浙江大学
ZHEJIANG UNIVERSITY

Computer Architecture Experiments

Topic 1. Pipelined CPU supporting RISC-V RV32I Instructions

College of Computer Science & Technology

2022年9月

Outline

- Experiment Purpose
- Experiment Task
- Basic Principle
- Operating Procedures
- Checkpoints

Experiment Purpose

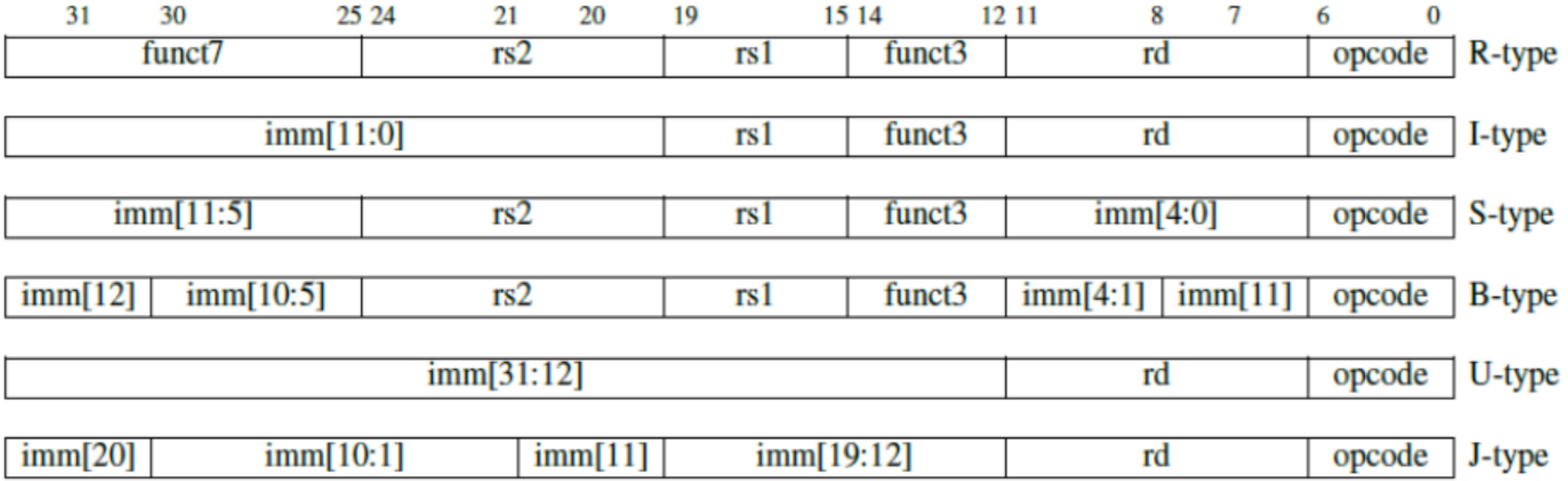
- Understand RISC-V RV32I instructions
- Master the design methods of pipelined CPU executing RV32I instructions
- Master the method of Pipeline Forwarding Detection and bypass unit design
- Master the methods of 1-cycle stall of Predict-not-taken branch design
- Master methods of program verification of Pipelined CPU executing RV32I instructions

Experiment Task

- **Design of Pipelined CPU executing RV32I instructions.**
  - Design datapath
  - Design Bypass Unit
  - Design CPU Controller

- **Verify the Pipelined CPU with program and observe the execution of program**

Pipelined CPU supporting execution of RV32I instructions



Pipelined CPU supporting execution of RV32I instructions

31 25	24 20	19 15	14 12	11 7	6 0	
imm[31:12]				rd	0110111	U lui
imm[31:12]				rd	0010111	U auipc
imm[20|10:1|11|19:12]				rd	1101111	J jal
imm[11:0]		rs1	000	rd	1100111	I jalr
imm[12|10:5]	rs2	rs1	000	imm[4:1|11]	1100011	B beq
imm[12|10:5]	rs2	rs1	001	imm[4:1|11]	1100011	B bne
imm[12|10:5]	rs2	rs1	100	imm[4:1|11]	1100011	B blt
imm[12|10:5]	rs2	rs1	101	imm[4:1|11]	1100011	B bge
imm[12|10:5]	rs2	rs1	110	imm[4:1|11]	1100011	B bltu
imm[12|10:5]	rs2	rs1	111	imm[4:1|11]	1100011	B bgeu

Pipelined CPU supporting execution of RV32I instructions

31 25	24 20	19 15	14 12	11 7	6 0	
imm[11:0]		rs1	000	rd	0000011	I lb
imm[11:0]		rs1	001	rd	0000011	I lh
imm[11:0]		rs1	010	rd	0000011	I lw
imm[11:0]		rs1	100	rd	0000011	I lbu
imm[11:0]		rs1	101	rd	0000011	I lhu
imm[11:5]	rs2	rs1	000	imm[4:0]	0100011	S sb
imm[11:5]	rs2	rs1	001	imm[4:0]	0100011	S sh
imm[11:5]	rs2	rs1	010	imm[4:0]	0100011	S sw

Pipelined CPU supporting execution of RV32I instructions

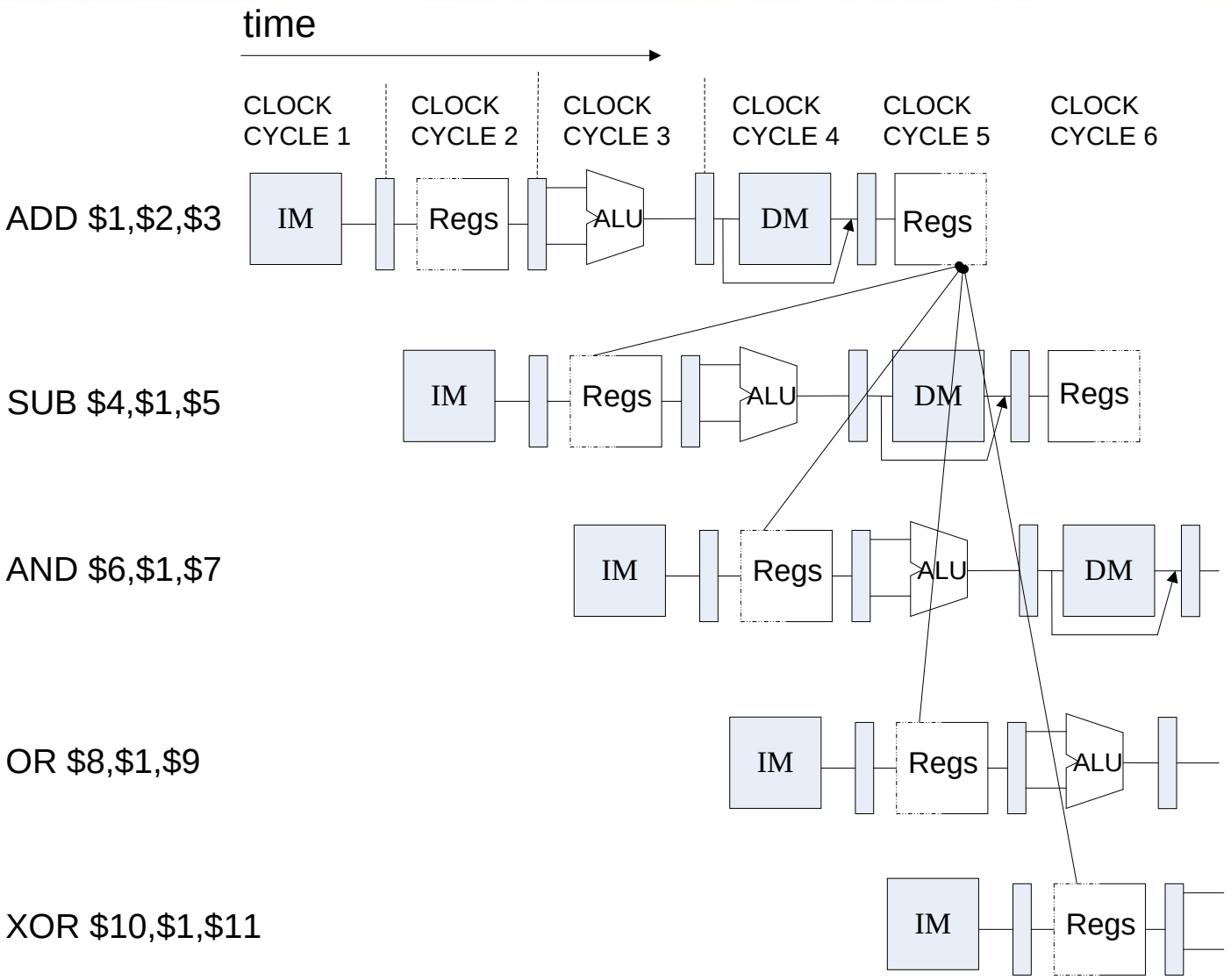
31 25	24 20	19 15	14 12	11 7	6 0	
imm[11:0]		rs1	000	rd	0010011	I addi
imm[11:0]		rs1	010	rd	0010011	I slti
imm[11:0]		rs1	011	rd	0010011	I sltiu
imm[11:0]		rs1	100	rd	0010011	I xori
imm[11:0]		rs1	110	rd	0010011	I ori
imm[11:0]		rs1	111	rd	0010011	I andi
0000000	shamt	rs1	001	rd	0010011	I slli
0000000	shamt	rs1	101	rd	0010011	I srli
0100000	shamt	rs1	101	rd	0010011	I srai

Pipelined CPU supporting execution of RV32I instructions

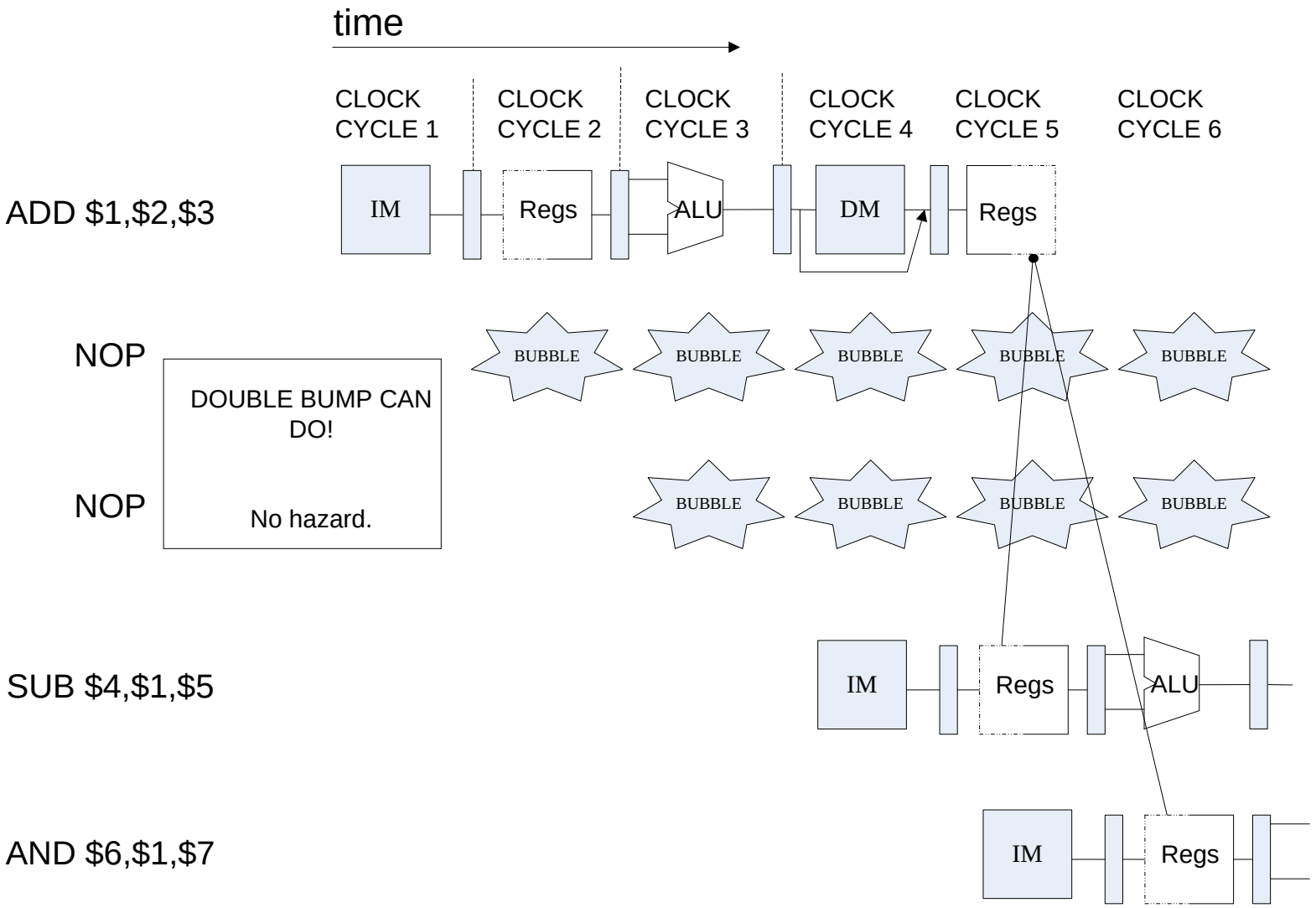
31 25	24 20	19 15	14 12	11 7	6 0	
0000000	rs2	rs1	000	rd	0110011	R add
0100000	rs2	rs1	000	rd	0110011	R sub
0000000	rs2	rs1	001	rd	0110011	R sll
0000000	rs2	rs1	010	rd	0110011	R slt
0000000	rs2	rs1	011	rd	0110011	Rsltu
0000000	rs2	rs1	100	rd	0110011	R xor
0000000	rs2	rs1	101	rd	0110011	R srl
0100000	rs2	rs1	101	rd	0110011	R sra
0000000	rs2	rs1	110	rd	0110011	R or
0000000	rs2	rs1	111	rd	0110011	R and

