Introduction to Network on Chip (1)

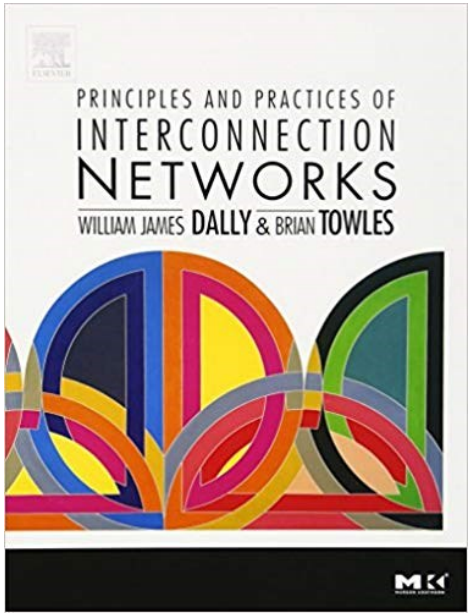
Kun-Chih (Jimmy) Chen 陳坤志

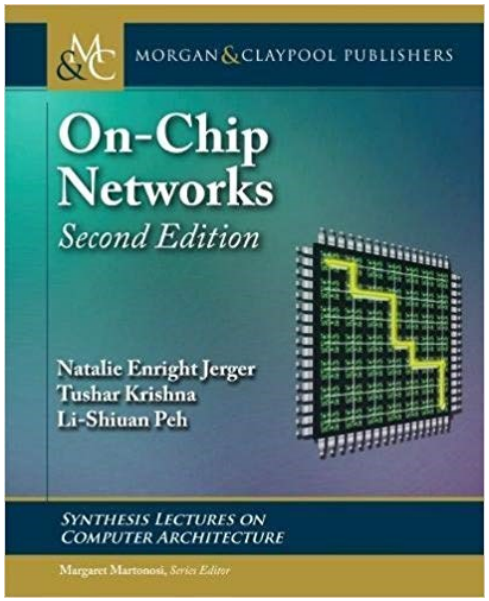
kcchen@nycu.edu.tw

Institute of Electronics,
National Yang Ming Chiao Tung University

Reference Books

- **Reference**
  - William J. Dally and B. Towles, ***Principles and Practices of Interconnection Networks***, Elsevier, 2004. (ISBN: 978-0122007514)
  - Natalie E. Jerger and L.-S. Peh, ***On-Chip Networks*****,** Morgan & Claypool, 2009. (ISBN: 978-1598295849)





NoC - Part 1

- Application requirements
- NoC: A paradigm shift in VLSI Design
- Critical problems addressed by NoC
- Traffic abstractions
- Data abstraction
- Network delay modeling

Wire Delay vs. Logic Delay

Operation	Delay (.13mico)	Delay (.05micro)
32-bit ALU Operation	650ps	250ps
32-bit Register read	325ps	125ps
Read 32-bit from 8KB RAM	780ps	300ps
Transfer 32-bit across chip (10mm)	**1400ps**	**2300ps**
Transfer 32-bit across chip (200mm)	2800ps	4600ps

2:1 global on-chip communication to operation delay

9:1 in 2010

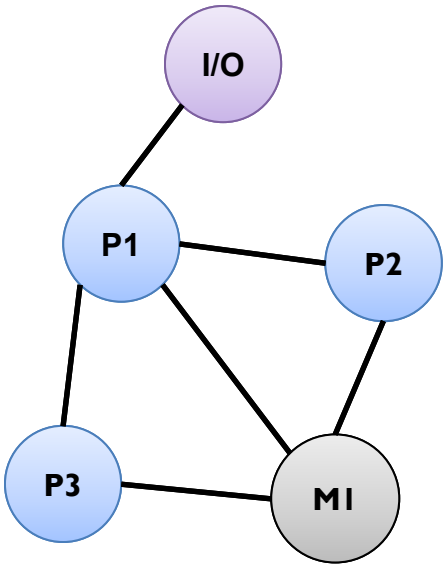
Ref: W.J. Dally HPCA Panel presentation 2002

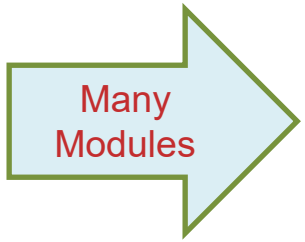
Communication Reliability

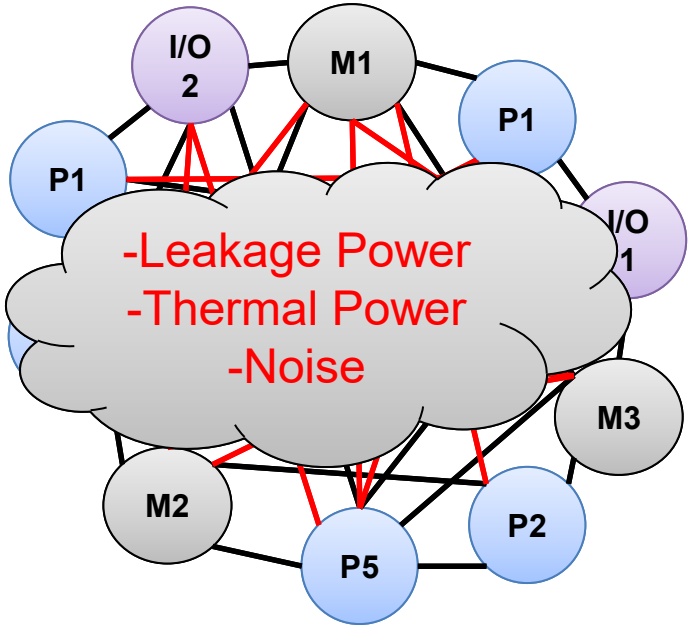
- Information transfer is inherently unreliable at the electrical level, due to:
  - Timing errors
  - Cross-talk
  - Electro-magnetic interference (EMI)
  - Soft errors

- The problem will get increasingly worse as technology scales down

On-chip Interconnection Types

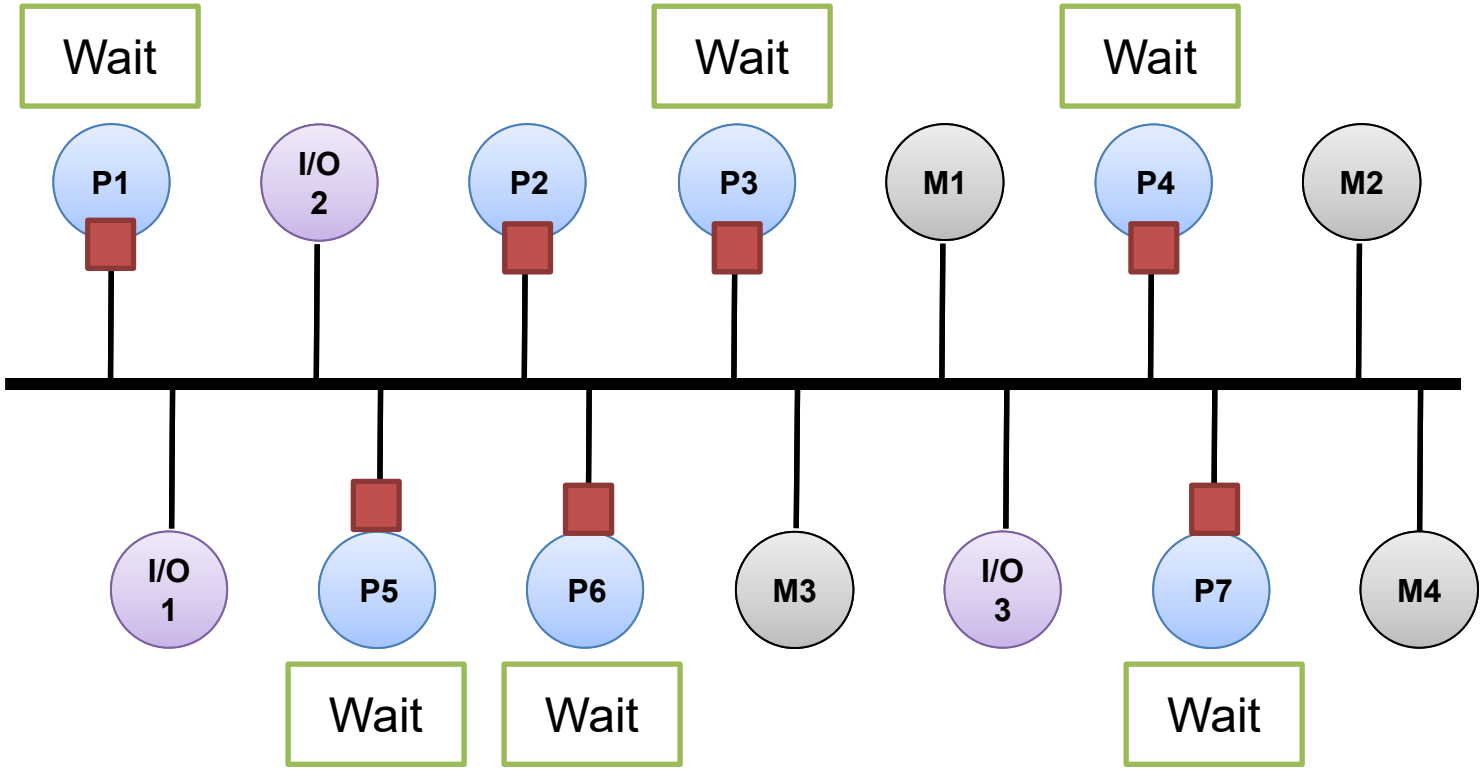






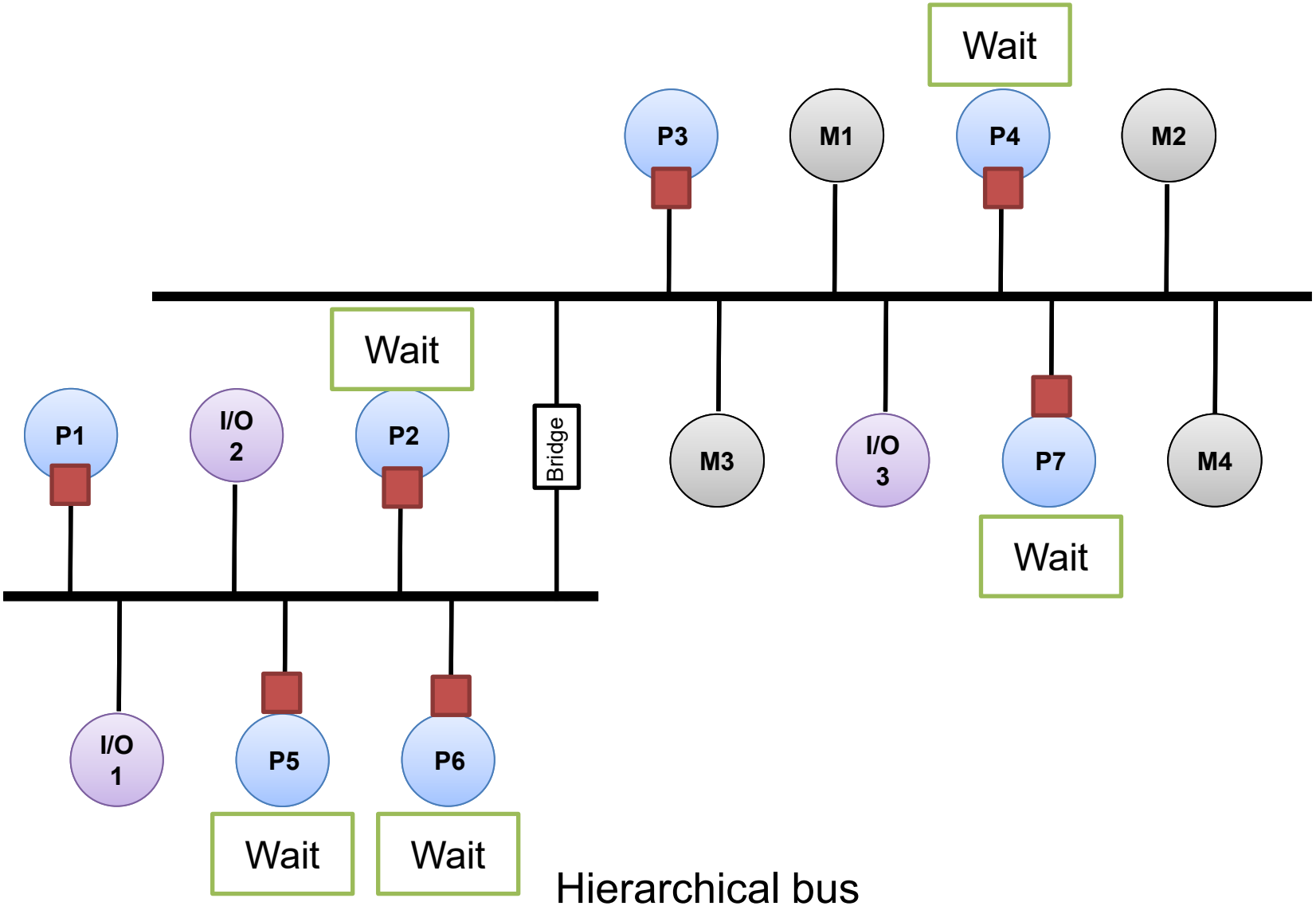
Point-to-Point

On-chip Interconnection Types



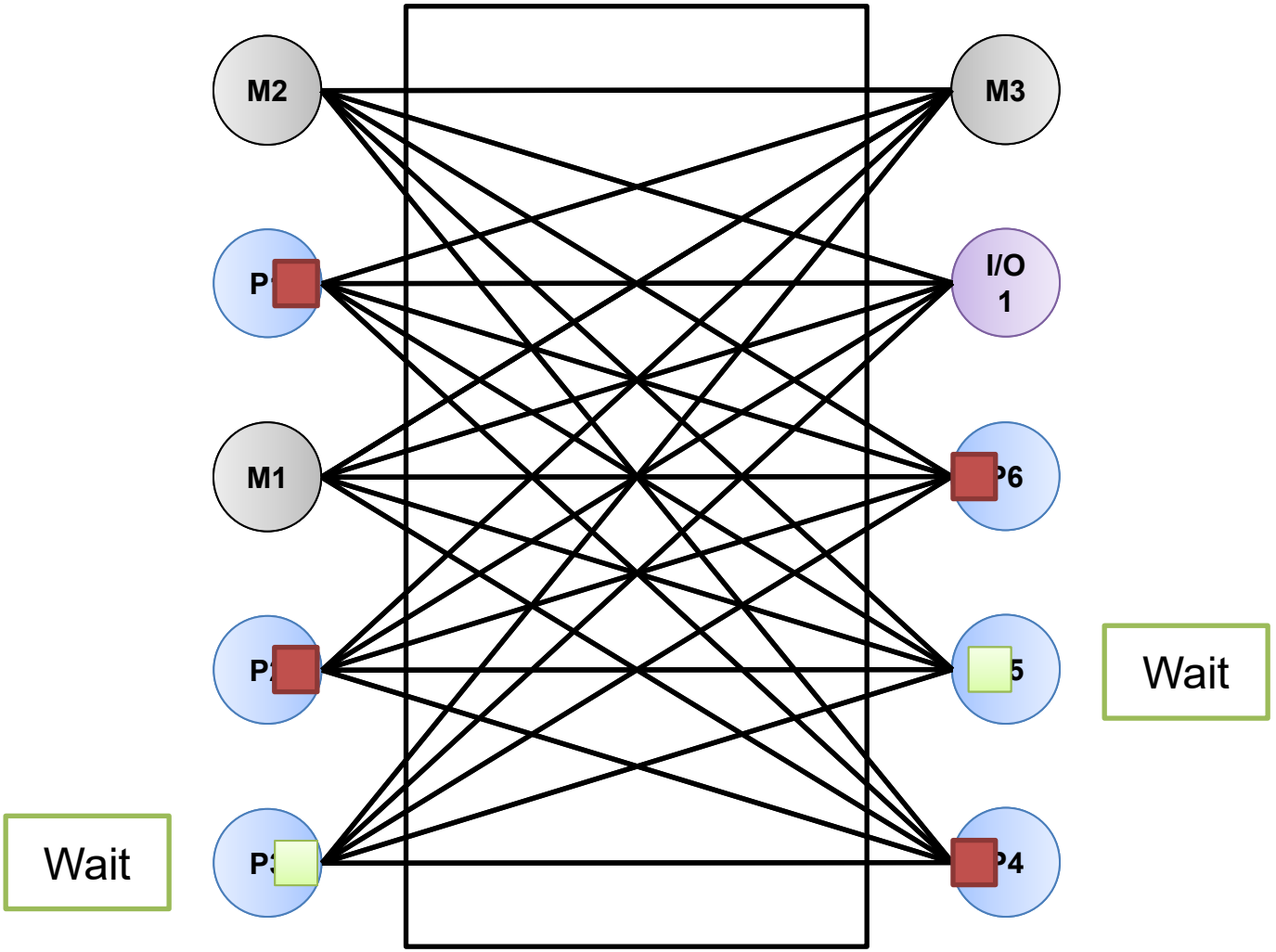
Shared bus

On-chip Interconnection Types



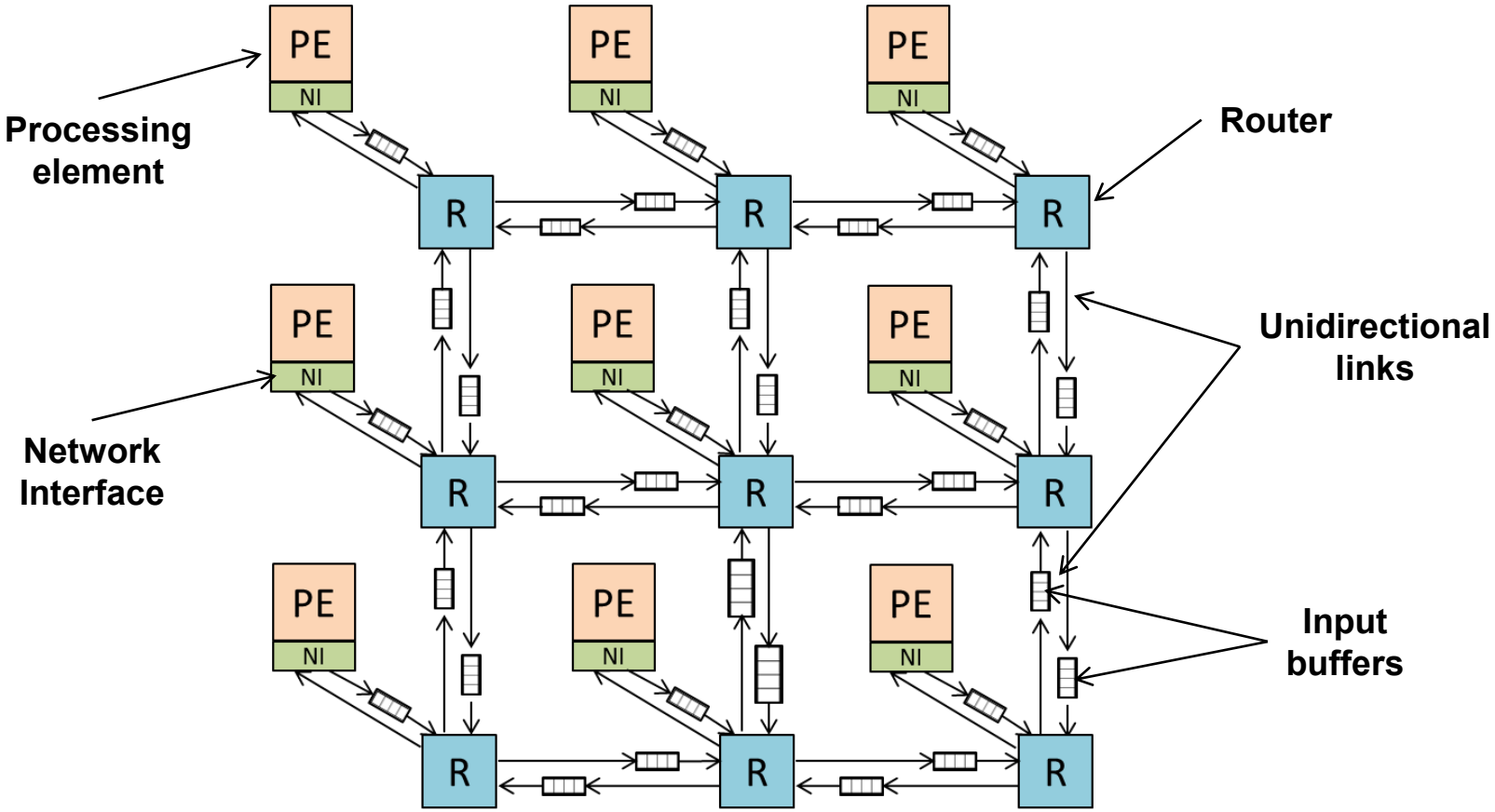
Hierarchical bus

On-chip Interconnection Types



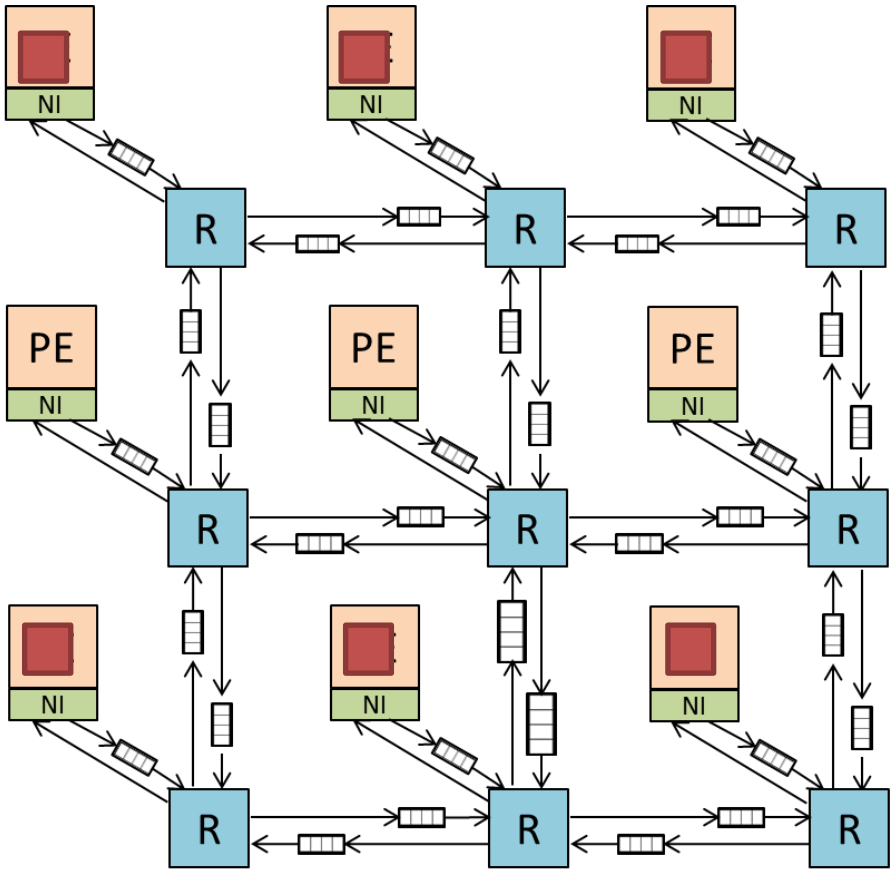
Bus matrix

On-chip Interconnection Types



Network-on-Chip -> our main topic in this lecture.

On-chip Interconnection Types



Network-on-Chip -> our main topic in this lecture

Seminal Works in Early 2000

Route Packets, Not Wires: On-Chip Interconnection Networks

William J. Dally and Brian Towles
Computer Systems Laboratory
Stanford University

DAC 2001
Google citation: 3768

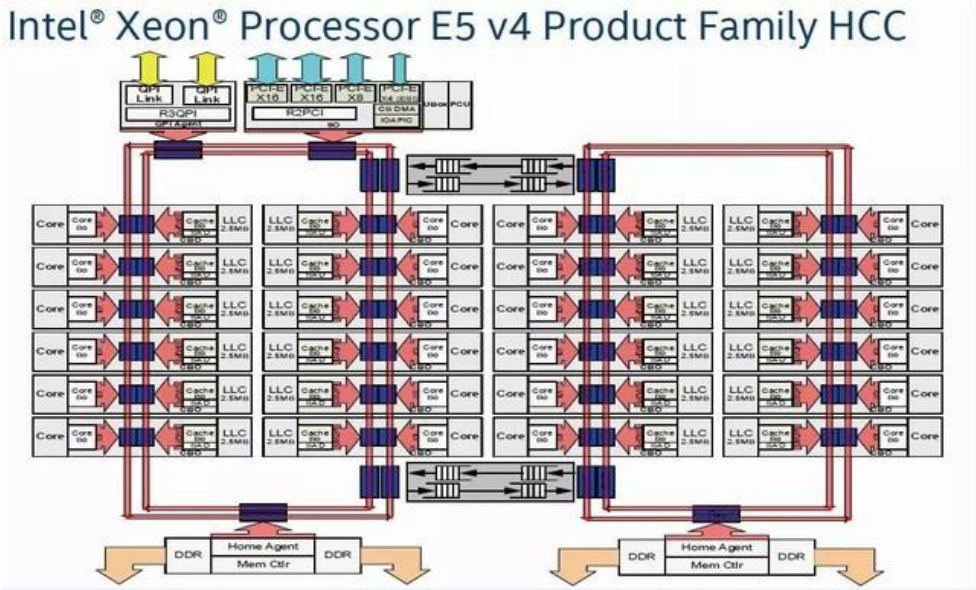
Networks on Chips: A New SoC Paradigm

IEEE Computer
Jan. 2002
Google citation: 4127

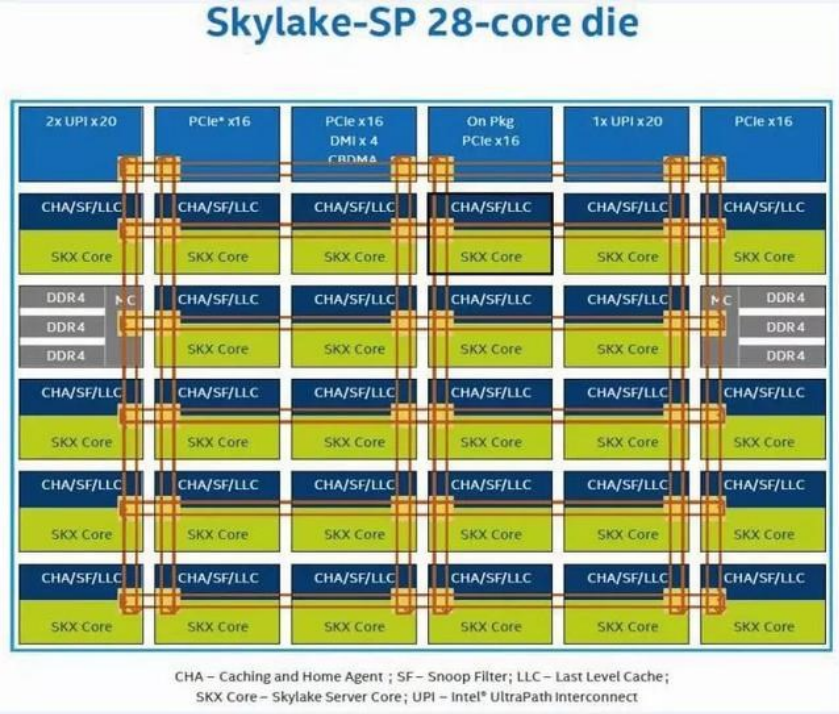
On-chip micronetworks, designed with a layered methodology, will meet the distinctive challenges of providing functionally correct, reliable operation of interacting system-on-chip components.

