Chapter 4

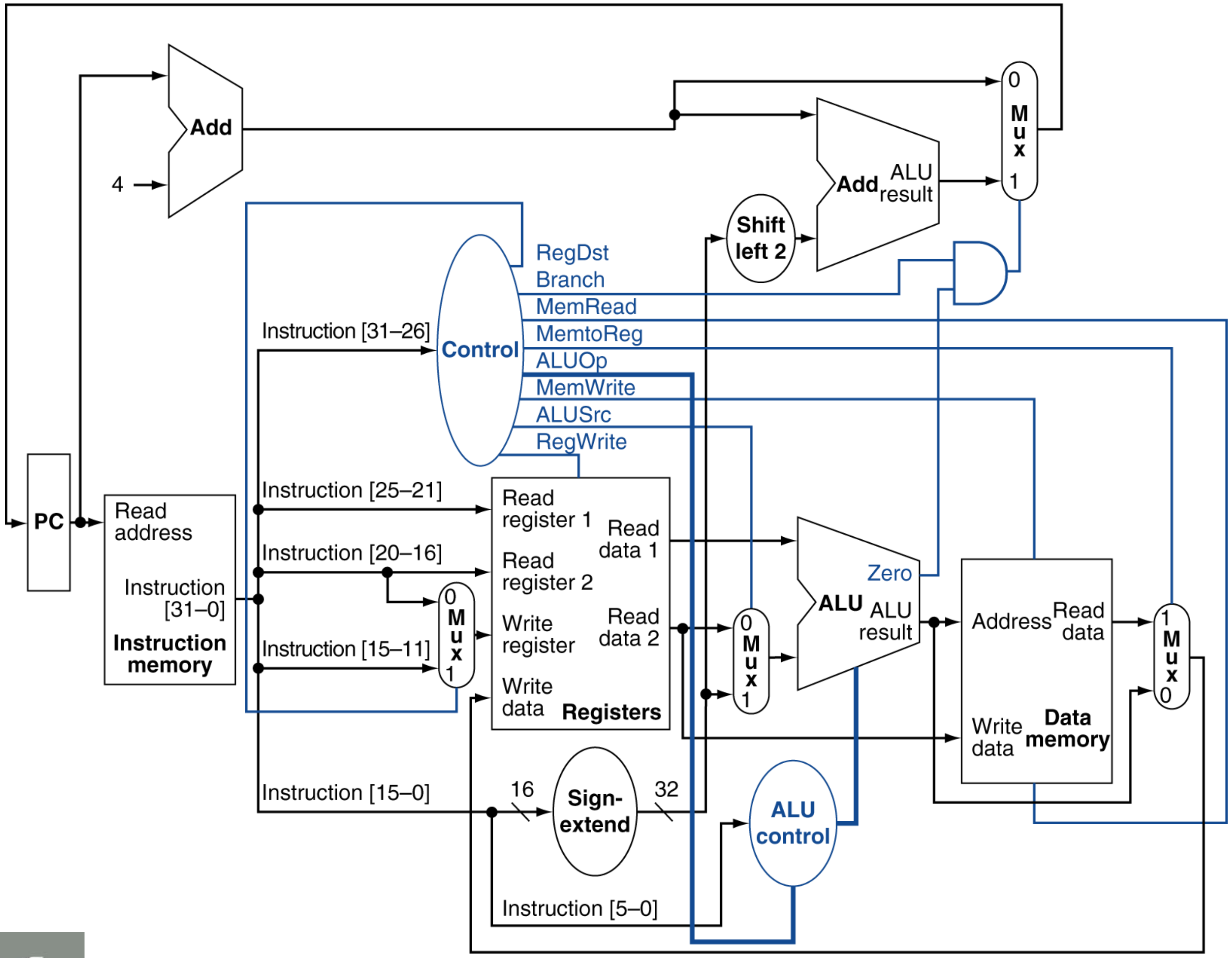
The Processor

Dr. Randa Mohamed

Agenda

- Performance Issues
- MIPS Pipeline:
  - An overview of pipelining (Process, Performance)
  - Pipelined Datapath and Control

Remember: Datapath With Control



Performance Issues

- Longest delay determines clock period
  - Critical path: load instruction
  - Instruction memory → register file → ALU → data memory → register file

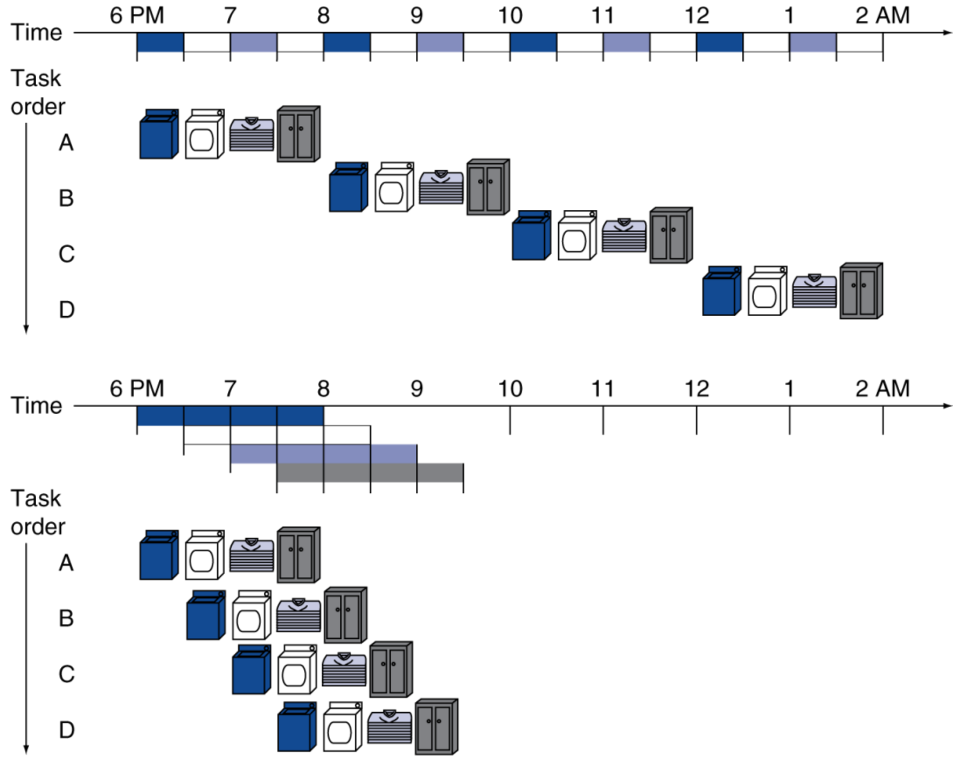
Instr	Instr fetch	Register read	ALU op	Memory access	Register write	Total time
lw	200ps	100 ps	200ps	200ps	100 ps	800ps
sw	200ps	100 ps	200ps	200ps		700ps
R-format	200ps	100 ps	200ps		100 ps	600ps
beq	200ps	100 ps	200ps			500ps

Performance Issues

- Not feasible to vary period for different instructions
- We will improve performance by pipelining

Pipelining Analogy

- Pipelined laundry: overlapping execution
  - Parallelism improves performance



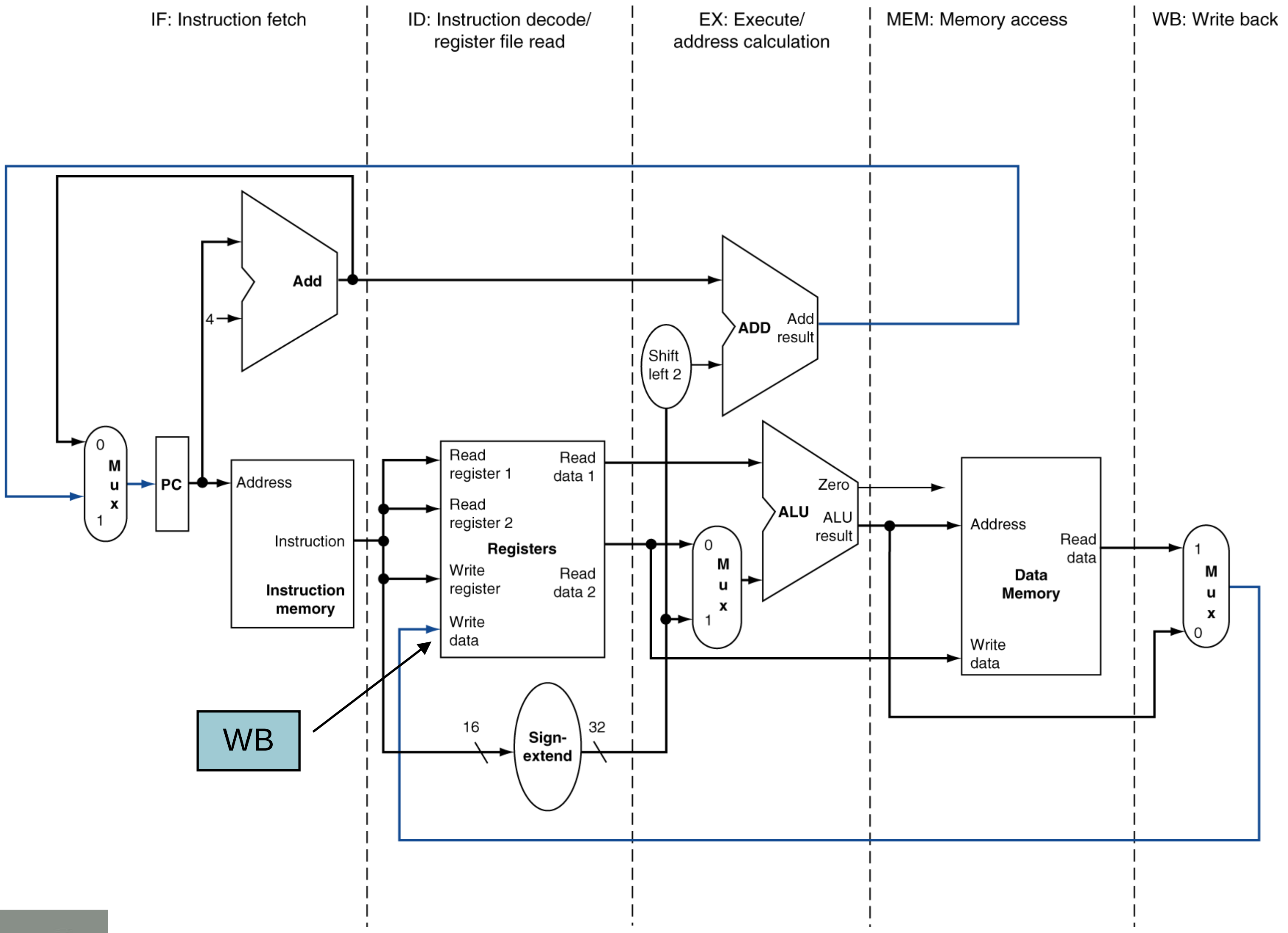
- Four loads:
  - Speedup = 8/3.5 = 2.3

MIPS Pipeline

- Five stages, one step per stage
  1. IF: Instruction fetch from memory
  2. ID: Instruction decode & register read
  3. EX: Execute operation or calculate address
  4. MEM: Access memory operand
  5. WB: Write result back to register

