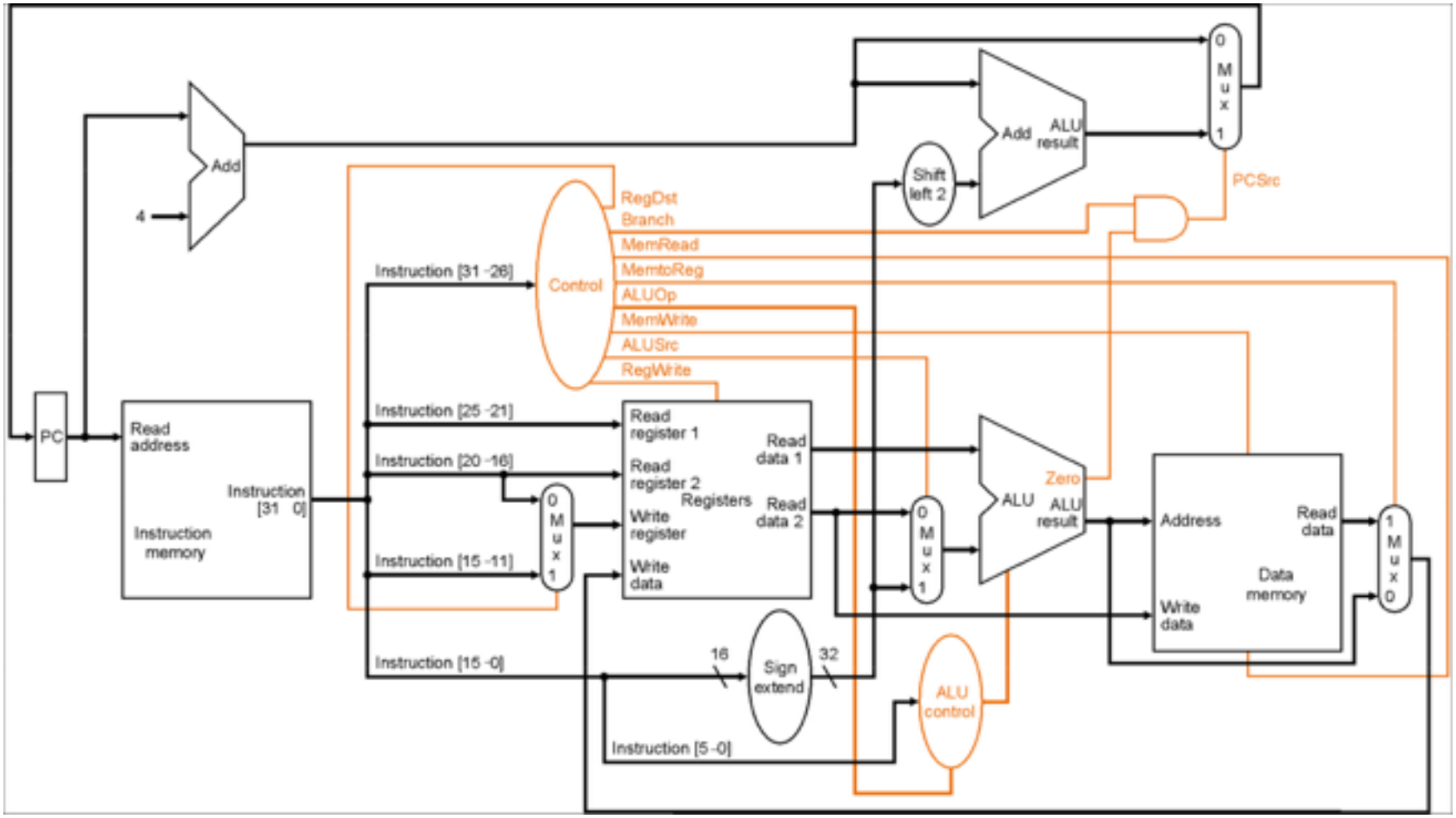
# Designing a Pipelined CPU

# Review -- Single Cycle CPU

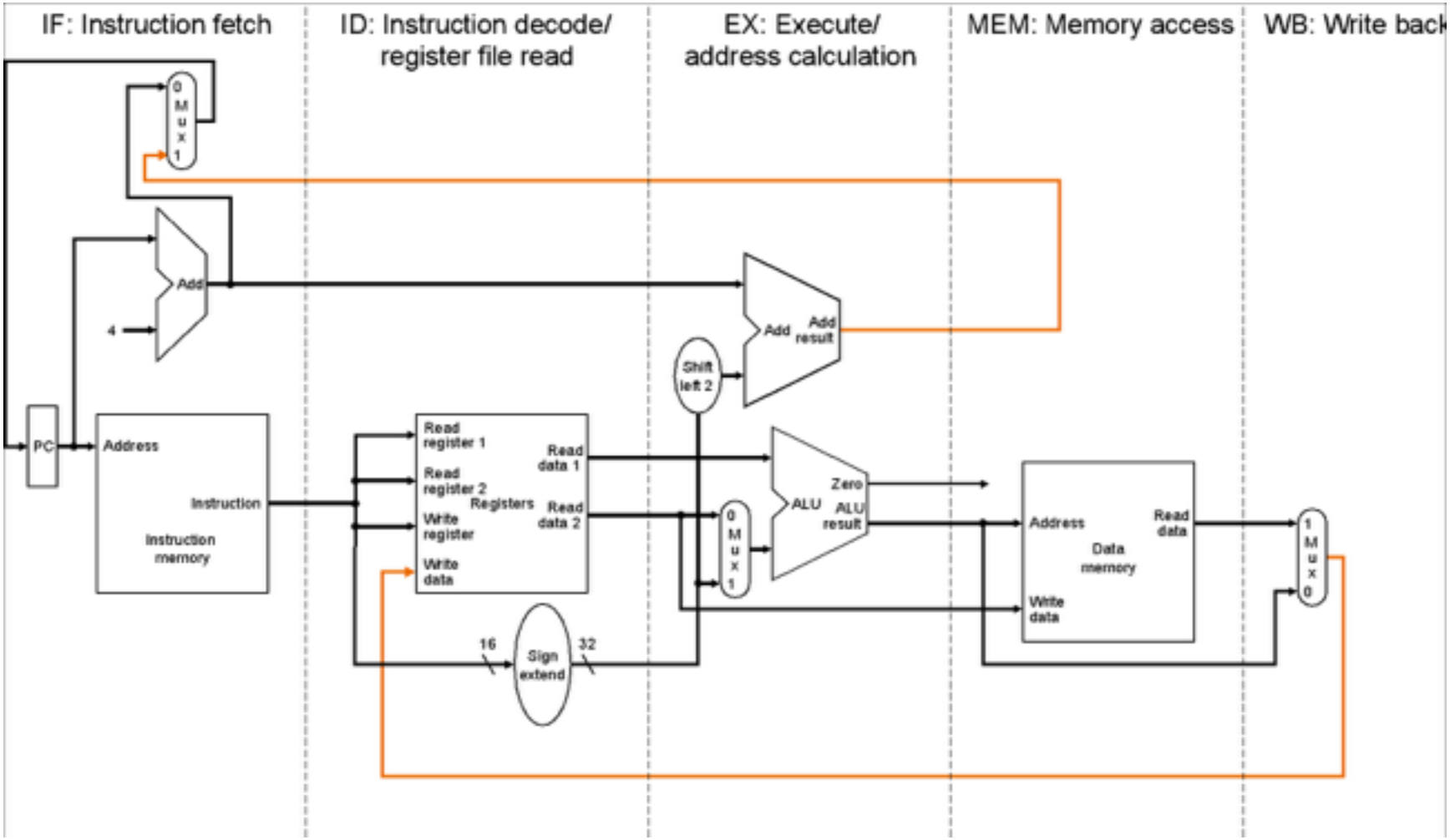


# Instruction Latencies and Throughput

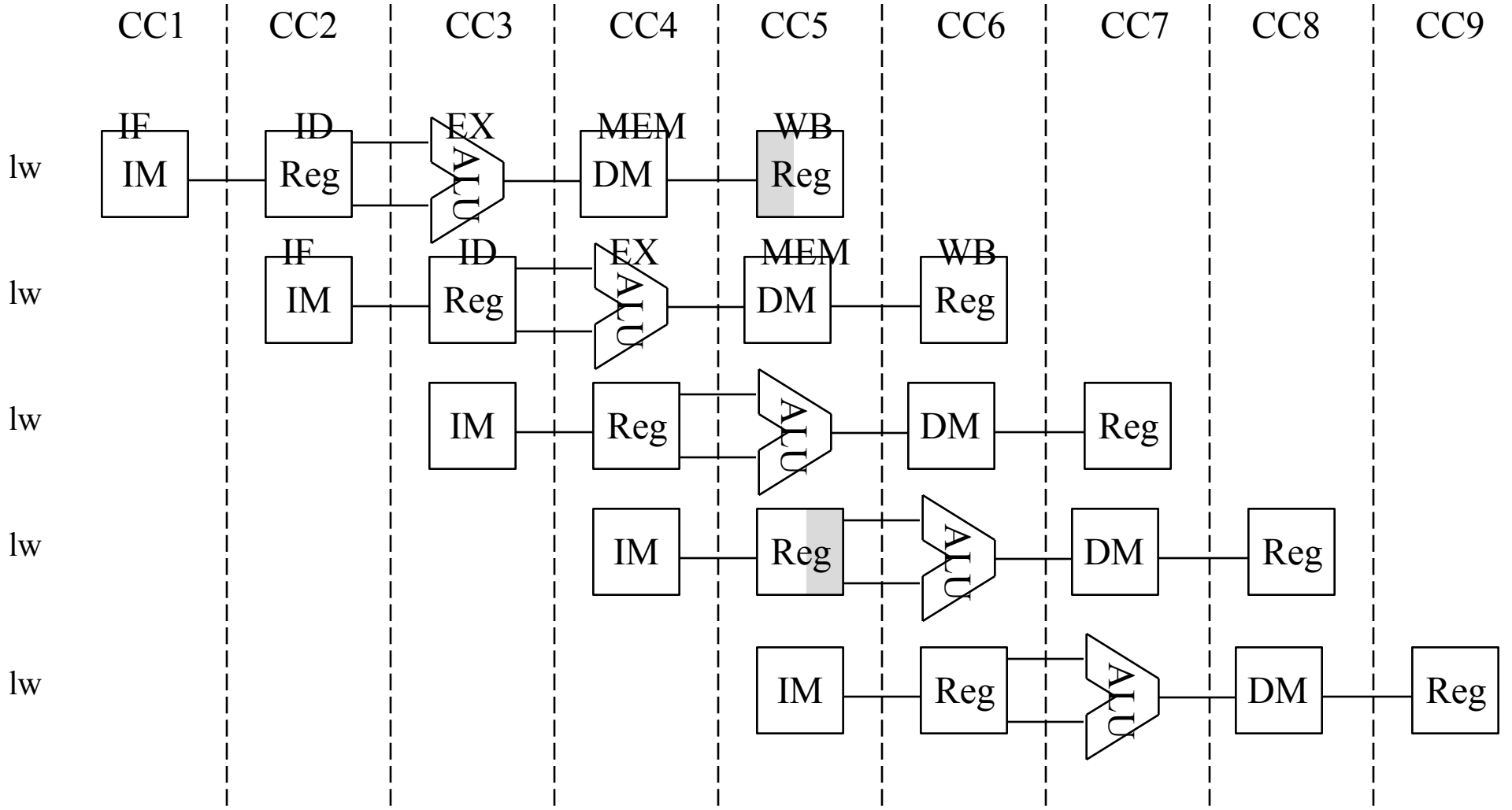
Which of the statements below is true about a pipelined processor?

| Choice | Statement |
|---|---|
| A | Both instruction latency and throughput remain essentially unchanged |
| B | Instruction latency remains essentially unchanged from single-cycle;<br>Instruction throughput increases by a factor of 5 |
| C | Instruction latency improves by a factor of 5 over single-cycle;<br>Instruction throughput remains essentially unchanged |
| D | Both latency and throughput improve by a factor of 5 over single-cycle |
| E | None of the above |

