Unit-10

Multiprocessors

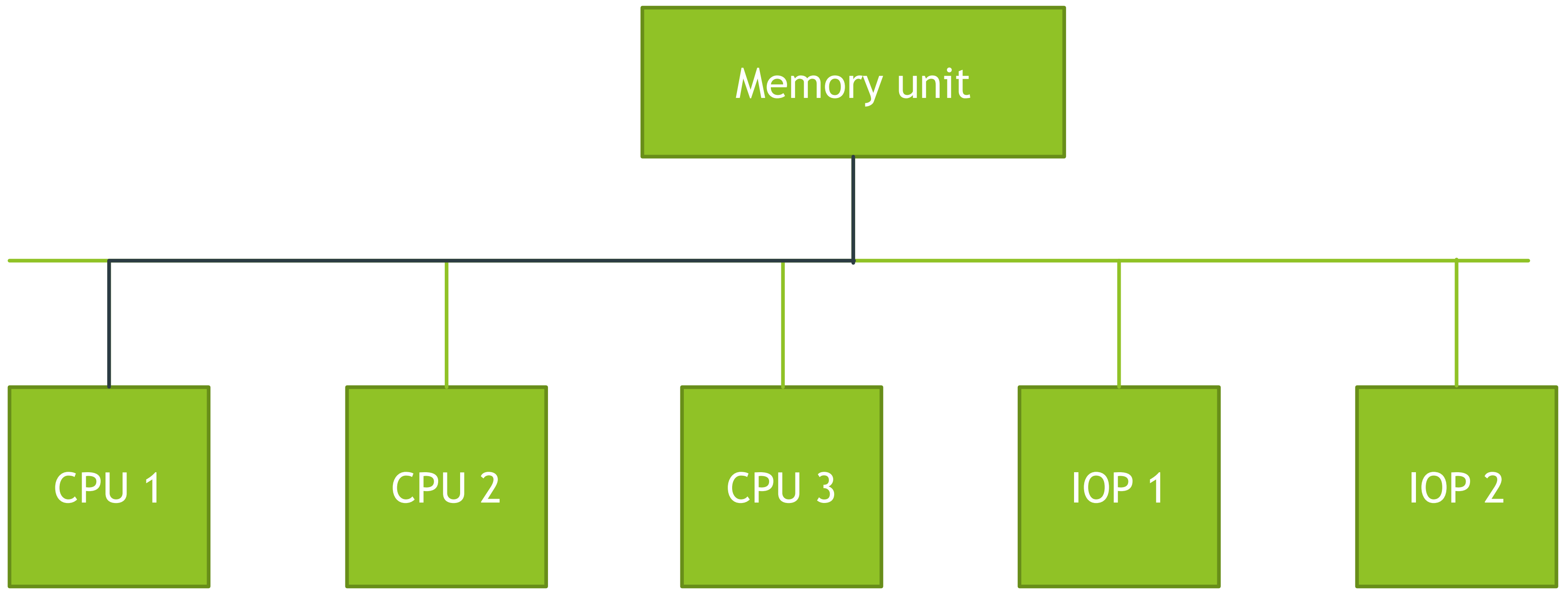
Tightly coupled V/S Loosely coupled

Tightly Coupled System	Loosely Coupled System
Tasks and/or processors communicate in a highly synchronized fashion.	Tasks or processors do not communicate in a synchronized fashion.
Communicates through a common shared memory.	Communicates by message passing packets.
Shared memory system.	Distributed memory system.
Overhead for data exchange is lower comparatively.	Overhead for data exchange is higher comparatively.

