* or by *table look-up*
    - *Look-ahead routing* is possible: the route one hop away is supplied



Distributed routing

port identifiers

packet header

routing table

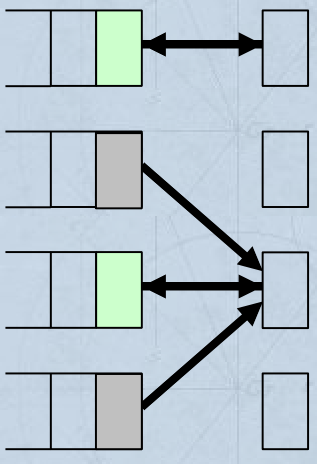
Routing, Arbitration, and Switching

Arbitration

- Performed at each switch, regardless of topology
- Determines use of paths supplied to packets (*When allocated?*)
- Needed to *resolve conflicts for shared resources* by requestors
- ***Ideally:***
  - *Maximize the matching between available network resources and packets requesting them*
  - At the switch level, arbiters maximize the matching of free switch output ports and packets located at switch input ports
- *Problems:*
  - ***Starvation***
    - Arises when packets can never gain access to requested resources
    - *Solution:* Grant resources to packets with *fairness*, even if prioritized
- Many straightforward distributed arbitration techniques for switches
  - Two-phased arbiters, three-phased arbiters, and iterative arbiters

Routing, Arbitration, and Switching

Arbitration

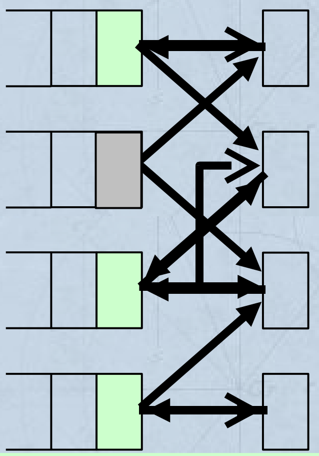


request phase

grant phase

Only two matches out of four requests (*50%* matching)

Two-phased arbiter



request phase

grant phase

accept phase

Now, three matches out of four requests (*75%* matching)

Three-phased arbiter

Optimizing the matching can increase ρ *(i.e., ρ_A)*

Routing, Arbitration, and Switching

Switching

- Performed at each switch, regardless of topology
- Establishes the connection of paths for packets (How allocated?)
- Needed to *<u>increase utilization of shared resources</u>* in the network
- ***Ideally:***
  - *Establish or “switch in” connections between network resources (1) only for as long as paths are needed and (2) exactly at the point in time they are ready and needed to be used by packets*
  - Allows for efficient use of network bandwidth to competing flows
- *Switching techniques:*
  - Circuit switching
    - pipelined circuit switching
  - Packet switching
    - Store-and-forward switching
    - Cut-through switching: virtual cut-through and wormhole

Routing, Arbitration, and Switching

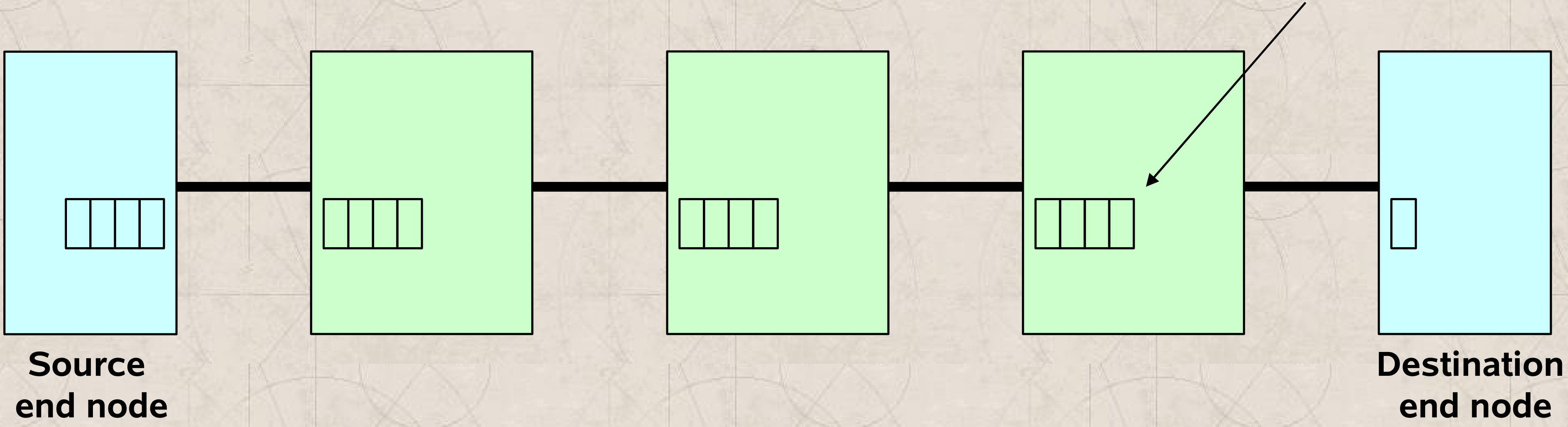
Switching

- *Circuit switching*
  - A "circuit" path is established *a priori* and torn down after use
  - Possible to pipeline the establishment of the circuit with the transmission of multiple successive packets along the circuit
    - *pipelined circuit switching*
  - Routing, arbitration, switching performed once for train of packets
    - Routing bits not needed in each packet header
    - Reduces latency and overhead
  - Can be highly wasteful of scarce network bandwidth
    - Links and switches go under utilized
      - during path establishment and tear-down
      - if no train of packets follows circuit set-up

Routing, Arbitration, and Switching

Switching

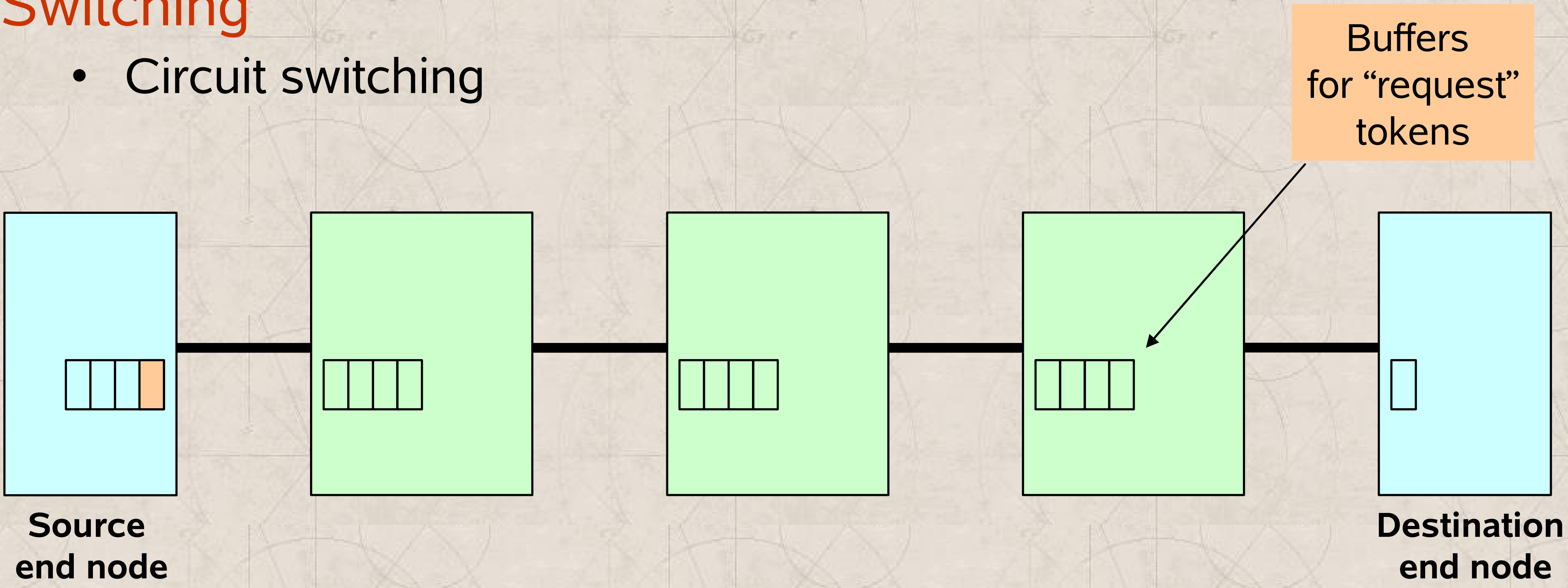
- Circuit switching



Routing, Arbitration, and Switching

Switching

- Circuit switching

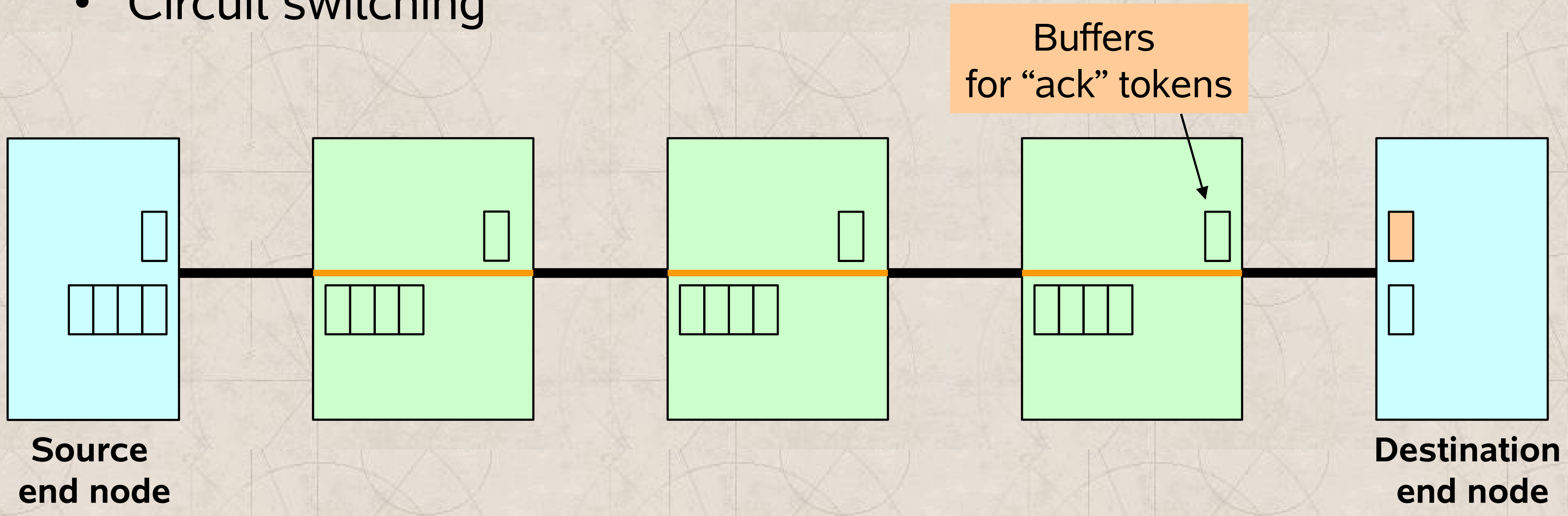


Request for circuit establishment (routing and arbitration is performed during this step)

Routing, Arbitration, and Switching

Switching

- Circuit switching



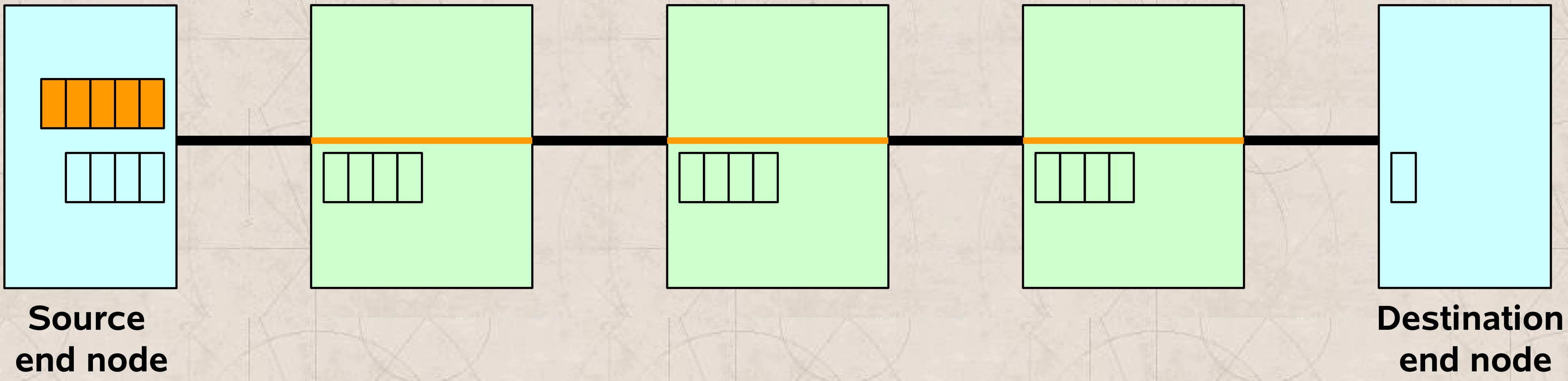
Request for circuit establishment

Acknowledgment and circuit establishment
(as token travels back to the source, connections are established)

Routing, Arbitration, and Switching

Switching

- Circuit switching



Request for circuit establishment

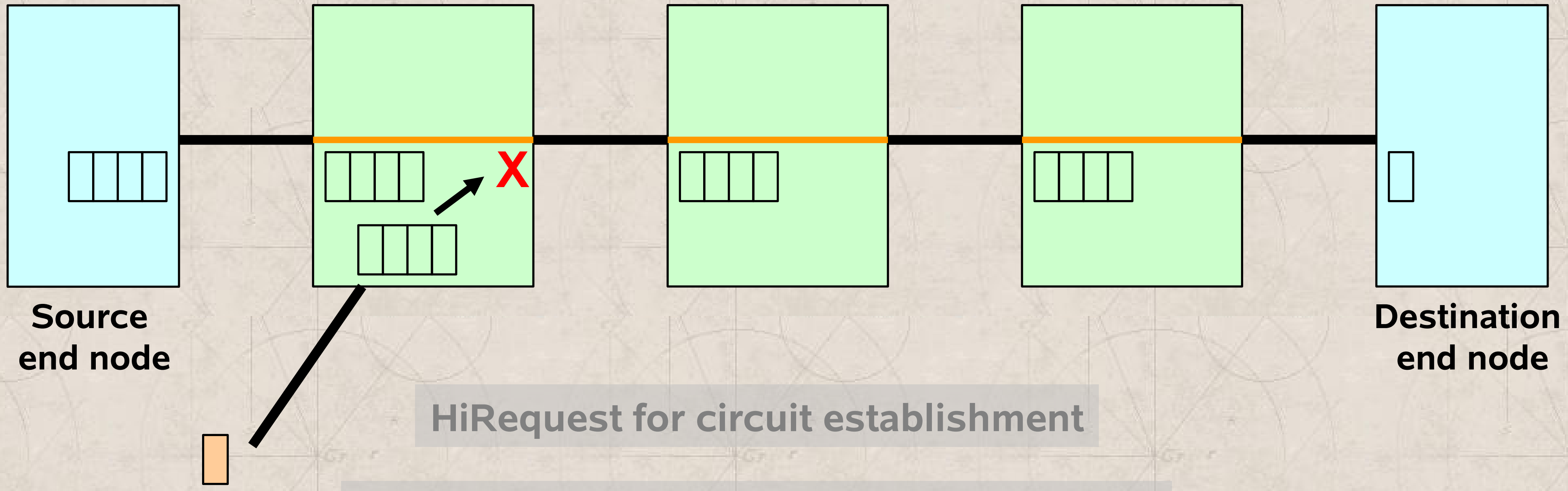
Acknowledgment and circuit establishment

Packet transport
(neither routing nor arbitration is required)

Routing, Arbitration, and Switching

Switching

- Circuit switching



HiRequest for circuit establishment

Acknowledgment and circuit establishment

Packet transport

High contention, low utilization (ρ) → *low throughput*

Routing, Arbitration, and Switching

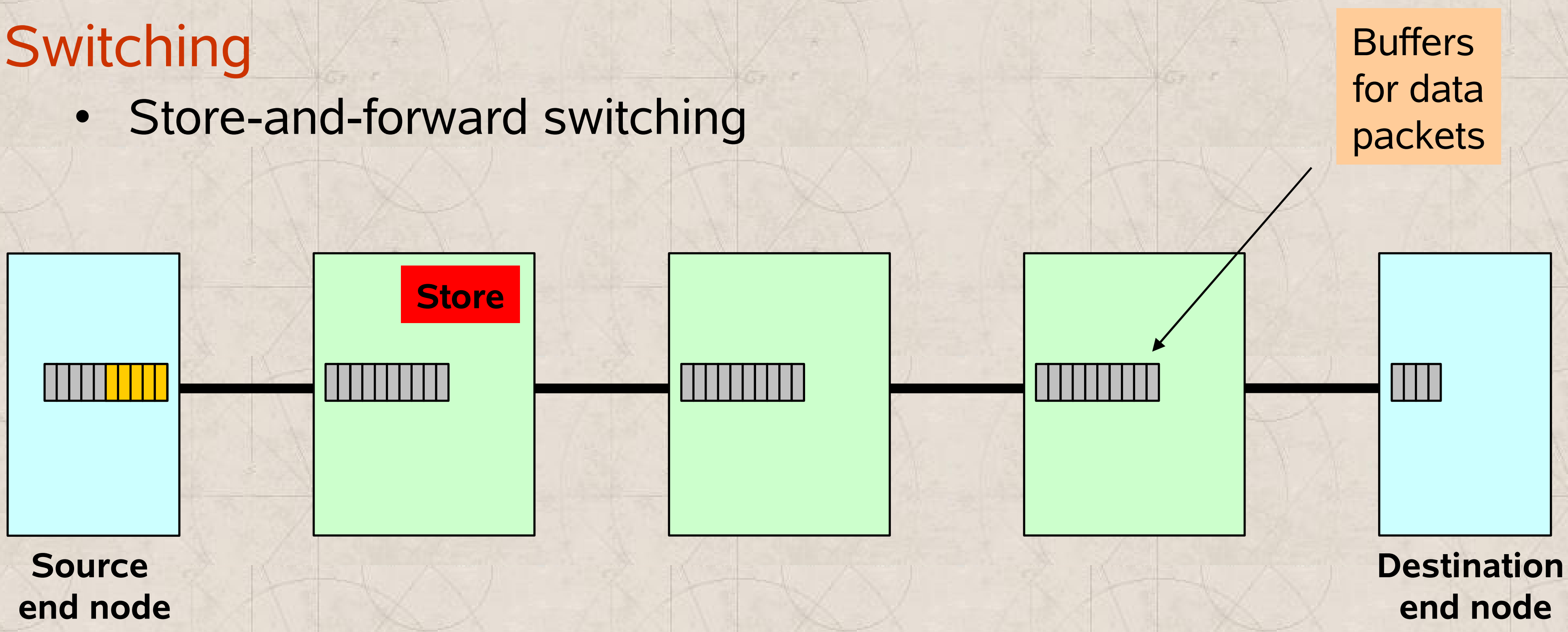
Switching

- *Packet switching*
  - Routing, arbitration, switching is performed on a per-packet basis
  - Sharing of network link bandwidth is done on a per-packet basis
  - More efficient sharing and use of network bandwidth by multiple flows if transmission of packets by individual sources is more intermittent
  - *Store-and-forward* switching
    - Bits of a packet are forwarded only after entire packet is first stored
    - Packet transmission delay is *multiplicative* with hop count, *d*
  - *Cut-through* switching
    - Bits of a packet are forwarded once the header portion is received
    - Packet transmission delay is *additive* with hop count, *d*
    - *Virtual cut-through*: flow control is applied at the packet level
    - *Wormhole*: flow control is applied at the *flow unit* (*flit*) level
    - *Buffered wormhole*: flit-level flow control with centralized buffering

Routing, Arbitration, and Switching

Switching

- Store-and-forward switching

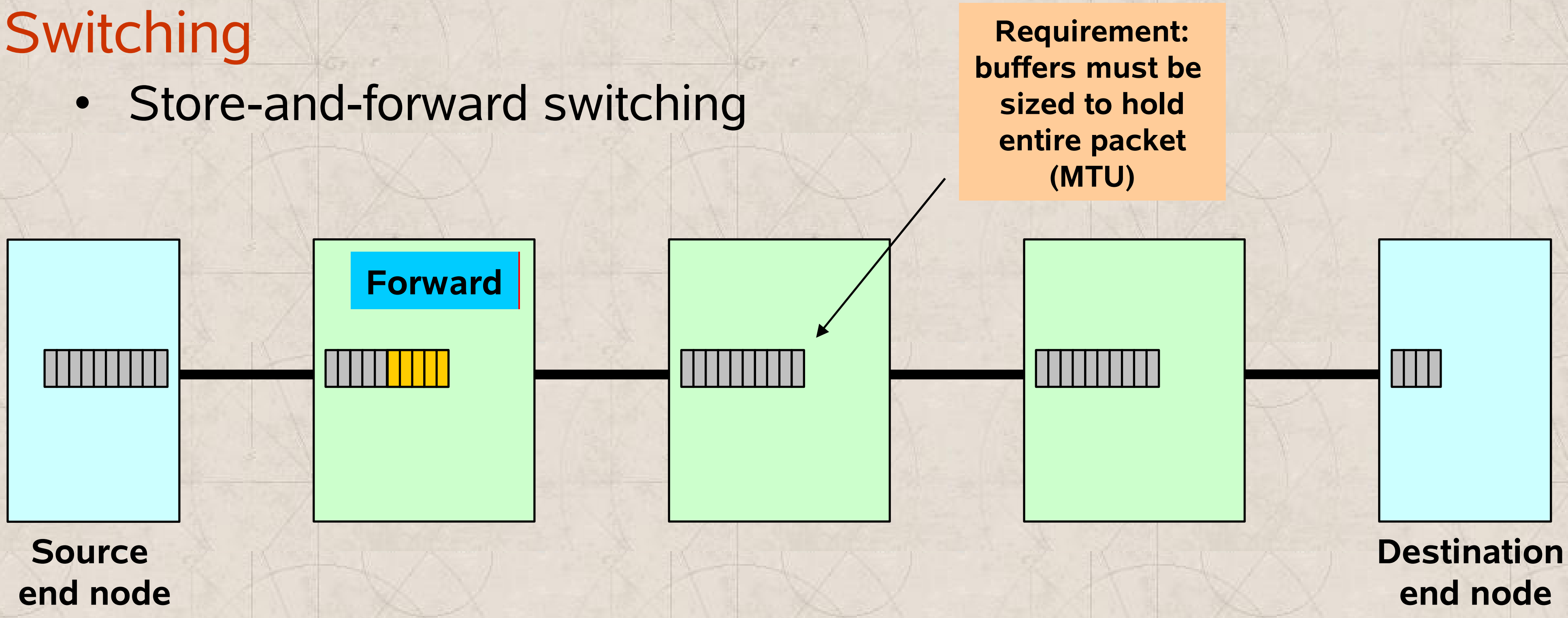


Packets are completely stored before any portion is forwarded

Routing, Arbitration, and Switching

Switching

- Store-and-forward switching

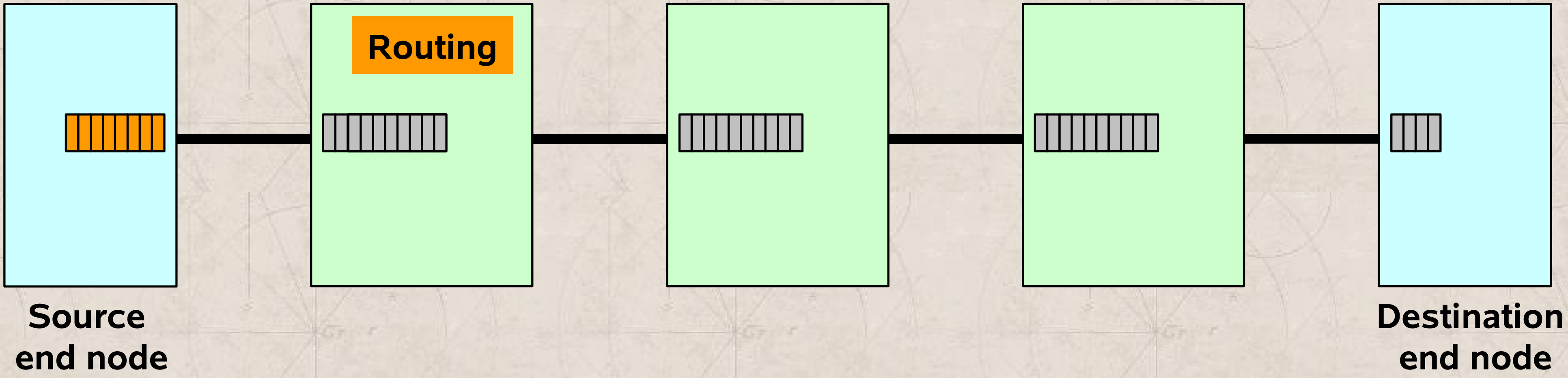

Requirement: buffers must be sized to hold entire packet (MTU)
Forward
Source end node
Destination end node

Packets are completely stored before any portion is forwarded

Routing, Arbitration, and Switching

Switching

- Cut-through switching

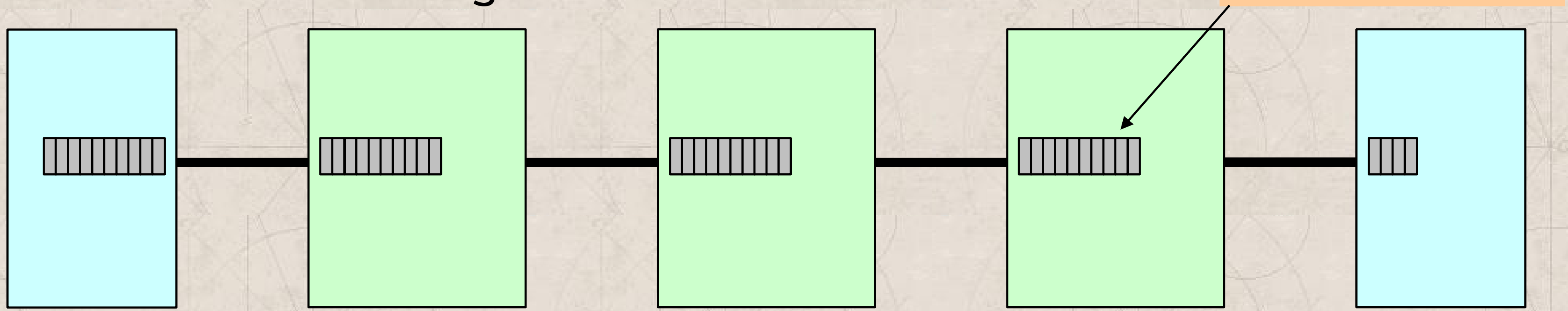


Portions of a packet may be forwarded ("cut-through") to the next switch before the entire packet is stored at the current switch

Routing, Arbitration, and Switching

Switching

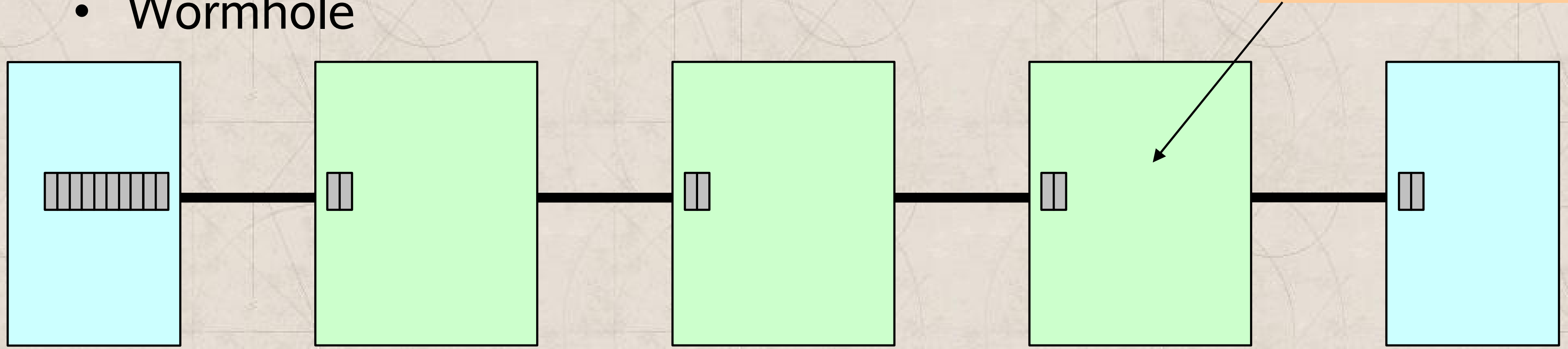
- Virtual cut-through



Buffers for data packets
Requirement: buffers must be sized to hold entire packet (MTU)

Source end node

- Wormhole



Buffers for flits: packets can be larger than buffers

Source end node

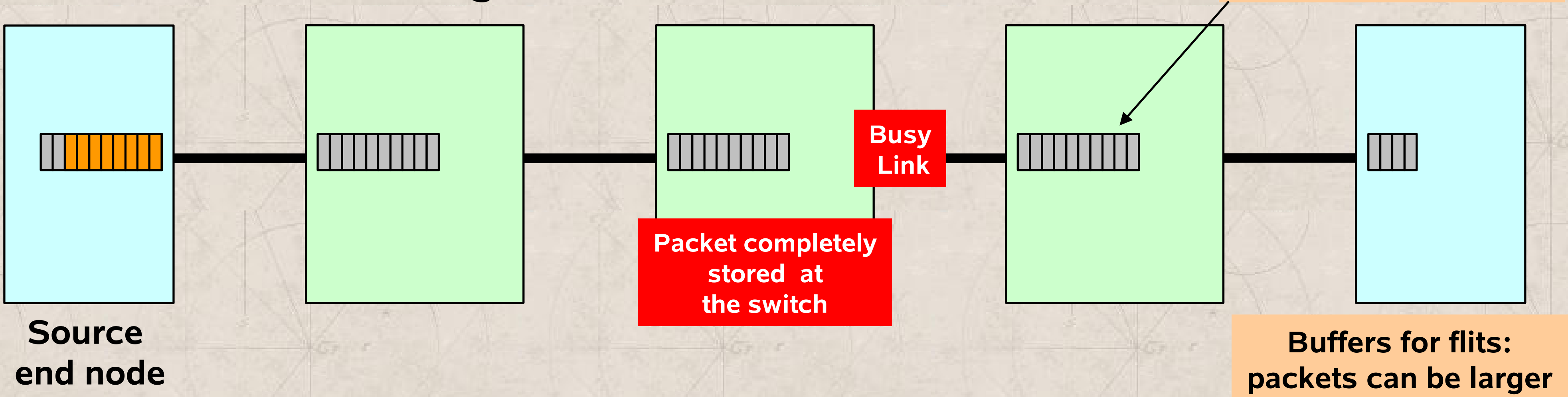
Destination end node

"Virtual Cut-Through: A New Computer Communication Switching Technique," P. Kermani and L. Kleinrock, *Computer Networks*, 3, pp. 267–286, January, 1979.

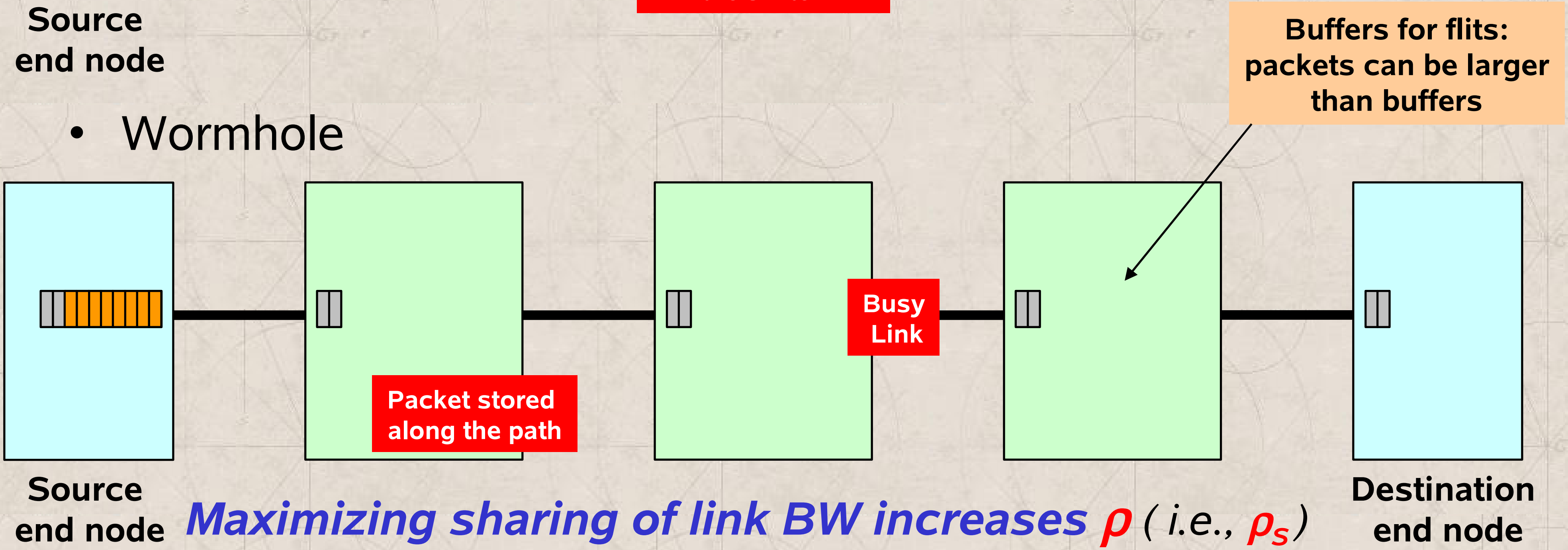
Routing, Arbitration, and Switching

Switching

- Virtual cut-through


Buffers for data packets
Requirement: buffers must be sized to hold entire packet (MTU)
Busy Link
Packet completely stored at the switch
Source end node

- Wormhole


Buffers for flits: packets can be larger than buffers
Busy Link
Packet stored along the path
Source end node
Destination end node

Maximizing sharing of link BW increases ρ *(i.e., ρ_s)*

"Virtual Cut-Through: A New Computer Communication Switching Technique," P. Kermani and L. Kleinrock, *Computer Networks*, 3, pp. 267–286, January, 1979.

Routing, Arbitration, and Switching

Characterizing Performance: Latency & Effective Bandwidth

- At low network loads, routing and arbitration have little effect on performance as there is very little contention for shared resources
- Effective bandwidth affected by *network efficiency factor*, $0 < \rho \leq 1$
  - Routing can distribute traffic more evenly across bisection links
  - Arbitration can maximize input-output matching, switch efficiency
  - Switching can increase the degree of resource (link) sharing
  - $\rho = \rho_L \times \rho_R \times \rho_A \times \rho_S \times \ldots$

$$\textit{Latency} = \textit{Sending overhead} + T_{LinkProp} \times (d+1) + (T_r + T_a + T_s) \times d + \frac{\textit{Packet} + (d \times \textit{Header})}{\textit{Bandwidth}} + \textit{Receiving overhead}$$

(*cut-through switching*)

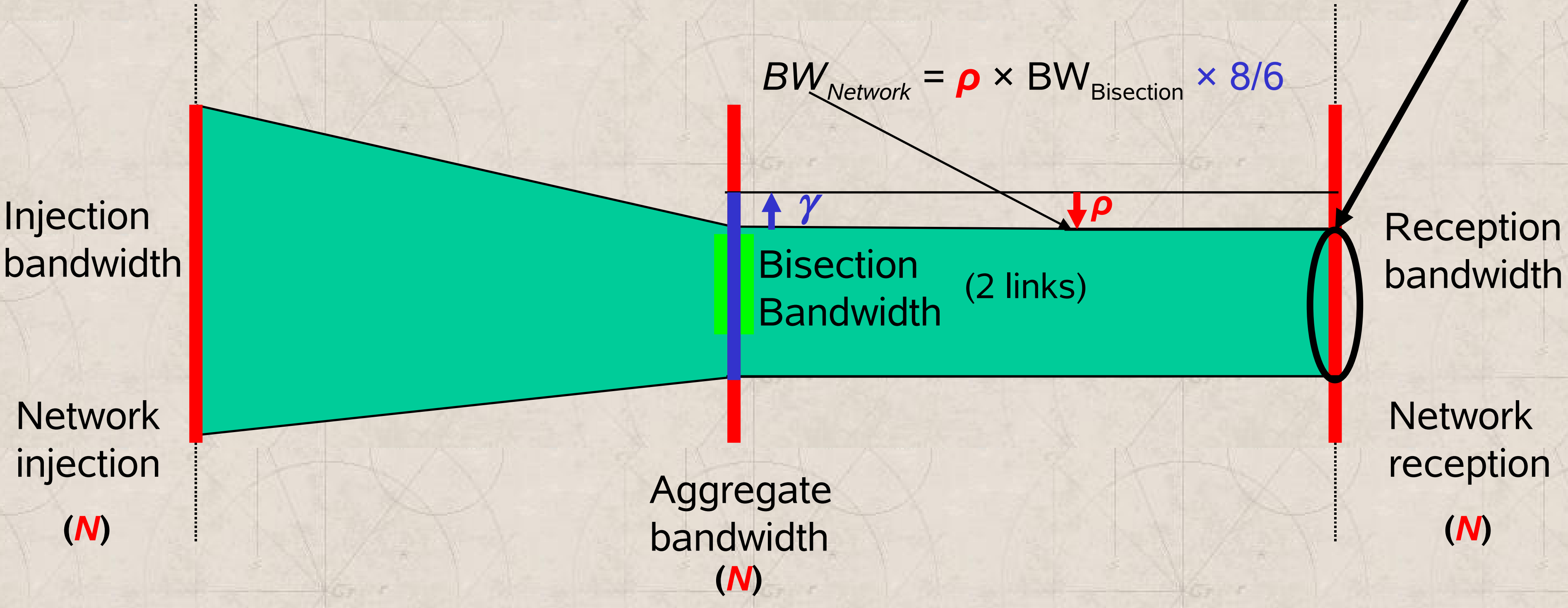
lower bound (contention delay not included)

upper bound (contention effects not ***fully*** included)

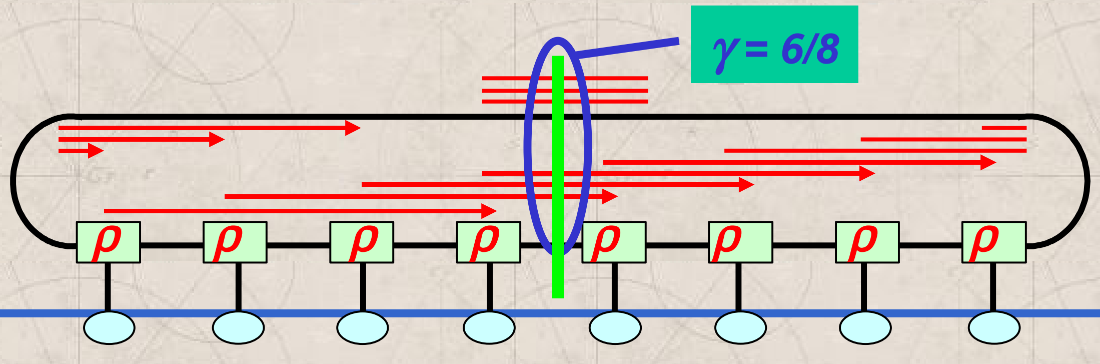
$$\textit{Effective bandwidth} = \min\left(N \times BW_{LinkInjection}, \underbrace{\frac{\rho \times BW_{Bisection}}{\gamma}}_{BW_{Network}}, \sigma \times N \times BW_{LinkReception}\right)$$

Routing, Arbitration, and Switching

Characterizing Performance: Latency & Effective Bandwidth



$BW_{Network} = \rho \times BW_{Bisection} \times 8/6$



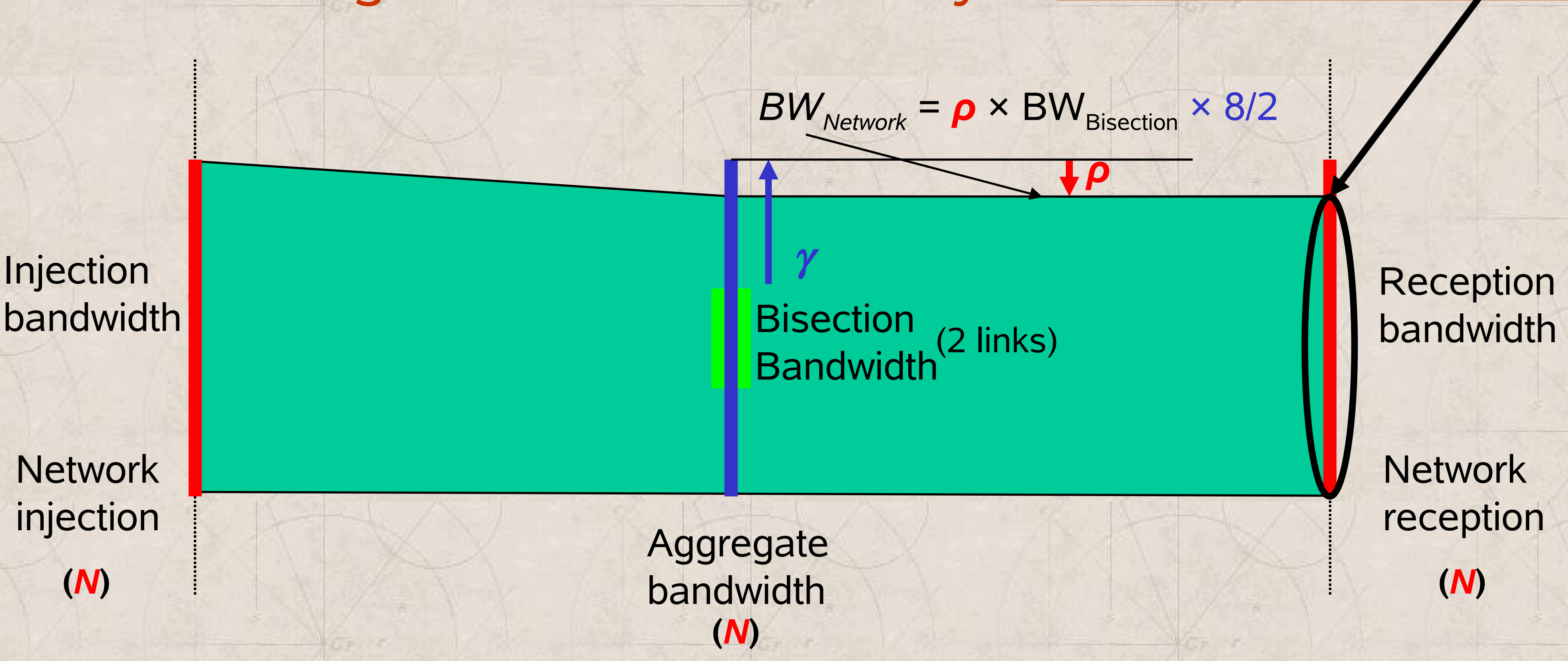
tornado traffic: node *i* sends to node *i + (N/2-1)* mod *N*

