**实验四实验报告**

1. **一、实验目的**

**1、 掌握快速加法器 CLA 和先行进位逻辑 CLU 的设计方法。**

**2、 掌握 32 位先行进位加法器及相关标志位的实现方法。**

**3、 掌握 ALU 的设计方法，根据指令要求实现 6 种操作的 ALU器件。**

1. **实验环境**

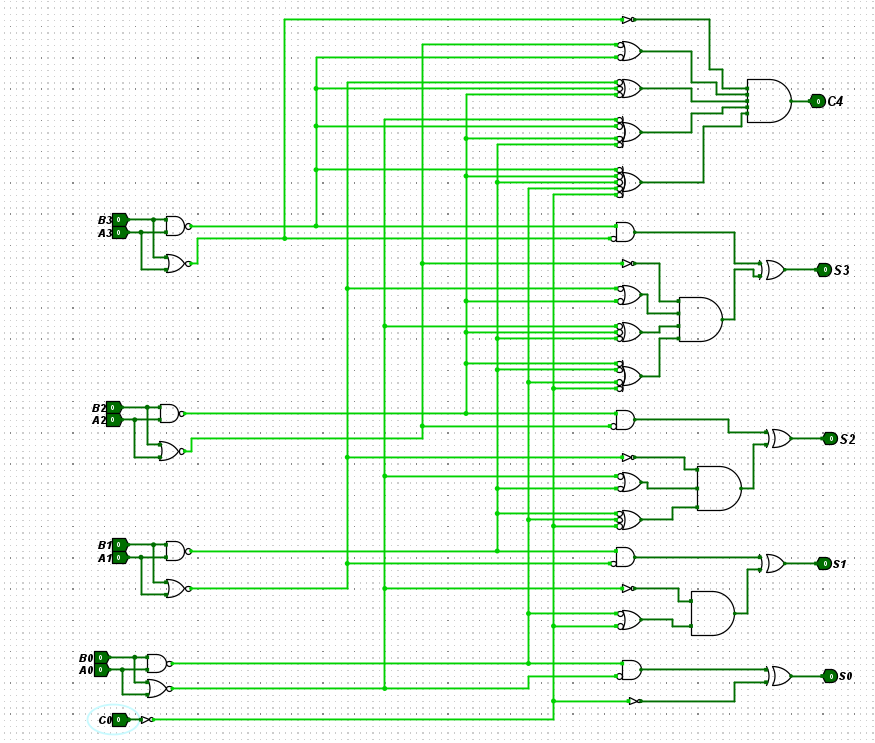
Logisim软件

1. **实验内容**
2. 根据给出的电路原理图，实现并验证4位快速加法器CLA。

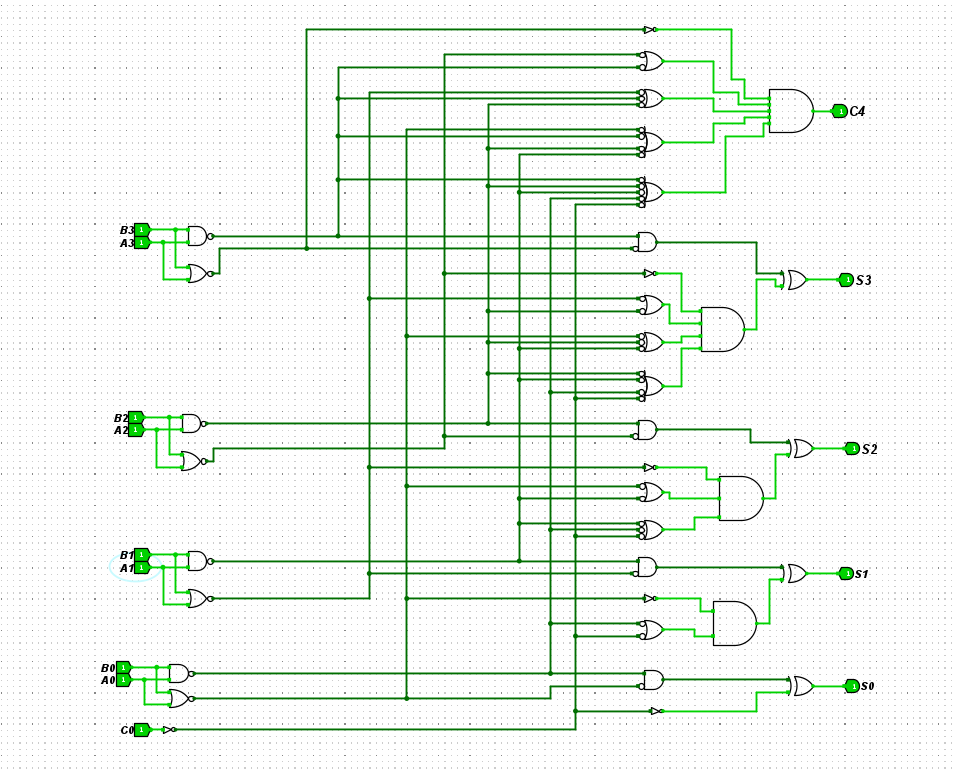
1） 添加逻辑门：放置需要的逻辑门、输入、输出引脚等，并布局到适当位置 打开 Logisim 软件，通过快捷工具栏放置与门、输入引脚、输出引脚。

