UNIT V

MEMORY AND I/O

MEMORY HIERARCHY IN COMPUTER ARCHITECTURE

In the design of the computer system, a processor, as well as a large amount of memory devices, has been used. However, the main problem is, these parts are expensive. So the **memory organization** of the system can be done by memory hierarchy. It has several levels of memory with different performance rates. But all these can supply an exact purpose, such that the access time can be reduced. The memory hierarchy was developed depending upon the behavior of the program. This article discusses an overview of the memory hierarchy in computer architecture.

What is Memory Hierarchy?

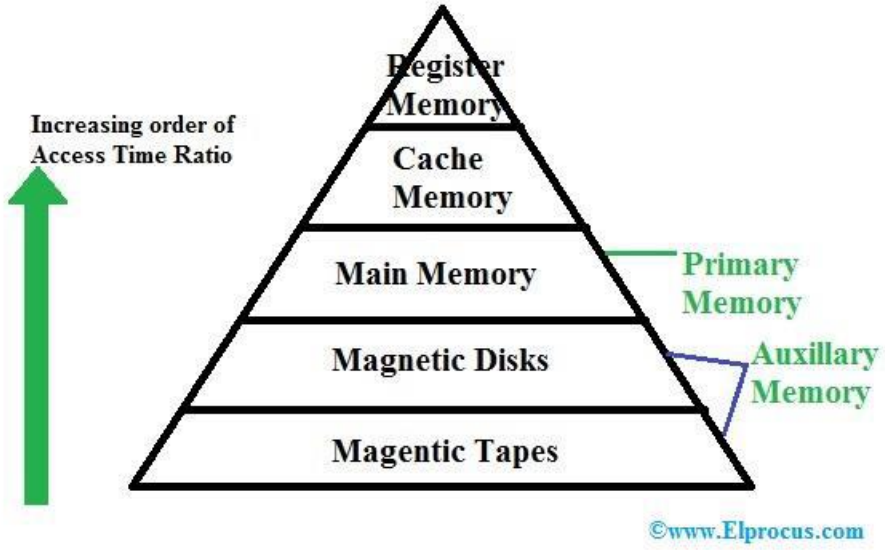
The memory in a computer can be divided into five hierarchies based on the speed as well as use. The processor can move from one level to another based on its requirements. The five hierarchies in the memory are registers, cache, main memory, magnetic discs, and magnetic tapes. The first three hierarchies are volatile memories which mean when there is no power, and then automatically they lose their stored data. Whereas the last two hierarchies are not volatile which means they store the data permanently.

A memory element is the set of storage devices which stores the binary data in the type of bits. In general, the storage of memory can be classified into two categories such as volatile as well as non- volatile.

Memory Hierarchy in Computer Architecture

The **memory hierarchy design** in a computer system mainly includes different storage devices. Most of the computers were inbuilt with extra storage to run more powerfully beyond the main memory capacity. The following **memory hierarchy diagram** is a hierarchical pyramid for computer memory. The designing of the memory hierarchy is divided into two types such as primary (Internal) memory and secondary (External) memory.

Memory Hierarchy



Primary Memory

The primary memory is also known as internal memory, and this is accessible by the processor straightly. This memory includes main, cache, as well as CPU registers.

Secondary Memory

The secondary memory is also known as external memory, and this is accessible by the processor through an input/output module. This memory includes an optical disk, magnetic disk, and magnetic tape.

Characteristics of Memory Hierarchy

The memory hierarchy characteristics mainly include the following.

Performance

Previously, the designing of a computer system was done without memory hierarchy, and the speed gap among the main memory as well as the CPU registers enhances because of the huge disparity in access time, which will cause the lower performance of the system. So, the enhancement was mandatory. The enhancement of this was designed in the memory hierarchy model due to the system's performance increase.

Ability

The ability of the memory hierarchy is the total amount of data the memory can store. Because whenever we shift from top to bottom inside the memory hierarchy, then the capacity will increase.

Access Time

The access time in the memory hierarchy is the interval of the time among the data availability as well as request to read or write. Because whenever we shift from top to bottom inside the memory hierarchy, then the access time will increase

Cost per bit

When we shift from bottom to top inside the memory hierarchy, then the cost for each bit will increase which means an internal Memory is expensive compared with external memory.

Memory Hierarchy Design

The memory hierarchy in computers mainly includes the following.

Registers

Usually, the register is a static RAM or SRAM in the processor of the computer which is used for holding the data word which is typically 64 or 128 bits. The program counter register is the most important as well as found in all the processors. Most of the processors use a status word register as well as an accumulator. A status word register is used for decision making, and the accumulator is used to store the data like mathematical operation. Usually, computers like **complex instruction set computers** have so many registers for accepting main memory, and **RISC- reduced instruction set** computers have more registers.

Cache Memory

Cache memory can also be found in the processor, however rarely it may be another **IC (integrated circuit)** which is separated into levels. The cache holds the chunk of data which are frequently used from main memory. When the processor has a single core then it will have two (or) more cache levels rarely. Present multi-core processors will be having three, 2-levels for each one core, and one level is shared.

Main Memory

The main memory in the computer is nothing but, the memory unit in the CPU that communicates directly. It is the main storage unit of the computer. This memory is fast as well as large memory used for storing the data throughout the operations of the computer. This memory is made up of RAM as well as ROM.

Magnetic Disks

The magnetic disks in the computer are circular plates fabricated of plastic otherwise metal by magnetized material. Frequently, two faces of the disk are utilized as well as many disks may be stacked on one spindle by read or write heads obtainable on every plane. All the disks in computer turn jointly at high speed. The tracks in the computer are nothing but bits which are stored within the magnetized plane in spots next to concentric circles. These are usually separated into sections which are named as sectors.

Magnetic Tape

This tape is a normal magnetic recording which is designed with a slender magnetizable covering on an extended, plastic film of the thin strip. This is mainly used to back up huge data. Whenever the computer requires to access a strip, first it will mount to access the data. Once the data is allowed, then it will be un mounted. The access time of memory will be slower within magnetic strip as well as it will take a few minutes for accessing a strip.

Advantages of Memory Hierarchy

The need for a memory hierarchy includes the following.

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

Thus, this is all about **memory hierarchy**. From the above information, finally, we can conclude that it is mainly used to decrease the bit cost, access frequency, and to increase the capacity, access time.

MEMORY MANAGEMENT

What do you mean by memory management?

Memory is the important part of the computer that is used to store the data. Its management is critical to the computer system because the amount of main memory available in a computer system is very limited. At any time, many processes are competing for it. Moreover, to increase performance, several processes are executed simultaneously. For this, we must keep several processes in the main memory, so it is even more important to manage them effectively

Memory management plays several roles in a computer system.ibers get early access to certain merchandise!

o Memory manager is used to keep track of the status of memory locations, whether it is free or allocated. It addresses primary memory by providing abstractions so that software perceives a large memory is allocated to it.

o Memory manager permits computers with a small amount of main memory to execute programs larger than the size or amount of available memory. It does this by moving information back and forth between primary memory and secondary memory by using the concept of swapping.

o The memory manager is responsible for protecting the memory allocated to each process from being corrupted by another process. If this is not ensured, then the system may exhibit unpredictable behavior.

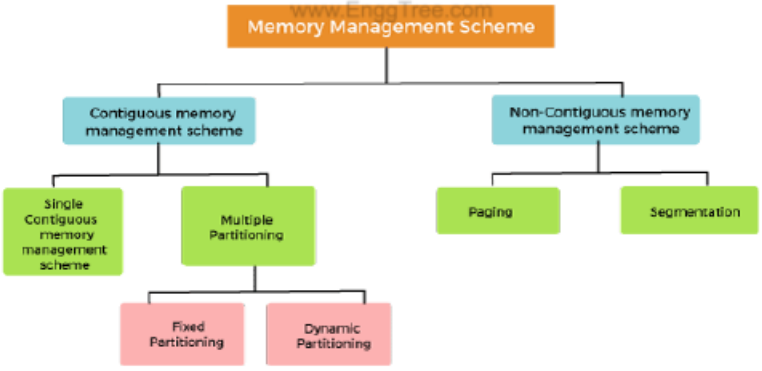
o Memory managers should enable sharing of memory space between processes. Thus, two programs can reside at the same memory location although at different times.

Memory management Techniques:

The Memory management Techniques can be classified into following main categories:

o Contiguous memory management schemes

o Non-Contiguous memory management schemes



Contiguous memory management schemes: In a Contiguous memory management scheme, each program occupies a single contiguous block of storage locations, i.e., a set of memory locations with consecutive addresses.

