UNIT 5

Pipelining: Introduction, Arithmetic Pipeline, Instruction Pipeline, RISC Pipeline, Vector Processing, Array Processors, Hazards.

Multiprocessors: Characteristics of Multiprocessors, Interconnection Structures, Inter processor arbitration, Inter processor communication and synchronization, Cache coherence.

Characteristics of Multiprocessors

- A multiprocessor system is an interconnection of two or more CPUs with memory and I/O equipment.
- IOPs are generally not included in the definition of multiprocessor system unless they have computational facilities comparable to CPUs.
- Multiprocessor are MIMD system.
- Multicomputer system includes number of computers connected together by means of communication lines.

- It improves reliability.

- If one system fails, the whole system continue to function with perhaps low efficiency.

- The computation can proceed in parallel in two ways:
  - Multiple independent jobs operate in parallel
  - A single job can be partitioned into multiple parallel tasks

- Multiprocessor are classified by the way their memory is organized:

- Shared memory or tightly coupled multiprocessor
- Most commercial tightly coupled multiprocessors provide a cache memory with each CPU.

- In addition, there is a global common memory that all CPUs can access.

- Information can therefore be shared among the CPUs by placing it in the common global memory.

- Distributed memory or loosely coupled system
  - Each processor element in a loosely coupled system has its own private local memory.
  - The processors are tied together by a switching scheme designed to route information from one processor to another through a message-passing scheme.
  - Loosely coupled systems are most efficient when the interaction between tasks is minimal.
  - Tightly coupled systems can tolerate a higher degree of interaction between tasks.

Interconnection Structures

- The components that forms the multiprocessor system are:
  - CPUs
  - IOPs
  - I/O devices
  - Memory
- Physical forms available for establishing an interconnection network are:
  - Time Shared Common Bus
  - Multiport Memory

- Crossbar Switch
- Multistage switching network
- Hypercube System

Time Shared Common Bus

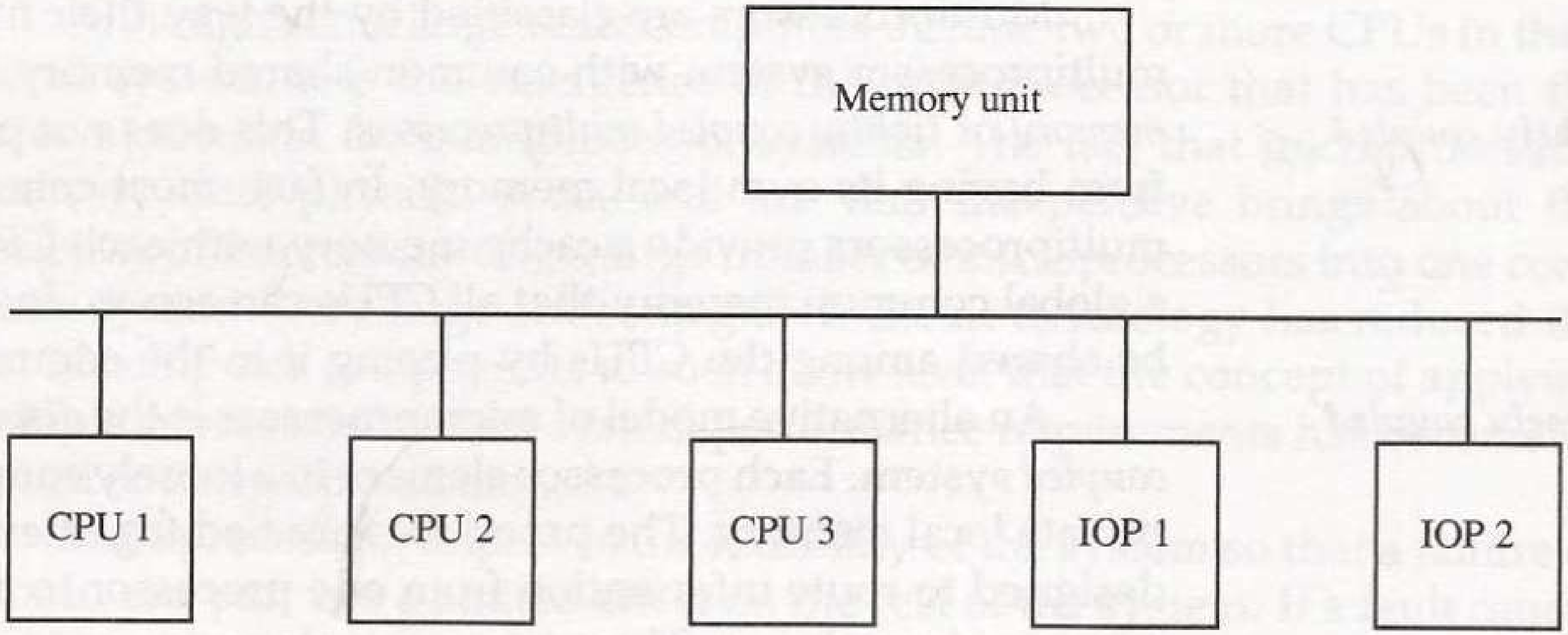


Figure Time-shared common bus organization.