Instruction Demo

Data Hazard的条件：最多间隔一条指令

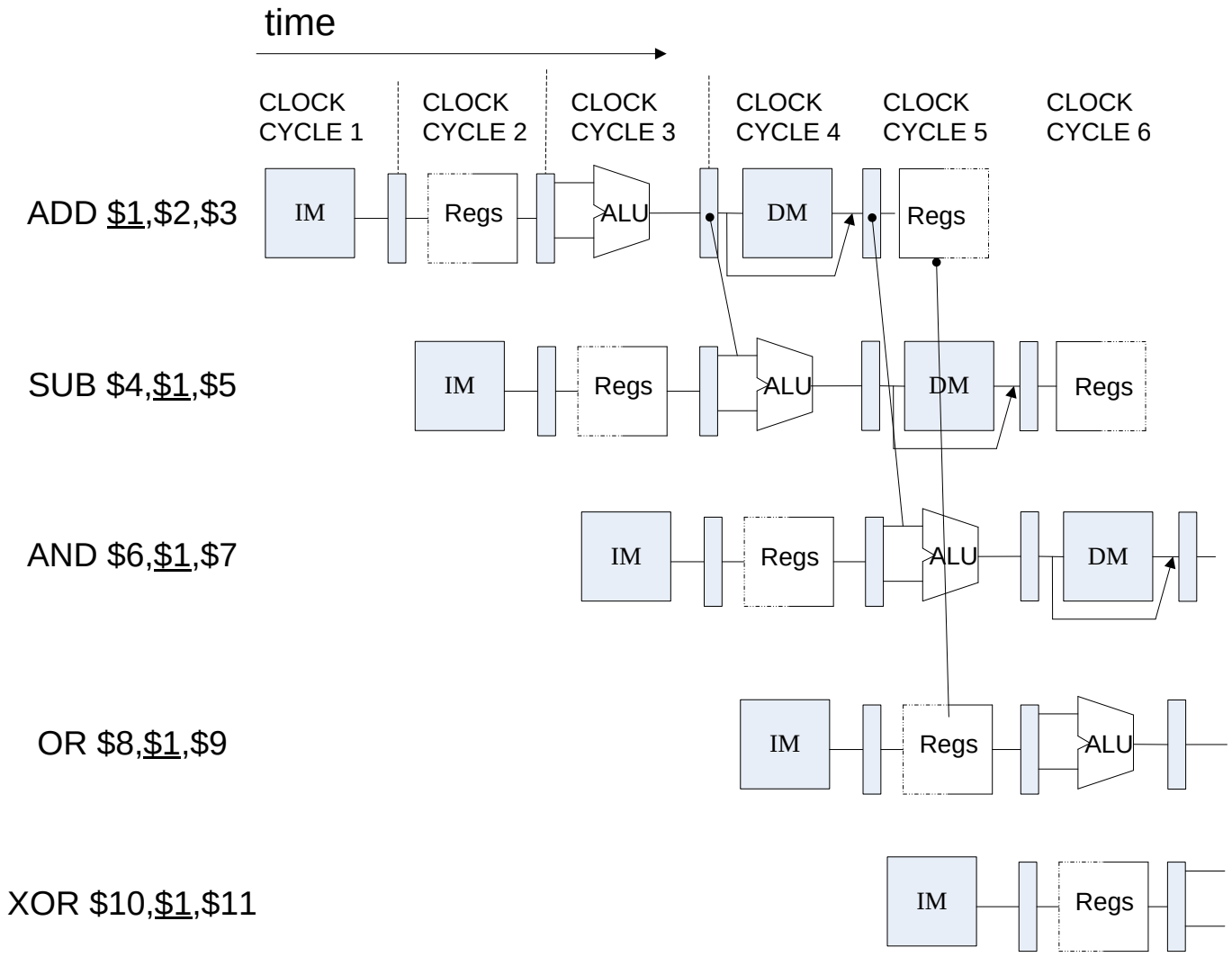


Data Hazard Causes Stalls



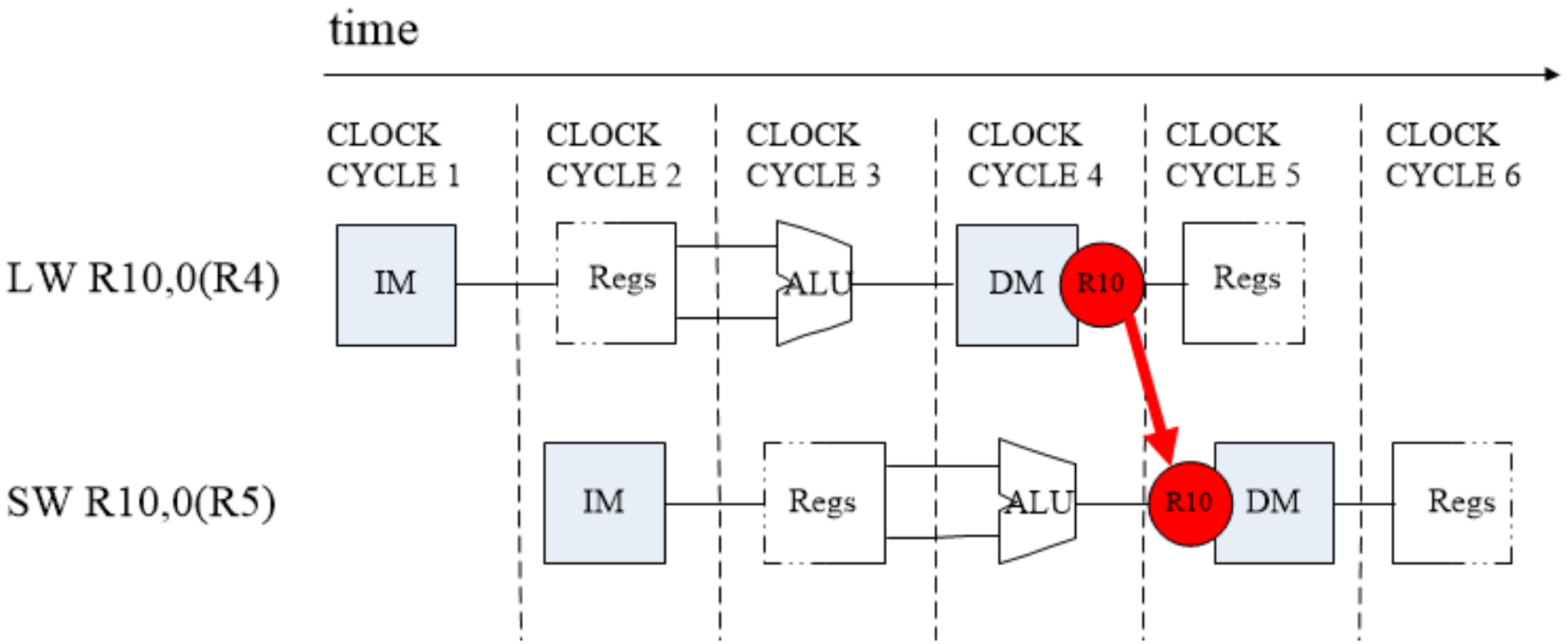
Pipeline Forward to Avoid the Data hazard

