Operating System

BIM IV Semester

Credits: 3

Lecture Hours:48

Er. Santosh Bhandari,
(Master Computer Science)

Unit-6

Device Management:

Introduction

-Device management, which controls how software applications interact with the hardware attached to the computer system.

-It is the process of implementation, operation, and maintenance of a device by an operating system is called device management.

Functions of Device Management

- Keeps track of all devices and the program that is responsible for performing called the I/O controller.
- Monitoring the status of each device.
- Enforcing preset policies and making a decision on which process gets the device when and for how long.
- Allocates and deallocates the device efficiently.

Types of Device Management

1. Boot Device: Boot devices are the part of the hardware that consists of the data or files necessary to start the computer or PC. Eg: Hard disk, CD-ROM drive, floppy disk drive, DVD drive, and USB.

2. Character Device: Any device in which to give input or take the output the stream of characters is passed as they don't have their own memory. Example: printer, keyboards

3. Network Device: The devices help to connect the computer to a network by transmitting the data packs.

Features of Device Management

1. The OS is responsible in managing device communication through their respective drivers.
2. The OS keeps track of all devices using input-output controller.
3. It decides which process to assign to CPU and for how long.
4. O.S. is responsible for fulfilling the request of devices to access the process.
5. It connects the devices to various programs in an efficient way.
6. Deallocate devices when they are not in use.

Types of Devices

1. Dedicated Device

-Assigned to only one task at a time.

-when they are allocated to a process, they are not released until the job is completed.

-They don’t have their own memory.

Example: Plotters, printers, tape drives

2. Shared Devices

-These devices could be assigned to a variety of processes.

-Devices that can be shared or allocated between two or more processes at the same time.

Example: HDDs or SDDs,

Types of Devices

3. Virtual Device

-Combination of dedicated and shared devices.

-when the dedicated devices are converted into shared devices then they are known as virtual devices.

For example: Sharable printer.

Techniques for Accessing a Device

Polling: In this instance, a CPU keeps an eye on the status of the device to share data.

Interrupt-Driven I/O: Notifying the associated driver of the device's availability is the device controller's job.

DMA(Direct Memory Access) : Data motions are carried out by using a second controller.

Double Buffering: This mode of access has two buffers. One fills up while the other is utilized, and vice versa. To hide the line-by-line scanning from the viewer, this technique is frequently employed in animation and graphics.

Classification of I/O devices

1. Input Devices:

-Used to send signals to the computer for performing tasks.

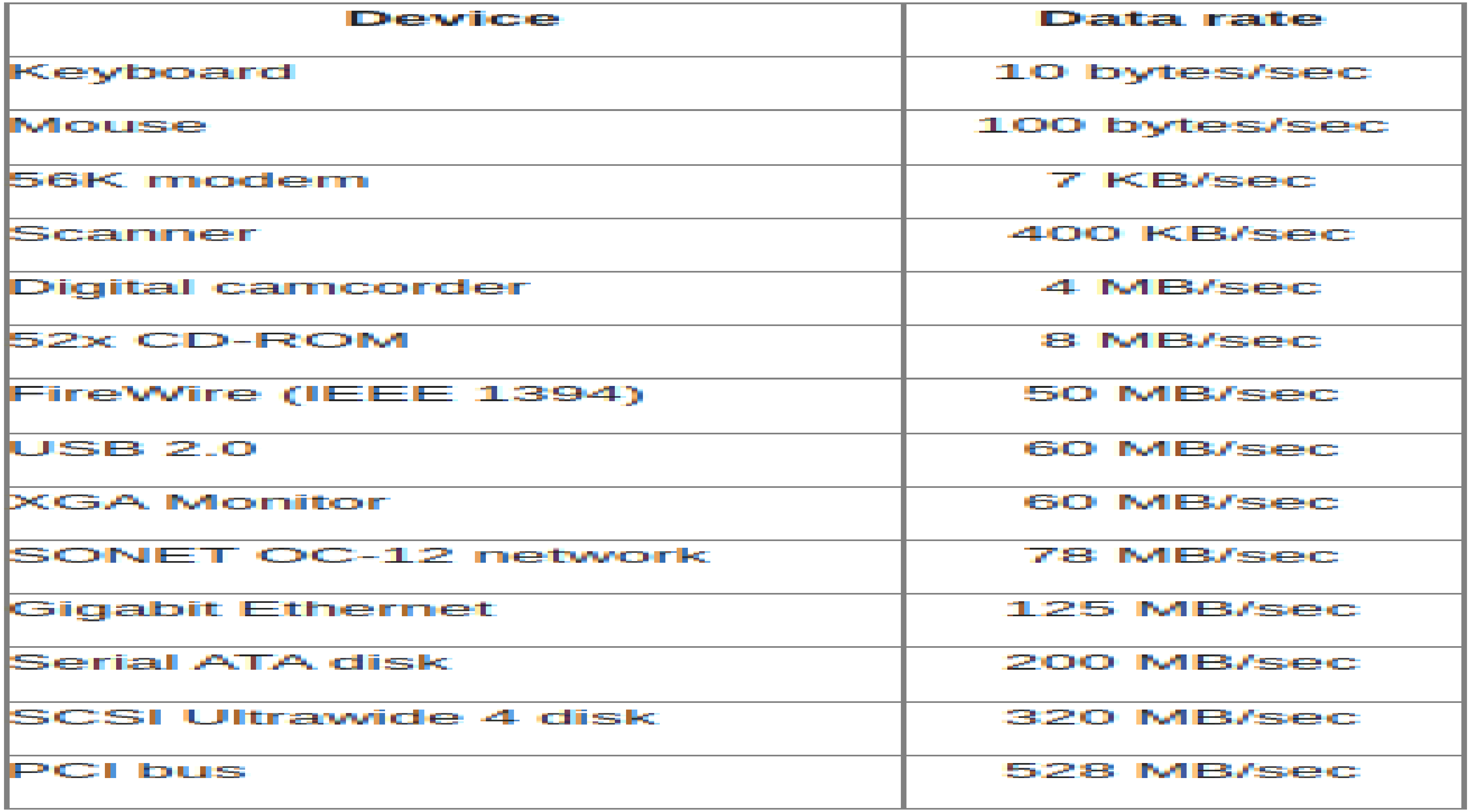
Classifications of Input devices are:

- Keyboard Devices
- Pointing Devices
- Composite Devices
- Game Controller
- Visual Devices
- Audio Input Devices

2. Output Devices: Output Devices are the devices that show us the result after giving the input to a computer system.

Categories: Screen device and Speaker devices

Data rate



Device	Data rate
Keyboard	10 bytes/sec
Mouse	100 bytes/sec
56K modem	7 KB/sec
Scanner	400 KB/sec
Digital camcorder	4 MB/sec
52x CD-ROM	8 MB/sec
FireWire (IEEE 1394)	50 MB/sec
USB 2.0	60 MB/sec
XGA Monitor	60 MB/sec
SONET OC-12 network	78 MB/sec
Gigabit Ethernet	125 MB/sec
Serial ATA disk	200 MB/sec
SCSI Ultrawide 4 disk	320 MB/sec
PCI bus	528 MB/sec

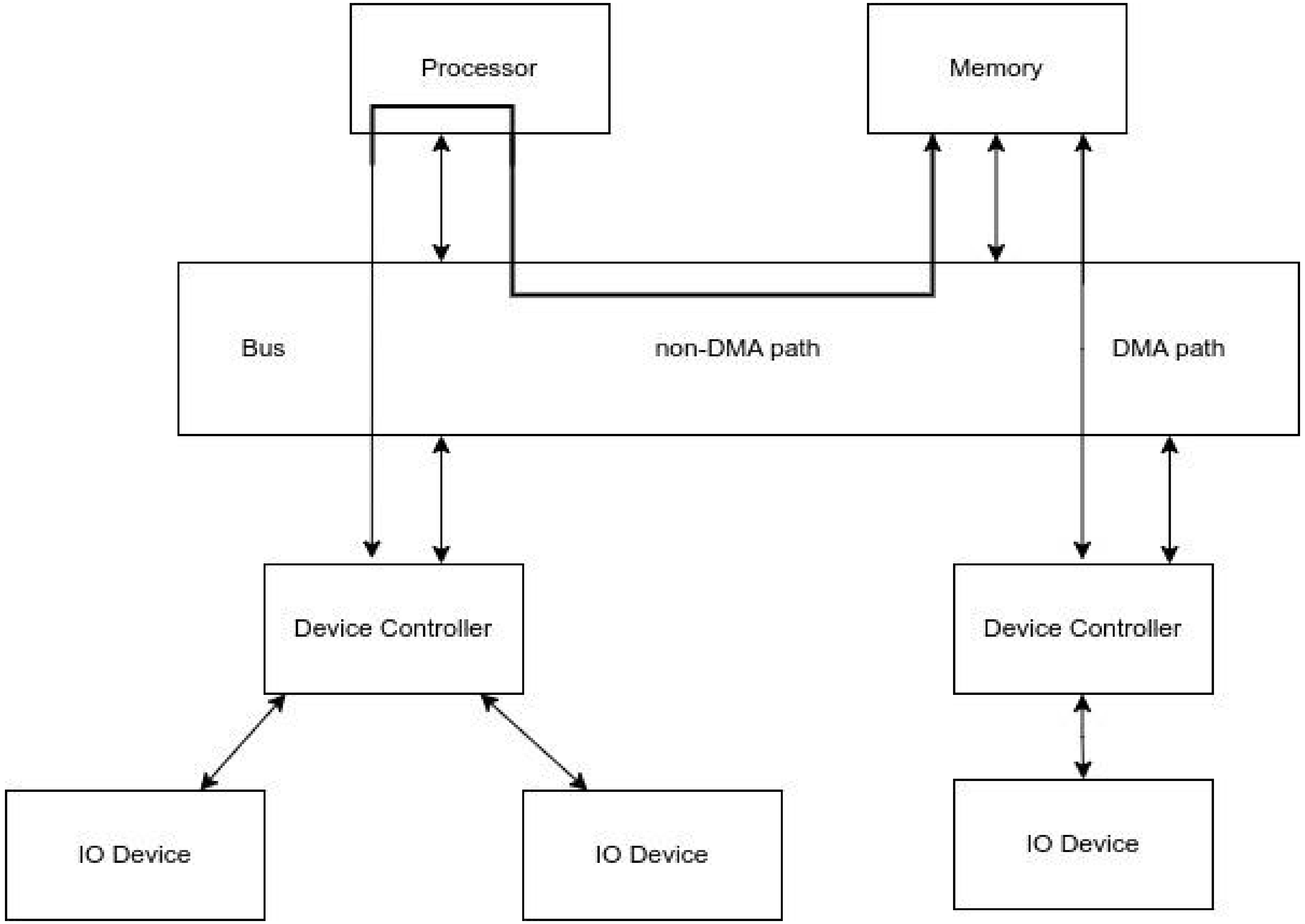
Device Controller

-The OS manages its task with the help of device controller.

-The device controller knows how to communicate with the operating system and I/O devices.

- Device controller is an interface between the OS and I/O devices.

-The device controller communicates with the system using the system bus.



Memory Mapped I/O

-CPU needs to communicate with the various memory and input-output devices (I/O).

-Data between the processor and these devices flow with the help of the system bus.

There are three ways in which system bus can be allotted to them :

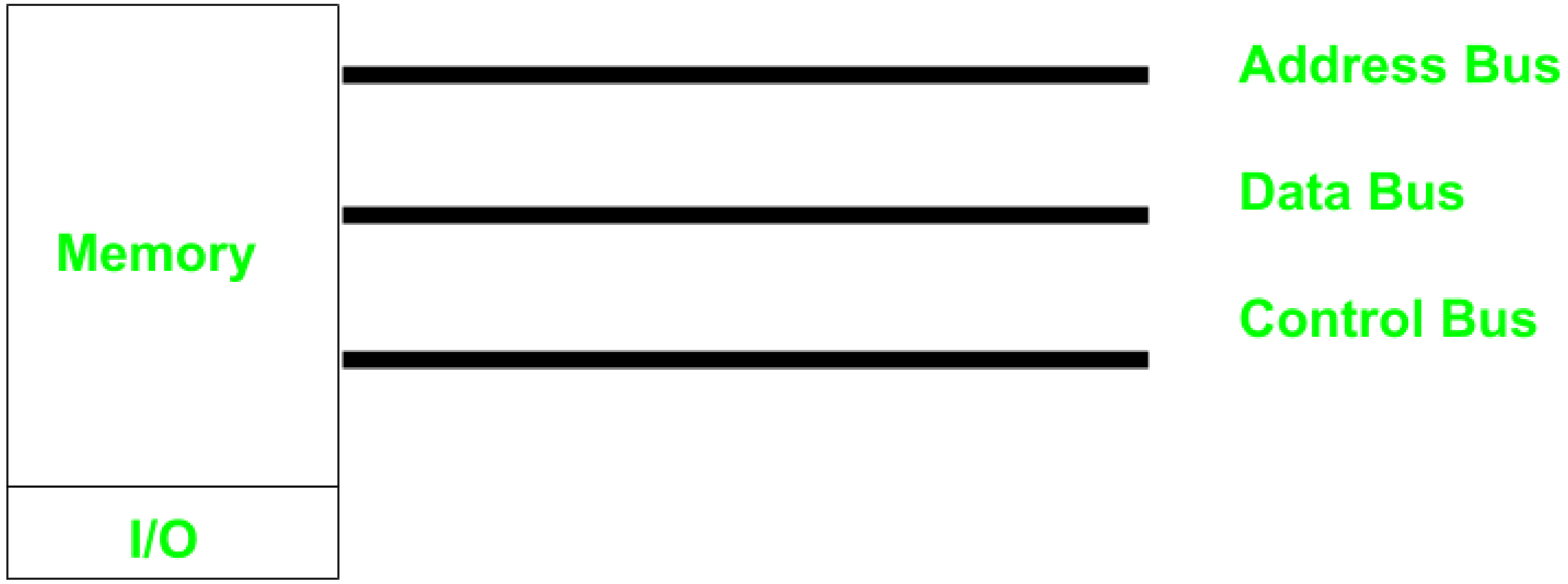
1. Separate set of address, control and data bus to I/O and memory.
2. Have common bus (data and address) for I/O and memory but separate control bus.
3. Have common bus (data, address, and control) for I/O and memory.

Memory Mapped I/O

Memory Mapped I/O :

-In this case, every bus is in common due to which the same set of instructions work for memory and I/O.

Hence we manipulate I/O same as memory and both have same address space, due to which the addressing capability of memory becomes less because some part is occupied by the I/O.



Application of Memory Mapped I/O

Graphics processing: Memory-mapped I/O is often used in graphics cards to provide fast access to frame buffers and control registers. The graphics data is mapped directly to memory, allowing the CPU to read from and write to the graphics card as if it were accessing regular memory.

Network communication: Network interface cards (NICs) often utilize memory-mapped I/O to transfer data between the network and the system memory. The NIC registers are mapped to specific memory addresses, enabling efficient data transfer and control over network operations.

