实验 4 报告

学号: 2017K8009929013
姓名:
箱子号: 72

一、实验任务（10%）

理解单发射五级流水 cpu 中可能出现的冲突，并用阻塞方式解决数据相关中的寄存器写后读冲突

二、实验设计（40%）

（一）总体设计思路

总体：ID 与 EXE、MEM、WB 阶段满足“一定关系”时，实现“阻塞”

1. “一定关系”：

   * 处于译码级的指令：具有来自于非 0 号寄存器的源操作数

   * 处于执行级、访存级或写回级的指令：寄存器写回信号拉高（目的操作数的寄存器号一定非 0

   * 上述两寄存器号相同

2. “实现阻塞”：拉低 ID 段的 readygo 信号

（二）重要模块设计：阻塞实现模块

1、 代码

```
assign ewd={es_gr_we,es_dest};
assign mwd={ms_gr_we,ms_dest};
assign wwd={ws_gr_we,ws_dest};
```

```
wire block;
assign block=
(
((|rf_raddr1))&  //addr1!=0
((ewd[5]&(ewd[4:0]==rf_raddr1))|  //one of them ==addr1
(mwd[5]&(mwd[4:0]==rf_raddr1))|
(wwd[5]&(wwd[4:0]==rf_raddr1)))
)
|
(
((|rf_raddr2))&  //addr2!=0,use data2
((ewd[5]&(ewd[4:0]==rf_raddr2))|   //one of them ==addr2
(mwd[5]&(mwd[4:0]==rf_raddr2))|
(wwd[5]&(wwd[4:0]==rf_raddr2)))
);

assign ds_ready_go    = (block==1'b1)?0:((block==1'b0)?1:1);
```

2、 **功能描述**

ewd、mwd、wwd 分别为从 exe，mem，wb 传递过来的写使能信号 wen+写地址 waddr

block 信号为阻塞判断信号，在满足条件时（id 段读地址不为 0&后 3 段写使能拉高&两地址冲突）拉高

ds 阶段的 readygo 信号在阻塞信号有效且拉高时置零

三、实验过程（50%）

（一）实验流水账

周一上午：打开任务书，记笔记，分析任务内容和要实现的逻辑

周一下午：配置环境，改 inst_ram，生成 trace，试着看未实现阻塞时 cpu 仿真的情况，发现出现数据无效问题

周一晚上：阻塞块设计初步完成，开始实现代码并调试

周二上午：继续调试，至仿真通过，综合实现，等待上板

（二）debug 记录

1. block 初始几个周期无效

一开始直接把 ~ block 赋给 ds_readygo，block 在初始几个周期因为没有 ewd 那些信号所以是 x，导致 readygo 一直没拉高，整个流水从开始就被阻塞

修改为：assign ds_ready_go　　= (block==1'b1)?0:((block==1'b0)?1:1);

2. 持续阻塞

因为各阶段执行完毕后，到阻塞结束前，这几个阶段的值都不会再改变，所以如果不能及时解除，就会一直阻塞下去

修改：在每个阶段各自的 valid 无效时（执行完毕，开始被阻塞），拉低写使能信号，破除阻塞

3. 数码管写数据报错

一直沿着信号找，最后发现这个信号还是来自 cou 中的寄存器读得的数据，但是错误处是一个 sw 指令，这个寄存器数据并没有用于这条指令，只是单单输出。因为我一开始的设计在判断阻塞条件时加上了“寄存器值需要被 alu 使用”，所以这里并没有考虑写后读，没阻塞，就导致数据错误

修改：取消了“寄存器值需要被 alu 使用“这一阻塞条件