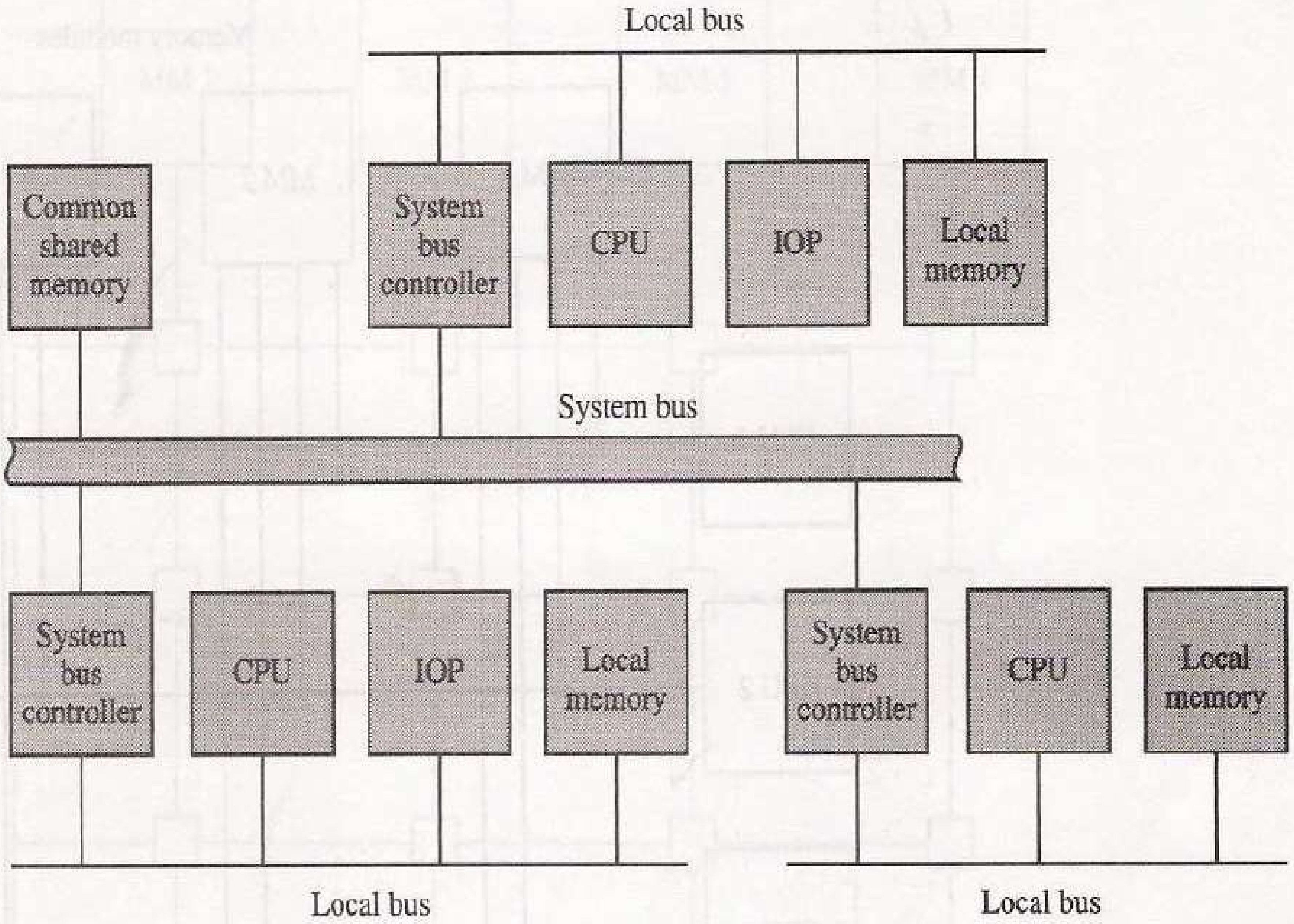


Figure System bus structure for multiprocessors.

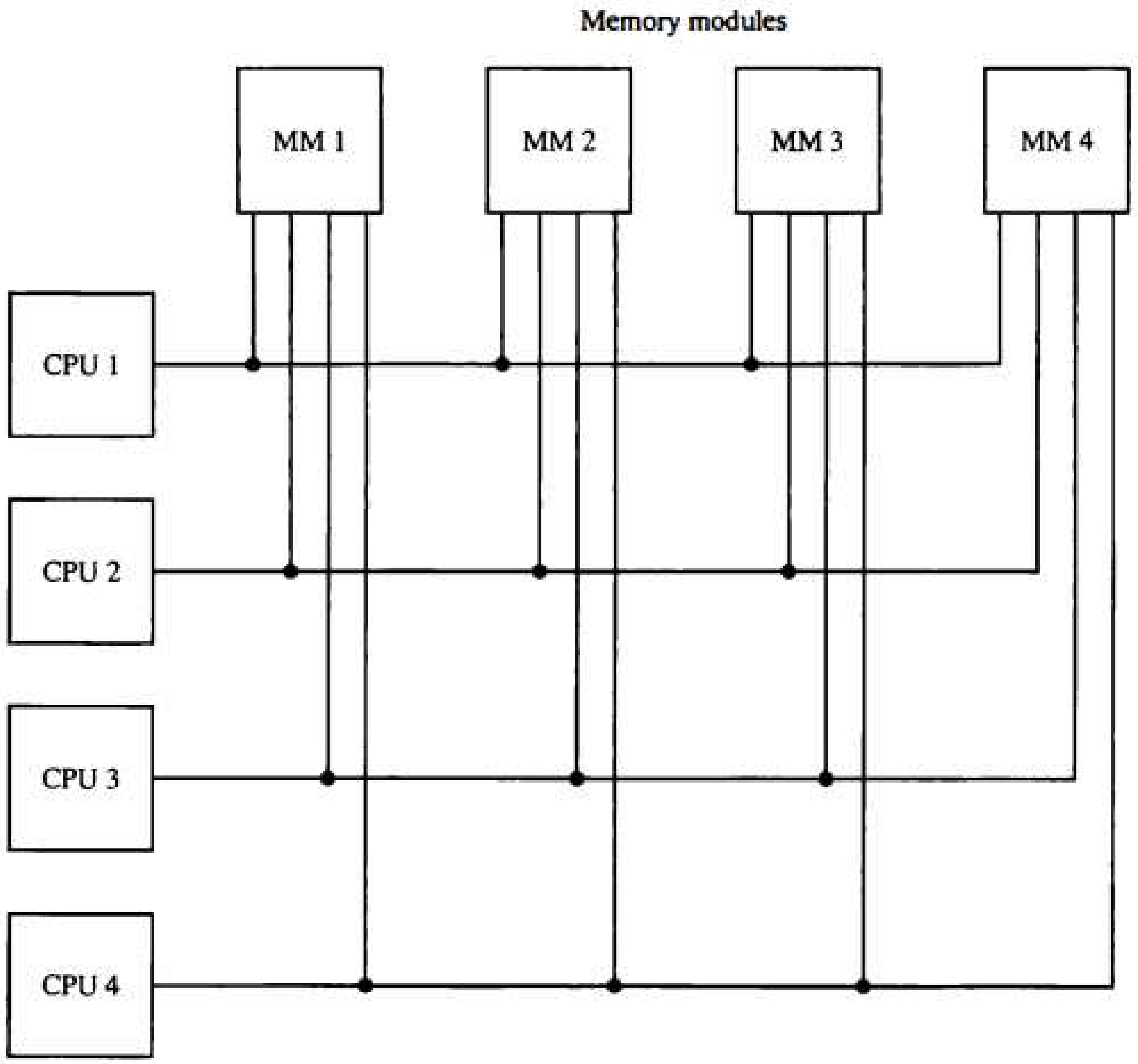


Figure 13-3 Multiport memory organization.

