Computer Architecture

16. Interconnection Networks

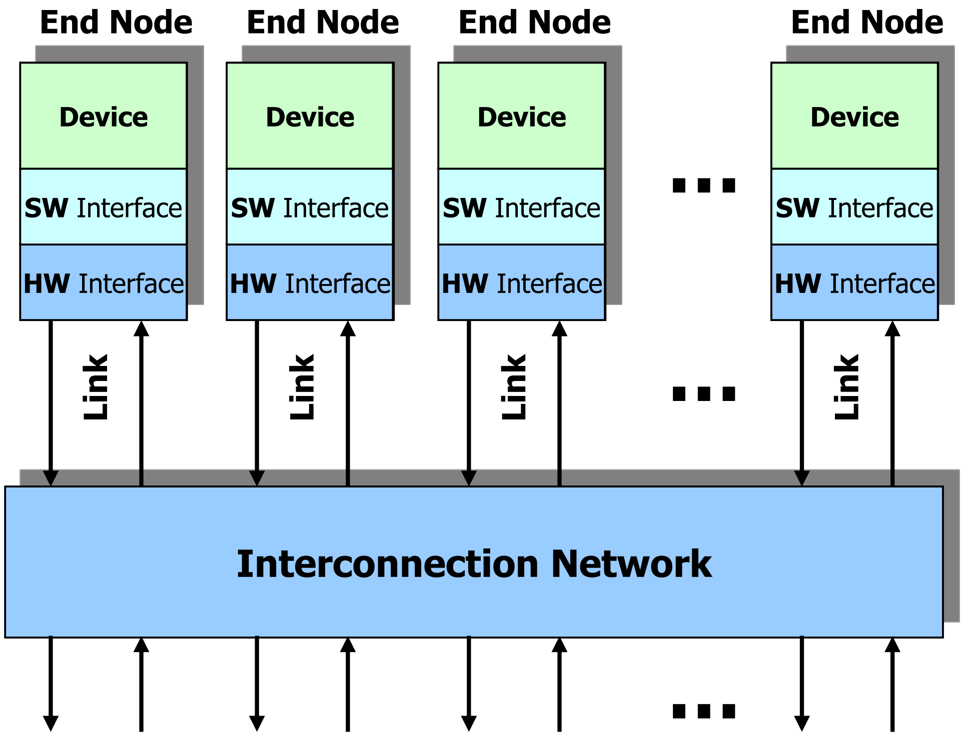
Jianhua Li
College of Computer and Information
Hefei University of Technology

Outline

- **Introduction**
- Network Topology
- Flow Control
- Example IN

How to connect individual devices together?

- 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

Reasons to concern 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

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**

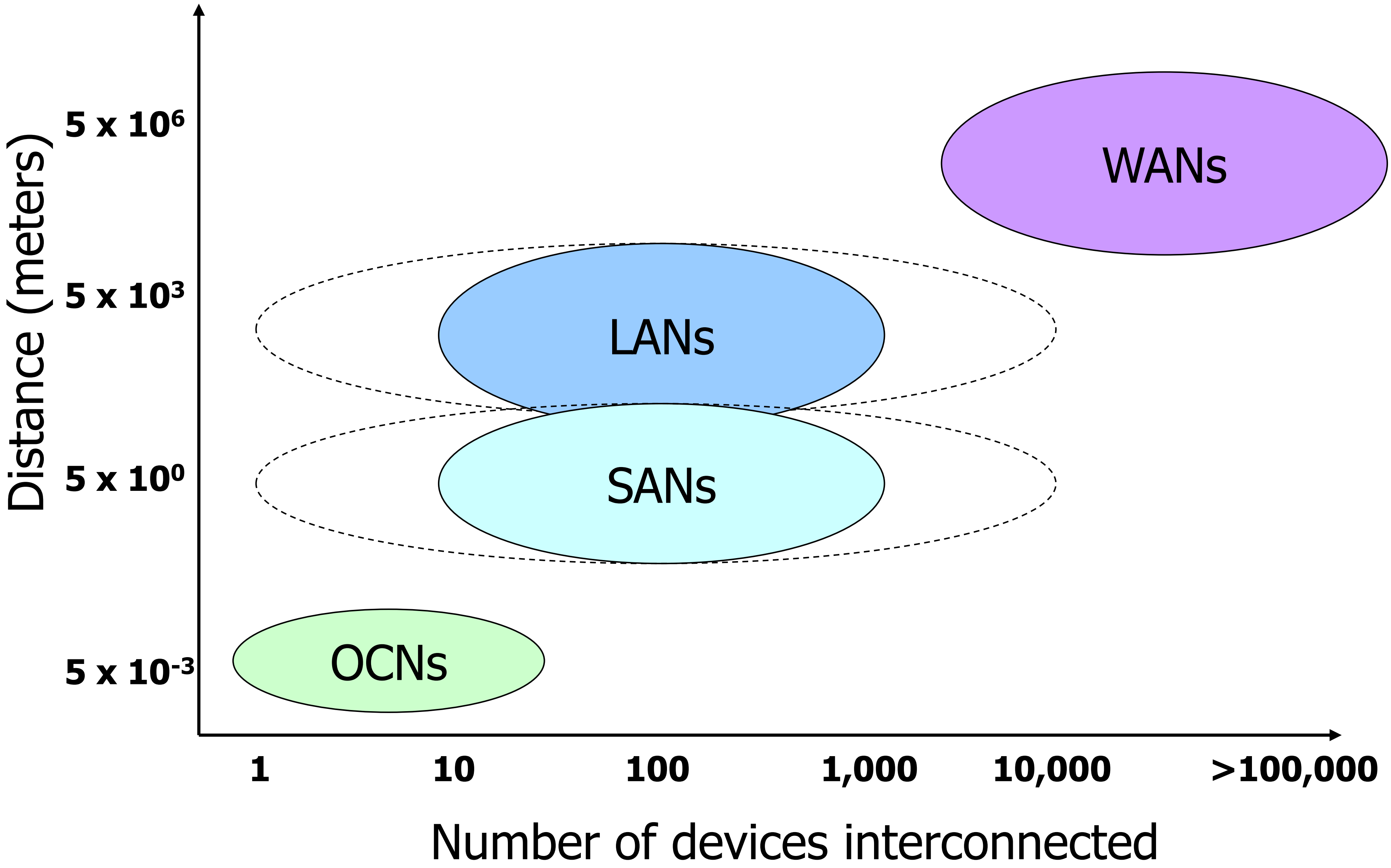
Interconnection Network Domains

- 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

Interconnection Network Domains

- Local area networks (LANs)
  - 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** (some with distance spans of a few tens of kilometers)
  - 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

Interconnection Network Domains



Outline

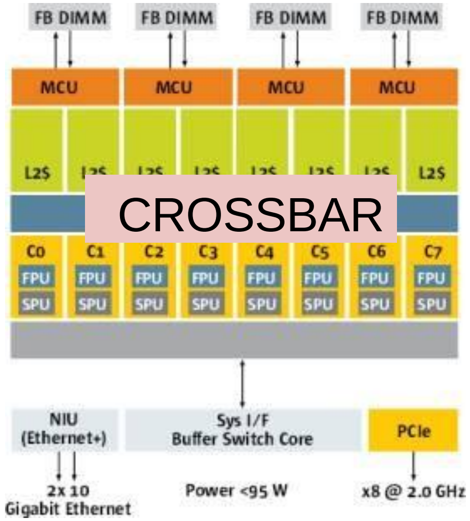
- Introduction
- **Network Topology**
- Flow Control
- Example IN

Topology Overview

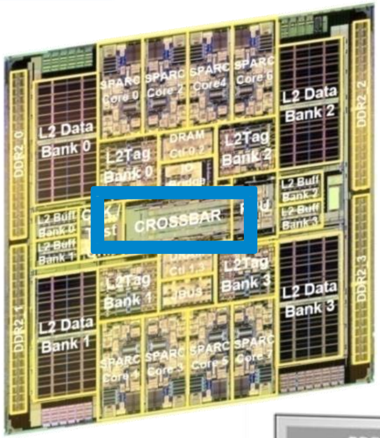
- Definition: **determines arrangement of channels and nodes in network**
  - Analogous to road map
- Often first step in network design
- Significant impact on network cost-performance
  - Determines number of hops
    - Latency
    - Network energy consumption
  - Implementation complexity
    - Node degree
    - Ease of layout

Typical Topologies in Processors

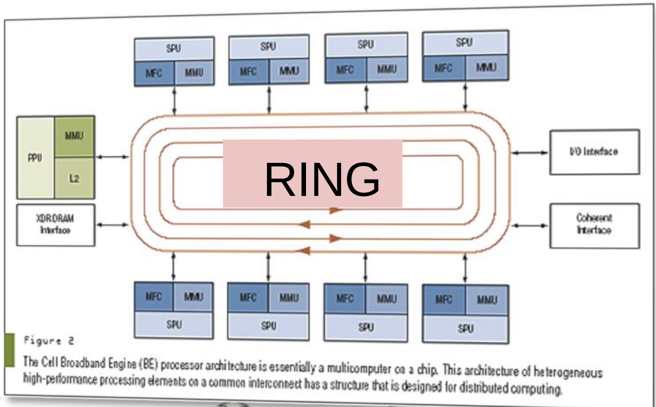
NIAGARA



CROSSBAR

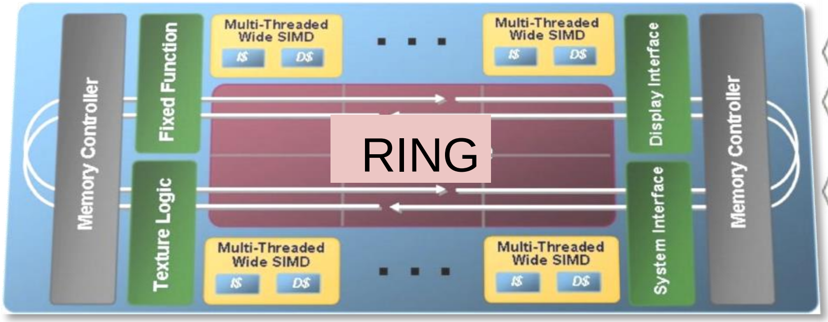


CELL EIB



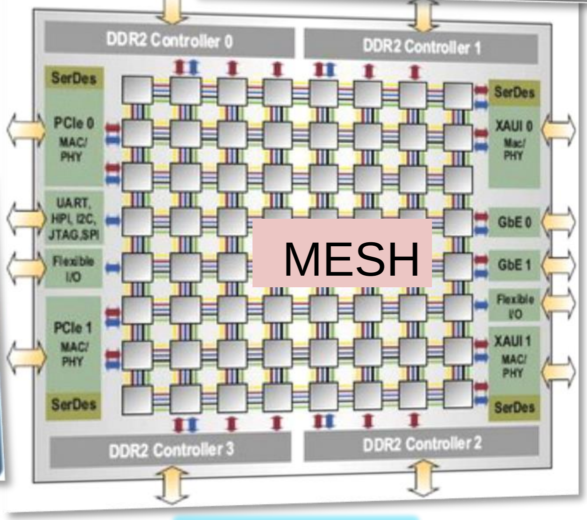
RING

Figure 2
The Cell Broadband Engine (BE) processor architecture is essentially a multicomputer on a chip. This architecture of heterogeneous high-performance processing elements on a common interconnect has a structure that is designed for distributed computing.



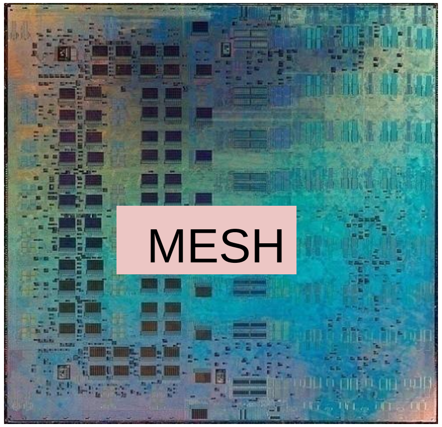
RING

LARABEE



MESH

TILERA



MESH

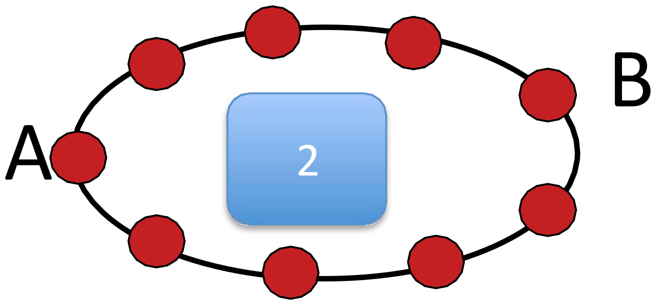
TRIPS

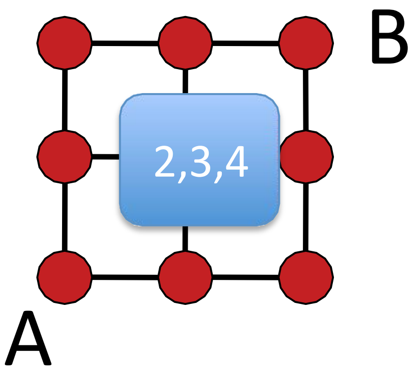
Network Metrics

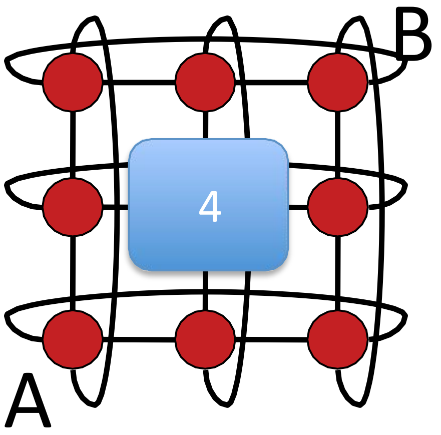
- Metrics are used to **evaluate the performance and cost** of network with given topology
- Also influenced by routing/flow control
  - At this stage, we skip the impacts
    - Assume ideal routing (perfect load balancing)
    - Assume ideal flow control (no idle cycles on any channel)

(1) Degree

- Switch Degree: **number of links at a node**
  - Proxy for estimating cost
    - › Higher degree requires more links and port counts at each router



