Memory Hierarchy

- **Overview. Asymmetry.**
- The details:
  - RAM
  - Disks

**An Example Memory Hierarchy**

L5: Remote secondary storage (tapes, distributed file systems, Web servers)
L4: Local secondary storage (local disks)
L3: Main memory (DRAM)
L2: L2 cache (SRAM)
L1: L1 cache (SRAM)
L0: CPU registers

Larger, slower, cheaper per byte
Smaller, faster, costlier per byte

CPU registers hold words retrieved from L1 cache
L1 cache holds cache lines retrieved from L2 cache
L2 cache holds cache lines retrieved from main memory
Main memory holds disk blocks retrieved from local disks
Local disks hold files retrieved from disks on remote network servers

CIS 2107
Memory Hierarchy

- **Overview. Asymmetry.**
- The details:
  - RAM
  - Disks

thank you, CMU
### Pretend for a minute

<table>
<thead>
<tr>
<th>location</th>
<th>access time</th>
<th>location</th>
<th>access time</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1 cache</td>
<td>0.5 ns</td>
<td>L1 cache</td>
<td>1 sec</td>
</tr>
<tr>
<td>L2 cache</td>
<td>7 ns</td>
<td>L2 cache</td>
<td>7 ns</td>
</tr>
<tr>
<td>RAM</td>
<td>100 ns</td>
<td>RAM</td>
<td>100 ns</td>
</tr>
<tr>
<td>hard drive</td>
<td>10 ms</td>
<td>hard drive</td>
<td>10 ms</td>
</tr>
<tr>
<td>DVD</td>
<td>140 ms</td>
<td>DVD</td>
<td>140 ms</td>
</tr>
</tbody>
</table>

If we keep the ratios the same as on the LHS, what are the remaining numbers on the RHS?

### Numbers Everyone Should Know

**Jeff Dean talk at Stanford**

<table>
<thead>
<tr>
<th>operation</th>
<th>access time</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1 cache reference</td>
<td>0.5 ns</td>
</tr>
<tr>
<td>Branch mispredict</td>
<td>5 ns</td>
</tr>
<tr>
<td>L2 cache reference</td>
<td>7 ns</td>
</tr>
<tr>
<td>Mutex lock/unlock</td>
<td>100 ns</td>
</tr>
<tr>
<td>Main memory reference</td>
<td>100 ns</td>
</tr>
<tr>
<td>Compress 1K bytes with Zippy</td>
<td>10,000 ns</td>
</tr>
<tr>
<td>Send 2K bytes over 1 Gbps network</td>
<td>20,000 ns</td>
</tr>
<tr>
<td>Read 1 MB sequentially from memory</td>
<td>250,000 ns</td>
</tr>
<tr>
<td>Round trip within same datacenter</td>
<td>500,000 ns</td>
</tr>
<tr>
<td>Disk seek</td>
<td>10,000,000 ns</td>
</tr>
<tr>
<td>Read 1 MB sequentially from network</td>
<td>10,000,000 ns</td>
</tr>
<tr>
<td>Read 1 MB sequentially from disk</td>
<td>30,000,000 ns</td>
</tr>
<tr>
<td>Send packet CA $\rightarrow$ Netherlands$\rightarrow$CA</td>
<td>150,000,000 ns</td>
</tr>
</tbody>
</table>

### Fake Problem

<table>
<thead>
<tr>
<th>location</th>
<th>access time</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1 cache</td>
<td>0.5 ns</td>
</tr>
<tr>
<td>L2 cache</td>
<td>7 ns</td>
</tr>
<tr>
<td>RAM</td>
<td>100 ns</td>
</tr>
<tr>
<td>hard drive</td>
<td>10 ms</td>
</tr>
<tr>
<td>DVD</td>
<td>140 ms</td>
</tr>
</tbody>
</table>
The moral of the story

- If you’re baking a cake and you have to walk to Nome, AK to get the flour:
  1. get what you need for the rest of the recipe
  2. pick up some sugar

Localidad

- Temporal locality
- Spatial locality

<table>
<thead>
<tr>
<th>Location</th>
<th>Fake Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1 cache</td>
<td>2 feet</td>
</tr>
<tr>
<td>L2 cache</td>
<td>28 feet</td>
</tr>
<tr>
<td>RAM</td>
<td>400 feet</td>
</tr>
<tr>
<td>Hard drive</td>
<td>40,000,000 feet or about 7,500 miles</td>
</tr>
<tr>
<td></td>
<td>about twice the distance from Philly to Nome, AK</td>
</tr>
<tr>
<td>DVD</td>
<td>560,000,000 feet or about 106,000 miles</td>
</tr>
<tr>
<td></td>
<td>or about the distance around the earth 4.25 times</td>
</tr>
</tbody>
</table>

