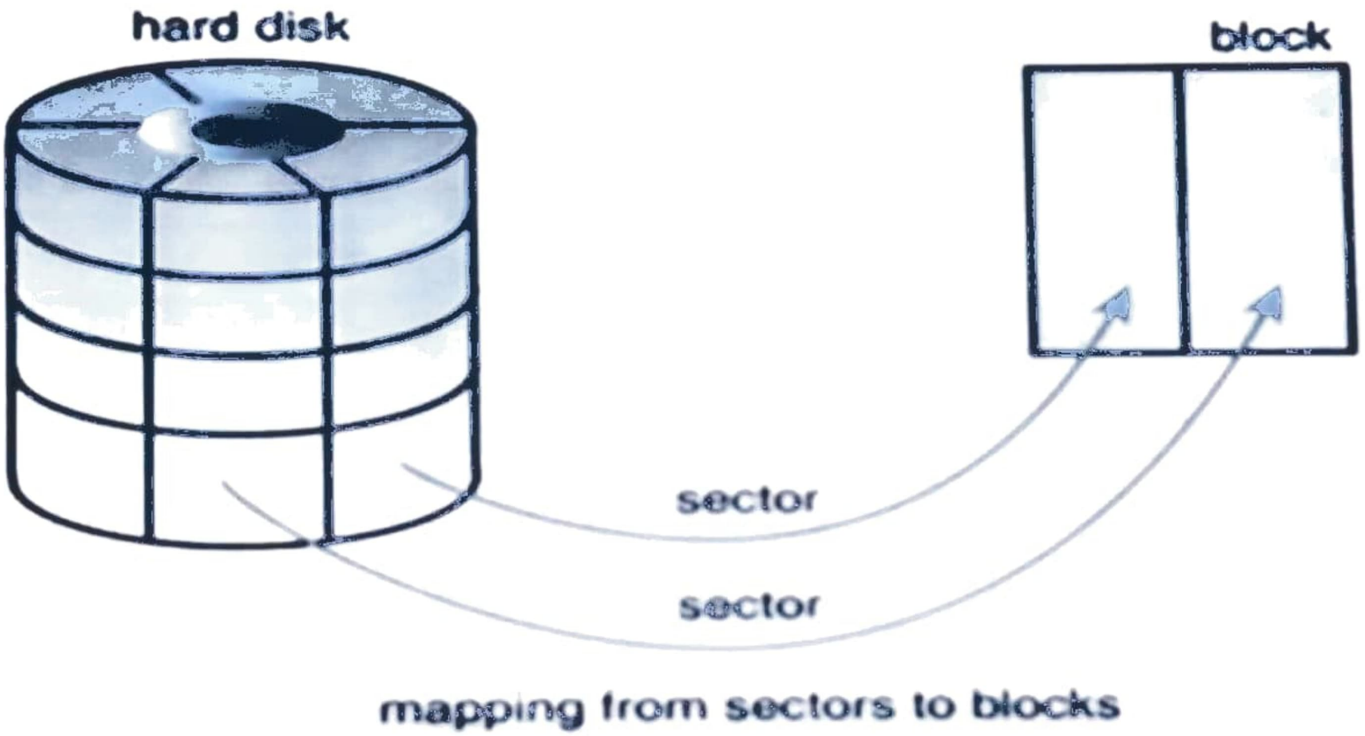
I/O Hardware

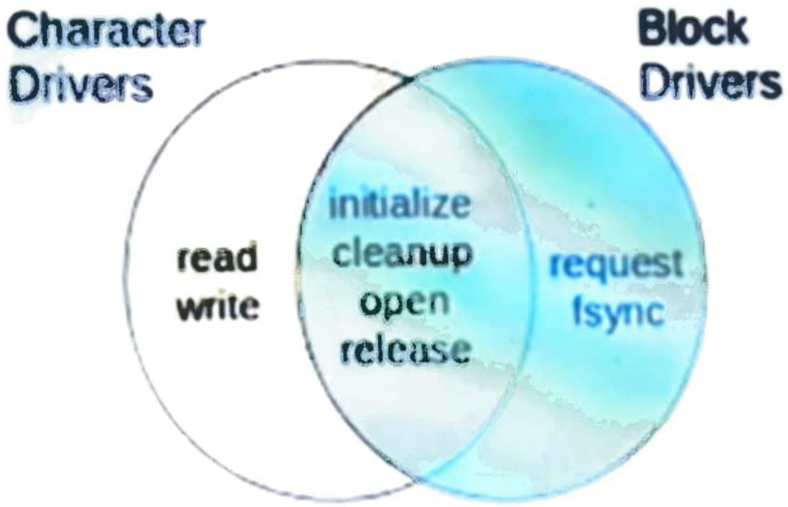
- One of the important jobs of an Operating System is to manage various I/O devices including mouse, keyboards, touch pad, disk drives, display adapters, USB devices, Bit-mapped screen, LED, Analog-to-digital converter, On off switch, network connections, audio I/O, printers etc.
- An I/O system is required to take an application I/O request and send it to the physical device, then take whatever response comes back from the device and send it to the application. I/O devices can be divided into two categories
- **Block devices** – A block device is one with which the driver communicates by sending entire blocks of data. For example, Hard disks, USB cameras, Disk-On-Key etc.
- **Character devices** – A character device is one with which the driver communicates by sending and receiving single characters (bytes, octets). For example, serial ports, parallel ports, sounds cards etc

Block devices A block device is one with which the driver communicates by sending entire blocks of data. For example, Hard disks, USB cameras, Disk-On-Key etc.



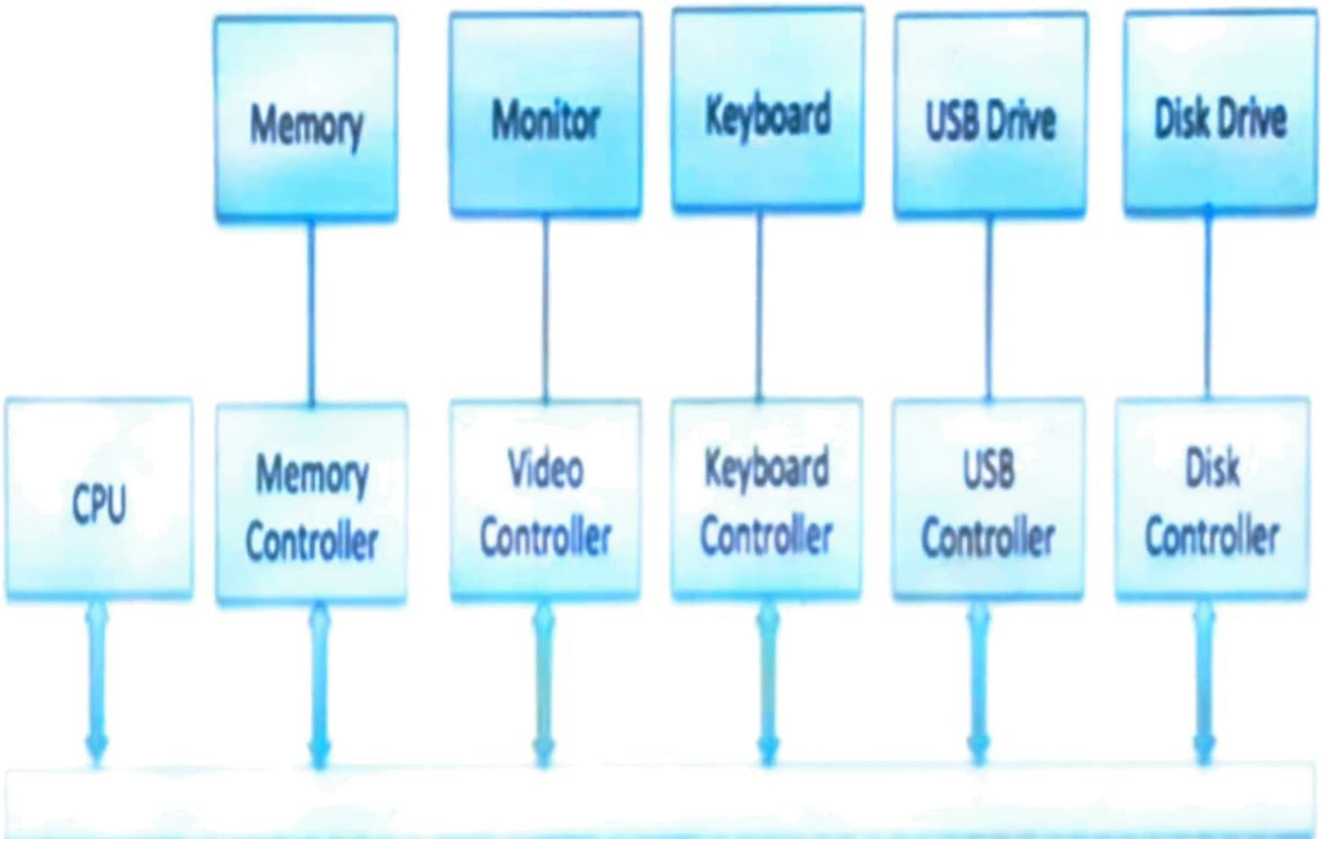
mapping from sectors to blocks

Character devices – A character device is one with which the driver communicates by sending and receiving single characters (bytes, octets). For example, serial ports, parallel ports, sounds cards etc



Device Controllers

- Device drivers are software modules that can be plugged into an OS to handle a particular device. Operating System takes help from device drivers to handle all I/O devices.
- The Device Controller works like an interface between a device and a device driver. I/O units (Keyboard, mouse, printer, etc.) typically consist of a mechanical component and an electronic component where electronic component is called the device controller.
- There is always a device controller and a device driver for each device to communicate with the Operating Systems. A device controller may be able to handle multiple devices. As an interface its main task is to convert serial bit stream to block of bytes, perform error correction as necessary.
- Any device connected to the computer is connected by a plug and socket, and the socket is connected to a device controller. Following is a model for connecting the CPU, memory, controllers, and I/O devices where CPU and device controllers all use a common bus for communication.


