Advanced Computer Architecture

Storage

Fall 2016

Pejman Lotfi-Kamran

Adapted from slides originally developed by Profs. Hill, Hoe, Falsafi and Wenisch of CMU, EPFL, Michigan, Wisconsin

Where Are We?

Fr	Sa	Su	Mo	Tu
	27-Shahrivar		29-Shahrivar	
	3-Mehr		5-Mehr	
	10-Mehr		12-Mehr	
	17-Mehr		19-Mehr	
	24-Mehr		26-Mehr	
	1-Aban		3-Aban	
	8-Aban		10-Aban	
	15-Aban		17-Aban	
	22-Aban		24-Aban	
	29-Aban		1-Azar	
	6-Azar		8-Azar	
	13-Azar		15-Azar	
	20-Azar		22-Azar	
	27-Azar		29-Azar	
	4-Dey		6-Dey	

- This Lecture
  - Storage
- Next Lecture:
  - Scaling

I/O Introduction: Storage Devices & RAID

Jason Hill

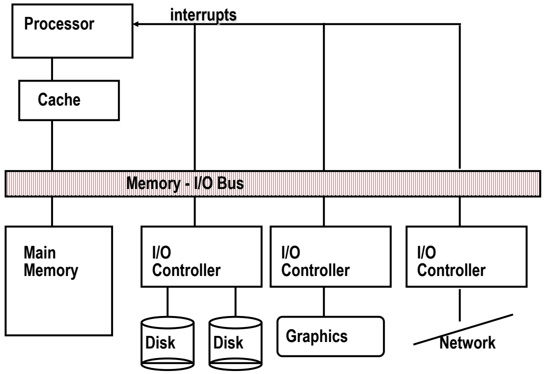
Motivation: Who Cares About I/O?

- CPU Performance: 60% per year
- I/O system performance limited by *mechanical* delays (disk I/O)
  < 10% per year (IO per sec)
- Amdahl's Law: system speed-up limited by the slowest part!
  10% IO & 10x CPU => 5x Performance (lose 50%)
  10% IO & 100x CPU => 10x Performance (lose 90%)
- I/O bottleneck:
  Diminishing fraction of time in CPU
  Diminishing value of faster CPUs

Big Picture: Who cares about CPUs?

- Why still important to keep CPUs busy vs. IO devices ("CPU time"), as CPUs not costly?
  - Moore's Law leads to both large, fast CPUs but also to very small, cheap CPUs
  - 2001 Hypothesis: 600 MHz PC is fast enough for Office Tools?
  - PC slowdown since fast enough unless games, new apps?
- People care more about storing information and communicating information than calculating
  - "Information Technology" vs. "Computer Science"
  - 1960s and 1980s: Computing Revolution
  - 1990s and 2000s: Information Age

I/O Systems



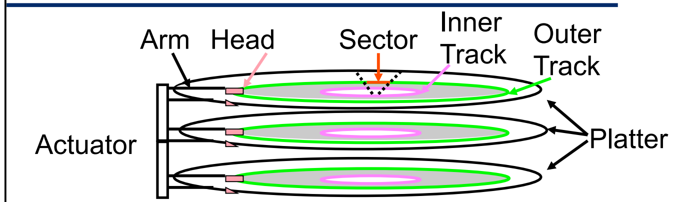
Storage Technology Drivers

- Driven by the prevailing computing paradigm
  - 1950s: migration from batch to on-line processing
  - 1990s: migration to ubiquitous computing
    - computers in phones, books, cars, video cameras, …
    - nationwide fiber optical network with wireless tails
- Effects on storage industry:
  - Embedded storage
    - smaller, cheaper, more reliable, lower power
  - Data utilities
    - high capacity, hierarchically managed storage

