Chapter 2 Lecture 3

Computer Systems Organization

The Memory Hierarchy Pyramid

Registers: CPU local storage, single clock cycle access, small capacity (2-10’s) Inside CPU

Cache: High speed buffering memory, multiple clock cycles, small to moderate capacity (100’s to 1000’s of words) 1st level usually on CPU, 2nd and higher discrete SRAM

Main Memory: Moderate speed primary memory storage, 10’s to 100’s of clock cycles moderate to large capacity (64 MB, 2 Mwords @ 32 bit words, and higher) CPU to Memory Bridge chips to access DRAM

Magnetic Disk: Slow speed on-line program and data storage, 100’s of clock cycles or more large capacity (2 GB, 64 Mwords @ 32 bits, and higher) Bus to Hard Driver chips to access Hard Disk Drives (typ. EIDE or SCSI I/F) Network to disk server farms

Off-Line: Simple, slow speed removable storage, 1000’s of clock cycles or more moderate capacity per disk, CD, DVD, etc. (144 MG, 750 MB, etc.) Bus to peripheral chips to access drives (typ. USB or EIDE or SCSI I/F)

Archival: Large, slow speed removable storage, “overnight archives” significant capacity desired (GB per day), archive tapes, “Data Safe Storage” Network to bulk storage devices
IC Memory Types and Cycles

<table>
<thead>
<tr>
<th>Memory Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| ROM         | Read-only Memory (NEC)  
Permanent storage (boot code, embedded code) |
| SRAM        | Static Random Access Memory (Alliance Semi)  
Cache and high speed access |
| DRAM        | Dynamic Random Access Memory (Micron)  
Main Memory |
| EPROM       | Electrically programmable read-only memory (Atmel)  
Replace ROM when reprogramming required |
| EEPROM      | Electrically erasable, programmable read-only memory (Atmel)  
Alternative to EPROM, limited but regular reprogramming,  
Device configuration info during power down (USB memories) |
| FLASH       | An advancement on EEPROM technology allowing blocks of memory location to be written and cleared at one time instead. (Samsung). Found in thumb drives/memory stick or as solid-state hard disks. Note: EEPROM and FLASH have lifetime write cycle limitations! |

Datasheets are readily available from manufacturers on the web

Notice the speeds!
Average Memory Access Time (Registers and Main Memory) \( t_{aa} \)

If data is in either registers or memory:

\[
t_{aa} = t_{reg} \cdot p_{reg} + t_{mm} \cdot p_{mm}
\]

- \( t_{reg} \) Register Access Time
- \( p_{reg} \) Probability that the data is in the register
- \( t_{mm} \) Main Memory Access Time
- \( p_{mm} \) Probability that the data is in the register

For this problem, note that

\[
p_{reg} + p_{mm} = 1
\]

We define

\[
p_{reg} = h_{reg} \quad \text{The hit rate for data in the register}
\]
\[
p_{mm} = (1 - h_{reg}) \quad \text{The miss rate for data in the register}
\]

For \( t_{reg} = 1 \text{ns} \), \( h_{reg} = 50\% \), and \( t_{mm} = 100 \text{ns} \)

\[
t_{aa} = t_{reg} \cdot h_{reg} + t_{mm} \cdot (1 - h_{reg})
\]

\[
t_{aa} = 1 \cdot 0.5 + 100 \cdot (1 - 0.5) = 50.5 \text{ns}
\]

\[
t_{aa} = 0.5 + 50 = 50.5 \text{ns}
\]
Cache Memory

From the French *cacher*, to hide, it tries to hide main memory latency
Pronounced: “cash”

The success of cache in speeding up access to main memory is due to: the **Locality of Reference Principles**

- **Sequential** Programs and data storage tend to be sequential
- **Spatial** Data storage areas tend to be in spatial blocks
- **Temporal** Memory recently used in time

Cache Storage Concept: Cache is stored as “lines” or blocks
Typical line sizes are 32 to 128 bytes

Memory is preferably moved between cache and main memory as cache lines, that is in bursts of bytes. This has led to significant design changes in DRAMs and speed increases in DRAM access.