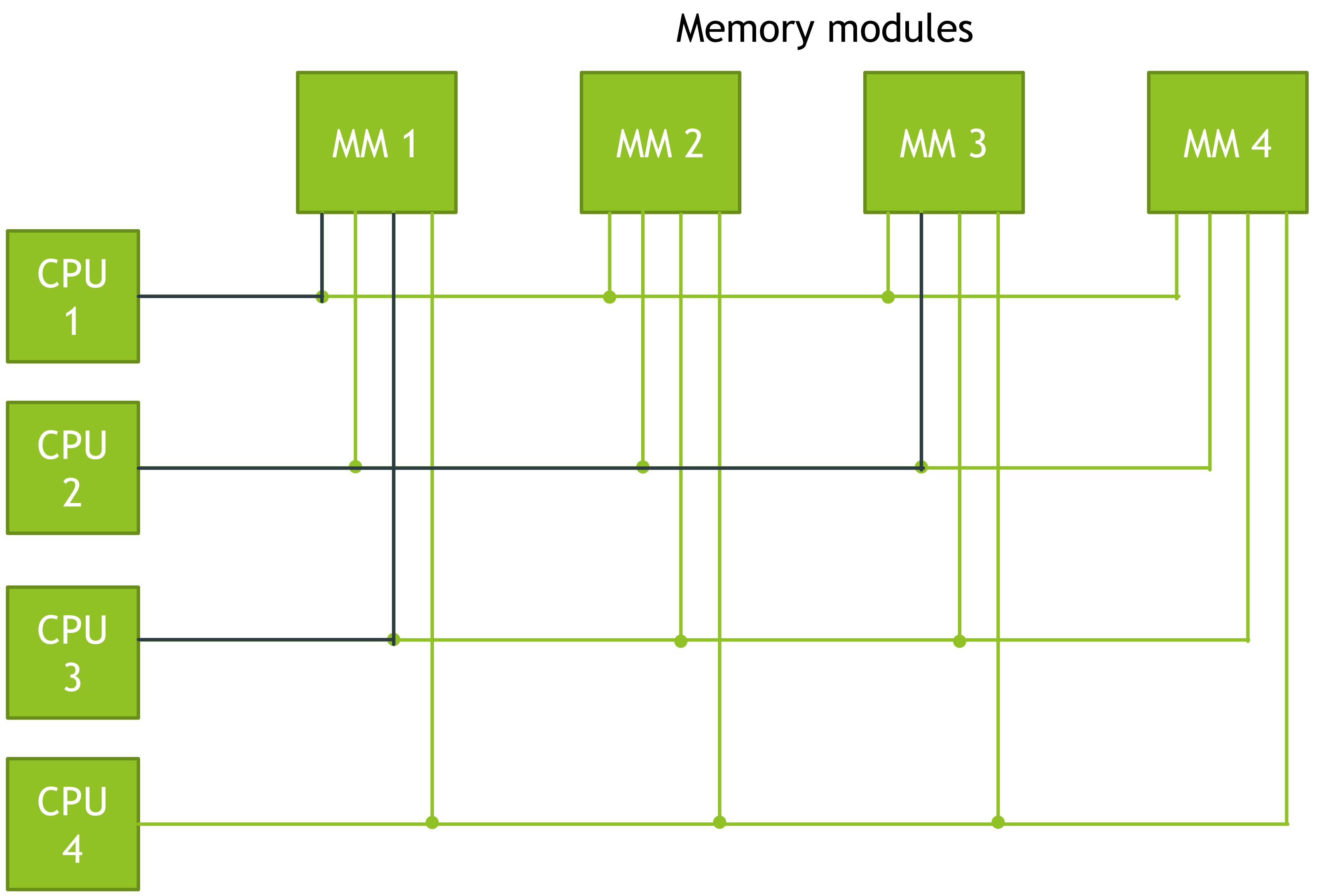
Interconnection Structures

1. Time-shared common bus
2. Multiport memory
3. Crossbar switch
4. Multistage switching network
5. Hypercube system

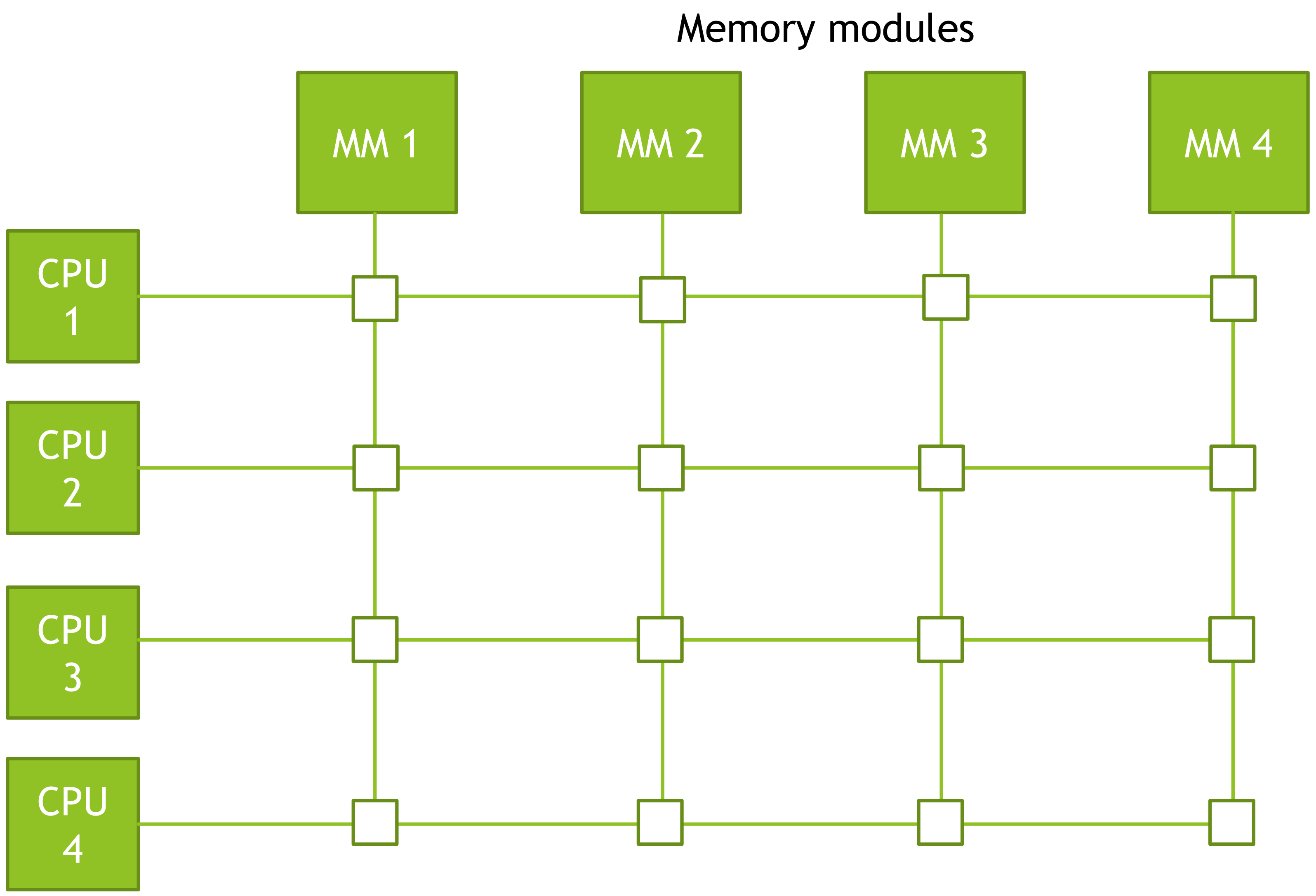
1. Time-shared common bus



2. Multiport Memory

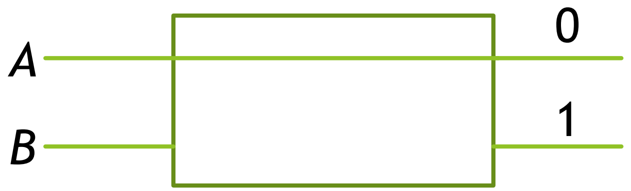


3. Crossbar switch

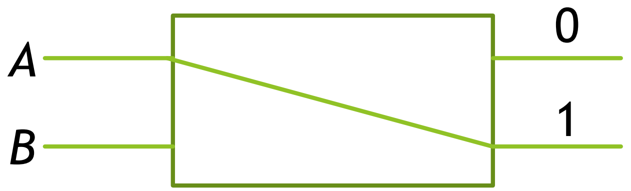


4. Multistage switching network

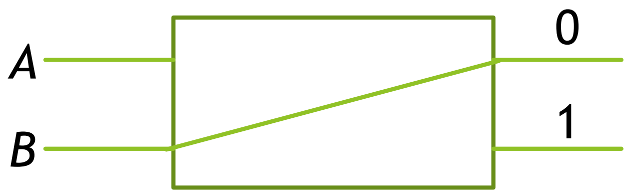
Operation of 2 X 2 interchange switch



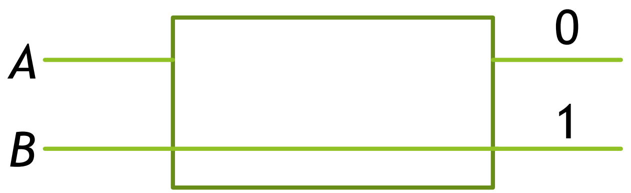
A connected to 0



A connected to 1

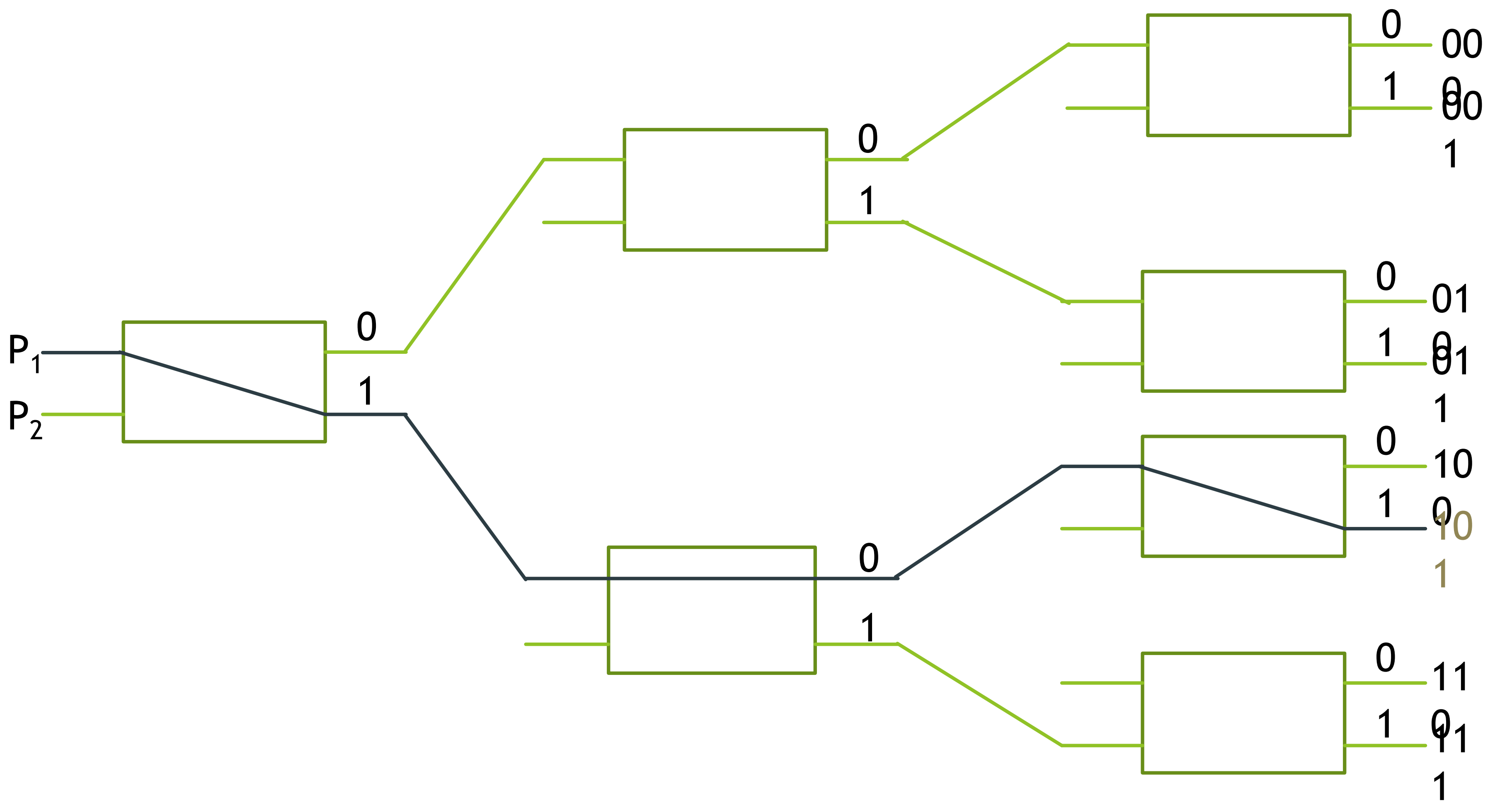


B connected to 0

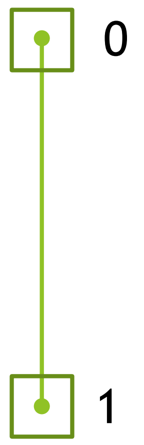


B connected to 1

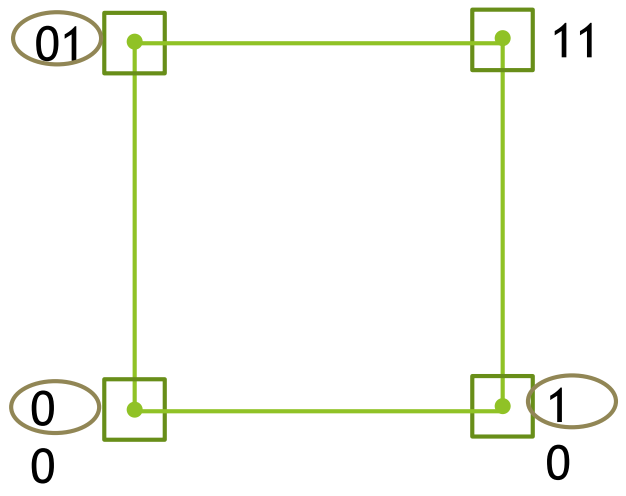
4. Multistage switching network



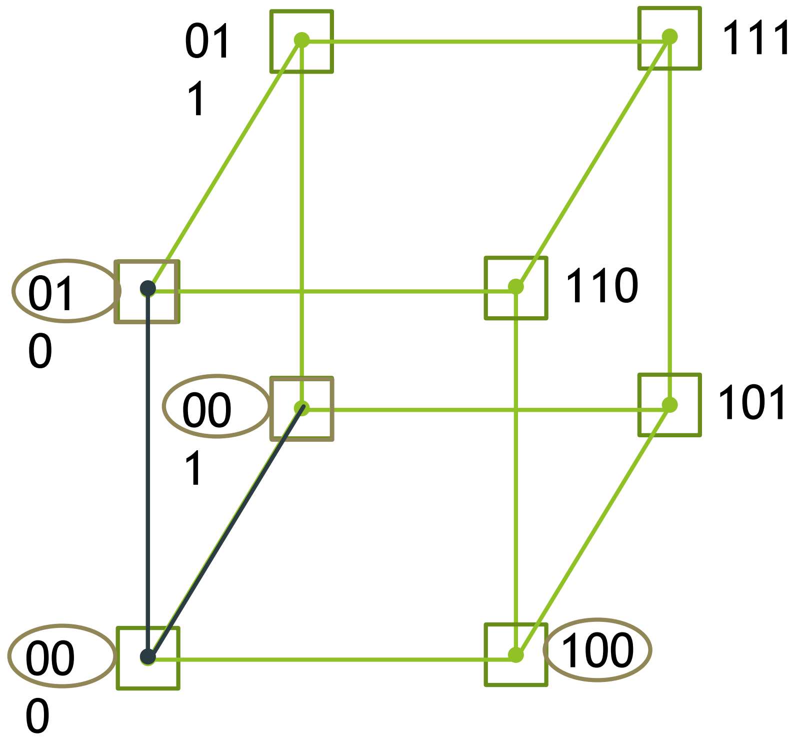
5. Hypercube Interconnection



One-cube



Two-cube



Three-
cube

010 x-or 001 =
011

Dynamic Arbitration Algorithm

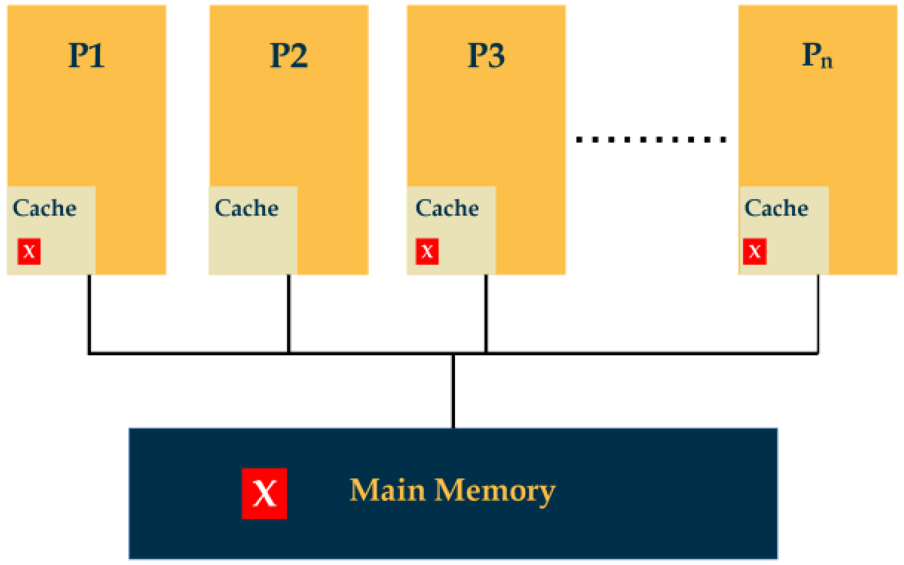
