Fall 2020
EE 3613: Computer Organization
Chapter 1: Performance

Avinash Karanth
Department of Electrical Engineering & Computer Science
Ohio University, Athens, Ohio 45701
E-mail: karanth@ohio.edu
Website:
http://oucsace.cs.ohiou.edu/~avinashk/classes/ee461a/ee461a.htm

1

Outline

- Performance Factors
- Power Wall
- Amdahl's Law

2

Performance Metrics

- Purchasing perspective
  - Given a collection of machines, which has the best performance?
  - Best cost/performance ratio?
- Design perspective
  - Faced with a set of design options, which has the
    - Best performance?
    - Best performance improvement?

Airplane	Capacity	Range	Speed	Passenger Throughput (Speed x Capacity)
Boeing 777	375	4630	610	228750
Boeing 747	470	4150	610	286700
BAC/Sud Concorde	132	4000	1350	178200
Douglas DC-8-50	146	8750	544	79424

3

Computer Performance: TIME

- Response Time (Latency)
  - How long does it take for my job to run ?
  - How long does it take to execute a job ?
  - How long should I wait for the database query?
- Throughput
  - How many jobs can run at once on the machine ?
  - What is the average execution time ?
  - How much work is getting done ?

- If we upgrade a processor with a faster processor, what do we decrease?
- If we add a new machine, what do we improve?

4

CPU Clock Cycles

- Wall clock time, Elapsed time, or Response time
  - Counts everything (disk, memory, I/O)
  - A useful number, but often not good for comparison
- CPU Time
  - Doesn't count I/O or time spent by other programs
  - Can be broken up into system and user time
- Our Focus: **User CPU Time**
  - Time spent executing the lines of codes that are "in" our program

5

Defining Performance

- Reducing response time
  - To maximize performance, need to minimize response time

$$\text{Performanc e (x)} = \frac{1}{\text{Execution Time (x)}}$$

- If performance of X is greater than Y by a factor of "n"

$$\frac{\text{Performanc e (X)}}{\text{Performanc e (Y)}} = \frac{\text{Execution Time (Y)}}{\text{Execution Time (X)}} = n$$

- Decreasing the response time mostly increases the throughput

6

Performance Factors

- Another way of reporting the execution time is to use cycles

CPU execution time for a program = # CPU clock cycles for a program x clock cycle time

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

- Clock "ticks" indicate when to start an activity
- Cycle time = time between ticks = seconds per cycle
- Clock rate (frequency) = cycles per second

7

Improving Performance

- Said another way,

$$\frac{\text{seconds}}{\text{program}} = \frac{\text{cycles}}{\text{program}} \times \frac{\text{seconds}}{\text{cycle}}$$

- So we can improve performance (everything being equal) in the following way:
  - _____ # of required cycles for a program
  - _____ the clock cycle time,
  - _____ the clock rate

- Improve performance by reducing either the length of the clock cycle or the number of clock cycles required for a program

8

Cycles Required for a Program

- Can we assume number of cycles = number of instructions ?
- This assumption is incorrect, different instructions take different amounts of time on different machines, Why?
  - Remember that these are machine instructions, not lines of C code
  - Multiplication takes more time than addition
  - Floating point operations take more time than integers
  - Accessing memory takes more time than accessing registers
  - NOTE: **Changing the cycle time often changes the number of cycles required for various instructions**

9

Example Problem

- Our favorite program runs in 10 seconds on computer A, which has a 4 GHz clock. We are trying to help a computer designer build a new machine B, that will run this program in 6 seconds. The designer can use new (or perhaps more expensive) technology to substantially increase the clock rate, but has informed us that this increase will affect the rest of the CPU design, causing machine B to require 1.2 times as many clock cycles as machine A for the same program. What clock rate should we tell the designer to target?"

10

Effective CPI

- Computing the overall effective CPI is done by summing the different instructions and their individual clock cycles

$$Overall\ Effective\ CPI = \frac{\sum_{i=1}^{n} CPI_i \times IC_i}{IC}$$

  - Where IC_i is the count of the number of instructions of class i executed
  - CPI_i is the (average) number of clock cycles per instruction for that instruction class
  - n is the number of instruction classes
  - IC is the total instruction count

- The overall effective CPI varies by instruction mix – a measure of dynamic frequency of instructions across many programs

11

CPI Example

	CPI for this instruction class		
	A	B	C
CPI	1	2	3

Code Sequence	Instruction count for instruction class		
	A	B	C
1	2	1	2
2	4	1	1

- Which code sequence executes the most instructions? Which will be faster?

12

Performance Equation

- Our basic performance equation becomes

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

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

- These factors separate three factors that affect performance
  - Can measure CPU execution time (run the program)
  - Clock rate is given
  - Can measure the overall instruction count using profilers/simulators
  - CPI varies by the instruction type and implementation details

13

Determines CPU Performance

CPU time = Instruction_count x CPI x clock_cycle

	Instruction_ count	CPI	clock_cycle
Algorithm			
Programming language			
Compiler			
ISA			
Processor organization			
Technology			

