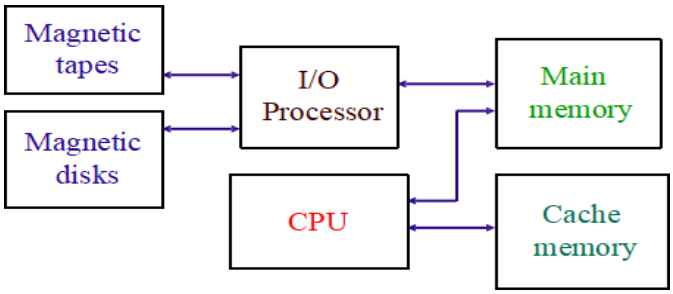
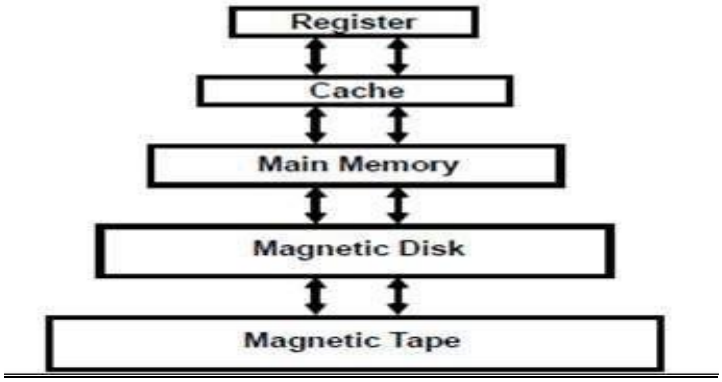
COMPUTER ORGANIZATION AND ARCHITECTURE
UNIT –IV
MEMORY ORGANIZATION
TOPIC- MEMORY HIERARCHY

Memory Hierarchy

- The memory unit is an essential component in any digital computer since used for storing programs and data.
- A very small computer with a limited application may be able to fulfill its intended task without the need of additional storage capacity.
- Most general purpose computers would run more efficiently if they were equipped with additional storage beyond the capacity of the main memory. There is just not enough space in one memory unit to accommodate all the programs used in a typical computer.
- Moreover, most computer users accumulate and continue to accumulate large amounts of data-processing software. Not all accumulated information is needed by the processor at the same time. Therefore, it is more economical to use low-cost storage devices to serve as a backup for storing the information that is not currently used by the CPU.
- Figure below illustrates the **components in a typical memory hierarchy**



- The total memory capacity of a computer can be visualized as being a hierarchy of components.
- The memory hierarchy system consists of all storage devices employed in a computer system from the slowbut high-capacity auxiliary memory to a relatively faster main



memory, to an even smaller and faster cache memory accessible to the high-speed processing logic.

- The computer memory can be divided into 5 major hierarchies based on use and speed.
- They are: **Registers, Cache memory, Main memory, Magnetic disc, and Magnetic tape.**
- At the bottom of the hierarchy are the relatively slow magnetic tapes used to store removable files.
- Next are the magnetic disks used as backup storage.
- The main memory occupies a central position by being able to communicate directly with the CPU and with auxiliary memory devices through an IO processor.
- When programs not residing in main memory are needed by the CPU, they are brought in from auxiliary memory.
- Programs not currently needed in main memory are transferred into auxiliary memory to provide space for currently used programs and data.

CPU Register

- They are also known as Internal Processor Memory.
- The data or instructions which have to be executed are kept in these registers.
- The register is usually an static RAM (SRAM) in the computer processor that is used to hold the data word that is typically 64 bits or 128 bits.

Main Memory

- Main memory is CPU's memory unit that communicates directly.
- It's the primary storage unit of a computer system.
- The main memory is very fast and a very large memory that is used for storing the information throughout the computer's operations.
- This type of memory is made up of ROM as well as RAM.

Auxiliary Memory

- Devices that provide backup storage are called auxiliary memory.
- The most common auxiliary memory devices used in computer systems are magnetic disks and tapes.
- They are used for storing system, programs, large data files, and other backup information.
- Only programs and data currently needed by the processor reside in main memory.
- All other information is stored in auxiliary memory and transferred to main memory when needed.

Cache memory

- **Cache Memory_**is employed in computer system to compensate for the speed differential between main memory access time and processor.
- CPU is usually faster than main memory access time, with result that processing speed is limited primarily by the speed of main memory.
- The cache is used for storing segments of programs currently being executed in the CPU and temporary data frequently needed in the present calculations.

Multiprogramming

- Many operating systems are designed to enable the CPU to process a number of independent programs concurrently – known as multiprogramming.
- Sometimes a program is too long to be accommodated in total space available in main memory.
- A program with its data normally resides in auxiliary memory. When a program or a segment of program is to be executed, it is transferred to main memory to be executed by the CPU.
- The part of the computer system that supervises the flow of information between auxiliary and main memory is called the **memory management System**

Characteristics of Memory Hierarchy

1. Capacity

It refers to the total volume of data that a system's memory can store. The capacity increases moving from the top to the bottom in the hierarchy from registers to cache to main memory (RAM) to secondary storage (hard drives, SSDs)

2. Access Time

It refers to the time it takes to retrieve data from memory, which increases as you move down the hierarchy. Faster memory (like registers and cache) has shorter access times, while slower memory (like hard drives) has longer access time

3. Performance

A memory hierarchy aims to improve overall system performance by ensuring that frequently accessed data is stored in faster memory levels, reducing the time it takes to retrieve that data

4. Cost per bit

The cost of storing a single bit of data, which generally increases as you move up the hierarchy. Registers and cache are expensive, while main memory and secondary storage are more cost-effective.

COMPUTER ORGANIZATION AND ARCHITECTURE
UNIT –IV
MEMORY ORGANIZATION
TOPIC- MAIN MEMORY PART-1

Main Memory

Introduction

- Main memory is central storage unit in computer system.
- It is a relatively large and fast memory used to store programs and data during the computer operation.
- The principal technology used for main memory is based on semiconductor integrated circuits.

Random Access Memory (RAM)

The integrated circuit RAM chips are available in two possible operating models: **Static and Dynamic.**

- **Static RAM (SRAM)**: In an SRAM, binary values are stored using traditional flip-flop constructed with the help of transistors. A static RAM will hold its data as long as power is supplied to it.

  It is easier to use and extremely fast but it is very expensive**.**
  It is used in the design of cache memory.

- **Dynamic RAMs (DRAMs):** They are used for implementing the main memory. DRAMs stores data in the form of electric charges in small capacitors provided by CMOS transistors.
  DRAM offer reduced power consumption and larger capacity compared to SRAM but SRAM are faster.

Most of the main memory in a general purpose computer is made up of RAM integrated circuit chips, but a portion of the memory may be constructed with ROM chips.

Read Only Memory (ROM)

- **Read Only Memory** –Store programs that are permanently resident in the computer.
- The ROM portion of main memory is needed for storing an initial program called a **bootstrap loader.**
- Boot strap loader –function is start the computer software operating when power is turned on.
- Boot strap program loads a portion of operating system from disc to main memory and control is then transferred to operating system.

RAM and ROM Chips

1. RAM Chip

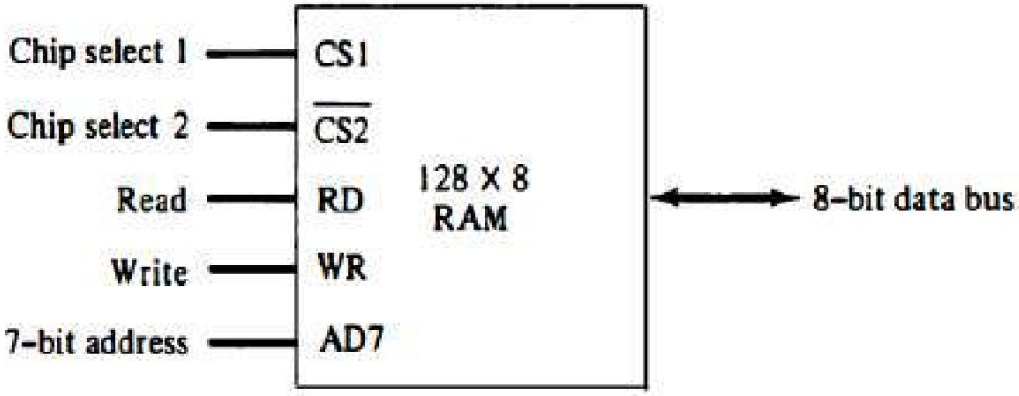
- A **RAM chip** is better suited for communication with the CPU.

- It uses a bidirectional data bus that allows the transfer of data either from memory to CPU during a read operation or from CPU to memory during a write operation.
- A **bidirectional bus** can be constructed with three-state buffers.
- A three-state buffer output can be placed in one of three possible states: a signal equivalent to logic 1, a signal equivalent to logic 0 or a high impedance state.

The **block diagram** of a RAM chip is shown in **Figure12-2.**
- The capacity of the memory is 128 words of eight bits (one byte) per word.
- This requires a 7-bit address and an 8-bit bidirectional data bus.
- The read and write inputs specify the memory operation and two chips select (CS) control inputs are for enabling the chip only when it is selected by the microprocessor.

The **function table listed** in Fig. 12-2(b) specifies the operation of the RAM chip.
- When **CS1=1 and $\overline{CS2}$ = 0**, the memory can be placed in a write or read mode.
- When the WR input is enabled, the memory stores a byte from the data bus into a location specified by the address input lines.
- When the RD input is enabled, the content of the selected byte is placed into the data bus.

Figure 12-2 Typical RAM chip.



(a) Block diagram

CSI	$\overline{CS2}$	RD	WR	Memory function	State of data bus
0	0	x	x	Inhibit	High-impedance
0	1	x	x	Inhibit	High-impedance
1	0	0	0	Inhibit	High-impedance
1	0	0	1	Write	Input data to RAM
1	0	1	x	Read	Output data from RAM
1	1	x	x	Inhibit	High-impedance

(b) Function table

2. ROM Chip

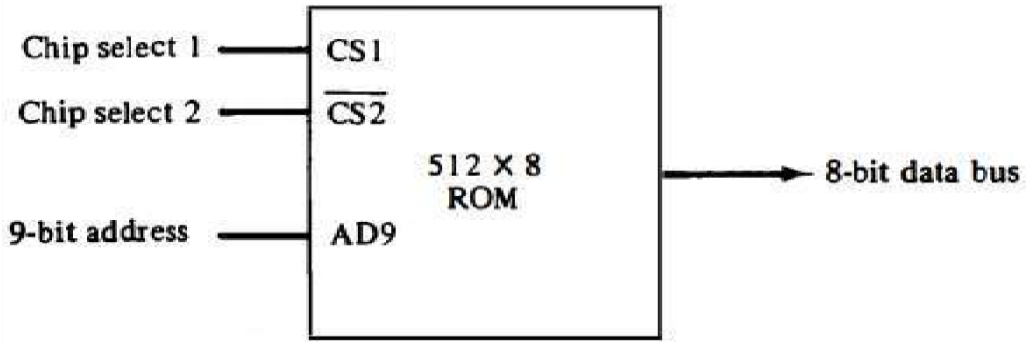


Figure 12-3 Typical ROM chip.

- The block diagram of a ROM chip is shown in Figure12-3.
- Since the ROM can only read, the data bus can only be in an output mode.
- No need of READ and WRITE control.
- The nine address lines in the ROM chip specify any one of the 512 bytes stored in it.
- The chip select inputs must be CS1=1 and $\overline{CS2}$=0 for the unit to operate. Otherwise, data bus is in a high-impedance state.

