INTRODUCTION TO PIPELINING

Ayman Hajja, PhD

SINGLE-CYCLE PERFORMANCE

- Assume time for actions are
  - 100ps for register read or write, 200 ps for other events
- Clock period is:

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

- Clock rate (cycles/second = Hz) = 1/Period (seconds/cycle)

GOTTA DO LAUNDRY

- Washer takes 30 minutes
- Dryer takes 30 minutes
- Folding takes 30 minutes

GOTTA DO LAUNDRY

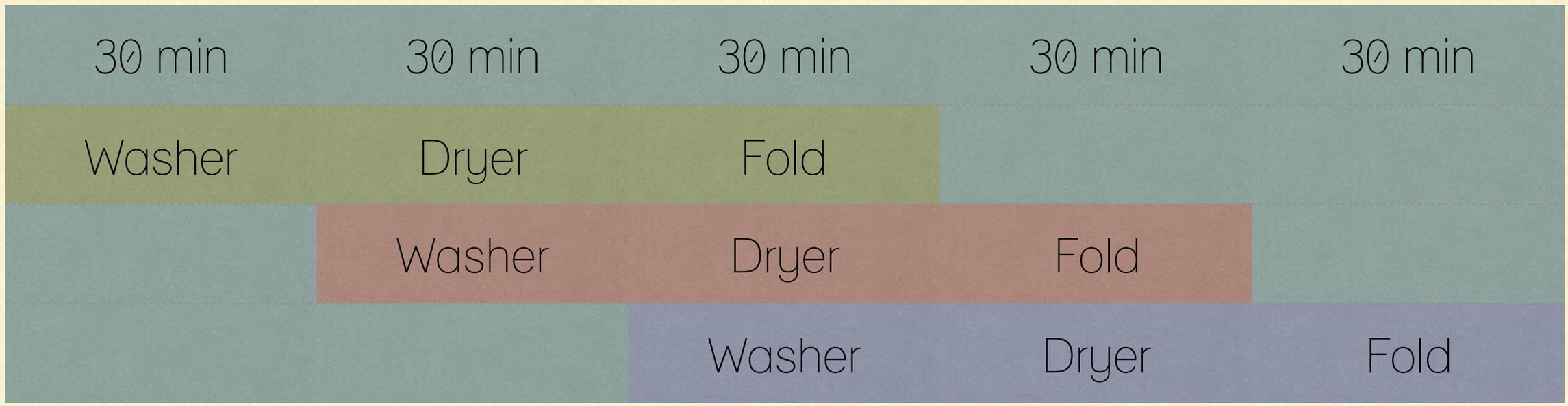
- Washer takes 30 minutes
- Dryer takes 30 minutes
- Folding takes 30 minutes

30 min	30 min	30 min	30 min	30 min	30 min	30 min	30 min	30 min
Washer	Dryer	Fold	Wash	Dryer	Fold	Wash	Dryer	Fold

- Total 4.5 hours

GOTTA DO LAUNDRY

- Washer takes 30 minutes
- Dryer takes 30 minutes
- Folding takes 30 minutes



- Pipeline laundry takes 2.5 hours

GOTTA DO LAUNDRY

- Pipelining doesn't help latency of single task, it helps throughput of overall workload
- Multiple tasks operate simultaneously using different resources
- Potential speedup = Number pipe stages
- Time to "fill" pipeline and time to "drain" it reduces speedup

GOTTA DO LAUNDRY

- Suppose new dryer takes 20 minutes. How much faster is pipeline?

GOTTA DO LAUNDRY

- Suppose new dryer takes 20 minutes. How much faster is pipeline?

- Pipeline rate limited by slowest pipeline stage
- Unbalanced lengths of pipe stages reduces speedup