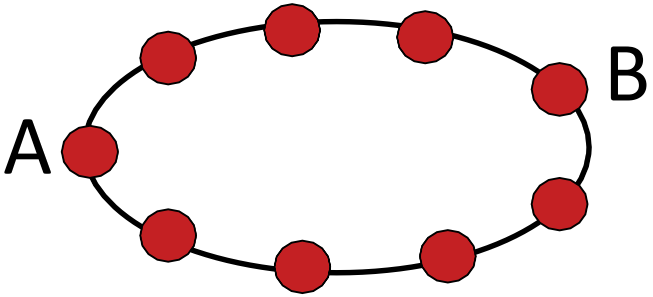




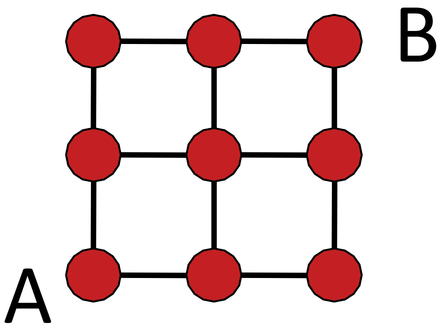
(2) Hop Count

- Path: ordered set of channels between source and destination
- Hop Count: number of hops a message takes from source to destination
  - **Simple, useful proxy for network latency**
    - Every node, link incurs some propagation delay even when no contention
- Minimal hop count: smallest hop count connecting two nodes
- Network diameter: largest min hop count in network
- Average minimum hop count: average across all src/dst pairs
  - Implementation may incorporate non-minimal paths
    - Increases average hop count

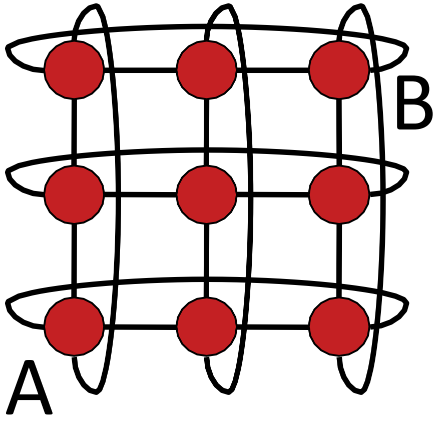
(2) Hop Count



Max = 4
Avg = 2.2



Max = 4
Avg = 1.77



Max = 2
Avg = 1.33

- Uniform random traffic
  - Ring > Mesh > Torus

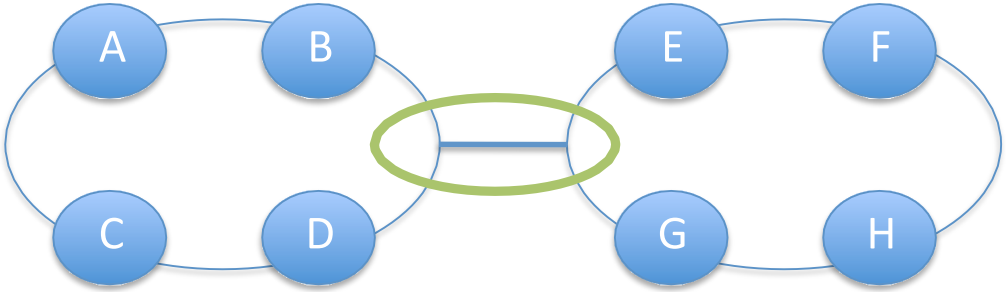
(3) Latency

- Time for packet to traverse network
  - Start: head arrives at input port
  - End: tail departs output port
- Latency = Head latency + serialization latency
  - Serialization latency: time for packet with Length L to cross channel with bandwidth b (L/b)
- Approximate with hop count
  - Other design choices (routing, flow control) impact latency
    - We skip these impacts now

(4) Maximum Channel Load

- Estimate max bandwidth the network can support
  - Max bits per second (bps) that can be injected by every node before it saturates
    - **Saturation: network cannot accept any more traffic**
  - Determine the most congested link
    - For given traffic pattern
    - Will limit overall network bandwidth
    - Estimate load on this channel

Maximum Channel Load Example

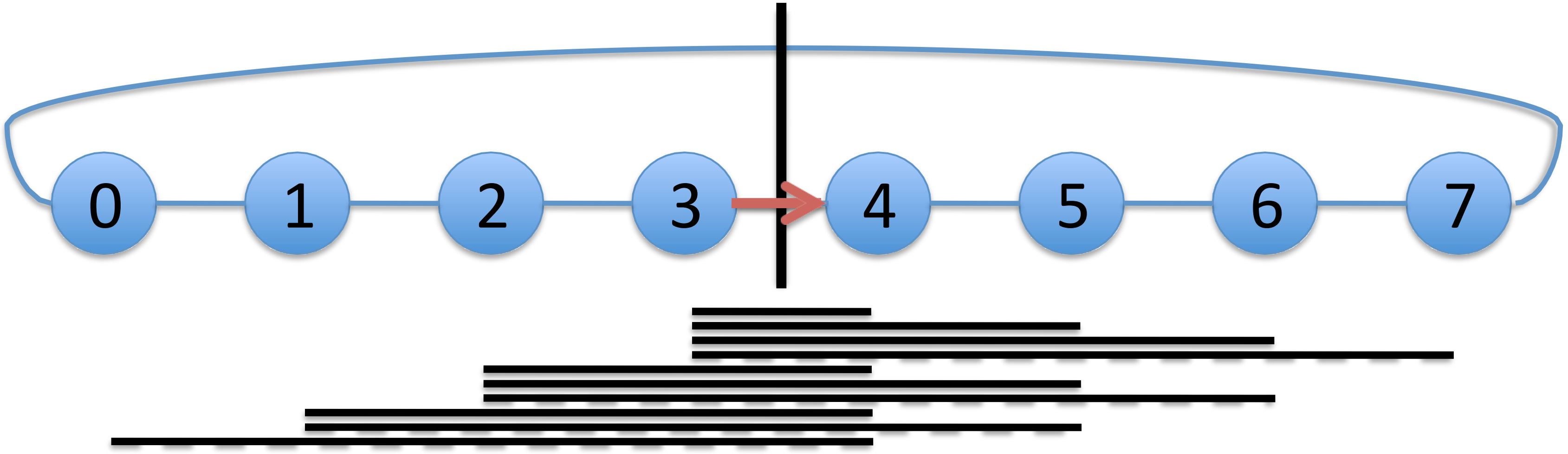


- Uniform random
  - Every node has equal probability of sending to every node
- Identify bottleneck channel
- Half of traffic from every node will cross bottleneck channel
  - 8 x ½ = 4
- Network saturates at ¼ injection bandwidth

(5) Bisection Bandwidth

- Common off-chip metric (片下指标)
  - Proxy for cost
- Amount of global wiring that will be necessary
  - Less useful for on-chip
    - Global on-chip wiring considered abundant
- Cuts: partition all the nodes into two disjoint sets
  - Bandwidth of a cut
- Bisection
  - A cut which divides all nodes into **nearly half**
  - Channel bisection ⟶ **min. channel count over all bisections**
  - Bisection bandwidth ⟶ **min. bandwidth over all bisections**
- With uniform random traffic
  - ½ of traffic crosses bisection

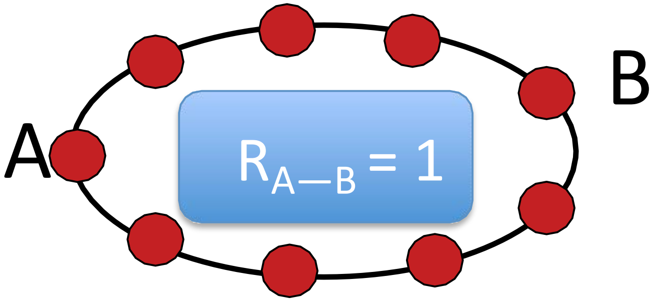
Throughput Example



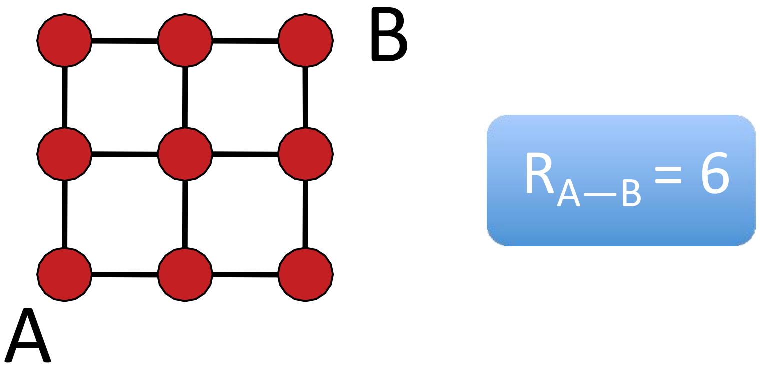
- Bisection = 4 (2 in each direction)
- With uniform random traffic
  - 3 sends 1/8 of its traffic to 4,5,6
  - 3 sends 1/16 of its traffic to 7 (2 possible shortest paths)
  - 2 sends 1/8 of its traffic to 4,5
  - Etc
- Suppose channel load = 1

(6) Path Diversity

- Multiple **shortest** paths (R) between source/destination pair
- **Fault tolerance**
- Better **load balancing** in network
- Routing algorithm should be able to **exploit** path diversity



$R_{A—B} = 1$



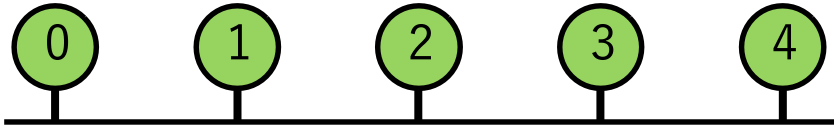
$R_{A—B} = 6$

Many possible network topologies

- Bus
- Crossbar
- Ring
- Tree
- Omega
- Hypercube
- Mesh
- Torus
- Butterfly
- ………..

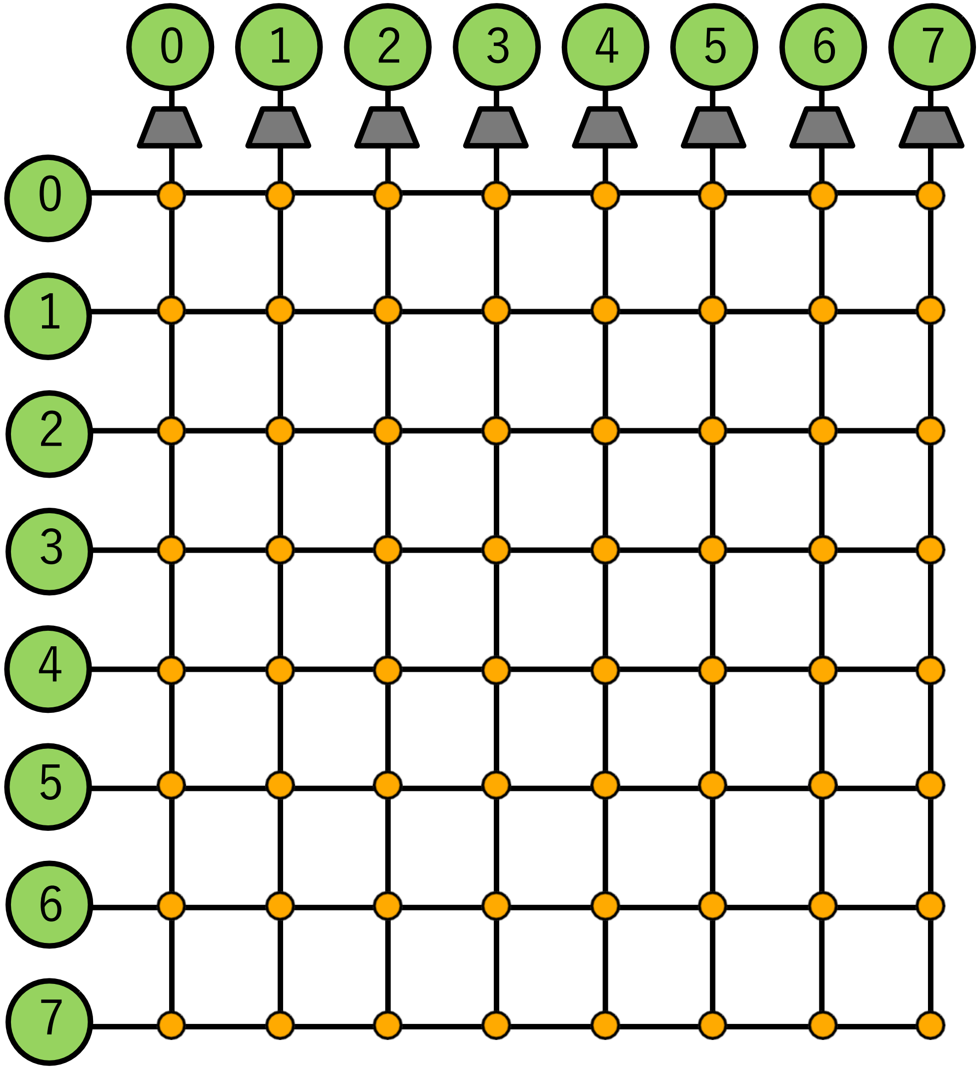
Bus interconnect

- Good:
  - Simple design
  - Cost effective for a small number of nodes
  - Easy to implement **coherence** (via snooping (侦听))
- Bad:
  - Contention: all nodes contend for shared bus
  - Limited bandwidth: all nodes communicate over same wires
  - High electrical load = low frequency, high power



