inst.eecs.berkeley.edu/~cs61c

CS61C : Machine Structures

Lecture #28 – I/O, networks, disks

2008-8-07

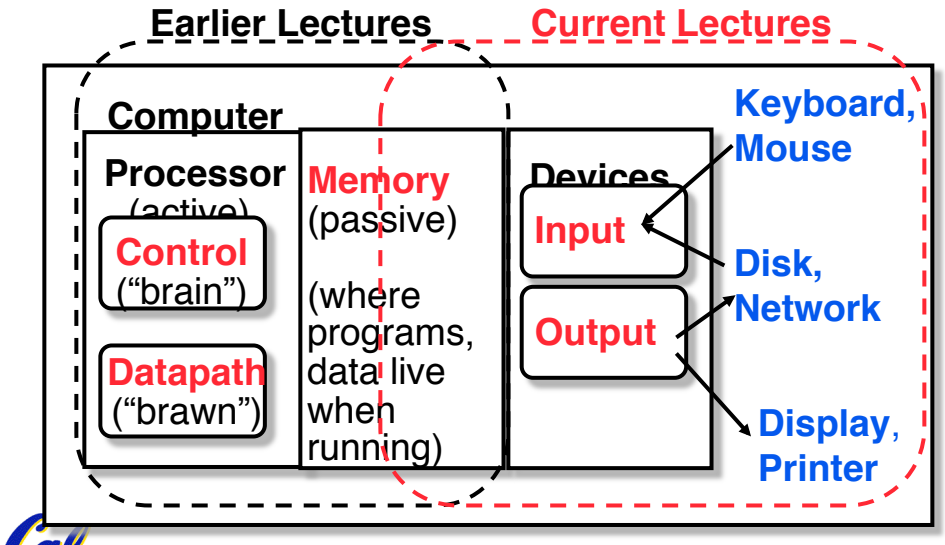
Cal legend
Jason Kidd

Albert Chae, Instructor

Most important topics today

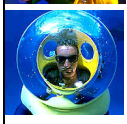
- I/O bus (computer bus)
- Polling vs interrupts
- Switch vs shared network
- Network protocol layers
- Various types of RAID

Recall : 5 components of any Computer



Motivation for Input/Output

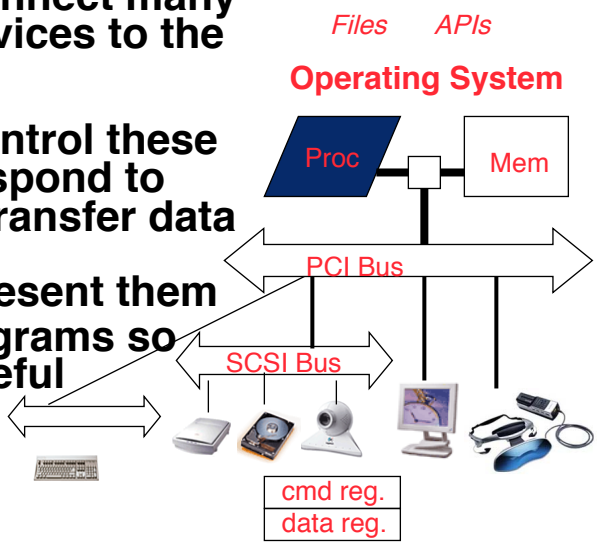
- I/O is how humans interact with computers
- I/O gives computers long-term memory.
- I/O lets computers do amazing things:



- Read pressure of synthetic hand and control synthetic arm and hand of fireman
- Control propellers, fins, communicate in BOB (Breathable Observable Bubble)
- Computer without I/O like a car without steering wheel or door; great technology, but gets you nowhere

What do we need to make I/O work?

- A way to connect many types of devices to the Proc-Mem
- A way to control these devices, respond to them, and transfer data
- A way to present them to user programs so they are useful

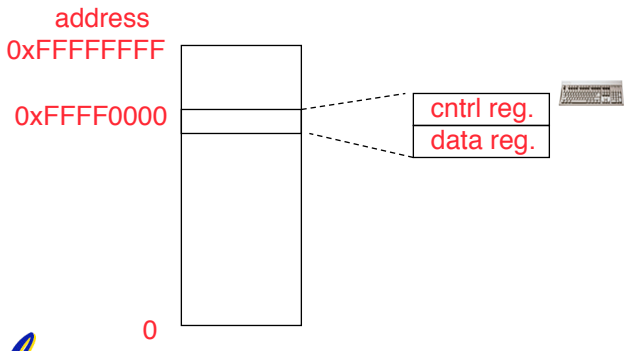


Instruction Set Architecture for I/O

- What must the processor do for I/O?
  - Input: reads a sequence of bytes
  - Output: writes a sequence of bytes
- Some processors have special input and output instructions
- Alternative model (used by MIPS):
  - Use loads for input, stores for output
  - Called "Memory Mapped Input/Output"
  - A portion of the address space dedicated to communication paths to Input or Output devices (no memory there)

Memory Mapped I/O

- Certain addresses are not regular memory
- Instead, they correspond to registers in I/O devices



Processor-I/O Speed Mismatch

- 1GHz microprocessor can execute 1 billion load or store instructions per second, or 4,000,000 KB/s data rate
  - I/O devices data rates range from 0.01 KB/s to 125,000 KB/s
- Input: device may not be ready to send data as fast as the processor loads it
  - Also, might be waiting for human to act
- Output: device not be ready to accept data as fast as processor stores it

What to do?

I/O Device Examples and Speeds

- I/O Speed: bytes transferred per second (from mouse to Gigabit LAN: 10-million-to-1)

• Device	Behavior	Partner	Data Rate
Keyboard	Input	Human	0.01
Mouse	Input	Human	0.02
Voice output	Output	Human	5.00
Floppy disk	Storage	Machine	50.00
Laser Printer	Output	Human	100.00
Magnetic Disk	Storage	Machine	10,000.00
Wireless Network	I or O	Machine	10,000.00
Graphics Display	Output	Human	30,000.00
Wired LAN Network	I or O	Machine	125,000.00

When discussing transfer rates, use 10^x

Processor Checks Status before Acting

- Path to device generally has 2 registers:
  - Control Register, says it's OK to read/write (I/O ready) [think of a flagman on a road]
  - Data Register, contains data
- Processor reads from Control Register in loop, waiting for device to set Ready bit in Control reg (0 ⇒ 1) to say its OK
- Processor then loads from (input) or writes to (output) data register
  - Load from or Store into Data Register resets Ready bit (1 ⇒ 0) of Control Register

I/O Example (don't need to understand code)

- Input: Read from keyboard into `$v0`

```
                lui   $t0, 0xffff #ffff0000
Waitloop:       lw    $t1, 0($t0) #control
                andi  $t1,$t1,0x1
                beq   $t1,$zero, Waitloop
                lw    $v0, 4($t0) #data
```

- Output: Write to display from `$a0`

```
                lui   $t0, 0xffff #ffff0000
Waitloop:       lw    $t1, 8($t0) #control
                andi  $t1,$t1,0x1
                beq   $t1,$zero, Waitloop
                sw    $a0, 12($t0) #data
```

- Processor waiting for I/O called "Polling"
- "Ready" bit is from processor's point of view!

