Device Management

Unit-4 Part -II

So far…

- We have covered CPU and memory management
- Computing is not interesting without I/Os
- ***Device management:*** the OS component that manages hardware devices
  - Provides a uniform interface to access devices with different physical characteristics
  - Optimizes the performance of individual devices

Basics of I/O Devices

- Three categories
  - A ***block device*** stores information in fixed-size blocks, each one with its own address
    - e.g., disks
  - A ***character device*** delivers or accepts a stream of characters, and individual characters are not addressable
    - e.g., keyboards, printers
  - A ***network device*** transmit data packets

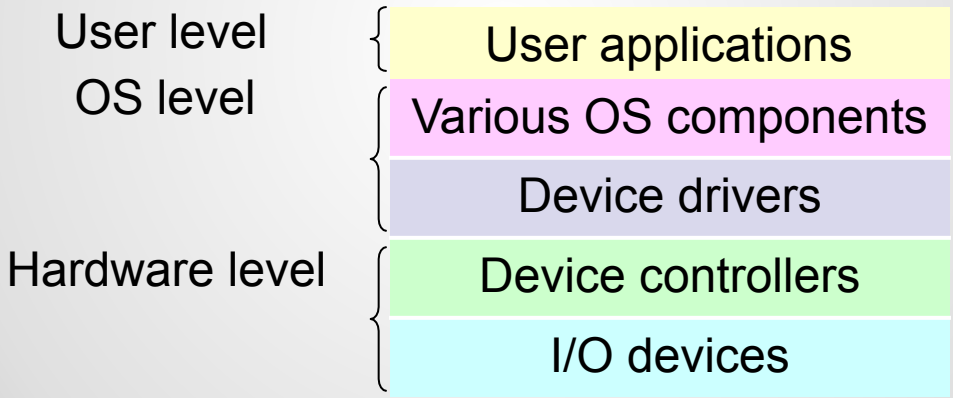
Device Controller

- Converts between serial bit stream and a block of bytes
- Performs error correction if necessary
- Components:
  - Device registers to communicate with the CPU
  - Data buffer that an OS can read or write

Device Driver

- An OS component that is responsible for hiding the complexity of an I/O device
- So that the OS can access various devices in a uniform manner

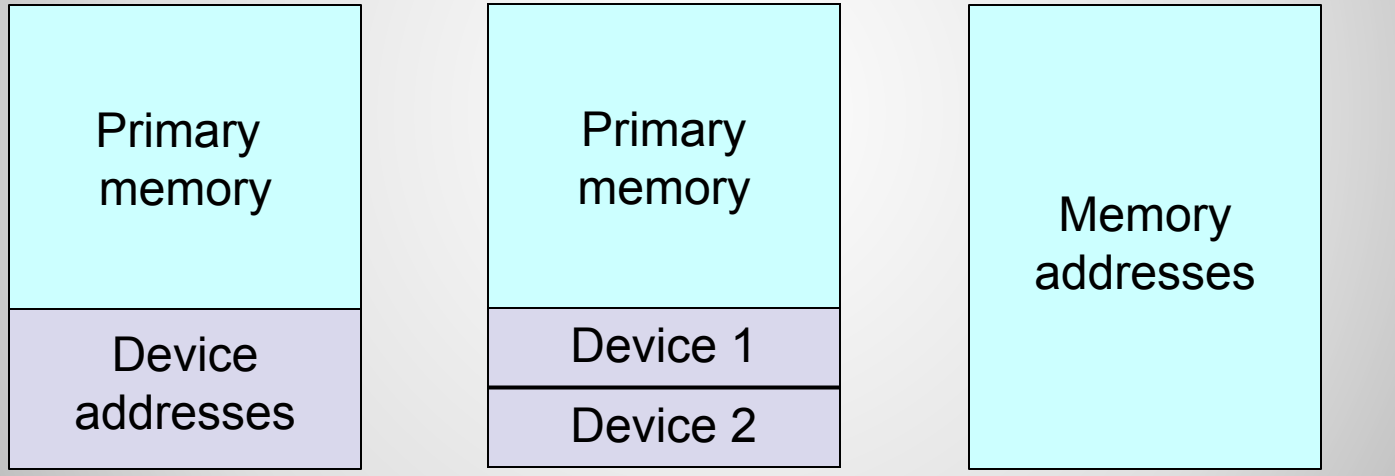
Device Driver Illustrated



Device Addressing

- Two approaches
  - Dedicated range of device addresses in the physical memory
    - Requires special hardware instructions associated with individual devices
  - ***Memory-mapped I/O:*** makes no distinction between device addresses and memory addresses
    - Devices can be access the same way as normal memory, with the same set of hardware instructions

Device Addressing Illustrated



Separate device addresses

Memory-mapped I/Os

Ways to Access a Device

- ***Polling:*** a CPU repeatedly checks the status of a device for exchanging data
  + Simple
  - Busy-waiting

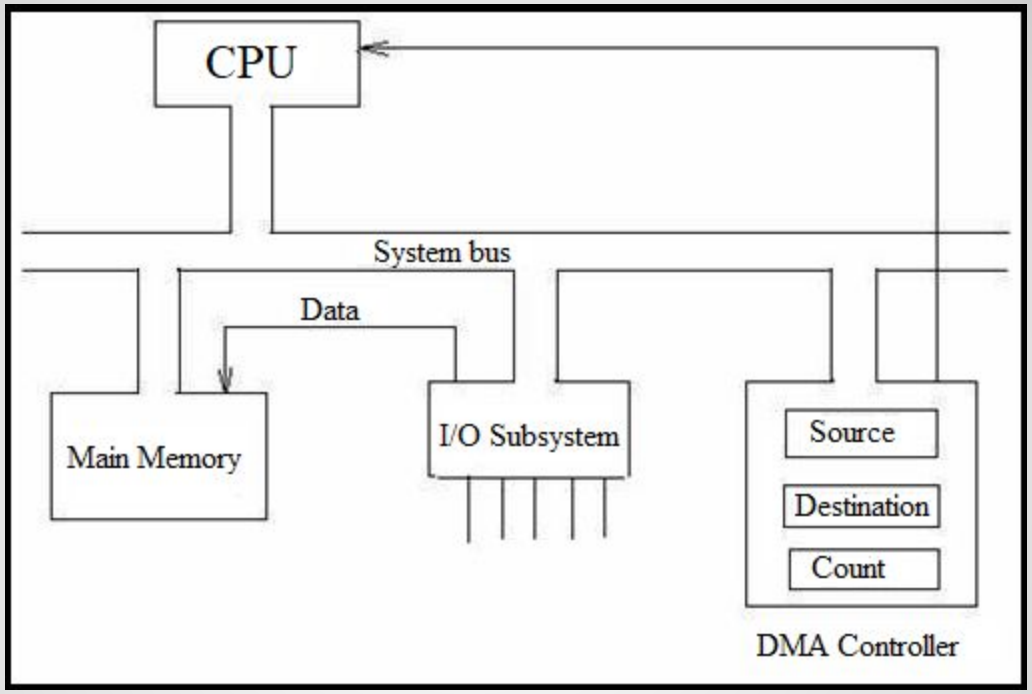
Ways to Access a Device

- ***Interrupt-driven I/Os:*** A device controller notifies the corresponding device driver when the device is available
  + More efficient use of CPU cycles
  - Data copying and movements
  - Slow for character devices (i.e., one interrupt per keyboard input)