Memory
Monitor
Keyboard
USB Drive
Disk Drive
CPU
Memory Controller
Video Controller
Keyboard Controller
USB Controller
Disk Controller

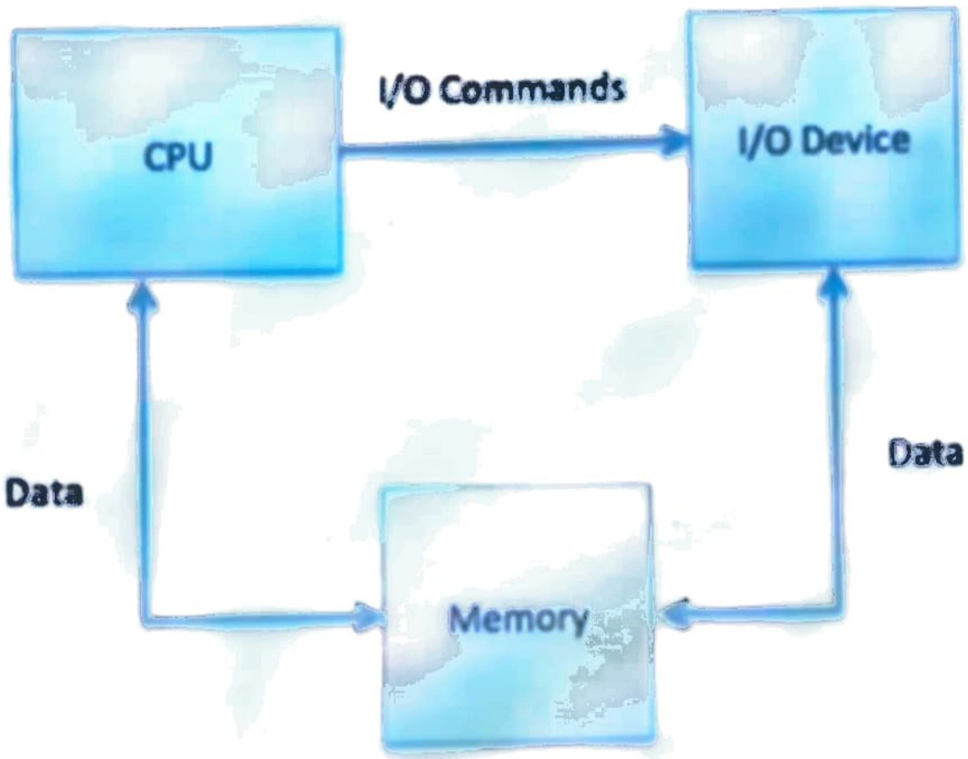
Communication to I/O Devices

- The CPU must have a way to pass information to and from an I/O device. There are three approaches available to communicate with the CPU and Device.

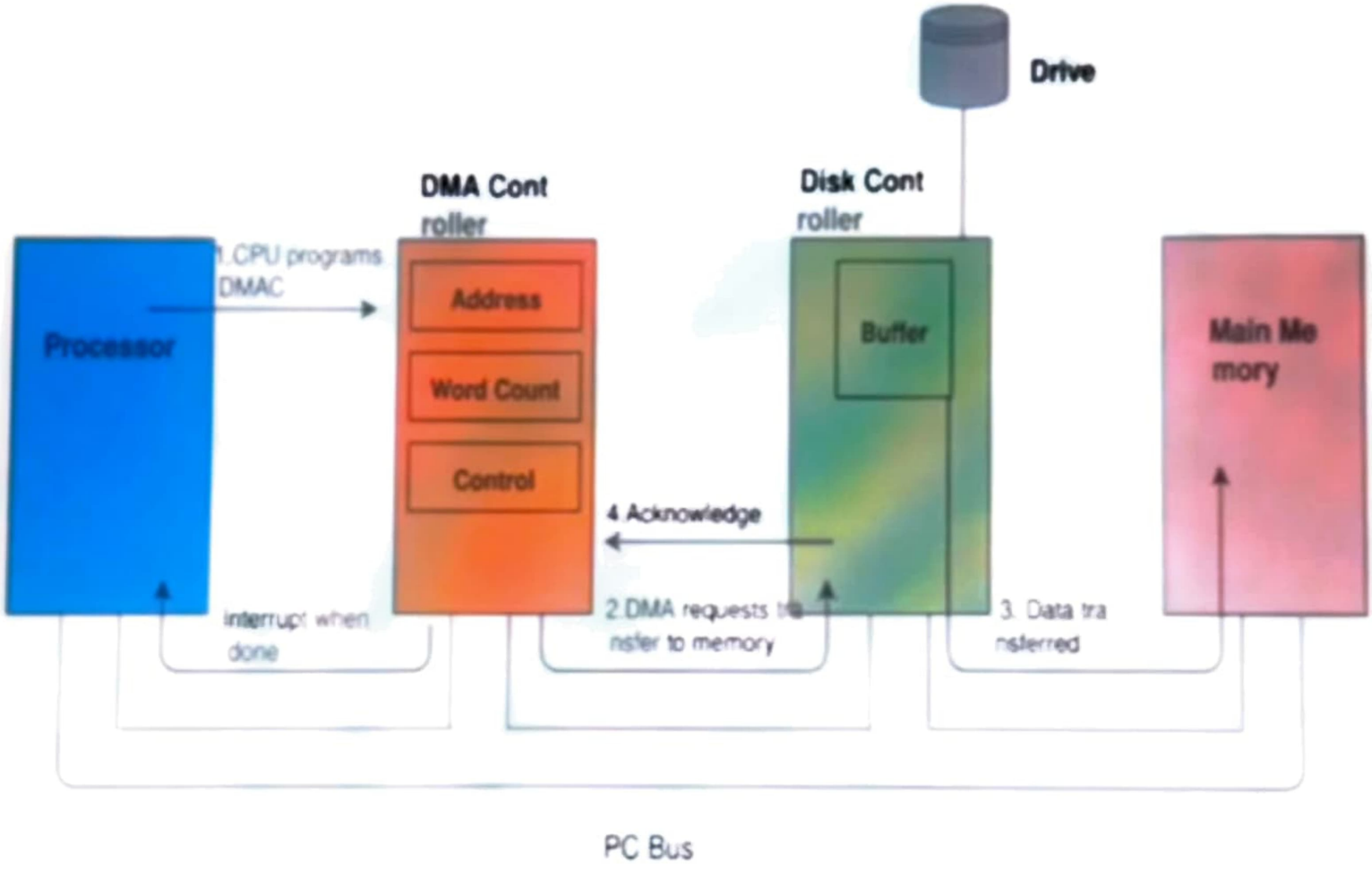
- Special Instruction I/O
- Memory-mapped I/O
- Direct memory access (DMA)

- Special Instruction I/O
- This uses CPU instructions that are specifically made for controlling I/O devices. These instructions typically allow data to be sent to an I/O device or read from an I/O device.

- Memory-mapped I/O
- When using memory-mapped I/O, the same address space is shared by memory and I/O devices. The device is connected directly to certain main memory locations so that I/O device can transfer block of data to from memory without going through CPU.
- While using memory mapped IO, OS allocates buffer in memory and informs I/O device to use that buffer to send data to the CPU. I/O device operates asynchronously with CPU, interrupts CPU when finished.
- The advantage to this method is that every instruction which can access memory can be used to manipulate an I/O device. Memory mapped IO is used for most high-speed I/O devices like disks, communication interfaces.


CPU
I/O Commands
I/O Device
Data
Data
Memory

Direct Memory Access (DMA)



- Slow devices like keyboards will generate an interrupt to the main CPU after each byte is transferred. If a fast device such as a disk generated an interrupt for each byte, the operating system would spend most of its time handling these interrupts. So a typical computer uses direct memory access (DMA) hardware to reduce this overhead.
- Direct Memory Access (DMA) means CPU grants I/O module authority to read from or write to memory without involvement. DMA module itself controls exchange of data between main memory and the I/O device. CPU is only involved at the beginning and end of the transfer and interrupted only after entire block has been transferred.
- Direct Memory Access needs a special hardware called DMA controller (DMAC) that manages the data transfers and arbitrates access to the system bus. The controllers are programmed with source and destination pointers (where to read/write the data), counters to track the number of transferred bytes, and settings, which includes I/O and memory types, interrupts and states for the CPU cycles.

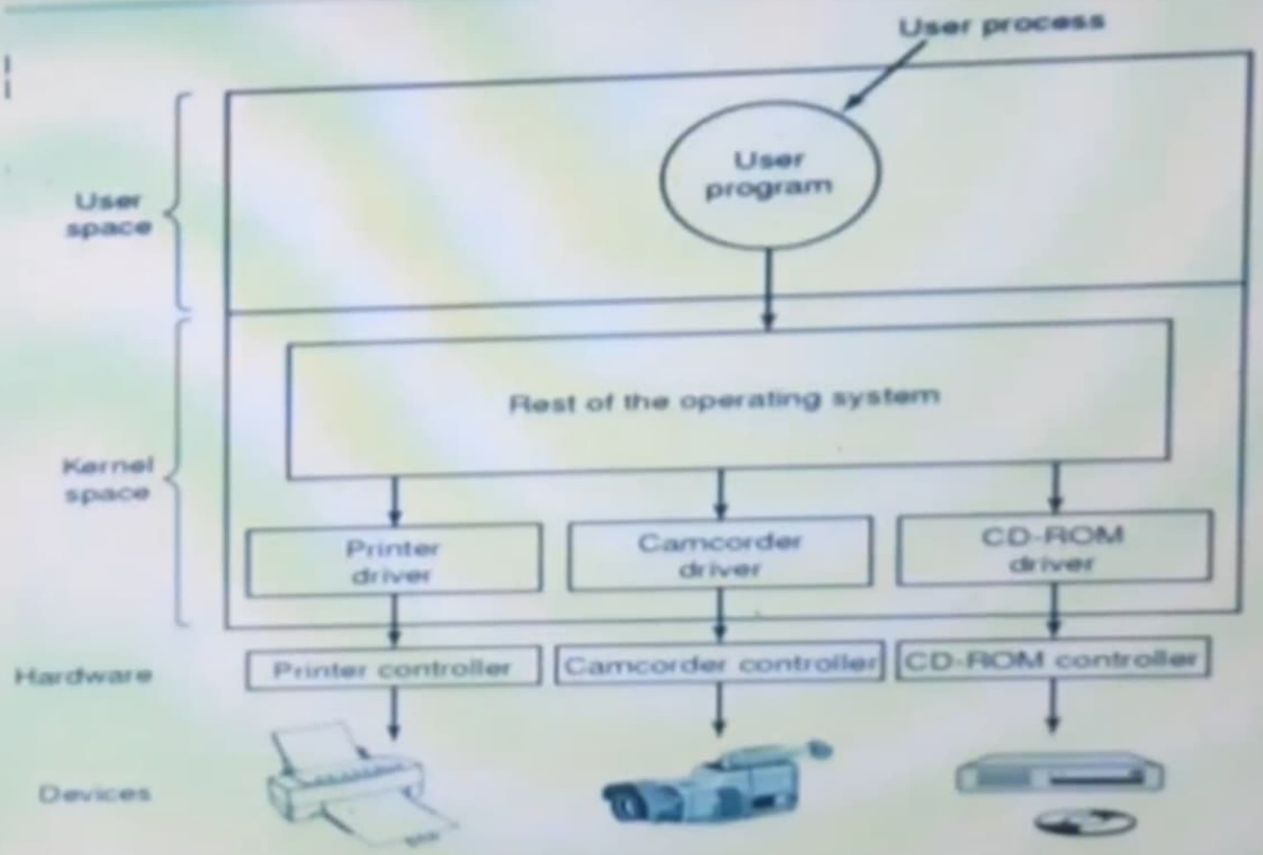
I/O Software

- I/O software is often organized in the following layers −

- User Level Libraries − This provides simple interface to the user program to perform input and output. For example, stdio is a library provided by C and C++ programming languages.

