

# CS61C: Fall 2021

## Lecture 20

### *Input/Output*

# The 3 “R”s of Computing:



*input*

*data-structures  
algorithms*

*output*

*Ron Ayres*

*Input/output gives computers a way to interact with the world (and us!)*

# Outline

- I/O Devices and Interconnects
- Direct Memory Access
- Disks
- Networking
- And in Conclusion ...

# Outline

- I/O Devices and Interconnects
  - Direct Memory Access
  - Disks
  - Networking
  - And in Conclusion ...

# Some Input/output Devices (peripherals):

<b><i>name</i></b>	<b><i>type</i></b>	<b><i>approx. max data-rate</i></b>	<b><i>where</i></b>	<b><i>use</i></b>
<b>magnetic disk drive</b>	in/out	200 MB/s	servers, desktops	file system, virtual
<b>solid-state drive (SSD)</b>	in/out	550 MB/s	servers, desktops, laptops, handheld	file system, virtual
<b>display</b>	out	100 MB/s	desktops, laptops, handheld	HCI (images, video)
<b>keyboard</b>	in	10 B/s	desktops, laptops	HCI (typing)
<b>speakers / headphones</b>	out	200 KB/s	desktops, laptops, handheld	HCI (audio)
<b>microphone</b>	in	200 KB/s	desktops, laptops, handheld,	Audio in
<b>mouse</b>	in	100 B/s	laptops	HCI (point/click)
<b>trackpad</b>	in	100 B/s	laptops	HCI (point/click)
<b>inertial measurement unit</b>	in	1 KB/s	handheld, embedded	motion tracking
<b>video camera</b>	in	100 MB/s	desktops, laptops, handheld, embed	video input
<b>trackball</b>	in	100 B/s	desktops	HCI (point/click)
<b>haptic joystick</b>	in/out	100 B/s	desktops	HCI (point/click)
<b>printer</b>	out	2 KB/s	desktops, laptops	graphics/text
<b>touch screen</b>	in/out	100 B/s, 100MB/s	laptops, handheld	display/point/click
<b>cellular radio</b>	in/out	1 GB/s	handheld	wireless access
<b>compute accelerator</b>	in/out	AFAP	servers, desktops, laptops, handheld	energy efficiency
<b>other computer/server</b>	in/out	AFAP	all	MP/www
<b><i>your idea here!</i></b>				

# Challenges

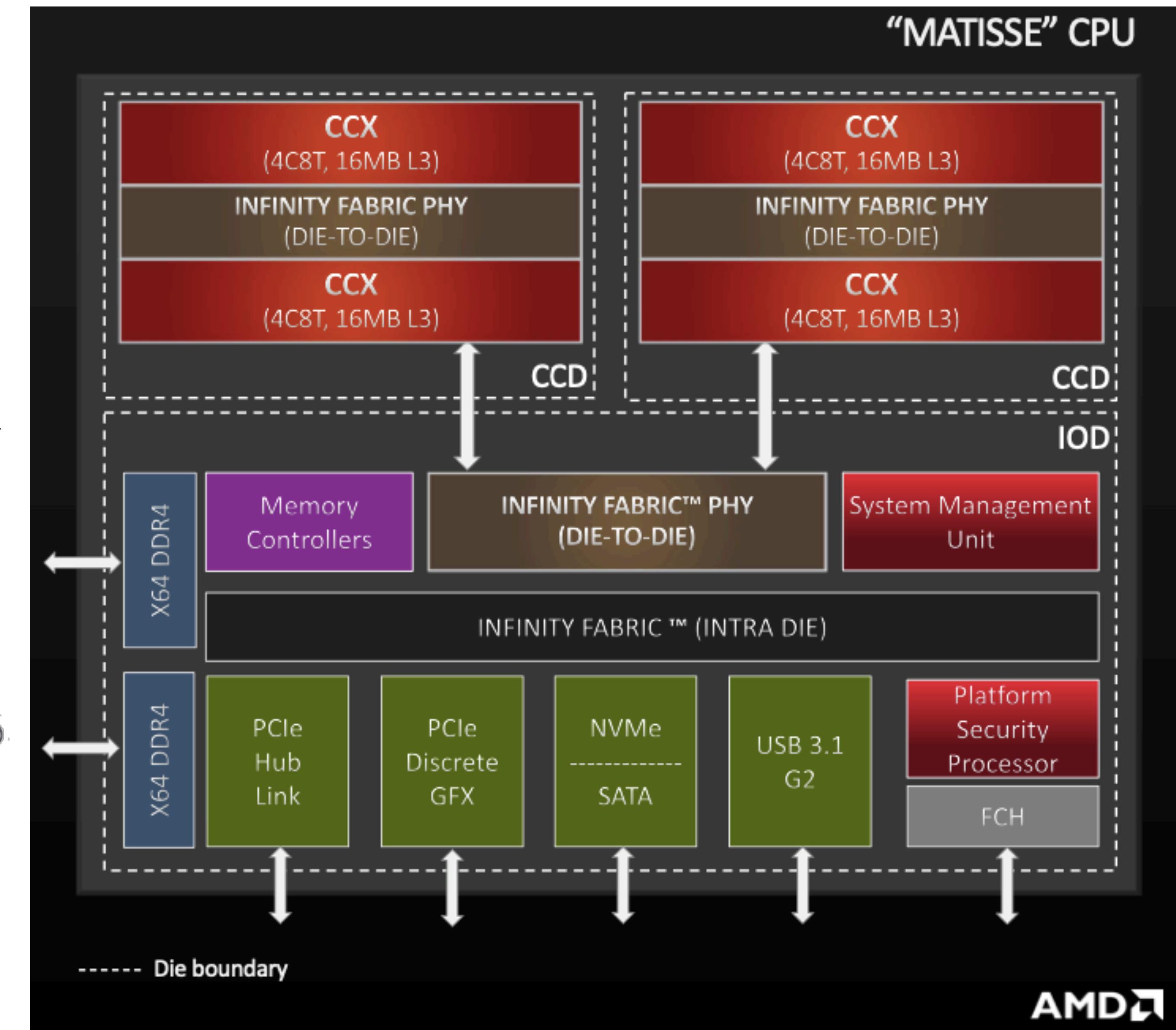
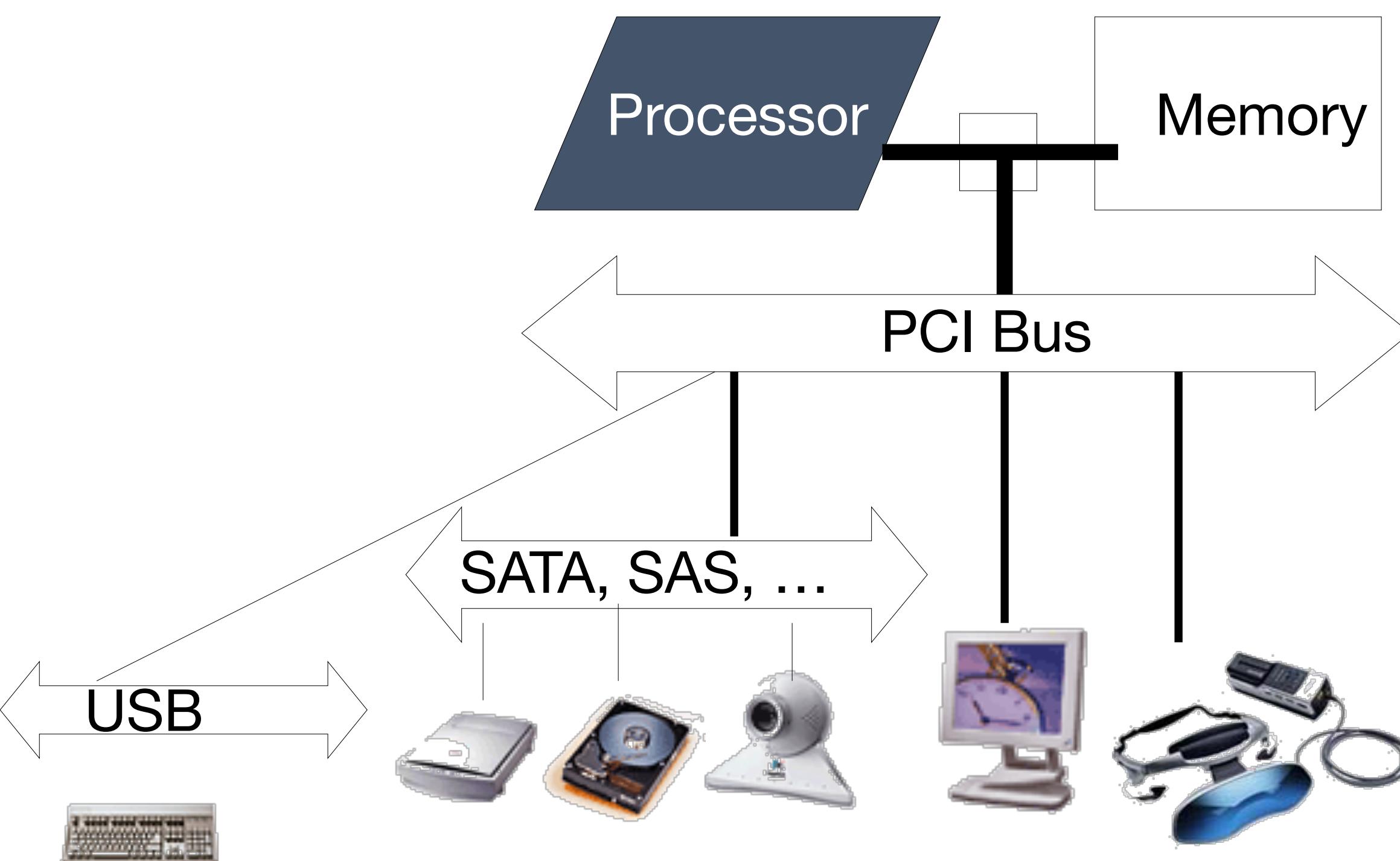
1. *Extremely wide range of data rates*
2. *Differing physical interfaces*
3. *Device specific semantics (initialization, control, data-movement)*
4. *Unknown future devices*

