Introduction to Computer Architecture

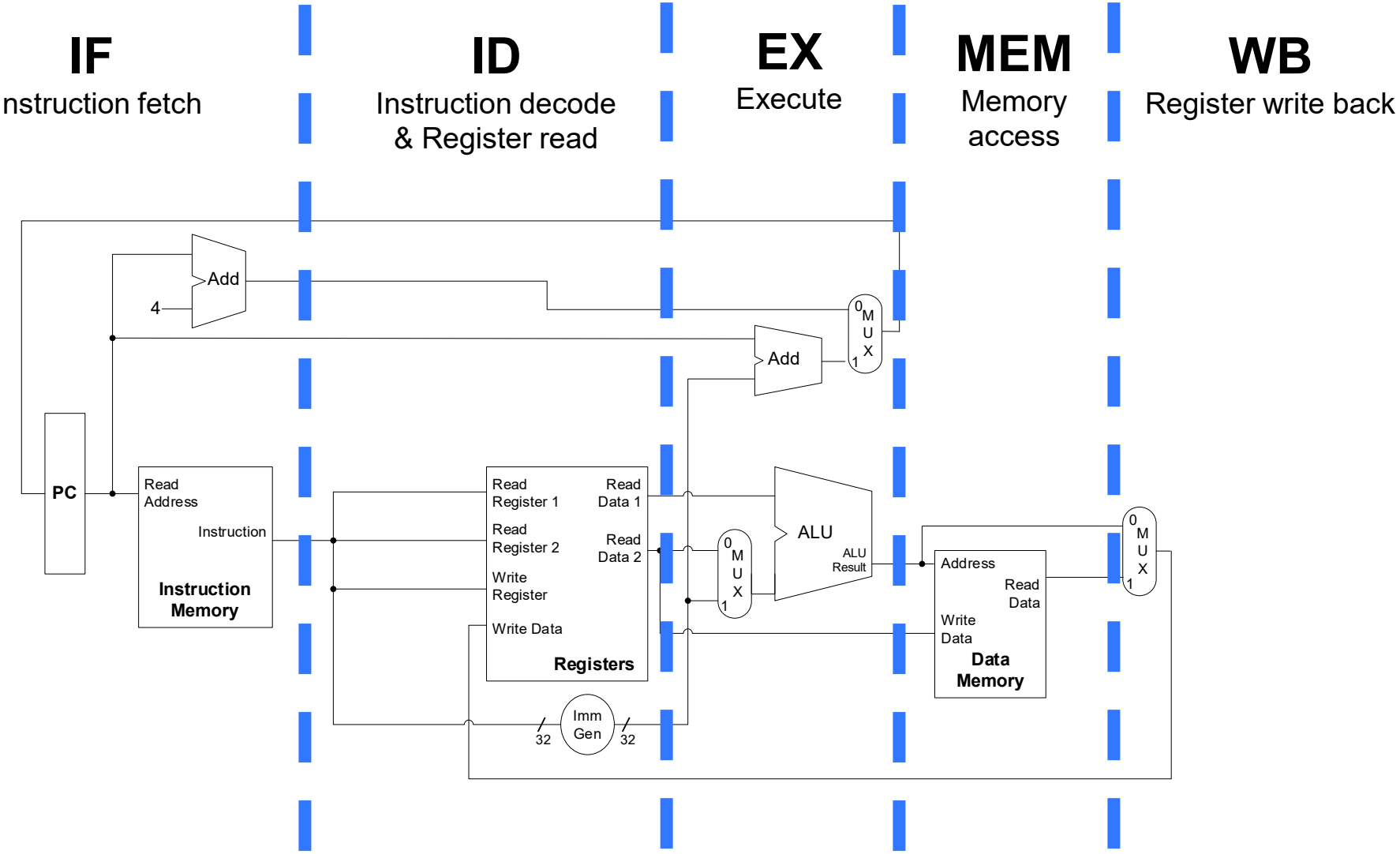
Chapter 4

Pipelined Datapath and Control

Hyungmin Cho

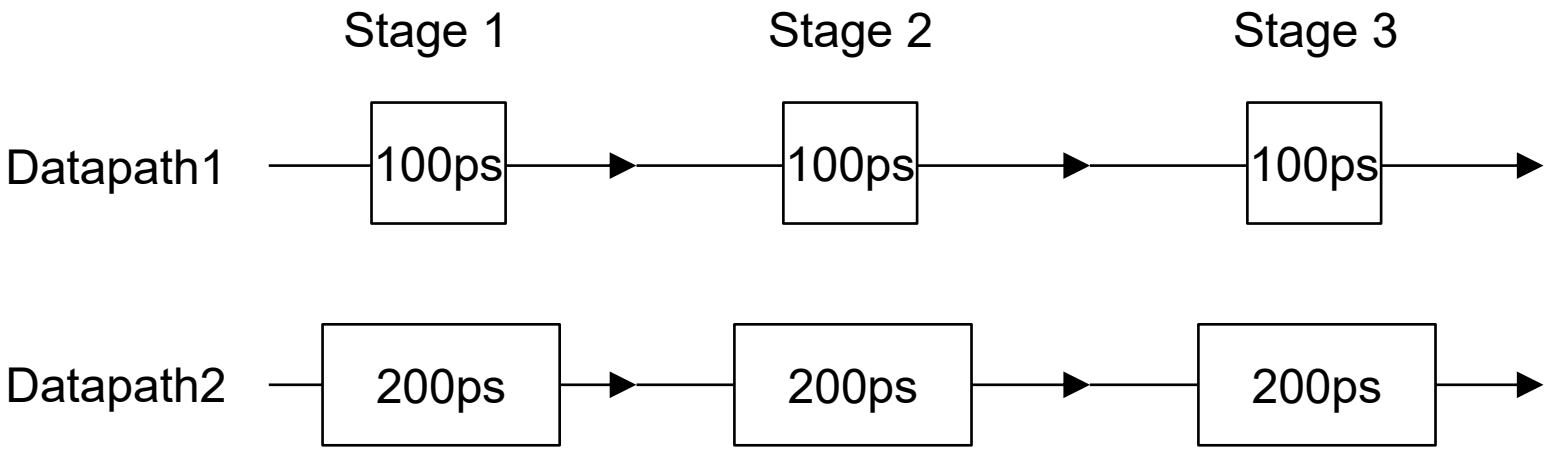
Department of Computer Science and Engineering
Sungkyunkwan University

RISC-V Pipelined Datapath



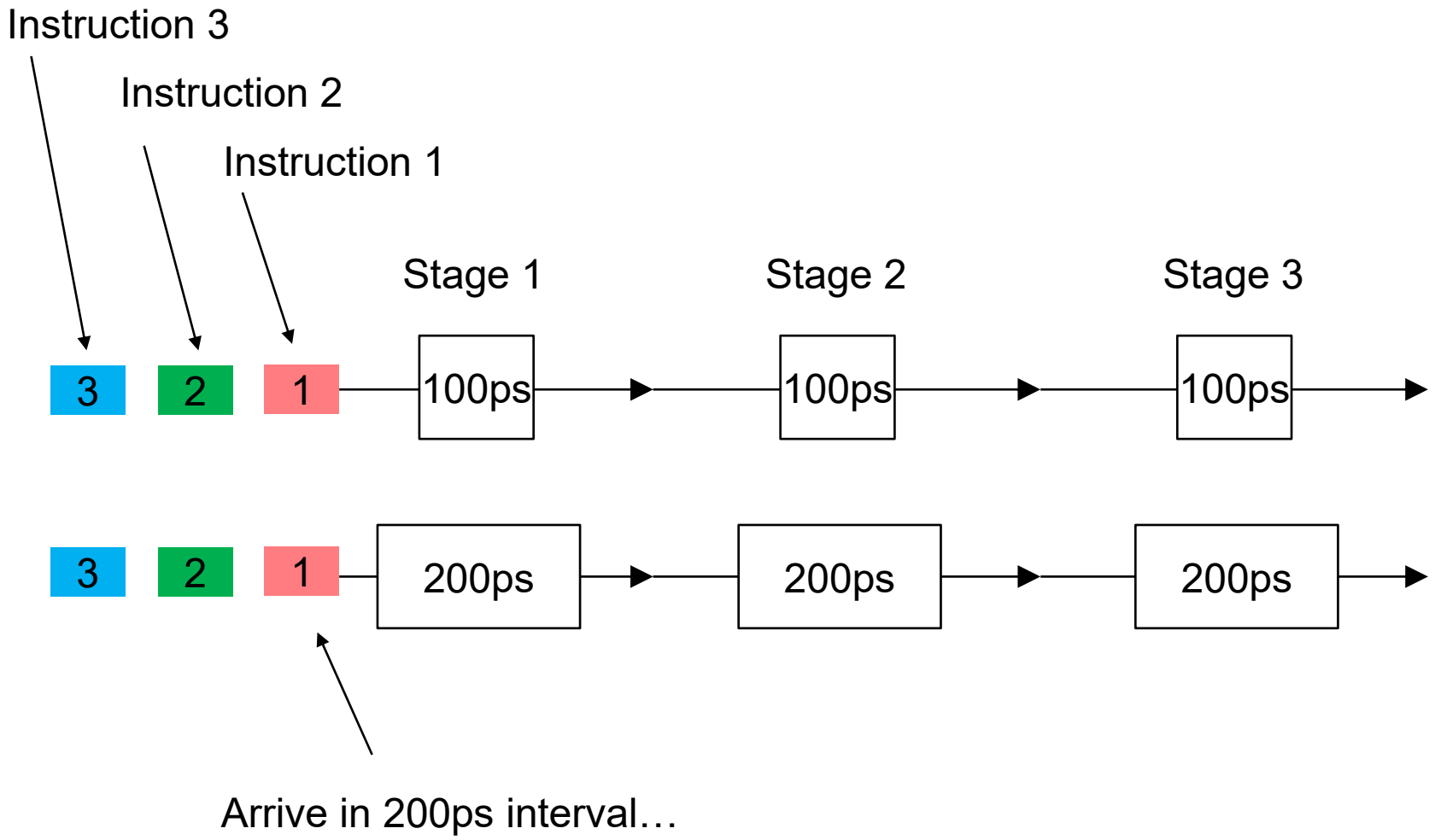
Dividing Pipeline Stages

- Need to synchronize multiple datapaths



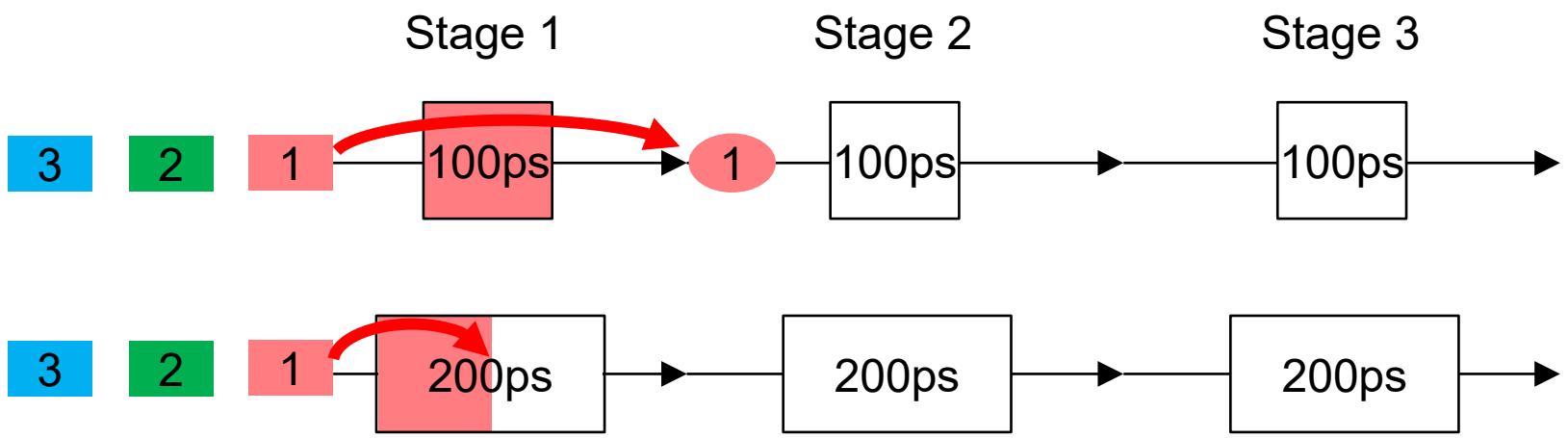
- (Almost) impossible to do it with combinational circuits only

No separation?



