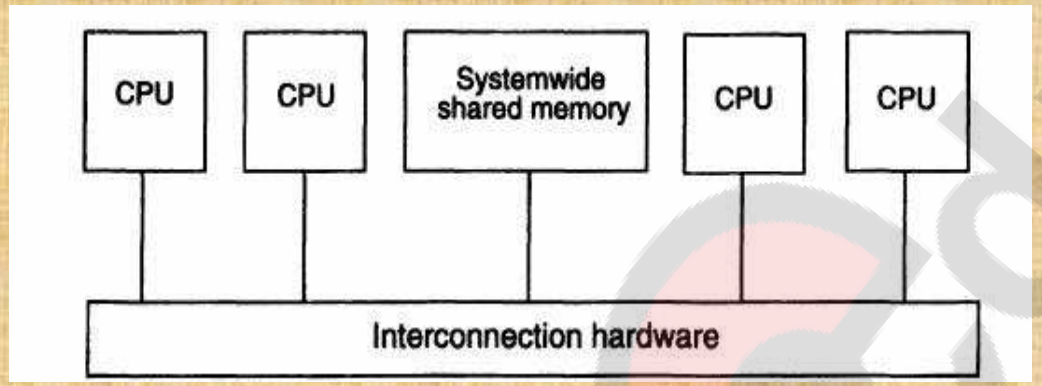
Introduction to Parallel Computing

duclash
Just Another Way To Learn

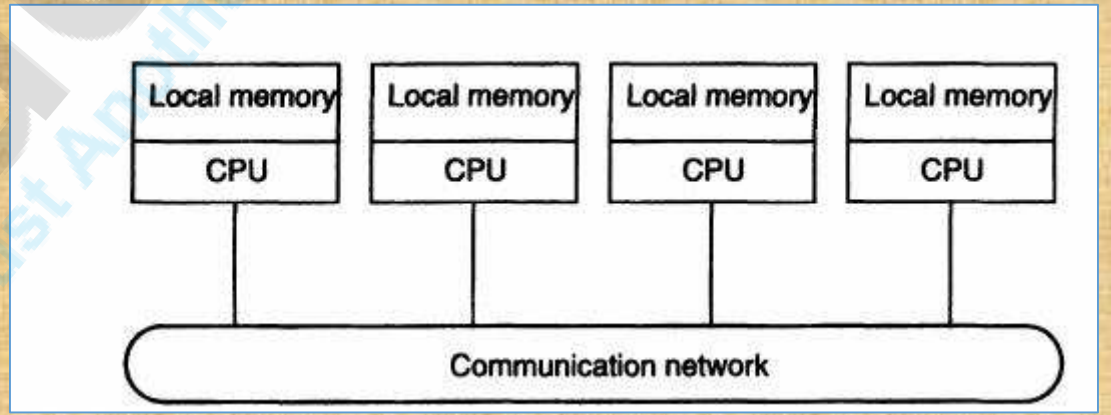
Parallel vs Distributed System

Computer Architecture

Tightly Coupled → **Parallel Processing System**



Loosely Coupled → Distributed Computing System

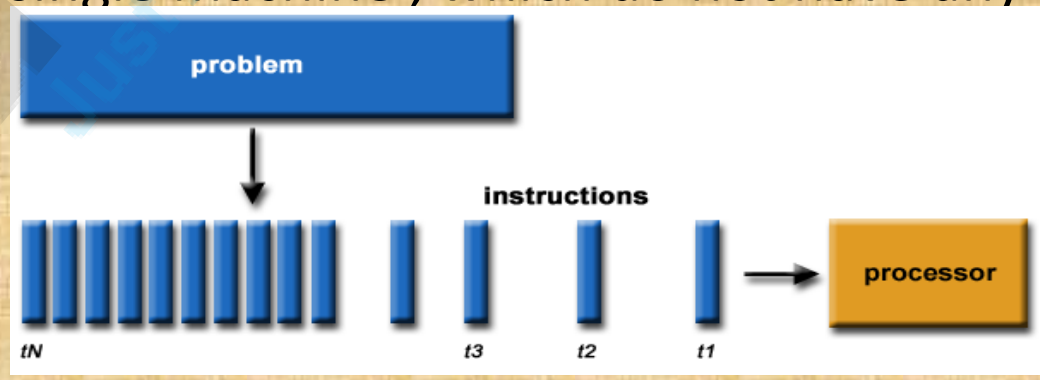


Parallel v.s. Distributed Systems

	Parallel Systems	**Distributed Systems**
Memory	**Tightly coupled shared** memory UMA, NUMA	**Distributed** memory Message passing, RPC, and/or used of distributed shared memory
Control	**Global clock** control SIMD, MIMD	**No global clock** control Synchronization algorithms needed
Processor interconnection	Order of **Tbps** Bus, mesh, tree, mesh of tree, and hypercube (-related) network	Order of **Gbps** Ethernet(bus), token ring and SCI (ring), myrinet(switching network)
Main focus	**Performance** Scientific computing	Performance(**cost and scalability**) **Reliability/availability** **Information/resource sharing**

Serial Computing

- Two approaches of Computing →Serial and Parallel
- Serial Computing
  - Given problem is divided in to discrete series of instruction , and these instruction executed sequentially in a single processor
  - Used in monolithic applications on single machine , which do not have any time constraint

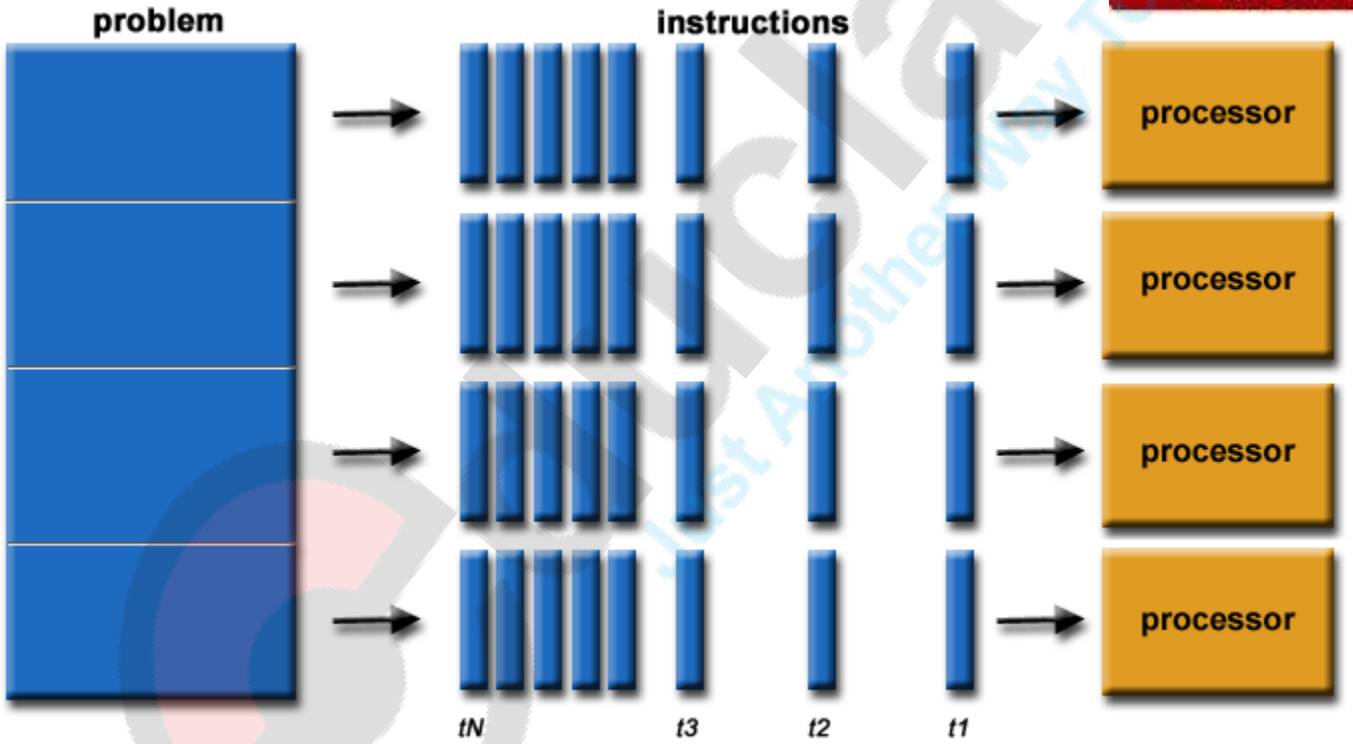


Serial Computing-Benefits and Limitations

- Advantages
  - Faster execution of smaller tasks
  - Ease of implementation
  - Suited for Monolithic application
- Disadvantages
  - Significant constraints to build faster computers
  - Hardware limitation on transmission speed
  - Expensive
  - Consumes too much power

Just Another Way To Learn

Parallel Computing



Parallel Architecture

Educlash
Just Another Way To Learn

Parallel Architecture

Ref:Parallel and Distributed systems Kulkarni,Giri,Joshi,Jadhav Chapter 1.3

- Architecture of parallel computers is intended to provide high speed for computation of complex task using different mechanisms
- Basic Parallel Architecture Components
  - Processors
  - Memory → Shared or Distributed?
  - Communication→ synchronous or Asynchronous ?
  - Control → Centralized or Distributed?

Educlash
Just Another Way To Learn

Parallel Architecture

Ref:Parallel and Distributed systems Kulkarni,Giri,Joshi,Jadhav Chapter 1.3

- Parallel Architecture classified under following categories.
  1. Specific Type of Parallel Architecture
  2. Classification based on architectural Schemes
  3. Classification based on memory access
  4. Classification based on interconnection among processing elements and memory modules

Educlash
Just Another Way To Learn

1. Specific Type of Parallel Architecture

a) Pipeline Computers
b) Array Processors
c) Multiprocessors
d) Systolic Architecture
e) Dataflow Architecture

Educlash
Just Another Way To Learn

1.a) Pipeline Computers

- Process of one instruction execution in digital computer have four stages
  1. Instruction Fetch
  2. Instruction Decoding
  3. Operand Fetch
  4. Execution

Pipelined computers will execute successive instruction in overlapped fashion i.e. new inputs are accepted at one end before previously accepted inputs appear as out put on the other side

Reduces the idle time of Hardware

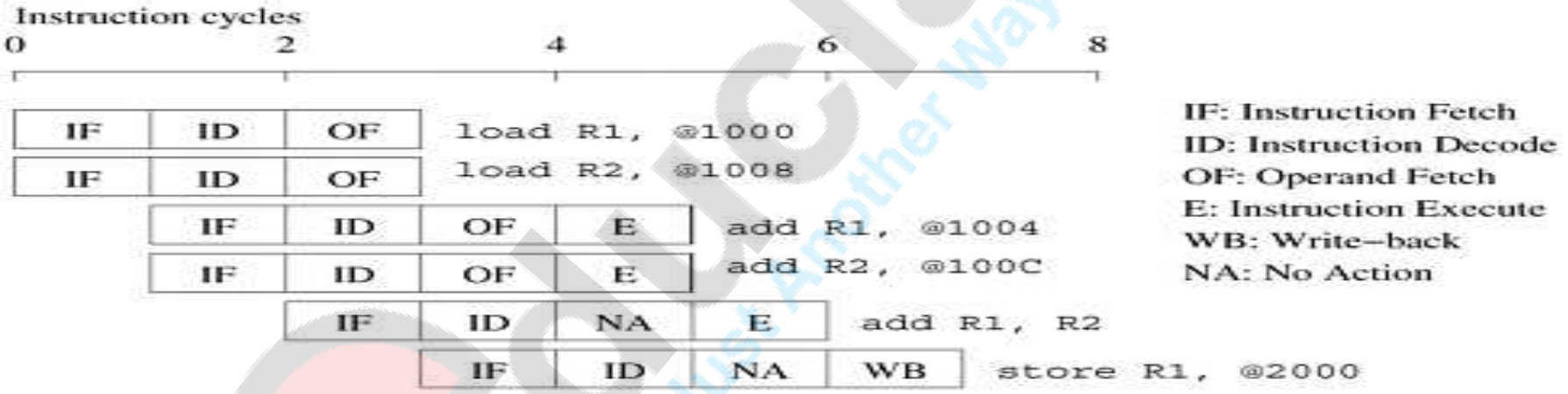
Educlash
Just Another Way To Learn

Multiple Instruction @same cycle-Superscalar Execution

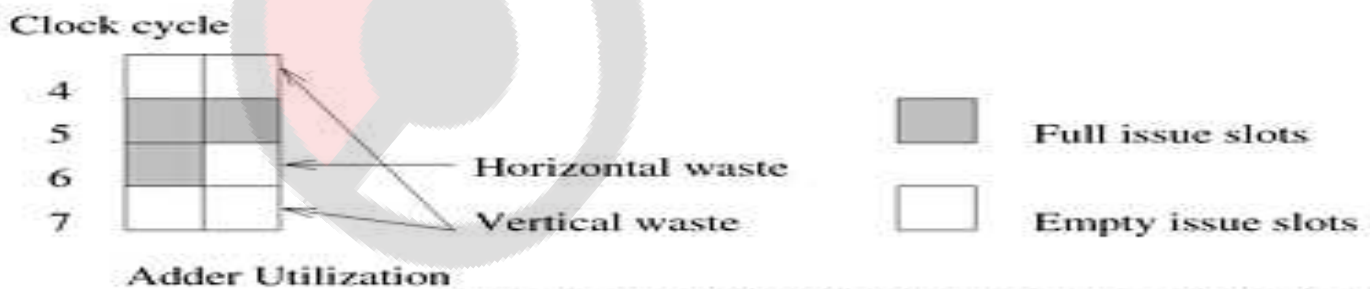
```
1. load R1, @1000        1. load R1, @1000        1. load R1, @1000
2. load R2, @1008        2. add R1, @1004         2. add R1, @1004
3. add R1, @1004         3. add R1, @1008         3. load R2, @1008
4. add R2, @100C         4. add R1, @100C         4. add R2, @100C
5. add R1, R2            5. store R1, @2000       5. add R1, R2
6. store R1, @2000                                6. store R1, @2000
```

(i) (ii) (iii)

(a) Three different code fragments for adding a list of four numbers.



