目录

1． 只读存储器实验

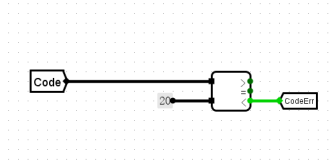
1.1 需求

利用 ASCII 码可显示字符点阵字库，在 Logisim 的 LED 点阵组件上显示 ASCII 码字符形状。

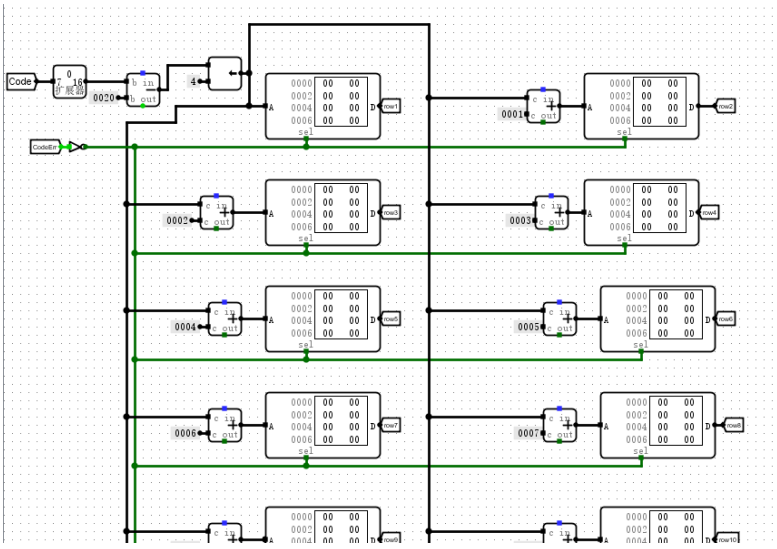
1.2 电路实现

CodeErr 的实现

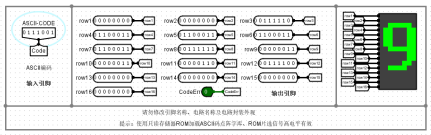
通过比较 Code 是否在正确的范围内生成 CodeErr 标志位

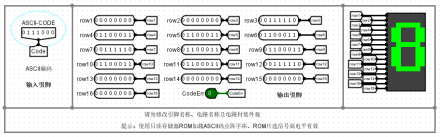


电路实现



1.3 仿真测试





