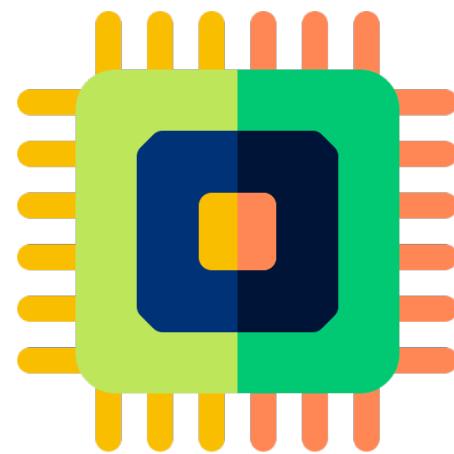


任务

- 掌握支持异常与中断的流水线CPU设计方法。



架构实验2

支持异常与中断的流水线CPU

Chenlu Miao 10/2022

概述

- RISC - V特权级别
- 控制状态寄存器 (CSR)
- CSR指令
- 机器模式特权指令
- 陷阱处理流程
- 支持异常和中断的流水线CPU
- 代码：异常单元

RISC - V特权级别

RISC-V特权级别

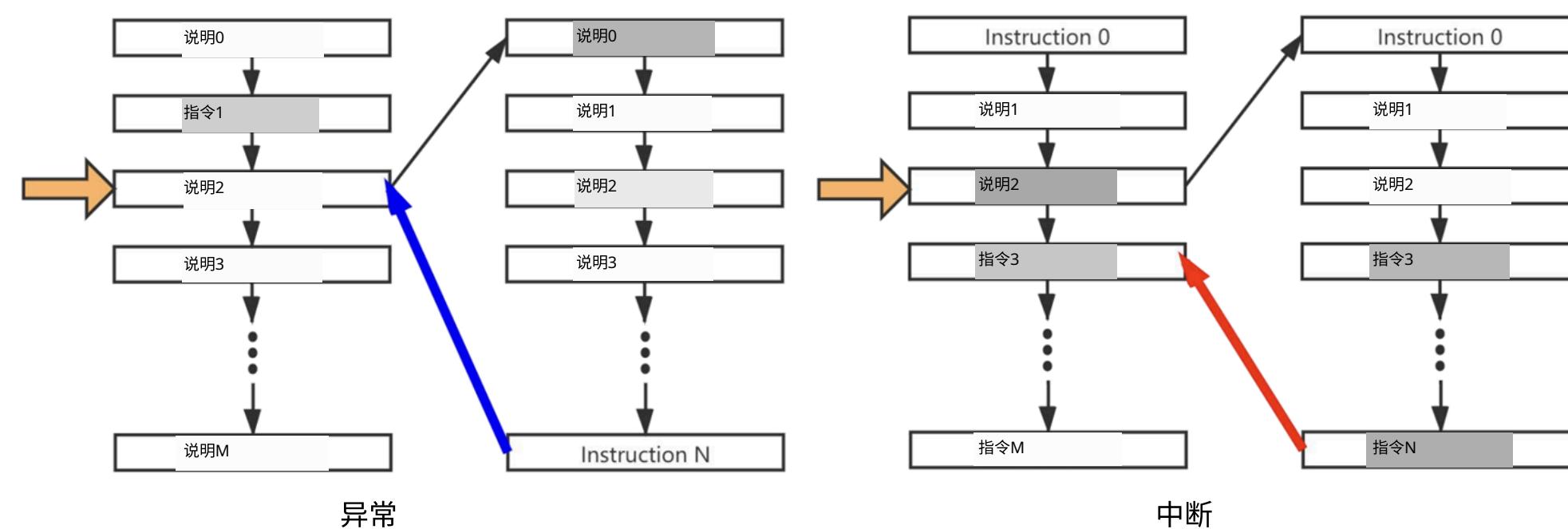


RISC - V特权级别

- 机器模式 (M)
 - 管理程序模式 (H)
 - 监督模式 (S)
 - 用户 (U)
- 陷阱：由异常或**%d**引起的控制转移到陷阱处理程序
中断
• 异常：运行时发生的与**%d**相关的异常情况
当前RISC - V硬件线程中的指令
O中断：一种外部异步事件，可能导致RISC - V硬件线程经历意外的控制转移

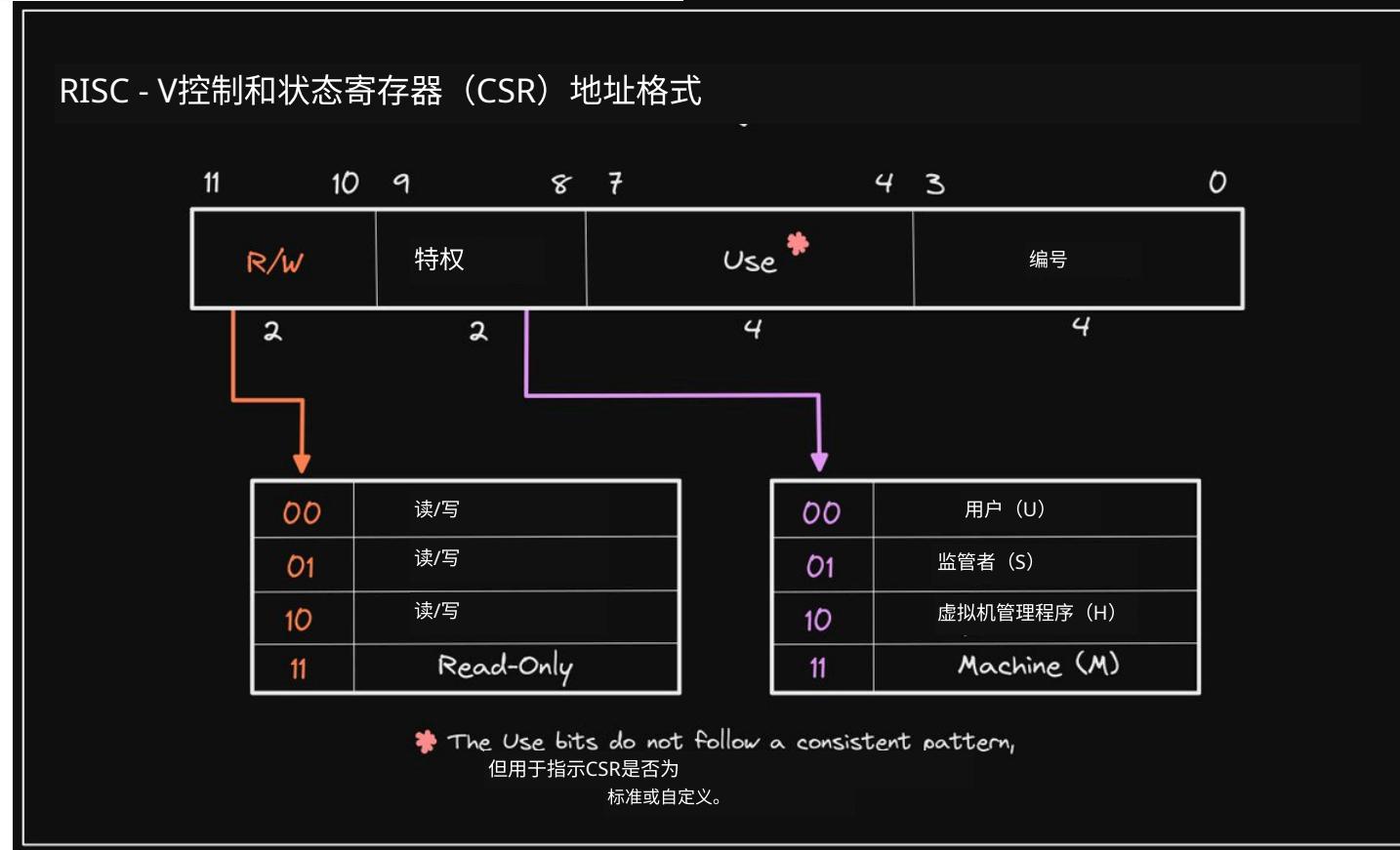
<https://github.com/riscv-non-isa/riscv-trace-spec/issues/10>

RISC - V特权级别



控制状态寄存器

CSR (控制状态寄存器)

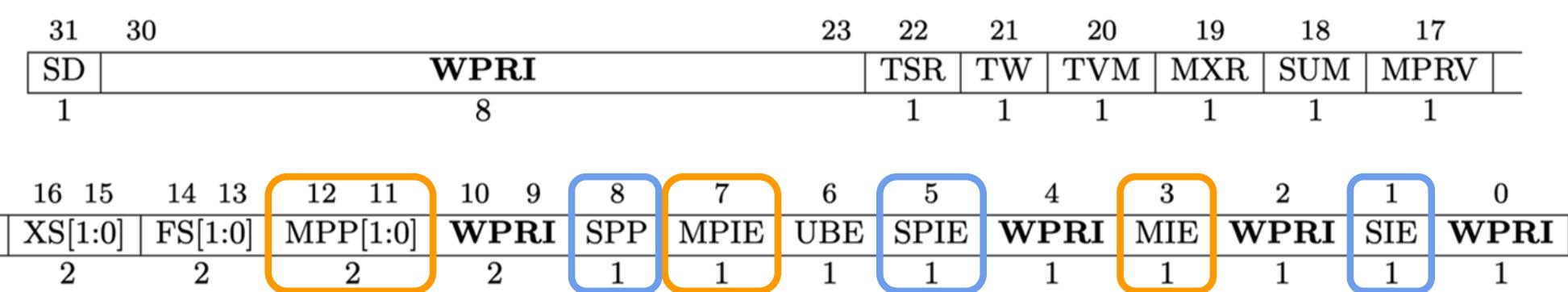


<https://danielmangum.com/risc-v-tips/2021-12-12-access-privilege-csr-addresses/>