# Challenges

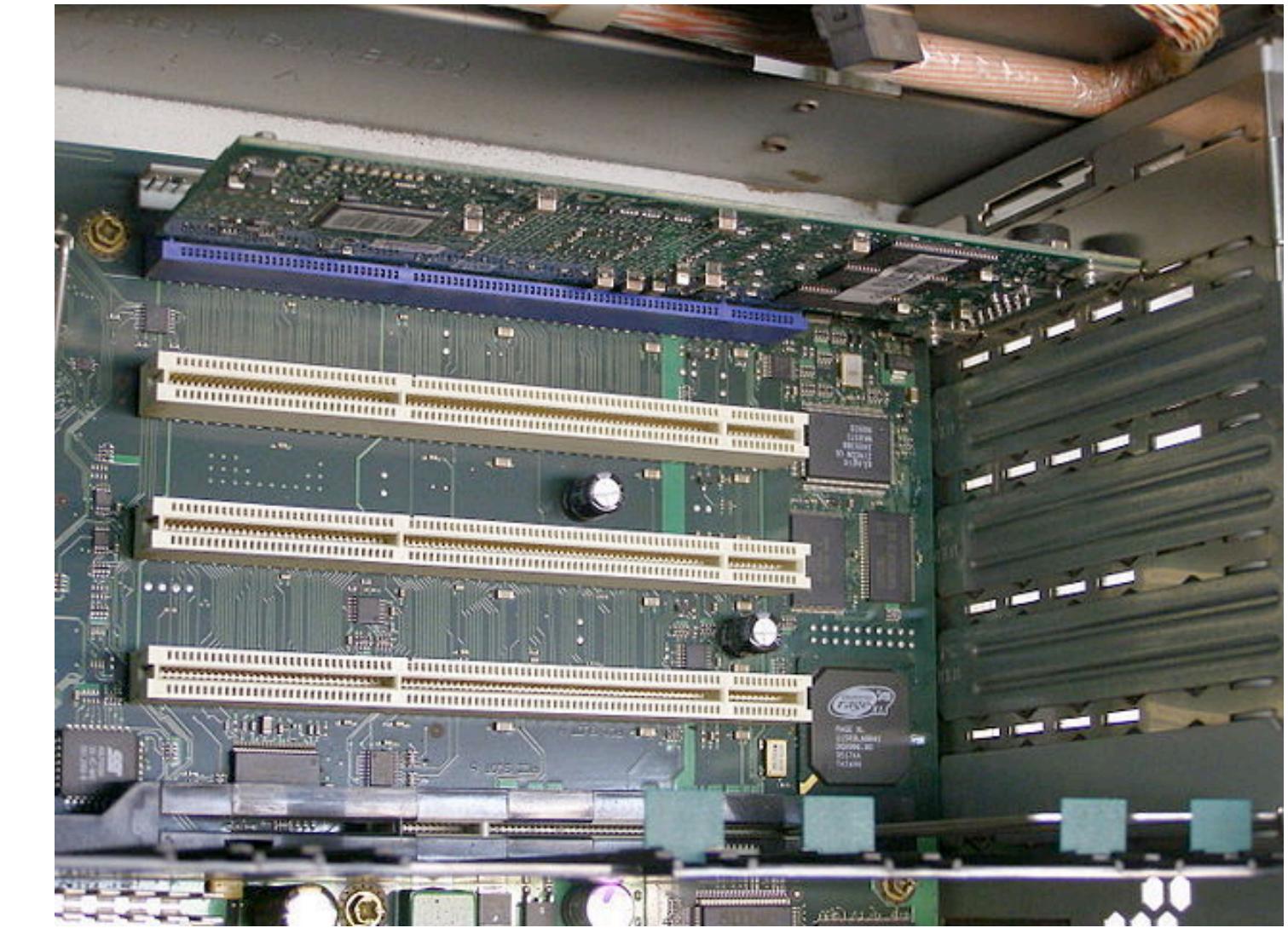
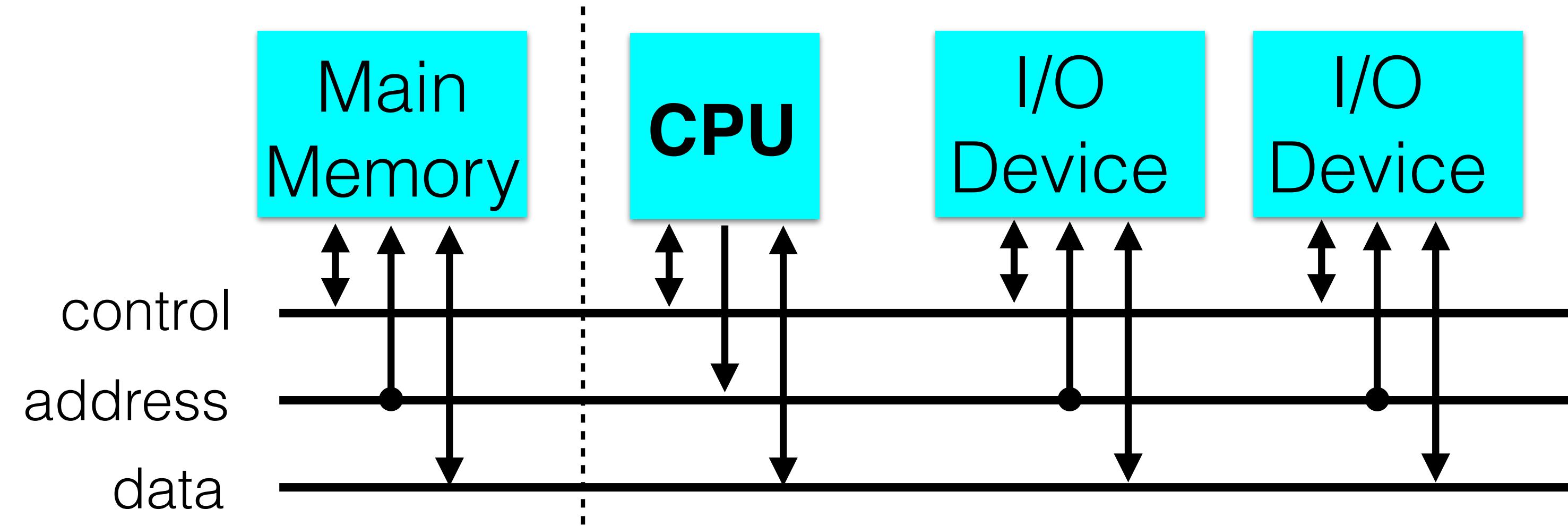
## Solutions:

- Present CPU with unified view of device world (*abstraction!*)
  - Standard Interconnect for connecting to CPU: PCI, USB, SATA, Ethernet
  - Standard programming model
    - devices controlled and data transferred through memory addresses (device registers and buffers) - memory mapped I/O
    - OS device drivers: devices abstracted as files
    - OS input/output library (device control and data transfer)
    - Language specific I/O library
- Controller/interface chips:
  - convert device physical interface (electrical signal, timing) to standard interconnect
  - contain registers/buffers for memory mapped abstraction
  - In some cases interface chips present compressed view of device (GPU, postscript printers)
- Support for multiple interaction and data transfer models
  - Interrupts, Polling, DMA

# Computer system architecture:



# Traditional “Bus” based interconnect architecture

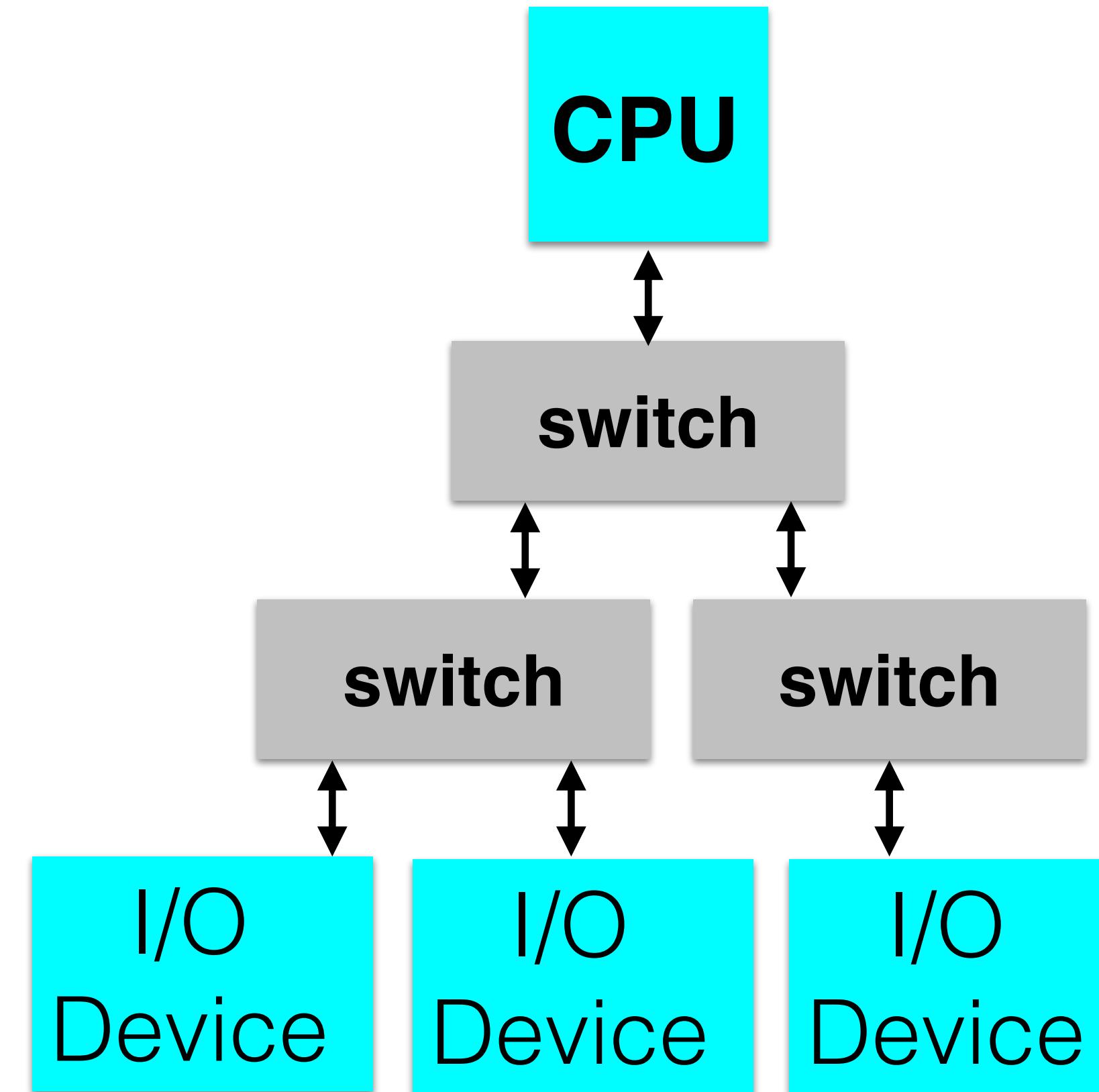


- Simple/inexpensive
- Arbitration needed for shared wires
- Capacitance/inductance limits performance / scalability
- Single transfer at a time

“relatively” modern example: Peripheral Component Interconnect (PCI)

*Fun facts:* 1) The original Ethernet had a shared bus architecture using a single conductor!  
2) Wireless is often same architecture (ex: WiFi)

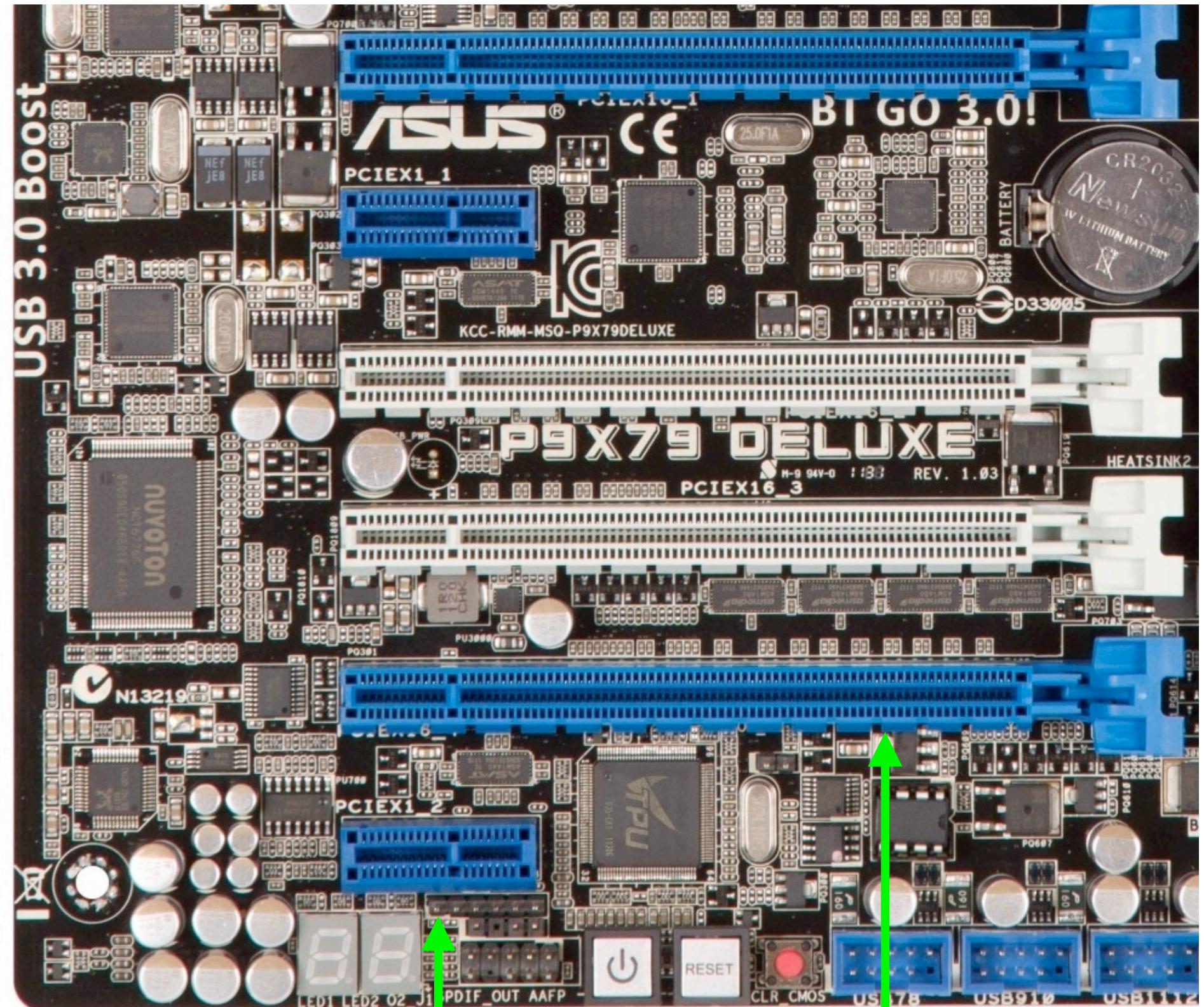
# Switched Interconnect Architecture



- Control/Address/Data messages “routed” from source to destination, usually in packets
- Flexible topology (ring, all-to-all, ...)
- Versus Shared Bus:
  - Mitigates “one at a time” constraint
  - Much higher performance
  - Decentralized arbitration/control
  - Limited electrical loading - high frequency
- Used by most high-performance peripheral and CPU to CPU (cache/cache) interconnect
- Ethernet, PCIe, proprietary intra/inter processor interconnect

