COS 318: Operating Systems

Storage Devices

Andy Bavier
Computer Science Department
Princeton University

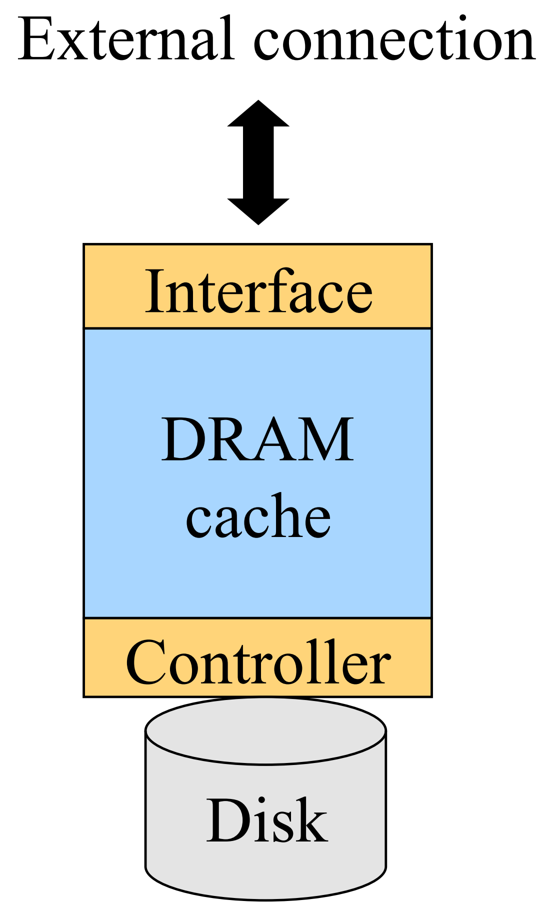
http://www.cs.princeton.edu/courses/archive/fall10/cos318/

Today's Topics

- Magnetic disks
- Magnetic disk performance
- Disk arrays
- Flash memory

A Typical Magnetic Disk Controller

- External connection
  - IDE/ATA, SATA
  - SCSI, SCSI-2, Ultra SCSI, Ultra-160 SCSI, Ultra-320 SCSI
  - Fibre channel
- Cache
  - Buffer data between disk and interface
- Controller
  - Read/write operation
  - Cache replacement
  - Failure detection and recovery

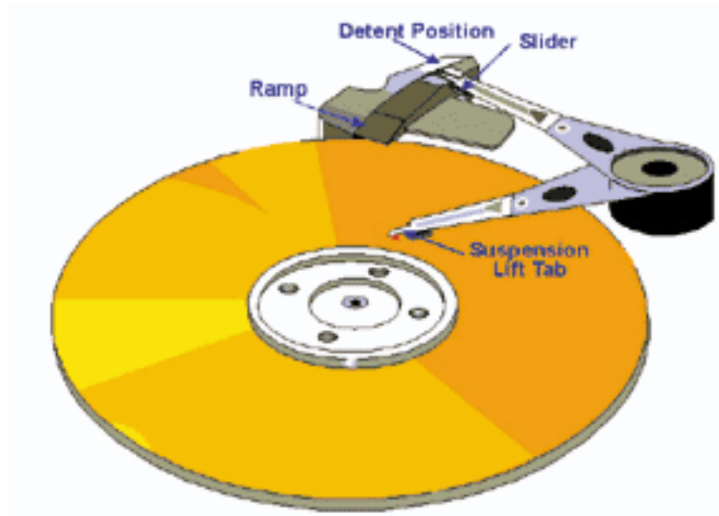


Disk Caching

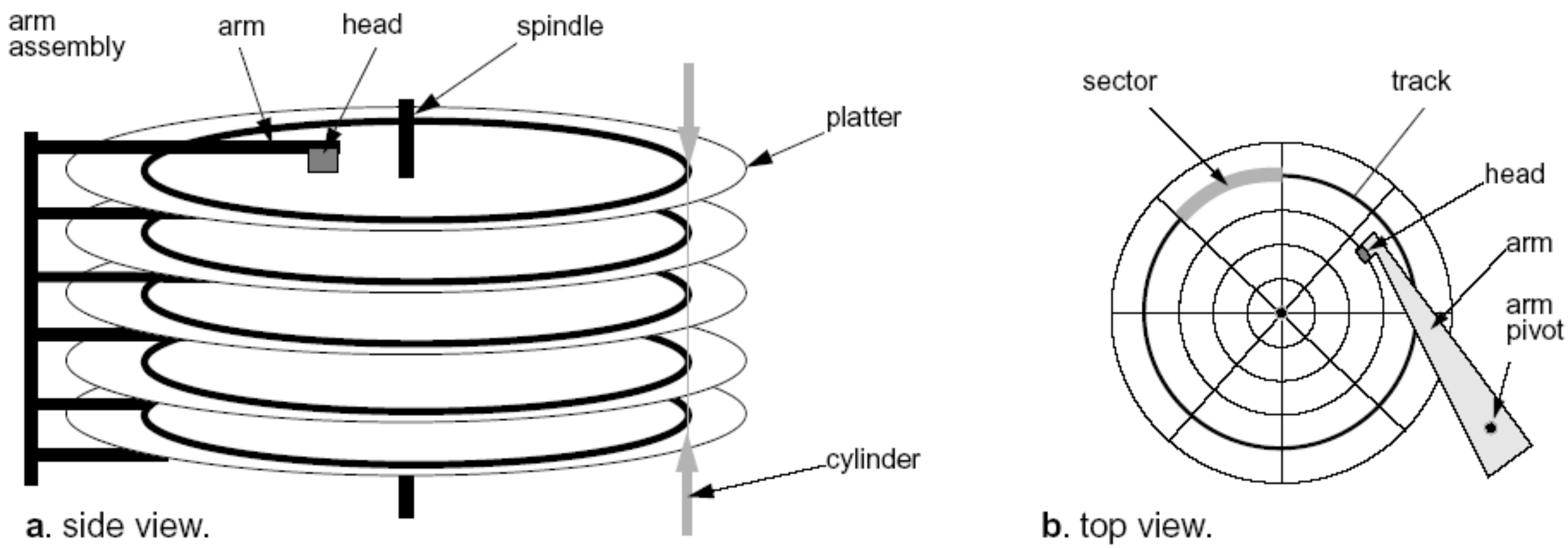
- Method
  - Use DRAM to cache recently accessed blocks
    - Most disk has 16MB
    - Some of the RAM space stores “firmware” (an embedded OS)
  - Blocks are replaced usually in an LRU order
- Pros
  - Good for reads if accesses have locality
- Cons
  - Need to deal with reliable writes