14

The Power Wall

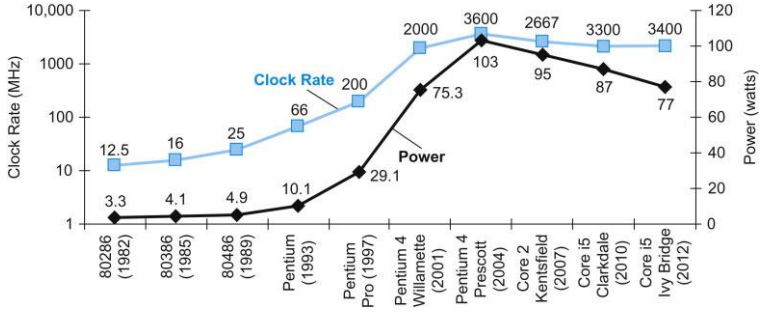


FIGURE 1.16 Clock rate and Power for Intel x86 microprocessors over eight generations and 25 years. The Pentium 4 made a dramatic jump in clock rate and power but less so in performance. The Prescott thermal problems led to the abandonment of the Pentium 4 line. The Core 2 line reverts to a simpler pipeline with lower clock rates and multiple processors per chip. The Core i5 pipelines follow in its footsteps.

15

Quantify Power

- For CMOS chips, traditional dominant energy consumption has been in switching transistors, called dynamic power

$$\text{Power}_{\text{Dynamic}} = \frac{1}{2} \times \text{Capacitive Load} \times \text{Voltage}^2 \times \text{Frequency}$$

$$\text{Energy}_{\text{Dynamic}} = \frac{1}{2} \times \text{Capacitive Load} \times \text{Voltage}^2$$

- For fixed task, slowing clock rate (frequency switched) reduces power, but not energy

16

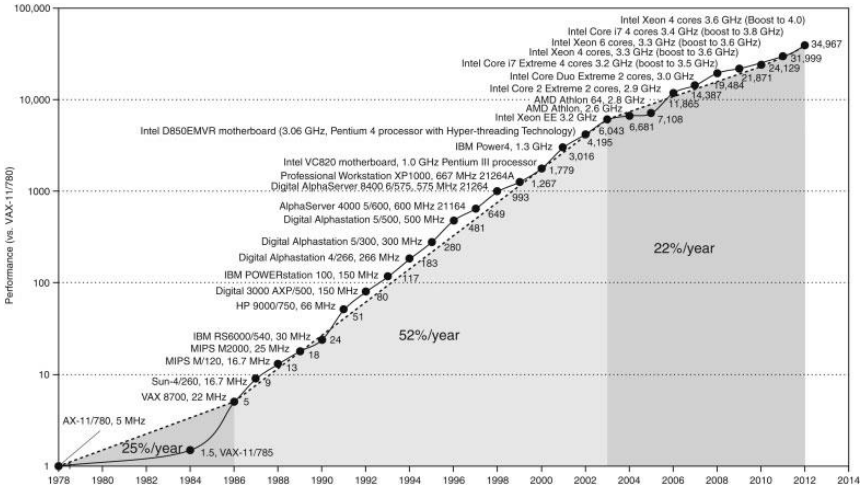
16

Quantify Power

- Capacitive load is a function of the number of transistors connected to the output and technology
  - Determines capacitance of wires and transistors
- Dropping voltage helps both, so went from 5 V to 1 V
- To save energy and dynamic power, most CPUs now turn off the clock of inactive modules (Eg: Floating Point Unit)

- Problem: Suppose 15% reduction in voltage results in a 15% reduction in frequency and the new processor has the capacitance load of 85% of the more complex old processor, then what is the impact on dynamic power?

Growth in Microprocessor Performance



Multi-Cores, Clock Rate and Power

- The Sea Change: From uniprocessor designs to multicores!

Product	AMD Opteron X4 (Barcelona)	Intel Nehalem	IBM Power 6	Sun Ultra SPARC T2 (Niagara 2)
Cores per chip	4	4	2	8
Clock rate	2.5 GHz	~2.5 GHz?	4.7 GHz	1.4 GHz
Microprocessor power	120 W	~100 W ?	~100 W ?	94 W

19

Amdahl's Law (Law of Diminishing Returns)

- Every improvement typically affects a fraction of the program
- Execution time is generally divided into two classes: affected class and unaffected class

$$T(\text{after improvement}) = T(\text{unaffected}) + \frac{T(\text{affected})}{\text{Improvement Factor}}$$

$$\text{Speedup} = \frac{T(\text{before improvement})}{T(\text{after improvement})}$$

- Example: Suppose a program runs in 100 seconds and multiply is responsible for 80 seconds of this time. How much do we have to improve the speed of multiplication if we want the program to run 4 times faster?
- How about making it 5 times faster?

20

MIPS (Million Instructions Per Second)

$$\text{MIPS} = \frac{\text{Instruction Count}}{\text{Execution Time} \times 10^6}$$

- MIPS rating misleading
- Execution time is the only valid and unimpeachable measure of performance

CPU time = Instruction count x CPI x clock cycle

$$\frac{\text{seconds}}{\text{program}} = \frac{\text{Instructions}}{\text{program}} \times \frac{\text{Clock cycle}}{\text{Instruction}} \times \frac{\text{seconds}}{\text{clock cycle}}$$

21

Summary

- Performance is specific to a particular program
  - Total execution time is a consistent summary of performance
- For a given architecture, performance comes from
  - Increases the clock rate (without adversely affecting CPI)
  - Improvements in processor organization that lower the CPI
  - Compiler enhancements that lower CPI and/or instruction count
- Pitfall
  - Expecting improvement in one aspect of a machine's performance to affect the total performance

22