MEMORY AND I/O DEVICES

COMPUTER ORGANIZATION AND ARCHITECTURE

4TH SEMESTER

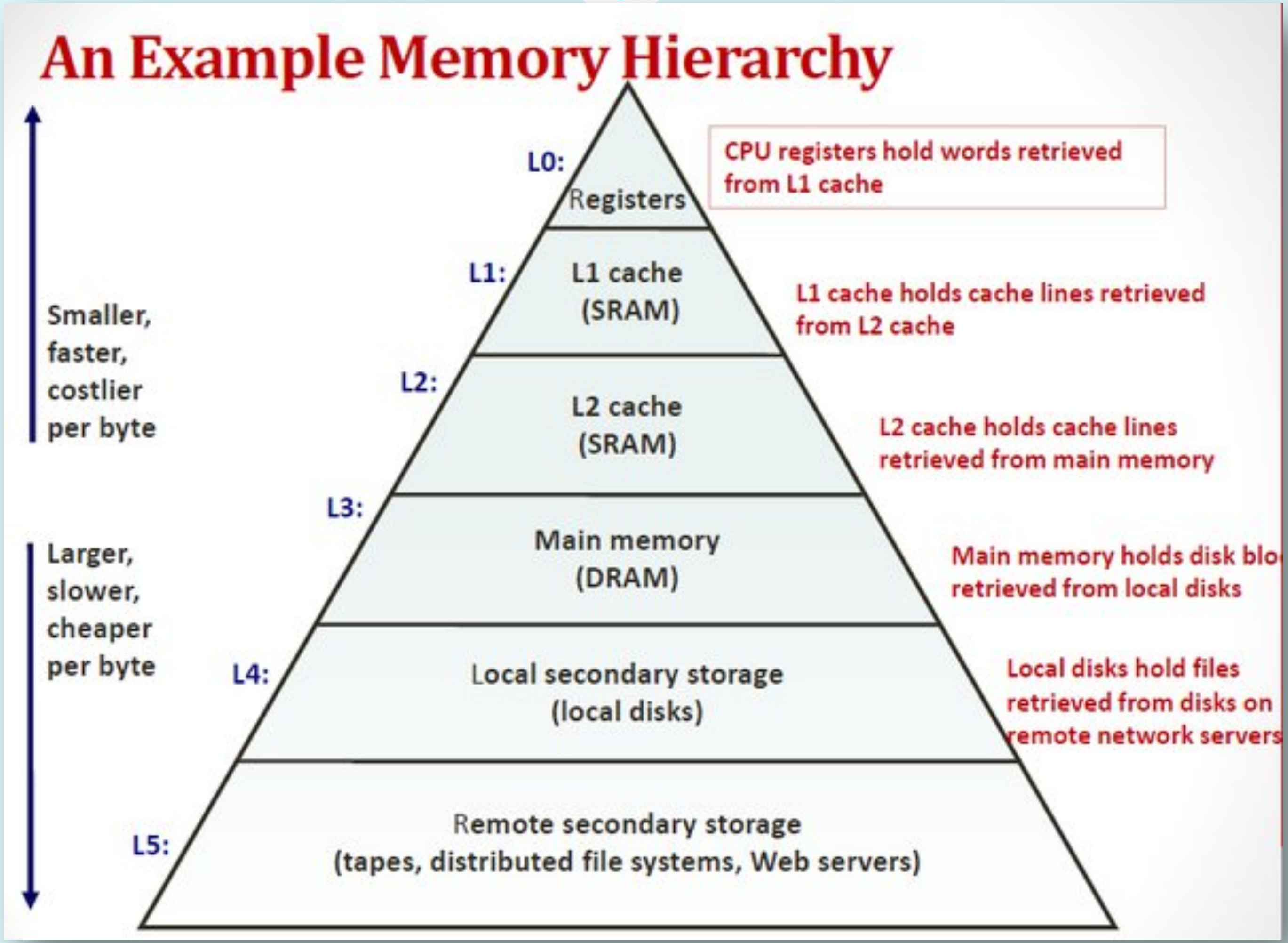
BRANCH: CSE & CST

PRESENTED BY : TAPAS CH. SINGH

Topics to be covered

MEMORY AND I/O : Need for a hierarchical memory system, Types and characteristics of memories ,Memory location and address, Endianness of memory representation, Cache memories, Improving cache performance, Virtual memory ,Memory management techniques, cache mapping and its techniques ,Associative memories. Page Replacement Algorithms. Accessing I/O devices Programmed Input/output, Interrupts, Direct Memory Access ,Interface circuits ,Need for Standard I/O Interfaces like PCI, SCSI, USB.

Hierarchical memory system


An Example Memory Hierarchy
Smaller, faster, costlier per byte
Larger, slower, cheaper per byte
L0: Registers
L1: L1 cache (SRAM)
L2: L2 cache (SRAM)
L3: Main memory (DRAM)
L4: Local secondary storage (local disks)
L5: Remote secondary storage (tapes, distributed file systems, Web servers)
CPU registers hold words retrieved from L1 cache
L1 cache holds cache lines retrieved from L2 cache
L2 cache holds cache lines retrieved from main memory
Main memory holds disk blo
retrieved from local disks
Local disks hold files retrieved from disks on remote network servers

Need for a memory hierarchy

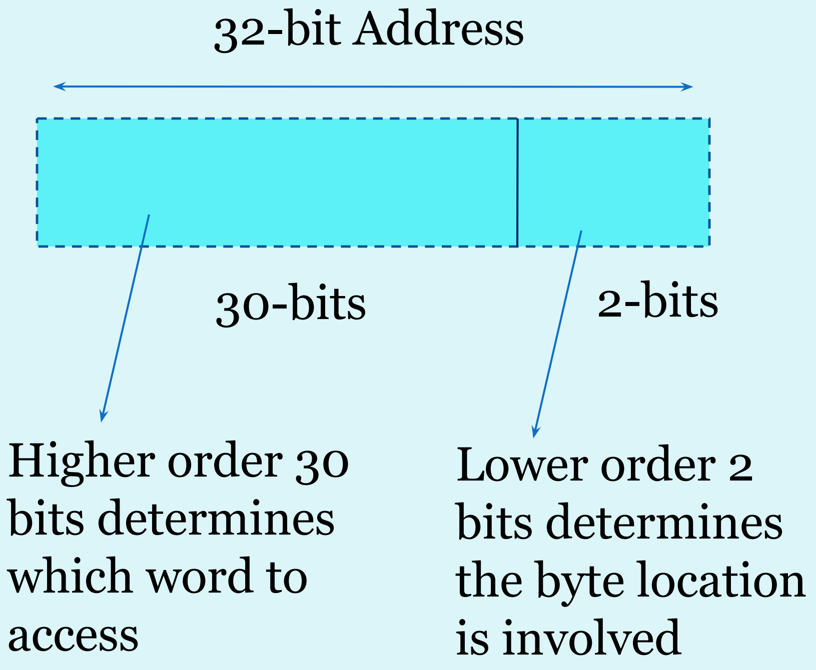
- Memory distributing is simple and economical
- Removes external destruction
- Data can be spread all over
- Permits demand paging & pre-paging
- Swapping will be more proficient

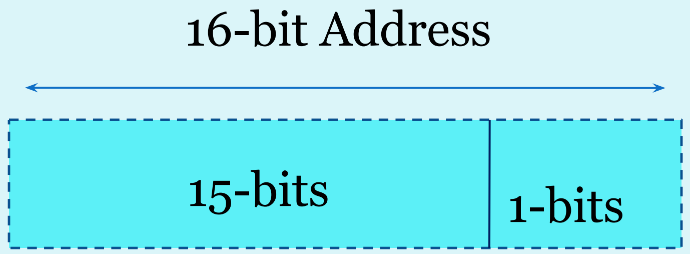
Basic Concepts

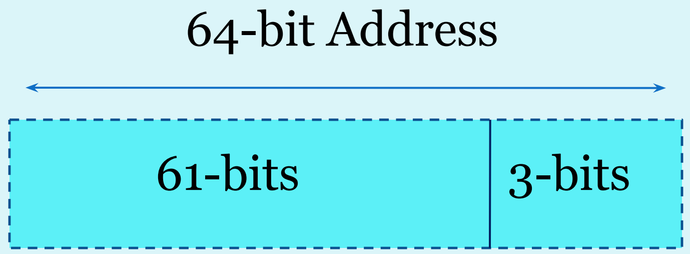
- The maximum size of the memory that can be used in any computer is determined by the addressing scheme

- 16-bit addresses is capable of addressing up to 2^16 = 64K (kilo) memory locations.
- 32-bit addresses can utilize a memory that contains up to 232 = 4G locations
- 64-bit addresses can access up to 264 = 16E (exa) ≈ 16 × 10^18 locations

- 32-bit Address

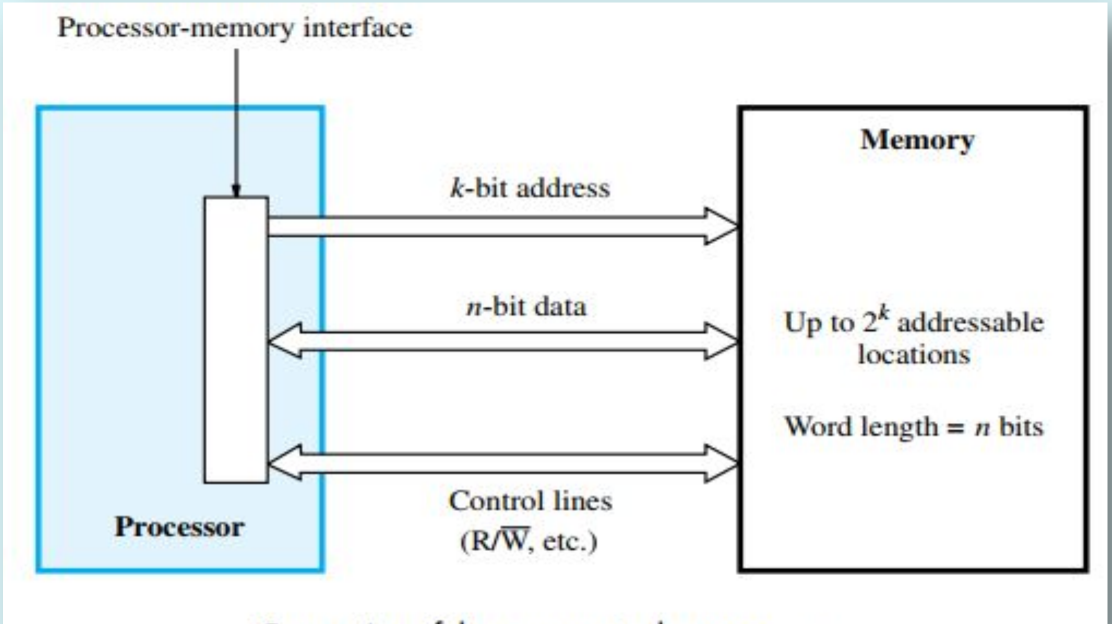

32-bit Address
30-bits
2-bits
Higher order 30 bits determines which word to access
Lower order 2 bits determines the byte location is involved





Connection of memory to the processor

- **Address lines**: to specify the memory location involved in a data transfer operation
- **Control lines :** carry the command indicating a Read or a Write operation and whether a byte or a word is to be transferred.
- Provide the necessary timing information
- It asserts the MFC signal , when memory operation has been completed.