COMPUTER ORGANIZATION AND ARCHITECTURE
UNIT –IV
MEMORY ORGANIZATION
TOPIC- MAIN MEMORY PART-2

Main Memory

Memory connection to CPU

- RAM and ROM chips are connected to a CPU through the data and address buses.
- The connection of memory chips to the CPU is shown in **Figure 12-4**.
- This configuration gives a memory capacity of 512 bytes of RAM and 512 bytes of ROM.
- Each RAM receives the seven low-order bits of the address bus to select one of 128 possible bytes.
- This is done through a 2 X 4 decoder whose outputs go to the CS1 inputs in each RAM chip
- The particular RAM chip selected is determined from lines 8 and 9 in the address bus.
- When the address lines 8 and 9 are equal to 00, the first RAM chip is selected, and so on.
- The RD and WR outputs from the microprocessor are applied to the inputs of each RAM chip.
- The selection between RAM and ROM is achieved through bus line 10. The RAMs are selected when the bit in this line is 0 and the ROM when the bit is 1.
- The other chip select input in the ROM is connected to the RD control line for the ROM chip to be enabled only during a read operation.
- Address bus lines 1 to 9 are applied to the input address of ROM without going through the decoder.
- This assigns addresses 0 to 511 to RAM and 512 to 1023 to ROM.

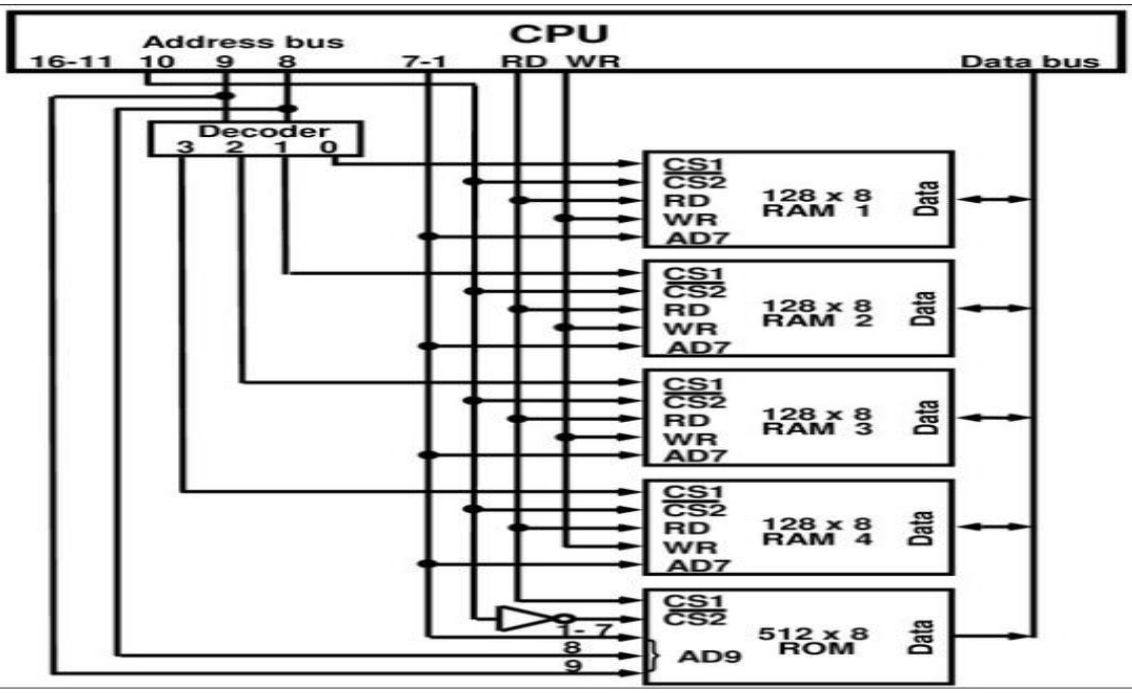


Figure 12.4: CONNECTION OF MEMORY TO CPU

Memory Address Map

- Memory Address Map is a pictorial representation of assigned address space for each chip in the system.
- Assume computer system with 512 bytes of RAM and 512 bytes of ROM.
- The memory address map is shown in **TABLE 12-1**.
- The component column specifies whether a RAM or a ROM chip is used.
- The hexadecimal address column assigns a range of hexadecimal equivalent addresses for each chip.
- The address bus lines are listed in the third column.
- The RAM chips have 128 bytes and need seven address lines.
- The ROM chip has 512 bytes and needs 9 address lines.
- When line 10 is 0, the CPU selects a RAM, and when this line is equal to 1, it selects the ROM.

TABLE 12-1 Memory Address Map for Microprocomputer

Component	Hexadecimal address	Address bus									
		10	9	8	7	6	5	4	3	2	1
RAM 1	0000–007F	0	0	0	x	x	x	x	x	x	x
RAM 2	0080–00FF	0	0	1	x	x	x	x	x	x	x
RAM 3	0100–017F	0	1	0	x	x	x	x	x	x	x
RAM 4	0180–01FF	0	1	1	x	x	x	x	x	x	x
ROM	0200–03FF	1	x	x	x	x	x	x	x	x	x

COMPUTER ORGANIZATION AND ARCHITECTURE
UNIT –IV
MEMORY ORGANIZATION
TOPIC- AUXILLARY MEMORY

Auxiliary Memory

Auxiliary memory is also known as secondary memory or non volatile memory that holds programs and data for long term storage and is not directly accessed by the CPU.

Devices that provide backup storage are called auxiliary memory.

For example: Magnetic disks and tapes are commonly used auxiliary devices. Other components not frequently used are magnetic drums, magnetic bubble memory, and optical disks.

It offers lower cost and higher capacity than primary memory(RAM).

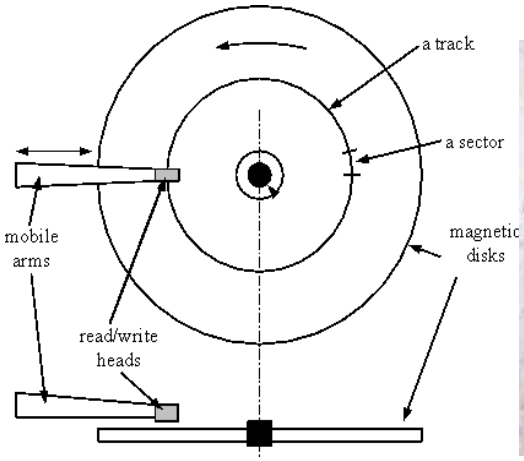
- The important characteristics of any device are its access mode, access time, transfer rate, capacity, and cost.
- The average time required to reach a storage location in memory and obtain its contents is called the access time.
- In electromechanical devices with moving parts such as disks and tapes, the access time consists of a seek time required to position the read-write head to a location and a transfer time required to transfer data to or from the device.
- A record is a specified number of characters or words.
- Reading or writing is always done on entire records.
- The transfer rate is the number of characters or words that the device can transfer per second, after it has been positioned at the beginning of the record.
- Magnetic drums and disks are quite similar in operation.
- Both consist of high-speed rotating surfaces coated with a magnetic recording medium.
- The rotating surface of the drum is a cylinder and that of the disk, a round flat plate.
- The recording surface rotates at uniform speed and is not started or stopped during access operations.
- Bits are recorded as magnetic spots on the surface as it passes a stationary mechanism called a write head.
- Stored bits are detected by a change in magnetic field produced by a recorded spot on the surface as it passes through a read head.
- The amount of surface available for recording in a disk is greater than in a drum of equal physical size.
- Therefore, more information can be stored on a disk than on a drum of comparable size.
- So, disks have replaced drums in more recent computers.

Magnetic disk

- Magnetic disk is a circular plate constructed of metal or plastic coated with magnetized material.
- Often both sides of the disk are used and several disks may be stacked on one spindle with read/write heads available on each surface.
- All disks rotate together at high speed and are not stopped or started for access

purposes.

- Bits are stored in the magnetized surface in spots along concentric circles called tracks.
- The tracks are commonly divided into sections called sectors.
- The subdivision of one disk surface into tracks and sectors is shown in Fig. 3-5.
- Some units use a single read/write head for each disk surface.
- In this type of unit, the track address bits are used by a mechanical assembly to move the head into the specified track position before reading or writing.
- In other disk systems, separate read/write heads are provided for each track in each surface. The address bits can then select a particular track electronically through a decoder circuit.
- This type of unit is more expensive and is found only in very large computer systems.
- A disk system is addressed by address bits that specify the disk number, the disk surface, the sector number and the track within the sector.
- After the read/write heads are positioned in the specified track, the system has to wait until the rotating disk reaches the specified sector under the read/write head.
- Information transfer is very fast once the beginning of a sector has been reached.
- Disks may have multiple heads and simultaneous transfer of bits from several tracks at the same time.
- A track in a given sector near the circumference is longer than a track near the center of the disk. If bits are recorded with equal density; some tracks will contain more recorded bits than others.
- To make all the records in a sector of equal length, some disks use a variable recording density with higher density on tracks near the center than on tracks near the circumference.
- This equalizes the number of bits on all tracks of a given sector.
- Disks that are permanently attached to the unit assembly and cannot be removed by the occasional userare called **hard disks**.



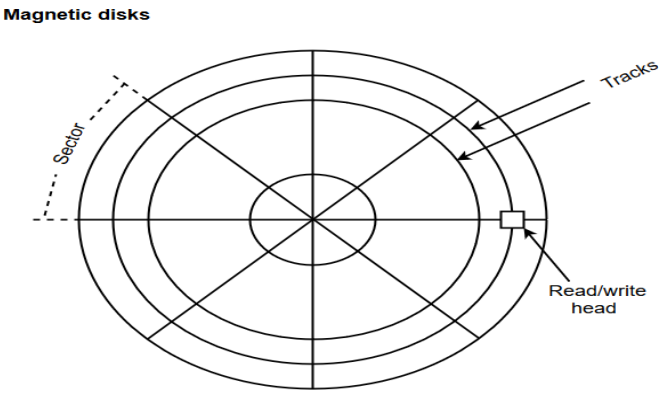
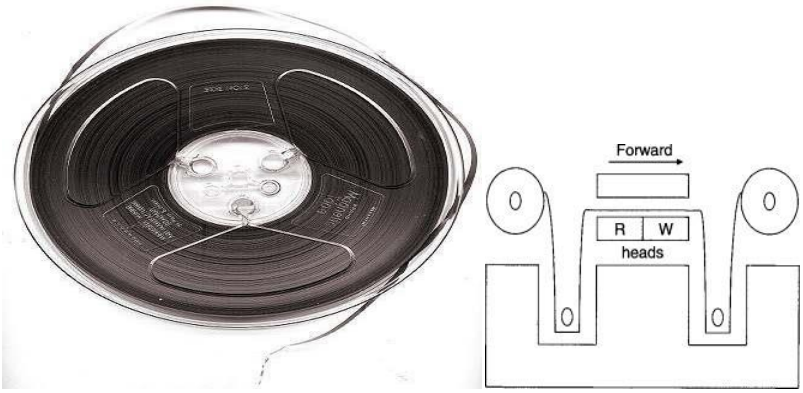


Figure 3.5: Magnetic Disk

- A disk drive with removable disks is called a **floppy disk**.
- These are made of plastic coated with magnetic recording material.
- There are two sizes commonly used, with diameters of 5.25 and 3. 5 inches.
- The 3.5-inch disks are smaller and can store more data than can the 5.25-inch disks.
- Floppy disks are extensively used in personal computers as a medium for distributing software to computer users.

