CS232 Midterm Exam 2
April 8, 2002

Name: ____

- This exam has 7 pages, including this cover.
- There are three questions worth a total of 100 points.
- You have 50 minutes. Budget your time!
- No written references or calculators are allowed.
- To make sure you receive full credit, write clearly and show your work.
- We will not answer questions regarding course material.

Question	Maximum	Your Score
1	30	
2	35	
3	35	
Total	100	

Question 1: Single-cycle datapath (30 points)

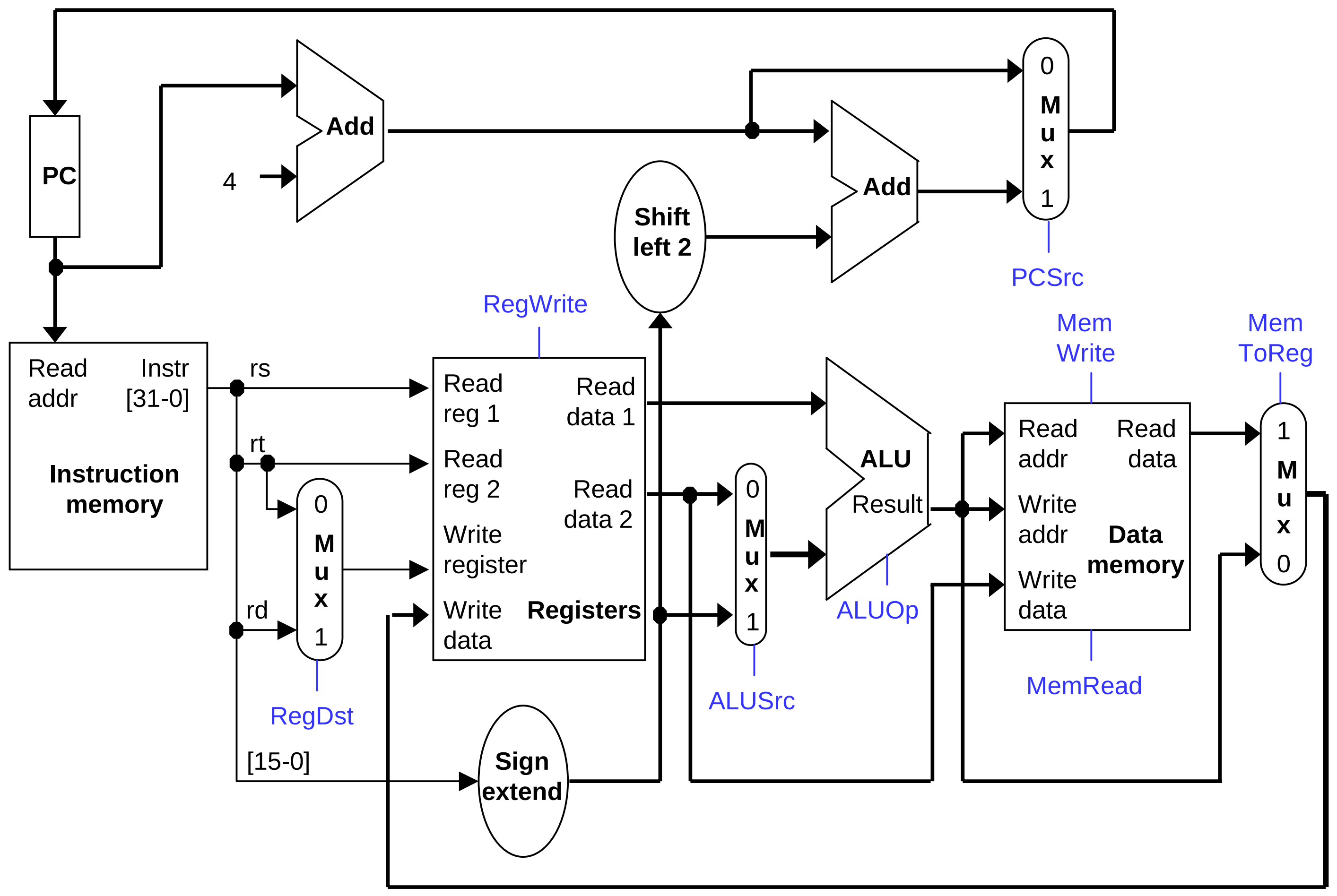
Let's simplify the MIPS instruction set architecture a little by *removing* the original lw and sw instructions and replacing them with ones that do not contain a constant offset. Our new loads and stores will have the following general forms.

```
lw  rt, rs        # rt = Mem[rs]
sw  rt, rs        # Mem[rs] = rt
```