Connection of the memory to the processor.

- ***Memory access time* : T**ime that elapses between the initiation of an operation to transfer a word of data and the completion of that operation.
- ***Memory cycle time* : M**inimum time delay required between the initiation of two successive memory operations,

- The cycle time is usually slightly longer than the access time, depending on the implementation details of the memory unit.

MEMORY-LOCATIONS & ADDRESSES

- Memory consists of many millions of storage cells (flip-flops).
- Each cell can store a bit of information i.e. 0 or 1 (Figure).
- Each group of n bits is referred to as a word of information, and n is called the word length.
- The word length can vary from 8 to 64 bits.
- A unit of 8 bits is called a byte.
- Accessing the memory to store or retrieve a single item of information (word/byte) requires distinct addresses for each item location. (It is customary to use numbers from 0 through 2^k-1 as the addresses of successive-locations in the memory).
- If 2^k = no. of addressable locations; then 2^k addresses constitute the address-space of the computer. For example, a 24-bit address generates an address-space of 2^24 locations (16 MB).

- The memory is organized so that a group of *n* bits can be stored or retrieved in a single, basic operation

- Each group of *n* bits is referred to as a *word* of information, and *n* is called the *word length*.

- Modern computers have word lengths that typically range from 16 to 64 bits.

-

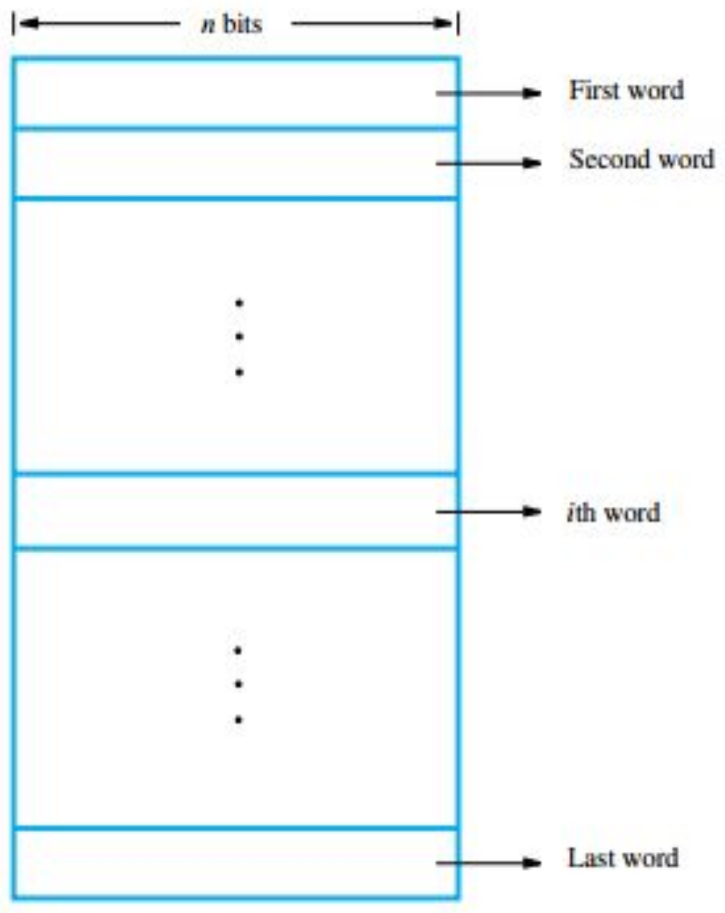
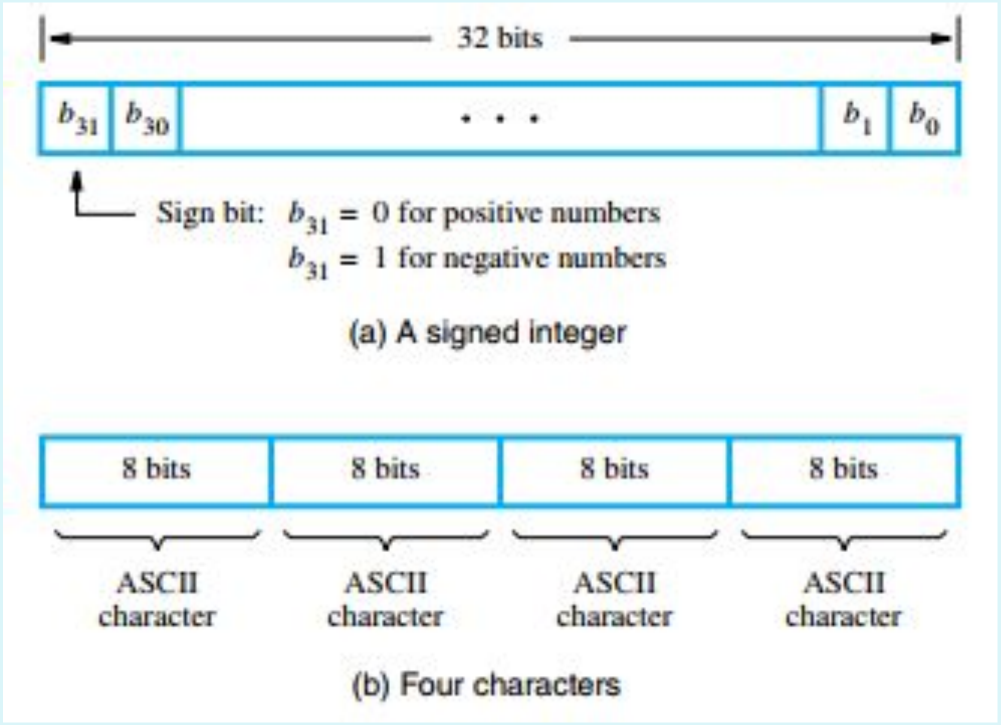


Figure 2.1 Memory words.

- If the word length of a computer is 32 bits, a single word can store a 32-bit signed number or four ASCII-encoded characters, each occupying 8 bits



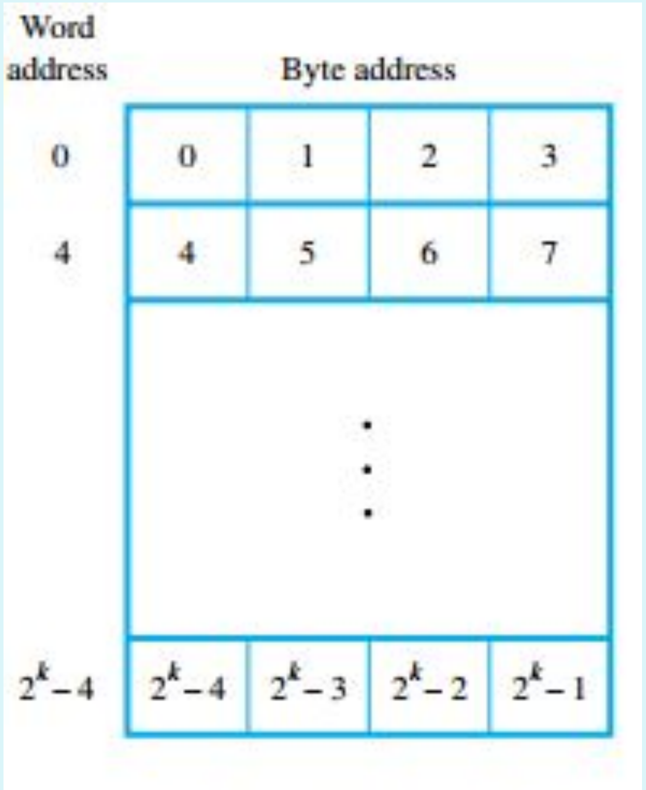
BYTE-ADDRESSABILITY

- A byte is always 8 bits, but the word length typically ranges from 16 to 64 bits.
- In byte-addressable memory, successive addresses refer to successive byte locations in the memory.
- Byte locations have addresses 0, 1, 2. . . . .
- If the word-length is 32 bits, successive words are located at addresses 0, 4, 8. . with each word having 4 bytes.

- There are two ways that byte addresses can be assigned across words,
  - ***big-endian assignment***
  - ***little-endian assignment***

- The name *big-endian* is used when lower byte addresses are used for the more significant bytes (the leftmost bytes) of the word.

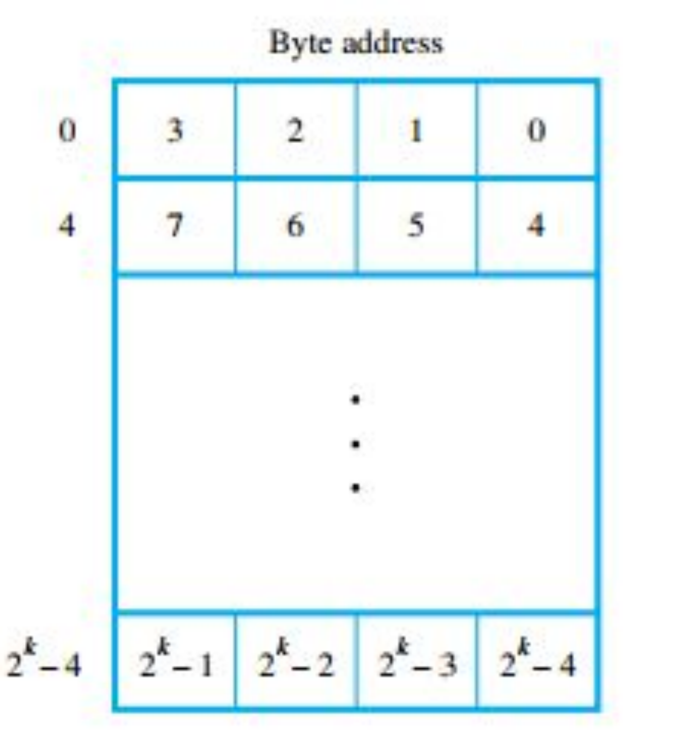
-



(a) Big-endian assignment

- The name *little-endian* is used for the opposite ordering, where the lower byte addresses are used for the less significant bytes (the rightmost bytes) of the word.

- In both cases, byte addresses 0, 4, 8, . . . , are taken as the addresses of successive words in the memory of a computer with a 32-bit word length.



