13

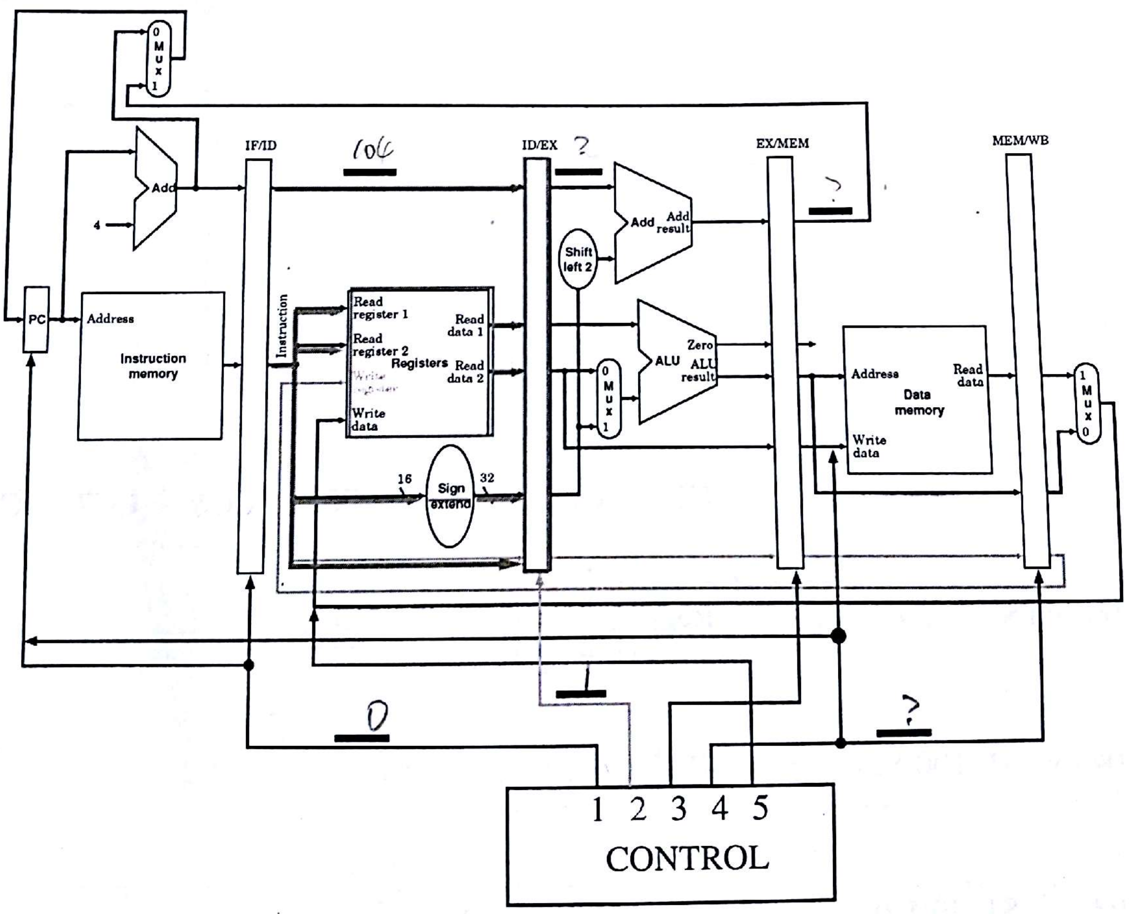
/10

CS 251, Fall 2016, Assignment 4.1.1

3% of course mark

Due Wednesday, November 9, 4:30PM

1. (5 points) In the diagram below, the multicycle computer from the course notes is executing the instruction `100 add $1,$2,$3` in the ID state. The ID/EX register bank has NOT yet been written. In the figure below, there are six dark lines, each above one of the data lines in the datapath. Above each dark line write the value on the dataline; if the value on the data line can not be determined from the information given, write '?'.