Single contiguous memory management schemes: The Single contiguous memory management scheme is the simplest memory management scheme used in the earliest generation of computer systems. In this scheme, the main memory is divided into two contiguous areas or partitions. The operating systems reside permanently in one partition, generally at the lower memory, and the user process is loaded into the other partition.

Advantages of Single contiguous memory management schemes:

o Simple to implement.

o Easy to manage and design.

o In a Single contiguous memory management scheme, once a process is loaded, it is given full processor's time, and no other processor will interrupt it.

Disadvantages of Single contiguous memory management schemes:

o Wastage of memory space due to unused memory as the process is unlikely to use all the available memory space.

o The CPU remains idle, waiting for the disk to load the binary image into the main memory.

o It can not be executed if the program is too large to fit the entire available main memory space.

o It does not support multiprogramming, i.e., it cannot handle multiple programs simultaneously.

Multiple Partitioning:

The single Contiguous memory management scheme is inefficient as it limits computers to execute only one program at a time resulting in wastage in memory space and CPU time. The problem of inefficient CPU use can be overcome using multiprogramming that allows more than one program to run concurrently. To switch between two processes, the operating systems need to load both processes into the main memory. The operating system needs to divide the available main memory into multiple parts to load multiple processes into the main memory. Thus multiple processes can reside in the main memory simultaneously.

The multiple partitioning schemes can be of two types:

o Fixed Partitioning

o Dynamic Partitioning

Fixed Partitioning

The main memory is divided into several fixed-sized partitions in a fixed partition memory management scheme or static partitioning. These partitions can be of the same size or different sizes. Each partition can hold a single process. The number of partitions determines the degree of multiprogramming, i.e., the maximum number of processes in memory. These partitions are made at the time of system generation and remain fixed after that.

Advantages of Fixed Partitioning memory management schemes:

o Simple to implement.

o Easy to manage and design.

Disadvantages of Fixed Partitioning memory management schemes:

o This scheme suffers from internal fragmentation.

o The number of partitions is specified at the time of system generation.

Dynamic Partitioning

The dynamic partitioning was designed to overcome the problems of a fixed partitioning scheme. In a dynamic partitioning scheme, each process occupies only as much memory as they require when loaded for processing. Requested processes are allocated memory until the entire physical memory is exhausted or the remaining space is insufficient to hold the requesting process. In this scheme the partitions used are of variable size, and the number of partitions is not defined at the system generation time.

Advantages of Dynamic Partitioning memory management schemes:

o Simple to implement.

o Easy to manage and design.

Disadvantages of Dynamic Partitioning memory management schemes:

APEC

o This scheme also suffers from internal fragmentation.

o The number of partitions is specified at the time of system segmentation.

Non-Contiguous memory management schemes:

In a Non-Contiguous memory management scheme, the program is divided into different blocks and loaded at different portions of the memory that need not necessarily be adjacent to one another. This scheme can be classified depending upon the size of blocks and whether the blocks reside in the main memory or not.

What is paging?

Paging is a technique that eliminates the requirements of contiguous allocation of main memory. In this, the main memory is divided into fixed-size blocks of physical memory called frames. The size of a frame should be kept the same as that of a page to maximize the main memory and avoid external fragmentation.

Advantages of paging:

o Pages reduce external fragmentation.

o Simple to implement.

o Memory efficient.

o Due to the equal size of frames, swapping becomes very easy.

o It is used for faster access of data.

What is Segmentation?

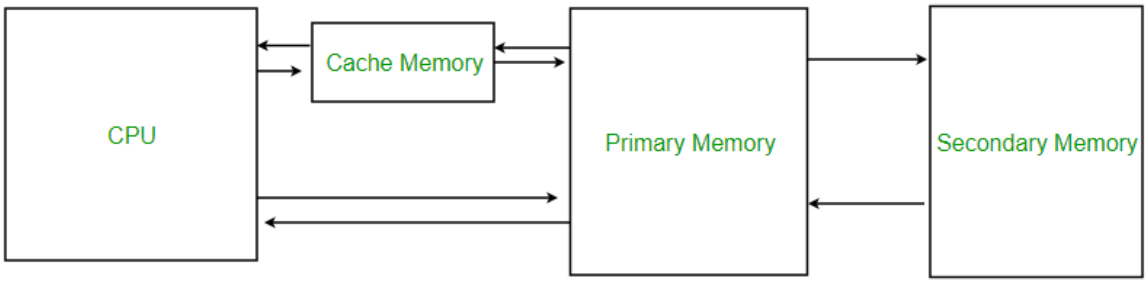
Segmentation is a technique that eliminates the requirements of contiguous allocation of main memory. In this, the main memory is divided into variable-size blocks of physical memory called segments. It is based on the way the programmer follows to structure their programs. With segmented memory allocation, each job is divided into several segments of different sizes, one for each module. Functions, subroutines, stack, array, etc., are examples of such modules.

CACHE MEMORY

Cache Memory is a special very high-speed memory. It is used to speed up and synchronize with high-speed CPU. Cache memory is costlier than main memory or disk memory but more economical than CPU registers. Cache memory is an

APEC

extremely fast memory type that acts as a buffer between RAM and the CPU. It holds frequently requested data and instructions so that they are immediately available to the CPU when needed. Cache memory is used to reduce the average time to access data from the Main memory. The cache is a smaller and faster memory that stores copies of the data from frequently used main memory locations. There are various different independent caches in a CPU, which store instructions and data.



Levels of memory:

 Level 1 or Register – It is a type of memory in which data is stored and accepted that are immediately stored in CPU. Most commonly used register is accumulator, Program counter, address register etc.

 Level 2 or Cache memory – It is the fastest memory which has faster access time where data is temporarily stored for faster access.

 Level 3 or Main Memory – It is memory on which computer works currently. It is small in size and once power is off data no longer stays in this memory.

 Level 4 or Secondary Memory – It is external memory which is not as fast as main memory but data stays permanently in this memory.

Cache Performance: When the processor needs to read or write a location in main memory, it first checks for a corresponding entry in the cache.

 If the processor finds that the memory location is in the cache, a **cache hit** has occurred and data is read from the cache.

 If the processor **does not** find the memory location in the cache, a **cache miss** has occurred. For a cache miss, the cache allocates a new entry and copies in data from main memory, then the request is fulfilled from the contents of the cache.

The performance of cache memory is frequently measured in terms of a quantity called **Hit ratio.**

Hit ratio = hit / (hit + miss) = no. of hits/total accesses

We can improve Cache performance using higher cache block size, and higher associativity, reduce miss rate, reduce miss penalty, and reduce the time to hit in the cache.

Cache Mapping: There are three different types of mapping used for the purpose of cache memory which is as follows: Direct mapping, Associative mapping, and Set-Associative mapping. These are explained below.

A. Direct Mapping

The simplest technique, known as direct mapping, maps each block of main memory into only one possible cache line. or In Direct mapping, assign each memory block to a specific line in the cache. If a line is previously taken up by a memory block when a new block needs to be loaded, the old block is trashed. An address space is split into two parts index field and a tag field. The cache is used to store the tag field whereas the rest is stored in the main memory. Direct mapping`s performance is directly proportional to the Hit ratio.