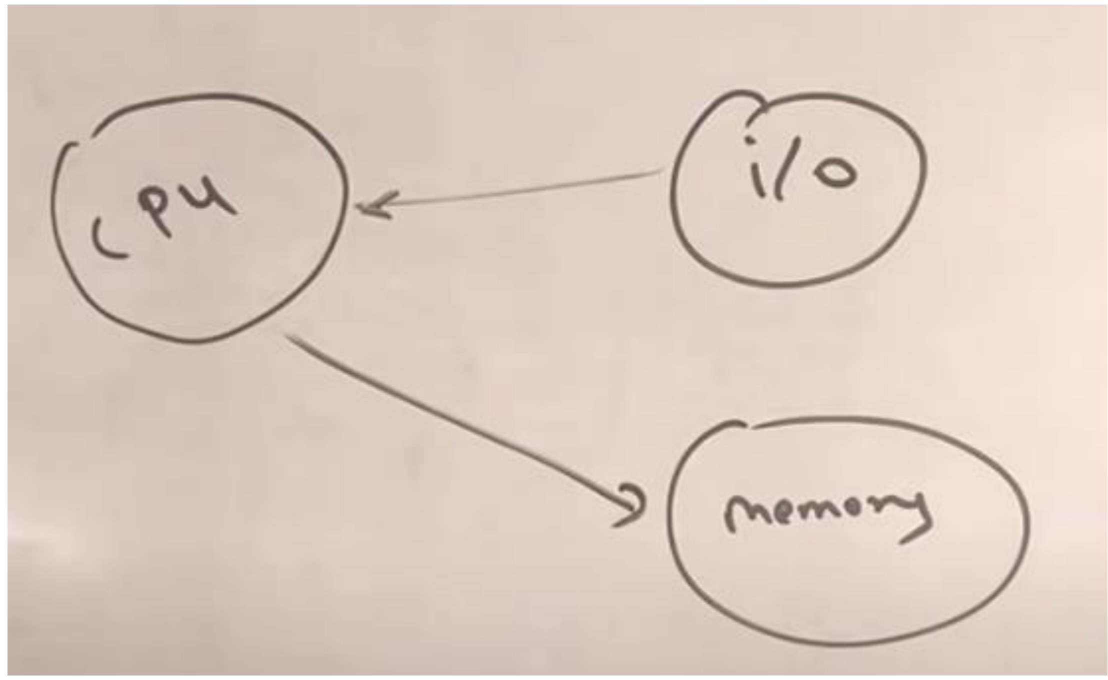
Direct memory access (DMA): DMA controllers employ memory-mapped I/O to facilitate high-speed data transfers between devices and system memory without CPU intervention. By mapping the DMA controller registers to memory, data can be transferred directly between devices and memory, reducing CPU overhead.

Direct memory access (DMA):

Problem:

If I/O wants to access memory it goes to the CPU and access memory: which is two cycle and is Time consuming.

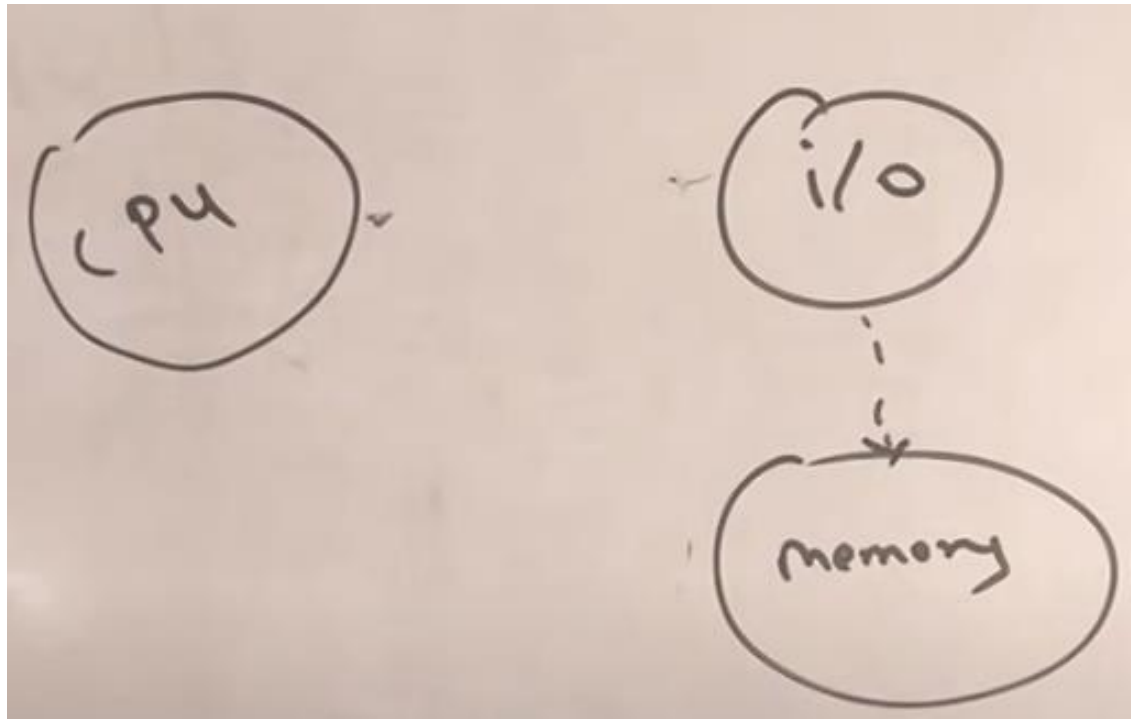
To solve this DMA Comes into the picture.



What is DMA?

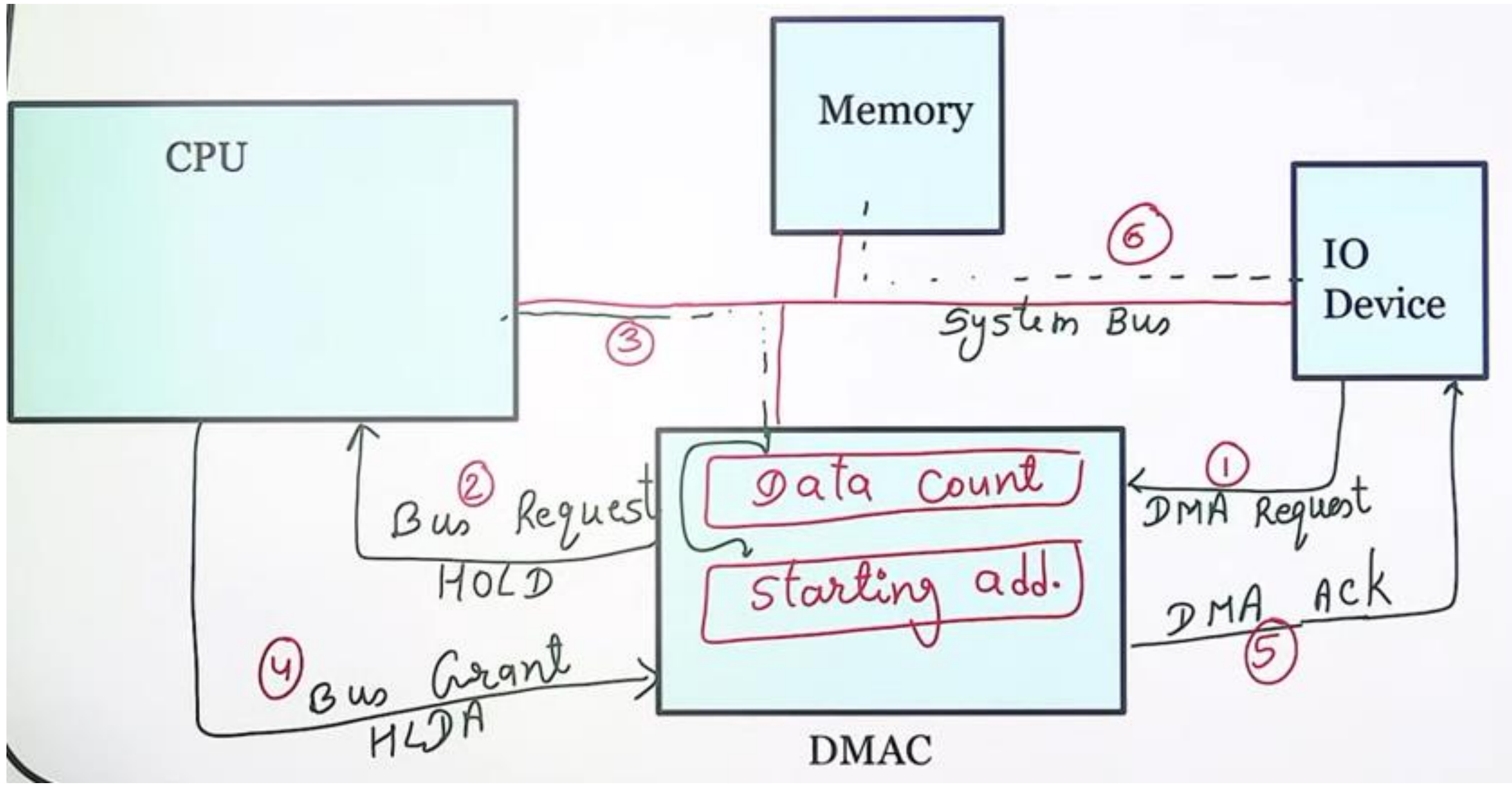
-DMA (Direct memory access) is the special feature within the computer system that transfers the data between memory and I/O without the intervention of the CPU.

-This is done by the OS, with the help of DMA controller by telling where the data lives on the memory, how much to copy, and which device it should send.



How DMA Works?

1. I/O wants to send data to memory, so it sends a DMA request to the DMA controller
2. The DMA controller sends the bus request (HOLD) to the CPU because the BUS is under the control of the CPU. However, the DMA controller wants to access the bus for data transfer.
3. CPU sends HOLD acknowledgment (HLDA) to the DMA controller through BUS and provides the BUS access with two information (Data Count and Starting Address.
4. DMA sends ACK to I/O device about direct BUS access for data transfer
5. I/O directly sends data to the memory



Data Transfer Mode in DMA?

Modes of Data Transfer in DMA: There are 3 modes.

Burst Mode: In Burst Mode, buses are handed over to the CPU by the DMA if the whole data is completely transferred, not before that.
Cycle Stealing Mode: In Cycle Stealing Mode, buses are handed over to the CPU by the DMA after the transfer of each byte.
Transparent Mode: Transparent Mode in DMA does not require any bus in the transfer of the data as it works when the CPU is executing the transaction.

Types of Direct Memory Access (DMA)

There are four popular types of DMA.

1. Single-Ended DMA: Single-Ended DMA Controllers operate by reading and writing from a single memory address. They are the simplest DMA.

2. Dual-Ended DMA: Dual-Ended DMA controllers can read and write from two memory addresses. Dual-ended DMA is more advanced than single-ended DMA.

3. Arbitrated-Ended DMA: Arbitrated-Ended DMA works by reading and writing to several memory addresses. It is more advanced than Dual-Ended DMA.

4. Interleaved DMA: Interleaved DMA are those DMA that read from one memory address and write from another memory address.

Advantages of DMA Controller

Advantages of DMA Controller

- Data Memory Access speeds up memory operations and data transfer.
- CPU is not involved while transferring data.
- DMA requires very few clock cycles while transferring data.
- DMA distributes workload very appropriately.
- DMA helps the CPU in decreasing its load.

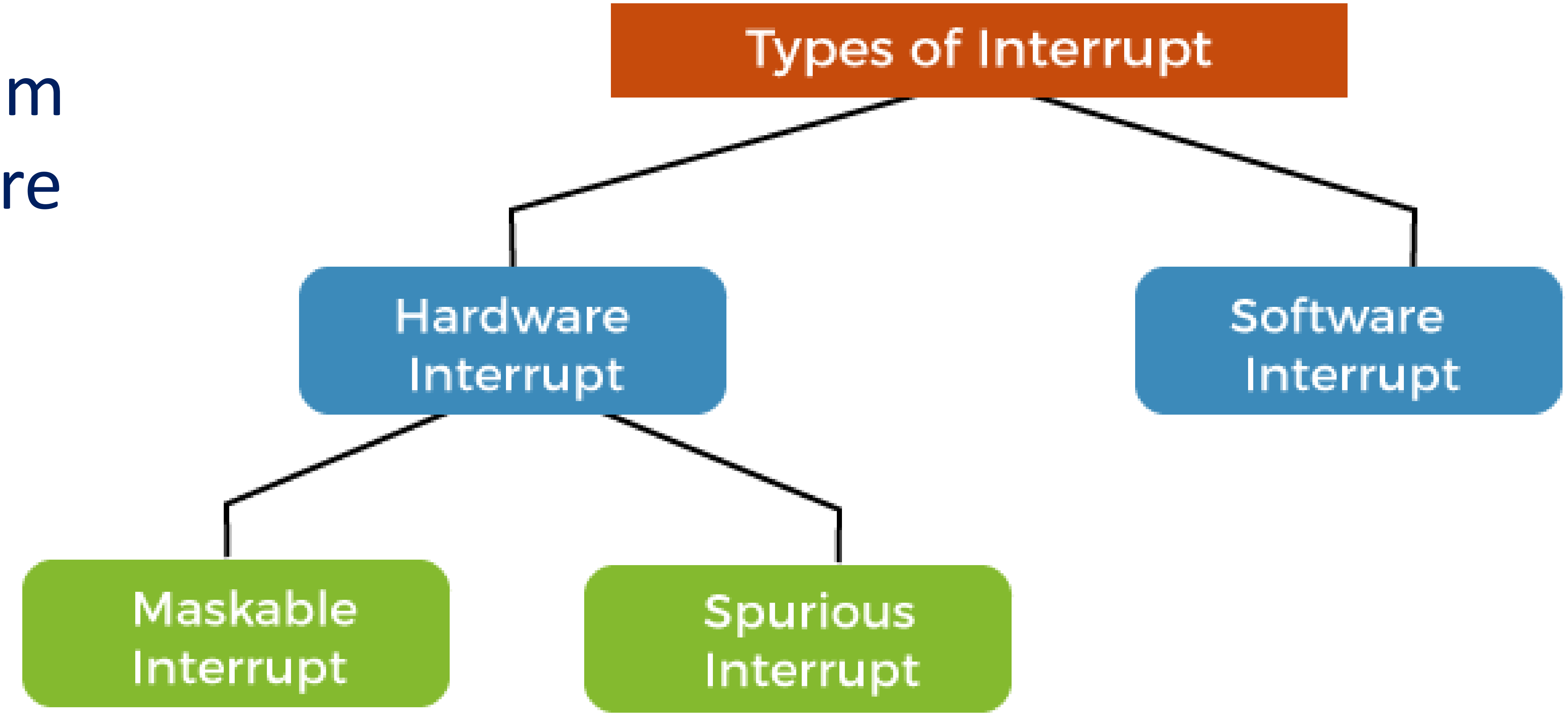
Interrupt

-An interrupt is a signal emitted by hardware or software when a process or an event needs immediate attention.
-In I/O devices, one of the bus control lines is dedicated for this purpose and is called the Interrupt Service Routine (ISR).
-Interrupt signals the operating system to stop and figure out what to do next.

Types of Interrupt

1. Hardware interrupts

The interrupt signal generated from external devices and i/o devices are made interrupt to CPU when the instructions are ready.



For example – In a keyboard if we press a key to do some action this pressing of the keyboard generates a signal that is given to the processor to do action, such interrupts are called hardware interrupts.

Types of Interrupt

Hardware interrupts are classified into two types

Maskable Interrupt – The hardware interrupts that can be delayed when a highest priority interrupt has occurred to the processor.

Non-Maskable Interrupt – The hardware that cannot be delayed and immediately be serviced by the processor.

Types of Interrupt

2. Software interrupts
The interrupt signal generated from internal devices and software programs need to access any system call then software interrupts are present.

Exception – Exception is nothing but an unplanned interruption while executing a program. For example – while executing a program if we got a value that is divided by zero is called an exception.

