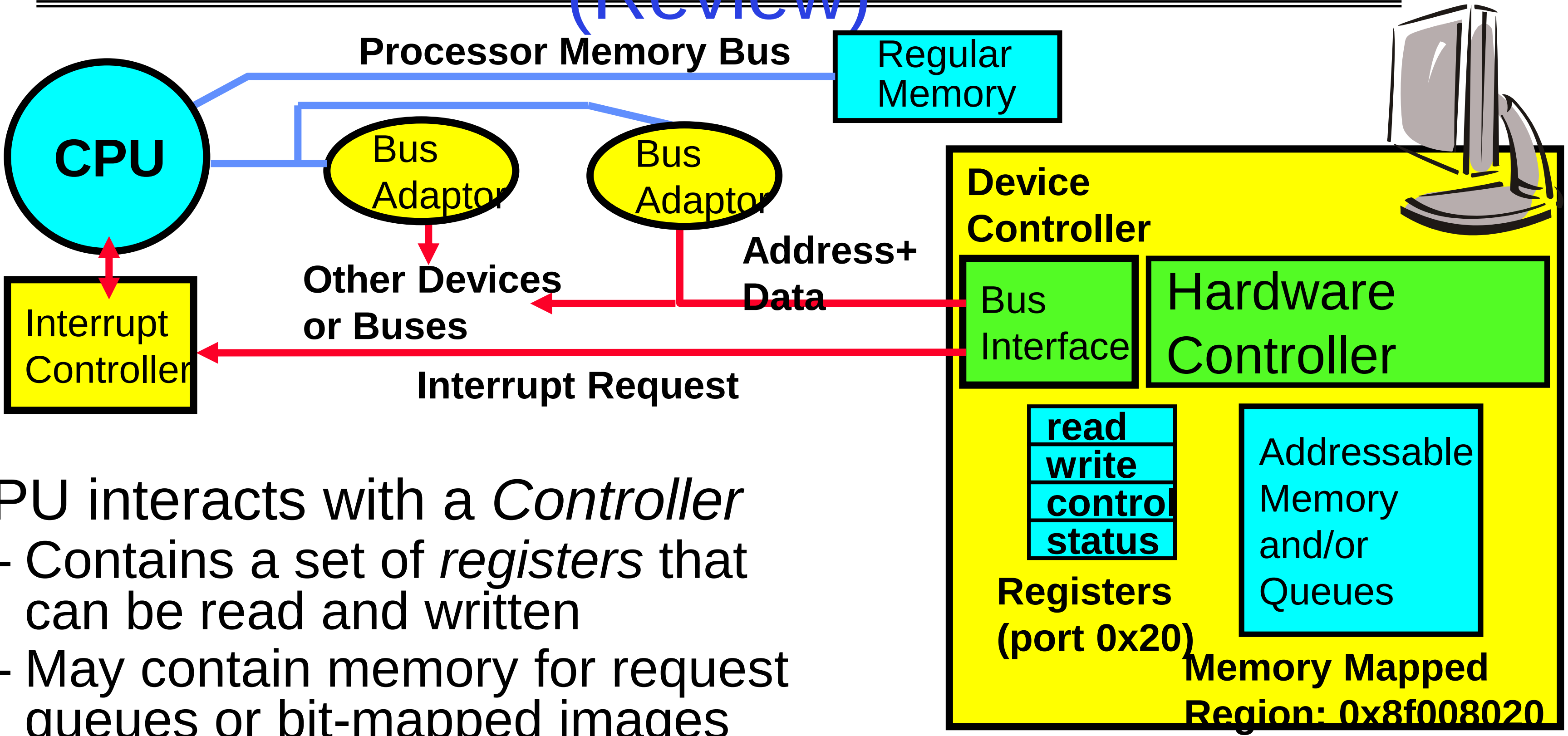
CSE150
Operating Systems
Lecture 19

Disk/SSDs
and File Systems

Want Standard Interfaces to Devices (Review)

- Block Devices: *e.g.*, disk drives, tape drives, DVD-ROM, flash
  - Access blocks of data
  - Commands include `open()`, `read()`, `write()`, `seek()`
  - Raw I/O or file-system access
  - Memory-mapped file access possible
- Character/Byte Devices: *e.g.*, keyboards, mice, serial ports, some USB devices
  - Single characters at a time
  - Commands include `get()`, `put()`
  - Libraries layered on top allow line editing
- Network Devices: *e.g.*, Ethernet, Wireless, Bluetooth
  - Different enough from block/character to have own interface
  - Unix and Windows include socket interface
    - » Separates network protocol from network operation
    - » Includes `select()` functionality