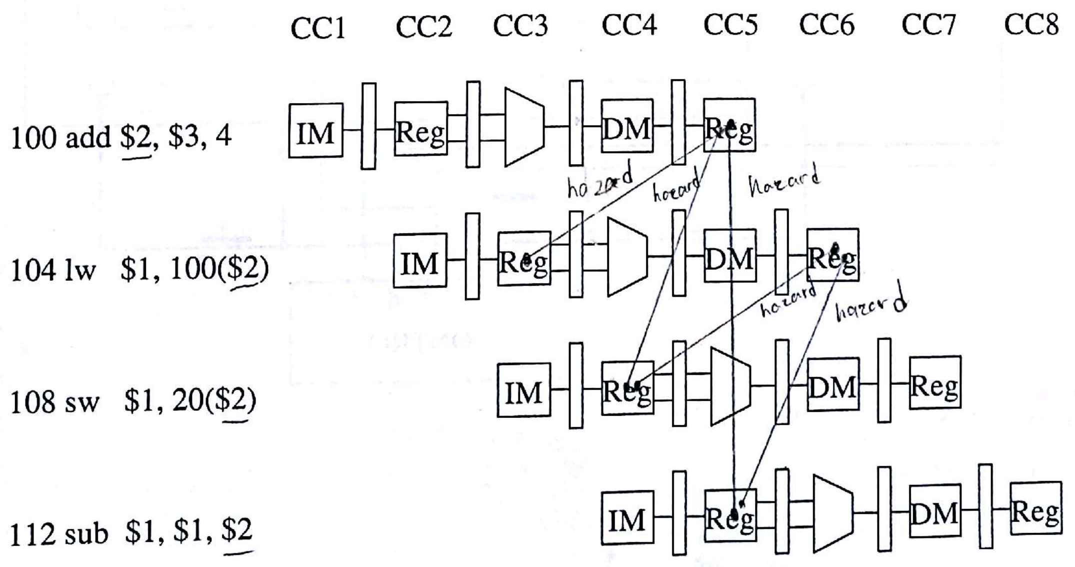


2. (10 points) Consider the following MIPS code sequence:

```
100 add  $2, $3, $4
104 lw   $1, 100($2)
108 sw   $1, 20($2)
112 sub  $1, $1, $2
```

(a) (5 pts) In the diagram below, each row is labeled with an executed instruction. Mark all *data dependencies* by drawing straight lines (similar to the grey lines of on page 6-9 of the notes (the blue lines of Fig. 4.52 of the 5th edition) between when the result is in the register file and when it needs to be taken from the register file. Assume that the code is to run on the pipelined datapath of page 6-8 of the notes (Fig. 4.51 of the 5th edition; this datapath implements neither stalling nor forwarding).

For each data dependency, label it either as a **hazard** or as a **non-hazard**.



(b) (5 pts) Modify the code to remove the data hazards by inserting a **minimum number** of NOPs. In your solution, give both new and old line numbers.

New Line #	Old line #	Code
100	100	add \$2 \$3 \$4
104	—	NOP
108	—	NOP
112	—	NOP
116	104	lw \$1 , 100(\$2)
120	—	NOP
124	—	NOP
128	—	NOP
132	108	sw \$1 20(\$2)
136	112	sub \$1 \$1 \$2