- Kernel Level Modules − This provides device driver to interact with the device controller and device independent I/O modules used by the device drivers.

- Hardware − This layer includes actual hardware and hardware controller which interact with the device drivers and makes hardware alive.

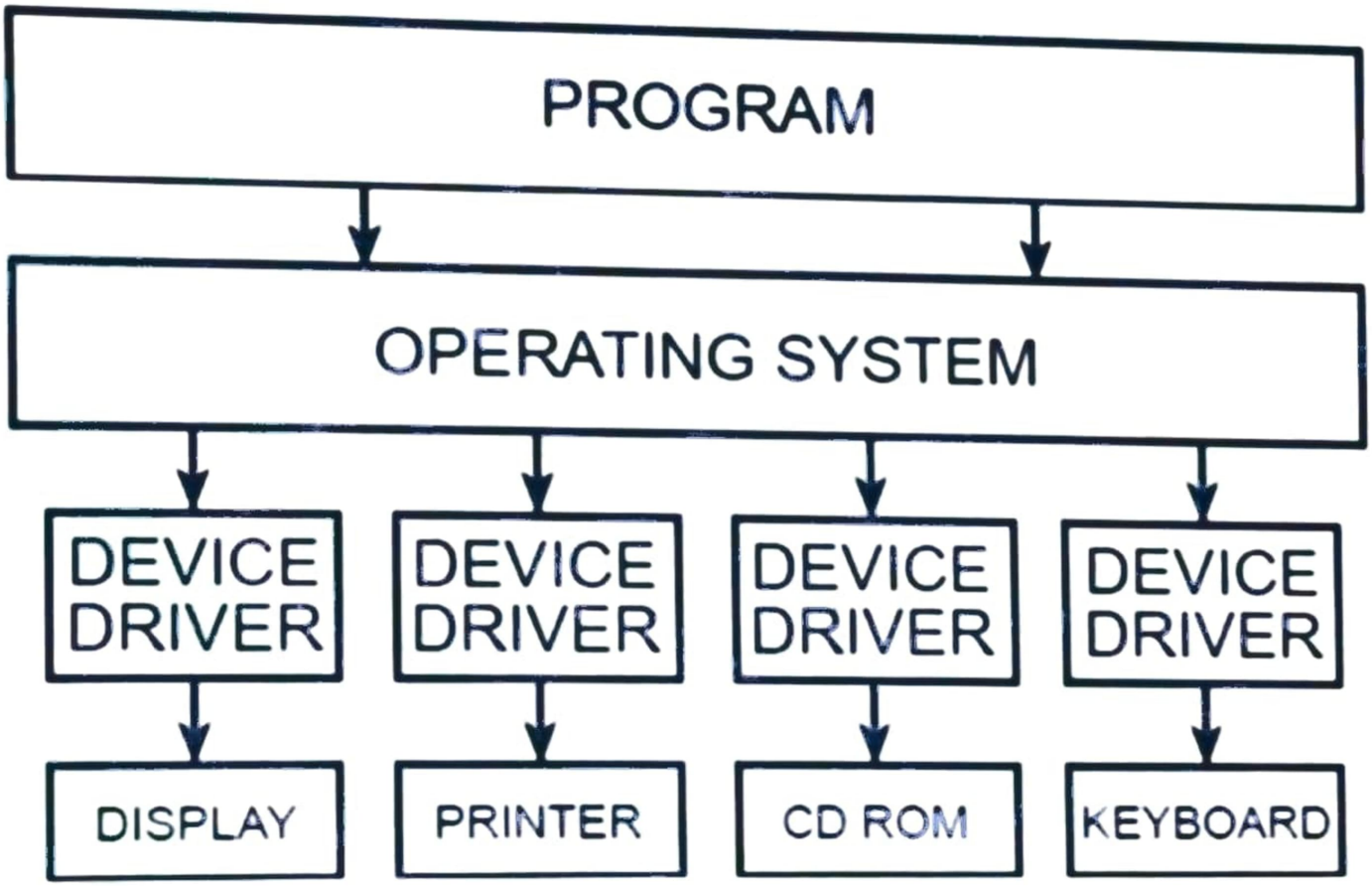

User process
User space
User program
Kernel space
Rest of the operating system
Printer driver
Camcorder driver
CD-ROM driver
Hardware
Printer controller
Camcorder controller
CD-ROM controller
Devices

Device Drivers

- Device drivers are software modules that can be plugged into an OS to handle a particular device. Operating System takes help from device drivers to handle all I/O devices.

- Device drivers encapsulate device-dependent code and implement a standard interface in such a way that code contains device-specific register reads/writes.

- Device driver, is generally written by the device's manufacturer and delivered along with the device on a CD-ROM.


PROGRAM
OPERATING SYSTEM
DEVICE DRIVER
DEVICE DRIVER
DEVICE DRIVER
DEVICE DRIVER
DISPLAY
PRINTER
CD ROM
KEYBOARD

- A device driver performs the following jobs –

- To accept request from the device independent software above to it.
- Interact with the device controller to take and give I/O and perform required error handling
- Making sure that the request is executed successfully

- Suppose a request comes to read a block N. If the driver is idle at the time a request arrives, it starts carrying out the request immediately. Otherwise, if the driver is already busy with some other request, it places the new request in the queue of pending requests.

Device-Independent I/O Software

- A key concept in the design of I/O software is that it should be device independent where it should be possible to write programs that can access any I/O device without having to specify the device in advance.

- For example, a program that reads a file as input should be able to read a file on a floppy disk, on a hard disk, or on a CD-ROM, without having to modify the program for each different device.

- The basic function of the device-independent software is to perform the I/O functions that are common to all devices and to provide a uniform interface to the user-level software.

- Though it is difficult to write completely device independent software but we can write some modules which are common among all the devices.

- **Following is a list of functions of device-independent I/O Software –**
- **Uniform interfacing for device drivers**
- **Device naming - Mnemonic names mapped to Major and Minor device numbers**
- **Device protection**
- **Providing a device-independent block size**
- **Buffering because data coming off a device cannot be stored in final destination.**
- **Storage allocation on block devices**
- **Allocation and releasing dedicated devices**
- **Error Reporting**

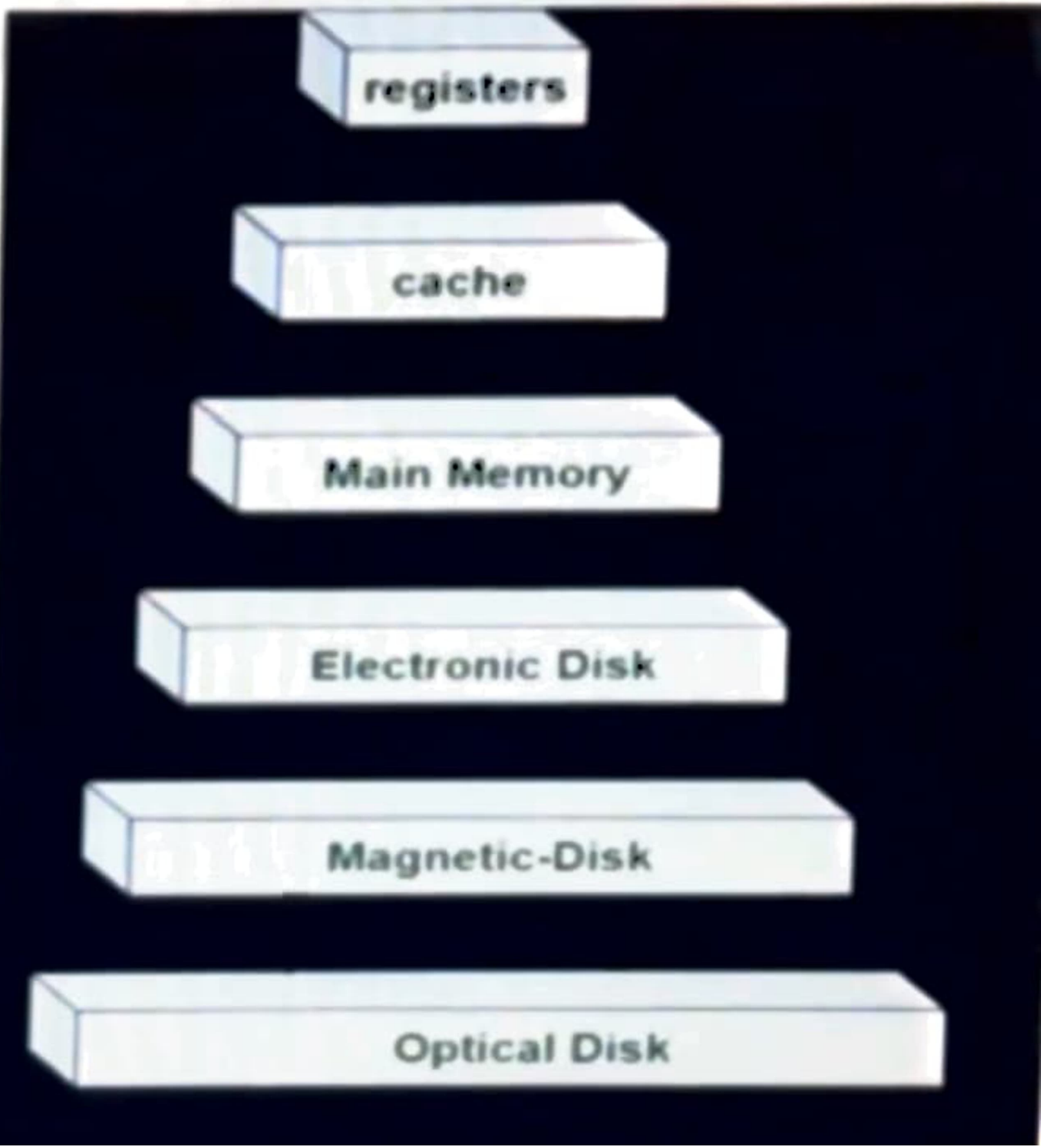
Secondary-Storage Structure

- It is used as an extension of main memory. Secondary storage devices can hold the data permanently.

