# Storage Technologies

CSCI 237: Computer Organization
21st Lecture, Apr 11, 2025

**Jeannie Albrecht**

# Administrative Details

- Lab 4
  - Due next Tue/Wed
  - Good job!
  - Any questions?

# Last Time

- Wrap-Up of PIPE Design (Ch 4.5)
  - Dealing with data hazards

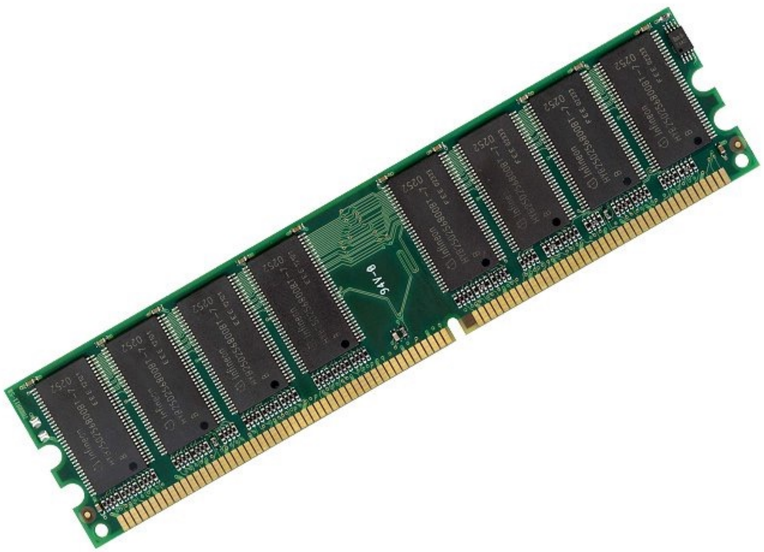
# Today

- Moving on to storage technologies and trends (Ch 6.1)
  - Memory technologies
  - Disk storage
  - Solid state disks

# Random-Access Memory (RAM)

- Key features
  - RAM is traditionally packaged as a chip
  - Basic storage unit is normally a cell (one bit per cell)
  - Multiple RAM chips form a **memory**

- RAM comes in two varieties:
  - SRAM (Static RAM)
    - Stores bits in bistable cells
  - DRAM (Dynamic RAM)
    - Stores bits as charge on capacitors

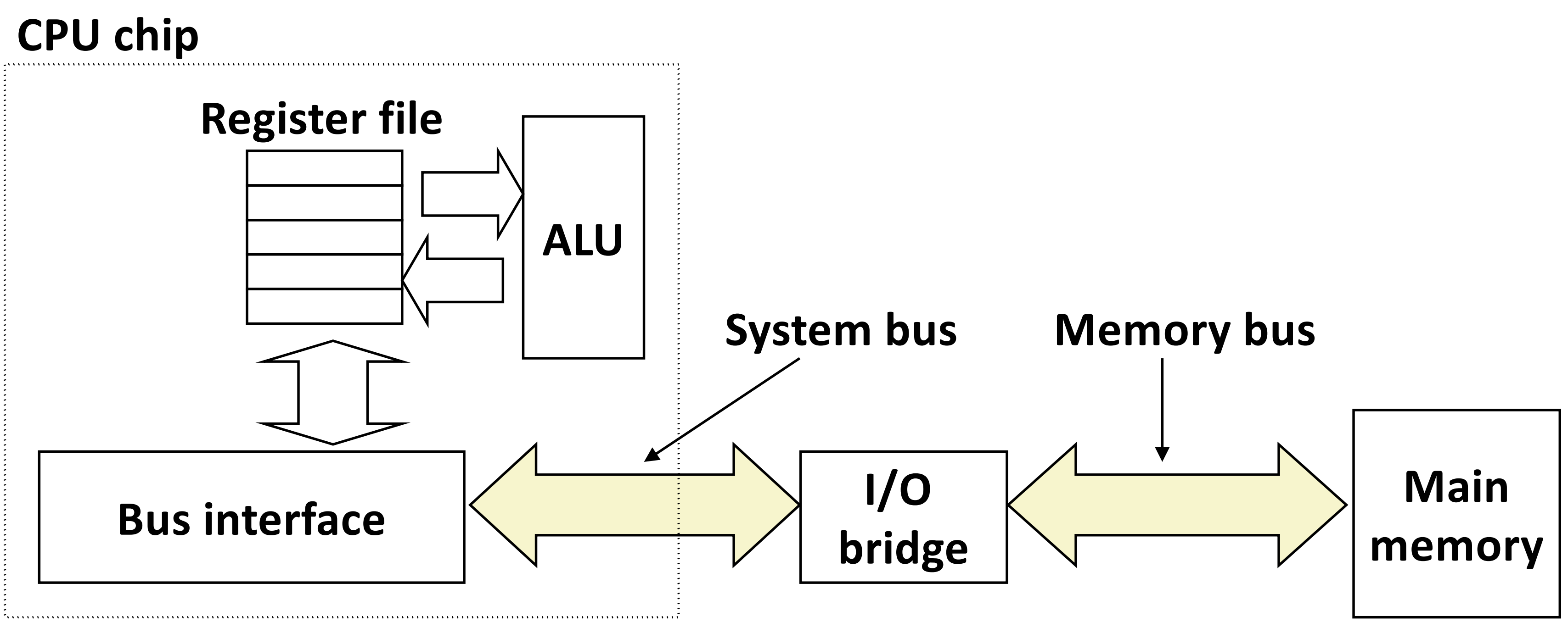


# Nonvolatile Memories

- DRAM and SRAM are **volatile** memories
  - Lose information if powered off
- Nonvolatile memories retain value even if powered off
  - Read-only memory (ROM): programmed during production
  - Programmable ROM (PROM): can be programmed once
  - Eraseable PROM (EPROM): can be bulk erased w/ UV light
  - Electrically eraseable PROM (EEPROM): electronic erase capability
  - Flash memory: EEPROMs. with partial (block-level) erase capability
    - Wears out after about 100,000 erasings
- Uses for Nonvolatile Memories
  - Firmware programs stored in a ROM (BIOS, controllers for disks, network cards, graphics accelerators, security subsystems,…)
  - **Solid state disks** (popular replacement for rotating disks)
  - Disk caches

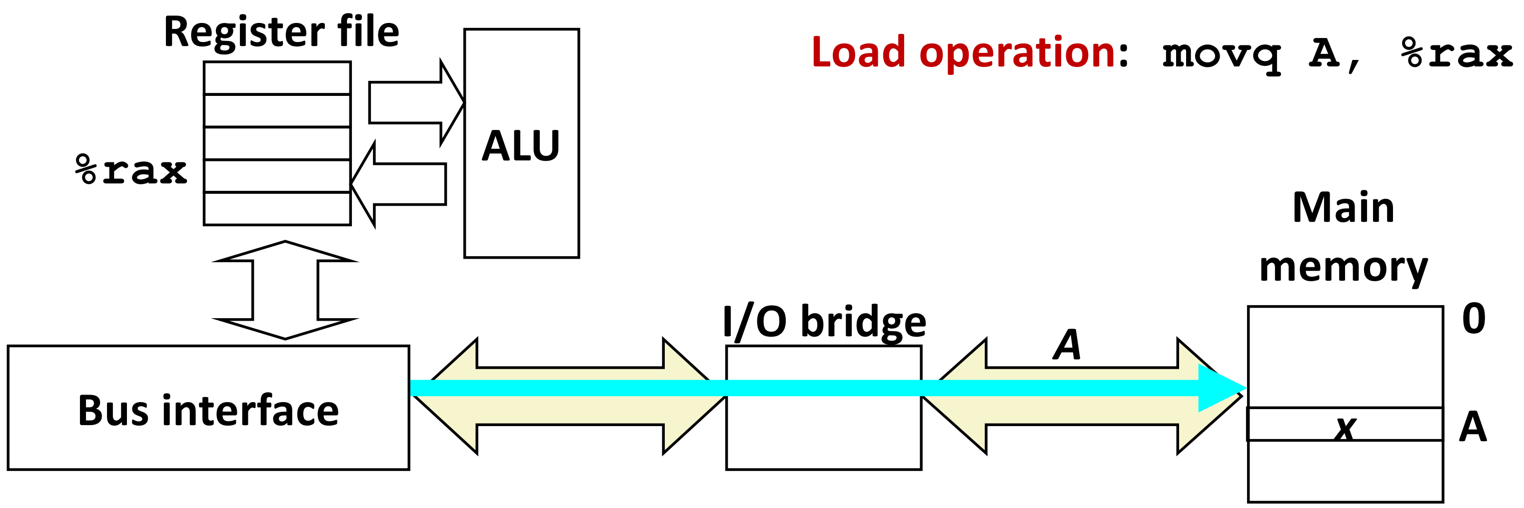
# Traditional Bus Structure Connecting CPU and Memory

- A bus is a collection of parallel wires that carry address, data, and control signals.
- Buses are typically shared by multiple devices.



