CS 33

Hardware: The Memory Hierarchy

 Copyright © 2020 Thomas W. Doeppner. All rights reserved.

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
- Static RAM (SRAM)
  - each cell stores a bit with a four- or six-transistor circuit
  - retains value indefinitely, as long as it is kept powered
  - relatively insensitive to electrical noise (EMI), radiation, etc.
  - faster and more expensive than DRAM
- Dynamic RAM (DRAM)
  - each cell stores bit with a capacitor; transistor is used for access
  - value must be refreshed every 10-100 ms
  - more sensitive to disturbances (EMI, radiation,…) than SRAM
  - slower and cheaper than SRAM

SRAM vs DRAM Summary

	Trans. per bit	Access time	Needs refresh?	Needs EDC?	Cost	Applications
SRAM	4 or 6	1X	No	Maybe	100x	Cache memories
DRAM	1	10X	Yes	Yes	1X	Main memories, frame buffers

- **EDC = error detection and correction**
  - **to cope with noise, etc.**

What's Inside A Disk Drive?

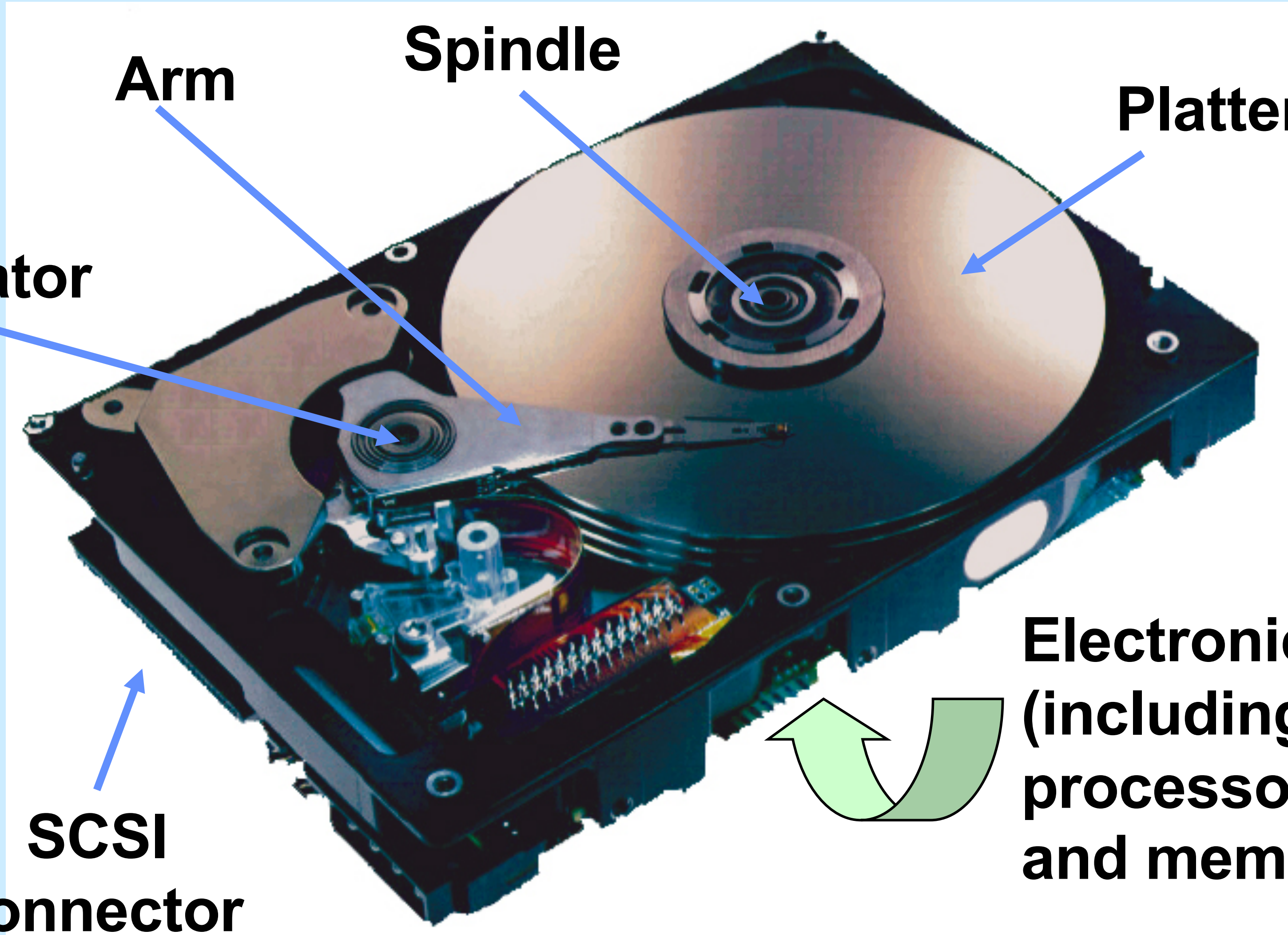
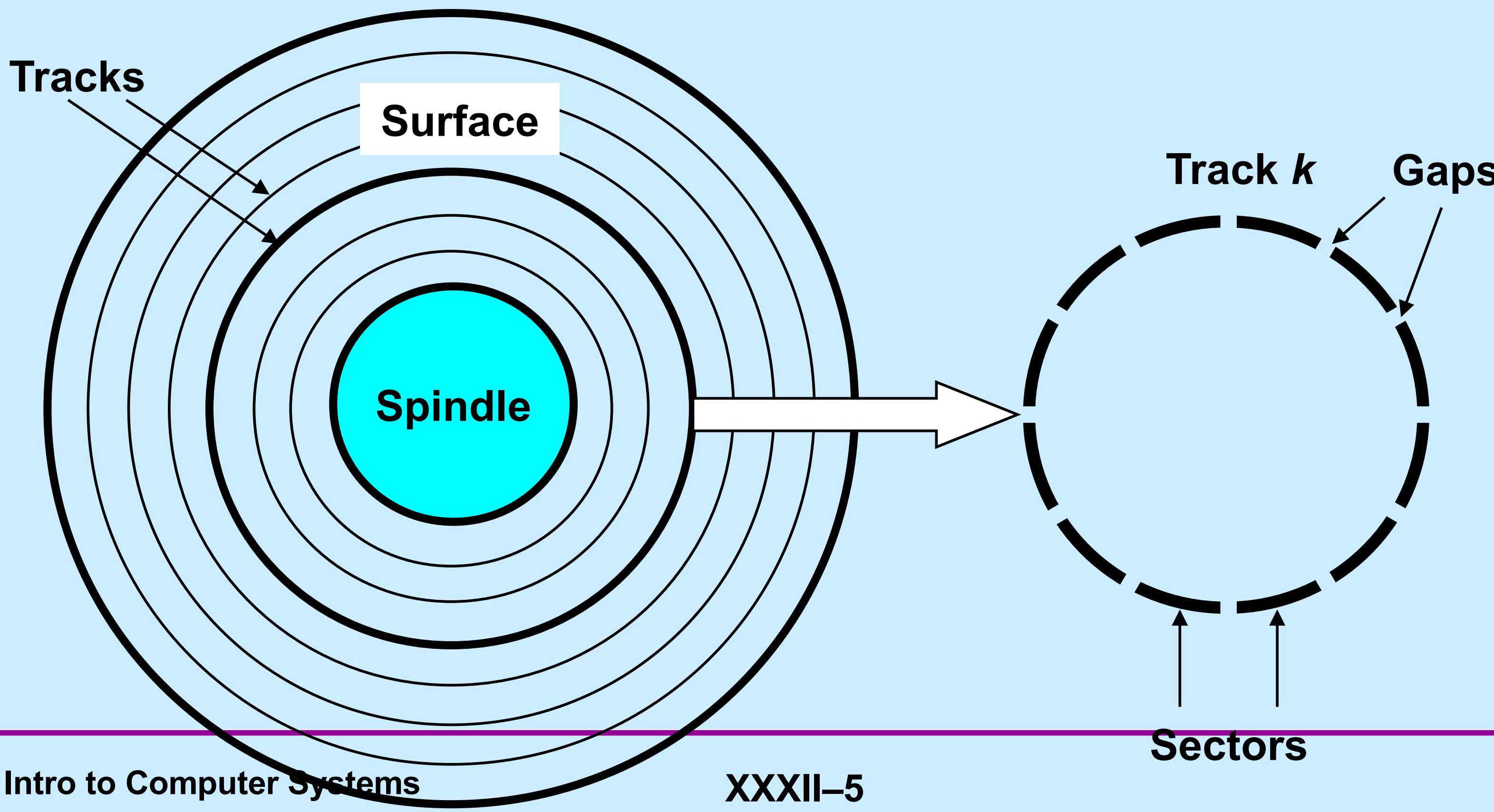