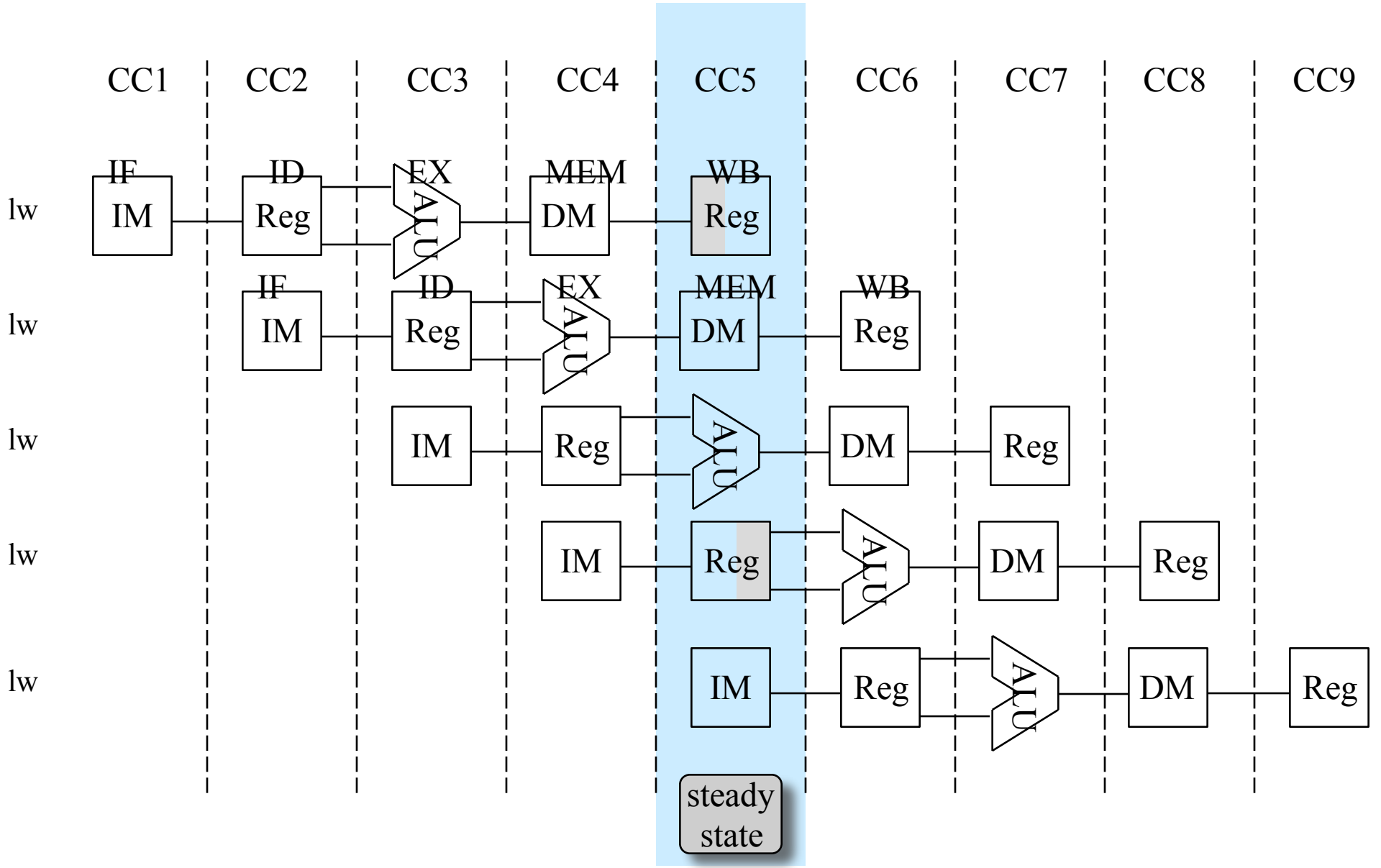
# Idea for a Pipelined Datapath



# Execution in a Pipelined Datapath



# Execution in a Pipelined Datapath

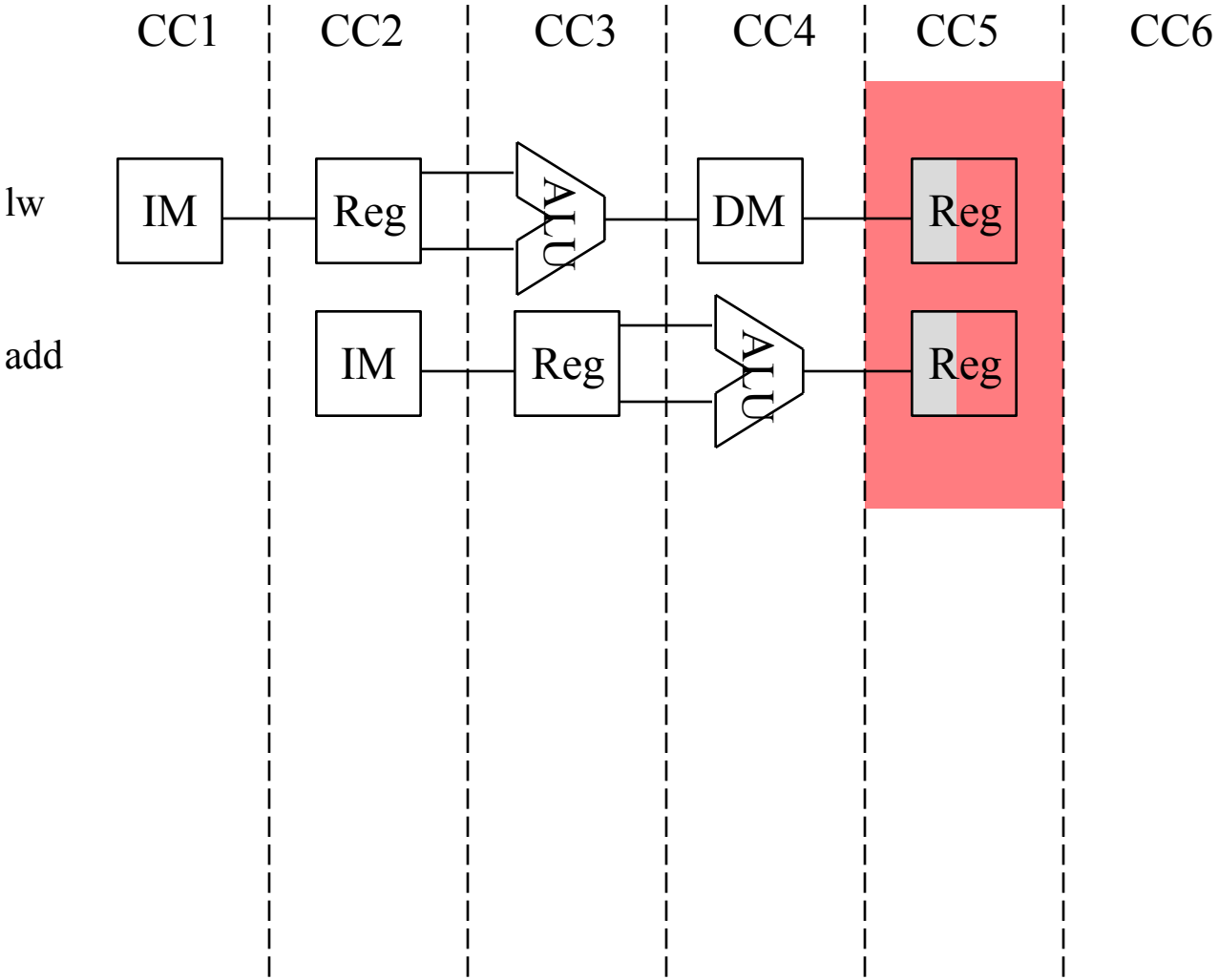


# Pipeline Stages

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

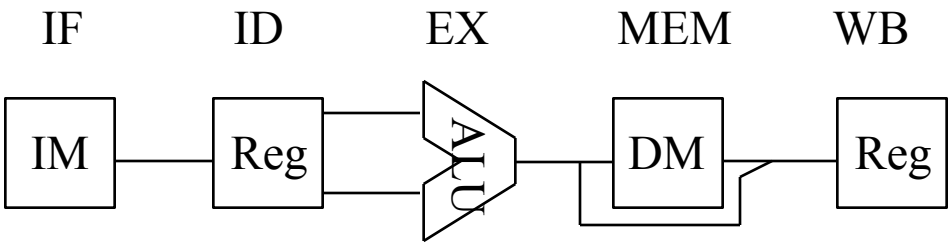
| Selection | Yes/No | Reason (Choose BEST answer) |
|---|---|---|
| A | Yes | Decreasing R-type to 4 cycles improves instruction throughput |
| B | Yes | Decreasing R-type to 4 cycles improves instruction latency |
| C | No | Decreasing R-type to 4 cycles causes hazards |
| D | No | Decreasing R-type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