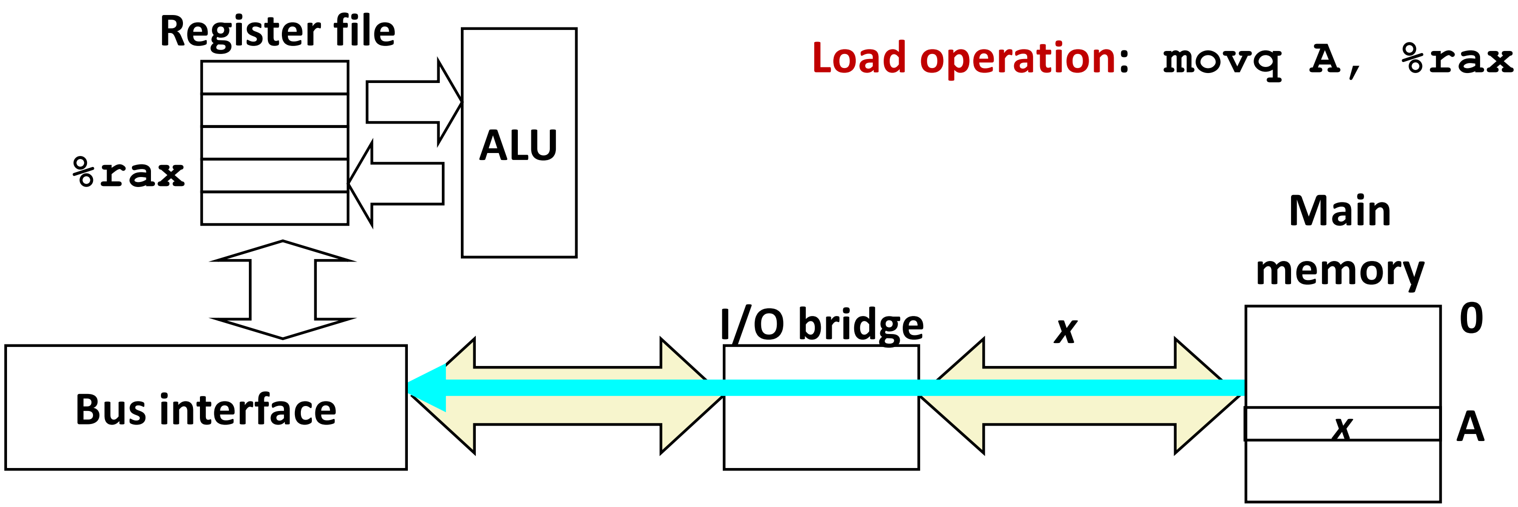
# Memory Read Transaction (1)

- CPU places memory address A on the memory bus.



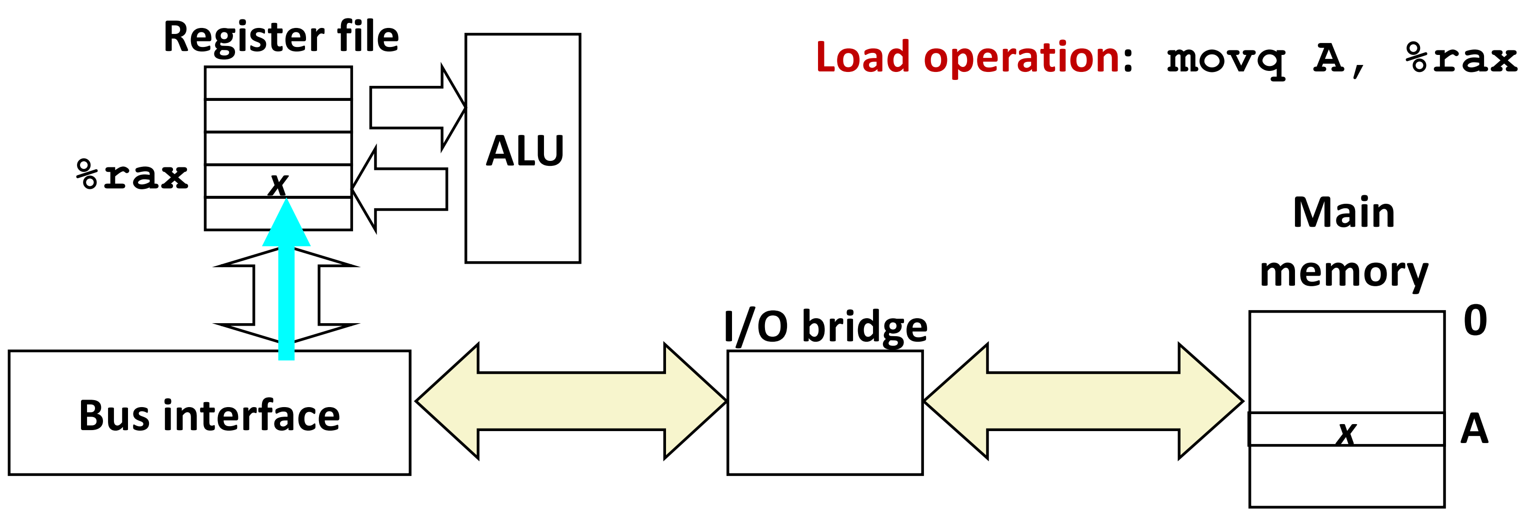
# Memory Read Transaction (2)

- Main memory reads A from the memory bus, retrieves word x, and places it on the bus.



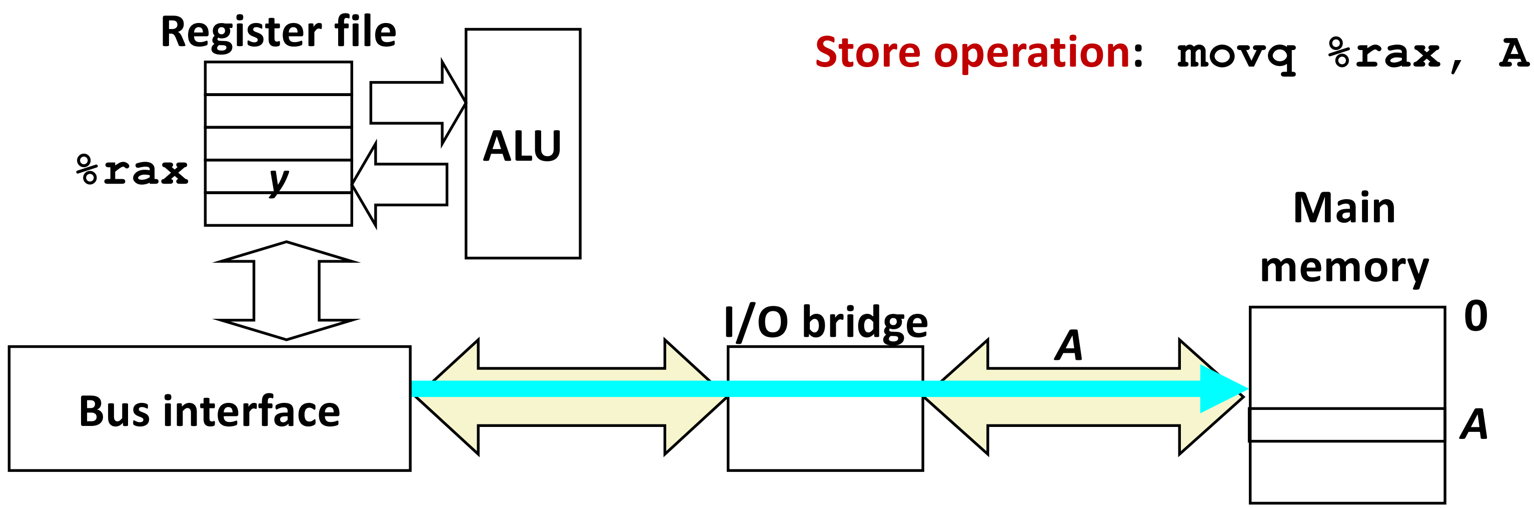
# Memory Read Transaction (3)

- CPU reads word x from the bus and copies it into register %rax.



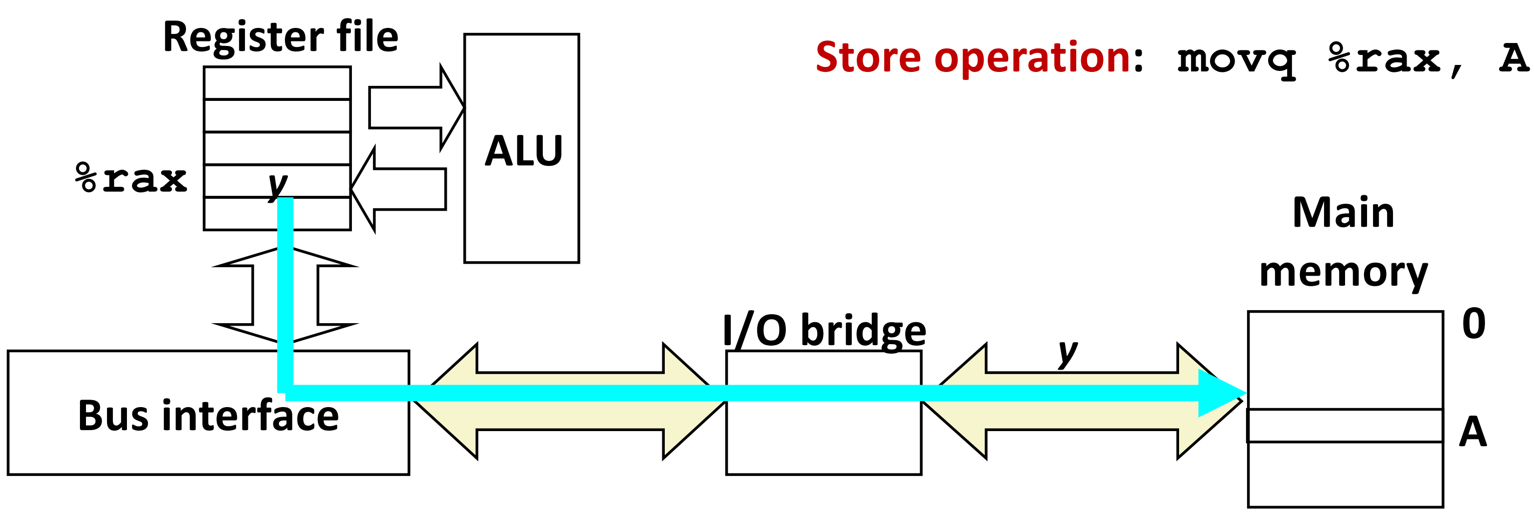
# Memory Write Transaction (1)

- CPU places address A on bus. Main memory reads it and waits for the corresponding data word to arrive.