经测试，电路运行正确。

2． 数据存储器实验

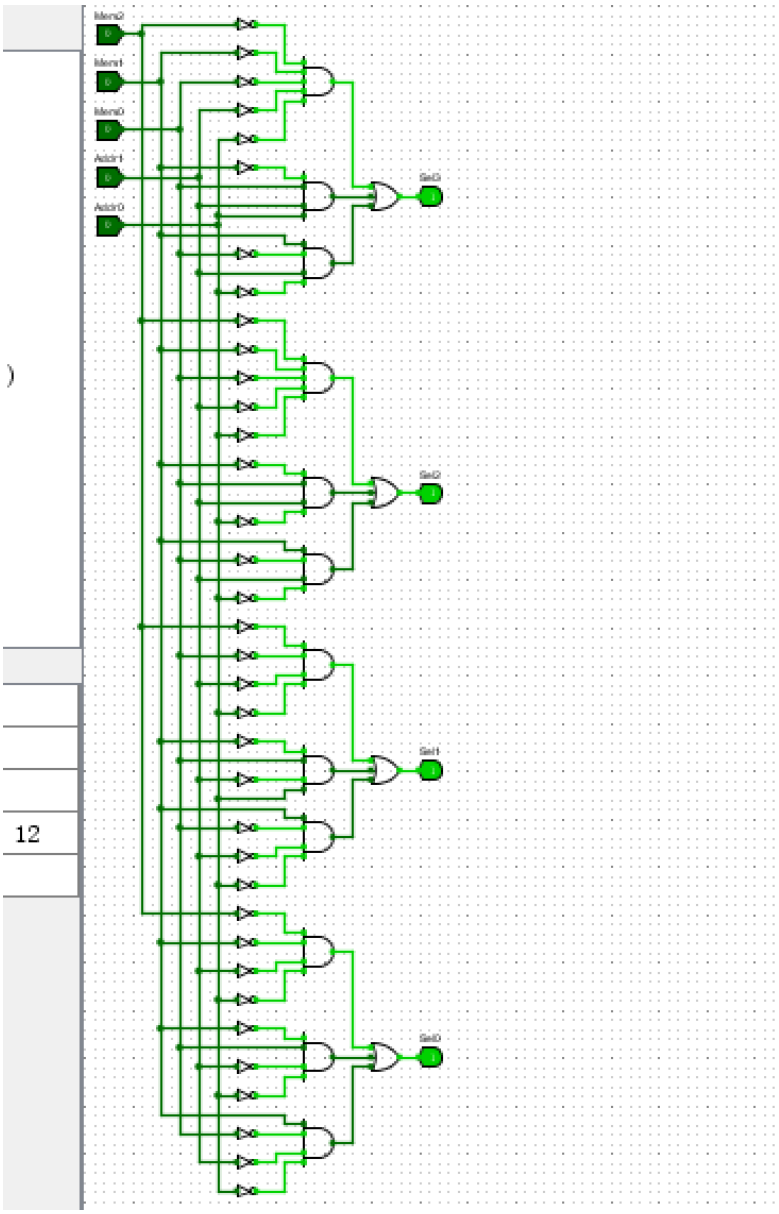
2.1 需求

实验要求设计一个 256KB 的数据存储器，可按照字节进行存取操作，数据字长为 32 位，则地址位为 18 位。

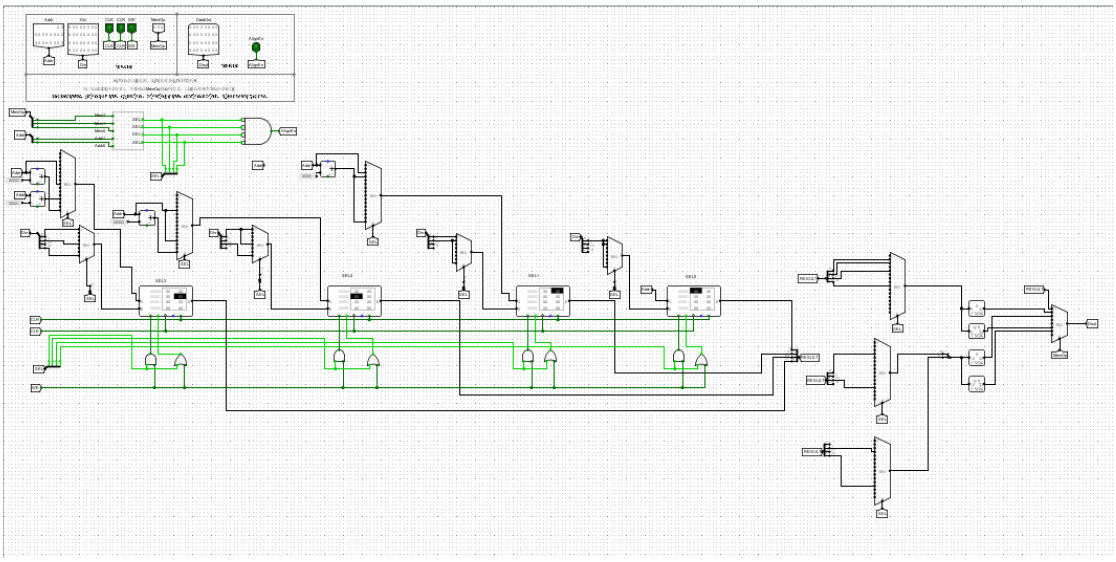
2.2 电路实现

SEL

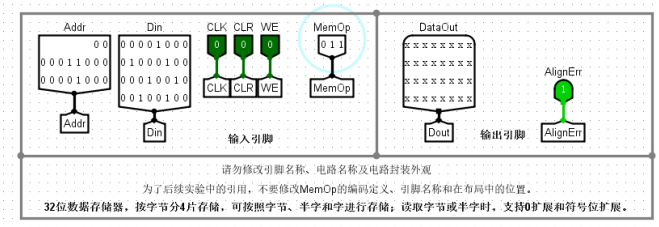
通过真值表生成 SEL 的电路。

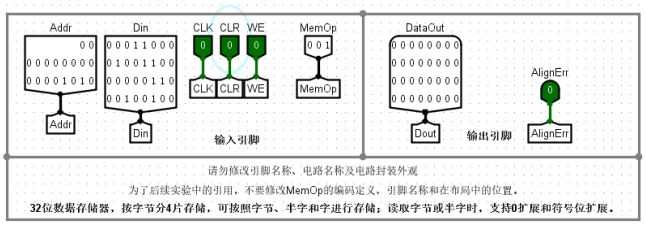


完整电路实现



2.3 仿真测试





经测试，功能可以正常运行，也可以通过 OJ。

3． 取指令部件实验

3.1 需求

实验要求根据初始地址 Initial Address、立即数寄存器 Imm 和 rs1 寄存器的数据 BusA，以及控制信号 Reset、NxtASrc 和 NxtBSrc 的赋值输出当前指令的地址寄存器 PC 和指令存储器的输出内容 IR。

在程序执行过程中，下一条指令地址的计算有多种情形：1、顺序执行指令，则 PC=PC+4。2、无条件跳转指令，jal 指令，PC = PC + 立即数 imm；jalr 指令，PC = R[rs1] + 立即数 imm。3、分支转移指令，根据比较运算的结果和 Zero 标志位来判断，如果条件成立则 PC = PC + 立即数 imm，否则 PC=PC+4。

