浙江大学
ZHEJIANG UNIVERSITY

Computer Architecture Experiment

Topic 2. Pipelined CPU supporting exception & interrupt

浙江大学计算机学院

Outline

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

Experiment Purpose

- **Understand the principle of CPU exception & interrupt and its processing procedure.**
- **Master the design methods of pipelined CPU supporting exception & interrupt.**
- **master methods of program verification of Pipelined CPU supporting exception & interrupt.**

Experiment Task

- **Design of Pipelined CPU supporting exception & interrupt.**
  - Design datapath
  - Design Co-processor & Controller

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

Pipelined CPU supporting exception & interrupt

Level	Encoding	Name	Abbreviation
0	00	User/Application	U
1	01	Supervisor	S
2	10	*Reserved*	
3	11	Machine	M

Number of levels	Supported Modes	Intended Usage
1	M	Simple embedded systems
2	M, U	Secure embedded systems
3	M, S, U	Systems running Unix-like operating systems

Pipelined CPU supporting exception & interrupt

31 25	24 20	19 15	14 12	11 7	6 0	
csr	rs1	001	rd	1110011	I csrrw
csr	rs1	010	rd	1110011	I csrrs
csr	rs1	011	rd	1110011	I csrrc
csr	zimm	101	rd	1110011	I csrrwi
csr	zimm	110	rd	1110011	I cssrrsi
csr	zimm	111	rd	1110011	I csrrci

Pipelined CPU supporting exception & interrupt

Number	Privilege	Name	Description
Machine Information Registers			
0xF11	MRO	mvendorid	Vendor ID.
0xF12	MRO	marchid	Architecture ID.
0xF13	MRO	mimpid	Implementation ID.
0xF14	MRO	mhartid	Hardware thread ID.
Machine Trap Setup			
0x300	MRW	mstatus	Machine status register.
0x301	MRW	misa	ISA and extensions
0x302	MRW	medeleg	Machine exception delegation register.
0x303	MRW	mideleg	Machine interrupt delegation register.
0x304	MRW	mie	Machine interrupt-enable register.
0x305	MRW	mtvec	Machine trap-handler base address.
0x306	MRW	mcounteren	Machine counter enable.
Machine Trap Handling			
0x340	MRW	mscratch	Scratch register for machine trap handlers.
0x341	MRW	mepc	Machine exception program counter.
0x342	MRW	mcause	Machine trap cause.
0x343	MRW	mtval	Machine bad address or instruction.
0x344	MRW	mip	Machine interrupt pending.

Pipelined CPU supporting exception & interrupt

Environment Call and Breakpoint

31 — 20	19 — 15	14 — 12	11 — 7	6 — 0
funct12	rs1	funct3	rd	opcode
12	5	3	5	7
ECALL	0	PRIV	0	SYSTEM
EBREAK	0	PRIV	0	SYSTEM

Trap-Return Instructions

31 — 20	19 — 15	14 — 12	11 — 7	6 — 0
funct12	rs1	funct3	rd	opcode
12	5	3	5	7
MRET/SRET/URET	0	PRIV	0	SYSTEM

Pipelined CPU supporting exception & interrupt

在 M 模式运行期间可能发生的同步例外有五种：

- 访问错误异常 当物理内存的地址不支持访问类型时发生（例如尝试写入 ROM）。
- 断点异常 在执行 ebreak 指令，或者地址或数据与调试触发器匹配时发生。
- 环境调用异常 在执行 ecall 指令时发生。
- 非法指令异常 在译码阶段发现无效操作码时发生。
- 非对齐地址异常 在有效地址不能被访问大小整除时发生 。

Machine-mode status register (mstatus) for RV32.