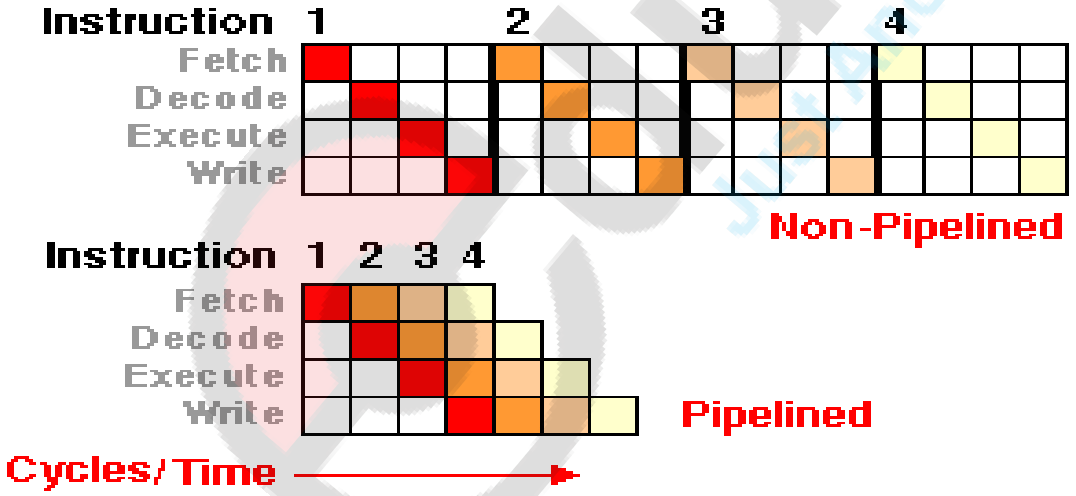
(b) Execution schedule for code fragment (i) above.



(c) Hardware utilization trace for schedule in (b).

Pipeline Computer

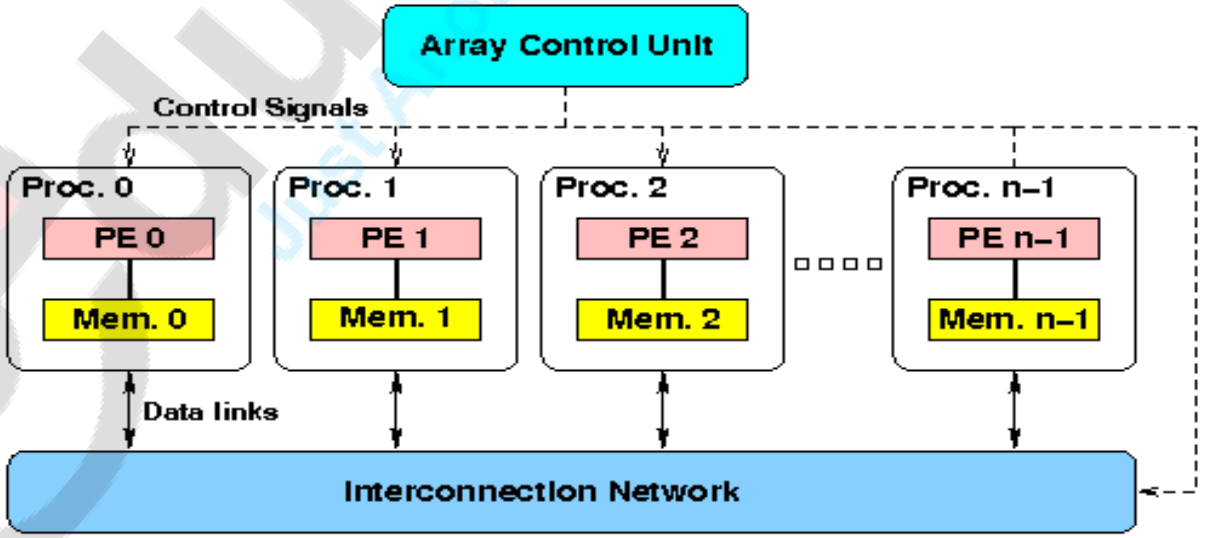
- Instruction cycle consists of multiple pipeline cycles
- Pipelined computer will execute the instruction in every pipeline cycle
- Non-Pipeline computer takes 4 pipeline cycles to execute one instruction
- Once pipeline in filled an out put result is produces from the pipeline of each cycle



Educlash
Just Another Way To Learn

1.b) Array Processors

- Synchronous Parallel Computer with multiple ALU called Processing Element(PE) that operate in Parallel
- All PE Synchronized and connected by interconnection network
- Each PE have some register and Local memory
- Instruction for the PEs are fetched by a common control unit
- Array Processors are used to solve problems that are expressed in matrix or vector format



1.c)Multiprocessors

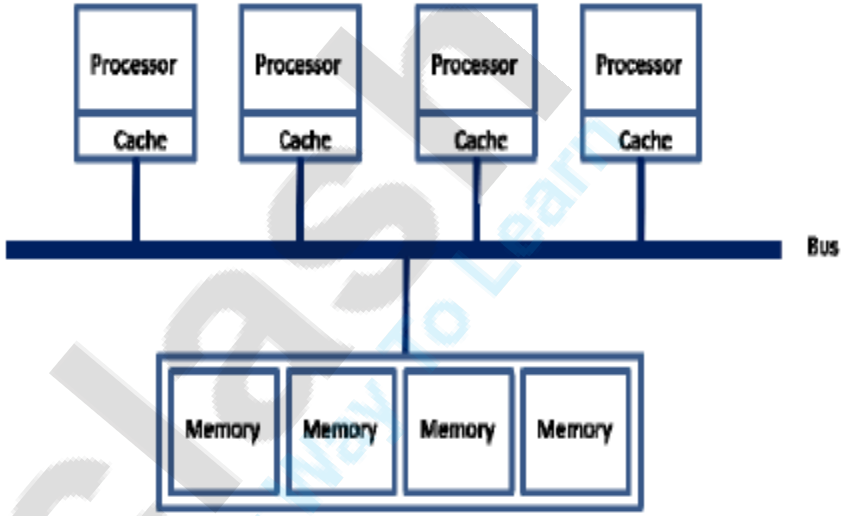
- Two or more processors
- All share common sets of memory modules,IO Channels and peripheral Devices
- Entire system is controlled by single integrated operating system
- OS controls interaction between Processors
- Hardware organization decided by interconnection structure between memory and devices

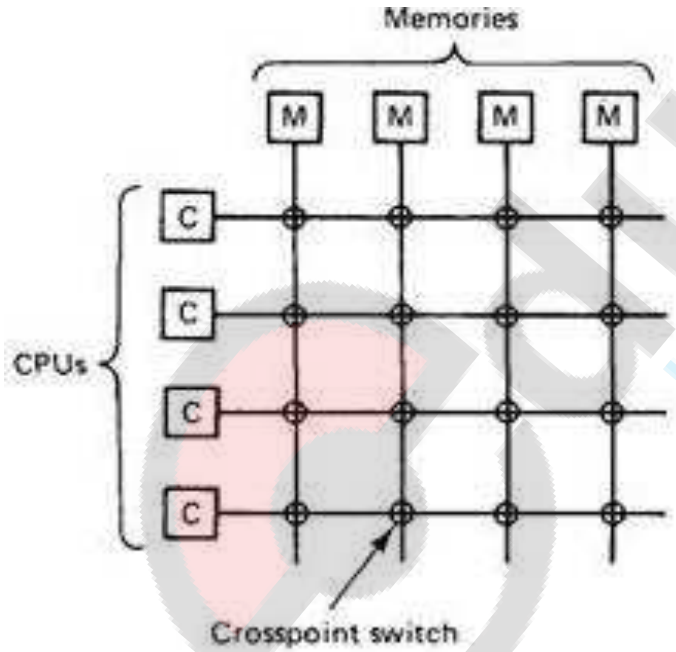
1. Bus-Based Interconnection
2. Switch-Based Interconnection

eduClash Just Another Way To Learn

1.c)Multiprocessors

- Bus-Based interconnection
- Switch-Based interconnection

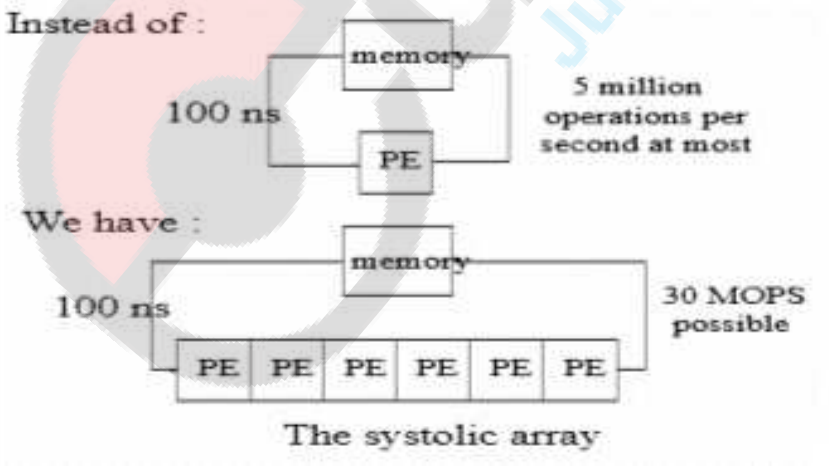




EduClash Just Another Way To Learn

1.d)Systolic Architecture

- Highly synchronized, multi processing with high degree of pipelining Array Architecture ,which consist of an array of PE called cells connected to neighboring cells
- Data to be processed is taken from memory and is processed by first cell ,processed data passed to neighboring cell for further processing
- Each cell performs an operation or a small number of operation on a data item and passes to neighbor
- Processed data by last cell stored in memory
- Useful for dataflow with high throughput with less memory access\

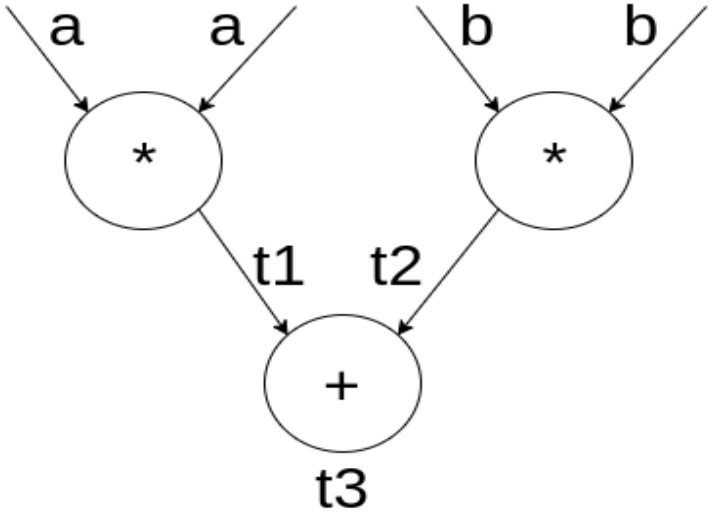


1.d)Dataflow Architecture

- Data Driven Model in which data is represented using directed acyclic graph
- Graph contains nodes and edges
- Node represent instruction edge represent data dependency relationship between connected nodes
- Firing rule : A node can be scheduled for execution if and only if its input data is valid for the consumption
    - Example a*b+c*d
- ☺High potential for parallelism &High throughput for complex computation
- ☹Time loss-Waiting for unnecessary arguments,

High control overhead , Difficult to manipulate

data structure



Educlash
Just Another Way To Learn

Classification based on architectural Schemes

- Flynn’s Classification
- Shore’s Classification

Educlash
Just Another Way To Learn

Flynn's Classification

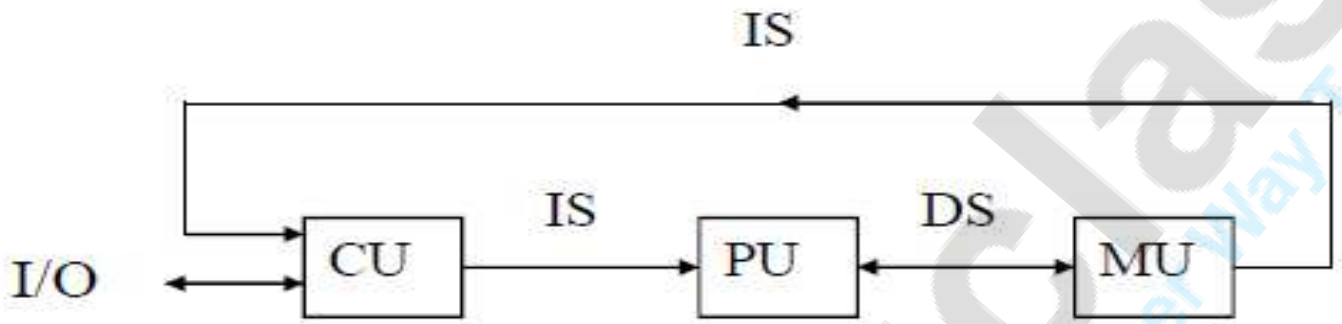
- By Michel Flynn classification in terms of number of instruction streams over data streams

  - SISD–Single Instruction, Single Data Stream
  - MISD-Multiple Instruction Multiple Data Stream
  - SIMD-Single Instruction Multiple Data Stream
  - MIMD-Multiple Instruction Multiple Data Stream.

