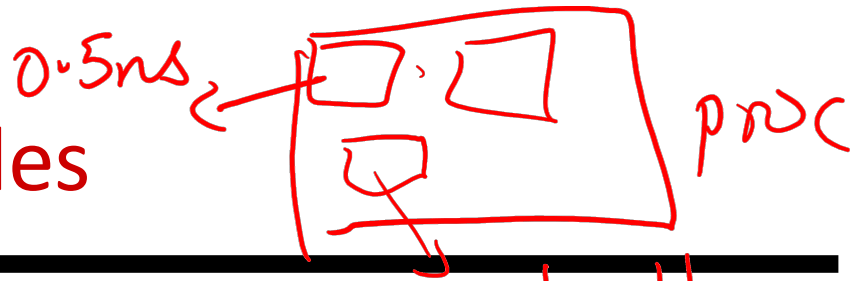
Lecture 3: Performance/Power, MIPS Instructions

- Today's topic:
  - More performance/power equations, examples
  - MIPS instructions
- HW1 is due on Thursday (+ 1.5 days)
- TA office hours (CADE Lab, TA queue)

A Primer on Clocks and Cycles



clk speed = how many cycles in 1 sec

$= \frac{1}{0.5\,ns}$

$= 2 \times 10^9$ Hz

= 2 GHz

2 billion cycles in 1 second

clk start

(Hz = 1/s)

cycle time = 0.5 ns (slowest cct blk) = 500 ps

stop the last task & move to the next

IPC or CPI = cycles per instr = 1 if busy every cycle

Performance Equation - I

cycles = exec time / cycle time

= exec time x clk speed
= 10s x 3GHz

for a program

CPU execution time = CPU clock cycles x Clock cycle time

Clock cycle time = 1 / Clock speed

program runs for 6 B cycles × 1/3 GHz = 2 secs

If a processor has a frequency of 3 GHz, the clock ticks 3 billion times in a second – as we'll soon see, with each clock tick, one or more/less instructions may complete

If a program runs for 10 seconds on a 3 GHz processor, how many clock cycles did it run for?

30 B cycles

If a program runs for 2 billion clock cycles on a 1.5 GHz processor, what is the execution time in seconds?

exec time = 2B × 1/1.5GHz = 1.33 s

Performance Equation - II

$\frac{10s \times 2 \times 10^9 Hz}{3} = 6.66 \times 10^9$ instrs = # instrs

CPU clock cycles = number of instrs x avg clock cycles per instruction (CPI)

Substituting in previous equation,

Execution time = clock cycle time x number of instrs x avg CPI

If a 2 GHz processor graduates an instruction every third cycle, how many instructions are there in a program that runs for 10 seconds?

CPI = 3

6.66 B

$10s = \frac{1}{2 \times 10^9 Hz} \times \#instrs \times 3$

Factors Influencing Performance

Perf = 1 / exec time

Execution time = clock cycle time x number of instrs x avg CPI

- Clock cycle time: manufacturing process (how fast is each transistor), how much work gets done in each pipeline stage (more on this later)

Freq ↑

- Number of instrs: the quality of the compiler and the instruction set architecture ← next few weeks
- CPI: the nature of each instruction and the quality of the architecture implementation ← after spr brk

Example

MIPS exectime = $\frac{1}{1\,GHz} \times 4 \times 10^9 \times 1.5 = 6$ secs

x86 exectime = $\frac{1}{1.5\,GHz} \times 2 \times 10^9 \times 6 = 8$ secs

secs secs

Execution time = clock cycle time x number of instrs x avg CPI

(1/Hz)

Which of the following two systems is better?

- A program is converted into 4 billion MIPS instructions by a compiler ; the MIPS processor is implemented such that each instruction completes in an average of 1.5 cycles and the clock speed is 1 GHz

MIPS ISA (language)

CPI = 1.5

- The same program is converted into 2 billion x86 instructions; the x86 processor is implemented such that each instruction completes in an average of 6 cycles and the clock speed is 1.5 GHz

x86

CPI = 6

Power and Energy

- Total power = dynamic power + leakage power
- Dynamic power α activity x capacitance x voltage^2 x frequency
- Leakage power α voltage
- Energy = power x time
  (joules) (watts) (sec)
- For a CPU-bound program,
  Execution time α cycle time α 1 / clock speed

Example Problem

$E_{Base} = P_{base} \times t_{base}$

$= (DP + LP)_{base} \times t_{base}$

$= (70 + 30)W \times 100s = 10J$

$= 10\,KJ$

- A 1 GHz processor takes 100 seconds to execute a CPU-bound program, while consuming 70 W of dynamic power and 30 W of leakage power. Does the program consume less energy in Turbo boost mode when the frequency is increased to 1.2 GHz?

