

实验 4 报告

学号：2016K8009929011
姓名：段江飞

一、实验任务（10%）

本实验要求添加例外中的系统调用支持，其中需要添加 cp0 寄存器中的 status、cause 和 epc 寄存器，添加指令 mfc0、mtc0 和 eret，这些指令和寄存器保证系统调用能够正确执行。

二、实验设计（30%）

（一）软件实现

例外处理由 start.S 中实现。

（二）硬件实现

1、CP0 寄存器

根据讲解，CP0 寄存器并不完全需要 32 位的 reg 实现。Status 寄存器的 im，exl 和 ie 位由 reg 实现，其他位是只读位；Cause 寄存器的 bd，ti，ip 和 exccode 由寄存器实现，epc 是 32 位寄存器。

2、MTC0 和 MFC0

对每个 CP0 寄存器，有一个写使能信号，当写使能信号为高时，会写 CP0 寄存器，写数据就是之前的 rdata2，mtc0 单独建立了数据通路，将写使能信号传到写回级，写数据利用之前的数据通路；所有的 CP0 寄存器共用一个读数据，在译码阶段，根据指令得到读数据，同时，mfc0 的数据通路复用之前的数据通路，在译码阶段，根据多路选择器，选择 alu_sec，将 ALU 的两个操作数分别选为 CP0_rdata 和 0，利用加法操作得到结果。

通过复用数据通路，相关的处理也迎刃而解。

3、刷新流水线

通过将流水线的 valid 信号置为 0 实现刷新流水线。

4、ERET

Eret 在译码阶段得知，然后刷新流水线，将 STATUS 的 EXL 位置为 0，将 EPC 写入 PC，这时候会利用前级处理 EPC 的数据相关。

5、精确例外

在当前指令为分支跳转指令时，下一条指令就是在分支延迟槽中，会利用寄存器记录起来，在例外提交时，根据该信号确定写入 EPC 的 PC 值，并置 cause 的 bd 位，实现精确例外。

例外提交统一在访存阶段提交，集中处理例外。

（三）验证

1、仿真验证

在 vivado 上仿真，测试 69 个功能点，通过测试。

2、上板验证

数码管从 0 加到 69，高低 8 位同步累加。

三、实验过程（60%）

（一）实验流水账

1、11 月 16 日下午 2 点到 17 日凌晨 1 点 40

数据通路，debug。

2、10 月 17 日上午 10 点到下午 4 点

找出组合环，完成试验，写实验报告。

（二）错误记录

1、错误 1

（1）错误现象

刚写完数据通路，进行仿真的时候，发现波形阻塞在开始的几条指令，无法继续执行。

（2）分析定位过程

通过看波形图发现是阻塞的原因，之前的阻塞都通过了测试，现在又出现了问题，我感觉很奇怪，我先把阻塞的信号抓取出来，发现有的信号是不定值，导致选择出错，于是我先注释掉流水线中的阻塞信号，继续仿真，发现和 PC 选择相关的信号出现了高阻，然后我去仔细检查代码，发现了问题。

（3）错误原因

打错字母了，将 `eret_cmt` 连线是敲成了 `eret_cmp`。

（4）修正效果

添加阻塞，程序能正常通过之前的测试了。

2、错误 2

（1）错误现象

```
[ 3177 ns] Error!!!  
reference: PC = 0xbfc003f4, wb_rf_wnum = 0x1a, wb_rf_wdata = 0xbfc0130c  
mycpu      : PC = 0xbfc003f4, wb_rf_wnum = 0x1a, wb_rf_wdata = 0x00000000
```

图一

（2）分析定位过程

对应的 PC 是 `mfc0 k0`, `c0_epc`，显然，之前写入 `cp0_epc` 的 PC 出错。抓取发生系统调用处的 `cp0` 信号，发现没有写 `epc` 寄存器，查阅文档发现，是判断条件错误。

（3）错误原因

在 EXL 为 1 的时候不写 EPC 寄存器，我的判断条件恰好反了。

(4) 修正效果

能够正确读取和写入 epc 寄存器。

3、错误 3

(1) 错误现象

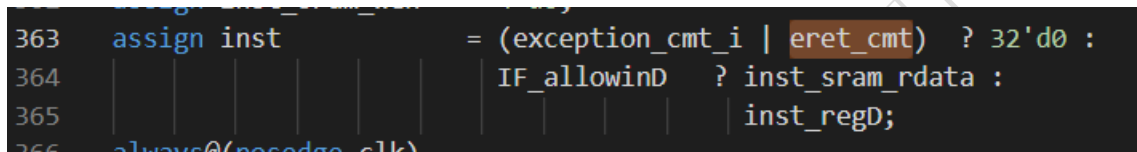
波形停止。

(2) 分析定位过程

我也不知道怎么分析，找出来组合环的电路，看不到到底是哪里出现了组合环，仔细分析自己加的东西，波形最后停在了 eret 指令，分析 eret 的逻辑，发现了组合环。

(3) 错误原因

和国庆写出来组合环的原因一样，eret_cmt 由 inst 译码产生，又用来控制 inst 的选择。



```
363 assign inst = (exception_cmt_i | eret_cmt) ? 32'd0 :  
364 IF_allowinD ? inst_sram_rdata :  
365 inst_regD;  
366 always@(posedge clk)
```

图二

(4) 修正效果

添加阻塞，程序能正常通过之前的测试了。

4、错误 4

(1) 错误现象

比对错误，PC 比对出错。

(2) 分析定位过程

通过反汇编查找比对出错的指令，发现是 syscall 指令和 div 指令，系统调用指令之后的 div 指令执行了，不满足例外处理要求。

(3) 错误原因

通过对 div 指令相关的信号分析，发现是我对刷新流水线的处理是强行将一些信号置为 0，这次是没有置完全，有些信号处理出错，发现流水线的控制信号 valid 控制当前流水级是否有效，将该信号置为 0 可以实现刷新流水线的功能。

(4) 修正效果

通过了系统调用的测试。

四、实验总结

又写出了组合环。