- In existing high-performance prototyping & commercial chips, **mesh-based topology** is mostly adopted for its scalability.

Design Example 1:
Intel i9 processor, 2017



Ring version



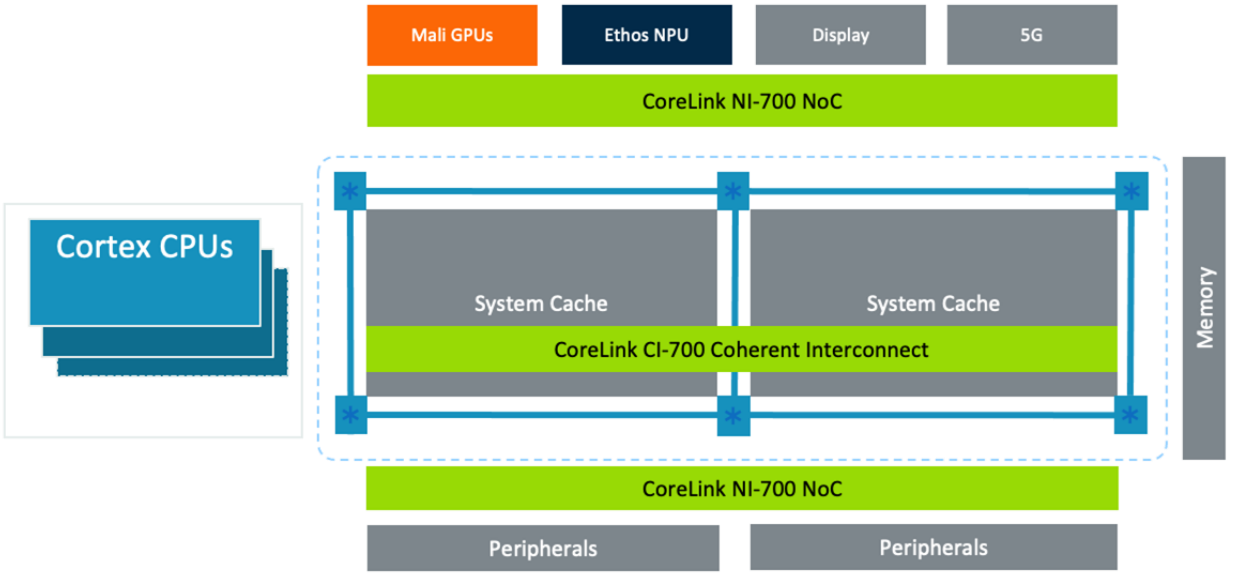
Mesh version

Example 2:
AMD/Xilinx Versal platform, 2019

- Adaptive Compute Acceleration Platform (ACAP)
  - New name of “programmable logic”
  - Connect each SoC component via NoC interconnection
  - Target the 5G and AI application

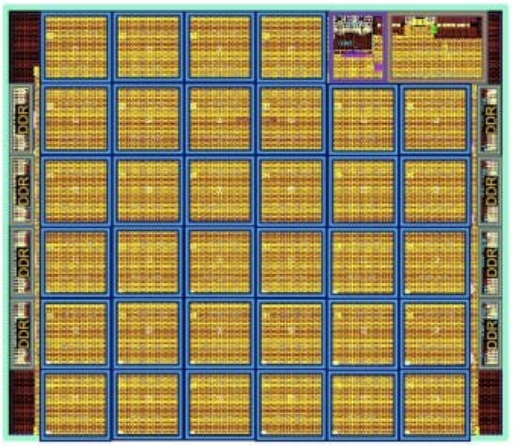
Example 3:
ARM CoreLink NI-700, 2021

- ARM CoreLink NI-700,
  - A new flexible packetized network-on-chip interconnect for both high-bandwidth accelerators
  - Scalable and highly configurable network-on-chip (NoC)
  - all the AMBA transactions are converted to a packetized format
    - Reduce the wire count by 30% on average

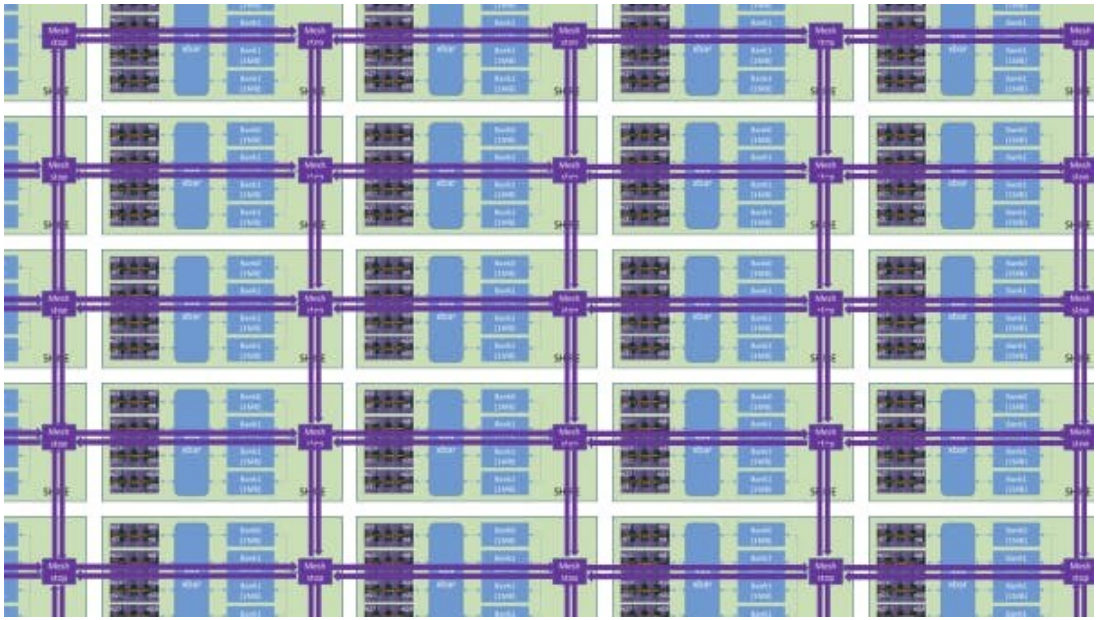


Example 4:
Esperanto ET-SoC-1, 2021

- Esperanto ET-SoC-1
  - 1093 RISC-V AI Accelerator
  - The first AI processor in Esperanto Inc.

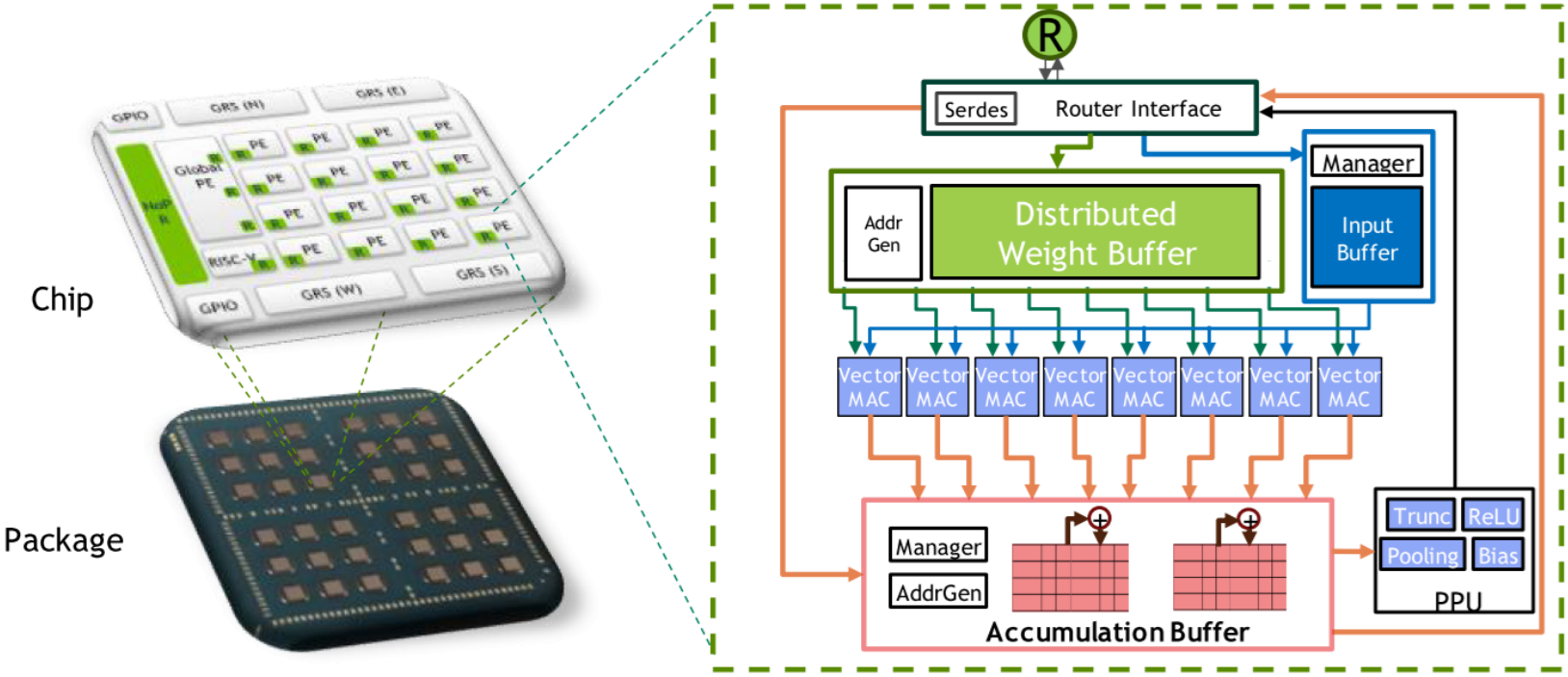


- TSMC 7nm
- 4 ET-Maxion
  (superscalar out-of-order core)
- 1089 ET-Minion
  (in-order multithreaded core)
- 32GB DRAM
- 137GB/sec memory bandwidth
- 256-bit wide interface



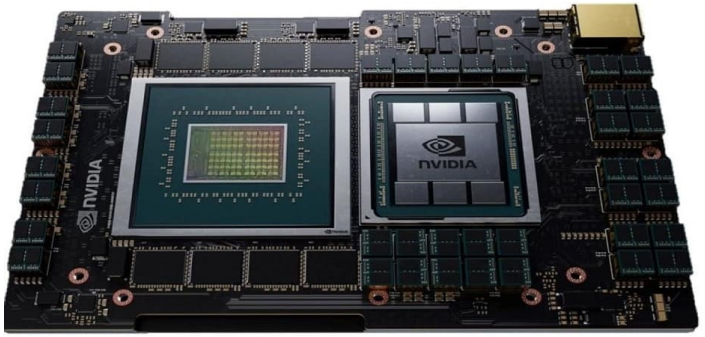
Design Example 5: NVIDIA Muti-Chip-Module (MCM), 2019

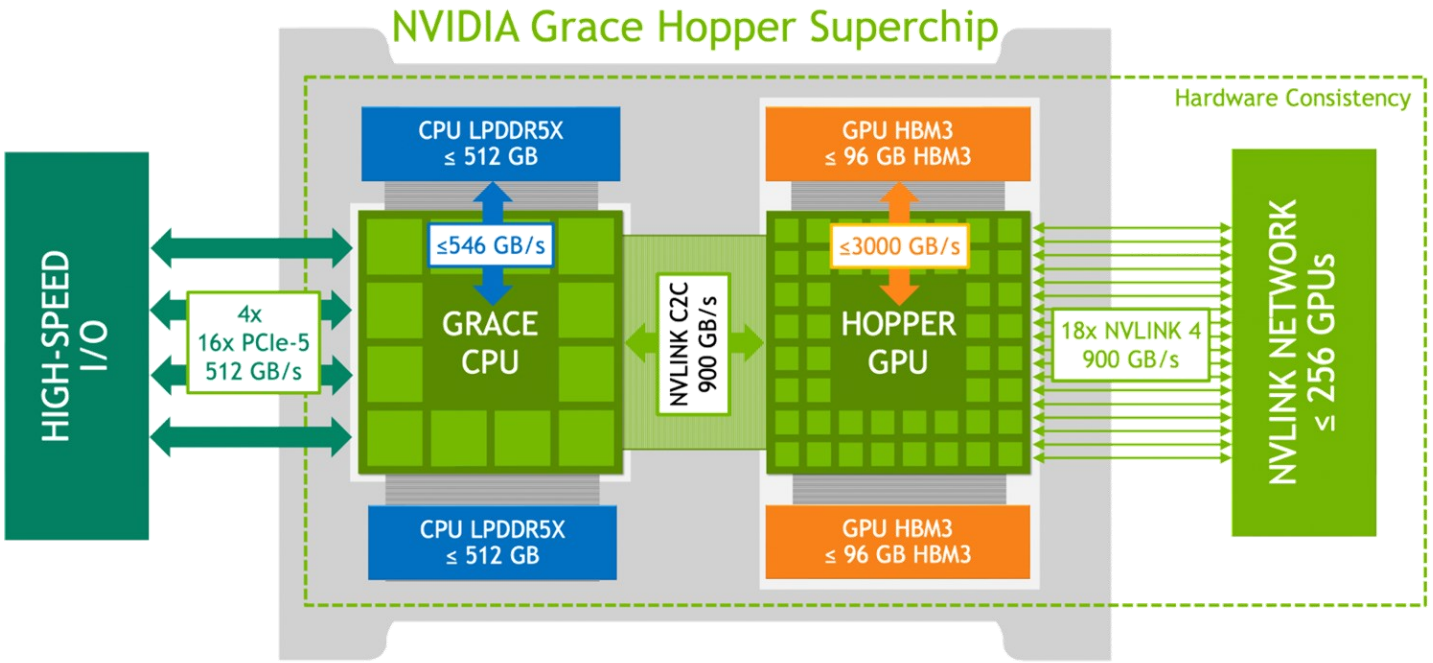
- Multi-Chip-Module (MCM)
  - Hierarchical NoC interconnection
    - 4x5 mesh topology connects 16 PEs, 1 global PE, and 1 RISC-V
    - 6x6 mesh topology connects 36 chips in package



Design Example 6: NVIDIA Grace-Hopper, 2022

- NVIDIA Grace CPU
  - 72 ARM Neoverse V2 core
- Chiplet with NVLink4.0 protocol





Design Example 7: Arteris's FlexNoC IP, 2017

Neural Network and Machine Learning SoCs

PUBLICLY-ANNOUNCED COMPANIES USING ARTERISIP FOR ML, AI AND NEURAL NETWORK SOCS

		FlexNoC Non-Coherent Interconnect	Ncore Cache Coherent Interconnect	ISO 26262 Functional Safety Mechanisms (Resilience Package)	Information Links
Movidius	Movidius (Intel)	✓			More Info
Mobileye	Mobileye (Intel)	✓		✓	More Info
NXP	NXP	✓	✓	✓	More Info
Toshiba	Toshiba	✓	✓	✓	More Info
HUAWEI HiSilicon	HiSilicon (Huawei)	✓			More Info
Cambricon	Cambricon	✓			More Info
Dream Chip	Dream Chip	✓		✓	More Info
nextchip	Nextchip	✓			More Info
intellifusion	Intellifusion	✓			More Info

ARTERISIP

Copyright © 2017 Arteris | 15

News:
NoC is a good solution in MPSoC market!

EE Times

MY CART

HOME NEWS PERSPECTIVES DESIGNLINES PODCAST EDUCATION STORE SPECIAL PROJ

DESIGNLINES | SOC DESIGNLINE

Qualcomm Buys Arteris Tech, Team

By Rick Merritt 10.31.2013 0

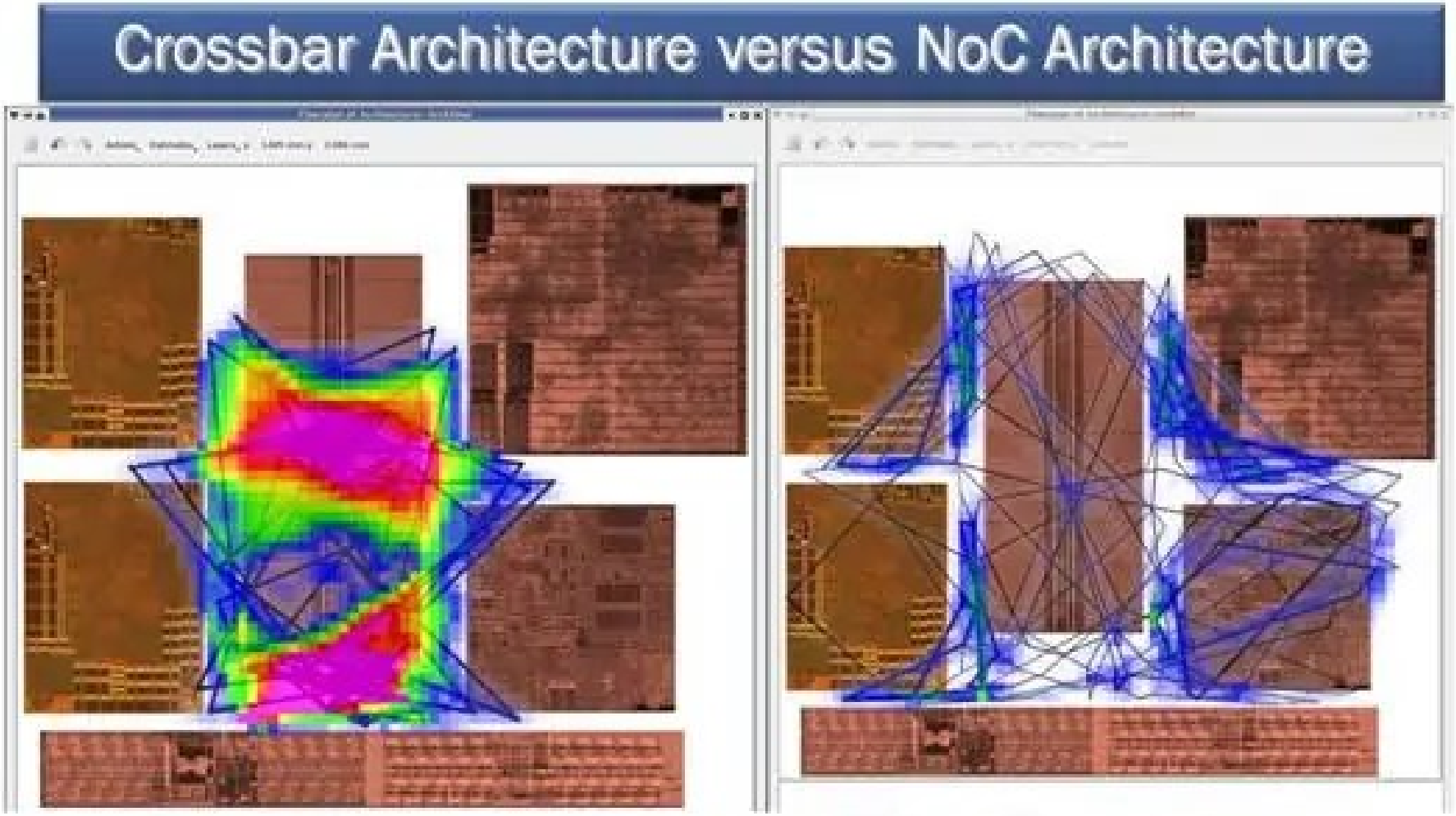
Share Post Share on Facebook Share on Twitter

#1 eFPGA: most nodes, sizes, flexibility, customers

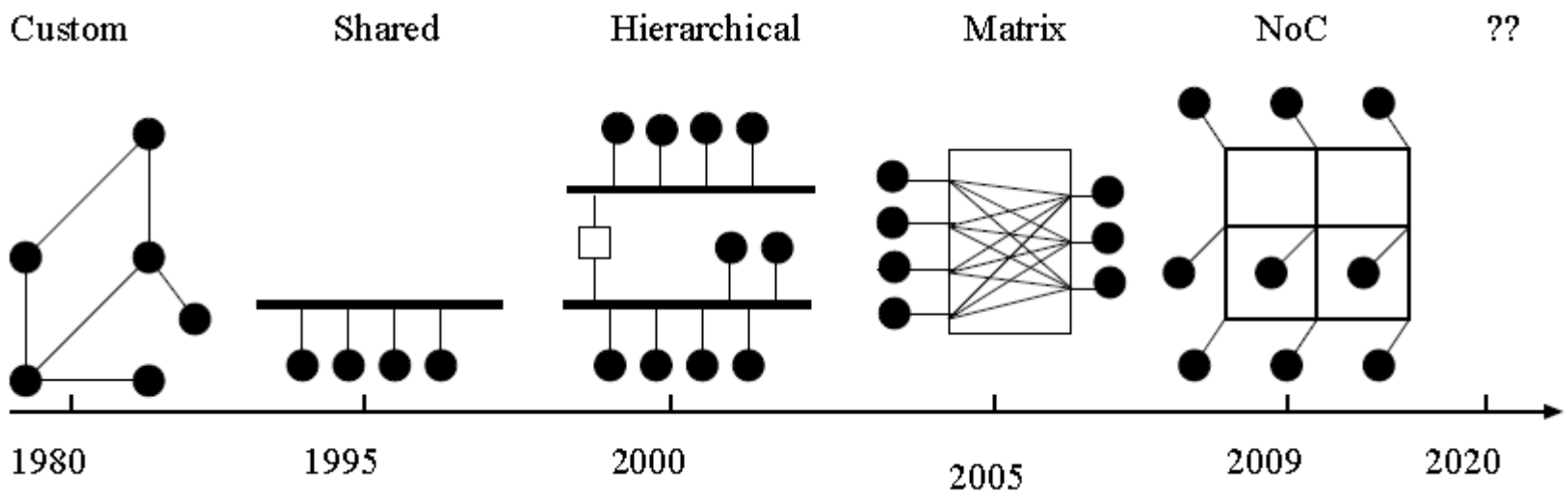
flexlogix AI + eFPGA ®

SAN JOSE, Calif. — Can you maintain a business after your top customer buys your technology and the engineering team that developed it? That's the question Arteris will attempt to answer after Qualcomm decided it liked the FlexNoc network-on-chip so much it bought the company — or most of it.