CSR (控制状态寄存器)

编号	缩写	名称	描述
0x300	mstatus	机器状态寄存器	处理器状态, mstatus的MIE域和MPIE域用于反映全局中断使能
0x304		机器中断使能寄存器	用于控制不同类型中断的局部中断使能
0x305		机器陷阱向量	定义进入异常的程序PC地址
0x341	mepc	机器异常程序计数器	用于保存异常的返回地址
0x342	mcause	机器原因寄存器	反映进入异常的原因
0x343		机器陷阱值寄存器	反映进入异常的信息
0x344		机器中断待处理寄存器	反映不同类型中断的等待状态

控制和状态寄存器 (CSR): mstatus (机器状态)

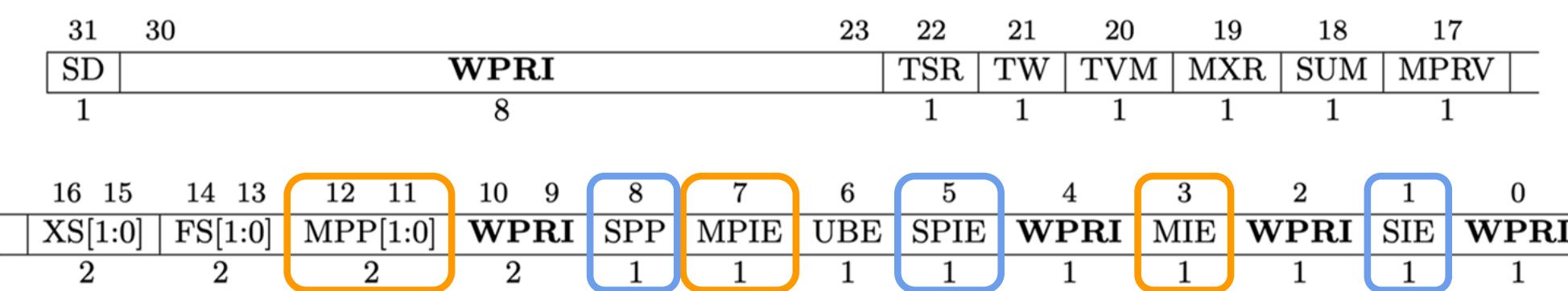


mstatus.xIE: x模式下的中断使能

mstatus.xPIE: x模式下的上一次中断使能

mstatus.xPP: 直至x模式的上一次特权模式

控制和状态寄存器 (CSR): mstatus (机器状态)



mstatus.xIE: x模式下的中断使能

mstatus.xPIE: x模式下的上一次中断使能

mstatus.xPP: 直至x模式的上一次特权模式

- 👉 进入陷阱: mstatus.MPP = 特权级别; mstatus.MPIE = mstatus.MIE; mstatus.MIE = 0;
- 👉 退出陷阱: mstatus.MIE = mstatus.MPIE; mstatus.MPIE = 1; 特权级别 = mstatus.MPP

控制状态寄存器 (CSR): 机器陷阱向量基址寄存器 (mtvec)



模式 = 直接模式:

PC ← 基址

模式 = 向量模式:

(异常) PC ← BASE

(Interrupt) PC ← 基址 + 4 * 原因

控制状态寄存器 (CSR): 机器原因寄存器 (mcause)

中断	异常代码	描述
1	0	保留
1	1	监管者软件中断
1	2	保留
1	3	机器软件中断
1	4	保留
1	5	监管者定时器中断
1	6	Reserved
1	7	机器定时器中断
1	8	保留
1	9	管理模式外部中断
1	10	保留
1	11	机器模式外部中断
1	12-15	Reserved
1	≥16	供平台使用
0	0	指令页未对齐
0	1	指令访问故障
0	2	非法指令
0	3	断点
0	4	加载地址未对齐
0	5	加载访问故障
0	6	存储/原子内存操作 (AMO) 地址未对齐
0	7	存储/原子内存操作 (AMO) 访问故障
0	8	来自用户模式 (U模式) 的环境调用
0	9	来自监督模式 (S模式) 的环境调用
0	10	保留
0	11	来自机器模式 (M模式) 的环境调用
0	12	指令页错误
0	13	加载页错误
0	14	保留
0	15	保留
0	16-23	保留
0	24-31	供自定义使用
0	32-47	保留
0	48-63	指定用于自定义用途
0	≥64	保留

控制状态寄存器 (CSR): 机器异常程序计数器 (mepc)

异常: 机器异常程序计数器 (mepc) ← 当前程序计数器 (PC)

中断: mepc ← 下一代PC

返回: PC ← mepc

为何是当前个人电脑? 为何是下一代个人电脑

• 将? → 软件集恢复到下一代个人电脑

控制状态寄存器 (CSR): 机器异常程序计数器 (mepc)

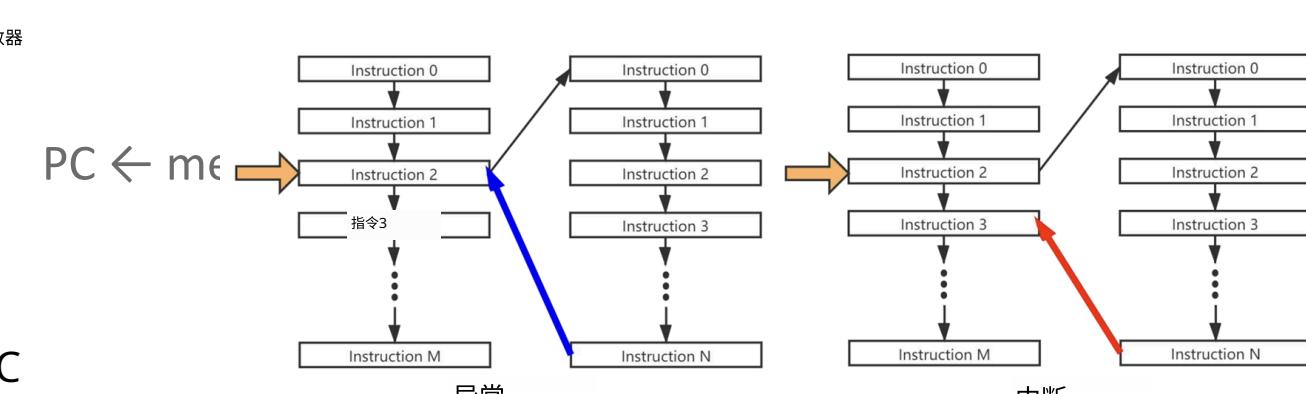
异常: 机器异常程序计数器 (mepc) ← 当前程序计数器 (PC)

中断: mepc ← 下一代程序计数器

返回:

为何是当前PC? 为何是下一代PC

⌚ 将? → 软件设置为下一代PC

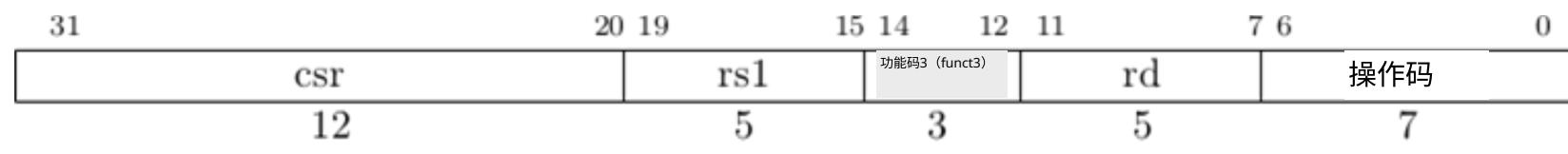


控制和状态寄存器 (CSR) 指令

将一个 CSR[csr] 的值读取到 rd, 然后将 rs1 写入该 CSR	$t \leftarrow \text{CSRs}[csr], \text{CSRs}[csr] \leftarrow x[rs1], x[rd] \leftarrow t$	读取一个 CSR[csr] 的值到 rd, 然后把 rs1 写入该 CSR
从控制和状态寄存器 (CSR) 读取值到目标寄存器 (rd), 源寄存器 (rs1) 参与操作	$t \leftarrow \text{CSRs}[csr], \text{CSRs}[csr] \leftarrow t x[rs1], x[rd] \leftarrow t$	读取一个 CSR 的值到 rd, 然后把该 CSR 中 rs1 指定的 bit 置 1
从控制和状态寄存器 (CSR) 读取值到目标寄存器 (rd), 源寄存器 (rs1) 参与操作 (清除操作)	$t \leftarrow \text{CSRs}[csr], \text{CSRs}[csr] \leftarrow t \& \sim x[rs1], x[rd] \leftarrow t$	读取一个 CSR 的值到 rd, 然后把该 CSR 中 rs1 指定的 bit 置 0
从控制和状态寄存器 (CSR) 读取值到目标寄存器 (rd), 使用立即数 (zimm[4:0])	$x[rd] \leftarrow \text{CSRs}[csr], \text{CSRs}[csr] \leftarrow \text{立即数}$	
CSR 读保留并设置立即数指令, 读取控制和状态寄存器, zimm[4:0]	$x[rd] \leftarrow \text{CSRs}[csr], \text{CSRs}[csr] \leftarrow t zimm$	
CSR 读保留并清零立即数指令, 读取控制和状态寄存器, zimm[4:0]	$x[rd] \leftarrow \text{CSRs}[csr], \text{CSRs}[csr] \leftarrow t \& \sim zimm$	

