

# Arch2025 RISC-V Lab4

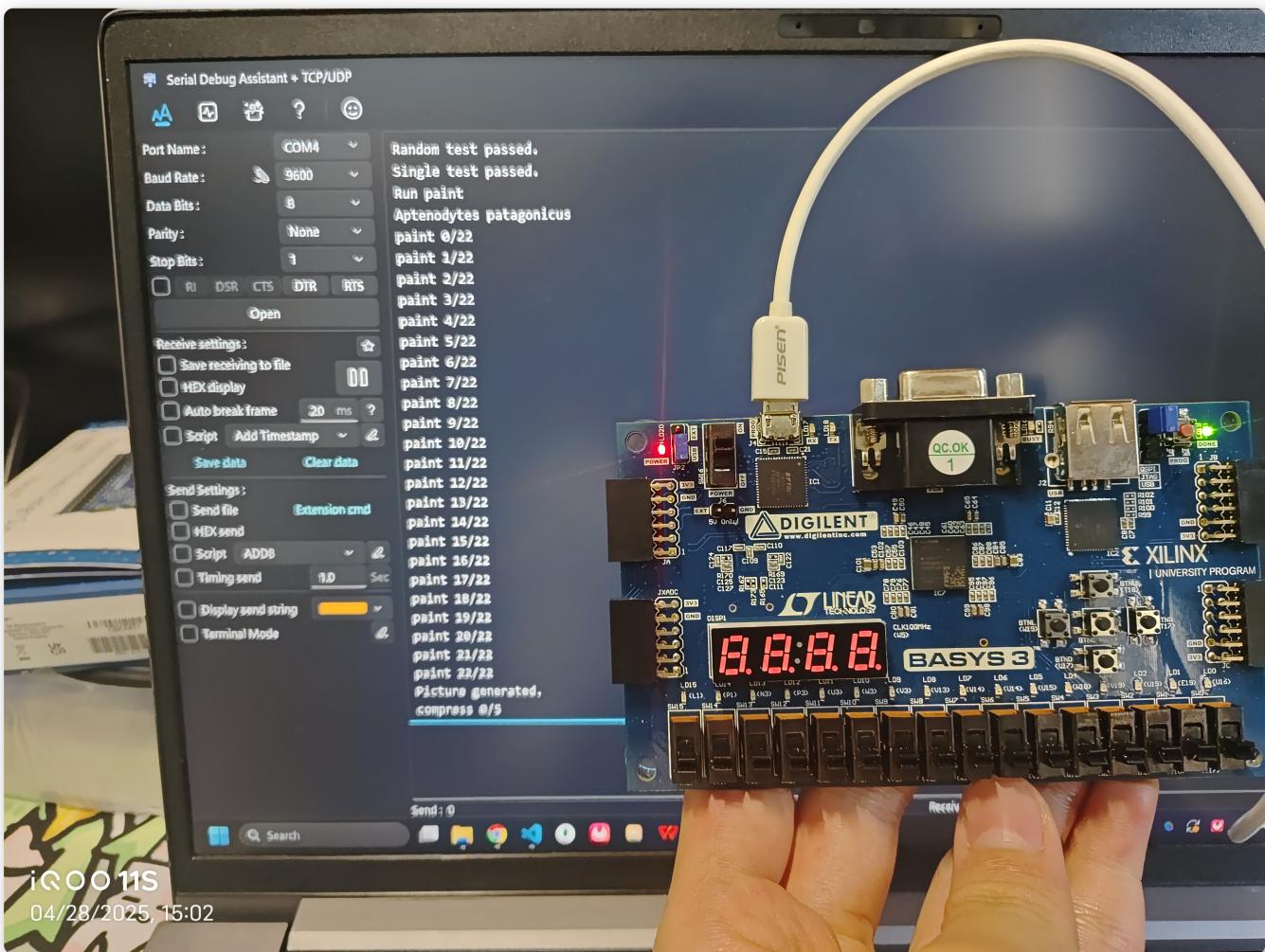
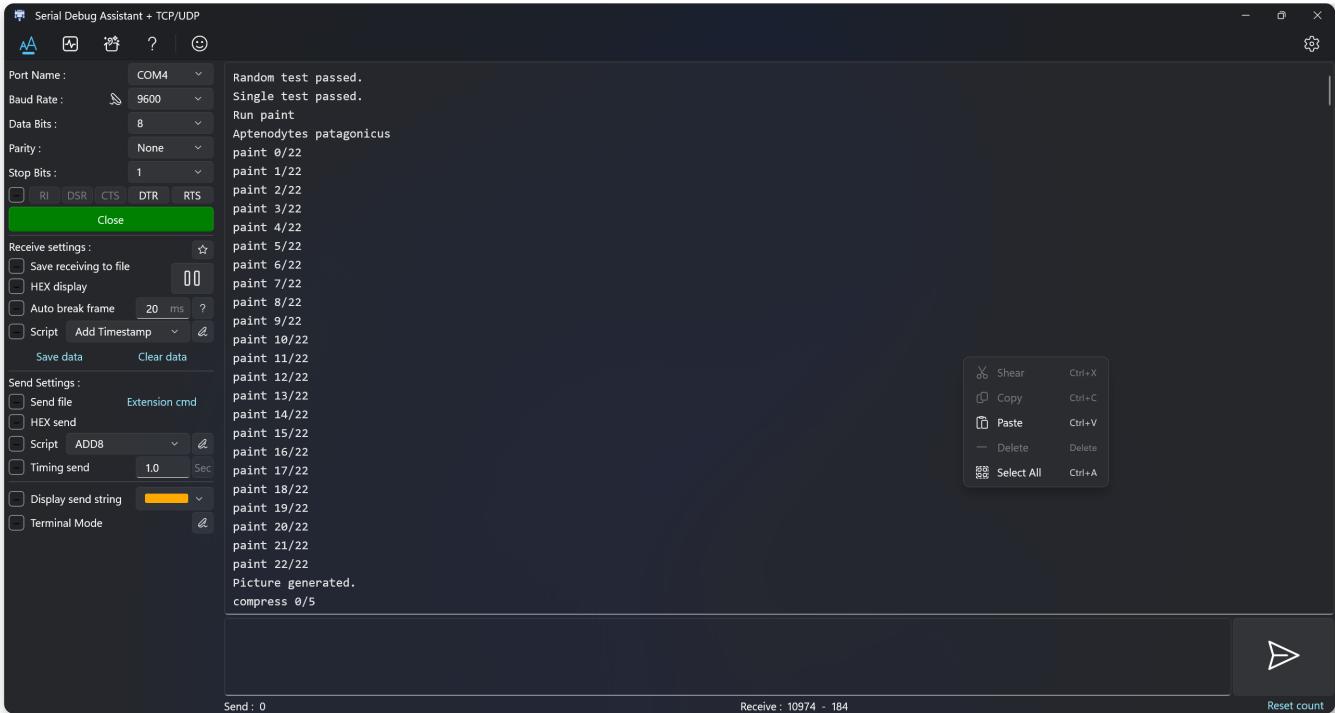
Title	Student Name	Student ID	Date
CSR Registers	Zecyel (朱程炀)	23300240014	2025.4.29

