RAID

- Redundant Array of Inexpensive (Independent) Disks
  - Use multiple smaller disks (c.f. one large disk)
  - Parallelism improves performance
  - Plus extra disk(s) for redundant data storage
- Provides fault tolerant storage system
  - Especially if failed disks can be “hot swapped”

- RAID 0
  - No redundancy (“AID”?)
    - Just stripe data over multiple disks
  - But it does improve 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
  - Redundant disk stores parity for a group of blocks
- Read access
  - Read only the disk holding the required block
- Write access
  - Just 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

New Data 1. Read 2. Read 3. Read

D0' D0 D1 D2 D3 P

+ XOR

D0' D1 D2 D3 P'

4. Write 5. Write

New Data 1. Read

D0' D0 D1 D2 D3 P

+ XOR

D0' D1 D2 D3 P'

2. Read

3. Write 4. Write

Chapter 6 — Storage and Other I/O Topics — 33
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 two lots of parity
  - 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!
- See “Hard Disk Performance, Quality and Reliability”
I/O System Design

- Satisfying latency requirements
  - For time-critical operations
  - If system is unloaded
    - Add up latency of components

- Maximizing throughput
  - Find “weakest link” (lowest-bandwidth component)
  - Configure to operate at its maximum bandwidth
  - Balance remaining components in the system

- If system is loaded, simple analysis is insufficient
  - Need to use queuing models or simulation
Server Computers

- Applications are increasingly run on servers
  - Web search, office apps, virtual worlds, …
- Requires large data center servers
  - Multiple processors, networks connections, massive storage
  - Space and power constraints
- Server equipment built for 19” racks
  - Multiples of 1.75” (1U) high
Rack-Mounted Servers

Sun Fire x4150 1U server

- 2 Redundant power Supplies
- 3 PCI Express Slots
- System Status LEDs
- Management NIC
- 2 USB Ports
- 4 Gigabit NICs
- Video

Chapter 6 — Storage and Other I/O Topics — 39
Sun Fire x4150 1U server

- 4 cores each
- 16 x 4GB = 64GB DRAM
I/O System Design Example

- Given a Sun Fire x4150 system with
  - Workload: 64KB disk reads
    - Each I/O op requires 200,000 user-code instructions and 100,000 OS instructions
  - Each CPU: $10^9$ instructions/sec
  - FSB: 10.6 GB/sec peak
  - DRAM DDR2 667MHz: 5.336 GB/sec
  - PCI-E 8× bus: $8 \times 250\text{MB/sec} = 2\text{GB/sec}$
  - Disks: 15,000 rpm, 2.9ms avg. seek time, 112MB/sec transfer rate

- What I/O rate can be sustained?
  - For random reads, and for sequential reads
Design Example (cont)

- I/O rate for CPUs
  - Per core: \( \frac{10^9}{(100,000 + 200,000)} = 3,333 \)
  - 8 cores: 26,667 ops/sec

- Random reads, I/O rate for disks
  - Assume actual seek time is average/4
  - Time/op = seek + latency + transfer
    \[ = \frac{2.9\text{ms}}{4} + \frac{4\text{ms}}{2} + \frac{64\text{KB}}{112\text{MB/s}} = 3.3\text{ms} \]
  - 303 ops/sec per disk, 2424 ops/sec for 8 disks

- Sequential reads
  - \( \frac{112\text{MB/s}}{64\text{KB}} = 1750 \text{ ops/sec per disk} \)
  - 14,000 ops/sec for 8 disks
Design Example (cont)

- PCI-E I/O rate
  - 2GB/sec / 64KB = 31,250 ops/sec

- DRAM I/O rate
  - 5.336 GB/sec / 64KB = 83,375 ops/sec

- FSB I/O rate
  - Assume we can sustain half the peak rate
  - 5.3 GB/sec / 64KB = 81,540 ops/sec per FSB
  - 163,080 ops/sec for 2 FSBs

- Weakest link: disks
  - 2424 ops/sec random, 14,000 ops/sec sequential
  - Other components have ample headroom to accommodate these rates
Fallacy: Disk Dependability

- If a disk manufacturer quotes MTTF as 1,200,000hr (140yr)
  - A disk will work that long
- Wrong: this is the mean time to failure
  - What is the distribution of failures?
  - What if you have 1000 disks
  - How many will fail per year?

\[
\text{Annual Failure Rate (AFR)} = \frac{1000 \text{ disks} \times 8760 \text{ hrs/disk}}{1200000 \text{ hrs/failure}} = 0.73\%
\]
Fallacies

- Disk failure rates are as specified
  - Studies of failure rates in the field
    - Schroeder and Gibson: 2% to 4% vs. 0.6% to 0.8%
    - Pinheiro, et al.: 1.7% (first year) to 8.6% (third year) vs. 1.5%
  - Why?

- A 1GB/s interconnect transfers 1GB in one sec
  - But what’s a GB?
  - For bandwidth, use $1\text{GB} = 10^9\text{B}$
  - For storage, use $1\text{GB} = 2^{30}\text{B} = 1.075\times10^9\text{B}$
  - So 1GB/sec is 0.93GB in one second
    - About 7% error
Pitfall: Offloading to I/O Processors

- Overhead of managing I/O processor request may dominate
  - Quicker to do small operation on the CPU
  - But I/O architecture may prevent that
- I/O processor may be slower
  - Since it’s supposed to be simpler
- Making it faster makes it into a major system component
  - Might need its own coprocessors!
Pitfall: Backing Up to Tape

- Magnetic tape used to have advantages
  - Removable, high capacity
- Advantages eroded by disk technology developments
- Makes better sense to replicate data
  - E.g, RAID, remote mirroring
Fallacy: Disk Scheduling

- Best to let the OS schedule disk accesses
  - But modern drives deal with logical block addresses
    - Map to physical track, cylinder, sector locations
    - Also, blocks are cached by the drive
  - OS is unaware of physical locations
    - Reordering can reduce performance
    - Depending on placement and caching
Pitfall: Peak Performance

- Peak I/O rates are nearly impossible to achieve
  - Usually, some other system component limits performance
  - E.g., transfers to memory over a bus
    - Collision with DRAM refresh
    - Arbitration contention with other bus masters
  - E.g., PCI bus: peak bandwidth ~133 MB/sec
    - In practice, max 80MB/sec sustainable
Concluding Remarks

- I/O performance measures
  - Throughput, response time
  - Dependability and cost also important
- Buses used to connect CPU, memory, I/O controllers
  - Polling, interrupts, DMA
- I/O benchmarks
  - TPC, SPECSFS, SPECWeb
- RAID
  - Improves performance and dependability