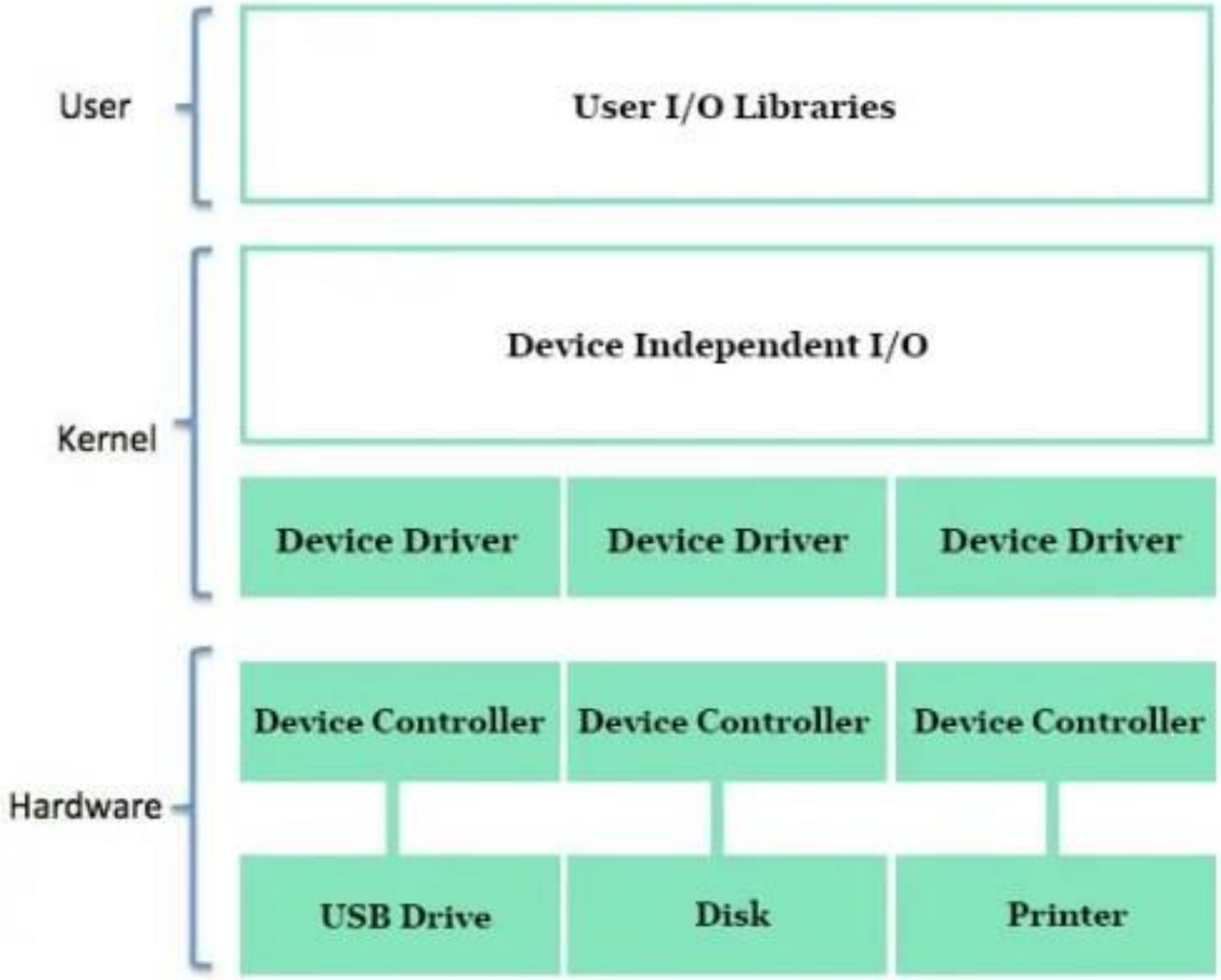
I/O Software

-I/O software: is a critical component of an operating system.
-Its primary function is to manage the interaction between the computer's hardware and software.
-This interaction involves the transfer of data between the central processing unit (CPU) and various peripheral devices,

Organizational levels of I/O Software

Three Layer:

1. User Level
2. Kernel Level
3. Hardware Level



Organizational levels of I/O Software

1. User Level Libraries:

Role: User Level Libraries are software components that operate in user space, outside the kernel. They provide a high-level interface for application programs to interact with I/O devices

Functions: User Level Libraries offer functions and APIs (Application Programming Interfaces) that allow applications to request I/O operations, such as reading from or writing to files, sockets, or other devices.

Example: Standard C library functions like fread, fwrite, and fprintf for file I/O.

Organizational levels of I/O Software

2. Kernel Level Modules:

Role: Kernel Level Modules operate within the operating system’s kernel, which is the core of the operating system.

Functions: Kernel Level Modules include device drivers, interrupt handlers, and other components responsible for handling I/O requests.

3. Hardware:

Role: Hardware represents the physical devices that perform input and output operations. These can include components like hard drives, SSDs, graphics cards, network adapters, and peripheral devices like keyboards, mice, and printers.

Functions: Hardware devices are responsible for the actual data transfer between the computer system and the external world.

Input/output Software Layers

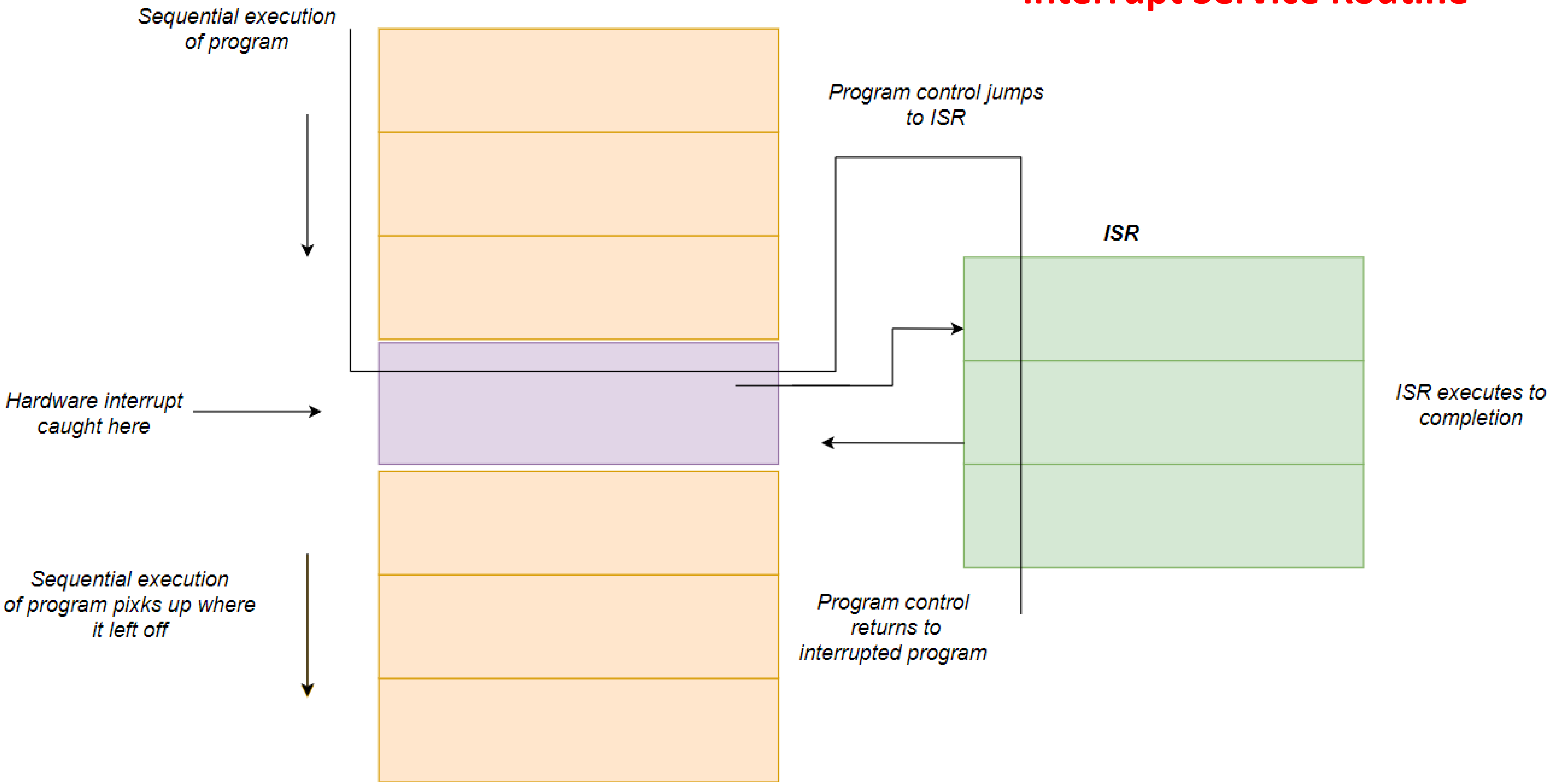
Input/output software have four layers:

1. Interrupt handlers
2. Device drivers
3. Device-independent input/output software
4. User-space input/output software

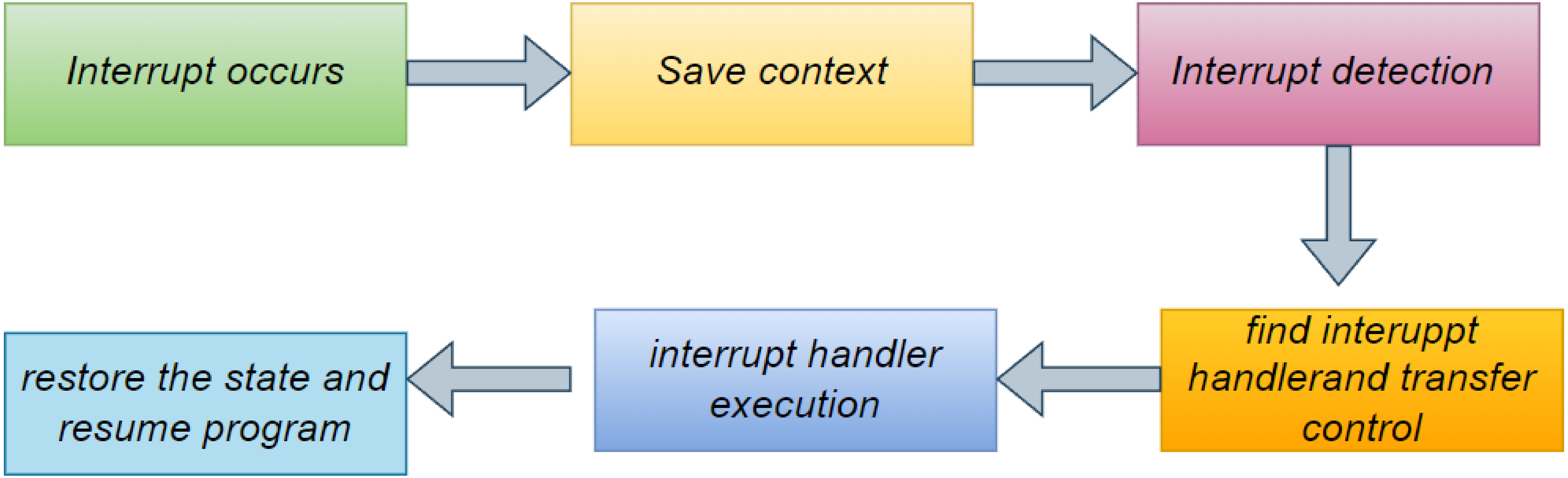
Interrupt handlers

Interrupt handlers are responsible for handling interrupts promptly and effectively. Their primary role is to service interrupts and perform necessary actions associated with specific events. Interrupt handlers ensure that time-critical tasks, such as responding to user input or processing data from hardware devices, are addressed in a timely manner.

Interrupt handlers


Interrupt Service Routine
Sequential execution of program
Program control jumps to ISR
ISR
Hardware interrupt caught here
ISR executes to completion
Sequential execution of program pixks up where it left off
Program control returns to interrupted program

Workflow for Interrupt handlers



Interrupt occurs

Save context

Interrupt detection

find interuppt handlerand transfer control

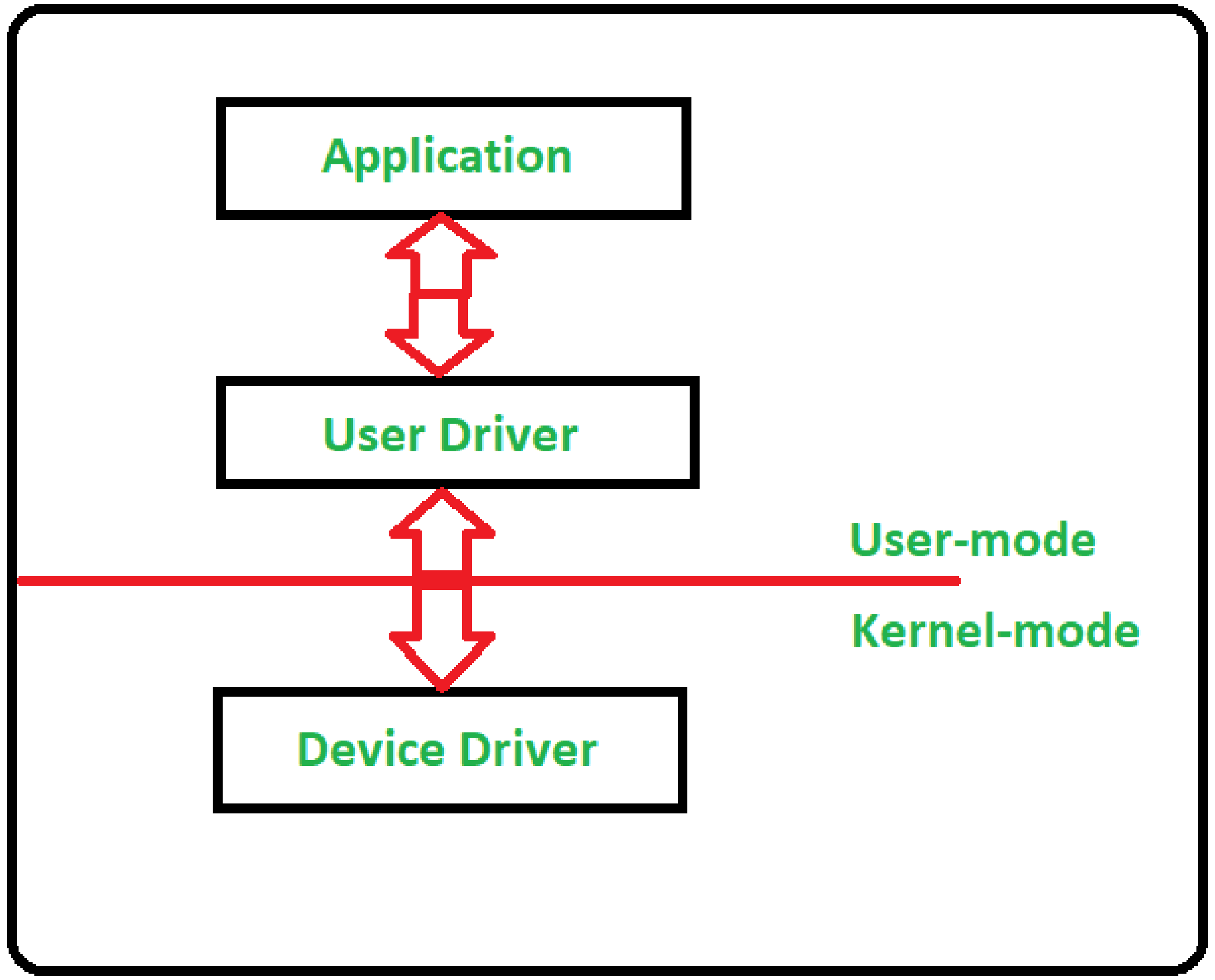
interrupt handler execution

restore the state and resume program

Device Drivers

What is a device driver?