Image courtesy of Seagate Technology

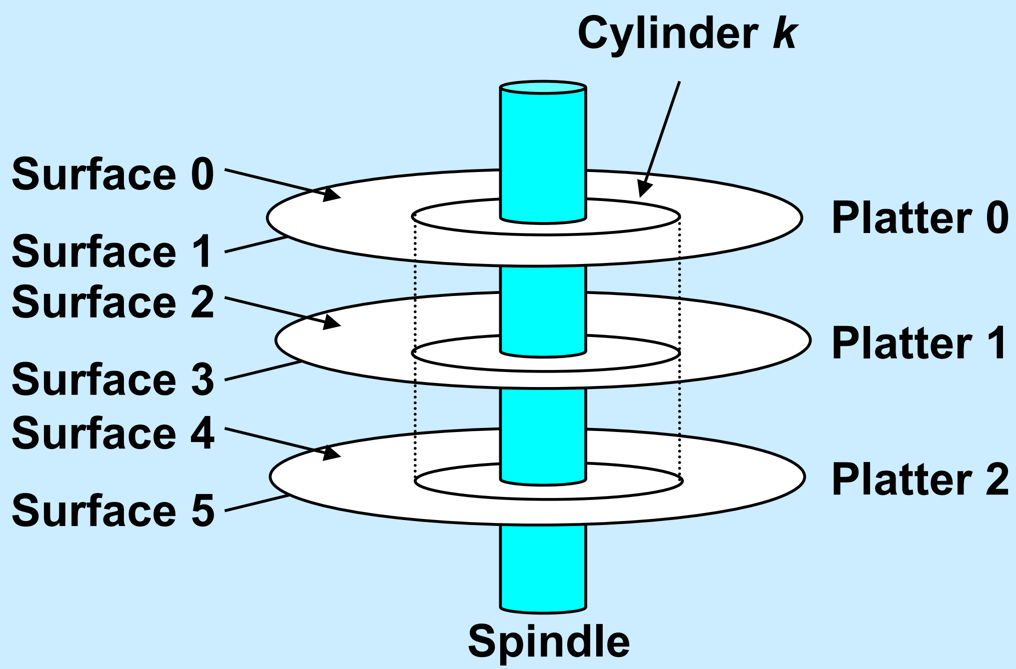
Disk Geometry

- 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 Capacity

- **Capacity**: maximum number of bits that can be stored
  - capacity expressed in units of gigabytes (GB), where 1 GB = 2^{30} Bytes ≈ 10^9 Bytes
- Capacity is determined by these technology factors:
  - **recording density** (bits/in): number of bits that can be squeezed into a 1 inch segment of a track
  - **track density** (tracks/in): number of tracks that can be squeezed into a 1 inch radial segment
  - **areal density** (bits/in^2): product of recording and track density
- Modern disks partition tracks into disjoint subsets called **recording zones**
  - each track in a zone has the same number of sectors, determined by the circumference of innermost track
  - each zone has a different number of sectors/track

Computing Disk Capacity

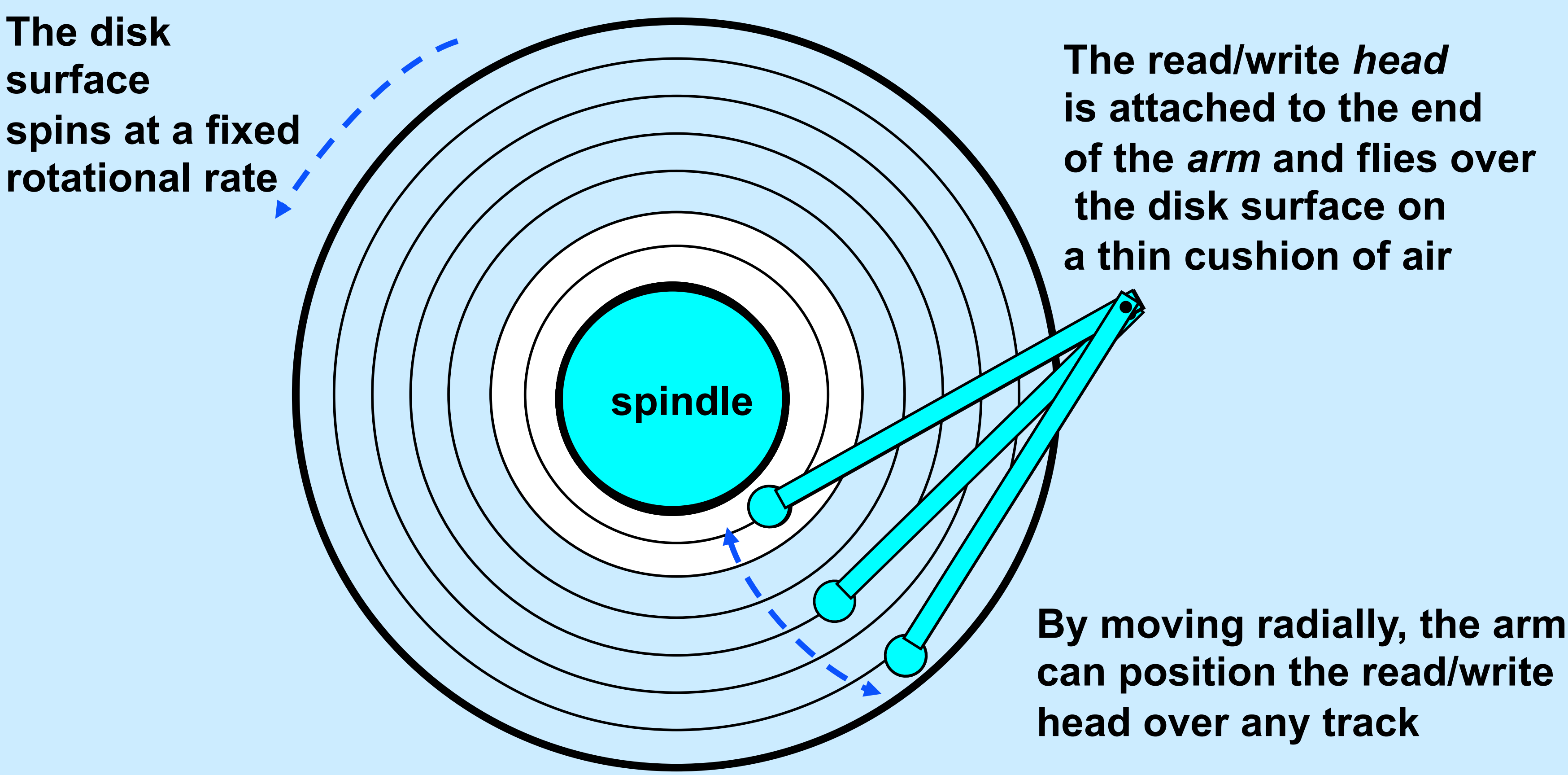
Capacity = (# bytes/sector) x (avg. # sectors/track) x
(# tracks/surface) x (# surfaces/platter) x
(# platters/disk)

Example:

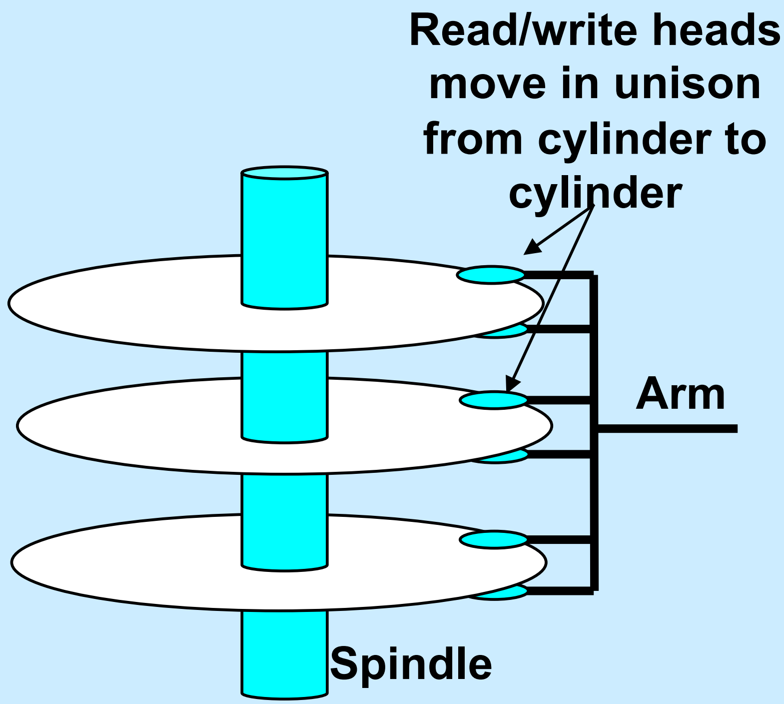
- 512 bytes/sector
- 600 sectors/track (on average)
- 40,000 tracks/surface
- 2 surfaces/platter
- 5 platters/disk

