CSCI 210: Computer Architecture
Lecture 30: Pipelining the Datapath

Stephen Checkoway

Oberlin College

May 6, 2022

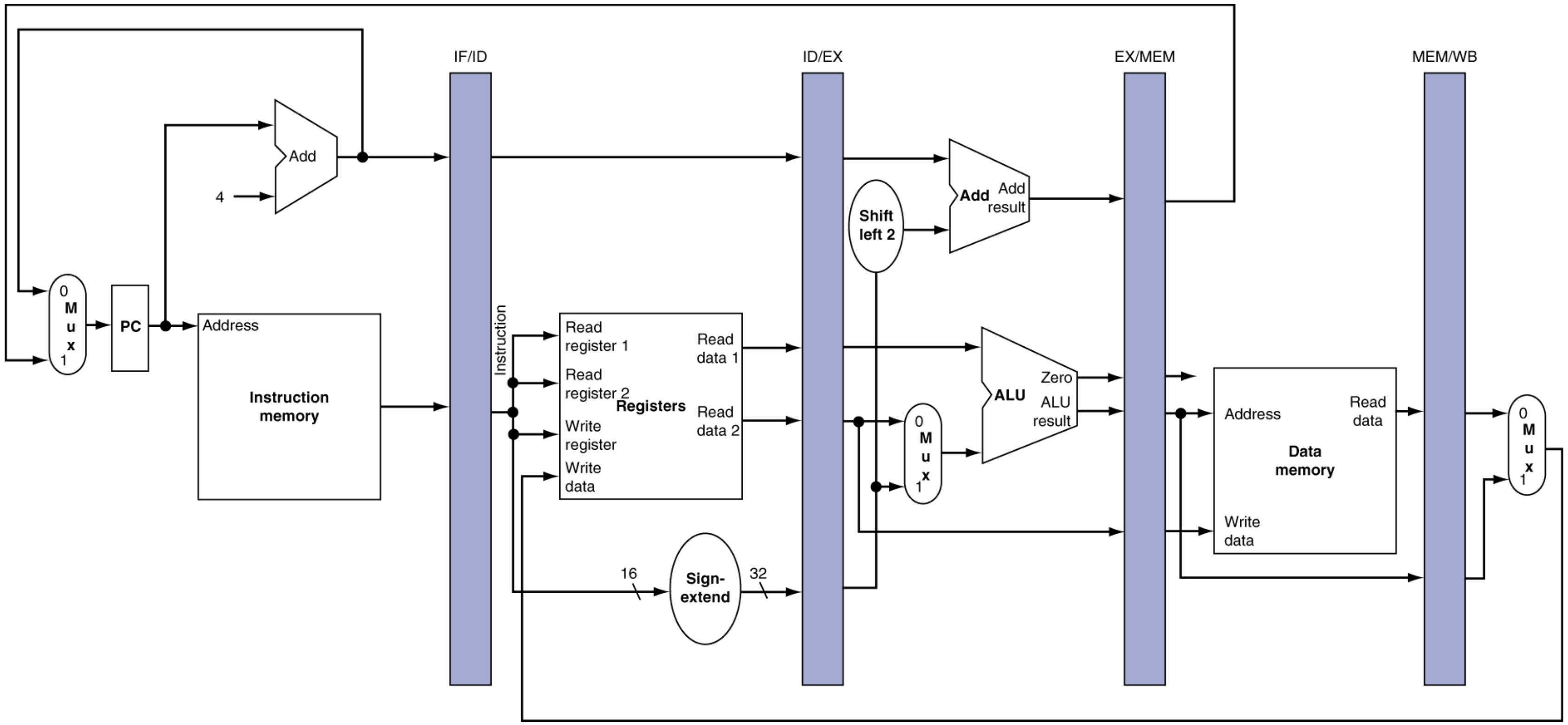
Slides from Cynthia Taylor

Announcements

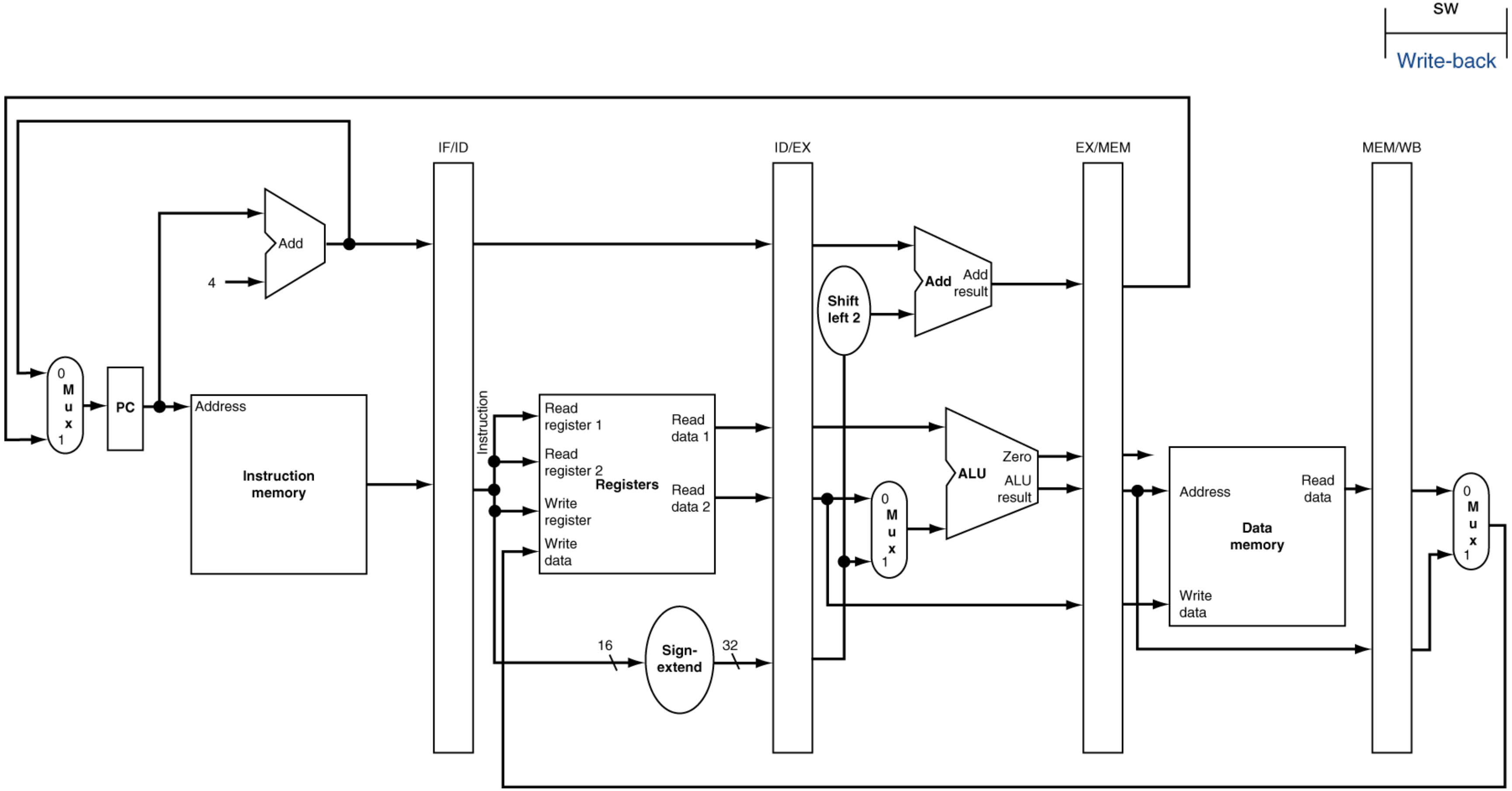
- Problem Set 10 due a week from today
- Lab 8 due Sunday
- Office Hours Friday 13:30 – 14:30

Pipeline registers

- Need registers between stages
  - – To hold information produced in previous cycle