Hard disks also use a cache to speed reads and writes.
Average memory access time (Registers, Cache and Main Memory)

If data is in either registers, cache or memory:

\[
t_{aa} = t_{reg} \cdot p_{reg} + t_{cache} \cdot p_{cache} + t_{mm} \cdot p_{mm}
\]

- \(t_{reg}\) Register Access Time
- \(p_{reg}\) Probability that the data is in the register
- \(t_{cache}\) Cache Access Time
- \(p_{cache}\) Probability that the data is in the cache
- \(t_{mm}\) Main Memory Access Time plus Cache Miss Time (MMU)
- \(p_{mm}\) Probability that the data is in the register

We define

\[
p_{reg} = h_{reg} \quad \text{The hit rate for data in the register}
\]
\[
p_{cache} = (1 - h_{reg}) \cdot h_{cache} \quad \text{The cache hit rate for data in the cache}
\]
\[
p_{mm} = (1 - h_{reg}) \cdot (1 - h_{cache}) \quad \text{The miss rate for data not in the register or cache}
\]

\[
t_{aa} = t_{reg} \cdot h_{reg} + t_{cache} \cdot (1 - h_{reg}) \cdot h_{cache} + t_{mm} \cdot (1 - h_{reg}) \cdot (1 - h_{cache})
\]

For \(t_{reg} = 1\, ns\), \(h_{reg} = 50\%\), \(t_{cache} = 20\, ns\), \(h_{cache} = 90\%\), and \(t_{mm} = 100\, ns\):

\[
t_{aa} = 1 \cdot 0.5 + 20 \cdot (1 - 0.5) \cdot 0.9 + 100 \cdot (1 - 0.5) \cdot (1 - 0.9)
\]

\[
t_{aa} = 0.5 + 9 + 5 = 14.5\, ns
\]
Average memory access time (L1 & L2 Cache and Main Memory)

\[ t_{aa} = t_{L1\text{cache}} \cdot P_{L1\text{cache}} + t_{L2\text{cache}} \cdot P_{L2\text{cache}} + t_{mm} \cdot P_{mm} + t_{other} \cdot P_{other} \]

- \( t_{L1\text{cache}} \): L1 cache Access Time
- \( P_{L1\text{cache}} \): Probability that the data is in the L1 cache
- \( t_{L2\text{cache}} \): L2 Cache Access Time plus L1 Cache Miss Time (L1CMU)
- \( P_{L2\text{cache}} \): Probability that the data is in the L2 cache
- \( t_{mm} \): Main Memory Access Time plus L1 and L2 Cache Miss Time
- \( P_{mm} \): Probability that the data is in the register
- \( t_{other} \): Data Access Time for other storage media or I/O plus previous Time
- \( P_{other} \): Probability that the data is in other storage media or I/O

Assume registers are “zero” access time and relate memory only access time.

\[ t_{aa} = t_{L1\text{cache}} \cdot h_{L1\text{cache}} + t_{L2\text{cache}} \cdot (1 - h_{L1\text{cache}}) \cdot h_{L2\text{cache}} + t_{mm} \cdot (1 - h_{L1\text{cache}}) \cdot (1 - h_{L2\text{cache}}) \cdot h_{mm} + \cdots \]
Cache Architecture Locations

Historic System Architecture
Memory Management Units

Historic Cache System Architecture
Frontside Cache with MMU

Backside Cache Architecture Locations

Backside Cache

L1 Backside Cache and L2 Cache

Note: These are example configurations. Optimizing data flow and minimizing latency define how a computer architecture uses caches, memory management and bus interfaces.
Cache Architectures: Unified and Split Cache, P6 Architecture

There are two options when “inserting” a cache based on “system architecture”.

