单周期CPU设计

实验报告

姓名：谢威宇

学号：1120172150

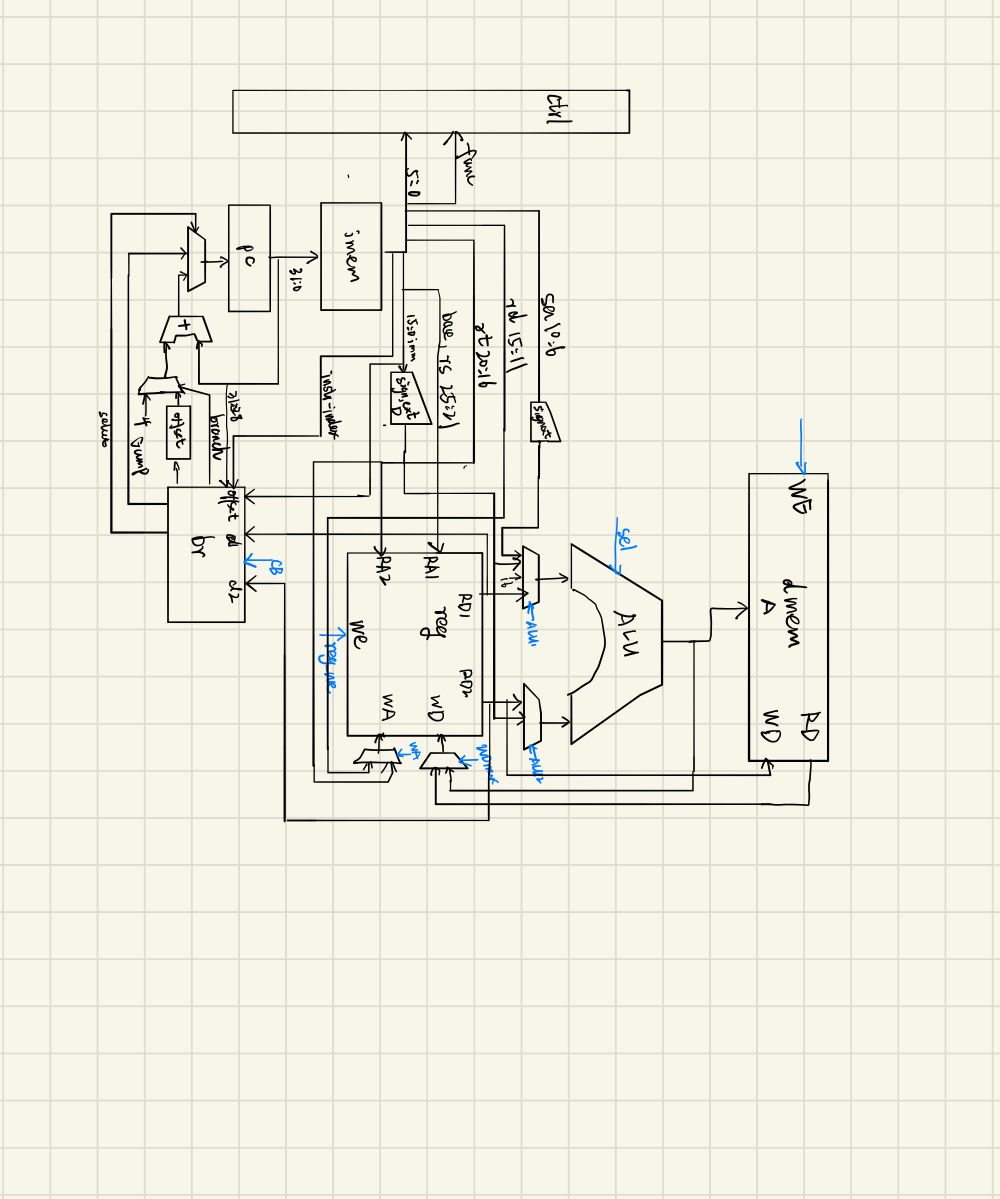
班级：07111701

日期：2020年9月13日

1. **实验目的：**

**使用Verilog HDL语言设计一个支持MIPS子指令集的单周期CPU，并进行仿真和给出结果。**

1. **CPU设计：**
2. **CPU支持指令集。本实验设计的单周期CPU支持以下共13条MIPS指令lui、addiu、add、sub、lw、sw、beq、j、ori、divu、addu、sll、srl。其中divu指令是我抽取到的随机指令。**
3. **CPU架构。本实验设计的CPU架构为哈佛架构，即数据与指令分开存储的架构。**
4. **本实验CPU的数据通路图如下所示。**

