CS2100

Tutorial #10

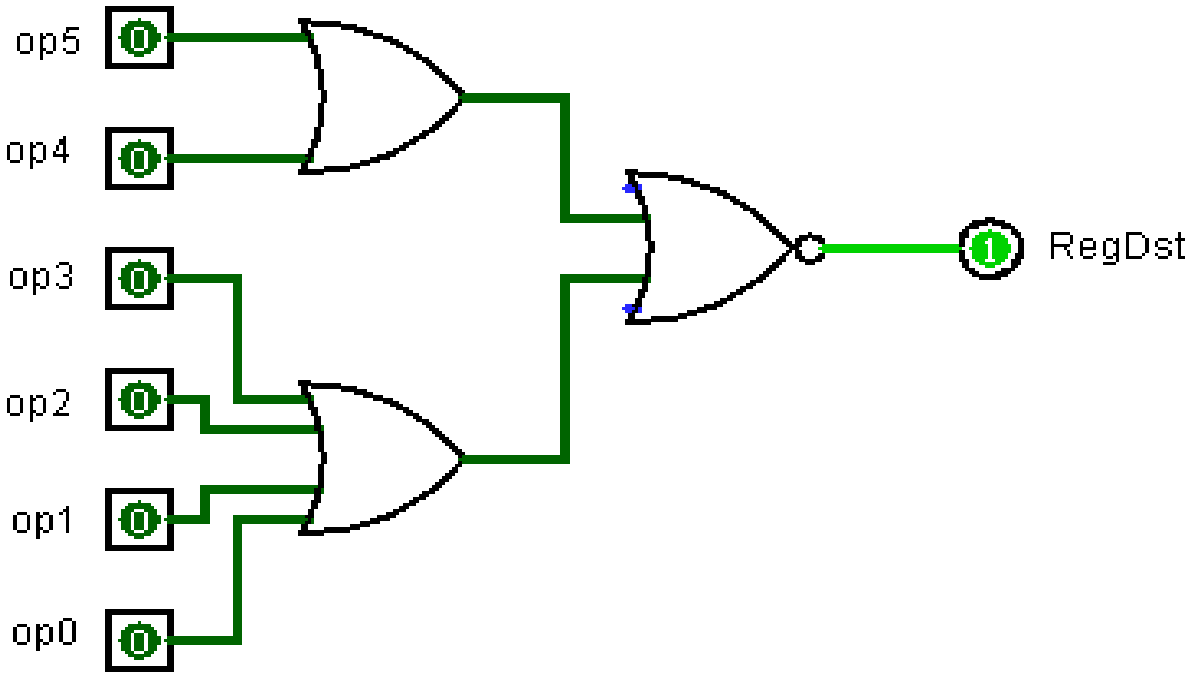
Pipelining

QUESTION 1

Q1(a) RegDst

	RegDst	ALUSrc	MemtoReg	MemWrite	ALUop	
					op1	op0
R-type (0_{16})	1	0	0	0	1	0
lw (23_{16})	0	1	1	0	0	0
sw ($2b_{16}$)	X	1	X	1	0	0
beq (4_{16})	X	0	X	0	0	1

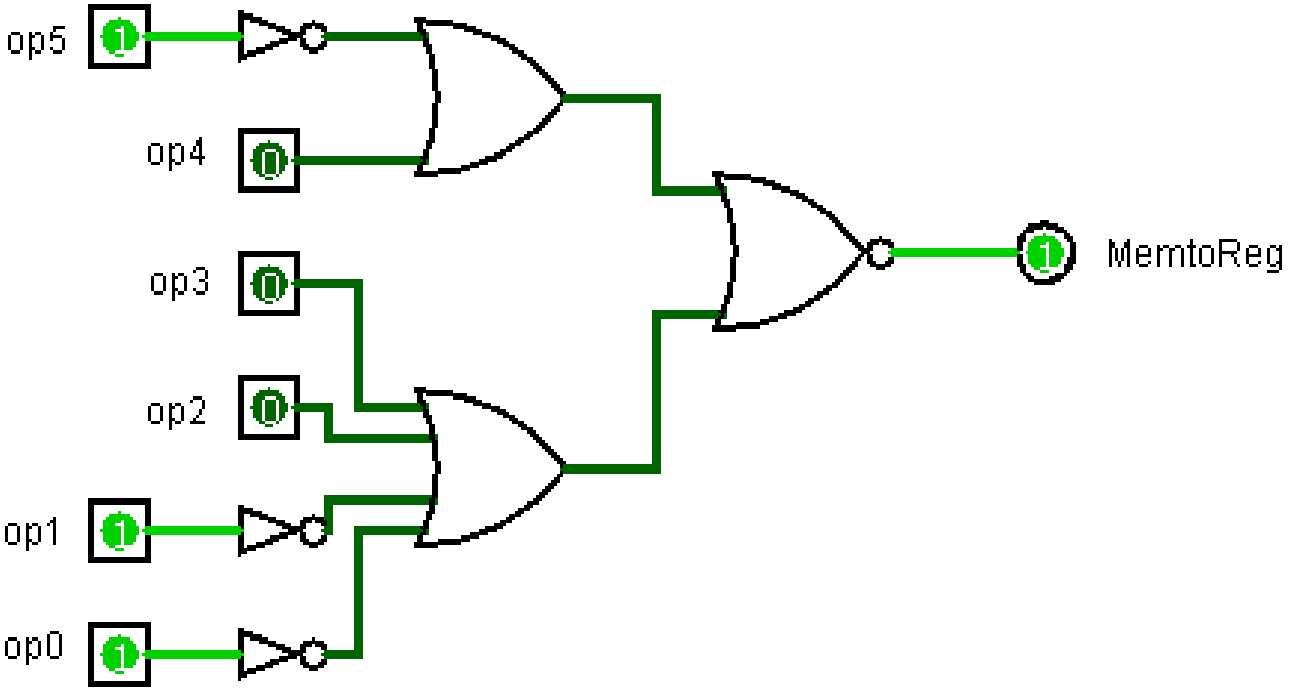
Opcode for R-type = 0x0 = 0b000000



Q1(b) MemtoReg

	RegDst	ALUSrc	MemtoReg	MemWrite	ALUop	
					op1	op0
R-type (0_{16})	1	0	0	0	1	0
lw (23_{16})	0	1	1	0	0	0
sw ($2b_{16}$)	X	1	X	1	0	0
beq (4_{16})	X	0	X	0	0	1

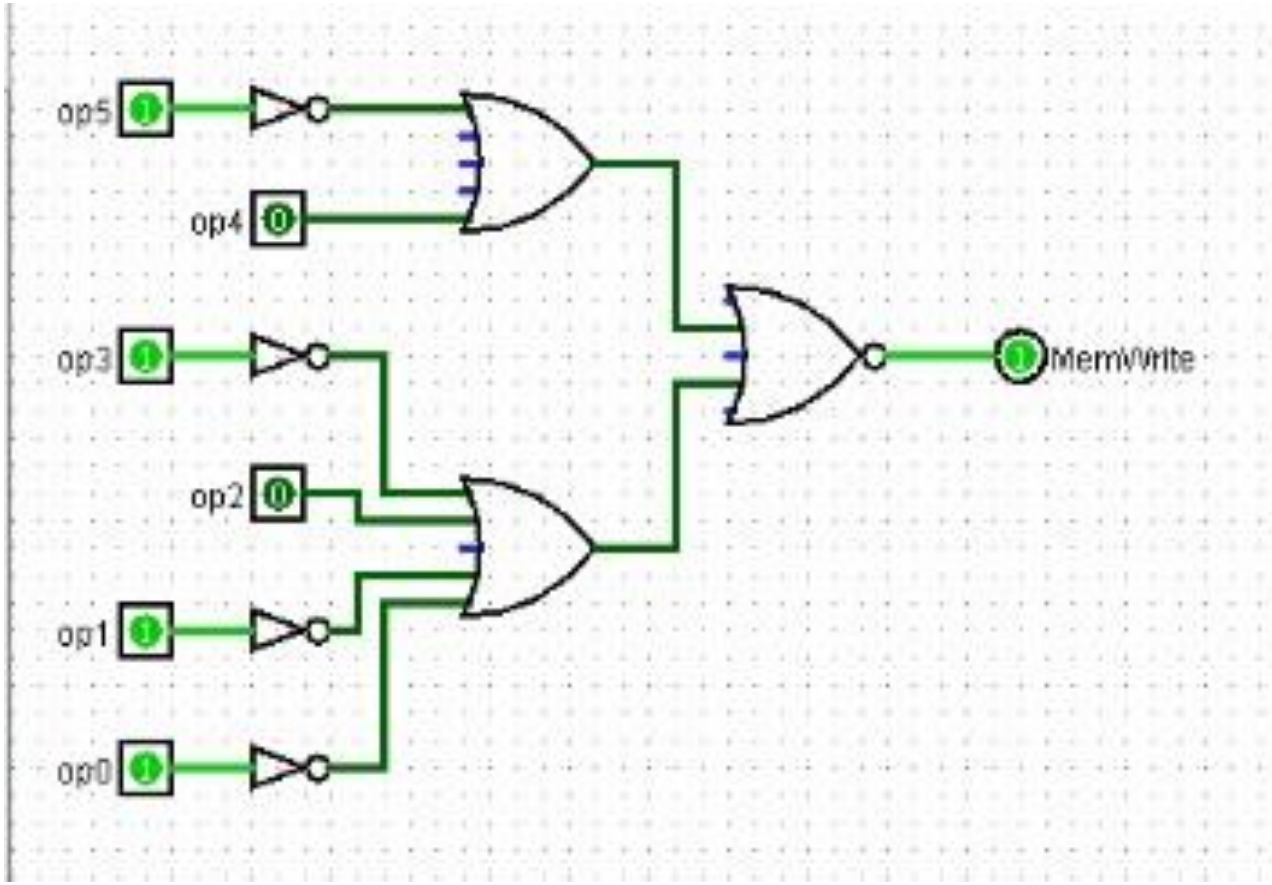
Opcode for lw = 0x23 = 0b100011



Q1(c) MemWrite

	RegDst	ALUSrc	MemtoReg	MemWrite	ALUop	
					op1	op0
R-type (0_{16})	1	0	0	0	1	0
lw (23_{16})	0	1	1	0	0	0
sw ($2b_{16}$)	X	1	X	1	0	0
beq (4_{16})	X	0	X	0	0	1

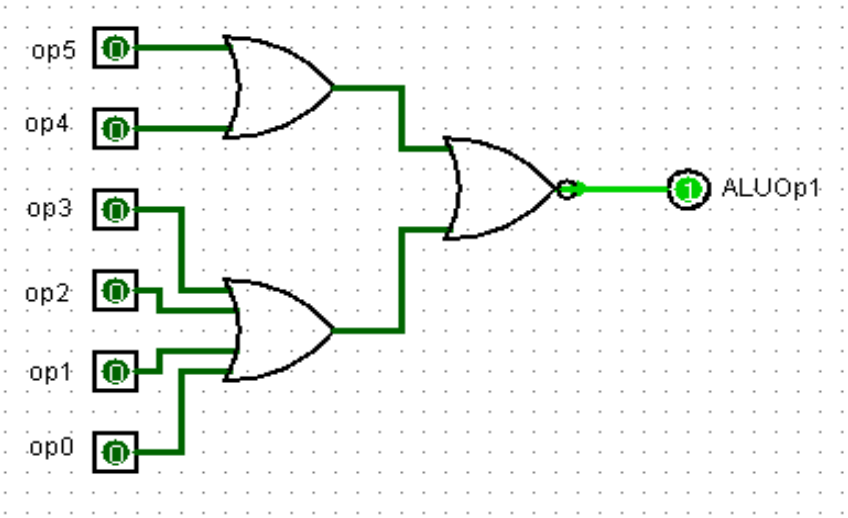
Opcode for sw = 0x2B = 0b101011

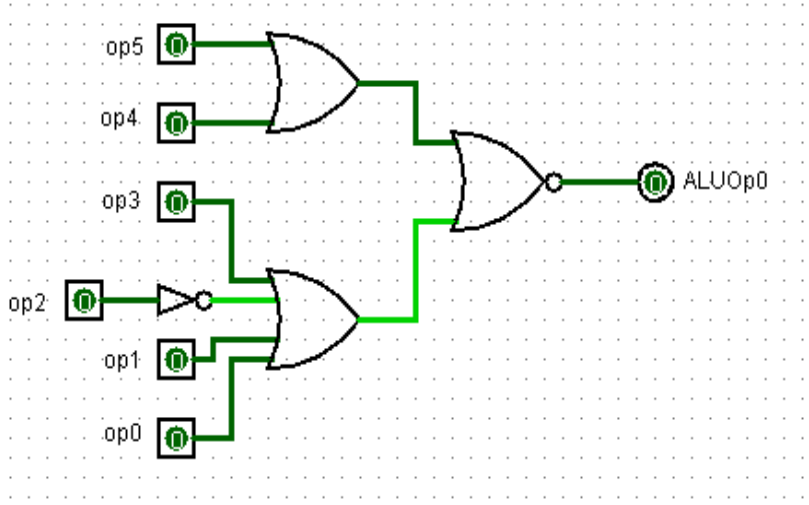


Q1(d) ALUOp

	RegDst	ALUSrc	MemtoReg	MemWrite	ALUop	
					op1	op0
R-type (0_{16})	1	0	0	0	1	0
lw (23_{16})	0	1	1	0	0	0
sw ($2b_{16}$)	X	1	X	1	0	0
beq (4_{16})	X	0	X	0	0	1

ALUOp1 = 0 for lw/sw (0x23/0x2B), 1 for beq (0x4) and 2 for R-type (0x0).





Q1(e) ALUControl

ALUControl takes in 8 input bits and produces 4 output bits. The following table summarizes the function.

Instruction	ALUOp	Funct field	ALU Operation	ALUControl
LW	00	XXXXXX	Add	0010
SW	00	XXXXXX	Add	0010
BEQ	01	XXXXXX	Subtract	0110
ADD	10	100000	Add	0010
SUB	10	100010	Subtract	0110
AND	10	100100	AND	0000
OR	10	100101	OR	0001
SLT	10	101010	Set on less than	0111
NOR	10	100111	NOR	1100

Let's start by naming the ALUOp bits as 'ALUOp1' and 'ALUOp0' as before, the funct bits as $F_5F_4F_3F_2F_1F_0$, and the ALUControl bits as $A_3A_2A_1A_0$.

