

# 计算机组成实验六

## 简单的类MIPS多周期流水化处理器实现

### 5140219355

### 朱耀明

[计算机组成实验六 1](#_Toc451282657)

[简单的类MIPS多周期流水化处理器实现 1](#_Toc451282658)

[5140219355 1](#_Toc451282659)

[朱耀明 1](#_Toc451282660)

[1.总述 3](#_Toc451282661)

[1.1 实验名称 3](#_Toc451282662)

[1.2 实验目的 3](#_Toc451282663)

[1.3 实验范围 3](#_Toc451282664)

[2.流水线 3](#_Toc451282665)

[2.1工程文件配置 3](#_Toc451282666)

[2.2流水线分析 4](#_Toc451282667)

[2.3指令内存模块 5](#_Toc451282668)

[2.4 Top模块修改 6](#_Toc451282669)

[2.5 寄存器更新 8](#_Toc451282670)

[2.6 各模块实例化与MUX，reg,wire间赋值 10](#_Toc451282671)

[2.7 冒险处理 11](#_Toc451282672)

[3.仿真 12](#_Toc451282673)

[3.1MIPS指令 12](#_Toc451282674)

[3.2仿真结果 12](#_Toc451282675)

[4.上板 12](#_Toc451282676)

[5.小结 14](#_Toc451282677)

[5.1心得 14](#_Toc451282678)

[5.2实验细节与建议 14](#_Toc451282679)

# 1.总述

## 1.1 实验名称

简单的类MIPS多周期流水化处理器实现。

## 1.2 实验目的

理解CPU的Pipeline，对Data Hazard，Branch Hazard有初步认识。

## 1.3 实验范围

1. ISE的使用

2. VirtexII Pro实验板的使用

3. 使用VerilogHDL进行逻辑设计

# 2.流水线

## 2.1工程文件配置

工程文件配置与之前一致，在 Device Properties中各属性填写如下：

Product Category: ALL

Family: Spartan3E

Device: XC3S500E

Package: FG320

Speed: -4

Synthesis Tool: XST(VHDL/Verilog)

Simulator: ISim(VHDL/Verilog)，也可用Modelsim仿真

Preferred Language: Verilog

实验指导书上lab6的配置与之前略有不同，在最后的实验细节环节分析

## 2.2流水线分析

MIPS处理器的采用了5级的流水线：

IF(instruction fetch，取指)->ID(instruction decode，译码)->EX(execution，执行)->MEM(memory access，访问内存)->WB(write back,回写)。与控制信号以及硬件的大致联系见Fig.1