控制和状态寄存器 (CSR) 指令

控制和状态寄存器 (CSR) 指令

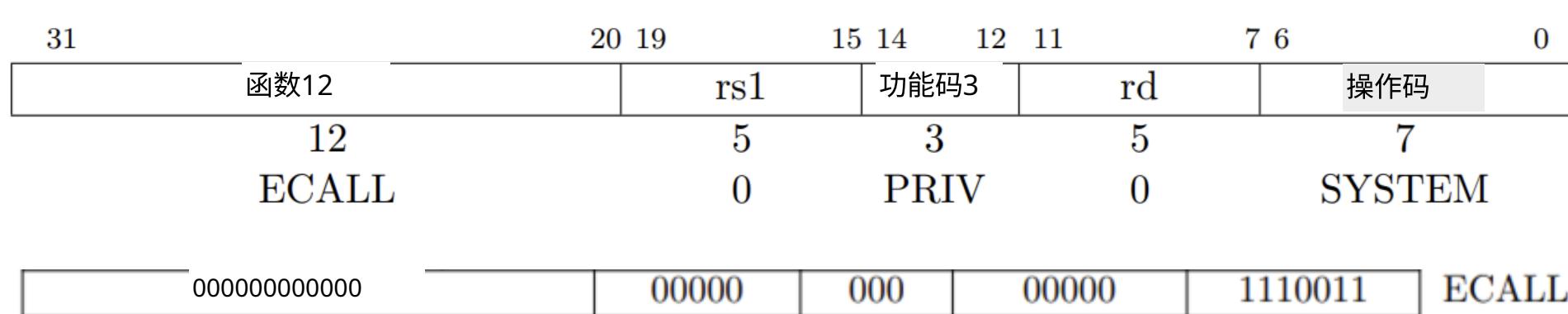


RV32/RV64 Zicsr标准扩展

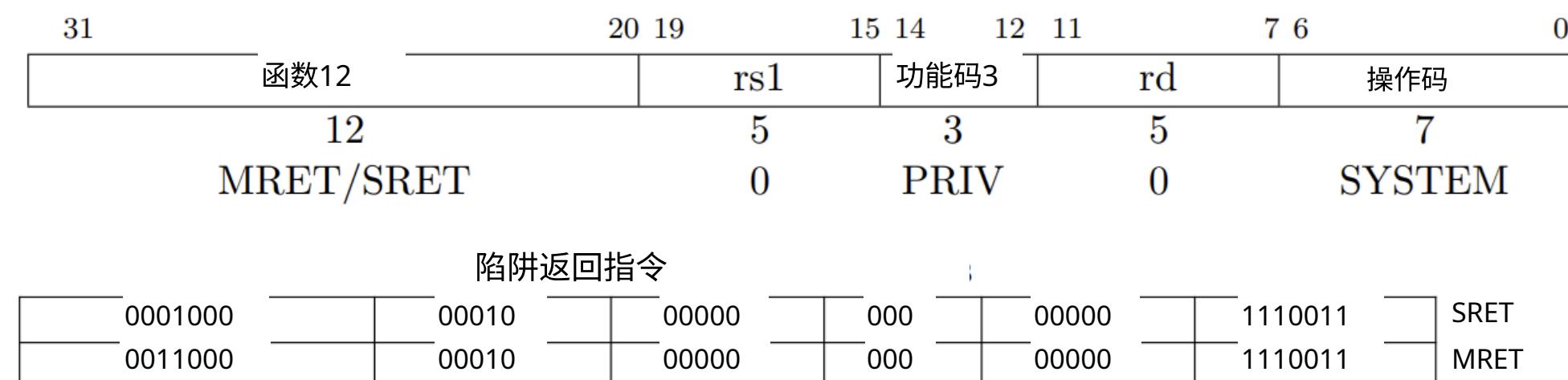
CSR	rs1	001	rd	1110011	CSRRW
控制和状态寄存器 (CSR)	rs1	010	rd	1110011	CSRRS
CSR	rs1	011	rd	1110011	CSRRC
CSR	uimm	101	rd	1110011	CSRRWI
CSR	uimm	110	rd	1110011	CSRRSI
CSR	uimm	111	rd	1110011	CSRRCI

机器模式特权指令

机器模式特权指令 - ECALL



机器模式特权指令 - 机器模式返回指令



退出陷阱: mstatus.MIE = mstatus.MPIE; mstatus.MPIE = 1; 权限 = mstatus.MPP

陷阱处理流程

陷阱处理流程 - 进入陷阱

- 停止当前程序的执行
- 从 CSR mtvec 定义的 PC 地址开始。
- 更新 CSR 寄存器: mcause、mepc 和 mstatus
 - mstatus (mstatus[7]=mstatus[3], mstatus[3]=0)
 - mepc (中断: 下一 PC, 异常: 当前 PC)
 - mcause (中断? 系统调用? 非法指令? 加载故障? 存储故障?)

陷阱处理流程 - 陷阱处理程序

NO.	指令	地址	标签	ASM	注释
30	34102cf3	78	陷阱:	从控制和状态寄存器读取数据到x25, 寄存器地址为0x341	#机器异常程序计数器
31	34202df3	7C		从控制和状态寄存器读取数据到x27, 寄存器地址为0x342	#机器异常原因寄存器
32	30002e73	80		从控制和状态寄存器读取数据到x28, 寄存器地址为0x300	#机器状态寄存器
33	30402ef3	84		从控制和状态寄存器读取数据到x29, 寄存器地址为0x304	#机器中断使能
34	34402f73	88		从控制和状态寄存器读取数据到x30, 寄存器地址为0x344	#机器中断挂起
35	004c8113	8C		将x25的值加4后存入x2	
36	34111073	90		将x2的值写入控制和状态寄存器, 寄存器地址为0x341	
37	30200073	94		机器返回	#30200073 机器返回

陷阱处理流程 - 退出陷阱

- 停止当前程序的执行
- 从CSR mepc定义的PC地址开始。
- 更新CSR mstatus。
 - mstatus (mstatus[3] = mstatus[7], mstatus[7] = 1)

支持异常与中断的流水线CPU

精确异常：

- 故障指令之前的所有指令均执行完毕。
- 故障指令及其后续指令
instruction , 不改变机器状态。

支持异常与中断的流水线CPU

支持异常与中断的流水线CPU

取指阶段 (IF): 内存故障 (非法内存地址)

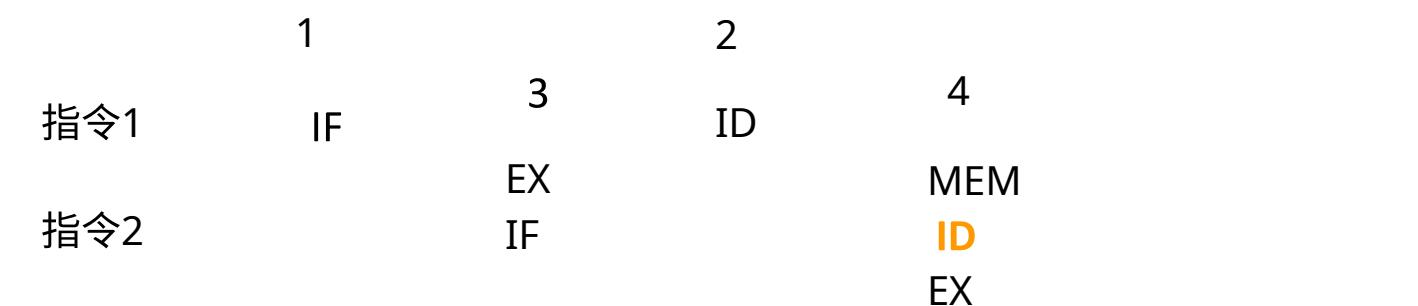
译码阶段 (ID): 非法指令

示例：算术异常

内存：内存故障 (非法内存地址)

