University of Maryland Baltimore County
Department of Computer Science and Electrical Engineering

CMSC 411, Computer Architecture

***Assignment #1* Solutions**

Due: Thursday 9/21/17 in class

Question 1: *(40 Points)*

Consider two different implementations, P1 and P2, of the same instruction set. There are five classes of instructions (A, B, C, D, and E) in the instruction set.

P1 has a clock rate of 4 GHz. P2 has a clock rate of 6 GHz. The average number of cycles for each instruction class for P1 and P2 is as follows:

Class	CPI for this class on P1	CPI for this class on P2
A	1	2
B	2	2
C	3	2
D	4	4
E	3	4

A) Assume that peak performance is defined as the fastest rate that a computer can execute any instruction sequence. What are the peak performances of P1 and P2 expressed in instructions per second? **10 points for each P1 and P2 (total 20 points)**, (5 pts for performace elaboration, 5 pts for final answer)

Peak performance → smallest CPI

P1: CPI class A
P2: CPI class A, B or C

$$\text{Performance } P_1 = \frac{1}{\text{Execution Time } P_1} = \frac{1}{\text{CPI A}*\text{clock cycle}} = \frac{\text{Clock rate}}{\text{CPI A}} = \frac{4*10^9}{1} = 4*10^9 \text{ inst/sec}$$ ***QED***

$$\text{Performance } P_2 = \frac{1}{\text{Execution Time } P_2} = \frac{1}{\text{CPI A}*\text{clock cycle}} = \frac{\text{Clock rate}}{\text{CPI A}} = \frac{6*10^9}{2} = 3*10^9 \text{ inst/sec}$$ ***QED***

B) If the number of instructions executed in a certain program is divided equally among the classes of instructions except for class A, which occurs twice as often as each of the others, how much faster is P2 than P1? **10 points** (5 pts for correct average CPI cylces and execution times, 5 pts for speedup answer) At what frequency does P2 have the same performance as P1 for this instruction mix? **10 points** (5 pts for expression, 5 pts for final answer)

Average CPI for P_1 = ((2*1) +2 + 3 + 4 + 3)/6 = 7/3
Average CPI for P_2 = ((2*2) +2 + 2 + 4 + 4)/6 = 8/3

$$\text{Execution\_time}_{P_1} = \text{IC} * (7/3) * \left(\frac{1}{4*10^9}\right) = \left(\frac{7}{3}*\frac{1}{4}\right)*10^{-9} \text{ sec} = 0.583 \text{ ns}$$

$$\text{Execution\_time}_{P_2} = \text{IC} * (8/3) * \left(\frac{1}{6*10^9}\right) = \left(\frac{8}{3}*\frac{1}{6}\right)*10^{-9} \text{ sec} = 0.444 \text{ ns}$$

$\therefore\ P_2$ is faster by $\left(\frac{7}{12} * \frac{18}{8}\right) = 1.3125 \approx 1.31$ ***QED***

Or speedup $= \frac{\text{Performance } P_2}{\text{Performance } P_1} = \frac{\text{Execution Time } P_1}{\text{Execution Time } P_2} = \frac{0.583}{0.444} = 1.313 \approx 1.31$ ***QED***

To make P2 having the same performance as P1 (which is degraded), we need to find out the new clock rate for P2 using Execution Time of P1

$$\text{Performance } P_1 = \frac{1}{\text{Execution Time } P_1} = \frac{1}{0.583\ ns} = 1.715 * 10^9 \text{ inst/sec or } 1715 \text{ MIPS}$$

$$1.715 * 10^9 = \frac{1}{\text{Execution Time } P_2} = \frac{\text{Clock Rate}}{\text{CPI}_{P_2} * \text{IC}} = \frac{\text{Clock Rate}}{(\frac{8}{3}) * \text{IC}}$$

$$\text{Clock Rate} = (\tfrac{8}{3})\text{IC} * 1.715 * 10^9 = 4.573 * 10^9 \approx 4.57 \text{ GHz}$$ ***QED***

Note: Since the IC for P1 and P2 are the same, so we refrain the symbol.

Question 2: *(35 Points)*

Consider adding a new index addressing mode to MIPS. The addressing mode adds two registers and an 11-bit signed offset to get the effective address. Our compiler will be changed so that the following two code sequences will be replaced as indicated:

```
(1)  ADD    R1, R1, R2  }
                        }     LW     Rd, 100 (R1, R2)
     LW     Rd, 100(R1) }

(2)  ADD    R1, R1, R2  }
                        }     SW     Rd, 100 (R1, R2)
     SW     Rd, 100(R1) }
```

Use the instruction frequencies shown in the following table.

Instruction	load	store	add	sub	mul	compare	load imm	cond branch	jump	call	return	shift	and	or	other
Frequency	22.8%	14.3%	14.6%	0.5%	0.1%	12.4%	6.8%	11.5%	1.3%	1.1%	1.5%	6.2%	1.6%	4.2%	1.1%

A) Assume that the addressing mode can be used for 10% of the displacement loads and stores (accounting for both the frequency of this type of address calculation and the shorter offset). What is the ratio of instruction count on the enhanced MIPS compared to the original MIPS? **20 points** (5 pts for expression - compromised, 10 pts for correct IC formula and its values, 5 pts for final answer)

Let's consider the original MIPS format;

```
ADD R1,R1,R2                ; R1 = R1 + R2
LW Rd,100(R1)               ; load from 100 + R1
```

The ADD instruction is used to compute an intermediate value of the effective address for the LW, then the LW instruction adds the offset value of 100 to produce the intended address and the value from such address will be loaded into Rd.