News:
Qualcomm evaluated FlexNoC in 2013



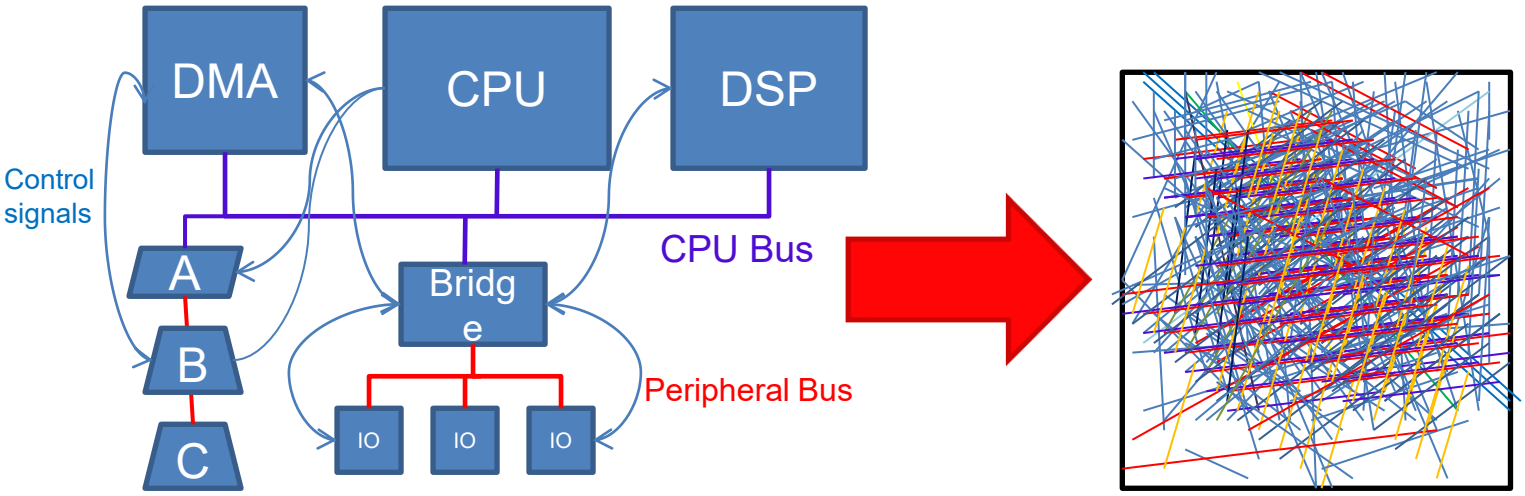
Summary of all On-chip Interconnection Types



Ref.: Abderazek Ben Abdallah, Multicore Systems-on-Chip: Practical Hardware/Software Design, 2nd Edition, *Publisher: Springer, (2013) , ISBN-13: 978-9491216916*

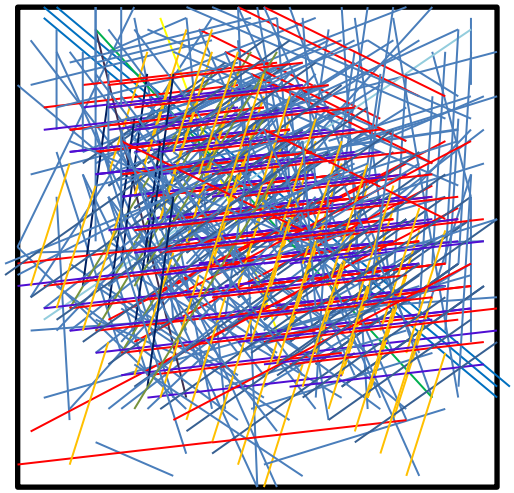
Traditional SoC Nightmare

- Variety of dedicated interfaces
- Design and verification complexity
- Unpredictable performance
- Many underutilized wires

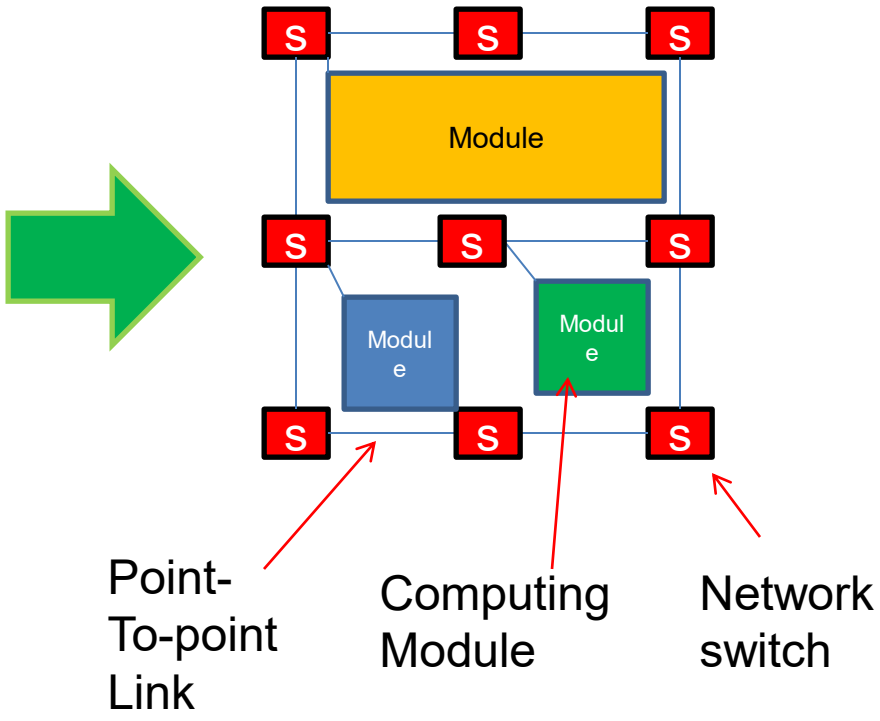


NoC: A paradigm Shift in VLSI

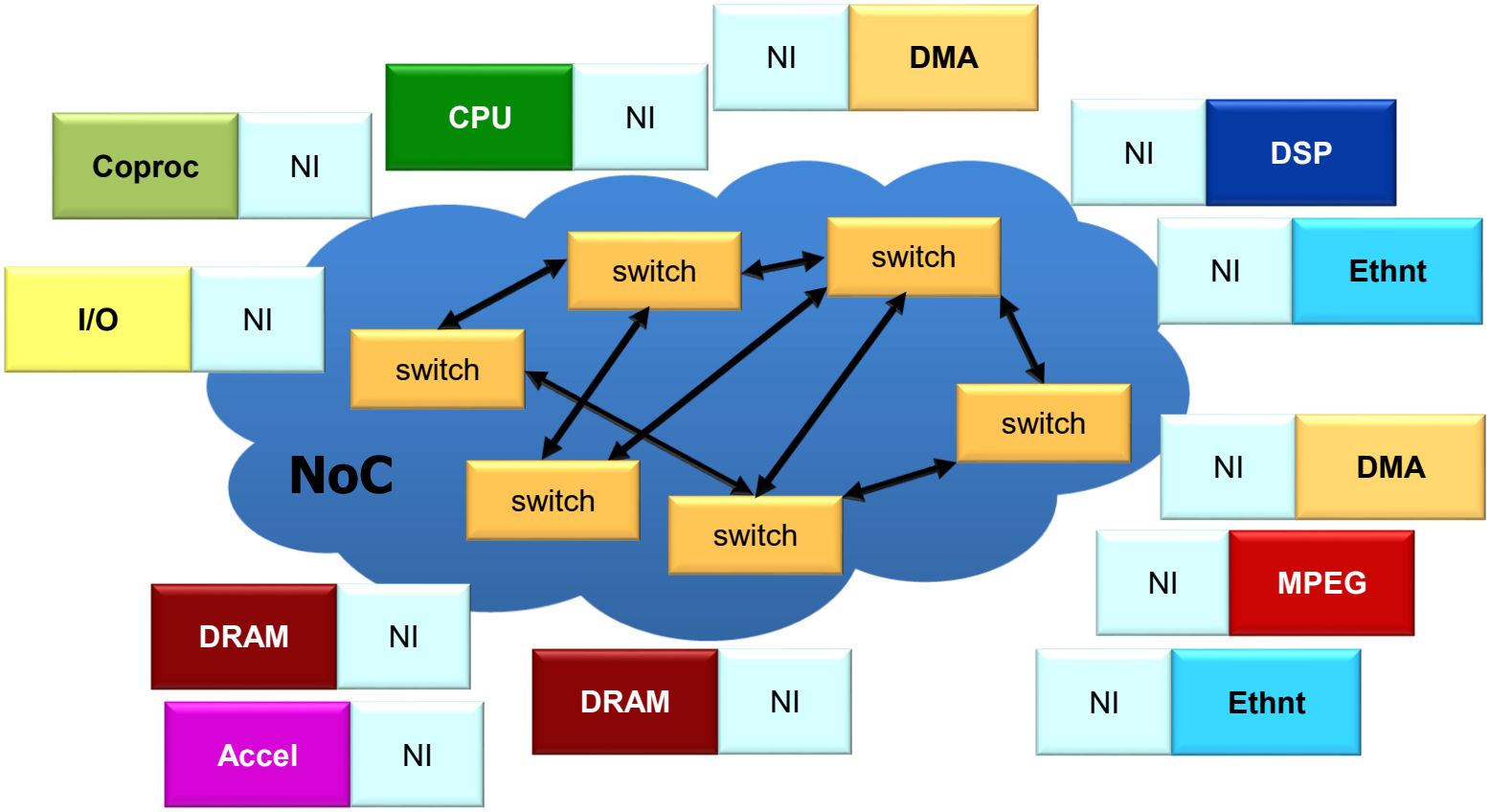
From: Dedicated signal wires



To: Shared network

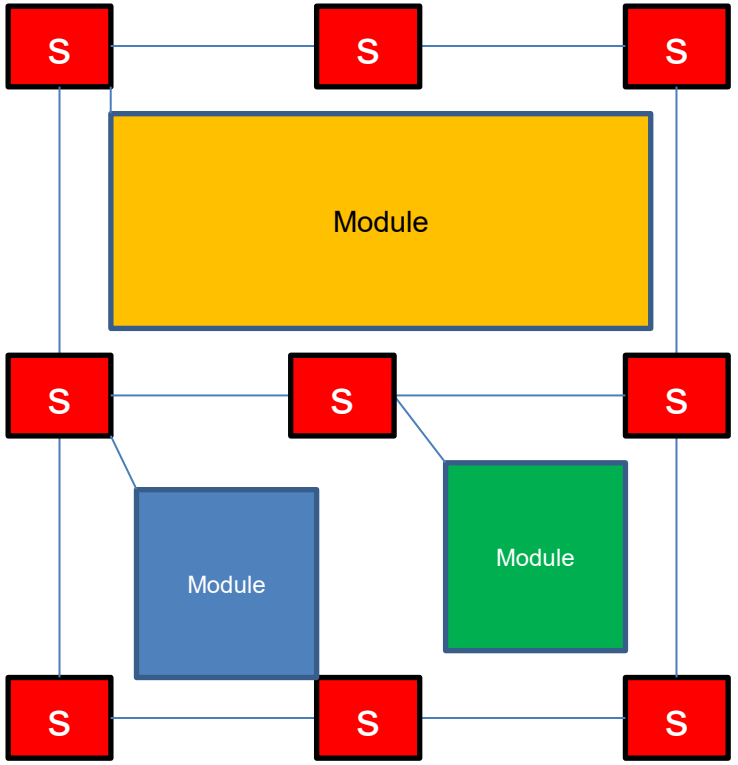


NoC: A paradigm Shift in VLSI

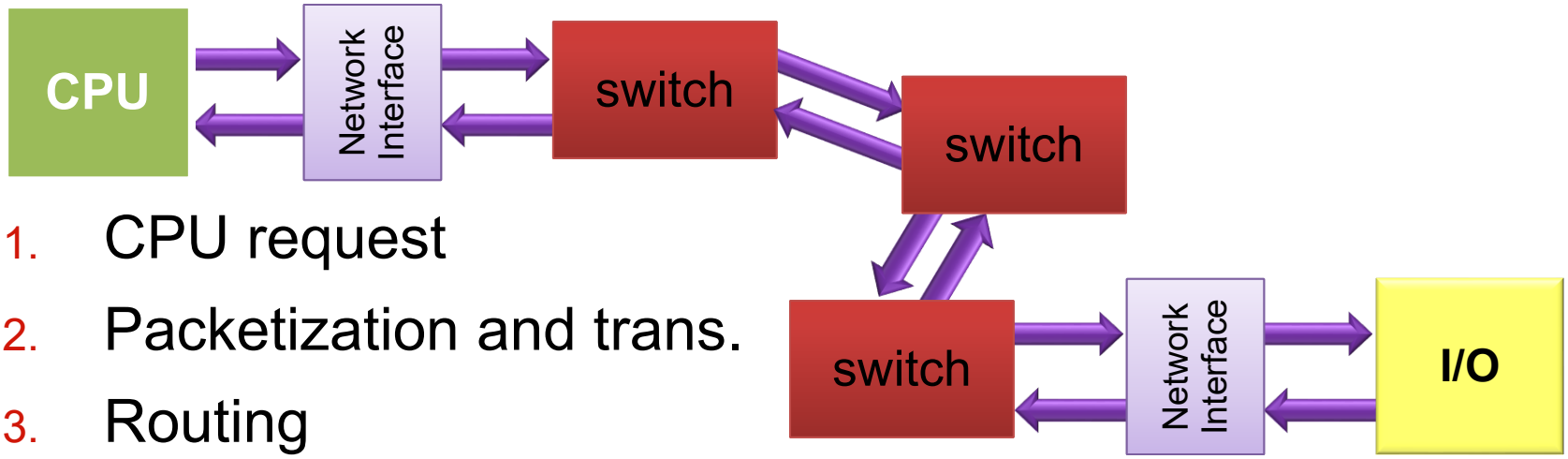


NoC Essential

- Communication by packets of bits
- Routing of packets through several hops via switches
- Efficient sharing of wires
- Parallelism



NoC Operation Example

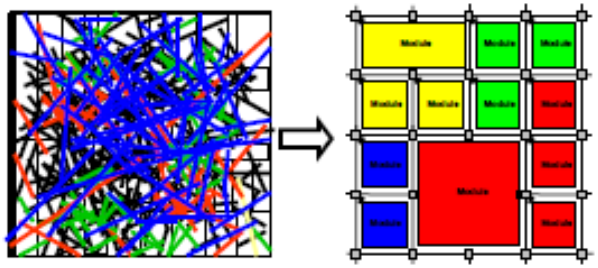


1. CPU request
2. Packetization and trans.
3. Routing
4. Receipt and unpacketization (AHB, OCP, ... pinout)
5. Device response
6. Packetization and transmission
7. Routing
8. Receipt and unpacketization

Critical problems addressed by NoC

1) Global interconnect design problem:
delay, power, noise, scalability, reliability

2) System integration
productivity problem



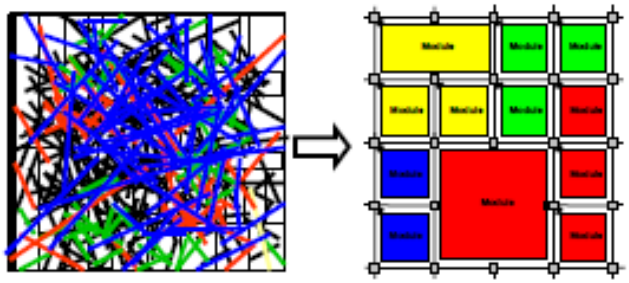
3) Multicore Processors
key to power-efficient computing

Why now is the time for NoC ?

Difficulty of DSM wire design

Productivity pressure

Multicore



Layers of Abstraction in Network Modeling

- **Software layers**
  - Application, OS
- **Network & transport layers**
  - Network topology e.g., crossbar, ring, mesh, torus, fat tree,…
  - Switching Circuit / packet switching(SAF, VCT), wormhole
  - Addressing Logical/physical, source/destination, flow, transaction
  - Routing Static/dynamic, distributed/source, deadlock avoidance
  - Quality of Service e.g., guaranteed-throughput, best-effort
  - Congestion control, end-to-end flow control
- **Data link layer**
  - Flow control
  - Handling of contention
  - Correction of transmission errors
- **Physical layer**
  - Wires, drivers, receivers, repeaters, signaling, circuits,..

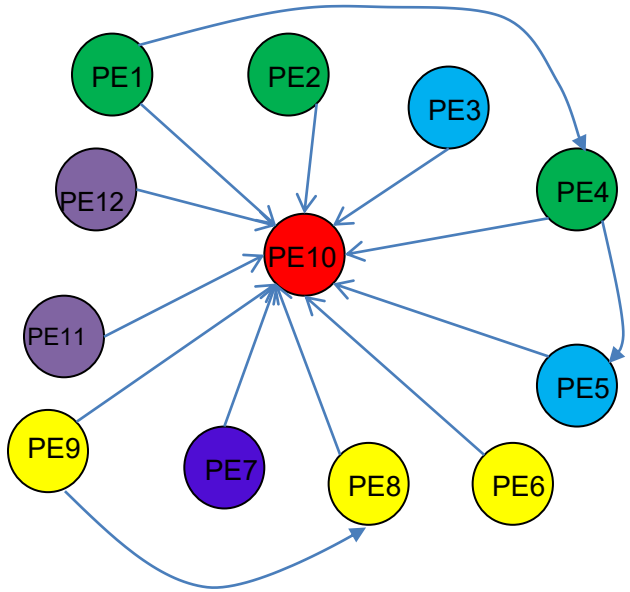
VLSI CAD Problems in NoC

- Application mapping (map tasks to cores)
- Floorplanning (within the network)
- Routing (of messages)
- Buffer sizing (size of FIFO queues in the routers)
- Simulation (Network simulation, traffic/delay/power modeling)
- Other NoC design problems (topology synthesis, switching, virtual channels, arbitration, flow control,……)

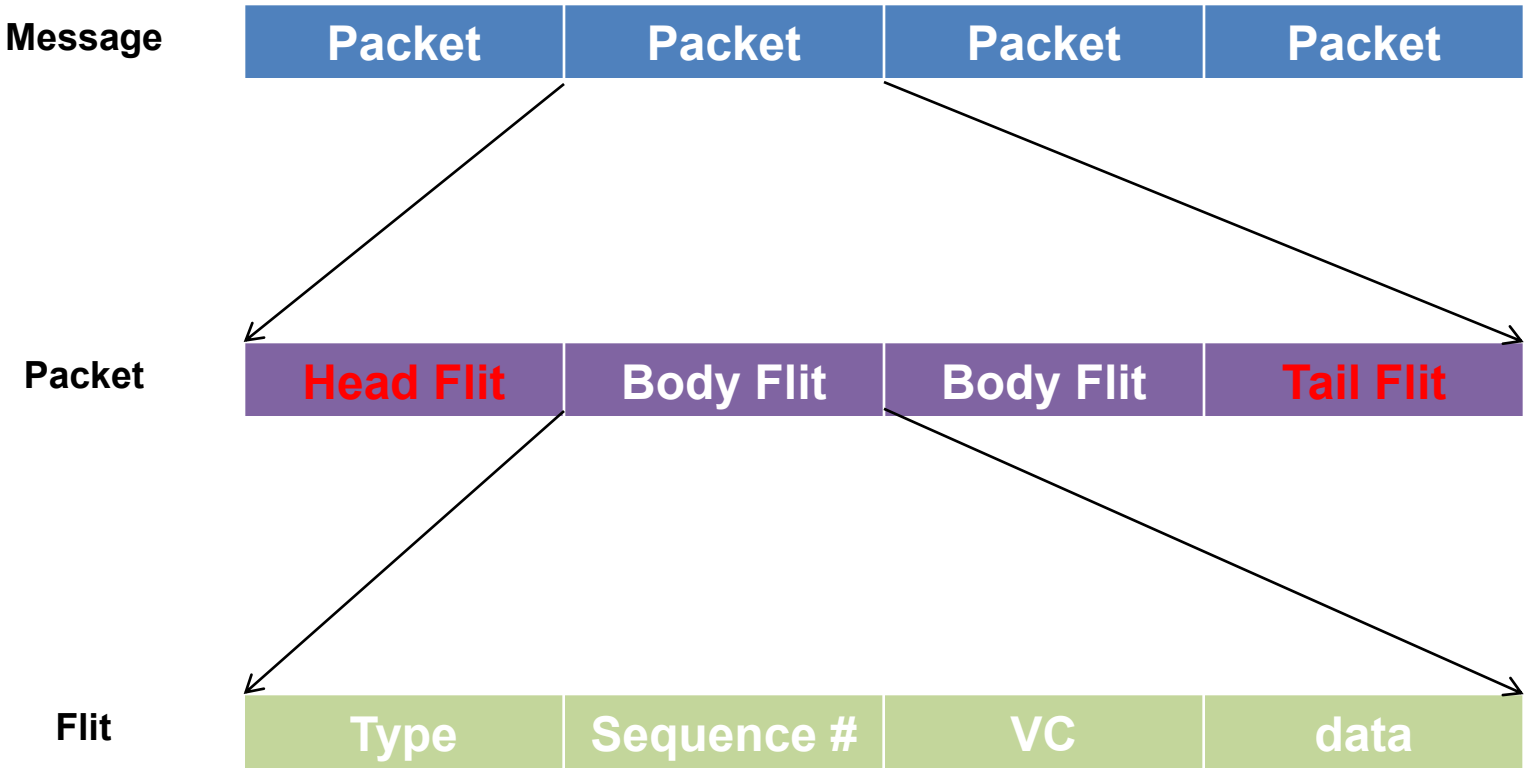
Traffic Abstractions

- Traffic model are generally captured from actual traces of functional simulation
- A statically distribution is often assumed for message

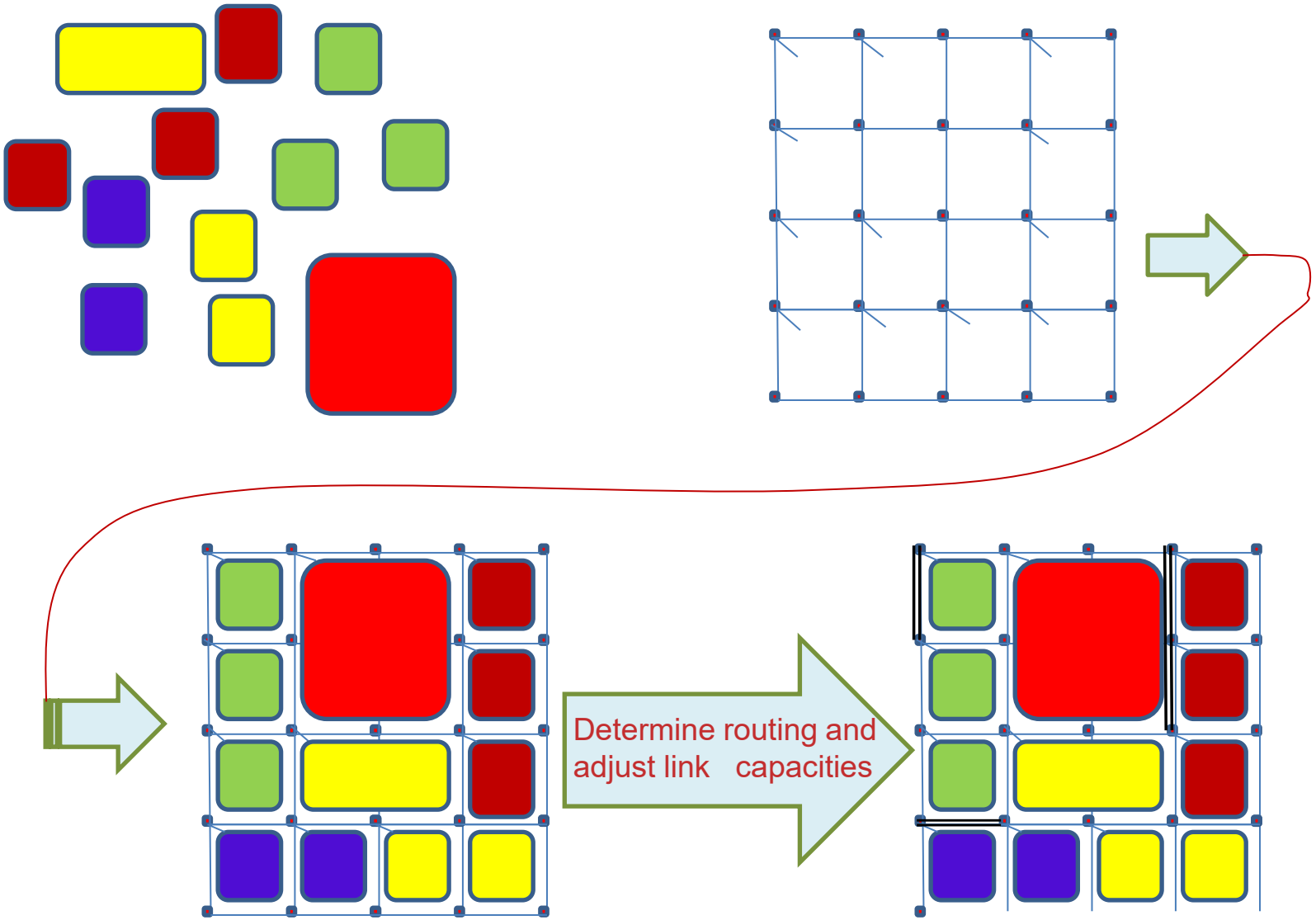
Flow	Bandwidth	Packet size	Latency
1 ->10	400kb/s	1kb	5ns
2->10	1.8Mb/s	3kb	12ns
1->4	230kb/s	2kb	6ns
4->10	50kb/s	1kb	3ns
4->5	300kb/s	3kb	4ns
3->10	34kb/s	0.5kb	15ns
5->10	400kb/s	1kb	4ns
6->10	699kb/s	2kb	1ns
8->10	300kb/s	3kb	12ns
9->8	1.8mb/s	5kb	7ns
9->10	200kb/s	5kb	10ns
7->10	200kb/s	3kb	12ns
11->10	300kb/s	4kb	10ns
12->10	500kb/s	5kb	12ns



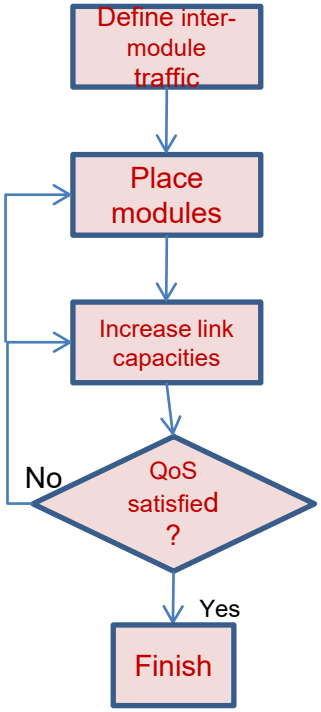
Data Abstractions

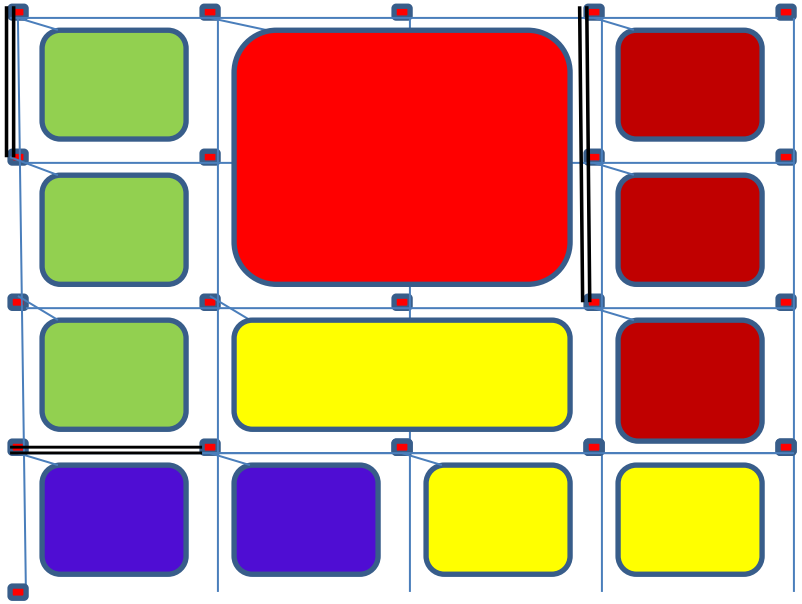


Typical NoC Design Flow



Timing Closure in NoC





- Too long capacity results in poor QoS
- Too high capacity wastes area
- Uniform link capacities are a waste in ASIP system

NoC - Part 2

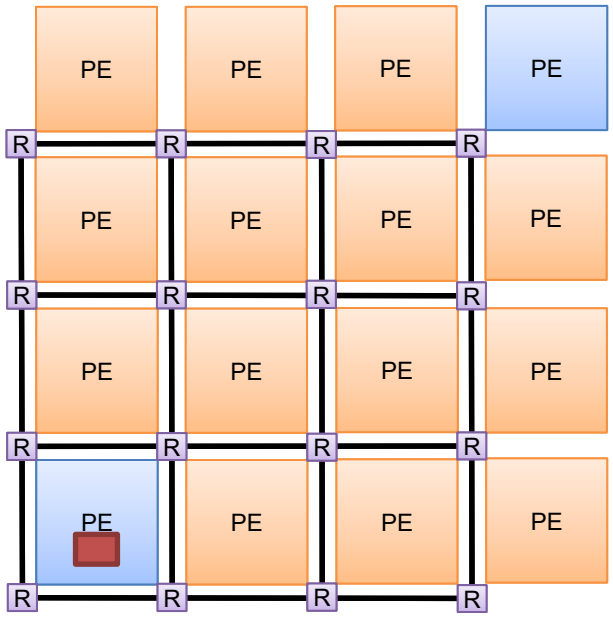
- Topology
- Routing
- Control flow
- Network interface
- Router architecture

NoC Topology

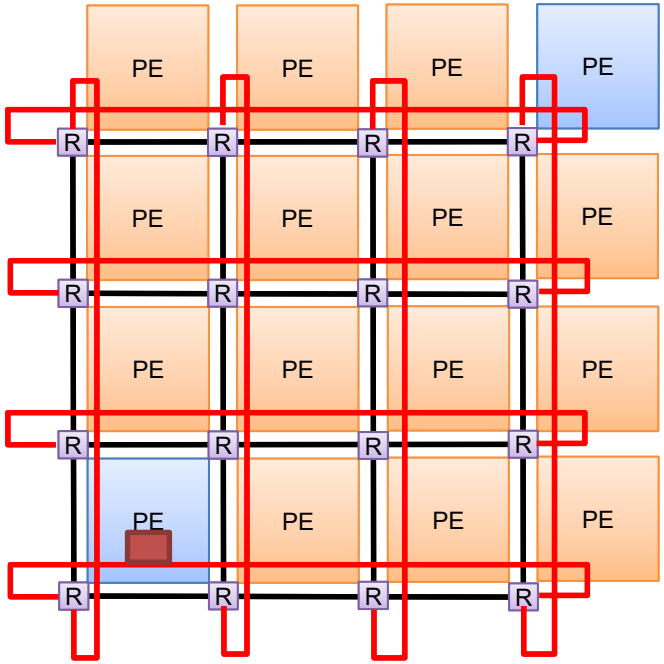
NoC topology is the connection map between PEs.

