UNIT 9
Multiprocessor System

Characteristics of Multiprocessor

- A multiprocessor system is an interconnection of two or more CPUs with memory and input-output equipment.
- The term "processor" in multiprocessor can mean either a central processing unit (CPU) or an input-output processor (IOP). However, a system with a single CPU and one or more IOPs is usually not included in the definition of a multiprocessor system unless the IOP has computational facilities comparable to a CPU.
- Multiprocessors are classified as multiple instruction stream, multiple data MMD stream (MIMD) systems.
- Multiprocessor vs Multicomputer system:
  - There are some similarities between them as they both support concurrent processing.
  - Distinction:
    - Computers are interconnected with each other by means of communication lines to form a computer network.
    - The network consists of several autonomous computers that may or may not communicate with each other.
    - A multiprocessor system is controlled by one operating system that provides interaction between processors and all the components of the system cooperate in the solution of a problem.
- Multiprocessing improves the reliability of the system.

Characteristics of Multiprocessor

- The benefit derived from a multiprocessor organization is an improved system performance.
- The system derives its high performance from the fact that computations can proceed in parallel in one of two ways.
  - Multiple independent jobs can be made to operate in parallel.
  - A single job can be partitioned into multiple parallel tasks.
- Multiprocessing can improve performance by decomposing a program into parallel executable tasks. This can be achieved in one of two ways.:
  - The user can explicitly declare that certain tasks of the program be executed in parallel. This must be done prior to loading the program by specifying the parallel executable segments.
  - The other is to provide a compiler with multiprocessor software that can automatically detect parallelism in a user's program.The compiler checks for data dependency in the program.

Characteristics of Multiprocessor

- Multiprocessors are classified by the way their memory is organized.
  - Tightly Coupled Multiprocessor:
    - A multiprocessor system with common shared memory is classified as a shared memory or tightly coupled multiprocessor.
    - Tightly coupled systems can tolerate a higher degree of interaction between tasks.
  - Loosely coupled Multiprocessor:
    - distributed-memory or loosely coupled system is the one where each processor element has its own private local memory.
    - Loosely coupled systems are most efficient when the interaction between tasks is minimal.

Interconnection Structure

- The components that form a multiprocessor system are CPUs, IOPs connected to input-output devices, and a memory unit that may be partitioned into a number of separate modules.
- The interconnection between the components can have different physical configurations, depending on the number of transfer paths that are available between the processor and memory in a shared memory system or among the processing elements in a loosely coupled system.
- There are several physical forms available for establishing an interconnection network:
  1. Time-shared common bus
  2. Multiport memory
  3. Crossbar switch
  4. Multistage switching network
  5. Hypercube system

Interconnection Structure

Time Shared Common Bus:

- A common-bus multiprocessor system consists of a number of processors connected through a common path to a memory unit.
- Disadvantages:
  - Only one processor can communicate with the memory or another processor at any given time.
  - As a consequence, the total overall transfer rate within the system is limited by the speed of the single path.
- An implementation of a dual bus structure is depicted in figure 8.2

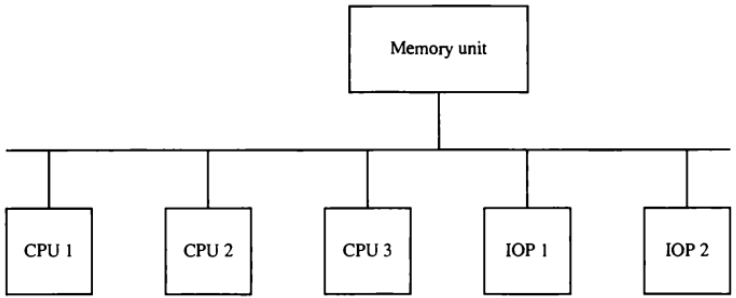


Figure 1 Time-shared common bus organization.