WB for Store

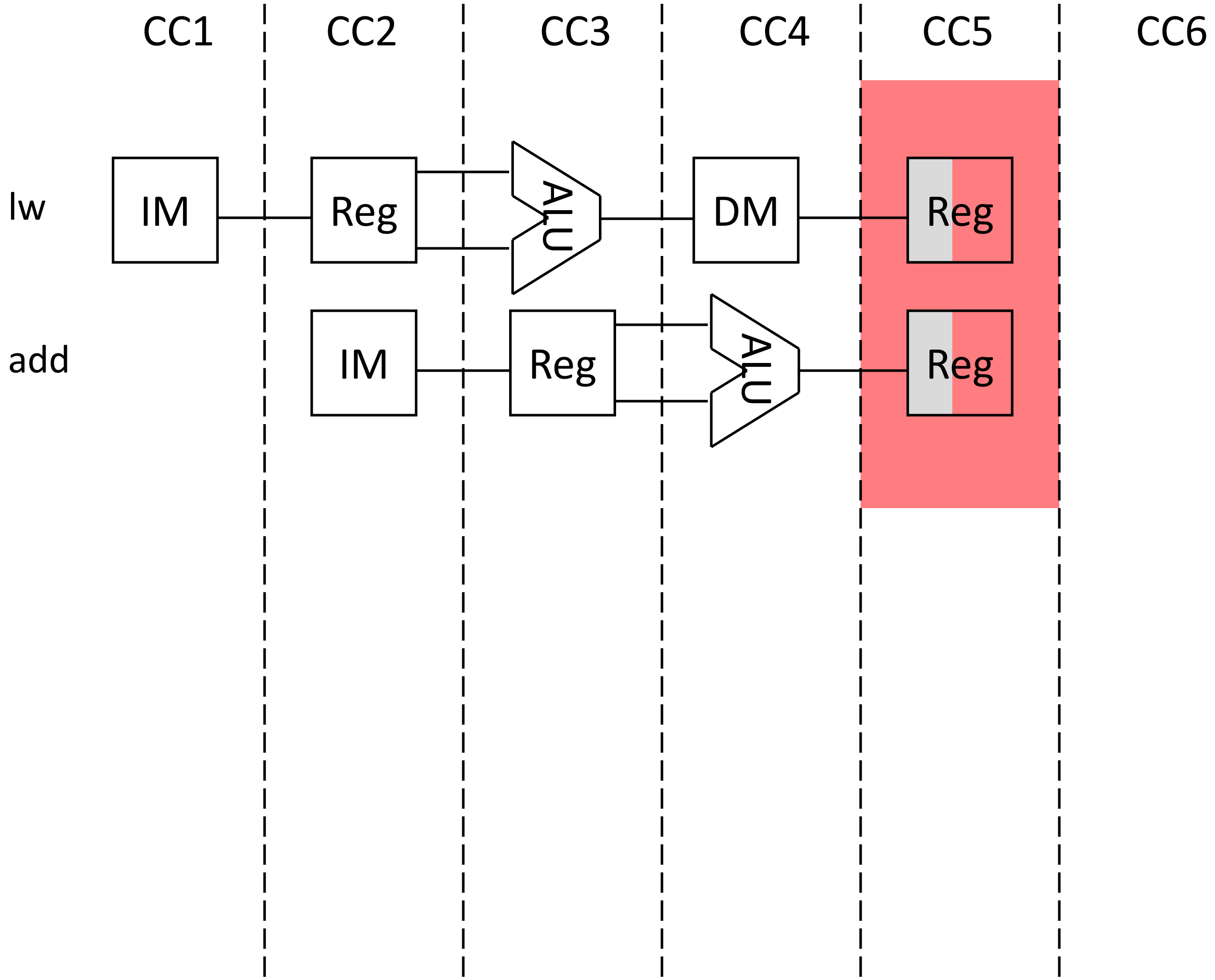

sw
Write-back
IF/ID
ID/EX
EX/MEM
MEM/WB
Add
4
0
M
u
x
1
PC
Address
Instruction memory
Instruction
Read register 1
Read register 2
Registers
Write register
Write data
Read data 1
Read data 2
16
Sign-extend
32
Shift left 2
Add
Add result
ALU
Zero
ALU result
0
M
u
x
1
Address
Read data
Data memory
Write data
0
M
u
x
1

Pipeline Stages

Should we force every instruction to go through all 5 stages? Can we break it up, with R-type taking 4 cycles instead of 5?