Crossbar interconnect

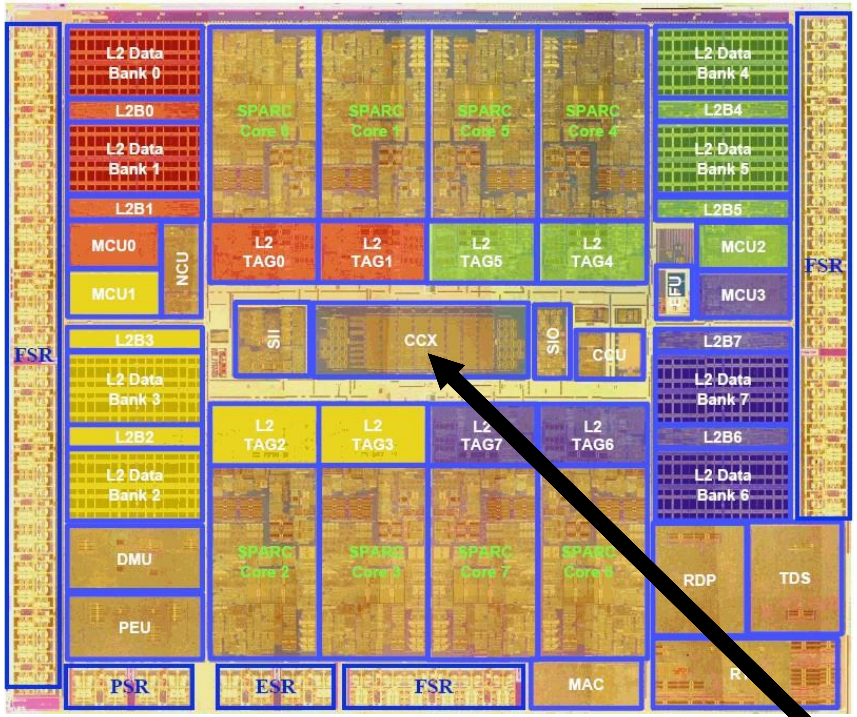
- Each node is connected to every other node
- Good:
  - O(1) latency and high bandwidth
- Bad:
  - Not scalable: $O(N^2)$ switches
  - High cost
  - Difficult to arbitrate at scale



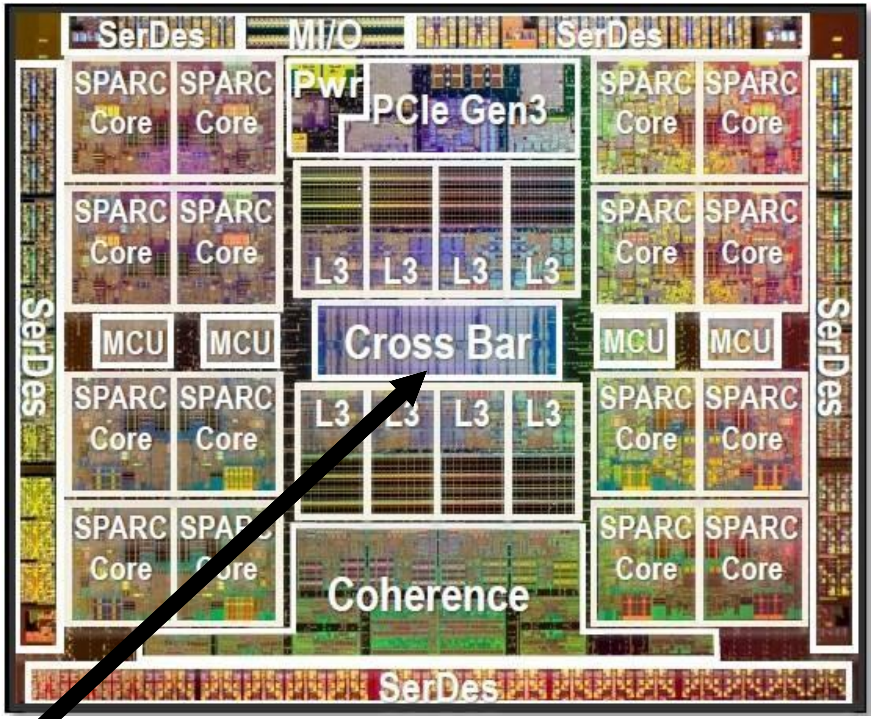
8-node crossbar network (N=8)

Note: in this network illustration, each node is drawn twice for clarity (at left and at top)

Crossbars used in Sun/Oracle processors



Sun SPARC T2 (8 cores, 8 L2 cache banks)

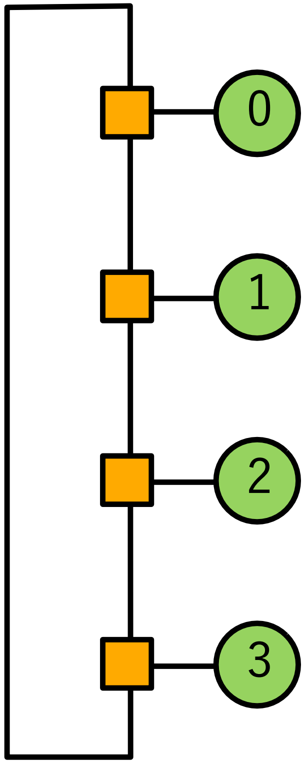


Oracle SPARC T5 (16 cores, 8 L3 cache banks)

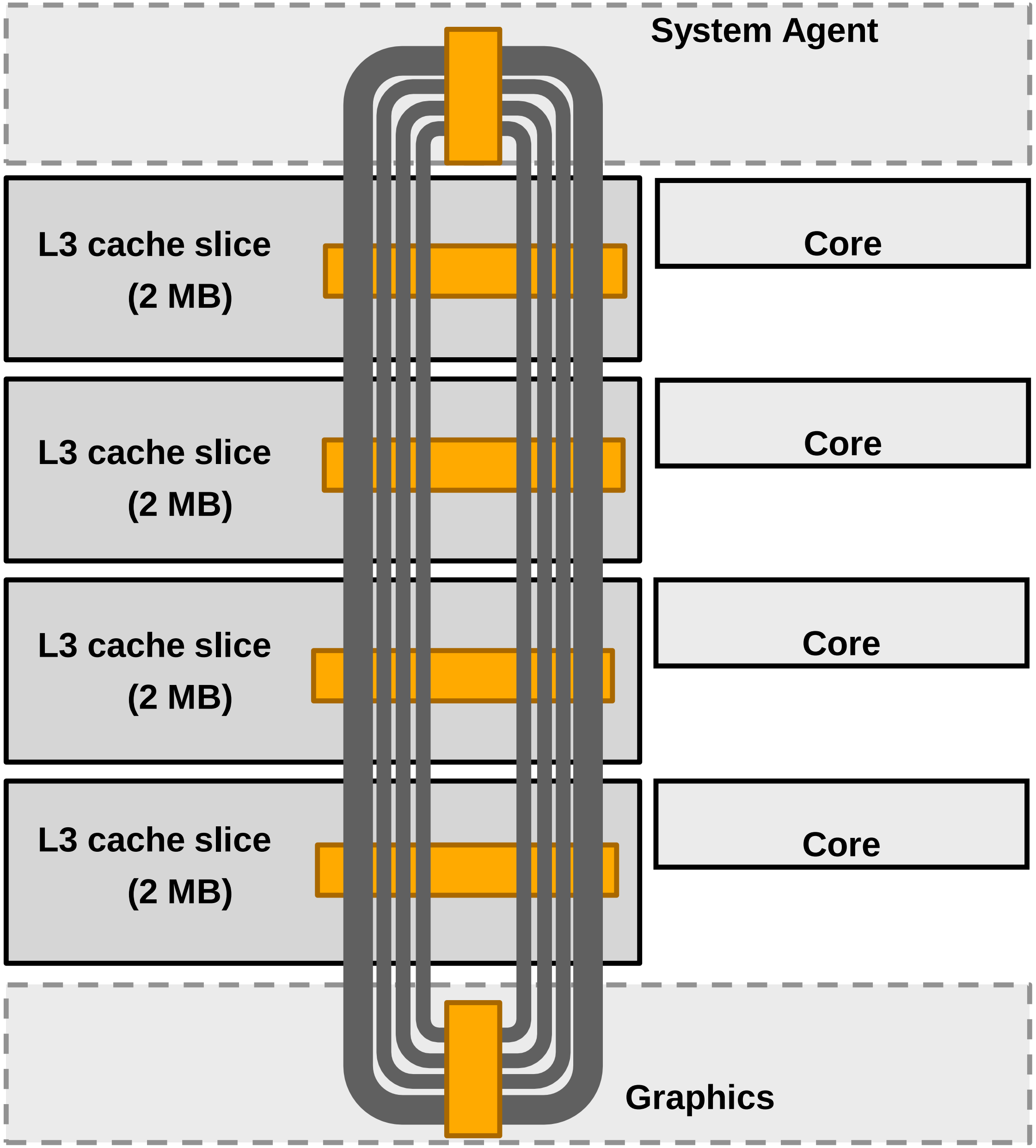
Note that crossbar (CCX) occupies about the same chip area as a core

Ring

- Good:
  - Simple
  - Cheap: O(N) cost
- Bad:
  - High latency: O(N)
  - Bisection bandwidth remains constant as nodes are added (scalability issue)
- Used in recent Intel architectures
  - Core i7, Xeon Phi
- Also used in IBM CELL Broadband Engine (9 cores)



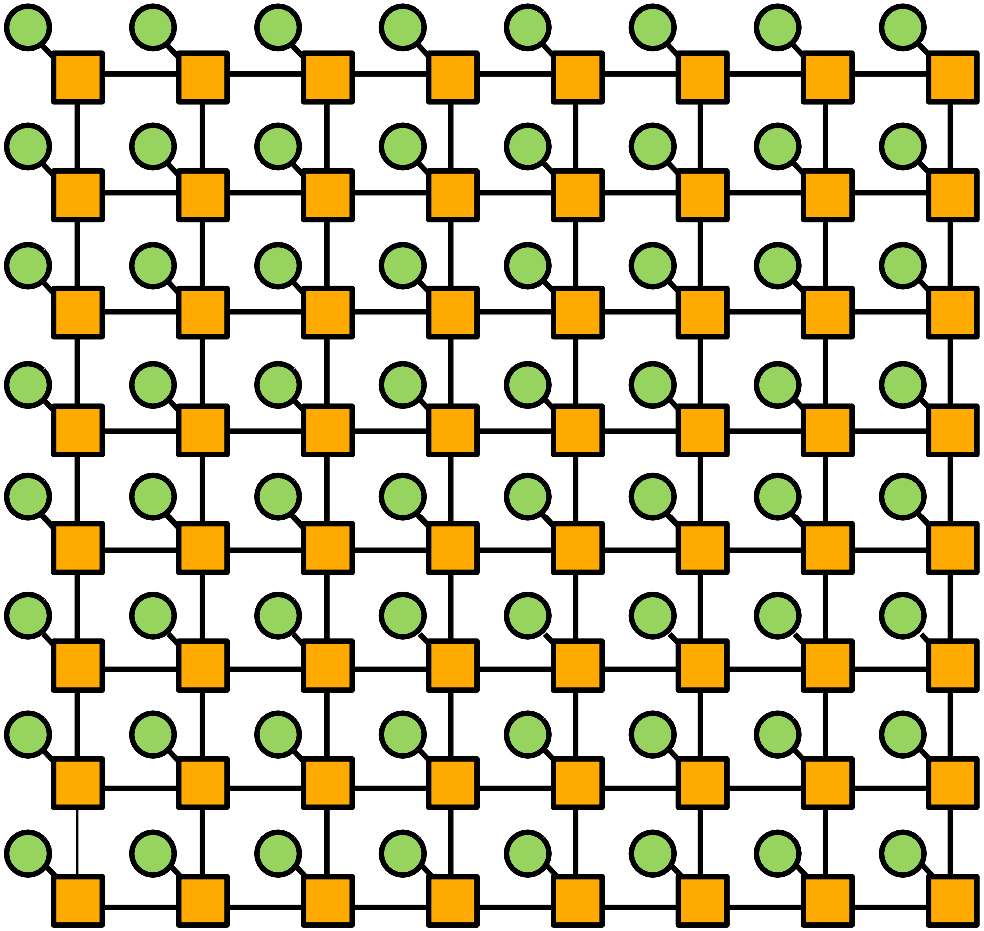
Intel’s ring interconnect (Sandy Bridge Microarch.)



- **Four rings**
  - request
  - snoop
  - ack
  - data (32 bytes)
- **Six interconnect nodes: four “slices” of L3 cache + system agent + graphics**
- **Each bank of L3 connected to ring bus twice**
- **Theoretical peak BW from cores to L3 at 3.4 GHz is approx. 435 GB/sec**
  - When each core is accessing its local slice

Mesh

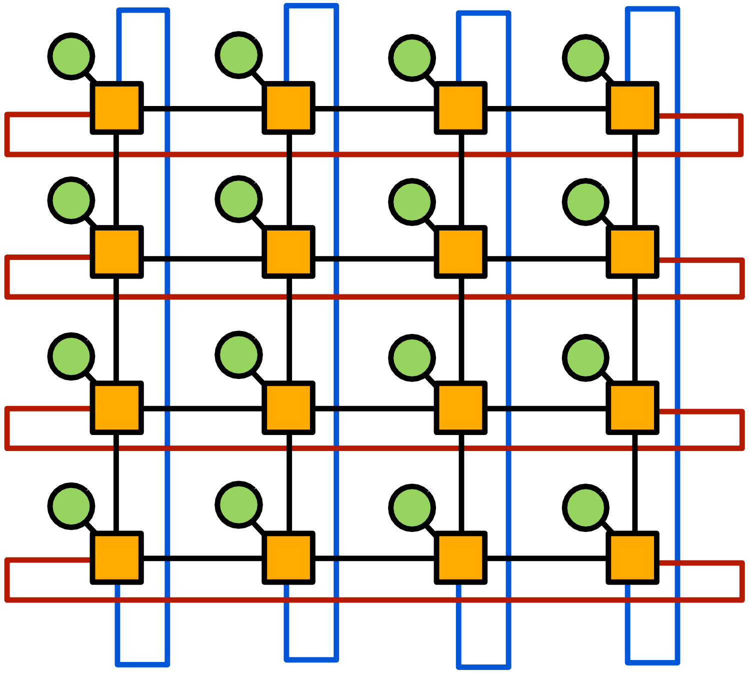
- Direct network
- O(N) cost
- Average latency:O(sqrt(N))
- Easy to lay out on chip: fixed-length links
- Path diversity: many ways for message to travel from one node to another
- Used by:
  - Tilera processors
  - Prototype Intel chips



2D Mesh

Torus

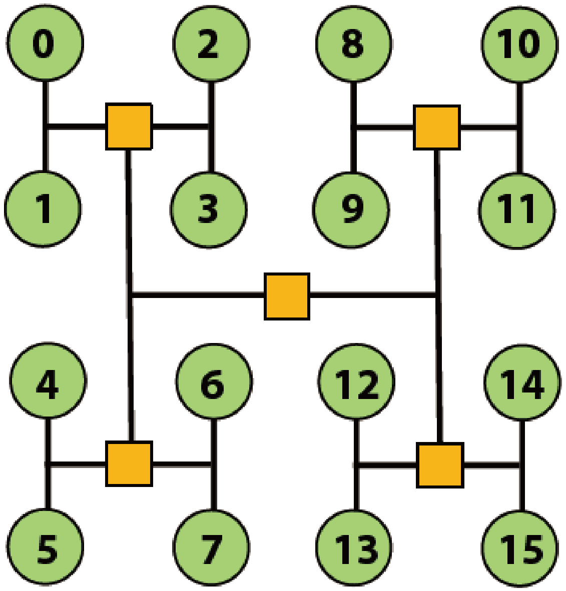
- In mesh topology, node near edge or in middle of network is different
  - Torus topology introduces new links to avoid this problem
- Still O(N) cost, but higher cost than 2D grid
- Higher path diversity and bisection BW than mesh
- Higher complexity
  - Difficult to layout on chip
  - Unequal link lengths



