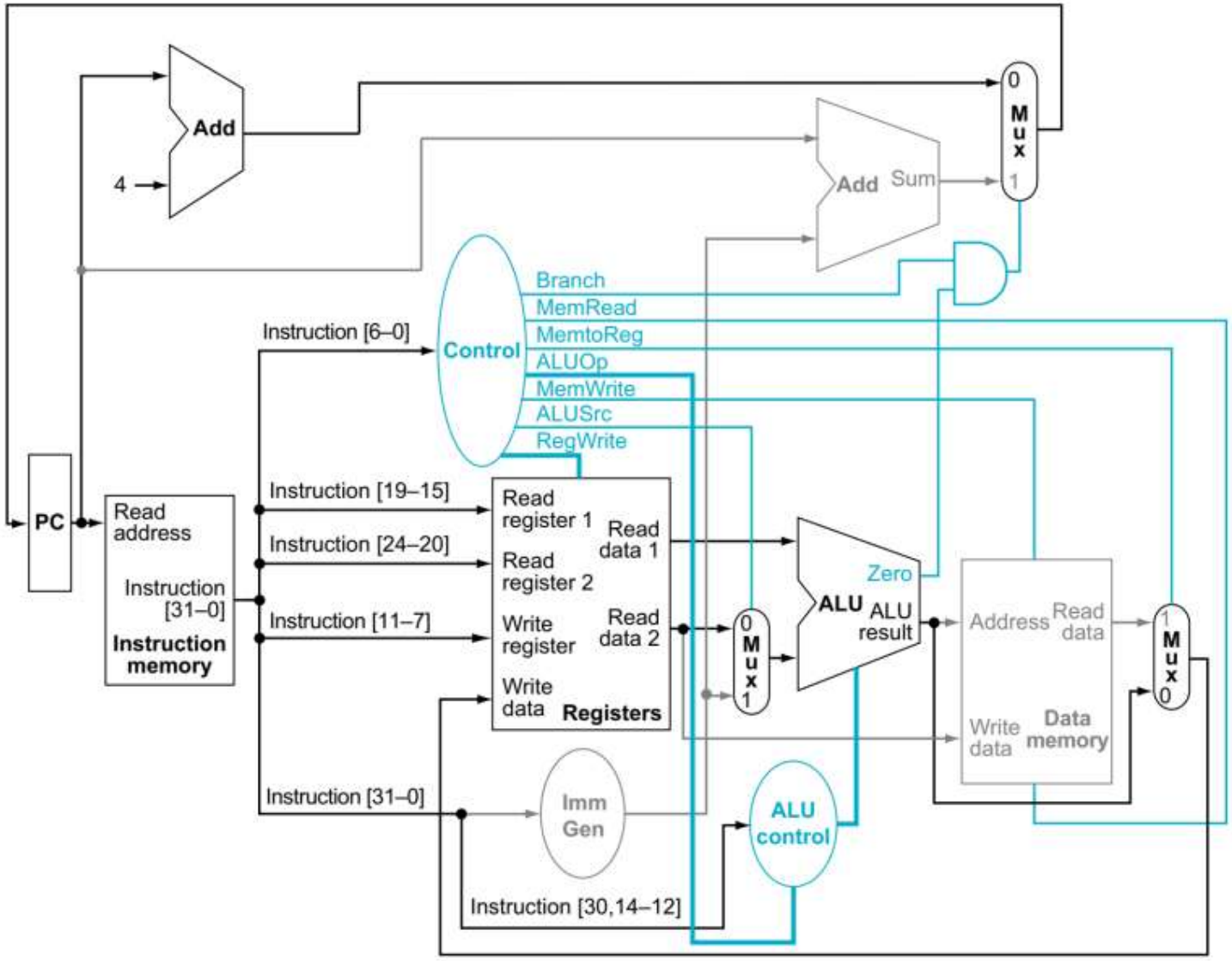
Ex1

```
AND Rd,Rs1, Rs2
#Reg[Rd] = Reg[Rs1] AND Reg[Rs2]
```

