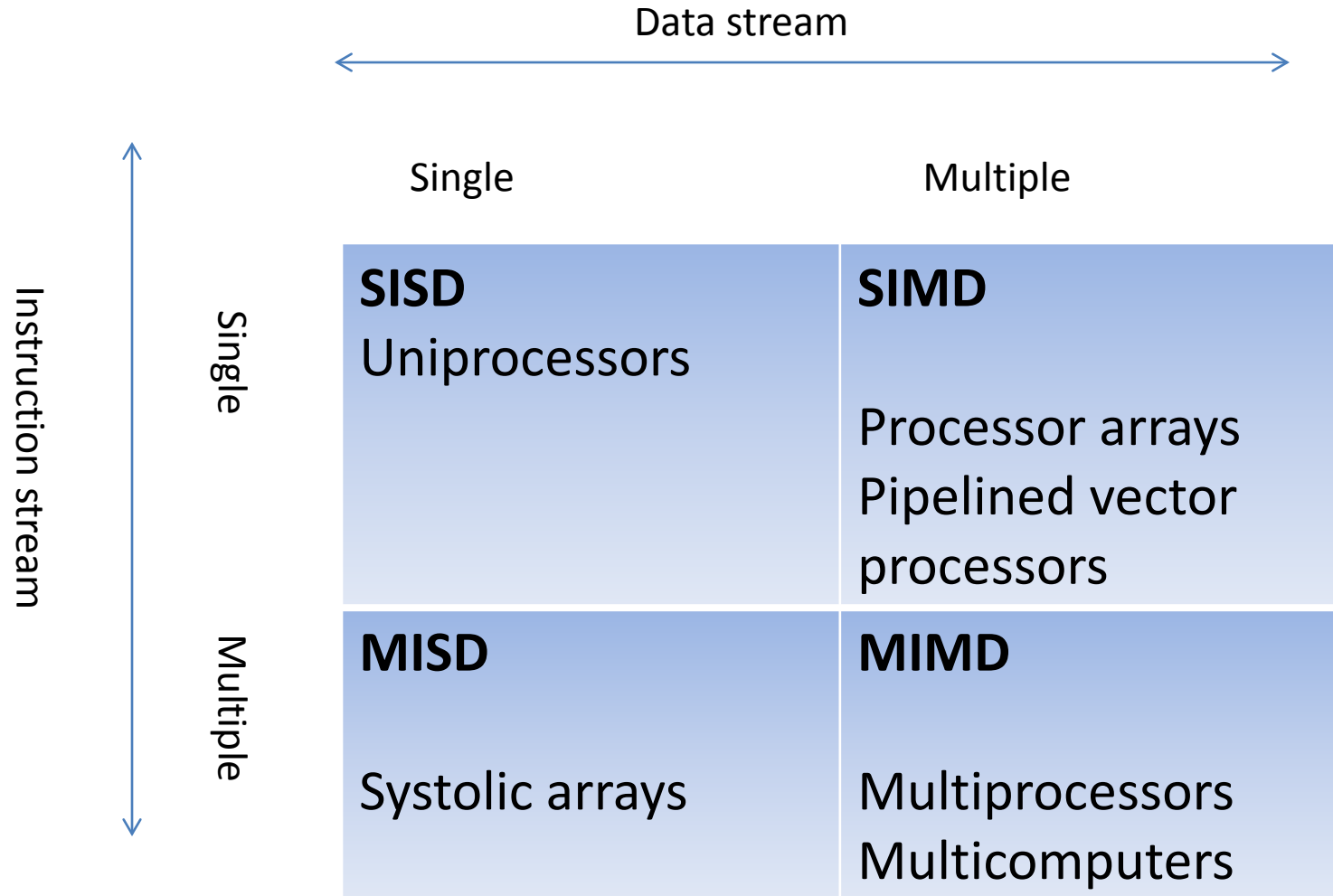


Lecture 2 Parallel Programming Platforms

Flynn's Taxonomy

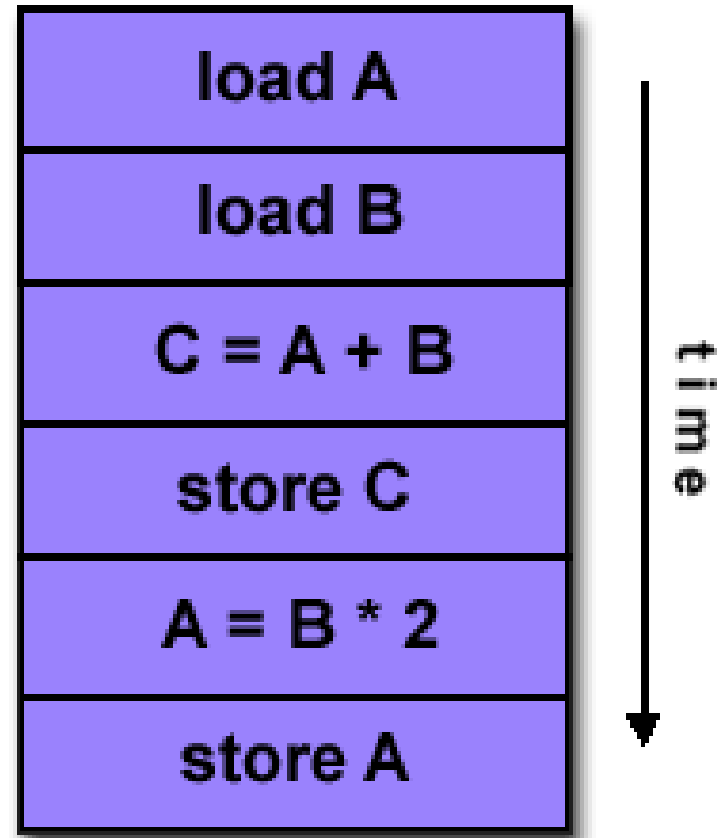
In 1966, Michael Flynn classified systems according to numbers of instruction streams and the number of data stream.



SISD Machine

Example: single CPU computers (serial computer)

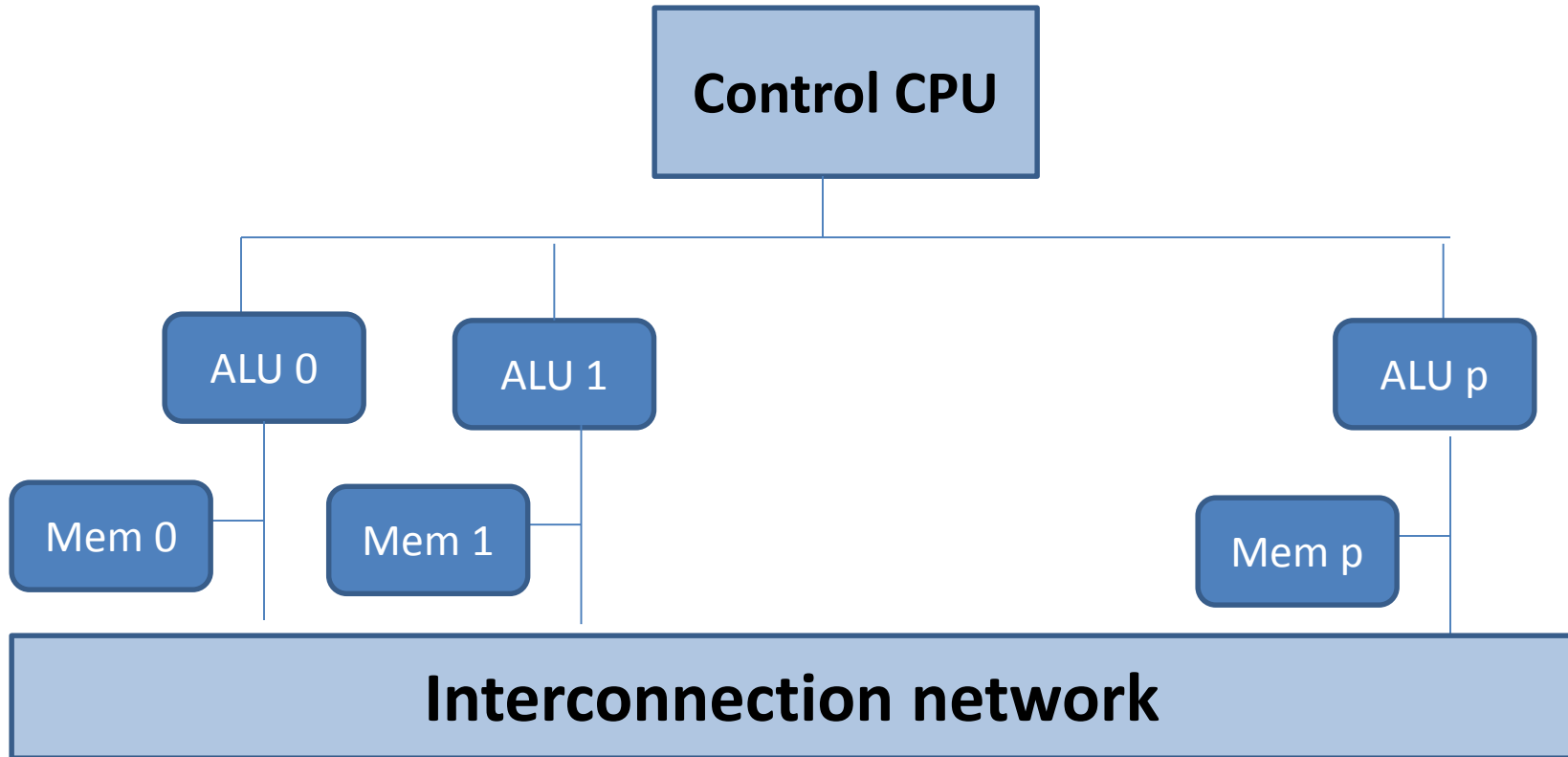
- Single instruction: Only one instruction stream is acted on by CPU during one clock cycle
- Single data: Only one data stream is used as input during one clock cycle
- Deterministic execution



