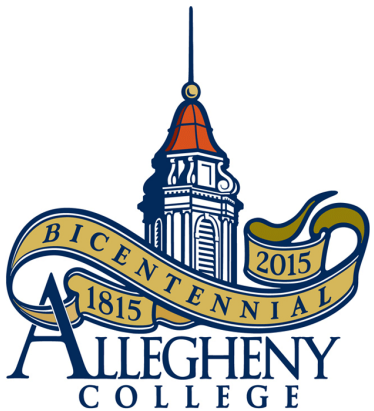
CS200 - Computer Organization

Performance Metrics - Part 2

Aravind Mohan

Allegheny College

September 23, 2021



So far:

- Reviewed basic performance metrics
- Discussed techniques to compare machines at the hardware-level.

Quick follow up

- computer A has:
  1. 4GHz clock rate
  2. 8s CPU time
- computer B:
  1. 6s CPU time
  2. Number of clock cycles is 1.4 times as much as the number of clock cycles of Computer A.
- computer C:
  1. 4s CPU time
  2. Number of clock cycles is 1.8 times as much as the number of clock cycles of Computer B.

- So what is the clock rate of computer C?? and which among the three computers is faster?

Quick follow up

$$\text{Clock Cycles}_A = \text{CPU Time}_A \times \text{Clock Rate}_A$$
$$= 8s \times 4GHz = 32 \times 10^9$$

$$\text{Clock Rate}_B = \frac{\text{Clock Cycles}_B}{\text{CPU Time}_B}$$
$$= \frac{1.4 \times \text{Clock Cycles}_A}{6s}$$
$$= \frac{1.4 \times 32}{6s}$$
$$= 7.47 \text{ GHz}$$

Quick follow up

$$\text{Clock Cycles}_B = 44.8 \times 10^9 \quad \textbf{(see prev slide)}$$

$$\text{Clock Rate}_C = \frac{\text{Clock Cycles}_C}{\text{CPU Time}_C}$$

$$= \frac{1.8 \times \text{Clock Cycles}_B}{4s}$$

$$= \frac{1.8 \times 44.8}{4s}$$

$$= 20.16 \text{ GHz}$$

Final verdict: Computer C is the fastest.

Instruction Count and CPI

Clock Cycles = Instruction Count $\times$ Cycles per Instruction

CPU Time = Instruction Count $\times$ CPI $\times$ Clock Cycle Time

$$= \frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock Rate}}$$

Instruction Count and CPI (contd)

- Instruction Count for a program
  - Determined by program, ISA and compiler
- Average cycles per instruction
  - Determined by CPU hardware
  - If different instructions have different CPI

- ISA: Instruction Set Architecture. This architecture guides the processing of instructions on our computers.

CPI Example - 1

- Computer A: Cycle Time = 250ps, CPI = 2.0
- Computer B: Cycle Time = 500ps, CPI = 1.2
- Same ISA
- Which is faster, and by how much?

CPI Example - 1(contd)

$$\text{CPU Time}_A = \text{Instruction Count} \times \text{CPI}_A \times \text{Cycle Time}_A$$

$$I \times 2.0 \times 250\text{ps} = I \times 500\text{ps}$$

$$\text{CPU Time}_B = \text{Instruction Count} \times \text{CPI}_B \times \text{Cycle Time}_B$$

$$I \times 1.2 \times 500\text{ps} = I \times 600\text{ps}$$

A is faster…

CPI Example - 1(contd)

$$\frac{\text{CPU Time}_B}{\text{CPU Time}_A} = \frac{\mathbf{I} \times 600\text{ps}}{\mathbf{I} \times 500\text{ps}} = 1.2$$

…by this much

CPI in More Detail

- If different instruction classes take different numbers of cycles

$$\text{Clock Cycles} = \sum_{i=1}^{n} (\text{CPI}_i \times \text{Instruction Count}_i)$$

- Weighted average CPI

$$\text{CPI} = \frac{\text{Clock Cycles}}{\text{Instruction Count}}$$

$$= \sum_{i=1}^{n} \frac{(CPI_i \times \text{Instruction Count}_i)}{\text{Instruction Count}}$$

CPI Example - 2

- Alternative compiled code sequences using instructions in classes A, B, C

	A	**B**	**C**
CPI for class	1	2	3
IC in sequence 1	2	1	2
IC in sequence 2	4	1	1

CPI Example - 2

- Sequence 1: IC = 5
  - Clock Cycles = $2 \times 1 + 1 \times 2 + 2 \times 3 = 10$
  - Avg. CPI = 10/5 = 2.0
- Sequence 2: IC = 6
  - Clock Cycles = $4 \times 1 + 1 \times 2 + 1 \times 3 = 9$
  - Avg. CPI = 9/6 = 1.5

Thinking Exercise

- A program is run in two machines, A and B. Both machines take the same number of clock cycle time. Because we are executing the same program, no of instruction(s) is the same on both machines. Assume that CPI on machine A takes longer than CPI on machine B. So, now which among the two machines complete executing the program first?