Disk Arm and Head

- Disk arm
  - A disk arm carries disk heads
- Disk head
  - Mounted on an actuator
  - Read and write on disk surface
- Read/write operation
  - Disk controller receives a command with <track#, sector#>
  - Seek the right cylinder (tracks)
  - Wait until the right sector comes
  - Perform read/write



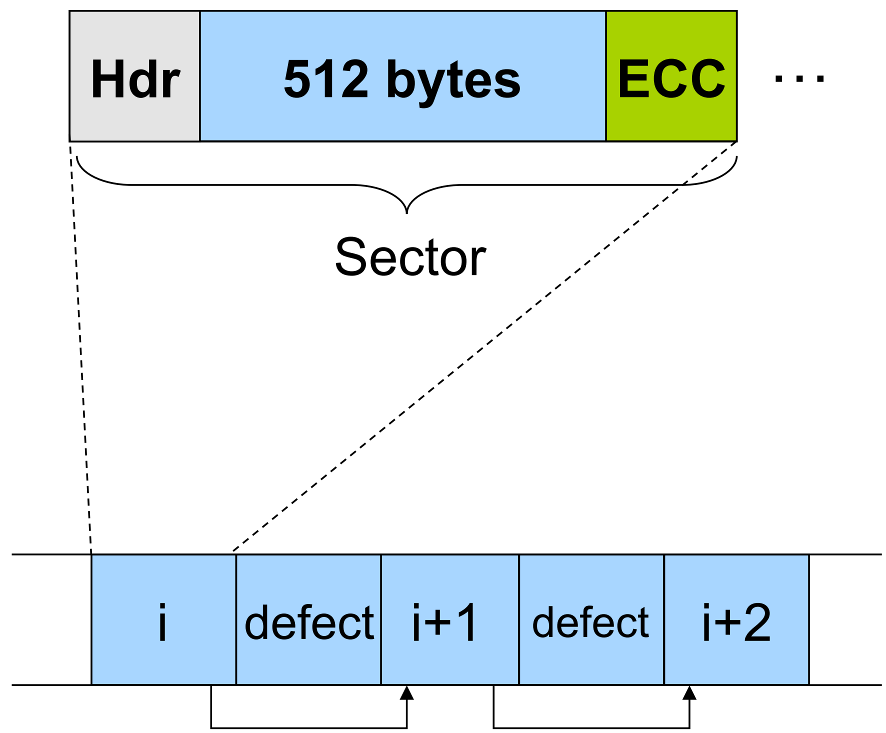
Mechanical Component of A Disk Drive



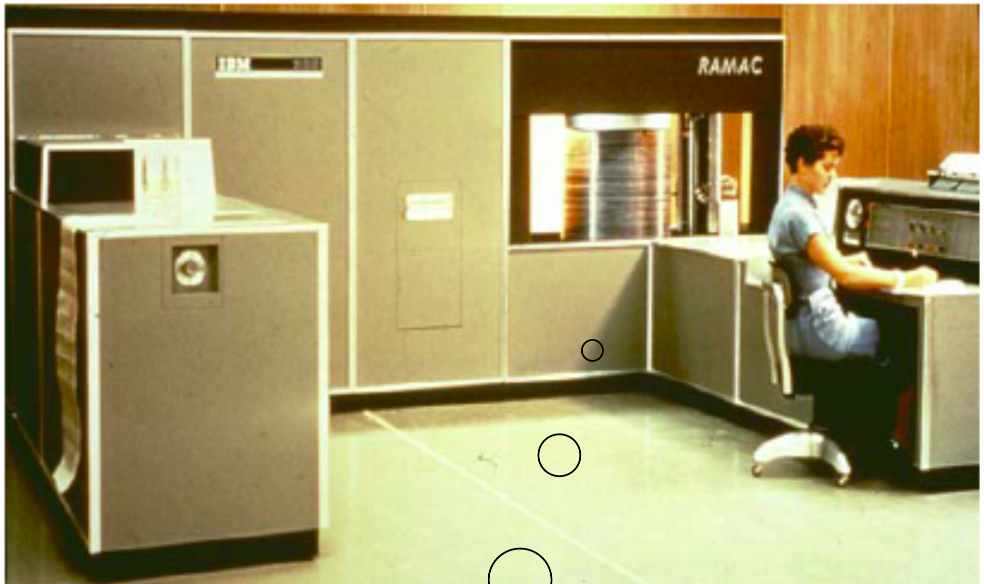
- Tracks
  - Concentric rings around disk surface, bits laid out serially along each track
- Cylinder
  - A track of the platter, 1000-5000 cylinders per zone, 1 spare per zone
- Sectors
  - Each track is split into arc of track (min unit of transfer)