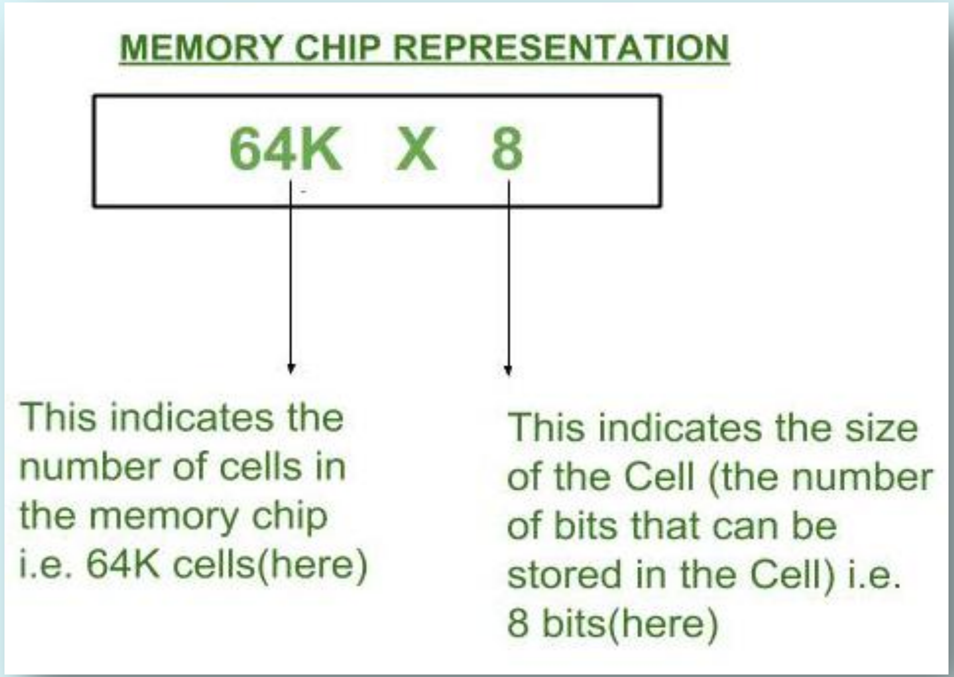
(b) Little-endian assignment

Difference

Byte Addressable Memory	Word Addressable Memory
When the data space in the cell = 8 bits then the corresponding address space is called as <u>Byte Address</u>.	When the data space in the cell = word length of CPU then the corresponding address space is called as <u>Word Address</u>.
Based on this data storage i.e. Bytewise storage, the memory chip configuration is named as <u>Byte Addressable Memory</u>.	Based on this data storage i.e. Wordwise storage, the memory chip configuration is named as <u>Word Addressable Memory</u>.
For eg. : 64K X 8 chip has 16 bit Address and cell size = 8 bits (1 Byte) which means that in this chip, data is stored <u>byte by byte</u>.	For eg. : For a 16-bit CPU, 64K X 16 chip has 16 bit Address & cell size = 16 bits (Word Length of CPU) which means that in this chip, data is stored <u>word by word</u>.

Memory Chip Representation

- **1. Data Space in the Chip =** 64K X 8
- **2. Data Space in the Cell =** 8 bits
- **3. Address Space in the Chip =**
  64*1024= 16 bits


MEMORY CHIP REPRESENTATION
64K X 8
This indicates the number of cells in the memory chip i.e. 64K cells(here)
This indicates the size of the Cell (the number of bits that can be stored in the Cell) i.e. 8 bits(here)

Questions

Q1. Assume a computer can use at most 18-bits to store a main memory address. Determine the maximum size of the main memory if it is byte-addressable.

Q2. Assume a 16K-byte memory. What are the lowest and highest address if the memory is word-addressable, assuming a 32-bit word?

Q3. How many bits would you need to address a memory with 2M X 4 – byte words if

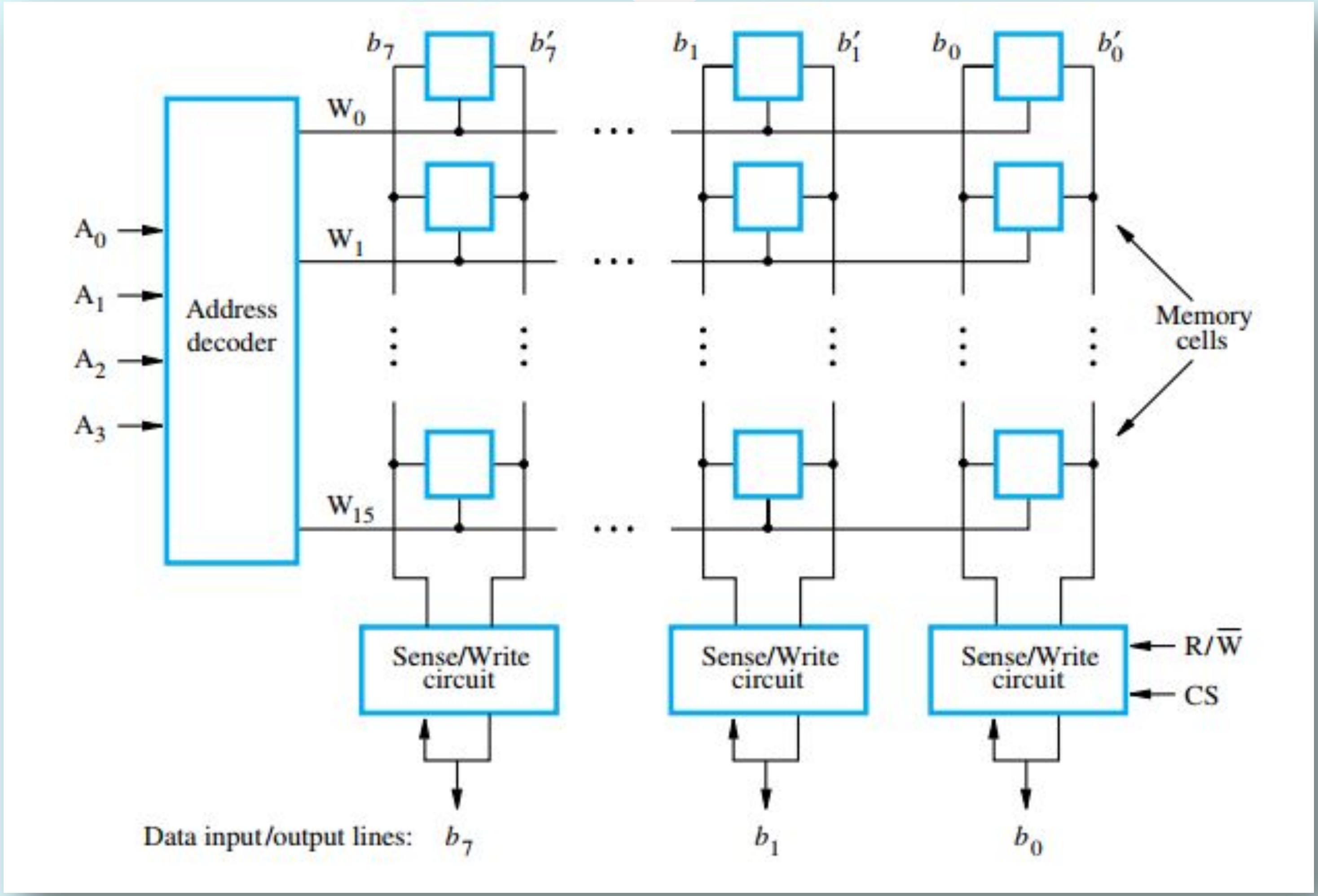
a) The memory is byte-addressable?

b) The memory is word-addressable?

c) the memory is word-addressable with a word size of 16 bits?

d) the memory is word-addressable with a word size of 32 bits?

Organization of bit cells in a memory chip



Cont..

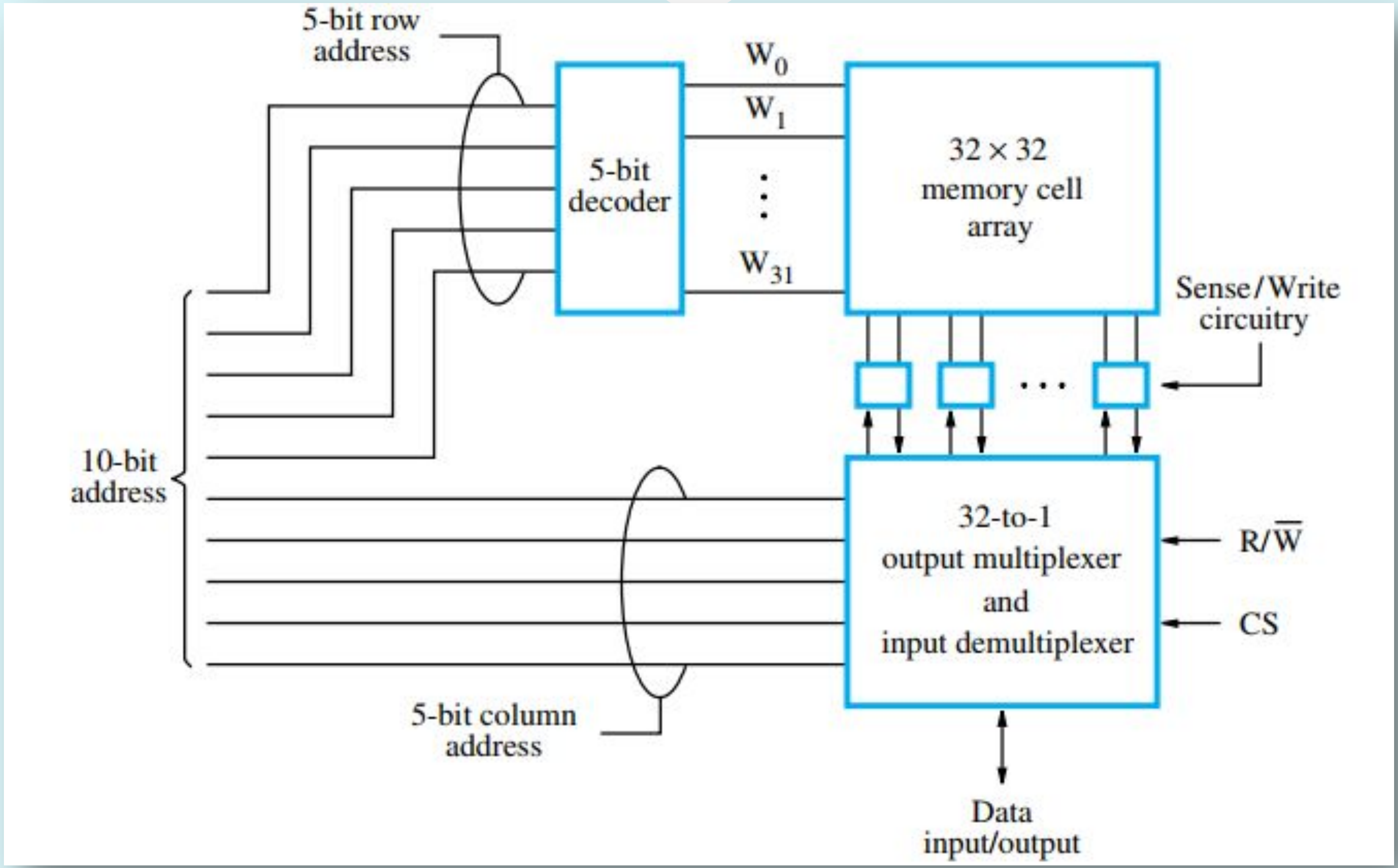
- **Memory cells are usually organized in the form of an array, in which each cell is capable of storing one bit of information.**
- **Each row of cells constitutes a memory word, and all cells of a row are connected to a common line referred to as the *word line,* , which is driven by the address decoder on the chip.**
- **The cells in each column are connected to a Sense/Write circuit by two *bit lines,* and the Sense/Write circuits are connected to the data input/output lines of the chip.**
- **Two control lines, $R/\overline{W}$ and CS, are provided. The $R/\overline{W}$ (Read/$\overline{\text{Write}}$) input specifies the required operation, and the CS (Chip Select) input selects a given chip in a multichip memory system.**
- **During a Read operation, these circuits sense, or read, the information stored in the cells selected by a word line and place this information on the output data lines.**
- **During a Write operation, the Sense/Write circuits receive input data and store them in the cells of the selected word.**
- **The memory circuit in Figure can stores 128 bits and requires 14 external connections for address, data, and control lines. It also needs two lines for power supply and ground connections.**
-