Ways to Access a Device

- ***Direct memory access (DMA):*** uses an additional controller to perform data movements
  + CPU is not involved in copying data
  - A process cannot access in-transit data

- **How Does Direct Memory Access Work?**



To perform input, output, or memory-to-memory operations, the host processor initializes the DMA controller with the number of words to transfer and the memory address to use.

Modes of Operation

- **Burst Mode-**In burst mode, the full data block is transmitted in a continuous sequence.
- **Cycle Stealing Mode -**The cycle stealing mode is used in a system where the CPU cannot be disabled for the length of time required for the burst transfer mode.
- **transparent Mode**

The transparent mode takes the longest time to transfer data blocks, but it is also the most efficient mode in terms of overall system performance. In transparent mode, the Direct Memory Access controller transfers data only when the CPU performs operations that do not use the system buses.

Disk Structure

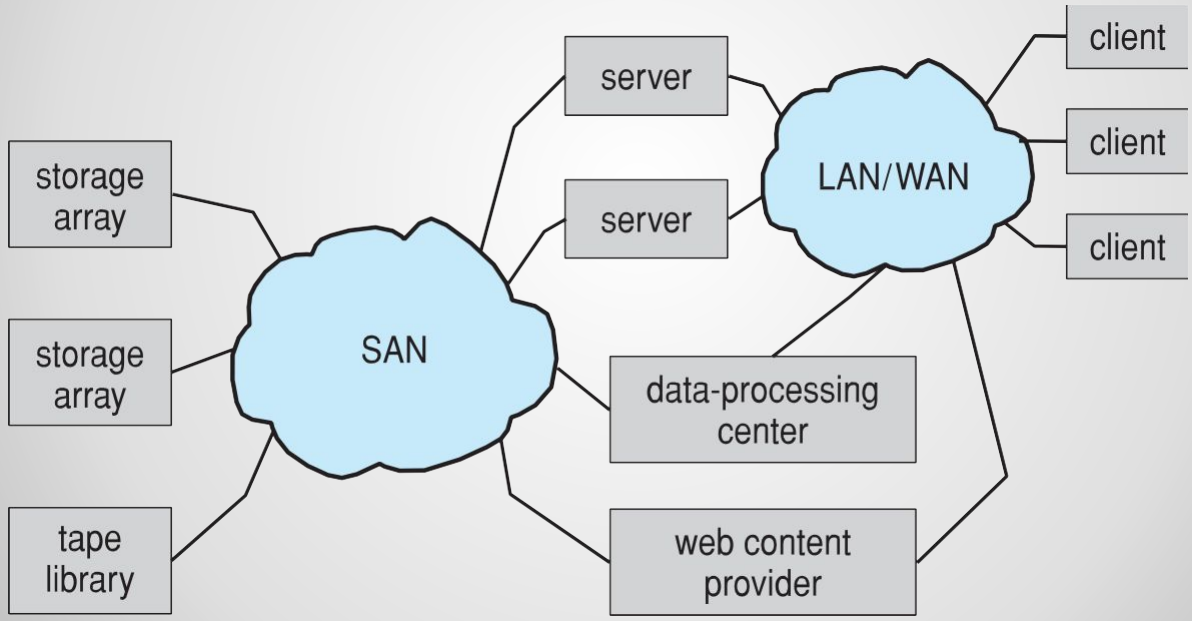
- Disk drives are addressed as large 1-dimensional arrays of **logical blocks**, where the logical block is the smallest unit of transfer
  - Low-level formatting creates **logical blocks** on physical media
- The 1-dimensional array of logical blocks is mapped into the sectors of the disk sequentially
  - Sector 0 is the first sector of the first track on the outermost cylinder
  - Mapping proceeds in order through that track, then the rest of the tracks in that cylinder, and then through the rest of the cylinders from outermost to innermost
  - Logical to physical address should be easy
    - Except for bad sectors
    - Non-constant # of sectors per track via constant angular velocity

Disk Attachment

- Host-attached storage accessed through I/O ports talking to I/O busses
- SCSI itself is a bus, up to 16 devices on one cable, **SCSI initiator** requests operation and **SCSI targets** perform tasks
  - Each target can have up to 8 **logical units** (disks attached to device controller)
- FC is high-speed serial architecture
  - Can be switched fabric with 24-bit address space – the basis of **storage area networks (SANs)** in which many hosts attach to many storage units
- I/O directed to bus ID, device ID, logical unit (LUN)

Storage Area Network

- Common in large storage environments
- Multiple hosts attached to multiple storage arrays - flexible

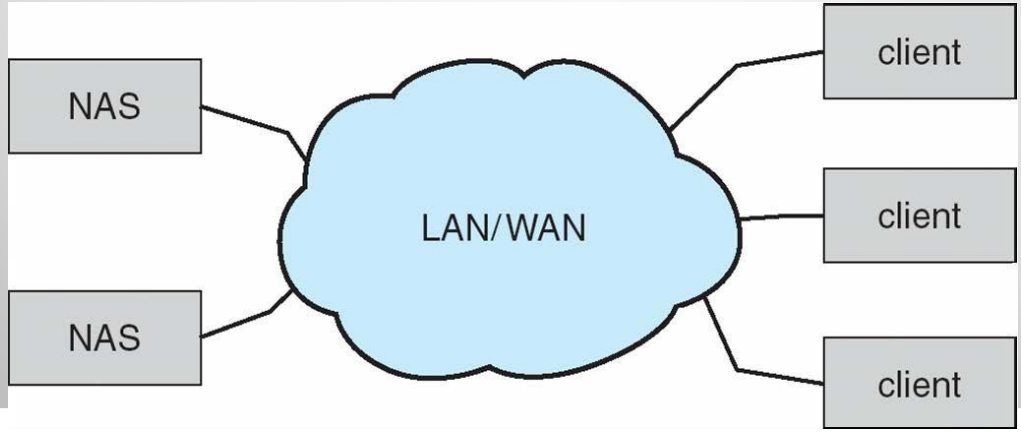


Storage Area Network (Cont.)