Outline

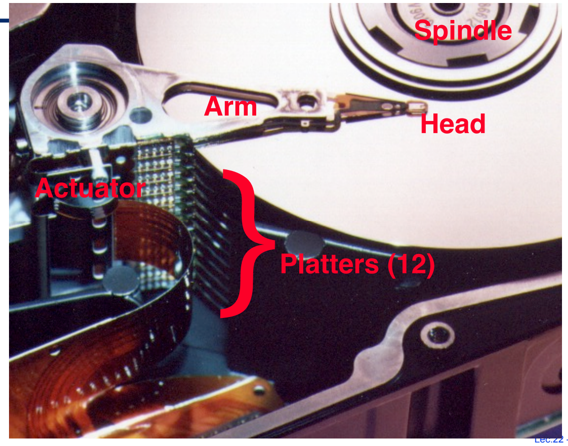
- Disk Basics
- Disk History
- Disk options in 2000
- Disk fallacies and performance
- FLASH
- Tapes
- RAID

Disk Device Terminology

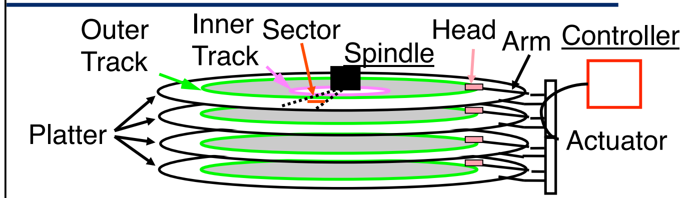


- ◆ Several <u>platters</u>, with data recorded magnetically on both <u>surfaces</u> (usually)
- Bits recorded in <u>tracks</u>, which in turn divided into <u>sectors</u> (e.g., 512 Bytes)
- <u>Actuator</u> moves <u>head</u> (end of <u>arm</u>,1/surface) over track (“<u>seek</u>”), select <u>surface</u>, wait for <u>sector</u> rotate under <u>head</u>, then read or write
  - “<u>Cylinder</u>”: all tracks under heads

Photo of Disk Head, Arm, Actuator



Disk Device Performance



Disk Latency = Seek Time + Rotation Time + Transfer Time + Controller Overhead

- ◆ Seek Time? depends no. tracks move arm, seek speed of disk
- ◆ Rotation Time? depends on speed disk rotates, how far sector is from head
- ◆ Transfer Time? depends on data rate (bandwidth) of disk (bit density), size of request

Disk Device Performance

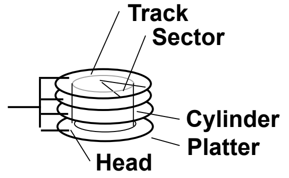
- ◆ Average distance sector from head?
- ◆ 1/2 time of a rotation
  - 10000 Revolutions Per Minute ⇒ 166.67 Rev/sec
  - 1 revolution = 1/ 166.67 sec ⇒ 6.00 milliseconds
  - 1/2 rotation (revolution) ⇒ 3.00 ms
- ◆ Average no. tracks move arm?
  - Sum all possible seek distances from all possible tracks / # possible
    - ▲Assumes average seek distance is random
  - Disk industry standard benchmark

Data Rate: Inner vs. Outer Tracks

- To keep things simple, originally kept same number of sectors per track
  - Since outer track longer, lower bits per inch
- Competition ⇒ decided to keep BPI the same for all tracks (“constant bit density”)
  - ⇒ More capacity per disk
  - ⇒ More of sectors per track towards edge
  - ⇒ Since disk spins at constant speed, outer tracks have faster data rate
- Bandwidth outer track 1.7X inner track!
  - Inner track highest density, outer track lowest, so not really constant
  - 2.1X length of track outer / inner, 1.7X bits outer / inner

Devices: Magnetic Disks

- Purpose:
  - Long-term, nonvolatile storage
  - Large, inexpensive, slow level in the storage hierarchy
- Characteristics:
  - Seek Time (~8 ms avg)
    - ▲positional latency
    - ▲rotational latency
- Transfer rate
  - 10-40 MByte/sec
  - Blocks
- Capacity
  - Terabytes
  - Quadruples every 2 years



7200 RPM = 120 RPS => 8 ms per rev
ave rot. latency = 4 ms
128 sectors per track => 0.25 ms per sector
1 KB per sector => 16 MB / s