Pretend again. Let’s bake a cake.

if the counter is the CPU and the L1 cache is the cabinet two feet above ...
<table>
<thead>
<tr>
<th><strong>SRAM</strong></th>
<th><strong>DRAM</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>• Fast</td>
<td>• Used in main memory, graphics framebuffer</td>
</tr>
<tr>
<td>• Expensive</td>
<td>• “Stores each bit as a charge on a capacitor”</td>
</tr>
</tbody>
</table>
| • Used in cache memory (on and off chip) | • Sensitive to disturbance (elec. noise, radiation) 
  – Use as digital cam sensor |
| • How much? A few MB | • Leakage: 
  – Needs to be refreshed every 10-100 ms 
  – How? Just read value and write it again |
| • More transistors/circuit | • Slower than SRAM |
| • Stable. Retain value as long as there’s power | • Cheaper than SRAM |
| • No need for refresh | |

<table>
<thead>
<tr>
<th><strong>Memory Hierarchy</strong></th>
<th><strong>Random Access Memory. RAM.</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>• Overview. Asymmetry.</td>
<td>• Random Access?</td>
</tr>
<tr>
<td>• The details:</td>
<td>• SRAM – <em>static</em> RAM</td>
</tr>
<tr>
<td>– RAM</td>
<td>• DRAM – <em>dynamic</em> RAM</td>
</tr>
<tr>
<td>– Disks</td>
<td></td>
</tr>
</tbody>
</table>
Traditional Bus Structure Connecting CPU and Memory

- A bus is a collection of parallel wires that carry address, data, and control signals.
- Buses are typically shared by multiple devices.

SRAM vs DRAM

<table>
<thead>
<tr>
<th></th>
<th>transistors</th>
<th>access time</th>
<th>needs refresh?</th>
<th>cost</th>
<th>where?</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRAM</td>
<td>6</td>
<td>1x</td>
<td>no</td>
<td>100x</td>
<td>cache memory</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(on or off chip)</td>
</tr>
<tr>
<td>DRAM</td>
<td>1</td>
<td>10x</td>
<td>yes</td>
<td>1x</td>
<td>main memory,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>graphics framebuffers</td>
</tr>
</tbody>
</table>

volatile vs. non-volatile storage

- volatile – value lost on power off
- non-volatile –
  - Hard disk
  - “ROMs”, “PROMs”, “firmware”
  - Solid state disks

Memory Read Transaction (1)

- CPU places address A on the memory bus.

Load operation: movl A, %eax
Memory Write Transaction (1)

- CPU places address A on bus. Main memory reads it and waits for the corresponding data word to arrive.

\[ \text{Store operation: movl } \%\text{eax}, A \]

Memory Write Transaction (2)

- CPU places data word y on the bus.

\[ \text{Store operation: movl } \%\text{eax}, A \]

Memory Read Transaction (2)

- Main memory reads A from the memory bus, retrieves word x, and places it on the bus.

\[ \text{Load operation: movl } A, \%\text{eax} \]

Memory Read Transaction (3)

- CPU read word x from the bus and copies it into register \( \%\text{eax} \).

\[ \text{Load operation: movl } A, \%\text{eax} \]
What’s Inside A Disk Drive?

- **Arm**
- **Spindle**
- **Platters**
- **Actuator**
- **SCSI connector**

Electronics (including a processor and memory!)

*Image courtesy of Seagate Technology*

---

Disk Geometry

- Disks consist of **platters**, each with two **surfaces**.
- Each surface consists of concentric rings called **tracks**.
- Each track consists of **sectors** separated by **gaps**.

---

Memory Write Transaction (3)

- Main memory reads data word \( y \) from the bus and stores it at address \( A \).

```
Store operation: movl %eax, A
```

---

Memory Hierarchy

- Overview. Asymmetry.
- The details:
  - RAM
  - Disks
Disk Operation (Single-Platter View)

- The disk surface spins at a fixed rotational rate.
- The read/write head is attached to the end of the arm and flies over the disk surface on a thin cushion of air.
- By moving radially, the arm can position the read/write head over any track.

Rotation rate these days: 5400, 7200 RPM

Disk Geometry (Multiple-Platter View)

- Aligned tracks form a cylinder.