IF	**Dec**	**Exec**	**Mem**	**WB**

MIPS Pipelined Datapath



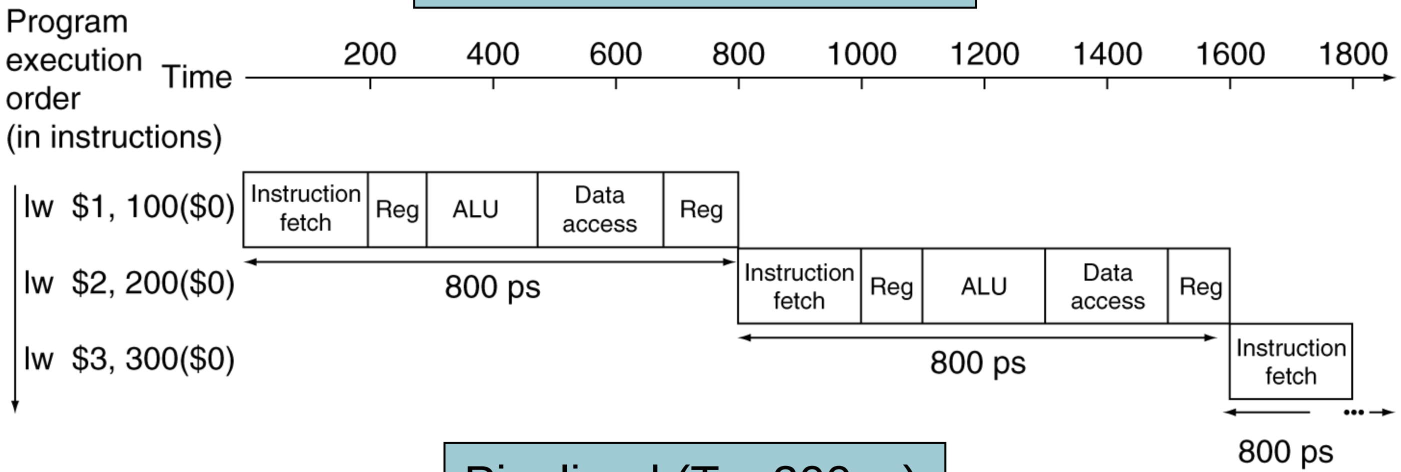
Pipeline Performance

- Assume time for stages is
  - 100ps for register read or write
  - 200ps for other stages
- Compare pipelined datapath with single-cycle datapath

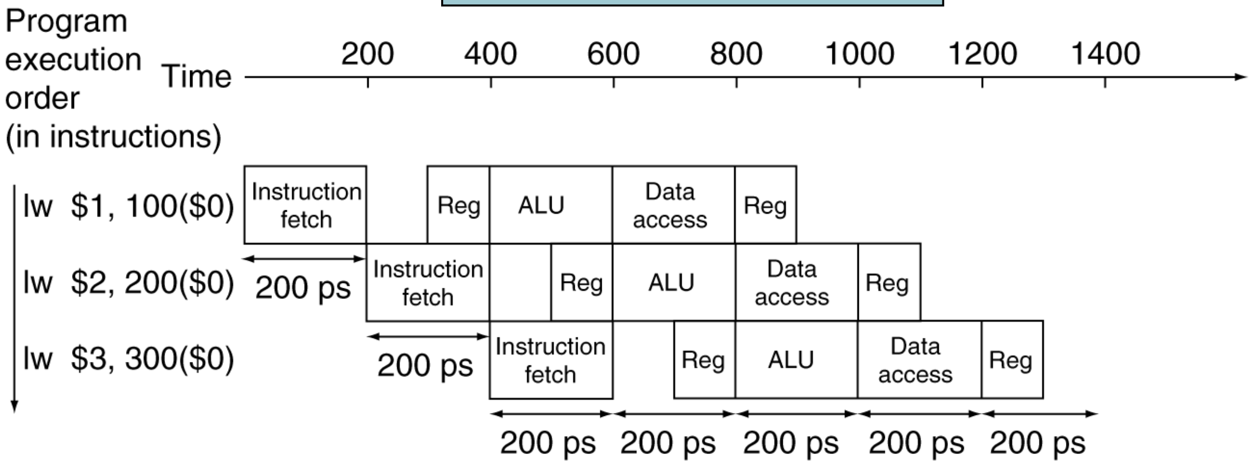
Instr	Instr fetch	Register read	ALU op	Memory access	Register write	Total time
lw						
sw						
R-format						
beq						

Pipeline Performance

Single-cycle (T_c= 800ps)



Pipelined (T_c= 200ps)

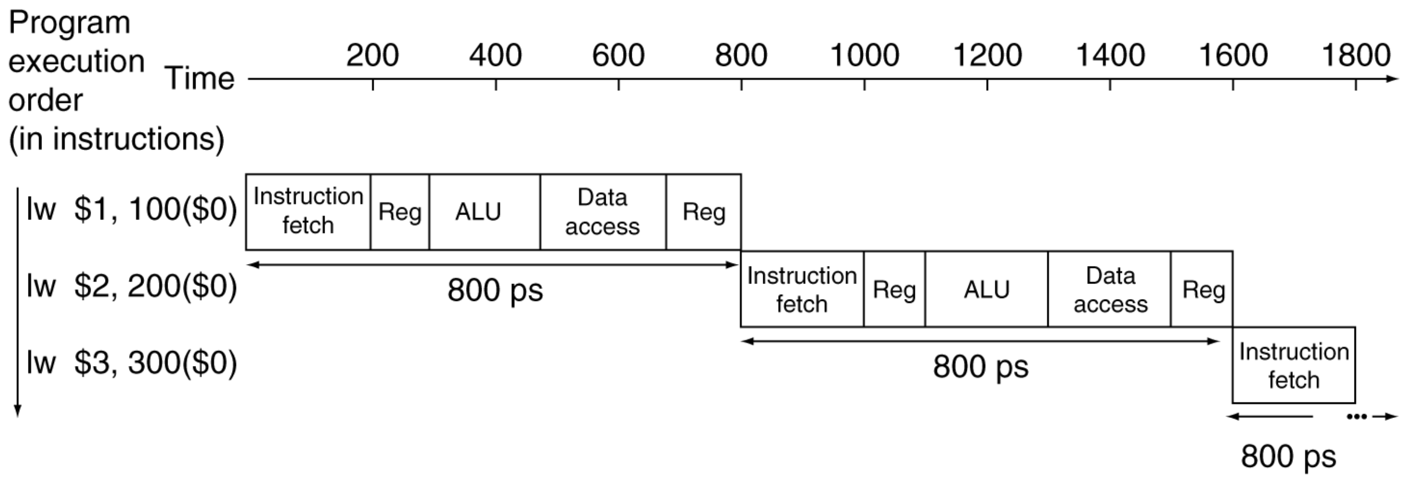


Pipeline Speedup

- Speedup = $\dfrac{\text{CPUtime}_N}{\text{CPUtime}_P}$
  - $\text{CPUtime}_N = \text{IC} \times T_N$
  - $\text{CPUtime}_P = \text{No. Stages} \times T_P + (\text{IC}-1) \times T_P$

Pipeline Speedup

- $\text{Speedup} = \dfrac{\text{CPUtime}_N}{\text{CPUtime}_P}$
  - $\text{CPUtime}_N = \text{IC} \times T_N$ 3×800=2400

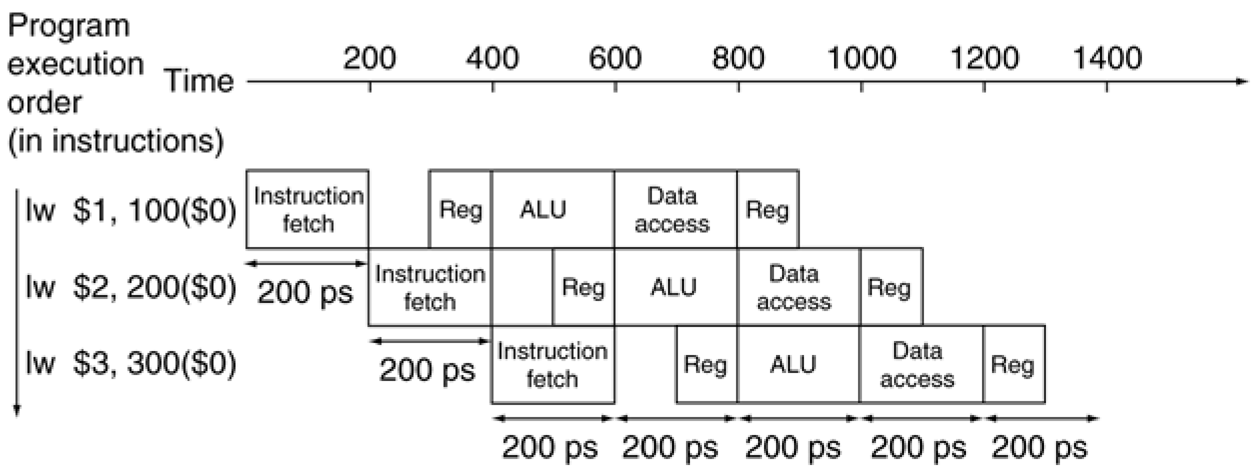


Single-cycle (T_c= 800ps)

Pipeline Speedup

- Speedup = $\dfrac{\text{CPUtime}_N}{\text{CPUtime}_P}$
  - $\text{CPUtime}_P = \text{No. Stages} \times T_P + (\text{IC}-1) \times T_P$

$(5\times200)+(2\times200)=1400$



Pipelined (T_c= 200ps)

Pipeline Speedup

- Speedup = $\dfrac{\text{CPUtime}_N}{\text{CPUtime}_P}$
  - $\text{CPUtime}_N = \text{IC} \times T_N$
  - $\text{CPUtime}_P = \text{No. Stages} \times T_P + (\text{IC}-1) \times T_P$
- Speedup = $\dfrac{\text{IC} \times T_N}{\text{No. Stages} \times T_P + (\text{IC}-1) \times T_P}$

  $= \dfrac{\text{IC} \times T_N}{T_P\,(\text{No. Stages} + \text{IC}-1)}$

Pipeline Speedup

- Speedup = $\dfrac{\text{CPUtime}_N}{\text{CPUtime}_P}$
  - $\text{CPUtime}_N = \text{IC} \times T_N$
  - $\text{CPUtime}_P = \text{No. Stages} \times T_P + (\text{IC}-1) \times T_P$

As IC is increased, IC>> No.Stages-1

- Speedup = $\dfrac{\text{IC} \times T_N}{\text{No. Stages} \times T_P + (\text{IC}-1) \times T_P}$

$$= \frac{\text{IC} \times T_N}{T_P\,(\text{No. Stages} -1+ \text{IC})}$$

Pipeline Speedup

IC=3, No.Stages =5, T_P=200,T_N=800

- $$\text{Speedup} = \frac{\text{CPUtime}_N}{\text{CPUtime}_P}$$
  - $\text{CPUtime}_N = \text{IC} \times T_N$ 3×800=2400
  - $\text{CPUtime}_P = \text{No. Stages} \times T_P + (\text{IC-1}) \times T_P$ (5×200)+ (2×200)=1400
- $$\text{Speedup} = \frac{\text{IC} \times T_N}{\text{No. Stages} \times T_P + (\text{IC-1}) \times T_P}$$ 1.7

$$= \frac{\text{IC} \times T_N}{T_P \times \text{IC}} = \frac{T_N}{T_P} = \frac{\text{No. Stages} \times T_P}{T_P}$$