Time Shared Common Bus

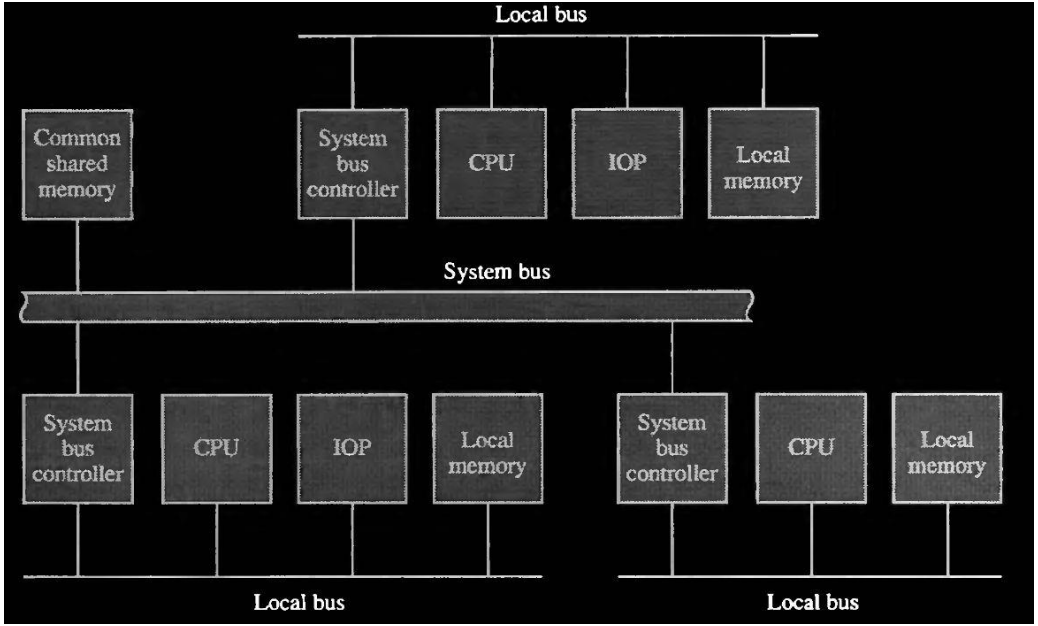


Figure 8.2: System bus structure for multiprocessors.

Interconnection Structure

Multiport Memory:

- A multiport memory system employs separate buses between each memory module and each CPU. i.e each processor bus is connected to each memory module.
- In the given figure 8.3 the memory module is said to have four ports and each port accommodates one of the buses.
- The module must have internal control logic to determine which port will have access to memory at any given time.
- Memory access conflicts are resolved by assigning fixed priorities to each memory port.
- Advantage:
  - high transfer rate that can be achieved because of the multiple paths between processors and memory.
- Disadvantages
  - The disadvantage is that it requires expensive memory control logic and a large number of cables and connectors.