Capacity = 512 x 600 x 40000 x 2 x 5
= 122,880,000,000
= 113.88 GB

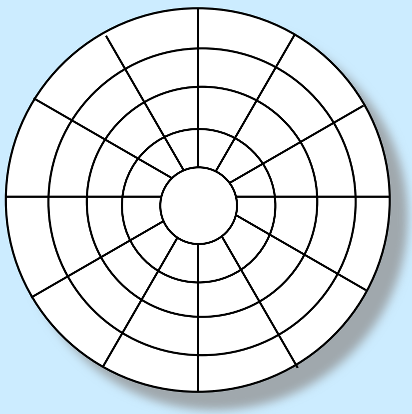
Disk Operation (Single-Platter View)



Disk Operation (Multi-Platter View)



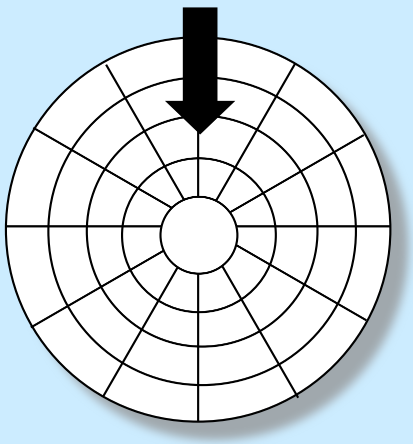
Disk Structure: Top View of Single Platter