# Peripheral Component Interconnect (PCI) / PCIe details

- PCI (1992/1993) - *shared bus*
  - 32-bit / 33MHz – 133MB/sec
  - 64-bit / 66MHz – 533MB/sec
- PCI-X (1999)
  - Up to 1066MB/sec with 64-bit / 133MHz
- PCI Express – aka PCIe (2002): *switched point-to-point*
  - Version 5: 32 GT/s, yielding 63 GB/s in each direction in a 16-lane configuration
  - Version 6: 64 GT/s, yielding 126 GB/s in each direction in a 16-lane configuration
- Directly maps devices to CPU address space (64 bits)
- Supports interrupts and direct memory access (DMA)
- Dual Simplex point-to-point serial connection for each link
  - Version 4: 2 GB/s per lane
- Capacity scaling from x1 to x16 (1-16 lanes)
- Packet based transaction protocol
- Link level ACK/NAK
- End to end CRC error detection

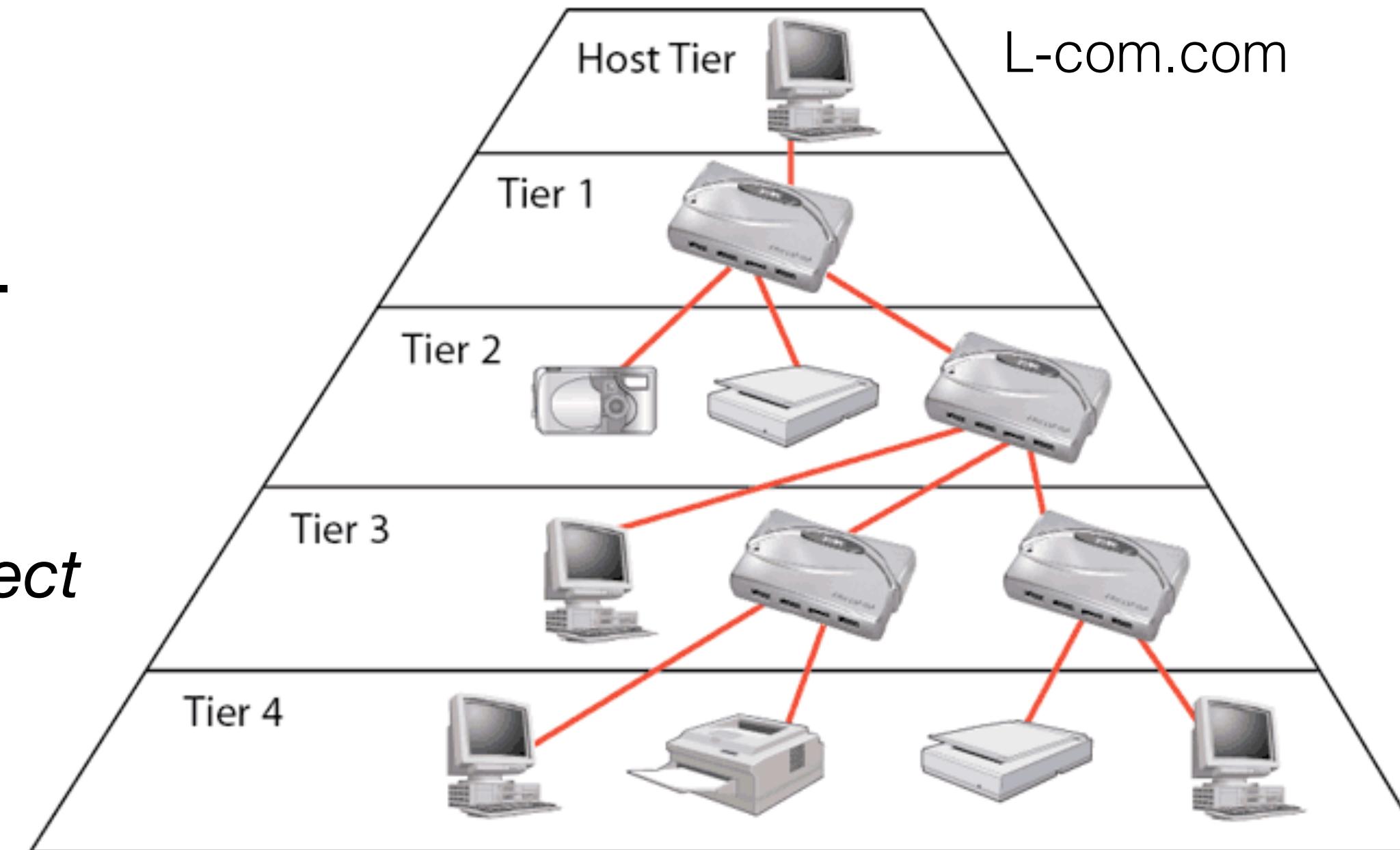


PCIe Connectors

# Universal Serial Bus (USB) Details



- Specifications for cables, connectors and protocols for connection, communication and power supply between computers, peripherals and other computers. 14 different connectors! USB-C the most recent.
- Bit-serial single lane with (limited) power supplied to peripheral.
- Four generations: USB 1.x, USB 2.0, USB 3.x, and USB4.
- USB1 (1996): 1.5 Mbit/s (Low Speed) and 12 Mbit/s (Full Speed).
- USB3: 400 MB/s (3.2 Gbit/s) achieved throughput.
- USB4: supports 40 Gbit/s throughput.
- Hubs often used, but structured as a bus *not switched interconnect*
- Devices cannot interact with one another except via the host.
- Self-configuring, no need for the user to adjust the device's settings for speed or data format, and input/output addresses.
- Hot-swappable (devices can be exchanged without rebooting the host computer).
- USB cables are limited in length, intended for peripherals on the same table-top, not between rooms or buildings.
- Several different transfer modes (low-latency, streaming real-time data), but no true interrupts.



# Outline

- I/O Devices and Interconnects
- Direct Memory Access
- Disks
- Networking
- And in Conclusion ...

# So What Happens When Data Arrives?

- Input is asynchronous
  - It may occur at any time without coordination with what the OS is doing
- Option 1: Trigger an interrupt
  - Jumps control to the interrupt handler which has to figure out what to do...
- Option 2: Wait for the OS to poll the device
  - Its the OS's job to check whether something is available

# More on Interrupt-Driven I/O

- Highly responsive
  - When data comes in, the interrupt triggers
- Interesting efficiency tradeoff:
  - For low rate it's **very** efficient while still very responsive
    - The computer is doing other things except when data comes
  - For high rates it's inefficient
    - Interrupts are relatively expensive!  
You effectively **have** to flush all cache state, flush the pipeline, flush the TLBs etc. going to/from the OS
- Common design:
  - Interrupts by default
  - Then if high rates needed, poll instead
    - Ex: Multi-gigabit Network interfaces

# Working with real devices

- “Memory mapped I/O”: Device control/data registers mapped to CPU address space
- CPU synchronizes with I/O device:
  - Polling
  - Interrupts
- “Programmed I/O”
  - CPU execs lw/sw instructions for all data movement to/from devices
  - Generally, CPU spends time doing 2 things:
    - Getting data from device to main memory, from main memory to a device
    - Using data to compute

# Working with real devices

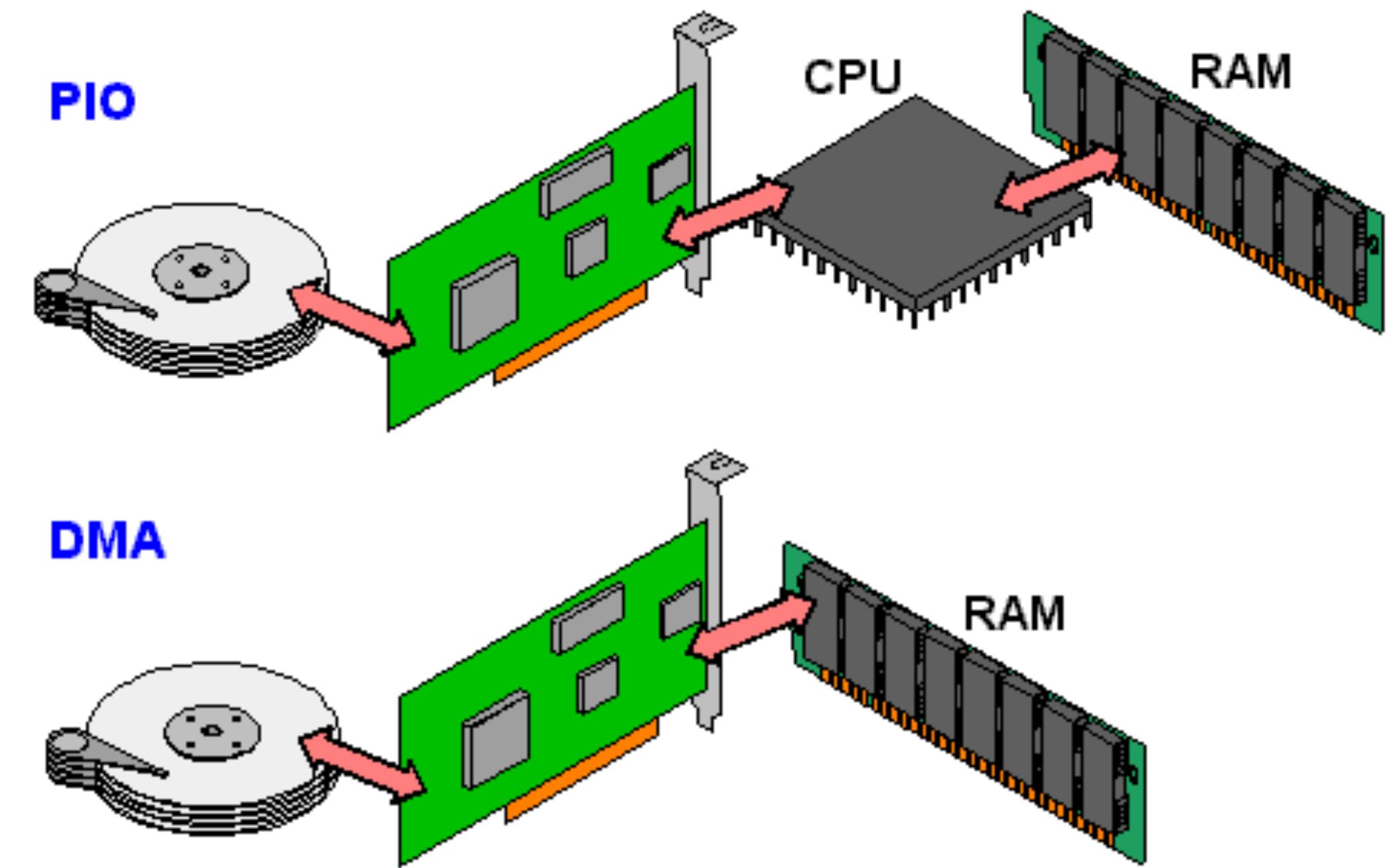
- “Memory mapped I/O”: Device control/data registers mapped to CPU address space
- CPU synchronizes with I/O device:
  - Polling
  - Interrupts
- “Programmed I/O”: DMA
  - CPU execs ~~lw/sw~~ instructions for all data movement to/from devices
  - CPU spends time doing 1 thing:
    - Getting data from device to main memory, from main memory to a device
    - Using data to compute

# What's wrong with Programmed I/O?

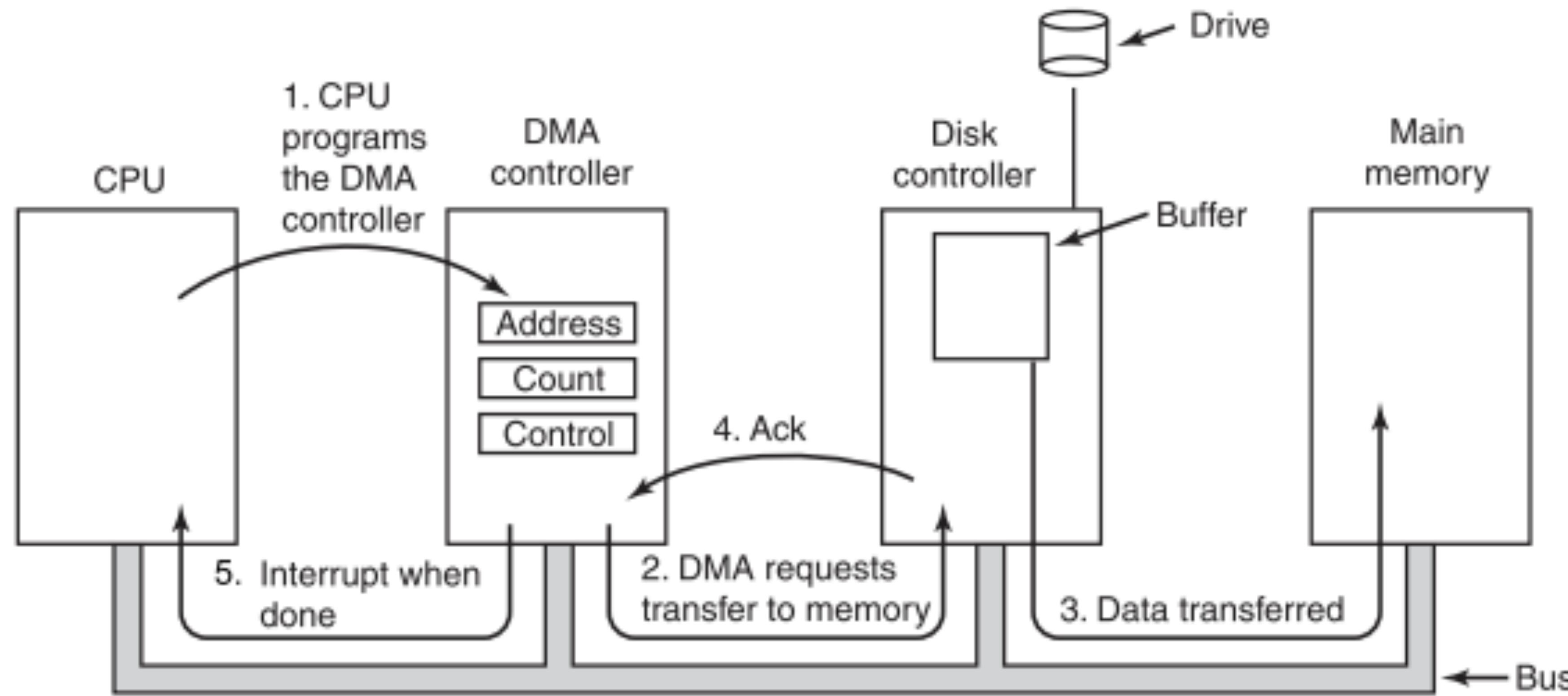
- Not ideal because ...
  - CPU has to execute all transfers, could be doing other work
  - Device speeds don't align well with CPU speeds
  - Energy cost of using beefy general-purpose CPU where simpler hardware would suffice
- Until now CPU has sole control of main memory