- Storage devices consists of Registers, Cache, Main Memory, Electronic-Disk, Magnetic-Disk, Optical Disk, Magnetic-Tapes.

- Each storage system provides the basic system of storing a datum and of holding the datum until it is retrieved at a later time.

- All the storage devices differ in speed, cost, size and volatility. The most common Secondary-storage device is a Magnetic-disk, which provides storage for both programs and data.


registers
cache
Main Memory
Electronic Disk
Magnetic-Disk
Optical Disk

- In this hierarchy all the storage devices are arranged according to speed and cost.

- The higher levels are expensive, but they are fast. As we move down the hierarchy, the cost per bit generally decreases, where as the access time generally increases.

- The storage systems above the Electronic disk are Volatile, where as those below are Non-Volatile.

- **An Electronic disk can be either designed to be either Volatile or Non-Volatile.**

- **During normal operation, the electronic disk stores data in a large DRAM array, which is Volatile.**

- **But many electronic disk devices contain a hidden magnetic hard disk and a battery for backup power.**

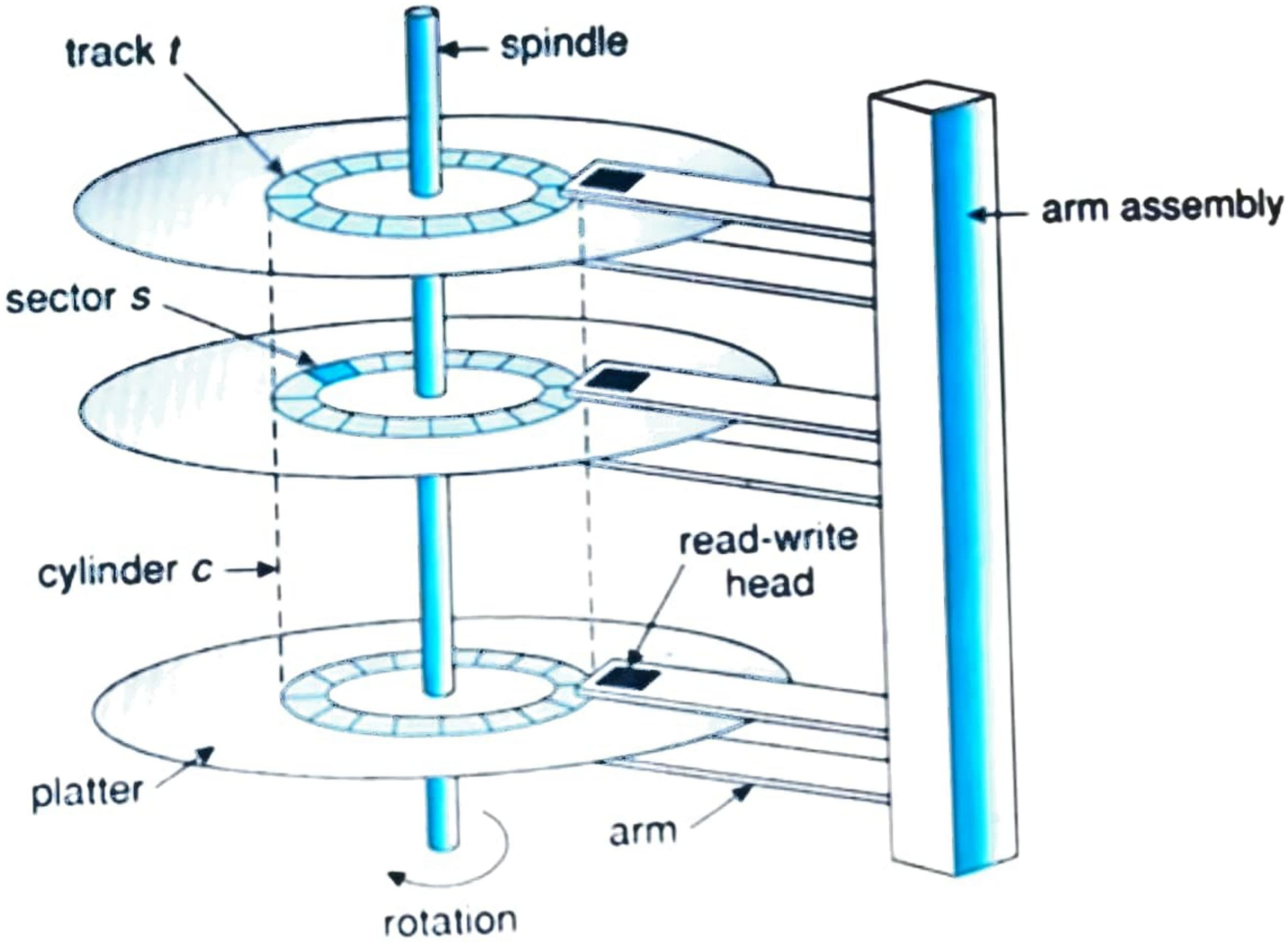
- **If external power is interrupted, the electronic disk controller copies the data from RAM to the magnetic disk.**

- **When external power is restored, the controller copies the data back into the RAM.**

Disk Structure

- Each modern disk contains concentric tracks and each track is divided into multiple sectors. The disks are usually arranged as a one dimensional array of blocks, where blocks are the smallest storage unit.

- Blocks can also be called as sectors. For each surface of the disk, there is a read/write desk available. The same tracks on all the surfaces is known as a cylinder.


track t
spindle
arm assembly
sector s
cylinder c
read-write
head
platter
arm
rotation

- **Disk Size =**

  Platters*Surface*Tracks*Sectors*Data

- **Let us say, Calculate for 8 platters**

  8*2*256*512*512

Disk Scheduling Algorithms

- Disk scheduling is done by operating systems to schedule I/O requests arriving for the disk. Disk scheduling is also known as I/O scheduling.

- Disk scheduling is important because:
- Multiple I/O requests may arrive by different processes and only one I/O request can be served at a time by the disk controller. Thus other I/O requests need to wait in the waiting queue and need to be scheduled.
- Two or more request may be far from each other so can result in greater disk arm movement.
- Hard drives are one of the slowest parts of the computer system and thus need to be accessed in an efficient manner.

- **Seek Time: Seek time is the time taken to locate the disk arm to a specified track where the data is to be read or write. So the disk scheduling algorithm that gives minimum average seek time is better.**

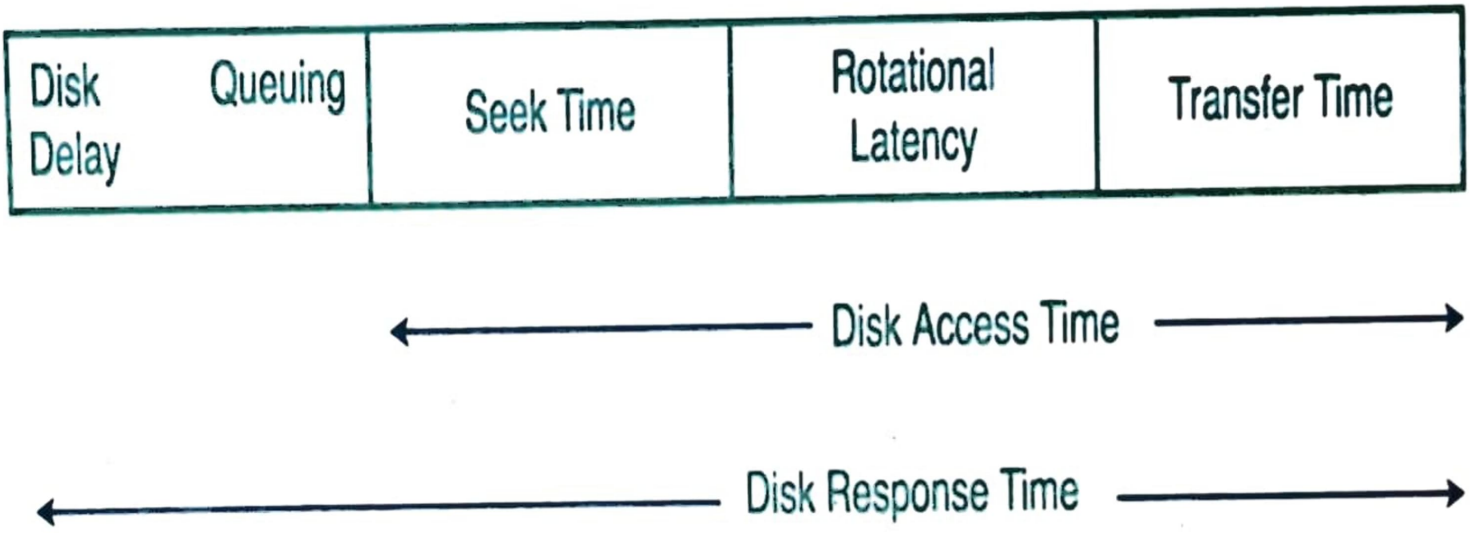
- **Rotational Latency: Rotational Latency is the time taken by the desired sector of disk to rotate into a position so that it can access the read/write heads. So the disk scheduling algorithm that gives minimum rotational latency is better.**

- **Transfer Time: Transfer time is the time to transfer the data. It depends on the rotating speed of the disk and number of bytes to be transferred.**

