

SYSTEM I/O

Slides borrowed from:

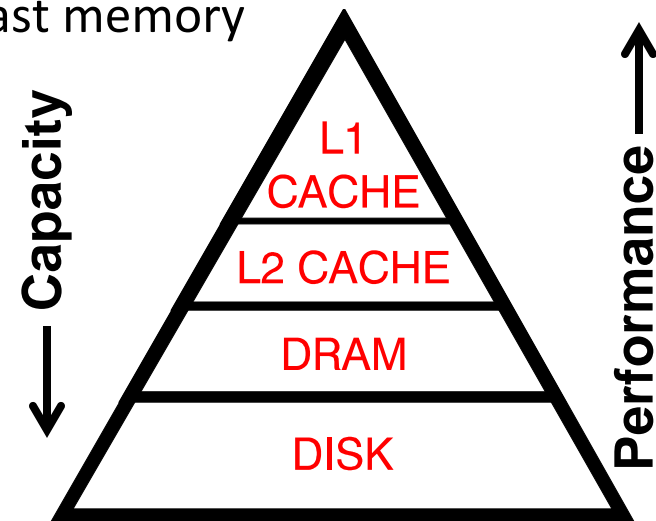
- **Computer Systems: Randal Bryant and David O'Hallaron(CMU site -course on architecture) and Operating systems by Silberschatz, et. al. book resources web site**

Disk-based storage in computers

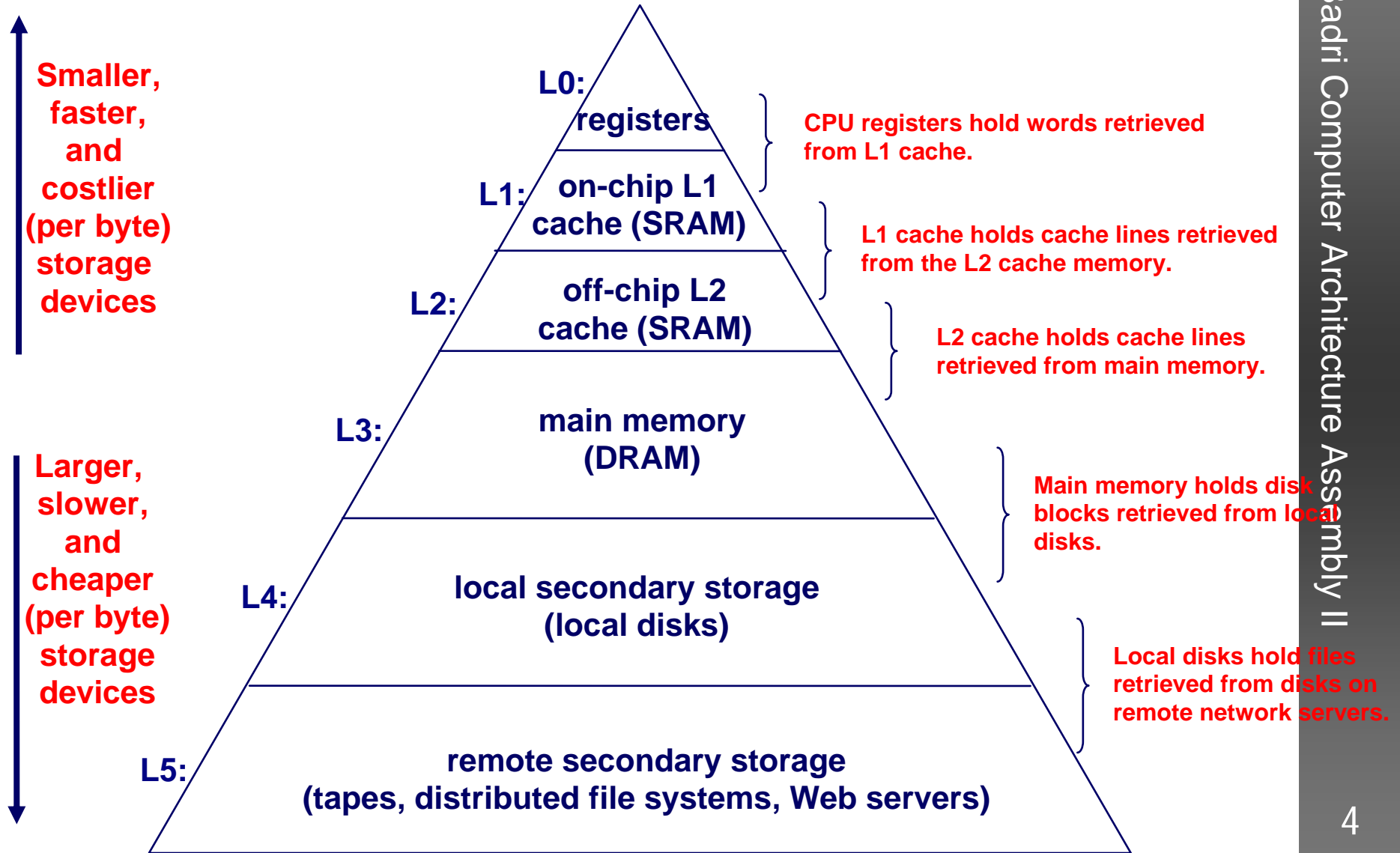
- **Memory/storage hierarchy**
 - Combining many technologies to balance costs/benefits
 - Recall the memory hierarchy and virtual memory lectures

Memory/storage hierarchies

- Balancing performance with cost
 - Small memories are **fast but expensive**
 - Large memories are **slow but cheap**
- Exploit locality to get the best of both worlds
 - locality = re-use/nearness of accesses
 - allows most accesses to use small, fast memory



An Example Memory Hierarchy



Disk-based storage in computers

- Memory/storage hierarchy
 - Combining many technologies to balance costs/benefits
 - Recall the memory hierarchy and virtual memory lectures
- **Persistence**
 - Storing data for lengthy periods of time
 - DRAM/SRAM is “volatile”: contents lost if power lost
 - Disks are “non-volatile”: contents survive power outages
 - Disk are blocks access (read/write blocks)
 - Conventional magnetic disks
 - Newer: Solid state disks

What's Inside A Disk Drive?