SIMD Machine (I)

- A parallel computer
- It typically has a single CPU devoted exclusively to control, a large number of subordinate ALUs, each with its own memory and a high-bandwidth internal network.
- Control CPU broadcasts an instruction to all subordinate ALUs, and each of the subordinate ALUs either executes the instruction or it is idle.
- Example: CM-1, CM-2, IBM9000

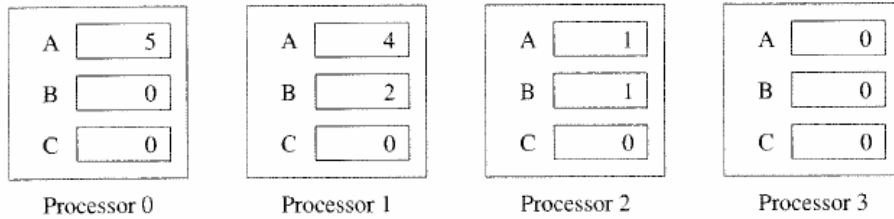
SIMD Machine (2)



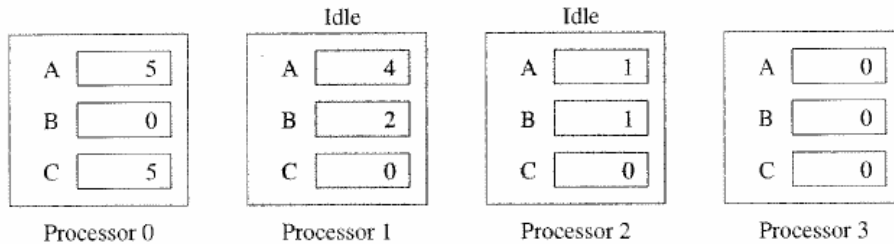
SIMD Machine (3)

```
if (B == 0)
    C = A;
else
    C = A/B;
```

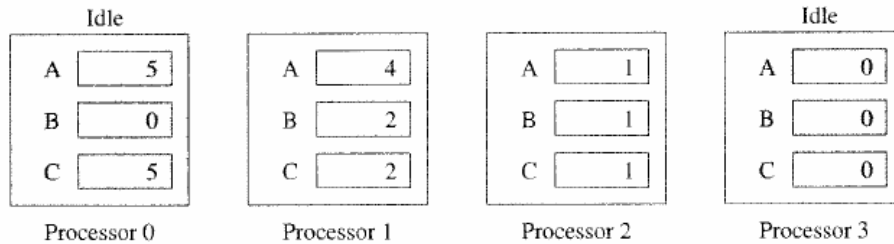
(a)



Initial values



Step 1



Step 2

(b)

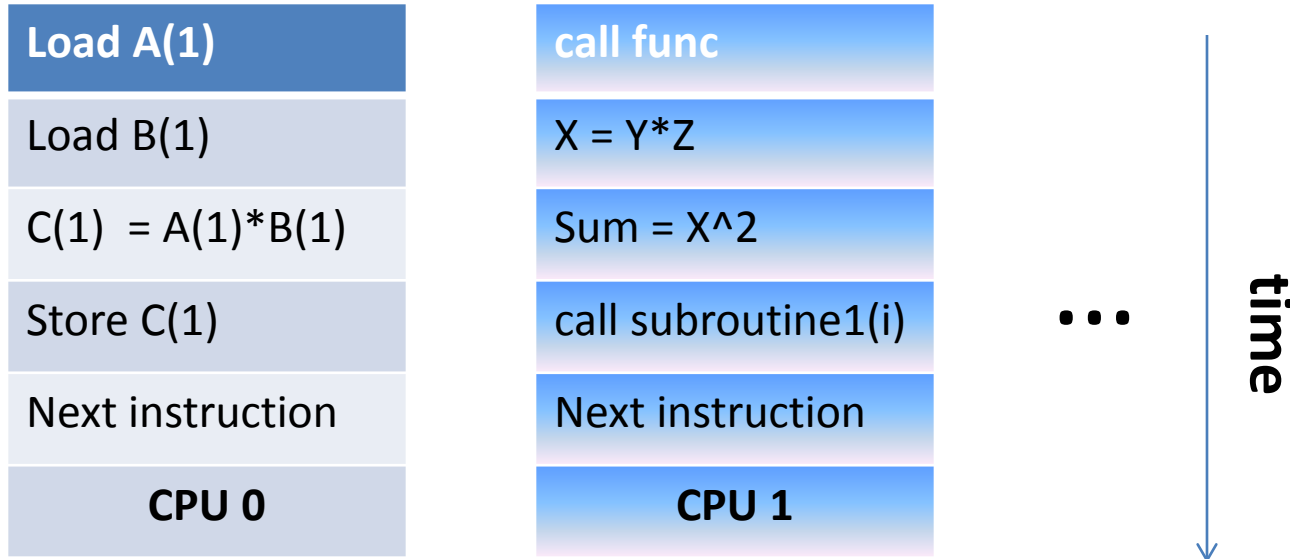
From Introduction to Parallel Computing

Figure 2.4 Executing a conditional statement on an SIMD computer with four processors: (a) the conditional statement; (b) the execution of the statement in two steps.

MIMD Machine (I)

- Most popular parallel computer architecture
- Each processor is a full-fledged CPU with both a control unit and an ALU. Thus each CPU is capable of executing its own program at its own space.
- Execution is *asynchronous*. Processors can also be specifically programmed to synchronize with each other.
- Examples: networked parallel computers, symmetric multiprocessor (SMP) computer.

MIMD Machine (II)



Further classification according to memory access:

- **Shared-memory** system
- **Distributed-memory** system (Message-passing)

Shared-Memory MIMD Machine (I)

- Multiple processors can operate independently, but share the same memory resources (a global address space).
- Change in a memory location made by one processor is visible to all other processors.
- Two classes of shared-memory architecture based on network connecting memory modules: Bus-based shared-memory architecture (SGI Challenge XL); Switch-Based architecture (Convex SPP1200).
- Classes of shared-memory systems based on time taken by a processor to access any memory: uniform memory access (**UMA**), **NUMA**.

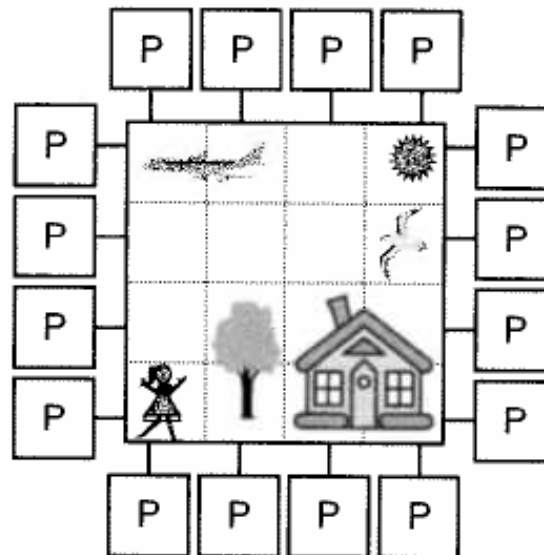
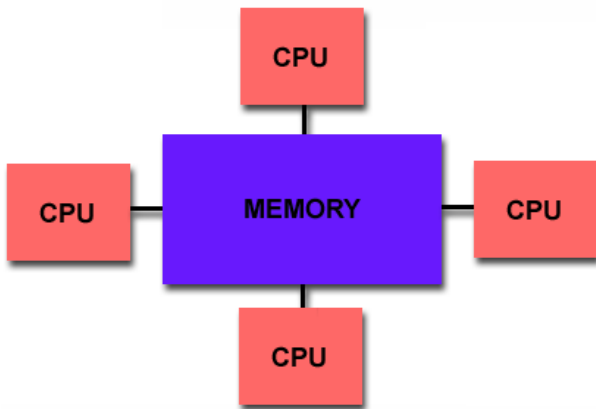
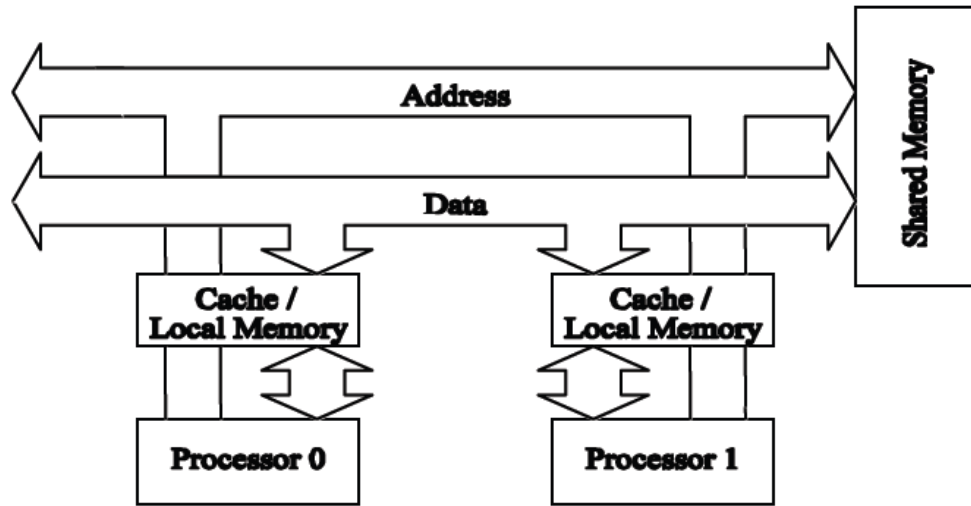


