Name: **Id#**

COE 301/ICS 233, Term 172

Computer Architecture & Assembly Language

Quiz# 7 Solution

Date: Tuesday, May 1, 2018

Q1. 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.

Q2. 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.

Q3. We want to compare the performance of a **single-cycle CPU design** with a **multicycle CPU**. Suppose we add the multiply and divide instructions. The operation times are as follows:

Instruction memory access time = 190 ps, Data memory access time = 190 ps
Register file read access time = 150 ps, Register file write access = 150 ps
ALU delay for basic instructions = 190 ps, Delay for multiply or divide = 550 ps

Ignore the other delays in the multiplexers, control unit, sign-extension, etc.

Assume the following instruction mix: 30% ALU, 15% multiply & divide, 30% load & store, 15% branch, and 10% jump.

i. What is the total delay for each instruction class and the clock cycle for the single-cycle CPU design?

Instruction Class	Instruction Memory	Register Read	ALU Operation	Data Memory	Register Write	Total
ALU	190	150	190		150	680 ps
Load	190	150	190	190	150	870 ps
Store	190	150	190	190		720 ps
Branch	190	150	190			530 ps
Jump	190					190 ps
Mul/div	190	150	550		150	1040 ps

Clock cycle = 1040 ps determined by the longest delay.

ii. Assume we fix the clock cycle to 200 ps for a multi-cycle CPU, what is the CPI for each instruction class and the speedup over a fixed-length clock cycle? Note that this implies that multiply and divide operations will be performed in multiple cycles.

Instruction Class	CPI
ALU	4
Load	5
Store	4
Branch	3
Jump	2
Mul/div	6

Average CPI= 4*0.3 + 5*0.15 + 4*0.15 + 3*0.15 + 2*0.1 + 6*0.15=4.1
Note that we assumed that load and store instructions have equal percentage.
Speedup = 1040 ps / (4.1*200 ps) = 1.268.