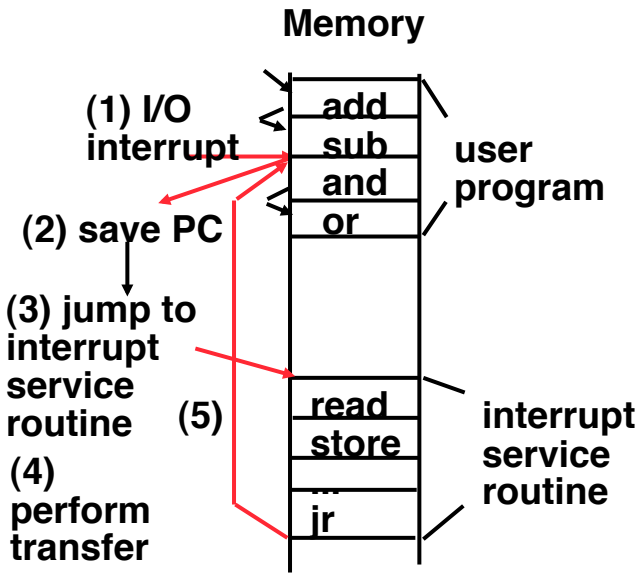
What is the alternative to polling?

- Wasteful to have processor spend most of its time "spin-waiting" for I/O to be ready
- Would like an unplanned procedure call that would be invoked only when I/O device is ready
- Solution: use exception mechanism to help I/O. Interrupt program when I/O ready, return when done with data transfer

I/O Interrupt

- An I/O interrupt is like overflow exceptions except:
  - An I/O interrupt is "asynchronous"
  - More information needs to be conveyed
- An I/O interrupt is asynchronous with respect to instruction execution:
  - I/O interrupt is not associated with any instruction, but it can happen in the middle of any given instruction
  - I/O interrupt does not prevent any instruction from completion

Interrupt-Driven Data Transfer



Generalizing Interrupts

- We can handle all sorts of exceptions with interrupts.
- Big idea: jump to handler that knows what to do with each interrupt, then jump back
- Our types: syscall, overflow, mmio ready.

Peer Instruction

A. A faster CPU will result in faster I/O.

B. Hardware designers handle mouse input with interrupts since it is better than polling in almost all cases.

C. Low-level I/O is actually quite simple, as it's really only reading and writing bytes.

	ABC
0:	FFF
1:	FFT
2:	FTF
3:	FTT
4:	TFF
5:	TFT
6:	TTF
7:	TTT

Peer Instruction Answer

A. Less sync data idle time

B. Because mouse has low I/O rate polling often used

C. Concurrency, device requirements vary!

A. A faster CPU will result in faster I/O. TRUE

B. Hardware designers handle mouse input with interrupts since it is better than polling in almost all cases. FALSE

C. Low-level I/O is actually quite simple, as it's really only reading and writing bytes. FALSE

	ABC
0:	FFF
1:	FFT
2:	FTF
3:	FTT
4:	TFF
5:	TFT
6:	TTF
7:	TTT

Administrivia

- Lab 14
- Proj3 - Face to face grading?
- Proj4 out soon. Find a partner.
- Quiz 13,14 up, due Monday
- Final 8/14 – 9:30-12:30pm in 105 North Gate
- Final Review Session probable on Tuesday, 1-3pm
- Course Surveys
  - HKN Course Survey during Tuesday lecture
  - Online Survey posted early next week
  - 1 points extra added for each survey (still

Upcoming Calendar

Time	Monday	Tuesday	Wednesday	Thursday
Lecture	Performance (Bill)	Summary, Upper div demos, Course Evaluations	Summary, Upper div demos	FINAL 9:30-12:30 pm @ 105 North Gate
Afternoon/ Evening	Extended Office hours?	Review Session 1-3 pm TBD	Last Discussion Section	

Why Networks?

- Originally *sharing I/O devices* between **computers**
  ex: printers
- Then *communicating* between **computers**
  ex: file transfer protocol
- Then *communicating* between **people**
  ex: e-mail
- Then *communicating* between **networks of computers**
  ex: file sharing, www, …

How Big is the Network (2007)?

~30 in **273 Soda**

~525 in **inst.cs.berkeley.edu**

~6,400 in **eecs & cs .berkeley.edu**

(1999) ~50,000 in **berkeley.edu**

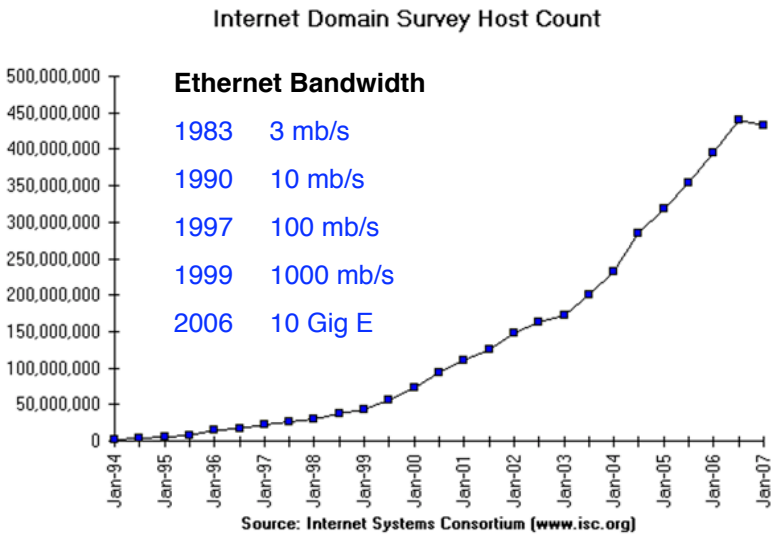
~10,000,000 in **.edu** (2005: ~9,000,000)

~258,941,310 in **US** (2005: ~217,000,000, 2006: ~286.5E6)
(.net .com .edu .arpa .us .mil .org .gov)

~433,190,000 in **the world** (2005: ~317,000,000, 2006: ~439,000,000)