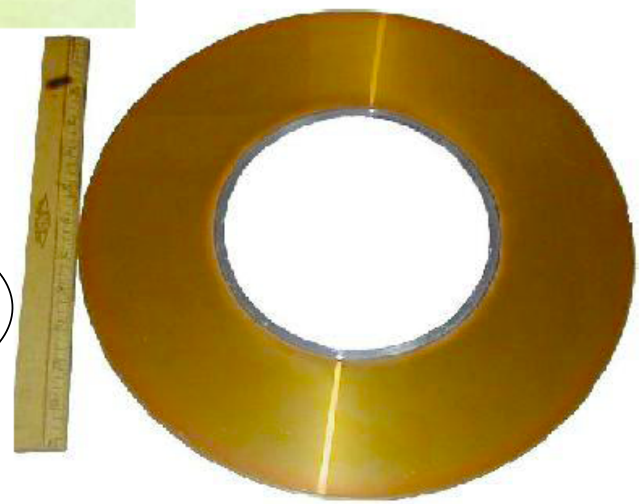
Disk Sectors

- Where do they come from?
  - Formatting process
  - Logical maps to physical
- What is a sector?
  - Header (ID, defect flag, …)
  - Real space (e.g. 512 bytes)
  - Trailer (ECC code)
- What about errors?
  - Detect errors in a sector
  - Correct them with ECC
  - If not recoverable, replace it with a spare
  - Skip bad sectors in the future



Disks Were Large





First Disk:
IBM 305 RAMAC (1956)
5MB capacity
50 disks, each 24”

