UNIT - IV
MEMORY AND I/O ORGANIZATION

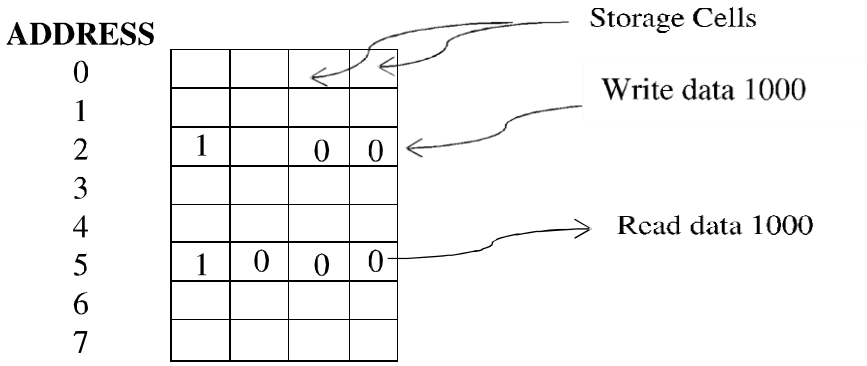
4.1. Memory Hierarchy:-

* A **Memory Unit** is considered as a collection of cells, in which cells is capable of storing a bit of information. It stores information in group of bits called **byte** or **word**.
* Memories are made up of registers in the memory is one storage location also called **memory location.**
* Each memory location is identified by an **address**.
* The total number of bits that a memory can store is its **capacity.**

1 byte = 8 bits

* Each register consist of storage element, which stores one bit of data, is called a cell.

Storage Cells



* The data stored in a memory by a process called writing a data.
* The data stored in a memory by a process called reading a data.
* Key operations
    - Read
        - Data is retrieved from the memory
        - Also called as Fetch operation
    - Write
        - Data is stored into the memory
        - Also called as Store operation

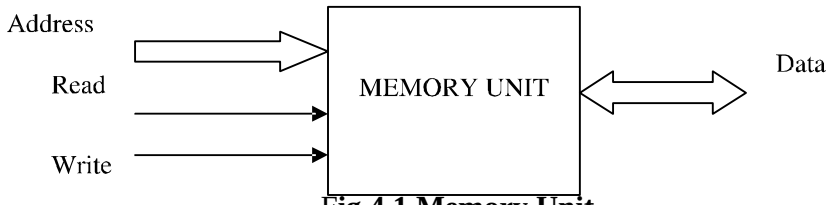


Fig 4.1 Memory Unit

A Memory unit stores binary information in groups of bits called **word**.

* To transfer the data between memory and processor two registers are used:
  1. Memory Address Register(MAR).
  2. Memory Buffer Register(MBR).
* **Memory Address Register(MAR):** It holds the address of the data to be transferred.
* **Memory Buffer Register(MBR):** It contains the data to be transferred to/from the main memory.

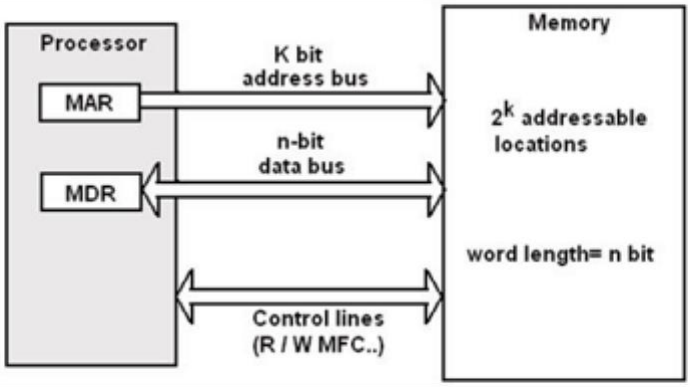


Fig 4.2 Data transfer between memory and processor

4.2. MEMORY HIERACHY

* A structure that uses multiple levels of memories is called **memory hierarchy.**
* A memory hierarchy consists of multiple levels of memory with different **SPEED AND SIZES.**
* The **faster** memories are more expensive per bit than the slow memories and thus are smaller.
* As the distance from the processor increases, the size of the memories and the access time both increases.

I. PROCESSOR REGISTER

The fastest access of data is possible only if data is **available in processor registers.** The top of the speed of access is very small portion of the required memory.

II. PRIMARY CACHE OR LEVEL 1 CACHE

A small amount of memory can be implemented directly on the processor chip. It **holds the copies of instruction** and data that are needed **currently for execution.**

III. SECONDARY CACHE

A secondary cache is placed between the **primary cache(Level 1 cache)** and rest of the memory. It is referred to as **Level 2 cache(L2).** It is larger than the primary cache. It is usually implemented using SRAM chips.

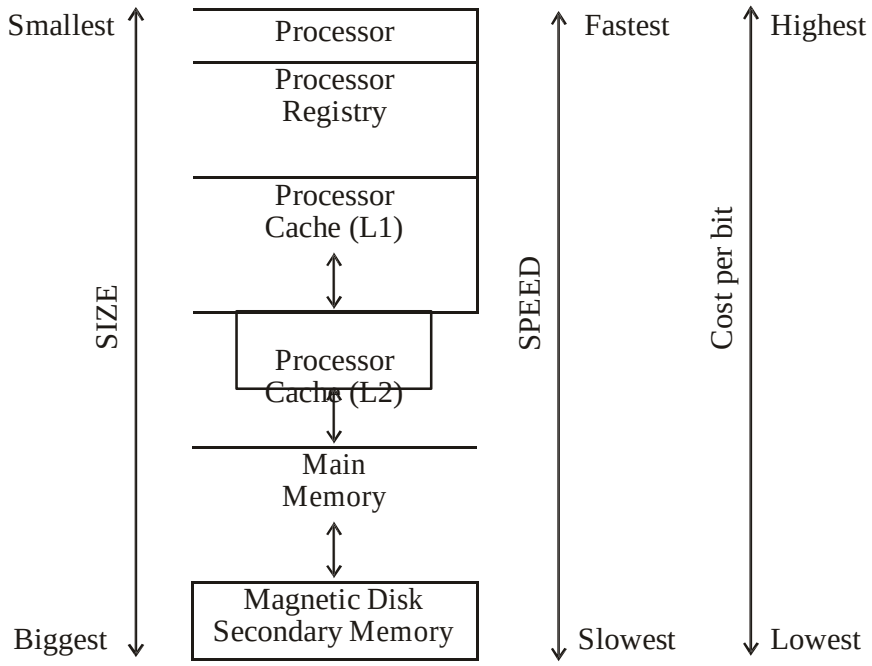


Fig 4.3 Secondary Cache Memory

IV. MAIN MEMORY

This is large memory and is implemented using dynamic memory components like

- Single in-line memory module(SIMM)
- Dual in-line memory module(DIMM)
- Rambus in-line memory module(RIMM)

V. SECONDARY MEMORY

Magnetic disk storage units are called as secondary memory. This provide **huge amount of low cost** storage. They are very slow compared to the semi-conductor devices which are used to implement primary memory.

Tabel 4.1 Characteristics of Memory

Memory	Speed	Size	Cost
Primary cache	High	Lower	Low
Secondary cache	Low	Low	Low
Main memory	Lower than Secondary cache	High	High
Secondary Memory	Very low	Very High	Very High

4.3 MEMORY TECHNOLOGIES

There are four primary technologies used. They are

- Static Random Access Memory (SRAM).
- Dynamic Random Access Memory (DRAM).
- ROM and Flash Memory.
- Magnetic Disk.

4.3.1. SRAM

* It is very part of the RAM. It is main memory and located very close to the processor.
* SRAM's are simply **integrated circuits** that are memory array with a single access port that can provide either a **read or write.**
* SRAM have a **fixed access time to any data,** but the read and write access time may differ.
* SRAM **don't need to refresh** and so the access time is very close to the cycle time.
* SRAM use **6 to 8 transistors** per bit.
* SRAM needs only **minimal power** to retain the charge in standby mode.

4.3.2. DRAM

* In a SRAM, as long as power is applied the value can be kept indefinitely.
* In DRAM, the value is kept in a cell and is stored as a charge in a capacitor,
* DRAM has **single transistor** used to access this stored charge, either to **read the value** or to overcome the charge stored here.

- Asynchronous DRAM.
- Synchronous DRAM.
- Rambus Memory.

4.3.3. ASYNCHRONOUS DRAM

- DRAM's stores all the charge on a capacitor so cannot be kept indefinitely and must periodically be refreshed. That is why this memory is called dynamic.
- In DRAM to refresh the cell, we **read its contents** and **write it back.** The charge can be kept for several milliseconds.

4.3.4 SYNCHRONOUS DRAM

- Improves the access time significantly, to improve the interface to **processor's added clocks** with its and it is called as synchronous DRAM or SDRAM.
- The advantage of SDRAM is that the use of clock eliminates the time for the memory and processor to synchronize.
- SDRAM's transfer the bits in the burst. The fastest version is called **Double Data Rate(DDR).**
- **DDR-** It means data transfers on both the **rising and failing** edge of the clock, thereby getting twice as much bandwidth as we expect based on the clock rate and data width.
- The latest version of this technology is called **DDR4.**

Table 4.2 Difference between SRAM and DRAM

SRAM	DRAM
Information is stored in 1 bit cell called Flip Flop	Information is represented as charge across a capacitor
Information will be stored as long as the power is ON	Information may be lost, if power loss
No refreshing is needed	Refreshing is needed
Less packaging density	High packaging density
More complex hardware	Less complex hardware
More expensive	Less expensive

4.3.5 RAMBUS MEMORY

- Rambus Dynamic Random Access memory(RDRAM) in short Rambus memory is the fastest type of computer memory available. Typical SDRAM can be transfer data at speed upto 133 MHZ, but standard RDRAM can transfer data over 1 GHZ.
- RDRAM is used for **video memory on graphics** accelerator cards, for **cache memory** and for **system only** in **high performance workstations and servers.**
- An improvement to RDRAM called Direct Rambus(DRDRAM)allows for even faster data transfer rates.

4.3.6 ROM and FLASH MEMORY

* The read operation is same but write operation is different in non-volatile memory.
* A special writing process is needed to store information into this memory.
* There are different variations in non-volatile memory. They are
  a) Read only memory (ROM).
  b) Programmable ROM (PROM).
  c) ErasablePROM (EPROM).
  d) Electrically EPROM (EEPROM).
  e) Flash memory.

4.3.6.1 ROM (READ ONLY MEMORY)

- In ROM, permanent data and programs are stored. It is permanently in-built in the computer at the time of its production.
- Since, it is **non-volatile memory** it holds data even if the power is turned off.

4.3.6.2 PROM (PROGRAMMABLE READ ONLY MEMORY)

- Permanent data and programs are stored in ROM. But ROM designs allow the user **to load programs and data.** Such ROM designs are called Programmable ROM (PROM).
- PROM is once programmable and it is more flexiable and convenient than ROM.
- PROM are faster and less expensive.

4.3.6.3 EPROM (ERASABLE PROGRAMMABLE READ ONLY MEMORY)

- ROM designs allow the stored data to be erased and new data to be loaded. Such an Erasable Re-Programmable ROM (EPROM).

- **ADVANTAGE**
  - Its content can be erased and re-programmed.
- **DISADVANTAGE**
  - When erasing, entire EPROM chip content is erased.
  - If chip must be physically removed from the chip for re-programming, the entire content is erased.

4.3.6.4 EEPROM (ELECTRICALLY ERASABLE PROGRAMMABLE READ ONLY MEMORY)

- It can be both programmed and erased electrically such chips are called as Electrically EPROM.
- They need not to be removed for erasing.
- It is possible to erase the chip contents selectively.

- **DISADVANTAGE**
  - Different voltages are needed for erasing, reading and writing the data.

4.3.6.5 FLASH MEMORY

- Flash memory is a type of EEPROM. But still there are some major difference between EEPROM and flash memory.
- **In EEPROM,** it is possible to read and write the contents of a single cell.
- **In FLASH MEMORY,** it is possible to read the contents of a single cell, but during writing operations the entire block of cells must be written. The previous contents of the clock of cells must be erased before writing.

- **ADVANTAGE**
  - Flash memory devices have greater density.
  - Higher capacity and lower cost per bit.

* Consumes less power in their operation.
* Flash memory usages in various devices.
* There are two popular high capacity flash memory devices are
  1) Flash cards.
  2) Flash drivers.

4.3.6.6 **DISK MEMORY**

- A magnetic hard disk consists of a collection of platters, which rotate on a spindle at 5400 to 15,000 revolutions per minute.