WB

支持异常与中断的流水线CPU



支持异常与中断的流水线CPU

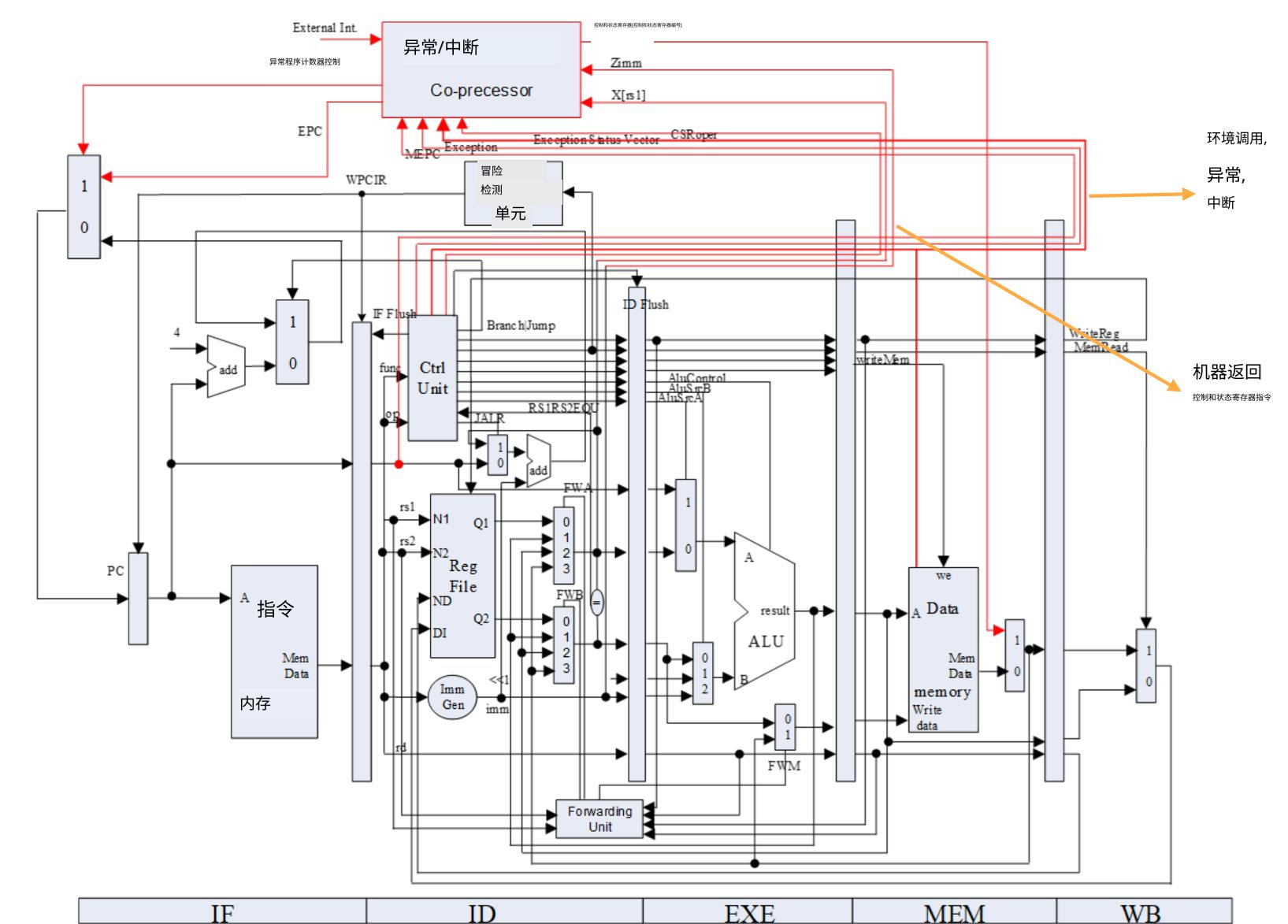
取指阶段：内存故障 (非法内存地址)

译码阶段：非法指令

执行阶段：算术异常

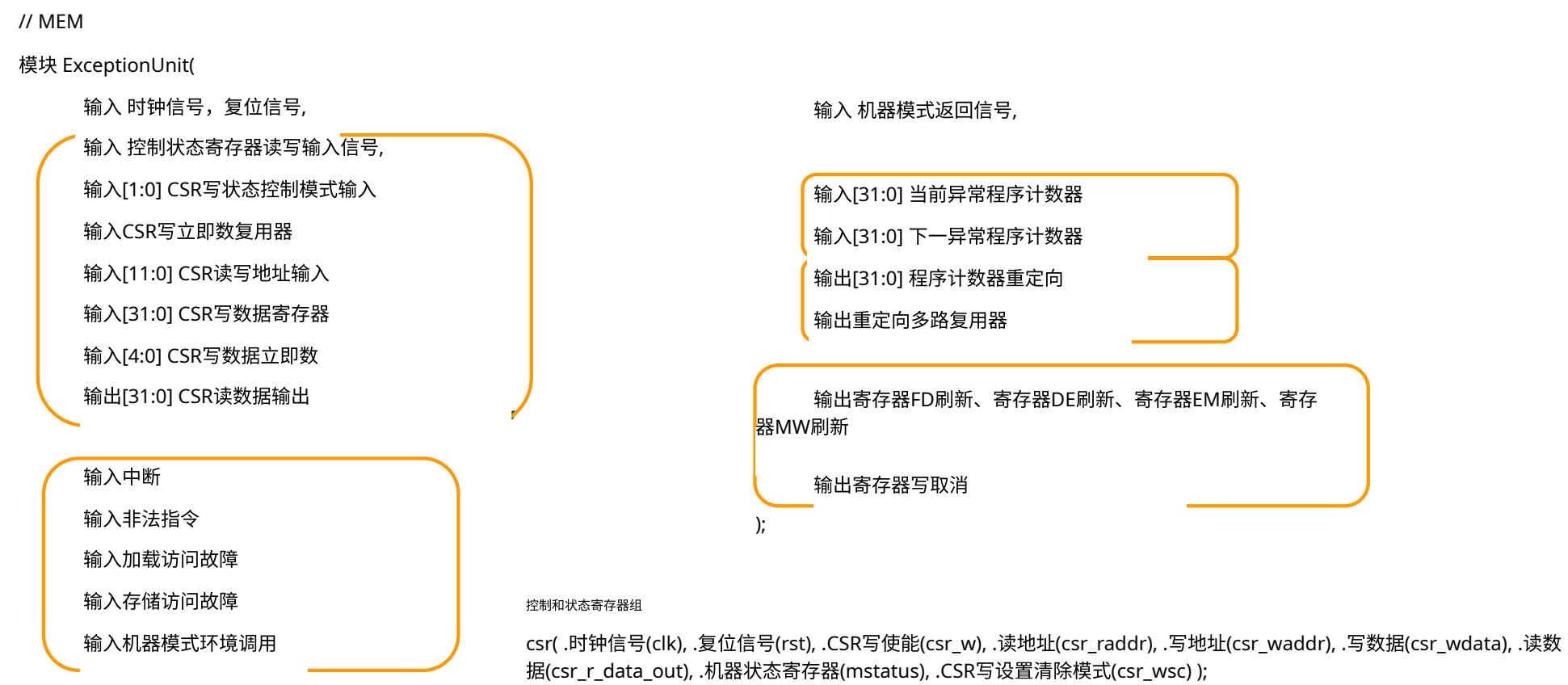
MEM：内存故障 (非法内存地址)