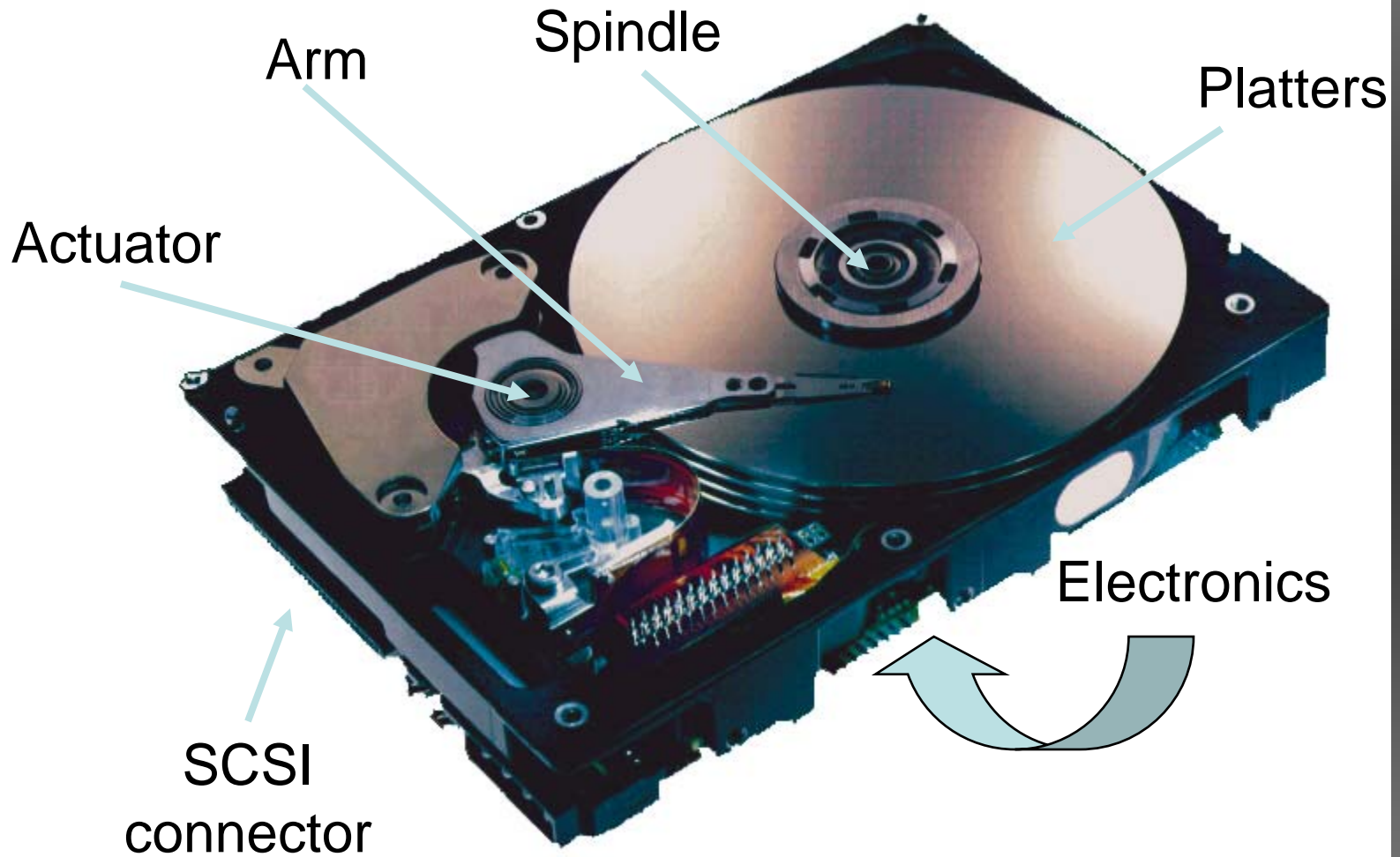
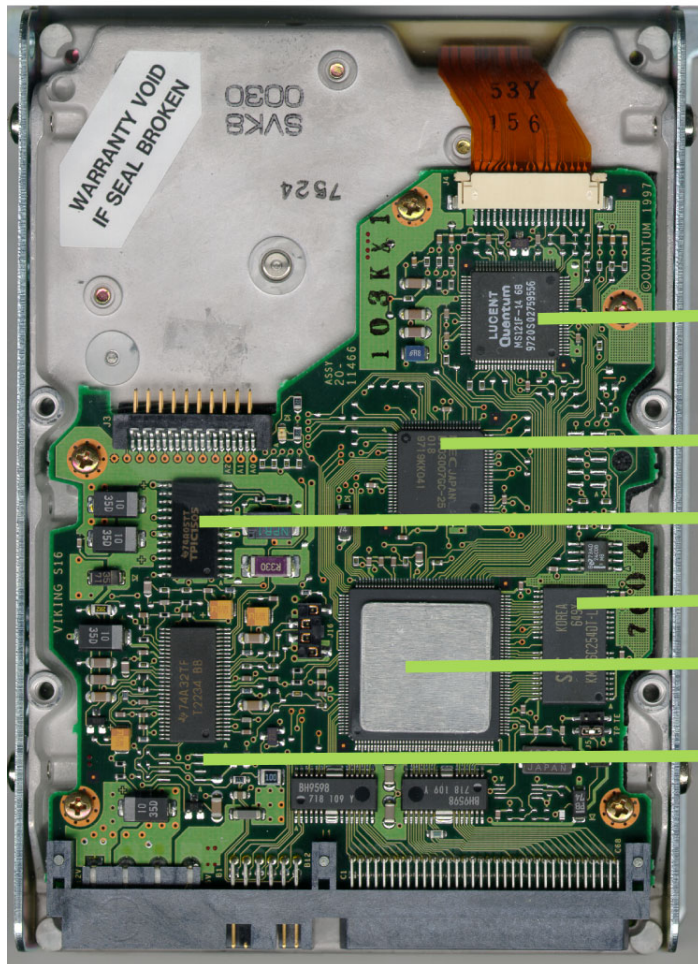


Image courtesy of Seagate Technology

Disk Electronics

Quantum Viking (circa 1997)



6 Chips

Just like a small computer – processor, memory, network interface

R/W Channel

- Connect to disk

uProcessor
32-bit, 25 MHz

- Control processor

Power Array

- Cache memory

2 MB DRAM

- Control ASIC

Control ASIC
SCSI, servo, ECC

- Connect to motor

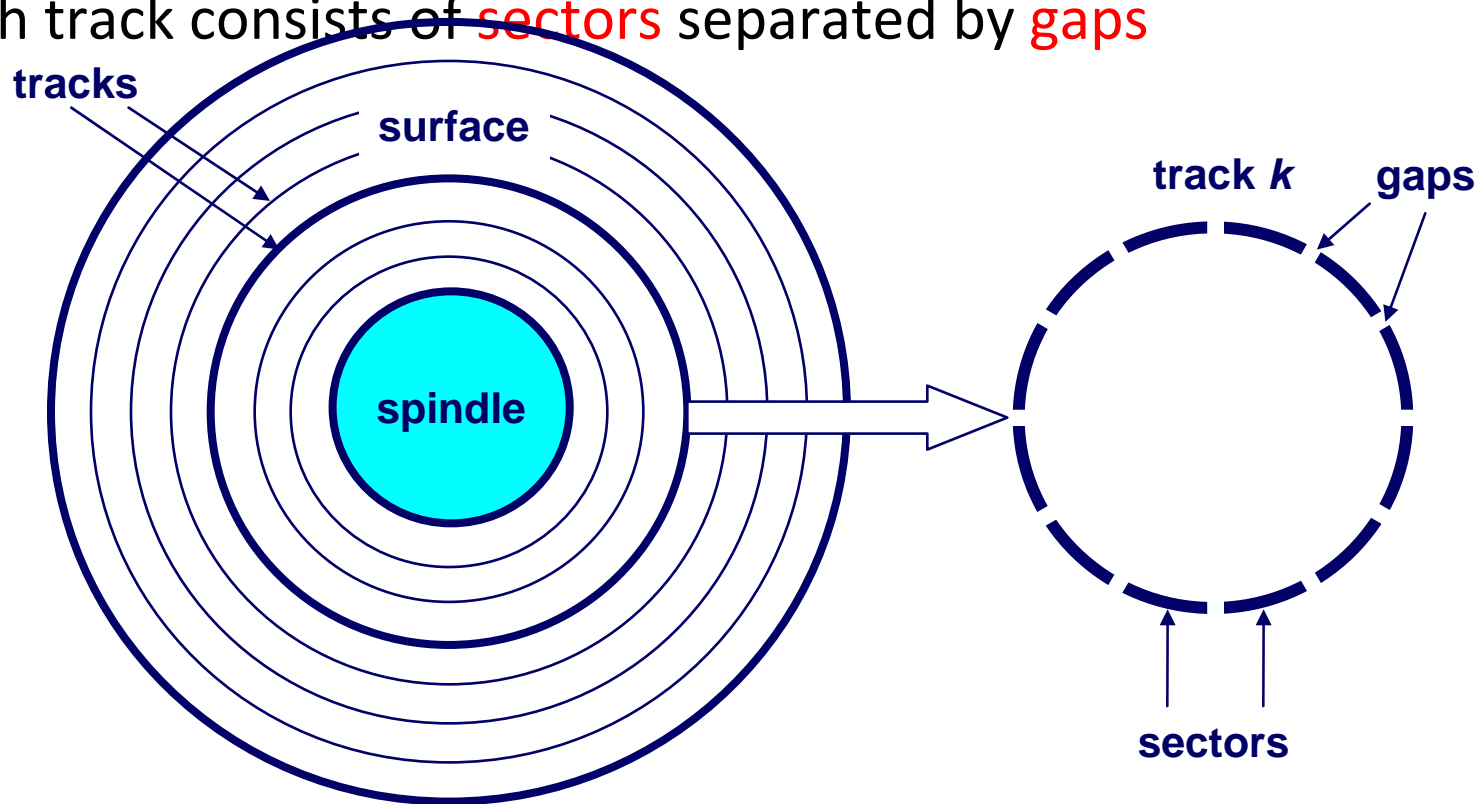
Motor/Spindle

Disk “Geometry”