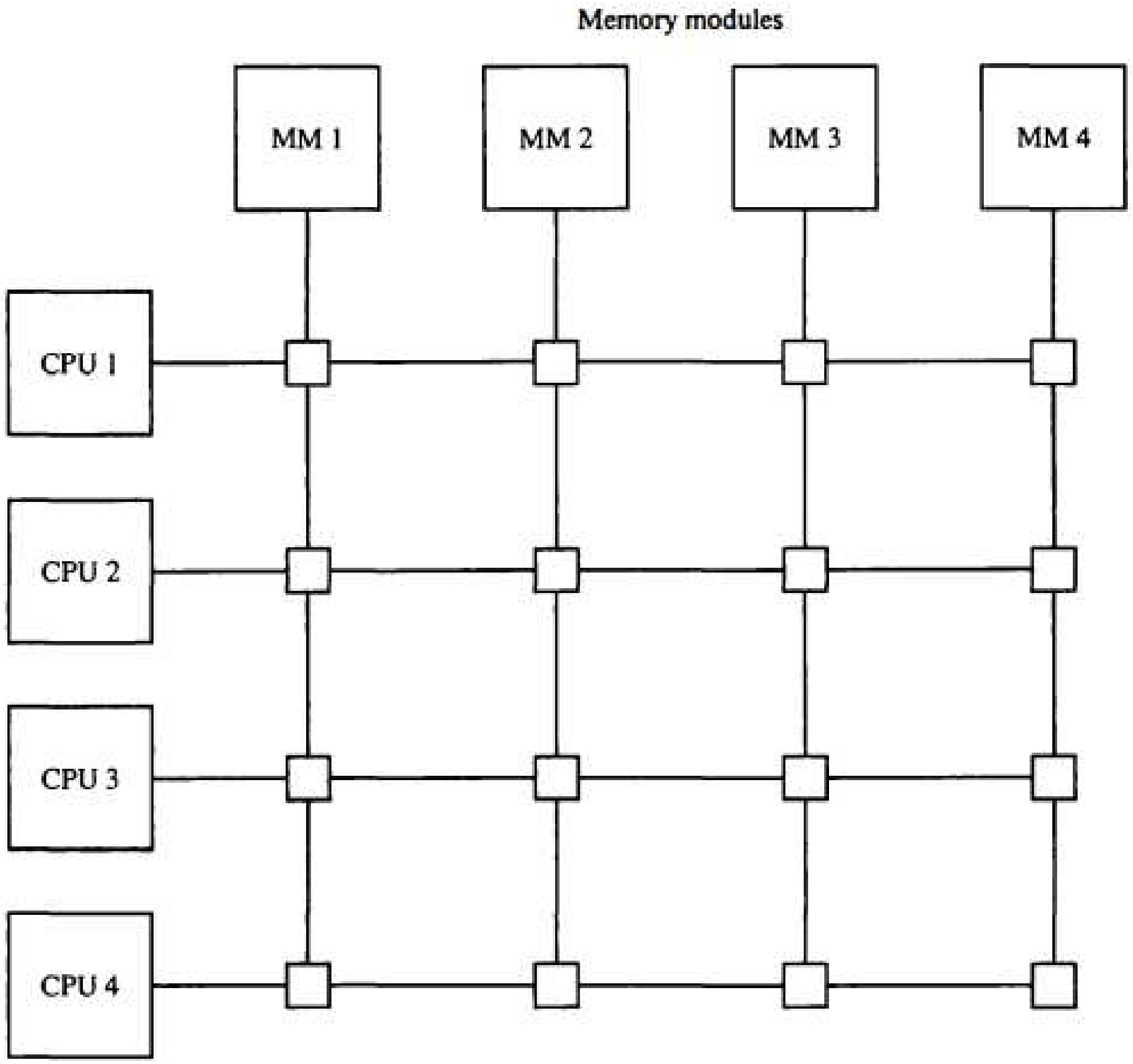
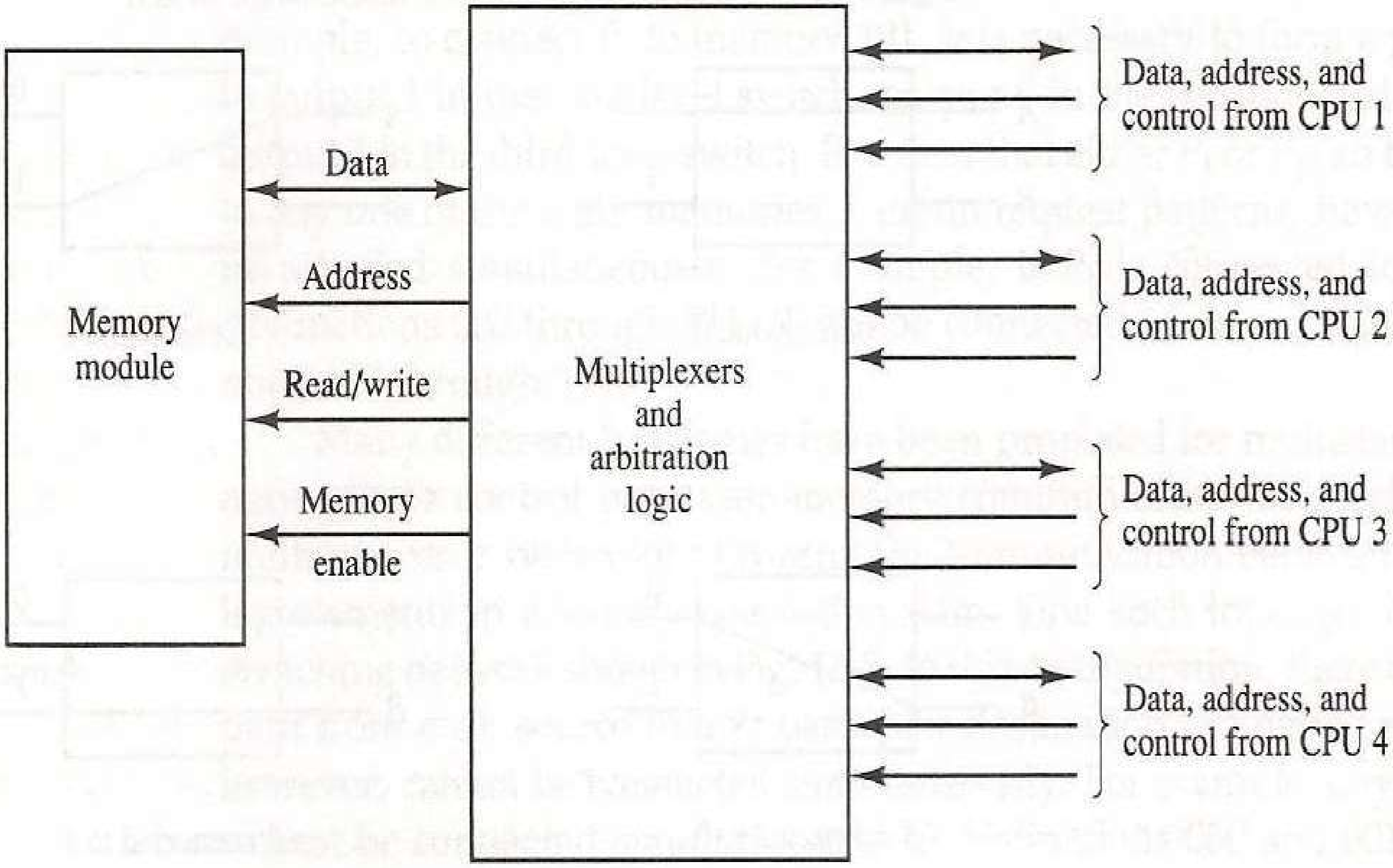


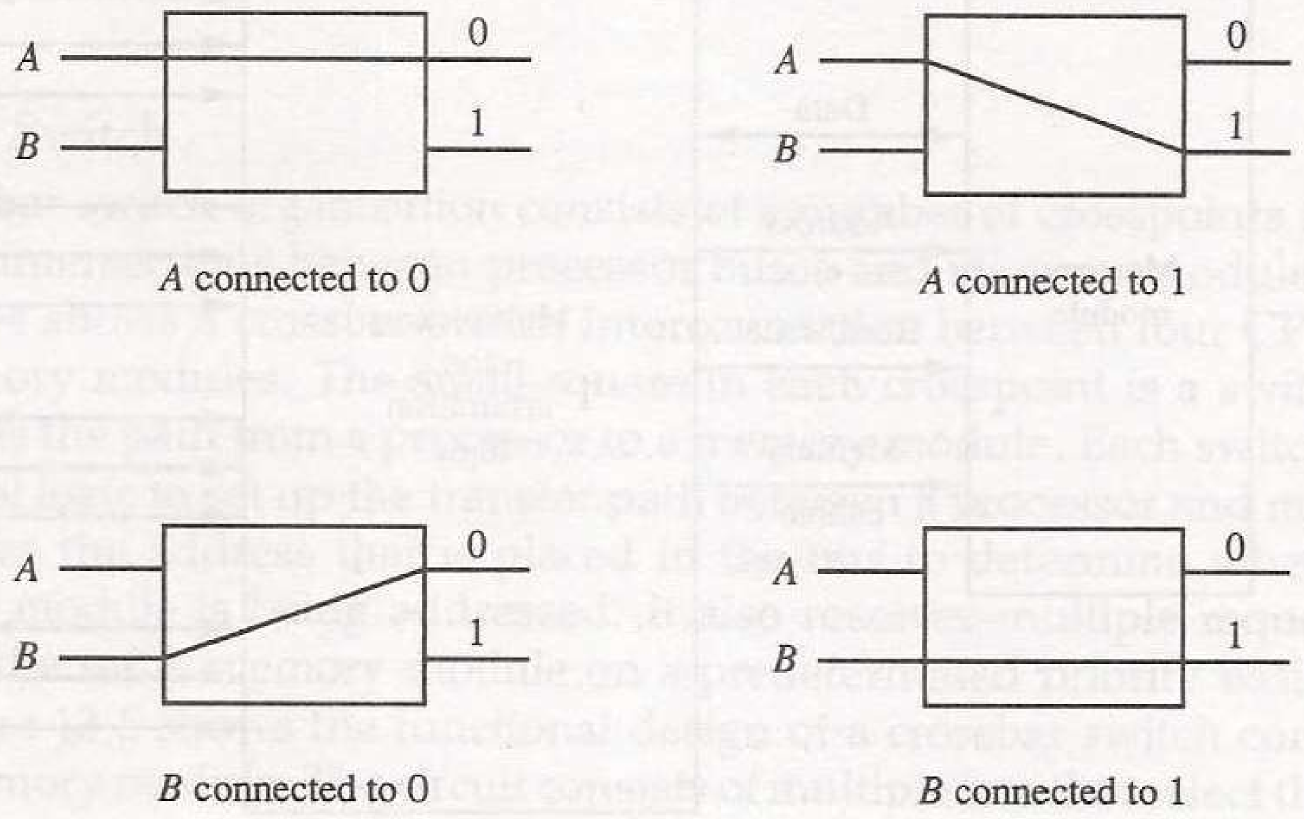
Figure 13-4 Crossbar switch.