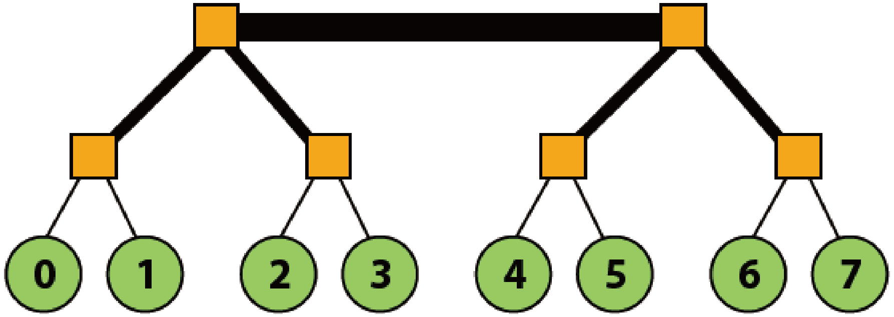
2D Torus

Trees

- Planar, hierarchical topology
- Like mesh/torus, good when traffic has locality
- Latency: O(lg N)
- Use “fat trees” to alleviate root bandwidth problem



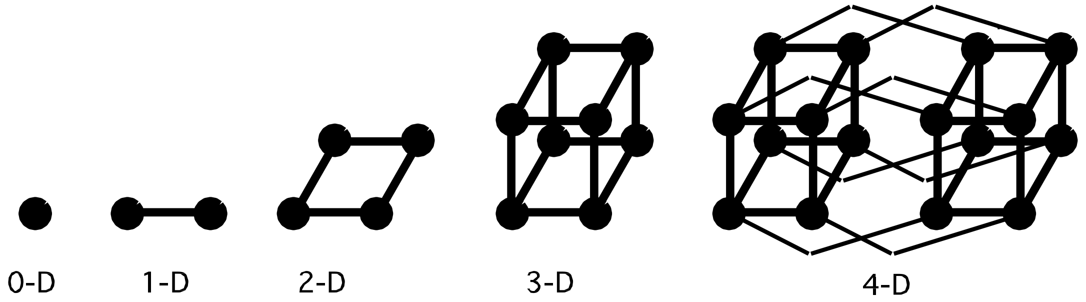
H-Tree



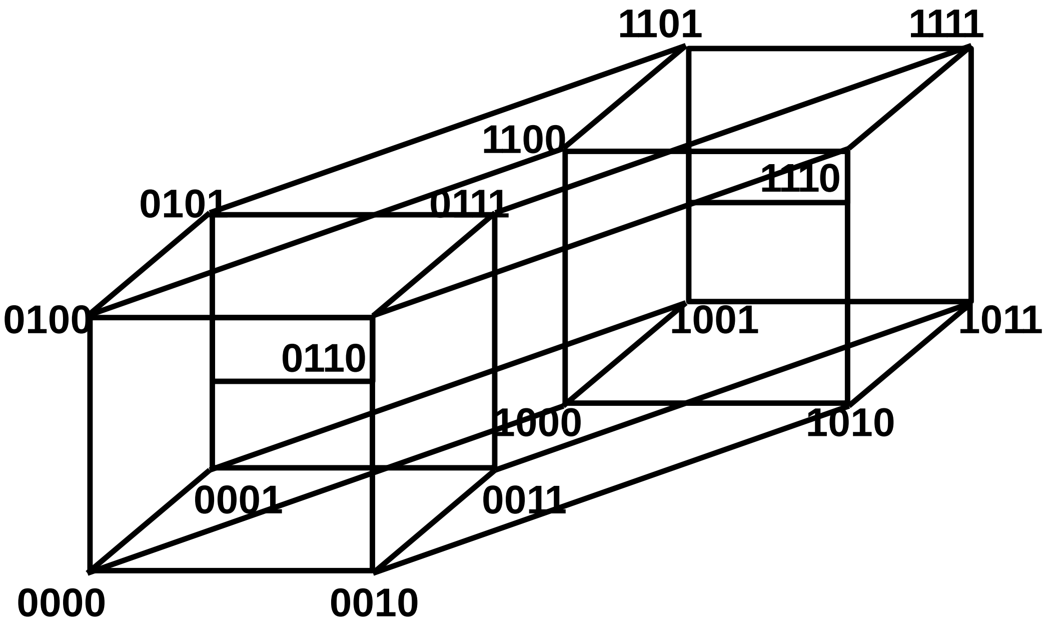
Fat Tree

Hyper cube in different dimension

- Low latency: O(lg N)
- Radix: O(lg N)
- Number of links O(N lg N)

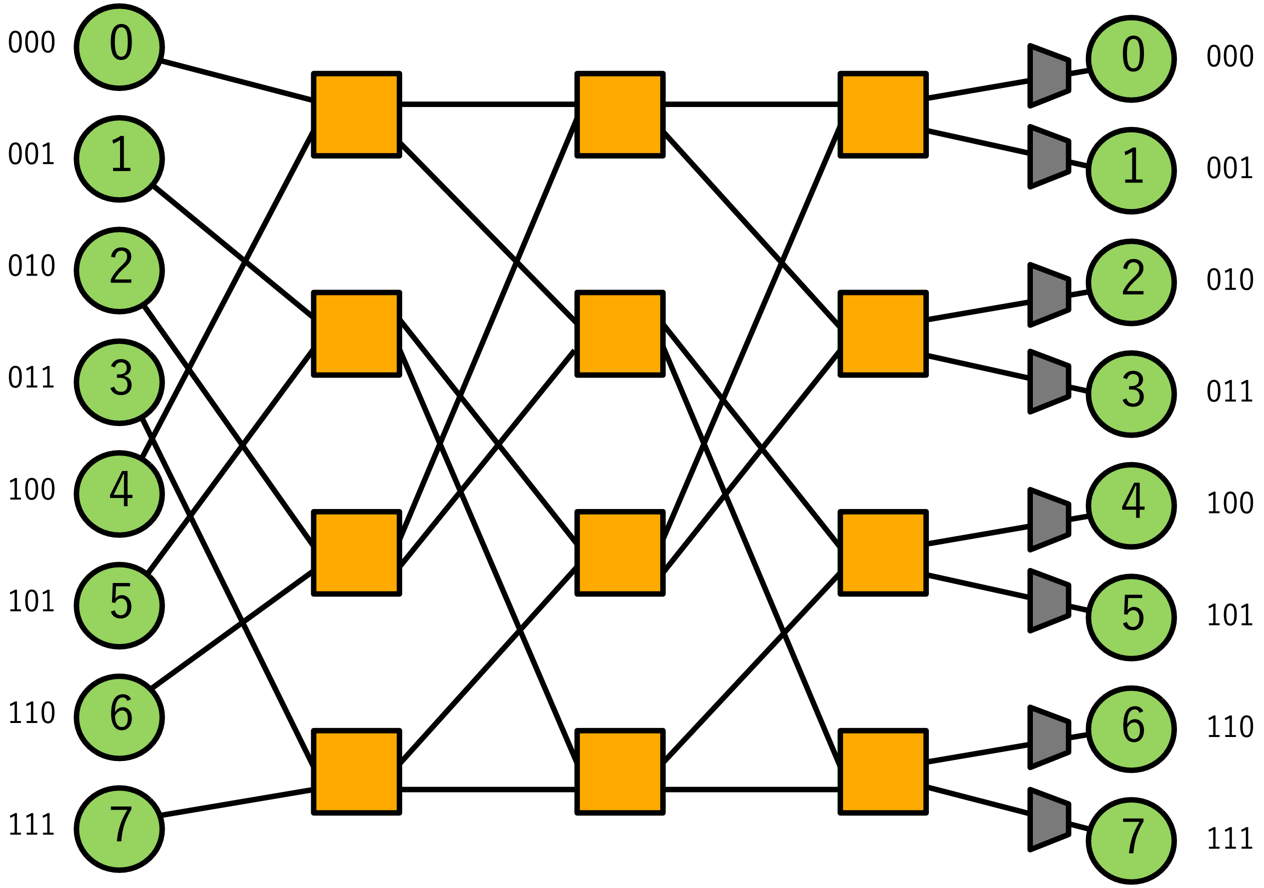


- 6D hypercube used in 64-core Cosmic Cube computer developed at Caltech in the 80s.



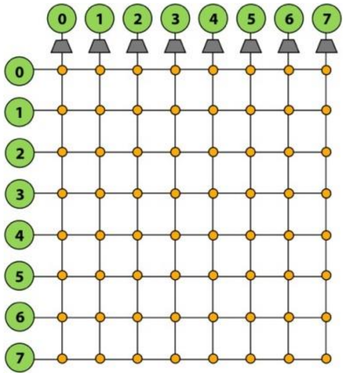
Multi-stage logarithmic

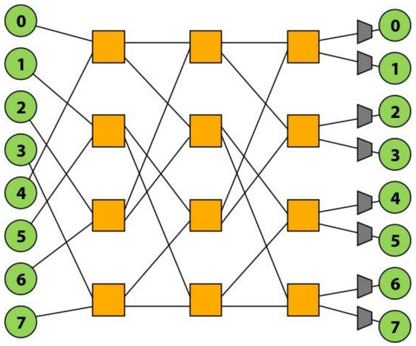
- Indirect network with multiple switches between terminals
  - Cost: O(N lg N)
  - Latency: O(lg N)
- Many variations: Omega, butterfly, Clos networks, etc…

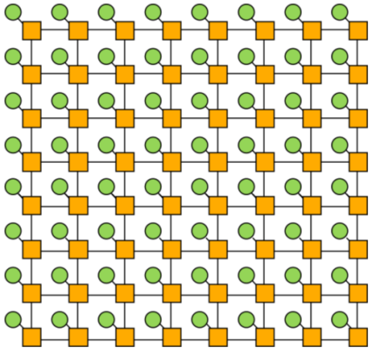


Omega Network

Review: network topologies







Topology	Crossbar	Multi-stage log.	Mesh
Direct/Indirect	Indirect	Indirect	Direct
Blocking/ Non-blocking	Non-blocking	Blocking	Blocking
Cost	$O(N^2)$	O(N lg N)	O(N)
Latency	O(1)	O(lg N)	O(sqrt(N)) (average)

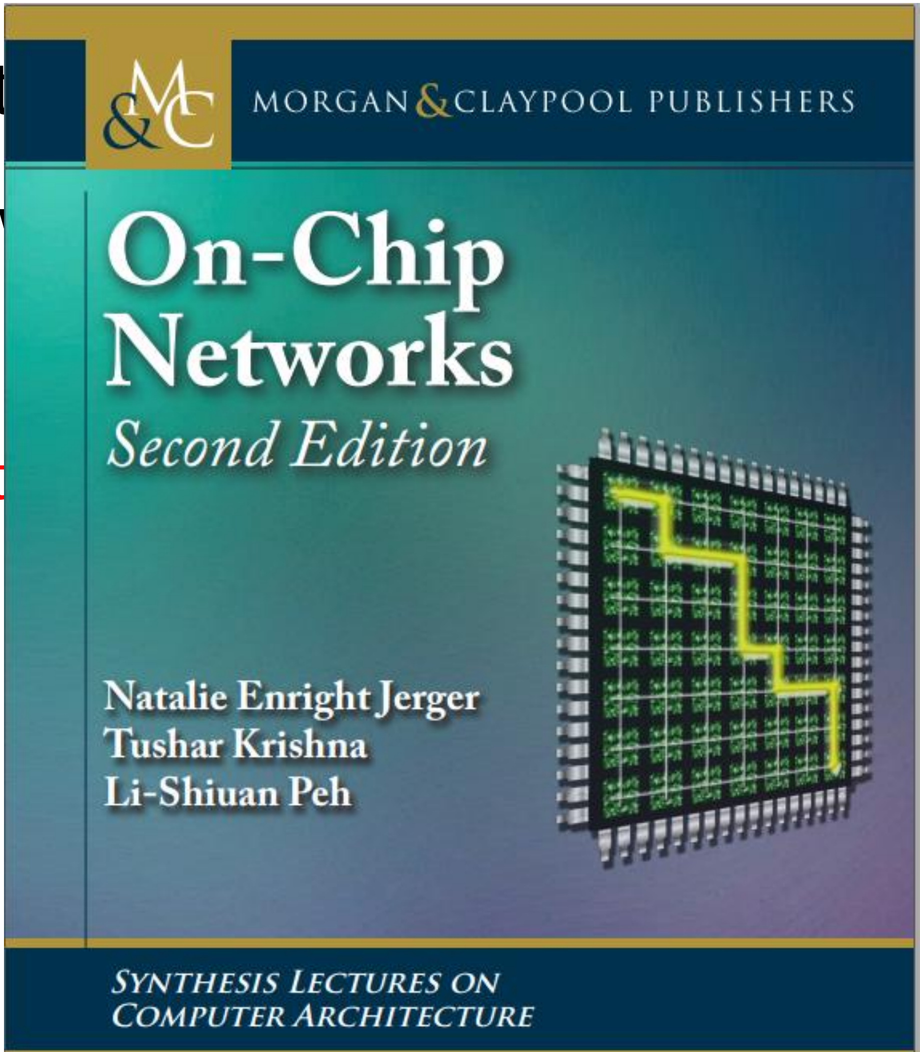
Outline

- Introduction
- Network Topology
- **Flow Control**
- Example IN

Flow Control Overview

- Topology: determines connectivity of network
- **Routing: determines paths through network**
- Flow Control: determine allocat

  messages as they traverse net
  - Buffers and links
  - Significant impact on throughput



