Performance & Technology

Seth Goldstein
CS 740

Sept 3, 2003

Topics:
- Course Info
- Performance measures
- Relating performance measures
- Memory Technology
  - SRAM, DRAM
- Disk Technology

Course Goals
- Understanding computers
  (Not a course on processor design)
- Improve “Use” of computers
  - Getting the most from the system
  - Improving system performance
  - How to build system
- Ability to predict how computers will affect you
  - Understand structure of computer
  - Understand trends that influence computer performance
- Understand abstraction and what lie behind them

Structure of Course
- Lectures
- Textbooks
- Schedule
- Homework
- Papers
- Participation
- Projects

- Course has roughly 2 parts
  - Uniprocessors
  - Multiprocessors

Performance expressed as a time

Absolute time measures
- difference between start and finish of an operation
- synonyms: running time, elapsed time, response time, latency,
  completion time, execution time
- most straightforward performance measure

Relative (normalized) time measures
- running time normalized to some reference time
- (e.g. time/reference time)

Guiding principle: Choose performance measures that track running time.
Performance expressed as a rate

Rates are performance measures expressed in units of work per unit time.

Examples:
- millions of instructions / sec (MIPS)
- millions of floating point instructions / sec (MFLOPS)
- millions of bytes / sec (MBytes/sec)
- millions of bits / sec (Mbits/sec)
- images / sec
- samples / sec
- transactions / sec (TPS)

Key idea: Report rates that track execution time.

Example: Suppose we are measuring a program that convolves a stream of images from a video camera.

Bad performance measure: MFLOPS
- number of floating point operations depends on the particular convolution algorithm: \( n^2 \) matrix-vector product vs \( n \log n \) fast Fourier transform. An FFT with a bad MFLOPS rate may run faster than a matrix-vector product with a good MFLOPS rate.

Good performance measure: images/sec
- a program that runs faster will convolve more images per second.

Fallacy: Peak rates track running time.

Example: the i860 is advertised as having a peak rate of 80 MFLOPS (40 MHz with 2 flops per cycle).

However, the measured performance of some compiled linear algebra kernels (icc -O2) tells a different story:

<table>
<thead>
<tr>
<th>Kernel</th>
<th>1d fft</th>
<th>sasum</th>
<th>saxpy</th>
<th>sdot</th>
<th>sgemm</th>
<th>sgemv</th>
<th>sgemm</th>
<th>sgemm</th>
</tr>
</thead>
<tbody>
<tr>
<td>MFLOPS</td>
<td>8.5</td>
<td>3.2</td>
<td>6.1</td>
<td>10.3</td>
<td>6.2</td>
<td>15.0</td>
<td>8.1</td>
<td></td>
</tr>
<tr>
<td>%peak</td>
<td>11%</td>
<td>4%</td>
<td>7%</td>
<td>13%</td>
<td>8%</td>
<td>19%</td>
<td>10%</td>
<td></td>
</tr>
</tbody>
</table>

Suppose that for some program we have:
- \( T \) seconds running time (the ultimate performance measure)
- \( C \) clock ticks, \( I \) instructions, \( P \) seconds/tick (performance measures of interest to the system designer)

\[
T \text{ secs} = \frac{I \text{ inst}}{I \text{ inst}} \times C \text{ ticks} \times P \text{ secs/tick}
\]

\[
T \text{ secs} = I \text{ inst} \times (C \text{ ticks}/I \text{ inst}) \times P \text{ secs/tick}
\]
Pipeline latency and throughput

Based on slides by TCM

Based on slides by TCM

Video system performance

Based on slides by TCM

Pipelining the video system

Based on slides by TCM

Pipelined video system performance

Based on slides by TCM
Relating time to latency & throughput

In general:
- \( T = L + (N-1)/R \)

The impact of latency and throughput on running time depends on \( N \):
- \( N = 1 \) \( \Rightarrow \) \( T = L \)
- \( N \gg 1 \) \( \Rightarrow \) \( T = N/R \)

To maximize throughput, we should try to maximize the minimum throughput over all stages (i.e., we strive for all stages to have equal throughput).

