易 灵 思

RISCV 培训

进阶篇

8/30/2023

CSR寄存器

- Control and Status Registers
  - 特权级配置
    - M-Mode
      - 最高权限，没有任何限制
    - S-Mode
      - 特权级，通常和用户级同时使用，用于实现现代操作系统权限管理
    - U-Mode
      - 用户级，大部分应用程序运行在次特权，CSR寄存器访问受限
  - PMP
    - 内存存取权限配置
  - Mtvec/Mtval
    - 异常/错误抓取配置

mcause寄存器 – 中断

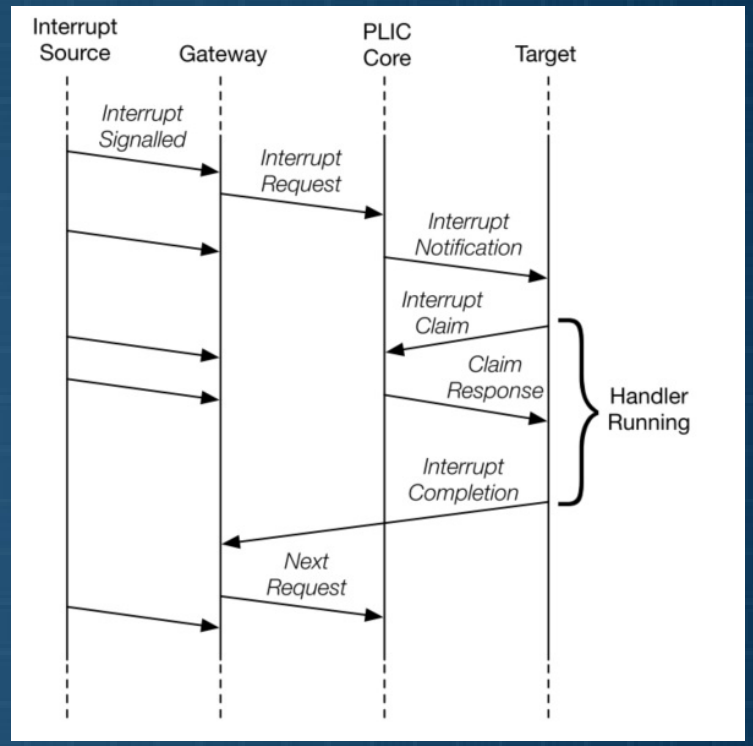
- RISCV的中断类型：
  - 外部中断
    - 最常用的外部输入中断
  - 计时中断
    - 通常用于分时操作
  - 软中断
    - 通常用于调度操作
- RISCV中断处理流程
  - 配置mtvec寄存器，指向trap_entry函数
  - 配置mie寄存器启用位MIE_MEIE
  - 用户逻辑->Plic->mcause寄存器

Interrupt	Exception Code	Description
1	0	*Reserved*
1	1	Supervisor software interrupt
1	2	*Reserved*
1	3	Machine software interrupt
1	4	*Reserved*
1	5	Supervisor timer interrupt
1	6	*Reserved*
1	7	Machine timer interrupt
1	8	*Reserved*
1	9	Supervisor external interrupt
1	10	*Reserved*
1	11	Machine external interrupt
1	12–15	*Reserved*
1	≥16	*Designated for platform use*

PLIC

- PLIC配置
  - 配置mtval指向异常处理入口（trap)
  - 配置mie寄存器，启用异常抓取
  - 配置mstatus掩码，抓取mie寄存器状态

```
csr_write(mtvec, trap_entry);
csr_set(mie, MIE_MEIE);
csr_write(mstatus, MSTATUS_MPP | MSTATUS_MIE);
```