Surface organized into tracks

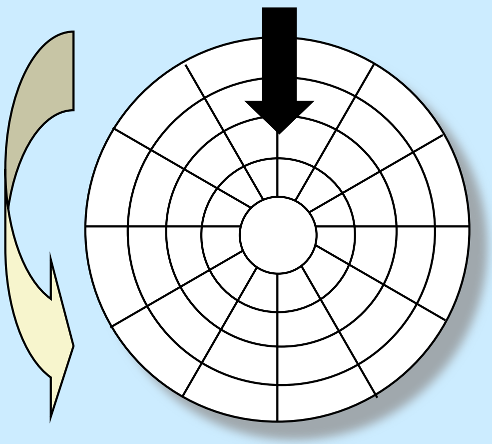
Tracks divided into sectors

Disk Access



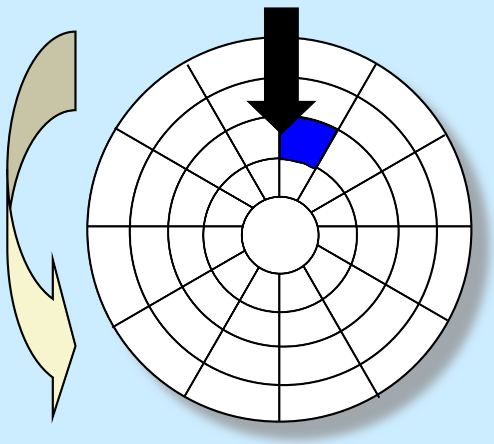
Head in position above a track

Disk Access



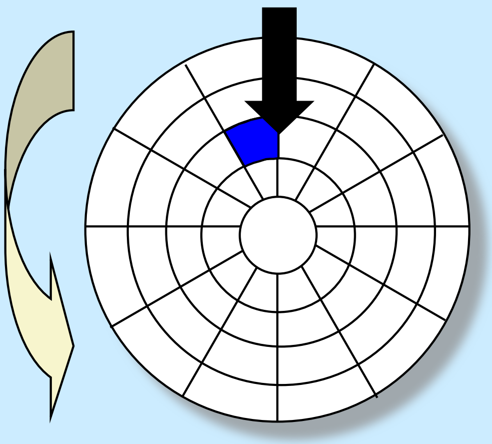
Rotation is counter-clockwise

Disk Access – Read



About to read blue sector

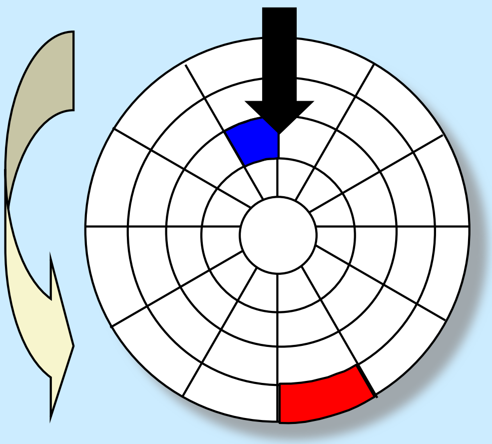
Disk Access – Read



After **BLUE**
read

After reading blue sector

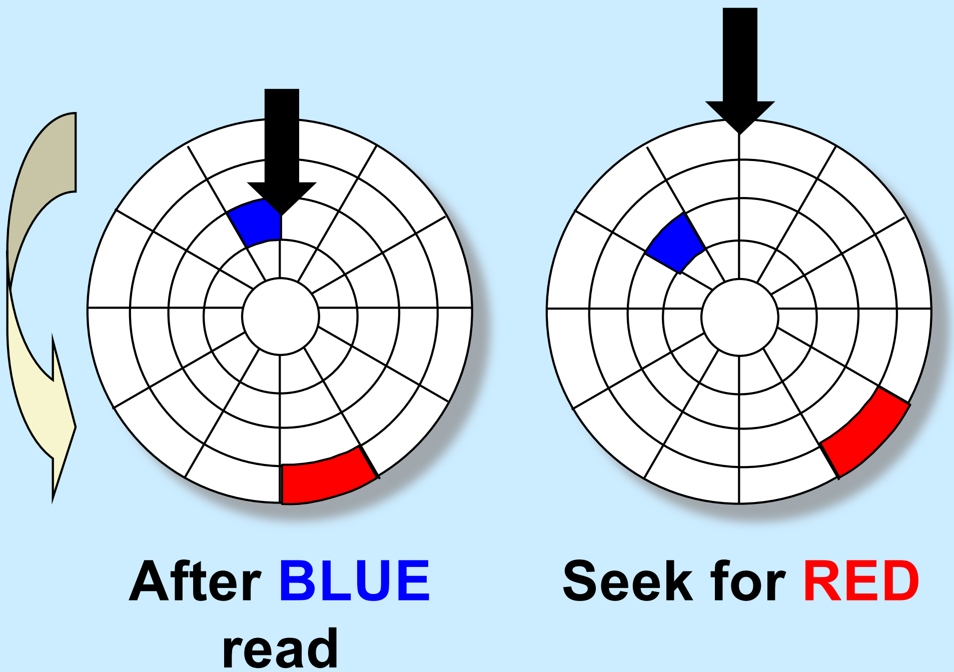
Disk Access – Read



After BLUE read

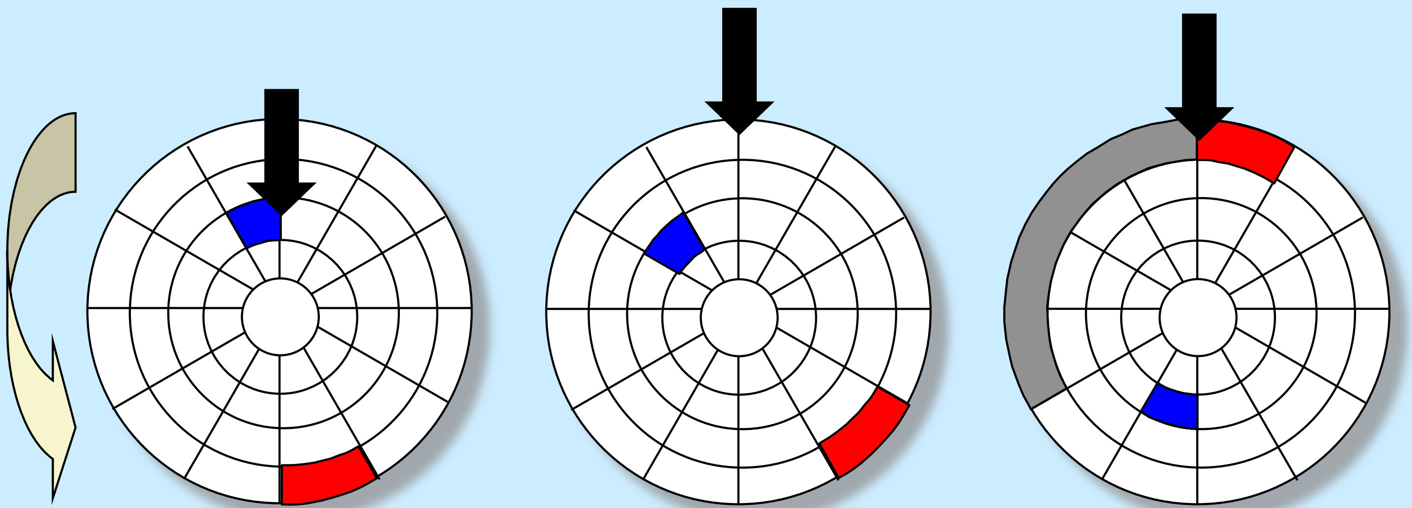
Red request scheduled next

Disk Access – Seek



Seek to red's track

Disk Access – Rotational Latency



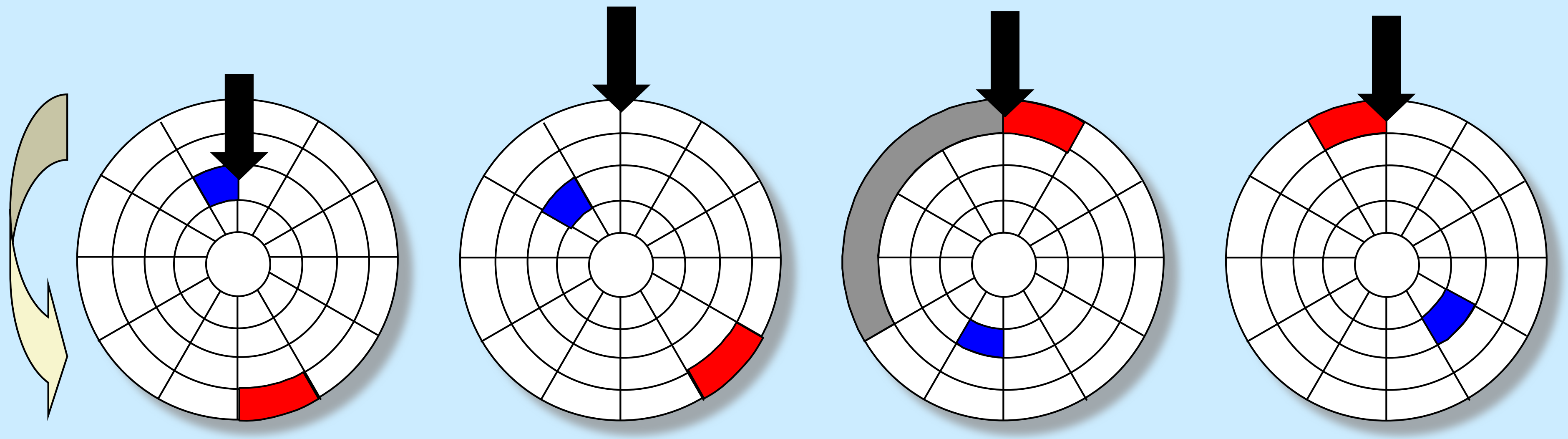
After **BLUE** read

Seek for **RED**

Rotational latency

Wait for red sector to rotate around

Disk Access – Read



After **BLUE** read

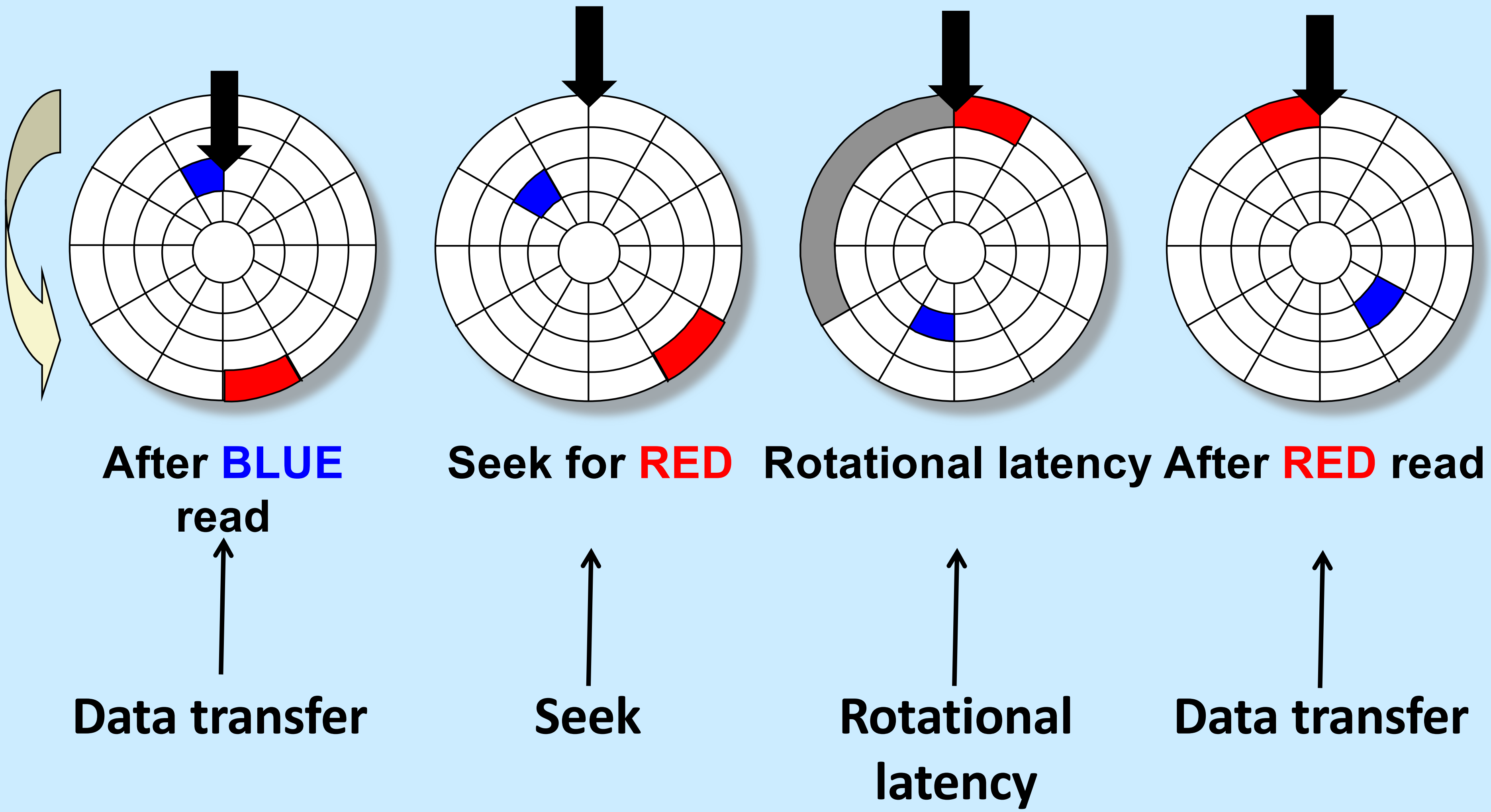
Seek for **RED**

Rotational latency

After **RED** read

Complete read of red

Disk Access – Service Time Components



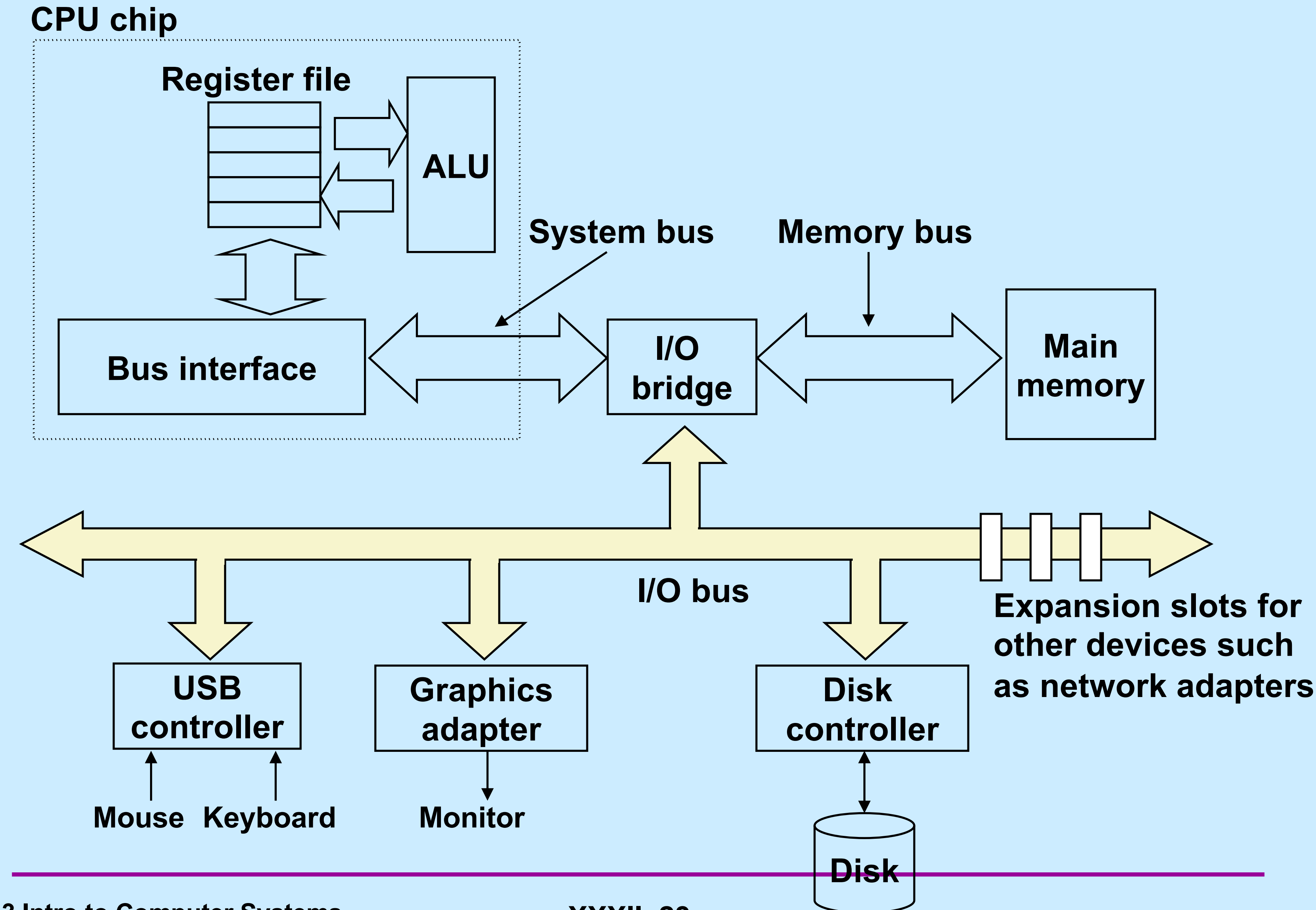