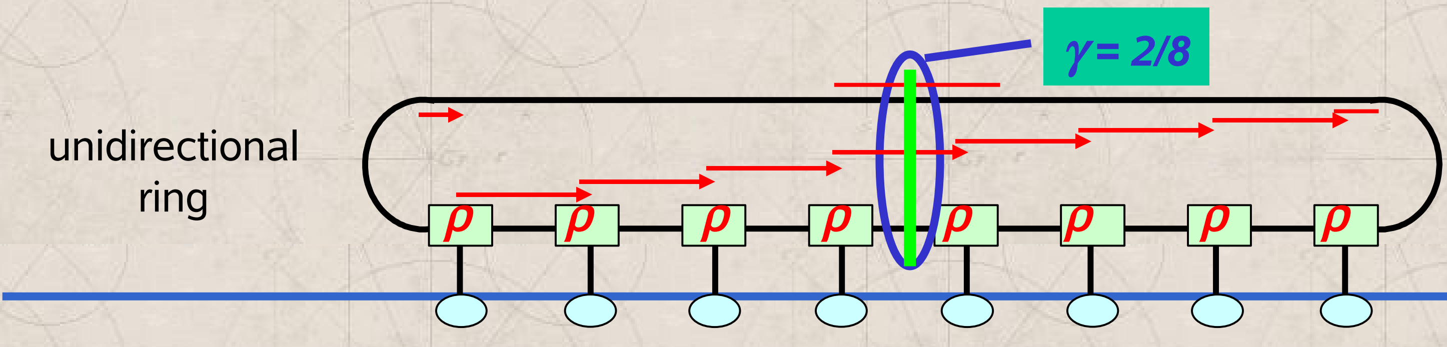
$\sigma = 1$

Routing, Arbitration, and Switching

Characterizing Performance: Latency & Effective Bandwidth

$BW_{Network} = \rho \times BW_{Bisection} \times 8/2$



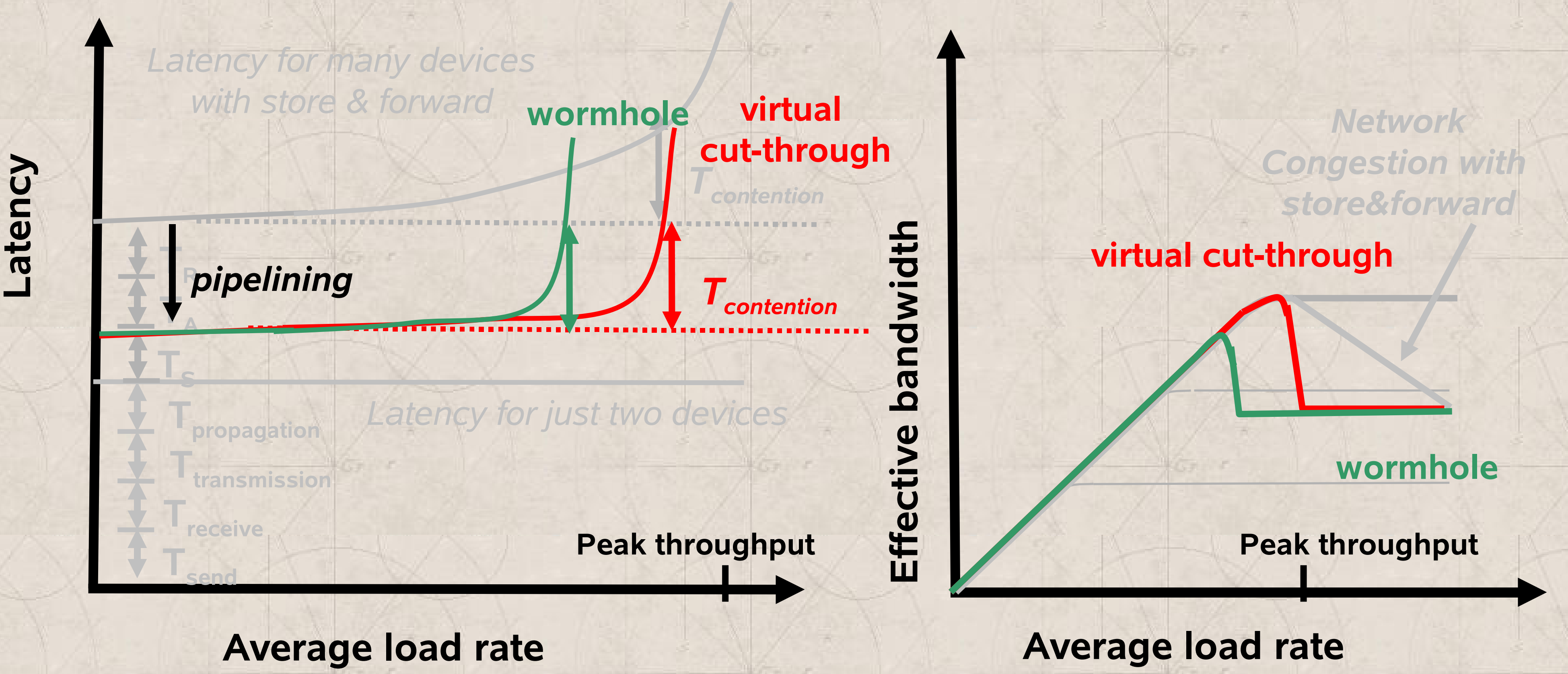


***near-neighbour traffic*:**
node *i* sends to node
(i + 1) mod *N*
σ = *1*

Routing, Arbitration, and Switching

Characterizing Performance: Latency & Effective Bandwidth

- Characteristic performance plots: latency vs. average load rate; throughput (effective bandwidth) vs. average load rate

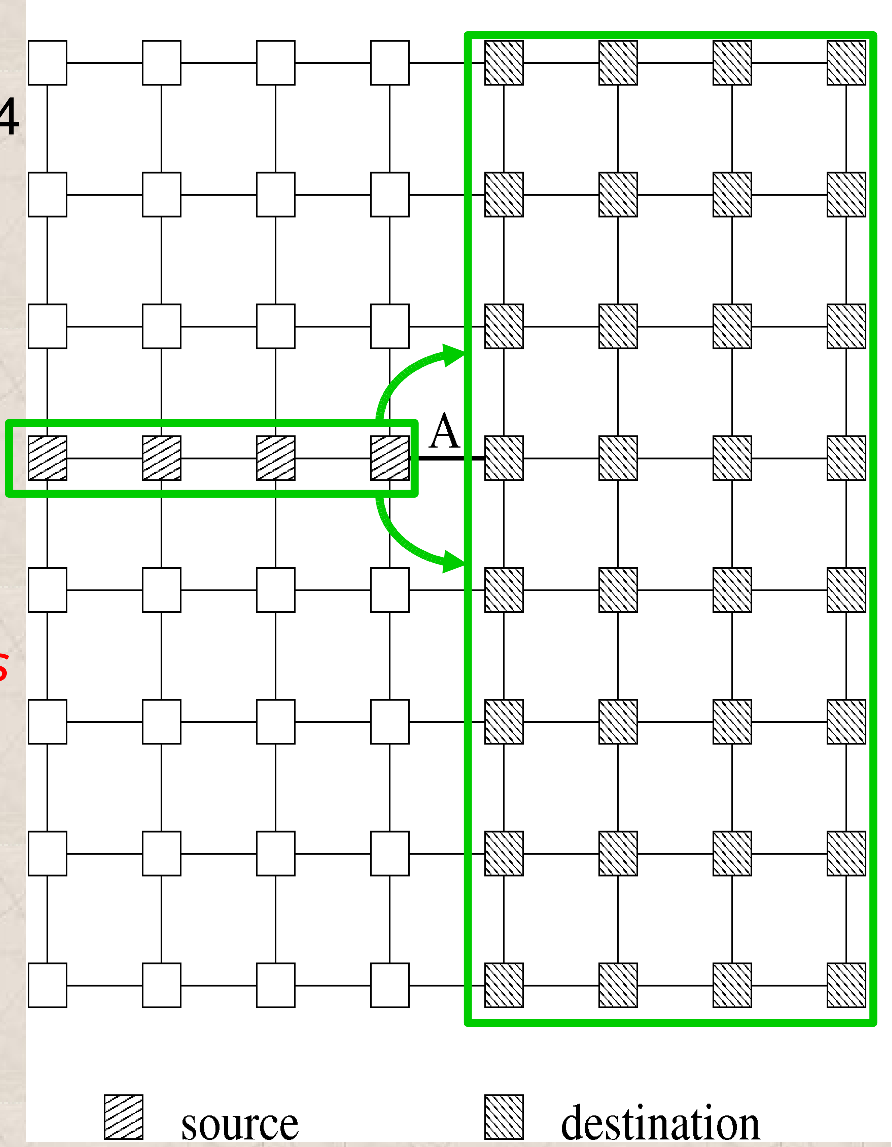


Routing, Arbitration, and Switching: ρ

Characterizing Performance: Latency & Effective Bandwidth

- Dimension-order routing on 2D mesh, *N*=64
- Uniformly distributed traffic
- *$BW_{Bisection}$* = 2 x 8 = 16 link BW units
- ∀ γ = 2 x (32 x 32) / (64 x 64) = 0.5
- *$BW_{Network}$* = 32 BW units, if ρ = 100%
  - – Fraction of overall traffic each link would carry is $\gamma/Links_{Bisection}$= 0.03125
- DOR restrictions *evenly load bisection links*
  - – Link A carries (4 x 32)(64 x 64) = 0.03125 fraction of overall traffic
  - – Keeps *$BW_{Network}$* at max of 32 BW units

    ρ = 32/ 32 = 100% (best case)



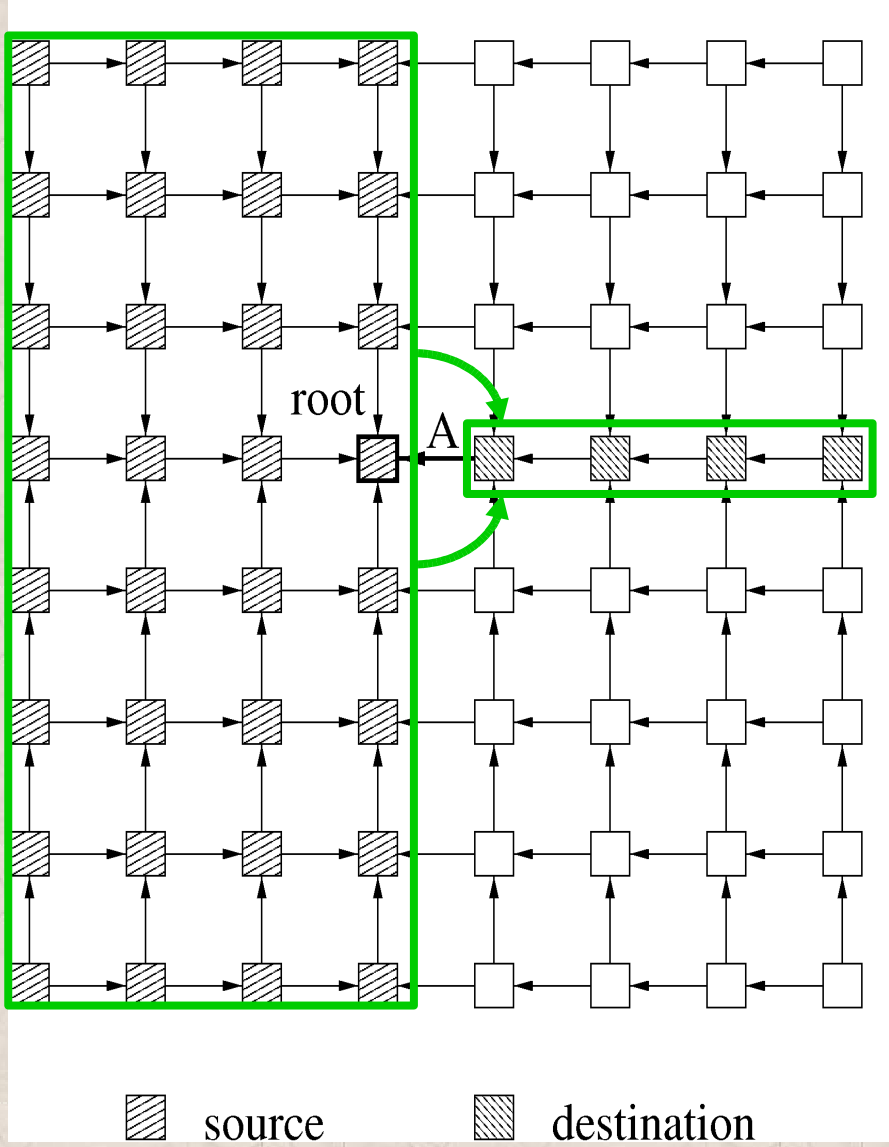
Routing, Arbitration, and Switching: ρ

Characterizing Performance: Latency & Effective Bandwidth

- Up*/down* routing on 2D mesh, $N = 64$
- Uniformly distributed traffic
- $BW_{Bisection}$ = 2 x 8 = 16 link BW units

∀ γ = 2 x (32 x 32) / (64 x 64) = 0.5

- $BW_{Network}$ = 32 BW units, if ρ = 100%
  - fraction of overall traffic each link would carry is $\gamma/Links_{Bisection}$= 0.03125
- U*/D* routing restrictions *overload Link A*
  - ***Case 1***: carries (32 x 4)/(64 x 64) = 0.03125 fraction of overall traffic

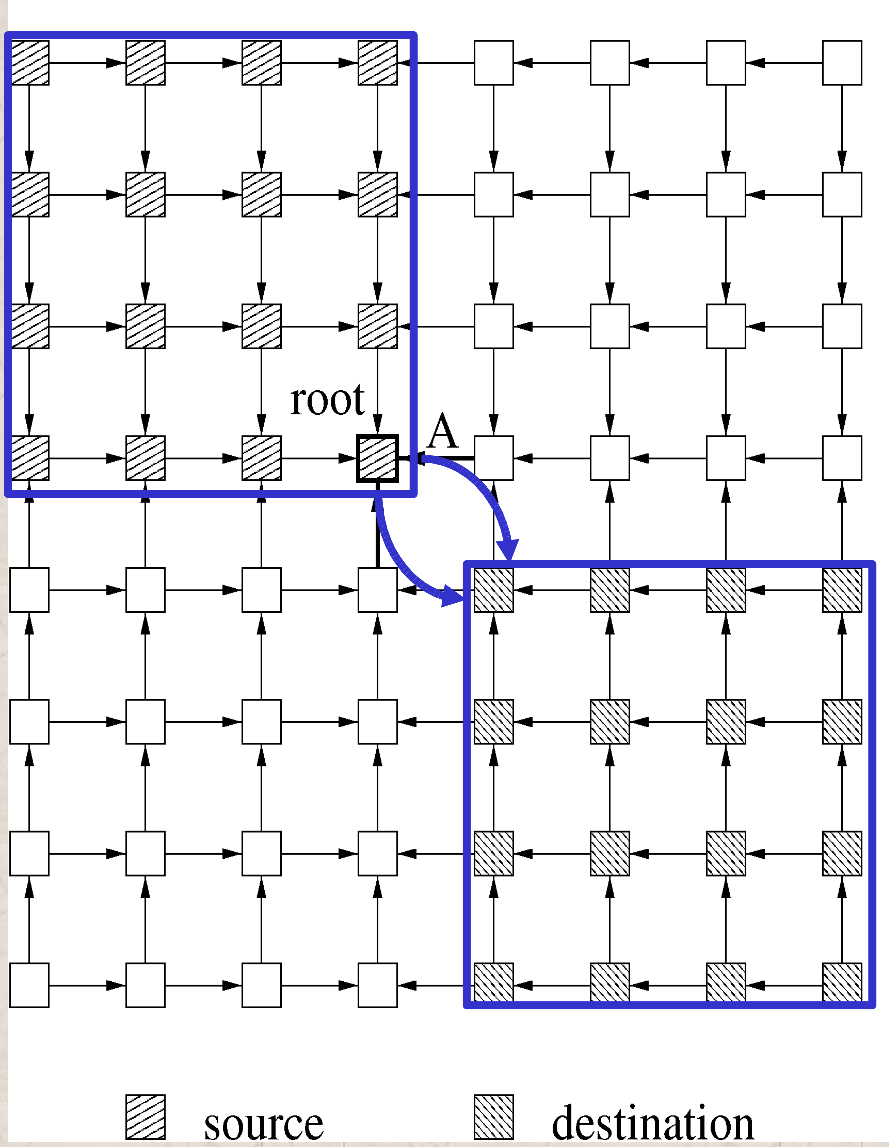


Routing, Arbitration, and Switching: *ρ*

Characterizing Performance: Latency & Effective Bandwidth

- Up*/down* routing on 2D mesh, $N = 64$
- Uniformly distributed traffic
- $BW_{Bisection}$ = 2 x 8 = 16 link BW units
- ∀ γ = 2 x (32 x 32) / (64 x 64) = 0.5
- $BW_{Network}$ = 32 BW units, if ρ = 100%
  - fraction of overall traffic each link would carry is $\gamma / Links_{Bisection}$ = 0.03125

- U*/D* routing restrictions *overload Link A*
  - ***Case 1***: carries (32 x 4)/(64 x 64) = 0.03125 fraction of overall traffic
  - ***Case 2***: carries ½(16 x 16)/(64 x 64) = 0.03125 fraction of traffic



Routing, Arbitration, and Switching: ρ

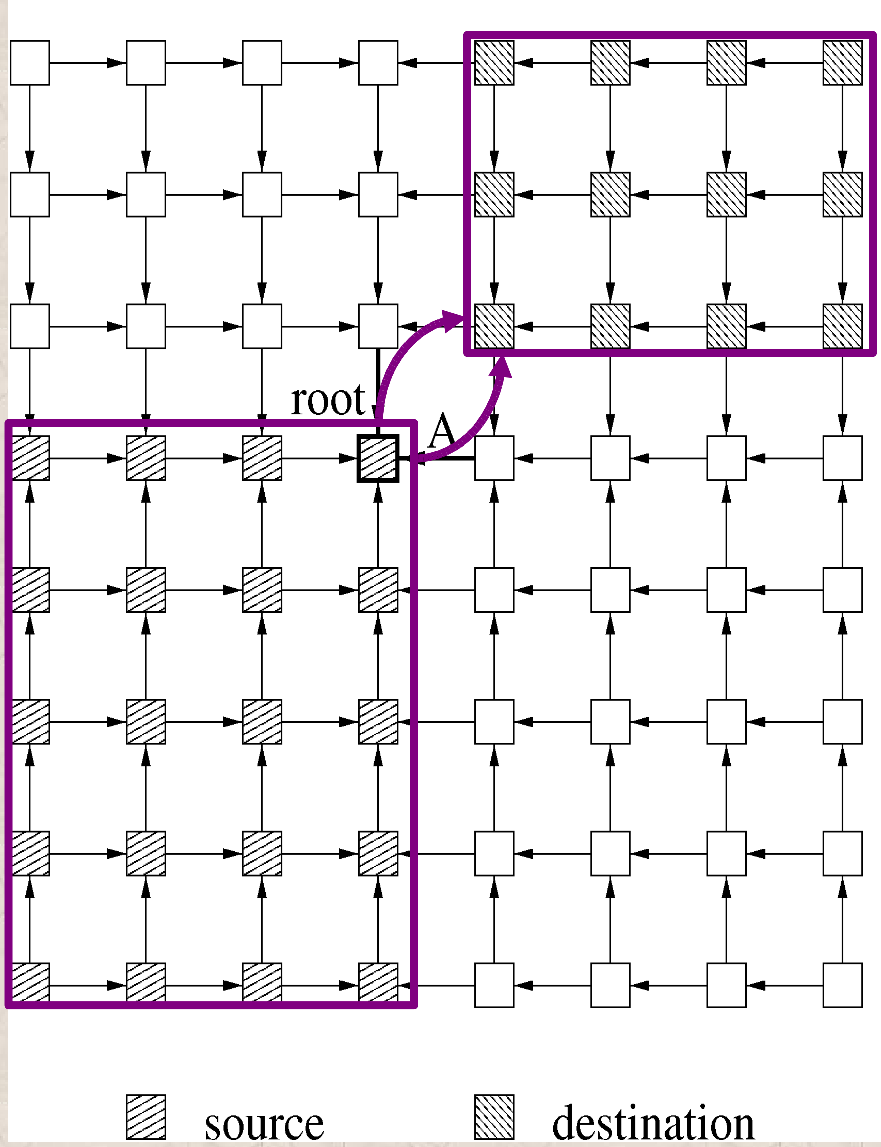
Characterizing Performance: Latency & Effective Bandwidth

- Up*/down* routing on 2D mesh, *N* = 64
- Uniformly distributed traffic
- $BW_{Bisection}$ = 2 x 8 = 16 link BW units

∀ γ = 2 x (32 x 32) / (64 x 64) = 0.5

- $BW_{Network}$ = 32 BW units, if ρ = 100%
  - fraction of overall traffic each link would carry is $\gamma / Links_{Bisection}$= 0.03125

- U*/D* routing restrictions *overload Link A*
  - ***Case 1***: carries (32 x 4)/(64 x 64) = 0.03125 fraction of overall traffic
  - ***Case 2***: carries ½(16 x 16)/(64 x 64) = 0.03125 fraction of traffic
  - ***Case 3***: carries ½(20 x 12)/(64 x 64) = 0.02930 fraction of traffic

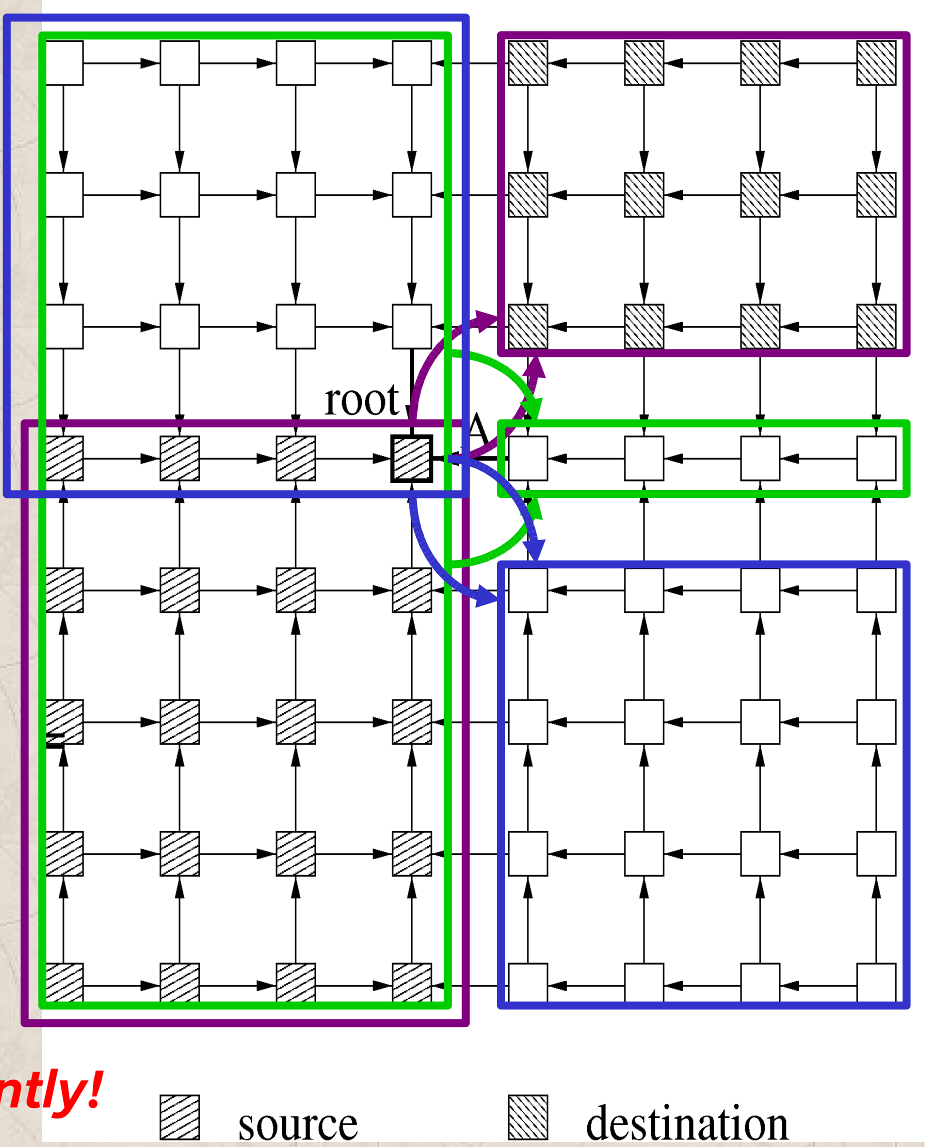


Routing, Arbitration, and Switching: ρ

Characterizing Performance: Latency & Effective Bandwidth

- Up*/down* routing on 2D mesh, $N = 64$
- Uniformly distributed traffic
- $BW_{Bisection}$ = 2 x 8 = 16 link BW units
- ∀ γ = 2 x (32 x 32) / (64 x 64) = 0.5
- $BW_{Network}$ = 32 BW units, if ρ = 100%
  - fraction of overall traffic each link would carry is $\gamma/Links_{Bisection}$ = 0.03125
- U*/D* routing restrictions *overload Link A*
  - carries ***0.03125*** + ***0.03125*** + ***0.02930*** = *0.09180* fraction of overall traffic!!
  - Limits $BW_{Network}$ to only *10.9 BW units*

  ρ = 10.9 / 32 = 34% (at most)



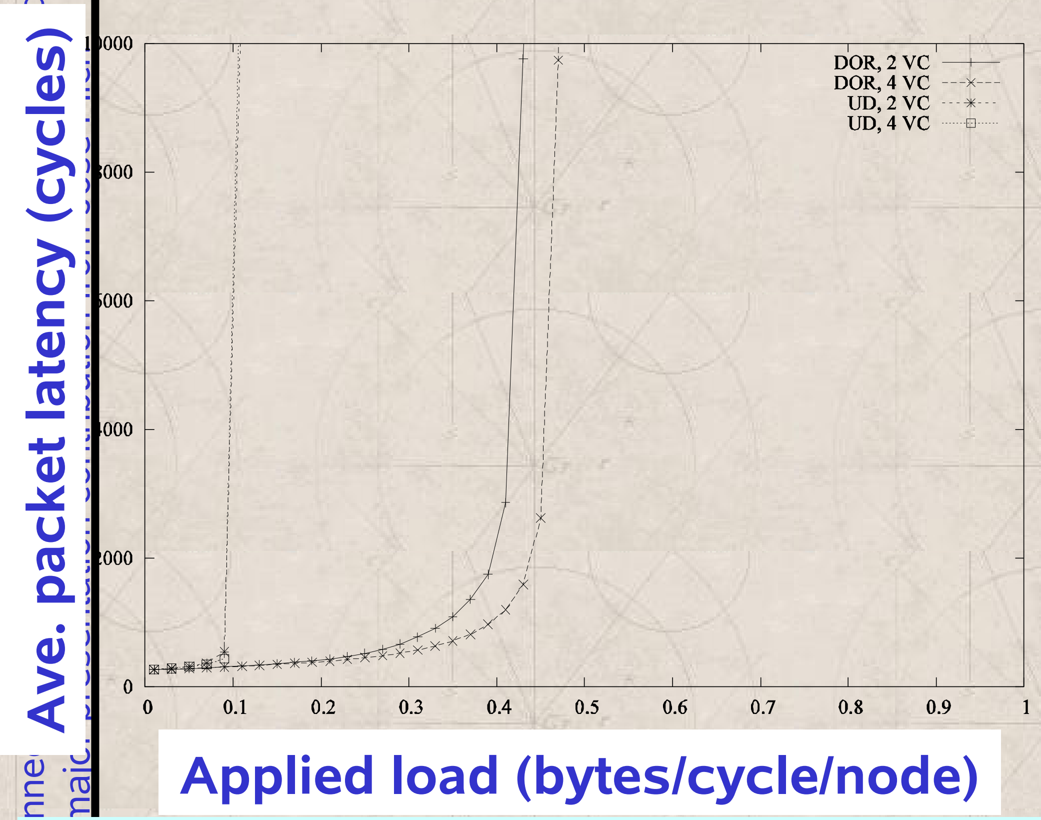
Routing algorithm can impact ρ significantly!

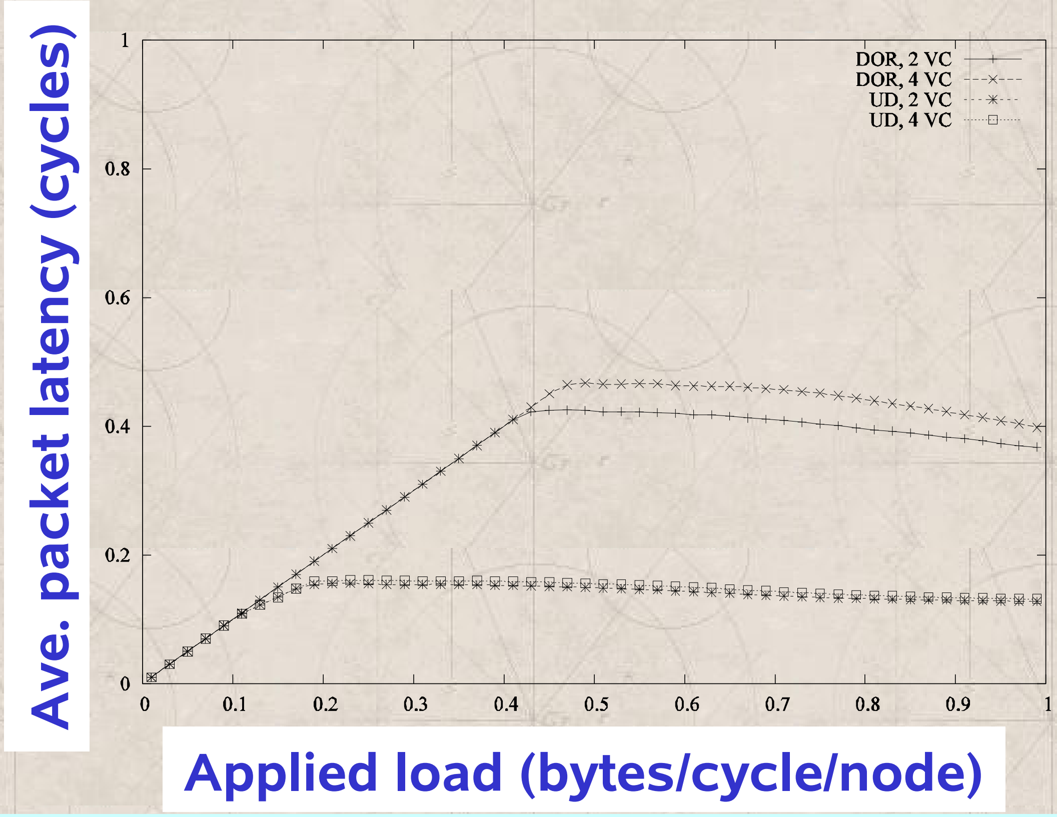
Routing, Arbitration, and Switching: *ρ*

Characterizing Performance: Latency & Effective Bandwidth

- *Measured ρ*: DOR vs U*/D* on 2D Mesh, N = 64 (via simulation)

DOR: 0.40 and 0.47 bytes/cycle/node (2 & 4 VCs) → *ρ* = 80% & 94%
U*/D*: 0.15 and 0.16 bytes/cycle/node (2 & 4VCs) → *ρ* = 30% & 32%



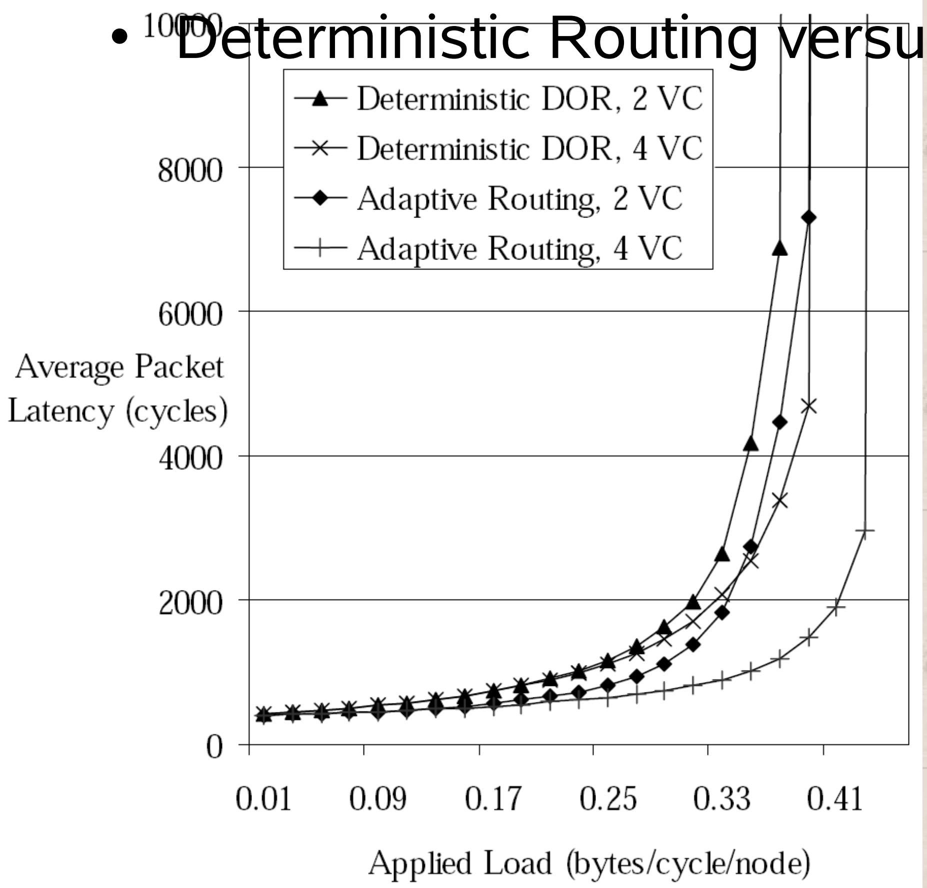


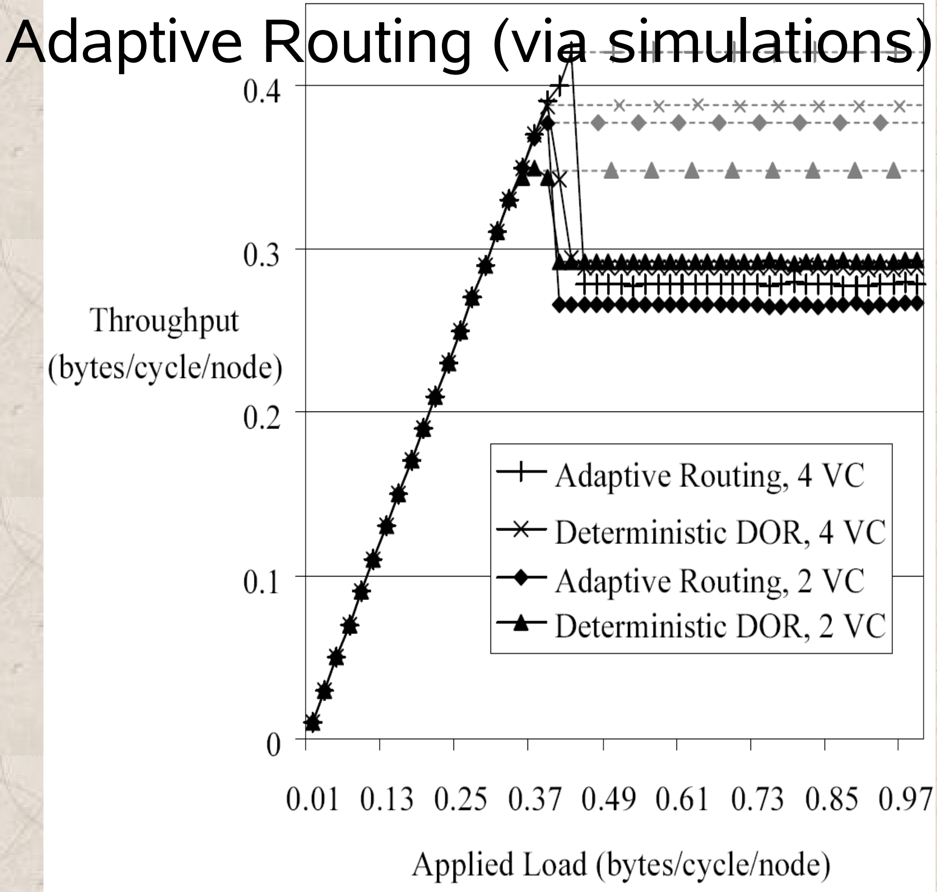
2D Mesh, *N* = 64 (8 x 8), virtual cut-through, 2 & 4 virtual channels (VCs), uniform traffic assumed.
DOR vs. Up*/Down* routing used on all VCs. ***Ideal throughput is 0.5 bytes/cycle/node.***

Routing, Arbitration, and Switching

Characterizing Performance: Latency & Effective Bandwidth

- Deterministic Routing versus Adaptive Routing (via simulations)





3-D Torus, 4,096 nodes (16 x 16 x 16), virtual cut-through switching, three-phase arbitration, 2 and 4 virtual channels. Bubble flow control in dimension order is used in one virtual channel; the other virtual channel is supplied in dimension order (***deterministic routing***) or along any shortest path to destination (***adaptive routing***). Uniform traffic is assumed.

