Name: KEY **Id#**

ICS 233, Term 141

Computer Architecture & Assembly Language

Quiz# 5

Date: Thursday, Dec. 4, 2014

Q.1. Consider two different implementations, M1 and M2, of the same instruction set. There are five classes of instructions (A, B, C, D and E) in the instruction set. M1 has a clock rate of 4 GHz and M2 has a clock rate of 6 GHz.

Class	CPI on M1	CPI on M2
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 M1 and M2 expressed in instructions per second?

The peak performance of M1 is when it executes instructions of class A=4 x 10^9 instructions per second.
The peak performance of M2 is when it executes instructions of class A, B or C=6/2 x 10^9 = 3 x 10^9 instructions per second.

(b) If the number of instructions executed in a certain program is divided equally among the classes of instructions, except that for class A, which occurs twice as often as each of the others, how much faster is M2 than M1?

CPI for M1 = (2x1+1x2+1x3+1x4+1x3)/(2+1+1+1+1)=14/6=2.33
CPI for M2 = (2x2+1x2+1x2+1x4+1x4)/(2+1+1+1+1)=16/6=2.67

M2 is faster than M1 by a factor = (IC x 2.33 x 6 x 10^9)/ (IC x 2.67 x 4 x 10^9)= 1.31

Q.2. A benchmark program runs for 100 seconds. We want to improve the speedup of the benchmark by a factor of 3. We enhance the floating-point hardware to make floating point instructions run 5 times faster. How much of the initial execution time would floating-point instructions have to account for to show an overall speedup of 3 on this benchmark?

Speedup = 1 / (f/s + (1-f) => 3 = 1 / (f/5+(1-f) => f/5 + 1-f = 1/3 => f + 5 – 5f = 5/3 => 4f = 3.33 => f =0.833

Thus, floating-point instructions must account for 83.3% of the initial execution time to show an overall speedup of 3 on this benchmark.

Q.3. Consider the following fragment of MIPS code. Assume that **a** and **b** are arrays of words and the base address of **a** is in **$a0** and the base address of **b** is in **$a1**. How many instructions are executed during the running of this code? If ALU instructions (**addu** and **addiu**) take 1 cycle to execute, load/store (**lw** and **sw)** take 5 cycles to execute, and the branch (**bne**) instruction takes 3 cycles to execute, how many cycles are needed to execute the following code (all iterations). What is the average CPI?

```
        addu $t0, $zero, $zero    # i = 0
        addu $t1, $a0, $zero      # $t1 = address of a[i]
        addu $t2, $a1, $zero      # $t2 = address of b[i]
        addiu $t3, $zero, 101     # $t3 = 101 (max i)
loop:   lw $t4, 0($t2)            # $t4 = b[i]
        addu $t5, $t4, $s0        # $t5 = b[i] + c
        sw $t5, 0($t1)            # a[i] = b[i] + c
        addiu $t0, $t0, 1         # i++
        addiu $t1, $t1, 4         # address of next a[i]
        addiu $t2, $t2, 4         # address of next b[i]
        bne $t0, $t3, loop        # loop if (i != 101)
```

The loop body will be executed 101 times. Thus, the total number of instructions executed per class is:

Class	Instruction Count
addu and **addiu**	4 + 101x4 = 408
lw and **sw**	101x2=202
bne	101

Thus, the total number of instruction executed = 408 + 202 + 101 = 711 instruction.
Total number of cycles needed to execute the code = 408x1+202x5+101x3=1721 cycle.
The average CPI = 1721/711 = 2.42