- SAN is one or more storage arrays
  - Connected to one or more Fibre Channel switches
- Hosts also attach to the switches
- Storage made available via **LUN Masking** from specific arrays to specific servers
- Easy to add or remove storage, add new host and allocate it storage
  - Over low-latency Fibre Channel fabric
- Why have separate storage networks and communications networks?
  - Consider iSCSI, FCOE

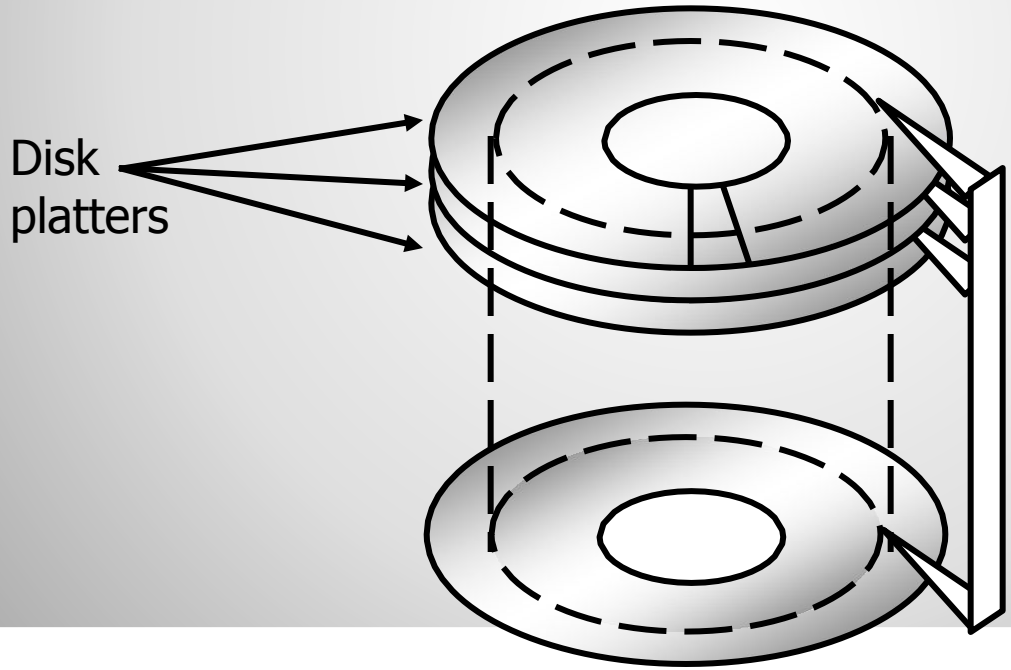
Network-Attached Storage

- Network-attached storage (**NAS**) is storage made available over a network rather than over a local connection (such as a bus)
  - Remotely attaching to file systems
- NFS and CIFS are common protocols
- Implemented via remote procedure calls (RPCs) between host and storage over typically TCP or UDP on IP network
- **iSCSI** protocol uses IP network to carry the SCSI protocol
  - Remotely attaching to devices (blocks)



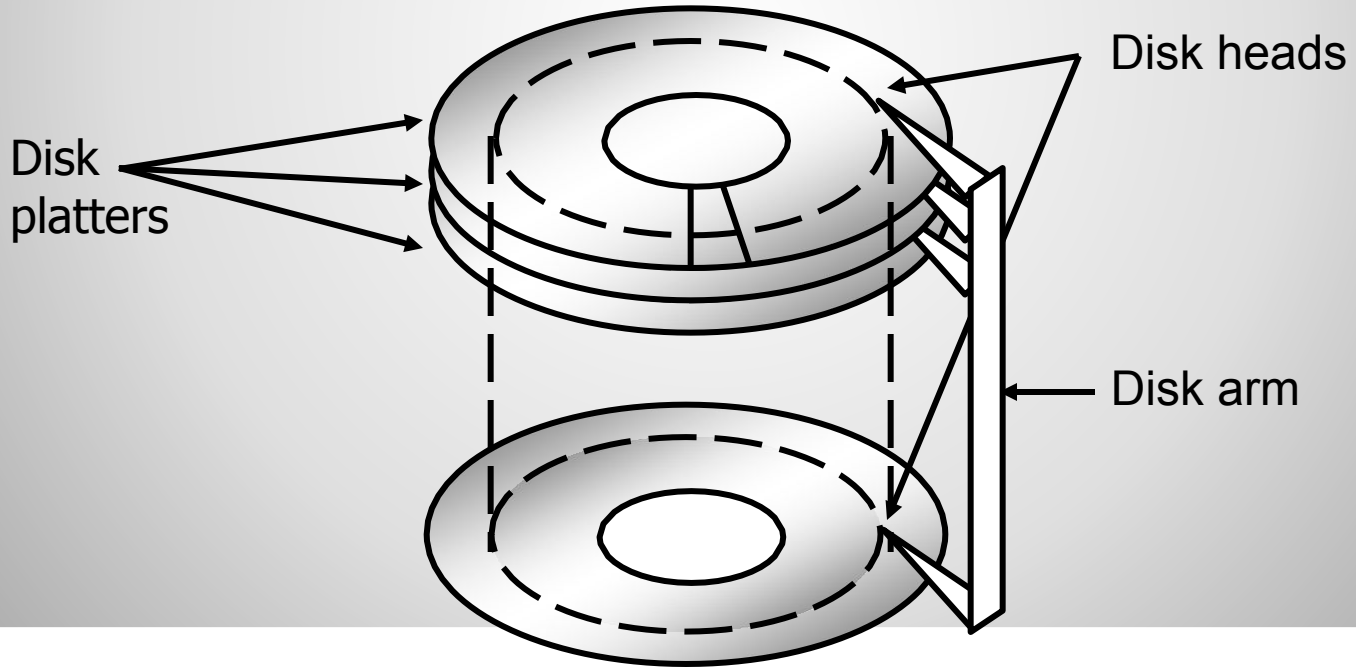
Disk Characteristics

- ***Disk platters:*** coated with magnetic materials for recording



Disk Characteristics

- ***Disk arm:*** moves a comb of ***disk heads***
  - Only one disk head is active for reading/writing