A device driver is a piece of software that enables communication between an operating system or application and hardware or peripheral devices. It serves as a bridge between the different components of a computer, allowing them to interact with each other.

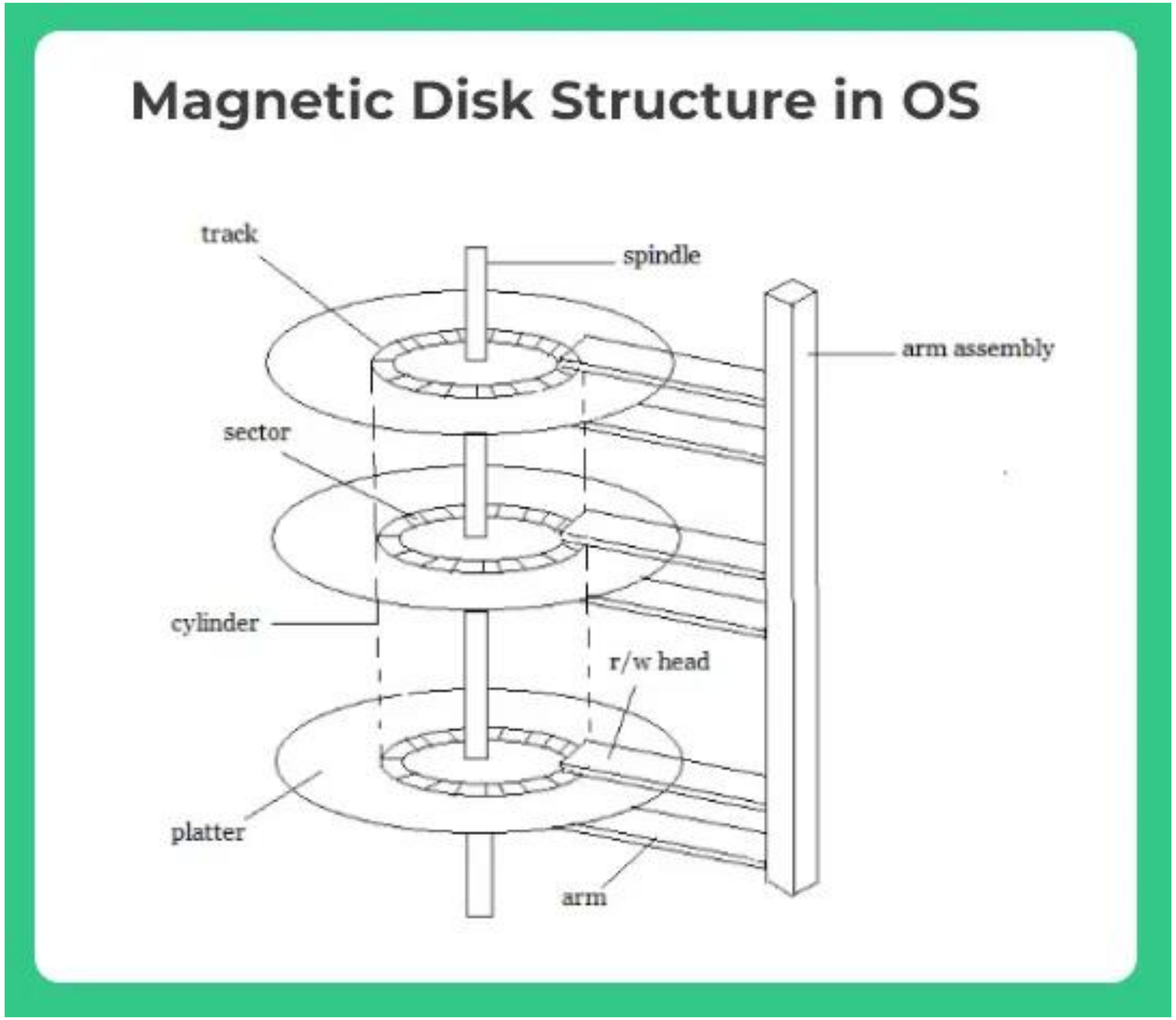


Device Drivers are essential for a computer system to work properly because without a device driver the particular hardware fails to work accordingly.

Disk Structure

1. Platter: it has two surfaces (Upper and lower)
2. Each platter consists of tracks on both the upper and lower surface.
3. Each track is further divided into sectors.
4. Each sector consists of data
5. The spindle revolves the platters and is controlled by r/w unit of OS.
6. Arm Assembly is there which keeps a pointy r/w head on each disk to read or write on a particular disk.

Platter->surface->Track->Sector->data



Disk Structure

Considering there are 8 platters and 2 surfaces in each platter, each surface consists of 256 tracks, each track has 512 sectors, and each sector consists of 512KB data.

1. Calculate the disk size.
2. Calculate the number of bits to represent disk size

Disk size = **Platter*surface*Track*Sector*data**
=8*2*256*512*512KB
=2^3 * 2^1 * 2^8 * 2^9 * 2^9 * 2^10
=2^40 Byte
=1TB
Number of bits 40.

1 KB = 2^10 byte
1 MB = 2^20 byte
1 GB = 2^30 byte
1 TB = 2^40 byte

Disk Structure

Suppose there are eight platters in the hard disk drive. Each Platter has two surfaces so that 16-surfaces will be in the hard drive. Therefore, the required R/W head will also be 16. Suppose there are 1,024 cylinders and 128 sectors in each track. The sector size is 512 bytes. Then,

Calculate Disk Size.

Solution

Size of Hard Disk = Cylinder x Heads X Sectors x Sector-Size

= 1,024 x 16 x 128 x 512 Bytes

= 2^{10}x 2^{4} x 2^{7} x2^{9} Bytes = 2^{30}Bytes = 2GB

Disk Access time

The speed of the disk is measured in two parts:

1. Transfer rate: This is the rate at which the data moves from the disk to the computer.

2. Random access time: It is the sum of the seek time and rotational latency.

Seek time is the time taken by the arm (R/W) head to reach the desired track.

Rotation Time: Time taken for 1 full rotation (360)

Rotational latency: Time taken by the arm to reach the required sector in the track (Half of rotation time)

Disk Structure

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

Transfer time = Data to be transfer/transfer rate

Transfer Rate = Number of head (Surface) * Capacity of one track * Number of rotation in one second.

Disk Structure

Data Transfer Time: Data Transfer Time is the total time taken to transfer a specific amount of data from the disk. Data Transfer time depends on the data transfer rate of the disk.

Data Transfer Rate: Data transfer rate is define as the amount of data transfer in per unit time for example 30 MB/Sec.

Average Access Time: Average access time is calculated as:
Average Access Time = Seek Time + Rotational Latency + Data Transfer Time

Disk Structure

Important Formulas

1. Disk Access Time is calculated as

Disk access time = Seek time + Rotational delay + Transfer time + Controller overhead + Queuing delay

2. Average Disk Access Time is calculated as

Average disk access time = Average seek time + Average rotational delay + Transfer time + Controller overhead + Queuing delay

3. Average Seek Time is calculated as

Average seek time = 1 / 3 x Time taken for one full stroke

Disk Structure

Important Formulas

1. Disk Access Time is calculated as

Disk access time = Seek time + Rotational delay + Transfer time + Controller overhead + Queuing delay

2. Average Disk Access Time is calculated as

Average disk access time = Average seek time + Average rotational delay + Transfer time + Controller overhead + Queuing delay

3. Average Seek Time is calculated as

Average seek time = 1 / 3 x Time taken for one full stroke

Disk Structure

4. Average Rotational Latency is calculated as

Average rotational latency = 1 / 2 x Time taken for one full rotation

5. Capacity Of Disk Pack is calculated as

Capacity of a disk pack = Total number of surfaces x Number of tracks per surface x Number of sectors per track x Storage capacity of one sector

6. Formatting Overhead is calculated as

Formatting overhead= Number of sectors x Overhead per sector

7. Formatted Disk Space-Formatted disk space also called as usable disk space is the disk space excluding formatting overhead. It is calculated as

Formatted disk space= Total disk space or capacity – Formatting overhead

Disk Structure

8. Recording Density Or Storage Density is calculated as

Storage density of a track = Capacity of the track / Circumference of the track

9. Track Capacity is calculated as

Capacity of a track = Recording density of the track x Circumference of the trace

10. Data Transfer Rate is calculated as

Data transfer rate = Number of heads x Bytes that can be read in one full rotation x Number of rotations in one second or

Data transfer rate= Number of heads x Capacity of one track x Number of rotations in one second

11. Tracks Per Surface is calculated as

Total number of tracks per surface = (Outer radius – Inner radius) / Inter track gap

Disk Structure

Q1. Consider a disk with 4 platters, 2 surfaces per platter, 1000 tracks per surface, 50 sectors per track, and 512 bytes per sector. What is the disk capacity?

Solution − To calculate the disk capacity, we can follow these steps:
Step 1 − Calculate the total number of tracks on the disk−
Total tracks = No.of platters * No.of surfaces per platter * No.of tracks per surface
Total tracks = 4 * 2 * 1000 = 8000
Step 2 − Calculate the total number of sectors on the disk−
Total sectors = Total tracks * Number of sectors per track
Step 3 − Calculate the total storage capacity of the disk
Total capacity = Total sectors * Number of bytes per sector
Total capacity = 400,000 * 512 = 204,800,000 bytes
Therefore, the disk capacity is 204,800,000 bytes or approximately 204.8 megabytes.

Disk Structure

Consider a disk pack with the following specifications- 16 surfaces, 128 tracks per surface, 256 sectors per track and 512 bytes per sector.

Answer the following questions-

1. What is the capacity of disk pack?
2. What is the number of bits required to address the sector?
3. If the format overhead is 32 bytes per sector, what is the formatted disk space?
4. If the format overhead is 64 bytes per sector, how much amount of memory is lost due to formatting?
5. If the diameter of innermost track is 21 cm, what is the maximum recording density?
6. If the diameter of innermost track is 21 cm with 2 KB/cm, what is the capacity of one track?
7. If the disk is rotating at 3600 RPM, what is the data transfer rate?
8. If the disk system has rotational speed of 3000 RPM, what is the average access time with a seek time of 11.5 msec?

Disk Hardware

Given

Number of surfaces = 16

Number of tracks per surface = 128

Number of sectors per track = 256

Number of bytes per sector = 512 bytes

Part-01: Capacity of Disk Pack-

Capacity of disk pack = Total number of surfaces x Number of tracks per surface x Number of sectors per track x Number of bytes per sector

= 16 x 128 x 256 x 512 bytes

= 2^{28} bytes

= 256 MB

Disk Hardware

Part-02: Number of Bits Required To Address Sector-

Total number of sectors = Total number of surfaces x Number of tracks per surface x Number of sectors per track

= 16 x 128 x 256 sectors = 524288

Number of bits required= $\log_2$(Total number of sectors)

Thus, Number of bits required to address the sector = $\log_2(524288) = 19$

Part-03: Formatted Disk Space- Formatting overhead= Total number of sectors x overhead per sector

= 2^{19} x 32 bytes

= 2^{19} x 2^5 bytes

= 2^{24} bytes

= 16 MB

Now, Formatted disk space = Total disk space – Formatting overhead

= 256 MB – 16 MB

= 240 MB

Disk Hardware

Part-04: Formatting Overhead-

Amount of memory lost due to formatting

= Formatting overhead

= Total number of sectors x Overhead per sector

= 2^{19} x 64 bytes

= 2^{19} x 2^{6} bytes

= 2^{25} bytes

= 32 MB

Disk Hardware

Part-05: Maximum Recording Density- Storage capacity of a track

= Number of sectors per track x Number of bytes per sector

= 256 x 512 bytes

= 2^8 x 2^9 bytes

= 2^{17} bytes

= 128 KB

Circumference of innermost track

= 2 x π x radius

= π x diameter

= 3.14 x 21 cm

= 65.94 cm

Now, Maximum recording density

= Recording density of innermost track

= Capacity of a track / Circumference of innermost track

= 128 KB / 65.94 cm

= 1.94 KB/cm

Disk Hardware

Part-06: Capacity Of Track-

Circumference of innermost track

= 2 x π x radius

= π x diameter

= 3.14 x 21 cm

= 65.94 cm

Capacity of a track = Storage density of the innermost track x Circumference of the innermost track

= 2 KB/cm x 65.94 cm

= 131.88 KB

≅ 132 KB

Disk Hardware

Part-07: Data Transfer Rate-

Number of rotations in one second

= (3600 / 60) rotations/sec

= 60 rotations/sec

Now, Data transfer rate

= Number of heads x Capacity of one track x Number of rotations in one second

= 16 x (256 x 512 bytes) x 60

= 2^4 x 2^8 x 2^9 x 60 bytes/sec

= 60 x 2^{21} bytes/sec

= 120 MBps

Disk Hardware

Part-08: Average Access Time

Time taken for one full rotation= (60 / 3000) sec

= (1 / 50) sec

= 0.02 sec=

20 msec

Average rotational delay= 1/2 x Time taken for one full rotation

= 1/2 x 20 msec

= 10 msec

Now, average access time= Average seek time + Average rotational delay + Other factors

= 11.5 msec + 10 msec + 0

= 21.5 msec

Disk scheduling algorithm

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.

Key Terms

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 written.

Disk scheduling algorithm

Rotational Latency: Rotational Latency is the time taken by the desired sector of the disk to rotate into a position so that it can access the read/write heads.

Transfer Time: Transfer time is the time to transfer the data.

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

Total Seek Time = Total head Movement * Seek Time

Disk Scheduling Algorithms

FCFS (First Come First Serve)

SSTF (Shortest Seek Time First)

SCAN (Elevator Algorithm)

C-SCAN (Circular SCAN)

LOOK

C-LOOK

RSS

LIFO (Last-In First-Out)

N-Step SCAN

F-SCAN

FCFS (First Come First Serve)

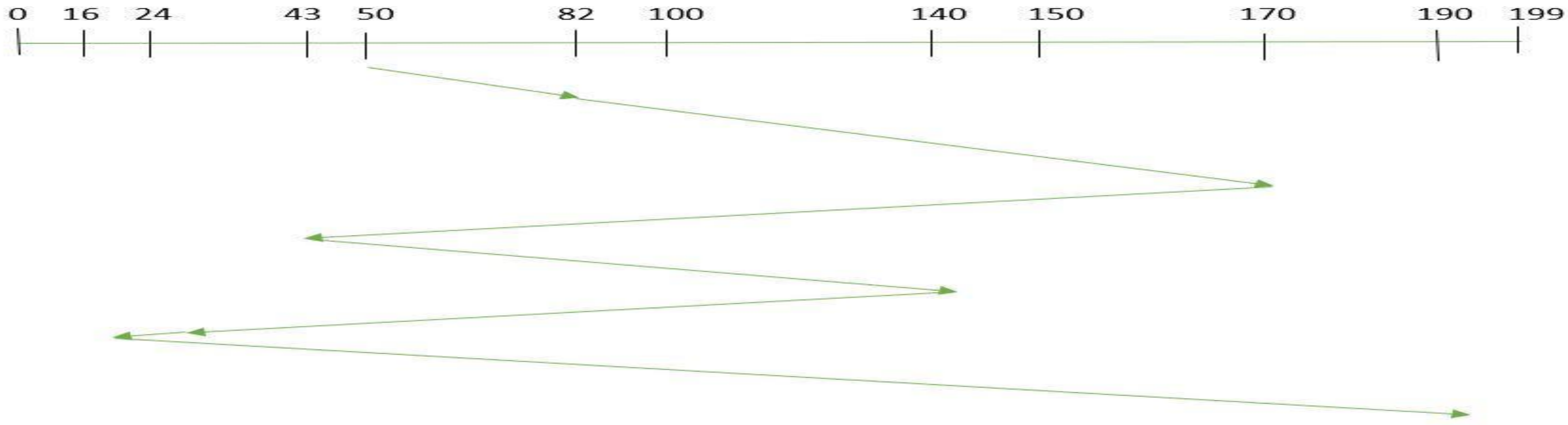
FCFS is the simplest of all Disk Scheduling Algorithms. In FCFS, the requests are addressed in the order they arrive in the disk queue.