= No. Stages

Pipeline Speedup

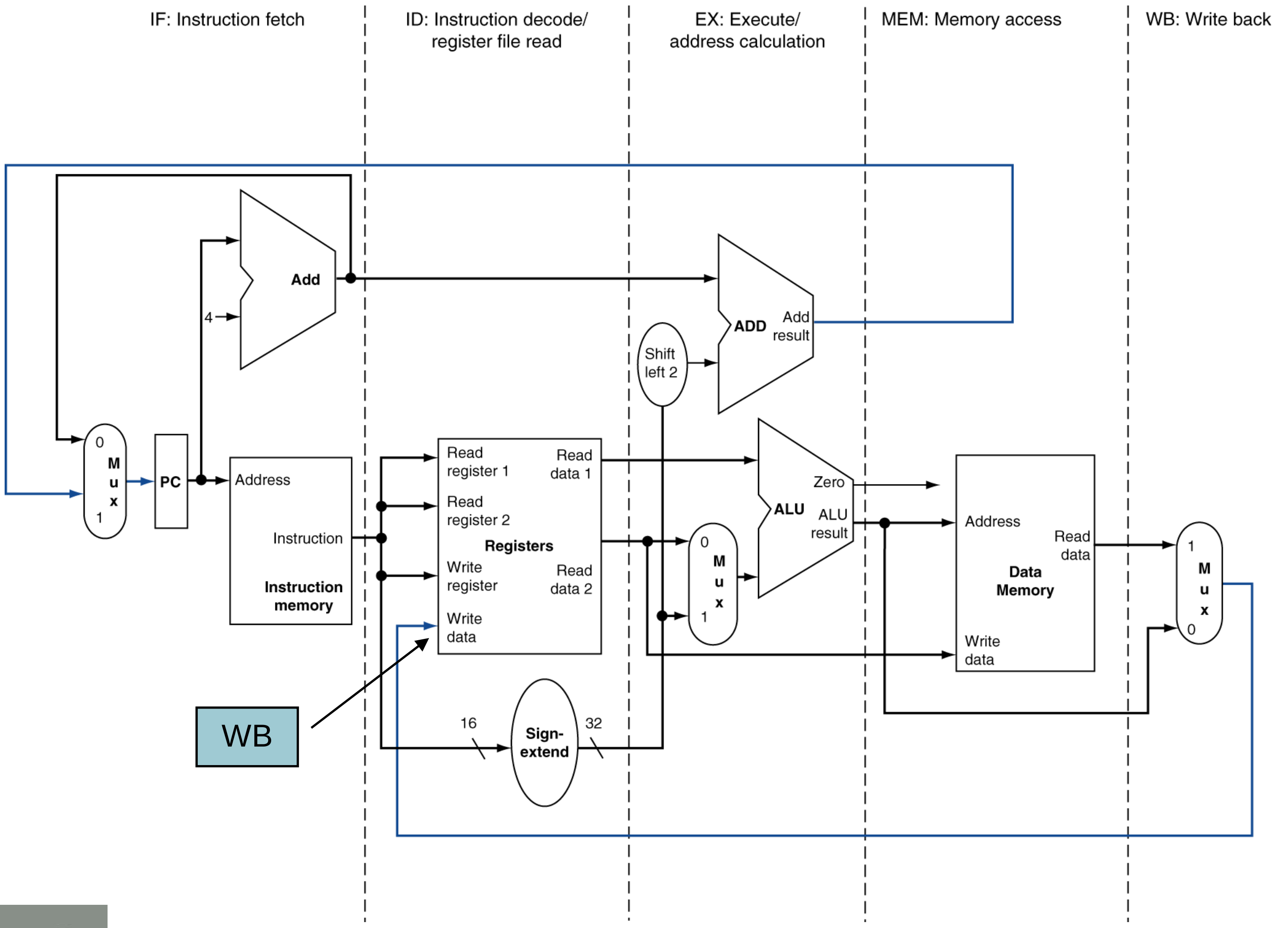
IC=1000, No.Stages =5, T_P=200,T_N=800

- $\text{Speedup} = \dfrac{\text{CPUtime}_N}{\text{CPUtime}_P}$
  - $\text{CPUtime}_N = \text{IC} \times T_N$ 1000×800=800000
  - $\text{CPUtime}_P = \text{No. Stages} \times T_P + (\text{IC}-1) \times T_P$ (5×200)+ (999×200)=200800
- $\text{Speedup} = \dfrac{\text{IC} \times T_N}{\text{No. Stages} \times T_P + (\text{IC}-1) \times T_P}$ 3.9

$$= \frac{\text{IC} \times T_N}{T_P \times \text{IC}} = \frac{T_N}{T_P} = \frac{\text{No. Stages} \times T_P}{T_P}$$

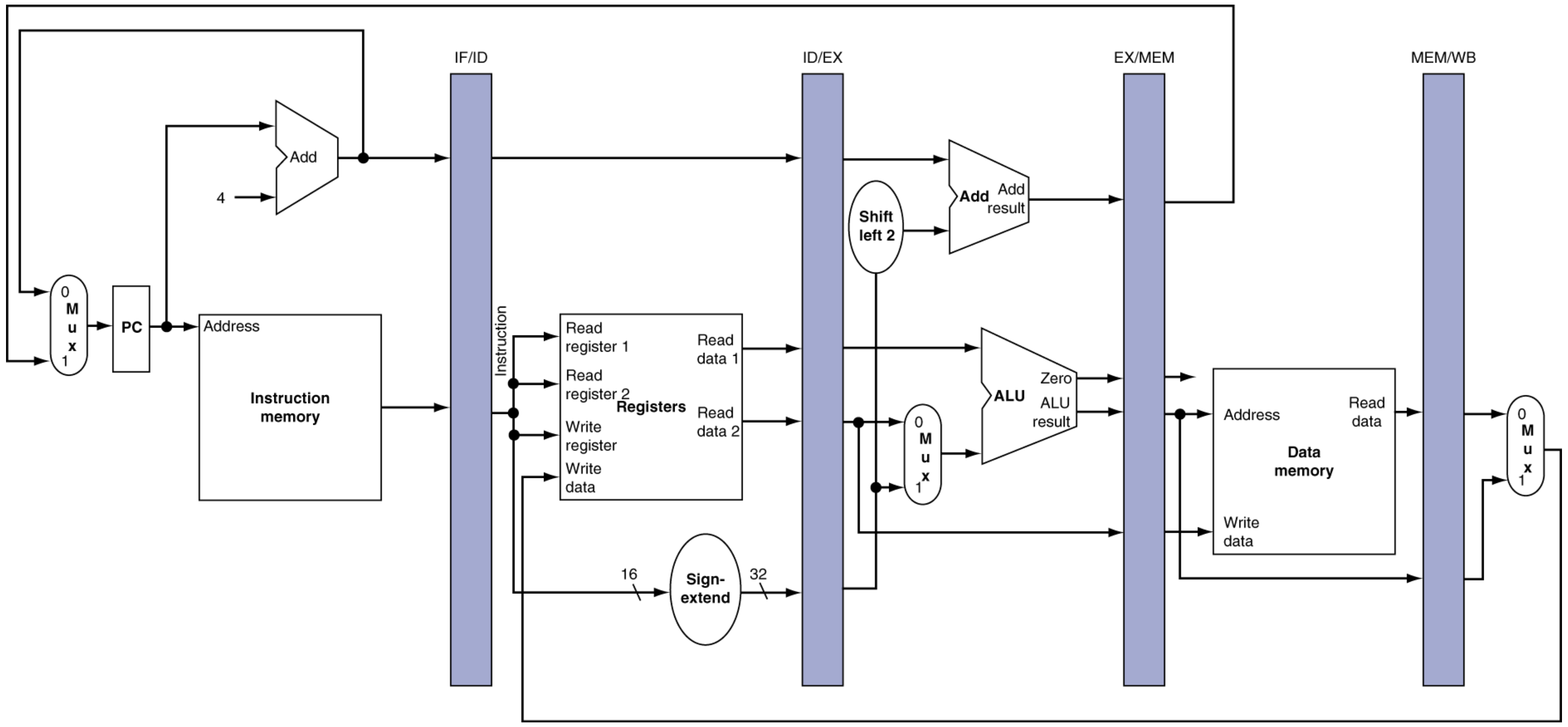
$$= \text{No. Stages}$$

MIPS Single-Cycle Datapath



Pipelined Datapath

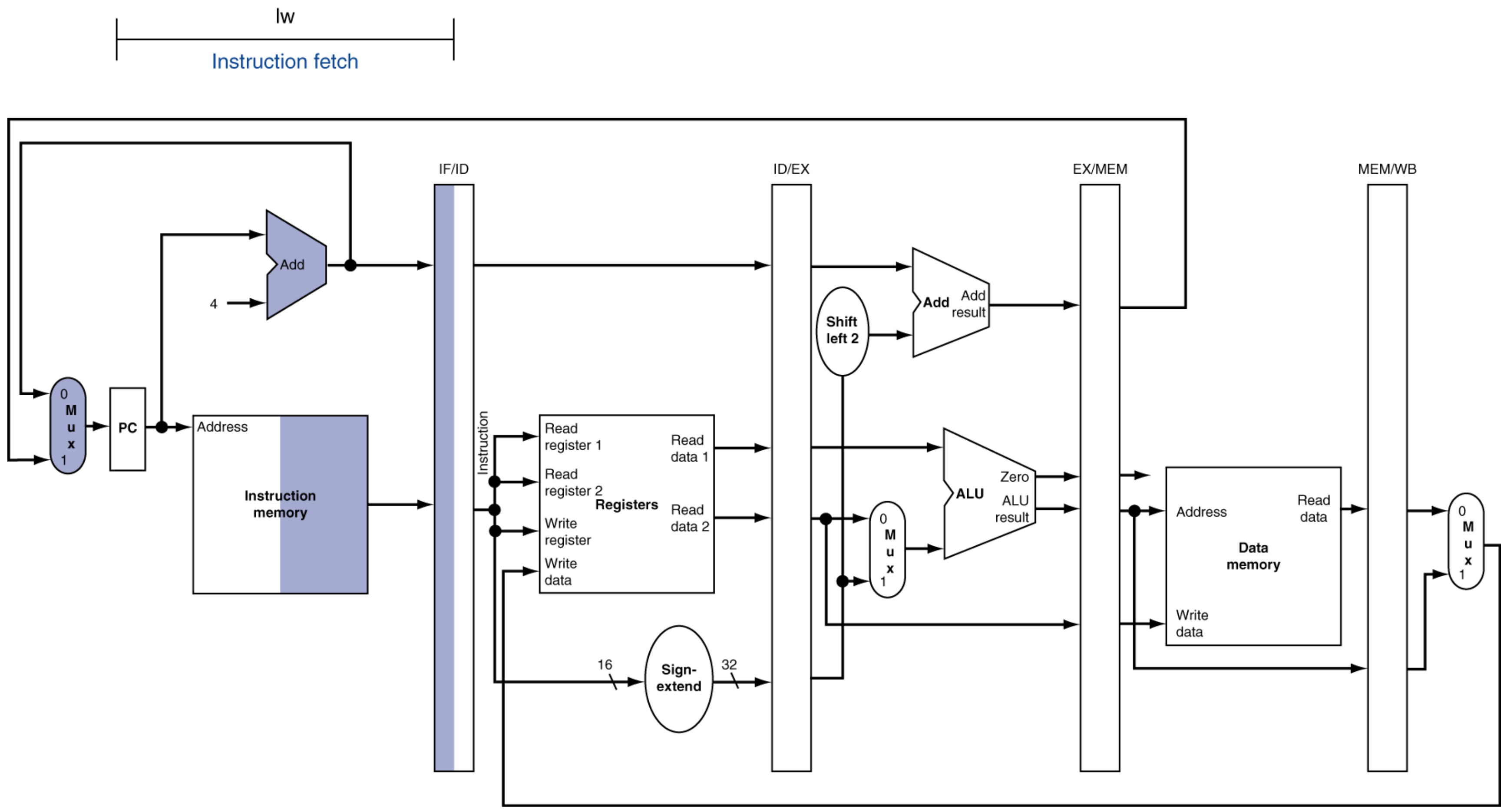
- Need registers between stages
  - To hold information produced in previous cycle