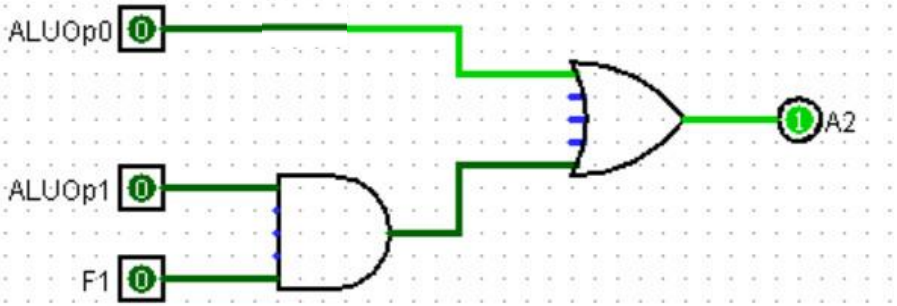
Q1(e) ALUControl

Instruction	ALUOp	Funct field	ALU Operation	ALUControl
LW	00	XXXXXX	Add	0010
SW	00	XXXXXX	Add	0010
BEQ	01	XXXXXX	Subtract	0110
ADD	10	100000	Add	0010
SUB	10	100010	Subtract	0110
AND	10	100100	AND	0000
OR	10	100101	OR	0001
SLT	10	101010	Set on less than	0111
NOR	10	100111	NOR	1100

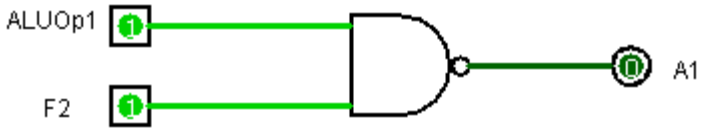
A0:

$$A_0 = ALUOp_1 \cdot ((F_0 \oplus F_1) \cdot (F_2 \oplus F_3))$$

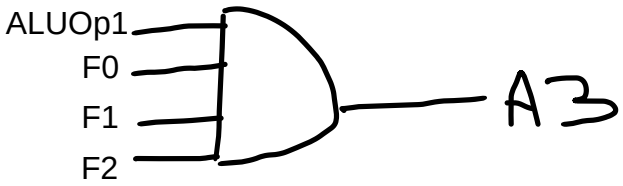
A2:



A1:

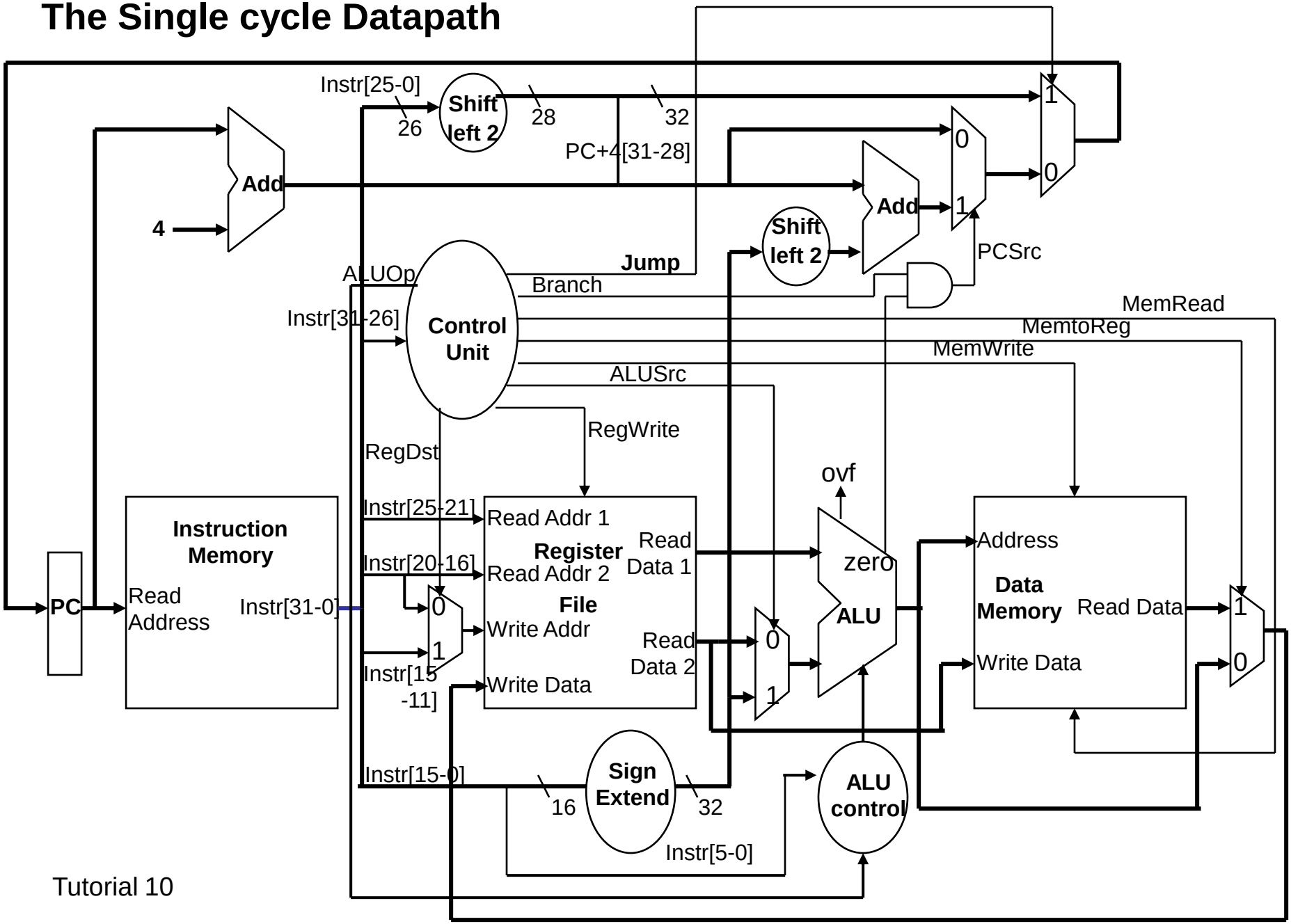


A3:

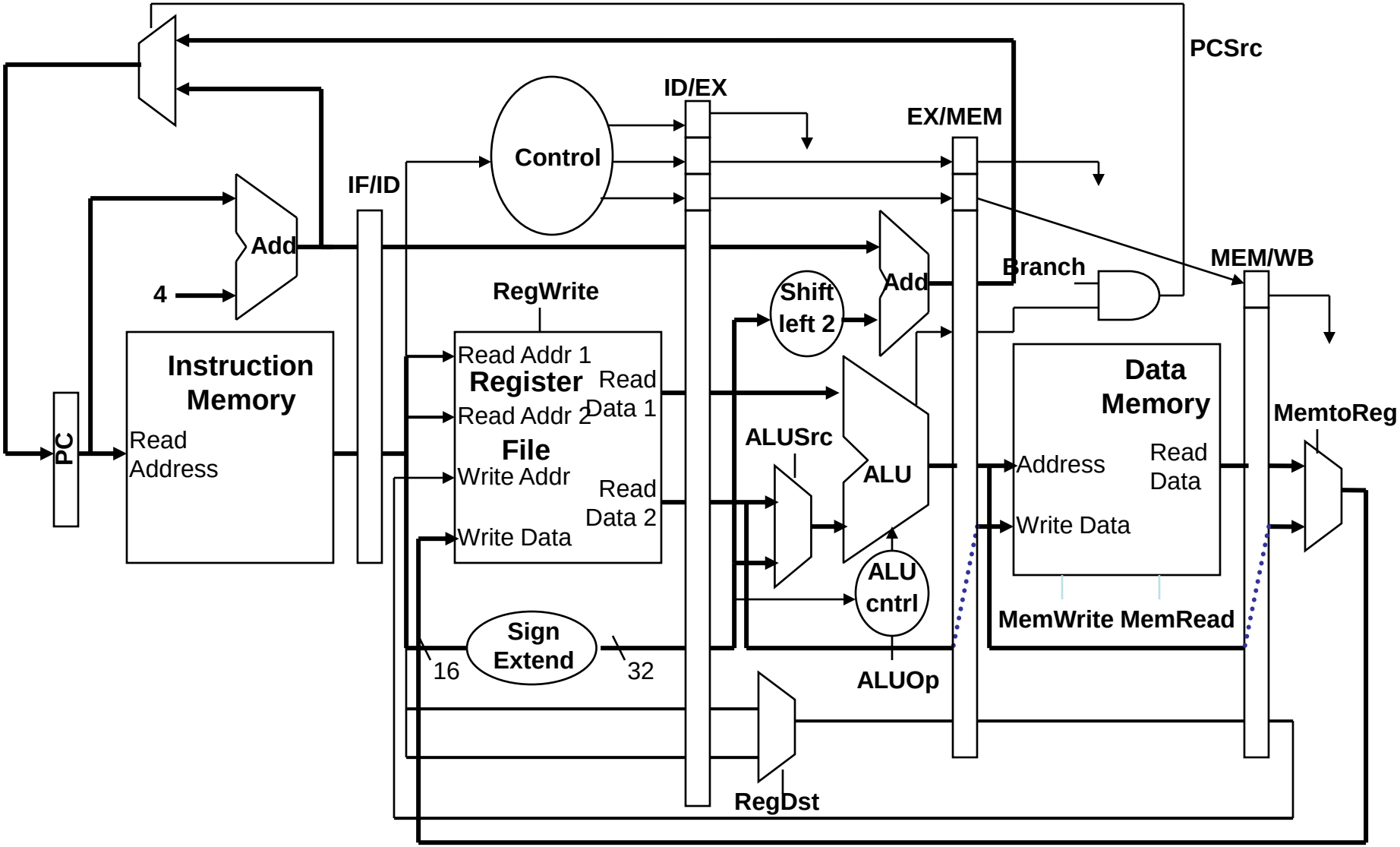


DATAPATH

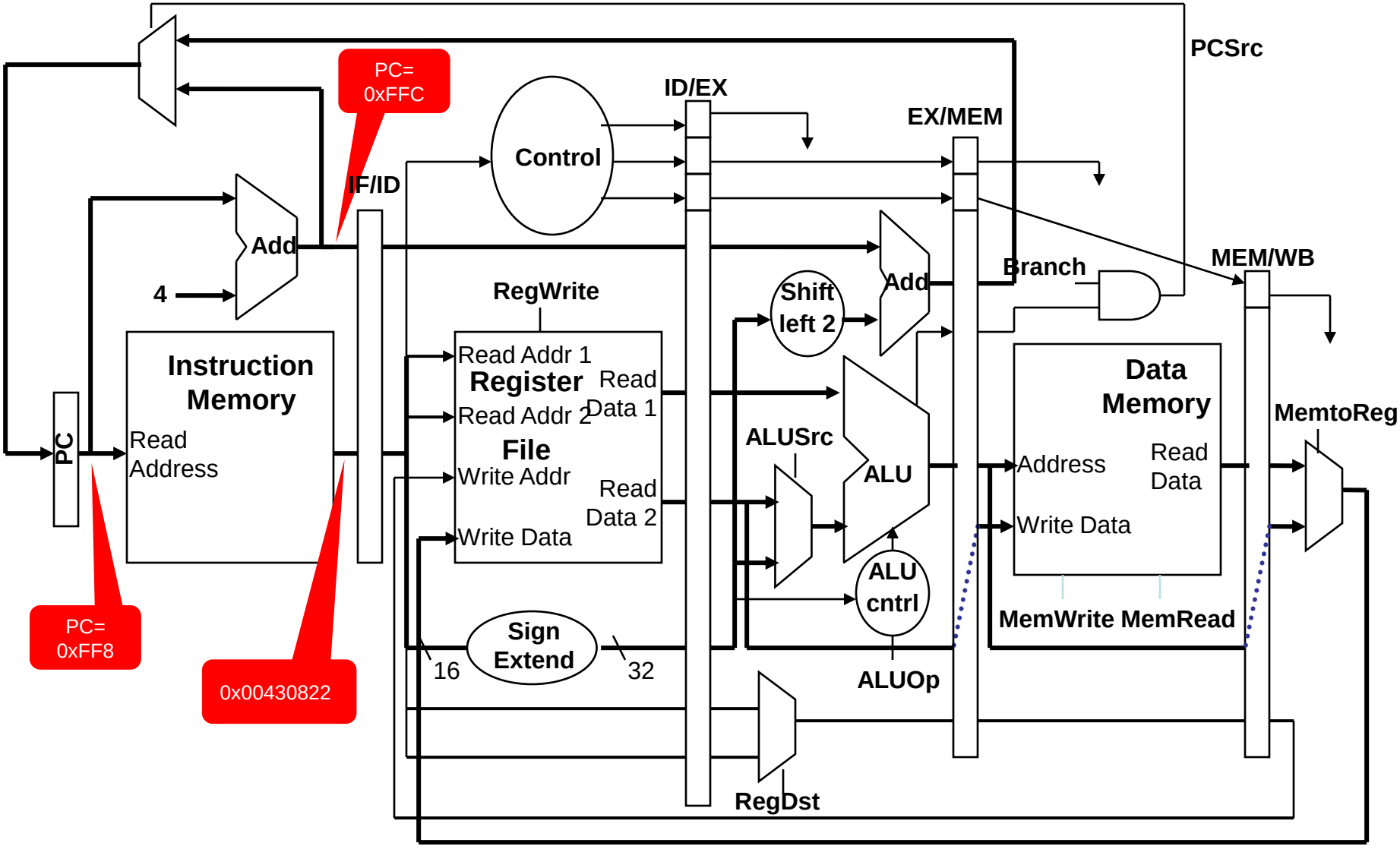
The Single cycle Datapath



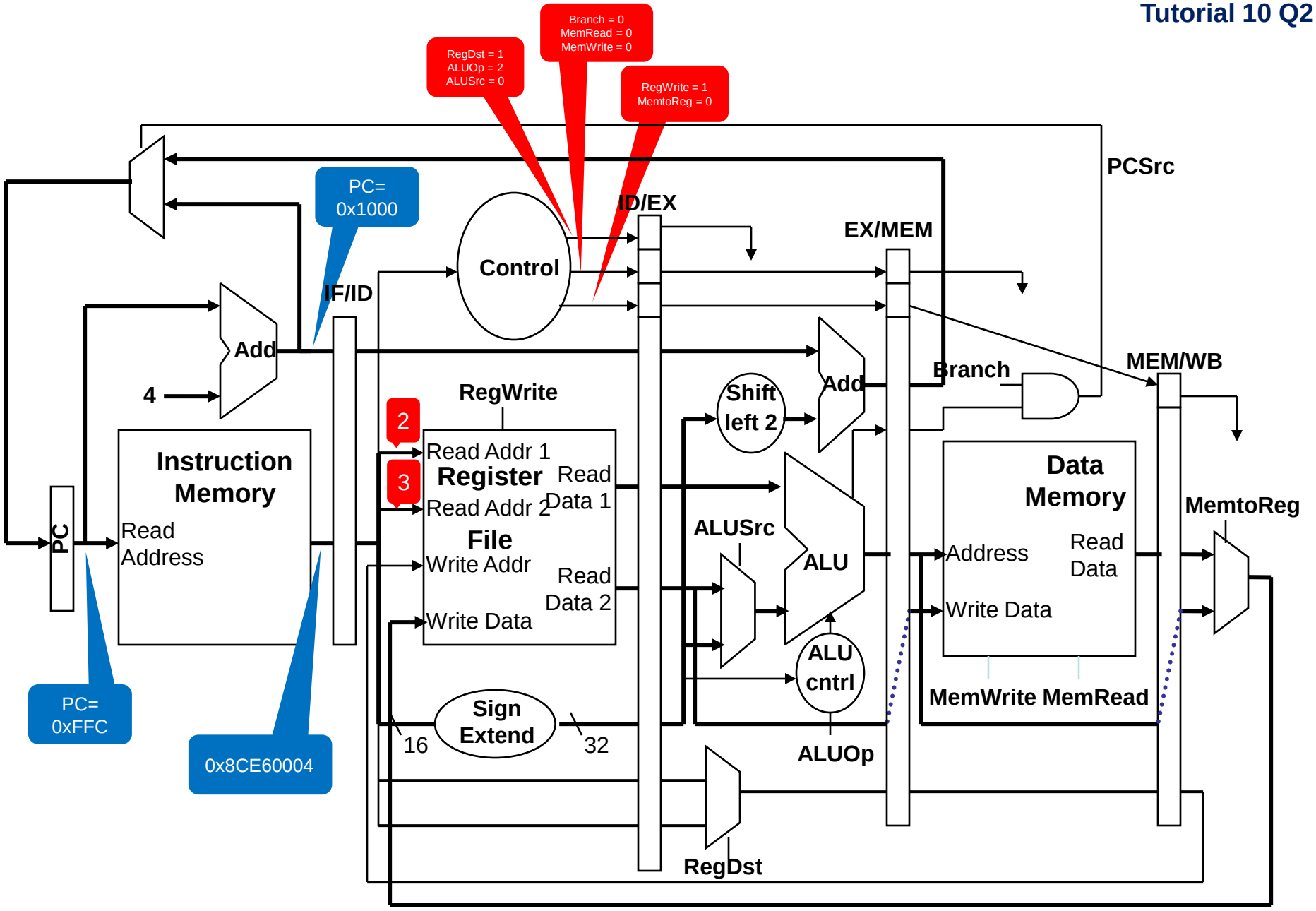
Pipelined Datapath



QUESTION 2

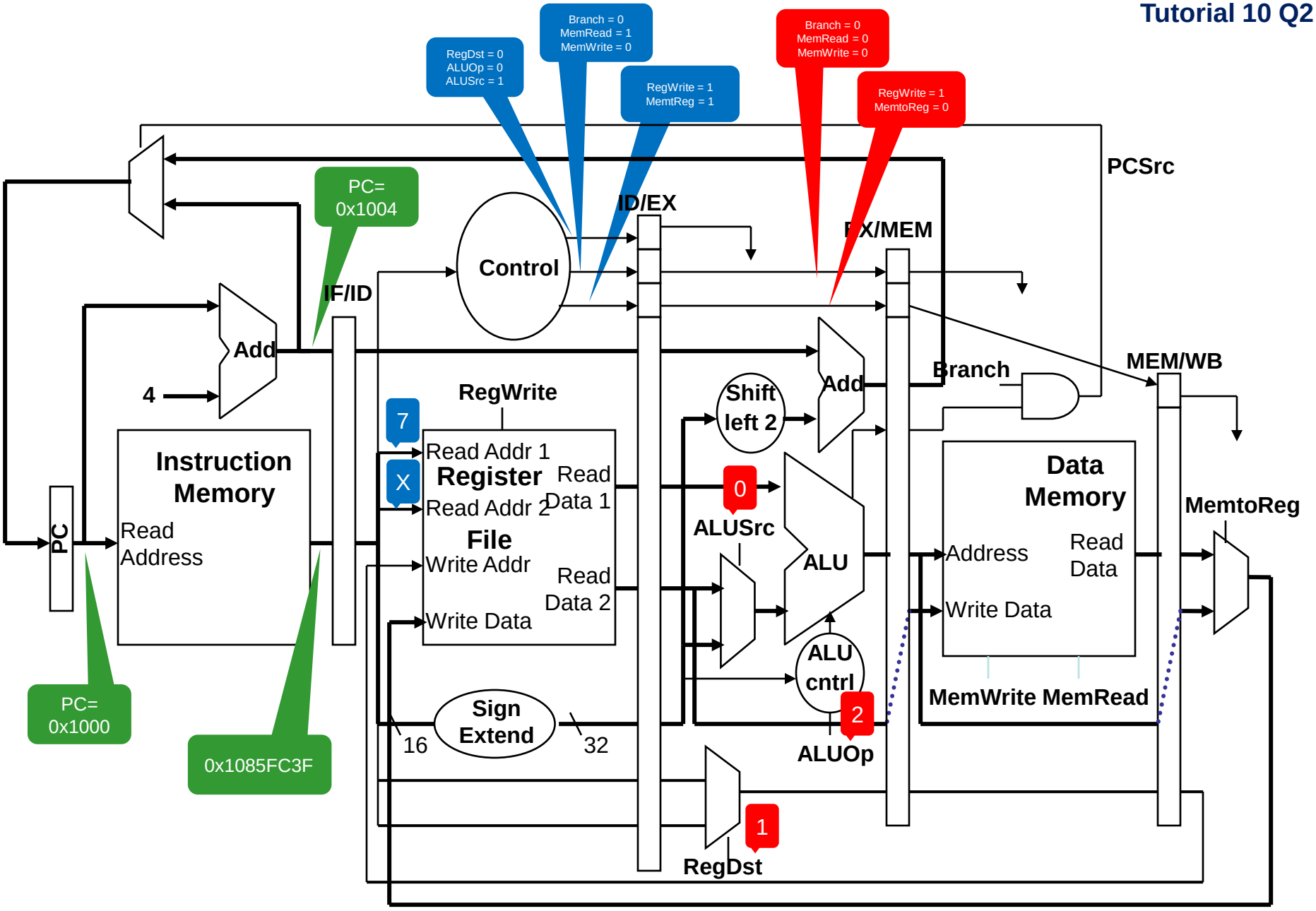

PCSrc
PC= 0xFFC
ID/EX
EX/MEM
Control
IF/ID
Add
4
MEM/WB
Branch
RegWrite
Shift left 2
Add
Instruction Memory
Read Address
PC
Read Addr 1
Register
Read Addr 2
File
Write Addr
Write Data
Read Data 1
Read Data 2
ALUSrc
ALU
Data Memory
Address
Read Data
Write Data
MemtoReg
ALU cntrl
MemWrite MemRead
PC= 0xFF8
0x00430822
Sign Extend
16
32
ALUOp
RegDst

```
sub $1, $2, $3
```



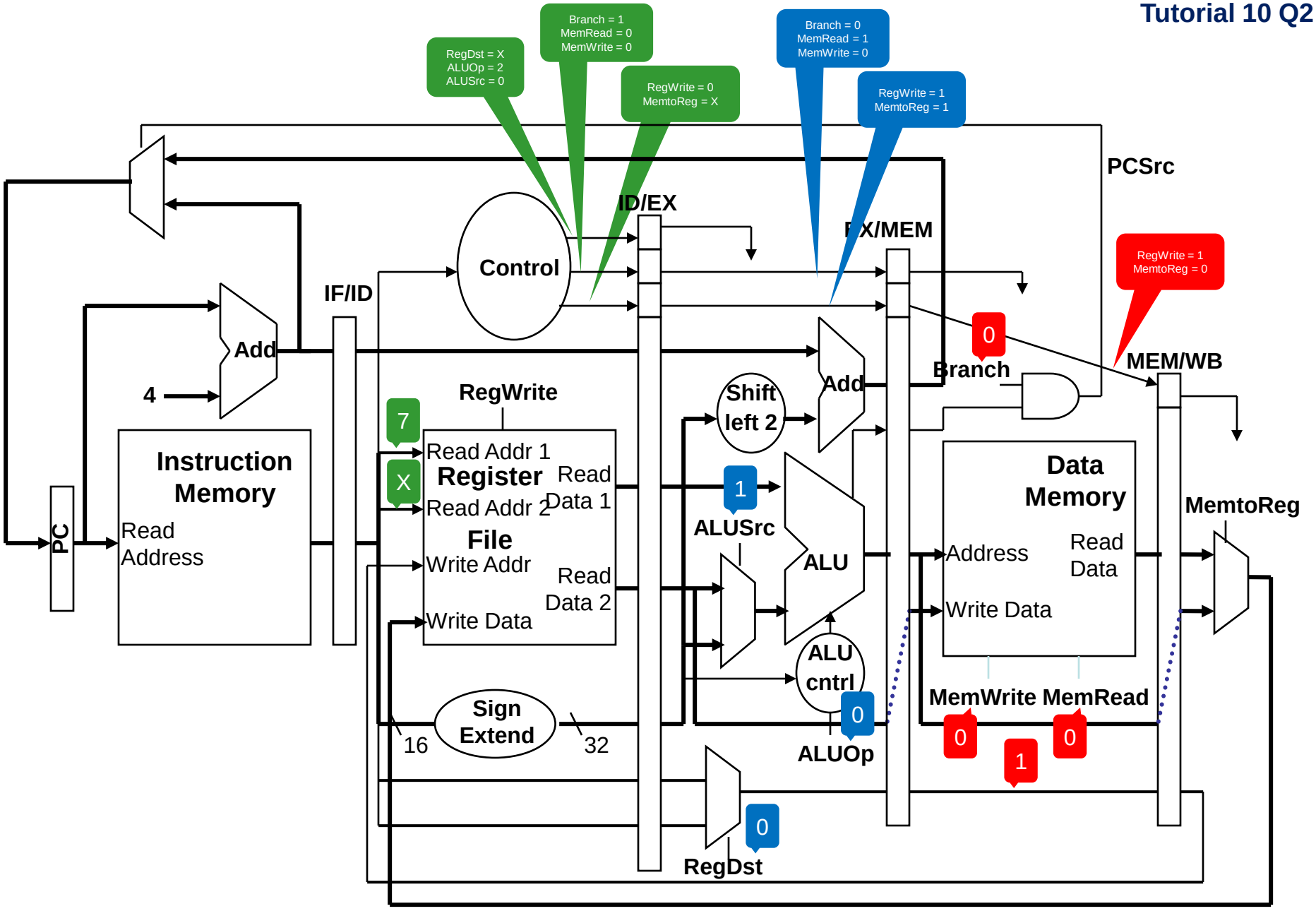

```
lw $6, 4($7)        sub $1, $2, $3
```