Educlash
Just Another Way To Learn

Flynn's Classification of Computer Architectures

(Derived from Michael Flynn, 1972)



(a) SISD Uniprocessor Architecture

Captions:

CU - Control Unit ; PU – Processing Unit

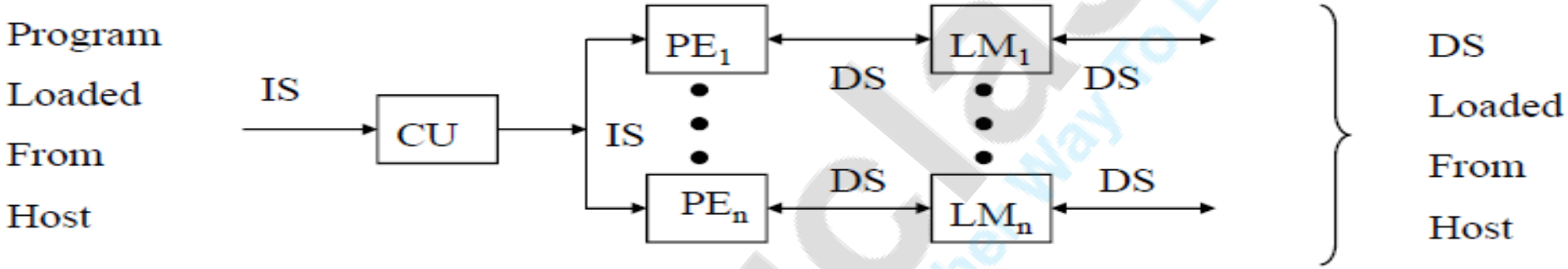
MU – Memory Unit ; IS – Instruction Stream

DS – Date Stream

Educlash
Just Another Way To Learn

Flynn's Classification of Computer Architectures

(Derived from Michael Flynn, 1972) **(contd...)**



(b) SIMD Architecture (with Distributed Memory)

Captions:

CU - Control Unit ; PU - Processing Unit

MU - Memory Unit ; IS - Instruction Stream

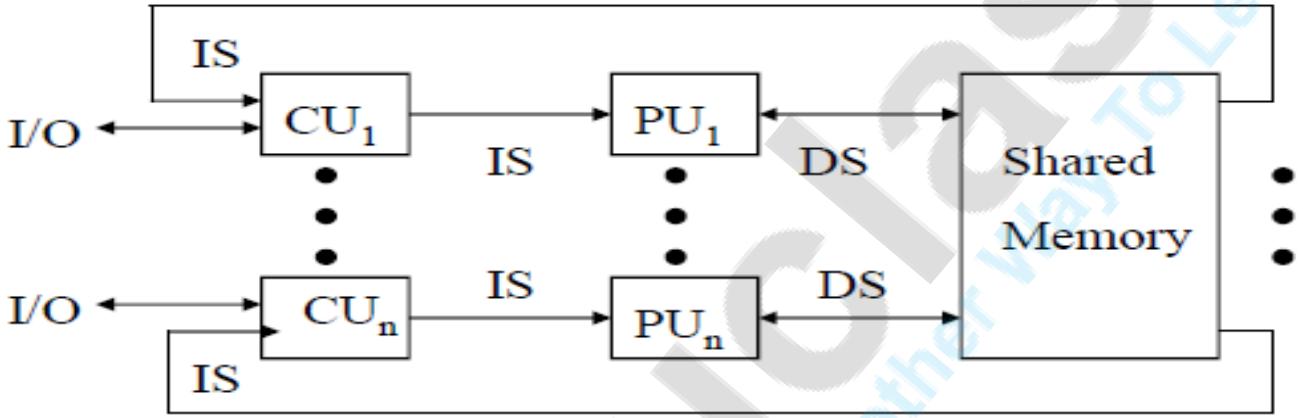
DS - Date Stream ; PE – Processing Element

LM – Local Memory

Educlash Just Another Way To Learn

Flynn's Classification of Computer Architectures

(Derived from Michael Flynn, 1972) **(contd...)**



(c) MIMD Architecture (with Shared Memory)

Captions:

CU - Control Unit ; PU - Processing Unit

MU - Memory Unit ; IS - Instruction Stream

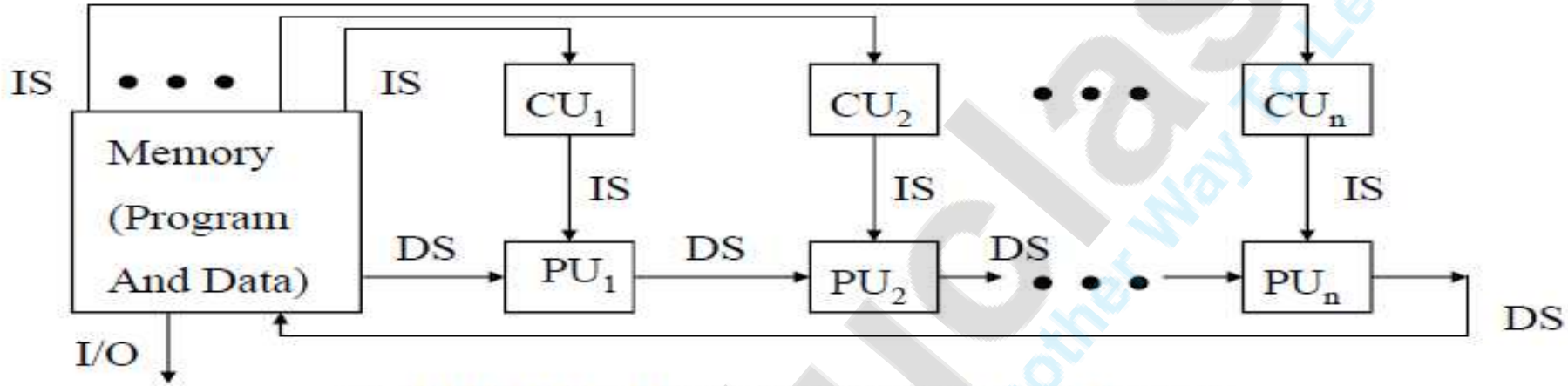
DS - Date Stream ; PE – Processing Element

LM – Local Memory

Educlash Just Another Way To Learn

Flynn's Classification of Computer Architectures

(Derived from Michael Flynn, 1972) (contd...)



(d) MISD Architecture (the Systolic Array)

Captions:

CU - Control Unit ; PU - Processing Unit

MU - Memory Unit ; IS - Instruction Stream

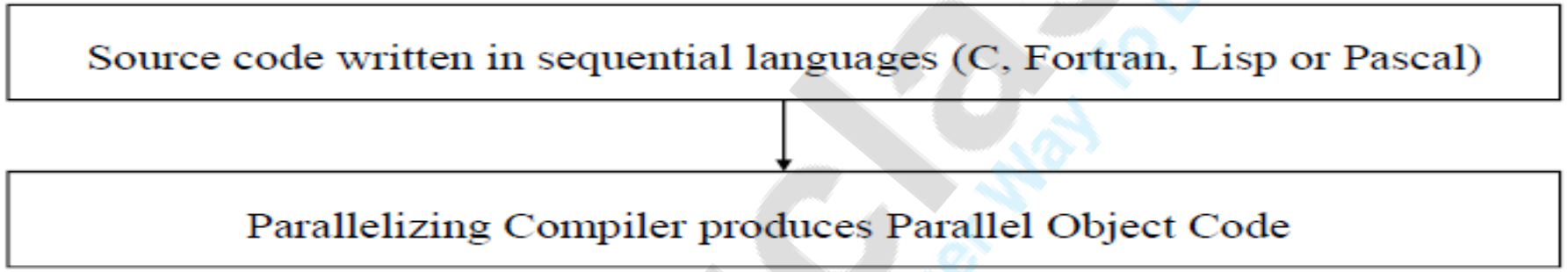
DS - Date Stream ; PE – Processing Element

LM – Local Memory

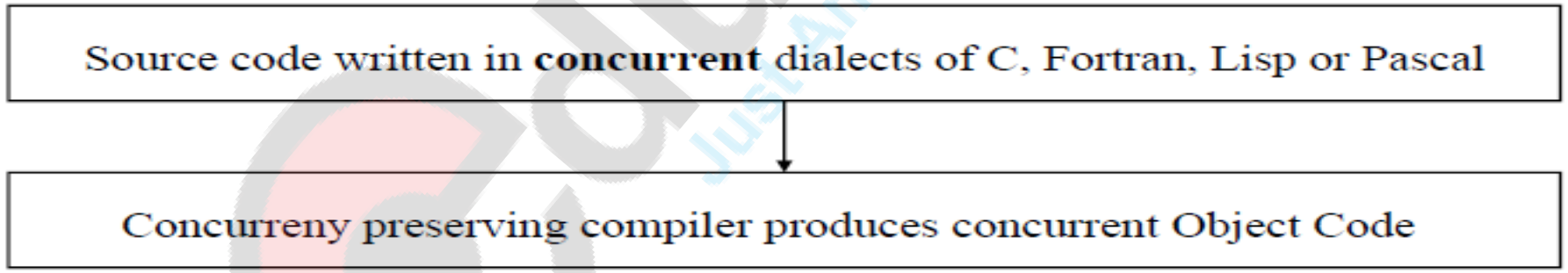
Educlash Just Another Way To Learn

Two Approaches to Parallel Programming

a) Implicit Parallelism



b) Explicit Parallelism



Educlash
Just Another Way To Learn

Two Categories of Parallel Computers

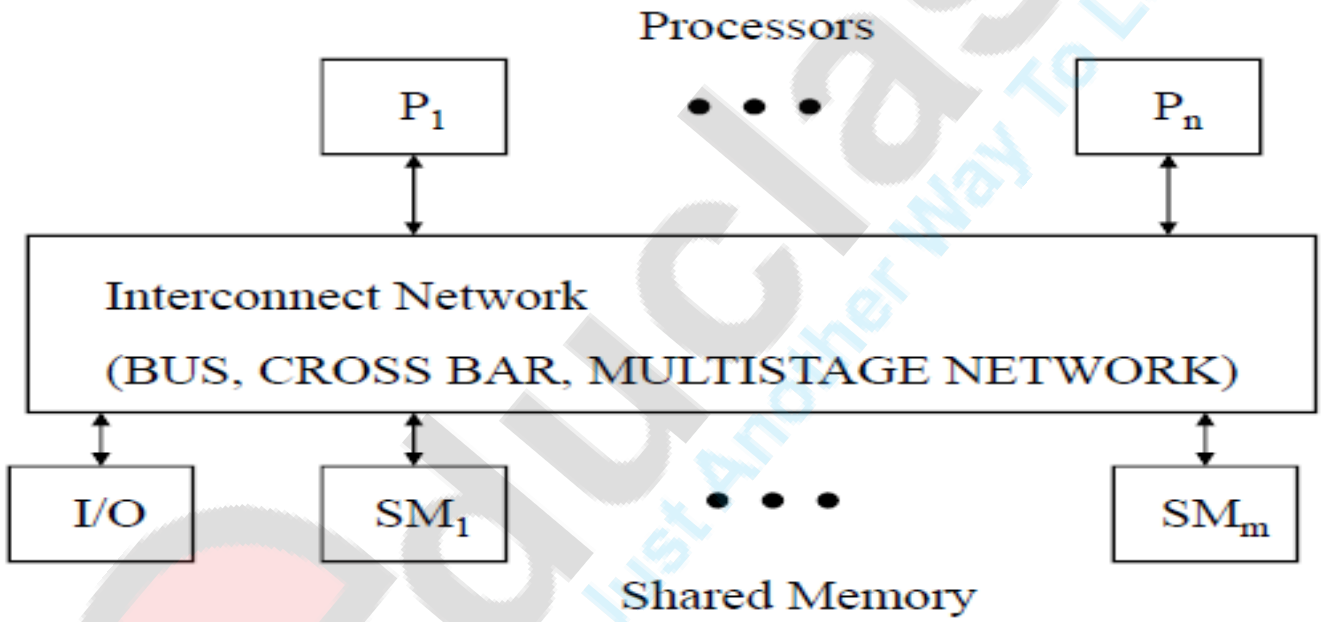
1. Shared Memory Multiprocessors (tightly coupled systems
2. Message Passing Multicomputers

SHARED MEMORY MULTIPROCESSOR MODELS:

a. Uniform Memory Access (UMA)
b. Non-Uniform Memory Access (NUMA)
c. Cache-Only Memory Architecture (COMA)