t_{new} = new exec time

= old exec x old freq / new freq

= 100s x 1GHz / 1.2GHz = 83.3 secs

clk spd ↑ 1.2

t ↓ 1.2

exec time $\propto \frac{1}{\text{clk spd}}$

$DP \propto freq$ ⇒ oldDP x new freq / old freq = 70 x 1.2 = 84 W

$LP \propto V \times I_{draw}$ = 30W

$E = (84 + 30) \times 83.33$

$= 9500\,J$

Example Problem

- A 1 GHz processor takes 100 seconds to execute a CPU-bound program, while consuming 70 W of dynamic power and 30 W of leakage power. Does the program consume less energy in Turbo boost mode when the frequency is increased to 1.2 GHz?

Normal mode energy = 100 W x 100 s = 10,000 J
Turbo mode energy = (70 x 1.2 + 30) x 100/1.2 = 9,500 J

Note:
Frequency only impacts dynamic power, not leakage power.
We assume that the program's CPI is unchanged when frequency is changed, i.e., exec time varies linearly with cycle time.

Benchmark Suites

2017

- Each vendor announces a SPEC rating for their system
  - a measure of execution time for a fixed collection of programs
  - is a function of a specific CPU, memory system, IO system, operating system, compiler
  - enables easy comparison of different systems

The key is coming up with a collection of relevant programs

SPEC CPU

- SPEC: System Performance Evaluation Corporation, an industry consortium that creates a collection of relevant programs
- SPEC 2006 includes 12 integer and 17 floating-point applications
- The SPEC rating specifies how much faster a system is, compared to a baseline machine – a system with SPEC rating 600 is 1.5 times faster than a system with SPEC rating 400
- Note that this rating incorporates the behavior of all 29 programs – this may not necessarily predict performance for your favorite program!
- Latest version: SPEC 2017

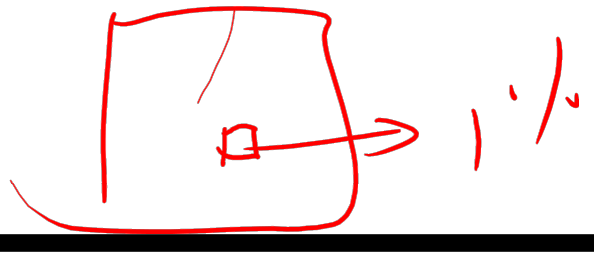
Deriving a Single Performance Number

How is the performance of 29 different apps compressed into a single performance number?

$GM(a,b) = \sqrt{a \times b}$

- SPEC uses geometric mean (GM) – the execution time of each program is multiplied and the N^{th} root is derived
- Another popular metric is arithmetic mean (AM) – the average of each program's execution time
- Weighted arithmetic mean – the execution times of some programs are weighted to balance priorities

Amdahl's Law



- Architecture design is very bottleneck-driven – make the common case fast, do not waste resources on a component that has little impact on overall performance/power

- Amdahl's Law: performance improvements through an enhancement is limited by the fraction of time the enhancement comes into play

- Example: a web server spends 40% of time in the CPU and 60% of time doing I/O – a new processor that is ten times faster results in a 36% reduction in execution time (speedup of 1.56) – Amdahl's Law states that maximum execution time reduction is 40% (max speedup of 1.66)

Common Principles

- Amdahl's Law
- Energy: performance improvements typically also result in energy improvements – less leakage
- 90-10 rule: 10% of the program accounts for 90% of execution time
- Principle of locality: the same data/code will be used again (temporal locality), nearby data/code will be touched next (spatial locality)

Recap

- Knowledge of hardware improves software quality: compilers, OS, threaded programs, memory management
- Important trends: growing transistors, move to multi-core and accelerators, slowing rate of performance improvement, power/thermal constraints, long memory/disk latencies
- Reasoning about performance: clock speeds, CPI, benchmark suites, performance and power equations
- Next: assembly instructions

Instruction Set

- Understanding the language of the hardware is key to understanding the hardware/software interface
- A program (in say, C) is compiled into an executable that is composed of machine instructions – this executable must also run on future machines – for example, each Intel processor reads in the same x86 instructions, but each processor handles instructions differently
- Java programs are converted into portable bytecode that is converted into machine instructions during execution (just-in-time compilation)
- What are important design principles when defining the instruction set architecture (ISA)?

A Basic MIPS Instruction

C code: a = b + c ;

Assembly code: (human-friendly machine instructions)

```
add   a, b, c      #  a is the sum of b and c
```

Machine code: (hardware-friendly machine instructions)

00000010001100100100000000100000

Translate the following C code into assembly code:

a = b + c + d + e;