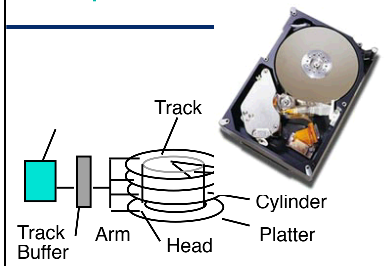
Response time
= Queue + Controller + Seek + Rot + Xfer

Service time

Disk Performance Model /Trends

- Capacity
  + 100%/year (2X / 1.0 yr)
- Transfer rate (BW)
  + 40%/year (2X / 2.0 yrs)
- Rotation + Seek time
  – 8%/ year (1/2 in 10 yrs)
- Capacity/$
  > 100%/year (2X / 1.0 yr)
  Fewer chips + areal density

Example: Barracuda 180 c.a. 2000



Latency =
per access + *per byte* {
Queuing Time +
Controller time +
Seek Time +
Rotation Time +
Size / Bandwidth

- 181.6 GB, 3.5 inch disk
- 12 platters, 24 surfaces
- 24,247 cylinders
- 7,200 RPM; (4.2 ms avg. latency)
- 7.4/8.2 ms avg. seek (r/w)
- 64 to 35 MB/s (internal)
- 0.1 ms controller time
- 10.3 watts (idle)

source: www.seagate.com

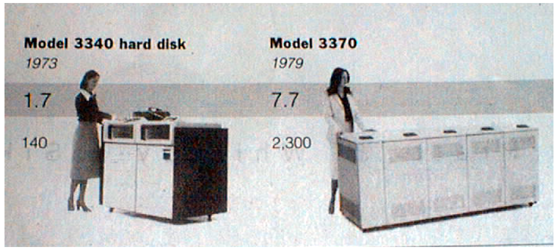
Historical Perspective

- 1956 IBM Ramac — early 1970s Winchester
  - Developed for mainframe computers, proprietary interfaces
  - Steady shrink in form factor: 27 in. to 14 in
- Form factor and capacity drives market, more than performance
- 1970s: Mainframes ⇒ 14 inch diameter disks
- 1980s: Minicomputers,Servers ⇒ 8”,5 1/4” diameter
- PCs, workstations Late 1980s/Early 1990s:
  - Mass market disk drives become a reality
    - industry standards: SCSI, IPI, IDE
  - Pizzabox 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
- 2000s:
  - 1 inch for cameras, cell phones?

Disk History

Data density Mbit/sq. in.

Capacity of Unit Shown Megabytes

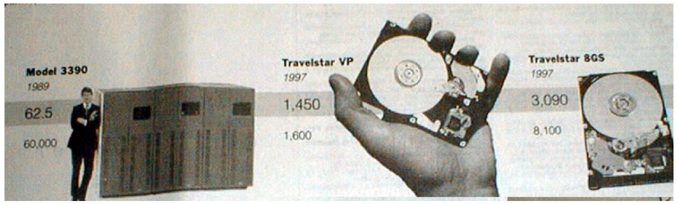


1973:
1. 7 Mbit/sq. in
140 MBytes

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

source: New York Times

Disk History



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

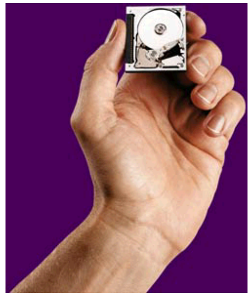
1997:
1450 Mbit/sq. in
2300 MBytes

1997:
3090 Mbit/sq. in
8100 MBytes

source: New York Times

1 inch disk drive!