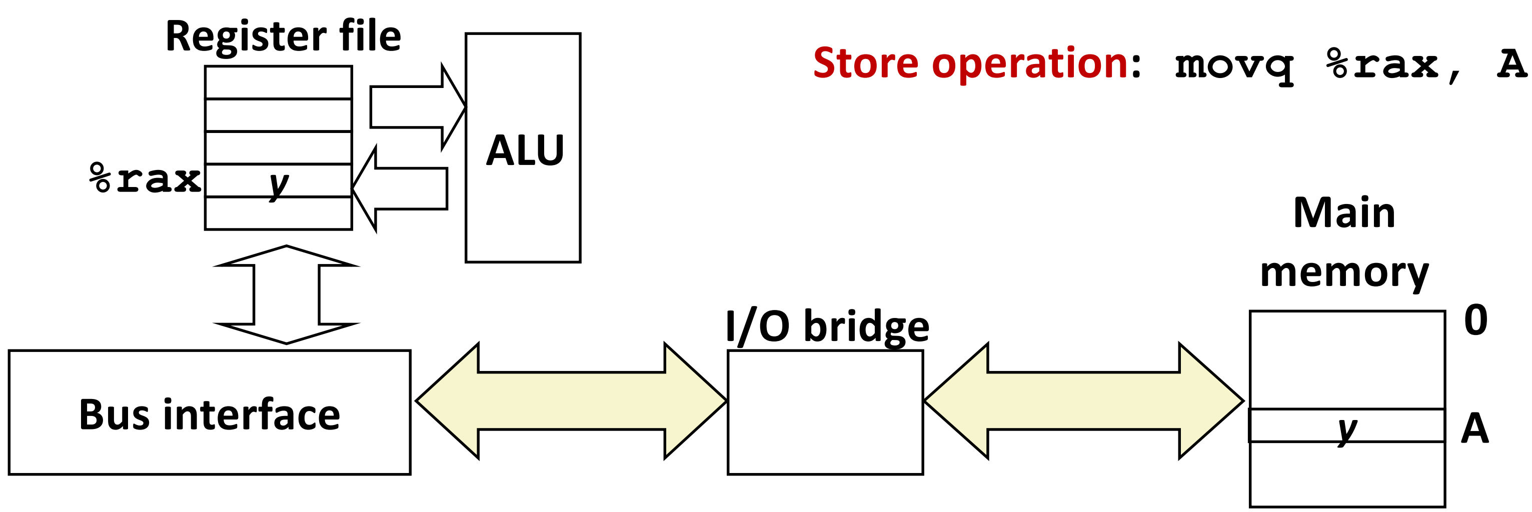
# Memory Write Transaction (2)

- CPU places data word y on the bus.



# Memory Write Transaction (3)

- Main memory reads data word y from the bus and stores it at address A.



# Next up: Disks! What's Inside A Disk Drive?

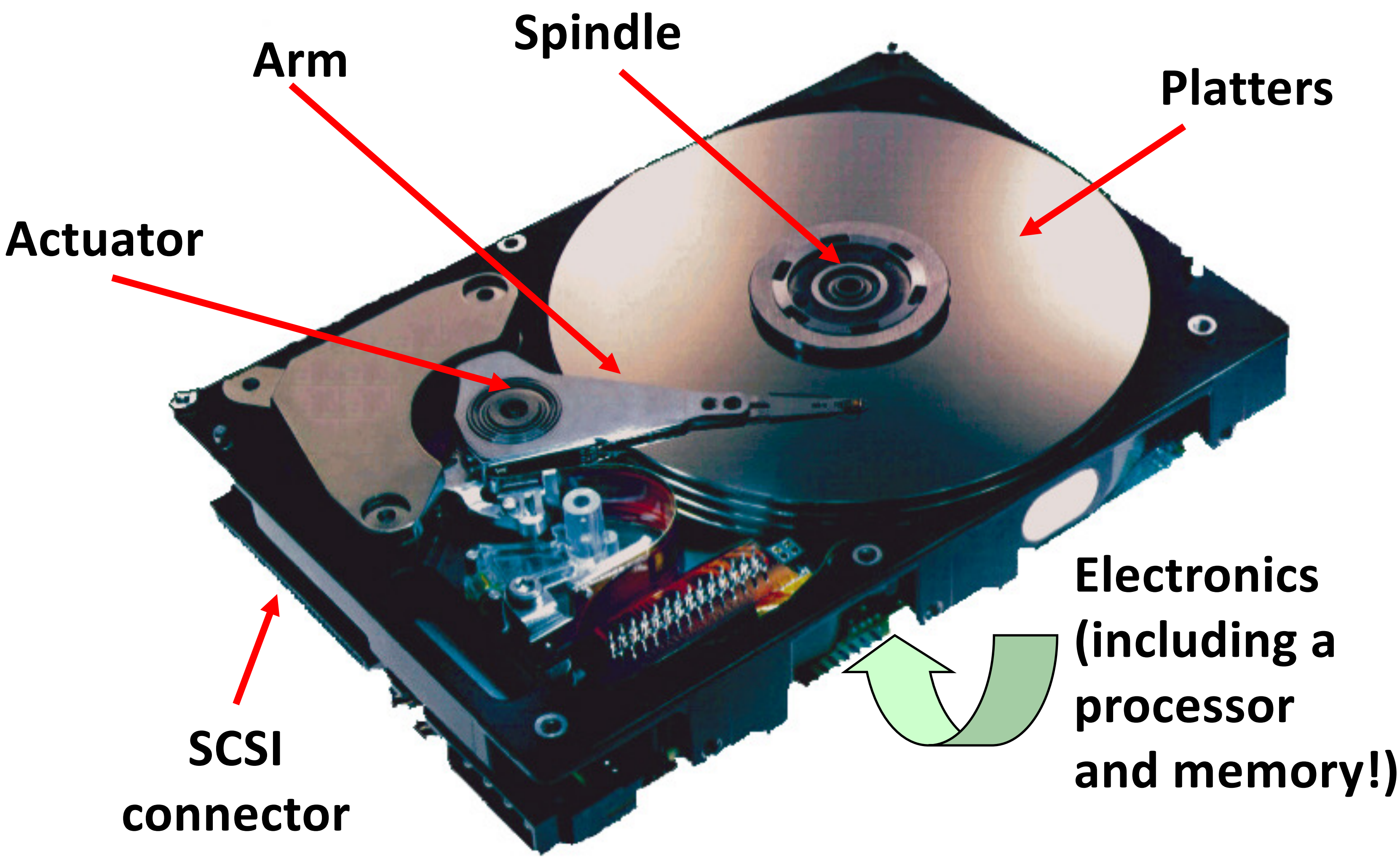
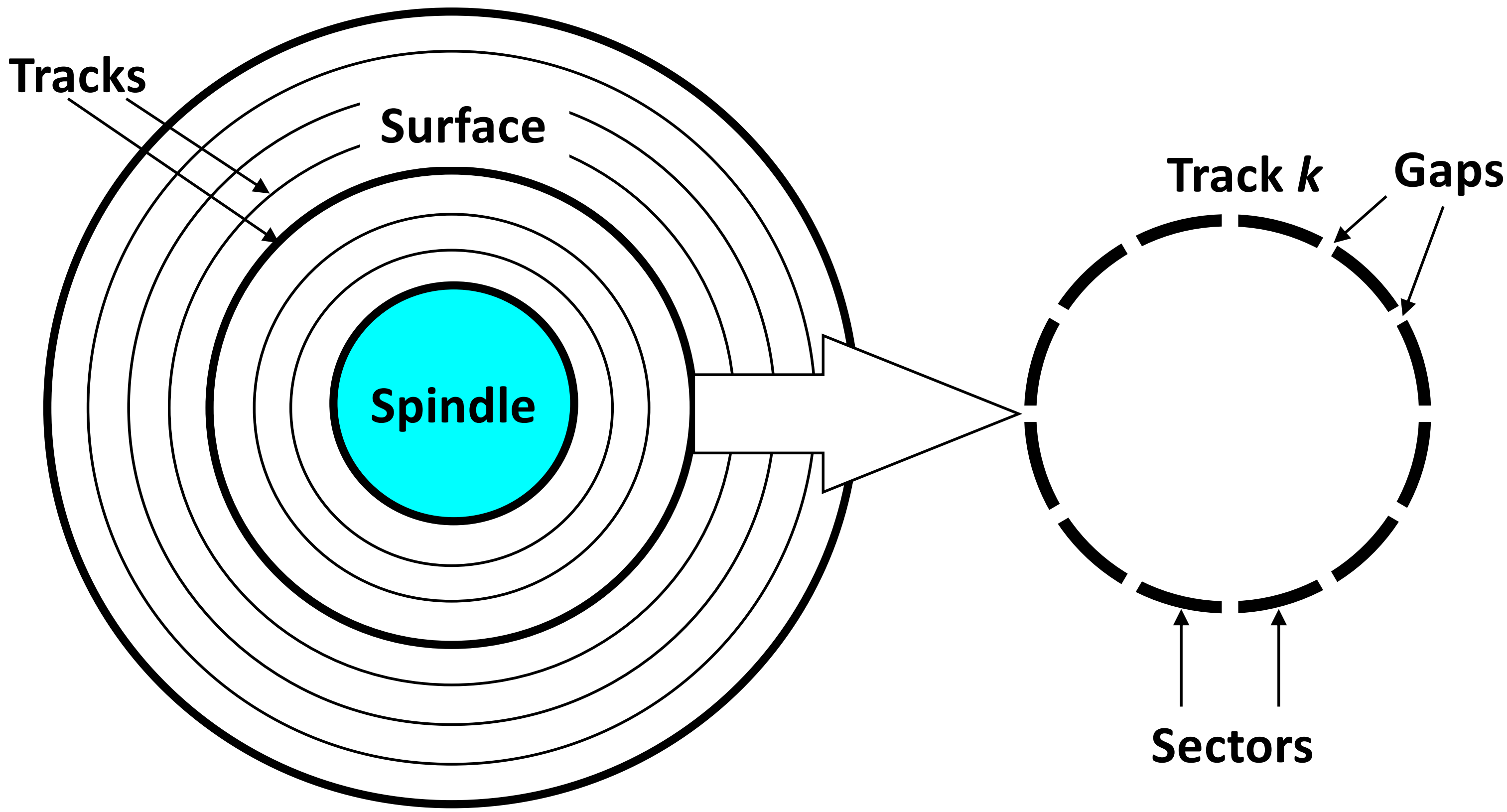