<table>
<thead>
<tr>
<th>Inst. Unit</th>
<th>Proc. Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>I-Cache</td>
<td>D-Cache</td>
</tr>
<tr>
<td>Inst. Unit</td>
<td>Proc. Unit</td>
</tr>
</tbody>
</table>

Unified Cache (Von Neumann)  Split Cache (Harvard Architecture)

Cache Architecture  Instructions and Data are needed at different parts of instruction cycles
Instructions and Data do have differences in their dependence on specific
Locality of Reference Principles

Modern machines typically have split L1 caches.

Level 1 Instruction Cache: Optimized for threads of execution, support for branching, integrated with intelligent instruction fetch and preprocessing units. Focus on filling from higher level caches and main memory. It does not typically need to write-back instruction to main memory.

Level 1 Data Cache: Optimized for data handling. Facilitate data fetching for computations and data write back of results.

Therefore, the cache controllers are concerned about different “cache policies and procedures”.
Intel P6 Processor Architecture with Caches


Figure 1: Basic block diagram
Intel P6 based PC Architecture

Figure 2-30. A typical modern PC with a PCI bus and an ISA bus. The modem and sound card are ISA devices; the SCSI controller is a PCI device.

Another version: [http://computer.howstuffworks.com/pci1.htm](http://computer.howstuffworks.com/pci1.htm)
Secondary Memory
Magnetic Disk, Floppy Disk, Optical Disk, etc.

Magnetic Memory    Hard Drives and Floppy Drives

Track: A radial spaced circle on a platter for storing data. The disk read/write head remains at a fixed radial distance while reading or writing a track. Typically 5,000 to 10,000 tracks per centimeter on a platter or 1-2 micron track widths.

Sector: A fixed bit-length section of a track. There are multiple sectors in a track.

A typical sector contains: a preamble, 512 Bytes or 4096 bits, and error correction code bits. Between sectors on a track are inter-sector gaps. “linear” bit densities of 50,000 to 100,000 bits/cm have been achieved. Marketing speak: quote disk size in unformatted bits … take off 15% for reality.
Cylinder: When multiple platters exist in a hard disk, multiple tracks from different platters can be accessed simultaneously at the same radial distance. This makes up a cylinder.

Note: at different radial distances, the cylinder/track may have more sectors. Zones with set number of sectors can be (are typically) defined

Zones: Based on the radial distance of the cylinder from the center to the outer edge of the platters, there can be a different number of sectors per track.

Original “stored data” constant rpm; therefore, bits on inner rings were more dense than bits on outer rings. (fixed pie-slices for each sector). With variable rpm, the “bit density” can be made consistent, but then different radial “Zones” end up having different numbers of sectors. The hard disk controller must know this!
Western Digital Drive Info -

See www.howstuffworks.com for descriptions and another overview, specifically:
http://computer.howstuffworks.com/hard-disk.htm
Reading and Writing a Disk:

*Seek Time:* First the head must be positioned on the right track (radial distance).
Typical numbers: 5-15 msec.
“Avg. seek time”: not best case, not worst case, average!
Min. time – one track to the next
Max. time – outer track to inner track or vice versa

*Rotational Latency:* The head must arrive at the correct sector on a track (rotational distance)
Disks have fixed rotation rates, such as: 3,600, 5,400, 7200, or 10,800 RPM
RPM in revolutions per minute

<table>
<thead>
<tr>
<th>RPM</th>
<th>rev/sec.</th>
<th>msec/rev</th>
<th>Avg. Rotation Latency (msec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3600</td>
<td>60</td>
<td>16.67</td>
<td>8.33</td>
</tr>
<tr>
<td>5400</td>
<td>90</td>
<td>11.11</td>
<td>5.56</td>
</tr>
<tr>
<td>7200</td>
<td>120</td>
<td>8.33</td>
<td>4.17</td>
</tr>
<tr>
<td>10800</td>
<td>180</td>
<td>5.56</td>
<td>2.78</td>
</tr>
</tbody>
</table>

*Access Time:* = Seek Time + Rotational Time
typically defined as an average – avg. seek time + avg. rotational latency

**Example:** A drive with 8.9 msec avg. seek time at 7200 RPM
(Western Digital WD800JBRTL drive)

Access Time = 8.9 msec + 4.17 msec = 13.07 msec.