i = j modulo m

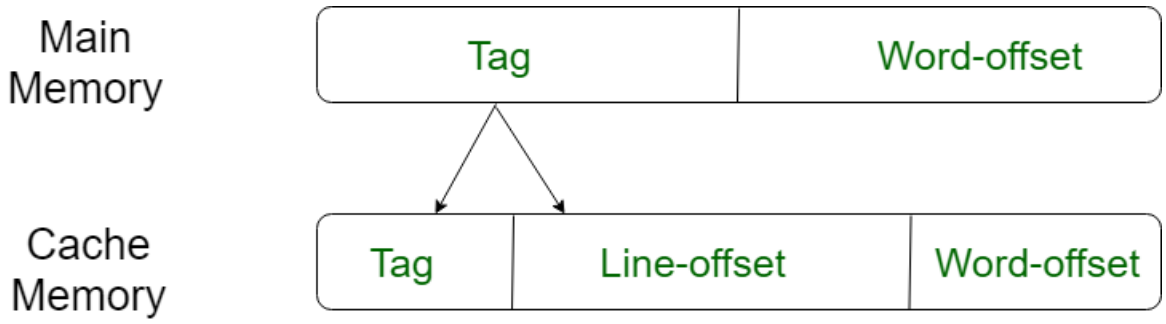
where

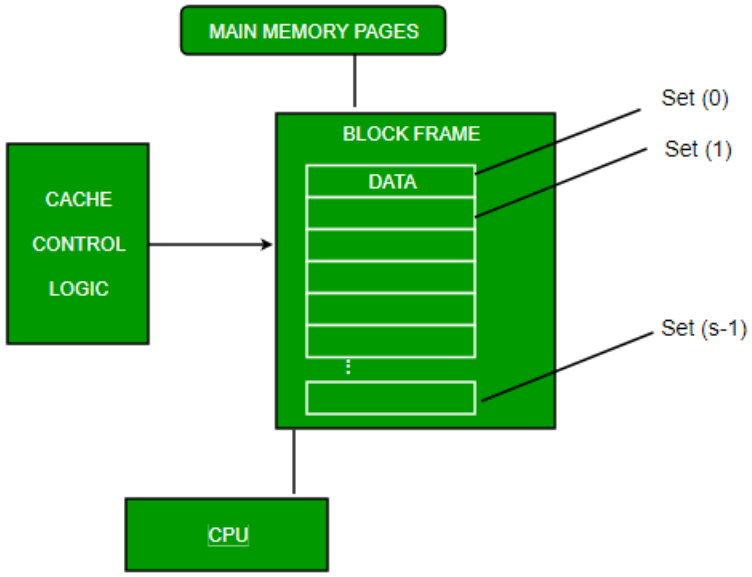
i=cache line number

j= main memory block number

m=number of lines in the cache

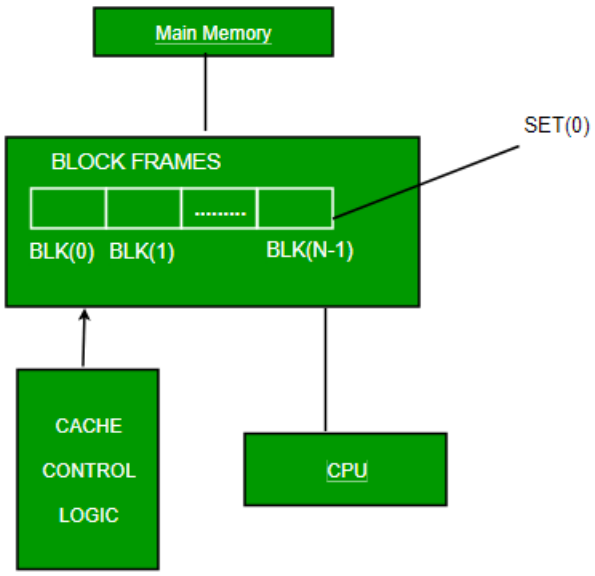
1. For purposes of cache access, each main memory address can be viewed as consisting of three fields. The least significant w bits identify a unique word or byte within a block of main memory. In most contemporary machines, the address is at the byte level. The remaining s bits specify one of the 2s blocks of main memory. The cache logic interprets these s bits as a tag of s-r bits (most significant portion) and a line field of r bits. This latter field identifies one of the m=2r lines of the cache. Line offset is index bits in the direct mapping.





B. Associative Mapping

In this type of mapping, the associative memory is used to store content and addresses of the memory word. Any block can go into any line of the cache. This means that the word id bits are used to identify which word in the block is needed, but the tag becomes all of the remaining bits. This enables the placement of any word at any place in the cache memory. It is considered to be the fastest and the most flexible mapping form. In associative mapping the index bits are zero.



C. Set-associative Mapping

This form of mapping is an enhanced form of direct mapping where the drawbacks of direct mapping are removed. Set associative addresses the problem of possible thrashing in the direct mapping method. It does this by saying that instead of having exactly one line that a block can map to in the cache, we will group a few lines together creating a ***set***. Then a block in memory can map to any one of the lines of a specific set. Set-associative mapping allows that each word that is present in the cache can have two or more words in the main memory for the same index address. Set associative cache mapping combines the best of direct and associative cache mapping techniques. In set associative mapping the index bits are given by the set offset bits. In this case, the cache consists of a number of sets, each of which consists of a number of lines. The relationships arem = v * k

i= j mod v

where

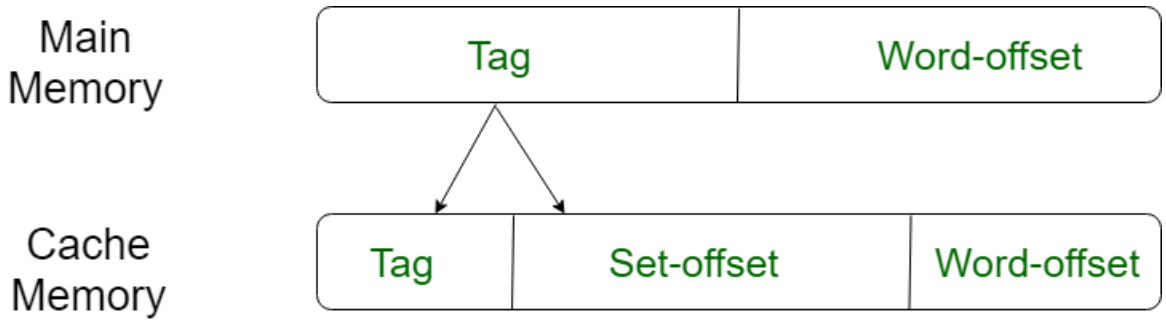
i=cache set number

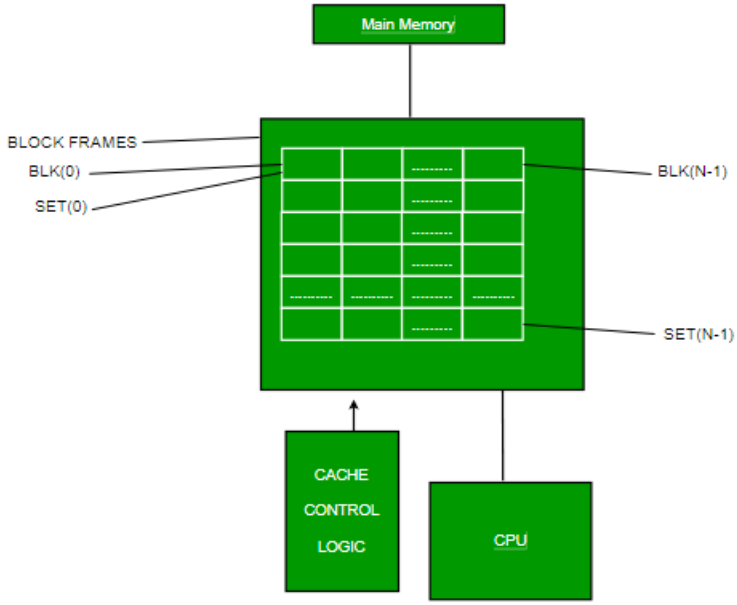
j=main memory block number

v=number of sets

m=number of lines in the cache number of sets

k=number of lines in each set





Application of Cache Memory:

1. Usually, the cache memory can store a reasonable number of blocks at any given time, but this number is small compared to the total number of blocks in the main memory.

2. The correspondence between the main memory blocks and those in the cache is specified by a mapping function.

3. **Primary Cache –** A primary cache is always located on the processor chip. This cache is small and its access time is comparable to that of processor registers.

4. **Secondary Cache –** Secondary cache is placed between the primary cache and the rest of the memory. It is referred to as the level 2 (L2) cache. Often, the Level 2 cache is also housed on the processor chip.

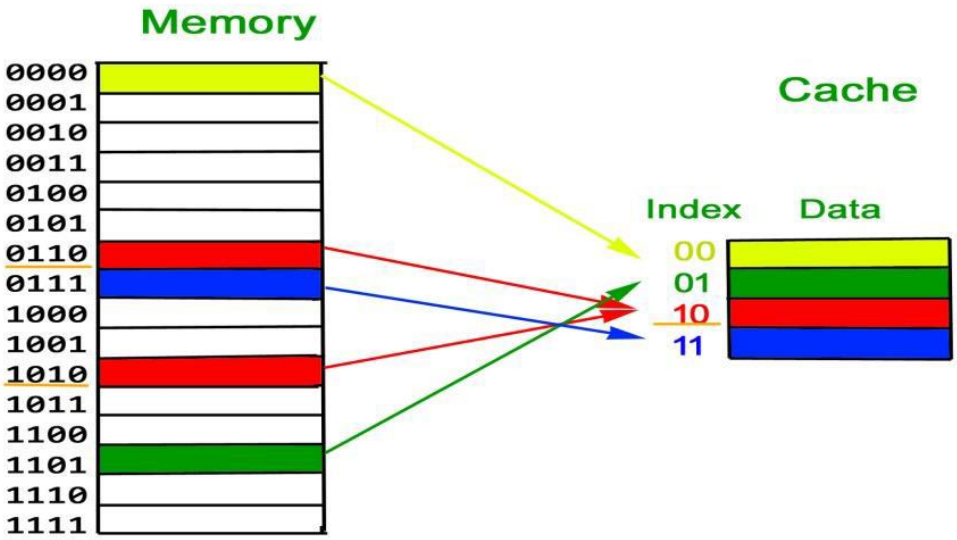
5. **Spatial Locality of reference** This says that there is a chance that the element will be present in close proximity to the reference point and next time if again searched then more close proximity to the point of reference.