Organization of a 1K × 1 memory chip

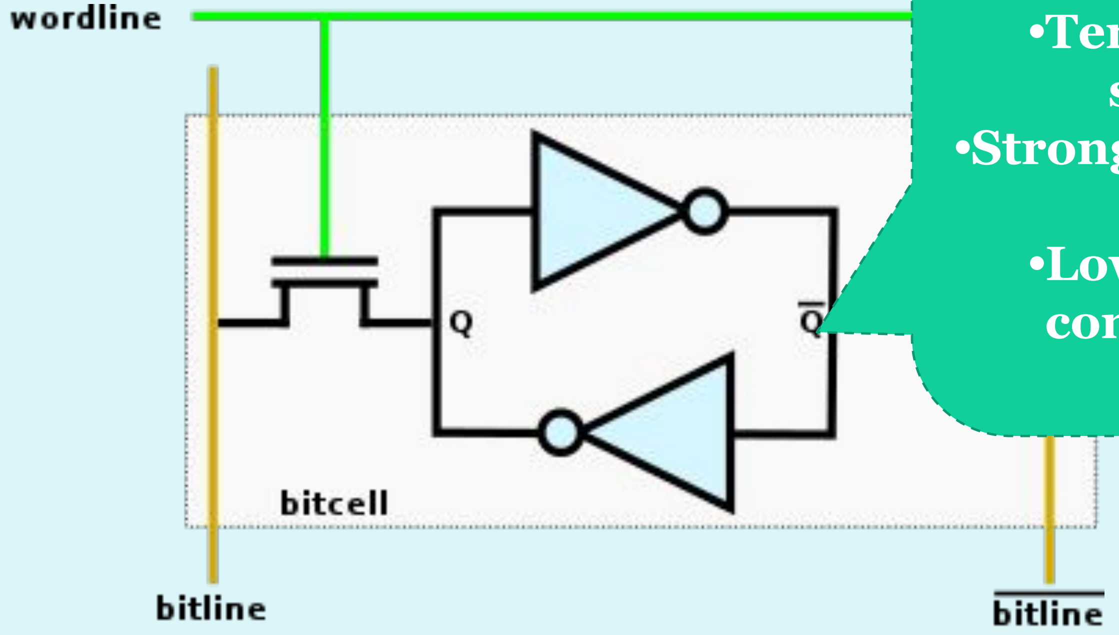
- Consider now a slightly larger memory circuit, one that has 1K (1024) memory cells. This circuit can be organized as a 128 × 8 memory, requiring a total of 19 external connections.

- Alternatively, the same number of cells can be organized into a 1K × 1 format. In this case, a 10-bit address is needed, but there is only one data line, resulting in 15 external connections.

Organization of a 1K × 1 memory chip



Semiconductor RAM Memories



CMOS ADVANTAGES:

•Temperature stability
•Stronger anti noise ability
•Lower power consumption

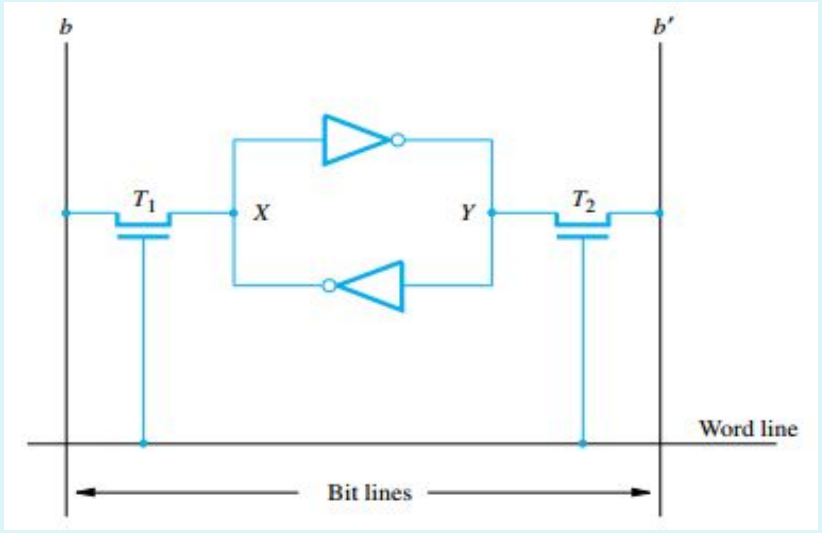
Static Memories

- Memories that consist of circuits capable of retaining their state as long as power is applied are known as *static memories*.
-

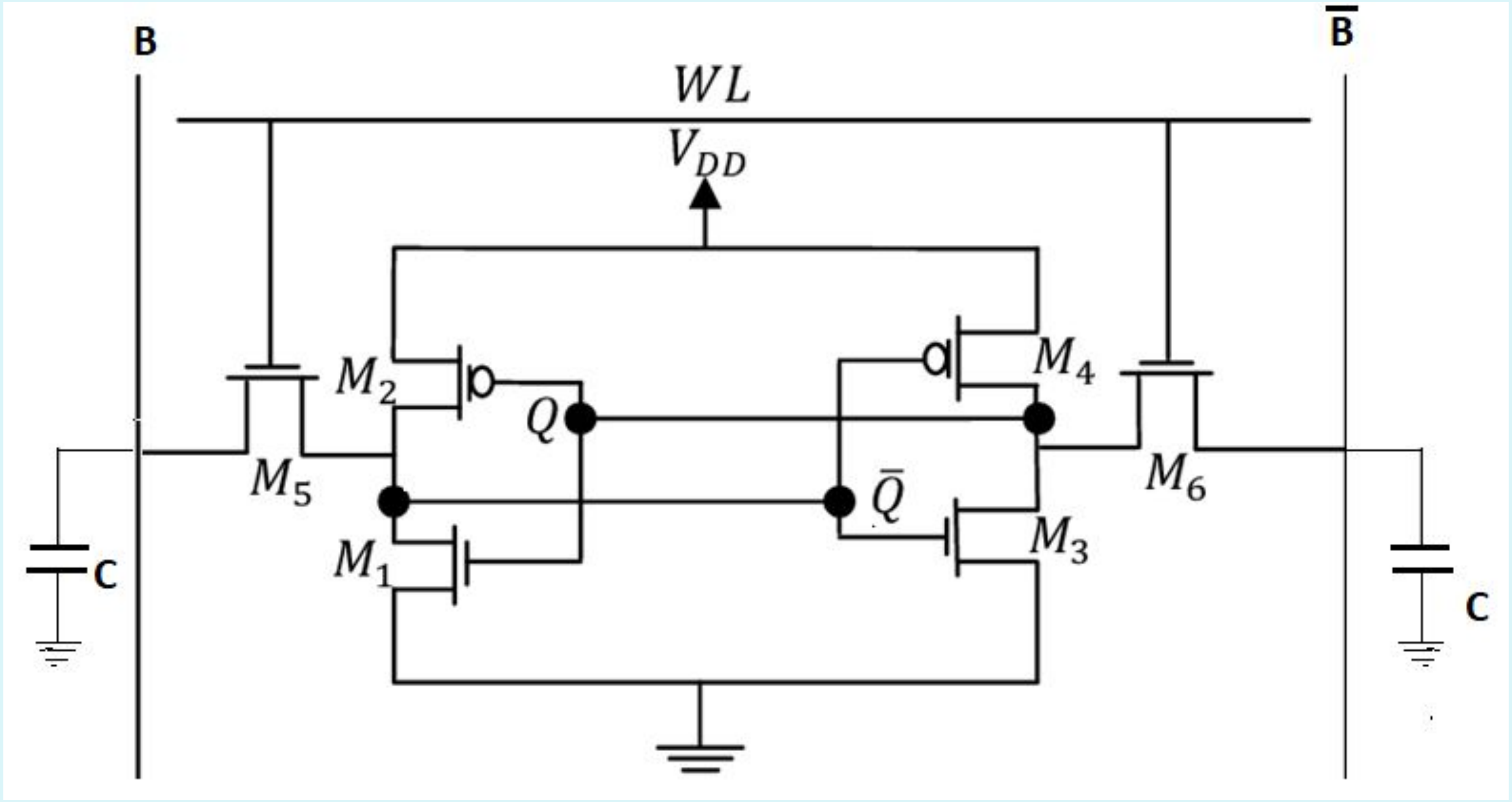
•Two inverters are cross-connected to form a latch. The latch is connected to two bit lines by transistors *T*1 and *T*2. These transistors act as switches that can be opened or closed under control of the word line.

•When the word line is at ground level, the transistors are turned off and the latch retains its state.