Source: Internet Software Consortium: www.isc.org

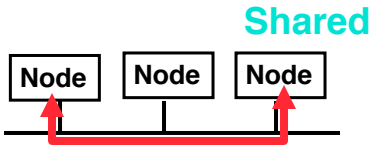
Growth Rate

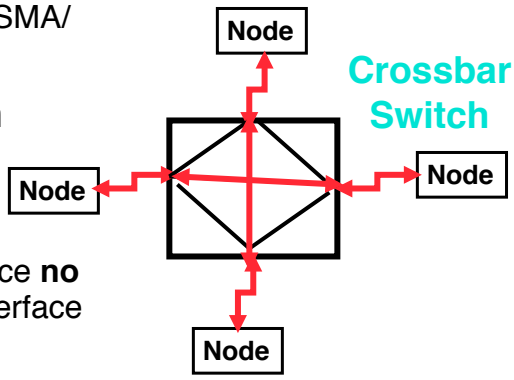


en.wikipedia.org/wiki/10_gigabit_ethernet

Shared vs. Switched Based Networks

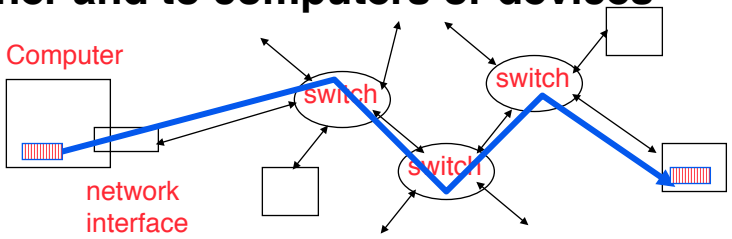
- **Shared vs. Switched:**
  - **Switched:** pairs ("point-to-point" connections) communicate at same time
  - **Shared:** 1 at a time (CSMA/CD)
- **Aggregate bandwidth (BW) in switched network is many times shared:**
  - point-to-point faster since **no arbitration**, simpler interface





What makes networks work?

- **links connecting switches 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)**

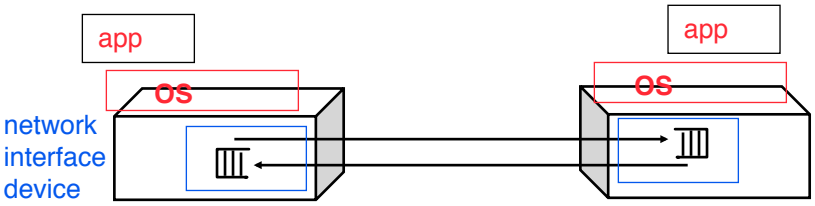
Typical Types of Networks

- Local Area Network (Ethernet)
  - Inside a building: Up to 1 km
  - (peak) Data Rate: 10 Mbits/sec, 100 Mbits /sec, 1000 Mbits/sec (1.25, 12.5, 125 MBytes/s)
  - Run, installed by network administrators
- Wide Area Network
  - Across a continent (10km to 10000 km)
  - (peak) Data Rate: 1.5 Mb/s to 40000 Mb/s
  - Run, installed by telecommunications companies (Sprint, UUNet[MCI], AT&T)
- Wireless Networks (LAN), ...

The Sprint U.S. Topology (2001)

ABCs of Networks: 2 Computers

- **Starting Point**: Send bits between 2 computers



- Queue (First In First Out) on each end
- Can send both ways ("Full Duplex")
  - One-way information is called "Half Duplex"
- Information sent called a "message"
  - Note: Messages also called packets

A Simple Example: 2 Computers

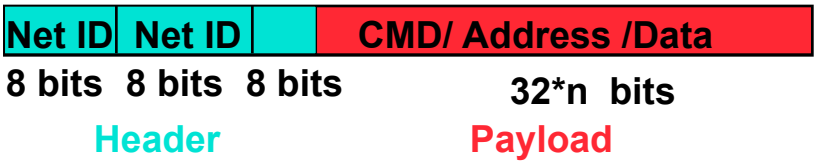
- What is Message Format?
  - Similar idea to Instruction Format
  - Fixed size? Number bits?

8 bit 32 x Length bits

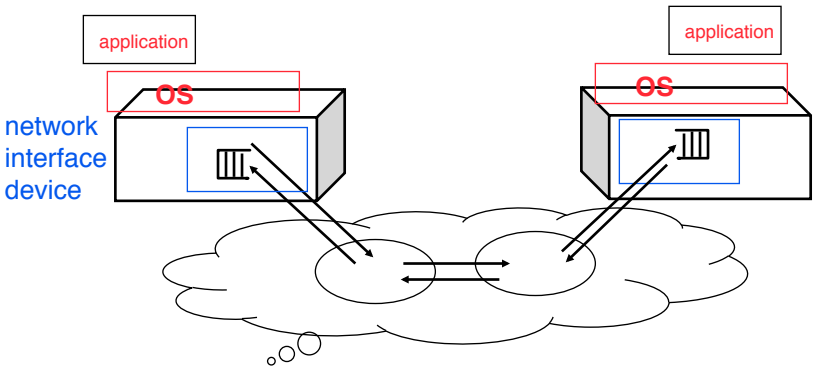
- Header (Trailer): information to deliver message
- Payload: data in message
- What can be in the data?
  - anything that you can represent as bits
  - values, chars, commands, addresses...

Questions About Simple Example

- What if more than 2 computers want to communicate?
  - Need computer "address field" in packet to know:
    - which computer should receive it (destination)
    - which computer to reply to (source)
  - Just like envelopes!



ABCs: many computers



- switches and routers interpret the header in order to deliver the packet
- source encodes and destination decodes content of the payload

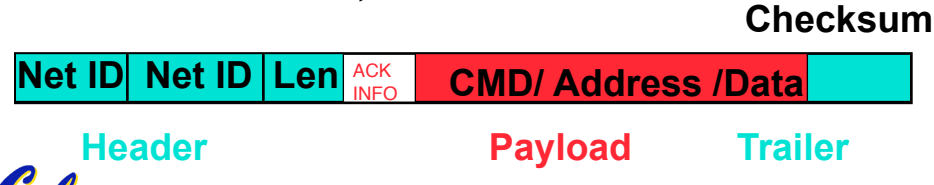
Questions About Simple Example

- What if message is garbled in transit?
- Add redundant information that is checked when message arrives to be sure it is OK
- 8-bit sum of other bytes: called "Check sum"; upon arrival compare check sum to sum of rest of information in message. `xor` also popular.

Learn about Checksums in Math 55/CS 70…

Questions About Simple Example

- What if message never arrives?
- Receiver tells sender when it arrives
  - Send an ACK (ACKnowledgement) [like registered mail]
  - Sender retries if waits too long
- Don't discard message until it is ACK'ed
- If check sum fails, don't send ACK



Observations About Simple Example

- Simple questions (like those on the previous slides) lead to:
  - more complex procedures to send/receive message
  - more complex message formats
- Protocol: algorithm for properly sending and receiving messages (packets)

  …an agreement on how to communicate

Software Protocol to Send and Receive

- SW Send steps
  1: Application copies data to OS buffer
  2: OS calculates checksum, starts timer
  3: OS sends data to network interface HW and says start
- SW Receive steps
  3: OS copies data from network interface HW to OS buffer
  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

Protocol for Networks of Networks?

