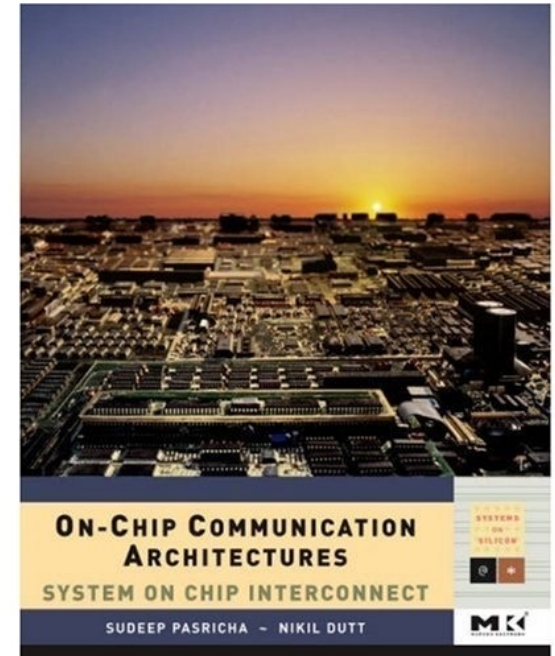


On-Chip Communication Architectures

Networks-on-Chip

Sudeep Pasricha and Nikil Dutt
Slides based on book chapter 12
and from Moraes....



Outline

- Introduction
- NoC Topology
- Switching strategies
- Routing algorithms
- Flow control schemes
- Clocking schemes
- QoS
- NoC Architecture Examples

Introduction

- Scaling
 - Estimating delays becomes **harder**
 - wire geometry determined later in design flow
 - In ultra-deep submicron processes, 80% of the delay of critical path will be due to interconnects
 - Electrical noise due to crosstalk, delay variations and synchronization failure results in **bit upset**
- Conclusion: transmission of digital values on wires will be **slow, power hungry** and **unreliable**

Introduction

- Evolution of on-chip communication architectures

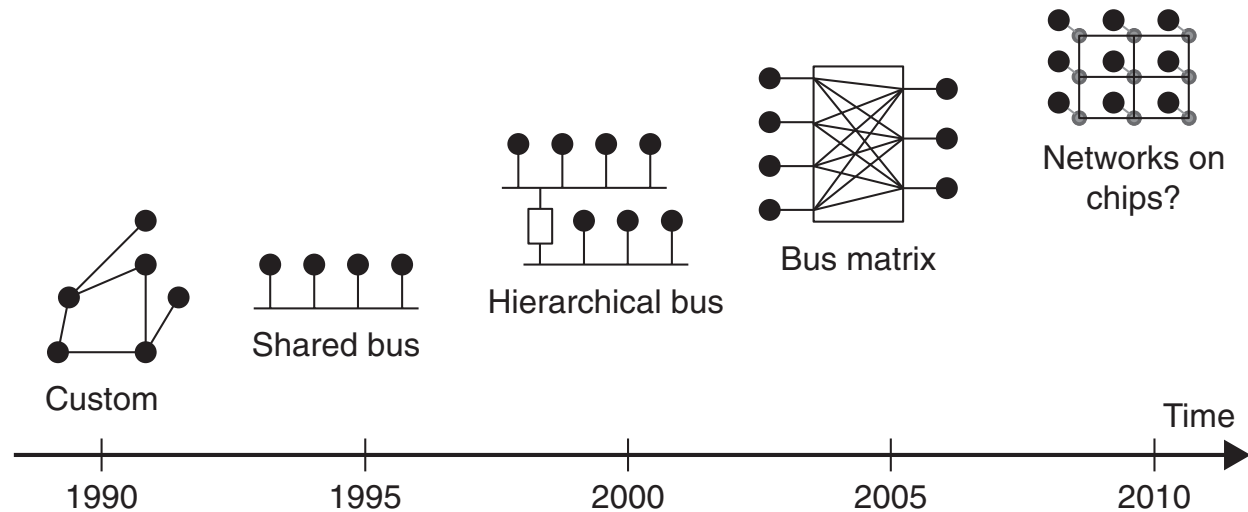


FIGURE 12.3

Evolution of on-chip communication architectures

NoC Definition

- NoC allows decoupling processing cores from communication fabric
 - The need for global synchronization is eliminated
- Benefits
 - Explicit parallelism
 - Modularity
 - Minimize the usage of global wires
 - Power minimization
 - Scalability
 - Better performance

Introduction

- Network-on-chip (NoC) is a packet switched on-chip communication network designed using a layered methodology
 - “routes packets, not wires”
- NoCs use packets to route data from the source to the destination PE via a network fabric that consists of
 - switches (routers)
 - interconnection links (wires)

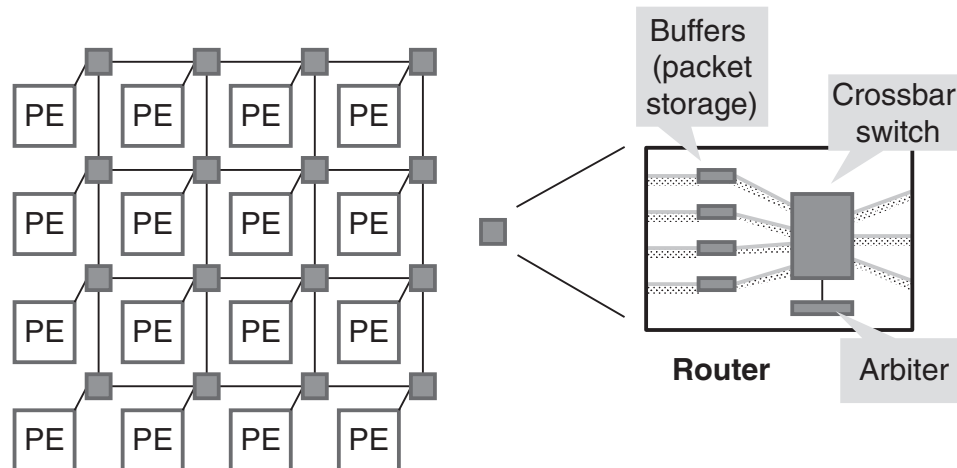


FIGURE 12.1

An example of a (mesh type) NoC interconnection fabric

NOCS: advantages over bus-based designs

	NOC-BASED DESIGN			BUS-BASED DESIGN
Bandwidth and speed	<ul style="list-style-type: none"> Nonblocked switching guarantees multiple concurrent transactions. Pipelined links: higher throughput and clock speed. Regular repetition of similar wire segments, which are easier to model as DSM interconnects. 	☺	☹	<ul style="list-style-type: none"> A transaction blocks other transactions in a shared bus. Every unit attached adds parasitic capacitance; therefore electrical performance degrades with growth.
Resource utilization	<ul style="list-style-type: none"> Packet transactions share the link resources in a statistically multiplexing manner. 	☺	☹	<ul style="list-style-type: none"> A single master occupies a shared bus during its transaction.
Reliability	<ul style="list-style-type: none"> Link-level and packet-basis error control enables earlier detection and gives less penalty. Shorter switch-to-switch link, more error-reliable signaling. Reroute is possible when a fault path exists (self-repairing). 	☺	☹	<ul style="list-style-type: none"> End-to-end error control imposes more penalty. Longer bus-wires are prone to error. A fault path in a bus is a system failure.
Arbitration	<ul style="list-style-type: none"> Distributed arbiters are smaller, thus faster. 	☺	☹	<ul style="list-style-type: none"> All masters request a single arbiter; thus the arbiter becomes big and slow, which obstructs bus speed.
	<ul style="list-style-type: none"> ☹ Distributed arbiters use only local information, not a global traffic condition. 	☹	☺	<ul style="list-style-type: none"> A central arbitration may make a better decision.
Transaction energy	<ul style="list-style-type: none"> Point-to-point connection consumes the minimum transaction energy. 			<ul style="list-style-type: none"> A broadcast transaction needs more energy
Modularity and complexity	<ul style="list-style-type: none"> A switch/link design is instantiated, and thus less design time. Decoupling b/w communicational and computational designs 	☺	☹	<ul style="list-style-type: none"> A bus design is specific, thus not reusable.
Scalability	<ul style="list-style-type: none"> Aggregated bandwidth scales with network size. 	☺	☹	<ul style="list-style-type: none"> A shared bus becomes slower as the design gets bigger and thus is less scalable.
Clocking	<ul style="list-style-type: none"> Plesiochronous, mesochronous, and GALS fashion do not need a globally synchronized clock; much advantageous for high- speed clocking. 	☺	☹	<ul style="list-style-type: none"> A global clock needs to be synchronized over the whole chip bus area.

... But there is no free lunch

	NOC-BASED DESIGN			BUS-BASED DESIGN
Latency	<ul style="list-style-type: none"> ☹ Internal network contention causes a packet latency. ☹ Repeated arbitration on each switch may cause cumulative latency. ☹ Packetizing, synchronizing, and interfacing cause additional latency. 	☹	☺	<ul style="list-style-type: none"> • Bus latency means a wire speed once a master has a grant from an arbiter
Overheads	<ul style="list-style-type: none"> ☹ Additional routers/switches and buffers consume area and power. 	☹	☺	<ul style="list-style-type: none"> • Less area is consumed. • Less buffers are used.
Standardization	<ul style="list-style-type: none"> ☹ There is no NoC-oriented global standard protocol yet; however we can use legacy interfaces such as OCP, AXI, etc. 	☹	☺	<ul style="list-style-type: none"> • AMBA and OCP protocols are widely used and designed for many functional IPs.

Introduction

- NoCs are an attempt to scale down the concepts of largescale networks, and apply them to the embedded system-on-chip (SoC) domain
- NoC Properties
 - Regular geometry that is scalable
 - Flexible QoS guarantees
 - Higher bandwidth
 - Reusable components
 - Buffers, arbiters, routers, protocol stack
 - No long global wires (or global clock tree)
 - **No problematic global synchronization**
 - **GALS: Globally asynchronous, locally synchronous design**
 - Reliable and predictable electrical and physical properties

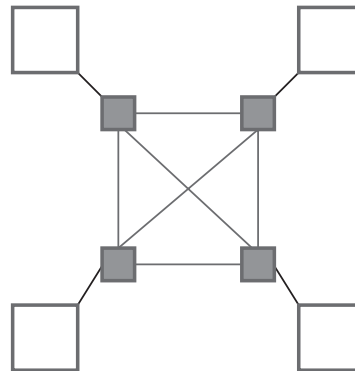
Outline

- Introduction
- **NoC Topology**
- Switching strategies
- Routing algorithms
- Flow control schemes
- Clocking schemes
- QoS
- NoC Architecture Examples

NoC Topology

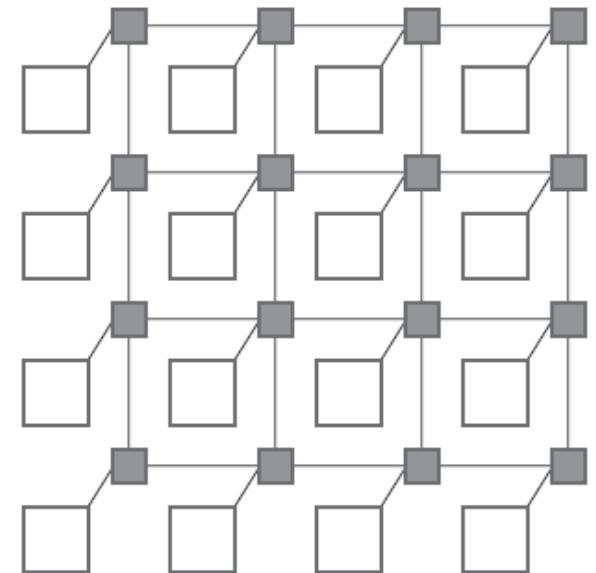
- Direct Topologies

- each node has direct point-to-point link to a subset of other nodes in the system called neighboring nodes
- nodes consist of computational blocks and/or memories, as well as a NI block that acts as a router
- e.g. Nostrum, SOCBUS, Proteo, Octagon
- as the number of nodes in the system increases, the total available communication bandwidth also increases
- fundamental trade-off is between connectivity and cost



NoC Topology

- Most direct network topologies have an orthogonal implementation, where nodes can be arranged in an n-dimensional orthogonal space
 - routing for such networks is fairly simple
 - e.g. n-dimensional mesh, torus, folded torus, hypercube, and octagon
- 2D mesh is most popular topology
 - all links have the same length
 - eases physical design
 - area grows linearly with the number of nodes
 - **must be designed in such a way as to avoid traffic accumulating in the center of the mesh**

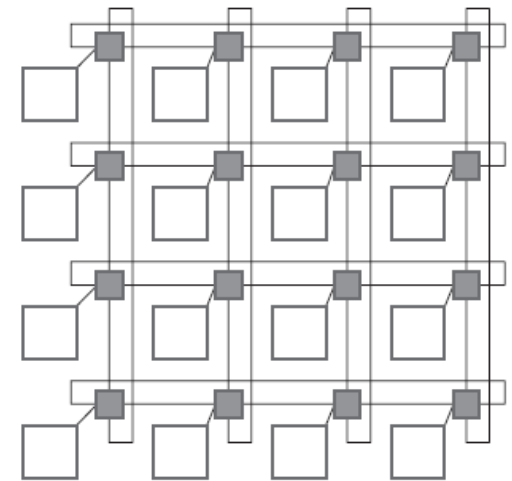


NoC Topology

- Torus topology, also called a k-ary n-cube, is an n-dimensional grid with k nodes in each dimension
 - k-ary 1-cube (1-D torus) is essentially a ring network with k nodes
 - limited scalability as performance decreases when more nodes

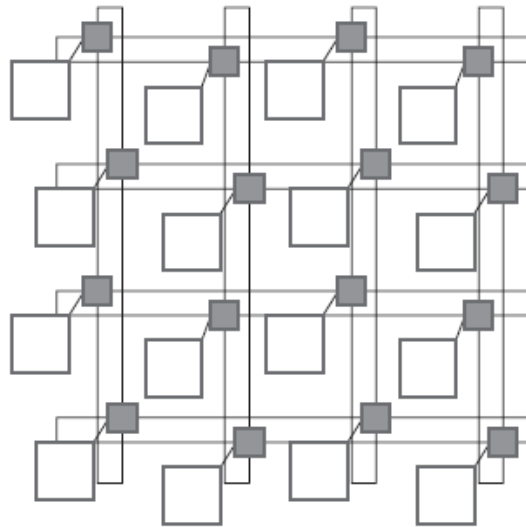


- k-ary 2-cube (i.e., 2-D torus) topology is similar to a regular mesh
 - except that nodes at the edges are connected to switches at the opposite edge via wrap-around channels
 - long end-around connections can, however, lead to excessive delays



NoC Topology

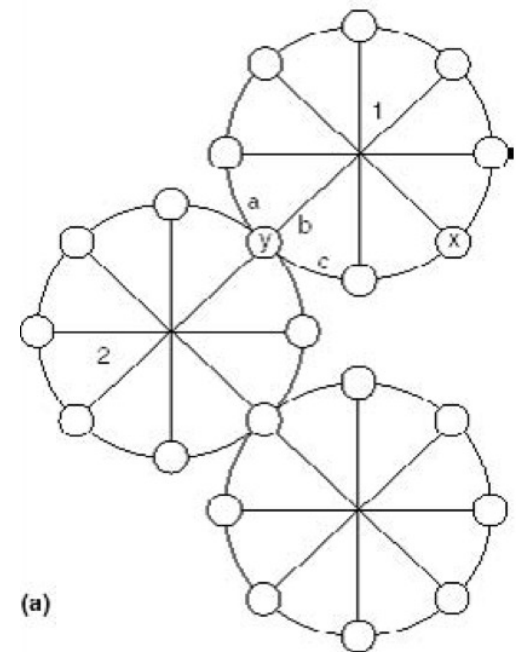
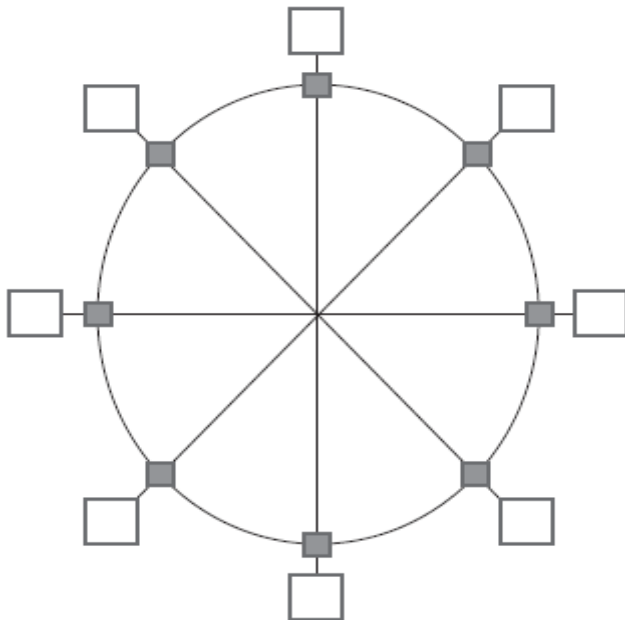
- Folding torus topology overcomes the long link limitation of a 2-D torus
 - links have the same size



- Meshes and tori can be extended by adding bypass links to increase performance at the cost of higher area

NoC Topology

- Octagon topology is another example of a direct network
 - messages being sent between any 2 nodes require at most two hops
 - more octagons can be tiled together to accommodate larger designs
 - by using one of the nodes is used as a bridge node