# Direct Memory Access (DMA)

- Allows I/O devices to directly read/write main memory
- New Hardware: the DMA Engine
- DMA engine contains CSR registers written by CPU:
  - Memory address to write/read data,
  - # of bytes
  - I/O device #, direction of transfer
  - unit of transfer, amount to transfer per burst



# Operation of a DMA Transfer



**Figure 5-4.** Operation of a DMA transfer.

[From Section 5.1.4 Direct Memory Access in *Modern Operating Systems* by Andrew S. Tanenbaum, Herbert Bos, 2014]

Note: On PCI/PCIe, any I/O device can be a DMA engine (controller, master)

# DMA: Incoming Data

1. Receive interrupt from device
2. CPU takes interrupt, initiates transfer
  - Instructs DMA engine/device to place data @ certain address
3. Device/DMA engine handle the transfer
  - CPU is free to execute other things
4. Upon completion, Device/DMA engine interrupt the CPU again

# DMA: Outgoing Data

1. CPU decides to initiate transfer, confirms that external device is ready
2. CPU initiates transfer
  - Instructs DMA engine/device that data is available @ certain address
3. Device/DMA engine handle the transfer
  - CPU is free to execute other things
4. Device/DMA engine interrupt the CPU again to signal completion

# DMA: Some new problems

- Where in the memory hierarchy do we plug in the DMA engine?  
Two extremes:
  - Between L1 and CPU:
    - Pro: Free coherency
    - Con: Trash the CPU's working set with transferred data
  - Between Last-level cache and main memory:
    - Pro: Don't mess with caches
    - Con: Need to explicitly manage coherency
- Or just treat like another node in a multiprocessor
  - Cache-coherence is supported by most modern multiprocessors
  - This is what modern computers do: the DMA engine just acts like another processor for the ***cache coherence*** mechanisms we will discuss later

# Outline

- I/O Devices and Interconnects
- Direct Memory Access
- **Disks**
- Networking
- And in Conclusion ...

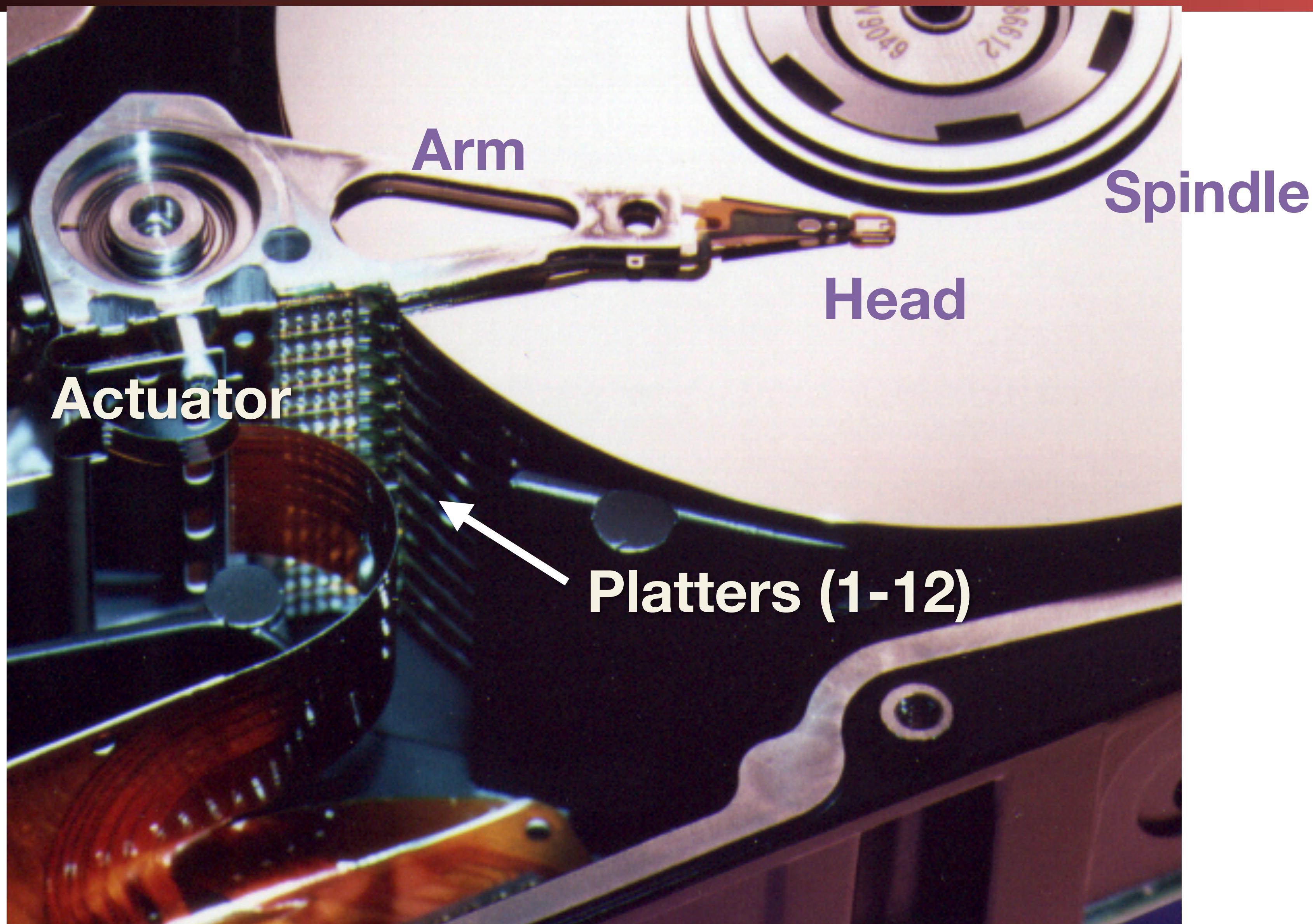
# Magnetic Disk – common I/O device

Computer Science 61C Fall 2021

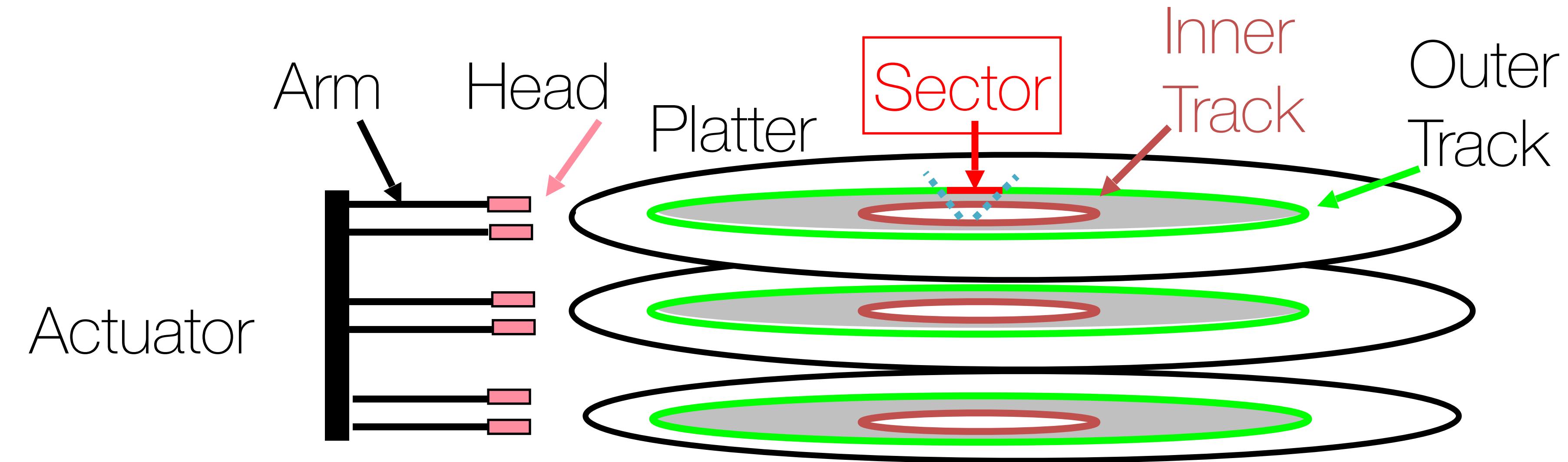
- A kind of computer memory
  - Information stored by magnetizing ferrite material on surface of rotating disk
    - Similar to tape recorder except digital rather than analog data
- A type of *non-volatile* storage
  - Retains its value without applying power to disk.
- Two Types of Magnetic Disk
  - Hard Disk Drives (HDD) – faster, more dense, non-removable.
  - Floppy disks – slower, less dense, removable (now replaced by USB “flash drive”).
- Purpose in computer systems (Hard Drive):
  - Working file system + long-term backup for files
  - Secondary “backing store” for main-memory. Large, inexpensive, slow level in the memory hierarchy (virtual memory)



