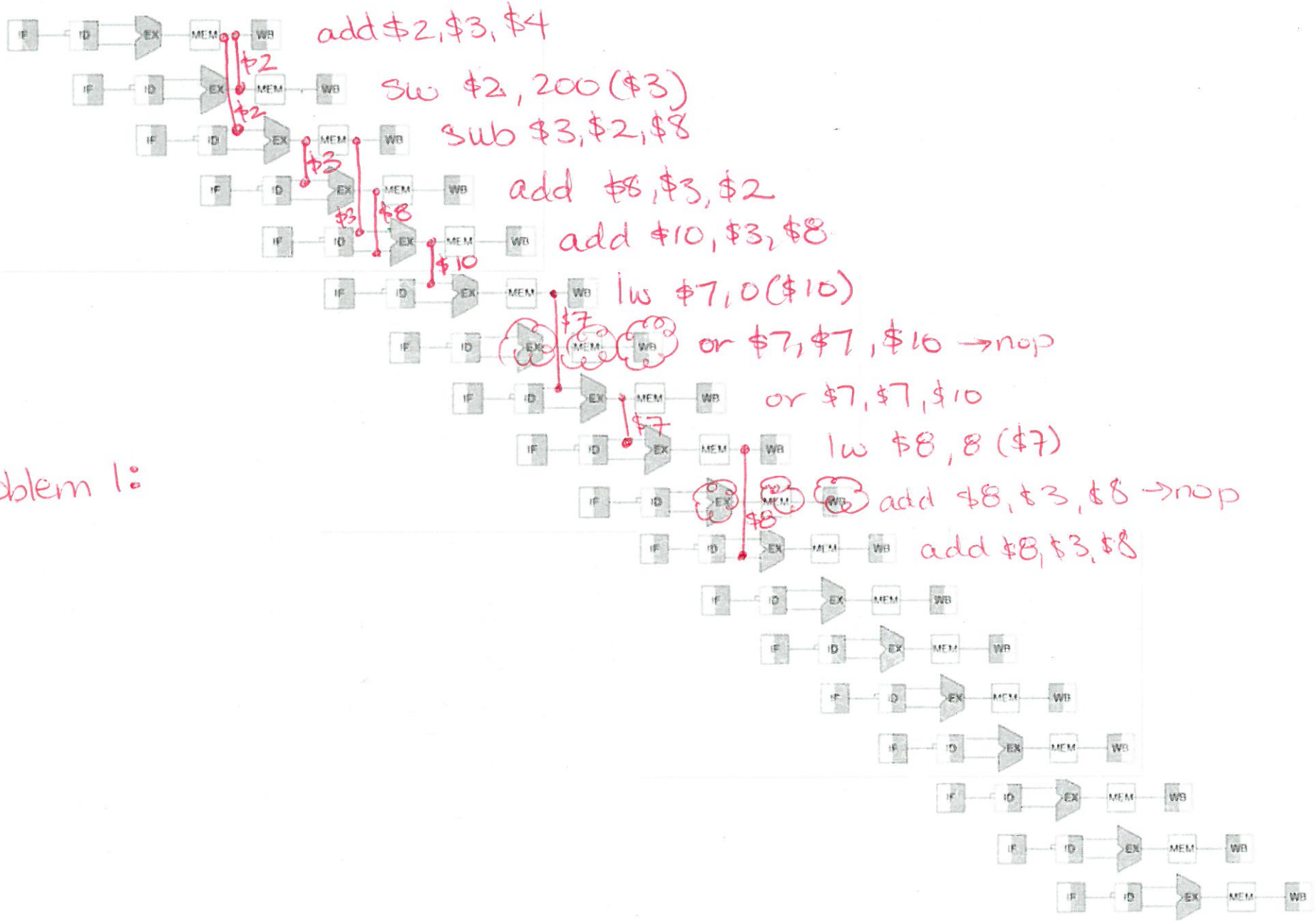
Problem 1:


add $2, $3, $4
sw $2, 200($3)
sub $3, $2, $8
add $8, $3, $2
add $10, $3, $8
lw $7, 0($10)
or $7, $7, $10 → nop
or $7, $7, $10
lw $8, 8($7)
add $8, $3, $8 → nop
add $8, $3, $8
IF
ID
EX
MEM
WB

Problem 2:

MIPS Pipeline Stages

Problem 4:

(a) Fetch & Ex: ALU (PC+4) Hazard, Mem Hazard if combined Instr/Data Memory

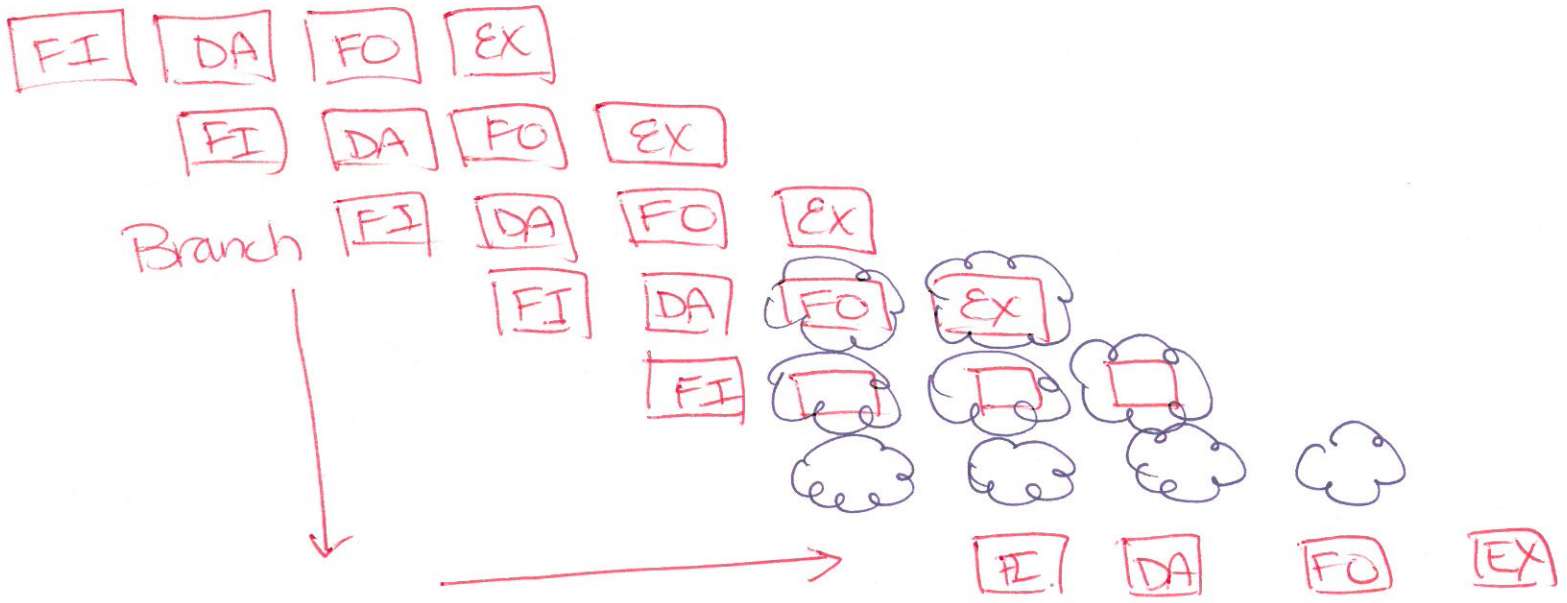
Fetch & DA & Ex: ALU Hazard if only 1 unit

(b) No data Hazards. All calculations & RegWrite occur in same cycle (EX). Therefore values written back before FO.

(c) None. No data Hazards

(d) No stalls for lw.
Stalls for branch only. Control Hazards!

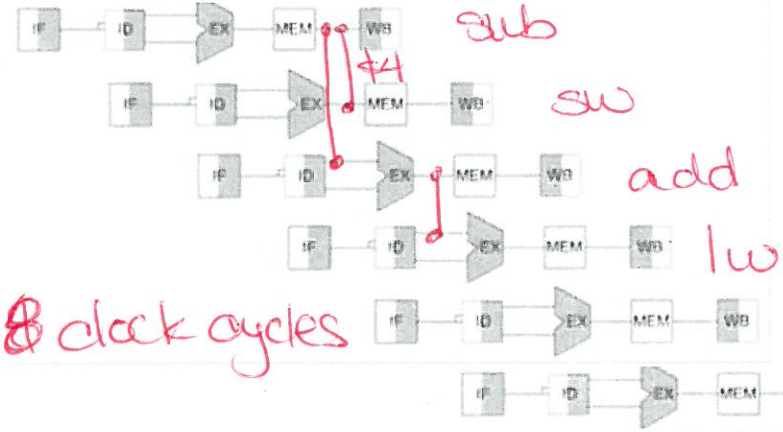
(e)