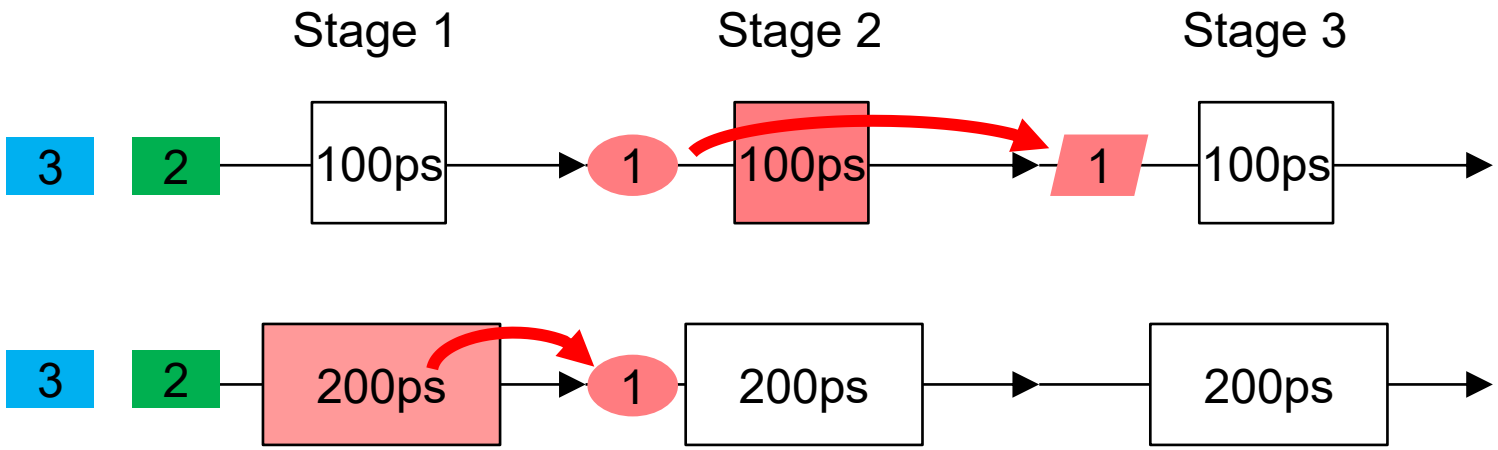
No separation?

After 100ps…



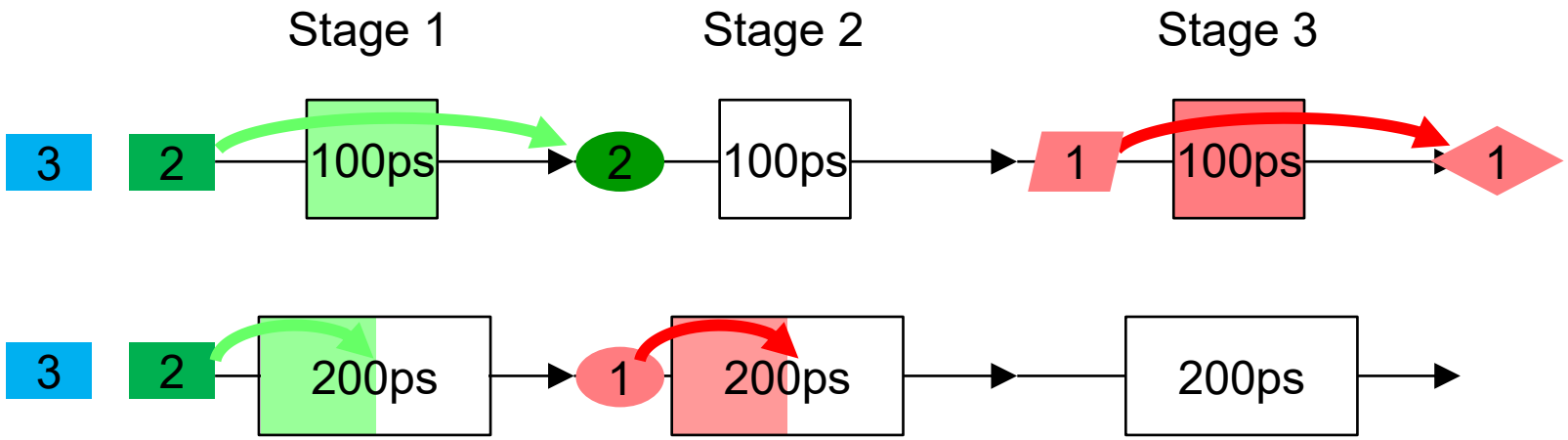
No separation?

After 200ps…



No separation?

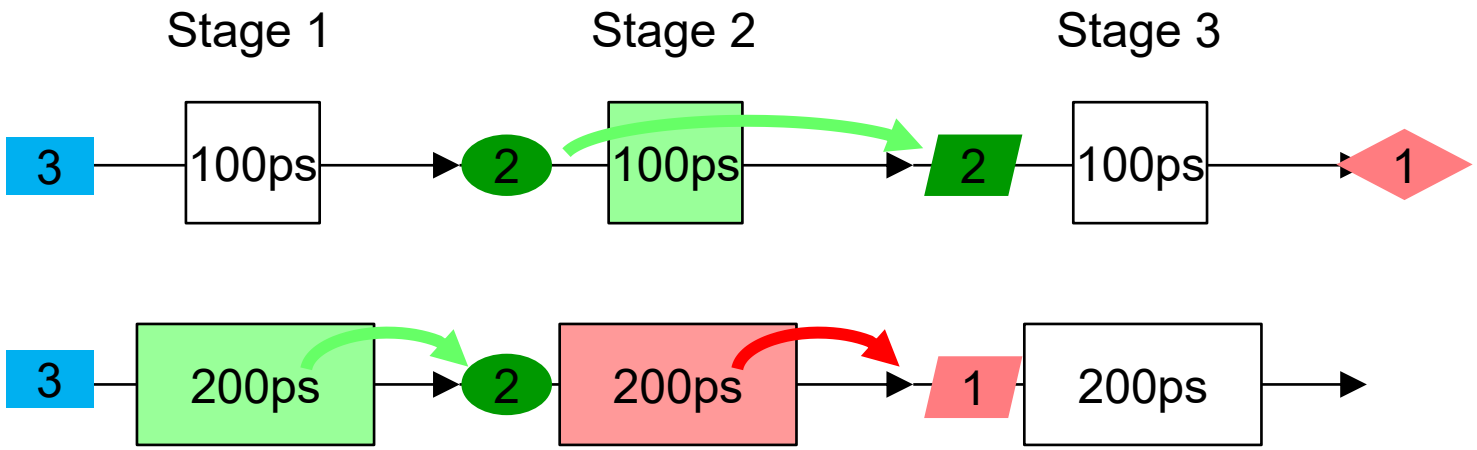
After 300ps…



Instructions in stage 2 do not match!

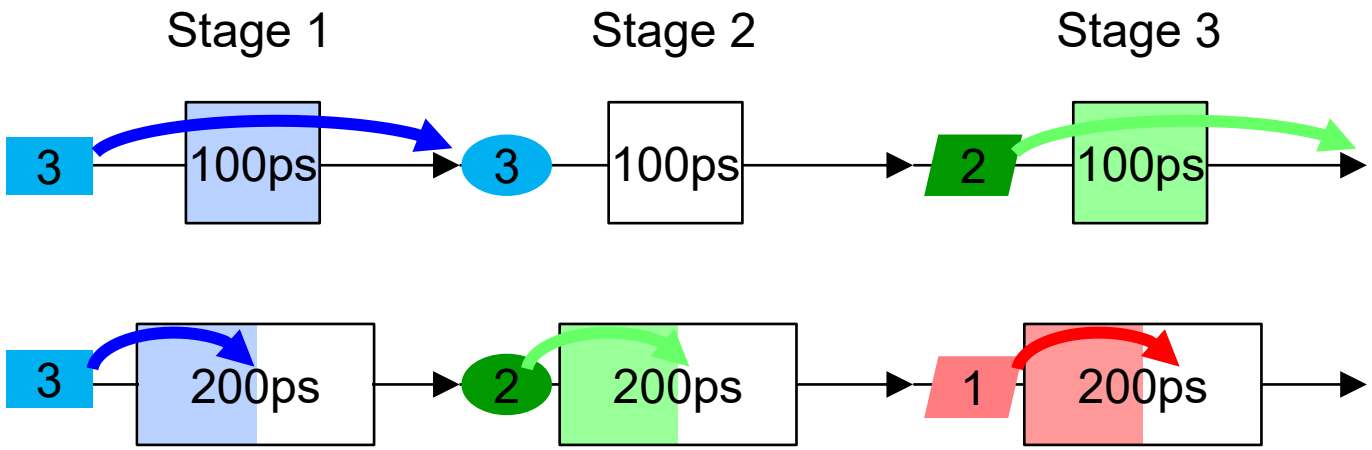
No separation?

After 400ps…



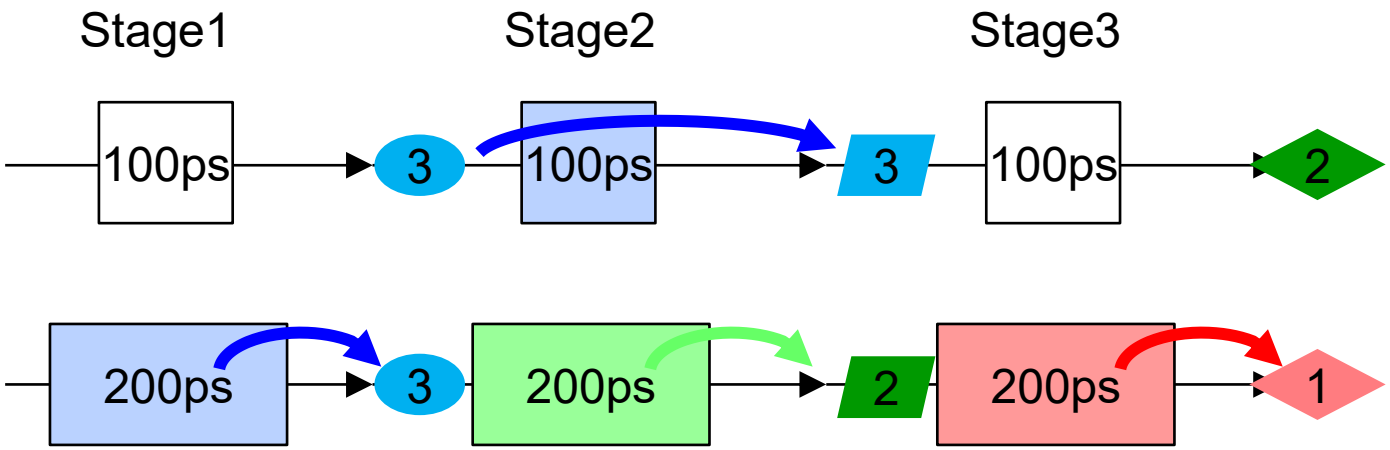
No separation?

