



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





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





31	25 24	20	19 15	14 12	,11 7	76 0	4
	csr		rs1	001	rd	1110011	I csrrw
	csr		rs1	010	rd	1110011	I csrrs
	csr		rs1	011	rd	1110011	I estre
	csr		zimm	101	rd	1110011	I csrrwi
	csr		zimm	110	rd	1110011	I essrrsi
	csr		zimm	111	rd	1110011	I esrrei





Number	Privilege	Name	Description							
		Mach	ine 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										
0x301										
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.							
		M	achine 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.							





Environment Call and Breakpoint

31	20 19	15 14 12	11	7 6	0
funct12	rs1	funct3	$_{\mathrm{rd}}$	opcode	
12	5	3	5	7	
ECALL	0	PRIV	0	SYSTEM	
EBREAK	0	PRIV	0	SYSTEM	

Trap-Return Instructions

31		20 19	1	15 14	12	11	7 6	0
	funct12		rs1	fun	ct3	rd	opc	ode
,	12		5	3	3	5	7	
	MRET/SRET/URET		0	PR	IV	0	SYST	EM





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.

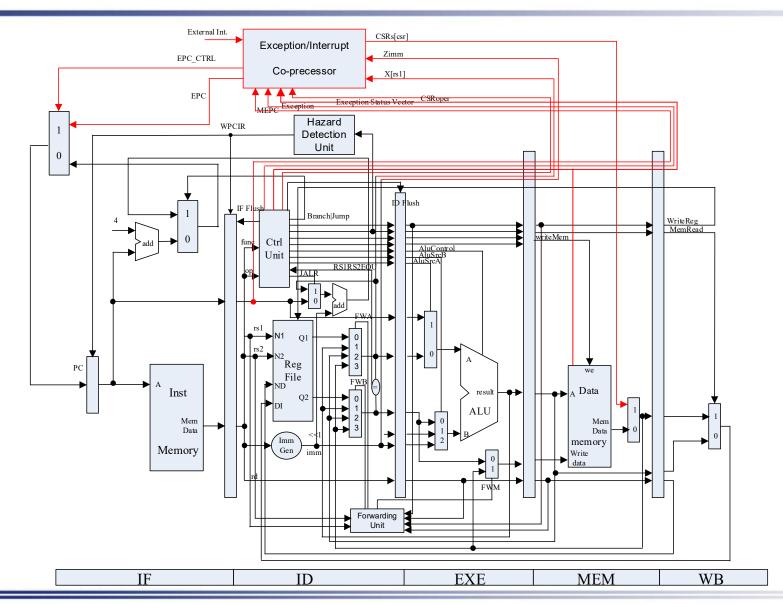
	31 30						23	22	21	20	19	18	17	
	SD		V	VPRI				TSR	TW	TVM	MXR	SUM	MPR	V
	1			8				1	1	1	1	1	1	
	16 15	14 13	12 11	10 9	8	7	(3	5	4	3	2	1	0
	XS[1:0]	FS[1:0]	MPP[1:0]	WPRI	SPP	MPIE	WF	PRI	SPIE	UPIE	MIE	WPRI	SIE	UIE
1.	2	2	2	2	1	1	1	L	1	1	1	1	1	1



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。







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	С		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	







	NO.	Instruction	Addr.	Label	ASM	Comment
	15	0000013	3C		addi x0, x0, 0	
	16	0000012	40		addi x0, x0, 0	# change to illegal
	17	0000013	44		addi x0, x0, 0	
	18	07f02083	48		lw x1, 127(x0)	
	19	08002083	4C		lw x1, 128(x0)	# I access fault
	20	0000013	50		addi x0, x0, 0	
	21	08102023	54		sw x1, 128(x0)	# s access fault
	22	0000013	58		addi x0, x0, 0	
	23	00000013	5C		addi x0, x0, 0	
	24	00000013	60		addi x0, x0, 0	
	25	0000013	64		addi x0, x0, 0	
	26	0000013	68		addi x0, x0, 0	
	27	0000013	6C		addi x0, x0, 0	
	28	0000013	70		addi x0, x0, 0	
4	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	0000013	98		addi x0, x0, 0	
39	0000013	9C		addi x0, x0, 0	
40	0000013	Α0		addi x0, x0, 0	
41	0000013	A4		addi x0, x0, 0	
(4)					



Data Mem.

