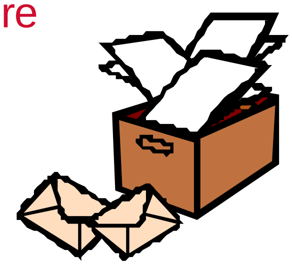
Memory System Architecture

Storage Systems

Erik Riedel
Electrical and Computer Engineering
Carnegie Mellon University
riedel@cs.cmu.edu



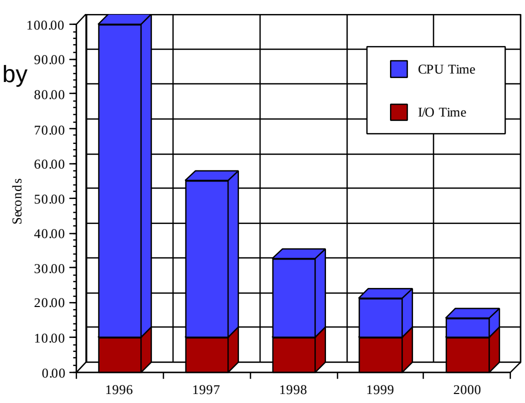
"I/O certainly has been lagging in the last decade."
- Seymour Cray (1976)

"Also, I/O needs a lot of work."
- David Kuck, *15th ISCA* (1988)

Quotes courtesy of Hennessy & Patterson, 2nd Edition

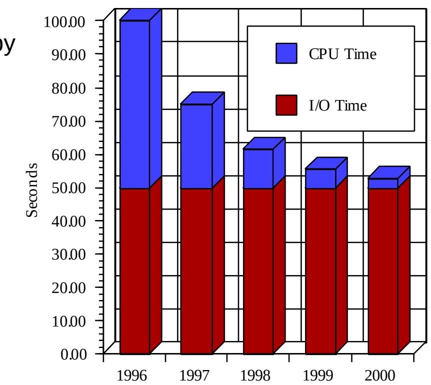
Application Performance

- 1996 - 1997
  - CPU performance improves by
    - N = 400/200 = 2
  - program performance improves by
    - N = 100/55 = 1.81
- 1997 - 1998
  - CPU performance - factor of 2
  - program performance
    - N = 55/32.5 = 1.7
- 1998 - 1999
  - CPU performance -f actor of 2
  - program performance
    - N = 32.5 / 21.25 = 1.53
- 1999 - 2000
  - CPU Performance - factor of 2
  - program performance
    - N = 21.25 / 15.6 = 1.36



Performance for Web Surfing

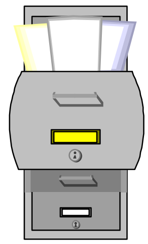
- Assume 50 seconds CPU & 50 seconds I/O
- 1996 - 1997
  - CPU performance improves by
    - N = 400/200 = 2
  - program performance improves by
    - N = 100/75 = 1.33
- 1997 - 1998
  - CPU performance - factor of 2
  - program performance
    - N = 75/62.5= 1.2
- 1998 - 1999
  - CPU performance -f actor of 2
  - program performance
    - N = 62.5/56.5 = 1.11



Who Cares About I/O Anyway?

- Most popular applications in the computer architecture literature are the SPEC benchmark suite
  - lots of scientific code, small working sets, small data sets
- Most popular application in the world is Windows 9x/NT
  - last time I checked, most programs didn't have a lot more data in them, but did have a lot more instructions
- Most widely used computer is an ATM
  - one rarely visits an ATM more than once a day
  - what is the likelihood that your account data will be *cached* at the ATM when you walk up to it?
  - more likely, your account data will be lounging around on some disk drive halfway between here and Minnesota
- Most people don't do CPU-intensive scientific computing
  - Mom doesn't really need to compute finite-element meshes or eigenvalues to make breakfast
  - she does need to program her microwave
  - and find a decent french toast recipe

Magnetic Storage Is Cheaper Than Paper



- File cabinet:

cabinet (four drawer)	$250
paper (24,000 sheets)	$250
space (2x3 @ 10$/ft^2)	$180
total	$700

3¢/sheet

- Disk:

disk (4 GB)	$200
ASCII = 2 million pages	

0.01¢/sheet (300x cheaper)

- Image:

200,000 pages

0.4¢/sheet (8x cheaper)

- Conclusion - Store Everything on Disk

But What Do We Have To Store?

