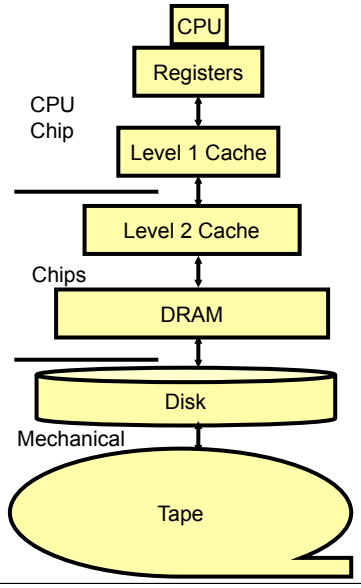
Lecture 21: Storage

- Administration
  - Take QUIZ 15 over P&H 6.1-4, 6.8-9 before 11:59pm today
  - Project: Cache Simulator, Due April 29, 2010
  - NEW OFFICE HOUR TIME: Tuesday 1-2, McKinley
- Last Time
  - Exam discussion
- Today
  - Reliable and Available Storage
    - Memory technology
    - Disk
    - Flash

Price/performance Modern Memory Hierarchy



Year: 2010

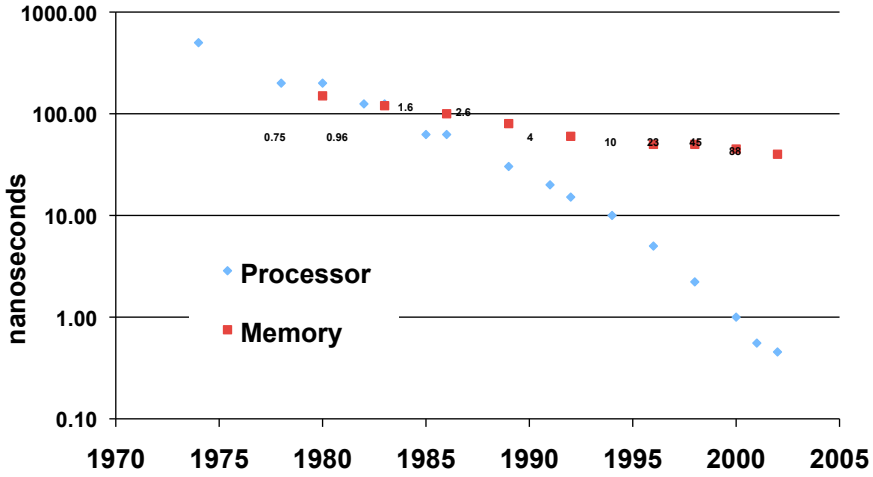
SRAM Cache
- access time: 0.5-2.5ns
- cost: ~$5000 per GB

DRAM Memory
- access time: 50-70ns
- cost: ~$20-75 per GB

Flash Memory
- access time: 70-150ns
- cost: ~$4-12 per GB

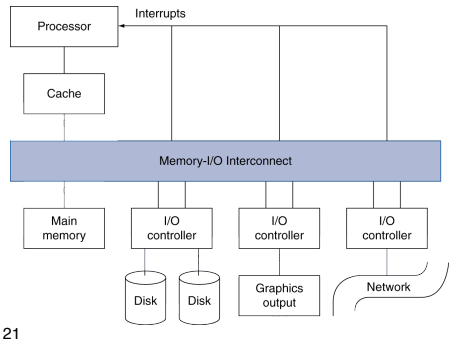
Magnetic Disk
- access time: 5-20ms
- cost: ~$0.02-2.0 per GB

Historical Perspective on Processor/Memory Gap



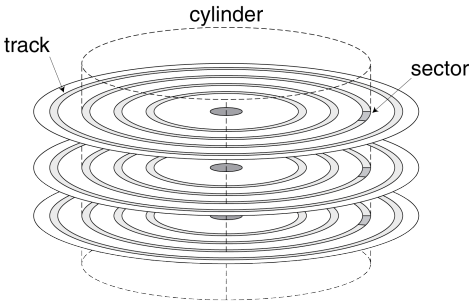
How do disks & other I/O devices fit in to the system architecture?

- Disk & other I/O device characteristics
  - Behavior: input, output, storage
  - Partner: human or machine
  - Data rate: bytes/sec, transfers/sec



Disk Storage

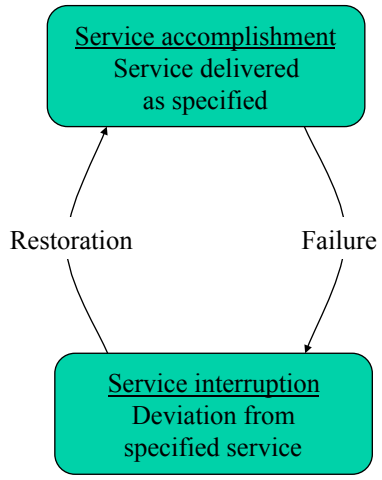
- Nonvolatile, rotating magnetic storage



What do we want from disk storage?