- 2000 IBM MicroDrive:
  - 1.7” x 1.4” x 0.2”
  - 1 GB, 3600 RPM, 5 MB/s, 15 ms seek
  - Digital camera, PalmPC?
- 9 GB, 50 MB/s!
  - Assuming it finds a niche in a successful product
  - Assuming past trends continue



Disk Characteristics in 2000

	Seagate Cheetah ST173404LC Ultra160 SCSI	IBM Travelstar 32GH DJSA - 232 ATA-4	IBM 1GB Microdrive DSCM-11000
Disk diameter (inches)	3.5	2.5	1.0
Formatted data capacity (GB)	73.4	32.0	1.0
Cylinders	14,100	21,664	7,167
Disks	12	4	1
Recording Surfaces (Heads)	24	8	2
Bytes per sector	512 to 4096	512	512
Avg Sectors per track (512 byte)	~ 424	~ 360	~ 140
Max. areal density(Gbit/sq.in.)	6.0	14.0	15.2
	$828	**$447**	**$435**

Fallacy: Use Data Sheet "Average Seek" Time

- Manufacturers needed standard for fair comparison ("benchmark")
  - Calculate seeks from all tracks, divide by # of seeks => "average"
- Real average would be based on how data laid out on disk, where seek in real applications, then measure performance
  - Usually, tend to seek to tracks nearby, not to random track
- Rule of Thumb: observed average seek time ~ 1/4 to 1/3 of quoted seek time (i.e., 3X-4X faster)
  - Barracuda 180 X avg. seek: 7.4 ms ⇒ 2.5 ms

Fallacy: Use Data Sheet Transfer Rate

- Manufacturers quote the speed off the data rate off the surface of the disk
- Sectors contain an error detection and correction field (can be 20% of sector size) plus sector number as well as data
- There are gaps between sectors on track
- Rule of Thumb: disks deliver about 3/4 of internal media rate (1.3X slower) for data
- For example, Barracuda 180X quotes 64 to 35 MB/sec internal media rate

⇒ 47 to 26 MB/sec external data rate (74%)

Disk Performance Example

Calculate time to read 64 KB for UltraStar 72 again, this time using 1/3 quoted seek time, 3/4 of internal outer track bandwidth; (12.7 ms before)

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

= (0.33 * 7.4 ms) + 0.5 * 1/(7200 RPM) + 64 KB / (0.75 * 65 MB/s) + 0.1 ms

= 2.5 ms + 0.5 /(7200 RPM/(60000ms/M)) + 64 KB / (47 KB/ms) + 0.1 ms

= 2.5 + 4.2 + 1.4 + 0.1 ms = 8.2 ms (64% of 12.7)

Future Disk Size and Performance

- Improvements in capacity (60%/yr) and bandwidth (40%/yr)
- Slow improvement in seek, rotation (8%/yr)
- Time to read whole disk

Year	Sequentially	Randomly (1 sector/seek)
1990	4 minutes	6 hours
2000	12 minutes	1 week(!)

- 3.5” form factor make sense in 5 yrs?
  - What is capacity, bandwidth, seek time, RPM?
  - Assume today 80 GB, 30 MB/sec, 6 ms, 10000 RPM

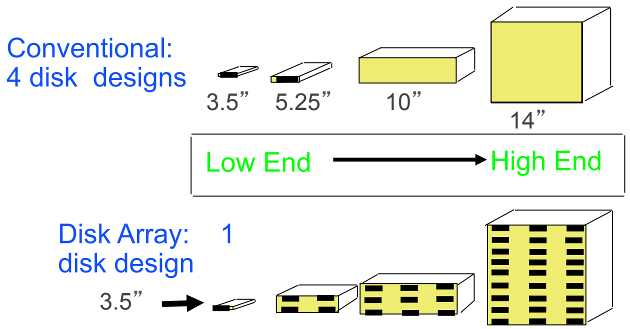
What about FLASH

- Compact Flash Cards
  - Intel Strata Flash
    - 16 Mb in 1 square cm. (.6 mm thick)
  - 100,000 write/erase cycles.
  - Standby current = 100uA, write = 45mA
  - Compact Flash 256MB~=$120 512MB~=$542
  - Transfer @ 3.5MB/s
- IBM Microdrive 1G~370
  - Standby current = 20mA, write = 250mA
  - Efficiency advertised in watts/MB
- VS. Disks
  - Nearly instant standby wake-up time
  - Random access to data stored
  - Tolerant to shock and vibration (1000G of operating shock)

Use Arrays of Small Disks?

