实验 2-2 报告

学号：2016K8009908007
姓名：薛峰

一、实验任务

实验目的：

（一）、在第一阶段的基础上新增如下 19 条机器指令：ADD、ADDI、SUB、SLTI、SLTIU、ANDI、ORI、XORI、SLLV、SRAV、SRLV、DIV、DIVU、MULT、MULTU、MFHI、MFLO、MTHI、MTLO；

（二）、考虑数据相关，并采用前递；

（三）、实现乘法器和除法器，其中乘法采用 booth 算法+华莱士、除法采用迭代算；

检验方法：

要求仿真和上板运行 lab2_func_2 通过。

二、实验设计

（一）总体设计思路

该项目总体分为以下六个模块：mycpu_top，控制模块（cpu_control），寄存器堆(reg_file)，ALU，mul（乘法模块）和 div（除法模块）。

其中 mycpu_top 模块为顶层模块，控制其他模块进行工作。本次实验新增乘法器和除法器，另外 mycpu_top 稍有修改以实现前递，控制模块增添指令，其他模块未作修改。

（二）mul 模块设计

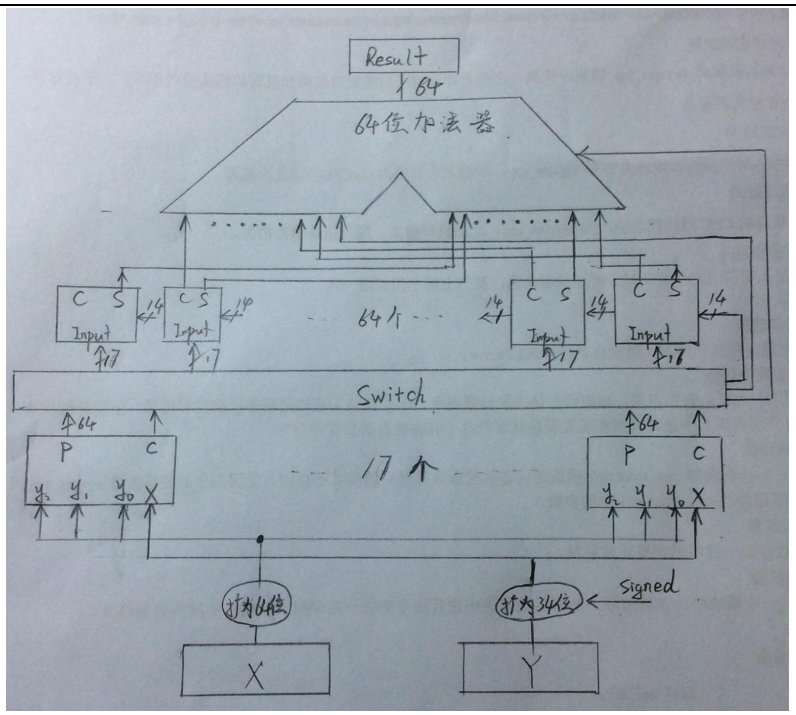
1、工作原理

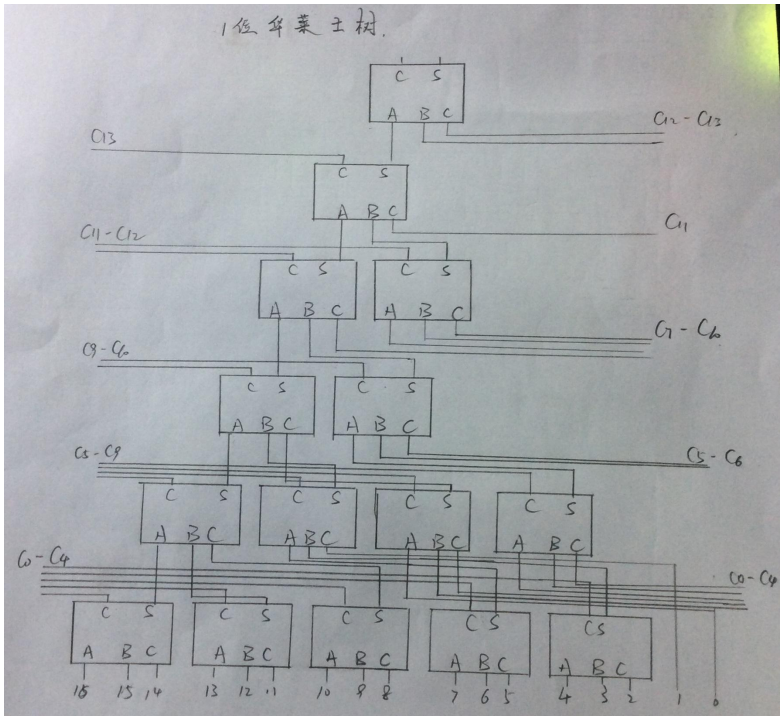
采用 booth 乘法和华莱士树的方式实现，主要分为四部分。

首先，第一部分根据两个乘数得出 17 个部分积。第二部分将 17 个 64bit 的部分积转化程 64 个 17bit 的数。第三部分是华莱士树部分，根据 64 个输入产生两个 64bit 的数。第四部分将两个 64bit 的数和两个最低位的进位信号相加。

其中在华莱士树和 64bit 全加器之间增加一个寄存器，用于切分流水线。

2、结构设计图





3、功能描述

信号 mul_signed 可指定无符号乘法还是有符号乘法，该模块根据输入 x，y 产生乘积，其过程需要两拍。

（三）div 模块设计

1、工作原理

该模块采用恢复余数法实现。其过程分为三步：

1）计算除数和被除数的绝对值；

2）迭代运算求出商和余数的绝对值，每次判断除数和被除数对应位置的大小，如果够减，则商对应位置设为

1，被除数对应位置改为相减的结果，如果不够减则商对应位置设为 0；