NO.	Data	Addr.	Comment	NO.	Instruction	Addr.	Comment
0	000080BF	0		16	00000000	40	
1	8000000	4		17	00000000	44	
2	0000010	8		18	00000000	48	
3	0000014	С		19	00000000	4C	
4	FFFF0000	10		20	A3000000	50	
5	0FFF0000	14		21	27000000	54	
6	FF000F0F	18		22	79000000	58	
7	F0F0F0F0	1C		23	15100000	5C	
8	00000000	20		24	00000000	60	
9	00000000	24		25	00000000	64	
10	00000000	28		26	00000000	68	
11	00000000	2C		27	00000000	6C	
12	00000000	30		28	00000000	70	
13	0000000	34		29	00000000	74	
14	00000000	38		30	00000000	78	
3 , 15	0000000	3C		31	00000000	7C	

Test Bench

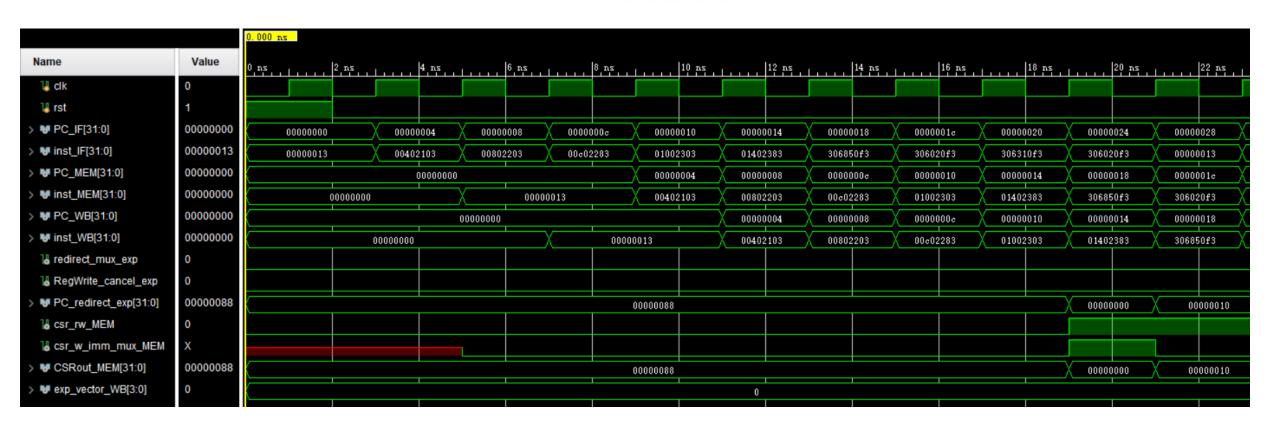


```
RV32core core(
   .debug_en(1'b0),
   .debug_step(1'b0),
   .debug_addr(7'b0),
   .debug_data(),
   .clk(clk),
   .rst(rst),
   .interrupter(1'b0)
initial begin
   clk = 0;
   rst = 1;
   #2 rst = 0;
end
always #1 clk = ^{\sim}clk;
```



Simulation (1)







Simulation (2)



Name	Value	20 ns	22 ns	24 ns	26 ns	28 ns	30 ns	32 ns	34 ns	36 ns	38 ns	40 ns	42 ns
¼ dk	0												
¼ rst	1												
> ₩ PC_IF[31:0]	00000000	00000024	00000028	0000002c	00000030	00000034	00000038	0000003c	00000040	00000044	00000048	0000004c	00000078
■ inst_IF[31:0]	00000013	306020 f 3	00000013	07800093	30509073	00000013	00000073	00000013	00000012	00000013	07f02083	08002083	34102cf3
₩ PC_MEM[31:0]	00000000	00000018	0000001c	00000020	00000024	00000028	00000020	00000030	00000034	00000038	0000003c	00000040	00000044
■ inst_MEM[31:0]	00000000	306850 f 3	306020f3	306310f3	306020f3	00000013	07800093	30509073	00000013	00000073	00000013	000	00000
₩ PC_WB[31:0]	00000000	00000014	00000018	0000001c	00000020	00000024	00000028	0000002c	00000030	00000034	00000038	00000030	00000040
₩ inst_WB[31:0]	00000000	01402383	306850f3	306020f3	306310f3	306020f3	00000013	07800093	30509073	00000013	00000073	000	00000
laredirect_mux_exp	0												
RegWrite_cancel_exp	0												
PC_redirect_exp[31:0]	88000000	00000000	0000	0010		ffff0000		00000000			00000078		
le csr_rw_MEM	0												
d csr_w_imm_mux_MEM	X												
♥ CSRout_MEM[31:0]	88000000	00000000	0000	0010		ffff0000		00000000			00000078		
exp_vector_WB[3:0]	0					0					X 4	χ '	0