- Abstraction to cope with complexity of communication
- Networks are like onions
  - Hierarchy of layers:
    - Application (chat client, game, etc.)
    - Transport (TCP, UDP)
    - Network (IP)
    - Physical Link (wired, wireless, etc.)

Networks are like onions.

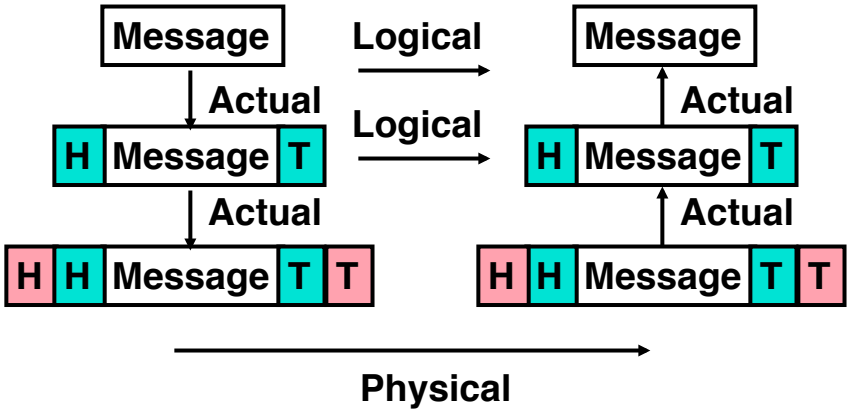
They stink?

Yes. No!

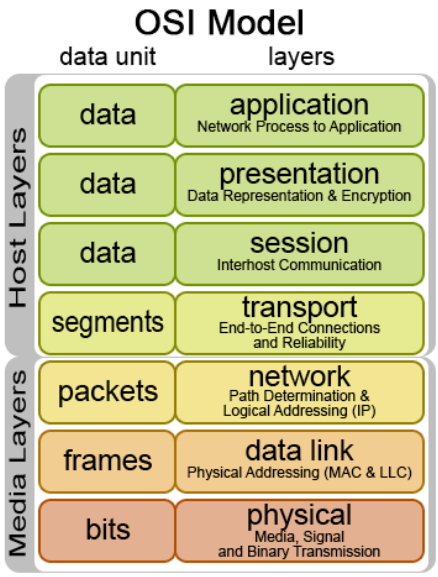
Oh, they make you cry.

No!… Layers. Onions have layers. Networks have layers.

Protocol Family Concept



OSI Model



http://wiki.go6.net/images/2/2b/Osi-model.png

Protocol Family Concept

- 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"
- Fragmentation: break packet into multiple smaller packets and reassemble

Overhead vs. Bandwidth

- Networks are typically advertised using peak bandwidth of network link: e.g., 100 Mbits/sec Ethernet ("100 base T")
- Software overhead to put message into network or get message out of network often limits useful bandwidth
- Assume overhead to send and receive = 320 microseconds (μs), want to send 1000 Bytes over "100 Mbit/s" Ethernet
  - Network transmission time: 1000Bx8b/B /100Mb/s = 8000b / (100b/μs) = 80 μs
  - Effective bandwidth: 8000b/(320+80)μs = 20 Mb/s

Cost of Polling?

- Assume for a processor with a 1GHz clock it takes 400 clock cycles for a polling operation (call polling routine, accessing the device, and returning). Determine % of processor time for polling
  - Mouse: polled 30 times/sec so as not to miss user movement
  - Floppy disk: transfers data in 2-Byte units and has a data rate of 50 KB/second. No data transfer can be missed.
  - Hard disk: transfers data in 16-Byte chunks and can transfer at 16 MB/second. Again, no transfer can be missed.

% Processor time to poll [p. 677 in book]

Mouse Polling, Clocks/sec

= 30 [polls/s] * 400 [clocks/poll] = 12K [clocks/s]

- % Processor for polling:

  $12*10^3$ [clocks/s] / $1*10^9$ [clocks/s] = 0.0012%

  ⇒ Polling mouse little impact on processor

Frequency of Polling Floppy

= 50 [KB/s] / 2 [B/poll] = 25K [polls/s]

- Floppy Polling, Clocks/sec

  = 25K [polls/s] * 400 [clocks/poll] = 10M [clocks/s]

- % Processor for polling:

  $10*10^6$ [clocks/s] / $1*10^9$ [clocks/s] = 1%

% Processor time to poll hard disk

Frequency of Polling Disk

= 16 [MB/s] / 16 [B] = 1M [polls/s]

- Disk Polling, Clocks/sec

  = 1M [polls/s] * 400 [clocks/poll]
  = 400M [clocks/s]

- % Processor for polling:

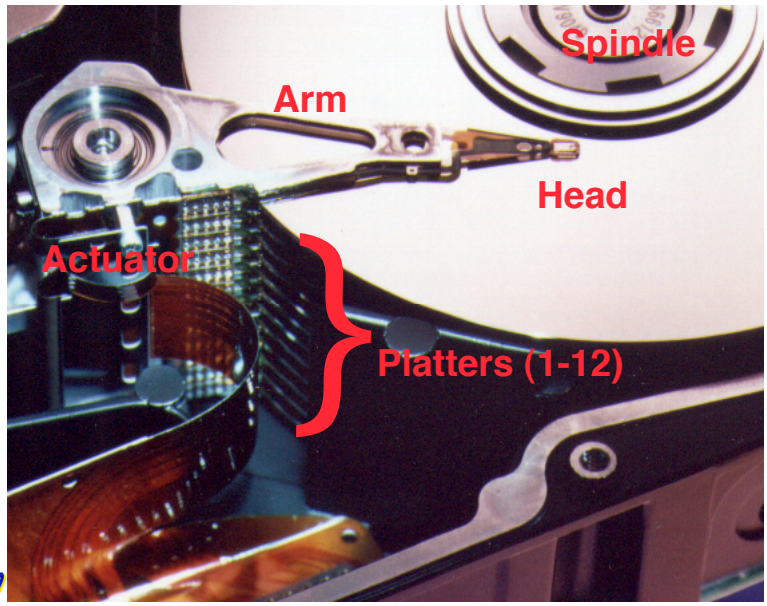
  $400*10^6$ [clocks/s] / $1*10^9$ [clocks/s] = 40%