NoC Topology

- Spidergon

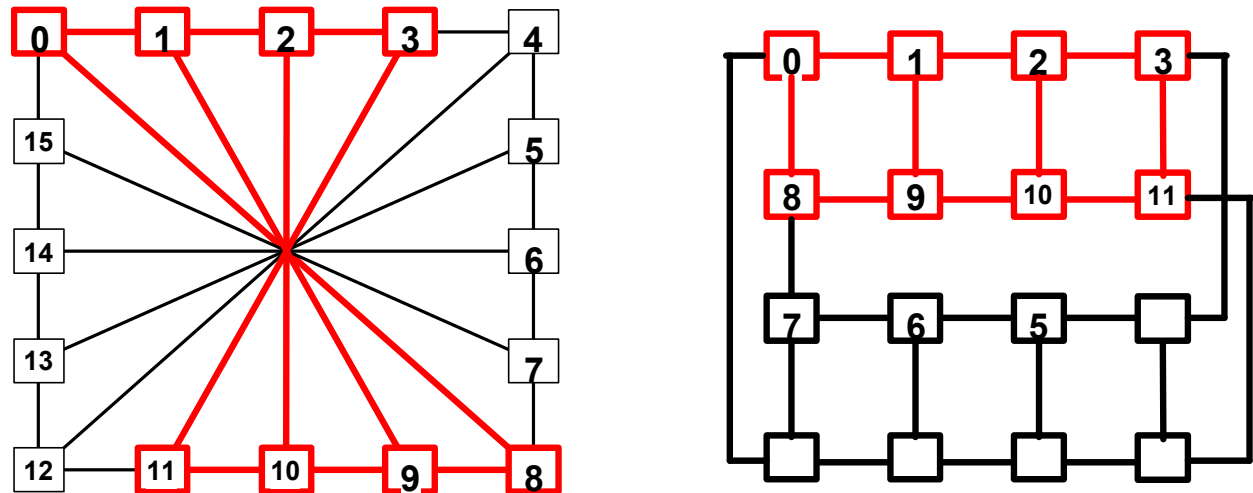


FIGURE 3.27: Equivalent representation of Spidergon STNoC for $N = 16$

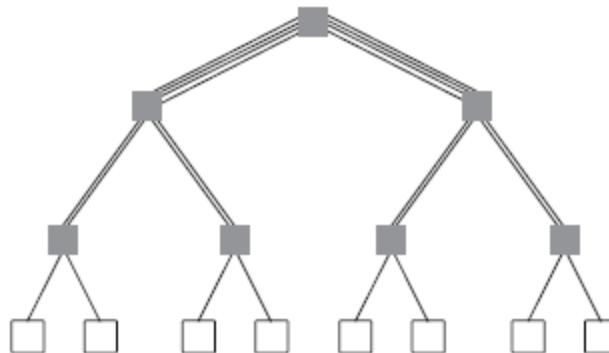
NoC Topology

- **Indirect Topologies**

- each node is connected to an external switch, and switches have point-to-point links to other switches
- switches do not perform any information processing, and correspondingly nodes do not perform any packet switching
- e.g. SPIN, crossbar topologies

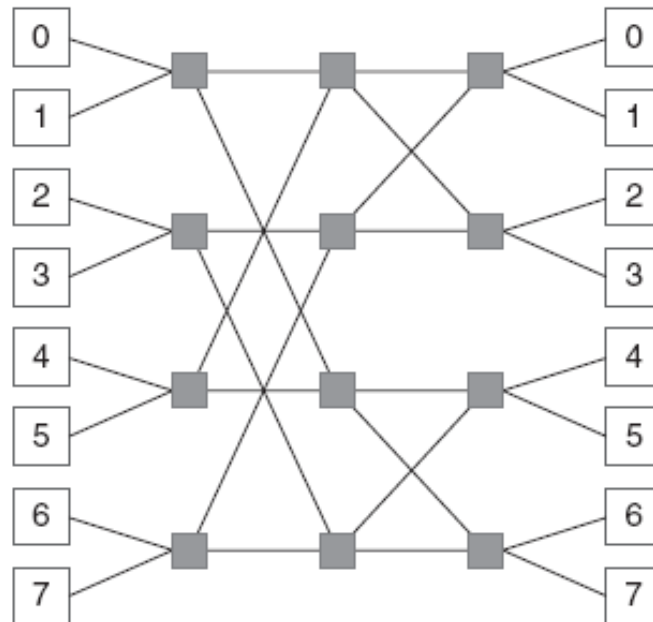
- **Fat tree topology**

- nodes are connected only to the leaves of the tree
- more links near root, where bandwidth requirements are higher



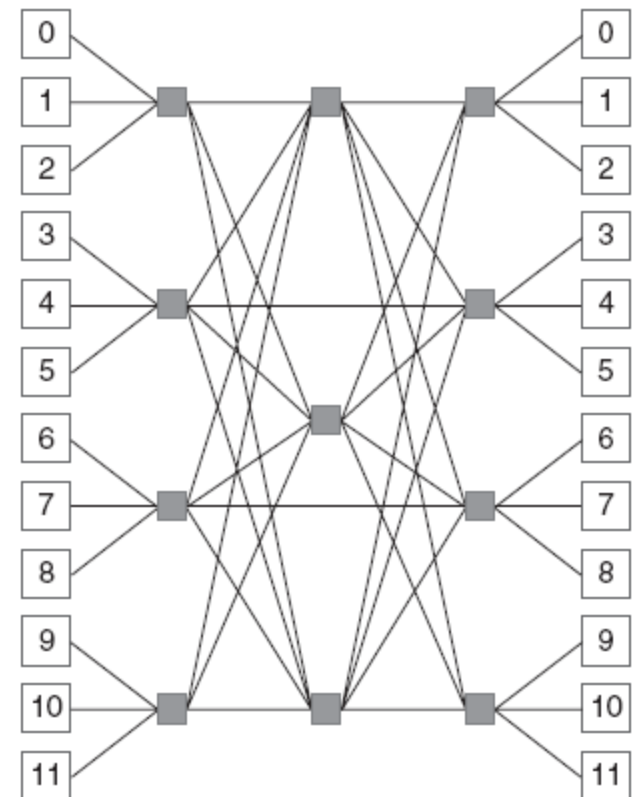
NoC Topology

- k-ary n-fly butterfly network
 - blocking multi-stage network – packets may be temporarily blocked or dropped in the network if contention occurs
 - k^n nodes, and n stages of k^{n-1} $k \times k$ crossbar
 - e.g. 2-ary 3-fly butterfly network



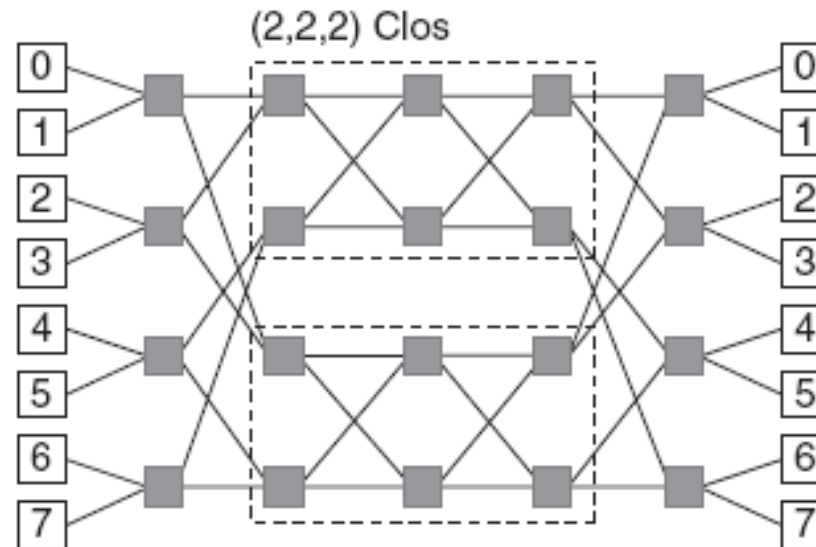
NoC Topology

- (m, n, r) symmetric Clos network
 - three-stage network in which each stage is made up of a number of crossbar switches
 - m is the no. of middle-stage switches
 - n is the number of input/output nodes on each input/output switch
 - r is the number of input and output switches
 - e.g. $(3, 3, 4)$ Clos network
 - non-blocking network
 - expensive (several full crossbars)



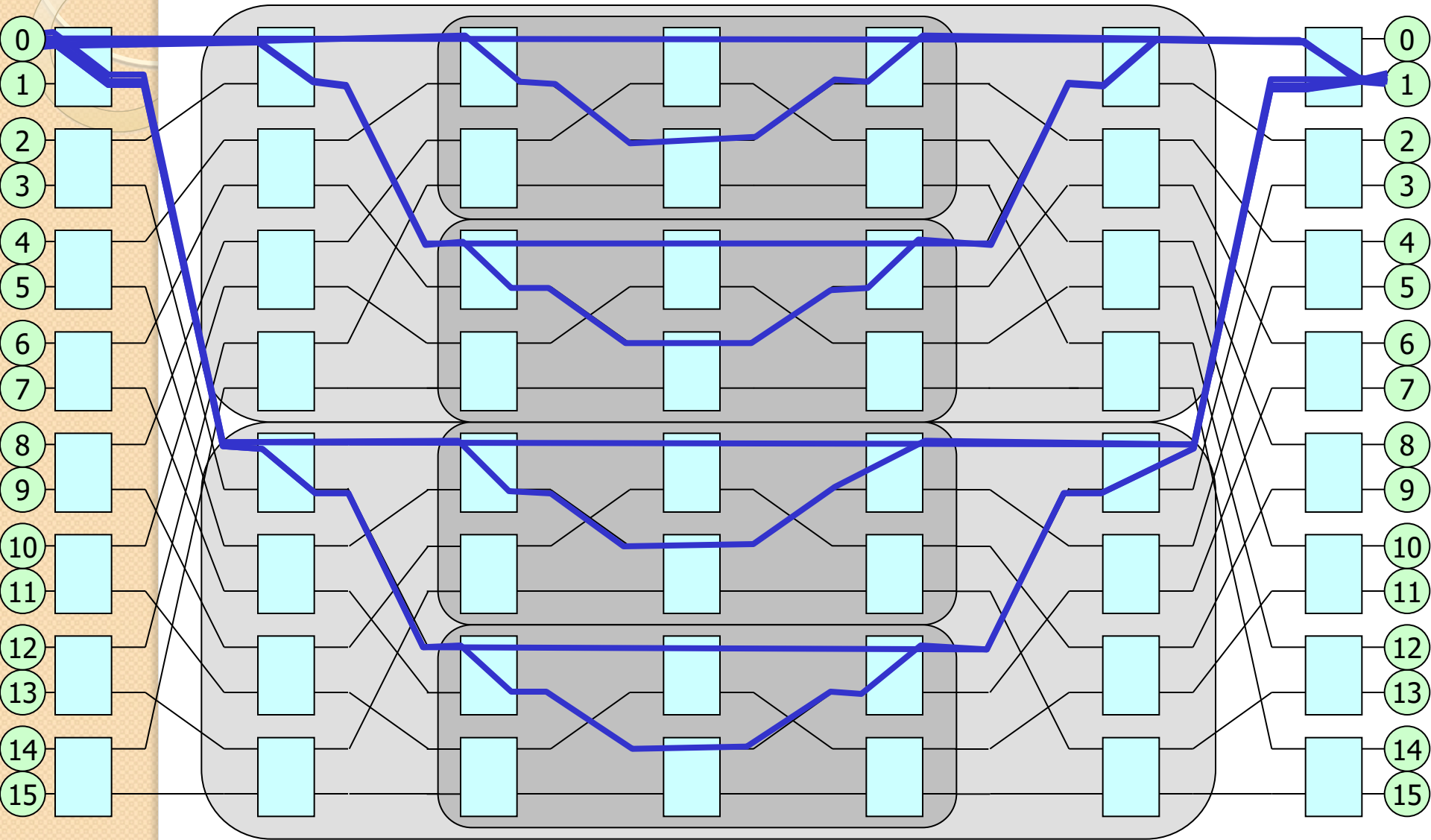
NoC Topology

- Benes network
 - rearrangeable network in which paths may have to be rearranged to provide a connection, requiring an appropriate controller
 - Clos topology composed of 2×2 switches
 - e.g. (2, 2, 4) re-arrangeable Clos network constructed using two (2, 2, 2) Clos networks with 4×4 middle switches



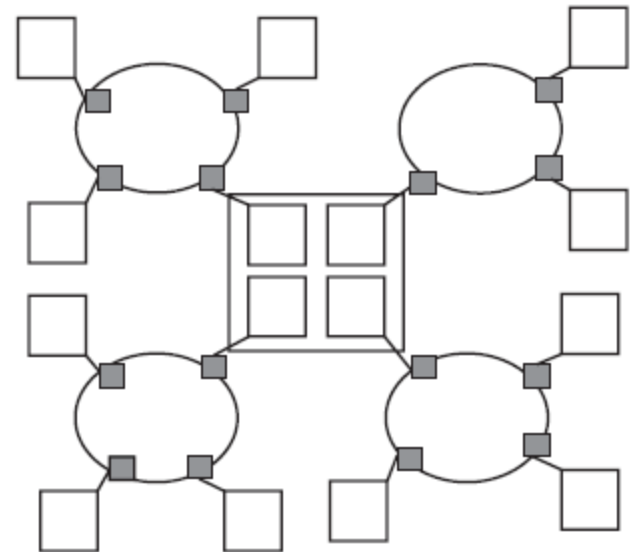
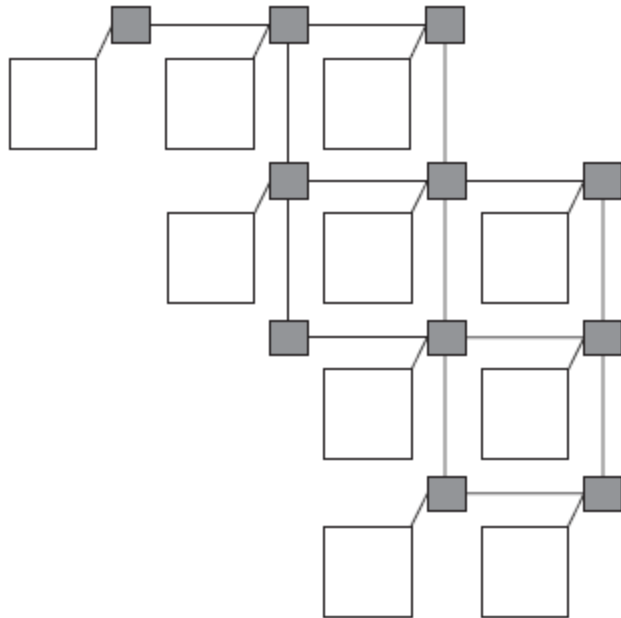
NoC Topology

Alternative paths from 0 to 1. 16 port



NoC Topology

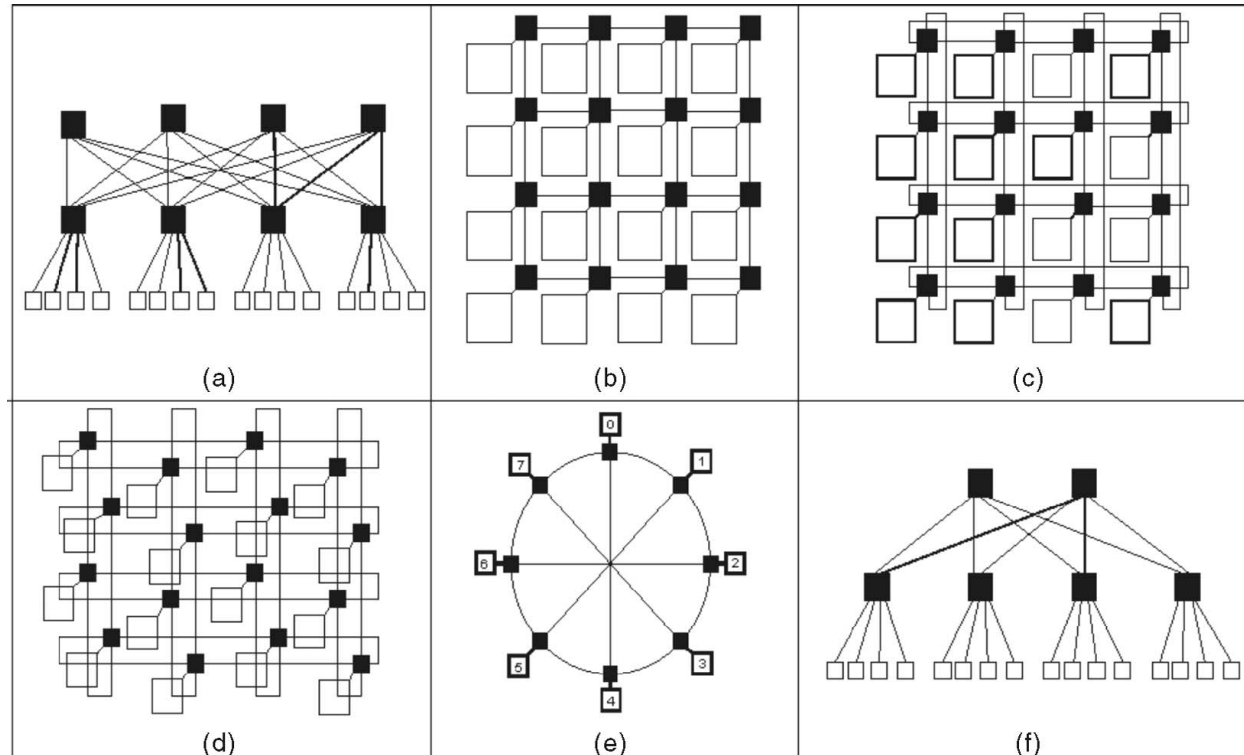
- Irregular or ad hoc network topologies
 - **customized for an application**
 - usually a mix of shared bus, direct, and indirect network topologies
 - e.g. reduced mesh, cluster-based hybrid topology



NoC Topology

- ❑ Heritage of networks with new constraints
 - Need to accommodate interconnects in a 2D layout
 - Cannot route long wires (clock frequency bound)

- a) SPIN,
- b) CLICHE'
- c) Torus
- d) Folded torus
- e) Octagon
- f) BFT.