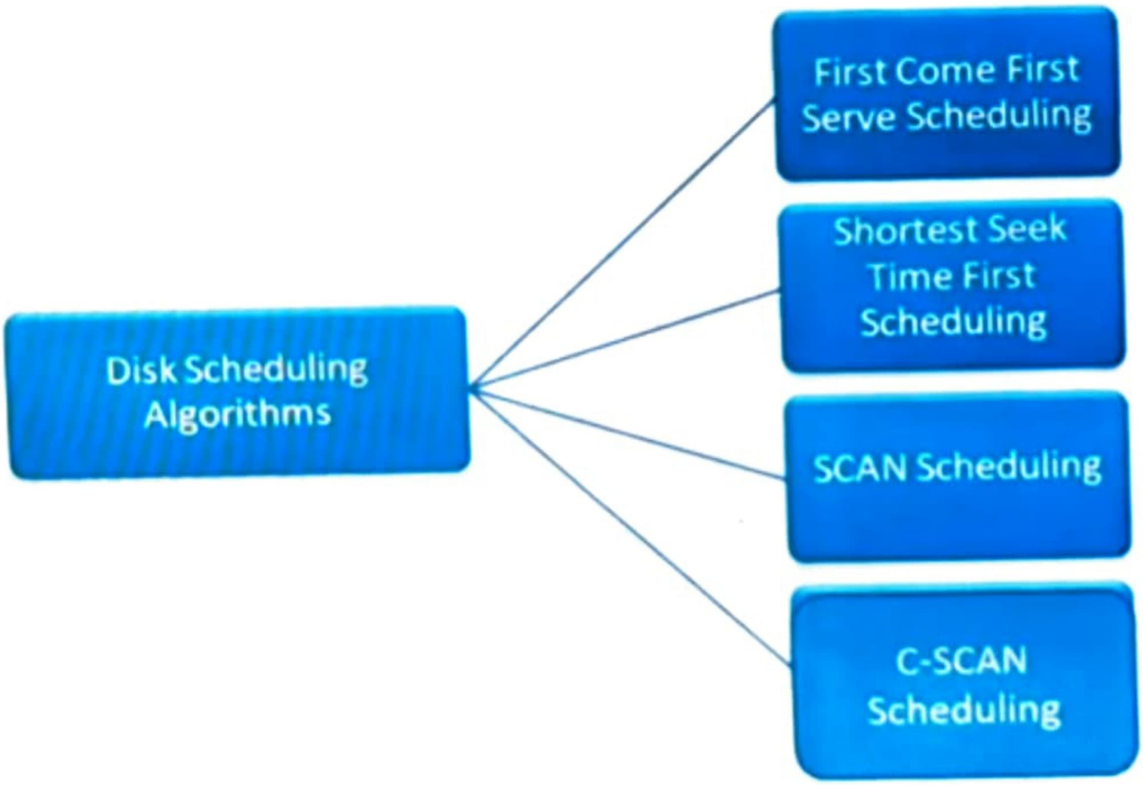
- **Disk Access Time: Disk Access Time is:**

Disk Access Time = Seek Time + Rotational Latency + Transfer Time

- **Disk Response Time: Response Time is the average of time spent by a request waiting to perform its I/O operation.**
- ***Average Response time* is the response time of the all requests.**


Disk Queuing Delay
Seek Time
Rotational Latency
Transfer Time
Disk Access Time
Disk Response Time

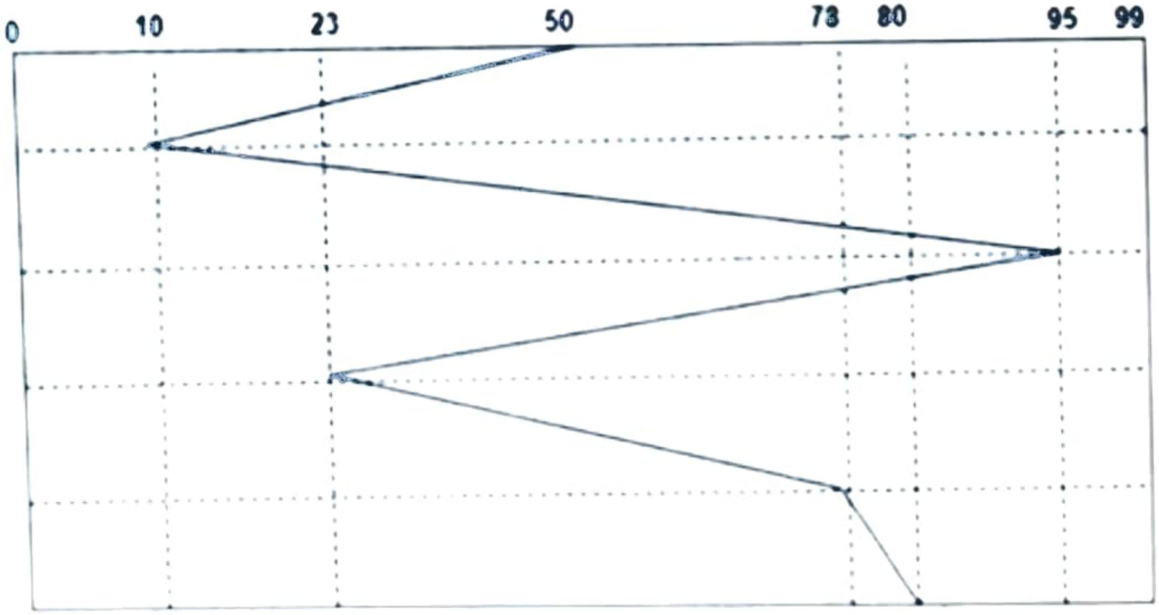
Types of Disk Scheduling Algorithms



First Come First Serve Scheduling (FCFS)

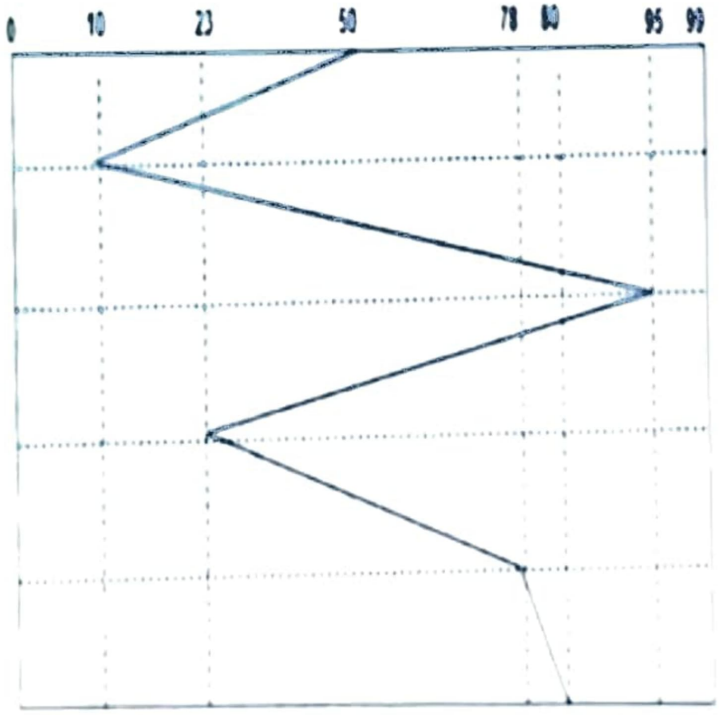
Request Queue = 10 95 23 78 80

Head starts at 50



Request Queue = 10 95 23 78 80

Head starts at 50



(50-10)+(95-10)+(95-23)+(78-23)+(80-78)

= 254

(50-10)+(95-10)+(95-23)+(80-23)

= 254

- In first come first served scheduling, the requests are serviced in their coming order.

- This algorithm is fair as it allows all requests a chance but it does not provide the fastest possible service.

- The requests are serviced in the order they appear i.e 10, 95, 23, 78, 80. The seek head is initially positioned at 50 and it starts from there.

- Advantages:

- **Every request gets a fair chance**
- **No indefinite postponement**

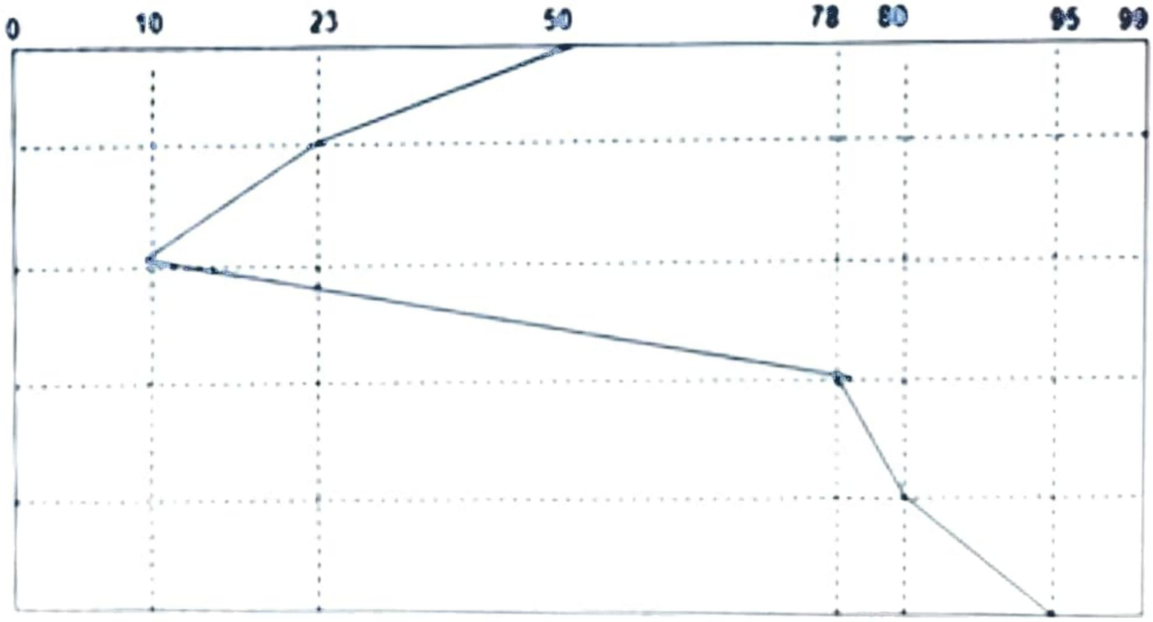
- Disadvantages:

- **Does not try to optimize seek time**
- **May not provide the best possible service**

Shortest Seek Time First Scheduling

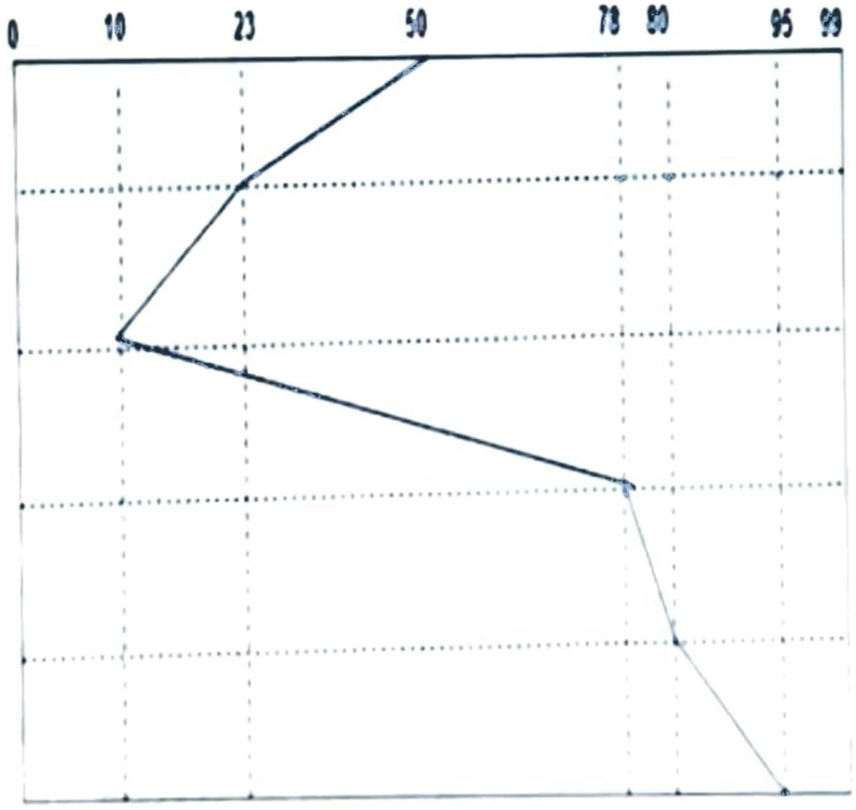
Request Queue = 10 95 23 78 80

Head starts at 50



Request Queue = 10 95 23 78 80

Head starts at 50



(50-23)+(23-10)+(78-10)+
(80-78)+(95-80)

= 125

(50-10)+(95-10)

= 125

- The requests that are closest to the current head are served first before moving away in shortest seek time first scheduling algorithm.