Image recognition.
Image partitioned into 16 sections, each being analyzed by a different CPU. (Tanenbaum, Structured Computer Organization)

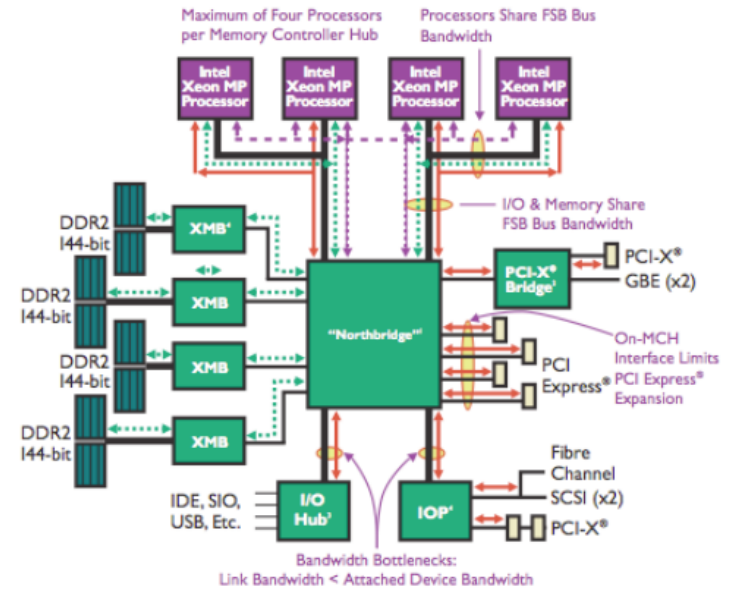
Shared-Memory MIMD Machine (II)

Bus-based shared-memory architecture



Bus-based interconnect
with local memory/cache

Intel Xeon MP Processor-based 4P Server



Dual-bus (circa 2005)

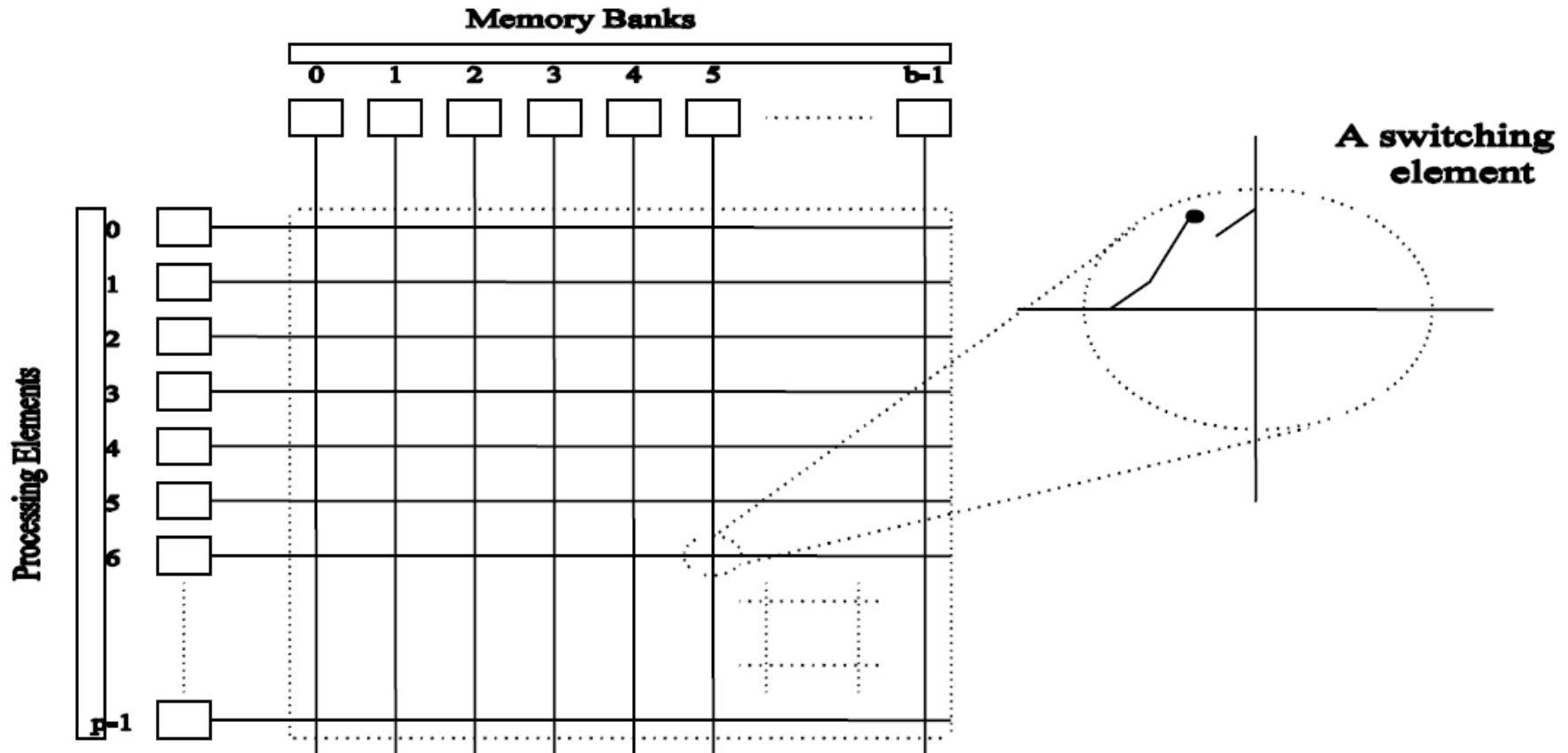
In principle, at a time, only one message is allowed to be sent. So poor performance.

Bus-based network

- A bus-based network consists of a shared medium that connects all nodes
- The cost of a bus-based network scales linearly with respect to the number of processors p
- The distance between any two nodes in the network is const. $O(1)$.
- Ideal for broadcasting information
- Disadvantage: bounded bandwidth & blocking. Performance is not scalable with respect to the number of processors p .

Shared-Memory MIMD Machine (III)

Switch-based shared-memory architecture, supports point-to-point communication among pairs of processing node and memory banks.



- Excellent performance
- Expensive
- Switch complexity is difficult to realize at high data rates

Crossbar network

- A crossbar network uses a grid of switches (or switching nodes) to connect p processors to b memory banks.
- It is non-blocking: connection of a processing node to a memory bank does not block the connection of any other processing nodes to other memory banks.
- Usually, b is at least on the order of p , the complexity of the crossbar network is $\Omega(p * p)$.
- The total number of switches is $\Theta(pb)$
- Good performance scalability, bad cost scalability.

Shared-Memory MIMD Machine (IV)

- Cache coherence

For any shared-memory architecture that allows the caching of shared variables, if processor A update a shared variable x in its cache, how to make sure values of all copies of x are current.