After 500ps…

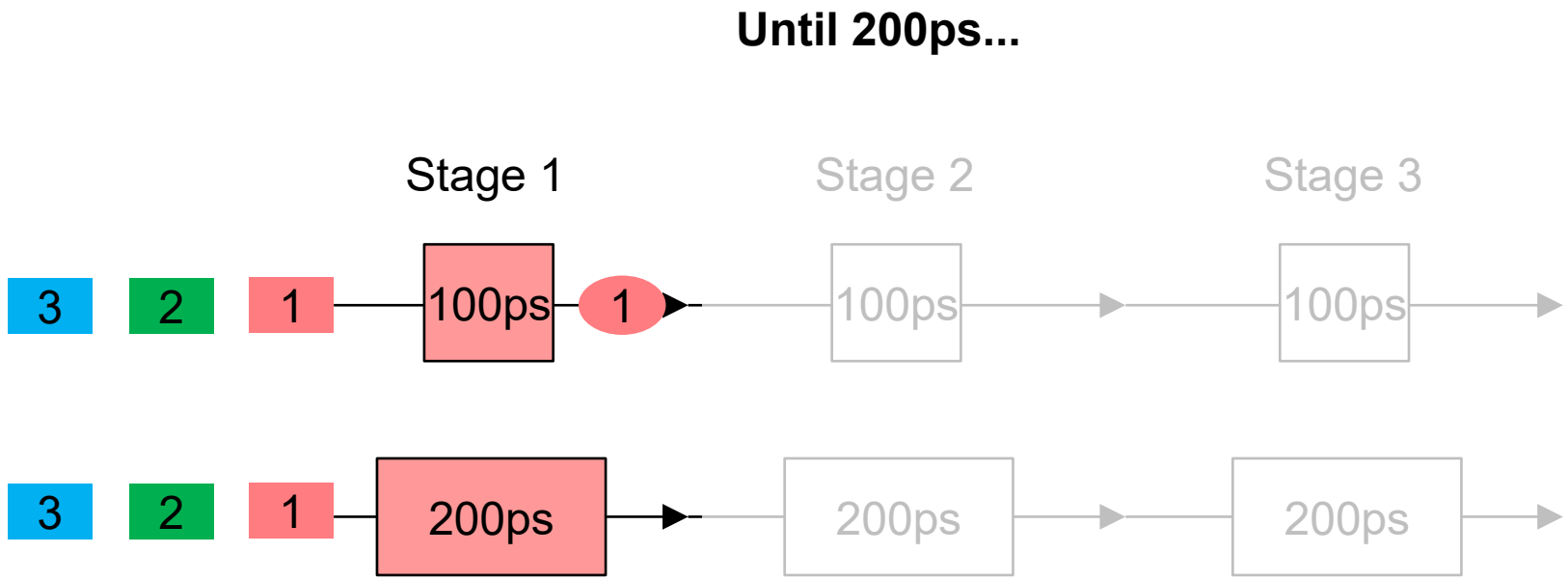


No separation?

After 600ps…

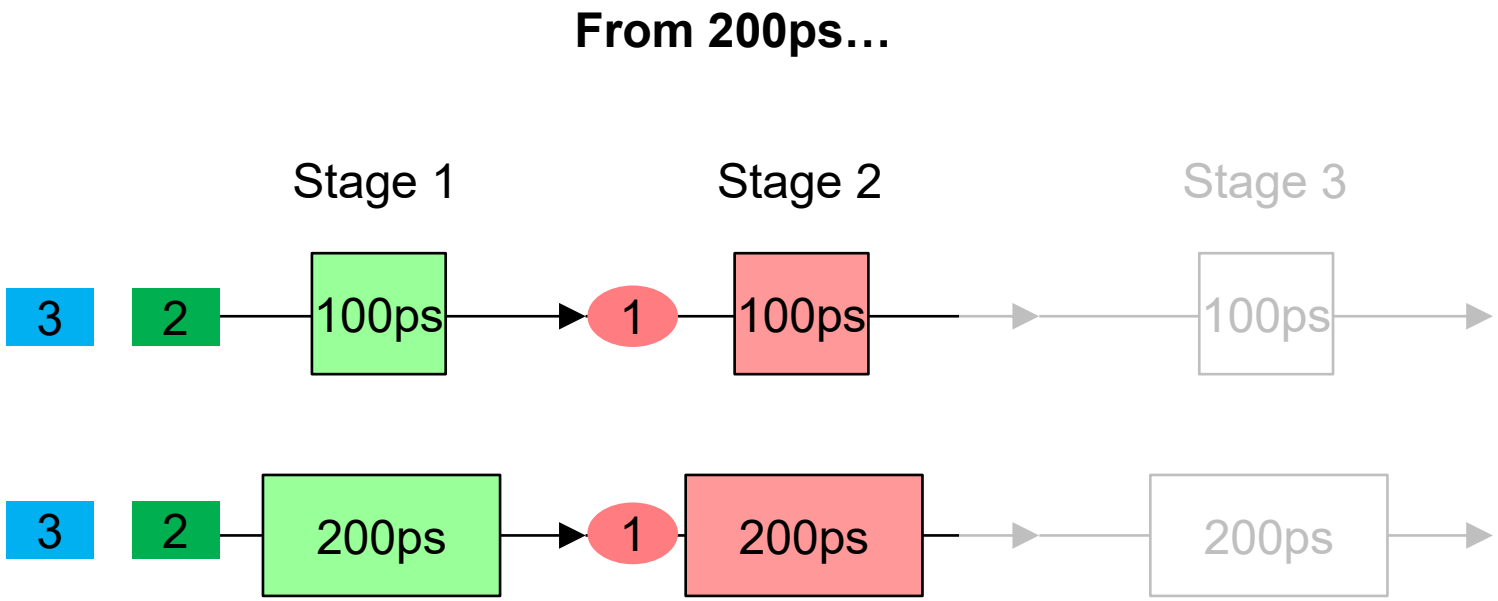


Dividing Stages – What should be done…



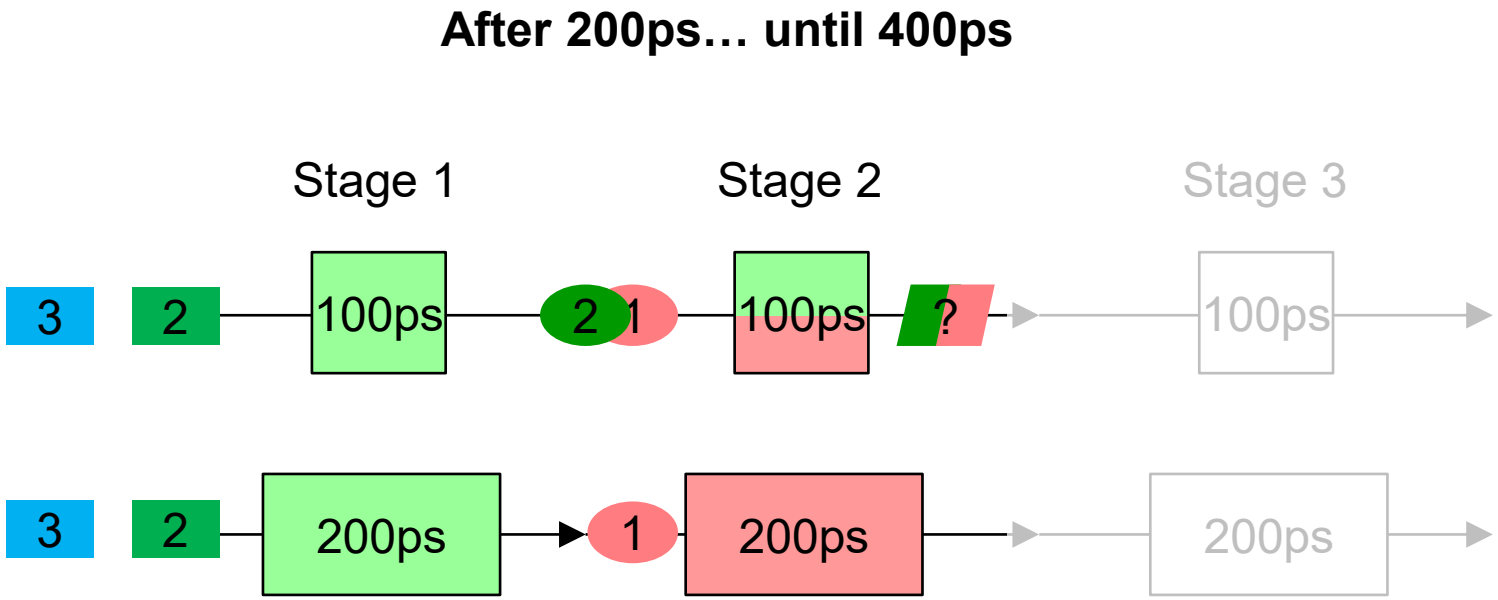
- Do not start stage2 yet…
- Wait until datapath2 also completes

Dividing Stages – What should be done…



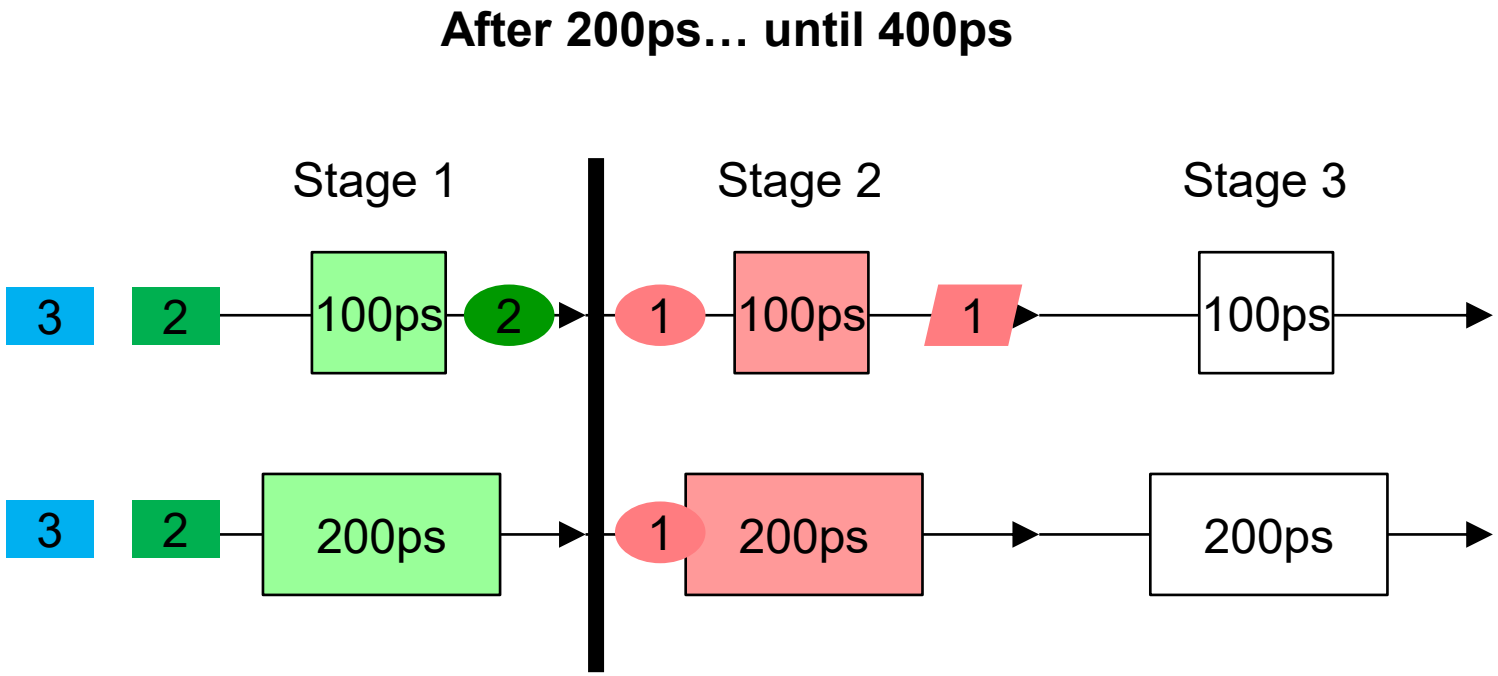
- Drive stage2 (i.e., give input data to stage2)

Dividing Stages – What should be done…



- Stage1 produces new output for Instruction2
- However, we still need to drive stage2 with Instruction1