How Does the Processor Talk to Devices? (Review)



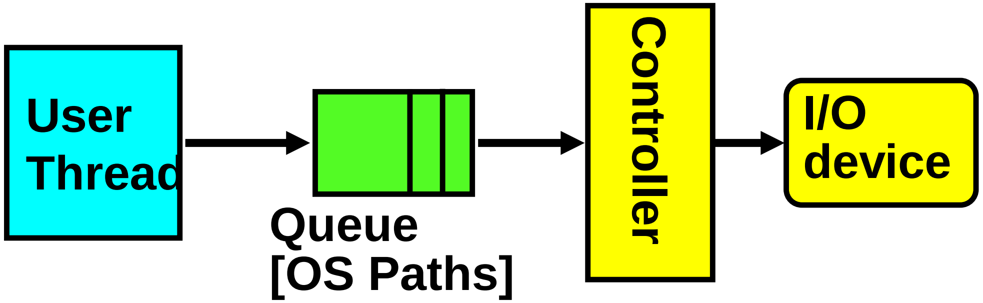
- CPU interacts with a *Controller*
  - Contains a set of *registers* that can be read and written
  - May contain memory for request queues or bit-mapped images
- Regardless of the complexity of the connections and buses, processor accesses registers in two ways (IA):
  - I/O instructions: in/out instructions (e.g., Intel's 0x21,AL)
  - Memory mapped I/O: load/store instructions
    - » Registers/memory appear in physical address space
    - » I/O accomplished with load and store instructions

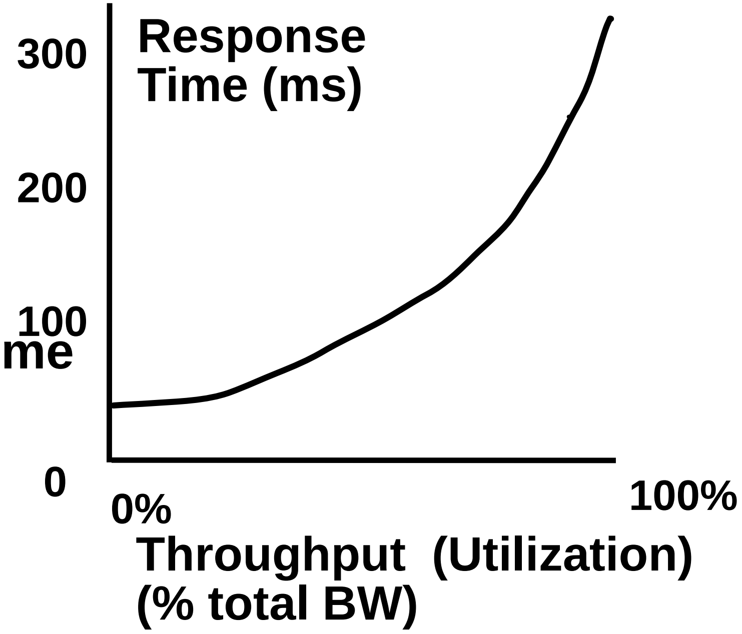
Today

- Disks and SSDs
  - Hardware Performance Parameters
  - Disk Scheduling
- Important Storage Policies and Patterns
- File System Structures