- Mainly adopted from large-scale networks and parallel computing
- A good topology allows to fulfill the requirements of the traffic at reasonable costs
- Topology classifications:
  1. Direct topologies
  2. Indirect topologies

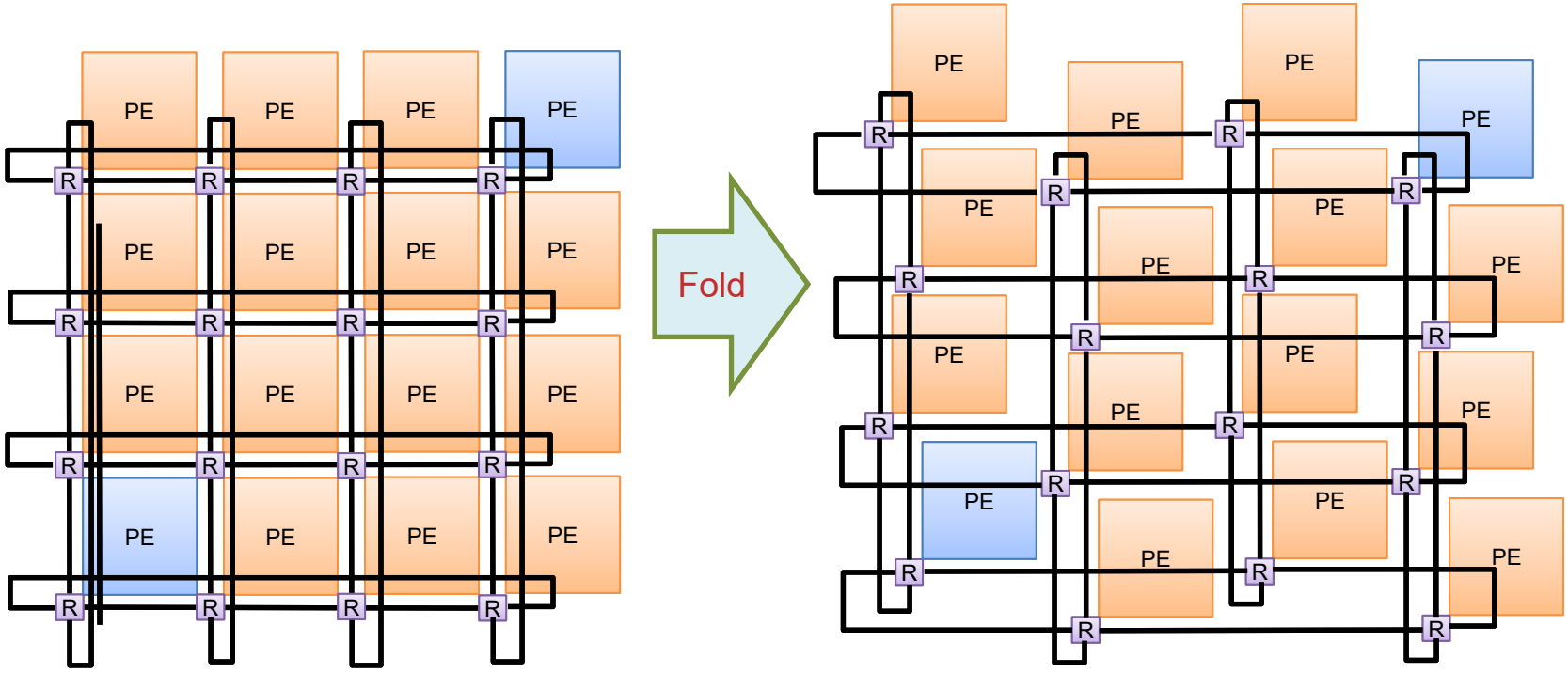
Direct Topology: Mesh



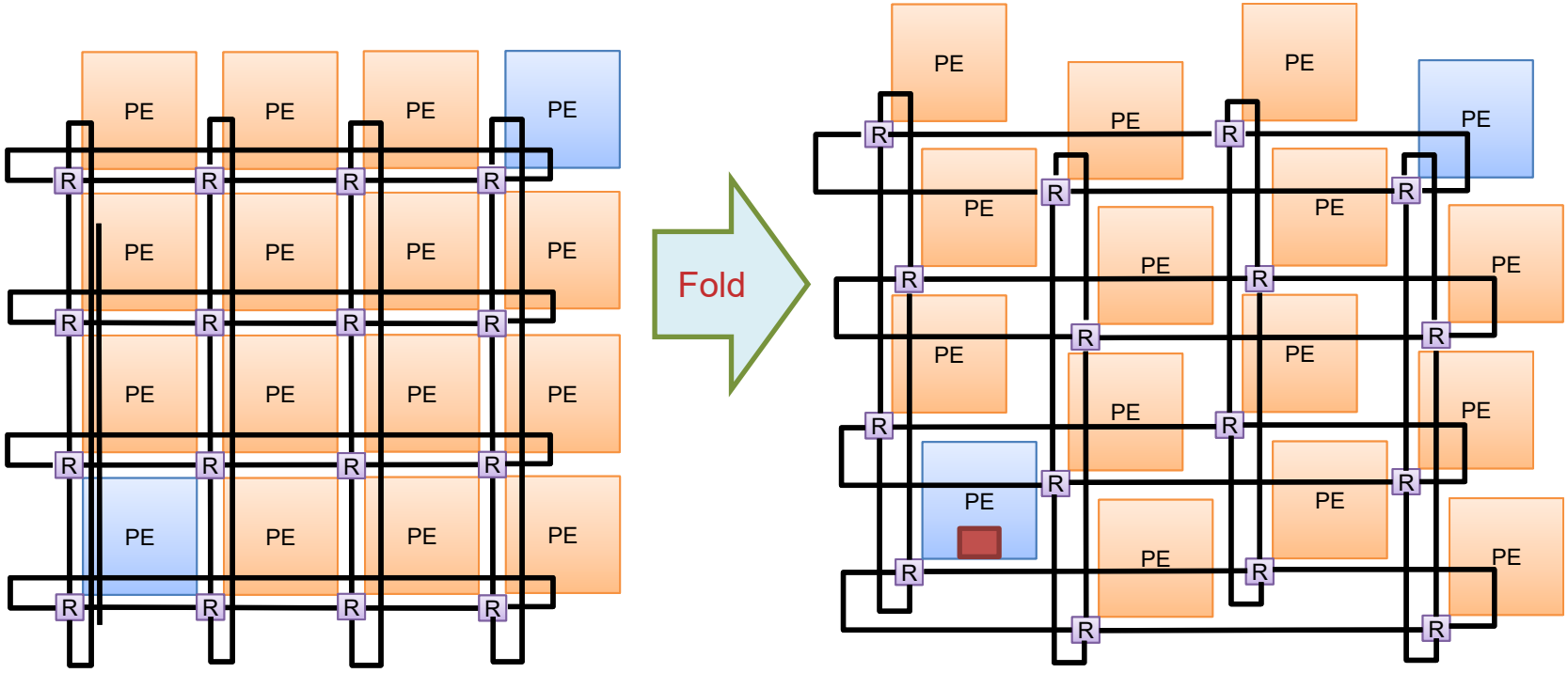
Direct Topology: Torus



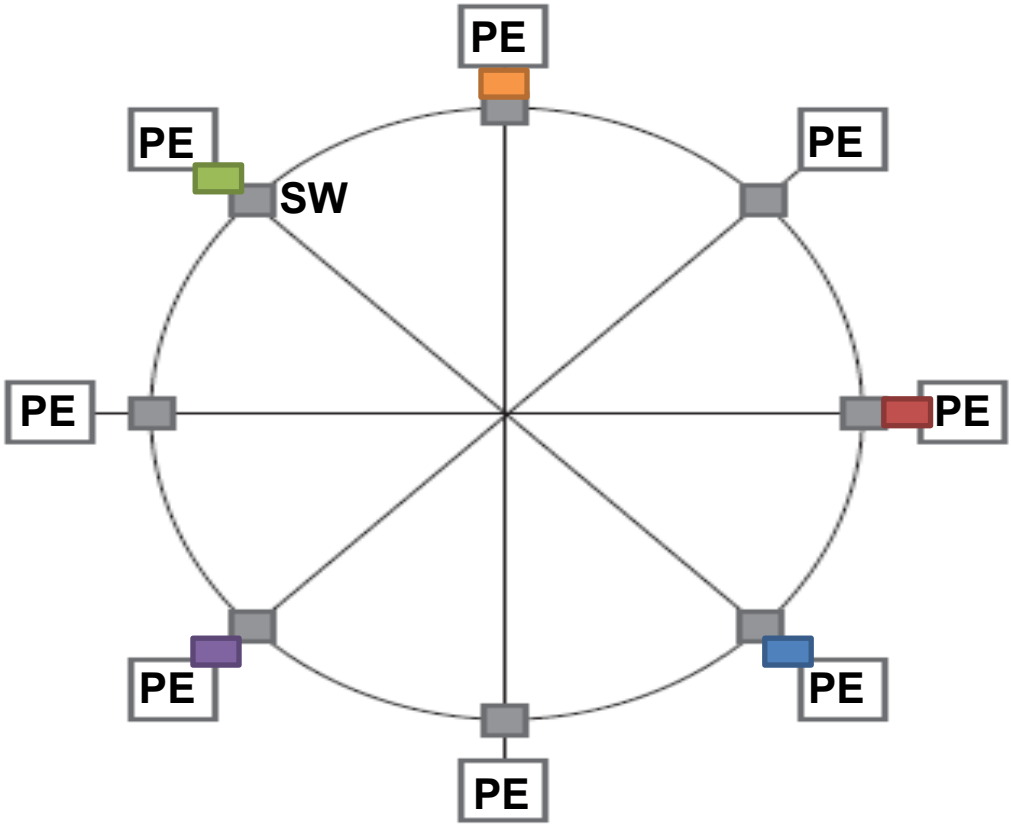
Direct Topology: Folded Torus



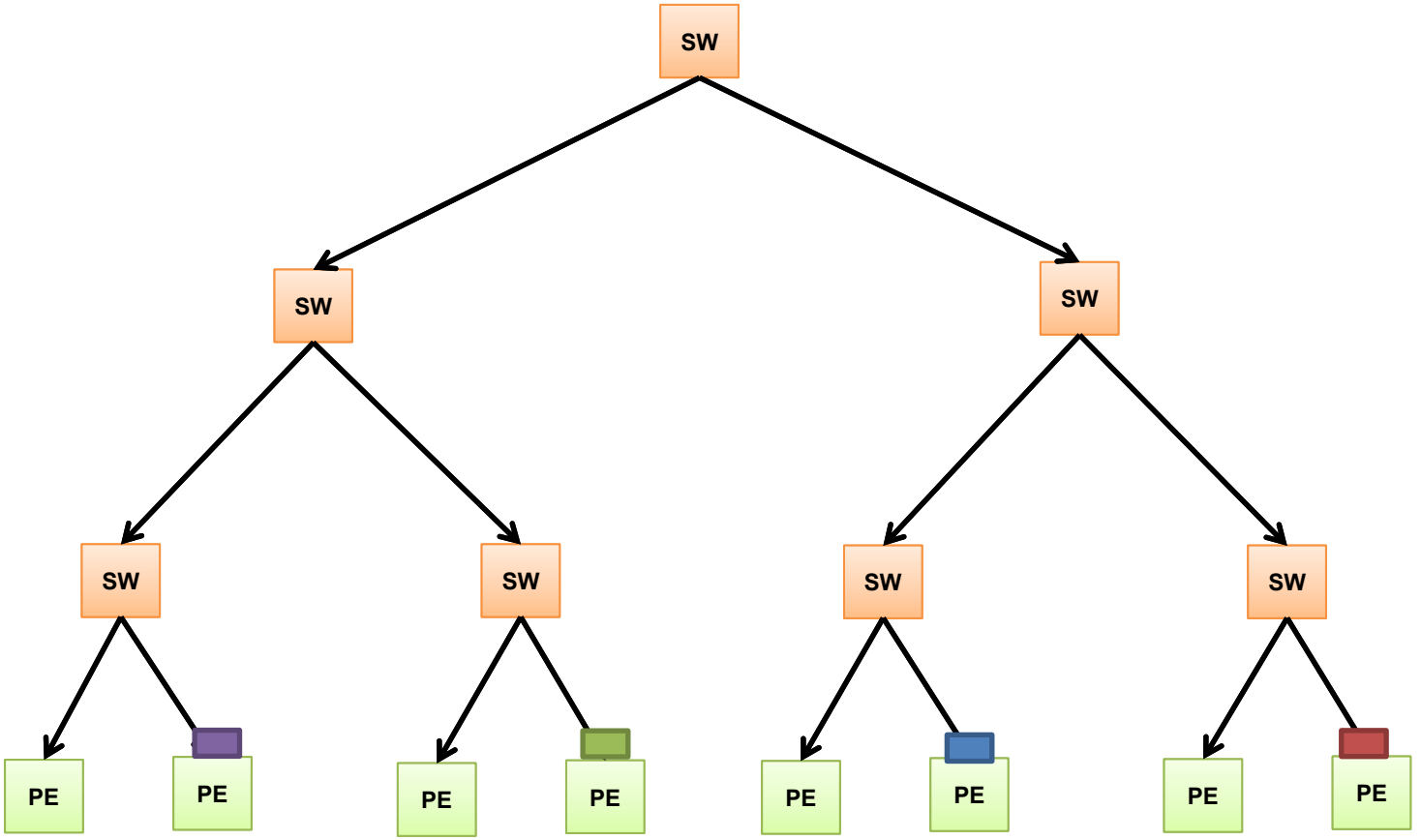
Direct Topology: Folded Torus



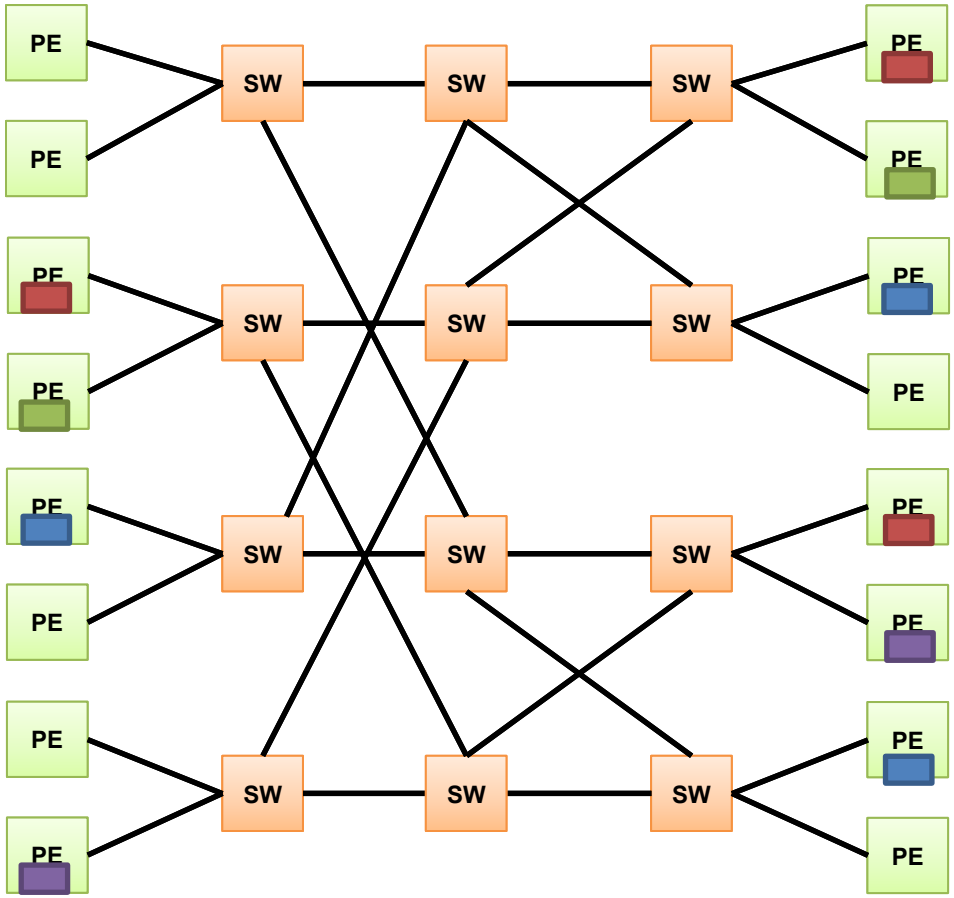
Direct Topology: Octagon



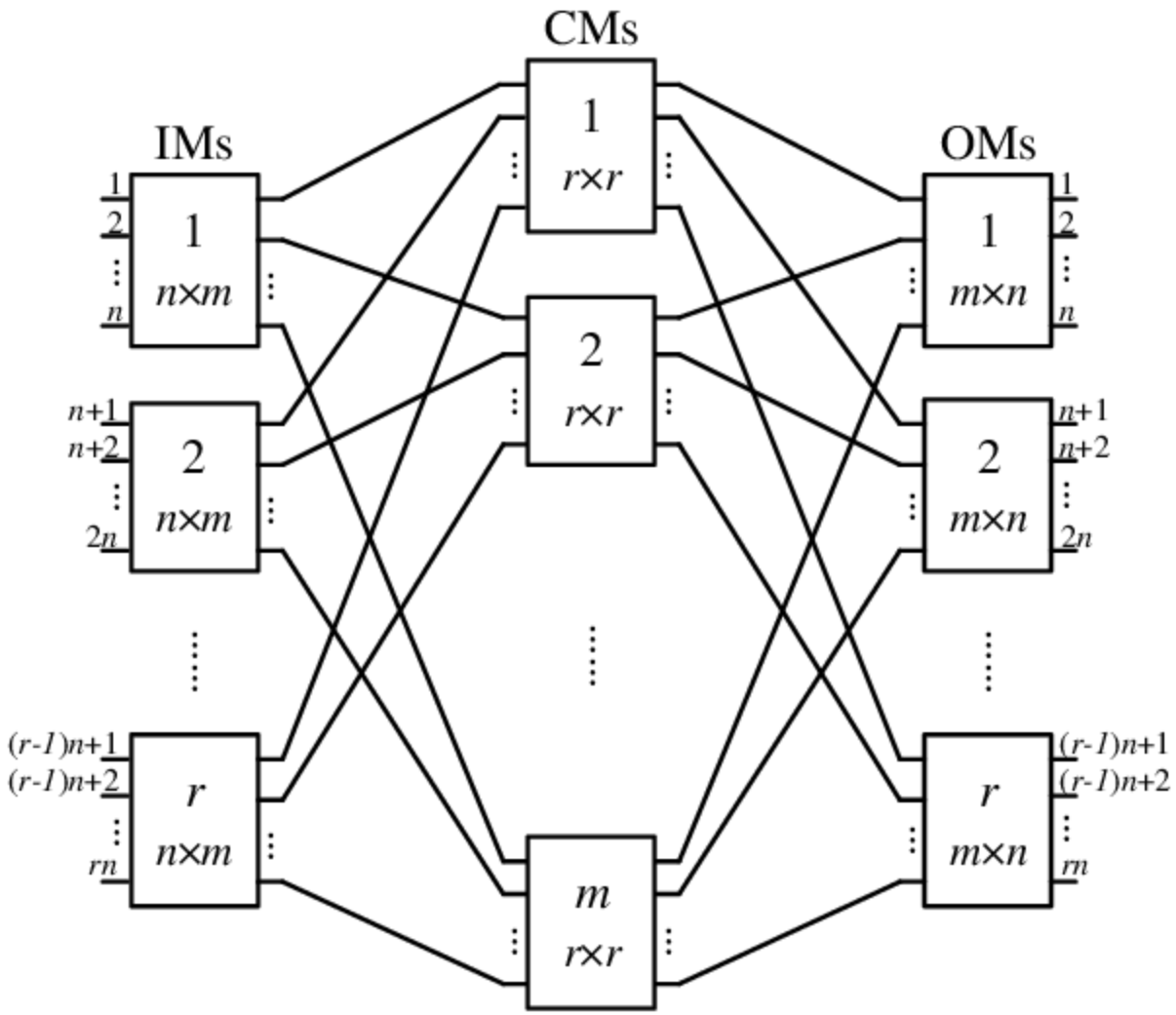
Indirect Topology: Fat Tree



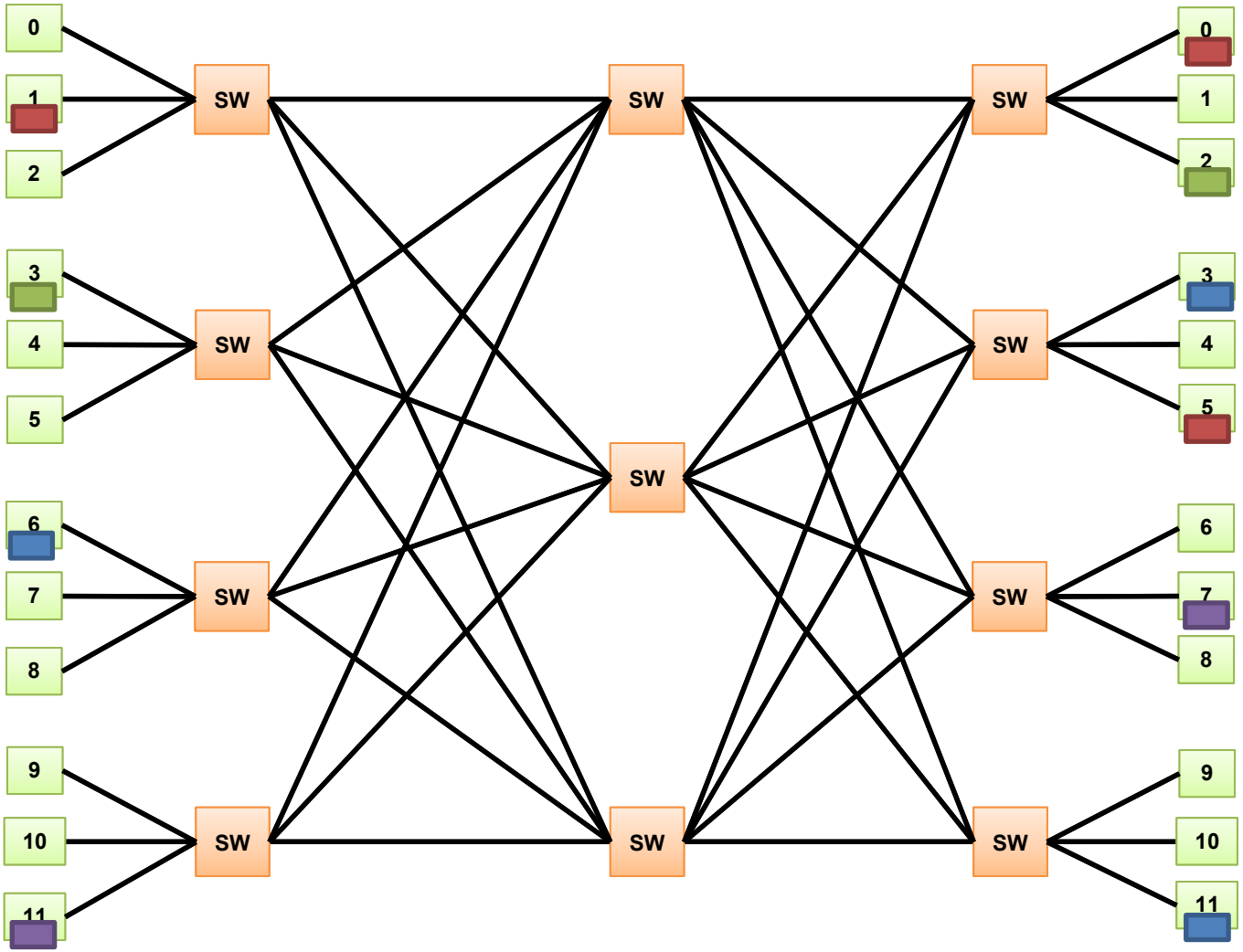
Indirect Topology:
k-ary n-fly butterfly network



Indirect Topology: (m, n, r) symmetric Clos network

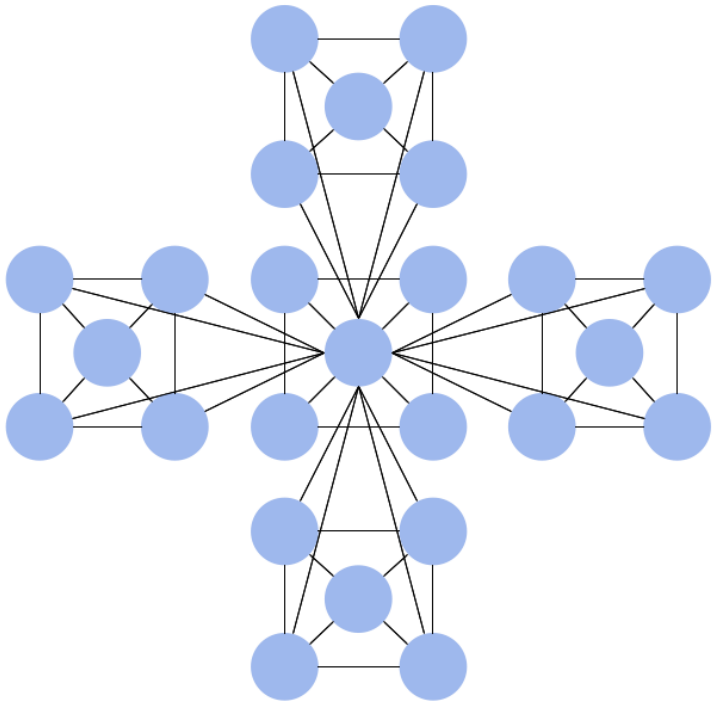


Indirect Topology:
(m, n, r) symmetric Clos network



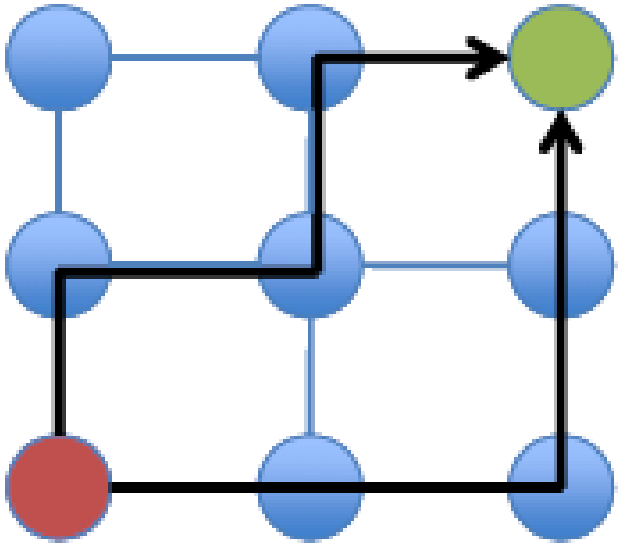
How to Select a Topology ?

