Last Time

- What’s inside my computer?
- Operating systems & systems software
- Compilation
- Introduction to C
Performance

• When we say “Computer A has better performance than Computer B,” what exactly are we describing?
• What are different ways of measuring the performance of computers?
• Is there any definitive way to state that one computer is better than another?
Moore’s Law

• “The number of transistors in a dense integrated circuit doubles approximately every two years.”

• “Chip performance doubles every 18 months.”
Defining Performance

- Think about airplanes:
  - Is performance defined in terms of speed?
  - Is it defined in terms of # of passengers carried?
  - Is it defined in terms of maximum distance?

<table>
<thead>
<tr>
<th>Airplane</th>
<th>Passenger Capacity</th>
<th>Cruising Range (miles)</th>
<th>Cruising Speed (MPH)</th>
<th>Passenger Throughput (passengers * MPH)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boeing 777</td>
<td>375</td>
<td>4630</td>
<td>610</td>
<td>228750</td>
</tr>
<tr>
<td>Boeing 747</td>
<td>470</td>
<td>4150</td>
<td>610</td>
<td>286700</td>
</tr>
<tr>
<td>BAD/Sud Concorde</td>
<td>132</td>
<td>4000</td>
<td>1350</td>
<td>178200</td>
</tr>
<tr>
<td>Douglas DC-8-50</td>
<td>146</td>
<td>8720</td>
<td>544</td>
<td>79424</td>
</tr>
</tbody>
</table>
Defining Performance

• **Response Time** – “The total time required for the computer to complete a task, including disk accesses, memory accesses, I/O activities, operating system overhead, CPU execution time, and so on.”

• **Throughput** – “The number of tasks completed per unit time.”

Smaller response time = Larger throughput
Defining Performance (with Math)

Performance = \frac{1}{\text{Execution Time}}

To maximize performance...

...we want to minimize response time (or execution time) for some task
Comparing Performance

Given two computers X and Y, if the performance of X is greater than the performance of Y, then...

\[ \text{Performance}_x > \text{Performance}_y \]

\[ \frac{1}{\text{ExecutionTime}_x} > \frac{1}{\text{ExecutionTime}_y} \]

\[ \text{ExecutionTime}_x < \text{ExecutionTime}_y \]

... the execution time of X is less than the execution time of Y!
Comparing Performance

\[
\frac{\text{Performance}_x}{\text{Performance}_y} = \frac{\text{ExecutionTime}_y}{\text{ExecutionTime}_x} = n
\]

“X is \(n\) times faster than Y.”

“The execution time on Y is \(n\) times longer than it is on X.”
Example

• If computer A runs a program in 10 seconds, and computer B runs the same program in 15 seconds, how much faster is A than B?

\[
\frac{\text{Execution Time}_y}{\text{Execution Time}_x} = n
\]

\[
\frac{15}{10} = 1.5
\]

• Computer A is 1.5 times faster than computer B.
Measuring Performance

- **Elapsed time** – “The difference in time between the beginning and ending of program execution.” (Also called response time, wall clock time)

- Problems:
  - Computers are shared and the CPU could switch to another program.
  - Program could wait longer for input because a user isn’t paying attention.
Measuring Performance

• **CPU execution time** – “The amount of time the CPU spends computing for a task.”
  – Doesn’t include time waiting for other programs
  – Doesn’t include time waiting for I/O
  – Can be further divided into user CPU time and system CPU time (tough to do)
OK, Now How About “Time”...

- **Clock cycle** – “The operation of one CPU instruction.” (Also called tick, clock tick, cycle)
- **Clock period** – “The length of each clock cycle.”
- **Clock rate** – “The number of clock cycles in one wall clock second.”
OK, Now How About “Time”...

- **Clock cycle** – “The operation of one CPU instruction.” (Also called tick, clock tick, cycle)
- **Clock period** – “The length of each clock cycle.”
- **Clock rate** – “The number of clock cycles in one wall clock second.”

- Example:
  - Clock rate = 4 GHz (4 billion cycles/second)
  - Clock period = \( \frac{1}{4 \times 10^9} = 250 \text{ ps} \)
CPU Performance and Factors

CPU execution time for a program = CPU clock cycles needed × Clock cycle time

Because block rate and clock cycle time are inverses...