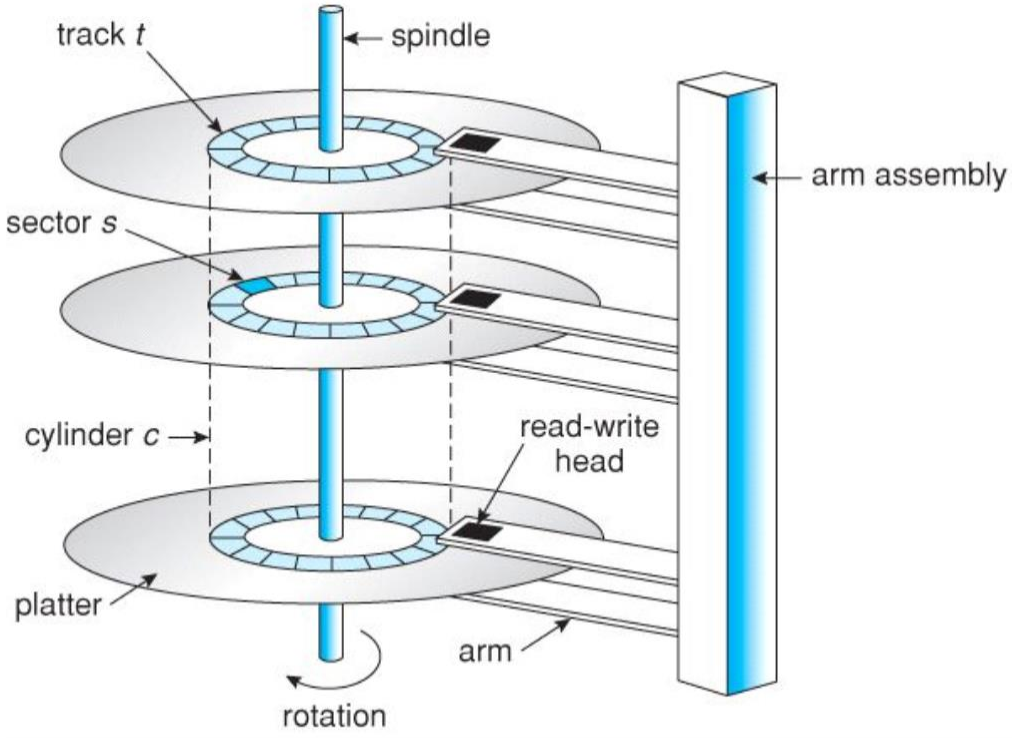


Fig 4.4 DISK MEMORY

- The metal platters are covered with magnetic recording material on both sides.
- Arm - containing small electromagnetic coil called a **read-write head.**
- Each disk surface is divided into concentric circles called **Tracks.**
- Each tracks is in turn divided into **sectors** that contain the information

1 track = 1000 sectors.

- Sectors are typically 512 to 4096 bytes in size.
- **Cylinder** is used to refer to all the tracks under the heads of given point of all the surfaces.

Performance Measures of Disks

The main measures of the qualities of a disk are

1. Capacity
2. Access Time
3. Data Transfer Rate
4. Reliability

Capacity

- The storage capacity of a single disk ranges from 10MB to 10GB

Access time

- Time from when a read or write request is issued when data transfer begins.
- The time for repositioning the arm is called **seek time**, and it increases with the distance the arm must move.
- Average seek time is the average of the seek time;
- It is measured over a sequence of (uniformly distributed) random requests, and it is about one third of the worst-case seek time.

Data transfer rate

- Rate at which data can be retrieved from or stored to the disk.

Reliability

- Measured by the mean time to failure
- The typical mean time to failure of disks ranges from 30,000 to 800,000 hours

4.4. CACHE MEMORY

* The speed of main memory is very low when compared to the speed of modern processor. The processor cannot spend much of its time in waiting to access instructions and data from main memory.
* Hence, a scheme that reduces the time needed to access the information from main memory is required. An efficient is to use a fast **cache memory.**

The program and data which are currently being executed is accessed from Processor and it is added between the processor and main memory to speed up the execution process is called Cache memory.

- Cache is a small amount of memory used to speed up the system performance
- Placed between the processor and main memory
- Used to store programs and data currently being executed and temporary data frequently used by the CPU
- To speed up the execution process, high speed memory such as cache (SRAM) is used. It is a type of memory that contains small portion of code and data
- To execute a program in a computer system
  - Program is loaded into the main memory
  - Processor fetches the instructions (code) and data from the main memory
  - Execute the program.

4.4.1 Cache Memory System

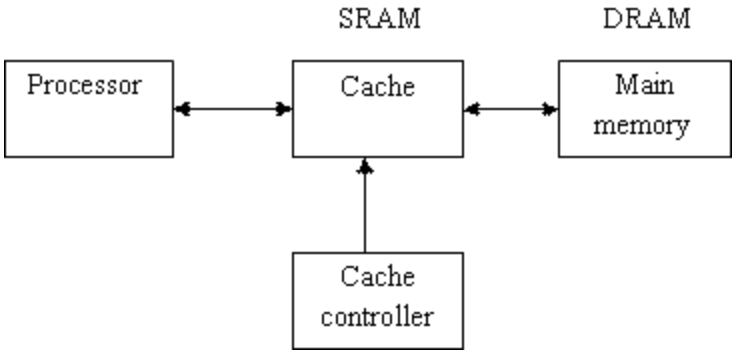


Fig 4.5 Cache Memory System

- Cache memory includes small amount of fast memory (SRAM) and large amount of slow memory (DRAM).
- If the processor requests the data in cache, which is not available in cache it is referred as **Cache Miss,** then the desired block is copied from the main memory to cache using cache controller.
- Cache controller decides which memory block should be moved in or out of cache and main memory.

LOCALITY OF REFERENCE

* The **effectiveness of the cachemechanism**is based on the **property of computer programs** called Locality of reference.

* In a program, instructions in localized areas are executed repeatedly whereas the reminder of the program is accessed less frequently. The concept is referred to as **locality of reference.**
* There are two forms of locality of reference
    a) Temporal
    b) Spatial

a) The **Temporal locality of reference** means that a recently executed instruction is likely to be executed again very soon. (i.e) data is 1st needed, it may be needed again very soon.

a. Example
    i. Loops
    ii. Reuse

In temporal, whenever the data or instruction is needed, it should be brought into the cache and it remains in main memory until it is needed again.

b) The **Spatial locality of reference** means that instructions that are close to each other are to be executed again very soon. (i.e) instead of fetching just one item from the main memory to cache. It is useful to fetch a block of data that are stored in adjacent address.

a. Example
    i. Straight-line code
    ii. Array access

c) In Spatial, instead of bringing one data or instruction from the main memory to cache, it brings a block of data and instruction.

Note: Block refers to set of contiguous address of some size.

4.4.2 CACHE BASICS

* When a read request is received from a processor, a block of memory words containing the location are transformed to cache.
* Cache memory can store reasonable number of blocks, but this number is small compared to the total number of blocks in the main memory.
* The correspondence between the main memory blocks those in the cache are specified by mapping function.

Main Memory

Cache

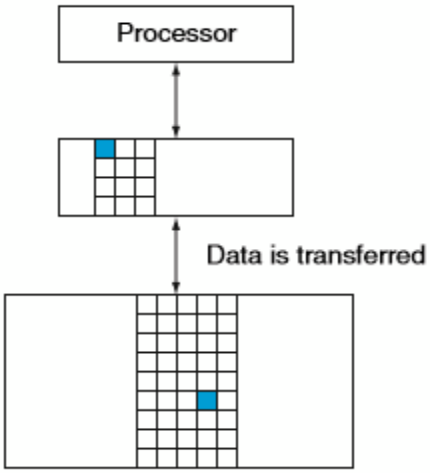


Fig 4.6. Main Memory Cache

* Cache was the name chosen to represent the level of memory hierarchy between the processor and main memory.
* Cache is a **safe place for hiding or storing things** that we need to examine.
* When the CPU finds a requested data item in the cache it is called as **cache hit.**
* When the CPU does not find a requested data item in the cache it is called as **cache miss.**
* A fixed size collection of data containing the requested word called a **block.**

4.4.3 Types of Cache Memory

1. Primary Cache
   - Also referred as Processor Cache, Level1 or L1 cache
   - Always located on the processor chip
2. Secondary Cache
   - Also referred as Level2 or L2 cache
   - Placed between the primary cache and the main memory

Advantages

- Cache memory is faster than main memory
- Consumes less access time as compared to main memory
- Stores program that can be executed within a short period of time
- Stores data for temporary use

Disadvantages

- Limited capacity
- Very expensive

4.4.4 BLOCK REPLACEMENT

* When the cache is full and a memory word is not in the cache is referenced, the cache control hardware replaces a block from the cache to store the new block.
* The algorithm used for this process is called **Replacement algorithm.**
* The fraction of memory accesses found in a level of the memory hierarchy is called as **hit rate** or **hit ratio.**

WRITE OPERATION

* There are two ways for write operation
  - Write-through.
  - Write-back.
  - Write-buffer.
* **WRITE-THROUGH**
  - The cache location and the main memory location are updated simultaneously.
* **WRITE-BACK**
  - Only the cache location is updated a flag bit is used to specify the updation. It is called **dirty bit** or **modified bit.**
* **WRITE-BUFFER**
  - Stores the data while it is waiting to be written into the memory.
  - After writing data, processor can continue execution.
  - Write completes - write buffer is free.

Table 4.3 Both write back and write through have their advantages

WRITE-BACK	WRITE-THROUGH
1) Individually words can be written by the processor at the rate that the cache rather than the memory can accept them.	1) Misses are simpler and cheaper because they never require a block to be written back to the lower level.
2) Multiple words within a block require only one write to the lower level in the hierarchy.	**2) Write-through is easier** to implement than **write-back.**

3) When blocks are written back, the can make effective use of high bandwidth transfer since entire block is written.	

READ MISS

* When the addressed word in a read operation is **not in the cache,** it is called as **Read miss.**
* Alternatively, this word may be sent to the processor as soon as it is read from the memory. This approach is called **Load-through** or **early restart.**

WRITE MISS

* During write operation, word is not in the cache, it is called as **write miss.**

MISS RATE = 1 - HIT RATE

MISS RATE

* The fraction of memory accesses not found in the level of the memory hierarchy is called **miss rate.**

MISS PENALTY

* It is the **time to replace a block** in the upper level with the corresponding block from the lower level, plus the **time to deliver** this block to processor.

AMAT (AVERAGE MEMORY ACCESS TIME)

* It is the average time to access memory consider both **hits and misses** and the frequent of different accesses.

AMAT = TIME FOR A BIT + MISS RATE * MISS PENALTY

4.5 MEASURING AND IMPROVING CACHE PERFORMANCE

Two different techniques used to improve cache performance

- Reducing the miss rate by reducing the probability that two different memory blocks will contend for the same cache location
- Reducing the miss penalty by adding an additional level to the hierarchy called multilevel caching

4.5.1 CPU execution time

- CPU execution time incorporated with cache performance. The performance is then the product of the clock cycle time and the sum of the CPU cycles and the memory stall cycles
- **Formula**

  CPU execution time = (CPU clock cycles + Memory stall clock cycles) * Clock cycle time

Memory stall clock cycles

- The number of cycles during which the CPU is stalled waiting for a memory access is called memory stall cycles
- Memory stall clock cycles is defined as the sum of read-stall cycle and write-stall cycle
- **Formula**

  Memory stall clock cycles= Read – stall cycles + Write – stall cycles
- Read-stall cycles
  - Defined in terms of the number of read accesses per program, the miss penalty in clock cycles for a read, and the read miss rate

o Formula

$$Read-stall\,cycles=\frac{Reads}{Program}\times Read\ miss\ rate\times Read\ miss\ \text{penalty}$$

- Write -stall cycles
  - Defined in terms of sum of number of write accesses per program, the miss penalty in clock cycles for a read, the read miss rate and Write buffer stalls
  - Formula

$$Write-stall\ cycles=\left(\frac{Writes}{Program}\times Write\ miss\ rate\times Write\ miss\ penalty\right)+Write\ buffer\ stalls$$

- In write-through cache organizations, the read and write miss penalties are the same (i.e.,) the time to fetch the block from memory.
- By combining the reads and writes by using a single miss rate and the miss penalty:

$$Memory-stall\ clock\ cycles=\frac{Memory\ accesses}{Program}\times Miss\ rate\ \times Miss\ penalty$$

- Factoring the above equation

$$Memory-stall\ clock\ cycles = \frac{Instructions}{Program} \times \frac{Misses}{Instruction} \times Miss\ penalty$$

4.5.2 Reducing Cache Miss using mapping function

- The Cache memory stores a reasonable number of blocks at a given time but the size of cache is small compared to the total number of blocks available in the Main Memory.
- The correspondence between the main memory blocks and cache is specified by mapping function.
- It defines how memory blocks are placed in cache.
- The mapping functions are used.

4.5.2.1 MAPPING FUNCTION

* Mapping functions are used to specify where the memory blocks are placed in cache.
* There are three types of mapping functions
  A. Direct Mapping.
  B. Associative Mapping.
  C. Set-Associative Mapping.

4.5.2.1 A. DIRECT MAPPING

* It is the simplest method to determine cache location to store memory blocks. A cache structure in which each memory location is mapped to **exactly one location** in the cache is called direct-mapping cache.
* The position of a memory block is

(BLOCK ADDRESS) MODULO (NUMBER OF BLOCKS IN THE CACHE)

* Direct mapping cache line table.

CACHE LINE	MAIN MEMORY BLOCKS HELD
0	**0,m,2m,3m, …. ,2s-m**
1	**1,m+1,2m+1, …, 2s-m+1**
2	**2,m+2,2m+2, …., 2s-m+2**
.	**.**
.	**.**
m-1	**m-1,2m-1,3m-1, … , 2s-1**

* To implement direct mapping the 16-bit memory address is divided into 3-fields.

TAG(5)	BLOCK(7)	WORD(4)
MSB	16-bits	LSB

- Example
  - Bblock" **j**" of the main memory maps onto block "**i modulo 128**" of the cache.
  - Main memory blocks 0, 128, 256,…… is loaded into the cache and it is stored in cache block 0
  - Blocks 1, 129, 257, …..are loaded into the cache and it is stored in cache block 1.