*Image courtesy of Seagate Technology*

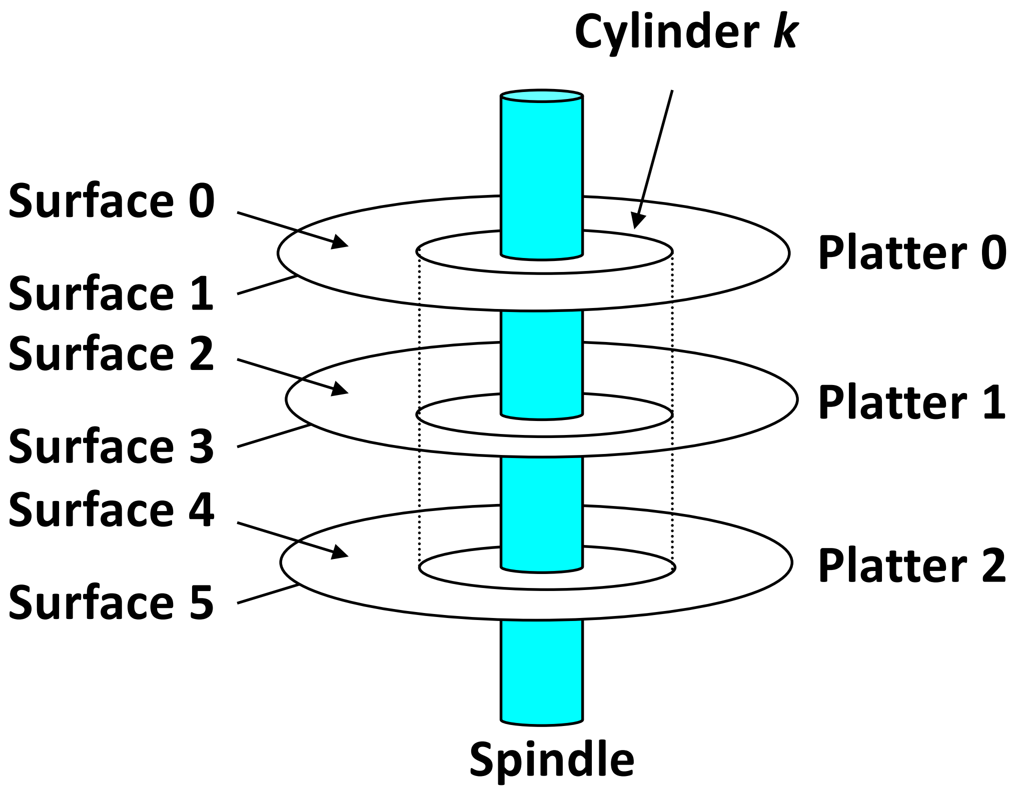
# Disk Geometry

- Disks consist of platters, each with two surfaces.
- Each surface consists of concentric rings called tracks.
- Each track consists of sectors separated by gaps.



# Disk Geometry (Multiple-Platter View)

- Aligned tracks form a cylinder.

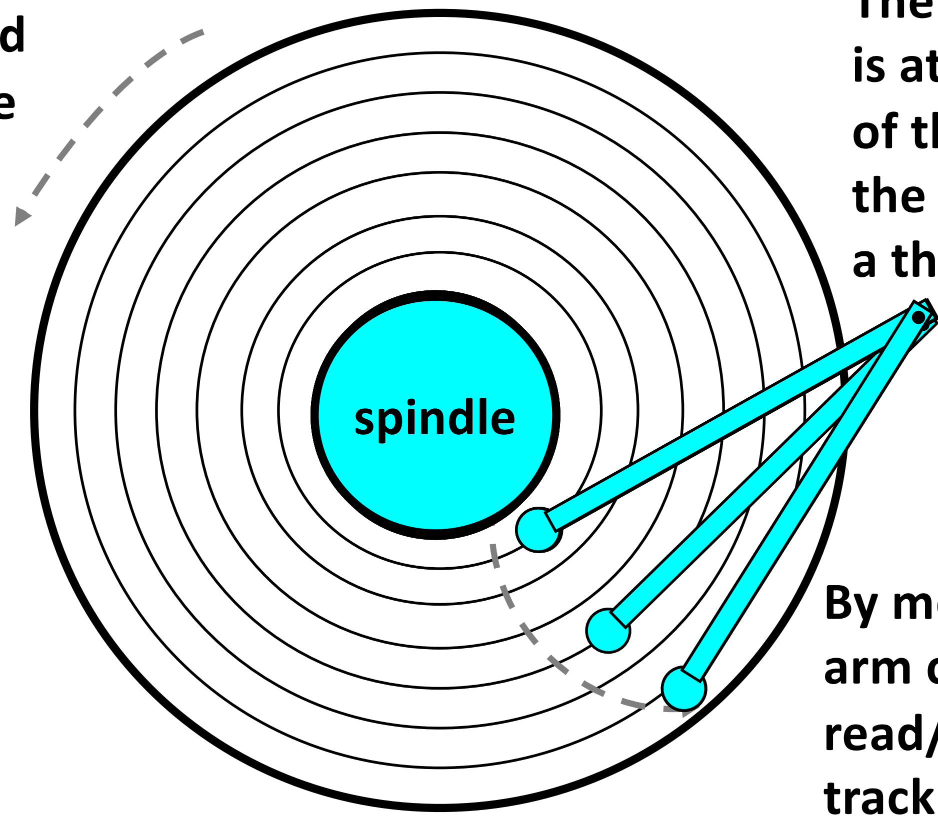


# Disk Operation (Single-Platter View)

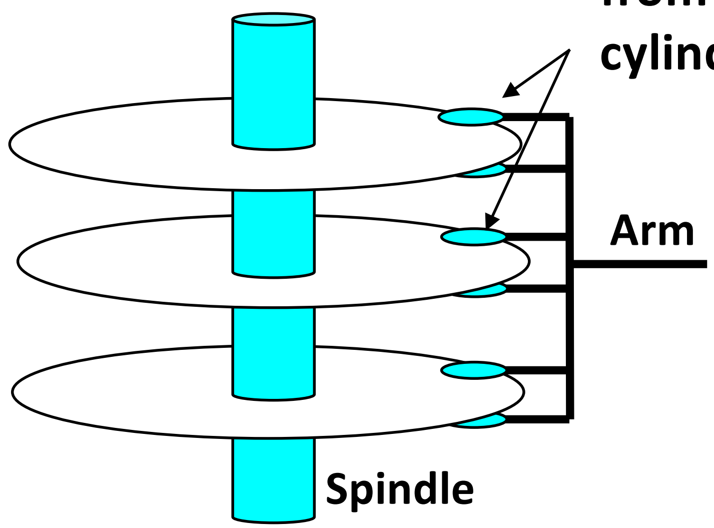
The disk surface spins at a fixed rotational rate

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

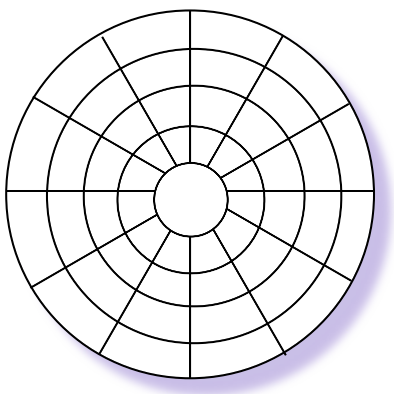
By moving radially, the arm can position the read/write head over any track.



# Disk Operation (Multi-Platter View)



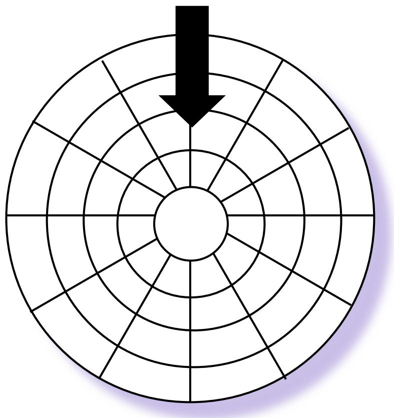
# Disk Structure - top view of single platter