Educlash
Just Another Way To Learn

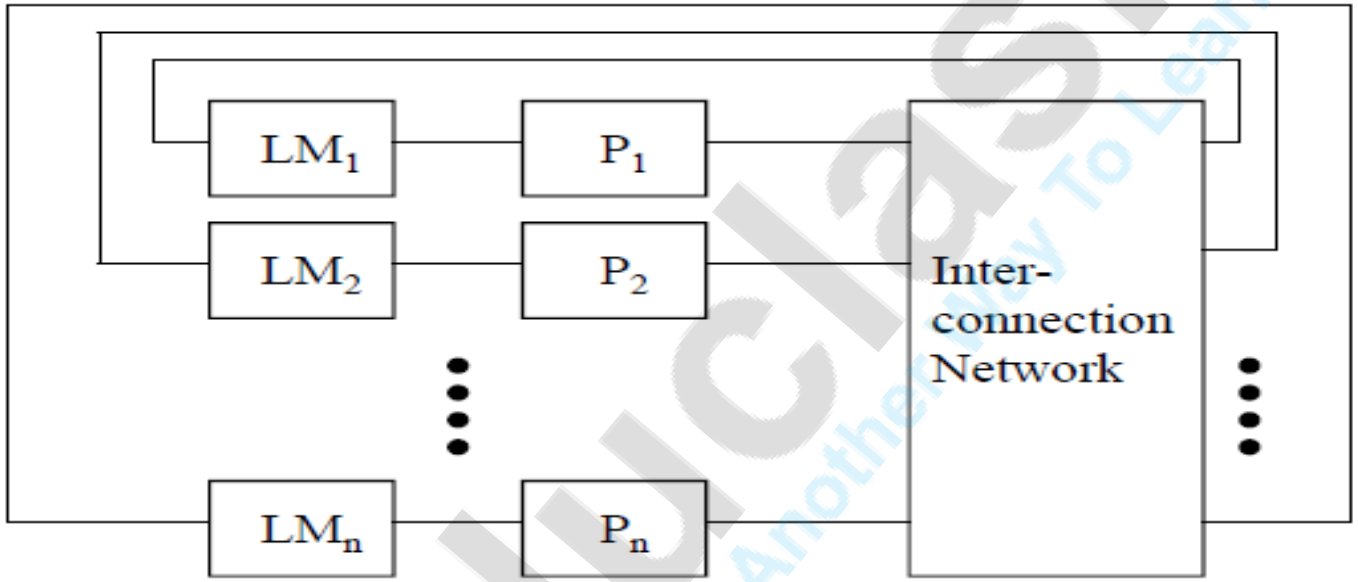
SHARED MEMORY MULTIPROCESSOR MODELS



The **UMA multiprocessor model** (e.g., the Sequent Symmetry S-81)

Educlash
Just Another Way To Learn

SHARED MEMORY MULTIPROCESSOR MODELS (contd...)

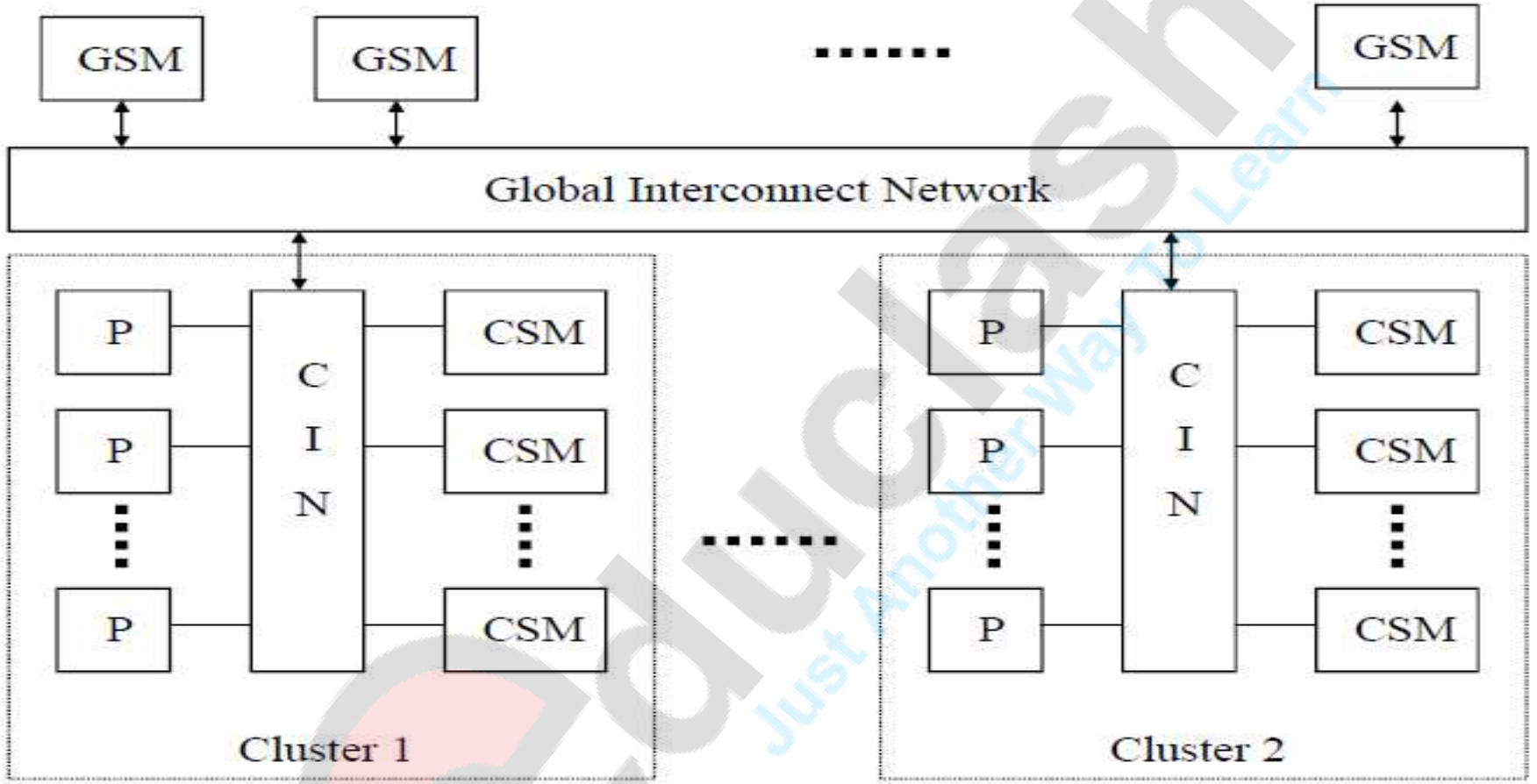


(a) Shared local Memories (e.g., the BBN Butterfly)

NUMA Models for Multiprocessor Systems

Educlash
Just Another Way To Learn

SHARED MEMORY MULTIPROCESSOR MODELS (contd...)



(b) A hierarchical cluster model (e.g., the Cedar system at the University of Illinois)

NUMA Models for Multiprocessor Systems

Educlash
Just Another Way To Learn

Types of Parallelism

1. Data Parallelism: Identical operations on data performed concurrently
2. Task Parallelism/Function Parallelism/Control Parallelism: independent task together
3. Hybrid Parallelism: data + task Parallelism
4. Stream Parallelism : different process in different pipeline
5. Instruction –Level Parallelism :more than one instruction simultaneously
6. Thread-Level Parallelism: split program to independent small parts and run as threads
7. Bit –level parallelism :Passing multiple bits of data parallel

Educlash
Just Another Way To Learn

Parallel Algorithm Models

Educlash
Just Another Way To Learn

Parallel Algorithm Models

- An algorithm model is the representation of a parallel algorithm by selecting a strategy for dividing the data and processing technique and applying the appropriate method to reduce interactions. The various models available are:

    1) The data parallel model
    2) The task graph model
    3) The work pool model
    4) The master slave model
    5) The pipeline or producer consumer model
    6) Hybrid models

Educlash
Just Another Way To Learn

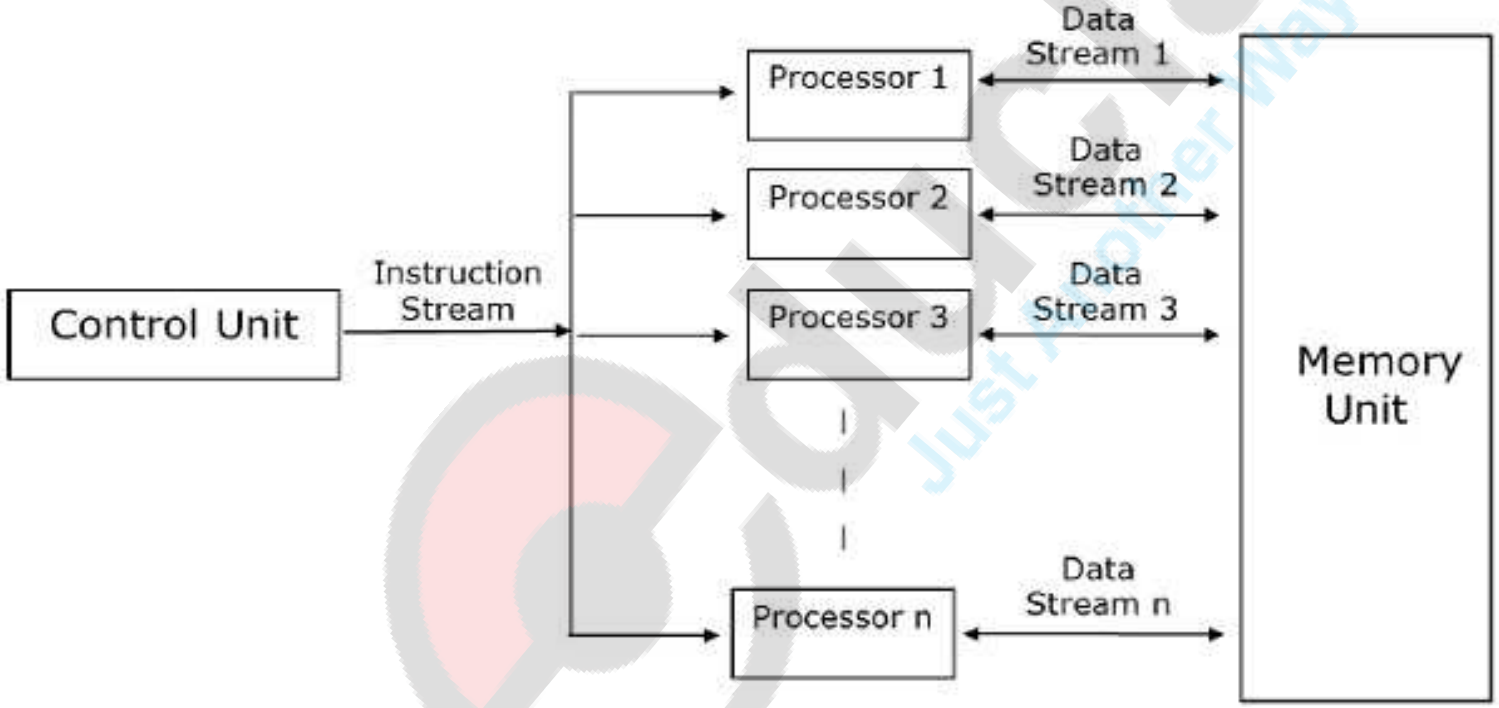
Data Parallel Model

- tasks are assigned to processes and each task performs similar types of operations on different data.
- single operations being applied on multiple data items
- Interaction overheads can be reduced by
  - selecting a locality preserving decomposition
  - using optimized collective interaction routines
  - overlapping computation and interaction.
- the intensity of data parallelism increases with the size of the problem, which in turn makes it possible to use more processes to solve larger problems.

Educlash
Just Another Way To Learn

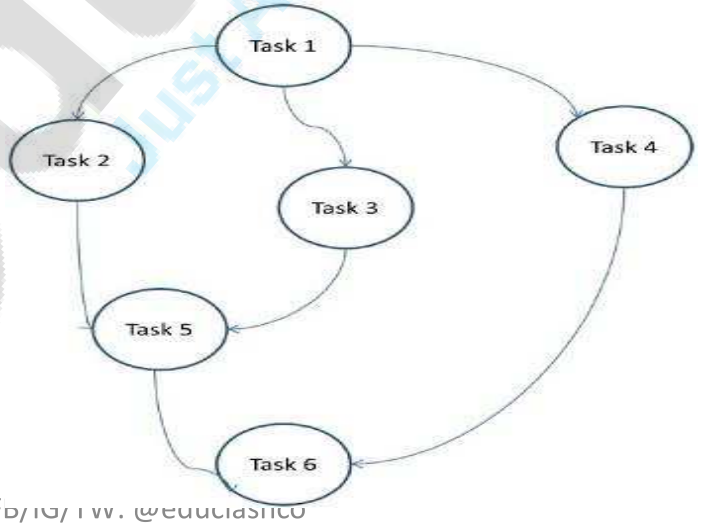
Data Parallel Model

- Example – Dense matrix multiplication.



Task Graph Model

- Parallelism is expressed by a **task graph**
- The correlation among the tasks are utilized to promote locality or to minimize interaction costs.
- This model is enforced to solve problems in which the quantity of data associated with the tasks is huge compared to the number of computation associated with them.
- The tasks are assigned to help improve the cost of data movement among the tasks.



Edulash
Just Another Way To Learn

Work Pool Model

- Tasks are dynamically assigned to the processes for balancing the load.
- This model is used when the quantity of data associated with tasks is comparatively smaller than the computation associated with the tasks.
- No desired pre-assigning of tasks onto the processes.
- Assigning of tasks is centralized or decentralized.