Disk Access Time

- Average time to access some target sector approximated by :
  - Taccess = Tavg seek + Tavg rotation + Tavg transfer
- Seek time (Tavg seek)
  - time to position heads over cylinder containing target sector
  - typical Tavg seek is 3–9 ms
- Rotational latency (Tavg rotation)
  - time waiting for first bit of target sector to pass under r/w head
  - typical rotation speed R = 7200 RPM
  - Tavg rotation = 1/2 x 1/R x 60 sec/1 min
- Transfer time (Tavg transfer)
  - time to read the bits in the target sector
  - Tavg transfer = 1/R x 1/(avg # sectors/track) x 60 secs/1 min

Disk Access Time Example

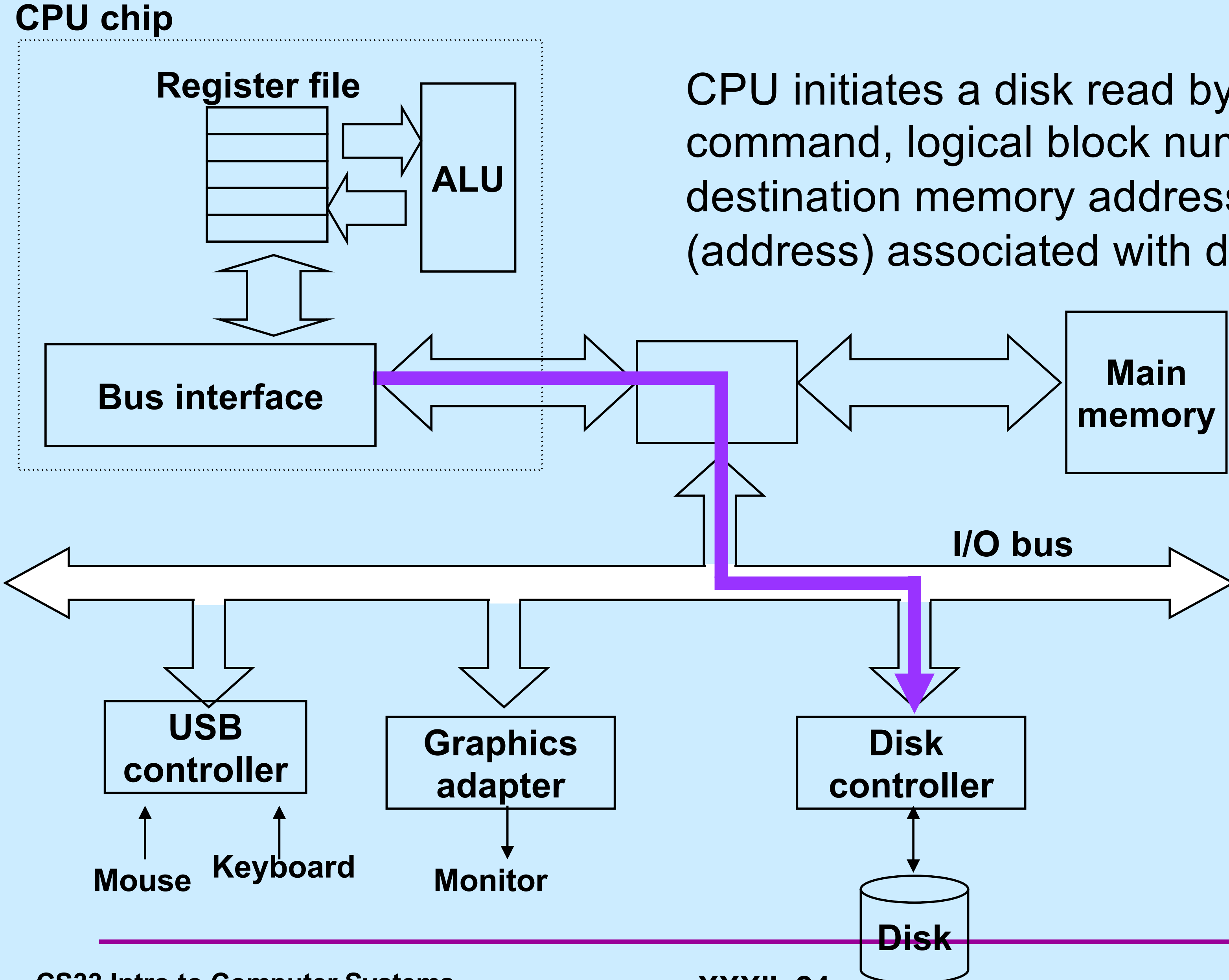
- Given:
  - rotational rate = 7,200 RPM
  - average seek time = 9 ms
  - avg # sectors/track = 600
- Derived:
  - Tavg rotation = 1/2 x (60 secs/7200 RPM) x 1000 ms/sec = 4 ms
  - Tavg transfer = 60/7200 RPM x 1/600 sects/track x 1000 ms/sec = 0.014 ms
  - Taccess = 9 ms + 4 ms + 0.014 ms
- Important 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 than DRAM