**Surface organized into tracks**

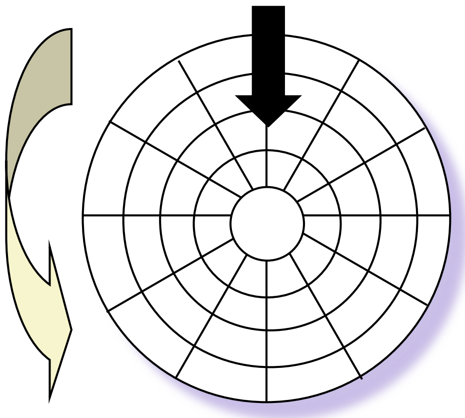
**Tracks divided into sectors**

# Disk Access



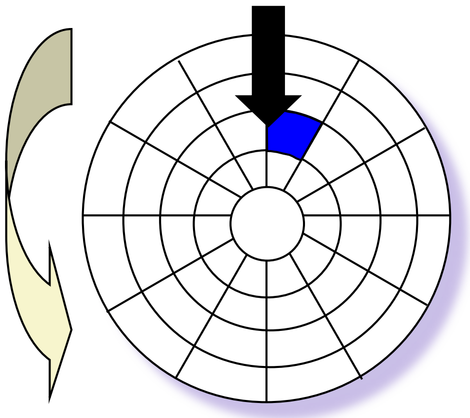
**Head in position above a track**

# Disk Access



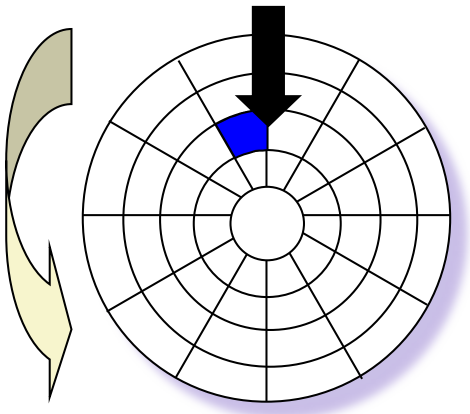
**Rotation is counter-clockwise**

# Disk Access – Read



**About to read blue sector**

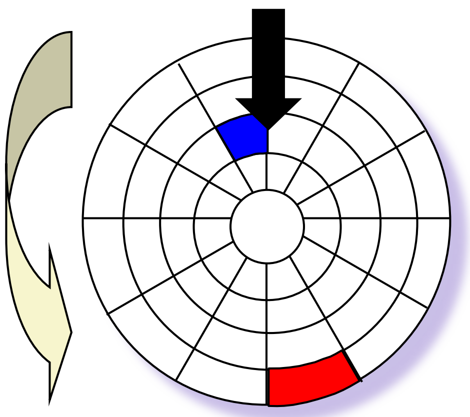
# Disk Access – Read



**After BLUE read**

**After reading blue sector**

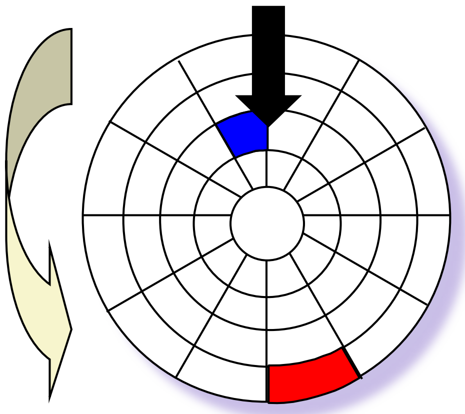
# Disk Access – Read



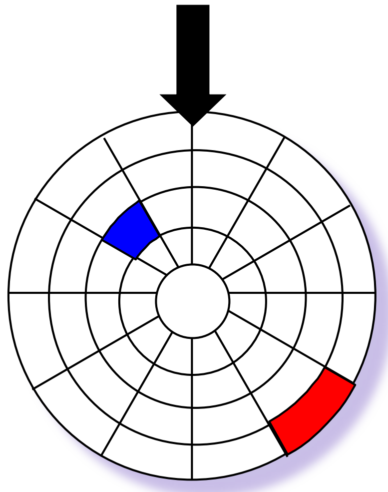
**After BLUE read**

**Red request scheduled next**

# Disk Access – Seek



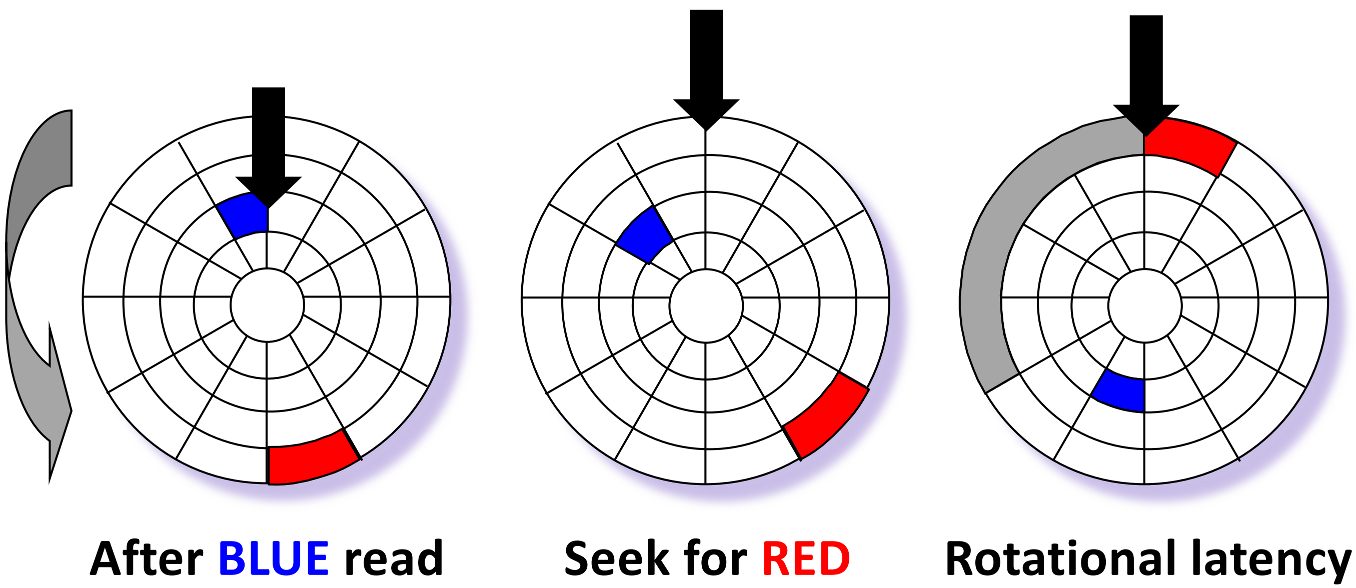
After **BLUE** read



Seek for **RED**

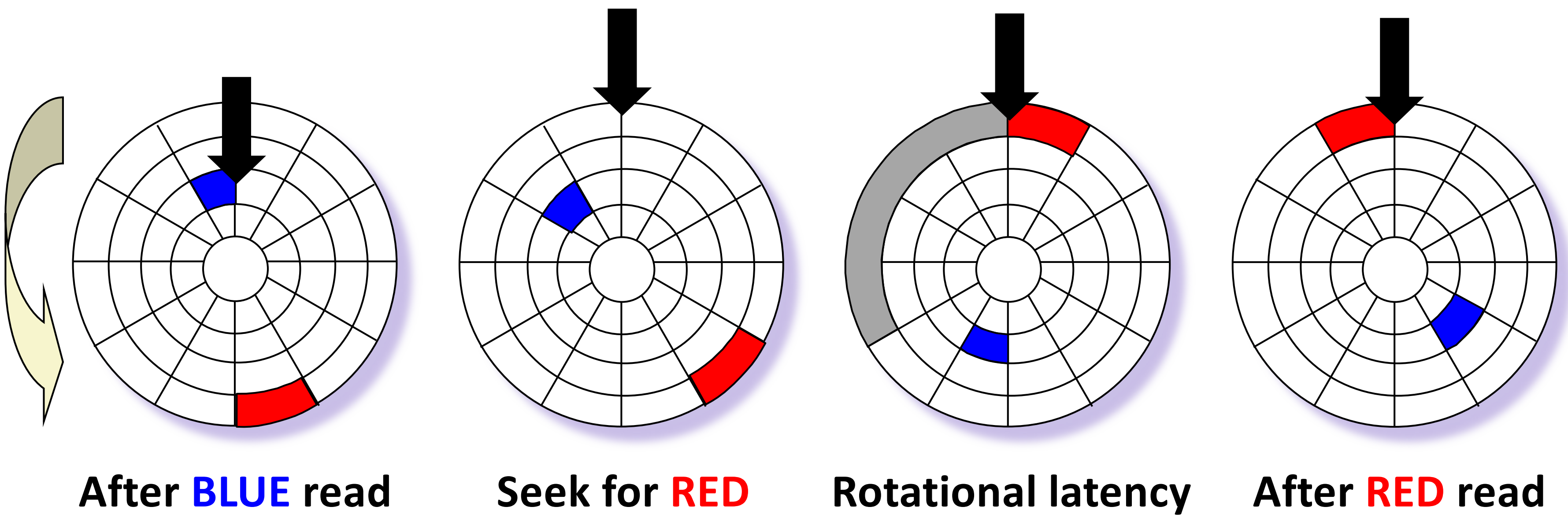
**Seek to red's track**

# Disk Access – Rotational Latency



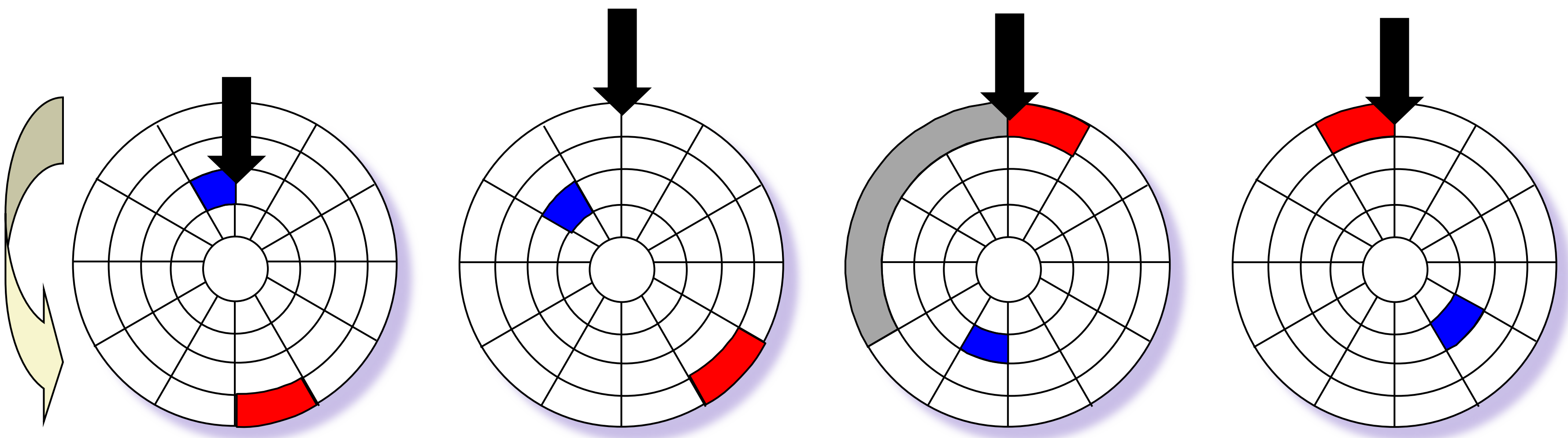

**Wait for red sector to rotate around**

# Disk Access – Read




**Complete read of red**

# Disk Access – Service Time Components



After BLUE read | Seek for RED | Rotational latency | After RED read

Data transfer | Seek | Rotational latency | Data transfer

# Disk Access Time

- Average time to access some target sector approximated by:
  - $T_{access} = T_{avg\ seek} + T_{avg\ rotation} + T_{avg\ transfer}$
- Seek time ($T_{avg\ seek}$)
  - Time to position heads over cylinder containing target sector.
  - Typical $T_{avg\ seek}$ is 3—9 ms
- Rotational latency ($T_{avg\ rotation}$)
  - Time waiting for first bit of target sector to pass under r/w head.
