# lab6 logisim综合实验

姓名:宋玮 学号: PB20151793 实验日期: 2022.5.26

# 实验题目

lab6 logisim综合实验

# 实验平台

Rars, logisim

# 实验结果

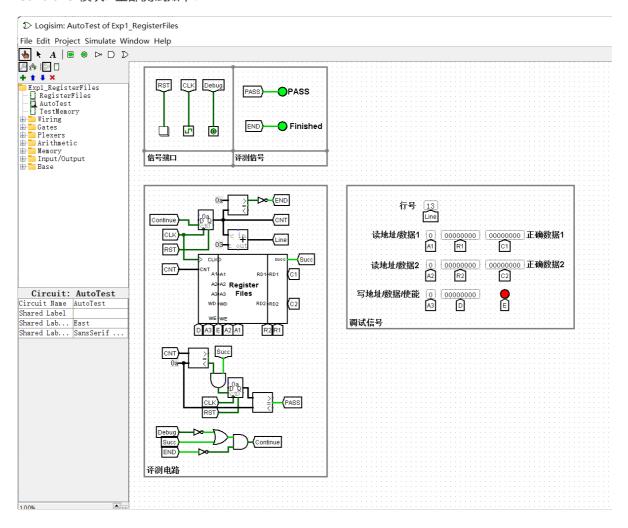
在logisim平台上实现了以下内容:

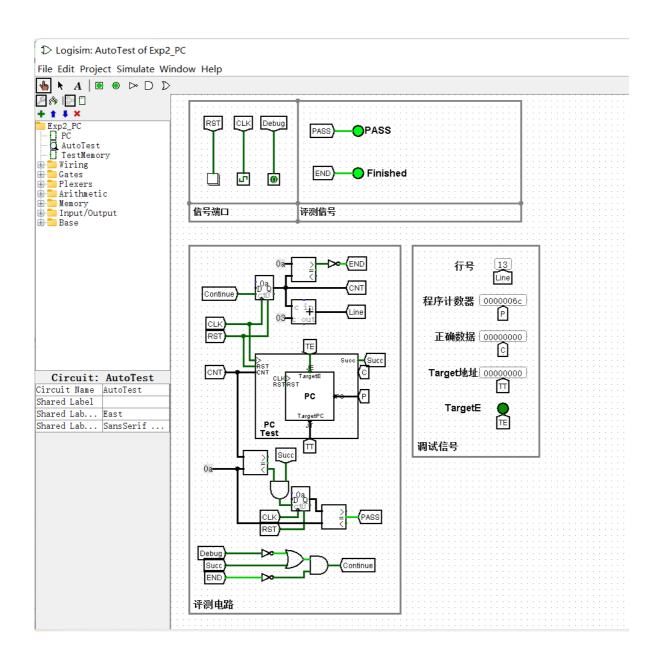
- 1. 单周期CPU
- 2. 流水线CPU
- 3. 能实现ecall指令以及外设输入的流水线CPU

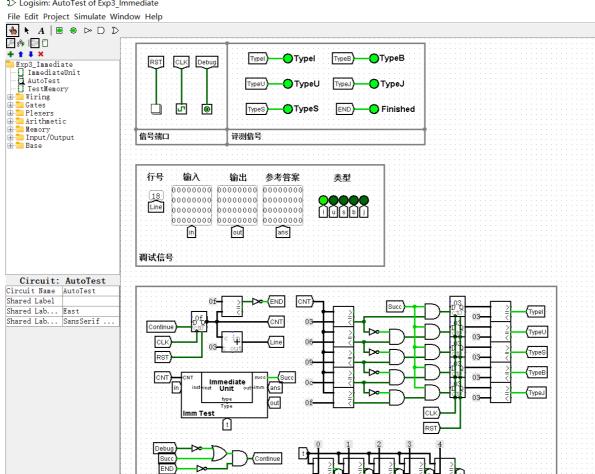
# 实验过程

## 一.基础模块

基础模块包括从Exp1~Exp7的Registerfiles, PC, Immediate, Branch, ALU, Memory, Controller模块. 全部测试如下:





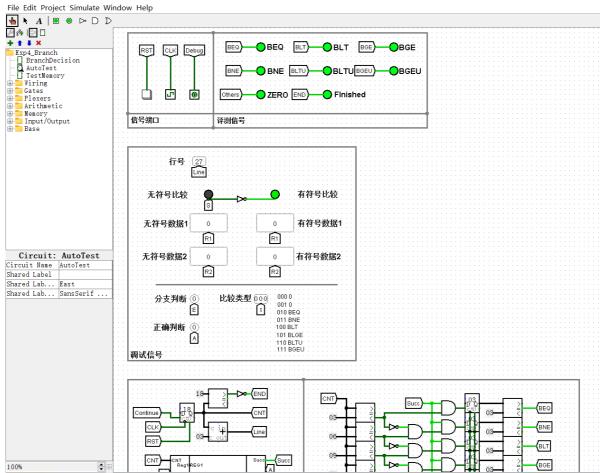


D Logisim: AutoTest of Exp4\_Branch

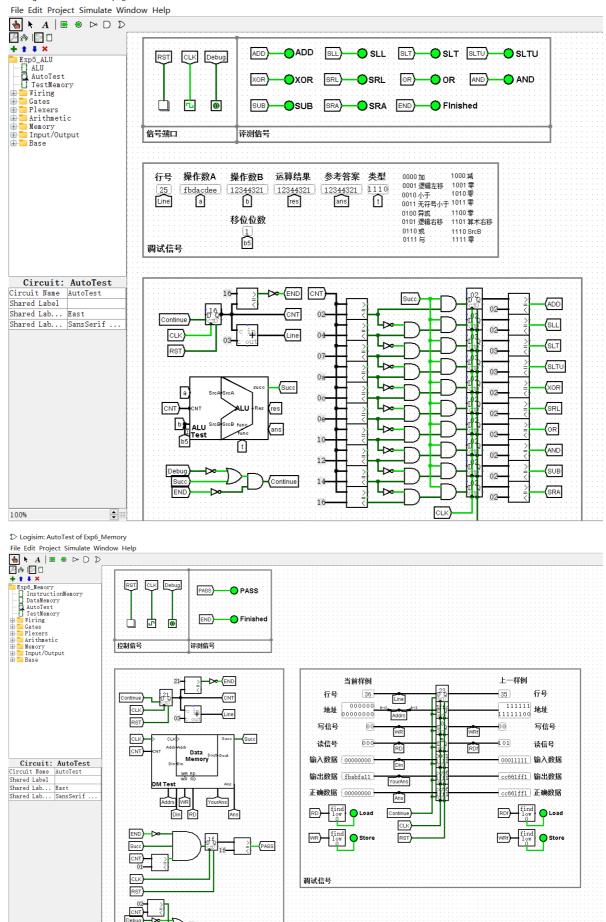
•

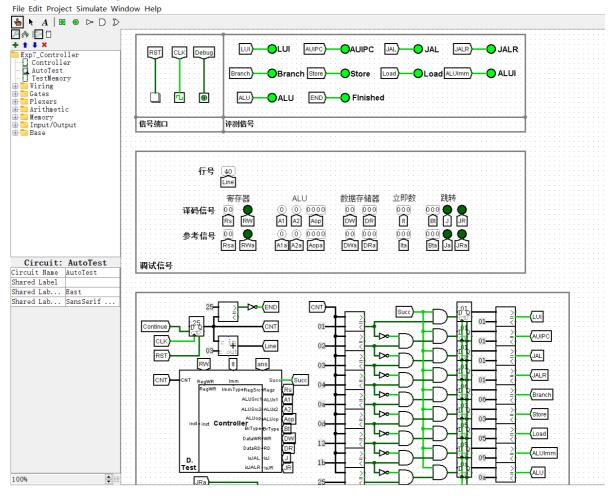
100%

评测电路



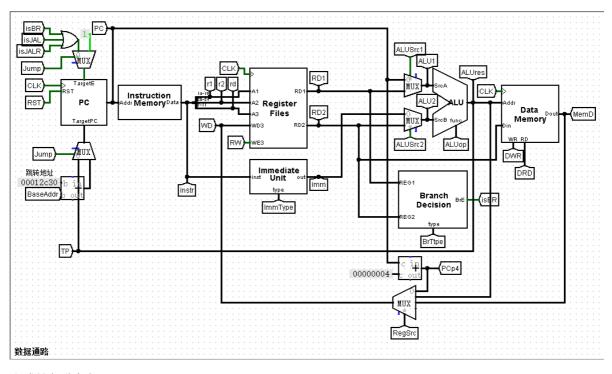
100%





## 二.单周期CPU

#### 完整数据通路如下:



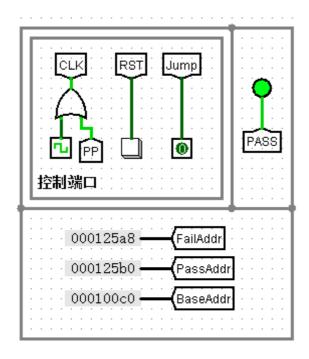
#### 完成的部分包括:

- ALU的输入信号
- 分支判断器的输入信号

- 寄存器文件的写回数据 WD 的选择
- 数据存储器的输入信号

向指令存储器和数据存储器导入相应文件

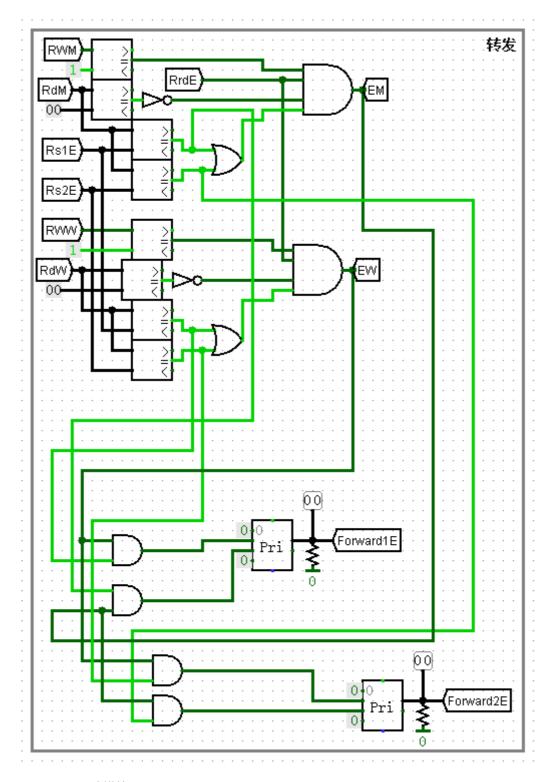
最终测试结果如下:



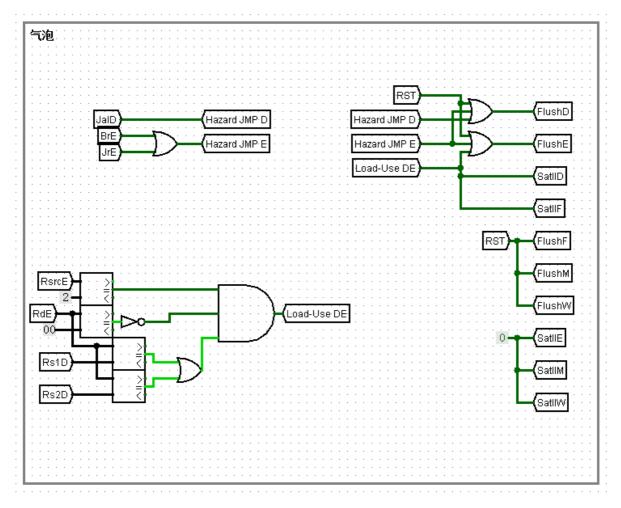
# 三.流水线CPU

主要完成的模块:

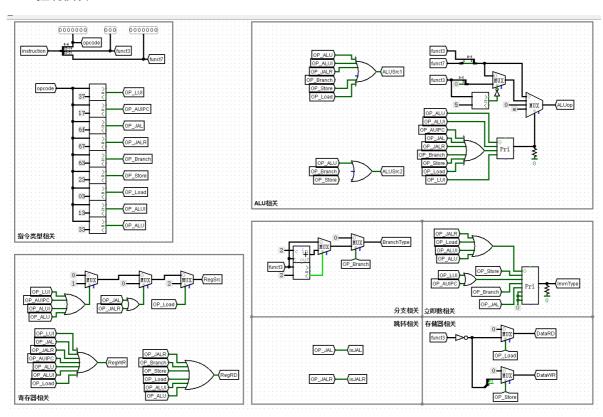
1. 前递模块



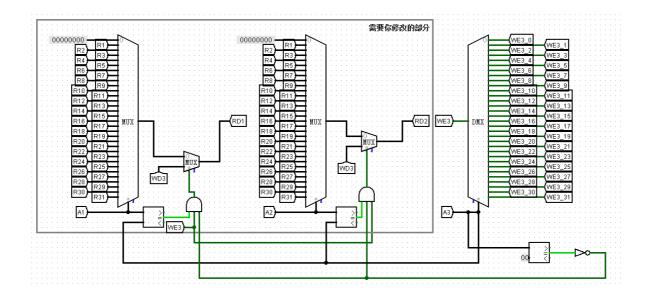
2. load-use和冒险模块



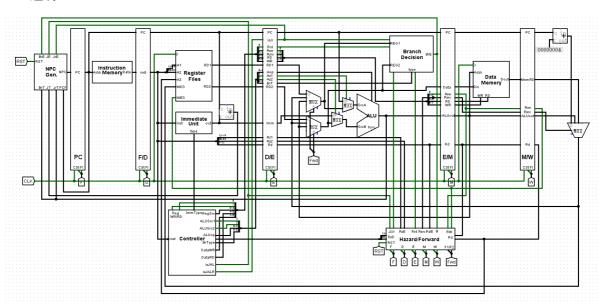
## 3. 控制模块



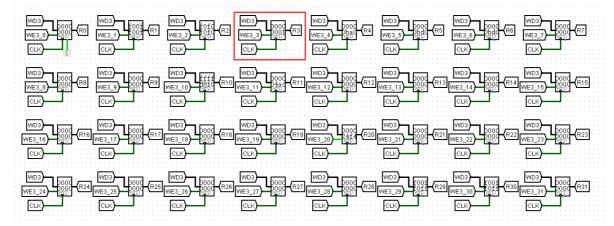
4. 寄存器模块



#### CPU连线

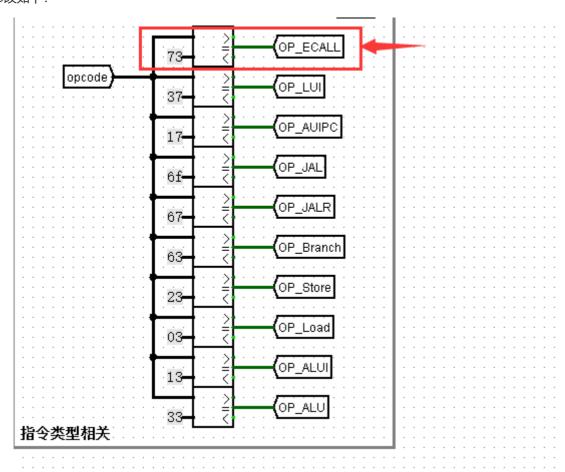


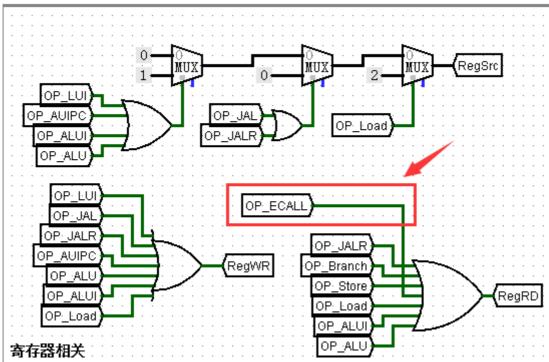
#### 最终运行结果:

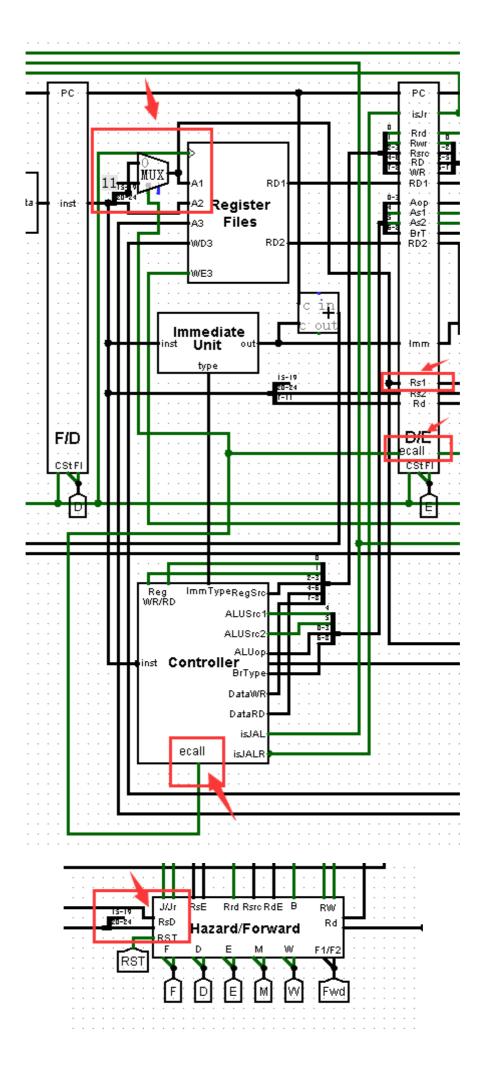


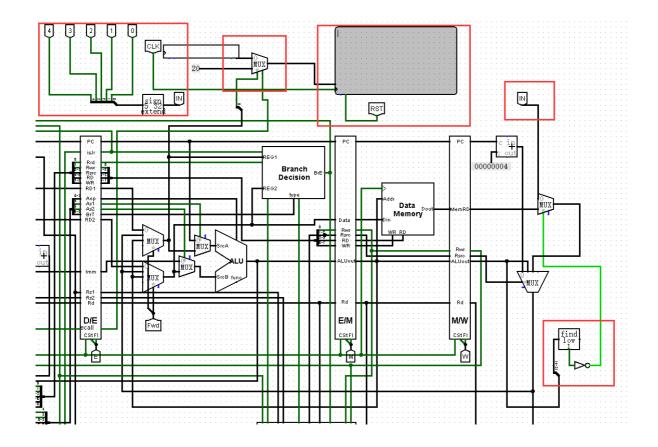
## 四.能实现ecall指令以及外设输入的流水线CPU

在(三)流水线CPU的基础上,通过修改controller模块,前递冒险模块,向D/E寄存器添加ecall信号,修改A1,RS1的输入值,以及增加输入外设来完成一个能实现ecall指令以及外设输入的流水线CPU. 修改如下:









## 说明:

1. ecall指令为00000073

通过寄存器x17的值,ecall指令执行相应功能:

x17 = 0; 输出框中输出键盘 (keyboard) 输入的值;

x17 = 1; 输出框中输出space (空格);

其余情况,输出框中随clk输出?

由于ecall指令需要用到x17的值,因此该指令同样存在前递情况。

2. 在外设输入端,可以通过5个输入口选择要输入的数。

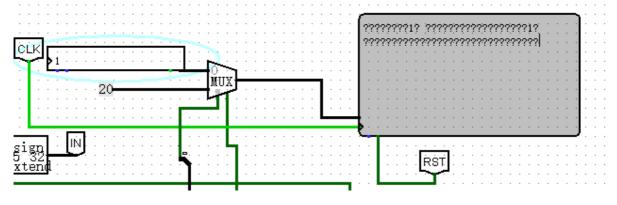
在汇编代码中,该外设的地址为除data\_memory的有效地址外的任意地址。即地址的14~31位含1即可。

#### 3. 测试汇编如下:

```
1 start:
2 #test data hazards
3 addi x1, x0, 1 #x1=1
4 addi x2, x1, 1 #x2=2
5 add x3, x1, x2 \#x3=3
6 add x4, x1, x3 #x4=4
7
   add x5, x1, x4 #x5=5
8 addi x17, x0, 0
9
   eca11
10 addi x17, x0, 1
11
   ecal1
12 add x6, x1, x2 \#x6=3
13 add x6, x6, x3 \#x6=6
```

```
14
    add x6, x6, x4 #x6=10
15
    add x6, x6, x5 \#x6=15
16
    #test load-use hazard
17
    lui x14 ,8
18
    1w x7, 0x4(x14) #x7=in
19
    addi x8, x7, 1 \#x8=in+1
20
    addi x9, x8, -1 #x9=in
21
22
23
    #test control hazard
24 beq x9, x0, start #if (in==0) start
    add x10, x9, x5
25
26
    add x10, x10, x6
27
    stop: jal x0, stop
28
29
    #do not execute
30 add x11, x9, x10
31 add x12, x10, x11
32 add x13, x11, x12
```

## 测试结果:



输入1