31	30 … 23	22	21	20	19	18	17
SD	WPRI	TSR	TW	TVM	MXR	SUM	MPRV
1	8	1	1	1	1	1	1

16 15	14 13	12 11	10 9	8	7	6	5	4	3	2	1	0
XS[1:0]	FS[1:0]	MPP[1:0]	WPRI	SPP	MPIE	WPRI	SPIE	UPIE	MIE	WPRI	SIE	UIE
2	2	2	2	1	1	1	1	1	1	1	1	1

Pipelined CPU supporting exception & interrupt

发生异常/中断时，硬件自动经历如下的状态转换：

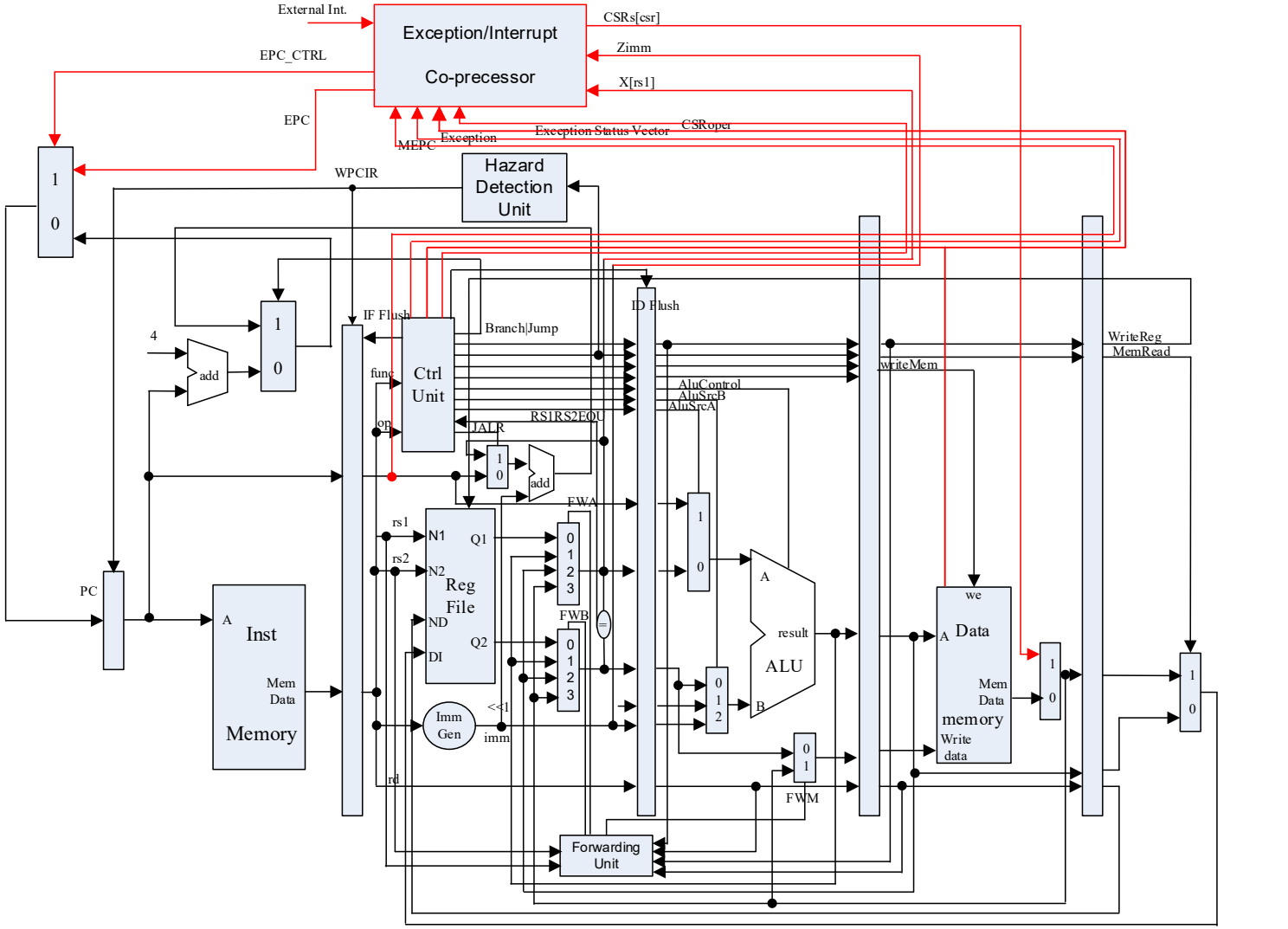
- 异常指令的PC被保存在mepc中，PC被设置为mtvec。mepc指向导致异常的指令；对于中断，它指向中断处理后应该恢复执行的位置。