Amdahl’s law

You plan to visit a friend in Normandy France and must decide whether it is worth it to take the Concorde SST ($3,100) or a 747 ($1,021) from NY to Paris, assuming it will take 4 hours Pgh to NY and 4 hours Paris to Normandy.

<table>
<thead>
<tr>
<th></th>
<th>time NY→Paris</th>
<th>total trip time</th>
<th>speedup over 747</th>
</tr>
</thead>
<tbody>
<tr>
<td>747</td>
<td>8.5 hours</td>
<td>16.5 hours</td>
<td>1</td>
</tr>
<tr>
<td>SST</td>
<td>3.75 hours</td>
<td>11.75 hours</td>
<td>1.4</td>
</tr>
</tbody>
</table>

Taking the SST (which is 2.2 times faster) speeds up the overall trip by only a factor of 1.4!

Amdahl’s law (cont)

Old program (unenhanced)

- \( T_1 \) = time that can NOT be enhanced.
- \( T_2 \) = time that can be enhanced.

Old time: \( T = T_1 + T_2 \)

New program (enhanced)

- \( T'_1 = T_1 \)
- \( T'_2 = T_2 \) <= \( T_2 \)
- \( T' = T'_1 + T'_2 \)

New time: \( T' = T'_1 + T'_2 \)

Speedup: \( S_{\text{overall}} = T / T' \)

Amdahl’s Law:

\[
S_{\text{overall}} = \frac{T}{T'} = \frac{1}{(1-F_{\text{enhanced}}) + F_{\text{enhanced}}/S_{\text{enhanced}}} 
\]

Key idea: Amdahl’s law quantifies the general notion of diminishing returns. It applies to any activity, not just computer programs.
**Amdahl's law (cont)**

Trip example: Suppose that for the New York to Paris leg, we now consider the possibility of taking a rocket ship (15 minutes) or a handy rip in the fabric of space-time (0 minutes):

<table>
<thead>
<tr>
<th></th>
<th>time NY-&gt;Paris</th>
<th>total trip time</th>
<th>speedup over 747</th>
</tr>
</thead>
<tbody>
<tr>
<td>747</td>
<td>8.5 hours</td>
<td>16.5 hours</td>
<td>1</td>
</tr>
<tr>
<td>SST</td>
<td>3.75 hours</td>
<td>11.75 hours</td>
<td>1.4</td>
</tr>
<tr>
<td>rocket</td>
<td>0.25 hours</td>
<td>8.25 hours</td>
<td>2.0</td>
</tr>
<tr>
<td>rip</td>
<td>0.0 hours</td>
<td>8 hours</td>
<td>2.1</td>
</tr>
</tbody>
</table>

**Useful corollary to Amdahl’s law:**

\[
1 \leq S_{\text{overall}} \leq \frac{1}{1 - F_{\text{enhanced}}}
\]

<table>
<thead>
<tr>
<th>( F_{\text{enhanced}} )</th>
<th>Max ( S_{\text{overall}} )</th>
<th>( F_{\text{enhanced}} )</th>
<th>Max ( S_{\text{overall}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>1</td>
<td>0.9375</td>
<td>16</td>
</tr>
<tr>
<td>0.5</td>
<td>2</td>
<td>0.96875</td>
<td>32</td>
</tr>
<tr>
<td>0.75</td>
<td>4</td>
<td>0.984375</td>
<td>64</td>
</tr>
<tr>
<td>0.875</td>
<td>8</td>
<td>0.9921875</td>
<td>128</td>
</tr>
</tbody>
</table>

Moral: It is hard to speed up a program.

Moral++: It is easy to make premature optimizations.