CPU execution time for a program = CPU clock cycles needed
Clock rate
Example

A program runs in 10 seconds on computer A, which has a 2 GHz clock. We are trying to help a computer designer build computer B, which will run this program in 6 seconds. The designer has determined that a substantial increase in the clock rate is possible, but this increase will affect the rest of the CPU design, causing computer B to require 1.2 times as many clock cycles as computer A for this program. What clock rate should we tell the designer to target?
Example

CPU Time_A = \frac{\text{CPU Clock Cycles}_A}{\text{Clock Rate}_A}

10 seconds = \frac{\text{CPU Clock Cycles}_A}{2 \times 10^9 \text{ cycles/sec}}

Cycles_A = 10 \text{ sec} \times 2 \times 10^9 \text{ cycles/sec}

CPU Clock Cycles_A = 2 \times 10^{10} \text{ cycles}

CPU Time_B = \frac{1.2 \times \text{CPU Clock Cycles}_A}{\text{Clock Rate}_B}

6 seconds = \frac{1.2 \times 2 \times 10^{10} \text{ cycles/sec}}{\text{Clock Rate}_B}

Clock Rate_B = \frac{1.2 \times 2 \times 10^{10} \text{ cycles/sec}}{6 \text{ seconds}}

Clock Rate_B = 0.2 \times 2 \times 10^{10} \text{ cycles/sec}

Clock Rate_B = 4 \times 10^9 \text{ cycles/sec}

Clock Rate_B = 4 \text{ GHz}
Instruction Performance

• To this point, we’ve been assuming that each instruction takes 1 CPU cycle to compute... so what if we’re wrong?

• **Clock cycles per instruction** – “The average number of clock cycles each instruction takes to execute.” (Abbreviated CPI)
Classic CPU Performance Equation

\[ \text{CPU Time} = \frac{\text{Instruction Count}}{\text{Clock Rate}} \times \text{CPI} \times \text{Clock Cycle Time} \]

Since the clock rate is the inverse of clock cycle time...

\[ \text{CPU Time} = \text{Instruction count} \times \text{CPI} \times \frac{1}{\text{Clock rate}} \]
Example

Suppose we have two implementations of the same instruction set architecture. Computer A has a clock cycle time on 250ps and a CPI of 2.0 for some program, and computer B has a clock cycle time of 500ps and a CPI of 1.2 for the same program. Which computer is faster for this program, and by how much?
Example

**Computer A**

CPU Clock Cycles_A = I × 2.0

CPU Time_A = CPU Clock Cycles_A × Clock cycle Time

CPU Time_A = I × 2.0 × 250ps
= 500 × I ps

Performance_A = ExecutionTime_B / ExecutionTime_A
= 600 × I ps / 500 × I ps
= 1.2

**Computer B**

CPU Clock Cycles_B = I × 1.2

CPU Time_B = CPU Clock Cycles_B × Clock cycle Time

CPU Time_B = I × 1.2 × 500ps
= 600 × I ps

Computer A is 1.2 times faster than computer B.
# Measuring Basic Components of Performance

<table>
<thead>
<tr>
<th>Components of Performance</th>
<th>Units of Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU execution time for a program</td>
<td>Seconds for the program to run</td>
</tr>
<tr>
<td>Instruction count</td>
<td>Number of instructions executed while the program is running</td>
</tr>
<tr>
<td>Clock cycles per instruction (CPI)</td>
<td>Average number of clock cycles per instruction</td>
</tr>
<tr>
<td>Clock cycle time</td>
<td>Seconds per clock cycle</td>
</tr>
</tbody>
</table>
So, Moore’s Law Will Continue on Forever, Right?
So Moore’s Law Will Continue on Forever, Right?
Dynamic Power Equation

\[
\text{Power} = \text{Capacitive Load} \times \text{Voltage}^2 \times \text{Frequency Switched}
\]

Suppose we develop a new processor that has 85% of the capacitive load of a more complex old processor. Further, assume that the new processor has reduced voltage by 15%, which results in a 15% shrink in frequency. What is the impact on dynamic power?

\[
\frac{\text{Power}_{\text{new}}}{\text{Power}_{\text{old}}} = \frac{(\text{CapLoad} \times .85) \times (\text{Voltage} \times .85)^2 \times (\text{FreqSwitch} \times .85)}{\text{CapLoad} \times \text{Voltage}^2 \times \text{FreqSwitch}}
\]

\[
= 0.85^4 = 0.52
\]
So How Do We Get Around This Wall?

- Tune in next time!
Material Credits

- HP sections 1.4, 1.5
- “Moore’s Law” (https://en.wikipedia.org/wiki/Moore%27s_law)

Image Credits

- Slide 13: Processor Clock Cycle (HP section 1.4)
- Slide 23: Clock Rate and Power (HP section 1.5)
- Slide 24: Moore’s Law Continuing (HP section 1.5)
- Slide 26: Batman ending (http://antonynbritt.files.wordpress.com/2013/08/aug-18-same-bat-time-same-bat-channel.jpg)