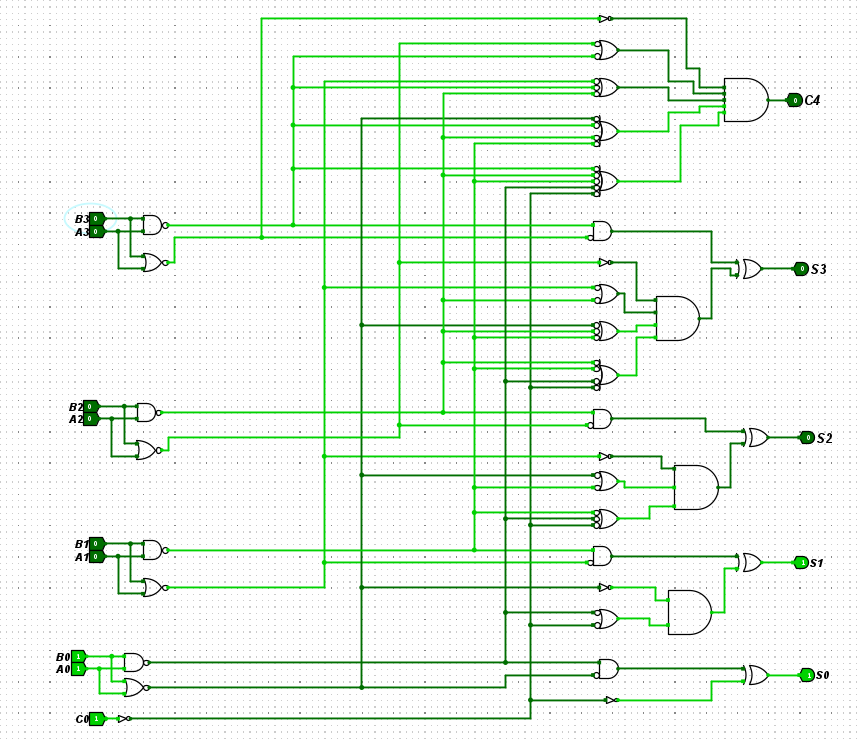
2） 添加线路：将输入引脚、逻辑门的输入端、输出端、输出引脚等通过连接线相连

3） 添加标识符：添加注释文字，便于电路的理解,得到电路图：



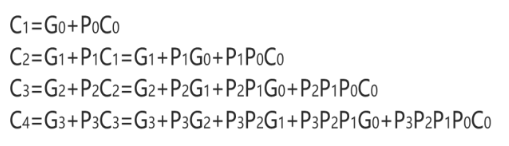
4） 仿真测试：进入仿真状态，验证电路功能：





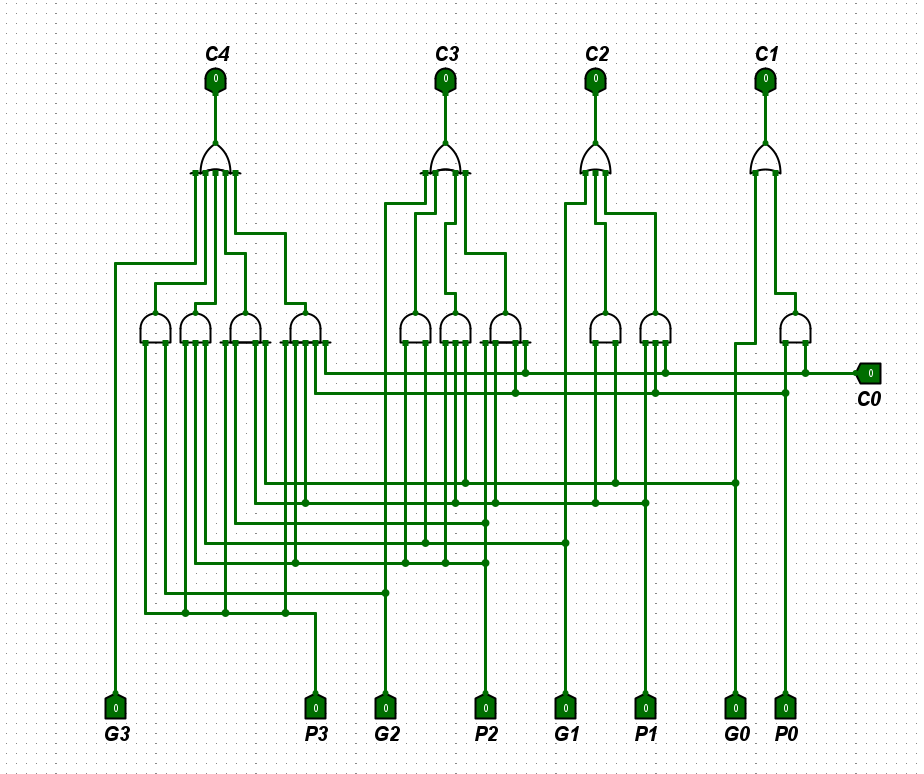
2、 根据给出的逻辑表达式，选择合适的逻辑门，实现并验证 4 位先行进位逻辑单元 CLU。

(1)实验原理: 逻辑表达式:

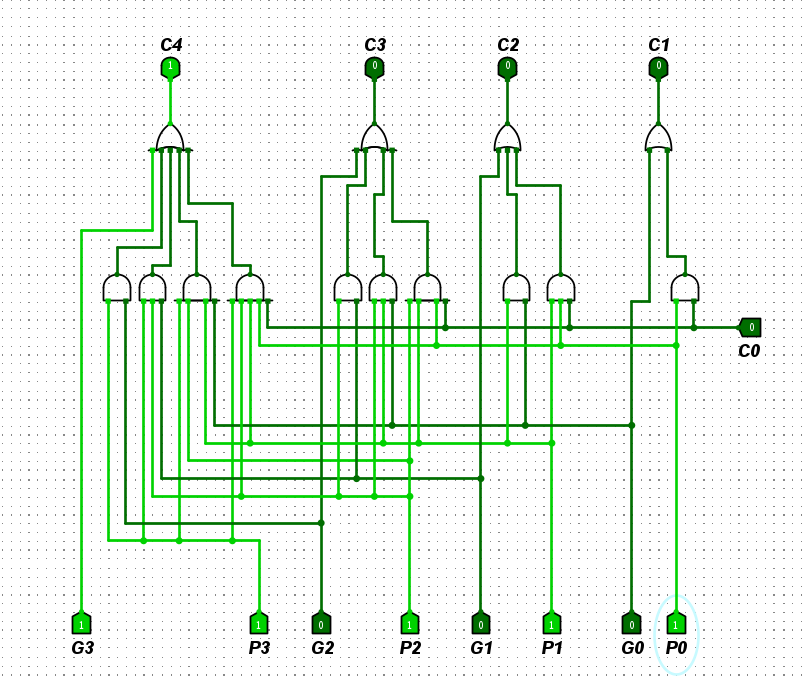


2） 添加线路：将输入引脚、逻辑门的输入端、输出端、输出引脚等通过连接线相连

3） 添加标识符：添加注释文字，便于电路的理解,得到电路图即可：



4） 仿真测试：进入仿真状态，验证电路功能：



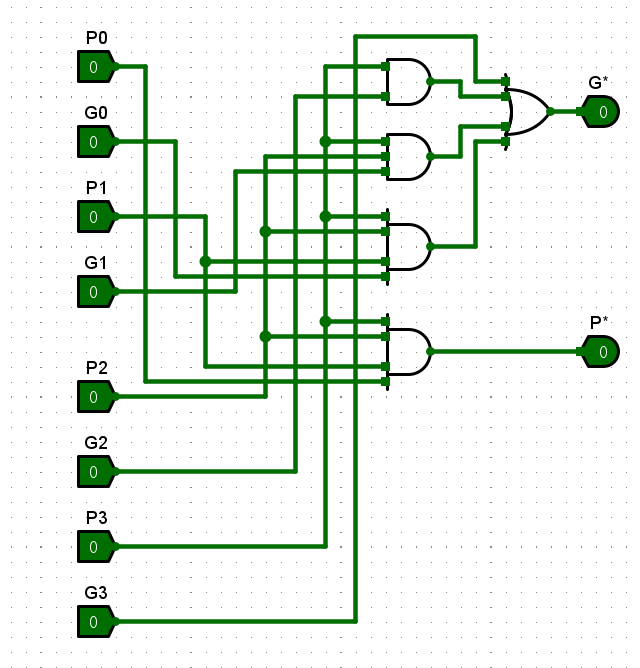
3、 据给出的逻辑表达式和电路原理图，在 4 位 CLA 中增加支持组间并联的 Gg、Pg 输出 端，加上 4 位CLU，设计并实现 16 位先行进位加法器。