STEPS IN EXECUTING MIPS

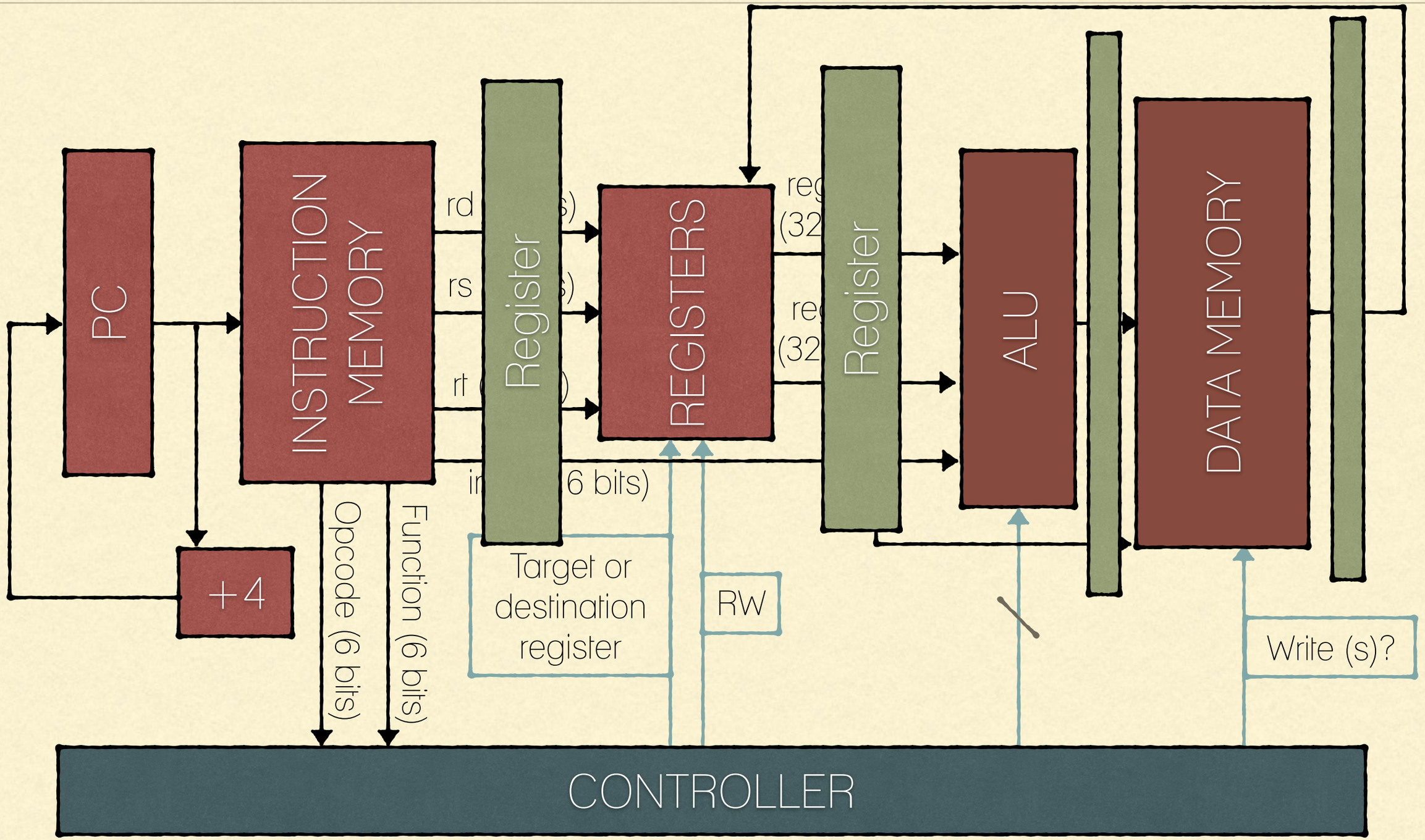
1. IFtch: Instruction Fetch, Increment PC
2. Dcd: Instruction Decode, Read Registers
3. Exec:
    1. Mem-ref: Calculate Address
    2. Arith-log: Perform Operation
4. Mem:
    1. Load: Read Data from Memory
    2. Store: Write Data to Memory
5. WB: Write Data Back to Register

STEPS IN EXECUTING MIPS

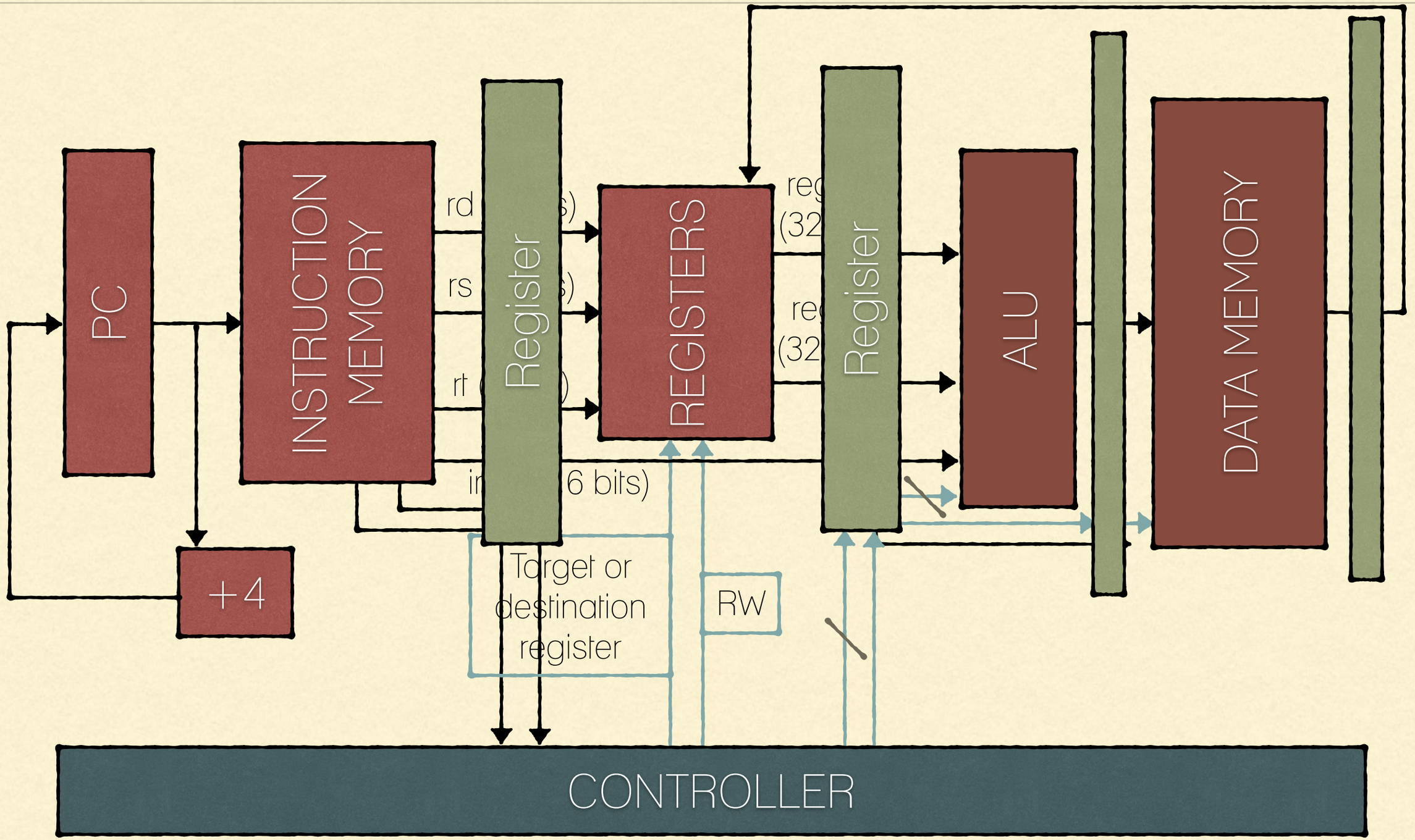
IF	ID	EX	MEM	WB				
	IF	ID	EX	MEM	WB			
		IF	ID	EX	MEM	WB		
			IF	ID	EX	MEM	WB	
				IF	ID	EX	MEM	WB