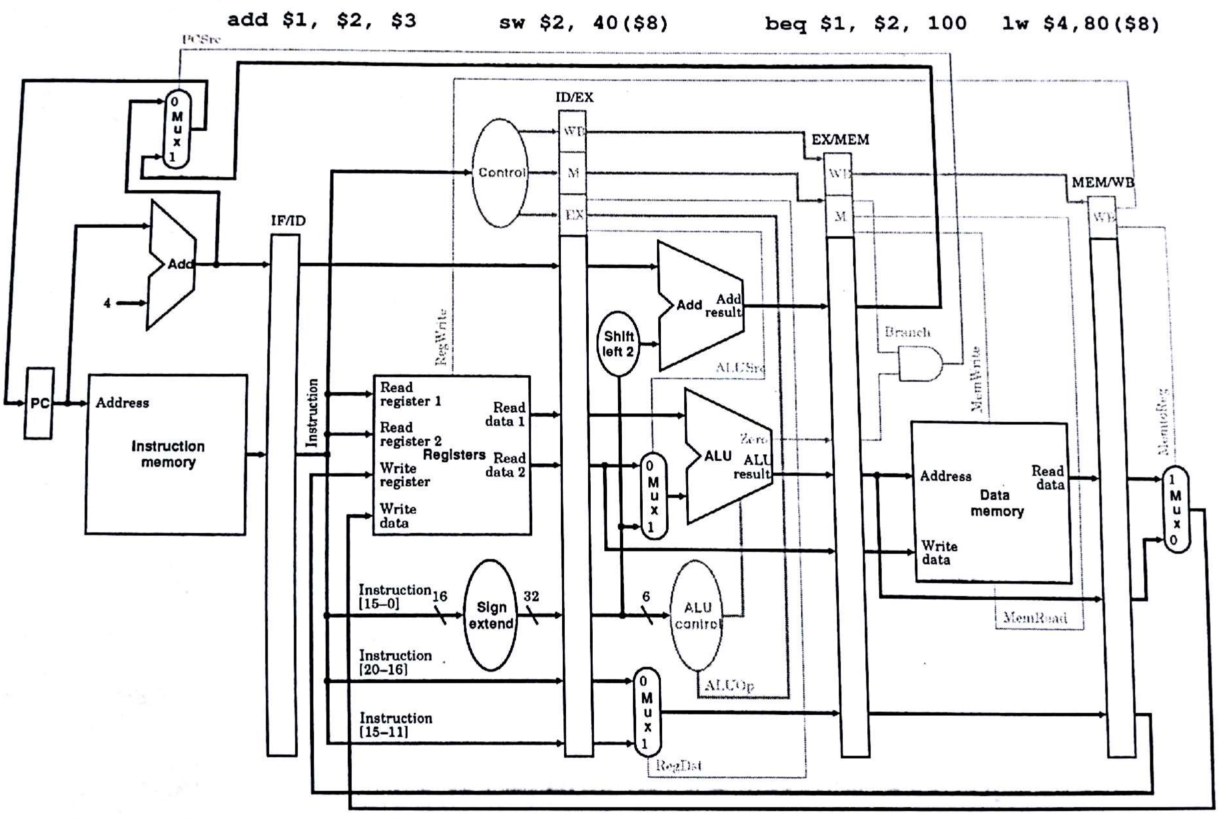
3. (5 points) This question refers to the pipelined datapath without forwarding, shown below and in Figure 4.51 in the textbook.

Consider the instructions

```
100 lw $4, 80($8)
104 beq $1, $2, 100
108 sw $2, 40($8)
112 add $1, $2, $3
```

Consider the pipeline when the 100 `lw` instruction is in the WB stage, the 104 `beq` instruction is in the MEM stage, the 108 `sw` instruction is in the EX stage, and the 112 `add` instruction is in the ID stage. In the figure below, label all of the *control signals* (including both those coming directly out of the control unit and those coming out of the pipeline registers) with their appropriate values.

In this figure, the instructions have been drawn above the appropriate set of pipeline registers. We have also filled in the solution for the WB stage, and given the names of the control signals whose values you need to determine for the other three stages.



