CSE 378
Assignment 6
Due Monday, May 21, 2001

Do the following problems and hand in the solutions before class on Monday, May 21.

1) Problem 6.15

2) Problem 6.16

3) Problem 6.17

4) Problem 6.20

5) Problem 6.26

6) Consider the following pipeline for a high speed version of the R3000.

IF1	Instruction fetch starts.
IF2	Instruction fetch completes.
ID	Instruction decode and register fetch; begin computing branch target.
EX	Effective address or ALU result available; branch condition tested; branch target computation completed.
MEM1/ ALUW	First part of memory cycle; write of ALU results to the register file.
MEM2	Memory access completes.
LW	Write the value for a load instruction into the register file.

Assume that the PC is stored in a separate register, and has its own incrementer. Also assume that there are separate memory ports to both the instruction and data caches, so that both types of accesses can take place in the same cycle. The general purpose register file can be written and read in different phases of the same cycle.

The pipeline looks like this drawn out. (MEM1/ALUW is abbreviated M1 so that the pipeline fits on the page; ditto with MEM2 and M2.)

IF1	IF2	ID	EX	M1	M2	LW									
	IF1	IF2	ID	EX	M1	M2	LW								
		IF1	IF2	ID	EX	M1	M2	LW							
			IF1	IF2	ID	EX	M1	M2	LW						
				IF1	IF2	ID	EX	M1	M2	LW					
					IF1	IF2	ID	EX	M1	M2	LW				
						IF1	IF2	ID	EX	M1	M2	LW			
							IF1	IF2	ID	EX	M1	M2	LW		
								IF1	IF2	ID	EX	M1	M2	LW	
									IF1	IF2	ID	EX	M1	M2	LW

a) How many cycles is the branch delay?

b) How many cycles is the load-use delay (the delay between when a load instruction produces a value and a consumer instruction needs it)? Assume there is forwarding.

c) This question asks you to identify two different data hazards that are caused by load instructions. To answer the question, fill in the table below. For each data hazard:

- First, show an example of a two instruction code sequence (two instructions, one right after the other) that will cause the hazard. The first instruction must be a **`lw`**. Since you are identifying two different data hazards, the instruction in each code sequence other than the **`lw`** must be in a different class of instructions. (Remember that all Rtype instructions are in the same class, computational.) The second instruction cannot be another **`lw`**.
- Second, if forwarding will either eliminate or reduce the delay caused by the hazard, say between what two stages it will occur. If forwarding does not help, i.e., the hazard cannot be eliminated or reduced, say so. If forwarding is not needed, i.e., there is no delay, say so.
- If you used forwarding in the last part, how many stalls remain after forwarding, if any?

As an example, one data hazard that does not involve a load has been done for you. Only the registers that are involved in the hazard are written explicitly.

An example code sequence	Stages involved in forwarding	# of stall cycles remaining
add $4, -, - add -, $4, -	EX to EX	0

You can use the pipeline on the previous page to help you detect the hazards, construct forwarding logic and find stalls.