  ⇒ Unacceptable

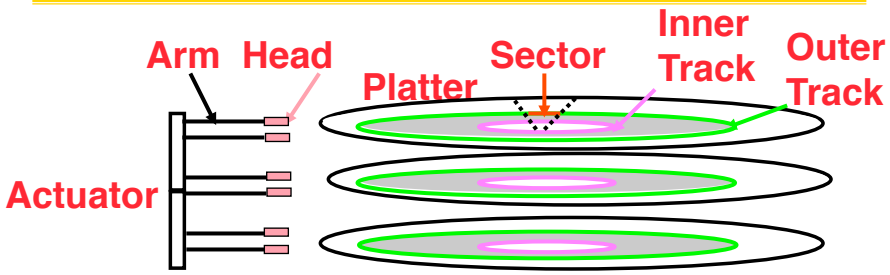
Magnetic Disk – common I/O device

- A kind of computer memory
  - Information sorted by magnetizing ferrite material on surface of rotating disk (similar to tape recorder except digital rather than analog data)
- Nonvolatile storage
  - retains its value without applying power to disk.
- Two Types
  - Floppy disks – slower, less dense, removable.
  - Hard Disk Drives (HDD) – faster, more dense, non-removable.
- Purpose in computer systems (Hard Drive):
  - Long-term, inexpensive storage for files
  - "Backup" for main-memory. Large, inexpensive, slow level in the memory hierarchy (virtual memory)

Photo of Disk Head, Arm, Actuator

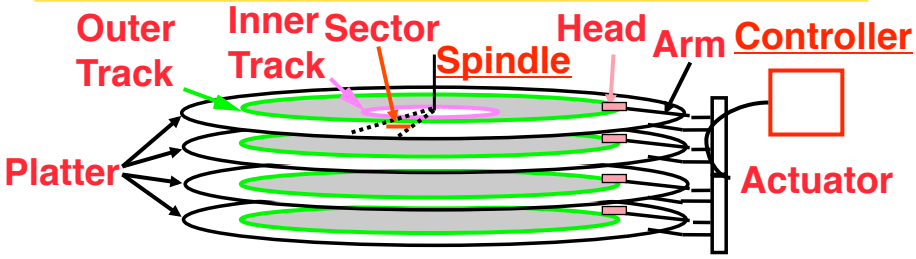


Disk Device Terminology



- 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

Disk Device Performance (1/2)



- Disk Latency = Seek Time + Rotation Time + Transfer Time + Controller Overhead
  - Seek Time? depends on no. tracks to move arm, speed of actuator
  - Rotation Time? depends on speed disk rotates, how far sector is from head
  - Transfer Time? depends on data rate (bandwidth) of disk (f(bit density,rpm)), size of request

Disk Device Performance (2/2)

- Average distance of sector from head?
- 1/2 time of a rotation
  - 7200 Revolutions Per Minute ⇒ 120 Rev/sec
  - 1 revolution = 1/120 sec ⇒ 8.33 milliseconds
  - 1/2 rotation (revolution) ⇒ 4.17 ms
- Average no. tracks to move arm?
  - Disk industry standard benchmark:
    - Sum all time for all possible seek distances from all possible tracks / # possible
    - Assumes average seek distance is random
- Size of Disk cache can strongly affect perf!
  - Cache built into disk system, OS knows nothing

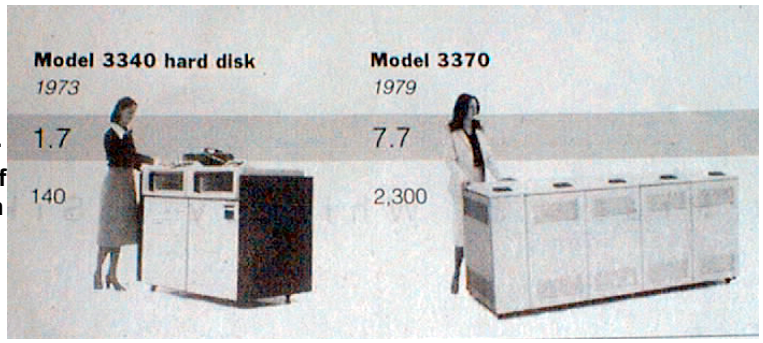
Data Rate: Inner vs. Outer Tracks

- To keep things simple, originally same number of sectors per track
  - Since outer track longer, lower bits per inch
- Competition ⇒ decided to keep bits per inch (BPI) high for all tracks ("constant bit density")
  - ⇒ More capacity per disk
  - ⇒ More sectors per track towards edge
  - ⇒ Since disk spins at constant speed, outer tracks have faster data rate
- Bandwidth outer track 1.7x inner track!

Early Disk History (IBM)

Data density Mbit/sq. in.

Capacity of unit shown Megabytes

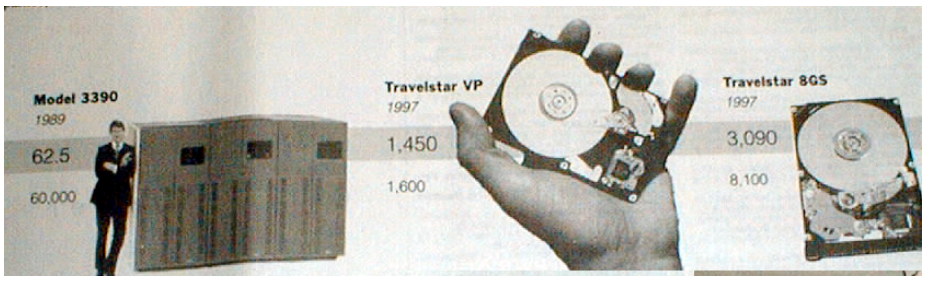


1973:
1. 7 Mbit/sq. in
140 MBytes

1979:
7. 7 Mbit/sq. in
2,300 MBytes

source: New York Times, 2/23/98, page C3,
"Makers of disk drives crowd even more data into even smaller spaces"

Early Disk History



1989:
63 Mbit/sq. in
60,000 MBytes

1997:
1450 Mbit/sq. in
1600 MBytes

1997:
3090 Mbit/sq. in
8100 MBytes

source: New York Times, 2/23/98, page C3,
"Makers of disk drives crowd even more data into even smaller spaces"

Disk Performance Model /Trends

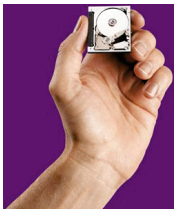
- Capacity : + 100% / year (2X / 1.0 yrs)
  - Over time, grown so fast that # of platters has reduced (some even use only 1 now!)
- Transfer rate (BW) : + 40%/yr (2X / 2 yrs)
- Rotation+Seek time : – 8%/yr (1/2 in 10 yrs)
- Areal Density
  - Bits recorded along a track: Bits/Inch (BPI)
  - # of tracks per surface: Tracks/Inch (TPI)
  - We care about bit density per unit area
  - Called Areal Density = BPI x TPI
  - "~120 Gb/In^2 is longitudinal limit"
  - "230 Gb/In^2 now with perpendicular"

- GB/$: > 100%/year (2X / 1.0 yrs)
  - Fewer chips + areal density

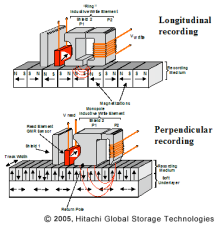
State of the Art: Two camps (2006)