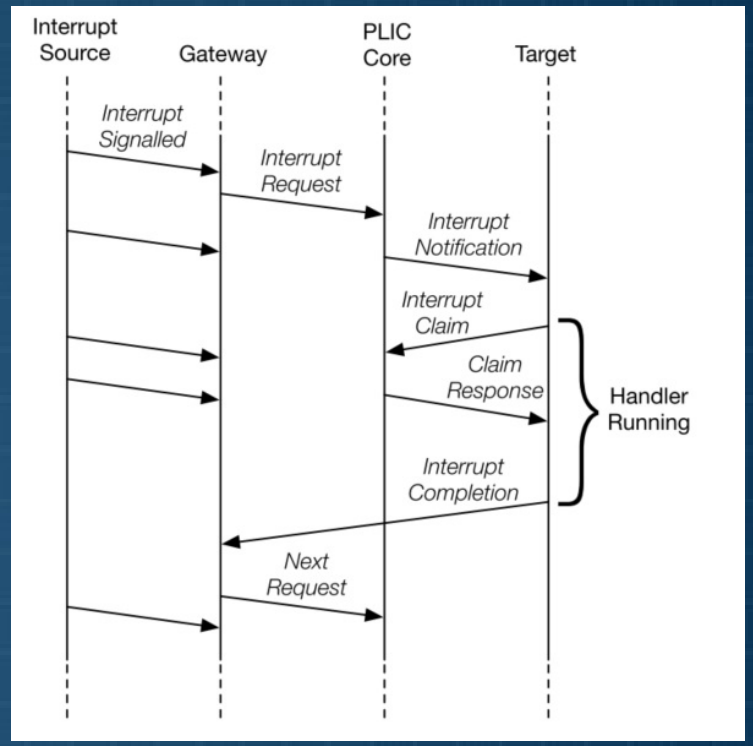


PLIC

- Handler = trap函数
  - 获取claim
  - 判断claim
  - 释放claim

```
void UartInterrupt()
{

    uint32_t claim;
    //While there is pending interrupts
    while(claim = plic_claim(BSP_PLIC, BSP_PLIC_CPU_0)){
        switch(claim){
        case SYSTEM_PLIC_SYSTEM_UART_0_IO_INTERRUPT: UartInterrupt_Sub(); break;
        default: crash(); break;
        }
        //unmask the claimed interrupt
        plic_release(BSP_PLIC, BSP_PLIC_CPU_0, claim);
    }
}
```