*Transfer Time:* Actual time to transfer a sector … read the bits
Typically from 2 to 150 MB/sec “bursts”
(ATA/100, ATA/150 interfaces with cache buffers)

Western Digital WD800JBRTL buffer to disk max.: 525 Mbits/sec
using an 8 MB cache
Maximum Possible Single Sector Read Rates (not realistic) – multiply by cylinder heads.

Example: If there are 16 or 64 sectors on a track and the rotation rate is 7200 RPM, what is the maximum possible data transfer rate to read one sector on the disk.

7200 RPM → 8.33 msec/rev
For 16 sectors → 8.33/16 = 0.52 msec/sector
4096 bits/sector → 4096/0.00052 = 7,876,923 bits/sec ~ 0.98 MB/sec.

7200 RPM → 8.33 msec/rev
For 64 sectors → 8.33/64 = 0.13 msec/sector
4096 bits/sector → 4096/0.00013 = 31,507,692 bits/sec ~ 3.94 MB/sec.

Example: If there are 128 sectors on a track and the rotation rate is 10800 RPM, what is the maximum possible data transfer rate to read the disk.

10800 RPM → 5.56 msec/rev
128 sectors → 5.56/128 = 0.0434 msec/sector
4096 bits/sector → 4096/0.0000434 = 94,371,840 bits/sec ~ 11.8 MB/sec.

**NOTE: This is a maximum burst rate. Don’t forget the access time to get to the data!**

Note: to support (or even get close to) ATA rates of 100-150 MB/sec and quoted WD rate, *multiple tracks in a cylinder must be read simultaneously!*

Multiply the rates defined by the number of surfaces simultaneously read, and higher rates can be achieved!
Example Average Access Time plus Transfer Time

Time to read one sector given: 10,800 RPM, 8.0 msec avg. seek time and 64 sectors per track.

\[
\text{Avg. Access Time} = 8.0 \text{ msec} + (5.56/2 \text{ msec}) = 10.78 \text{ msec} \\
\text{Transfer time} = 5.56 \text{ msec/rev} / 64 \text{ sectors/rev} = 0.0869 \text{ msec} \\
\text{Total Time} = 10.78 \text{ msec} + 0.0869 \text{ msec} = 10.8669 \text{ msec}
\]

If this is the average data transfer rate we have.

\[
4096 \text{ bits/10.8669 msec} = 376,924 \text{ bits/sec or 46.01 kB/sec} \ldots \text{not very fast for “random sectors”}
\]

Fortunately, we usually read multiple sectors at a time (64).

\[
\text{Avg. Access Time} = 8.0 \text{ msec} + (2.78 \text{ msec}) = 10.78 \text{ msec} \\
\text{Transfer time} = 5.56 \text{ msec/rev} / \text{all sectors/rev} = 5.56 \text{ msec} \\
\text{Total Time} = 10.78 \text{ msec} + 5.56 \text{ msec} = 16.34 \text{ msec}
\]

\[
64 \times 4096 \text{ bits/16.34 msec} = 16,043,084 \text{ bits/sec or 2.005 MB/sec} \ldots \text{much better}
\]

We might want to transfer data from hard disk to main memory as pages?
Note: 512 Bytes vs. 32 kBytes shown above. (but nominal pages are 8 kB)

Note the maximum burst transfer rate for 64 sector 10,800 RPM would be

\[
10,800 \text{ RPM} \rightarrow 5.56 \text{ msec/rev} \\
64 \text{ sectors} \rightarrow 5.56/64 = 0.0869 \text{ msec/sector} \\
4096 \text{ bits/sector} \rightarrow 4096/0.0000869 = 47,148,201 \text{ bits/sec} \sim 5.893 \text{ MB/sec}.
\]
Speeding up the transfers:

Disk Controllers: Digital logic that provides the standard interface to the processor.