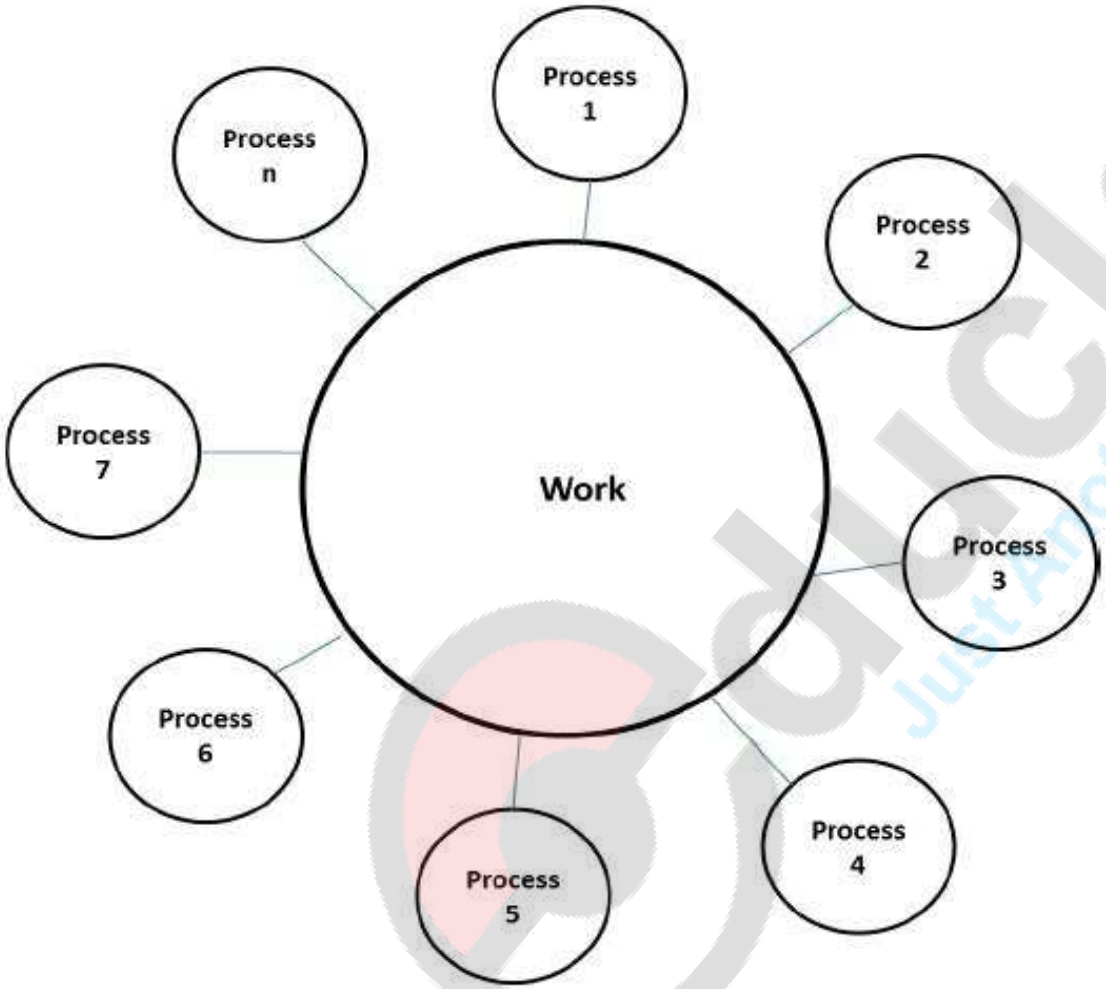
Pointers to the tasks are saved in a physically shared list, in a priority queue, or in a hash table or tree, or they could be saved in a physically distributed data structure.

- The task may be available in the beginning, or may be generated dynamically.

If the task is generated dynamically and a decentralized assigning of task is done, then a termination detection algorithm is required so that all the processes can actually detect the completion of the entire program and stop looking for more tasks.

eduClash Just Another Way To Learn

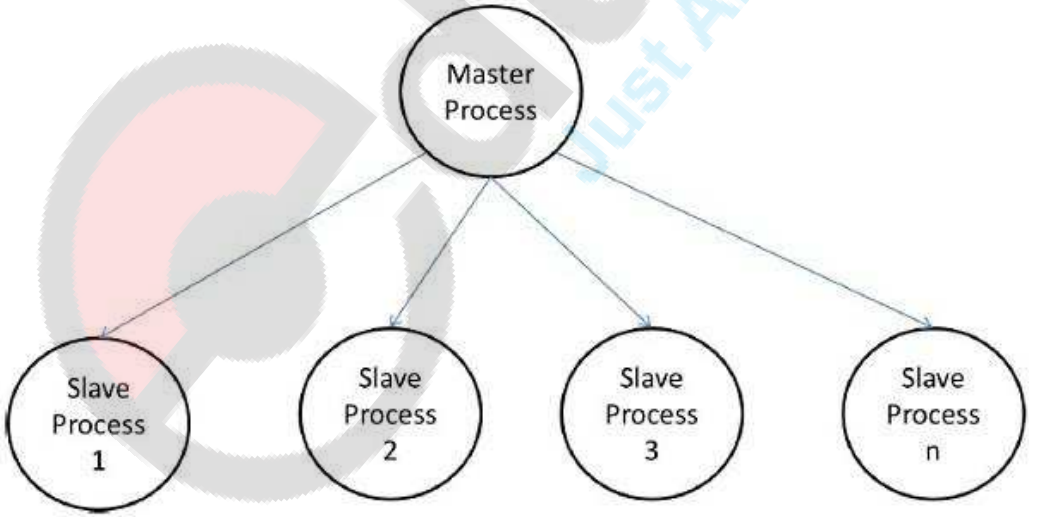
Work Pool Model



Educlash
Just Another Way To Learn

Master-Slave Model

- one or more master processes generate task and allocate it to slave processes.
- The tasks may be allocated beforehand if –
  - the master can estimate the volume of the tasks, or
  - a random assigning can do a satisfactory job of balancing load, or
  - slaves are assigned smaller pieces of task at different times.
- suitable to shared-address-space or message-passing paradigms



Eduklash
Just Another Way To Learn

Pipeline Model

- Also known as the producer-consumer model.
- Set of data is passed on through a series of processes, each of which performs some task on it.
- Here, the arrival of new data generates the execution of a new task by a process in the queue.
- The processes could form a queue in the shape of linear or multidimensional arrays, trees, or general graphs with or without cycles.
- Chain of producers and consumers.
  - Each process in the queue can be considered as a consumer of a sequence of data items for the process preceding it in the queue and as a producer of data for the process following it in the queue.

Educlash
Just Another Way To Learn

Hybrid Models

- A hybrid algorithm model is required when more than one model may be needed to solve a problem.

- A hybrid model may be composed of either multiple models applied hierarchically or multiple models applied sequentially to different phases of a parallel algorithm.

Educlash
Just Another Way To Learn

Topologies in Processor Organization

Ref:introduction to parallel computing

Ananth Grama,Anshul gupta,George karypis,vipin kumar

Educlash
Just Another Way To Learn

Interconnection Networks for Parallel Computers

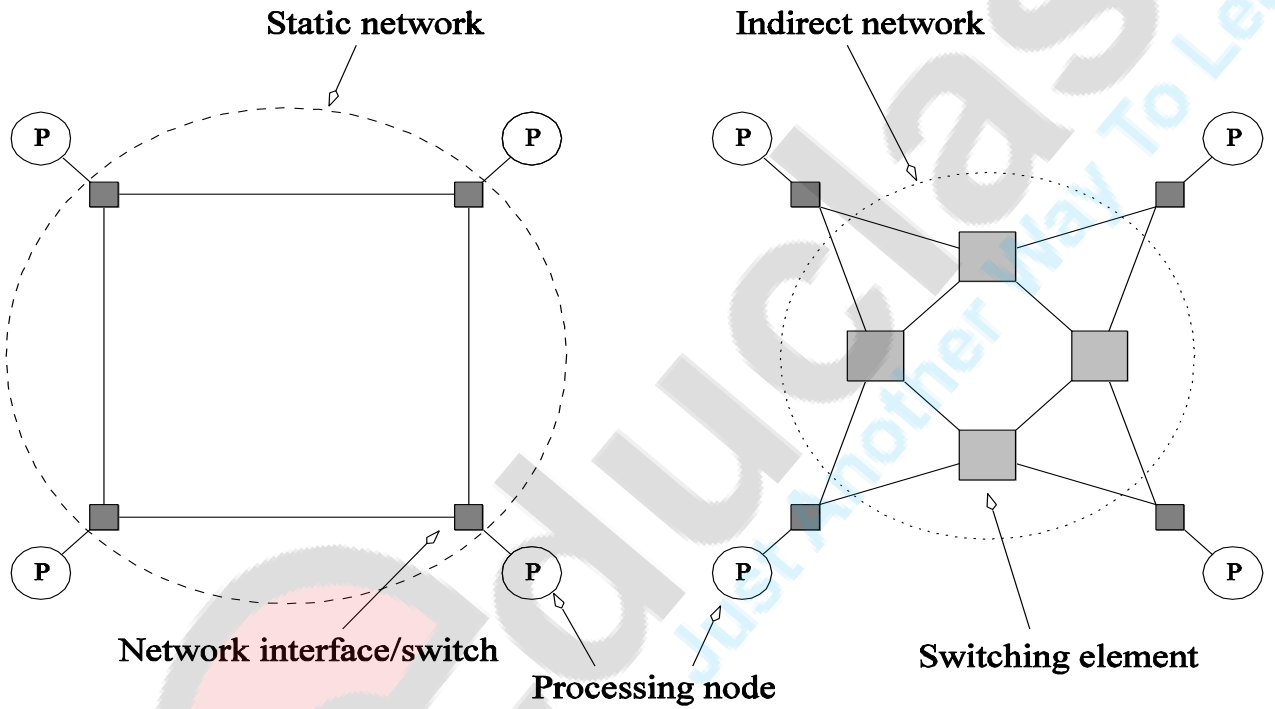
- carry data between **processors** and to **memory**.
- Interconnects are made of **switches** and **links** (wires, fiber).
- Interconnects broadly classified as **static** or **dynamic**.

- Static networks →point-to-point communication links among processing nodes and are also referred to as *direct* networks.

- Dynamic networks→ using switches and communication links. Dynamic networks are also referred to as *indirect* networks.

Educlash Just Another Way to Learn

Static and Dynamic Interconnection Networks



Classification of interconnection networks: (a) a static network; and (b) a dynamic network.

Educlash
Just Another Way To Learn

Interconnection Networks

- Switches map a fixed number of inputs to outputs.
- The total number of ports on a switch is the *degree* of the switch.

- The cost of a switch grows as the square of the degree of the switch, the peripheral hardware linearly as the degree, and the packaging costs linearly as the number of pins.

Educlash
Just Another Way To Learn

Interconnection Networks : Network Interfaces

- Processors talk to the network via a network interface.
- Responsibilities:
  - Packetizing Data
  - Computing routing info
  - Buffer incoming and out going data
  - Error checking
- The network interface may hang off the I/O bus or the memory bus.

Educlash
Just Another Way To Learn

Network Topologies

- A variety of network topologies have been proposed and implemented.
- These topologies tradeoff performance against cost and scalability .
- Commercial machines often implement hybrids of multiple topologies for reasons of packaging, cost, and available components.

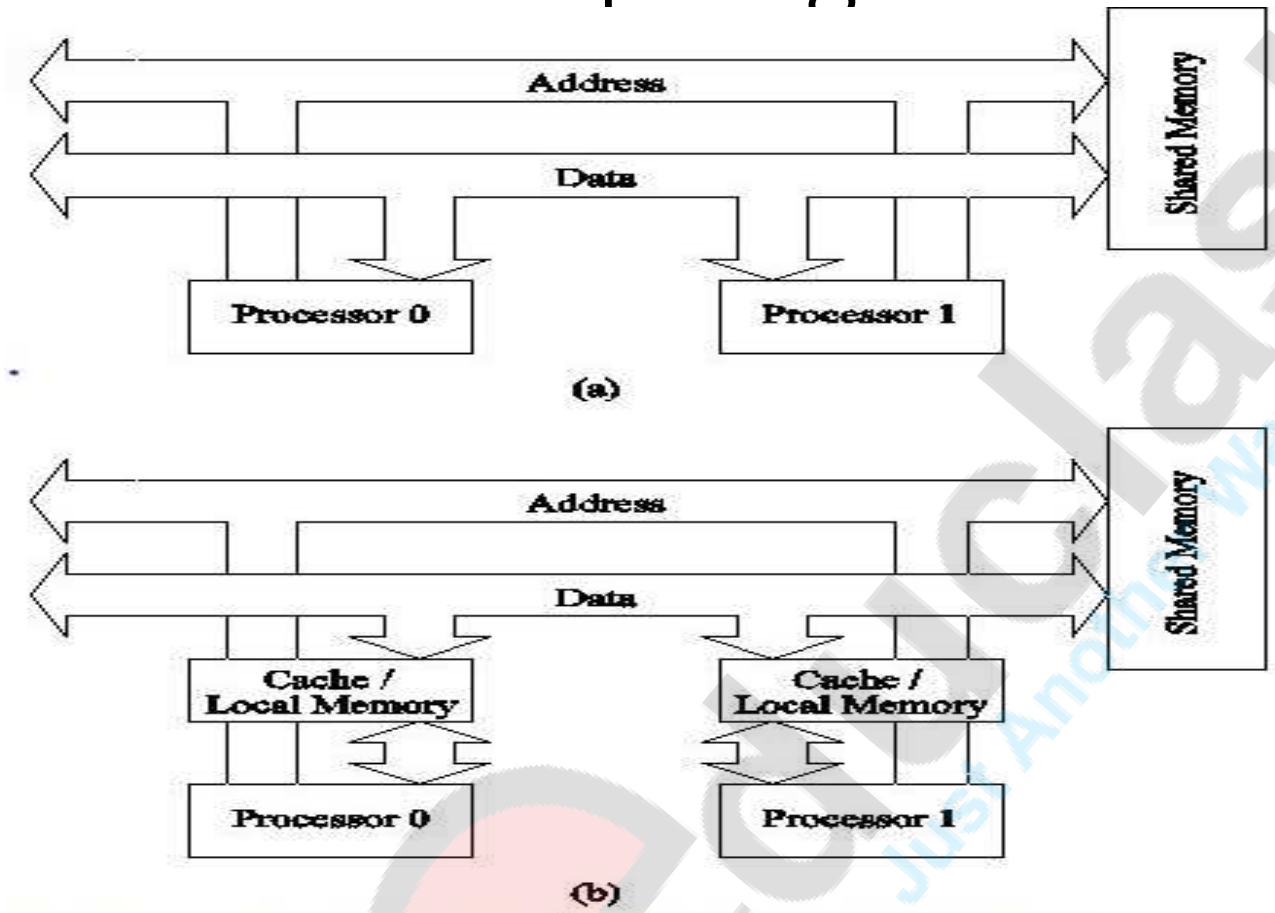
Educlash
Just Another Way To Learn

Network Topologies: Buses

- Some of the 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.
- Convenient broadcast media.
- ☹**bandwidth** ⬇ as # of nodes ⬆ → bus based machines are limited to dozens of nodes.
- Sun Enterprise servers and Intel Pentium based shared-bus multiprocessors are examples of such architectures.