Outline

- Introduction
- NoC Topology
- Switching strategies
- Routing algorithms
- Flow control schemes
- Clocking schemes
- QoS
- NoC Architecture Examples

Switching strategies

- Determine how data flows through routers in the network
- Define granularity of data transfer and applied switching technique
 - phit is a unit of data that is transferred on a link in a single cycle
 - typically, phit size = flit size

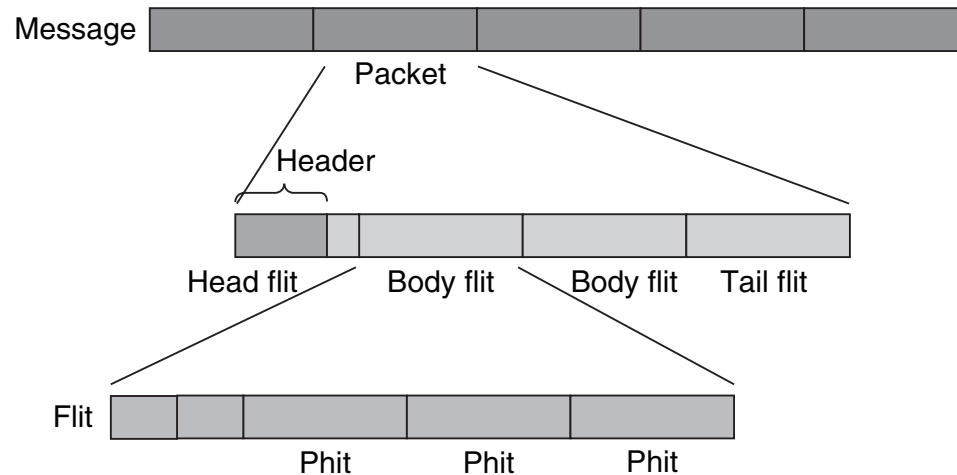


FIGURE 12.7

Structure of messages, packets, flits, and phits

Switching strategies

- Two main modes of transporting flits in a NoC are circuit switching and packet switching
- Circuit switching
 - physical path between the source and the destination is reserved prior to the transmission of data
 - message header flit traverses the network from the source to the destination, reserving links along the way
 - Advantage: low latency transfers, once path is reserved
 - Disadvantage: pure circuit switching does not scale well with NoC size
 - several links are occupied for the duration of the transmitted data, even when no data is being transmitted
 - for instance in the setup and tear down phases

Switching strategies

- Virtual circuit switching
 - creates virtual circuits that are multiplexed on links
 - number of virtual links (or virtual channels (VCs)) that can be supported by a physical link depends on buffers allocated to link
 - Possible to allocate either one buffer per virtual link or one buffer per physical link
 - Allocating one buffer per virtual link
 - depends on how virtual circuits are spatially distributed in the NoC, routers can have a different number of buffers
 - can be expensive due to the large number of shared buffers
 - multiplexing virtual circuits on a single link also requires scheduling at each router and link (end-to-end schedule)
 - conflicts between different schedules can make it difficult to achieve bandwidth and latency guarantees

Switching strategies

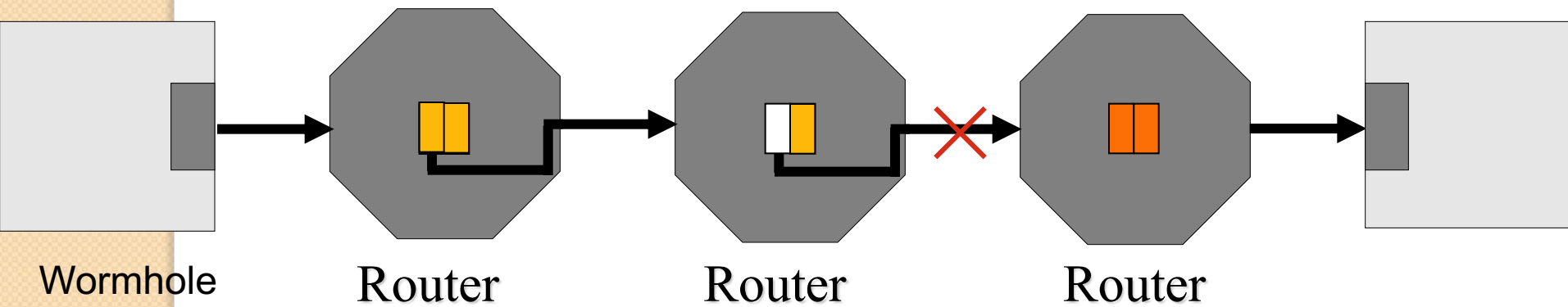
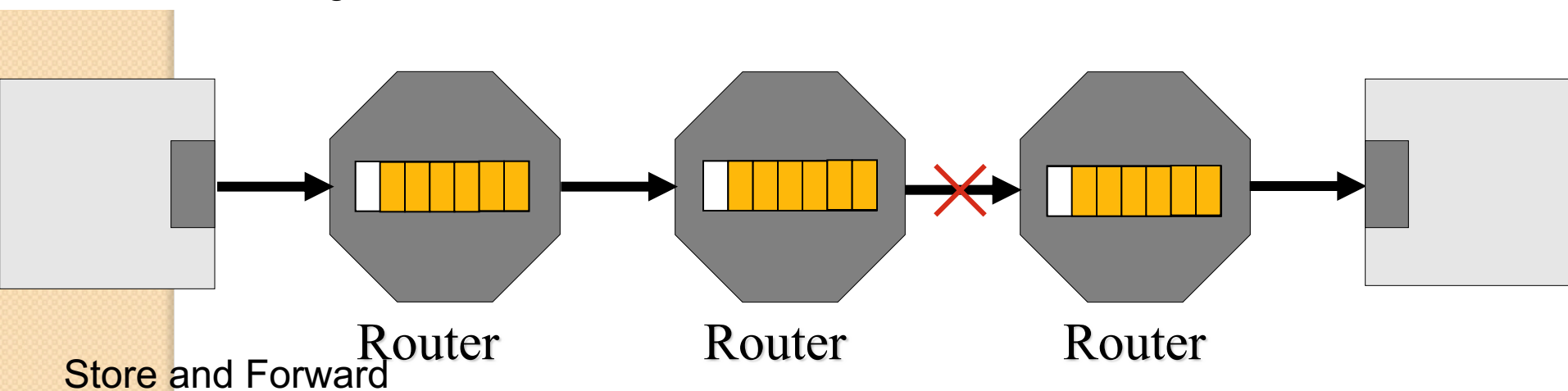
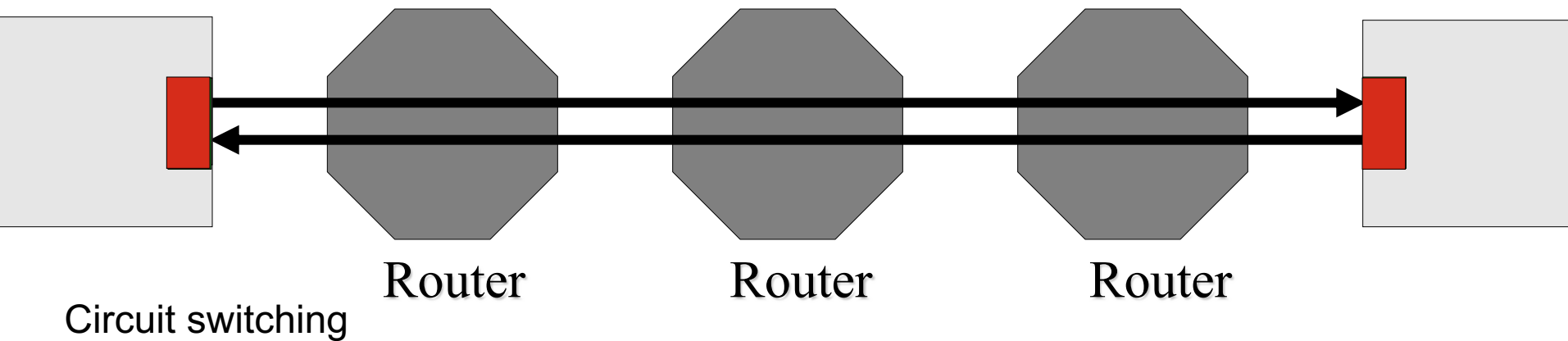
- Allocating one buffer per physical link
 - virtual circuits are time multiplexed with a single buffer per link
 - uses time division multiplexing (TDM) to statically schedule the usage of links among virtual circuits
 - flits are typically buffered at the NIs and sent into the NoC according to the TDM schedule
 - global scheduling with TDM makes it easier to achieve end-to-end bandwidth and latency guarantees
 - less expensive router implementation, with fewer buffers

Switching strategies

- Packet Switching
 - packets are transmitted from source and make their way independently to receiver
 - possibly along different routes and with different delays
 - zero start up time, followed by a variable delay due to contention in routers along packet path
 - QoS guarantees are harder to make in packet switching than in circuit switching
 - three main packet switching scheme variants
- SAF switching
 - packet is sent from one router to the next only if the receiving router has buffer space for entire packet
 - buffer size in the router is at least equal to the size of a packet
 - Disadvantage: excessive buffer requirements

Switching strategies

- VCT Switching
 - reduces router latency over SAF switching by forwarding first flit of a packet as soon as space for the entire packet is available in the next router
 - if no space is available in receiving buffer, no flits are sent, and the entire packet is buffered
 - same buffering requirements as SAF switching
- WH switching
 - flit from a packet is forwarded to receiving router if space exists for that flit
 - parts of the packet can be distributed among two or more routers
 - buffer requirements are reduced to one flit, instead of an entire packet
 - more susceptible to deadlocks due to usage dependencies between links



Outline

- Introduction
- NoC Topology
- Switching strategies
- **Routing algorithms**
- Flow control schemes
- Clocking schemes
- QoS
- NoC Architecture Examples

Routing algorithms

- Responsible for correctly and efficiently routing packets or circuits from the source to the destination
- Choice of a routing algorithm depends on trade-offs between several potentially conflicting metrics
 - minimizing power required for routing
 - minimizing logic and routing tables to achieve a lower area footprint
 - increasing performance by reducing delay and maximizing traffic utilization of the network
 - improving robustness to better adapt to changing traffic needs
- Routing schemes can be classified into several categories
 - static or dynamic routing
 - distributed or source routing
 - minimal or non-minimal routing

Routing algorithms

- Static and dynamic routing
 - static routing: fixed paths are used to transfer data between a particular source and destination
 - does not take into account current state of the network
 - advantages of static routing:
 - easy to implement, since very little additional router logic is required
 - in-order packet delivery if single path is used
 - dynamic routing: routing decisions are made according to the current state of the network
 - considering factors such as availability and load on links
 - path between source and destination may change over time
 - as traffic conditions and requirements of the application change
 - more resources needed to monitor state of the network and dynamically change routing paths
 - able to better distribute traffic in a network

Routing algorithms

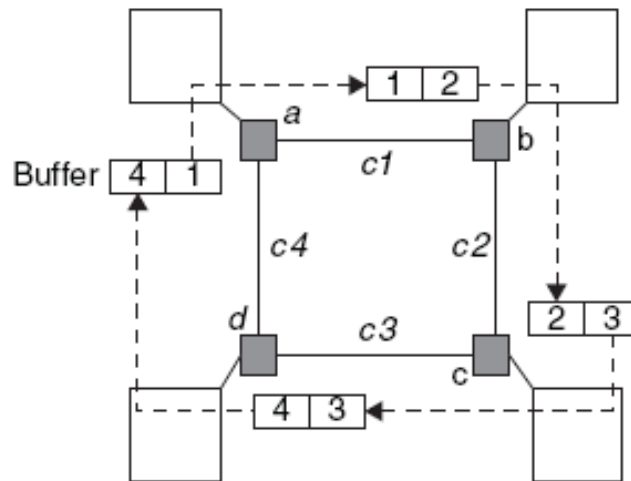
- Distributed and source routing
 - static and dynamic routing schemes can be further classified depending on where the routing information is stored, and where routing decisions are made
 - distributed routing: each packet carries the destination address
 - e.g., XY co-ordinates or number identifying destination node/router
 - routing decisions are made in each router by looking up the destination addresses in a routing table or by executing a hardware function
 - source routing: packet carries routing information
 - pre-computed routing tables are stored at a nodes' NI
 - routing information is looked up at the source NI and routing information is added to the header of the packet (increasing packet size)
 - when a packet arrives at a router, the routing information is extracted from the routing field in the packet header
 - does not require a destination address in a packet, any intermediate routing tables, or functions needed to calculate the route

Routing algorithms

- Minimal and non-minimal routing
 - minimal routing: length of the routing path from the source to the destination is the shortest possible length between the two nodes
 - e.g. in a mesh NoC topology (where each node can be identified by its XY co-ordinates in the grid) if source node is at (0, 0) and destination node is at (i, j), then the minimal path length is $|i| + |j|$
 - source does not start sending a packet if minimal path is not available
 - non-minimal routing: can use longer paths if a minimal path is not available
 - by allowing non-minimal paths, the number of alternative paths is increased, which can be useful for avoiding congestion
 - disadvantage: overhead of additional power consumption

Routing algorithms

- Routing algorithm must ensure freedom from deadlocks
 - common in VH switching
 - e.g. cyclic dependency shown below

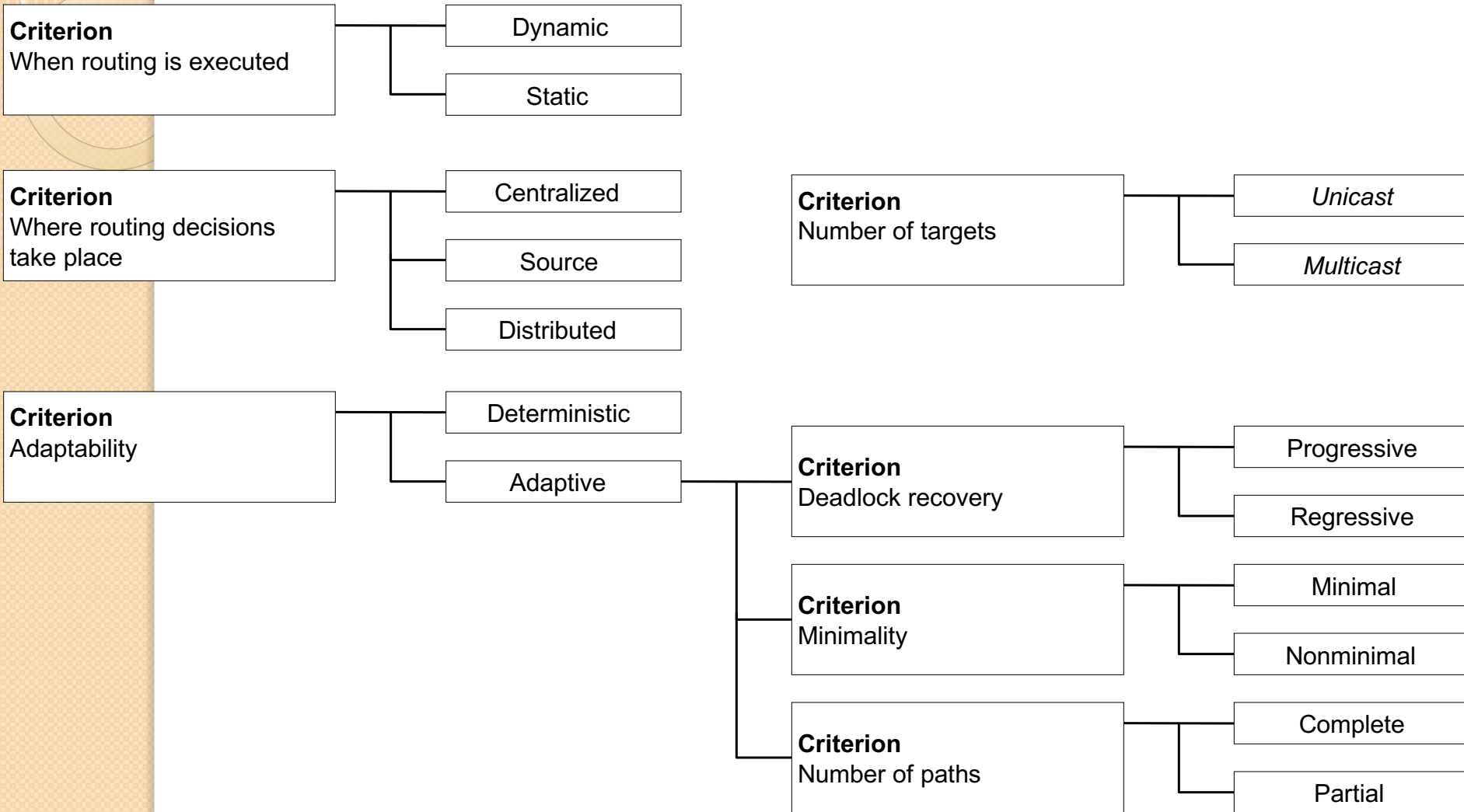


- freedom from deadlocks can be ensured by allocating additional hardware resources or imposing restrictions on the routing
- usually dependency graph of the shared network resources is built and analyzed either statically or dynamically