Magnetic Tape

- A magnetic tape consists of the electrical, mechanical, and electronic components to provide the parts and control mechanism for a magnetic-tape unit.
- The tape itself is a strip of plastic coatedwith a magnetic recording medium.
- Bits are recorded as magnetic spots on the tape along several tracks.
- Usually, seven or nine bits are recorded simultaneously to form a character together with a parity bit.
- Read/write heads are mounted one in each track so that data can be recorded and read as a sequence of characters.
- Magnetic tape units can be stopped, started to move forward or in reverse, or can be rewound.
- Each record on tape has an identification bit pattern at the beginning and end.
- By reading the bit pattern at the beginning, the tape control identifies the record number.
- By reading the bit pattern at the end of the record, the control recognizes the beginning of a gap.
- A tape unit is addressed by specifying the record number and the number of characters in the record.
- Records may be of fixed or variable length.



COMPUTER ORGANIZATION AND ARCHITECTURE
UNIT –IV
MEMORY ORGANIZATION
TOPIC- ASSOCIATE MEMORY (OR)
CONTENT ADDRESSABLE MEMORY (CAM) PART-2

Associative Memory

Match Logic

- The match logic for each word can be derived from the comparison algorithm for two binary numbers.
- First, we neglect the key bits and compare the argument in A with the bits stored in the cells of the words.
- Word i is equal to the argument in A if $A_j = F_{ij}$ for j=1, 2, ...,n.
- Two bits are equal if they are both 1 or both 0. The equality of two bits can be expressed logically by the Boolean function

$$x_j = A_j F_{ij} + A_j' F_{ij}'$$

- Where $x_i = 1$ if the pair of bits in position j are equal; otherwise, $x_i = 0$.
- For a word i to be equal to the argument in A we must have all x_i variables equal to 1. This is the condition for setting the corresponding match bit M_i to 1. The Boolean function for this condition is

$$M_i = x_1 x_2 x_3 \cdots x_n$$

and constitutes the AND operation of all pairs of matched bits in a word.

- We now include the key bit K_j in the comparison logic. The requirement is that if K_j=0, the corresponding bits of A_j and F_{ij} need no comparison. Only when K_j= 1 must they be compared. This requirement is achieved by OR ing each term with K_j', thus:

$$x_j + K_j' = \begin{cases} x_j & \text{if } K_j = 1 \\ 1 & \text{if } K_j = 0 \end{cases}$$

- When $K_j = 1$, we have $K_j' = 0$ and $x_j + 0 = x_j$. When $K_j = 0$, then $K_j' = 1$ and x; + 1 = 1.
- A term ($x_j + K_j'$) will be in the 1 state if its pair of bits is not compared.
- This is necessary because each term is AND ed with all other terms so that an output of 1 will have no effect. The comparison of the bits has an effect only when $K_j = 1$.
- The match logic for word i in an associative memory can now be expressed by the following Boolean function:

$$M_i = (x_1 + K_1')(x_2 + K_2')(x_3 + K_3') \cdots (x_n + K_n')$$

- Each term in the expression will be equal to 1 if its corresponding $K_j = 0$. If $K_j = 1$, the term will be either 0 or1depending on the value of X_j.
- A match will occur and M_i will be equal to 1 if all terms are equal to 1.
- If we substitute the original definition of x_j, the Boolean function above can be expressed as follows:

$$M_i = \prod_{j=1}^{n} (A_j F_{ij} + A_j' F_{ij}' + K_j')$$

- Where π is a product symbol designating the AND operation of all n terms. We need m such functions, one for each word i=1, 2,3,...,m.

Applications:

- **Database Systems:** CAM can be used to speed up database searches.
- **Network Routing:** CAM can be used in routers to quickly find the correct path for data packets.
- **Image Processing:** CAM can be used to quickly search for specific patterns in images.

Advantages

- CAM is designed for very-high-speed searching applications where search time needs to be short.
- Suitable for parallel searches as it can search for multiple items simultaneously.
- Often used to speed up databases searches.

Disadvantages

- More expensive than RAM.
- Each cell must have strong capability and logical circuits for matching its content with external argument.

COMPUTER ORGANIZATION AND ARCHITECTURE
UNIT –IV
MEMORY ORGANIZATION
TOPIC- ASSOCIATE MEMORY (OR)
CONTENT ADDRESSABLE MEMORY (CAM) PART-1

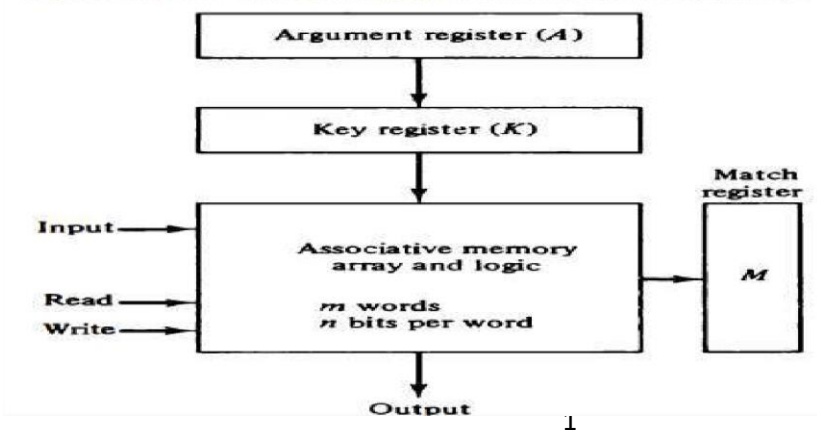
Associate Memory (or) Content Addressable Memory

- To search for a particular data in the memory, data is read from certain address and compared.
- If the match is not found, content of the next address is accessed and compared. This goes on until the required data is found.
- The number of access depends on the location of data and efficiency of searching algorithm.
- The time required to find an object stored in memory can be reduced considerably if the objects are based on their contents, not on their locations.

- A memory unit is accessed by content is called an **Associative memory or Content Addressable Memory (CAM).**
- This type of memory is accessed simultaneously and in parallel on the basis of data content rather than by specific address or location.
- When a write operation is performed on associative memory,no address or memory location is given to the word.
- The memory itself is capable of finding an empty unused location to store the word.
- On other hand, when a word is to be read from an associative memory, the content of the word, or part of the word, is specified.
- The words which match the specified content are located by the memory and are marked for reading.
- An associative memory is more expensive than a random access memory because each cell must have storage capability as well as logic circuits for matching its content with an external argument.
- For this reason, associative memories are used in applications where the search time is very critical, and must be very short.

Hardware Organization

Figure 12-6 Block diagram of associative memory.

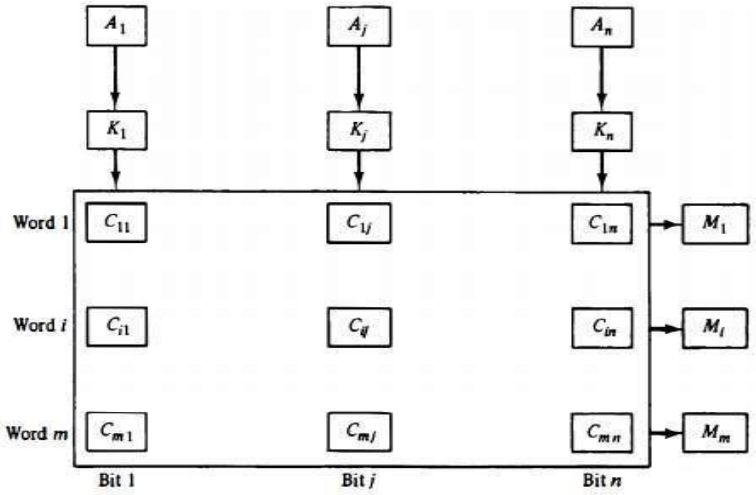


- The block diagram of an associative memory is shown in **Figure 12-6.**
- It consists of a memory array and logic for m words with n bits per word.
- The argument register A and key register k each have n bits, one for each bit of a word.
- The match register M has m bits, one for each memory word.
- Each word in memory is compared in parallel with the content of the **argument register**.
- The words that match the bits of the argument register set a corresponding bit in the **match register**.
- After the matching process, those bits in the match register that have been set indicate the fact that their corresponding words have been matched.
- Reading is accomplished by a sequential access to memory for those words whose corresponding bits in the match register have been set.
- The key register provides a mask for choosing a particular field or key in the argument word.
- As example, suppose that the argument register A and the key register K have the bit configuration shown below.
- Only the three leftmost bits of A are compared with memory words because K has 1's in these positions.

A	101 111100	
K	111 000000	
Word 1	100 111100	no match
Word 2	101 000001	match

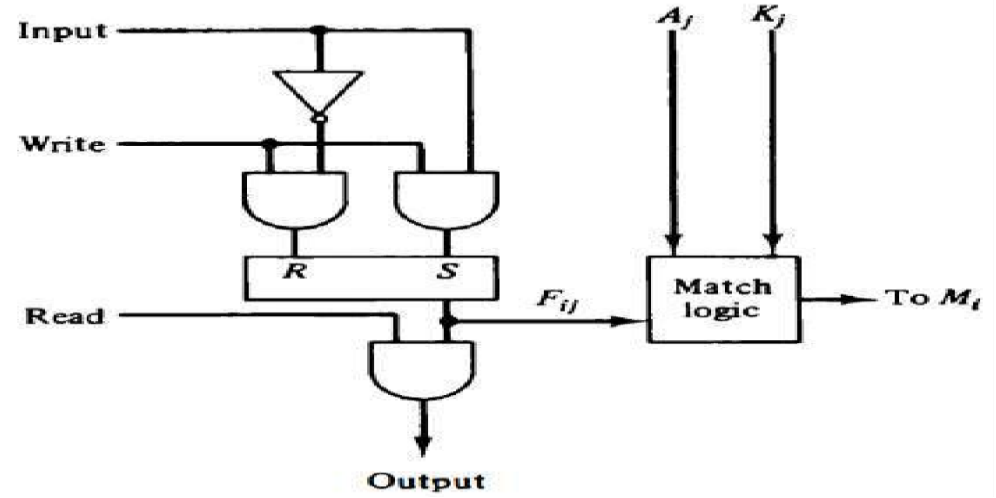
- Word 2 matches the unmasked argument field because the three left most bits of the argument and the words are equal.
- The relation between the memory array and external registers in an associative memory is shown in **Fig. 12-7**.
- The cell C_{ij} is the cell for bit j in word i. A bit A_j in the argument register is compared with all the bits in column j of the array provided that K_j =1.
- This is done for all columns j=1, 2… n. If a match occurs between all the unmasked bits of the argument and the bits in word i, the corresponding bit M_i in the match register is set to 1.
- If one or more unmasked bits of the argument and the word don't match, M_i is cleared to 0.

Figure 12-7 Associative memory of m word, n cells per word.



- The internal organization of a typical cell C_{ij} is shown in **Figure 12-8**.
- It consists of a flip-flop storage element F_{ij} and the circuit for reading, writing and matching the cell.
- The input bit is transferred into the storage cell during a write operation. The bit stored is read out during a read operation.
- The match logic compares the content of the storage cell with the corresponding unmasked bit of the argument and provides an output for the decision logic that sets the bit in M_i.

Figure 12-8 One cell of associative memory.