- 根据异常来源设置mcause，并将mtval设置为出错的地址或者其它适用于特定异常的信息字。

- 把控制状态寄存器mstatus中的MIE位置零以禁用中断，并把先前的MIE值保留到MPIE中。

- 发生异常之前的权限模式保留在mstatus的MPP域中，再把权限模式更改为M。

Pipelined CPU supporting exception & interrupt



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	00c02283	C		lw x5, 12(x0)	
4	01002303	10		lw x6, 16(x0)	
5	01402383	14		lw x7, 20(x0)	
6	306850f3	18		csrrwi x1, 0x306, 16	
7	306020f3	1C		csrr x1, 0x306	
8	306310f3	20		csrrw x1, 0x306, x6	
9	306020f3	24		csrr x1, 0x306	
10	00000013	28		addi x0, x0, 0	
11	07800093	2C		addi x1, x0, 120	
12	30509073	30		csrw 0x305, x1	
13	00000013	34		addi x0, x0, 0	
14	00000073	38		ecall	

Instr. Mem.(2)

NO.	Instruction	Addr.	Label	ASM	Comment
15	00000013	3C		addi x0, x0, 0	
16	00000012	40		addi x0, x0, 0	# change to illegal
17	00000013	44		addi x0, x0, 0	
18	07f02083	48		lw x1, 127(x0)	
19	08002083	4C		lw x1, 128(x0)	# l access fault
20	00000013	50		addi x0, x0, 0	
21	08102023	54		sw x1, 128(x0)	# s access fault
22	00000013	58		addi x0, x0, 0	
23	00000013	5C		addi x0, x0, 0	
24	00000013	60		addi x0, x0, 0	
25	00000013	64		addi x0, x0, 0	
26	00000013	68		addi x0, x0, 0	
27	00000013	6C		addi x0, x0, 0	
28	00000013	70		addi x0, x0, 0	
29	00000067	74		jr x0	

Instr. Mem.(3)

NO.	Instruction	Addr.	Label	ASM	Comment
30	34102cf3	78	trap:	csrr x25, 0x341	# mepc
31	34202df3	7C		csrr x27, 0x342	# mcause
32	30002e73	80		csrr x28, 0x300	# mstatus
33	30402ef3	84		csrr x29, 0x304	# mie
34	34402f73	88		csrr x30, 0x344	# mip
35	004c8113	8C		addi x2, x25, 4	
36	34111073	90		csrw 0x341, x2	
37	30200073	94		mret	# 30200073 mret
38	00000013	98		addi x0, x0, 0	
39	00000013	9C		addi x0, x0, 0	
40	00000013	A0		addi x0, x0, 0	
41	00000013	A4		addi x0, x0, 0	

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	

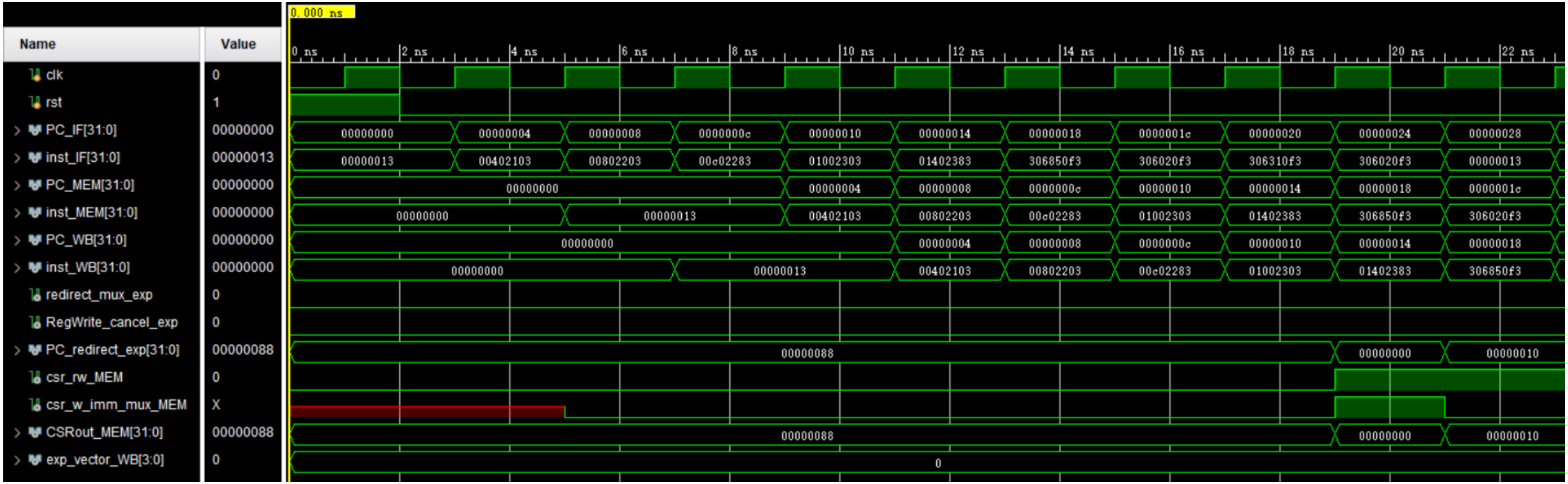
NO.	Instruction	Addr.	Comment
16	00000000	40	
17	00000000	44	
18	00000000	48	
19	00000000	4C	
20	A3000000	50	
21	27000000	54	
22	79000000	58	
23	15100000	5C	
24	00000000	60	
25	00000000	64	
26	00000000	68	
27	00000000	6C	
28	00000000	70	
29	00000000	74	
30	00000000	78	
31	00000000	7C	