Figure Block diagram of crossbar switch.


Memory module
Data
Address
Read/write
Memory enable
Multiplexers and arbitration logic
Data, address, and control from CPU 1
Data, address, and control from CPU 2
Data, address, and control from CPU 3
Data, address, and control from CPU 4

Multistage Switching Network

Figure Operation of a 2×2 interchange switch.



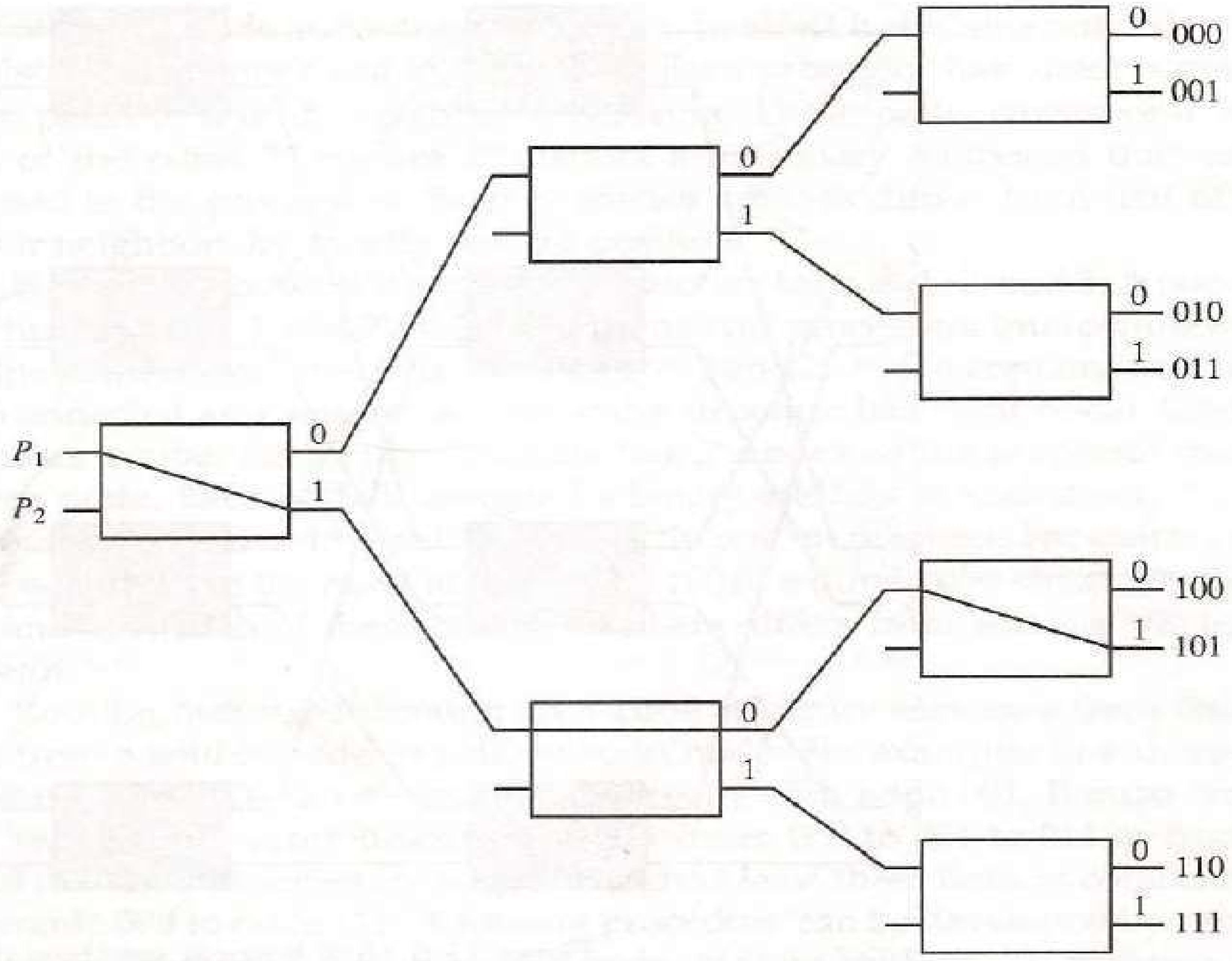


Figure Binary tree with 2 × 2 switches.