- Reliability!
  - Down time is a pain, but permanent data loss is a disaster
  - Analogy: oil change versus a car crash
- Performance!
  - Latency (response time)
  - Throughput (bandwidth)

Reliability



Fault: component failure
- May or may not lead to system failure

We can measure availability

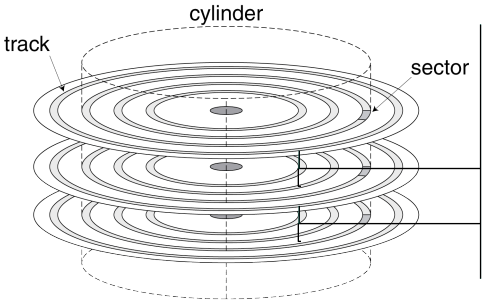
- Component Reliability: mean time to failure (MTTF)
- Service interruption: mean time to repair (MTTR)
- Mean time between failures
  - MTBF = MTTF + MTTR
- **Availability** = MTTF / (MTTF + MTTR)

- **Examples**:
  - MTTF = 1,000,000 hours (from vendor specs)
    - Availability = 99.5%
  - MTTF = 300,000 hours (Schroeder & Gibson 2007, Google)
    - Availability = 98 to 87%

Improving Availability

- Increase MTTF:
  - fault avoidance
  - fault tolerance
    - Redundancy
    - RAID: Redundant Arrays of Independent Disks
  - fault forecasting
- Reduce MTTR:
  - improved tools and processes for diagnosis and repair

How Disk Storage Works

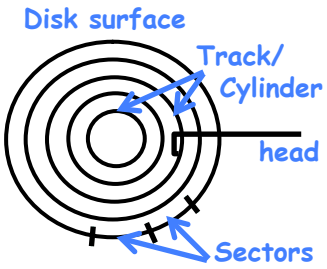


- Disk packs stack platters & use both sides of platters, except at the ends
- Each comb has 2 read/write assemblies on each arm
- Cylinders are matching sectors on each surface

Disk operations are in radial coordinates (track, sector)

1. move arm to track (cylinder)
2. Select & transfer sector as it spins by

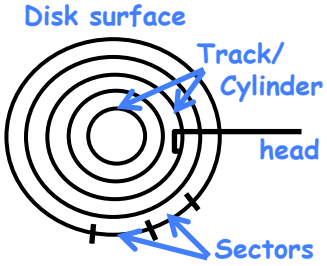
How Disk Storage Works



- Disk is always spinning (like a CD)
- Tracks are concentric rings on disk with bits laid out serially on tracks
- Bits are evenly spaced, thus there are more bits on outer tracks
- Each track is split into sectors or blocks, the minimum unit of transfer from the disk

- Sector size: 512 bytes
- Biggest: 1.5 TB Internal hard drive (June 2009)
- High End Desk Top (~$5,000): 4096 GB! (April 2010)
- Low End Desk Top (<$1000): 160 GB
- Iphone ($100-300): 4GB-32GB

How Disk Storage Works



To read or write a disk block:

Overhead time to start & schedule disk operation

Seek (latency) time to position head over track/cylinder. How fast does the hardware move the arm?

Rotational delay (latency) time for sector to rotate under head

Transfer (bandwidth) time to move bytes from disk to memory

I/O time = overhead + seek + rotational delay + transfer

Disk Access Example

Given disk spec:

- 512B sector
- 15,000 rpm
- 4ms average seek time,
- 100MB/s transfer rate,
- 0.2ms milliseconds controller overhead for idle disk

Average read time

= 4ms seek time
+ ½ / (15,000/60) = 2ms rotational latency
+ 512 / 100MB/s = 0.005ms transfer time
+ 0.2ms controller delay

= 6.2ms

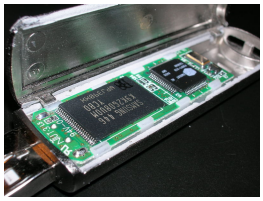
If actual average seek time is 1ms
Average read time = 3.2ms

Disk Performance Issues

- Manufacturers quote average seek time
  - Based on all possible seeks
  - Locality and OS scheduling lead to smaller actual average seek times
- Smart disk controller allocate physical sectors on disk
  - Present logical sector interface to host
  - SCSI, ATA, SATA
- Disk drives include caches
  - Prefetch sectors in anticipation of access
  - Avoid seek and rotational delay
- OS/disk drive scheduler increases locality

Flash Storage

- Nonvolatile semiconductor storage
  - 100× – 1000× faster than disk
  - Smaller, lower power, more robust
  - But more $/GB (between disk and DRAM)



Flash Types

- NOR flash: bit cell like a NOR gate
  - Random read/write access
  - Used for instruction memory in embedded systems
- NAND flash: bit cell like a NAND gate
  - Denser (bits/area)
  - block-at-a-time access
  - Cheaper per GB
  - Used for USB keys, media storage, …