COMPUTER ORGANIZATION AND ARCHITECTURE
UNIT –IV
MEMORY ORGANIZATION
TOPIC- CACHE MEMORY

Cache Memory

●Cache memory is a small and very high speed semiconductor memory which can speed up and synchronizing with high-speed CPU.
●It acts as a buffer between the CPU and the main memory.
●It is used to reduce the average time to access the data from the main memory
●The data or contents of the main memory that are used frequently by CPU are stored in the cache memory so that the processor can easily access that data in a shorter time.
●Whenever the CPU needs to access memory, it first checks the cache memory. If the data is not found in cache memory, then the CPU moves into the main memory.
●The cache is the fastest component in the memory hierarchy and approaches the speed of CPU components.

Principles of Cache

● The block diagram for a cache memory can be represented as:

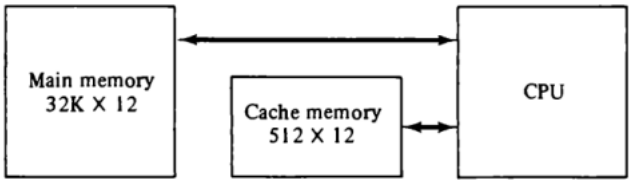


Figure 12-10 Example of cache memory.

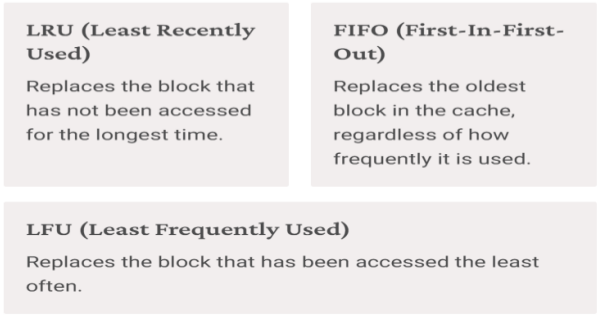
Fig.3.2. Example of Cache Memory

- The main memory can store 32k words of 12 bits each.
- The cache is capable of storing 512 of these words at any given time.
- For every word stored, there is a duplicate copy in main memory.
- The CPU communicates with both memories.

The basic operation of a cache memory is as follows:

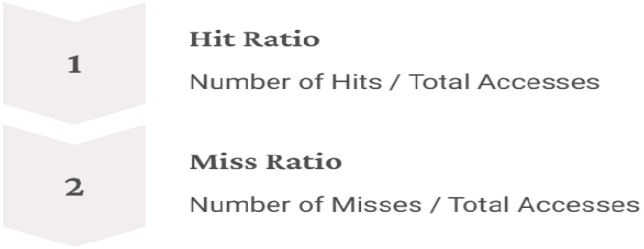
- When the processor needs to read or write a location in main memory, it first checks for a corresponding entry in the cache.
- It first sends a 15 bit address to cache.
- The cache is then examined. If the word is found in the cache (cache hit), it is read from the cache memory.
- If the word addressed by the CPU is not found in the cache (cache miss), the main memory is accessed to read the word.
- When a read request is received by the main memory from CPU, contents of a block of memory words containing the location specified are transferred in to cache.
- For a cache miss, the cache allocates a new entry and copies in data from main memory, and then the request is fulfilled from the contents of the cache.
- Assume cache is full and memory word not in cache is referenced.

● Control hardware decides which block from cache is to be removed to create space for new block containing referenced word from memory.
● Collection of rules for making this decision is called "Replacement algorithm "



Measuring cache performance: Hit Ratio

●The performance of cache memory is frequently measured in terms of a quantity called **Hit ratio.**
●When the CPU refers to memory and finds the word in cache, it is said to produce a **hit or cache hit**.
●If the word is not found in the cache, it is in main memory and it counts as a **miss or cache miss**. **(CPU must read main memory).**



●A higher hit ratio indicates better cache performance, as it means the CPU is spending less time waiting for data.
●The goal of any cache system is to maximize the hit ratio.

Advantages of cache memory

- Cache memory is faster than main memory.
- It consumes less access time as compared to main memory. So programs can be executed within a short period of time.
- It stores the most recent data.

Disadvantages of cache memory

- Cache memory has limited capacity.
- It is very expensive than RAM.
- Complexity in management.
- Increased power consumption.
- Potential for data inconsistency.

COMPUTER ORGANIZATION AND ARCHITECTURE
UNIT –IV
MEMORY ORGANIZATION
TOPIC- CACHE MEMORY MAPPING

Cache Memory

Cache Mapping

- Cache mapping refers to a technique using which the content present in the main memory is brought into the cache.
- It helps us to define how a certain block that is present in the main memory gets mapped to the memory of a cache in the case of any cache miss.
- There are three different types of mapping used which are as follows:

1. Associative mapping,
2. Direct mapping and
3. Set-Associative mapping.

1. Associative mapping

- It is the fastest and most flexible cache organization uses an associative memory. The associative memory stores both the address and content (data) of the memory word.
- The address value of 15 bits is a five-digit octal number and its corresponding 12 -bit word is a four-digit octal number.
- A CPU address of 15 bits is placed in the argument register and the associative memory is searched for a matching address. if the address is found, the corresponding 12-bit data is read and sent to the CPU.
- If no match occurs, the main memory is accessed for the word.
- The address-data pair is then transferred to the associative cache memory.
- If the cache is full, an address-data pair must be displaced to make room for a pair that is needed and not presently in the cache.
- The decision as to what pair is replaced is determined from the replacement algorithm that the designer chooses for the cache.

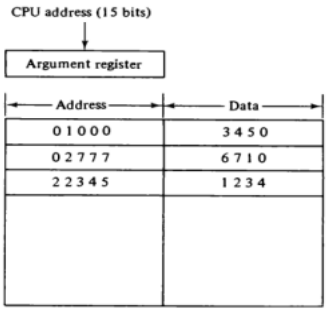


Fig.3.3. Associative Mapping Cache

If cache is full need to make a free slot than some data needs to be removed
1.FIFO
2.LRU(least recently used)
3.ROUND ROBIN

Advantages

Associative mapping overcomes the limitations of direct mapping by allowing each block of main memory to be placed in any line of the cache. This offers greater flexibility and reduces the conflict rate.

- Low conflict rate
- More efficient

Disadvantages

Hardware required to implement associative mapping is more complex and costly than direct mapping, as it requires a comparison of the tag field of the CPU's address with the tag of every line in the cache.

- More complex to implement
- Higher cost

2. Direct mapping

- Direct mapping is the simplest cache mapping technique.
- Associative memories are expensive compared to random-access memories because of the added logic associated with each cell.
- The CPU address of 15 bits is divided into two fields.
- The nine least significant bits constitute the index field and the remaining six bits form the tag field.

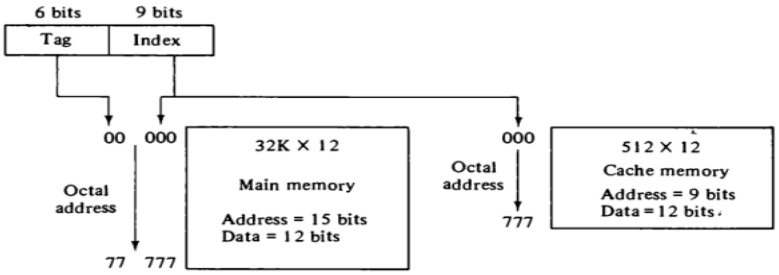


Fig.3.4. Addressing relationship between Main and Cache Memory

- The main memory needs an address that includes both the tag and the index bits.
- Each word in cache consists of the data word and its associated tag.
- When a new word is first brought into the cache, the tag bits are stored alongside the data bits.
- When the CPU generates memory request, the index field is used for the address to access the cache.
- The tag field of the CPU address is compared with the tag in the word read from the cache.
- If the two tags match, there is a hit and the desired data word is in cache. If there is no match, there is a miss and the required word is read from main memory.
- It is then stored in the cache together with the new tag, replacing the previous valu
- The disadvantage of direct mapping is that the hit ratio can drop considerably if two or more words whose addresses have the same index but different tags are accessed repeatedly.

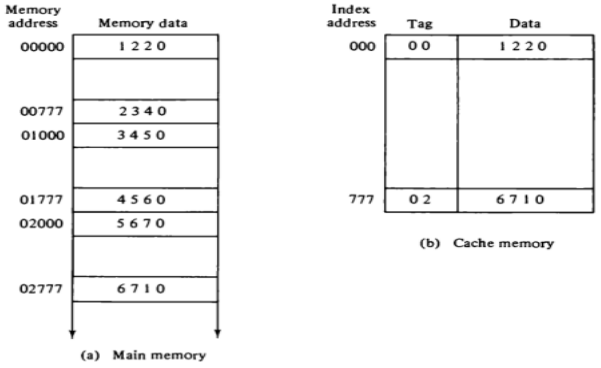


Fig.3.5. Direct Mapping cache organization

Direct mapping is the simplest cache mapping technique. But it may lead to high conflict rate means that multiple memory blocks frequently map to the same cache location, leading to frequent cache misses as one block overwrites the other, even when the cache has capacity.

Advantages

- Simple to implement
- Low cost

Disadvantages

- High conflict rate
- Not very efficient

3.Set-Associative mapping.

- Disadvantage of direct mapping is that two words with the same index in their address but with different tag values cannot reside in cache memory at the same time.
- A third type of cache organization, called set-associative mapping, is an improvement over the direct mapping organization in that each word of cache can store two or more words of memory under the same index address.
- Each data word is stored together with its tag and the number of tag-data items in one word of cache is said to form a set.

Index	Tag	Data	Tag	Data
000	01	3450	02	5670
777	02	6710	00	2340

- Each index address refers to two data words and their associated tags. Each tag requires six bits and each data word has 12 bits, so the word length is 2(6 + 12) = 36 bits.
- Thus, the size of cache memory is 512x36. It can accommodate 1024 words of main memory since each word of cache contains two data words.
- In general, a set-associative cache of set size k will accommodate k words of main memory in each word of cache.
- The words stored at addresses 01000 and 02000 of main memory are stored in cache

memory at index address 000.

- Similarly, the words at addresses 02777 and 00777 are stored in cache at index address 777.
- When the CPU generates a memory request, the index value of the address is used to access the cache.
- The tag field of the CPU address is then compared with both tags in the cache to determine if a match occurs.
- The comparison logic is done by an associative search of the tags in the set similar to an associative memory search: thus the name "set-associative."
- The hit ratio will improve as the set size increases because more words with the same index but different tags can reside in cache.
- An increase in the set size increases the number of bits in words of cache and requires more complex comparison logic.
- When a miss occurs in a set-associative cache and the set is full, it is necessary to replace one of the tag-data items with a new value.
- The most common replacement algorithms used are: random replacement, first-in, first out (FIFO), and least recently used (LRU).

Advantages

- Improved hit ratio.
- Flexible storage.
- Efficient memory access.
- Reduces the conflict rate compared to direct mapping while still being less complex and costly than fully associative mapping.

Disadvantages

- Increased complexity since it requires more complex comparison logic.
- Higher cost.
- More power consumption.

COMPUTER ORGANIZATION AND ARCHITECTURE
UNIT –IV
INPUT-OUTPUT ORGANIZATION
TOPIC- INPUT-OUTPUT INTERFACE

Input-Output Interface

- Peripherals connected to a computer need special communication links for interfacing with CPU.
- In computer system, there are special hardware components between the CPU and peripherals to control or manage the input-output transfers. These components are called input output interface units.
- They provide communication links between processor bus and peripherals. They provide a method for transferring information between internal system and input-output devices.
- Input-output interface provides a method for transferring information between internal storage and external I/O devices.
- The main purpose of I/O interface is to resolve differences between CPU and the peripherals.