MORGAN & CLAYPOOL PUBLISHERS

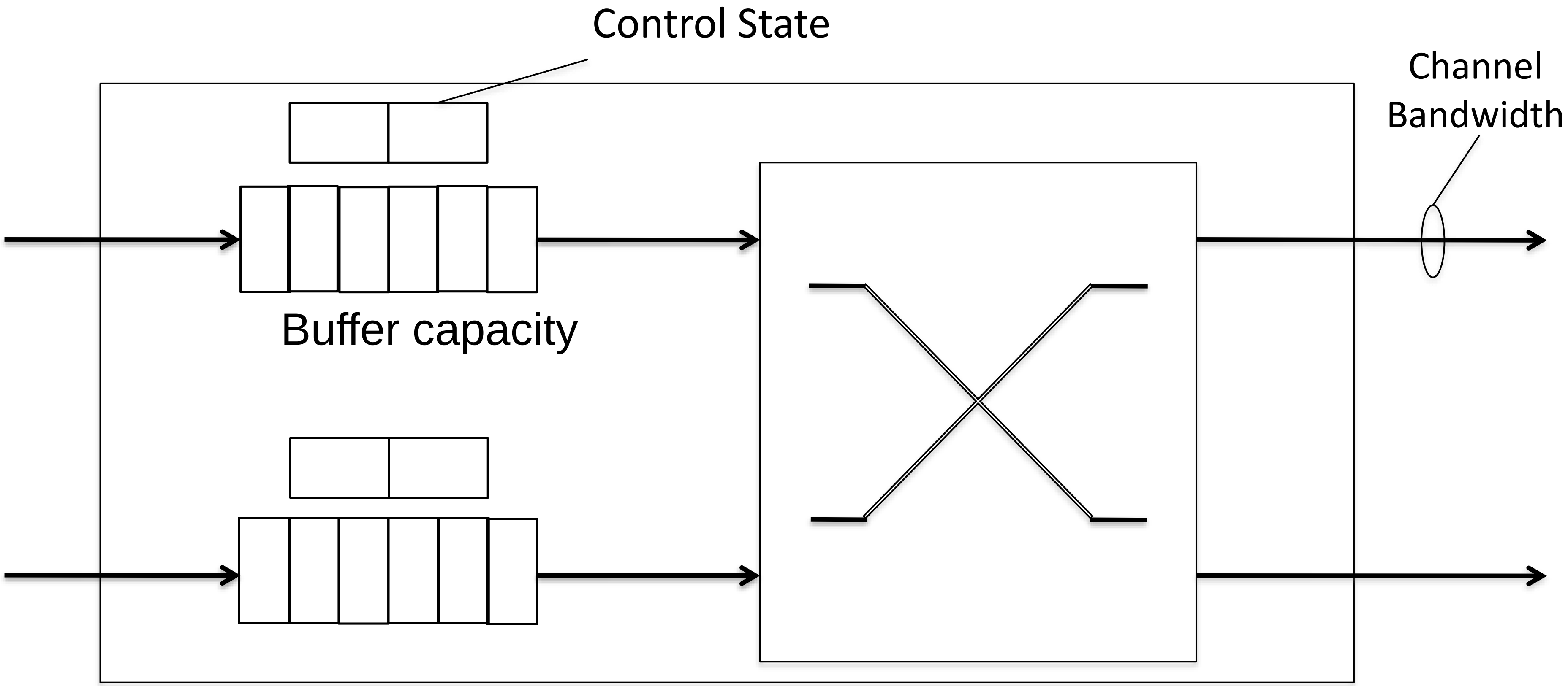
On-Chip Networks

Second Edition

Natalie Enright Jerger
Tushar Krishna
Li-Shiuan Peh

SYNTHESIS LECTURES ON COMPUTER ARCHITECTURE

Flow Control

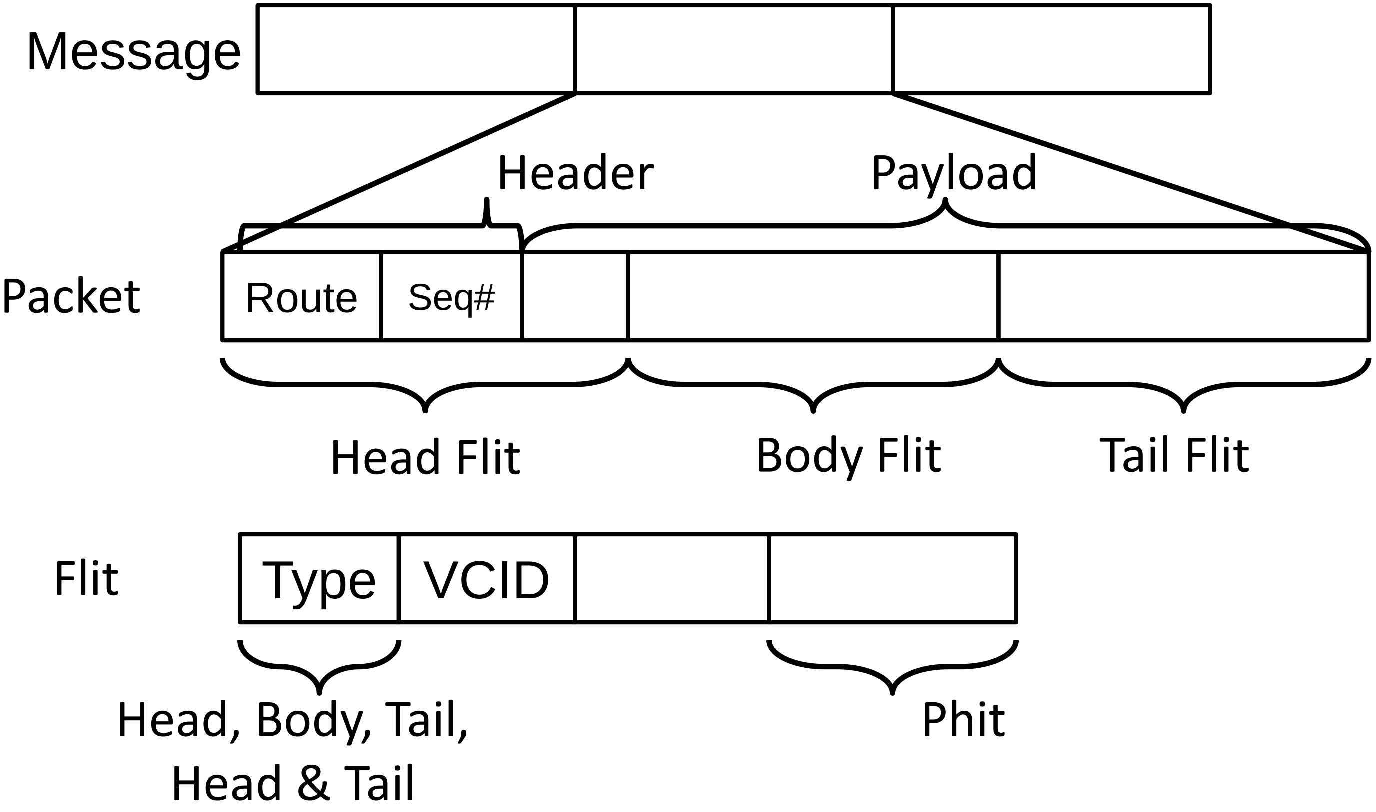


- Control state records
  - allocation of channels and buffers to packets
  - current state of packet traversing node
- Channel bandwidth advances flits along nodes
- Buffers hold flits waiting for channel bandwidth

Packets (1)

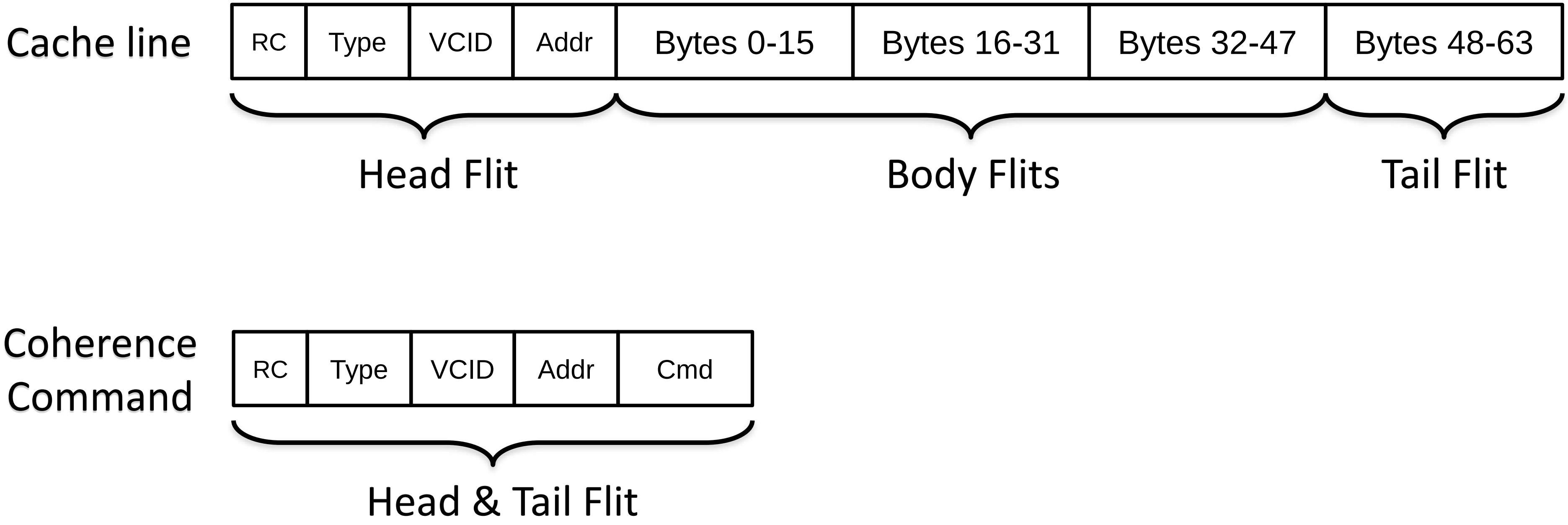
- Messages: composed of one or more packets
  - If message size is <= maximum packet size only one packet created
- Packets: composed of one or more flits
- Flit: flow control digit
- Phit: physical digit
  - Subdivides flit into chunks = to link width

Packets (2)



- Off-chip: channel width limited by pins
  - Requires phits
- On-chip: **abundant wiring means phit size == flit size**

Packets(3)



Cache line

RC	Type	VCID	Addr	Bytes 0-15	Bytes 16-31	Bytes 32-47	Bytes 48-63

Head Flit — Body Flits — Tail Flit

Coherence Command

RC	Type	VCID	Addr	Cmd

Head & Tail Flit

- Packet contains destination/route information
  - Flits may not → **all flits of a packet must take same route**

Switching

- Different flow control techniques based on **granularity**
  - Message-based: allocation made at message granularity (circuit-switching)
  - Packet-based: allocation made to whole packets
  - Flit-based: allocation made on a flit-by-flit basis

Message-Based Flow Control

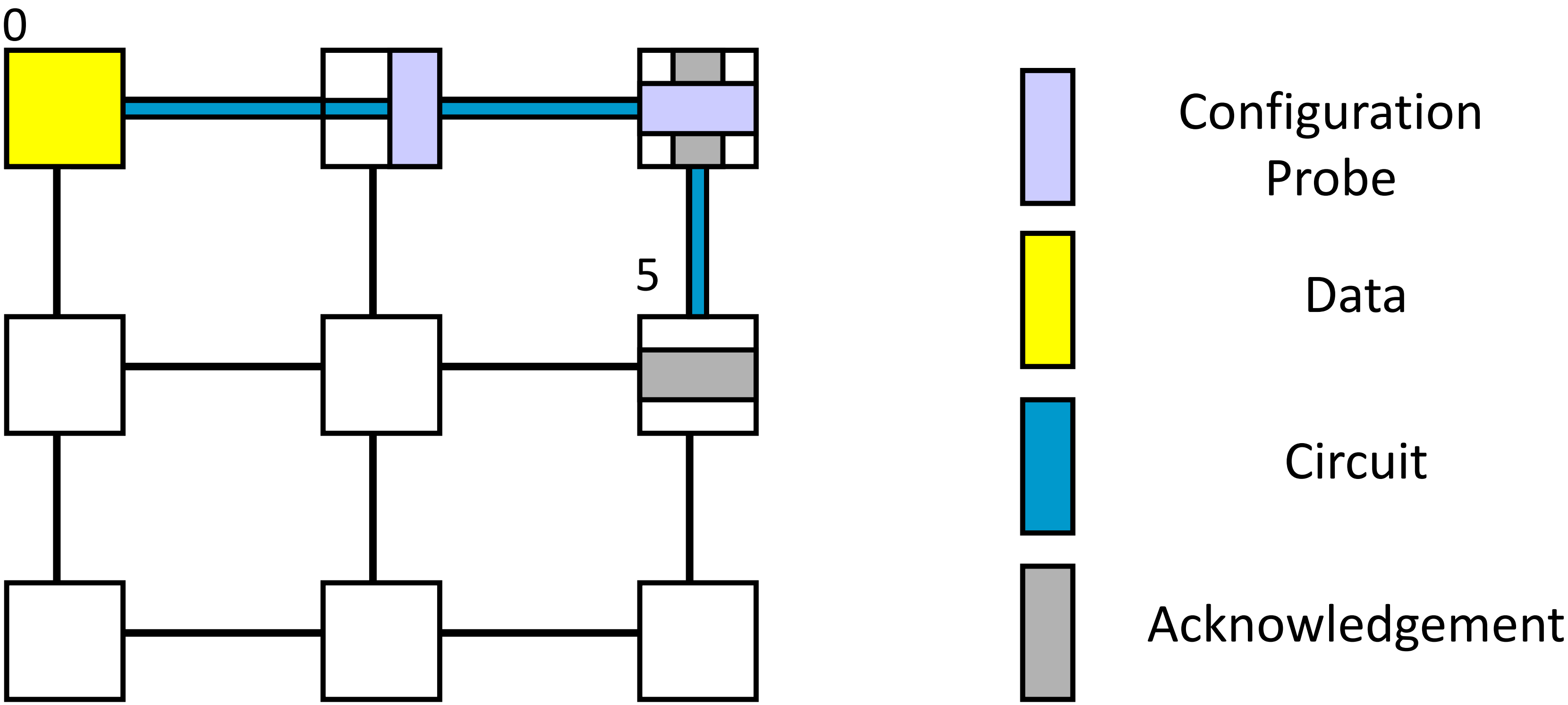
- Coarsest granularity

- Circuit-switching
  - Pre-allocates resources across multiple hops
    - Source to destination
    - Resources = links
    - Buffers are not necessary
  - Probe sent into network to reserve resources