(1)实验原理: 逻辑表达式:

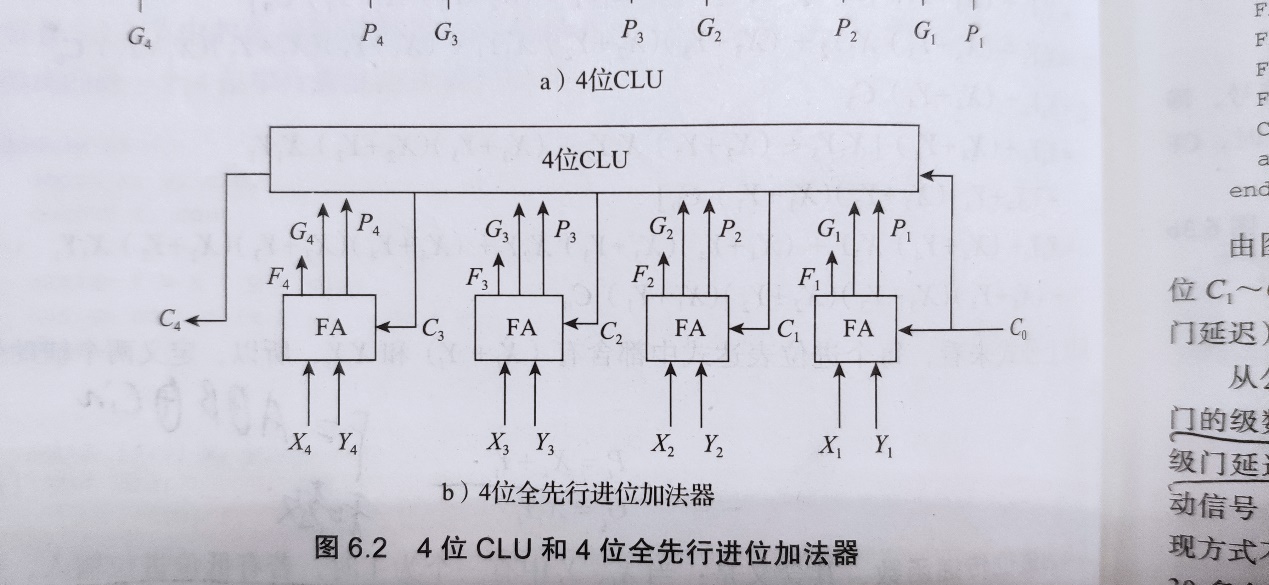
根据Gg=G3+P3G2+P3P2G1+P3P2P1G0

Pg=P3P2P1P0

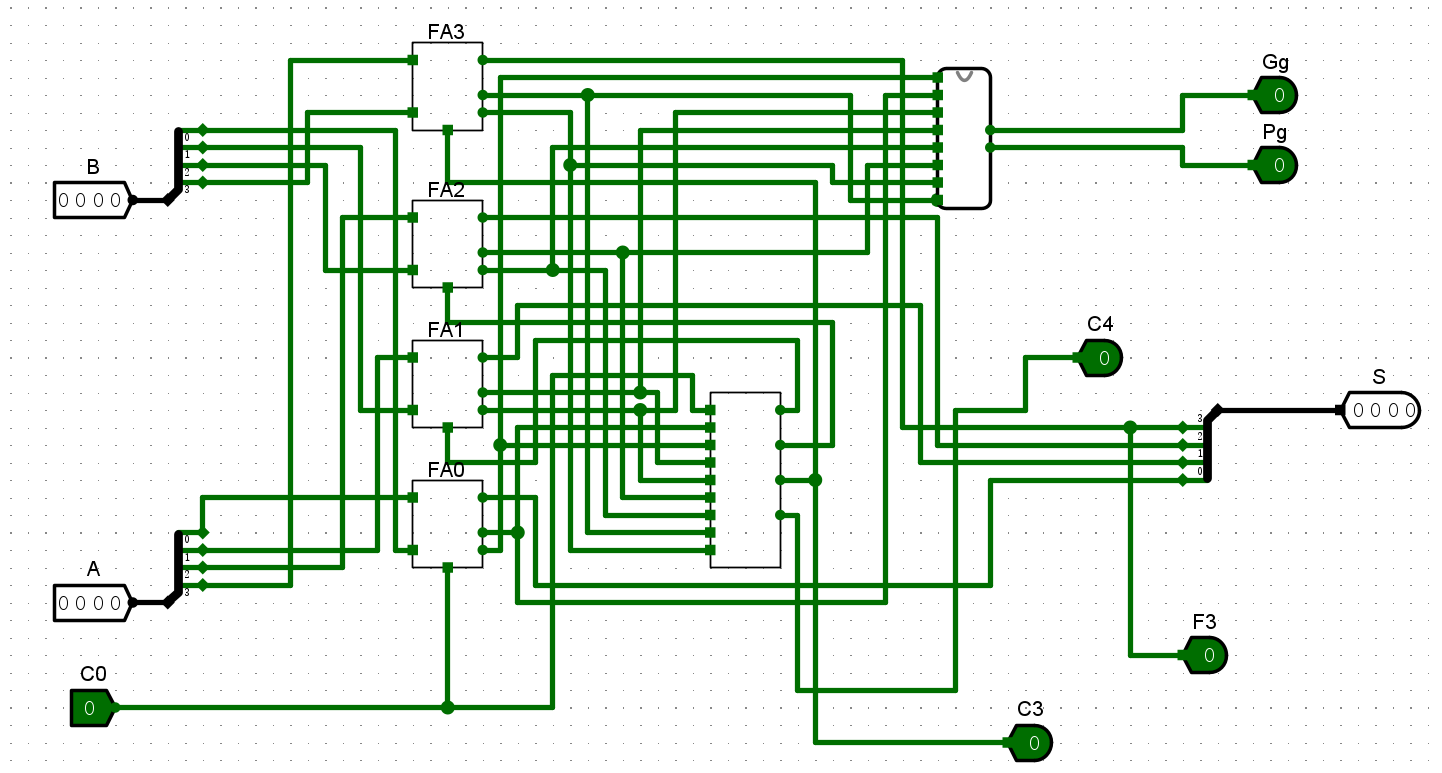
来实现辅助进位函数Gg和Pg，得到电路图：



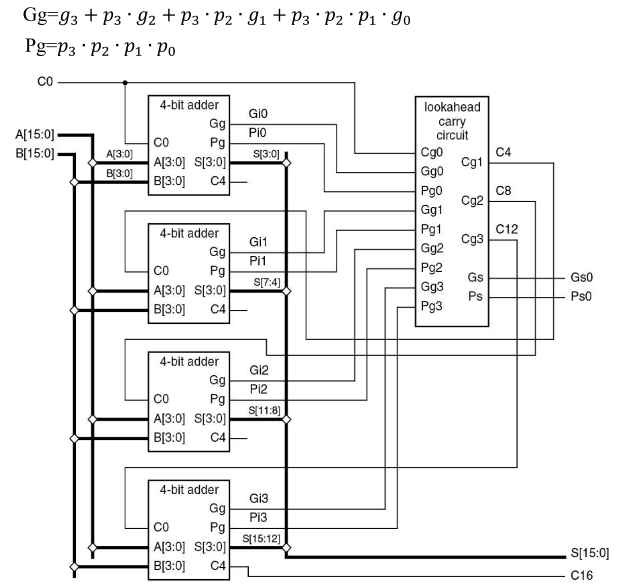
2） 添加线路：将输入引脚、逻辑门的输入端、输出端、输出引脚等通过连接线相连。根据四位加法器原理图实现四位加法器：