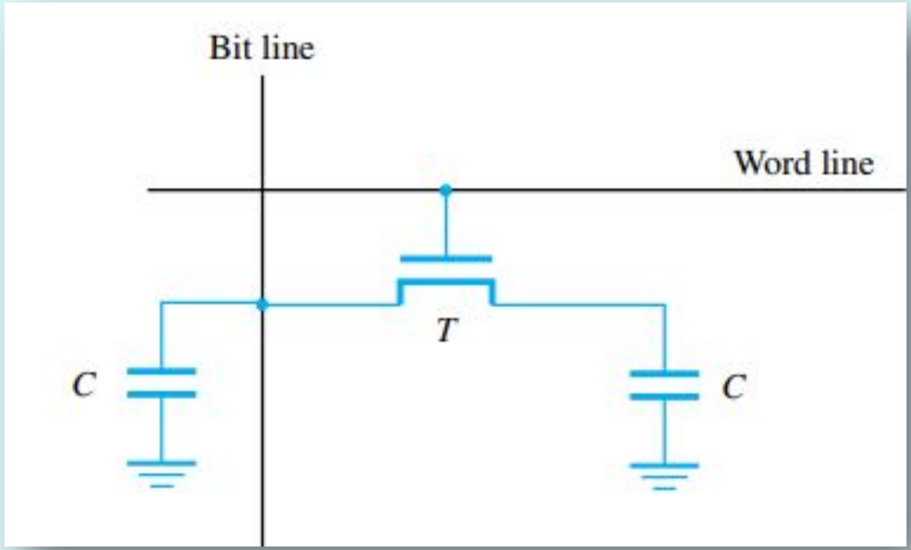
•

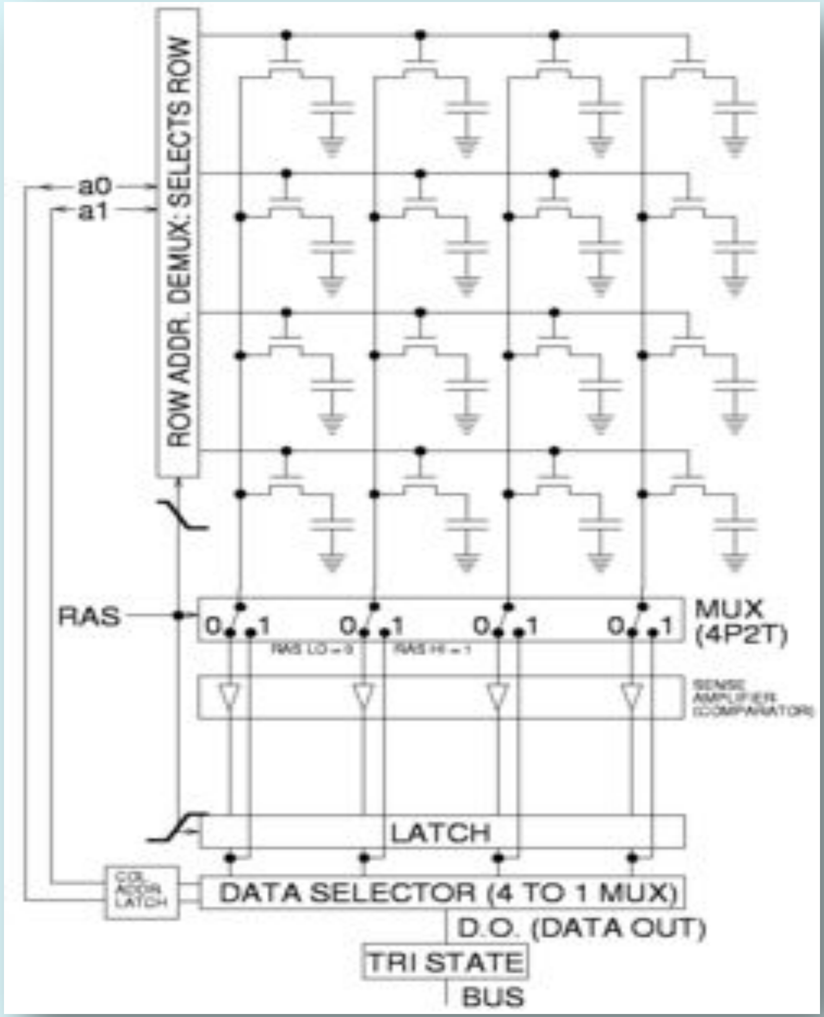


SRAM



DRAM


Bit line
Word line
T
C
C


ROW ADDR. DEMUX: SELECTS ROW
a0
a1
RAS
0 1
MUX (4P2T)
RAS LO = 0
RAS HI = 1
SENSE AMPLIFIER (COMPARATOR)
LATCH
COL ADDR. LATCH
DATA SELECTOR (4 TO 1 MUX)
D.O. (DATA OUT)
TRI STATE
BUS

- Information is stored in a dynamic memory cell in the form of a charge on a capacitor.
- Dynamic: DRAM needs periodic refreshment.
- Volatile: losses data when powered off.
- Single transistor, therefore smaller and cheaper.
- Charge stored $Q= C\ V_{dd}$

- A dynamic memory cell that consists of a capacitor, *C*, and a transistor, *T*, is shown in Figure

During a Read operation:

- The transistor in a selected cell is turned on . A sense amplifier connected to the bit line detects whether the charge stored in the capacitor is above or below the threshold value.

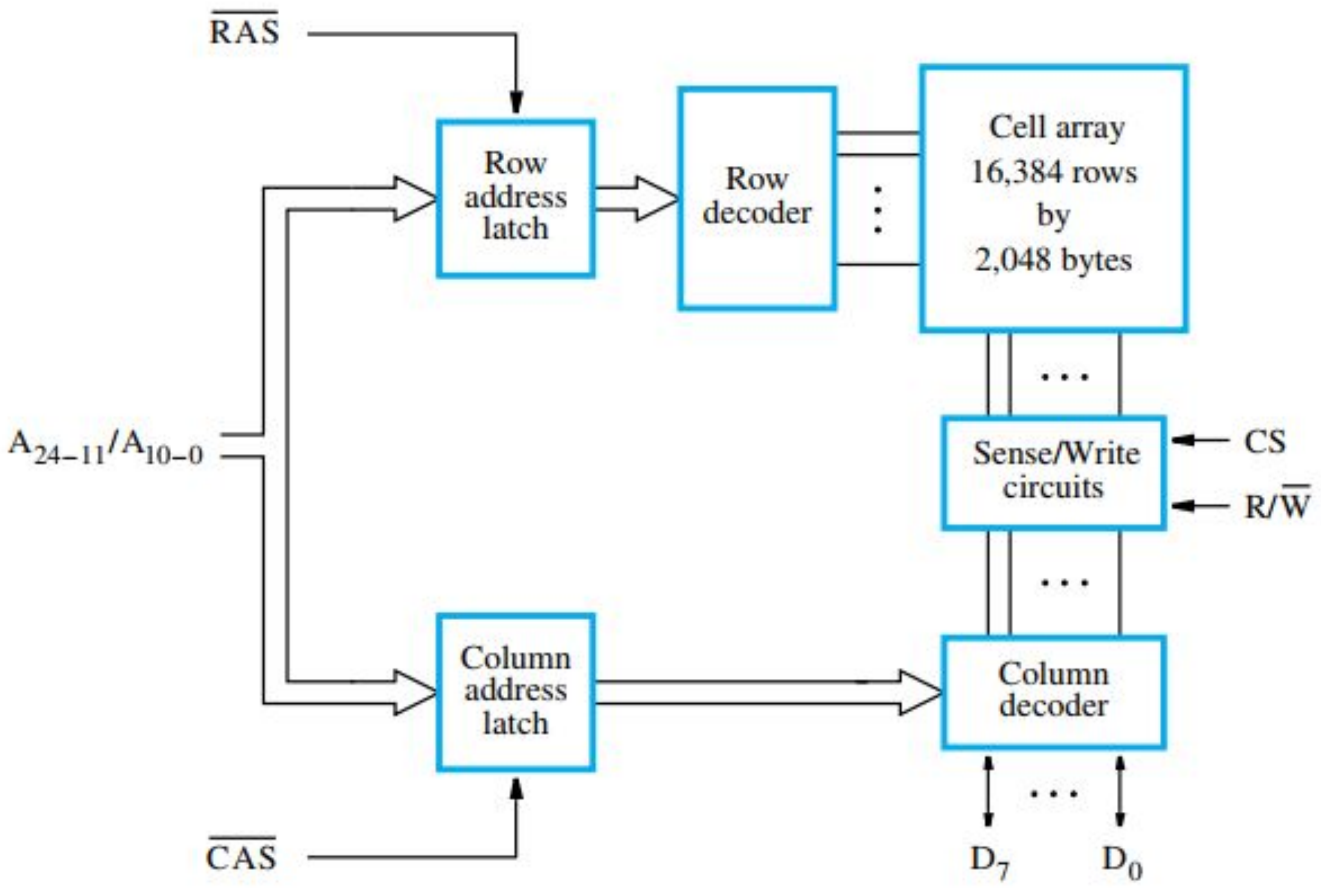
- If the charge is above the threshold, the sense amplifier drives the bit line to the full voltage representing the logic value 1. As a result, the capacitor is recharged to the full charge corresponding to the logic value 1.

- If the sense amplifier detects that the charge in the capacitor is below the threshold value, it pulls the bit line to ground level to discharge the capacitor fully.

- Thus, reading the contents of a cell automatically refreshes its contents. Since the word line is common to all cells in a row, all cells in a selected row are read and refreshed at the same time.

- **During a Write operation:**
- To store information in this cell, transistor *T* is turned on and an appropriate voltage is applied to the bit line. This causes a known amount of charge to be stored in the capacitor .
- After the transistor is turned off, the charge remains stored in the capacitor, but not for long. The capacitor begins to discharge. This is because the transistor continues to conduct a tiny amount of current, measured in picoamperes , after it is turned off. Hence, the information stored in the cell can be retrieved correctly only if it is read before the charge in the capacitor drops below some threshold value. to store information for a much longer time, its contents must be periodically *refreshed* by restoring the capacitor charge to its full value.
-

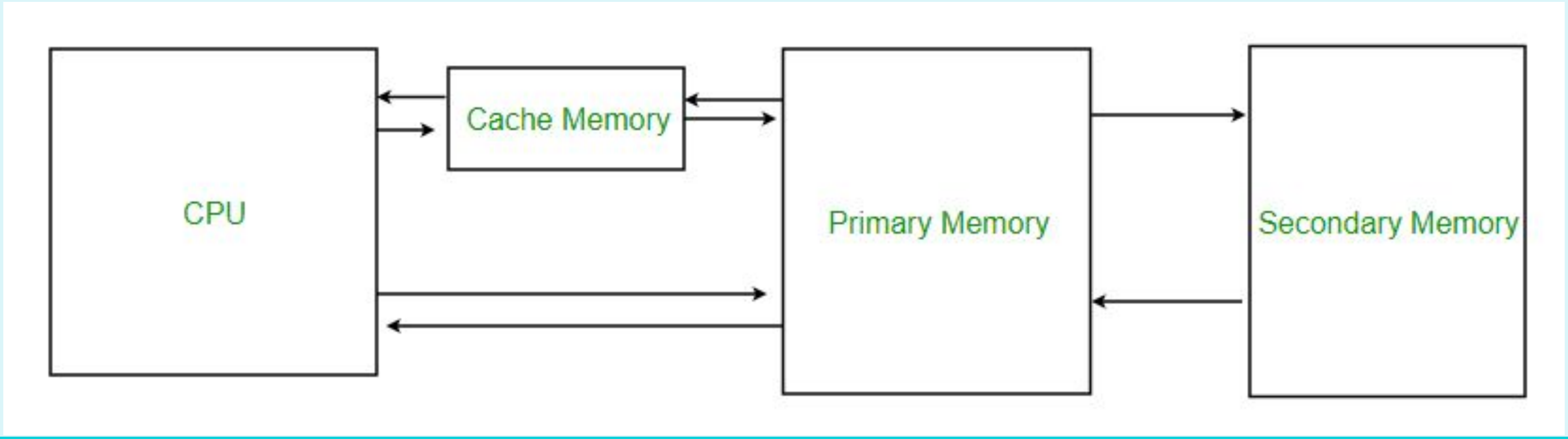
Internal organization of dynamic memory chip



Internal organization of a 32M × 8 dynamic memory chip.

Cache Memories

- Cache memory is used to reduce the average time to access data from the Main memory.
- The cache is a smaller and faster memory which stores copies of the data from frequently used main memory locations.
- Cache Memory is a special very high-speed memory. It is used to speed up and synchronizing with high-speed CPU



Cache Operation

It is based on the principle of locality of reference. There are two ways with which data or instruction is fetched from main memory and get stored in cache memory. These two ways are the following:

- **Temporal Locality**
  Temporal locality means current data or instruction that is being fetched may be needed soon. So we should store that data or instruction in the cache memory so that we can avoid again searching in main memory for the same data.