Routing, Arbitration, and Switching

Characterizing Performance: Latency & Effective Bandwidth

- Efficiency factor (ρ) for deterministic routing vs. adaptive routing
  - Ideal (maximum) throughput: 0.5 bytes/cycle/node
    - ρ = 100%
    - Enough injection and reception bandwidth (i.e., network bandwidth poses as the "pipe" bottleneck)
    - Bisection bandwidth (16x16x4 unidirectional links)
      - 1024 bytes/cycle bandwidth at the bisection
      - 0.25 bytes/cycle/node bandwidth at the bisection
      - γ= 0.5
    - Ideal throughput: 100% x ($BW_{Bisection}$ / γ) = 0.5 bytes/cycle/node
  - Network efficiency, ρ, = measured throughput / ideal throughput
    - Adaptive routing with four VCs: **ρ = 86%**
    - Deterministic routing with two VCs: ***ρ = 74%***

Routing, Arbitration, and Switching

R, A, & S Characteristics of Commercial Machines

Company	System [Network] Name	Max. compute nodes [x #CPUs]	Basic network topology	Network routing algorithm	Switch arbitration scheme	Network switching technique
Intel	ASCI Red Paragon	4,510 [x 2]	2-D mesh 64 x 64	distributed dimension-order routing	2-phased RR, distributed across switch	wormhole w/ no virtual channels
IBM	ASCI White SP Power3 [Colony]	512 [x 16]	BMIN w/8-port bidirect. switches (fat-tree or Omega)	source-based LCA adaptive, shortest-path routing	2-phased RR, centralized & distributed at outputs for bypass paths	buffered WH & VCT for multicasting, no VCs
Intel	Thunter Itanium2 Tiger4 [QsNet^{II}]	1,024 [x 4]	fat tree w/8-port bidirectional switches	source-based LCA adaptive, shortest path routing	2-phased RR, priority, aging, distributed at output ports	WH with 2 VCs
Cray	XT3 [SeaStar]	30,508 [x 1]	3-D torus 40 x 32 x 24	distributed table-based dimension-order	2-phased RR, distributed at output ports	VCT with 4 VCs
Cray	X1E	1,024 [x 1]	4-way bristled 2-D torus (~ 23 x 11) with express links	distributed table-based dimension-order	2-phased RR, distributed at output ports	VCT with 4 Vcs
IBM	ASC Purple pSeries 575 [Federation]	>1,280 [x 8]	BMIN w/8-port bidirect. switches (fat-tree or Omega)	source and distrib. table-based LCA adapt. shortest path	2-phased RR, centralized & distributed at outputs for bypass paths	buffered WH & VCT for multicasting, 8 VCs
IBM	Blue Gene/L eServer Sol. [Torus Net]	65,536 [x 2]	3-D torus 32 x 32 x 64	distributed adaptive with bubble escape Duato's Protocol	2-phased SLQ, distributed at input & output	VCT with 4 VCs

Outline

- E.1 Introduction *(Lecture 1)*
- E.2 Interconnecting Two Devices *(Lecture 1)*
- E.3 Interconnecting Many Devices *(Lecture 2)*
- E.4 Network Topology *(Lecture 2)*
- E.5 Network Routing, Arbitration, and Switching *(Lecture 3)*
- **E.6 Switch Microarchitecture** *(Lecture 4)*
  - **Basic Switch Microarchitecture**
  - **Buffer Organizations**
  - **Routing and Arbitration Unit**
  - **Pipelining the Switch Microarchitecture**
  - **Characterizing Performance: Latency & Effective Bandwidth**
- E.7 Practical Issues for Commercial Interconnection Networks *(Lecture 4)*
- E.8 Examples of Interconnection Networks *(Lecture 5)*
- E.9 Internetworking (skipped)
- E.10 Crosscutting Issues for Interconnection Networks (skipped)
- E.11 Fallacies and Pitfalls *(Lecture 5)*
- E.12 Concluding Remarks and References *(Lecture 5)*

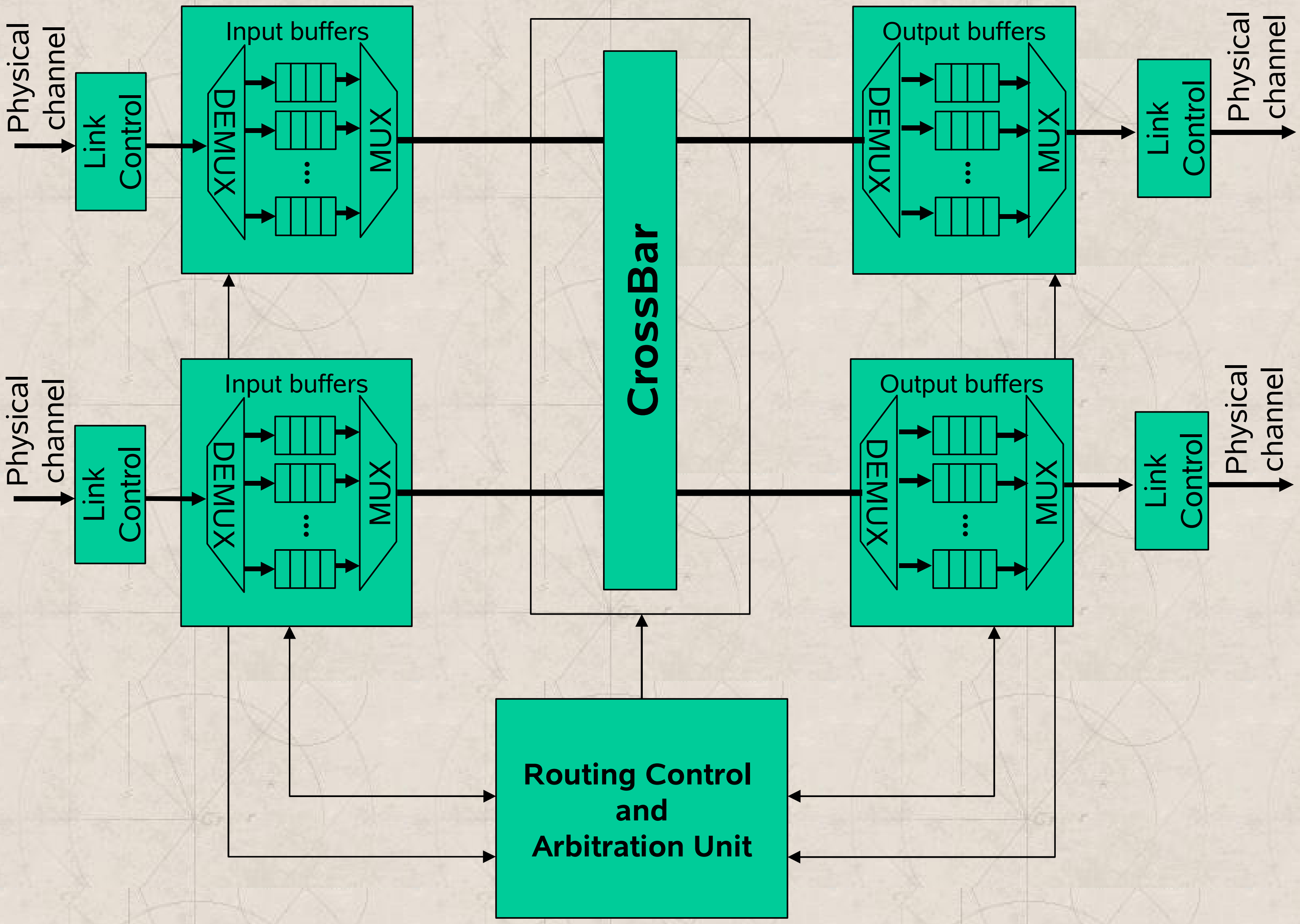
Switch Microarchitecture

Basic Switch Microarchitecture

- Internal data path
  - Implements flow control, routing, arbitration, and switching
  - Provides connectivity between switch input and output ports
  - A crossbar is commonly used to provide internal connectivity
    - Non-blocking, concurrent connectivity
  - Other components along the internal datapath consist of
    - link (flow) control units, I/O buffers, routing and arbitration unit
- *Speedup:* ratio of provided bandwidth to required bandwidth
  - Implemented within the internal data path of a switch by
    - Increased clock frequency (time) or internal datapath width (space)
    - Multiple datapaths via increased # of crossbar access points
    - Alternatively, multiple datapaths via a buffered crossbar switch
      - Arbitration made simpler (independent, distributed arbiters)
      - Expensive architecture

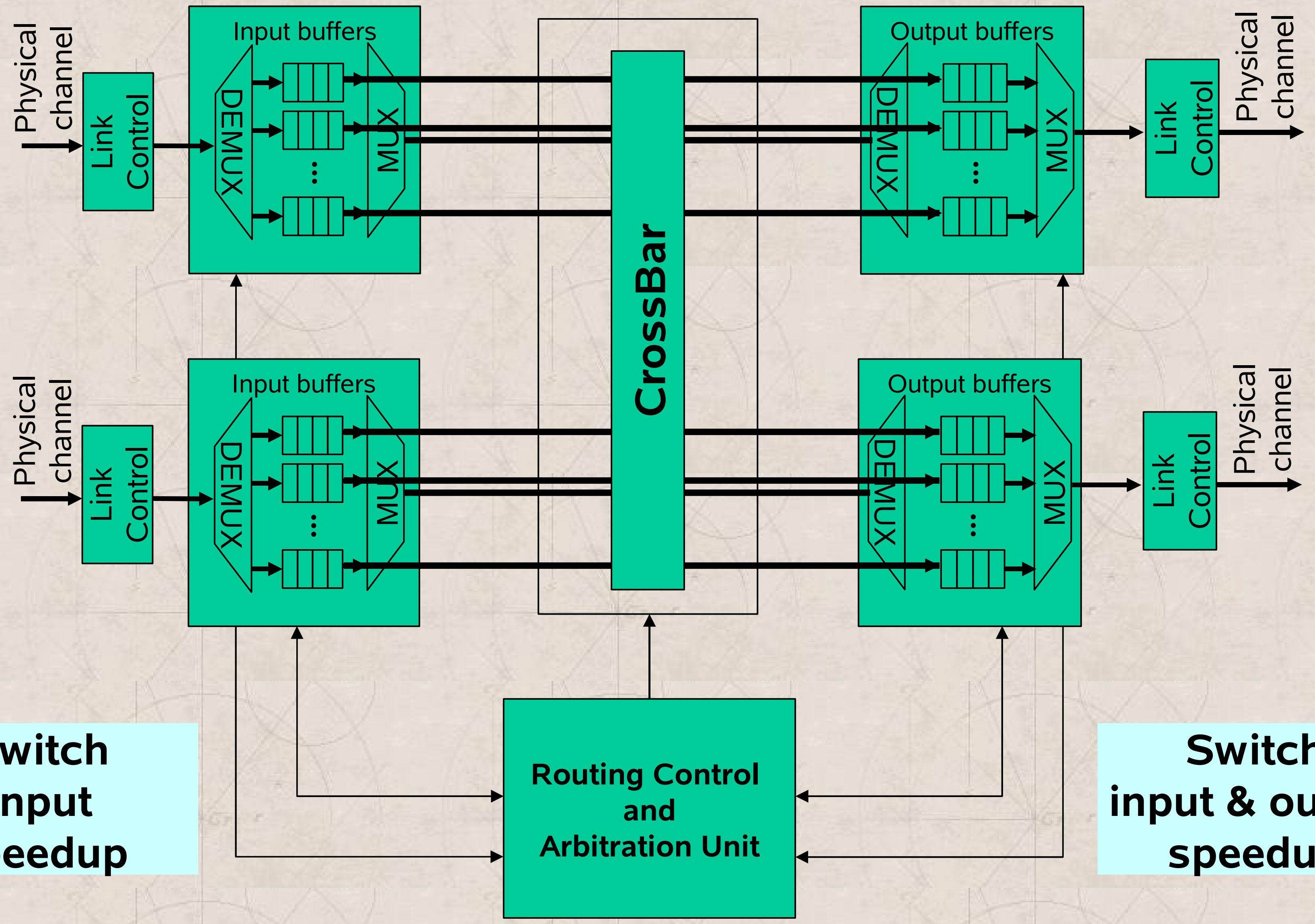
Switch Microarchitecture

Basic Switch Microarchitecture



Switch Microarchitecture

Basic Switch Microarchitecture



Switch input speedup

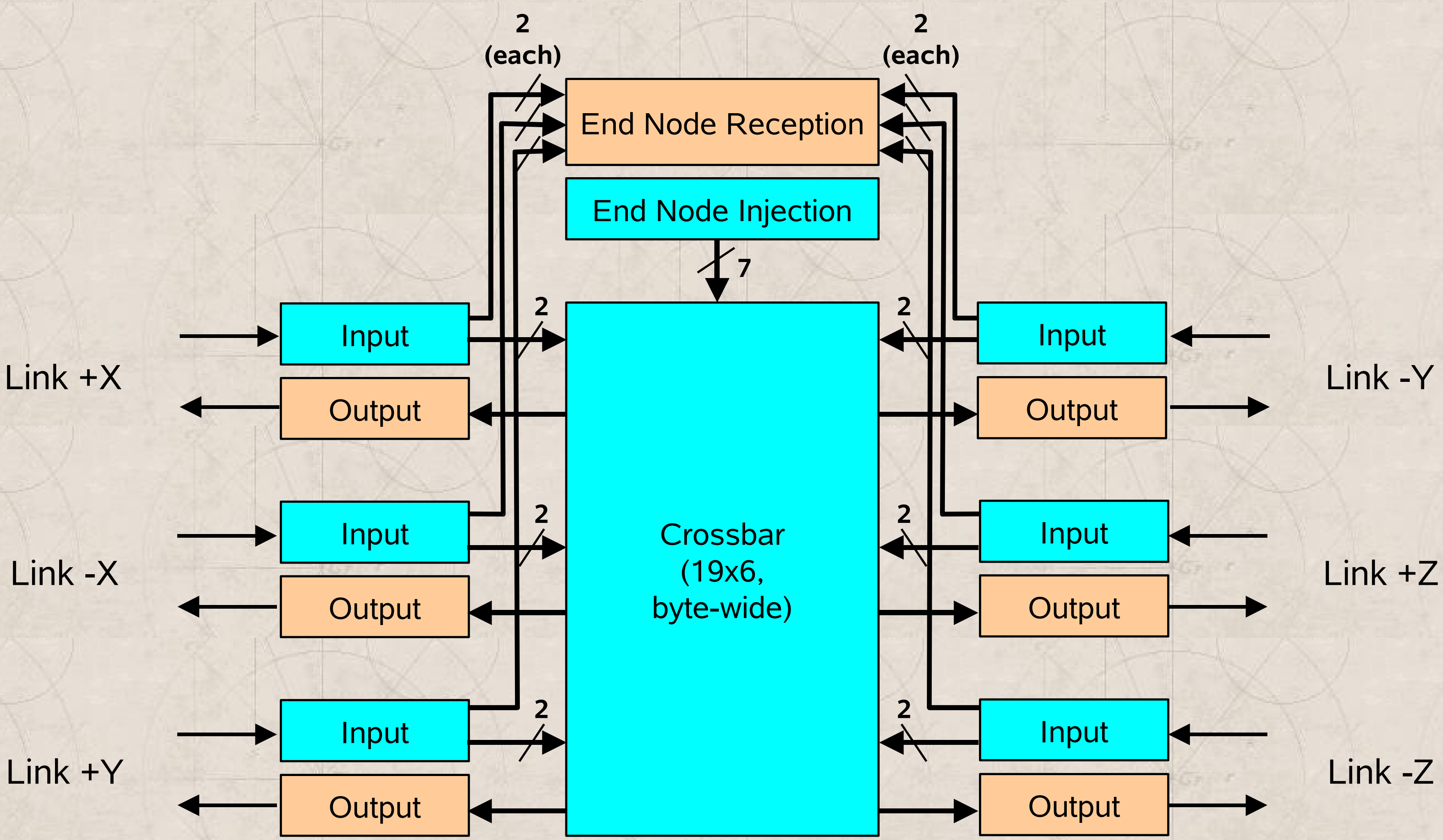
Switch input & output speedup

Maximizing use of internal switch datapath can increase ρ (i.e., $\rho_{\mu Arch}$)

Switch Microarchitecture

Basic Switch Microarchitecture

- *Input speedup* implemented in the Blue Gene/L network switch



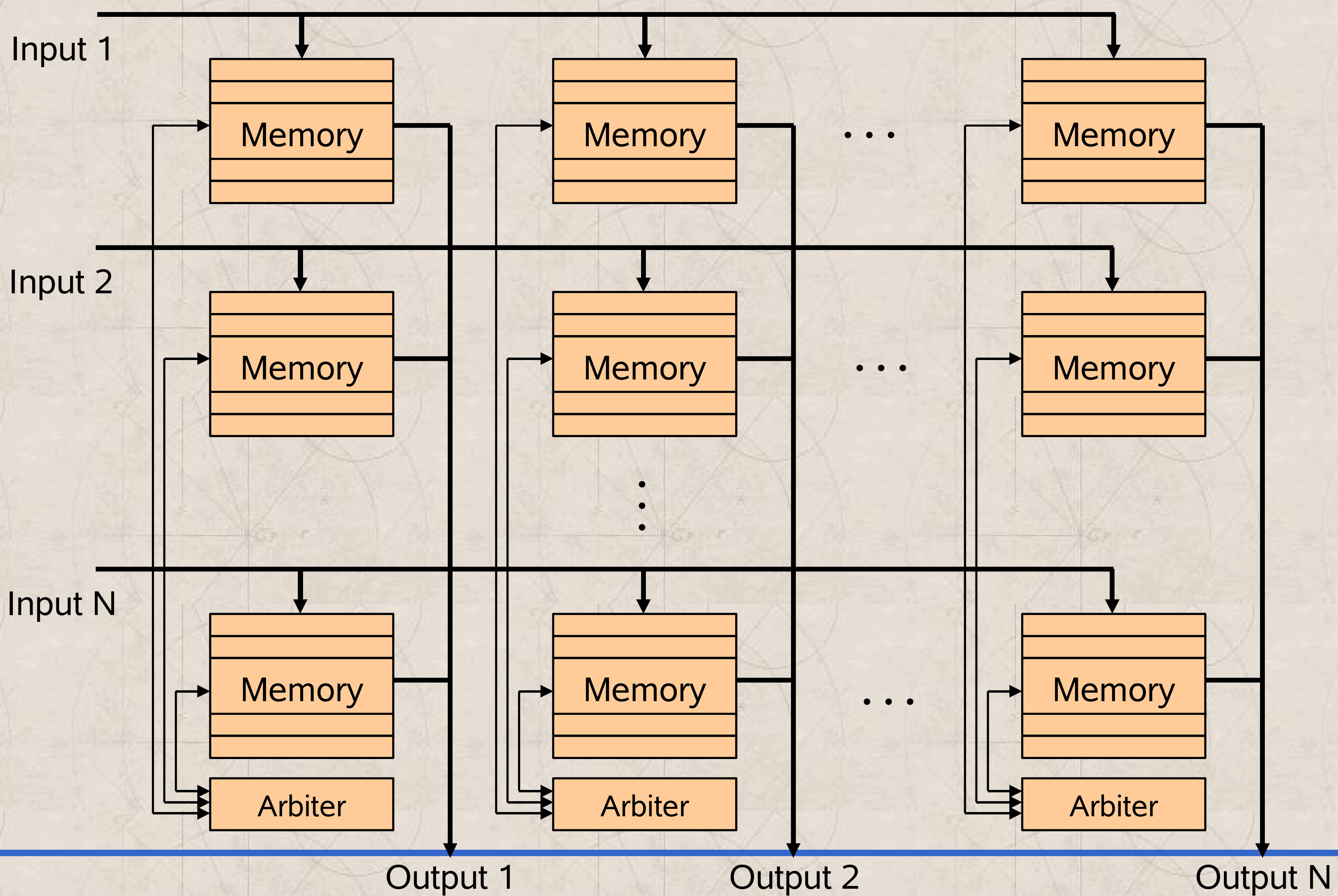
Blue Gene/L Switch

6 input & 6 ouput, 175MBps external links; 7 injection & 12 reception, 175MBps internal links

Switch Microarchitecture

Basic Switch Microarchitecture

- Buffered Crossbar Architecture



Switch Microarchitecture

Buffer Organizations

- Implemented as FIFOs, circular queues, central memory, or dynamically allocated multi-queues (*DAMQs*) in SRAMs
  - › Input ports (*input-buffered switch*)
  - › Output ports (*output-buffered switch*)
  - › Centrally within switch (*centrally-buffered switch* or *buffered Xbar*)
  - › At both input and output ports (*input-output-buffered switch*)
- Must guard against *head-of-line* (*HOL*) *blocking*
  - – Arises from two or more packets buffered in the same queue
  - – A blocked packet at the head of the queue prevents other packets in the queue from advancing that would otherwise be able to advance if they were at the queue head
  - – Output-buffered switches eliminate HOL blocking within switch
    - › *k*-way speedup required for a *k* x *k* output-buffered switch
    - › Implementations with moderate (< *k*) speedup must drop packets

Switch Microarchitecture

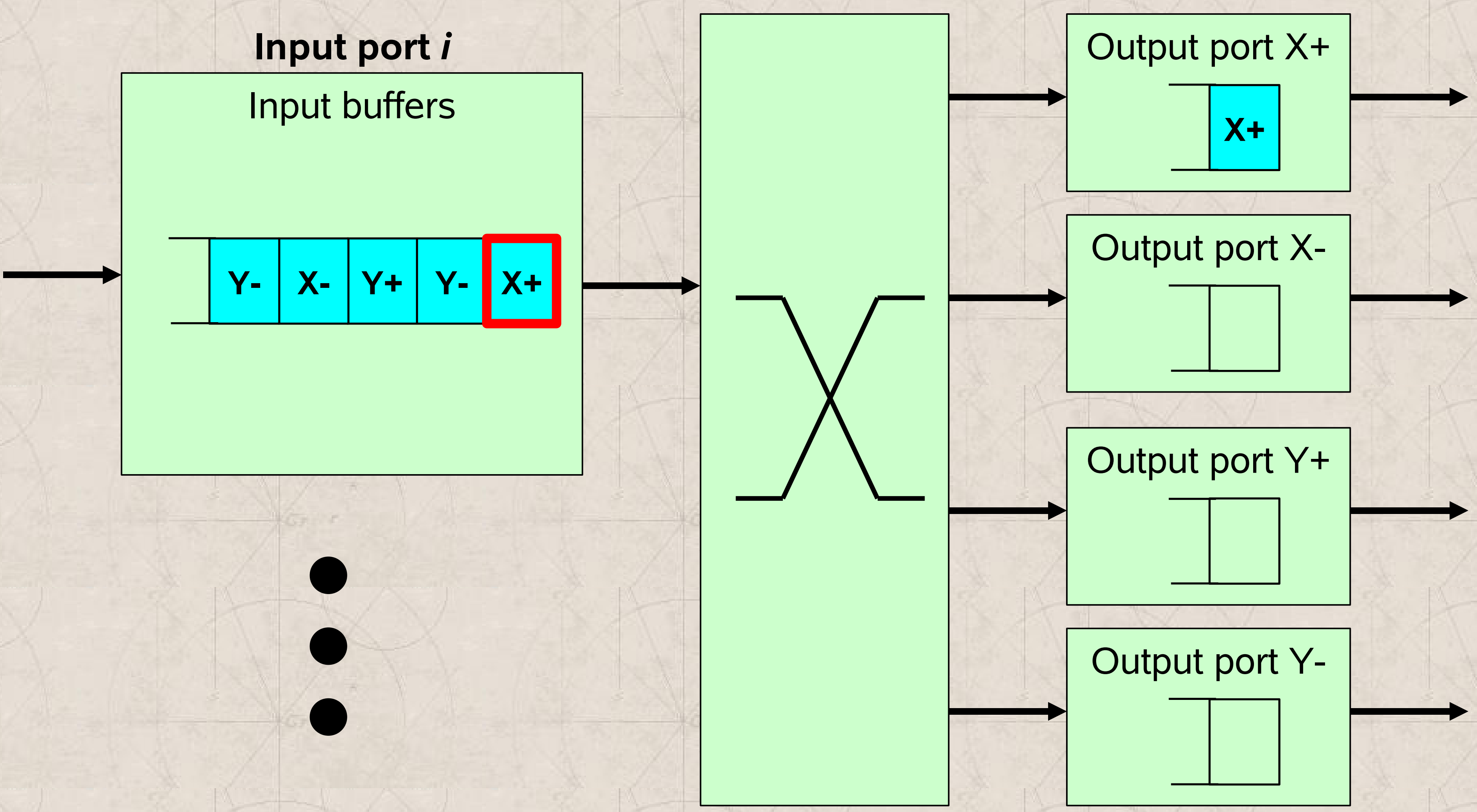
Buffer Organizations

- Head-of-line (HOL) blocking (continued)
  - Input-buffered switches do not require speedup
    - HOL blocking may appear: <60% switch efficiency w/ uniform traffic
    - *Virtual channels* can *mitigate*, but do not *eliminate*, HOL blocking
    - *Virtual Output Queues* (*VOQs*) avoid HOL blocking *within a switch*
      - As many queues in each input port as there are output ports
      - Costly, not scalable: # of queues grow quadratically w/ # ports
      - Does not eliminate HOL blocking of flows that span across multiple switches (unless as many VOQs as there are dest's)
  - Combined input-output-buffered switches
    - Reduces (but not eliminates) HOL blocking and required speedup
    - Decouples packet transmission through links and internal crossbar
  - Buffered crossbar switch
    - HOL blocking is eliminated *within a switch*
    - Again, a very expensive architecture

Switch Microarchitecture

Buffer Organizations

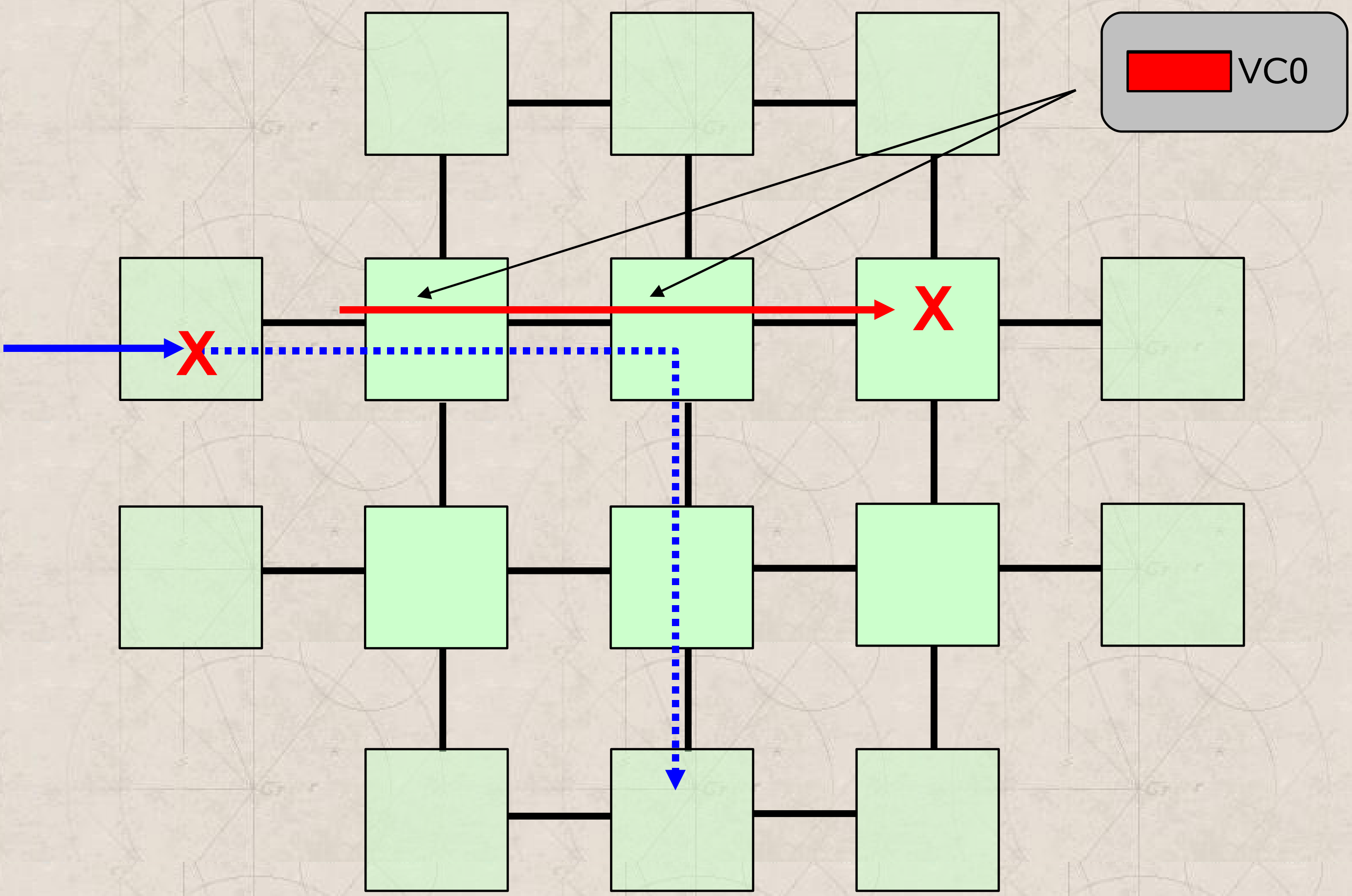
- HOL blocking at an input port



Switch Microarchitecture

Buffer Organizations

- HOL blocking at an input port using a single queue per port

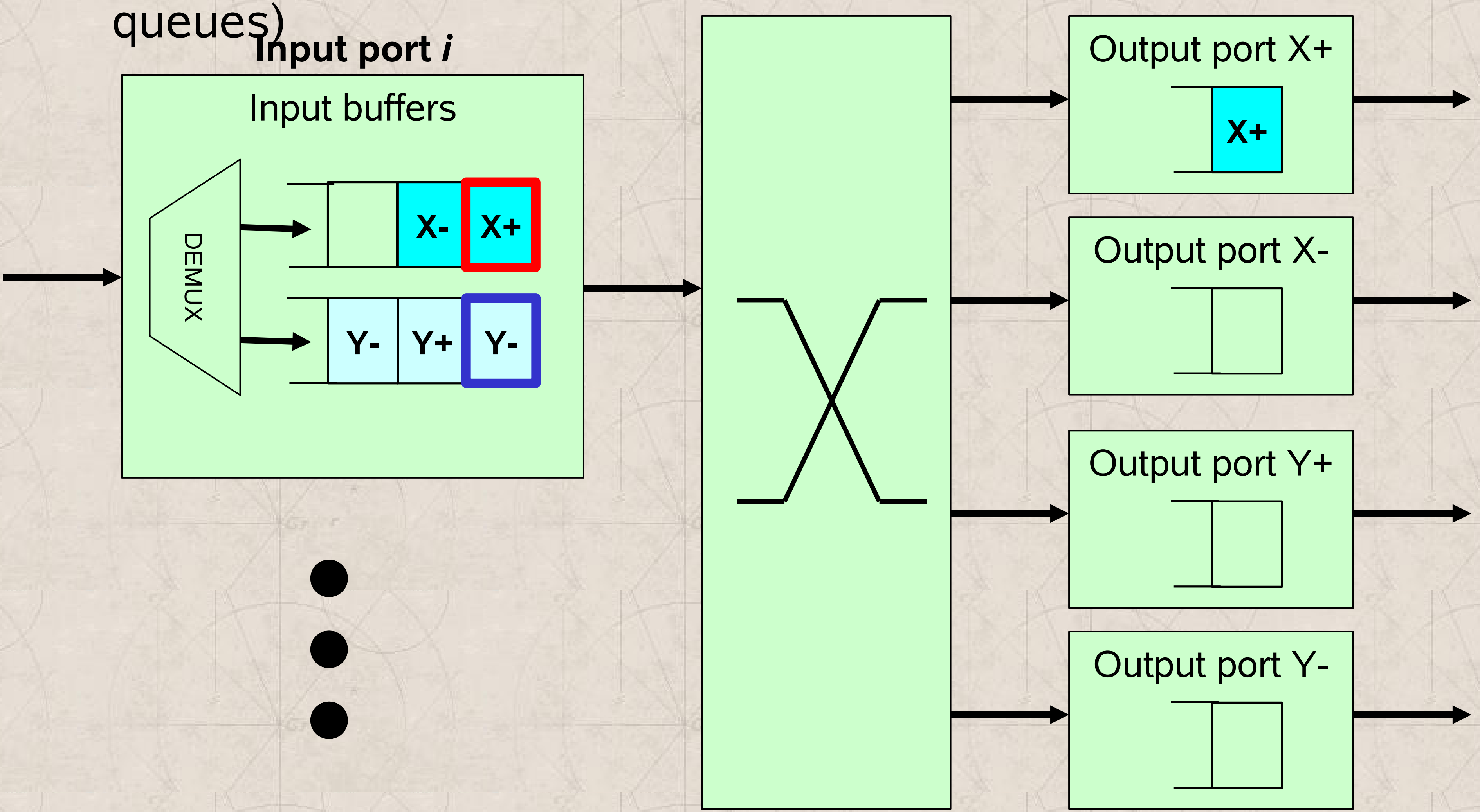


2D mesh, no VCs, DOR routing

Switch Microarchitecture

Buffer Organizations

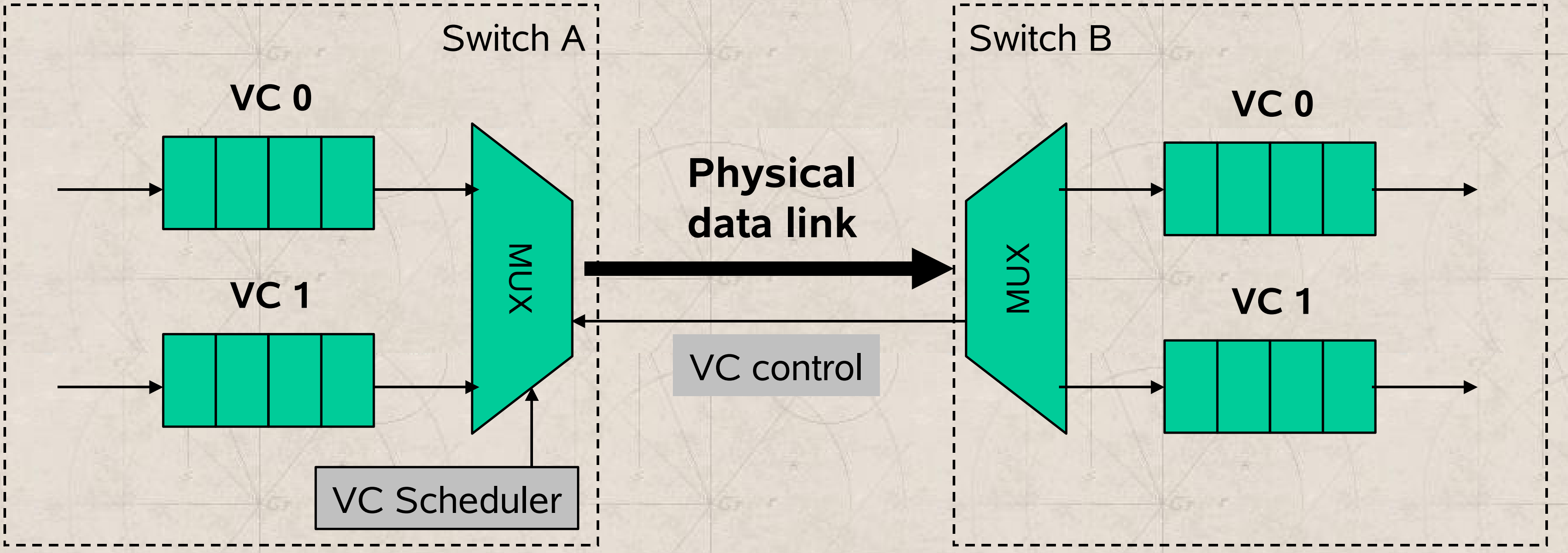
- HOL blocking is *reduced* when using *virtual channels* (2 queues)



Switch Microarchitecture

Buffer Organizations

- Use of virtual channels must be scheduled between switches

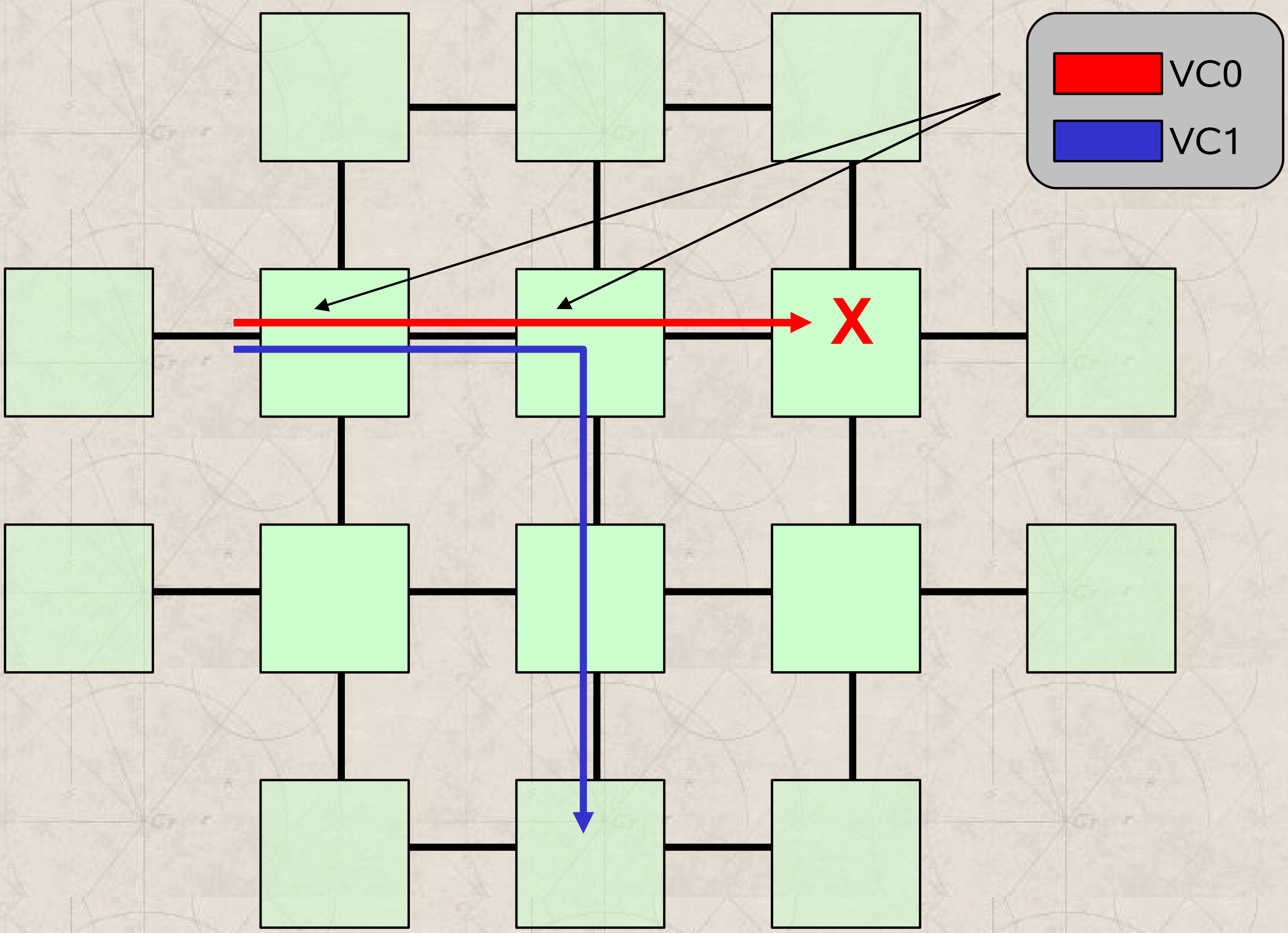


"Virtual Channel Flow Control," W. J. Dally, *IEEE Trans. on Parallel and Distributed Systems*, Vol 3, No. 2, pp. 194–205, March, 1992.