Circuit Switching

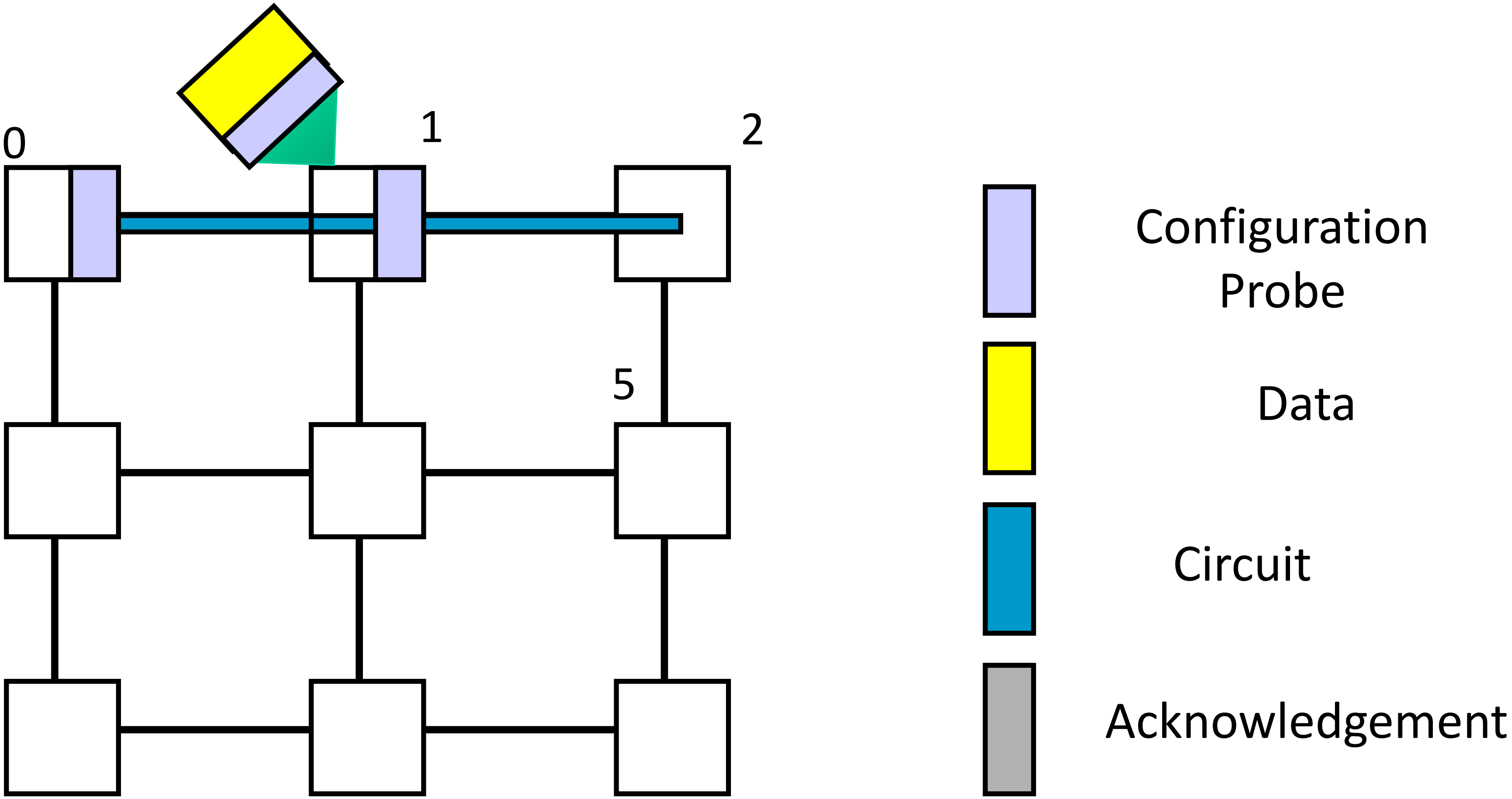
- Once probe sets up circuit
  - Message does not need to perform any routing or allocation at each network hop
  - Good for transferring large amounts of data
    - Can amortize circuit setup cost by sending data with very low per-hop overheads
- No other message can use those resources until transfer is complete
  - Throughput can suffer due to setup and hold time for circuits
  - Links are idle until setup is complete

Circuit Switching Example



- Significant latency overhead prior to data transfer
  - Data transfer does not pay per-hop overhead for routing and allocation

Circuit Switching Example (2)



- When there is contention
  - Significant wait time
  - Message from 1 → 2 must wait

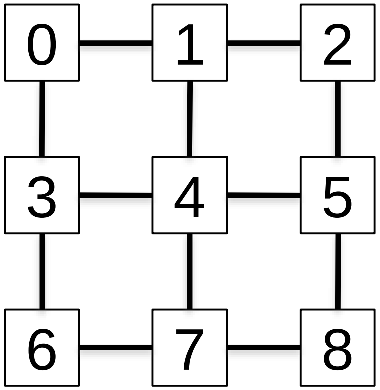
Time-Space Diagram: Circuit-Switching

Location	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
0	S_0 8	▨	▨	▨	▨	▨	▨	▨	▨	A_0 8	D_0 8	D_0 8	▨	D_0 8	D_0 8	▨	T_0 8				
1		S_0 8	▨	▨	▨	▨	▨	▨	A_0 8	▨	▨	D_0 8	D_0 8	▨	D_0 8	D_0 8	▨	T_0 8			
2			S_0 8	▨	▨	S_2 8	▨	A_0 8	▨	▨	▨	▨	D_0 8	D_0 8	▨	D_0 8	D_0 8	▨	T_0 8	S_2 8	▨
5				S_0 8	▨	▨	A_0 8	▨	▨	▨	▨	▨	▨	D_0 8	D_0 8	▨	D_0 8	D_0 8	▨	T_0 8	S_2 8
8					S_0 8	A_0 8	▨	▨	▨	▨	▨	▨	▨	▨	D_0 8	D_0 8	▨	D_0 8	D_0 8	▨	T_0 8

Time

Time to setup+ack circuit from 0 to 8

Time setup from 2 to 8 is **blocked**



Packet-based Flow Control

- Break messages into packets

- Interleave packets on links
  - Better resource utilization

- Requires per-node buffering to store in-flight packets

- Two types of packet-based techniques

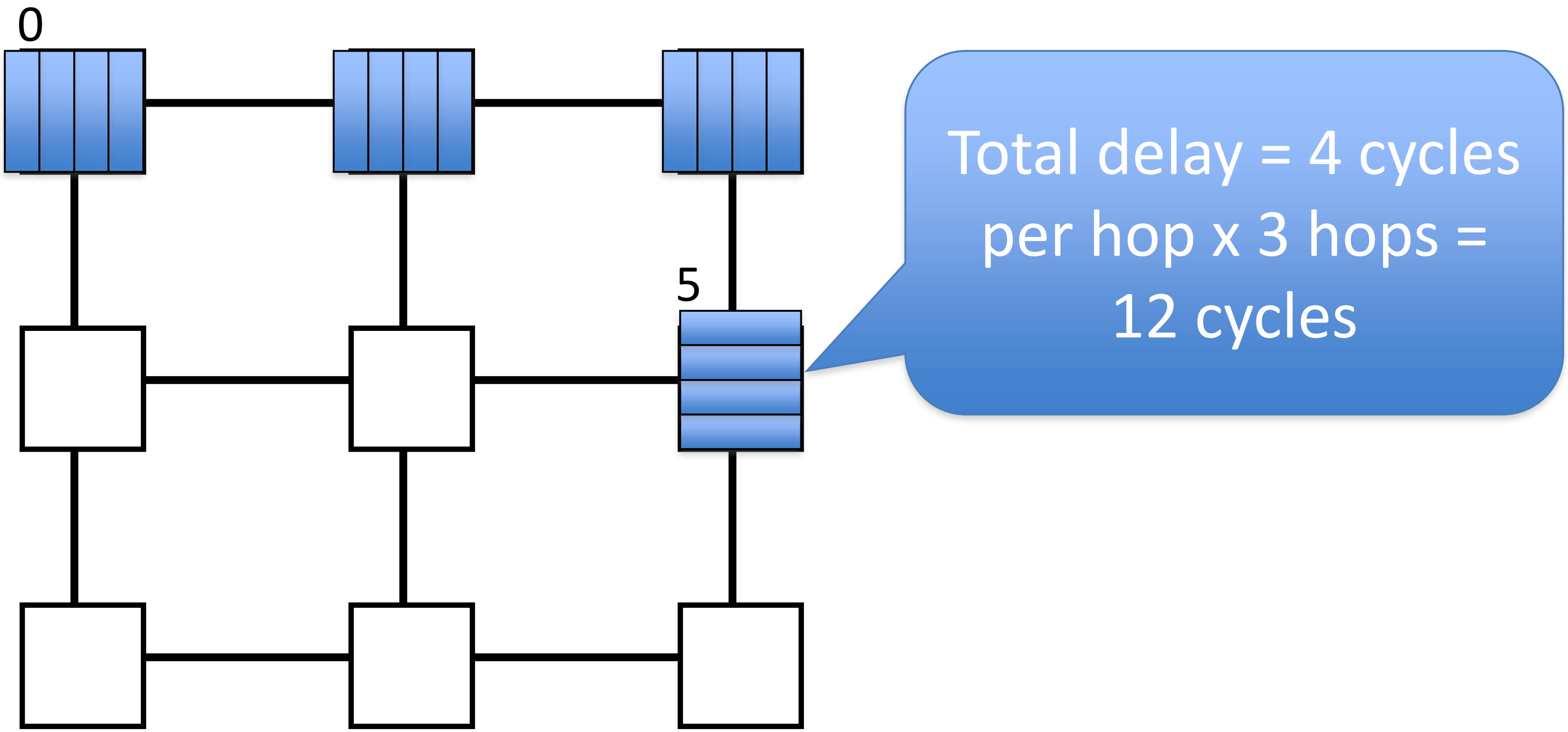
Store and Forward

- Links and buffers are allocated to entire packet

- Head flit waits at router until entire packet is received before being forwarded to the next hop

- Not suitable for on-chip
  - Requires buffering at each router to hold entire packet
    - Packet cannot traverse link until buffering allocated to entire packet
  - **Incurs high latencies (pays serialization latency at each hop)**

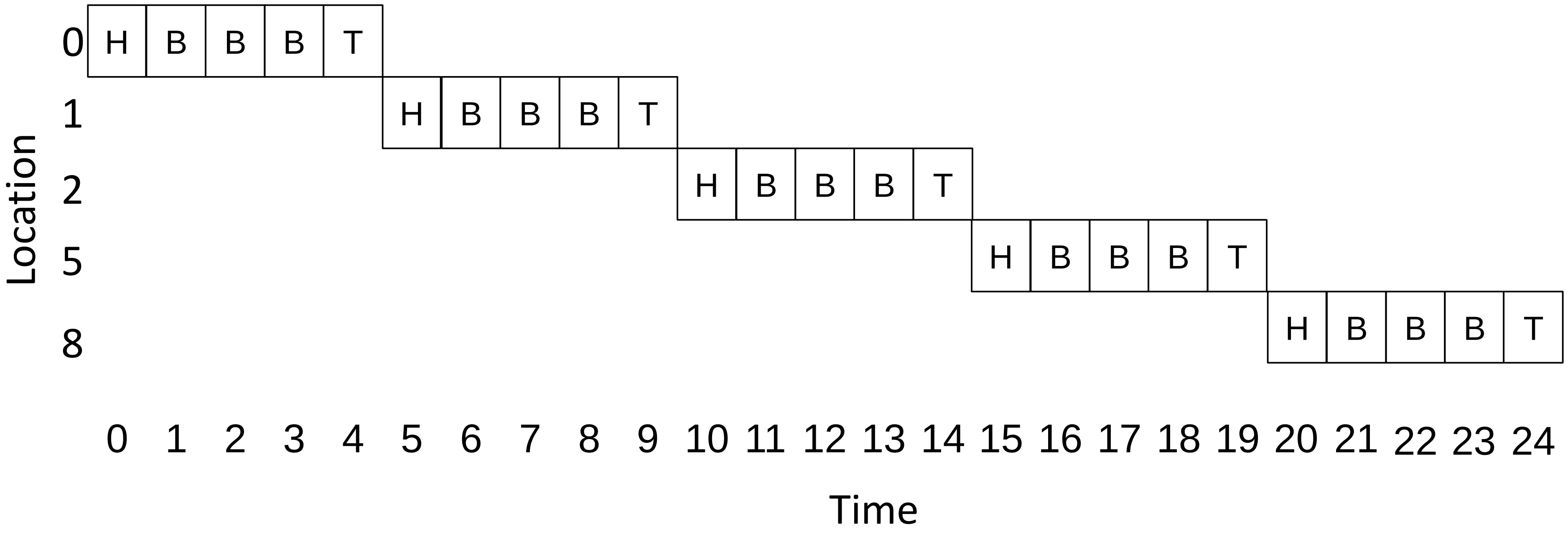
Store and Forward Example



Total delay = 4 cycles per hop x 3 hops = 12 cycles

- High per-hop latency
  - Serialization delay paid at each hop
- Larger buffering required

Time-Space Diagram: Store and Forward



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
0	H	B	B	B	T																				
1						H	B	B	B	T															
2											H	B	B	B	T										
5																H	B	B	B	T					
8																					H	B	B	B	T