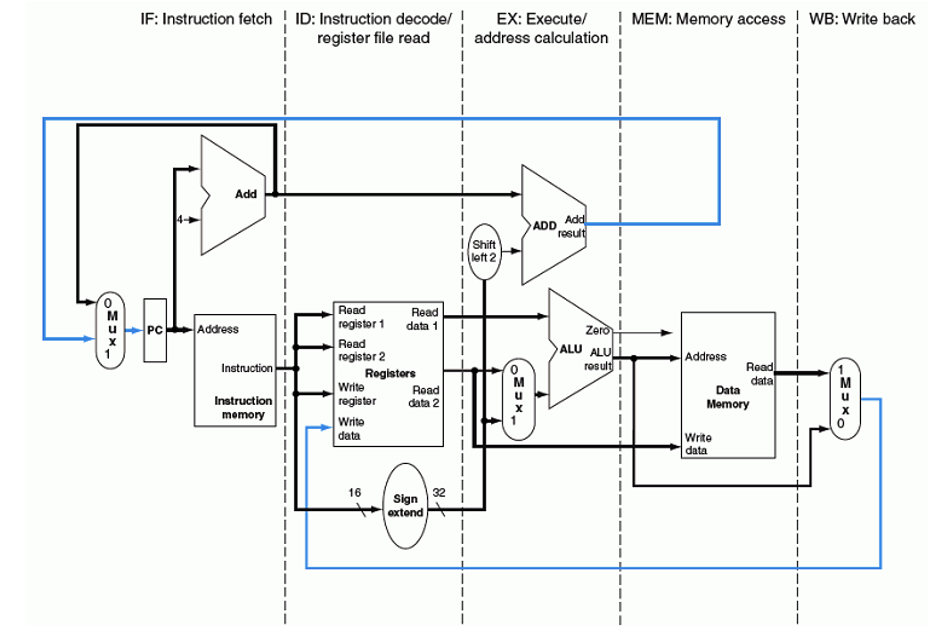


Figure 1

Control 的输出需要被保存下来，供后续每级流水使用。见Fig.2，Fig.3

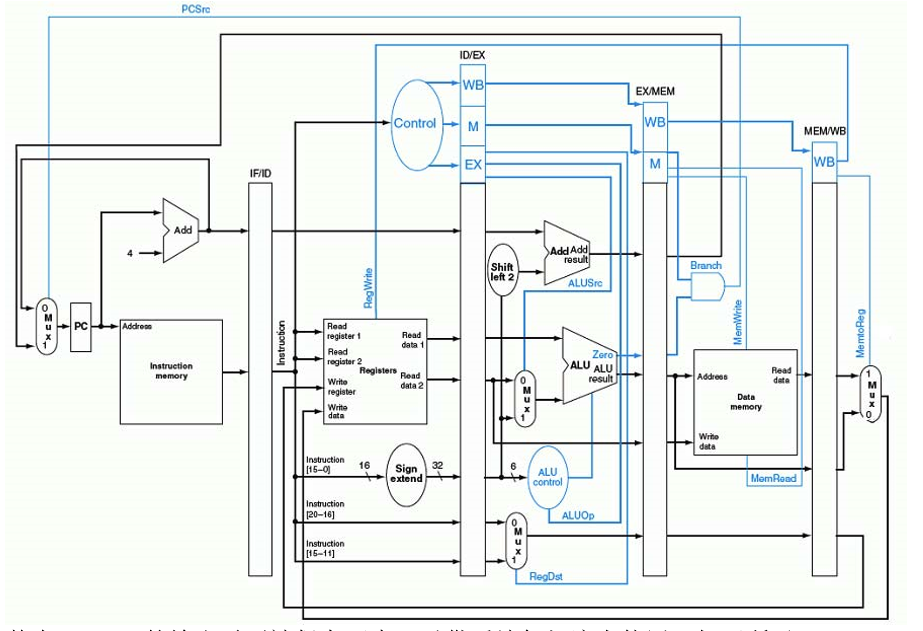


Figure 2

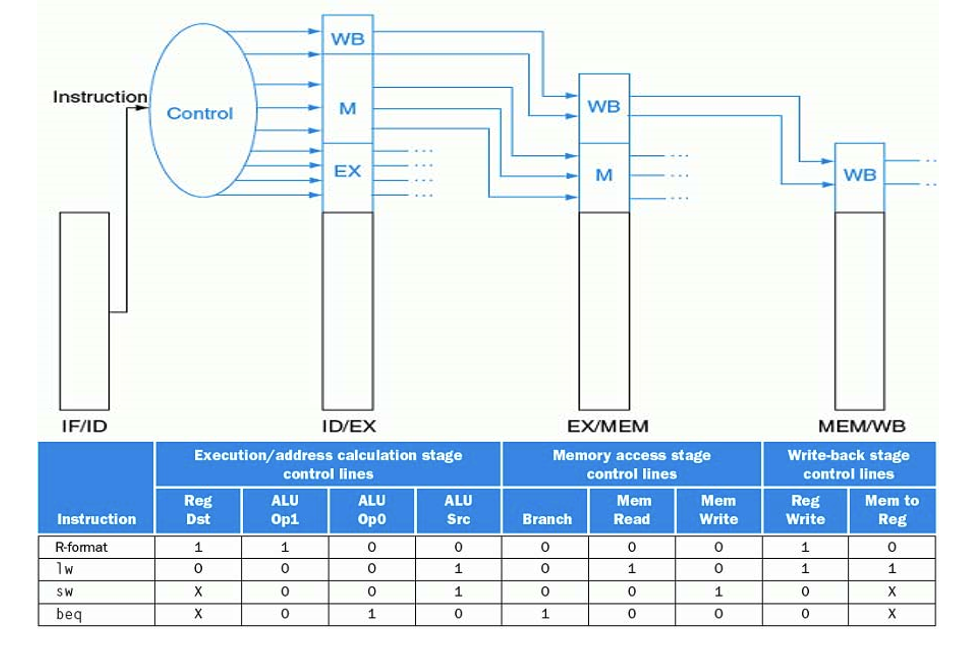


Figure 3

这样会就需要寄存器reg型变量取代之前的wire变量存储控制变量。由Fig.3，控制变量可以分为IF/ID，ID/EX，EX/MEM，MEM/WB四级中间寄存器单元。每一级在时钟周期信号的上升沿会继承一部分上一级的变量。