Need for I/O Interface

- The main purpose of I/O interface is to resolve differences between CPU and the peripherals which are:

1. Peripherals are electromechanical and electromagnetic devices and their way of operation is different from the operation of the CPU and memory which are electronic devices. Therefore, a conversion of signal values may be required.
2. The data transfer rate of peripherals is usually slower than the transfer rate of the CPU and consequently, a synchronization mechanism may be needed.
3. Data codes and formats in peripherals differ from the word format in the CPU and memory.
4. The operating modes of peripherals are different from each other and each must be controlled so as not to disturb the operation of other peripherals connected to the CPU.

- To resolve these differences, computer systems include special hardware components between the CPU and peripherals to supervise and synchronize all input and output transfers.
- These components are called interface units because they interface between the processor bus and the peripheral device.

I/O Bus and Interface Modules

- I/O Bus and Interface Modules synchronize the data flow & supervisor the transfer between the CPU & peripherals.
- A typical communication link between the processor and several peripherals shown in Figure. 11-1.

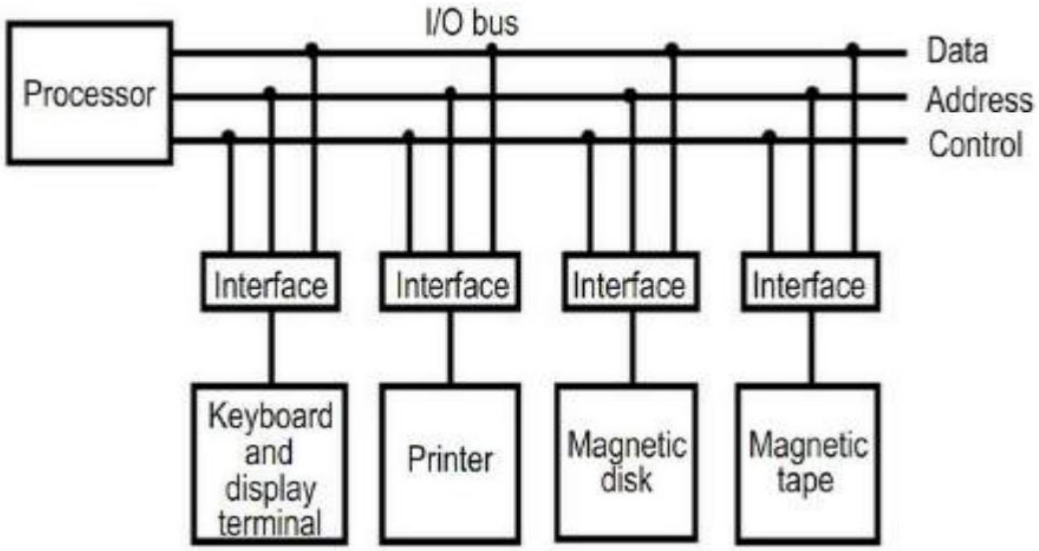


Figure 11-1 Connection of I/O bus to input-output devices.

- The I/O bus consists of data lines, address lines and control lines.
- The I/O bus from the processor is attached to all peripheral interfaces.
- To communicate with a particular device, the processor places a device address on the address lines.
- Each interface attached to the I/O bus contains an address decoder that monitors the address lines.
- When the interface detects its own address, it activates the path between the bus lines and the device that it controls.

Instructions executed in the interface and its attached peripheral devices are known as I/O commands as following:

1. A **Control command** is issued to activate the peripheral and to inform it what to do. For (a magnetic tape unit may be instructed to backspace tape by one record, to rewind the tape, or to start the tape moving in forward direction).

2. A **Status command** is used to test various status conditions in the interface and the peripheral (the computer may wish to check the status of the peripheral before a transfer is initiated).

3. A **Data output command** causes the interface to respond by transferring from the bus into one of its registers.

4. **Data input command** is the opposite of the data output. In this case interface receives an item of data from the peripheral and places it in its buffer register.

Functions of an I/O interface

1. Speed Synchronization

The interface ensures that the CPU's operating speed is synchronized with the input-output devices. This prevents data loss due to speed mismatches.

2. Processor Communication

The interface accepts and decodes commands from the processor, reports the current status, and recognizes its unique address.

3. Signal Control

It generates and manages control and timing signals needed for data transfer, ensuring smooth communication between the CPU and peripherals.

4. Data Buffering

The interface enables buffering, which temporarily stores data as it moves between devices and the CPU, helping manage the difference in processing speeds.

5. Error Detection

The interface can detect errors in data transmission, ensuring that errors are flagged and corrected before they affect system performance.

6. Data Conversion

It converts serial data to parallel data and vice versa, as well as converting digital data to analog signals and vice versa, and ensures the format is compatible with the receiving device.

7. Status Reporting

The interface reports the current status of the peripheral device to the processor.

I/O versus Memory Bus

I/O Bus: Communication between CPU and all interface units is via a common I/O bus. An interface connected to a peripheral device may have a number of data registers, a control register, and a status register. A command is passed to the peripheral by sending to the appropriate interface register

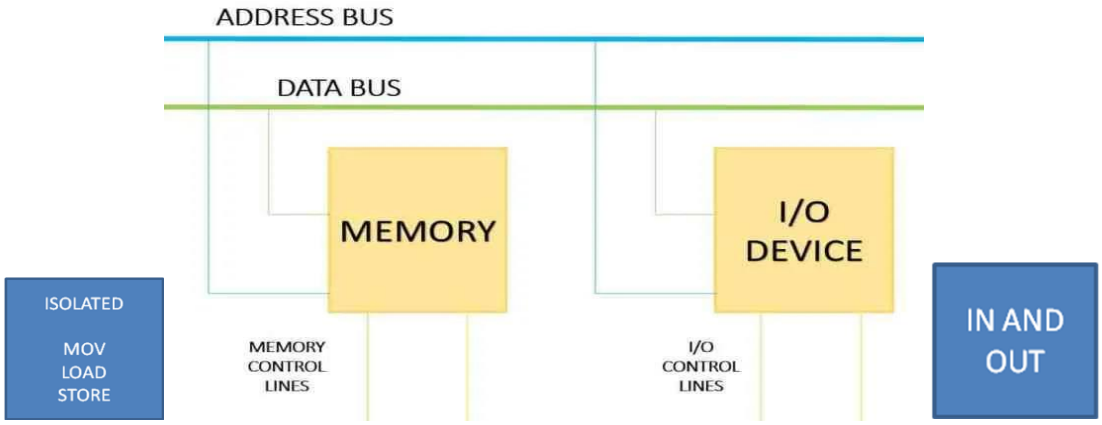
Memory Bus: used for information transfers between CPU and the main memory.

There are three ways that computer buses a used to communicate with memory and I/O:

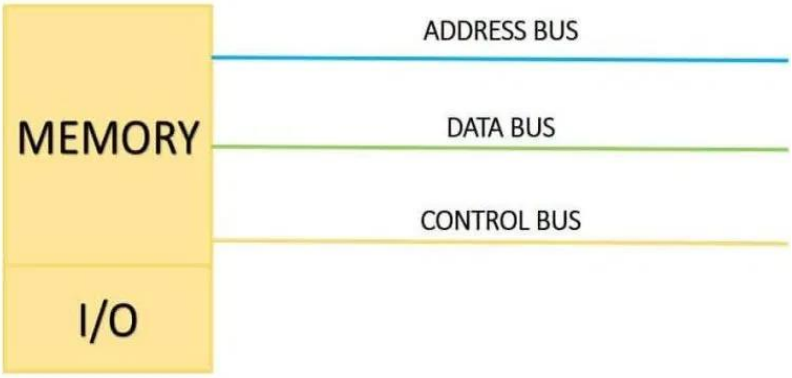
1. Use two separate buses, one for memory and the other for I/O.
2. Use one common bus for both memory and I/O but have separate control lines for each.
3. Use one common bus for memory and I/O with common control lines.

Isolated versus Memory-Mapped I/O

- Many computers use one common bus to transfer information between memory or I/O and the CPU.
- The distinction between a memory transfer and I/O transfer is made through separate read and write lines.
- The CPU specifies whether the address on the address lines is for a memory word or for an interface register by enabling one of two possible read or write lines.
- The I/O read and I/O write control lines are enabled during an I/O transfer.
- The memory read and memory write control lines are enabled during a memory transfer.
- This configuration isolates all I/O interface addresses from the addresses assigned to memory and is referred to as the isolated I/O.



- The other alternative is to use the same address space for both memory and I/O.
- This is the case in computers that employ only one set of read and write signals and do not distinguish between memory and I/O addresses.
- This configuration is referred to as memory-mapped I/O.
- The computer treats an interface register as being part of the memory system.
- The assigned addresses for interface registers cannot be used for memory words, which reduce the memory address range available.



Differences between Isolated I/O and Memory Mapped I/O.

Isolated I/O	No.	Memory Mapped I/O
Isolated I/O uses separate memory space.	01	Memory mapped I/O uses memory from the main memory.
Limited instructions can be used. Those are IN, OUT, INS, OUTS.	02	Any instruction which references to memory can be used.
The addresses for Isolated I/O devices are called ports.	03	Memory mapped I/O devices are treated as memory locations on the memory map.
IORC & IOWC signals expands the circuitry.	04	*IORC & IOWC* signals has no functions in this case which reduces the circuitry.
Efficient I/O operations due to using separate bus	05	Inefficient I/O operations due to using single bus for data and addressing
Comparatively larger in size	06	Smaller in size
Uses complex internal logic	07	Common internal logic for memory and I/O
Slower operations	08	Faster operations

COMPUTER ORGANIZATION AND ARCHITECTURE
UNIT –IV
INPUT-OUTPUT ORGANIZATION
TOPIC- ASYNCHRONOUS DATA TRANSFER PART-2

Asynchronous Data Transfer

Asynchronous Serial Transmission

- The transfer of data between two units is serial or parallel.
- In parallel data transmission, n bit in the message must be transmitted through n separate conductor path.
- In serial transmission, each bit in the message is sent in sequence one at a time.
- Parallel transmission is faster but it requires many wires.
- It is used for short distances and wherespeed is important. Serial transmission is slower but is less expensive.
- In Asynchronous serial transfer, each bit of message is sent a sequence at a time, and binary information is transferred only when it is available.
- When there is no information to be transferred, lineremains idle.
- A serial asynchronous data transmission technique used in many interactive terminals employs special bits which are inserted at both ends of the character code.

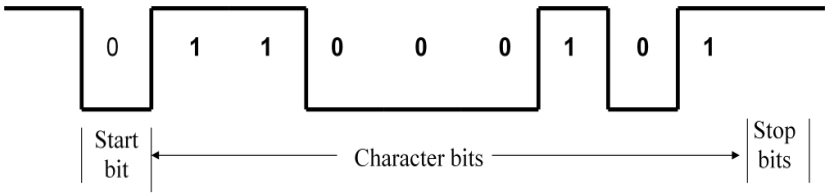
- With this technique each character consists of three parts:

- **Start bit:** is always a 0 and is used to indicate the beginning of a character.

- **Character bits:** Bits in between the start bit and the stop bit are known as character bits. Thecharacter bits always follow the start bit.

- **Stop bits:** Last bit, called stop bit is always one and used to indicate end of characters.

An example of this format is shown in Figure.