# Mixed Instructions in the Pipeline

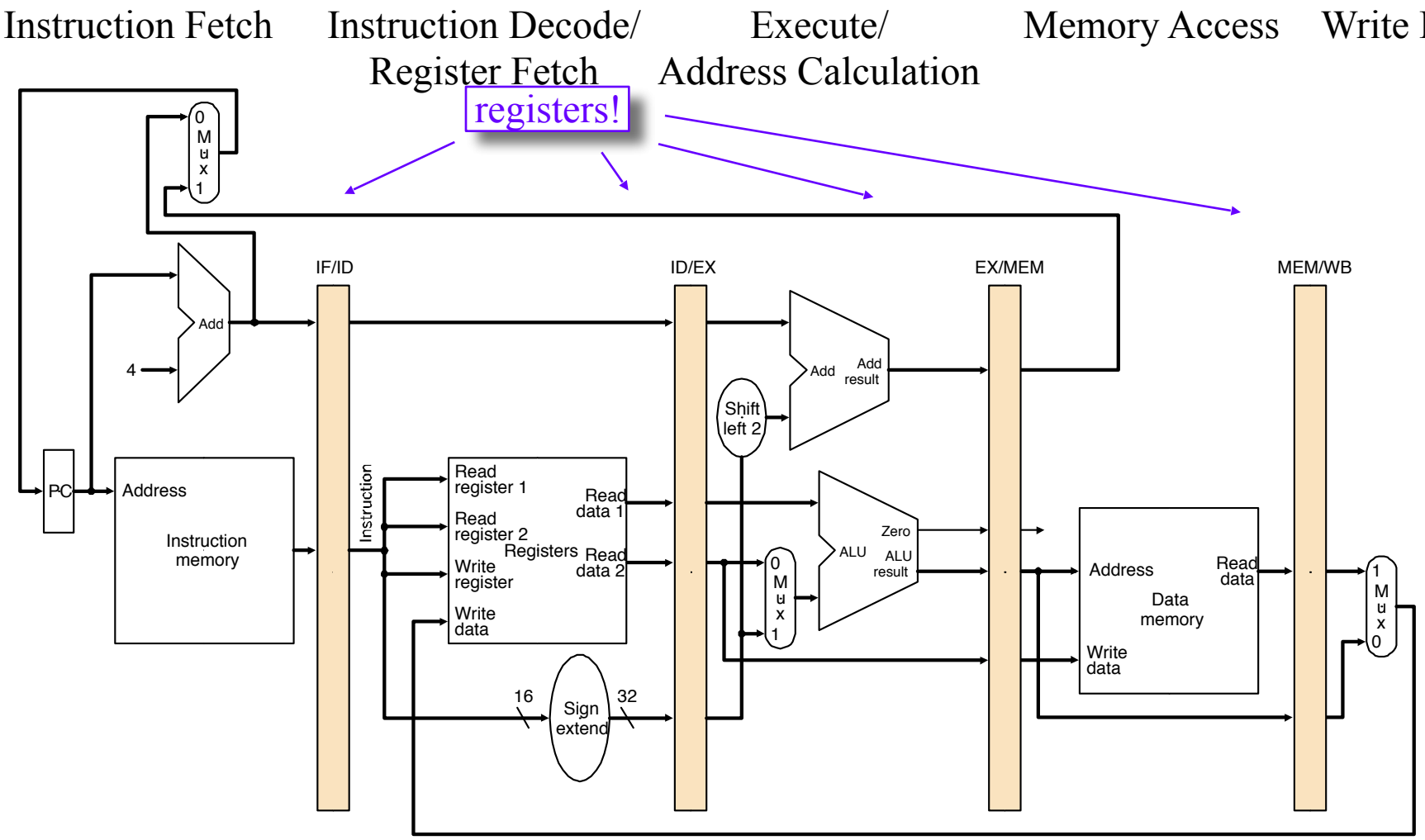


# Pipeline Principles

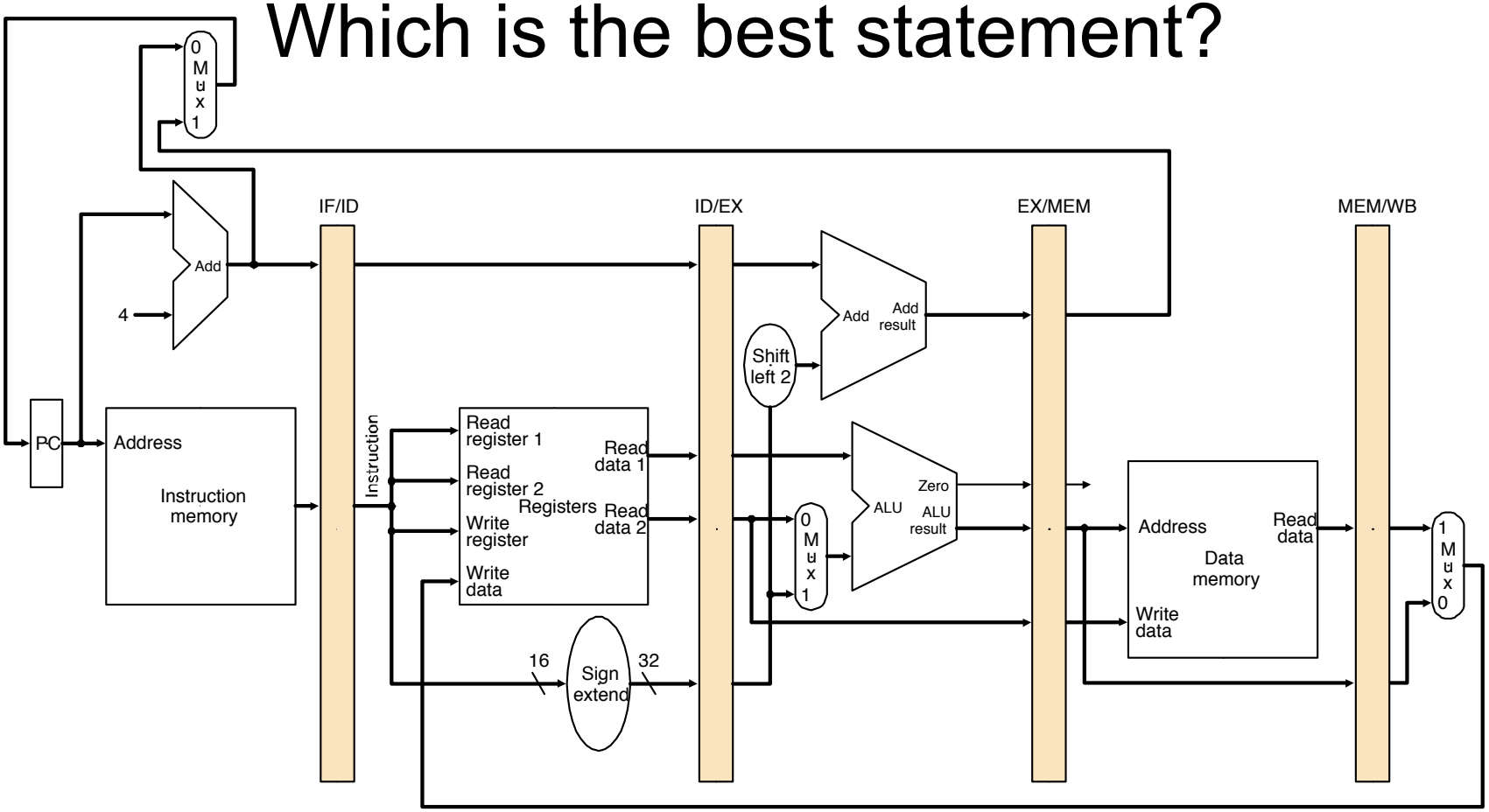
- All instructions that share a pipeline must have the same *stages* in the same *order*.
  - therefore, *add* does nothing during Mem stage
  - *sw* does nothing during WB stage
- All intermediate values must be latched each cycle.
- There is no functional block reuse