- Flash bits wears out after 1000's of accesses
  - Not suitable for direct RAM or disk replacement
  - Wear leveling: remap data to less used blocks

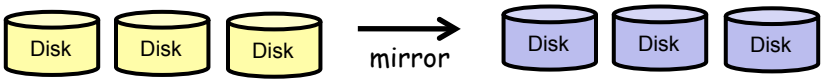
RAID: Disk Error Tolerance

- Redundant Array of Independent Disks
  - Use multiple smaller disks (c.f. one large disk)
  - Parallelism improves performance
  - Use extra disk(s) for redundant data storage
- Provides fault tolerance
  - Especially if failed disks can be "hot swapped"
- RAID 0
  - No redundancy ("AID"?)
    - Stripe data over multiple disks
  - Improves performance

RAID 1 & 2

RAID 1: Mirroring

- N + N disks, replicate data
  - Write data to both data disk and mirror disk
  - On disk failure, read from mirror

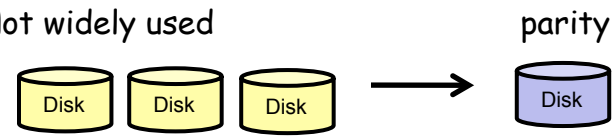


RAID 2: Error Correcting Code (ECC)

- N + E disks (e.g., 10 + 4)
- Split data at bit level across N disks
- Generate E-bit ECC
- Too complex, not used in practice

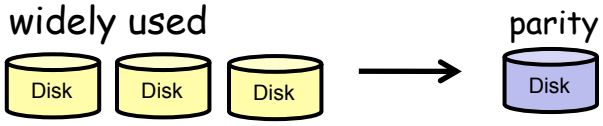
RAID 3: Bit-Interleaved Parity

- N + 1 disks
  - Data striped across N disks at byte level
  - Redundant disk stores parity
  - Read access
    - Read all disks
  - Write access
    - Generate new parity and update all disks
  - On failure
    - Use parity to reconstruct missing data
- Not widely used

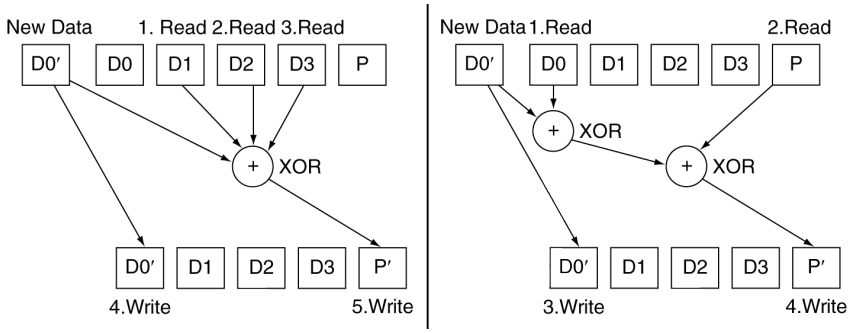


RAID 4: Block-Interleaved Parity

- N + 1 disks
  - Data striped across N disks at block level (vs bytes in RAID 3)
  - Redundant disk stores parity for a group of blocks
  - Read access
    - Read only the disk holding the required block
  - Write access
    - Read disk containing modified block, and parity disk
    - Calculate new parity, update data disk and parity disk
  - On failure: Use parity to reconstruct missing data
- Not widely used

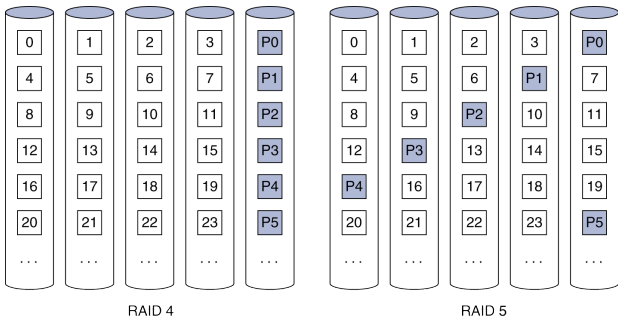


RAID 3 vs RAID 4



RAID 5: Distributed Parity

- N + 1 disks
  - Like RAID 4, but parity blocks distributed across disks
    - Avoids parity disk being a bottleneck
- Widely used



RAID 6: P + Q Redundancy

- N + 2 disks
  - Like RAID 5, but store parity bits twice
  - Greater fault tolerance through more redundancy
- Multiple RAID
  - More advanced systems give similar fault tolerance with better performance

RAID Summary

- RAID can improve performance and availability
  - High availability requires hot swapping
- Assumes independent disk failures
  - Too bad if the building burns down!

Summary

- Disk storage
  - Reliable, Available, Error tolerant
  - MTTF
  - Performance
  - RAID for performance and reliability
- Next Time
  - Interconnects & I/O
- Reading: P&H 6.6-10, 6.12-14