Asynchronous Serial Transmission

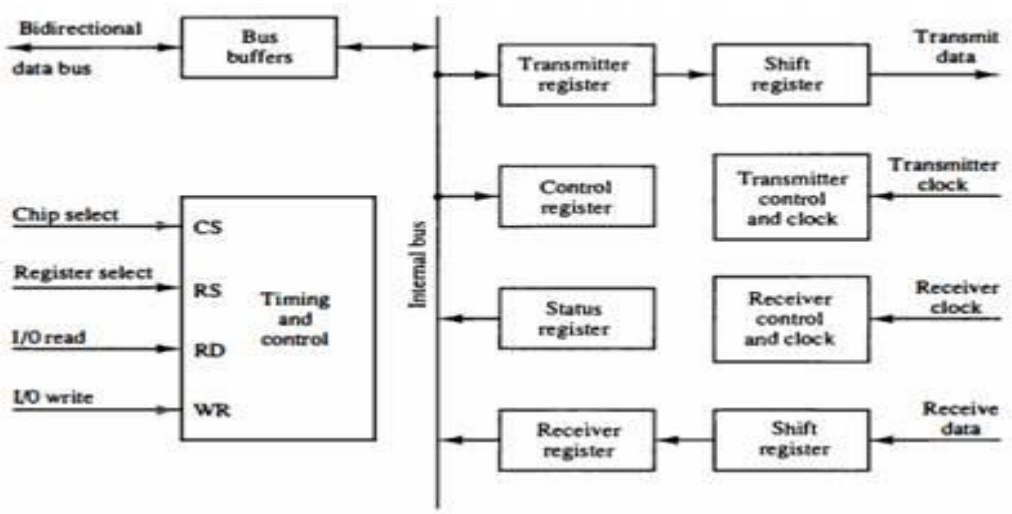
Serial Transmission of Asynchronous is done by two ways:

a) Asynchronous Communication Interface

b) First In First out Buffer

Asynchronous Communication Interface:

- A typical block diagram of asynchronous communication interface is shown below:



CS	RS	Operation	Register selected
0	×	×	None: data bus in high-impedance
1	0	WR	Transmitter register
1	1	WR	Control register
1	0	RD	Receiver register
1	1	RD	Status register

Figure 11-8 Block diagram of a typical asynchronous communication interface.

- It works as both a receiver and a transmitter.
- Its operation is initialized by CPU by sending a byte to the control register.
- The transmitter register accepts a data byte from CPU through the data bus and transferred to a shift register for serial transmission.
- The receive portion receives information into another shift register, and when a complete data byte is received it is transferred to receiver register.
- CPU can select the receiver register to read the byte through the data bus. Data in the status register is used for input and output flags.

First In First Out Buffer (FIFO):

- A First In First out (FIFO) Buffer is a memory unit that stores information in such a manner that thefirst item is in the item first out.
- A FIFO buffer comes with separate input and output terminals.
- The important feature of this buffer is that it can input data and output data at two

different rates.

- When placed between two units, the FIFO can accept data from the source unit at one rate, rate of transfer and deliver the data to the destination unit at another rate.
- If the source is faster than the destination, the FIFO is useful for source data arrive in bursts that fills out the buffer.
- FIFO is useful in some applications when data are transferred asynchronously.

COMPUTER ORGANIZATION AND ARCHITECTURE
UNIT –IV
INPUT-OUTPUT ORGANIZATION
TOPIC- ASYNCHRONOUS DATA TRANSFER PART-1

Asynchronous Data Transfer

- In a computer system, CPU and an I/O interface are designed independently of each other.
- When internal timing in each unit is independent from the other and when registers in interface and registers of CPU uses its own private clock.
- In that case the two units are said to be asynchronous to each other. CPU and I/O device must coordinate for data transfers.

Methods Used In Asynchronous Data Transfer

Strobe Control: This is one way of transfer i.e. by means of strobe pulse supplied by one of the units to indicate to the other unit when the transfer has to occur.

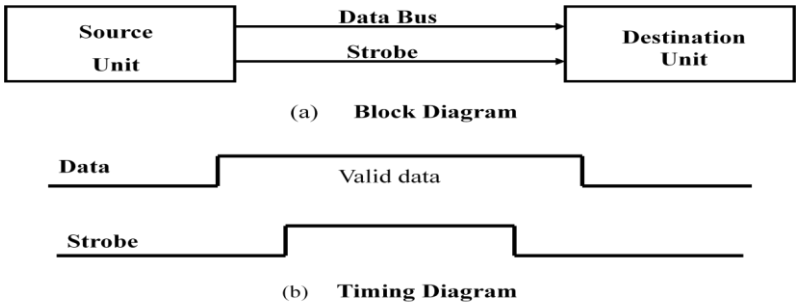
Handshaking: This method is used to accompany each data item being transferred with a control signal that indicates the presence of data in the bus. The unit receiving the data item responds with another control signal to acknowledge receipt of the data.

Strobe Control

- Asynchronous data transfer between two independent units requires that control signals be transmitted between the communicating units to indicate the time at which data is being transmitted.
- One way of achieving this is by means of a strobe pulse supplied by one of the units to indicate to the other unit when the transfer has to occur.
- The strobe may be activated by either source or the destination unit.
- The Strobe signal is disabled indicates that the data bus does not contain valid data.
- New valid data will be available only after the strobe is enabled again.
- Strobe control method of data transfer uses a single control signal for each transfer.
- The strobe may be activated by either the source unit or the destination unit in 2 ways:.
    - Source Initiated Strobe
    - Destination Initiated Strobe

Data Transfer Initiated by Source Unit

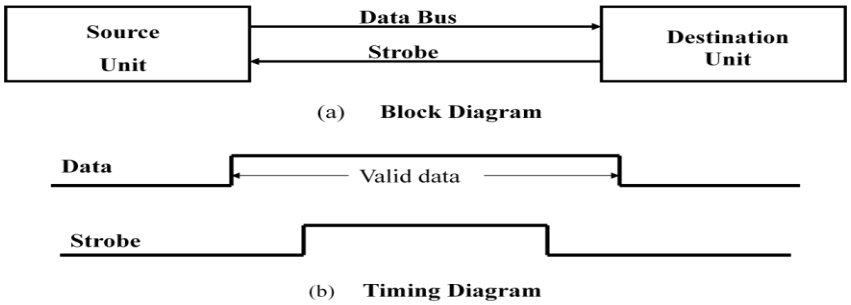
- In the block diagram in fig. (a), the data bus carries the binary information from source to destination unit.
- Typically, the bus has multiple lines to transfer an entire byte or word.
- The strobe is a single line that informs the destination unit when a valid data word is available.



Source-Initiated strobe for Data Transfer

- The timing diagram in fig. (b) the source unit first places the data on the data bus.
- The information on the data bus and strobe signal remains in the active state to allow the destination unit to receive the data.

Data Transfer Initiated by Destination Unit



Destination-Initiated strobe for Data Transfer

- In this method, the destination unit activates the strobe pulse, to informing the source to providethe data.
- The source will respond by placing the requested binary information on the data bus.
- The data must be valid and remain in the bus long enough for the destination unit to accept it.
- When accepted ,the destination unit then disables the strobe and the source unit removes the data from the bus.

Disadvantage of Strobe control

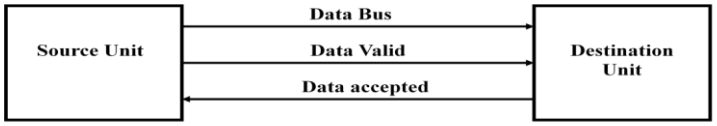
- The disadvantage of the strobe method is that, the source unit initiates the transfer has no way of knowing whether the destination unit has actually received the data item that was places in the bus.
- Similarly, a destination unit that initiates the transfer has no way of knowing whether the source unithas actually placed the data on bus.
- The Handshaking method solves this problem.

Handshaking method

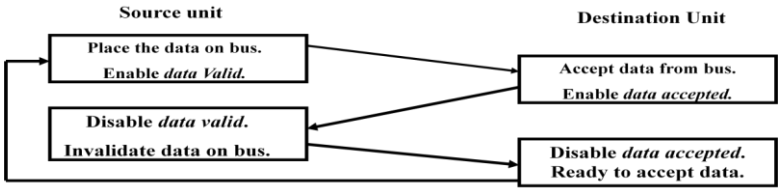
- Another method commonly used is to accompany each data item being transferred with a control signal that indicates the presence of data in the bus.
- The unit receiving the data item responds with another control signal to acknowledge receipt of the data.
- This type of agreement between two independent units is referred to as handshaking. The handshaking may be activated by either source or the destination unit.
- The handshaking method solves the problem of strobe method by introducing a second control signal that provides a reply to the unit that initiates the transfer.
- There are two control lines in handshaking technique:
    - Source to Destination unit.
    - Destination to source unit.

Source Initiated Transfer using Handshaking

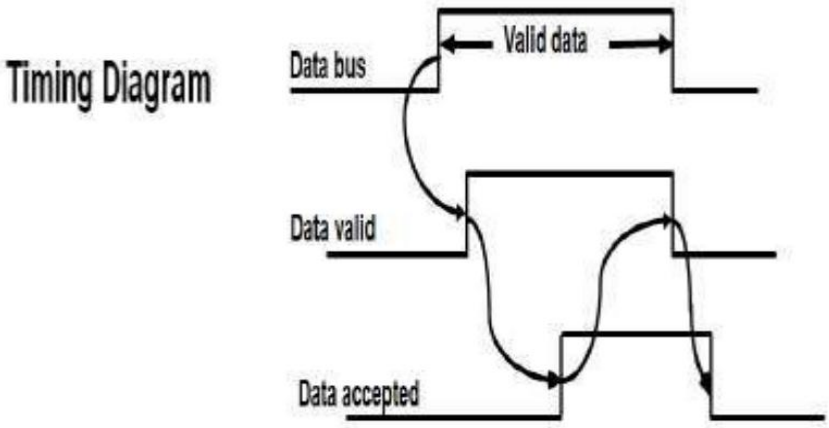
- The source unit initiates the transfer by placing the data on the bus and enabling its *data valid* signal.
- The *data accepted* signal is activated by the destination unit after it accepts the data from the bus.
- The source unit then disables its *data valid* signal and the destination disables the data accepted signal and the system goes into its initial state.



(a) Block Diagram

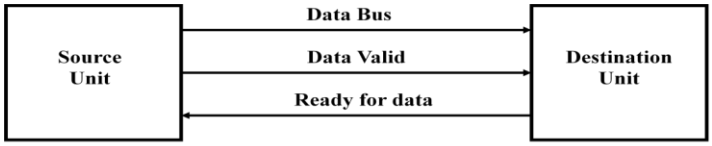


(b) Sequence of events

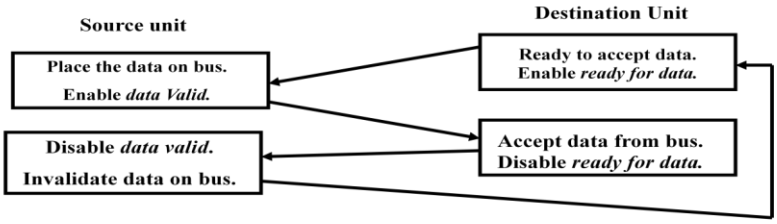


Destination Initiated Transfer Using Handshaking

- The name of the signal generated by the destination unit has been changed to *ready for data* to reflect its new meaning.
- The source unit in this case does not place data on the bus until after it receives the *ready for data* signal from the destination unit.
- From there on, the handshaking procedure follows the same pattern as in the source initiated case.
- The only difference between the Source Initiated and the Destination Initiated transfer is in theirchoice of Initial sate.

