Hard Disk Drives

Chapter 37

Previously in CS212…

- We talked about I/O Devices

- How they are connected
  - The bus architecture

- How can we communicate with them
  - Programmed I/O (PIO)
  - Memory Mapped I/O

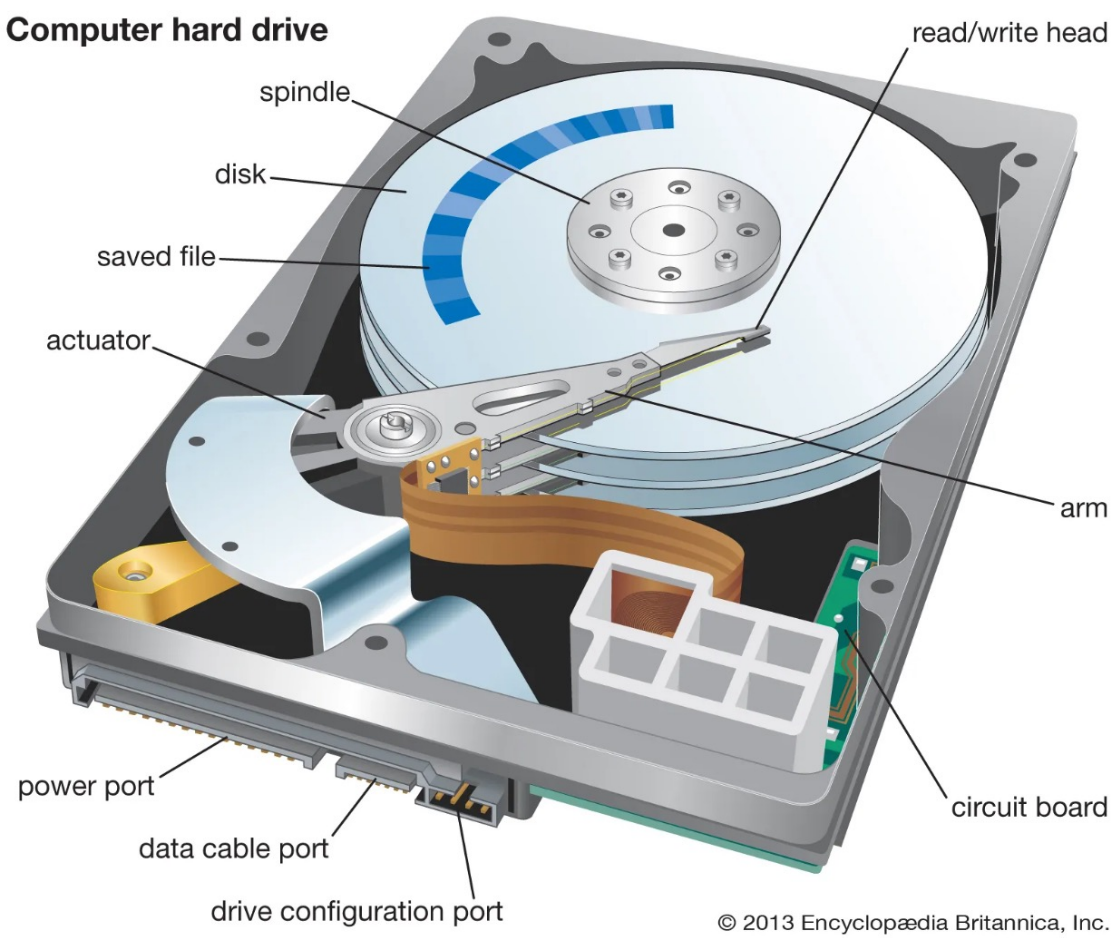
- More efficient CPU resource utilization
  - Direct Memory Access (DMA)
  - Interrupts (sometimes more efficient…)