- **Application decides the topology type**
  - if *PEs = few* tens → Mesh is recommended
  - if *PEs = 100 or more* → Hierarchical star is recommended
- Some topologies are better for certain designs than others



NoC Routing

Routing algorithm determine path(s) from source to destination.
Routing must prevent deadlock, livelock , and starvation.



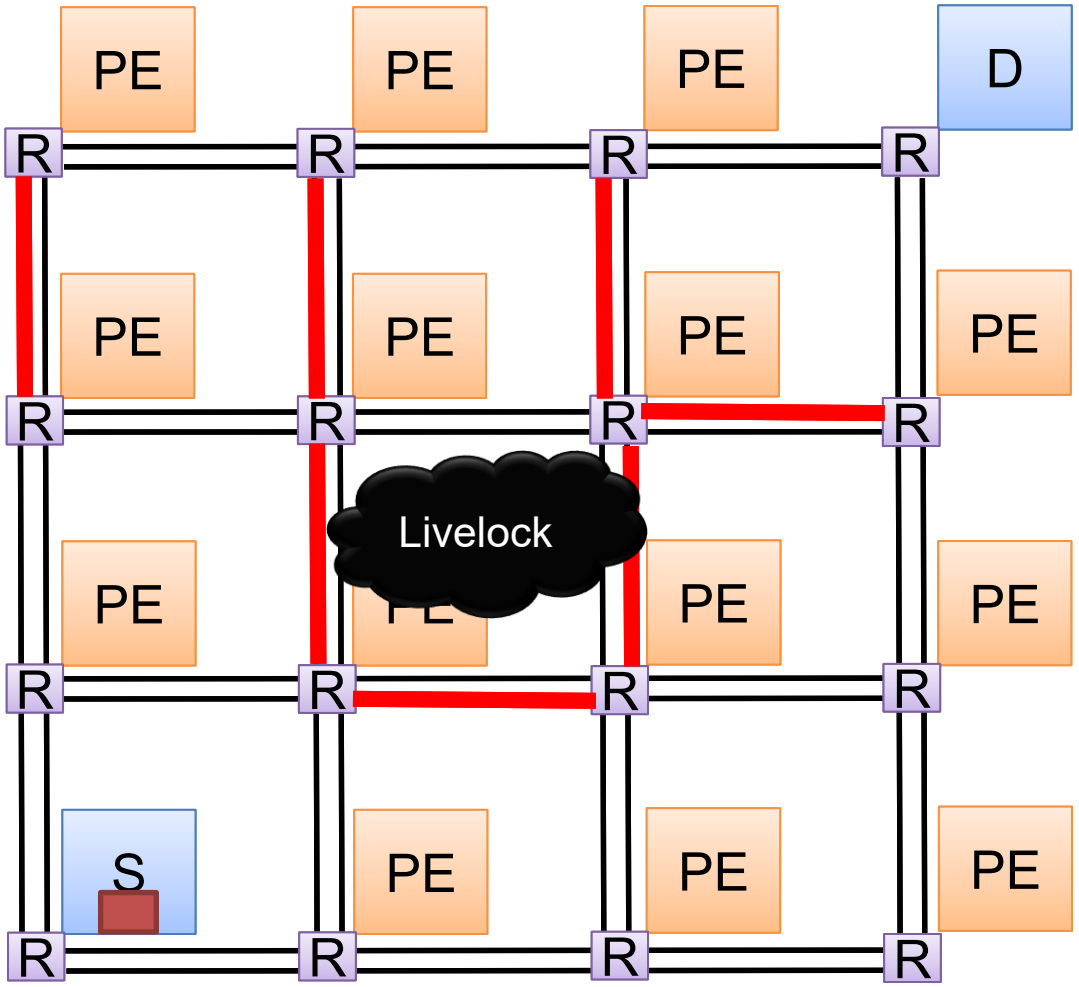
Deadlock, Livelock, and Starvation

Deadlock: A packet does not reach its destination, because it is blocked at some intermediate resource.

Livelock: A packet does not reach its destination, because it enters a cyclic path.

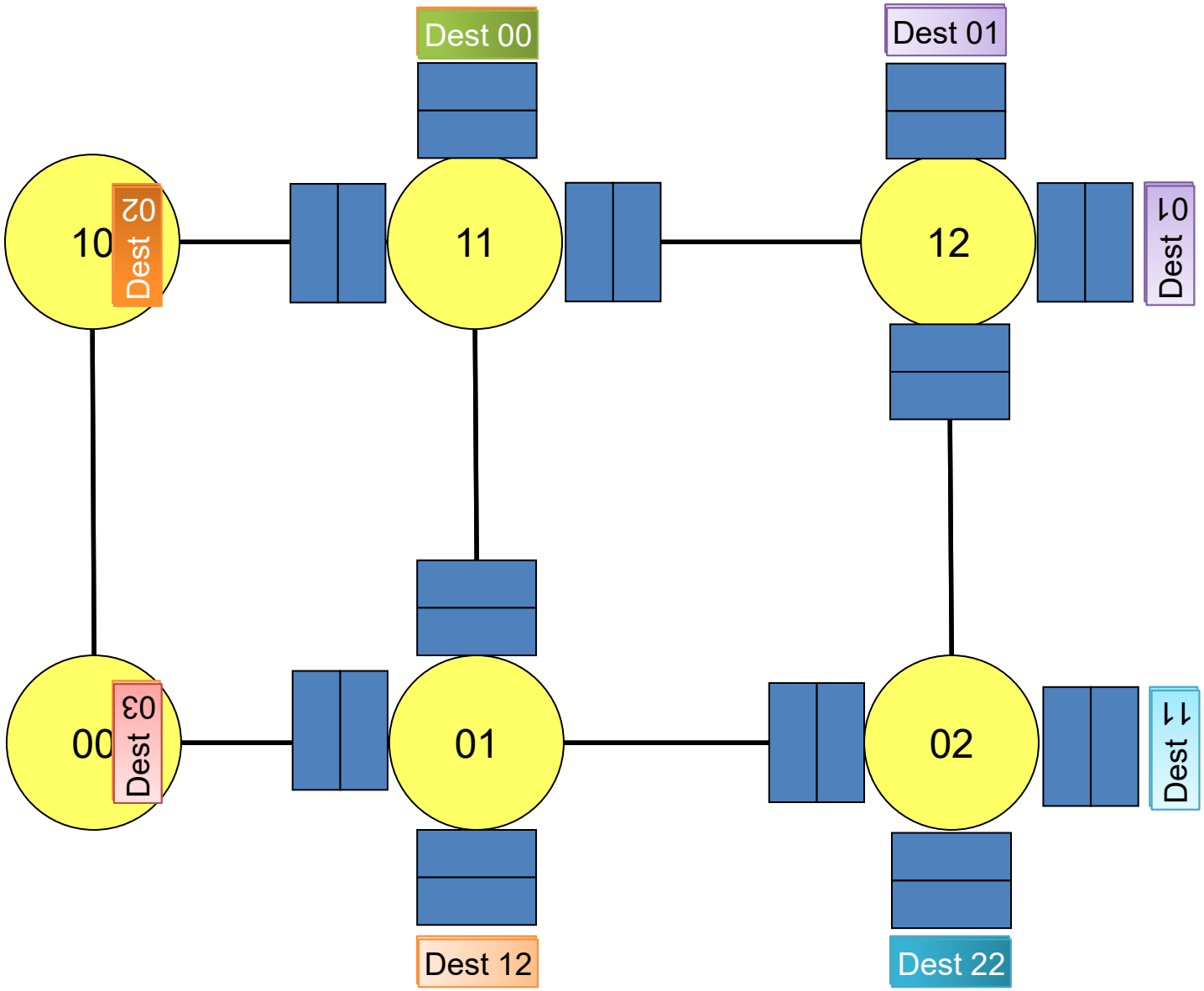
Starvation: A packet does not reach its destination, because some resource does not grant access (wile it grants access to other packets).

Livelock

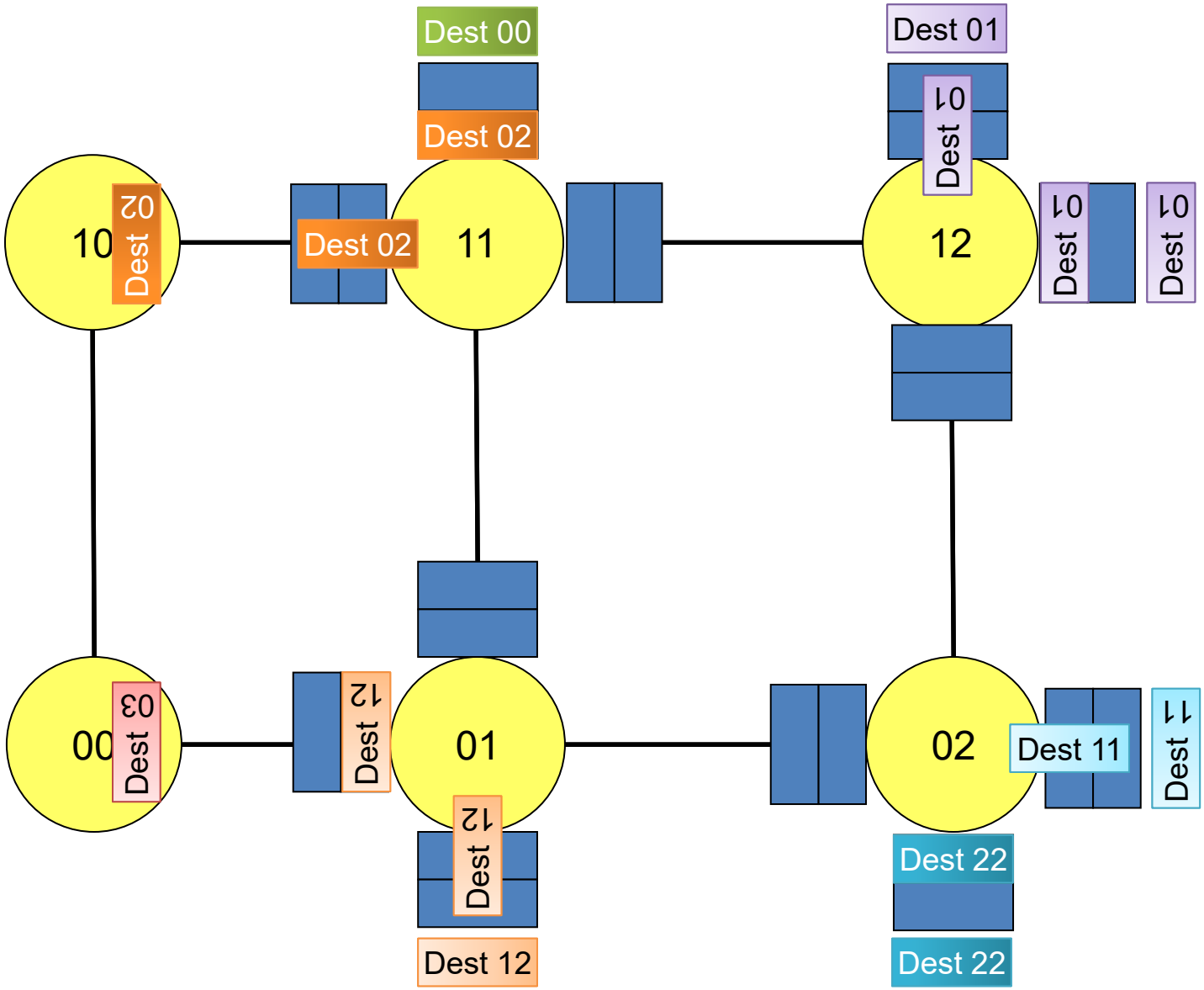


Congested channel

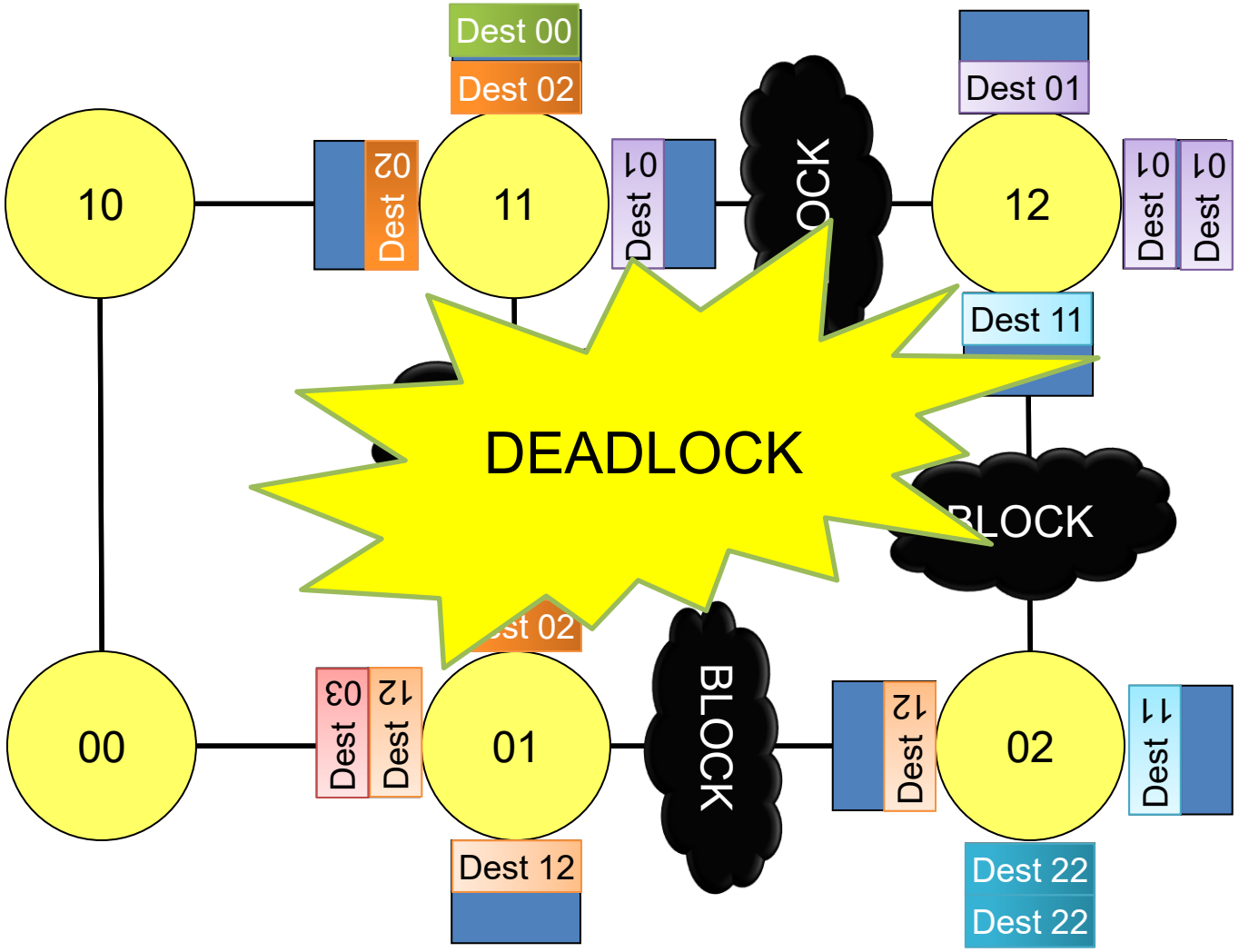
Deadlock


Dest 00
Dest 01
10
11
12
Dest 02
Dest 10
00
01
02
Dest 03
Dest 11
Dest 12
Dest 22

Deadlock

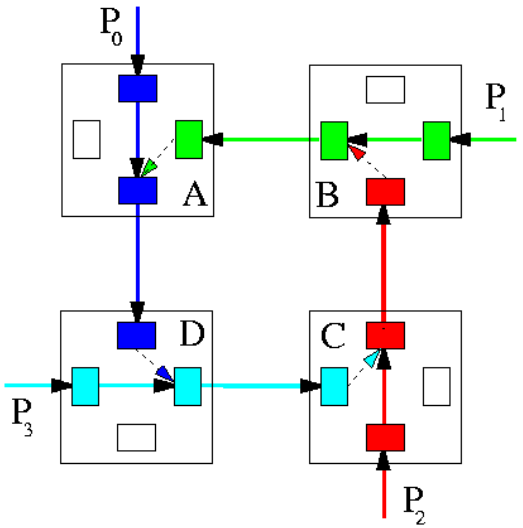

Dest 00
Dest 02
Dest 01
Dest 01
10
Dest 02
Dest 02
11
12
Dest 01
Dest 01
00
Dest 03
Dest 12
01
Dest 12
02
Dest 11
Dest 11
Dest 22
Dest 12
Dest 22

Deadlock


Dest 00
Dest 02
Dest 01
10
11
12
Dest 11
DEADLOCK
00
01
02
Dest 12
Dest 22
Dest 22

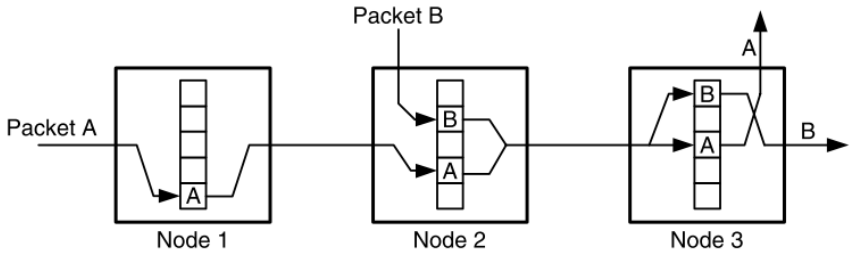
Deadlock Avoidance

- Deadlock can be avoided by eliminating cycles in the resource dependence graph (i.e., cyclic dependence).

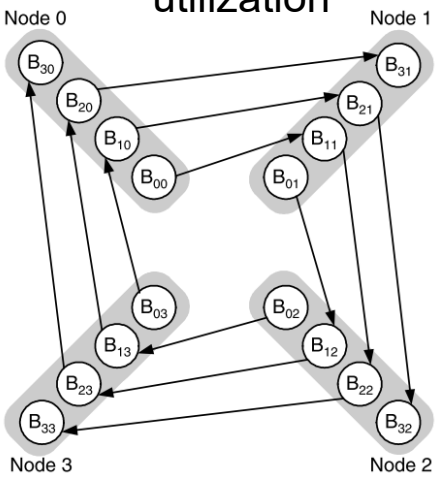


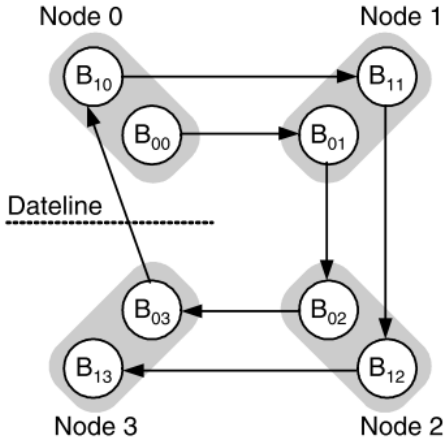
- Categories of deadlock avoidance methods
  - **Resource classes:** Group the resources into numbered classes and restrict allocation of resources
  - **Dimension Order Routing:** In *k*-ary *n*-meshes, this is the simplest way to guarantee deadlock freedom
  - **The Turn Model:**
    - A more general framework for restricting routing algorithms in mesh networks
    - Possible deadlock cycles are defined in terms of the particular turns needed to create them.

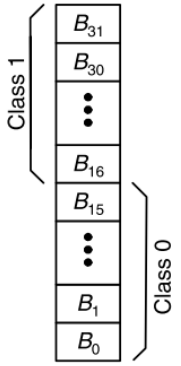
Deadlock Avoidance: Resource Classes

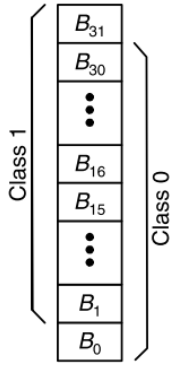


- Packets acquire resources from classes in ascending order
  - **Distance classes:** Packets acquire resources from classes in ascending order. At each hop, the packet acquires a resource of the next highest class.
    - Result in imbalanced traffic
  - **Dateline:** It is used to reduce the number of buffer classes. Switch to the higher class only crossing the dateline.
    - Overlapping resource classes can further improve the efficiency of resource utilization



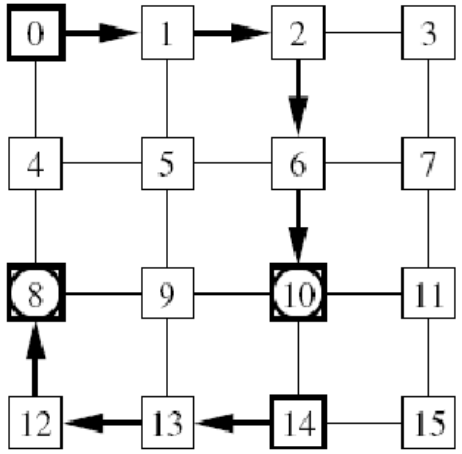






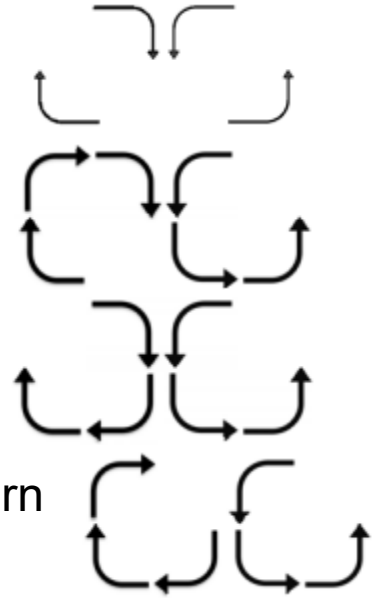
Deadlock Avoidance: Dimension Order Routing and Turn Model

- Dimension-order routing routes packets by crossing dimensions in increasing order, nullifying the offset in one dimension before routing in the next one.

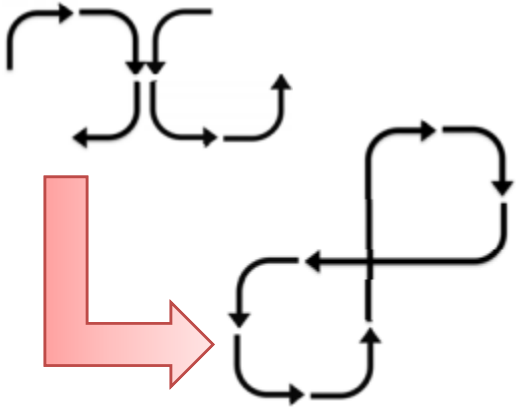


- Turn Model
  - All possible turns in a 2-D Mesh

  - X-Y turn
  - West-First turn
  - North-Last turn
  - Negative-First turn



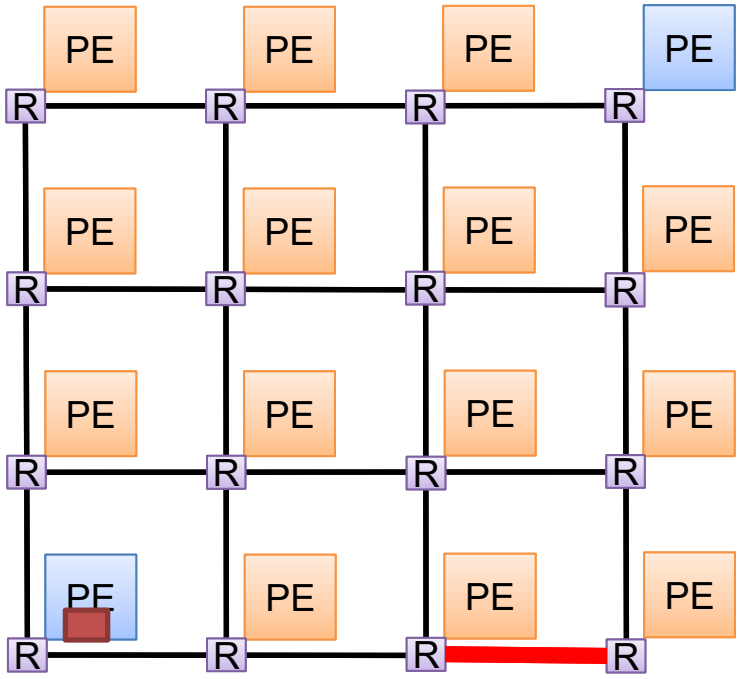
How about this?



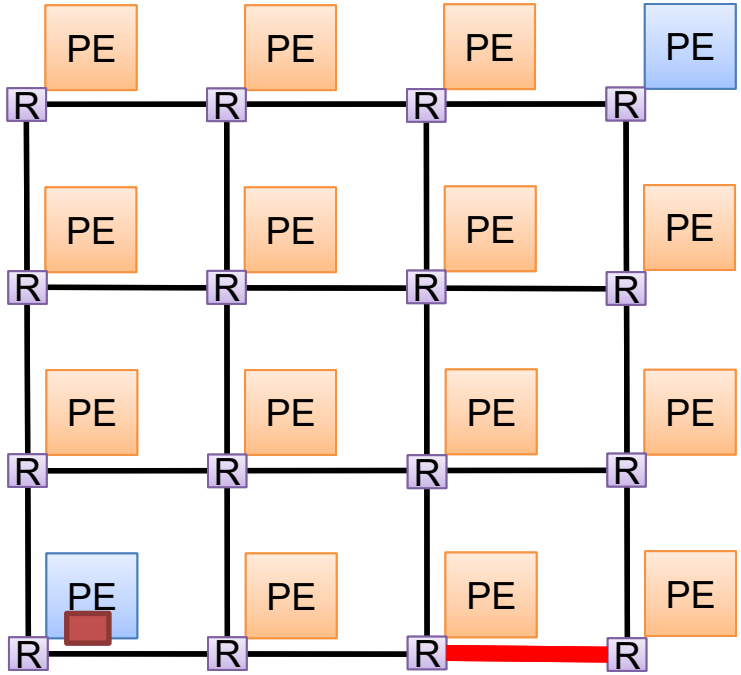
Routing Algorithm Attributes

- Number of destinations
  - Unicast, Multicast, Broadcast?
- Adaptively
  - Deterministic, Oblivious or Adaptive
- Implementation (Mechanisms)
  - Source or node routing?
  - Table or circuit?