Slides adapted from CS162 Course Material at UC Berkeley

I/O Performance

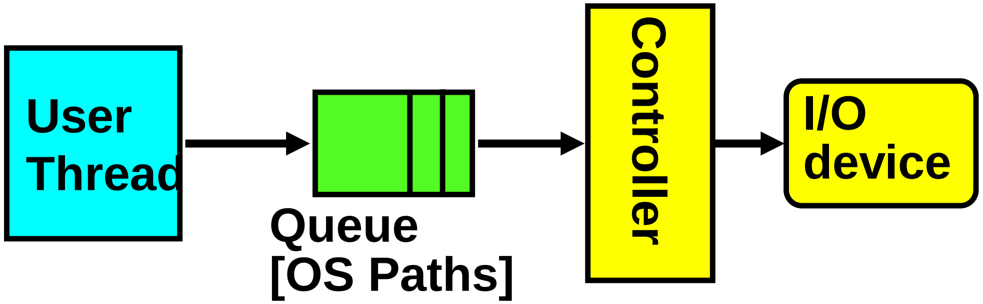


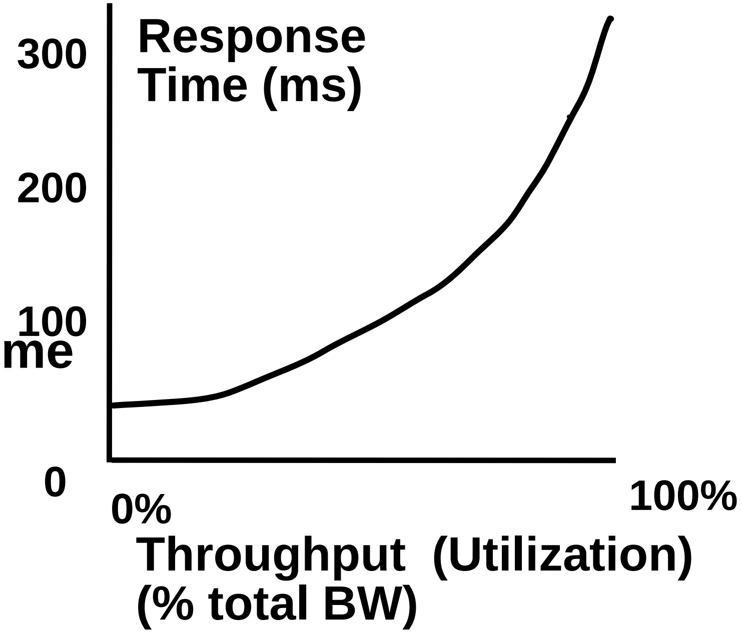


Response Time = Queue + I/O device service time

- Performance of I/O subsystem
  - Metrics: Response Time, Throughput
  - Contributing factors to latency:
    - » Software paths (can be loosely modeled by a queue)
    - » Hardware controller
    - » I/O device service time
- Queuing behavior:
  - Can lead to big increases of latency as utilization approaches 100%
  - Solutions?

