实验 5 报告

第 24 小组
张传奇、谈清扬、孔静

一、 实验目的

熟悉并掌握流水 CPU 的原理和设计

通过实践加深对课本上理论的理解理解片上系统的概念，并学会简单 SoC 的搭建

二、 实验任务

（一）设计

1、取指

（1）工作内容

根据 PC，计算出 nextPC，从而从指令 ROM 中取得下一条指令。

（2）工作目标

从指令 ROM 中获取所需指令。

2、译码

（1）工作内容

根据指令，分析指令，生成各种控制信号。

（2）工作目标

分析指令获得控制信号。

3、执行

（1）工作内容

受控制信号控制，执行各类的计算，生成访存地址传给 data ram。

（2）工作目标

正确执行计算。

4、访存

（1）工作内容

根据控制信号以及数据，在数据 RAM 内读写，并处理读出的数据。

（2）工作目标

正确读写数据 RAM。

5、写回

（1）工作内容

将所需的数据，写回到寄存器堆，或者 cp0，hi，lo 寄存器内，并能够处理例

外，修改相应 cp0 的值。

（2）工作目标

成功写回寄存器堆，并能处理例外。

（二）实现

根据所给资料，整理 61 条指令的信息，整合，写好各种控制信号的生成。分别完成五级流水数据通路图中大大小小的模块，传递好数据，将每一部分以相应的时序逻辑/组合逻辑完成。最后将这五级连线，整理成一个完整的五级流水线。

（三）验证

1、ISE 仿真

用老师给的 lab5_test 进行仿真，比对正确答案，运行成功会出现 PASS。

2、FPGA

能在 FPGA 板上成功运行。

三、 实验设计

（一）设计方案

1、总体设计思路

先按不同的级来设计模块，最后在总体上连接各个模块，完成前递信号的逻辑和例外入口的生成，组装成整个 cpu。

2、取指

因为使用了同步 RAM 来存储指令，所以我们需要每次都传入的是下一条指令的 pc 值，因此需要通过译码级的逻辑来判定转移相关数据和控制信号，根据不同的跳转类型，经过加法器和选择来生成 nextpc 信号，pc 每次会更新成为 nextpc 的值。

3、译码

根据 61 条指令的特征，定义了 61 个信号分别表示是否是这个指令。然后定义了 branch_type[5:0]、rs_need、rt_need、rt_write、rd_write、bypass_op、logic_type[4:0]、shift_type[2:0]、imm_op、unsigned_op、sub_op、link_op、store_type[2:0]、store_op、alu_res_sel[1:0]、load_op、hi_we、lo_we、cp0_write、cp0_cs[4:0]、cp0_sel[2:0]、wb_mux[1:0]、we 这二十四种信号（详情见附表 lab5ctrl.xlsx）。

根据跳转相关信号，和各种判断逻辑，生成是否跳转的信号和计算跳转相关的数据

根据数据相关，需要的前递信息：寄存器写入地址，寄存器写入信息，信号是否准备好和信号是否有效，来进行冲突和前递的判断。

4、执行

获取上一层传递的经译码获得的控制信号 pipe2_ctrl_info_in，和经访问寄存器获取的操作数和立即数 pipe2_data_info_in 等，进行各类数值或逻辑运算，生成仿存控制或数据，并查出地址不对齐，溢出等例外。

各类运算模块，按照实验指导书的设计，在考虑电路特性的情况下，进行了设计，包括了加法器及溢出判断逻辑、less 信号生成，移位器，逻辑运算器。对于两类访存指令，将利用加法器获得的地址值传给下一级所需的 addr。并截取最后两位作为 offset，根据访存类型：字、半字，检查是否发生地址不对齐例外。Store 类指令，还需通过访存类型，生成写使能掩码，和拼接出写入数据，

此级检测两种例外，运算类的溢出例外，访存类的地址不对齐例外。由于不太理解指导书中的例外写法，所以对加法器做了一定的修改，提取出第 30 和第 31 位进位进行溢出判断。

考虑到流水线可能发生的数据相关阻塞，本级还留出了数据输出口，进行前递。

5、访存

获取上一层传来的数据，pipe3_ctrl_info_in，包含了这层所需的 load_type，load_op 还有 exc-type 的部分数据。pipe3_data_info_in，由上一层的 mem_value 以及 offset 组成。

数据 RAM 模块，已经提供好，只需要正确接入即可。

load 数据处理模块，根据上一级传来的控制信号 load_type，以及上一级传来的数据信号 mem_value、offset，还有从数据 RAM 获得的 dout 对数据进行处理。学习了老师的书写方式，一个 assign 搞定（这样觉得可能就不需要单独搞个 module，不过写都写了）。

由 load_op 和例外信号决定传入下一级的 pipe3_data_info_out，并将下一级所需的 pipe3_ctrl_info_out 控制信号传出。

6、写回

获取上一层传来的数据，pipe4_ctrl_info_in，由例外类型 exc-type、pc、写回寄存器的地址 dest、指令是否为分支延迟槽指令 btype、高位寄存器写使能 hiwe、低位寄存器写使能 lowe、cp0 写使能 cp0we、cp0 读/写地址 cp0cs、写回寄存器数据选择器 wbmux、寄存器写使能 regwe 信号。pipe3_data_info_in，即上一层传来的 wb_value。