- A problem with SSTF algorithm is that it may cause starvation for some requests.

- The requests are serviced in the order 23, 10, 78, 80, 95. The seek head is initially positioned at 50 and it starts from there. 23 is closest to 50 so it is services first. Then 10 is closer to 23 than 78 so it is services next. After this 78, 80 and 95 are serviced.

- **Advantages:**

- **Average Response Time decreases**
- **Throughput increases**

- **Disadvantages:**

- **Overhead to calculate seek time in advance**
- **Can cause Starvation for a request if it has higher seek time as compared to incoming requests**
- **High variance of response time as SSTF favors only some requests**

Bad Blocks In Disk Management

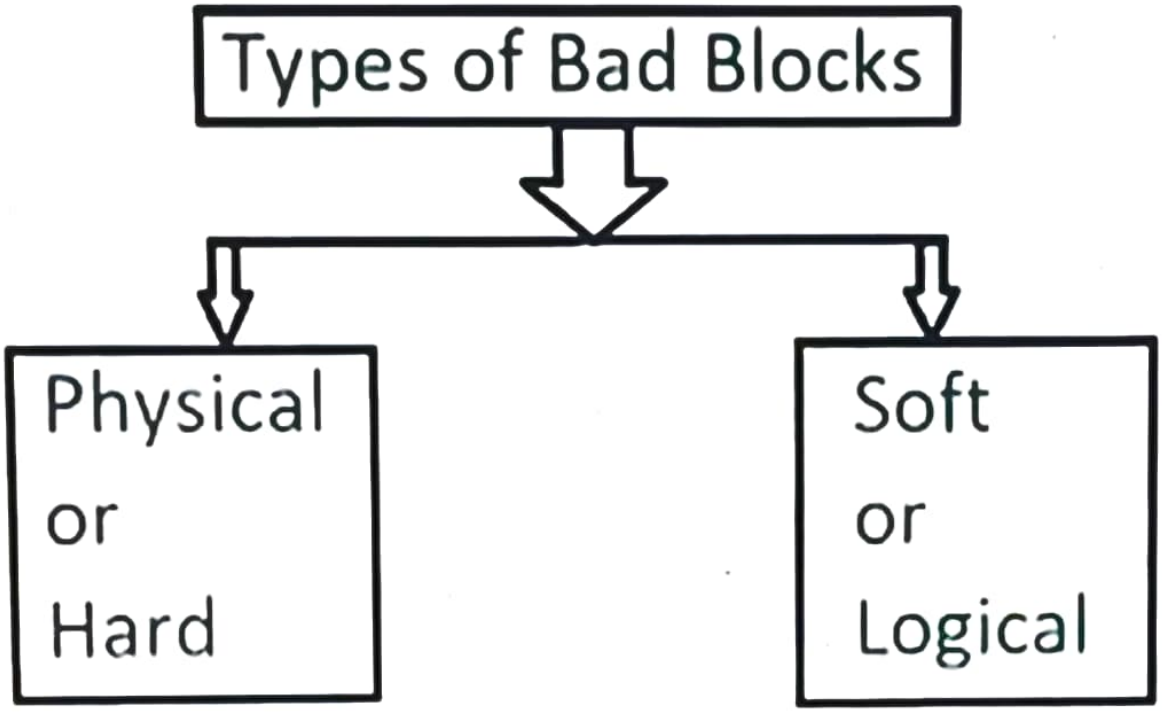
Badblocks

- **Badblocks is a Linux utility to assess unhealthy sectors on a disk power. It creates a list of those sectors which can be used with other applications like mkfs, so that they aren't used at some point and for that reason does not corrupt the data.**

- **Bad Block is an area of storing media that is no longer reliable for the storage of data because it is completely damaged or corrupted. We know disk have moving parts and have small tolerances, they are prone to failure**

- **Bad blocks are also referred to as bad sectors. There are two types of bad blocks: A physical, or hard, bad block comes from damage to the storage medium.**

- **There are mainly 2 types of Bad Blocks present.**


Types of Bad Blocks
Physical
or
Hard
Soft
or
Logical

Types of bad blocks

Physical bad block

- A physical, or hard, bad block comes from damage to the storage medium

- Bad blocks can happen when a location on the recording surface is defective or damaged. On NAND flash drives, blocks can become worn from use, making them unreliable or unusable after a certain number of write and erase cycles.

logical bad block

- **A soft, or logical, bad block occurs when the operating system (OS) is unable to read data from a sector.**

- **Examples of a soft bad block include when the cyclic redundancy check (CRC), or error correction code (ECC), for a particular storage block does not match the data read by the disk.**

Causes

- Storage drives can ship from the factory with defective blocks that originated in the manufacturing process. Before the device leaves the factory, these bad blocks are marked as defective and remapped to the drive's extra memory cells.

- A bad block can also result from physical damage to a device that makes it impossible for the OS to access data. On HDDs, mishaps, such as dropping a laptop, can cause the drive head to damage the platter. Dust and natural wear can also damage HDDs.

- Soft bad sectors are caused by software problems. For instance, if a computer unexpectedly shuts down, the hard drive could turn off in the middle of writing to a block. In this case, the block could contain data that doesn't match its CRC error correction code and would be identified as a bad sector.

- Damage to a solid-state drive (SSD) can occur when a memory transistor fails. Storage cells can also become unreliable over time, as the NAND flash substrate in a cell becomes unusable after a certain number of program-erase cycles.

- The erase process on an SSD requires sending a large electrical charge through the flash cell. Over time, this degrades the oxide layer that separates the floating gate transistors from the flash memory silicon substrate and the bit error rates increase. The drive's controller can use error detection and correction mechanisms to fix these errors.

- However, at some point, the errors can outstrip the controller's ability to correct them and the cell can become unreliable.

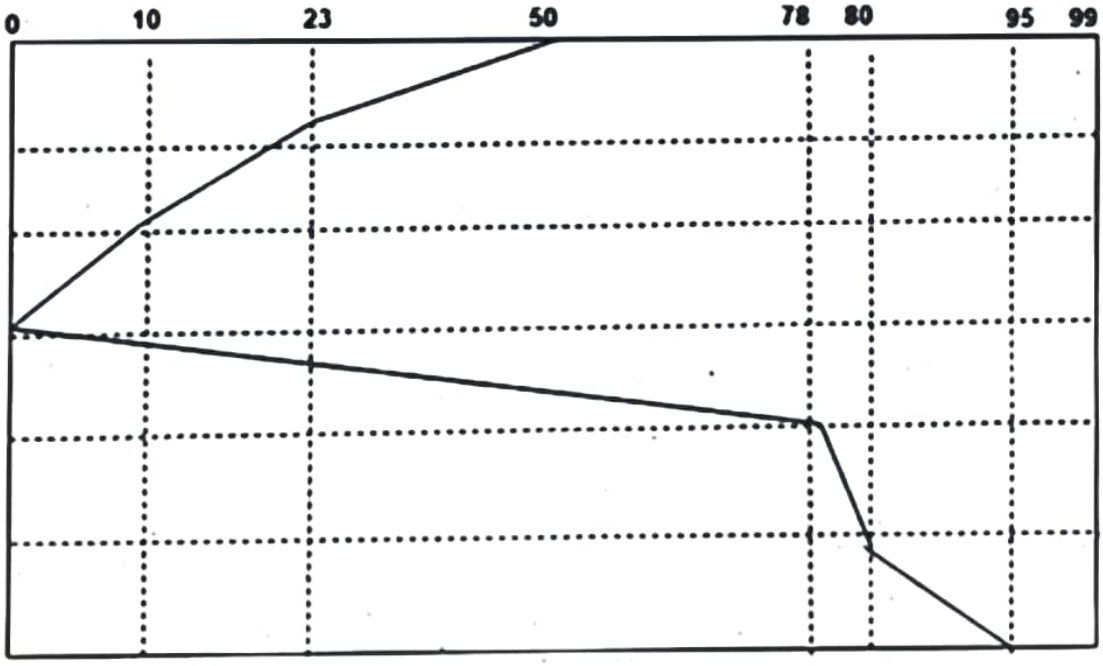
SCAN Scheduling Algorithm

Disk Scheduling Algorithms

SCAN Scheduling

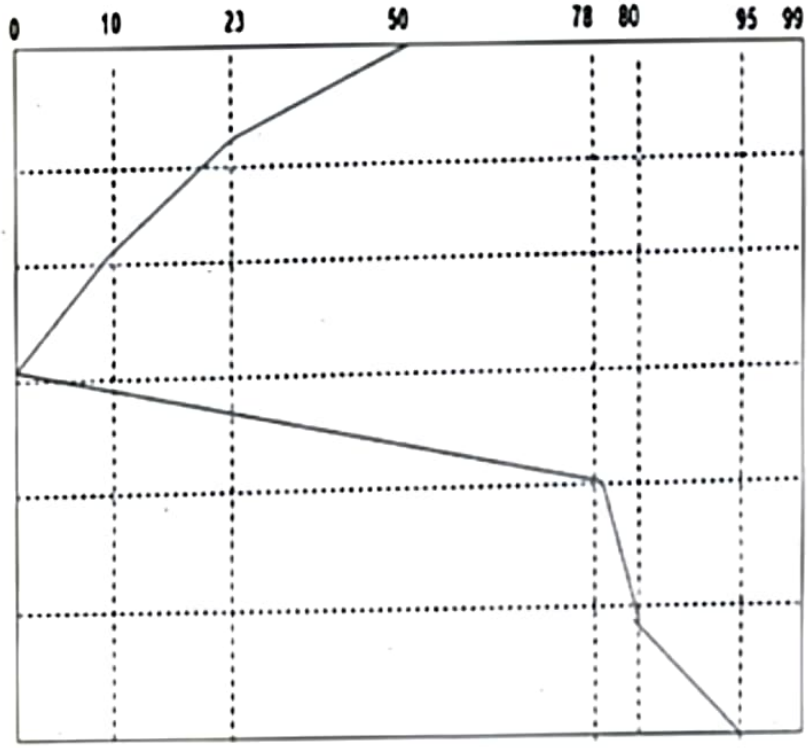
Request Queue = 10 95 23 78 80

Head starts at 50



Request Queue = 10 95 23 78 80

Head starts at 60



(50-23)+(23-10)+(10-0)
+(78-0)+(80-78)+(95-80)