- Performance
  - Enterprise apps, servers
- E.g., Seagate Cheetah 15K.5
  - Ultra320 SCSI, 3 Gbit/sec, Serial Attached SCSI (SAS), 4Gbit/sec Fibre Channel (FC)
  - 300 GB, 3.5-inch disk
  - 15,000 RPM
  - 13 watts (idle)
  - 3.5 ms avg. seek
  - 125 MB/s transfer rate
  - 5 year warrantee
  - $1000 = $3.30 / GB
- Capacity
  - Mainstream, home uses
- E.g., Seagate Barracuda 7200.10
  - Serial ATA 3Gb/s (SATA/300), Serial ATA 1.5Gb/s (SATA/150), Ultra ATA/100
  - 750 GB, 3.5-inch disk
  - 7,200 RPM
  - 9.3 watts (idle)
  - 8.5 ms avg. seek
  - 78 MB/s transfer rate
  - 5 year warrantee
  - $350 = $0.46 / GB
- Uses Perpendicular Magnetic Recording (PMR)!!
  - What's that, you ask?

1 inch disk drive!



- Hitachi 2007 release
  - Development driven by iPods & digital cameras
  - 20GB, 5-10MB/s (higher?)
  - 42.8 x 36.4 x 5 mm
- Perpendicular Magnetic Recording (PMR)
  - FUNDAMENTAL new technique
  - Evolution from Logitudinal
    - Starting to hit physical limit due to superparamagnetism
  - They say 10x improvement



www.hitachi.com/New/cnews/050405.html
www.hitachigst.com/hdd/research/recording_head/pr/

Where does Flash memory come in?

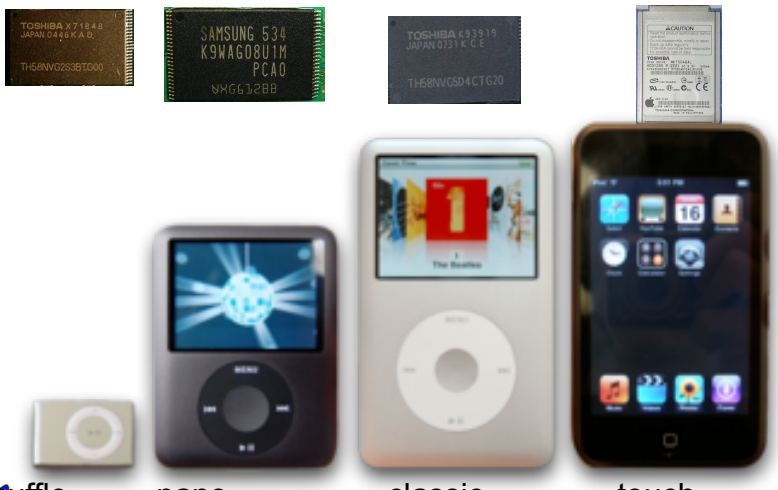
- Microdrives and Flash memory (e.g., CompactFlash) are going head-to-head
  - Both non-volatile (no power, data ok)
  - Flash benefits: durable & lower power (no moving parts, need to spin μdrives up/down)
  - Flash limitations: finite number of write cycles (wear on the insulating oxide layer around the charge storage mechanism)
- How does Flash memory work?
  - NMOS transistor with an additional conductor between gate and source/drain which "traps" electrons. The p absence is a 1 or 0.

en.wikipedia.org/wiki/Flash_memory

en.wikipedia.org/wiki/Ipodwww.apple.com/ipod

What does Apple put in its iPods?

Toshiba flash 1, 2GB	Samsung flash 4, 8GB	Toshiba flash 8, 16, 32GB	Toshiba 1.8-inch HDD 80, 160GB

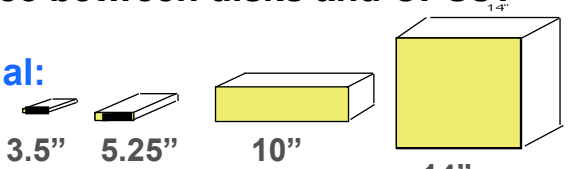


shuffle, nano, classic, touch

Use Arrays of Small Disks…

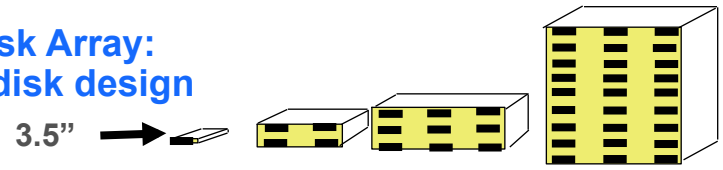
- Katz and Patterson asked in 1987:
  - Can smaller disks be used to close gap in performance between disks and CPUs?

Conventional: 4 disk designs



Low End → High End

Disk Array: 1 disk design



Replace Small Number of Large Disks with Large Number of Small Disks! (1988 Disks)

	IBM 3390K	IBM 3.5" 0061	x70	
Capacity	20 GBytes	320 MBytes	23 GBytes	
Volume	97 cu. ft.	0.1 cu. ft.	11 cu. ft.	9X
Power	3 KW	11 W	1 KW	3X
Data Rate	15 MB/s	1.5 MB/s	120 MB/s	8X
I/O Rate	600 I/Os/s	55 I/Os/s	3900 IOs/s	6X
MTTF	250 KHrs	50 KHrs	??? Hrs	
Cost	$250K	$2K	$150K	

Disk Arrays potentially high performance, high MB per cu. ft., high MB per KW,
but what about reliability?

Array Reliability

- Reliability - whether or not a component has failed
  - measured as Mean Time To Failure (MTTF)
- Reliability of N disks
  = Reliability of 1 Disk ÷ N
  (assuming failures independent)
  - 50,000 Hours ÷ 70 disks = 700 hour
- Disk system MTTF:
  Drops from 6 years to 1 month!
- Disk arrays too unreliable to be useful!