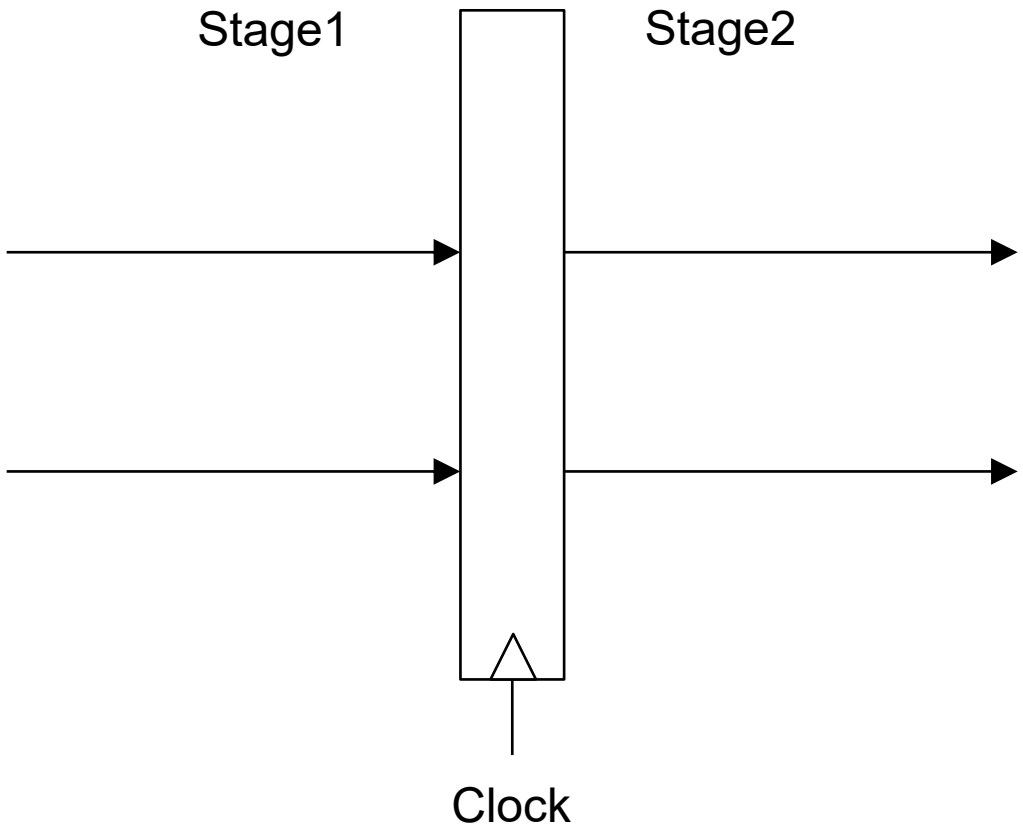
Dividing Stages – What should be done…



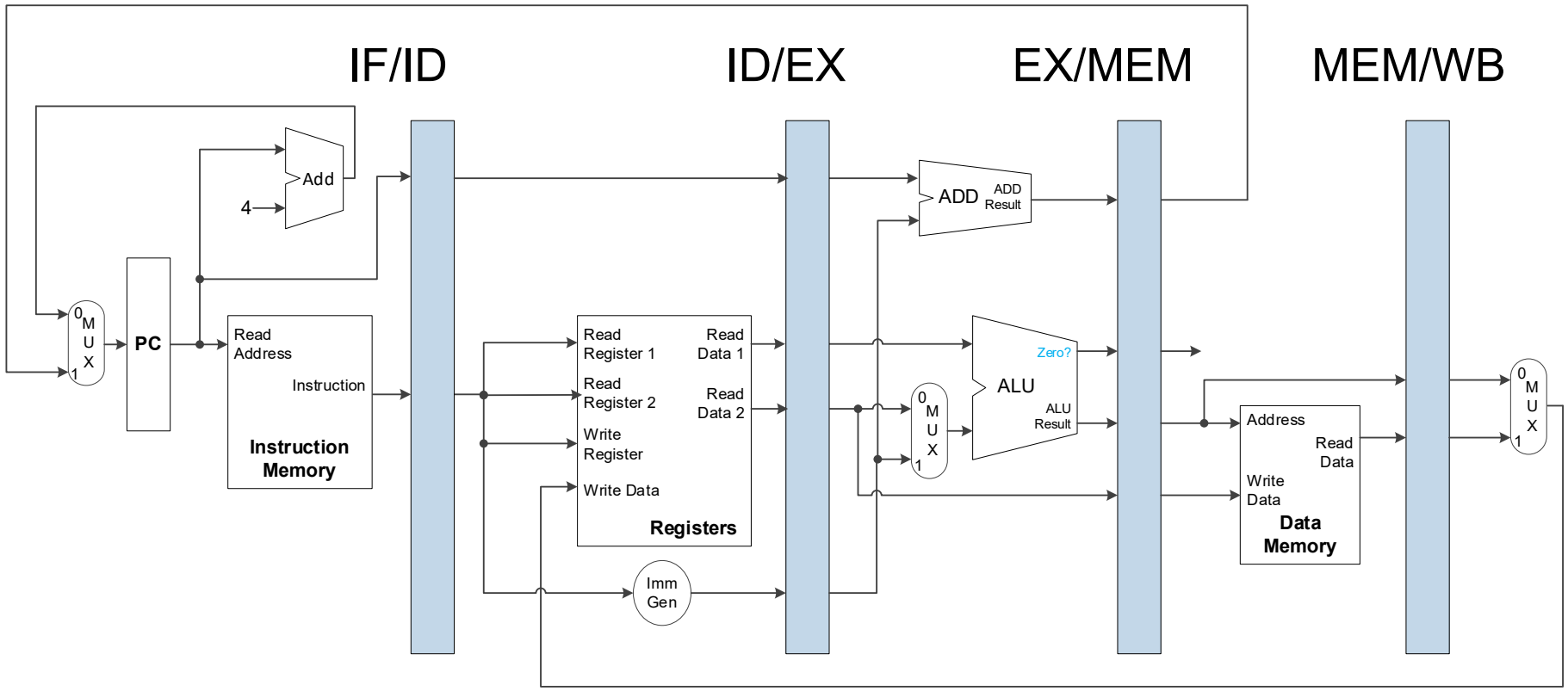
Between stages, we need a "**barrier**" that remembers the previous instruction and hold the next instruction

Using Registers

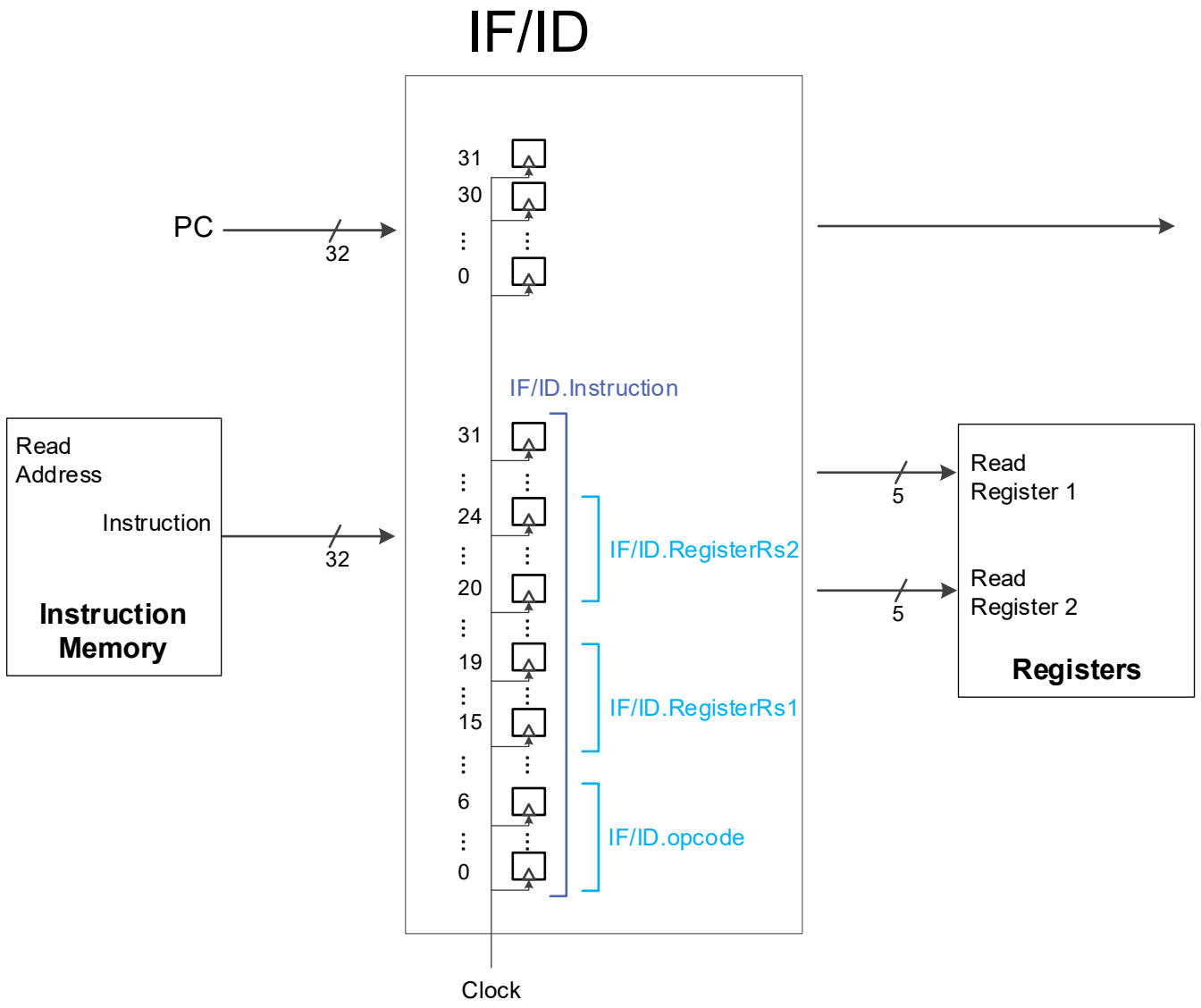
- Need registers (flip-flops) between stages