WB

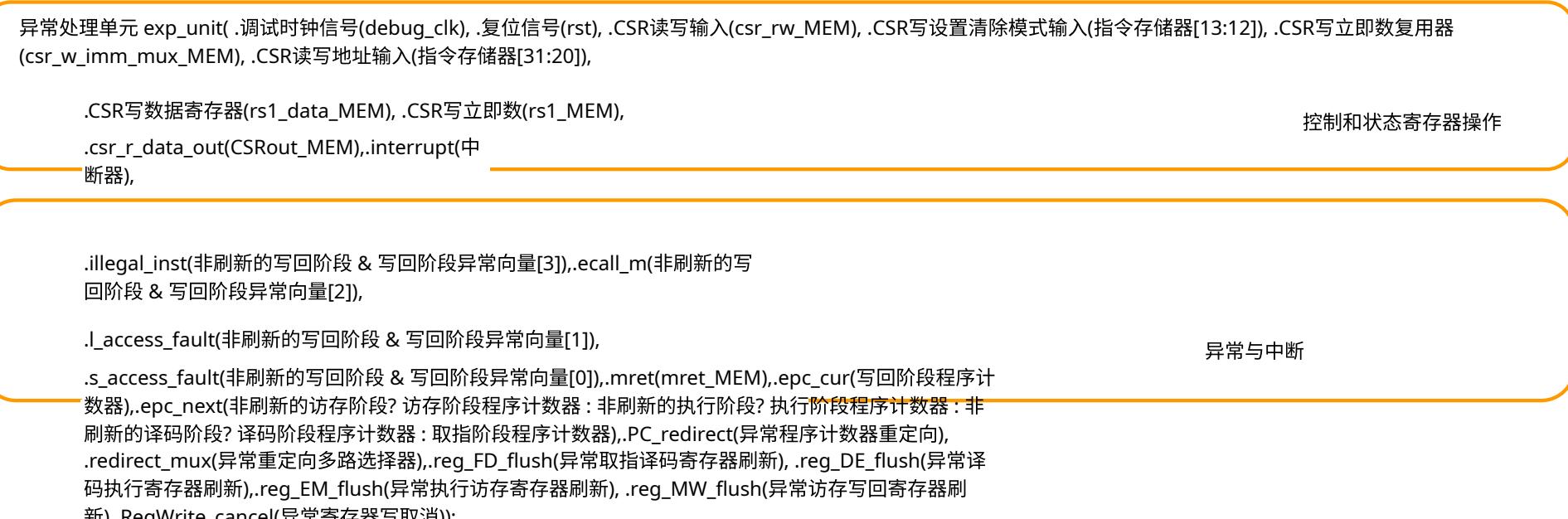


代码：异常单元

异常单元



异常处理单元

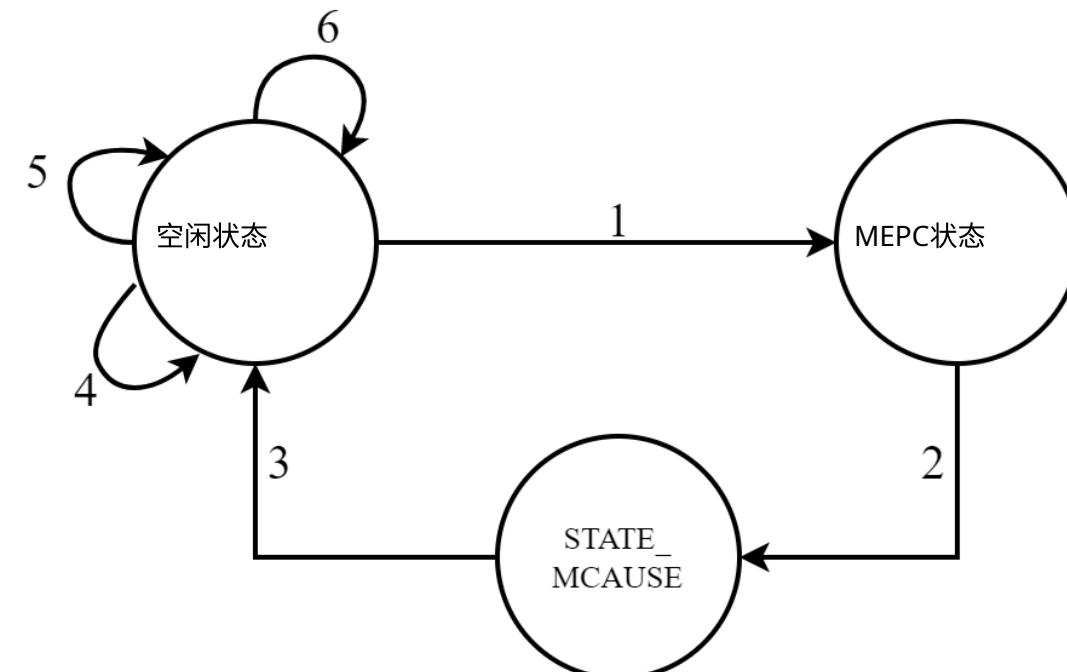


控制和状态寄存器

模块 CSRRegs

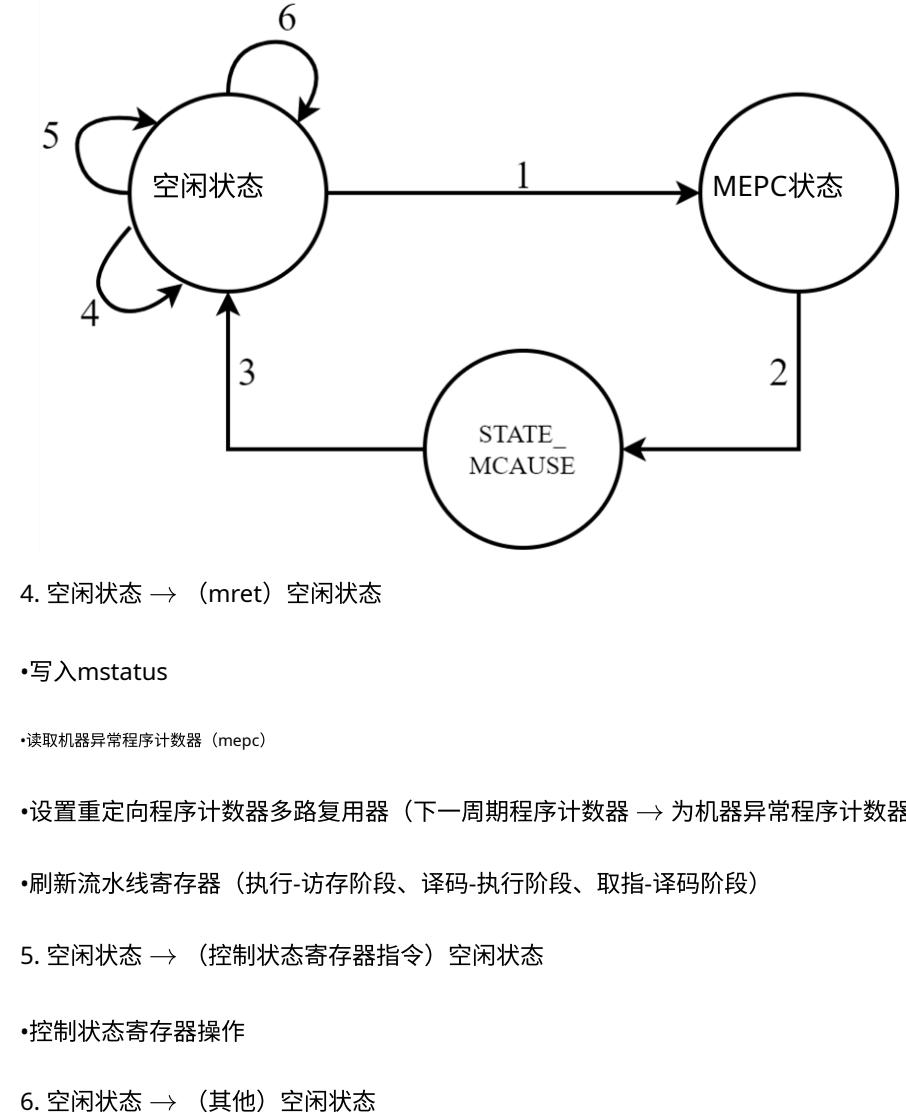
输入 时钟信号, 复位信号, 一个读端口 / 写端口
 输入[11:0] 读地址, 写地址, 读地址 / 写地址
 输入[31:0] 写数据,
 输入 CSR 写信号,
 输入[1:0] CSR写设置清除模式, 写/设置/清除
 输出[31:0] 读数据, 输出[31:0] 机器状态);

异常单元



异常单元

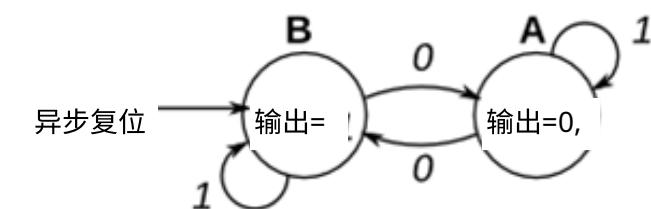
1. 空闲状态 → (异常或中断) MEPC状态 - 写入mstatus - 刷新所有流水线寄存器 - 如果是异常 (非中断), 取消寄存器写入 - 记录epc和原因2. MEPC状态 → MCAUSE状态 - 将epc写入mepc - 读取mtvec - 刷新流水线寄存器 (FD) - 设置重定向PC多路复用器 (下一周期PC → mtvec) 3. MCAUSE状态 → 空闲状态 - 将原因写入mcause



Verilog有限状态机

```
始终 @(*) begin // 这是一个组合始终块// 状态转换逻辑
    // 始终在时钟上升沿时开始
    // 这是一个时序always块// 状态触发器如果(复位) 状态 <= B;否则
    // 状态 <= 下一状态;结束

```



异常单元

本地参数 STATE_IDLE = 2'b00;

本地参数 STATE_MEPC = 2'b01;

始终在时钟信号上升沿触发时开始执行:

当前状态 <= 下一状态;结束

本地参数 STATE_MCAUSE = 2'b10;

寄存器[1:0] 存储当前状态和下一状态;

Save epc and cause in always @(posedge clk)

控制状态寄存器(.时钟信号(clk),复位信号(rst),控制状态寄存器写使能(csr_w),读地址(csr_raddr),写地址(csr_waddr),写数据(csr_w_data),读数据(csr_r_data_out),机器状态寄存器(mstatus),控制状态寄存器写设置清除模式(csr_w_mw),控制状态寄存器写数据(csr_w_data_out),机器状态寄存器(mstatus),控制状态寄存器写设置清除模式(csr_w_mw),控制状态寄存器写数据(csr_w_data_out))

异常处理单元

始终 @* 开始 情况(当前状态) 空闲

状态: 开始

```

如果 (陷阱进入) 开始
    控制和状态寄存器...
    下一状态 = ...
结束
否则如果 (机器返回指令) 开始
    控制和状态寄存器 ...
    下一状态 = ...
结束
否则, 如果 (csr读写输入) 开始
    控制状态寄存器 ...
    下一状态 = ...
结束
否则开始
    循环冗余校验寄存器 (CSR) 写操作值为0;
    下一状态 = 空闲状态;结束结束

```

异常处理单元

赋值PC重定向 = 控制和状态寄存器读取数据输出;

赋值 重定向多路复用器 = ...

赋值寄存器MW刷新 = ...

赋值寄存器EM刷新 = ...

赋值寄存器DE刷新 = ...