6. **Temporal Locality of reference** In this Least recently used algorithm will be used. Whenever there is page fault occurs within a word will not only load word in main memory but complete page fault will be loaded because the spatial locality of reference

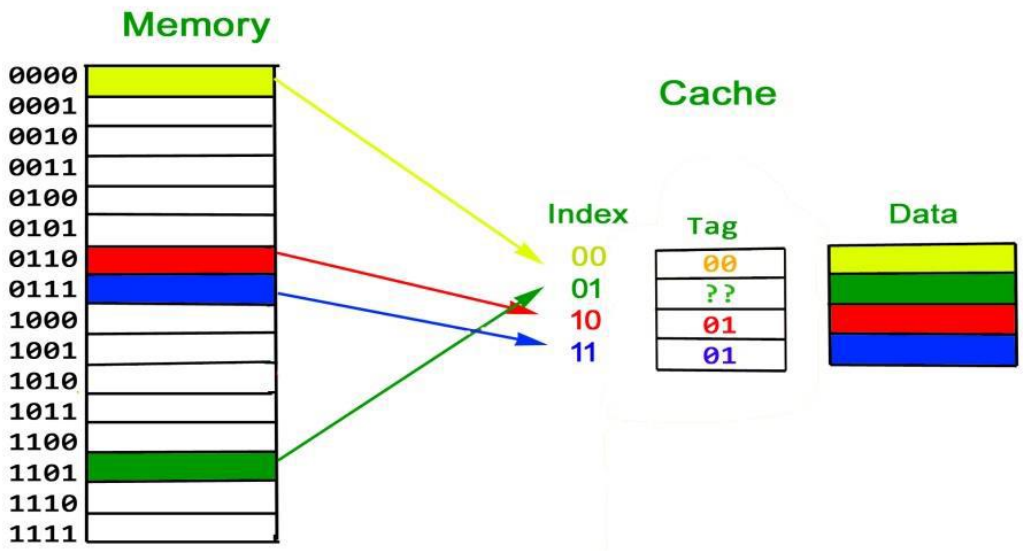
rule says that if you are referring to any word next word will be referred to in its register that's why we load complete page table so the complete block will be loaded.

Cache Organization

Cache is close to CPU and faster than main memory. But at the same time is smaller than main memory. The cache organization is about mapping data in memory to a location in cache. **A Simple Solution:** One way to go about this mapping is to consider last few bits of long memory address to find small cache address, and place them at the found address. **Problems With Simple Solution:** The problem with this approach is, we lose the information about high order bits and have no way to find out the lower order bits belong to which higher order bits.

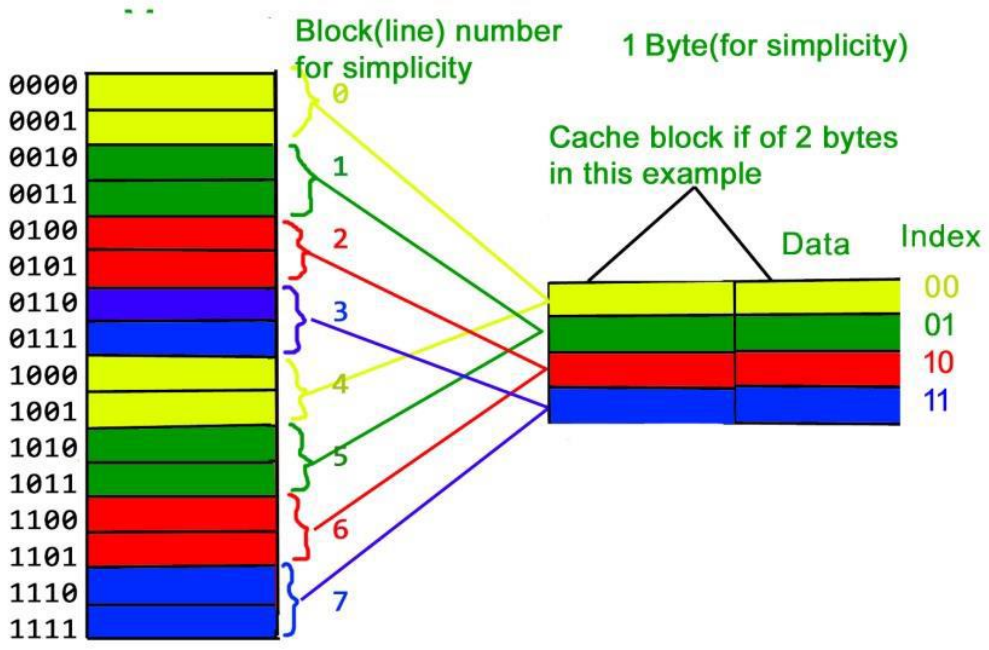


Solution is Tag: To handle above problem, more information is stored in cache to tell which block of memory is stored in cache. We store additional information as **Tag**



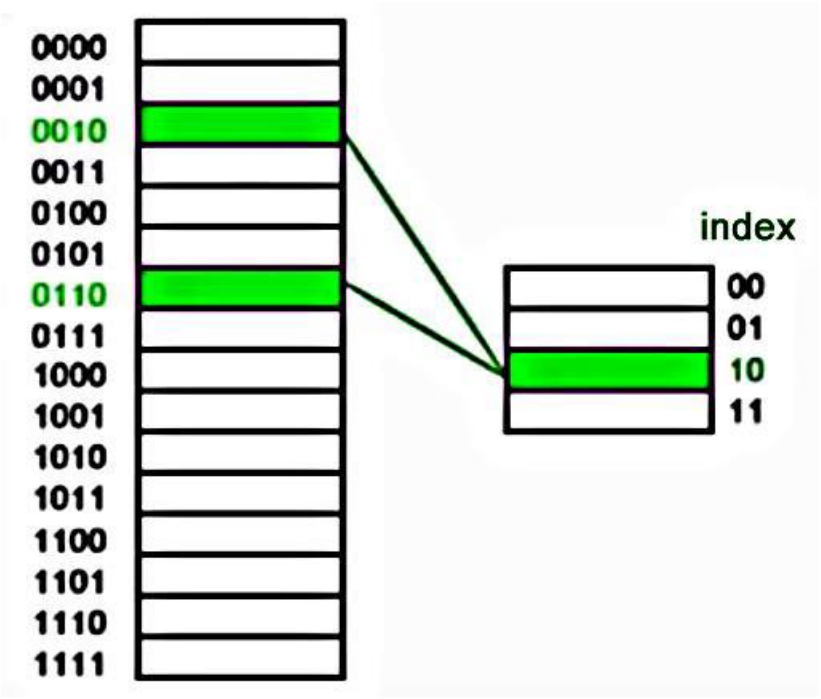
What is a Cache Block? Since programs have Spatial Locality (Once a location is retrieved, it is highly probable that the nearby locations would be retrieved in near future).

So a cache is organized in the form of blocks. Typical cache block sizes are 32 bytes or 64 bytes.