Tutorial 10 Q2

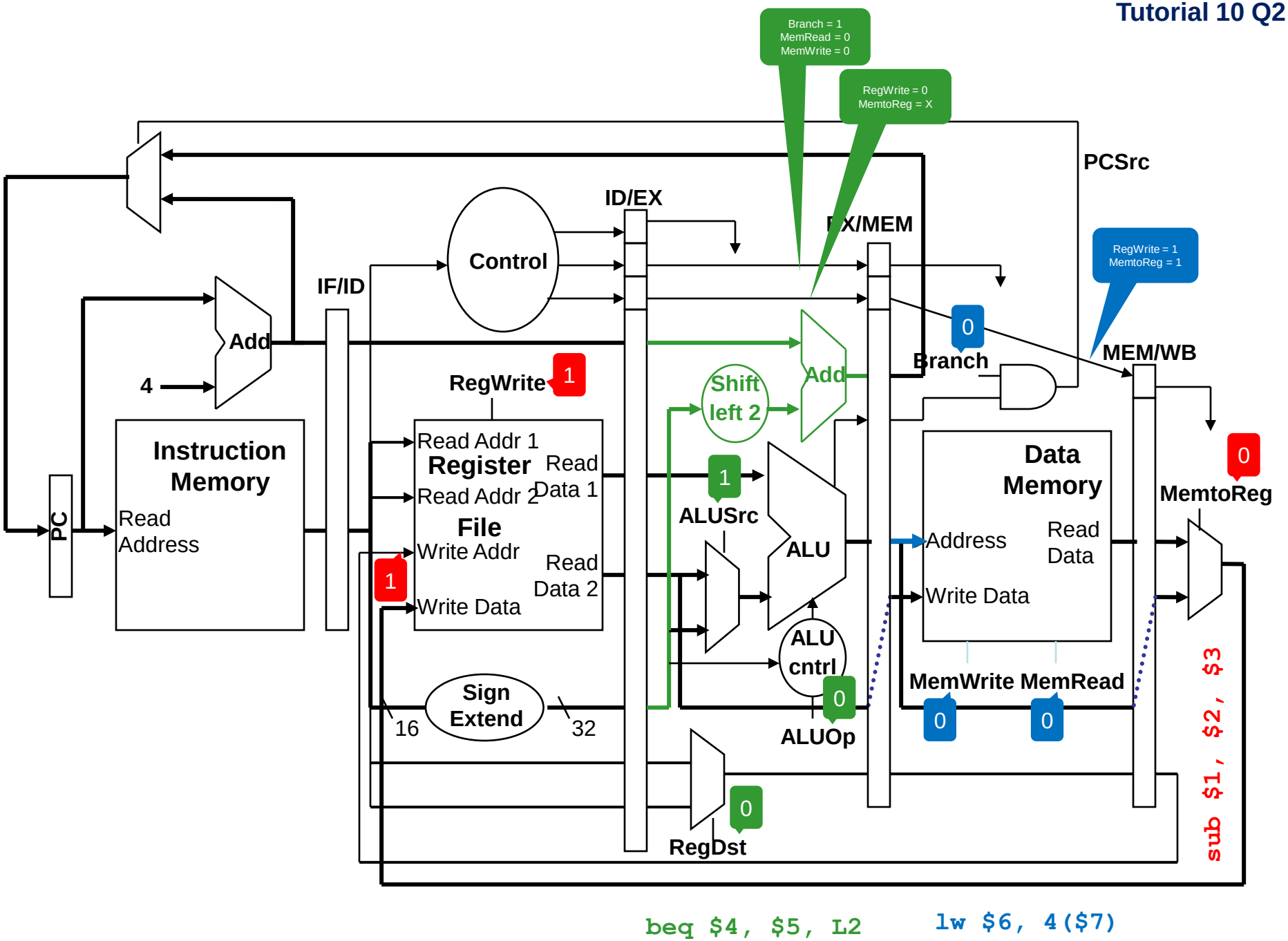


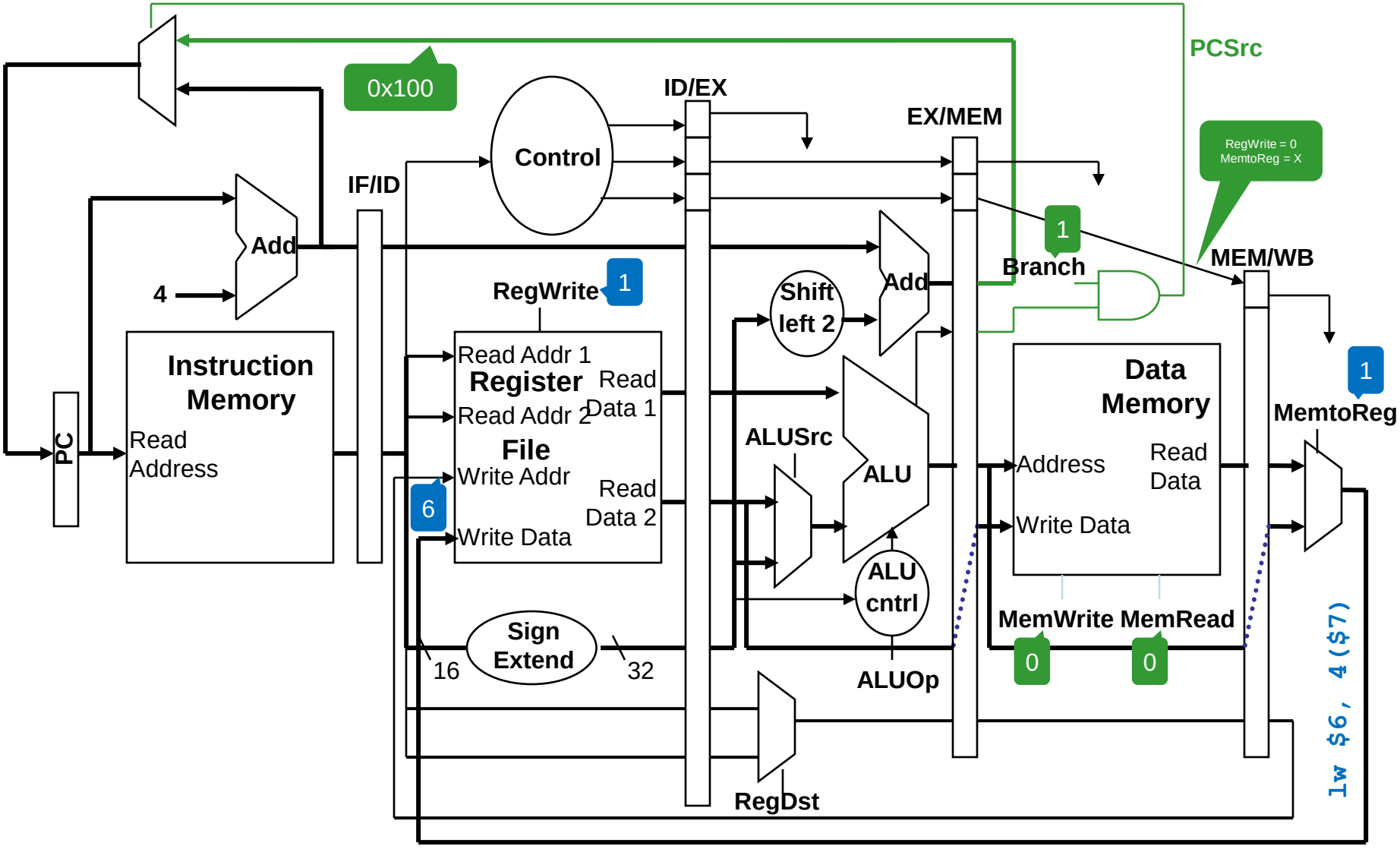

```
beq $4, $5, L2        lw $6, 4($7)     sub $1, $2, $3
```

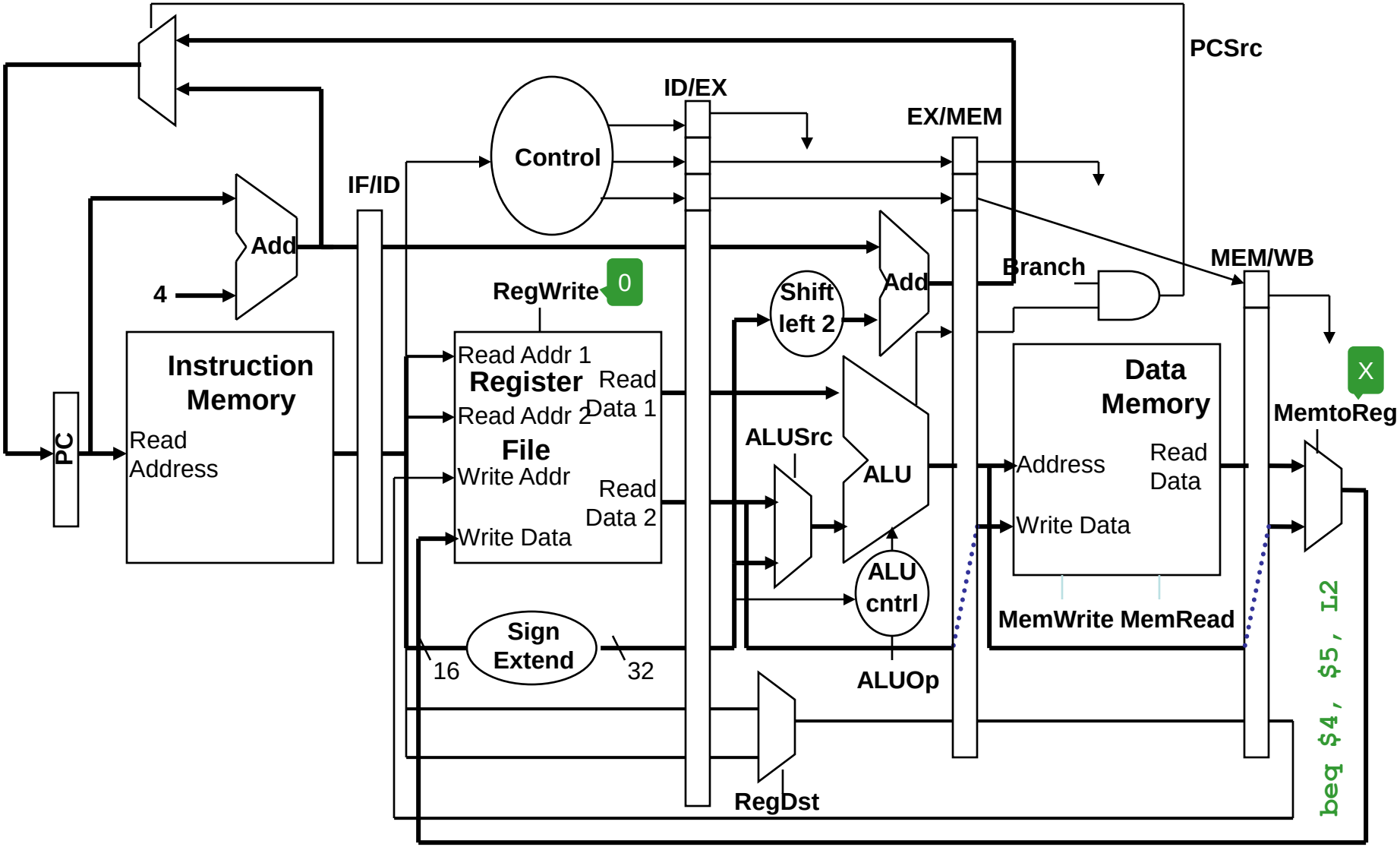
Tutorial 10 Q2



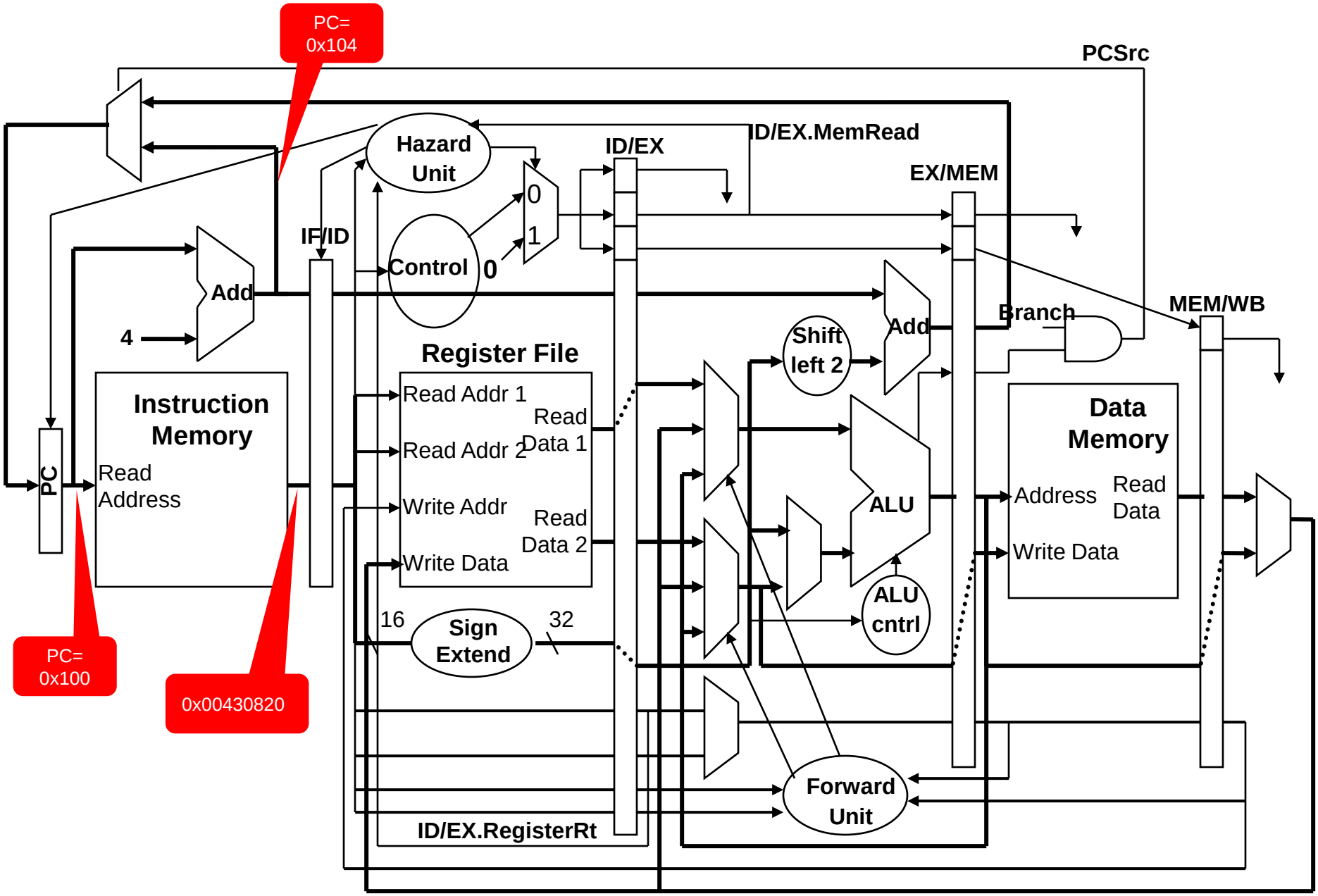
Tutorial 10 Q2



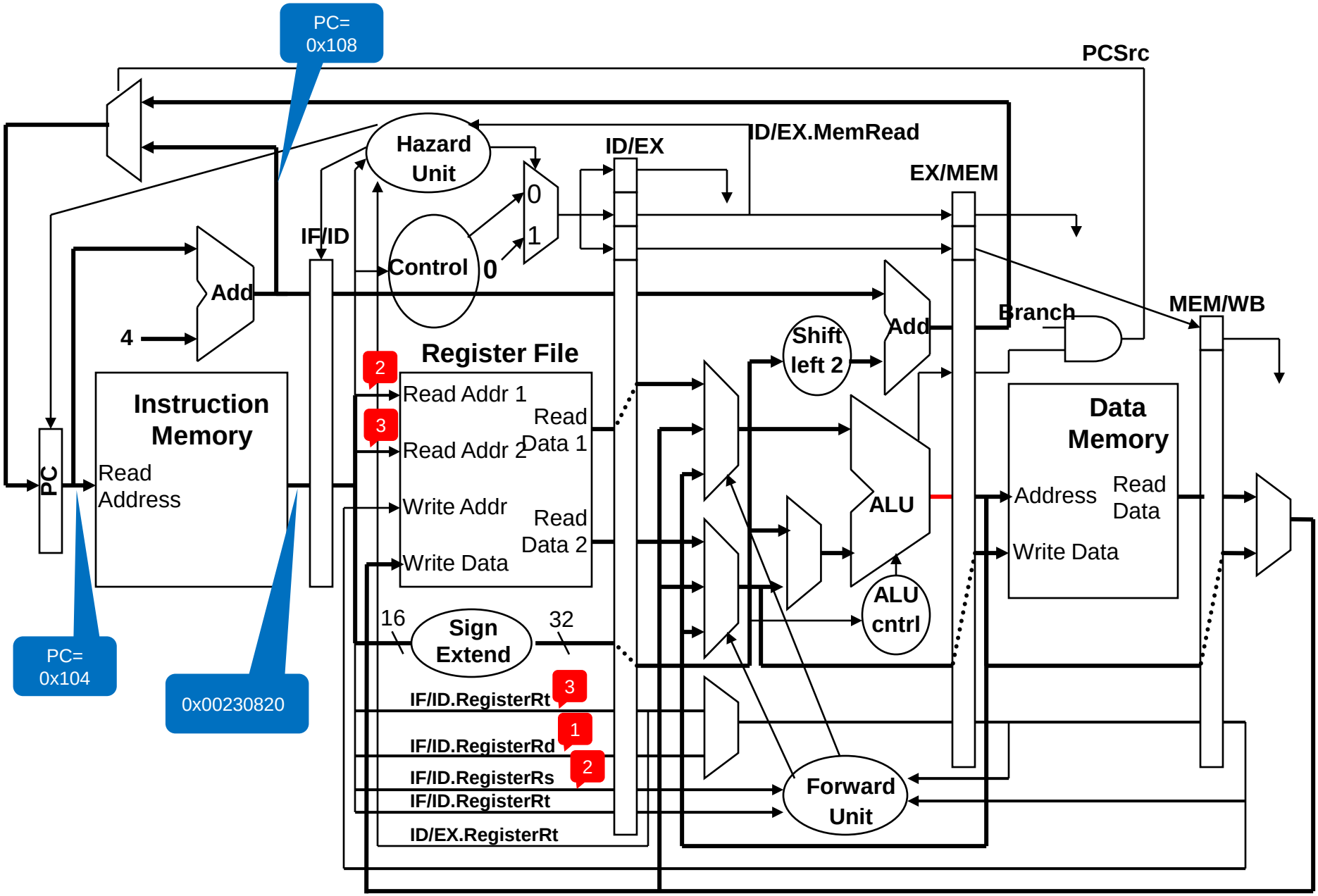

PCSrc
0x100
ID/EX
EX/MEM
Control
IF/ID
RegWrite = 0
MemtoReg = X
Add
4
1
Branch
MEM/WB
RegWrite
1
Shift
left 2
Add
Instruction
Memory
Read
Address
PC
Read Addr 1
Register
Read Addr 2
File
Write Addr
Write Data
Read
Data 1
Read
Data 2
6
ALUSrc
ALU
Data
Memory
Address
Read
Data
Write Data
1
MemtoReg
ALU
cntrl
Sign
Extend
16
32
ALUOp
MemWrite
MemRead
0
0
RegDst
lw $6, 4($7)
beq $4, $5, L2