Katz and Patterson asked in 1987:

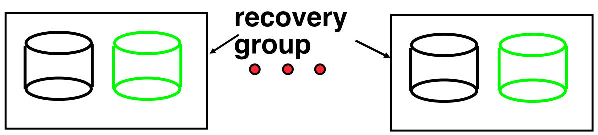
Can smaller disks be used to close gap in performance between disks and CPUs?



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 parts 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

Redundant Arrays of Inexpensive Disks
RAID 1: Disk Mirroring/Shadowing

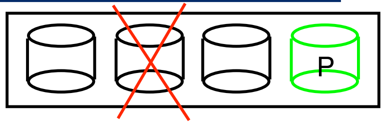


- Each disk is fully duplicated onto its "mirror"
  Very high availability can be achieved
- Bandwidth sacrifice on write:
  Logical write = two physical writes
  - Reads may be optimized
- Most expensive solution: 100% capacity overhead

Redundant Array of Inexpensive Disks
RAID 2 & 3: Parity Disk

10010011
11001101
10010011
. . .

logical record



			P
1	1	1	1
0	1	0	1
1	0	1	0
0	0	0	0
0	1	0	1
0	1	0	1
1	0	1	0
1	1	1	1

Striped physical records

P contains sum of other disks per stripe mod 2 ("parity")
If disk fails, subtract P from sum of other disks to find missing information

RAID 2 & RAID 3

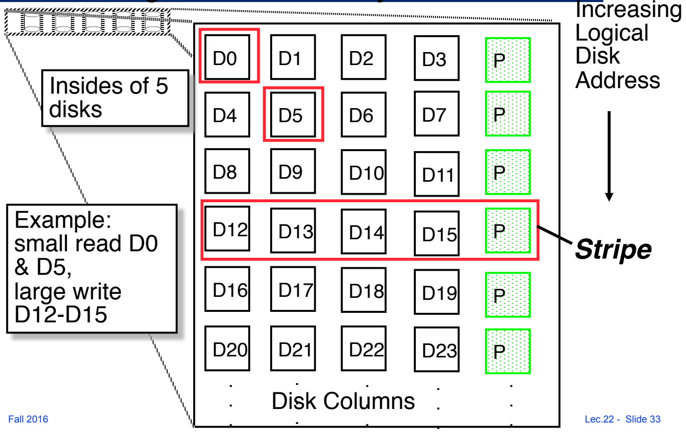
RAID 2 (bit-level) RAID 3 (byte-level) striping

- Sum computed across recovery group to protect against hard disk failures, stored in P disk
- Logically, a single high capacity, high transfer rate disk: good for large transfers
- Wider arrays reduce capacity costs, but decreases availability
- 33% capacity cost for parity in this configuration

Inspiration for RAID 4

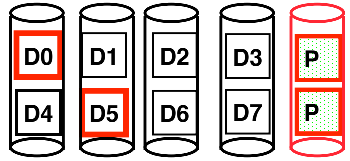
- RAID 3 relies on parity disk to discover errors on Read
- But every sector has an error detection field
- Rely on error detection field to catch errors on read, not on the parity disk
- Allows independent reads to different disks simultaneously
- Uses block-level striping (dedicated parity disk)