---

**Computer System**

Levels in Memory Hierarchy

- **CPU**: 8 B, 32 KB/4 MB, 128 MB, 20 GB
- **Cache**: 32 B, 8 KB
- **Memory**: 8 B, 32 B, 8 KB
- **Disk Memory**: 20 GB

Size: 200 B, 32 KB/4 MB, 128 MB, 20 GB

Speed: 3 ns, 6 ns, 60 ns, 8 ms

$/\text{Mbyte}$: $100/MB, $0.30/MB, $0.005/MB

Block size: 8 B, 32 B, 8 KB

Larger, slower, cheaper

Based on slides by TCM
## Scaling to 0.1µm

- **Semiconductor Industry Association, 1992 Technology Workshop**
  - Projected future technology based on past trends

<table>
<thead>
<tr>
<th>Year</th>
<th>Feature size</th>
<th>DRAM capacity</th>
<th>Chip area (cm²)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1992</td>
<td>0.5</td>
<td>16M</td>
<td>2.5</td>
</tr>
<tr>
<td>1995</td>
<td>0.35</td>
<td>64M</td>
<td>4.0</td>
</tr>
<tr>
<td>1998</td>
<td>0.25</td>
<td>256M</td>
<td>6.0</td>
</tr>
<tr>
<td>2001</td>
<td>0.18</td>
<td>1G</td>
<td>8.0</td>
</tr>
<tr>
<td>2004</td>
<td>0.12</td>
<td>4G</td>
<td>10.0</td>
</tr>
<tr>
<td>2007</td>
<td>0.10</td>
<td>16G</td>
<td>12.5</td>
</tr>
</tbody>
</table>

- **Industry is slightly ahead of projection**
- **DRAM capacity**: Doubles every 1.5 years
- **Prediction on track**
- **Chip area (cm²)**: Way off! Chips staying small

---

## More on Moore’s Law

You can buy this for $188 today.

In 1983 dollars, the equivalent
- cost >$1500.00
- Fit in 5 boxes
NRE and Mask Costs

<table>
<thead>
<tr>
<th>SIA Roadmap Generation</th>
<th>Wafer Exposures/Mask</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.35 um</td>
<td>3000</td>
</tr>
<tr>
<td>0.25 um</td>
<td>500</td>
</tr>
<tr>
<td>0.18 um</td>
<td>500</td>
</tr>
<tr>
<td>0.13 um</td>
<td>500</td>
</tr>
</tbody>
</table>

Affordable Total Cost / Wafer Level Exposure

Power?

- Power densities too high to keep junctions at low temps

Technology Shifts

- Size of Devices
  ⇒ Inches to Microns to Nanometers
- Type of Interconnect
  ⇒ Rods to Lithowires to Nanowires
- Method of Fabrication
  ⇒ Hammers to Light to Self-Assembly
- Largest Sustainable System
  ⇒ $10^1$ to $10^8$ to $10^{12}$
- Reliability
  ⇒ Bad to Excellent to Unknown

What Comes Next?

- Combination of Hans Moravac + Larry Roberts + Gordon Bell
  WordSize*ops/sysprice

From Gray Turing Award Lecture
**Another Moore’s Law**

![Graph showing Moore's Law](image)

**Static RAM (SRAM)**

- **Fast**
  - ~4 nsec access time
- **Persistent**
  - As long as power is supplied
  - No refresh required
- **Expensive**
  - ~$100/MByte
  - 6 transistors/bit
- **Stable**
  - High immunity to noise and environmental disturbances
- **Technology for caches**

**Anatomy of an SRAM Cell**

![SRAM Cell Diagram](image)

**SRAM Cell Principle**

**Inverter Amplifies**
- Negative gain
- Slope < -1 in middle
- Saturates at ends

**Inverter Pair Amplifies**
- Positive gain
- Slope > 1 in middle
- Saturates at ends

**Terminology**

- Bit line: carries data
- Word line: used for addressing

**Write:**
1. Set bit lines to new data value
2. Raise word line to "high"
3. Sets cell to new state (may involve flipping relative to old state)

**Read:**
1. Set bit lines high
2. Set word line high
3. See which bit line goes low
Bistable Element

**Stability**
- Require $V_{in} = V_2$
- Stable at endpoints
  - recover from perturbation
- Metastable in middle
  - Fall out when perturbed