- Every instruction must take same number of steps, so some stages will be idle
  - e.g. MEM stage for any arithmetic instruction

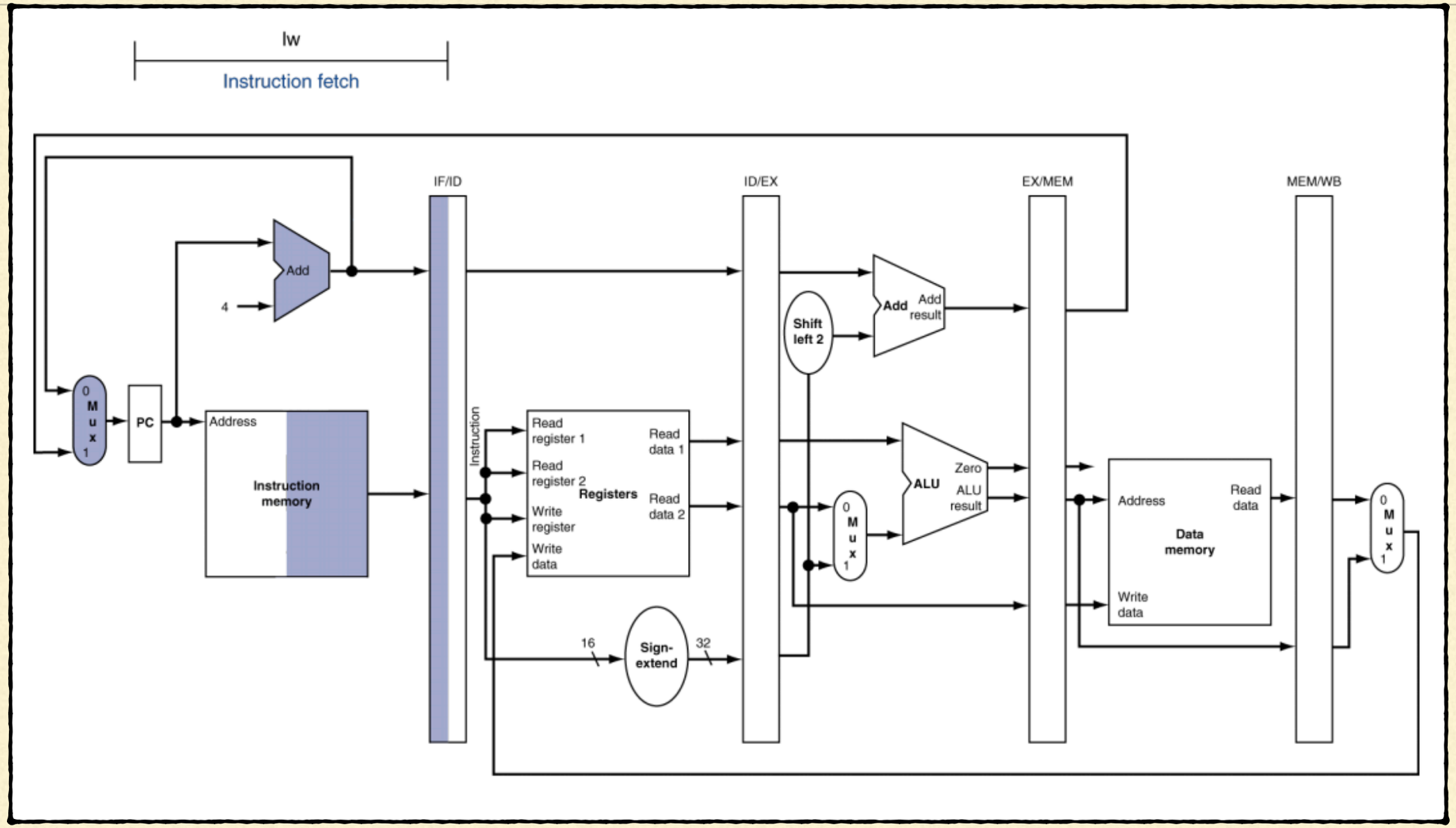
WHAT'S THE PROBLEM HERE?