Disks contain **platters**, each with two **surfaces**

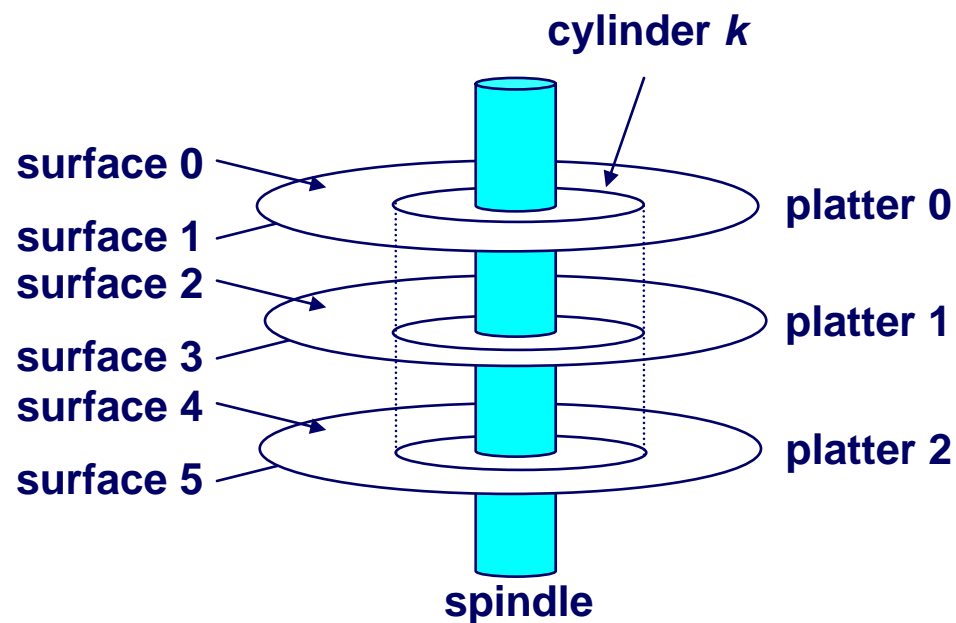
Each surface organized in concentric rings called **tracks**

Each track consists of **sectors** separated by **gaps**

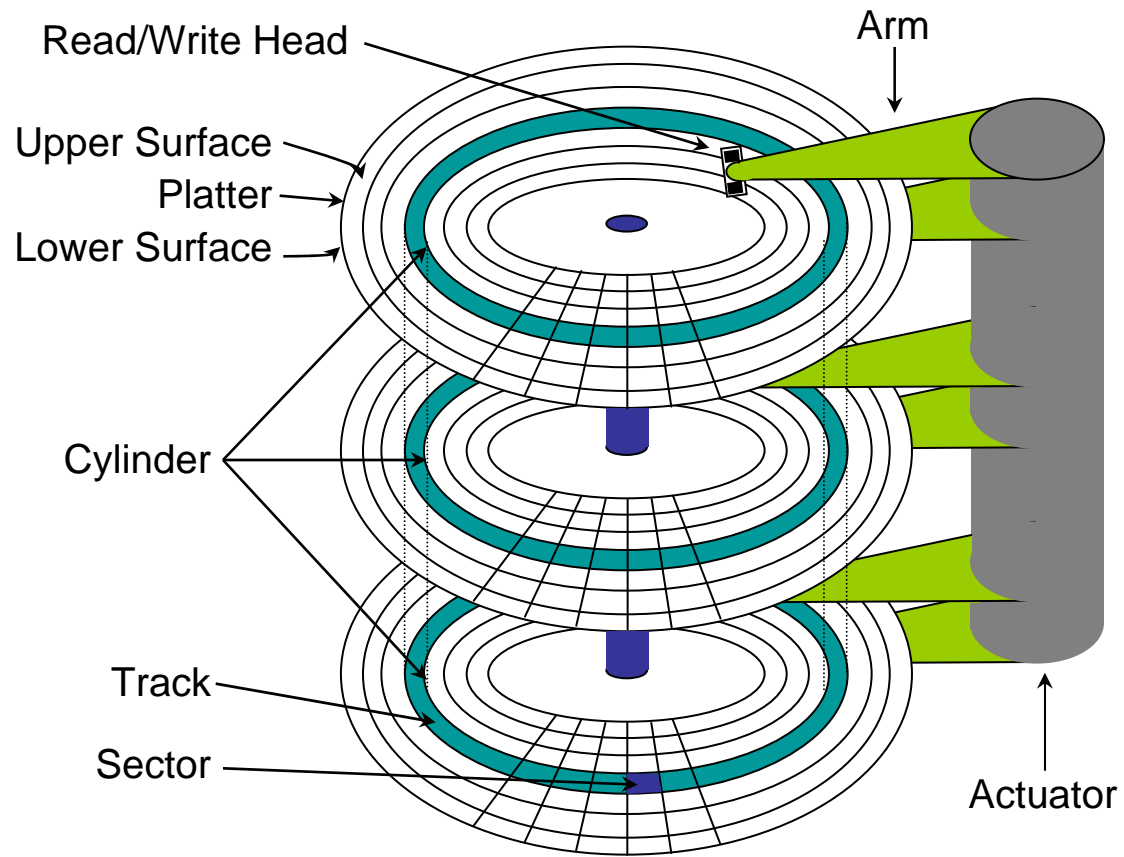


Disk Geometry (Multiple-Platter View)

Aligned tracks form a cylinder

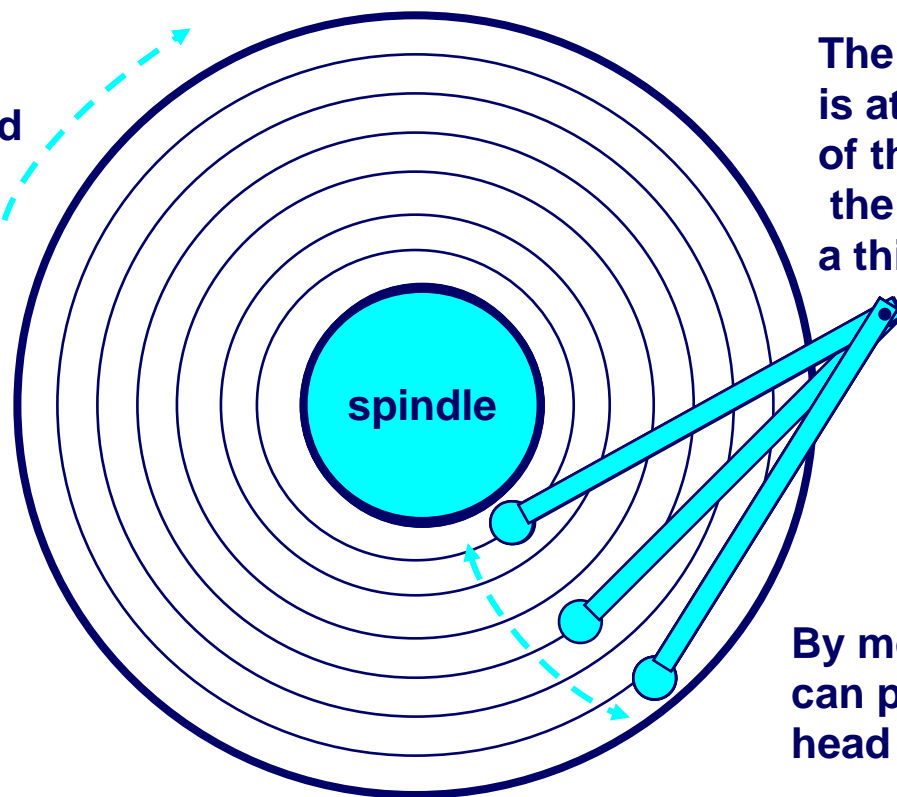


Disk Structure



Disk Operation (Single-Platter View)

