实验 3 报告

学号: 2017K8009929013
姓名:
箱子号: 72

一、实验任务（10%）

通过分析代码，理解五级流水 CPU 架构和的工作过程；

调试代码中的 bug，掌握一定的调试技能

二、实验设计（40%）

（一）总体设计思路

指令处理划分为 5 个阶段：取指，译码，执行，访存，写回

5 个阶段分成了 5 个模块，在顶层文件中调用

5 个模块并行，通过 valid 和 allowin 信号握手交互，通过 bus 传递信息（控制信号和数据）：分析上一阶段的 bus 获取信息，完成本阶段的处理后，将结果放到新的 bus 中传给下一级

（二）重要模块 1 设计：IF 模块

1、 重要接口介绍

br_bus：从上一指令的 ID 段传过来的跳转信息

2、 功能描述

主要介绍 PC_next 的生成过程：从 PC+4 和 branch_target 中二选一，选择条件为 b_taken 信号（taken 信号和 target 都由 br_bus 中传递过来）

（三）重要模块 2 设计：ID 模块

1、功能描述

ID 段完成指令的解析，生成控制信号和数据信息，同时判断跳转与否，生成 br_bus 传递给 IF 阶段，并利用 wb 阶段的 regfile 信息完成对 regfile 的写入更新