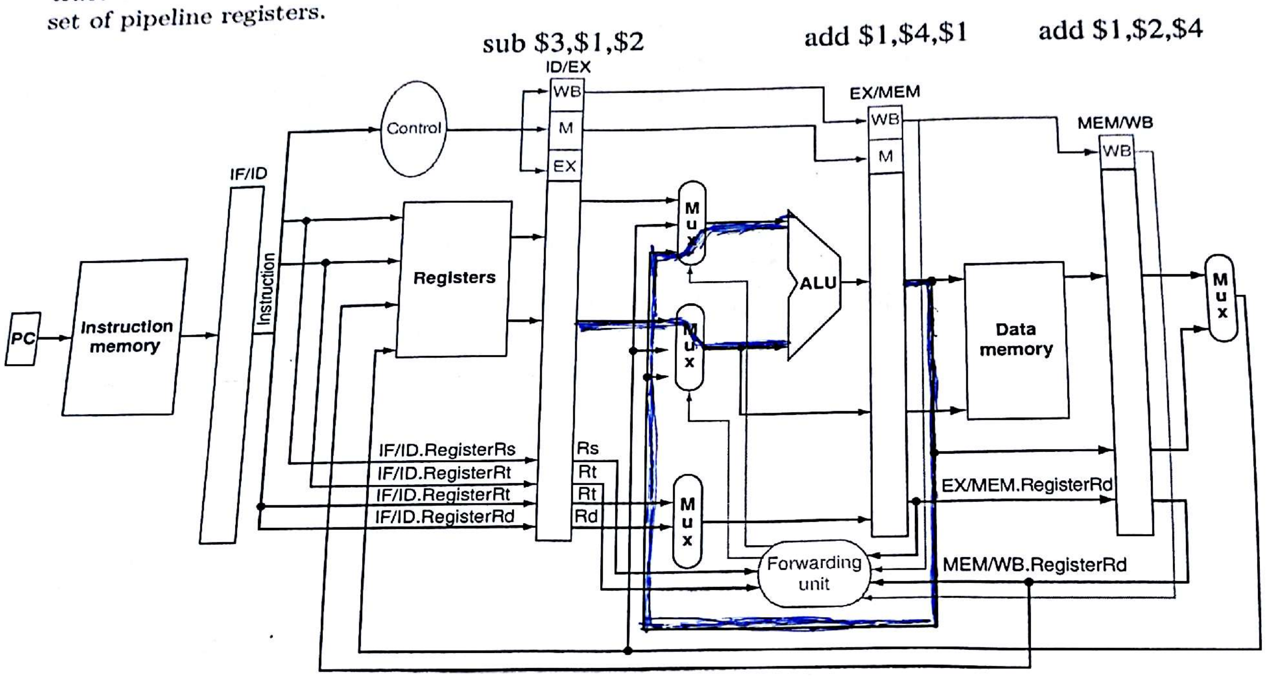
MemtoReg= 0	RegWrite= 1	MemtoReg= X	RegWrite= 0	MemtoReg= X	RegWrite=1
Branch= 0	MemRead= 0	Branch= 0	MemRead= 0	RegWrite= 0	MemToReg=1
MemWrite= 0		MemWrite= 1		Branch= 1	
RegDst= 1		RegDst= X		MemRead= 0	
ALUOp= 10		ALUOp= 00		MemWrite= 0	
ALUSrc= 0		ALUSrc= 1			

4. (5 points) This question refers to the pipelined datapath with forwarding in Figure 4.56 of the textbook.

Consider the instructions

```
100 add $1, $2, $4
104 add $1, $4, $1
108 sub $3, $1, $2
```

Consider the situation when the 100 add instruction is in the WB stage, the 104 add instruction is in the MEM stage, and the 108 sub instruction is in the EX stage. In the figure below, trace back each of the two inputs to the ALU through the MUXes back to the appropriate set of pipeline registers.



5. (5 pts) Given the following code segment below, re-write the code (code rearrangement) to avoid as many stalls and hazards as possible. Assume the datapath *implements data forwarding and stalling* but does not implement Branch Flushing. Indicate any instructions that have a data hazard between itself and a prior instruction using (*) beside that instruction. You may insert NOP instructions, only if code rearrangement cannot be used. Rewrite the code and perform code rearrangement where necessary to remove data hazards and control hazards.

Original	Code Rearrangement
100 add $4, $3, $7	100 add $4 $3 $7
104 add $5, $5, $6	104 add $5 $5 $6
108 lw $6, 100($5)	108 lw $6 100($5)
112 lw $7, 0($6)	112 sub $3 $3 $4
116 sw $7, 100($5)	116 lw $7 0($6)
120 lw $6 , 100($7)	* 120 sw $7 100($5)
124 add $5, $6, $4	124 lw $6, 100($7)
128 slt $7, $4, $6	* 128 add $5 $6 $4
132 sub $3, $3, $4	132 slt $7 $4 $6
136 addi $5, $5, 4	136 addi $5 $5 4