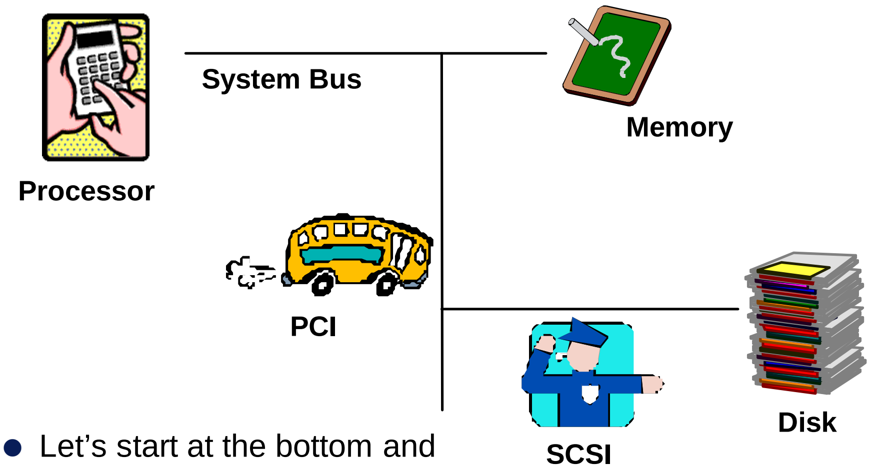
One popular suggestion:

Databases

Information at Your Fingertips™
Information Network™
Knowledge Navigator™

- You might record everything you
  - read - 10 MB/day, 400 GB/lifetime
    - (eight tapes *today*)
  - hear - 400 MB/day, 16 TB/lifetime
    - (three tapes/year *today*)
  - see - 1 MB/s, 40GB/day, 1.6 PB/lifetime
    - (maybe someday)
- All information will be in an online database (somewhere)

System-Level View



- Let's start at the bottom and work our way up...

What's Inside A Disk Drive?

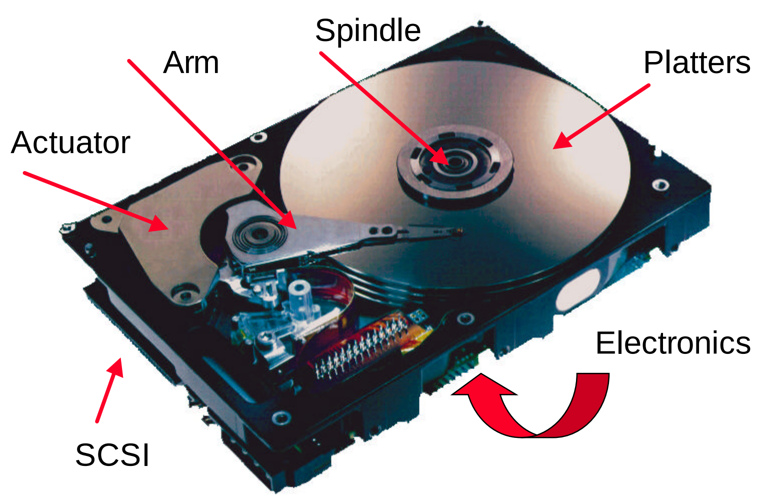
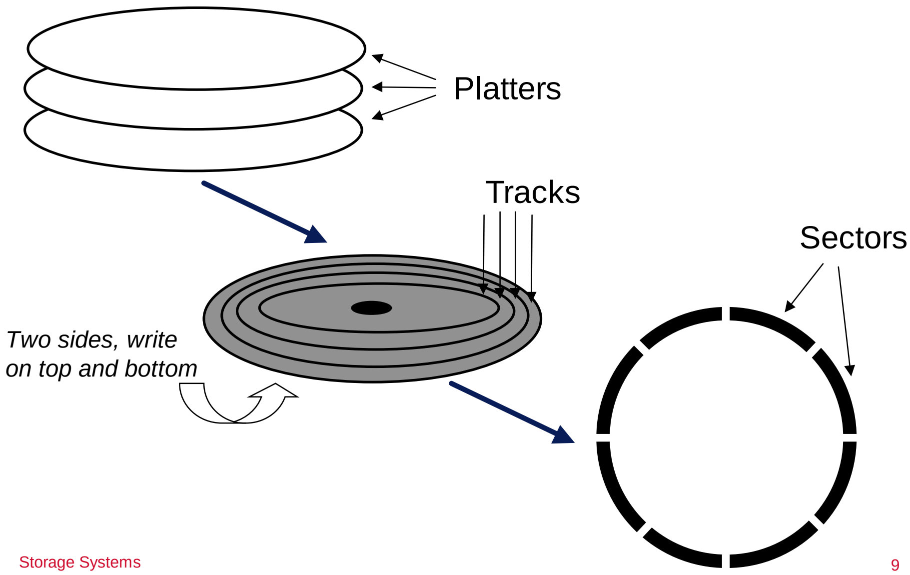