RR



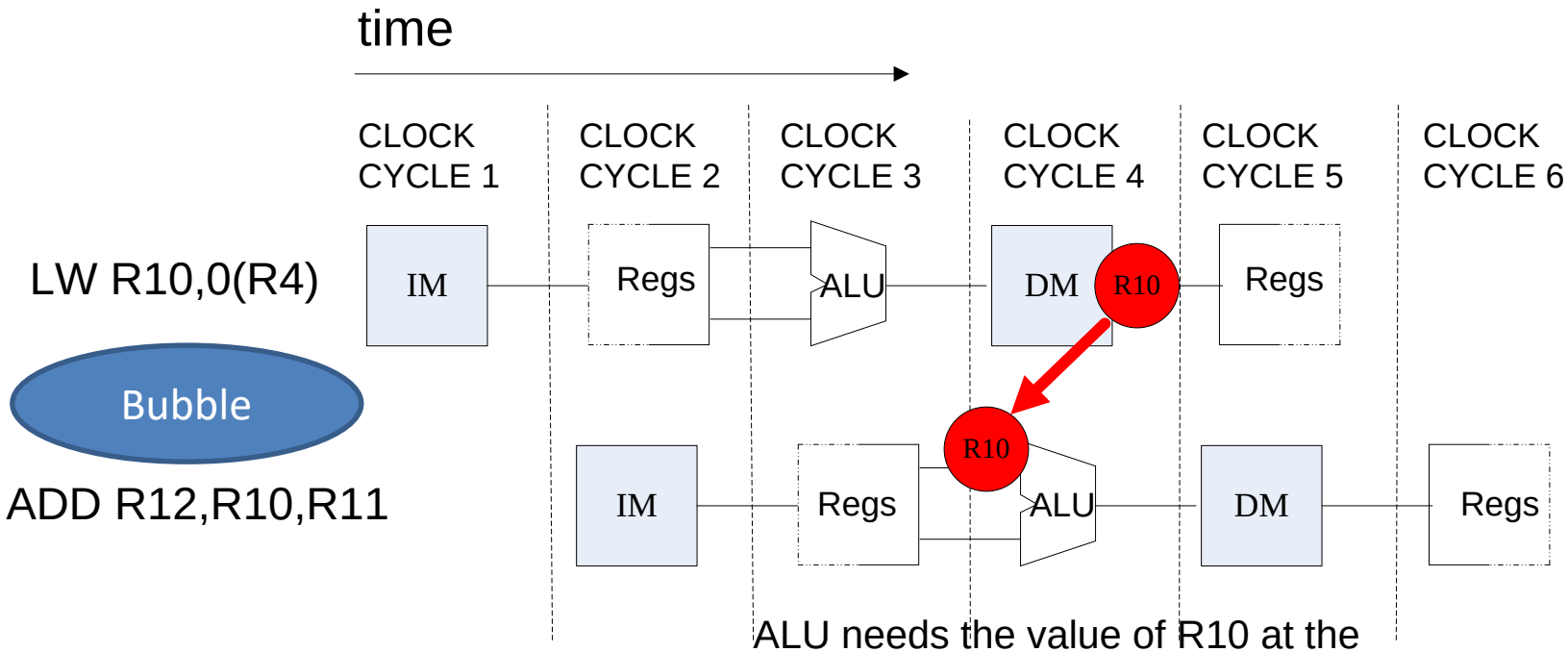
Special case: SW After LW

LS



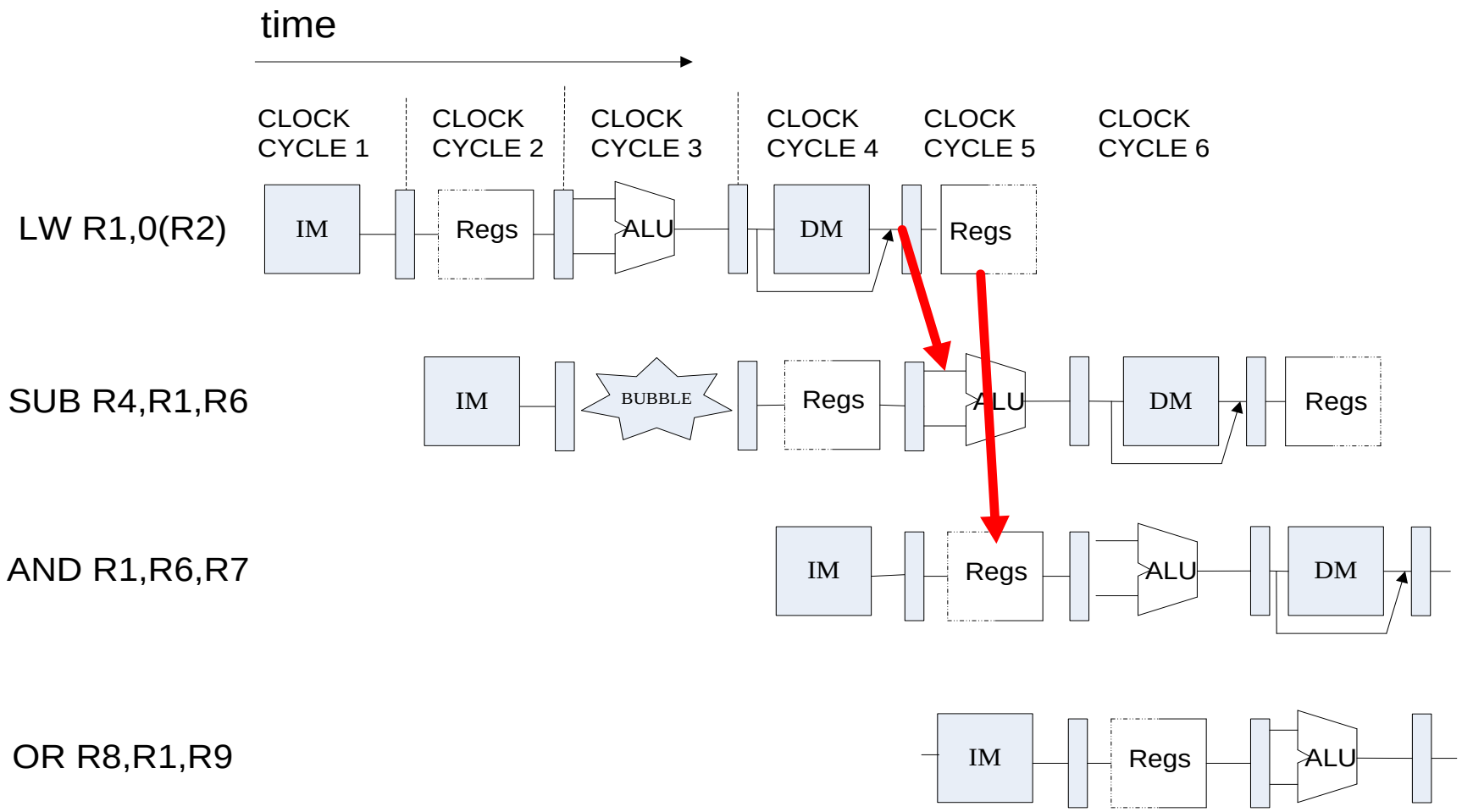
Condition in Which Bypass Unit doesn't work

LR



ALU needs the value of R10 at the beginning of Clock Cycle 4, but at the end of Clock Cycle 4 DM outputs the R10

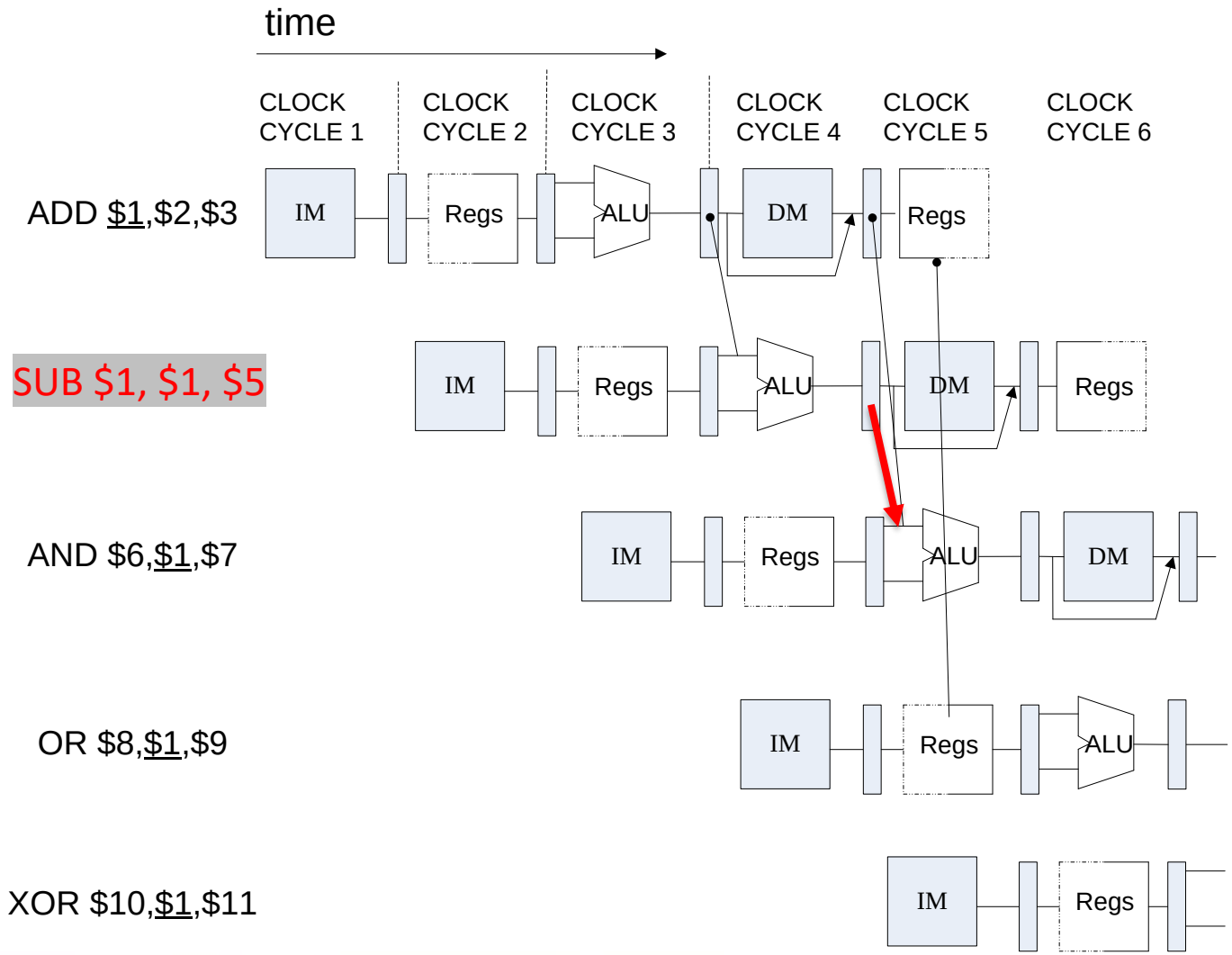
Pipeline Stalls



Double Data Hazard

```
1    add x1, x2, x3
2    add x1, x5, x1   # 采用此x1
3    add x6, x1, x1
```

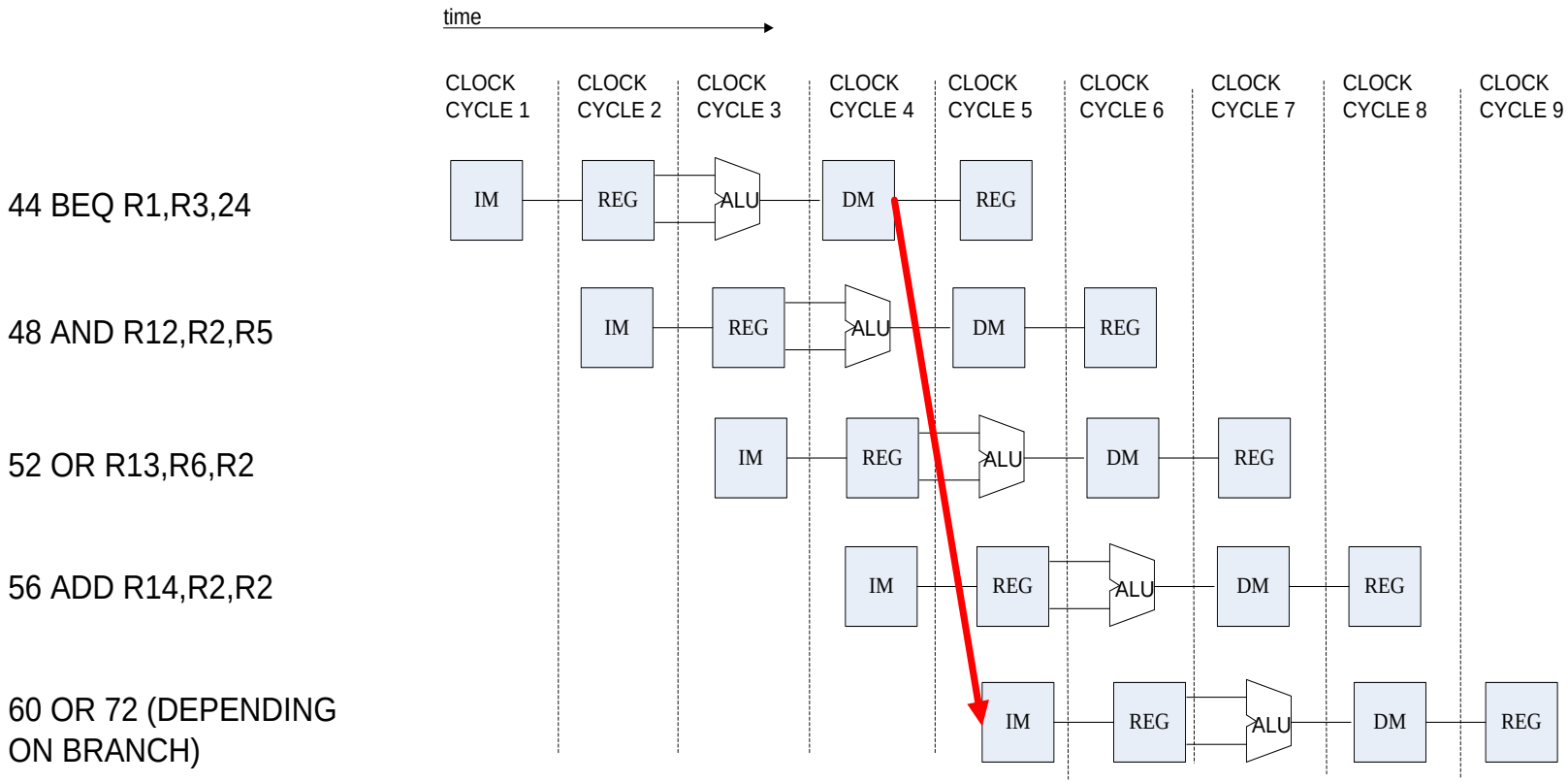
```
1    lw x1, 0(x6)
2    add x1, x3, x4   # 采用此x1
3    add x5, x1, x1
```



Instruction Demo

	Address	Instruction
	36	Nop
	40	Add R30,R30,R30
Branch to 72→	44	Beq R1,R3,24
if R1!=R3, it executes in sequence	48	And R12,R2,R5
	52	Or R13,R6,R2
	56	And R14,R2,R2
	60	
	64	
	68	
If R1=R3, it executes these instruction	72	Lw R14,50(R7)
	76	