Test Bench

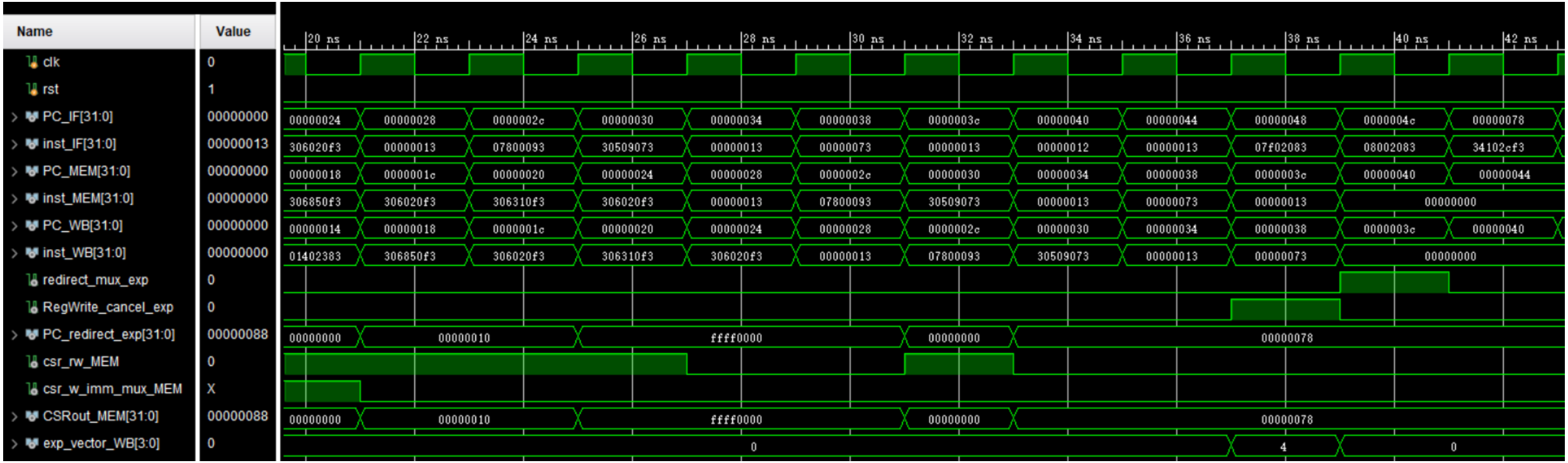
```
RV32core core(
    .debug_en(1'b0),
    .debug_step(1'b0),
    .debug_addr(7'b0),
    .debug_data(),