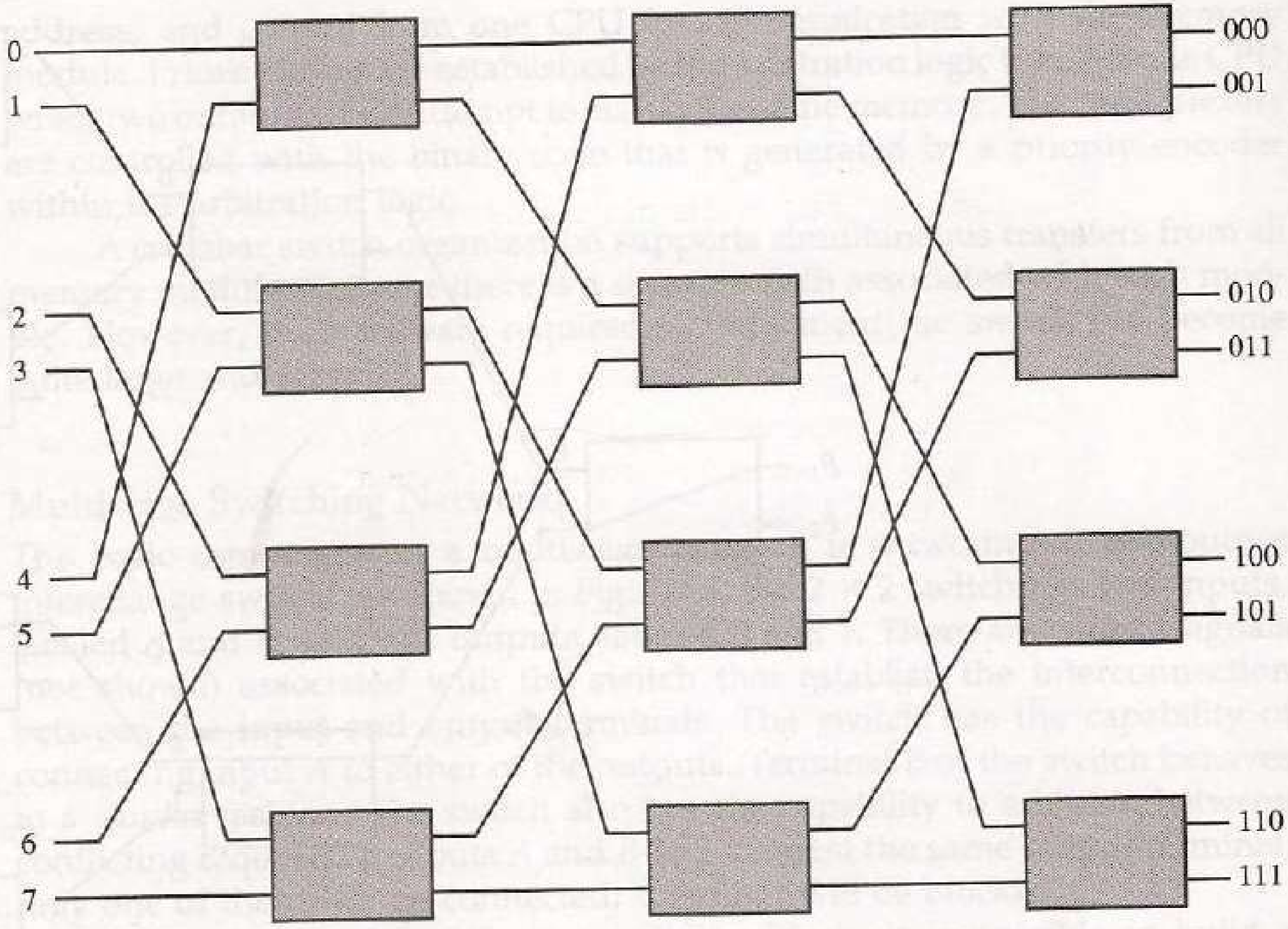


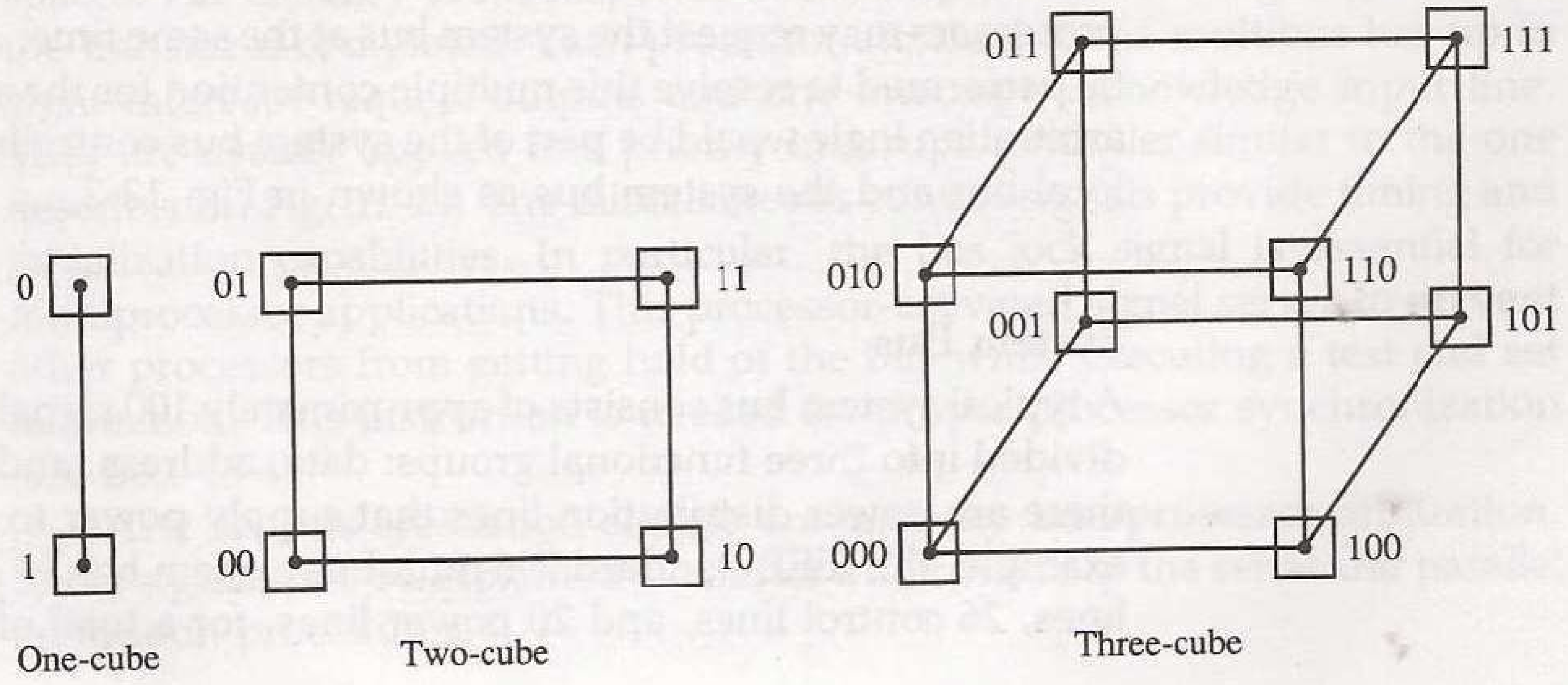
Figure 8 × 8 omega switching network.

Hypercube Interconnection

- Hypercube or binary n-cube multiprocessor structure is a loosely coupled system.
- It composed of N=2^n processors interconnected in n-dimensional binary cube.
- Each processor form the node of the cube.
- Each processor has direct communication path with n other neighbor processors.
- There are 2^n distinct n-bit binary address that can be assigned to each processor.

Hypercube Connection

Figure Hypercube structures for n = 1,2,3.



Interprocessor Arbitration

- Computer system contain a number of buses at various levels to facilitate the transfer of information.

- A bus that connects major components in a multiprocessor system such as CPU, IOP and memory, is called system bus.

- Arbitration logic is the part of system bus controller placed between local bus and system bus that resolve the multiple contention for shared resources.

System Bus

- A typical system bus consists of approximately 100 signal lines.
- These lines are divided into three functional groups: data, address and control.
- In addition there are power distribution lines that supply power to the components.
- Example IEEE standard 796 multi bus system has 16 data lines, 24 address lines, 26 control lines and 20 power lines for total of 86 lines.

- Data lines provide a path for the transfer of data between processor and common memory.

- The number of data lines are usually multiple of 8, with 16 and 32 being most common.

- Address lines are used to indentify memory location or any other source and destination units.

- The number of address lines determine the maximum possible memory capacity in the system.

- The control lines provides signals for controlling information transfer between units.

- Timing signals indicate validity of data and address.
- Command signals specify the operations to be performed.
- Data transfer on the system bus can be
  - Synchronous
  - In a synchronous bus, each data item is transferred during a time slice known in advance to both source and destination units. Synchronization is achieved by driving both units from a common clock source.

- Asynchronous

- In an asynchronous bus, each data item being transferred is accompanied by handshaking control signals to indicate when the data are transferred from the source and received by the destination.

TABLE 13-1 IEEE Standard 796 Multibus Signals

	Signal name
Data and address	
Data lines (16 lines)	DATA0–DATA15
Address lines (24 lines)	ADRS0–ADRS23
Data transfer	
Memory read	MRDC
Memory write	MWTC
IO read	IORC
IO write	IOWC
Transfer acknowledge	TACK
Interrupt control	
Interrupt request (8 lines)	INT0–INT7
Interrupt acknowledge	INTA
Miscellaneous control	
Master clock	CCLK
System initialization	INIT
Byte high enable	BHEN
Memory inhibit (2 lines)	INH1–INH2
Bus lock	LOCK
Bus arbitration	
Bus request	BREQ
Common bus request	CBRQ
Bus busy	BUSY
Bus clock	BCLK
Bus priority in	BPRN
Bus priority out	BPRO
Power and ground (20 lines)	

Arbitration Procedures

- Arbitration procedures service all processor requests on the basis of established priorities.
- It can be implemented in two ways:
  - Serial Connection of Units
  - Parallel Connection of Units

Serial Arbitration Procedure

- It is obtained from daisy chain connection of bus arbitration circuit similar to the case of priority interrupt.
- Each processor has its own arbiter logic which are arranged according to the priorities from highest to the lowest.