Routing algorithms

- Routing algorithm must ensure freedom from livelocks
 - livelocks are similar to deadlocks, except that states of the resources involved constantly change with regard to one another, without making any progress
 - occurs especially when dynamic (adaptive) routing is used
 - e.g. can occur in a deflection “hot potato” routing if a packet is bounced around over and over again between routers and never reaches its destination
 - livelocks can be avoided with simple priority rules
- Routing algorithm must ensure freedom from starvation
 - under scenarios where certain packets are prioritized during routing, some of the low priority packets never reach their intended destination
 - can be avoided by using a fair routing algorithm, or reserving some bandwidth for low priority data packets

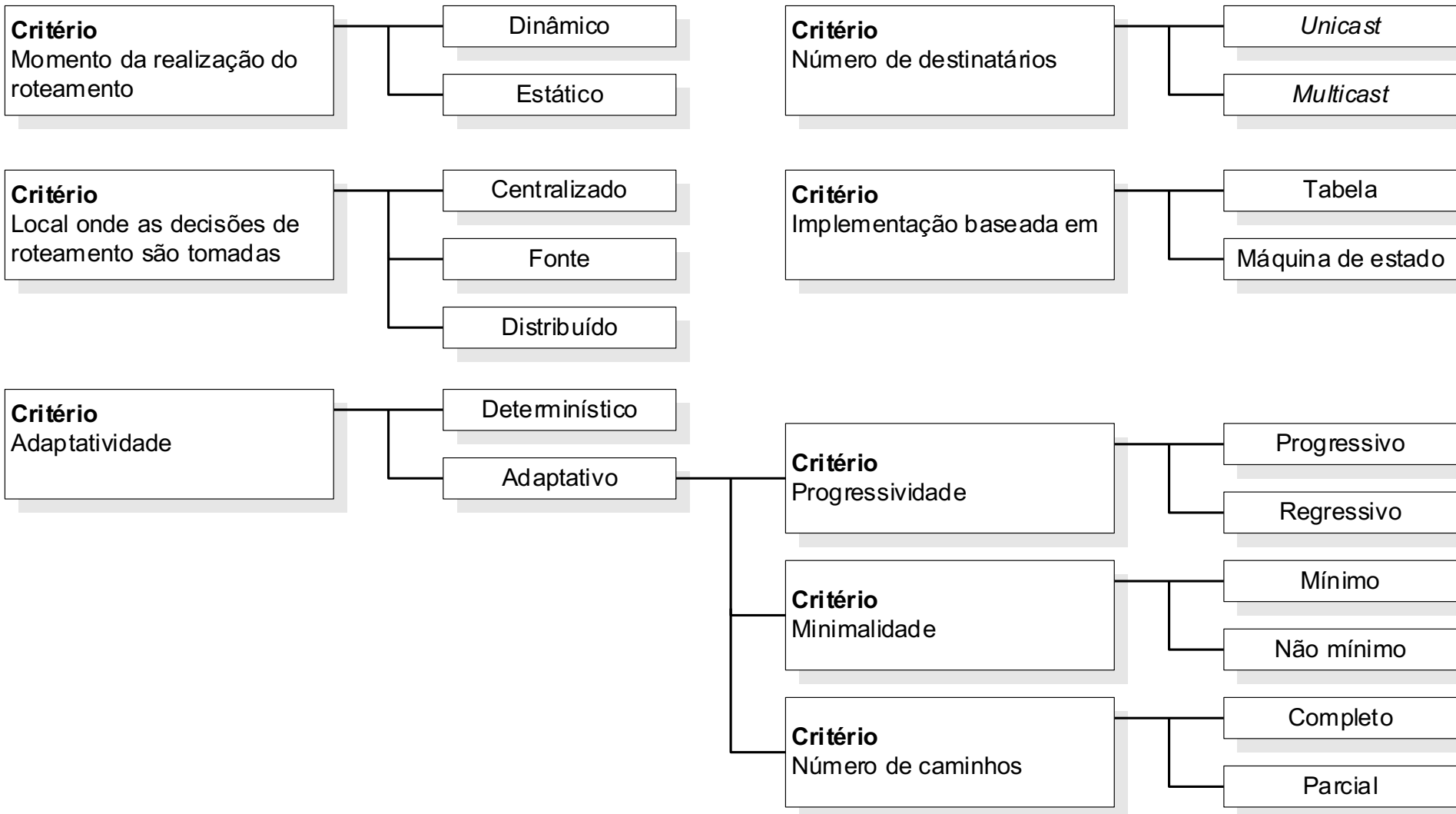
NoC routing techniques: classification



NoC Routing Algorithms

Fernando Moraes

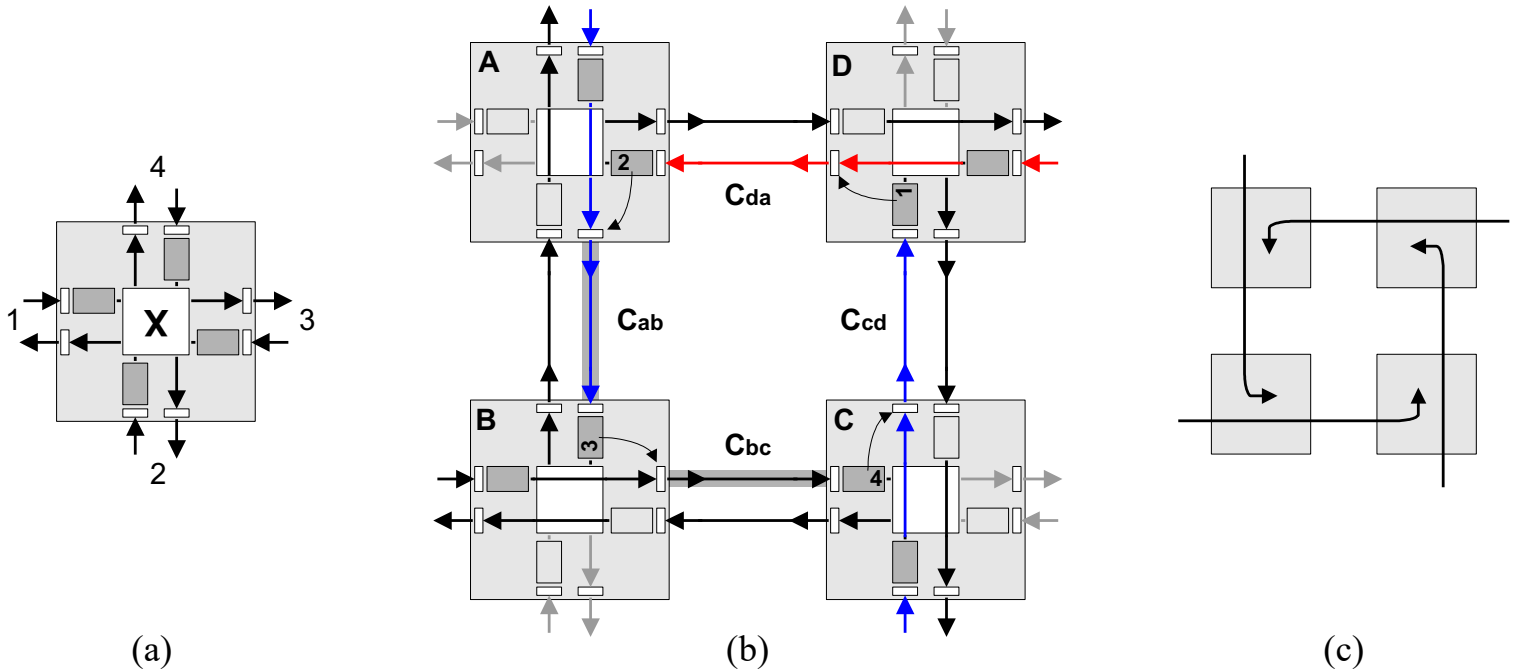
Roteamento: Classificações



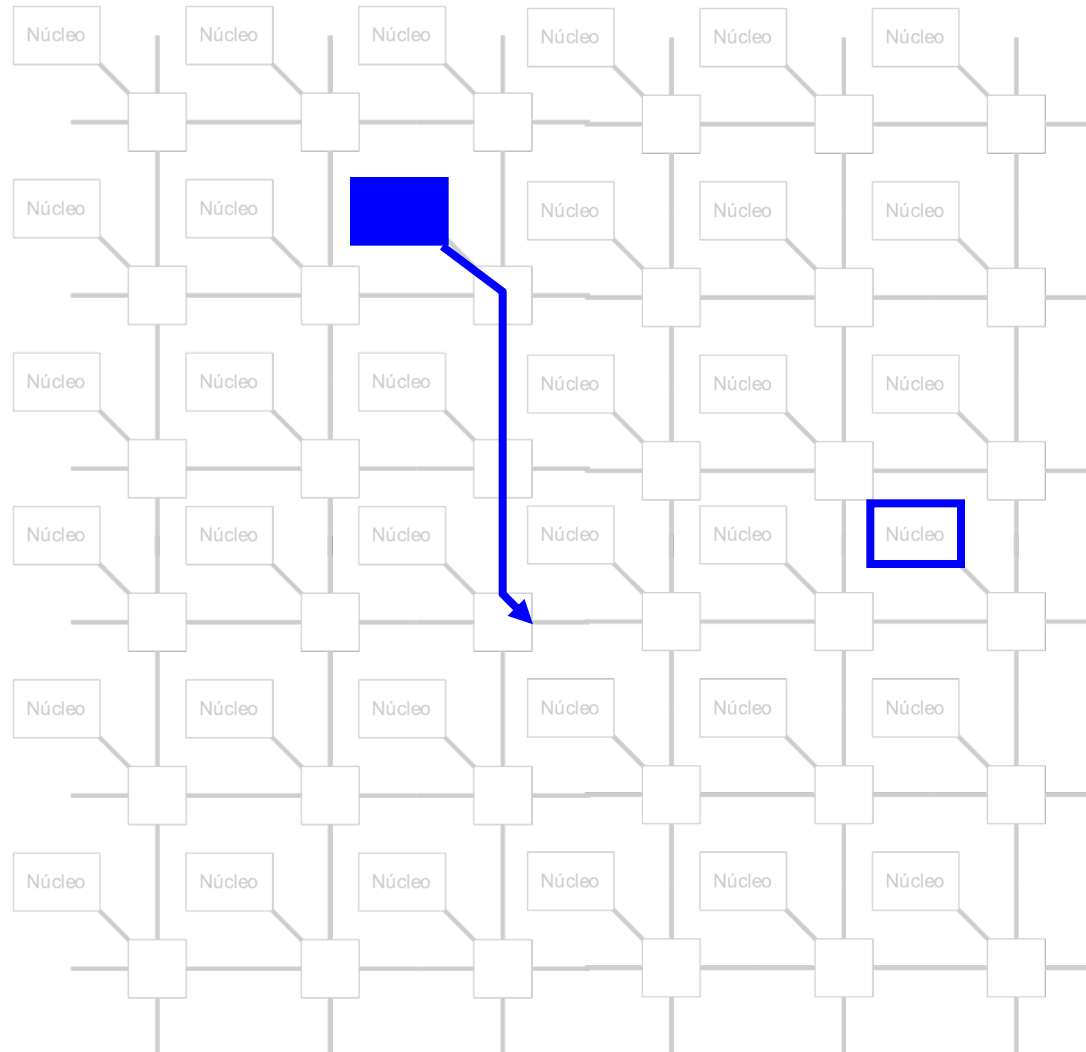
Técnicas de roteamento

Livelock e Deadlock

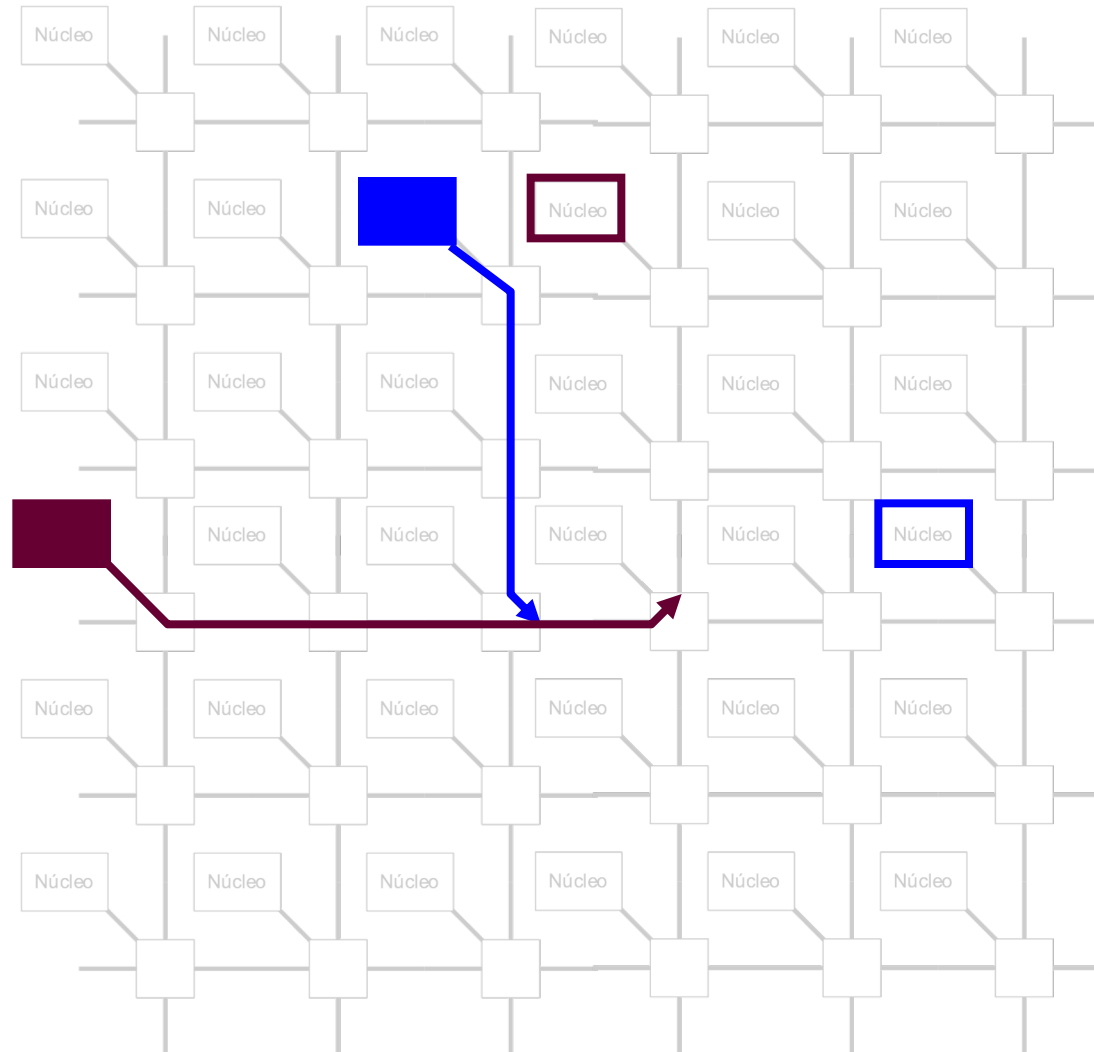
- *Livelock*
 - Quando uma mensagem trafega permanentemente pela rede sem chegar ao seu destino
- *Deadlock*
 - Ocorre quando existe uma dependência cíclica de recursos na rede e as mensagens são paralisadas