    .clk(clk),
    .rst(rst),
    .interrupter(1'b0)
);

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

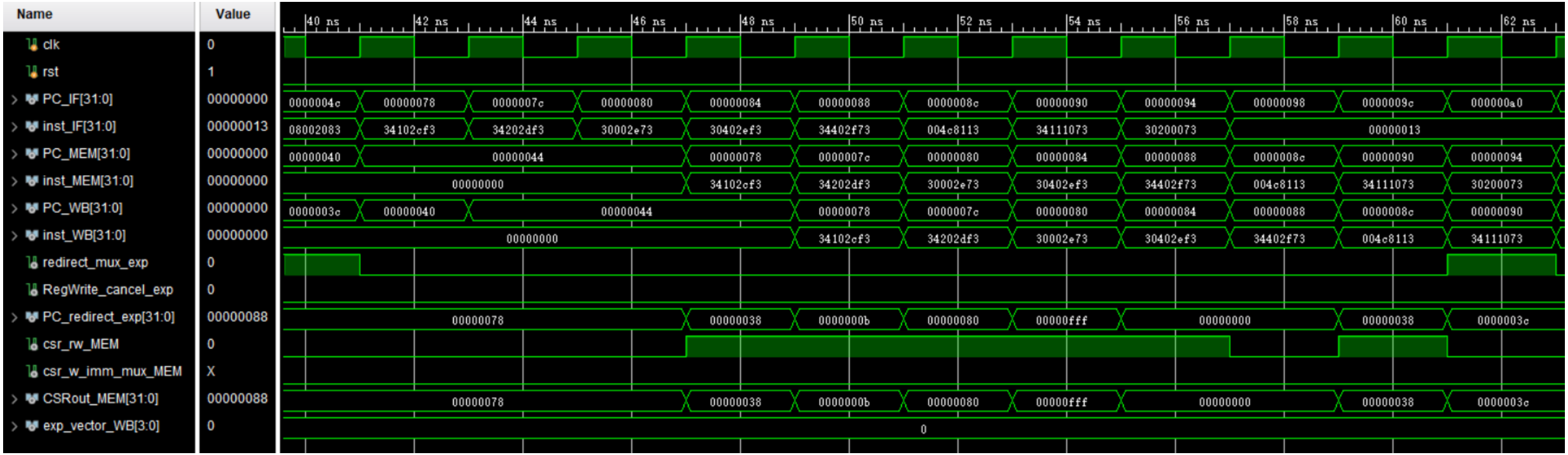