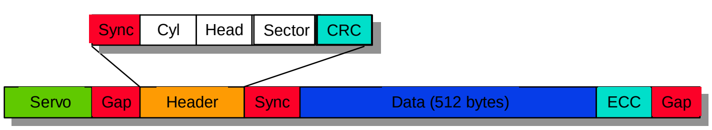
Image courtesy of Seagate Technology Corporation

And If You Look More Closely



And If You Look Even Closer

- Addressable unit is a sector



- Sector breaks down into several different fields
  - Typical size - 512 bytes
  - Typical format
    - `sync` followed by address field (`cyl`, `head`, `sector`, `crc`)
      - `crc` used to verify `cyl`, `head`, `sector` info
    - `gap` followed by the data
    - `ecc` over the data
      - verify data and correct bit errors
    - header, ECC and gaps typically use between 40 and 100 bytes

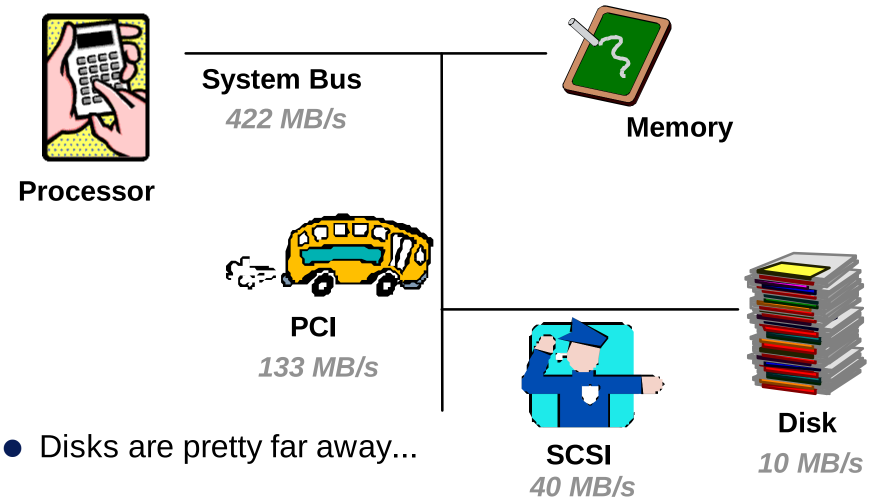
Disk Drive Performance

- Seek time
  - move head to the desired track
  - today's drives - 15 to 5 ms
  - average Seek = (0.33)(distance from outer to inner track)
- Rotational latency
  - 1 / (speed of disk)
  - today's drives - 5,400 to 12,000 RPM
  - average rotational latency = (0.5)(rotational latency)
    - on average, distance to desired sector is 1/2 of a disk rotation
- Transfer time
  - time to transfer a sector
  - today's drives - 20 to 160 MBytes/second
- Controller time
  - overhead on-drive electronics adds to manage drive
  - but also gives prefetching and caching