I/O Bus

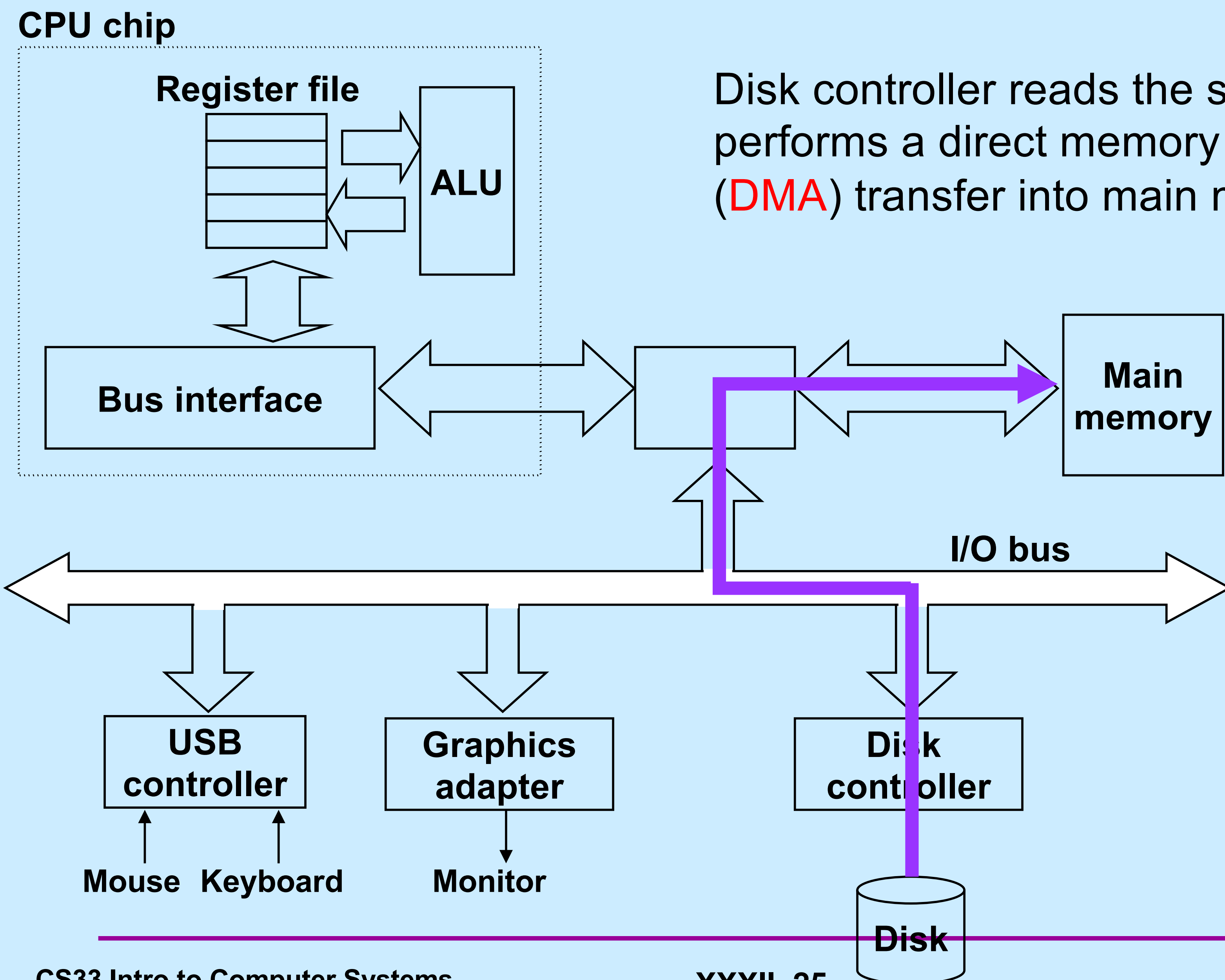

CPU chip
Register file
ALU
System bus
Memory bus
Bus interface
I/O bridge
Main memory
I/O bus
Expansion slots for other devices such as network adapters.
USB controller
Graphics adapter
Disk controller
Mouse
Keyboard
Monitor
Disk

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



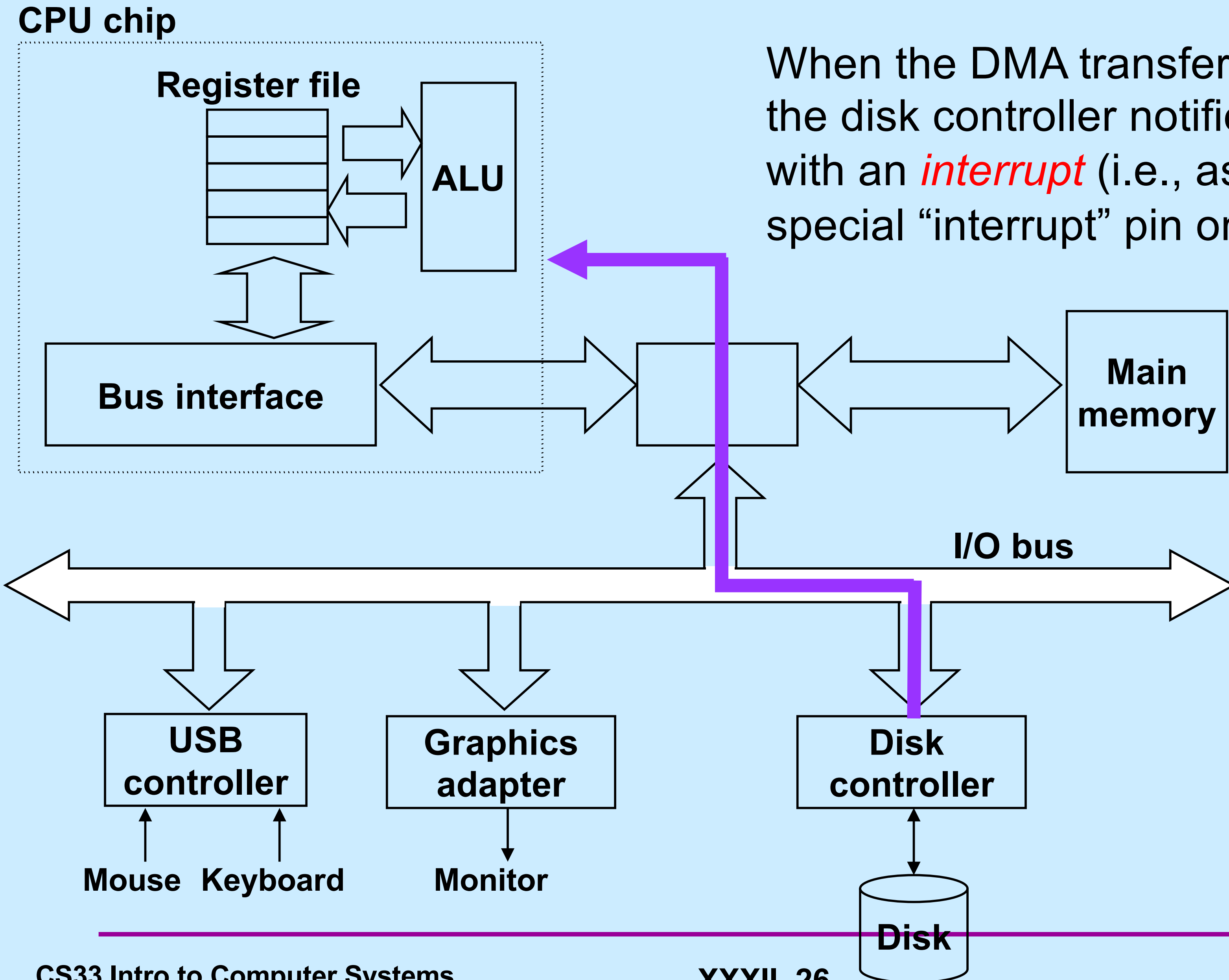
Reading a Disk Sector (2)



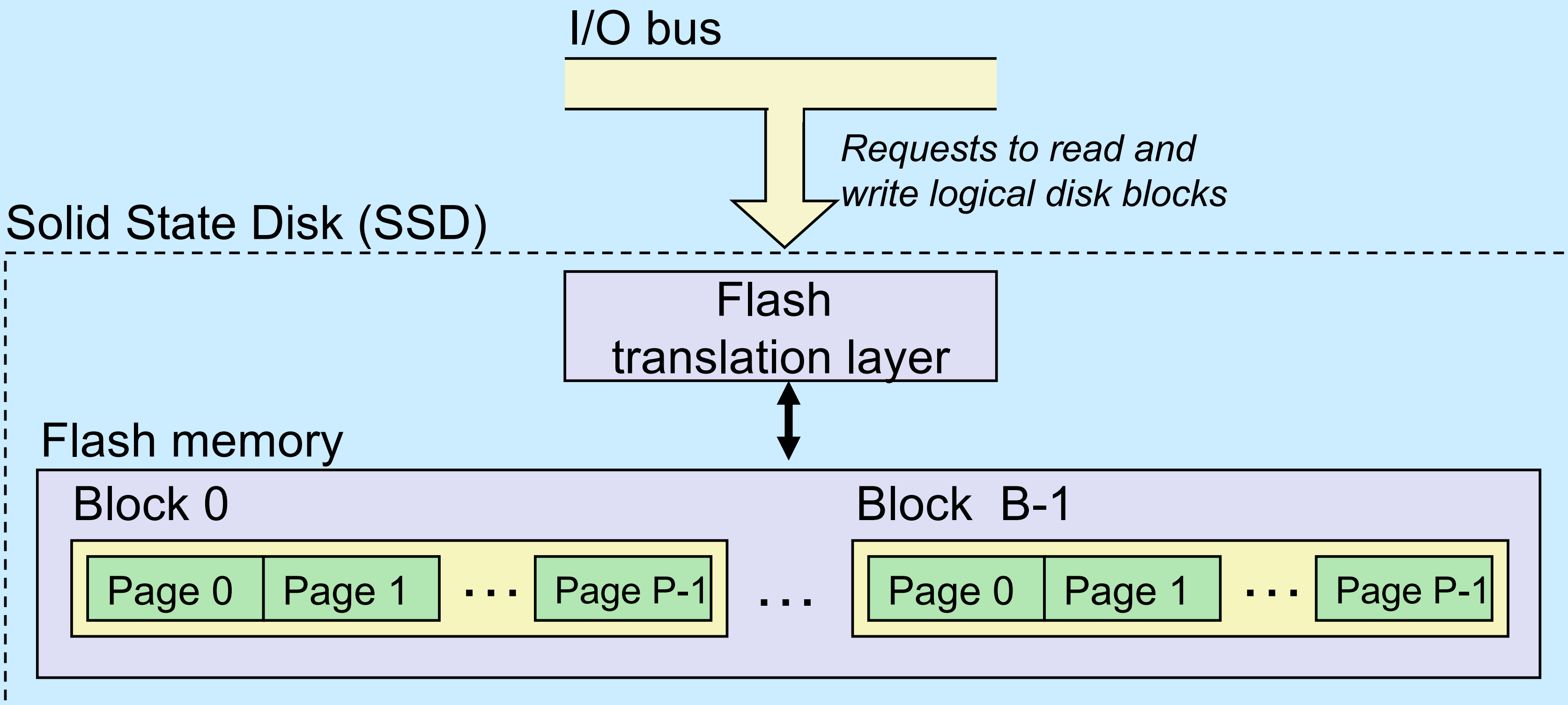
Disk controller reads the sector and performs a direct memory access (DMA) transfer into main memory