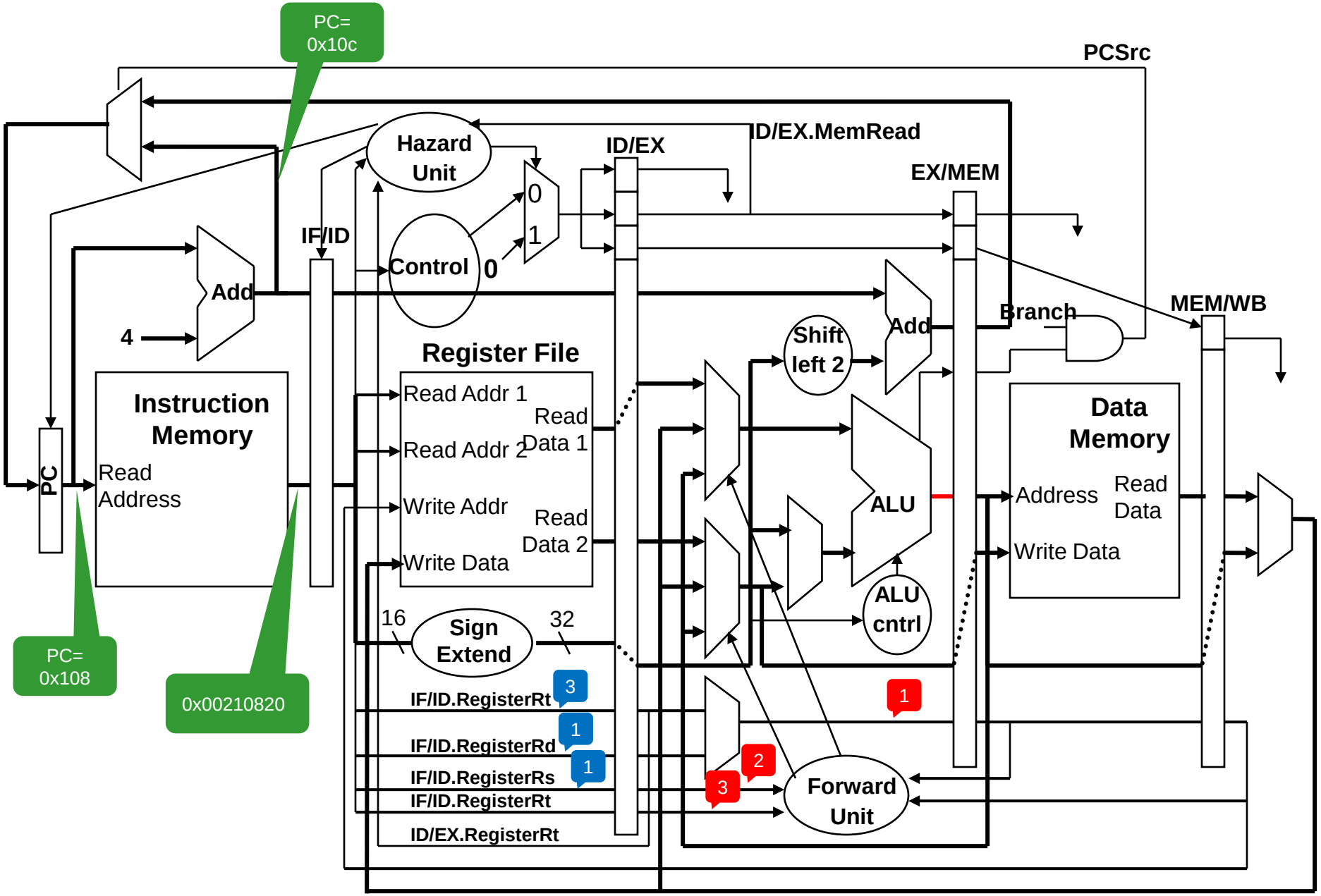

PCSrc
ID/EX
EX/MEM
MEM/WB
Control
IF/ID
Add
4
RegWrite
0
Shift
left 2
Add
Branch
Instruction
Memory
Read
Address
PC
Read Addr 1
Register
Read Addr 2
File
Write Addr
Write Data
Read
Data 1
Read
Data 2
ALUSrc
ALU
Data
Memory
Address
Read
Data
Write Data
MemtoReg
X
ALU
cntrl
MemWrite MemRead
Sign
Extend
16
32
ALUOp
RegDst
beq $4, $5, L2

QUESTION 3


PC= 0x104
PCSrc
Hazard Unit
ID/EX
ID/EX.MemRead
EX/MEM
IF/ID
Control
0
1
Add
4
Add
Branch
MEM/WB
Shift left 2
Register File
Instruction Memory
Read Address
PC
Read Addr 1
Read Addr 2
Write Addr
Write Data
Read Data 1
Read Data 2
ALU
ALU cntrl
Data Memory
Address
Read Data
Write Data
16
Sign Extend
32
PC= 0x100
0x00430820
Forward Unit
ID/EX.RegisterRt

```
add $1, $2, $3
```

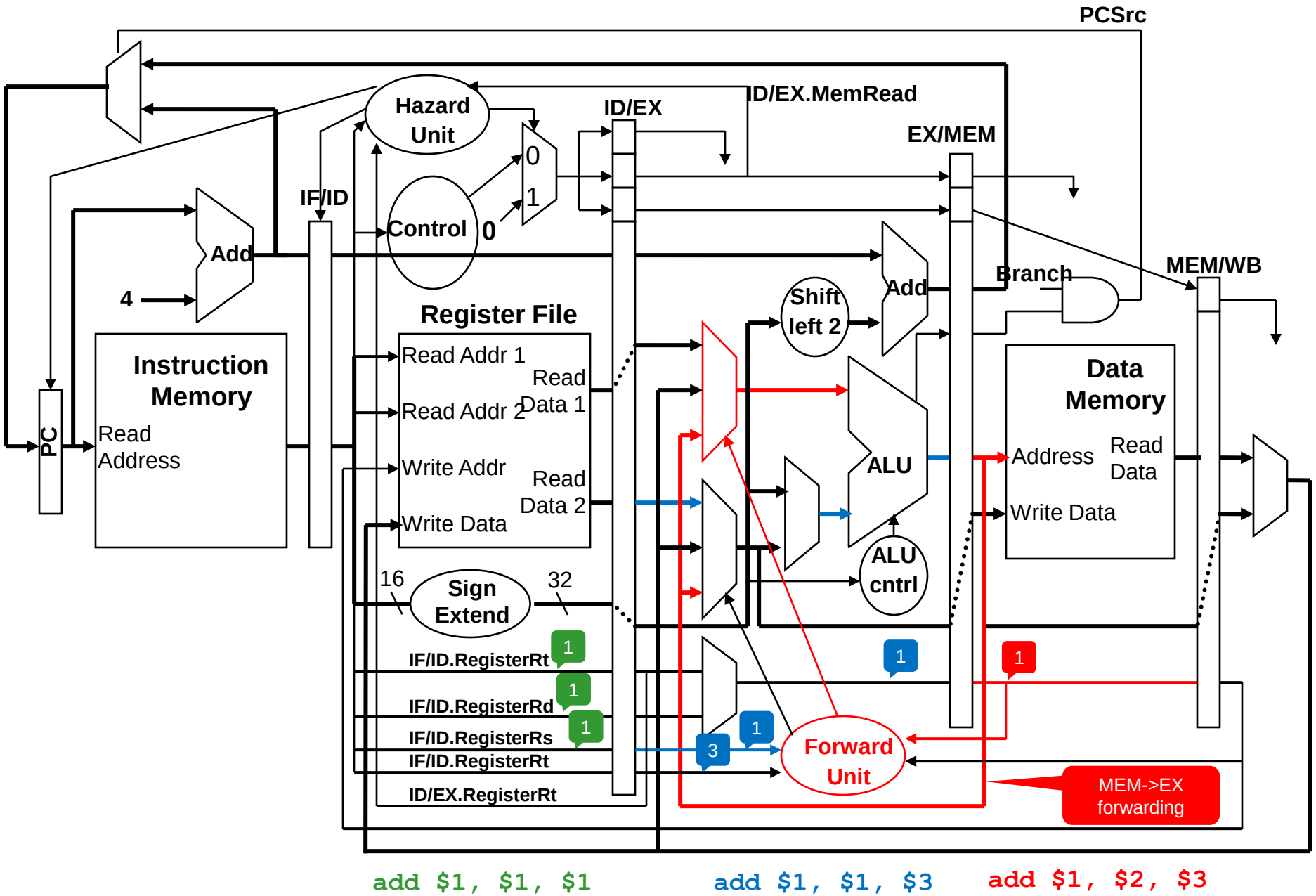

PC= 0x108
PCSrc
Hazard Unit
ID/EX.MemRead
ID/EX
EX/MEM
MEM/WB
IF/ID
Control
0
1
Add
4
Branch
Shift left 2
Register File
Instruction Memory
Read Address
PC
Read Addr 1
Read Addr 2
Write Addr
Write Data
Read Data 1
Read Data 2
ALU
ALU cntrl
Data Memory
Address
Read Data
Write Data
16
Sign Extend
32
IF/ID.RegisterRt
IF/ID.RegisterRd
IF/ID.RegisterRs
IF/ID.RegisterRt
ID/EX.RegisterRt
Forward Unit
PC= 0x104
0x00230820
add $1, $1, $3
add $1, $2, $3