- Typically includes FIFO memory space for rapid burst transfers.
- May (must) allow simultaneous access to the multiple tracks in a cylinder. (Multiply the burst transfer rate by the number of cylinder read/write heads)
- Performs the ECC generation, testing, and corrections
- Keep track of zones
- Can provide a mapping table of good and bad sectors.
- Transfer data blocks as pages into main memory (1 kB to 64 kB pages)
Floppy Drives (are dead?)

They use the same general concept as a magnetic hard disk, but they are smaller and slower.

Two sizes discussed, 5.25” and 3.5” but those currently used are exclusively HD 3.5”.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>LD 5.25”</th>
<th>HD 5.25”</th>
<th>LD 3.5”</th>
<th>HD 3.5”</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size (inches)</td>
<td>5.25</td>
<td>5.25</td>
<td>3.5</td>
<td>3.5</td>
</tr>
<tr>
<td>Capacity (bytes)</td>
<td>360K</td>
<td>1.2M</td>
<td>720K</td>
<td>1.44M</td>
</tr>
<tr>
<td>Tracks</td>
<td>40</td>
<td>80</td>
<td>80</td>
<td>80</td>
</tr>
<tr>
<td>Sectors/track</td>
<td>9</td>
<td>15</td>
<td>9</td>
<td>18</td>
</tr>
<tr>
<td>Heads</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Rotations/min</td>
<td>300</td>
<td>360</td>
<td>300</td>
<td>300</td>
</tr>
<tr>
<td>Data rate (kbps)</td>
<td>250</td>
<td>500</td>
<td>250</td>
<td>500</td>
</tr>
<tr>
<td>Type</td>
<td>Flexible</td>
<td>Flexible</td>
<td>Rigid</td>
<td>Rigid</td>
</tr>
</tbody>
</table>

The maximum burst transfer rate for 1 sector at 300 RPM would be

300 RPM → 200 msec/rev
18 sectors → 200/18 = 11.11 msec/sector
4096 bits/sector → 4096/0.01111 = 368,640 bits/sec ~ 45.00 kB/sec.
For 2 heads, this could double to 737,280 bits/sec ~ 90.00 kB/sec

Assume an average seek times of approximately 94 msec

The average transfer rate for 18 sectors (one track) at 300 RPM would be

Avg. Access Time = 94 msec + (100 msec) = 194 msec
Transfer time = 200 msec/rev / all sectors/rev = 200 msec
Total Time = 194 msec + 200 msec = 394 msec
18 x 4096 bits/394 msec = 187,126.9 bits/sec or 22.84 kB/sec
For 2 heads, this could double to 45.68 kB/sec
Accessing Drive and Peripherals

The formats in the PC arena
- IDE and the more advanced Extended/Enhanced IDE (EIDE)
- EIDE evolved into Parallel ATA and then Serial ATA
- SCSI

BIOS – Basic Input Output System

ROM based programming code to perform Input and Output. BIOS initially controlled the disk controller card operations.

IDE – Integrated Device Electronics

IDE bus provide register transfers for BIOS, the device electronics connected to the IDE bus but were built into the drives.