在同一时刻，流水线的应用使得每个部分能单独工作，提高了CPU的吞吐率。但可能造成冒险问题，MIPS处理器可能会面临结构冒险，数据冒险和控制冒险三种冒险。解决会在之后叙述。

## 2.3指令内存模块

相比于lab5，lab6的指令内存不再是直接写在Top模块里，而是额外撰写了InstructionMemory模块。

新的指令内存模块输入输出定义与内存单元定义如Fig.4

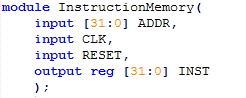


Figure 4

ADDR为指令地址，INST输出ADDR对应的指令。instMemFile储存指令，这里之模拟了64\*4B大小的指令内存。为了之后的上板实验方便，初始化我继续采用了直接赋值而不是读取二进制文本的方式。如Fig.5

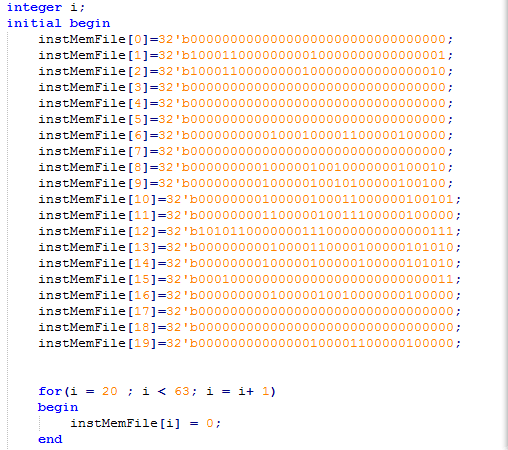


Figure 5

因为lab6使用的储存单位也为32bit，因此取指方法与lab5中相似，见Fig.6

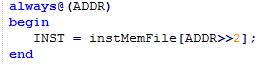


Figure 6

## 2.4 Top模块修改

输入输出与lab5相比没有变化。

如前所述，lab6使用了流水线后，需要额外的寄存器储存控制信号等。实验中控制变量非常多，需要命名规范。

本次实验我的命名规范如下：对于变量名的主体，和lab5一样使用全大写加下划线的方式命名，且尽量直接使用lab5中出现过的名称。

寄存器变量在变量名的主体前加上前缀，说明该变量属于哪一级。例如属于ID/EX的加上前缀ID\_EX\_，以此类推。见Fig.7

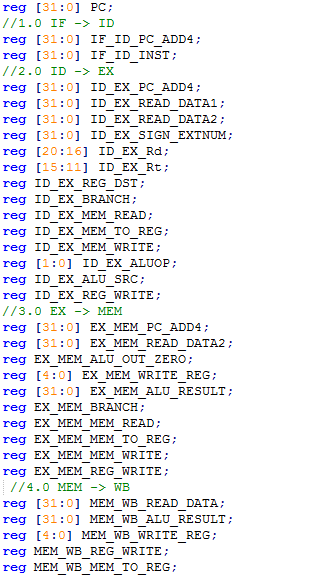


Figure 7

此外，因为每个时钟周期reg型变量都会被更新，而这各个模块又在每个周期需要输入输出，因此不能直接将reg变量与各个模块相连，这就需有一部分wire型变量。wire型变量都是在流水线特定的一级生效的，直接采用该级作为前缀。如WRITE\_DATA为写入的数据，在WB回写阶段生效，则命名为WB\_WRITE\_DATA。如Fig.8

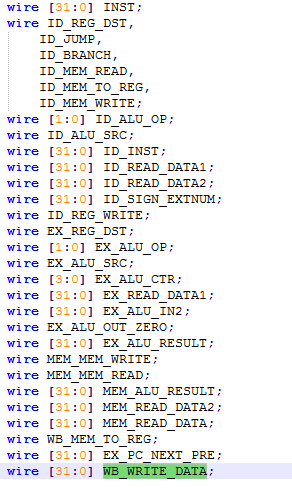


Figure 8

## 2.5 寄存器更新

时钟上升沿完成该步操作。因为这一部分代码太长，就不将always@(posedge clk)这条语句放入截图中了。因为lab6中控制信号使用了寄存器，为了赋值时关联的寄存器变量不造成混乱，使用非阻塞赋值操作，在当前所有语句结束后完成赋值。见Fig.9