**Ball on Ramp Analogy**

Example SRAM Configuration (16 x 8)

Dynamic RAM (DRAM)

**Slower than SRAM**
- access time ~60 nsec

**Nonpersistant**
- every row must be accessed every ~1 ms (refreshed)

**Cheaper than SRAM**
- ~$0.50 / MByte
- 1 transistor/bit

**Fragile**
- electrical noise, light, radiation

Workhorse memory technology

Anatomy of a DRAM Cell

**Writing**

**Reading**

$\Delta V \sim \frac{C_{node}}{C_{BL}}$
### Addressing Arrays with Bits

**Array Size**
- R rows, $R = 2^r$
- C columns, $C = 2^c$
- $N = R \times C$ bits of memory

**Addressing**
- Addresses are $n$ bits, where $N = 2^n$
- $\text{row(address)} = \text{address} / C$
  - leftmost $r$ bits of address
- $\text{col(address)} = \text{address} \mod C$
  - rightmost bits of address

**Example**
- $R = 2$
- $C = 4$
- address = 6

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>000</td>
<td>001</td>
<td>010</td>
<td>011</td>
</tr>
<tr>
<td>1</td>
<td>100</td>
<td>101</td>
<td>110</td>
<td>111</td>
</tr>
</tbody>
</table>

### Example 2-Level Decode DRAM (64Kx1)

- Provide 16-bit address in two 8-bit chunks

### DRAM Operation

**Row Address (~50ns)**
- Set Row address on address lines & strobe RAS
- Entire row read & stored in column latches
- Contents of row of memory cells destroyed

**Column Address (~10ns)**
- Set Column address on address lines & strobe CAS
- Access selected bit
  - READ: transfer from selected column latch to Dout
  - WRITE: Set selected column latch to Din

**Rewrite (~30ns)**
- Write back entire row

### Observations About DRAMs

**Timing**
- Access time ($\approx 60\text{ns}$) < cycle time ($\approx 90\text{ns}$)
- Need to rewrite row

**Must Refresh Periodically**
- Perform complete memory cycle for each row
- Approximately once every 1ms
- $\sqrt{n}$ cycles
- Handled in background by memory controller

**Inefficient Way to Get a Single Bit**
- Effectively read entire row of $\sqrt{n}$ bits
Enhanced Performance DRAMs

Conventional Access
- Row + Col
- RAS CAS RAS CAS ...

Page Mode
- Row + Series of columns
- RAS CAS CAS CAS ...
- Gives successive bits

Other Acronyms
- EDORAM - "Extended data output"
- SDRAM - "Synchronous DRAM"

Typical Performance

<table>
<thead>
<tr>
<th></th>
<th>Row Access Time</th>
<th>Col Access Time</th>
<th>Cycle Time</th>
<th>Page Mode Cycle Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50ns</td>
<td>10ns</td>
<td>90ns</td>
<td>25ns</td>
</tr>
</tbody>
</table>

Video RAM

Performance Enhanced for Video / Graphics Operations
- Frame buffer to hold graphics image
- Writing
  - Random access of bits
  - Also supports rectangle fill operations
  - Set all bits in region to 0 or 1
- Reading
  - Load entire row into shift register
  - Shift out at video rates

Performance Example
- 1200 X 1800 pixels / frame
- 24 bits / pixel
- 60 frames / second
- 2.8 GBits / second

DRAM Driving Forces

Capacity
- 4X per generation
  - Square array of cells
- Typical scaling
  - Lithography dimensions 0.7X
    » Areal density 2X
  - Cell function packing 1.5X
  - Chip area 1.33X
- Scaling challenge
  - Typically $C_{\text{node}} / C_{BL} = 0.1-0.2$
  - Must keep $C_{\text{node}}$ high as shrink cell size