由 fpga_reset 以及 2 个寄存器，生成只持续一拍的 rst 信号，并输出。

由 pc 受 is_bd 控制得到结果传入 cp0_reg。

根据 cp0 相关信号，优先级，有例外处理例外，有 eret 处理 eret，最后有指令就处理 mfc0 和 mtc0 操作（沉迷于模仿老师代码，都不是很愿意写 case），并将 is_bd 信号放在里面，使其能同步通信。并将 statues 寄存器[1]exl，cause 寄存器[6:2]exc_code，epc 寄存器组合成 e_out 传出

由高位/低位寄存器的写使能信号控制，是否将数据写入其中。

由写回寄存器数据选择器控制 wdata 的值，waddr 连线 pipe4_ctrl_info_in 的 dest 部分，we 连线 pipe4_ctrl_info_in 的 regwe 部分，控制写回寄存器的操作。

（二）验证方案

1、总体验证思路

先用老师提供的 lab5_test 进行不同指令的仿真，然后跑 FPGA 板。

2、验证环境

ISE 仿真软件和 FPGA 板。

3、验证计划

跑 ISE 仿真以及 FPGA 板，与正确答案进行比对。

四、 实验实现

（一）实验交付说明

都放在 ../lab5_test/mycpu_verif/mycpu_rtl 下面。

（二）实现说明

cpu.v 是 cpu 的顶层

pipe0_if 是取指级

chg_vaddr 是虚实地址转换部件

pipe1_id 是译码级

branch_check 是分支判断部件

hazard_check 是前递机制和冲突检测

pipe2_ex 是执行级

exec_gadgets 是执行级需要的一些部件

pipe3_mem 是访存级

pipe4_wb 是写回级

mux4 是个 32 位 4 选 1

reg_file 是寄存器堆

五、 实验测试

（一）测试过程

先修改了大量编译错误，然后发现第五个周期才获得下一个指令，然后追踪发现传递读指令地址的时候，传递了高位，忽略了低位，所以四个周期才能获得一个新的指令。

然后各种地方卡指令，发现了大量写的时候的粗心错误，或者看任务指导书看错的错误。以下是我还记得的东西。

发生整数运算溢出例外的时候，12 计算失误，写成了 5’b10100，应该是 5’b01100。

写 cp0_count 寄存器的时候，与 cp0_count 的加一操作，冲突，修改为 if/else。

发生 eret 的时候，对只读的 cause 寄存器进行了写入。与 track 不符，后删去发生 eret 时对 cause 寄存器的操作。

lui 指令，提取立即数部分错误，赋值成了扩展立即数扩展符号位部分的值。

sltiu 指令，没有考虑无符号数溢出判断的特殊性。

add 指令，拷贝代码时候错误，进位信号没有全部替换更改。

sll，移位信号忘记声明，补上信号位数后，逻辑移动超过 2 位数（3’b）的时候，不知为啥 verilog 的>>>符号失效，模仿老师给的代码中的部分，重写了一

个。

mfhi/mflo/mthi/mtlo，都以为是对 rt 对应寄存器操作，所以全部错误，修改译码级的信号分配修复了 bug。

前递路径必须包含这个信息是否有效，不然会因为无效信息而导致流水线错过有效信息而空等。

（二）测试结果

PASS。

六、 成员分工

小组成员三人，主力大佬传奇完成了取指译码以及统筹规划、小谈完成了执行、小静完成了访存和写回，一起调试 bug、写报告。

七、 实验总结

（一）张传奇

累。

（二）谈清扬

从大一上完数电的 verilog 部分就跟自己讲，以后绝对不能碰 verilog，绝对不去干硬件，然后就上到了第三个有 verilog 编程的学期 orz 心累。第一学期可能是语法，第二学期组成原理开始用 verilog 实现一些稍复杂的逻辑，这一学期的重点可能就是能够真正的上版验证了。读任务书的时候，感受到了老师多次强调脑袋里面要有电路！

然后下面的同学说错了，是我最弱，嗯。

（三）孔静

61 条 mips 指令的五级流水线 cpu，真的是一个工程浩大的任务。

鉴于我是组里最弱的人，所以被分配了最简单的访存和写回，然后帮助队友干了一些 dirty work（无脑的大量的烦躁的工作）。其实真的不烦计系这些课的作业，只是很烦，当它们堆在一起的时候，通宵达旦一周到周四写完数据库 12306

火车票系统，才赶来写体系结构，又是新一轮的住在图书馆里，就有点烦躁。

不过，毕竟作业，不能对不起队友，还是得做啊，所以说这些任务培养了我的耐心，让我不那么容易烦躁。

最后，感谢一下老师们的倾情奉献，很详细的指导书（虽然我没看完），但看过的部分，感觉就好像明白了很多。

八、参考文献

无。