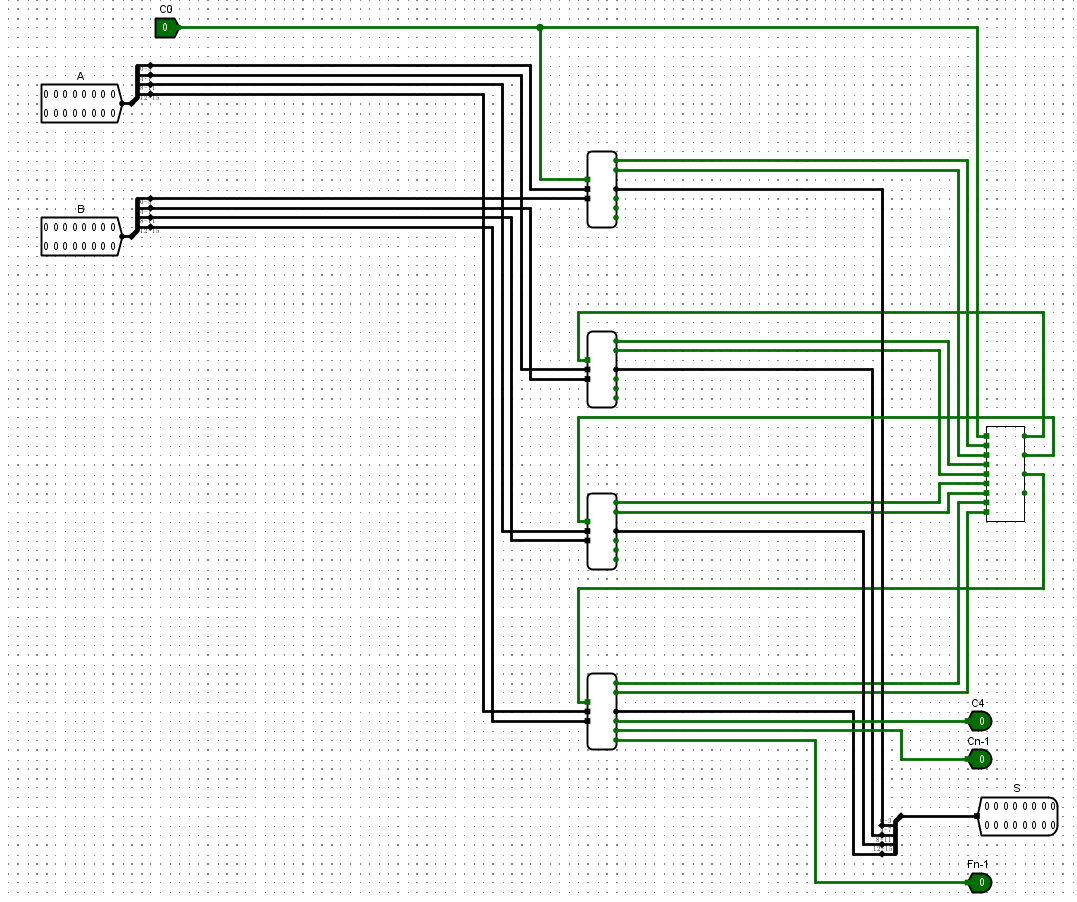
同时，加入之前生成好的Pg和Gg函数，得到四位加法器电路图：



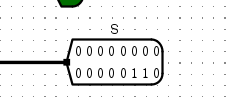
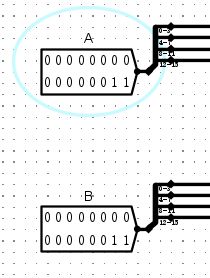
根据十六位加法器电路原理图实现十六位加法器

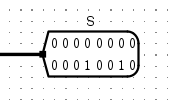
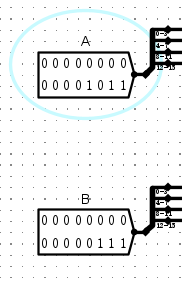


3） 添加标识符：添加注释文字，便于电路的理解,得到电路图即可：



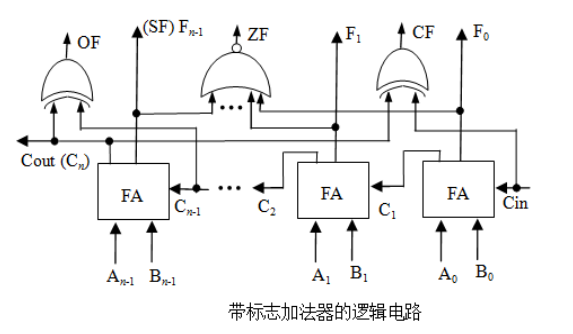
4） 仿真测试：进入仿真状态，验证电路功能：





4、 利用两个 16 位先行进位加法器构建一个 32 位加法器，并根据给出的标志位生成电路 原理图，在 32 位加法器中生成CF、SF、OF、ZF标志位。

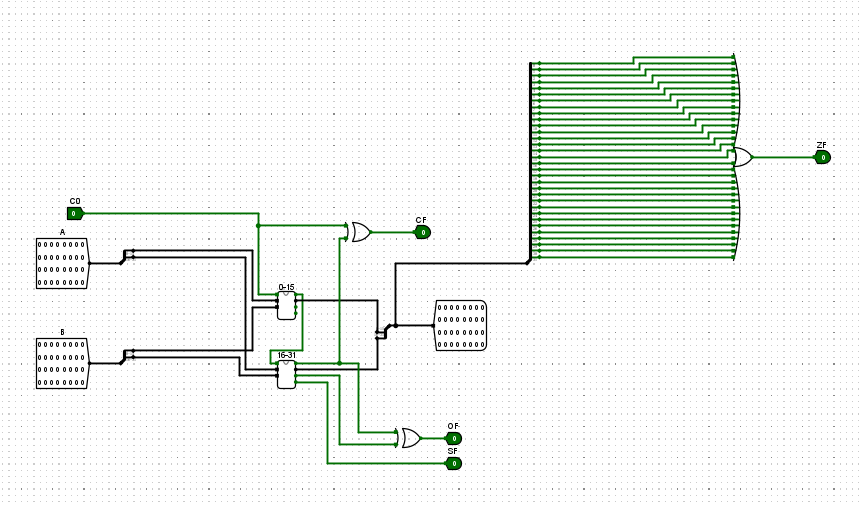
1. 实验原理：



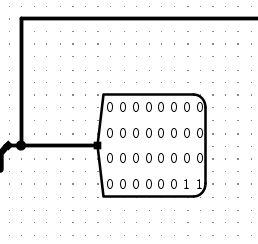
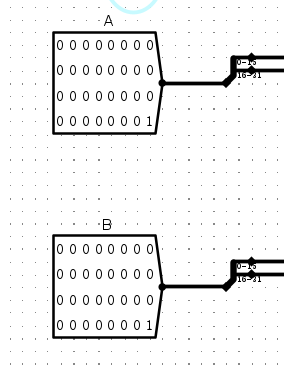
（2）连接电路：