Static vs. Adaptive Routing



Static

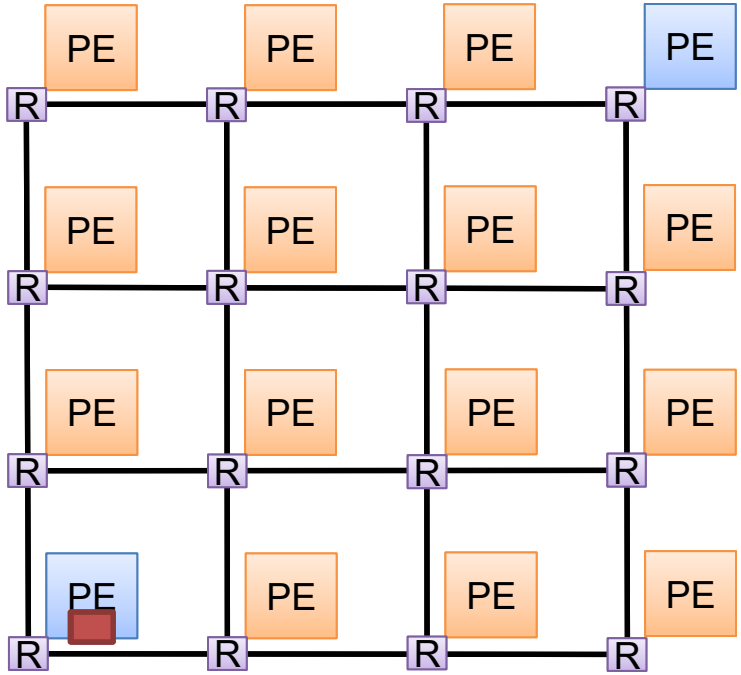


Adaptive

Congested channel

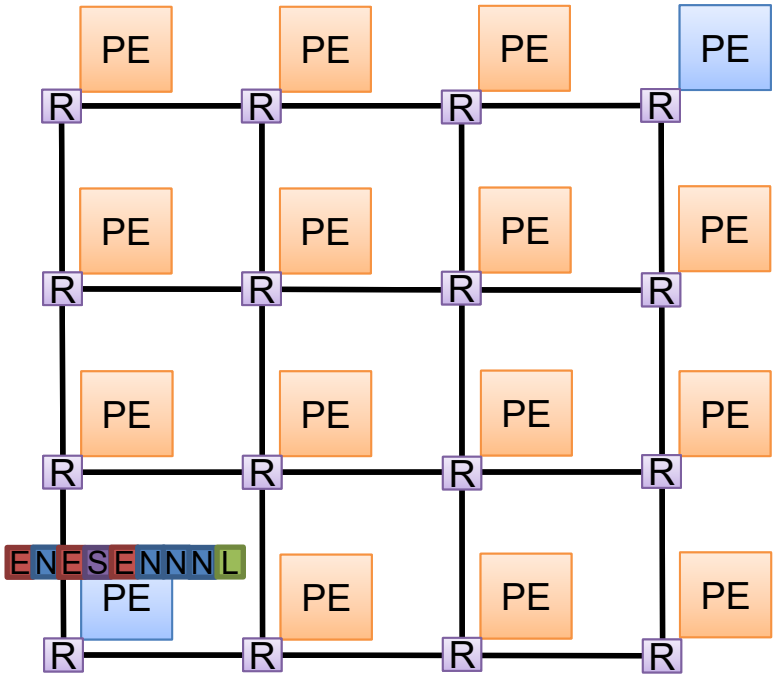
Minimal vs. Non-Minimal

Minimal

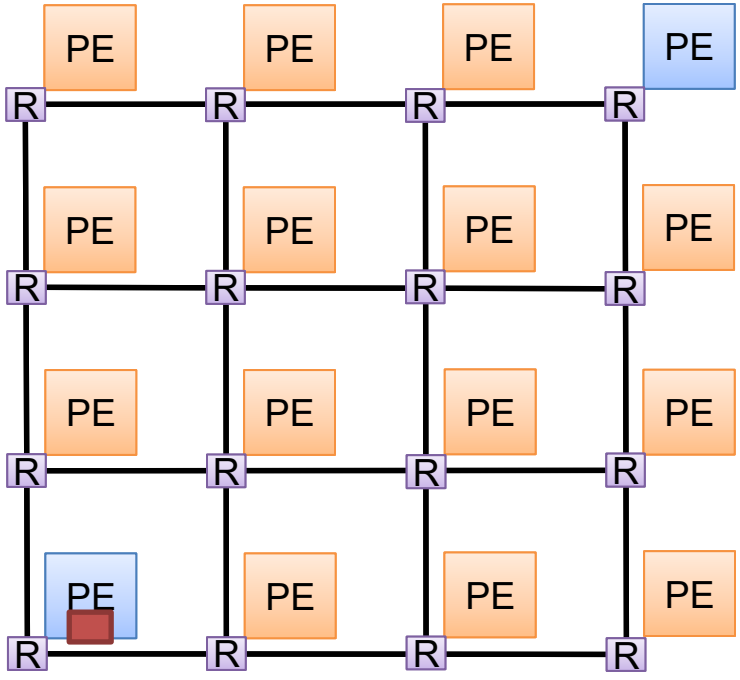


Non-Minimal

Source vs. Distributed

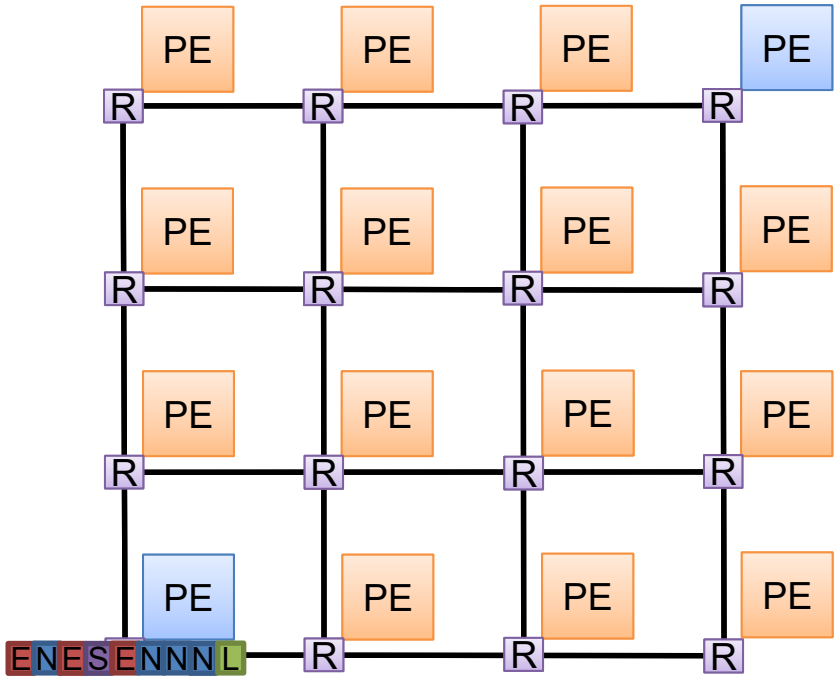


Source

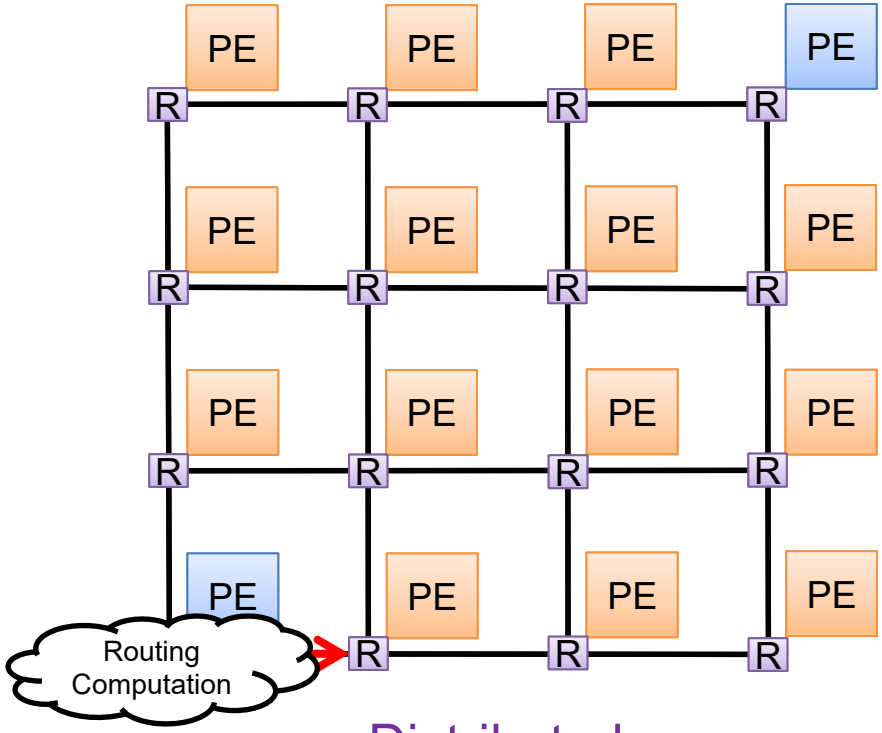


Distributed

Source vs. Distributed

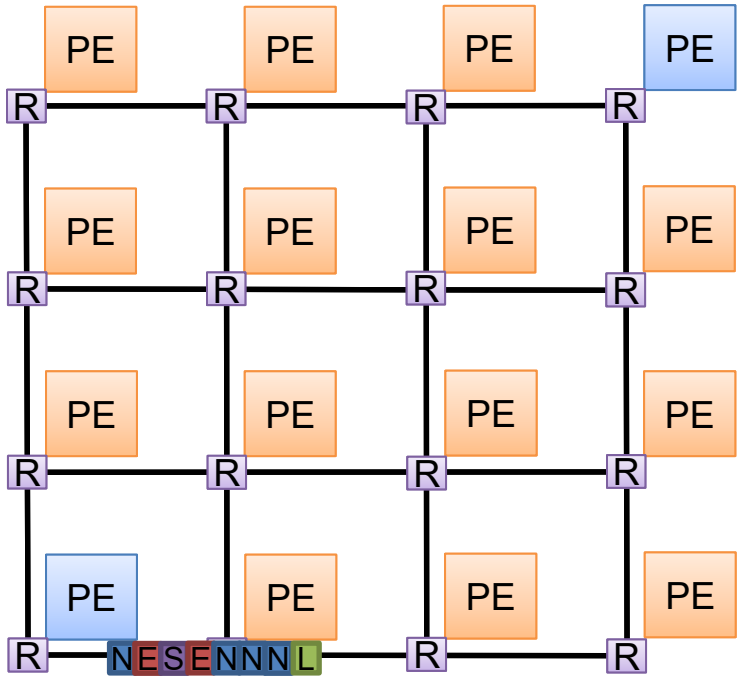


Source

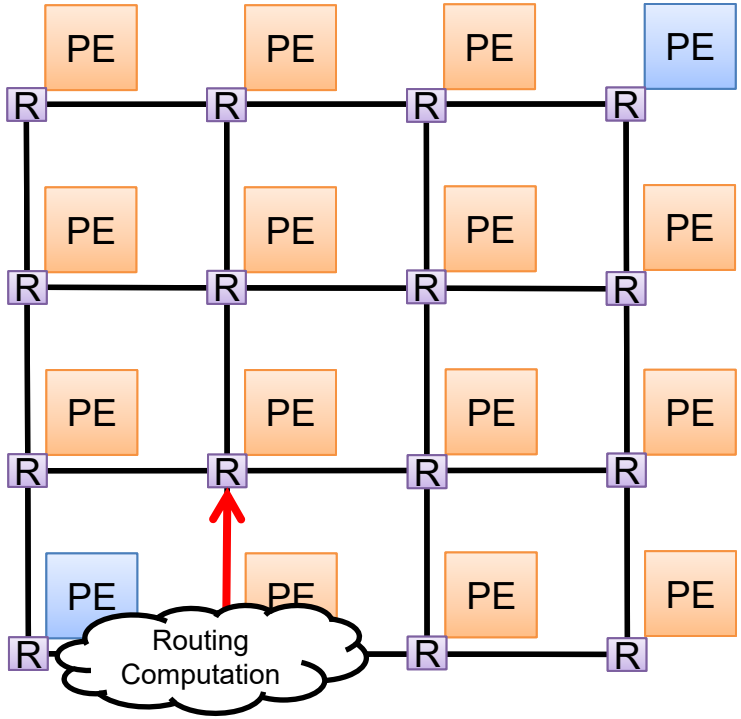


Distributed

Source vs. Distributed

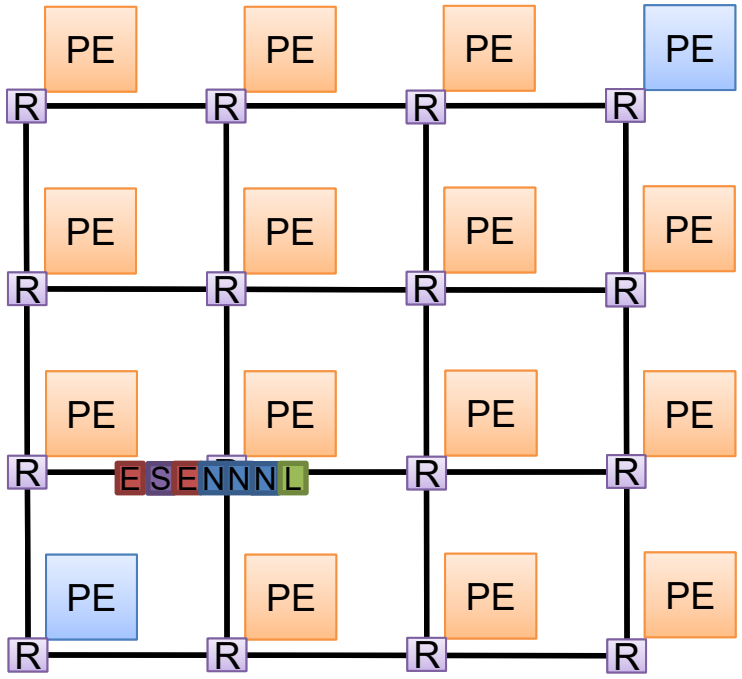


Source

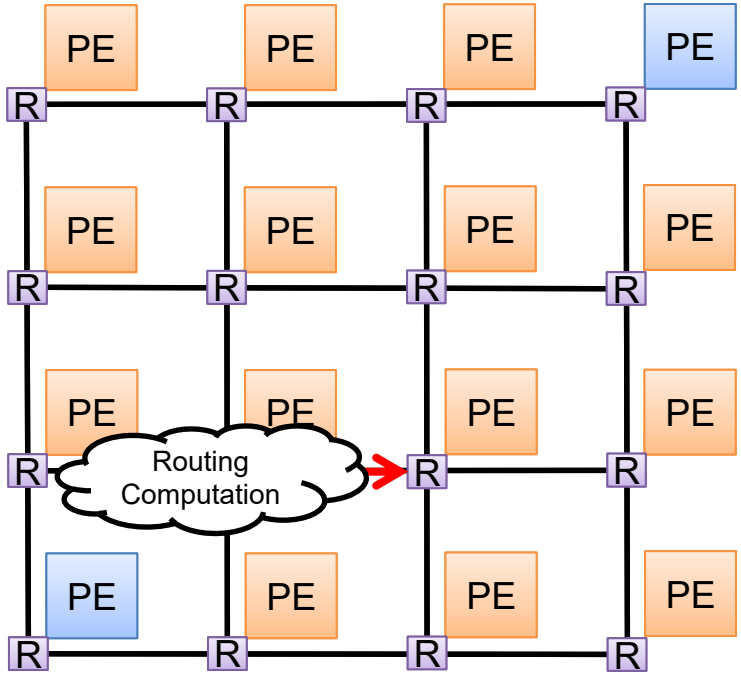


Distributed

Source vs. Distributed

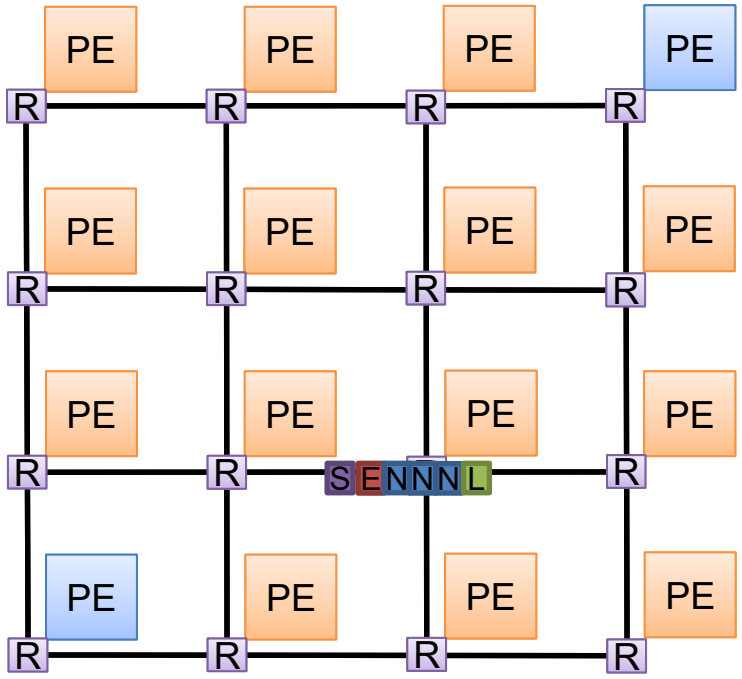


Source

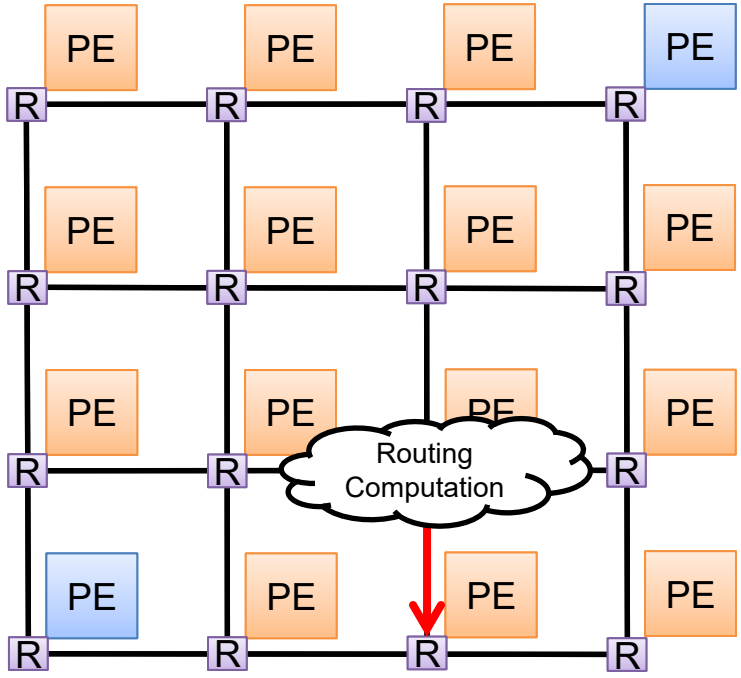


Distributed

Source vs. Distributed

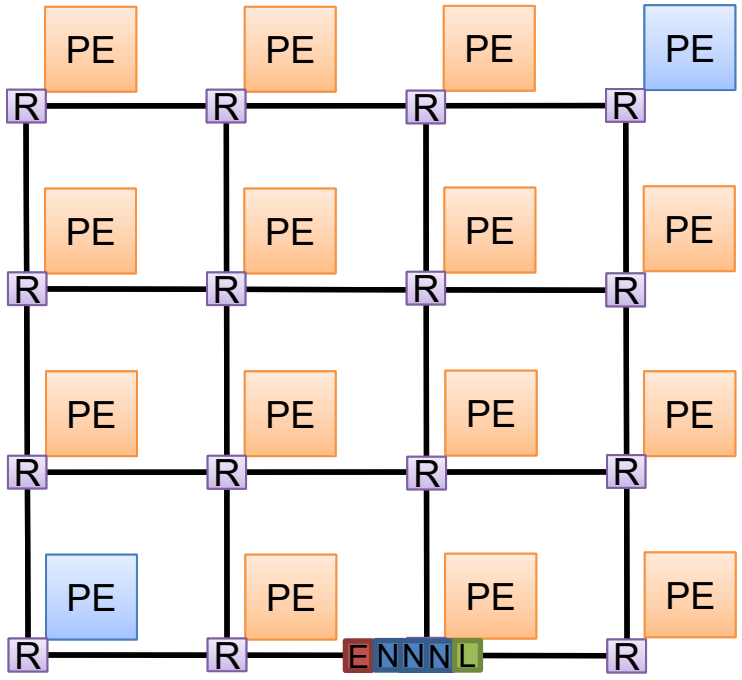


Source

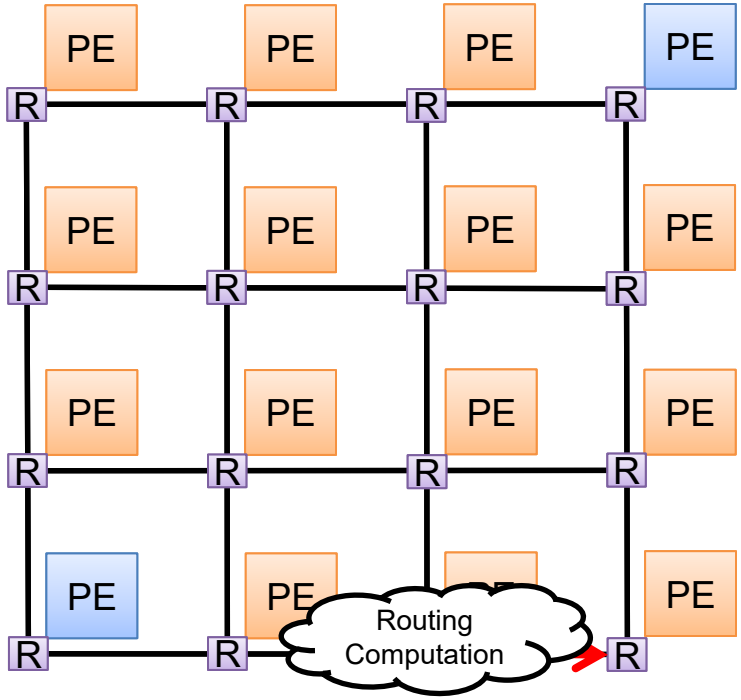