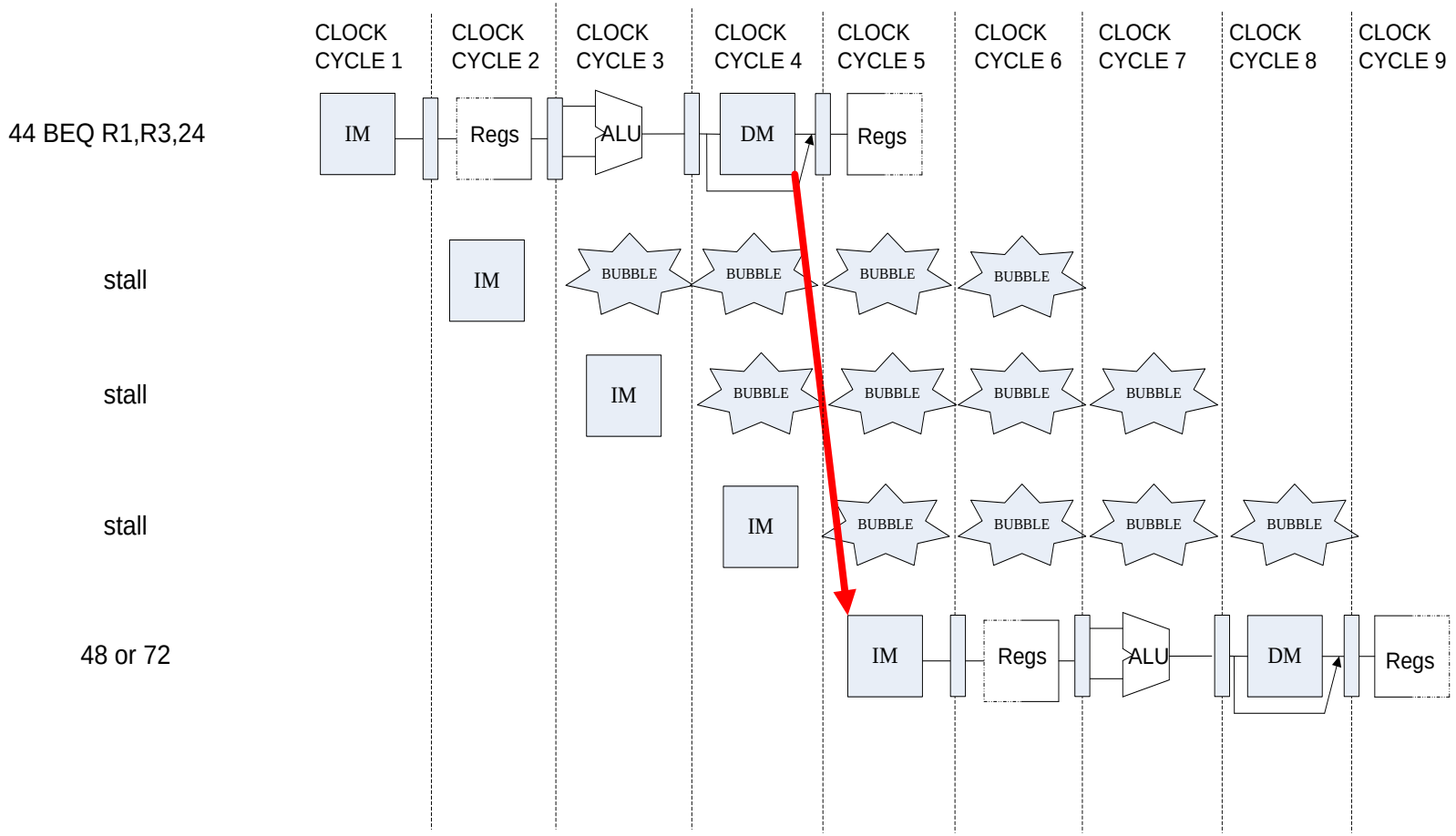
Execution result



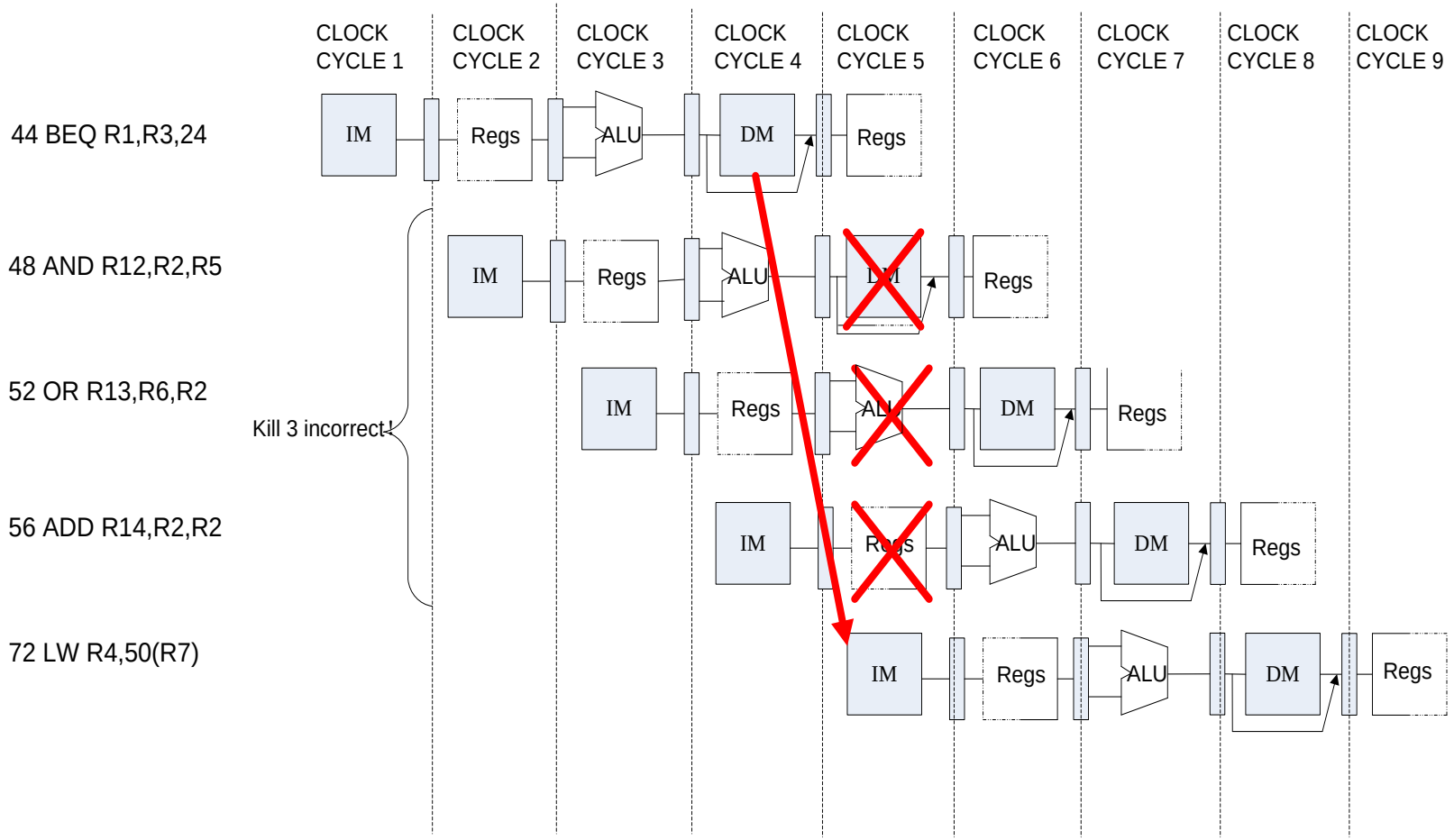
Methods of resolving Control hazards

- **Freeze or flush the pipeline**
- **Predict-not-taken**
- **Predict-taken**
- **Delayed branch**

Freeze method



Predict-not-taken



Predict-taken method

- **60% of branch result is taken**
- **Bring forward calculation of branch condition from MEM Stage to EX Stage, stall reduce from 3-cycle to 2-cycle.**
- **Then bring forward from EX to ID, stall reduce from 2-cycle to 1-cycle.**
- **1-cycle stall may be used for 1 delay slot**

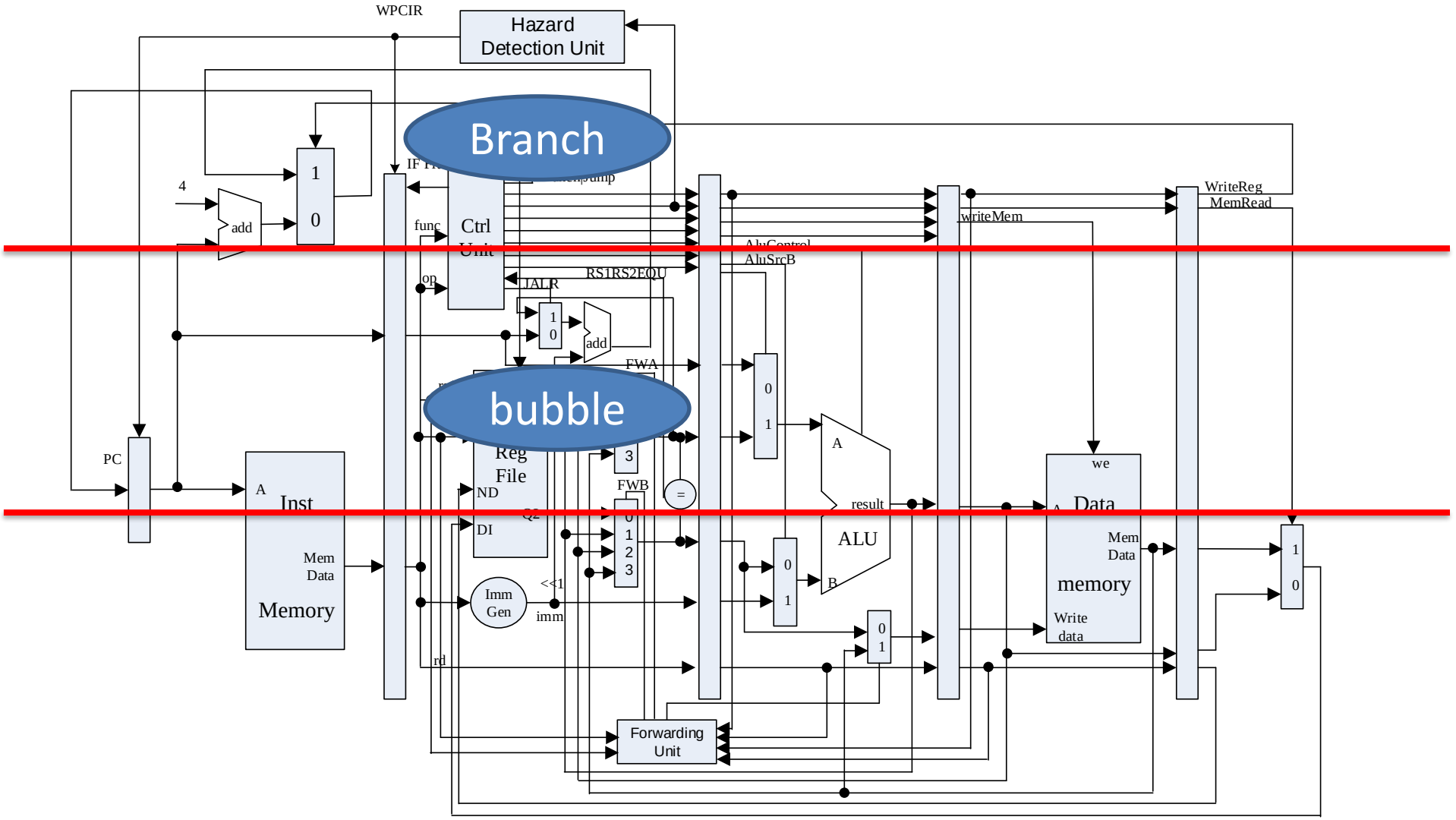
Stalls of Predict Methods for Control Hazard