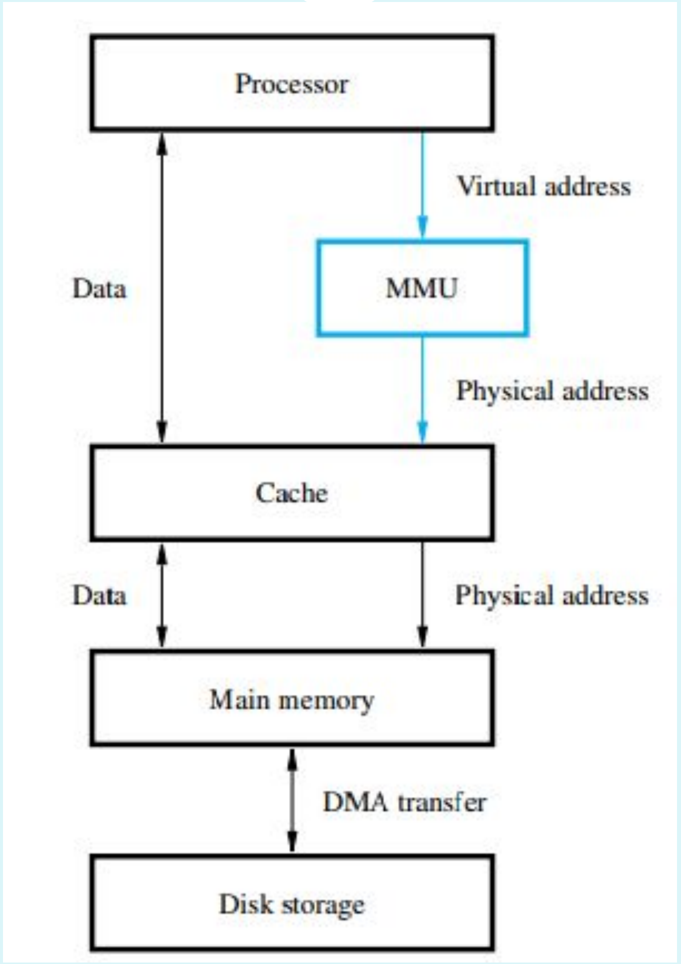
- **Spatial Locality**
  Spatial locality means instruction or data near to the current memory location that is being fetched, may be needed soon in the near future. This is slightly different from the temporal locality. Here we are talking about nearly located memory locations while in temporal locality we were talking about the actual memory location that was being fetched.

Difference between Spatial Locality and Temporal Locality

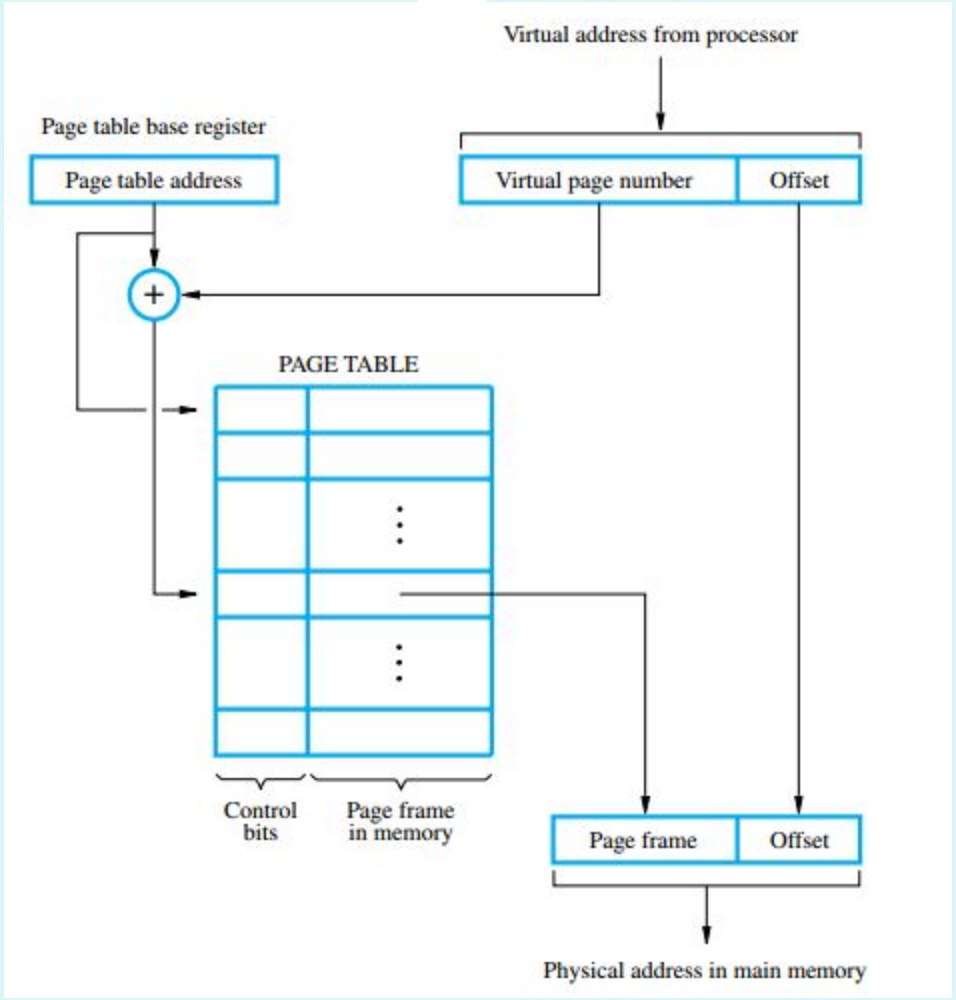
S.N	Spatial Locality	Temporal Locality
1.	In Spatial Locality, nearby instructions to recently executed instruction are likely to be executed soon.	In Temporal Locality, a recently executed instruction is likely to be executed again very soon.
2.	It refers to the tendency of execution which involve a number of memory locations .	It refers to the tendency of execution where memory location that have been used recently have a access.
3.	It is also known as locality in space.	It is also known as locality in time.
4.	It only refers to data item which are closed together in memory.	It repeatedly refers to same data in short time span.
5.	Each time new data comes into execution.	Each time same useful data comes into execution.
6.	Example : Data elements accessed in array (where each time different (or just next) element is being accessing).	Example : Data elements accessed in loops (where same data elements are accessed multiple times).

Cache Performance

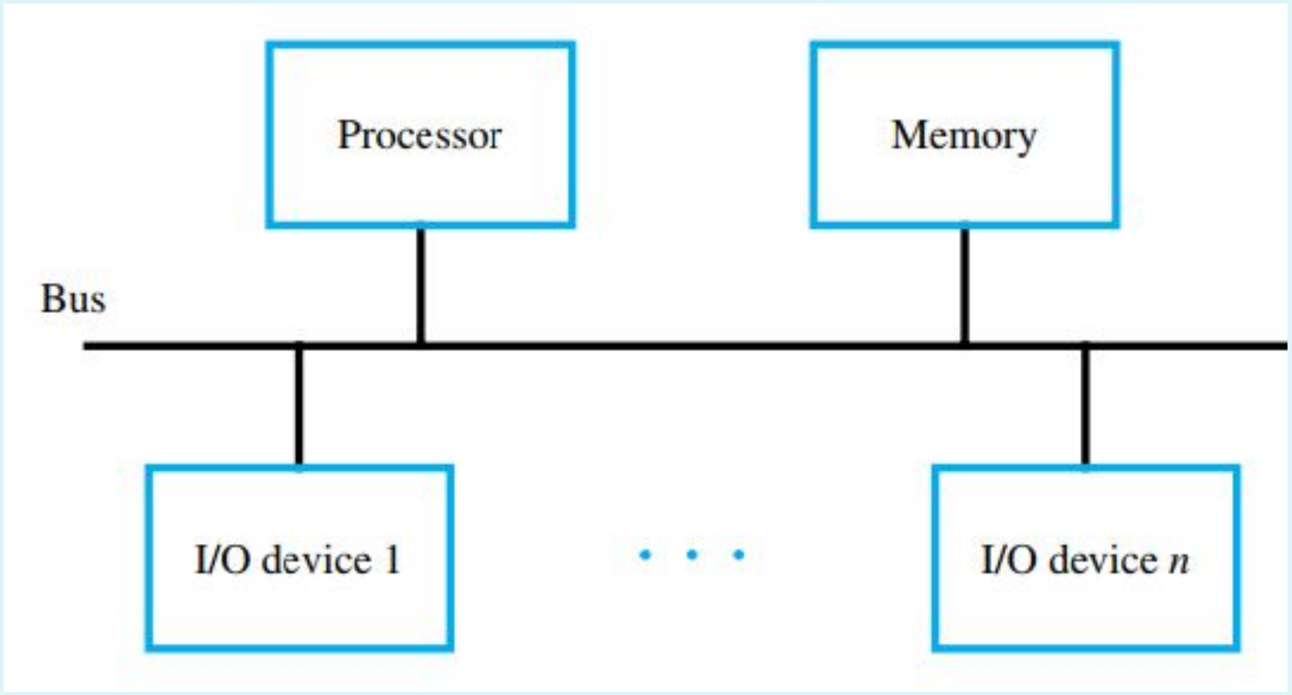
Virtual Memory



Virtual-memory address translation



Input-output subsystems

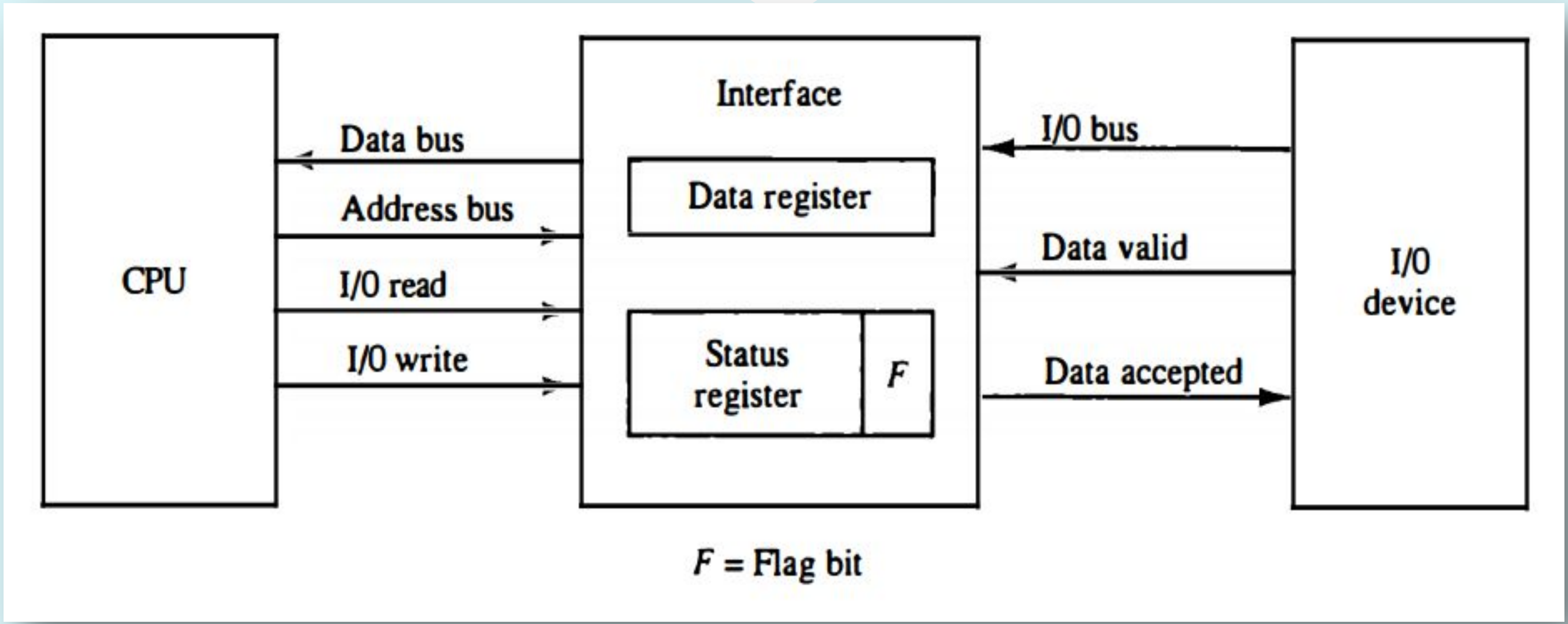


Modes of data Transfer

- Data transfer to and from peripherals may be handled in one of three possible modes:

  1. Programmed I/O
  2. Interrupt-initiated I/O
  3. Direct memory access (DMA)

Data transfer from I/O device to CPU.