Multiport Memory

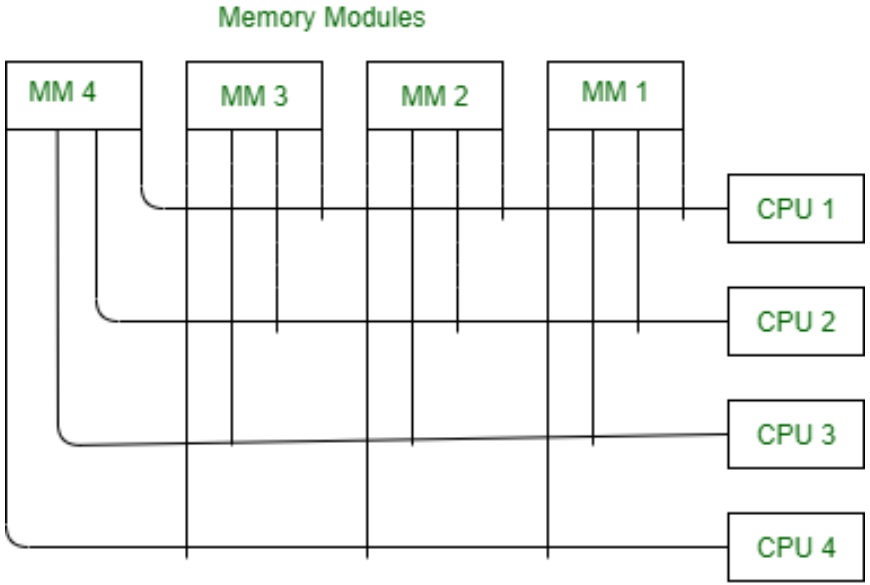


Figure - Multiport Memory System

Figure 8.2: MultiPort Memory Organization

Interconnection Structure

Crossbar Switch

- The crossbar switch organization consists of a number of crosspoints that are placed at intersections between processor buses and memory module paths.
- The small square in each crosspoint is a switch that determines the path from a processor to a memory module.
- Each switch point has control logic to set up the transfer path between a processor and memory.
- It examines the address that is placed in the bus to determine whether its particular module is being addressed.
- It also resolves multiple requests for access to the same memory module on a predetermined priority basis.
- Adv:
  - Supports simultaneous transfers from all memory modules
- Disadv.:
  - The hardware required to implement the switch can become quite large and complex.

Crossbar Switch

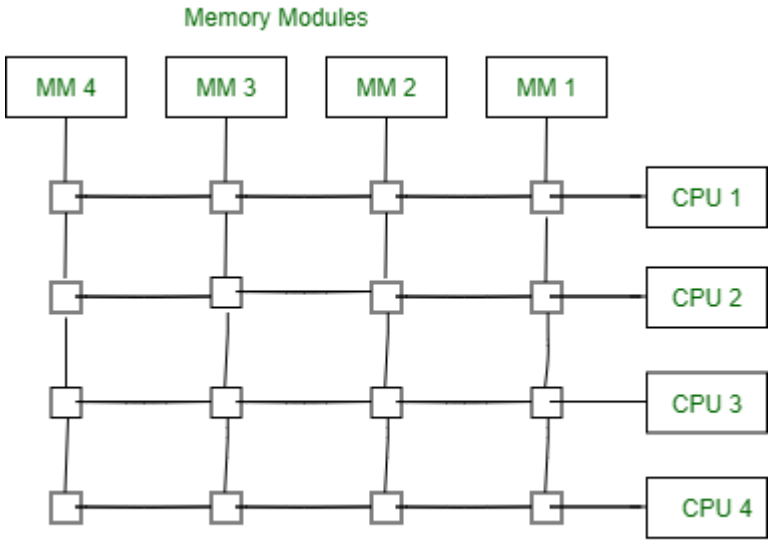


Figure - Crossbar Switch System

Figure 8.3: Crossbar Switch

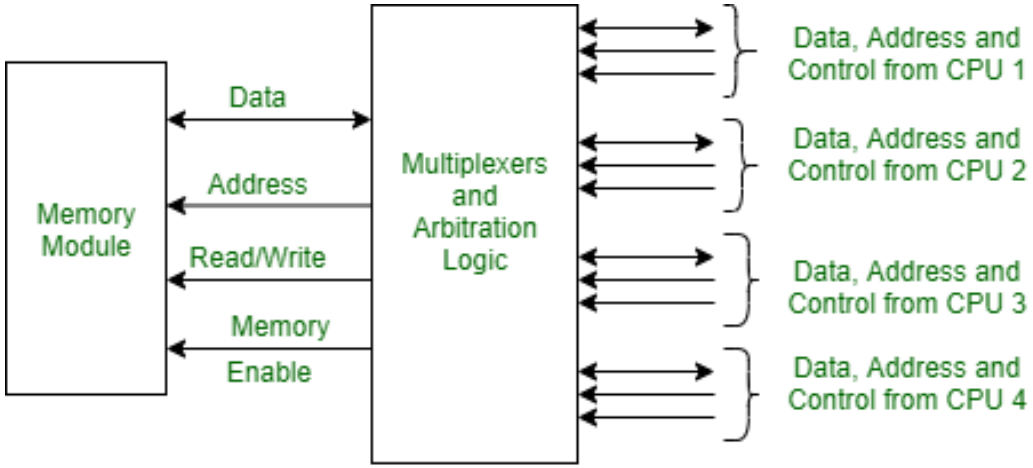


Figure - Crossbar Switch

Figure 8.4: Block diagram of crossbar switch

Interconnection Structure

Multistage Switching Network

The basic component of a multistage network is a two-input, two-output interchange switch. As shown in Fig. 8.5, the 2x2 switch has two input labeled A and B, and two outputs, labeled 0 and 1.

The switch has the capability connecting input A to either of the outputs. Terminal B of the switch behave in a similar fashion.

The switch also has the capability to arbitrate between conflicting requests.