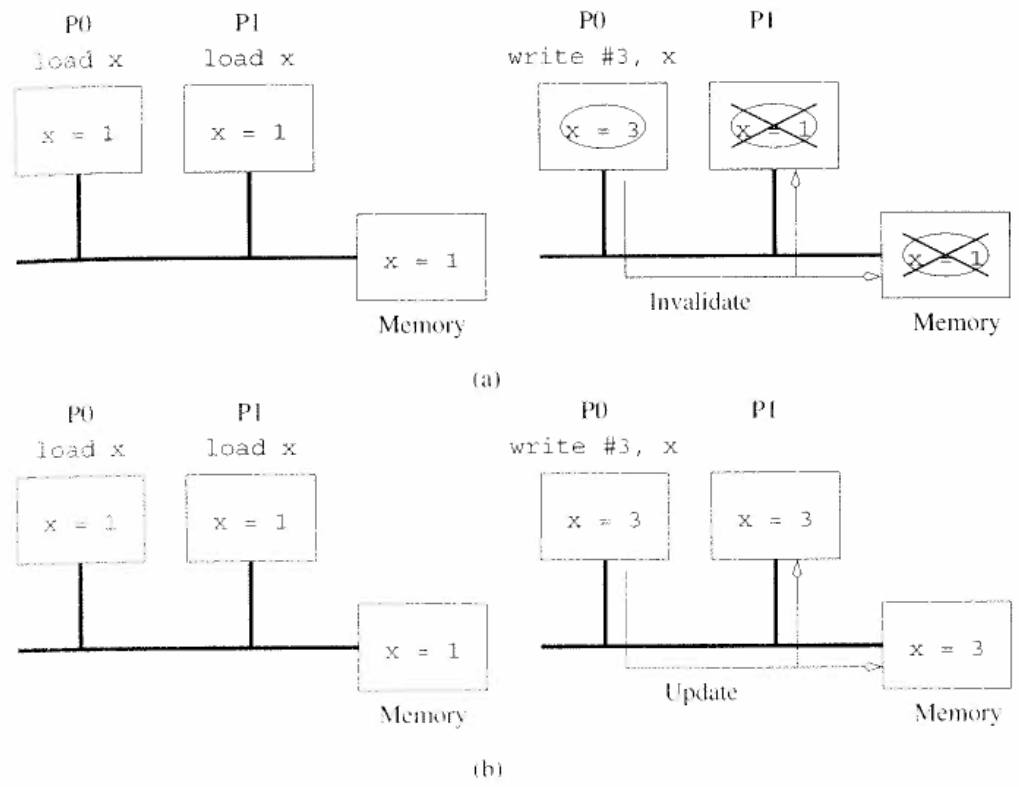


Figure 2.21 Cache coherence in multiprocessor systems: (a) Invalidate protocol; (b) Update protocol for shared variables.

Good News: Cache coherence is achieved at the hardware level through **snoopy protocol** etc.

Distributed-Memory MIMD Machine (I)

- Each processor has its own private memory.
- A communication network is built to connect inter-processor memory
- No concept of global address space of memory across all processors
- No cache coherence concept
- Data exchange is through message passing

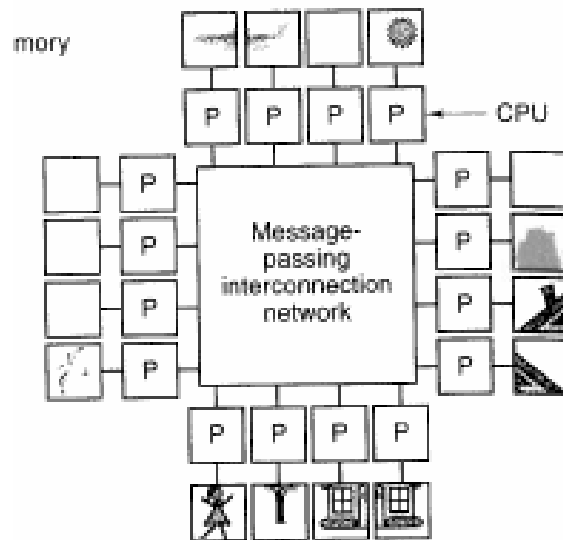
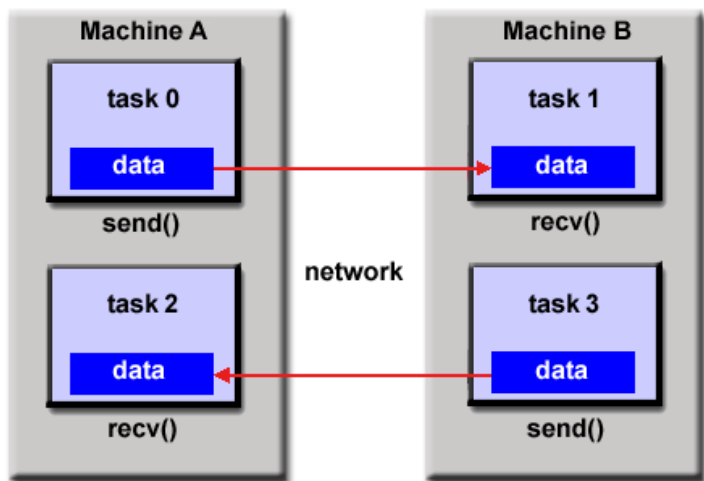
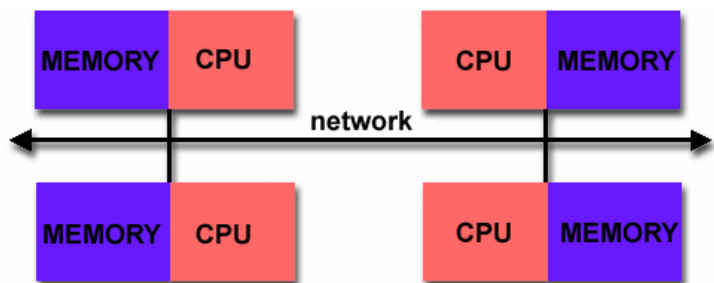
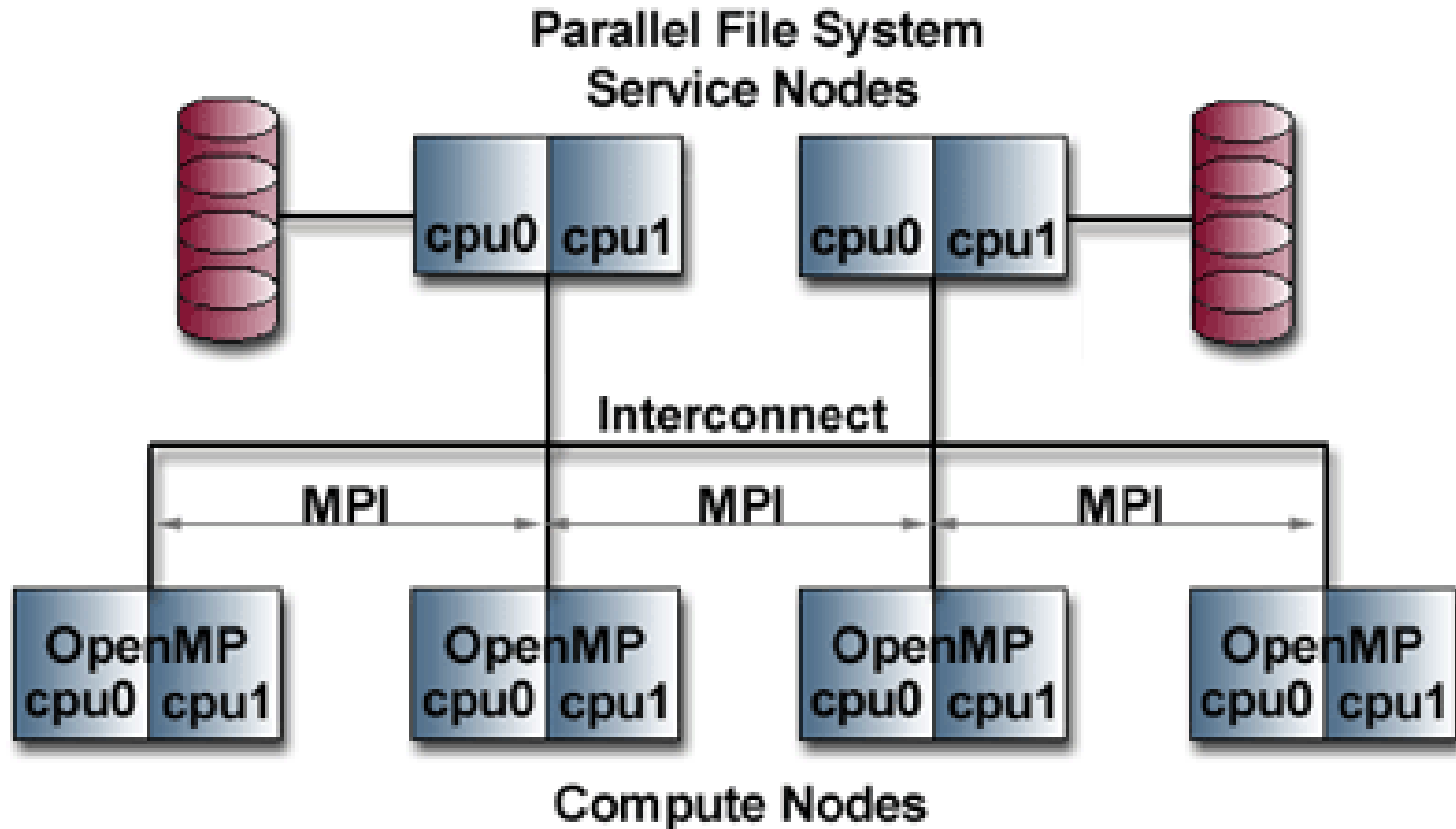