  Post your reflection file in Week5 submission folder, to get participation credits for today.

Performance Summary

- **The BIG Picture**

$$\text{CPU Time} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Clock cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Clock cycle}}$$

Performance Summary

- Performance depends on
  - Algorithm: affects IC, possibly CPI
  - Programming language: affects IC, CPI
  - Compiler: affects IC, CPI
  - Instruction set architecture: affects IC, CPI, T where T, is the Clock Time.

CPI Example - 3

Suppose the following C fragment:

j = i + j;

compiles into the following machine language (for a madeup machine):

```
load i // 5 clock cycles
load j // 5
add // 3
store j // 5
```

Suppose the machine's clock rate is 1.4GHz.

What is the average CPI?

CPI Example - 3(contd)

The average CPI for this fragment is (5+5+3+5)/4 = 4.5

Then the total execution time of the program fragment is:

= (4 ins) $\times$ (4.5 cyc/ins) / (1.4×10^9 cyc/sec)

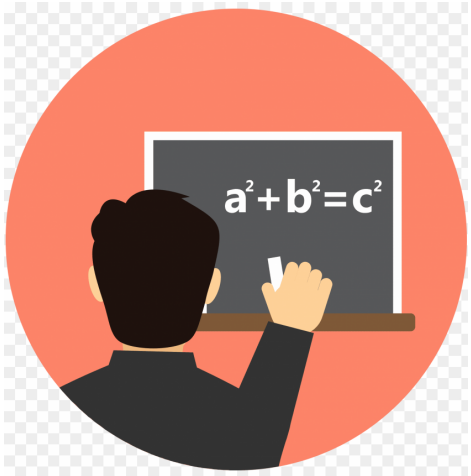
= (18/1.4) $\times$ 10^{-9} = 12.86 ns

Note that 1 billionth of a second = 1×10^{-9} sec

= 1 ns.

Use units to help you determine what to use where:

CPI Example - 3 (The Math)



$$\frac{\text{ins} * \frac{\text{cyc}}{\text{ins}}}{\frac{\text{cyc}}{\text{sec}}} = \cancel{\text{ins}} * \frac{\cancel{\text{cyc}}}{\cancel{\text{ins}}} * \frac{\text{sec}}{\cancel{\text{cyc}}} = \text{sec}$$

CPI Example - 4

What if 60% of a program's instructions are load and store instructions and the rest are add instructions? What is the running time of a program containing 1000 instructions? Use a weighted average CPI.

CPI Example - 4

= (1000 ins) $\times$ (.6 $\times$ 5 + .4 $\times$ 3) (cyc/ins) / (1.4 $\times 10^9$ cyc/sec)

= (1000 ins) * (4.2 cyc/ins) / (1.4 $\times$ 10^9 cyc/sec)

