Exam 1

CIS 501, Spring 1998
Sample exam

1. Consider a workstation with no floating-point hardware which has to emulate FP operations with sequences of integer operations. Its clock cycle time is 10 ns. The following table lists the FP operation types and their relative frequencies, as well as the number of integer instructions needed to emulate each type of FP operation.

FP Operation	Frequency	# Integer instr to emulate
Add	50%	6
Multiply	40%	10
Divide	10%	20

Now consider adding a floating-point coprocessor to the workstation which allows FP operations to be performed directly (i.e. without emulation). However, adding the coprocessor will increase the cycle time by 20%. Consider a benchmark workload of 150 million instructions, 20% of which are FP operations and 80% of which are integer instructions. Assume a base CPI of 1 for integer instructions (and a CPI of 1 for FP instructions executed by the coprocessor).

(a) (6 points) Calculate the MIPS rating for each machine.

(b) (6 points) Calculate the speedup from adding the coprocessor by comparing execution times.

(c) (6 points) Assuming the coprocessor could be added without increasing the cycle time, calculate the speedup from adding it using Amdahl's Law.

2. Consider the standard DLX FP pipeline (i.e. with a pipelined floating point adder with a 4 cycle latency, and a pipelined FP multiplier with a 7 cycle latency). Furthermore, assume the pipeline uses delayed branches with a branch delay slot, and that branch resolution has been moved up into the ID stage. Thus the branch target may begin its IF stage at the same time the branch is beginning its EX stage. Consider the following segment of code:

```
loop: LD    F0, 0(R2)
      ADDI  R2, R2, #8
      MULTD F0, F0, F4
      SUB   R5, R4, R2
      BNZ   R5, loop
      ADDD  F2, F0, F2    ; branch delay slot
```

(a) (10 points) Show the timing for one iteration of this loop, assuming that no forwarding is available, but that a register fetch done in the same cycle as a write back will retrieve the value just written.

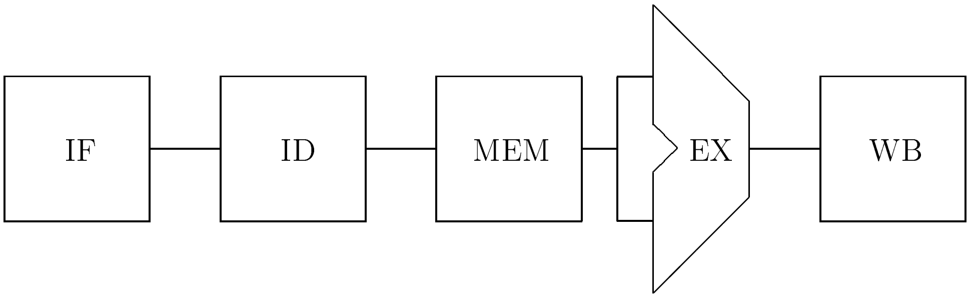
	clock cycle																			
Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
LD F0,0(R2)																				
ADDI R2, R2, #8																				
MULTD F0, F0, F4																				
SUB R5, R4, R3																				
BNZ R5, loop																				
ADDD F2, F0, F2																				
LD F0,0(R2)																				

(b) (10 points) Show the timing for one iteration of ths loop, assuming that all possible forwarding is available (and again that a register fetch done in the same cycle as a write back will retrieve the value just written).

	clock cycle																			
Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
LD F0,0(R2)																				
ADDI R2, R2, #8																				
MULTD F0, F0, F4																				
SUB R5, R4, R3																				
BNZ R5, loop																				
ADDD F2, F0, F2																				
LD F0,0(R2)																				

(c) (4 points) Explain why this loop cannot be rescheduled to produce fewer stalls per iteration.

3. We are designing a new machine, the Eniac-2001, and want to eliminate the load stalls found in the DLX. To achieve this, we decide to only support the register indirect addressing mode for loads and stores. This change lets us exchange the order of the EX and MEM stages in the pipeline (see diagram) and thus eliminate the load stall. The rest of the pipe is the same as the unoptimized DLX pipeline you implemented in the project.



(a) (5 points) Show how you would synthesize a register displacement instruction like `LW R1, 200(R3)` on the Eniac-2001.

(b) (5 points) How many cycles does it take to execute the instructions used for the synthesis?

(c) (7 points) Explain how this change to the pipeline affects control hazards.

(d) Explain how this change to the pipeline affects data forwarding/stalls for:

i. (6 points) ALU ops following ALU ops

ii. (6 points) Load/stores following ALU ops

iii. (6 points) ALU ops following Loads

(Question 3, continued)

(e) (5 points) Explain how you could determine if the new pipeline would have improved performance.

4. Suppose we now also wanted to further change the Eniac-2001 from Problem 3 to once more support a displacement addressing mode for loads and stores by including an adder in the ID stage for effective address calculation. Call this machine the Eniac-2001++.

   (a) (9 points) How would the data forwarding of the Eniac-2001++ be different from the Eniac-2001?

   (b) (9 points) Would the performance of the Eniac-2001++ be better than that of the Eniac-2001? Why or why not?