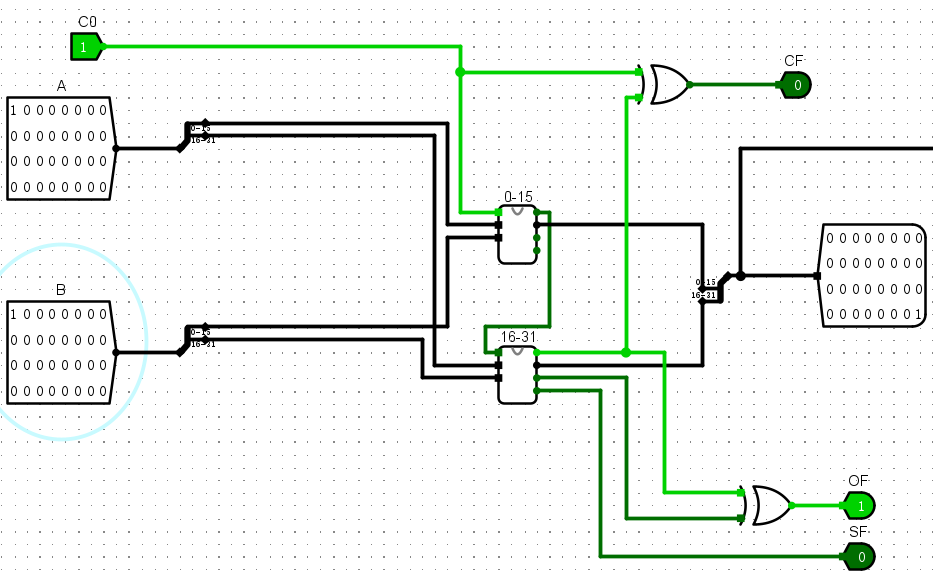
将两个16位加法器串联起来，将低十六位的进位输出，连接高十六位的进位输入，CF=Cin⊕Cout；ZF=~(F0|F1…Fn-1)；OF=Cn-1⊕cout; SF=Fn-1;

将c0与高十六位的cout异或得到CF，将高十六位的Cn-1与cout异或得到OF;连接电路得到电路图。



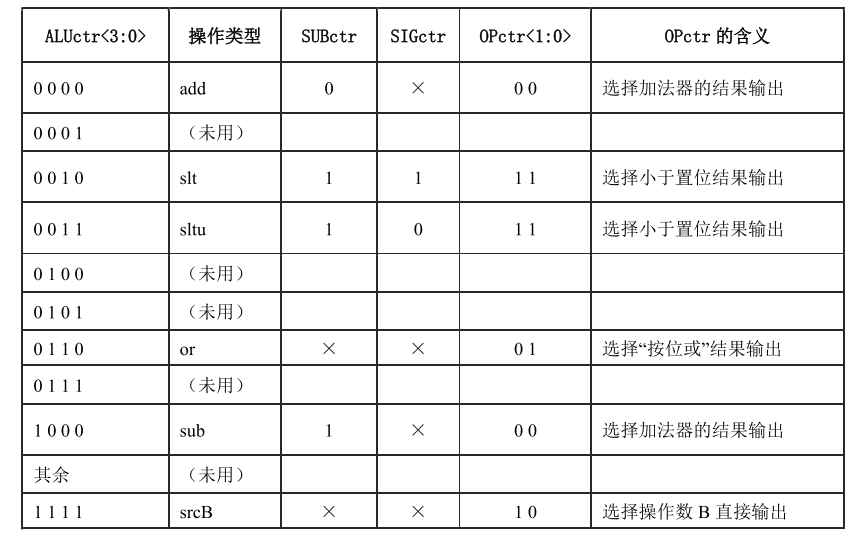
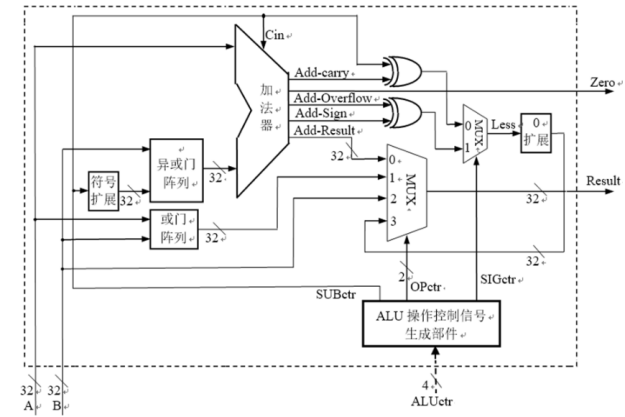
（3）电路仿真：进入仿真状态，验证电路功能：



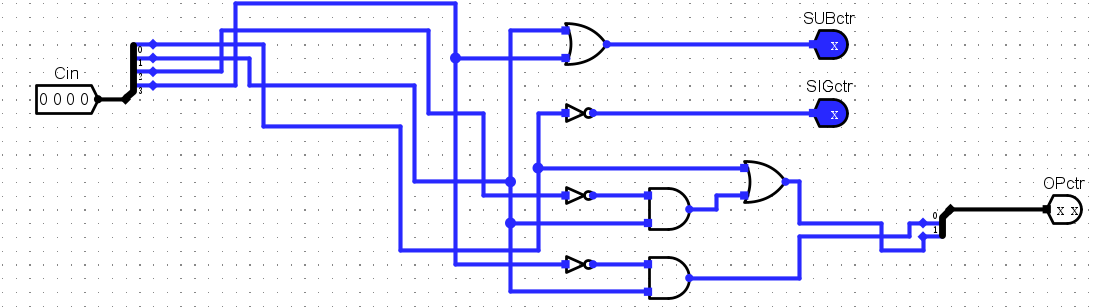


5.根据给出的电路原理图和ALU 引脚定义，设计一个支持 9 条RV32I 指令所包含的 6 种 操作（add、or、slt、sltu、srcB、sub(判 0)）的 32 位ALU。