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 100,000 repeated writes**

SSD Performance Characteristics

Sequential read tput	250 MB/s	Sequential write tput	170 MB/s
Random read tput	140 MB/s	Random write tput	14 MB/s
Random read access	30 us	Random write access	300 us

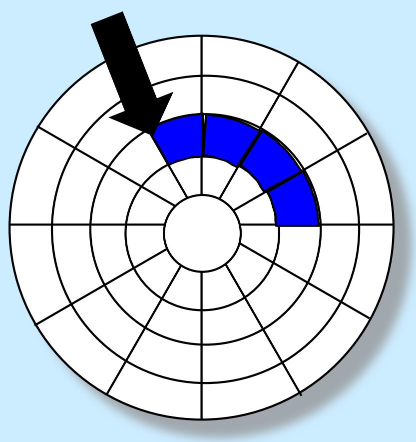
- Why are random writes so slow?
  - erasing a block is slow (around 1 ms)
  - modifying a page triggers a copy of all useful pages in the block
    - » find a used block (new block) and erase it
    - » write the page into the new block
    - » copy other pages from old block to the new block

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 X25 guarantees 1 petabyte (10^{15} bytes) of random writes before they wear out
  - in 2010, about 100 times more expensive per byte
  - in 2017, about 6 times more expensive per byte
- **Applications**
  - smart phones, laptops
  - Apple "Fusion" drives

Reading a File on a Rotating Disk

- **Suppose the data of a file are stored on consecutive disk sectors on one track**
  – **this is the best possible scenario for reading data quickly**
    » **single seek required**
    » **single rotational delay**
    » **all sectors read in a single scan**



 Copyright © 2020 Thomas W. Doeppner. All rights reserved.

Quiz 1

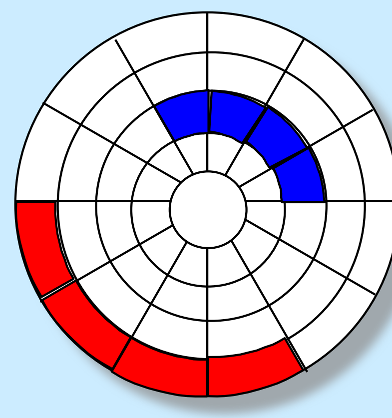
We have two files on the same (rotating) disk. The first file's data resides in consecutive sectors on one track, the second in consecutive sectors on another track. It takes a total of *t* seconds to read all of the first file then all of the second file.

Now suppose the files are read concurrently, perhaps a sector of the first, then a sector of the second, then the first, then the second, etc. Compared to reading them sequentially, this will take

a) a lot less time

b) around the same amount of time

c) a lot more time



Copyright © 2020 Thomas W. Doeppner. All rights reserved.

Quiz 2

We have two files on the same solid-state disk. Each file's data resides in consecutive blocks. It takes a total of *t* seconds to read all of the first file then all of the second file.

Now suppose the files are read concurrently, perhaps a block of the first, then a block of the second, then the first, then the second, etc. Compared to reading them sequentially, this will take

a) a lot less time

b) around the same amount of time

c) a lot more time

Copyright © 2020 Thomas W. Doeppner. All rights reserved.

Storage Trends

SRAM

Metric	1985	1990	1995	2000	2005	2010	2015	*2015:1985*
$/MB	2,900	320	256	100	75	60	25	116
access (ns)	150	35	15	3	2	1.5	1.3	115

DRAM

Metric	1985	1990	1995	2000	2005	2010	2015	*2015:1985*
$/MB	880	100	30	1	0.1	0.06	0.02	44,000
access (ns)	200	100	70	60	50	40	20	10
typical size (MB)	0.256	4	16	64	2,000	8,000	16,000	62,500

Disk

Metric	1985	1990	1995	2000	2005	2010	2015	*2015:1985*
$/GB	100,000	8,000	300	10	5	.3	0. 03	3,333,333
access (ms)	75	28	10	8	5	3	3	25
typical size (GB)	.01	.16	1	20	160	1,500	3,000	300,000

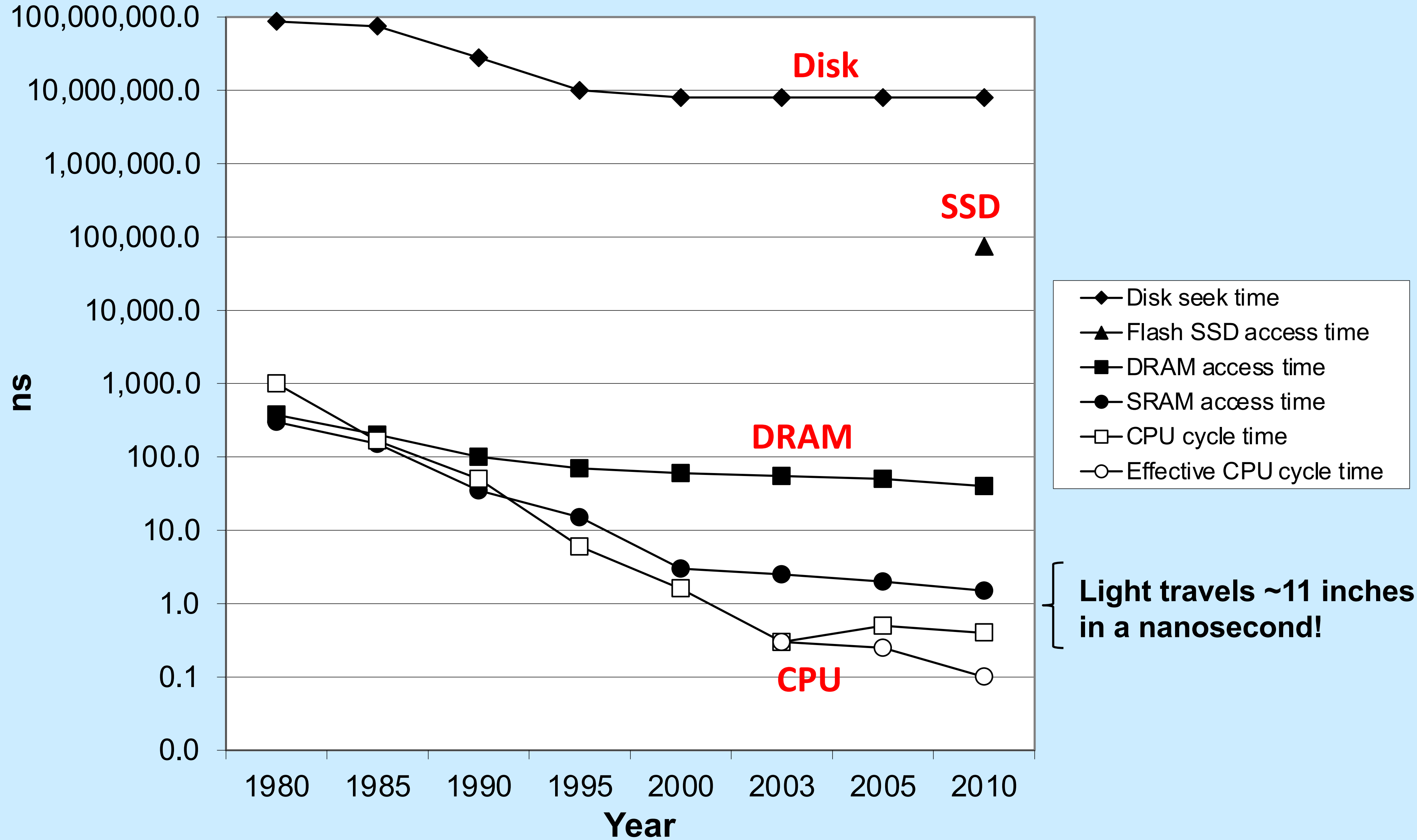
CPU Clock Rates

Inflection point in computer history when designers hit the “Power Wall”