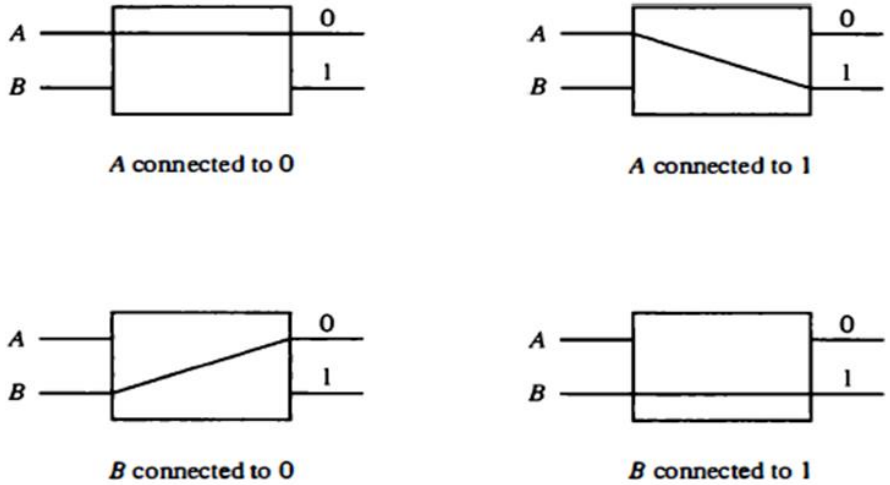


Figure 8.5: Operation of a 2 x 2 interchange switch.

Interconnection Structure

Multistage Switching Network

- Using the 2x2 switch as a building block, it is possible to build multistage network to control the communication between a number of source and destinations.
- To see how this is done, consider the binary tree shown in Fig 8.6
- The two processors Pi and P2 are connected through switches to eight memory modules marked in binary from 000 through 111.
- The path from source to a destination is determined from the binary bits of the destination number.
- The first bit of the destination number determines the switch output in the first level, second bit or the second level, and so on.
- Certain request patterns, however, cannot be satisfied simultaneously. i.e., if P 1 → 000~011, then P 2 → 100~111

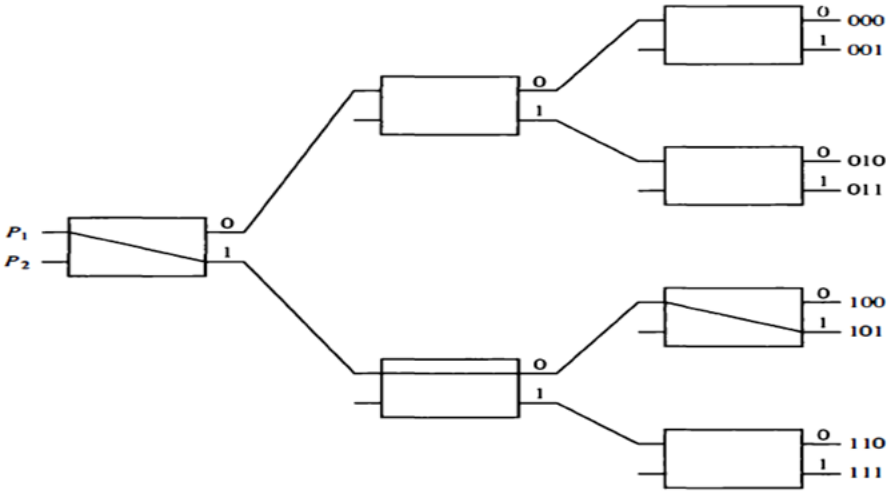


Figure 8.3: Binary tree with 2x2 switches.