Redundant Arrays of (Inexpensive) Disks

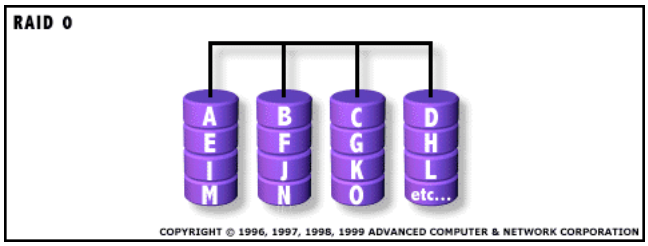
- Files are "striped" across multiple disks
- Redundancy yields high data availability
  - Availability: service still provided to user, even if some components failed
- Disks will still fail
- Contents reconstructed from data redundantly stored in the array
  - ⇒ Capacity penalty to store redundant info
  - ⇒ Bandwidth penalty to update redundant info

Berkeley History, RAID-I

Berkeley
University of California

- RAID-I (1989)
  - Consisted of a Sun 4/280 workstation with 128 MB of DRAM, four dual-string SCSI controllers, 28 5.25-inch SCSI disks and specialized disk striping software
- Today RAID is > tens billion dollar industry, 80% non-PC disks sold in RAIDs

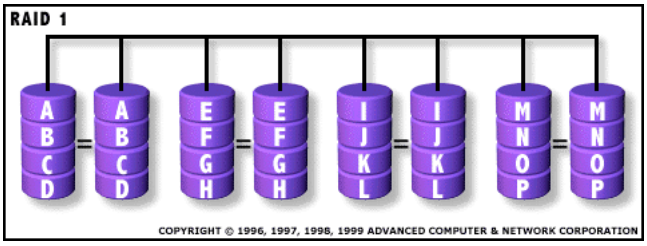
"RAID 0": No redundancy = "AID"



- Assume have 4 disks of data for this example, organized in blocks
- Large accesses faster since transfer from several disks at once

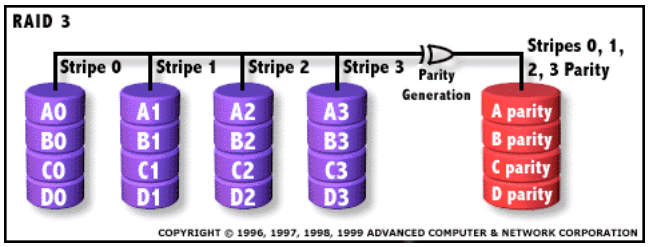
This and next 5 slides from RAID.edu, http://www.acnc.com/04_01_00.html
`http://www.raid.com/04_00.html` also has a great tutorial

RAID 1: Mirror data



- Each disk is fully duplicated onto its "mirror"
  - Very high availability can be achieved
- Bandwidth reduced on write:
  - 1 Logical write = 2 physical writes
- Most expensive solution: 100% capacity overhead

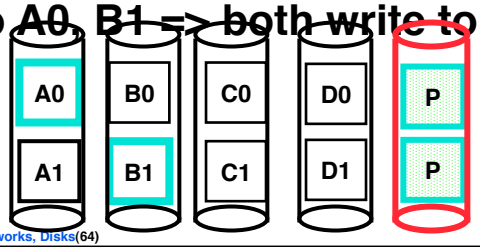
RAID 3: Parity



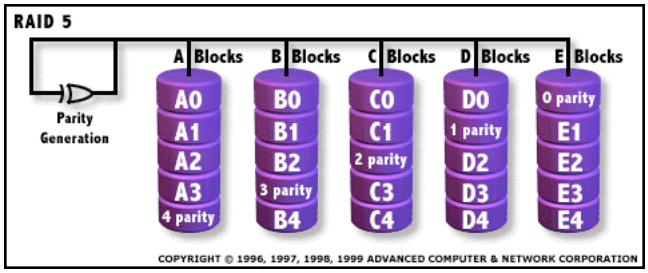
- Parity computed across group to protect against hard disk failures, stored in P disk
- Logically, a single high capacity, high transfer rate disk
- 25% capacity cost for parity in this example vs. 100% for RAID 1 (5 disks vs. 8 disks)

Inspiration for RAID 5 (RAID 4 block-striping)

- Small writes (write to one disk):
  - Option 1: read other data disks, create new sum and write to Parity Disk (access all disks)
  - Option 2: since P has old sum, compare old data to new data, add the difference to P: 1 logical write = 2 physical reads + 2 physical writes to 2 disks
- Parity Disk is bottleneck for Small writes: Write to A0, B1 => both write to P disk



RAID 5: Rotated Parity, faster small writes



- Independent writes possible because of interleaved parity
  - Example: write to A0, B1 uses disks 0, 1, 4, 5, so can proceed in parallel
  - Still 1 small write = 4 physical disk accesses

en.wikipedia.org/wiki/Redundant_array_of_independent_disks

Peer Instruction

1. RAID 1 (mirror) and 5 (rotated parity) help with performance and availability
2. RAID 1 has higher cost than RAID 5
3. Small writes on RAID 5 are slower than on RAID 1

	ABC
0:	FFF
1:	FFT
2:	FTF
3:	FTT
4:	TFF
5:	TFT
6:	TTF
7:	TTT

Peer Instruction Answer

1. All RAID (0-5) helps with performance, only RAID0 doesn't help availability. TRUE
2. Surely! Must buy 2x disks rather than 1.25x (from diagram, in practice even less) TRUE
3. RAID5 (2R,2W) vs. RAID1 (2W). Latency worse, throughput (II writes) better. TRUE

1. RAID 1 (mirror) and 5 (rotated parity) help with performance and availability
2. RAID 1 has higher cost than RAID 5
3. Small writes on RAID 5 are slower than on RAID 1

	ABC
0:	FFF
1:	FFT
2:	FTF
3:	FTT
4:	TFF
5:	TFT
6:	TTF
7:	TTT

Summary – I/O

- I/O gives computers their 5 senses
- I/O speed range is 100-million to one
- Processor speed means must synchronize with I/O devices before use
- Polling works, but expensive
  - processor repeatedly queries devices
- Interrupts works, more complex
  - devices causes an exception, causing OS to run and deal with the device
- I/O control important factor of Operating Systems

Summary - Network

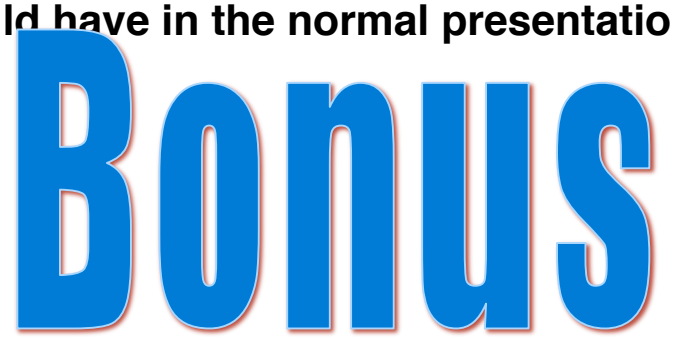
- Protocol suites allow networking of heterogeneous components
  - Another form of principle of abstraction
  - Protocols ⇒ operation in presence of failures
  - Standardization key for LAN, WAN
- Integrated circuit ("Moore's Law") revolutionizing network switches as well as processors
  - Switch just a specialized computer
- Trend from shared to switched networks to get faster links and scalable bandwidth
- Interested?