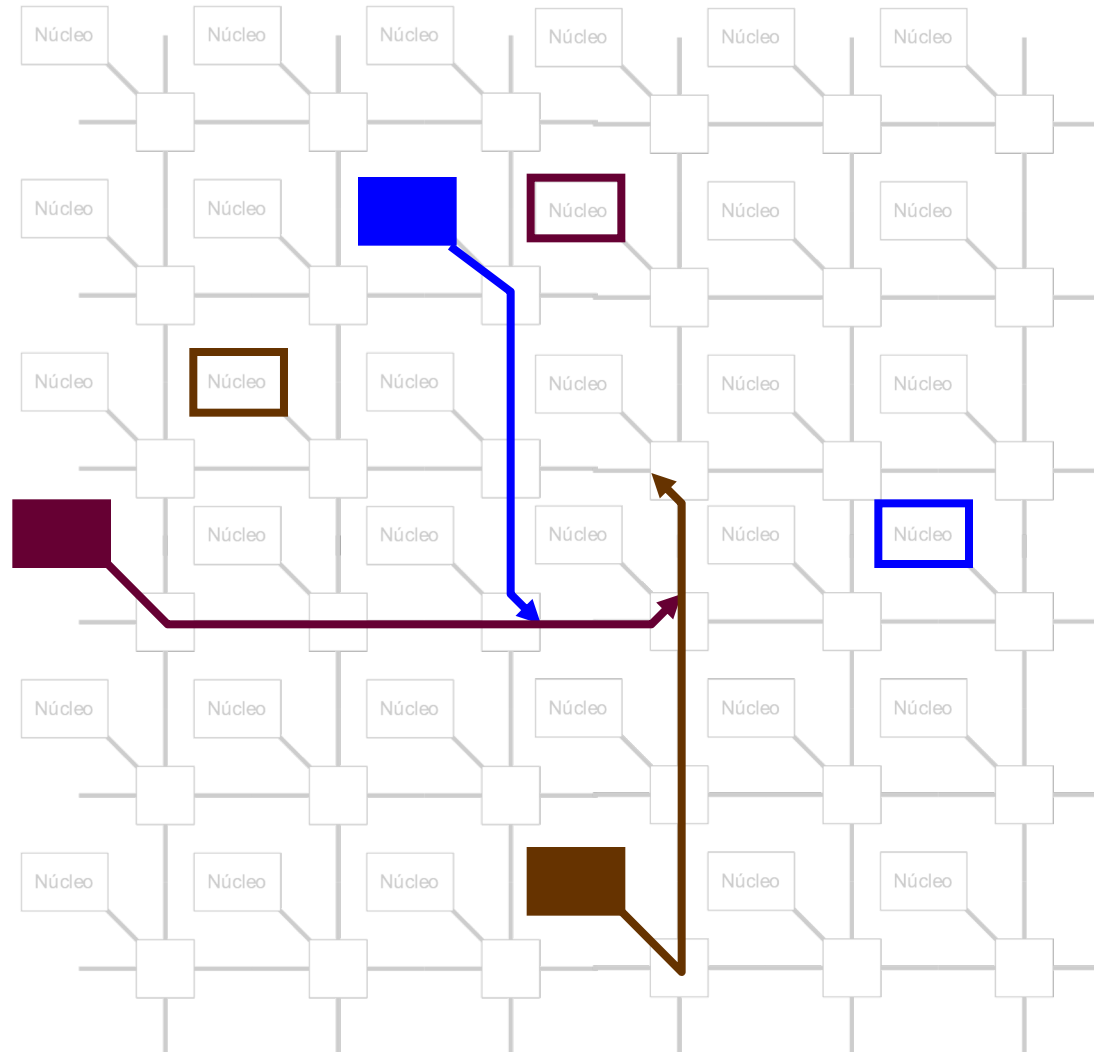
Roteamento: Exemplo de deadlock



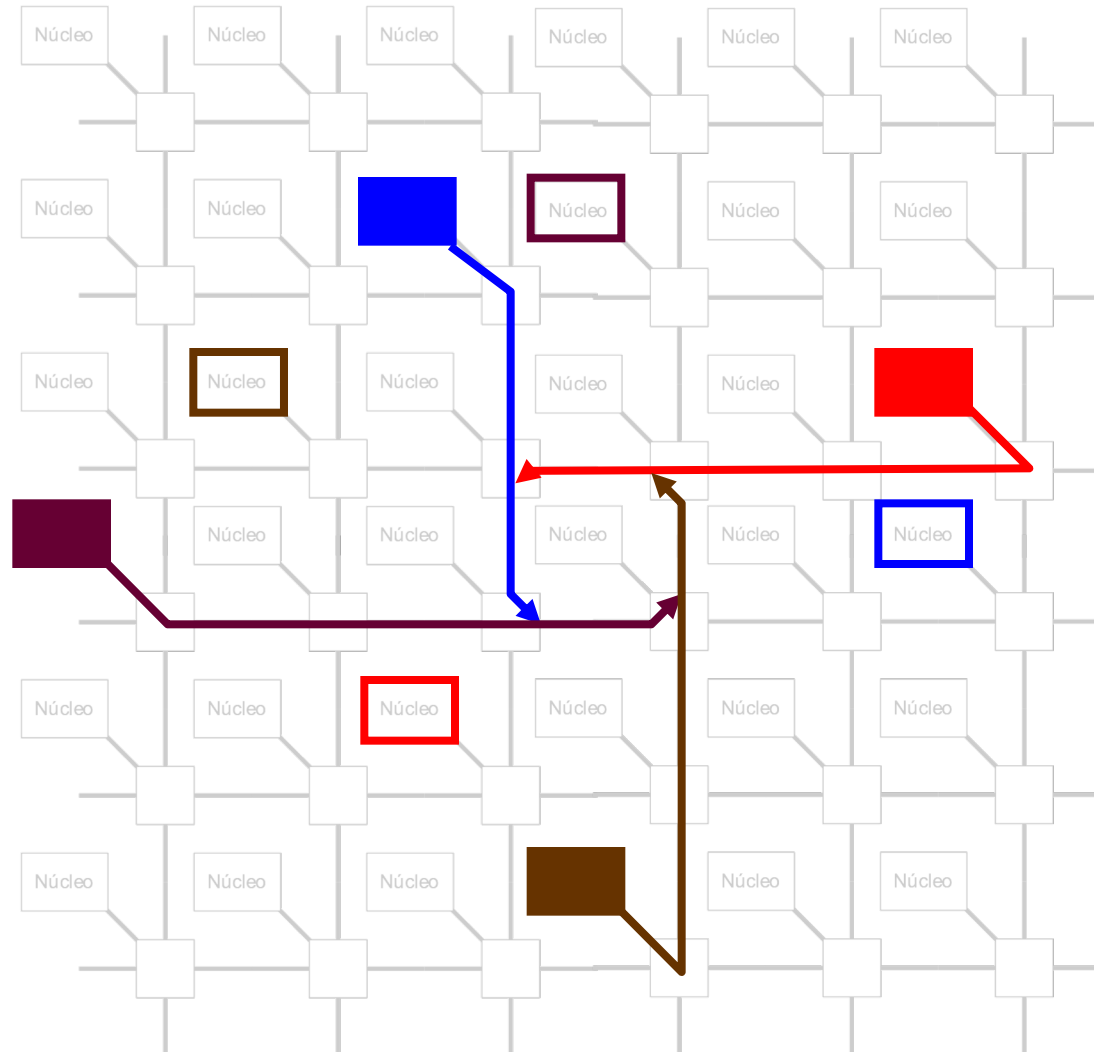
Roteamento: Exemplo de deadlock



Roteamento: Exemplo de deadlock

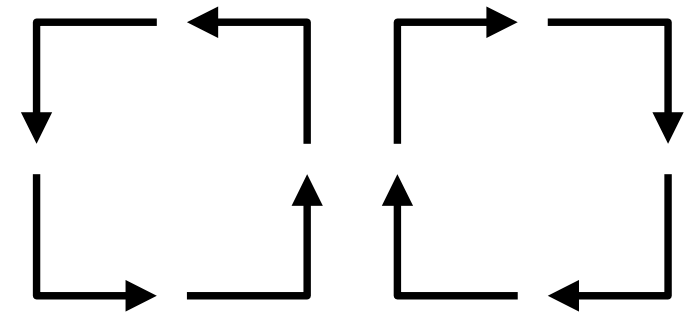
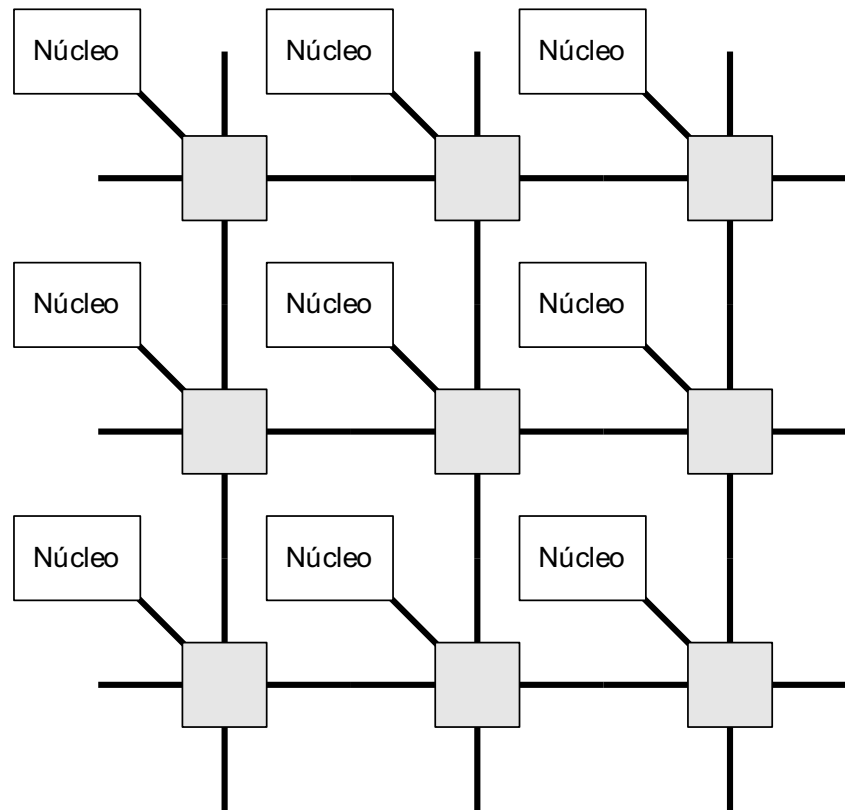


Roteamento: Exemplo de deadlock



Roteamento: Evitando o deadlock

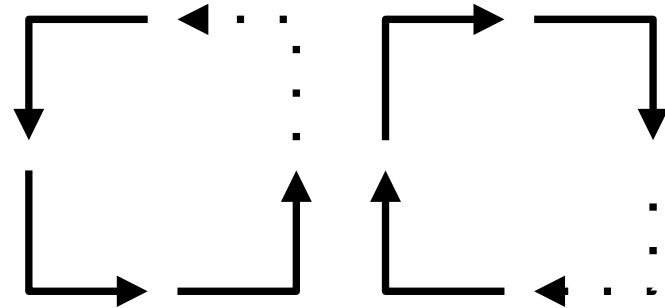
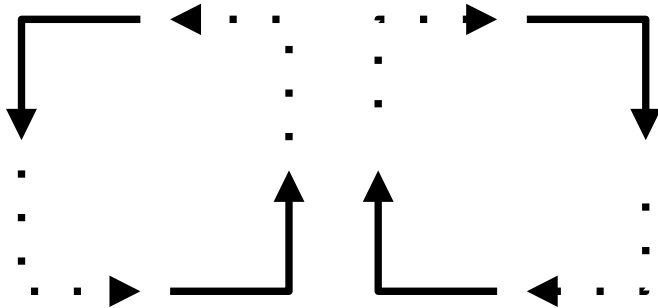
- Deve-se evitar o surgimento de ciclos na rede



Ciclos na malha 2-D

Roteamento: Evitando o deadlock

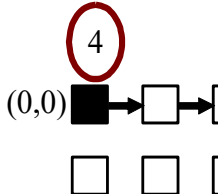
- Soluções
- Roteamento XY
 - Proíbe qualquer curva do tipo Y-para-X
 - É determinístico
- Roteamento West-first
 - Proíbe apenas as voltas em direção ao oeste
 - É parcialmente adaptativo

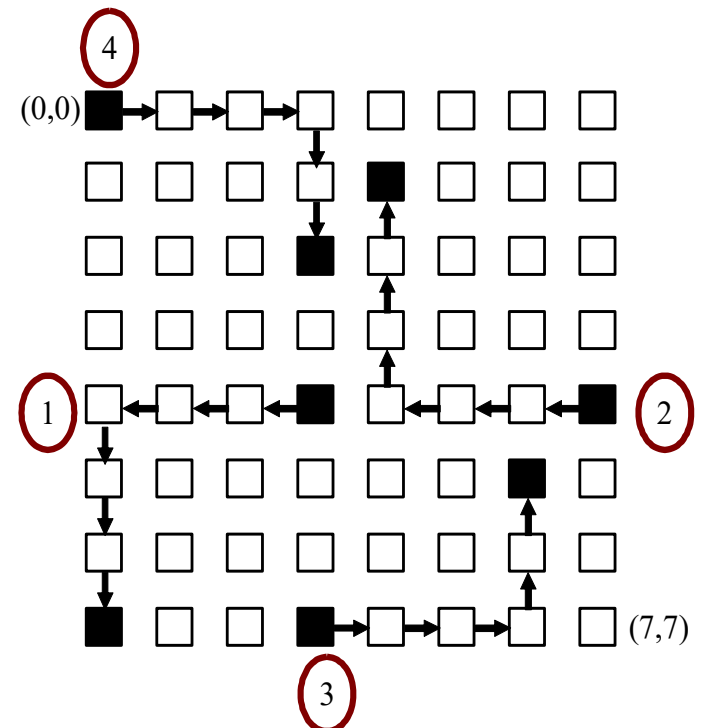
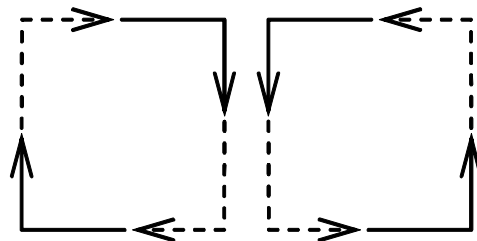


The Turn Model

- Turn-based model to avoid deadlock
- Possible turns = {NW, NE, SW, SE, WN, WS, EN, ES}
- Disallow ≥ 2 turns
- XY routing only allows turns from X to Y {EN, ES, WN, WS}
- West-first routing prohibits turns to west {NW, SW}
 - Offers full adaptiveness to paths that route east
 - Not fair to all paths

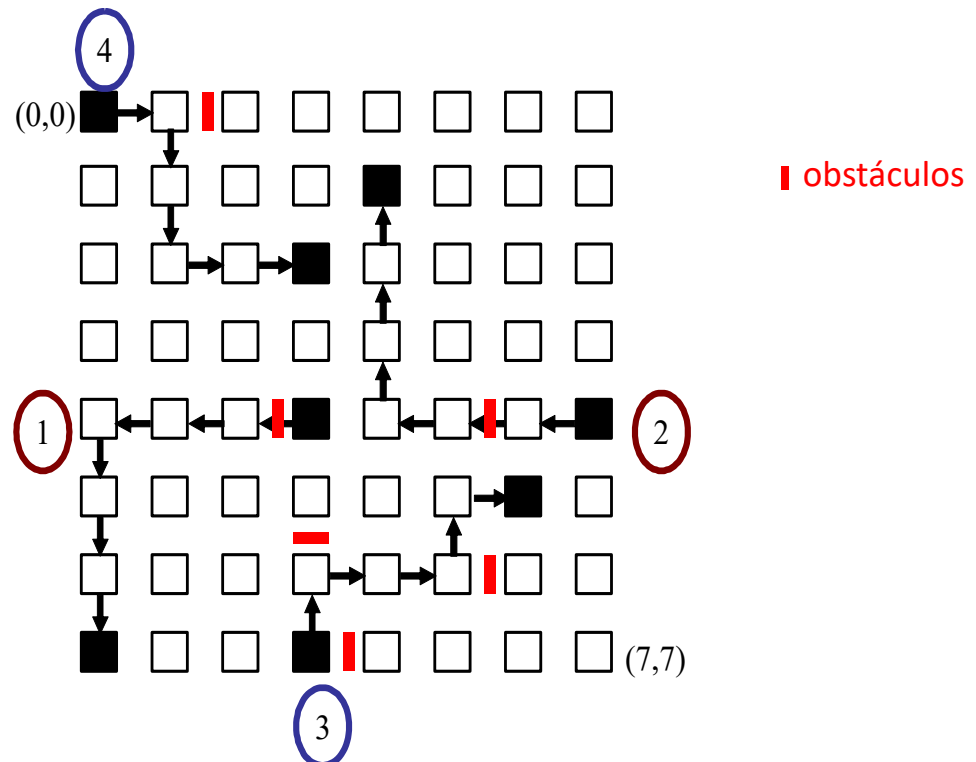
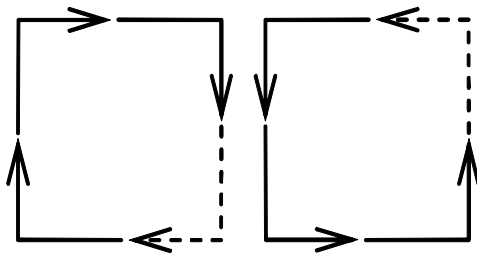
Roteamento: Algoritmo XY

- Deterministic
 - All messages from Src to Dest will traverse the same path
 - Common example: **Dimension Order Routing (DOR)**
 - Message traverses network dimension by dimension
 - Aka XY routing
 - Cons:
 - Eliminates any path diversity provided by topology
 - Poor load balancing
 - Pros:
 - Simple and inexpensive to implement
 - Deadlock free
- 



West-first routing algorithm

- The prohibited turns are the two to the West
 - if $XT \leq XS$, packets are routed deterministically, as in the XY algorithm, (paths 1 and 2)
 - if $XT > XS$ packets can be routed adaptively in East, North or South directions (paths 3 and 4)

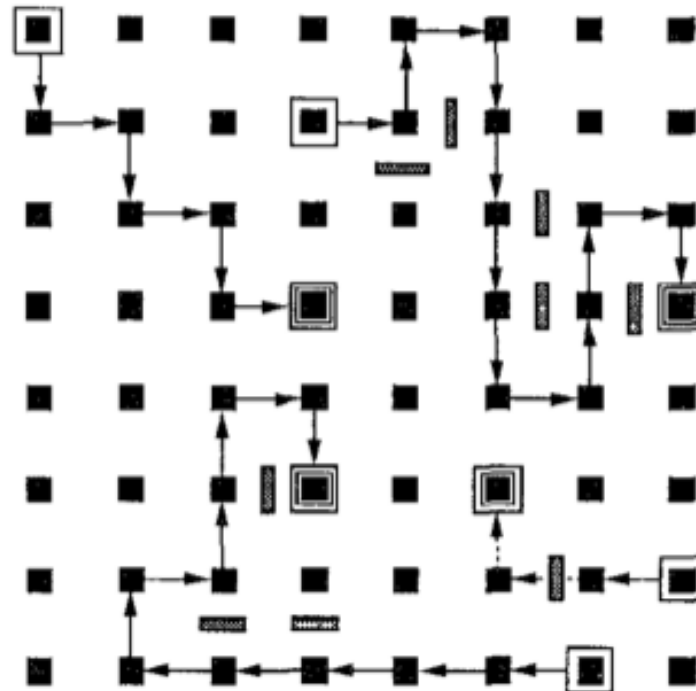
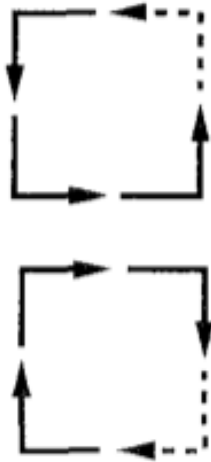