赋值 reg_FD_flush = ...

赋值 RegWrite_cancel = ...

R
O
M

NO.	指令	地址	标签	ASM	注释
0	00000013	0	—开始:	将立即数0加到x0寄存器, 结果存于x0	
1	00402103	4		从x0寄存器的值加4的地址处加载字到x2寄存器	
2	00802203	8		从x0寄存器的值加8的地址处加载字到x4寄存器	
3	00c02283	C		从x0寄存器的值加12的地址处加载字到x5寄存器	
4	01002303	10		从x0寄存器的值加16的地址处加载字到x6寄存器	
5	01402383	14		将内存地址为x0 + 20处的数据加载到x7寄存器	
6	306850f3	18		将立即数16写入控制状态寄存器0x306, 并将结果存于x1寄存器	
7	306020f3	1C		从控制状态寄存器0x306读取数据到x1寄存器	从控制状态寄存器读取数据到x1寄存器
					0x306, x0
8	306310f3	20		将x6寄存器的值写入控制状态寄存器0x306, 并将寄存器值存于x1寄存器	
9	306020f3	24		从控制状态寄存器0x306读取数据到x1寄存器	
10	00000013	28		将立即数0加到寄存器x0并将其结果存于x0	
11	07800093	2C		将立即数120加到寄存器x0并将其结果存于x1	
12	30509073	30		将寄存器x1的值写入控制状态寄存器0x305	设置
					机器模式异常向量基址寄存器设置为0x78
13	00000013	34		将立即数0加到寄存器x0并将其结果存于x0	
14	00000073	38		回叫	

NO.	指令	地址	标签	ASM	注释
15	00000013	3C		将立即数0加到寄存器x0, 并将其结果存于x0	
16	00000012	40		将立即数0加到寄存器x0, 并将其结果存于x0	# 更改为非法操作
17	00000013	44		将立即数0加到寄存器x0, 并将其结果存于x0	
18	07f02083	48		从地址x0 + 128处加载字数据到寄存器x1	
19	08002083	4C		从地址x0 + 128处加载字数据到寄存器x1	#I访问故障
20	00000013	50		将立即数0加到寄存器x0并将其结果存于x0	
21	08102023	54		将寄存器x1的值存储到寄存器x0的值加128为地址的内存位置	#S访问故障
22	00000013	58		将立即数0加到寄存器x0并将其结果存于x0	
23	00000013	5C		将立即数0加到寄存器x0, 并将其结果存于x0	
24	00000013	60		将立即数0加到寄存器x0, 并将其结果存于x0	
25	00000013	64		将立即数0加到寄存器x0, 并将其结果存于x0	
26	00000013	68		将立即数0加到寄存器x0, 并将其结果存于x0	
27	00000013	6C		将立即数0加到寄存器x0, 并将其结果存于x0	
28	00000013	70		将立即数0加到寄存器x0, 并将其结果存于x0	
29	00000067	74		跳转到寄存器x0所指向的地址	

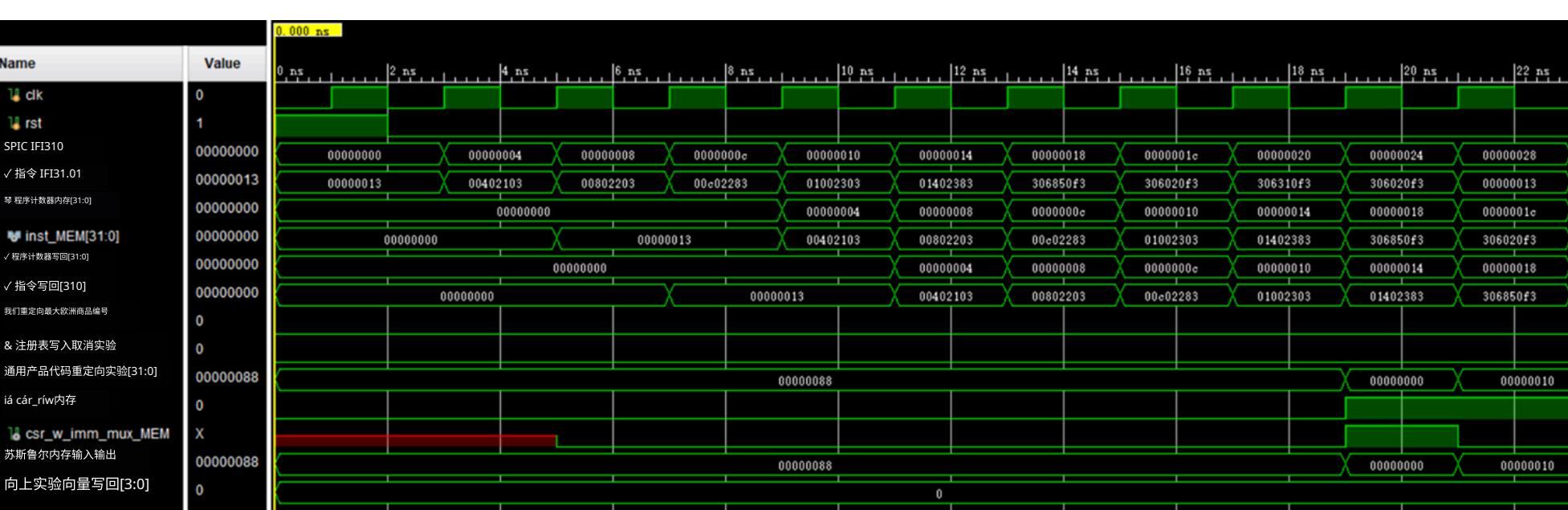
设置mtvec
后测试中断

NO.	指令	地址	标签	ASM	注释
			陷阱:	将控制状态寄存器0x341的值读取到x25寄存器	#机器异常处理上下文
				从控制和状态寄存器读取数据到x27, 地址为0x342	#机器异常原因
				从控制和状态寄存器读取数据到x28, 地址为0x300	#机器状态
				从控制和状态寄存器读取数据到x29, 地址为0x304	#mie
				从控制状态寄存器读取数据到x30寄存器, 地址为0x344	#mip
35	004c8113	8C		将x25寄存器的值加4后存入x27寄存器	
36	34111073	90		将x27寄存器的值写入控制状态寄存器, 地址为0x341	#机器异常程序计数器 - 机器异常程序计数器 + 4
37	30200073	94		机器返回	#30200073 机器返回
38	00000013	98		将立即数0加到寄存器x0, 并将其结果存于x0	
39	00000013	9C		将立即数0加到寄存器x0, 并将其结果存于x0	
40	00000013	A0		将立即数0加到寄存器x0, 并将其结果存于x0	
41	00000013	A4		将立即数0加到寄存器x0, 并将其结果存于x0	

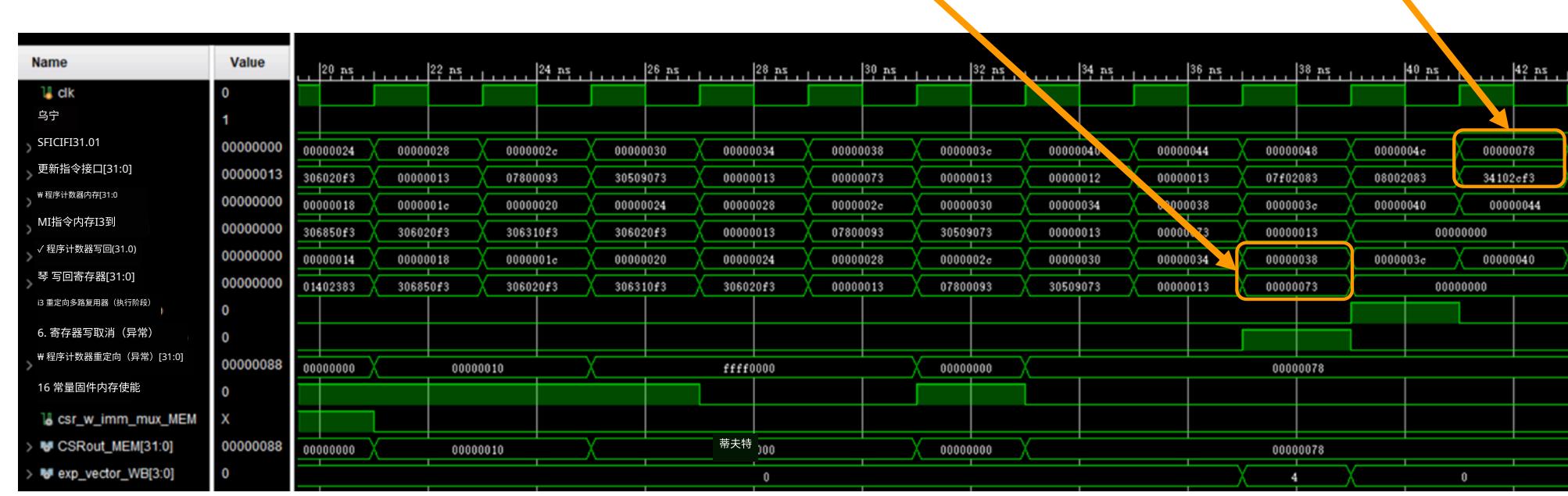
RRAM

NO.	数据	地址
0	000080BF	0
1	00000008	4
2	00000010	8
3	00000014	C
4	FFFFF000	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