3）调整商和余数的符号

2、功能描述

信号 div 拉高，表示需要做除法运算，启动除法器；根据符号判断信号 div_signed，除数和被除数得出结果，并将 complete 拉高，表示除法运算完成。

（四）前递功能设计

1、工作原理

本次实验只需考虑写后读的数据相关，即寄存器的值还未写进，就需要读寄存器。也就是说需要考虑前递写寄存器内部的值。这里只考虑读寄存器堆，因为读 HI 和 LO 本质上和读寄存器堆相同。

因为寄存器的值需要在 ID 阶段用到，所以有五种情况：

1）EX 寄存器中的指令为 load，此时写寄存器的值需要到 WB 阶段才能返回，因此需要等待两拍；

2）EX 寄存器内的指令需要写寄存器，且不是 load 指令，则需要将要写的值前递给 ID 寄存器内的指令；

3）MEM 寄存器中的指令为 load，此时写寄存器的值需要到 WB 阶段才能返回，因此需要等待一拍；

4）MEM 寄存器内的指令需要写寄存器，且不是 load 指令，则需要将要写的值前递给 ID 寄存器内的指令；

5）WB 寄存器内的指令需要写寄存器，则需要将要写的值前递给 ID 寄存器内的指令；

其中阻塞操作只需要让相应的 ready_go 信号赋值为 0 即可实现。

2、功能描述

通过前递技术，可以大大减少流水线的阻塞，同时保证了读取数据得正确性。

三、实验过程

（一）实验流水账

时间	记录
10 月 2 日 20：00~22：20	一、阅读课本和讲义乘法器部分， 并开始写乘法器代码。
10 月 3 日 10：00~11：50	一、对乘法器模块进行行为仿真，修改代码； 二、仿真通过。
10 月 3 日 14：30~16：20	一、阅读讲义除法器部分， 并开始写除法器代码； 二、对除法器模块进行行为仿真，修改代码，上板验证通过。
10 月 6 日 19：20~22：20	一、添加 ready_go, valid, allowin 等信号，添加新的指令； 二、添加前递功能。
10 月 7 日	一、进行行为仿真，修改代码。

10：20~12：10 13：40~18：30	
10 月 8 日 17：30~次日 3：30	一、进行行为仿真，修改代码，最终行为仿真通过； 二、上板验证通过。