	1985	1990	1995	2000	2003	2005	2015	*2015:1985*
CPU	286	386	Pentium	P-III	P-4	Core 2	Core i7	---
Clock rate (MHz)	6	20	150	600	3300	2000	3000	500
Cycle time (ns)	166	50	6	1.6	0.3	0.50	0.33	500
Cores	1	1	1	1	1	2	4	4
Effective cycle time (ns)	166	50	6	1.6	0.3	0.25	0.08	2075

The CPU-Memory Gap

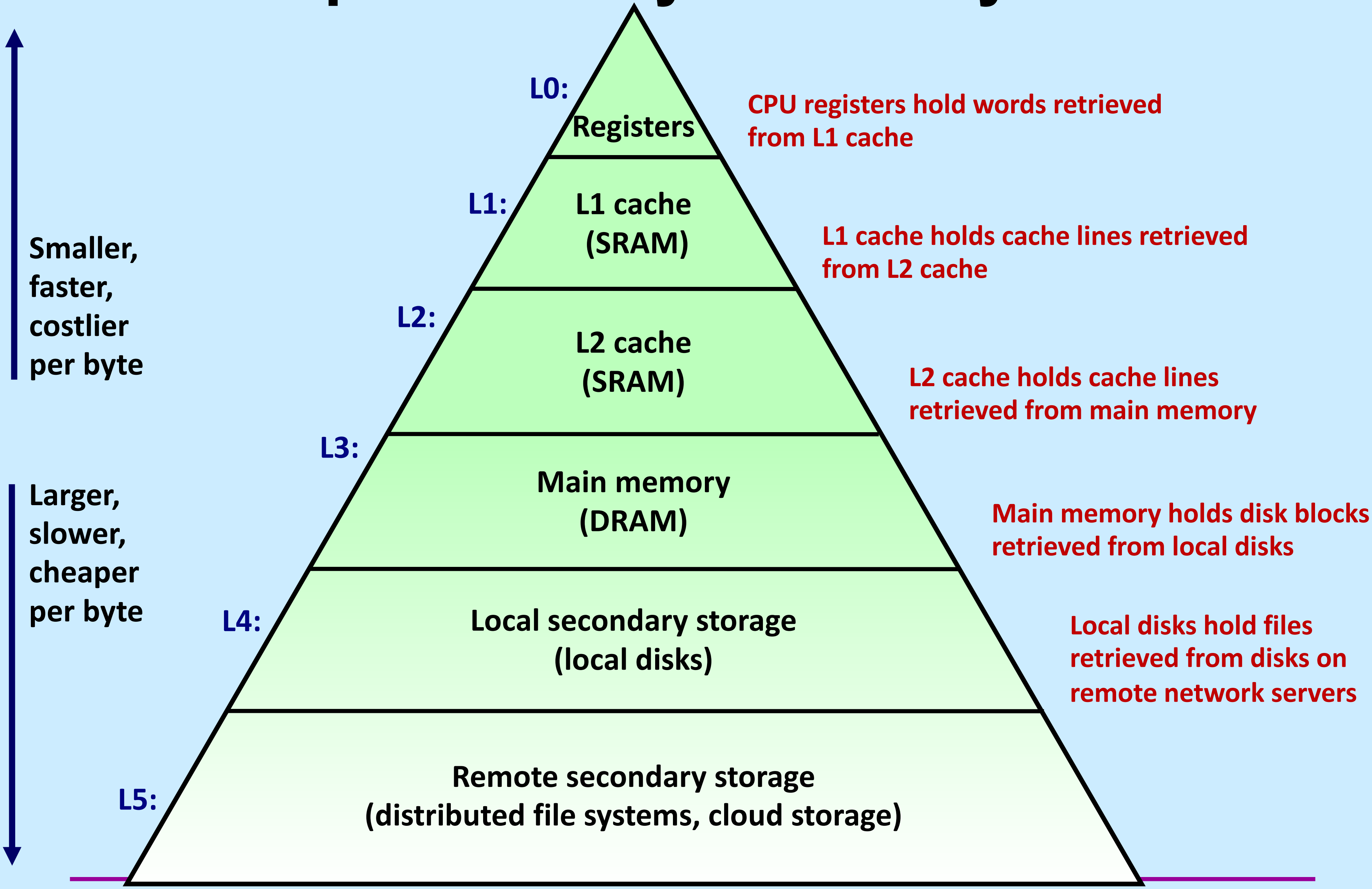
The gap widens between DRAM, disk, and CPU speeds



Memory Hierarchies

- Some fundamental and enduring properties of hardware and software:
  - fast storage technologies cost more per byte, have less capacity, and require more power (heat!)
  - the gap between CPU and main memory speed is widening
  - well written programs tend to exhibit good locality
- These fundamental properties complement each other beautifully
- They suggest an approach for organizing memory and storage systems known as a memory hierarchy

An Example Memory Hierarchy



Putting Things Into Perspective ...

- Reading from:
  - ... the L1 cache is like grabbing a piece of paper from your desk (3 seconds)
  - ... the L2 cache is picking up a book from a nearby shelf (14 seconds)
  - ... main system memory is taking a 4-minute walk down the hall to talk to a friend
  - ... a hard drive is like leaving the building to roam the earth for one year and three months

 Copyright © 2020 Thomas W. Doeppner. All rights reserved.

Disks Are Important

- Cheap
  - cost/byte much less than SSDs
- (fairly) Reliable
  - data written to a disk is likely to be there next year
- Sometimes fast
  - data in consecutive sectors on a track can be read quickly
- Sometimes slow
  - data in randomly scattered sectors takes a long time to read

 Copyright © 2020 Thomas W. Doeppner. All rights reserved.

Abstraction to the Rescue

- **Programs don't deal with sectors, tracks, and cylinders**
- **Programs deal with *files***
  - **maze.c rather than an ordered collection of sectors**
  - **OS provides the implementation**

Copyright © 2020 Thomas W. Doeppner. All rights reserved.

Implementation Problems

- Speed
  - use the hierarchy
    - copy files into RAM, copy back when done
  - optimize layout
    - put sectors of a file in consecutive locations
  - use parallelism
    - spread file over multiple disks
    - read multiple sectors at once

Copyright © 2020 Thomas W. Doeppner. All rights reserved.

Implementation Problems

- Reliability
  - computer crashes
    - » what you thought was safely written to the file never made it to the disk — it's still in RAM, which is lost
    - » worse yet, some parts made it back to disk, some didn't
      - you don't know which is which
      - on-disk data structures might be totally trashed
  - disk crashes
    - » you had backed it up … yesterday
  - you screw up
    - » you accidentally delete the entire directory containing your malloc solution

Copyright © 2020 Thomas W. Doeppner. All rights reserved.

Implementation Problems

- **Reliability solutions**
  - computer crashes
    - » transaction-oriented file systems
    - » on-disk data structures always in well defined states
  - disk crashes
    - » files stored redundantly on multiple disks
  - you screw up
    - » file system automatically keeps "snapshots" of previous versions of files

Copyright © 2020 Thomas W. Doeppner. All rights reserved.

You'll Soon Finish CS 33 …

- You might
  - celebrate

  - take another systems course
    - » 32
    - » 138
    - » 166
    - » 167

  - become a 33 TA

 Copyright © 2020 Thomas W. Doeppner. All rights reserved.

Systems Courses Next Semester

- **CS 32 (Intro to Software Engineering)**
  - you've mastered low-level systems programming
  - now do things at a higher level
  - learn software-engineering techniques using Java, XML, etc.
- **CS 138 (Distributed Systems)**
  - you now know how things work on one computer
  - what if you've got lots of computers?
  - some may have crashed, others may have been taken over by your worst (and smartest) enemy
- **CS 166 (Computer Systems Security)**
  - liked buffer?
  - you'll really like 166
- **CS 167/169 (Operating Systems)**
  - still mystified about what the OS does?
  - write your own!

Copyright © 2020 Thomas W. Doeppner. All rights reserved.

The End

Well, not quite …

Database is due on 12/10

No office hours Wednesday through Sunday

A few of us will be monitoring Piazza

Office hours resume next week

Happy coding and happy thanksgiving!

Copyright © 2020 Thomas W. Doeppner. All rights reserved.