指令解析：分析指令类型，生成对应的控制信号（运算/访存操作数选择/）和操作数（立即数/寄存器读取）

br_bus 生成：br_bus = br_taken + br_target, 根据指令类型判断是否跳转以及生成怎样的跳转地址

regfile 访问与更新：写接口从 wb_bus 中解析的，读接口从指令中解析得

（四）重要模块 3 设计：ALU 模块

1、接口介绍

op 指定操作类型，src 指定操作数

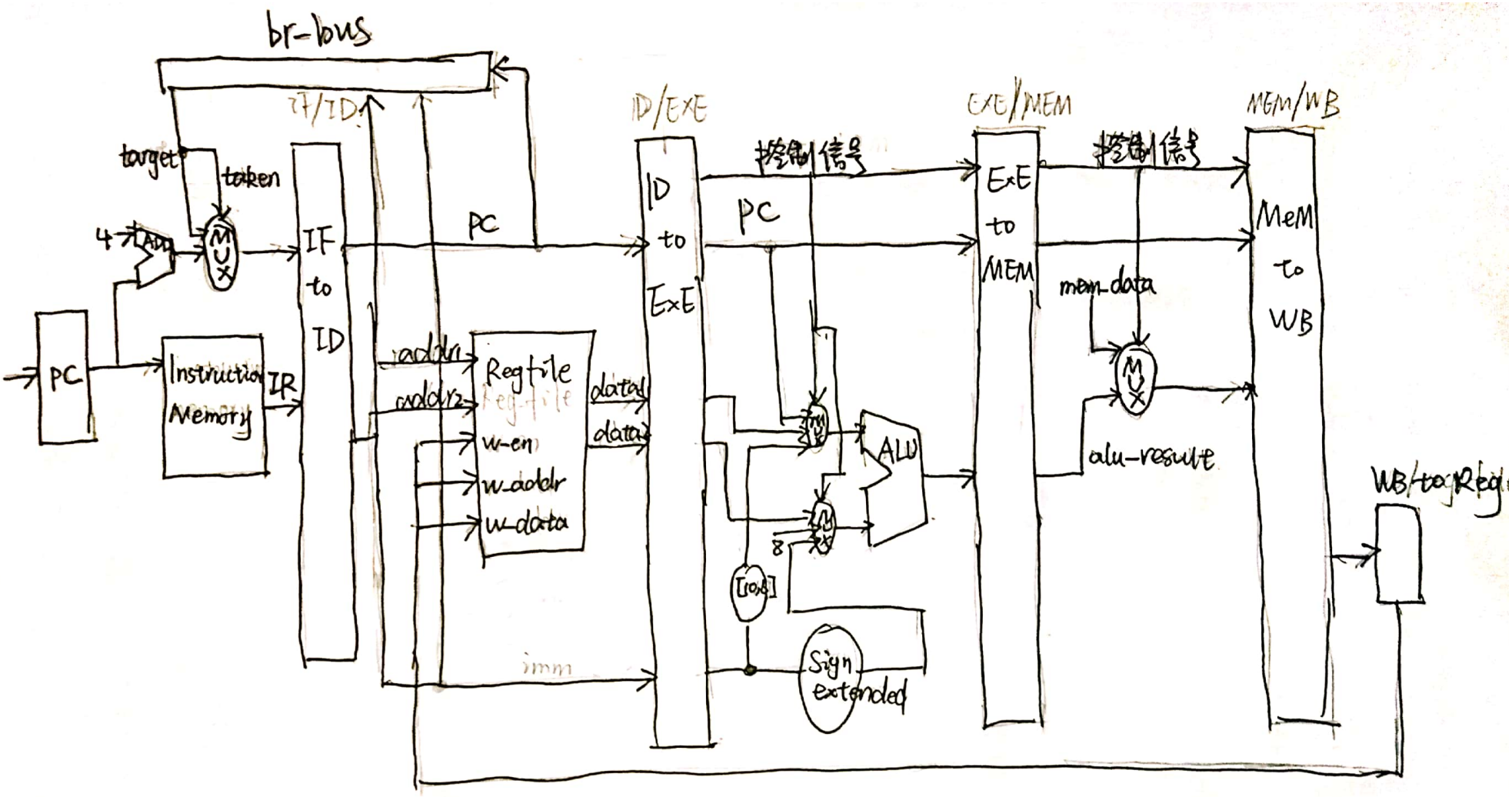
2、功能描述

根据 op 对两个操作数进行运算，输出结果。

内部计算过程：每一种运算结果用一个变量表示，都用组合逻辑处理 src 后得到，最后根据 op 从众多结果变量中选取一个赋给最终 result

（五）子任务一：myCPU 整体架构

见下页手绘图


br_bus
IF/ID
ID/EXE
EXE/MEM
MEM/WB
target
taken
4
Add
MUX
PC
IF to ID
PC
Instruction Memory
IR
raddr1
raddr2
Regfile
w_en
w_addr
w_data
data1
data2
ID to EXE
PC
imm
[10,8]
8
ALU
Sign extended
EXE to MEM
mem_data
alu_result
MEM to WB

三、实验过程（50%）

（一）实验流水账

9 月 14 号上午，完成 func 编译、基准 trace 生成和 CPU 仿真

9 月 15 号下午，对着讲义分析 CPU 设计和 trace 比对的流程

9 月 16 号，分析 CPU 源码，仿真，找 bug

9 月 17 号，截图，写报告

（二）错误记录：7 个 bug

1. ID 模块缺少 ds_valid 的赋值

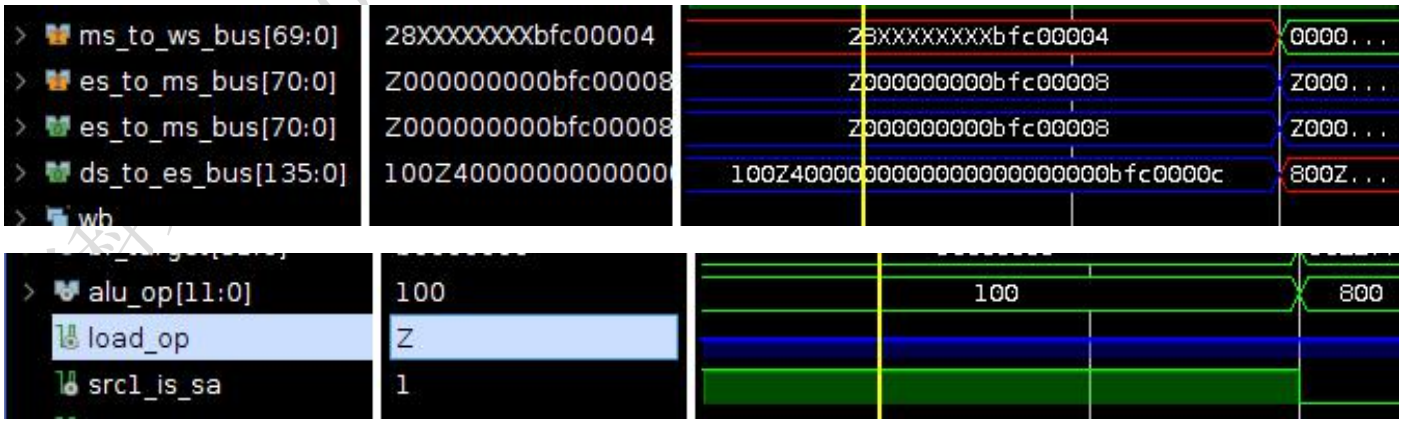
发现过程：

在分析 CPU 代码画框图时，对模块间的几个信号（allow_in, ready_go, valid）的功能和逻辑不是太清楚，就逐行的看它们的代码理逻辑，然后发现其他模块的 valid 几乎都有相同的赋值逻辑（reset 时为 0，之后就是 if (allowin) b_ valid = a_to_b_valid;），只有 ID 模块 ds_valid 没有赋值，推测这里是一个错误