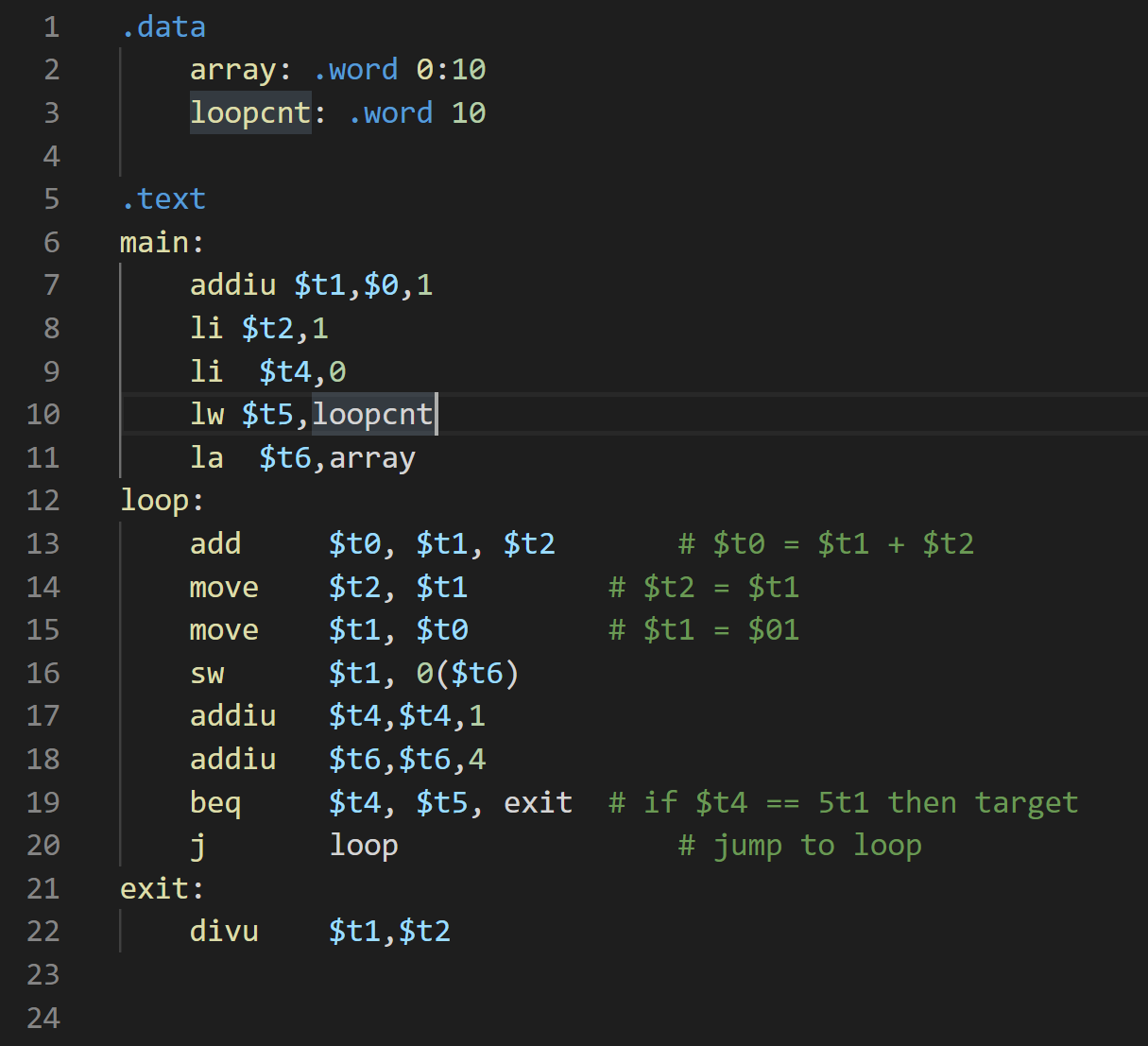
****

1. **主要指令的实现：**
2. **Addiu指令，直接将rs与立即数相加，结果置入rt中。**
3. **Beq指令，通过单独实现的br模块，读取rs和rt的值，比较后决定是否跳转。**
4. **J指令，同样通过br模块判断是否跳转。**
5. **一些特殊指令的实现：**
6. **Lui指令是把一个立即数装入rt寄存器的高位，并且把低位清零。在本实验中，先通过alu进行移位运算，得到的结果写入寄存器内即可，所以在alu左边数据接口的mux中有16这个选项。**
7. **Divu指令是用rs和rt寄存器内的数值相除，得到的商放入lo，余数放入hi。由于hi和lo是寄存器，需要保存数据，所以本实验将hi和lo置入alu模块中。因此，alu模块也需要时钟信号来是寄存器正常工作。**