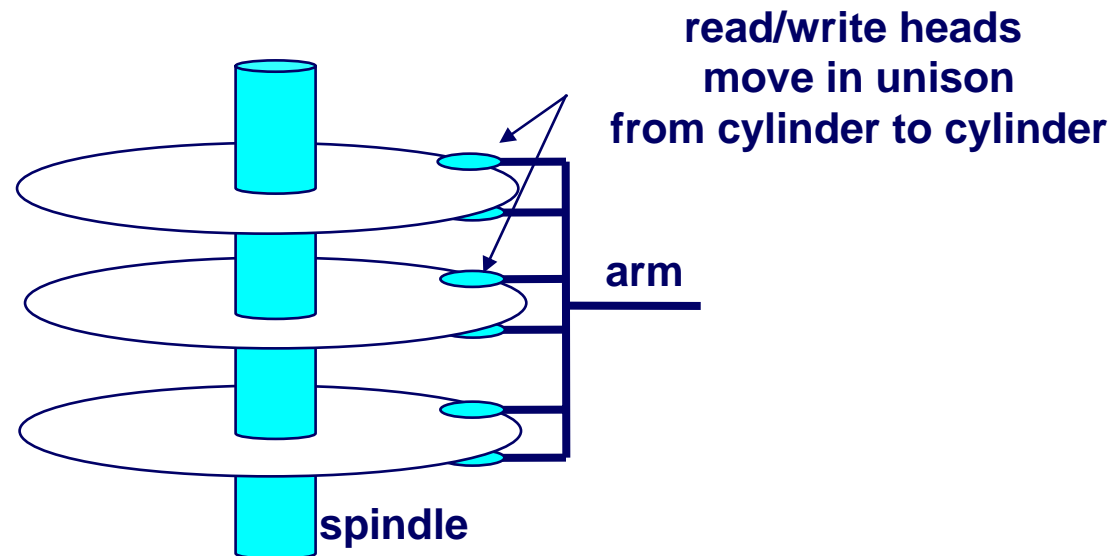
The disk surface spins at a fixed rotational rate



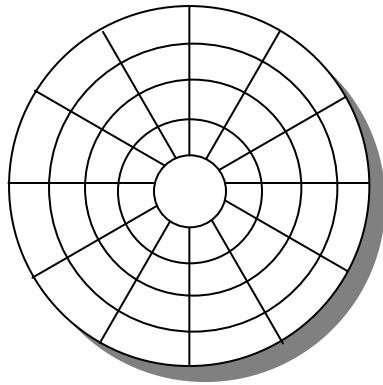
The read/write *head* is attached to the end of the *arm* and flies over the disk surface on a thin cushion of air

By moving radially, the arm can position the read/write head over any track

Disk Operation (Multi-Platter View)



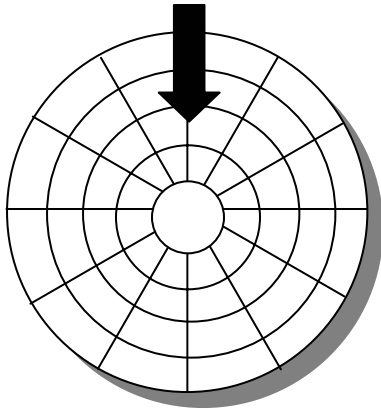
Disk Structure - top view of single platter