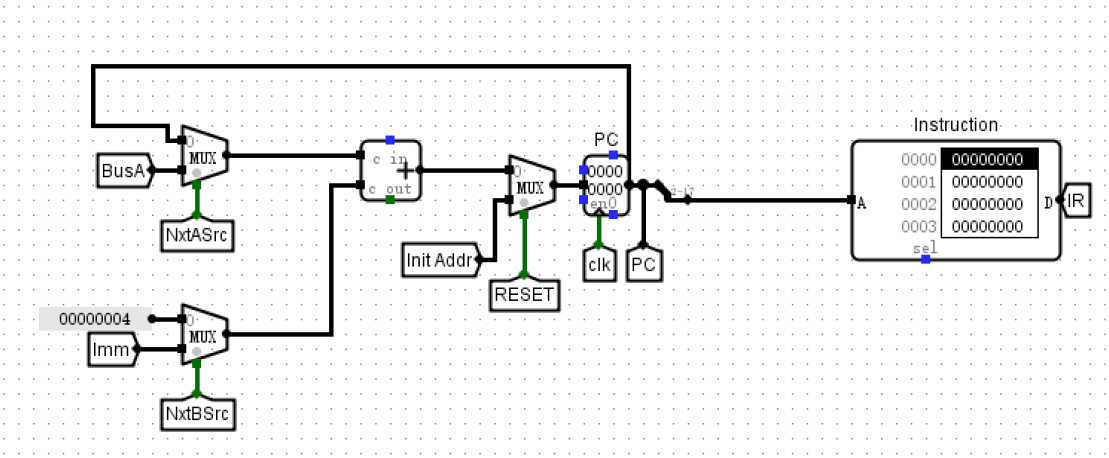
NxtASrc 和 NxtBSrc 赋值定义表

表 5.3 NxtASrc 和 NxtBSrc 赋值定义表

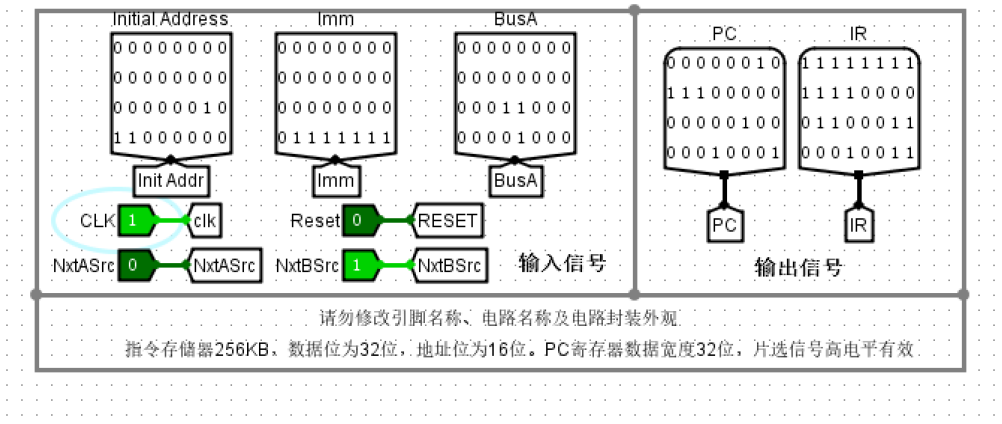
指令类型	NxtASrc	NxtBSrc	下条指令地址
顺序指令	0	0	PC=PC+4
无条件跳转 jal	0	1	PC=PC+Imm
无条件跳转 jalr	1	1	PC= R[rs1]+Imm
分支跳转指令	0	1	条件成立 PC=PC+Imm，否则 PC=PC+4

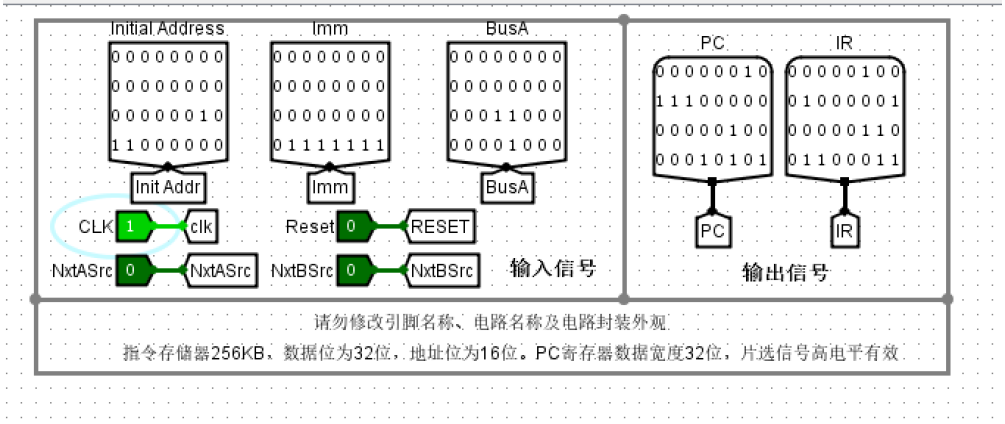
当 NxtASrc=0 时，选择 PC 寄存器的值，否则选择 Rs1 寄存器值 BusA。当 NxtBSrc =0 时选择常量 4，否则选择立即数 Imm。