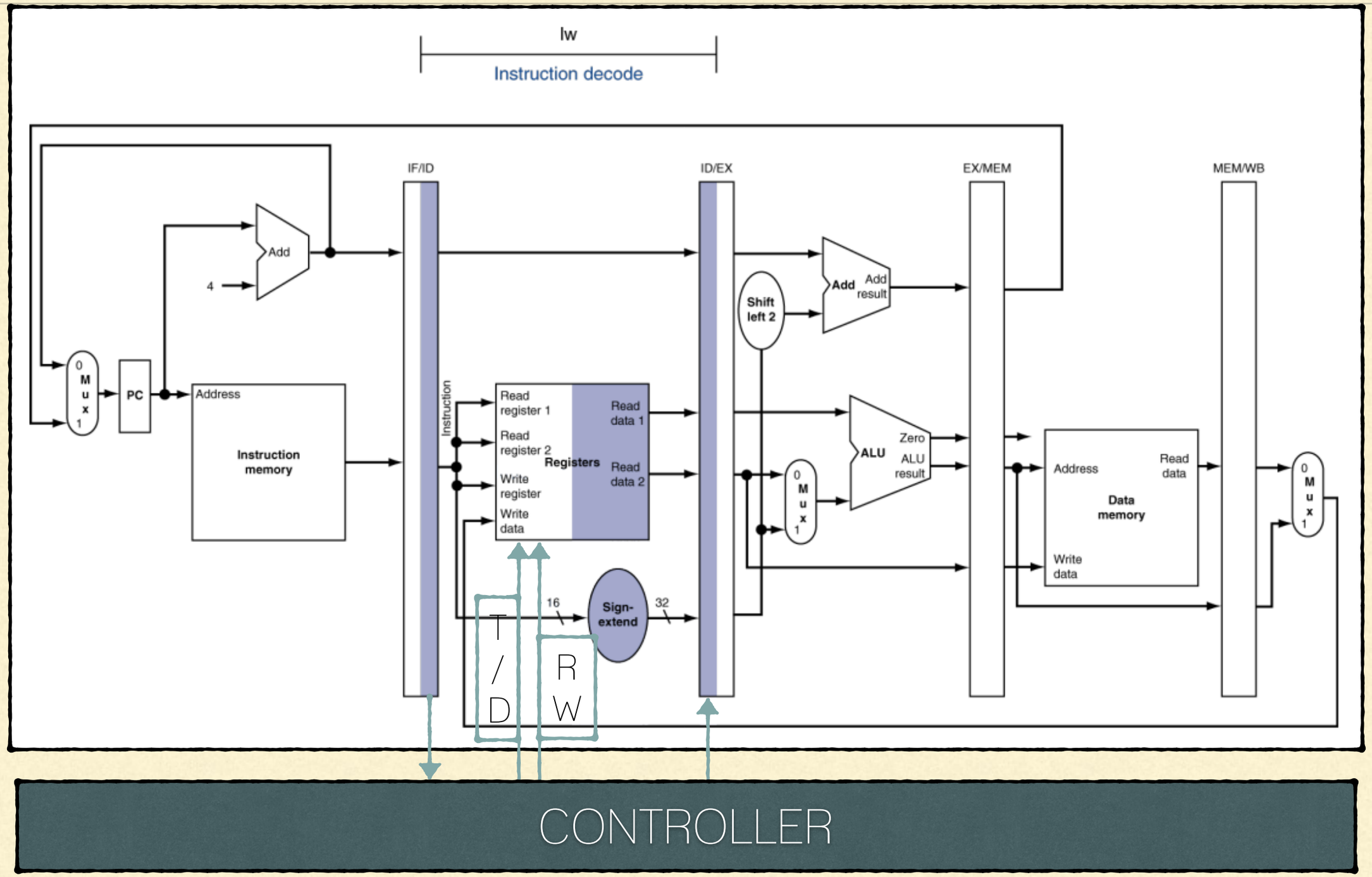
STAGES OF EXECUTION ON PIPELINED DATAPATH