Educlash
Just Another Way To Learn

Network Topologies: Buses

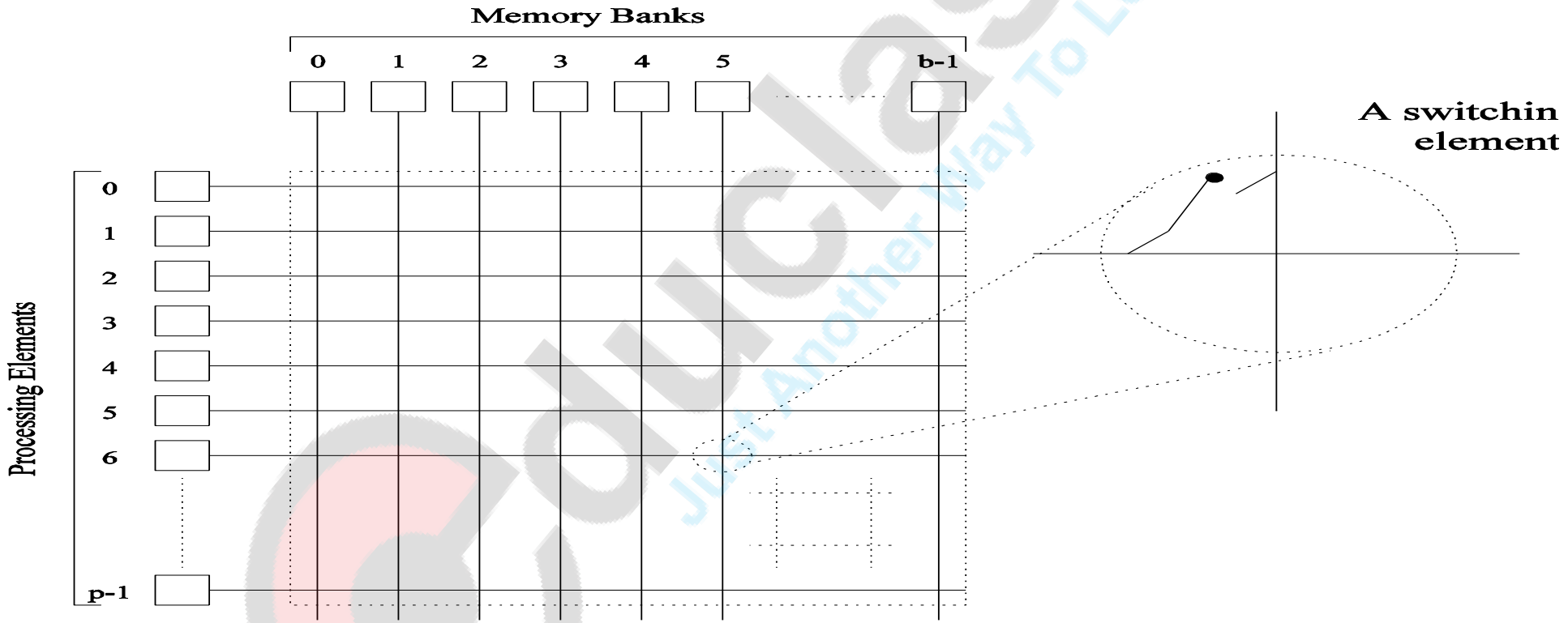


Bus-based interconnects (a) with no local caches; (b) with local memory/caches.

Since much of the data accessed by processors is local to the processor, a local memory can improve the performance of bus-based machines.

Network Topologies: Crossbars

A crossbar network uses an *p*×*m* grid of switches to connect *p* inputs to m outputs in a non-blocking manner.



A completely non-blocking crossbar network connecting *p* processors to b memory banks.

Educlash
Just Another Way To Learn

Network Topologies: Crossbars

- The cost of a crossbar of *p* processors grows as $O(p^2)$.
- *Non-Blocking Network*
- *# of switching → Big Theta(pb).....Number of b must at least p*
- *Component count (Complexity) ohm(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.

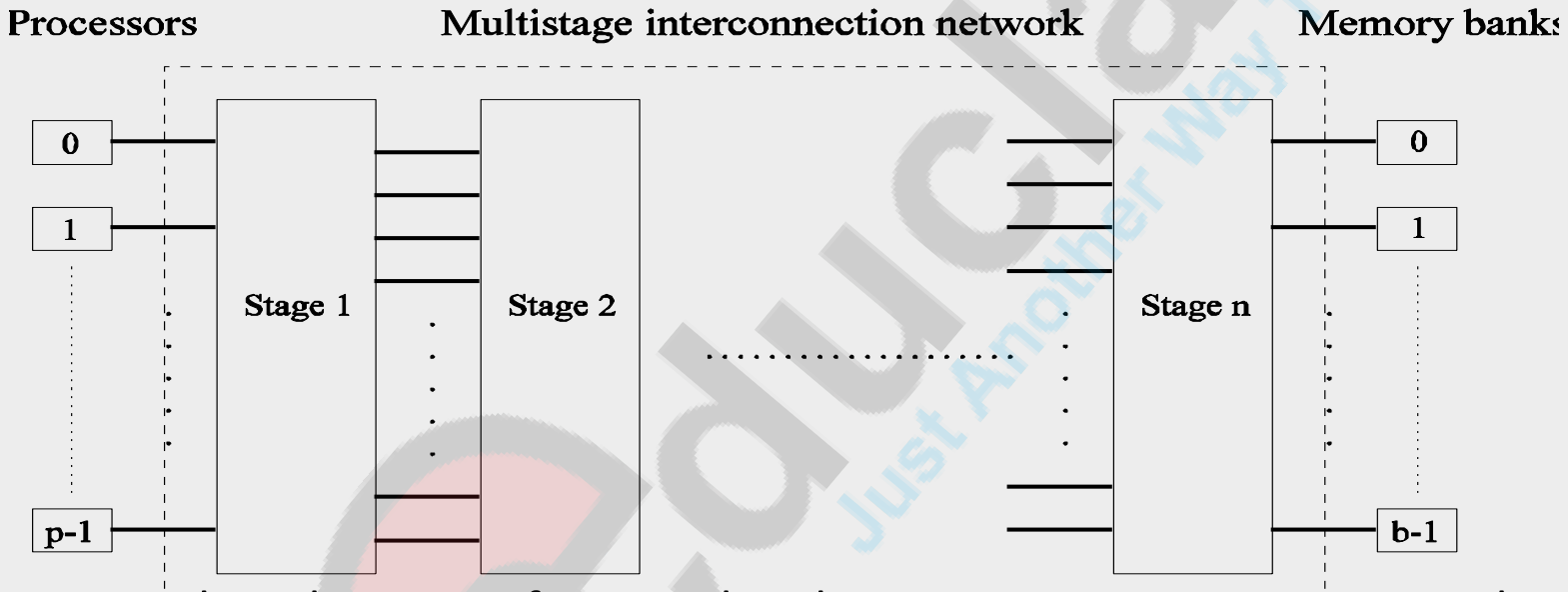
Educlash
Just Another Way To Learn

Network Topologies : Multistage Networks

- Crossbars have excellent performance scalability but poor cost scalability.
- Buses have excellent cost scalability, but poor performance scalability.
- Multistage interconnects strike a compromise between these extremes.

Educlash
Just Another Way To Learn

Network Topologies: Multistage Networks



The schematic of a typical multistage interconnection network.

Educlash
Just Another Way To Learn

Network Topologies: Multistage Omega Network

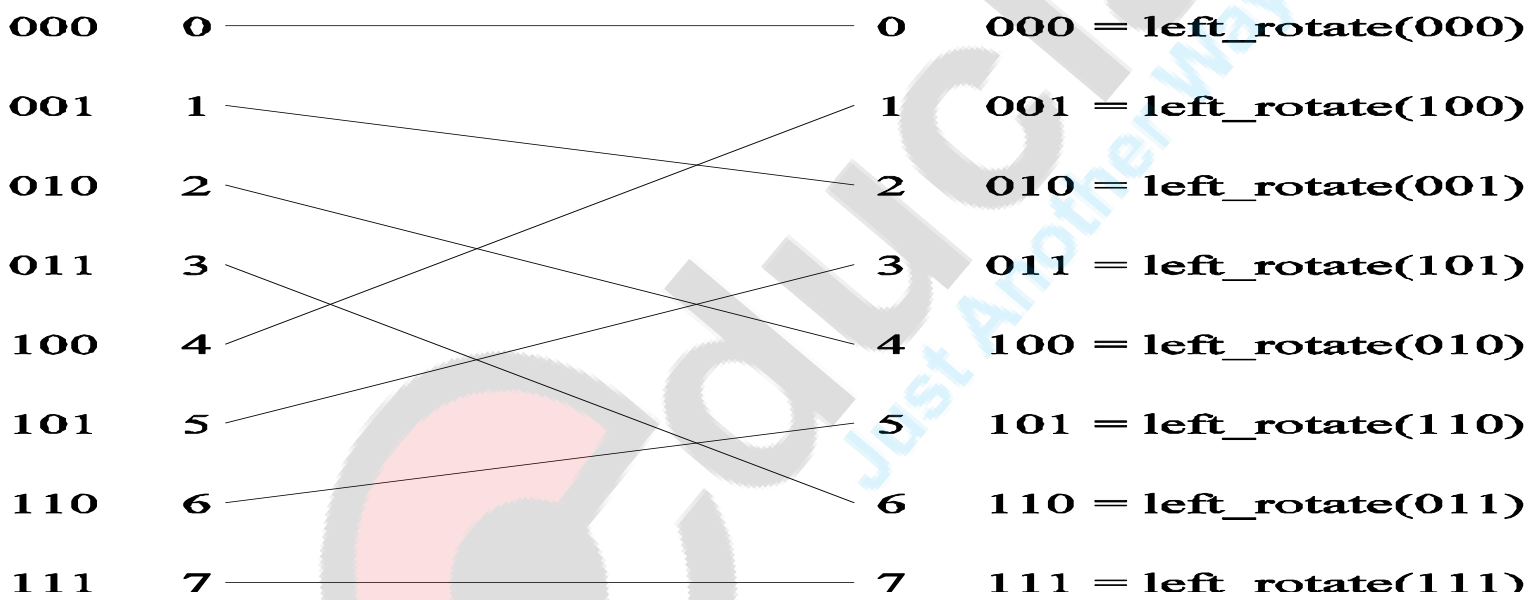
- One of the most commonly used multistage interconnects is the Omega network.
- This network consists of *log p* stages, where *p* is the number of inputs/outputs.
- At each stage, input *i* is connected to output *j* if:

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

Educlash
Just Another Way To Learn

Network Topologies: Multistage Omega Network

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



A perfect shuffle interconnection for eight inputs and outputs.

Network Topologies:Multistage Omega Network

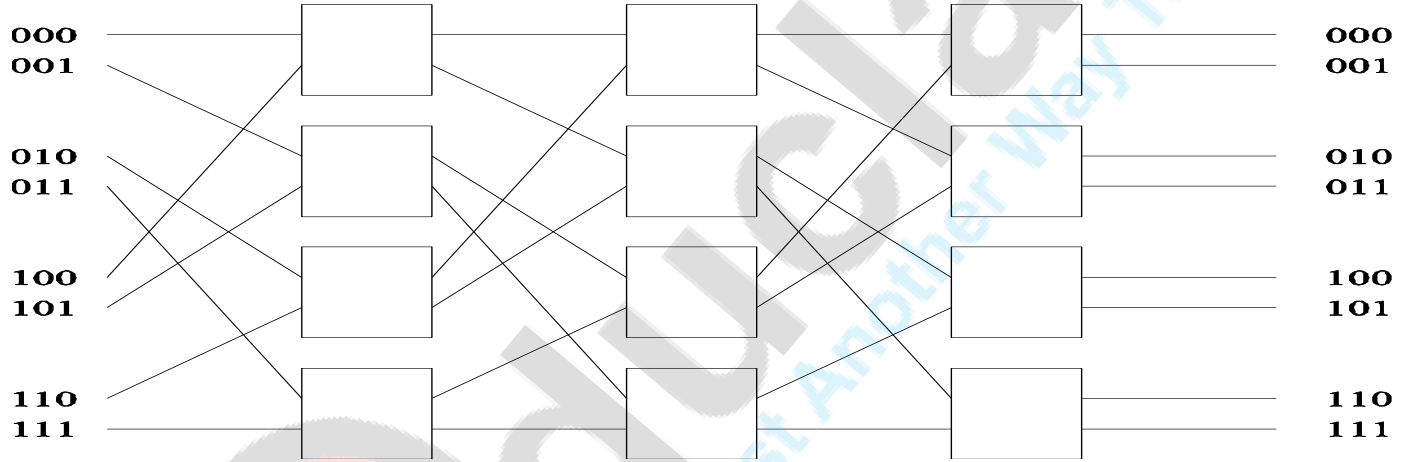
- The perfect shuffle patterns are connected using 2×2 switches.
- The switches operate in two modes – crossover or passthrough.