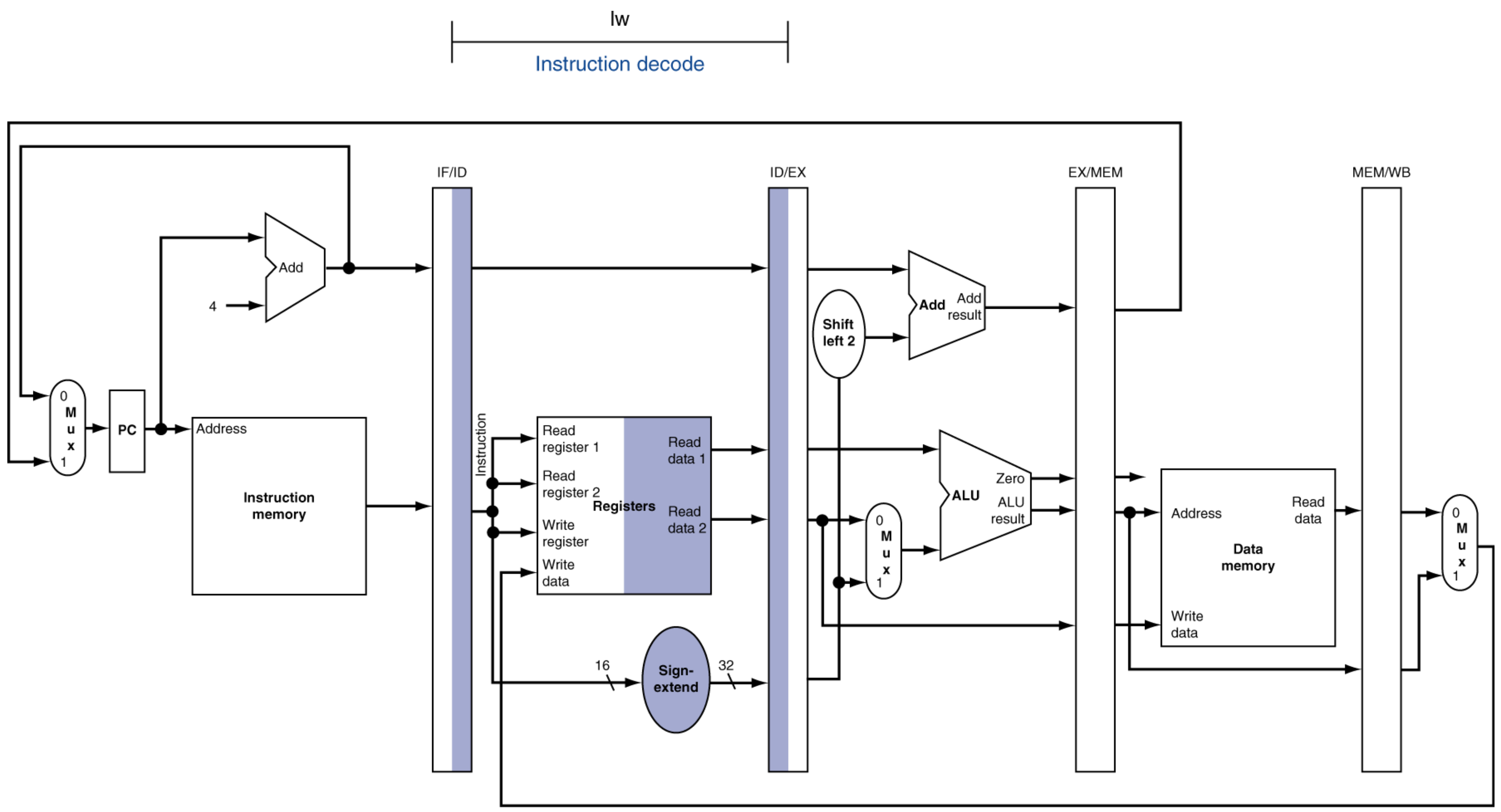
Pipeline Operation

- Cycle-by-cycle flow of instructions through the pipelined datapath
  - “Single-clock-cycle” pipeline diagram
    - Shows pipeline usage in a single cycle
    - Highlight resources used
  - c.f. “multi-clock-cycle” diagram
    - Graph of operation over time
- We’ll look at “single-clock-cycle” diagrams for load & store