BIOS calling conventions (maintained for backward compatibility)
- Addressing for head, cylinder, and sector
  - Heads a 4-bit field numbered starting from 0 (0-15)
  - Cylinders a 10-bit field numbered starting from 0 (0-1023)
  - Sectors a 6-bit field numbered starting from 1 (1-63)

Original BIOS maximum drive size:
- 16 heads * 63 sectors * 1024 cylinders = 1,032,192 sectors
- 512 Bytes/sector \( \Rightarrow \) 528,482,304 Bytes or 528 MB

Disk drives began to remap the “virtual addressing” into the actual addressing/geometry
EIDE – Extended Integrated Device Electronics

Supports a second addressing scheme called Logical Block Addressing (LBA)

Sectors are numbered using a 24-bit integer and are numbered from 0 to $2^{24}$-1.

Drive size:

$$2^{24} \text{ sectors } \times 512 \text{ Bytes/sectors} = 2^{23} \text{ kB} = 2^{13} \text{ MB}$$

$$2^{12} \text{ MB} \Rightarrow 8 \text{ GB} \ldots$$

Typical PC has two EIDE interfaces that each handle two devices …

Guess what! Further extensions exist!

http://www.allensmith.net/Hardware/Storage/HDDlimit/Address.htm

http://computer.howstuffworks.com/hard-disk.htm

The following is from:
http://www.webopedia.com/TERM/A/ATA.html

ATA: Short for Advanced Technology Attachment, a disk drive implementation that integrates the controller on the disk drive itself. There are several versions of ATA, all developed by the Small Form Factor (SFF) Committee:

ATA: Known also as IDE, supports one or two hard drives, a 16-bit interface and PIO modes 0, 1 and 2.

ATA-2: Supports faster PIO modes (3 and 4) and multiword DMA modes (1 and 2). Also supports logical block addressing (LBA) and block transfers. ATA-2 is marketed as Fast ATA and Enhanced IDE (EIDE).

ATA-3: Minor revision to ATA-2.

Ultra-ATA: Also called Ultra-DMA, ATA-33, and DMA-33, supports multiword DMA mode 3 running at 33 MBps.

ATA/66: A version of ATA proposed by Quantum Corporation, and supported by Intel, that doubles ATA's throughput to 66 MBps.

ATA/100: An updated version of ATA/66 that increases data transfer rates to 100 MBps.
Parallel-ATA and Serial-ATA

To get faster transfers at low commodity price costs.

High data throughput parallel cables become expensive: paired grounds, require shielding.

Wider cables with more bits take up lots of connector space and are increasingly hard to “package” in a product (folding or bending to get from point A to B).

At ATAPI-7 there was a break to serial data transfers from parallel and Serial-ATA was born.

http://www.sata-io.org/


“What is SATA?”

“SATA is an evolutionary replacement for the Parallel ATA physical storage interface. SATA is scalable and allows for enhancements to the computing platform. These include easier integration, faster performance, and more efficient design.”

“How will the industry benefit from adopting SATA?”

“Adoption of the Serial ATA specification provides low-cost storage for the industry, improved speed and bandwidth, and serves as an evolutionary replacement for the Parallel ATA interface. Implementation of Serial ATA allows for easy integration due to improved cabling, greater flexibility in regard to system configuration and hot plugability. With these enhancements, system builders can create new solutions with fewer limitations and will experience greater interoperability with other interfaces.”

“What is the long-term road map for SATA?”

SATA defines a roadmap starting at 1.5 gigabits per second (equivalent to a data rate of 150MB/s) and migrating to 3.0 gigabits per second (300 MB/s), evolving to today’s 6.0 gigabits per second transfer rate, and then to even higher data rates. With its widespread usage in the computing industry, the SATA bus architecture will continue to evolve and mature over time. New technologies and techniques in performance, power management, physical configuration, connection options and other storage-centric solutions are constantly being defined and developed, giving SATA a long and promising future.”

SATA is the internal hard disk interface you will find in current PC. Older PC may still use parallel ports. (Look inside the case before buying a drive … it is easy to tell the difference.)
### SCSI – Small Computer System Interface

Similar organization in terms of cylinders, tracks, and sectors

SCSI buses use a different interface with higher data transfer rates.