Redundant Arrays of Inexpensive Disks RAID 4: High I/O Rate Parity

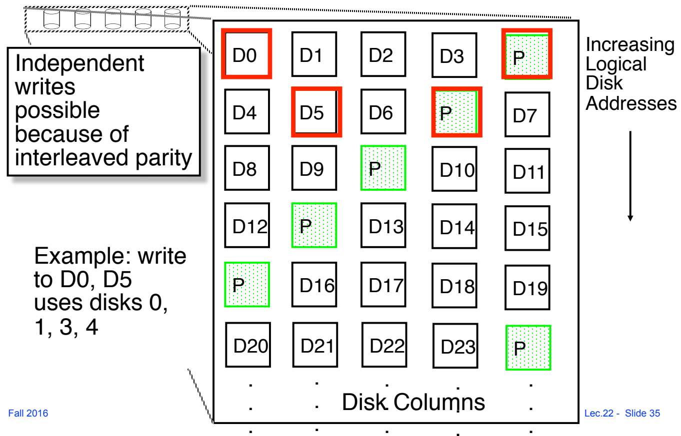


Inspiration for RAID 5

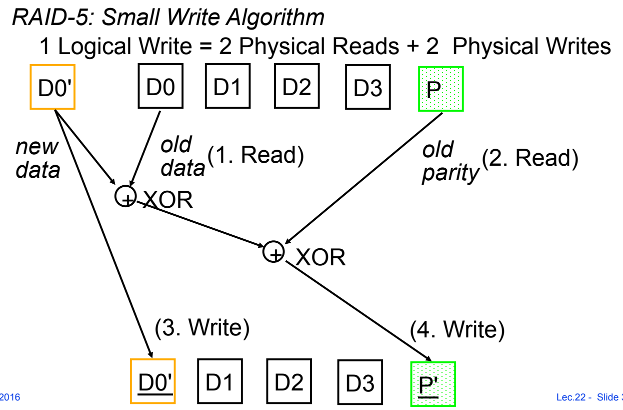
- RAID 4 works well for small reads
- Small writes (write to one disk):
  - Option 1: read other data disks, create new sum and write to Parity Disk
  - Option 2: since P has old sum, compare old data to new data, add the difference to P
- Small writes are limited by Parity Disk: Write to D0, D5 both also write to P disk



Redundant Arrays of Inexpensive Disks RAID 5: High I/O Rate Interleaved Parity



Problems of Disk Arrays: Small Writes

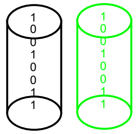


Berkeley History: RAID-I

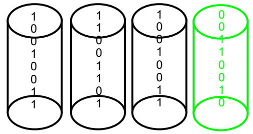
- RAID-I (1989)
  - Consisted of a Sun 4/280 workstation with 128 MB of DRAM, four dual-string SCSI controllers, 28 5.25-inch SCSI disks and specialized disk striping software
- Today RAID is $19 billion dollar industry, 80% nonPC disks sold in RAIDs

Summary: RAID Techniques: Goal was performance, popularity due to reliability of storage

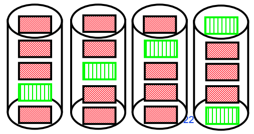
- Disk Mirroring, Shadowing (RAID 1)
  - Each disk is fully duplicated onto its "shadow"
  - Logical write = two physical writes
  - 100% capacity overhead



- Parity Data Bandwidth Array (RAID 3)
  - Parity computed horizontally
  - Logically a single high data bw disk



- High I/O Rate Parity Array (RAID 5)
  - Interleaved parity blocks
  - Independent reads and writes
  - Logical write = 2 reads + 2 writes



Summary Storage

- Disks:
  - Extraodinary advance in capacity/drive, $/GB
  - Currently 17 Gbit/sq. in. ; can continue past 100 Gbit/sq. in.?
  - Bandwidth, seek time not keeping up: 3.5 inch form factor makes sense? 2.5 inch form factor in near future? 1.0 inch form factor in long term?
- Tapes
  - No investment, must be backwards compatible
  - Are they already dead?
  - What is a tapeless backup system?

The following slides are from Shimin Chen of Intel.