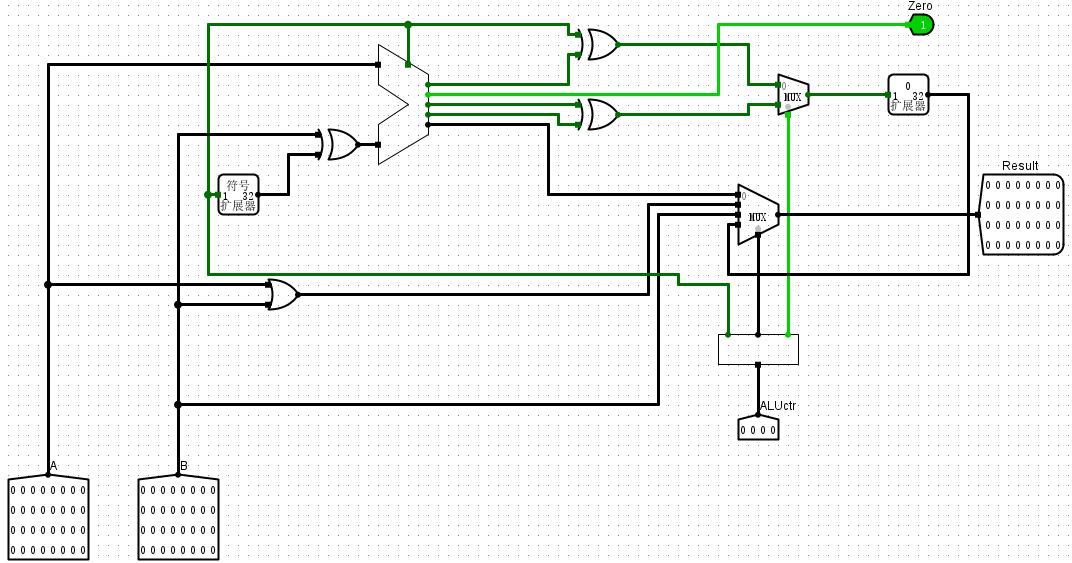
（1）实验原理：



根据ALuctr的编码生成aluctr的电路图：8

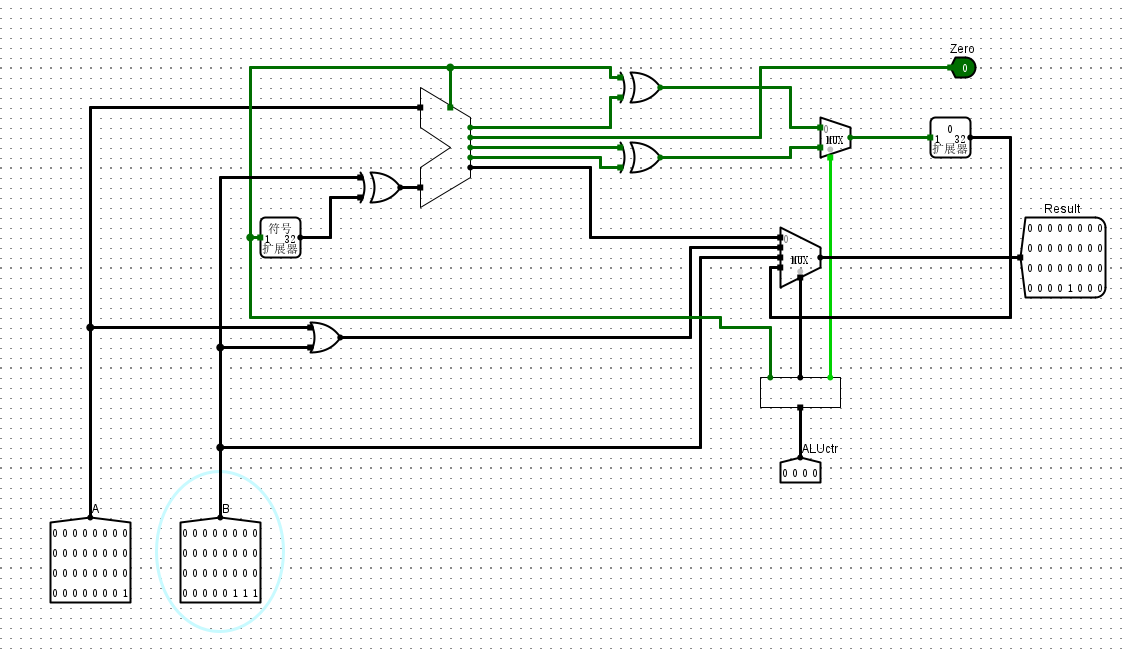


2） 添加线路：将输入引脚、逻辑门的输入端、输出端、输出引脚等通过连接线相连，将加法器，多路选择器，0扩展，符号扩展等按照电路原理图连接，得到电路图：

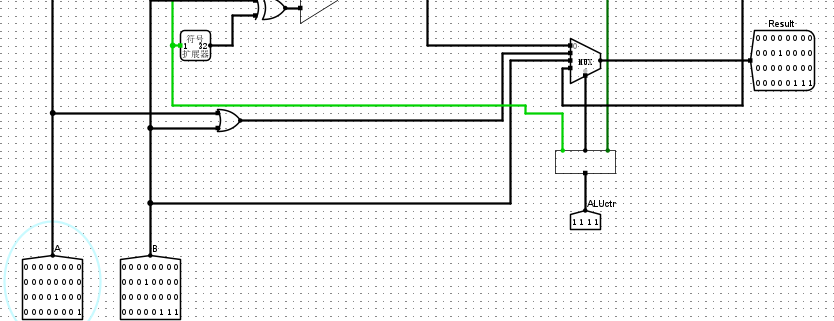


（3）电路仿真：

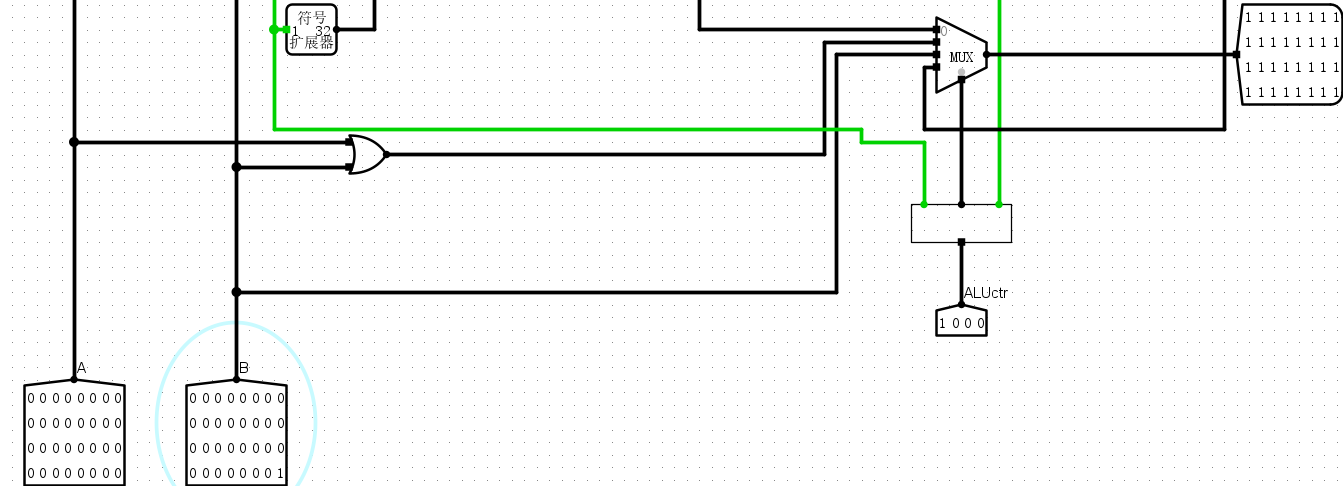
Aluctr为全零做加法运算：



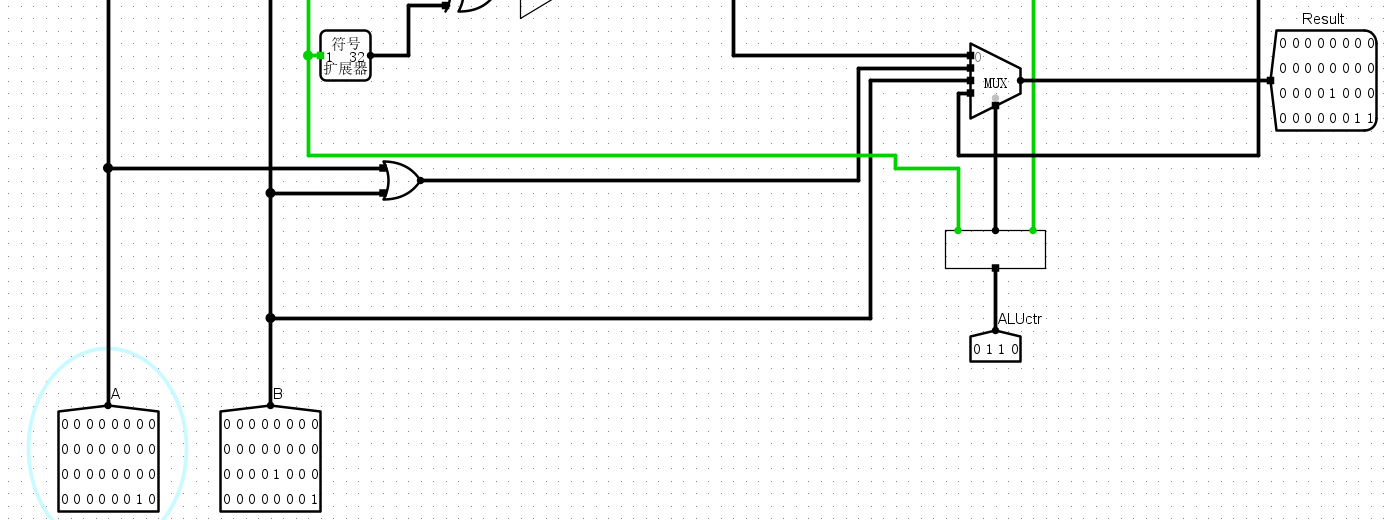
Aluctr为全1是选择B直接输出：



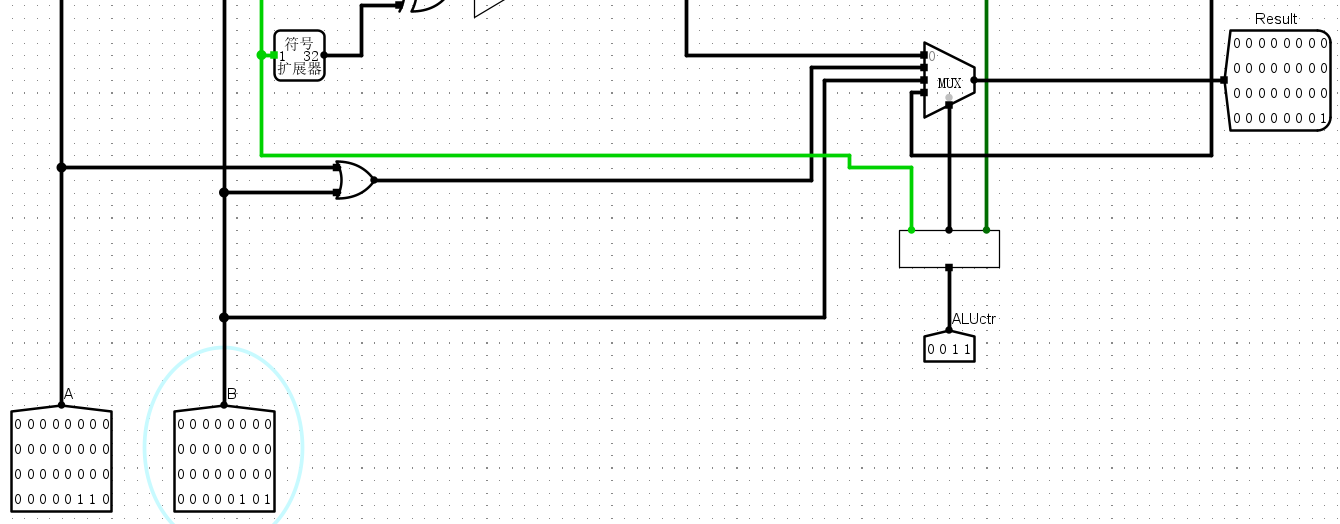
Aluctr为1000时做减法a-b：