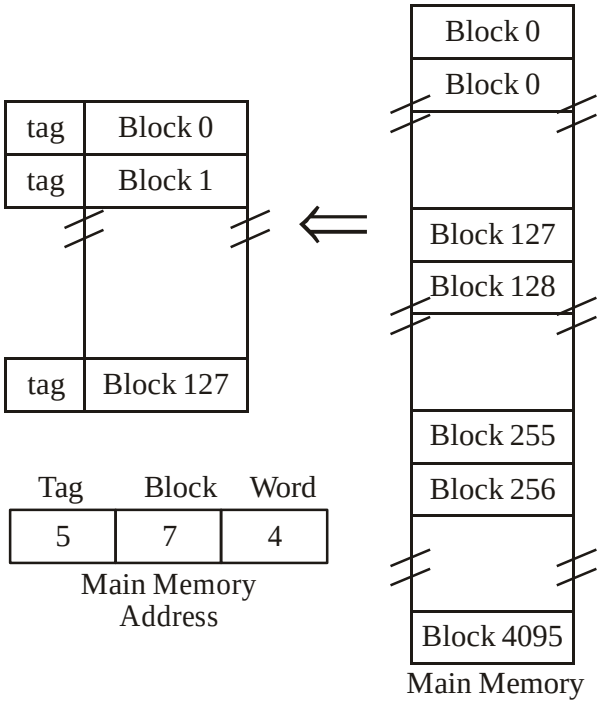
j = i modulo m

where i= Main memory block number

j= Cache block number

m=Number of blocks in cache

Fig 4.7 Direct mapping Memory



TAG

* A field in a table for a memory hierarchy that contains the address information required to identify whether block in the order mapping to a requested word.

ADVANTAGE

* It is simple and easy to implement.

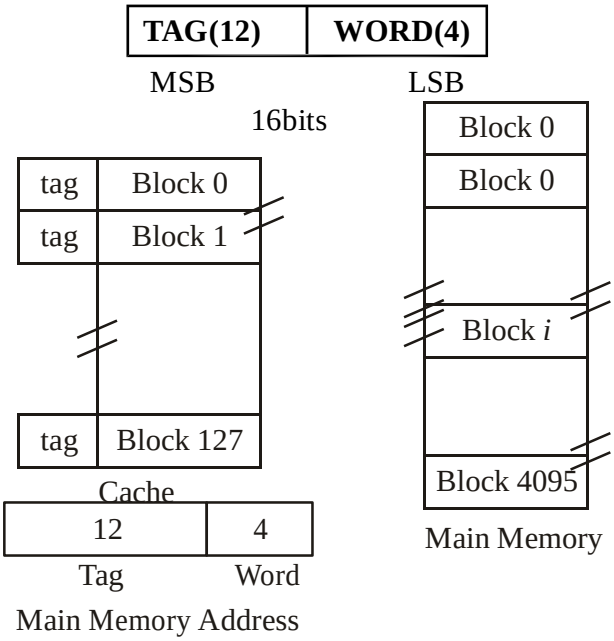
DISADVANTAGE

* It is not flexible.
* Contention may arrive for the position, if cache is not full.

4.5.2.1 B. ASSOCIATIVE MAPPING

* A cache structure in which a block can be placed in any location in the cache is called **fully associative cache.**
* It is much more flexible than direct mapping.
* Main memory block **can be placed into any cache block position.**

Fig 4.8 Associate mapping Memory



* 12-tag bits are required to identify a memory block.
* 4-word bits are used to select one of the 16-bits in a block.

ADVANTAGE

* It gives complete freedom in choosing a cache location to load a memory block in a cache.
* Space in the cache can be used efficiently.
* A new block replaces an existing block only if the cache is full.

DISADVANTAGE

* Cost of associative mapping cache is higher than the cost of direct mapping cache.
* All the 128 tags are searched to find whether a given block is in the cache or not.

4.5.2.1 C. SET-ASSOCIATIVE MAPPING

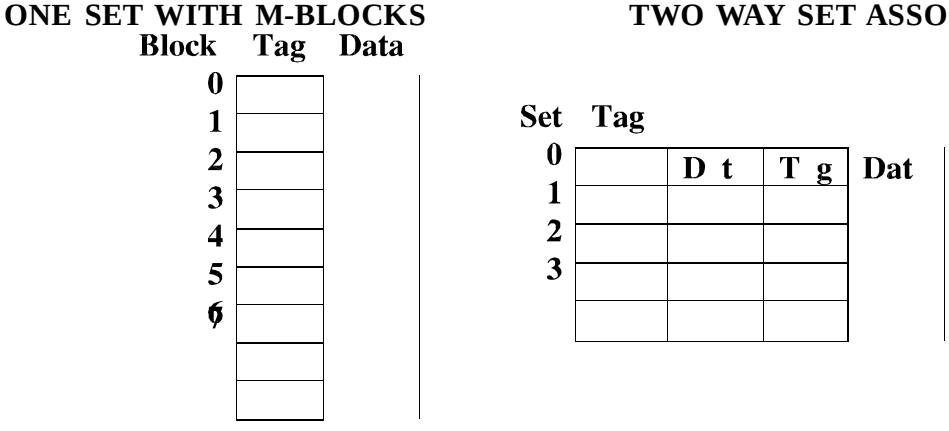
* A cache that has fixed number of location(at least two) where each block can be placed is called set-associative mapping.
* A combination of direct and associative mapping techniques are called set-associative mapping.
* Blocks of cache are grouped into sets. In set-associative cache, the set containing a memory blocks is given by,

(BLOCK NUMBER) MODULO (NUMBER OF SETS IN THE CACHE)

A direct-mapped cache is simply a one-way set-associative cache. Each cache entry holds one block each set has one element.

* A **fully associative cache** with m-entries is simply an m-way set-associative cache. It has one set with m-blocks.

Fig 4.9 SET-ASSOCIATIVE mapping



FOUR WAY SET ASSOCIATIVE

Set	T g	Data	Tag	Data	Tag	D t	T	
0								
1								

g D t

* To implement set-associative mapping, the 16 bit memory address is divided into three field.

TAG(6)	SET(6)	WORD(4)

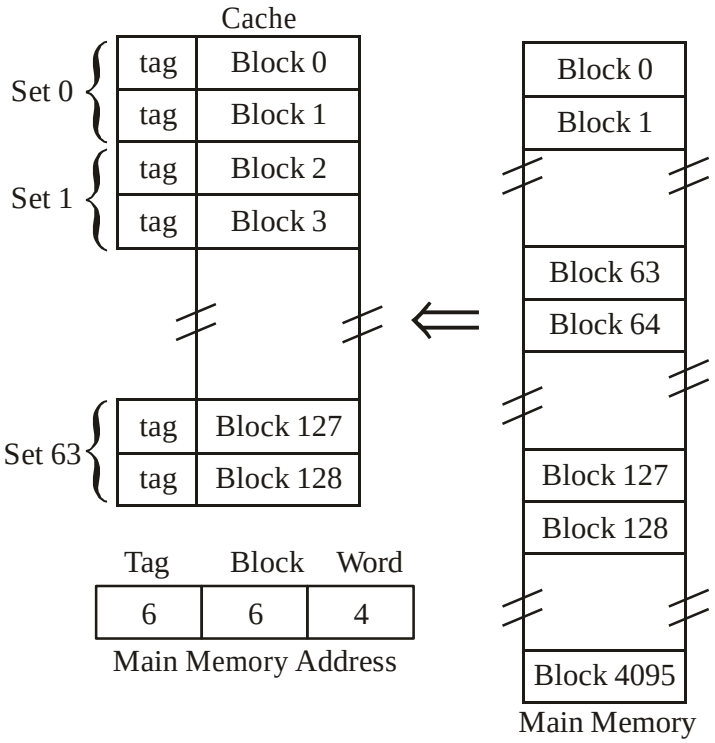


Fig 4.10 FOUR WAY SET ASSOCIATIVE

ADVANTAGE

* Contention method of direct mapping technique is solved by having a few choices for block replacement.
* Hardware cost is reduced by decreasing the size of associative search.
* It is simple to implement.

4.5.3 REPLACEMENT ALGORITHM

* When a new block is brought into the cache, one of the existing blocks must be replaced by a new block.
* In a **direct-mapped cache,** the position of each block is predetermined, hence there is **no need of replacement algorithm.**

* In **associative and set-associative caches,** there is a choice of replace existing block. Cache Controller must decide which block to overwrite.
    * There are 4-most common replacement algorithms
        - Least Recently used (LRU).
        - First in First out (FIFO).
        - Least Frequently used (LFU).
        - Random.

➢ **LRU -** The block replaced is the one that has been unused for longest time.
➢ **RANDOM -** Blocks are randomly selected, possibly using some

4.6 VIRTUAL MEMORY

* In modern computers, the operating system moves programs and data automatically between the main memory and secondary storage.
* Techniques that automatically move program and data blocks into the physical main memory when they are required for execution are called **Virtual memory** technique.
* The main memory can act as a "cache" for the secondary storage. This technique is called **virtual memory.**
* **MMU –** The memory management unit controls this virtual memory. **It translates virtual address into physical addresses.**
* A simple method for translating virtual address into physical address is to assume that all the programs data are composed of fixed length unit called **pages**.
* Transfer of data between the disk and the main memory is performed using **Direct Memory Access (DMA).**

VIRTUAL MEMORY

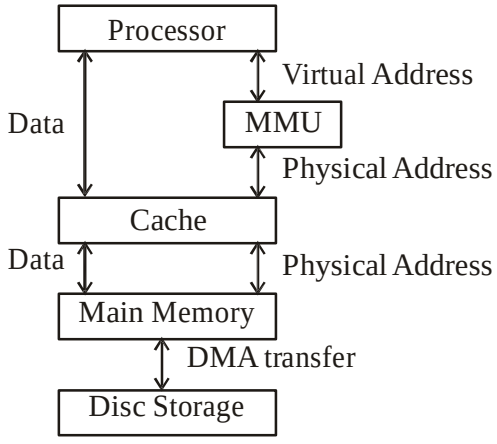


Fig 4.11 Virtual Memory

4.6.1 ADDRESS TRANSLATION

* A virtual memory address translation method based on the concept of fixed-length.
* Each virtual address generated by the processor consists of two fields:
    i. Virtual page number (Higher order bits)
    ii. Offset (Lower order bit).

i. Virtual page number (Higher order bits)

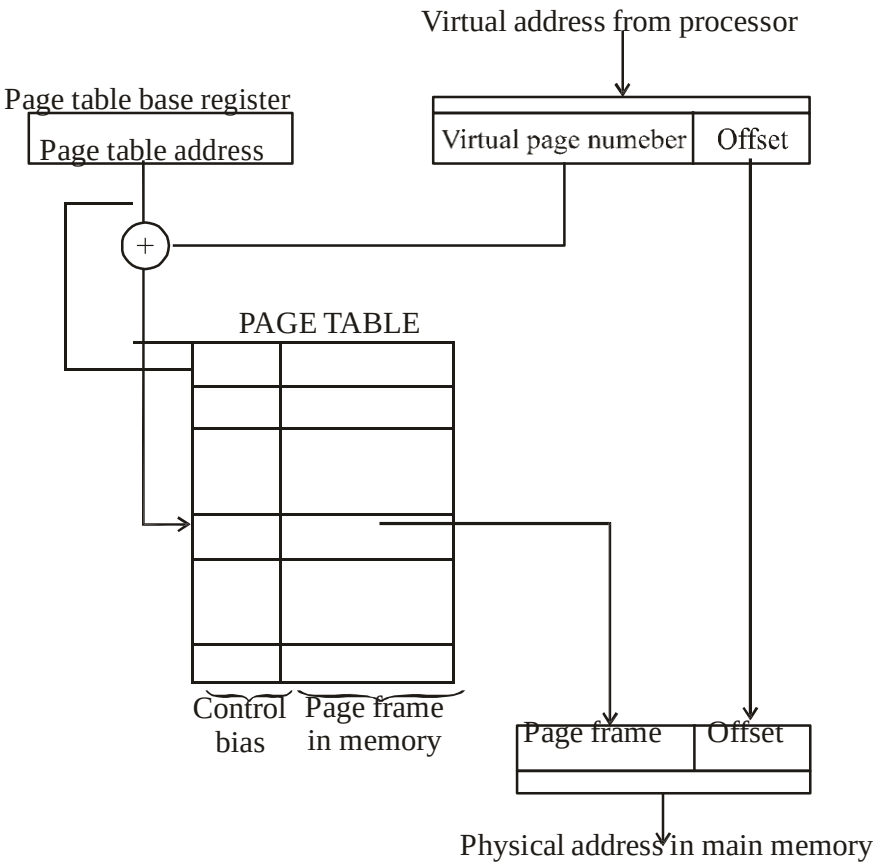


Fig 4.12 Virtual Memory Address translation

ii. Offset (Lower order bit).