Simulation (1)



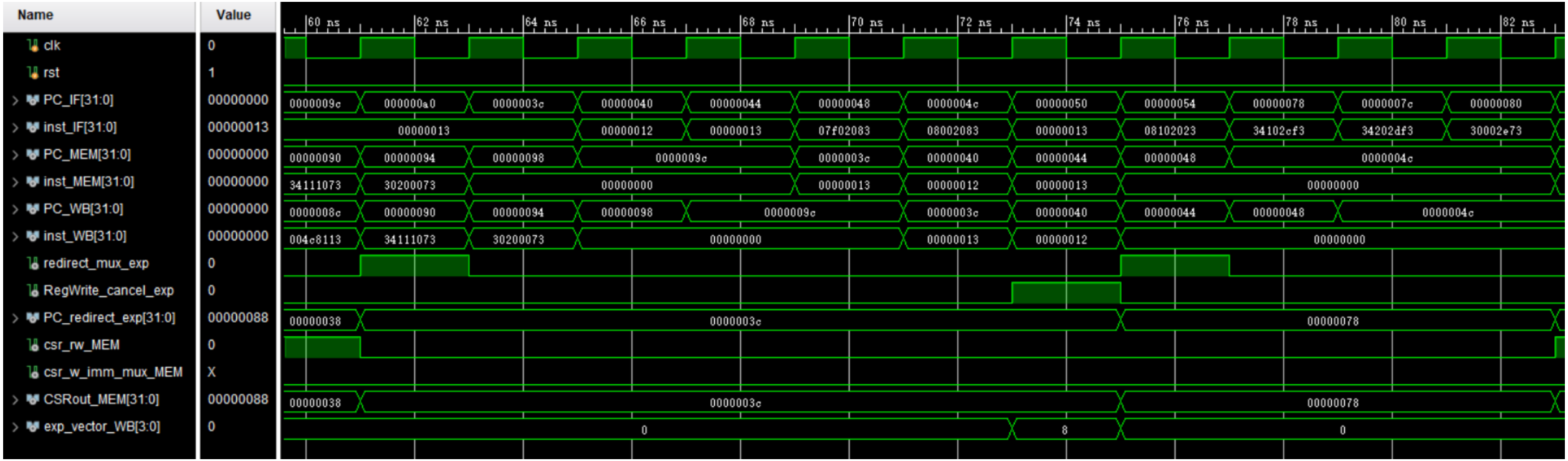
Simulation (2)



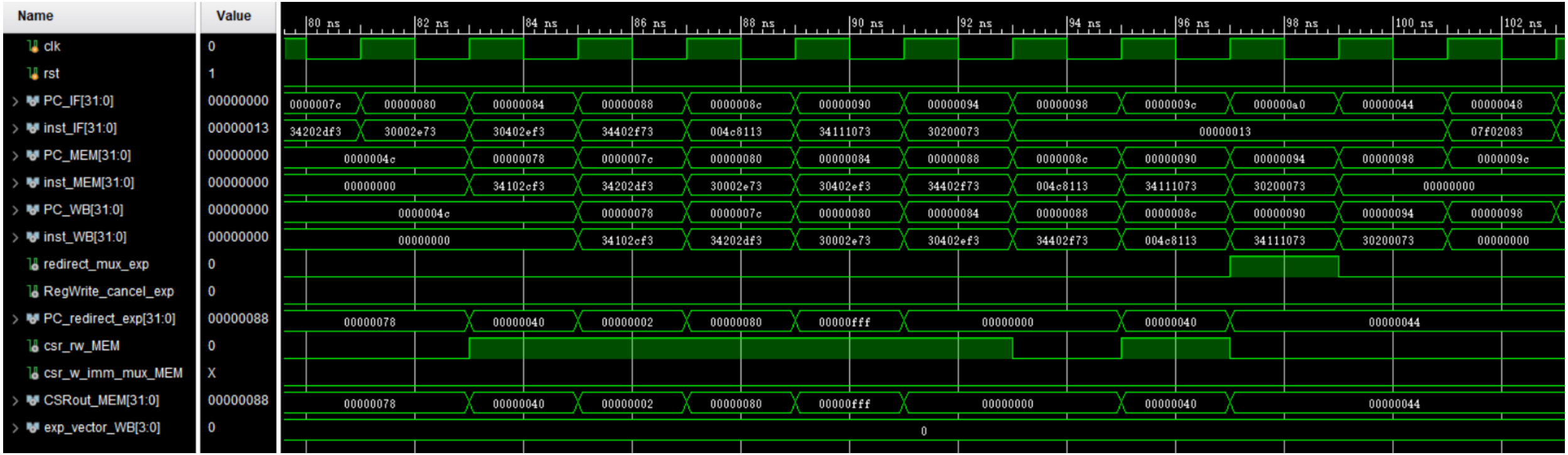
Simulation (3)