FCFS (First Come First Serve)

Example: Suppose the order of request is- (82,170,43,140,24,16,190)

And current position of Read/Write head is: 50

Calculate total overhead movement (seek time).



Total overhead movement (total distance covered by the disk arm) = (82-50)+(170-82)+(170-43)+(140-43)+(140-24)+(24-16)+(190-16) =642

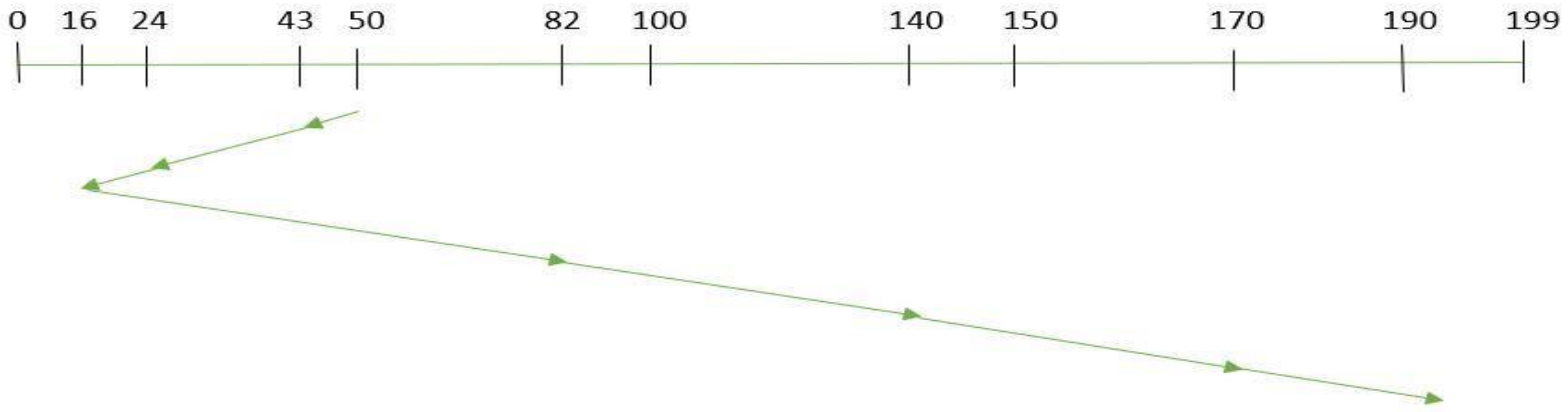
SSTF (Shortest Seek Time First)

In SSTF (Shortest Seek Time First), requests having the shortest seek time are executed first. So, the seek time of every request is calculated in advance in the queue and then they are scheduled according to their calculated seek time.

SSTF (Shortest Seek Time First)

Suppose the order of request is- (82,170,43,140,24,16,190)

And current position of Read/Write head is: 50



total overhead movement (total distance covered by the disk arm) = (50-43)+(43-24)+(24-16)+(82-16)+(140-82)+(170-140)+(190-170) =208

SCAN (Elevator) Algorithm

In the SCAN Disk Scheduling Algorithm, the head starts from one end of the disk and moves towards the other end, servicing requests in between one by one and reaching the other end. Then the direction of the head is reversed and the process continues as the head continuously scans back and forth to access the disk. So, this algorithm works as an elevator and is hence also known as the elevator algorithm.

SCAN (Elevator) Algorithm

Consider the disk contains 200 tracks (0-199).

Request sequence = {176, 79, 34, 60, 92, 11, 41, 114}

Initial head position = 50

Direction = left (We are moving from right to left)

Calculate

1. The total number of seek operations

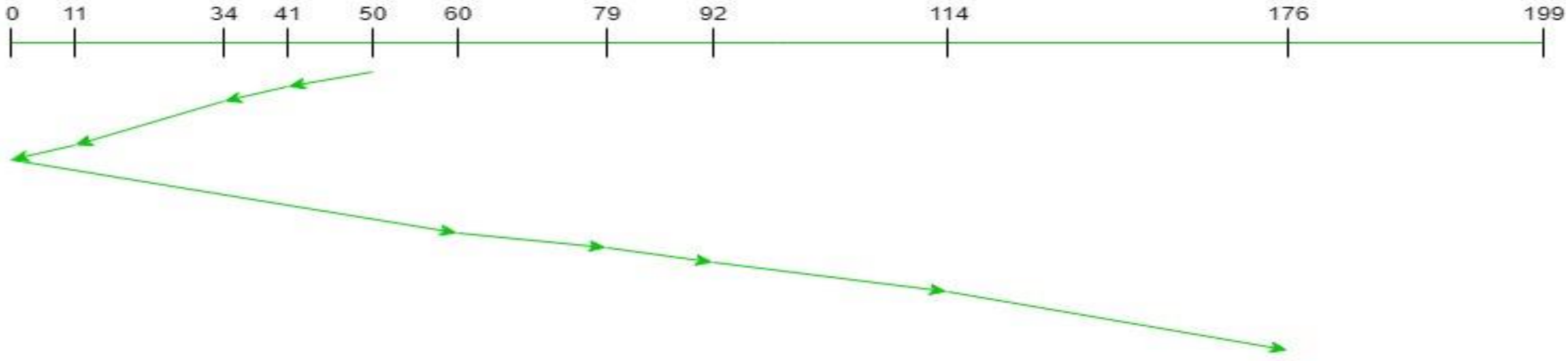
2. Seek Sequence

SCAN (Elevator) Algorithm

Request sequence = {176, 79, 34, 60, 92, 11, 41, 114}

Initial head position = 50

Direction = left (Right to left)



total Seek count =
= (50-41) + (41-34) + (34-11) + (11-0) + (60-0) + (79-60) + (92-79) + (114-92) + (176-114)
= 226

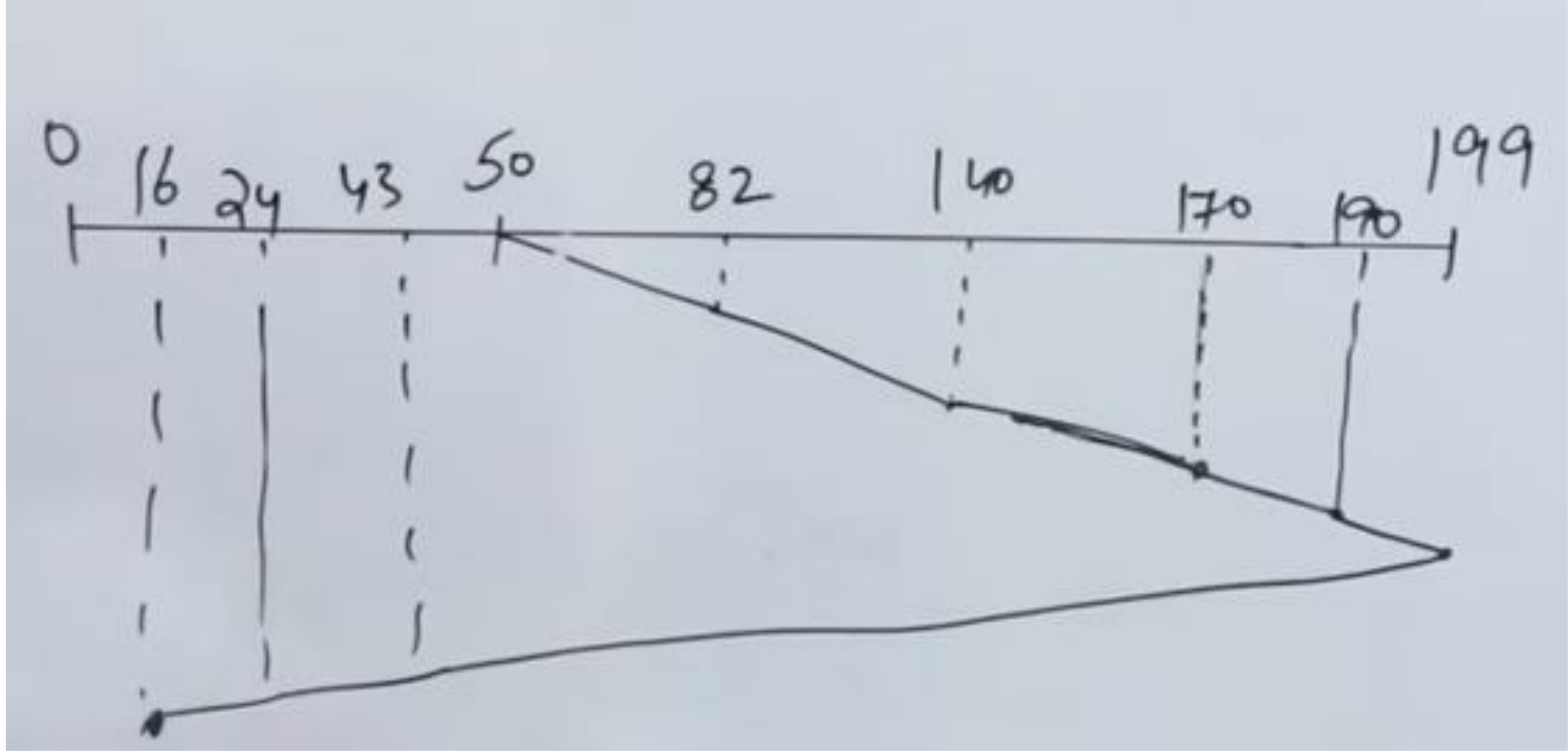
Seek Sequence is: 41, 34, 11, 0, 60, 79, 92, 114, 176

SCAN (Elevator) Algorithm

Request sequence = {82,170,43,140,24,16,190}

Initial head position = 50

Direction = Right (Left to right)



total Seek count =

= (199-50) + 100-16)

= 332

SCAN (Elevator) Algorithm

Advantages of SCAN (Elevator) Algorithm

- This algorithm is simple and easy to understand.
- SCAN algorithm has no starvation.
- This algorithm is better than the FCFS Disk Scheduling algorithm.

Disadvantages of the SCAN (Elevator) Algorithm

- More complex algorithm to implement.
- Starvation can occur

C-SCAN Algorithm

The Circular SCAN (C-SCAN) Scheduling Algorithm is a modified version of the SCAN.

That deals with the inefficiency of the SCAN algorithm.

Like SCAN (Elevator Algorithm), C-SCAN moves the head from one end servicing all the requests to the other end.

However, as soon as the head reaches the other end, it immediately returns to the beginning of the disk without servicing any requests on the return trip.

Direction = right(We are moving from left to right)

C-SCAN Algorithm

Consider the disk contains 200 tracks (0-199).

Request sequence = {176, 79, 34, 60, 92, 11, 41, 114}

Initial head position = 50

Direction = left (We are moving from right to left)

Calculate

1. The total number of seek operations

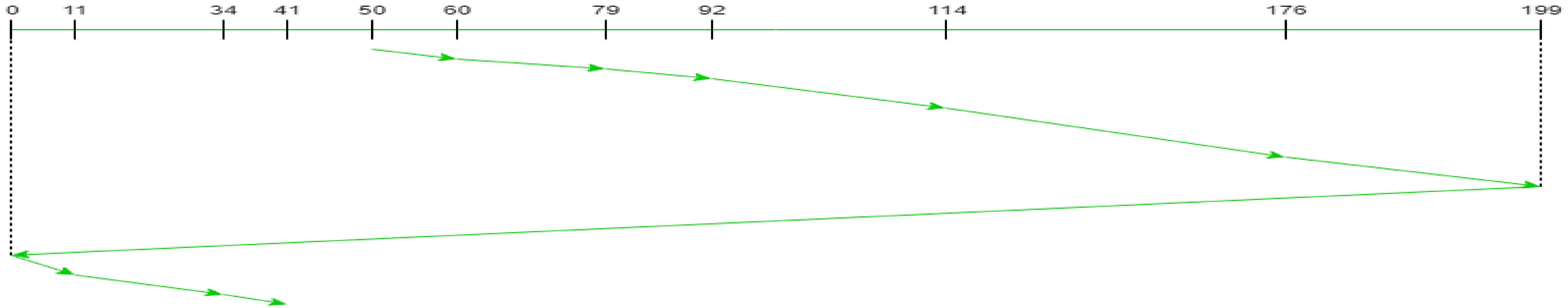
2. Seek Sequence is

C-SCAN Algorithm

Request sequence = {176, 79, 34, 60, 92, 11, 41, 114}

Initial head position = 50

Direction = Right



total Seek count =
= (60-50) + (79-60) + (92-79) + (114-92) + (176-114) + (199-176) + (199-0) + (11-0) + (34-11) + (41-34) = 389

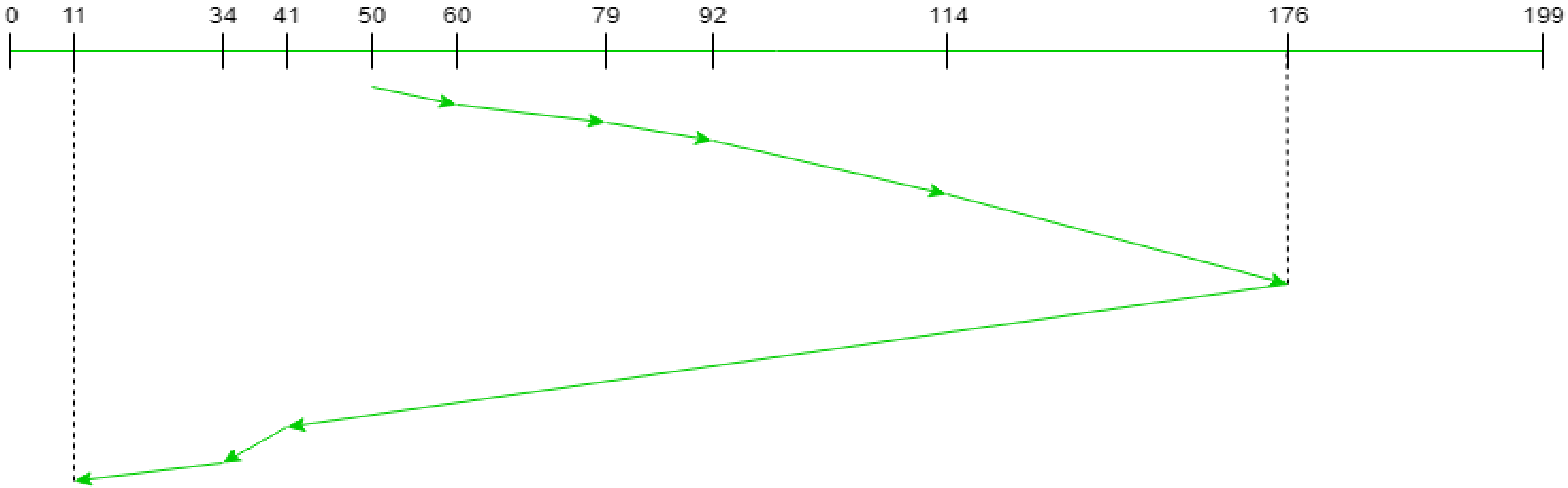
Seek Sequence: 60, 79, 92, 114, 176, 199, 0, 11, 34, 41