These are I-type instructions, but you may *not* make any assumptions about the instructions' constant fields.

Part (a)

Show what changes must be made to the single-cycle datapath below so the new lw and sw instructions can avoid going through the ALU. Please try to keep your modifications as neat as possible! (10 points)



Question 1 continued

Part (b)
Fill in the table below to show the correct control signals for the new lw and sw instructions. You must write ‘X’ to indicate any don’t-care conditions. (5 points)

	RegDst	RegWrite	ALUSrc	ALUOp	MemWrite	MemRead	MemToReg	PCSrc
lw								
sw								

Part (c)
Assume that memories and the ALU have 2ns delays, and the registers have a 1ns delay. Find the minimum clock cycle times for *both* the original single-cycle datapath and your modified one. (5 points)

Part (d)
What general conclusions can you draw, if any, about the performance of the original processor as compared to your modified one? (10 points)

Question 2: Multicycle CPU implementation (35 points)

MIPS is a register-register architecture, where arithmetic source and destinations must both be registers. But let's think about including a register-memory addition instruction.

addm rd, rs, rt   # rd = rs + Mem[rt]

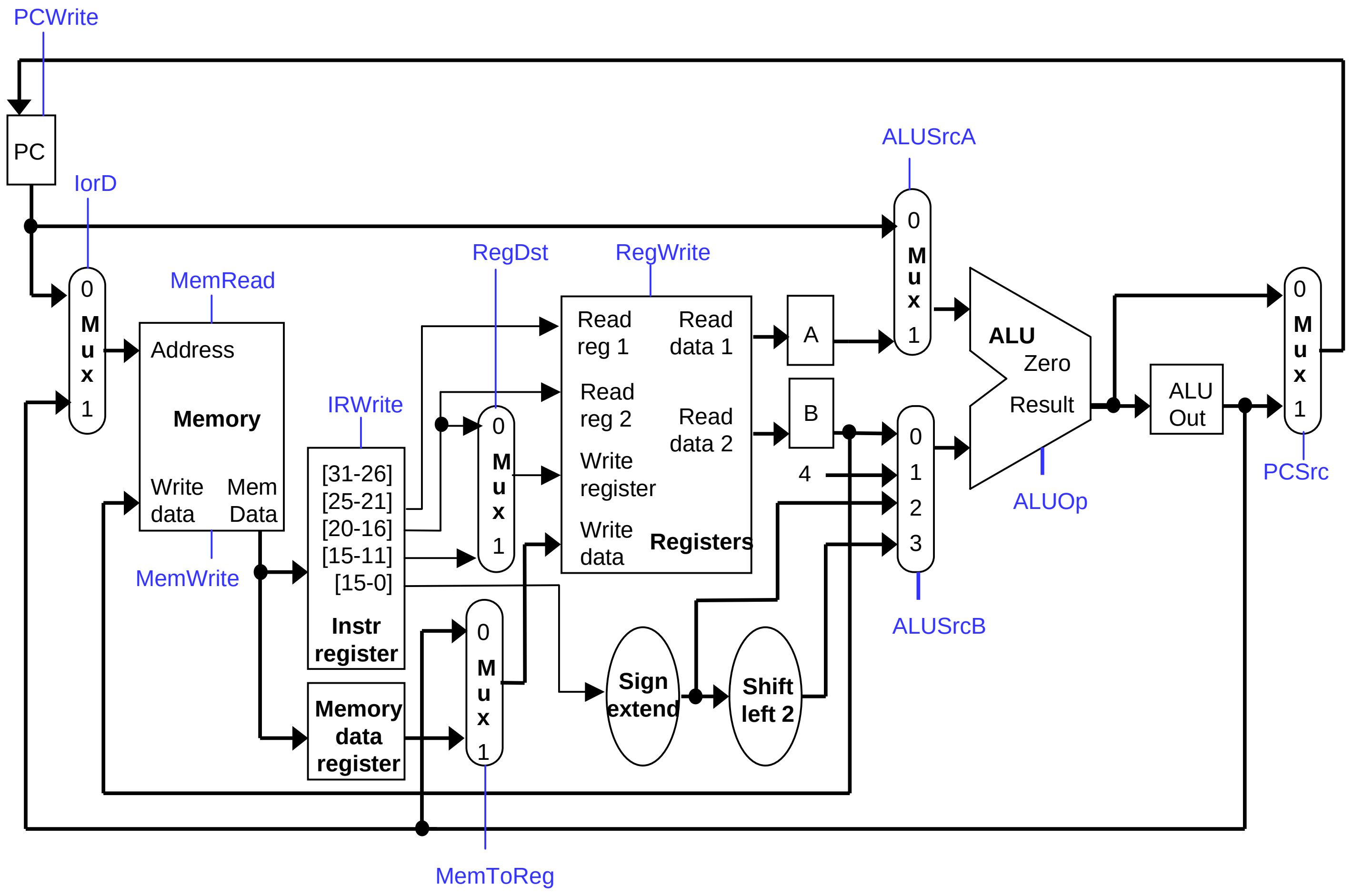
In other words, register rt contains a memory address which is read to produce the ALU's operand. The instruction format is given here for your reference (shamt and func are not used).