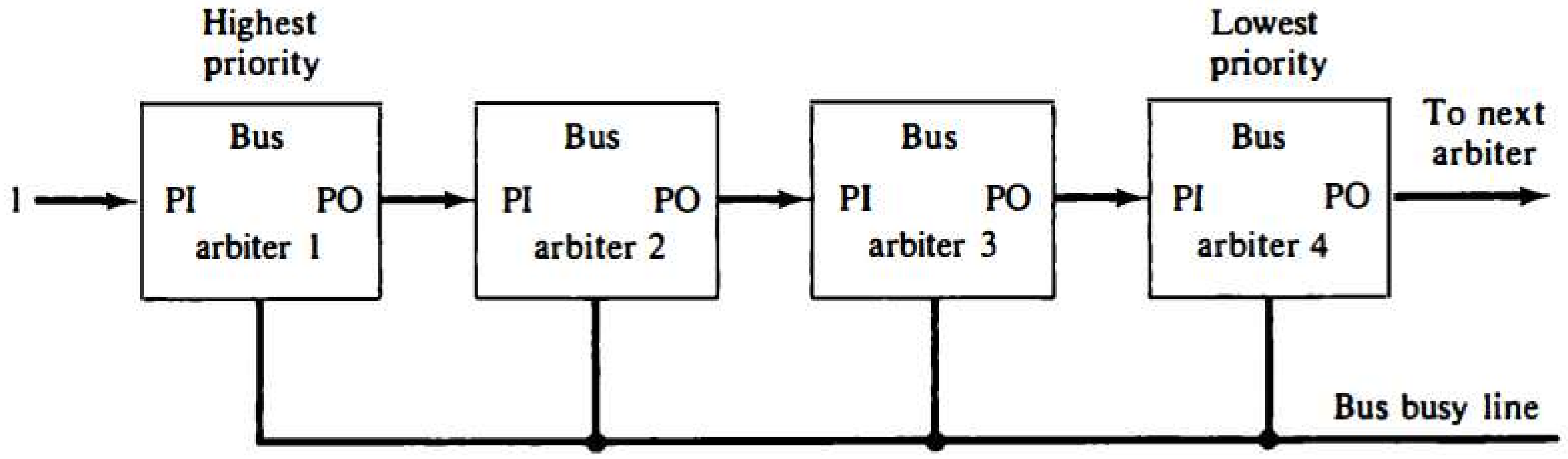


Figure 13-10 Serial (daisy-chain) arbitration.

Parallel Arbitration Logic

- It uses an external priority encoder and decoder.
- Each bus arbiter has a bus request output lines and a bus acknowledge input lines.
- Each arbiter enables request lines when its processor is requesting the system bus.
- The one with highest priority determine by the output of the decoder get access to the bus.

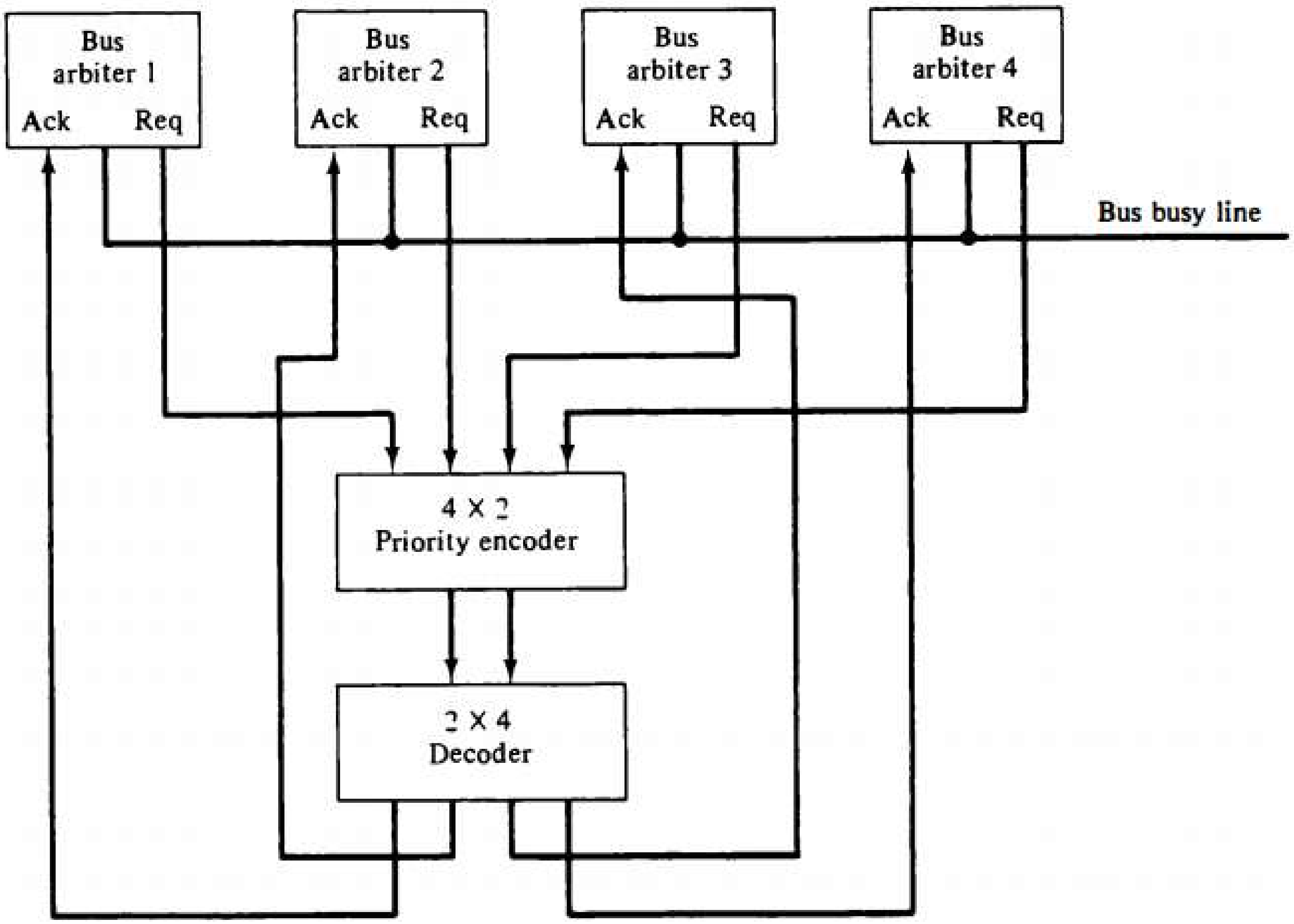


Figure 13-11 Parallel arbitration.

Dynamic Arbitration Algorithms

- Serial and parallel bus arbitration are static since the priorities assigned is fixed.
- In dynamic arbitration priorities of the system changes while the system is in operation.
- The various algorithms used are:
  - Time Slice
  - Polling
  - Least Recently Used(LRU)
  - First In First Out(FIFO)
  - Rotating Daisy-Chain

Inter processor Communication

- All the processors in the multi processor system need to communicate with each other.

- The most common way is to set aside portion of memory that is accessible to all processors.

- Sending processor puts the data and the address of the receiving processor in the memory.

- All the processors periodically check the memory for any information.
- If they find their address they read the data.
- This procedure is time consuming.

- Another procedure is to send the interrupt signal to the receiving processor whenever the sending processor leaves the message.

- In addition to shared memory, multiprocessor system may have other shared resources.