Surface organized into tracks

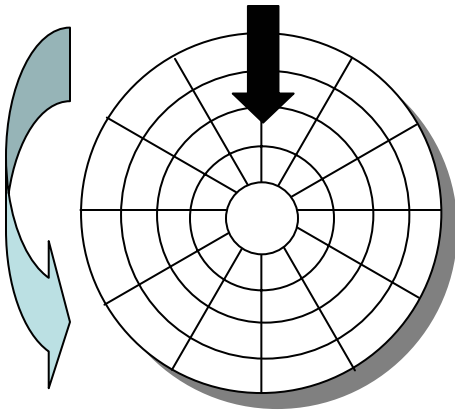
Tracks divided into sectors

Disk Access



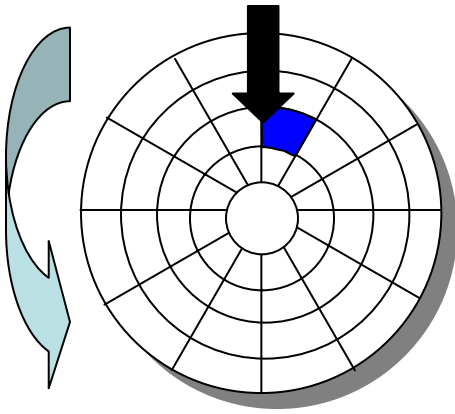
Head in position above a track

Disk Access



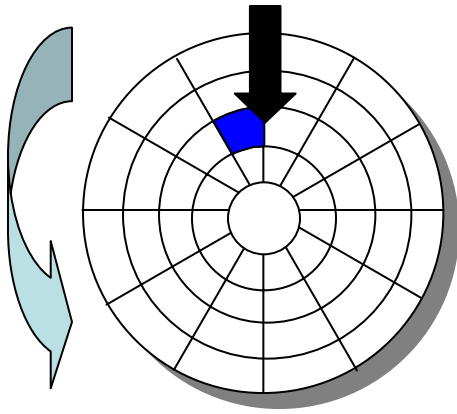
Rotation is counter-clockwise

Disk Access - Read



About to read blue sector

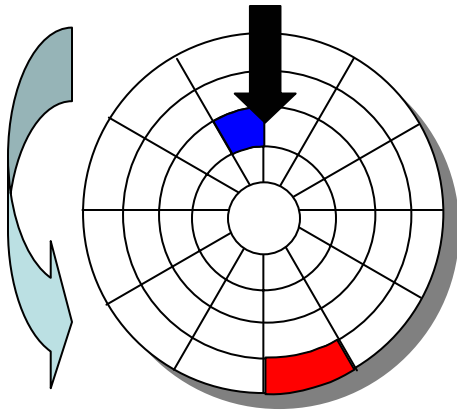
Disk Access - Read



After **BLUE** read

After reading blue sector

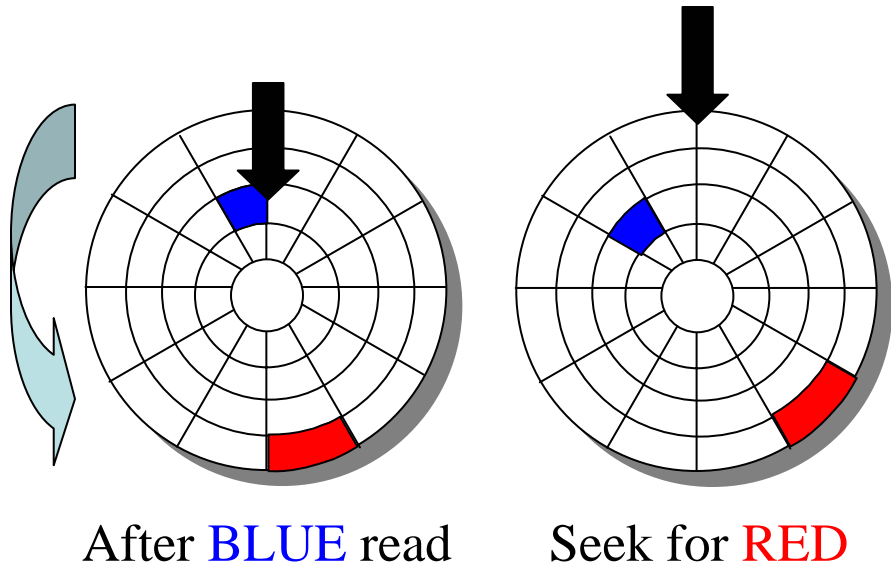
Disk Access - Read



After BLUE read

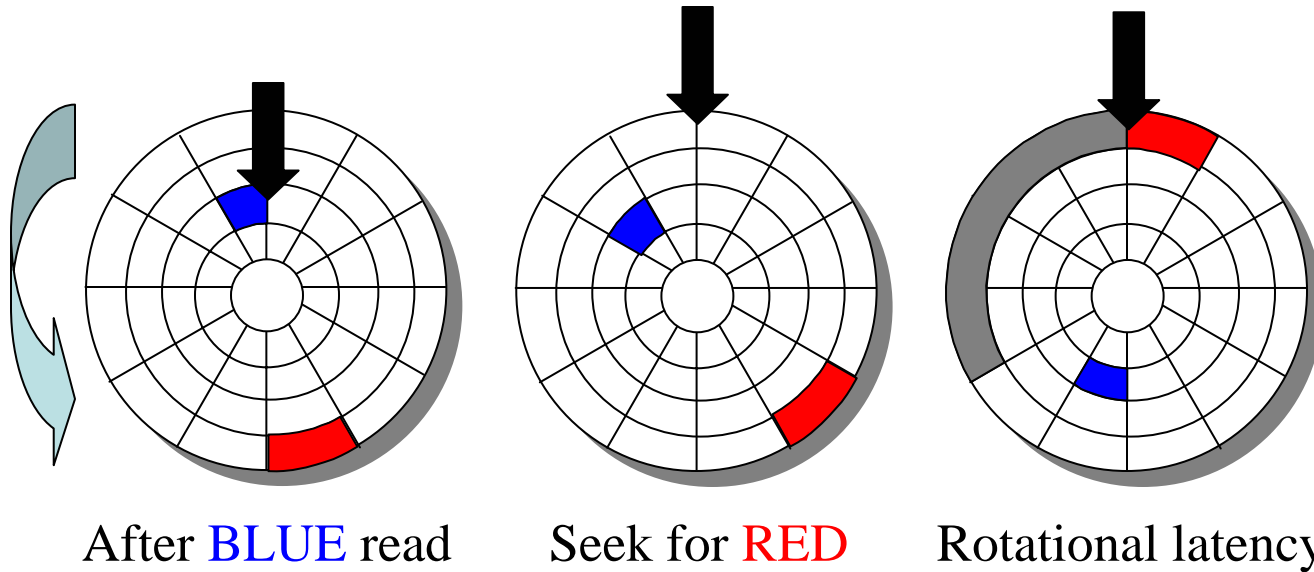
Red request scheduled next

Disk Access – Seek



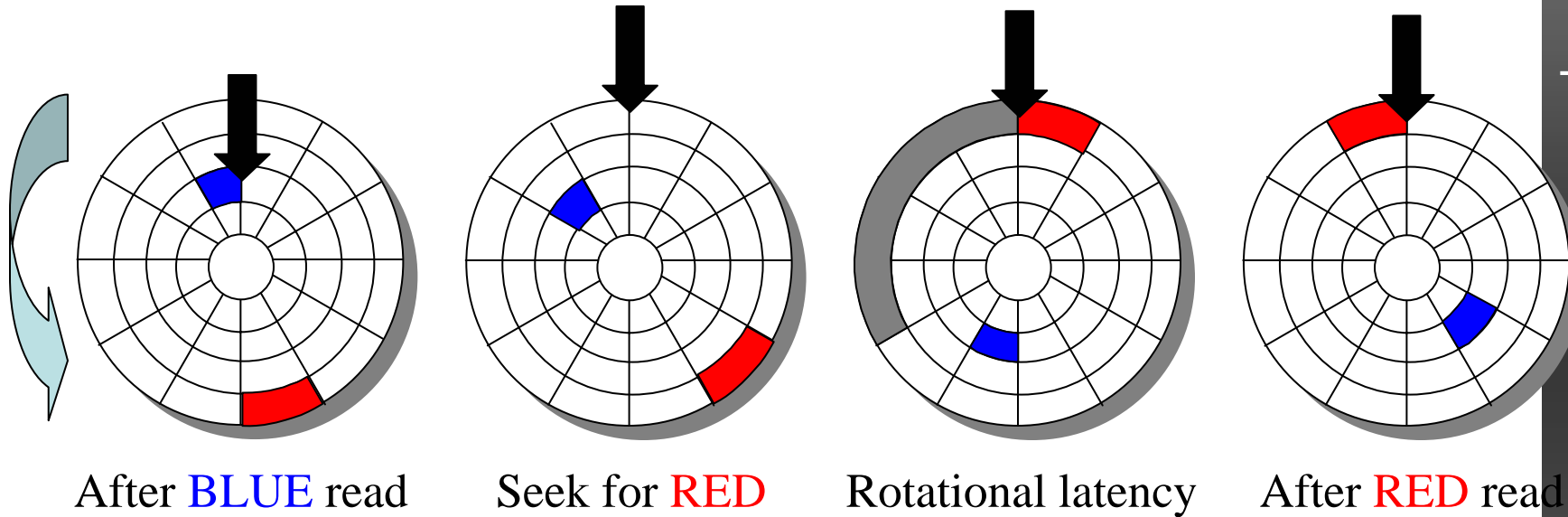
Seek to red's track

Disk Access – Rotational Latency



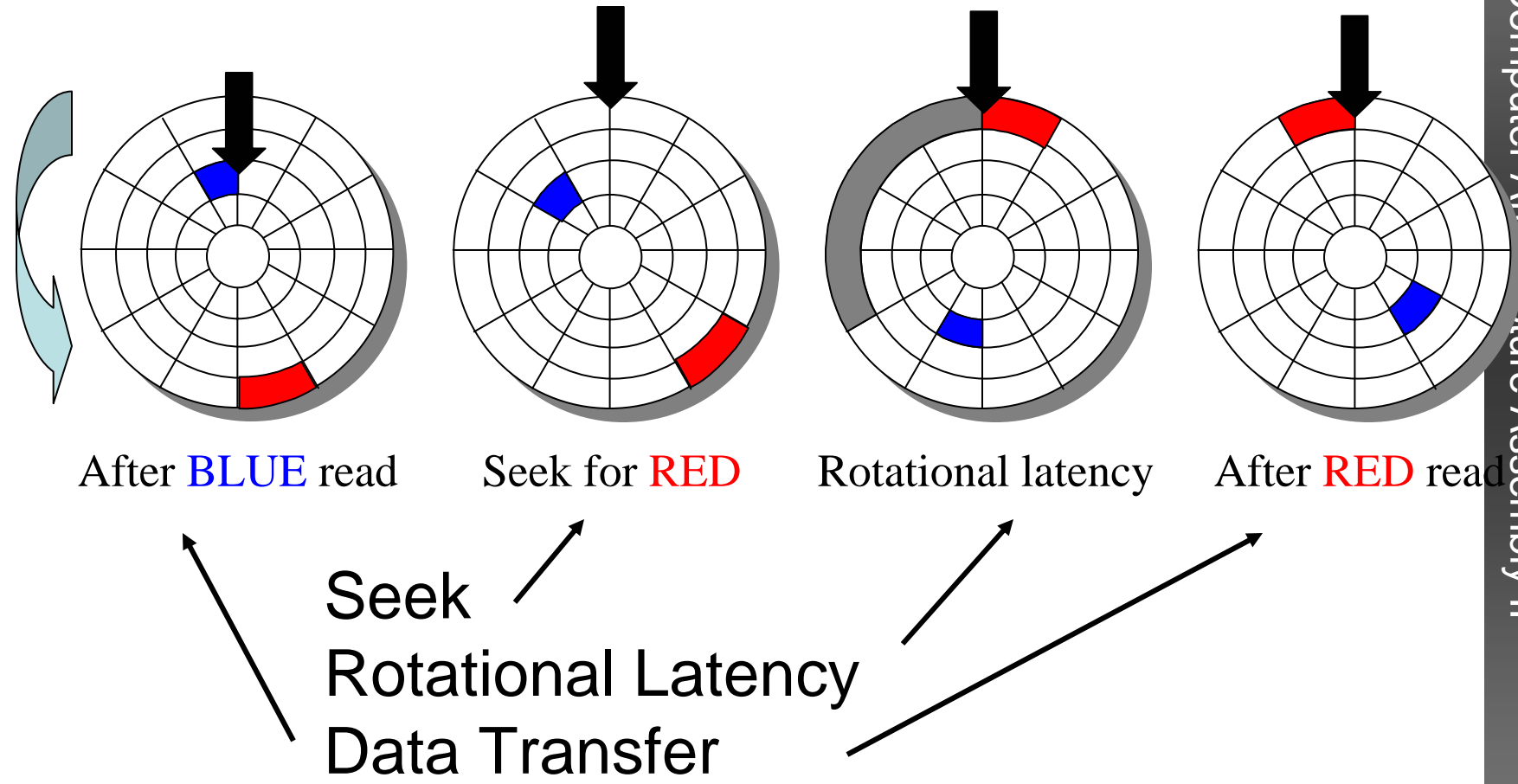
Wait for red sector to rotate around

Disk Access - Read



Complete read of red

Disk Access - Service Time Components



Disk Access Time

Average time to access a specific sector approximated by:

- $T_{\text{access}} = T_{\text{avg seek}} + T_{\text{avg rotation}} + T_{\text{avg transfer}}$

Seek time ($T_{\text{avg seek}}$)

- Time to position heads over cylinder containing target sector
- Typical $T_{\text{avg seek}} = 3\text{-}5\text{ ms}$

Rotational latency ($T_{\text{avg rotation}}$)

- Time waiting for first bit of target sector to pass under r/w head
- $T_{\text{avg rotation}} = \frac{1}{2} \times \frac{1}{\text{RPMs}} \times 60\text{ sec}/1\text{ min}$
 - e.g., 3ms for 10,000 RPM disk

Transfer time ($T_{\text{avg transfer}}$)

- Time to read the bits in the target sector
- $T_{\text{avg transfer}} = \frac{1}{\text{RPM}} \times \frac{1}{(\text{avg \# sectors}/\text{track})} \times 60\text{ secs}/1\text{ min}$