- Consider the instruction AND

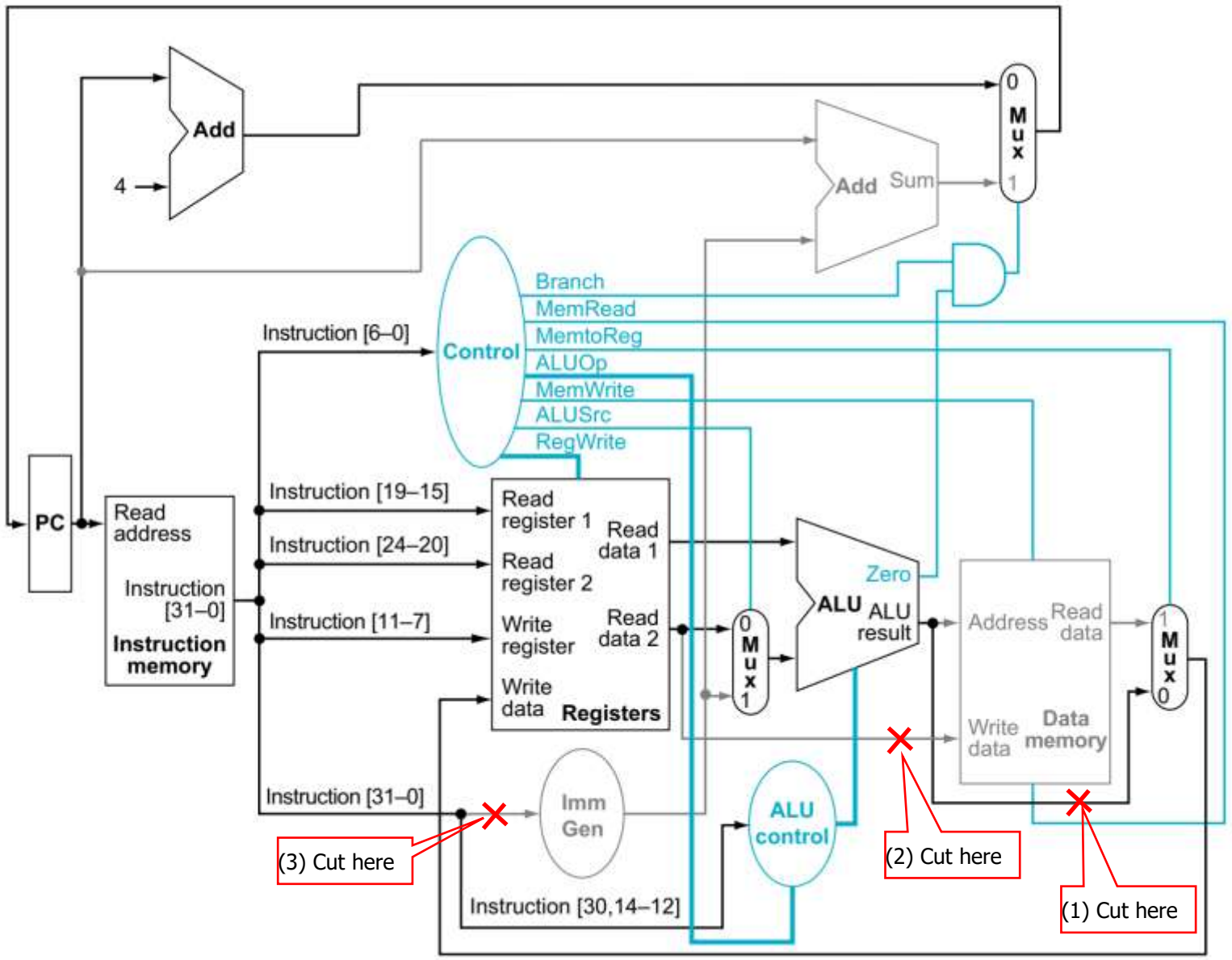
a) Show the values of control signals

b) Which blocks/components produce useful output for this instruction?