  - $T_{avg\ rotation}$ = 1/2 x 1/RPMs x 60 sec/1 min
  - Typical $T_{avg\ rotation}$ = 7,200 RPMs
- Transfer time ($T_{avg\ transfer}$)
  - Time to read the bits in the target sector.
  - $T_{avg\ transfer}$ = 1/RPM x 1/(avg # sectors/track) x 60 secs/1 min.

# Disk Access Time Example

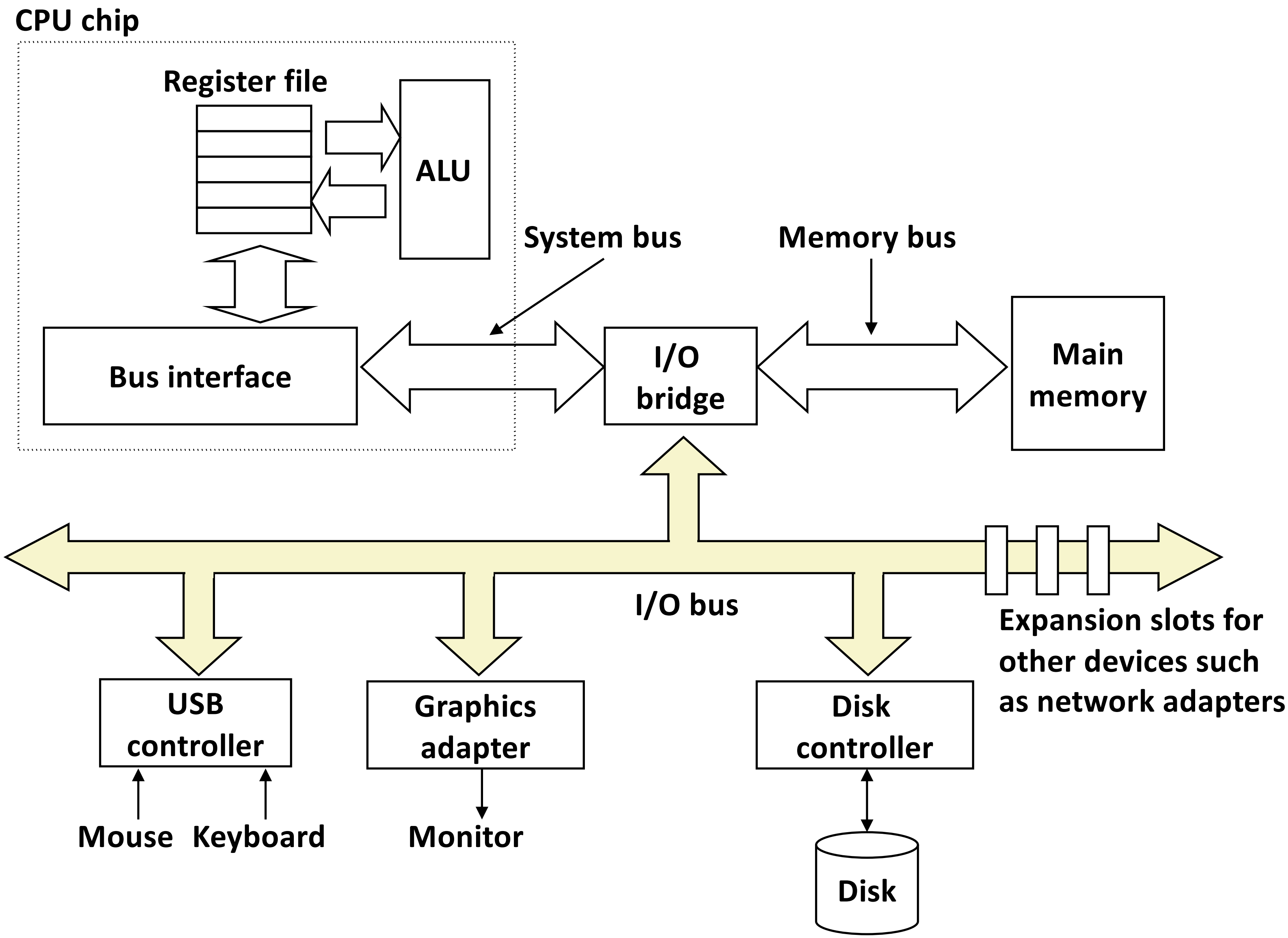
- Given:
  - Rotational rate = 7,200 RPM
  - Average seek time = 9 ms
  - Avg # sectors/track = 400
- Derived:
  - $T_{\text{avg rotation}}$ =
  - $T_{\text{avg transfer}}$ =
  - $T_{\text{access}}$ =

- **Average time to access some target 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 $T_{\text{avg seek}}$ is 3—9 ms
- **Rotational latency ($T_{\text{avg rotation}}$)**
  - Time waiting for first bit of target sector to pass under r/w head.
  - $T_{\text{avg rotation}}$ = 1/2 x 1/RPMs x 60 sec/1 min
  - Typical $T_{\text{avg rotation}}$ = 7,200 RPMs
- **Transfer time ($T_{\text{avg transfer}}$)**
  - Time to read the bits in the target sector.
  - $T_{\text{avg transfer}}$ = 1/RPM x 1/(avg # sectors/track) x 60 secs/1 min.

# Disk Access Time Example

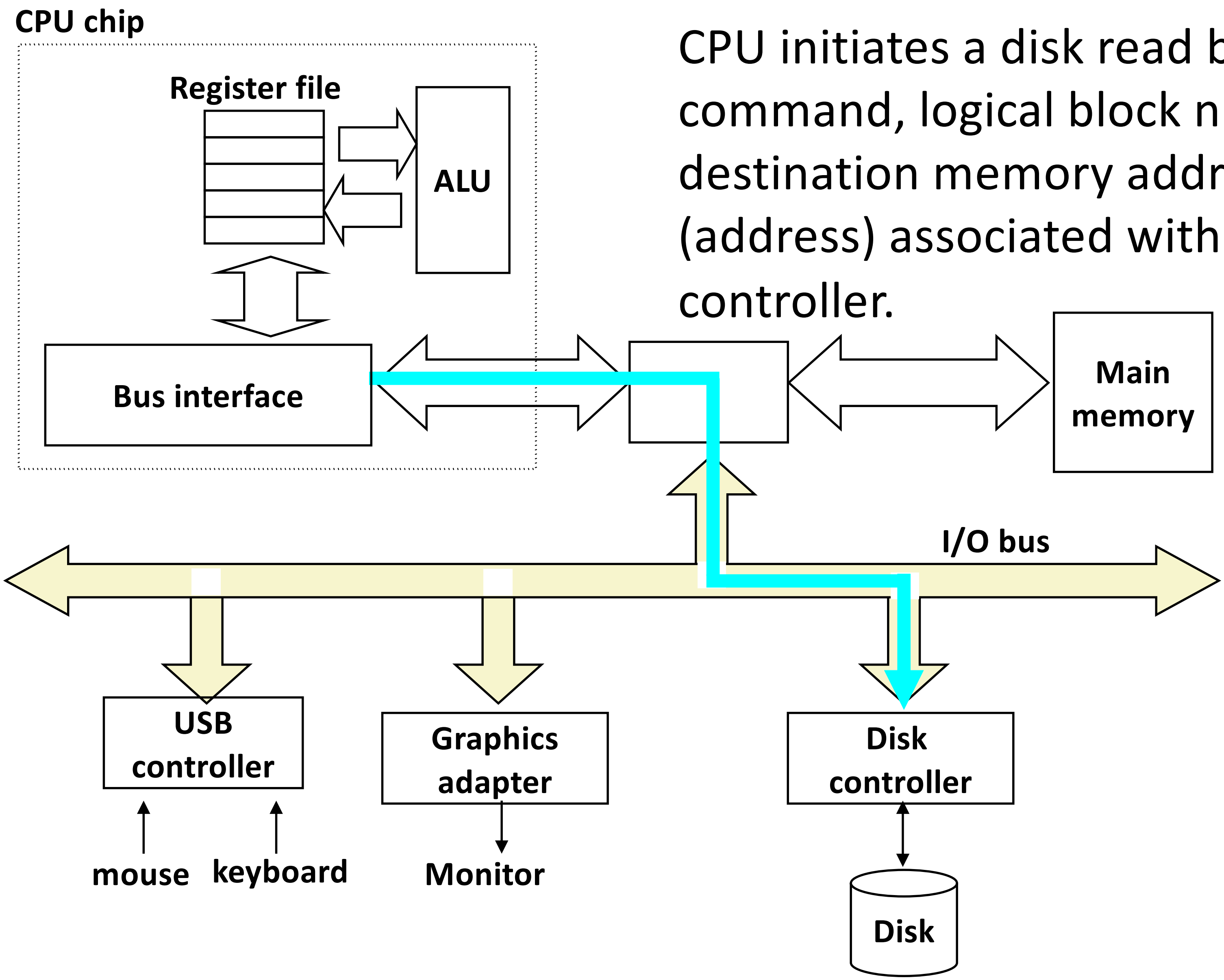
- Given:
  - Rotational rate = 7,200 RPM
  - Average seek time = 9 ms
  - Avg # sectors/track = 400
- Derived:
  - $T_{avg\ rotation}$ = 1/2 x (60 secs/7200 RPM) x 1000 ms/sec = 4 ms
  - $T_{avg\ transfer}$ = 60/7200 RPM x 1/400 secs/track x 1000 ms/sec = 0.02 ms
  - $T_{access}$ = 9 ms + 4 ms + 0.02 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***
    - ***Disk is about 40,000 times slower than SRAM,***
    - ***2,500 times slower then DRAM.***