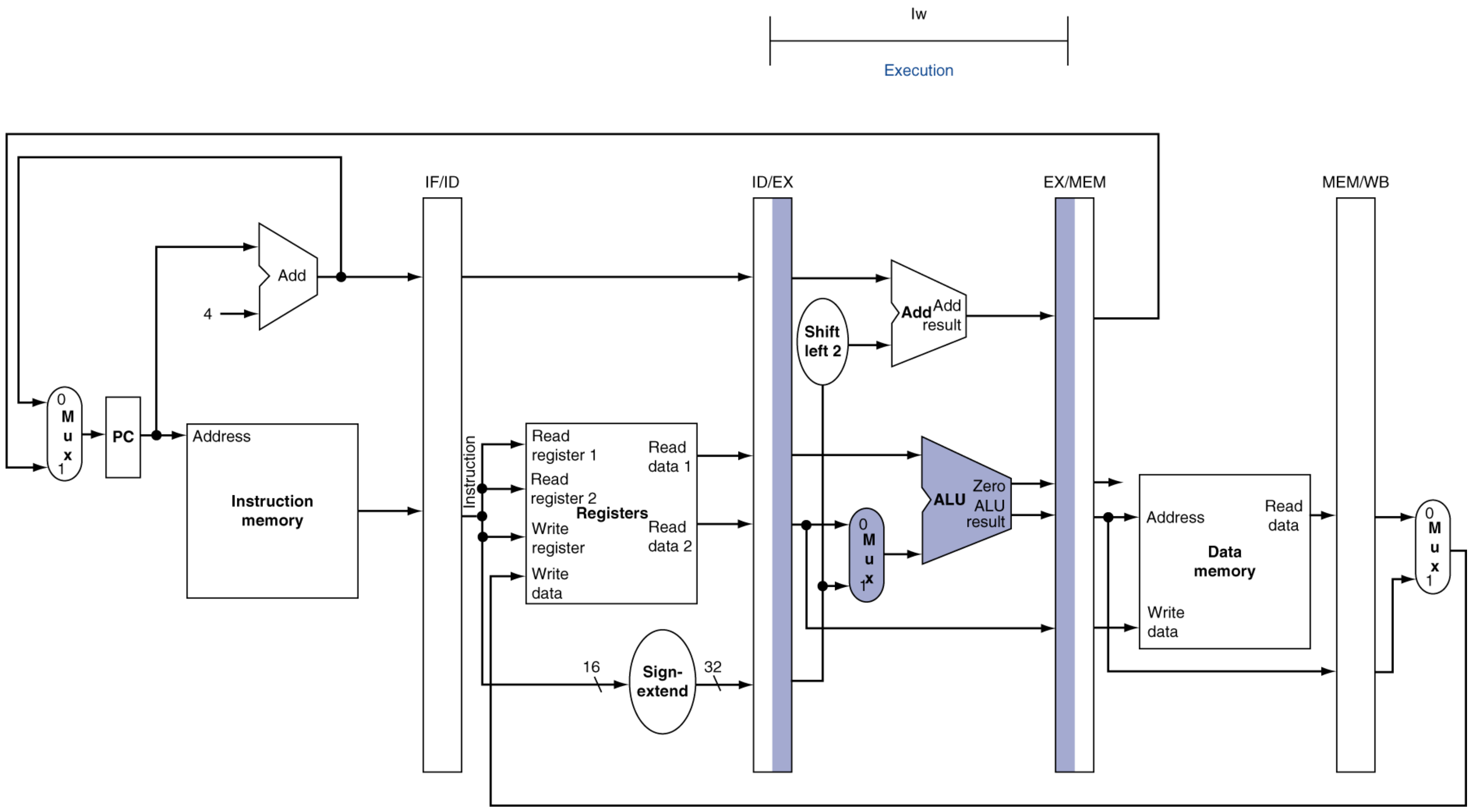
IF for Load, Store, …



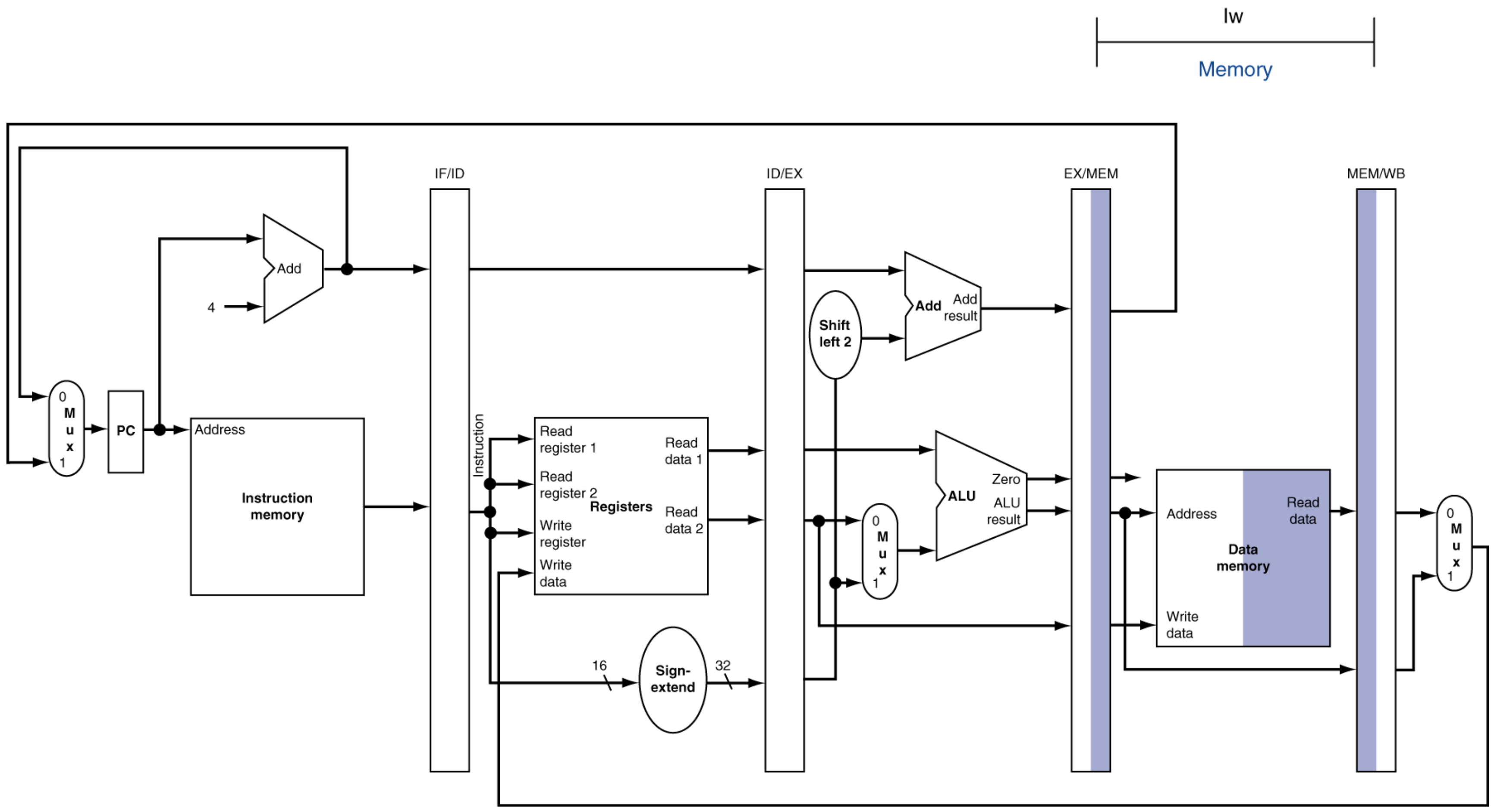
ID for Load, Store, …



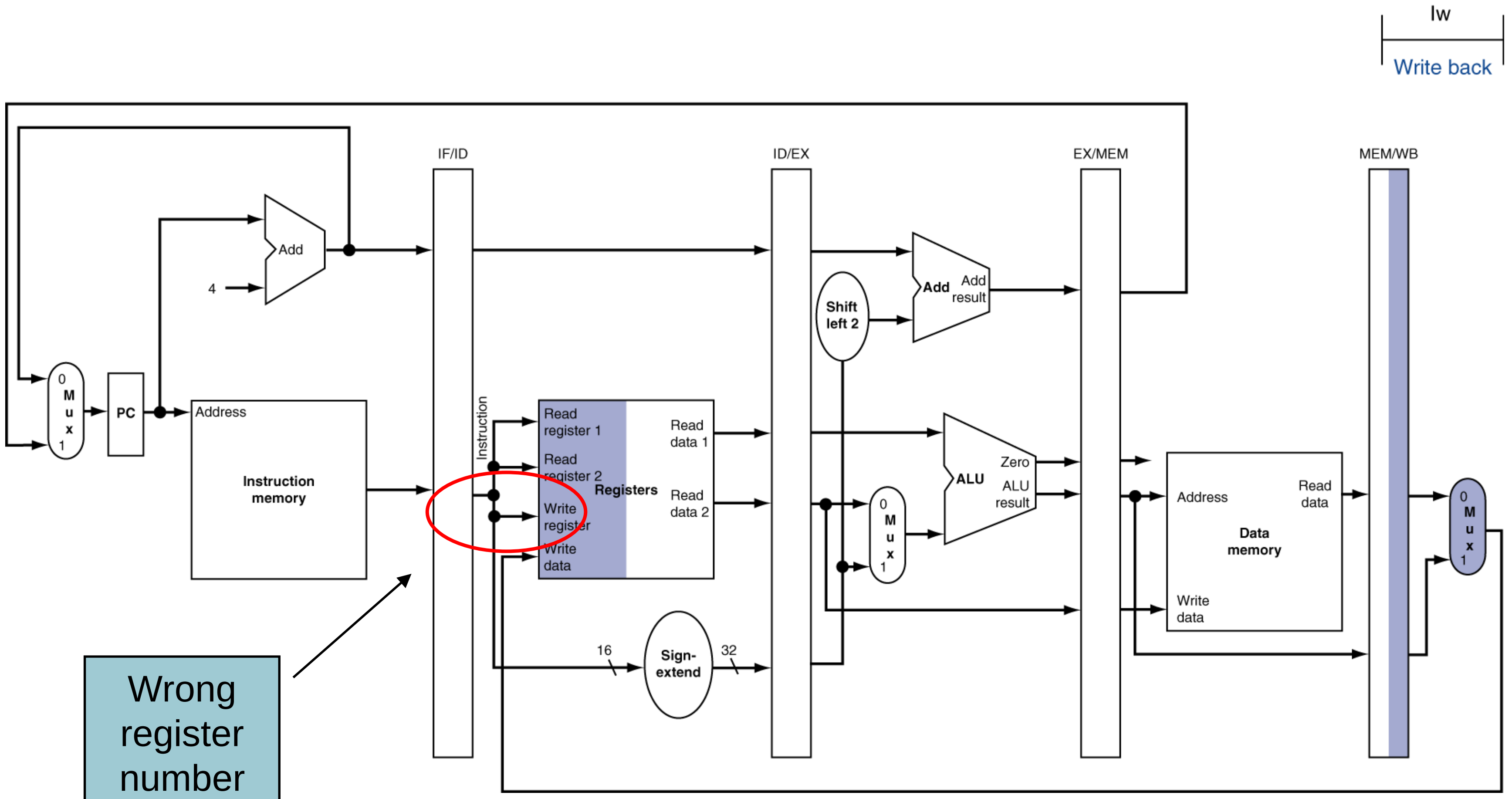
EX for Load


lw
Execution
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
0
M
u
x
1
ALU
Zero
ALU result
Address
Data memory
Read data
Write data
0
M
u
x
1

MEM for Load

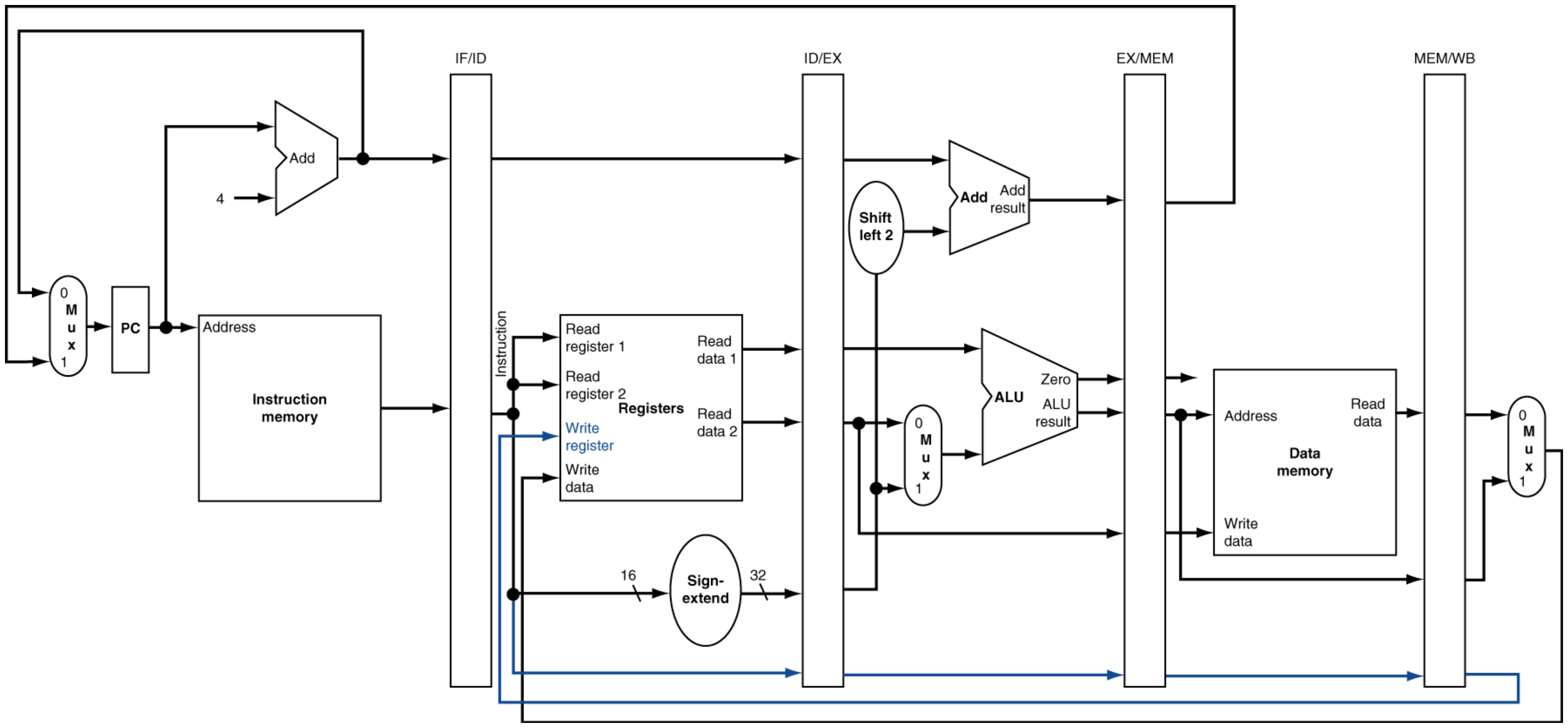

lw
Memory
IF/ID
ID/EX
EX/MEM
MEM/WB
Add
4
0
M
u
x
1
PC
Address
Instruction memory
Instruction
Read register 1
Read register 2
Registers
Write register
Write data
Read data 1
Read data 2
16
Sign-extend
32
Shift left 2
Add
Add result
ALU
Zero
ALU result
Address
Data memory
Write data
Read data