Problem 3

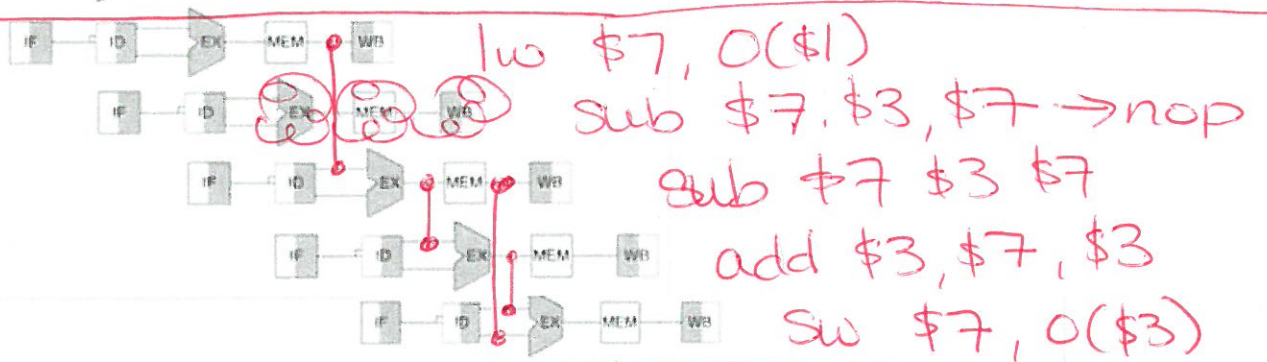
(a)
```
sub $4, $0, $5
sw  $4, 0($6)
add $6, $4, $5
lw  $4, 0($6)
```
all data hazards

