**实验八报告**

学号 2017K8009929032 2017K8009929034

姓名 杨程远 杨宇恒

箱子号 15

一、实验任务

增加对eret、mtc0、mfc0、syscall的支持。通过func\_lab8仿真和上板验证。

二、实验设计

（一）总体设计思路

1、增加mfc0、和mtc0指令，我们把mtc0通过alu的通路，这就要增加一个src1\_is\_0的控制信号。

2、设计CP0\_reg模块，处理软件指令cp0的读写，和硬件例外及返回对cp0的自动操作；产生根据ip和屏蔽位中断信号发送给译码级。

3、例外和例外返回产生清除流水线的控制信号flush，以及控制IF级nextpc的多选器信号exc\_eret\_bus。

4、在br\_bus中增加is\_bd标志位，id级有效时把当前周期if级的指令做上标记if\_bd。

5、禁止mfc0从执行级和访存级的前递。

6、当es、ms、ws为例外或eret时，禁止hi、lo寄存器的更新以及sram的写。

（二）重要模块1设计：CP0\_reg模块

1. 工作原理

将CP0寄存器堆、CP0操作译码、是否有待处理中断判断逻辑封装到一起。没有分成多个模块是因为需要CP0寄存器堆里的信息，来判断硬件对CP0寄存器堆的操作，放到一个模块里可以避免一个信号在两个模块间穿梭。

1. 接口定义

| **­**  **类别** | | **名称** | **方向** | **位宽** | **功能描述** |
| --- | --- | --- | --- | --- | --- |
| 软件的指令操作 | | cp0\_addr | IN | 8 | 软件读写地址 |
| cp0\_rdata | OUT | 32 | 软件读数据 |
| cp0\_wen | IN | 1 | 软件写使能 |
| cp0\_wdata | IN | 32 | 软件写数据 |
| 硬件例外或ERET自动完成的操作 | 对所有例外有效 | exc\_type | IN | 8 | 每一位表示一种例外发生 |
| PC | IN | 32 | 发生例外指令的地址 |
| is\_slot | IN | 1 | 例外发生在延迟槽指令 |
| 对特定例外有效 | int\_num | IN | 5 | 中断号 |
| bad\_vaddr | IN | 32 | 内存访问错误地址 |
| EPC | OUT | 32 | EPC输出 |
| int\_happen | OUT | 1 | 有需要处理的中断 |
| eret | IN | 1 | eret指令 |

1. 重要内部中间结果：硬件CP0操作的译码结果

| **名称** | **位宽** | **功能描述** |
| --- | --- | --- |
| cp0\_XX\_XX\_set | 1 | 对应寄存器硬件自动置位信号 |
| cp0\_XX\_XX\_clear | 1 | 对应寄存器硬件自动清零信号 |
| cp0\_XX\_XX\_wen | 1 | 对应寄存器硬件自动写有效 |
| exccode | 5 | 例外编码，由例外类型译出 |

1. 功能描述

按照讲义实现寄存器更新，每一位写的优先级为rst，set，clear，wen，cp0\_wen，Cause\_ti寄存器除外，其clear优先级高于set。在本模块中完成中断IP被EXL、IE、IM屏蔽后是否有待处理的中断的逻辑判断，生成是否有待处理中断的int\_happen信号。

（三）重要设计实现1：例外和例外返回清除流水线

1. 工作原理

保证精确异常。

1. 功能描述

在wb级有效并且有wb级清除流水线控制信号时，向所有流水级发送flush信号。

把除取指以外的Xs\_valid寄存器在下一个周期清为0。把取指的fs\_valid在下一个周期置为1，并且下一个周期的fs\_pc更新为这个周期的next\_pc，同时将iram的读使能有效。

三、实验过程

（一）实验流水账

10月28日17:00-10月29日1:30：完成全部实验设计和代码。

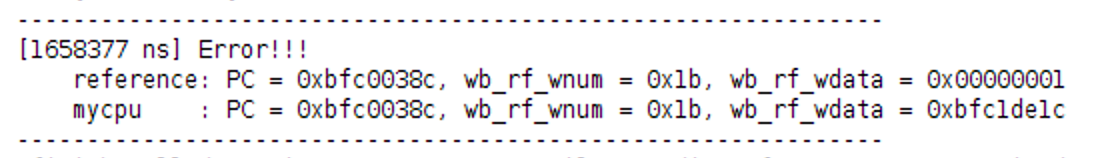
10月29日13:00-10月29日17:00：通过测试。

（二）错误记录

1、错误1：例外发生后load指令出错

（1）错误现象

ld的写回值不对。



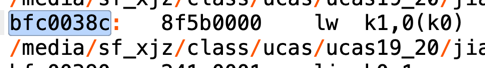


图1 ld指令出错

（2）分析定位过程

很容易想到，ld出错是因为之前store的问题，这时想到在exe级阻止hi、lo更新的时候没有同时阻止dsram的写使能。查看代码，发现果然如此。

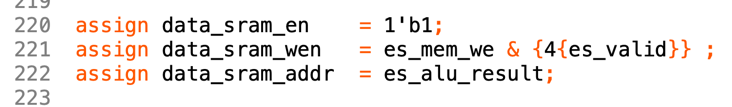


图2 st使能未做修改

（3）错误原因

发生例外时store没有精确例外

（4）修正效果

在后面流水级例外的时候无效掉sram的写，之后仿真通过这一部分。

2、错误2：除法器没有被flush

（1）错误现象

例外返回后的除法指令无法得到结果。



图3 除法器busy不接受新的除法运算

（2）分析定位过程

检查上一次除法器的使用，发现在使用除法器的时候发生了系统调用，这时，由于除法有效信号消失，除法器输出有效也随之消失，这样除法器没有收到输出有效而认为一直处在忙碌状态。

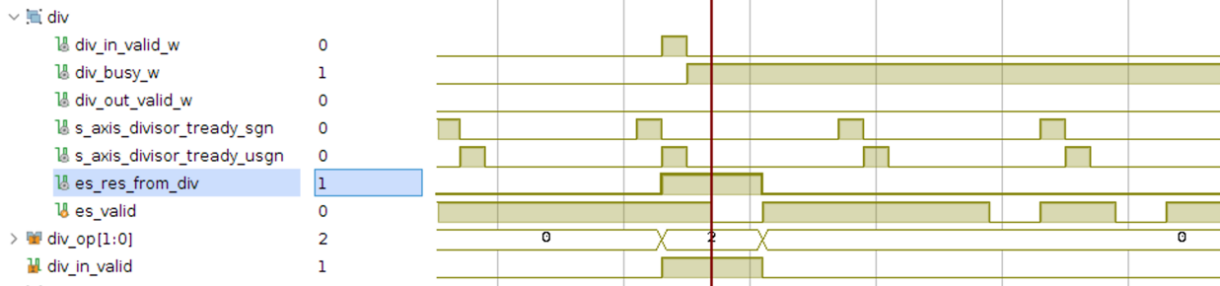


图4 上一次除法操作没有去除除法器busy状态

（3）错误原因

除法器没有被例外flush。

（4）修正效果

对除法器模块输入flush信号，当flush为1时，保存此信号以进入特殊状态，这个状态可以在busy状态（除法模块不可输入状态）抛弃除法IP核产生的下一个输出，之后清除busy位，重新让除法模块处于可以接受输入的状态。