## 1. 实验要求&运行结果

- 实现一个支持 CSRRW, CSRRS, CSRRC, CSRRWI, CSRRSI, CSRRCI 指令和 mstatus, mtvec, mip, mie, mscratch, mcause, mtval, mepc, mcycle, mhartid, satp 寄存器的五级流水线 CPU。
- 在 Vivado 上进行仿真，并在仿真通过后在 FPGA 板上进行测试。

```
TEST= ./build/emu --diff /mnt/c/Users/Zecyel/Desktop/arch/arch-2025/ready-to-run/riscv64-nemu-interpreter
Emu compiled at Feb 20 2025, 15:59:20
The image is ./ready-to-run/lab4/lab4-test.bin
Using simulated 256MB RAM
Using /mnt/c/Users/Zecyel/Desktop/arch/arch-2025/ready-to-run/riscv64-nemu-interpreter-so for difftest
[src/device/io/mmio.c:19,add_mmio_map] Add mmio map 'clint' at [0x38000000, 0x3800ffff]
[src/device/io/mmio.c:19,add_mmio_map] Add mmio map 'uartlite' at [0x40600000, 0x4060000c]
[src/device/io/mmio.c:19,add_mmio_map] Add mmio map 'uartlite1' at [0x23333000, 0x2333300f]
The first instruction of core 0 has committed. Difftest enabled.
[WARNING] difftest store queue overflow
[src/cpu/cpu-exec.c:393,cpu_exec] nemu: HIT GOOD TRAP at pc = 0x000000008001ffff8
[src/cpu/cpu-exec.c:394,cpu_exec] trap code:0
[src/cpu/cpu-exec.c:74,monitor_statistic] host time spent = 5426 us
[src/cpu/cpu-exec.c:76,monitor_statistic] total guest instructions = 32766
[src/cpu/cpu-exec.c:77,monitor_statistic] simulation frequency = 6038702 instr/s
Program execution has ended. To restart the program, exit NEMU and run again.
sh: 1: spike-dasm: not found

===== Commit Group Trace (Core 0) =====
commit group [0]: pc 008001ffc0 cmtcnt 1
commit group [1]: pc 008001ffc4 cmtcnt 1
commit group [2]: pc 008001ffc8 cmtcnt 1
commit group [3]: pc 008001ffcc cmtcnt 1
```