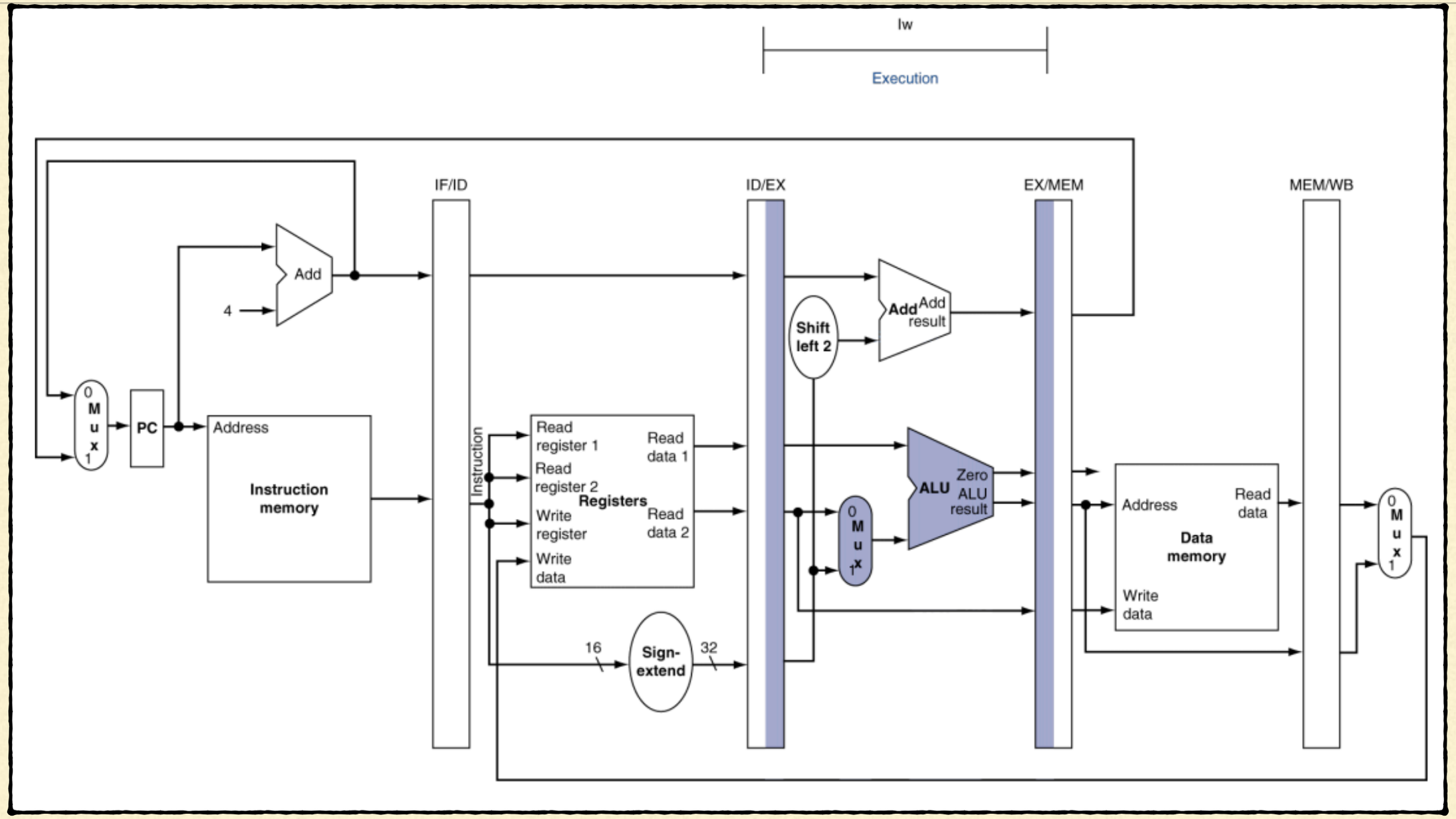
INSTRUCTION FETCH FOR 'LOAD WORD'



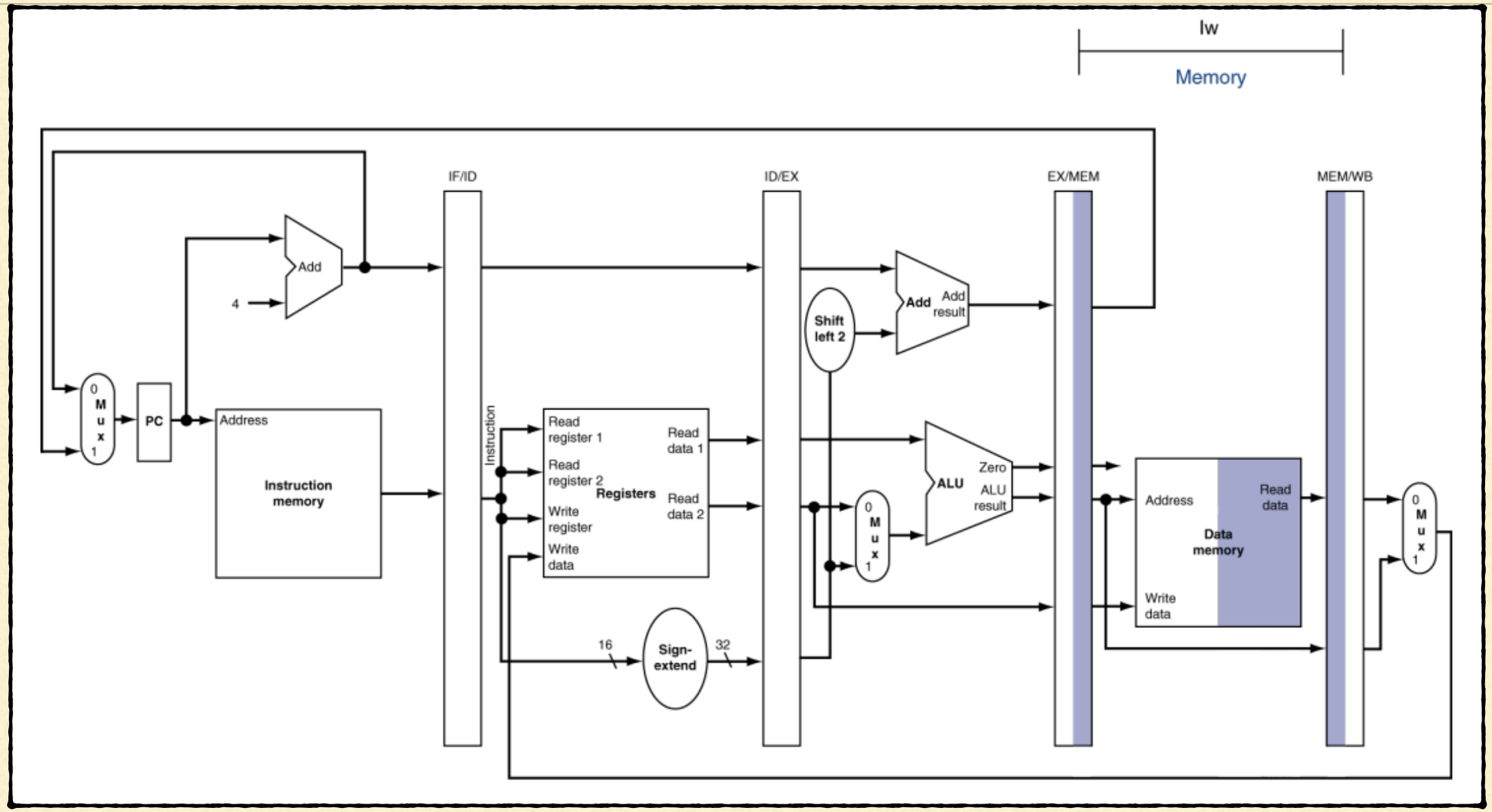
INSTRUCTION DECODE FOR 'LOAD WORD'