Disk Drive Performance (con't)

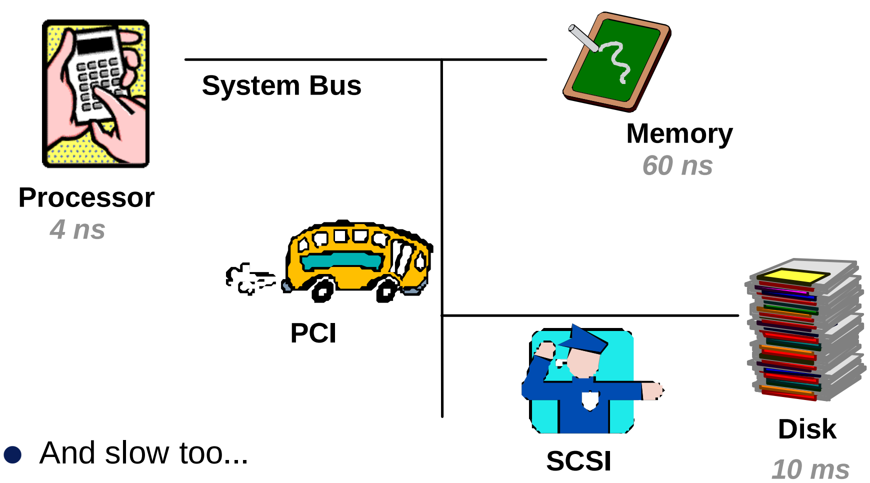
- Average access time =
  - (seek time) + (rotational latency) + (transfer) + (controller time)
- Track and cylinder skew
  - cylinder switch time
    - delay to change from one cylinder to the next
      - may have to wait an extra rotation
    - solution - drives incorporate skew
      - offset sectors between cylinders to account for switch time
  - head switch time
    - change heads to go from one track to next on same cylinder
      - incur additional settling time
- Prefetching
  - disks usually read entire track at a time
  - assuming that request for the next sector will come soon
- Caching
  - limited amount of caching across requests, but prefetching is preferred

System-Level View - Bandwidth



- Disks are pretty far away...

System-Level View - Latency



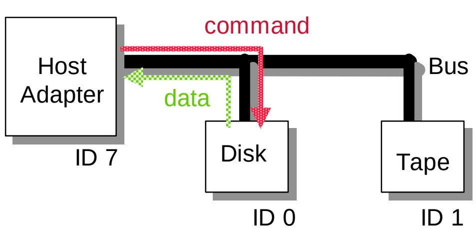
- And slow too...

How Does the CPU Talk to the Drive?

- Basic ways of doing I/O
  - programmed I/O (the old way)
    - CPU directly moves data between memory and storage
  - DMA (direct memory access)
    - CPU tells DMA engine to move data between memory and storage
- Popular drive interfaces
  - IDE
    - low-end, programmed I/O (until recently, now with *UltraDMA)*
  - SCSI (Small Computer Systems Interface)
    - always been DMA, multiple requests outstanding
- Let's focus on SCSI
  - originally developed in 1979 by Al Shugart
    - Shugart Associates => Seagate
    - designed to support *logical* addressing of data
  - standardized by ANSI in 1984, finalized in 1986
  - first product delivered by NCR in 1983

Overview of SCSI

- Device independent I/O bus
  - allows variety of devices to be linked via a single bus
  - defines a set of electrical characteristics and a protocol for the bus
- SCSI devices
  - bus can address up to 8 devices (0..7)
  - devices can either be *initiator* or *target*
    - initiator is the device that begins a transaction
    - target carries out the requested task
    - devices can be both initiator and target (just not at the same time)
- Host adapter
  - connects host system to bus
    - (usually has id 7)