Image recognition. The image is split among the 16 memories. (Tanenbaum, Structured Computer Organization)

Case Study: LLNL Linux cluster architecture

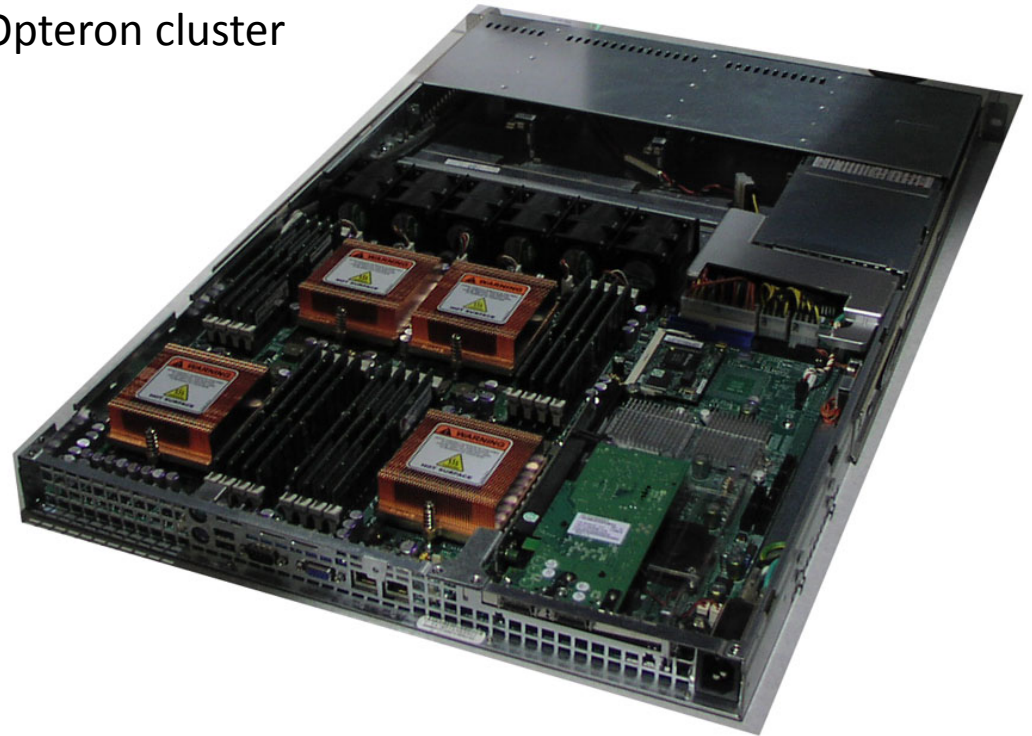


From https://computing.llnl.gov/tutorials/linux_clusters/

Nodes



Front view of compute nodes from LC Optron cluster



Quad-core, quad-socket Optron compute node

Frames/Racks



An SU consists of: Nodes (compute, login, management, gateway)

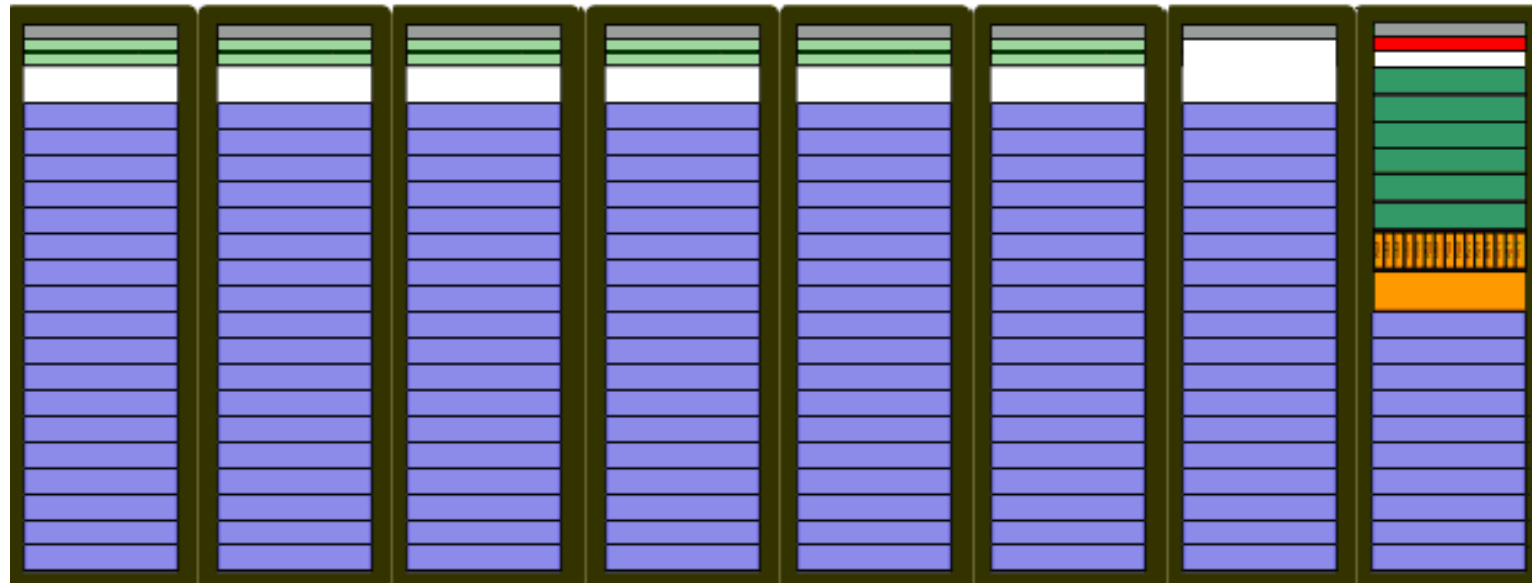
First stage switches that connect to each node directly




Miscellaneous management hardware


Frames sufficient to house all of the hardware

Additionally, a second stage switch is also needed for every 2 SUs in a multi-SU cluster (not shown).

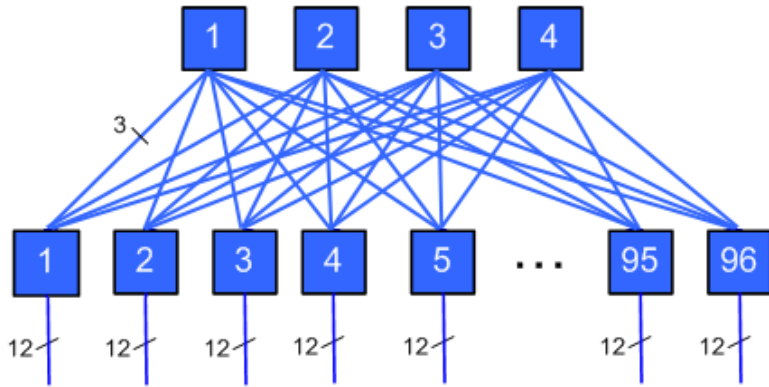
Scalable Unit (SU) - TLCC Clusters



 Compute node
 24-port IB 4x DDR switch
 Management hardware

 Management node
 Gateway node
 Login node

Interconnect overview



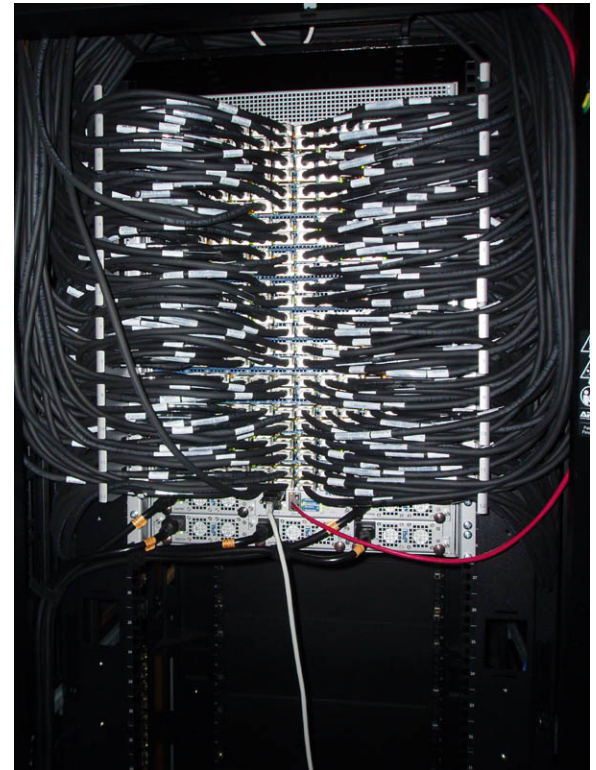
Two-stage interconnect, Atlas, Juno – 8SU



Adapter card:
Processing node's
link to interconnect



First stage switch: Voltaire 24-
port Switches and Nodes, Back



Second stage switch:
Voltaire 288-port Switch, back.
All used ports connect to
first stage switches .