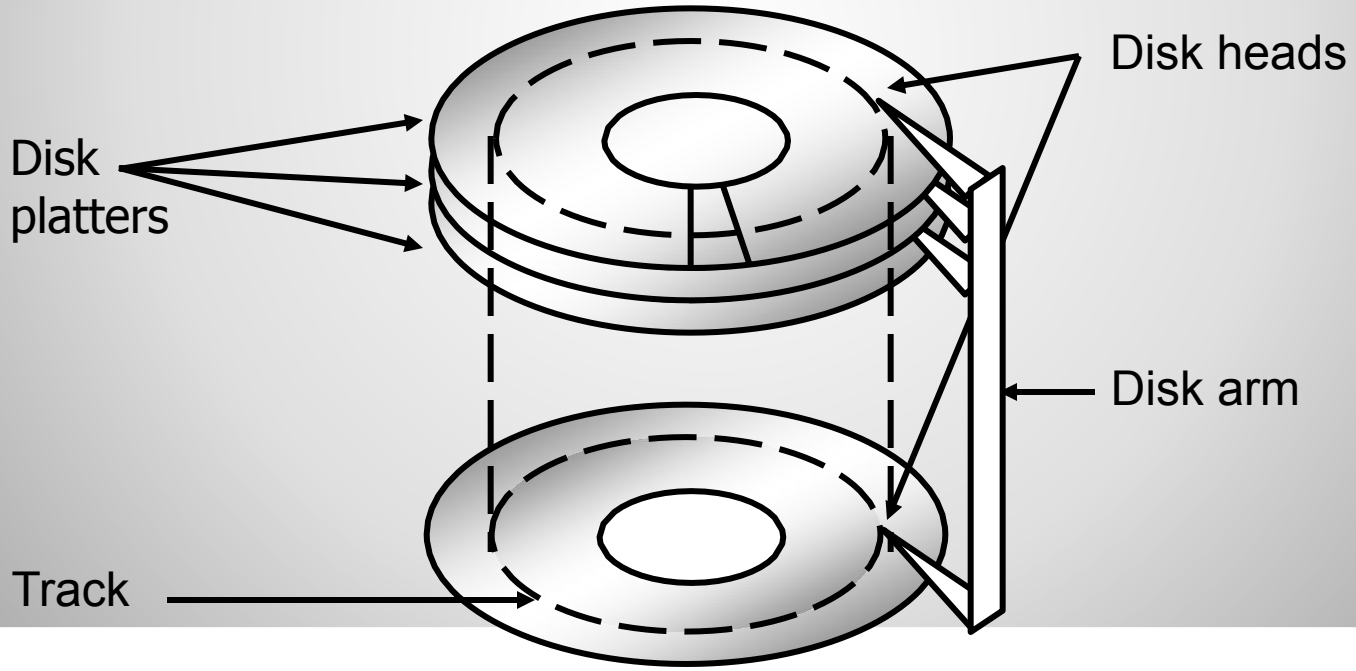


Hard Disk Trivia…

- Aerodynamically designed to fly
  - As close to the surface as possible
  - No room for air molecules
- Therefore, hard drives are filled with special inert gas
- If head touches the surface
  - Head crash
  - Scrapes off magnetic information