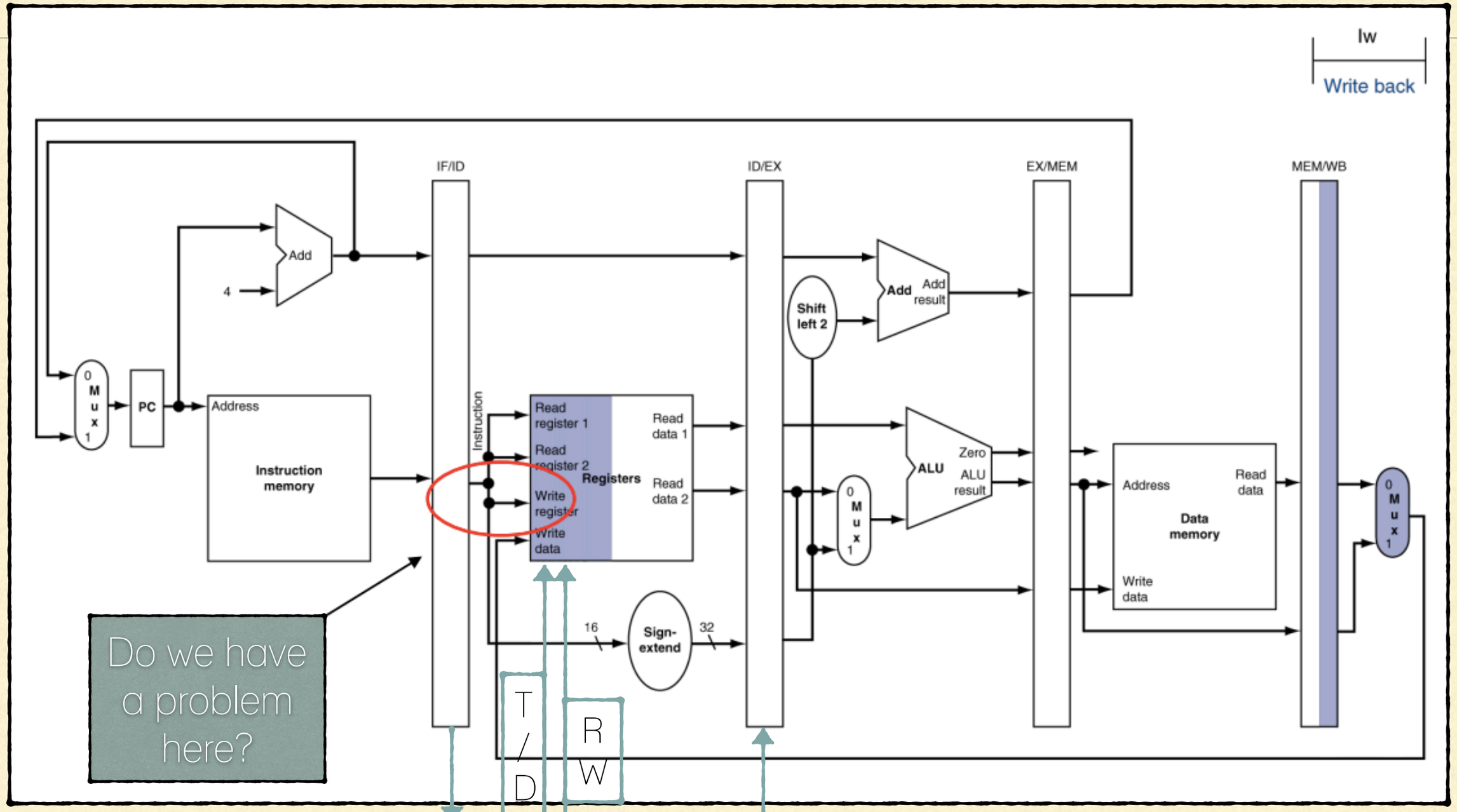
ALU FOR 'LOAD WORD'