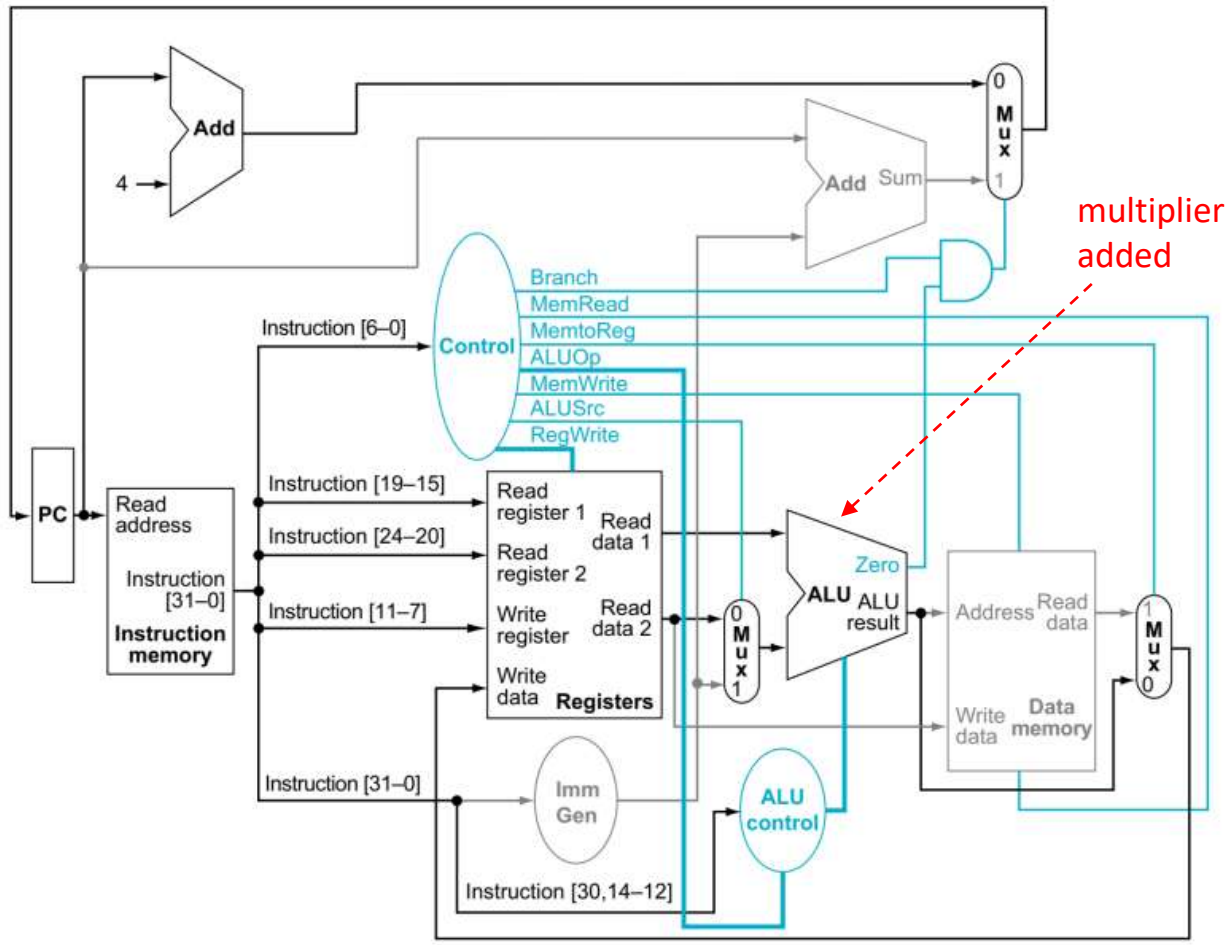
Ex2

- Consider the RISC-V datapath on the right. The lines marked with X may be cut individually.
- For each cut (1), (2), (3):
  - Show an instruction that will fail to execute if the line is cut. Explain.
  - Show an instruction that still works.