Look Algorithm

Request sequence = {176, 79, 34, 60, 92, 11, 41, 114}

Initial head position = 50

Direction = Right



Total Seek Time = (60-50) + (79-60) + (92-79) + (114-92) + (176-114) + (176-41) + (41-34) + (34-11) = 291

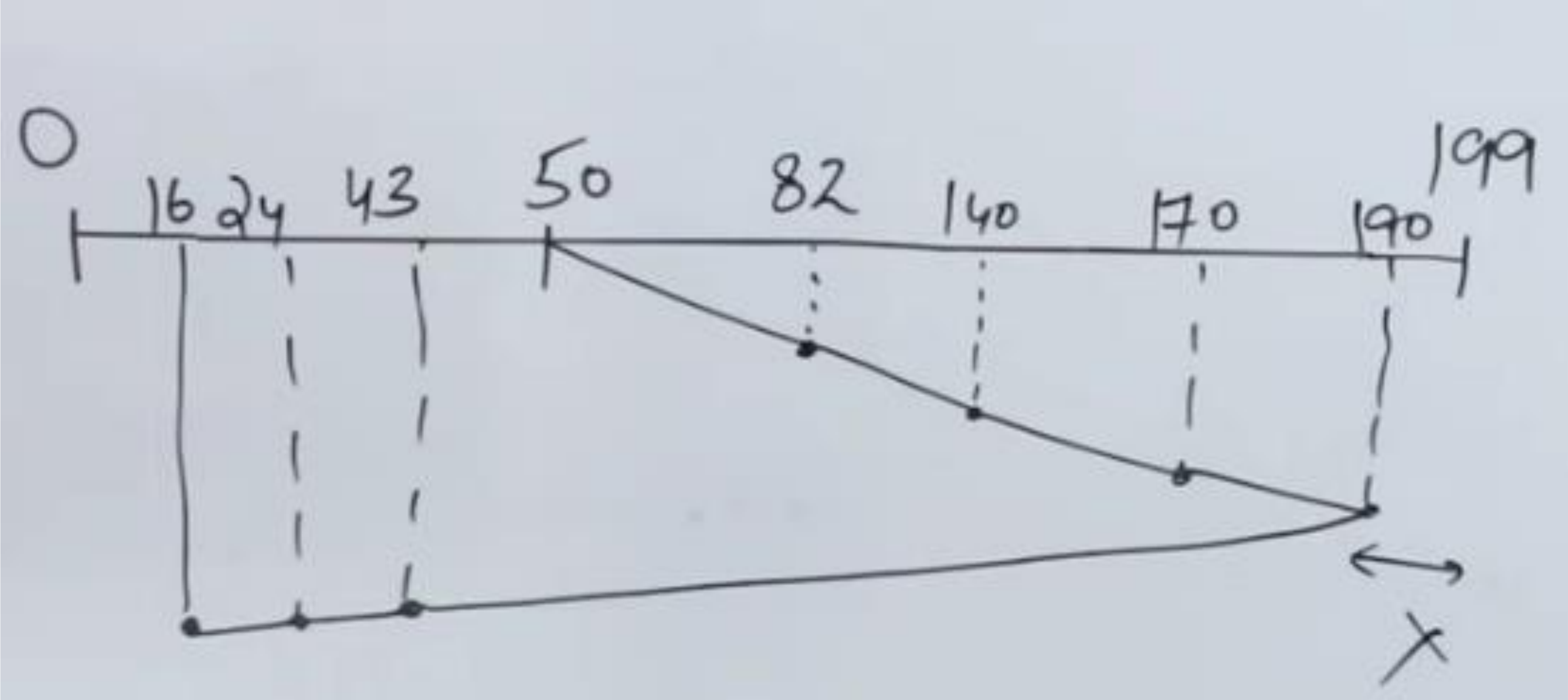
Seek Sequence: 60, 79, 92, 114, 176, 41, 34, 11

Look Algorithm

Request sequence = {82,170,43,140,24,16,190}

Initial head position = 50

Direction = Right



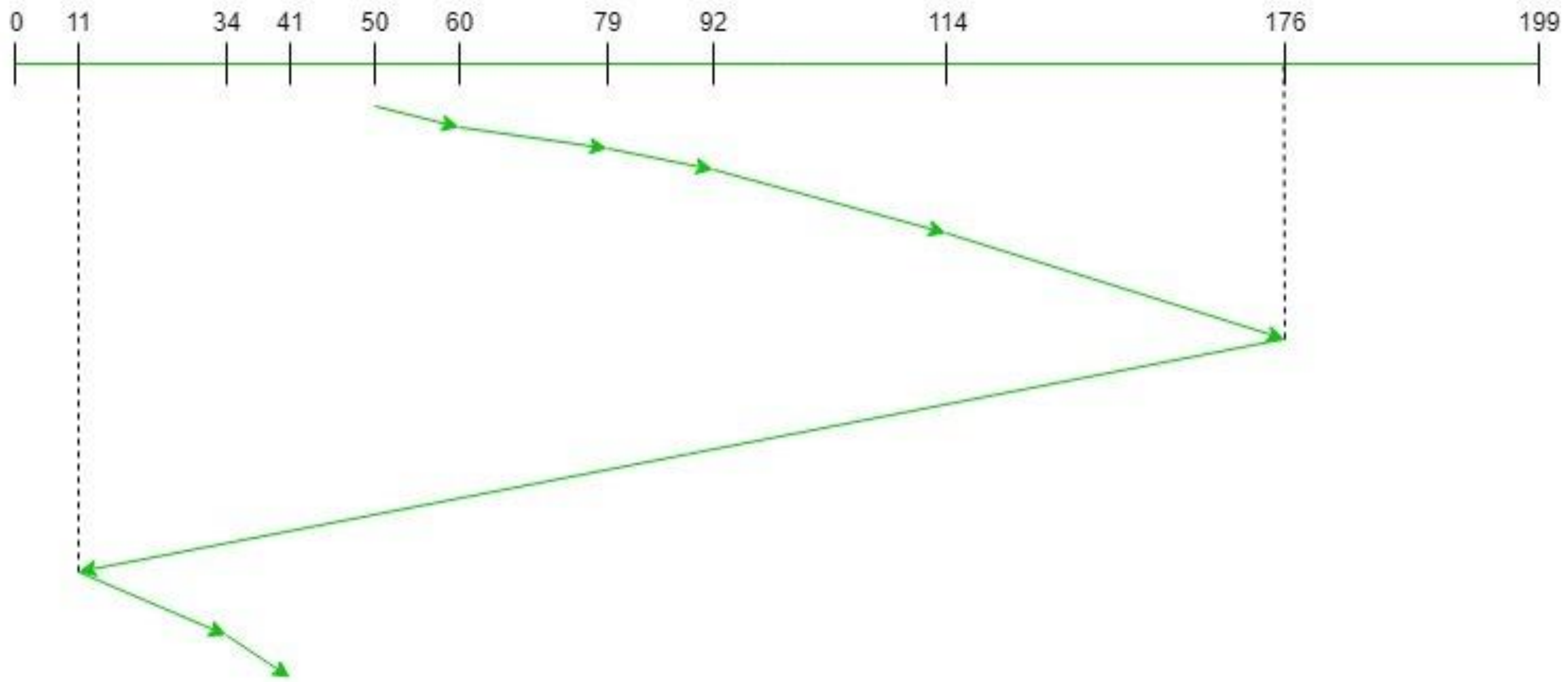
Total Seek Time = (190-50) + (190-16) = 314
Seek Sequence:

C-Look Algorithm

Request sequence = {176, 79, 34, 60, 92, 11, 41, 114}

Initial head position = 50

Direction = Right



total seek count = (60 - 50) + (79 - 60) + (92 - 79) + (114 - 92) + (176 - 114) + (176 - 11) + (34 - 11) + (41 - 34) = 321

Seek Sequence is 60, 79 , 92 , 114 , 176, 11 , 34, 41

Disk formatting

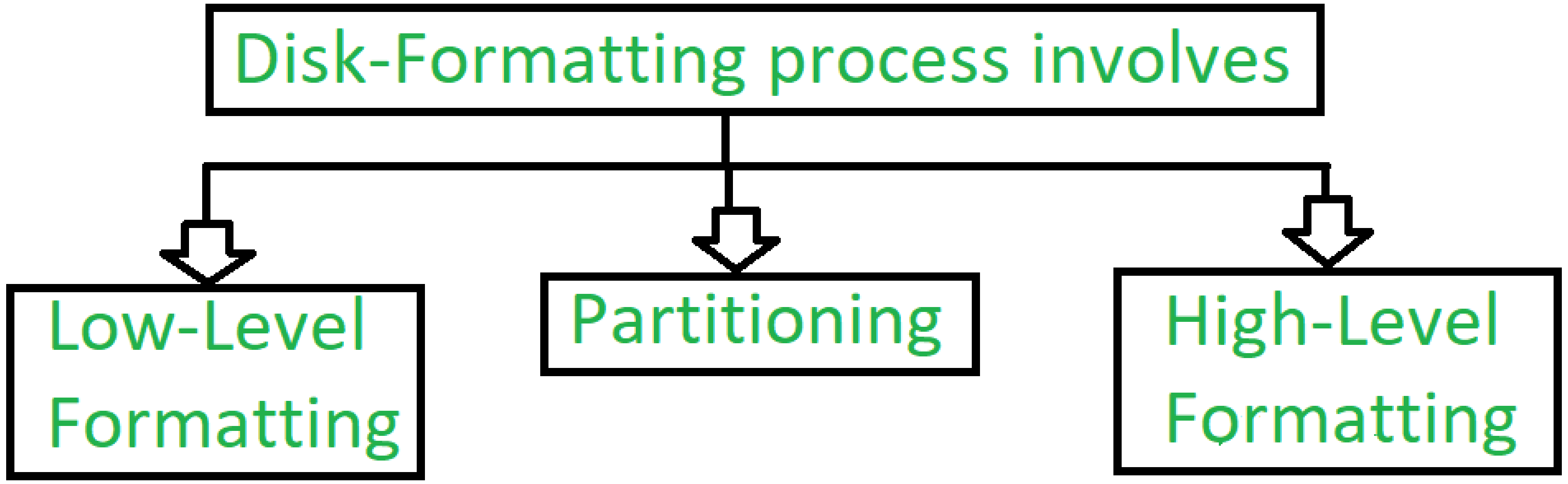
Disk formatting is like giving a blank slate to a storage device, preparing it to store new data by erasing any existing information, and creating a structure for organizing files.

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.

Types of Disk formatting

ss



Disk-Formatting process involves

Low-Level Formatting

Partitioning

High-Level Formatting

Low level formatting

Low-level formatting is the process of marking out cylinders and tracks for a blank hard disk, and then dividing tracks into multiple sectors with sector markers.

This process is actually a kind of physical formatting. And it is now often performed by hard disk manufacturers.

If users perform low-level formatting when data have been installed, all existing files will be erased, and it is almost impossible to recover them.

High level formatting

-High-level formatting is the process of creating a file system and directory structure on a disk, allowing it to be recognized and used by an operating system.

-This type of formatting is usually performed by users when they first acquire a new disk, or when they want to erase the data on an existing disk and start fresh.

Partitioning

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.

RAID (Redundant Arrays of Independent Disks)

RAID is a technique that makes use of a combination of multiple disks instead of using a single disk for increased performance, data redundancy, or both.

Key Evaluation Points:

Reliability: How many disk faults can the system tolerate?

Availability: How available is the system for actual use?

Performance: How good is the response time? How high is the throughput (rate of processing work)?

Capacity: How much useful capacity is available to the user?

RAID Levels

RAID-0 (Stripping)

RAID-1 (Mirroring)

RAID-2 (Bit-Level Stripping with Dedicated Parity)

RAID-3 (Byte-Level Stripping with Dedicated Parity)

RAID-4 (Block-Level Stripping with Dedicated Parity)

RAID-5 (Block-Level Stripping with Distributed Parity)

RAID-6 (Block-Level Stripping with two Parity Bits)

RAID-0 (Stripping)

-Blocks are “stripped” across disks. Data are split into disks.

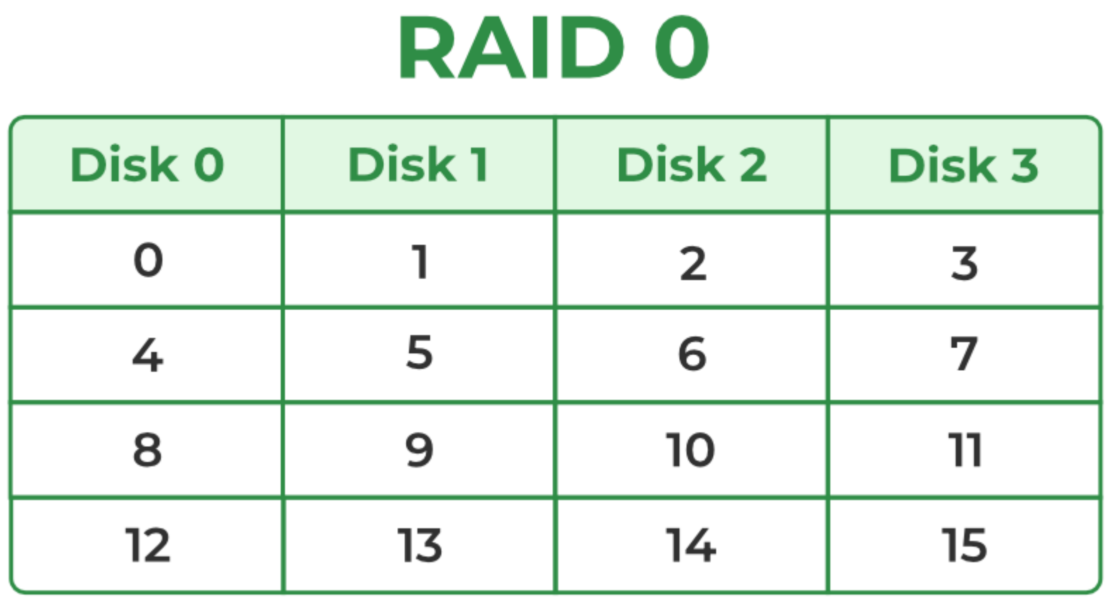
Advantages

-Ensures performance, It is easy to implement.

-It utilizes the storage capacity in a better way.