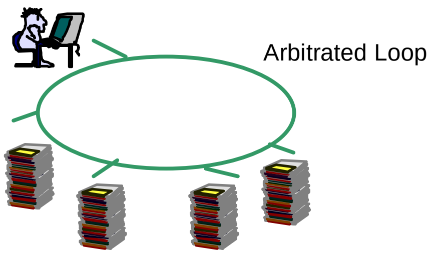


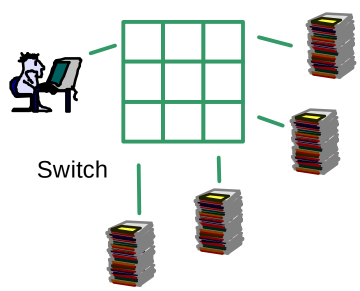
Overview of SCSI (con't)

- Messaging
  - *commands*, *messages* and *status* are sent using asynchronous transfers
    - sender and receiver use request/acknowledge handshake
    - asynchronous transfers relatively slow (lots of overhead)
  - *data* transferred synchronously - enabling maximum bandwidth
    - between 20 and 160 MB/s today
      - depending on how well you play electrical games
      - higher transfer rates typically imply shorter cables
- Flavors of SCSI
  - SCSI (5 MB/s)
  - Fast SCSI (10 MB/s)
  - Wide SCSI (10 or 20 MB/s)
    - 16-bit transfers by adding additional data lines in cable
  - Ultra SCSI (20 MB/s)
  - Single-Ended vs. Differential
    - differential enables longer cable lengths (up to 25 meters)
  - Ultra2, Ultra3, LVD

And, For Our Next Trick

- FibreChannel
  - it's a network, only we've made it fast

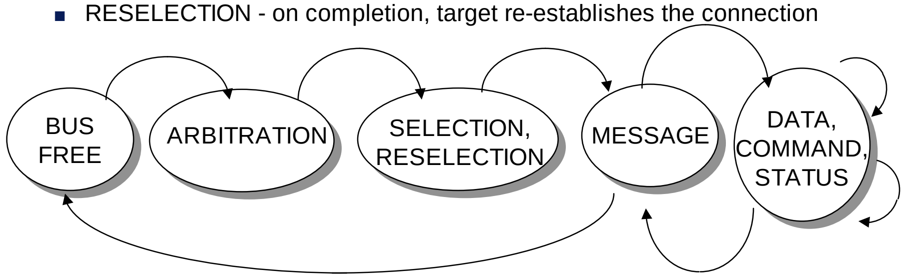




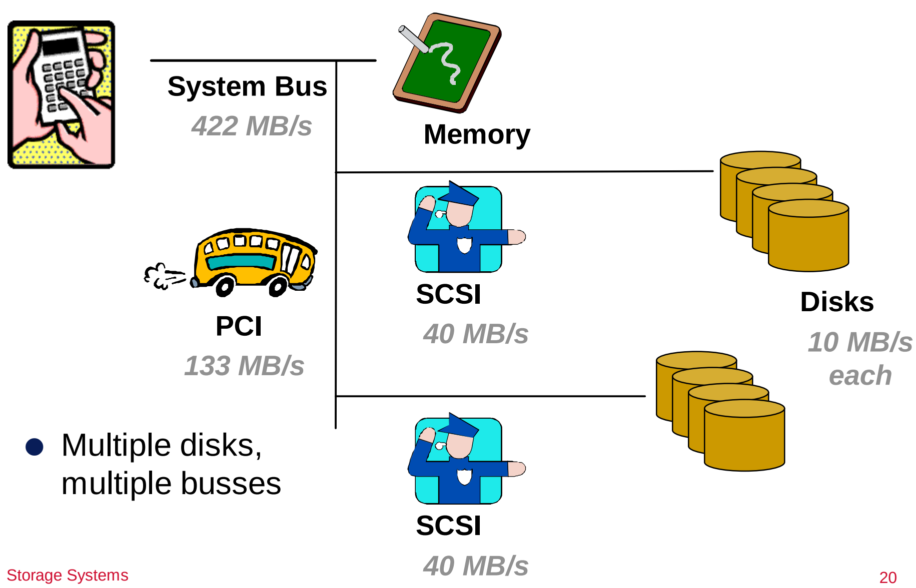
  - eliminates addressing limits
  - provides redundant links
  - enables multiple-host access