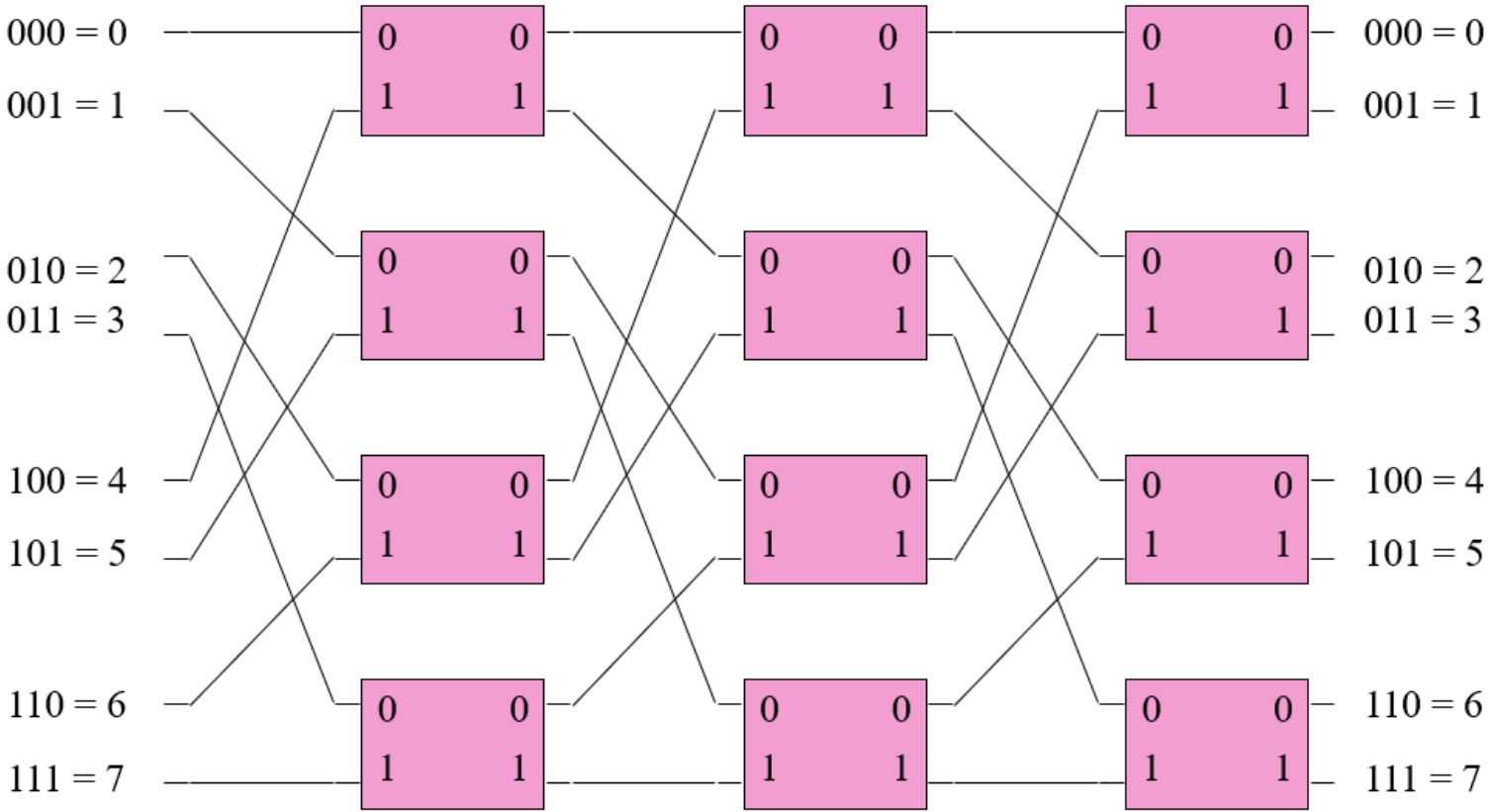
Interconnection Structure

Multistage Switching Network

Omega Switching Network

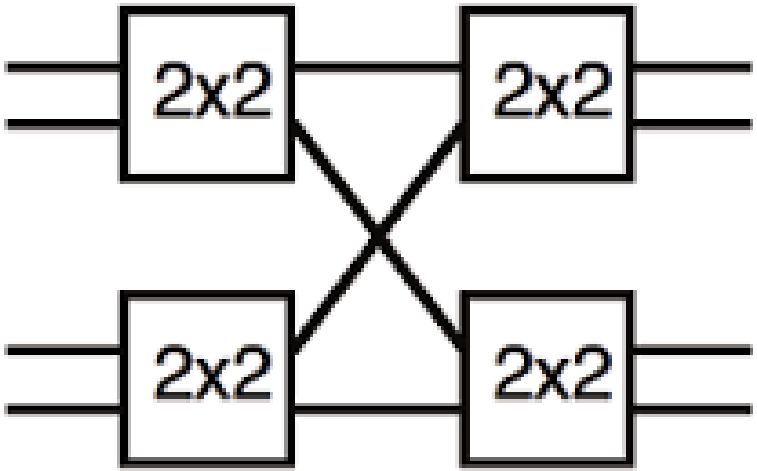
- Many different topologies have been proposed for multistage switching networks.
- One such topology is the omega switching network
- In a tightly coupled multiprocessor system, the source is a processor and the destination is a memory module.
- In a loosely coupled multiprocessor system, both the source and destination are processing elements.
- Some request patterns cannot be connected simultaneously.
- $\log_2 N$ stages and N/2 switches on each stage.