- Time Slice: The time slice algorithm allocates a fixed-length time slice of bus time that is offered sequentially to each processor, in round-robin fashion.
- Polling: In a bus system that uses polling, the bus grant signal is replaced by a set of lines called poll lines which are connected to all units.
- LRU: The Least Recently Used (LRU) algorithm gives the highest priority to the requesting device that has not used the bus for the longest interval.
- FIFO: In first-come, first-serve scheme, requests are served in the order received.
- Rotating daisy-chain: The rotating daisy-chain procedure is a dynamic extension of the daisy-chain algorithm. In this scheme there is no central bus-controller, and the priority line is connected from the priority-out of the last device back to the priority-in of the first device in a closed loop.

Cache Coherence Problem

- A cache coherence issue results from the concurrent operation of several processors and the possibility that various caches may hold different versions of the identical memory block. The practice of cache coherence makes sure that alterations in the contents of associated operands are quickly transmitted across the system.

Cache coherence has three different levels:

1. Each writing operation seems to happen instantly.
2. Each operand's value changes are seen in every processor in precisely the same order.
3. Non-coherent behavior results from many processors interpreting the same action in various ways.



Scenario when processors shared data block and the data block is consistent

Methods to resolve Cache Coherence

The two methods listed below can be used to resolve the cache coherence issue:

•Write Through

•Write Back

- **Write Through**
- The easiest and most popular method is to write through. Every memory write operation updates the main memory. If the word is present in the cache memory at the requested address, the cache memory is also updated simultaneously with the main memory.