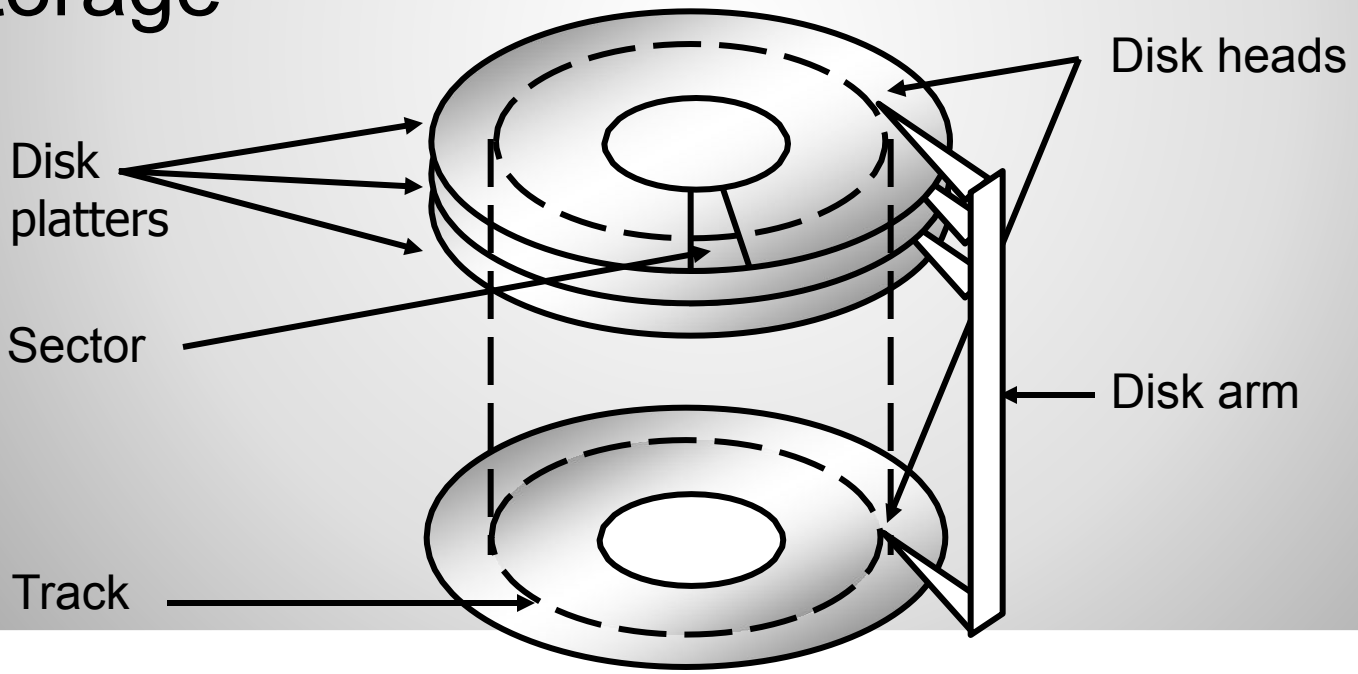
Disk Characteristics

- Each disk platter is divided into concentric *tracks*



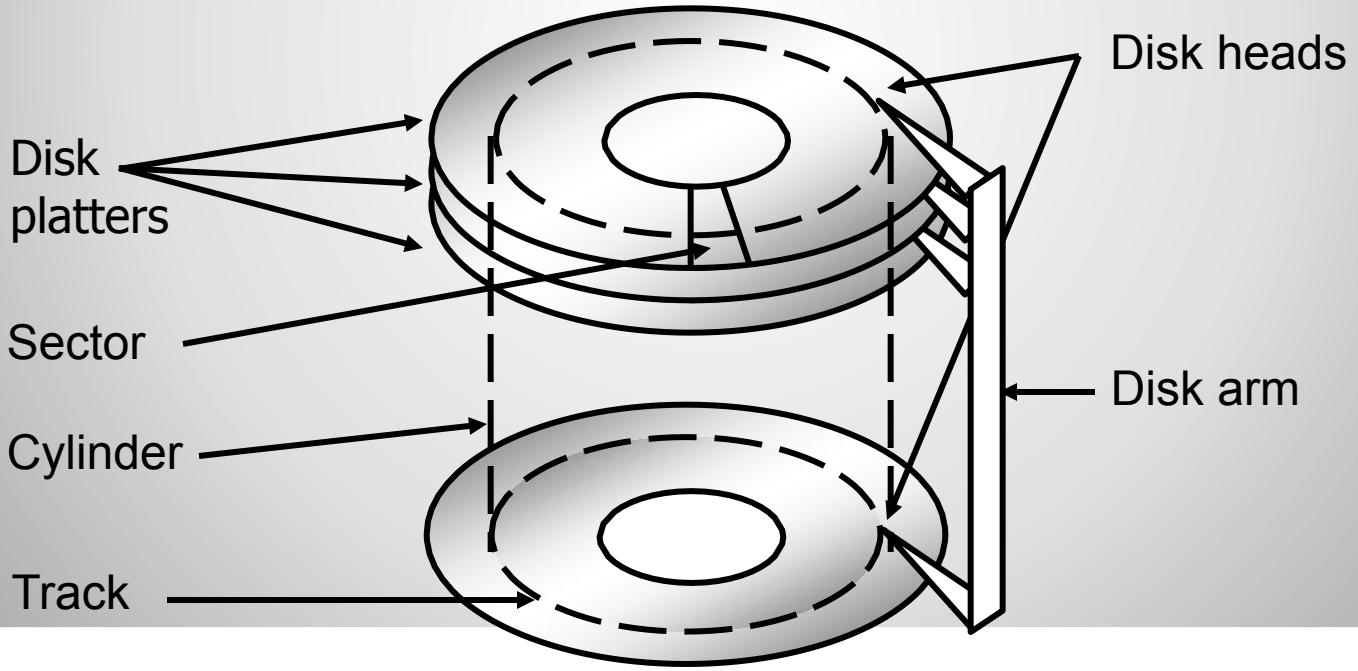
Disk Characteristics

- A track is further divided into ***sectors.*** A sector is the smallest unit of disk storage