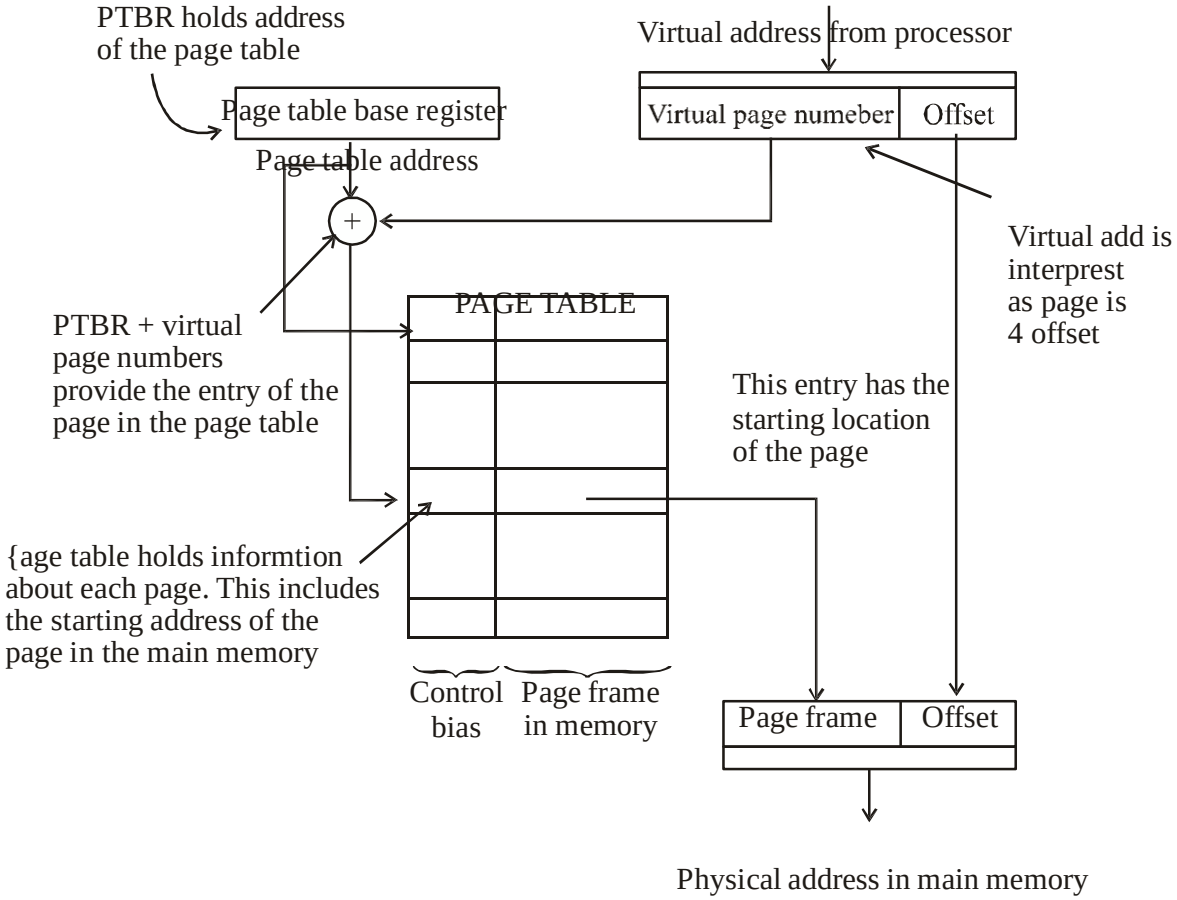


Fig 4.13. Virtual Memory Address translation

4.6.2 PAGE TABLE

Page table contains **information about the main memory location** of each page.

- This information includes the **main memory address** where the page is stored and **current status** of the page.
- An area in the main memory that holds one page is called **page frame.**
- Starting address of the page table is kept in a page table base register.
- The content of **page table base register is added with the virtual page number** to get the corresponding entry in the page table.
- Each entry in the page table also includes some control bits.
- Two important control bits are

1) **Valid bit –** indicates validity of the page.
2) **Modify bit –** indicates whether the page has been modified during residency in the memory.

The following steps are used in address translation:

- Virtual address generated and divided into two parts
  - **Virtual page number.**
  - **Offset.**
- Virtual page number in the address is added with the contents of page table base register to the entry in the page table.

4.6.3. TRANSLATION LOOKASIDE BUFFER (TLB)

* Virtual memory processor has to access page table is kept in the memory. To reduce the access time and degradation of performance, a small portion of page table is accommodated in MMU. This small cache (portion) is called as **Translation lookaside buffer (TLB).**

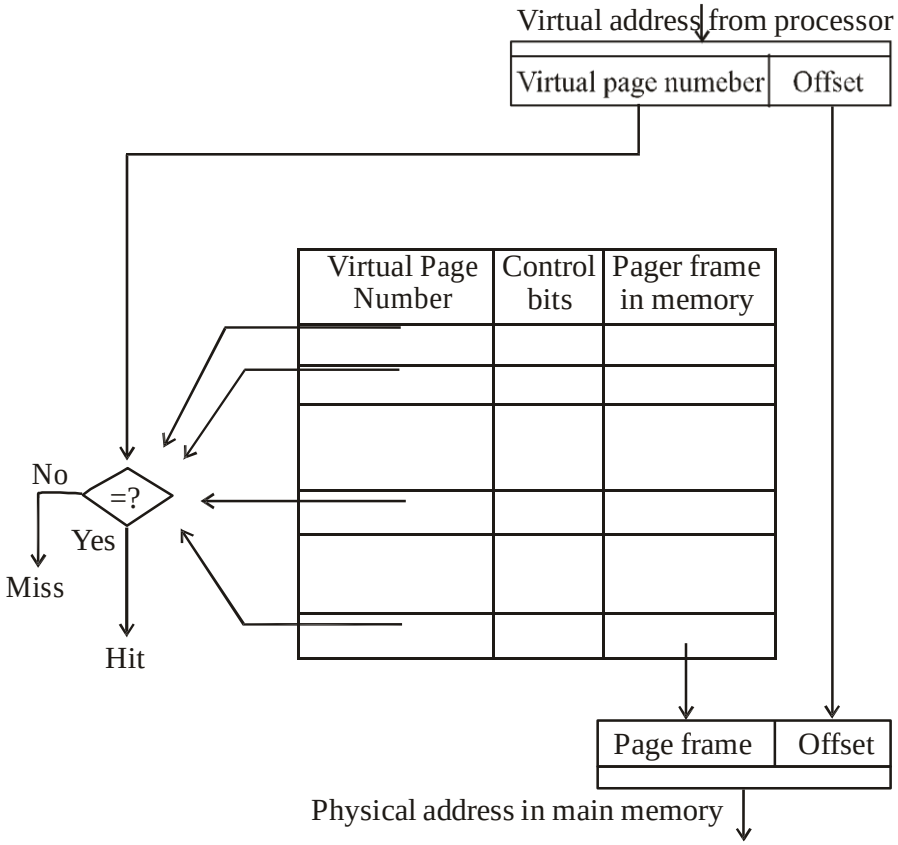


Fig 4.14. Translation lookaside buffer

* TLB contains virtual address of entry, in addition to the page table entry.

Following steps are used in address translation process with TLB:

- Processor generates the virtual addresses.
- MMU looks in the TLB for the referenced page.
- If the page table entry for this page **is found** in the TLB, the physical address is obtained immediately.
- If the page table entry is **is not found,** then the corresponding page entry is obtained from the page table in the main memory.
- Then the TLB is updated accordingly.
    - **Page fault**
    - **Page replacement.**
    - **Write operation.**

4.6.3.1.PAGE FAULT

* When a program generates an address **request to a page** that is **not available in the main memory,** then a **page fault** is occurred.
* If a valid bit for a virtual page is off, it causes a page fault. If page fault occur then the control has given to operating system.
* When the MMU detects a page fault it asks the OS to handle the situation by raising a exception (interrupt). The operating system copies the contents of the page from the disk into the main memory and transfers the control back to the interrupted task.

4.6.3.2. PAGE REPLACEMENT

* When a new page is brought from the disk if the main memory is full,a page must be replaced from the main memory.
* LRU replacement algorithms can be used for page replacement.

4.6.3.3 WRITE OPERATION

* A modified page has to be written back to the disk before it is removed from the main memory.
* Access time of disk is so long, only **write-back** protocol is suitable for virtual memories, write through protocol is not suitable.

4.7. INPUT/ OUTPUT SYSTEM:

The important component of any computer system are CPU, memory and IO devices.(Peripherals) The CPU fetches instructions from memory, processes them and stores results in memory. The other components of the computer system (I/O devices) may be loosely called the **Input/ Output system.**

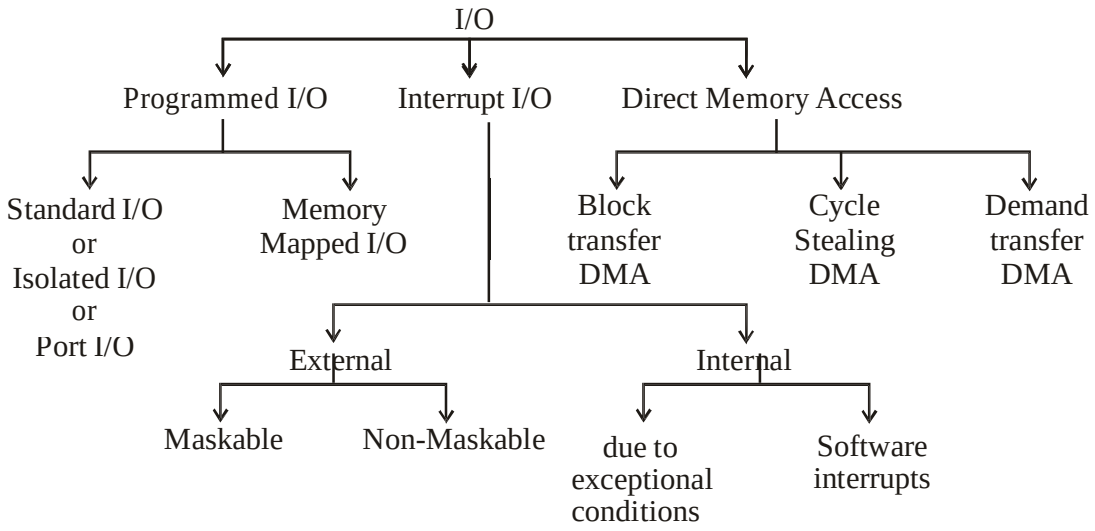


Fig 4.15 I/O System

4.7.1. PROGRAMMED IO:

- **IO** operations are distinguished by the extent to which the CPU is involved in their execution. If IO operations is said to be using **Programmed IO**
- **The IO device** does not have direct access to main memory. A data transfer from an IO device to memory requires the CPU to execute several instructions.
  - **MEMORY – MAPPED IO**
  - **IO – MAPPED IO**

4.7.1.1. MEMORY – MAPPED IO:

In PROGRAMMED IO systems, the CPU, memory and IO devices communicate via system bus. The address line of the system bus that are used to select memory locations can also be used to select IO devices. An IO device is connected to the bus via as IO port.

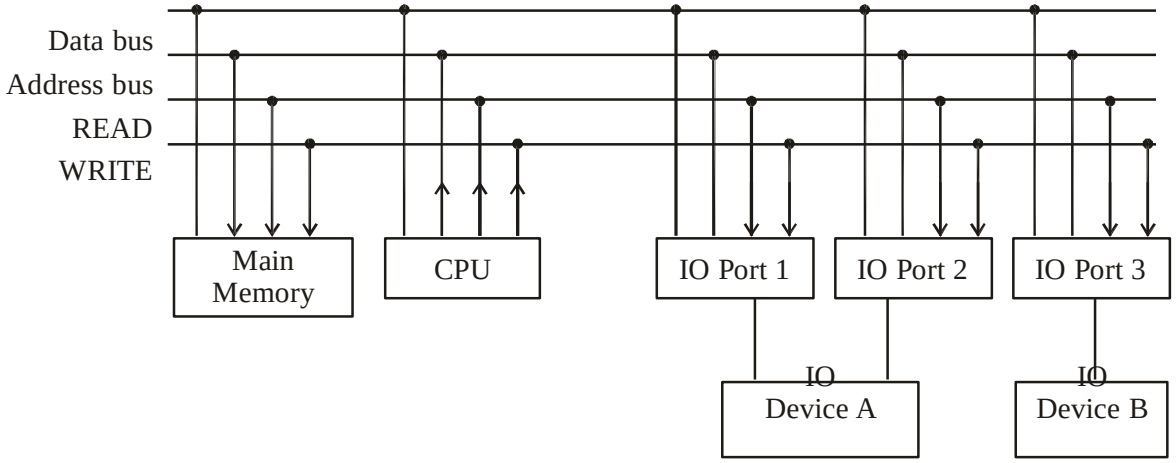


Fig 4.16 Programmed IO with Memory - Mapped IO

- The control lines READ & WRITE are activated by the CPU for processing memory reference instructions and IO instructions.
- But, major disadvantage of this system is, CPU spend a lot of time in performing IO related functions.

4.7.1.2. IO – MAPPED IO:

- In IO – mapped IO systems, the memory and IO address space are separate. Similarly, the control lines used for activating memory.
- Two sets of control lines are available. READ M and WRITE IO control lines.