= 145

(50-0)+(95-0)

= 145

- **In this scheduling algorithm, the head moves towards one direction while servicing all the requests in that direction until it reaches the end of the disk. After that it starts moving towards the other direction.**

- **In this way, the head continuously scans back and forth across the disk.**

- **The requests are serviced in the order 23, 10, 78, 80, 95. The head is initially at 50 and moves towards the left while servicing requests 23 and 10. When it reaches the end of the disk, it starts moving right and services 78, 80 and 95 as they occur.**

- **Advantages:**

- **High throughput**
- **Low variance of response time**
- **Average response time**

- **Disadvantages:**

- **Long waiting time for requests for locations just visited by disk arm**

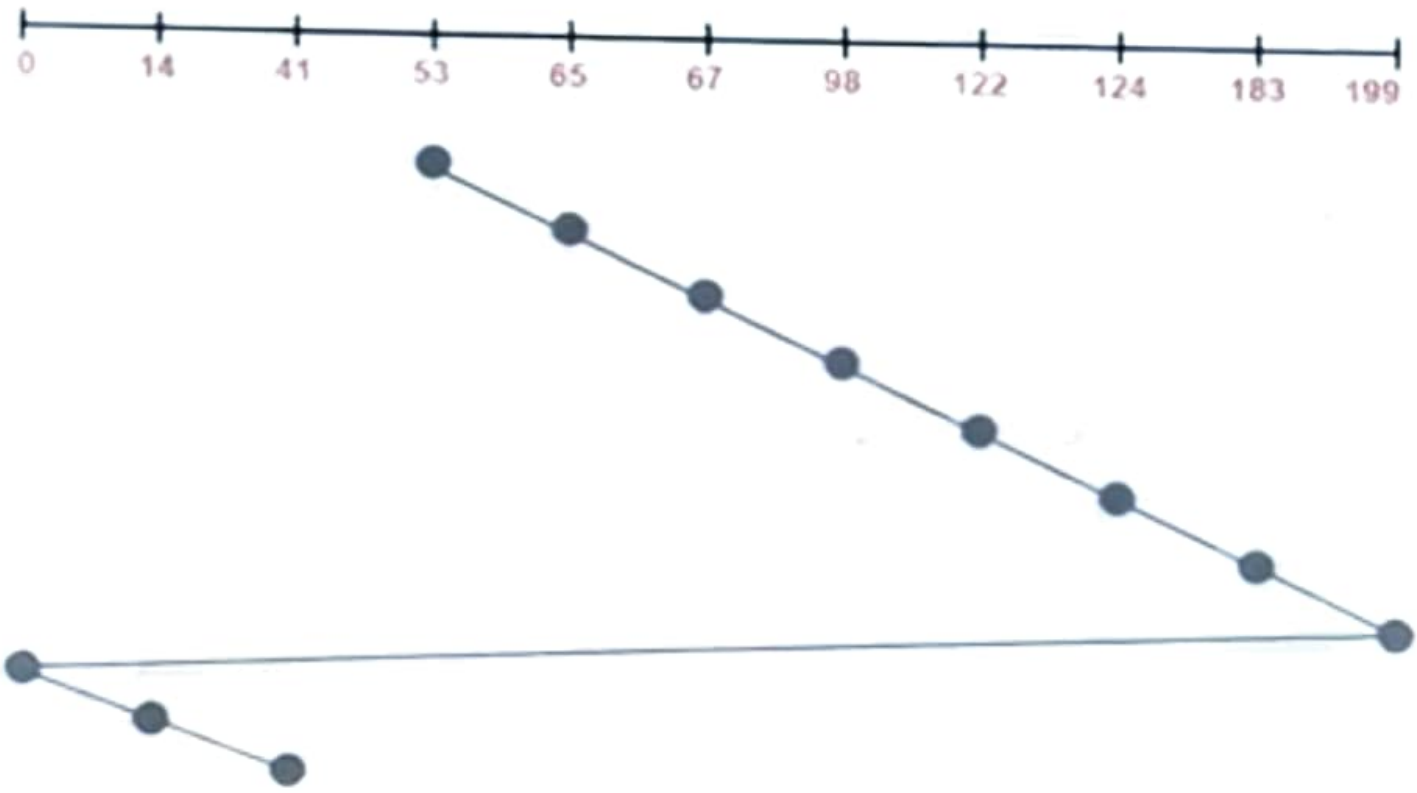
C - SCAN Algorithm

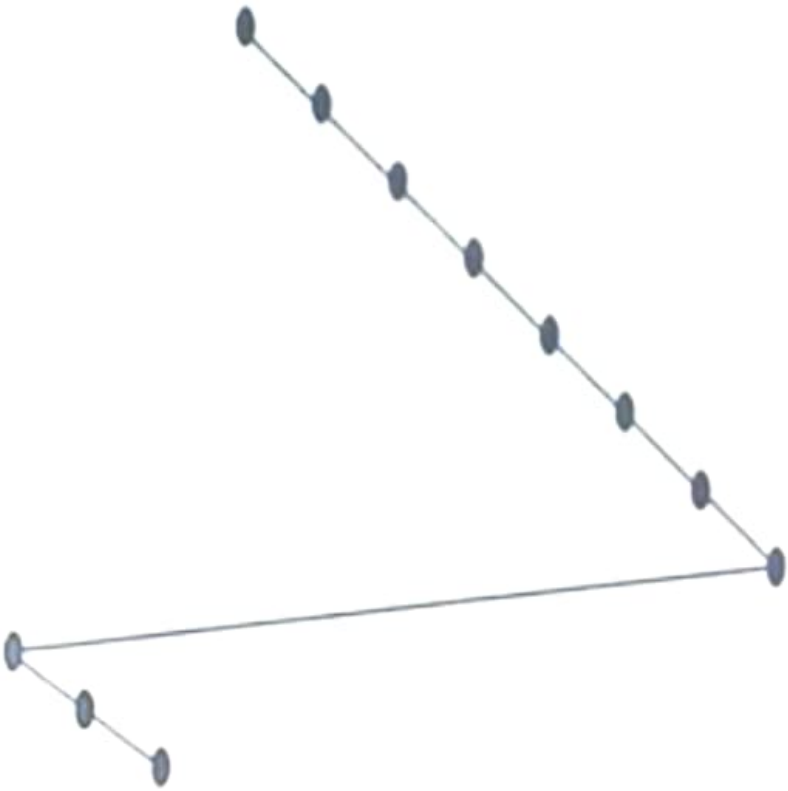
Disk Scheduling Algorithm

Circular – SCAN

Request Queue : 14,67,41,183,53,65,98,122,124,199 (Direction : To Large Value)

Head Starts at 50





(65-53)+(67-65)+(98-67)
+(122-98)+(124-122)+
(183-124)+(199-183)+
(199-0)+(14-0)+(41-14)

= 386

(199-53)+(199-0)+(41-0)

= 386

- In SCAN algorithm, the disk arm again scans the path that has been scanned, after reversing its direction.
- So, it may be possible that too many requests are waiting at the other end or there may be zero or few requests pending at the scanned area.
- These situations are avoided in *CSAN* algorithm in which the disk arm instead of reversing its direction goes to the other end of the disk and starts servicing the requests from there.
- So, the disk arm moves in a circular fashion and this algorithm is also similar to SCAN algorithm and hence it is known as C-SCAN (Circular SCAN).

Advantages

- Provides more uniform wait time compared to SCAN

Disk Reliability And Disk Formatting

Disk Reliability

- When using a disk drive, whether it is a flash disk or a rotating media disk, there may be repercussions due to a power failure during a disk write.

- Power failures during disk writes can result in data loss, lost clusters, and invalid directory entries, all of which can make a disk drive unusable until it is fixed up or reformatted.

- Reliability is the ability of the disk system to accommodate a single- or multi-disk failure and still remain available to the users. Performance is the ability of the disks to efficiently provide information to the users. Adding redundancy almost always increases the reliability of the disk system.

Disk Formatting

- Disk formatting is a process to configure the data-storage devices such as hard-drive, floppy disk and flash drive when we are going to use them for the very first time or we can say initial usage.

- Disk formatting is usually required when new operating system is going to be used by the user. It is also done when there is space issue and we require additional space for the storage of more data in the drives.

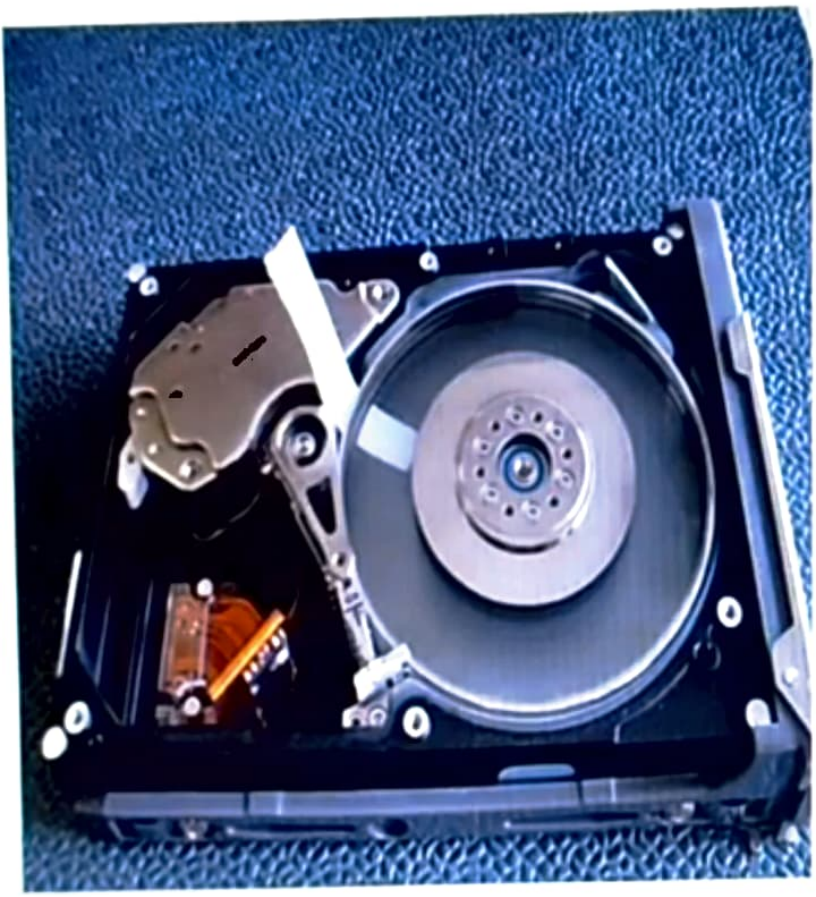
- When we format the disk then the existing files within the disk is also erased.