The Hard Disk Drive (HDD) Interface

- Data is stored on the disk in **sectors**
  - Each sector is a 512-byte block of storage
- Sectors can be thought of like an array
  - A disk of size *n* has an address space of 0 to *n* – 1 sectors.
- While we tend to write more than one sector at a time (remember 4KB pages or higher)
  - Only a 512-byte write to a sector is **atomic**
  - If power fails only a portion of data may be complete (**torn write**)
- It is generally assumed that writing data in consecutive blocks is the fastest access mode
  - Sequential reads or writes

HDD Anatomy

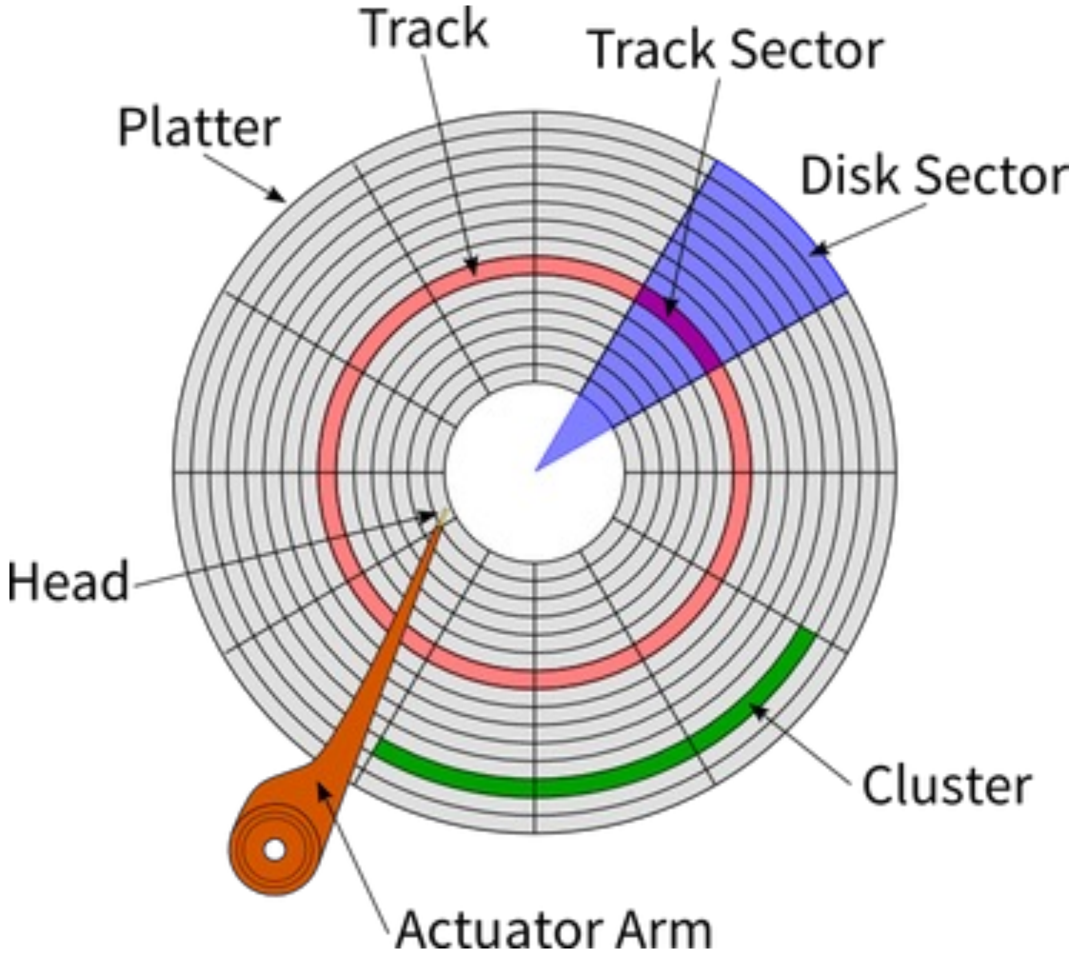
- **Platter** – double sided metal disks coated with a magnetic layer.
- **Surface** – each side of the platter
- **Spindle** – runs through all the platters and holds them together so they can be spun
  - Speed ranges can be 5,400 – 15,000 RPM*