Ex3

- Given latencies and cost of main blocks in the table of a single-cycle CPU.
- The **mul**tiplier is added to ALU. It causes ALU latency increased by 300 and cost increased by 600, but instruction count decreased by 5%.
- Find:
  - Clock cycle time with/without MUL.
  - Is the CPU speed up or slow down after adding MUL?
  - Is adding MUL a good design choice?



Block	I-Mem	Add	Mux	ALU	Regs (R/W)	D-Mem	Control
Latency (ps)	400	100	30	120	100	350	100
Cost	1000	30	10	100	200	2000	500

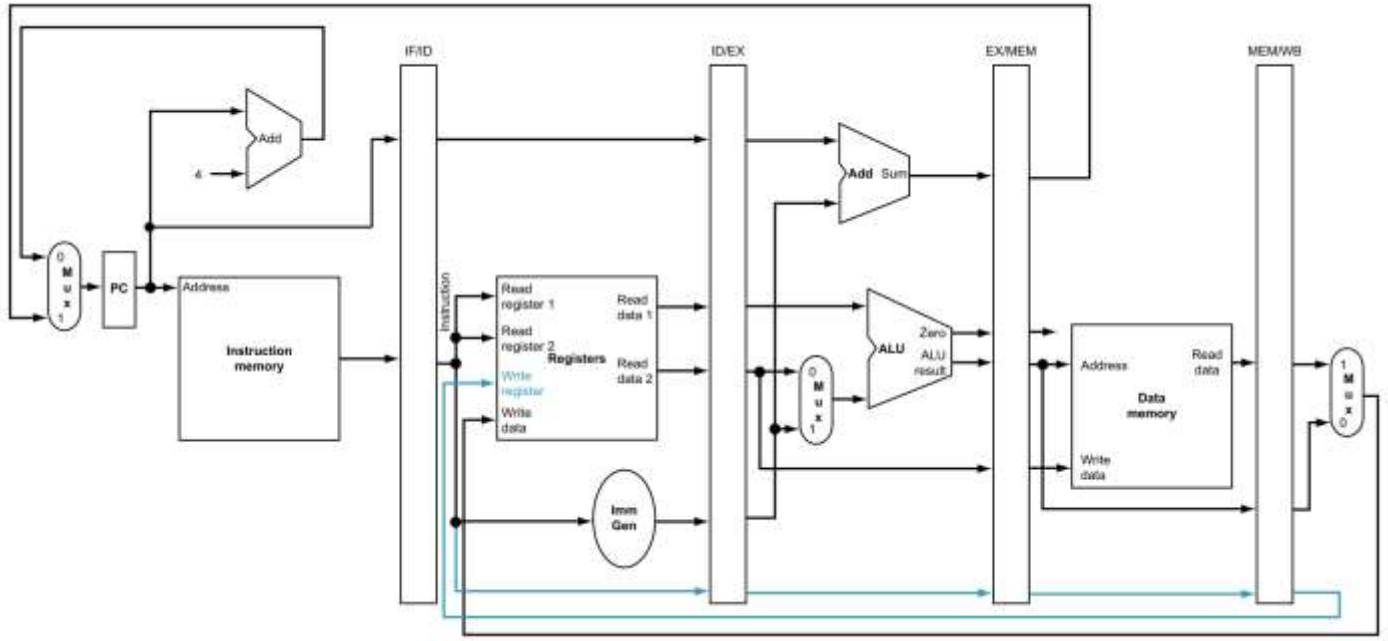
Ex4

- Given a pipelined CPU executing a program with breakdown of instructions as in the table.

add	addi	not	beq	lw	sw
20%	20%	0%	25%	25%	10%

a) In what fraction of all cycles is the data memory used?

b) In what fraction of all cycles is the output of the immediate generator (imm-gen) unit useful? Assume that there is no hazard.



Ex5

- Assume that individual stages of the datapath have the following latencies.

IF	ID	EX	MEM	WB
250ps	350ps	150ps	300ps	200ps

- And the breakdown of executed instructions are as follow

alu	beq	lw	sw
45%	20%	20%	15%

(a) What is the clock cycle time in a pipelined and non-pipelined processor?

(b) What is the total latency of an LW instruction in a pipelined and non-pipelined processor?

(c) What is the utilization of the instruction memory in a pipelined and non-pipelined processor, assuming there is no hazard?

(d) Consider a pipelined CPU with no stalls or hazards, what is the utilization of the data memory and write-register port of register file?

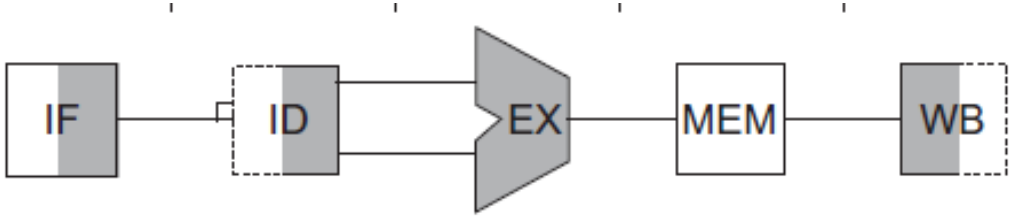
Ex6

Detect and determine type of hazards in the code below

a)
```
lw t0, 0(t0)
add t1, t0, t0
```

b)
```
add t1, t0, t0
addi t2, t0, 5
addi t4, t1, 4
```

Ex7



- Given the code fragment below.

```
add r1,r2,r3
add r2,r1,r4
add r1,r1,r2
```

a) Determine the potential hazard.

b) There are 3 pipeline CPU versions
  - CPU1: no forwarding path, CC = 250 ps
  - CPU2: full forwarding path, CC = 300 ps
  - CPU3: ALU-ALU forwarding only, CC = 290 ps
- Compare performance of the 3 CPU versions when executing the above code
- Which one is the fastest and what is its limitation?

Ex8

- Given the code fragment below. The latency of each pipeline stage is shown on the right.

a) What is clock cycle of this CPU?

b) Assume all branches are perfectly predicted. However, this CPU has only a single memory for both data and instructions. When structural hazard happens, the instructions accessing data will have higher priority. Draw the pipeline diagram showing execution of this program.

```
sw x16, 12(x6)
lw x16, 8(x6)
beq x5, x4, Label # Assume x5!=x4
add x5, x1, x4
slt x5, x15, x4
```

IF	ID	EX	MEM	WB
200ps	120ps	150ps	190ps	100ps

Ex9

- Given the code fragment below.

```
loop: lw x1, 0(x1)
and  x1, x1, x2
lw x1, 0(x1)
lw x1, 0(x1)
beq x1, x0, loop
```