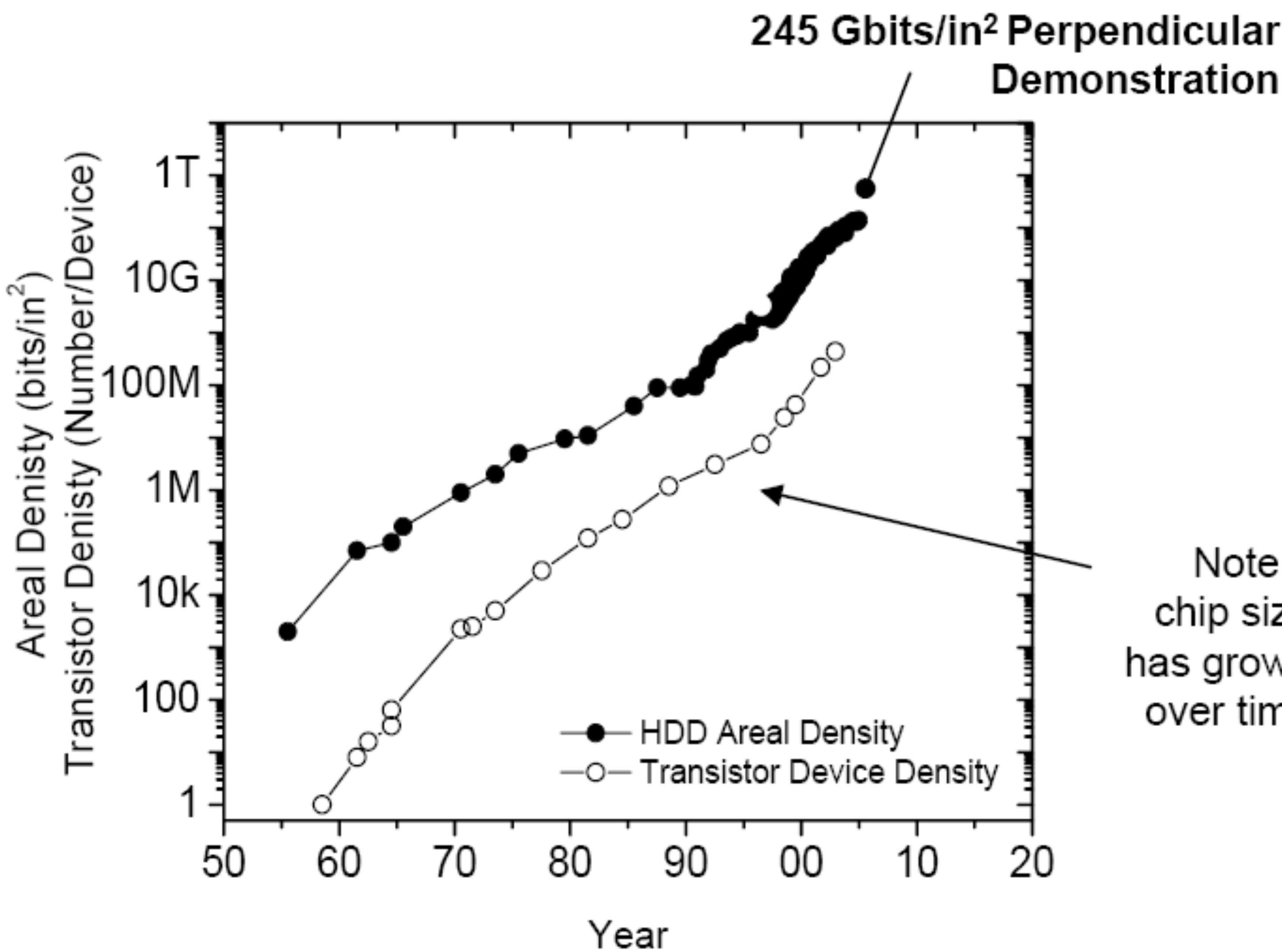
They Are Now Much Smaller

Form factor:
.5-1"× 4"× 5.7"
Storage:
0.5-2TB

Form factor:
.4-.7" × 2.7" × 3.9"
Storage:
160-320GB

Form factor:
.2-.4" × 2.1" × 3.4"
Storage:
1GB-8GB
Replaced by Flash

Areal Density vs. Moore's Law



Note – chip size has grown over time

(Mark Kryder at SNW 2006)

50 Years (Mark Kryder at SNW 2006)

	IBM RAMAC (1956)	Seagate Momentus (2006)	Difference
Capacity	5MB	160GB	32,000
Areal Density	2K bits/in^2	130 Gbits/in^2	65,000,000
Disks	50 @ 24” diameter	2 @ 2.5” diameter	1 / 2,300
Price/MB	$1,000	$0.01	1 / 100,000
Spindle Speed	1,200 RPM	5,400 RPM	5
Seek Time	600 ms	10 ms	1 / 60
Data Rate	10 KB/s	44 MB/s	4,400
Power	5000 W	2 W	1 / 2,500
Weight	~ 1 ton	4 oz	1 / 9,000

Sample Disk Specs (from Seagate)

	Cheetah 15k.7	Barracuda XT
Capacity		
Formatted capacity (GB)	600	2000
Discs	4	4
Heads	8	8
Sector size (bytes)	512	512
Performance		
External interface	Ultra320 SCSI, FC, S. SCSI	SATA
Spindle speed (RPM)	15,000	7,200
Average latency (msec)	2.0	4.16
Seek time, read/write (ms)	3.5/3.9	8.5/9.5
Track-to-track read/write (ms)	0.2-0.4	0.8/1.0
Internal transfer (MB/sec)	1,450-2,370	600
Transfer rate (MB/sec)	122-204	138
Cache size (MB)	16	64
Reliability		
Recoverable read errors	1 per 10^{12} bits read	1 per 10^{10} bits read
Non-recoverable read errors	1 per 10^{16} bits read	1 per 10^{14} bits read

Disk Performance

- Seek
  - Position heads over cylinder, typically 3.5-9.5 ms
- Rotational delay
  - Wait for a sector to rotate underneath the heads
  - Typically 8 - 4 ms (7,200 – 15,000RPM)
    or ½ rotation takes 4 - 2ms
- Transfer bytes
  - Transfer bandwidth is typically 40-138 Mbytes/sec
- Performance of transfer 1 Kbytes
  - Seek (4 ms) + half rotational delay (2ms) + transfer (0.013 ms)
  - Total time is 6.01 ms or 167 Kbytes/sec! (1/360 of 60MB/s!)

More on Performance

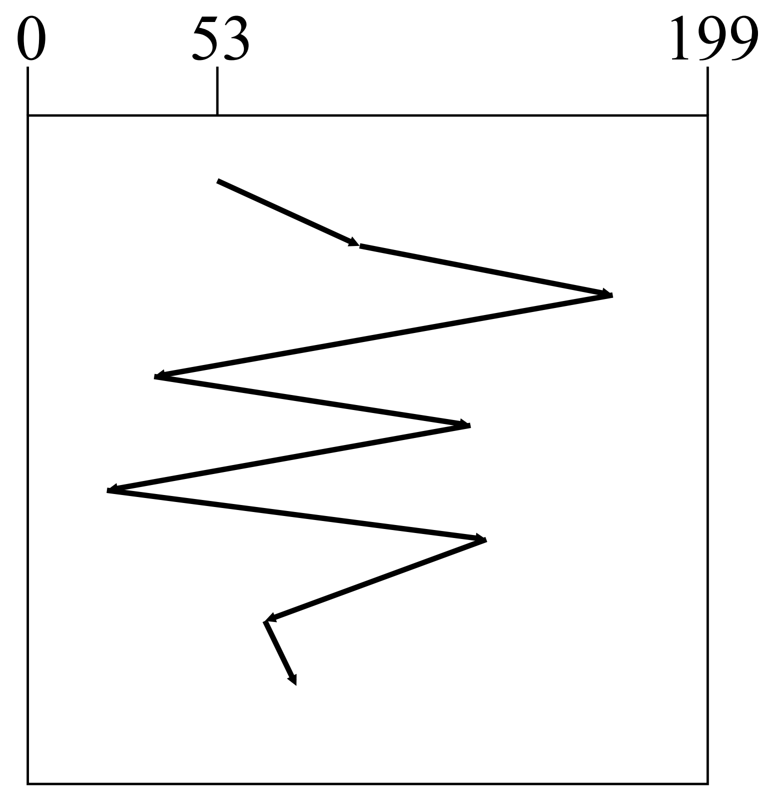
- What transfer size can get 90% of the disk bandwidth?
  - Assume Disk BW = 60MB/sec, ½ rotation = 2ms, ½ seek = 4ms
  - BW * 90% = size / (size/BW + rotation + seek)
  - size = BW * (rotation + seek) * 0.9 / 0.1
    = 60MB * 0.006 * 0.9 / 0.1 = 3.24MB

