实验 5：单周期处理器的控制器设计实验

一、实验目的

1、 理解随机访问存储器 RAM和只读存储器ROM的操作原理。

2、 理解指令类型与指令格式之间的关系，掌握取指部件、指令解析和立即数扩展器 的设计方法。

3、 理解每条目标指令的功能和数据通路，掌握单周期处理器的控制器设计方法。

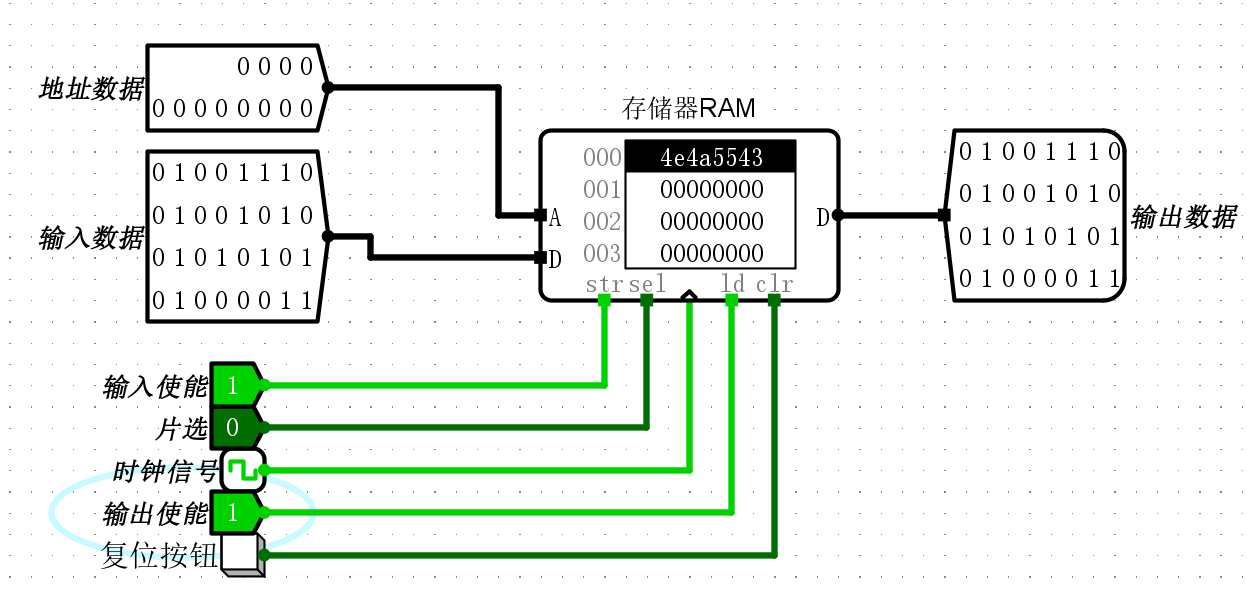
二、实验环境 Logisim-ITA V2.16.1.0。

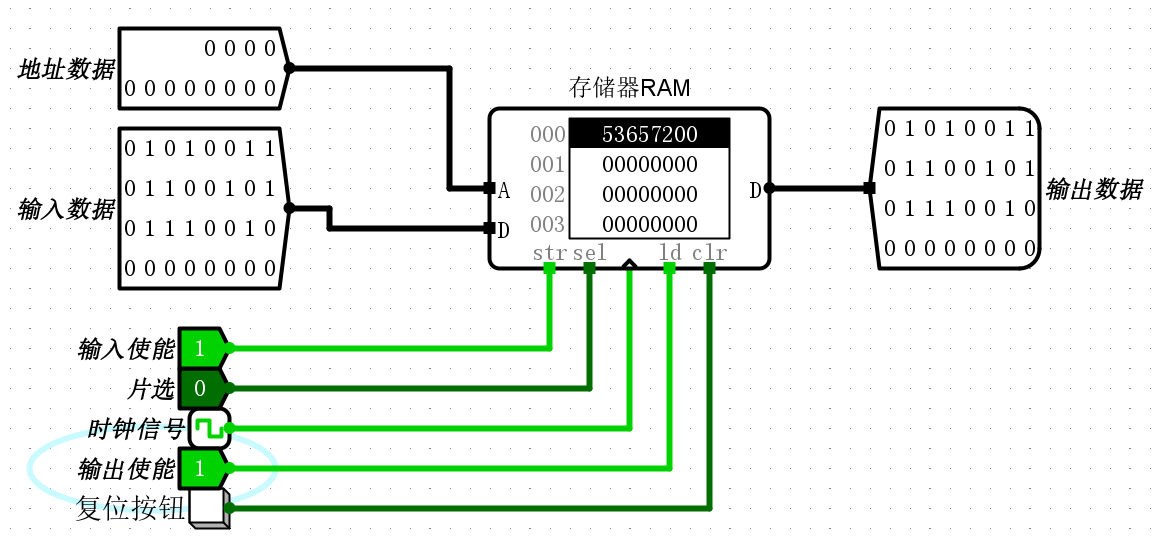
三、实验内容

1.利用 Logisim中的 RAM组件进行数据读写操作实验

实验过程与验证步骤如下：

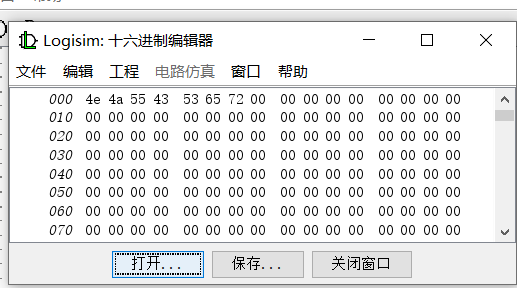
（1）设置数据位宽为 32 位，即可访问空间大小为 16KB；连接必要的输入输出信号 并选择合适的控制信号；从 0 地址处开始顺序写入以下两个 32 位的十六进制数 据：0x4E4A5543、0x53657200；读出所存储的如下：





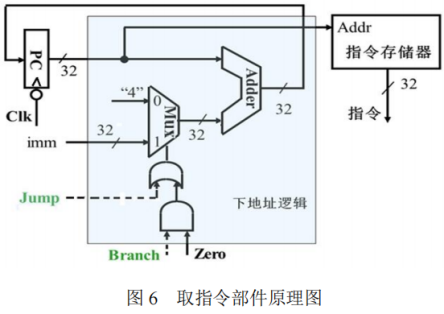
可见，读出数据与输入数据相同。

（2）Logisim中 RAM和 ROM组件的数据输入采用 Logisim十六进制编辑器 和直接读取二进制编码文件的方法实现，打开十六进制编辑器，通过键盘直接将数据输入，然后保存即可。