Interconnection Network (I)

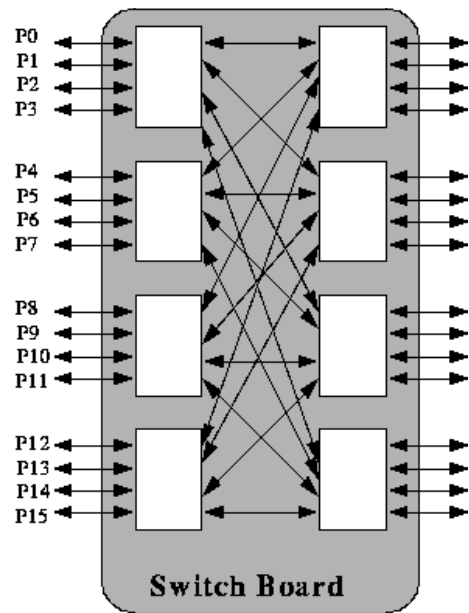


Figure 1. Switch Board Connections (16-way)

- **Dynamic network switch**

Degree of the switch = number of ports on a switch

Switch functions:

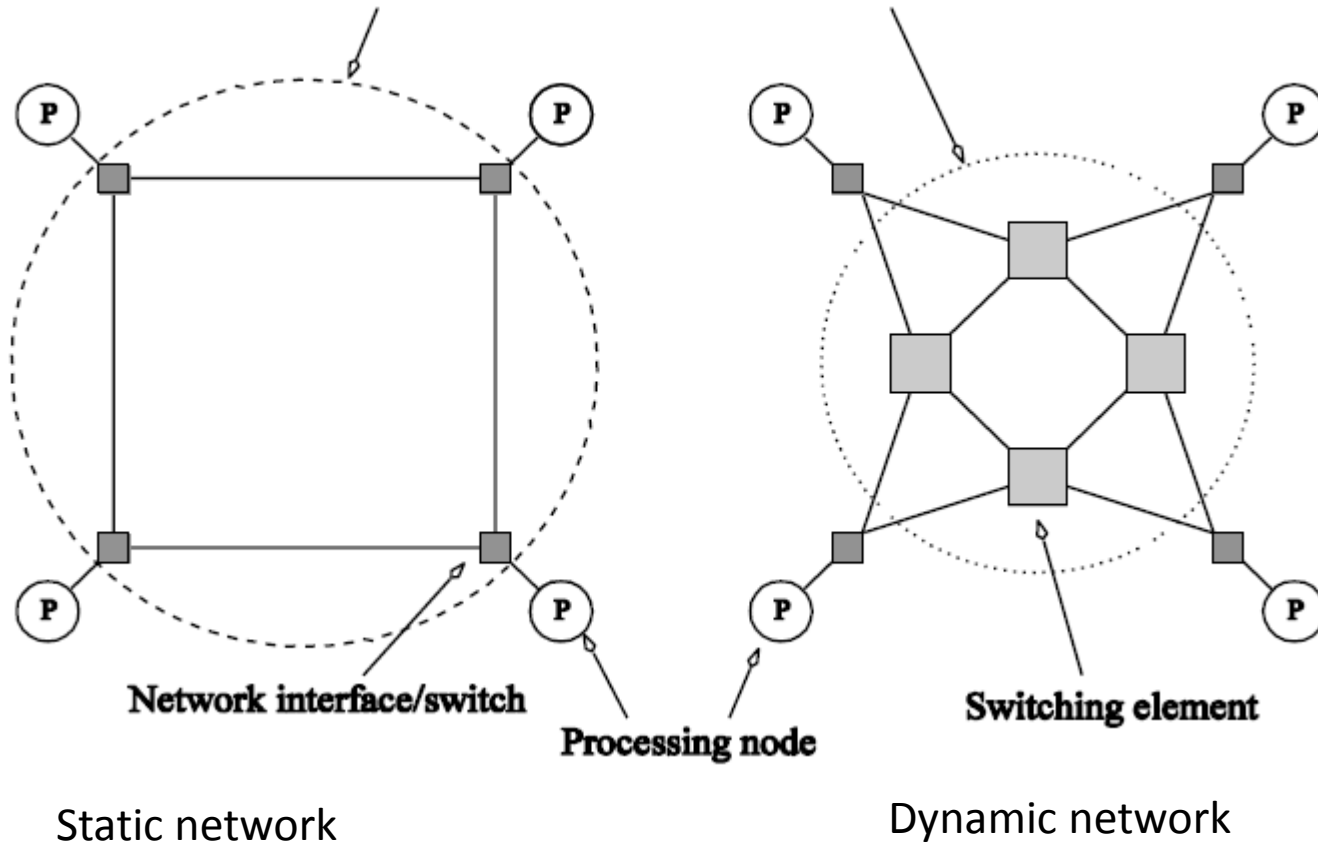
- mapping from input to output ports
- internal buffering (when the requested output port is busy)
- routing (to reduce network congestion)
- multicasting (same output on multiple ports)
- non-blocking: disjoint paths between each pair of independent inputs and outputs

Network interface

- Network interface is to handle the connectivity between the node and the network
- It has input and output ports that pipe data from and to the network
- Function:
 - packetizing data
 - computing routing information
 - buffering incoming and outgoing data
 - error checking

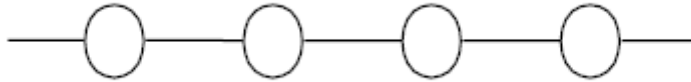
Interconnection Network (II)

- **Static network (direct network):** point-to-point communication links between computing nodes
- **Dynamic network (indirect network):** built using switches and communication links. Communication links are connected to one another dynamically by switches to establish paths.

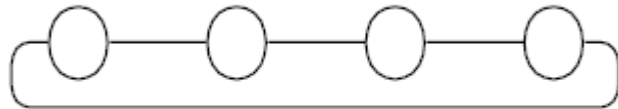


Interconnection Network (III)

- Linear array: each node has two neighbors

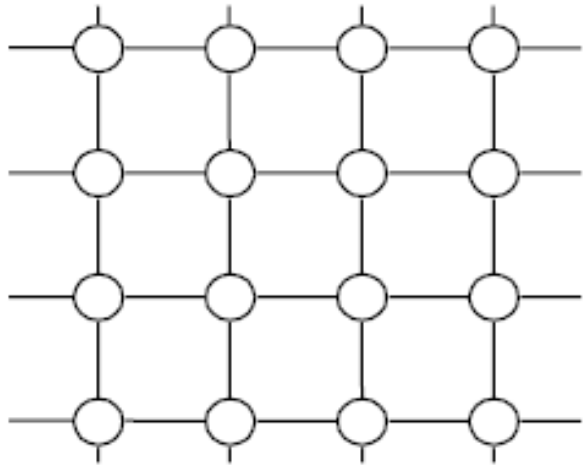


- 1D torus (ring)

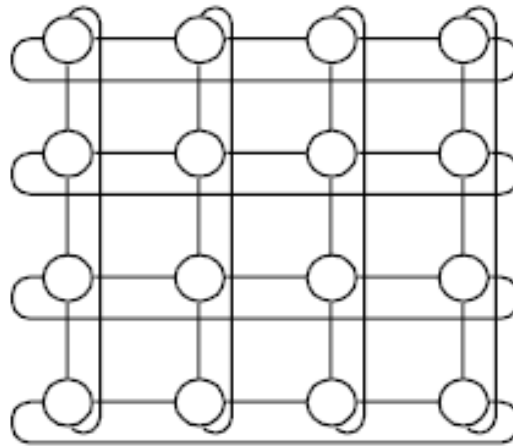


Interconnection Network (IV)

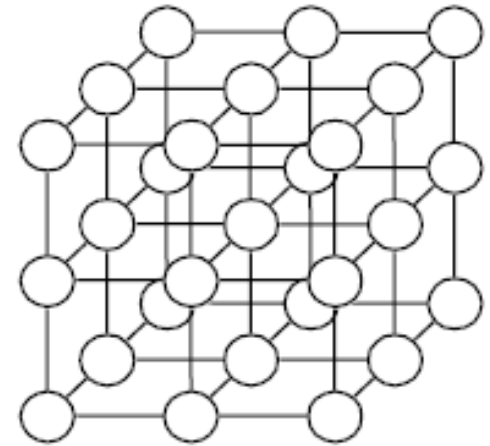
- K-dimensional mesh: nodes have $2k$ neighbors



2D mesh



2D mesh with wraparound link (2D torus)