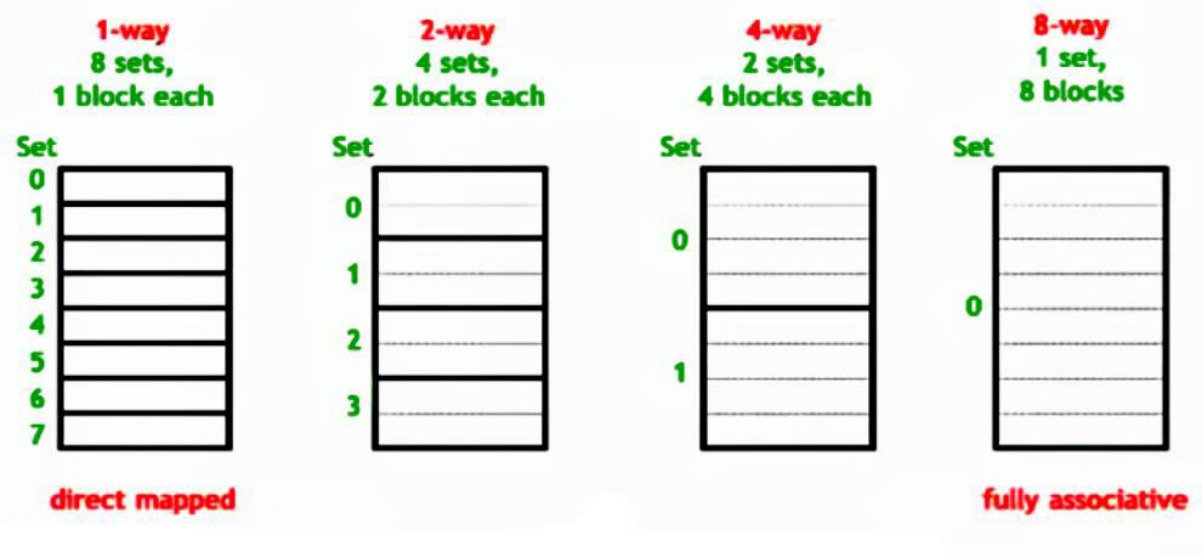


The above arrangement is Direct Mapped Cache and it has following problem We have discussed above that last few bits of memory addresses are being used to address in cache and remaining bits are stored as tag. Now imagine that cache is very small and addresses of 2 bits. Suppose we use the last two bits of main memory address to decide the cache (as shown in below diagram). So if

a program accesses 2, 6, 2, 6, 2, …, every access would cause a hit as 2 and 6 have to be stored in same location in cache.



Solution to above problem – Associativity What if we could store data at any place in cache, the above problem won't be there? That would slow down cache, so we do something in between.



VIRTUAL MEMORY

Virtual Memory

Virtual Memory is a storage scheme that provides user an illusion of having a very big main memory. This is done by treating a part of secondary memory as the main memory.

In this scheme, User can load the bigger size processes than the available main memory by having the illusion that the memory is available to load the process.

Instead of loading one big process in the main memory, the Operating System loads the different parts of more than one process in the main memory.

By doing this, the degree of multiprogramming will be increased and therefore, the CPU utilization will also be increased.

How Virtual Memory Works?

In modern word, virtual memory has become quite common these days. In this scheme, whenever some pages needs to be loaded in the main memory for the execution and the memory is not available for those many pages, then in that case, instead of stopping the pages from entering in the main memory, the OS search for the RAM area that are least used in the recent times or that are not referenced and copy that into the secondary memory to make the space for the new pages in the main memory.

Since all this procedure happens automatically, therefore it makes the computer feel like it is having the unlimited RAM.

Demand Paging

Demand Paging is a popular method of virtual memory management. In demand paging, the pages of a process which are least used, get stored in the secondary memory.

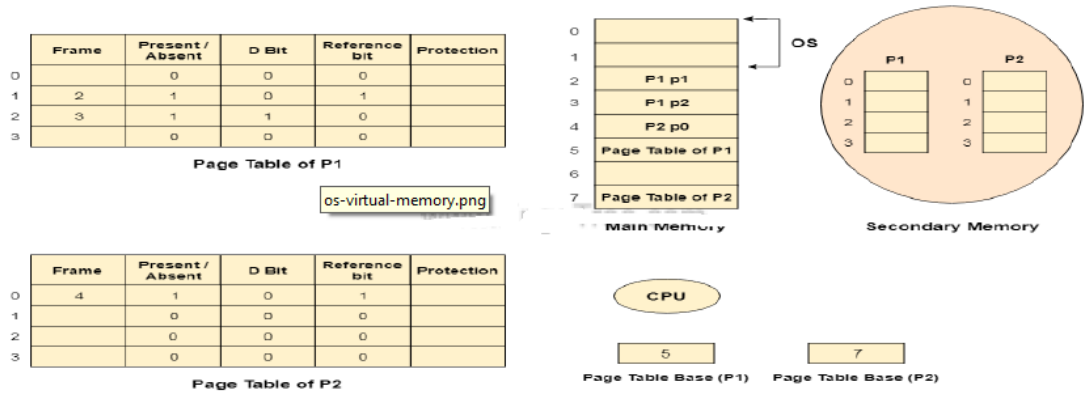
A page is copied to the main memory when its demand is made or page fault occurs. There are various page replacement algorithms which are used to determine the pages which will be replaced. We will discuss each one of them later in detail.

Snapshot of a virtual memory management system

Let us assume 2 processes, P1 and P2, contains 4 pages each. Each page size is 1 KB. The main memory contains 8 frame of 1 KB each. The OS resides in the first two partitions. In the third partition, 1st page of P1 is stored and the other frames are also shown as filled with the different pages of processes in the main memory.

The page tables of both the pages are 1 KB size each and therefore they can be fit in one frame each. The page tables of both the processes contain various information that is also shown in the image.

The CPU contains a register which contains the base address of page table that is 5 in the case of P1 and 7 in the case of P2. This page table base address will be added to the page number of the Logical address when it comes to accessing the actual corresponding entry.



Advantages of Virtual Memory

1. The degree of Multiprogramming will be increased.
2. User can run large application with less real RAM.
3. There is no need to buy more memory RAMs.

Disadvantages of Virtual Memory

1. The system becomes slower since swapping takes time.
2. It takes more time in switching between applications.
3. The user will have the lesser hard disk space for its use.

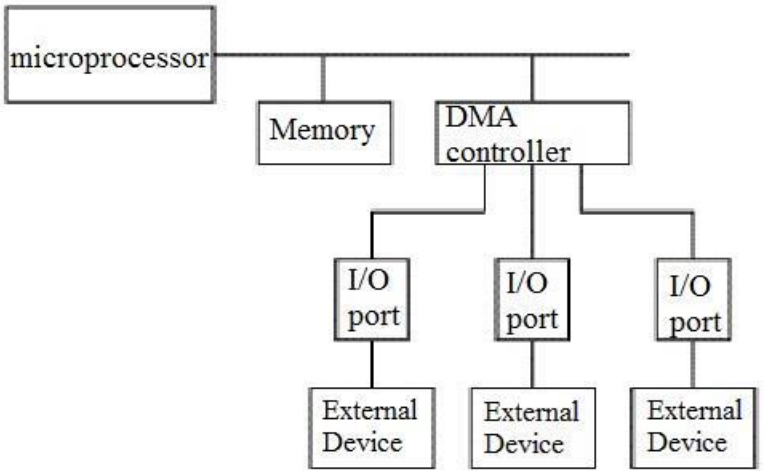
DIRECT MEMORY ACCESS

For the execution of a computer program, it requires the synchronous working of more than one component of a computer. For example, Processors – providing necessary control information, addresses…etc, buses – to transfer information and data to and from memory to I/O devices…etc. The interesting factor of the system would be the way it handles the transfer of information among processor, memory and I/O devices. Usually, processors control all the process of transferring data, right from initiating the transfer to the storage of data at the

destination. This adds load on the processor and most of the time it stays in the ideal state, thus decreasing the efficiency of the system. To speed up the transfer of data between I/O devices and memory, DMA controller acts as station master. DMA controller transfers data with minimal intervention of the processor.

What is a DMA Controller?

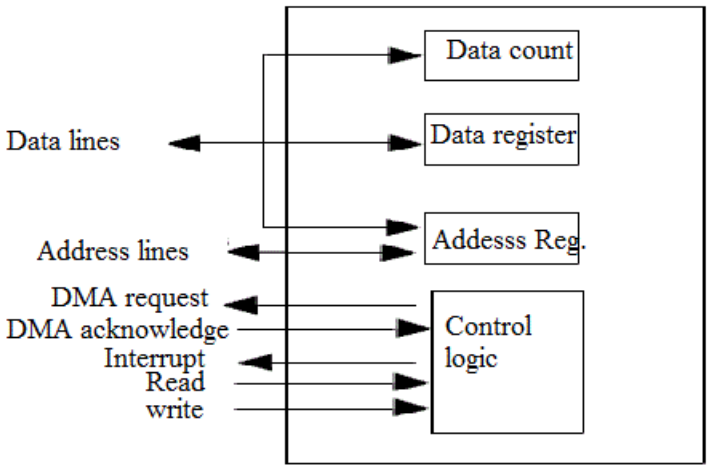
The term DMA stands for direct memory access. The hardware device used for direct memory access is called the DMA controller. DMA controller is a control unit, part of I/O device's interface circuit, which can transfer blocks of data between I/O devices and main memory with minimal intervention from the processor.



DMA Controller Diagram in Computer Architecture

DMA controller provides an interface between the bus and the input-output devices. Although it transfers data without intervention of processor, it is controlled by the processor. The processor initiates the DMA controller by sending the starting address, Number of words in the data block and direction of transfer of data .i.e. from I/O devices to the memory or from main memory to I/O devices. More than one external device can be connected to the DMA controller.