Figure 8.3: 8x8 omega switching network.



4x4 omega switching network.

4 x 4 switch example



Interconnection Structure

Hypercube

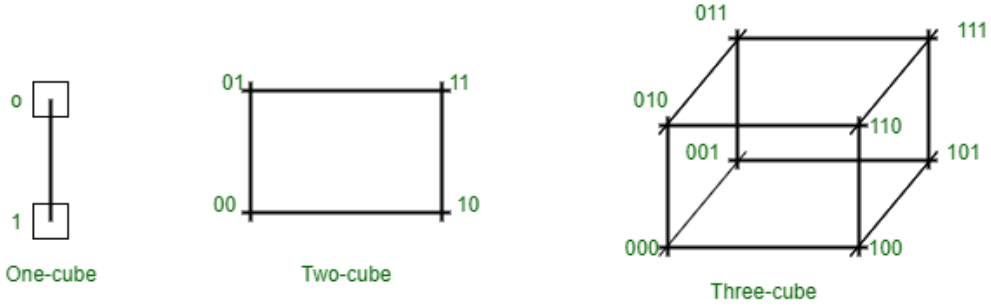


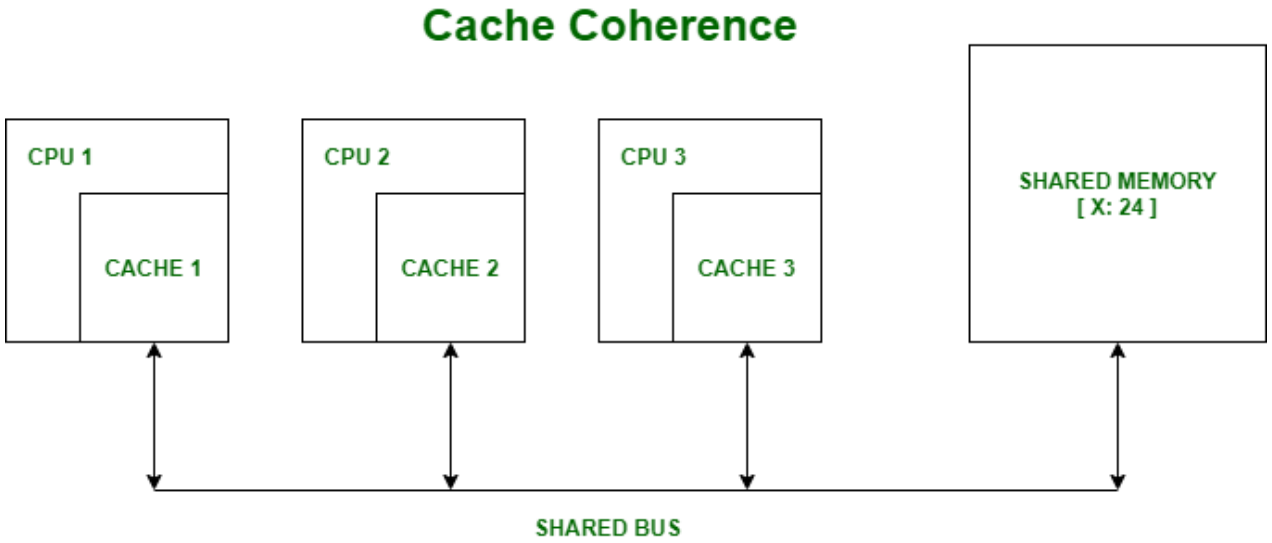
Figure - Hypercube Structures For n = 1,2,3

- The hypercube or binary n-cube multiprocessor structure is a loosely coupled system composed of N = 2" processors interconnected in an n - dimensional binary cube.
- Each processor forms a node of the cube. Although it is customary to refer to each node as having a processor, in effect it contains not only a CPU but also local memory and I/O interface.
- Each processor has direct communication paths to n other neighbor processors. These paths correspond to the edges of the cube. There are 2^n distinct n-bit binary addresses that can be assigned to the processors.