模拟 (1)



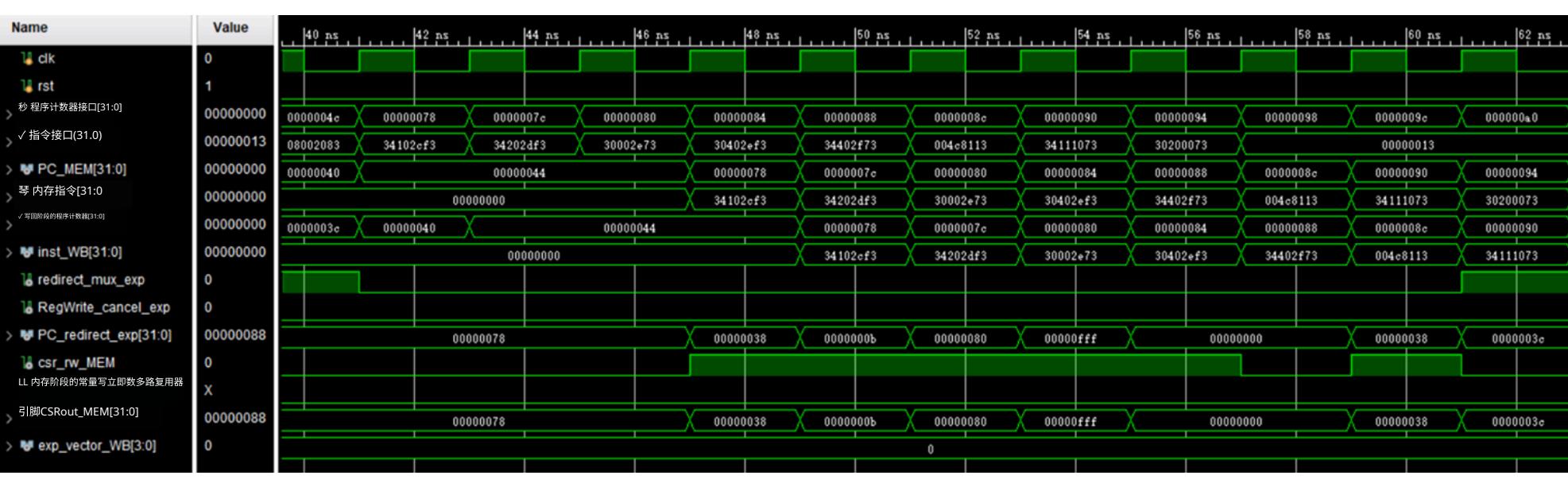
模拟 (2)



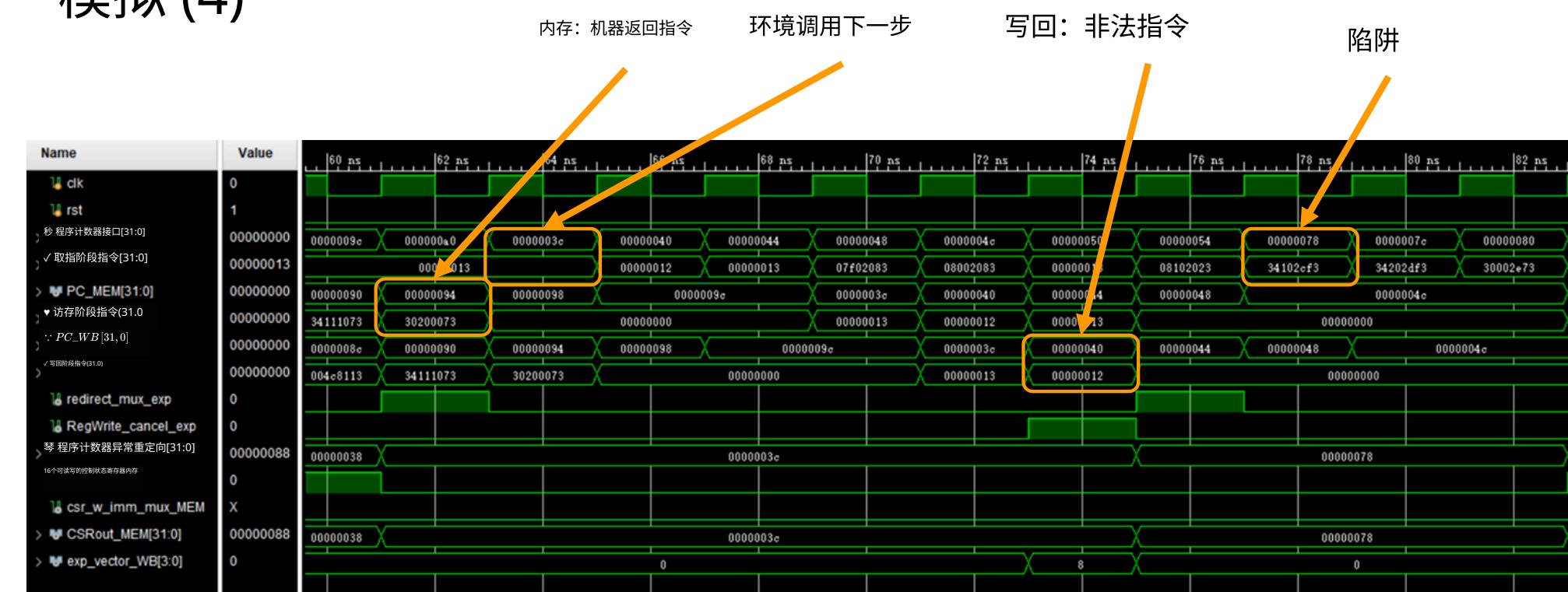
Trap

WB: 远程调用

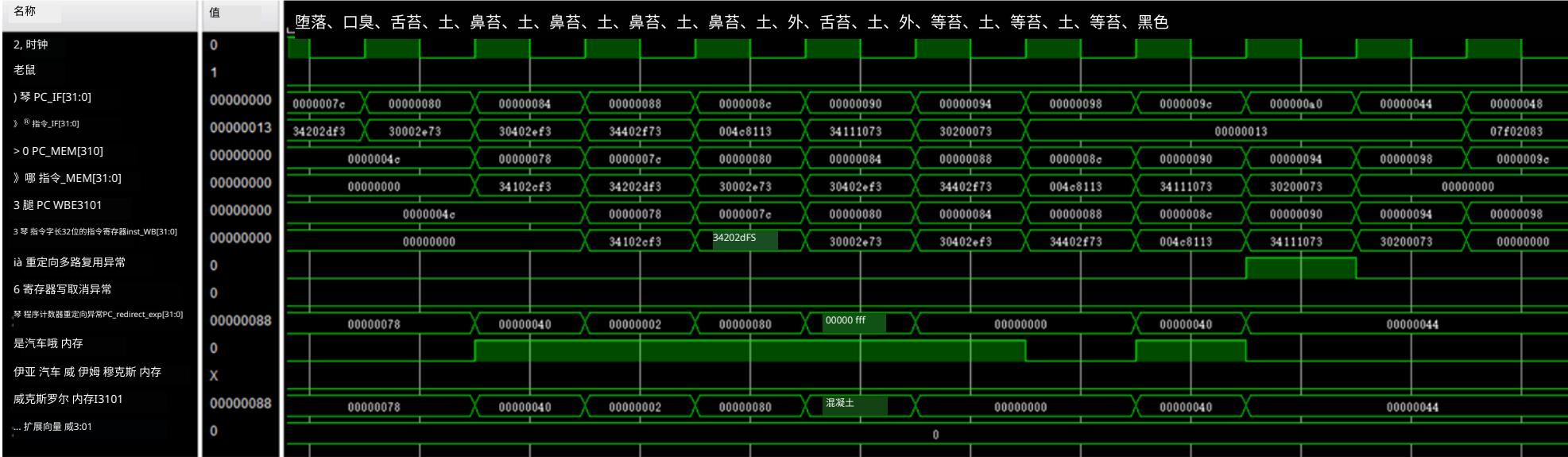
模拟 (3)