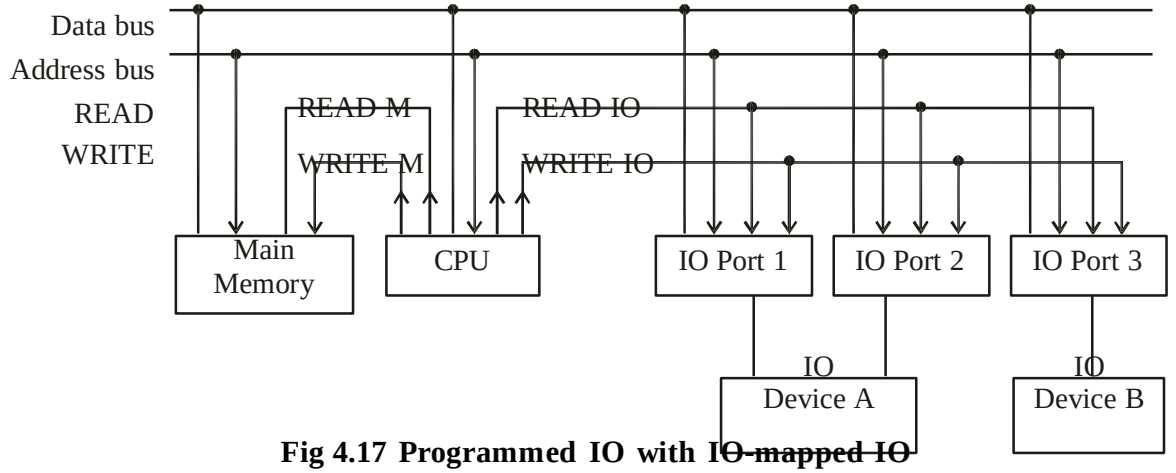


Fig 4.17 Programmed IO with IO-mapped IO

Table 4.4 Comparison of memory- mapped IO and IO- mapped IO:

S.No.	Parameter	memory- mapped IO	IO-mapped IO
1	Address Space	Memory & IO devices share the entire address space	Memory & IO devices have separate address space
2	Hardware	No additional hardware required	Additional hardware required
3	Implementation	Easy to Implement	Difficult to Implement
4	Address	Same address cannot to use to refer both memory and IO device	Same address can to use to refer both memory and IO device
5	Control Lines	Memory control lines are used to control IO devices also	Different set of control lines are used to control memory and IO
6	Control Lines used	The control lines are : READ, WRITE	The control lines are : READ M, WRITE M, READ IO, WRITE IO

4.7.2 IO INSTRUCTIONS

Two IO instructions are used to implement programmed IO. Intel 80X86 series of microprocessor series have two IO instructions called **IN & OUT.**

IN: The instruction IN X causes a word to be transferred from IO port X to the accumulator register A.

OUT: The instruction OUT X transfers a word from the accumulator register A to the port X.

4.7.3 IO DATA Transfer:

- When the CPU executes an IO instruction such as IN or OUT, the addressed IO port must be ready to respond to the instruction.
- In programmed IO, the CPU can be programmed to test the IO device's status before initiating the IO data transfer. The status is specified by a single bit information.
- The CPU must perform the following steps to determine the status of an IO device

1. Read the IO device's status bit
2. Test the status bit to determine if the device is ready to transfer data.
3. If ready, proceed with data transfer; otherwise go to step 1

- **Limitations of Programmed IO:**

Two limitations

1. The speed of the CPU is reduced due to low speed IO devices.
2. Most of the CPU time is wasted.

4.7.4 DIRECT MEMORY ACCESS:

A special control unit is provided to allow transfer of a block of data directly between an external device and the main memory, without continuous intervention by the processor. This approach is called **direct memory access, or DMA.**

When the CPU is used for IO transfer, there are two difficulties.

i. It there are many IO devices in the system, each device must be tested by the CPU which causes a delay.

ii. Programmed IO systems transfer's data through CPU rather than transferring data directly between IO device and memory.

The above problem is solved with some additional hardware. It enables an IO device to transfer a block of information to or from memory without CPU intervention. This task requires an **Interface Controller.**

The CPU & IO control is called **Direct Memory Access (DMA).**

The IO device interface controller is called as **DMA Controller.**

- **DMA Hardware:**

All the access to main memory is done via a shared system bus. The IO device is connected to the system bus via a DMA controller.

DMA controller maintains three registers. These registers are used by the DMA controller to transfer data to or from a contiguous region of memory. The registers are:

i. **Address Register (IOAR):** is used to hold the address of the next word to be transferred. It is automatically inc or dec after each word transfer.

ii. **Data Counter (DC):** is used to store the number of words that remains to be transferred. It is automatically decremented after each transfer & tested for zero. When the data count reaches zero, the DMA transfer halts.

iii. **Data Register (IODR):** is used to store the data to be transferred. When the data transfer over, the DMA controller sends an interrupt signal to the CPU to notify the end of IO data transfer.

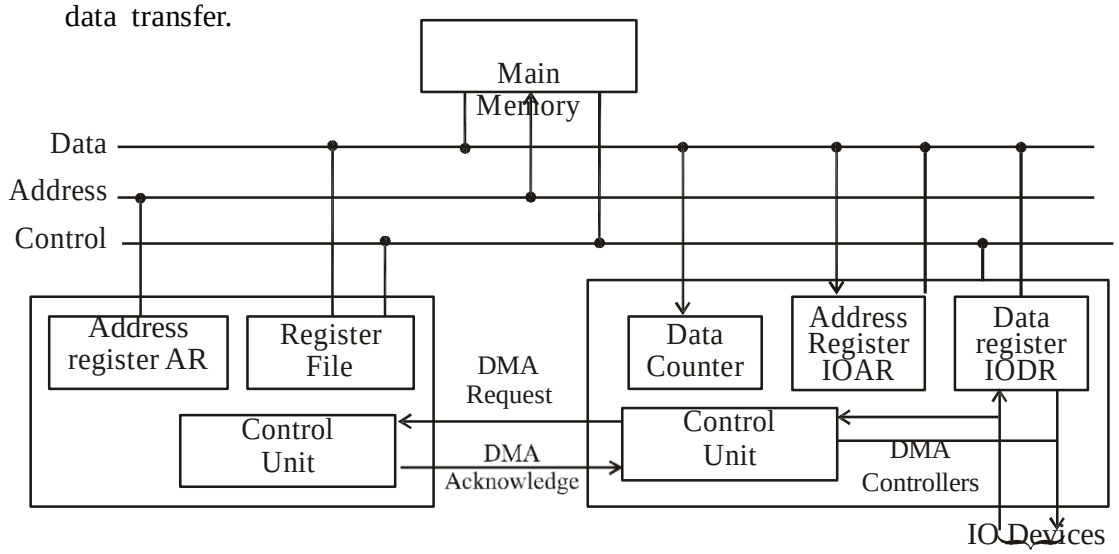


Fig 4.18 DIRECT MEMORY ACCESS HARDWARE

- **Modes of Operation:**

  DMA works in different modes. It is based on the degrees of overlap between the CPU and DMA operations. They are,

  i. Block Transfer

  ii. Cycle stealing

  iii. Transparent DMA

-> Block Transfer:

- In block transfer mode, a block of data of arbitrary length is transferred in a single burst. It is also called as burst transfer.
- During this data transfer, DMA controller is the master of the memory bus. It is useful for memories like disk drives.
- Dis adv: it make CPU inactive for a long period

-> Cycle stealing

- Cycle stealing allows the system bus to transfer one data word, after that it returns the control of the bus to the CPU.

- Cycle stealing reduces the IO transfer rate.
- It also reduces the interference by the DMA controller in CPU's memory access.

-> **Transparent DMA**

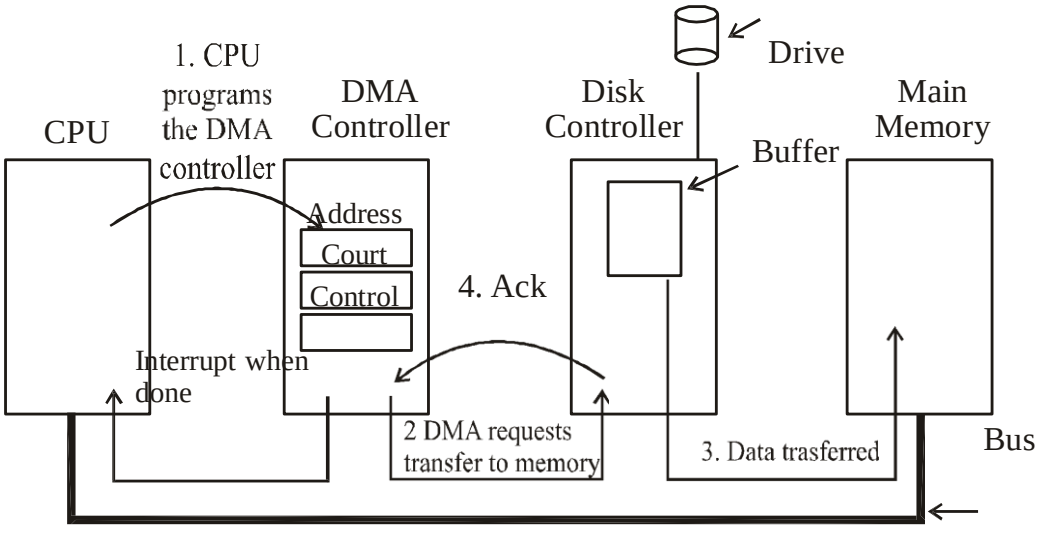
- A variation of cycle stealing mode is called transparent DMA. In this mode, DMA interface is designed in such a way that the DMA controller interferes the CPU only when the CPU is not actually using the system bus. Hence the DMA operation does not affect the CPU operation.

- **DMA Data transfer:**

  The following steps are used in DMA data transfer:

  1. The CPU executes two IO instructions to initialize the DMA registers.
     a. IOAR is initialized with the base address of the memory block to be used in data transfer.
     b. Dc is initialized with number of words to be transferred to or from that memory region.
  2. When the DMA controller is ready to transmit or receive data, it activates

Interaction of CPU and DMA:-



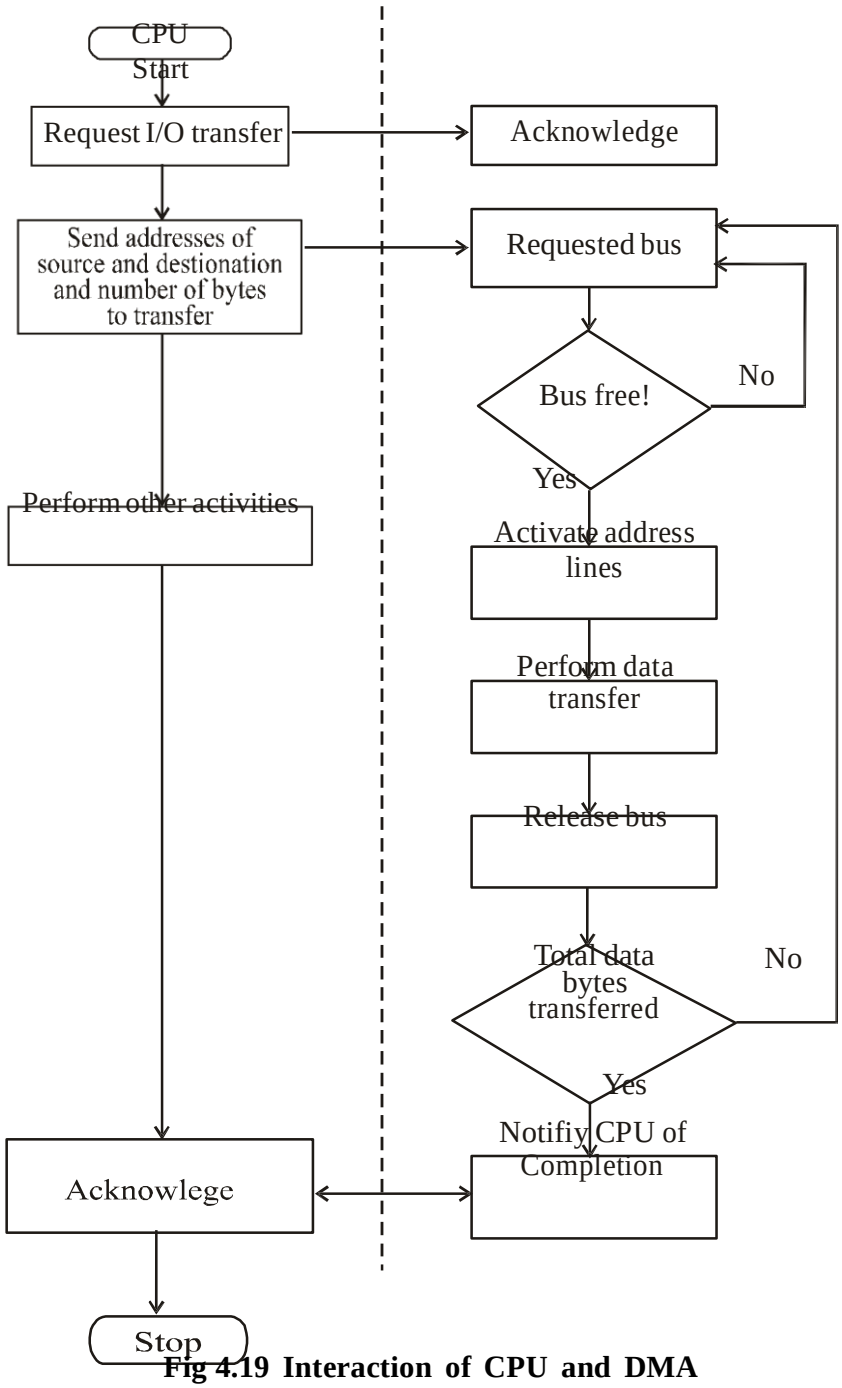


Fig 4.19 Interaction of CPU and DMA

4.7.5 DMA BUS ARBITRATION:

- Bus Arbitration refers to the process by which the current bus master accesses and then leaves the control of the bus and passes it to the another bus requesting processor unit.
- The device that performs data transfers on the bus at any given time is called a bus master.
- A conflict may arise if the number of DMA controllers or other controllers or processors try to access the common bus at the same time, but access can be given to only one of those.
- Only one processor or controller can be Bus master at the same point of time.
- To resolve these conflicts, Bus Arbitration procedure is implemented to coordinate the activities of all devices requesting memory transfers.
- The selection of the bus master must take into account the needs of various devices by establishing a priority system for gaining access to the bus.
- The Bus Arbiter decides who would become current bus master.
- The selection of bus master is usually done on the priority basis.
- There are two approaches to bus arbitration:
  1. **Centralized bus arbitration** – A single bus arbiter performs the required arbitration. The bus arbiter may be the processor or a separate controller connected to the bus.
  2. **Distributed bus arbitration** – All devices participate in the selection of the next bus master.