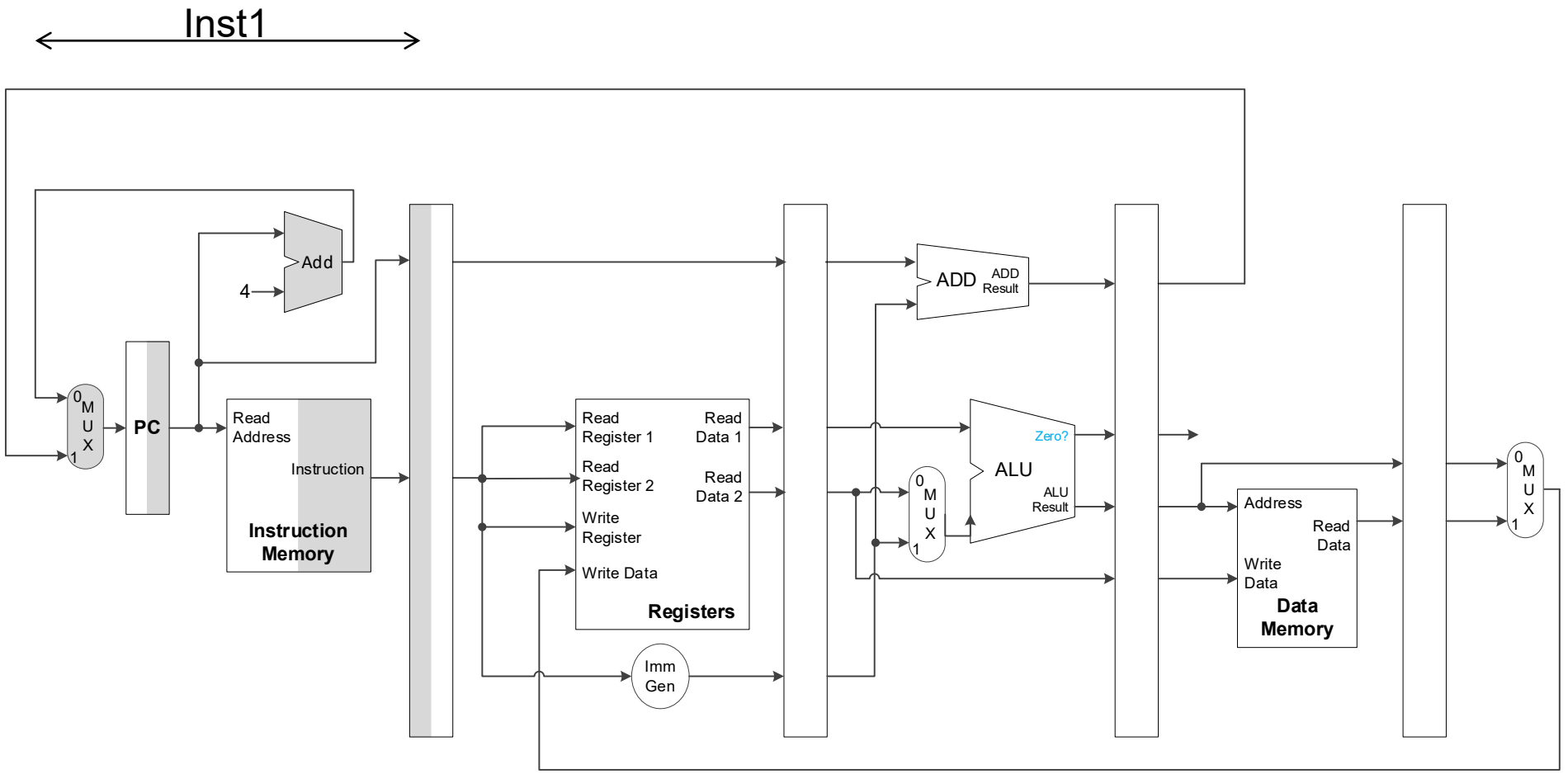
Pipeline Registers



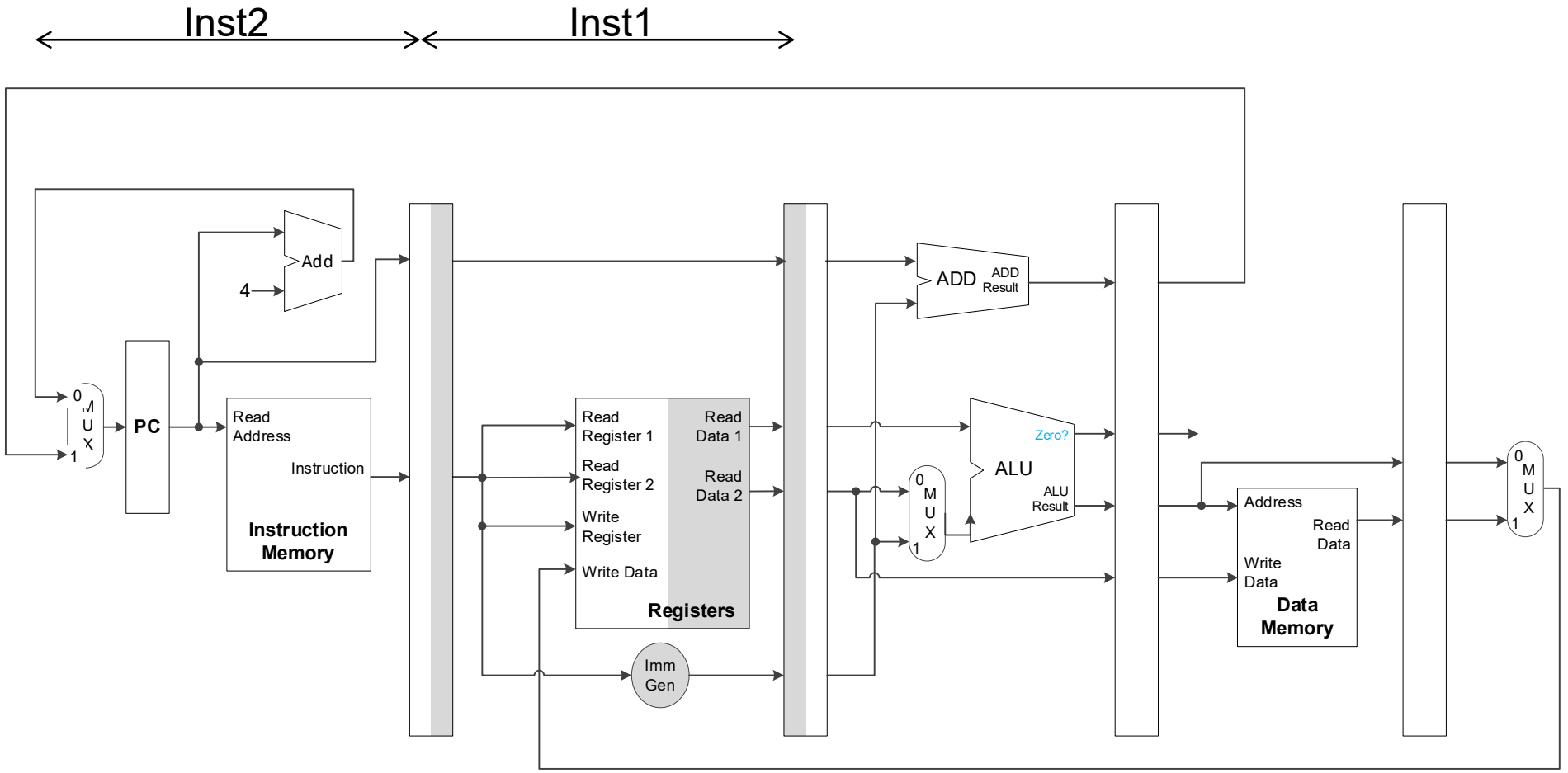
Pipeline Register Details



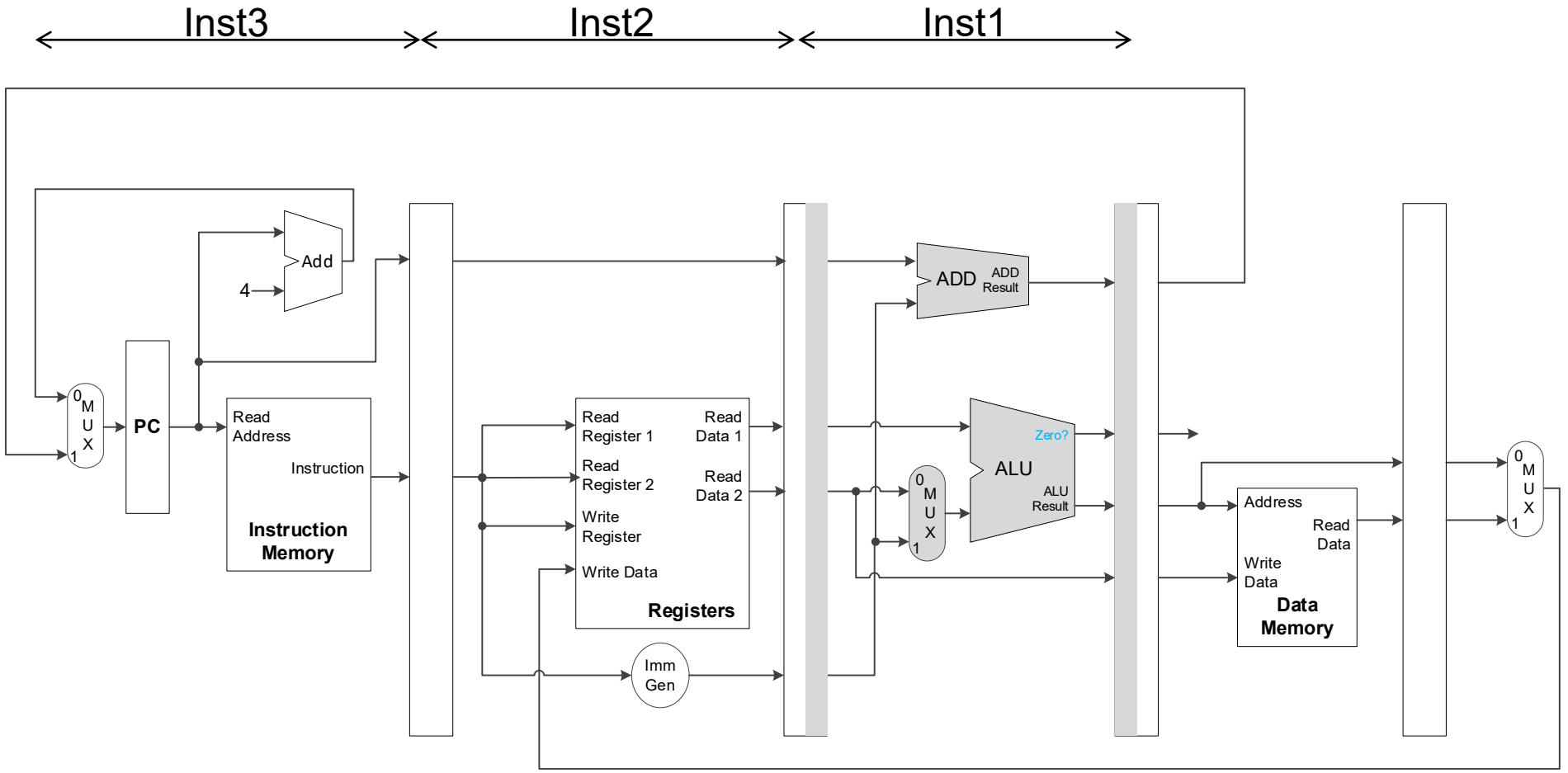
IF stage


Inst1
Add
4
0
M
U
X
1
PC
Read
Address
Instruction
Instruction
Memory
Read
Register 1
Read
Register 2
Write
Register
Write Data
Read
Data 1
Read
Data 2
Registers
Imm
Gen
ADD
ADD
Result
0
M
U
X
1
Zero?
ALU
ALU
Result
Address
Read
Data
Write
Data
Data
Memory
0
M
U
X
1

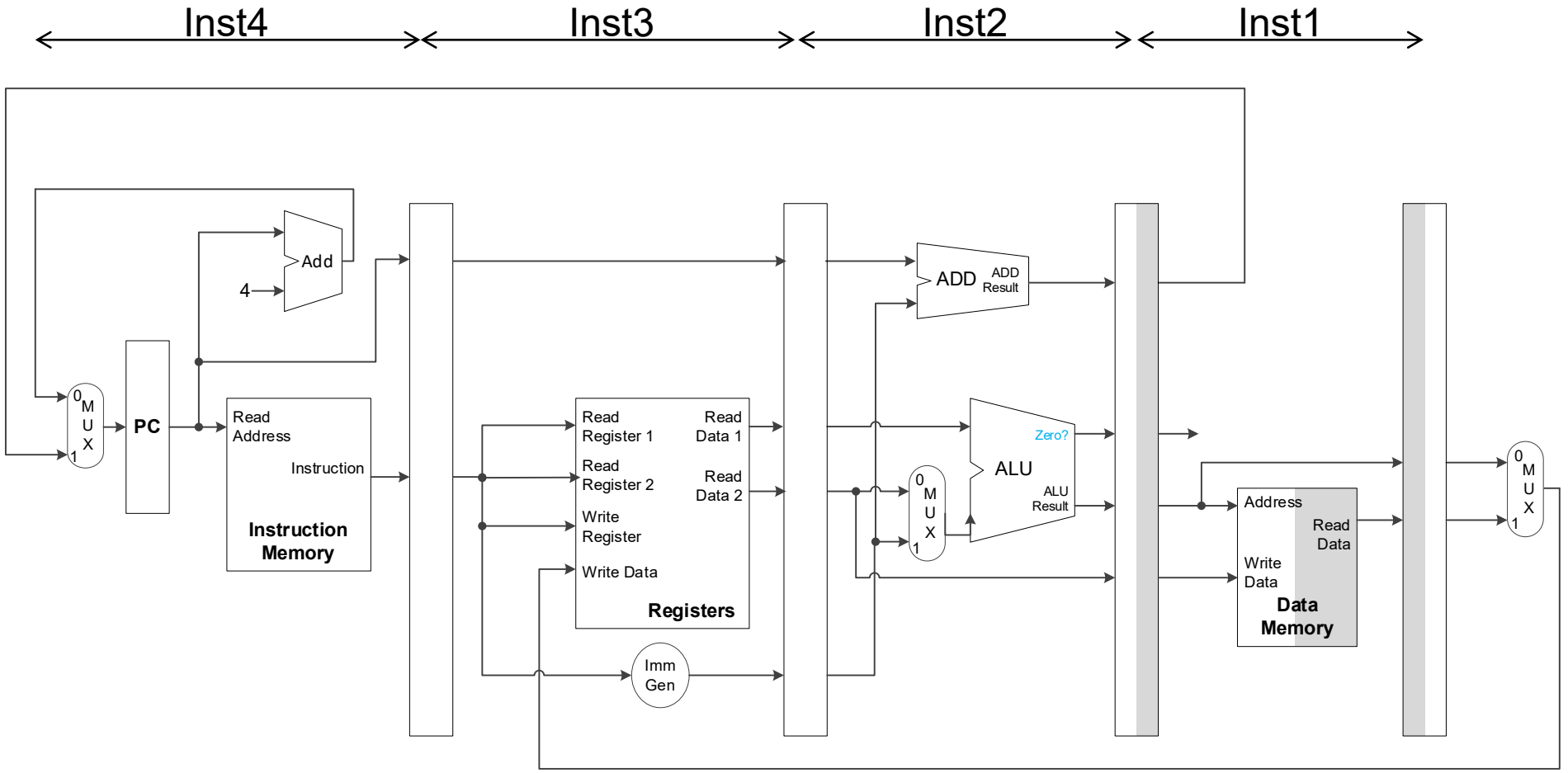
ID stage


Inst2
Inst1
Add
4
0
M
U
X
1
PC
Read
Address
Instruction
Instruction
Memory
Read
Register 1
Read
Register 2
Write
Register
Write Data
Read
Data 1
Read
Data 2
Registers
Imm
Gen
ADD
ADD
Result
0
M
U
X
1
ALU
Zero?
ALU
Result
Address
Read
Data
Write
Data
Data
Memory
0
M
U
X
1