# Photo of Disk Head, Arm, Actuator



# Disk Device Terminology

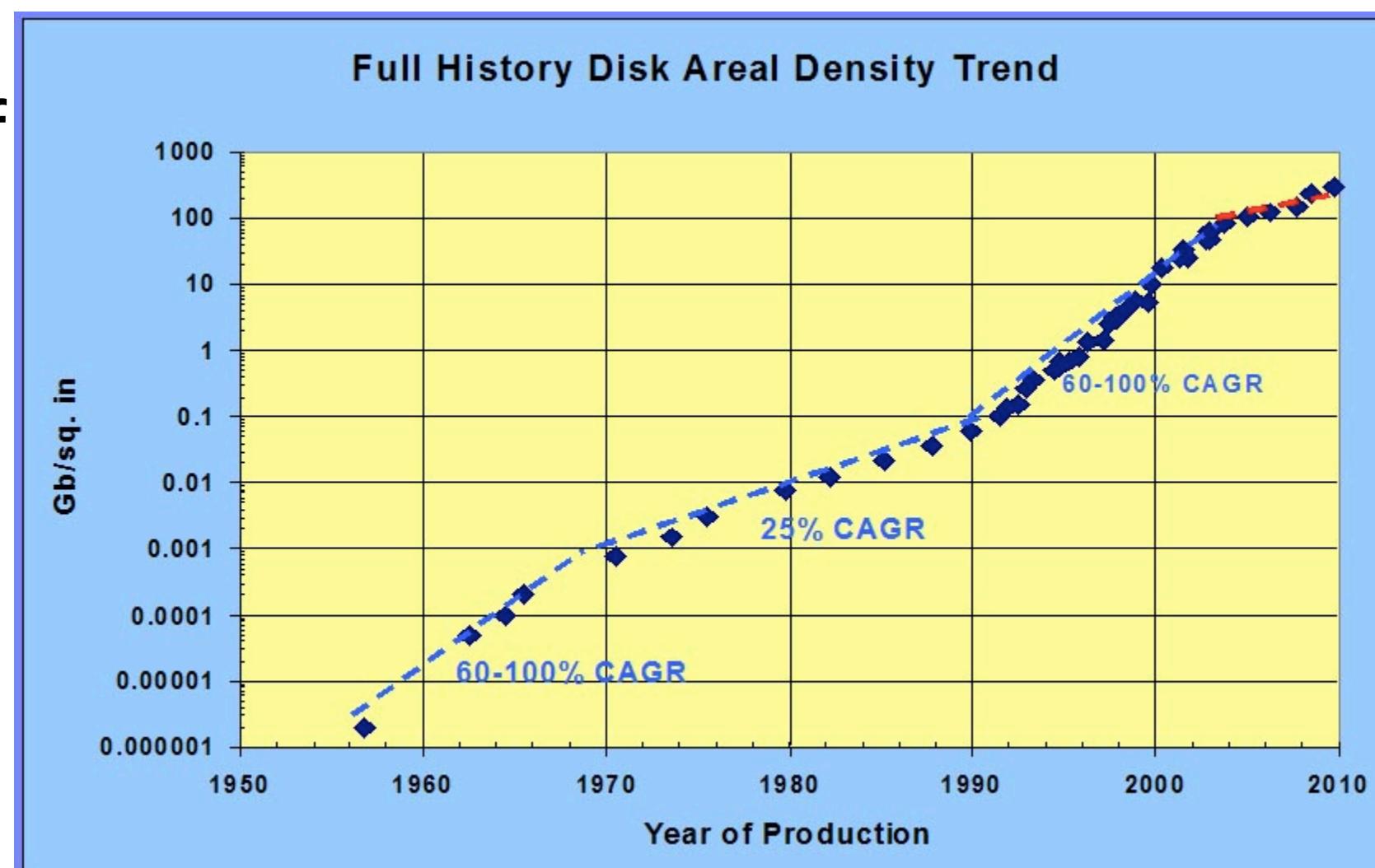
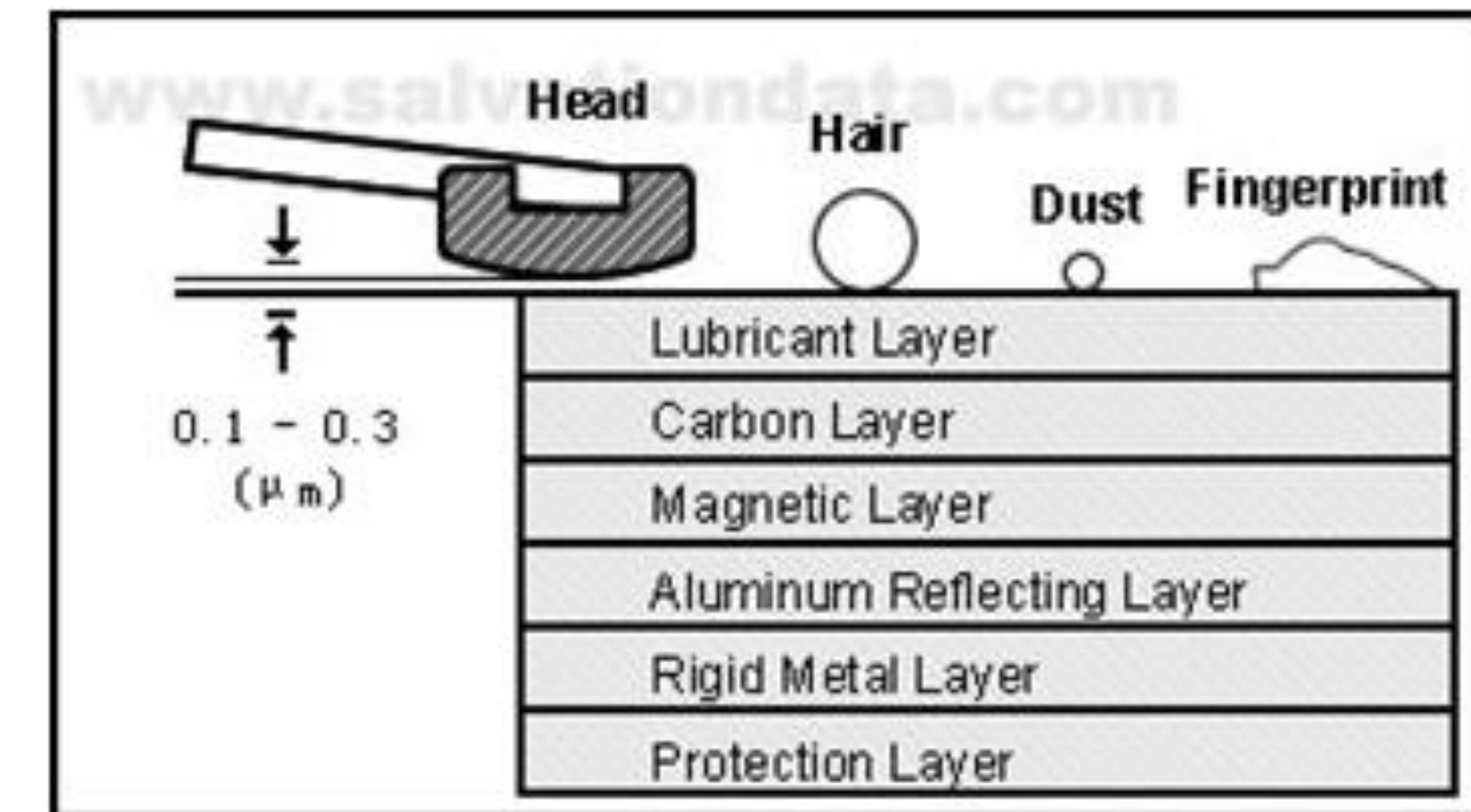


- Several platters, with information recorded magnetically on both surfaces (usually)
- Bits recorded in tracks, which in turn divided into sectors (e.g., 512 Bytes)
- Actuator moves head (end of arm) over track (“seek”), wait for sector rotate under head, then read or write

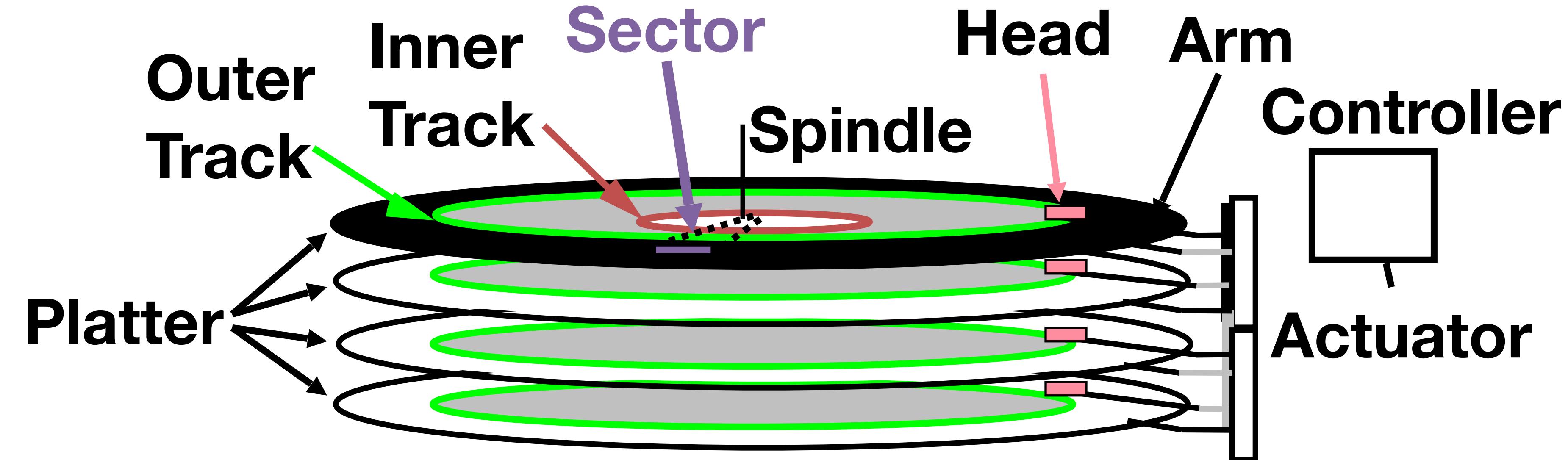
[Video of hard disk in action](#)

# Hard Drives are Sealed

- The closer the head to the disk, the smaller the “spot size” and thus the denser the recording.
  - Measured in Gbit/in<sup>2</sup>
  - ~900 Gbit/in<sup>2</sup> is state of the art
  - Started out at 2 Kbit/in<sup>2</sup>
  - ~450,000,000x improvement in ~60 years
- Disks are sealed to keep the dust out.
  - Heads are designed to “fly” at around 3-20nm above the surface of the disk.
  - 99.999% of the head/arm weight is supported by the air bearing force (air cushion) developed between the disk and the head
- Some drives are even sealed with Helium
  - Lower drag than air



# Disk Device Performance (1/2)



- **Disk Access Time = Seek Time + Rotation Time + Transfer Time + Controller Overhead**
  - Seek Time = time to position the head assembly at the proper cylinder
  - Rotation Time = time for the disk to rotate to the point where the first sectors of the block to access reach the head
  - Transfer Time = time taken by the sectors of the block and any gaps between them to rotate past the head

# Disk Device Performance (2/2)

- Average values to plug into the formula:
- Rotation Time: Average distance of sector from head?
  - 1/2 time of a rotation
    - 7200 Revolutions Per Minute  $\Rightarrow$  120 Rev/sec
    - 1 revolution =  $1/120$  sec  $\Rightarrow$  8.33 milliseconds
    - 1/2 rotation (revolution)  $\Rightarrow$  4.17 ms
- Seek time: Average no. tracks to move arm?
  - Number of tracks/3 (see CS186 for the math)
  - Then, seek time = number of tracks moved  $\times$  time to move across one track

# Disk Performance Analysis

- We have the following disk:
  - 15000 tracks, 1 ms to cross 1000 tracks
  - 15000 RPM = 4 ms per rotation
  - Want to copy 1 MB, transfer rate of 1000 MB/s
  - 1 ms controller processing time
- What is the access time?

Seek = # tracks/3 \* time =  $15000/3 * 1\text{ms}/1000 \text{ cylinders} = 5\text{ms}$

Rotation = time for  $\frac{1}{2}$  rotation =  $4 \text{ ms} / 2 = 2 \text{ ms}$

Transfer = Size / transfer rate =  $1 \text{ MB} / (1000 \text{ MB/s}) = 1 \text{ ms}$