WB for Load

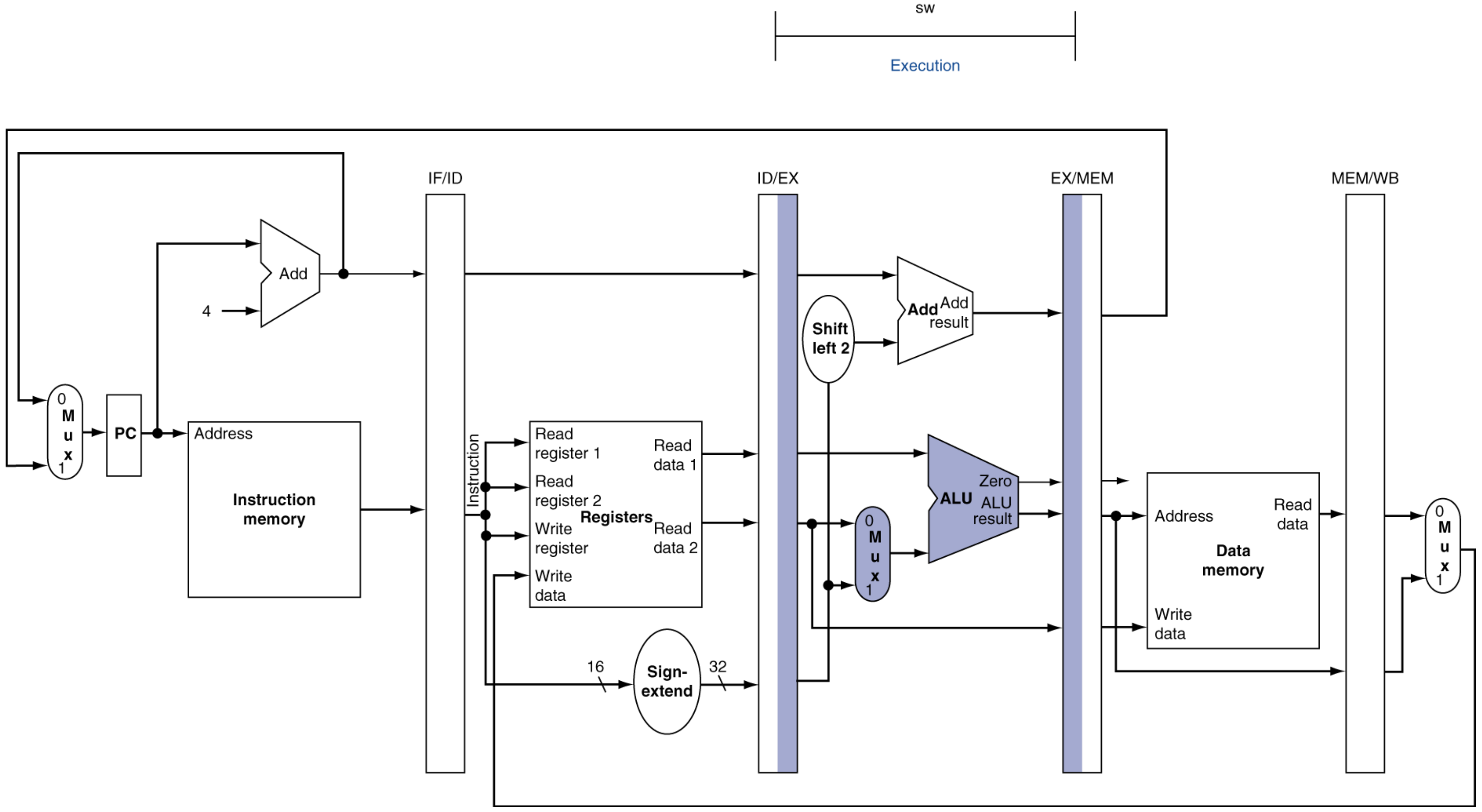


Wrong register number

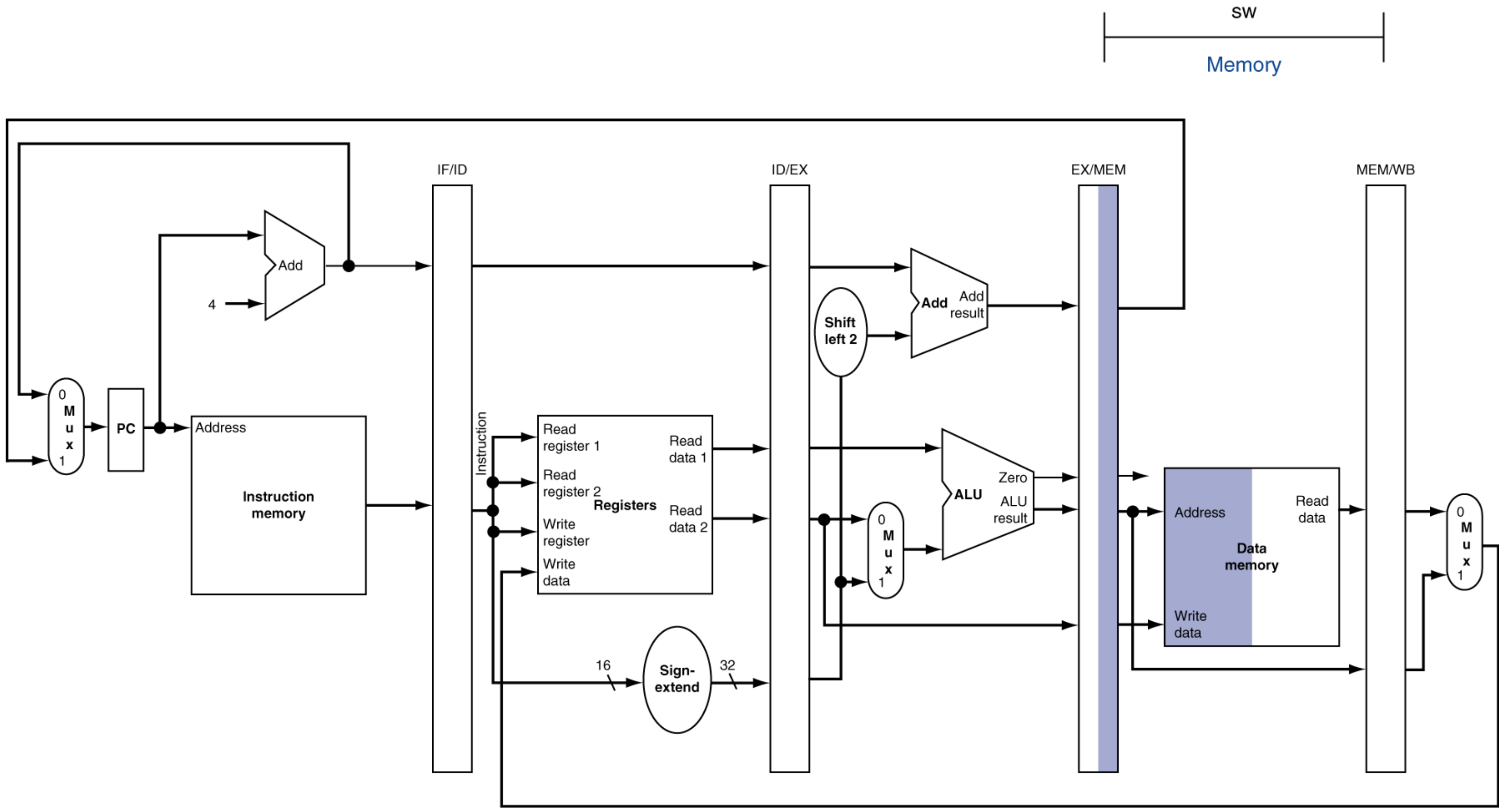
Corrected Datapath for Load



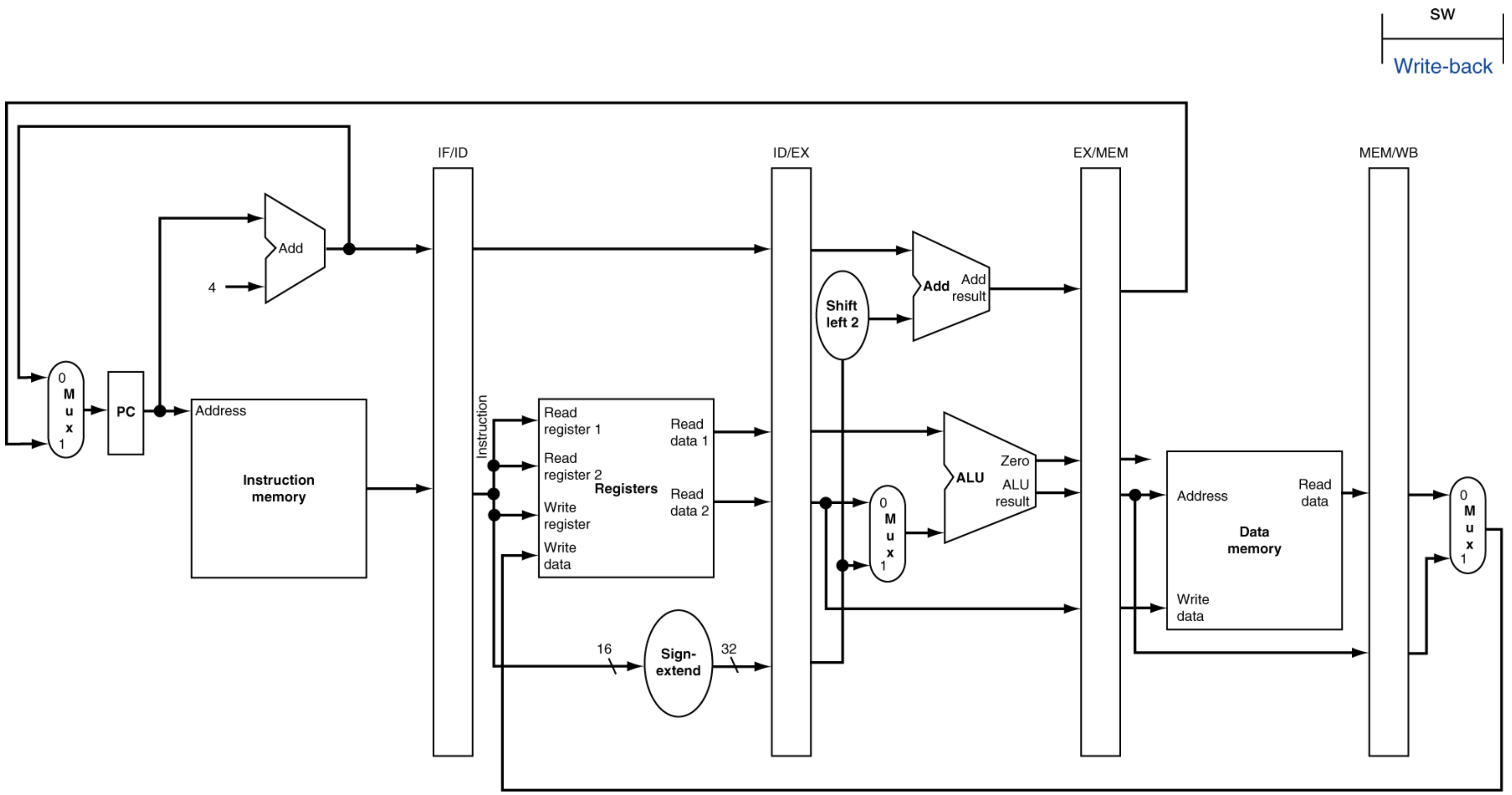
EX for Store


sw
Execution
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
Write register
Write data
Registers
Read data 1
Read data 2
16
Sign-extend
32
Shift left 2
Add
Add result
0
M
u
x
1
ALU
Zero
ALU result
Address
Data memory
Write data
Read data
0
M
u
x
1