Predict Methods	Status	Original Datapath	Address Calculation Forward	Condition Comparison Forward
Predict-taken	Hit (Need Branch)	3 stall	1 stall	1 stall
Predict-taken	Miss (No Branch)	3 stall	3 stall	1 stall
Predict-not-taken	Hit (No Branch)	go on	go on	go on
Predict-not-taken	Miss (Need Branch)	3 stall	3 stall	1 stall

Control Hazard

t0

t1



Data Hazard & Control Hazard in the meantime

RB

```
1   add x1, x2, x3
2   beq x1, x2, label_1
```

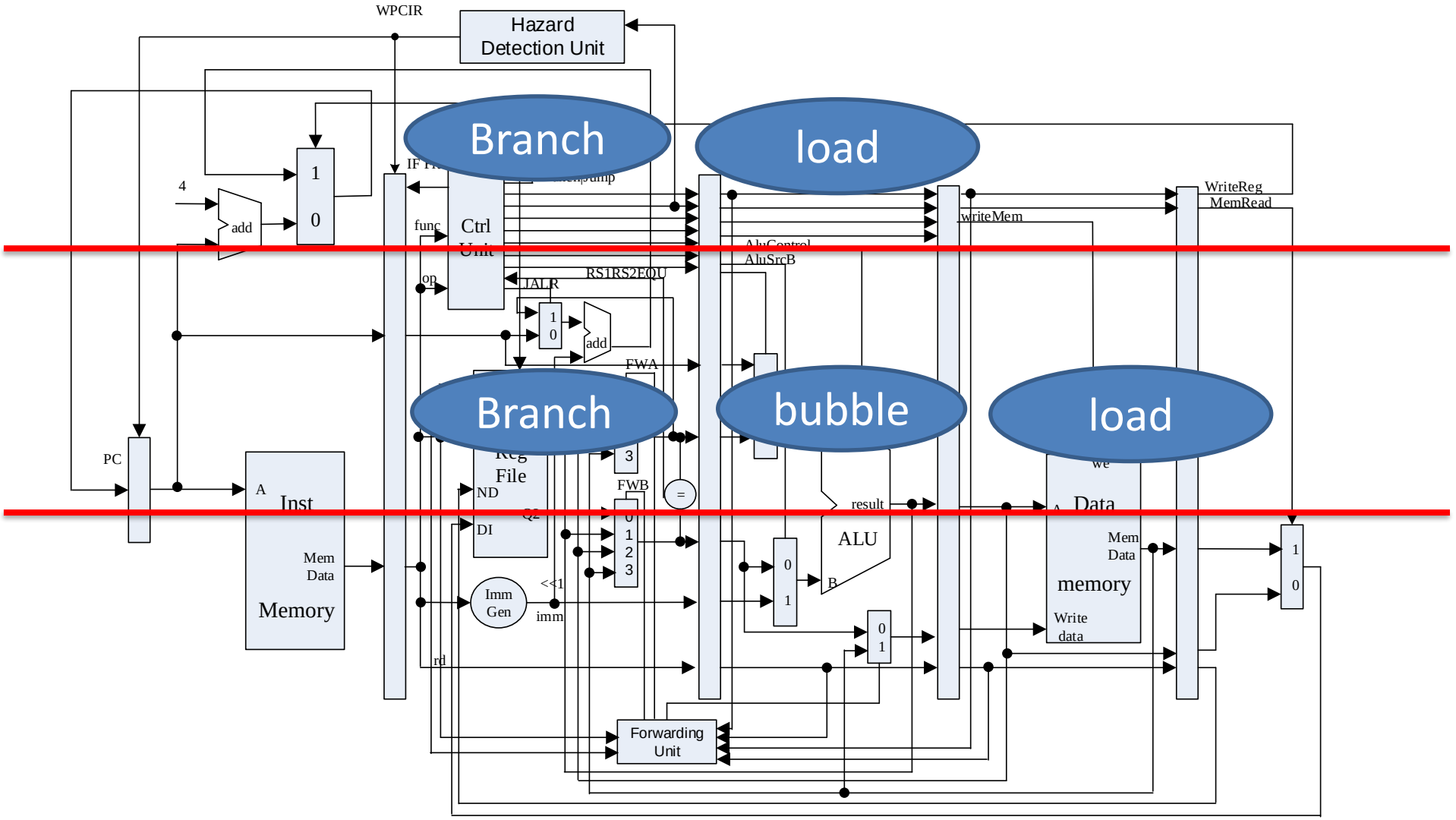
LB

```
1   lw x1, 0(x6)
2   jal x1, 12
```

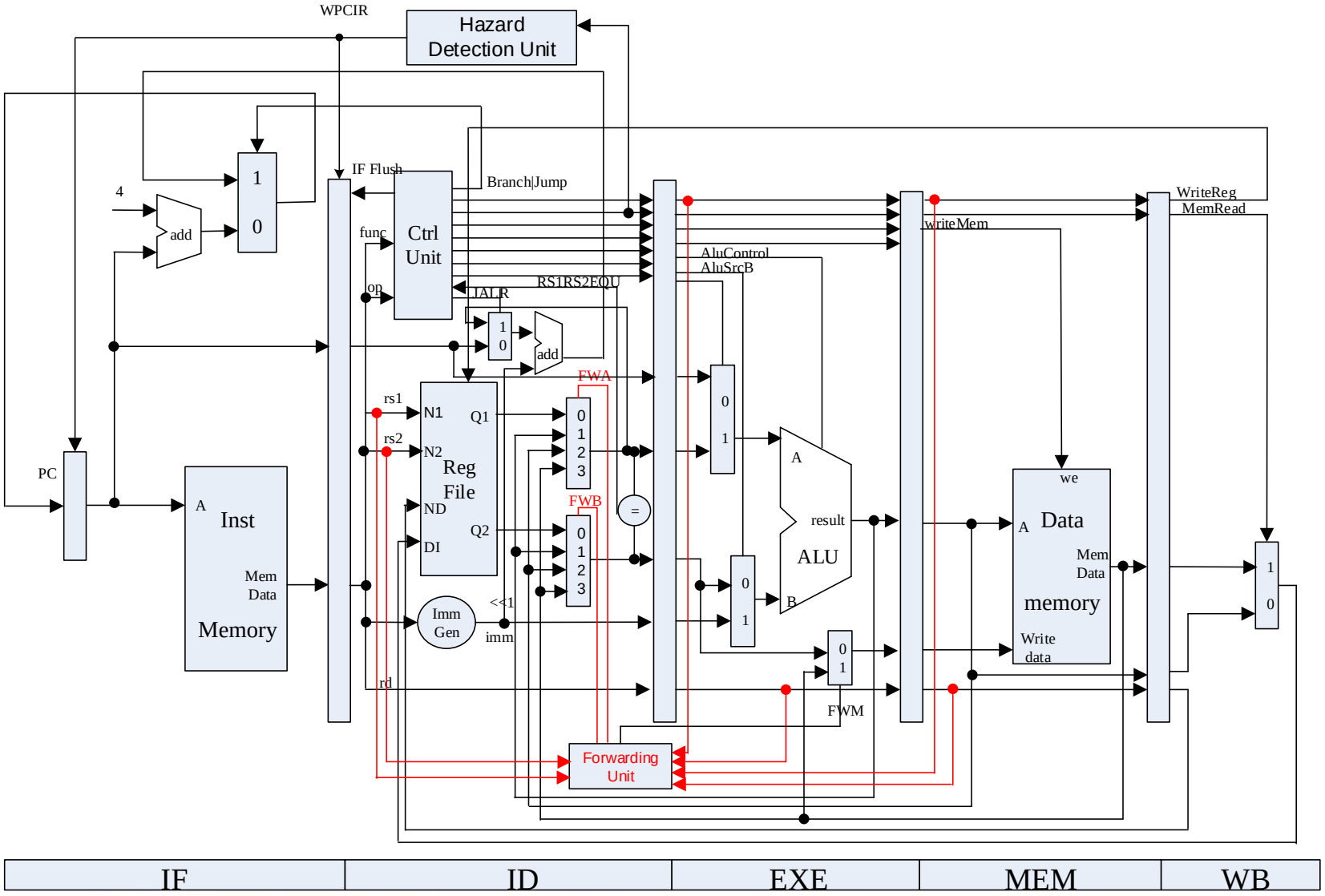
Data Hazard & Control Hazard in the meantime