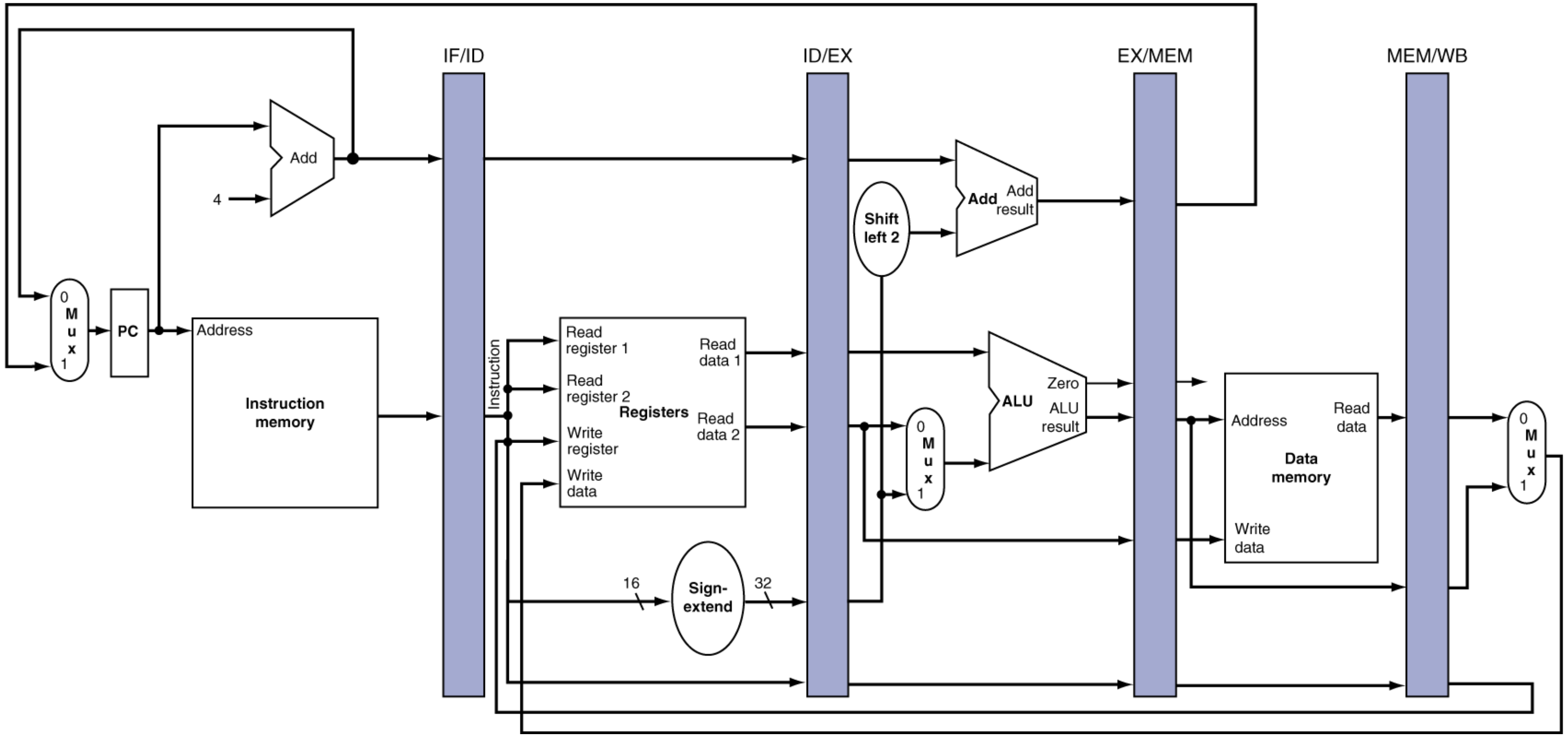
Selection	Yes/No	Reason (Choose BEST answer)
A	Yes	Decreasing R-type to 4 cycles improves instruction throughput
B	Yes	Decreasing R-type to 4 cycles improves instruction latency
C	No	Decreasing R-type to 4 cycles causes hazards
D	No	Decreasing R-type to 4 cycles causes hazards and doesn’t impact throughput
E	No	Decreasing R-type to 4 cycles causes hazards and doesn’t impact latency