Retention Time
- Typically 16-256 ms
- Want higher for low-power applications

DRAM Storage Capacitor

Planar Capacitor
- Up to 1Mb
- $C$ decreases linearly with feature size

Trench Capacitor
- 4-256 Mb
- Lining of hole in substrate

Stacked Cell
- > 1Gb
- On top of substrate
- Use high $\varepsilon$ dielectric

Drum Capacitor
- Planar Capacitor
  - Up to 1Mb
  - $C$ decreases linearly with feature size

Trench Capacitor
- 4-256 Mb
- Lining of hole in substrate

Stacked Cell
- > 1Gb
- On top of substrate
- Use high $\varepsilon$ dielectric
**Trench Capacitor**

**Process**
- Etch deep hole in substrate
  - Becomes reference plate
- Grow oxide on walls
  - Dielectric
- Fill with polysilicon plug
  - Tied to storage node

**IBM DRAM Evolution**

- IBM J. R&D, Jan/Mar '95
- Evolution from 4 - 256 Mb
- 256 Mb uses cell with area 0.6 \(\mu\)m\(^2\)

**4 Mb Cell Structure**

**Mitsubishi Stacked Cell DRAM**

- IEDM '95
- Claim suitable for 1 - 4 Gb

**Technology**
- 0.14 \(\mu\)m process
  - Synchrotron X-ray source
- 8 nm gate oxide
- 0.29 \(\mu\)m\(^2\) cell

**Storage Capacitor**
- Fabricated on top of everything else
- Rubidium electrodes
- High dielectric insulator
  - 50X higher than SiO\(_2\)
  - 25 nm thick
- Cell capacitance 25 femtofarads

**Molecular Memory**

10X mag each time:
- A - Wafer (625 test structures)
- B - Memories with test structures
- C - Single test structure
- D - Again, can see nanowires
- E - The cross bar
- F - and again
**Molecular Memory**

- Uses new principals (not charge, but conformation)
- Features sizes of <1nm
- Pitch of <3nm
- Non-volatile
- Inexpensive (due to self-assembly)

**Magnetic Disks**

The read/write head floats over the disk surface and moves back and forth on an arm from track to track.

- Disk surface spins at 3600–7200 RPM
- The surface consists of a set of concentric magnetized rings called tracks
- Each track is divided into sectors

**Disk Capacity**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>18GB Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number Platters</td>
<td>12</td>
</tr>
<tr>
<td>Surfaces / Platter</td>
<td>2</td>
</tr>
<tr>
<td>Number of tracks</td>
<td>6962</td>
</tr>
<tr>
<td>Number sectors / track</td>
<td>213</td>
</tr>
<tr>
<td>Bytes / sector</td>
<td>512</td>
</tr>
<tr>
<td>Total Bytes</td>
<td>18,221,948,928</td>
</tr>
</tbody>
</table>

**Disk Operation**

**Operation**
- Read or write complete sector

**Seek**
- Position head over proper track
- Typically 6-9ms

**Rotational Latency**
- Wait until desired sector passes under head
- Worst case: complete rotation
  - 10,025 RPM ⇒ 6 ms

**Read or Write Bits**
- Transfer rate depends on # bits per track and rotational speed
- E.g., 213 * 512 bytes @10,025 RPM = 18 MB/sec.
- Modern disks have external transfer rates of up to 80 MB/sec
- DRAM caches on disk help sustain these higher rates
### Disk Performance

**Getting First Byte**
- Seek + Rotational latency = 7,000 - 19,000 µsec

**Getting Successive Bytes**
- ~ 0.06 µsec each
  - roughly 100,000 times faster than getting the first byte!

**Optimizing Performance:**
- Large block transfers are more efficient
- Try to do other things while waiting for first byte
  - switch context to other computing task
  - processor is interrupted when transfer completes

### Disk / System Interface

1. **Processor Signals Controller**
   - Read sector X and store starting at memory address Y

2. **Read Occurs**
   - "Direct Memory Access" (DMA) transfer
   - Under control of I/O controller

3. **I / O Controller Signals Completion**
   - Interrupts processor
   - Can resume suspended process