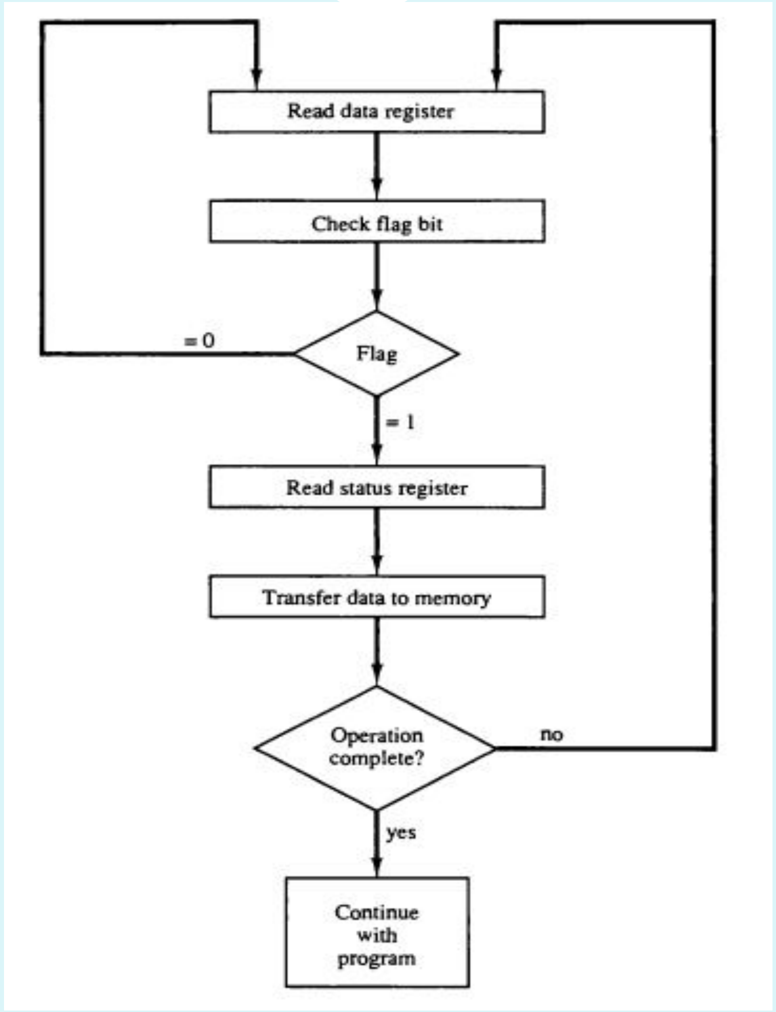


- The transfer of each byte requires three instructions:
- **1 Read the status register.**
  2. Check the status of the flag bit and branch to step 1 if not set or to step if set.
  3. Read the data register.

Data transfer from I/O device to CPU

- The device transfers bytes of data one at a time as they are available. When a byte of data is available, the device places it in the I/O bus and enables its data valid line.
- The interface accepts the byte into its data register and enables the data accepted line. The interface sets a bit in the status register that we will refer to as an F or "flag" bit.
- The device can now disable the data valid line, but it will not transfer another byte until the data accepted line is disabled by the interface.
- A program is written for the computer to check the flag in the status register to determine if a byte has been placed in the data register by the I/O device.
- This is done by reading the status register into a CPU register and checking the value of the flag bit. If the flag is equal to 1, the CPU reads the data from the data register.
- The flag bit is then cleared to 0 by either the CPU or the interface, depending on how the interface circuits are designed. Once the flag is cleared, the interface disables the data accepted line and the device can then transfer the next data byte.
-

Flowchart for CPU program to input data



Interrupt-Initiated I/O

- An alternative to the CPU constantly monitoring the flag is to let the interface inform the computer when it is ready to transfer data.
- This mode of transfer uses the interrupt facility. While the CPU is running a program, it does not check the flag.
- However, when the flag is set, the computer is momentarily interrupted from proceeding with the current program and is informed of the fact that the flag has been set.
- The CPU deviates from what it is doing to take care of the input or output transfer.
- After the transfer is completed, the computer returns to the previous program to continue what it was doing before the interrupt.
- The CPU responds to the interrupt signal by storing the return address from the program counter into a memory stack and then control branches to a service routine that processes the required I/O transfer. The way that the processor chooses the branch address of the service routine varies from one unit to another.

Cont..

- There are two methods for accomplishing this. One is called vectored interrupt and the other, non-vectored interrupt.
- In a non-vectored interrupt, the branch address is assigned to a fixed location in memory.
- In a vectored interrupt, the source that interrupts supplies the branch information to the computer. This information is called the interrupt vector.
- In some computers the interrupt vector is the first address of the I/O service routine. In other computers the interrupt vector is an address that points to a location in memory where the beginning address of the I/O service routine is stored.

Programmed I/O	Interrupt Initiated I/O
Data transfer is initiated by the means of instructions stored in the computer program. Whenever there is a request for I/O transfer the instructions are executed from the program.	The I/O transfer is initiated by the interrupt command issued to the CPU.
The CPU stays in the loop to know if the device is ready for transfer and has to continuously monitor the peripheral device.	There is no need for the CPU to stay in the loop as the interrupt command interrupts the CPU when the device is ready for data transfer.
This leads to the wastage of CPU cycles as CPU remains busy needlessly and thus the efficiency of system gets reduced.	The CPU cycles are not wasted as CPU continues with other work during this time and hence this method is more efficient.
CPU cannot do any work until the transfer is complete as it has to stay in the loop to continuously monitor the peripheral device.	CPU can do any other work until it is interrupted by the command indicating the readiness of device for data transfer
Its module is treated as a slow module.	Its module is faster than programmed I/O module.
It is quite easy to program and understand.	It can be tricky and complicated to understand if one uses low level language.
The performance of the system is severely degraded.	The performance of the system is enhanced to some extent.

Direct Memory Access (DMA)

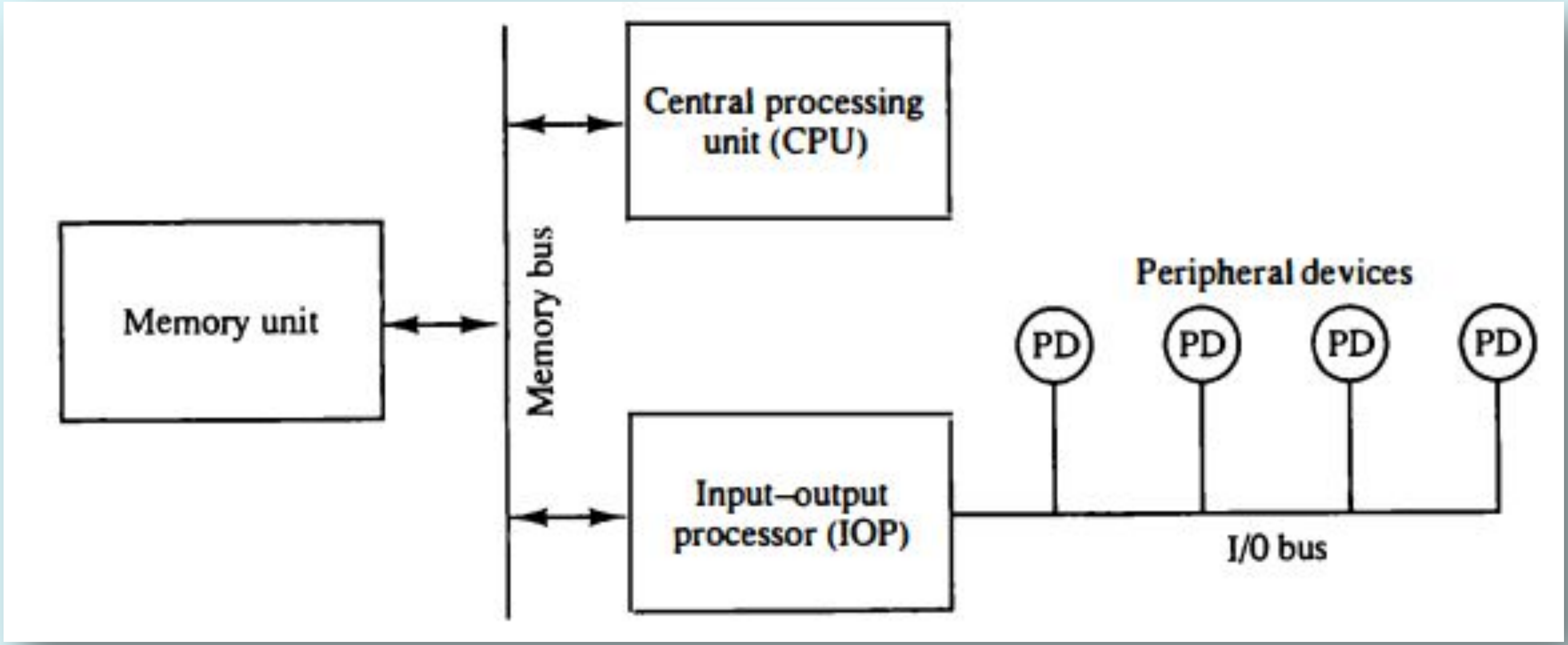
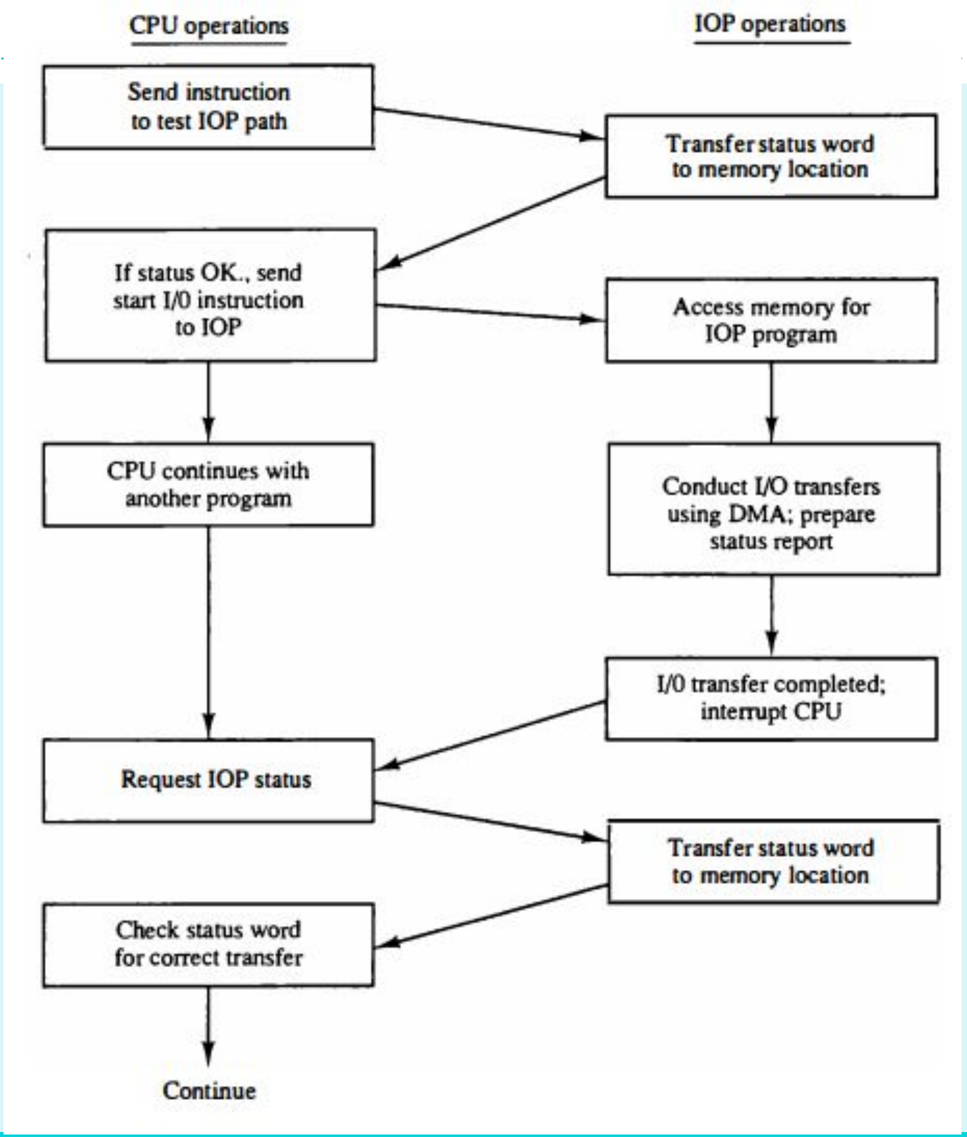