# Pipelined Datapath

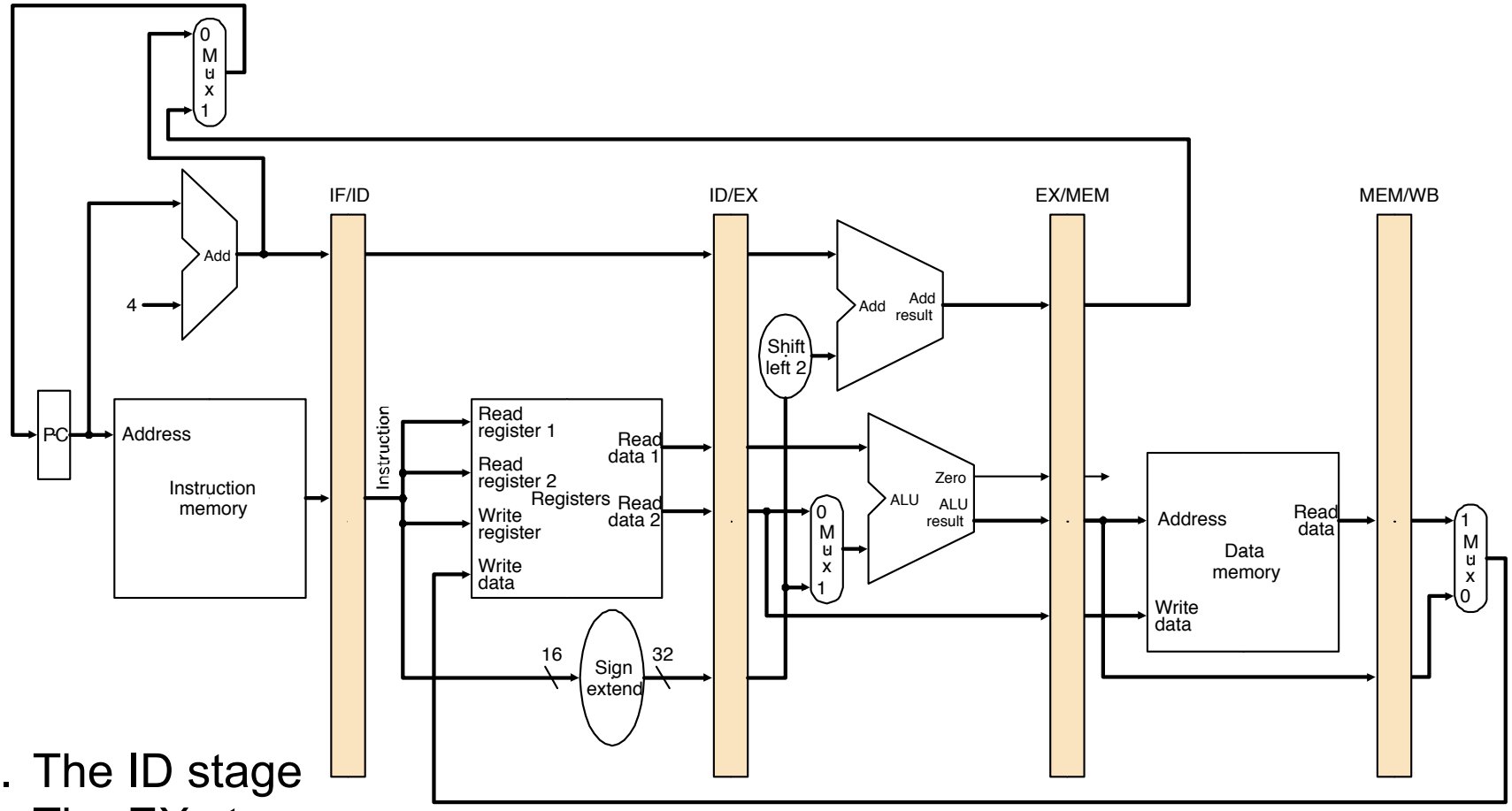


# Which is the best statement?



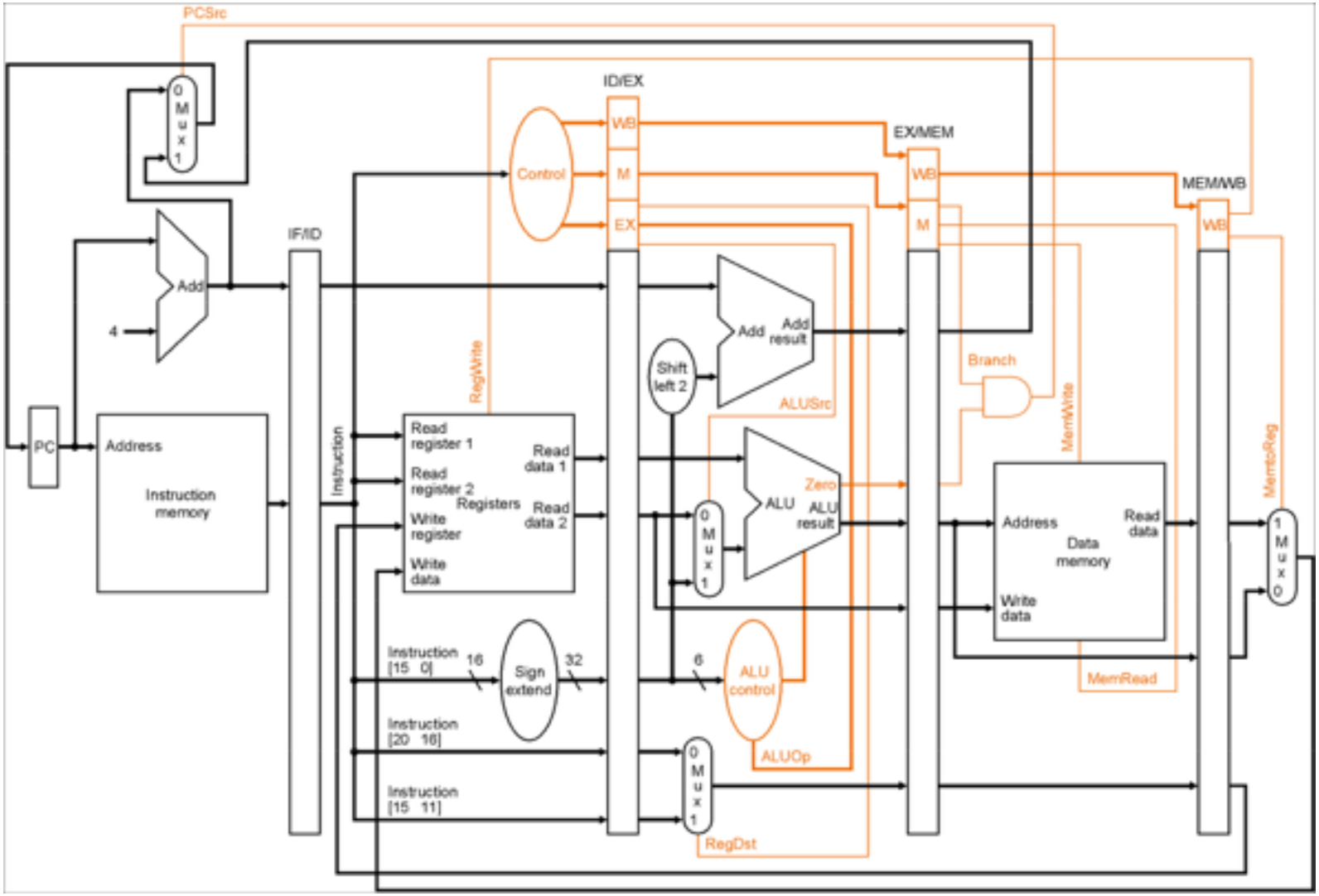

A. All the pipeline registers are the same size.
B. The ID/EX register is the same size as the EX/MEM register.
C. The IF/ID register is the largest; MEM/WB is the smallest.
D. The ID/EX register is the largest; IF/ID is the smallest.
E. None of the above.