 - e.g., 0.006ms for 10,000 RPM disk with 1,000 sectors/track
 - given 512-byte sectors, $\sim 85\text{ MB/s}$ data transfer rate

Disk Access Time Example

Given:

- Rotational rate = 7,200 RPM
- Average seek time = 5 ms
- Avg # sectors/track = 1000

Derived average time to access random sector:

- $T_{\text{avg rotation}} = 1/2 \times (60 \text{ secs}/7200 \text{ RPM}) \times 1000 \text{ ms/sec} = 4 \text{ ms}$
- $T_{\text{avg transfer}} = 60/7200 \text{ RPM} \times 1/1000 \text{ secs/track} \times 1000 \text{ ms/sec} = 0.008 \text{ ms}$
- $T_{\text{access}} = 5 \text{ ms} + 4 \text{ ms} + 0.008 \text{ ms} = 9.008 \text{ ms}$
 - Time to read sector: 0.008 ms

Important points:

- Access time dominated by seek time and rotational latency
- First bit in a sector is the most expensive, the rest are free
- SRAM access time is about 4 ns/doubleword, DRAM about 60 ns
 - ~100,000 times longer to access a word on disk than in DRAM

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.
- Access time has two major components
 - *Seek time* is the time for the disk are to move the heads to the cylinder containing the desired sector.
 - *Rotational latency* is the additional time waiting for the disk to rotate the desired sector to the disk head.
- Minimize seek time
- Seek time \approx 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 Scheduling

- Several algorithms exist to schedule the servicing of disk I/O requests.
- We illustrate them with a request queue (0-199).

98, 183, 37, 122, 14, 124, 65, 67

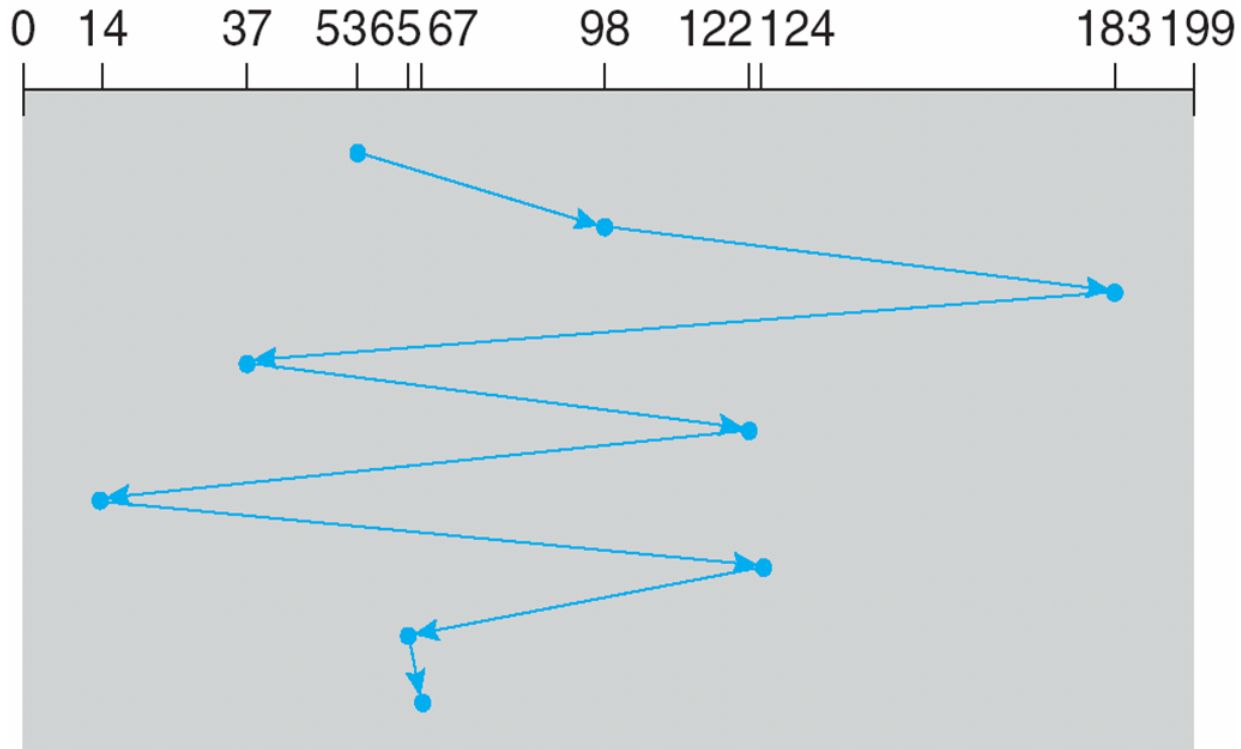
Head pointer 53

FCFS

Illustration shows total head movement of 640 cylinders.

