# Appendix D: Storage Systems

Instructor: Josep Torrellas

CS433

# Storage Systems : Disks

Used for → long term storage of files

→ temporarily store parts of pgm when running

Disk : collection of 1-12 platters

rotating at 3,600-15,000 RPM's

magnetic material on both sides (surfaces)

diameter = 1.0 - 3.5 in

each surface = 5,000 - 30,000 tracks

each track ≈ 100-500 sectors

sector : smallest unit that can be read /written

# Storage Systems : Disks

- Traditionally, all tracks = same # sectors → outer tracks record info at lower density; now more sectors on outer tracks
- Movable arm contains Rd/Wr head over each surface
- Arms move together
- Cylinder : all tracks under the arms at a point in time
- Time :

  → seek : move arm to proper track

  → rotational delay : average latency = 1/2 rotation

  Avg Rot. Del = 0.5/10,000 rpm = 3.0 ms

# Storage Systems : Disks

→ transfer time : Time taken to transfer 1 block under the rd/wr head (3-65 MB/s)

→ disk controller time :

→ queuing delay : time until disk is available

Avg time to rd 512 B sector, seek = 5 ms , Xfer rate = 40 MB/s
rotation 10,000 rpm ; controller ovh = 0.1ms ; no queuing

$$5\text{ms} + \frac{0.5}{10000} + \frac{0.5\text{ KB}}{40.0\text{ MB/s}} + 0.1\text{ms}$$

5 + 3.0 + 0.013 + 0.1 = 8.11 ms

# Storage Systems : Disks

Areal density in bits per square inch:

Tracks /inch on a disk surface * bits/inch on a track

100% increase per year => double every year

(20 Billion bits/sq in now)

Magnetic disks are challenged flash memory:

- Non-volatile
- Latency 100-1000 times lower than disks
- But: Wearout

# RAID: Redundant Arrays of Inexpensive Disks

- Disk arrays: Have many disk drives and, therefore, many disk arms (rather than a single disk arm):
  - increase potential throughput
  - unfortunately, with many more devices, dependability decreases: N devices generally have 1/Nth of the reliability of a single device.
  - Result: disk array has many more faults than a smaller number of larger disks
- Add redundant disks to tolerate faults:
  - dependability increases
  - if a single disk fails: the lost information is reconstructed from the redundant information
- Result: RAID: redundant array of inexpensive disks

# Issues

- Spread the data over multiple disks: Striping
- If second disk fails while the first one is being repaired, cannot recover
- Not a problem: MTTF of a disk is tens of years, while MTTR is hours --> redundancy makes the measured reliability of 100 disks much higher than that of a single disk.

# Other Issues

- Detecting disk faults: usually feasible
- Design of RAIDs that decrease the MTTR: include **hot spares on the system:** extra disks not used in normal operation that are pressed into service if a failure occurs
- Data missing from the failed disk are reconstructed onto the hot spare using the redundant data from the other RAID disks
- Done automatically, which reduces MTTR
- **Hot Swapping:** Components are replaced without shutting down the computer
- Overall: a system with hot spares and hot swapping never goes offline.

# Different RAID levels (Fig D.4)

| RAID level | | Disk failures tolerated, check space overhead for 8 data disks | Pros | Cons | Company products |
|---|---|---|---|---|---|
| 0 | Nonredundant striped | 0 failures, 0 check disks | No space overhead | No protection | Widely used |
| 1 | Mirrored | 1 failure, 8 check disks | No parity calculation; fast recovery; small writes faster than higher RAIDs; fast reads | Highest check storage overhead | EMC, HP (Tandem), IBM |
| 2 | Memory-style ECC | 1 failure, 4 check disks | Doesn't rely on failed disk to self-diagnose | ~ Log 2 check storage overhead | Not used |
| 3 | Bit-interleaved parity | 1 failure, 1 check disk | Low check overhead; high bandwidth for large reads or writes | No support for small, random reads or writes | Storage Concepts |
| 4 | Block-interleaved parity | 1 failure, 1 check disk | Low check overhead; more bandwidth for small reads | Parity disk is small write bottleneck | Network Appliance |
| 5 | Block-interleaved distributed parity | 1 failure, 1 check disk | Low check overhead; more bandwidth for small reads and writes | Small writes → 4 disk accesses | Widely used |
| 6 | Row-diagonal parity, EVEN-ODD | 2 failures, 2 check disks | Protects against 2 disk failures | Small writes → 6 disk accesses; 2× check overhead | Network Appliance |

# No Redundancy (RAID 0)

- Data are striped but there is no redundancy to tolerate disk failure
- Data appears to the software as laid out in a single large disk
- Improves the performance for large accesses because many disks operate in parallel

# Mirroring (RAID 1)

- Also called shadowing
- Use twice as many disks: when data are written to one disk, they are also written to a second one
- If a disk fails, the system goes to the mirror to get the desired information
- Very expensive

# Bit-Interleaved Parity (RAID 3)

- Have a **Protection Group**, composed of N disks. One additional disk is used to keep redundant data to restore lost information on a failure
- Popular implementation
- Parity is one example of this scheme
- Assumption: failures very rare
- Mirroring is a special case where N=1

# Block-Interleaved Parity (RAID 4)

- Use the same ratio of data disks and check disks as RAID3 but they access data differently
- Parity is stored as blocks and associated with a set of data blocks
- RAID 4: On a write, instead of reading N-1 disks and updating 1 disk and the parity, we read 1 disk and the parity and write one disk and the parity
- RAID 4: Good for small writes

# Distributed Block-Interleaved Parity (RAID 5)

- RAID 4: parity disk is updated on every write --> it becomes the bottleneck
- RAID 5: distribute the parity throughout all the disk so that there is no single bottleneck for writes

# P + Q Redundancy (RAID 6)

- Adds a second disk per protection group
- This second disk performs a second calculation over the data
- Allows the recovery from a second failure

# DMA (Direct Memory Access)

DMA : hardware to perform transfers of data Mem ↔ I/O without bothering CPU

→ DMA is like specialized processor

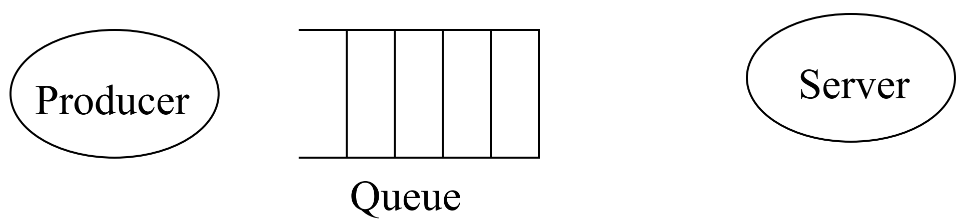
→ acts as a master of the bus

→ CPU sets the DMA regs (mem address,disk blocks,#bytes)

→ once DMA done , CPU is interrupted

# I/O Performance Measures

- Response time = time finish server – time deposited in queue
- Throughput = #tasks completed by server / unit time



# I/O Performance Measures

→ if server always busy ⇒ highest throughput

⇒ high response time

- computer transaction has :
  - entry time: time for user to enter the command

    →graphics = 0.25 sec ; keyboard 4.0s
  - system response time : time until the response is displayed
  - think time : time from reception of response to user begins to enter new command
- Transaction time

  productivity $\alpha \quad \frac{1}{\text{transaction time}}$

- Observation : If response time ↑ , think time also ↑
  as a result , transaction time ↑ ↑