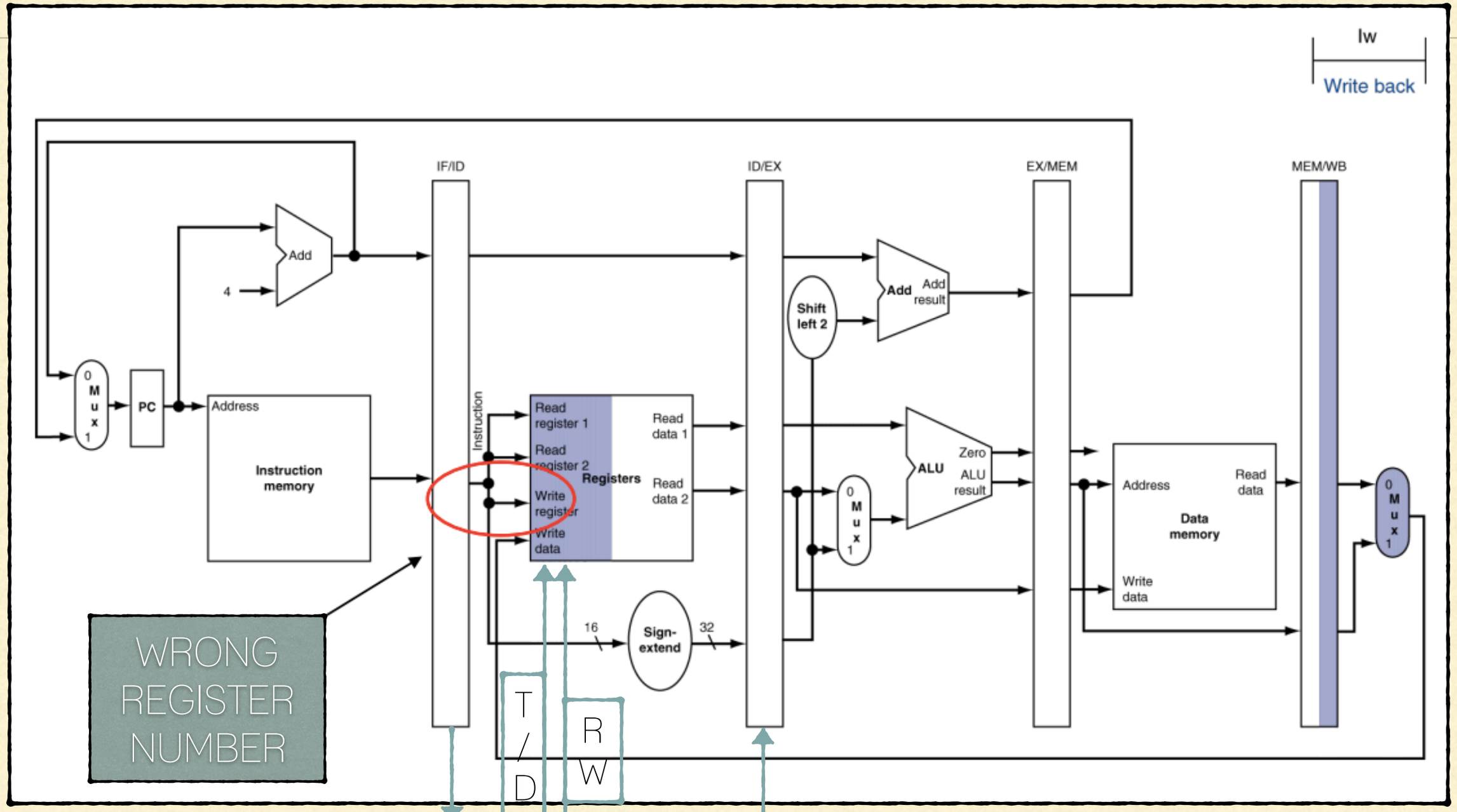
MEM FOR 'LOAD WORD'



WB FOR 'LOAD WORD'