I/O Performance

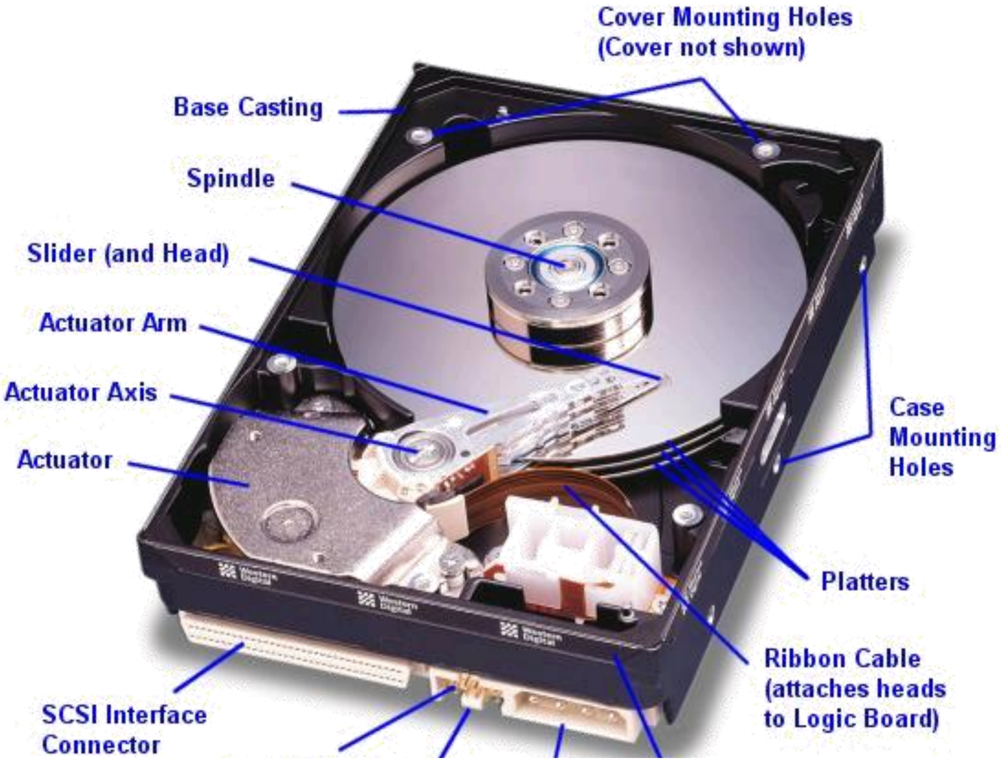




Response Time = Queue + I/O device service time

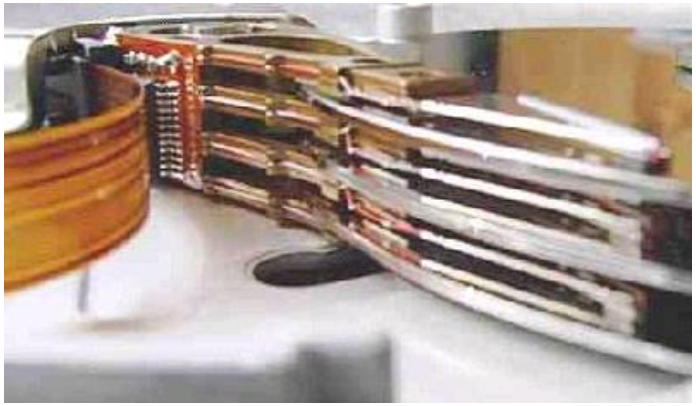
- Solutions?
  - Make everything faster ☺
  - Decouple systems
    - » multiple independent buses
    - » or tree-structured buses with higher root bandwidth
  - Buffering (as long as you don’t have to wait for it) and spooling
    - » Give the processor something to do that gets the data “closer” to its endpoint.

Hard Disk Drives (HDDs)



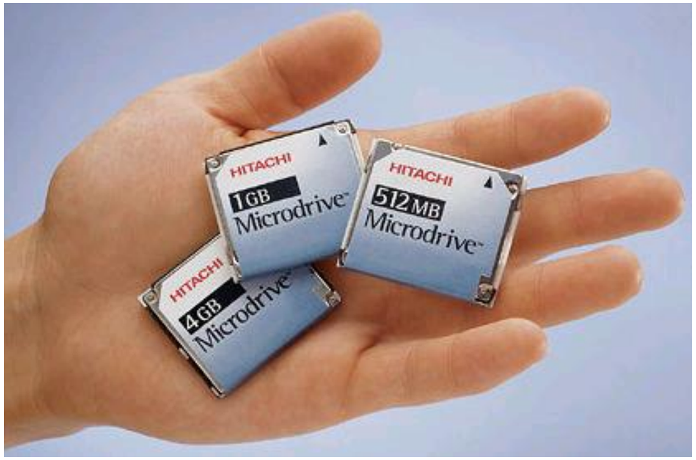
Western Digital Drive
http://www.storagereview.com/guide/

al



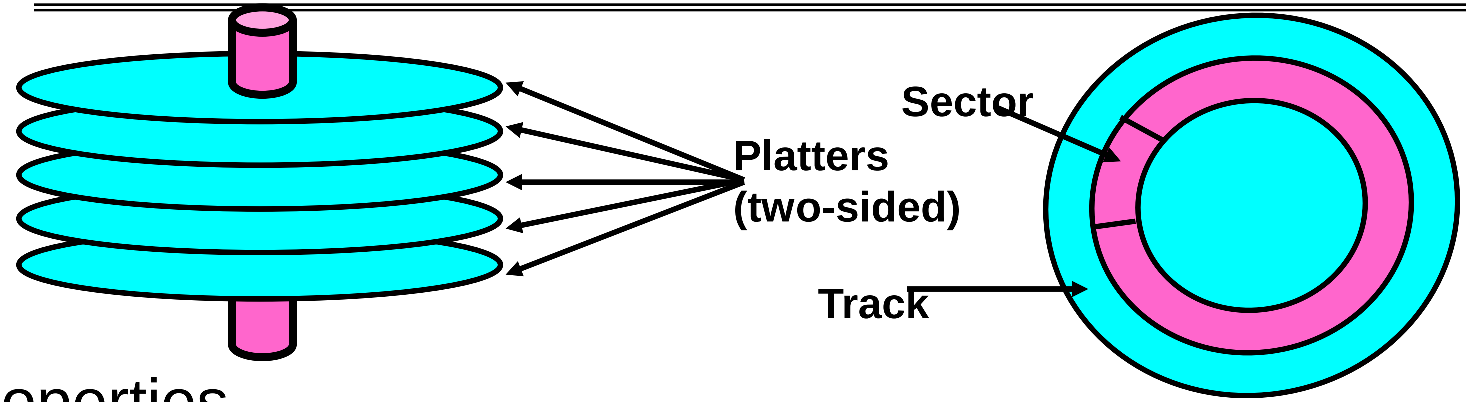
Read/Write Head Side View

IBM Personal Computer/AT (1986)
- 30 MB hard disk - $500
- 30-40ms seek time
- 0.7-1 MB/s (est.)



IBM/Hitachi Microdrive

Properties of a Magnetic Hard Disk

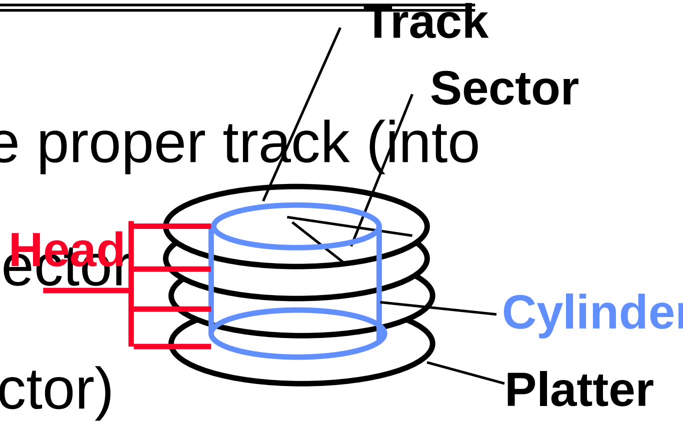


- Properties
  - Independently addressable element: sector
    - » OS always transfers groups of sectors together—"blocks"
  - A disk can access directly any given block either sequentially or randomly.

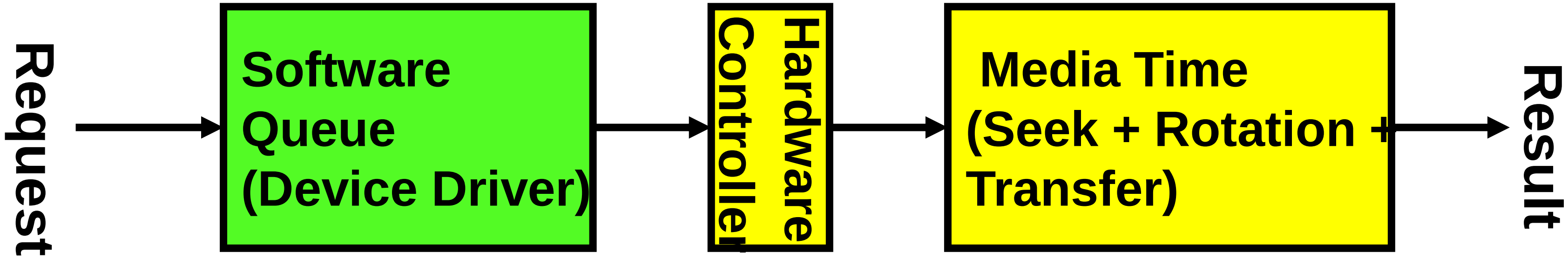
- Zoned bit recording
  - Constant bit density: more bits (sectors) on outer tracks
  - Speed varies with track location

Magnetic Disk Characteristic

- Read/write: three-stage process:
  - **Seek time**: position the head/arm over the proper track (into proper cylinder)
  - **Rotational latency**: wait for the desired sector to rotate under the read/write head
  - **Transfer time**: transfer a block of bits (sector) under the read-write head

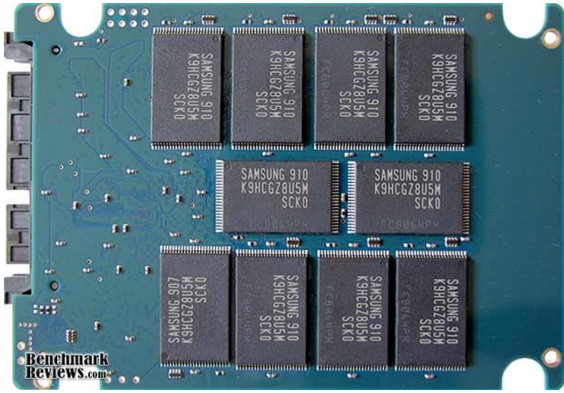


- Disk Latency = Queuing Time + Controller time +
  Seek Time + Rotation Time + Transfer Time



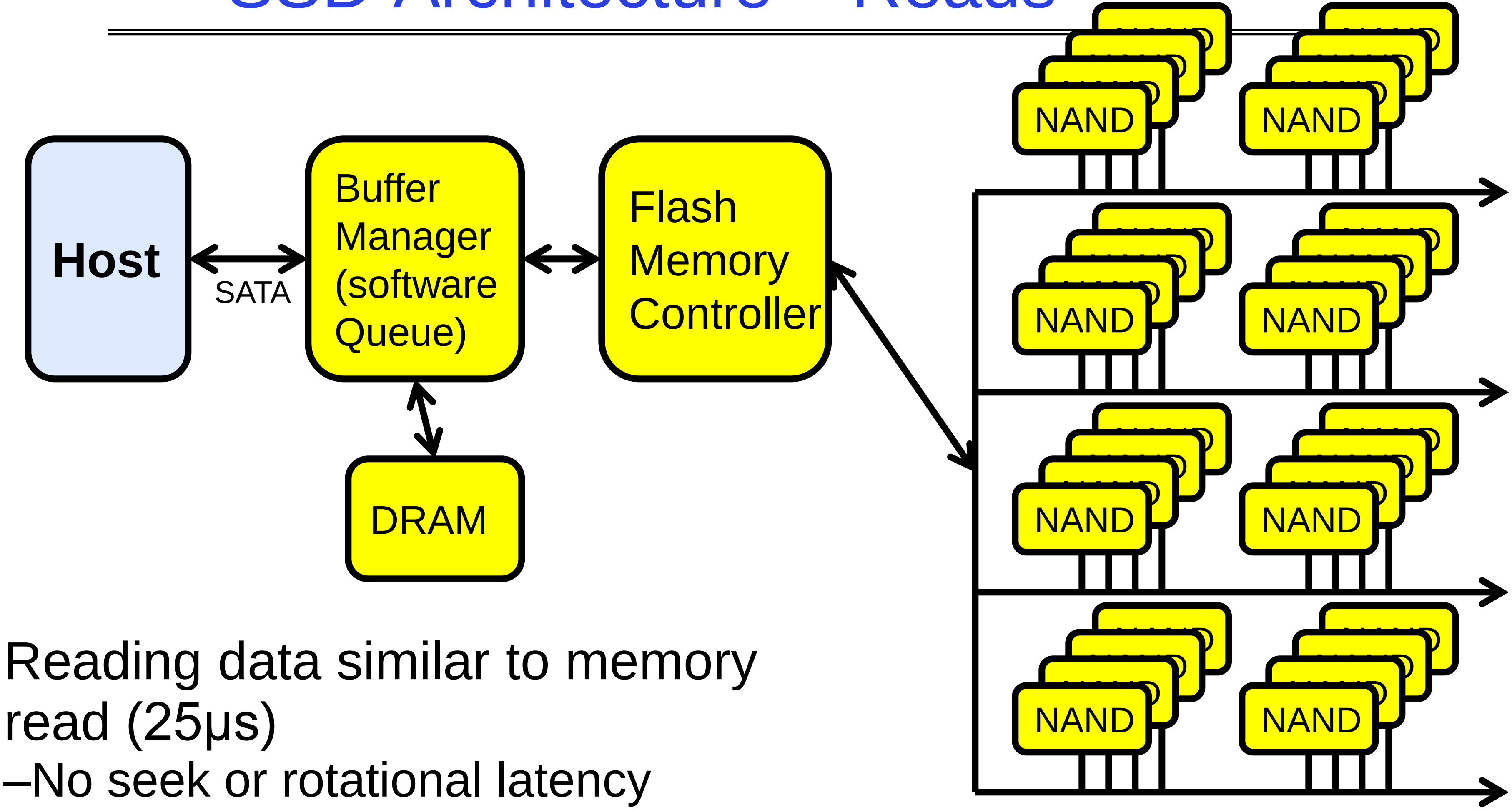
- Highest Bandwidth:
  - Transfer large group of blocks sequentially from one track

Solid State Disks (SSDs)



- 1995 – Replace rotating magnetic media with non-volatile memory (battery backed DRAM)
- 2009 – Use NAND Multi-Level Cell (2-bit/cell) flash memory
  - Sector (4 KB page) addressable, but stores 4-64 “pages” per memory block
- No moving parts (no rotate/seek motors)
  - Eliminates seek and rotational delay (0.1-0.2ms access time)
  - Very low power and lightweight

SSD Architecture – Reads



Reading data similar to memory read (25µs)

–No seek or rotational latency

–Transfer time: transfer a 4KB page

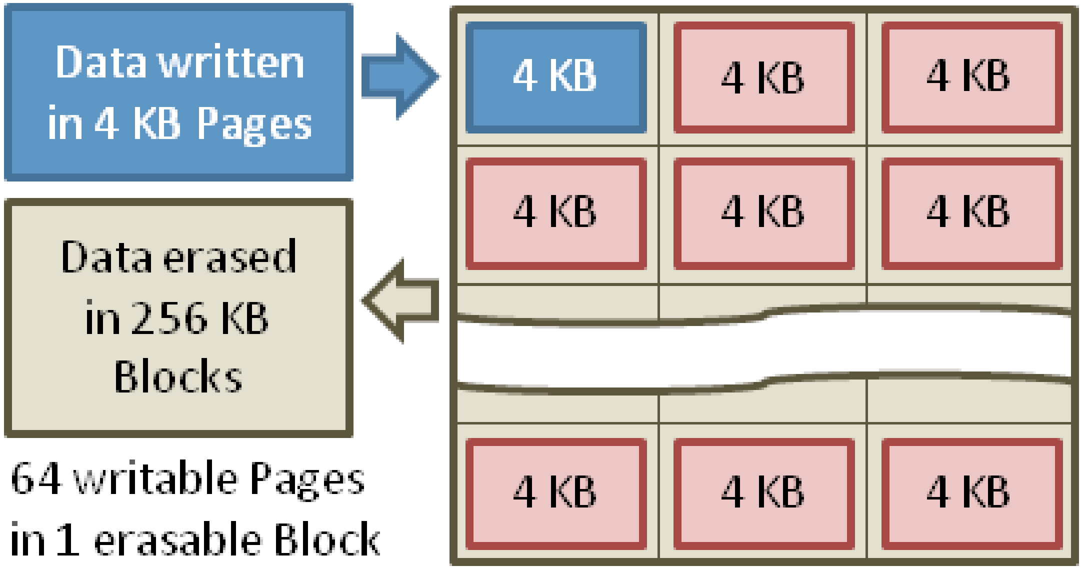
» Limited by controller and disk interface (SATA: 300-600MB/s)