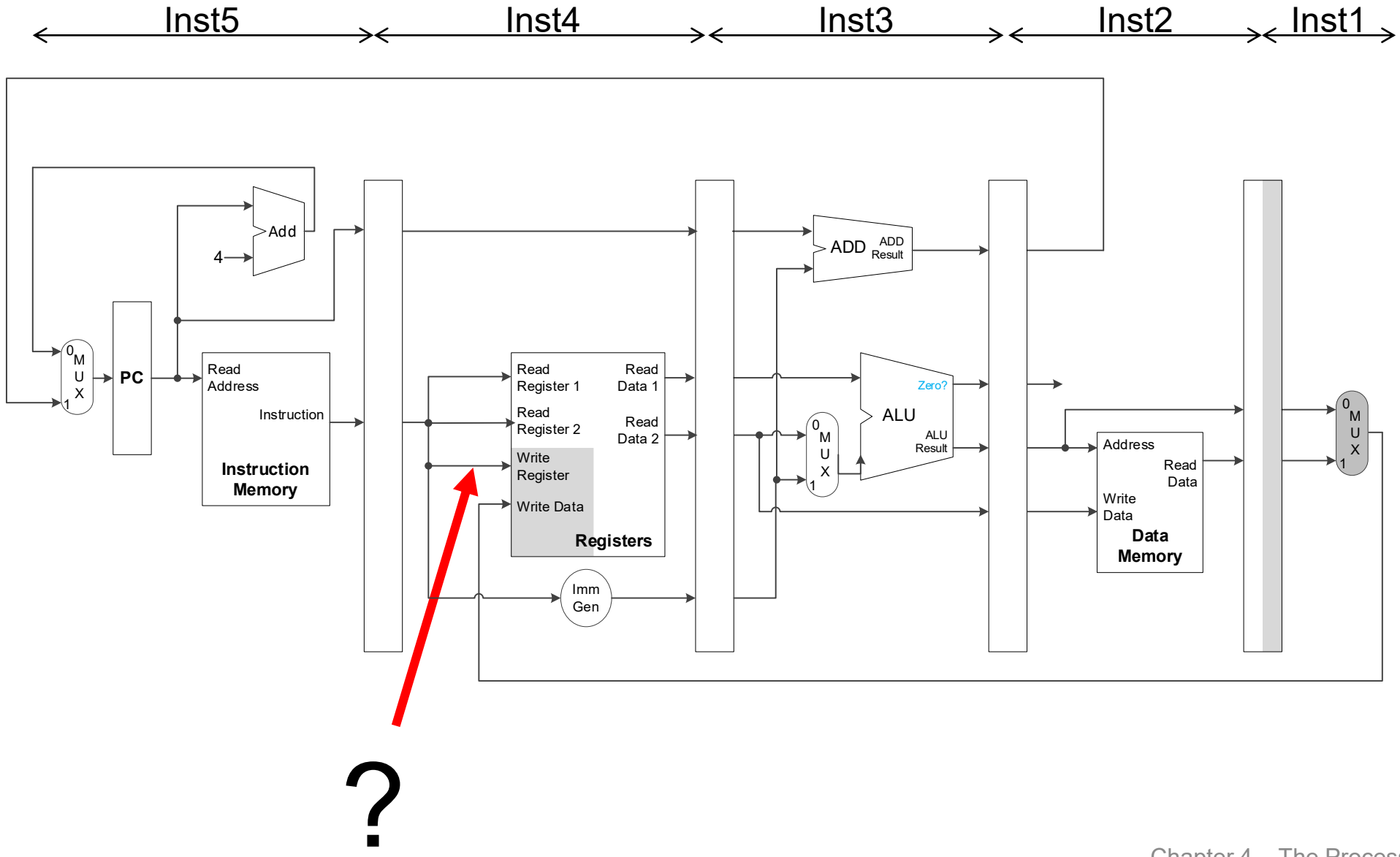
EX stage


Inst3
Inst2
Inst1
Add
4
0
M
U
X
1
PC
Read
Address
Instruction
Instruction
Memory
Read
Register 1
Read
Register 2
Write
Register
Write Data
Read
Data 1
Read
Data 2
Registers
Imm
Gen
ADD
ADD
Result
Zero?
ALU
ALU
Result
Address
Read
Data
Write
Data
Data
Memory

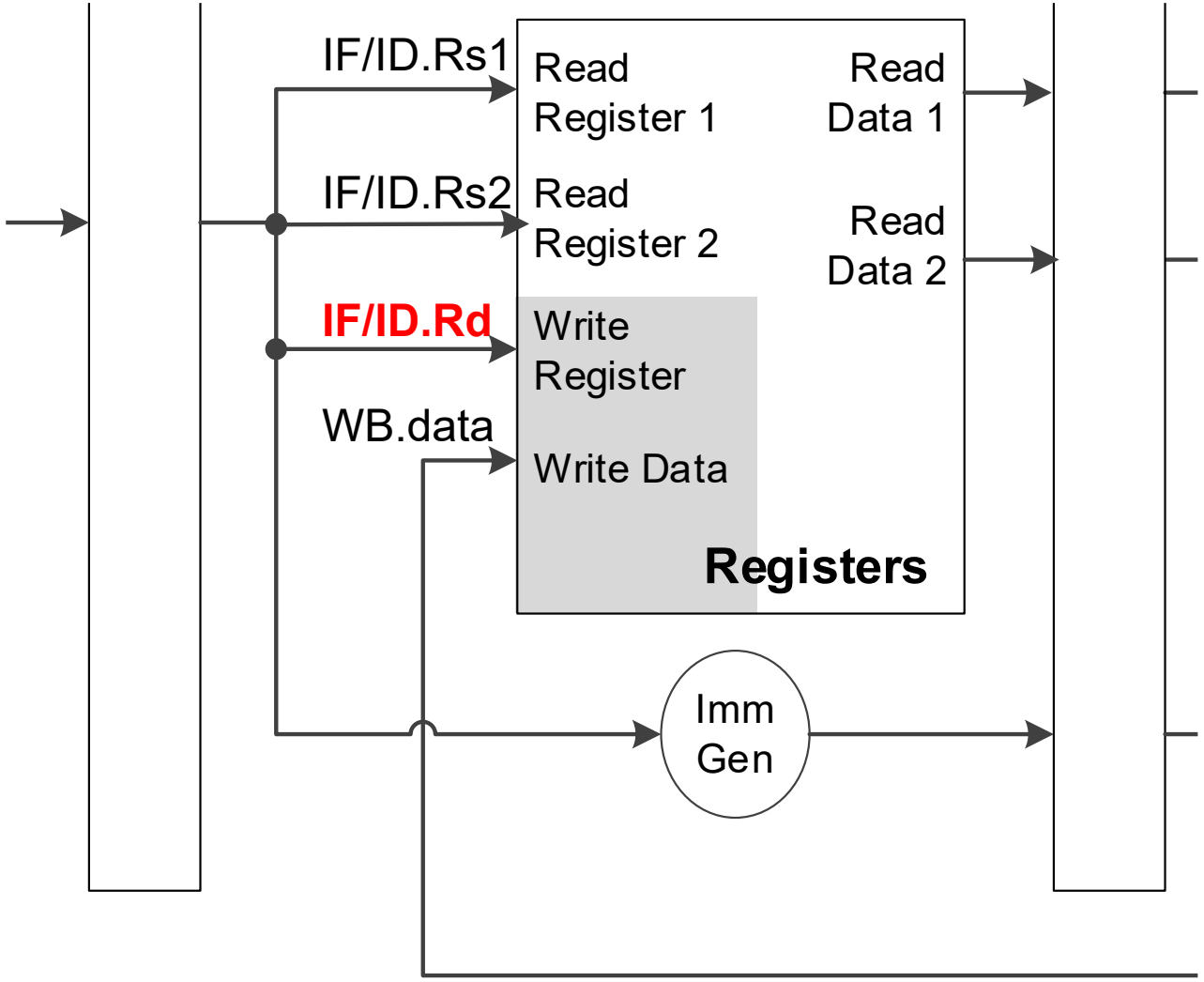
MEM stage


Inst4
Inst3
Inst2
Inst1
Add
4
0
M
U
X
1
PC
Read
Address
Instruction
Instruction
Memory
Read
Register 1
Read
Register 2
Write
Register
Write Data
Read
Data 1
Read
Data 2
Registers
Imm
Gen
ADD
ADD
Result
0
M
U
X
1
Zero?
ALU
ALU
Result
Address
Read
Data
Write
Data
Data
Memory
0
M
U
X
1

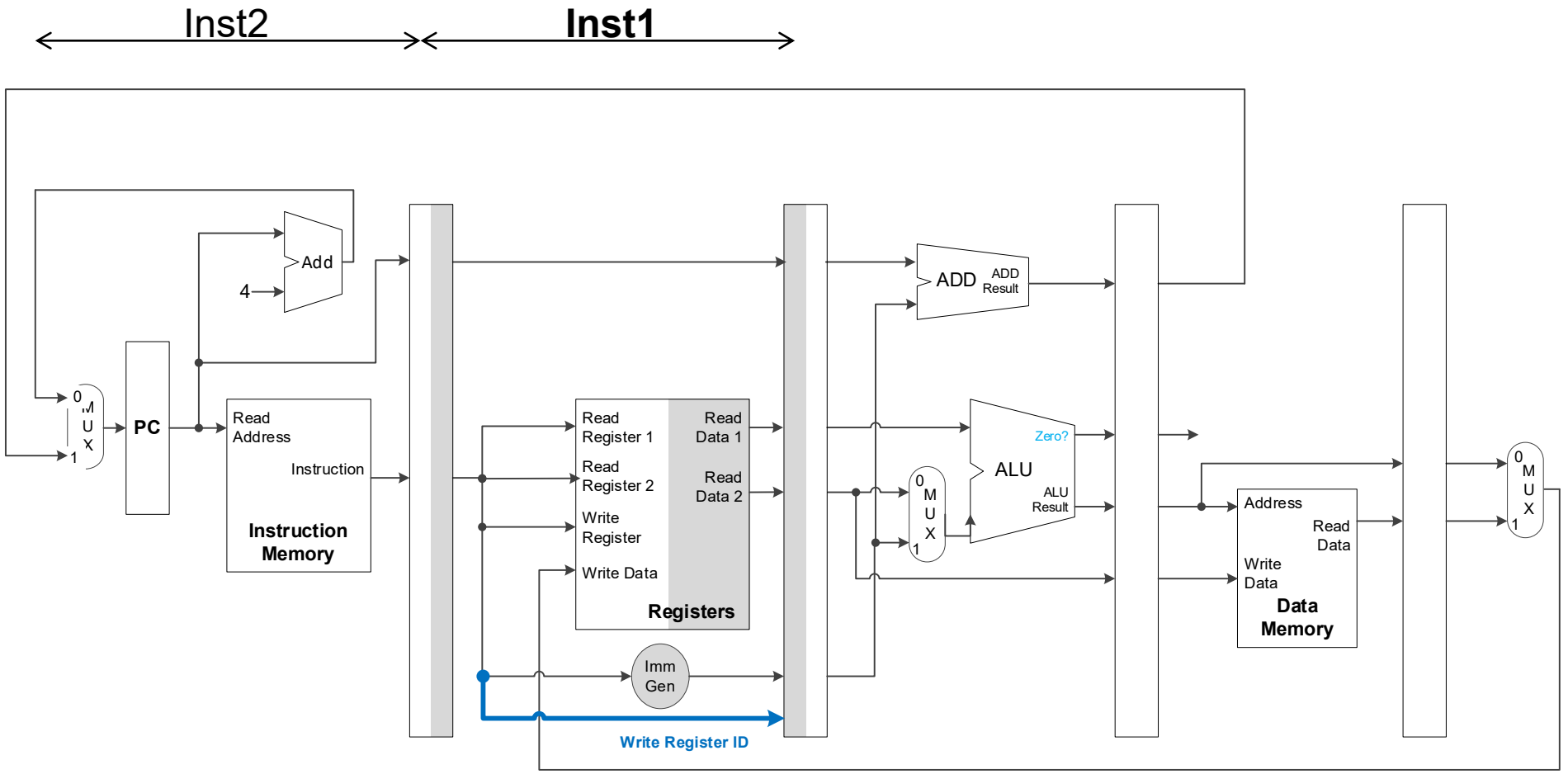
WB stage


Inst5
Inst4
Inst3
Inst2
Inst1
Add
4
M
U
X
PC
Read
Address
Instruction
Instruction
Memory
Read
Register 1
Read
Register 2
Write
Register
Write Data
Read
Data 1
Read
Data 2
Registers
Imm
Gen
ADD
ADD
Result
Zero?
ALU
ALU
Result
Address
Read
Data
Write
Data
Data
Memory
?