Distributed

Source vs. Distributed

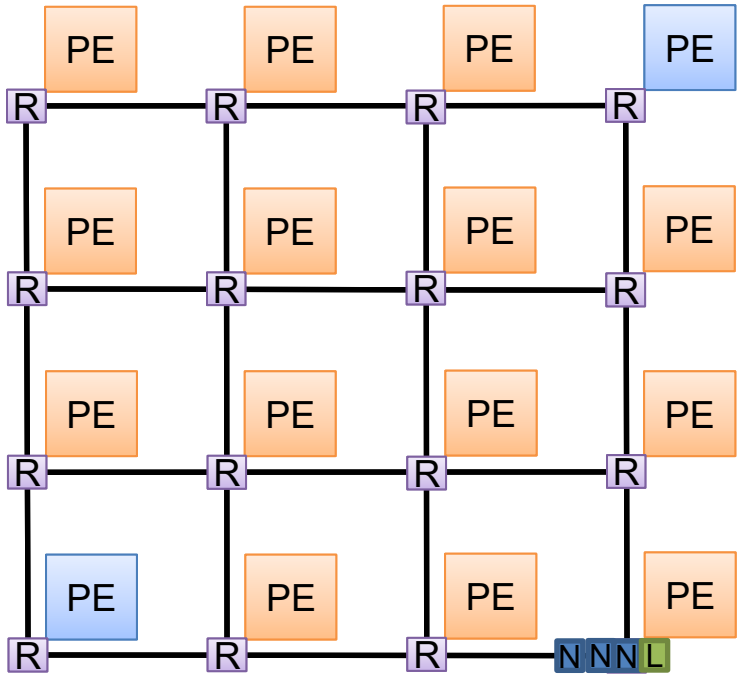


Source

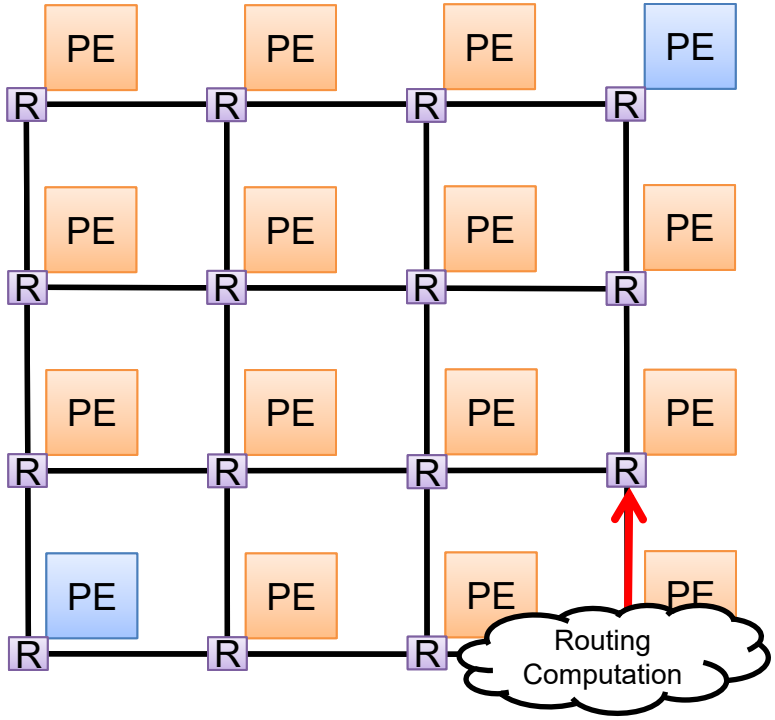


Distributed

Source vs. Distributed

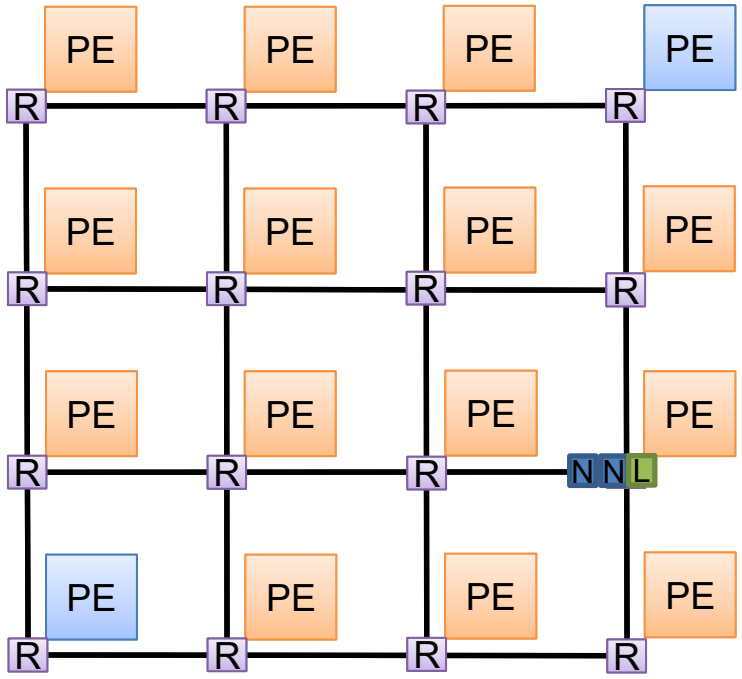


Source

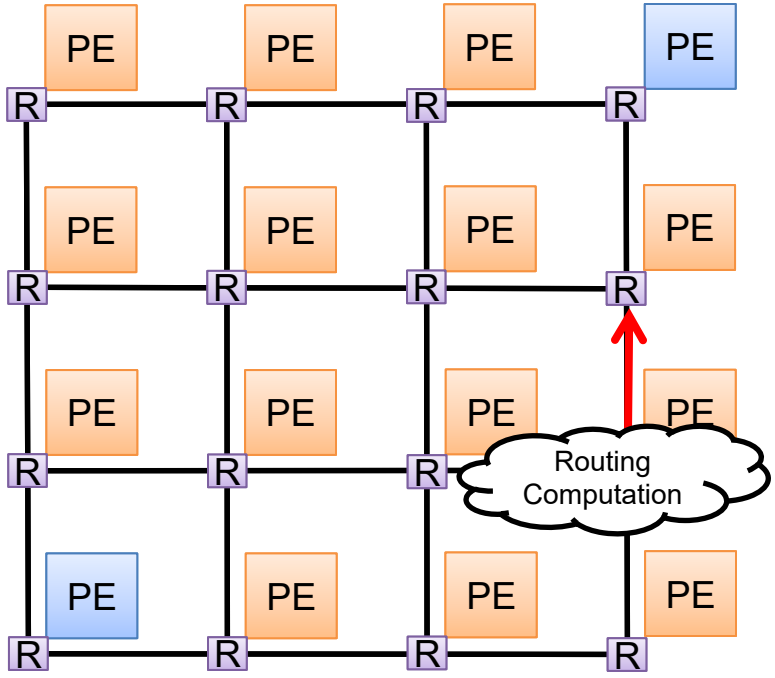


Distributed

Source vs. Distributed

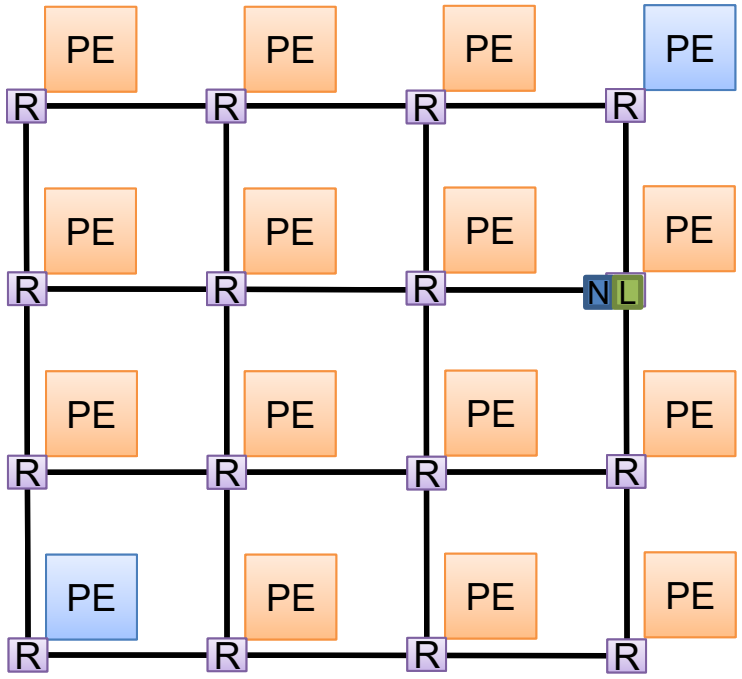


Source

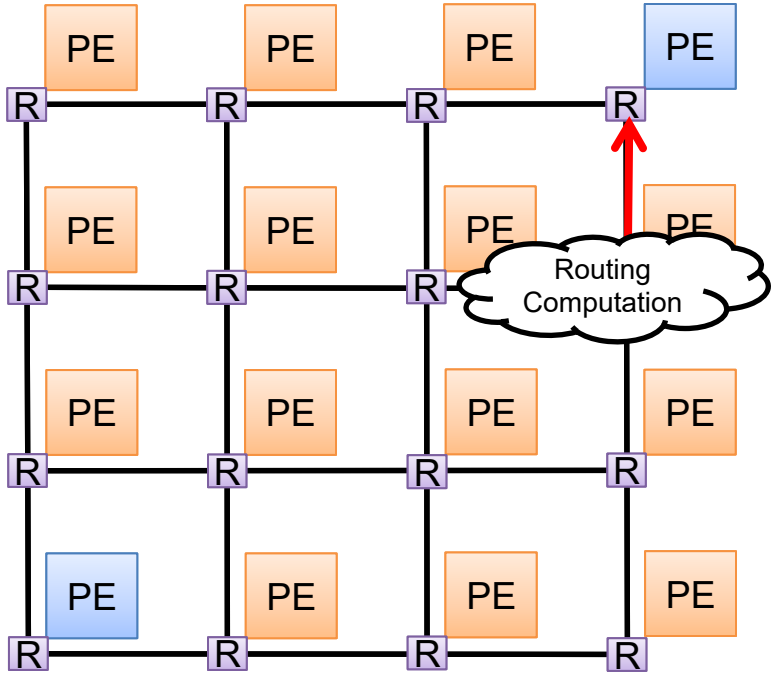


Distributed

Source vs. Distributed

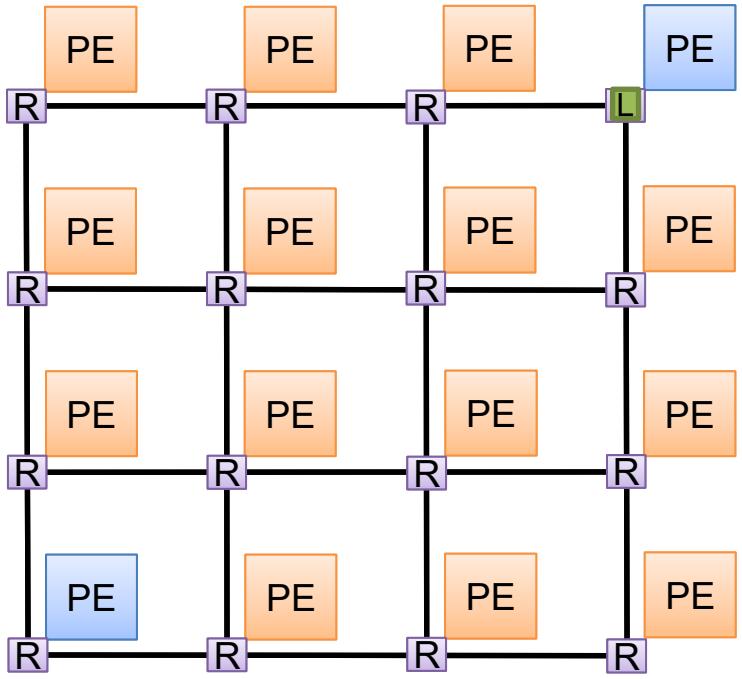


Source

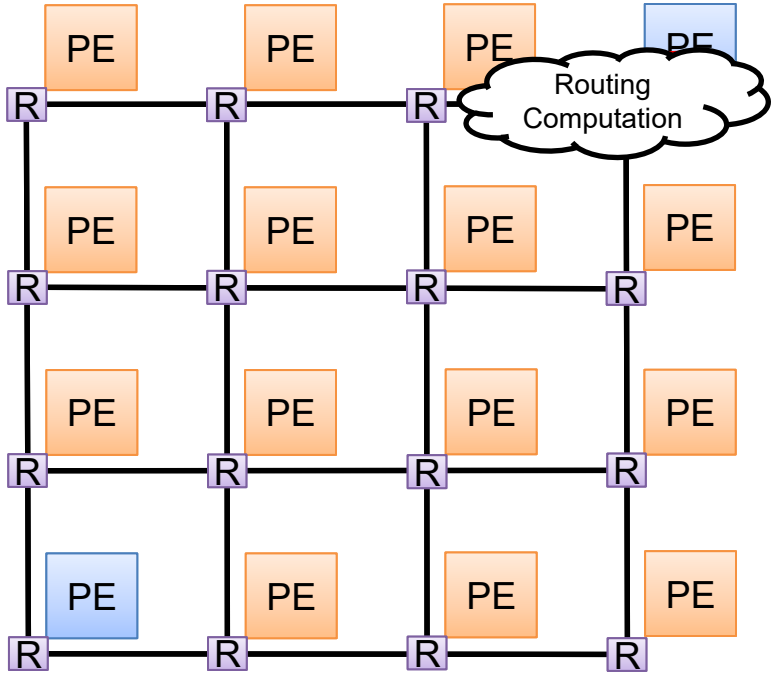


Distributed

Source vs. Distributed

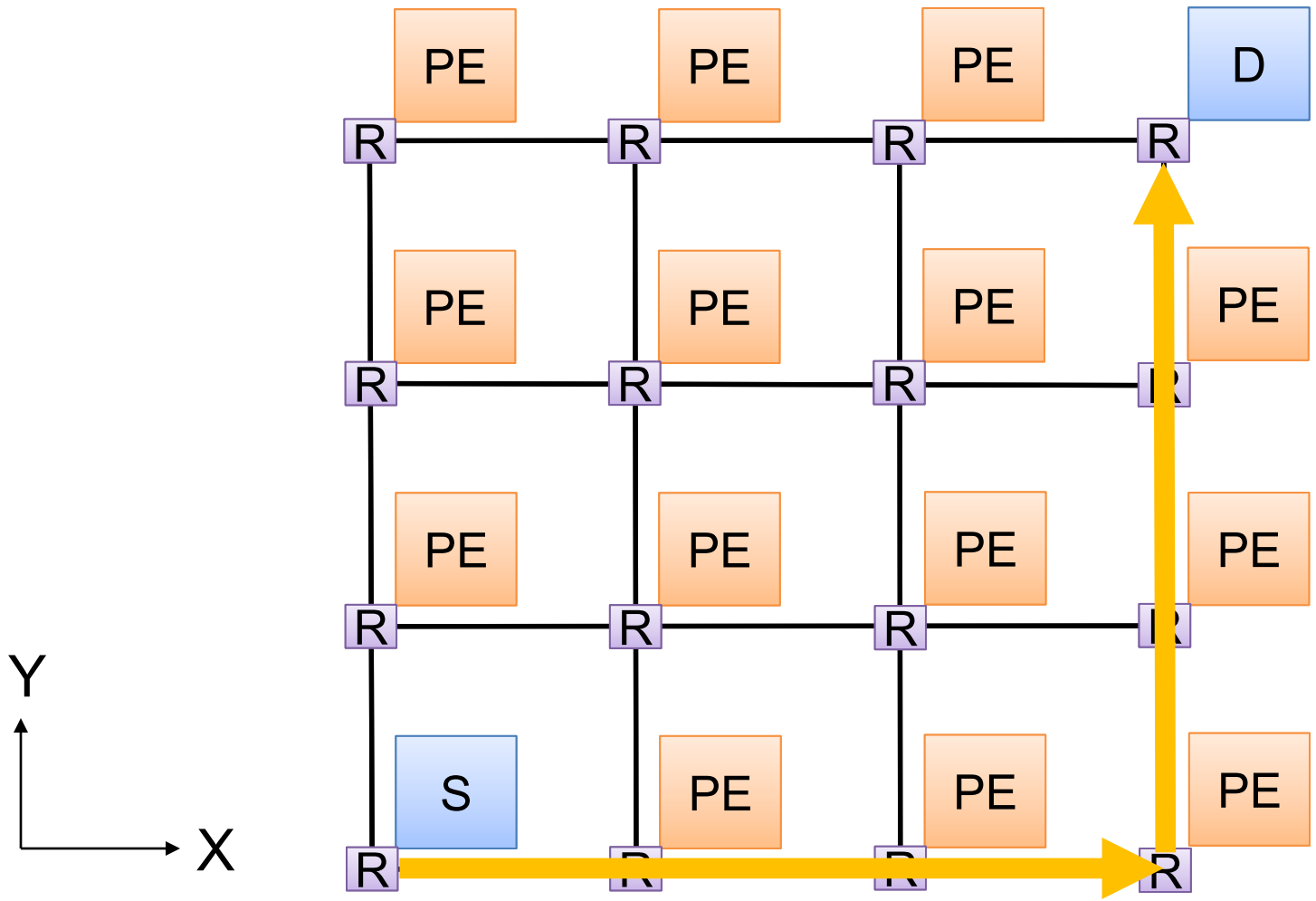


Source



Distributed

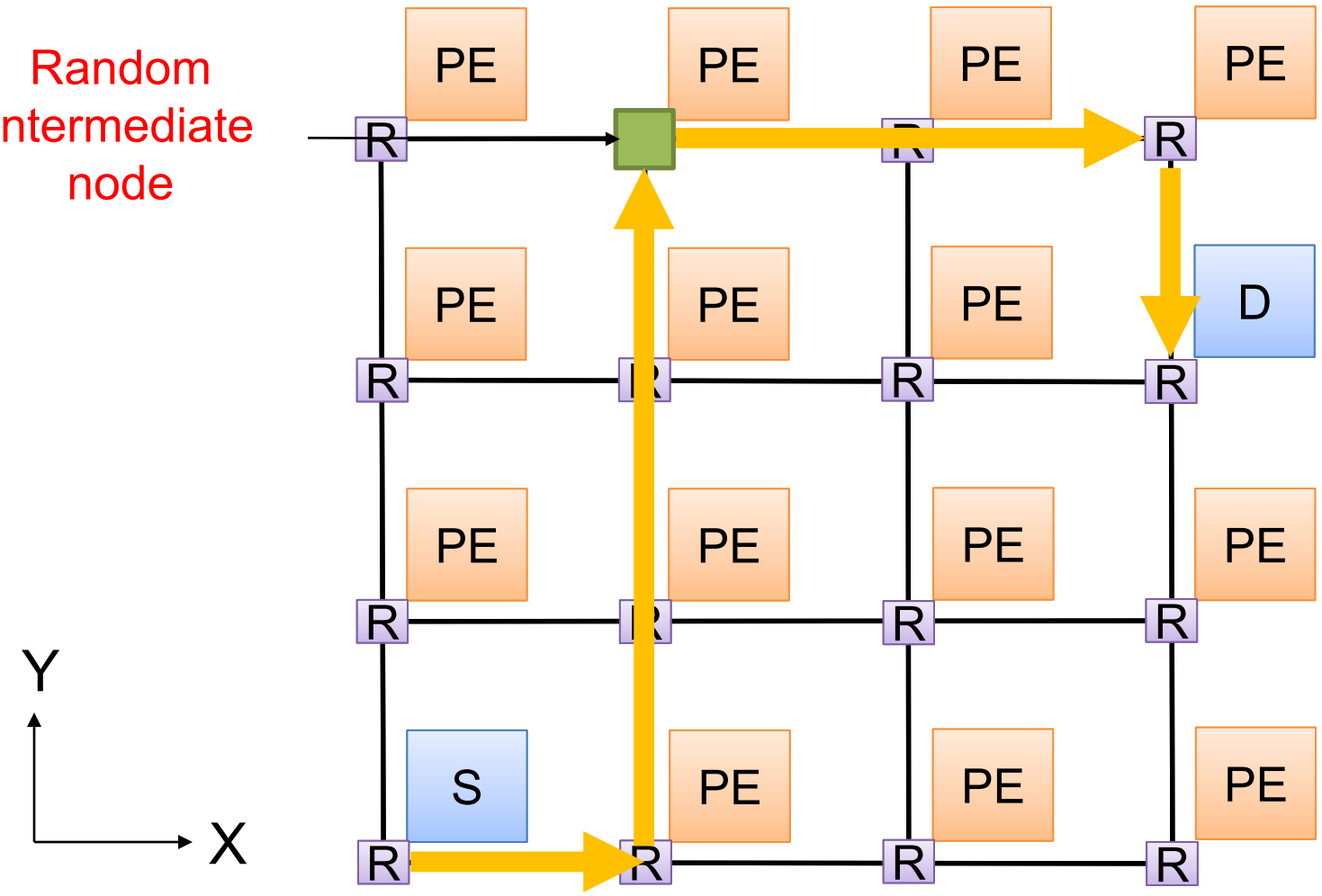
Routing examples



Dimension Ordered Routing
(XY Routing)

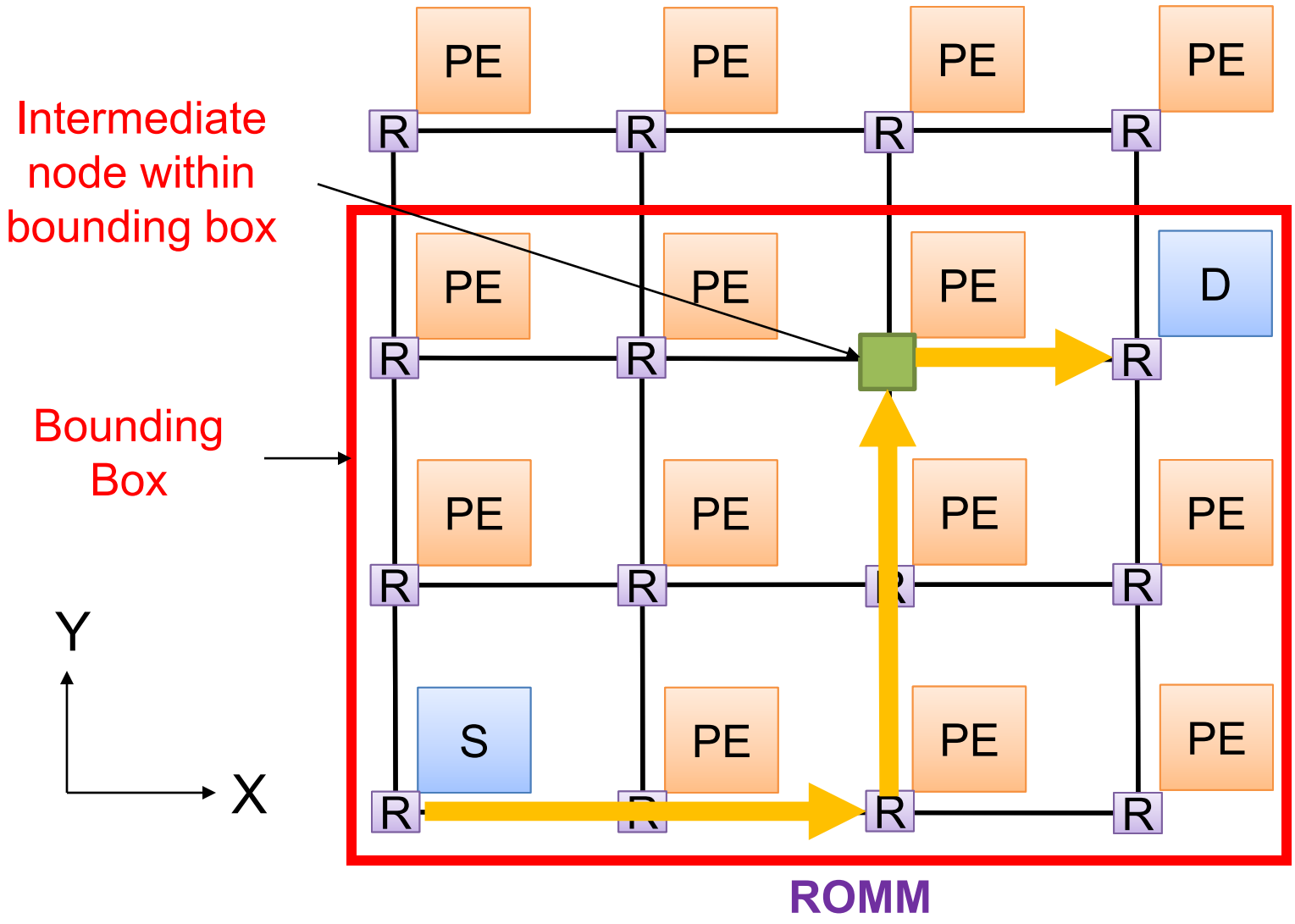
Routing examples

Random Intermediate node

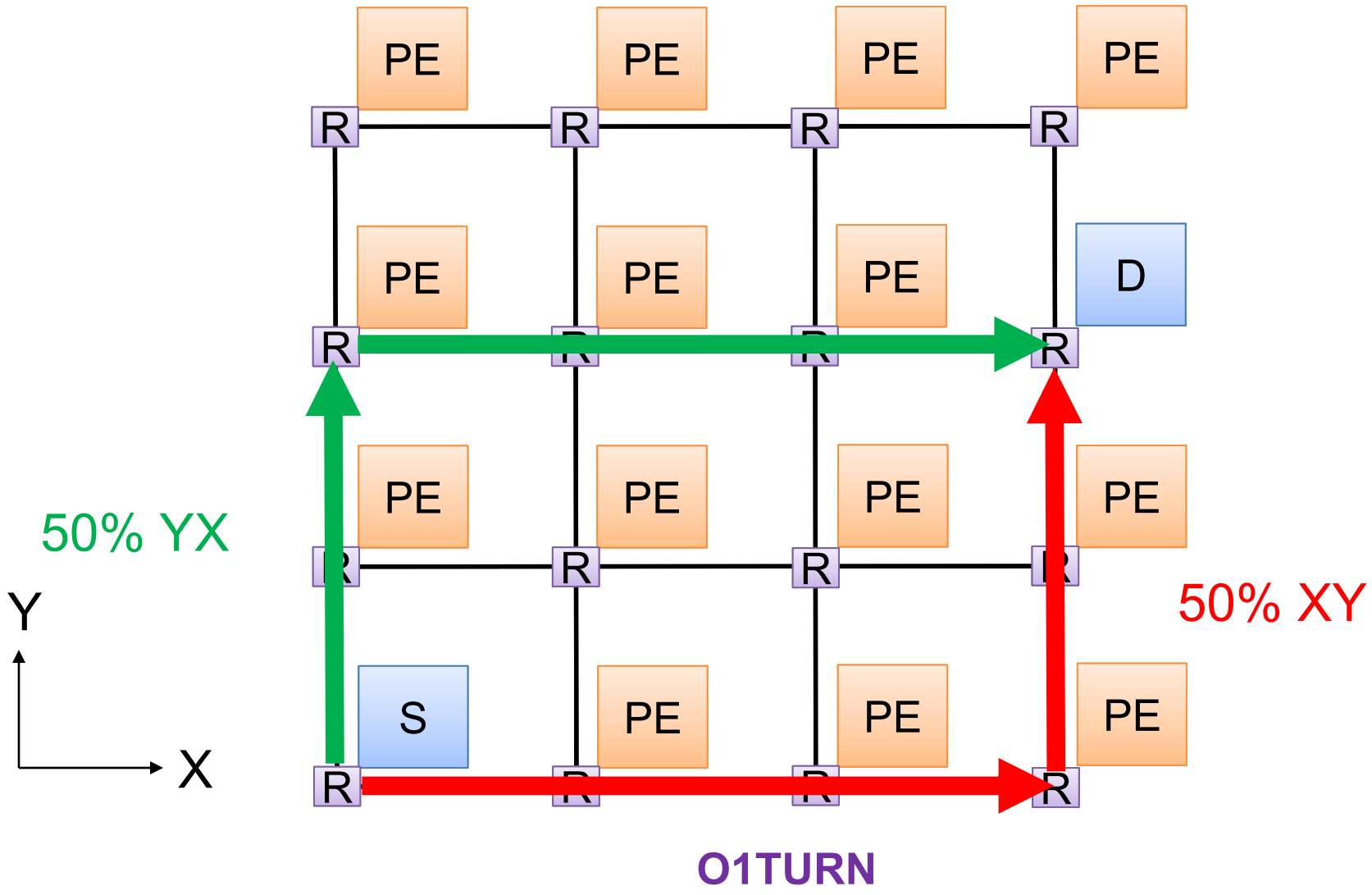


Valiant routing algorithm (VAL)