Simulation (3)



Name	Value	40 ns 42 ns	44 ns 46 ns	48 ns	50 ns	52 ns	54 ns	56 ns	58 ns	60 ns	62 ns
¼ dk	0										
¼ rst	1										
> W PC_IF[31:0]	00000000	0000004c 00000078	0000007c 00000080	00000084	00000088	0000008c	00000000	00000094	00000098	0000009c	00000040
> W inst_IF[31:0]	00000013	08002083 34102cf3	34202df3 30002e73	30402ef3	34402f73	004c8113	34111073	30200073		00000013	
> W PC_MEM[31:0]	00000000	00000040	00000044	00000078	0000007c	00000080	00000084	00000088	00000086	00000090	00000094
> W inst_MEM[31:0]	00000000	000	000000	34102cf3	34202df3	30002e73	30402ef3	34402f73	004c8113	34111073	30200073
> W PC_WB[31:0]	00000000	0000003c 00000040	00000044		00000078	0000007c	0800000	00000084	00000088	0000008c	00000090
> W inst_WB[31:0]	00000000		00000000		34102cf3	34202df3	30002e73	30402ef3	34402f73	004c8113	34111073
<pre>Tedirect_mux_exp</pre>	0										
RegWrite_cancel_exp	0										
> W PC_redirect_exp[31:0]	88000000	000	000078	00000038	ОООООООЪ	00000080	00000fff X	00000	0000	00000038	00000036
¹å csr_rw_MEM	0										
¼ csr_w_imm_mux_MEM	X										
> W CSRout_MEM[31:0]	88000000	000	000078	00000038	ОООООООЪ	00000080	00000fff X	00000	0000	00000038	00000030
> W exp_vector_WB[3:0]	0					0					



Simulation (4)



Name	Value	60 ns	62 ns	64 ns	66 ns	68 ns	70 ns	72 ns	74 ns	76 ns	78 ns	80 ns	82 ns
¼ dk	0												
¼ rst	1												
> W PC_IF[31:0]	00000000	0000009c	000000a0	0000003c	00000040	00000044	00000048	0000004c	00000050	00000054	00000078	0000007c	00000080
> W inst_IF[31:0]	00000013		00000013		00000012	00000013	07f02083	08002083	00000013	08102023	34102cf3	34202df3	30002e73
> W PC_MEM[31:0]	00000000	00000090	00000094	00000098	00000	0090	00000030	00000040	00000044	00000048	X	0000004c	X
> W inst_MEM[31:0]	00000000	34111073	30200073		00000000		00000013	00000012	00000013	Χ	0000	1000	X
> W PC_WB[31:0]	00000000	0000008c	00000090	00000094	00000098	000	0009c	0000003c	00000040	00000044	00000048	000	00004c
> W inst_WB[31:0]	00000000	004c8113	34111073	30200073	X	00000000		00000013	00000012	X	000	10000	
<pre>redirect_mux_exp</pre>	0												
RegWrite_cancel_exp	0												
> W PC_redirect_exp[31:0]	88000000	00000038	<u> </u>			0000003c				X	0000	1078	X
d csr_rw_MEM	0												
¼ csr_w_imm_mux_MEM	X												
> W CSRout_MEM[31:0]	88000000	00000038				0000003c				X	0000	1078	X
> W exp_vector_WB[3:0]	0				0	0			8	X		0	



Simulation (5)