- To prevent the conflicting use of shared resources by several processors there must be provision for assigning resources to processor.

- This task is handled by the Operating System.

- There are three organizations that have been used in the design of OS for multiprocessors:
  - Master-Slave Configuration
  - Separate OS
  - Distributed OS

- In master slave configuration, one processor designated as master execute OS function. If slave processor needs OS service, it must request to the master processor.

- In separate OS configuration, each processor can execute the OS routines it needed.

- This organization is suited for loosely coupled system where every processor have its own copy of the entire OS.

- In distributed OS, the OS routines are distributed among the processors.

- Each particular OS function is assigned to only one processor at a time.

- Since the routines float from one processor to another it is also called floating OS.

Inter processor Synchronization

- Synchronization is special case of communication where data used to communicate between processors is control information.
- It is needed to enforce correct sequence of processes and to ensure mutually exclusive access to shared writable data.
- A number of hardware mechanisms for mutual exclusion have been developed. One of the most popular is through the use of binary semaphore.

Mutual Exclusion with a Semaphore

- A proper functioning multi-processor should guarantee orderly access to shared resources. So that data can not be changed simultaneously by two processors.

- This is called mutual exclusion.

- Mutual exclusion must be provided in multi processor to enable one processor to lock out access to shared resources once it entered in critical section.

- Critical section is a program sequence, that must be completed once begun.

- Binary variable semaphore is used to indicate whether the processor is in critical section.
- Semaphore is software controlled flag stored in common memory.
- If it is 1, processor is executing critical section. If it is 0, it indicate the shared memory is available to any requesting processor.
- Testing and setting semaphore is itself a critical task and must be performed in single indivisible task.

- A semaphore can be initialized by means of test and set instruction in conjunction with a hardware lock mechanism.

- A hardware lock is processor generated signal that prevent other processors from using system bus as long as signal is active.

- Test and set instruction tests and sets semaphore and activates lock mechanism during the time that it is being executed.

Assume that the semaphore is a bit in the least significant position of a memory word whose address is symbolized by SEM. Let the mnemonic TSL designate the "test and set while locked" operation. The instruction

```
TSL   SEM
```

will be executed in two memory cycles (the first to read and the second to write) without interference as follows:

$R \leftarrow M[SEM]$	Test semaphore
$M[SEM] \leftarrow 1$	Set semaphore

- The last instruction in the program must be clear location SEM to ‘zero’ to release the shared resource to other processors.

- Lock signal must be active during execution of test and set instruction.

- It does not have to be active once semaphore is set.

Cache Coherence

- In shared memory multi-processor system, processors share memory and they have local memory(part or all of which is cache).

- To ensure the ability of the system to execute memory operations correctly, multiple copies of the data must be identical. This requirement imposes which is called cache coherence problem.

Conditions for Incoherence

- This condition arise when the processor need to share the writable data.
- In both policies write back and write through incoherence condition is created.
- In case of DMA also, IOP modify the data in main memory which reside in the cache and can’t be updated.