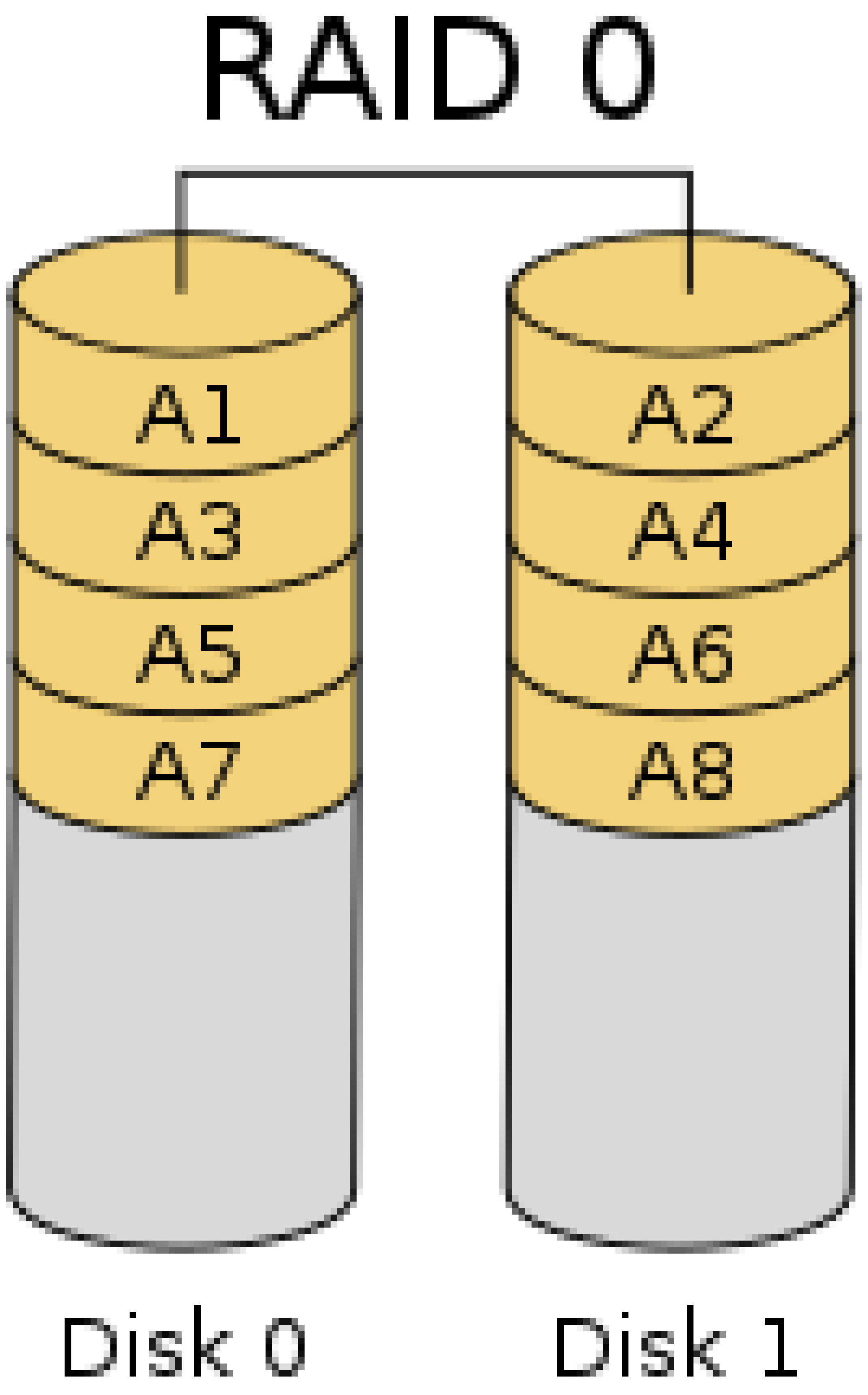
Disadvantages

A single drive loss can result in the complete failure of the system.



RAID 0

Disk 0	Disk 1	Disk 2	Disk 3
0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15



RAID-1 (Mirroring)

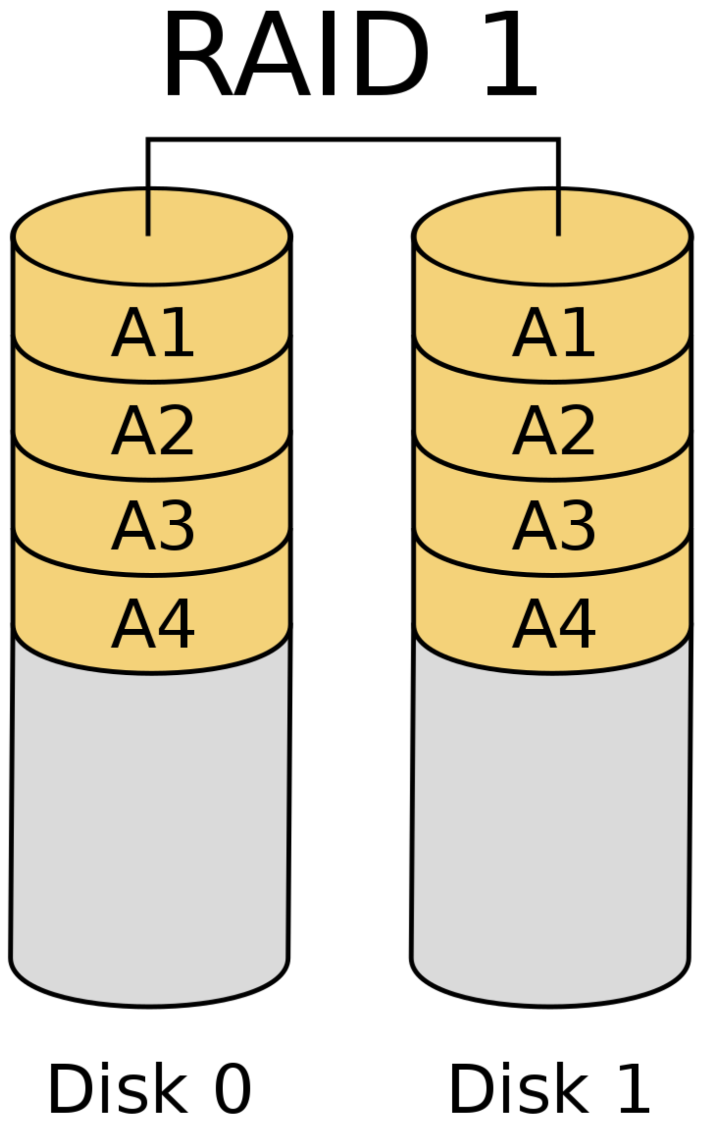
More than one copy of each block is stored in a separate disk. Thus, every block has two (or more) copies, lying on different disks.

Advantages

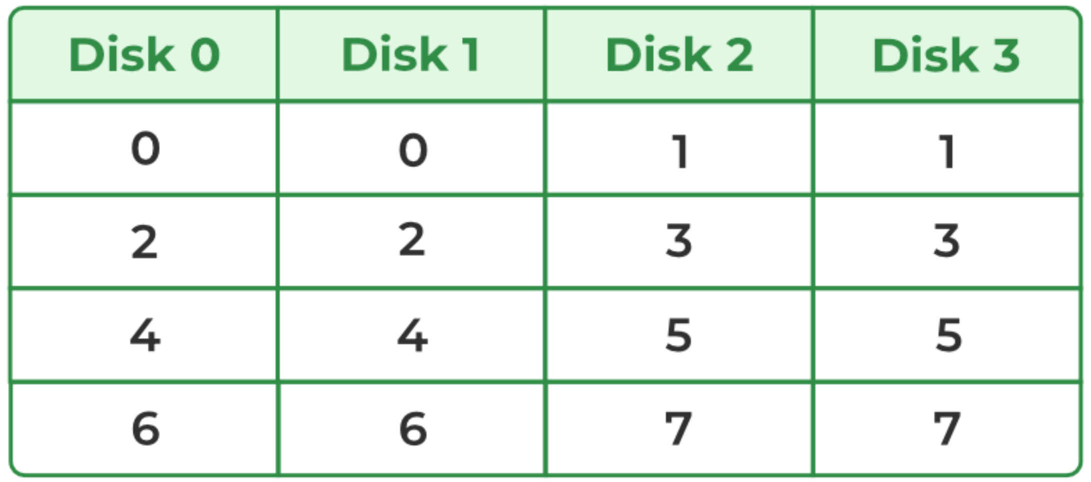
- It covers complete redundancy.
- It can increase data security and speed.

Disadvantages

- It is highly expensive.
- Storage capacity is less.



RAID 1



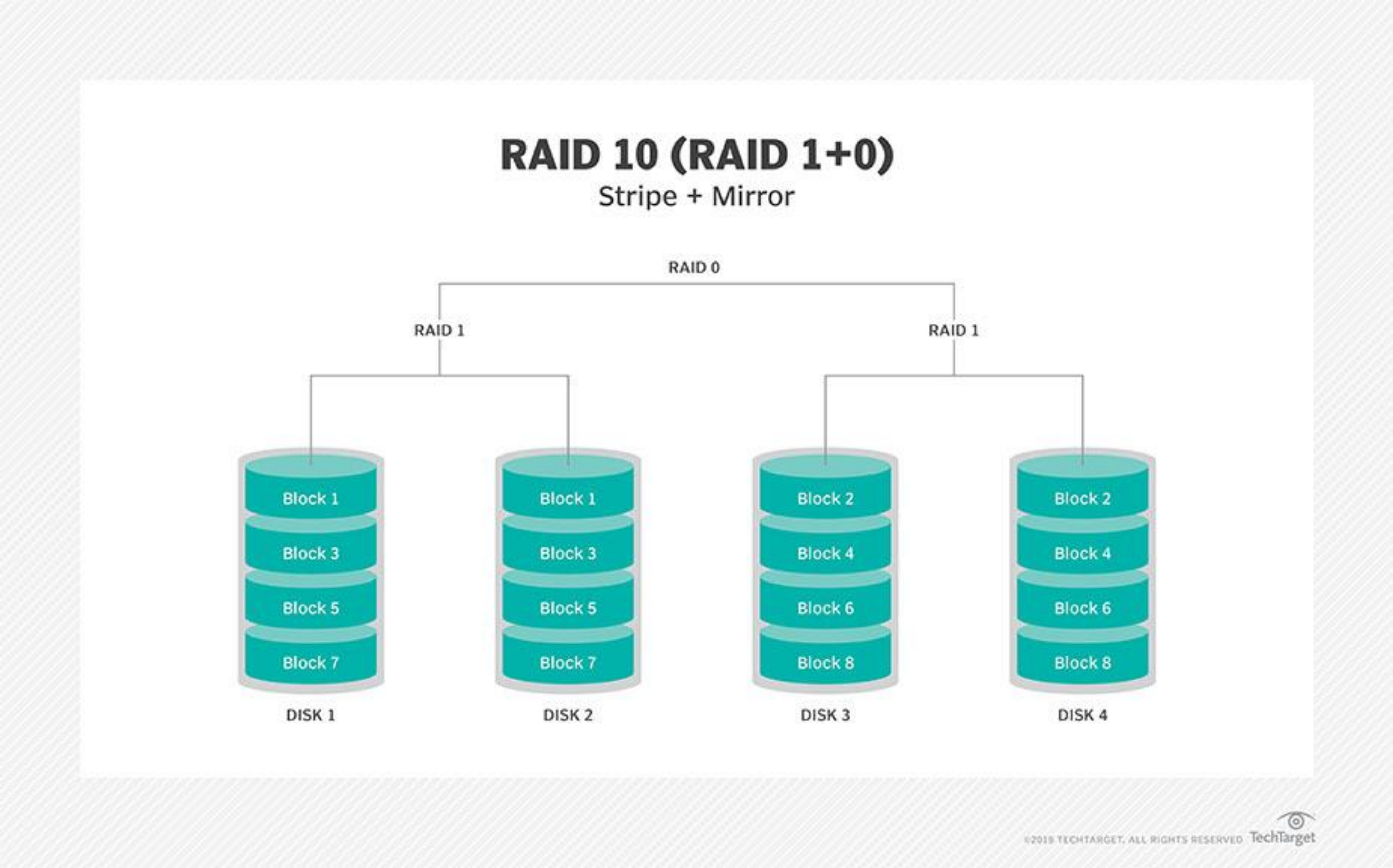
Disk 0	Disk 1	Disk 2	Disk 3
0	0	1	1
2	2	3	3
4	4	5	5
6	6	7	7

RAID-1 (Mirroring)

-Both split and mirror

Advantages

- Maintains security and performance



RAID-2 (Bit-Level Stripping with Dedicated Parity)

In Raid-2, the error of the data is checked at every bit level. Here, we use the Hamming Code Parity Method to find the error in the data.

It uses one designated drive to store parity.

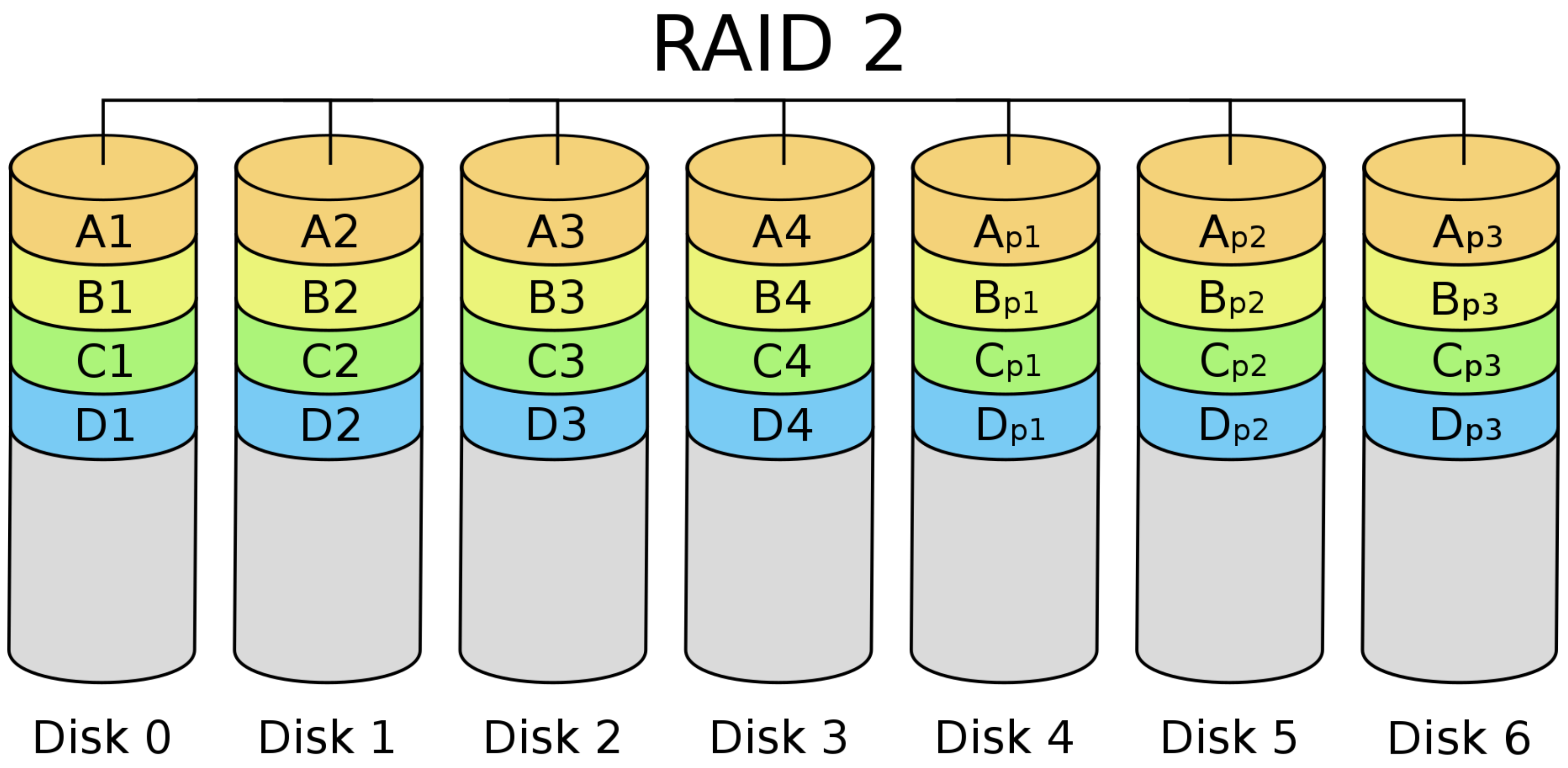
Not in use

Advantages

- It Uses one designated drive to store parity.