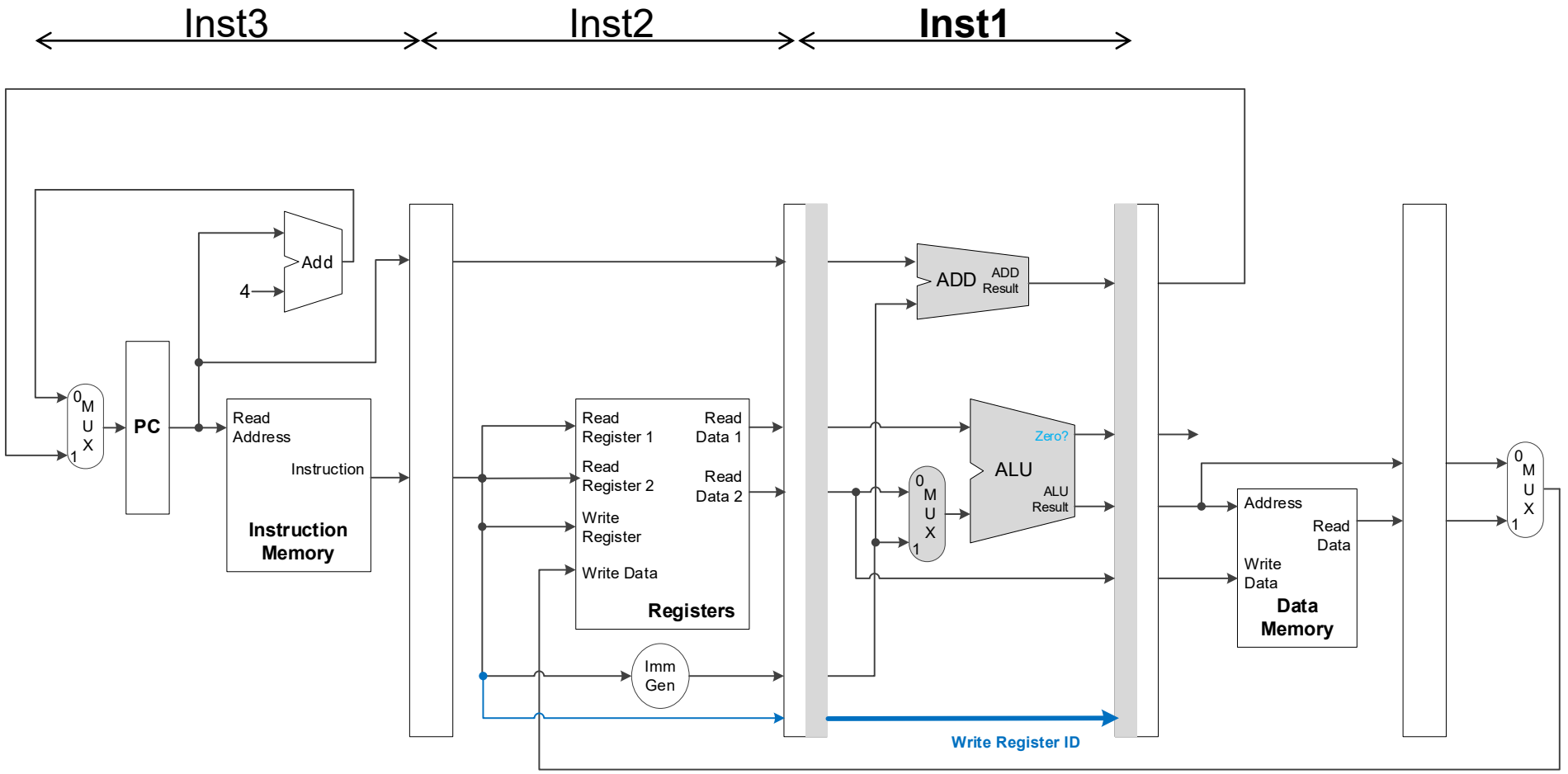
ID / WB stage


IF/ID.Rs1
Read
Register 1
Read
Data 1
IF/ID.Rs2
Read
Register 2
Read
Data 2
IF/ID.Rd
Write
Register
WB.data
Write Data
Registers
Imm
Gen

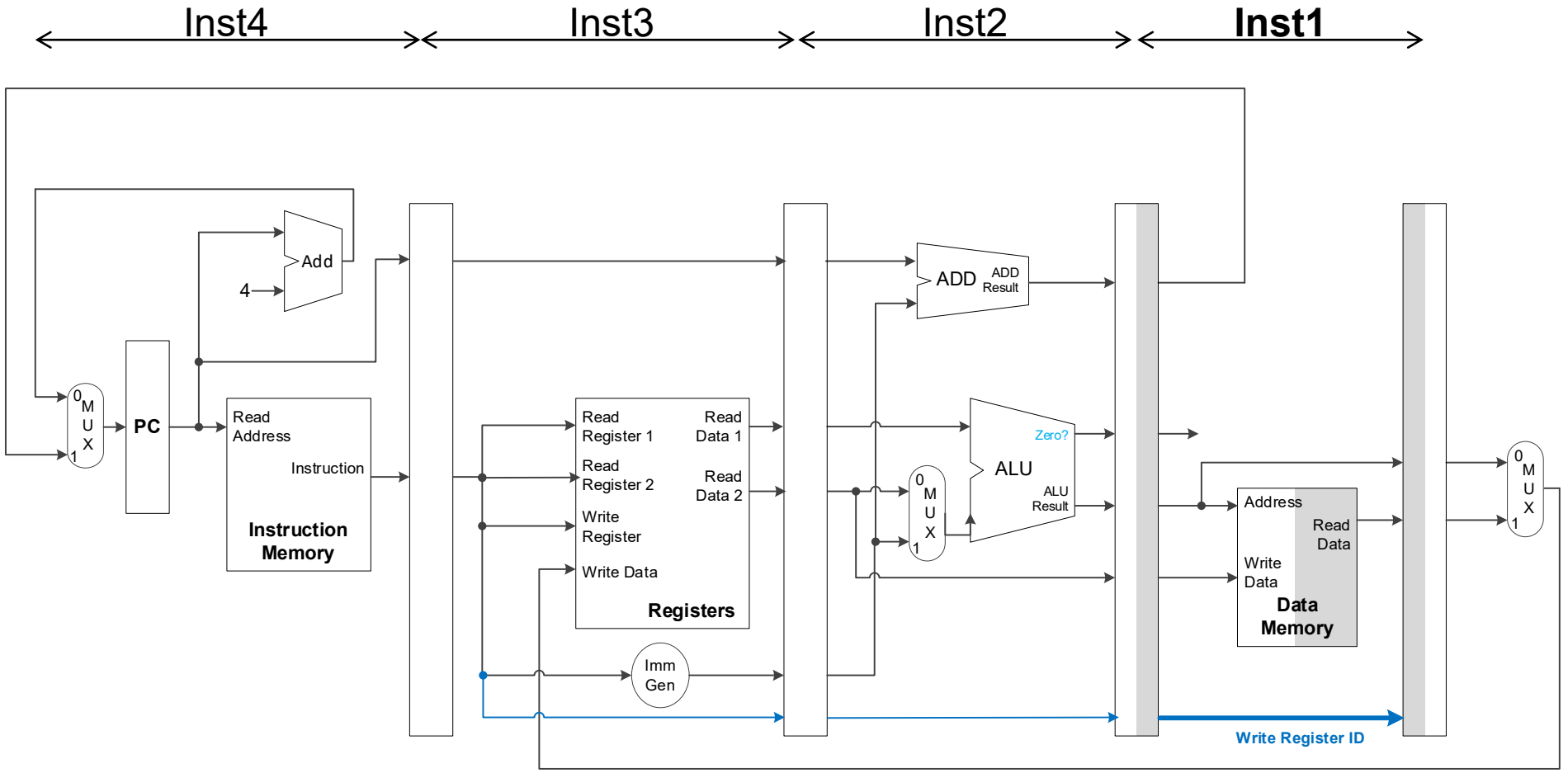
Corrected Datapath for Write Register ID



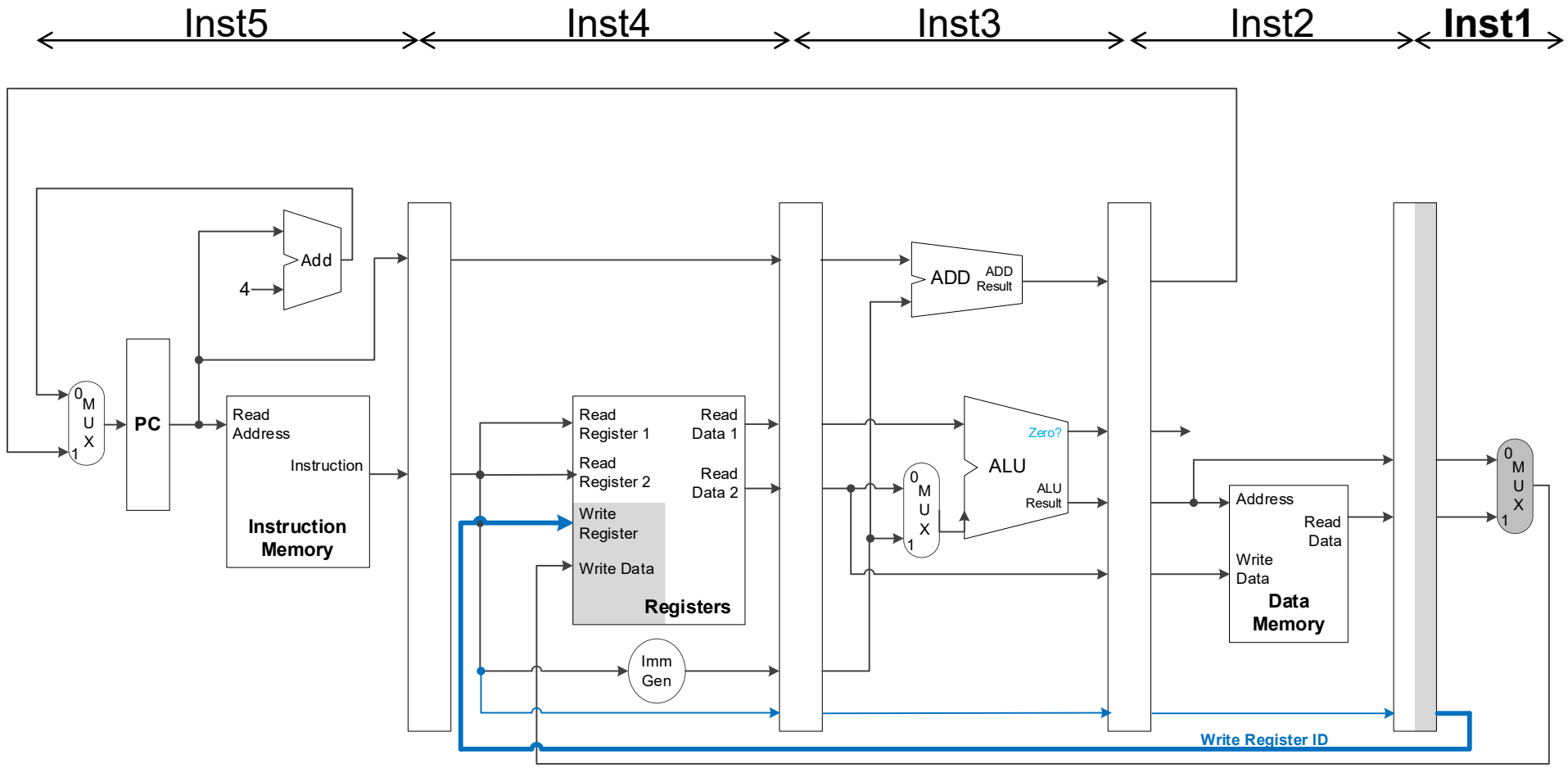
Corrected Datapath for Write Register ID



