实验 4-2 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 Break，ADEL，ADER，Overflow 支持，还要添加时钟中断例外，6 个硬件例外和 2 个软件例外。其中，时钟中断例外对应 cause 寄存器的 IP7,6 个硬件中断对应 cause 寄存器的 IP7-2,2 个软件中断例外对应 cause 寄存器的 IP1-0。为了例外支持，还需要添加 COUNT，COMPARE，BADVADDR 寄存器。

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

Count 和 Compare 寄存器都是 32 位 Reg，不进行初始化，count 寄存器每两个时钟周期会加 1。BADVADDR 寄存器存储在 ADEL 或 ADER 例外发生时的地址，在取指的 ADEL 发生时，BADVADDR 寄存器中存的是当时的访存级 PC，也就是发生例外的 PC，在 load 或 store 的地址错例外发生时，BADVADDR 存的是错的访存地址。

2、Break 例外

Break 例外对应的是一条指令，和 syscall 例外的处理一致，在译码时译码出 break 信号，然后带到访存级，在访存时提交例外，将 Cause 的 EXCcode 置为相应的例外码并将 EXL 置为 1，将精确例外地址写入 EPC。

3、Overflow 例外

Overflow 的信号在组成原理课上就已经实现了，有用现在 ALU 加法被一些其他指令复用了，导致可能加的时候出现溢出或者未知值，但是不应该报例外，于是在译码时新加一个信号表示此是 Overflow 的例外有效。Overflow 的例外就是正数加出负数或者负数加出正数会出现，减法转化为加法处理，例外提交时会写 EPC，CAAUSE.EXCCode 和置 EXL 位。

4、ADER 和 ADEL

地址错例外根据对 PC 和访存地址的后两位是否是对齐的来确定，PC 的低两位一定为 0，否则会报 ADEL 例外，而访存地址则根据指令，如果是非对齐指令，访存地址可以是非对齐的，但是 lh,sh 指令的最低位只能是 0，lw 和 sw 指令的低两位是 0，否则 load 报 ADEL 例外，store 报 ADER 例外。

当然，这几个例外是有优先级的，先判断取指的 ADEL 例外，再去判断其他例外，包括系统调用、break 等。

5、时钟中断

时钟中断在 COUNT 等于 COMPARE 且 STATUS 的 IM7 为 1 时发生，时钟中断例外发生时，也是挂到访存级的指令上，在访存级报例外，时钟中断发生时，将 CAUSE 的 IP7 置为 1，EXCCode 置为 0，TI 置为 1，STATUS 的 EXL 置为 1。时钟中断处理时，在写 COMPARE 寄存器时，会将 TI 和 IP7 清 0。

6、软中断

软中断由 CAUSE 的 IP1-0 控制，软中断也是挂在访存级指令上，在访存级提交，在 IP1-0 为 1，且相应的 IM1-0 为 1 时，触发软中断。

7、硬件中断

硬件中断和软中断一样，由 CAUSE 的 IP5-0 控制，软中断也是挂在访存级指令上，在访存级提交，在 IP7-2 为 1，且相应的 IM7-2 为 1 时触发。

（三）验证

1、仿真验证

在 vivado 上仿真，测试 94 个功能点，通过测试。

2、上板验证

数码管从 0 加到 94，高低 8 位同步累加。

（四）硬件中断测试

1、电子表

利用我的 CPU 去测试电子表，由于先仿真去看，而且又生成了比对文件，发现和龙芯的对 CP0 寄存器初始化有所不同，对寄存器做了一些修改之后，综合然后生成 bit 文件，测试发现功能正常。

2、记忆游戏

在顶层将 4 个硬件中断信号和 12-15 号按键连接，定制指令和数据 ram 之后生成 bit 文件，进行上板测试，发现功能能正常运行，灯闪了 8 次，按对之后，得分为 8。

三、实验过程（60%）

（一）实验流水账

1、11 月 22 日晚上 8 点到 23 日凌晨 1 点

数据通路，debug。

2、11 月 23 日晚上 9 点到 24 日凌晨 1 点

Debug，通过了中断前的测试。

3、11 月 24 日晚上 11 点到 25 日凌晨 2 点

Debug，通过了 lab4-2 测试。

4、11 月 25 日下午 1 点到 5 点

测试，上板，写实验报告。

（二）错误记录

1、错误 1

（1）错误现象

```
——[  20515 ns] Number 8'd02 Functional Test Point PASS!!!
--------------------------------------------------------------

[  21297 ns] Error!!!
    reference: PC = 0xbfc00380, wb_rf_wnum = 0x1a, wb_rf_wdata = 0x0001001a
    mycpu    : PC = 0xbfc0492c, wb_rf_wnum = 0x02, wb_rf_wdata = 0x64c76d7c
--------------------------------------------------------------
```

图一

（2）分析定位过程

对应的 PC 是例外入口，根据出错的 PC 找到对应指令为 add，是 Overflow 的测试，这条指令就是出错指令。

（3）错误原因

例外提交时流水线没有完全处理干净，溢出的这条指令还会写回，对写回级 RegWrite 根据例外提交进行置 0。

（4）修正效果

通过了溢出的测试。

2、错误 2

（1）错误现象

波形一直阻塞在例外入口地址 0xbfc00380。

（2）分析定位过程

根据进入例外的指令，发现是取指令的地址错误，去看相关的流水信号，发现例外提交一直为高。

（3）错误原因

第一条取指令地址错误，之后的指令每次加 4，导致一直会把例外信号置为高，然后由于例外信号没有在上一级无效是清 0，导致例外一直提交，是的 CPU 卡死。将 exceptionM 修改为在上一级无效是清 0。

（4）修正效果

去除了阻塞。

3、错误 3

（1）错误现象

写入 EPC 的数值出错。

（2）分析定位过程

根据错误 PC 发现是从 EPC 中取数对比，错误写入的值恰好差了 4，于是把判断指令是否在分支延迟槽的信号拉出来，发现错误。

（3）错误原因

判断指令是否在分支延迟槽的信号慢了一拍。

（4）修正效果

EPC 中值正确。

4、错误 4

（1）错误现象

该进入例外处理但是没有进入。

（2）分析定位过程

根据错误 PC 发现是 divu 和 eret 指令，我设计的 eret 在 ID 阶段提交，但是它前面是一个 divu 指令，会阻塞，这样在 eret 提交清流水线和 divu 之间出现了矛盾，eret 一直没法提交，但是提交信号高低交错，导致最后 divu 执行完之后，eret 的提交信号丢失。

（3）错误原因

Eret 提交的位置不太合适，将 eret 提交位置修改为例外提交一只，在访存时提交。

（4）修正效果

到此，通过了已经添加的几个例外的测试。时钟中断还未添加。

5、错误 5

（1）错误现象

```
[2700057 ns] Error!!!
    reference: PC = 0xbfc26350, wb_rf_wnum = 0x02, wb_rf_wdata = 0x0000000b
    mycpu    : PC = 0xbfc00380, wb_rf_wnum = 0x1a, wb_rf_wdata = 0x00004000
```

图二

（2）分析定位过程

根据错误 PC 发现是 lb 的测试，不该进入例外却进入了例外。

（3）错误原因

ADEL 例外处理的时候没有考虑非对齐指令，还有 ADER 例外处理，对这两个访存地址错误的例外处理进行了修改，加上对非对齐指令的判断。

（4）修正效果

通过了测试。