1. 取指令部件设计实验。

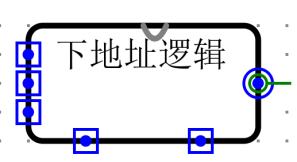
(1)首先根据取指令部件原理图实现取指令部件电路



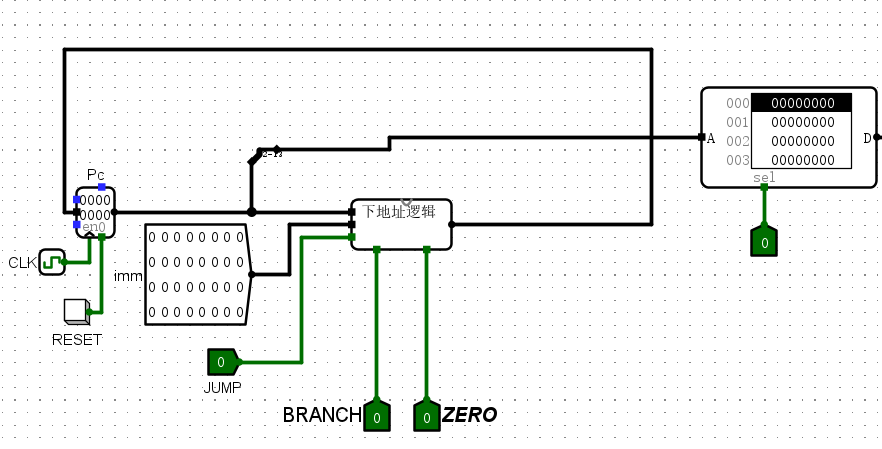
(i)先实现下地址逻辑:

根据原理图连接PC和二路选择器至32位加法器的输入端,将常数0x00000004与imm 连接至二路选择器输入端,将加法器输出连接至PC.添加标识符,得到电路图:

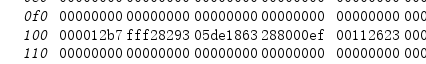
将该电路封装,便于使用.



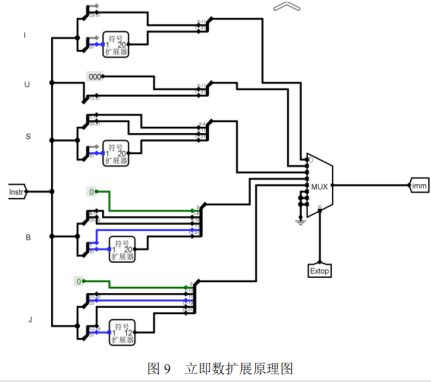
(ii)将封装好的下地址逻辑添加进来,实现取指令部件电路:



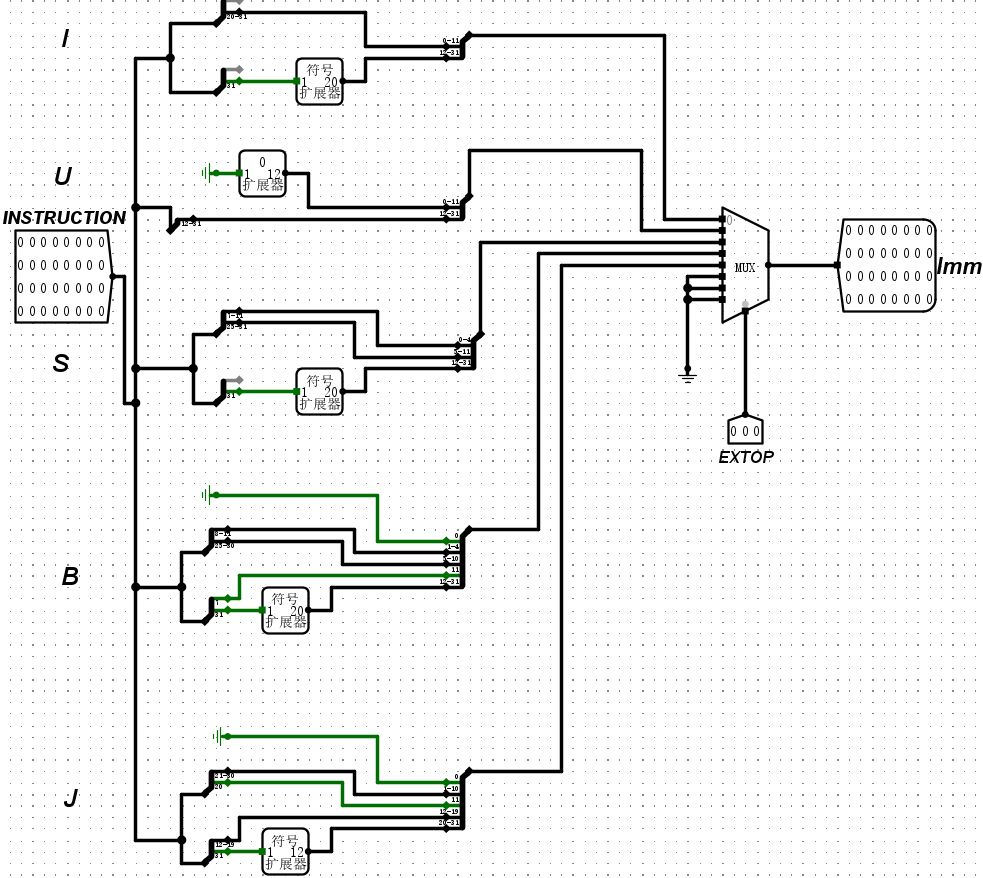
1. 实验验证:
2. 从指令存储器的地址第 100 单元开始，依次写入5 条指令,

.

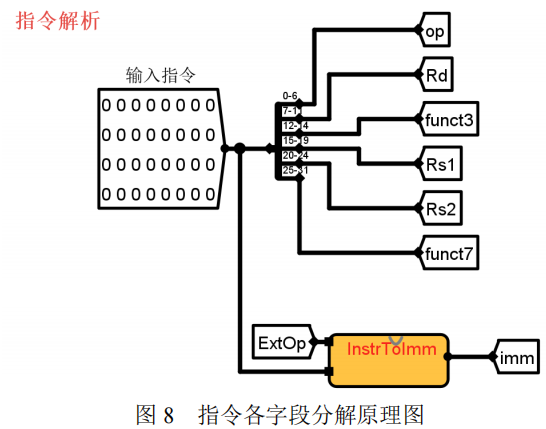
(ii)实现指令各字段分解电路,首先需要实现InstrToImm子电路:

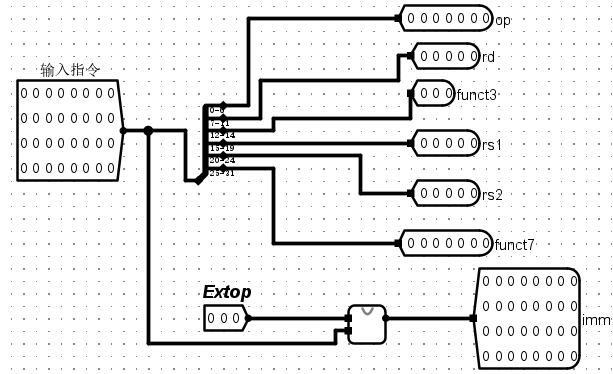


根据电路原理图实现该电路:

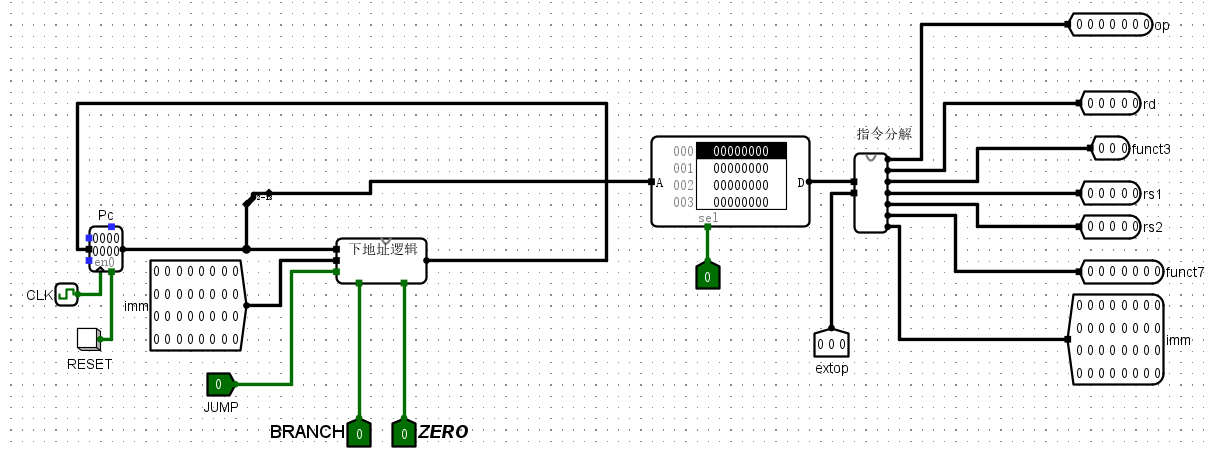


封装好该线路后,添加到指令分解电路中:

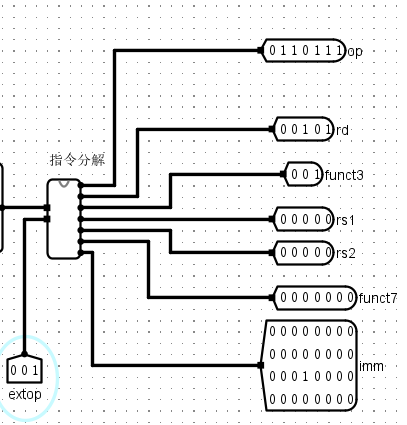




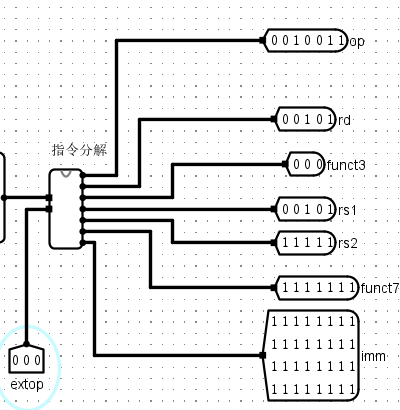
将该电路封装好加入到取指令部件之后,即将ROM输出连接至指令分解的数据输入端, 并加入输入和输出端口.得到电路图:



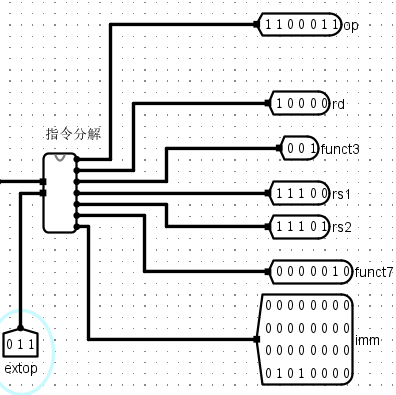
将ROM的地址输入端设置成100单元对应位置,此时右端指令分解部分会显示各字段的 值,通过比对输入与分解后是否相同来验证该电路的功能是否正确.同时将EXTOP设置 成对应的值来得到相应的立即数,结果如下:



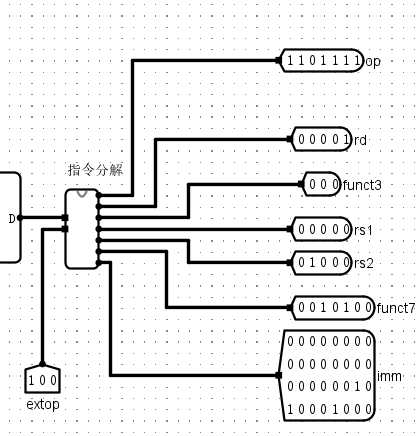
0000 0000 0000 0000 0001 00101 0110111（对应汇编指令 lui x5, 1)



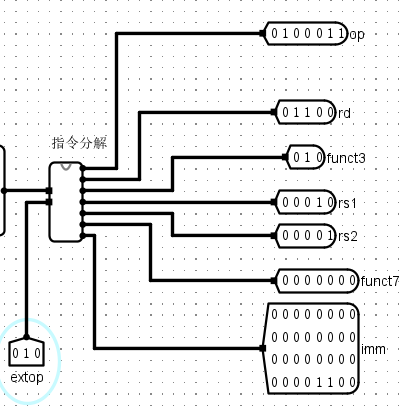
1111 1111 1111 00101 000 00101 0010011（对应汇编指令 addi x5, x5, -1）



0000010 11101 11100 001 10000 1100011（对应汇编指令 bne x28,x29,label1）

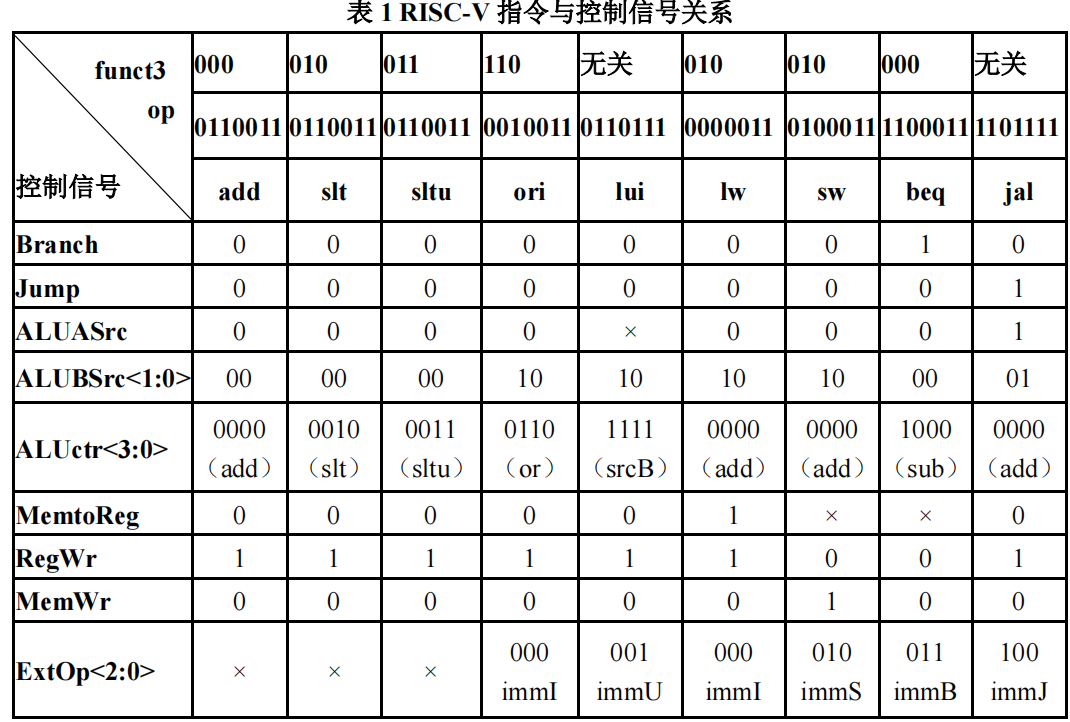


0010 1000 1000 0000 0000 00001 1101111（对应汇编指令 jal ra, printf ）



0000000 00001 00010 010 01100 0100011（对应汇编指令 sw ra,12(sp) ）

1. 设计实现该单周期处理器的控制电路，并验证电路的正确性。
2. 实验原理图:



(2)根据该表分别实现 Branch,Jump,ALUASRC,MemtoReg,ALUctr-<3:0>,RegWr,MemWr,ExtOp<2:0>,ALUBsc

首先,由图可知,

Branch = beq

ALUASRC = Jump = jal,

MemtoReg=~op<6>&~op<5>&~op<4>&~op<3>&~op<2>&~op<1>&~op<0>

RegWr=R型+I型+lui+Load+J型

ALUctr<3>=lui+B型

ALUctr<2>=lui+(R型+I型)&func<2>

ALUctr<1>=lui+(R型+I型)&func<1>

ALUctr<0>=lui+(R型+I型)&func<0>

MemWr=Store;

ALUBsrc<1>=lui+I型&func<1>&func<2>+(Load+Store)&func<1>

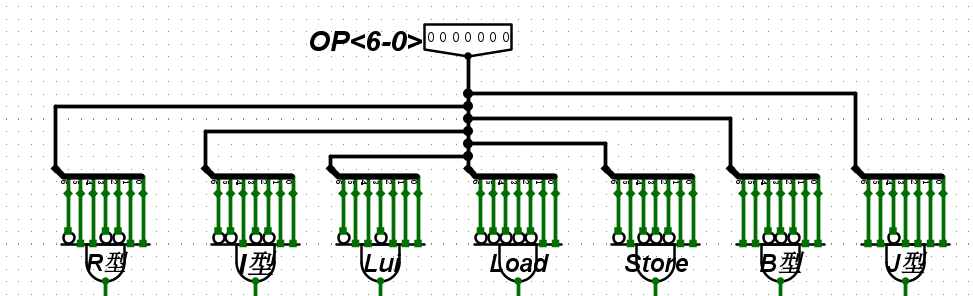
ALUBsrc<0>=jal

ExtOp<2>=jal

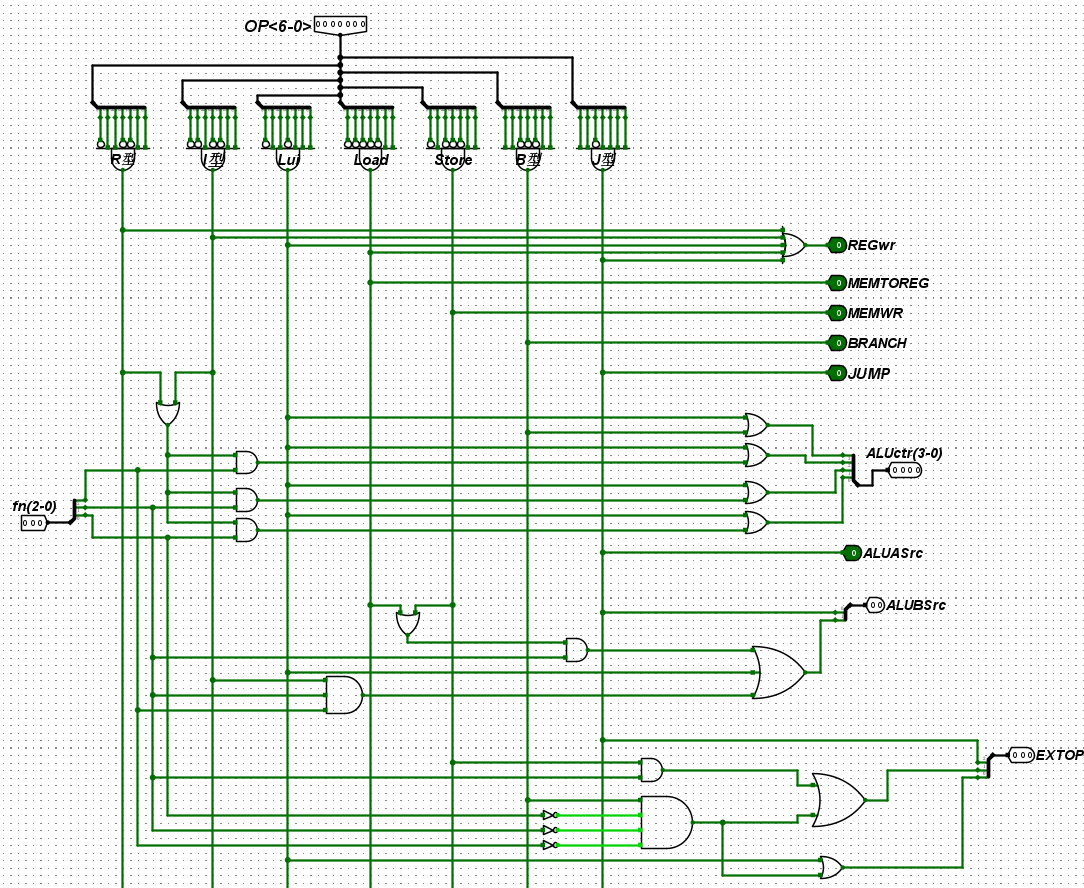
ExtOp<1>=Store&func<1>+Branch(~func<2>&~func<1>&~func<0>)

ExtOp<0>=lui+Branch(~func<2>&~func<1>&~func<0>)

首先实现R型,I型,Lui,Load,Store,B型,J型的控制电路:



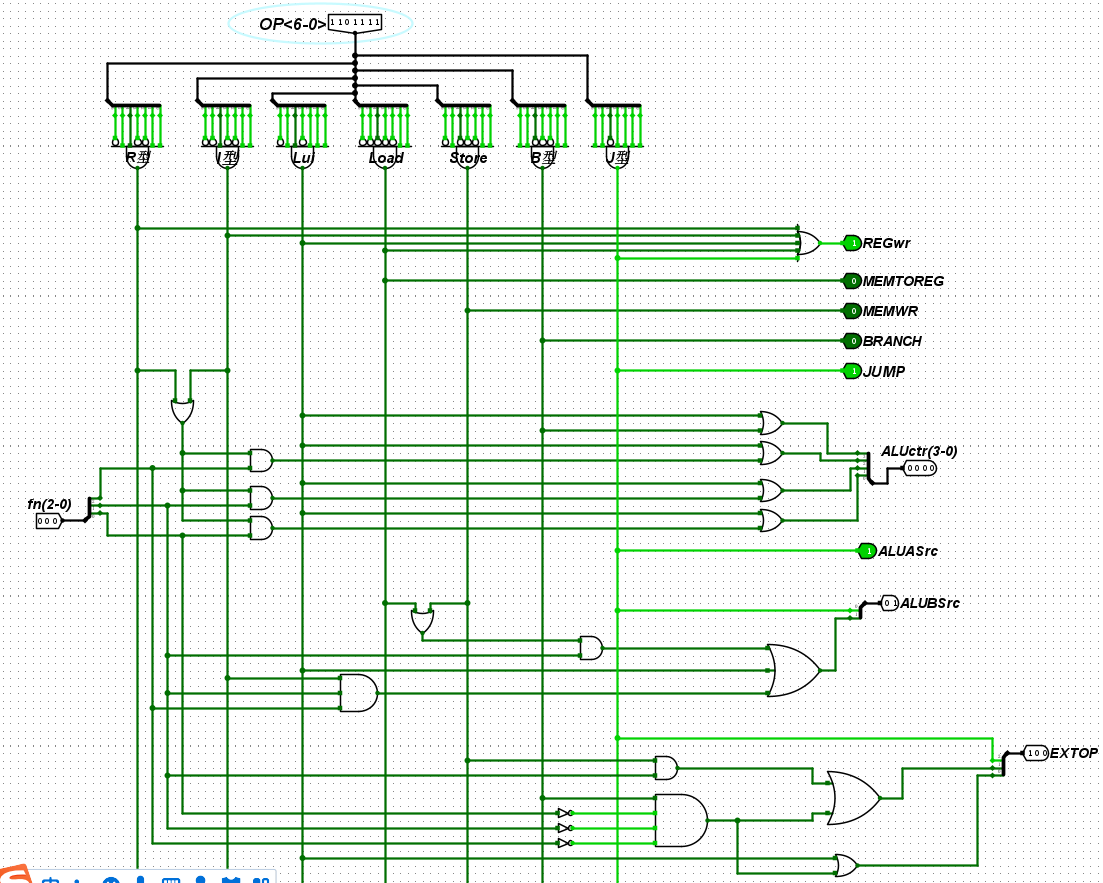
然后,接入func<3:0>,实现完整控制部分电路,添加相应的标识符:

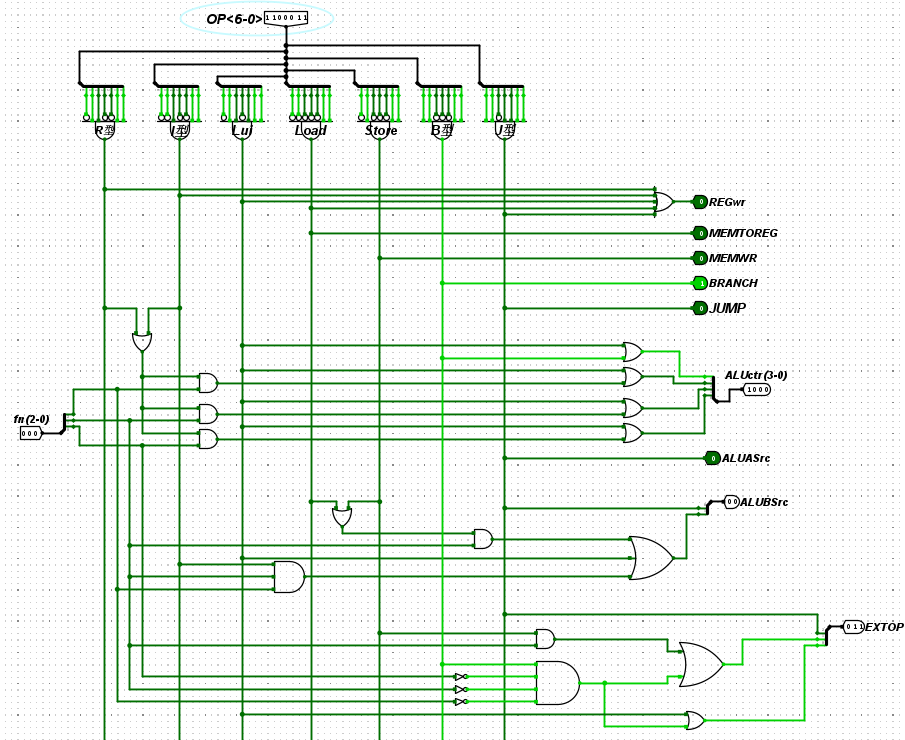


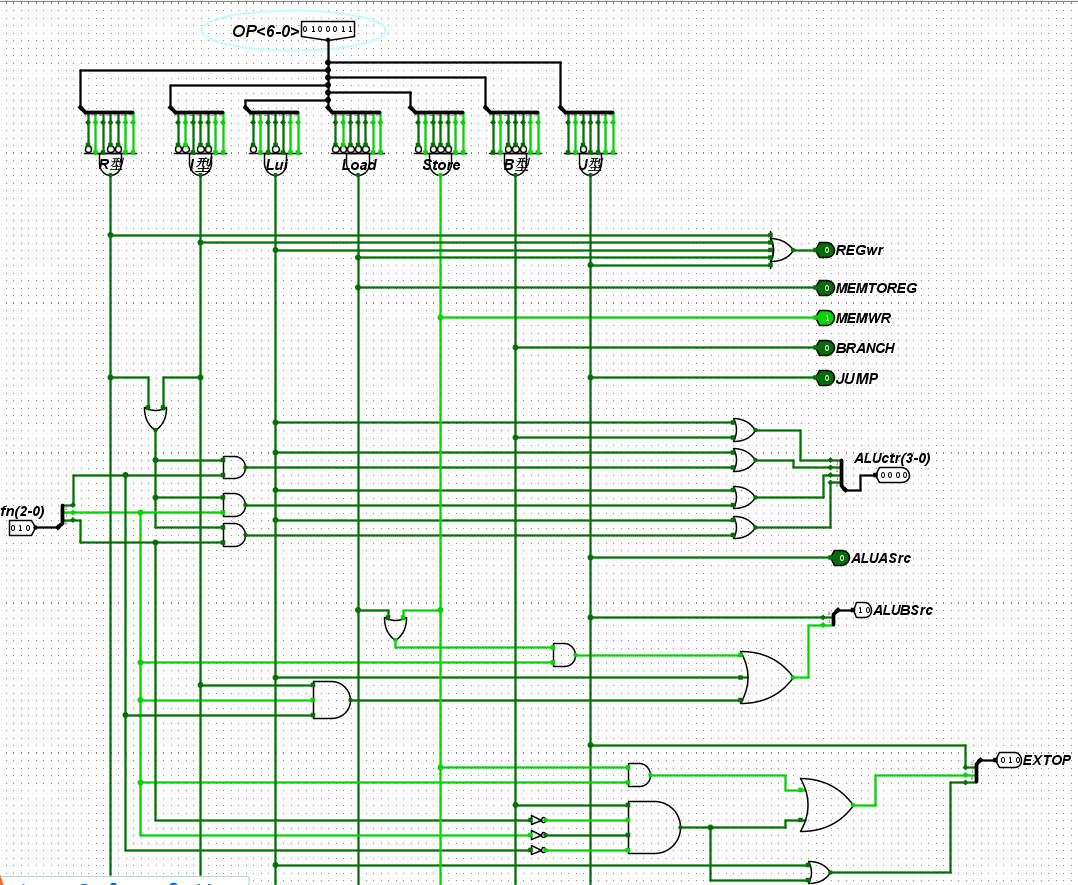
(3)验证功能:

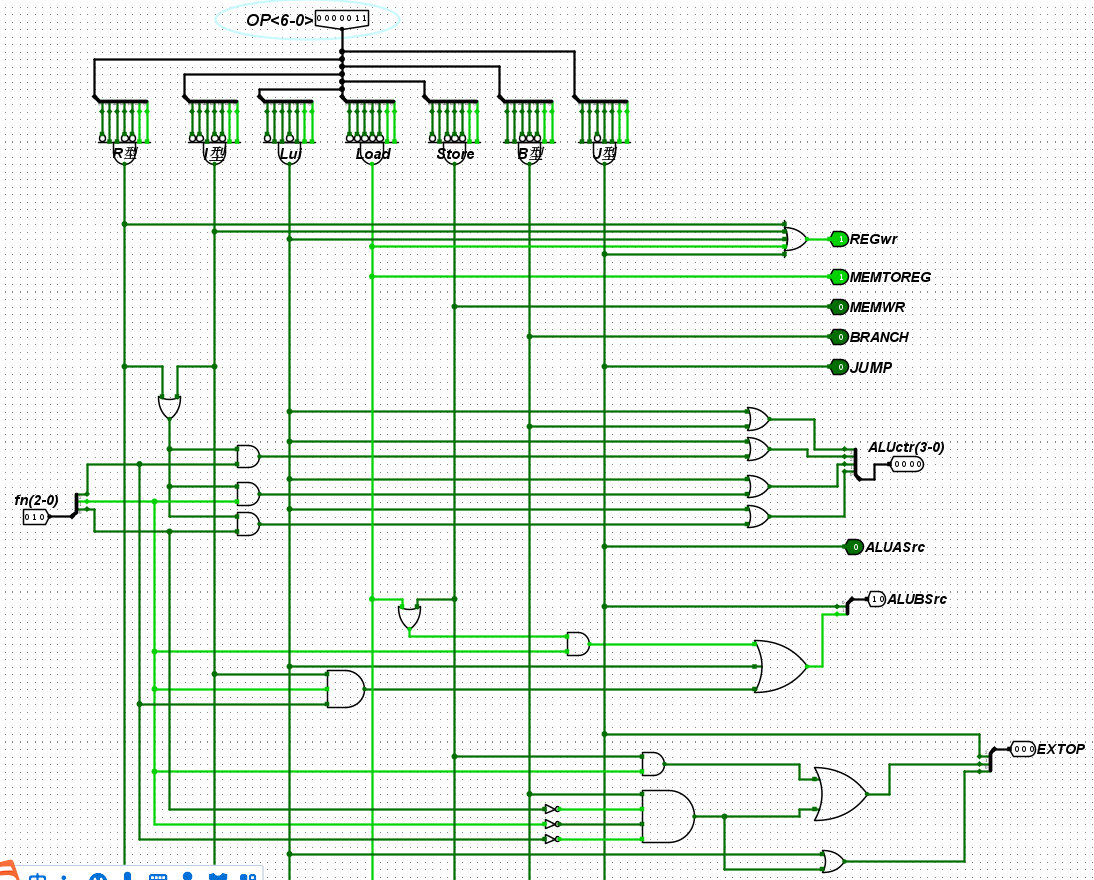
根据上表,分别输入op和func的指定编码,验证各个控制信号是否正确:

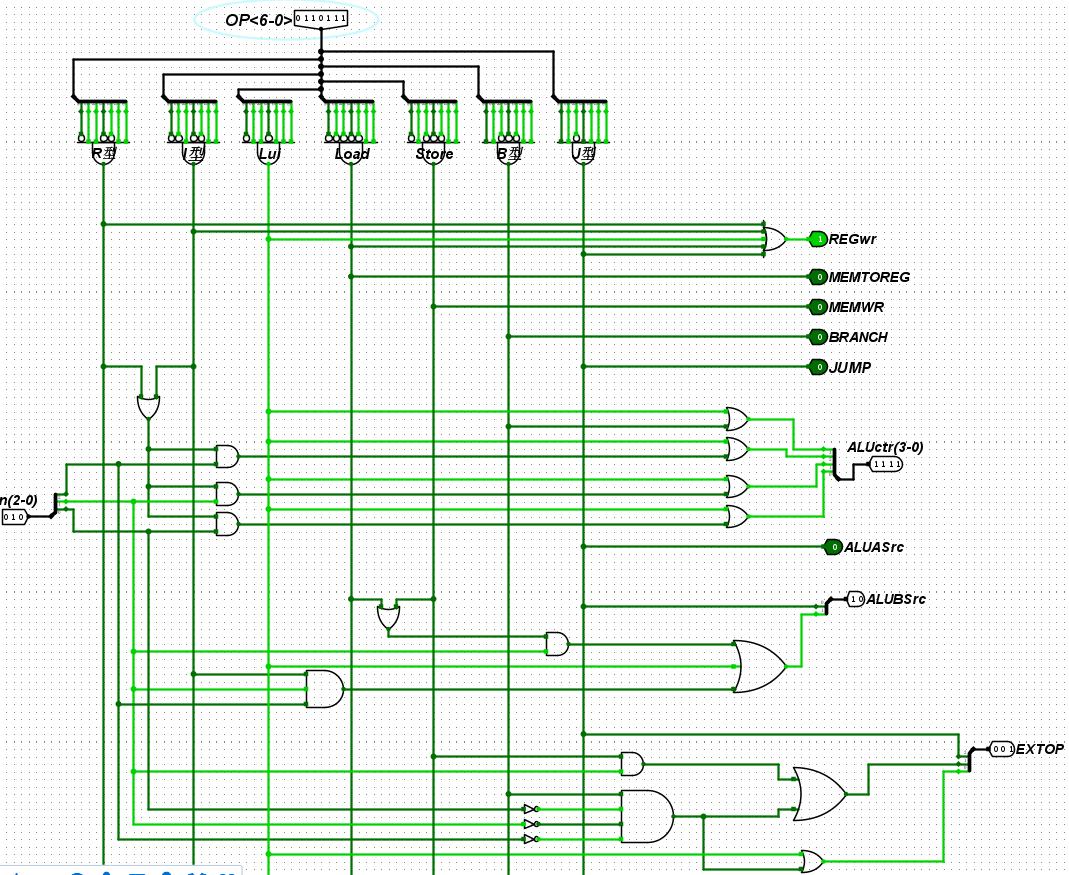
按照从右向左的顺序排列如下:

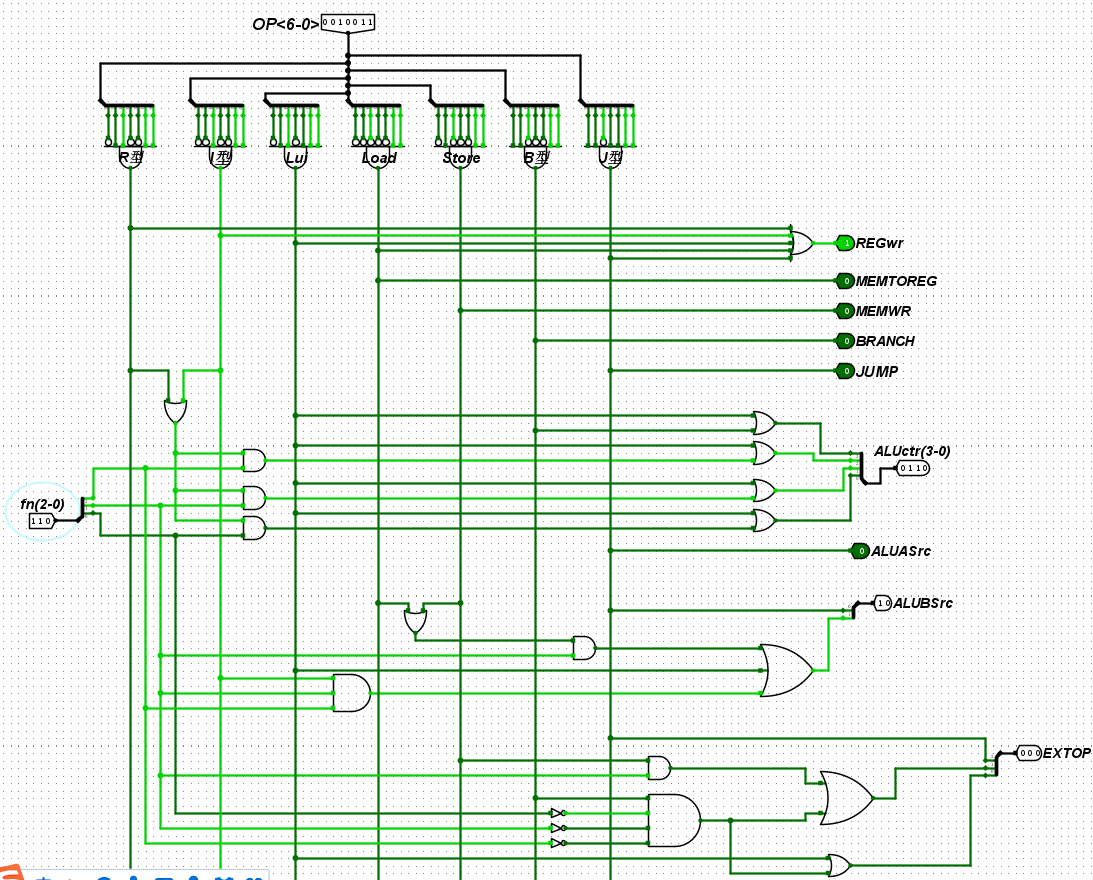


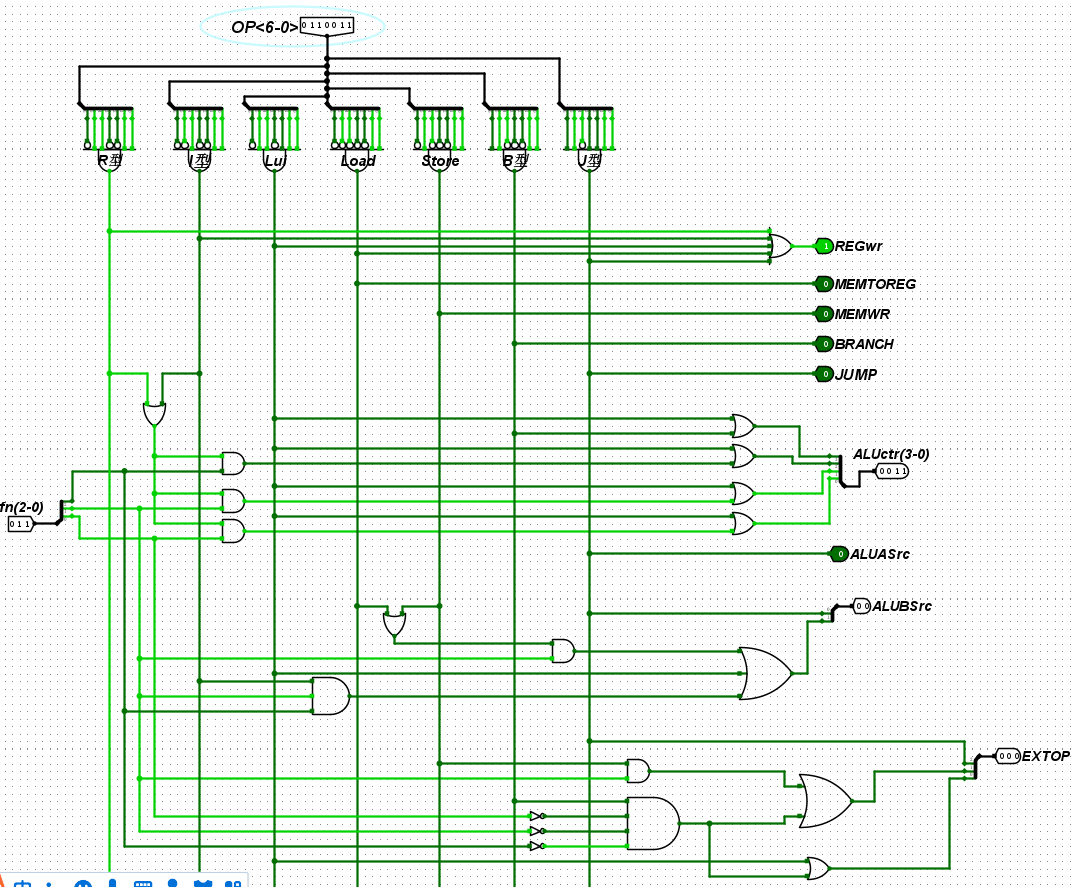


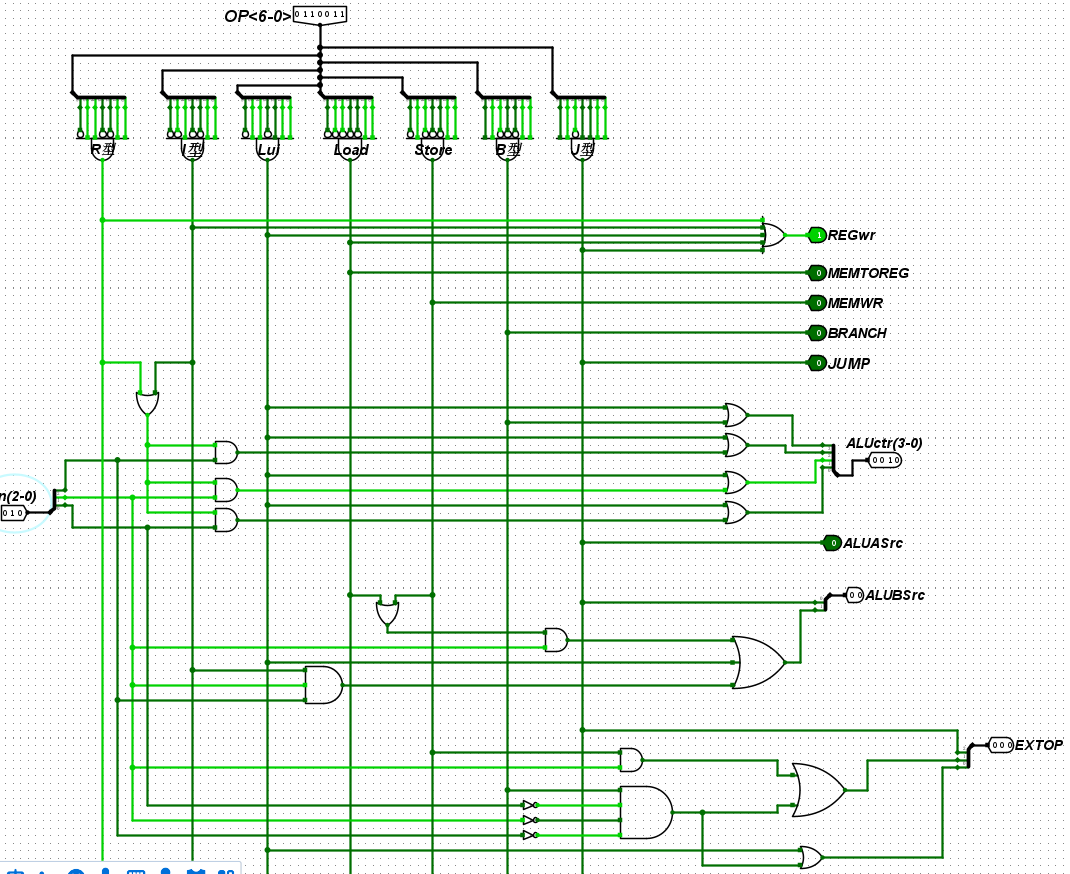


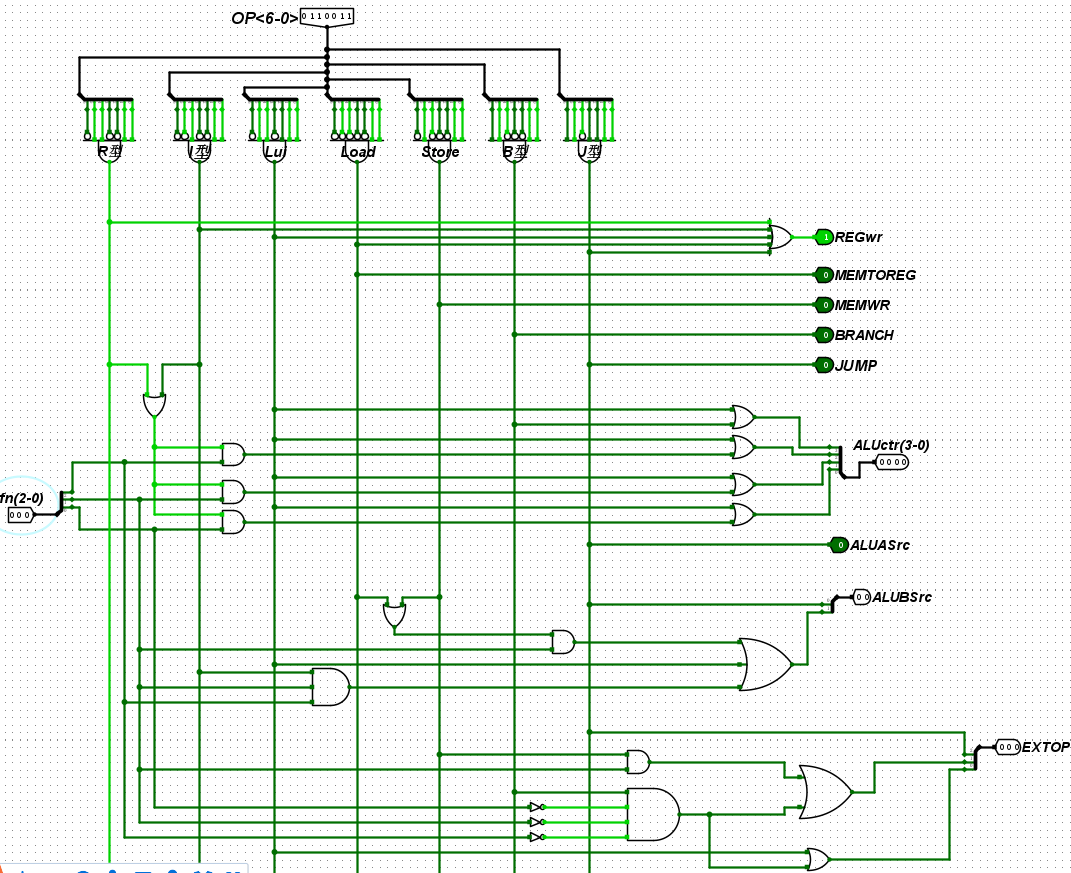












思考题:

1:图 10 RISC-V 单周期数据路通所示的数据通路是否可以执行 auipc 指令？增加 auipc 指令后，相应的控制器电路需要进行哪些修改？

答:不可以,因为auipc指令对应的op字段编码为0010111以上电路并没有该编码,故不 能.

增加 auipc 指令,首先需要添加对应的op编码0010111.

EXTOP=ImmU=1,MemWr=0,RegWr=1,MemtoReg=0,ALUctr=0000,ALUBsrc=10;

ALUAsrc=1;jump=branch=0;

其中jump,branch,MemtoReg无需修改;

ALUAsrc=jal+auipc;

RegWr=R型+I型+lui+Load+J型+aupic;