Controller = 1 ms

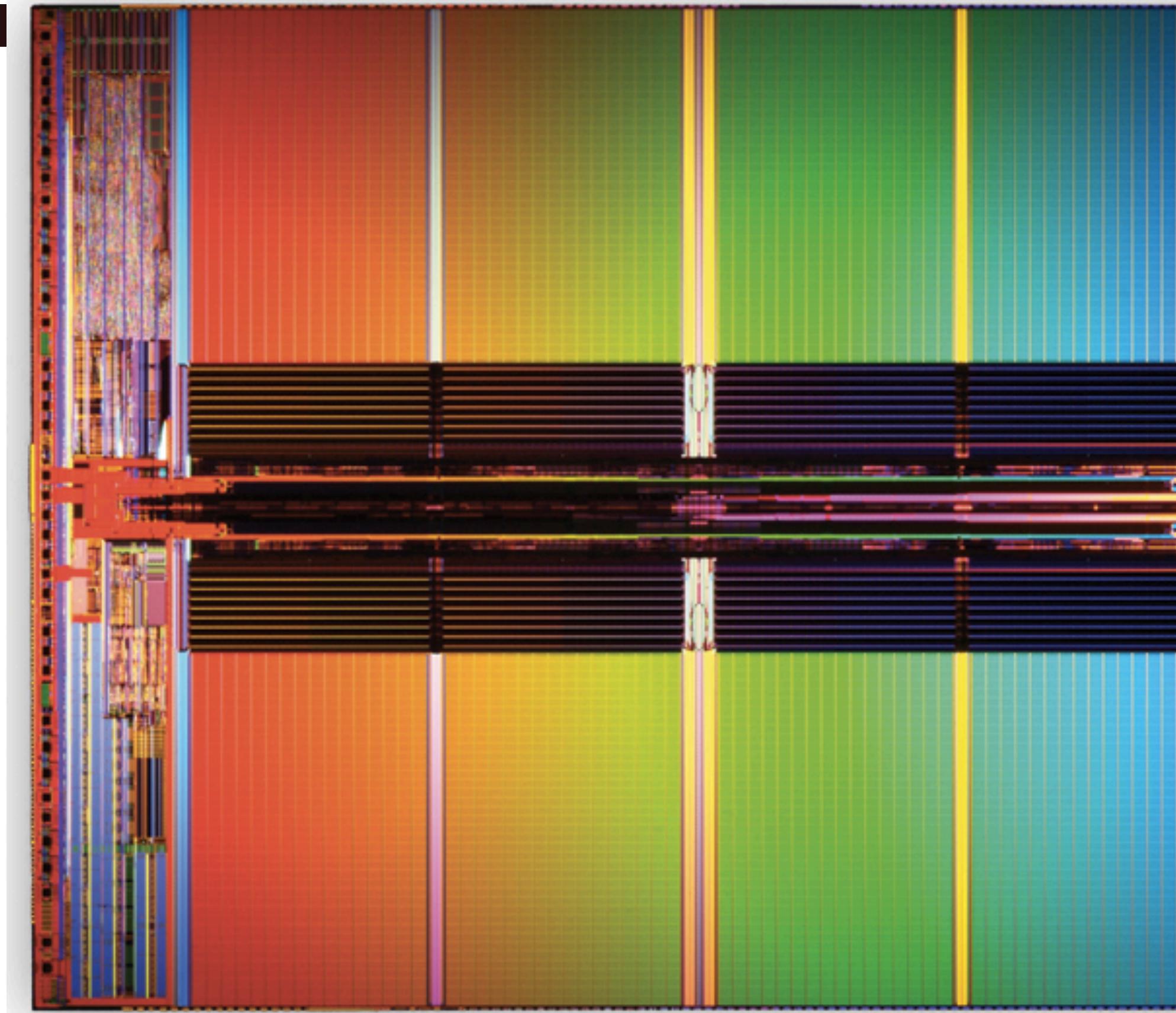
Total =  $5 + 2 + 1 + 1 = 9 \text{ ms}$

# But wait!

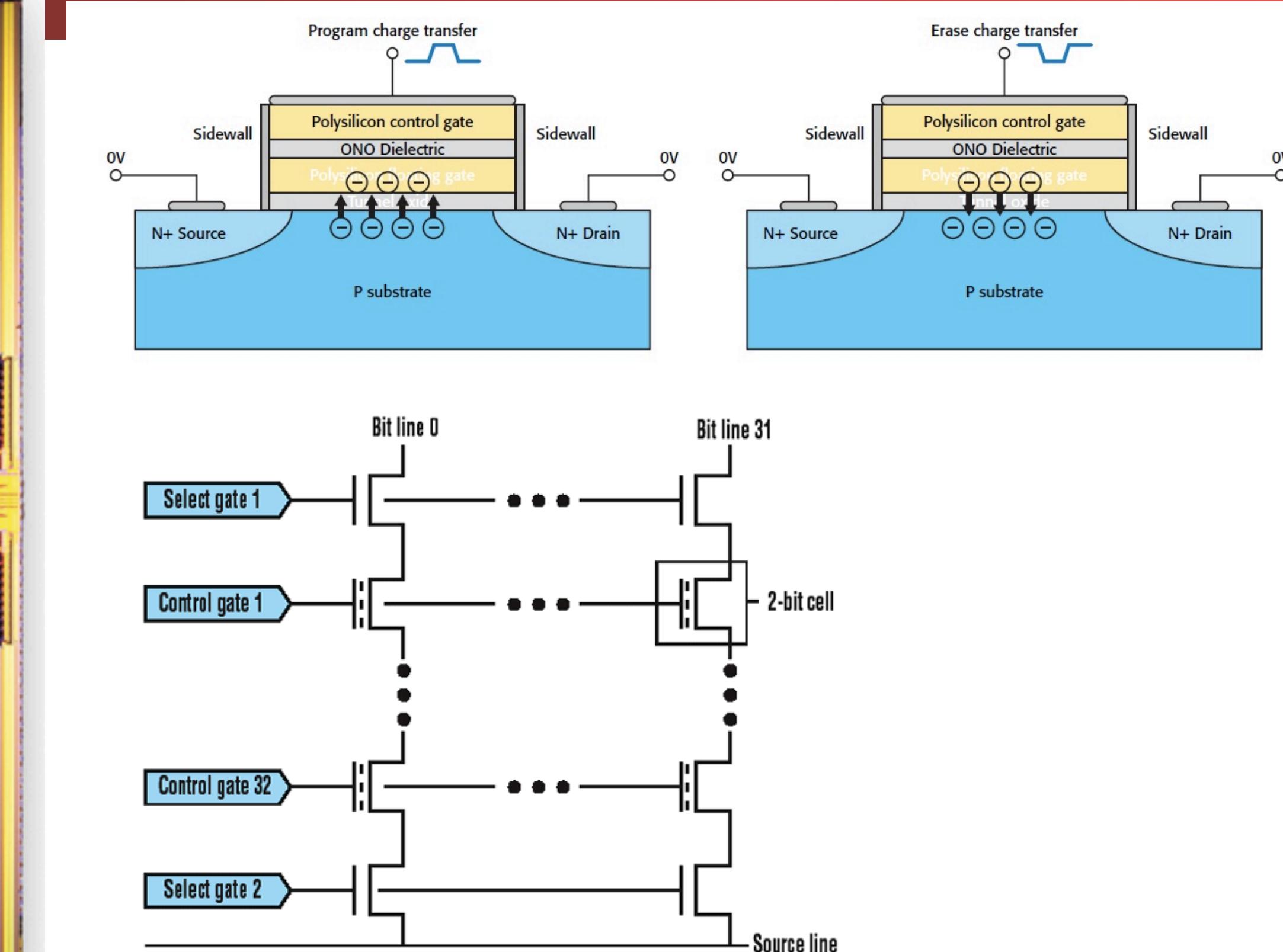
- Performance estimates are different in practice:
- Many disks have on-disk caches, which are completely hidden from the outside world
- Previous formula completely replaced with on-disk cache access time

# Flash Memory / SSD Technology

Computer Science 61C Fall 2021



2. Micron's triple-level cell (TLC) flash memory stores 3 bits of data in each transistor.



In the basic functional block used in multilevel NAND flash memories, 32 rows of bit lines and 32 control-gate lines form a building block that's repeated many times to form the memory array. The select gate lines are used with the control gate lines to control access to the array.

- NMOS transistor with an additional conductor between gate and source/drain which “traps” electrons. The presence/absence is a 1 or 0
- Memory cells can withstand a limited number of program-erase cycles. Controllers use a technique called *wear leveling* to distribute writes as evenly as possible across all the flash blocks in the SSD.

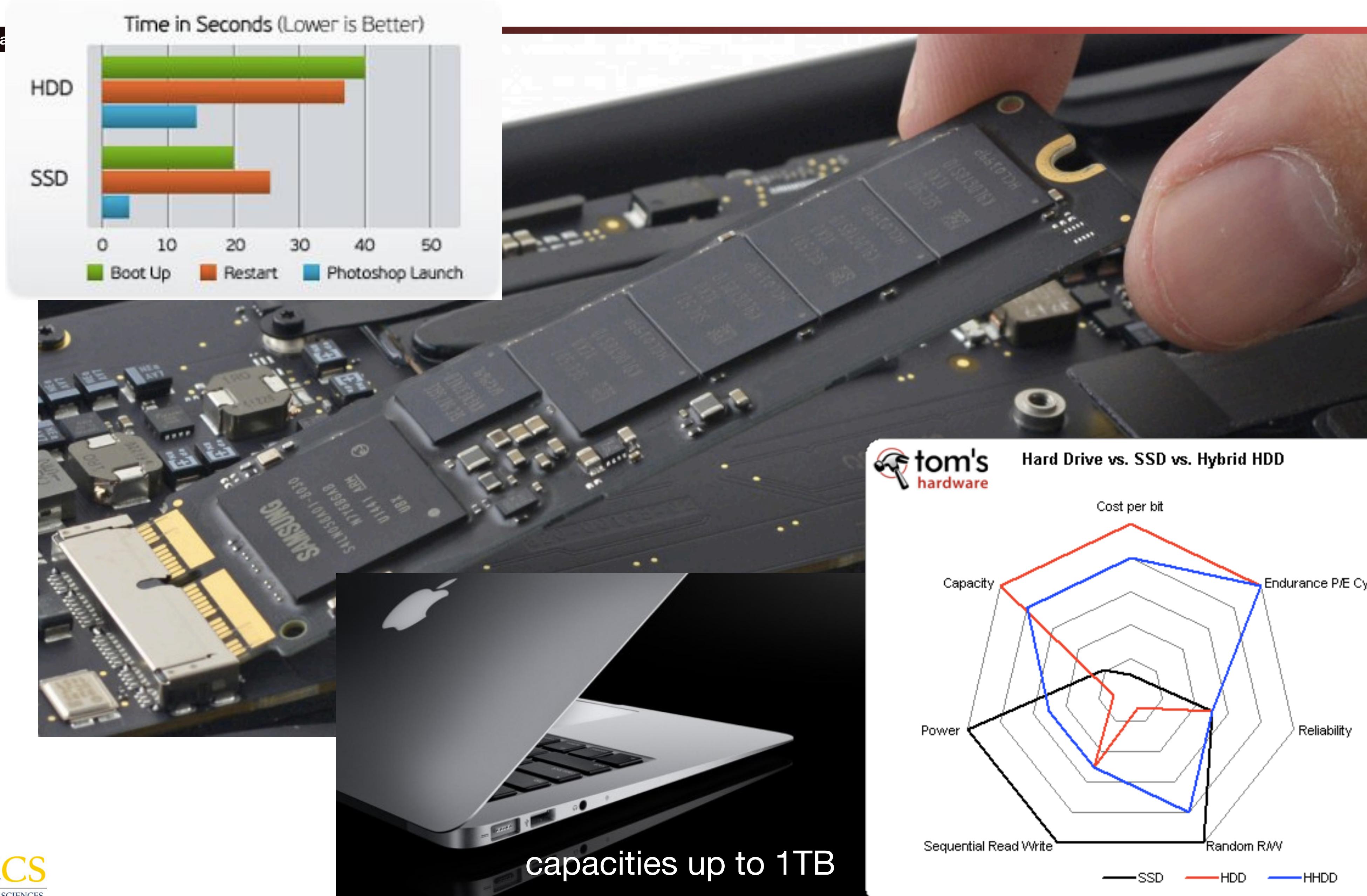
# Flash Memory Key to Success of Smart Phones



# Flash Memory in Laptops – Solid State Drive (SSD)

Computer Science 61C Fa

Wawrynek and Weaver



# Flash and Latency...

- Flash bandwidth is similar to spinning disk
  - And spinning disk is still a better storage/\$ and storage/cm<sup>3</sup>
- But Flash's big advantage: ***no seek time!***
  - No additional latency for random access vs sequential access of a block
- This is huge:
  - HDD access times are measured in milliseconds, SSD times are measured in microseconds

# Outline

- I/O Devices and Interconnects
- Direct Memory Access
- Disks
- Networking
- And in Conclusion ...

# Networks: Talking to the Outside World

- Originally sharing I/O devices between computers
  - E.g., printers
- Then communicating between computers
  - E.g., file transfer protocol
- Then communicating between people
  - E.g., e-mail
- Then communicating between networks of computers
  - E.g., file sharing, www, ...
- Then turning multiple cheap systems into a single computer
  - Warehouse scale computing

# The Internet (1962)

[www.computerhistory.org/internet\\_history](http://www.computerhistory.org/internet_history)

- History
  - 1963: J.C.R. Licklider, while at DoD's ARPA, writes a memo describing desire to connect the computers at various research universities: Stanford, Berkeley, UCLA, ...
  - 1969 : ARPA deploys 4 “nodes” @ UCLA, SRI, Utah, & UCSB
  - 1973 Robert Kahn & Vint Cerf invent TCP, now part of the Internet Protocol Suite
- Internet growth rates
  - Exponential since start



[www.greatachievements.org/?id=3736](http://www.greatachievements.org/?id=3736)

[en.wikipedia.org/wiki/Internet\\_Protocol\\_Suite](http://en.wikipedia.org/wiki/Internet_Protocol_Suite)

# The World Wide Web (1989)

[en.wikipedia.org/wiki/History\\_of\\_the\\_World\\_Wide\\_Web](https://en.wikipedia.org/wiki/History_of_the_World_Wide_Web)

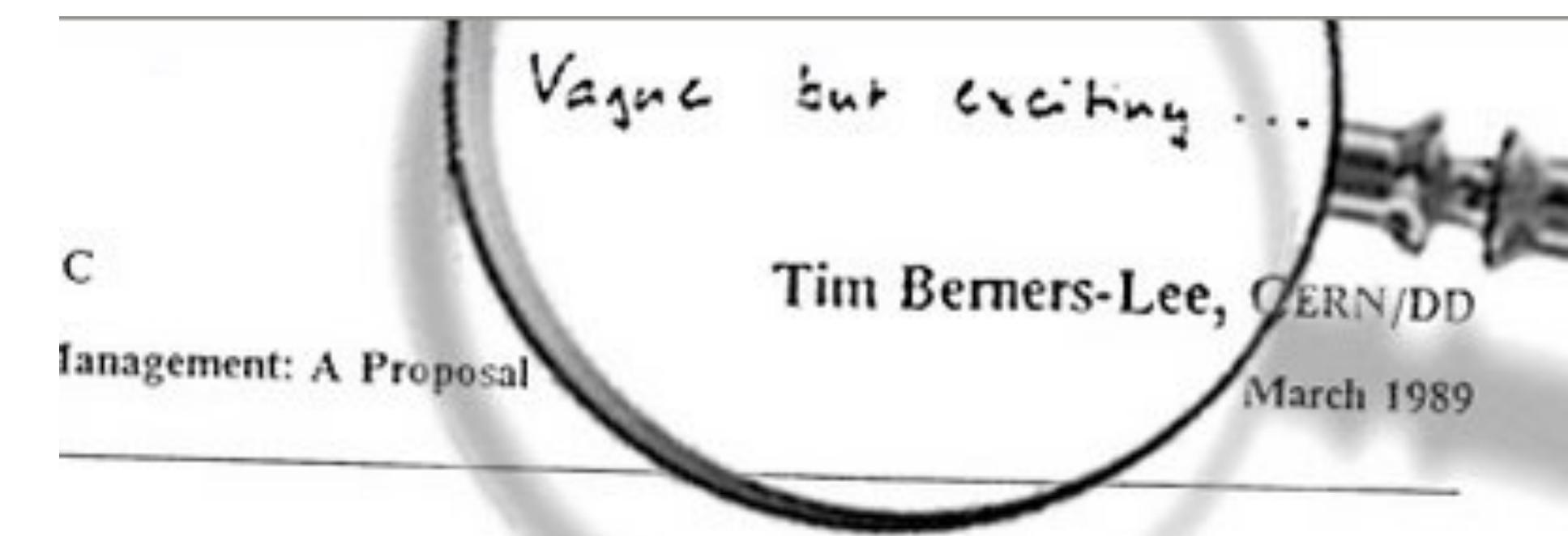
- “System of interlinked hypertext documents on the Internet”
- History
  - 1945: Vannevar Bush describes (a hypothetical electromechanical device and) hypertext system called “memex” in article
  - 1989: Sir Tim Berners-Lee proposed and implemented the first successful communication between a Hypertext Transfer Protocol (HTTP) client and server using the internet.
  - 1993: NCSA Mosaic: A graphical HTTP client
  - ~2000 Dot-com entrepreneurs rushed in, 2001 bubble burst
  - Today : Access anywhere!