Switch Microarchitecture

Buffer Organizations

- HOL blocking *removed* when using *virtual channels* (2 queues)

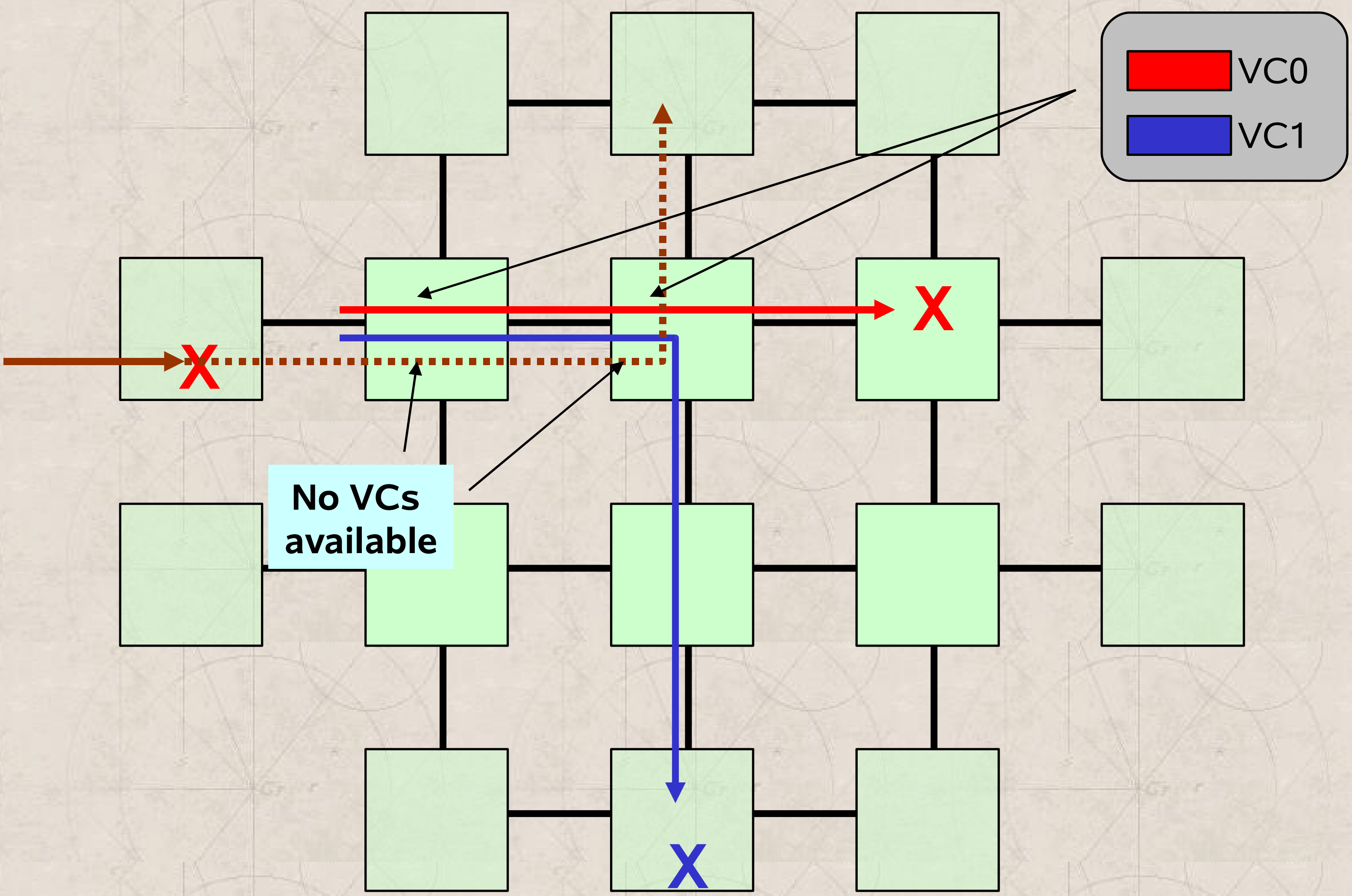


2D mesh, 2 VCs, DOR routing

Switch Microarchitecture

Buffer Organizations

- HOL blocking *remains* when using *virtual channels* (2 queues)

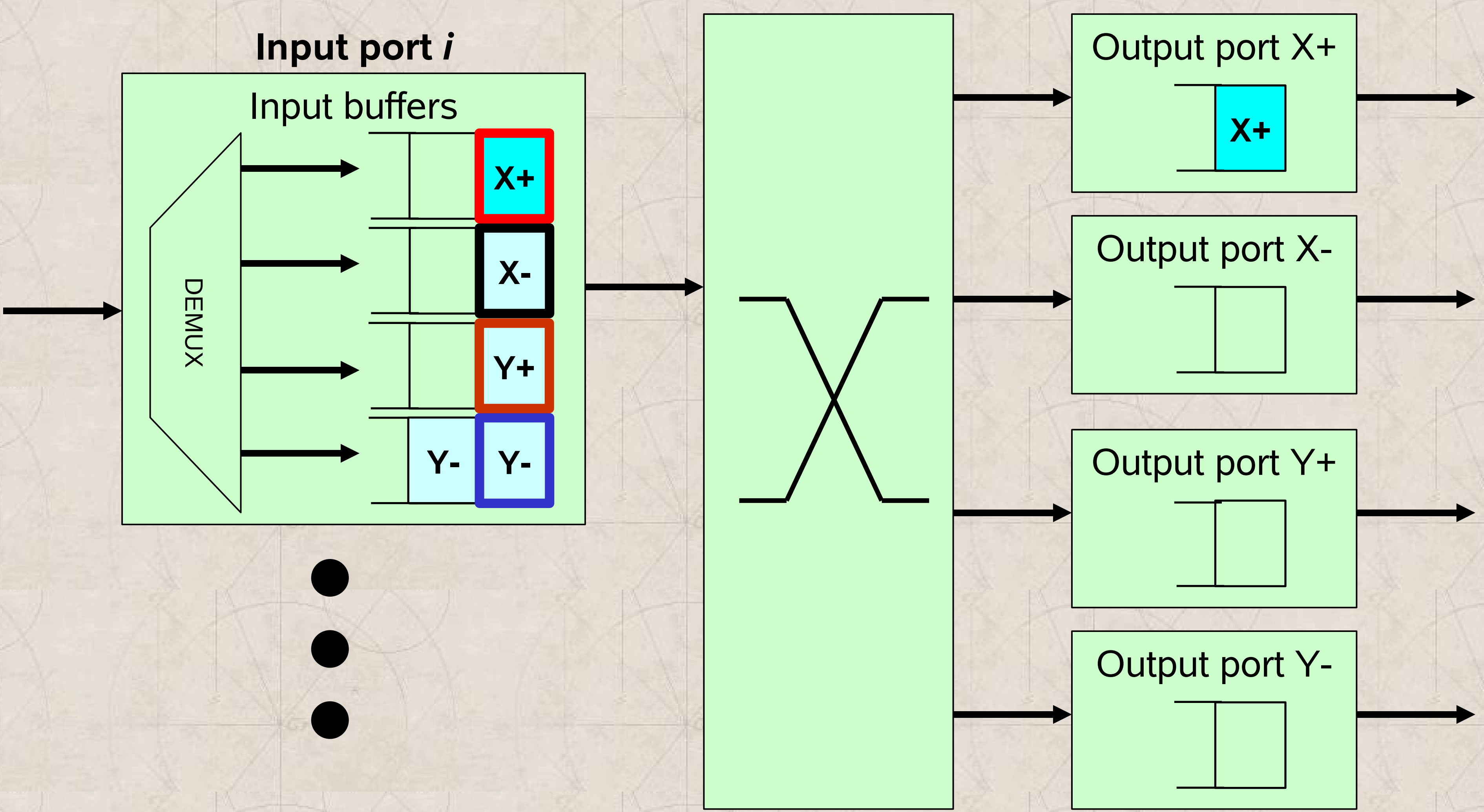


2D mesh, 2 VCs, DOR routing

Switch Microarchitecture

Buffer Organizations

- HOL blocking is *avoided at switch* using *VOQs* (need *k* queues)

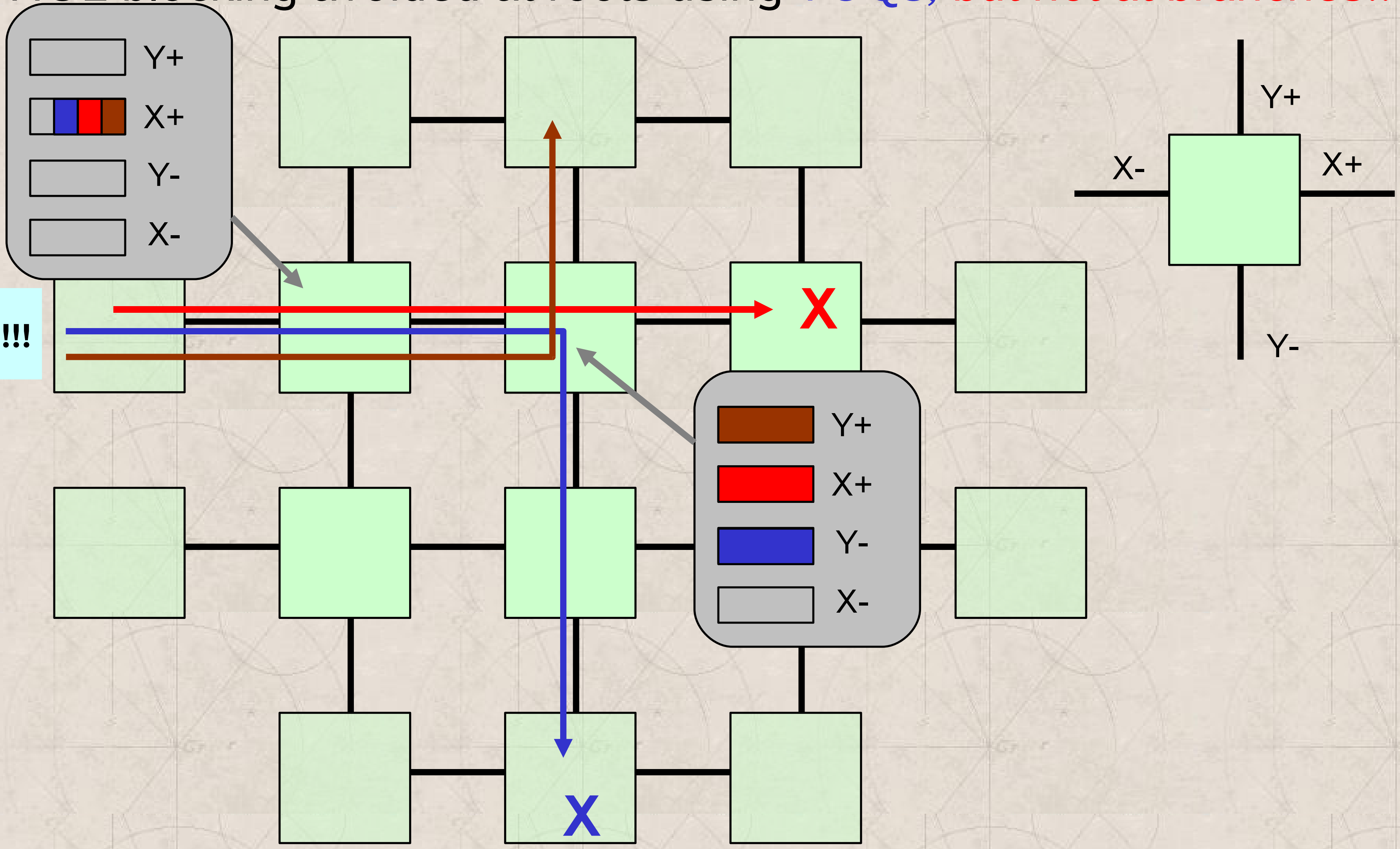


Switch Microarchitecture

Buffer Organizations

- HOL blocking avoided at roots using *VOQs, but not at branches!!*

HOL blocking at neighboring switch!!



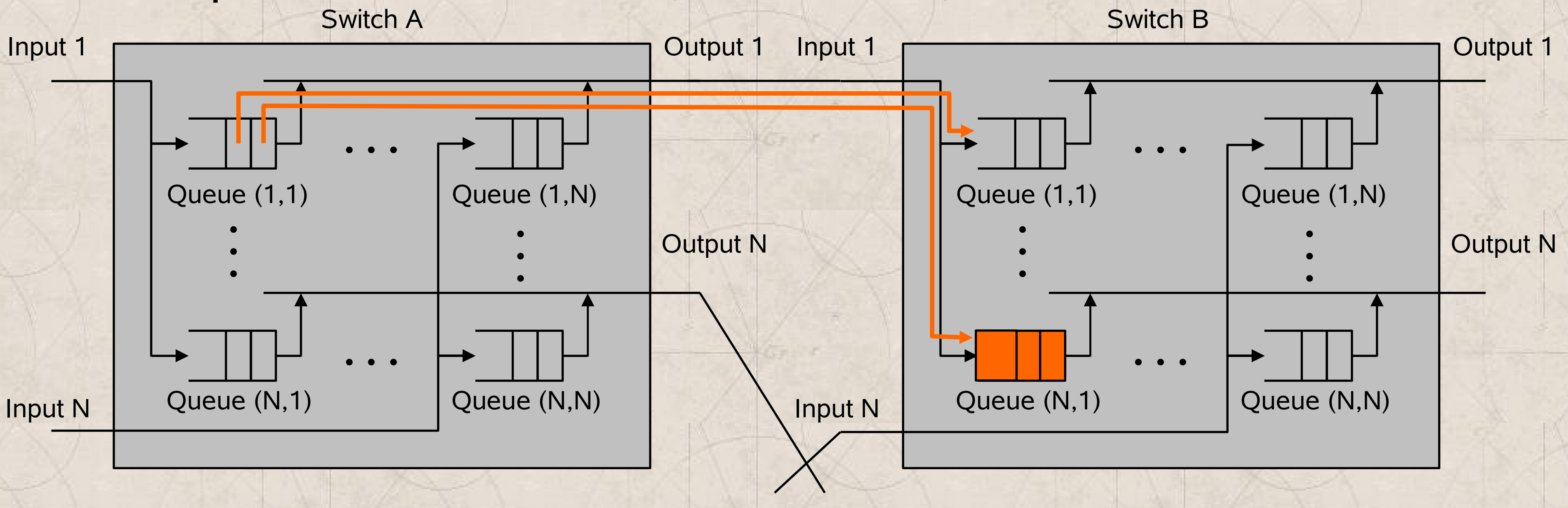
However!!!!

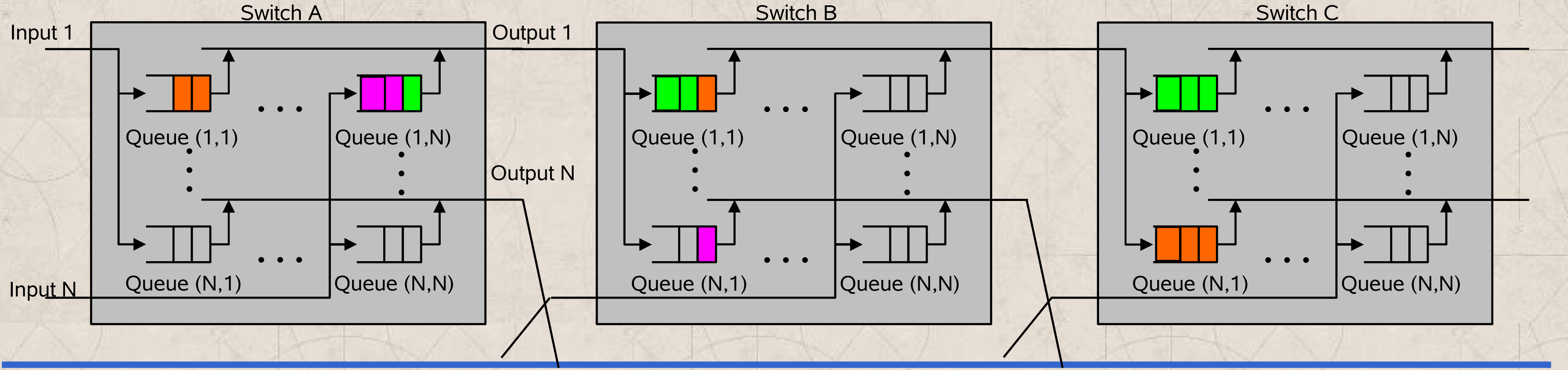
2D mesh, VOQs, DOR routing

Switch Microarchitecture

Buffer Organizations

- Implementation of VOQs via DAMQs



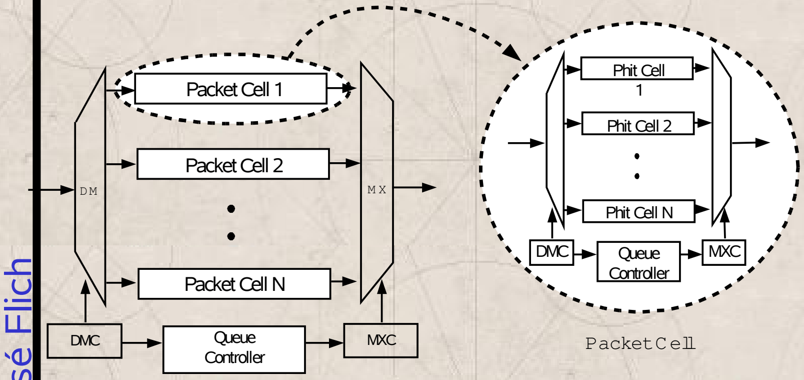


"Dynamically-Allocated Multi-Queue Buffers for VLSI Communication Switches," Y. Tamir and G. Frazier, *IEEE Trans. on Computers*, Vol. 41, No. 6, pp. 725–734, June, 1992.

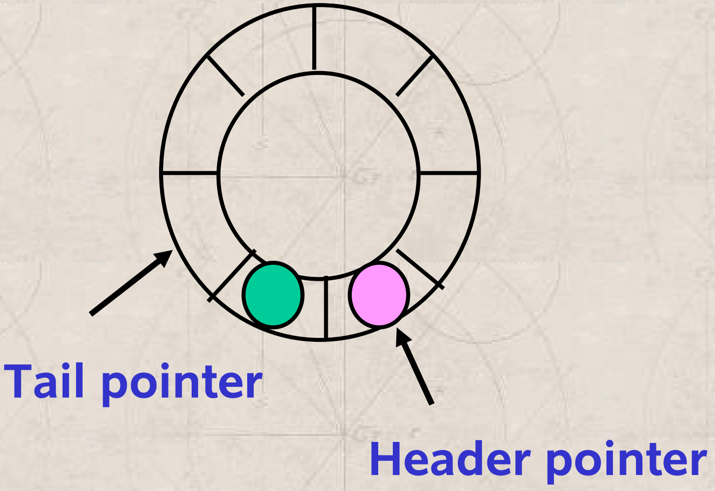
Switch Microarchitecture

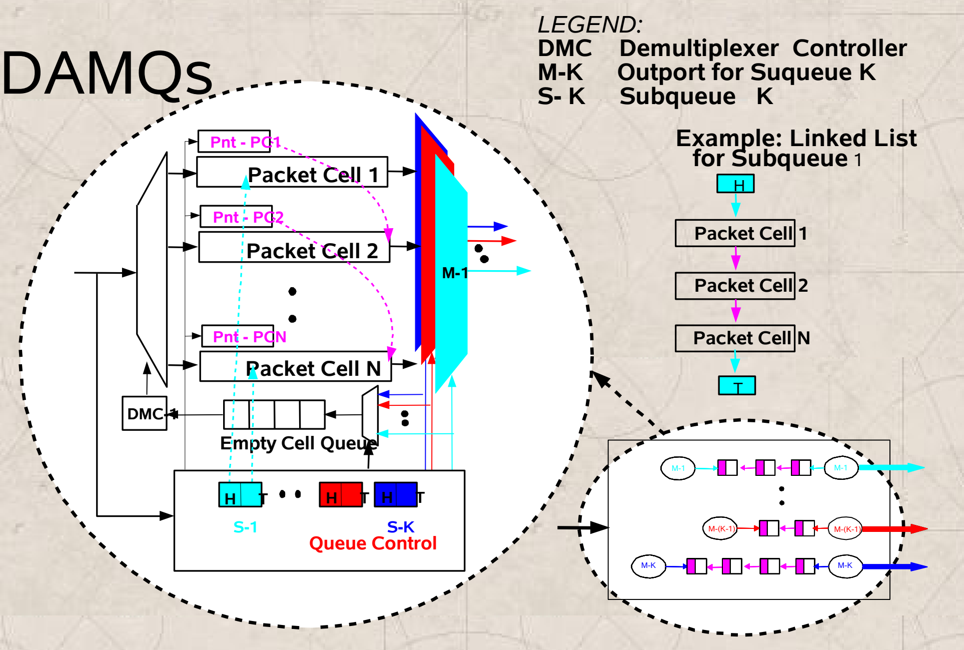
Buffer Organizations

- Implementing Circular Q’s & DAMQs

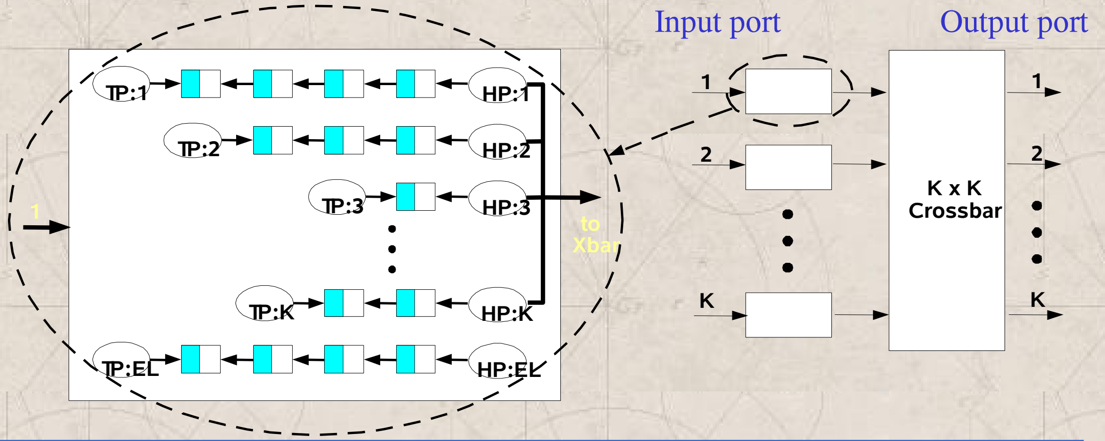


Circular Queue





DAMQ

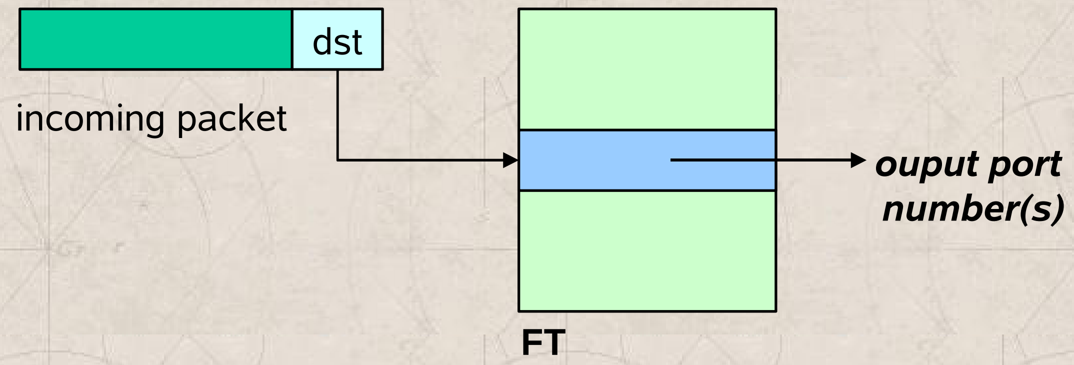


“Evaluation of Queue Designs for True Fully Adaptive Routers,” Y. Choi and T. Pinkston, *Journal of Parallel and Distributed Computing*, Vol. 64, No. 5, pp. 606–616, May, 2004.

Switch Microarchitecture

Routing and Arbitration Unit

- Usually is implemented as a centralized resource
  - Routing done on a per-packet basis
- Finite-state machine (FSM)
  - Based on routing information in the header, FSM computes the output port(s) (several if adaptive routing)
  - Routing info at header is usually stripped off or modified
- Forwarding table (FT)
  - Routing info used as an address to access the forwarding table
  - FT must be preloaded into switches



Switch Microarchitecture

Routing and Arbitration Unit

- Required when two or more packets request the same output port at the same time
- Centralized implementation
  - Request and status info transmitted to the arbitration unit
- Distributed implementation
  - Arbiter distributed among input and/or output ports
- Hierarchical arbitration
  - Local arbitration and global arbitration
  - Multiple arbitrations occur on each packet
  - Large number of arbitration requests (multiple queues per port)

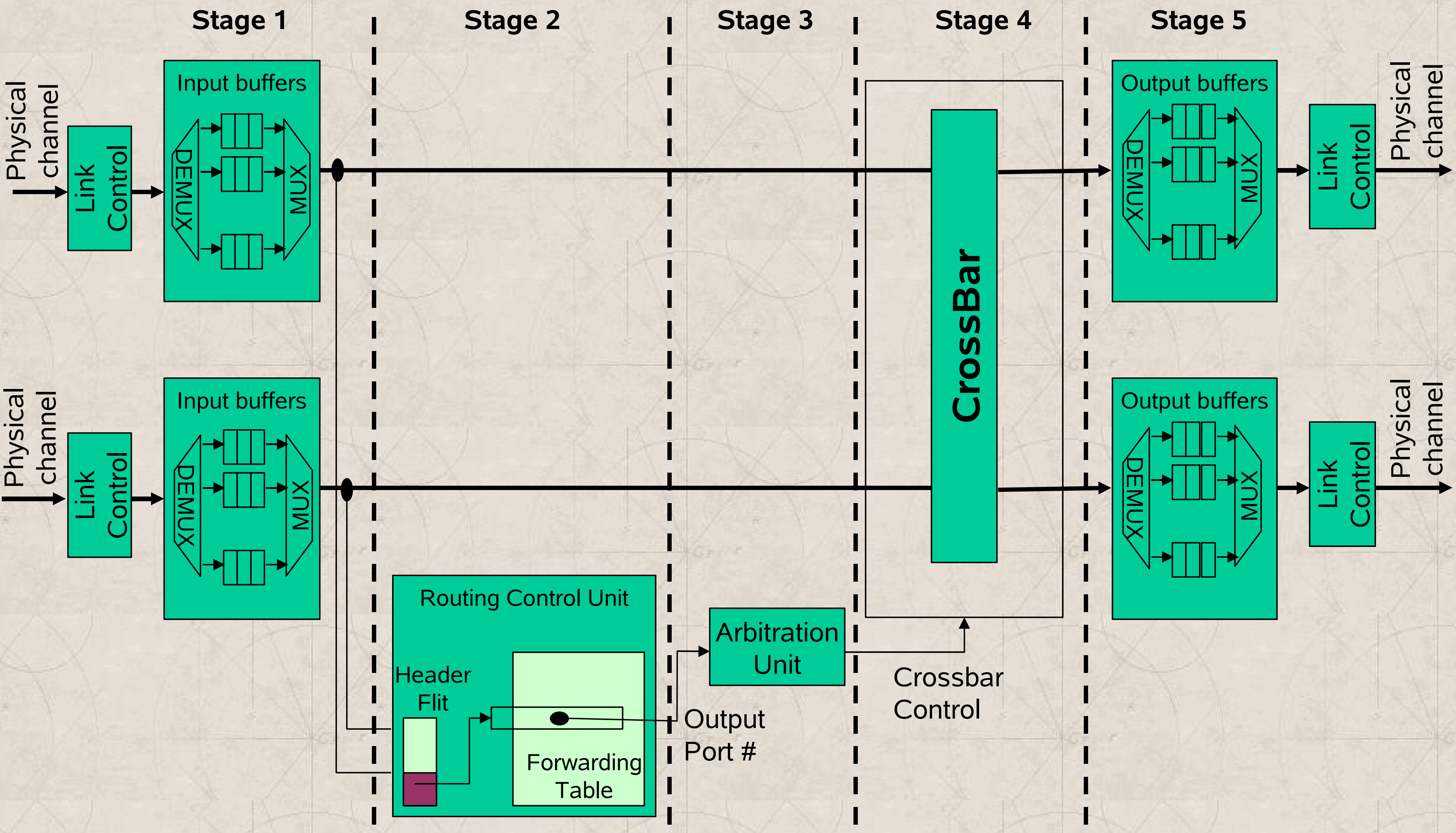
Switch Microarchitecture

Pipelining the Switch Microarchitecture

- Similarities with vector processors
  - Packet header indicates how to process the *ph*ysical un*its* (*phits*)
- Packets at different input ports are independent
  - Parallelism

Switch Microarchitecture

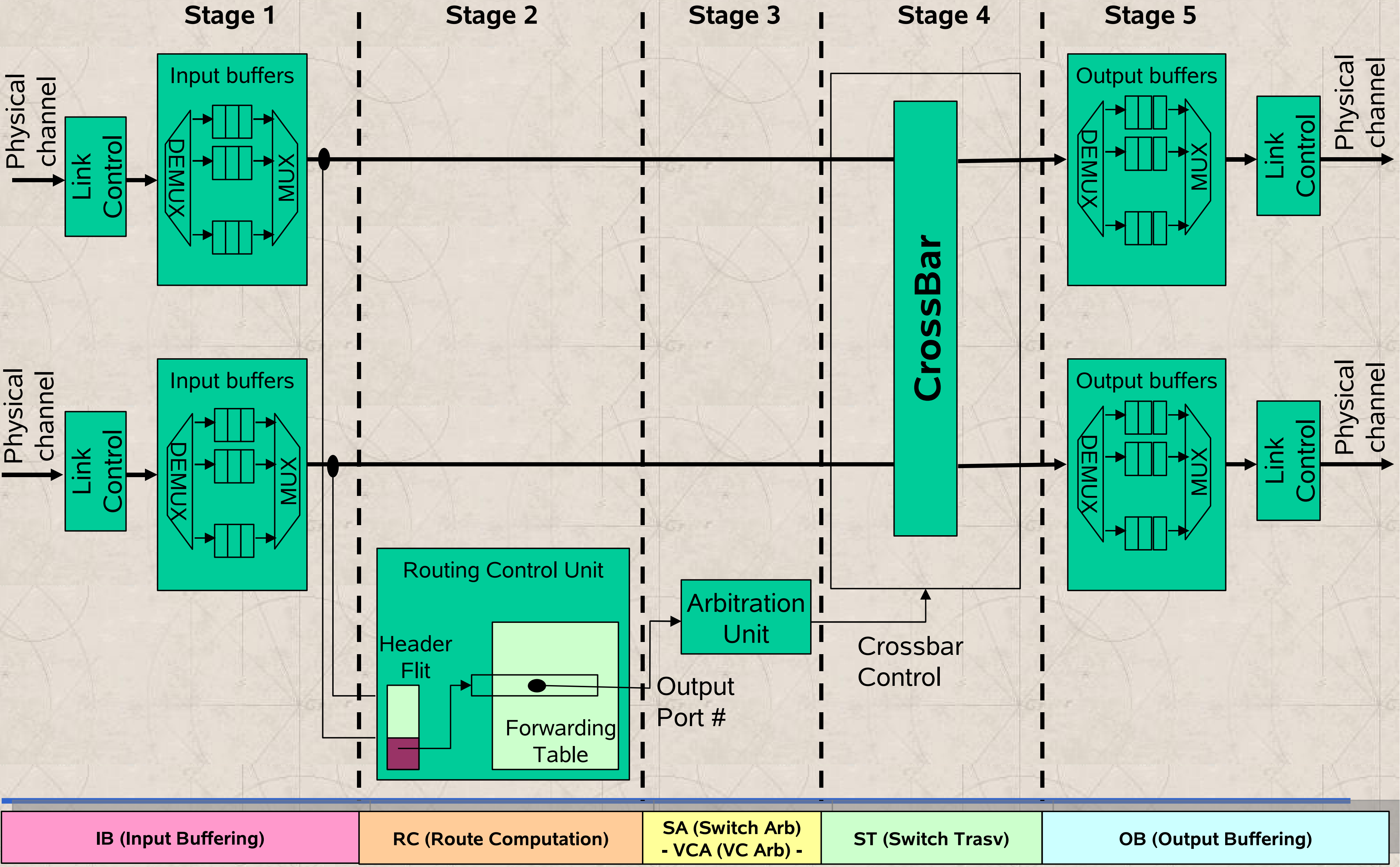
Pipelining the Switch Microarchitecture



IB (Input Buffering)	**RC (Route Computation)**	**SA (Switch Arb) - VCA (VC Arb) -**	**ST (Switch Trasv)**	**OB (Output Buffering)**

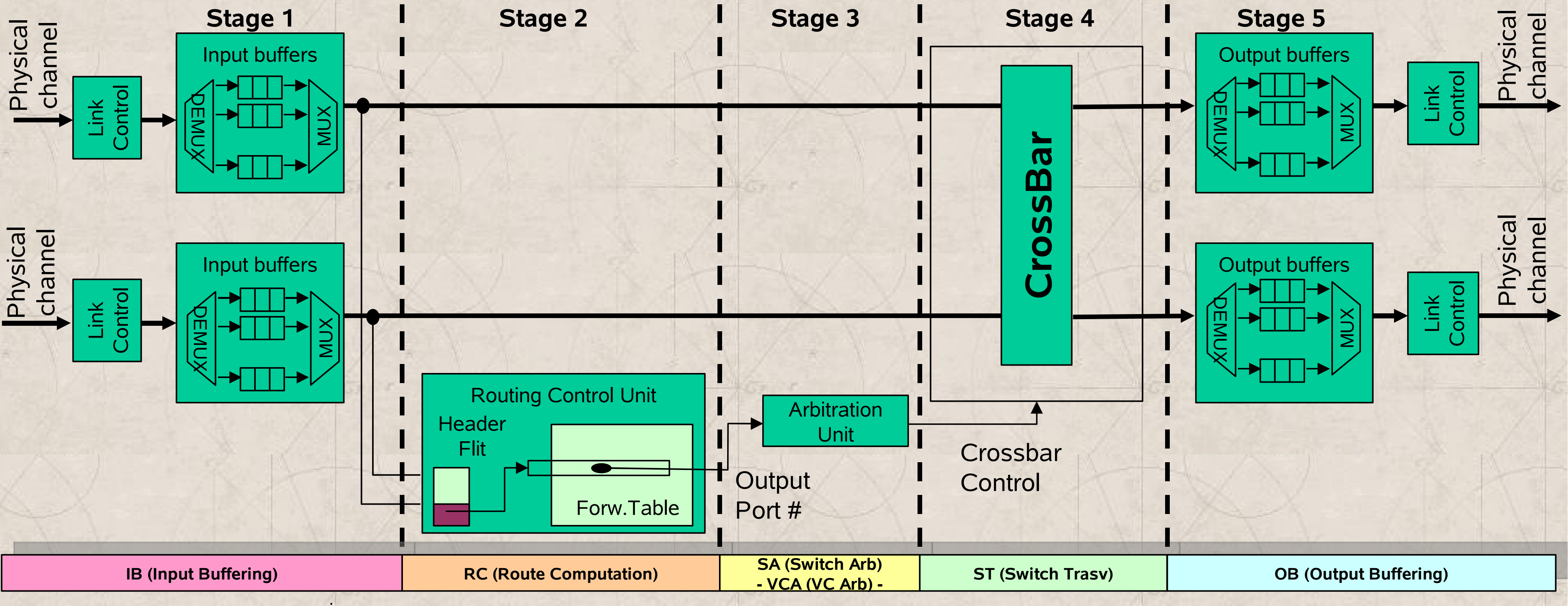
Switch Microarchitecture

Pipelining the Switch Microarchitecture



Switch Microarchitecture

Pipelining the Switch Microarchitecture



Packet header	IB	RC	SA	ST	OB		
Payload fragment		IB	IB	IB	ST	OB	
Payload fragment			IB	IB	IB	ST	OB
Payload fragment				IB	IB	IB	ST	OB

Matching the throughput of the internal switch datapath to the external link BW can increase ρ

(i.e., $\rho_{\mu Arch}$)

"A Delay Model and Speculative Architecture for Pipelined Routers," L. S. Peh and W. J. Dally, *Proc. of the 7th Int'l Symposium on High Performance Computer Architecture*, Monterrey, January, 2001.

Switch Microarchitecture

Characterizing Performance: Latency & Effective Bandwidth

- Values for T_r , T_a , T_s are determined by switch μarchitecture
  - pipelining; implementation of queuing, routing, and arbitration
- Network efficiency factor, ρ, is influenced by switch μarchitecture
  - internal switch speedup & reduction of contention *within* switches
  - buffer organizations to mitigate HOL blocking in & across switches
  - $\rho = \rho_L \times \rho_R \times \rho_A \times \rho_S \times \rho_{\mu Arch} \times \ldots$

$$\textit{Latency} = \textit{Sending overhead} + T_{LinkProp} \times (d+1) + (T_r + T_a + T_s) \times d + \frac{\text{Packet} + (d \times \text{Header})}{\textit{Bandwidth}} + \textit{Receiving overhead}$$

(cut-through switching)

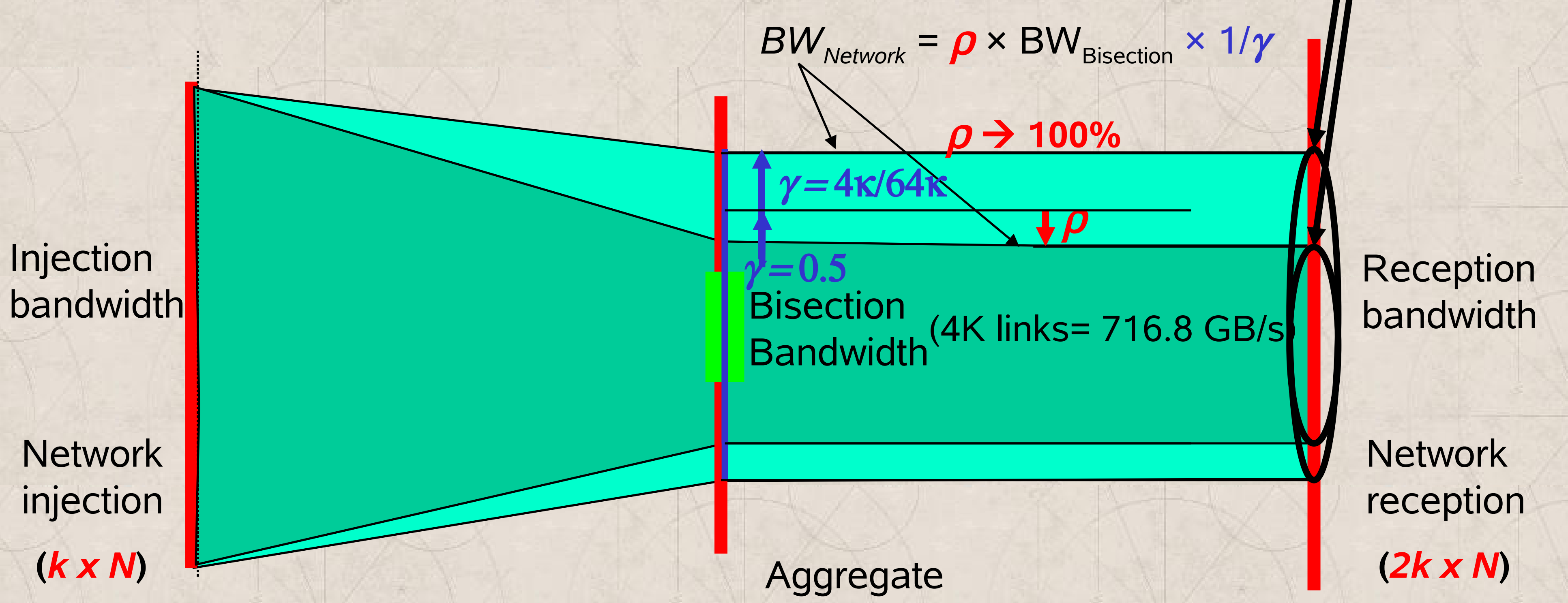
lower bound (contention delay not included)

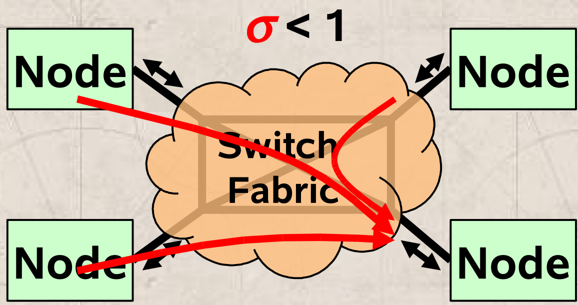
upper bound (contention effects not *fully* included)

$$\textit{Effective bandwidth} = \min\left(N \times BW_{LinkInjection},\ \underbrace{\frac{\rho \times BW_{Bisection}}{\gamma}}_{BW_{Network}},\ \sigma \times N \times BW_{LinkReception}\right)$$

Switch Microarchitecture

Characterizing Performance: Latency & Effective Bandwidth





BG/L: 3D torus, N = 32 x 32 x 64 = 64K nodes

$2k$ rec links/node increases rec bandwidth when σ <1

$BW_{Bisection}$ = 32 x 32 x 4 = 4096 links

γ depends on traffic, i.e., γ= 0.5 for uniform traffic,

ρ depends on routing, switching, arbitration (<100%)

Effective bandwidth = ρ x 1.434 TB/s max.