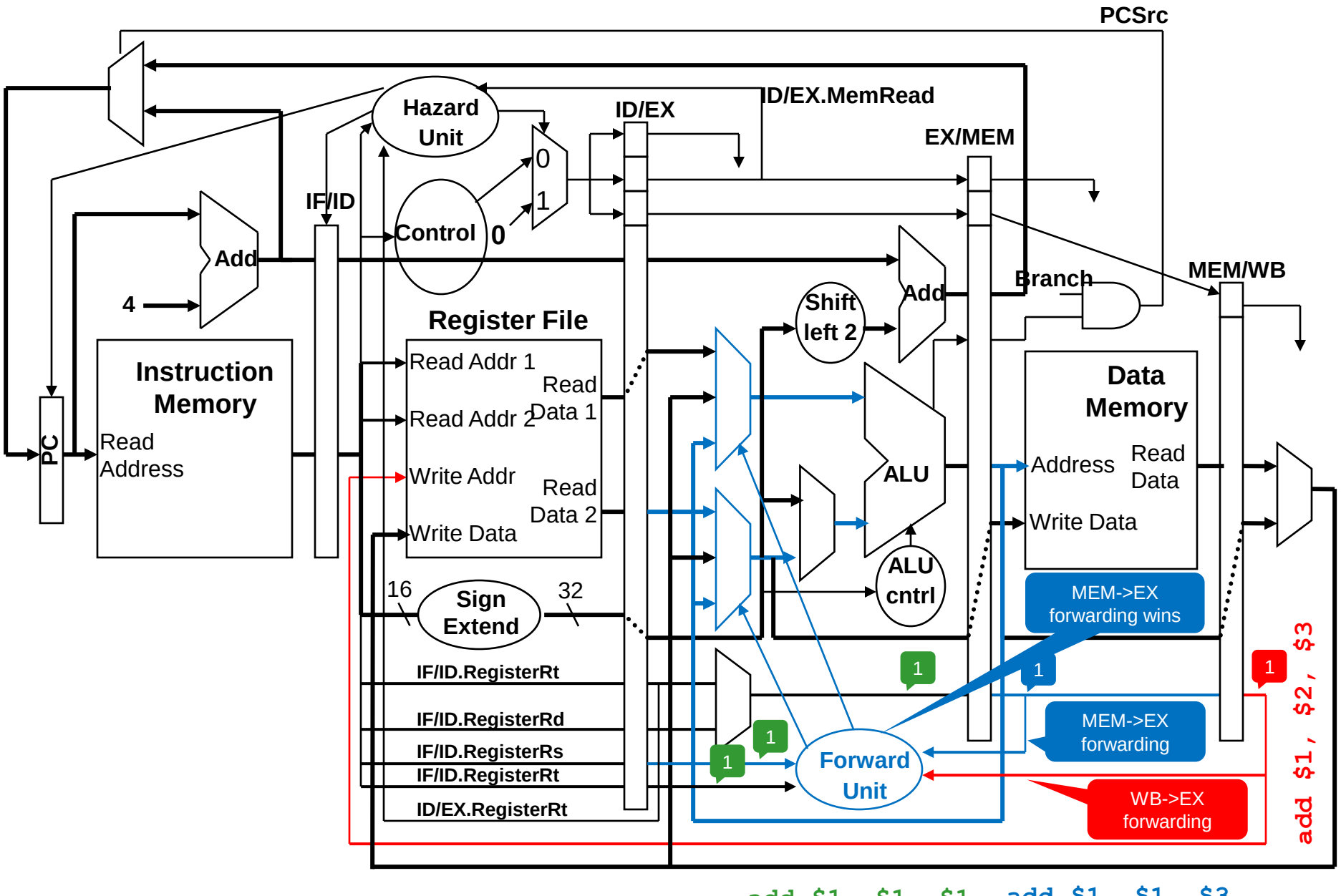


add $1, $1, $1

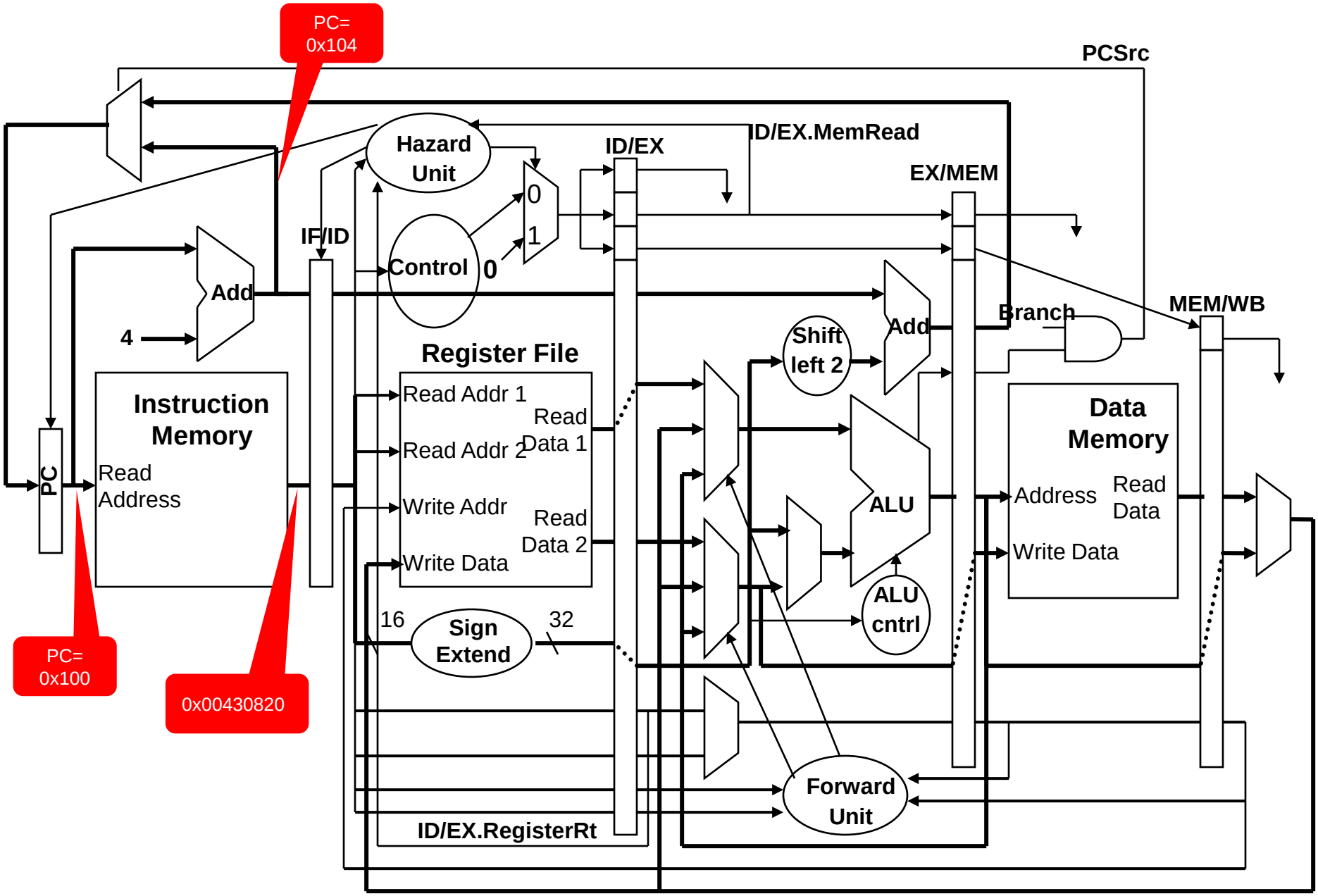
add $1, $1, $3

add $1, $2, $3

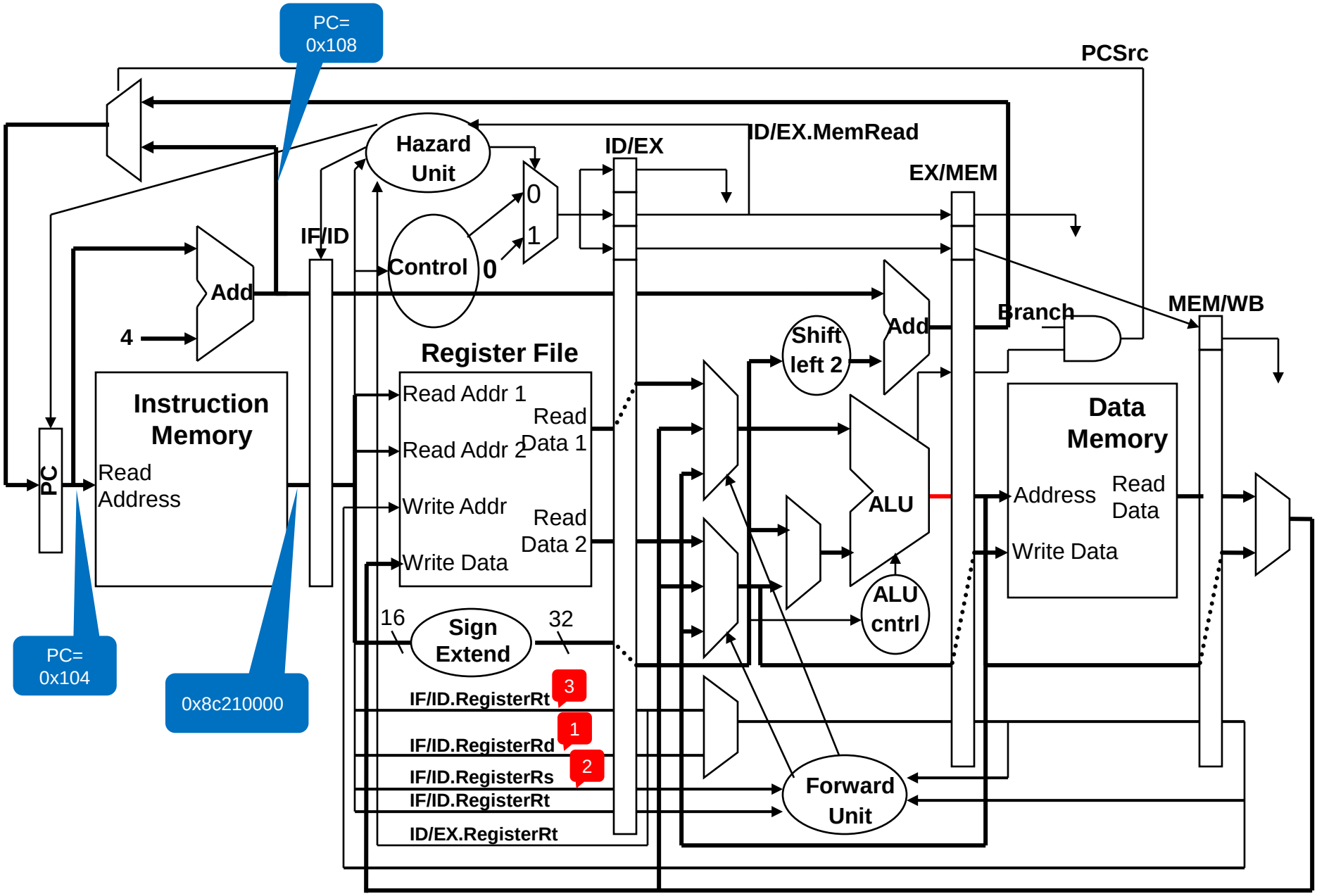

PCSrc
Hazard Unit
ID/EX
ID/EX.MemRead
EX/MEM
MEM/WB
IF/ID
Control
0
1
Add
4
Branch
Shift left 2
Register File
Instruction Memory
Read Address
PC
Read Addr 1
Read Addr 2
Write Addr
Write Data
Read Data 1
Read Data 2
ALU
ALU cntrl
Data Memory
Address
Read Data
Write Data
16
Sign Extend
32
IF/ID.RegisterRt
IF/ID.RegisterRd
IF/ID.RegisterRs
IF/ID.RegisterRt
ID/EX.RegisterRt
Forward Unit
MEM->EX forwarding
add $1, $1, $1
add $1, $1, $3
add $1, $2, $3


PCSrc
Hazard Unit
ID/EX.MemRead
ID/EX
EX/MEM
MEM/WB
IF/ID
Control
0
1
Add
4
Branch
Shift left 2
Register File
Instruction Memory
Read Address
PC
Read Addr 1
Read Addr 2
Write Addr
Write Data
Read Data 1
Read Data 2
ALU
ALU cntrl
Data Memory
Address
Read Data
Write Data
16
Sign Extend
32
IF/ID.RegisterRt
IF/ID.RegisterRd
IF/ID.RegisterRs
IF/ID.RegisterRt
ID/EX.RegisterRt
Forward Unit
MEM->EX forwarding wins
MEM->EX forwarding
WB->EX forwarding
add $1, $2, $3
add $1, $1, $1
add $1, $1, $3

QUESTION 4


PC= 0x104
PCSrc
Hazard Unit
ID/EX
ID/EX.MemRead
EX/MEM
IF/ID
Control
0
1
Add
4
Add
Branch
MEM/WB
Shift left 2
Register File
Instruction Memory
Read Address
PC
Read Addr 1
Read Addr 2
Write Addr
Write Data
Read Data 1
Read Data 2
ALU
ALU cntrl
Data Memory
Address
Read Data
Write Data
16
Sign Extend
32
PC= 0x100
0x00430820
Forward Unit
ID/EX.RegisterRt