*RPM == Rotations per Minute

HDD Anatomy – Data storage and Retrieval

- **Track** – concentric circles of sectors on the surface of the platter
- **Disk Head** – mechanism to read and write data data on the magnetic surface
- **Disk Arm** – swinging mechanism to move the disk head over the platter surface



*RPM == Rotations per Minute

HDD Operations

- Rotation - The platters of the HDD must spin to the correct sector(s) where data is stored
  - Rotational delay
- Seek time – The tracks (concentric circles) around the disk require the movement of the mechanical arm to position the read/write head over the correct track
  - Acceleration – arm begins to move
  - Coasting – full speed movement
  - Deceleration – arm slows
  - Settling – head is positioned over the correct track (can take .5 to 2 ms)
- Transfer – Read/write data to/from the the surface

Additional Considerations

- Track skew allows for sequential reads of data when the data crosses the boundaries between tracks
- HDD have a cache built in (8 or 16 MB)
  - Holds data for read/write operations
- Two types of caching:
  - **write back** to indicate that the write is done when the data is in the cache
  - **write through** to indicate that the data has been written to the disk

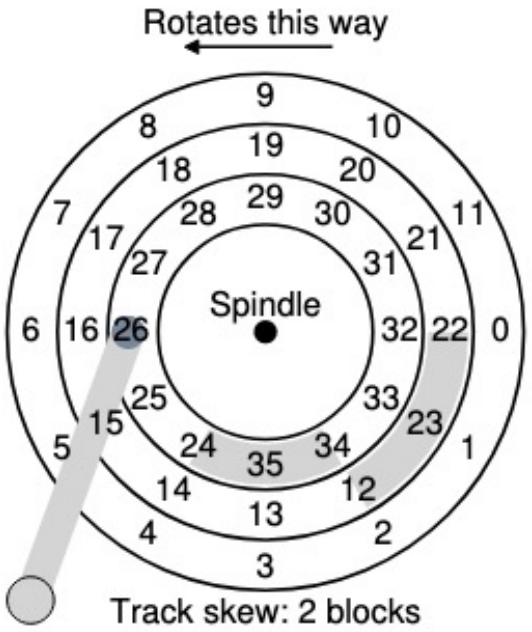


Figure 37.4: **Three Tracks: Track Skew Of 2**

Disk Performance

- I/O_Time
  - $T_{I/O} = T_{seek} + T_{rotation} + T_{transfer}$
- Rate of I/O
  - $R_{I/O} \frac{Size_{transfer}}{T_{I/O}}$
- Big Picture… USE DISKS SEQUENTIALLY!
  - If you can’t do sequential, large chunks
  - Bigger == Better

	Cheetah	Barracuda
$R_{I/O}$ Random	0.66 MB/s	0.31 MB/s
$R_{I/O}$ Sequential	125 MB/s	105 MB/s

Figure 37.6: **Disk Drive Performance: SCSI Versus SATA**

Disk Scheduling

- The OS decides the order of I/O (technically a set of I/O requests)

- The **disk scheduler** can estimate seek and rotational delay of the request to greedily apply a shortest job first type policy

- Approaches:
  - SSTF: Shortest Seek Time First
  - Elevator
  - SPTF: Shortest Positioning Time First

SSTF: Shortest Seek Time First

- Orders the queue of I/O requests by track
  - Picks the nearest track first
- The OS doesn't know the shape of the drive, just an array of locations, so instead we use **nearest-block-first (NBF)**
  - Schedule requests with the nearest block address first
- Problems?
  - **STARVATION**