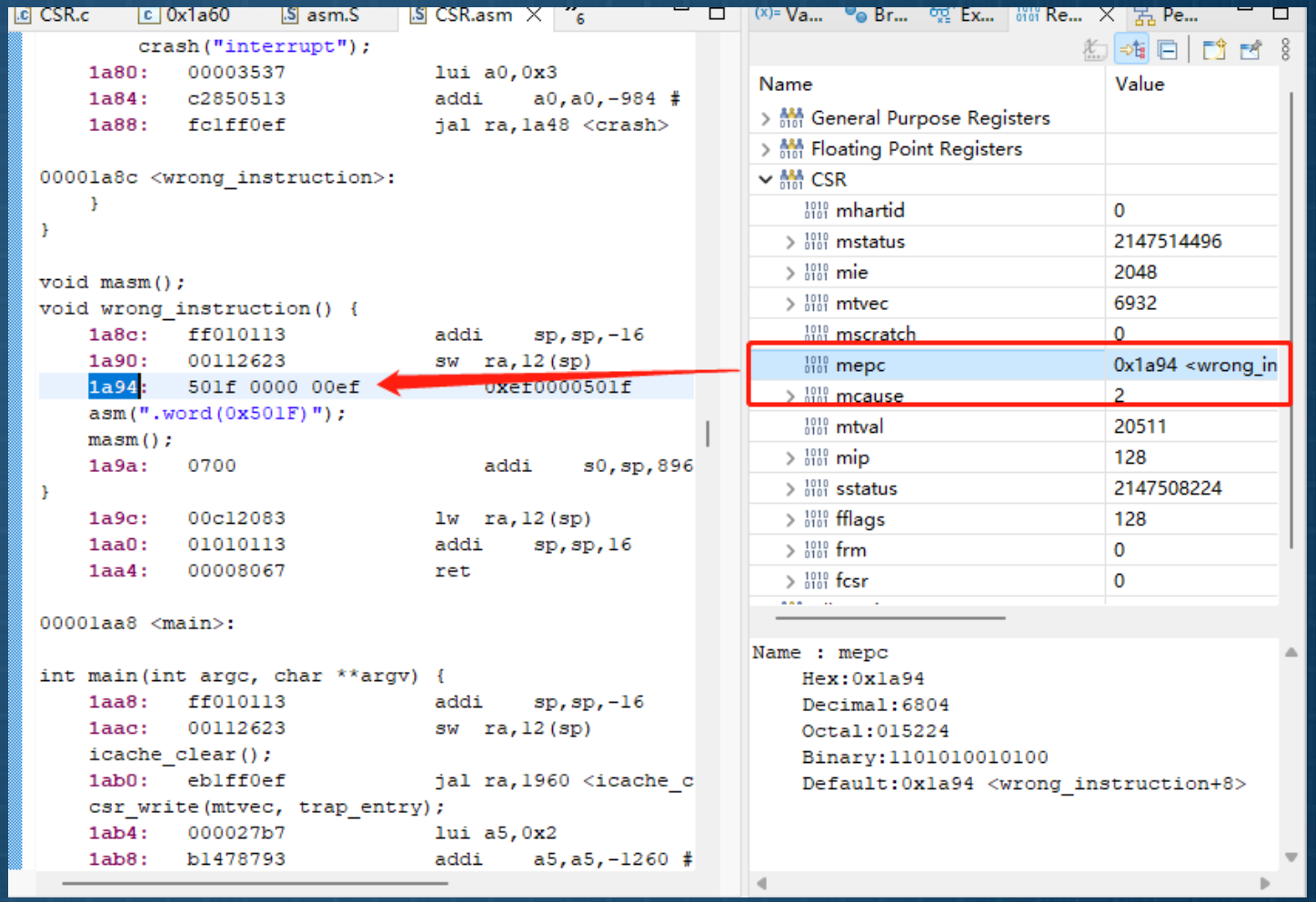
mstatus寄存器 – 异常

- 异常分类
  - 指令相关
    - 指令未对齐
    - 指令异常
    - 指令权限
  - 内存相关
    - 地址未对齐
    - 越界/权限不足
    - 内存页面相关（MMU+虚拟内存）

0	0	Instruction address misaligned
0	1	Instruction access fault
0	2	Illegal instruction
0	3	Breakpoint
0	4	Load address misaligned
0	5	Load access fault
0	6	Store/AMO address misaligned
0	7	Store/AMO access fault
0	8	Environment call from U-mode
0	9	Environment call from S-mode
0	10	*Reserved*
0	11	Environment call from M-mode
0	12	Instruction page fault
0	13	Load page fault
0	14	*Reserved*
0	15	Store/AMO page fault
0	16–23	*Reserved*
0	24–31	*Designated for custom use*
0	32–47	*Reserved*
0	48–63	*Designated for custom use*
0	≥64	*Reserved*