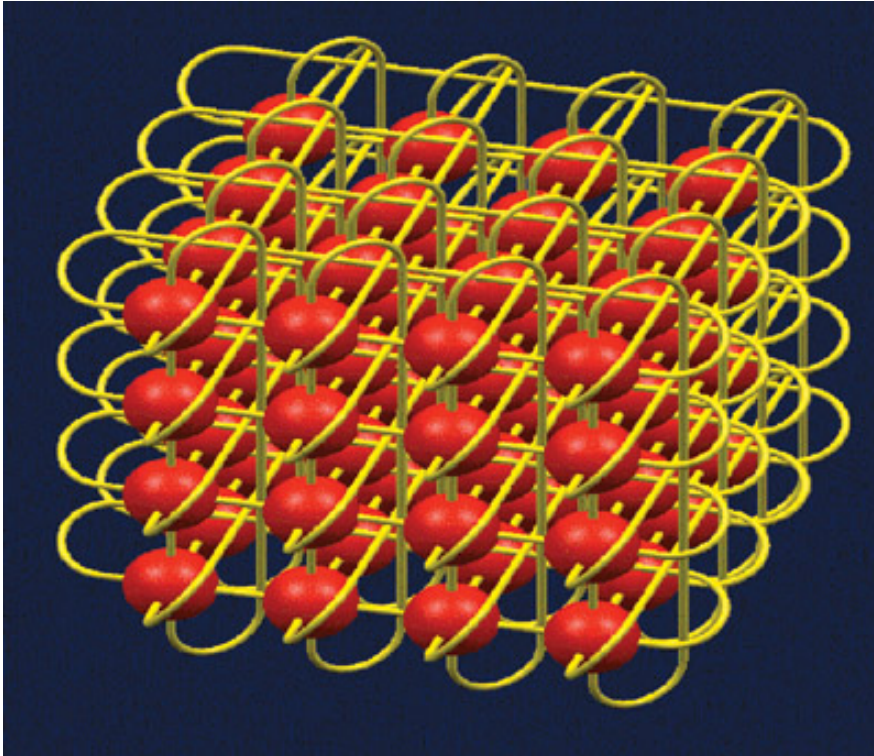


3D mesh

Cray T3E

- Use Alpha 21164A processor with 4-way superscalar architecture, 2 floating point instruction per cycle
- CPU clock 675 MHz, with peak rating 1.35 Gigaflops, 512 MB local memory
- Parallel systems with 40 to 2176 processors (with modules of 8 CPUs each)
- **3D torus interconnect** with a single processor per node
- Each node contains a router and has a processor interface and six full-duplex link (one for each direction of the cube)

- IBM BlueGene/L

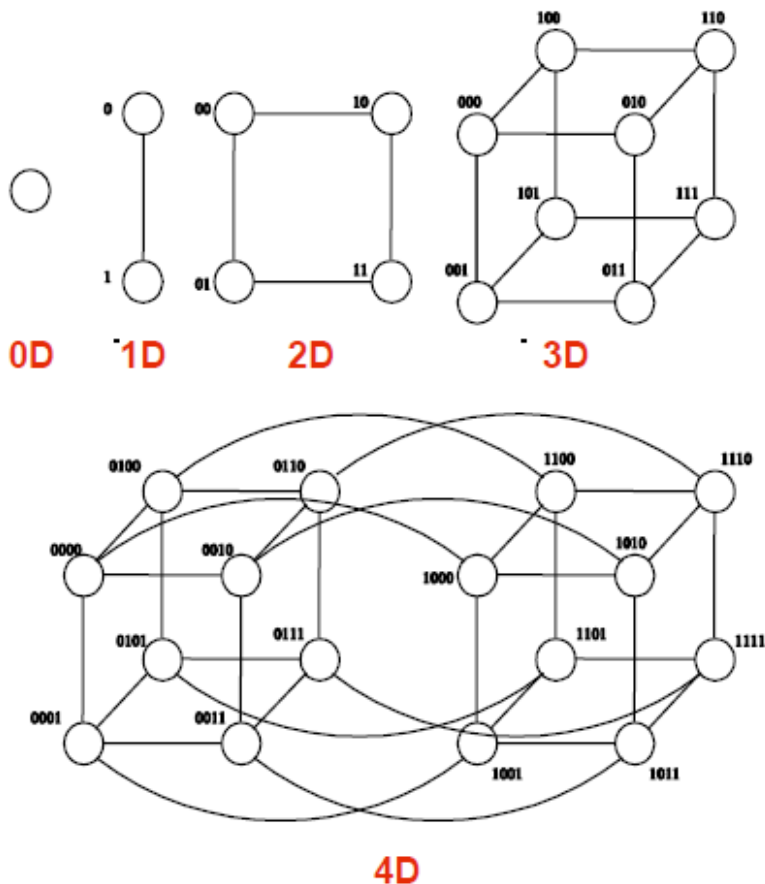


In 1999, IBM Research announced a 5-year \$100M project, named Blue Gene, to develop a petaflop computer for research in computational biology

- IBM BlueGene/L uses a three-dimensional (3D) torus network in which the nodes (red balls) are connected to their six nearest-neighbor nodes in a 3D mesh. In the torus configuration, the ends of the mesh loop back, thereby eliminating the problem of programming for a mesh with edges. Without these loops, the end nodes would not have six near neighbors.

Interconnection Network (V)

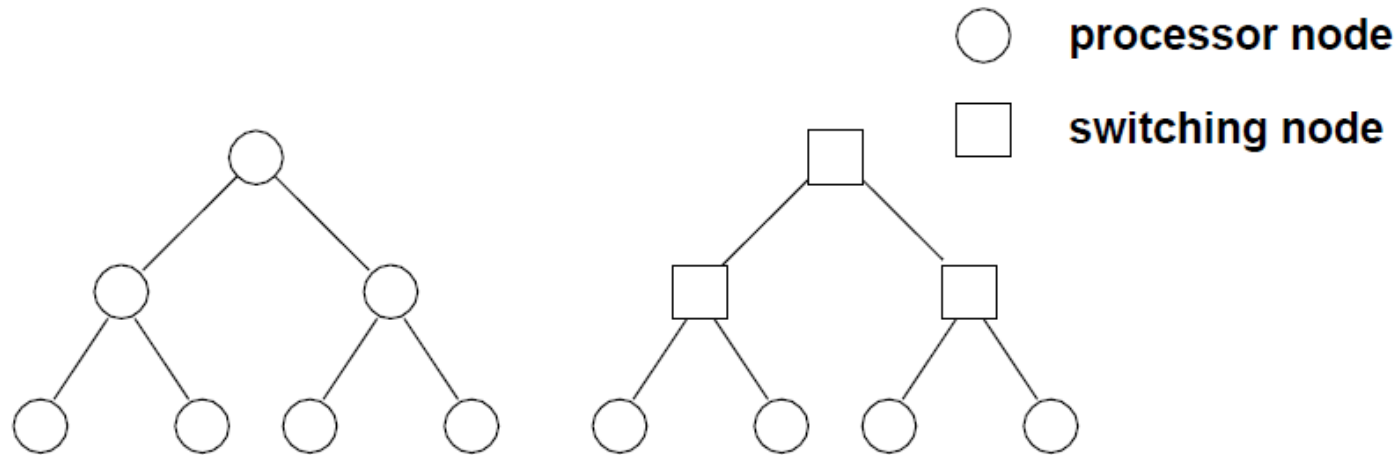
Hypercubes: the topology has two nodes along each dimension and $\log_2 p$ dimensions.



- D-dimensional cube is constructed by connecting corresponding nodes of two (D-1) dimensional cube
- D-dimensional cube: P nodes in total, $D = \log_2 P$
- Distance between any two nodes is at most $\log P$.

Interconnection Network (VI)

Binary Trees



static tree network

dynamic tree network

Distance between any two nodes is no more than $2 \log p$

- Problem

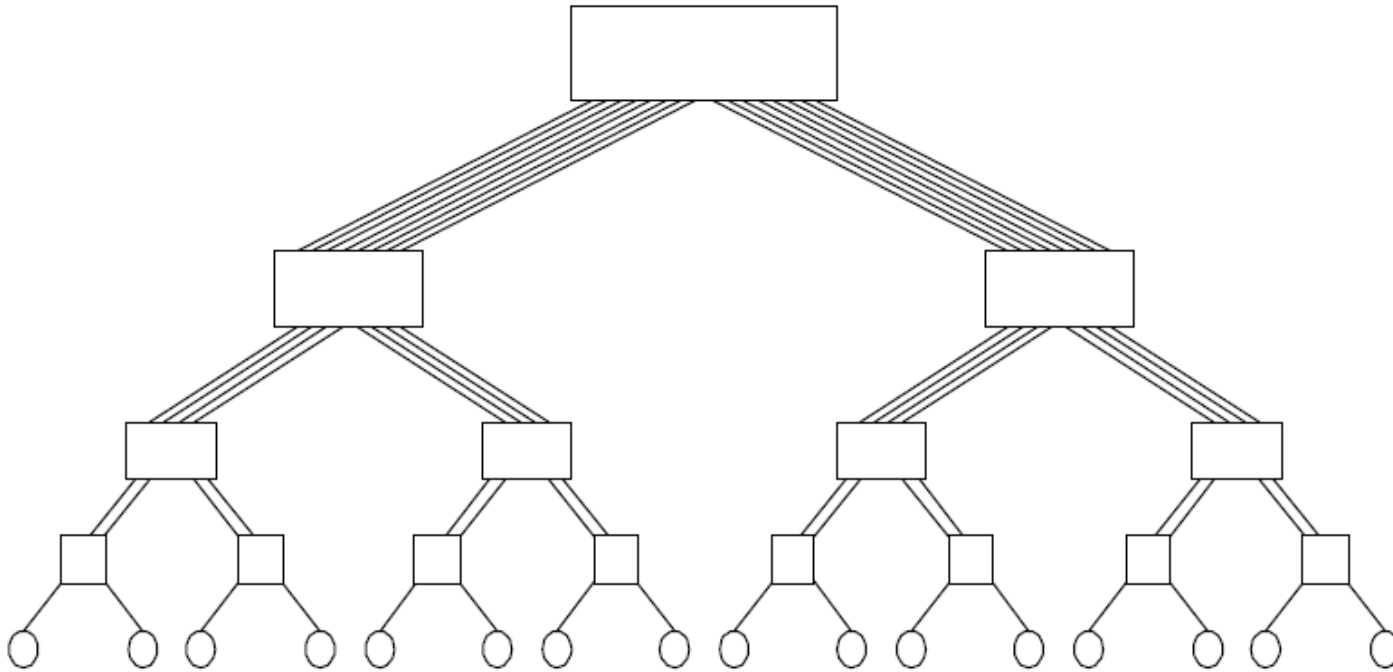
- Messages from one half tree to another half tree are routed through the top level nodes

- links closer to root carry more traffic than those at lower levels.

