

实验 2-1 报告

学号：2016K8009929029

姓名：张丽玮

一、实验任务（10%）

这个实验是在原有多周期 CPU 的基础上增加静态五级流水的功能，并实现延迟槽。延迟槽不只是原来的 nop 指令，可以是任意指令。五级流水的指令之间可不考虑相关性。

二、实验设计（30%）

总的来说，从 mips_cpu 顶层结构出发，直接跳过先前冗余繁杂的数据处理，而是将数据处理放入每个状态和模块中。因此五级流水就单独 IF、DE、EX_ID、MEM 和 WB 每个状态设计一个模块，而每个模块承接上一个模块的输出进行输入，从而达到五级流水的控制效果。

1、fetch 模块

```
input wire clk,
input wire rst,
// data passing from the PC calculate module
input wire [31:0] PC_next,
// interaction with inst_sram
output wire inst_sram_en,
output wire [31:0] inst_sram_addr,
input wire [31:0] inst_sram_rdata,
// data transferring to ID stage
output reg [31:0] PC_IF_ID,
output reg [31:0] PC_add_4_IF_ID,
output reg [31:0] Inst_IF_ID
```

fetch 模块主要完成一个取指操作，因此实际输入进来的下一条 PC 指令以及读入数据 rdata。而模块中用了两个 always 是为了实现一个两拍操作，达到下一条指令在下一拍取到的效果。

2、decode 模块

decode 模块是一个译码操作模块，接口过多不予列举，但这么多接口实际上就是完成了从指令出发，通过 control 模块获取指令应该进行的操作，后缀 ID 代表从译码阶段传去 PC 跳转判定的指令，而 ID_EXE 则代表从译码阶段传去执行阶段的相关数据。

3、execute 模块

这是一个执行模块，完成的主要操作是调用运算器进行执行。接口过多不予列举，不过接口命名可以看出是从哪个模块往哪个模块的数据。主体部分是三个多路选择器与一个运算器执行运算操作。

4、memory 模块

简单明了的存指操作，接口命名可以看出数据来源和去向。主体部分就是一个时钟节拍控制的赋值操作。

5、writeback 模块

一个写回模块。不需要时钟信号控制，直接赋值即可。

```
input wire clk,
input wire rst,
// control signals passing from MEM stage
input wire MemToReg_MEM_WB,
input wire [ 3:0] RegWrite_MEM_WB,
// data passing from MEM stage
input wire [ 4:0] RegWaddr_MEM_WB,
input wire [31:0] ALUResult_MEM_WB,
input wire [31:0] PC_MEM_WB,
input wire [31:0] MemRdata_MEM_WB,
// data that will be used to write back to Register files
// or be used as debug signals
output wire [ 4:0] RegWaddr_WB,
output wire [31:0] RegWdata_WB,
output wire [ 3:0] RegWrite_WB,
output wire [31:0] PC_WB
```

6、其他模块

实际上还有一个 pc 跳转模块，是为了根据是否有跳转指令来判断下一个 pc 是+4 还是跳转。control 模块较上次进行了一些改动，但是实现方式和结果相差不大。alu 和 reg_file 模块基本未改。decode 模块在 tools 文件中，基本就是龙芯的 tools 复制过来。

三、实验过程（60%）

（一）实验流水账

9月19日，晚六点到半夜不知道几点，重构了 cpu

9月20日，晚八点到一点，cpu 的 debug

9月21日，下午两点到晚十点，阅读五级流水讲义，改写 pipeline 模块

9月22日，下午两点到四点，尝试嵌入 pipeline 模块失败

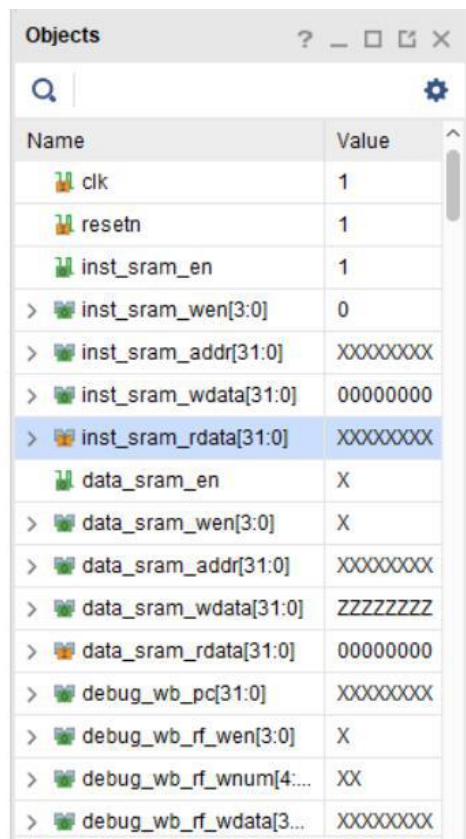
9月24日，下午两点到第二天早上，重新构建流水线结构，debug，调试测试通过

（二）错误记录

具体描述实验过程中的错误，环境问题、仿真阶段、上板阶段的都可以记录。

1、错误 1

（1）错误现象



Name	Value
clk	1
resetsn	1
inst_sram_en	1
inst_sram_wen[3:0]	0
inst_sram_addr[31:0]	XXXXXXXX
inst_sram_wdata[31:0]	00000000
inst_sram_rdata[31:0]	XXXXXXXX
data_sram_en	X
data_sram_wen[3:0]	X
data_sram_addr[31:0]	XXXXXXXX
data_sram_wdata[31:0]	ZZZZZZZZ
data_sram_rdata[31:0]	00000000
debug_wb_pc[31:0]	XXXXXXXX
debug_wb_rf_wen[3:0]	X
debug_wb_rf_wnum[4:...]	XX
debug_wb_rf_wdata[3:...]	XXXXXXXX