Aluctr为0 1 1 0选择“按位或”结果输出：

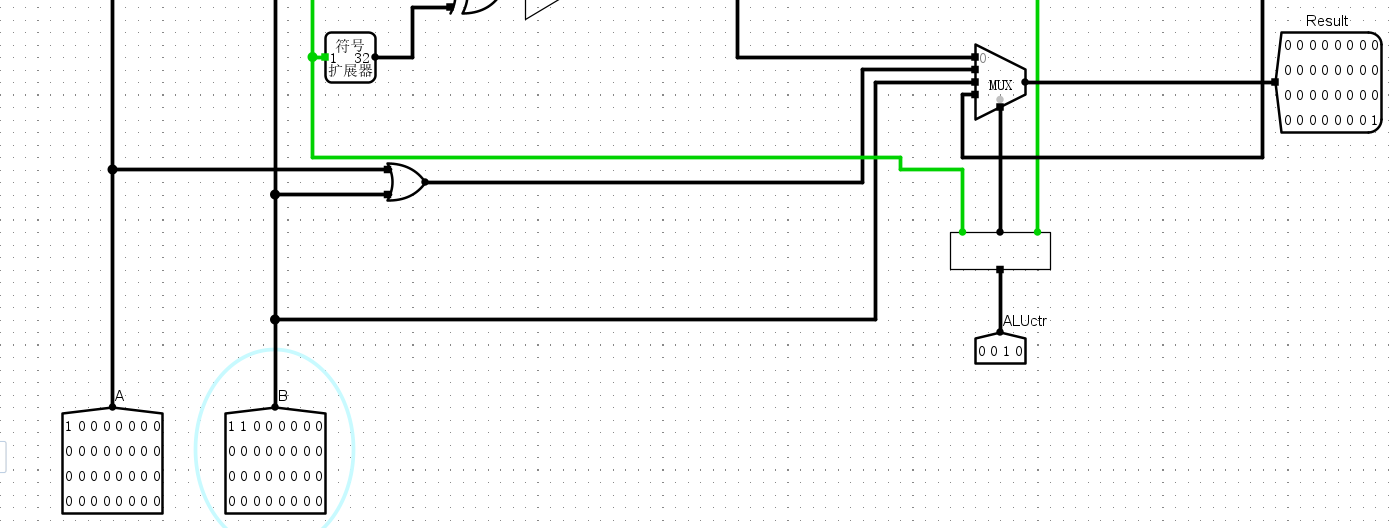


Aluctr为0 0 1 1选择无符号小于置位结果输出：



b <A时置1，

Aluctr为0 0 1 0选择有符号小于置位结果输出：b <A时置1，

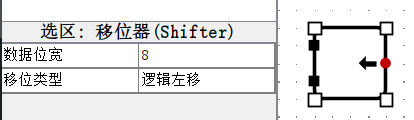


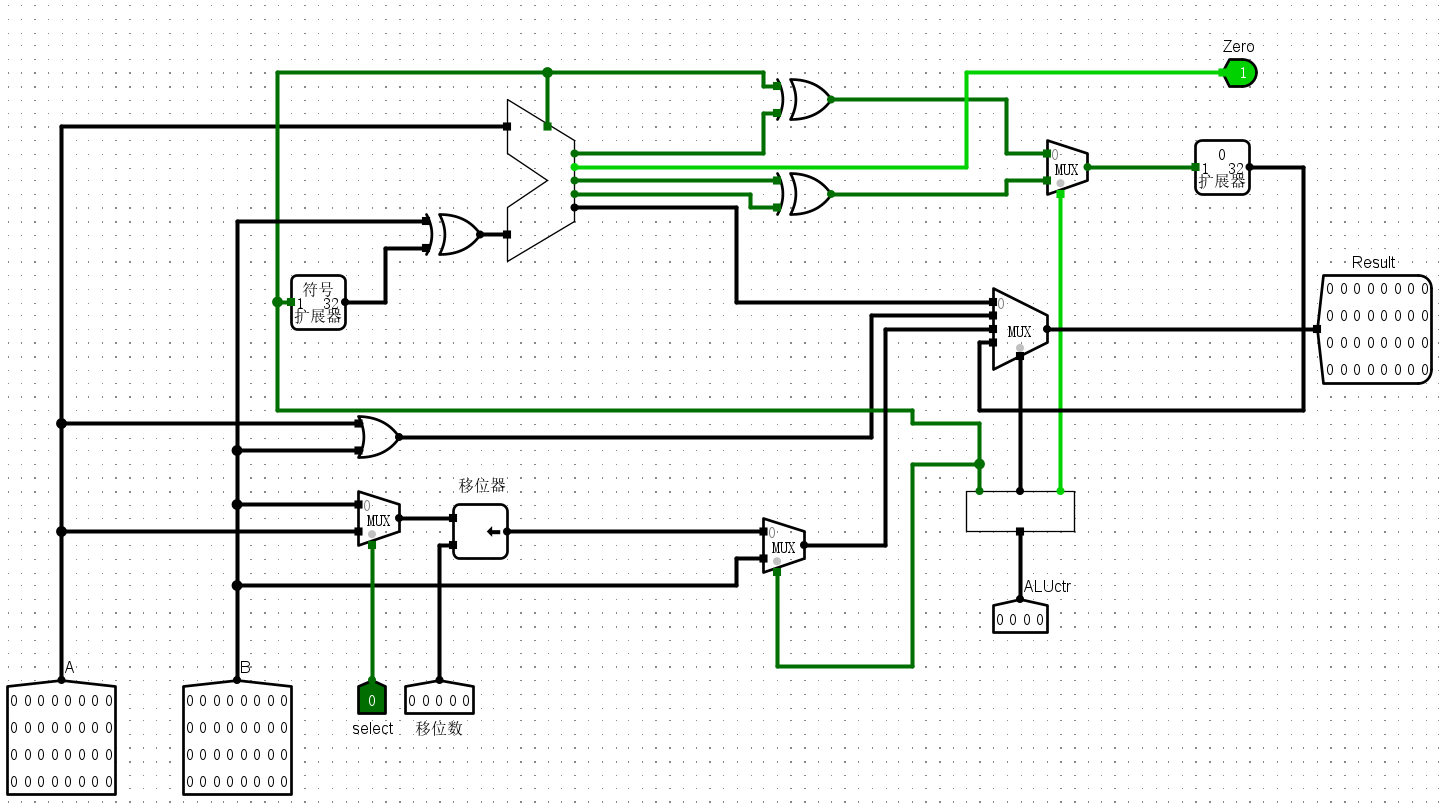
四、思考题：

1、 若需要增加一条“sub rd, rs1, rs2”指令，则在所设计的 32 位ALU中要做哪些修改？

答：由于ALU中已存在sub操作，只需将ALUctr置位1000即可完成减法操作。可以验证结果的正确性。

1. 若需要增加一条“sll rd, rs1, rs2”指令，则在所设计的 32 位ALU中要做哪些修改？

答：sll 表示逻辑左移,空位补0 . sll rd,rs1,rs2表示将rs1的存储数左移rs2位后存入rd寄存器。移位操作可以用logisim运算器中的移位器实现。同时，在ALU电路要能支持该移位操作，ALUctr中也加入相应的控制信号。

分析ALUctr的编码方案可知，0001是一个未用编码，可以使用它来作为sll操作的编码，由于在我实现的ALUctr中当ALUctr为0001时，OPctr也为10，可用SUBctr来加以区别。实现后的电路图如下：其中select用来选择对AB中的哪一个进行移位，移位数输入移位数目，将多路选择器与移位器相连接，将多路选择器与移位器和操作数B连接右侧MUX用来选择输出B还是移位后的操作数。当ALUctr为0001时进行移位操作，此时SUBctr为0，因此选择移位器端输出，当ALUctr为1111时，SUBctr为1，选择B数据直接输出。



1. 如何验证运算器的结果是否正确？

答：进入仿真模式，验证4位CLU是否正确，4位加法器是否正确，若正确，继续验证16位加法器是否正确，同理验证32位加法器是否正确。