Name	Value	80 ns	82 ns 84 ns	86 ns		90 ns	92 ns	94 ns	96 ns	98 ns	100 ns	102 ns
¼ clk	0											
¼ rst	1											
> W PC_IF[31:0]	00000000	0000007c X 0000	0080 00000084	00000088	0000008c	00000090	00000094	00000098	0000009c	000000a0	00000044	00000048
> W inst_IF[31:0]	00000013	34202df3 3000	2e73 30402ef3	34402f73	004c8113	34111073	30200073	X	0000	0013		07f02083
> W PC_MEM[31:0]	00000000	0000004c	00000078	0000007e	00000080	00000084	00000088	00000080	00000090	00000094	00000098	00000090
> W inst_MEM[31:0]	00000000	00000000	34102cf3	34202df3	30002e73	30402ef3	34402f73	004c8113	34111073	30200073	000	00000
> ₩ PC_WB[31:0]	00000000	00	00004c	00000078	0000007c	00000080	00000084	00000088	0000008c	00000090	00000094	00000098
> W inst_WB[31:0]	00000000	00	000000	34102cf3	34202df3	30002e73	30402ef3	34402f73	004c8113	34111073	30200073	00000000
<pre>redirect_mux_exp</pre>	0											
RegWrite_cancel_exp	0											
> ₩ PC_redirect_exp[31:0]	8800000	00000078	00000040	00000002	00000080	00000fff	0000	0000	00000040	X	00000044	
¼ csr_rw_MEM	0											
¹⊌ csr_w_imm_mux_MEM	X											
> ₩ CSRout_MEM[31:0]	8800000	00000078	00000040	00000002	00000080	00000fff	0000	0000	00000040	Χ	00000044	
> W exp_vector_WB[3:0]	0						0					



Simulation (6)



Name	Value	100 ns	102 n	s	104 ns	1	06 ns.		108 ns	ليبيا	110 ns		112 ns		114 ns		116 ns		118 ns		120 ns		122 ns
¼ dk	0																						
¼ rst	1																						
> W PC_IF[31:0]	00000000	00000044	00000048	00000	004c	000000	50	00000	054	00000	058	0000	005c	0000	0060	0000	0078	0000	007c	00000	080	00000	0084
> W inst_IF[31:0]	00000013	00000013	07f02083	08002	2083	000000	13	08102	023	\subset		0000	0013			3410	2cf3	3420	2df3	30002	e73	30402	2ef3
> W PC_MEM[31:0]	00000000	00000098	X 00	00009c		000000	44	00000	048	00000	04c	0000	0050	0000	0054			0000	0058			00000	0078
> W inst_MEM[31:0]	00000000		00000000			000000	13	07f02	083	08002	083	0000	0013	X			0000	0000				34102	2cf3
> W PC_WB[31:0]	00000000	00000094	0000c 00000		044	00000048 000000		004c	00000050		0000	0054	X		00000	058							
> W inst_WB[31:0]	00000000	30200073	X	00000	0000	00000013			013	07f02	083	0800	2083	00000000									
redirect_mux_exp	0																						
RegWrite_cancel_exp	0																						
> W PC_redirect_exp[31:0]	8800000					0000	0044							X			0000	0078				00000	004c
" csr_rw_MEM	0																						
d csr_w_imm_mux_MEM	X																						
> W CSRout_MEM[31:0]	00000088					0000	0044							X			0000	0078				00000	004c
> W exp_vector_WB[3:0]	0					0							2	X					0				



Simulation (7)