SCSI Bus Transactions

- Transactions composed of eight distinct bus phases
  - everything begins and ends with the BUS FREE phase
- Protocol phases
  - ARBITRATION - one or more initiators indicate their wish to use the bus
    - by putting their IDs on the bus
    - if more than one initiator, the one with the largest SCSI ID wins
  - SELECTION - choose a target to communicate with
  - RESELECTION - on completion, target re-establishes the connection



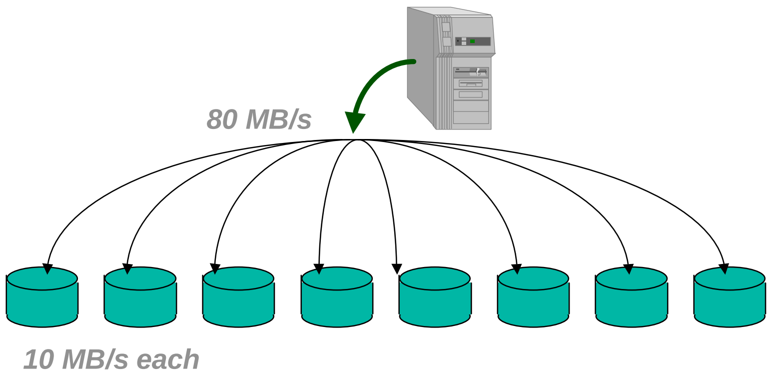
System-Level View - More Bandwidth



- Multiple disks, multiple busses

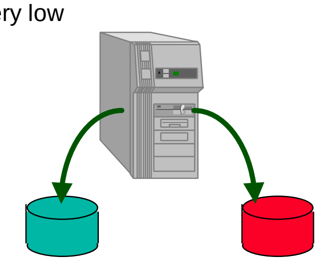
Disk Arrays

- Interleave data across multiple disks
  - striping provides aggregate bandwidth
  - stripe unit depends on application



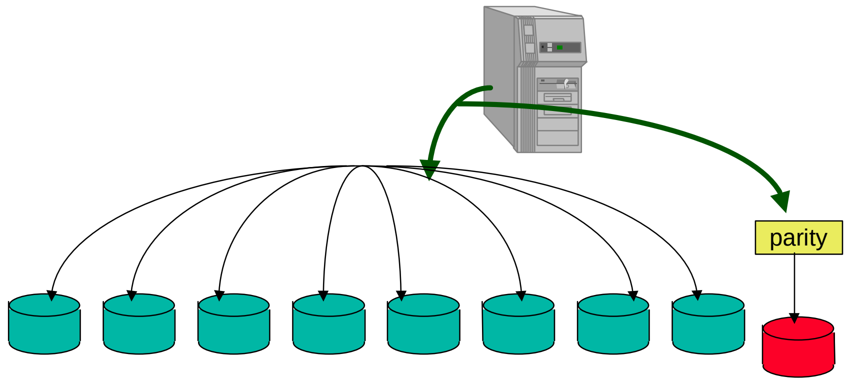
But What If Something Goes Wrong?

- The problem with disks is that if a drive fails, your data is gone (can't "reboot" to solve all problems)
  - backups help this, but backing up takes a long time and effort
  - backup doesn't help recover data lost during that day
  - *any* data loss is a big deal to a bank or stock exchange
- One solution is to mirror every data write onto two drives
  - the probability of two drives failing is very low
  - doubles the cost of storage
  - has a bit of performance benefit too



RAID - Redundant Arrays of Inexpensive Disks

- Write one unit per drive
- Compute the parity and store it on the eight drive
- Cheaper than mirroring
  - reduces overhead to 1/8