3.2 电路实现



3.3 仿真测试



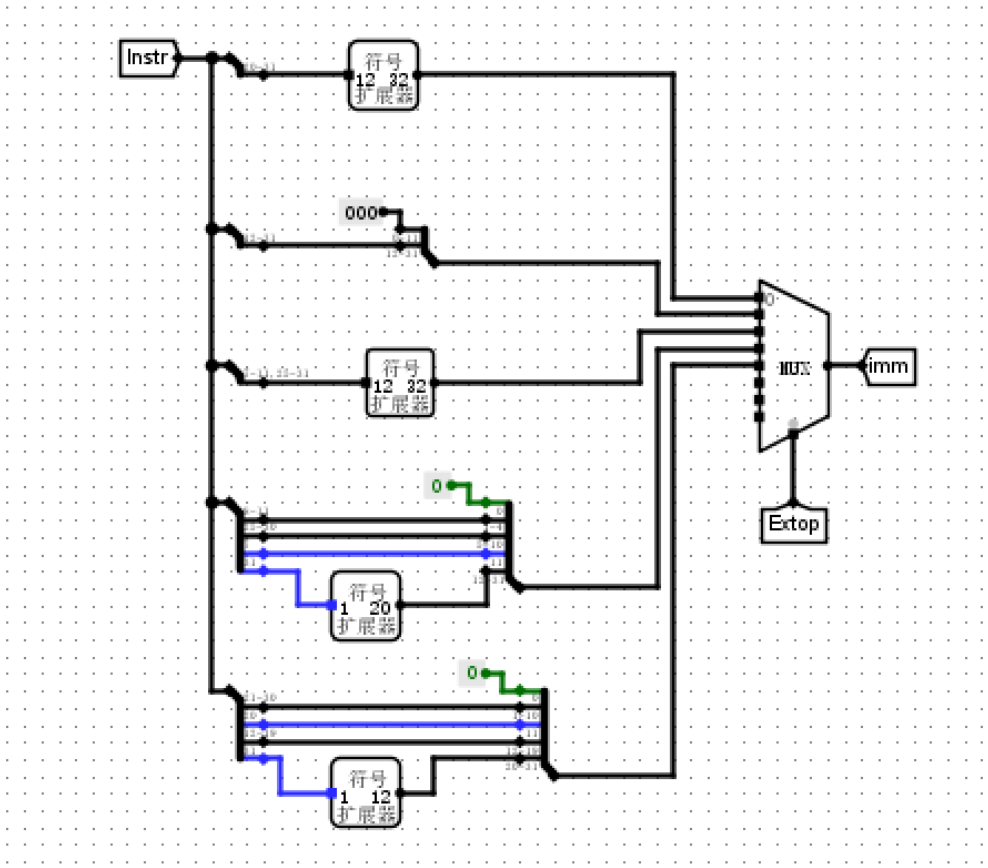


经测试，可以正确运行。

4． 取操作数部件 IDU 实验

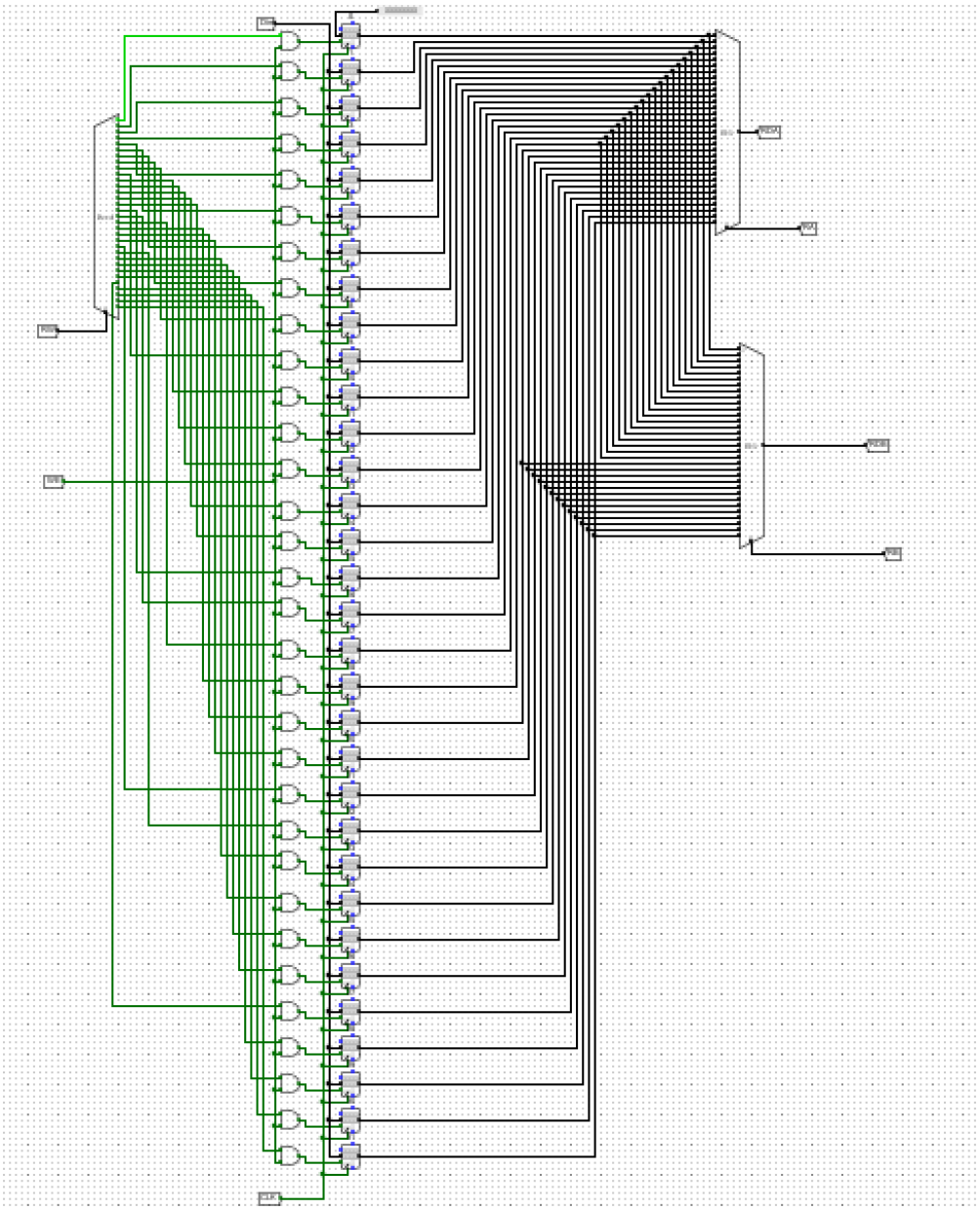
4.1 电路实现

立即数扩展器

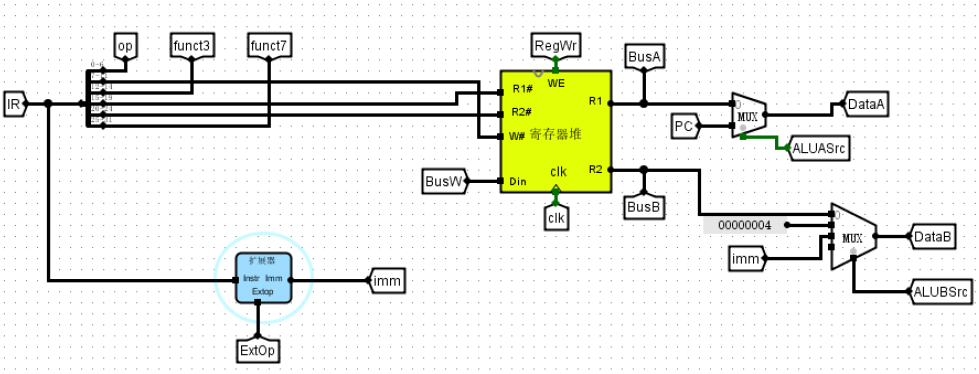


寄存器堆

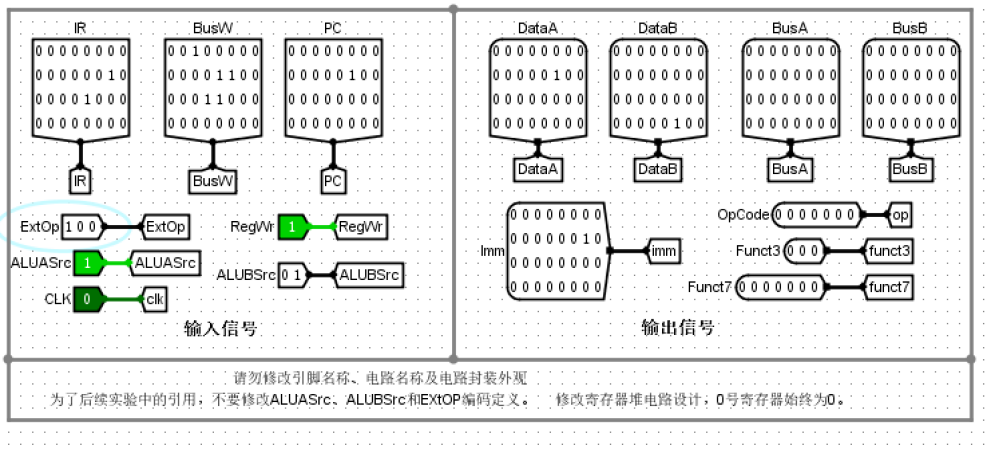
修改寄存器堆，使 0 号寄存器硬性为全 0.



电路实现



4.2 仿真测试



经测试，可以正确运行。

5． 数据通路实验

5.1 介绍与需求