For the new MIPS format;

```
LW Rd,100(R1,R2)            ; load from 100 + R1 + R2
```

The ADD has been included into the load (or store) instruction. To make this possible some of the bits of the constant offset field of the original displacement load (or store) instruction must be used to encode the name of the second register instead. The remaining offset bits can encode a reduced range of offset values.

The question says that 10% of the displacement loads and stores in the original code can use the new form. From the table, these **loads** and **stores** comprise 22.8% and 14.3% of the average instruction mix, respectively.

Thus, assuming that the number of instructions in original MIPS is $IC_{original}$ and the number of instructions in enhanced MIPS is $IC_{enhanced}$, then the ratio of instructions executed by the enhanced MIPS to the original MIPS. We need to find out the new IC for enhanced version of MIPS.

$$\frac{IC_{enhanced}}{IC_{original}} = \frac{IC_{original} - IC_{original}\,[(10\%*22.8\%)+(10\%*14.3\%)]}{IC_{original}} = \frac{IC_{original}\,(1-[(10\%*22.8\%)+(10\%*14.3\%)])}{IC_{original}}$$

$$= \frac{IC_{original}\,(1-[(0.1*0.228)+(0.1*0.143)])}{IC_{original}} = 1 - [(0.1*0.228) + (0.1*0.143)]$$

$$= 1 - (0.0128 + 0.0143) = 1 - 0.0371 = 0.9629$$

The enhanced MIPS executes 3.71% fewer instructions than the original MIPS. ***QED***

B) If the new addressing mode lengthens the clock cycle by 5%, which machine will be faster and by how much? **15 points** (5 pts for expression - compromised, 5 pts for correct speedup formula and its values, 5 pts for correct machine answer and the final value)

While the new addressing mode reduces the total instruction count, the work done by the new instruction requires addition of three operands to compute the effective address. It yields that (as hint by the question) this might require a slower clock cycle to allow effective address computation to be completed.

The question says that the increase is 5%. To determine which version of MIPS is faster, we use traditional equation as below:

Execution_time = CPI * Instuction Count * Clock Cycle

However, one of the required parameters is the CPI. Since there is no information about the CPI for both versions in the question, we assume that CPI is unchanged ($CPI_{enhanced} \approx CPI_{original}$). This is reasonable because the fraction of affected instructions in question A, 3.71%, is relatively small.

$$\text{Speedup} = \frac{\text{Exection\_Time}_{original}}{\text{Exection\_Time}_{enhanced}} = \frac{CPI_{original} * IC_{original} * CC_{original}}{CPI_{enhanced} * IC_{enhanced} * CC_{enhanced}}$$

$$= \frac{CPI_{original} * IC_{original} * CC_{original}}{CPI_{original} * (0.9629IC_{original}) * (1.05CC_{original})}$$

$$= \frac{1}{(0.9629*1.05)} = 0.9890756$$

Since the speedup is less than 1, it turns out, even though the instruction count is lower for the enhanced MIPS version, the increase in clock cycles time yields the penalty. The enhanced MIPS is actually 1.09% slower than the original MIPS, implying that this architectural feature is not one to include in a future version of machine.

∴ Machine 1 is faster than machine 2 by 1.09% ***QED***

Question 3: *(25 Points)*

Your company could speed up a Java program on their new computer by adding hardware support for garbage collection. Garbage collection currently comprises 20% of the cycles of the program. You have two possible changes to the machine. The first one would be to automatically handle garbage collection in hardware. This causes an increase in cycle time by a factor of 1.2. The second would be to provide for new hardware instructions to be added to the ISA that could be used during garbage collection. This would halve the number of instructions needed for garbage collections but increase the cycle time by 1.1. Which of these two options, if either, should you chose? (You MUST explain your answer). **25 points** (10 pts for clear explanation and assumption – question clearly stated, 5 pts for correct expression for the first machine, 5 pts for correct expression for the second machine, 5 pts for final answer of correct machine)

Let's denote; Machine$_{\text{original}}$ is the original machine for benchmarking
Machine$_{\text{A}}$ is the machine **without** new hardware for garbage collection
Machine$_{\text{B}}$ is the machine **with** the new hardware for garbage collection

Similar to Question 2, since the question doesn't provide the information about the CPI, so we assume that CPI is unchanged. Note that since the instructions for garbage collection but managed by hardware for both machines (the second one added another hardware) is the part of overall IC so that the IC for garbage collection should be recalculated from the entire IC for correct measurement.

Recall; Execution_time = CPI * Instruction Count * Clock Cycle

(1) Automatic garbage collection by hardware (Machine$_{\text{A}}$): the instructions for garbage collection (20%) is normal and cycle time increased by 1.2. We can say that this option reduces the number of instructions to 80%, but increases the time to 120%.

The execution time of Machine$_{\text{A}}$ is $(1 – 0.2) * 1.2 = 0.8 * 1.2 = 0.96$ times that of Machine$_{\text{original}}$. ***QED***

(2) Special garbage collection instructions (Machine$_{\text{B}}$), the instructions for garbage collection (20%) is reduced by half and cycle time increased by 1.1. We can say that this option removes 20%/2 = 10% of the instructions and increases the time taken to 110%.

The execution time of Machine$_{\text{B}}$ is $(1 – \frac{0.2}{2}) * 1.1 = 0.9 * 1.1 = 0.99$ times that of Machine$_{\text{original}}$. ***QED***

∴ Therefore, the first option (Machine$_{\text{A}}$) is faster of the two, thus the better choice. Moreover, it is also faster than the orginial, so you should have hardware automatically do garbage collection. ***QED***