# Where does multiplication execute?

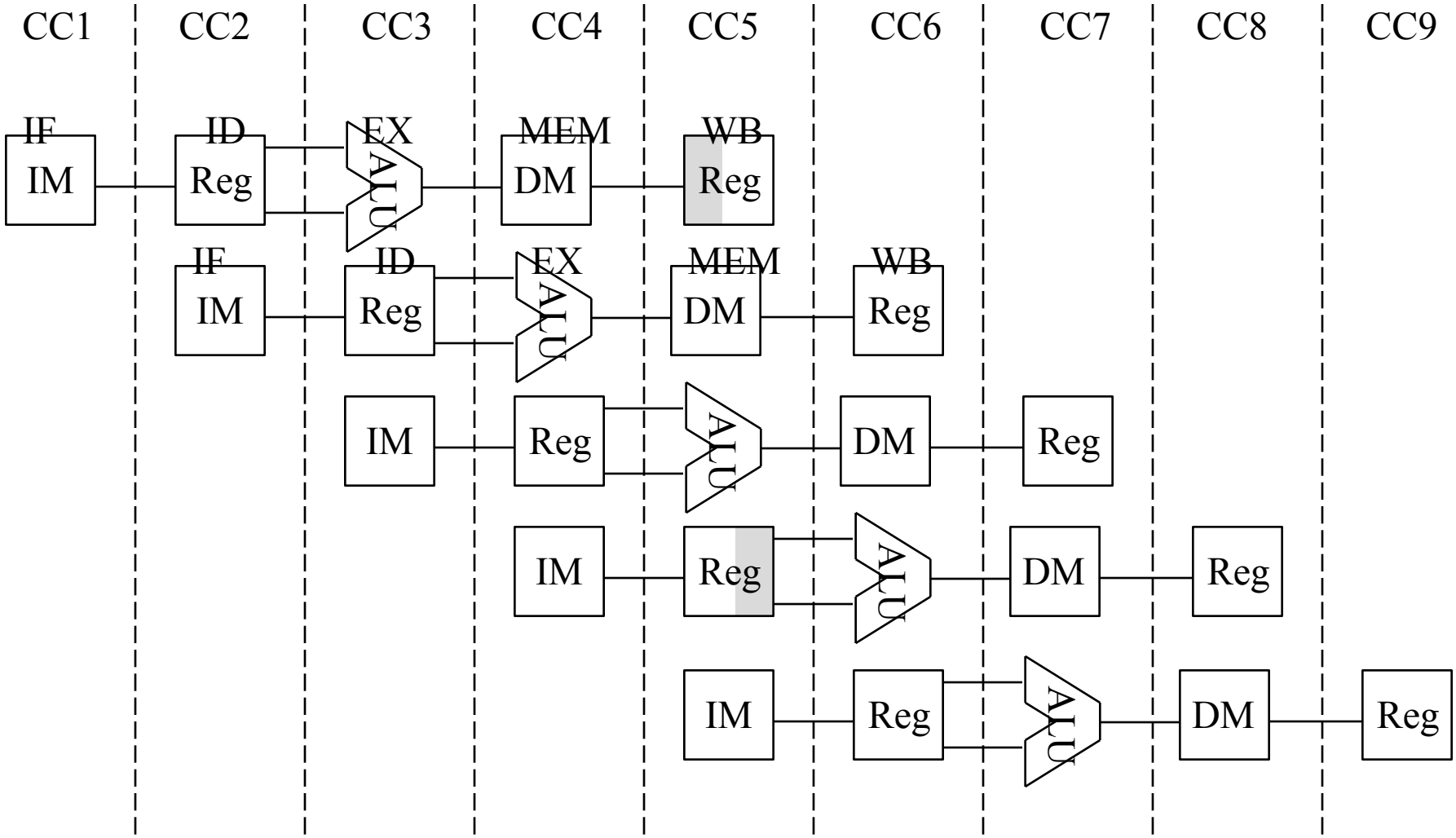


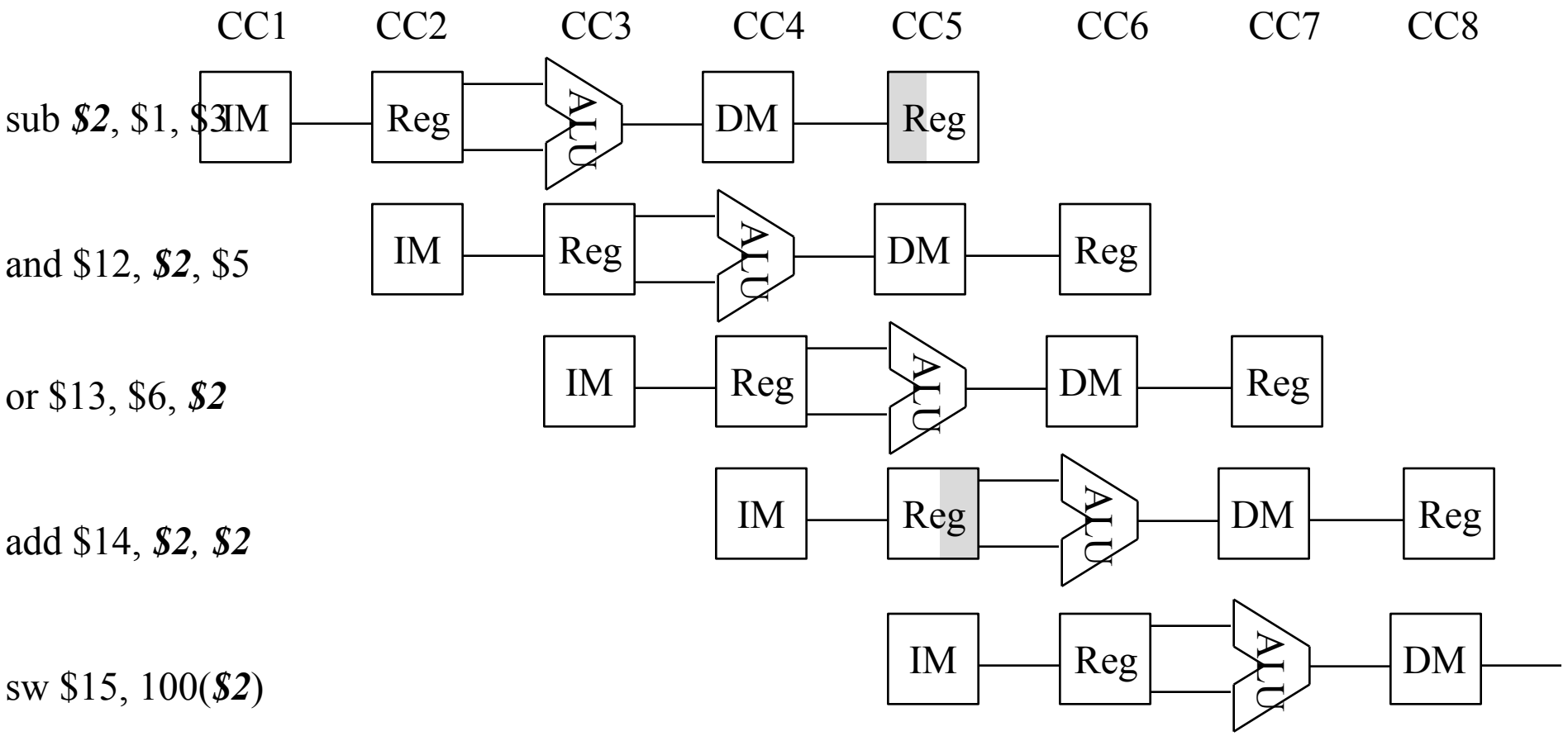

A. The ID stage
B. The EX stage
C. The MEM stage
D. The WB stage
E. None of the above

# The Pipeline with Control Logic



# Execution in a Pipelined Datapath



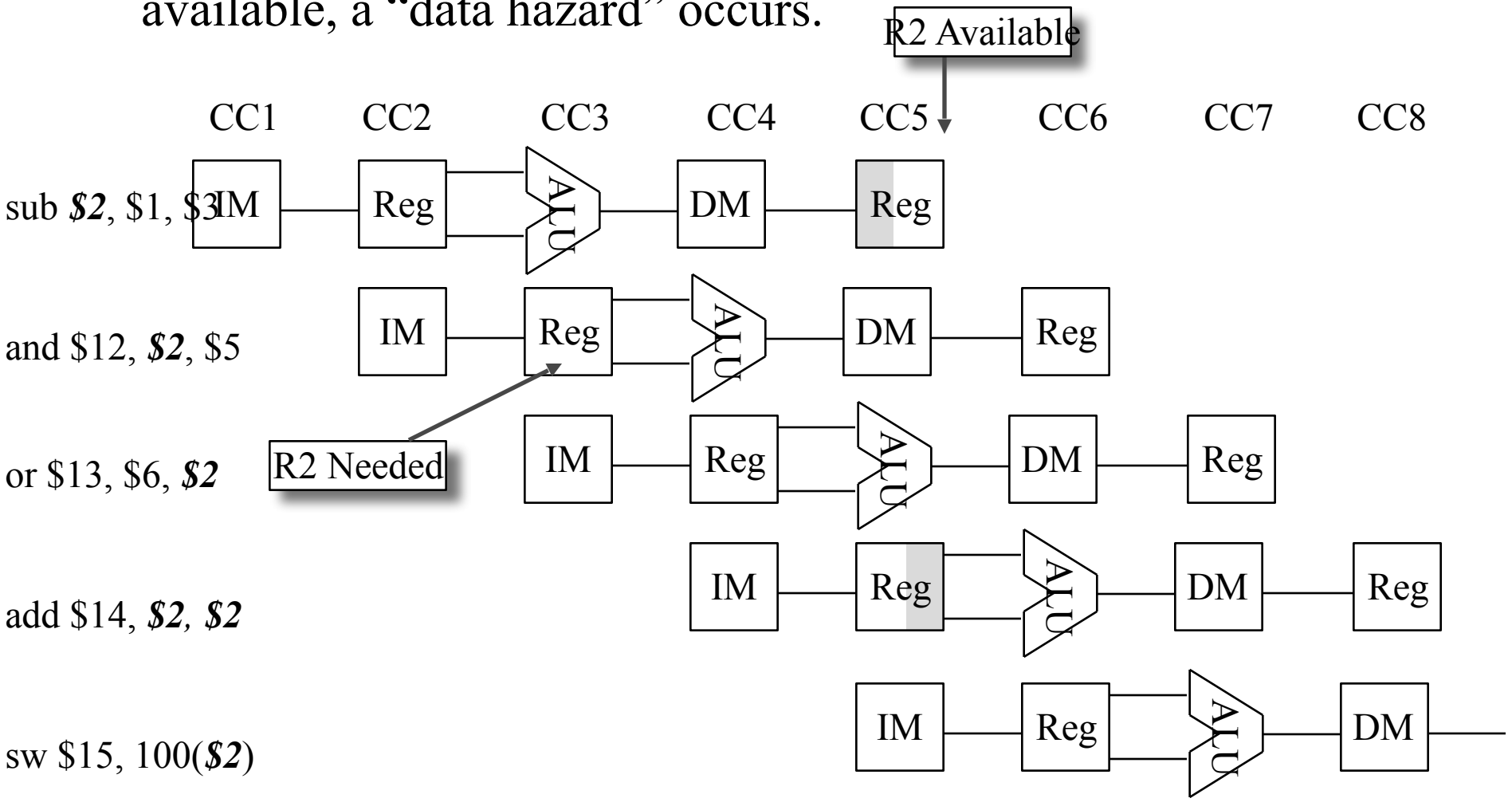



What just happened here which is problematic (BEST ANSWER)?

A. The register file is trying to read and write the same register.
B. The ALU and data memory are both active in the same cycle.
C. A value is used before it is produced.
D. Both A and B
E. Both A and C

# Data Hazards

- When a result is needed in the pipeline before it is available, a "data hazard" occurs.



# Hazards continued

- What happens when...

```
add $3, $10, $11
lw $8, 1000($3)
sub $11, $8, $7
```