= (4200/1.4) $\times$ 10^{-9} sec = 3000 $\times$ 10^{-9} sec

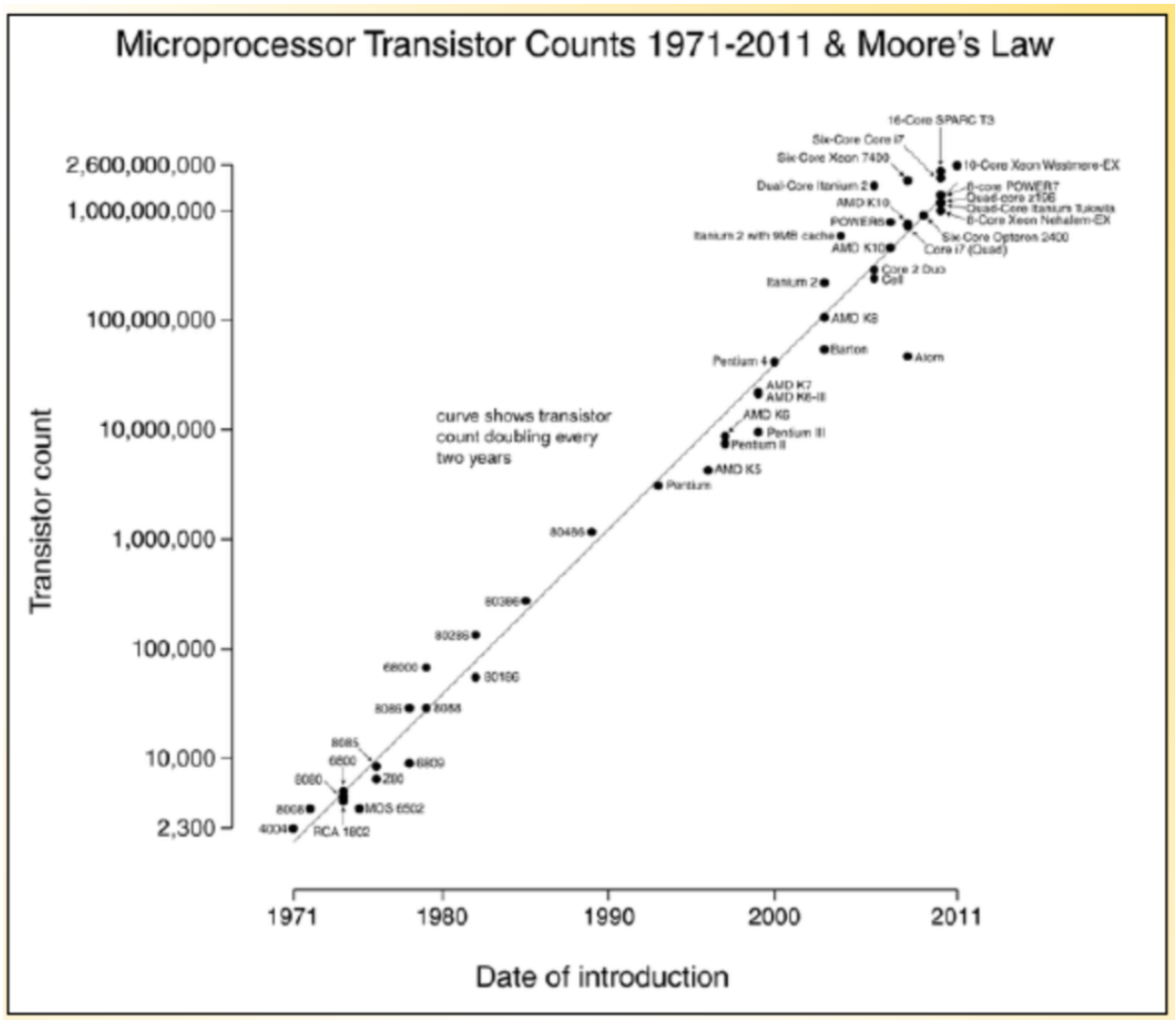
= 3 $\times$ 10^{-6} sec = 3μ s

NOTE: μs = microseconds = millionths of a second

More on Performance

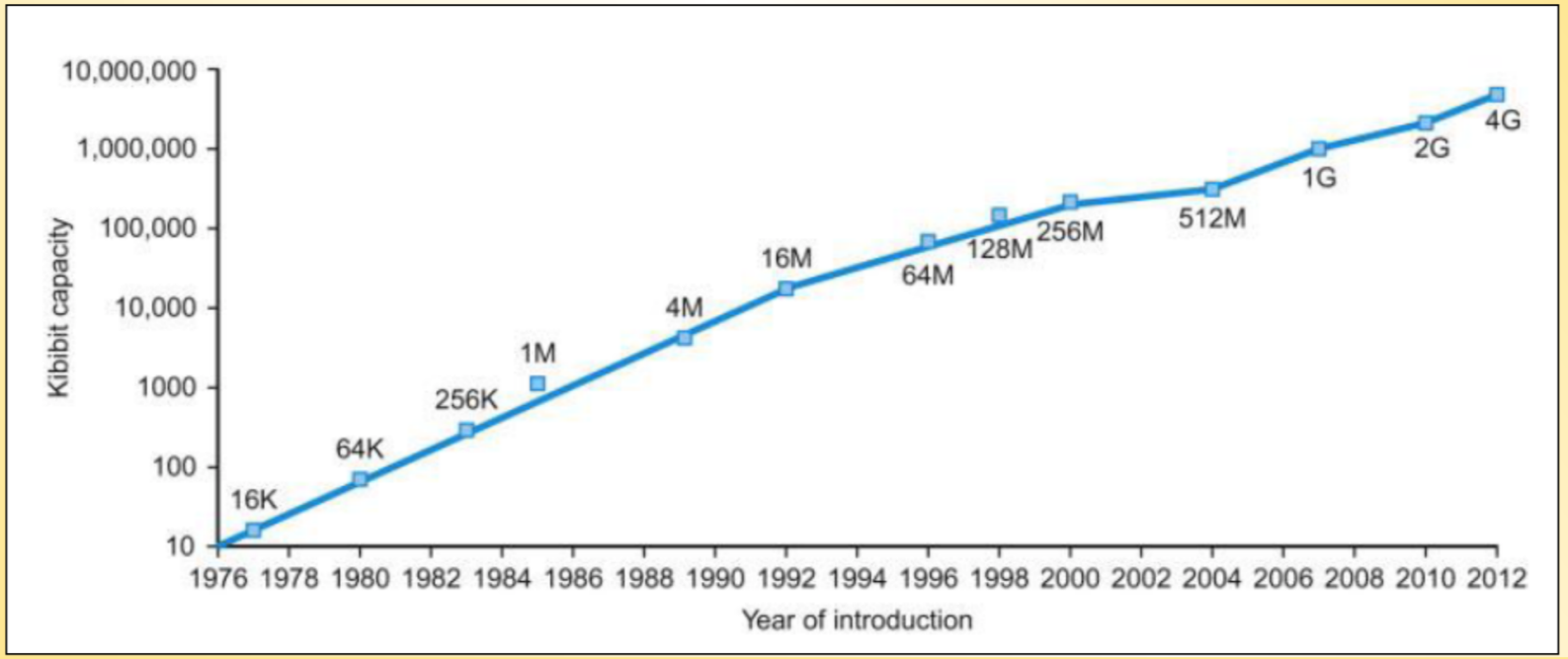
- **Moore's Law:** "The number of transistors in a dense integrated circuit doubles approximately every two years."
- "Chip performance doubles every 18 months."

Moore's Law


Microprocessor Transistor Counts 1971-2011 & Moore's Law
Transistor count
2,600,000,000
1,000,000,000
100,000,000
10,000,000
1,000,000
100,000
10,000
2,300
curve shows transistor count doubling every two years
16-Core SPARC T3
Six-Core Core i7
Six-Core Xeon 7400
10-Core Xeon Westmere-EX
Dual-Core Itanium 2
8-core POWER7
Quad-core z196
Quad-Core Itanium Tukwila
AMD K10
POWER6
8-Core Xeon Nehalem-EX
Itanium 2 with 9MB cache
Six-Core Opteron 2400
Core i7 (Quad)
AMD K10
Core 2 Duo
Cell
Itanium 2
AMD K8
Barton
Atom
Pentium 4
AMD K7
AMD K6-III
AMD K6
Pentium III
Pentium II
AMD K5
Pentium
80486
80386
80286
68000
80186
8086
8085
6800
6502
8080
Z80
8008
MOS 6502
4004
RCA 1802
1971
1980
1990
2000
2011
Date of introduction