(1) e (2) – caminhos determinísticos

(3) e (4) – caminhos adaptativos

West First não mínimo

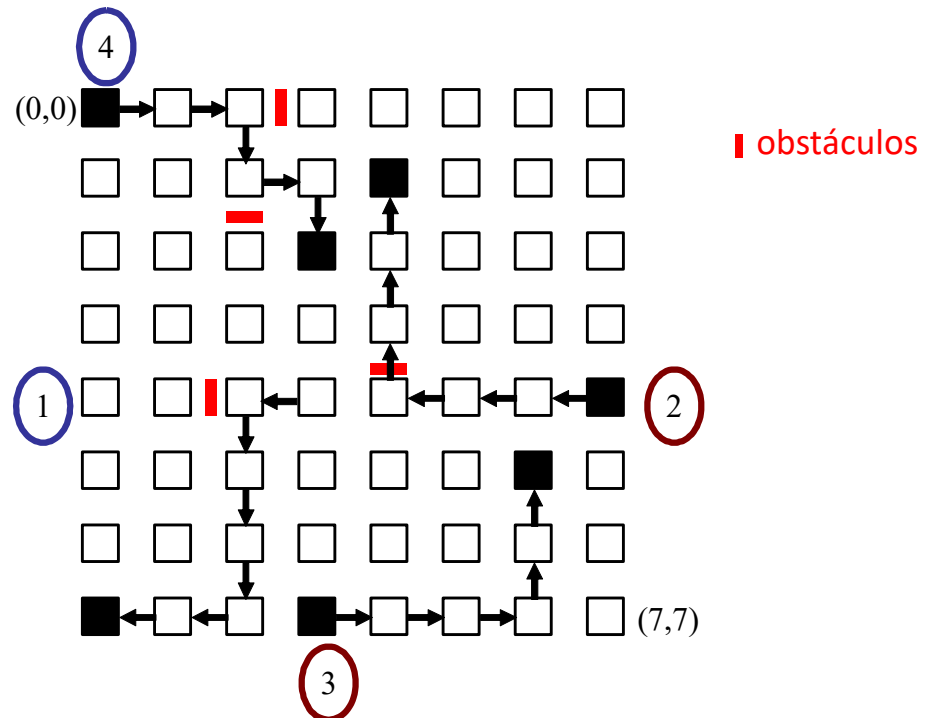
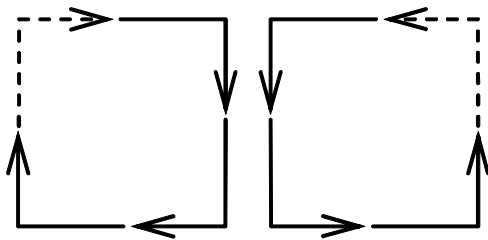
- Route a packet first west if necessary, and then adaptively south, east, and north.
- Both minimal and non-minimal paths are shown.
- [Dally and Seitz] proof show that a routing algorithm is deadlock free if the channels in the interconnection network can be numbered so that the algorithm routes every packet along channels with strictly decreasing numbers.



(a)

North-Last routing algorithm

- The prohibited turns are the two when traveling North
 - - if $YT \leq YS$ packets are routed deterministically (paths 2 and 3)
 - - if $YT > YS$ packets can be routed adaptively in West, East, or South directions (paths 1 and 4)

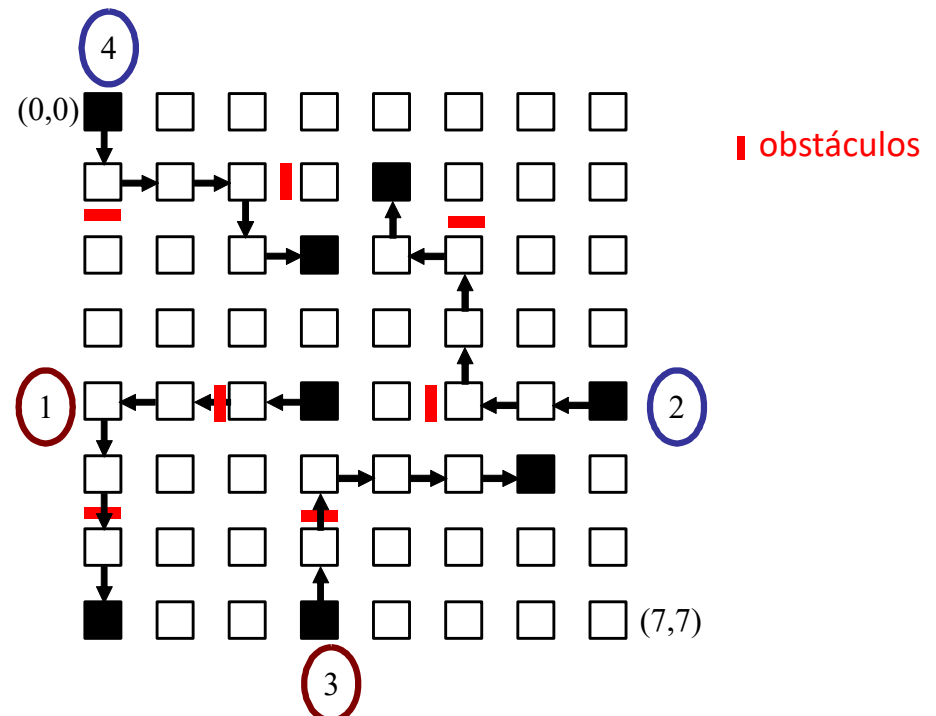
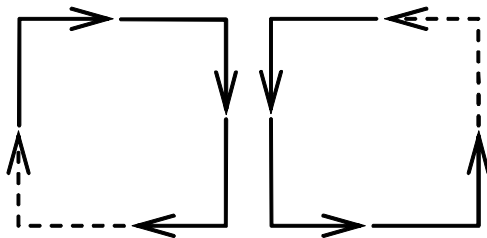


(2) e (3) – caminhos determinísticos

(1) e (4) – caminhos adaptativos

Negative-First routing algorithm

- packets are routed first in negative directions, i.e., to the North or to the West directions
 - if $(X_T \leq X_S \text{ and } Y_T \geq Y_S)$ or $(X_T \geq X_S \text{ and } Y_T \leq Y_S)$ packets are deterministically routed - paths 1 / 3
 - all other conditions allow some form of adaptive routing - paths 4 / 2

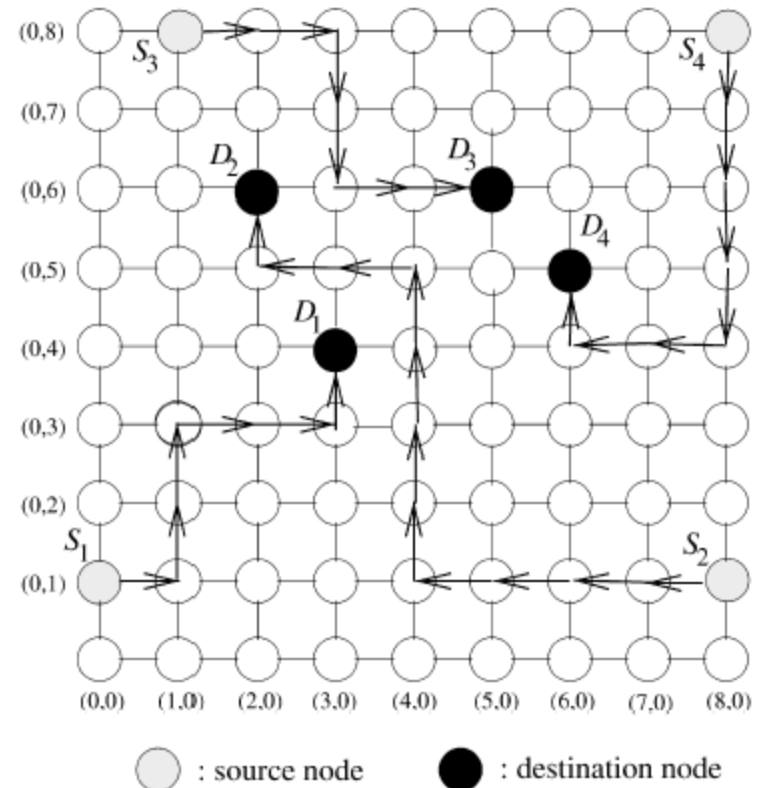


(1) e (3) – caminhos determinísticos

(2) e (4) – caminhos adaptativos

Odd-Even Wormhole Routing

- In the previous methods, at least half of S/D pairs are restricted to having one minimal path, while full adaptiveness is provided to the others
 - Unfair!
- Odd-even turn routing offers solution:
 - Even column: no EN or ES turn
 - Odd column: no NW or SW turn



Odd-Even Routing

Relative destination	Possible turns
NE	SE, WN
NW	SW , EN
SE	NE, WS
SW	NW , ES

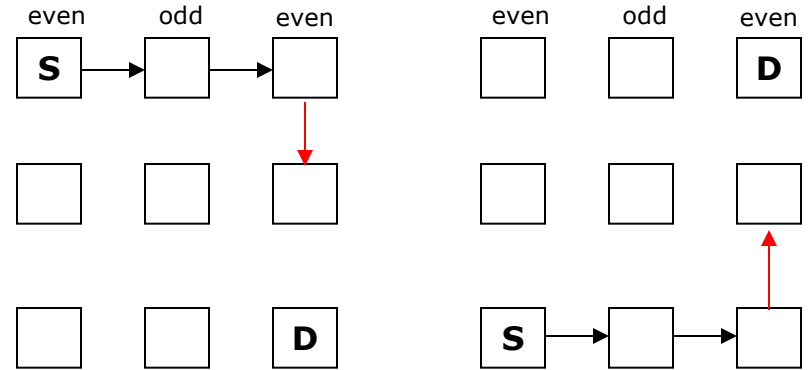
even col

odd col

On average, 2 routing options once for every 5 routes

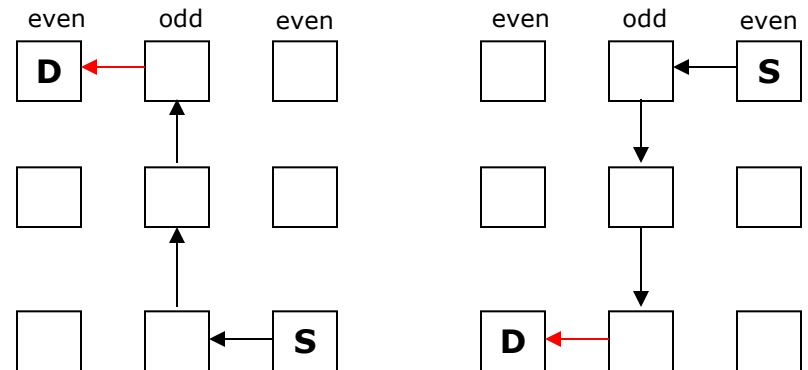
(1.2 opt/route)

when routing east and dest is in even col...



(don't go into dest. col unless row matches)

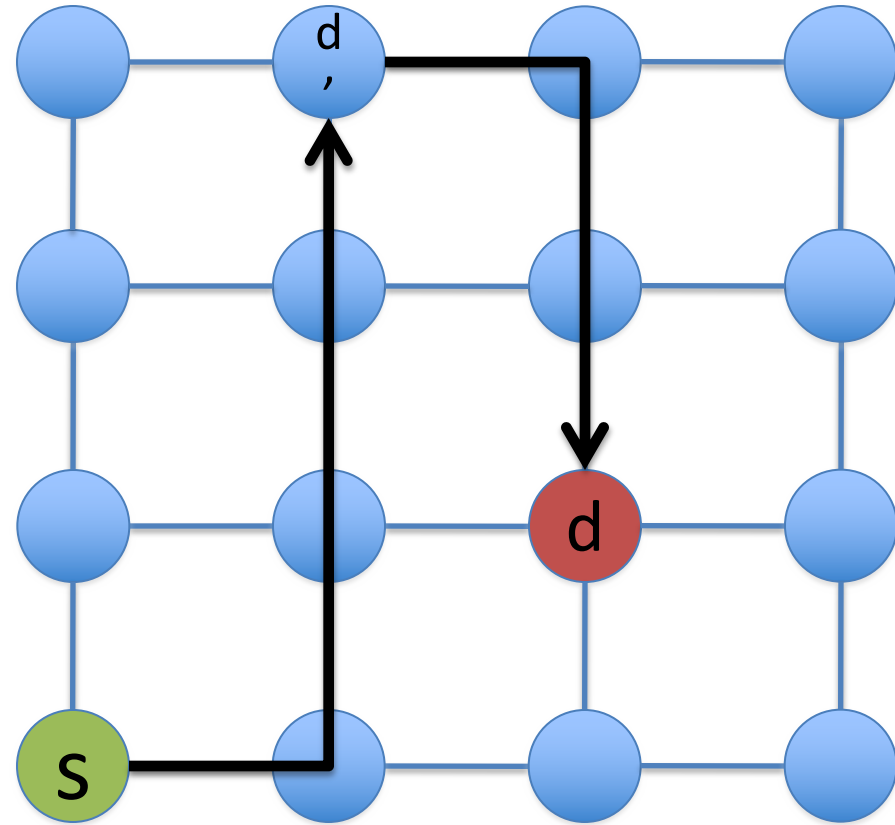
when routing west...



(don't go N/S in odd col)

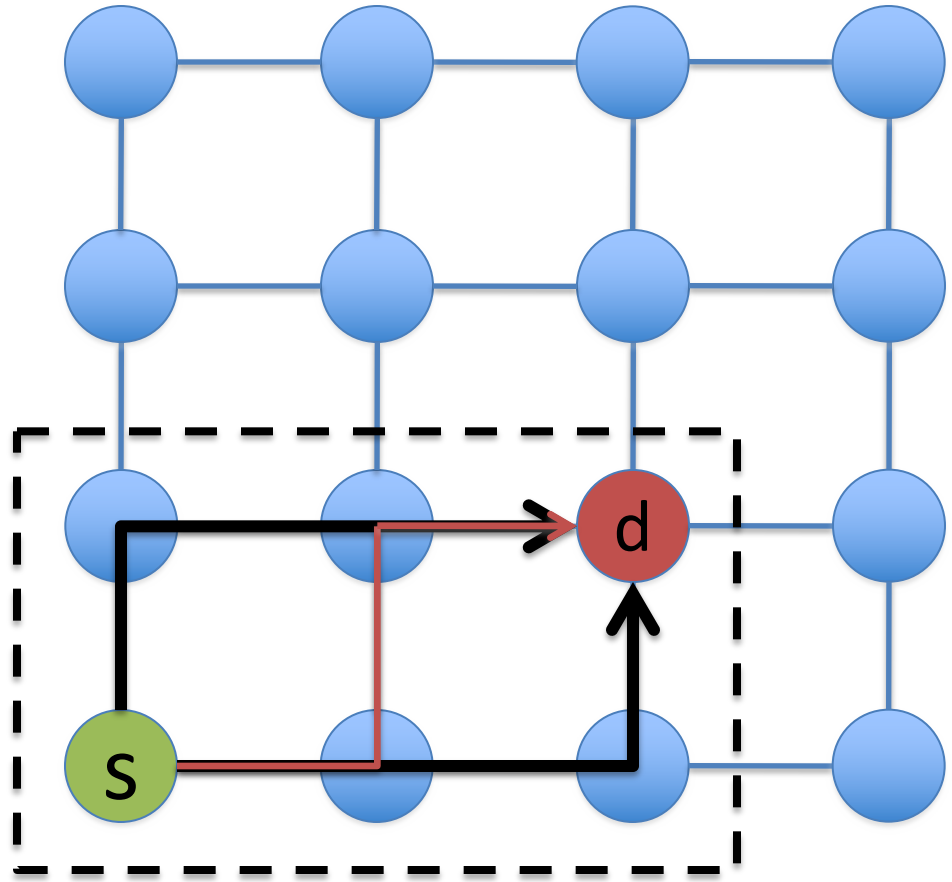
Valiant's Routing Algorithm

- To route from s to d , randomly choose intermediate node d'
 - Route from s to d' and from d' to d .
- Randomizes any traffic pattern
 - All patterns appear to be uniform random
 - Balances network load
- Non-minimal



Minimal Oblivious

- Valiant's: Load balancing comes at expense of significant hop count increase
 - Destroys locality
- Minimal Oblivious: achieve some load balancing, but use shortest paths
 - d' must lie within minimum quadrant
 - 6 options for d'
 - Only 3 different paths