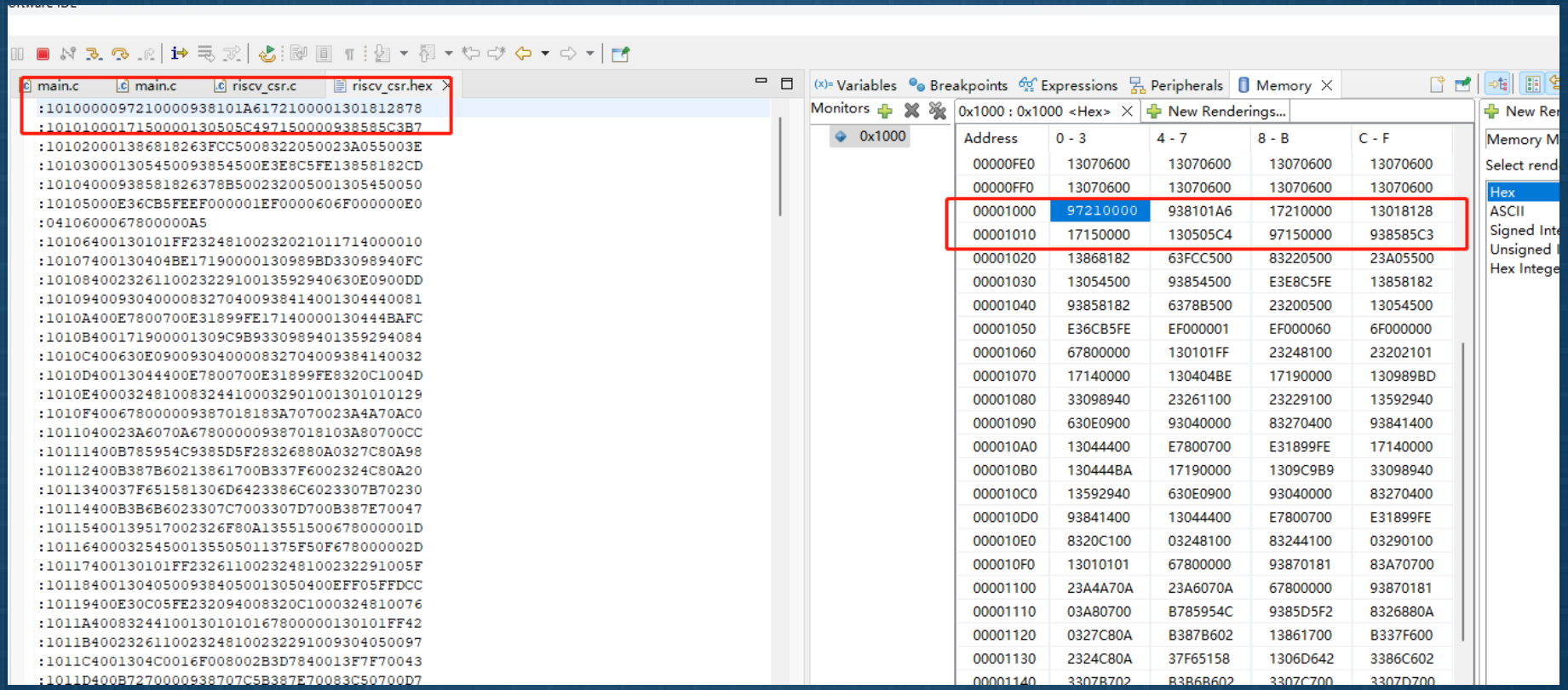
示例

- mcause 值为2，显示异常
- mepc 异常指令地址
- mtval 错误地址/指令



内存查看器

- Debug Mode -> Window -> Show View -> Memory -> Add Monitors
  - 查看、监控内存数据
  - 修改内存数据



缓存清理

- 用于清理指令和数据缓存
  - 数据缓存清理
    - 保证数据从cache写回内存
    - 腾出dcache空间，方便程序加速
    - 实现：asm(“.word(0x500f)”)
  - 指令缓存清理
    - 保证清空缓存指令，排除分支预测干扰
    - 实现：asm("fence.i");

```
void icache_clear() {
    asm("fence.i");
}

void dcache_clear() {
    asm(".word(0x500F)");
}

void cache_test() {
    typedef unsigned long ul;
    icache_clear();
    dcache_clear();
    ul st = clint_getTime(BSP_CLINT);
    for(int i = 0; i < 128; i++) {
        for(int j = 0; j < 64; j++)
            for(int k = 0; k < 32; k++)
                for(int m = 0; m < 4; m++)
                    a[i*64*32+m] = b[i*64*32+m] * 3.0;
    }
    ul ed = clint_getTime(BSP_CLINT) - st;
    bsp_printf("%s use: %ldms\n\r", __func__, ed/(SYSTEM_CLINT_HZ/1000));
}
```

易灵思

谢谢