- We can perform disk formatting on both magnetic platter hard-drives and solid-state drives.

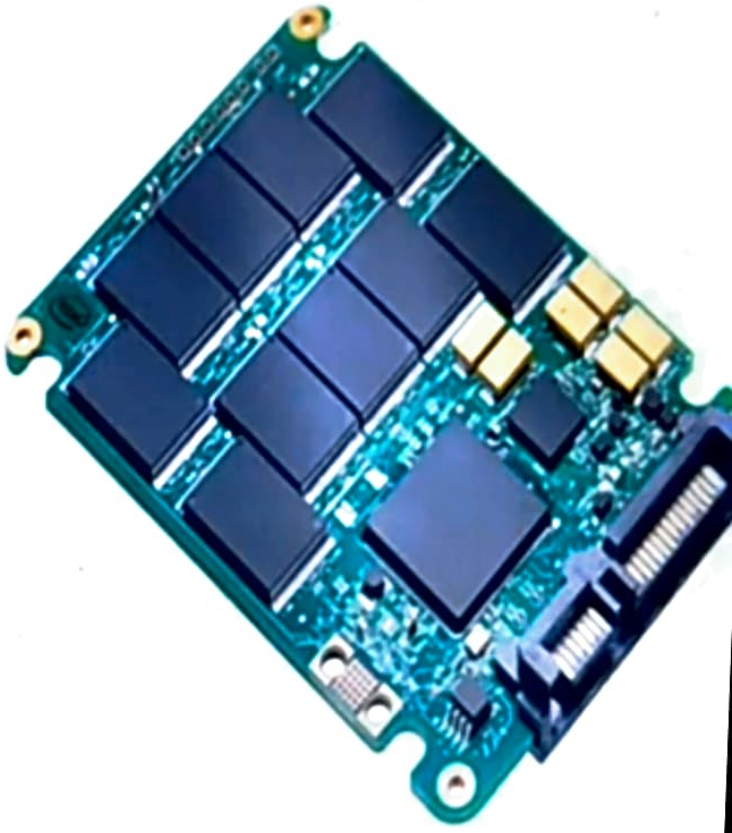
- When we are going to use hard-drive for initial use it is going to search for virus. It can scan for virus and repair the bad sectors within the drive. Disk formatting has also the capability to erase the bad applications and various sophisticated viruses.

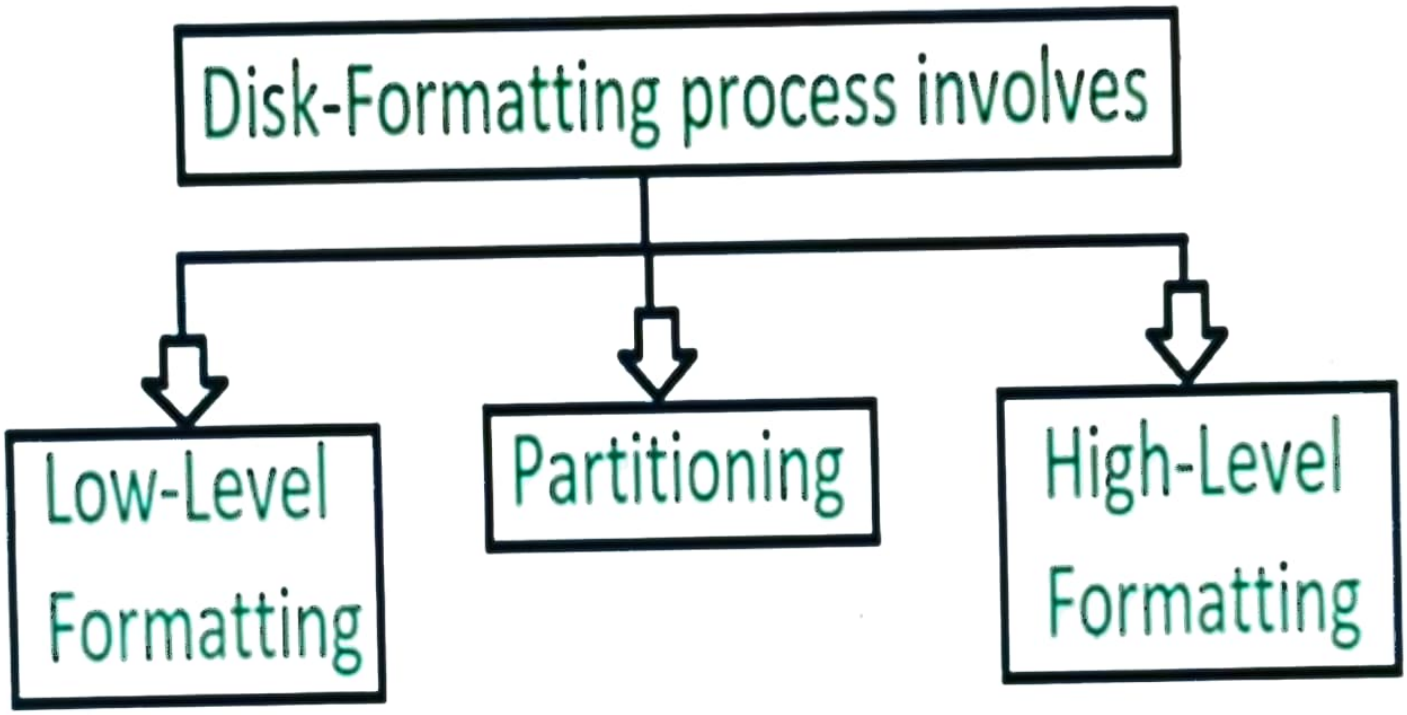
- As we know that disk formatting deletes data and removes all the programs installed with in the drive. So it can be done with caution. We must have the backup of all the data and applications which we require. No-doubt disk formatting requires time. But the frequent formatting of the disk decreases the life of the hard-drive.

platter hard-drives



solid-state drives




Disk-Formatting process involves
Low-Level Formatting
Partitioning
High-Level Formatting

Low-level Formatting

Low level formatting is a type of physical formatting. In is the process of marking of cylinders and tracks of the blank hard-disk. After this there is the division of tracks into sectors with the sector markers. Now-a-days low-level formatting is performed by the hard-disk manufactures themselves.

- We have data in our hard-disks and when we perform low-level formatting in the presence of data in the hard-disk all the data have been erased and it is impossible to recover that data. Some users make such a format that they can avoid their privacy leakage. Otherwise low-level will cause damage to hard-disk shortens the service-life.

- Therefore, this formatting is not suggested to users.

Partitioning

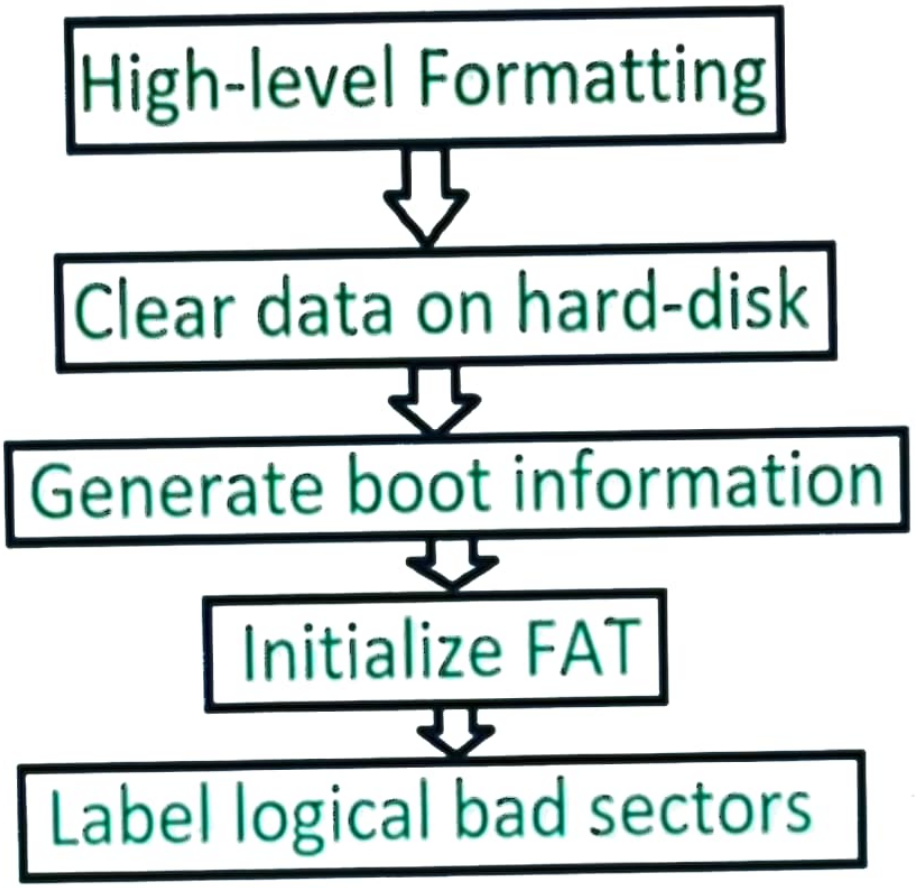
As suggesting from the name, partitioning means divisions. Partitioning is the process of dividing the hard-disk into one or more regions. The regions are called as partitions.

- It can be performed by the users and it will affect the disk performance.

High-level Formatting

- High-level formatting is the process of writing. Writing on a file system, cluster size, partition label, and so on for a newly created partition or volume.

- It is done to erase the hard-disk and again installing the operating system on the disk-drive.


High-level Formatting
Clear data on hard-disk
Generate boot information
Initialize FAT
Label logical bad sectors

- Firstly High-level formatting clears the data on hard-disk, then it will generate boot information, the it will initialize FAT after this it will go for label logical bad sectors when partition has existed.

- Formatting done by the user is the high-level formatting.

- Generally, It does not harm the hard-disk. It can be done easily with the Administrator, Windows snap-in Disk Management tool, disk part, etc.

- We can use such a format to use such a format to fix some problems like errors in the file system, corrupted hard-drive and develop bad sectors.