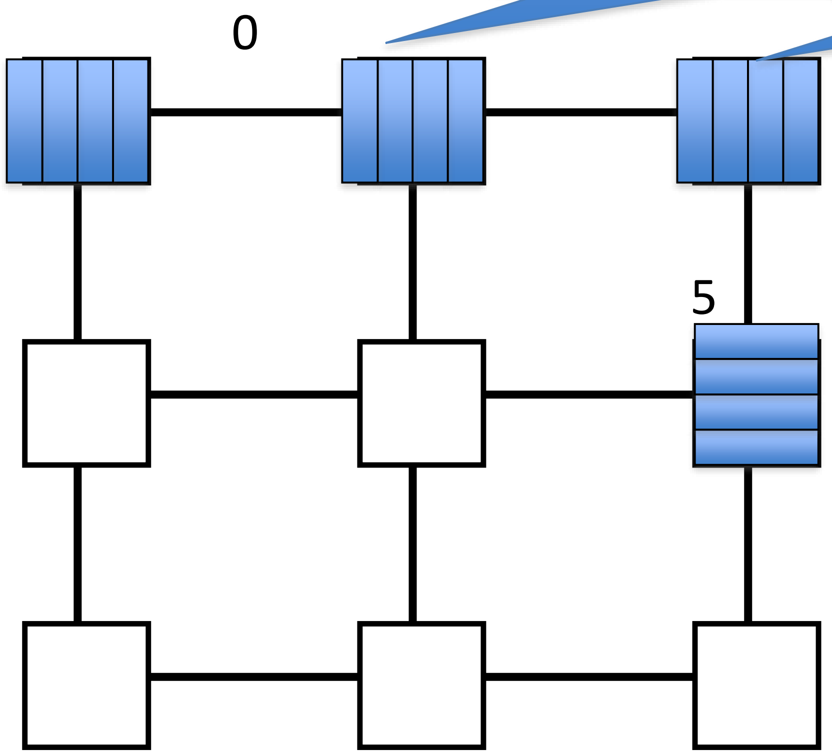
Packet-based: Virtual Cut Through

- Links and Buffers allocated to entire packets
- Flits can proceed to next hop before tail flit has been received by current router
  - But only if next router has enough buffer space for entire packet
- Reduces the latency significantly compared to SAF
- But **still requires large buffers**
  - Unsuitable for on-chip

Virtual Cut Throu

Allocate 4 f
buffers bef
proce

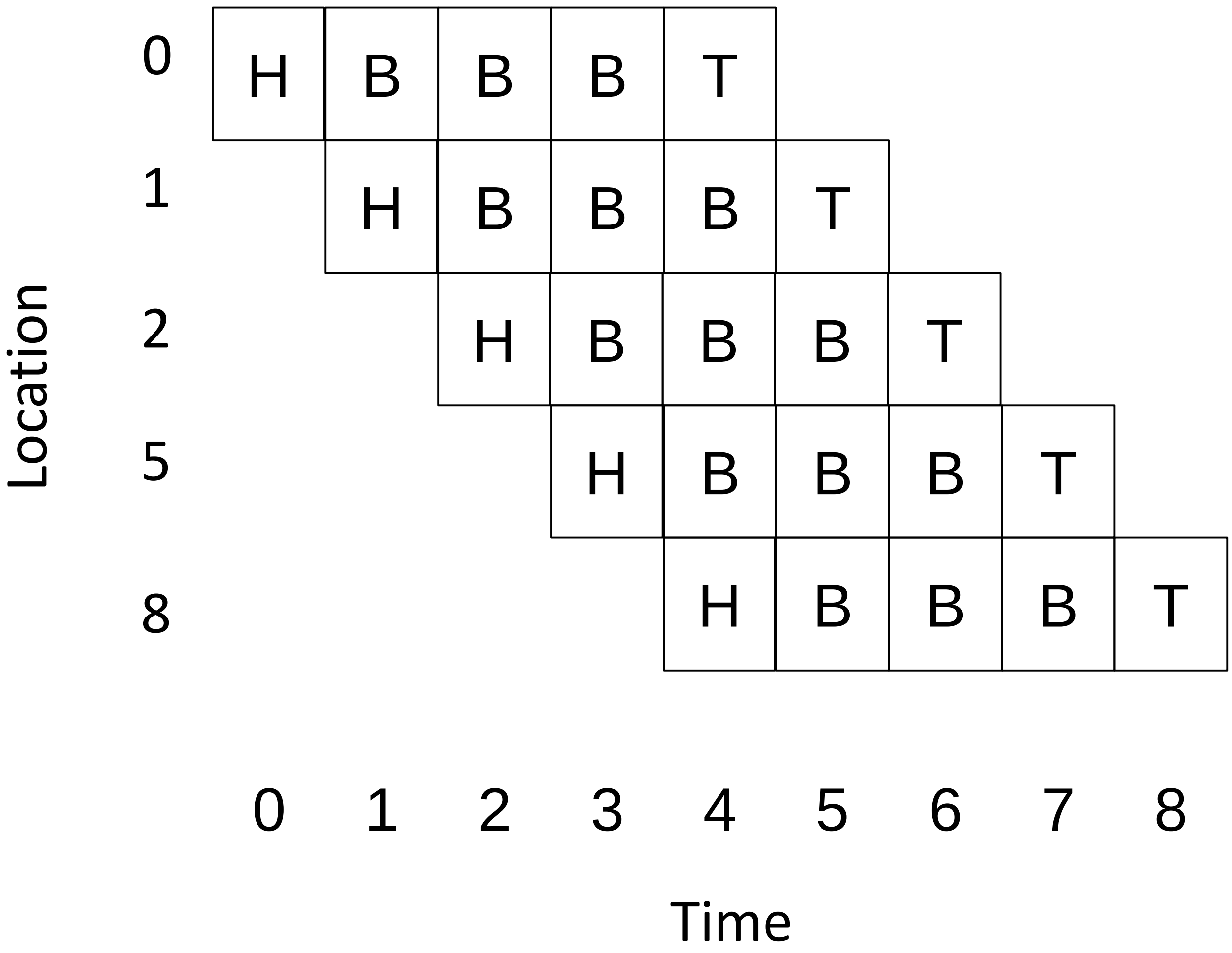
Allocate 4 flit-sized buffers before head proceeds



Total delay = 1 cycle per hop x 3 hops + serialization = 6 cycles

- Lower per-hop latency
- Large buffering required

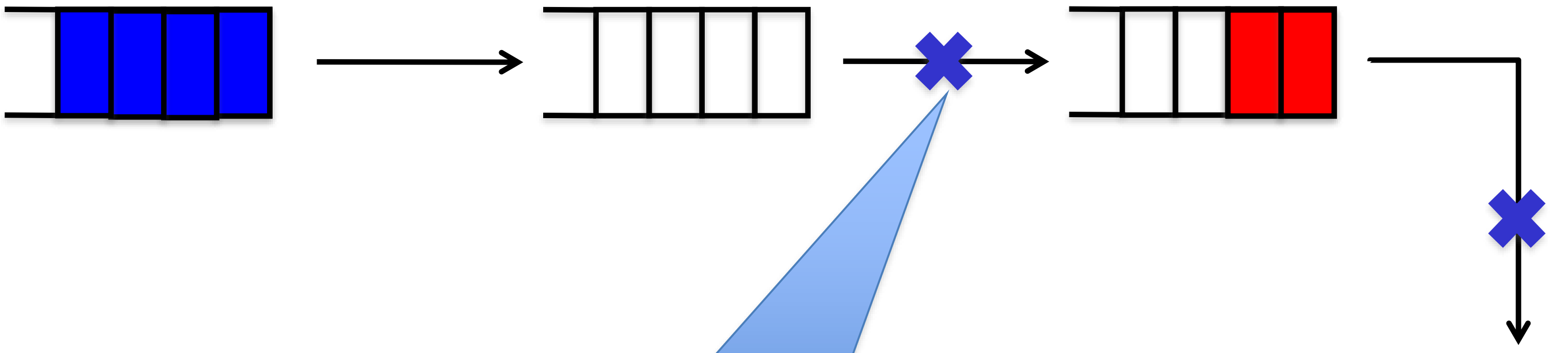
Time-Space Diagram: VCT



Location									
0	H	B	B	B	T				
1		H	B	B	B	T			
2			H	B	B	B	T		
5				H	B	B	B	T	
8					H	B	B	B	T
	0	1	2	3	4	5	6	7	8

Time

Virtual Cut Through



Cannot proceed because only 2 flit buffers available

- Throughput suffers from inefficient buffer allocation

Time-Space Diagram: VCT (2)

Location	0	1	2	3	4	5	6	7	8	9	10	11
0	H	B	B	B	T							
1		H	B	B	B	T						
2			H	B	B	B	T					
5				Insufficient Buffers			H	B	B	B	T	
8								H	B	B	B	T

Time

Flit-Level Flow Control

- Help routers meet tight area/power constraints

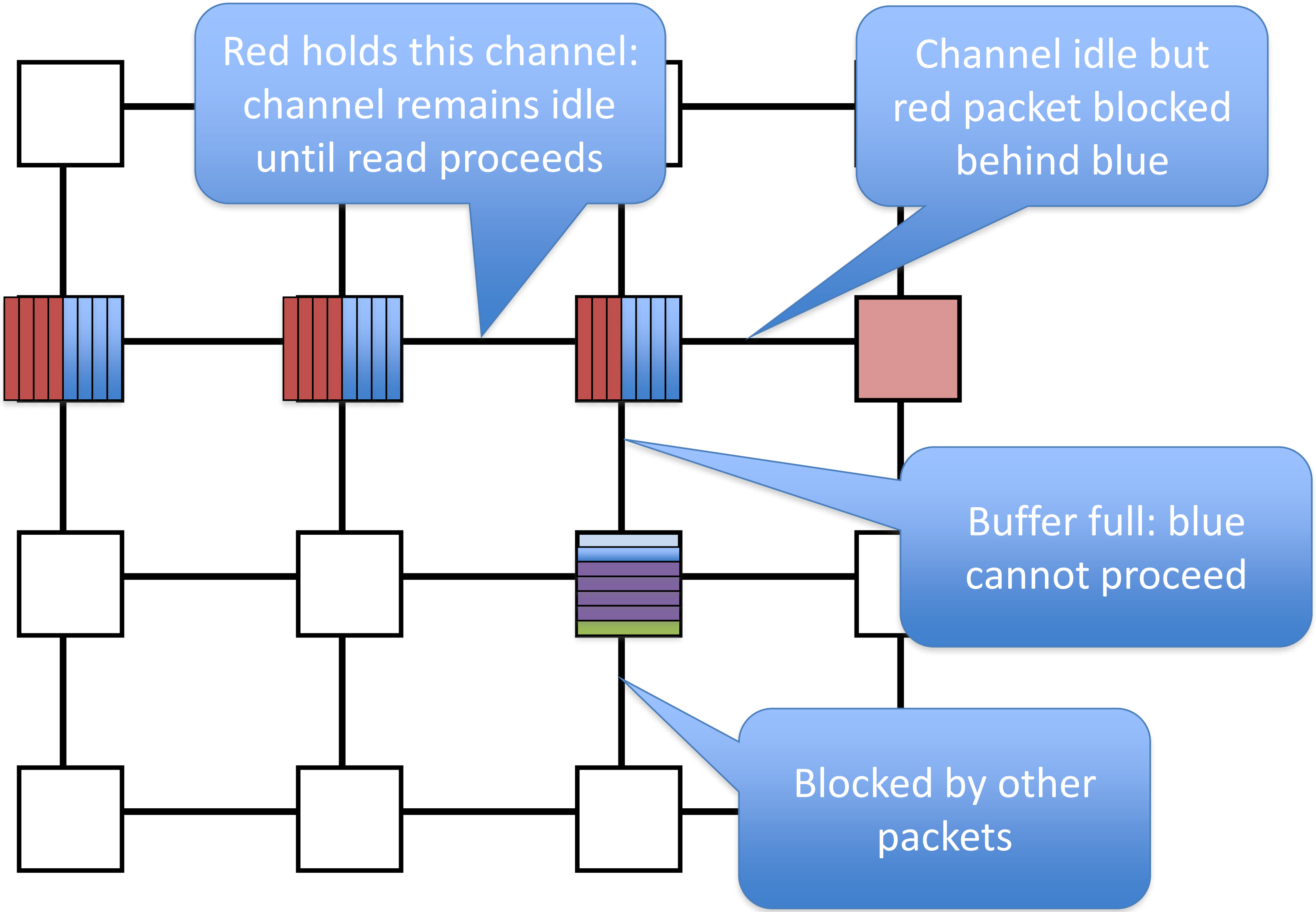
- Flit can proceed to next router when there is buffer space available for that flit
  - Improves over SAF and VCT by allocating buffers on a flit-by-flit basis

Wormhole Flow Control

- Pros
  - More efficient buffer utilization (good for on-chip)
  - Low latency

- Cons
  - Poor link utilization: if head flit becomes blocked, all links spanning length of packet are idle
    - Cannot be re-allocated to different packet
    - Suffers from head of line (HOL) blocking

Wormhole Example



Red holds this channel: channel remains idle until read proceeds

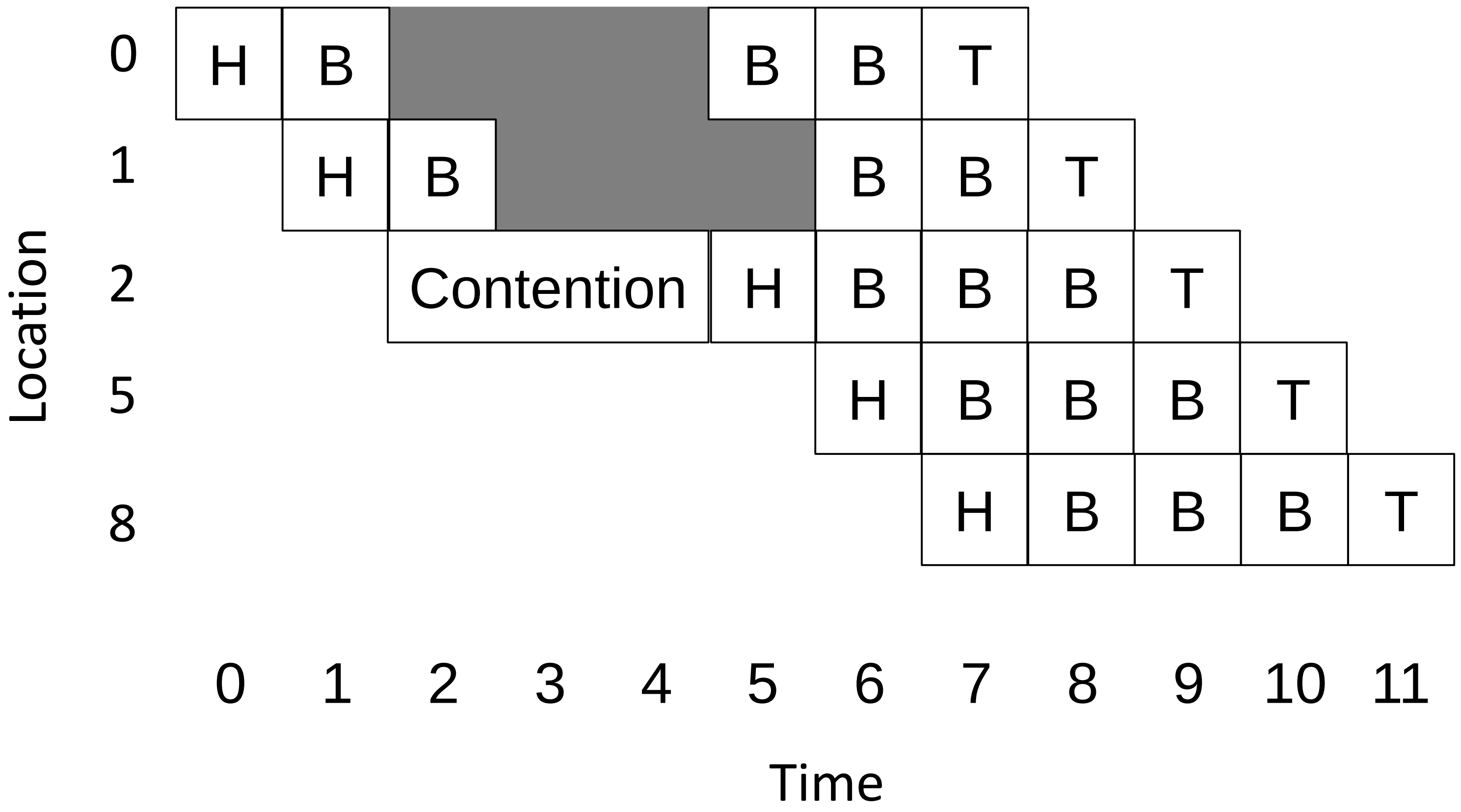
Channel idle but red packet blocked behind blue

