CSE 331: Computer Architecture **(Due: 12/10/22)**

CSE331 PS #1

Instructor: Dr. Alp Arslan Bayrakçi
Assistant: Gizem Süngü

Brief Information about Computation Performance

- Response time: How long it takes to do a task.
- Performance = $\frac{1}{ExecutionTime}$
- If "X is n time faster than Y", then $\frac{Performance_X}{Performance_Y} = \frac{Execution_Y}{Execution_X}$
- **Clock Period (Clock Cycle Time):** Duration of a clock cycle.
- **Clock Rate:** Cycles per second 4GHz = 400 MHz = 4 $\times 10^9$Hz.
- **CPI:** Cycles per instruction
- Clock Rate = $\frac{1}{ClockCycleTime}$
- CPU Time = CPU clock cycles ÷ CPU Clock rate
- CPU Clock Cycles = (instructions ÷ program) × (Clock cycles ÷ instruction) = Instruction count × CPI
- CPU Time = Instruction count × CPI ÷ CPU Clock rate
- CPU Time = Instruction count × CPI × CPU cycle time
- CPU Time = $\frac{instructions}{program} \times \frac{clockcycles}{instruction} \times \frac{seconds}{clockcycles}$

Problem 1: Computing Performance (0 points)

Following table gives information about two different processors P1 and P2 each using the same instruction set but different clock rates and cycles per instruction type.

	Clock Rate	CPI Class A	CPI Class B	CPI Class C	CPI Class D
P1	1.5 GHz	1 cycles	2 cycles	3 cycles	4 cycles
P2	2 GHz	2 cycles	2 cycles	2 cycles	2 cycles

Given a program divided into classes as follows: 15% class A, 25% class B, 40% class C and 20% class D. Which one of the P1 and P2 is faster and by how many times?

(Solution)
$CPI_{P1} = 0.15 \times 1 + 0.25 \times 2 + 0.40 \times 3 + 0.20 \times 4 = 0.15 + 0.5 + 1.2 + 0.8 = 2.65$

$CPI_{P2} = 0.15 \times 2 + 0.25 \times 2 + 0.40 \times 2 + 0.20 \times 2 = 0.3 + 0.5 + 0.8 + 0.4 = 2$

CPU Time$_{P1} = 2.65 \times I \times 1.5GHz =$ 1.76 I ns

CPU Time$_{P2} = 2 \times I \times 2GHz =$ I ns

P2 is faster than P1.

Problem 2: Computing Performance (0 points)

Consider two different implementations of the same instruction set architecture. The instructions can be divided into four classes according to their CPI (class A, B, C, and D). Program P1 clock rate is 2.5 GHz and Program P2 clock rate is 3 GHz.

Given a program with a instruction count of 10^6 instructions divided into classes as follows: 10% class A, 20% class B, 50% class C, and 20% class D. In P1, Class A, B, C and D instructions need 1, 2, 3, and 3 cycles, respectively. In P2, Class A, B, C and D instructions use 2, 2, 2, and 2 cycles, respectively. Which implementation is faster?

(Solution) Show your work for practice.

Problem 3: How to use MARS ISS? (0 points)

As of V4.0, MARS requires Java J2SE 1.5 (or later) SDK installed on your computer.

You can download MARS 4.5 is from here.

You can run a sample MIPS program on MARS.

Problem 4: Assembler Directives MIPS (0 points)

Directives are commands that are part of the assembler syntax but are not related to the MIPS instruction set. All assembler directives begin with a period (.).

- Directives tell the assembler how to function.
- Groups of directives
  - In which segment should following code or data be placed
  - Externally visible labels
  - Reserve space for data
- .text
  - Anything that follows is placed in the text segment
  - The text segment is where executable code exists
  - .text may be followed by an address
- .data
  - Anything that follows is placed in the data segment
  - The data segment is where static data stored in memory exists
  - .data may be followed by an address
  - Anything that follows is placed in the data segment beginning at the specified address

Assembler Data Value Directives

- .word w1, w2, . . .
  - The value of each operand (w1, w2, etc.) is stored in a 32-bit word in memory
  - The words are aligned on word boundaries
- .half h1, h2, ..
  - The value of each operand (h1, h2, etc.) is stored in a 16-bit halfword in memory
- .byte b1, b2,..

  - The value of each operand (b1, b2, etc.) is stored in a 8-bit byte in memory
  - No alignment is performed

- .ascii ”string”
  - The ”string” is stored in memory using ASCII values
  - No alignment is performed

- .asciiz ”string”
  - The ”string” is stored in memory using ASCII values with null-termination
  - No alignment is performed