### Magnetic Disk Technology

**Seagate ST-12550N Barracuda 2 Disk**
- Linear density 52,187. bits per inch (BPI)
  - Bit spacing 0.5 microns
- Track density 3,047. tracks per inch (TPI)
  - Track spacing 8.3 microns
- Total tracks 2,707. tracks
- Rotational Speed 7200. RPM
- Avg Linear Speed 86.4 kilometers / hour
- Head Floating Height 0.13 microns

**Analogy:**
- put the Sears Tower on its side
- fly it around the world, 2.5cm above the ground
- each complete orbit of the earth takes 8 seconds

### CD Read Only Memory (CDROM)

**Basis**
- Optical recording technology developed for audio CDs
  - 74 minutes playing time
  - 44,100 samples / second
  - 2 X 16-bits / sample (Stereo)
  - Raw bit rate = 172 KB / second
- Add extra 288 bytes of error correction for every 2048 bytes of data
  - Cannot tolerate any errors in digital data, whereas OK for audio

**Bit Rate**
- $172 \times \frac{2048}{288 + 2048} = 150$ KB / second
  - For 1X CDROM
  - $N \times$ CDROM gives bit rate of $N \times 150$
  - E.g., 12X CDROM gives 1.76 MB / second

**Capacity**
- 74 Minutes $\times$ 150 KB / second $\times$ 60 seconds / minute $= 650$ MB
Storage Trends

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SRAM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$/MB</td>
<td>19,200</td>
<td>2,900</td>
<td>320</td>
<td>256</td>
<td>100</td>
<td>190</td>
</tr>
<tr>
<td>access (ns)</td>
<td>300</td>
<td>150</td>
<td>35</td>
<td>15</td>
<td>2</td>
<td>100</td>
</tr>
<tr>
<td>DRAM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$/MB</td>
<td>8,000</td>
<td>880</td>
<td>100</td>
<td>30</td>
<td>1.5</td>
<td>5,300</td>
</tr>
<tr>
<td>access (ns)</td>
<td>375</td>
<td>200</td>
<td>100</td>
<td>70</td>
<td>60</td>
<td>6</td>
</tr>
<tr>
<td>typical size (MB)</td>
<td>0.064</td>
<td>0.256</td>
<td>4</td>
<td>16</td>
<td>64</td>
<td>1,000</td>
</tr>
<tr>
<td>Disk</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$/MB</td>
<td>500</td>
<td>100</td>
<td>8</td>
<td>0.30</td>
<td>0.05</td>
<td>10,000</td>
</tr>
<tr>
<td>access (ms)</td>
<td>87</td>
<td>75</td>
<td>28</td>
<td>10</td>
<td>8</td>
<td>11</td>
</tr>
<tr>
<td>typical size (MB)</td>
<td>1</td>
<td>10</td>
<td>160</td>
<td>1,000</td>
<td>9,000</td>
<td>9,000</td>
</tr>
</tbody>
</table>

Based on back issues of Byte and PC Magazine

Storage Price: $/MByte

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SRAM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DRAM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Disk</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Processor clock rates

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SRAM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DRAM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Disk</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Based on back issues of Byte and PC Magazine
The CPU vs. DRAM Latency Gap (ns)

Memory Technology Summary

Cost and Density Improving at Enormous Rates
Speed Lagging Processor Performance

Memory Hierarchies Help Narrow the Gap:
- Small fast SRAMS (cache) at upper levels
- Large slow DRAMS (main memory) at lower levels
- Incredibly large & slow disks to back it all up

Locality of Reference Makes It All Work
- Keep most frequently accessed data in fastest memory

System Cost

- Performance is not sole metric
- Performance/$ is a better metric for some applications
- Chip cost is still/now single most important cost
- Cost/mm² has remained basically constant (in volume!)
- NRE is soaring
- What does this mean for future computing systems?
  - Application space
  - System design
  - Programming methodology

Monday

- Start looking at ISA
- Assignment 1 handed out
- Alpha accounts ready
- Web site up