Disk terms

- Capacity – #bits that can be recorded
- Density
  - recording density #bits per 1-inch seg of track.
  - track density #tracks per 1-inch radial segment.
  - areal density (bits/in²) = recording density * track density
- Recording zones – partition of recording space
  - Each track in a zone – same number of sectors
  - Each zone – different number of sectors/track
Disk Structure - top view of single platter

- Surface organized into tracks
- Tracks divided into sectors

Disk Access

- Head in position above a track

from the book ...

- equivalent of Sears Tower on its side:
  - 1 inch above the surface of the Earth
  - each orbit takes 8 seconds
- Speck of dust equivalent to a boulder
- Head crash

Disk Operation (Multi-Platter View)
Disk Access – Read

Rotation is counter-clockwise

After reading blue sector

Red request scheduled next

About to read blue sector
Disk Access – Read

- After BLUE read
- Seek for RED
- Rotational latency
- After RED read

Complete read of red

Disk Access – Service Time Components

- After BLUE read
- Seek for RED
- Rotational latency
- After RED read

- Data transfer
- Seek
- Rotational latency
- Data transfer

Disk Access – Seek

- After BLUE read
- Seek for RED

Seek to red’s track

Disk Access – Rotational Latency

- After BLUE read
- Seek for RED
- Rotational latency

Wait for red sector to rotate around
Reading a Disk Sector (1)

CPU initiates a disk read by writing a command, logical block number, and destination memory address to a port (address) associated with disk controller.

Reading a Disk Sector (2)

Disk controller reads the sector and performs a direct memory access (DMA) transfer into main memory.

Logical Disk Blocks

- Modern disks present a simpler abstract view of the complex sector geometry:
  - The set of available sectors is modeled as a sequence of b-sized logical blocks \(0, 1, 2, \ldots\)
- Mapping between logical blocks and actual (physical) sectors
  - Maintained by hardware/firmware device called disk controller.
  - Converts requests for logical blocks into (surface,track,sector) triples.
- Allows controller to set aside spare cylinders for each zone.
  - Accounts for the difference in “formatted capacity” and “maximum capacity”.

I/O Bus

- Expansion slots for other devices such as network adapters.
SSD Performance Characteristics

<table>
<thead>
<tr>
<th>Access Type</th>
<th>Read Speed (MB/s)</th>
<th>Write Speed (MB/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequential read</td>
<td>250</td>
<td>170</td>
</tr>
<tr>
<td>Random read</td>
<td>140</td>
<td>14</td>
</tr>
<tr>
<td>Random read access</td>
<td>30 us</td>
<td>300 us</td>
</tr>
</tbody>
</table>

- Why are random writes so slow?
  - Erasing a block is slow (around 1 ms)
  - Write to a page triggers a copy of all useful pages in the block
    - Find an used block (new block) and erase it
    - Write the page into the new block
    - Copy other pages from old block to the new block

SSD Tradeoffs vs Rotating Disks

- Advantages
  - No moving parts → faster, less power, more rugged

- Disadvantages
  - Have the potential to wear out
    - Mitigated by “wear leveling logic” in flash translation layer
    - E.g. Intel X25 guarantees 1 petabyte (1015 bytes) of random writes before they wear out
  - In 2010, about 100 times more expensive per byte

- Applications
  - MP3 players, smart phones, laptops
  - Beginning to appear in desktops and servers

---

Reading a Disk Sector (3)

When the DMA transfer completes, the disk controller notifies the CPU with an interrupt (i.e., asserts a special “interrupt” pin on the CPU)

Solid State Disks (SSDs)

- Pages: 512KB to 4KB, Blocks: 32 to 128 pages
- Data read/written in units of pages.
- Page can be written only after its block has been erased
- A block wears out after 100,000 repeated writes.
The CPU-Memory Gap

The gap widens between DRAM, disk, and CPU speeds.

What to do about it

- Disk
- SSD
- DRAM
- CPU

Year:

- Disk seek time
- Flash SSD access time
- DRAM access time
- SRAM access time
- CPU cycle time
- Effective CPU cycle time

- Disk
- SSD
- DRAM
- CPU

- L0: Registers
- L1: Cache (SRAM)
- L2: Cache (SRAM)
- L3: Main memory (DRAM)
- L4: Local secondary storage (local disks)
- L5: Remote secondary storage (tapes, distributed file systems, Web servers)

- CPU registers hold words retrieved from L1 cache
- L1 cache holds cache lines retrieved from L2 cache
- L2 cache holds cache lines retrieved from main memory
- Main memory holds disk blocks retrieved from local disks
- Local disks hold files retrieved from disks on remote network servers