- The benefit of this approach is that the RAM and cache always hold the same information. In systems with direct memory access transfer, this quality is crucial. It makes sure the information in the main memory is up-to-date at all times so that a device interacting over DMA can access the most recent information.
- ***Advantage*** - It provides the highest level of consistency.
- ***Disadvantage*** - It requires a greater number of memory access.

- **Write Back**
- Only the catch location is changed during a write operation in this approach. When the word is withdrawn from the cache, the place is flagged, so it is replicated in the main memory. The right-back approach was developed because words may be updated numerous times while they are in the cache. However, as long as they are still there, it doesn't matter whether the copy that is stored in the main memory is outdated because requests for words are fulfilled from the cache.
- An accurate copy must only be transferred back to the main memory when the word is separated from the cache. According to the analytical findings, between 10% and 30% of all memory references in a normal program are written into memory.
- ***Advantage*** - A very small number of memory accesses and write operations.
- ***Disadvantage*** - Inconsistency may occur in this approach.

- ***Modified*** - The modified term signifies that the data stored in the cache and main memory are different. This means the data in the cache has been modified, and the changes need to be reflected in the main memory.
- ***Exclusive*** - The exclusive term signifies that the data is clean, i.e., the cache and the main memory hold identical data.
- ***Shared*** - Shared refers to the fact that the cache value contains the most current data copy, which is then shared across the whole cache as well as main memory.
- ***Owned*** - The owned term indicates that the block is currently held by the cache and that it has acquired ownership of it, i.e., complete privileges to that specific block.
- ***Invalid*** - When a cache block is marked as invalid, it means that it needs to be fetched from another cache or main memory.