2. ID 模块中 load_op 未赋值

发现过程：

```
------------------------------------------------------------
Test begin!
------------------------------------------------------------
[    2067 ns] Error!!!
    reference: PC = 0xbfc00000, wb_rf_wnum = 0x08, wb_rf_wdata = 0xffffffff
    mycpu    : PC = 0xbfc00000, wb_rf_wnum = 0x08, wb_rf_wdata = 0xxxxxxxx
```



控制台报错说 wb 阶段写回 data 是一串 x，就顺着 bus 往前找，发现最早在 ID 的 bus 中出现了蓝色波形，检查后发现是 load_op 未赋值

3. EXE 模块调用 ALU 是 src1 加错

发现过程：

```
------------------------------------------------------------
[    2067 ns] Error!!!
     reference: PC = 0xbfc00000, wb_rf_wnum = 0x08, wb_rf_wdata = 0xffffffff
     mycpu    : PC = 0xbfc00000, wb_rf_wnum = 0x08, wb_rf_wdata = 0xfffffffe
------------------------------------------------------------
```

控制台报错说 wb 阶段写回 data 出错，就顺着 bus 往前找，是 addiu 指令，发现 alu_result 有误，而 alu 计算并没有出错，接着往前找，发现 exe 阶段的 src 选择和 aluop 都没有出错，但是 src 就是不一样，查看调用代码，发现赋值出错

4. br_bus 位数出错

发现过程：

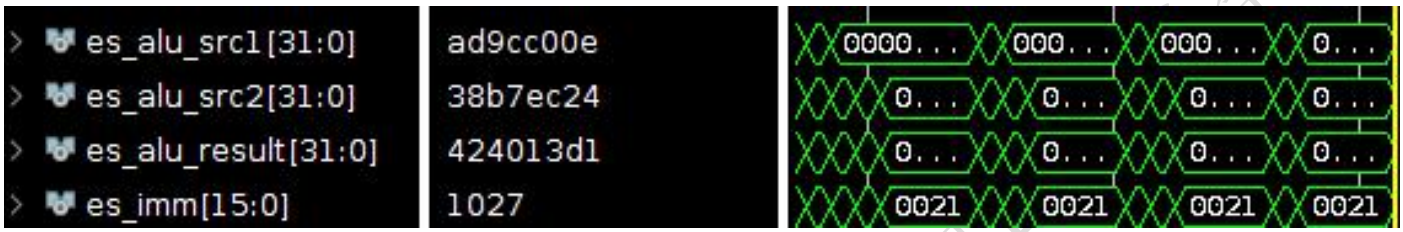
```
------------------------------------------------------------
[    2107 ns] Error!!!
     reference: PC = 0xbfc0038c, wb_rf_wnum = 0x04, wb_rf_wdata = 0xbfb00000
     mycpu    : PC = 0xbfc00010, wb_rf_wnum = 0x08, wb_rf_wdata = 0x80000000
------------------------------------------------------------
```

控制台报错说 pc 地址不对，意识到这里应该应该是个跳转，而 myCPU 还是继续累加，拉出 IF 段波形，检查 nextpc 波形，发现果然如此，就去检查 br_bus 的标志位和地址，发现 br_bus 只定义了 32 位，然后找到头文件进行了修改

5. ALU 中组合环

发现过程：

```
[ 702000 ns] Test is running, debug_wb_pc = 0xbfc90324
[ 712000 ns] Test is running, debug_wb_pc = 0xbfc912c4
----[ 716985 ns] Number 8'd09 Functional Test Point PASS!!!
```



控制台停止打印，但是仿真还在继续，检查波形图发现指令和地址都不再更新，只有 aluresult 一直在两个数间重复，意识到这里可能又一个环，产看指令类型，为 sll，检查 alu 中这个分支，果然有组合环

6. ALU 中 srl 指令赋值时位数不对

发现过程：

控制台报错说 wb 阶段写回 data 出错，就顺着 bus 往前找，alu_result 有误，找 2 个 clk 之前的 if 段，是 srl 指令，检查 alu 中 srl 分支的赋值，发现少了一位

7. MEM 模块中 bus 赋值时使用阻塞赋值

发现过程：

其实前 6 个改完之后，仿真是可以通过的，但是因为老师提醒了有 7 个 bug，就跟同学交流了一下……然后发现了这个阻塞赋值