Tim Berners-Lee



World's First web server in 1990

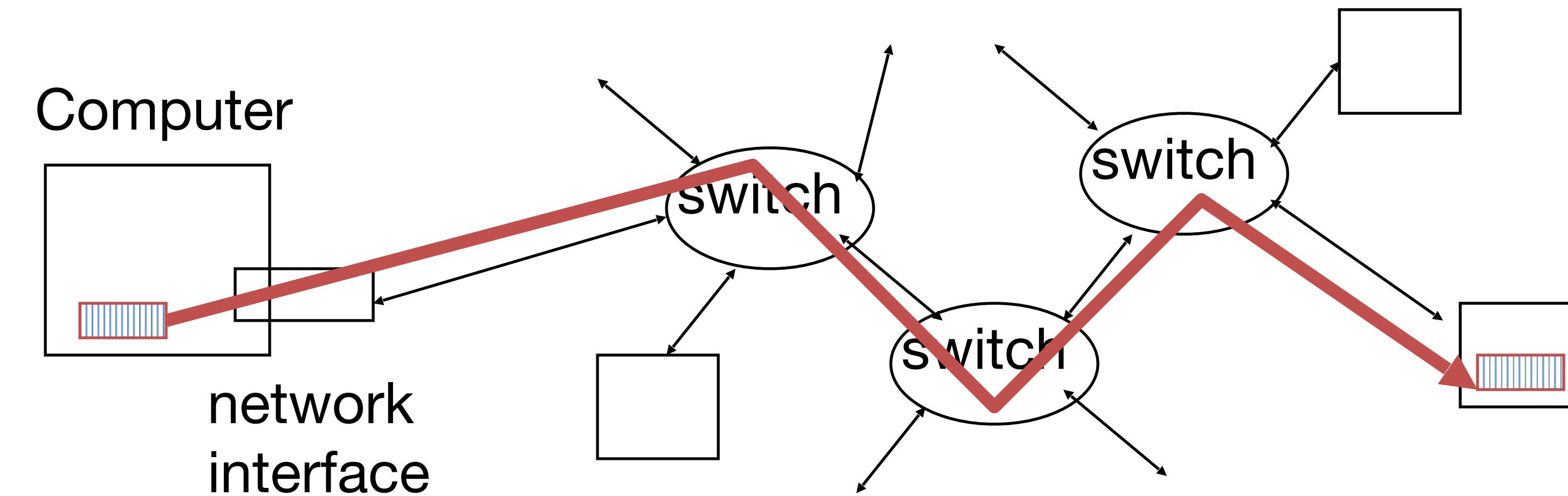


Information Management: A Proposal

Abstract

# What makes networks work?

- **links** connecting **switches and/or routers** to each other and to computers or devices

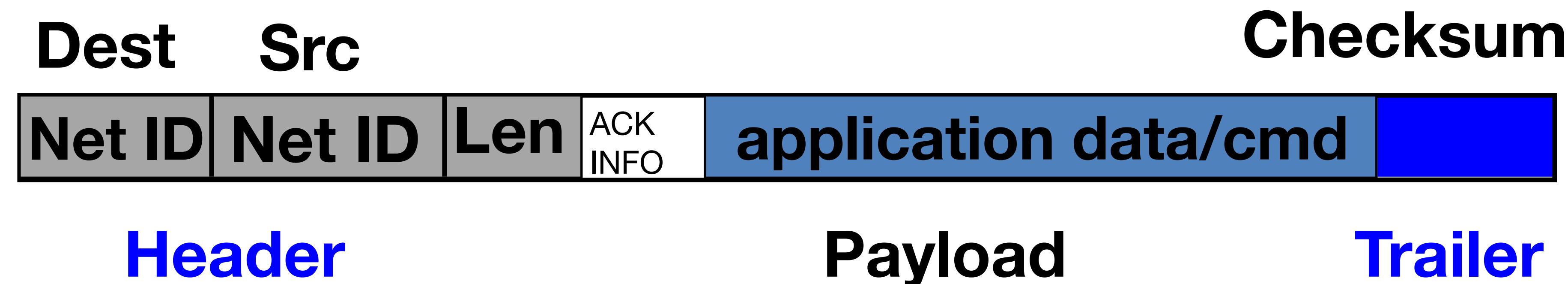


- Ability to name the components and to route packets of information - messages - from a source to a destination
- Layering, redundancy, protocols, and encapsulation as means of abstraction (61C big idea)

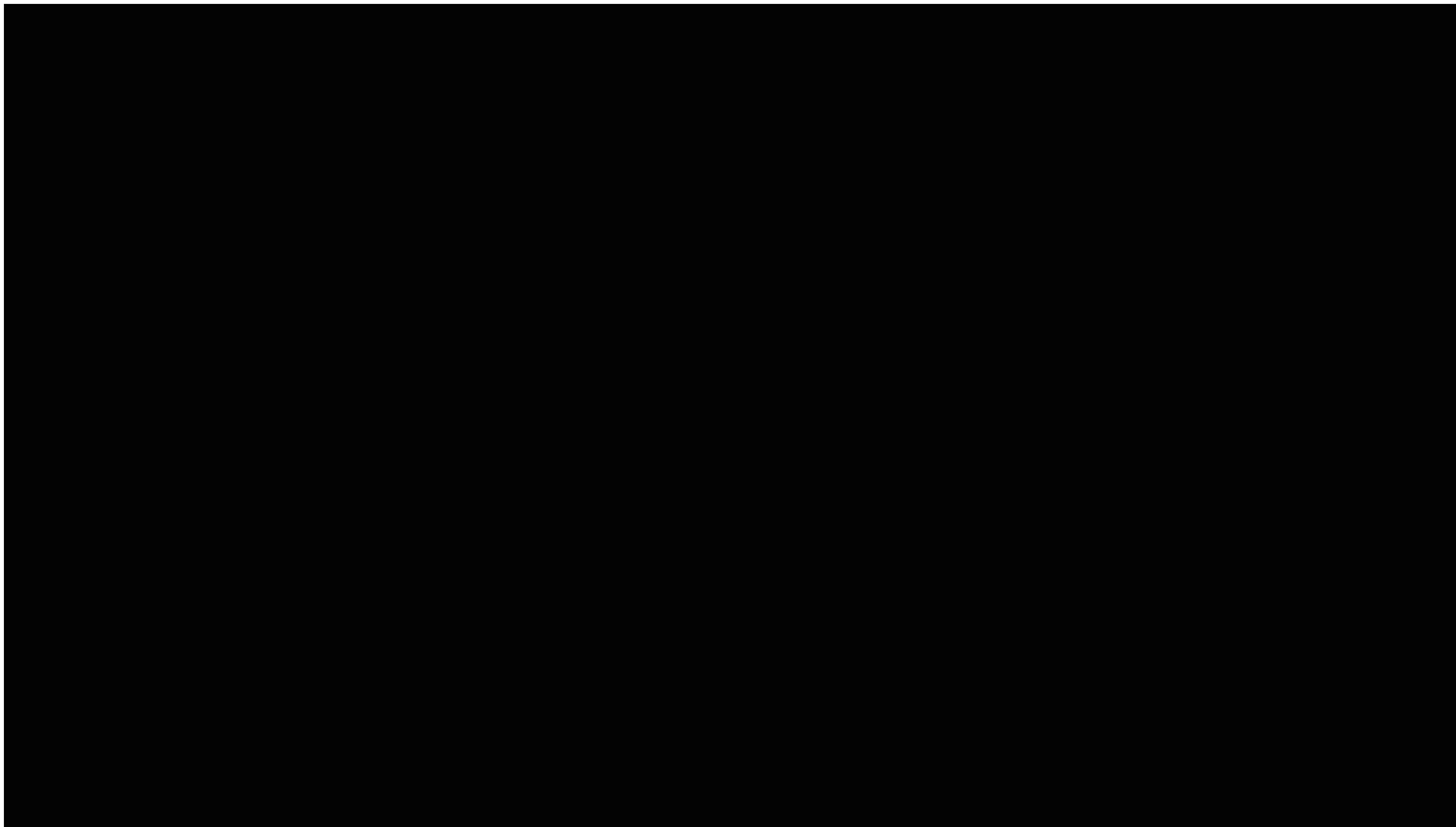


# Software Protocol to Send and Receive

- SW Send steps
  - 1: Application copies data to OS buffer
  - 2: OS calculates checksum
  - 3: OS sends DMA request to network interface HW and says start
- SW Receive steps
  - 3: Network interface copies data from network interface HW to OS buffer, triggers interrupt
  - 2: OS calculates checksum, if OK, send ACK; if not, delete message (sender resends when timer expires)
  - 1: If OK, OS copies data to user address space, & signals application to continue



# Networks are like Ogres (from Shrek)



[https://www.youtube.com/watch?v=\\_bMcXVe8zls](https://www.youtube.com/watch?v=_bMcXVe8zls)

# **Protocols** for Networks of Networks?

What does it take to send packets across the globe?

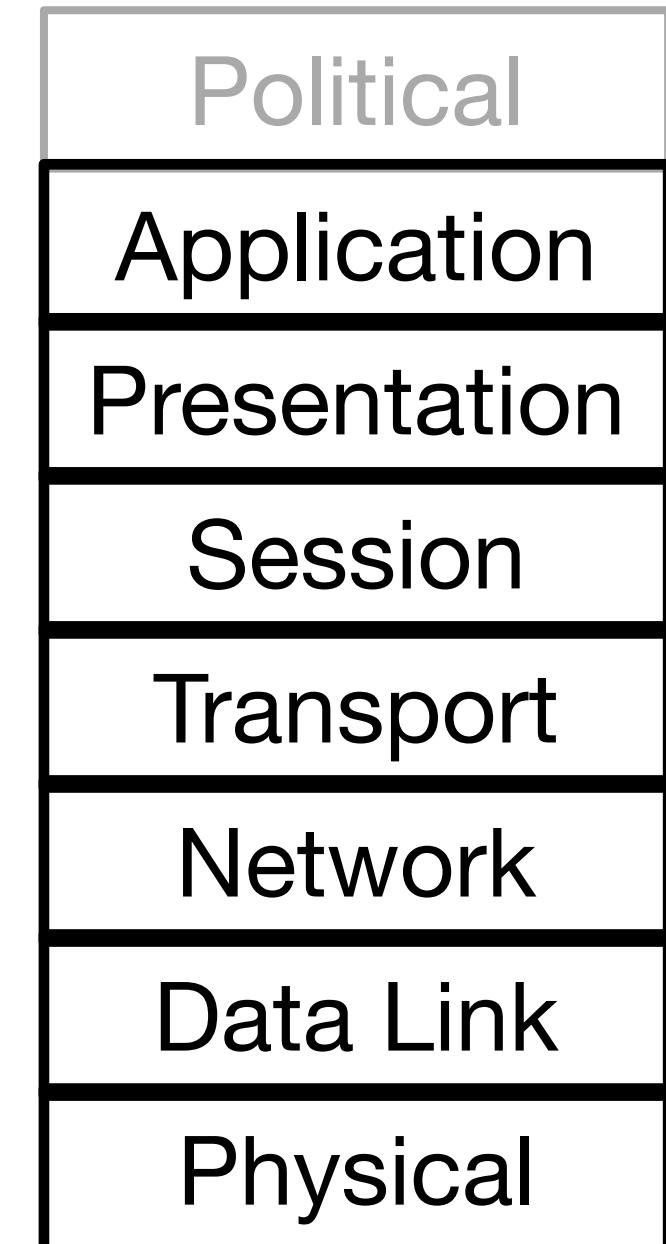
- Bits on wire or air
- Packets on wire or air
- Delivery packets within a single physical network
- Deliver packets across multiple networks
- Ensure the destination received the data
- Create data at the sender and make use of the data at the receiver

# Protocol for Networks of Networks?

Lots to do and at multiple levels!

Use abstraction to cope with complexity of communication

- Networks are like ~~egress~~ onions
  - Hierarchy of layers - network “stack”:
    - Application (chat client, game, etc.)
    - Transport (TCP, UDP)
    - Network (IP)
    - Data Link Layer (ethernet)
    - Physical Link (copper, wireless, etc.)