<table>
<thead>
<tr>
<th>Name</th>
<th>Data bits</th>
<th>Bus MHz</th>
<th>MB/sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCSI-1</td>
<td>8</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Fast SCSI</td>
<td>8</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Wide Fast SCSI</td>
<td>16</td>
<td>10</td>
<td>20</td>
</tr>
<tr>
<td>Ultra SCSI</td>
<td>8</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Wide Ultra SCSI</td>
<td>16</td>
<td>20</td>
<td>40</td>
</tr>
<tr>
<td>Ultra2 SCSI</td>
<td>8</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>Wide Ultra2 SCSI</td>
<td>16</td>
<td>40</td>
<td>80</td>
</tr>
<tr>
<td>Ultra3 SCSI</td>
<td>8</td>
<td>80</td>
<td>80</td>
</tr>
<tr>
<td>Wide Ultra3 SCSI</td>
<td>16</td>
<td>80</td>
<td>160</td>
</tr>
<tr>
<td>Ultra4 SCSI</td>
<td>8</td>
<td>160</td>
<td>160</td>
</tr>
<tr>
<td>Wide Ultra4 SCSI</td>
<td>16</td>
<td>160</td>
<td>320</td>
</tr>
</tbody>
</table>

SCSI ID: 0-7 originally
          0-15 for wide SCSI

Cables/Connectors 50 wire parallel cable
                  25 signal lines, 25 ground lines (signal pairs)

Std. Signal Lines 8-bit data
                  1-bit parity
                  9-bit control
                  7-bit power, misc, spare

Arbitrated bus activity, allows devices to be initiators or targets.

Typically applications would be in servers.
RAID - Redundant Array of Inexpensive/Independent Disks

Use a box full of hard disks to allow:
- Higher data transfer and storage rates (operate disks in parallel),
- Higher degrees of redundancy (save on multiple disks simultaneously), and
- Higher degree of error detection and correction (Additional ECC)

Multiple “Levels” of RAID have been defined to perform these tasks.

RAID typical operates by writing \( k \) sector strips to disks as shown in a process called Striping.

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>RAID 0</td>
<td>Block-level striping</td>
<td>Speed Increase</td>
</tr>
<tr>
<td>RAID 1</td>
<td>Mirroring</td>
<td>Redundancy</td>
</tr>
<tr>
<td>RAID 2</td>
<td>Bit-level striping with parity</td>
<td>Faster with conditional 1 drive failure recovery</td>
</tr>
</tbody>
</table>

Figure 2-23. RAID levels 0 through 2. Backup and parity drives are shown shaded.
Figure 2-23. RAID levels 3 through 5. Backup and parity drives are shown shaded.

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>RAID 3</td>
<td>Byte-level striping with parity</td>
<td>Faster with 1 drive failure recovery</td>
</tr>
<tr>
<td>RAID 4</td>
<td>Block-level striping with parity</td>
<td>Faster with 1 drive failure recovery</td>
</tr>
<tr>
<td>RAID 5</td>
<td>Block-level striping with distributed parity</td>
<td>Faster with 1 drive failure recovery</td>
</tr>
<tr>
<td>RAID 6</td>
<td>Block-level striping with double distributed parity</td>
<td>Faster with 2 drive failure recovery</td>
</tr>
</tbody>
</table>
Optical disks were first invented and marketed by Philips for recording and playing movies and television programs. The original format never took off … Then, Philips in cooperation with Sony developed the Compact Disk (CD) in 1980.