Disadvantages

- It has a complex structure and high cost due to extra drivee

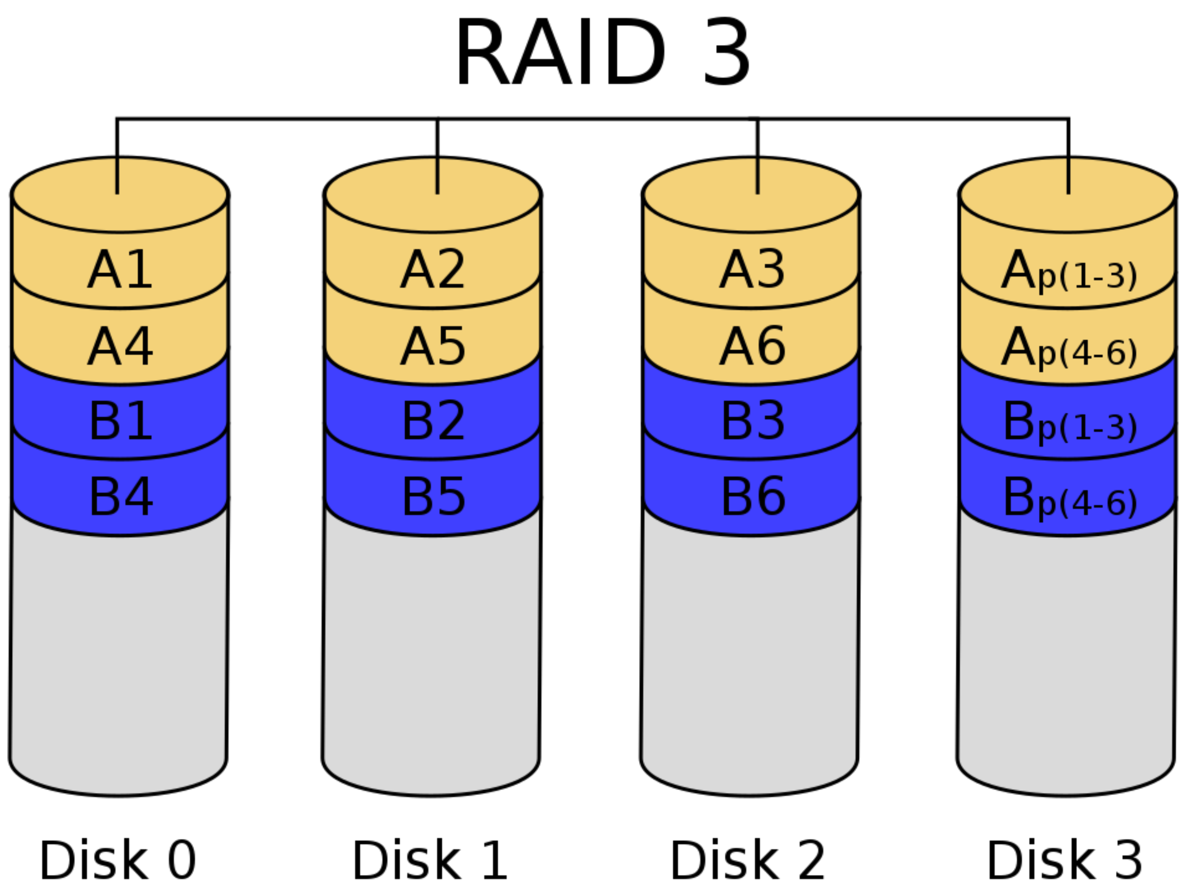


RAID-3 (Byte-Level Stripping with Dedicated Parity)

-It consists of byte-level striping with dedicated parity striping.

-Whenever failure of the drive occurs, it helps in accessing the parity drive, through which we can reconstruct the data.

-Here Disk 3 contains the Parity bits for Disk 0, Disk 1, and Disk 2. If data loss occurs, we can retrieve it with Disk 3.



Advantages

- Data can be transferred in bulk.
- Data can be accessed in parallel.

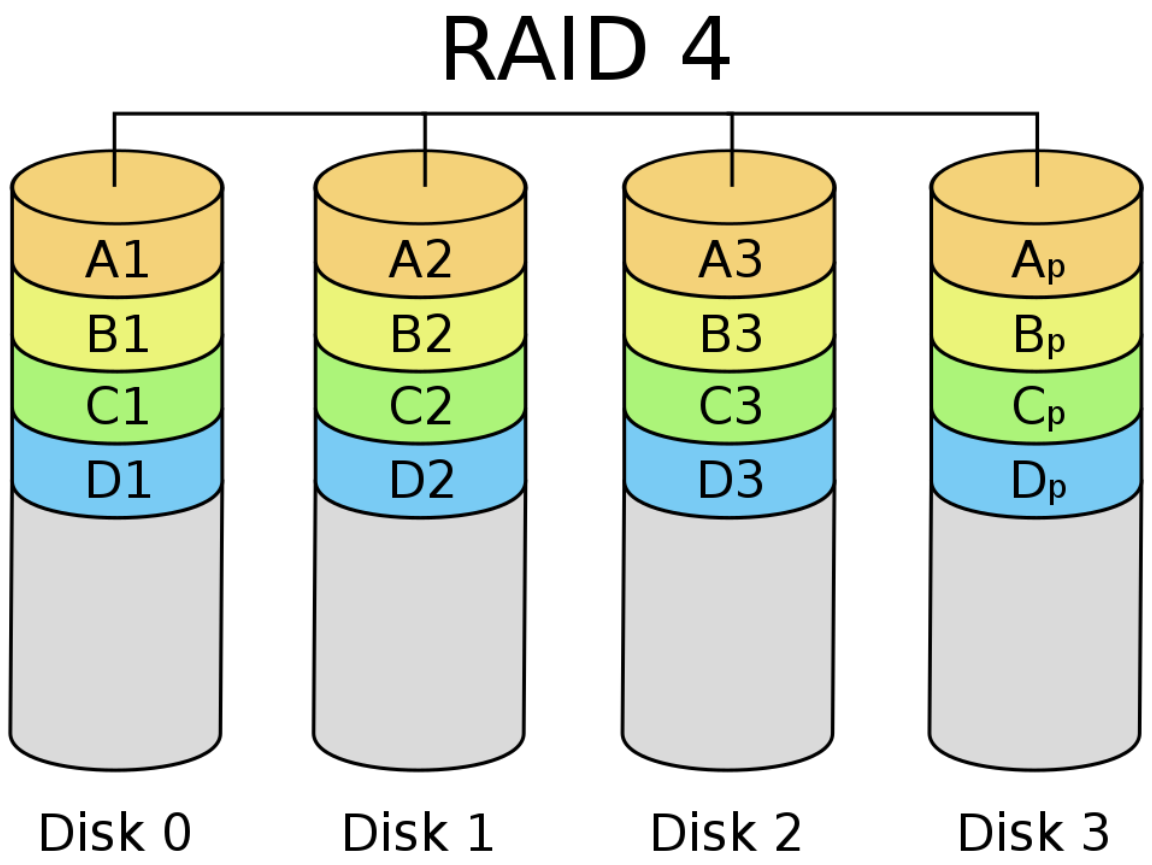
Disadvantages

- It requires an additional drive for parity

RAID-4 (Block-Level Stripping with Dedicated Parity)

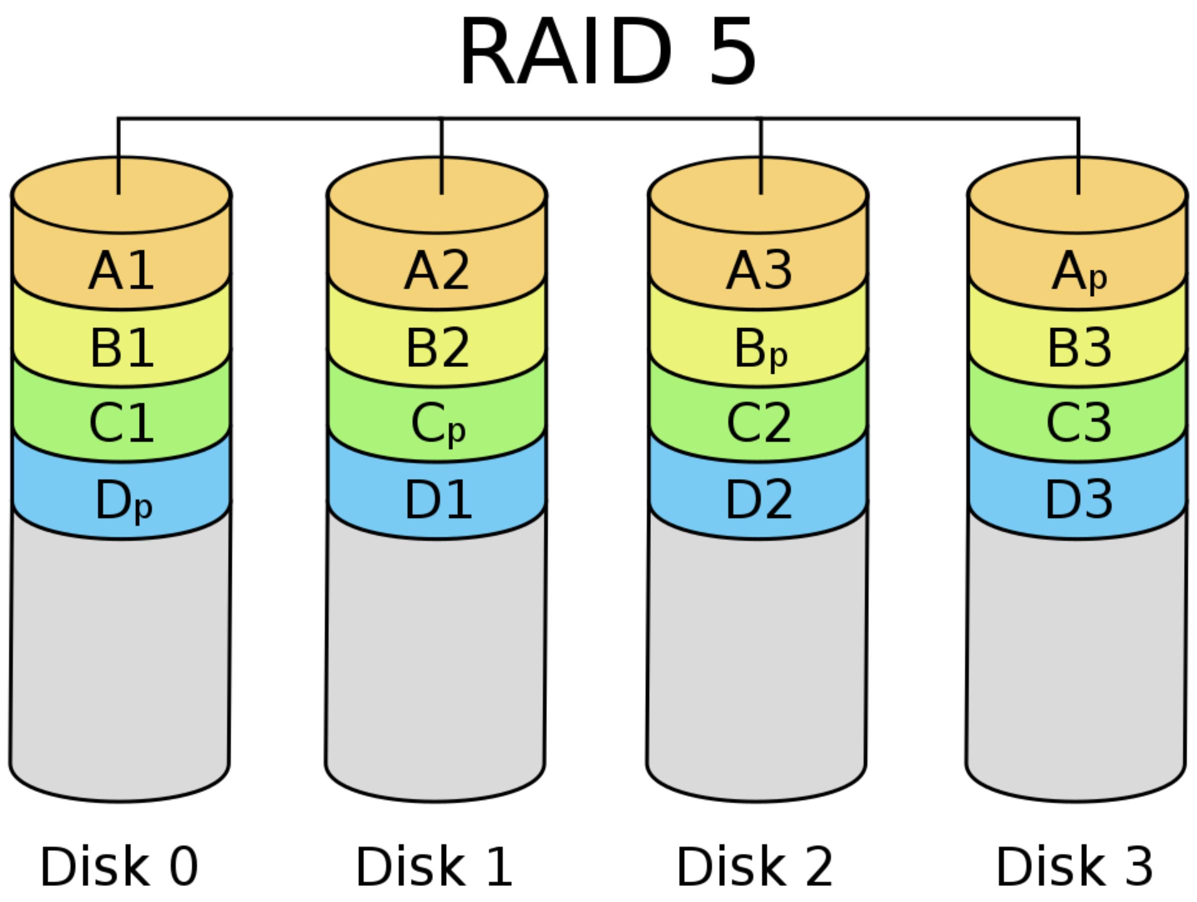
Instead of duplicating data, this adopts a parity-based approach.

RAID 3 and RAID 4 are similar



RAID-5 (Block-Level Stripping with Distributed Parity)

This is a slight modification of the RAID-4 system where the only difference is that the parity is distributed among the disks.



Advantages

- Data can be reconstructed using parity bits.
- It makes the performance better.

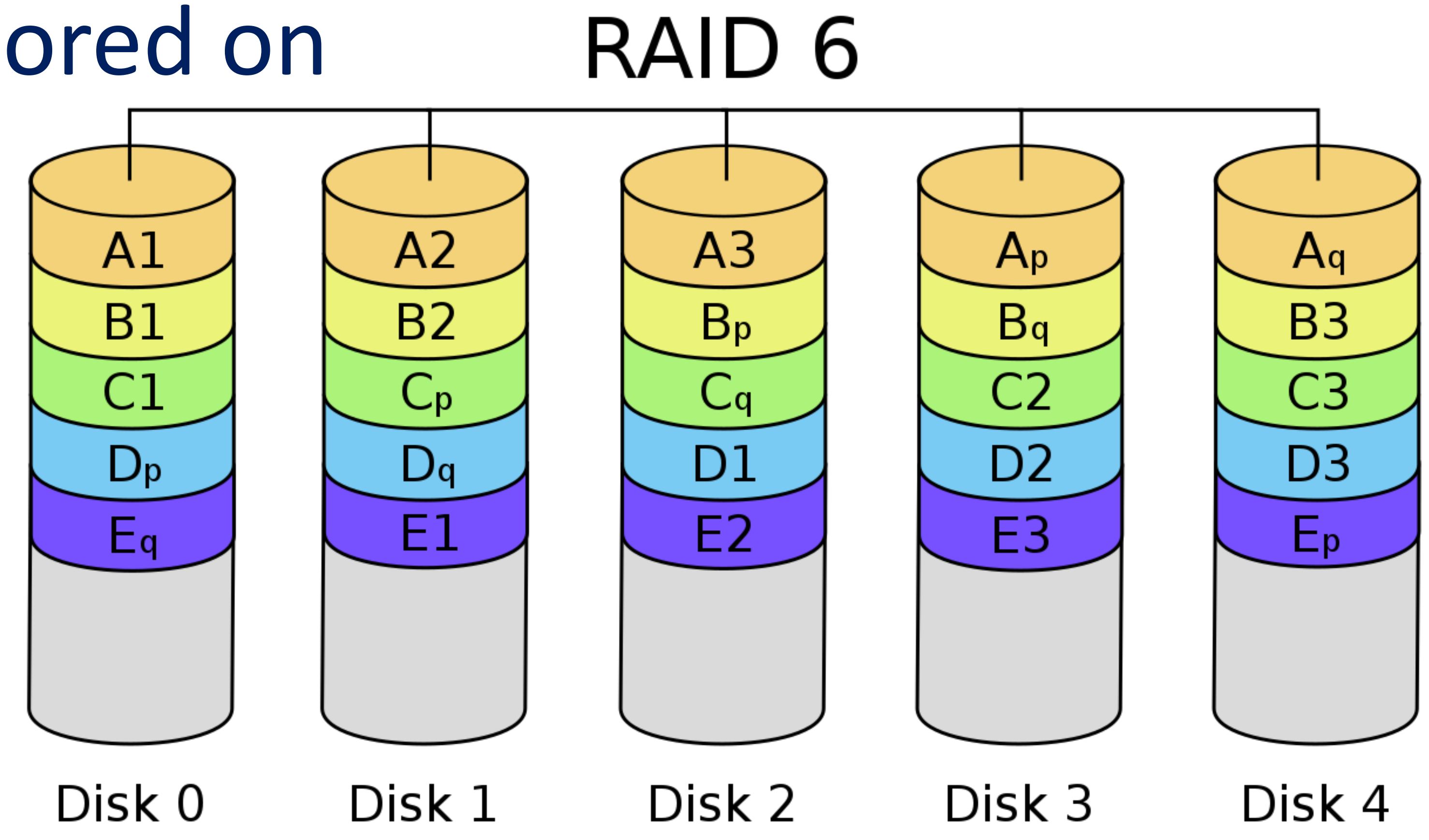
Disadvantages

- Its technology is complex and extra space is required.
- If both discs get damaged, data will be lost forever.

RAID-6 (Block-Level Stripping with two Parity Bits)

-Maintains two parity.

-Raid-6 helps when there is more than one disk failure. A pair of independent parities are generated and stored on multiple disks at this level.



Advantages

- Very high data Accessibility.
- Fast read data transactions.

Disadvantages

- Due to double parity, it has slow write data transactions.
- Extra space is required.

Find me

☎ 9851083215

✉ Santosh.it288@mail.com

🌐 www.phtechno.com

📍 Kathmandu