Interconnection Network (VII)

Fat Tree Network

- Increased the number of communication links and switching nodes closer to the root.
- The fat tree is suitable for dynamic networks



A fat tree network of 16 processing nodes

Interconnection Network (II)

- **Metrics for static network**

Diameter: longest distance between two nodes – Indication of maximum delay that a message will encounter in being communicated between a pair of nodes.

Connectivity: a measure of multiplicity of paths between any two nodes.

Arc connectivity: minimum number of arcs that must be removed from the network to break it into two disconnected networks.

Bisection width: minimum number of communication links that must be removed to partition the network into two equal halves – minimum volume of communication allowed between any two halves of the network.

Metrics for static network topology

Network	Diameter	Bisection Width	Arc Connect.	Number of Links
Fully conn-ted	1	$p^2 / 4$	$p-1$	$p(p-1)/2$
Star	2	1	1	$p-1$
Binary tree	$2\log((p+1)/2)$	1	1	$p-1$
Linear array	$p-1$	1	1	$p-1$
Ring	$ p-2 $	2	2	p
2D mesh	$2(\sqrt{p}-1)$	\sqrt{p}	2	$2(p-\sqrt{p})$
2D meshwrap	$2\lceil\sqrt{p}/2\rceil$	$2\sqrt{p}$	4	$2p$
Hypercube	$\log p$	$p/2$	$\log p$	$(p \log p)/2$

Cost Model for Communicating Messages

Time to communicate a message (t_{comm}) between two nodes in a network: sum of the time to prepare a message for transmission and the time taken by the message to traverse the network to its destination.

- **Startup time** (t_s): the time required to handle a message at sending and receiving nodes.
- **Pre-hop time** (t_h): the time taken by the header of a message to travel between two directly-connected nodes, which is also known as **node latency**. This time is directly related to the latency within the routing switch for determining which output buffer or channel the message should be forward to.
- **Per-word transfer time** (t_w): If the channel bandwidth is r words per second, then each word takes time $t_w = 1/r$ to traverse the link.

t_{comm} needs to send m words to traverse l communication link:

- **Store-and-forward switching:**

When a message traverses a path with multiple links, each intermediate node on the path forwards the message to the next node after it has received and stored the entire message.

$$t_{comm} = t_s + (t_h + t_w m)l$$

- **Cut-through switching:**

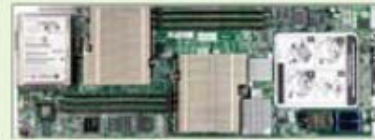
A message is broken into fixed size units call **flow control digits** of **flits**. After the connection from source to destination is established, the flits are sent one after the other.

$$t_{comm} = t_s + lt_h + t_w m$$

HP BladeSystem BL components



DDR/R
mezzanine HCA



HP BL2X220C
blade server
(two nodes
per blade)



HP IB switch blade

HP ProLiant DL/SL components

PCIe QDR HCA



HP ProLiant
DL server



HP ProLiant
SL server



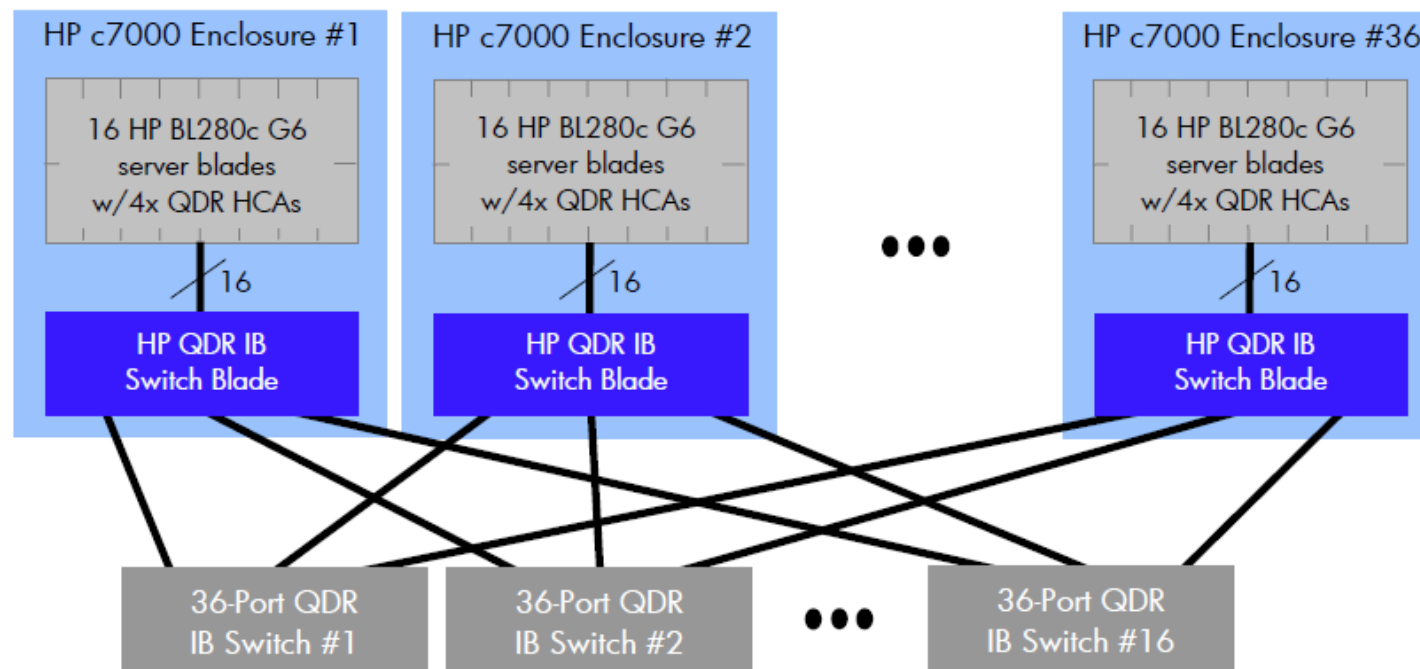
HP c7000
enclosure
with blades



HPC configuration with HP BladeSystem solutions

Figure 7 shows a full-bandwidth, fat-tree configuration of HP BladeSystem c-Class components providing 576 nodes in a cluster. Each c7000 enclosure includes an HP 4x QDR InfiniBand Switch Blade, with 16 downlinks for server blade connection and 16 QSFP uplinks for fabric connectivity. Sixteen 36-port QDR InfiniBand switches provide spine-level fabric connectivity.

Figure 7. HP BladeSystem c-Class 576-node cluster configuration using BL280c blades



Total nodes	576 (1 per blade)
Racks required for servers	Nine 42U (assumes four c7000 enclosures per rack)
Interconnect	1:1 full bandwidth (non-blocking), 3 switch hops maximum, fabric redundancy

Nersc Carver

- Carver, a liquid-cooled IBM iDataPlex system, has 1202 compute nodes (9,984 processor cores). This represents a theoretical peak performance of 106.5 Teraflops/sec.

Type of Node	Number	Cores/Node	Mem/Node	Mem/Core
Nehalem 2.67GHz "smallmem"	960	8	24GB 1333MHz	3 GB
Nehalem 2.67GHz "bigmem"	160	8	48GB 1066MHz	6 GB
Westmere 2.67GHz	80	12	48GB 1333MHz	4 GB
Nehalem-EX 2.00GHz	2	32	1TB 1066MHz	32 GB

Interconnect

All Carver nodes are interconnected by 4X QDR InfiniBand technology, meaning that 32 Gb/sec of point-to-point bandwidth is available for high-performance message passing and I/O. The interconnect consists of fiber optic cables arranged as local fat-trees within a global 2D mesh.

Additional Reference

- Using InfiniBand for a scalable compute infrastructure. Technology brief, 4th edition
- https://computing.llnl.gov/tutorials/linux_clusters/
- <http://www.nersc.gov/>
- A.S. Tanenbaum, Structured Computer Organization