## 2. 代码实现

就像实现另外的十几个寄存器一样就行了。但是由于这些 CSR 寄存器会在后面的 lab 中被特殊使用，所以就不把它和之前的寄存器混在一起了。

由于需要处理 mcycle 这一个特殊寄存器，所以可以这么写，在mcycle没有被写入的时候，不管是否使能了写 CSR 寄存器的信号，都将 mcycle 寄存器自增。

```
always_ff @(posedge clk or posedge rst) begin
    if (rst) begin
        csr_reg.mstatus <= 0;
        ...
    end else if (! csr_write_enable)
        csr_reg.mcycle <= csr_reg.mcycle + 1;
    else if (csr_write_enable) begin
        if (csr_dest_addr != CSR_MCYCLE)
            csr_reg.mcycle <= csr_reg.mcycle + 1;
        unique case (csr_dest_addr)
            CSR_MSTATUS: csr_reg.mstatus <= csr_write_data &
MSTATUS_MASK;
            ...
        endcase
    end
end
```

有一个槽点在于，CSR 的立即数指令和 I-type 指令不太一样，会把原来五位的寄存器地址用作立即数。

```
alu alu_inst (
    .op1(csr_immed ? { 59'b0, id_ex_state.reg1_addr } : op1),
    ...
);
```

刷新流水线的实现：从 lab3 开始，维护了一个“需要阻塞的指令”列表，遇到诸如 jalr, csrrw 等指令，fetch 模块就会一直等待这条指令执行完，避免本不应该执行的指令被加入流水线。

其余逻辑和写普通寄存器一样即可。

## 3. Bonus

### 3.1 CSR 寄存器作用

#### 3.1.1 mstatus

63	62	38	37	36	35	34	33	32	31	23	22	21	20	19	18
SD	WPRI	MBE	SBE	SXL[1:0]	UXL[1:0]		WPRI		TSR	TW	TVM	MXR	SUM		
1	25	1	1	2	2		9		1	1	1	1	1	1	
17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2
MPRV	XS[1:0]	FS[1:0]	MPP[1:0]	VS[1:0]	SPP	MPIE	UBE	SPIE	WPRI	MIE	WPRI	SIE	WPRI		0
1	2	2	2	2	1	1	1	1	1	1	1	1	1	1	

Figure 3.7: Machine-mode status register (`mstatus`) for RV64.

管理 CPU 状态的一个总的寄存器。例如，MIE 是全局的 M 模式中断开关，TVM 是全局的陷阱控制开关。

### 3.1.2 mtvec

最低两位 MODE 表示陷阱处理模式。剩余 62 位 BASE 是陷阱处理程序的入口地址（4 字节对齐）。

- MODE = 0 时，所有的陷阱都由 BASE 位置的程序处理。
- MODE = 1 时，由 BASE + cause \* 4 处的程序处理。

### 3.1.3 mip / mie

全称是 Machine Interrupt-Pending Register 和 Machine Interrupt-Enable Register。其中 MIE 寄存器是中断使能寄存器，表示当前有哪些中断可以被触发（前提是 `mstatus.MIE = 1`），包含了软件中断，外部中断和计时器中断，以及系统设置的中断。MIP 寄存器保存了当前有哪些中断已经触发但尚未处理。

### 3.1.4 mscratch / mepc

两个陷阱临时寄存器。

- mscratch 用于在陷阱处理中保存 sp。
- mepc 用于在陷阱处理中保存 pc。在 mret 的时候就用

### 3.1.5 mcause

最高位为 1 的时候表示陷阱的原因是中断，为 2 的时候是异常。后面的 63 位记录了陷阱的原因。可以根据表格查出对应的陷阱原因。

### 3.1.6 mtval

记录异常发生时的相关附加参数。当内存访问异常时，mtval 会保存触发异常的虚拟内存地址。当非法指令异常时，mtval 会保存故障指令。

### 3.1.7 mcycle

每个cpu时钟周期就计数，可以用于性能监测。

### 3.1.8 mhartid

标识硬件线程(hart)的唯一ID。例如，多核系统中，核0的mhartid=0，核1的mhartid=1。

### 3.1.9 satp

Supervisor Address Translation and Protection Register，用于S模式下的虚拟内存转义和保护。

63	60 59	44 43	0
MODE (WARL)	ASID (WARL)	PPN (WARL)	
4	16	44	

Figure 4.15: Supervisor address translation and protection register satp when SXLEN=64, for MODE values Bare, Sv39, Sv48, and Sv57.

MODE = 0 时表示不启用虚拟地址。MODE = 8~10 时可以设置页的长度。

## 3.2 为什么要在 CSR 指令刷新流水线

首先是，CSR指令会同时修改CSR寄存器和普通寄存器，如何不进行冲刷，会破坏CSR指令的原子性，暴露两次拷贝的中间状态。

其次，如果在CSR指令执行完之后，发生了中断/异常，之后可能需要跳转到其它地方执行。这时候就应该冲刷掉原本已经被加入流水线的指令。

## 4. 实验心得

助教们都很帅！很负责！很厉害！要是能给我的实验报告打高分就更帅了！