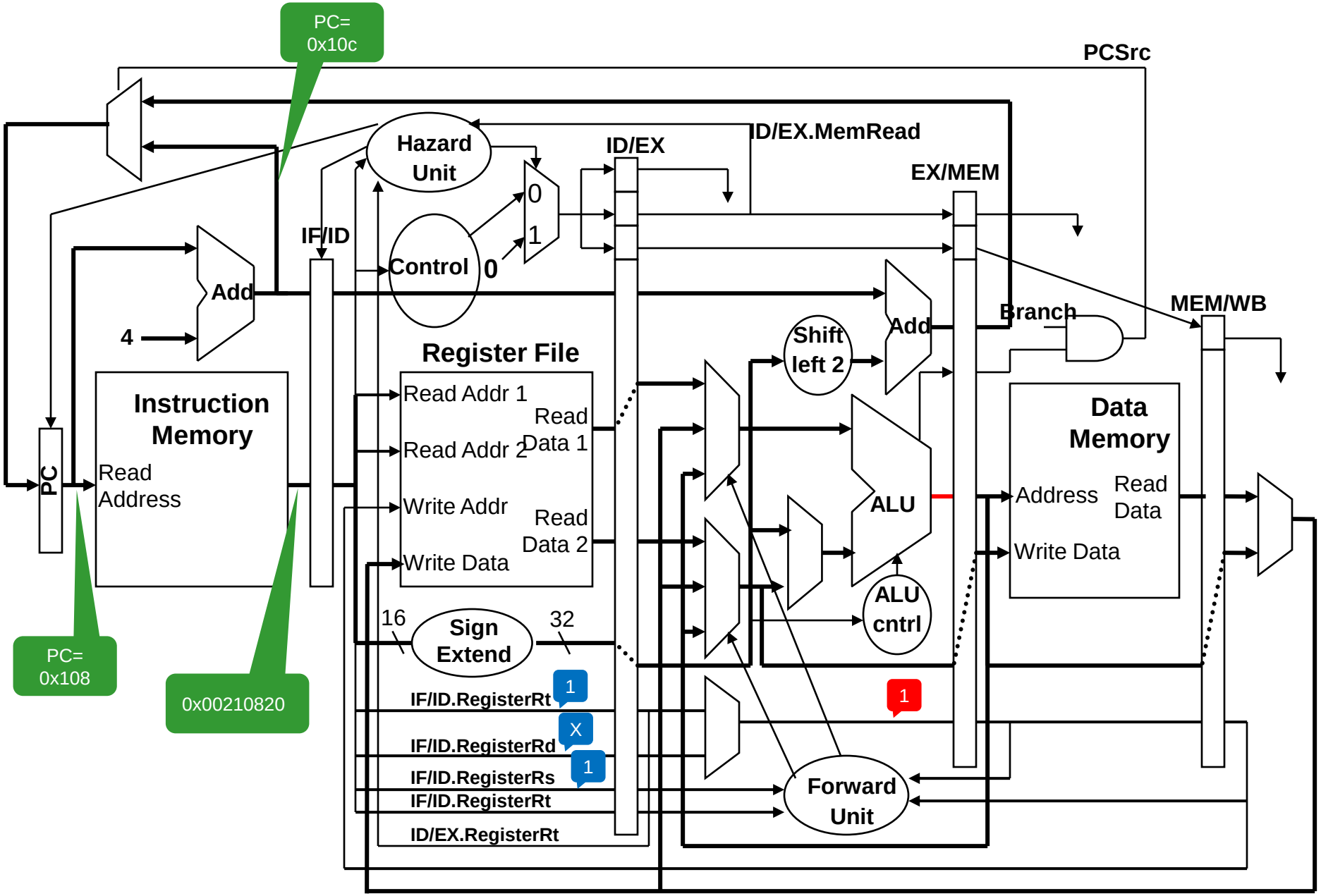
```
add $1, $2, $3
```



`lw $1, 0($1)`

`add $1, $2, $3`

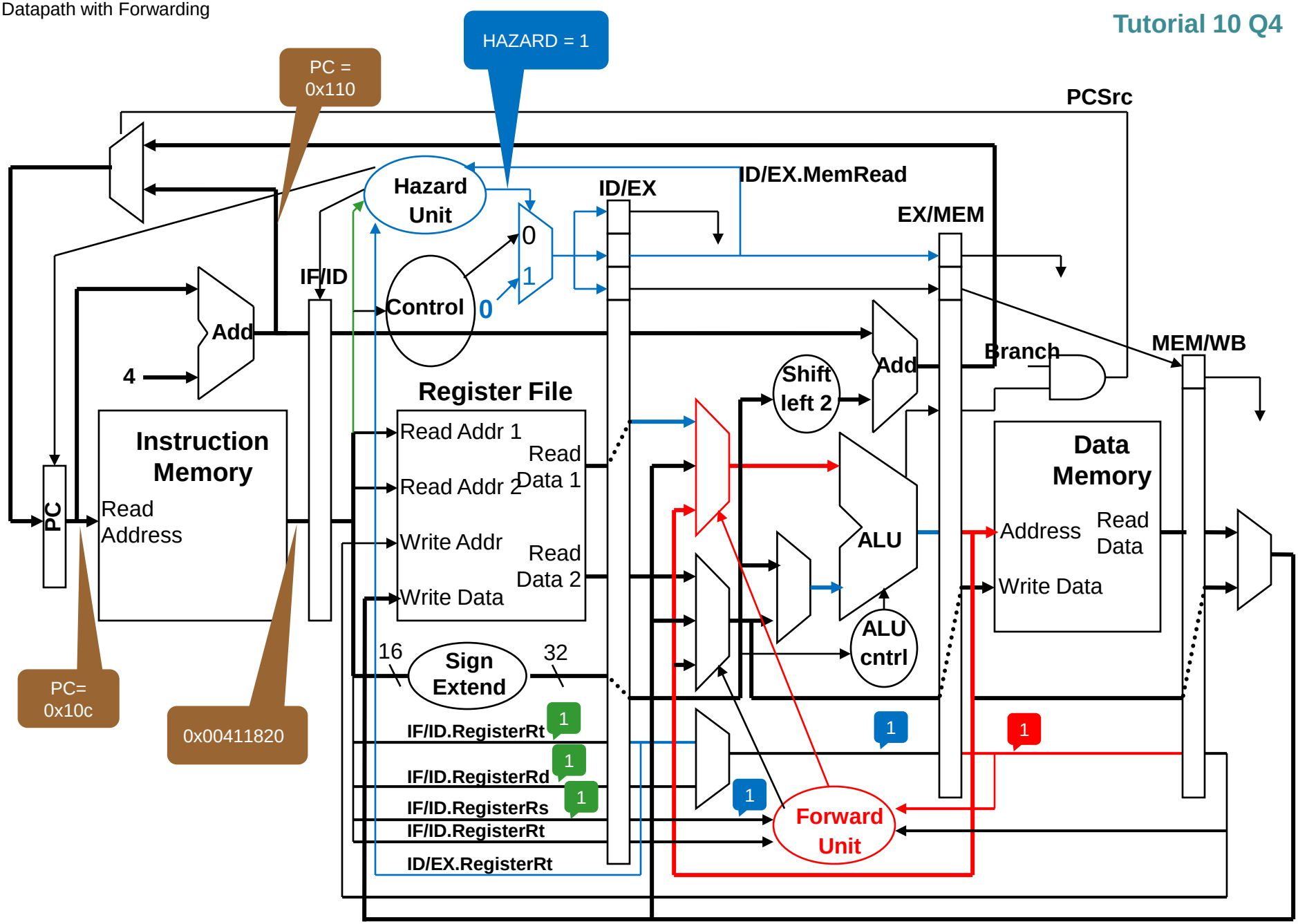
Tutorial 10 Q4




```
add $1, $1, $1        lw $1, 0($1)        add $1, $2, $3
```

Datapath with Forwarding

Tutorial 10 Q4

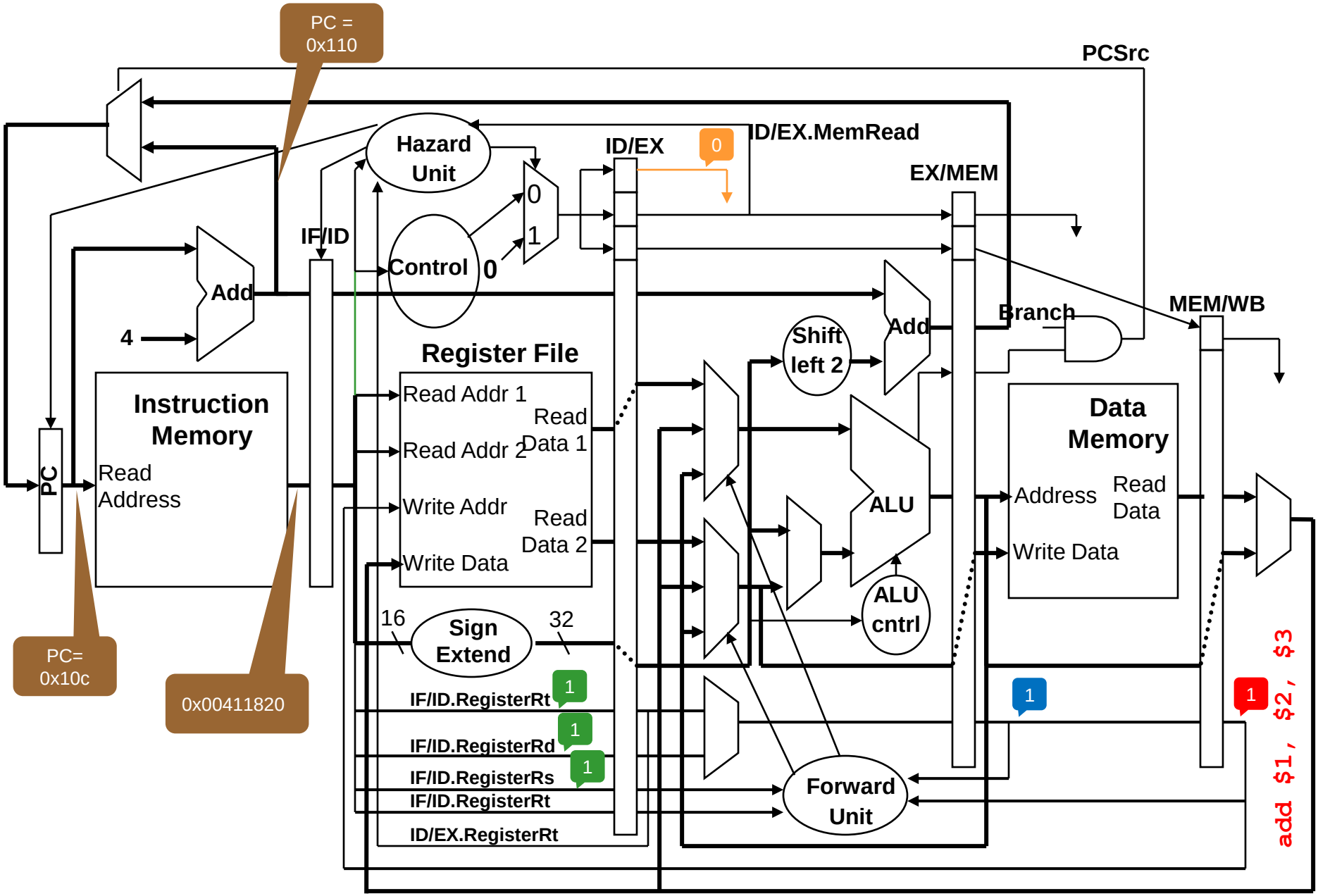


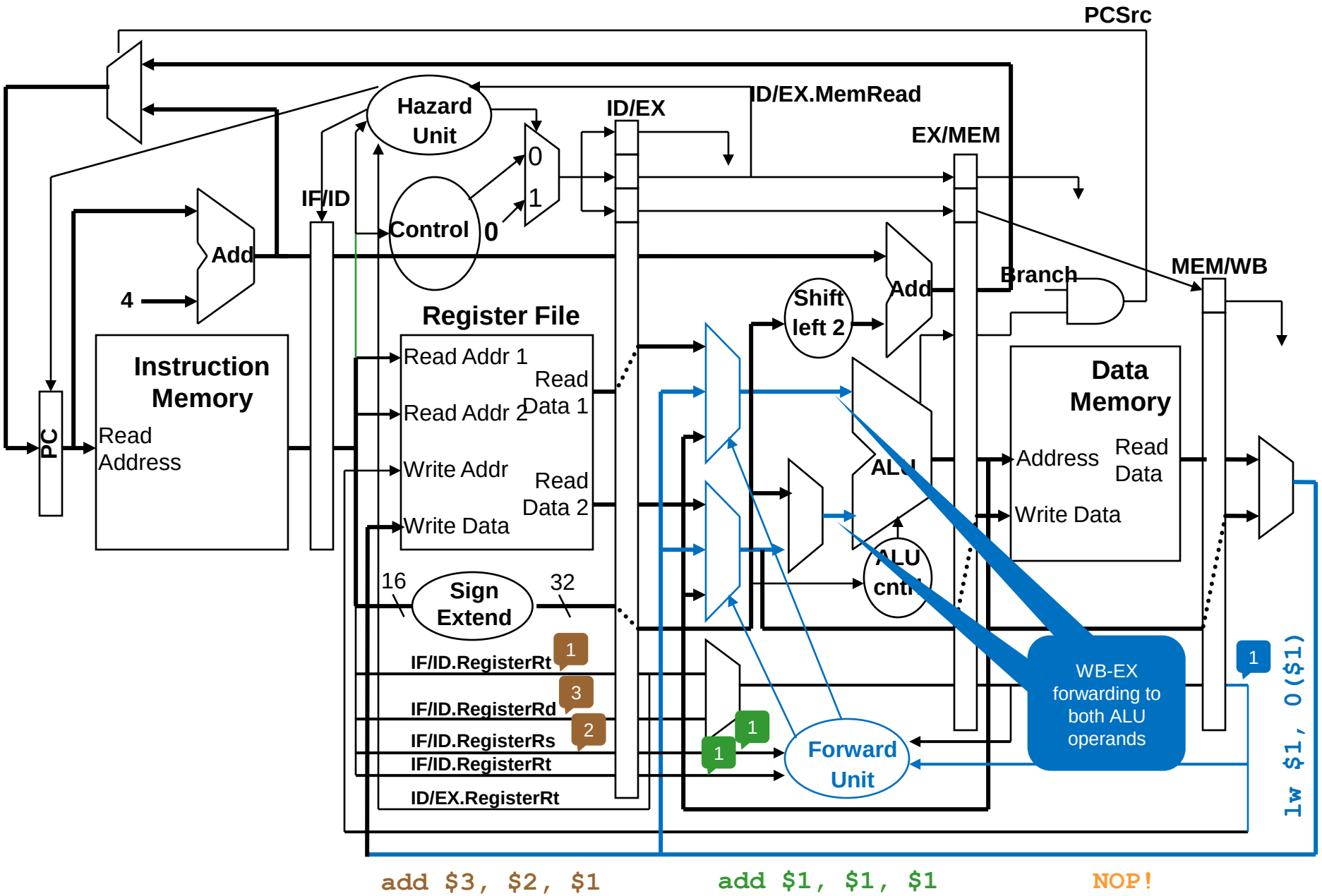
add $3, $2, $1

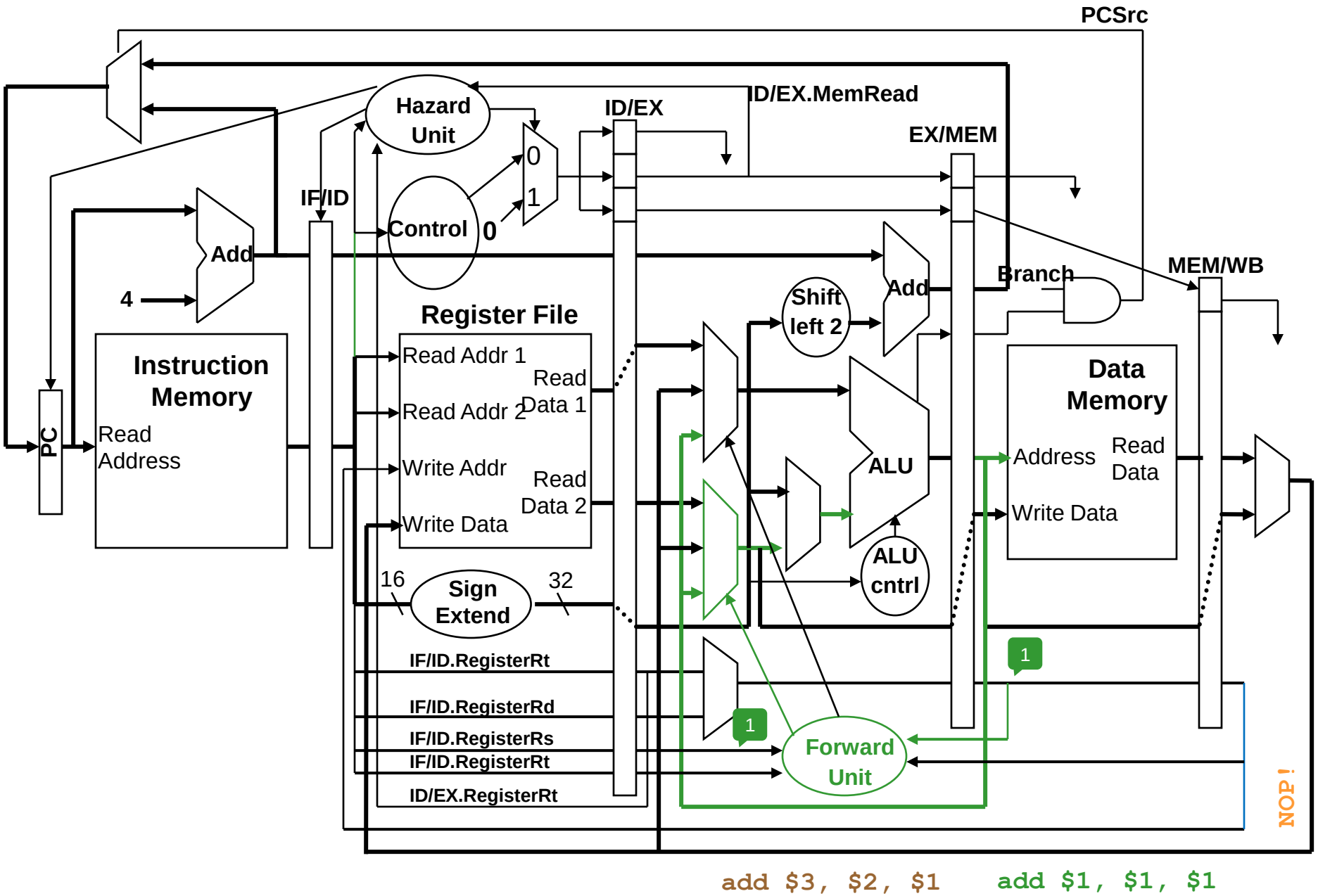
add $1, $1, $1

lw $1, 0($1)

add $1, $2, $3


PC = 0x110
PCSrc
Hazard Unit
ID/EX
0
ID/EX.MemRead
EX/MEM
IF/ID
Control
0
1
0
Add
4
Add
Branch
MEM/WB
Shift left 2
Register File
Instruction Memory
Read Address
PC
Read Addr 1
Read Addr 2
Write Addr
Write Data
Read Data 1
Read Data 2
ALU
ALU cntrl
Data Memory
Address
Read Data
Write Data
16
Sign Extend
32
PC= 0x10c
0x00411820
IF/ID.RegisterRt
IF/ID.RegisterRd
IF/ID.RegisterRs
IF/ID.RegisterRt
ID/EX.RegisterRt
Forward Unit
1
1
1
1
1
add $1, $2, $3
add $3, $2, $1
add $1, $1, $1
NOP!
lw $1, 0($1)

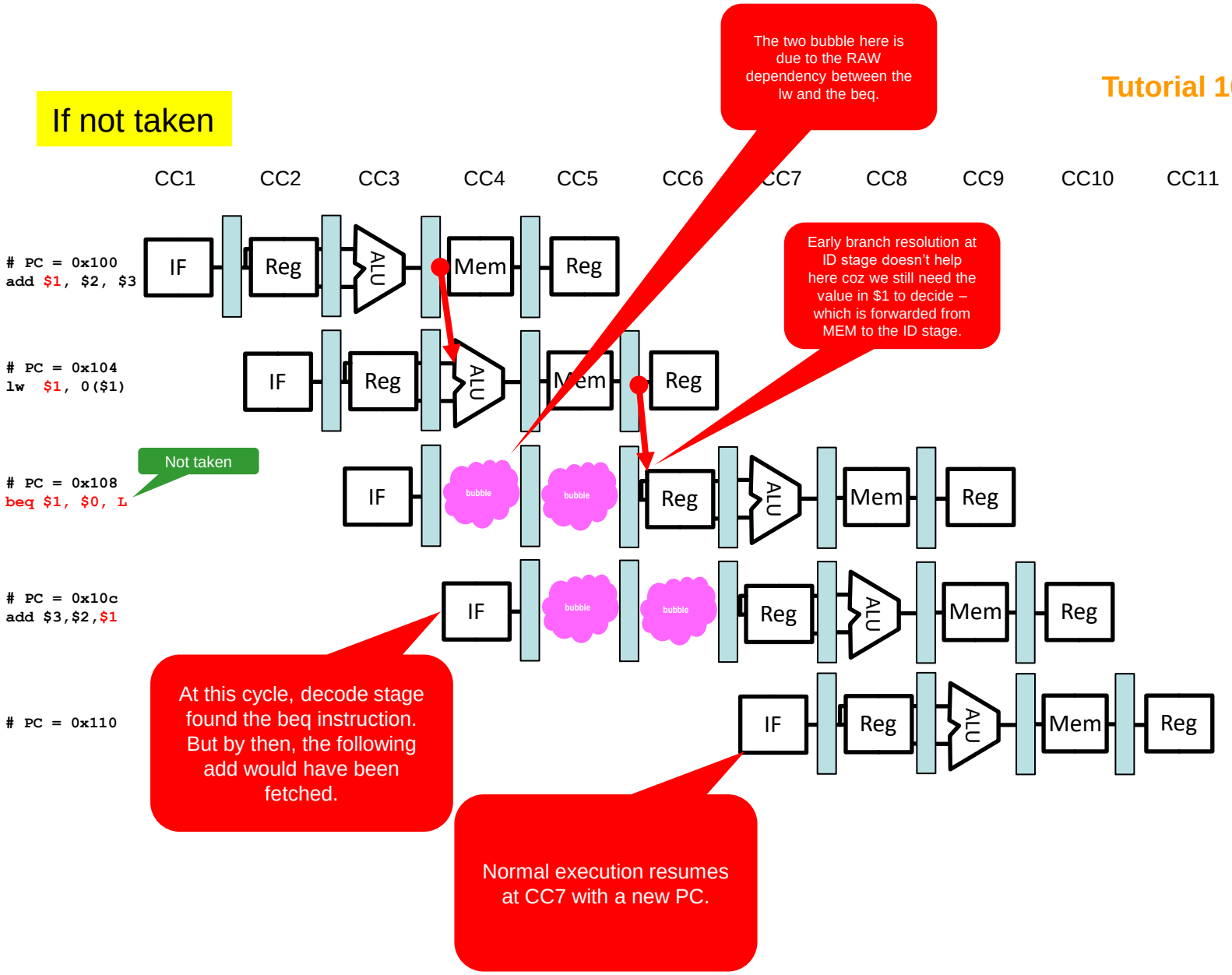

PCSrc
Hazard Unit
ID/EX.MemRead
ID/EX
EX/MEM
MEM/WB
IF/ID
Control
0
1
Add
4
Branch
Shift left 2
Register File
Instruction Memory
Read Address
PC
Read Addr 1
Read Addr 2
Write Addr
Write Data
Read Data 1
Read Data 2
Data Memory
Address
Read Data
Write Data
ALU
ALU cntrl
16
Sign Extend
32
IF/ID.RegisterRt
IF/ID.RegisterRd
IF/ID.RegisterRs
IF/ID.RegisterRt
ID/EX.RegisterRt
Forward Unit
WB-EX forwarding to both ALU operands
lw $1, 0($1)
add $3, $2, $1
add $1, $1, $1
NOP!


PCSrc
Hazard Unit
ID/EX.MemRead
ID/EX
EX/MEM
MEM/WB
IF/ID
Control
0
1
Add
4
Branch
Shift left 2
Register File
Instruction Memory
Read Address
PC
Read Addr 1
Read Addr 2
Write Addr
Write Data
Read Data 1
Read Data 2
ALU
ALU cntrl
Data Memory
Address
Read Data
Write Data
16
Sign Extend
32
IF/ID.RegisterRt
IF/ID.RegisterRd
IF/ID.RegisterRs
IF/ID.RegisterRt
ID/EX.RegisterRt
Forward Unit
NOP!
add $3, $2, $1
add $1, $1, $1

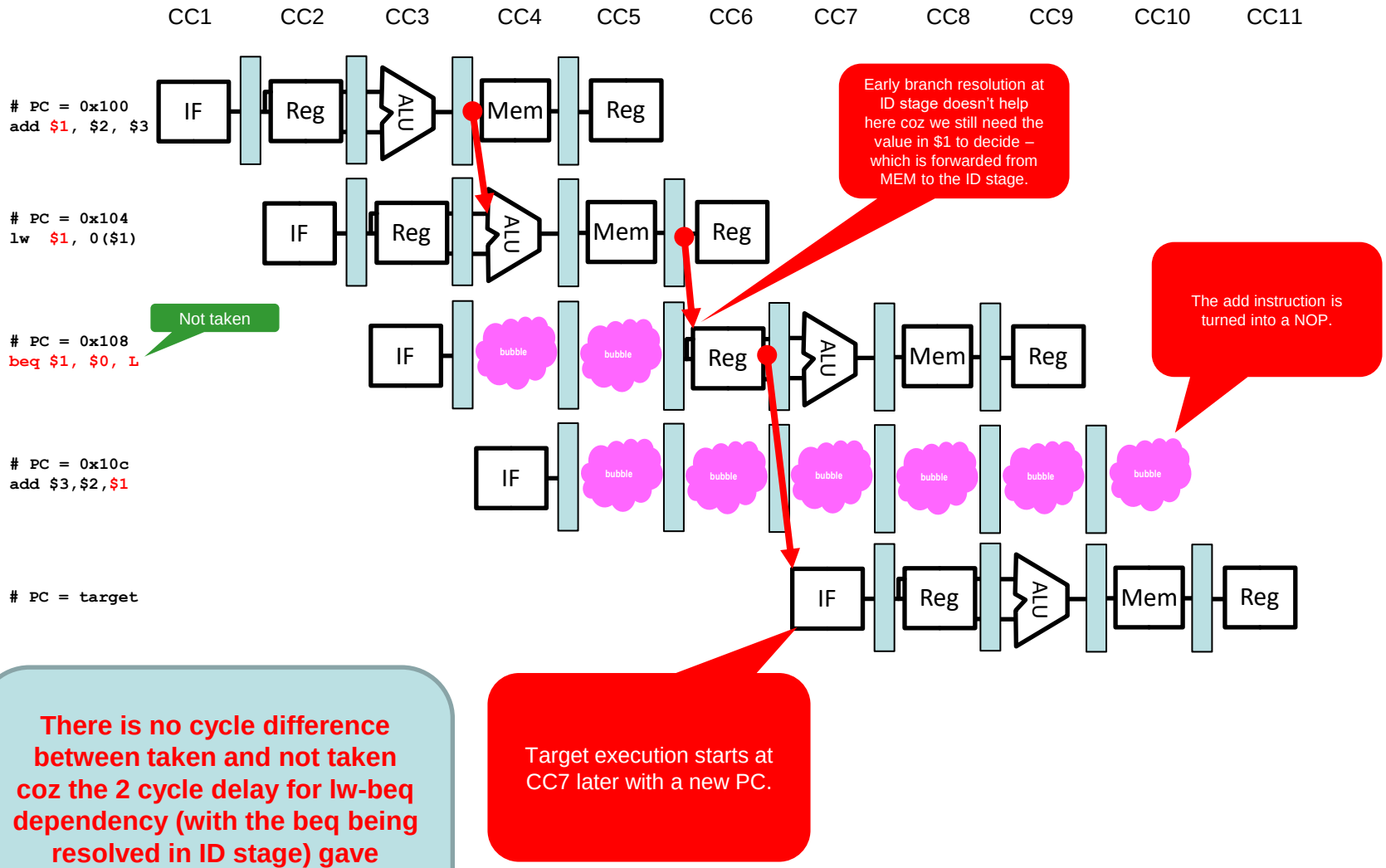
QUESTION 5

Tutorial 10 Q5a

If not taken



If taken



There is no cycle difference between taken and not taken coz the 2 cycle delay for lw-beq dependency (with the beq being resolved in ID stage) gave enough time for the PC to be changed to the target.

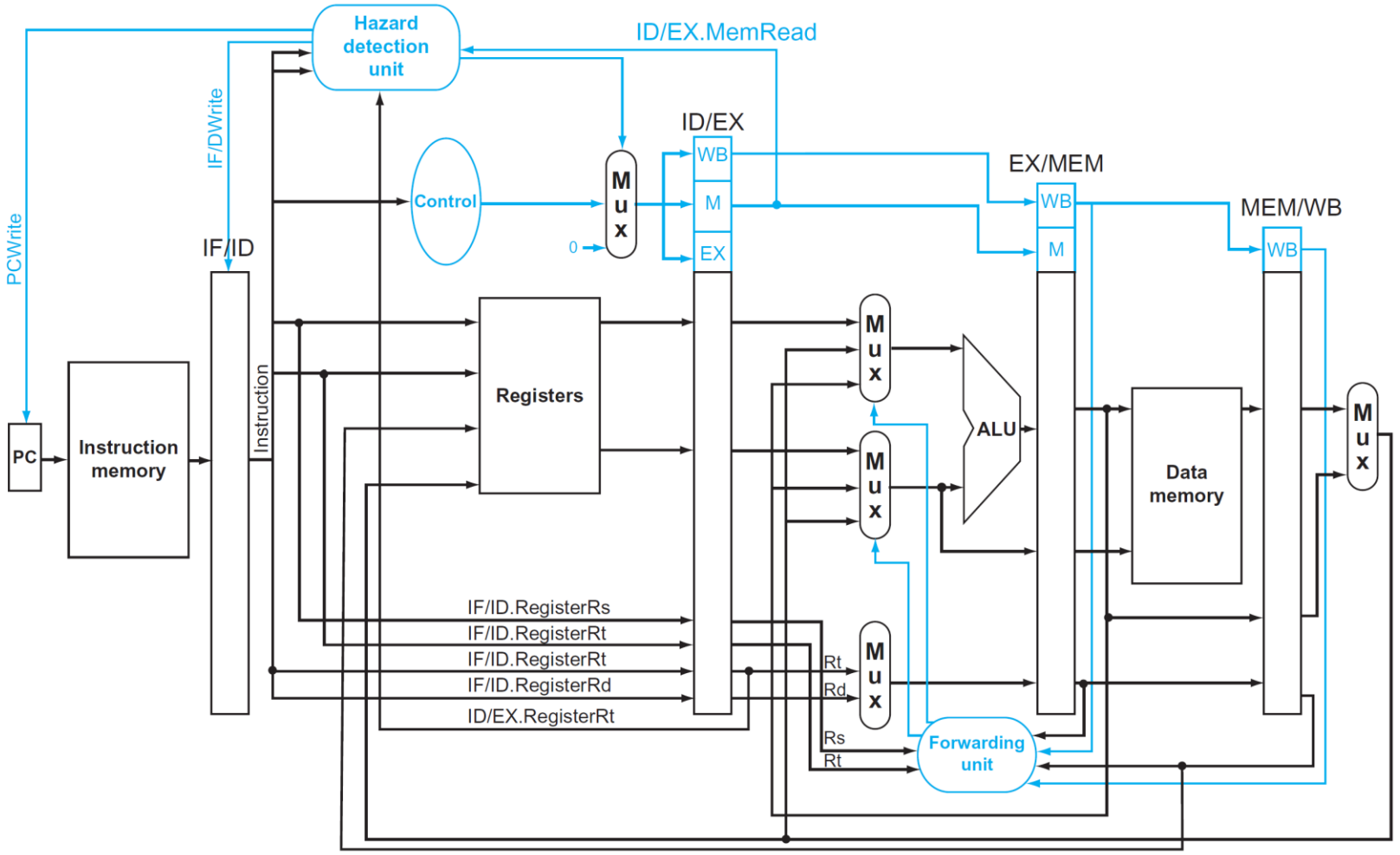


FIGURE 4.60 Pipelined control overview, showing the two multiplexors for forwarding, the hazard detection unit, and the forwarding unit. Although the ID and EX stages have been simplified—the sign-extended immediate and branch logic are missing