全线飘红，wdata 为高

（2）分析定位过程

有 Z 调 Z，实际上是 wdata 接口接错。

（3）错误原因

wdatas 所接接口接口名写错，该接口未定义。

（4）修正效果

更正接口名，成功便成 XXXX

（5）归纳总结（可选）

属于写代码和重构时的粗心。善用查找和替换。

2、错误 2

（1）错误现象

Name	Value
clk	0
reseth	1
inst_sram_en	1
> inst_sram_wen[3:0]	0
> inst_sram_addr[31:0]	XXXXXXXX
> inst_sram_wdata[31:0]	00000000
> inst_sram_rdata[31:0]	XXXXXXXX
data_sram_en	X
> data_sram_wen[3:0]	X
> data_sram_addr[31:0]	XXXXXXXX
> data_sram_wdata[31:0]	XXXXXXXX
> data_sram_rdata[31:0]	00000000
> debug_wb_pc[31:0]	XXXXXXXX
> debug_wb_rf_wen[3:0]	X
> debug_wb_rf_wnum[4:...	XX
> debug_wb_rf_wdata[3:...	XXXXXXXX

全线飘红，输入也为 XXXXX

(2) 分析定位过程

按照逻辑关系寻找 raddr，与 pc 有关，猜想是 pc 的问题。

(3) 错误原因

先前用的 pc 跳转逻辑在此流水线中不再适用，不再是简单多周期的跳转方式，需要另外判定。

(4) 修正效果

重新设计一个 pc 跳转模块，成功解决，

(5) 归纳总结（可选）

要从简单多周期的思维方式转变为流水线的思维方式。

3、错误 3

(1) 错误现象

```
[ 2177 ns] Error!!!
reference: PC = 0xbfc0038c, wb_rf_wnum = 0x04, wb_rf_wdata = 0xbfb00000
mycpu      : PC = 0xbfc00008, wb_rf_wnum = 0x08, wb_rf_wdata = 0xxxxxXxxx
```

(2) 分析定位过程

判断是 pc 出错，外部逻辑无误，定位到 alu。

(3) 错误原因

之前重构的 cpu，alu 采取 12 位 op 进行 control，后来因为不知道如何实现示例模块的 pipeline 调用而重新返回原本思路，但外部调用时忘记了 alu 此时是 12 位的 op，仍然当做四位来调用。

(4) 修正效果

重新采用 lab1 的 alu，成功解决。

(5) 归纳总结（可选）

tcl 的 warning 同样很重要，有些接口不符警告能够帮你发现问题。

3、错误 4

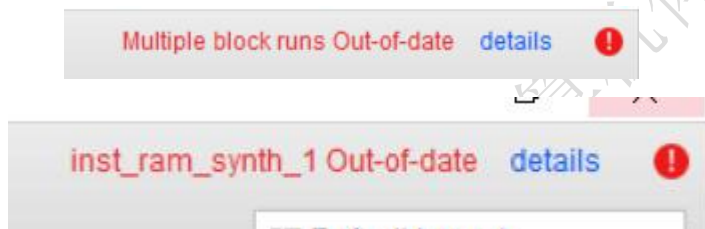
(1) 错误现象

```
[ 2137 ns] Error!!!  
reference: PC = 0xbfc0038c, wb_rf_wnum = 0x04, wb_rf_wdata = 0xbfb00000  
mycpu      : PC = 0xbfc0038c, wb_rf_wnum = 0x04, wb_rf_wdata = 0xffffbfb0
```

> Re...	1	Array
> Re...	04	Array
> AL...	XXXXXXXX	Array
> Me...	XXXXXXXX	Array
> PC...	bfc00390	Array
> AL...	00000000	Array
> AL...	ffffbfb0	Array
> AL...	ffffbfb0	Array
> Re...	05	Array

(2) 分析定位过程

定位到 regfile 写回，但是发现逻辑并无问题，然后询问同学发现



是 inst_sram 出了问题

(3) 错误原因

Inst_sram 的 generate 有问题

(4) 修正效果

自己重试了多次未果，包括重下环境，多次重新载入，之后求助同学。

(5) 归纳总结（可选）

四、实验总结（可选）

我可能要重写好多遍 cpu 才能完全理解一个实验。原本有一个根据示例代码改出来的很漂亮的 cpu 想要在上面直接调用 Pipeline 模块改出五级流水，并且以为这个比较简单就稍微拖了点时间，结果发现自己并没能完全理解老

师的思路，不知道该如何合理调用这个 pipeline，而 cpu 虽然看着漂亮但是思路不是我的习惯思路，很难下手，最写五级流水基本又相当于重写了一遍。

祝大家中秋快乐已经过时了，以后就祝大家中秋写完代码吧。

国科大B62009H计算机体系结构研讨课17-18秋季