ALUBsrc<1>=lui+I型&func<1>&func<2>+(Load+Store)&func<1>+aupic;

ALUBsrc<0>无需修改;

ALUctr<3:0>无需修改;

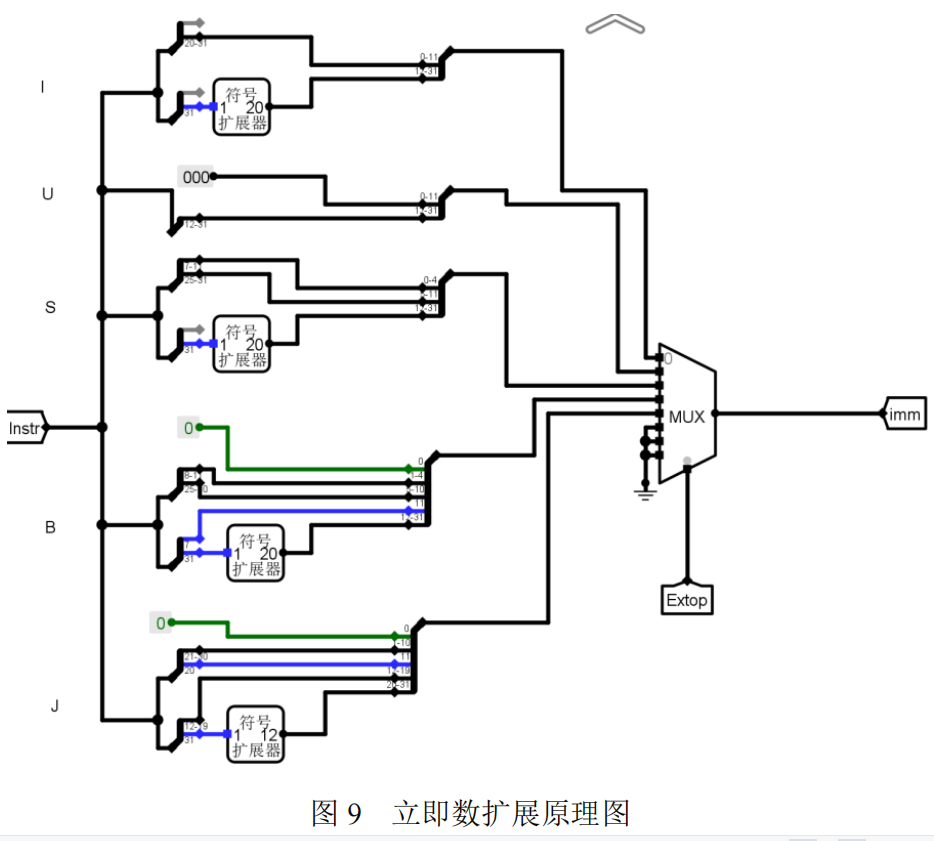
ExtOp<2>,ExtOp<1>无需修改;

ExtOp<0>=lui+aupic+Branch(~func<2>&~func<1>&~func<0>)

2.如果不采用如图 10 所示的统一扩展器进行 5 种立即数扩展，而是分别用 5 个扩展器

进行立即数扩展，则图 10 所示数据通路应该如何修改？

答:将立即数扩展原理图中的五个立即数扩展分开.



根据原理图可知,I型立即数和S型立即数都要对高20位进行符号扩展之后与Rs1的内容相加,把极速那结果送到目的寄存器Rd或得到访存地址,他们都不需要改变PC的值,区别在于低12位的来源不同,通过分线器选择不同的低12位之后在进行符号扩展即可.

对于B型立即数和J型立即数,他们都需要改变PC得值,但都不要输入ALU进行运算,因此可以将这两者归为一类.通过控制器生成的控制信号来进行选择要哪个立即数.原理图如下:

