Parallel and Distributed Computing
CS3006 (BDS-6A)
Lecture 04

Instructor: Dr. Syed Mohammad Irteza

Assistant Professor, Department of Computer Science, FAST

09 February, 2023

Previous Lecture

- Multi-computers (symmetric, asymmetric)
- Cluster vs. network of workstations
- Parallel vs. distributed computing
- Flynn's Taxonomy:
  - SISD
  - SIMD
  - MISD
  - MIMD

Reading Assignment

- Cache Coherence and Snooping
- Branch prediction and issues while pipelining the problem

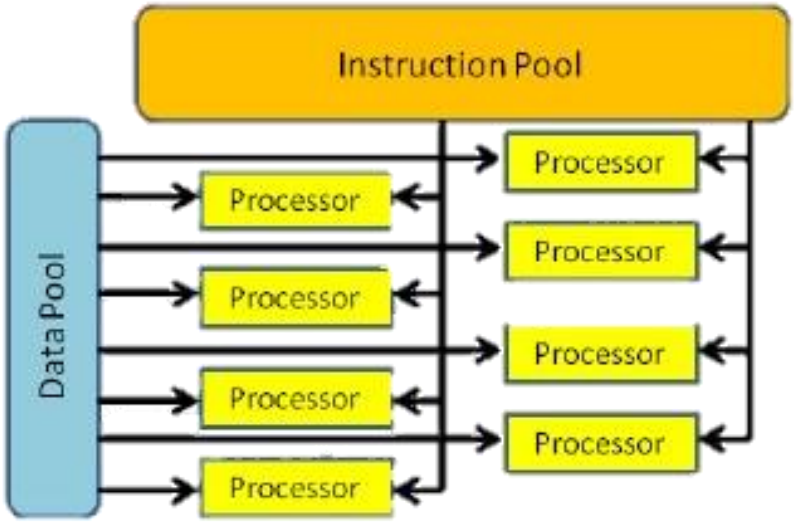
MISD

- A single data stream is transmitted to a set of processors, each of which executes a different instruction sequence
- Each processing unit operates on the data independently via independent instruction stream
- This structure is not commercially implemented
- An example of use could be multiple cryptography algorithms attempting to crack a coded message

Flynn's Taxonomy

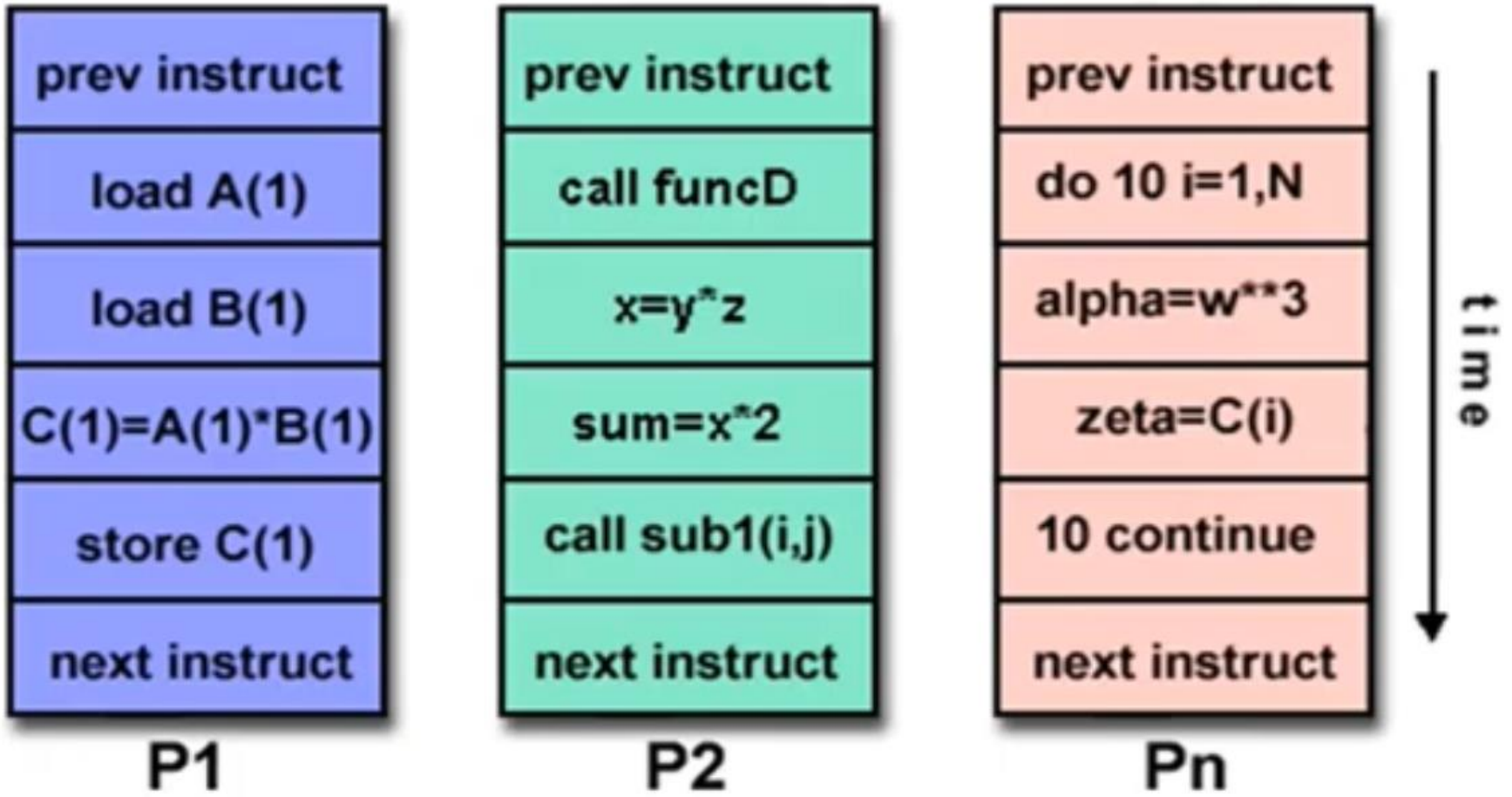
MIMD (Multiple Instructions Multiple Data)

- Multiple instruction streams and multiple data streams
- Different CPUs can simultaneously execute different instruction streams manipulating different data
- Most of the modern parallel architectures fall under this category e.g., **Multiprocessor** and **multicomputer** architectures
- Many MIMD architectures include SIMD executions by default.



e.g. super computers

Example of MIMD:



MIMD

- Multiple instruction: Every processor may execute a different instruction stream
- Multiple data: Every processor may work with a different data stream

- Examples
  - most of the current supercomputers
  - Grids
  - networked parallel computers
  - multiprocessor SMP computer

MIMD (continued)

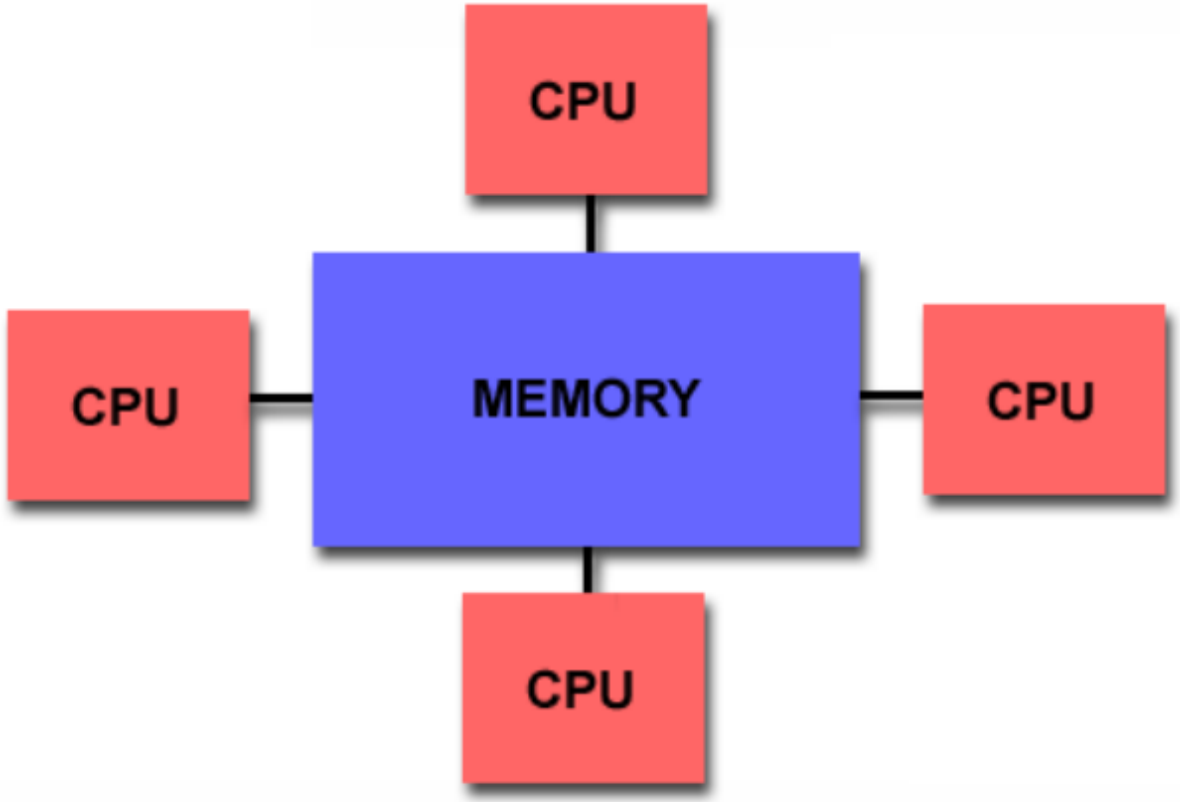
- MIMD systems are mainly:
  - Shared Memory (SM) systems:

    multiple CPUs all of which share the same address space (there is only one memory).
  - Distributed memory (DM) systems:

    each CPU has its own associated memory. CPUs are connected by some network (clusters).

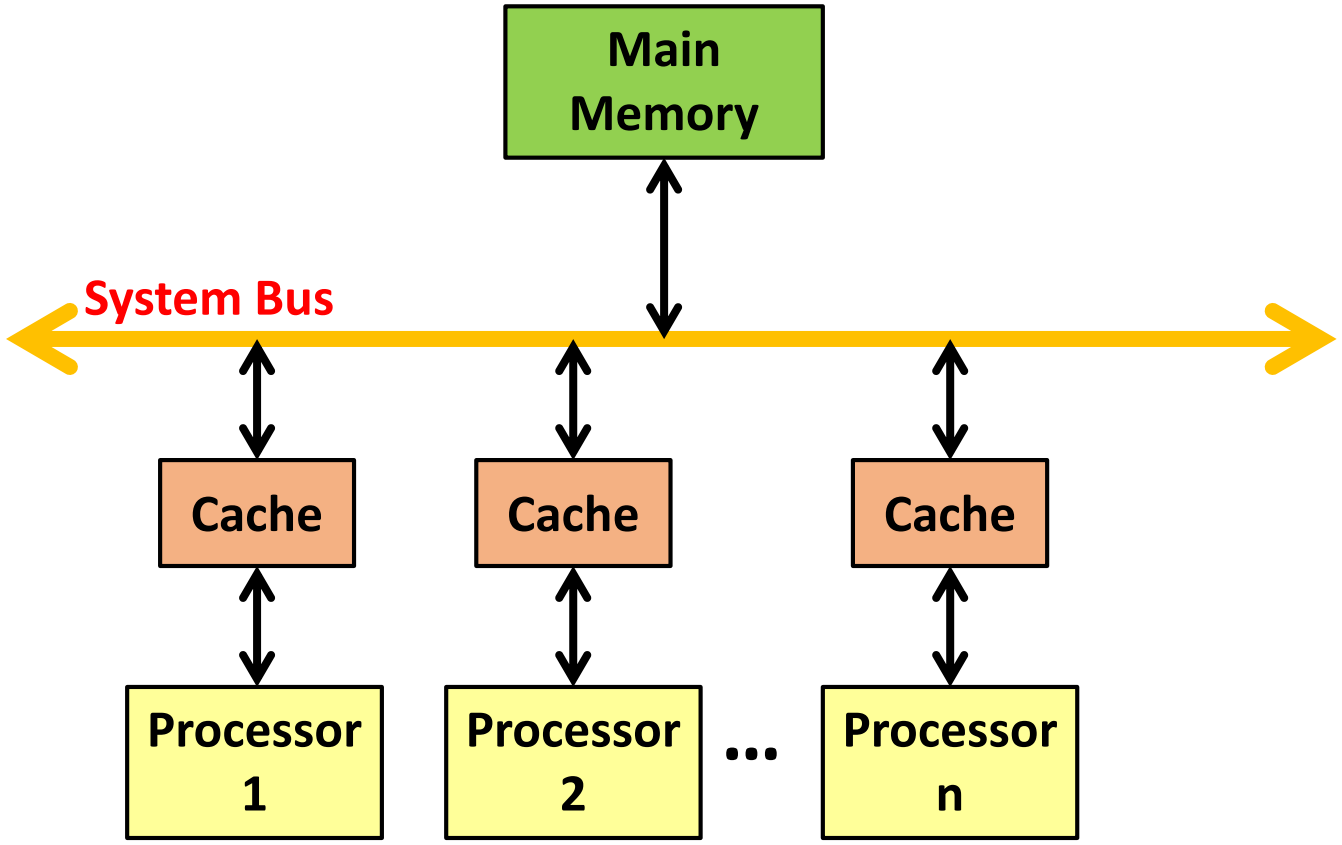
MIMD - Shared Memory

- All processors have access to all memory as a global address space

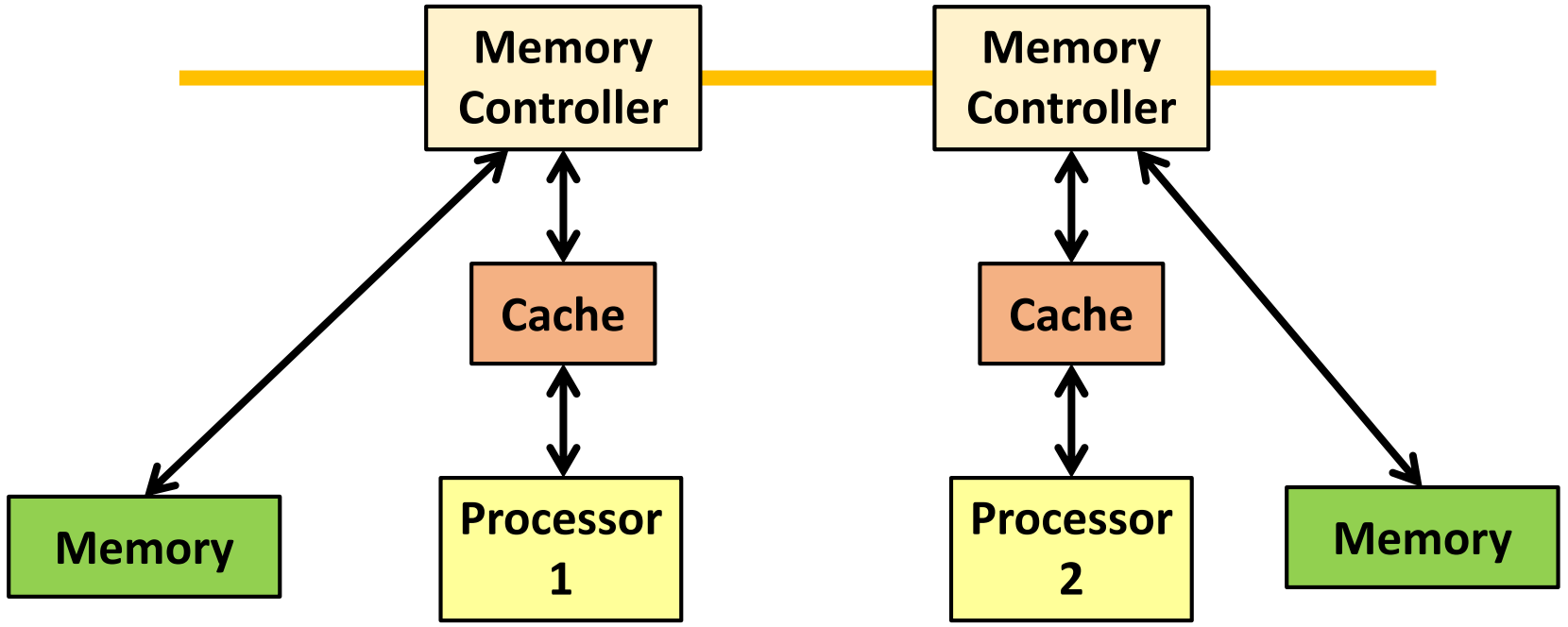


Uniform Memory Access (UMA)

- Mostly represented by Symmetric Multiprocessor (SMP) machines



Non-Uniform Memory Access (NUMA)

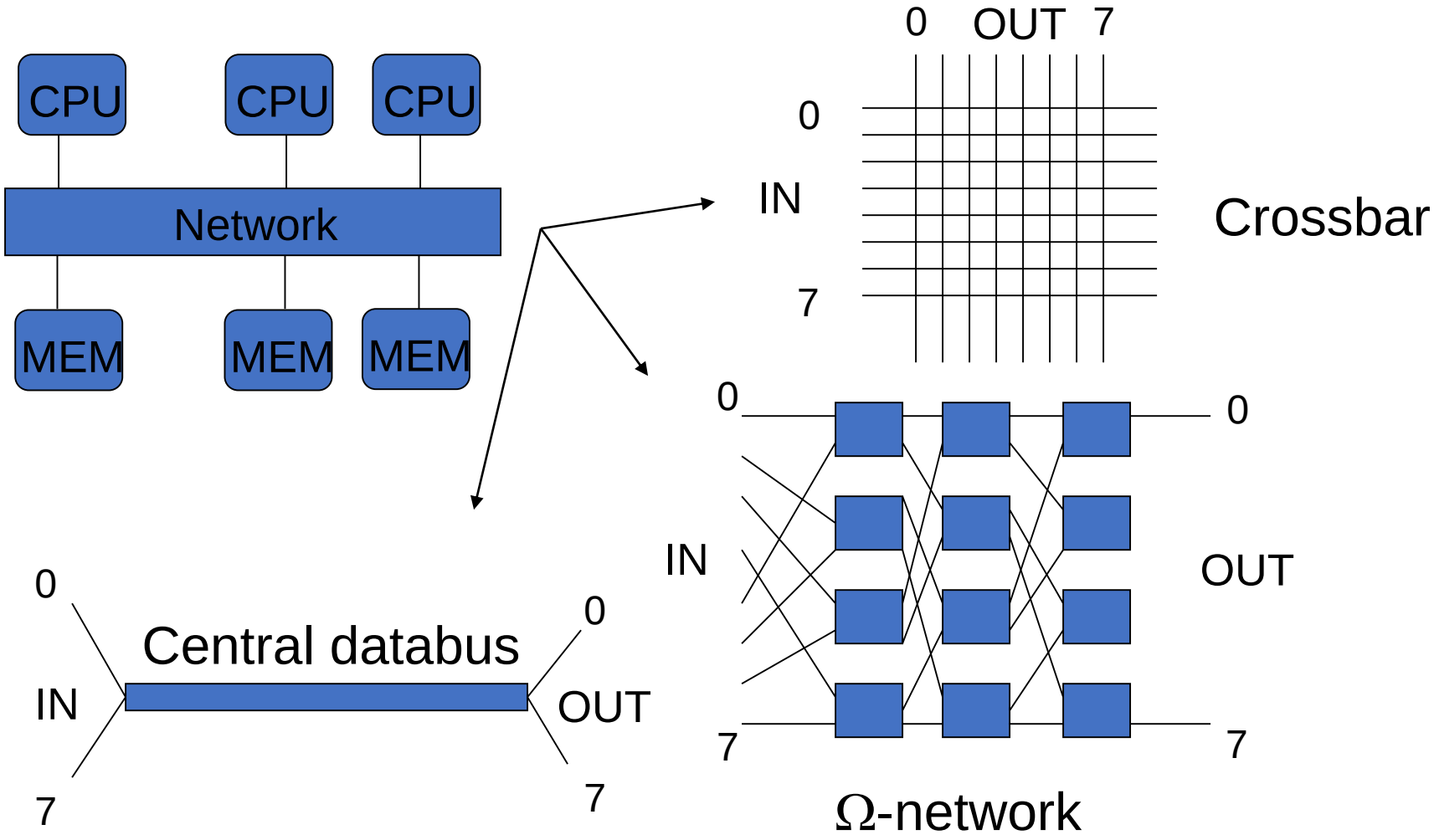


Shared Memory Interconnection Network

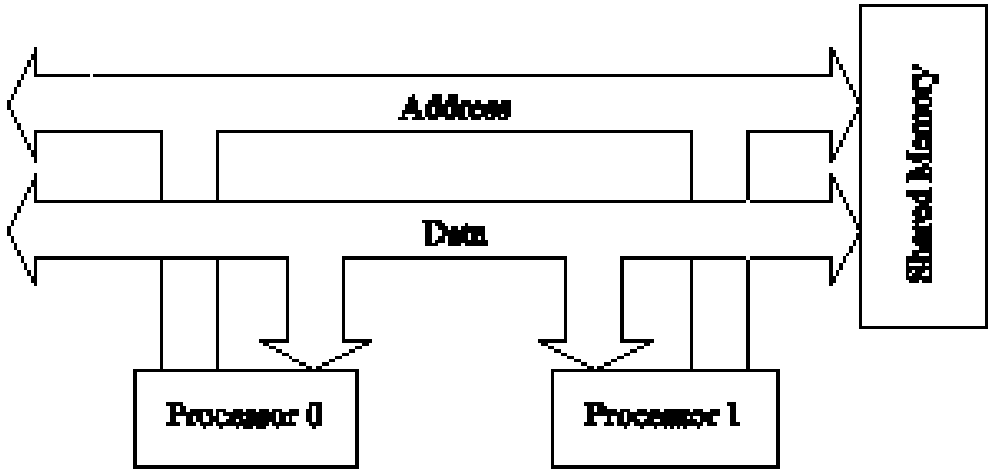
- The main problem → how to do interconnections of the CPUs, with each other and to the memory

- There are three main network topologies available:
  - Crossbar (n^2 connections - datapath without sharing)
  - Ω-network ($n \log_2 n$ connections - $\log_2 n$ switching stages and shared on a path)
  - Central databus (1 connections - n shared)

Shared Memory Interconnection (2)

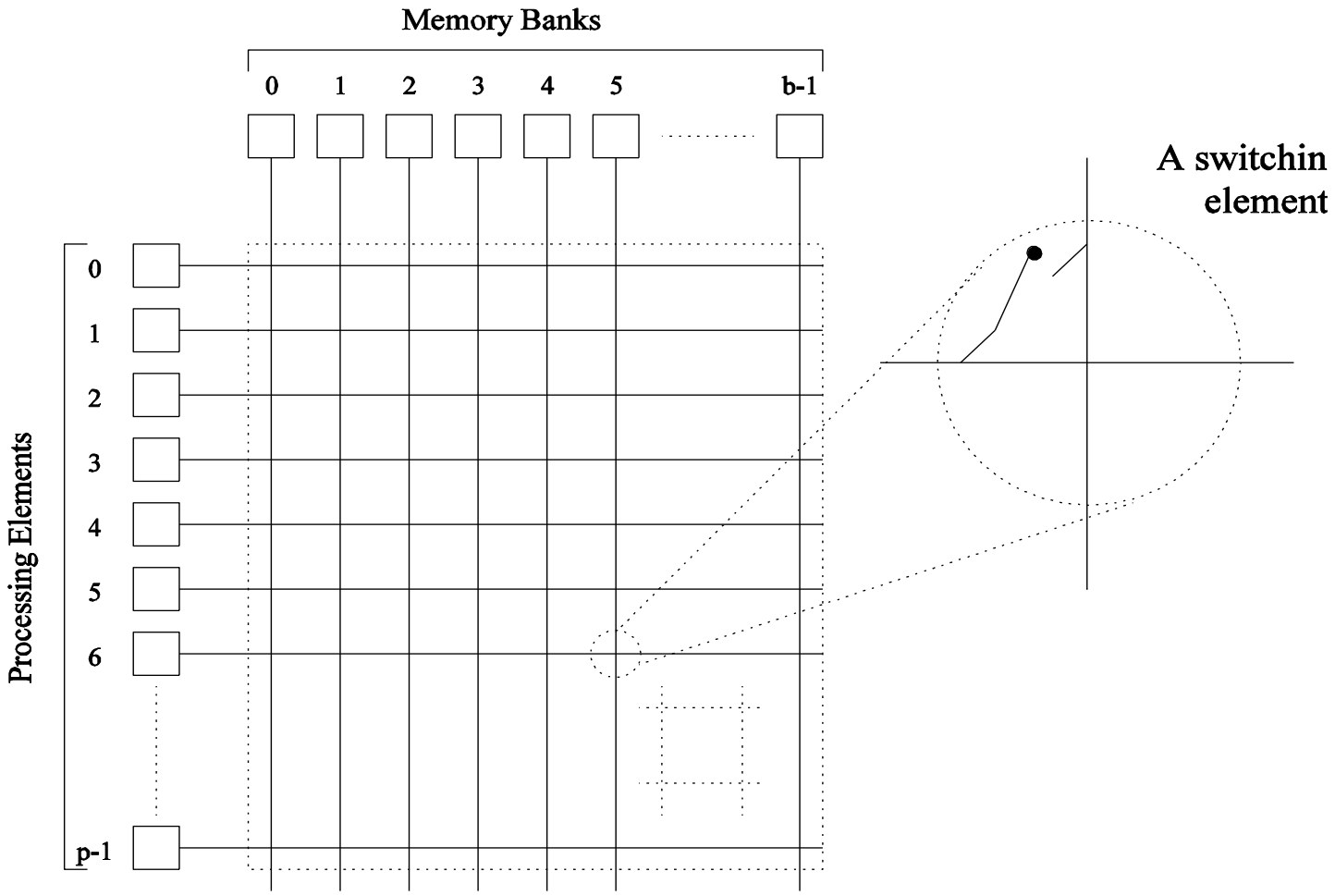


Network Topologies: Buses



- Simplest and earliest parallel machines used *buses*
- All processors access a *common bus* for exchanging data
- The distance between any two nodes is *O(1)* in a bus. The bus also provides a convenient broadcast media.
- The bandwidth of the shared bus is a *major bottleneck*
- Typical bus based machines are limited to dozens of nodes. Sun Enterprise servers and Intel Pentium based shared-bus multiprocessors are examples of such architectures.

Network Topologies: Crossbar



Network Topologies: Crossbar

- Uses an $p \times m$ grid of switches to connect p inputs to m outputs in a *non-blocking manner*
- The cost of a crossbar of p processors grows as $O(p^2)$
- This is generally difficult to scale for *large values of p*
- Examples of machines that employ crossbars include the Sun Ultra HPC 10000 and the Fujitsu VPP500

- Crossbars have excellent performance scalability but poor cost scalability
- Buses have excellent cost scalability, but poor performance scalability

Network Topologies: Multistage Omega Network

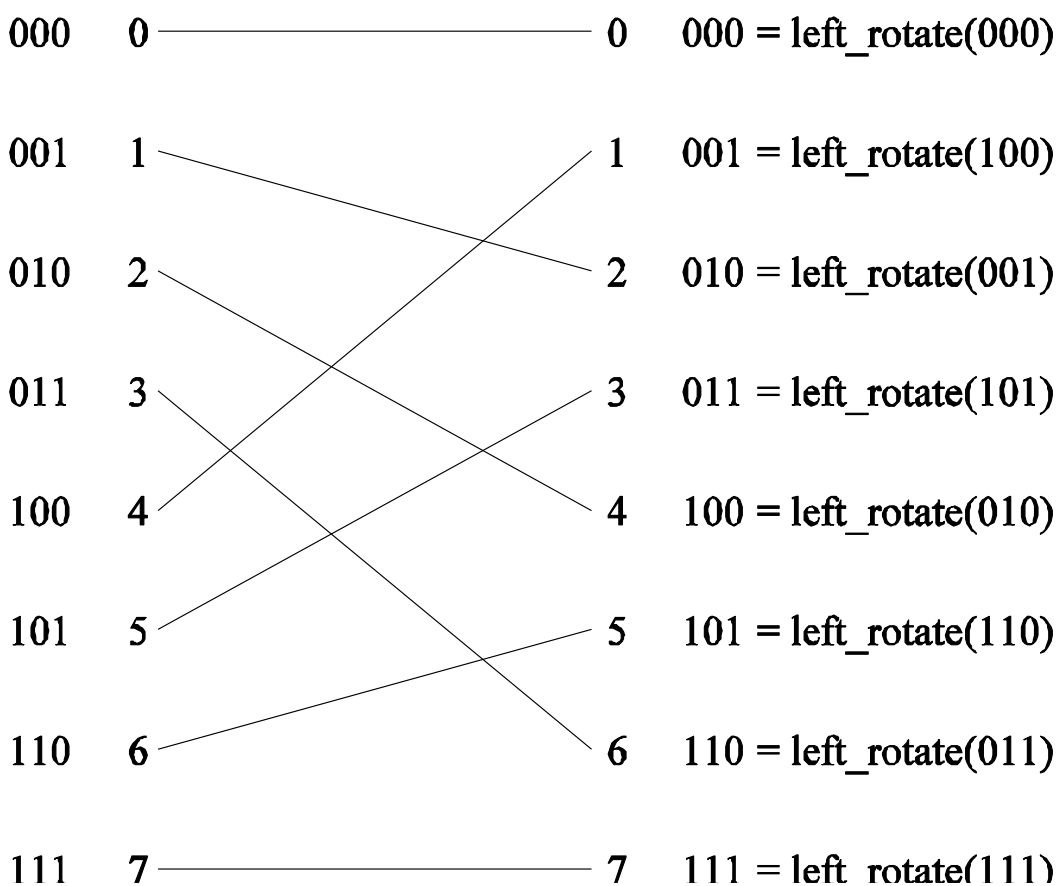
- One of the most commonly used multistage interconnects is the Omega network.
- This network consists of *log p* stages, where *p* is the number of inputs/outputs.

- At each stage, input *i* is connected j

$$j = \begin{cases} 2i, & 0 \leq i \leq p/2 - 1 \\ 2i + 1 - p, & p/2 \leq i \leq p - 1 \end{cases}$$

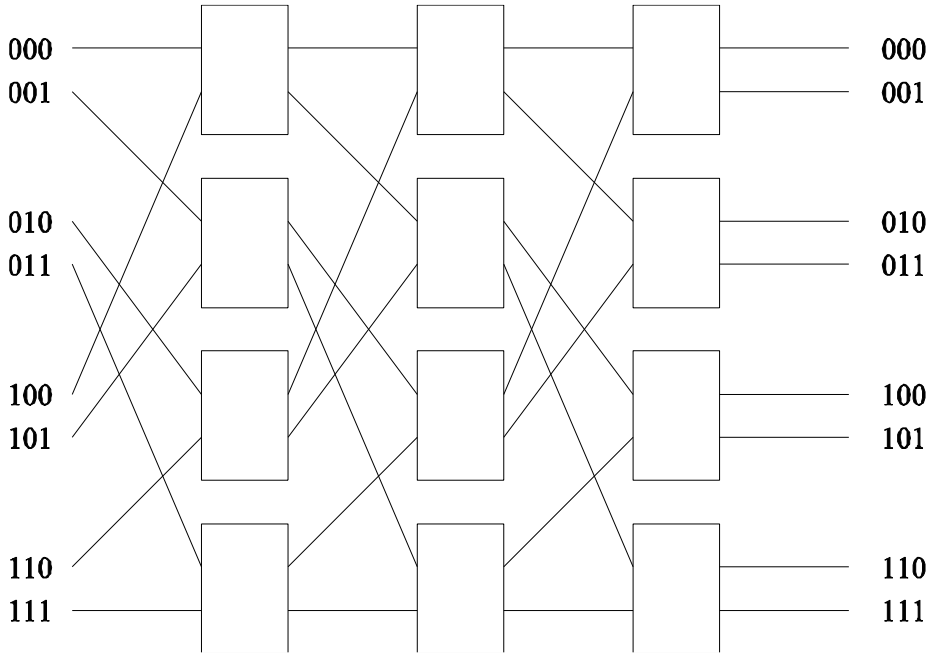
Network Topologies: Multistage Omega Network

- Each stage of the Omega network implements a perfect shuffle as follows:



8 x 8 Omega network

- A complete Omega network with the perfect shuffle interconnects and switches can now be illustrated:



- An omega network has $p/2 \times \log p$ switching nodes, and the cost of such a network grows as $(p \log p)$

Network Topologies: Multistage Omega Network

- Machines like IBM eServer p575 and SGI Altix 4000 use Ω-network
- Ω-network is much more interesting for a large number of processors
- Problem: the switches have to be fast enough, and also the width of the links is important. 16 bit parallel is better than serial links
- Multiprocessor vector-processors use crossbars instead (because at most only 32 processors)
- Synchronization is usually performed with special communication registers (CPU to CPU); if there is little synchronization shared memory is admitted

Advantages of shared memory machines

- User-friendly programming environment due to global address space
- Data sharing is fast and uniform due to proximity of memory to CPUs
- Memory coherence is managed by the operating system

Disadvantages of shared memory machines

- Performance degradation due to “memory contention” (several processors try to access the same memory location)
- Programmer’s responsibility for synchronization constructs (correct access to memory)
- Expensive to design shared memory computers with increasing numbers of processors
- Adding processors can geometrically increase traffic on the shared memory-CPU path and for cache coherence management

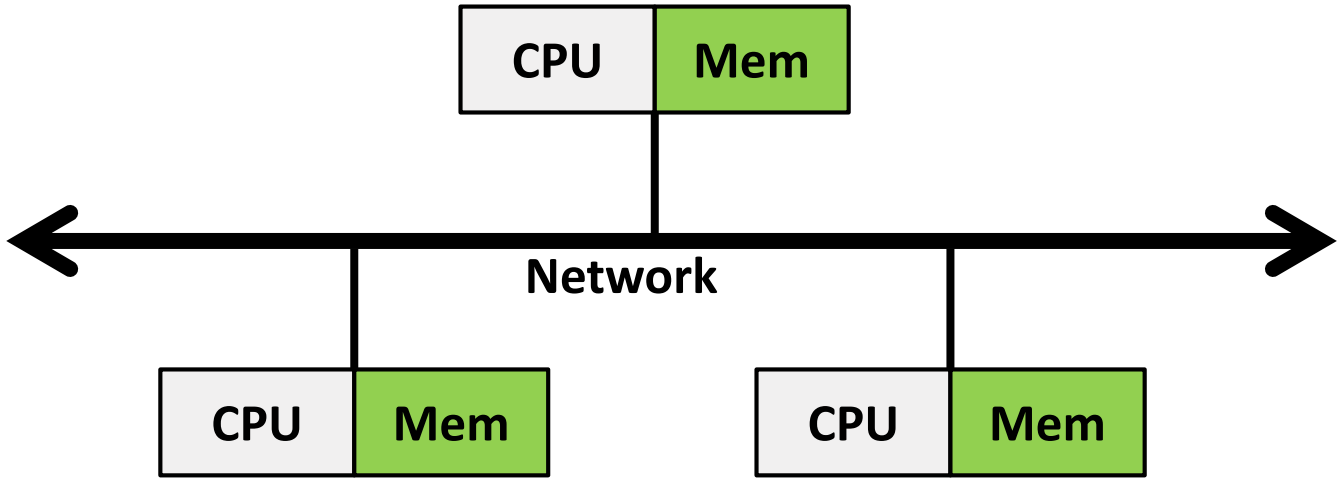
Distributed-memory SIMD (1)

- These machines are sometimes also known as processor-array machines
- They work in lock-step, i.e. all processors execute the same instruction at the same time (but on different data items); no synchronization is required
- A control processor issues the instructions that are to be executed by the processors in the processor array
- Processors sometimes are of a very simple bit-serial type (i.e., the processors operate on the data items bitwise, irrespective of their type), which can operate on operands of any length (when operands are little, this may result in speedups)

Distributed-memory SIMD (2)

- Graphical Processing Units (GPUs) are similar to processor array systems
- DM-SIMD are specialized on digital signal and image processing and on certain types of Monte Carlo simulations with virtually no exchange between processors
- Operations that cannot be executed by the processor array or by the control processor are off-loaded to the front-end system
- I/O may be through:
  - the front end system
  - the processor array (it is more efficient in providing data directly to the memory of the processor array)
  - both

Distributed-memory MIMD



- Processors have their own local memory
- No concept of global address space across all processors
- Distributed memory systems require a communication network to connect inter-processor memory

Examples

Shared-memory SIMD systems

- The Hitachi S3600 series

Distributed-memory SIMD systems

- The Alenia Quadrics
- The Cambridge Parallel Processing Gamma II
- The Digital Equipment Corp. MPP series
- The MasPar MP-1
- The MasPar MP-2

Shared-memory MIMD systems

- The Cray Research Inc. Cray J90-series, T90 series
- The Hitachi S3800 series
- The HP/Convex C4 series
- The Digital Equipment Corp. AlphaServer
- The NEC SX-4
- The Silicon Graphics Power Challenge
- The Tera MTA

Distributed-memory MIMD systems

- The Alex AVX 2
- The Avalon A12
- The C-DAC PARAM 9000/SS
- The Cray Research Inc. T3E
- The Fujitsu AP1000
- The Fujitsu VPP300 series
- The Hitachi SR2201 series
- The HP/Convex Exemplar SPP-1200
- The IBM 9076 SP2
- The Intel Paragon XP
- The Matsushita ADENART
- The Meiko Computing Surface 2
- The nCUBE 3
- The NEC Cenju-3
- The Parallel Computing Industries system
- The Parsys TA9000
- The Parsytec GC/Power Plus

DM-MIMD Routing Mechanism

- Routing mechanism determines the path a message takes through network to reach from source to destination node.
- Data and task decomposition have to be dealt with explicitly!
- The topology and speed of the data paths are crucial and have to be balanced with costs.
- Routing can be classified as:
  - Minimal
  - Non-minimal
- It can also be classified as:
  - Deterministic routing
  - Adaptive routing

Sources

- Slides of Dr. Haroon Mahmood & Dr. Rana Asif Rahman, FAST