1. Centralized Bus Arbitration:

i. Daisy chaining

- All masters make use of the same line for bus request.
- In response to bus request, the controller sends a bus grant BG if the bus is free.
- If it is busy,itactivates the BBSY line if more than one device makes a request at same time, then the device that is closer to the arbiter wil get the bus.
- BR-Bus Request line-This is a wired-OR line, the controller only knows that a request has been made by device, but does not known which device made the request.

- BG-Bus Grant line-bus grant signal is propagated through all of the devices through this line.
- BBSY-Bus Busy-The current bus master indicates to all devices that it is using the bus by activating this open controller line called bus busy.

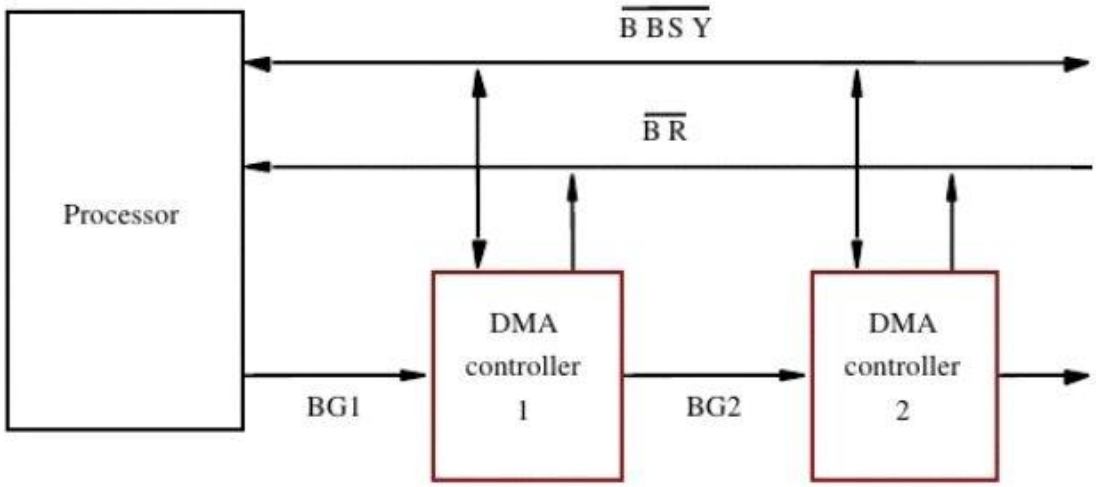


Fig 4.20 Bus Arbitration using daisy chaining method

Advantages –

- Simplicity and Scalability.
- The user can add more devices anywhere along the chain, up to a certain maximum value.

Disadvantages –

- The value of priority assigned to a device is depends on the position of master bus.
- Propagation delay is arises in this method.
- If one device fails then entire system will stop working.

ii. Polling method

- In this method controller is used to generate the address for the masters.
- All masters are connected to a local address bus. In response to bus request, the controller places the address of the bus master on the address bus.

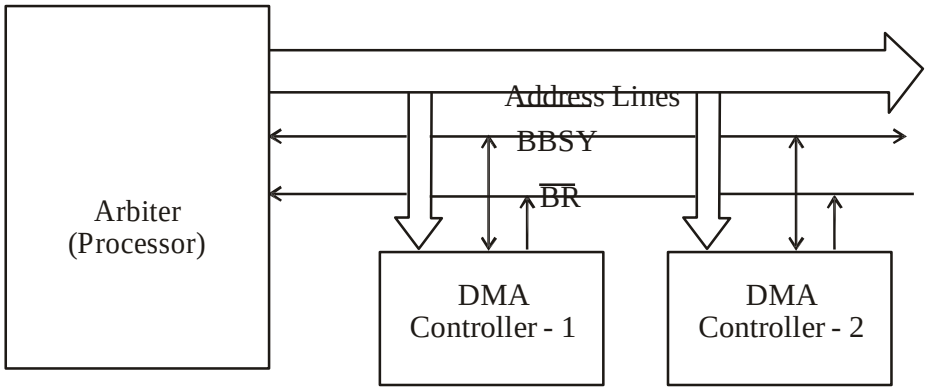


Fig 4.21 Bus arbitration using polling method

Advantages –

- This method does not favour any particular device and processor.
- The method is also quite simple.
- If one device fails then entire system will not stop working.

Disadvantages –

- Adding bus masters is different as increases the number of address lines of the circuit

iii. Independent request

- In this method each master has a separate pair of bus request and bus grant lines and each pair has priority assigned to it.
- The built in priority decoder within the controller selects the highest priority request and asserts the corresponding assigned to it.

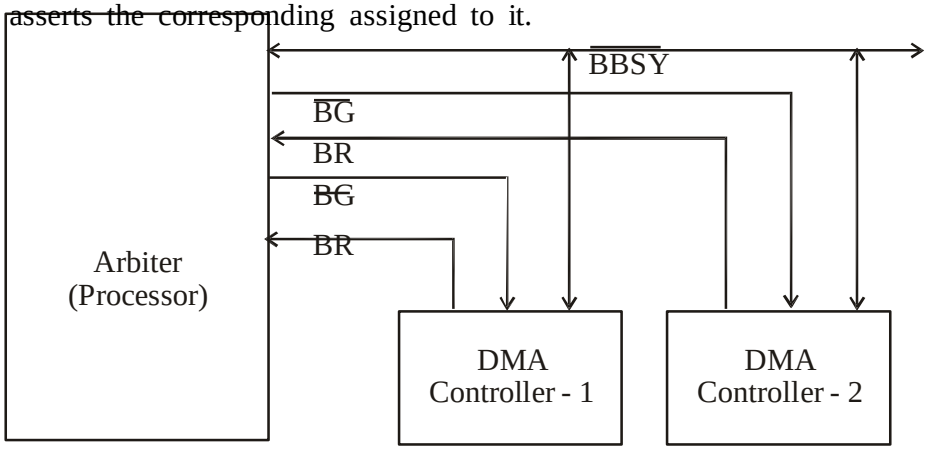


Fig 4.22 Bus arbitration using independent priority scheme

Advantages –

- This method generates fast response.

Disadvantages –

- Hardware cost is high as large no. of control lines are required.

2. Distributed Bus Arbitration:

- In distributed arbitration, all devices participate in the selection of the next bus master.
- In this scheme each device on the bus is assigned a4-bit identification number.
- The number of devices connected on the bus when one or more devices request for the control of bus, they assert the start-arbitration signal and place their 4-bit ID numbers on arbitration lines, ARB0 through ARB3.
- These four arbitration lines are all open-collector. Therefore, more than one device can place their 4-bit ID number to indicate that they need to control of bus.
- If one device puts 1 on the bus line and another device puts 0 on the same bus line, the bus line status will be 0. Device reads the status of all lines through inverters buffers so device reads bus status 0as logic 1. Scheme the device having highest ID number has highest priority.
- When two or more devices place their ID number on bus lines then it is necessary to identify the highest ID number on bus lines then it is necessary to identify the highest ID number from the status of bus line.
- Example:-
    - Consider that two devices A and B, having ID number 1 and 6, respectively are requesting the use of the bus.
    - Device A puts the bit pattern 0001, and device B puts the bit pattern 0110. With this combination the status of bus-line will be 1000; however because of inverter buffers code seen by both devices is 0111.
    - Each device compares the code formed on the arbitration line to its own ID, starting from the most significant bit.
    - If it finds the difference at any bit position, it disables its drives at that bit position and for all lower-order bits.
    - It does so by placing a 0 at the input of their drive. In our example, device detects a different on line ARB2 and hence it disables its drives on line ARB2, ARB1 and ARB0. This causes the code on the arbitration lines to change to 0110. This means that device B has won the race.

- The decentralized arbitration offers high reliability because operation of the bus is not dependent on any single device.

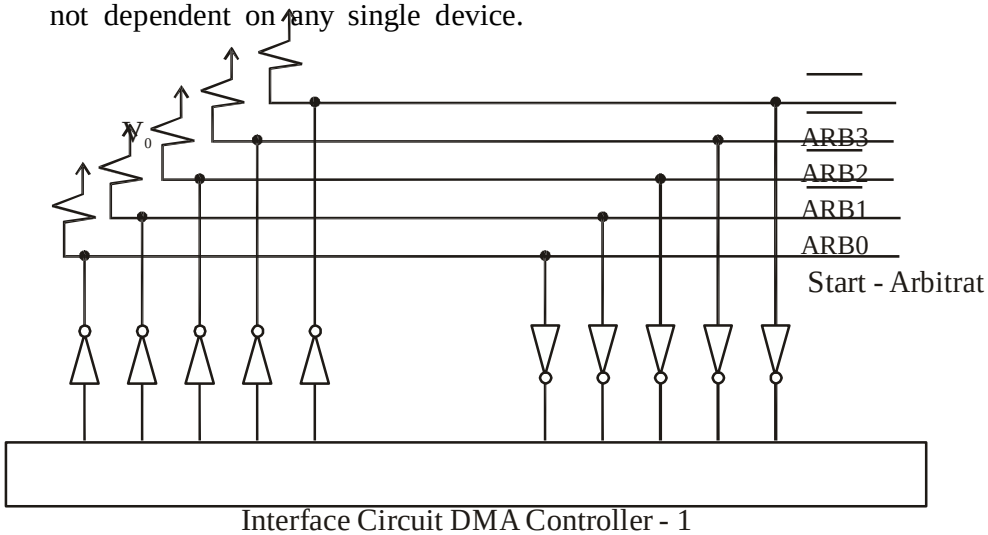


Fig 4.23 Distributed bus artbitration scheme

4.8 INTERRUPTS:

- Interrupt is a signal which has highest priority from hardware or software which processor should process its signal immediately.
- An exceptional event that's causes the cpu to temporarily transfer control from its current programs to another program.
- Interrupts are the primary means by which IO devices obtain the services of CPU.
- There are various sources of interrupts both internal and external to the CPU.IO interrupts are external to CPU.
- Interrupts can also be categorized as hardware interrupts and software interrupts.
- **Hardware Interrupts:**
  - If the signal for the processor is from external device or hardware is called hardware interrupts. Example: from keyboard we will press the key to do some action this pressing of key in keyboard will generate a signal which is given to the processor to do action, such interrupts are called hardware interrupts. Hardware interrupts can be classified into two types they are
  - **Maskable Interrupt:** The hardware interrupts which can be delayed when a much highest priority interrupt has occurred to the processor.

- **Non Maskable Interrupt:** The hardware which cannot be delayed and should process by the processor immediately.

- **Software Interrupts**: Software interrupt can also divided in to two types.
  - **Normal Interrupts:** the interrupts which are caused by the software instructions are called software instructions.
  - **Exception:** unplanned interrupts while executing a program is called Exception. For example: while executing a program if we got a value which should be divided by zero is called an exception.

4.8.1. Interrupt Handling Mechanism:-

- The instruction cycle consists of fetch, decode, execute and read/write functions. After every instruction cycle the processor will check for interrupts to be processed if there is no interrupt is present in the system it will go for the next instruction cycle which is given by the instruction register. If there is an interrupt present then it will trigger the interrupt handler, the handler will stop the present instruction which is processing and save its configuration in a register and load the program counter of the interrupt from a location which is given by the interrupt vector table.
- After processing the interrupt by the processor interrupt handler will load the instruction and its configuration from the saved register, process will start its processing where it's left. This saving the old instruction processing configuration and loading the new interrupt configuration is also called as context switching.
- The interrupt handler is also called as Interrupt service routine (ISR). There are different types of interrupt handler which will handle different interrupts. For example for the clock in a system will have its interrupt handler, keyboard it will have its interrupt handler for every device it will have its interrupt handler.
- The basic method of interrupting the CPU is activating a special control lines called INTERRUPT REQUEST which connects the interrupt source to the CPU.

The following steps are taken while handling the interrupt,

1. IO device enables the INT REQ control line.
2. Interrupt indicator is enabled in the CPU register.
3. The CPU identifies the source of the interrupt.
4. The CPU obtains the memory address of the required interrupt handler. Mostly this address is provided by the interrupting device along with its interrupt request.

5. The PC and other processor registers are saved in stack as a sun routine call.
6. The PC is loaded with address of the interrupt handler. Execution proceeds until return instruction is reached. Immediately control is transferred back to the interrupted system.

- **Interrupt Selection**
  - Interrupt selection is similar to bus arbitration process.
    1. Single Line Interrupt system
    2. Multi - Line Interrupt system
    3. Vectored interrupt

1. Single Line Interrupt system

- It is the simplest of all and uses less hardware.
- All IO ports share a common single INT REQ line.
- When an interrupt request is identified by the CPU, it must scan all the IO devices to determine the source of interrupt.
- Then an INT ACK line is activated by the CPU.
- The CPU uses polling method to identify the sources of interrupt.