Mixed Instructions in the Pipeline

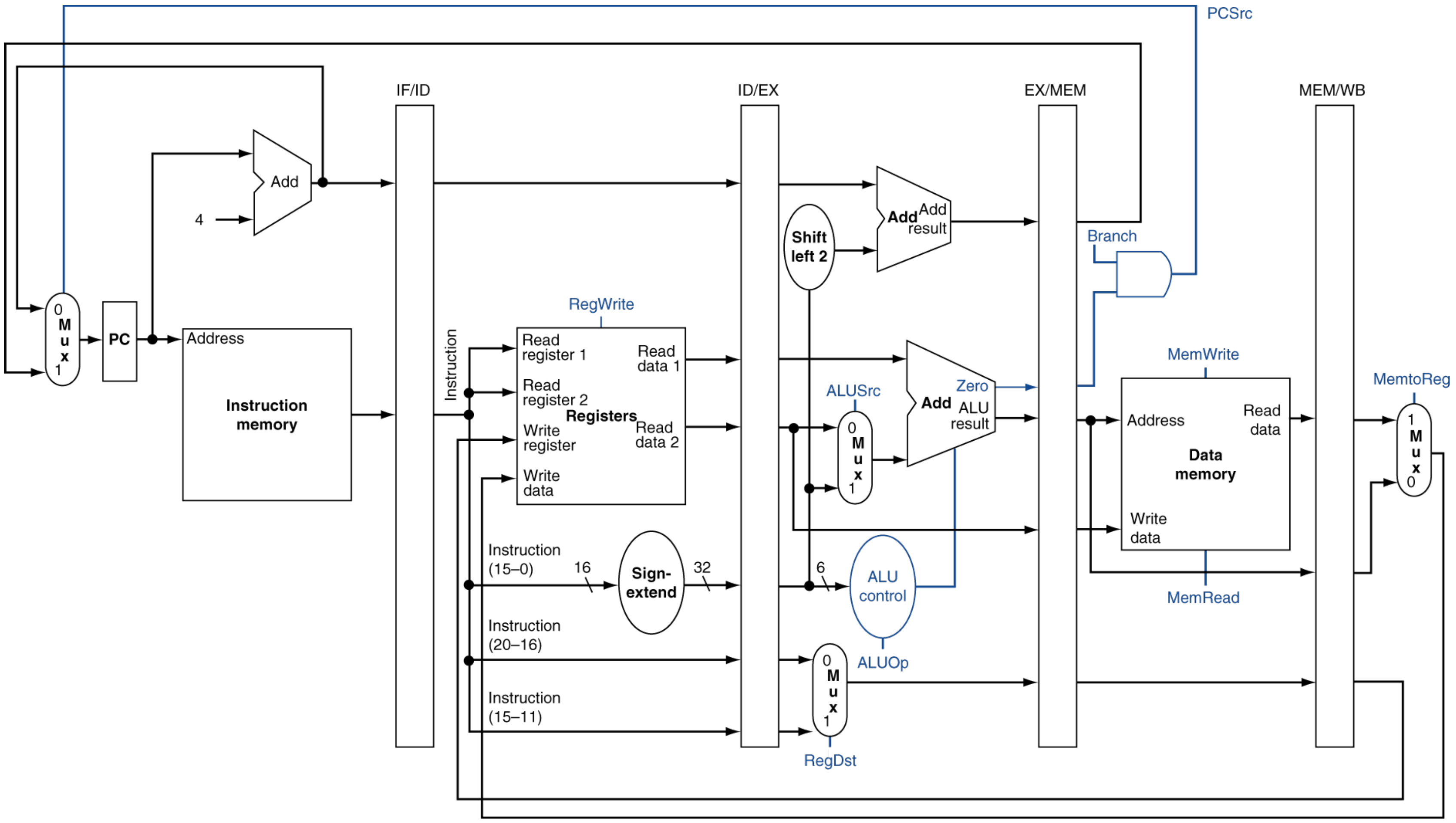


State of pipeline in a given cycle

add $14, $5, $6	lw $13, 24 ($1)	add $12, $3, $4	sub $11, $2, $3	lw $10, 20($1)
Instruction fetch	Instruction decode	Execution	Memory	Write-back



Pipelined Control



How do we control our pipelined CPU?

A. We need to add new control signals.

B. We need to forward the control values to the correct stage.

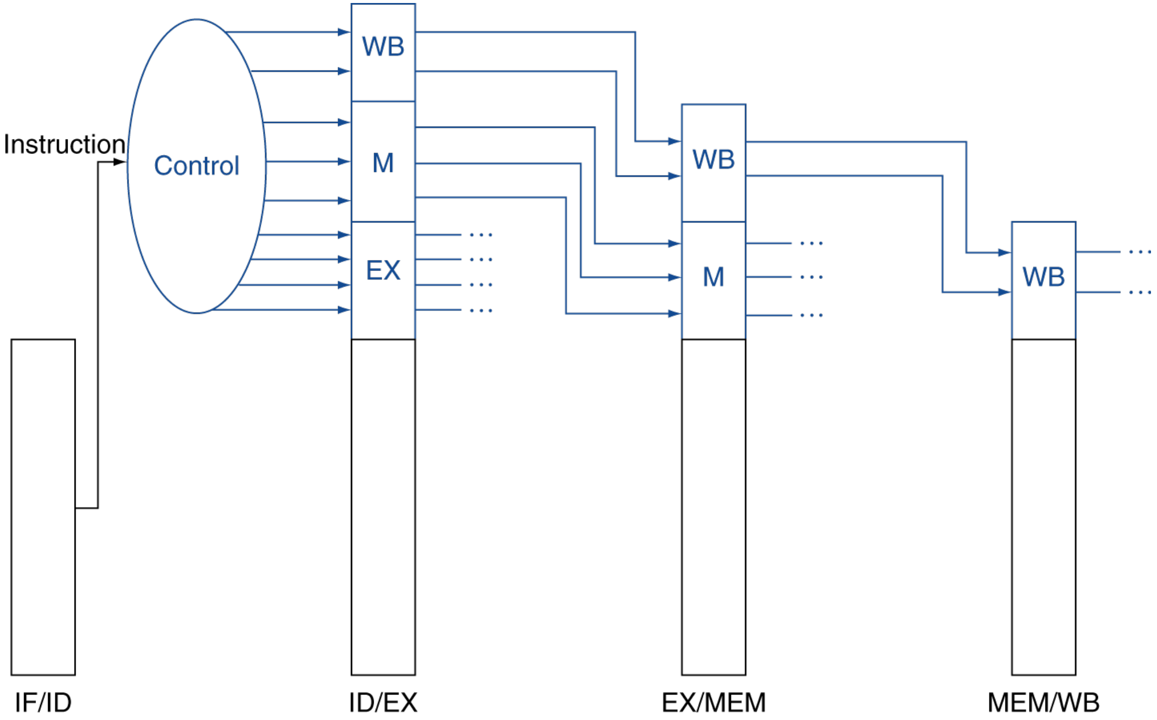
C. We don’t need to do anything special; it will work the way it is.