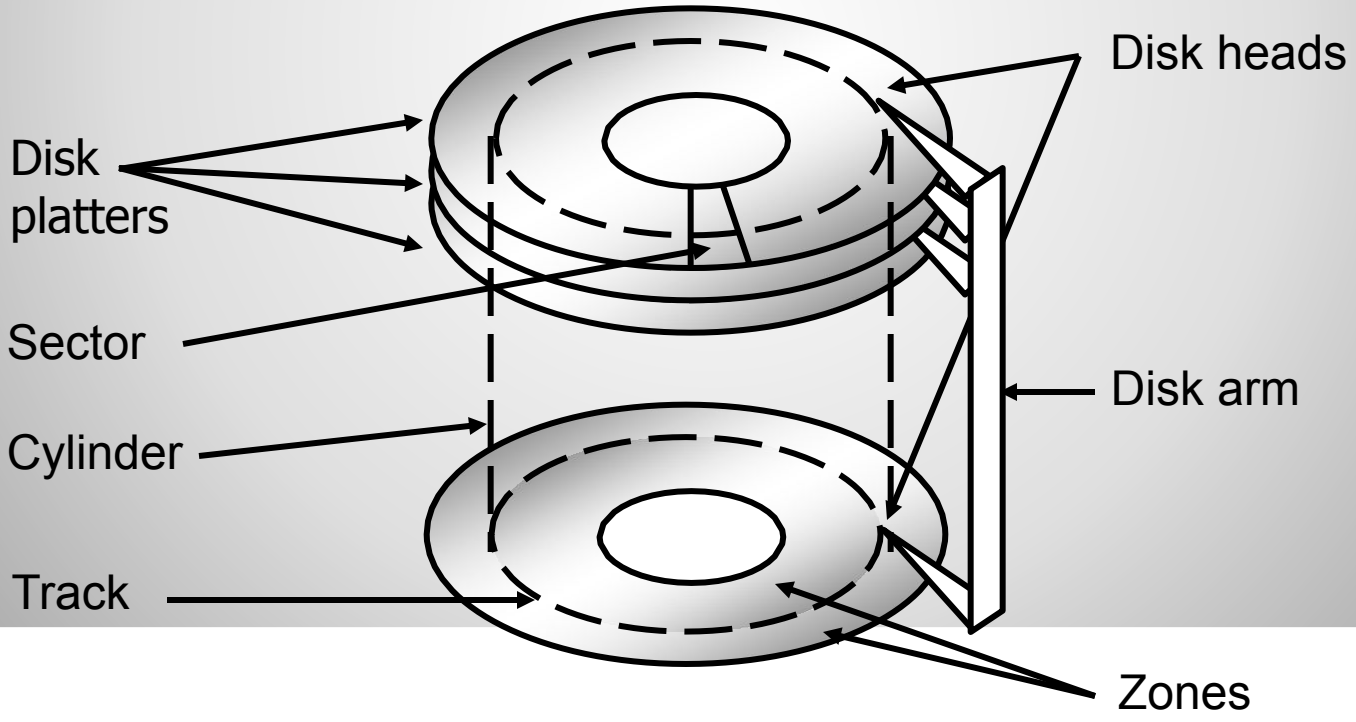
Disk Characteristics

- A ***cylinder*** consists of all tracks with a given disk arm position



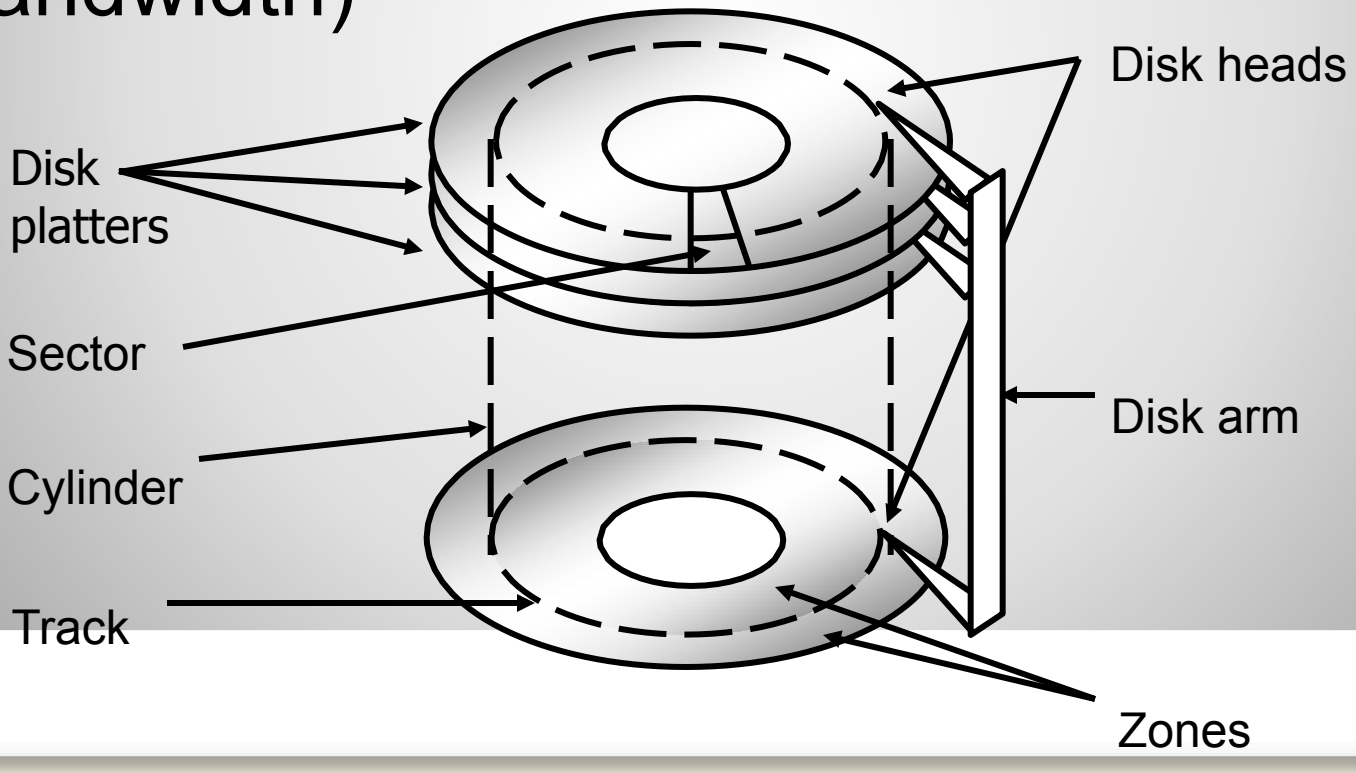
Disk Characteristics

- Cylinders are further divided into ***zones***



Disk Characteristics

- ***Zone-bit recording***: zones near the edge of a disk store more information (higher bandwidth)



Disk Access Time

- ***Seek time:*** the time to position disk heads (~4 msec on average)
- ***Rotational latency:*** the time to rotate the target sector to underneath the head
  - Assume 7,200 rotations per minute (RPM)
  - 7,200 / 60 = 120 rotations per second
  - 1/120 = ~8 msec per rotation
  - Average rotational delay is ~4 msec

Disk Access Time

- ***Transfer time:*** the time to transfer bytes
  - Assumptions:
    - 58 Mbytes/sec
    - 4-Kbyte disk blocks
  - Time to transfer a block takes 0.07 msec

- ***Disk access time***
  - Seek time + rotational delay + transfer time

Disk Performance Metrics

- ***Latency***
  - Seek time + rotational delay
- ***Bandwidth***
  - Bytes transferred / disk access time

Disk Tradeoffs

Sector size	Space utilization	Transfer rate
1 byte	8 bits/1008 bits (0.8%)	125 bytes/sec (1 byte / 8 msec)
4 Kbytes	4096 bytes/4221 bytes (97%)	500 Kbytes/sec (4 Kbytes / 8 msec)
1 Mbyte	(~100%)	58 Mbytes/sec (peak bandwidth)

- Larger sector size better bandwidth
- Wasteful if only 1 byte out of 1 Mbyte is needed

Disk Controller

- Few popular standards
  - IDE (integrated device electronics)
  - ATA (advanced technology attachment interface)
  - SCSI (small computer systems interface)
  - SATA (serial ATA)
- Differences
  - Performance
  - Parallelism

Disk Device Driver

- Major goal: reduce seek time for disk accesses
  - Schedule disk request to minimize disk arm movements

Disk Scheduling

- The operating system is responsible for using hardware efficiently — for the disk drives, this means having a fast access time and disk bandwidth
- Minimize seek time
- Seek time ≈ seek distance
- Disk **bandwidth** is the total number of bytes transferred, divided by the total time between the first request for service and the completion of the last transfer

Disk Arm Scheduling Policies

- ***First come, first serve (FCFS):*** requests are served in the order of arrival
  - \+ Fair among requesters
  - \- Poor for accesses to random disk blocks
- ***Shortest seek time first (SSTF):*** picks the request that is closest to the current disk arm position
  - \+ Good at reducing seeks
  - \- May result in starvation

Disk Arm Scheduling Policies

- ***SCAN:*** takes the closest request in the direction of travel (an example of elevator algorithm)

  \+ no starvation

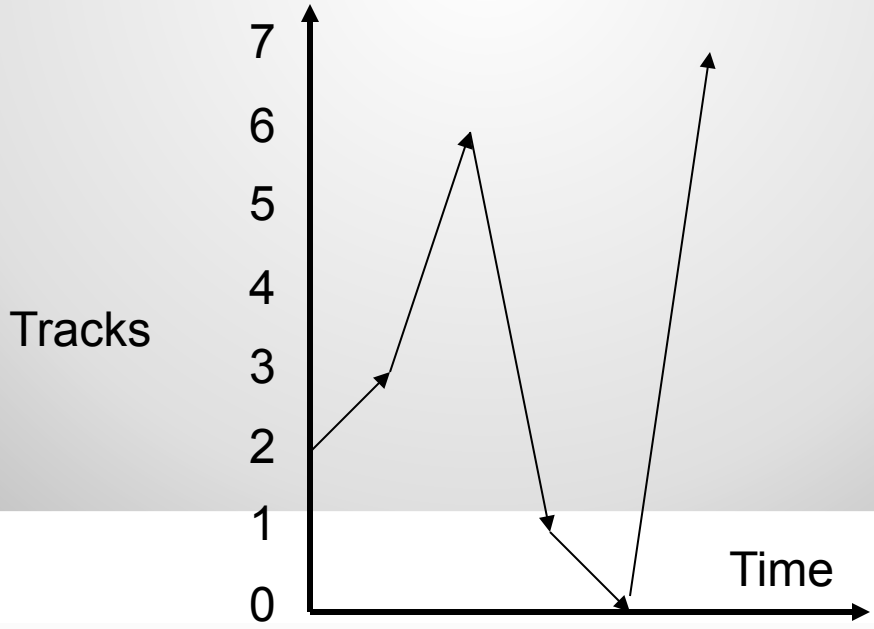
  \- a new request can wait for almost two full scans of the disk