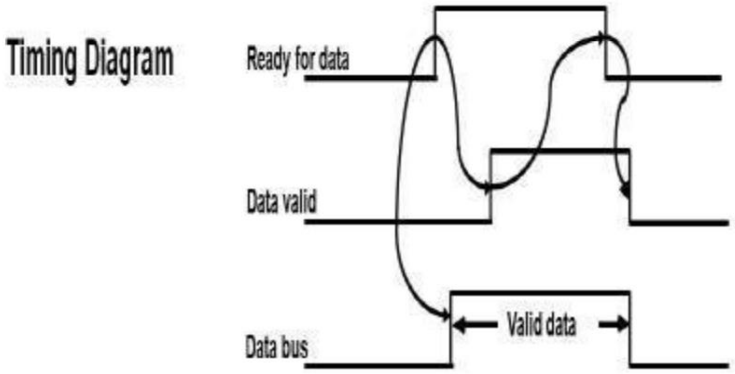


(a) Block Diagram



(b) Sequence of events

Destination-Initiated transfer using Handshaking



Advantage of the Handshaking method

- The Handshaking scheme provides degree of flexibility and reliability because the successful completion of data transfer relies on active participation by both units.
- If any of one unit is faulty, the data transfer will not be completed. Such an error can be detected by means of a *Timeout mechanism* which provides an alarm if the data is not completed within time.

COMPUTER ORGANIZATION AND ARCHITECTURE
UNIT –IV
INPUT-OUTPUT ORGANIZATION
TOPIC- MODES OF TRANSFER, PROGRAMMED I/O

Modes of Transfer

- The binary information that is received from an external device is usually stored in the memory unit.
- The information that is transferred from the CPU to the external device is originated from the memory unit. CPU merely processes the information but the source and target is always the memory unit.
- Data transfer between CPU and the I/O devices may be done in different modes.
- some of the modes use CPU as an intermediate path, others transfer the data directly to and from the memory unit and this can be handled by 3 following ways:
    1. Programmed I/O
    2. Interrupt-initiated I/O
    3. Direct Memory Access (DMA)

Programmed I/O

- Programmed 1/O operations are the result of I/O instructions written in the computer program.
- Each data item transfer is initiated by an instruction in the program.
- Transferring data under programmed I/O requires constant monitoring of the peripheral by the CPU.
- Once a data transfer is initiated, the CPU is required to monitor the interface to see when a transfer can again be made.
- In the programmed I/O method, the I/O device does not have direct access to memory.
- It is used to transfer data between I/O and memory with the intervention of the CPU.
- The device transfers bytes of data one byte 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 refer to as an F or “flag” bit to 1.
- 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.
- This is according to the handshaking procedure established.
- 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.

- Similarly the CPU can access I/O devices using address placed on the address bus.
- I/O read and I/O write to enable read and write operations by the CPU.

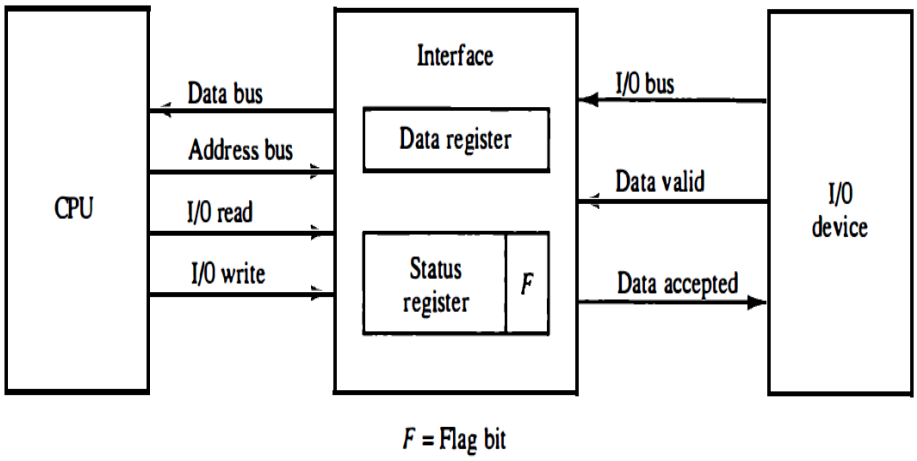


Figure: Data Transfer from I/O to CPU.

- It is assumed that the device is sending a sequence of bytes that must be stored in memory. 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 3 if set.
3. Read the data register.

- A flow chart of the program that must be written for the CPU is shown in Figure. 11-11.

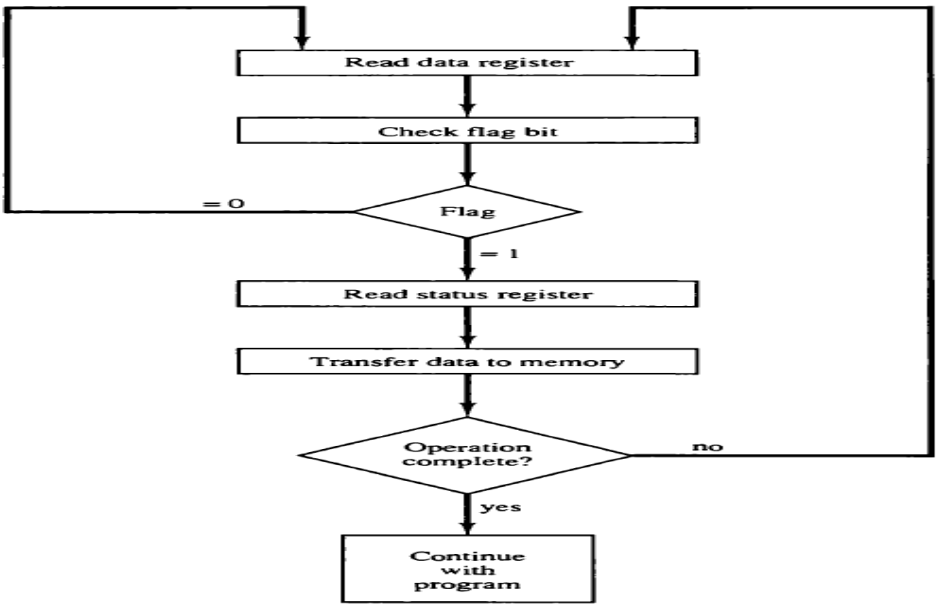


Figure 11-11 Flow chart for CPU program to input data

- CPU checks the status register of I/O interface for flag bit.
- If flag=1, it reads the data register & transfers to the memory.
- If flag=0, it again loops and keep checking until flag=1.
- It is the responsibility of the CPU to check the flag bits & not interface.
- If flag=0, CPU will continuously go to busy wait not performing any operation.

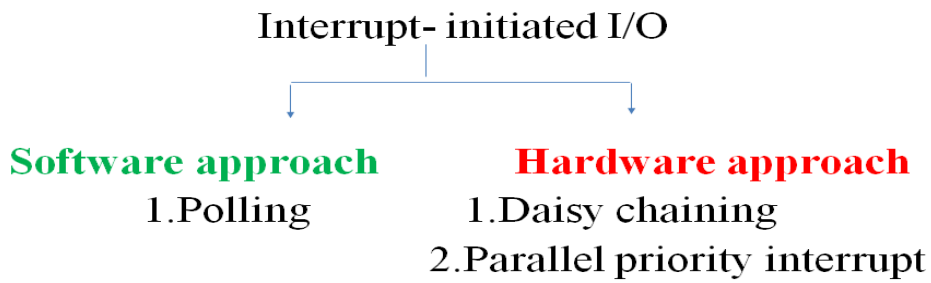
Drawbacks of the Programmed I/O

- It is very in-efficient approach.
- The main drawback of the Program Initiated I/O was that the CPU has to monitor the units all the times when the program is executing.
- Thus the CPU stays in a program loop until the I/O unit indicates that it is ready for data transfer.
- This is a time consuming process and the CPU time is wasted a lot in checking the flag bits rather than executing the programs.
- To remove this problem an Interrupt Initiated I/O is used.

COMPUTER ORGANIZATION AND ARCHITECTURE
UNIT –IV
INPUT-OUTPUT ORGANIZATION
TOPIC- INTERRUPT INITIATED I/O (OR) PRIORITY INTERRUPT

Interrupt Initiated I/O (Or) Priority Interrupt

- In this method an interrupt facility an interrupt command is used to inform the device about the start and end of transfer.
- In the meantime the CPU executes other program. When the interface determines that the device is ready for data transfer it generates an Interrupt Request and sends it to the computer.
- When the CPU receives such a signal, it temporarily stops the execution of the program and branches to a service program to process the I/O transfer and after completing it returns back to task, what it was originally performing.
- A priority interrupt is a system that establishes a priority over the various sources to determine which condition is to be serviced first when two or requests arrive simultaneously. The system may also determine which conditions are permitted to interrupt the computer while another interrupt is being serviced.
- Establishing the priority of simultaneous interrupts can be done by software or hardware.



Software priority interrupts: Polling

- A program is used to check (poll) if high priority program gave the interrupt signal or not .
- If higher priority device generated the interrupt signal than CPU executes the ISR program if not CPU checks for the higher priority device.

0 1 2 3

(H) (L)

- First 0 will be checked by CPU.....
- Two addresses:
- VAD: Vector Address: Every device has its ISR program address
- NVAD: Non-Vector Address: ISR Program stored at a fixed address.
- The disadvantage of the software method is that if there are many interrupts, the time required to poll them can exceed the time available to service the I/O device.
- In this situation a hardware priority-interrupt unit can be used to speed up the

operation.

Hardware priority interrupts:

- A hardware priority-interrupt unit functions as an overall manager in an interrupt system environment.
- It accepts interrupt requests from many sources, determines which of the incoming requests has the highest priority, and issues an interrupt request to the computer based on this determination.
- To speed up the operation, each interrupt source has its own interrupt vector to access its own service routine directly.
- Thus no polling is required because all the decisions are established by the hardware priority interrupt unit.
- The hardware priority function can be established by either a serial or a parallel connection of interrupt lines. T
- The serial connection is also known as the daisy chaining method and the parallel connection is known as Parallel Priority Encoder

Daisy-Chaining Priority:

- The daisy-chaining method of establishing priority consists of serial connection of all devices that request an interrupt.
- The device with the highest priority is placed in the first position, followed by lower-priority devices up to the device with the lowest priority, which is placed last in the chain and closest to the CPU.
- This method of connection between three devices and the CPU is shown in Figure. 11-12.

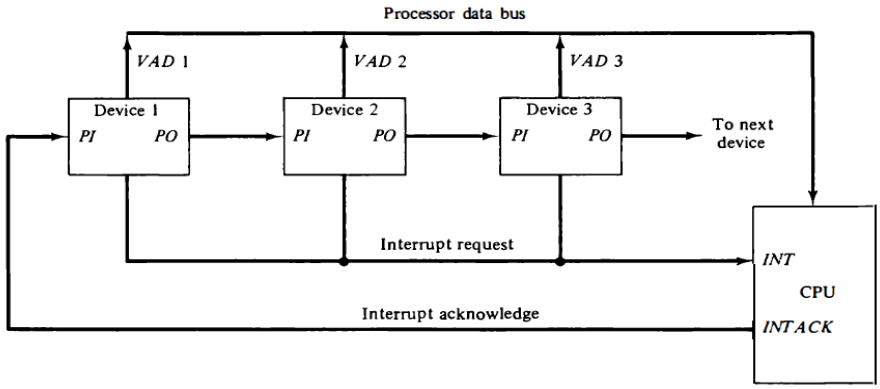


Figure 11-12: Daisy-chain priority Interrupt

- If any device want to interrupt the CPU,It generates the interrupt the corresponding line is enabled.
- The CPU responds to an interrupt request by enabling the interrupt acknowledge line.
- This signal is received by device 1 at its PI (Priority in) input.
- The acknowledge signal passes on to the next device through the PO (priority out) output only if device 1 is not requesting an interrupt.
- If 1 has a pending interrupt, it blocks the acknowledge signal to the next device by placing 0 in the PO output.