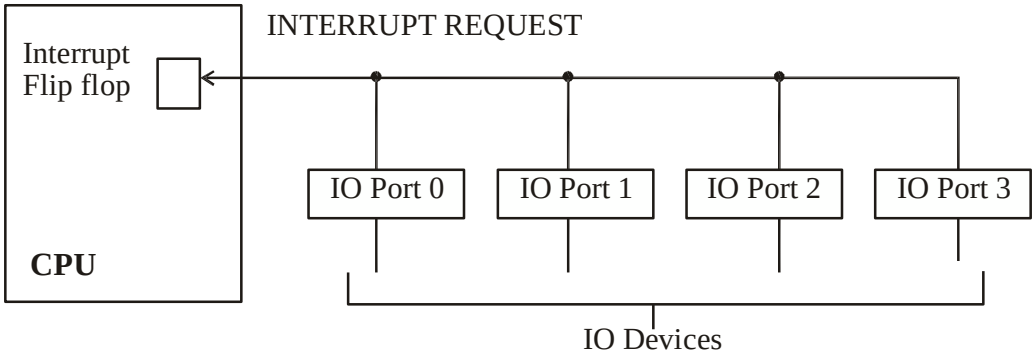
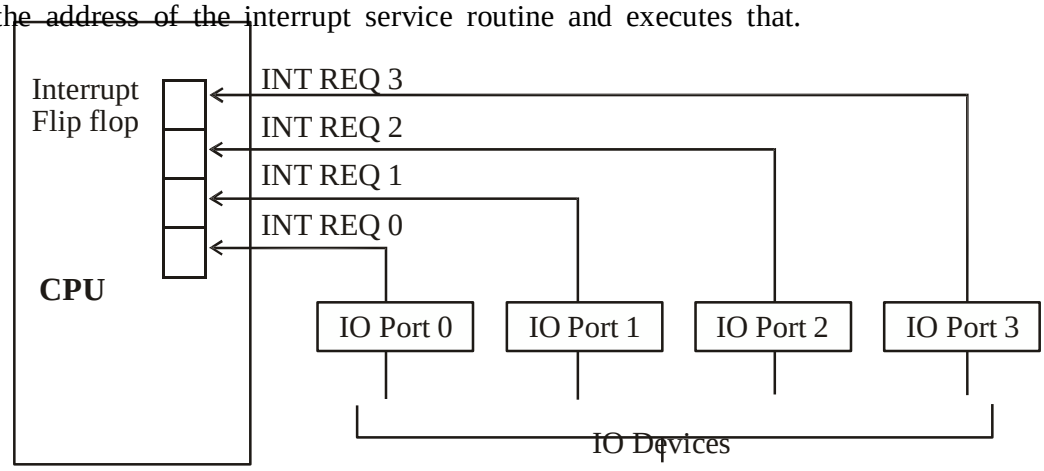


Fig 4.24 Single Interrupt System

2. Multi - Line Interrupt system

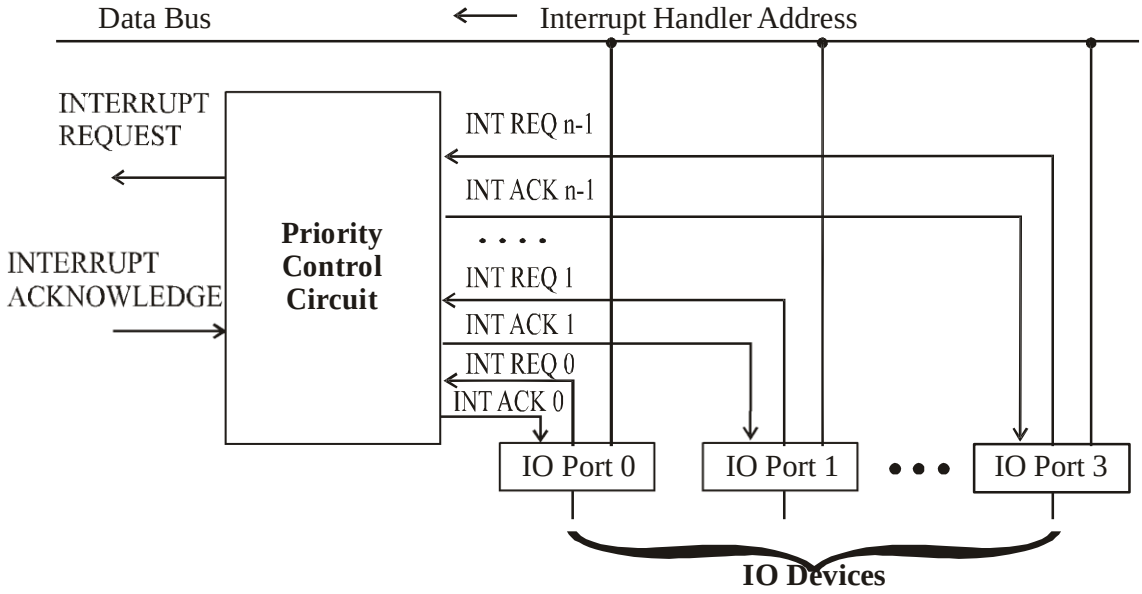
- Multi line interrupt system or Multi level interrupt system in which each device can send interrupt request independently.
- The processor has multiple interrupt lines. Each line is assigned a unique priority.

- In this method, the source of interrupt is immediately known to the CPU.CPU fetches the address of the interrupt service routine and executes that.



3. Vectored interrupt

- In vectored interrupt, the interrupting device must supply the CPU with the starting address or interrupt vector of the program. This method is most flexible in handling interrupts.



4.8.2 IO PROCESSORS (IOP)

- The DMA mode of data transfer reduces CPU's overhead in handling I/O operations. It also allows parallelism in CPU and I/O operations. Such parallelism is necessary

to avoid wastage of valuable CPU time while handling I/O devices whose speeds are much slower as compared to CPU.

- The concept of DMA operation can be extended to relieve the CPU further from getting involved with the execution of I/O operations. This gives rises to the development of special purpose processor called Input-Output Processor (IOP) or IO channel.
- The I/O processor is capable of performing actions without interruption or intervention from the CPU. The CPU only needs to initiate the I/O processor by telling it what activity to perform.
- Once the necessary actions are performed, the I/O processor then provides the results to the CPU. Doing these actions allow the I/O processor to act as a Bus to the CPU, like a CPU Bus, carrying out activities by directly interacting with memory and other devices in the computer. A more advanced I/O processor may also have memory built into it, allowing it to perform actions and activities more quickly.

For example, without an I/O processor, a computer would require the CPU to perform all actions and activities, reducing overall computer performance. However, a computer with an I/O processor would allow the CPU to send some activities to the I/O processor. While the I/O processor is performing the necessary actions for those activities, the CPU is free to carry out other activities, making the computer more efficient and increase performance.

Advantages –

- The I/O devices can directly access the main memory without the intervention by the processor in I/O processor based systems.
- It is used to address the problems that are arises in Direct memory access method.

IO instructions:

- When an IOP is used,the CPU executes only a few instruction that initate and terminate the execution of IO program via IOP.
  - **IO Instruction Executed by CPU**
    - START IO-Initates the IO operation.
    - HALT IO-it causes the IOP to terminate IO program execution
    - TEST IO-it is used to determine the status of the named io device and IOP.
  - **IO Instruction Executed by IOP**
    - **DATA TRNSFER INSTRUCTIONS-read,write,sense**
    - **BRANCH INSTRUCTION-next CCW**
    - **IO DEVICE CONTROL INSTRUCTION-rewind,seek address,print page.**

IO Organization

- The structure of IOP organization is both CPU and IOP shares the common memory via the system bus.
- There are two common kinds of programs,
  - CPU program
    - These are executed by CPU.
  - IOP program
    - These programs are executed by IOP.

ca ed OCR which s used o pass ng nforma on be ween he wo processo n he fo m of messages

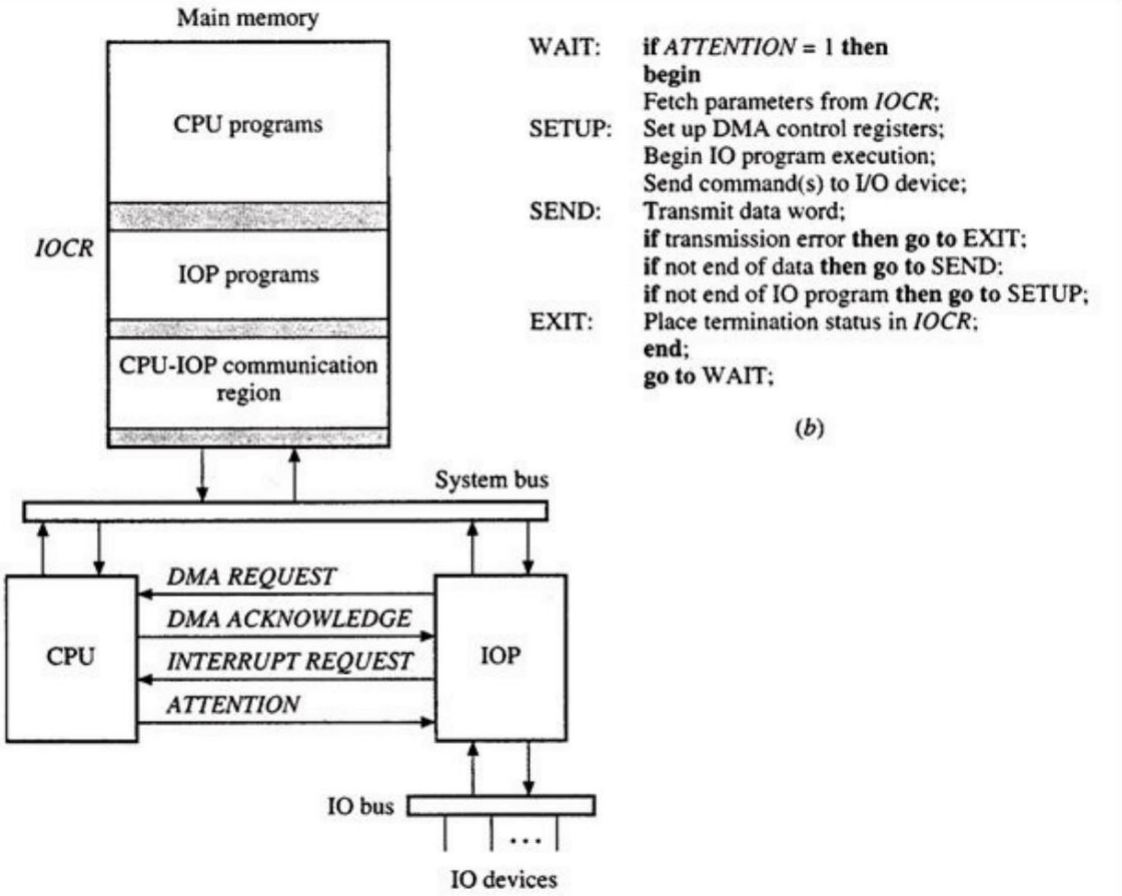
Both the programs are stored in memory. the memory also contains a communication region

IOP organ zation IOP and CPU share access to a common memoryM via he systembus

ll P i t i i ti t t t r i t r

.

i : t



Computer containing an IOP : (a) system organization and (b) CPU-IOP interaction

IOP Organization

CPU – IOP Interaction:-

There are four phases in the CPU-IOP Interaction.

They are

WAIT

- IOP checks the ATTENTION line if it is 1, then IOP fetches parameters from IOCR.

SETUP

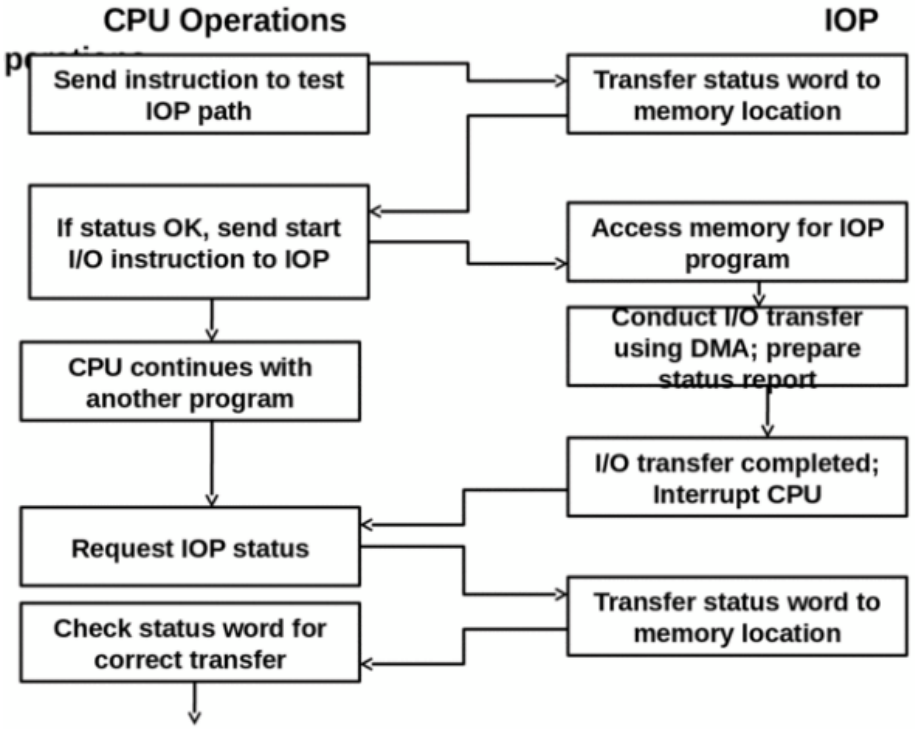
- DMA control registers are set up by CPU. The CPU begins IO program execution by sending commands to IO device.

SEND

- Data is transmitted between IO device and memory by IOP without the intervention of CPU. During transmission, if any error occurs, the control transferred to EXIT phase.

EXIT

- IOP places the termination status in IOCR and goes to WAIT phase.