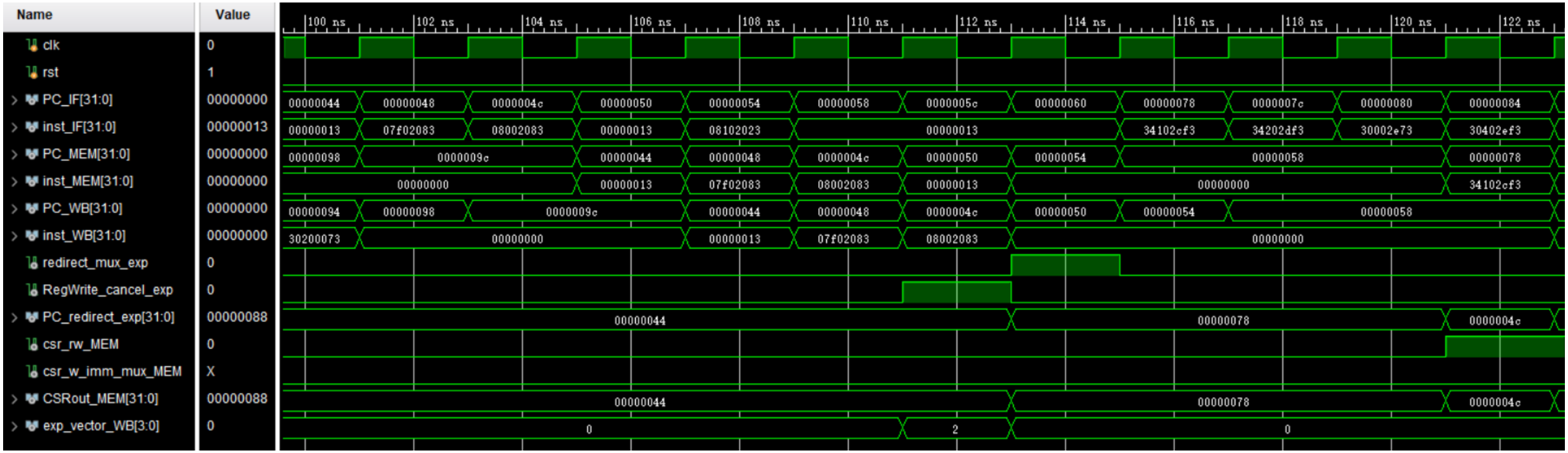
Simulation (4)



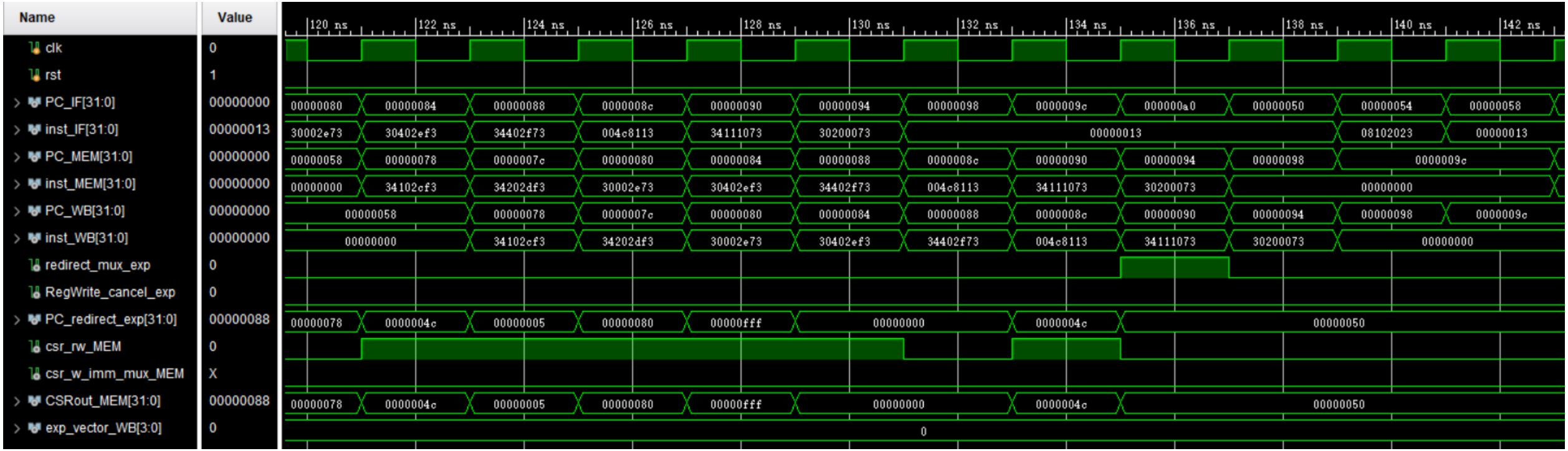
Simulation (5)



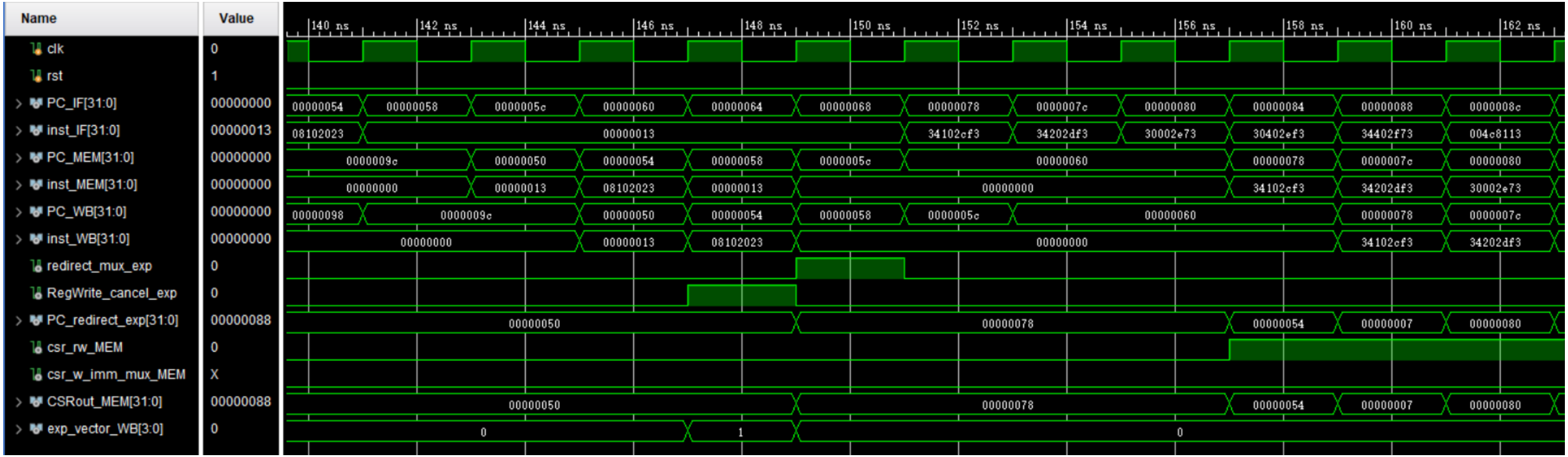
Simulation (6)