- It then proceeds to inserts its own interrupt vector address (VAD)into the data bus for the CPU to use during the interrupt cycle.
- A device that is requesting an interrupt and has a 1 on its PI input will intercept the acknowledge signal by placing a 0 on its PO output.
- If the device does not have pending interrupts, it transmits the acknowledge signal to the next device by placing a 1 on its PO output.
- Thus the device with PI=1 and PO=0 is one with the highest priority that is requesting an interrupt, and the device places its VAD on the data bus.
- The daisy chain arrangement gives the highest priority to the device that receives that interrupt acknowledge signal from the CPU.
- The farther the device is from the first position; the lower is its priority.

Disadvantage:

- The time taken to find the device that generated the interrupt is greater than the execution time of an interrupt service routine program.

Parallel Priority Interrupt:

- It consists of an interrupt register whose individual bits are set by external conditions and cleared by program instructions.
- The magnetic disk, being a high-speed device, is given the highest priority, printer has the next priority, followed by a character reader and a keyboard.
- The mask register has the same number of bits as the interrupt register assume all are 1's.
- Each interrupt bit and its corresponding mask bit are applied to an AND gate to produce the four inputs to a priority encoder.
- The priority logic for a system of four interrupt sources is shown in Figure. 11-14.

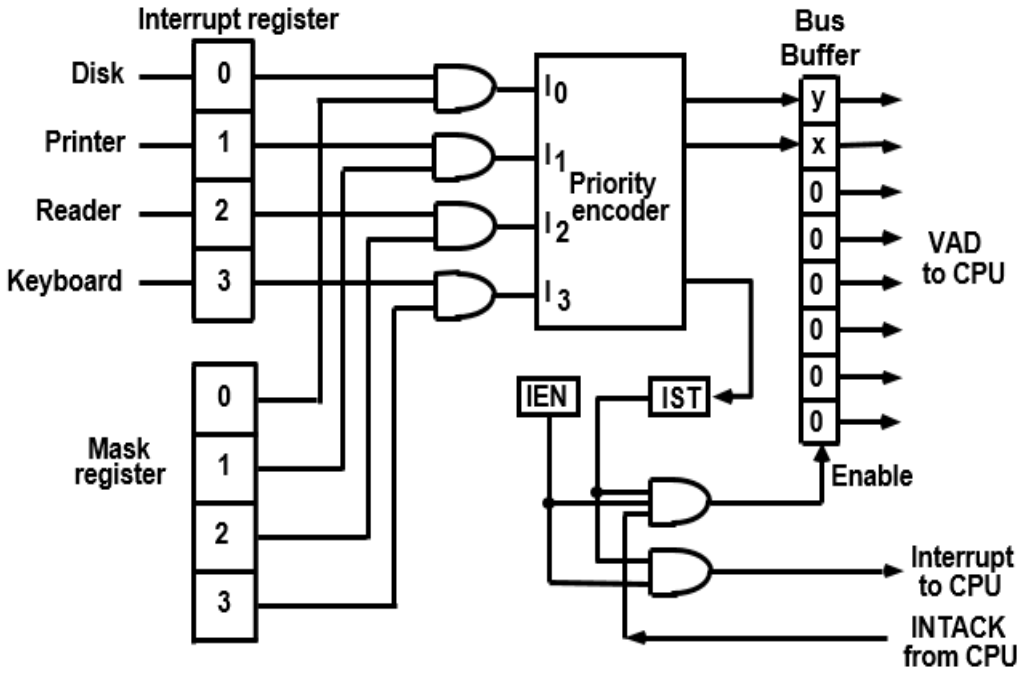


Figure 11-14: Priority logic for a system of four interrupt sources

- The priority encoder generates two bits of the vector address, which is transferred to the CPU.
- Another output from the encoder sets an interrupt status flip-flop IST when an interrupt that is not masked occurs.
- The interrupt enable flip-flop IEN can be set or cleared by the program to provide an overall control over the interrupt system.
- The outputs of IST AND ed with IEN provide a common interrupt signal for the CPU.
- The outputs of IST AND ed with IEN provide a common interrupt signal for the CPU.
- The interrupt acknowledge INTACK signal from the CPU enables the bus buffers in the output register and a vector address VAD is placed into the data bus.
- Priority Encoder: The priority encoder is a circuit that implements the priority function.
- The logic of the priority encoder is such that if two or more inputs arrive at the same time, the input having the highest priority will take precedence
- The truth table of a four input priority encoder is given in Table 11·2.

Inputs				Outputs			Boolean functions
I_0	I_1	I_2	I_3	x	y	IST	
1	×	×	×	0	0	1	
0	1	×	×	0	1	1	$x = I_0' I_1'$
0	0	1	×	1	0	1	$y = I_0' I_1 + I_0' I_2'$
0	0	0	1	1	1	1	$(IST) = I_0 + I_1 + I_2 + I_3$
0	0	0	0	×	×	0	

Table 11-2: Truth table of priority encoder

- The x's in the table designate don't-care conditions. Input I0 has the highest priority; so regardless of the values of other inputs, when this input is 1, the output generates an output xy : 00.
- I2 has the next priority level. The output is 01if I1 =1 provided that I0 = 0, regardless of the values of the other two lower priority inputs.
- The output for I2 is generated only if higher-priority inputs are 0, and so on down the priority level.
- The interrupt status IST is set only when one or more inputs are equal to 1.
- If all inputs are 0, IST is cleared to 0 and the other outputs of the encoder are not used, so they are marked with don't-care conditions.
- This is because the vector address is not transferred to the CPU when IST = 0

COMPUTER ORGANIZATION AND ARCHITECTURE
UNIT –IV
INPUT-OUTPUT ORGANIZATION
TOPIC- DIRECT MEMORY ACCESS(DMA)

Direct Memory Access

- DMA (Direct Memory Access) is a computer system feature that enables direct data transfer between memory and peripheral devices without the intervention of the CPU.
- In the Direct Memory Access (DMA) the interface transfer the data into and out of the memory unit through the memory bus.
- The transfer of data between a fast storage device such as magnetic disk and memory is often limited by the speed of the CPU.
- Removing the CPU from the path and letting the peripheral device manage the memory buses directly would improve the speed of transfer.
- This transfer technique is called Direct Memory Access (DMA).
- During the DMA transfer, the CPU is idle and has no control of the memory buses.
- A DMA Controller takes over the buses to manage the transfer directly between the I/O device and memory.
- The CPU may be placed in an idle state in a variety of ways.
- One common method used is to disable the buses through special control signals such as:
- Bus Request (BR)
- Bus Grant (BG)

- These two control signals in the CPU that facilitates the DMA transfer.
- The *Bus Request(BR)* input is used by the *DMA controller* to request the CPU.
- When this input is active, the CPU terminates the execution of the current instruction and places the address bus, data bus and read write lines into a *high impedance state.*
- High Impedance state means that the output is disconnected.
- The CPU activates the *Bus Grant (BG)* output to inform the DMA that it can now take control of the buses to perform memory transfer without processor.
- When the DMA completes the transfer, it disables the *Bus Request (BR)* line.
- The CPU disables the *Bus Grant (BG)*, takes control of the buses and return to its normal operation.

DMA Controller

- The DMA controller needs the circuits of an interface to communicate with the CPU and I/O device.
- The DMA controller has three registers:

1. Address Register- Address Register contains an address to specify the desired location in memory
2. Word Count Register- holds the number of words to be transferred which is incremented /decremented by one after each word transfer and internally tested for zero.
3. Control Register - Control Register specifies the mode of transfer

- For each word that is transferred, the DMA increments the address register and decrements the word count register.

Working of DMA Controller

- DMA controller has to share the bus with the processor to make the data transfer.
- The device that holds the bus at a given time is called bus master.
- When a transfer from I/O device to the memory or vice verse has to be made, the processor stops the execution of the current program, increments the program counter, moves data over stack then sends a DMA select signal to DMA controller over the address bus.
- If the DMA controller is free, it requests the control of bus from the processor by raising the bus request signal.
- Processor grants the bus to the controller by raising the bus grant signal, now DMA controller is the bus master.
- The processor initiates the DMA controller by sending the memory addresses, number of blocks of data to be transferred and direction of data transfer.
- After assigning the data transfer task to the DMA controller, instead of waiting ideally till completion of data transfer, the processor resumes the execution of the program after retrieving instructions from the stack.
- DMA controller now has the full control of buses and can interact directly with memory and I/O devices independent of CPU.
- It makes the data transfer according to the control instructions received by the processor.
- After completion of data transfer, it disables the bus request signal and CPU disables the bus grant signal thereby moving control of buses to the CPU.
- When an I/O device wants to initiate the transfer then it sends a DMA request signal to the DMA controller, for which the controller acknowledges if it is free.
- Then the controller requests the processor for the bus, raising the bus request signal.
- After receiving the bus grant signal it transfers the data from the device.

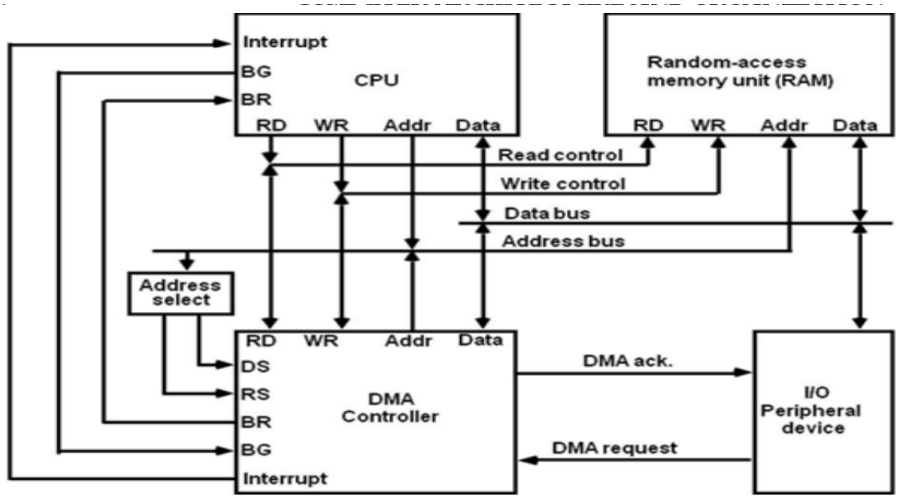


Fig: DMA transfer in a Computer System

DMA Transfer

- The data transfer in DMA can be made in :
- ❑ Burst transfer mode.
- ❑ Cycle Stealing mode.
- Burst transfer:- In this mode, a block sequence consisting of a number of memory words is transferred in continuous burst while the DMA controller is master of the memory buses.
- Cycle Stealing :- This mode allows the DMA controller to transfer one data word at a time, after which it must returns control of the buses to the CPU.