queue = 98, 183, 37, 122, 14, 124, 65, 67

head starts at 53



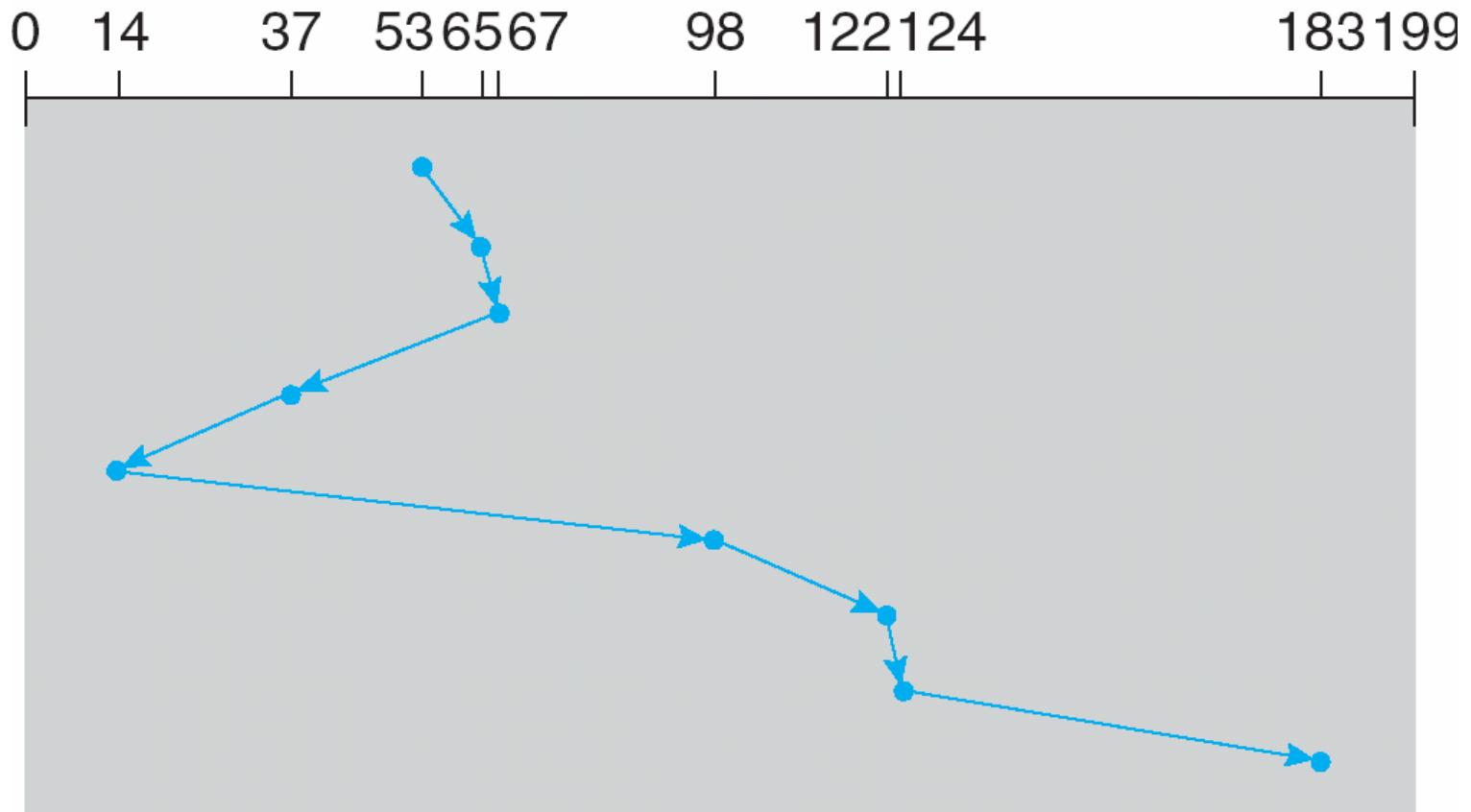
SSTF

- Selects the request with the minimum seek time from the current head position.
- SSTF scheduling is a form of SJF scheduling; may cause starvation of some requests.
- Illustration shows total head movement of 236 cylinders.

SSTF (Cont.)

queue = 98, 183, 37, 122, 14, 124, 65, 67

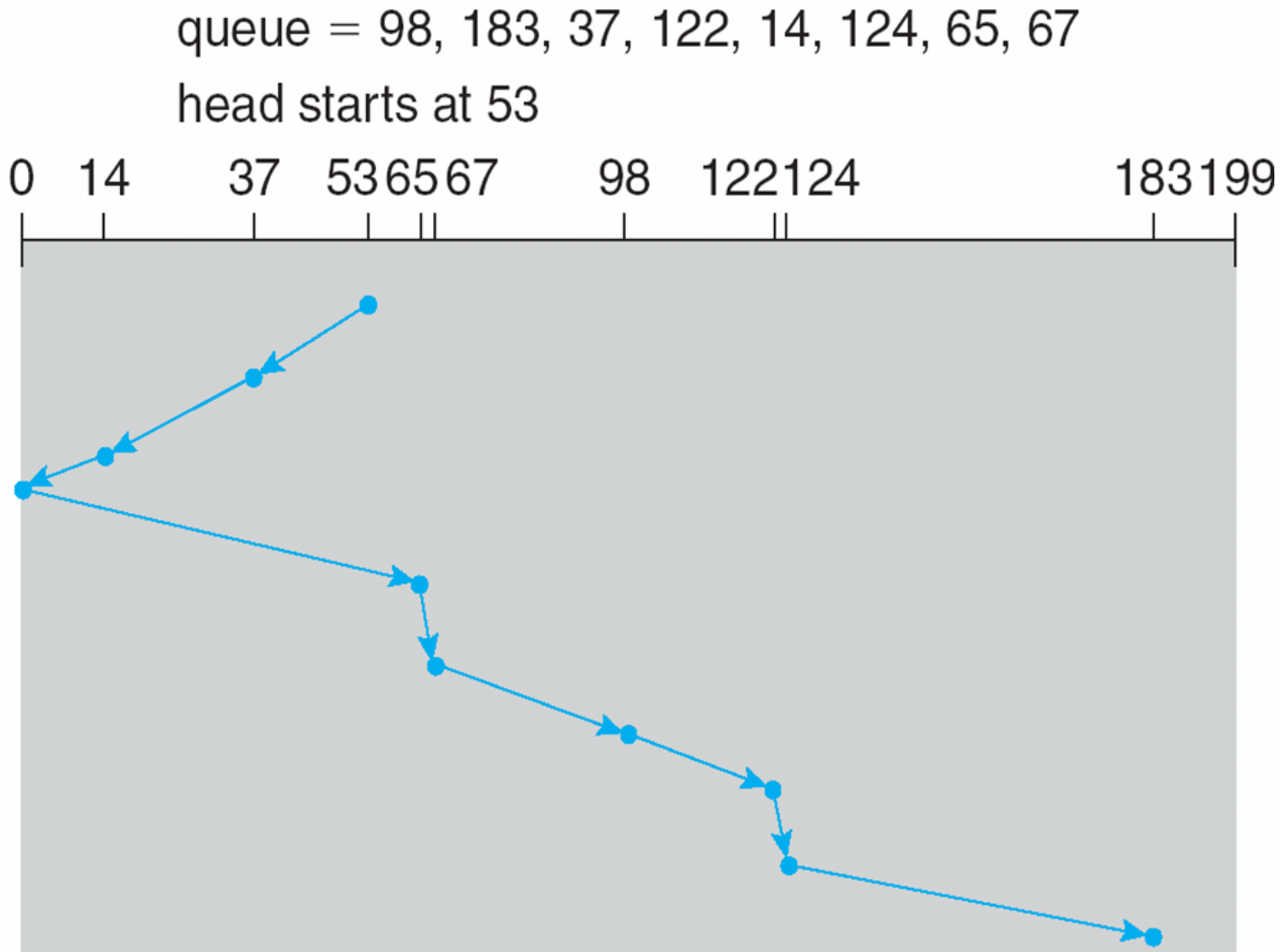
head starts at 53



SCAN

- The disk arm starts at one end of the disk, and moves toward the other end, servicing requests until it gets to the other end of the disk, where the head movement is reversed and servicing continues.
- Sometimes called the *elevator algorithm*.
- Illustration shows total head movement of 208 cylinders.