Two switching configurations of the 2 × 2 switch:
(a) Pass-through; (b) Cross-over.

Educlash
Just Another Way To Learn

Network Topologies: Multistage Omega Network

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



A complete omega network connecting eight inputs and eight outputs.

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

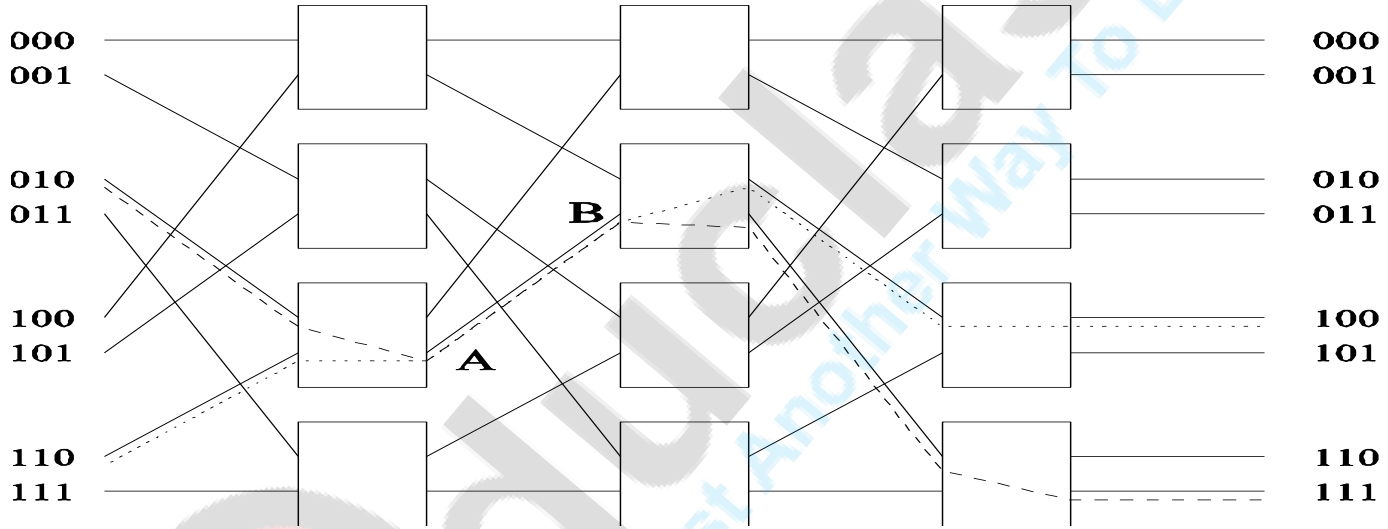
Educlash
Just Another Way To Learn

Network Topologies: Multistage Omega Network – Routing

- Let *s* be the binary representation of processor and t be that of the memory bank
- The data traverses the link to the first switching node. If the most significant bits of *s* and t are the same, then the data is routed in pass-through mode by the switch else, it switches to crossover.
- This process is repeated for each of the *log p* switching stages.
- Note that this is not a non-blocking switch.

EduClash Just Another Way To Learn

Network Topologies: Multistage Omega Network – Routing



An example of blocking in omega network: one of the messages (010 to 111 or 110 to 100) is blocked at link AB.

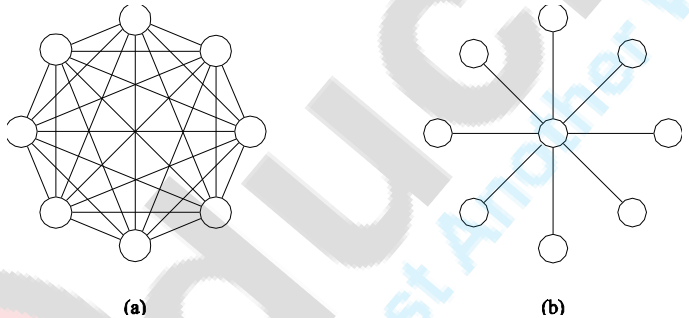
Network Topologies: Completely Connected Network

- Each processor is connected to every other processor.
- The number of links in the network scales as $O(p^2)$.
- While the performance scales very well, the hardware complexity is not realizable for large values of p.
- In this sense, these networks are static counterparts of crossbars.

Educlash
Just Another Way To Learn

Network Topologies: Completely Connected and Star Connected Networks

Example of an 8-node completely connected network.



(a) A completely-connected network of eight nodes;
(b) a star connected network of nine nodes.

Educlash
Just Another Way To Learn

Network Topologies: Star Connected Network

- Every node is connected only to a common node at the center.
- Distance between any pair of nodes is *O(1).* However, the central node becomes a bottleneck.
- In this sense, star connected networks are static counterparts of buses.

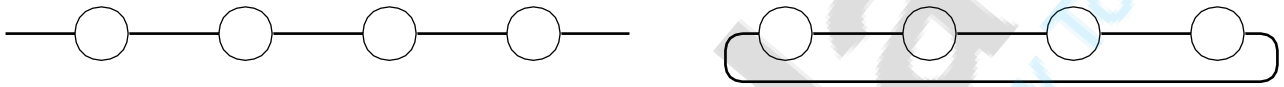
Educlash
Just Another Way To Learn

Network Topologies: Linear Arrays, Meshes, and *k-d* Meshes

- In a linear array, each node has two neighbors, one to its left and one to its right. If the nodes at either end are connected, we refer to it as a 1-D torus or a ring.
- A generalization to 2 dimensions has nodes with 4 neighbors, to the north, south, east, and west.
- A further generalization to *d* dimensions has nodes with *2d* neighbors.
- A special case of a *d*-dimensional mesh is a hypercube. Here, *d = log p*, where *p* is the total number of nodes.

Educlash
Just Another Way To Learn

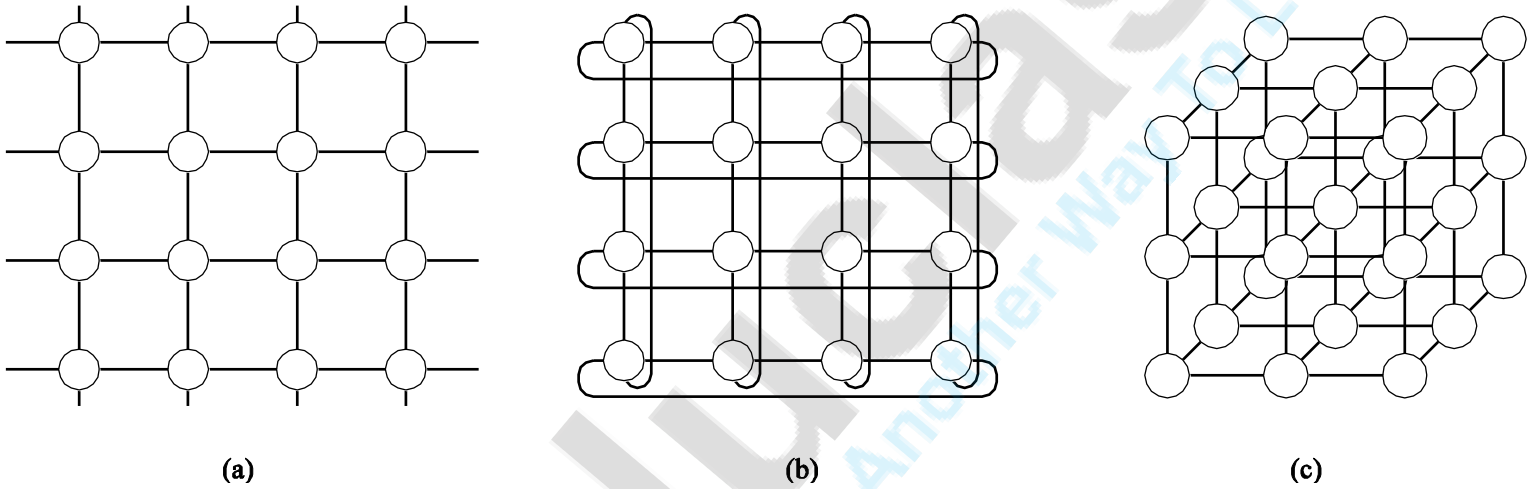
Network Topologies: Linear Arrays



Linear arrays: (a) with no wraparound links; (b) with wraparound link.

Educlash
Just Another Way To Learn

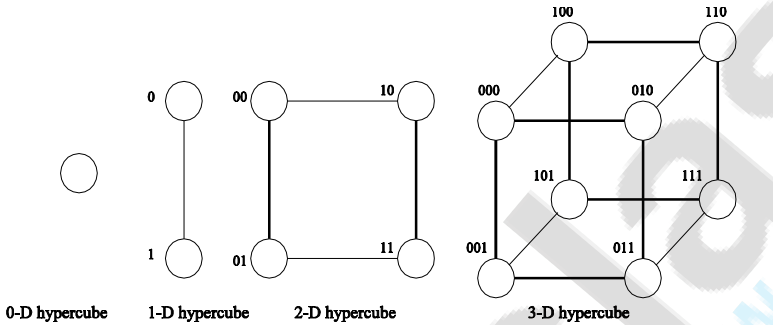
Network Topologies: Two- and Three Dimensional Meshes

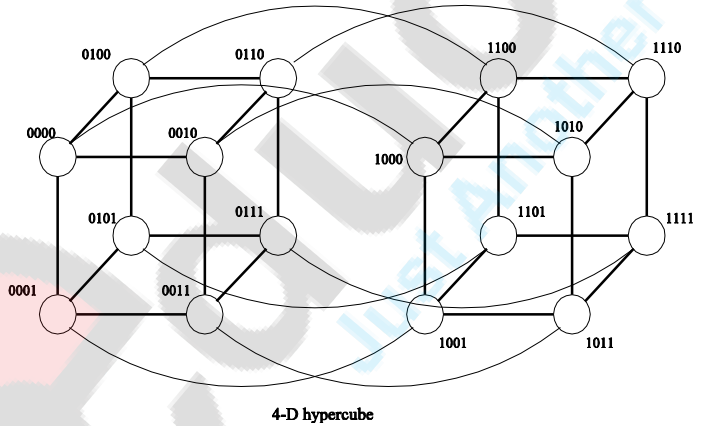


Two and three dimensional meshes: (a) 2-D mesh with no wraparound; (b) 2-D mesh with wraparound link (2-D torus); and (c) a 3-D mesh with no wraparound.

Eduqash
Just Another Way To Learn

Network Topologies: Hypercubes and their Construction





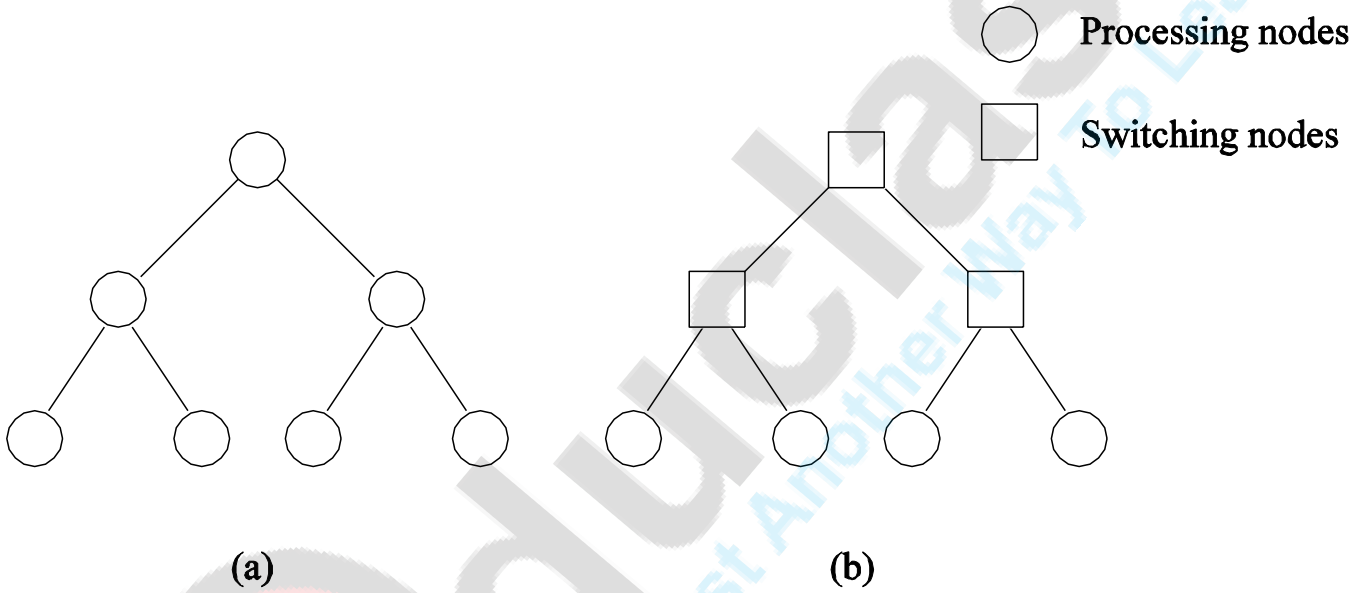
Construction of hypercubes from hypercubes of lower dimension.