t0

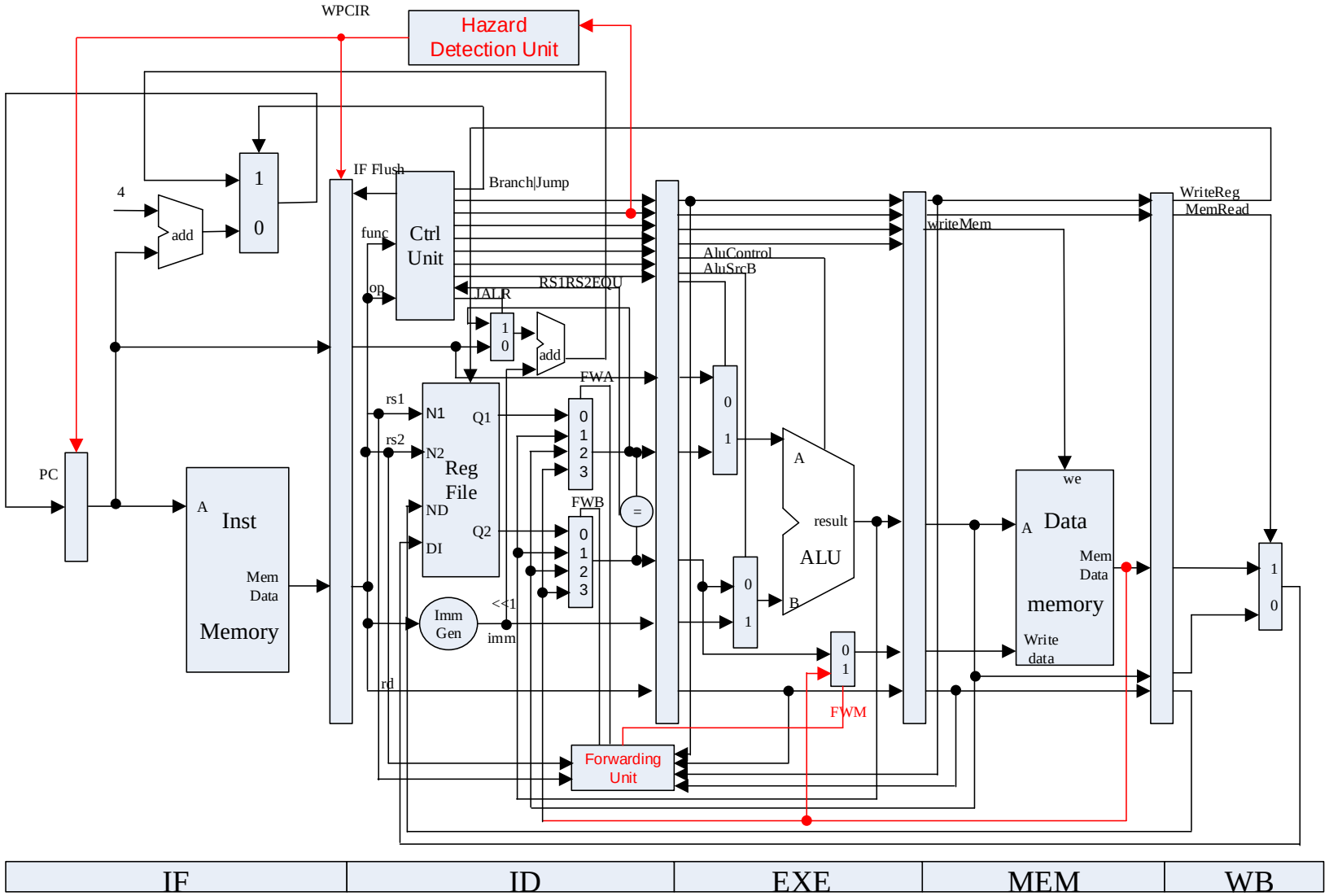
t1



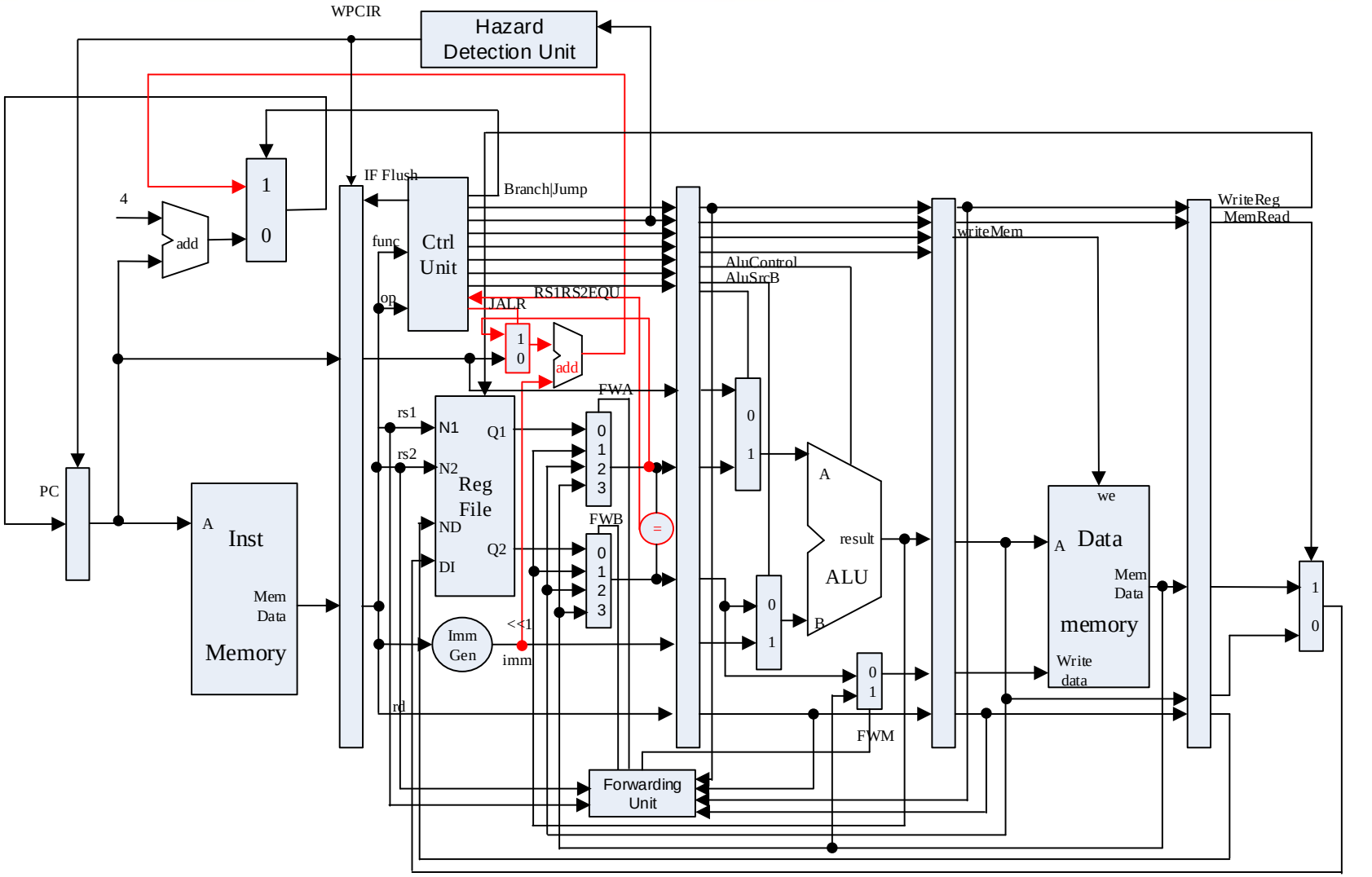
Pipelined CPU supporting RV32I instructions



Pipelined CPU supporting RV32I instructions



Pipelined CPU supporting RV32I instructions



Instr. Mem.(1)

NO.	Instruction	Addr.	Label	ASM	Comment
0	00000013	0	__start:	addi x0, x0, 0	
1	00402103	4		lw x2, 4(x0)	
2	00802203	8		lw x4, 8(x0)	
3	004100b3	C		add x1, x2, x4	
4	fff08093	10		addi x1, x1, -1	
5	00c02283	14		lw x5, 12(x0)	
6	01002303	18		lw x6, 16(x0)	
7	01402383	1C		lw x7, 20(x0)	
8	402200b3	20		sub x1,x4,x2	
9	002270b3	24		and x1,x4,x2	
10	002260b3	28		or x1,x4,x2	
11	002240b3	2C		xor x1,x4,x2	
12	002210b3	30		sll x1,x4,x2	
13	002220b3	34		slt x1,x4,x2	
14	004120b3	38		slt x1,x2,x4	

Instr. Mem.(2)

NO.	Instruction	Addr.	Label	ASM	Comment
15	002350b3	3C		srl x1, x6, x2	
16	402350b3	40		sra x1, x6, x2	
17	4023d0b3	44		sra x1, x7, x2	
18	007330b3	48		sltu x1, x6, x7	
19	0063b0b3	4C		sltu x1, x7, x6	
20	00000033	50		add x0,x0,x0	
21	ffd50093	54		addi x1,x10,-3	
22	00f27093	58		andi x1,x4,15	
23	00f26093	5C		ori x1,x4,15	
24	00f24093	60		xori x1,x4,15	
25	00f22093	64		slti x1,x4,15	
26	00121093	68		slli x1,x4,1	
27	00225093	6C		srli x1,x4,2	
28	40c35093	70		srai x1, x6, 12	
29	fff33093	74		sltiu x1, x6, -1	

Instr. Mem.(3)

NO.	Instruction	Addr.	Label	ASM	Comment
30	fff3b093	78		beq x4,x5,label0	
31	00520863	7C		beq x4,x4,label0	
32	00420663	80		addi x0,x0,0	
33	00000013	84		addi x0,x0,0	
34	00000013	88	label0:	bne x4,x4,label1	
35	00421863	8C		bne x4,x5,label1	
36	00521663	90		addi x0,x0,0	
37	00000013	94		addi x0,x0,0	
38	00000013	98	label1:	blt x5,x4,label2	
39	0042c863	9C		blt x4,x5,label2	
40	00524663	A0		addi x0,x0,0	
41	00000013	A4		addi x0,x0,0	
42	00000013	A8	label2:	bltu x6,x7,label3	
43	00736863	AC		bltu x7,x6,label3	
44	0063e663	B0		addi x0,x0,0	

Instr. Mem.(4)

NO.	Instruction	Addr.	Label	ASM	Comment
45	00000013	B4		addi x0,x0,0	
46	00000013	B8	label3:	bge x4,x5,label4	
47	00525863	BC		bge x5,x4,label4	
48	0042d663	C0		addi x0,x0,0	
49	00000013	C4		addi x0,x0,0	
50	00000013	C8	label4:	bgeu x7,x6,label5	
51	0063f863	CC		bgeu x6,x7,label5	
52	00737663	D0		addi x0,x0,0	
53	00000013	D4		addi x0,x0,0	
54	00000013	D8	label5:	bge x4,x4,label6	
55	00425663	DC		addi x0,x0,0	
56	00000013	E0		addi x0,x0,0	
57	00000013	E4	label6:	lui x1,4	
58	000040b7	E8		jal x1,12	
59	00c000ef	EC		addi x0,x0,0	

Instr. Mem.(5)

NO.	Instruction	Addr.	Label	ASM	Comment
60	00000013	F0		addi x0,x0,0	
61	00000013	F4		lw x8, 24(x0)	
62	01802403	F8		sw x8, 28(x0)	
63	00802e23	FC		lw x1, 28(x0)	
64	01c02083	100		sh x8, 32(x0)	
65	02801023	104		lw x1, 32(x0)	
66	02002083	108		sb x8, 36(x0)	
67	02800223	10C		lw x1, 36(x0)	
68	02402083	110		lh x1, 26(x0)	
69	01a01083	114		lhu x1, 26(x0)	
70	01a05083	118		lb x1, 27(x0)	
71	01b00083	11C		lbu x1, 27(x0)	
72	01b04083	120		auipc x1, 0xffff0	
73	ffff0097	124		jalr x1,0(x0)	
74	000000e7	128			

Data Mem.

NO.	Data	Addr.	Comment
0	000080BF	0	
1	00000008	4	
2	00000010	8	
3	00000014	C	
4	FFFF0000	10	
5	0FFF0000	14	
6	FF000F0F	18	
7	F0F0F0F0	1C	
8	00000000	20	
9	00000000	24	
10	00000000	28	
11	00000000	2C	
12	00000000	30	
13	00000000	34	
14	00000000	38	
15	00000000	3C	

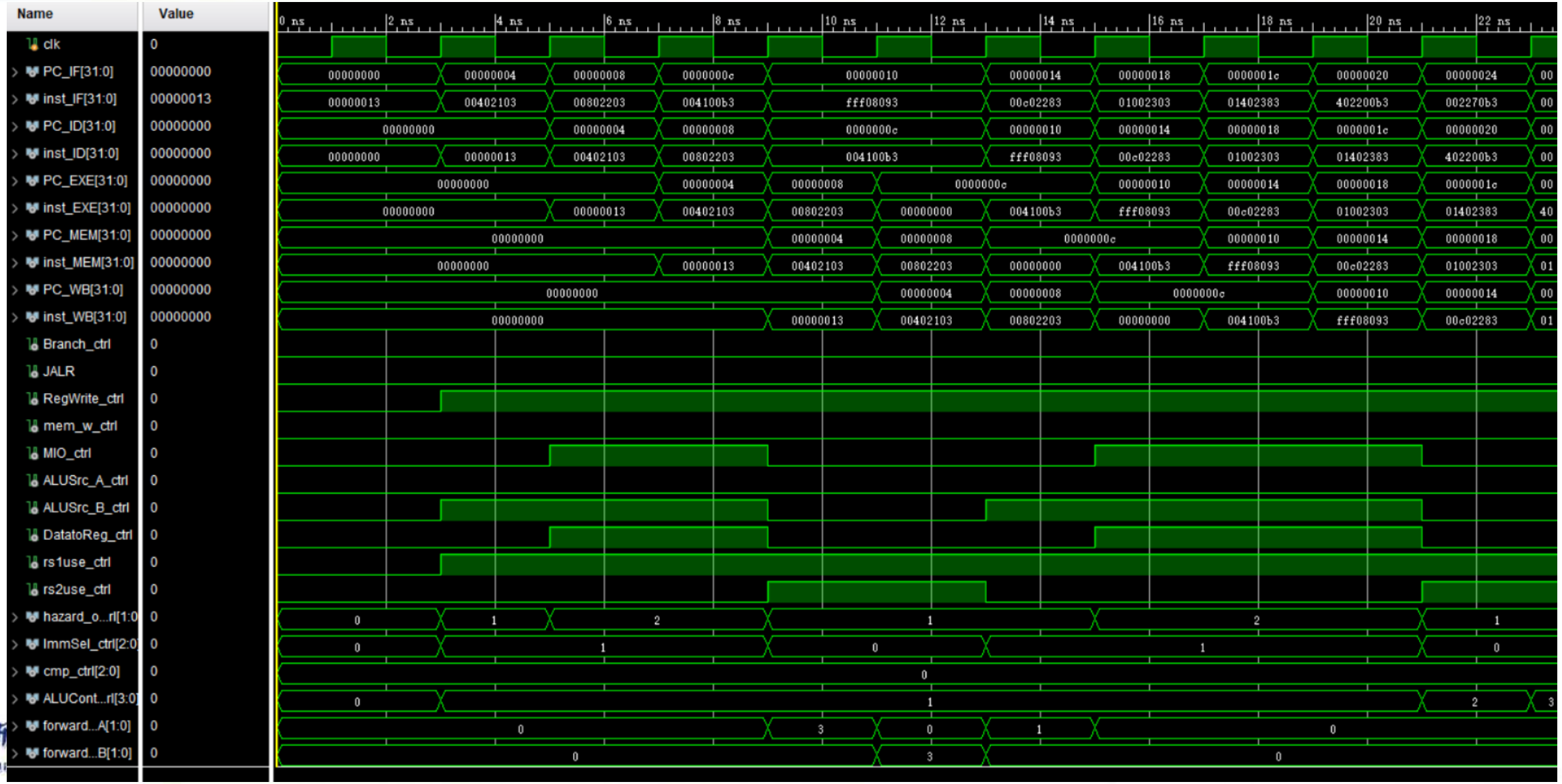
NO.	Instruction	Addr.	Comment
16	00000000	40	
17	00000000	44	
18	00000000	48	
19	00000000	4C	
20	A3000000	50	
21	27000000	54	
22	79000000	58	
23	15100000	5C	
24	00000000	60	
25	00000000	64	
26	00000000	68	
27	00000000	6C	
28	00000000	70	
29	00000000	74	
30	00000000	78	
31	00000000	7C	