Routing examples

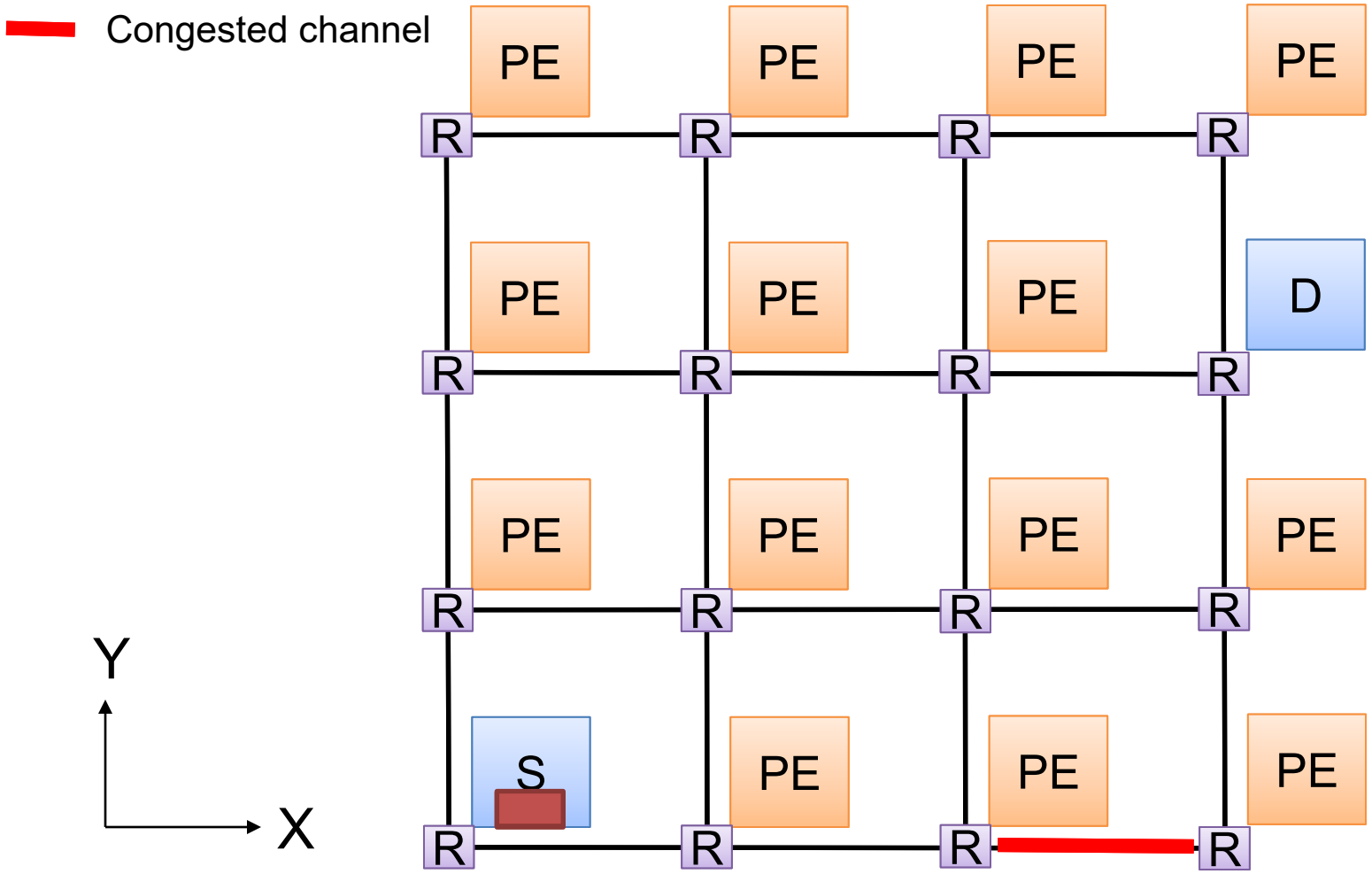


Routing examples



O1TURN

Routing examples



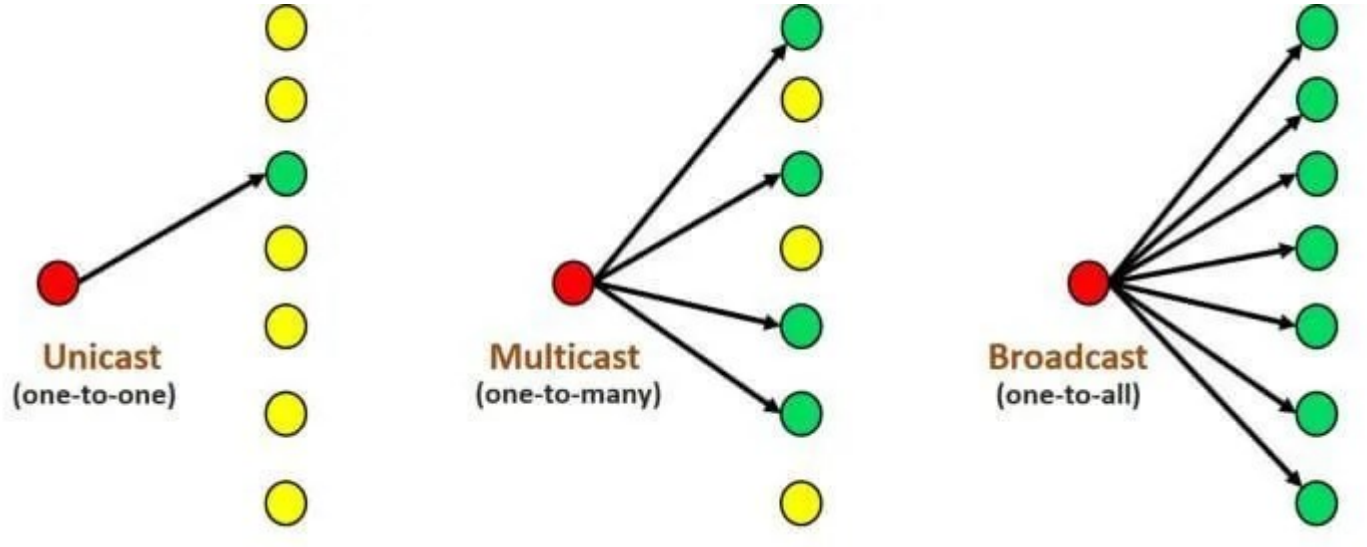
Dynamic XY (DyXY)

Summary of Routing Algorithms

- Deterministic algorithms are simple and inexpensive but they do utilize path diversity and thus are weak on load balancing
- Oblivious algorithms give often good results since they allow good load balancing and their effects are easy to analyse
- Adaptive algorithms although in theory superior, are complex and power hungry

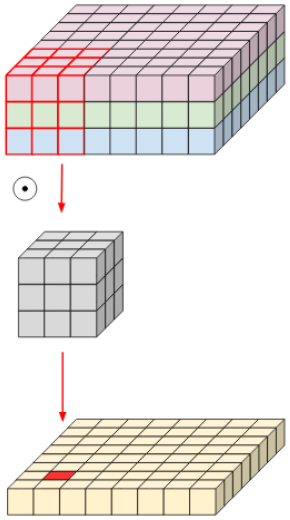
Data Transportation Methods

- **Unicast:** From one source to one destination i.e. One-to-One
- **Broadcast:** From one source to all possible destinations i.e. One-to-All
- **Multicast:** From one source to multiple destinations stating an interest in receiving the traffic i.e. One-to-Many

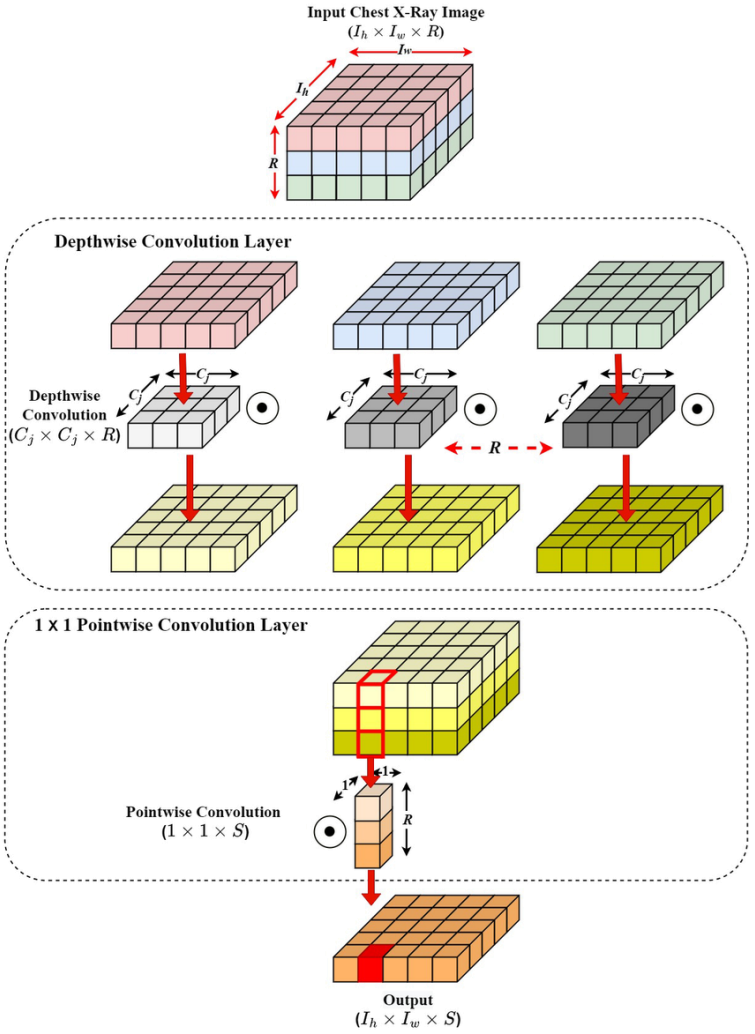


Data Transportation in DNN Operations (1/2)

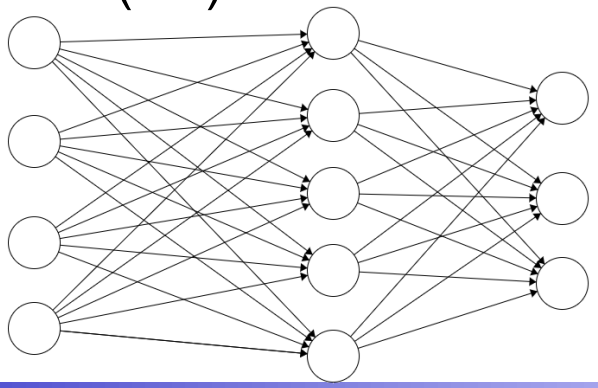
- Convolution (CONV)



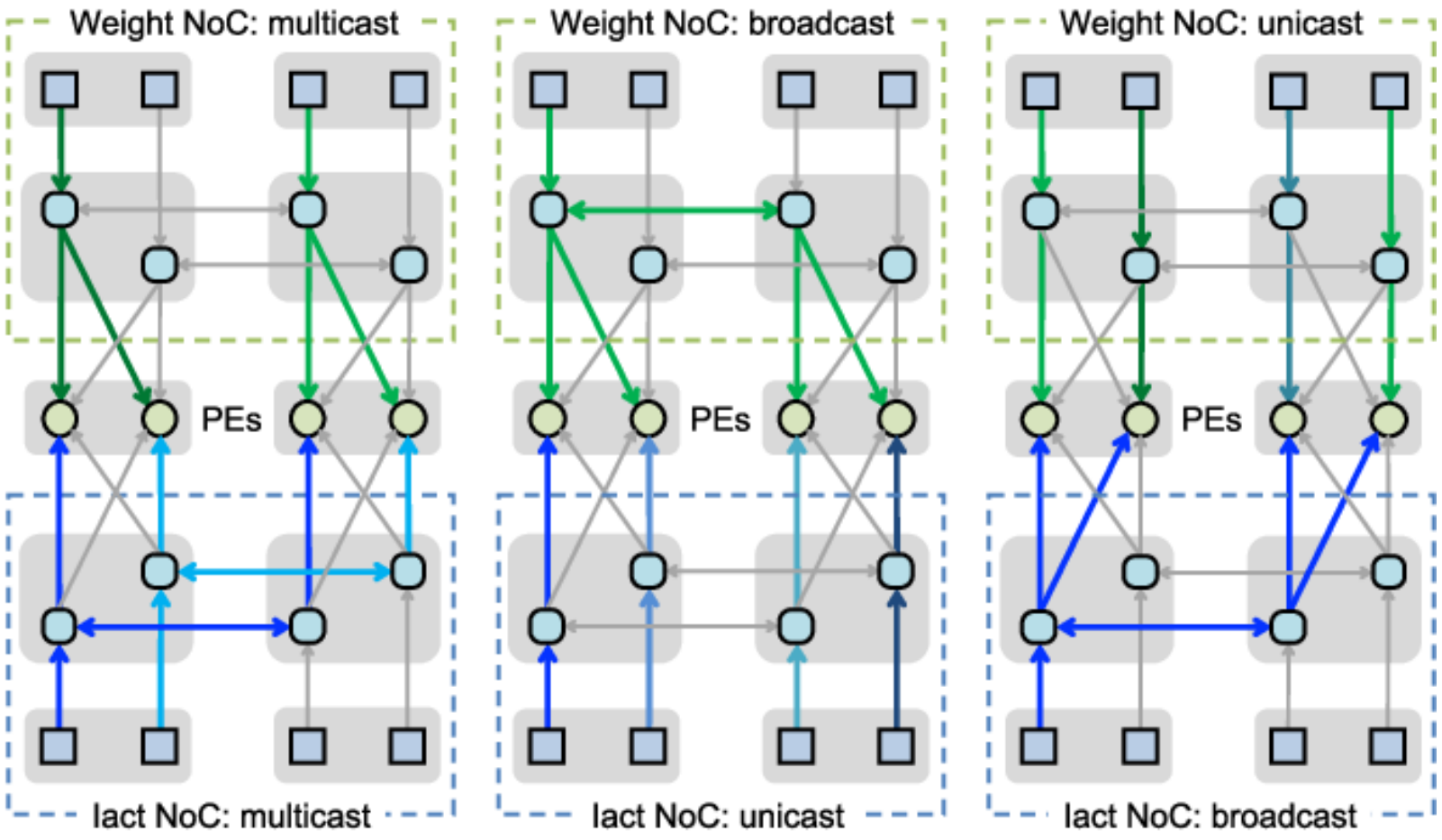
- Point-wise Convolution (PW CONV)



- Fully-connected (FC)



Data Transportation in DNN Operations



CONV **PW CONV** **FC**

Summary of Routing Algorithms

- Latency paramount concern
  - Minimal routing most common for NoC
  - Non-minimal can avoid congestion and deliver low latency
- NoC researchers favor DOR for simplicity and deadlock freedom
- Different data transportation methods are proper to different NN operations

Routing Mechanism

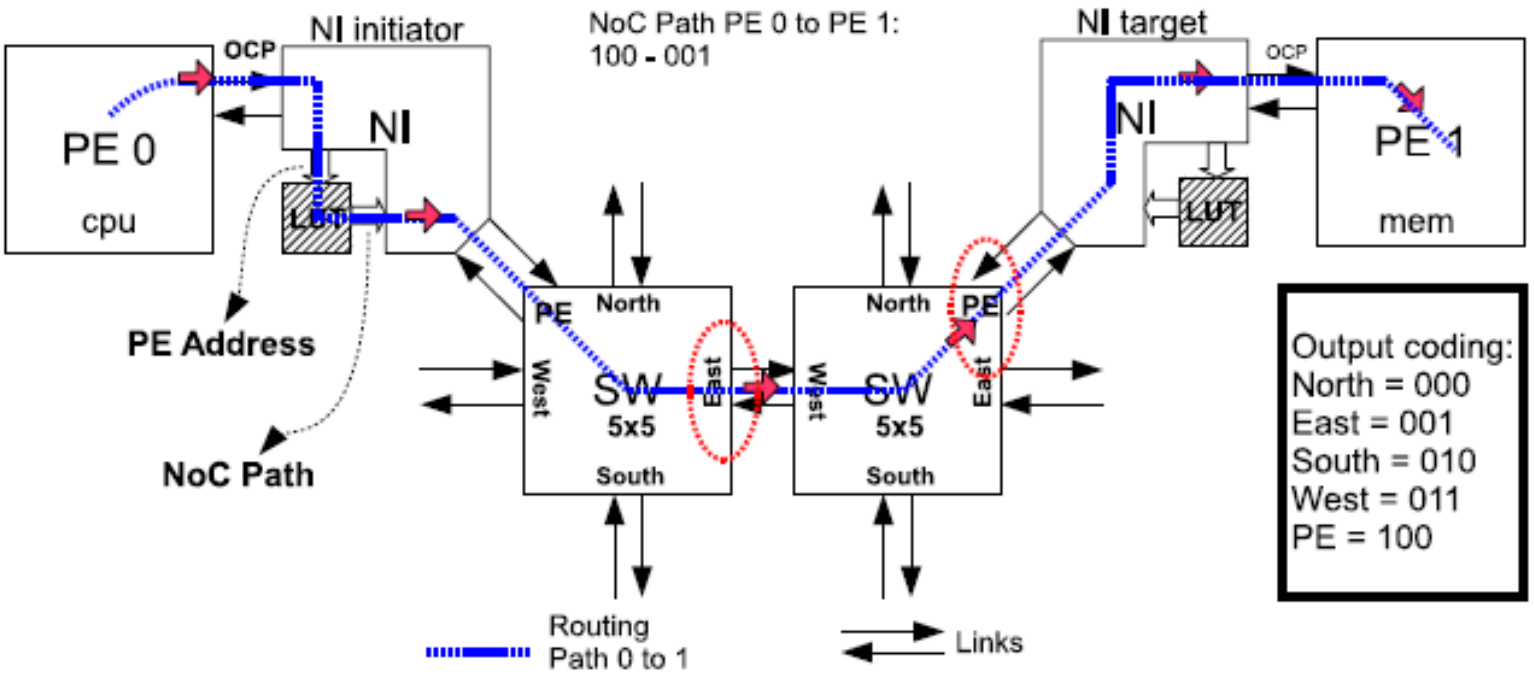
The term routing mechanics refers to the mechanism that is used to implement any routing algorithm.

- Two approaches:
  - Fixed routing tables at the source or at each hop
  - Algorithmic routing uses specialized hardware to compute the route or next hop at run-time

Table-based Routing

- Two approaches:
  - Source-table routing implements all-at-once routing by looking up the entire route at the source
  - Node-table routing performs incremental routing by looking up the hop-by-hop routing relation at each node along the route
- Major advantage
  - A routing table can support any routing relation on any topology.

Table-based Routing



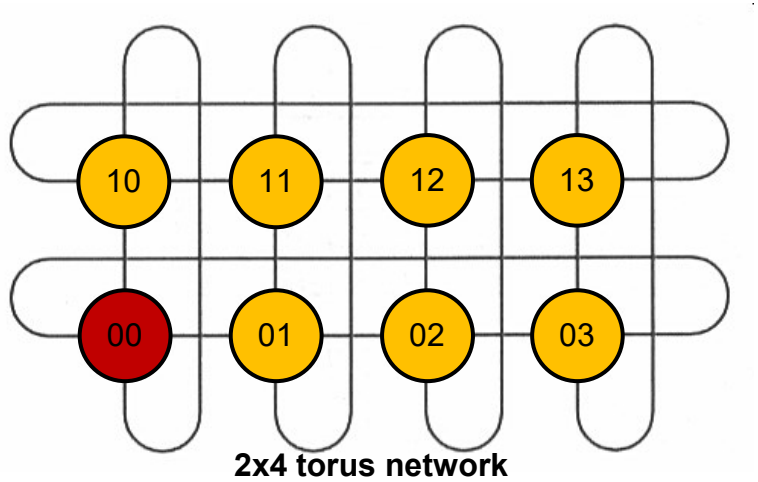
Example routing mechanism for deterministic source routing NoCs. The NI uses a LUT to store the route map.

Source Routing

- All routing decisions are made at the source terminal
- To route a packet we need
  - the table is indexed using the packet destination
  - a route or a set of routes are returned, one route is selected
  - the route is prepended and embedded in the packet
- Because of its speed, simplicity and scalability source routing is very often used for deterministic and oblivious routing

Source Routing - Example

- The example shows a routing table for a 2x4 torus network
- In this example there are two alternative routes for each destination
- Each node has its own routing table



2x4 torus network

Source routing table for node 00 of 4x2 torus network

Destination	Route 0	Route 1
00	X	X
01	EX	WWWX
02	EEX	WWX
03	WX	EEEX
10	NX	SX
11	NEX	ENX
12	NEEX	WWNX
13	NWX	WNX

index →

select

Example:
-Routing from 00 to 12
-Table is indexed with 12
-Two routes:
NEEX and WWNX

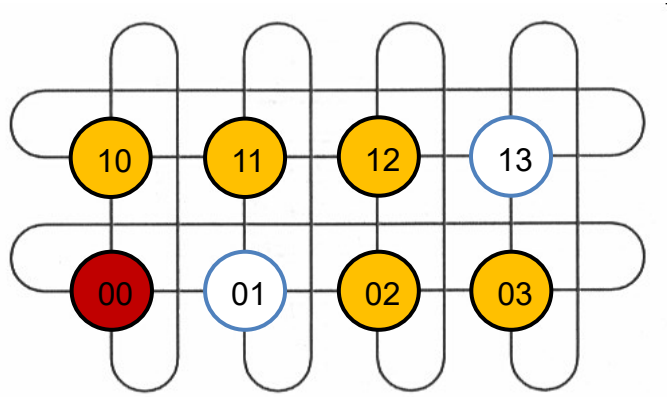
-The source arbitrarily selects NEEX

Node-Table Routing

- Table-based routing can also be performed by placing the routing table in the **routing nodes** rather than in the terminals
- Node-table routing is appropriate for **adaptive routing** algorithms, since it can use state information at each node
- A table lookup is required, when a packet arrives at a router, which takes additional time compared to source routing
- Scalability is sacrificed, since different nodes need tables of varying size
- Difficult to give two packets arriving from a different node a different way through the network without expanding the tables

Example of Node-Table Routing

- Table shows a set of routing tables
- There are two choices from a source to a destination



Routing Table for Node 00

	From															
To	00		01		02		03		10		11		12		13	
00	X	X	W	**N**	W	E	E	**N**	S	N	S	W	S	W	S	E
01	E	**N**	X	X	W	**S**	E	W	S	**W**	S	N	S	W	S	W
02	E	W	E	**N**	X	X	W	**N**	S	W	S	E	S	N	S	W
03	W	**N**	E	W	E	**N**	X	X	S	**E**	S	E	S	E	S	N
10	N	S	N	W	N	W	N	E	X	X	W	**N**	W	E	E	**N**
11	N	E	N	S	N	W	N	W	E	**S**	X	X	W	**N**	E	W
12	N	E	N	E	N	S	N	W	E	W	E	**N**	X	X	W	**N**
13	N	W	N	E	N	E	N	S	W	**S**	E	W	E	**N**	X	X

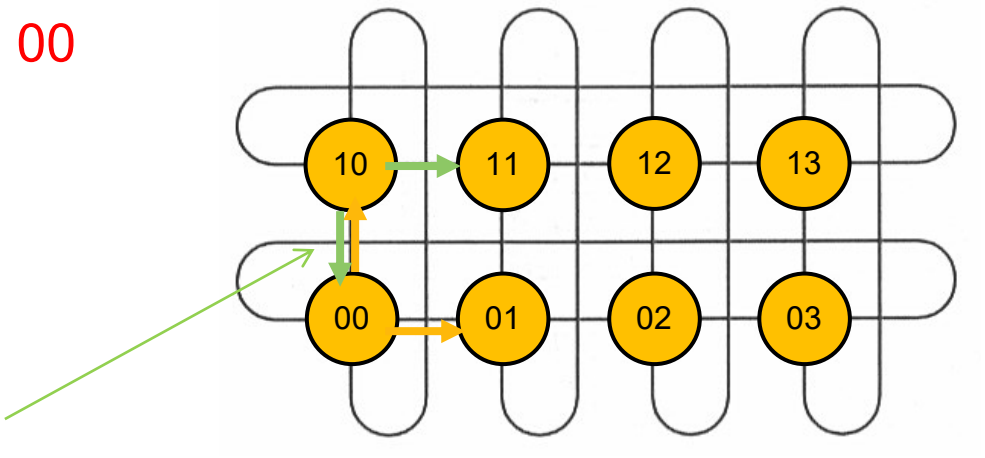
Note: Bold font ports are misroutes

Example of Node-Table Routing

Livelock can occur

A packet passing through node 00 destined for node 11.

If the entry for (00->11) is N , go to 10 and (10-> 11) is S
=> 00 <-> 10 (livelock)



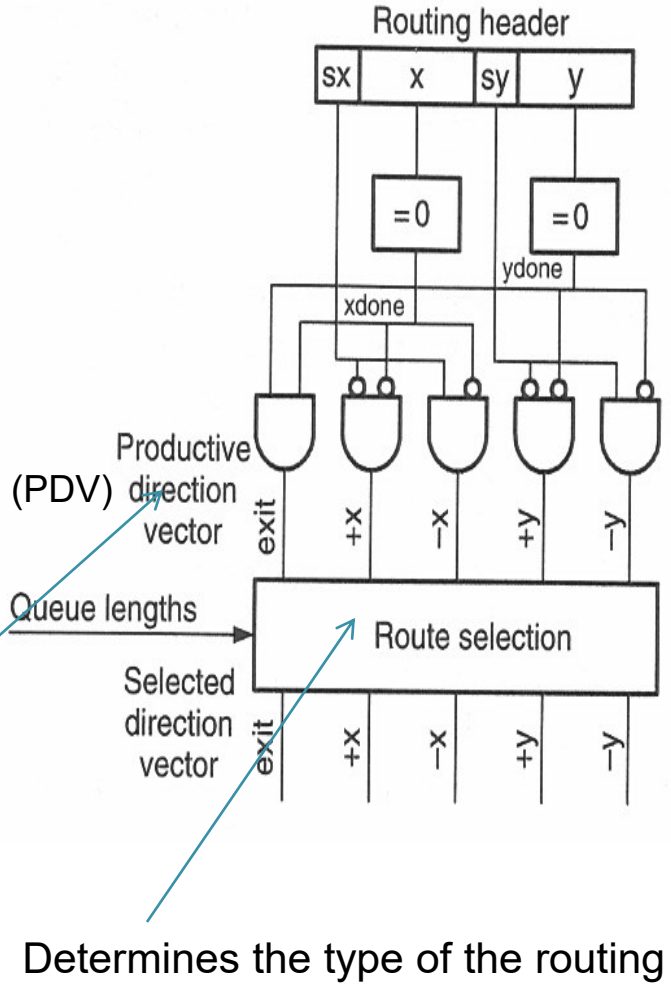
	From															
To	00		01		02		03		10		11		12		13	
00	X	X	W	**N**	W	E	E	**N**	S	N	S	W	S	W	S	E
01	E	**N**	X	X	W	**S**	E	W	S	**W**	S	N	S	W	S	W
02	E	W	E	**N**	X	X	W	**N**	S	W	S	E	S	N	S	W
03	W	**N**	E	W	E	**N**	X	X	S	**E**	S	E	S	E	S	N
10	N	S	N	W	N	W	N	E	X	X	W	**N**	W	E	E	**N**
11	N	E	N	S	N	W	N	W	E	**S**	X	X	W	**N**	E	W
12	N	E	N	E	N	S	N	W	E	W	E	**N**	X	X	W	**N**
13	N	W	N	E	N	E	N	S	W	**S**	E	W	E	**N**	X	X

Algorithmic Routing

- Instead of using a table, algorithms can be used to compute the next route
- In order to be fast, algorithms are usually not very complicated and implemented in hardware

Example: Algorithmic Routing

- Dimension-Order Routing
  - sx and sy indicated the preferred directions
    - sx=0, +x; sx=1, -x
    - sy=0, +y; sy=1, -y
  - x and y represent the number of hops in x and y direction
  - The PDV is used as an input for selection of a route



Determines the type of the routing

Indicates which channels advance the packet

Example: Algorithmic Routing

- A router for minimal routing – Implemented by randomly selecting one of the active bits of the PDV as the selected direction
- A router for minimal adaptive routing - Achieved by making selection based on the length of the respective output Qs.
- A router for fully adaptive routing – Implemented by picking up unproductive direction if Qs > threshold results

Generic NoC Router Architecture