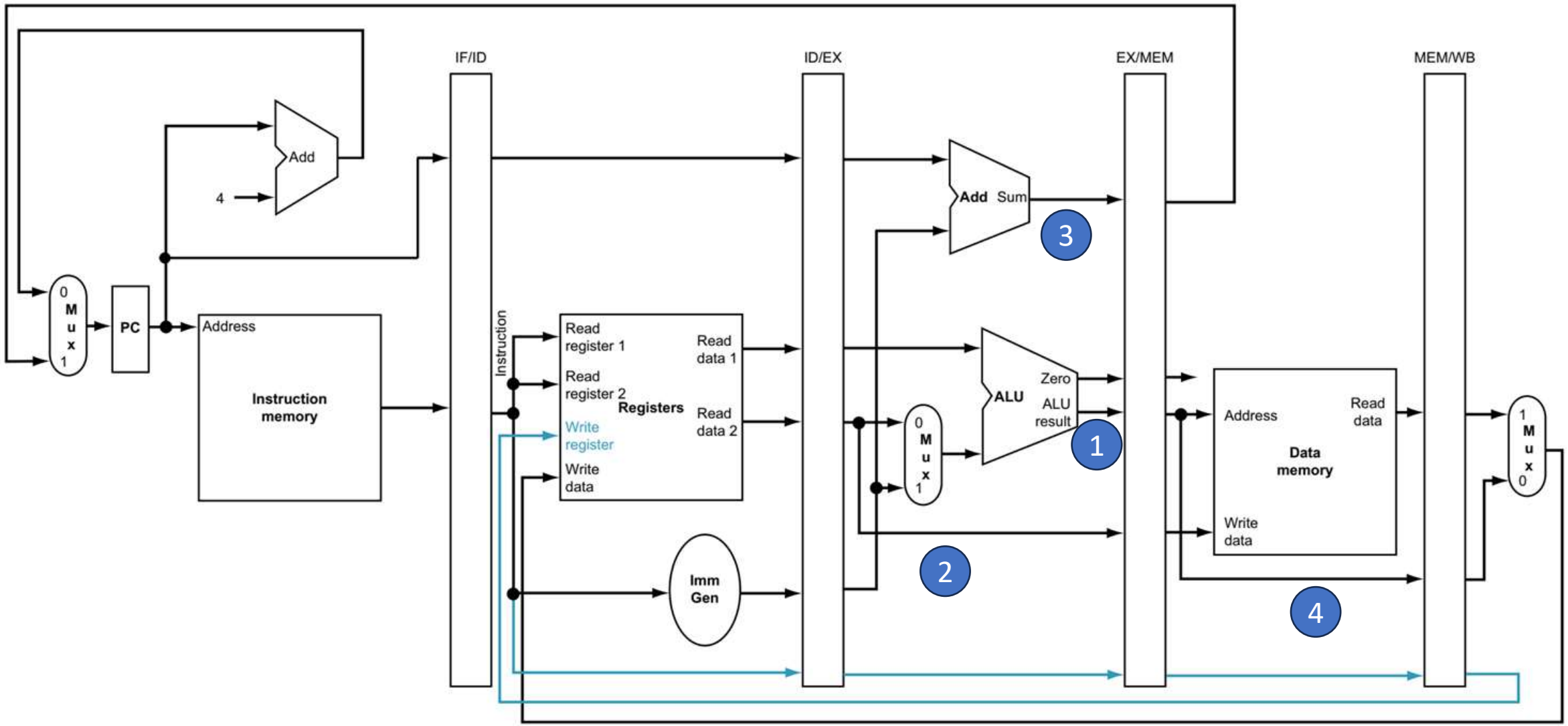
- Assume that perfect branch prediction is used, and pipeline has full forwarding and hazard detection.