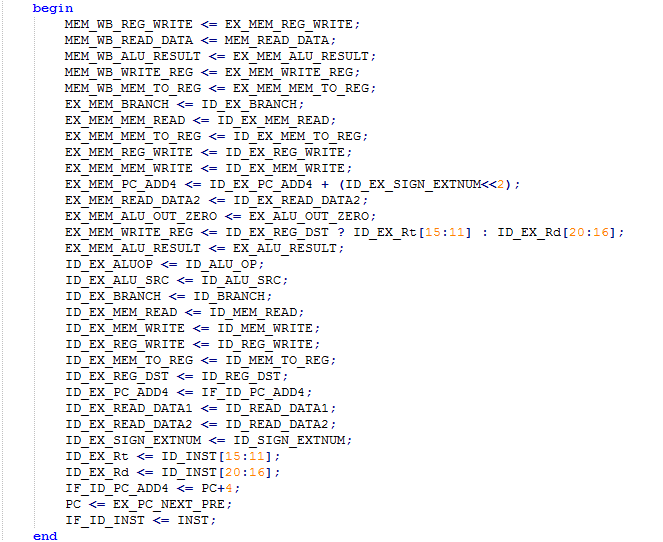


Figure 9

重置、初始化见Fig.10

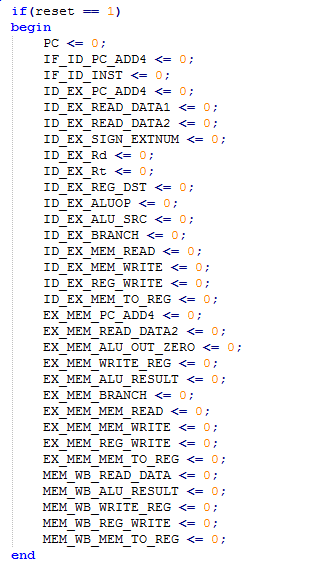


Figure 10

寄存器模块内部的重置与lab5完全相同，不再赘述。

## 2.6 各模块实例化与MUX，reg,wire间赋值

这一部分与lab5没有多少差别，需要注意的就是有许多变量主题相同，使用时需要注意应该为哪一级流水线。这里只截图了一部分代码，如Fig.11,Fig.12。详细代码见工程文件。

在这里也一并实现了前面所述wire变量与reg变量间赋值联系。因为主体名是相同的，所以找到wire对应的reg变量不难。

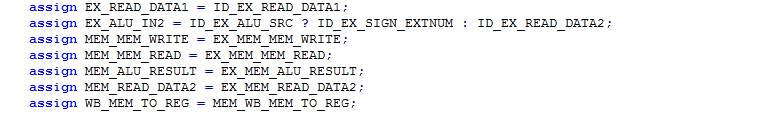


Figure 11

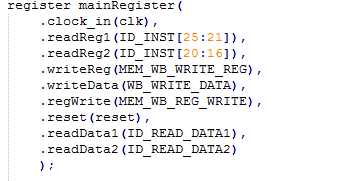


Figure 12

## 2.7 冒险处理

前面提到了结构冒险，数据冒险和控制冒险。结构冒险在本实验中没有考虑。数据冒险可以通过数据旁路解决，在我的设计中没有使用Verilog解决，而是在指令上解决，这会在之后仿真一部分叙述。重点放在了控制冒险上。

控制冒险是指遇到jump或者beq需要跳转地址的指令时，在指令跳转之前已经执行了后面的一些不应该执行的指令的情况。对此，我采用的方法是：当检测到需要跳转时，置零IF/ID，ID/EX,EX/MEM上流水线已完成且与写入相关的控制信号以防止内存、寄存器被错误修改，之后重新再载入流水线，见Fig.13