MEM for Store



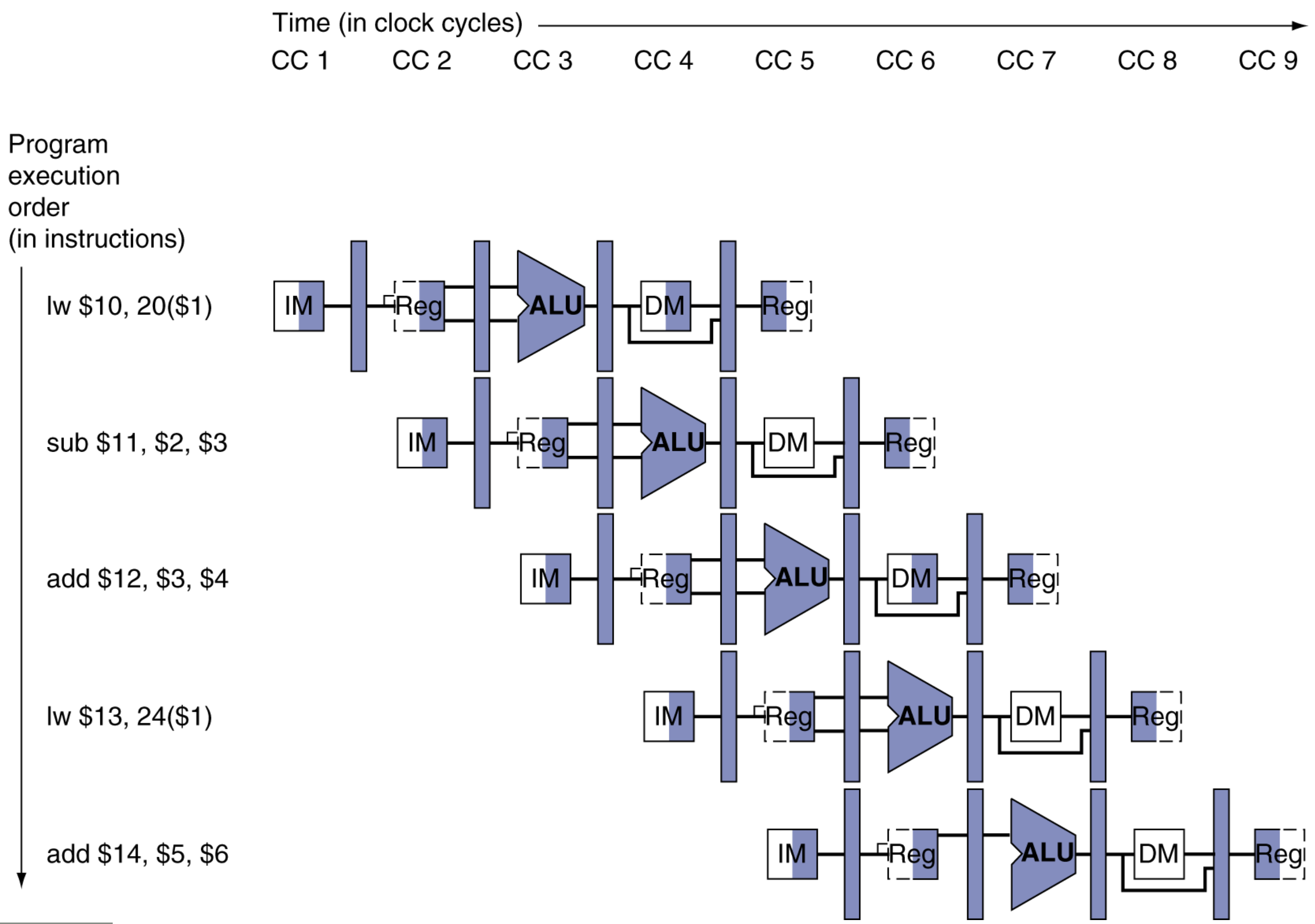
WB for Store



sw
Write-back
IF/ID
ID/EX
EX/MEM
MEM/WB
Add
4
Add
Add result
Shift left 2
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
0 M u x 1
ALU
Zero
ALU result
Address
Data memory
Write data
Read data
0 M u x 1