# I/O Bus

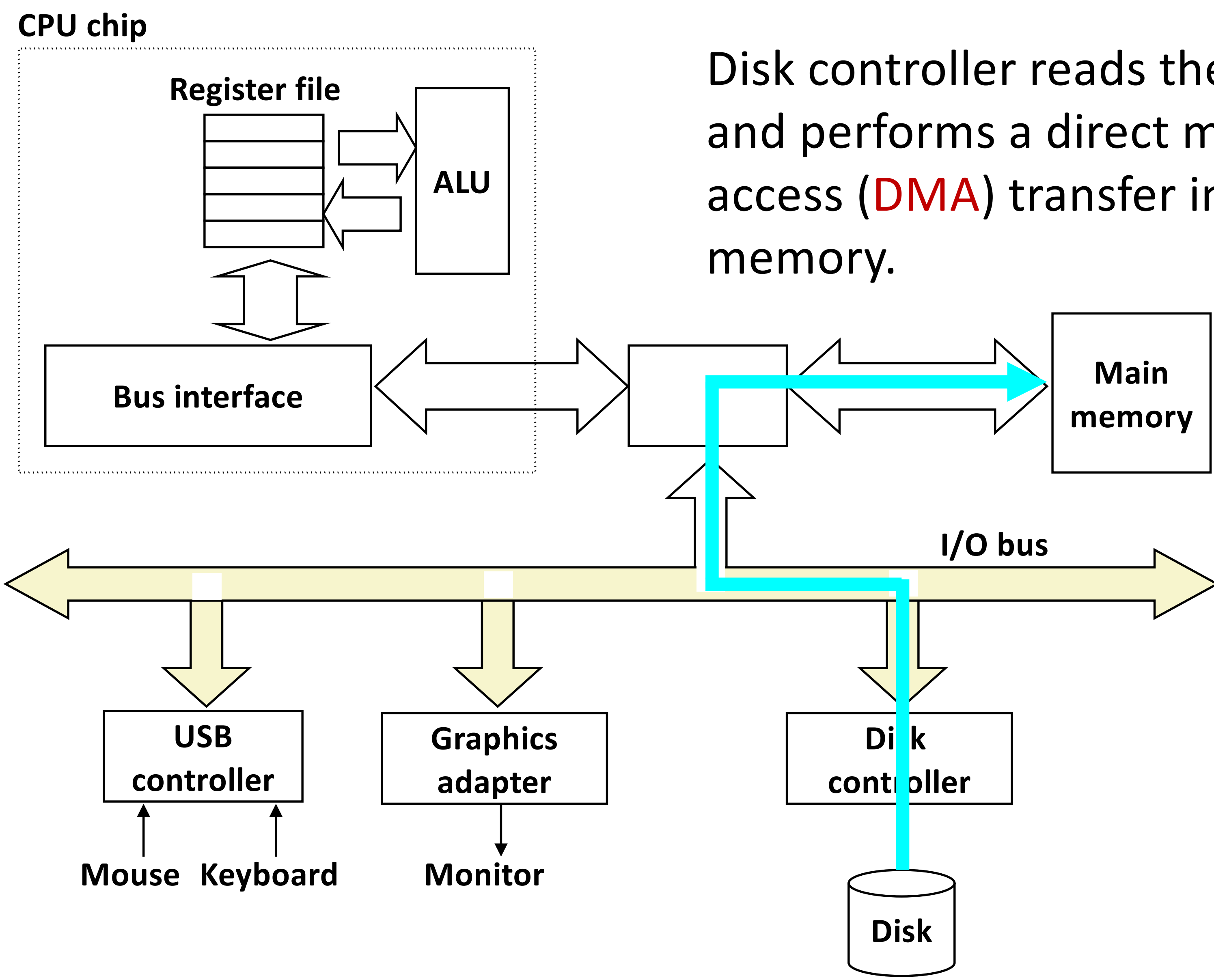

CPU chip
Register file
ALU
System bus
Memory bus
Bus interface
I/O bridge
Main memory
I/O bus
Expansion slots for other devices such as network adapters.
USB controller
Graphics adapter
Disk controller
Mouse
Keyboard
Monitor
Disk

# Reading a Disk Sector (1)

CPU initiates a disk read by writing a command, logical block number, and destination memory address to a port (address) associated with disk controller.



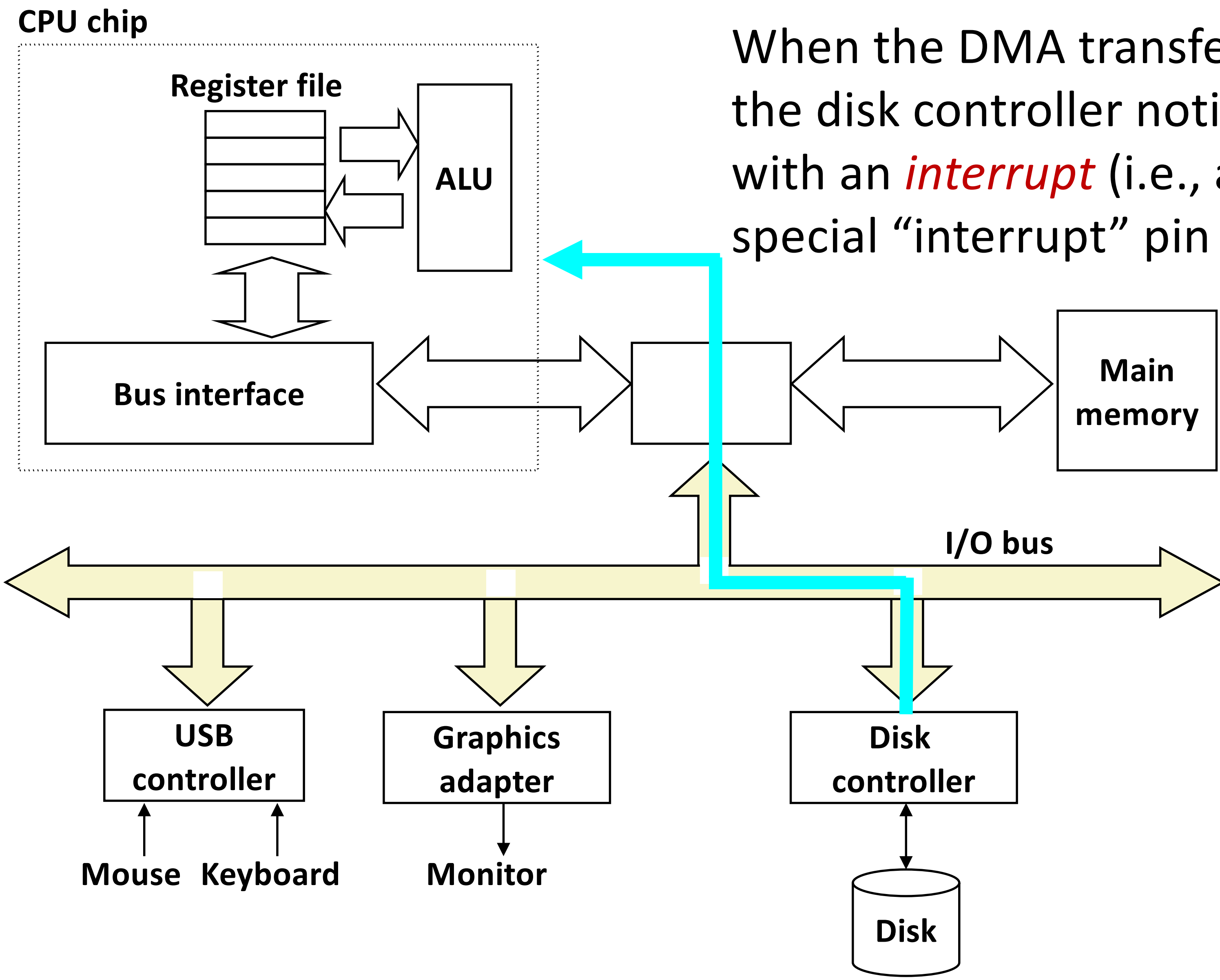
# Reading a Disk Sector (2)