OSI 7 Layer Network Model

# Protocol Family Concept

- Protocol: packet structure and control commands to manage communication
- Protocol families (suites): a set of cooperating protocols that implement the network stack
- Key to protocol families is that communication occurs logically at the same level of the protocol, called peer-to-peer...  
...but is implemented via services at the next lower level
- Encapsulation: carry higher level information within lower level “envelope”

# Inspiration...

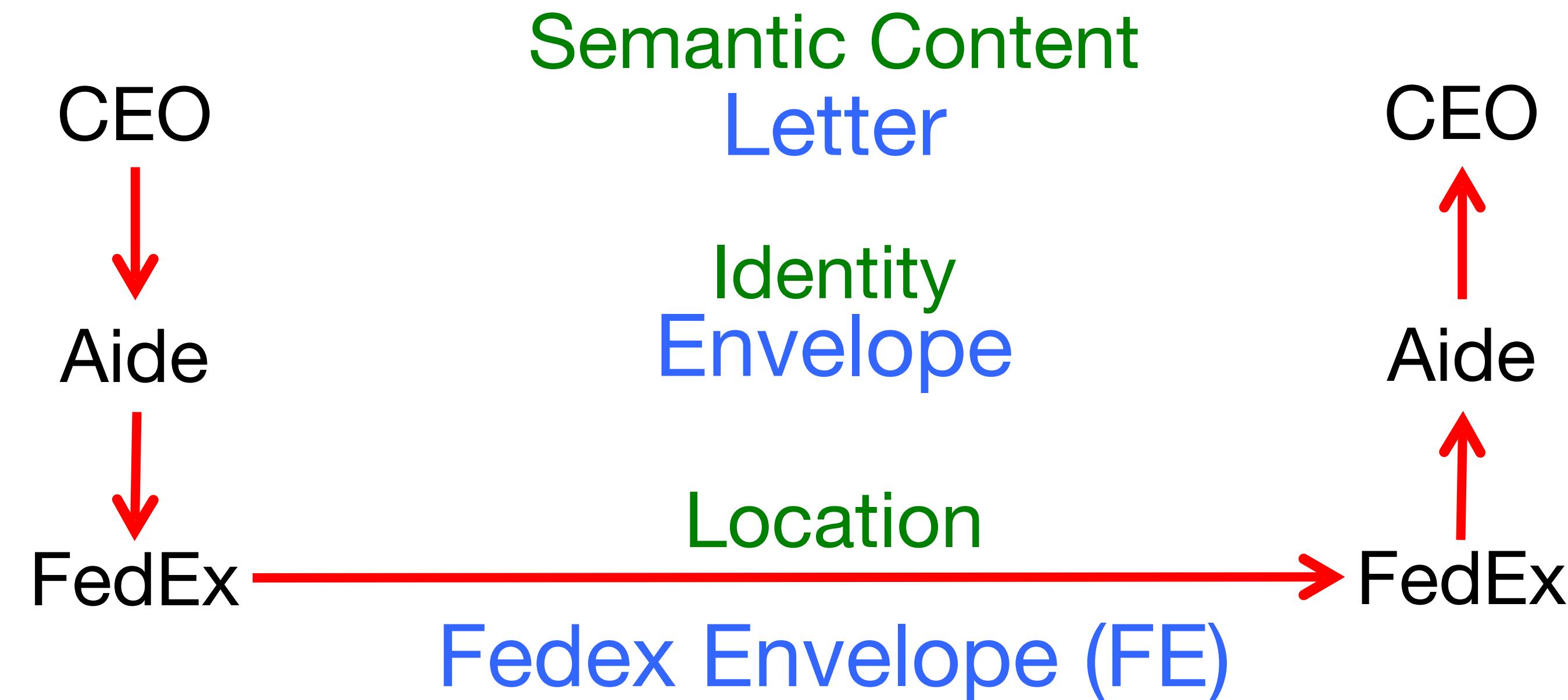
- CEO A writes letter to CEO B
  - Folds letter and hands it to assistant
- Assistant:
  - Puts letter in envelope with CEO B's full name
  - Takes to FedEx
- FedEx Office
  - Puts letter in larger envelope
  - Puts name and street address on FedEx envelope
  - Puts package on FedEx delivery truck
- FedEx delivers to other company

*Dear Blair,  
Your days are numbered.*

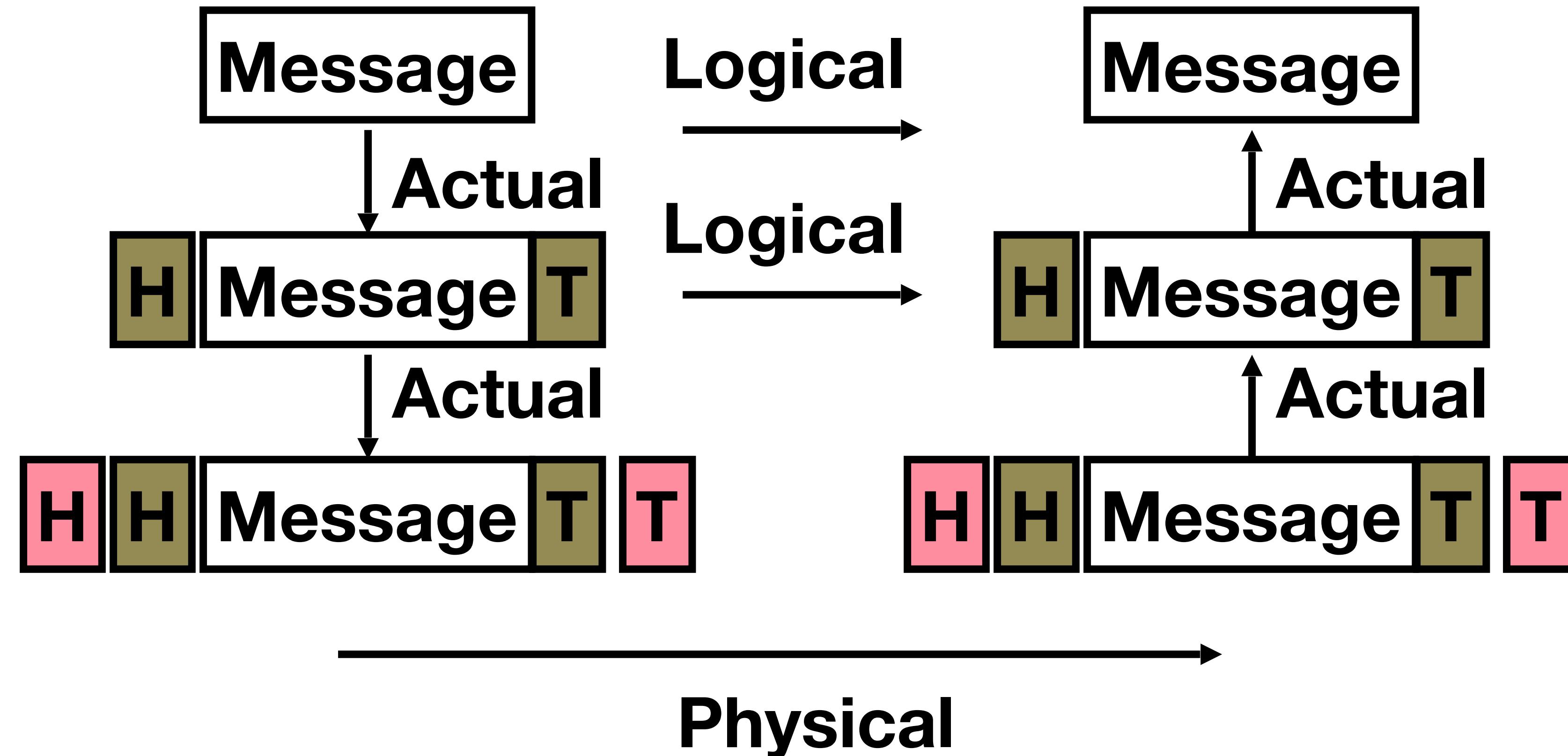
*-Pat*

# The Path of the Letter

“Peers” on each side understand the same things.  
No one else needs to.  
Lowest level has most packaging.



# Protocol Family Concept



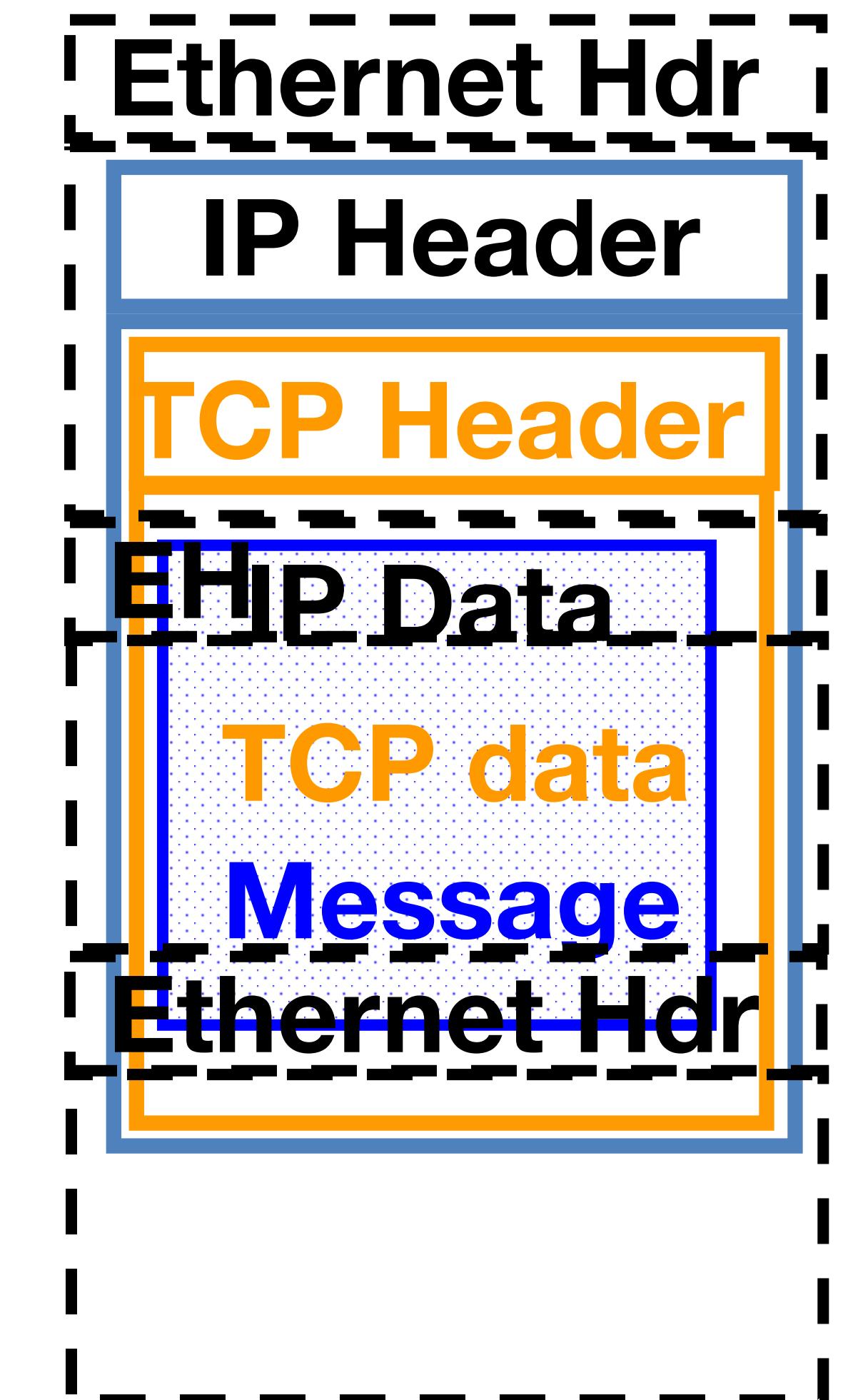
*Each lower level of stack “encapsulates” information from layer above by adding header and trailer.*

# Most Popular Protocol for Network of Networks

- Transmission Control Protocol/Internet Protocol (TCP/IP)
- This protocol family is the basis of the Internet, a WAN (wide area network) protocol
  - IP makes best effort to deliver
    - Packets can be lost, corrupted
      - But corrupted packets should be turned into lost packets
  - TCP guarantees ***reliable, in-order*** delivery
  - TCP/IP so popular it is used even when communicating locally: even across homogeneous LAN (local area network)

# TCP/IP packet, Ethernet packet, protocols

- Application sends message
  - TCP breaks into 64KiB segments, adds 20B header
  - IP adds 20B header, sends to network
  - If Ethernet, broken into 1500B packets with headers, trailers



# “And in conclusion...”

- I/O gives computers a way to interact with the world (sensing and actuation).
- I/O speed range is 100-million to one
- Polling vs. Interrupts
- DMA to avoid wasting CPU time on data transfers
- Disks for persistent storage, replaced by flash
- Network for communicating to the rest of the planet