数据通路是具体完成数据存取、运算的部件。单周期 CPU 的数据通道是指获取到指令之后，根据指令内容，读取操作数，进行操作，得到结果并写回的过程。

原理图

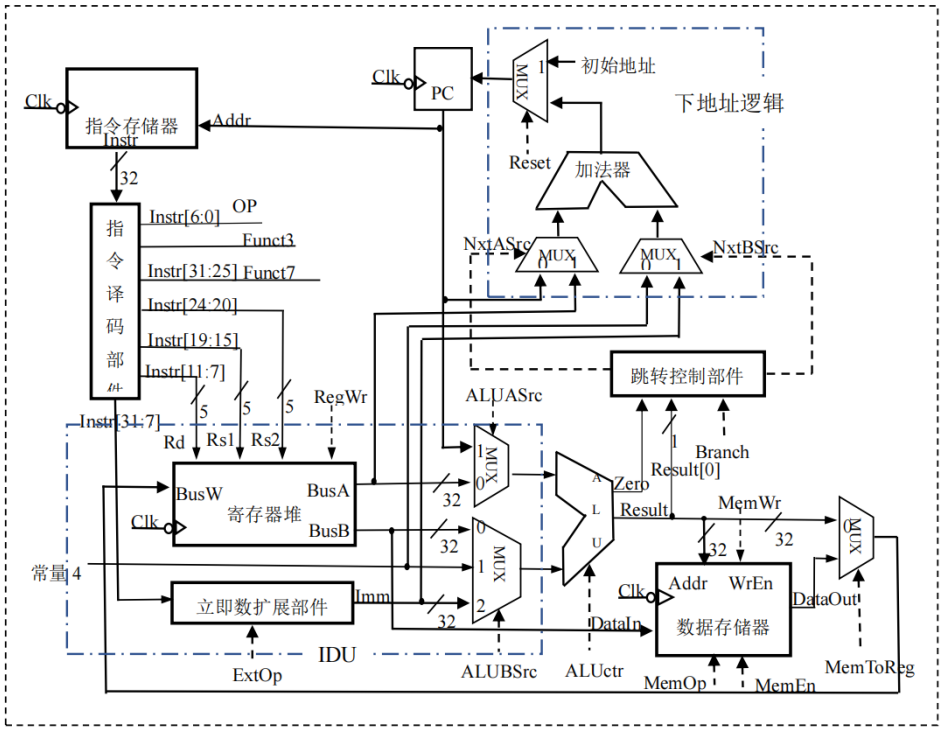
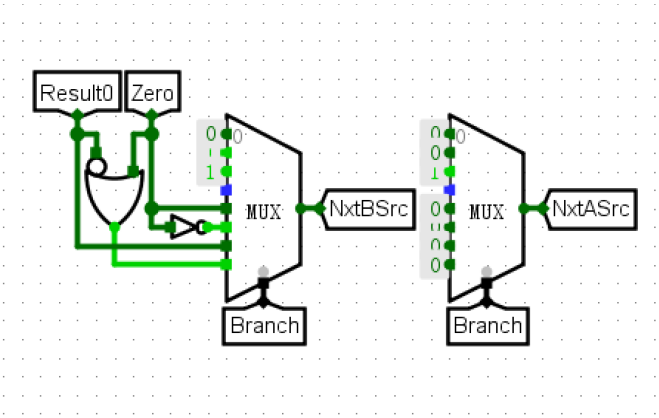