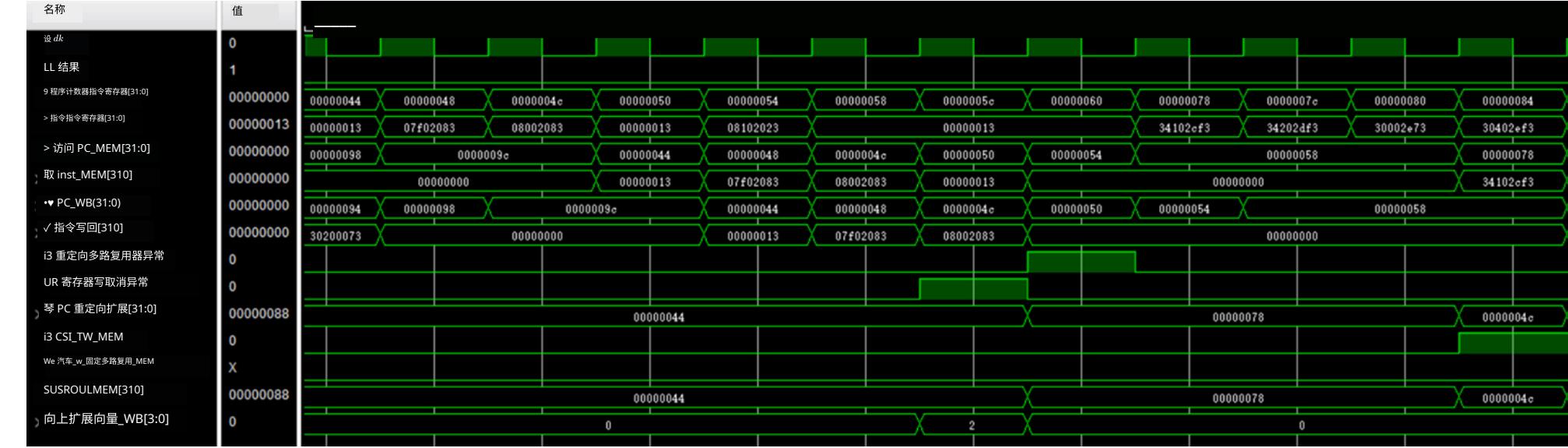
模拟 (4)



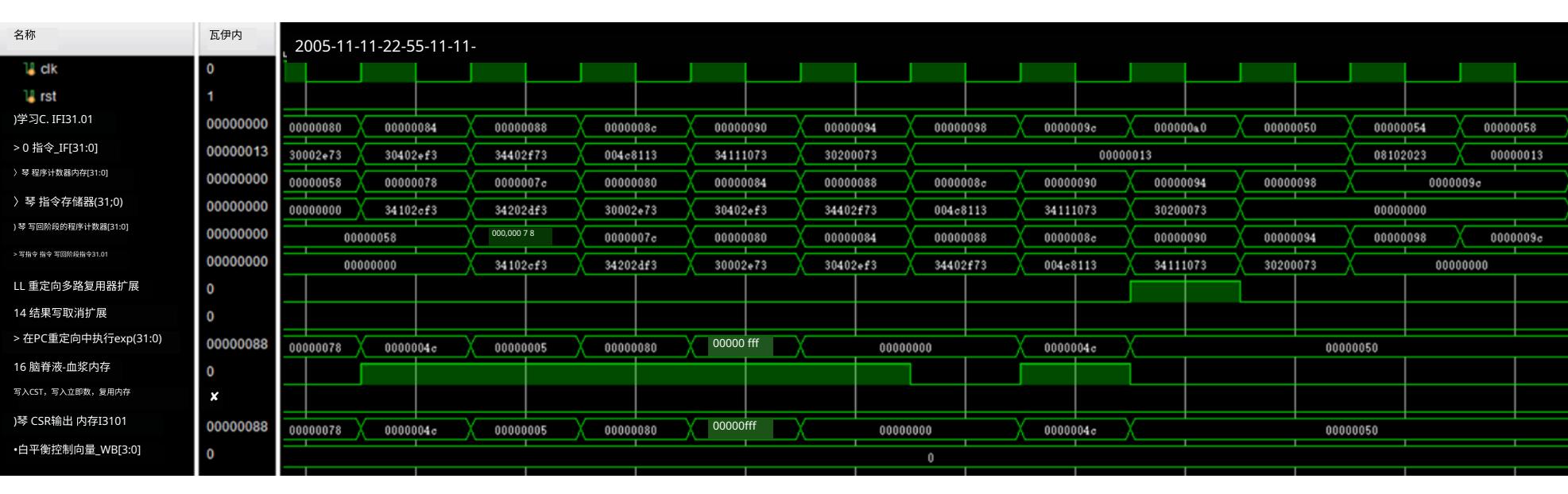
Simulation (5)



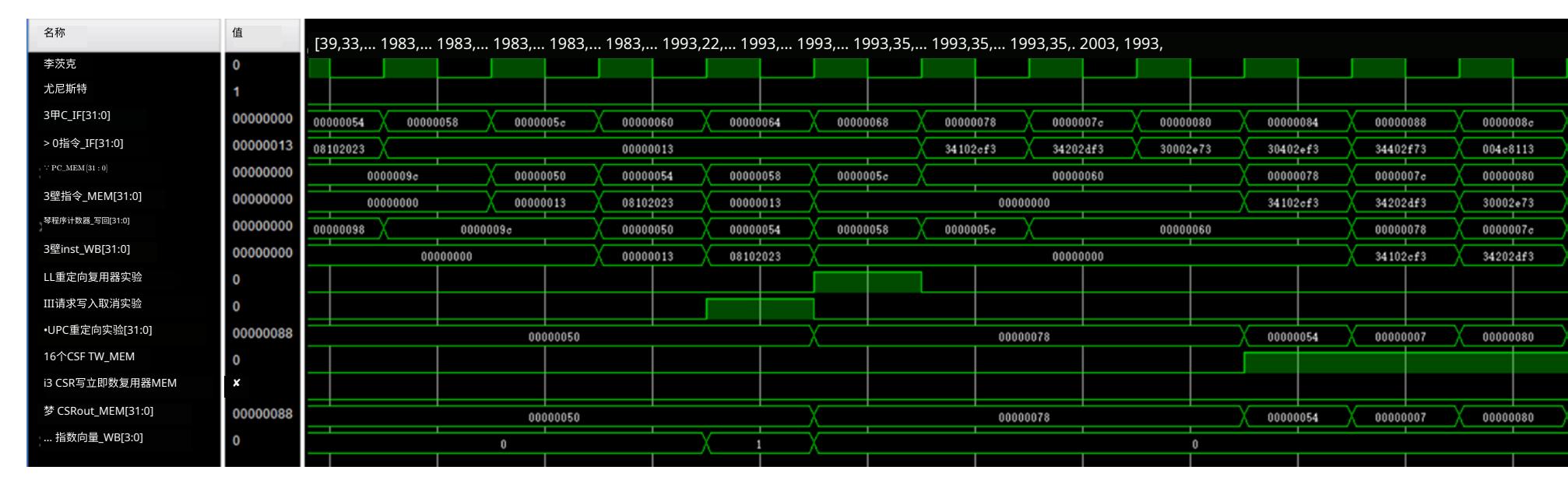
仿真 (6)



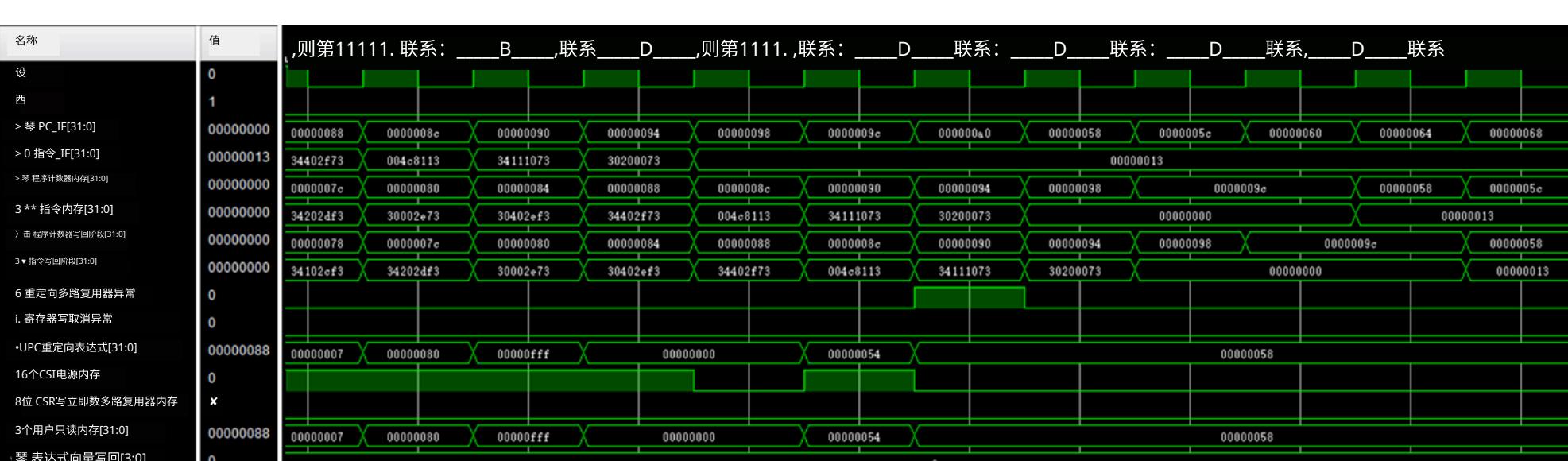
Simulation (7)



仿真 (8)



仿真 (9)



SW[12] 中断

Code2Inst.v CSR指令

参考文献

- <https://danielmangum.com/posts/risc-v-bytes-privilege-levels/>
- <https://www.notion.so/MODE-e78a2091e1fd4de199a63262508d15e5#0cb123be2cc3471d9990288c5ba1cd6c>
- <https://doc.nucleisys.com/nuclei/spec/isa/exception.html>
- <https://github.com/plctlab/riscv-operating-system-mooc/blob/main/slides/ch02-riscv-isa-introduction.pdf>
- <https://groups.google.com/g/comp.lang.verilog/c/2X9f9ds9XnE>