Disk Arm Scheduling Policies

- ***Circular SCAN (C-SCAN):*** disk arm always serves requests by scanning in one direction.
  - Once the arm finishes scanning for one direction
  - Returns to the 0^{th} track for the next round of scanning

First Come, First Serve

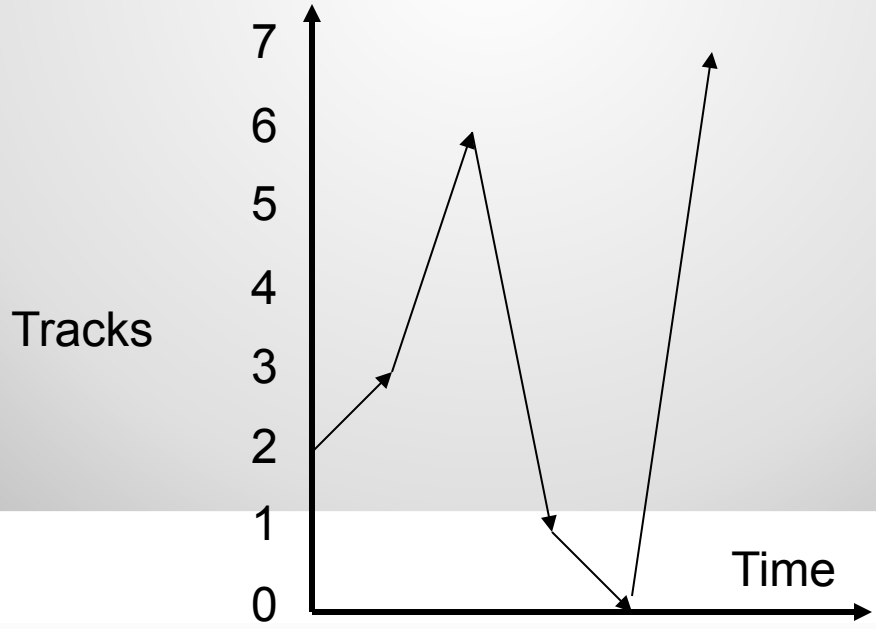
- Request queue: 3, 6, 1, 0, 7
- Head start position: 2



First Come, First Serve

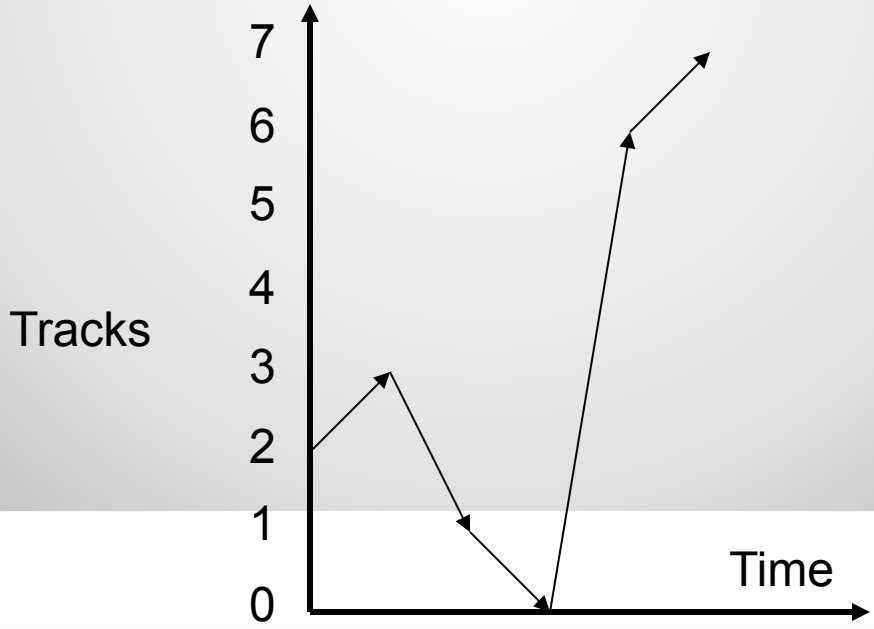
- Request queue: 3, 6, 1, 0, 7
- Head start position: 2
- Total seek distance: 1 + 3 + 5 + 1 + 7 =

17



Shortest Seek Distance First

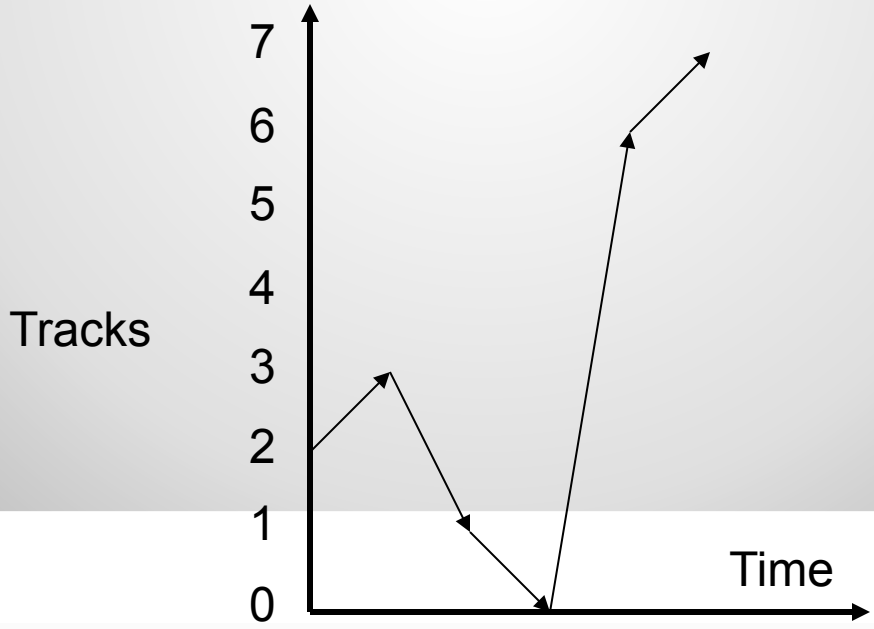
- Request queue: 3, 6, 1, 0, 7
- Head start position: 2