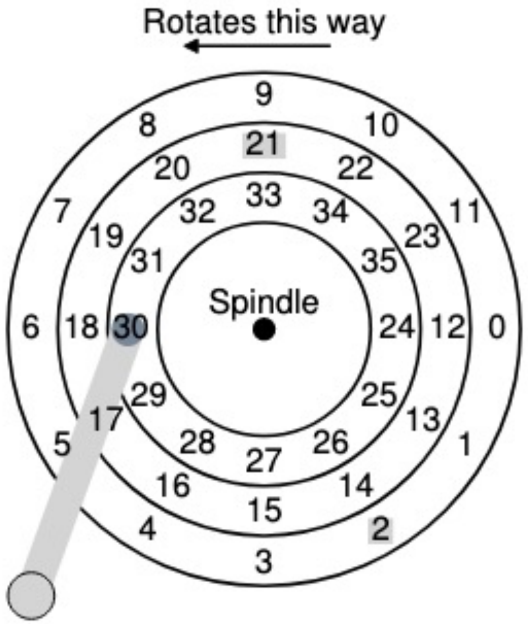


Figure 37.7: **SSTF: Scheduling Requests 21 And 2**

Elevator (SCAN or C-SCAN)

- **SCAN** services requests in order across the tracks as it sweeps across the disk (outer to inner or inner to outer tracks).
  - If a request has been made for a block on a track that was already serviced during the sweep, it must wait until the opposite sweep direction
- **F-SCAN** performs SCAN like scan but doesn’t immediately add requests to the service queue during a sweep
  - When the sweep finishes, then the requests are added to service queue
  - Avoids starving “far-away” requests with “closer-requests” that arrive late
- **C-SCAN** sweeps that disk in one direction, outer-to-inner and resets at the outer track
  - Supports fairness to inner and outer tracks by not favoring the middle

BUT WHAT ABOUT ROTATION?!

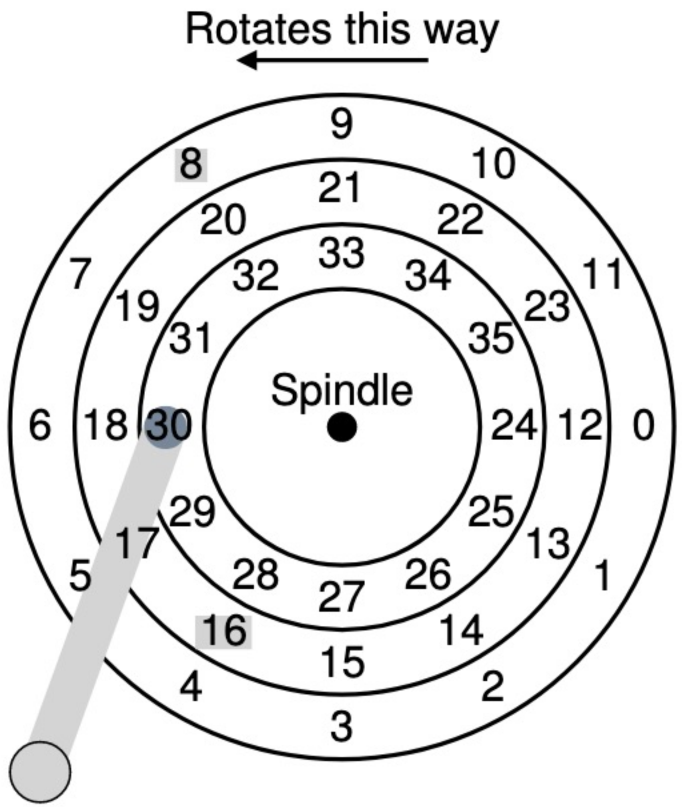
SCAN and SSTF doesn’t adhere closely to the SJF principle…

Shortest Positioning Time First (SPTF)

- Which should we serve first, 8 or 16?
  - Depends!
  - Seek time > rotational delay? 16 (SSTF)
  - Seek time < rotational delay? 8

- Modern drives use SPTF to consider both seek and rotation as they are roughly equivalent

- But how does the OS know all the information about the position of the HDD components?
  - It doesn’t, so what do we do?



With a little help from my friends

- HDD scheduling is a process split between the OS and the HDD hardware itself
- The OS picks some I/O jobs it believes are good choices and the HDD serves them in the best order it can
- The OS also can perform I/O merging to reduce the number of requests to the HDD
- The OS must also decide if it should wait to process I/O requests or do them immediately
  - Non-work-conserving vs work-conserving
  - Waiting may reveal a ”better” job request which can make for better performance

Next time

- We talk about RAID
  - More HDD stuff…