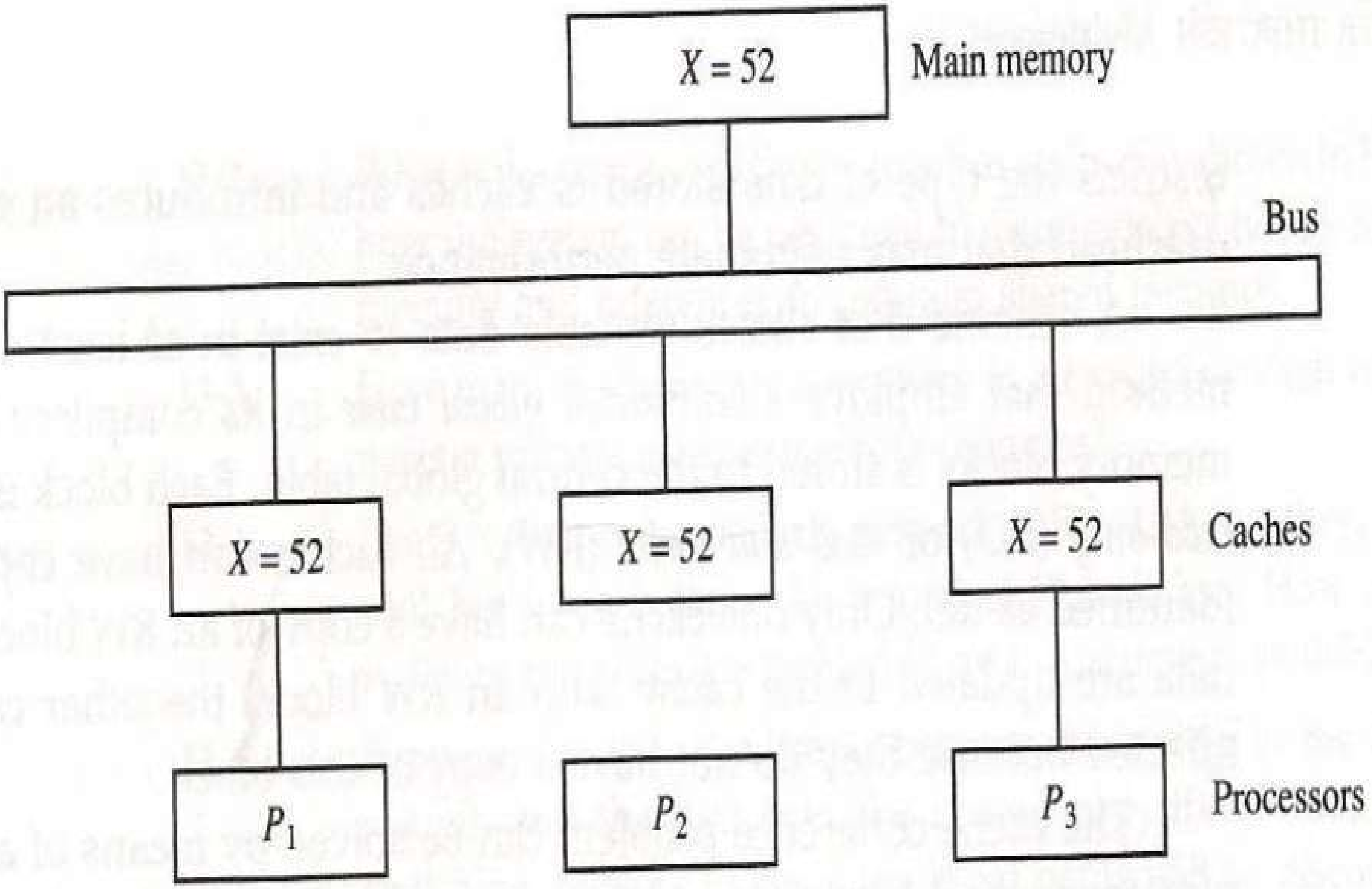
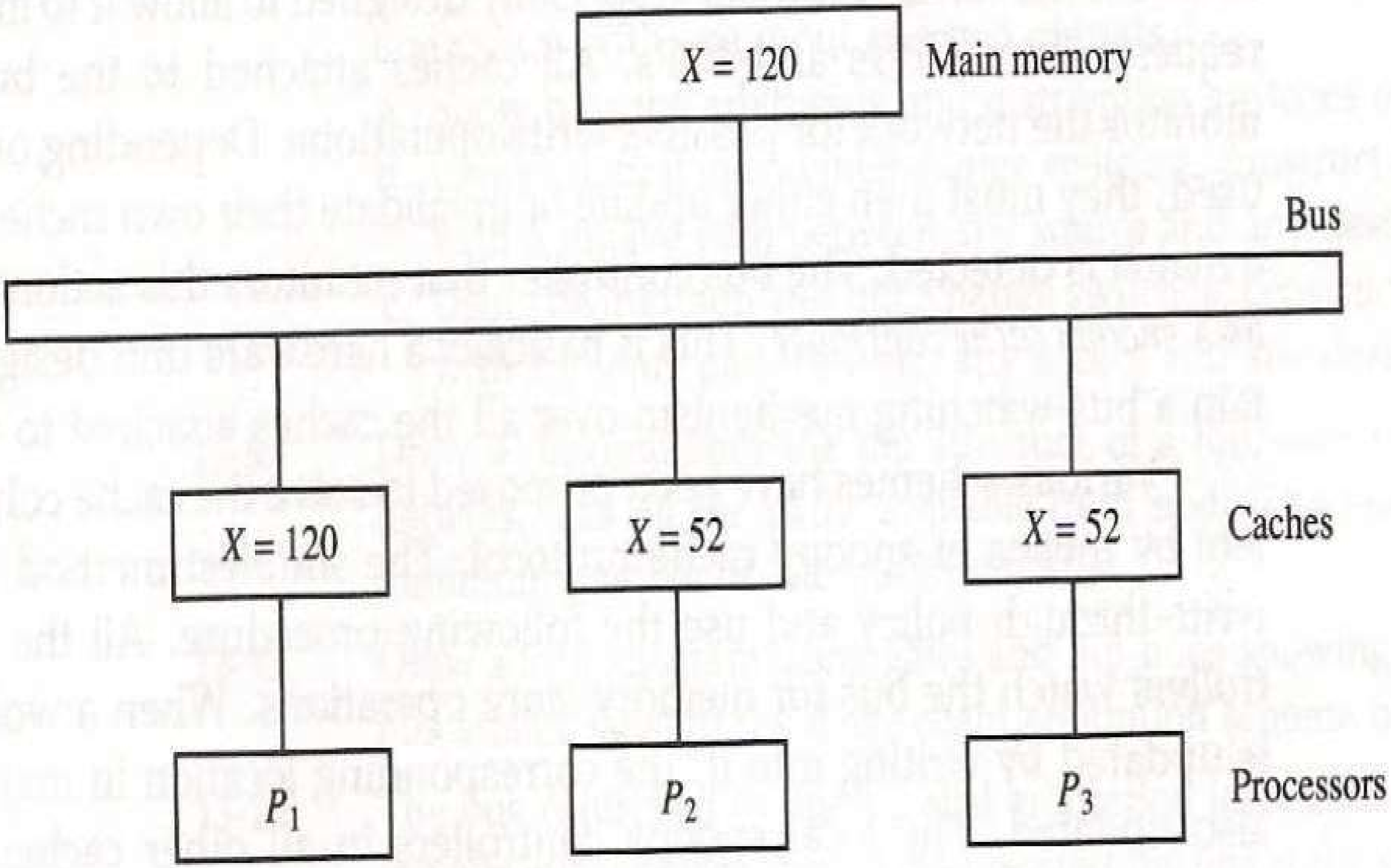
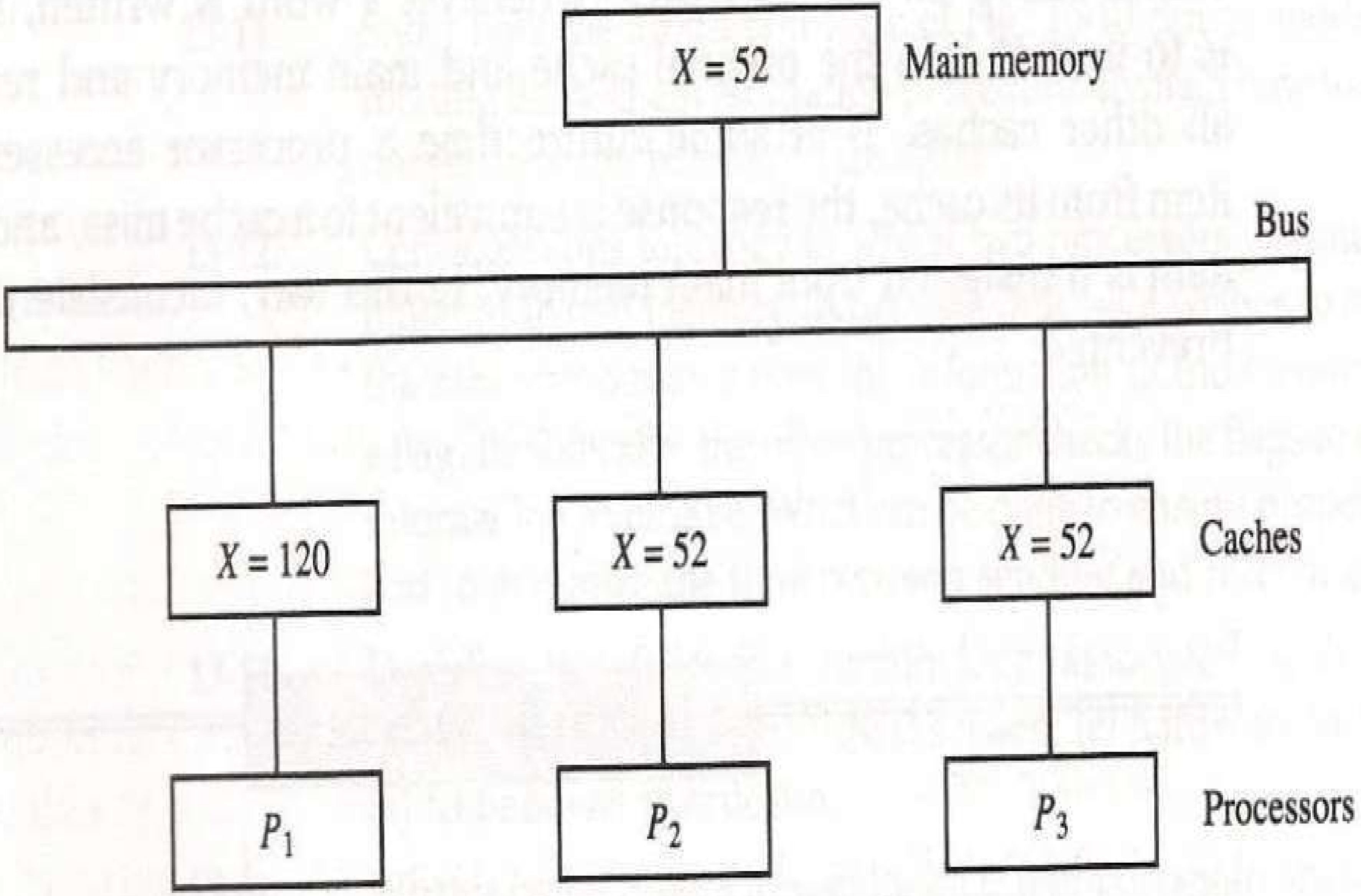


Figure Cache configuration after a load on X.

Figure Cache configuration after a store to X by processor P_1.



With write-through cache policy



With write-back cache policy

Solution using Software

- A simple way is to disallow to have private caches and use the shared cache memory for shared resources.

- Another way is to cache only the read only data in the local cache and use the common memory for writable data.

- A scheme that allows writable data to reside in at least in one cache is the use of centralized global table.
- In this table status of each memory block as read only(RO) or read write(RW) is stored.

- All local cache can store RO memory blocks.
- RW memory blocks are stored only in one cache that is shared by all.

Hardware Solution

- Snoopy cache controller is a hardware specially designed to watch the system bus for write operation.
- When a word is updated in cache, main memory is also updated.
- Local snoopy controller watch the cache if they have the address updated.
- If the copy exist, it is marked invalid.
- If CPU request the same address, it is fetched from the main memory and updated in cache.