Shortest Seek Distance First

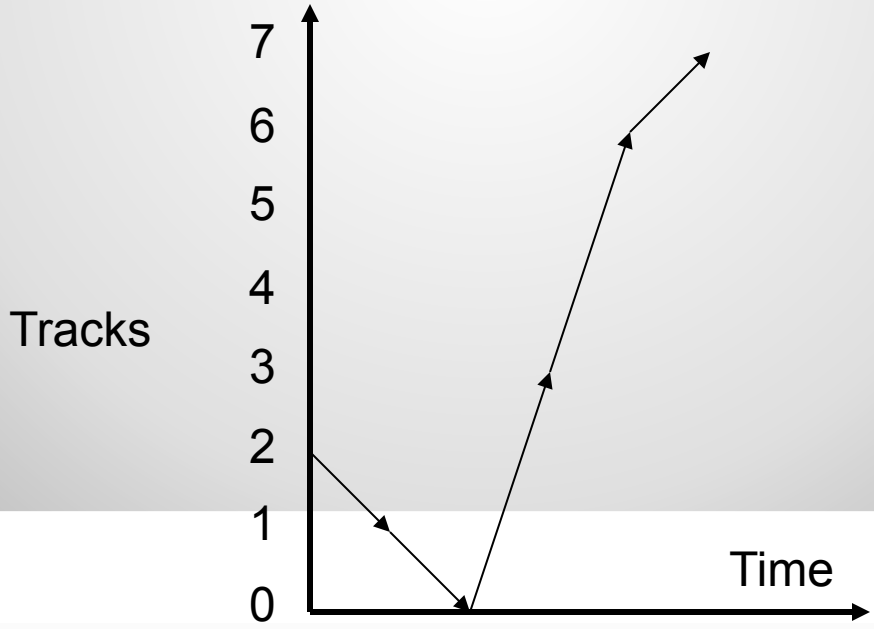
- Request queue: 3, 6, 1, 0, 7
- Head start position: 2
- Total seek distance: 1 + 2 + 1 + 6 + 1 =

10



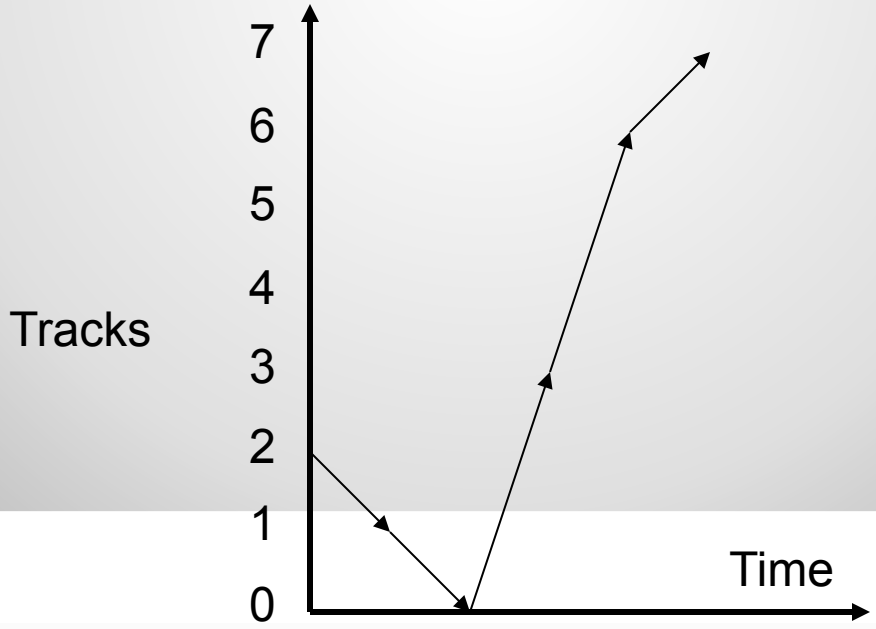
SCAN

- Request queue: 3, 6, 1, 0, 7
- Head start position: 2



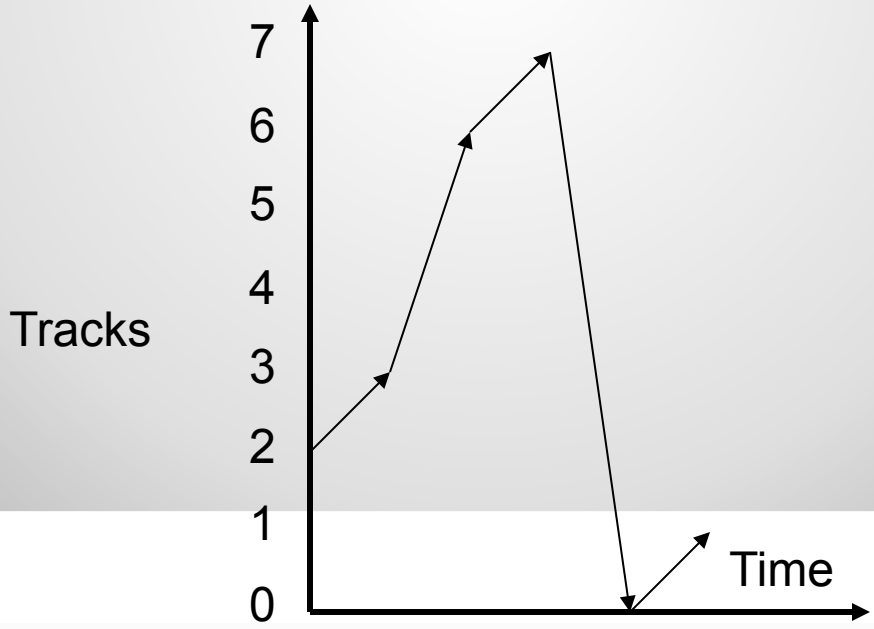
SCAN

- Request queue: 3, 6, 1, 0, 7
- Head start position: 2
- Total seek distance: 1 + 1 + 3 + 3 + 1 = 9



C-SCAN

- Request queue: 3, 6, 1, 0, 7
- Head start position: 2



C-SCAN

- Request queue: 3, 6, 1, 0, 7
- Head start position: 2
- Total seek distance: 1 + 3 + 1 + 7 + 1 =

13