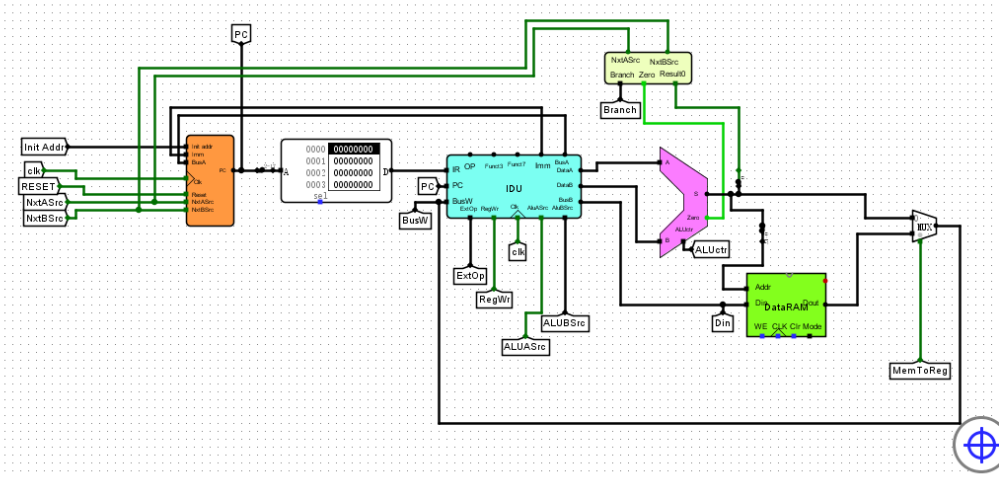
图 5.7 单周期 CPU 数据通路原理图

5.2 电路实现

Branch 的实现



电路的实现



5.3 测试

序号	指令 PC=0	RV32 汇编语句	控制信号赋值，未列出的控制信号值为 0	输出信号值
1	008000ef		ExtOp=4，RegWr=1，ALUBSrc=1，Branch=1	x1：4
2	ffdff06f		ExtOp=4，RegWr=1，ALUBSrc=1，Branch=1	x0：0
3	00100013		RegWr=1，ALUBSrc=2	x0：0
4	deadc2b7		ExtOp=1，RegWr=1，ALUBSrc=2，ALUCtr=15	x5：deadc000
5	eef28293		RegWr=1，ALUBSrc=2	x5：deadbeef
6	0052d333		RegWr=1，ALUCtr=5	x6：0001bd5b
7	00135313		RegWr=1，ALUBSrc=2， ALUCtr=5	x6：0000dead
8	4052d3b3		RegWr=1，ALUCtr=13	x7：ffffbd5b
9	00529e33		RegWr=1，ALUctr=1	x28：df778000

填表，符合预期。

6． 思考题

6.1 第一题

已线下验收。

6.2 第二题

在上升沿，PC 寄存器更新，程序计数器增加或跳转到新地址，同时数据存储器写入数据。在下降沿，寄存器堆写入指令结果或其他数据，准备下一条指令的执行。可以有效利用时钟周期的两个阶段，确保数据和指令的同步进行，减少数据一致性问题和竞争，从而提高程序执行的效率和可靠性，但是对时序分析的要求提高了，因为需要保证在下降沿到来前，前置操作都已经完成。

6.3 第三题

在程序结束时执行类 Halt 指令，使 CPU 将进入休眠状态。