Different Cache Coherence Protocols used in multiprocessor systems:

- MSI protocol (Modified, Shared, Invalid)
- MOSI protocol (Modified, Owned, Shared, Invalid)
- MESI protocol (Modified, Exclusive, Shared, Invalid)
- MOESI protocol (Modified, Owned, Exclusive, Shared, Invalid)

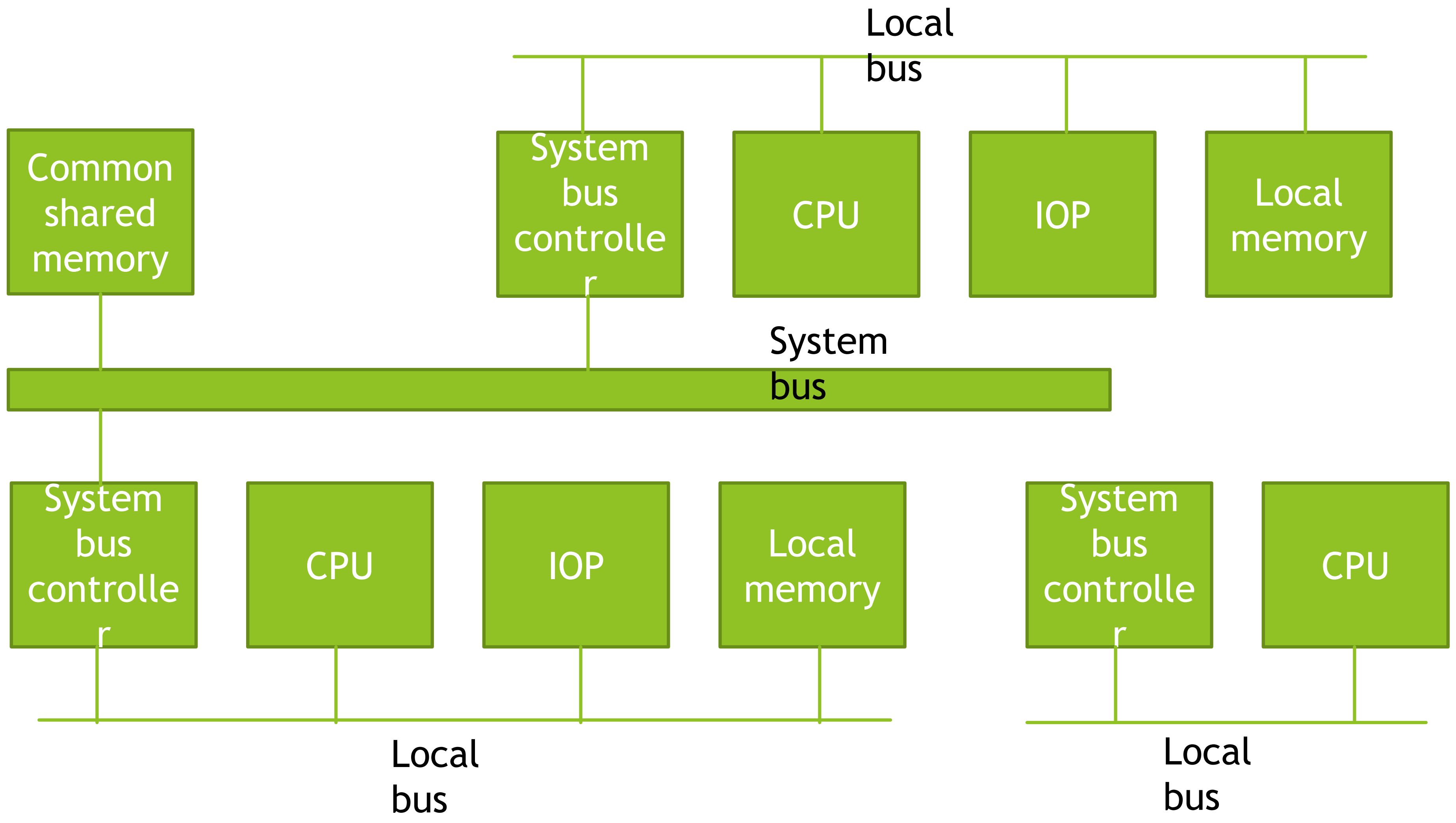
Types of Coherence:

- ***Directory Based*** - A directory-based system keeps the coherence amongst caches by storing shared data in a single directory. In order to load an entry from primary memory into its cache, the processor must request permission through the directory, which serves as a filter. The directory either upgrades or devalues the other caches that contain that record when a record is modified.
- ***Snooping*** - Individual caches watch address lines during the snooping/monitoring process to look for accesses to memory locations that they have cached. A write invalidate protocol is what it is known as. When a write activity is seen to a memory address for which a cache maintains a copy, the cache controller invalidates its own copy of the snooped memory location.
- ***Snarfing*** - A cache controller uses this approach to try and update its own copy of a memory location when a second master alters a place in the main memory by keeping an eye on both the address and the contents. The cache controller updates its own copy of the underlying memory location with the new data when a write action is detected to a place of which a cache holds a copy.

Interprocessor Communication and Synchronization

- The various processors in a multiprocessor system must be provided with a facility for communicating with each other.
- A communication path can be established through common input-Output channels.
- In a shared memory multiprocessor system, the most common procedure is to set aside a portion of memory that is accessible to all processors. The primary use of the common memory is to act as a message center similar to a mailbox, where each processor can leave messages for other processors and pick up messages intended for it
- The sending processor structures a request, a message, or a procedure, and places it in the memory mailbox. Status bits residing in common memory are generally used to indicate the condition of the mailbox, whether it has meaningful information, and for which processor it is intended.
- The receiving processor can check the mailbox periodically to determine if there are valid messages for it. The response time of this procedure can be time consuming since a processor will recognize a request only when polling messages. A more efficient procedure is for the sending processor to alert the receiving processor directly by means of an interrupt signal.

Shared Memory Architecture



Interprocessor Synchronization

- The instruction set of a multiprocessor contains basic instructions that are used to implement communication and synchronization between cooperating processes. Communication refers to the exchange of data between different processes.
- For example, parameters passed to a procedure in a different processor constitute interprocessor communication. Synchronization refers to the special case where the data used to communicate between processors is control information. Synchronization is needed to enforce the correct sequence of processes and to ensure mutually exclusive access to shared writable data.

Mutual Exclusion with a Semaphore

- A properly functioning multiprocessor system must provide a mechanism that will guarantee orderly access to shared memory and other shared resources. This is necessary to protect data from being changed simultaneously by two or more processors. This mechanism has been termed mutual exclusion.
- Mutual exclusion must be provided in a multiprocessor system to enable one processor to exclude or lock out access to a shared resource by other processors when it is in a critical section. A critical section is a program sequence that, once begun, must complete execution before another processor accesses the same shared resource.
- A binary variable called a semaphore is often used to indicate whether or not a processor is executing a critical section. A semaphore is a software-controlled flag that is stored in a memory location that all processors can access. When the semaphore is equal to 1, it means that a processor is executing a critical program, so that the shared memory is not available to other processors. When the semaphore is equal to 0, the shared memory is available to any requesting processor. Processors that share the same memory segment agree by convention not to use the memory segment unless the semaphore is equal to 0, indicating that memory is available.
- A semaphore can be initialized by means of a test and set instruction in conjunction with a hardware lock mechanism. A hardware lock is a processor-generated signal that serves to prevent other processors from using the system bus as long as the signal is active. The test-and-set instruction tests and sets a semaphore and activates the lock mechanism during the time that the instruction is being executed. This prevents other processors from changing the semaphore between the time that the processor is testing it and the time that it is setting it

Questions asked in GTU exam

1. Draw and explain shared memory architecture of multiprocessor system
2. Explain Inter-process communication
3. Explain any two interconnection structures that make it possible to form a multiprocessor system with diagram.
4. Differentiate tightly coupled and loosely coupled systems.
5. Give the features of a multiprocessor system.
6. What is cache coherence?