4.9. Universal Serial Bus (USB):-

- A Universal Serial Bus (USB) is a common interface that enables communication between devices and a host controller such as a personal computer (PC).
- It connects peripheral devices such as digital cameras, mice, keyboards, printers, scanners, media devices, external hard drives and flash drives.
- The commercial success of the USB is due to its simplicity and low cost.
- A USB is intended to enhance plug-and-play and allow hot swapping.
- Plug-and-play enables the operating system (OS) to spontaneously configure and discover a new peripheral device without having to restart the computer.
- Hot swapping allows removal and replacement of a new peripheral without having to reboot.
- The original USB specification supports two speeds of operation, called low-speed (1.5 Megabits/s) and full-speed (12 Megabits/s)
- Later introduced USB2, called High-Speed USB.It enables data transfers at speeds up to 480 Megabits/s.
- As I/O devices continued to evolve with even higher speed requirements, USB 3 (called Superspeed) 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

4.9.1 Device Characteristics:-

- The kinds of devices that may be connected to a computer cover a wide range of functionality. The speed, volume, and timing constraints associated with data transfers to and from these devices vary significantly.
- In the case of a keyboard, one byte of data is generated every time a key is pressed, which may happen at any time. These data should be transferred to the computer promptly. Since the event of pressing a key is not synchronized to any other event in a computer system, the data generated by the keyboard are called *asynchronous*.

- Computer mice and some of the controls and manipulators used with video games are good examples.
- Consider now a different source of data. Many computers have a microphone, eitherexternally attached or built in. The sound picked up by the microphone produces an analog electrical signal, which must be converted into a digital form before it can be handled by the computer. This is accomplished by sampling the analog signal periodically. For each sample, an analog-to-digital (A/D) converter generates an *n*-bit number representing the magnitude of the sample. The number of bits, *n*, is selected based on the desired precisionwith which to represent each sample. Later, when these data are sent to a speaker, a digital-to-analog (D/A) converter is used to restore the original analog signal from the digital format. A similar approach is used with video information from a camera.
- The sampling process yields a continuous stream of digitized samples that arrive at regular intervals, synchronized with the sampling clock. Such a data stream is called *isochronous*, meaning that successive events are separated by equal periods of time.

4.9.2. Plug and play:-

- When an I/O device is connected to a computer, the operating system needs some information about it.
- It needs to know what type of device it is so that it can use the appropriate device driver. It also needs to know the addresses of the registers in the device's interface to be able to communicate with it.
- The USB standard defines both the USB hardware and the software that communicates with it.
- Its *plug-and-play* feature means that when a new device is connected, the system detects its existence automatically.
- Thesoftware determines the kind of device and how to communicate with it, as well as any special requirements it might have.
- As a result, the user simply plugs in a USB device and begins to use it, without having to get involved in any of these details.
- The USB is also hot-pluggable, which means a device can be plugged into or removed from a USB port while power is turned on.

4.9.3 **USB Architecture:-**

- The USB uses point-to-point connections and a serial transmission format.
- Whenmultiple devices are connected, they are arranged in a tree structure.
- Each node of the tree has a device called a *hub*, which acts as an intermediate transfer point between the host computer and the I/O devices.
- At the root of the tree, a *root hub* connects the entire tree to the host computer. The leaves of the tree are the I/O devices: a mouse, a keyboard, a printer, an Internet connection, a camera, or a speaker. The tree structure makes it possible to connect many devices using simple point-to-point serial links.
- If I/O devices are allowed to send messages at any time, two messages may reach the hub at the same time and interfere with each other. For this reason, the USB operates strictly on the basis of polling.
- A device may send a message only in response to a poll message from the host processor. Hence, no two devices can send messages at the same time. This restriction allows hubs to be simple, low-cost devices.
- Each device on the USB, whether it is a hub or an I/O device, is assigned a 7-bit address. This address is local to the USB tree and is not related in any way to the processor's address space.
- The root hub of the USB, which is attached to the processor, appears as a single device. The host software communicates with individual devices by sending information to the root hub, which it forwards to the appropriate device in the USB tree.
- When a device is first connected to a hub, or when it is powered on, it has the address 0.
- Periodically, the host polls each hub to collect status information and learn about new devices that may have been added or disconnected.
- When the host is informed that a new device has been connected, it reads the information in a special memory in the device'sUSB interface to learn about the device's capabilities.
- It then assigns the device a unique USB address and writes that address in one of the device's interface registers. It is this initial connection procedure that gives the USB its plug-and-play capability.

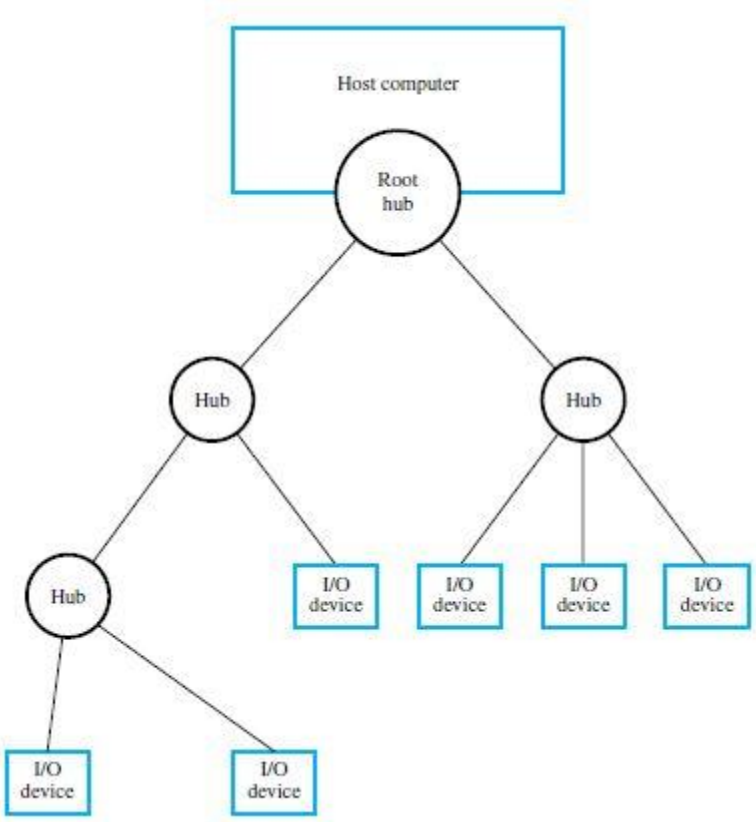
4.9.4 Isochronous Traffic on USB:-

- An important feature of the USB is its ability to support the transfer of isochronous data in a simple manner.
- Isochronous data need to be transferredat precisely timed regular intervals.
- To accommodate this type of traffic, the root hubtransmits a uniquely recognizable sequence of bits over the USB tree every millisecond.
- This sequence of bits, called a Start of Frame character, acts as a marker indicating the beginning of isochronous data, which are transmitted after this character.
- Thus, digitized audio and video signals can be transferred in a regular and precisely timed manner.

Electrical Characteristics:-

- USB connections consist of four wires, of which two carry power, +5 V and Ground, and two carry data.
- 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

The High-Speed USB uses an alternative arrangement known as *differential signalling*. The data signal is injected between two data wires twisted together. The ground wire is not involved. The receiver senses the voltage difference between the two signal wires directly, without reference to ground. The ground wire acts as a shield for the data on the twisted pair against interference from nearby wires. Differential signalling allows much lower voltages and much higher speeds to be used compared to single-ended signalling.



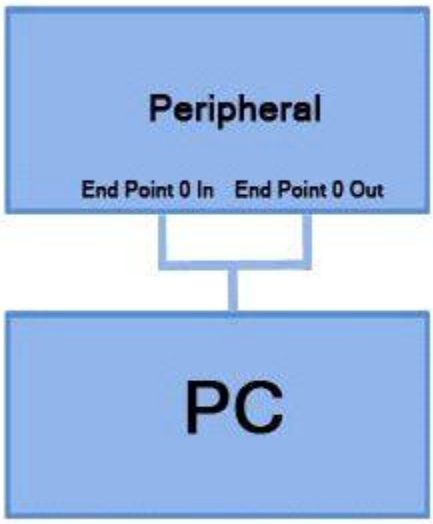
Universal Serial Bus tree structure.

USB System Architecture

- The USB is controlled by a host; there can be multiple peripherals but only one host per bus. The host can be taken as master and peripheral as slaves, whereby the former is responsible for managing the connection, transactions, and scheduling bandwidth.
- The USB system uses tiered start topology.
- It consists of 7-bit addressing; this means it can support up to 127 devices at once.
- The connection cable consists of 4 shielded wires out of which two are for power (+5V and Ground) and the remaining are twisted pair differential data signals which use NRZI (Non Return to Zero invert) scheme to transmit data.
- In order to synchronize the host and receiver clocks, Sync field is used.

Endpoints

- In USB, the information flows between host and device. The endpoints are source or sink of information in a communication channel.
- These are blocks of memory in a controller chip containing buffers for transmission and reception.
- The two endpoints can have same endpoint number but different directions.
- When the device is plugged in, only the default endpoint 0 is accessible. This endpoint receives control and status request from the host during enumeration process.
- The other endpoints are declared as per requirement after configuration of the device.



Default Endpoints

Pipes

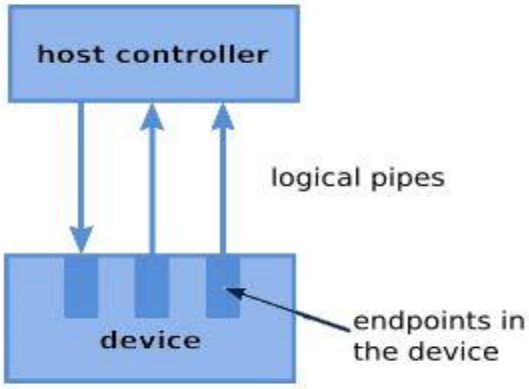
- A pipe is a logical data connection between host controller's software and device endpoint.
- The information is exchanged through this pipe. It is created during enumeration process. When the device is unplugged, unneeded pipes are removed.
- There are two types of pipes:

  Message pipes: These are bi-directional pipes which follow defined packet format. They are controlled by host and only support control transfer.

Stream pipes: These are unidirectional pipes which don't follow any specific data format. They can be controlled by host or device (peripheral) and support bulk, isochronous and interrupt types of transfer.

The **Default Control pipe** is a special type of message pipe which is bidirectional and supports control transfer type. It uses endpoint 0-IN and endpoint 0-OUT.

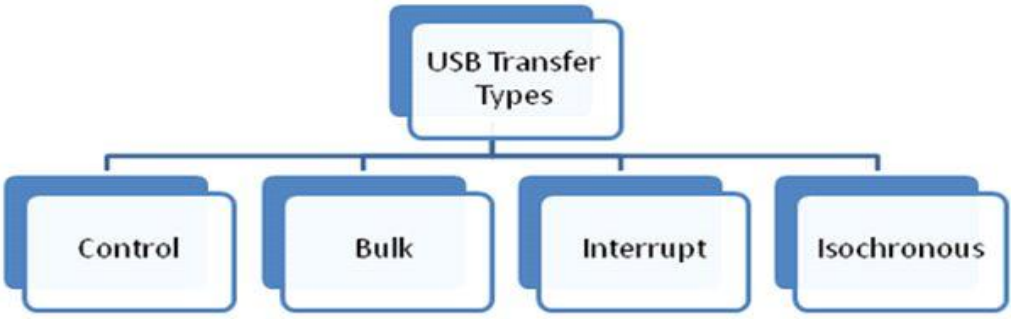
This pipe can be accessed when a device is plugged in.



Transfer Types

There are four types of transfer modes or types which can be used for communication:

- **Control Transfer:** This transfer type is used to transfer the control information while identifying and configuring the device.
- **Bulk Transfer:** In this transfer type, large amount of data is moved and time is not a critical factor here. It can be used as fillers.
- **Interrupt Transfer:** This type of transfer is for small data transmission with immediate attention.
- **Isochronous Transfer:** It is a transfer in which time is a critical factor. The transfer has to be completed in a given time.



Transaction

A single transaction contains transmission of up to three packets. These packets are:

- **Token Packet:** This packet is always sent by Host
- **Data Packet:** This packet can be sent by Host or Device.
- **Handshake Packet:** This packet provides **success/failure information** for the data packet received. If Host transmits the data packet, the device sends the handshake packet and vice versa.

Handshaking

Handshaking is a mechanism to check the success/failure of a request or to check the delivery of a packet. It is done to avoid loss of packets and to ensure successful transmission. Terms related to handshaking:

ACK – acknowledgment for data receive (success)

NAK – negative acknowledgment means no data

STALL – request not supported or endpoint halted

NYET – not yet. A case can be when device is busy and not ready for next data packet

ERR – split transaction error

Advantages of USB

- **Ease of Use**: USB was for obvious reasons designed to be a simplified interface. The simplicity of the interface lies with following aspects:

- **Single Interface for multiple devices**
- **Auto-configuratio**
- **Easy to expand**
- **Compact Size**
- **Speed**
- **Reliability**
- **Low cost**
- **Low power consumption**.

Limitations:

- **Speed:** With the introduction of USB 3.0, it is possible to achieve 5Gbits/sec. But it is still lower than Gigabit Ethernet. The FireWire 800(IEEE-1394b) is also a competitor for USB.
- **Peer to Peer Communication:** As per the USB standard, the communication takes place between the host and the peripheral. Two hosts cannot communicate directly to each other.
- **Distance:** According to USB standards, the connecting cable can be as long as 5 meters, beyond which, USB hubs need to be used for expanding the connectivity.
- **Broadcasting:** Universal Serial Bus does not provide the broadcasting feature, only individual messages can be communicated between host and peripheral.