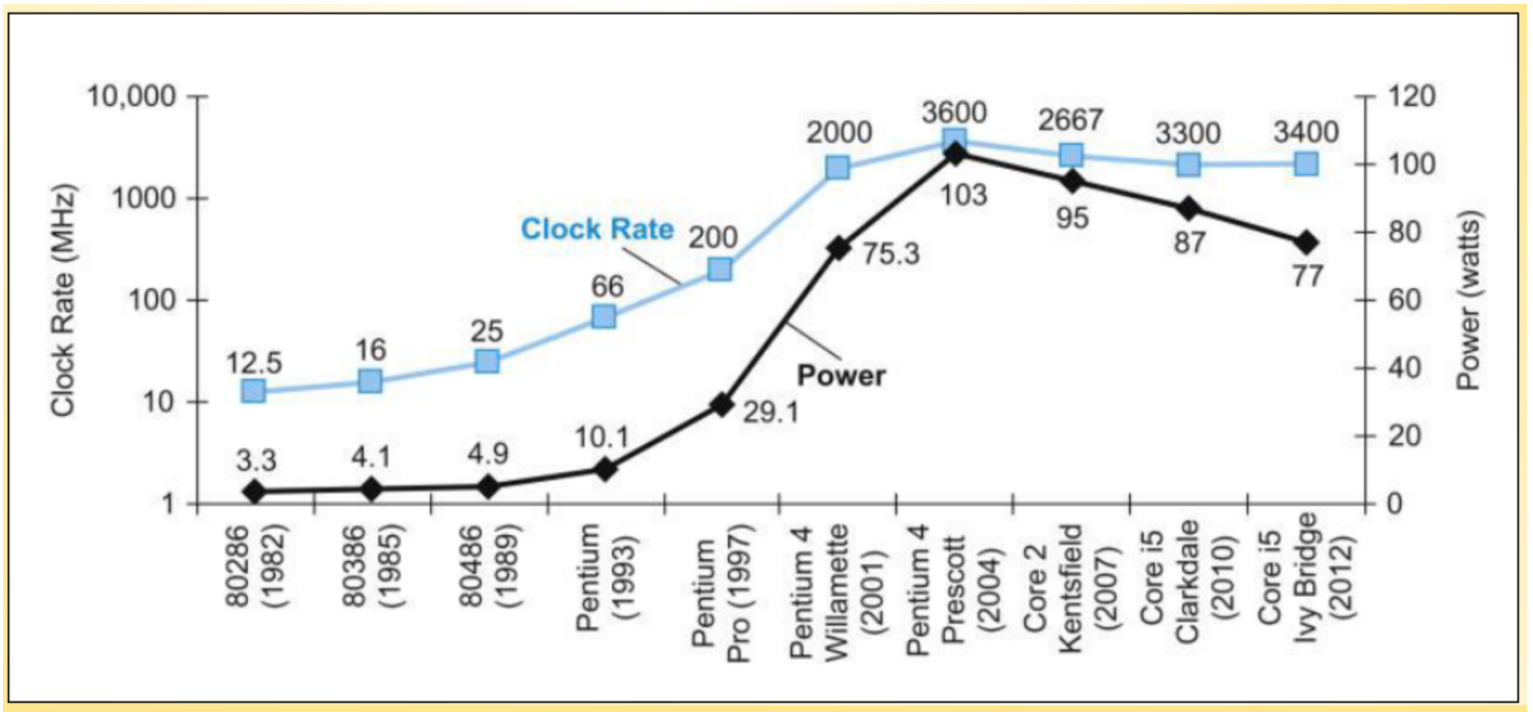
So Moore's Law Will Continue on Forever, Right?

- Growth of DRAM chip capacity over time.



So Moore's Law Will Continue on Forever, Right?

- Clock rate and power for x86 microprocessors.



How can we overcome this?

One Solution: Multicore Processors!

- Issue: Programmers need to rewrite applications to take advantage of mullple cores.
  - Parallelism is sometimes not obvious, sometimes difficult, and sometimes impossible to implement.
  - Instructions assigned to each core need to be balanced, so that scheduling/coordination overhead doesn't defeat parallel performance gains.
  - Now a program needs to be correct, provide a useful interface, AND be fast across multiple cores.
  - **Cloud Computing** is an extension to this approach.

Performance and Improvement!

Amdhal's Law:

- Expecting the improvement of one aspect of a computer to increase overall performance by an amount proportional to the size of the improvement.

Performance and Improvement (contd)

Amdhal's Law:

Execution Time Post Improvement =

$$\frac{\text{Time Affected by Improvement}}{\text{Amount of Improvement}} + \text{Execution Time Unaffected}$$

Improvement Example - 5

- Suppose a program runs in 100 seconds on a computer, with multiply operations responsible for 80 seconds of this time. How much do I have to improve the speed of multiplicaIon if I want my program to run five times faster?

Improvement Example - 5(cntd)

Solution:

Execution Time Post Improvement =

$$\frac{\text{Time Affected by Improvement}}{\text{Amount of Improvement}} + \text{Execution Time Unaffected}$$

$$20s = \frac{80s}{n} + (100s - 80s)$$

$$20s = \frac{80s}{n} + 20s \implies 0s = \frac{80s}{n}$$

Not possible to improve!

Improvement Example - 6

- Suppose a program runs in 100 seconds on a computer, with multiply operations responsible for 80 seconds of this time. How much do I have to improve the speed of multiplicaIon if I want my program to run **two** times faster?

Improvement Example - 6(cntd)

Solution:

Execution Time Post Improvement =

$$\frac{\text{Time Affected by Improvement}}{\text{Amount of Improvement}} + \text{Execution Time Unaffected}$$

$$50s = \frac{80s}{n} + (100s - 80s)$$

$$50s = \frac{80s}{n} + 20s \implies 30s = \frac{80s}{n}$$

$$n = \frac{80}{30} = 2.667x \textbf{ improvement}$$

Reading Assignment

Section 1.6, 1.7 in **PH**

Questions

Do you have any questions from this class discussion?