Field	op	rs	rt	rd	shamt	func
Bits	31-26	25-21	20-16	15-11	10-6	5-0

It is possible to include *addm* in a multicycle processor by modifying the datapath and control unit presented in class. Your implementation of *addm* should not need more than five stages for execution, and you should be careful not to modify any registers other than rd.

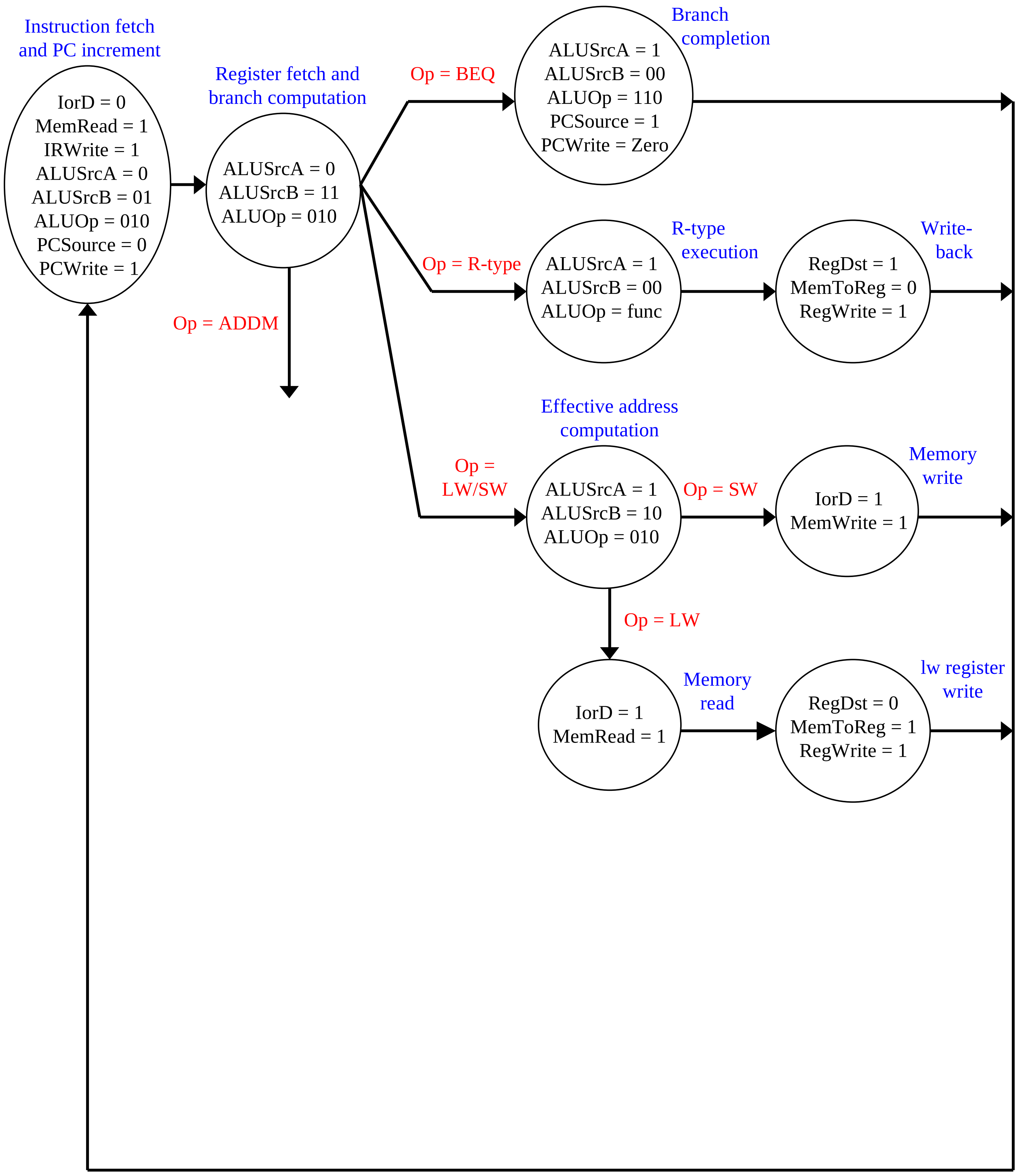
Part (a)

The multicycle datapath from lecture appears below. Show the changes needed to support *addm*. Try to keep your diagram neat! (10 points)



Question 2 continued

Part (b)
Complete this finite state machine diagram for the *addm* instruction. Control values not shown in each stage are assumed to be 0. Remember to account for any control signals that you added or modified in the previous part of the question. (25 points)



Question 3: Forwarding and stalling (35 points)

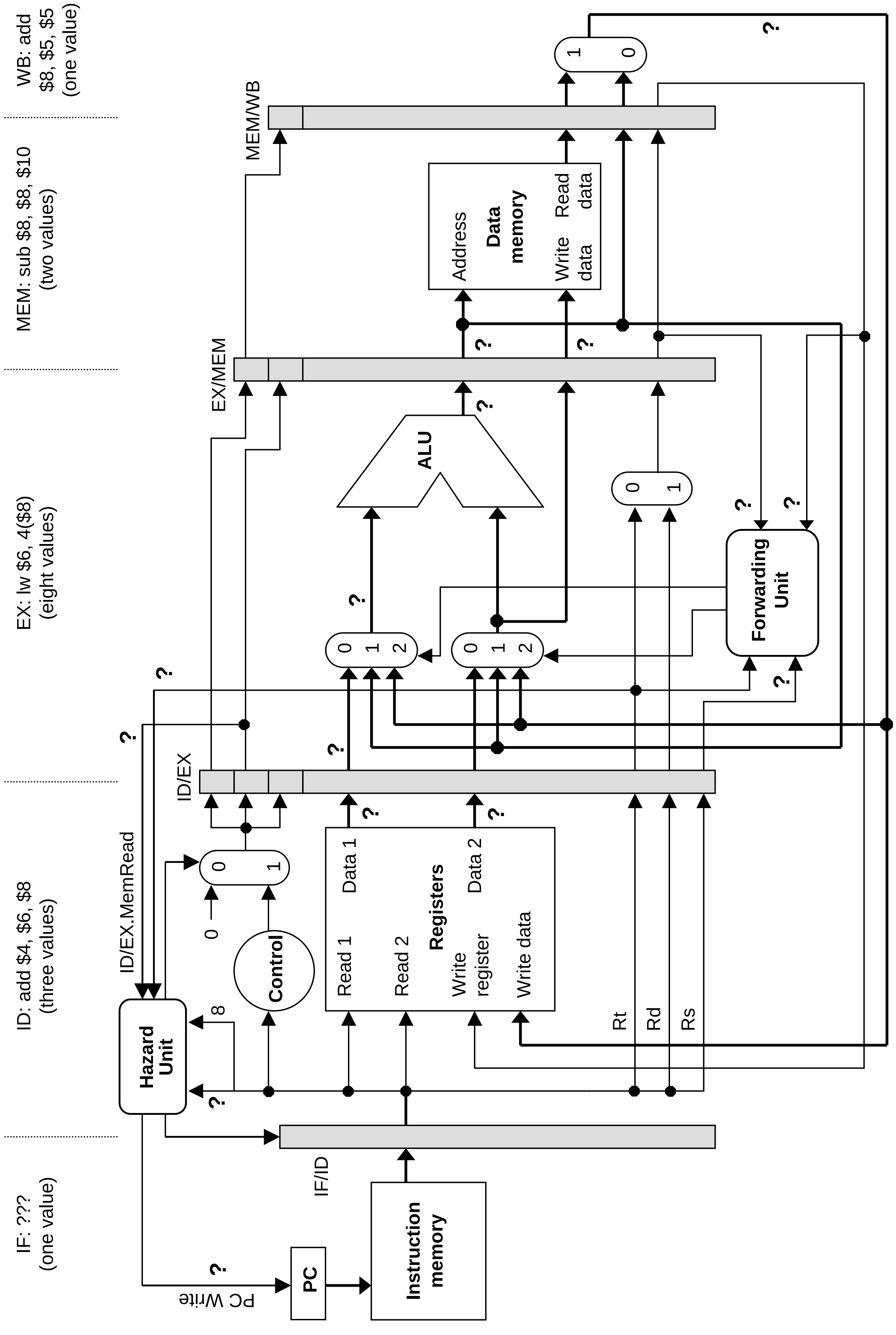
Part (a)
Show or list all of the dependencies in the following code fragment. Make sure that you clearly indicate which instructions *and* registers are involved in each dependency. (5 points)

```
add     $8,    $5,    $5

sub     $8,    $8,    $10

lw      $6,    4($8)

add     $4,    $6,    $8
```

Part (b)
The pipelined datapath on the next page shows the fifth cycle of executing this program. Fill in the correct datapath values for the *fifteen* question mark symbols **?** in the diagram. There is one **?** in the IF stage, three in the ID stage, eight in EX, two in MEM, and one in WB. (30 points; 2 points each)

- Assume that registers initially contain their number plus 100: $6 contains 106, $8 contains 108, etc.
- Write your values directly on the diagram, but please write clearly.
- Use decimal notation. You may write 'X' for any values that cannot be determined.

Question 3 continued


IF: ???
(one value)
ID: add $4, $6, $8
(three values)
EX: lw $6, 4($8)
(eight values)
MEM: sub $8, $8, $10
(two values)
WB: add $8, $5, $5
(one value)
PC Write
PC
Instruction memory
IF/ID
Hazard Unit
ID/EX.MemRead
8
0
Control
Read 1
Read 2
Write register
Write data
Registers
Data 1
Data 2
Rt
Rd
Rs
ID/EX
Forwarding Unit
ALU
EX/MEM
Address
Write data
Data memory
Read data
MEM/WB
0
1
2
?