DMA controller contains an address unit, for generating addresses and selecting I/O device for transfer. It also contains the control unit and data count for keeping counts of the number of blocks transferred and indicating the direction of transfer of data. When the transfer is completed, DMA informs the processor by raising an interrupt. The typical block diagram of the DMA controller is shown in the figure below.



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.

MA 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.

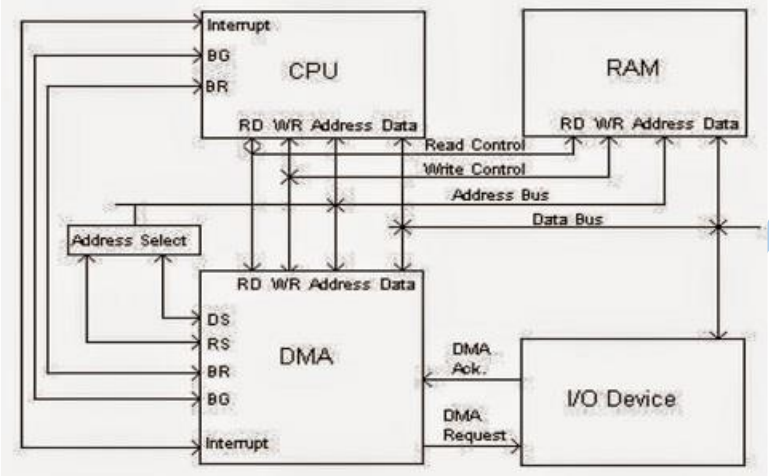
When an I/O device wants to initiate the transfer then it sends a DMA request signal to the DMA controller, for which the controller acknowledges if it is free. Then the controller requests the processor for the bus, raising the bus request signal. After receiving the bus grant signal it transfers the data from the device. For n channeled DMA controller n number of external devices can be connected.

The DMA transfers the data in three modes which include the following.

a) Burst Mode: In this mode DMA handover the buses to CPU only after completion of whole data transfer. Meanwhile, if the CPU requires the bus it has to stay ideal and wait for data transfer.

b) **Cycle Stealing Mode**: In this mode, DMA gives control of buses to CPU after transfer of every byte. It continuously issues a request for bus control, makes the transfer of one byte and returns the bus. By this CPU doesn't have to wait for a long time if it needs a bus for higher priority task.

c) **Transparent Mode:** Here, DMA transfers data only when CPU is executing the instruction which does not require the use of buses.



Advantages and Disadvantages of DMA Controller

The advantages and disadvantages of DMA controller include the following.

Advantages

 DMA speedups the memory operations by bypassing the involvement of the CPU.

 The work overload on the CPU decreases.

 For each transfer, only a few numbers of clock cycles are required

Disadvantages

 Cache coherence problem can be seen when DMA is used for data transfer.

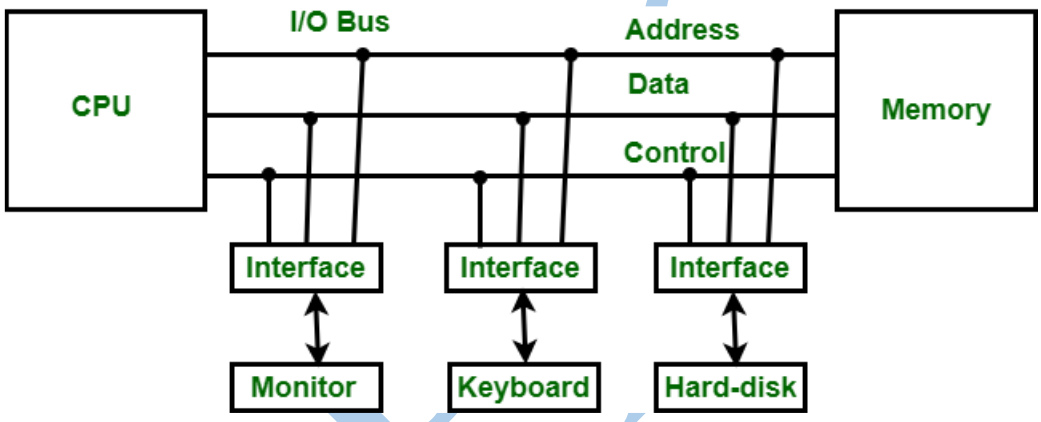
 Increases the price of the system.

DMA (Direct Memory Access) controller is being used in graphics cards, network cards, sound cards etc… DMA is also used for intra-chip transfer in

multi-core processors. Operating in one of its three modes, DMA can considerably reduce the load of the processor.

INTRODUCTION TO INPUT- OUTPUT INTERFACE

Input -Output Interface is used as an method which helps in transferring of information between the internal storage devices i.e. memory and the external peripheral device . A peripheral device is that which provide input and output for the computer, it is also called Input-Output devices. For Example: A keyboard and mouse provide Input to the computer are called input devices while a monitor and printer that provide output to the computer are called output devices. Just like the external hard-drives, there is also availability of some peripheral devices which are able to provide both input and output.



In micro-computer base system, the only purpose of peripheral devices is just to provide **special communication links** for the interfacing them with the CPU. To resolve the differences between peripheral devices and CPU, there is a special need for communication links.

The major differences are as follows:

1. The nature of peripheral devices is electromagnetic and electro-mechanical. The nature of the CPU is electronic. There is a lot of difference in the mode of operation of both peripheral devices and CPU.

2. There is also a synchronization mechanism because the data transfer rate of peripheral devices are slow than CPU.

3. In peripheral devices, data code and formats are differ from the format in the CPU and memory.

4. The operating mode of peripheral devices are different and each may be controlled so as not to disturb the operation of other peripheral devices connected to CPU.

There is a special need of the additional hardware to resolve the differences between CPU and peripheral devices to supervise and synchronize all input and output devices.

Functions of Input-Output Interface:

1. It is used to synchronize the operating speed of CPU with respect to input-output devices.

2. It selects the input-output device which is appropriate for the interpretation of the input-output device.

3. It is capable of providing signals like control and timing signals.

4. In this data buffering can be possible through data bus.

5. There are various error detectors.

6. It converts serial data into parallel data and vice-versa.

7. It also convert digital data into analog signal and vice-versa.

I/O Interface (Interrupt and DMA Mode)

The method that is used to transfer information between internal storage and external I/O devices is known as I/O interface. The CPU is interfaced using special communication links by the peripherals connected to any computer system. These communication links are used to resolve the differences between CPU and peripheral. There exists special hardware components between CPU and peripherals to supervise and synchronize all the input and output transfers that are called interface units.

Mode 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.

Data transfer to and from the peripherals may be done in any of the three possible ways

1. Programmed I/O.

APEC

2. Interrupt- initiated I/O.

3. Direct memory access(DMA).

Now let's discuss each mode one by one.

1. **Programmed I/O:** It is due to the result of the I/O instructions that are written in the computer program. Each data item transfer is initiated by an instruction in the program. Usually the transfer is from a CPU register and memory. In this case it requires constant monitoring by the CPU of the peripheral devices.

Example of Programmed I/O: In this case, the I/O device does not have direct access to the memory unit. A transfer from I/O device to memory requires the execution of several instructions by the CPU, including an input instruction to transfer the data from device to the CPU and store instruction to transfer the data from CPU to memory. In programmed I/O, the CPU stays in the program loop until the I/O unit indicates that it is ready for data transfer. This is a time consuming process since it needlessly keeps the CPU busy. This situation can be avoided by using an interrupt facility. This is discussed below.

2. **Interrupt- initiated I/O:** Since in the above case we saw the CPU is kept busy unnecessarily. This situation can very well be avoided by using an interrupt driven method for data transfer. By using interrupt facility and special commands to inform the interface to issue an interrupt request signal whenever data is available from any device. In the meantime the CPU can proceed for any other program execution. The interface meanwhile keeps monitoring the device. Whenever it is determined that the device is ready for data transfer it initiates an interrupt request signal to the computer. Upon detection of an external interrupt signal the CPU stops momentarily the task that it was already performing, branches to the service program to process the I/O transfer, and then return to the task it was originally performing.

Note: Both the methods programmed I/O and Interrupt-driven I/O require the active intervention of the processor to transfer data between memory and the I/O module, and any data transfer must transverse a path through the processor. Thus both these forms of I/O suffer from two inherent drawbacks.

 The I/O transfer rate is limited by the speed with which the processor can test and service a device.