Block Size (Kbytes)	% of Disk Transfer Bandwidth
1Kbytes	0.28%
1Mbytes	73.99%
3.24Mbytes	90%

- Seek and rotational times dominate the cost of small accesses
  - Disk transfer bandwidth are wasted
  - Need algorithms to reduce seek time

FIFO (FCFS) order

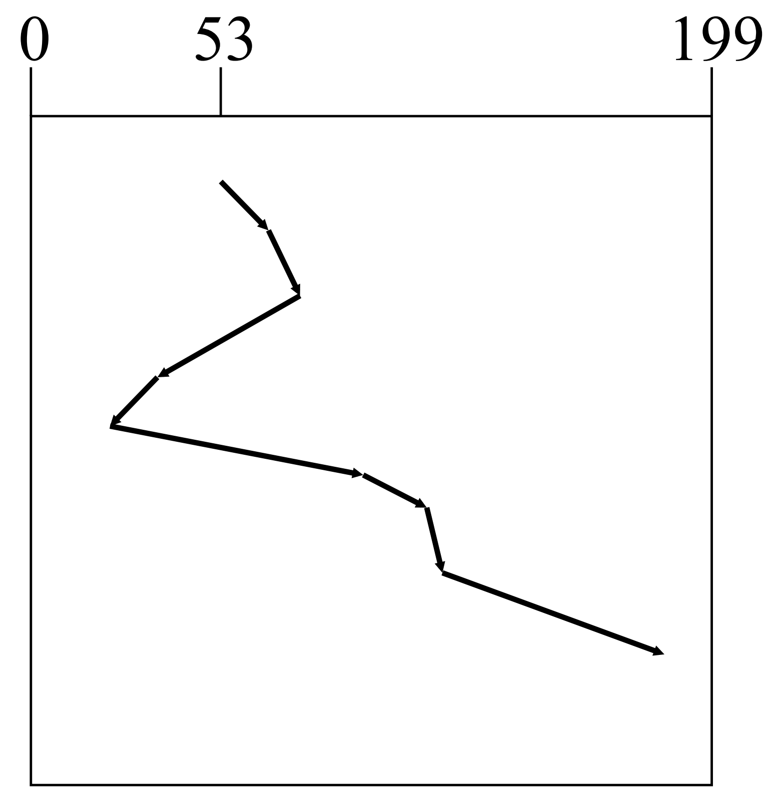
- Method
  - First come first serve
- Pros
  - Fairness among requests
  - In the order applications expect
- Cons
  - Arrival may be on random spots on the disk (long seeks)
  - Wild swing can happen



98, 183, 37, 122, 14, 124, 65, 67

SSTF (Shortest Seek Time First)

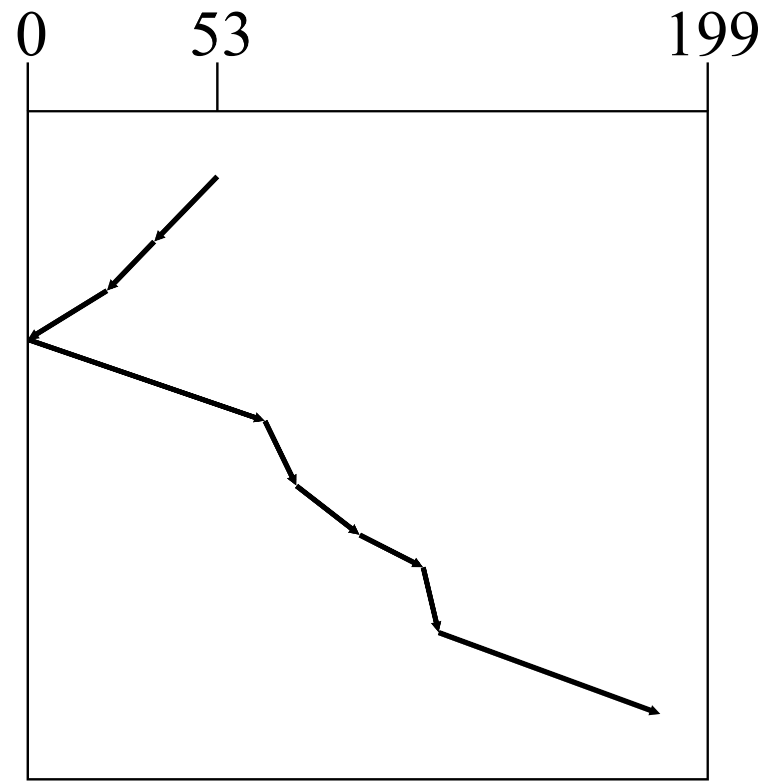
- Method
  - Pick the one closest on disk
  - Rotational delay is in calculation
- Pros
  - Try to minimize seek time
- Cons
  - Starvation
- Question
  - Is SSTF optimal?
  - Can we avoid the starvation?