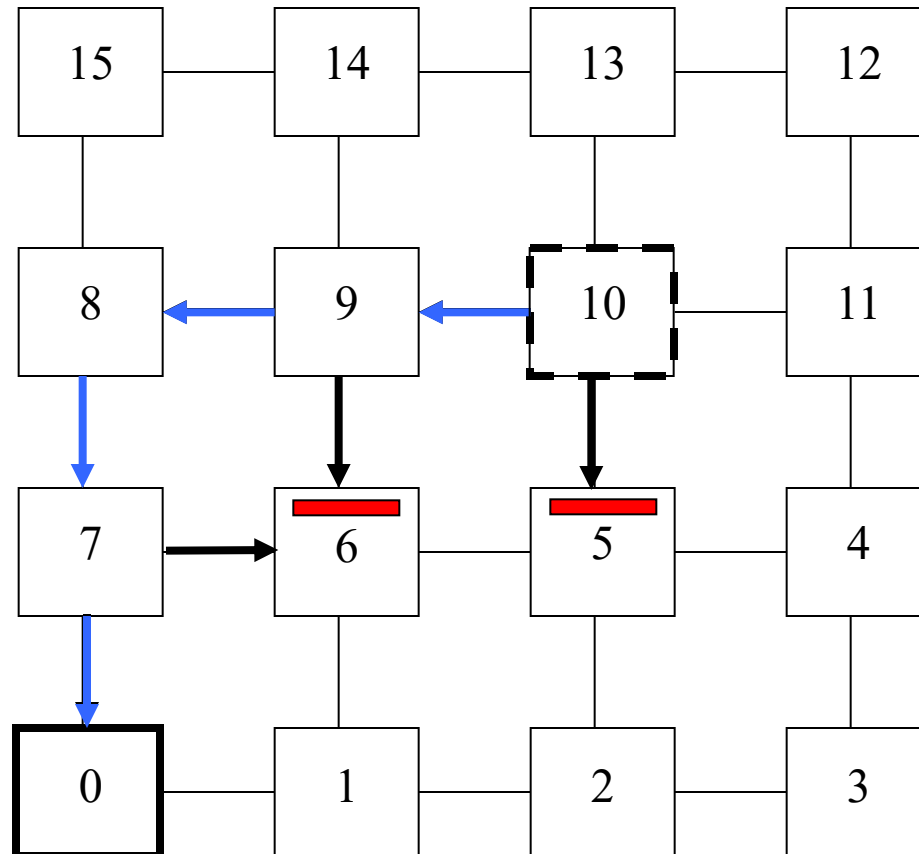


Rotamento Hamiltoniano

- Algoritmo de roteamento Hamiltoniano
 - Livre de *deadlock*
 - Simples de se obter uma versão determinística a partir da versão adaptativa
 - Suporte a algoritmos *multicast*
- Caminho Hamiltoniano
 - Caminho acíclico no qual é possível atingir todos os nodos de um grafo passando apenas uma vez em cada nodo
 - Roteadores rotulados de 0 a $N-1$
 - Caminho segue a ordem crescente ou decrescente dos rótulos

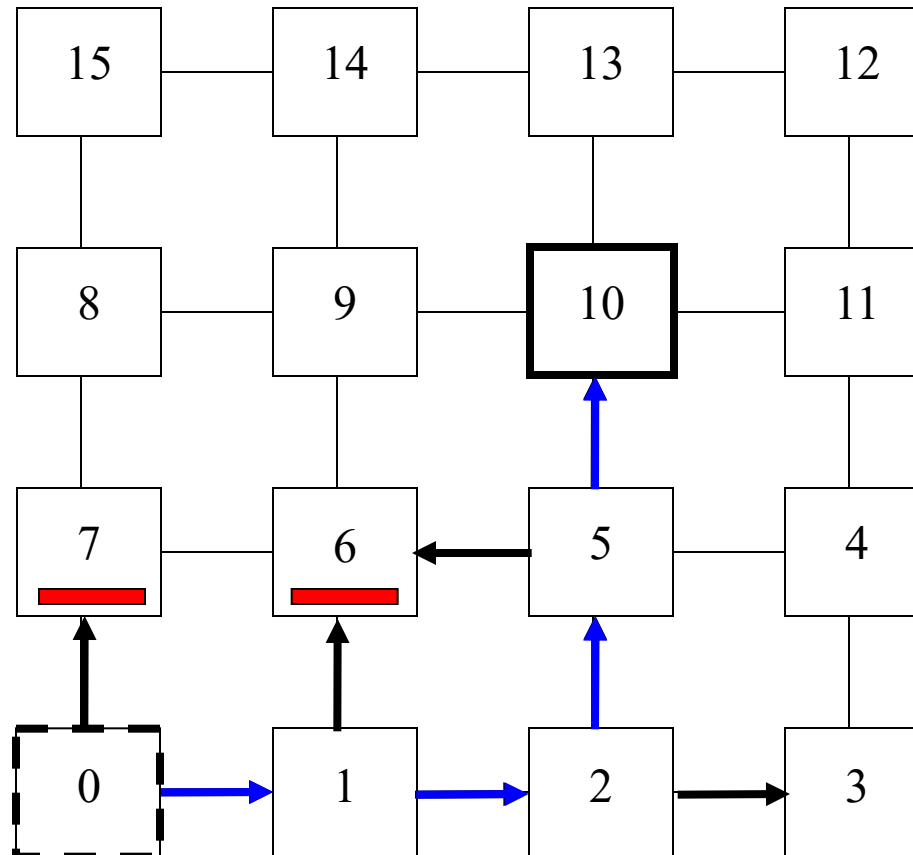
Serviço de comunicação com roteamento diferenciado

- Versão não-mínima parcialmente adaptativa



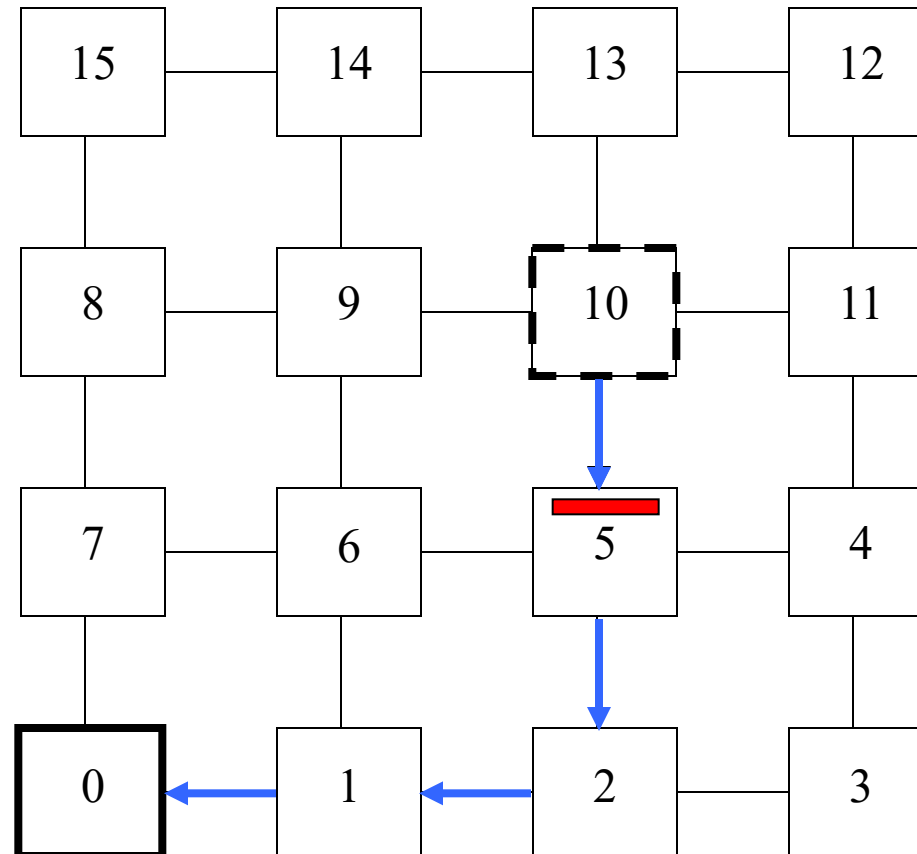
Serviço de comunicação com roteamento diferenciado

- Versão não-mínima parcialmente adaptativa



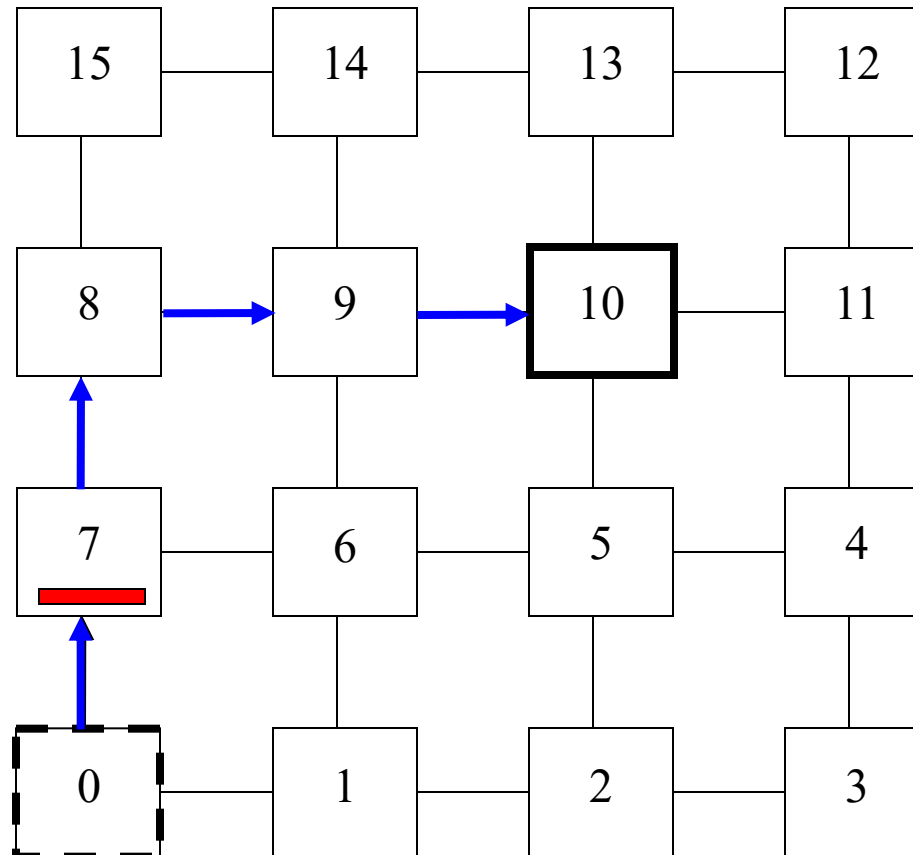
Serviço de comunicação com roteamento diferenciado

- Versão mínima determinística



Serviço de comunicação com roteamento diferenciado

- Versão mínima determinística

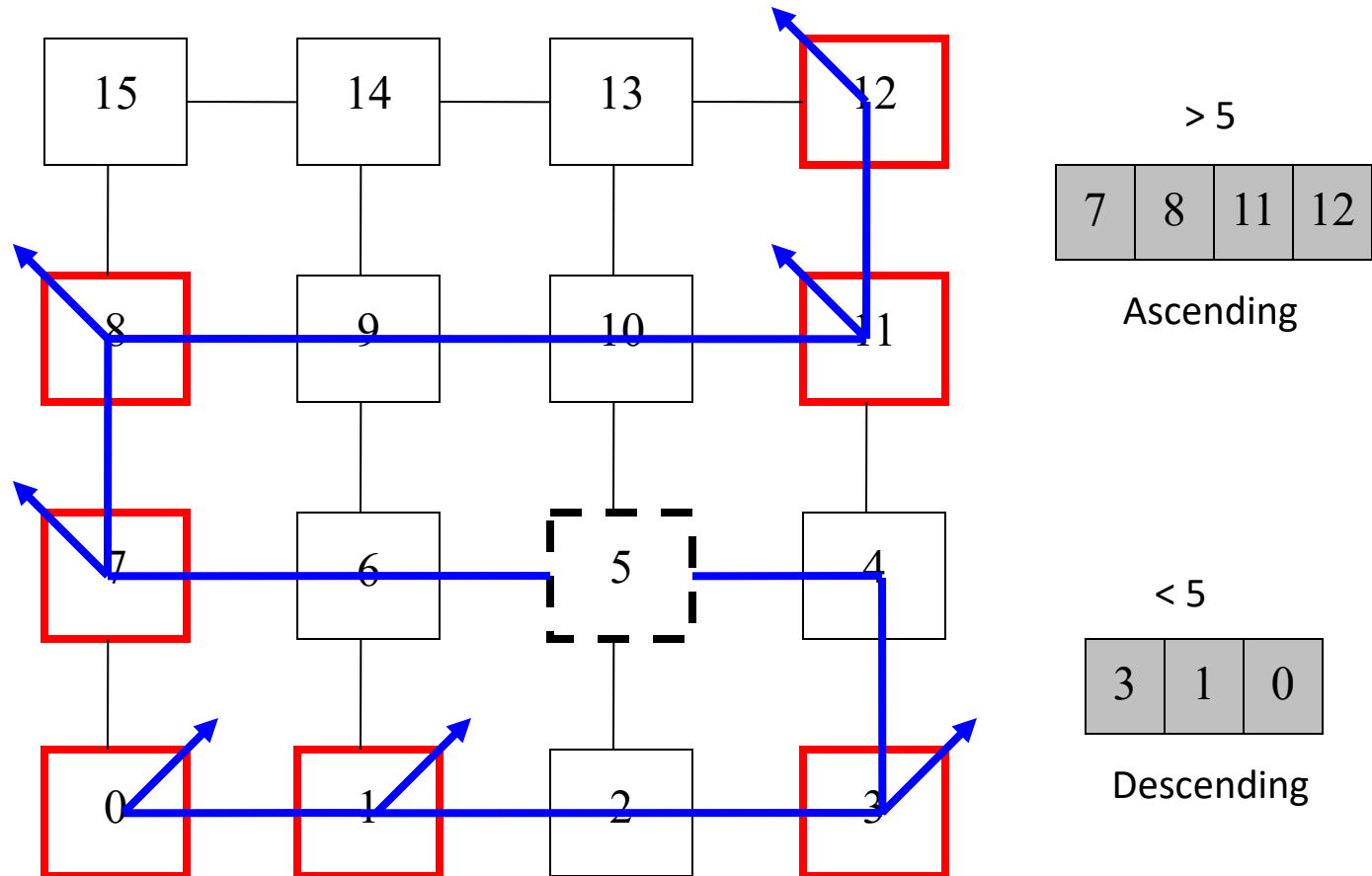


Collective communication service

- Dual-path multicast algorithm
 - Originally proposed for Multicomputers (1994)
 - 2D mesh topology
 - Hamiltonian routing algorithm
- Multicast messages
 - Parallel search algorithms
 - Blocked matrix multiplication
 - Cache coherence protocol
 - Control messages

Collective communication service

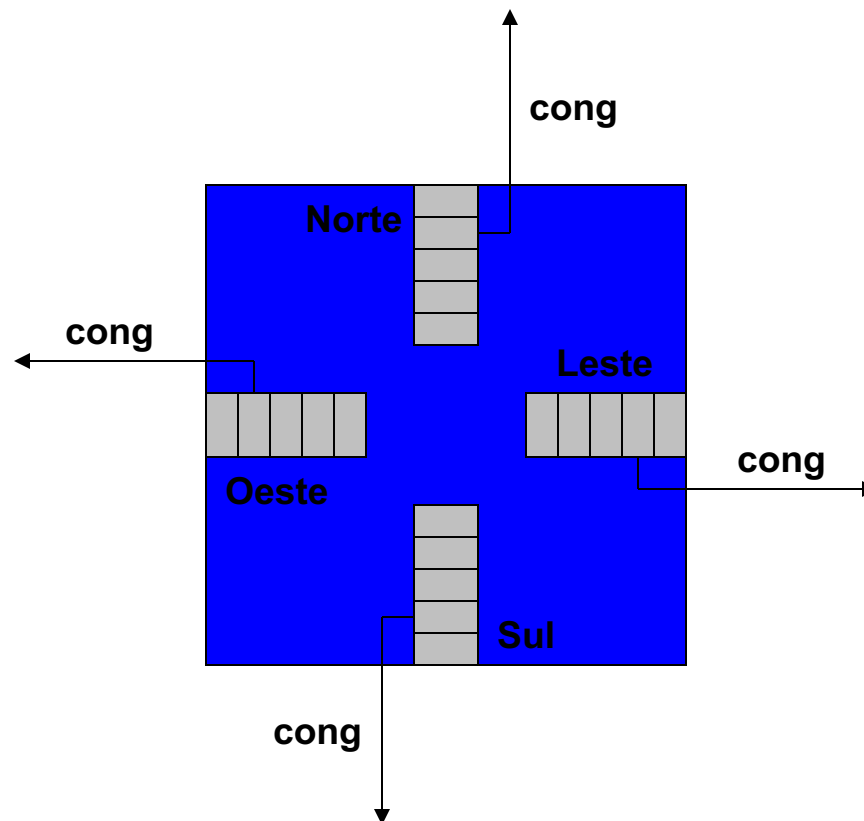
- Dual-path multicast algorithm
 - Targets set divided in 2 sub-sets
 - A message copy sent to each sub-set