 The processor is tied up in managing an I/O transfer; a number of instructions must be executed for each I/O transfer

3. **Direct Memory Access**: The data transfer between a fast storage media such as magnetic disk and memory unit is limited by the speed of the CPU. Thus we can allow the peripherals directly communicate with each other using the memory buses, removing the intervention of the CPU. This type of data transfer technique is known as DMA or direct memory access. During DMA the CPU is idle and it has no control over the memory buses. The DMA controller takes over the buses to manage the transfer directly between the I/O devices and the memory unit.

Bus Request : It is used by the DMA controller to request the CPU to relinquish the control of the buses.

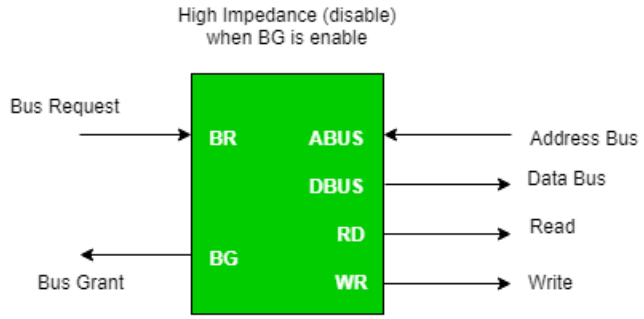


Figure - CPU Bus Signals for DMA Transfer

Bus Grant : It is activated by the CPU to Inform the external DMA controller that the buses are in high impedance state and the requesting DMA can take control of the buses. Once the DMA has taken the control of the buses it transfers the data. This transfer can take place in many ways.

Types of DMA transfer using DMA controller:

BurstTransfer : DMA returns the bus after complete data transfer. A register is used as a byte count, being decremented for each byte transfer, and upon the byte count reaching zero, the DMAC will release the bus. When the DMAC operates in burst mode, the CPU is halted for the duration of the data transfer. Steps involved are:

1. Bus grant request time.

2. Transfer the entire block of data at transfer rate of device because the device is usually slow than the speed at which the data can be transferred to CPU.

3. Release the control of the bus back to CPU So, total time taken to transfer the N bytes = Bus grant request time + (N) * (memory transfer rate) + Bus release control time.

Where,

X µsec =data transfer time or preparation time (words/block)

Y µsec =memory cycle time or cycle time or transfer time (words/block)

% CPU idle (Blocked)=(Y/X+Y)*100

% CPU Busy=(X/X+Y)*100

CyclicStealing : An alternative method in which DMA controller transfers one word at a time after which it must return the control of the buses to the CPU. The CPU delays its operation only for one memory cycle to allow the direct memory I/O transfer to "steal" one memory cycle. Steps Involved are:

4. Buffer the byte into the buffer

5. Inform the CPU that the device has 1 byte to transfer (i.e. bus grant request)

6. Transfer the byte (at system bus speed)

7. Release the control of the bus back to CPU.

Before moving on transfer next byte of data, device performs step 1 again so that bus isn't tied up and the transfer won't depend upon the transfer rate of device. So, for 1 byte of transfer of data, time taken by using cycle stealing mode (T). = time required for bus grant + 1 bus cycle to transfer data + time required to release the bus, it will be N x T

In cycle stealing mode we always follow pipelining concept that when one byte is getting transferred then Device is parallel preparing the next byte. "The fraction of CPU time to the data transfer time" if asked then cycle stealing mode is used.

Where,

X µsec =data transfer time or preparation time

(words/block)

Y µsec =memory cycle time or cycle time or transfer

time (words/block)

% CPU idle (Blocked) =(Y/X)*100

% CPU busy=(X/Y)*100

Interleaved mode: In this technique , the DMA controller takes over the system bus when the microprocessor is not using it.An alternate half cycle i.e. half cycle DMA + half cycle processor. ACCESSING I/O DEVICE

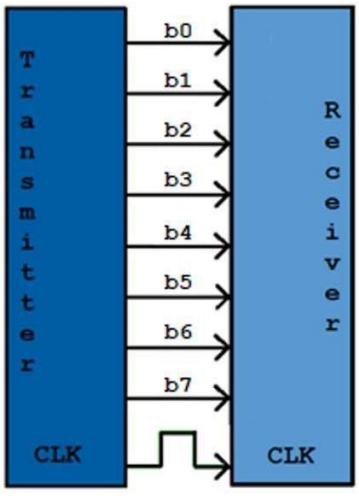
We can access the i/o device in two interfaces.

i. Serial interface

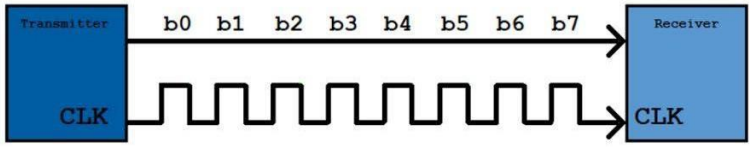
ii. Parallel interface

The main difference between the serial and parallel interfaces is how they transmit data. In serial interface the data is sent or received one bit at a time over a series of clock pulses.

In parallel mode the interface sends and receives 4 bits, 8 bits, or 16 bits of data at a time over multiple transmission lines. These two interface modes will be explained in further detail below.



8-bit Parallel Interface



8-bit Serial Interface

SERIAL INTERFACE

Serial interface send a byte by bit by bit in serial manner with defined clock pulse width (with parity).

Serial interfaces consist of 3 types each with their own pins:

1. I2C (Inter-Integrated Circuit): Serial Data In and Serial Clock

2. 3/4-wire SPI (Serial Peripheral Interface): Consists of Serial Data Out, Serial Data In, Serial Clock and an additional Chip Select pin for the 4-wire SPI

3. Serial synchronous control and data lines: Serial Data In, Register Select, Reset, and Serial Clock

Below is the serial interface connection example.

No.	Symbol	Description	Notes
1	SCL	Serial Clock	Output from master
2	CS	Chip Select, Low is active	Control Line
3	SDI/SDA	Serial Data In	Data lines
4	SDO	Serial Data Out	
5	A0	Register Select. 0: instruction, 1: data register	
6	RES	External Reset, Low is active	

Serial Interface Pros:

-Less data pins

-Cheaper

-Easy setup

PARALLEL INTERFACE

The parallel interface transmits 8-bits, or one byte, of data over multiple data bus lines over one clock pulse. This makes parallel transmission faster than serial but is typically more expensive and requires more data pins to beconnected. The parallel interface consists of 8 data pins and 3 control pins. The control pins are typically labeled: Register Select (RS), Enable (E), and Read/Write (R/W). Additional common parallel interface pins may include: Contrast adjust (V0), Chip Select (CS) and

Parallel interface consists of 2 standard types:

1. 8080 type: parallel 4-bit/8-bit data input with a write and a read line

2. 6800 type: parallel 4/8-bit data input with write, read and enable lines

Parallel Interface Pros

-Faster data transmission

-High performance

INTERCONNECTION STANDARDS

There are two interconnection standards

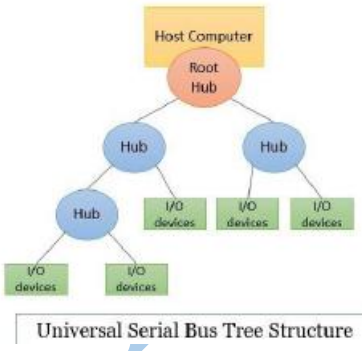
i. USB

ii. SATA (Serial ATA)

UNIVERSAL SERIAL BUS (USB)

The **universal serial bus (USB)** is a standard interface for connecting a wide range of devices to the computer such as keyboard, mouse, smartphones, speakers, cameras etc. The USB was introduced for commercial use in the year 1995 at that time it has a data transfer speed of 12 megabits/s.

With some improvement, a modified USB 2 was introduced which is also called a *highspeed USB* that transfers data at 480 megabits/s. With the evolution of I/O devices that require highspeed data transfer also leads to the development of USB 3 which is also referred to as *Superspeed USB* which transfers data at 5 gigabits/s. The recent version of USB can transfer data up to 20 gigabits/s.



Universal Serial Bus Tree Structure

Key Objectives of Universal Serial Bus

 The developed USB must be *simple* and a *low-cost* interconnection system that should be *easy* to use.

 The developed USB must be compatible with all new I/O devices, their bit rates, internet connections and audio, video application.

 The USB must support a *plug-and-play* mode of operation.

 The USB must support *low power* implementation.

 The USB must also provide support for *legacy hardware* and *software*.