Technical details, the specifications, were published

- Red Book - CD Audio (IS 10149)
- Yellow Book - CD-ROM, CD-ROM/XA
- Green Book - Compact Disc Interactive
- Orange Book - Recordable Compact Disc Standard for both M-O and W-O. Includes Multi-session Discs e.g. Photo CD

**Spiral Groove**
Write information as one continuous spiral (like a vinyl long-play record)

**Pits and Lands**
Data is encoded as the transitions or lack of transitions.
Bit=1 Transitions from pit to land OR transitions from land to pit
Bit=0 No transition of the data.
Physical Specifications:
CD 120 mm across, 1.2 mm thick, 15 mm hole  
Track spacing: 1.6 micrometer (um) (higher capacity 1.5 um)  
Track Width 0.5 um  
Minimum pit size 0.83 um  
Pit Depth 125 nanometer (nm)  
1x Spiral Rate 120 cm/sec  

Note: the CD rotation rate varies to sustain a stable linear read rate at the head!  
1x Read Rate  
  Inside spiral, nominal 530 RPM  
  Outside spiral, nominal 200 RPM  

Audio Format Storage Requirement:  
74 min. Audio: 44,100 samples/channel/second x 2 bytes/sample x 2 channels x 74 minutes x 60 seconds/minute = 783,216,000 bytes  
80 min. Audio: 44,100 samples/channel/second x 2 bytes/sample x 2 channels x 80 minutes x 60 seconds/minute = 846,720,000 bytes
Data Layout of Sectors

Symbols of 14 bits each

42 Symbols make 1 frame

Frames of 588 bits, each containing 24 data bytes

Preamble

Data

ECC

98 Frames make 1 sector

Mode 1 sector (2352 bytes)

Data Layout on CD
Symbol: 14 bits: 8 data bits, 6 ECC
Frame: 42 symbols: 24 data symbols, 18 ECC and control
Sector: 98 frames: producing 2352 Data Bytes

Digital Data Byte Structure (Mode 1)
Preamble 16 Bytes
Data 2048 Byte
ECC 288 Bytes

Audio Data Byte Structure (Mode 2)
Preamble 16 Bytes
Data 2336 Byte

Note: Three distinct error detection and correction schemes
  Data Symbols, Frame, and Sector
  Significant Overhead for error detection and correction

Audio to data capacity ratio: 73/64

Directly Comparing Audio to Data
74 min. Audio → 783,216,000 bytes or Digital Data → 686,655,111 bytes
80 min. Audio → 846,720,000 bytes or Digital Data → 742,329,850 bytes
Operating Rates – 1x speed

75 sectors per second

*Mode 1: (2048 data bytes, Digital Data)*

\[
75 \times 2048 = 153,600 \text{ Bytes/sec} = 150 \text{ kBytes/sec}
\]

Equivalent time of 74 minutes of audio = 4,440 sec

Total Data: \(681,984,000 \text{ Bytes} = 666,000 \text{ kBytes} \sim 650 \text{ MB}\)

*Mode 2: (2336 data bytes, Audio)*

\[
75 \times 2336 = 175,200 \text{ Bytes/sec} \approx 171 \text{ kBytes/sec}
\]

74 minutes of audio = 4,440 sec

Total Data: \(777,888,000 \text{ Bytes} \rightarrow 759,656 \text{ kBytes} \sim 742 \text{ MB}\)

Operating Rates – 52x speed

*Mode 1: (2048 data bytes, Digital Data)*

\[
52 \times 75 \times 2048 = 7,987,200 \text{ Bytes/sec} \rightarrow 7,800 \text{ kB/sec} \sim 7.6 \text{ MB/sec}
\]

Note: a little better than a SCSI-1 !! (not very fast)

*Mode 2: (2336 data bytes, Audio)*

\[
52 \times 75 \times 2336 = 9,110,400 \text{ Bytes/sec} \sim 8,897 \text{ kB/sec} \sim 8.7 \text{ MB/sec}
\]
File Systems for CD Data

Computer companies got together in Lake Tahoe, CA and developed one
**High Sierra** – 3 levels

- **Level 1** 8 character file names (MS-DOS)
  - 8-deep directory w no extensions
  - files all contiguous
  - directory is first on the spiral

- **Level 2** 32 character file names

- **Level 3** non-contiguous files