Test Bench

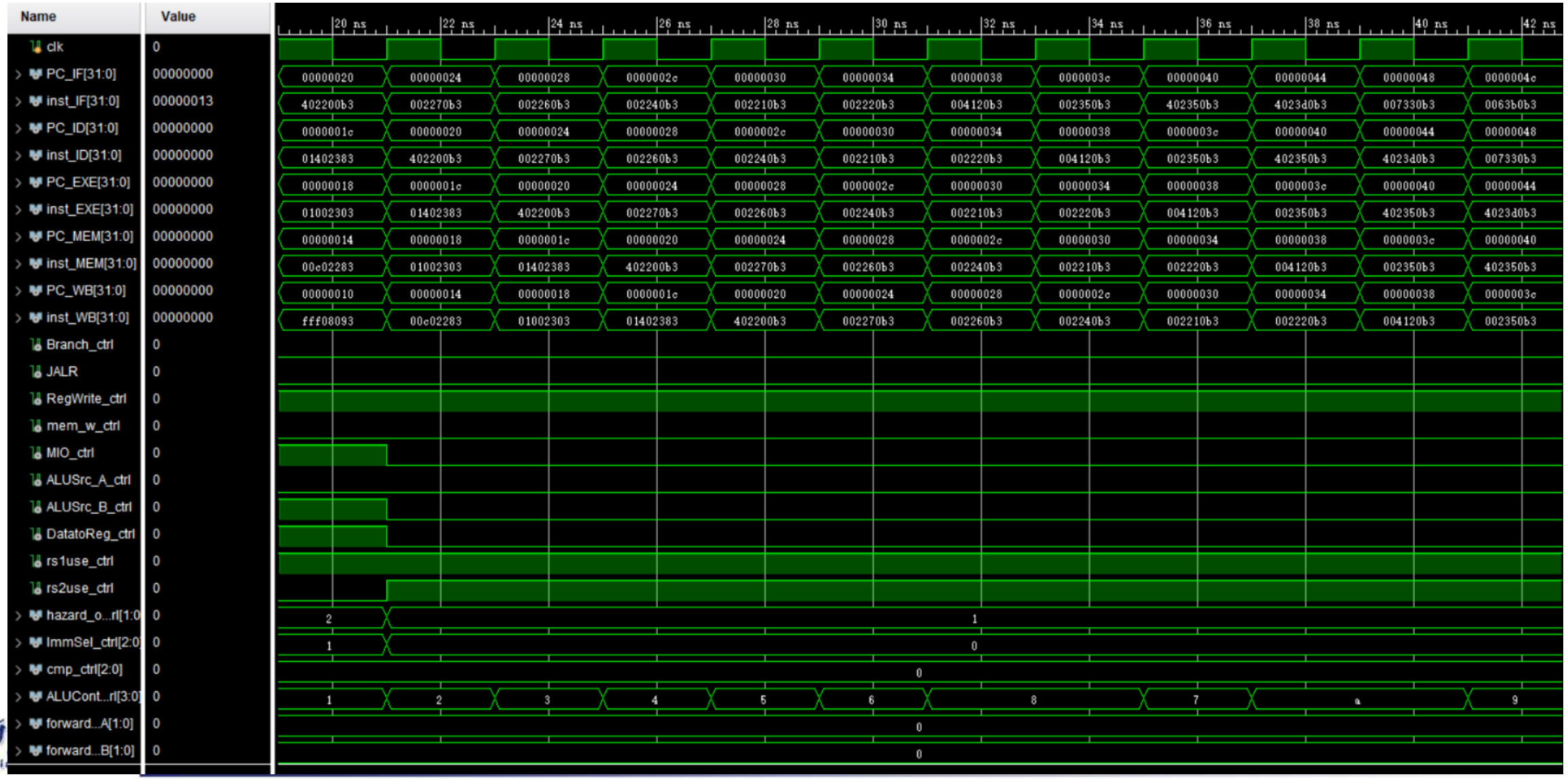
```
RV32core core(
    .debug_en(1'b0),
    .debug_step(1'b0),
    .debug_addr(7'b0),
    .debug_data(),
    .clk(clk),
    .rst(rst),
    .interrupter(1'b0)
);

initial begin
    clk = 0;
    rst = 1;
    #2 rst = 0;
end
always #1 clk = ~clk;
```

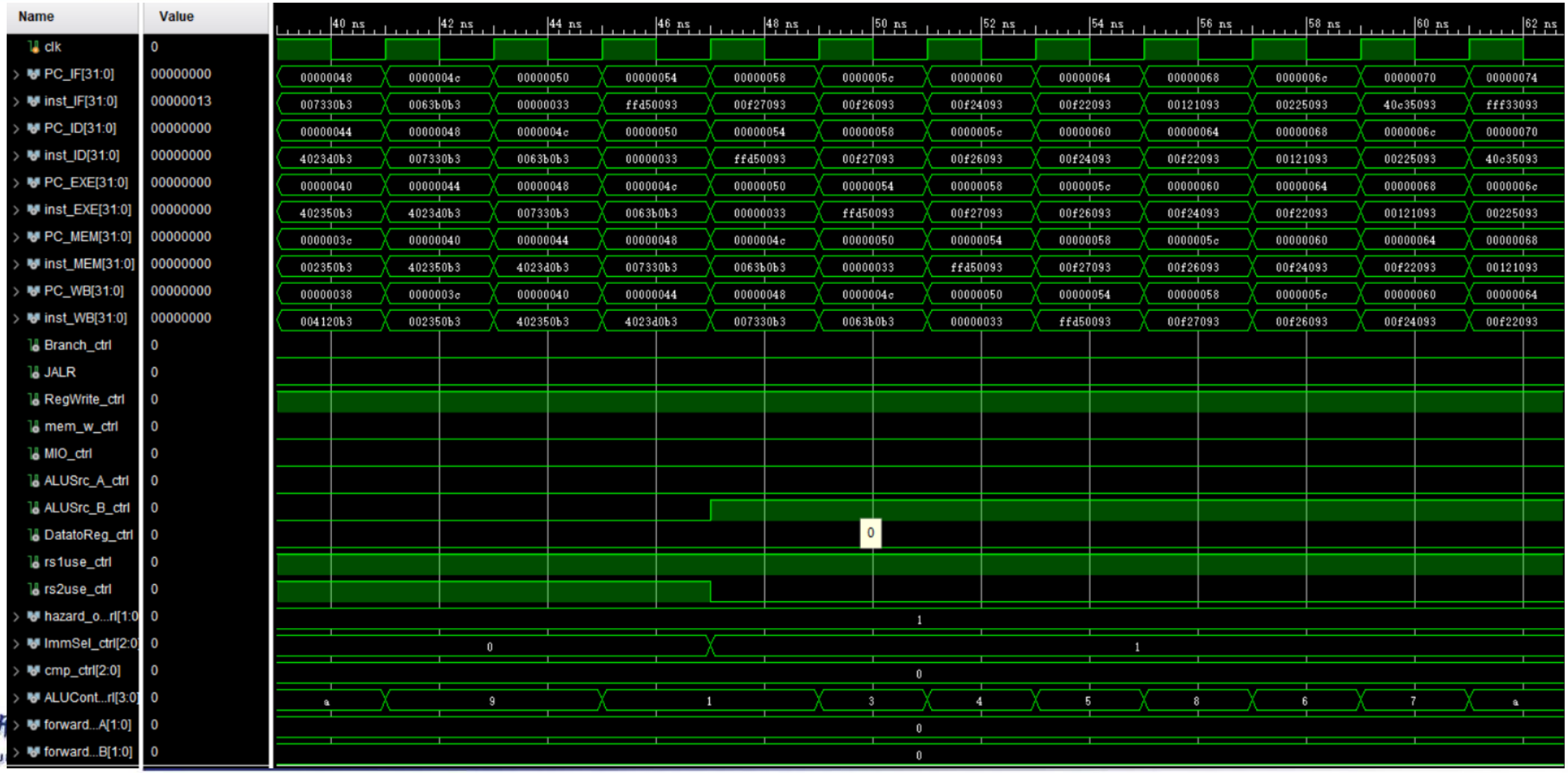
Simulation (1)



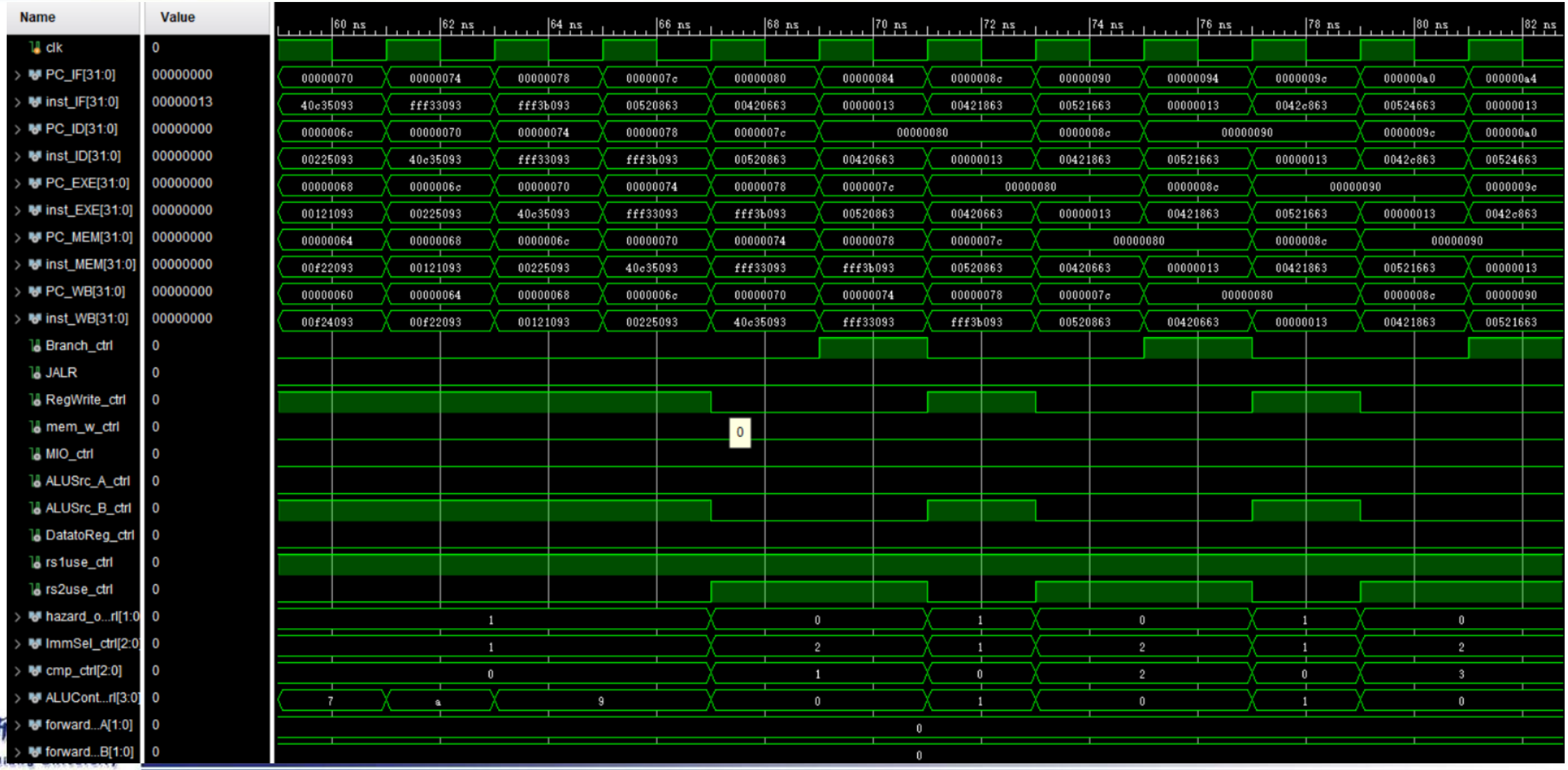
Simulation (2)