Pipeline Control

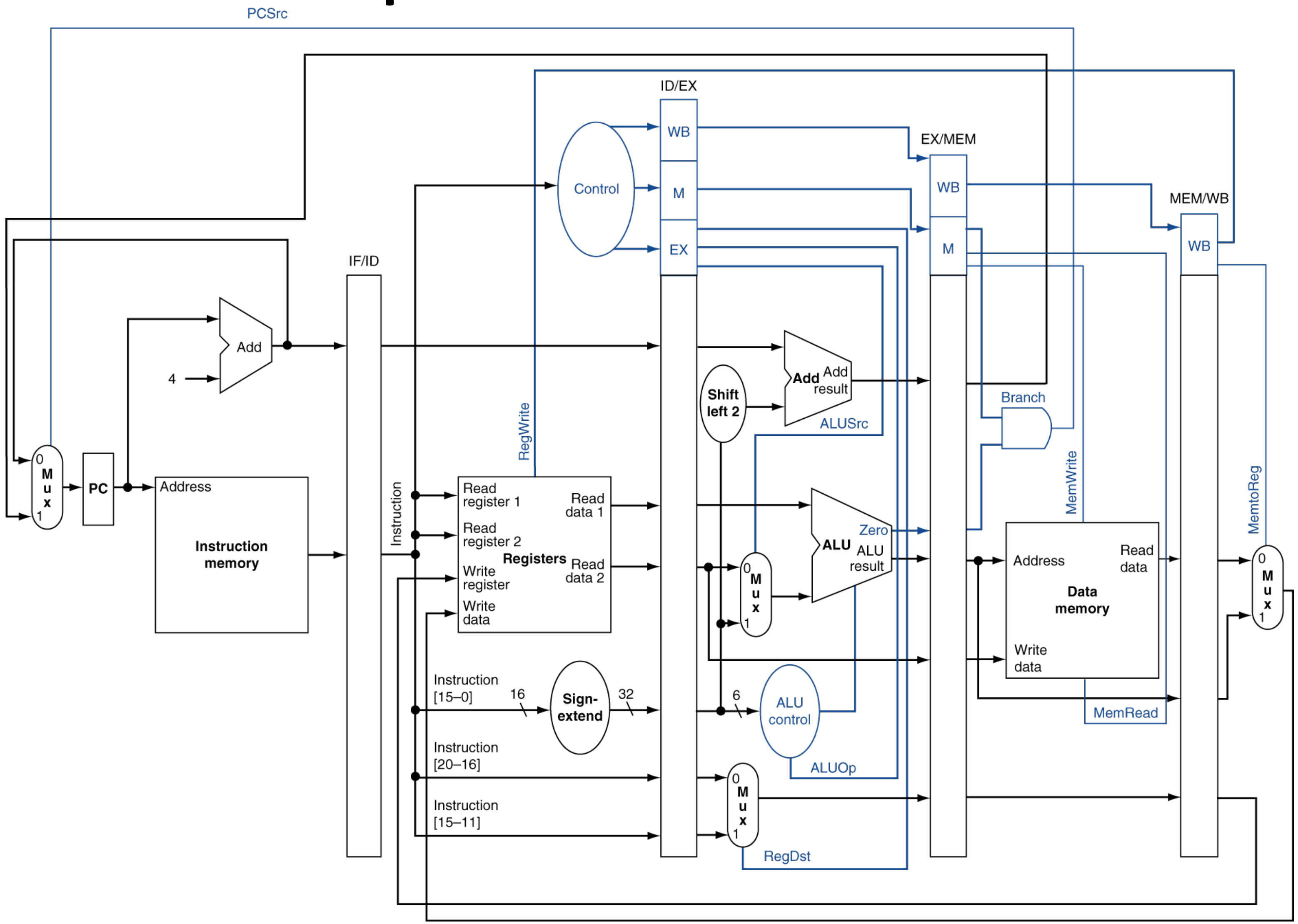
- IF Stage: read Instr Memory (always) and write PC (on System Clock)
- ID Stage: no optional control signals to set
- EX, MEM, and WB stages have control signals
  - The pipeline registers will need to store the control signals