Buffer full: blue cannot proceed

Blocked by other packets

- 6 flit buffers/input port

Time-Space Diagram: Wormhole



Location	0	1	2	3	4	5	6	7	8	9	10	11
0	H	B	▨	▨	▨	B	B	T				
1		H	B	▨	▨	▨	B	B	T			
2			Contention			H	B	B	B	T		
5							H	B	B	B	T	
8								H	B	B	B	T

Time

Virtual Channels

- First proposed for deadlock avoidance
  - We’ll not talk about deadlock in the class
  - Read 《On-Chip Networks》for reference

- Can be applied to any flow control
  - First proposed with wormhole

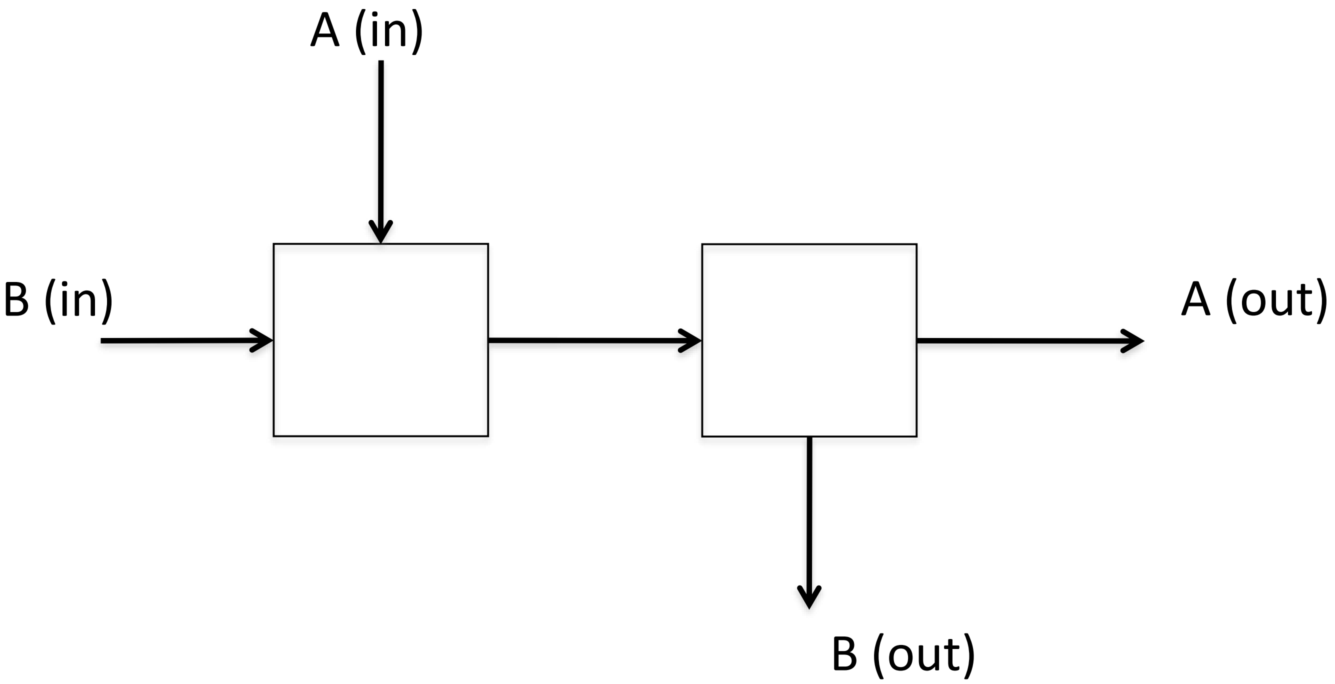
Virtual Channel Flow Control

- Virtual channels used **to combat HOL blocking** in wormhole

- Virtual channels: multiple flit queues per input port
  - Share same physical link (channel)
  - A case of virtualization

- Link utilization improved
  - Flits on different VC can pass blocked packet

Virtual Channel Flow Control (2)



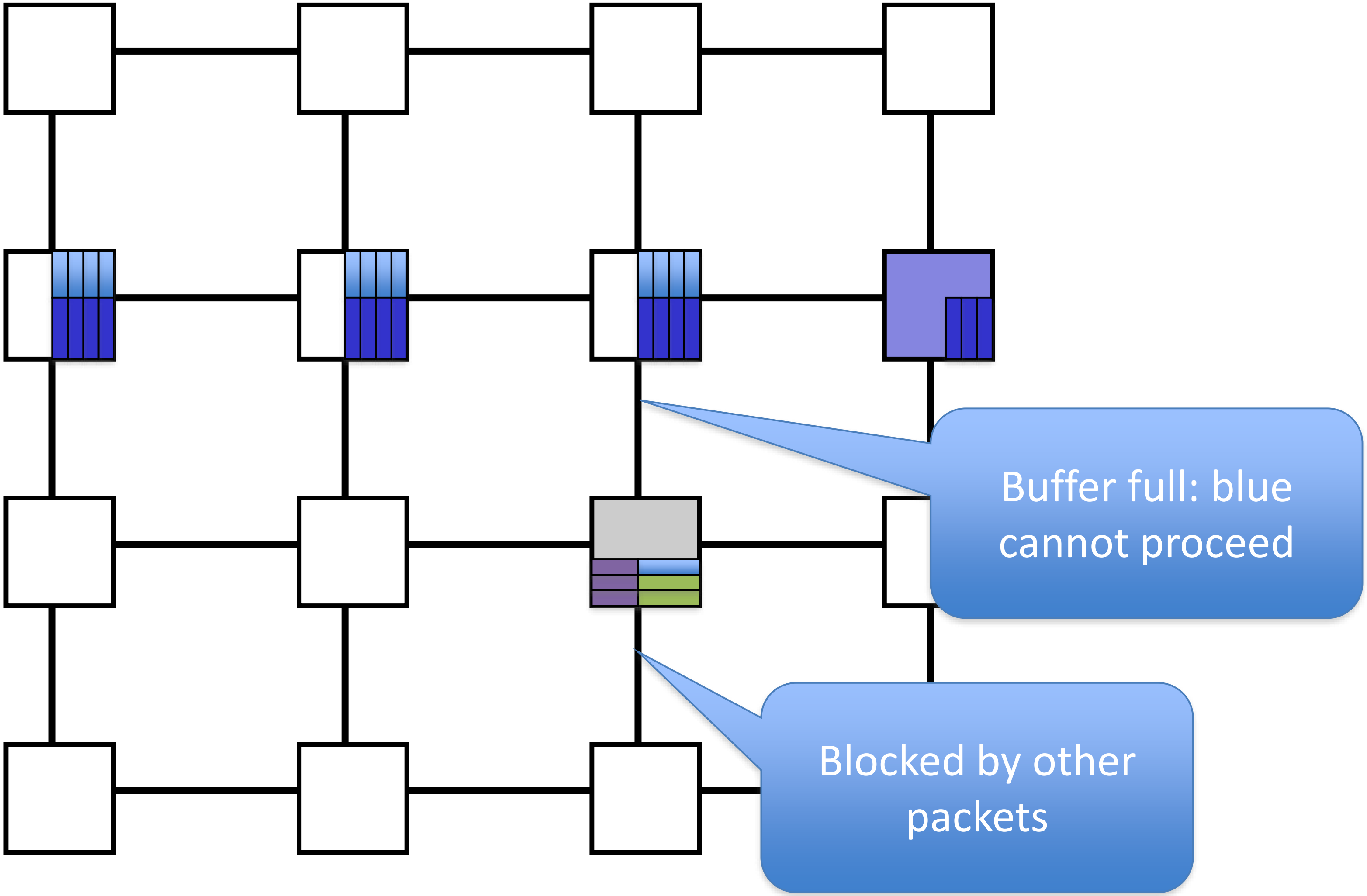
Virtual Channel Flow Control (3)

In1	A H	A1	A2	A3	A4	A5		AT								
	1	1	2	2	3	3	3	3	3	2	2	1	1			
In2	B H	B1	B2	B3	B4		B5		BT							
	1	2	2	3	3	3	3	3	3	3	2	2	1	1		
Out		A H	B H	A1	B1	A2	B2	A3	B3	A4	B4	A5	B5	AT	BT	
A Downstream			A H		A1		A2		A3		A4		A5		AT	
B Downstream				B H		B1		B2		B3		B4		B5		BT

Virtual Channel Flow Control (4)

- Packets compete for VC on flit by flit basis
- In example: on downstream links, flits of each packet are available every other cycle
- Upstream links throttle because of limited buffers
- Does not mean links are idle
  - May be used by packet allocated to other VCs

Virtual Channel Example



- 6 flit buffers/input port
- 3 flit buffers/VC

Summary of techniques

	Links	Buffers	Comments
Circuit-Switching	Messages	N/A (buffer-less)	Setup & Ack
Store and Forward	Packet	Packet	Head flit waits for tail
Virtual Cut Through	Packet	Packet	Head can proceed
Wormhole	Packet	Flit	HOL
Virtual Channel	Flit	Flit	Interleave flits of different packets

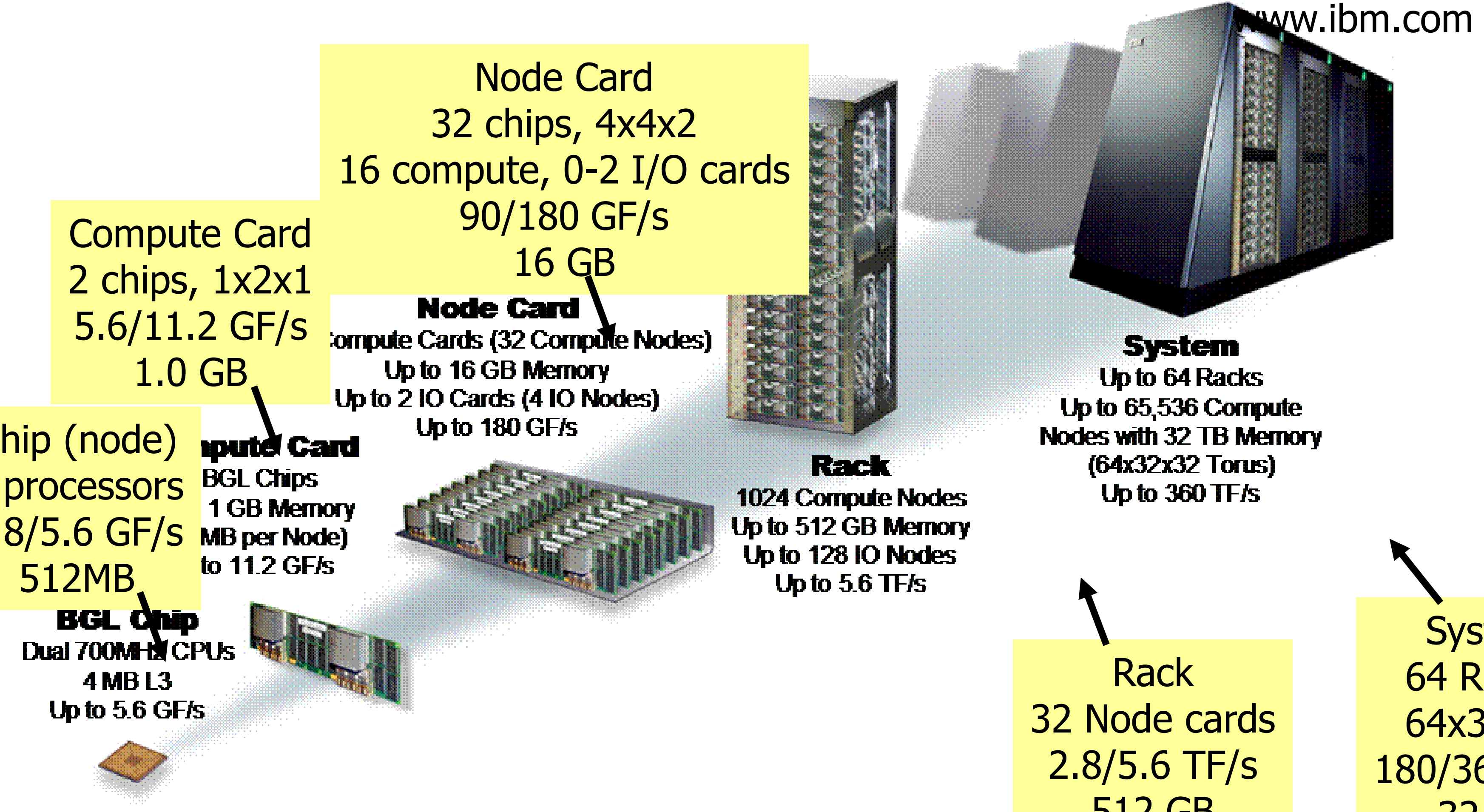
Outline

- Introduction
- Network Topology
- Flow Control
- **Example IN**

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

Blue Gene/L **3D Torus Network**



www.ibm.com

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

Next Topic

Cache Coherence