–Latency = Queuing Time + Controller time + Xfer Time

–Highest Bandwidth: Sequential OR Random reads

SSD Architecture – Writes (I)

- Writing data is complex! (~200μs – 1.7ms)
  - Can only write empty pages in a block
  - Erasing a block takes ~1.5ms
  - Controller maintains pool of empty blocks by combining used pages (read, erase, write).

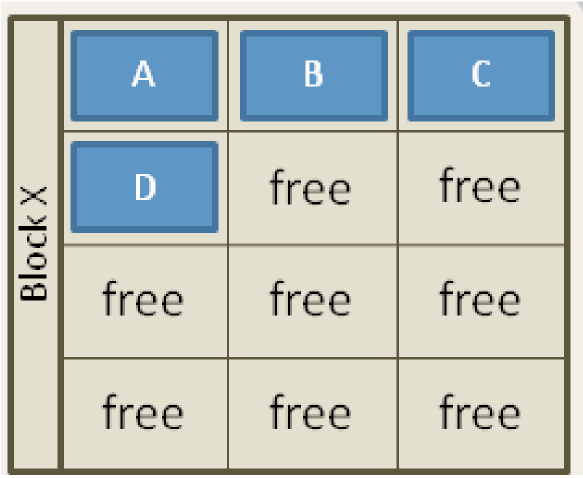

Data written in 4 KB Pages
Data erased in 256 KB Blocks
64 writable Pages in 1 erasable Block
4 KB
4 KB
4 KB
4 KB
4 KB
4 KB
4 KB
4 KB
4 KB

Typical NAND Flash Pages and Blocks

https://en.wikipedia.org/wiki/Solid-state_drive

SSD Architecture – Writes (II)

- Write A, B, C, D

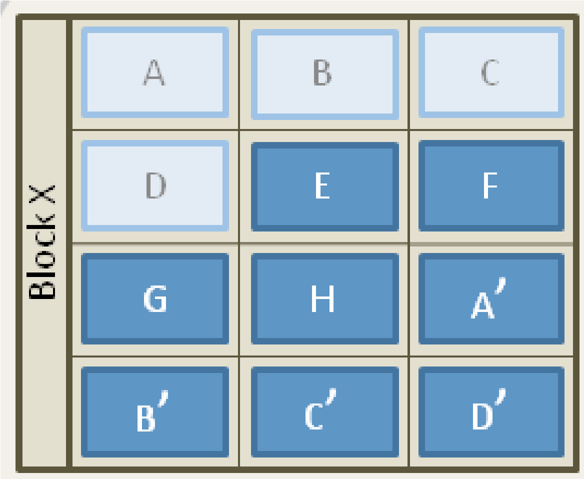


A	B	C
D	free	free
free	free	free
free	free	free

https://en.wikipedia.org/wiki/Solid-state_drive

SSD Architecture – Writes (II)