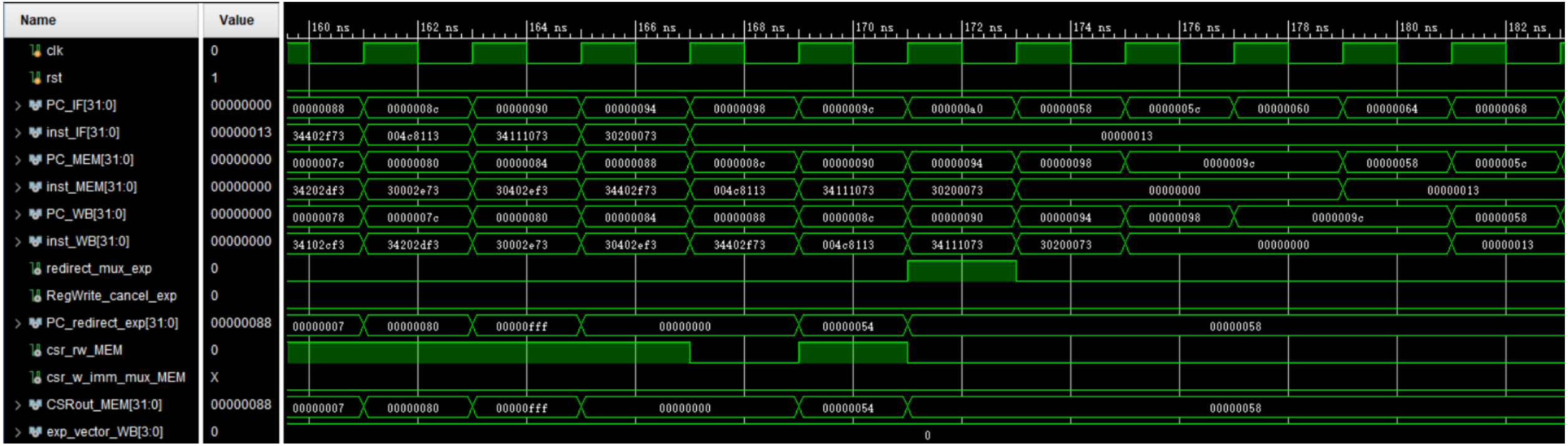
Simulation (7)



Simulation (8)



Simulation (9)



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!