- Each processor address differs from that of each of its n neighbors by exactly one bit position.
- Routing messages through an n-cube structure may take from one to n links from a source node to a destination node.
  - A routing procedure can be developed by computing the exclusive-OR of the source node address with the destination node address.
  - The message is then sent along any one of the axes that the resulting binary value will have 1 bits corresponding to the axes on which the two nodes differ.
- A representative of the hypercube architecture is the Intel iPSC computer complex.
  - It consists of 128(n=7) microcomputers, each node consists of a CPU, a floating-point processor, local memory, and serial communication interface units.

Cache Coherence

- In a multiprocessor system, data inconsistency may occur among adjacent levels or within the same level of the memory hierarchy.
- In a shared memory multiprocessor with a separate cache memory for each processor, it is possible to have many copies of any one instruction operand: one copy in the main memory and one in each cache memory. When one copy of an operand is changed, the other copies of the operand must be changed also.
- Cache coherency is a situation where multiple processor cores share the same memory hierarchy, but have their own L1 data and instruction caches.
- Incorrect execution could occur if two or more copies of a given cache block exist, in two processors' caches, and one of these blocks is modified



Suppose there are three processors, each having cache. Suppose the following scenario:-
Processor 1 read X : obtains 24 from the memory and caches it.
Processor 2 read X : obtains 24 from memory and caches it.
Again, processor 1 writes as X : 64, Its locally cached copy is updated. Now, processor 3 reads X, what value should it get?
Memory and processor 2 thinks it is 24 and processor 1 thinks it is 64.

Cache coherence

- As multiple processors operate in parallel, and independently multiple caches may possess different copies of the same memory block, this creates a cache coherence problem.
- Cache coherence is the discipline that ensures that changes in the values of shared operands are propagated throughout the system in a timely fashion. There are three distinct level of cache coherence :-
- Every write operation appears to occur instantaneously.
- All processors see exactly the same sequence of changes of values for each separate operand.
- Different processors may see an operation and assume different sequences of values; this is known as non-coherent behavior.

- There are various Cache Coherence Protocols in multiprocessor system. These are:
- MSI protocol (Modified, Shared, Invalid)
- MOSI protocol (Modified, Owned, Shared, Invalid)
- MESI protocol (Modified, Exclusive, Shared, Invalid)
- MOESI protocol (Modified, Owned, Exclusive, Shared, Invalid)
- **Modified –** It means that the value in the cache is dirty, that is the value in current cache is different from the main memory.
- **Exclusive –** It means that the value present in the cache is same as that present in the main memory, that is the value is clean.
- **Shared –** It means that the cache value holds the most recent data copy and that is what shared among all the cache and main memory as well.
- **Owned –** It means that the current cache holds the block and is now the owner of that block, that is having all rights on that particular blocks.
- **Invalid –** This states that the current cache block itself is invalid and is required to be fetched from other cache or main memory

Coherency mechanisms

- **Directory-based –** In a directory-based system, the data being shared is placed in a common directory that maintains the coherence between caches. The directory acts as a filter through which the processor must ask permission to load an entry from the primary memory to its cache. When an entry is changed, the directory either updates or invalidates the other caches with that entry.
- **Snooping –** First introduced in 1983, snooping is a process where the individual caches monitor address lines for accesses to memory locations that they have cached. It is called a write invalidate protocol. When a write operation is observed to a location that a cache has a copy of and the cache controller invalidates its own copy of the snooped memory location.
- **Snarfing –** It is a mechanism where a cache controller watches both address and data in an attempt to update its own copy of a memory location when a second master modifies a location in main memory. When a write operation is observed to a location that a cache has a copy of the cache controller updates its own copy of the snarfed memory location with the new data.

END OF unit 9