98, 183, 37, 122, 14, 124, 65, 67
(65, 67, 37, 14, 98, 122, 124, 183)

Elevator (SCAN)

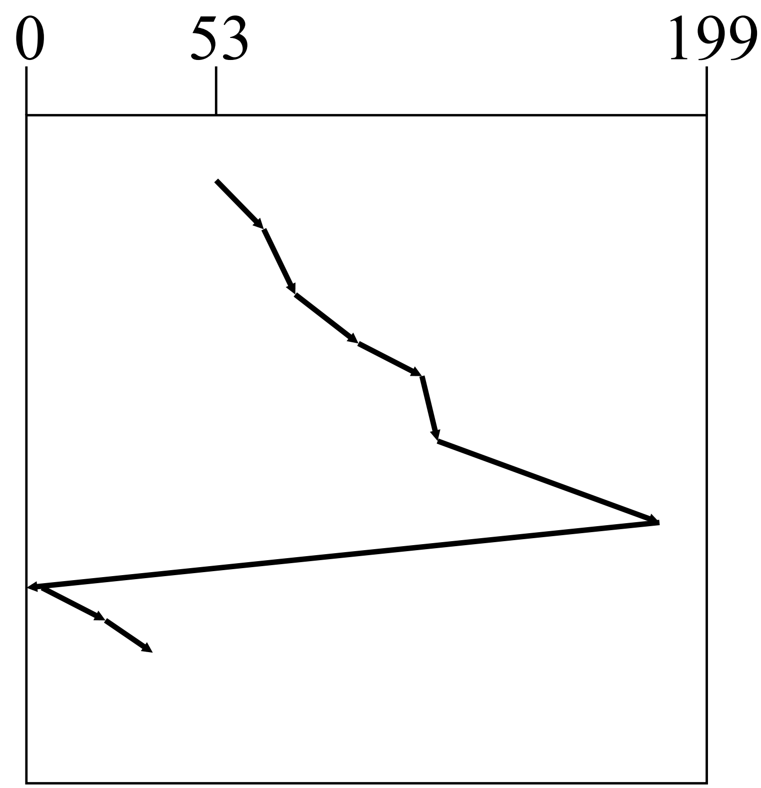
- Method
  - Take the closest request in the direction of travel
  - Real implementations do not go to the end (called LOOK)
- Pros
  - Bounded time for each request
- Cons
  - Request at the other end will take a while



98, 183, 37, 122, 14, 124, 65, 67
(37, 14, 65, 67, 98, 122, 124, 183)

C-SCAN (Circular SCAN)

- Method
  - Like SCAN
  - But, wrap around
  - Real implementation doesn't go to the end (C-LOOK)
- Pros
  - Uniform service time
- Cons
  - Do nothing on the return



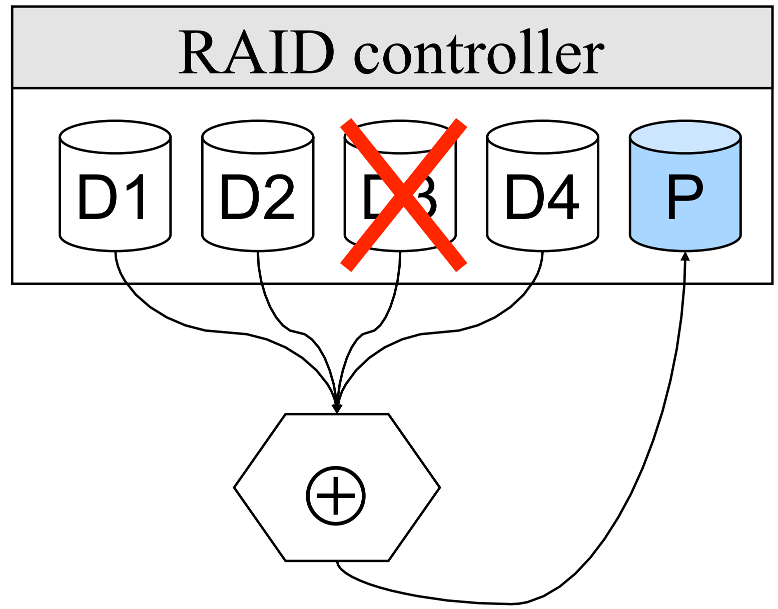
98, 183, 37, 122, 14, 124, 65, 67
(65, 67, 98, 122, 124, 183, 14, 37)

Discussions

- Which is your favorite?
  - FIFO
  - SSTF
  - SCAN
  - C-SCAN
- Disk I/O request buffering
  - Where would you buffer requests?
  - How long would you buffer requests?

RAID (Redundant Array of Independent Disks)

