EEL308: Computer Architecture
MAJOR

Answer all questions Max. Marks: 40; Time: 2 hrs

---

1. You are going to enhance a machine, and there are two possible improvements: either make multiply instructions run four times faster than before, or make memory access instructions run two times faster than before. You repeatedly run a program that takes 100 seconds to execute. Of this time, 20% is used for multiplication, 50% for memory access instruction, and 30% for other tasks.
   (a) What will be the speedup if you improve only multiplication?
   (b) What will be the speedup if you improve only memory access?
   (c) What will be the speedup if both improvements are made?
   (d) If you change the above program so that the percentages are not 20%, 50%, and 30% anymore. Assuming that none of the new percentage is 0, what sort of program would result in a tie (with regard to speed up) between the two individual improvements? Provide a formula. (2+2+2+2)

2. Consider **Fig. 5.32** for multi-cycle implementation. Indicate which categories of instruction will work/ will not work under each of the following single fault conditions (stuck -at -0 or stuck-at -1 fault) shown in column 1 of the following table. **(Fault condition stuck-at-0 means that a particular signal is stuck at value 0 due to some fault in logic circuit regardless of what it should be).** Similarly for stuck-at-1 fault a signal is stuck at 1 regardless of what it should be. For example if value of RegDst is stuck at 0 the corresponding multiplexer will not provide correct data path for R-type instruction, while other categories of instruction will not be affected due to this fault. Answer for this example is given in one of the rows for your benefit. **You have to fill in the table similarly with Yes/No for other fault conditions.** (7)

Single Fault Conditions	Instruction type that will work (Yes/No)				
	R-type	**Load word**	**Store word**	**beq**	**jump**
Example					
RegDst = 0	NO	YES	YES	YES	YES
1. RegDst = 1					
2. MemtoReg = 0					
3. MemtoReg = 1					
4. IorD = 0					
5. IorD = 1					
6. ALUSrcA = 0					
7. ALUSrcA = 1					

3. Write a Procedure *sum* in MIPS instruction along with the **comments** to compute the following.

   S = 1 + 3 + 5 + 7 +. . . + 21

The above numbers are decimal numbers (base 10). Assume that the last number '21' is stored in $a0 by the main program before calling the Procedure *sum*. Use this information to exit from the procedure *sum* that you are writing. The program should save the sum **S** in $v0 before returning to the main program using $ra. You may use registers $t0, $t1, and $t2 for computing intermediate values within the program.

(5)

4. Here is a series of address references given as word addresses:
1, 4, 8, 5, 20, 17, 19, 56, 9, 11, 4, 43, 5, 6, 9, 17

Show the hits and misses and **final of cache** contents for a fully associative cache with one-word blocks and a *total size* of 16 words. Give your result in tabular form. Assume the cache to be initially empty and also assume LRU replacement.

(3+3)

5. Let us be concerned with caches of unusual sizes. Can you make a fully associative cache containing 300 words of data? How about a set associative cache or direct-mapped cache containing exactly 300 words of data? For each of these, describe how or why not. Remember that $300 = 3 * 10^2$. (6)

6. Consider executing the following code on the pipelined datapath of **Fig.6.46** :

```
Add   $1,  $2,  $3
Add   $4,  $5,  $6
Add   $7,  $8,  $9
Add   $10, $11, $12
Add   $13, $14, $15
```

At the end of fifth cycle of execution, which registers are being read and which registers will be written? Explain what the forwarding unit and the hazard detection unit are doing during the fifth cycle of execution. If any comparisons are being made, mention them. List all the inputs and outputs of the forwarding unit in Fig. 6.46 and give the names and the number of bits for each input and output.

(2+2+2+2)