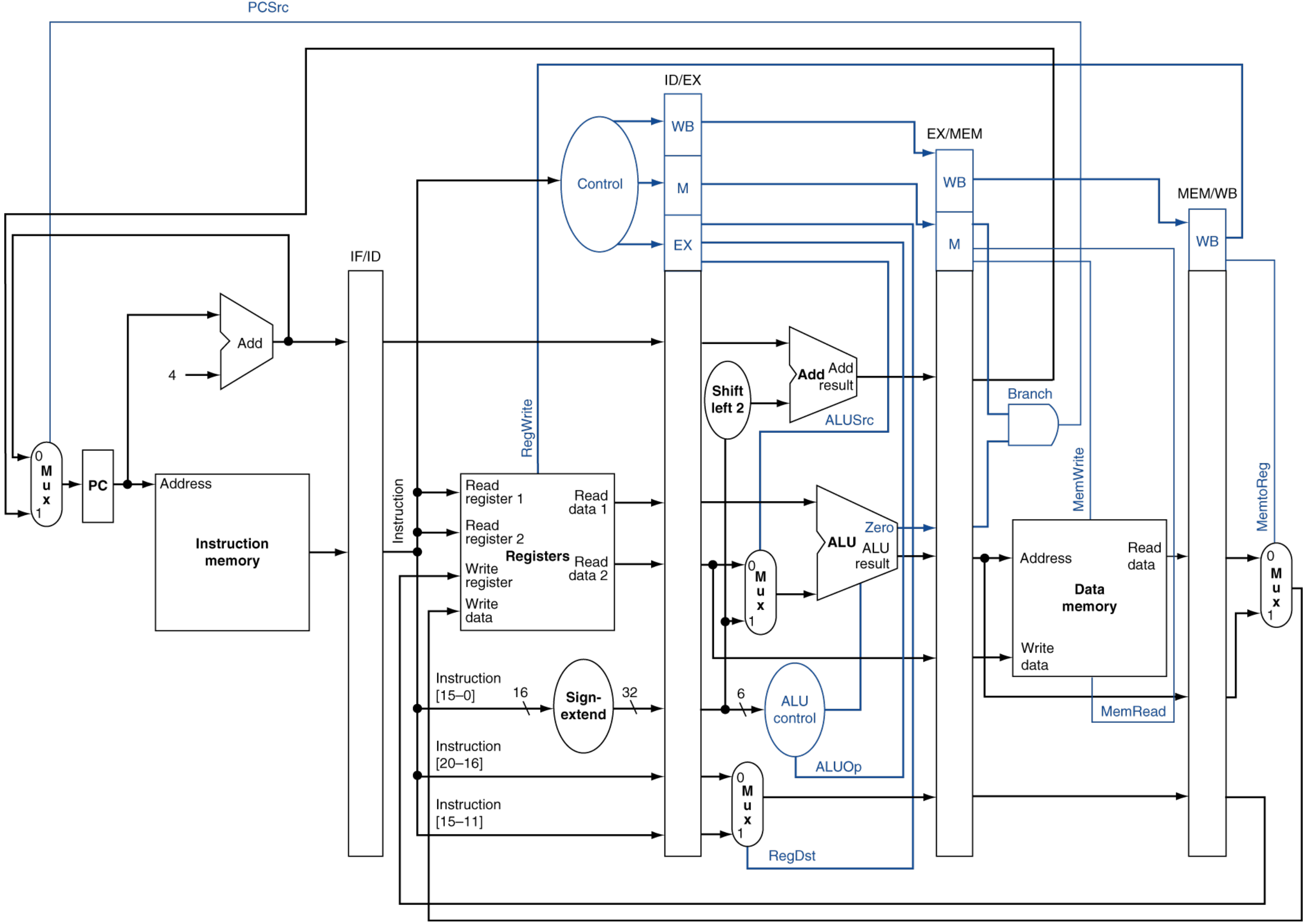
Pipelined Control

Control signals derived from instruction



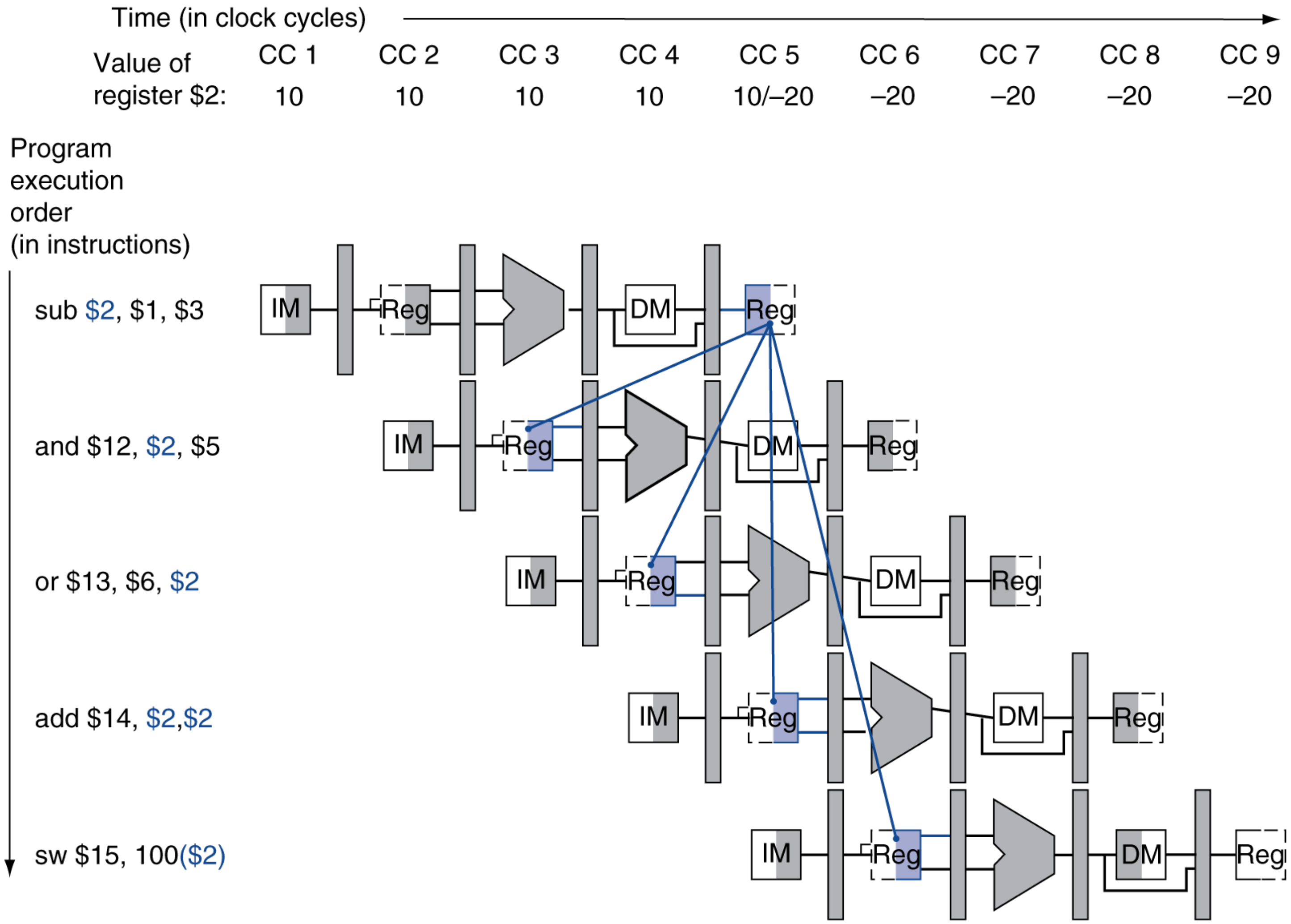
Pipelined Control





	EX Stage				MEM Stage			WB Stage	
	RegDst	ALUOp1	ALUOp0	ALUSrc	Brch	MemRead	MemWrite	RegWrite	Mem toReg
R	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

Dependencies & Forwarding



Time (in clock cycles)

Value of register $2:	CC 1	CC 2	CC 3	CC 4	CC 5	CC 6	CC 7	CC 8	CC 9
	10	10	10	10	10/–20	–20	–20	–20	–20

Program execution order (in instructions)

sub $2, $1, $3