γ= 4K/64K for near-neighbour traffic

ρ → 100% for near-neighbour traffic

Effective bandwidth=11.47 TB/s max

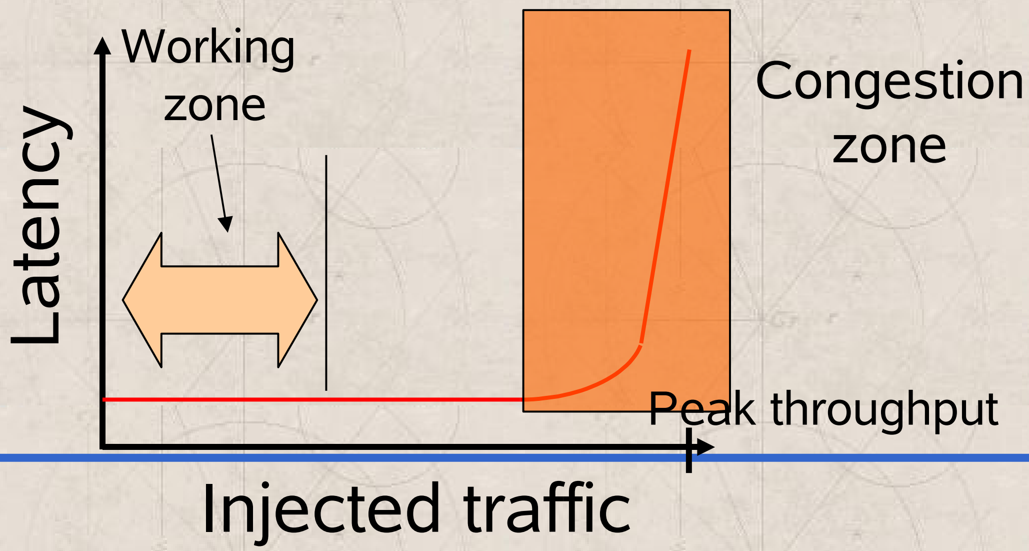
Outline

- E.1 Introduction *(Lecture 1)*
- E.2 Interconnecting Two Devices *(Lecture 1)*
- E.3 Interconnecting Many Devices *(Lecture 2)*
- E.4 Network Topology *(Lecture 2)*
- E.5 Network Routing, Arbitration, and Switching *(Lecture 3)*
- E.6 Switch Microarchitecture *(Lecture 4)*
- **E.7 Practical Issues for Commercial Interconnection Networks** *(Lecture 4)*
  - Connectivity (skipped)
  - Standardization (skipped)
  - **Congestion Management**
  - **Fault Tolerance**
- E.8 Examples of Interconnection Networks *(Lecture 5)*
- E.9 Internetworking (skipped)
- E.10 Crosscutting Issues for Interconnection Networks (skipped)
- E.11 Fallacies and Pitfalls *(Lecture 5)*
- E.12 Concluding Remarks and References *(Lecture 5)*

Practical Issues for Interconnection Networks

Congestion Management

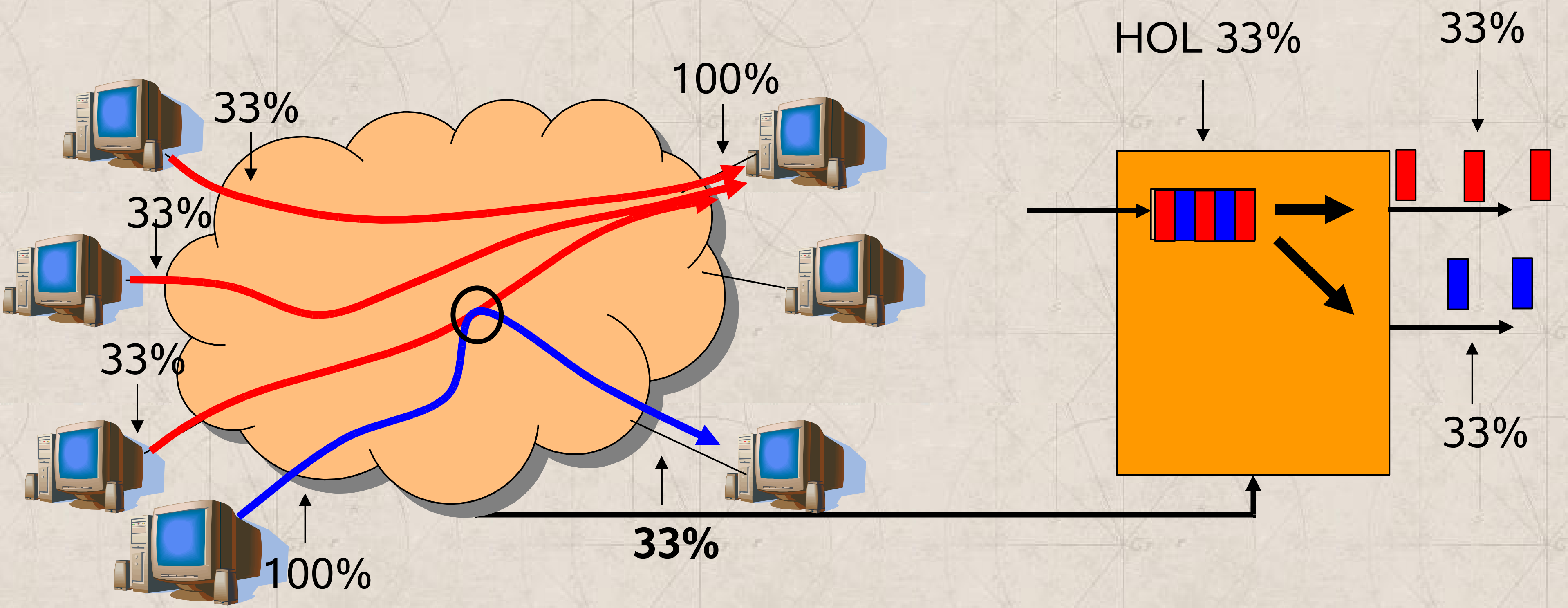
- *Congestion*
  - – Arises when too many packets use the same set of resources
  - – By itself, congestion does not degrade performance
    - › Congested links are simply running at their maximum capacity
  - – *HOL blocking* resulting from congestion can degrade performance of non-congested flows
    - › Non-congested flows whose paths cross a congestion tree may get throttled unnecessarily due to HOL blocking in shared resources



Practical Issues for Interconnection Networks

Congestion Management

- The Real Problem: *HOL blocking*



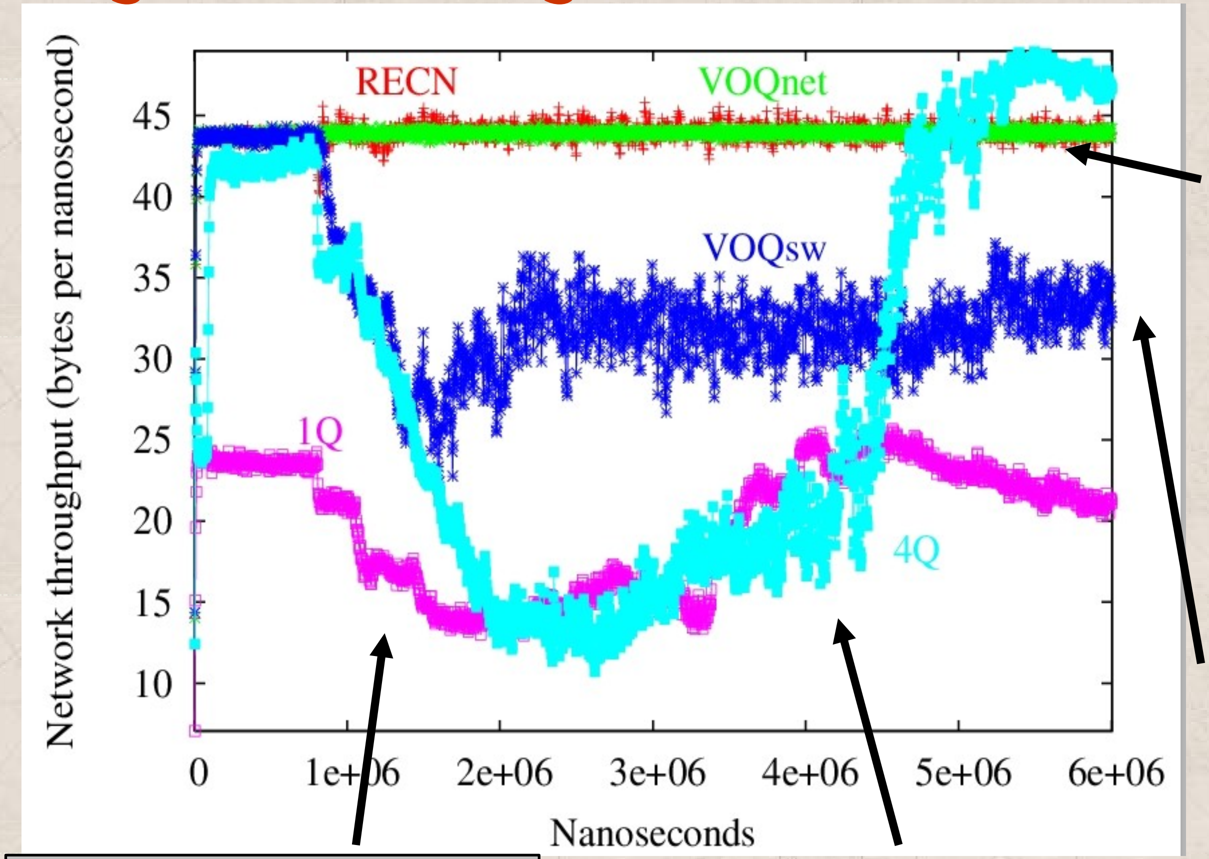
Congestion trees introduce HOL blocking,
which degrades network performance dramatically
(i.e., the non-congested blue flow is reduced to 33% throughput)

Minimizing (or eliminating) HOL blocking can increase ρ

Congestion Management

- Eliminating ***network level*** HOL blocking (not just at switches)
  - – Virtual Output Queuing (VOQ) applied at the network level
    - › A separate queue at *each* switch input port is required for *every* network end node (*destination)*
    - › Required resources grows at least quadratically with network size
  - – Regional Explicit Congestion Management (RECN)
    - › Congestion trees are exactly identified and traffic is segregated
    - › Packets belonging to congested flows are stored in separate Set Aside Queues (SAQs)
    - › Packets belonging to non-congested flows stored in a set of “common” queues
    - › Requires SAQs + “common” queues for each input port
      - » Much more scalable than VOQ applied at the network level
    - › Applicable to deterministic source routing only

Congestion Management



RECN dynamically detects and isolates congestion trees; thus HOL blocking is Eliminated and maximum performance is achieved

VOQsw also shows degraded performance; it does not completely eliminate the HOL blocking problem

1Q suffers from massive HOL blocking even when no congestion trees are in the network

4 VCs do not eliminate HOL blocking; instead, the congestion tree spreads over all the VCs

- 64 x 64 BMIN, uniform traffic
- suddenly a hot-spot occurs
- RECN with just 8 SAQs

"A New Scalable and Cost-effective Congestion Management Strategy for Lossless Multistage Interconnection Networks,"
J. Duato, I. Johnson, J. Flich, F. Naven, P. Garcia, T. Nachiondo, *Proc. 11th HPCA*, San Francisco, February, 2005.

Practical Issues for Interconnection Networks

Fault Tolerance

- Probability of system failure increases with
  - transistor integration density
  - number of interconnected devices
- Types of failures
  - Transient
    - Caused by electromagnetic interference
    - Recover by retransmitting packets at link level or end-to-end
  - Permanent
    - Resulting from some component not working within specifications
    - Caused by defects, overheating, over-biasing, overuse, aging, etc.
    - Recover by supplying alternative paths on which packets can route

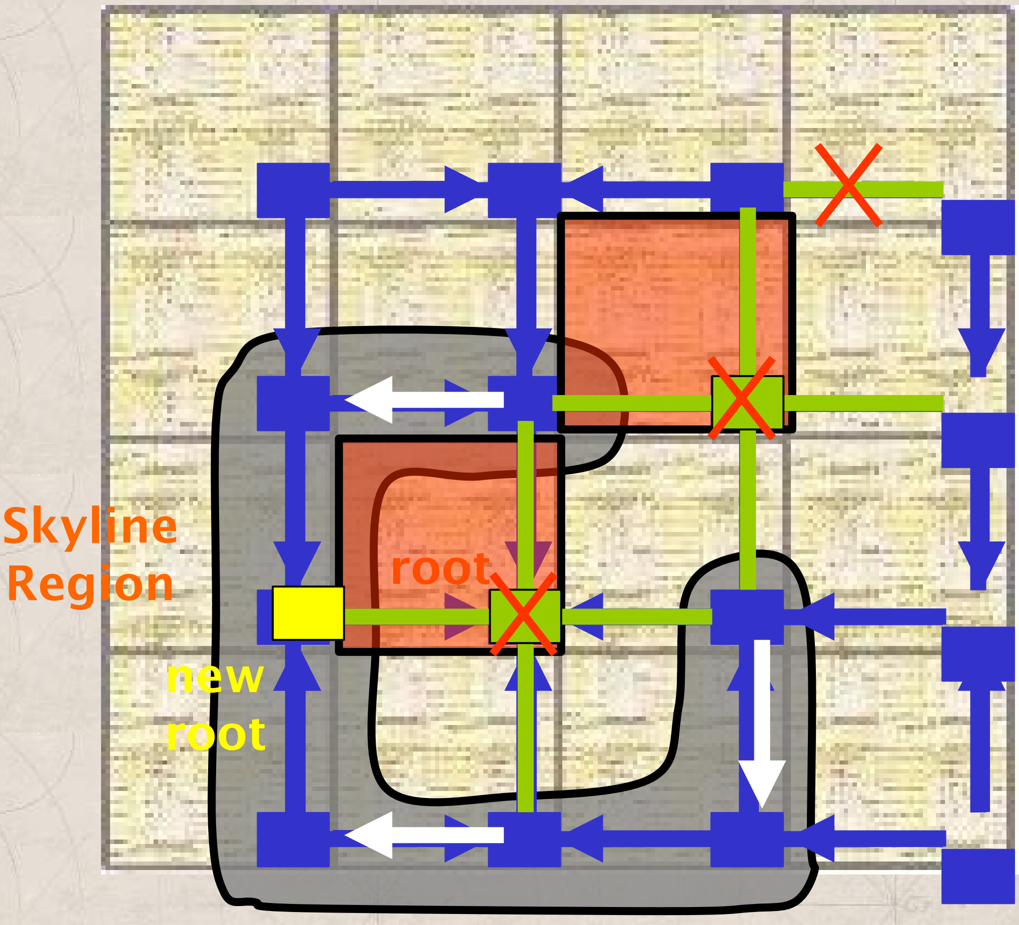
Practical Issues for Interconnection Networks

Fault Tolerance

- Three categories of techniques to deal with permanent failures
  - Resource sparing (redundancy)
    - › Faulty resources are bypassed and spare ones are switched in
      - » ServerNet, IBM Blue Gene/L (healthy resources be removed)
  - Fault-tolerant routing
    - › Alternative paths are incorporated into routing function from start
      - » Cray T3E
    - › May not account for all (many) possible fault combinations
  - Network reconfiguration
    - › A more general, less costly technique
      - » Myrinet, Quadrics, InfiniBand, Advanced Switching, etc.
    - › Routing function (i.e., forwarding tables) reconfigured either statically or dynamically (hot swapping) to reconnect the network
    - › Must guard against reconfiguration-induced deadlocks
      - » More than one routing function may be active (conflict) at a time

Practical Issues for Interconnection Networks

Reducing chip-kill in the presence of permanent faults with *dynamic reconfiguration* of on-chip networks



- A 2-D mesh network with XY dimension-order routing (DOR)
- If a core switch & link is faulty → causes five failed links
- Network can *dynamically* be reconfigured to up*/down* routing *remaining deadlock-free!*
- Later, if the u*/d* root fails → causes four links to fail
- Only the up*/down* link directions within the skyline region are affected by the fault
- Reconfigured again to regain connectivity → no chip-kill!!

"Part II: A Methodology for Developing Dynamic Network Reconfiguration Processes," O. Lysne, T. Pinkston, and J. Duato, *IEEE Trans. on Parallel and Distributed Systems* , Vol. 16, No. 5, pp. 428–443, May, 2005.

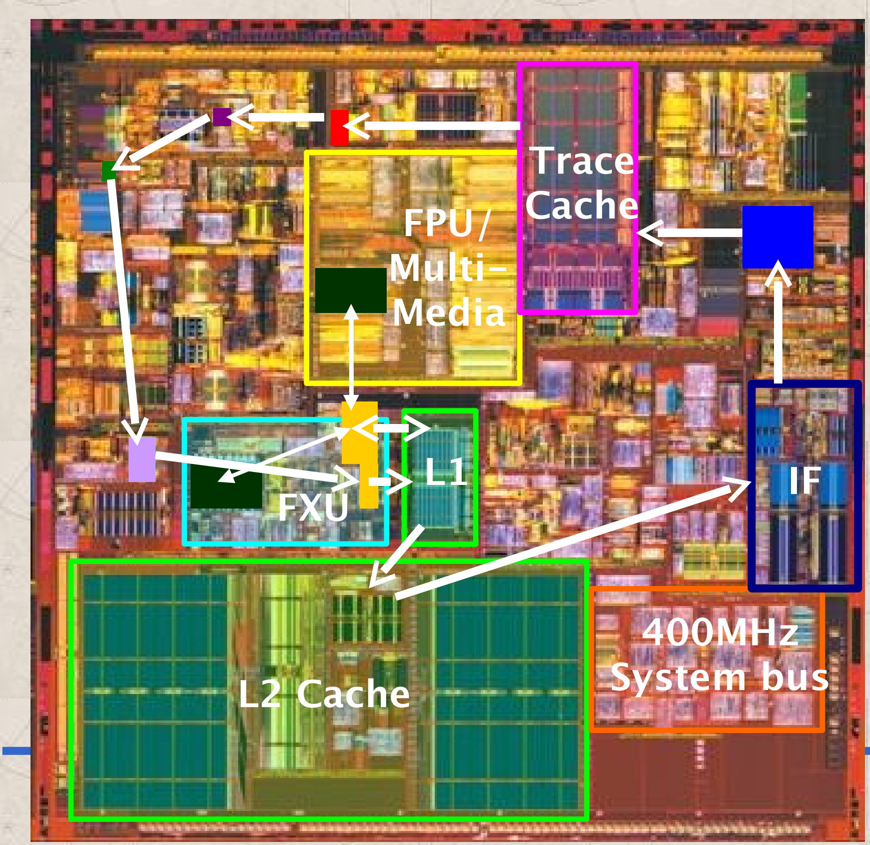
Outline

- E.1 Introduction *(Lecture 1)*
- E.2 Interconnecting Two Devices *(Lecture 1)*
- E.3 Interconnecting Many Devices *(Lecture 2)*
- E.4 Network Topology *(Lecture 2)*
- E.5 Network Routing, Arbitration, and Switching *(Lecture 3)*
- E.6 Switch Microarchitecture *(Lecture 4)*
- E.7 Practical Issues for Commercial Interconnection Networks *(Lec 4)*
- **E.8 Examples of Interconnection Networks** *(Lecture 5)*
  - **On-chip Networks (OCNs)**
  - **Cell Broadband Engine Element Interconnect Bus**
  - **System Area Networks (SANs)**
  - **Blue Gene/L 3D Torus Network**
- E.9 Internetworking (skipped)
- E.10 Crosscutting Issues for Interconnection Networks (skipped)
- E.11 Fallacies and Pitfalls *(Lecture 5)*
- E.12 Concluding Remarks and References *(Lecture 5)*

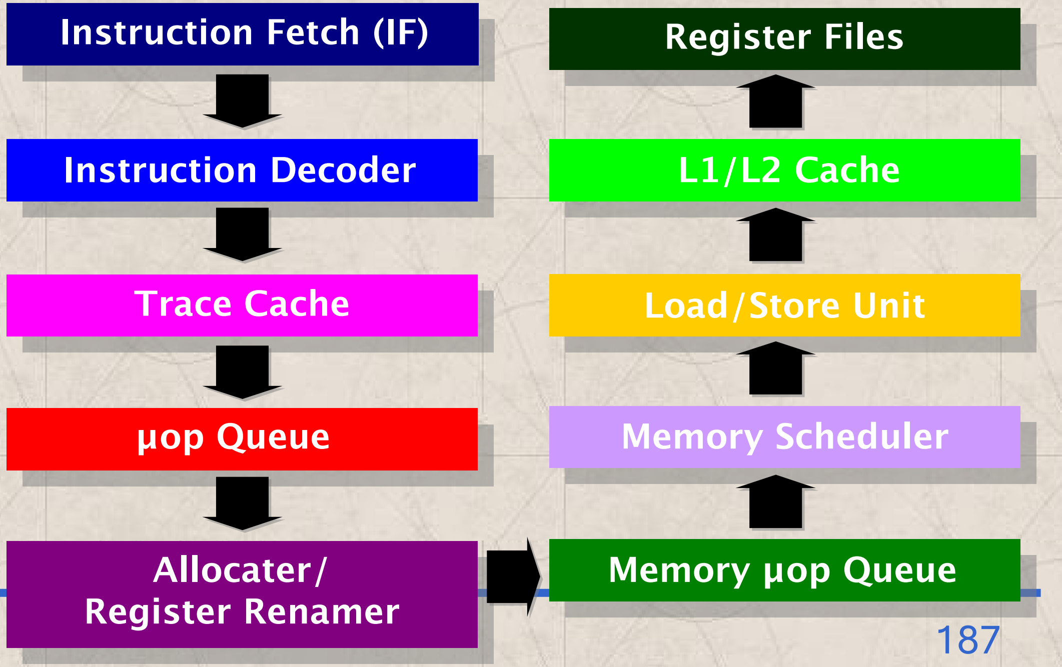
Examples of Interconnection Networks

On-Chip Networks (OCNs)

- *Multicore* architectures are displacing monolithic single cores
  - – Power/area/performance-efficiency: better with multiple simpler cores than with fewer (single) more complex cores
  - – Memory wall: cache miss latencies hidden with multiple threads
  - – Interconnect: wire delay more scalable (fewer chip-crossings)



Memory Operations (load or store instructions)


Instruction Fetch (IF)

Instruction Decoder

Trace Cache

µop Queue

Allocater/ Register Renamer

Memory µop Queue

Memory Scheduler

Load/Store Unit

L1/L2 Cache

Register Files

Examples of Interconnection Networks

On-Chip Networks (OCNs)

Institution & Processor [Network] name	Year built	Number of network ports [cores or tiles + other ports]	Basic network topology	# of data bits per link per direction	Link bandwidth [link clock speed]	Routing; Arbitration; Switching	# of chip metal layers; flow control; # VCs
MIT Raw [General Dynamic Network]	2002	16 port [16 tiles]	2-D mesh 4 x 4	32 bits	0.9 GBps [225 MHz, clocked at proc speed]	XY DOR w/ request-reply deadlock recovery; RR arbitration; wormhole	6 layers; credit-based; no VCs
IBM POWER5	2004	7 ports [2 PE cores + 5 other ports]	Crossbar	256 b Inst fetch; 64 b for stores; 256 b LDs	[1.9 GHz, clocked at proc speed]	Shortest-path; non-blocking; circuit switch	7 layers; handshaking; no virtual channels
U.T. Austin TRIPS EDGE [Operand Network]	2005	25 ports [25 execution unit tiles]	2-D mesh 5 x 5	110 bits	5.86 GBps [533 MHz clk scaled by 80%]	XY DOR; distributed RR arbitration; wormhole	7 layers; on/off flow control; no VCs
U.T. Austin TRIPS EDGE [On-Chip Network]	2005	40 ports [16 L2 tiles + 24 network interface tile]	2-D mesh 10 x 4	128 bits	6.8 GBps [533 MHz clk scaled by 80%]	XY DOR; distributed RR arbitration; VCT switched	7 layers; credit-based flow control; 4 VCs
Sony, IBM, Toshiba Cell BE [Element Interconnect Bus]	2005	12 ports [1 PPE and 8 SPEs + 3 other ports for memory, I/&O interface]	Ring 4 total, 2 in each direction	128 bits data (+16 bits tag)	25.6 GBps [1.6 GHz, clocked at half the proc speed]	Shortest-path; tree-based RR arb. (centralized); pipelined circuit switch	8 layers; credit-based flow control; no VCs
Sun UltraSPARC T1 processor	2005	Up to 13 ports [8 PE cores + 4 L2 banks + 1 shared I/O]	Crossbar	128 b both for the 8 cores and the 4 L2 banks	19.2 GBps [1.2 GHz, clocked at proc speed]	Shortest-path; age-based arbitration; VCT switched	9 layers; handshaking; no VCs

Examples of Interconnection Networks

Cell Broadband Engine Element Interconnect Bus

- Cell BE is successor to PlayStation 2's Emotion Engine
  - 300 MHz MIPS-based
  - Uses two vector elements
  - 6.2 GFLOPS (Single Precision)
  - 72KB Cache + 16KB Scratch Pad RAM
  - 240mm^2 on 0.25-micron process
- PlayStation 3 uses the Cell BE*
  - 3.2 GHz POWER-based
  - Eight SIMD (Vector) Processor Elements
  - >200 GFLOPS (Single Precision)
  - 544KB cache + 2MB Local Store RAM
  - 235mm^2 on 90-nanometer SOI process

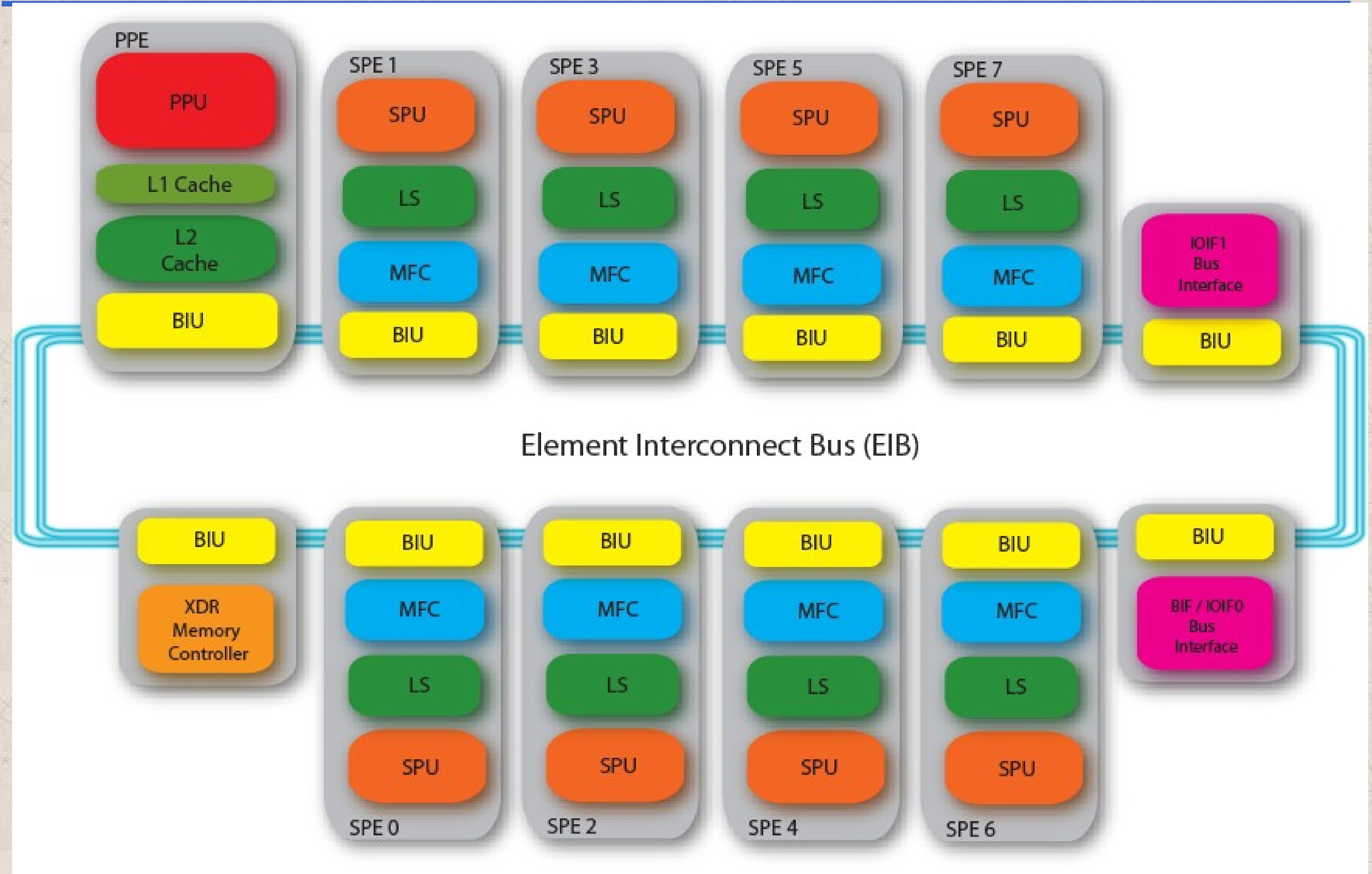
*Sony has decided to use only 7 SPEs for the PlayStation 3 to improve yield. Eight SPEs will be assumed for the purposes of this discussion.

Examples of Interconnection Networks

Cell Broadband Engine Element Interconnect Bus

- Cell Broadband Engine (Cell BE): 200 GFLOPS
  - 12 Elements (devices) interconnected by EIB:
    - *One* 64-bit Power processor element (PPE) with aggregate bandwidth of 51.2 GB/s
    - *Eight* 128-bit SIMD synergistic processor elements (SPE) with local store, each with a bandwidth of 51.2 GB/s
    - *One* memory interface controller (MIC) element with memory bandwidth of 25.6 GB/s
    - *Two* configurable I/O interface elements: 35 GB/s (out) and 25GB/s (in) of I/O bandwidth
  - Element Interconnect Bus (EIB):
    - *Four* unidirectional rings (*two in each direction*) each connect the heterogeneous 12 elements (end node devices)
    - Data links: 128 bits wide @ 1.6 GHz; data bandwidth: 25.6 GB/s
    - Provides coherent and non-coherent data transfer
    - Should optimize network traffic flow (throughput) and utilization while minimizing network latency and overhead

Examples of Interconnection Networks

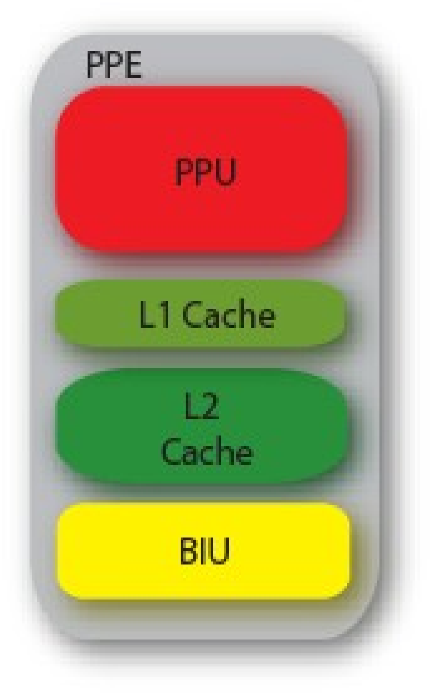


Examples of Interconnection Networks

Cell Broadband Engine Element Interconnect Bus

POWER Processing Element (PPE)

- In-order processor @ 3.2 GHz
  - Quad instruction fetch, Dual instruction issue
  - Limited out-of-order loads
- Dual-thread support
- 32KB L1 cache and 512KB L2 cache
- 128B cache lines
- Coherent transfers to/from system memory
- Instruction Set Architecture (ISA)
  - Supports 64-bit POWER architecture
  - VMX SIMD instructions are also supported



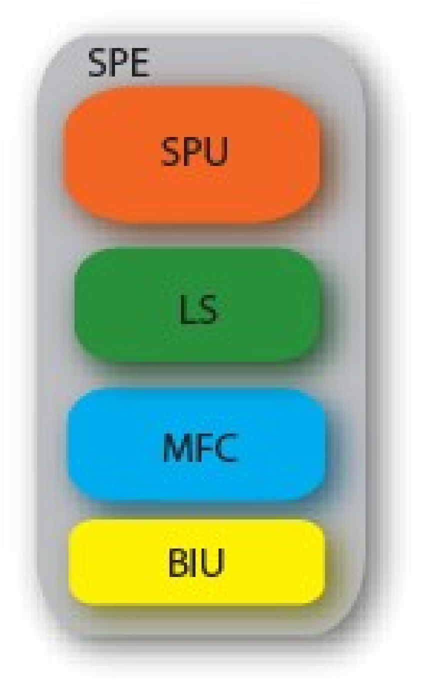
EIB Challenge: Must effectively service PPE and SPE nodes

Examples of Interconnection Networks

Cell Broadband Engine Element Interconnect Bus

Synergistic Processing Element

- 128-bit SIMD processor @ 3.2GHz
  - Dual-issue, in-order
  - 128 entry register file
  - Unique (VMX-like) ISA
- 256KB Local Store
- 128B memory blocks
- Non-coherent transfers from SPE to SPE
- Contains a *Memory Flow Controller (MFC)*
  - DMA engine
  - Memory Management Unit (MMU)
  - Atomic units for synchronization

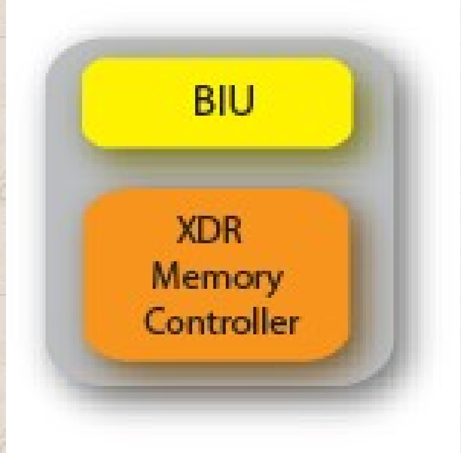


EIB Challenge: Limited LS in SPEs increases bandwidth requirements

Examples of Interconnection Networks

Cell Broadband Engine Element Interconnect Bus

Memory Interface Controller (MIC)



- RAMBUS XDR Interface with two XIO channels each operating at 400MHz with an Octal Data Rate (effective 3.2GHz)
- Supports up to 512MB of XDR RAM
- Coherent transfers
- 25.6GB/s Peak Memory Bandwidth
  - 1GB/s is lost to overhead
  - Interweaved read/write streams reduces bandwidth to 21 GB/s

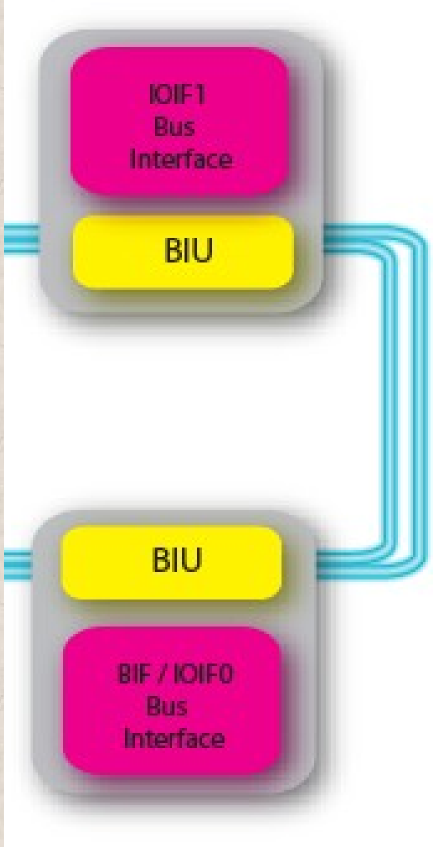
EIB Challenge: One node can saturate the MIC

Examples of Interconnection Networks

Cell Broadband Engine Element Interconnect Bus

FlexIO Bus Interface

- Operates @ 5GHz
- Twelve 1-byte lanes
  - 7 outgoing (35 GB/s)
  - 5 incoming (25 GB/s)
- Can be used to connect to another Cell BE chip through coherent I/O interface (BIF)
  - This operation is likely to cause much more contention on the EIB
  - Shared command bus
    - 2nd Cell BE becomes a slave to the first

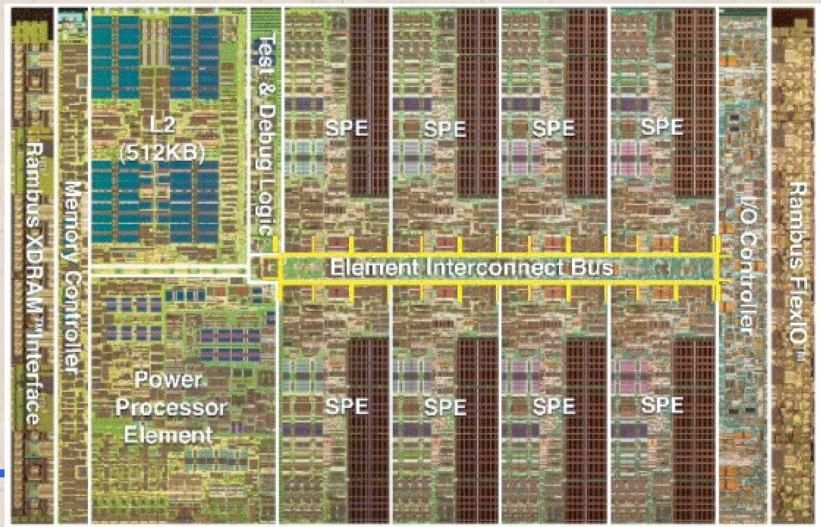


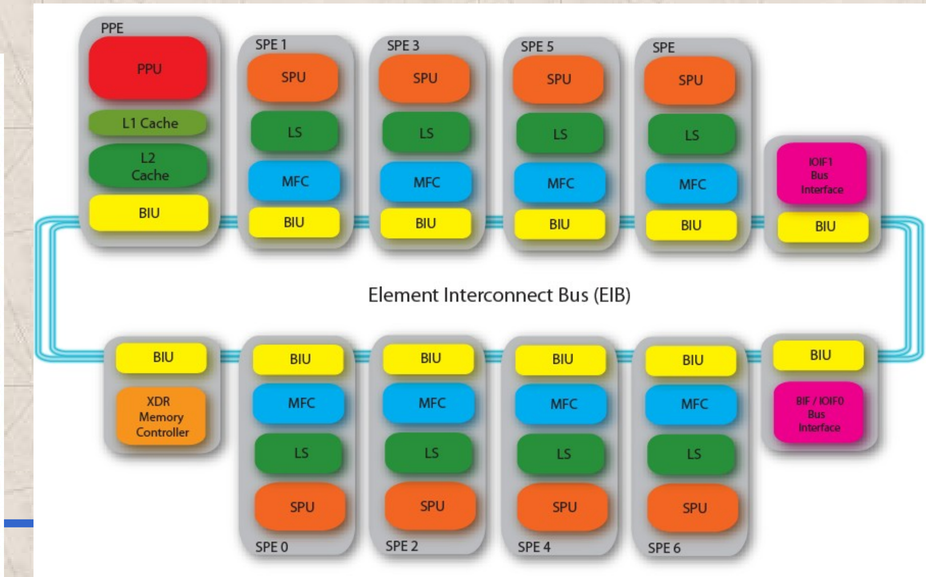
EIB Challenge: Supporting high throughput I/O

Examples of Interconnection Networks

Cell Broadband Engine Element Interconnect Bus

- *Element Interconnect Bus* (*EIB*)
  - Packet size: 16B – 128B (no headers); pipelined circuit switching
  - Credit-based flow control (command bus central token manager)
  - Two-stage, dual round-robin centralized network arbiter
  - Allows up to 64 outstanding requests (DMA)
    - 64 Request Buffers in the MIC; 16 Request Buffers per SPE
  - Latency: 1 cycle/hop, transmission time (largest packet) 8 cycles
  - Effective bandwidth: peak 307.2 GB/s, max. sustainable 204.8 GB/s





Examples of Interconnection Networks

Cell Broadband Engine Element Interconnect Bus

- 1.6 GHz bus clock rate (half the processor core frequency)
- Two-stage, round-robin command & data arbitration
  - Shared Command Bus:
    - 8-byte commands, tree-topology, fully pipelined
  - Dedicated-link Data Arbitration Network:
    - Central data arbiter controls the four 16 Byte wide data rings
    - 6 hop packet transfer limit over the rings
    - MIC has highest priority; other nodes have same default priority
- Resource Allocation Manager allows dynamic software management of EIB resources
  - An optional feature that allocates usage of resources to prevent a single node from being overwhelmed
  - Divides memory and IO resources into banks
  - Separates requestors into groups
  - Tokens are used to grant access to each requestor group

Examples of Interconnection Networks

Cell Broadband Engine Element Interconnect Bus

Characterizing the Network latency

- Four phases make up the end-to-end latency:

$$\textit{Latency} = \underbrace{\textit{Sending Phase} + \textit{Command Phase}}_{\textit{Sending Overhead}} + \underbrace{\textit{Data Phase}}_{\textit{Transport latency}} + \underbrace{\textit{Receiving Phase}}_{\textit{Receiving latency}}$$

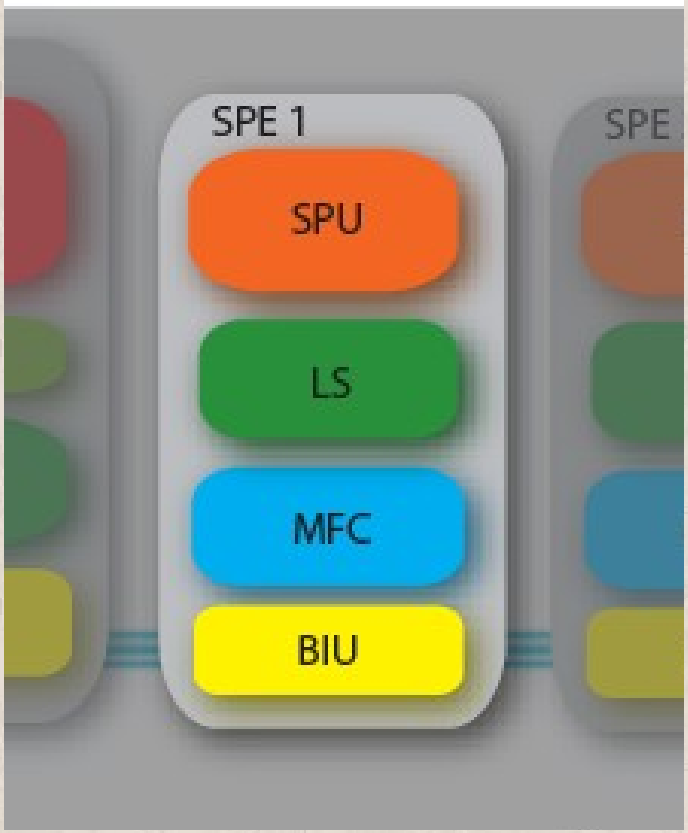
- EIB is arbitrated and contains transaction queues
  - For zero load: no waiting for other transactions by the arbiters
  - For zero load: all queues are empty
- Two transfer modes across the network
  - Differ in the size of transfers for each type of transaction
    1. Direct Memory Access (DMA): 128B packets, 16KB max transfer
       - » DMA list function allows for 2,048 consecutive DMAs
    2. Memory Mapped IO (MMIO): 4B – 8B transfers each

Examples of Interconnection Networks

Cell Broadband Engine Element Interconnect Bus

Sending Phase

- Responsible for EIB transaction initiation
- Includes all processor and DMA controller activities prior to transactions being sent to the EIB
- Required only once for each DMA transfer (even if multiple packets)
  - Latency can be amortized over multi-packet DMA transfers

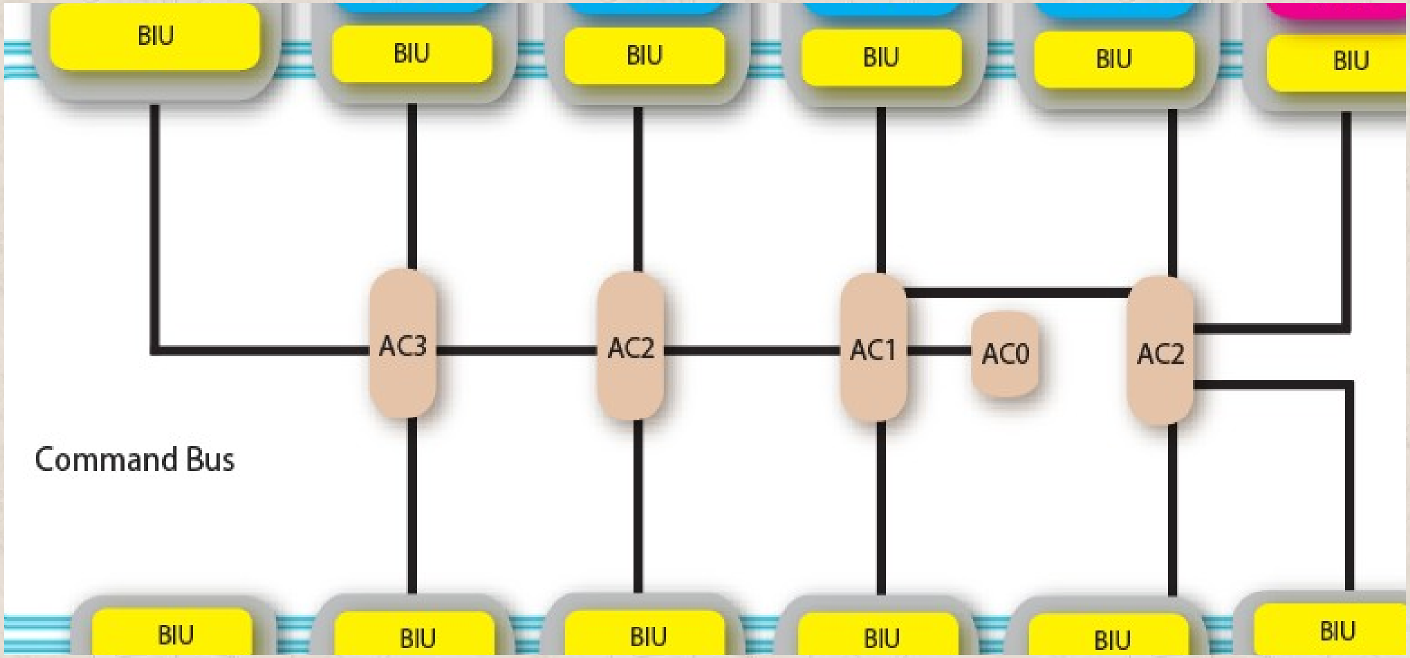


Examples of Interconnection Networks

Cell Broadband Engine Element Interconnect Bus

Command Phase

- Coordinates 128B transfers across the EIB (required for each packet)
  - Informs read/write target element of the impending transaction to allow for element to set up transaction (e.g., data fetch or buffer reservation)
  - Performs coherency checking across all elements, if necessary
- Handles inter-element (end-to-end) command communication
  - Start SPU execution, send SPU signals, synchronization