USB Architecture

When multiple I/O devices are connected to the computer through USB they all are organized in a tree structure. Each I/O device makes a *point-to-point* connection and transfers data using the *serial transmission format* we have discussed serial transmission in our previous content 'interface circuitA tree structure has a **root, nodes** and **leaves.** The tree structure connecting I/O devices to the computer using USB has nodes which are also referred to as a **hub**. Hub is the intermediatory connecting point between the I/O devices and the computer. Every tree has a root here, it is referred to as the **root hub** which connects the entire tree to the hosting computer. The leaves of the tree here are nothing but the I/O devices such as a mouse, keyboard, camera, speaker.

The USB works on the principle of polling. In **polling**, the processor keeps on checking whether the I/O device is ready for data transfer or not. So, the devices do not have to inform the processor about any of their statuses. It is the processor's responsibility to keep a check. This makes the USB simple and low cost.

Whenever a new device is connected to the hub it is addressed as 0. Now at a regular interval the host computer polls all the hubs to get their status which lets the host know of I/O devices that are either detached from the system or are attached to the system.

When the host becomes aware of the new device it gets to know about the capabilities of the device by reading the information present in the special memory of the device's USB interface. So that the host can use the appropriate device driver to communicate with the device.

The host then assigns an address to this new device, this address is written to the register of the device interface register. With this mechanism, USB serves plug-and-play capability.

The **plug and play** feature let the host recognize the existence of the new I/O device automatically when the device is plugged in. The host software determines the capabilities of the I/O devices and if it has any special requirement.



The USB is **hot-pluggable** which means the I/O device can be attached or removed from the host system without performing any restart or shutdown. That

means your system can keep running while the I/O device is plugged or removed.

Types of USB Connectors

The USB has different types of ports and connectors. Usually, the upstream port and connector are always the USB type A the downstream port and connector differ depending on the type of device connected. We will discuss all types of the USB connector.

USB Type A: This is the standard connector that can be found at one end of the USB cable and is also known as upstream. It has a flat structure and has four connecting lines as you can see in the image below.



USB Type A

USB Type B: This is an older standard cable and was used to connect the peripheral devices also referred to as downstream. It is approximately a square as you can see in the image below. This is now been replaced by the newer versions.



USB Type B

Mini USB: This type of USB is compatible with mobile devices. This type of USB is now superseded your micro-USB still you will get it on some devices.

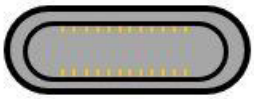


Mini USB

Micro USB: This type of USB is found on newer mobile devices. It has a compact 5 pin design.



USB Type C: This type of USB is used for transferring both data and power to the attached peripheral or I/O device. The USB C does not have a fixed orientation as it is reversible i.e. you can plug it upside down or in reverse.



USB Type C

USB 3.0 Micro B: This USB is a superspeed USB. This USB is used for a device that requires high-speed data transfer. You can find this kind of USB on portable hard drives



USB 3.0 Micro B

Electrical Characteristics of USB

The standard USB has four lines of connection among which two carry power (one carry +5 V and one is for Ground). The other two lines of connection are for data transfer. USB also supply power to connected I/O device that requires very low power.

Transferring of data over USB can be divided into two categories i.e., transferring data at low speed and transferring data at high speed.

The low-speed transmission uses **single-ended signalling** where varying high voltage is transmitted over one of the two data lines to represent the signal bit 0 or 1. The other data line is connected to the reference voltage i.e., ground. The single-ended signalling is prone to noise.

The high-speed data transmission uses the approach **differential signalling**. Here, the signal is transmitted over the two data lines that are twisted together. Here both the data lines are involved in carrying the signal no ground wire is required. The differential signalling is not prone to noise and uses low voltages as compared to single-ended transmission.

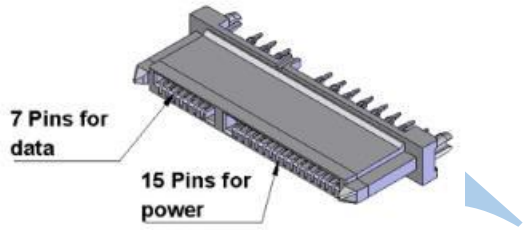
SATA

SATA stands for **Serial Advanced Technology Attachment or Serial ATA**.

SATA is an **interface that connects various storage devices such as hard disks, optical drives, SSD's, etc to the motherboard**. SATA was introduced in the year 2000 to replace the long-standing PATA (Parallel ATA) interface. We all know, in serial mode, data is transferred bit by bit and in parallel, there are several streams that carry the data. Despite knowing this fact, there is a drawback in PATA. PATA is highly susceptible to outside interferences and hence allows

SATA to operate at high speeds than PATA. SATA cables are thinner, more flexible and compact as compared to the PATA cables.

There were several industry groups that began their development in SATA late in the 2000s. It was only in the year 2003 that SATA-IO (SATA International Organization) was formed and it laid out the first SATA specifications.



A SATA controller is a device that is used to connect the computer's motherboard to the storage drives.

SATA operates on two modes –

1. **IDE mode:** IDE stands for Integrated Drive Electronics. This is a mode which is used to provide backward compatibility with older hardware, which runs on PATA, at the cost of low performance.

2. **AHCI mode:** AHCI is abbreviation for Advanced Host Controller Interface. AHCI is a high-performance mode that also provides support for hot-swapping.

Characteristics of SATA

 Low Voltage Requirement: SATA operates on 500mV (0.5V) peak-to-peak signaling. This help in promoting a much low interference and crosstalk between conductors.

 Simplified construction: PATA cables had 40-pin/80-wire ribbon cable. This was complex in construction. In comparison, SATA had a single 7 pin data cable and a 15 pin power cable. This cable resulted in a higher signaling rate, which translates to faster throughput of data.

 Differential Signaling: SATA uses differential signaling. Differential signaling is a technology which uses two adjacent wires to simultaneously the in-phase and out-of-phase signals. Thus, it is possible to transfer high-speed data with low operating voltage and low power consumption by detecting the phase difference between the two signals at the receiver's end.

 High data transfer rate: SATA has a high data transfer rate of 150/300/600 MBs/second. This capability of SATA allows for faster program loading, better picture loading and fast document loading.

Advantages of SATA

 Faster data transfer rate as compared to PATA.

 SATA cable can be of length upto 1 meter, whereas PATA cable can only have length of maximum 18 inches.

 SATA cables are smaller in size.

 Since, they are smaller in size, they take up less space inside the computer and increase the internal air flow. Increased air flow can decrease heat build-up and therefore increases the overall life of computer.

 Most modern computer motherboards today have SATA ports more than PATA ports.

 Low power consumption (0.5V).

Disadvantages of SATA

 Special device drivers are required sometimes to recognize and use the drive. However, a SATA hard drive can behave as a PATA drive. This eliminates the need for a specific driver to be installed.

 SATA cable supports only one hard drive to connect at a time, whereas PATA cable allows up to two PATA drives per cable.

 SATA is costlier as compared to PATA.

SATA standards and revisions

The nonprofit SATA-IO industry consortium authors the technical specifications governing Serial ATA device interfaces. The consortium revises SATA standards to reflect increased data transfer rates. These revisions include the following changes:

 SATA Revision 1. These devices were widely used in personal desktop and office computers, configured from PATA drives daisy chained together in a primary/secondary configuration. SATA Revision 1 devices reached a top transfer rate of 1.5 Gbps.

APEC

 SATA Revision 2. These devices doubled the transfer speed to 3.2 Gbps with the inclusion of port multipliers, port selectors and improved queue depth.

 SATA Revision 3. These interfaces supported drive transfer rates up to 6 Gbps. Revision 3 drives are backward-compatible with SATA Revision 1 and Revision 2 devices, though with lower transfer speeds.

 SATA Revision 3.1. This intermediate revision added final design requirements for SATA Universal Storage Module for consumer-based portable storage applications.

 SATA Revision 3.2. This update added the SATA Express specification. It supports the simultaneous use of SATA ports and PCI Express (PCIe) lanes.

 SATA Revision 3.3. This revision addressed the use of shingled magnetic recording

 SATA Revision 3.5. This change promoted greater integration and interoperability with PCIe flash and other I/O protocols.

COMPARISON

	USB 2.0	1394	Serial
Raw Interface speed	*480 Mbps*	*400 Mbps*	*1.5 Gbps (1500 Mbps)*
Benchmark comparison 64K read	*31.6 MB/sec*	*34.8 MB/sec*	*56.4 MB/sec*
Benchmark comparison 64K write	*26.5 MB/sec*	*26.7 MB/sec*	*54.2 MB/sec*
Burst Transfer Rate	*33.5 MB/sec*	*36.2 MB/sec*	*111.3 MB/sec*