(b) no stalls

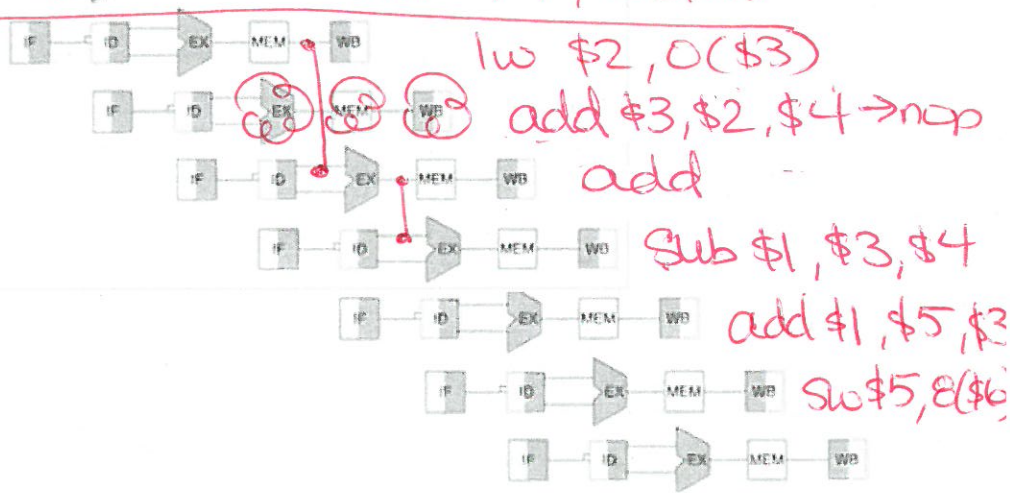


(c) 8 clock cycles

Problem 5: (a) Code 1



Code 2



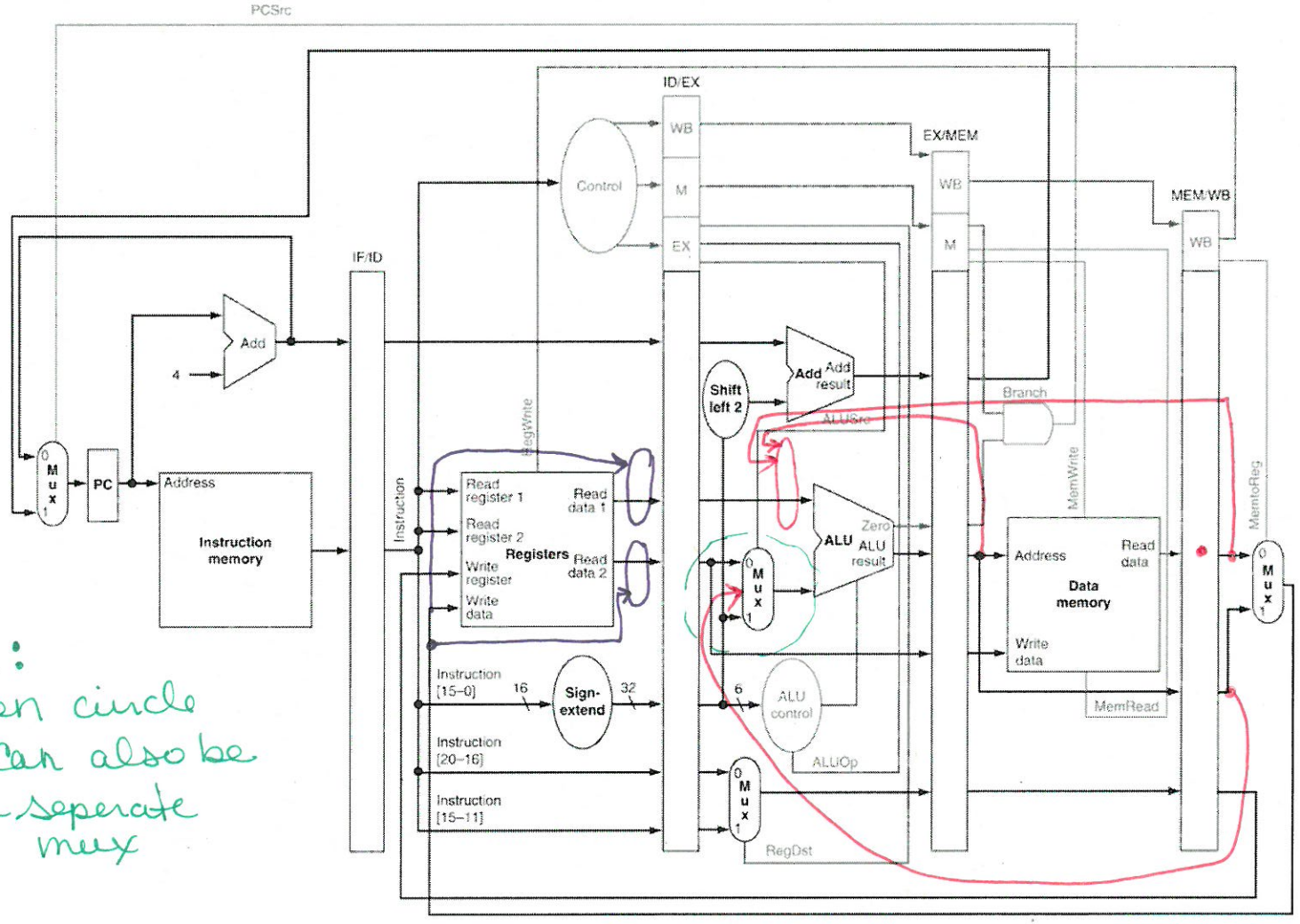
(b) Code 1 can not be reordered

Code 2

```
lw  $2, 0($3)
sw  $5, 8($6)
add $3, $2, $4
sub $1, $3, $4
add $1, $5, $3
```
dims stall

Problem 5
(b) (d)

Instruction	Execution/address calculation stage control lines				Memory access stage control lines			Write-back stage control lines	
	RegDst	ALUOp1	ALUOp0	ALUSrc	Branch	Mem-Read	Mem-Write	Reg-Write	Memto-Reg
R-format	1	1	0	0	0	0	0	1	0
lw	0	0	0	1	0	1	0	1	1
sw	X	0	0	1	0	0	1	0	X
beq	X	0	1	0	1	0	0	0	X



Note:
green circle can also be a seperate mux

Pipelined Datapath w/ Control Signals

Purple muxes are for Part (d) only