Examples of Interconnection Networks

Cell Broadband Engine Element Interconnect Bus

Command Phase (continued) – 5 Steps

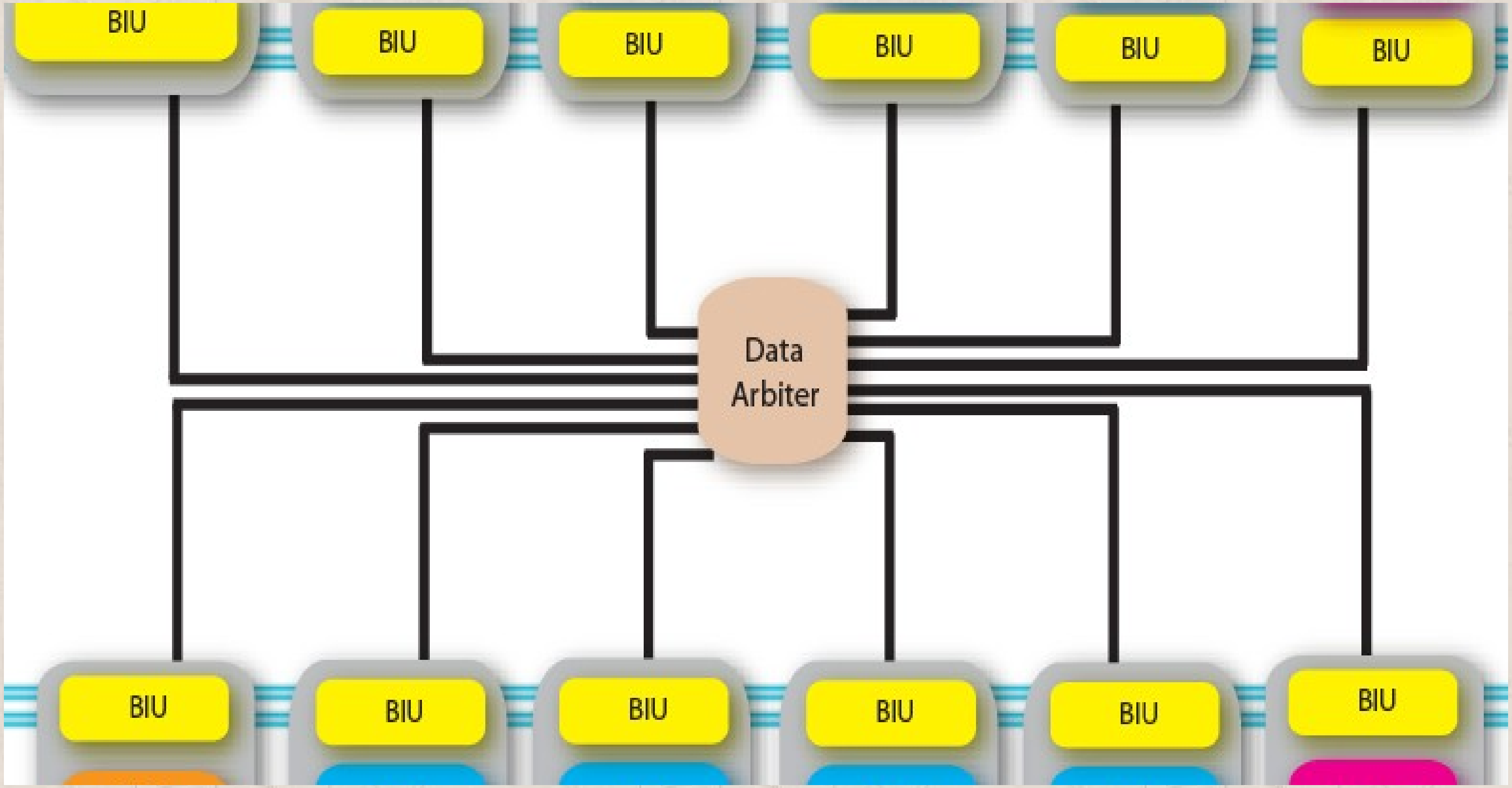
2. *Command Issue*
   - Element initiates a transaction on the command bus by issuing a command token
3. *Command Reflection*
   - Element command is presented to all other bus elements
4. *Snoop Response*
   - Elements respond to reflected command w/ a snoop response to root address concentrator (AC0)
5. *Combined Snoop Response*
   - Command bus distributes the combined result of all element responses back to all elements
6. *Final Snoop Response*
   - Command phase concludes at the initiating node, allowing the command initiator to begin the data phase

Examples of Interconnection Networks

Cell Broadband Engine Element Interconnect Bus

Data Phase

- Data ring arbitration and data transport on a per 128B packet basis
- Waits for free data ring segment (to dest) before access is granted
  - Assumes elements are ready based on successful command phase

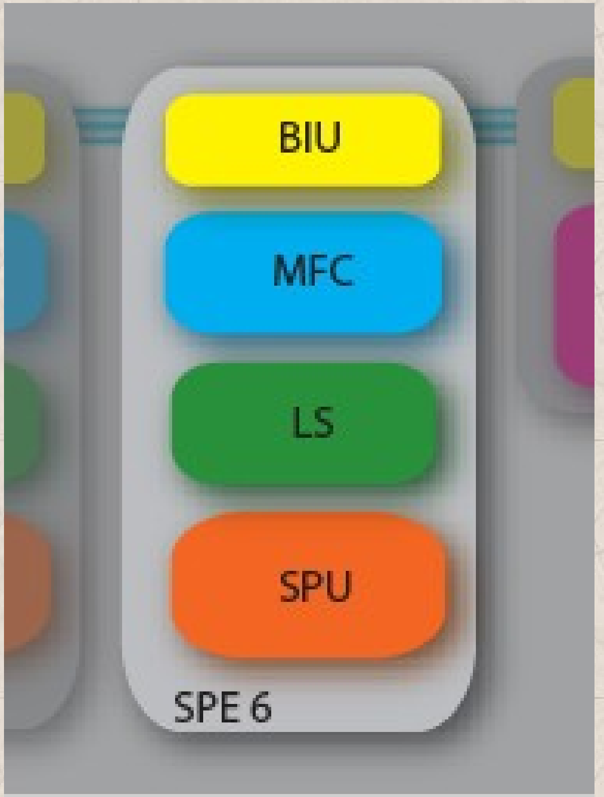


Examples of Interconnection Networks

Cell Broadband Engine Element Interconnect Bus

Receiving Phase

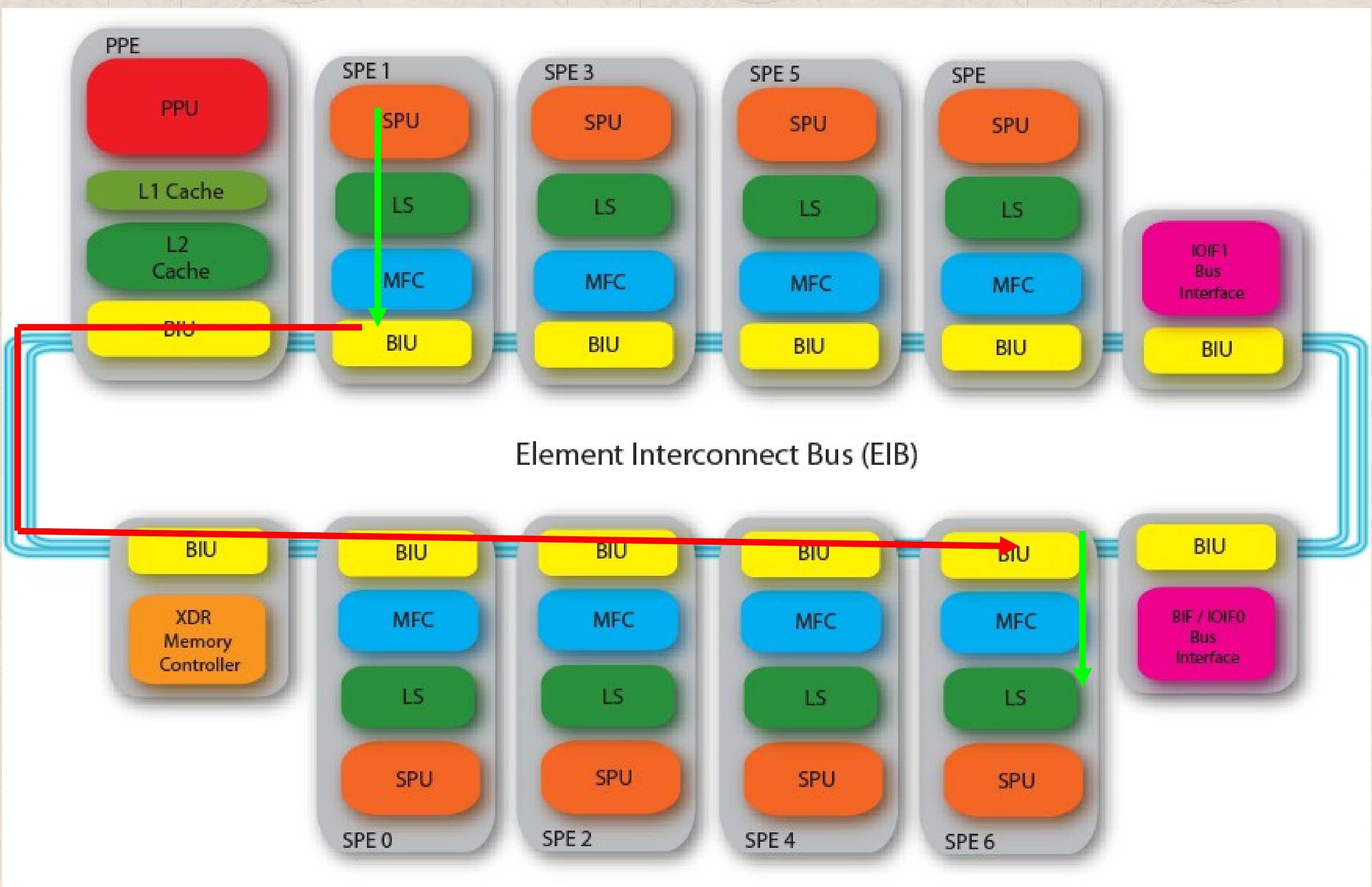
- Received data is directed to its final location
  - Local Store, Memory, or I/O
- For typical data transfers, no target “receive” processing required
  - BIU/MFC controls final data movement
  - Done for each 128B packet transfer



Examples of Interconnection Networks

Cell Broadband Engine Element Interconnect Bus

Network Latency: Non-coherent DMA transaction from SPE1 to SPE6
(Max length across a ring = 6 hops)

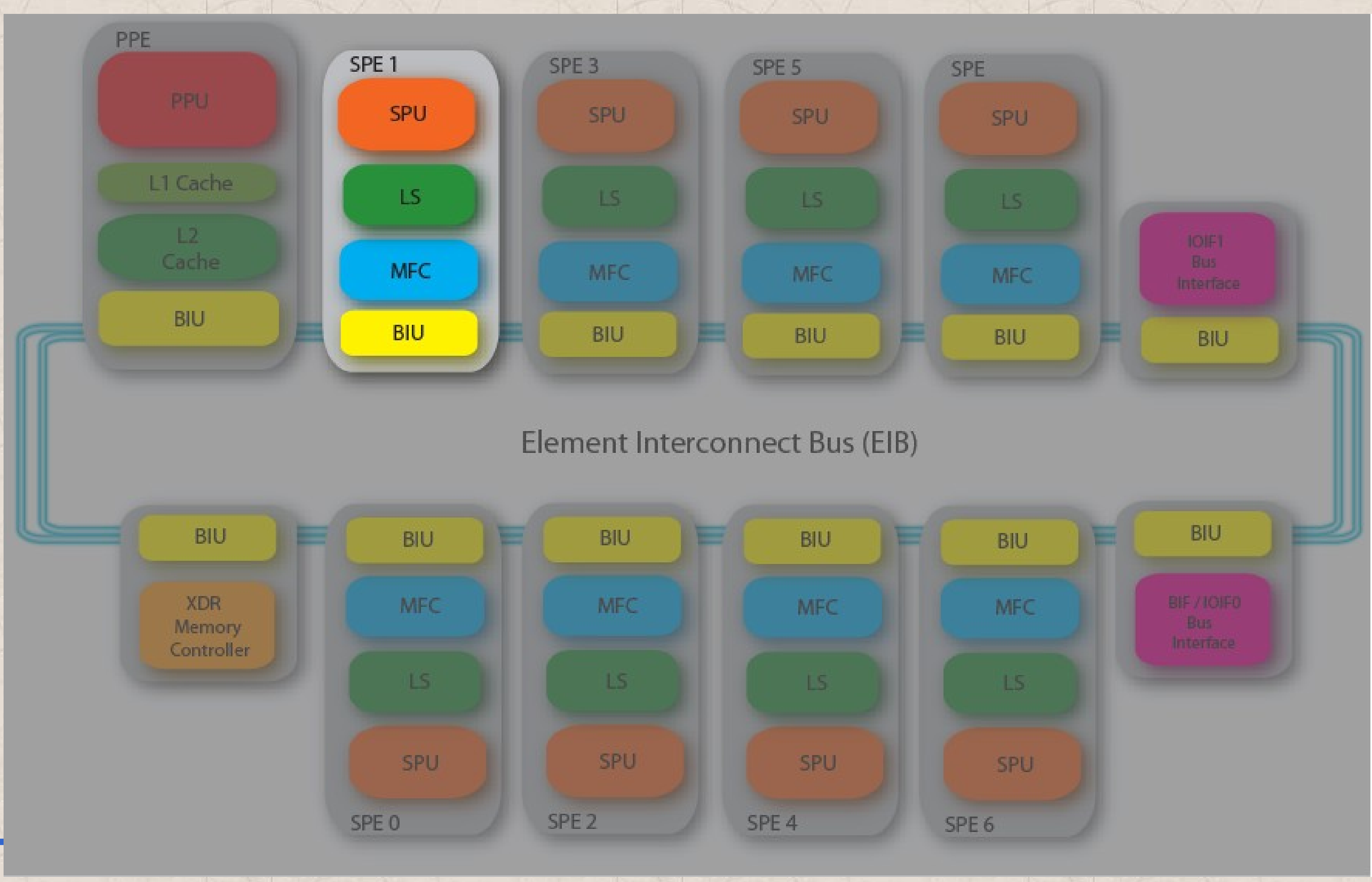


Interconnection Networks: © Timothy Mark Pinkston and José Duato
...with major presentation contribution from José Flich

Examples of Interconnection Networks

Cell Broadband Engine Element Interconnect Bus

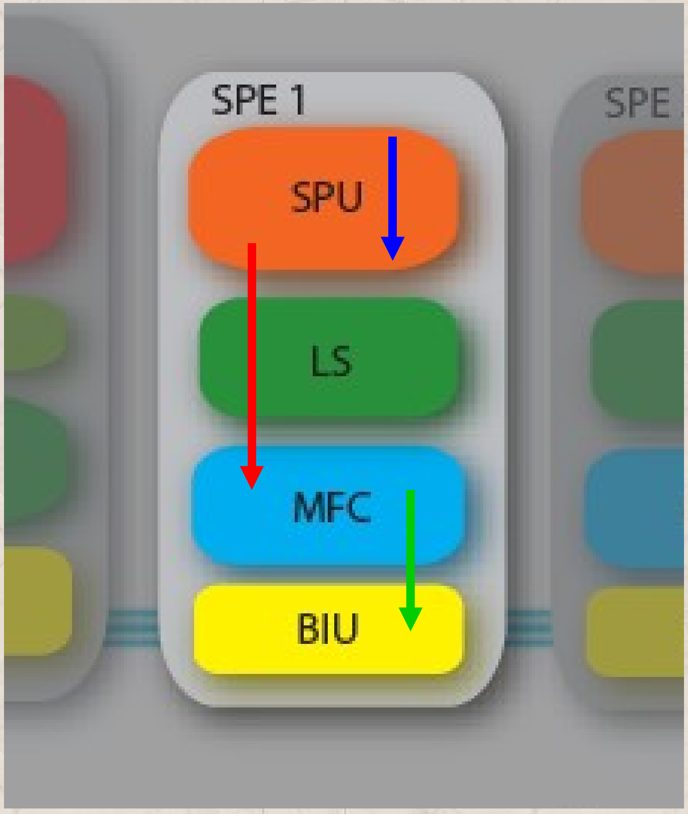
- *Network Latency:* Non-coherent DMA transaction from SPE1 to SPE6



Examples of Interconnection Networks

Cell BE EIB

Network Latency ***Sending Phase***



Pipeline Latency

= 23 CPU Clock Cycles

DMA Issue = 10 CPU Clocks

1. Write SPE local store address
2. Write effective address high
3. Write effective address low
4. Write DMA size
5. Write DMA command

DMA Controller Processing

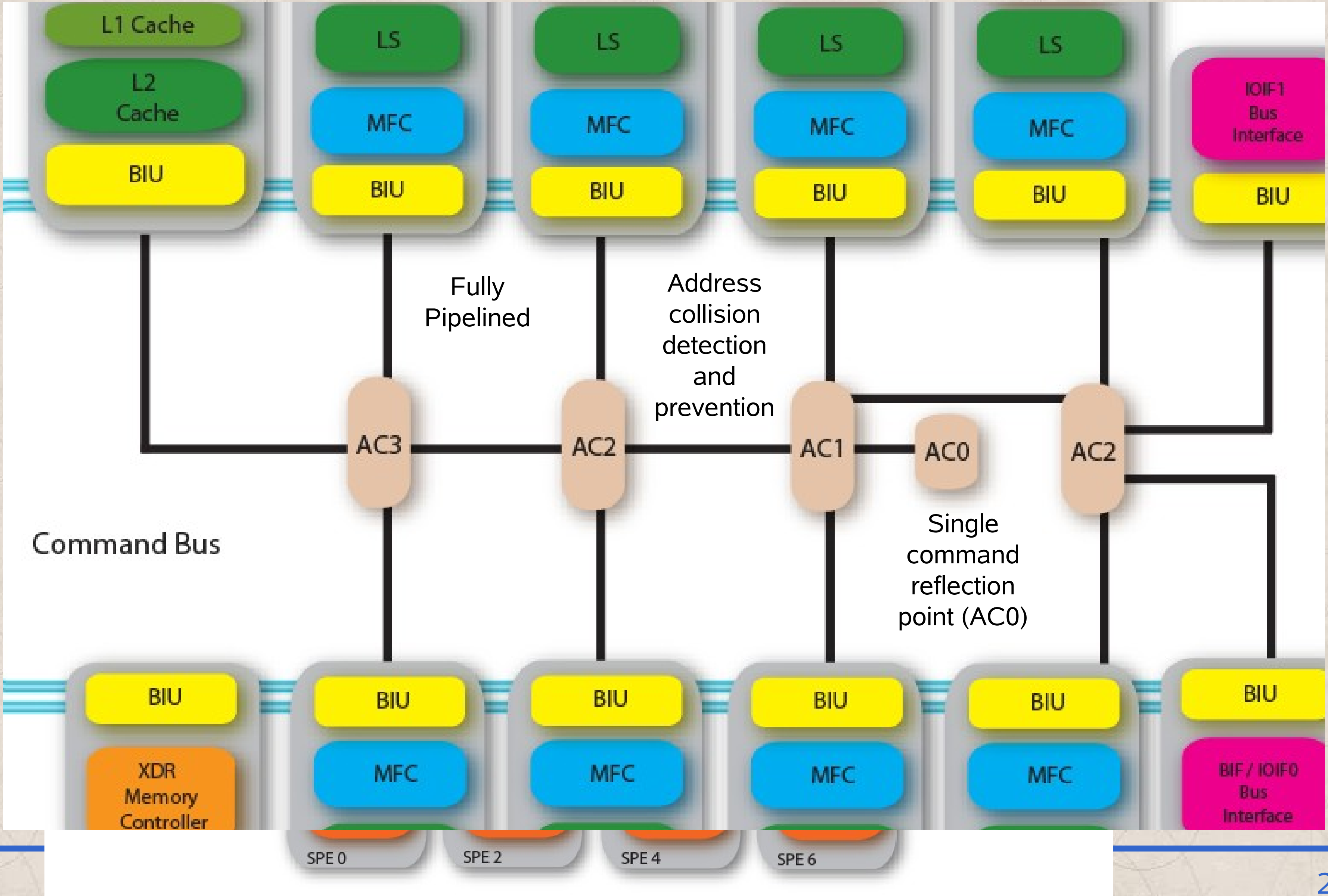
= 20 CPU Clock Cycles

Sending Phase (SP) = 53 CPU cycles = 26.5 Bus Cycles

Examples of Interconnection Networks

Cell BE EIB

Network Latency ***Command Phase***



Fully Pipelined

Address collision detection and prevention

Single command reflection point (AC0)

Examples of Interconnection Networks

Cell BE EIB

Network Latency ***Command Phase***

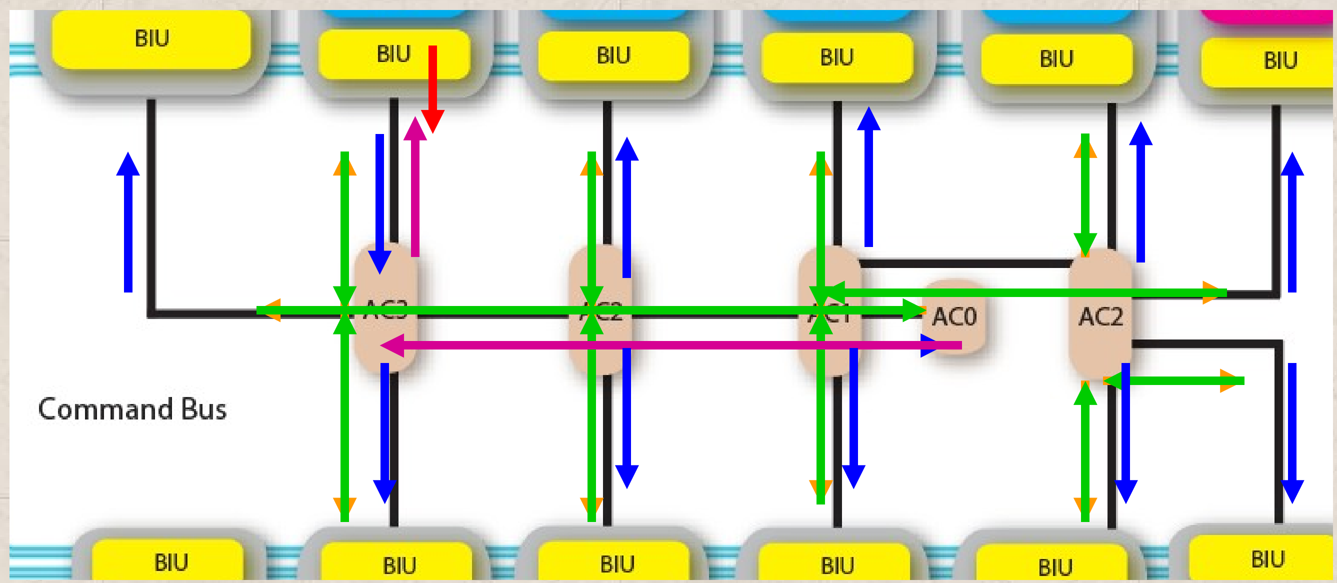
Command Issue = 3 Bus Cycles

Command Reflection (3) + AC3 Reflection (2) + AC2 Reflection (2) = 7 Bus Cycles

Snoop Response = 13 Bus Cycles

Combined Snoop Response = 5 Bus Cycles

Final Snoop Response = 3 Bus Cycles

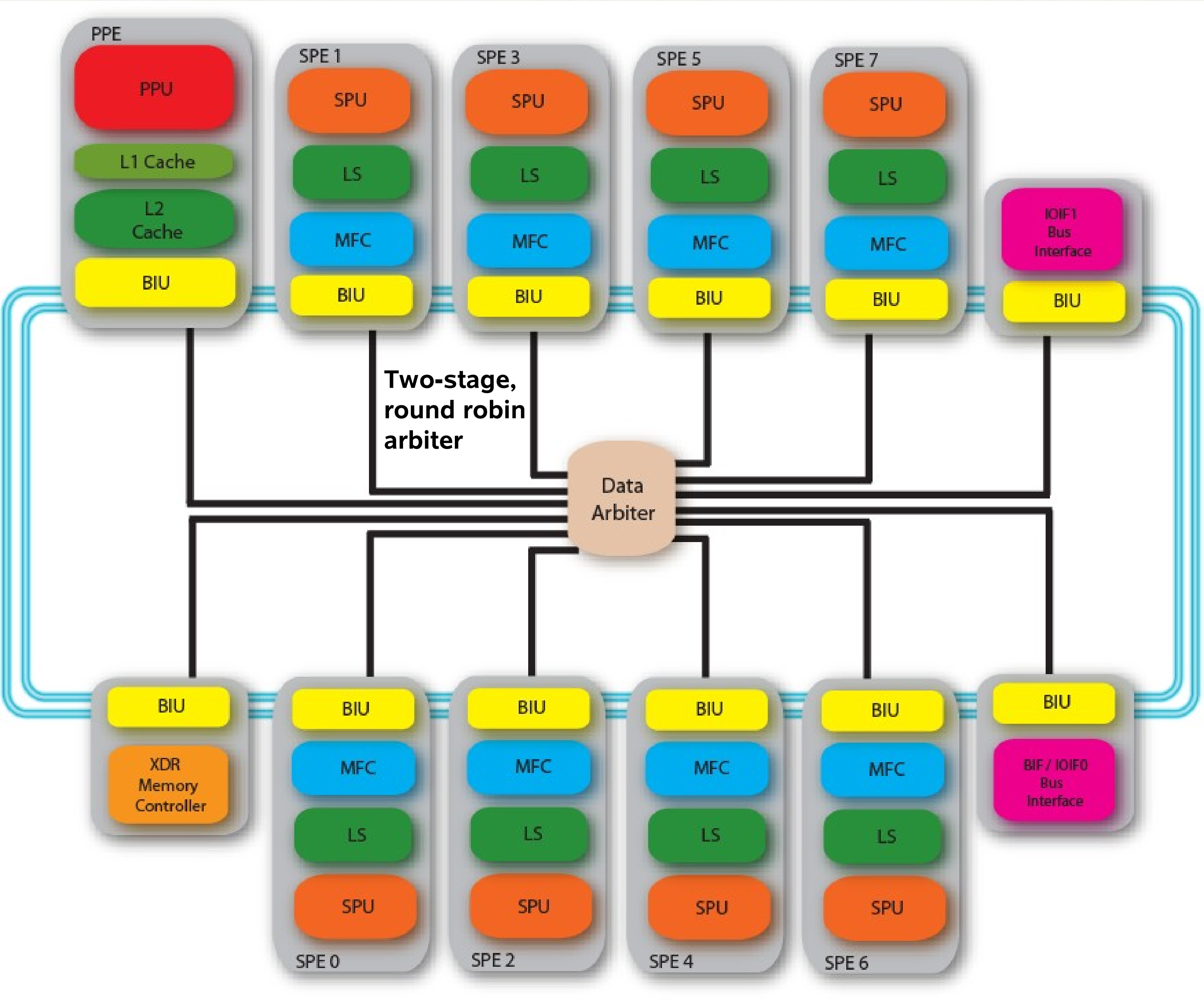


Command Phase (CP) = 31 Bus Cycles

Examples of Interconnection Networks

Cell BE EIB

Network Latency ***Data Phase***



Two-stage, round robin arbiter

Arbitrate for *four* 16B-wide data rings:

- **Two rings clockwise**
- **Two rings counter-clockwise**

Examples of Interconnection Networks

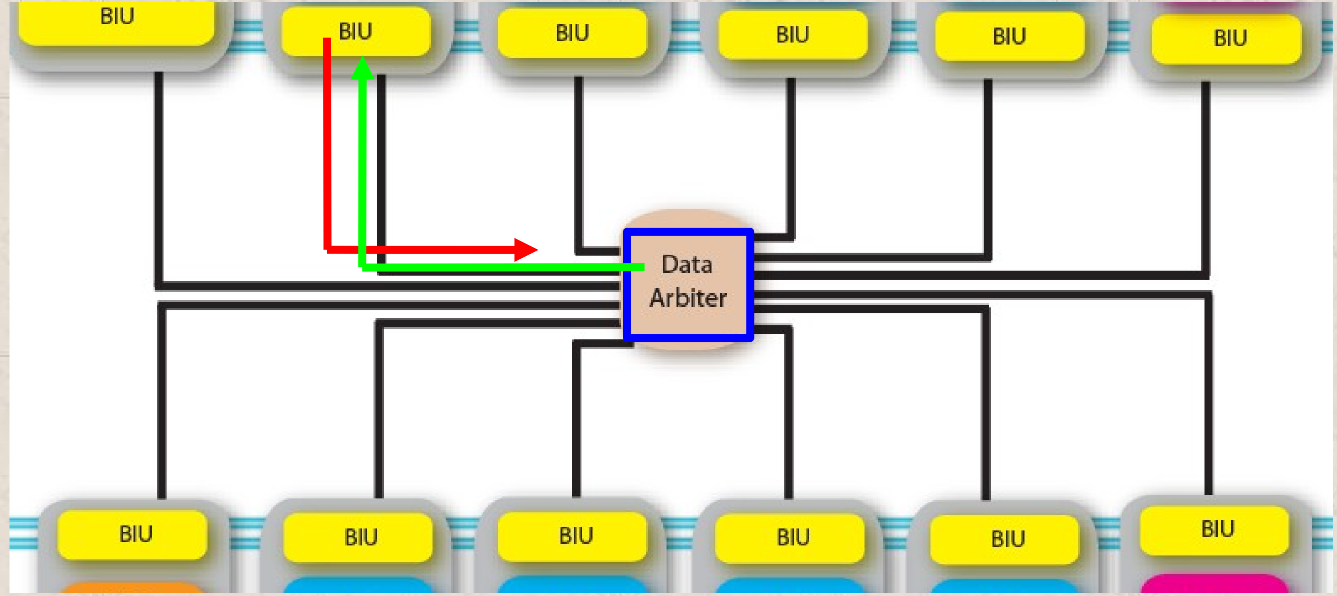
Cell BE EIB

Network Latency ***Data Phase***

Data Request to Arbiter = 2 Bus Cycles

Data Arbitration = 2 Bus Cycles

Data Bus Grant = 2 Bus Cycles



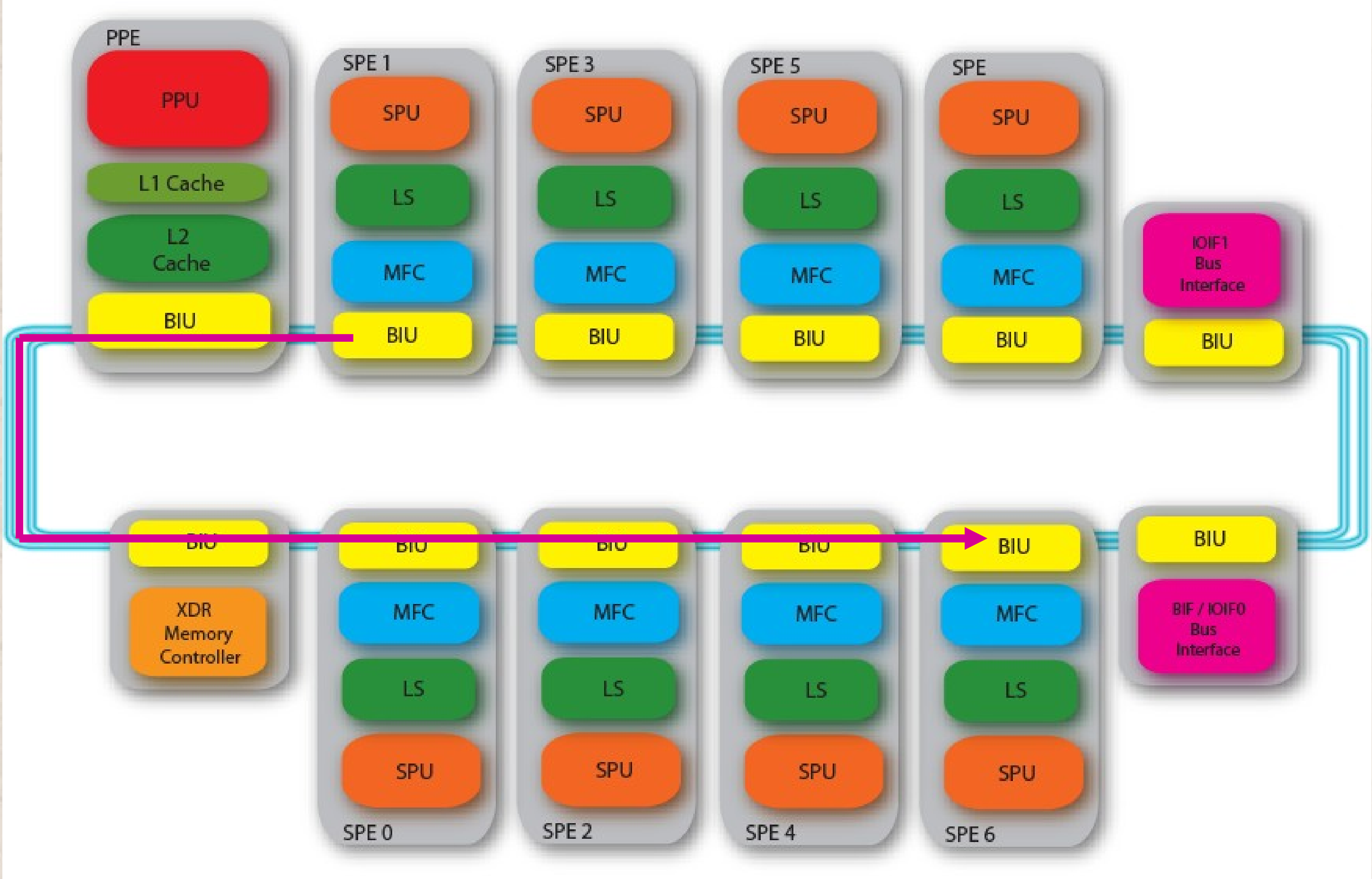
Examples of Interconnection Networks

Cell BE EIB

Network Latency ***Data Phase***

Data propagation delay = 6 Bus Cycles (1 cycle per hop)

Transmission time = 8 Bus Cycles (128B packets)

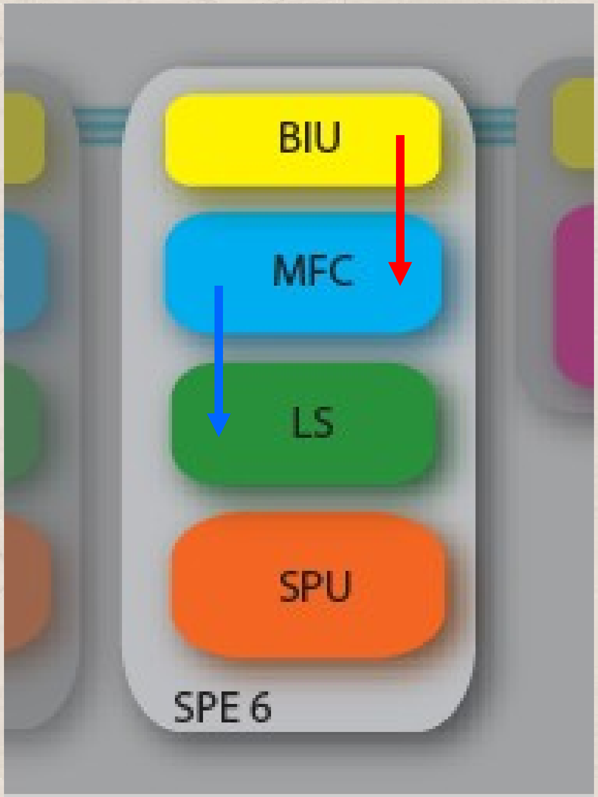


Data Phase (DP) = 20 Bus Cycles

Examples of Interconnection Networks

Cell BE EIB

Network Latency ***Receiving Phase***



Receiving Overhead = 2 Bus Cycles

- Move data from BIU to MFC = 1 Bus Cycle
- Move data from MFC to Local Store = 1 Bus Cycle

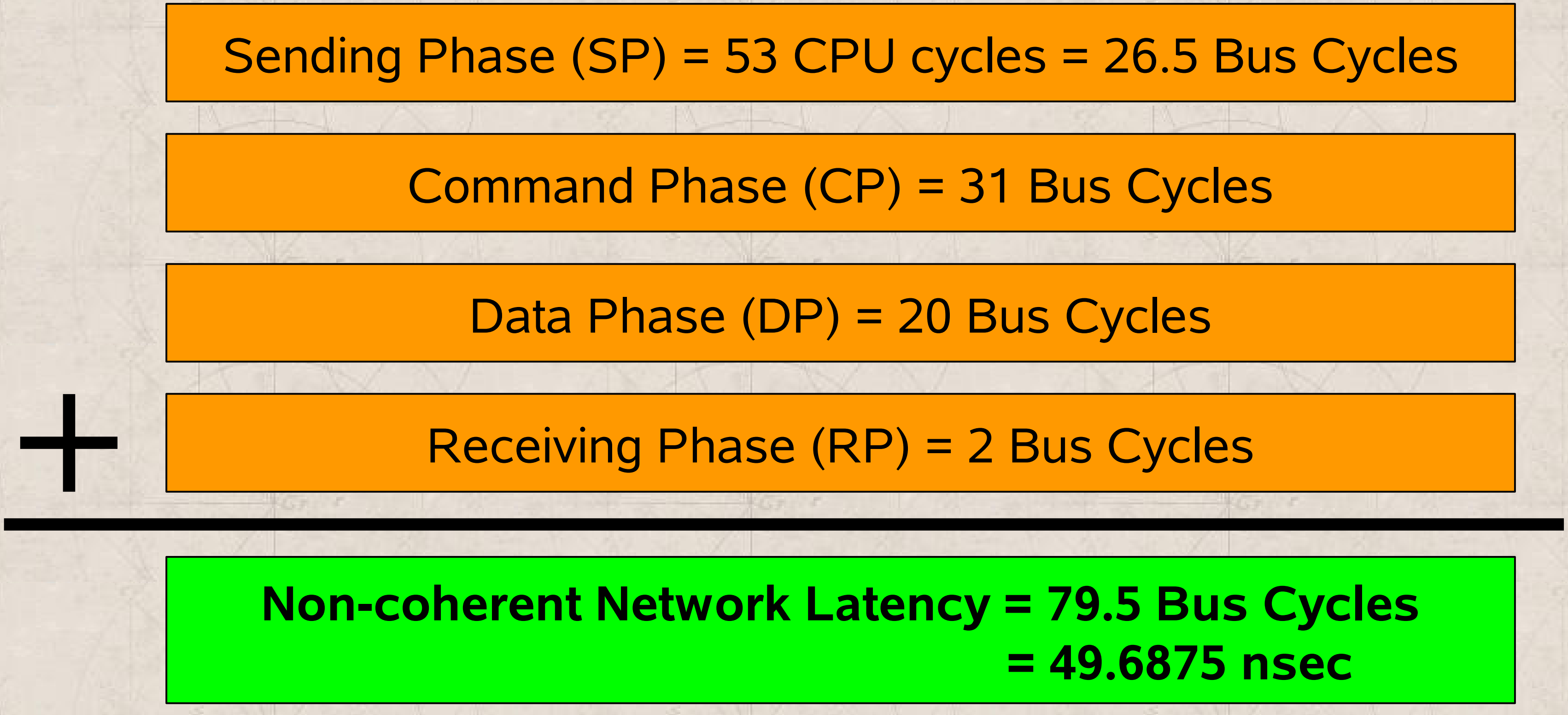
Receiving Phase (RP) = 2 Bus Cycles

Examples of Interconnection Networks

Cell BE EIB

Network Latency (*all phases*)

- For *non-coherent* DMA transfer from SPE1 to SPE6



Sending Phase (SP) = 53 CPU cycles = 26.5 Bus Cycles

Command Phase (CP) = 31 Bus Cycles

Data Phase (DP) = 20 Bus Cycles

+ Receiving Phase (RP) = 2 Bus Cycles

Non-coherent Network Latency = 79.5 Bus Cycles
= 49.6875 nsec

Examples of Interconnection Networks

Cell BE EIB

Network Latency (*all phases*)

- For *coherent* DMA transfer from SPE1 to SPE6
  - Command phase changes for coherent commands
    - Command Issue = 11 Bus Cycles
    - Combined Snoop Response = 9 Bus Cycles

Sending Phase (SP) = 53 CPU cycles = 26.5 Bus Cycles

Coherent Command Phase (CCP) = 43 Bus Cycles

Data Phase (DP) = 20 Bus Cycles

+ Receiving Phase (RP) = 2 Bus Cycles

Coherent Network Latency = 91.5 Bus Cycles
= 57.1875 nsec

Examples of Interconnection Networks

Cell BE EIB *Effective Bandwidth*

DMA vs. MMIO

- Latency for MMIO may be slightly shorter due to lack of DMA setup, but overall network latency will essentially be the same and always be coherent
- MMIO is significantly less efficient than DMA
  - DMA transfers use all 128B for each EIB data packet
  - MMIO transfers only 4 – 8 bytes per packet
    - Wastes 94% to 97% of the bandwidth provided to data packets

Link pipelining

- Allows pipelining and some overlapping of various components of latency from consecutive transactions: sending, command, data, and receiving phases
  - increases resource utilization and effective bandwidth
- However, *command phase* “stage” limits concurrency < 100%

Examples of Interconnection Networks

Cell BE EIB

Effective Bandwidth

$$BW_{LinkInjection} = \frac{\textit{Packet size}}{\textit{max (sending overhead, transmission time)}}$$

$$BW_{LinkReception} = \frac{\textit{Packet size}}{\textit{max (receiving overhead, transmission time)}}$$

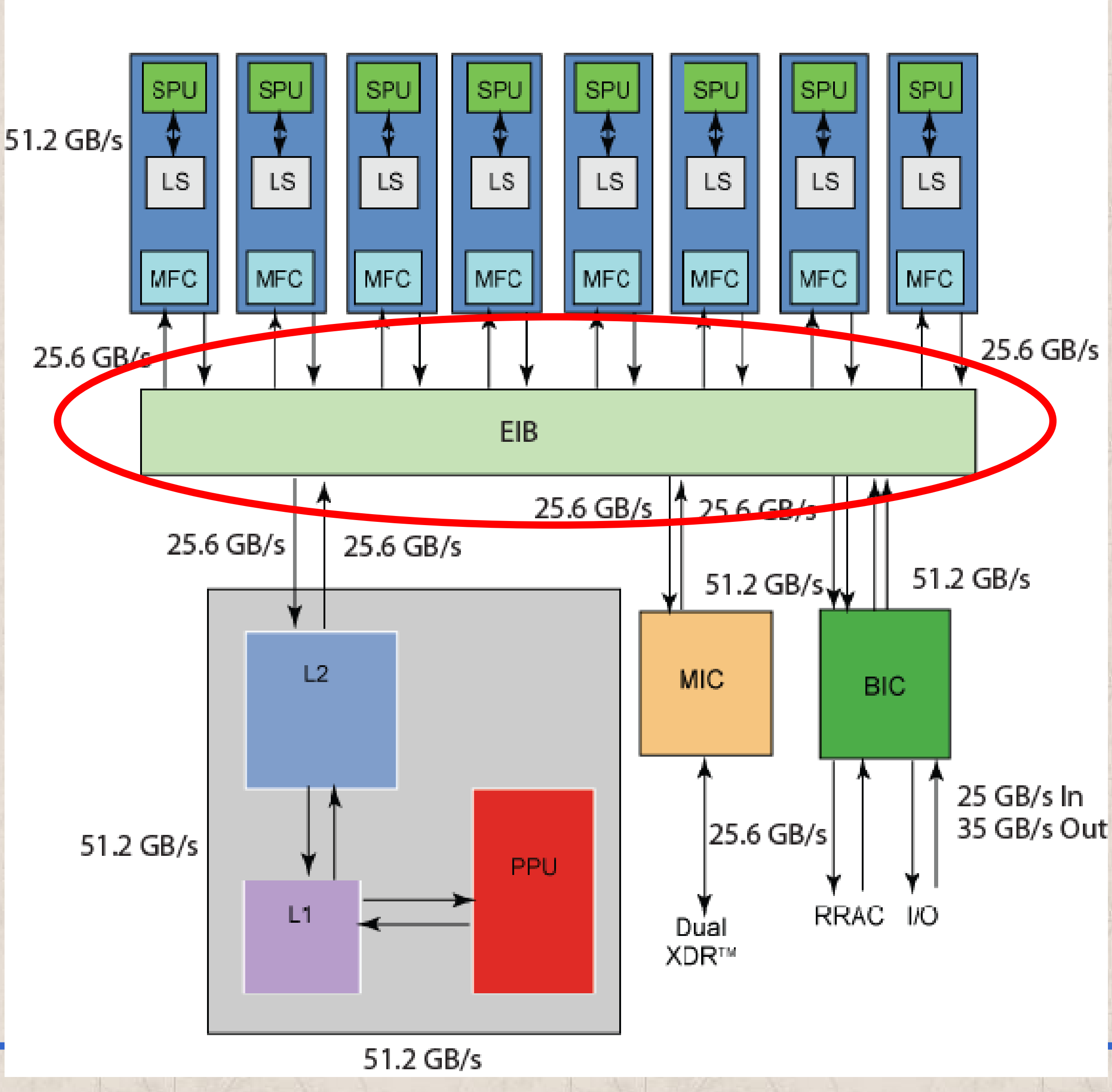
- *If bandwidth were limited by packet transmission* (with link pipelining):
  - Packet size = 128B
  - Transmission time (TT) = 128B/16B = 8 cycles
  - Bandwidth = Packet size / max(~~overhead~~, TT) = 16B/cycle
- Link Injection Bandwidth ($BW_{LinkInjection}$) = 16B/cycle = 25.6GB/s
- Link Reception Bandwidth ($BW_{LinkReception}$) = 16B/cycle = 25.6GB/s
- Network Injection Bandwidth ($BW_{NetworkInjection}$) = 12 x 25.6 = 307.2GB/s
- Network Reception Bandwidth ($BW_{NetworkReception}$) = 307.2GB/s

***Transmission-limited* Network Inj. (Rec.) Bandwidth = 307.2 GB/s**

Examples of Interconnection Networks

Cell BE EIB

Effective Bandwidth



EIB must enable 307.2 GB/s aggregate $BW_{Network}$ *in order NOT to be bottleneck*

Examples of Interconnection Networks

Cell BE EIB

Effective Bandwidth

- Network Injection (Reception) Bandwidth = **307.2 GB/s**
  - 12 elements each with one injection (reception) link
  - 16B per bus cycle per injection (reception) link
  - 16B x 1.6 GHz x 12 elements = 307.2 GB/s network injection (rec.)