SCAN (Cont.)

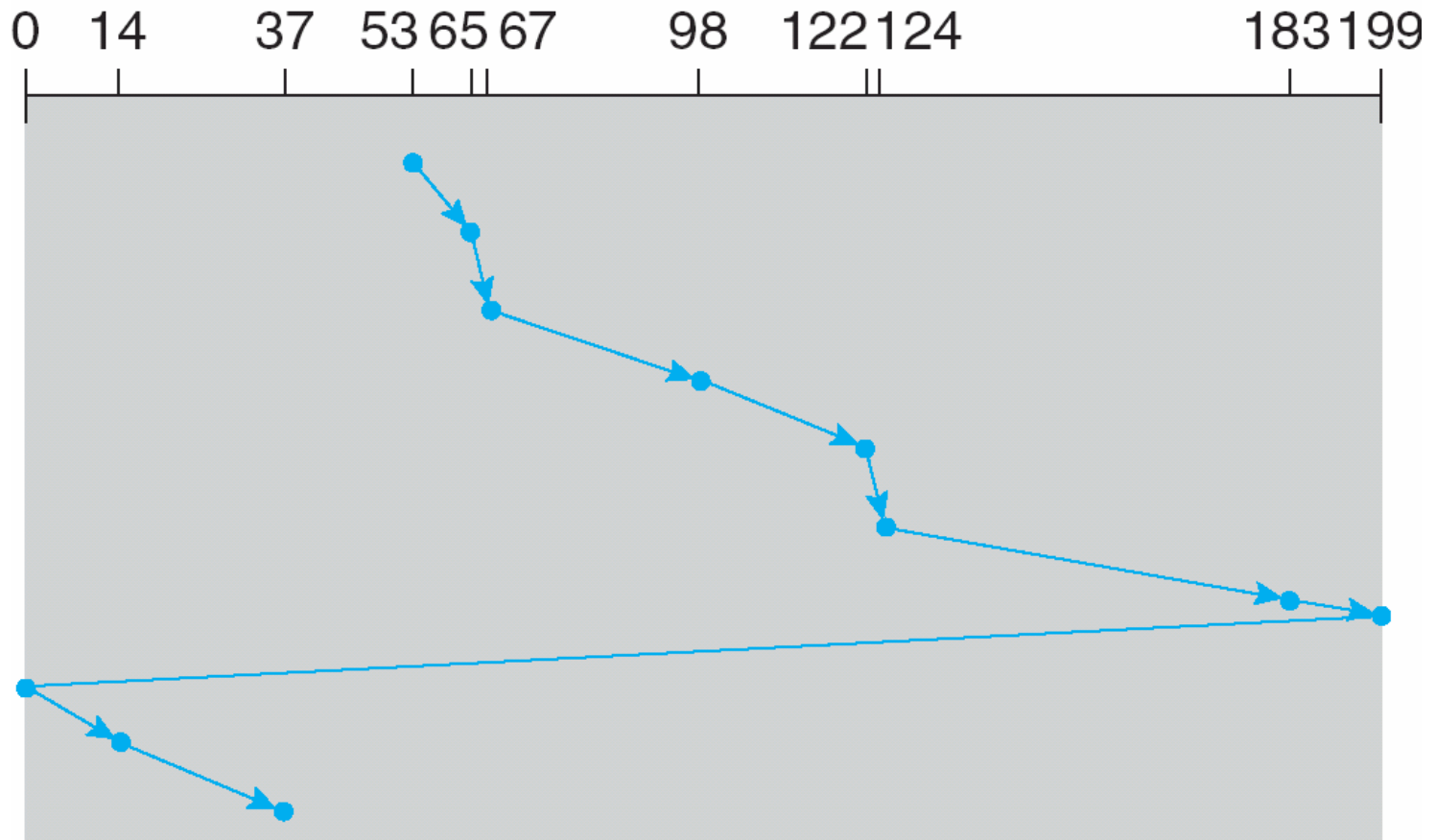


C-SCAN

- Provides a more uniform wait time than SCAN.
- The head moves from one end of the disk to the other, servicing requests as it goes. When it reaches the other end, however, it immediately returns to the beginning of the disk, without servicing any requests on the return trip.
- Treats the cylinders as a circular list that wraps around from the last cylinder to the first one.

C-SCAN (Cont.)

queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53



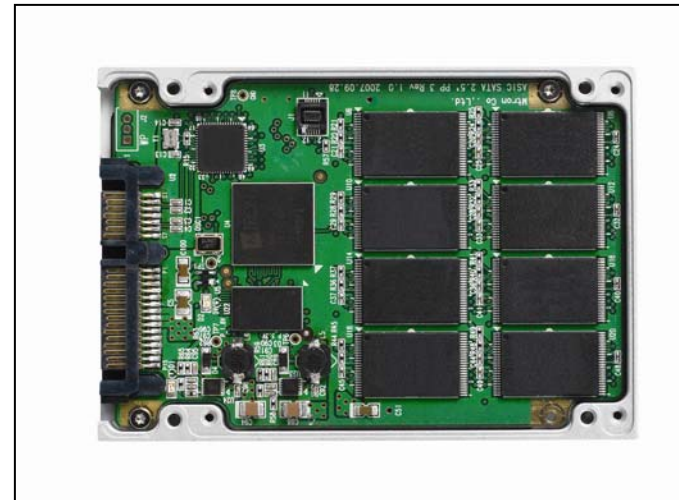
C-LOOK

- Version of C-SCAN
- Arm only goes as far as the last request in each direction, then reverses direction immediately, without first going all the way to the end of the disk.

Selecting a Disk-Scheduling Algorithm

- SSTF is common and has a natural appeal
- SCAN and C-SCAN perform better for systems that place a heavy load on the disk.
- Performance depends on the number and types of requests.
- Requests for disk service can be influenced by the file-allocation method.
- The disk-scheduling algorithm should be written as a separate module of the operating system, allowing it to be replaced with a different algorithm if necessary.
- Either SSTF or LOOK is a reasonable choice for the default algorithm.

Solid state disks



Solid state disks

- An array of flash memory devices
- Emulates conventional hard disk drive or HDD
- No moving parts
- Consumes less power than HDD
- Small reads (< 4K) are 20x faster
- Average reads comparable to HDD reads
- Writes are still slow $\frac{1}{2}$ x slower than HDD
- Capacity/cost (today)
- 0.15\$/GB-HDD , 2-3\$/GB-SSD

Solid State Drive

- The interface to the system looks like HDD
- Read, write and erase
- Memory consists of blocks
- Each block contains several pages
- Each page is 2K or 4K in size
- Unit of read/write are pages
- Need to erase before write!