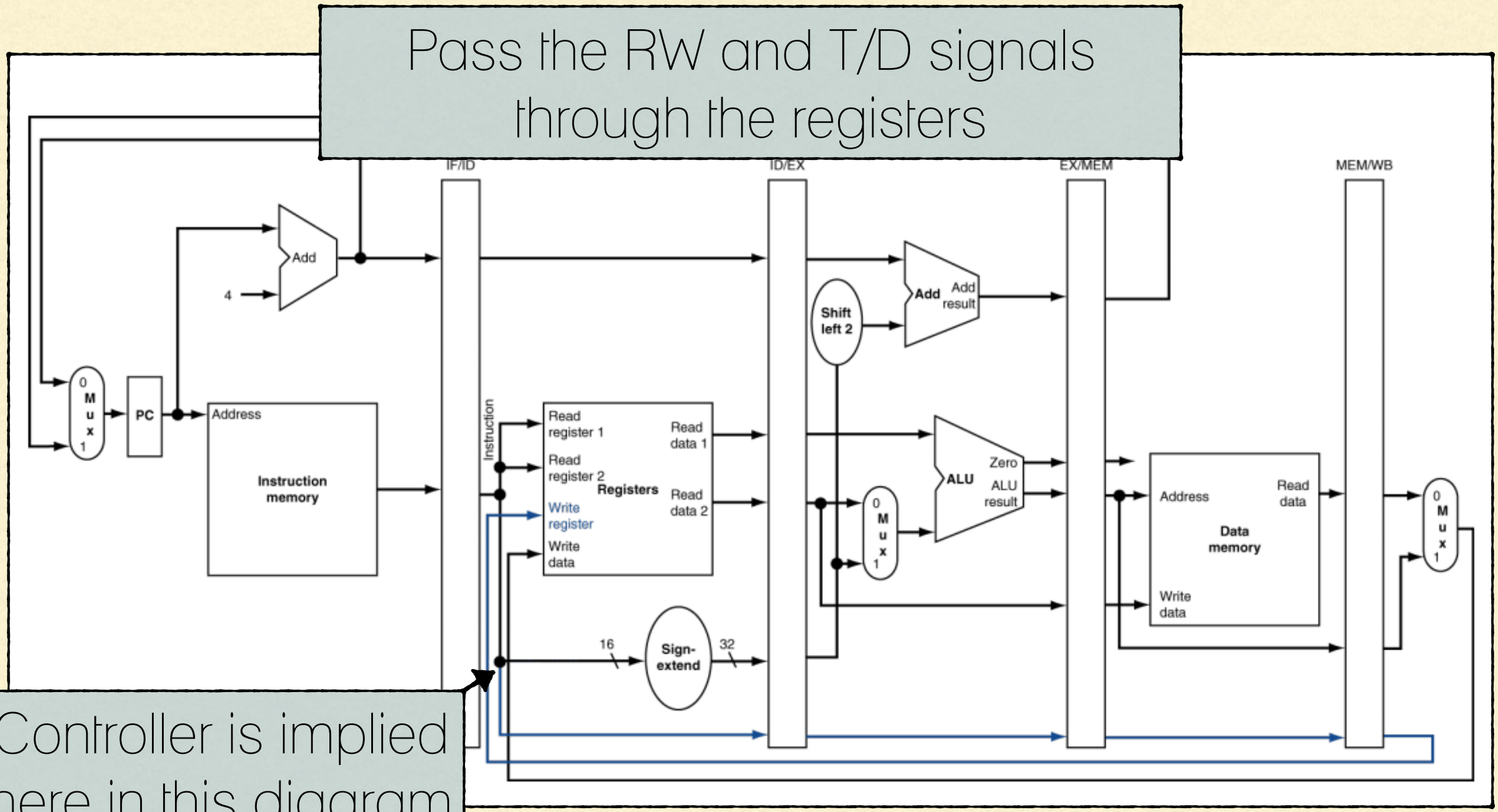


WB FOR 'LOAD WORD'



CORRECTED DATAPATH FOR LOAD

Pass the RW and T/D signals through the registers



Controller is implied here in this diagram

PIPELINE PERFORMANCE

- Use T_C ("Time between completion of instruction") to measure speedup:
- $T_{c,\ pipelined} \geq$ to $T_{c,\ single\text{-}cycle}$/ Num of stages
- Equality only achieved if stages are balanced (i.e. take the same amount of time)
- If not balanced, speed up is reduced
- Speedup due to increased throughput

SINGLE-CYCLE PERFORMANCE

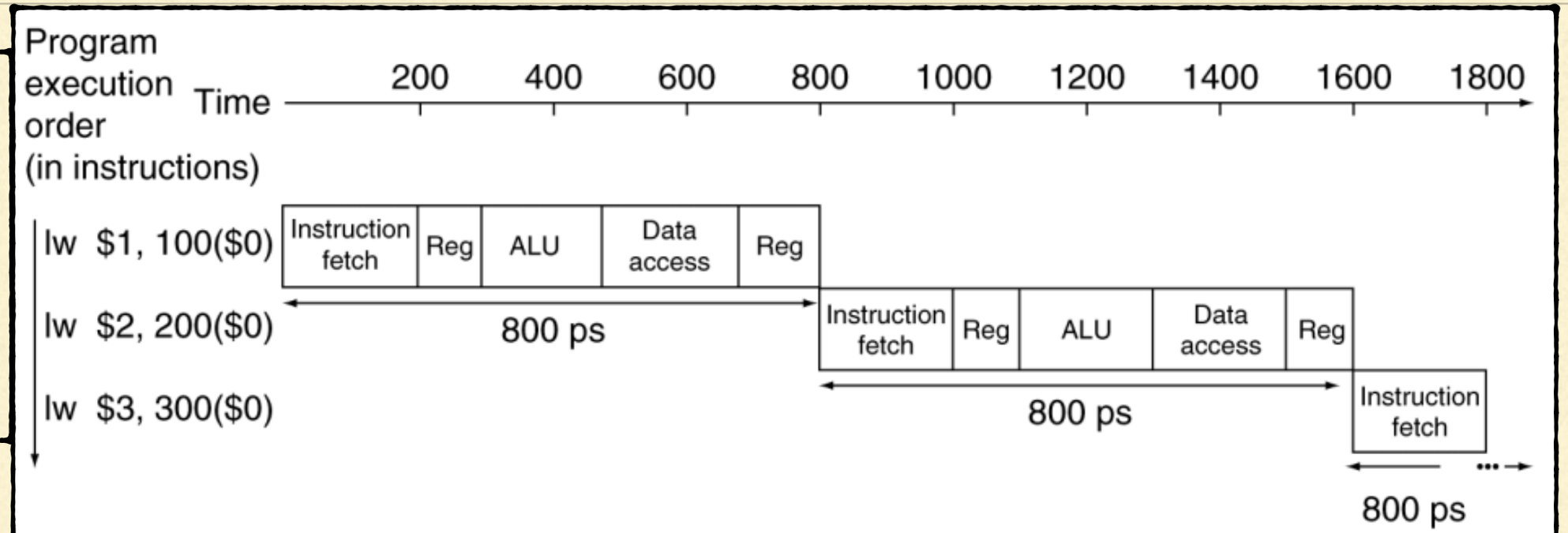
- Assume time for actions are
  - 100ps for register read or write, 200 ps for other events
- Clock period is:

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

- Clock rate (cycles/second = Hz) = 1/Period (seconds/cycle)

SINGLE-CYCLE PERFORMANCE VS PIPELINED PERFORMANCE

Single-cycle
T_c = 800 ps
F = 1,25GHz



Pipelined
T_c = 200 ps
F = 5GHz