8. **Sll指令是移位指令，其中移位数为sa。Sa只有5位，不同于立即数，所以它和立即数分别进行符号扩充。**
9. **控制信号：**

**对需要控制信号的mux等部件，输入相应指令的控制信号即可。**

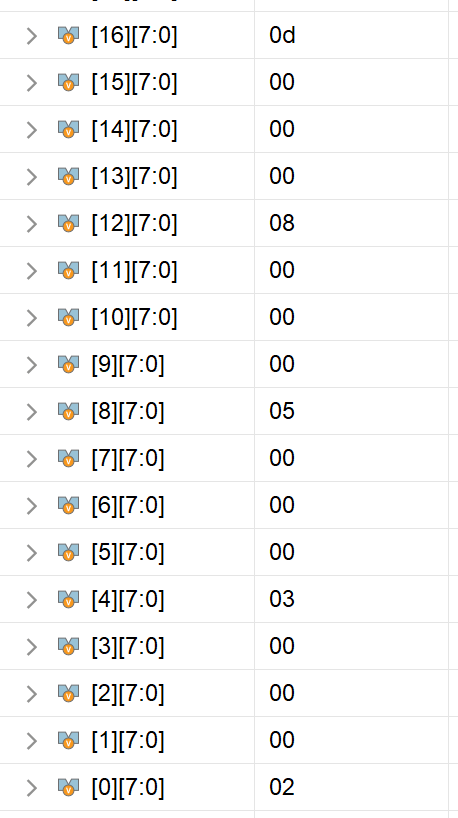
1. **Br模块：本实验将pc周边的组件全部收入br模块中，最终br模块的输出为下一周期的pc，方便了顶层组件的设计。**
2. **实验步骤**
3. **设计基本部件，例如寄存器堆，alu，指令存储器和数据存储器。**
4. **从PC开始，画出数据通路图，每条指令增加相应的数据通路和控制信号。**
5. **根据指令的功能列出控制信号真值表，并依此编写控制模块。**
6. **仿真，运行编写的MIPS程序。**
7. **实验结果**