and $12, $2, $5

or $13, $6, $2

add $14, $2,$2

sw $15, 100($2)

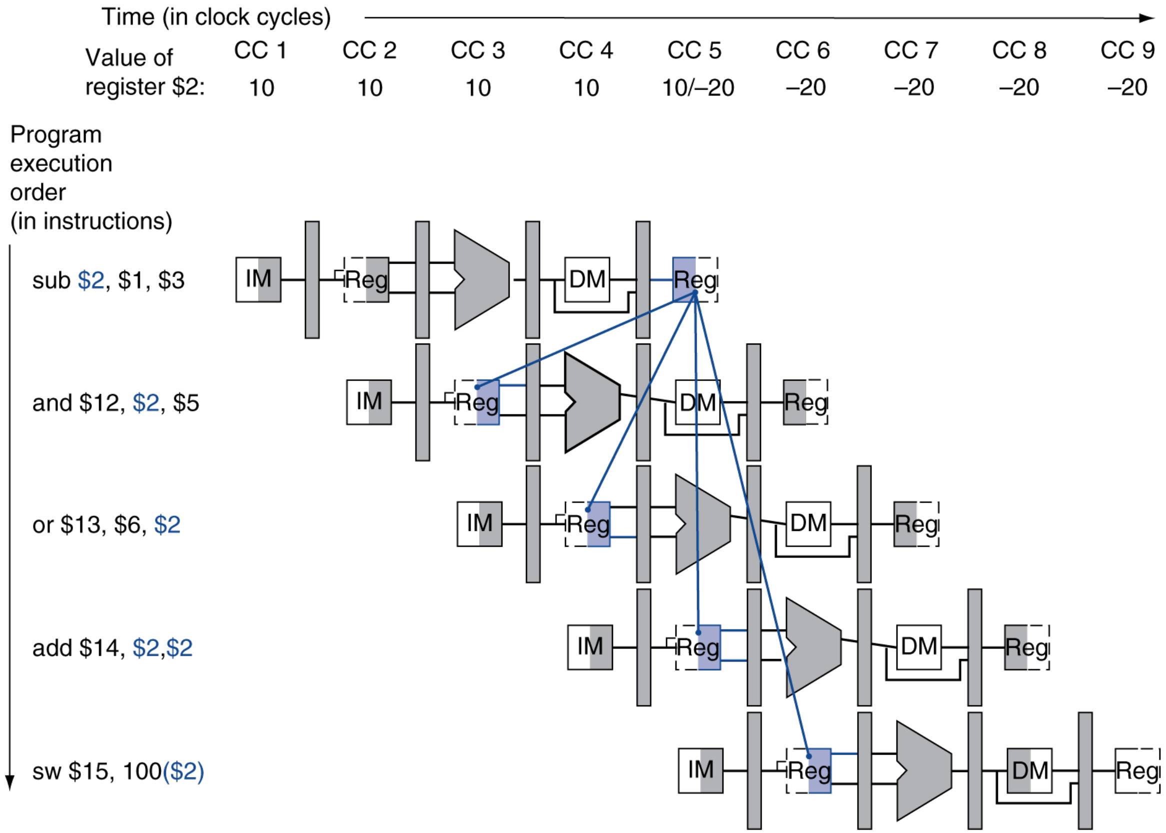
We can BEST solve these data hazards

A. By stalling.

B. By forwarding.

C. By combining forwards and stalls.

D. By doing something else.



Time (in clock cycles)

	CC 1	CC 2	CC 3	CC 4	CC 5	CC 6	CC 7	CC 8	CC 9
Value of register $2:	10	10	10	10	10/–20	–20	–20	–20	–20

Program execution order (in instructions)

sub $2, $1, $3

and $12, $2, $5

or $13, $6, $2

add $14, $2,$2

sw $15, 100($2)

Reading

- Next lecture: Data Hazards
  - Section 5.8