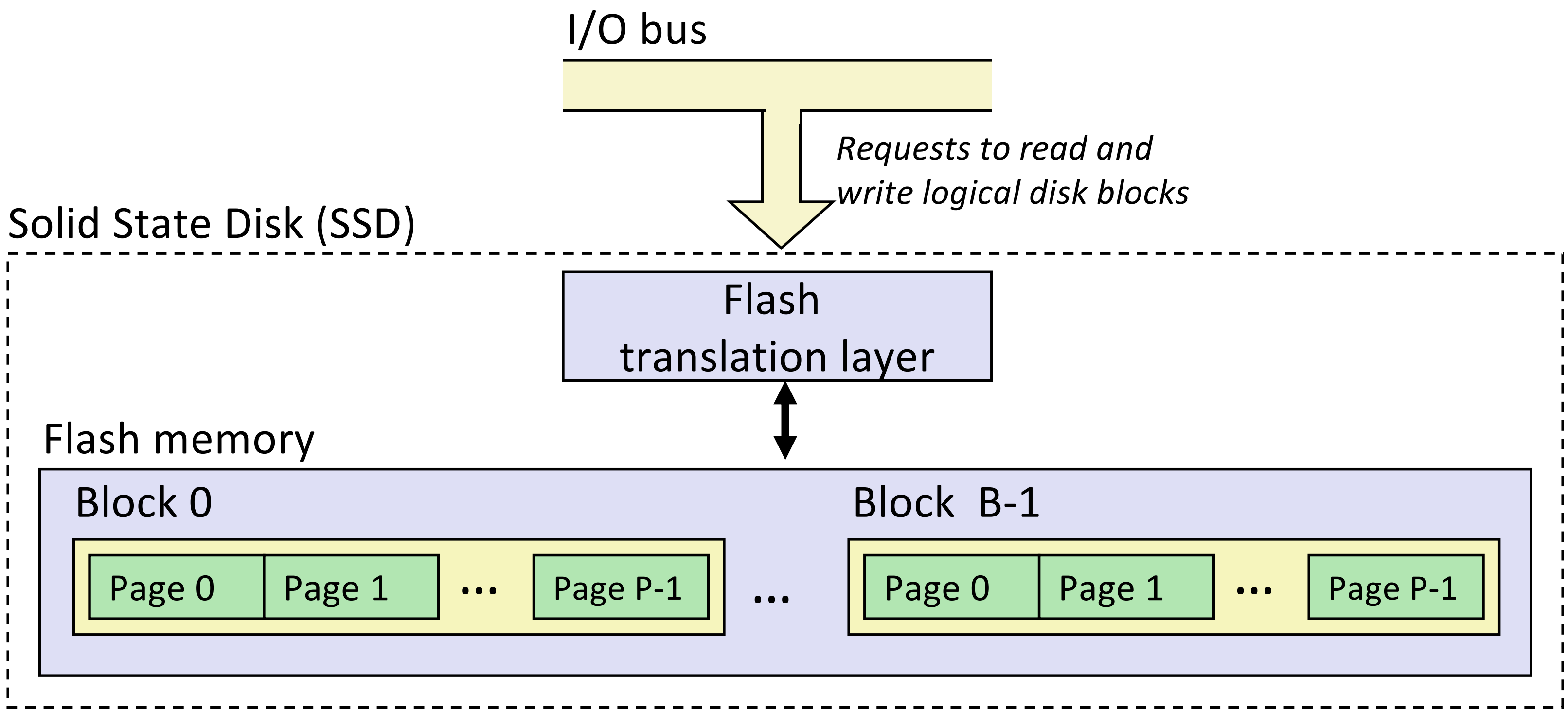

Disk controller reads the sector and performs a direct memory access (DMA) transfer into main memory.

# Reading a Disk Sector (3)

When the DMA transfer completes, the disk controller notifies the CPU with an *interrupt* (i.e., asserts a special "interrupt" pin on the CPU)



# Solid State Disks (SSDs)




- Pages: 512KB to 4KB, Blocks: 32 to 128 pages
- Data read/written in units of pages.
- Page can be written only after its block has been erased
- A block wears out after about 100,000 repeated writes.

# SSD Performance Characteristics

| | | | |
|---|---|---|---|
| **Sequential read tput** | **550 MB/s** | **Sequential write tput** | **470 MB/s** |
| **Random read tput** | **365 MB/s** | **Random write tput** | **303 MB/s** |
| **Avg seq read time** | **50 us** | **Avg seq write time** | **60 us** |

- Sequential access faster than random access
  - Common theme in the memory hierarchy
- Random writes are somewhat slower
  - Erasing a block takes a long time (~1 ms)
  - Modifying a block page requires all other pages to be copied to new block
  - In earlier SSDs, the read/write gap was much larger.

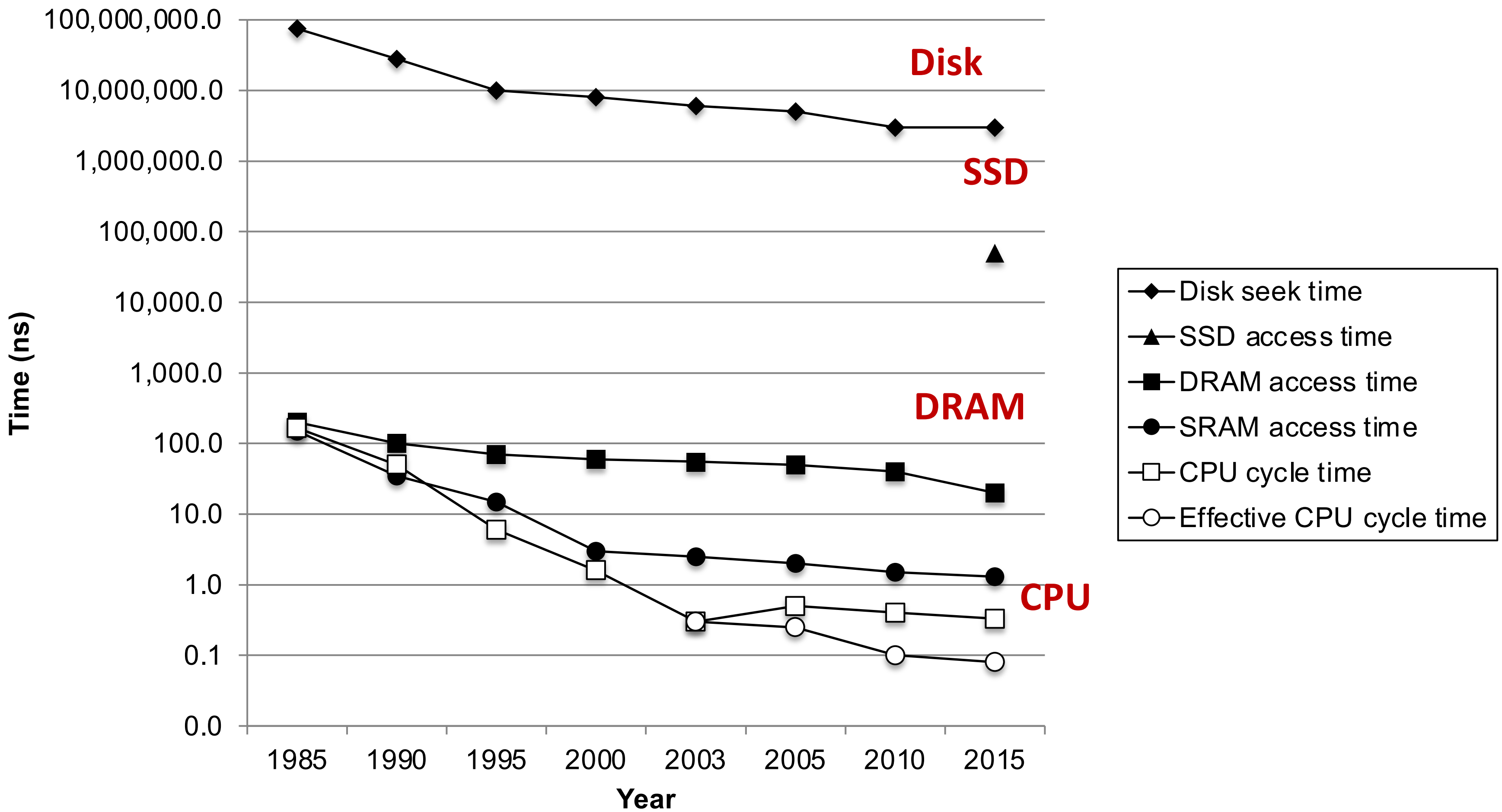
**Source: Intel SSD 730 product specification.**

# SSD Tradeoffs vs Rotating Disks

- Advantages
  - No moving parts → faster, less power, more rugged
- Disadvantages
  - Have the potential to wear out
    - Mitigated by "wear leveling logic" in flash translation layer
    - E.g. Intel SSD 730 guarantees 128 petabyte ($128 \times 10^{15}$ bytes) of writes before they wear out
  - More expensive, for now. (In 2015, about 30 times more expensive per byte. Only about 2-3 times more expensive in 2024.)
- Applications
  - Phones, laptops
  - Beginning to commonly appear in desktops and servers

# The CPU-Memory Gap

**Old graph, but notice that the gap *widens* between DRAM, disk, and CPU speeds.**



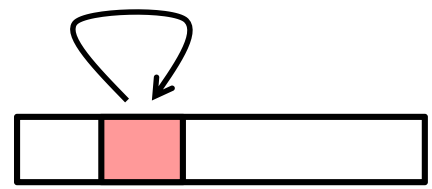
# Locality to the Rescue!

The key to bridging this CPU-Memory gap is a fundamental property of computer programs known as locality.

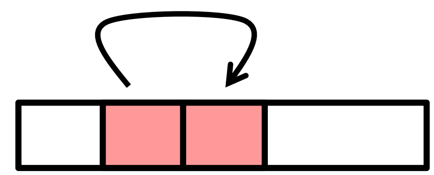
# Locality

- Principle of Locality: Programs tend to use data and instructions with addresses near or equal to those they have used recently

- Temporal locality:
  - Recently referenced items are likely to be referenced again in the near future



- Spatial locality:
  - Items with nearby addresses tend to be referenced close together in time



# Locality Example

```
sum = 0;
for (i = 0; i < n; i++)
    sum += a[i];
return sum;
```

- Data references
  - Reference array elements in succession (stride-1 reference pattern). **Spatial locality**
  - Reference variable **`sum`** each iteration. **Temporal locality**
- Instruction references
  - Reference instructions in sequence. **Spatial locality**
  - Cycle through loop repeatedly. **Temporal locality**