Fig: Block diagram of a computer with I/O processor.

CPU-lOP communication



CPU operations
IOP operations
Send instruction to test IOP path
Transfer status word to memory location
If status OK., send start I/0 instruction to IOP
Access memory for IOP program
CPU continues with another program
Conduct I/O transfers using DMA; prepare status report
I/0 transfer completed; interrupt CPU
Request IOP status
Transfer status word to memory location
Check status word for correct transfer
Continue

- The sequence of operations may be carried out as shown in the flowchart of Fig.
- The CPU sends an instruction to test the lOP path. The lOP responds by inserting a status word in memory for the CPU to check.
- The bits of the status word indicate the condition of the lOP and I/O device, such as lOP overload condition, device busy with another transfer, or device ready for I/O transfer.
- The CPU refers to the status word in memory to decide what to do next. If all is in order, the CPU sends the instruction to start I/O transfer. The memory address received with this instruction tells the lOP where to find its program.
- The CPU can now continue with another program while the lOP is busy with the I/O program.
- When the lOP terminates the execution of its program, it sends an interrupt request to the CPU. The CPU responds to the interrupt by issuing an instruction to read the status from the lOP.
-

- The lOP responds by placing the contents of its status report into a specified memory location. The status word indicates whether the transfer has been completed or if any errors occurred during the transfer.
- From inspection of the bits in the status word, the CPU determines if the I/O operation was completed satisfactorily without errors.
- The lOP takes care of all data transfers between several I/O units and the memory while the CPU is processing another program.
- The lOP and CPU are competing for the use of memory, so the number of devices that can be in operation is limited by the access time of the memory. It is not possible to saturate the memory by I/O devices in most systems, as the speed of most devices is much slower than the CPU.

Interface Circuits

- The I/O interface of a device consists of the circuitry needed to connect that device to the bus. On one side of the interface are the bus lines for address, data, and control.
- On the other side are the connections needed to transfer data between the interface and the I/O device. This side is called a port, and it can be either a parallel or a serial port.
- A parallel port transfers multiple bits of data simultaneously to or from the device. A serial port sends and receives data one bit at a time.
- Communication with the processor is the same for both formats; the conversion from a parallel to a serial format and vice versa takes place inside the interface circuit.

An I/O interface does the following:

1. Provides a register for temporary storage of data

2. Includes a status register containing status information that can be accessed by the processor

3. Includes a control register that holds the information governing the behaviour of the interface

4. Contains address-decoding circuitry to determine when it is being addressed by the processor

5. Generates the required timing signals

6. Performs any format conversion that may be necessary to transfer data between the processor and the I/O device, such as parallel-to-serial conversion in the case of a serial port

Interconnection Standards

Universal Serial Bus (USB)

- The original USB specification supports two speeds of operation, called low-speed (1.5 Megabits/s) and full-speed (12 Megabits/s).
- Later, USB 2, called High-Speed USB, was introduced. It enables data transfers at speeds up to 480Megabits/s.
- USB 3 (called Super speed) was developed. It supports data transfer rates up to 5 Gigabits/s.

The USB has been designed to meet several key objectives:

- • Provide a simple, low-cost, and easy to use interconnection system
  • Accommodate a wide range of I/O devices and bit rates, including Internet connections, and audio and video applications
  • Enhance user convenience through a "plug-and-play" mode of operation

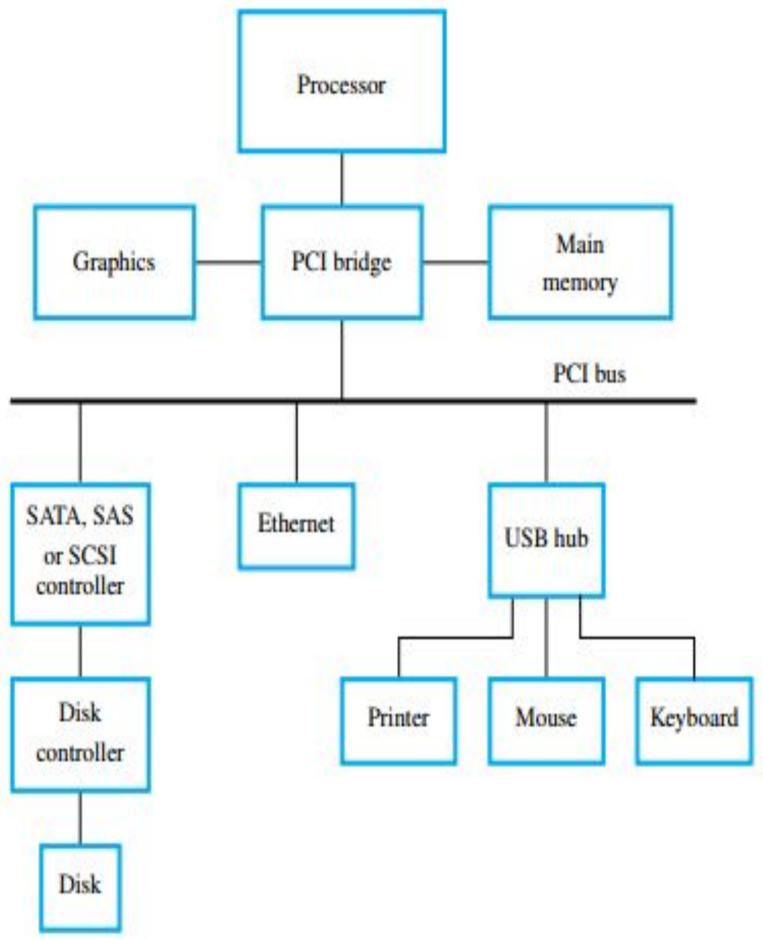
- USB connections consist of four wires, of which two carry power, +5 V and Ground, and two carry data. Thus, I/O devices that do not have large power requirements can be powered directly from the USB. This obviates the need for a separate power supply for simple devices such as a memory key or a mouse.

- Two methods are used to send data over a USB cable. When sending data at low speed, a high voltage relative to Ground is transmitted on one of the two data wires to represent a ‘0’ and on the other to represent a ‘1’. The Ground wire carries the return current in both cases. Such a scheme in which a signal is injected on a wire relative to ground is referred to as *single-ended* transmission.

PCI (Peripheral Component Interconnect)

- The PCI (Peripheral Component Interconnect) bus was developed as a low-cost, processor-independent bus.
- It is housed on the motherboard of a computer and used to connect I/O interfaces for a wide variety of devices. A device connected to the PCI bus appears to the processor as if it is connected directly to the processor bus. Its interface registers are assigned addresses in the address space of the processor.

- The PCI bus is connected to the processor bus via a controller called a bridge. The bridge has a special port for connecting the computer's main memory. It may also have another special high speed port for connecting graphics devices. The bridge translates and relays commands and responses from one bus to the other and transfers data between them.



Short answer type questions (2 Marks)

- Explain the memory hierarchy in modern computer. CO 3, PO 1
- Explain with neat diagram of static memory cell. CO 4, PO 1
- Explain the different types of endianness implemented the computer system for storing of data in memory. CO 2, PO 1
- Write about the following terms: CO 2, PO 1
  Byte-addressability
  Word alignment
- Write about the following terms: CO 2, PO 1
  Word address
  Byte location
- Define miss rate and miss penalty? CO 2, PO 1
- Define locality of reference? CO 2, PO 1
- What is the formula for calculating the average access time experienced by the processor? CO 2, PO 1
- What is the formula for calculating the average access time experienced by the processor in a system with two levels of caches? CO 2, PO 1
- What is replacement algorithm? CO 2, PO 1
- What is associative search? CO 2, PO 1

Long Type Questions (5Marks)

1. What is difference between SRAM and DRAM?
2. Draw the memory hierarchy structure of a computing device.
3. What is memory latency time?
4. Define locality of reference property of cache memory.
5. How does big endian byte addressability used in memory addressing system? Give one example.
6. How does little endian byte addressability used in memory addressing system? Give one example.
7. Write the memory alignment address of a 32- bit processor.
8. What is write through and write back protocol in cache memory?
9. Write different cache mapping functions used in memory.
10. For a 120 cache memory references if hit ratio is 0.75, cache access time is 25ns , miss penalty is 100ns then find the cache performance.
11. Why is virtual memory required in system?
12. What is DMA?
13. What is the necessity of Interrupts?

Long Type Questions

1. a) With a neat diagram, describe DMA transfer in a computer system 10M
2. b) Describe in detail about associative memory 5M
3. Discuss the different mapping techniques used in cache memories and their relative merits and demerits. 15M
4. Assume a cache miss penalty is 100 clock cycles, and all instructions take 1.0 clock cycles. Let the average miss rate is 2%, there is an average of 1.5 memory references per instructions, and the average number of cache misses per 1000 instructions is 30. What is the impact on the performance and calculate the impact using both misses per instruction and miss rate?
5. Illustrate the characteristics of some common memory technologies 15M
6. What is cache memory and explain different cache memory mapping techniques 15M
7. a) Describe virtual memory in detail. 7M
8. b) Compare and contrast between Asynchronous DRAM and Synchronous DRAM. 8M
9. a) What is an interrupt and explain different types of Interrupts.
10. What do you mean by Speed-Up of pipeline? Derive equations of Speed-Up and Efficiency for Pipeline, Super pipeline and Super scalar architecture.
11. What is virtual memory? How a logical address is mapped to physical address in virtual concept? Explain with example and diagram