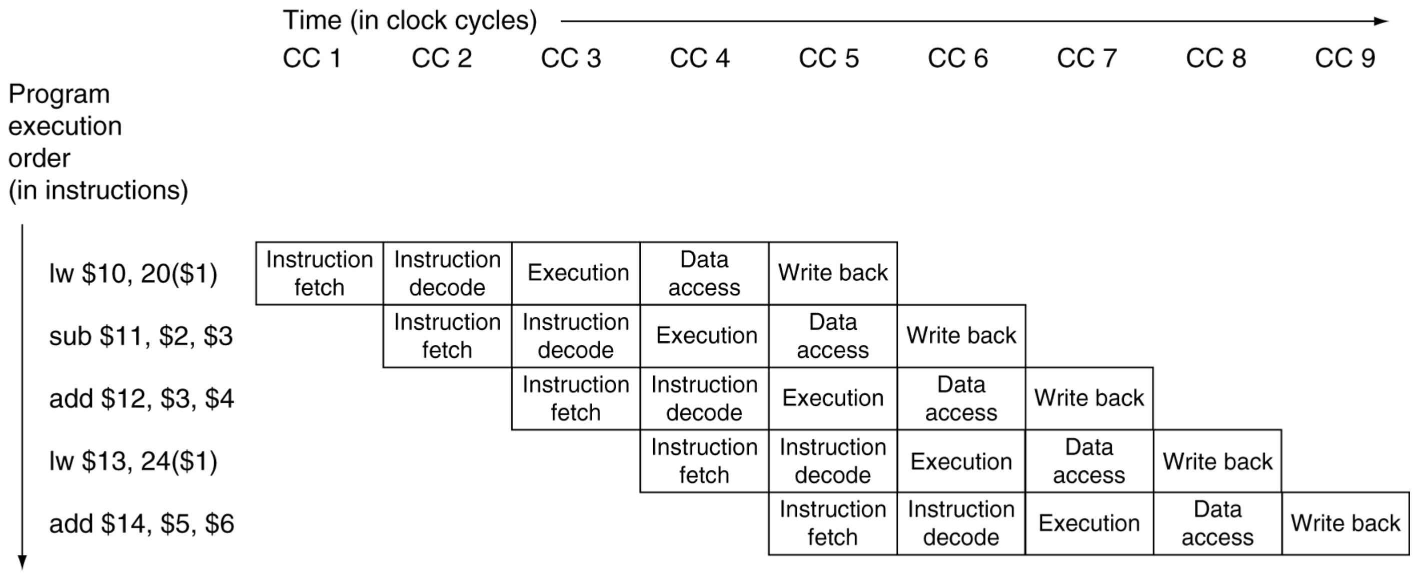
Multi-Cycle Pipeline Diagram

- Form showing resource usage



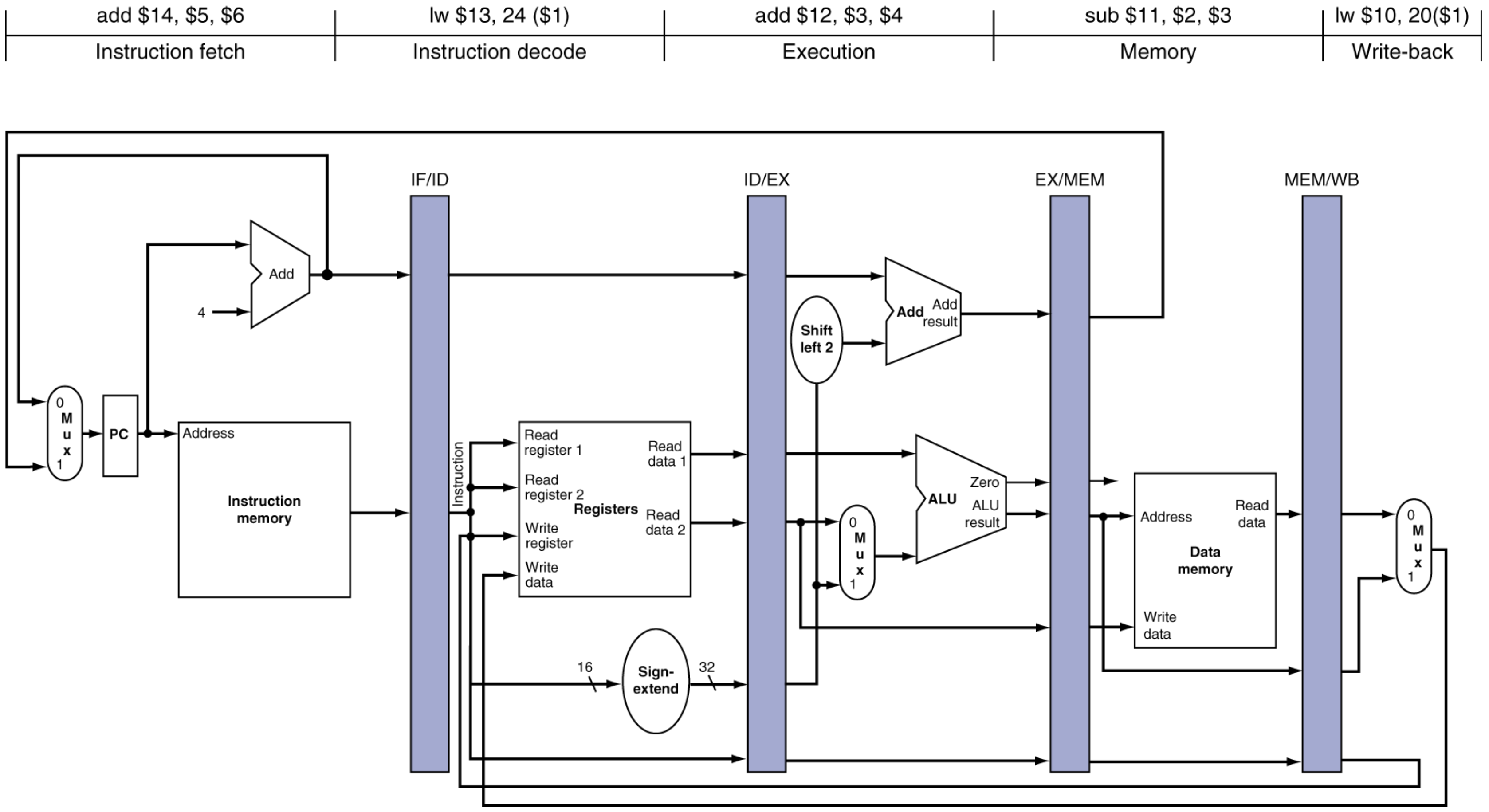
Multi-Cycle Pipeline Diagram

- Traditional form

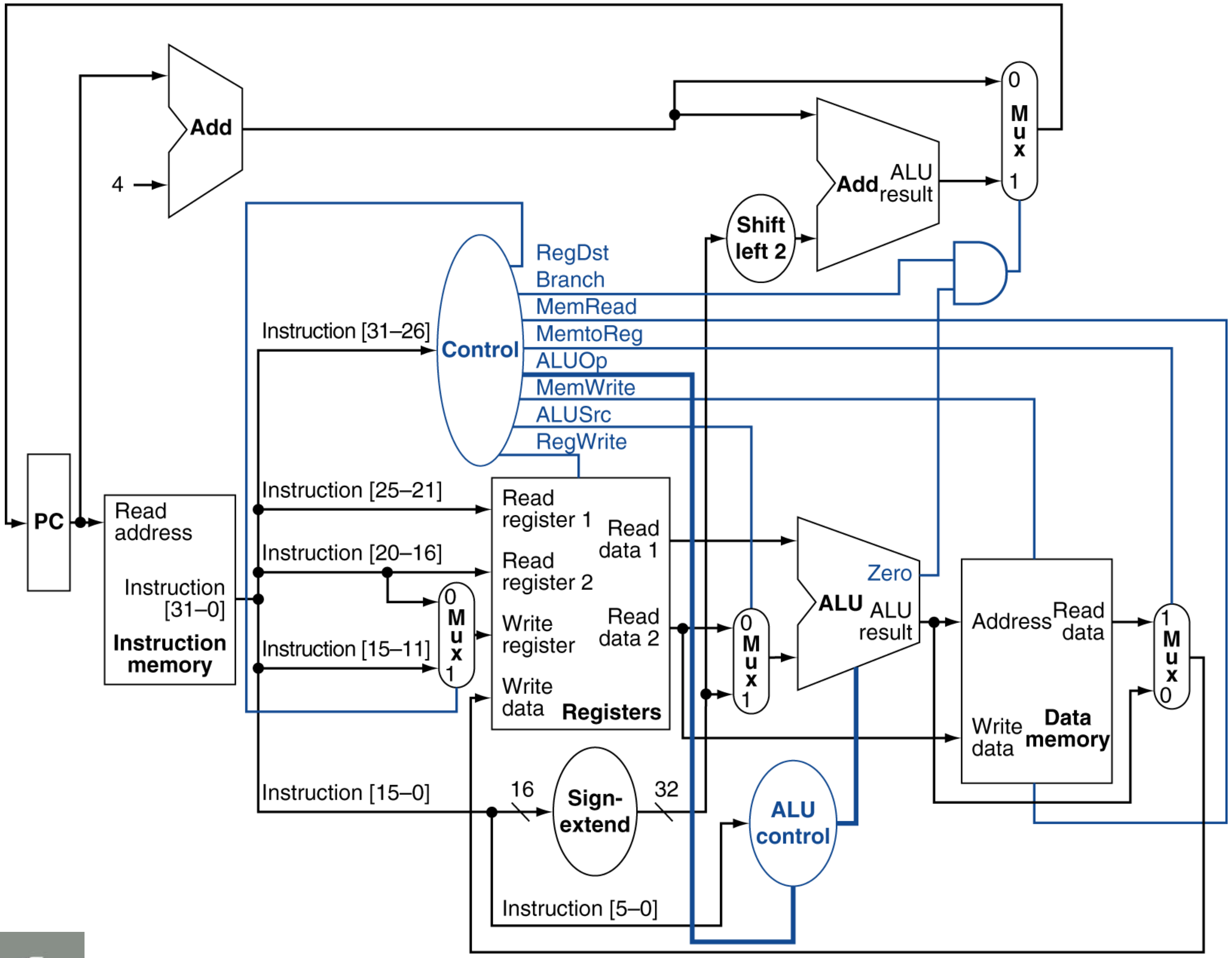


Single-Cycle Pipeline Diagram

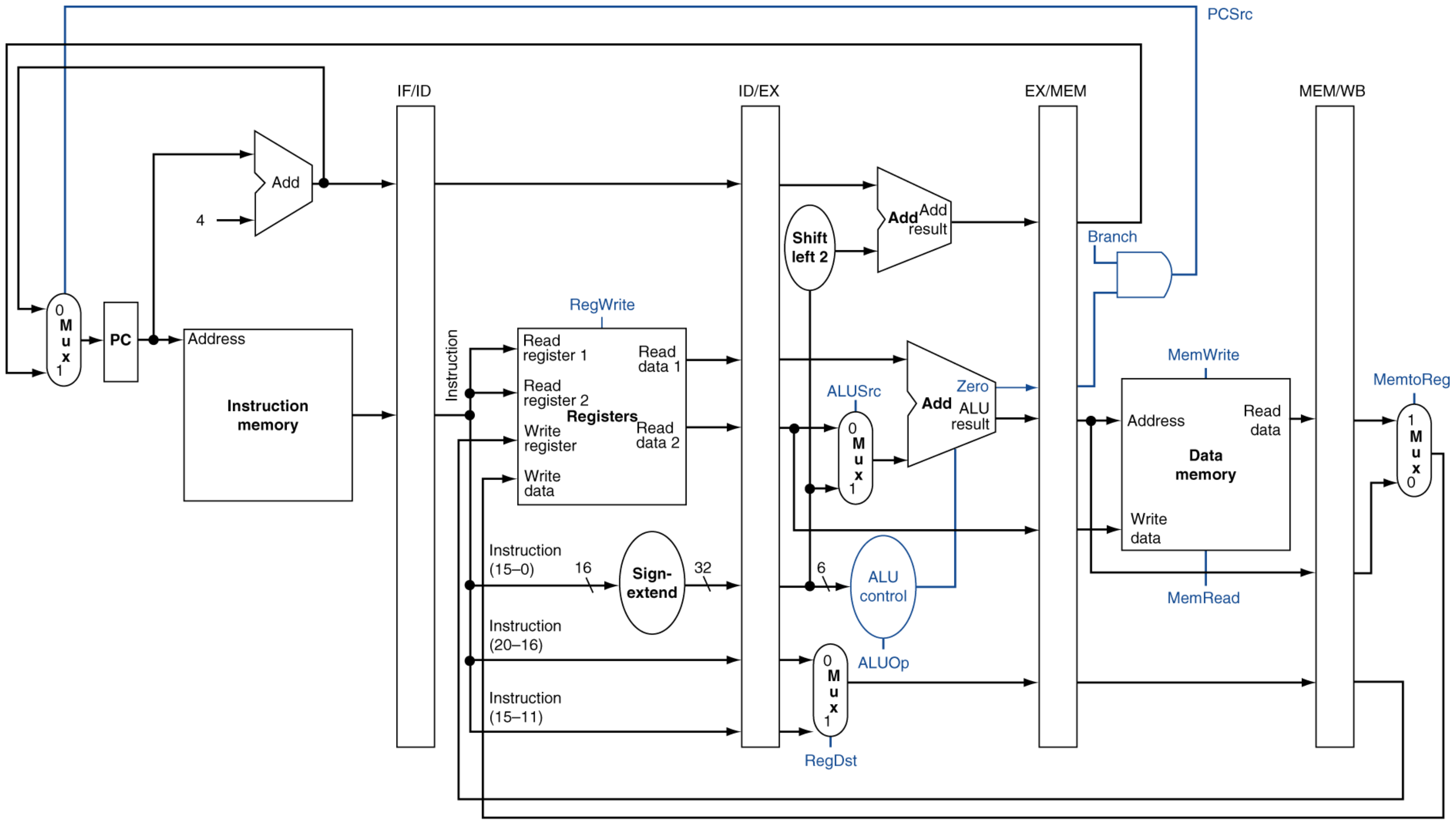
- State of pipeline in a given cycle



Datapath With Control

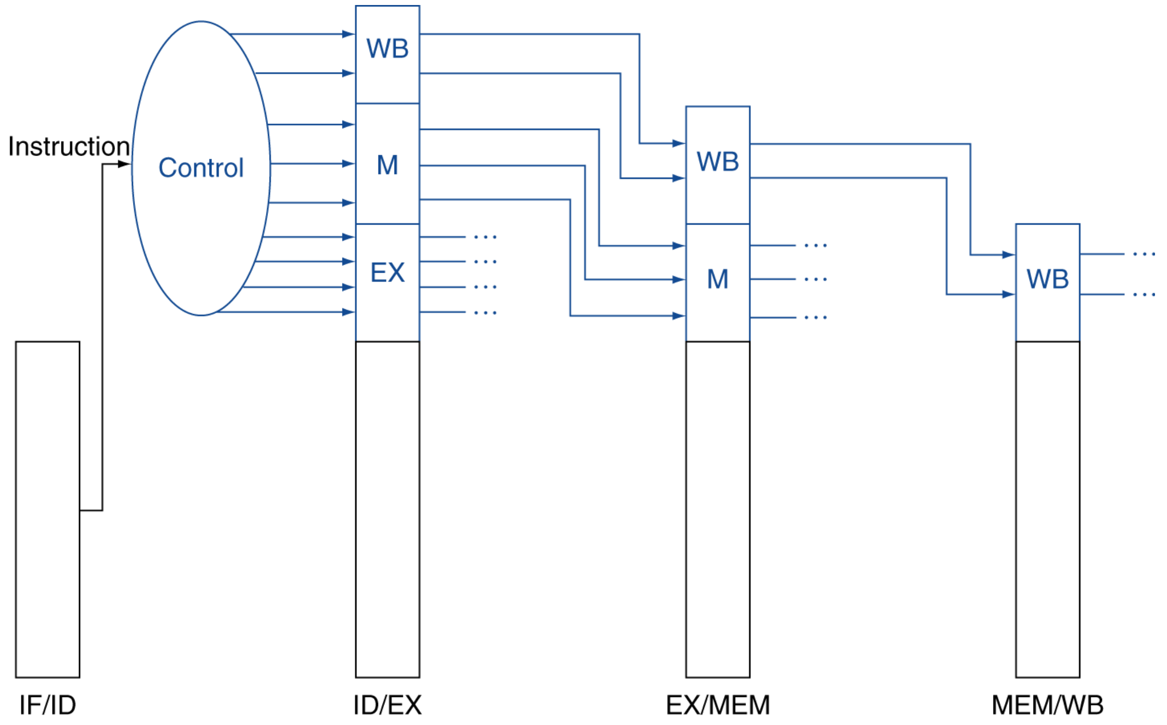


Pipelined Control (Simplified)

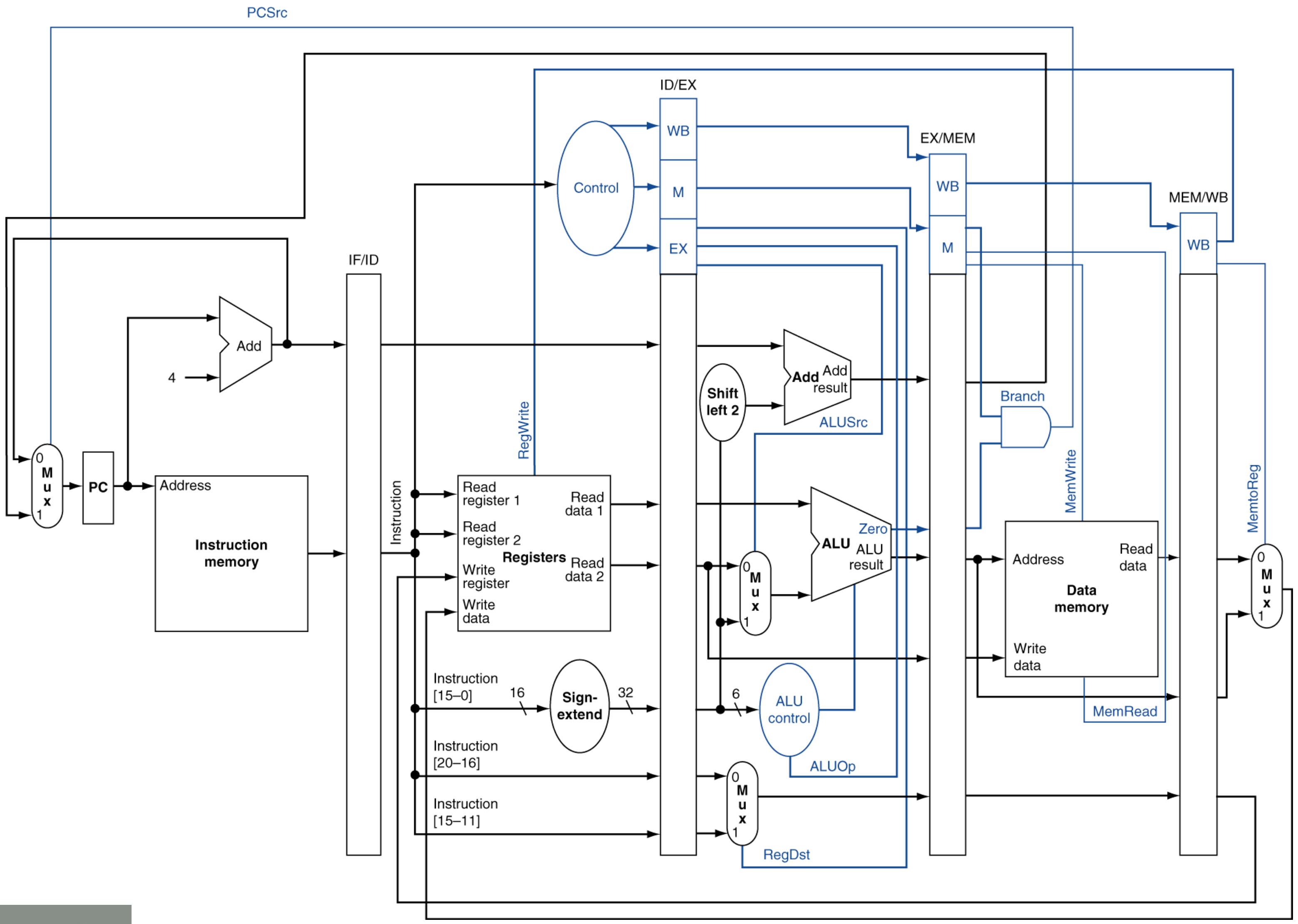


Pipelined Control

- Control signals derived from instruction
  - As in single-cycle implementation



Pipelined Control



Concluding Remarks

- ISA influences design of datapath and control
- Datapath and control influence design of ISA
- Pipelining improves instruction throughput using parallelism
  - More instructions completed per second
  - Latency for each instruction not reduced

Problems to solve

- 4.8.1, 4.8.2, 4.8.3, 4.8.6