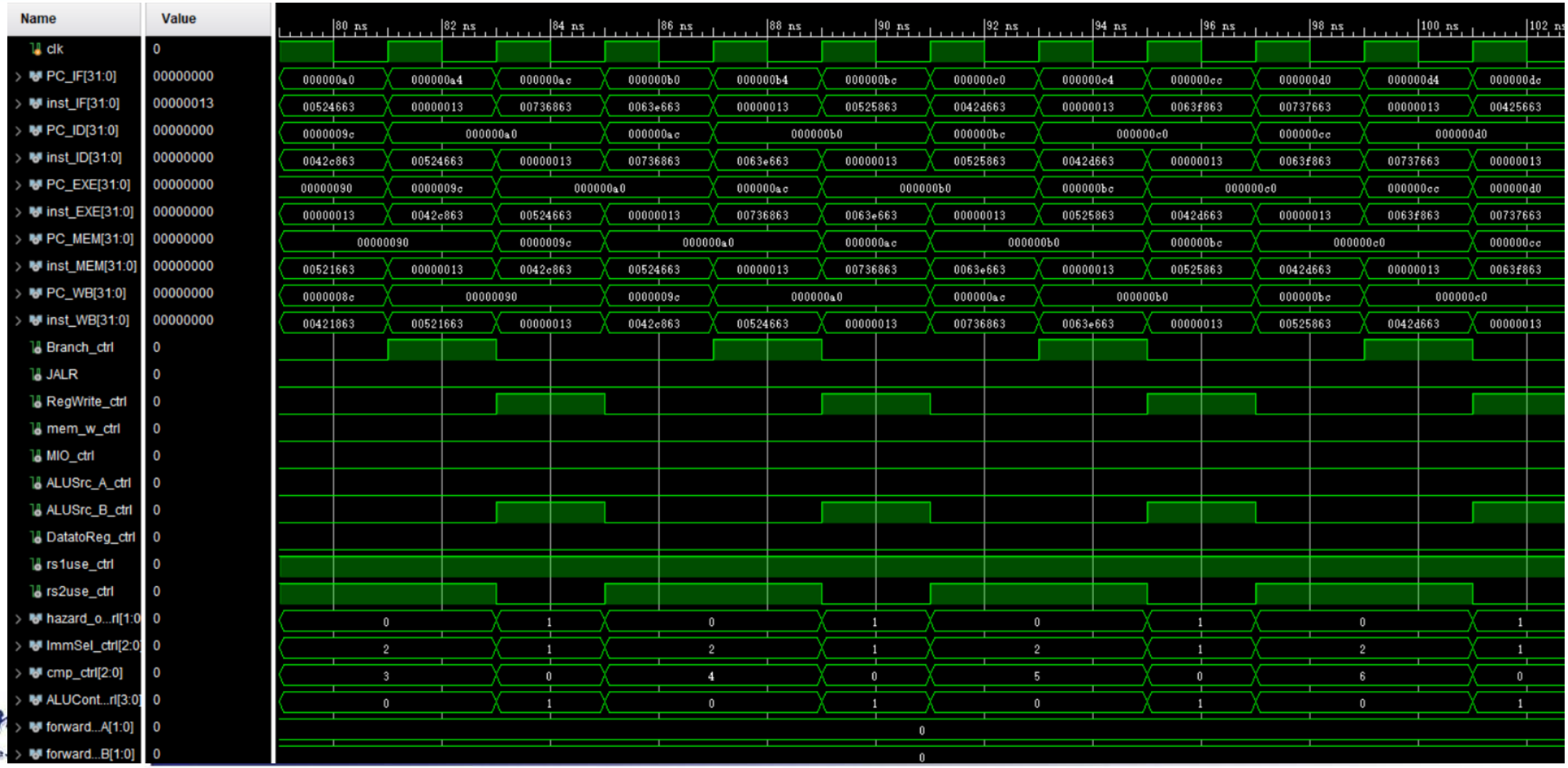
Simulation (3)



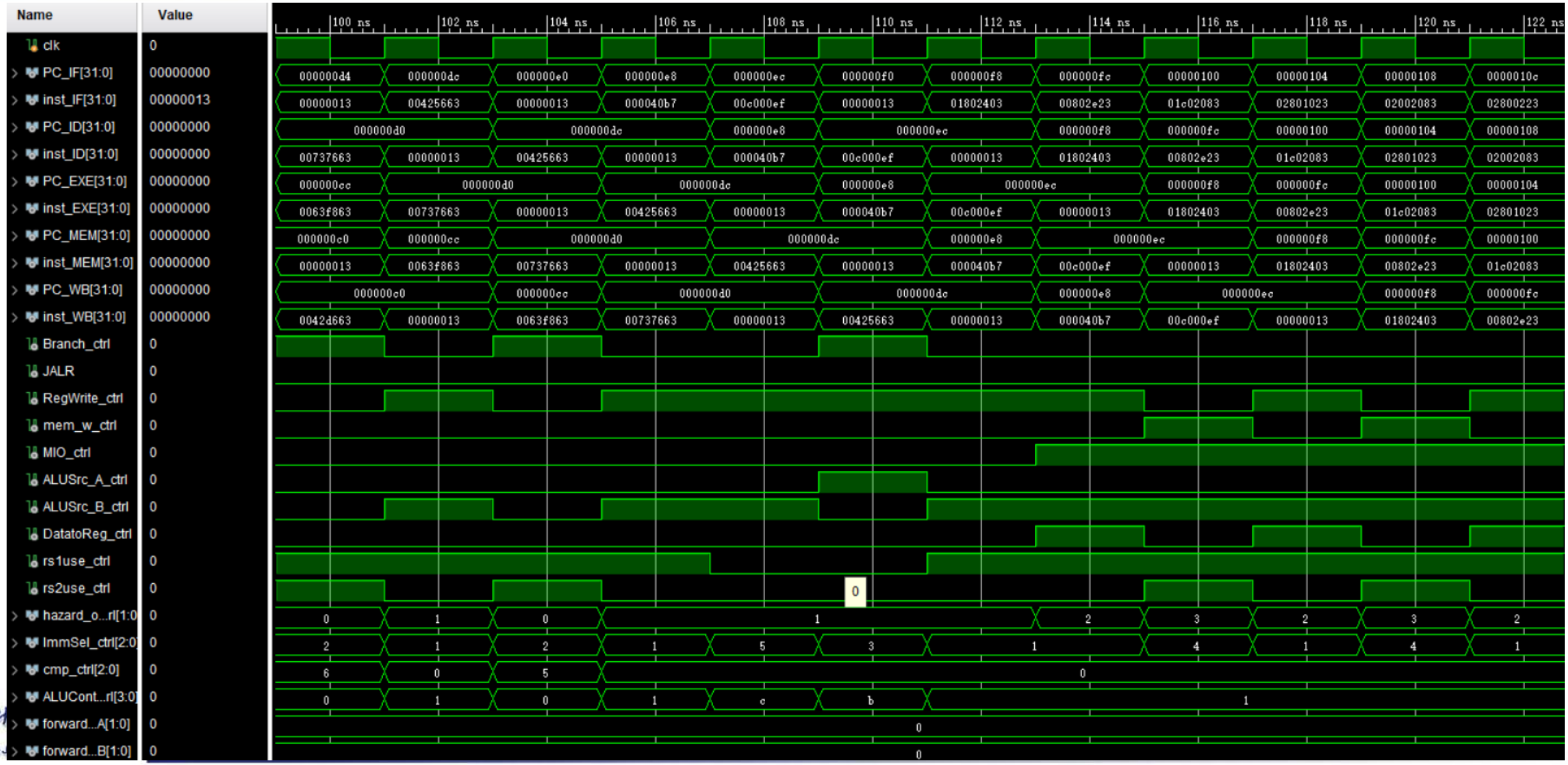
Simulation (4)



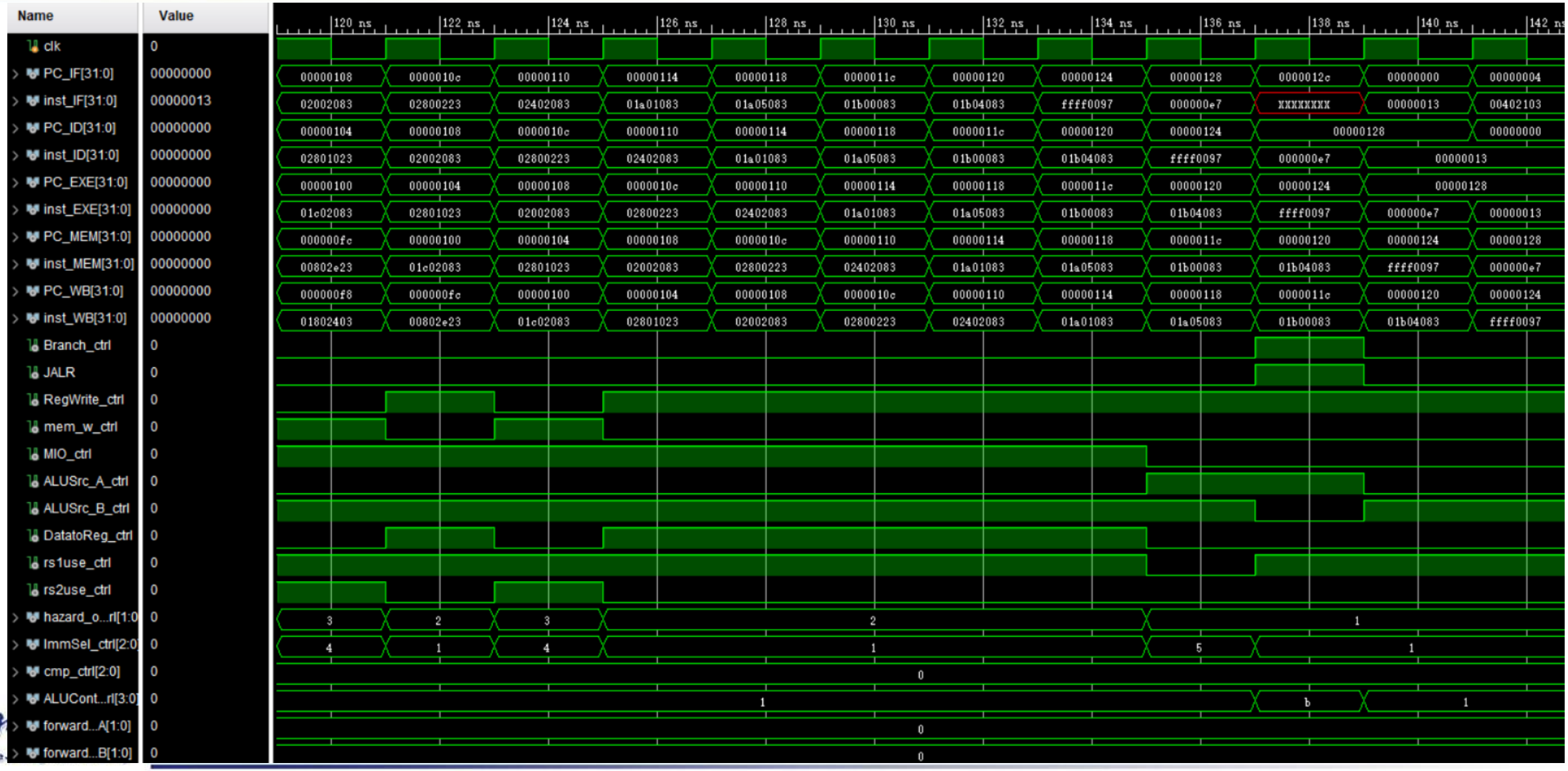
Simulation (5)



Simulation (6)



Simulation (7)



Checkpoints

- **CP 1:**

  Waveform Simulation of the Pipelined CPU with the verification program

- **CP 2:**

  FPGA Implementation of the Pipelined CPU with the verification program

Thanks!