**仿真所用MIPS程序功能为求斐波那契数列前十项，并把结果装入数组中。**

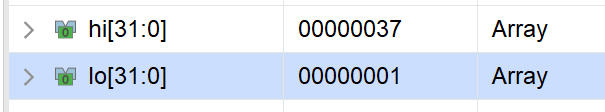


**loop cnt为10，表明循环运行10次，每次计算并存储相关的值。最后exit中divu为测试此指令专门写的代码。**

**程序运行结束后，数据存储器中的数据如下。**

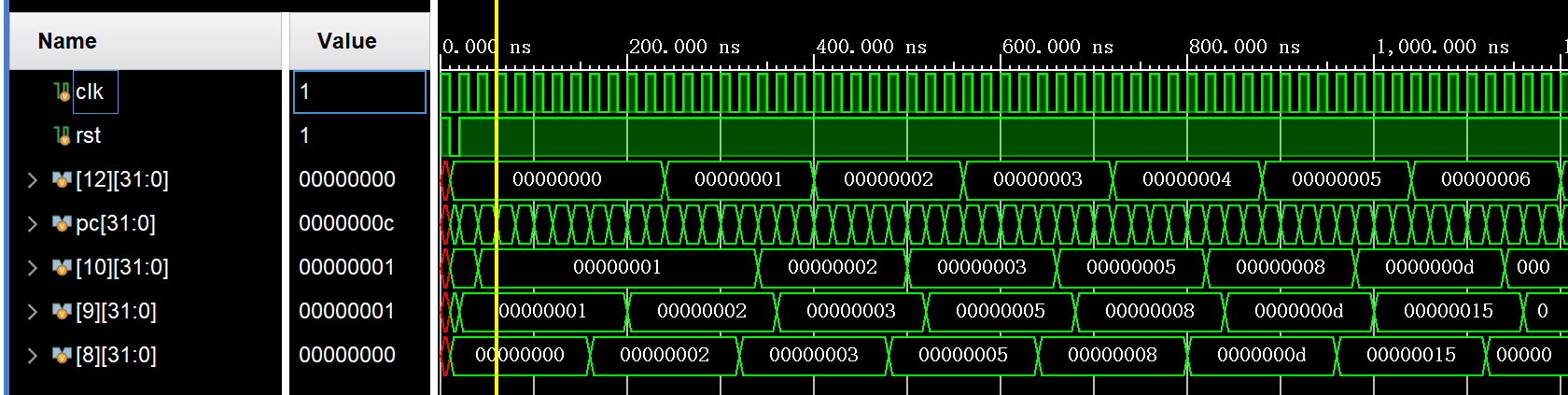


**可以发现成功的计算出了2，3，5，8，13等斐波那契数列值。再查看hi和lo的值。**

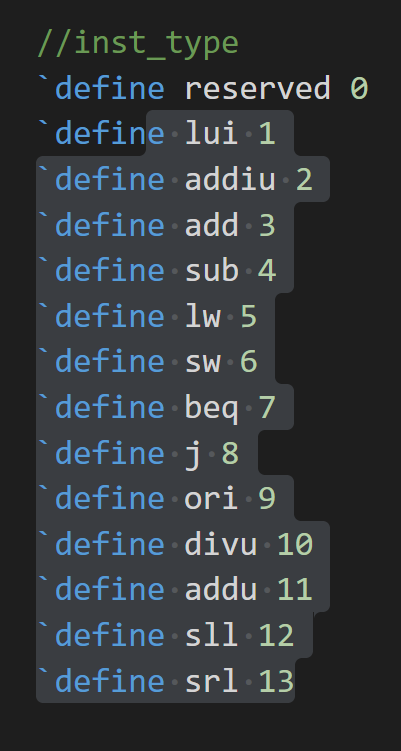


**可以发现是144/89 = 1······ 55 = 0x37，结果正确。**

**查看波形图，符合预期。**



1. **设计过程中遇到的问题以及解决方法。**
2. **Mars导出机器码难以读入。本实验设计的存储单元采用小端方式，字节寻址，但是Mars导出的文件为单字寻址方式。由于指令存储器其实不需要读写单个字节，所以我直接将指令存储器改为和Mars一样的寻址方式（当然和CPU的接口仍然是字节寻址，反正PC后两位一定为0）。但是对于数据存储器，我就不能将它改成和Mars一样的形式。所以我单独写了一个python脚本来解决这个问题。**
3. **控制模块的控制信号可读性差。由于控制信号基本是没有规律的，所以直接写二进制会导致可读性很差，给设计和调试代码麻烦。因此本实验采用宏的形式对各个控制信号进行编码。保证各文件的控制信号统一，并且可读性高。例如下图就是指令类型的控制信号约定。**



1. **Mux复用。设计中有许多不同位宽、不同输入数目的mux。为了减少代码的编写，本实验采用了参数化模块的mux设计，减少了代码量。**
2. **设计总结和心得体会**

**虽然单周期CPU的结构简单，但是作为一个CPU，它还是相当复杂。本实验主要参考了助教ppt上的设计，并按照自己的理解加以改进。所有代码都是个人参考示例代码独立编写。本实验的设计功能覆盖完全，具有一定的弹性，如果将来需要添加新的指令也可以很方便快捷地添加。**

**本实验加深了我对计算机底层世界的认识，让我明白了许多之前似是而非的问题。例如寄存器的底层工作原理等等。做完这个实验，我甚至有自信说如果让我穿越回计算机发明之前，我就能真正做出一台计算机。**