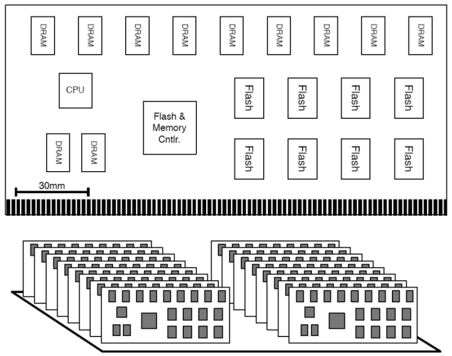
Introduction

- Gordon: a flash-based system architecture for massively parallel, data-centric computing.
  - Solid-state disks
  - Low-power processors
  - Data-centric programming paradigms
- Can deliver:
  - Up to 2.5X the computation per energy of a conventional cluster based system
  - Increasing performance by up to 1.5X

Outline

- Gordon's system architecture
- Gordon's storage system
- Configuring Gordon
- Discussion
- Summary

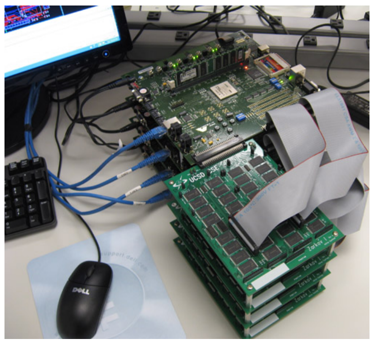
Gordon's System Architecture



(a) Gordon node

(b) 16 nodes in an enclosure

Prototype Photo:



ASPLOS' 09 paper uses simulation

http://www-cse.ucsd.edu/users/swanson/projects/gordon.html

Gordon node

- Configuration:
  - 256GB of flash storage
  - A flash storage controller (w/ 512MB dedicated DRAM)
  - 2GB ECC DDR2 SDRAM
  - 1.9Ghz Intel Atom processor
  - Running a minimal linux installation
- Power: no more than 19w
  - Compared to 81w of a server
- 900MB/s read and write bandwidth to 256GB disk

Enclosures

- Within an enclosure, 16 nodes plug into a backplane that provides 1Gb Ethernet-style network
- A rack holds 16 enclosures (16x16=256 nodes)
  - 64 TB of storage
  - 230 GB/s of aggregate IO bandwidth

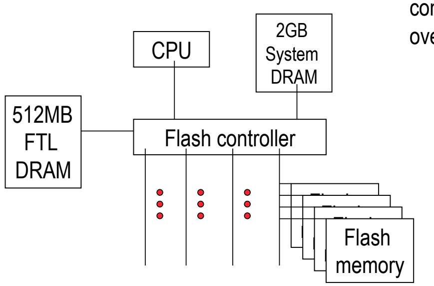
Programming

- From the SW and users' perspectives, a Gordon system appears to be a conventional computing cluster
- Benchmarks: Hadoop

Outline

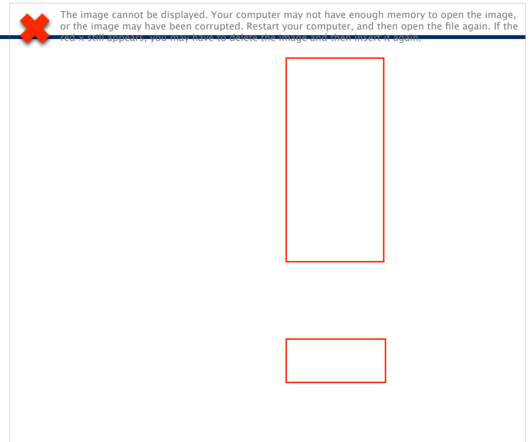
- Gordon's system architecture
- Gordon's storage system
- Configuring Gordon
- Discussion
- Summary

Flash Array Hardware



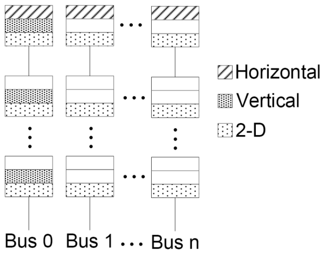
The flash controller has over 300 pins

4 buses, each support 4 flash packages



Super-Page

- Three ways to stripe data across flash memory
- Horizontal: across buses
- Vertical: across packages on the same bus
- 2-D: combined



Bypassing and Write Combining

- Read bypassing: merging read requests to the same page
- Write combining: merging write requests to the same page

Summary

- Use flash memory + low-power processor (Atom)
- Support data intensive computing: such as Map-Reduce operations
- The design choice is attractive because of higher power/performance efficiency