Network Topologies: Properties of Hypercubes

- The distance between any two nodes is at most *log p*.
- Each node has *log p* neighbors.
- The distance between two nodes is given by the number of bit positions at which the two nodes differ.

EduClash
Just Another Way To Learn

Network Topologies: Tree-Based Networks



Complete binary tree networks: (a) a static tree network; and (b) a dynamic tree network.

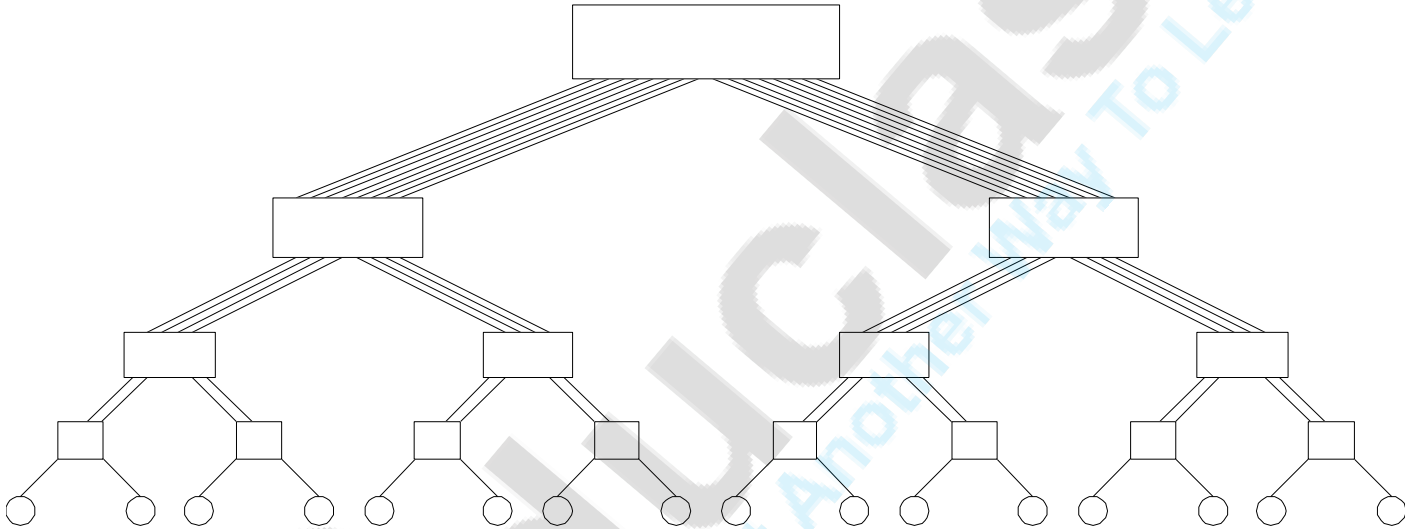
Educlash
Just Another Way To Learn

Network Topologies: Tree Properties

- The distance between any two nodes is no more than *2logp*.
- Links higher up the tree potentially carry more traffic than those at the lower levels.
- For this reason, a variant called a fat-tree, fattens the links as we go up the tree.
- Trees can be laid out in 2D with no wire crossings. This is an attractive property of trees.

Educlash
Just Another Way To Learn

Network Topologies: Fat Trees



A fat tree network of 16 processing nodes.

EduClash
Just Another Way To Learn

Evaluating Static Interconnection Networks

- *Diameter:* The distance between the farthest two nodes in the network. The diameter of a linear array is $p - 1$, that of a mesh is $2(\sqrt{p} - 1)$, that of a tree and hypercube is $\log p$, and that of a completely connected network is $O(1)$.
- *Bisection Width:* The minimum number of wires you must cut to divide the network into two equal parts. The bisection width of a linear array and tree is *1*, that of a mesh is $\sqrt{p}$, that of a hypercube is $p/2$ and that of a completely connected network is $p^2/4$.
- *Cost:* The number of links or switches (whichever is asymptotically higher) is a meaningful measure of the cost. However, a number of other factors, such as the ability to layout the network, the length of wires, etc., also factor in to the cost.

eduClash Just Another Way To Learn

Evaluating Static Interconnection Networks

Network	Diameter	Bisection Width	Arc Connectivity	Cost (No. of links)
Completely-connected	1	$p^2/4$	$p-1$	$p(p-1)/2$
Star	2	1	1	$p-1$
Complete binary tree	$2\log((p+1)/2)$	1	1	$p-1$
Linear array	$p-1$	1	1	$p-1$
2-D mesh, no wraparound	$2(\sqrt{p}-1)$	$\sqrt{p}$	2	$2(p-\sqrt{p})$
2-D wraparound mesh	$2\lfloor\sqrt{p}/2\rfloor$	$2\sqrt{p}$	4	$2p$
Hypercube	$\log p$	$p/2$	$\log p$	$(p\log p)/2$
Wraparound *k*-ary *d*-cube	$d\lfloor k/2\rfloor$	$2k^{d-1}$	$2d$	dp

Educlash Just Another Way To Learn

Evaluating Dynamic Interconnection Networks

Network	Diameter	Bisection Width	Arc Connectivity	Cost (No. of links)
Crossbar	1	p	1	p^2
Omega Network	$\log p$	$p/2$	2	$p/2$
Dynamic Tree	$2 \log p$	1	2	$p - 1$

Educlash
Just Another Way To Learn

Control Structure of Parallel Programs

- Parallelism can be expressed at various levels of granularity - from instruction level to processes.
- Between these extremes exist a range of models, along with corresponding architectural support.

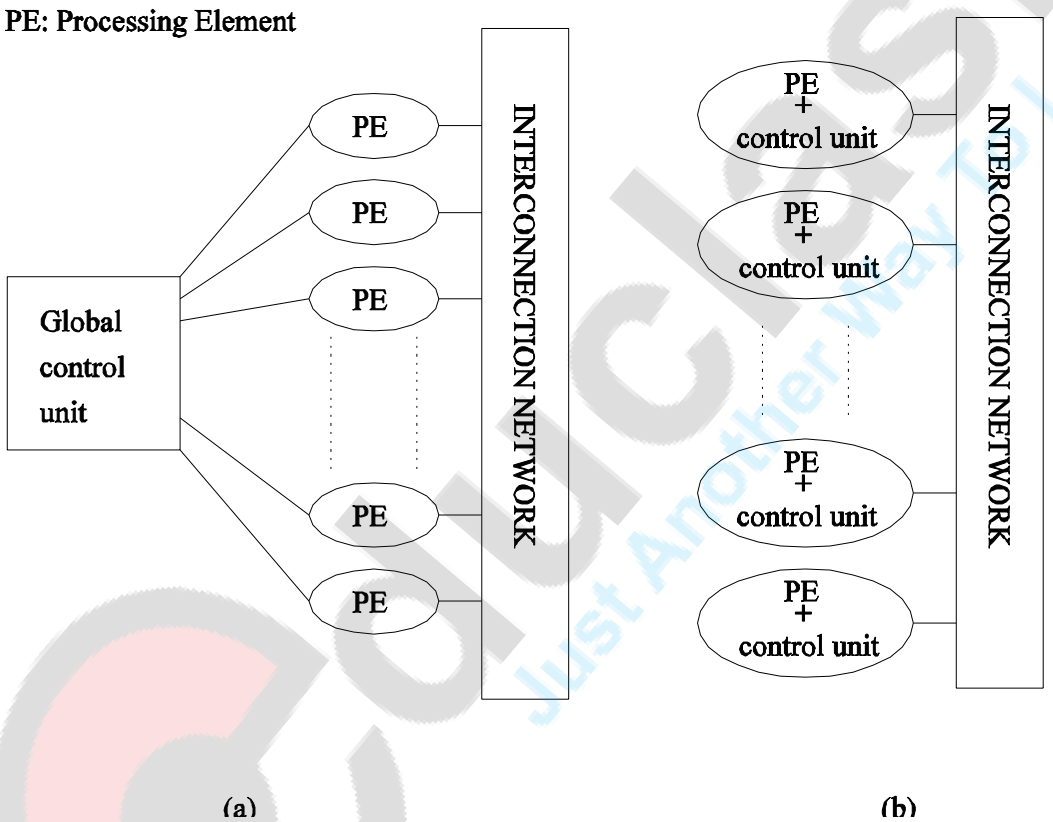
Educlash
Just Another Way To Learn

Control Structure of Parallel Programs

- Processing units in parallel computers either operate under the centralized control of a single control unit or work independently.
- If there is a single control unit that dispatches the same instruction to various processors (that work on different data), the model is referred to as single instruction stream, multiple data stream (SIMD).
- If each processor has its own control control unit, each processor can execute different instructions on different data items. This model is called multiple instruction stream, multiple data stream (MIMD).

eduClash
Just Another Place To Learn

SIMD and MIMD Processors



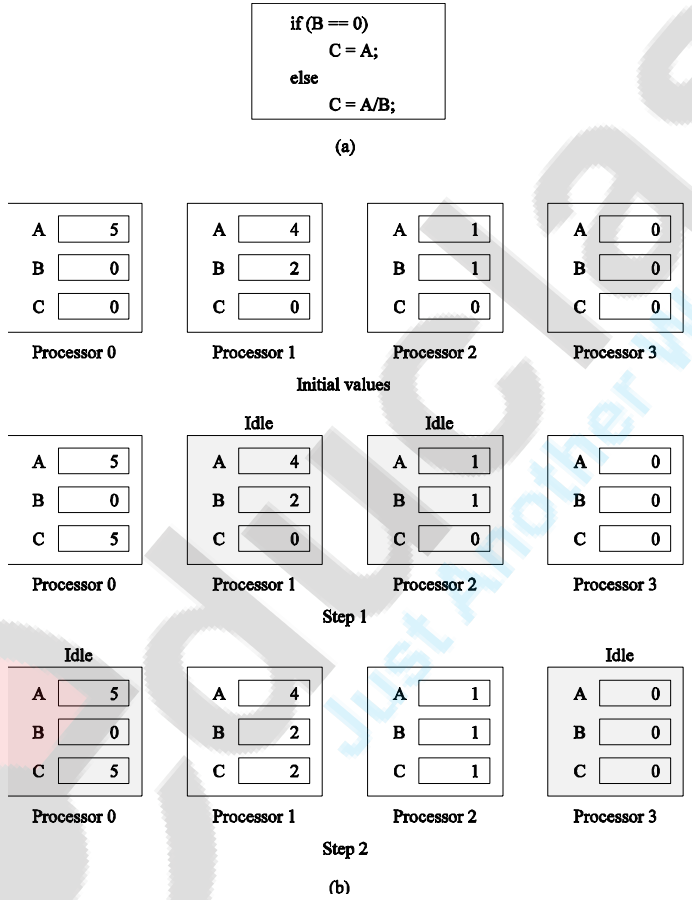
A typical SIMD architecture (a) and a typical MIMD architecture (b).

SIMD Processors

- Some of the earliest parallel computers such as the Illiac IV, MPP, DAP, CM-2, and MasPar MP-1 belonged to this class of machines.
- Variants of this concept have found use in co-processing units such as the MMX units in Intel processors and DSP chips such as the Sharc.
- SIMD relies on the regular structure of computations (such as those in image processing).
- It is often necessary to selectively turn off operations on certain data items. For this reason, most SIMD programming paradigms allow for an ``activity mask'', which determines if a processor should participate in a computation or not.

EduClash Just Another Way To Learn

Conditional Execution in SIMD Processors



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 Processors

- In contrast to SIMD processors, MIMD processors can execute different programs on different processors.
- A variant of this, called single program multiple data streams (SPMD) executes the same program on different processors.
- It is easy to see that SPMD and MIMD are closely related in terms of programming flexibility and underlying architectural support.
- Examples of such platforms include current generation Sun Ultra Servers, SGI Origin Servers, multiprocessor PCs, workstation clusters, and the IBM SP.

Educlash
Just Another Way To Learn

SIMD-MIMD Comparison

- SIMD computers require less hardware than MIMD computers (single control unit).
- However, since SIMD processors ae specially designed, they tend to be expensive and have long design cycles.
- Not all applications are naturally suited to SIMD processors.
- In contrast, platforms supporting the SPMD paradigm can be built from inexpensive off-the-shelf components with relatively little effort in a short amount of time.

Educlash
Just Another Way To Learn

Parallel Architecture

Ref:Parallel and Distributed systems

Kulkarni,Giri,Joshi,Jadhav

Chapter 1.3

Eduqlash
Just Another Way To Learn

Pipelining (Ref:Arun Kulkarni,Nupur Giri Chapter 2)

- Synchronous Pipeline
- Asynchronous Pipeline
- Advantages and limitations

Educlash
Just Another Way To Learn
FB/IG/TW: @educlashco

Parallel Architecture

- Pipeline Computers
- Array Processors
- Multi Processors
- Systolic architecture
- Data Flow Architecture

Educlash
Just Another Way To Learn