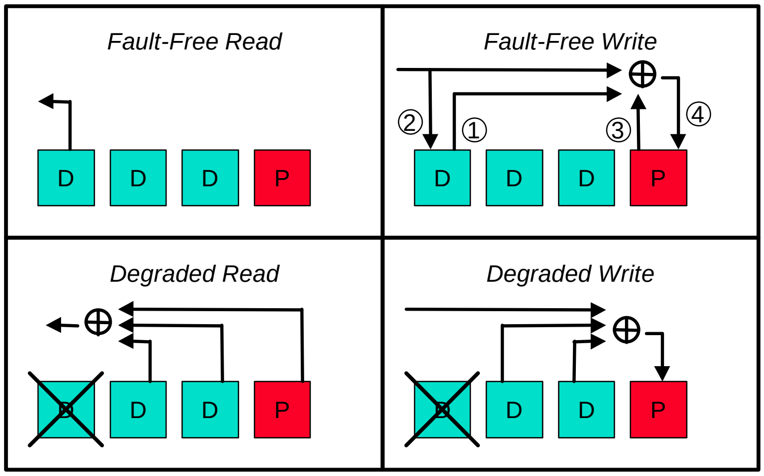


Error Recovery

- Parity
  - count number of 1's in a byte and store a parity bit with each byte of data
  - parity bit is computed as
    - If the number of 1's is even, store a 0
    - If the number of 1's is odd, store a 1
    - This is called even parity (# of ones is even)
  - example

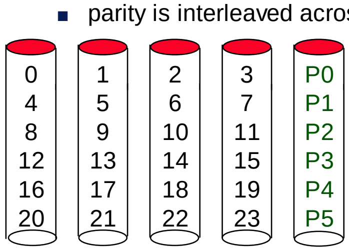
    0x54 == 0101 0100_2
    Three 1's --> odd parity
    Store 9 bits   101 0100 1
  - correct single-bit errors
  - works cheaply because disk failures are *erasures*, not *errors*
- Recovery
  - replace failed disk, reconstruct data using remaining disks and parity
  - if you're smart, can do this without the customer noticing
    - *hot spares* to swap in, replace failed drives during monthly PM

RAID 5 Functions

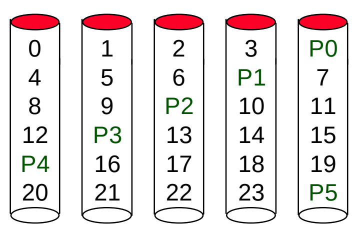


Different Levels of RAID

- RAID 1 - mirroring
  - uses twice as many disks to shadow the data
- RAID 3 - bit interleaved
  - reduces cost to 1/N, where N is the number of disks in a group
- RAID 4 - block interleaved
- RAID 5 - block-interleaved, distributed parity
  - parity is interleaved across disks in the array to balance load



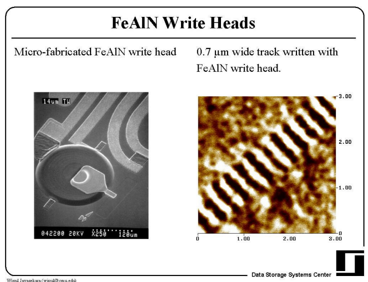
RAID 4 - Block Interleaved Parity



RAID 5 - Distributed Parity

Where Do We Go From Here?

- IBM Microdrive
  - 20 grams
  - 340 MB
  - 15 ms seek
  - 4500 RPM
  - can be powered by AA battery



- MEMS-based Storage
  - micromachines
  - 0.7 *micron* data tracks
  - single chip
    - compute, memory, storage

Images courtesy of International Business Machines Corporation and Carnegie Mellon Data Storage Systems Center

Review

- I/O matters
  - we may be at the bottom of the hierarchy
  - but this is where all the permanent data lives
- Lots of data to store
  - and increasing
  - plus, if that isn't enough, there's always the need to *retrieve* it
- Disks are most popular storage media
  - does caching and block prefetches, just like cache memory
  - interleaves across multiple "banks" just like main memory
  - much bigger, much slower
- Connections to CPUs and memory are a major concern
  - can't just run a few address and data lines
- Fault-tolerance complicates things
  - disks have to hold onto the data, no matter what