Name	Value	120 ns	122 ns	124 ns	126 ns	128 ns	130 ns	132 ns	134 ns	136 ns	138 ns	140 ns	142 ns
¼ dk	0												
¼ rst	1												
> W PC_IF[31:0]	00000000	08000000	00000084	00000088	0000008c	00000090	00000094	00000098	00000090	000000a0	00000050	00000054	00000058
> W inst_IF[31:0]	00000013	30002e73	30402ef3	34402f73	004c8113	34111073	30200073		0000	0013		08102023	00000013
> W PC_MEM[31:0]	00000000	00000058	00000078	0000007c	00000080	00000084	00000088	0000008c	00000090	00000094	00000098	0000	009c
> W inst_MEM[31:0]	00000000	00000000	34102cf3	34202df3	30002e73	30402ef3	34402f73	004c8113	34111073	30200073	X	00000000	
> W PC_WB[31:0]	00000000	00	000058	00000078	0000007c	00000080	00000084	00000088	0000008c	00000090	00000094	00000098	00000090
> W inst_WB[31:0]	00000000	00	000000	34102cf3	34202df3	30002e73	30402ef3	34402f73	004c8113	34111073	30200073	000	00000
<pre>Tedirect_mux_exp</pre>	0												
RegWrite_cancel_exp	0												
> W PC_redirect_exp[31:0]	88000000	00000078	0000004c	00000005	00000080	00000fff	00000	000	0000004c	(<u> </u>	000	00050	
¹₀ csr_rw_MEM	0												
dia csr_w_imm_mux_MEM	X												
> W CSRout_MEM[31:0]	88000000	00000078	0000004c	00000005	00000080	00000fff	00000	000	00000040		000	00050	
> w exp_vector_WB[3:0]	0							0					



Simulation (8)



Name	Value	140 ns	142 ns	ns 146	ns. 148 ns	150 ns	152 ns	154 ns	156 ns	158 ns	160 ns	162 ns
¼ dk	0											
¼ rst	1											
> W PC_IF[31:0]	00000000	00000054 0000	0000005	00000060	00000064	8900000	00000078	0000007c 000	00080	00000084	00000088	00000080
> 😽 inst_IF[31:0]	00000013	08102023		00000013			34102cf3	34202df3 300	02e73	30402ef3	34402f73	004c8113
> ₩ PC_MEM[31:0]	00000000	0000009c	00000050	00000054	00000058	00000050	Χ	00000060	X_	00000078	0000007c	00000080
> W inst_MEM[31:0]	00000000	00000000	00000013	08102023	00000013	X	0000	0000	X	34102cf3	34202df3	30002e73
> W PC_WB[31:0]	00000000	00000098	0000009c	00000050	00000054	00000058	0000005c	000	00060		00000078	0000007c
> W inst_WB[31:0]	00000000		000000	00000013	08102023	X		00000000			34102cf3	34202df3
√ redirect_mux_exp	0											
RegWrite_cancel_exp	0											
> W PC_redirect_exp[31:0]	00000088		00000	050		X	0000	0078	X_	00000054	00000007	00000080
d csr_rw_MEM	0											
To csr_w_imm_mux_MEM	Х											
> W CSRout_MEM[31:0]	00000088		00000	050		X	0000	0078	X	00000054	00000007	00000080
> W exp_vector_WB[3:0]	0		0		1	Х			0			



Simulation (9)



Name	Value	160 ns	162 ns	164 ns	166 ns	168 ns	170 ns	172 ns	174 ns	176 ns	178 ns	180	ns	182 ns
[™] clk	0													
¼ rst	1													
> W PC_IF[31:0]	00000000	00000088	0000008c	00000090	00000094	00000098	00000090	000000a0	00000058	0000005c 0	0000060	00000064	0000	00068
> W inst_IF[31:0]	00000013	34402f73	004c8113	34111073	30200073	Χ			000	00013				
> ₩ PC_MEM[31:0]	00000000	0000007c	00000080	00000084	00000088	0000008c	00000090	00000094	00000098	00000090	X	00000058	0000	0005c
> W inst_MEM[31:0]	00000000	34202df3	30002e73	30402ef3	34402f73	004c8113	34111073	30200073	X	00000000	X		00000013	
> W PC_WB[31:0]	00000000	00000078	0000007c	00000080	00000084	00000088	0000008c	00000090	00000094	00000098	000000	19c	0000	0058
> W inst_WB[31:0]	00000000	34102cf3	34202df3	30002e73	30402ef3	34402f73	004c8113	34111073	30200073	01	000000		000	000013
laredirect_mux_exp	0													
RegWrite_cancel_exp	0													
> W PC_redirect_exp[31:0]	88000000	00000007	00000080	00000fff	00000	0000	00000054			00000058				
d csr_rw_MEM	0													
d csr_w_imm_mux_MEM	X													
> W CSRout_MEM[31:0]	88000000	00000007	00000080	00000fff	00000	0000	00000054			00000058				
> w exp_vector_WB[3:0]	0							0						



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