- EIB Network Bandwidth (*$BW_{Network}$*)
  - Unidirectional ring data width = 16B
  - Each transfer takes 8 cycles (128B)
  - Each ring can start 1 operation every three cycles
  - 4 rings, two in each direction
  - 3 concurrent transfers (maximum) per ring

B/s

Maximum Network Bandwidth = 307.2 GB/s

But Effective Bandwidth is much less!!

Examples of Interconnection Networks

Cell BE EIB *Effective Bandwidth*

- **Command Phase Limitations (Non-coherent Transfers)**
  - ***Max. effective bandwidth = 204.8GB/s*** (non-coherent transfers)
    - › Command bus is limited to *1 request per bus cycle*
    - › Each request can transfer 128B

Bus Cycle	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Ring A Data 1	1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8
Ring A Data 2	▨	▨	▨		1	2	3	4	5	6	7	8	1	2	3	4
Ring A Data 3					▨	▨	▨									
Ring B Data 1		1	2	3	4	5	6	7	8	1	2	3	4	5	6	7
Ring B Data 2		▨	▨	▨		1	2	3	4	5	6	7	8	1	2	3
Ring B Data 3						▨	▨	▨								
Ring C Data 1			1	2	3	4	5	6	7	8	1	2	3	4	5	6
Ring C Data 2			▨	▨	▨		1	2	3	4	5	6	7	8	1	2
Ring C Data 3							▨	▨	▨							
Ring D Data 1				1	2	3	4	5	6	7	8	1	2	3	4	5
Ring D Data 2				▨	▨	▨		1	2	3	4	5	6	7	8	1
Ring D Data 3								▨	▨	▨						

- Command bus allows the issuing of up to one transaction per cycle

- Each ring can issue one new transaction every 3 cycles (grey cycles indicate a ring is unavailable)

- Command bus does not allow more than 8 concurrent transactions at any given time

Network Effective Bandwidth (non-coherent) = 204.8 GB/s

Examples of Interconnection Networks

Cell BE EIB *Effective Bandwidth*

- **Command Phase Limitations (Non-coherent Transfers)**
  - ***Max. effective bandwidth = 204.8GB/s*** (non-coherent transfers)
    - › Command bus is limited to *1 request per bus cycle*
    - › Each request can transfer 128B

Bus Cycle	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Ring A Data 1	1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8
Ring A Data 2	▨	▨	▨	1	2	3	4	5	6	7	8	1	2	3	4	5
Ring A Data 3				▨	▨	▨	1	2	3	4	5	6	7	8	1	2
Ring B Data 1		1	2	3	4	5	6	7	8	1	2	3	4	5	6	7
Ring B Data 2		▨	▨	▨	1	2	3	4	5	6	7	8	1	2	3	4
Ring B Data 3					▨	▨	▨	1	2	3	4	5	6	7	8	1
Ring C Data 1			1	2	3	4	5	6	7	8	1	2	3	4	5	6
Ring C Data 2			▨	▨	▨	1	2	3	4	5	6	7	8	1	2	3
Ring C Data 3						▨	▨	▨	1	2	3	4	5	6	7	8
Ring D Data 1																
Ring D Data 2																
Ring D Data 3																

… or three on some buses and an empty fourth bus

Network Effective Bandwidth (non-coherent) = 204.8 GB/s

Examples of Interconnection Networks

Cell BE EIB *Effective Bandwidth*

- **Command Phase Limitations (Coherent Transfers)**
  - ***Max. effective bandwidth = 102.4 GB/s*** (coherent transfers)
    - › Command bus is limited to *1 coherent request per 2 bus cycles*
    - › Each request transfers 128B

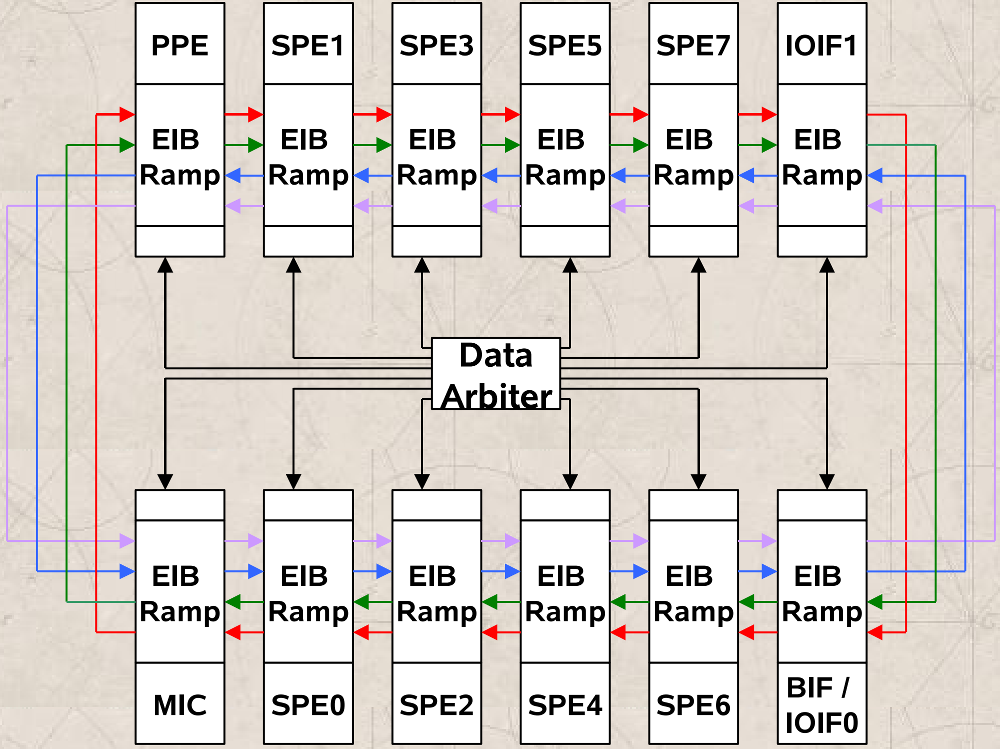
Bus Cycle	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Ring A Data 1	1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8
Ring A Data 2																
Ring A Data 3																
Ring B Data 1			1	2	3	4	5	6	7	8	1	2	3	4	5	6
Ring B Data 2																
Ring B Data 3																
Ring C Data 1					1	2	3	4	5	6	7	8	1	2	3	4
Ring C Data 2																
Ring C Data 3																
Ring D Data 1							1	2	3	4	5	6	7	8	1	2
Ring D Data 2																
Ring D Data 3																

Effective Network Bandwidth (coherent) = 102.4 GB/s

Examples of Interconnection Networks

Cell BE EIB *Effective Bandwidth*

- **Ring (Segmented Bus) Limitations**
  - ***Maximum of 3 non-overlapping transfers per ring*** (*traffic-dependent!*)
  - Non-overlap and pass-through restrictions limit the network throughput
  - PPE and MIC are located next to each other
  - Even-numbered SPEs communicate faster with other even SPEs
  - Likewise for the odd-numbered SPEs



Examples of Interconnection Networks

Cell BE EIB

Effective Bandwidth

- **Data Transfer Size Inefficiencies (< 128B)**
  - Each transfer holds allocated resources for 8 cycles
  - ***Any transfer < 128B in size results in wastage*** (under-utilization)
  - Not using DMA and full-size transfers greatly diminishes the effective bandwidth of the system
  - Fortunately, many transfers should be full sized
    - SPE memory block size = 128B
    - PPE L2 cache line size = 128B
  - MMIO will cause major performance impact
    - 4B - 8B vs. 128B

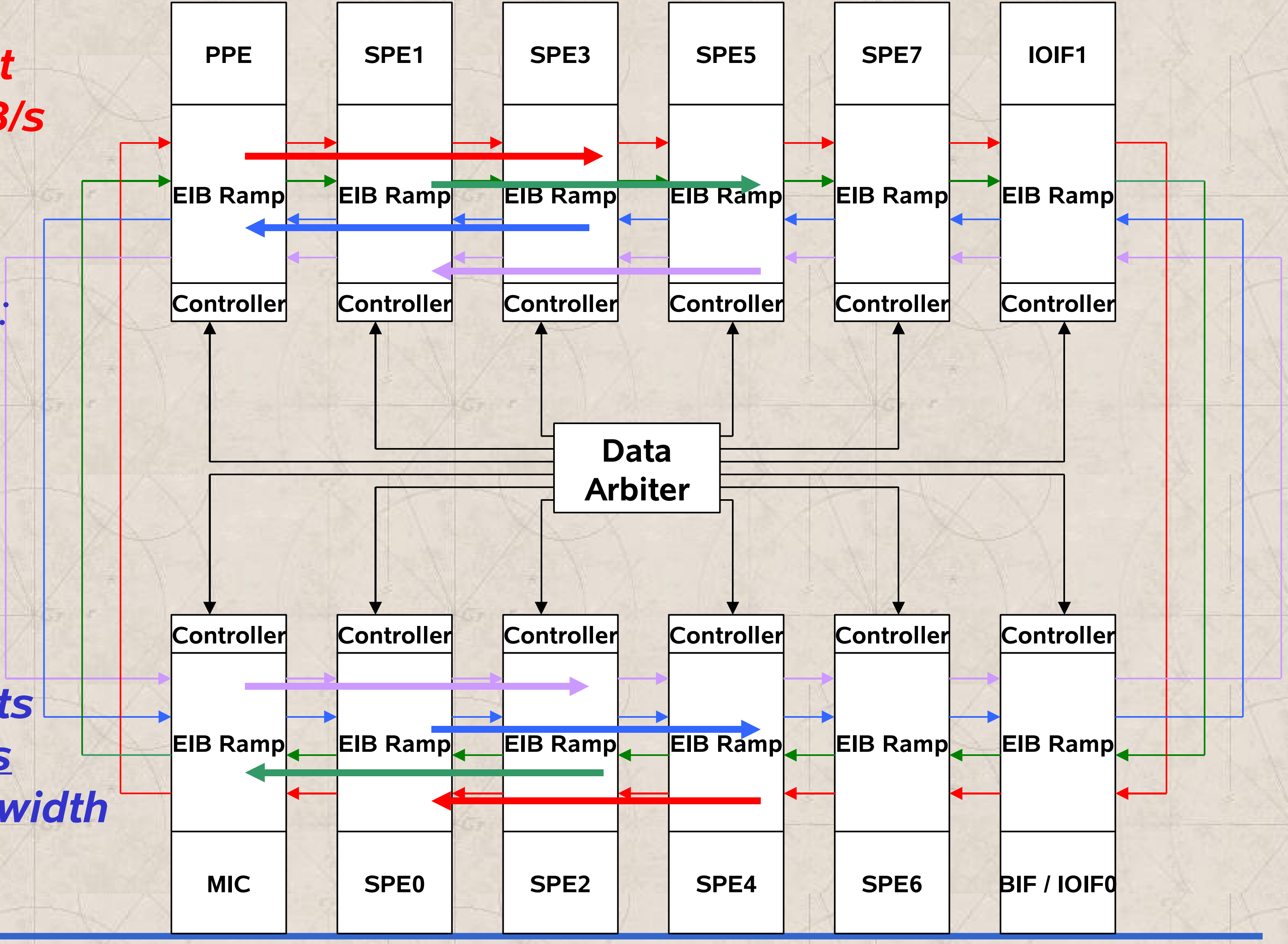
Examples of Interconnection Networks

Cell BE EIB *Measured vs. Calculated Best-case Effective Bandwidth*

- ***Non-coherent BW ≤ 204.8 GB/s***

- *Traffic pattern:* contention-free with eight concurrent transfers: _two_ per ring

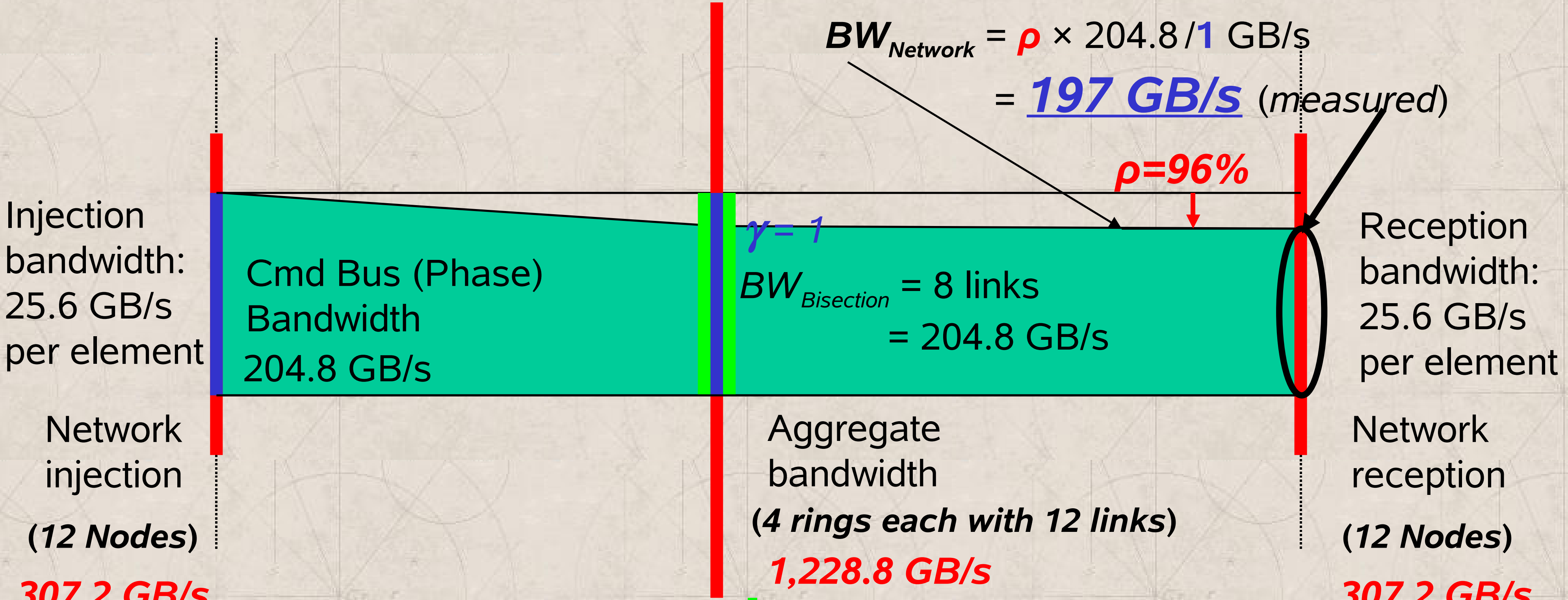
- ***Measurements show <u>197 GB/s</u> effective bandwidth***



Thomas Chen, Ram Raghavan, Jason Dale, Eiji Iwata, "Cell Broadband Engine Architecture and its first implementation: A performance view," 29 Nov 2005, http://www-128.ibm.com/developerworks/power/library/pa-cellperf/

Examples of Interconnection Networks

Cell BE EIB *Measured vs. Calculated Best-case Effective Bandwidth*



$BW_{Network}$ = ρ × 204.8 / 1 GB/s

= ***197 GB/s*** (*measured*)

ρ=96%

γ = 1

Injection bandwidth: 25.6 GB/s per element

Cmd Bus (Phase) Bandwidth 204.8 GB/s

$BW_{Bisection}$ = 8 links = 204.8 GB/s

Reception bandwidth: 25.6 GB/s per element

Network injection

(12 Nodes)

307.2 GB/s

Aggregate bandwidth

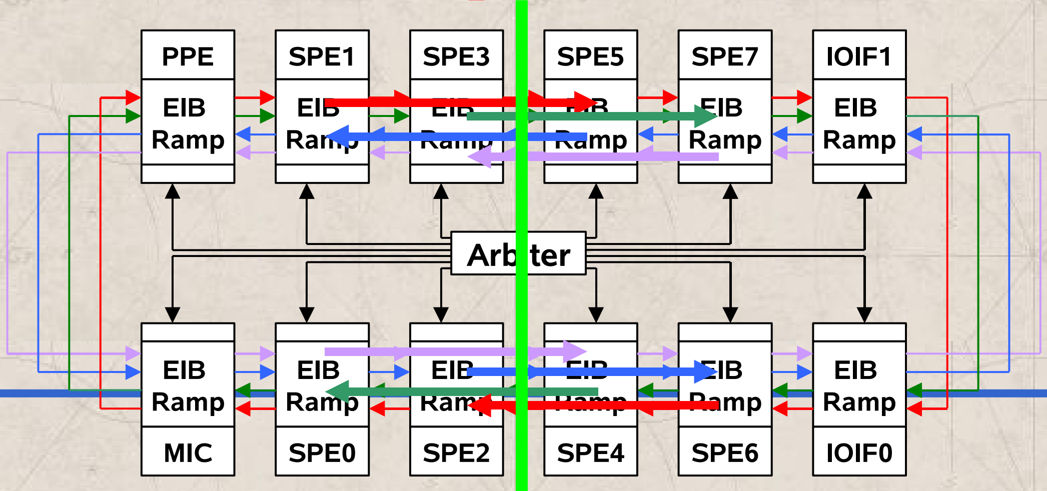
(4 rings each with 12 links)

1,228.8 GB/s

Network reception

(12 Nodes)

307.2 GB/s



*Contention-free traffic pattern in which **σ** = 1*

ρ can, at best, reach 100% since no ring interferrence

Examples of Interconnection Networks

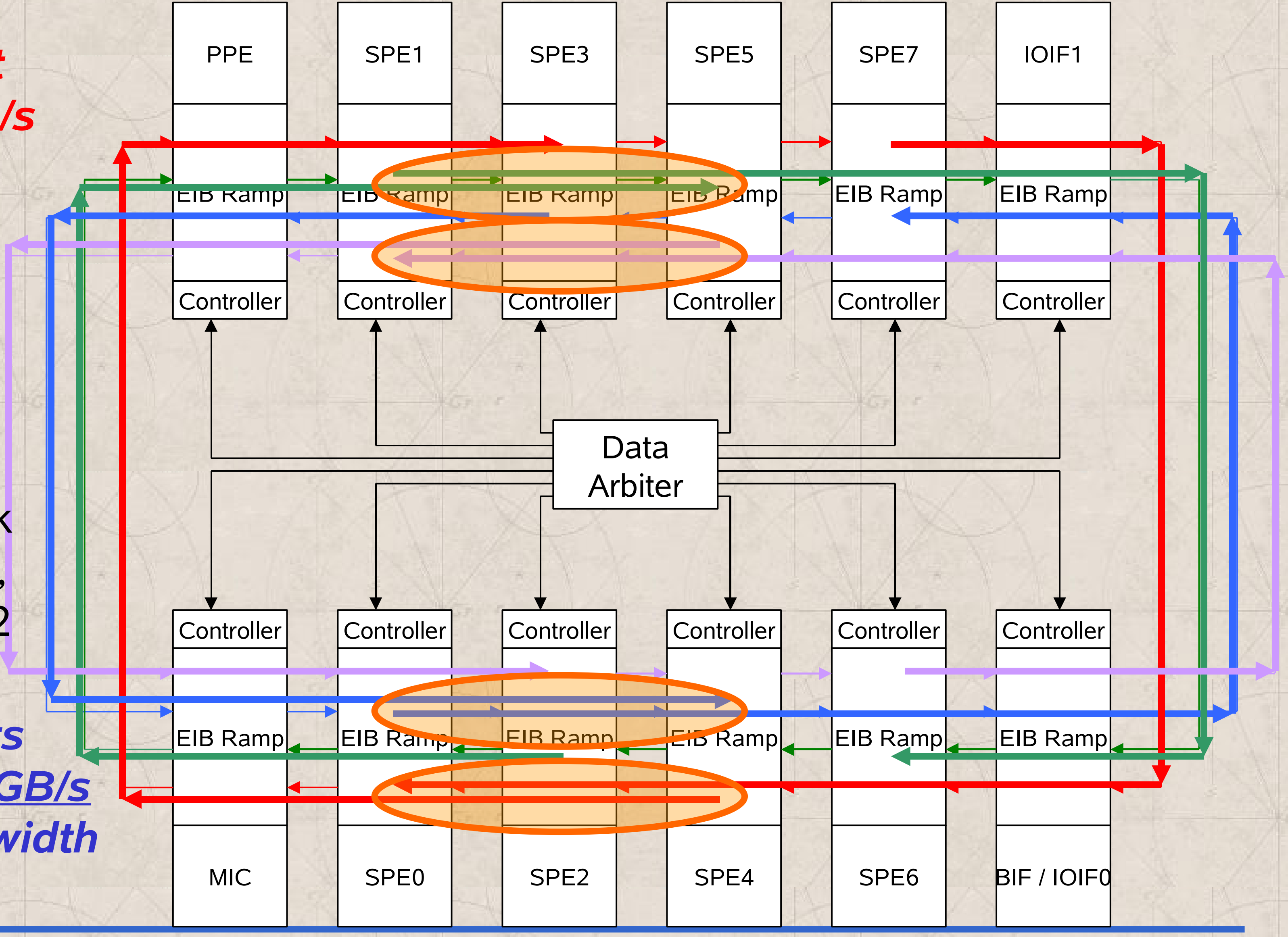
Cell BE EIB *Measured vs. Calculated Best-case Effective Bandwidth*

- ***Non-coherent BW ≤ 204.8 GB/s***

- *Traffic pattern:* ring contention allows only <u>*one*</u> transfer per ring

- Significantly reduces network efficiency factor, *ρ*, by at least 1/2

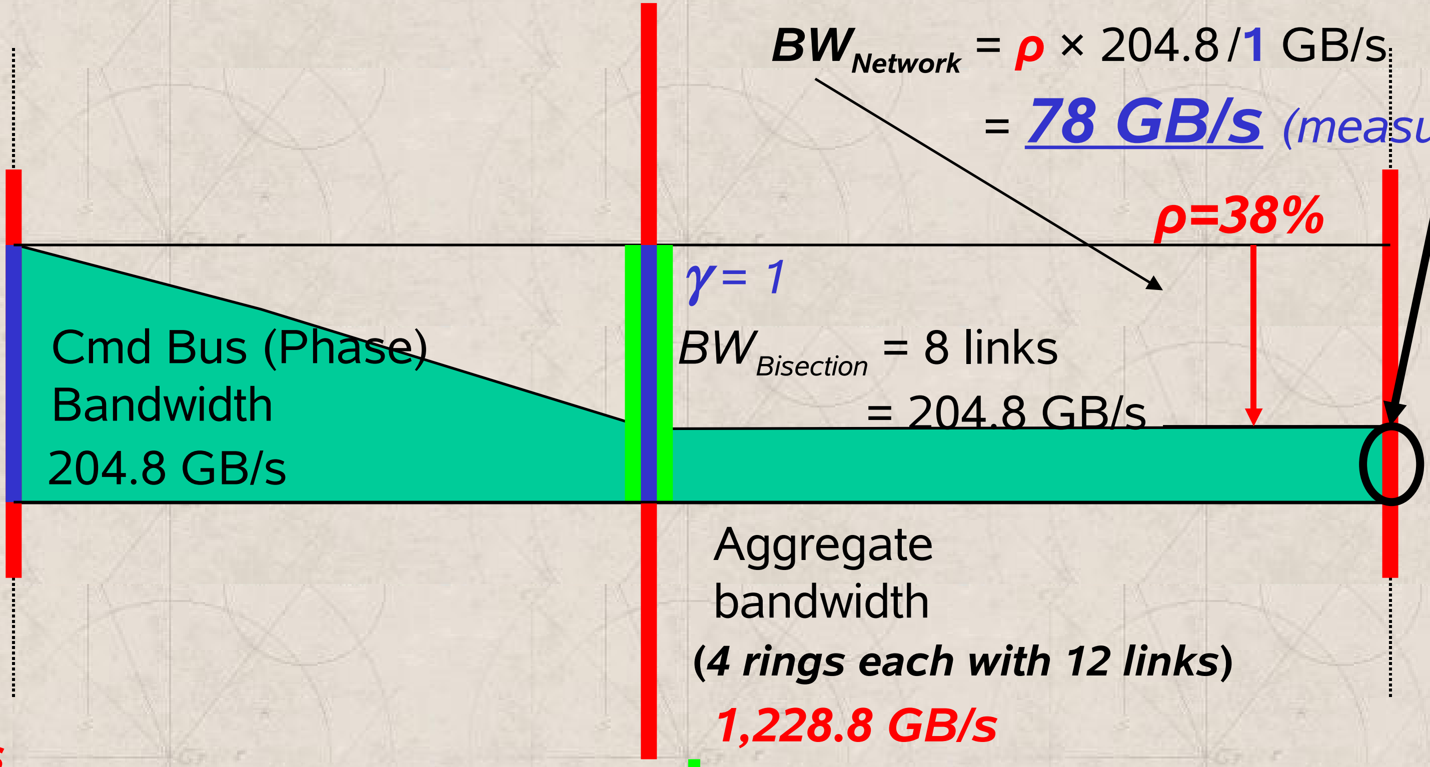
- ***Measurements show only <u>78 GB/s</u> effective bandwidth***



Thomas Chen, Ram Raghavan, Jason Dale, Eiji Iwata, “Cell Broadband Engine Architecture and its first implementation: A performance view,” 29 Nov 2005, http://www-128.ibm.com/developerworks/power/library/pa-cellperf/

Examples of Interconnection Networks

Cell BE EIB *Measured vs. Calculated Best-case Effective Bandwidth*



$BW_{Network} = \rho \times 204.8/1$ GB/s

= ***78 GB/s*** *(measured)*

ρ=38%

γ = 1

$BW_{Bisection}$ = 8 links

= 204.8 GB/s

Injection bandwidth: 25.6 GB/s per element

Cmd Bus (Phase) Bandwidth 204.8 GB/s

Reception bandwidth: 25.6 GB/s per element

Network injection

(12 Nodes)

307.2 GB/s

Aggregate bandwidth

(4 rings each with 12 links)

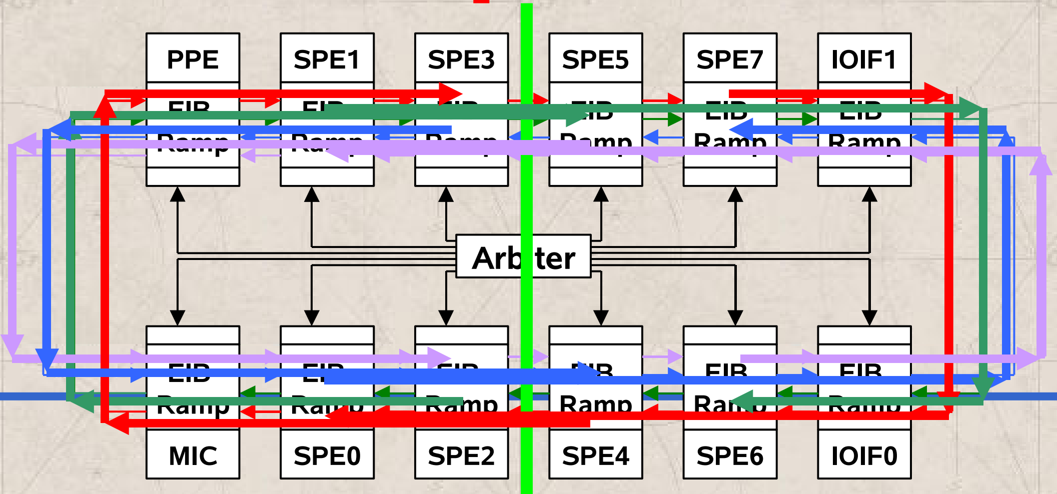
1,228.8 GB/s

Network reception

(12 Nodes)

307.2 GB/s

*Contention due to traffic pattern although **σ** = 1: ring interferrence*

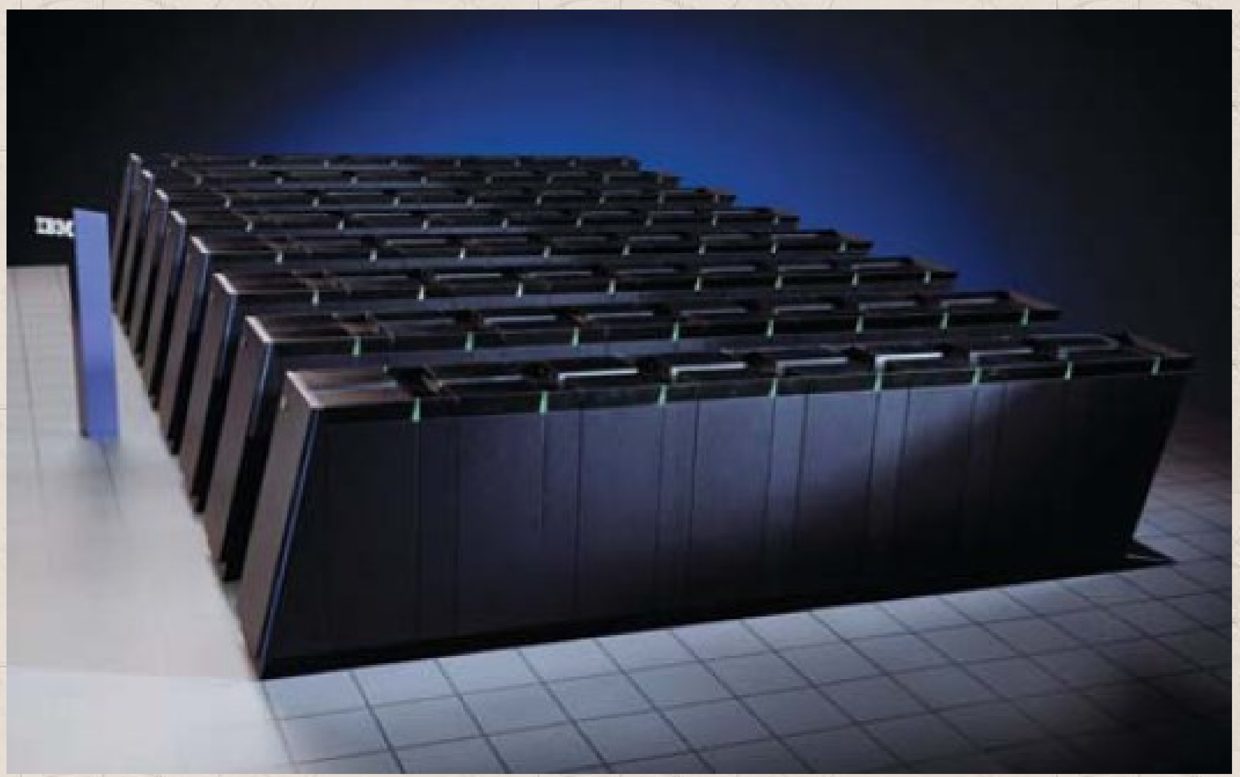


ρ limited, at best, to only 50% due to ring interferrence

Examples of Interconnection Networks

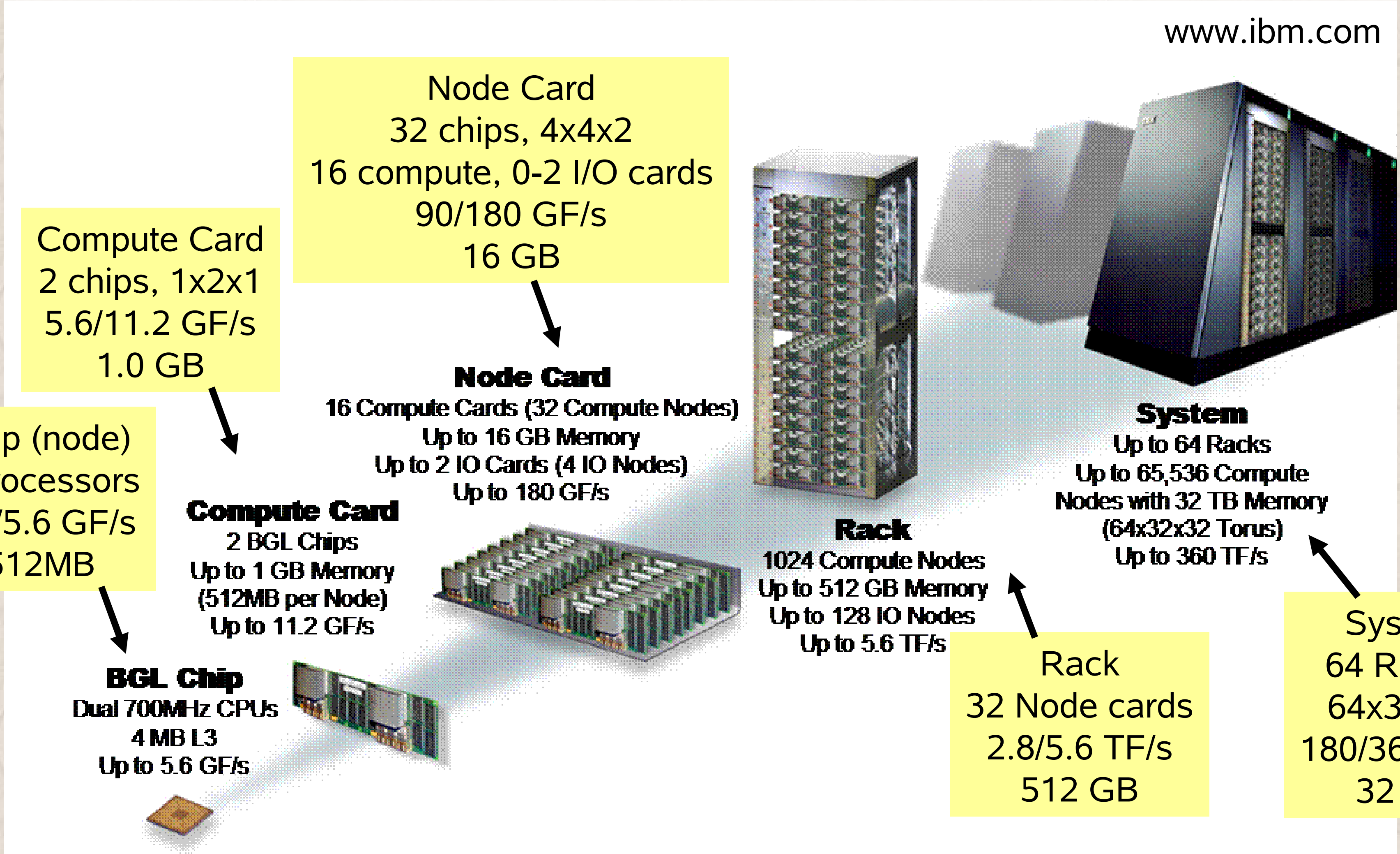
Blue Gene/L 3D Torus Network

- 360 TFLOPS (peak)
- 2,500 square feet
- Connects 65,536 dual-processor nodes and 1,024 I/O nodes
  - One processor for computation; other meant for communication



Examples of Interconnection Networks

Blue Gene/L 3D Torus Network



Node Card
32 chips, 4x4x2
16 compute, 0-2 I/O cards
90/180 GF/s
16 GB

Compute Card
2 chips, 1x2x1
5.6/11.2 GF/s
1.0 GB

Chip (node)
2 processors
2.8/5.6 GF/s
512MB

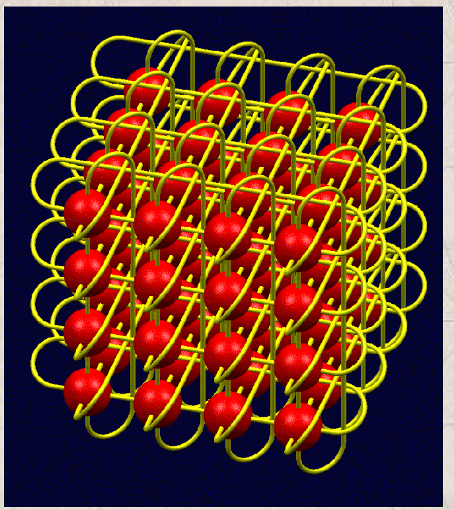
Rack
32 Node cards
2.8/5.6 TF/s
512 GB

System
64 Racks,
64x32x32
180/360 TF/s
32 TB

Node distribution: Two nodes on a 2 x 1 x 1 compute card, 16 compute cards + 2 I/O cards on a 4 x 4 x 2 node board, 16 node boards on an 8 x 8 x 8 midplane, 2 midplanes on a 1,024 node rack, 8.6 meters maximum physical link length

Examples of Interconnection Networks

Blue Gene/L 3D Torus Network



- Main network: 32 x 32 x 64 3-D torus
  - Each node connects to six other nodes
  - Full routing in hardware
- Links and Bandwidth
  - 12 bit-serial links per node (6 in, 6 out)
  - Torus clock speed runs at 1/4th of processor rate
  - Each link is 1.4 Gb/s at target 700-MHz clock rate (175 MB/s)
  - High internal switch connectivity to keep all links busy
    - External switch input links: 6 at 175 MB/s each (1,050 MB/s aggregate)
    - External switch output links: 6 at 175 MB/s each (1,050 MB/s aggregate)
    - Internal datapath crossbar input links: 12 at 175 MB/s each
    - Internal datapath crossbar output links: 6 at 175 MB/s each
    - Switch injection links: 7 at 175 MBps each (2 cores, each with 4 FIFOs)
    - Switch reception links: 12 at 175 MBps each (2 cores, each with 7 FIFOs)

Examples of Interconnection Networks

Blue Gene/L 3D Torus Network

- Routing
  - Fully-adaptive deadlock-free routing based on bubble flow control and Duato’s Protocol
    - DOR and bubble mechanism are used for escape path
  - Hint (direction) bits at the header
    - “100100” indicates the packet must be forwarded in X+ and Y-
    - Neighbor coordinate registers at each node
      - A node cancels hint bit for next hop based on these registers
  - A bit in the header allows for broadcast
  - Dead nodes or links avoided with appropiate hint bits

Examples of Interconnection Networks

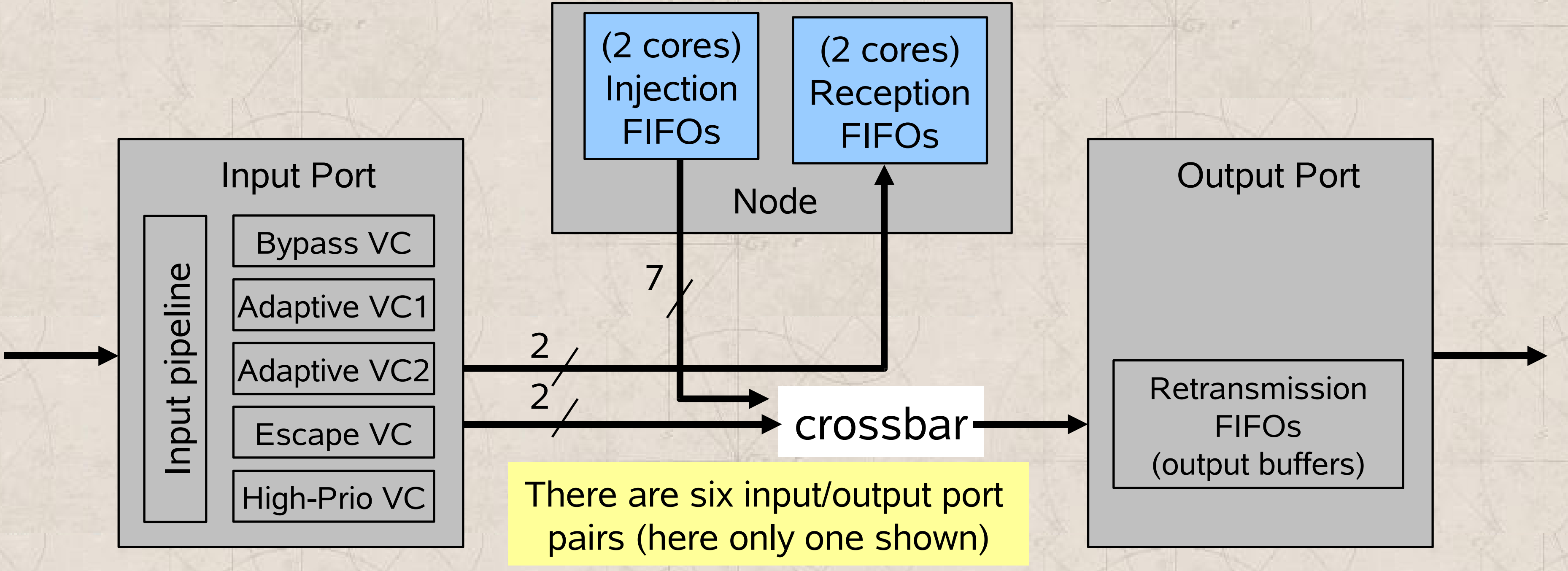
Blue Gene/L 3D Torus Network

- Routing
  - Variable packet sizes (m x 32, m = 1 ... 8)
  - First eight bytes for packet header
    - Sequence number
    - Routing information (destination, virtual channel, size)
    - CRC (1 byte) of packet header
  - Trailer
    - CRC (3 bytes) at link level (includes the CRC at the header)
    - One-byte valid indicator

Examples of Interconnection Networks

Blue Gene/L 3D Torus Network

- Torus logic (processor interface)
  - – 8 Injection FIFOs (injection queues) feed into 7 injection links
    - › 2 cores, each has 1 high-priority FIFO and 3 normal FIFOs
  - – 14 reception FIFOs (reception queues) fed by 12 reception links
    - › 2 cores, each has 1 high-priority FIFO and 6 normal FIFOs (one designated to receive from each switch dimension & direction)



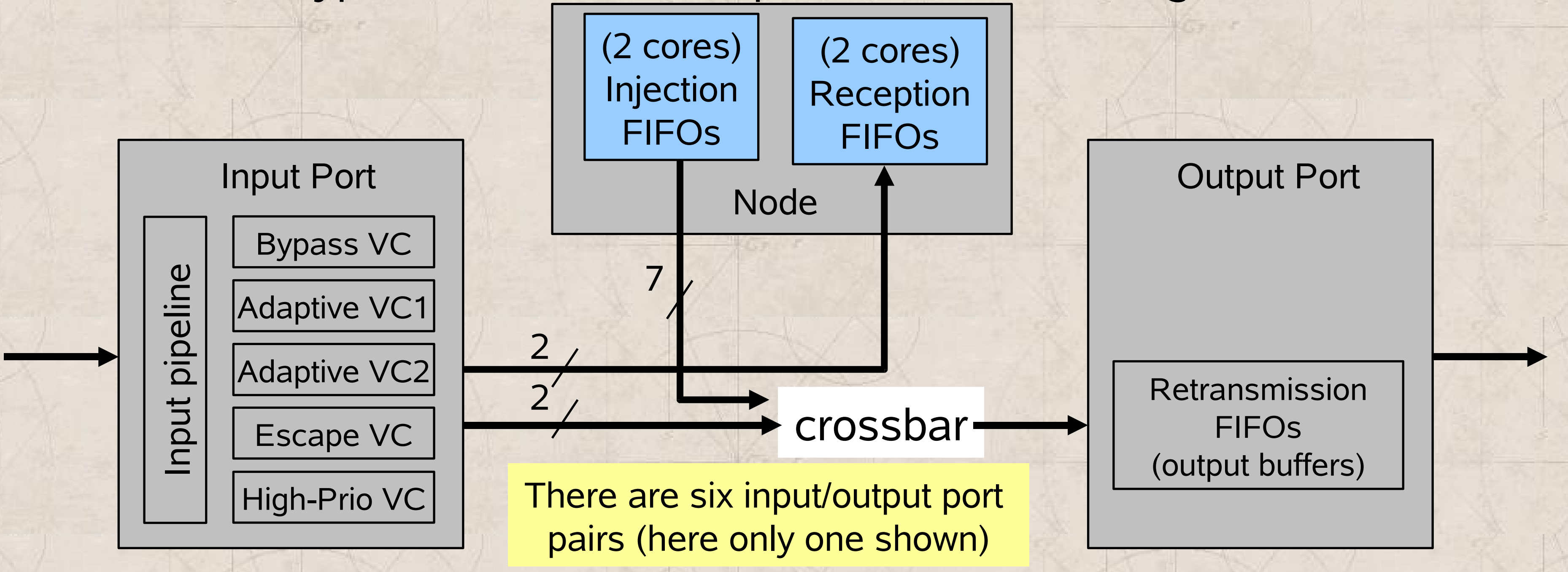
There are six input/output port pairs (here only one shown)

"Blue Gene/L Torus Interconnection Network," N. R. Adiga, et al., IBM J. Res. & Dev., Vol. 49, No. 2/3, pp. 265-276, March/May 2005.

Examples of Interconnection Networks

Blue Gene/L 3D Torus Network

- Torus logic (receive unit)
  - Eight-stage input pipeline, 4 VCs and a bypass channel:
    - Each VC has 1 KB of buffering (four full-sized packets)
    - HOL blocking reduction with VCs
    - Deadlock avoidance with *bubble escape VC with DOR*
    - Bypass channel allows packets to flow through switch

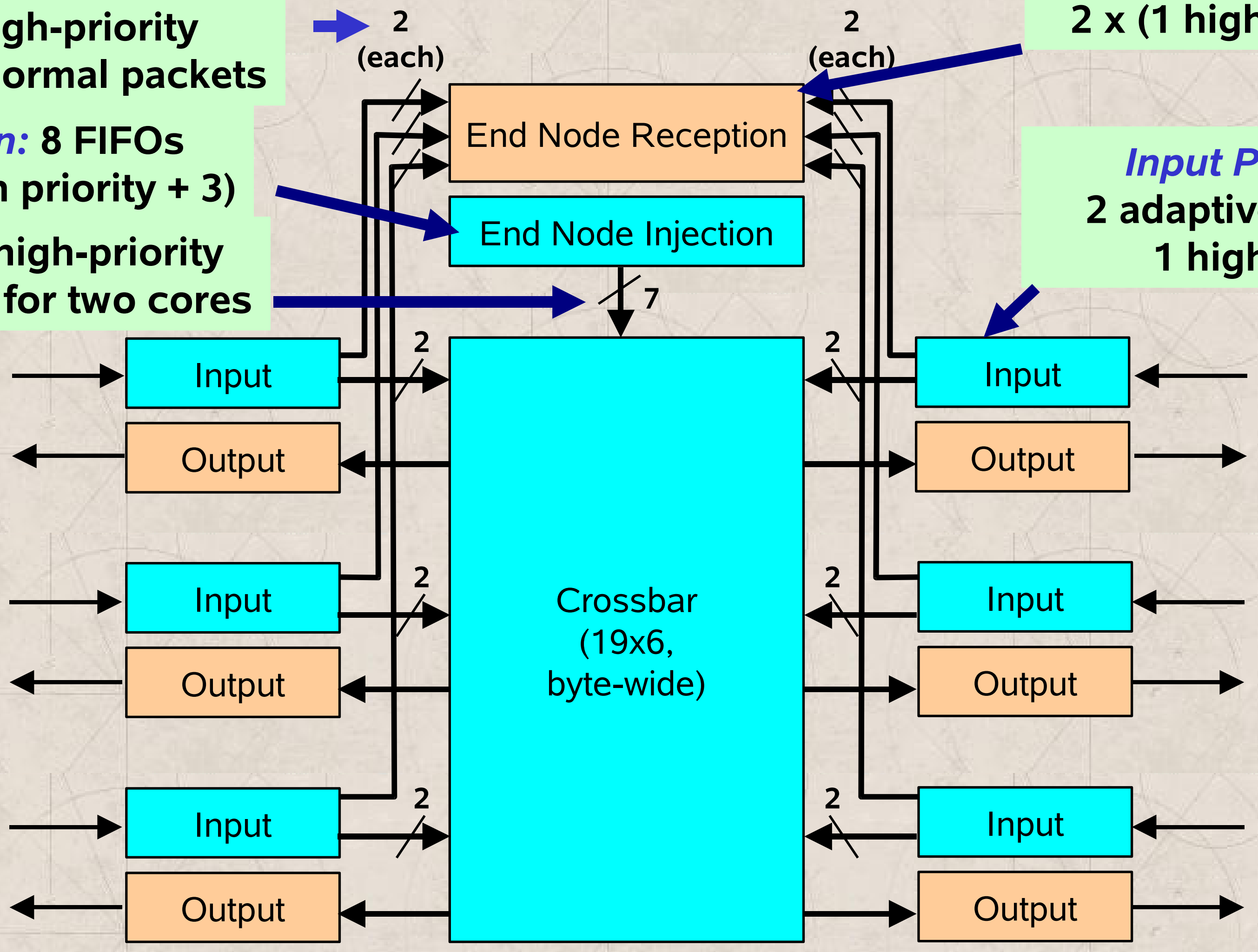


There are six input/output port pairs (here only one shown)

"Blue Gene/L Torus Interconnection Network," N. R. Adiga, et al., IBM J. Res. & Dev., Vol. 49, No. 2/3, pp. 265-276, March/May 2005.

Examples of Interconnection Networks

Blue Gene/L 3D Torus Network



1 for high-priority and 1 for normal packets

Injection: **8 FIFOs**
2 x (1 high priority + 3)

1 shared high-priority and 3 each for two cores

Reception: **14 FIFOs**
2 x (1 high-priority + 6)

Input Port: **4 VCs**
2 adaptive, 1 bubble, 1 high-priority

Blue Gene/L Switch

"Blue Gene/L Torus Interconnection Network," N. R. Adiga, et al., IBM J. Res. & Dev., Vol. 49, No. 2/3, pp. 265-276, March/May 2005.

Examples of Interconnection Networks

Blue Gene/L 3D Torus Network

- Flow control
  - Credit-based (token) flow-control per VC buffer
    - A token represents a 32-byte chunk
  - Bubble rules are applied to the escape VC
    - Tokens for one full-sized packet is required for a packet in the escape VC (bubble) to advance
    - Tokens for two full-sized packets are required for
      - A packet entering the escape VC or
      - A packet turning into a new direction
      - An adaptive VC packet enters the escape VC
    - Dimension-ordered routing on the escape VC

Examples of Interconnection Networks

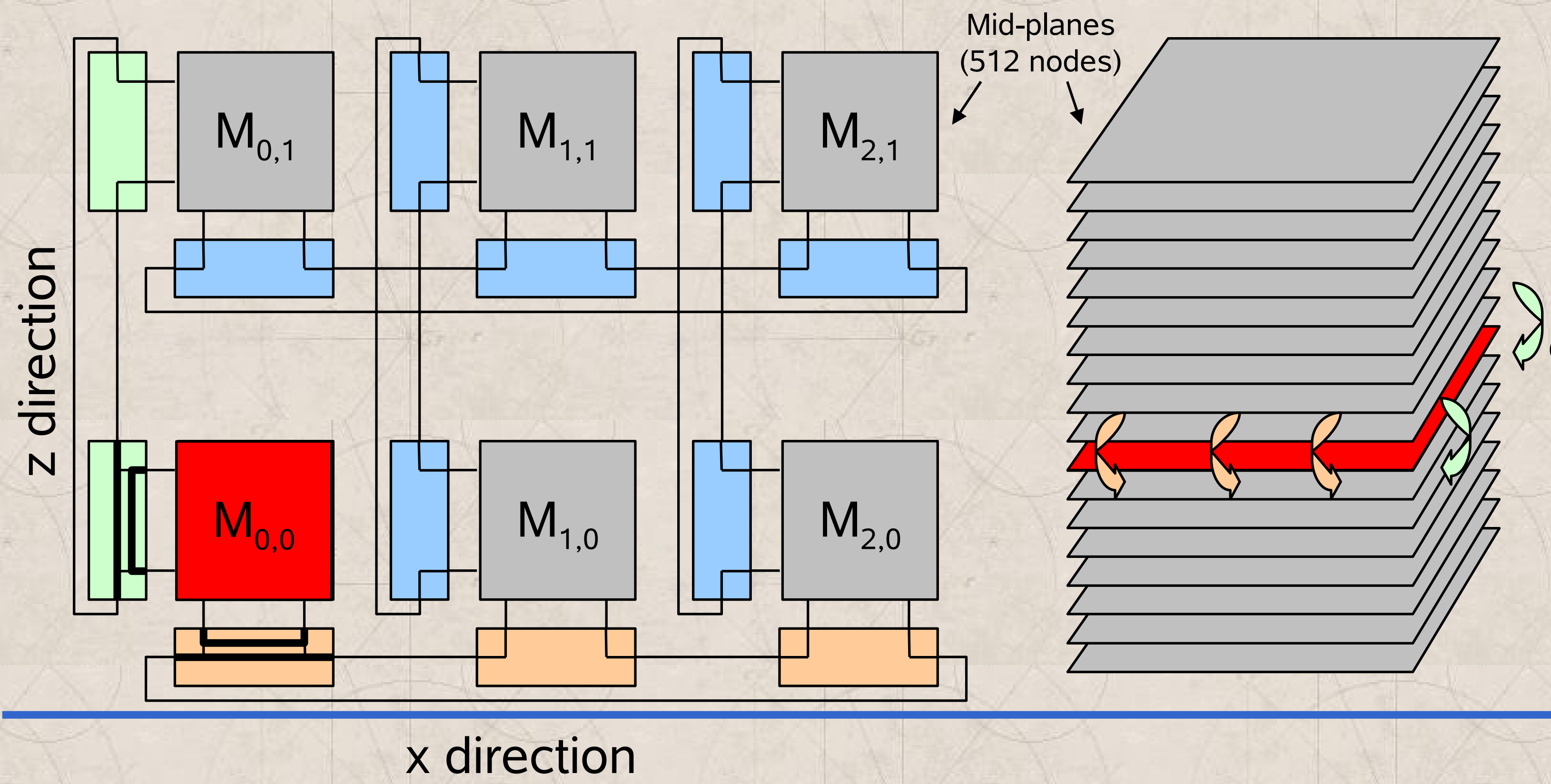
Blue Gene/L 3D Torus Network

- Distributed arbitration (two-phased arbitration)
  - Precomputation phase (computation of direction and requested virtual channel)
    - For each packet at the head of a VC FIFO or injection FIFO
    - Only one option is supplied for deterministically routed packets
    - Many options supplied for dynamically routed packets
    - JSQ (join-the-shortest-queue) algorithm
  - First phase (select one of the requesting packets at input port)
    - Packets in high-prior VC have maximum priority
    - Packets in bypass VC have default priority
    - SLQ (serve-the-longest-queue) algorithm
    - A fraction of cycles are used to select a packet randomly in case of a tie
  - Second phase (select one of the requestors at the output port)
    - Independent arbiter per output port
    - Highest priority is given to token and ack packets
    - SQL mostly, but periodically choose at random to avoid starvation

Examples of Interconnection Networks

Blue Gene/L Torus Network

- Fault tolerance
  - Static fault model with checkpointing
  - Additional links boards at each rack
    - Each rack can be connected with neighbor racks
    - Internal switches allow skiping one plane (512 nodes)



Inter mid-planes switches and links operating in normal “pass-thru” mode

Inter mid-planes switches and links operating in fault “bypass” mode

Topology remains the same

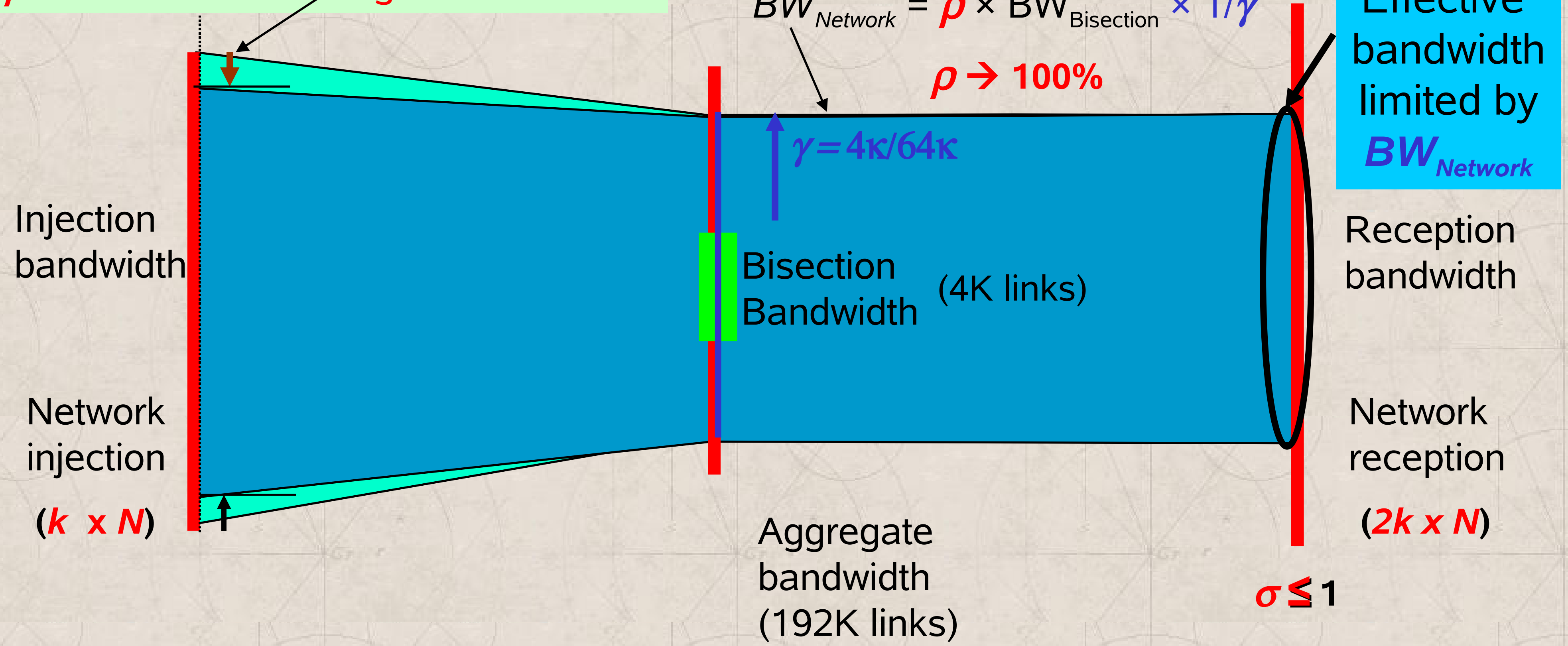
Application of Model to Blue Gene/L

*Throughput Model Applied to Blue Gene/L: **Effective BW***

$BW_{NetworkInjection}$ = N x $BW_{LinkInjection}$ x 7 = 64K x 87.5 MB/s x 7 = 38.28 TB/s, but
Sending latency = 3 µs, max packet size=256B,
packet transmission = 2.92 µs, so
$BW_{NetworkInjection}$ = 64K x **85.33 MB/s** x 7 = 37.33 TB/s (***97% of max. Inj. BW***)

Pipeline injection can overlap sending latency

ρ = 100% for near-neighbour traffic



$BW_{Network}$ = ρ × $BW_{Bisection}$ × $1/\gamma$

ρ → 100%

$\gamma = 4K/64K$

Effective bandwidth limited by ***$BW_{Network}$***

Injection bandwidth

Network injection

(***k* x *N***)

Bisection Bandwidth

(4K links)

Aggregate bandwidth (192K links)

Reception bandwidth

Network reception

(***2k* x *N***)

$\sigma \leqq 1$

Effective bandwidth=11.469 TB/s max

Examples of Interconnection Networks

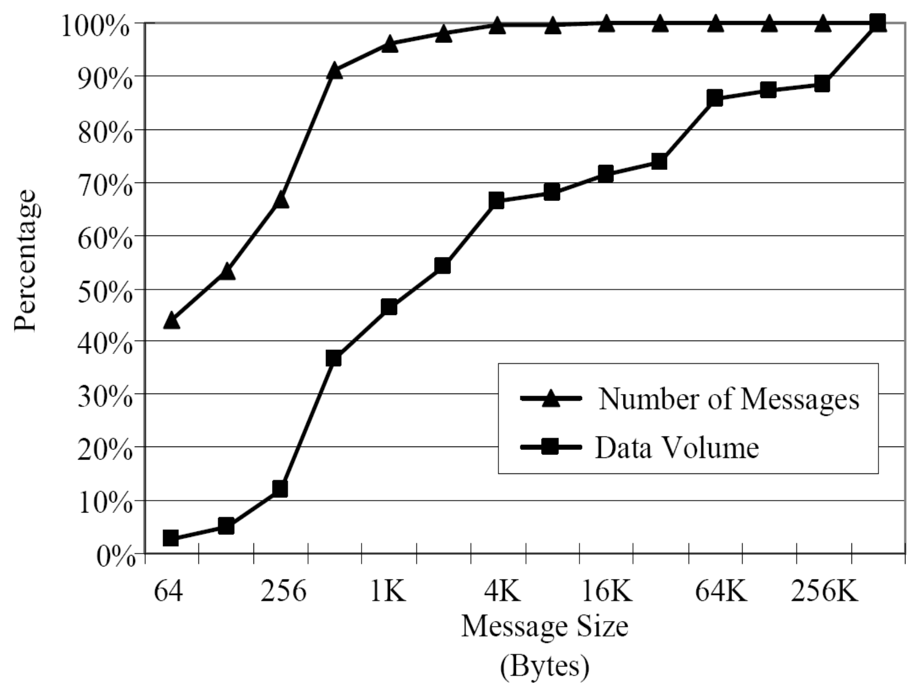
System Area Networks (SANs)

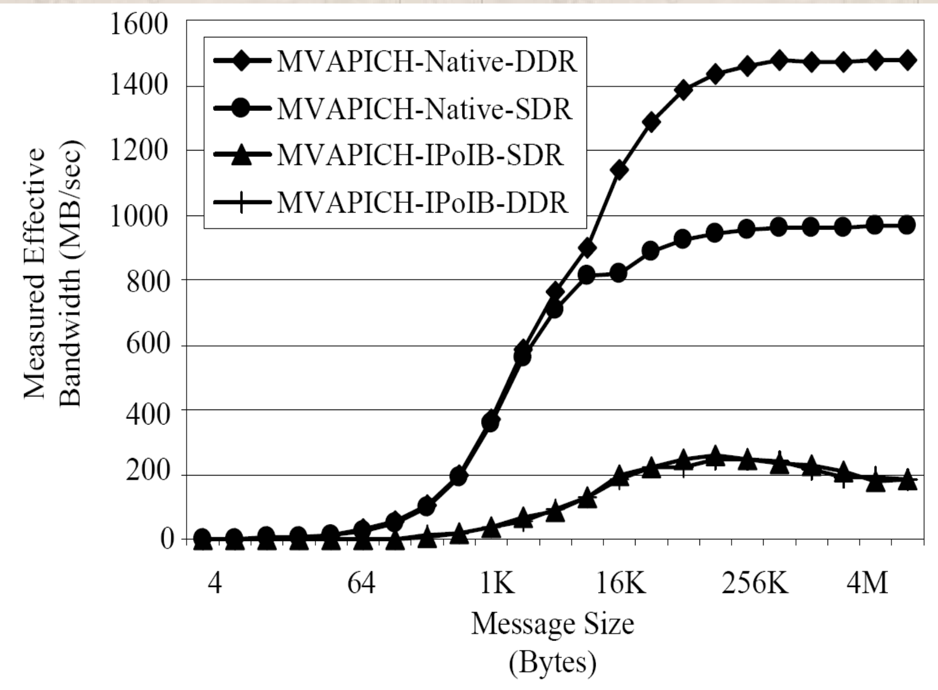
- *InfiniBand*
  - Industry-wide networking standard (InfiniBand Trade Association)
  - Can be applied to
    - System area networks for interprocessor communication
    - Storage area networks for server I/O
  - Switch-based interconnect, providing flexibility in
    - Topology
    - Routing
    - Arbitration
  - 2 – 120 Gbps/link per direction (300 meters max distance)
  - 16 VCs, 16 service levels (SLs) for quality of service (QoS)
  - Credit-based link-level flow control
  - Weighted round-robin fair scheduling of flows
  - Forwarding tables at switches (distributed routing)
  - Protocol off-load (to NIC) via MVAPICH

Examples of Interconnection Networks

System Area Networks (SANs)

- *InfiniBand*





Cummulative percentage of messages and volume of data transferred as message size varies for the Fluent application.

Effective bandwidth versus message size measured on an SDR and DDR InfiniBand networks running MVAPICH with OS-bypass (native) and without (IPoIB)

Data collected by D.K. Panda, S. Sur, and L. Chai, The Ohio State University, 2005:
http://nowlab.cse.ohio-state.edu/projects/mpi-iba

Examples of Interconnection Networks

System Area Networks (SANs)

- *Characteristics of SANs Implemented in Top 10 Supercomputers*

Network name [vendors]	**Used in top 10 supercomputer clusters (2005)**	**Number of nodes**	**Basic network topology**	**Raw link** bidirectional **BW**	**Routing algorithm**	**Arbitration technique**	**Switching technique; flow control**
InfiniBand [Mellanox, Voltair]	SGI Altrix, and Dell Poweredge Thunderbird	> millions (2^{128} GUID addresses, like IPv6)	completely configurable (arbitrary)	4 Gbps to 240 Gbps	arbitrary (table-driven) typically up*/down*	weighted RR fair scheduling (2-level priority)	cut-through, 16 VCs (15 for data); credit-based
Myrinet-2000 [Myricom]	Barcelona Supercomputer Center in Spain	8,192 nodes	Bidi. MIN w/16-port, bidi. switches (Clos network)	4 Gbps	source-based dispersive (adaptive) minimal routing	round-robin arbitration	cut-through switching w/ no VCs; Xon/Xoff
QsNet^{II} [Quadrics]	Intel Thunder Itanium2 Tiger4	> 10s of thousands	fat tree w/8-port bidirectional switches	21.3 Gbps	source-based LCA adaptive shortest-path routing	2-phased RR, priority, aging, distributed at output ports	wormhole w/2 VCs; credit-based

Outline

- E.1 Introduction *(Lecture 1)*
- E.2 Interconnecting Two Devices *(Lecture 1)*
- E.3 Interconnecting Many Devices *(Lecture 2)*
- E.4 Network Topology *(Lecture 2)*
- E.5 Network Routing, Arbitration, and Switching *(Lecture 3)*
- E.6 Switch Microarchitecture *(Lecture 4)*
- E.7 Practical Issues for Commercial Interconnection Networks *(Lec 4)*
- E.8 Examples of Interconnection Networks *(Lecture 5)*
- E.9 Internetworking (skipped)
- E.10 Crosscutting Issues for Interconnection Networks (skipped)
- **E.11 Fallacies and Pitfalls** *(Lecture 5)*
  - **Fallacies**
  - **Pitfalls**
- E.12 Concluding Remarks and References *(Lecture 5)*

Fallacies and Pitfalls

Fallacies

- The interconnection network is very fast and does not need to be improved
- Bisection bandwidth is an accurate cost constraint of a network
- Zero-copy protocols do not require copying messages or fragments from one buffer to another
- MINs are more cost-effective than direct networks
- Direct networks are more performance-effective than MINs

Fallacies and Pitfalls

Fallacies

- Low-dimensional direct networks achieve higher performance than high-dimensional networks such as hypercubes
- Wormhole switching achieves better performance than other switching techniques
- Implementing a few virtual channels always increases throughput by allowing packets to pass through blocked packets ahead
- Adaptive routing causes out-of-order packet delivery, thus introducing too much overhead to re-order packets
- Adaptive routing by itself is sufficient to tolerate network faults

Fallacies and Pitfalls

Pitfalls

- Using bandwidth (in particular, bisection bandwidth) as the *only* measure of network performance
- Not providing sufficient reception link bandwidth
- Using high-performance NICs, but forgetting the I/O subsystem
- Ignoring software overhead when determining performance
- Providing features only within the network versus end-to-end

Outline

- E.1 Introduction *(Lecture 1)*
- E.2 Interconnecting Two Devices *(Lecture 1)*
- E.3 Interconnecting Many Devices *(Lecture 2)*
- E.4 Network Topology *(Lecture 3)*
- E.5 Network Routing, Arbitration, and Switching *(Lecture 3)*
- E.6 Switch Microarchitecture *(Lecture 4)*
- E.7 Practical Issues for Commercial Interconnection Networks *(Lec 4)*
- E.8 Examples of Interconnection Networks *(Lecture 5)*
- E.9 Internetworking (skipped)
- E.10 Crosscutting Issues for Interconnection Networks (skipped)
- E.11 Fallacies and Pitfalls *(Lecture 5)*
- **E.12 Concluding Remarks and References** *(Lecture 5)*

Concluding Remarks and References

Concluding Remarks

- Interconnect design is an exciting area of computer architecture
  - on-chip networks between cores on within a chip
  - off-chip networks between chips and boards within a system
  - external networks between systems
- Interconnection networks should be designed to transfer the maximum amount of information within the least amount of time (and cost, power constraints) so as not to bottleneck the system
- The design of interconnection networks is end-to-end
  - injection links/interface, network fabric, reception links/interface
  - topology, routing, arbitration, switching, and flow control are among key concepts in realizing high-performance designs
  - *a simple, general throughput model can be used to guide design*
- Improving the performance of interconnection networks is critical to advancing our information- and communication-centric world

Concluding Remarks and References

References

Agarwal, A. [1991]. “Limits on interconnection network performance,” *IEEE Trans. on Parallel and Distributed Systems* 2:4 (April), 398–412.

Anderson, T. E., D. E. Culler, and D. Patterson [1995]. “A case for NOW (networks of workstations),” *IEEE Micro 15:1* (February), 54–64.

Anjan, K. V., and T. M. Pinkston [1995]. “An efficient, fully-adaptive deadlock recovery scheme: Disha,” *Proc. 22nd Int’l Symposium on Computer Architecture* (June), Italy.

Benes, V.E. [1962]. “Rearrangeable three stage connecting networks,” *Bell System Technical Journal* 41, 1481–1492.

Bertozzi, D., A. Jalabert, S. Murali, R. Tamhankar, S. Stergiou, L. Benini, and G. De Micheli [2005]. “NoC synthesis flow for customized domain specific multiprocessor systems-on-chip,” *IEEE Trans. on Parallel and Distributed Systems* 16:2 (February),113–130.

Bhuyan, L. N., and D. P. Agrawal [1984]. “Generalized hypercube and hyperbus structures for a computer network,” *IEEE Trans. on Computers* 32:4 (April), 323–333.

Clos, C. [1953]. “A study of non-blocking switching networks,” *Bell Systems Technical Journal* 32 (March), 406–424.

Dally, W. J. [1990]. “Performance analysis of k-ary n-cube interconnection networks,” *IEEE Trans. on Computers* 39:6 (June), 775–785.

Dally, W. J. [1992]. “Virtual channel flow control,” *IEEE Trans. on Parallel and Distributed Systems* 3:2 (March), 194–205.

Concluding Remarks and References

References

Dally, W. J. [1999]. "Interconnect limited VLSI architecture," *Proc. of the International Interconnect Technology Conference*, San Francisco (May).

Dally, W. J., and C. I. Seitz [1986]. "The torus routing chip," *Distributed Computing* 1:4, 187–196.

Dally, W. J., and B. Towles [2001]. "Route packets, not wires: On-chip interconnection networks," *Proc. of the Design Automation Conference*, Las Vegas (June).

Dally, W. J., and B. Towles [2004]. *Principles and Practices of Interconnection Networks*, Morgan Kaufmann Publishers, San Francisco.

Duato, J. [1993]. "A new theory of deadlock-free adaptive routing in wormhole networks," *IEEE Trans. on Parallel and Distributed Systems* 4:12 (Dec.) 1320–1331.

Duato, J., I. Johnson, J. Flich, F. Naven, P. Garcia, T. Nachiondo [2005]. "A new scalable and cost-effective congestion management strategy for lossless multistage interconnection networks," *Proc. 11th Int'l Symposium on High Performance Computer Architecture* (February), San Francisco.

Duato, J., O. Lysne, R. Pang, and T. M. Pinkston [2005]. "Part I: A theory for deadlock-free dynamic reconfiguration of interconnection networks," *IEEE Trans. on Parallel and Distributed Systems* 16:5 (May), 412–427.

Duato, J., and T. M. Pinkston [2001]. "A general theory for deadlock-free adaptive routing using a mixed set of resources," *IEEE Trans. on Parallel and Distributed Systems* 12:12 (December), 1219–1235.

Duato, J., S. Yalamanchili, and L. Ni [2003]. *Interconnection Networks: An Engineering Approach*, 2nd printing, Morgan Kaufmann Publishers, San Francisco.

Concluding Remarks and References

References

Glass, C. J., and L. M. Ni [1992]. “The Turn Model for adaptive routing,” *Proc. 19th Int’l Symposium on Computer Architecture* (May), Australia.

Gunther, K. D. [1981]. “Prevention of deadlocks in packet-switched data transport systems,” *IEEE Trans. on Communications* COM–29:4 (April), 512–524.

Ho, R., K. W. Mai, and M. A. Horowitz [2001]. “The future of wires,” *Proc. of the IEEE* 89:4 (April), 490–504.

Holt, R. C. [1972]. “Some deadlock properties of computer systems,” *ACM Computer Surveys* 4:3 (September), 179–196.

Infiniband Trade Association [2001]. *InfiniBand Architecture Specifications Release 1.0.a, www.infinibandta.org.*

Jantsch. A., and H. Tenhunen [2003]. *Networks on Chips*, eds., Kluwer Academic Publishers, The Netherlands.

Kermani, P., and L. Kleinrock [1979]. “Virtual Cut-Through: A New Computer Communication Switching Technique,” *Computer Networks*, 3 (January), 267–286.

Leiserson, C. E. [1985]. “Fat trees: Universal networks for hardware-efficient supercomputing,” *IEEE Trans. on Computers* C–34:10 (October), 892–901.

Merlin, P. M., and P. J. Schweitzer [1980]. “Deadlock avoidance in store-and-forward networks—I: Store-and-forward deadlock,” *IEEE Trans. on Communications* COM–28:3 (March), 345–354.

Metcalfe, R. M., and D. R. Boggs [1976]. “Ethernet: Distributed packet switching for local computer networks,” *Comm. ACM* 19:7 (July), 395–404.

Concluding Remarks and References

References

Peh, L. S., and W. J. Dally [2001]. “A delay model and speculative architecture for pipelined routers,” *Proc. 7th Int’l Symposium on High Performance Computer Architecture* (January), Monterrey.

Pfister, Gregory F. [1998]. *In Search of Clusters,* 2nd ed., Prentice Hall, Upper Saddle River, N.J.

Pinkston, T. M. [2004]. “Deadlock characterization and resolution in interconnection networks (Chapter 13),” *Deadlock Resolution in Computer-Integrated Systems*, edited by M. C. Zhu and M. P. Fanti, Marcel Dekkar/CRC Press, 445–492.

Pinkston, T. M., A. Benner, M. Krause, I. Robinson, T. Sterling [2003]. “InfiniBand: The ‘de facto’ future standard for system and local area networks or just a scalable replacement for PCI buses?” *Cluster Computing* (Special Issue on Communication Architecture for Clusters) 6:2 (April), 95–104.

Pinkston, T. M., and J. Shin [2005]. “Trends toward on-chip networked microsystems,” *International Journal of High Performance Computing and Networking* 3:1, 3–18.

Pinkston, T. M., and S. Warnakulasuriya [1997]. “On deadlocks in interconnection networks,” *Proc. 24th Int’l Symposium on Computer Architecture* (June), Denver.

Puente, V., R. Beivide, J. A. Gregorio, J. M. Prellezo, J. Duato, and C. Izu [1999]. “Adaptive bubble router: A design to improve performance in torus networks,” *Proc. 28th Int’l Conference on Parallel Processing* (September), Aizu-Wakamatsu, Japan.

Saltzer, J. H., D. P. Reed, and D. D. Clark [1984]. “End-to-end arguments in system design,” *ACM Trans. on Computer Systems* 2:4 (November), 277–288.

Concluding Remarks and References

References

Scott, S. L., and J. Goodman [1994]. “The impact of pipelined channels on k-ary n-cube networks ,” *IEEE Trans. on Parallel and Distributed Systems 5:1 (January), 1–16.*

Tamir, Y., and G. Frazier [1992]. “Dynamically-allocated multi-queue buffers for VLSI communication switches,” *IEEE Trans. on Computers* 41:6 (June), 725–734.

Taylor, M. B., W. Lee, S. P. Amarasinghe, and A. Agarwal [2005]. “Scalar operand networks,” *IEEE Trans. on Parallel and Distributed Systems* 16:2 (February), 145–162.

von Eicken, T., D. E. Culler, S. C. Goldstein, K. E. Schauser [1992]. “Active Messages: A mechanism for integrated communication and computation,”*Proc. 19th Int’l Symposium on Computer Architecture* (May), Australia.

Vaidya, A. S., A. Sivasubramaniam, and C. R. Das [1997]. “Performance benefits of virtual channels and adaptive routing: An application-driven study,” *Proceedings of the 1997 Int’l Conference on Supercomputing* (July), Austria.

Waingold, E., M. Taylor, D. Srikrishna, V. Sarkar, W. Lee, V. Lee, J. Kim, M. Frank, P. Finch, R. Barua, J. Babb, S. Amarasinghe, and A. Agarwal [1997]. “Baring it all to software: Raw Machines,” *IEEE Computer*, 30 (September), 86–93.

Yang, Y., and G. Mason [1991]. “Nonblocking broadcast switching networks,” *IEEE Trans. on Computers* 40:9 (September), 1005–1015.