

CSE 775  
Autumn 2008  
Homework #3

Instructor: Panda

Due: Monday, 24th November

1. (15 points) Consider the pipelined execution sequence of the following MIPS instructions. Assume 2 stages for floating-point addition and 10 stages for floating-point division. Assume interrupt for a floating-point operation occurs during the last stage of the EX step of that operation. Assume no interrupt occurring in the IF and ID steps.

LD	r1, 30(r2)
L.D	f1, 1000(r3)
L.D	f2, 2000(r3)
DADD	r5, r1, r3
ADD.D	f1, f1, f2
DIV.D	f2, f1, f2
SD	40(r2), r4
SD	3000(r3), f2

- (a) For each instruction, indicate all possible stages where *within* interrupts could occur.
  - (b) Determine the number of cycles for executing this program. Out of these cycles, how many cycles are prone to interrupt?
2. (25 points) Consider a 4.0 GHZ system and an user program with 50% load/store, 20% ALU, and the remaining as branch instructions. Let these instructions take 8, 6, and 5 clock cycles for execution, respectively.

Consider 10% of load/store and 5% of ALU instructions generating *within* interrupts. The load/store interrupt service routine has 15 load/store, 20 ALU, and 10 branch instructions. Similarly, the corresponding interrupt service routine for handling ALU interrupts has 20 load/store, 10 ALU, and 10 branch instructions. Assume that the load/store and ALU instructions inside these interrupt service routines DO NOT lead to any interrupt.

How slow does the user program run in the presence of interrupts?

3. (30 points) Consider pipelined execution of the following loop. f2 is loaded with a scalar value initially. Assuming the pipeline latencies from Fig. 2.2 of the textbook, unroll the following loop as many times as necessary to schedule it without any delays and collapsing the loop overhead instructions. Show the new schedule and determine the number of cycles taken per iteration of the loop.

Loop:	LD	f0, 1000(r1)
	MULT.D	f0, f0, f2
	LD	f4, 2000(r1)
	ADD.D	f0, f0, f4
	LD	f6, 3000(r1)

```
ADD.D    f0, f0, f6
SD       4000(r2), f0
DSUBI    r1, r1, #8
DSUBI    r2, r2, #8
BNEZ     r1, loop
```

4. (15 points) Consider the scoreboard design with one read-port port register file. Show a MIPS code sequence (involving 3-4 instructions) where the register file will be the bottleneck. Does the same problem arise in Tomasulo? Explain your answer.
5. (15 points) Consider Tomasulo algorithm and the design with one Common Data Bus (CDB). Only one write can take place on CDB at any given time. Show a MIPS code sequence (involving 3-4 instructions) where the CDB will be the bottleneck. Explain your answer.