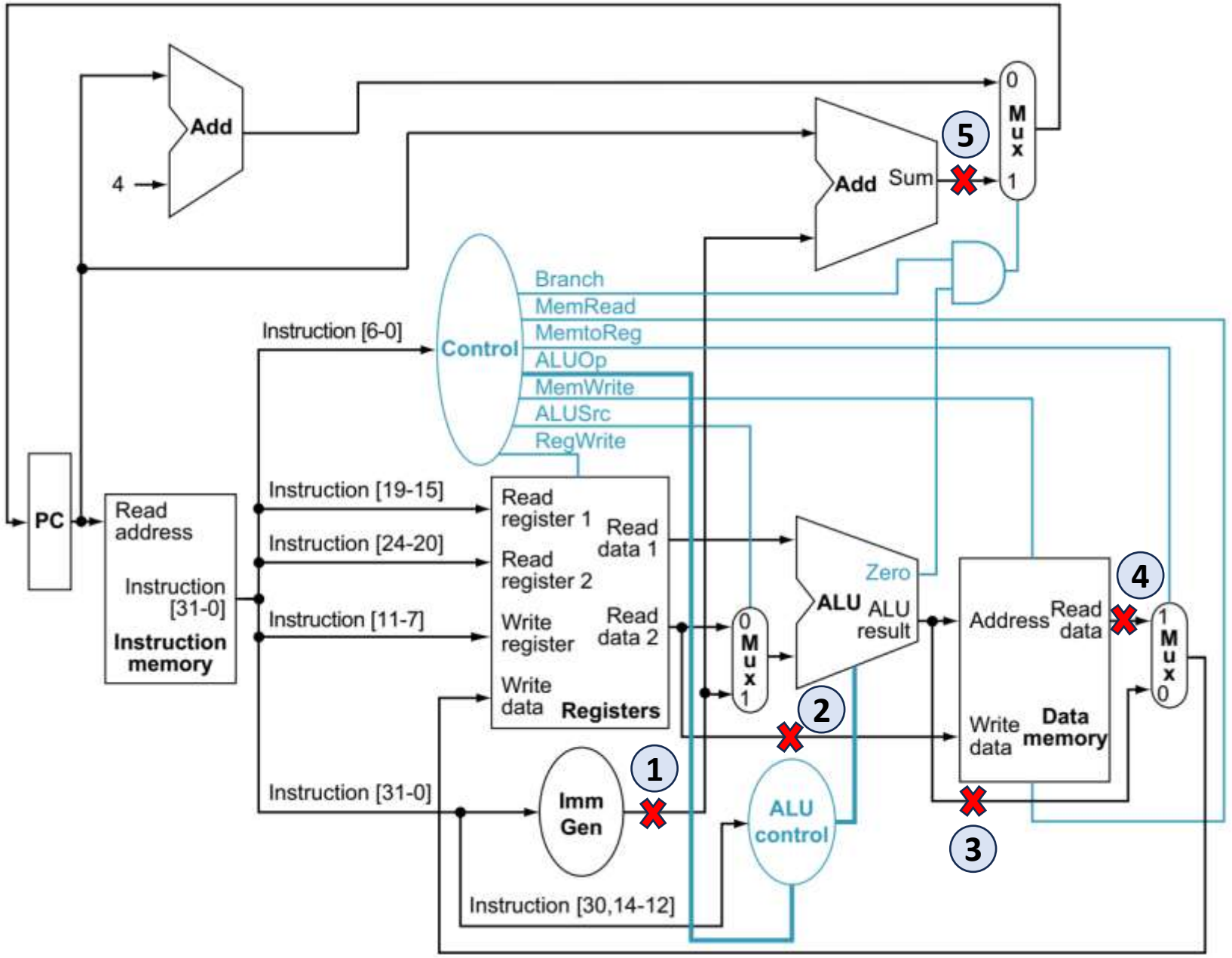
a) Show pipeline execution of the 2nd iteration.

b) Show the utilization of all 5 pipeline stages when the code is executed indefinitely?

What is the average CPI when CPU executes this loop?

Ex 10


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
Imm Gen
Add Sum
3
ALU
Zero
ALU result
1
2
Address
Read data
Data memory
Write data
4
1
M
u
x
0


Add
4
Add
Sum
0
M
u
x
1
5
Branch
MemRead
MemtoReg
ALUOp
MemWrite
ALUSrc
RegWrite
Control
Instruction [6-0]
PC
Read address
Instruction [31-0]
Instruction memory
Instruction [19-15]
Instruction [24-20]
Instruction [11-7]
Read register 1
Read register 2
Write register
Write data
Read data 1
Read data 2
Registers
0
M
u
x
1
Zero
ALU
ALU result
Address
Read data
Write data
Data memory
1
M
u
x
0
4
2
1
Instruction [31-0]
Imm Gen
ALU control
3
Instruction [30,14-12]