# 100
# *Performance*

ENGR 3410 – Computer Architecture

Mark L. Chang

Fall 2008

# What things are important when buying a computer?

- What features do you look for when buying one?

# Computer "Performance"

- MIPS (Million Instructions Per Second) vs. MHz (Million Cycles Per Second)
- Throughput (jobs/seconds) vs. Latency (time to complete a job)
- Measuring, Metrics, Evaluation – what is "best"?

3.09 GHz
Pentium 4

Hyper
Pipelined
Technology

The PowerBook G4 outguns Pentium III-based notebooks by up to 30 percent.*

* Based on Adobe Photoshop tests comparing a 500MHz PowerBook G4 to 850MHz Pentium III-based portable computers

# Performance Example: Planes

| Airplane | Passenger Capacity | Cruising Range (miles) | Cruising Speed (mph) | Passenger Throughput (passengermile/hour) |
|---|---|---|---|---|
| Boeing 777 | 375 | 4630 | 610 | 228,750 |
| Boeing 747 | 470 | 4150 | 610 | 286,700 |
| Concorde | 132 | 4000 | 1350 | 178,200 |
| Douglas DC8 | 146 | 8720 | 544 | 79,424 |

- Which is the “best” plane?
  - Which gets one passenger to the destination first?
  - Which moves the most passengers?
  - Which goes the furthest?

- Which is the “speediest” plane (between Seattle and NY)?
  - Latency: how fast is one person moved?
  - Throughput: number of people per time moved?

# Computer Performance

- Primary goal: execution time (time from program start to program completion)

$$Performance = \frac{1}{ExecutionTime}$$

- To compare machines, we say "X is n times faster than Y"

$$n = \frac{Performance_x}{Performance_y} = \frac{ExecutionTime_y}{ExecutionTime_x}$$

- Example: Machine *Orange* and *Grape* run a program
  Orange takes 5 seconds, Grape takes 10 seconds

- Orange is __2__ times faster than Grape

# Execution Time

- Elapsed Time
  - counts everything *(disk and memory accesses, I/O , etc.)*
  - a useful number, but often not good for comparison purposes
- CPU time
  - doesn't count I/O or time spent running other programs
  - can be broken up into system time, and user time
- Example: Unix “time” command

```
fpga.olin.edu> time javac CircuitViewer.java
3.370u 0.570s 0:12.44 31.6%
```

- Our focus: user CPU time
  - time spent executing the lines of code that are "in" our program

# CPU Time

$$\text{CPU execution time for a program} = \text{CPU clock cycles for a program} * \text{Clock period}$$

$$\text{CPU execution time for a program} = \text{CPU clock cycles for a program} * \frac{1}{\text{Clock rate}}$$

- Application example:

A program takes 10 seconds on computer *Orange*, with a 400MHz clock. Our design team is developing a machine *Grape* with a much higher clock rate, but it will require 1.2 times as many clock cycles. If we want to be able to run the program in 6 second, how fast must the clock rate be?

ORANGE: $10s = \frac{N \text{ cycles}}{400 \times 10^6 \text{ cycles}} = 4000 \times 10^6 \text{ cycles}$

GRAPE: $6s = \frac{1.2 \times 4000 \times 10^6 \text{ cycles}}{R}$

$R = 800 \text{ MHz}$

# CPI

- How do the # of instructions in a program relate to the execution time?

$$\text{CPU clock cycles for a program} = \text{Instructions for a program} * \text{Average Clock Cycles per Instruction } \mathbf{(CPI)}$$

$$\text{CPU execution time for a program} = \text{Instructions for a program} * \text{CPI} * \frac{1}{\text{Clock rate}}$$

# CPI Example

- Suppose we have two implementations of the same instruction set (ISA).
- For some program

  Machine A has a clock cycle time of 10 ns. and a CPI of 2.0
  Machine B has a clock cycle time of 20 ns. and a CPI of 1.2

- What machine is faster for this program, and by how much?

$\text{CPU Clock cycles}_A = I \times 2.0$

$\text{"}\qquad\qquad _B = I \times 1.2$

$\text{CPU Time}_A = I \times 2.0 \times 10ns = 20 \times I \; ns$

$\qquad\qquad _B = I \times 1.2 \times 20ns = 24 \times I \; ns$

A faster than B $\frac{24}{20} = \boxed{1.2}$

# Computing CPI

- Different types of instructions can take very different amounts of cycles
- Memory accesses, integer math, floating point, control flow

$$CPI = \sum_{types} \left( Cycles_{type} * Frequency_{type} \right)$$

| Instruction Type | Type Cycles | Type Frequency | Cycles * Freq |
|---|---|---|---|
| ALU | 1 | 50% | 0.5 |
| Load | 5 | 20% | 1.0 |
| Store | 3 | 10% | 0.3 |
| Branch | 2 | 20% | 0.4 |
| | | CPI: | 2.2 |

# CPI & Processor Tradeoffs

| Instruction Type | Type Cycles | Type Frequency |
|---|---|---|
| ALU | 1 | 50% |
| Load | 5 | 20% |
| Store | 3 | 10% |
| Branch | 2 | 20% |

*How much faster would the machine be if:*

1. A data cache reduced the average load time to 2 cycles?

   ORIG: 5x.2 ↕ 0.6 ⇒ 2.2 / 1.6 = 1.375x
   NEW: 2x.2

2. Branch prediction shaved a cycle off the branch time?

   0.4 ⇒ 0.2 2.2 / 2 = 1.1x

3. Two ALU instructions could be executed at once?

   1x.5 ⇒ 0.5x.5 → 2.2 / 1.8 = 1.22x

# Warning 1: Amdahl's Law

- The impact of a performance improvement is limited by what is NOT improved:

$$\text{Execution time after improvement} = \text{Execution time of unaffected} + \text{Execution time affected} * \frac{1}{\text{Amount of improvement}}$$

- Example: Assume a program runs in 100 seconds on a machine, with multiply responsible for 80 seconds of this time. How much do we have to speed up multiply to make the program run 4 times faster?

$25s$

$$25s = 20s + 80s \times \frac{1}{N}$$

$$N = 80/5 = 16$$

- 5 times faster?

$20s$

$$20s = 20s + 80s \times \frac{1}{N}$$

# Warning 2: MIPs, MHz ≠ Performance

- Higher MHz (clock rate) doesn't always mean better CPU

  *Orangе computer: 1000 MHz, CPI: 2.5, 1 billion instruction program*

  $T = \frac{1B \times 2.5}{1000} = 2.5s$

  *Grape computer: 500MHz, CPI: 1.1, 1 billion instruction program*

  $T = \frac{1B \times 1.1}{500} = 2.2s$

- Higher MIPs (million instructions per second) doesn't always mean better CPU

  1 MHz machine, with two different compilers

  Compiler A on program X: 10M ALU, 1M Load

  Compiler B on program X: 5M ALU, 1M Load

  A: 10M + 1.5M = 15M

  B: 5M + 1.5M = 10M

Execution Time: A 15s B 10s

MIPS: A ~~11~~ /15 B 6/10

.  .6

| Instruction Type | Type Cycles |
| --- | --- |
| ALU | 1 |
| Load | 5 |
| Store | 3 |
| Branch | 2 |

# Processor Performance Summary

- Machine performance:

$$\text{CPU execution time for a program} = \text{Instructions for a program} * \text{CPI} * \frac{1}{\text{Clock rate}}$$

- Better performance:

  ______ number of instructions to implement computations

  ______ CPI

  ______ Clock rate

- Improving performance must balance each constraint
  Example: RISC vs. CISC