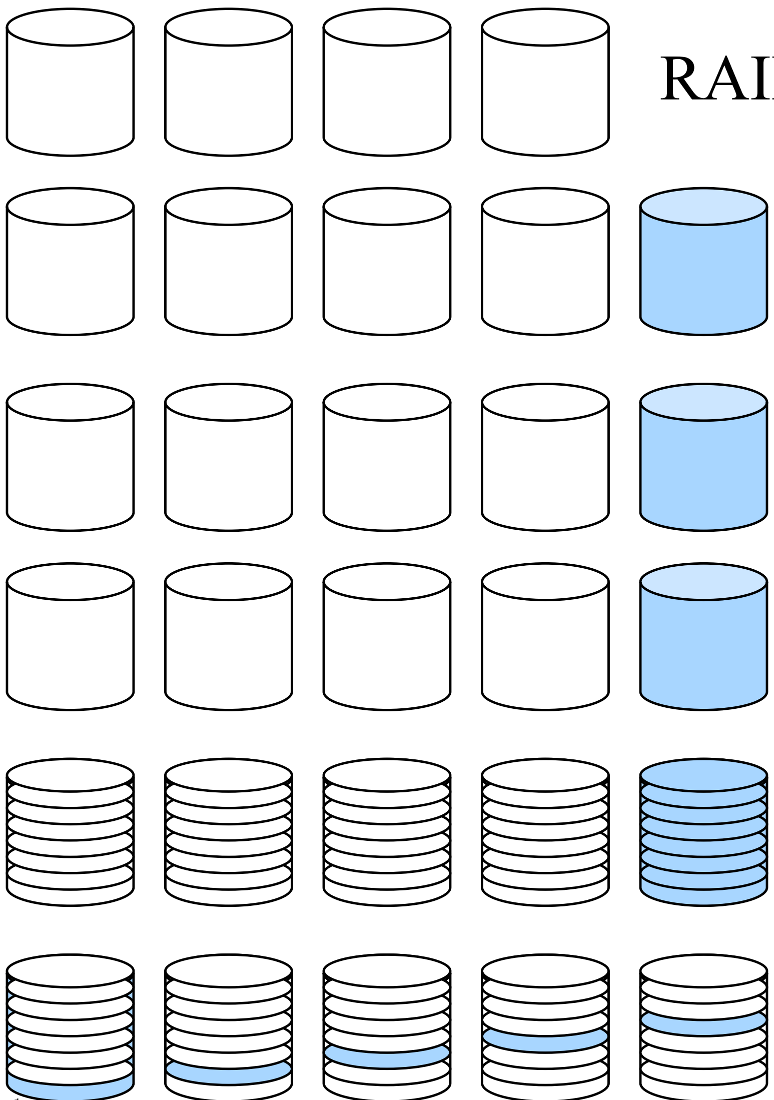
- Main idea
  - Store the error correcting codes on other disks
  - General error correcting codes are too powerful
  - Use XORs or single parity
  - Upon any failure, one can recover the entire block from the spare disk (or any disk) using XORs
- Pros
  - Reliability
  - High bandwidth
- Cons
  - Cost
  - The controller is complex



$$P = D1 \oplus D2 \oplus D3 \oplus D4$$

$$D3 = D1 \oplus D2 \oplus P \oplus D4$$

Synopsis of RAID Levels



RAID Level 0: Non redundant

RAID Level 1: Mirroring

RAID Level 2: Byte-interleaved, ECC

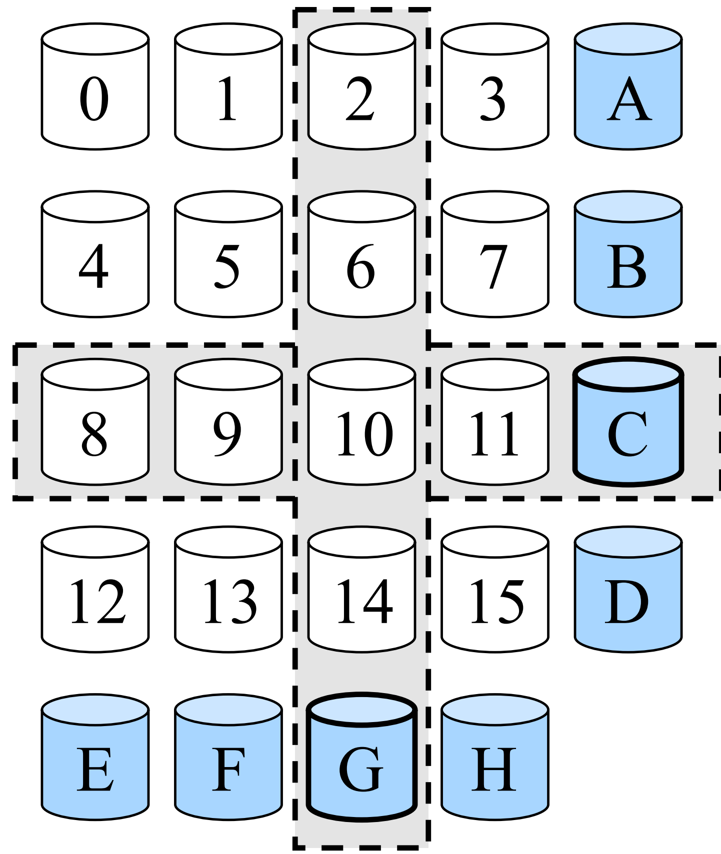
RAID Level 3: Byte-interleaved, parity

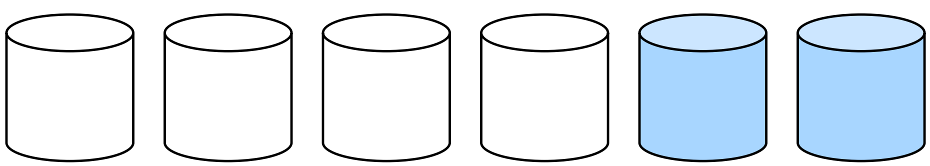
RAID Level 4: Block-interleaved, parity

RAID Level 5: Block-interleaved, distributed parity

RAID Level 6 and Beyond

- Goals
  - Less computation and fewer updates per random writes
  - Small amount of extra disk space
- Extended Hamming code
  - Remember Hamming code?
- Specialized Eraser Codes
  - IBM Even-Odd, NetApp RAID-DP, …
- Beyond RAID-6
  - Reed-Solomon codes, using MOD 4 equations
  - Can be generalized to deal with k (>2) disk failures