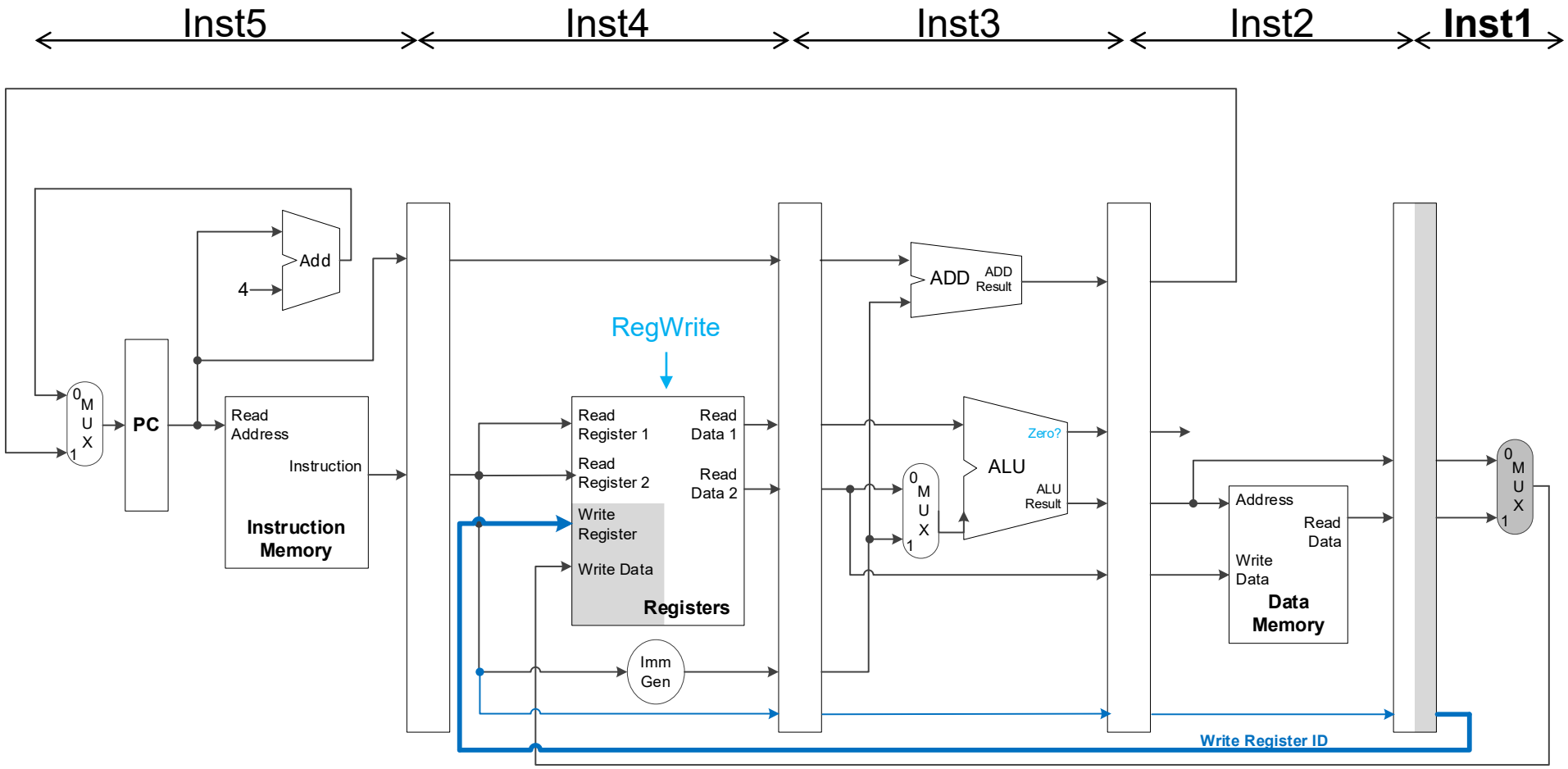
Corrected Datapath for Write Register ID



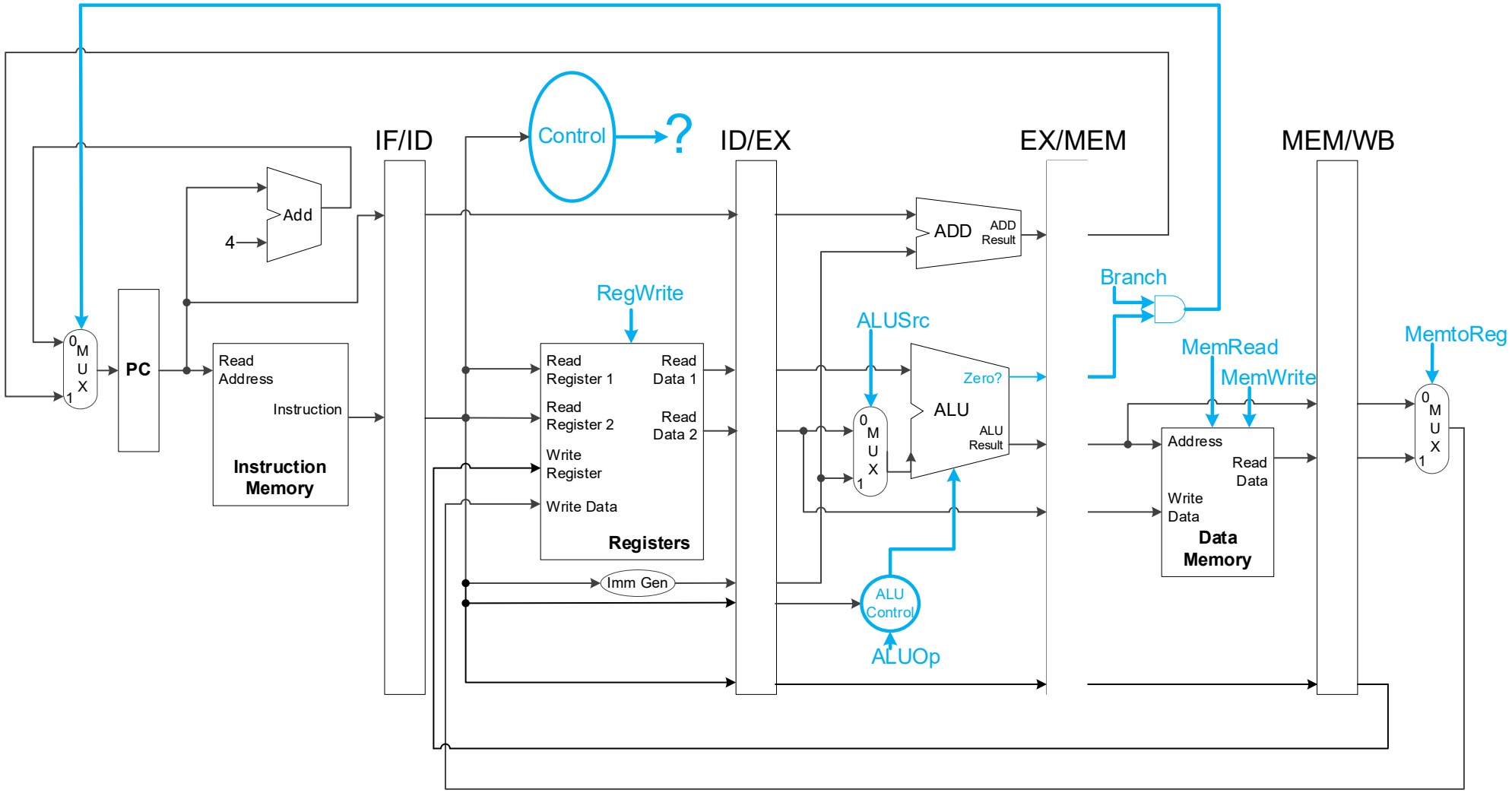
Corrected Datapath for Write Register ID



Control Signals?

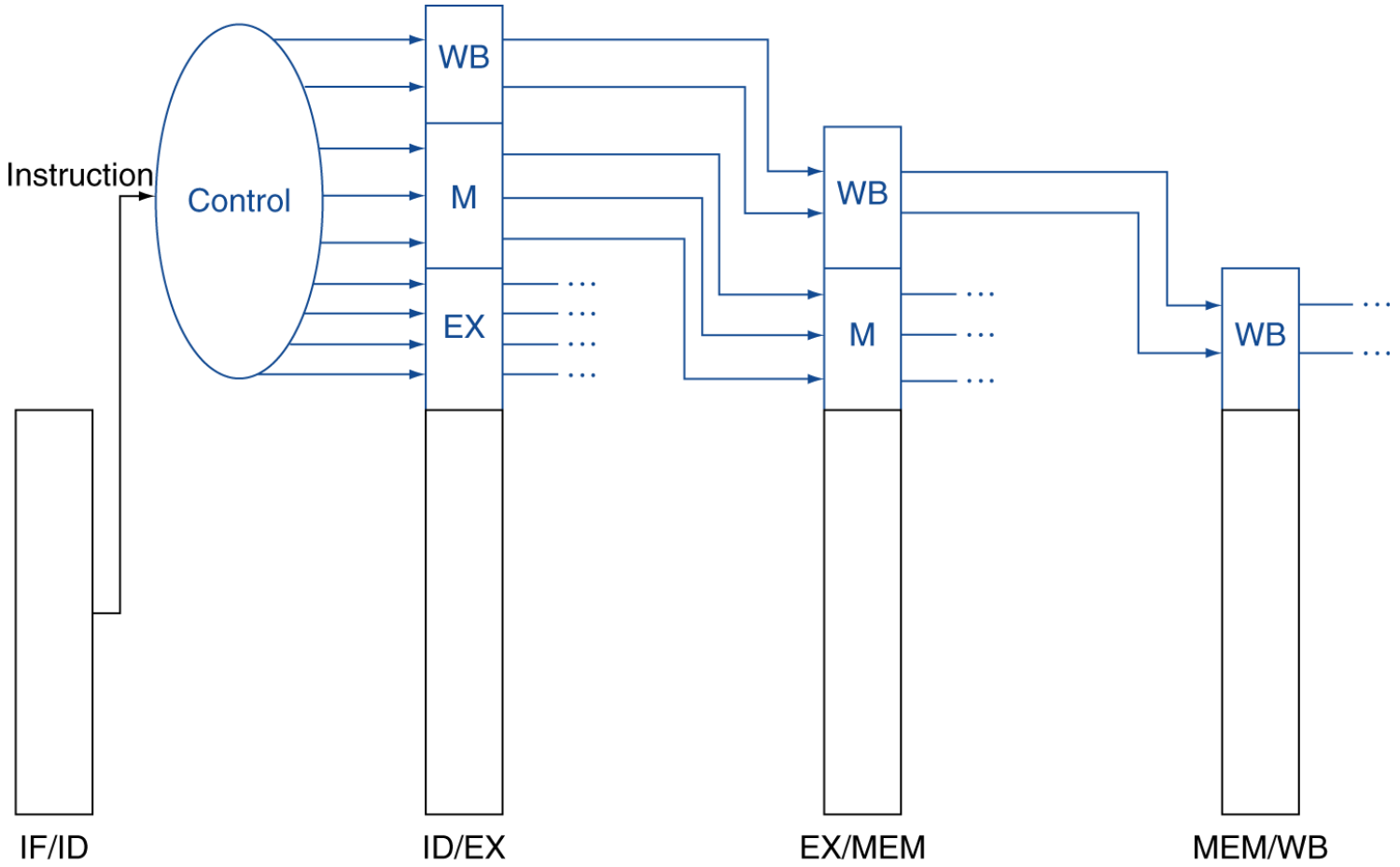


Control Signals in Pipeline



- Control signals derived from the opcode @ ID stage

Pipelined Control



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

Pipelined Control