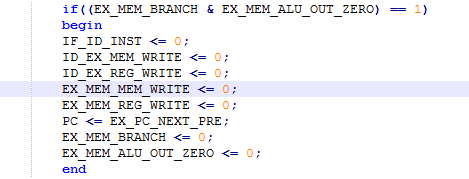


Figure 13

# 3.仿真

## 3.1MIPS指令

因为实验实现的处理器没有解决数据冒险的问题，因此需要在必要的位置加入空指令防止数据冒险。我对lab5的指令进行了略微修改，见Fig.14，与之前一样，借助了开源MIPS开发工具MARS

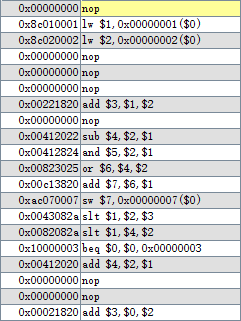


Figure 14

指令内存的初始化见前面的Fig.5

仿真激励文件等与lab5完全相同，不再赘述

## 3.2仿真结果

见Fig.15，可见之前的设计完成了流水线的要求

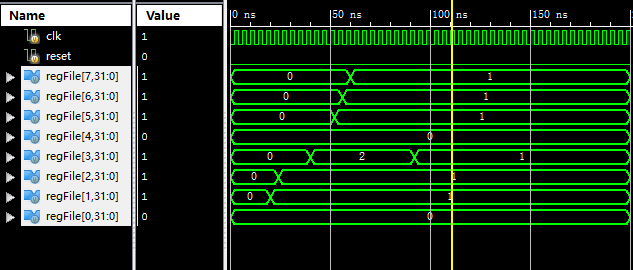


Figure 15

# 4.上板

设计的思路和lab5完全相同：

1.使用一个按动开关作为showPC，当按下时，LED显示当前PC后8bit；否则显示选中的寄存器后8bit

2.使用一个拨动开关作为reset，打开为1，重置数据

3.使用三个拨动开关作为寄存器选择器。可以选择0-7号一共8个寄存器，正好覆盖了自己撰写的MIPS代码中的所有被访问的寄存器。

对Top文件的修改（添加LED，分频器等等）也和lab5的修改如出一辙。因为仿真的初始化已经使用了直接在模块中赋值的方法，不需要再做修改了。下面只列出了ucf部分的代码，其他见工程文件。

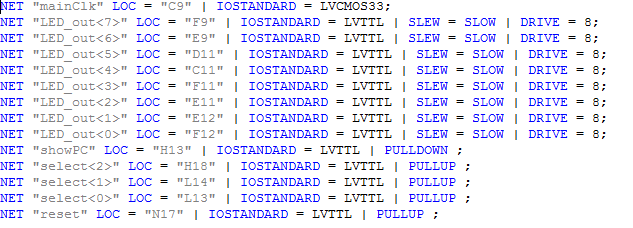


Figure 16

上板后进行观察，与最初的设计相符合。设计成功！至此，整套实验都已完成！

# 5.小结

## 5.1心得

lab6的变量名称相当多，这是我学习编程以来做过的变量最多的一次项目，这让我对命名规范性有了更深入的理解。除此之外，还有就是对流水线需要更仔细的分析，清楚各级指令直接的关系，还要对冒险处理机制有很清晰的认识。

最初我设计流水线时就只考虑了各级的控制信号，而忽视了设计用于各模块输入输出的信号，之后补加时对命名有不太规范，譬如EX级与MEM级都会用到ALU运算结果，因此造成了一些混乱。这也说明设计时要先有一个总体规划。

在lab5里我已经提到，设计一套指令集并且设计硬件对其进行实现是一件非常不容易的事情。而流水线这样精巧的设计、对各种冒险微妙的处理方案更是让我感慨前辈们对计算机性能设计孜孜不倦的追求才带来了几十年来计算机技术的惊天飞越。

6周的《计算机组成实验》课程让我学习到了很多，也感受了很多，在最后我要感谢老师与助教的悉心指导！

## 5.2实验细节与建议

lab6实验指导书上使用的是Xilinx ISE11.1，比其他几个指导书上的版本要低，仿真工具为Modelsim-SE (verilog)。当我进行这样的设置后，在波形仿真时ISE提示我找不到仿真工具。后来我将其修改为之前几个实验的配置，才得以成功仿真。我使用的ISE为14.7版本，这也是很多同学自行安装使用的版本，建议老师之后修改一下实验指导书，使其适用于较高版本的ISE。