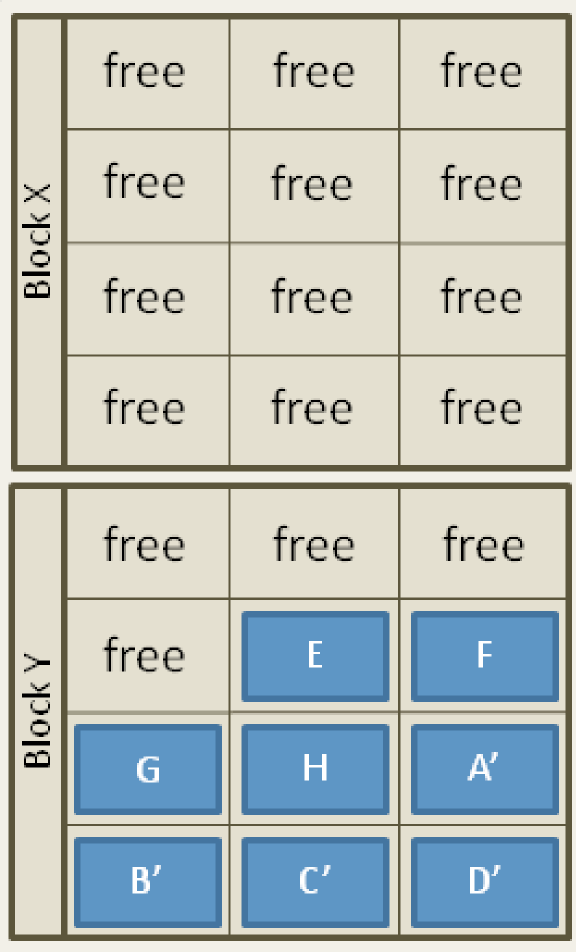
- Write A, B, C, D
- Write E, F, G, H and
  A’, B’, C’, D’
  - Record A, B, C, D as obsolete



https://en.wikipedia.org/wiki/Solid-state_drive

SSD Architecture – Writes (II)

- Write A, B, C, D
- Write E, F, G, H and A’, B’, C’, D’
  - Record A, B, C, D as obsolete
- Controller *garbage collects* obsolete pages by copying valid pages to new block
- Typical steady state behavior when SSD is almost full
  - One erase every 64 or 128 writes



Block X

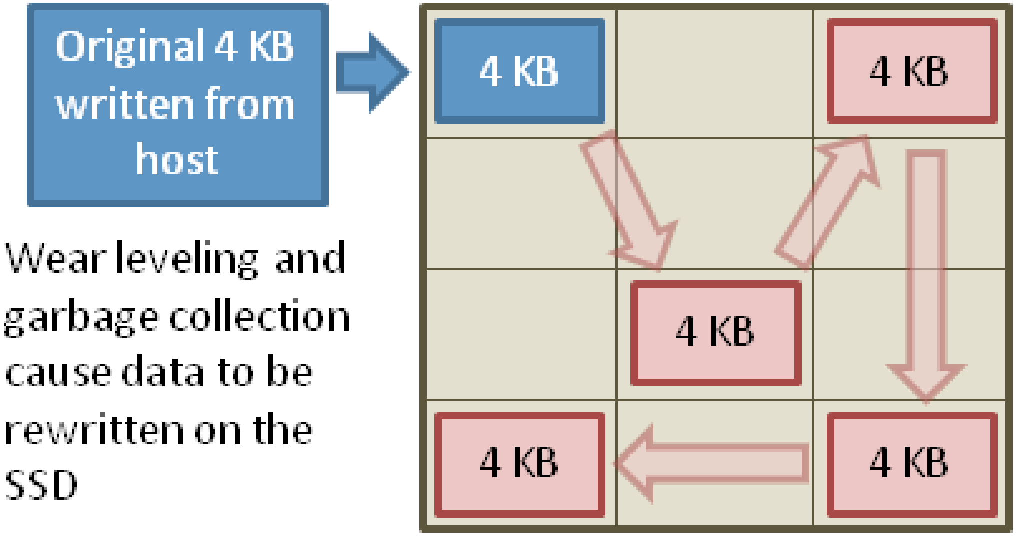
free	free	free
free	free	free
free	free	free
free	free	free

Block Y

free	free	free
free	E	F
G	H	A’
B’	C’	D’

SSD Architecture – Writes (III)

- Write and erase cycles require “high” voltage
  - Damages memory cells, limits SSD lifespan
  - Controller uses error correction.



Wear leveling and garbage collection cause data to be rewritten on the SSD

- Result is very workload dependent performance
  - Latency = Queuing Time + Controller time (Find Free Page) + Xfer Time
  - Highest BW: Seq. OR Random writes (limited by empty pages)

Rule of thumb: writes 10x more expensive than reads, and erases 10x more expensive than writes

SSD Summary

- Pros (vs. hard disk drives):
  - Low latency, high throughput (eliminate seek/rotational delay)
  - No moving parts:
    - » Very light weight, low power (0.3x disk), silent, very shock insensitive
  - Read at memory speeds (limited by controller and I/O bus)

- Cons
  - Smaller storage (0.5x disk), expensive (7~10x disk)
    - » Hybrid alternative: combine small SSD with large HDD
  - Cannot update a single page in a block.
  - Asymmetric block write performance: read pg/erase/write pg
    - » Controller garbage collection (GC) algorithms have major effect on performance