Routing Adaptivity in Function of Traffic

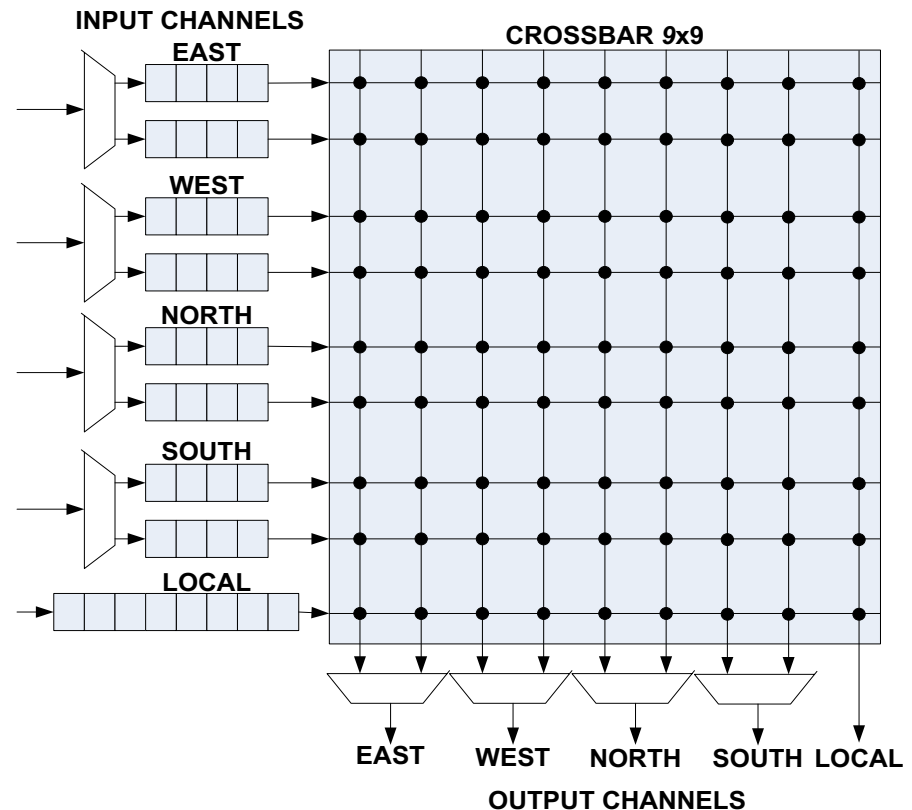
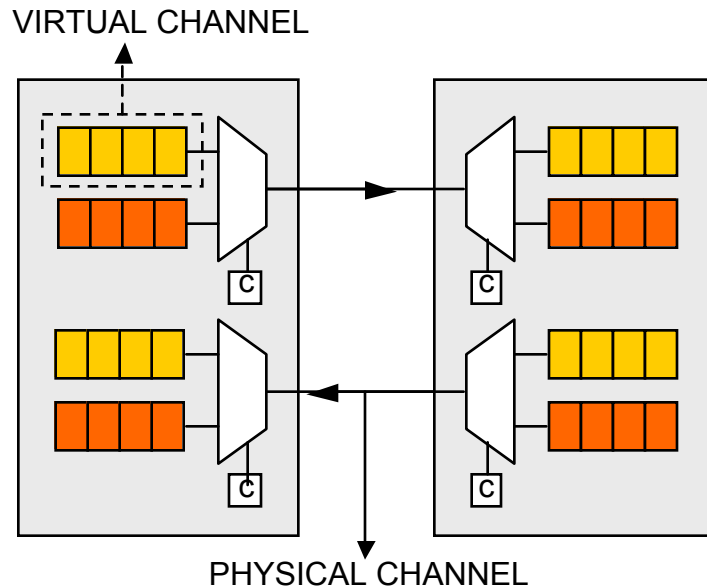
■ Congestion Signaling

- Single bit (congested/not congested)
- Bus (free slots, congestion levels)



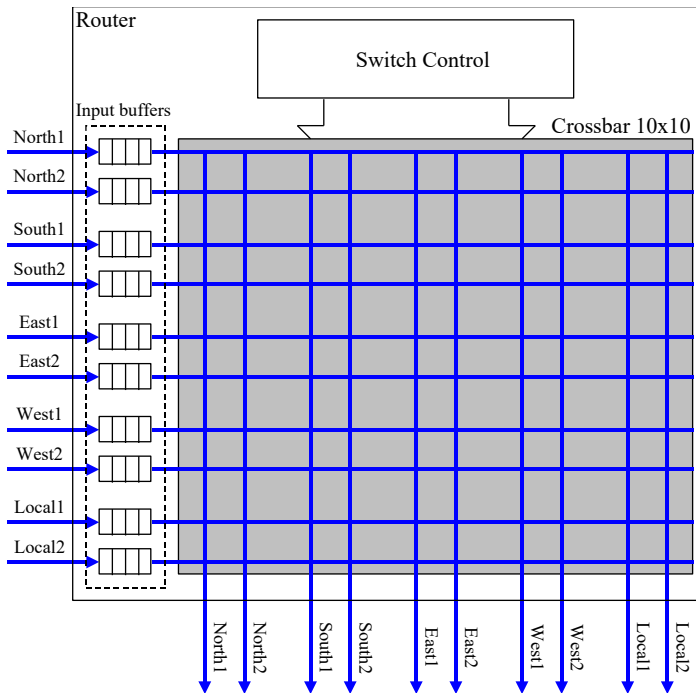
Fully Adaptive Routing with VCs

- Can achieve fully adaptive routing with VCs
 - Problem: minimize required number of VCs
 - Use of symmetrical routing algorithms



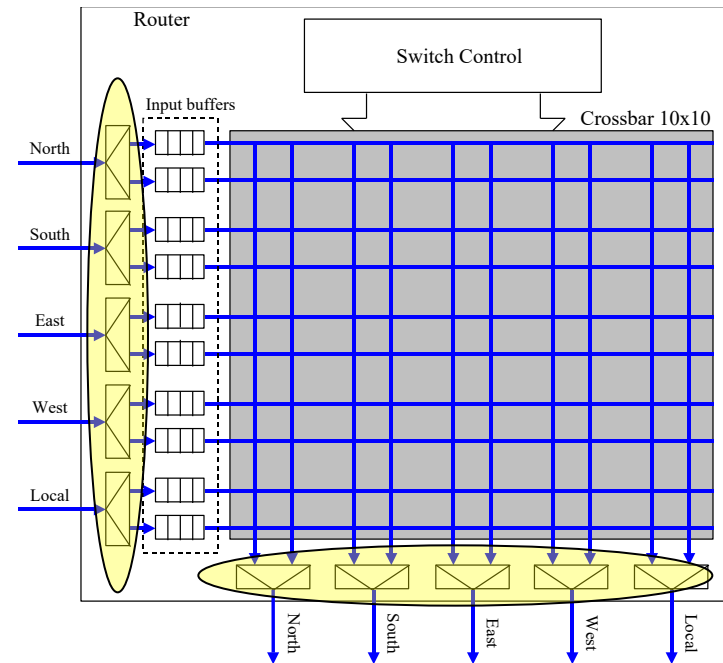
Replicated Channels x Virtual Channels

2 physical channels



X

2 virtual channels



- Smaller area: same crossbar, mux/demux suppressed

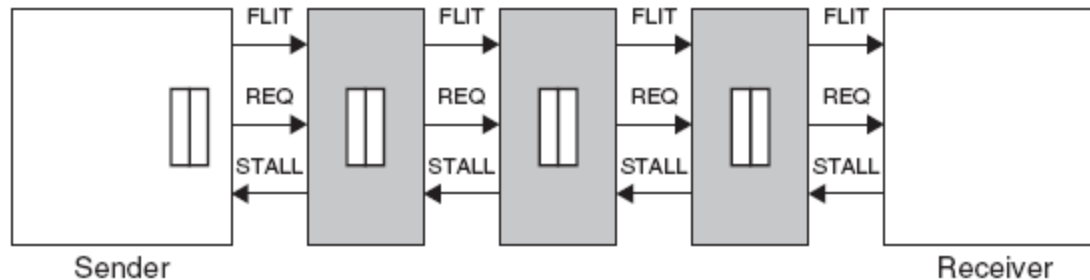
Outline

- Introduction
- NoC Topology
- Switching strategies
- Routing algorithms
- Flow control schemes
- Clocking schemes
- QoS
- NoC Architecture Examples

Flow control schemes

- Goal of flow control is to allocate network resources for packets traversing a NoC
 - can also be viewed as a problem of resolving contention during packet traversal
- At the data link-layer level, when transmission errors occur, recovery from the error depends on the support provided by the flow control mechanism
 - e.g. if a corrupted packet needs to be retransmitted, flow of packets from the sender must be stopped, and request signaling must be performed to reallocate buffer and bandwidth resources
- Most flow control techniques can manage link congestion
- But not all schemes can (by themselves) reallocate all the resources required for retransmission when errors occur
 - either error correction or a scheme to handle reliable transfers must be implemented at a higher layer

Flow control schemes



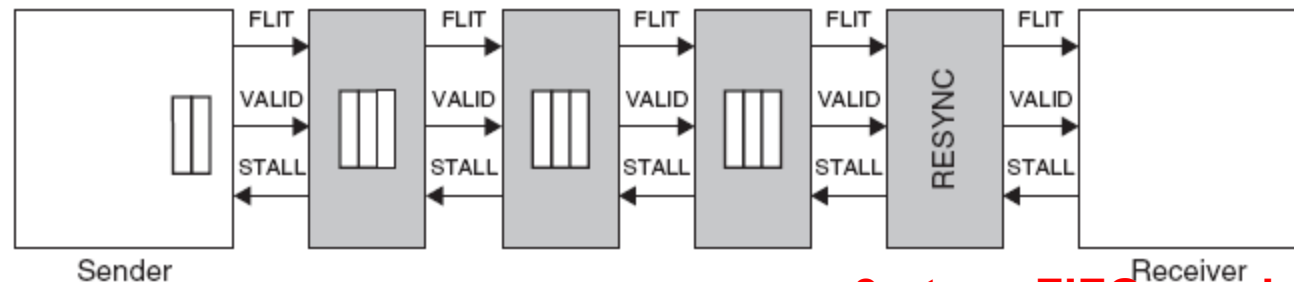
2-stage FIFO required

- **STALL/GO**

- low overhead scheme
- requires only two control wires
 - one going forward and signaling data availability
 - the other going backward and signaling either a condition of buffers filled (STALL) or of buffers free (GO)
- can be implemented with distributed buffering (pipelining) along link
- good performance – fast recovery from congestion
- does not have any provision for fault handling
 - higher level protocols responsible for handling flit interruption

Flow control schemes

- T-Error

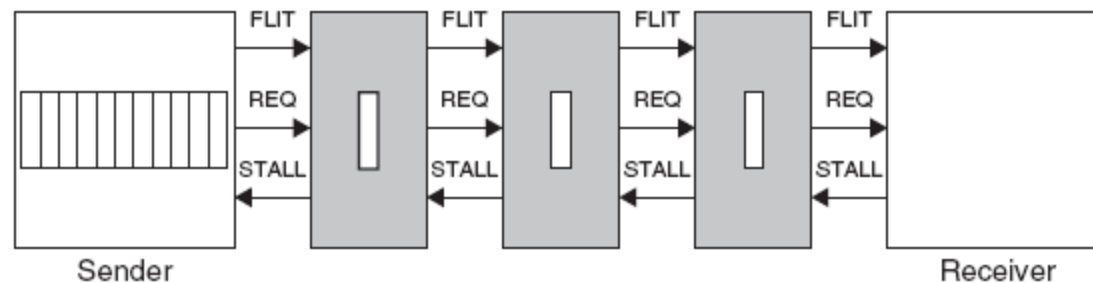


3-stage FIFO required

- more aggressive scheme that can detect faults
 - by making use of a **second delayed clock at every buffer stage**
- **delayed clock re-samples input data to detect any inconsistencies**
 - then emits a VALID control signal
- resynchronization stage added between end of link and receiving switch
 - to handle offset between original and delayed clocks
- timing budget can be used to provide greater reliability by configuring links with appropriate spacing and frequency
- does not provide a thorough fault handling mechanism

Flow control schemes

- ACK/NACK



- when flits are sent on a link, a **local copy** is kept in a buffer by sender
- when ACK received by sender, it deletes copy of flit from its local buffer
- when NACK is received, sender rewinds its output queue and starts resending flits, starting from the corrupted one
- implemented either end-to-end or switch-to-switch
- sender needs to have a buffer of size $2N + k$
 - N is number of buffers encountered between source and destination
 - k depends on latency of logic at the sender and receiver
- overall a minimum of $3N + k$ buffers are required
- fault handling support comes at cost of greater power, area overhead

Flow control schemes

- Network and Transport-Layer Flow Control
 - Flow Control without Resource Reservation
 - Technique #1: drop packets when receiver NI full
 - improves congestion in short term but increases it in long term
 - Technique #2: return packets that do not fit into receiver buffers to sender
 - to avoid deadlock, rejected packets must be accepted by sender
 - Technique #3: deflection routing
 - when packet cannot be accepted at receiver, it is sent back into network
 - packet does not go back to sender, but keeps hopping from router to router till it is accepted at receiver
 - Flow Control with Resource Reservation
 - credit-based flow control with resource reservation
 - credit counter at sender NI tracks free space available in receiver NI buffers
 - credit packets can piggyback on response packets
 - end-to-end or link-to-link

Outline

- Introduction
- NoC Topology
- Switching strategies
- Routing algorithms
- Flow control schemes
- **Clocking schemes**
- QoS
- NoC Architecture Examples

Clocking schemes

- Fully synchronous
 - single global clock is distributed to synchronize entire chip
 - hard to achieve in practice, due to process variations and clock skew
- Mesochronous
 - local clocks are derived from a global clock
 - not sensitive to clock skew
 - phase between clock signals in different modules may differ
 - deterministic for regular topologies (e.g. mesh)
 - non-deterministic for irregular topologies
 - synchronizers needed between clock domains
- Pleisochronous
 - clock signals are produced locally
- Asynchronous
 - clocks do not have to be present at all

Outline

- Introduction
- NoC Topology
- Switching strategies
- Routing algorithms
- Flow control schemes
- Clocking schemes
- QoS
- NoC Architecture Examples

Quality of Service (QoS)

- QoS refers to the level of commitment for packet delivery
 - refers to bounds on performance (bandwidth, delay, and jitter)
- Three basic categories
 - best effort (BE)
 - only correctness and completion of communication is guaranteed
 - usually packet switched
 - worst case times cannot be guaranteed
 - guaranteed service (GS)
 - makes a tangible guarantee on performance, in addition to basic guarantees of correctness and completion for communication
 - usually (virtual) circuit switched
 - differentiated service
 - prioritizes communication according to different categories
 - NoC switches employ priority based scheduling and allocation policies
 - cannot provide strong guarantees

Outline

- Introduction
- NoC Topology
- Switching strategies
- Routing algorithms
- Flow control schemes
- Clocking schemes
- QoS
- NoC Architecture Examples

Æthereal

- Developed by Philips
- Synchronous indirect network
- WH switching
- Contention-free source routing based on TDM
- GT as well as BE QoS
- GT slots can be allocated statically at initialization phase, or dynamically at runtime
- BE traffic makes use of non-reserved slots, and any unused reserved slots
 - also used to program GT slots of the routers
- Link-to-link credit-based flow control scheme between BE buffers
 - to avoid loss of flits due to buffer overflow

HERMES

- Developed at the Faculdade de Informática PUCRS, Brazil
- Direct network
- 2-D mesh topology
- WH switching with minimal XY routing algorithm
- 8 bit flit size; first 2 flits of packet contain header
- Header has target address and number of flits in the packet
- Parameterizable input queuing
 - to reduce the number of switches affected by a blocked packet
- Connectionless: cannot provide any form of bandwidth or latency GS

MANGO

- Message-passing Asynchronous Network-on-chip providing GS over open core protocol (OCP) interfaces
- Developed at the Technical University of Denmark
- Clockless NoC that provides BE as well as GS services
- NIs (or adapters) convert between the synchronous OCP domain and asynchronous domain
- Routers allocate separate physical buffers for VCs
 - For simplicity, when ensuring GS
- BE connections are source routed
 - BE router uses credit-based buffers to handle flow control
 - length of a BE path is limited to five hops
- static scheduler gives link access to higher priority channels
 - admission controller ensures low priority channels do not starve

Nostrum

- Developed at KTH in Stockholm
- Direct network with a 2-D mesh topology
- SAF switching with hot potato (or deflective) routing
- Support for
 - switch/router load distribution
 - guaranteed bandwidth (GB)
 - multicasting
- GB is realized using looped containers
 - implemented by VCs using a TDM mechanism
 - container is a special type of packet which loops around VC
 - multicast: simply have container loop around on VC having recipients
- Switch load distribution requires each switch to indicate its current load by sending a stress value to its neighbors

Octagon

- Developed by STMicroelectronics
- direct network with an octagonal topology
- 8 nodes and 12 bidirectional links
- Any node can reach any other node with a max of 2 hops
- Can operate in packet switched or circuit switched mode
- Nodes route a packet in packet switched mode according to its destination field
 - node calculates a relative address and then packet is routed either left, right, across, or into the node
- Can be scaled if more than 8 nodes are required
 - Spidergon

QNoC

- Developed at Technion in Israel
- Direct network with an irregular mesh topology
- WH switching with an XY minimal routing scheme
- Link-to-link credit-based flow control
- Traffic is divided into four different service classes
 - signaling, real-time, read/write, and block-transfer
 - signaling has highest priority and block transfers lowest priority
 - every service level has its own small buffer (few flits) at switch input
- Packet forwarding is interleaved according to QoS rules
 - high priority packets able to preempt low priority packets
- Hard guarantees not possible due to absence of circuit switching
 - Instead statistical guarantees are provided

SOCBUS

- Developed at Linköping University
- Mesochronous clocking with signal retiming is used
- Circuit switched, direct network with 2-D mesh topology
- Minimum path length routing scheme is used
- Circuit switched scheme is
 - deadlock free
 - requires simple routing hardware
 - very little buffering (only for the request phase)
 - results in low latency
- Hard guarantees are difficult to give because it takes a long time to set up a connection

SPIN

- Scalable programmable integrated network (SPIN)
- fat-tree topology, with two one-way 32-bit link data paths
- WH switching, and deflection routing
- Virtual socket interface alliance (VSIA) virtual component interface (VCI) protocol to interface between PEs
- Flits of size 4 bytes
- First flit of packet is header
 - first byte has destination address (max. 256 nodes)
 - last byte has checksum
- Link level flow control
- Random hiccups can be expected under high load
 - GS is not supported

Xpipes

- Developed by the Univ. of Bologna and Stanford University
- Source-based routing, WH switching
- Supports OCP standard for interfacing nodes with NoC
- Supports design of heterogeneous, customized (possibly irregular) network topologies
- go-back-N retransmission strategy for link level error control
 - errors detected by a CRC (cycle redundancy check) block running concurrently with the switch operation
- XpipesCompiler and NetChip compilers
 - Tools to tune parameters such as flit size, address space of cores, max. number of hops between any two network nodes, etc.
 - generate various topologies such as mesh, torus, hypercube, Clos, and butterfly