Next Generation: FLASH

- Flash chip density increases on the Moore's law curve
  - 1995 16 Mb NAND flash chips
  - 2005 16 Gb NAND flash chips
  - 2009 64 Gb NAND flash chips
    Doubled each year since 1995
- Market driven by Phones, Cameras, iPod,…
  Low entry-cost,
  ~$30/chip → ~$3/chip

- 2012 1 Tb NAND flash
  == 128 Gb chip
  == 1TB or 2TB "disk"
  for ~$400
  or 128GB disk for $40
  or 32GB disk for $5

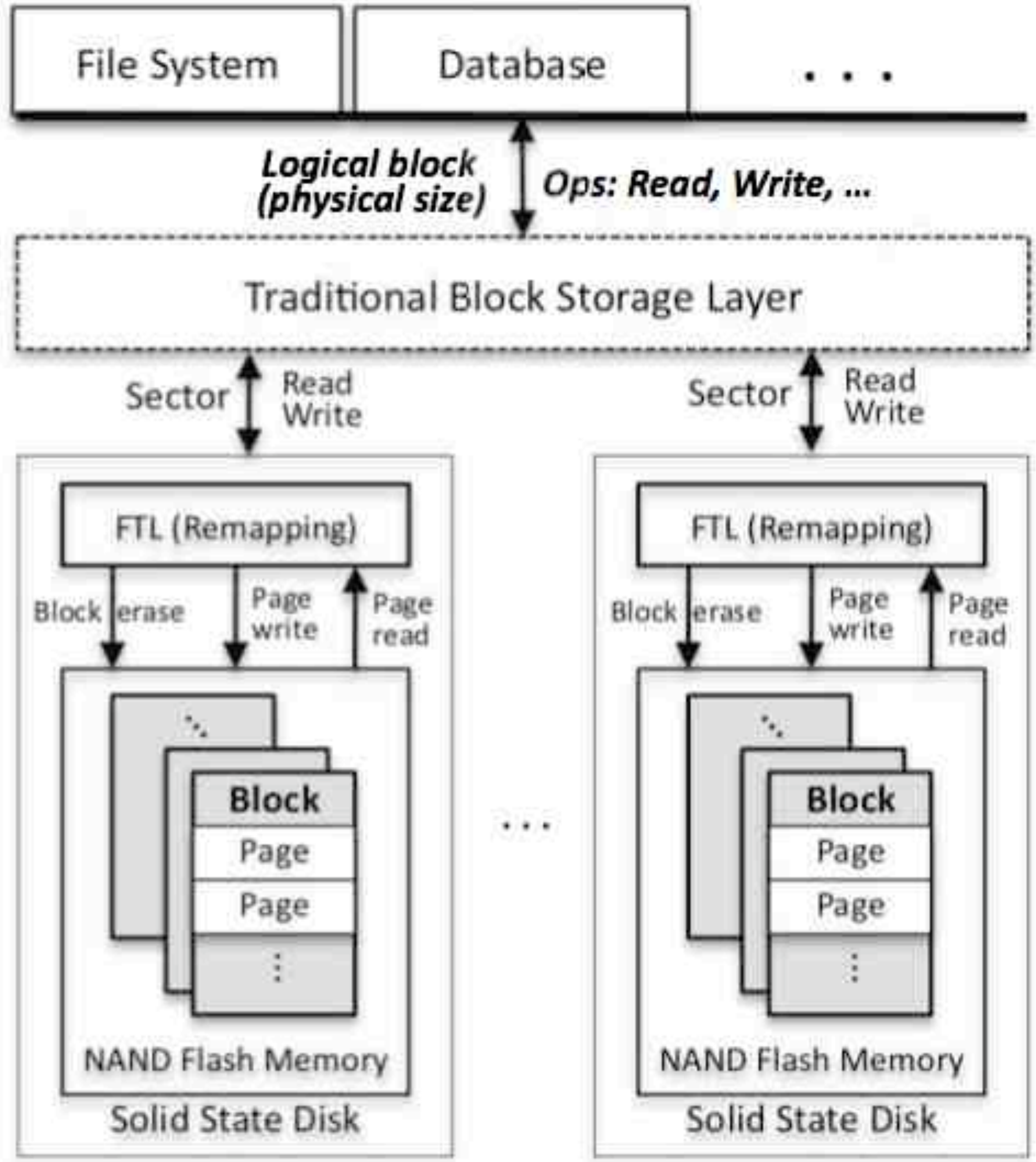
Samsung prediction

What's Wrong With FLASH?

- Expensive: $/GB
  - 2x less than cheap DRAM
  - 50x more than disk today, may drop to 10x in 2012
- Limited lifetime
  - ~50k to 100k writes / page (SLC)
  - ~15k to 60k writes / page (MLC)
  - But, suppose you do "wear leveling" and 200,000 writes/sec, If you have 1,000M pages on SLC flash (100k/page), it will take 15 years to wear out.
- Current performance limitations
  - Slow to write: can only write 0's, so erase (set all 1) then write
  - Large (e.g. 128K) blocks to erase

Current Development

- Flash Translation Layer (FTL)
  - Remapping
  - Wear-leveling
  - Write faster
- Form factors
  - SSD
  - USB, SD, Stick,…
  - PCI cards
- Performance
  - Fusion-IO with 2.5TB, 6GB/s r/w, 26µs latency



Summary

- Disk is complex
- Disk real density is on Moore's law curve
- Need large disk blocks to achieve good throughput
- System needs to perform disk scheduling
- RAID improves reliability and high throughput at a cost
- Flash memory has emerged at low and high ends