Summary - Disks

- Magnetic Disks continue rapid advance: 60%/yr capacity, 40%/yr bandwidth, slow on seek, rotation improvements, MB/$ improving 100%/yr?
  - Designs to fit high volume form factor
  - PMR a fundamental new technology
    - breaks through barrier
- RAID
  - Higher performance with more disk arms per $
  - Adds option for small # of extra disks
  - Can nest RAID levels
  - Today RAID is > tens-billion dollar industry, 80% nonPC disks sold in RAIDs, started at Cal

Bonus slides

- These are extra slides that used to be included in lecture notes, but have been moved to this, the "bonus" area to serve as a supplement.
- The slides will appear in the order they would have in the normal presentation



[Bonus] Protocol for Network of Networks

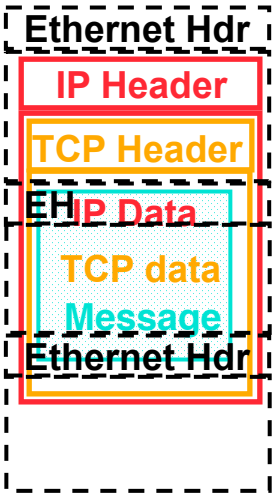
- **IP: Best-Effort Packet Delivery (Network Layer)**
- Packet switching
  - Send data in packets
  - Header with source & destination address
- "Best effort" delivery
  - Packets may be lost
  - Packets may be corrupted
  - Packets may be delivered out of order

[Bonus] Protocol for Network of Networks

- **Transmission Control Protocol/Internet Protocol (TCP/IP)**
  (TCP :: a Transport Layer)
  - This protocol family is the basis of the Internet, a WAN protocol
  - IP makes best effort to deliver
  - TCP guarantees delivery
  - TCP/IP so popular it is used even when communicating locally: even across homogeneous LAN

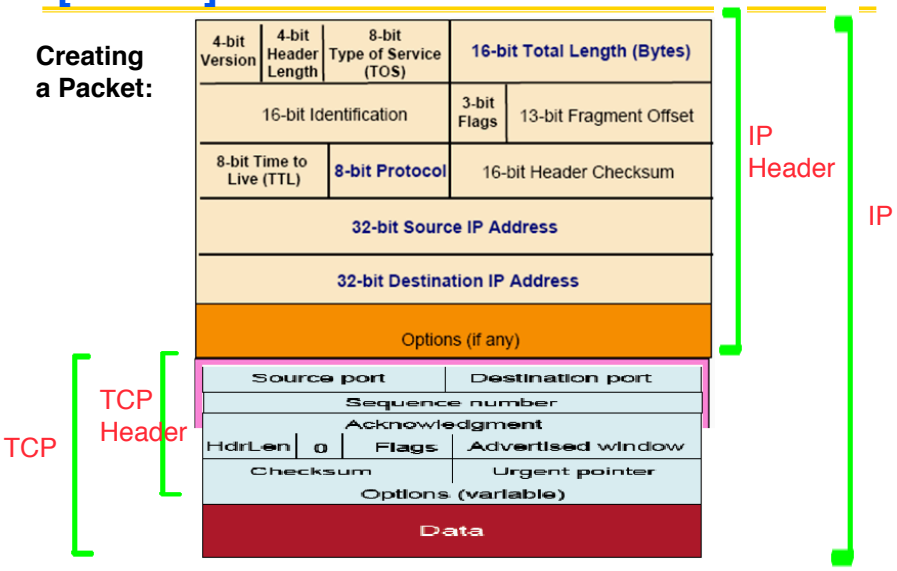
[Bonus] TCP/IP packet, Ethernet packet, protocols

- 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 (24B)
- All Headers, trailers have length field, destination, ...

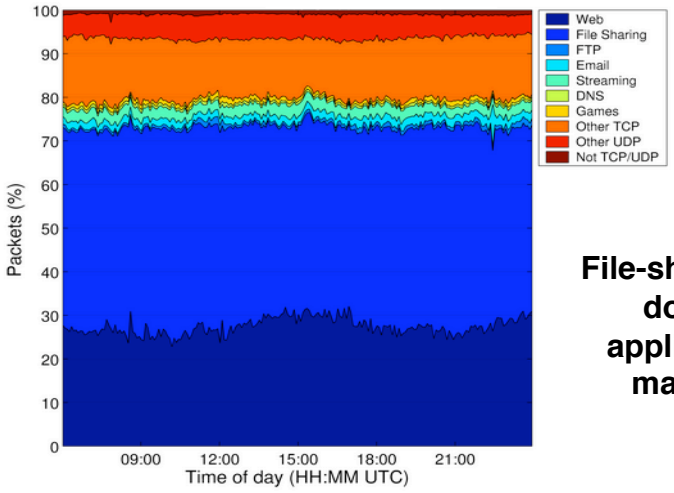


[Bonus] TCP/IP in action

Creating a Packet:



[Bonus] Backbone Link App Composition



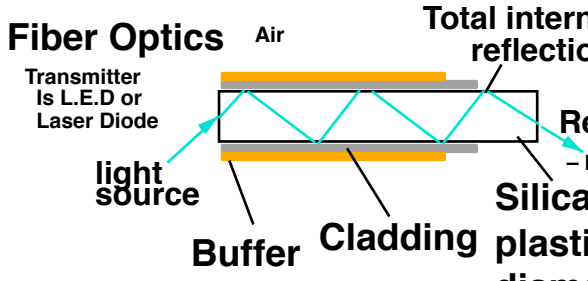
File-sharing is the dominant application on many links!

[Bonus] Example: Network Media

Twisted Pair ("Cat 5"):

Copper, 1mm think, twisted to avoid antenna effect

Fiber Optics



Light: 3 parts are cable, light source, light detector

Silica: glass or plastic; actually < 1/10 diameter of copper

[Bonus] Historical Perspective

- *Form factor* and *capacity* are more important in the marketplace than is performance
- Form factor evolution:

  1970s: Mainframes ⇒ 14 inch diameter disks

  1980s: Minicomputers, Servers
  ⇒ 8", 5.25" diameter disks

  Late 1980s/Early 1990s:
  - PCs ⇒ 3.5 inch diameter disks
  - Laptops, notebooks ⇒ 2.5 inch disks
  - Palmtops didn't use disks, so 1.8 inch diameter disks didn't make it

  Early 2000s:
  - MP3 players ⇒ 1 inch disks

[Bonus] Disk Performance Example

- Calculate time to read 1 sector (512B) for Deskstar using advertised performance; sector is on outer track

Disk latency = average seek time + average rotational delay + transfer time + controller overhead

= 8.5 ms + 0.5 * 1/(7200 RPM)
+ 0.5 KB / (100 MB/s) + 0.1 ms

= 8.5 ms + 0.5 /(7200 RPM/(60000ms/M))
+ 0.5 KB / (100 KB/ms) + 0.1 ms

= 8.5 + 4.17 + 0.005 + 0.1 ms = 12.77 ms

- How many CPU clock cycles is this?