（二）错误记录

1、错误 1

（1）错误现象

在进行乘法器模块行为仿真时，控制台在第一个测试数据报错。

（2）分析定位过程

逐步检查各部分数据的正确性，先检查 booth 模块，发现产生的 17 个部分积正确，随后检查 switch 部分。

（3）错误原因

发现在 switch 部分只考虑了前 16 个部分积，即产生了 64 个 16bit 的数。

（4）修正效果

在该部分加上第 17 个部分积，随后错误解决。

（5）归纳总结

该错误属于手误类型，今后写代码时需要更认真一些。

2、错误 2

（1）错误现象

在进行除法器模块行为仿真时，控制台在某测试数据报错。

（2）分析定位过程

发现余数错误，并且所得余数恰好是正确余数的相反数，所以查看给余数判断正负部分的代码。

（3）错误原因

发现该部分由于失误逻辑出现错误。

（4）修正效果

将该错误更改过后，行为仿真通过

3、错误 3

（1）错误现象

在对 cpu 进行行为仿真时，控制台在某处报错。

（2）分析定位过程

发现 PC 值正确，写数据错误，因此找该指令在 ID 阶段读寄存器的值，发现读出来的数据出错。并且查看 test.s 发现该指令处存在数据相关。

（3）错误原因

该指令在 ID 时，MEM 阶段和 WB 阶段的寄存器内存放的数据都是写某一寄存器（记为 S），该指令需要读寄存器 S，先然该指令需要读 MEM 寄存器内存放的写寄存器数据。而我之前的前递部分代码如下：

```
assign ReadData1_ID = (Esrc1_ID == rf_waddr_EX  && rf_wen_EX != 4'b0)?                          rf_wdata_temp_EX:
                      (Esrc1_ID == rf_waddr_WB  && rf_wen_WB != 4'b0)?                          rf_wdata:
                      (Esrc1_ID == rf_waddr_MEM && rf_wen_MEM != 4'b0 && MemRead_MEM != 1'b1)?  rf_wdata_temp_MEM:
                      (Esrc1_ID == rf_waddr_MEM && rf_wen_MEM != 4'b0 && MemRead_MEM == 1'b1)?  inst_sram_rdata:
                                                                                                ReadData1_ID_current;
```

其中将 WB 的优先级别放在了 MEM 之前，因此前递的 数据是错误的。

（4）修正效果

将 MEM 的优先级调整到 WB 之后，该错误成功解决

（5）归纳总结

该该错误是由于没有考虑到同时出现两个数据相关的这种情况，导致没有考虑优先级关系。

4、错误 4

（1）错误现象

在对 cpu 进行行为仿真时，控制台在某处报错。

（2）分析定位过程

发现 pc 不正确，因此判断是之前有跳转指令执行错误。

（3）错误原因

发现错误出现在以下指令处：

```
bfc006dc: 8d2a0000  lw  t2,0(t1)
/media/sf_xjz/class/ucas/ucas18_19/develop/lab_all/uca
bfc006e0: 15400007  bnez  t2,bfc00700 <wait_1s+0x30>
```

当 bnez 指令到 ID 阶段时，前一条 lw 指令处于 EX 阶段，其值在 WB 阶段才能返回，因此需要等待两拍，而我之前没有考虑这种情况。

（4）修正效果

当出现 EX 为 lw 指令，并且 ID 处的指令需要读该寄存器时，需要阻塞两拍。阻塞过后该错误消失

（5）归纳总结

该错误是由于对前递的情况未考虑清楚，忽略了一种情况。

5、错误 5

（1）错误现象

在对 cpu 进行行为仿真时，控制台在最后几个测试点报错。

（2）分析定位过程

发现 pc 不正确，因此判断是之前有跳转指令执行错误。发现在 beq 指令之前有一条 MFHI 指令，并且存在数据相关。

（3）错误原因

没有考虑 HI 和 LO 的数据相关，例如，如果 ID 阶段内的指令需要度寄存器 S，且 EX 阶段指令要将 HI 寄存器的值拷贝到 S 内，则应该前递 HI 内的数据。

（4）修正效果

增添关于 HI 和 LO 的数据相关后，错误成功解决。

（5）归纳总结

该错误是由于没有对前递考虑全面，之前认为 HI 和 LO 不会出现数据相关。

6、错误 6

（1）错误现象

仿真通过，但是上板发现 LED 灯没有变化。

（2）分析定位过程

发现 message 中有一些 critical warning，其中一个是 div 模块的一个 input 端口出现错误。

（3）错误原因

检查发现错将 div 模块的输出 s 定义为 input，因此上板错误。

（4）修正效果

将 input 改为 output 之后，重新上板验证通过。

四、实验总结

本来觉得开始的已经够早了，之前将乘法器和除法器写完后觉得假期后两天就能将这个实验完成，但是发现加上 valid，allowin 等信号，并且实现前递之后出现很多错误，尤其是加上 HI 和 LO 的数据相关时需要考虑很多方面。因此调试持续了很长时间，最后调了很长时间才调通。晚上两点多行为仿真通过后想着终于能够睡觉了，发现上板出错.....心态崩了。不过幸好最后还找到了错误。