

数字电路与数字系统实验报告

实验四：算术逻辑部件

院系：人工智能学院

姓名：方盛俊

学号：201300035

班级：人工智能 20 级 2 班

邮箱：201300035@smail.nju.edu.cn

时间：2021年5月19日

目录

- 实验四: 算术逻辑部件
 - 目录
 - 一, 实验目的
 - 二, 实验环境 / 器材
 - Logisim-ITA V2.16.1.2
 - 头歌线上评测平台
 - 三, 实验内容
 - 1. 4 位先行进位加法器
 - (a) 实验原理
 - (b) 实验步骤
 - (c) 仿真验证
 - (d) 实验结果
 - 2. 16 位先行进位加法器
 - (a) 实验原理
 - (b) 实验步骤
 - (c) 仿真验证
 - 3. 算术逻辑部件 (ALU)
 - (a) 实验原理
 - (b) 实验步骤
 - (c) 仿真验证
 - (d) 实验结果
 - 四, 实验中遇到的问题和解决方法
 - 1. 对于 Logisim 器件不熟悉
 - 2. 未看清楚题目

一, 实验目的

1. 掌握使用 Logisim 软件设计, 实现算术逻辑部件的方法
2. 学习 4 位先行进位加法器 CLA 和先行进位逻辑单元 CLU 的设计原理和实现方法
3. 学习 16 位先行进位加法器及相关标志位的设计原理和实现方法
4. 学习基本算术逻辑部件的设计原理和实现方法, 实现 6 种操作的 ALU 器件

二, 实验环境 / 器材

Logisim-ITA V2.16.1.2

<https://sourceforge.net/projects/logisimit/>

头歌线上评测平台

<https://www.educoder.net/classrooms/10924/>

三, 实验内容

1. 4 位先行进位加法器

(a) 实验原理

1 位全加器 (FA): 有两个位输入 X, Y 和进位输入 Cin, 输出加法计算结果 F, 进位传递位 P 和进位生成位 G.

其中逻辑表达式为

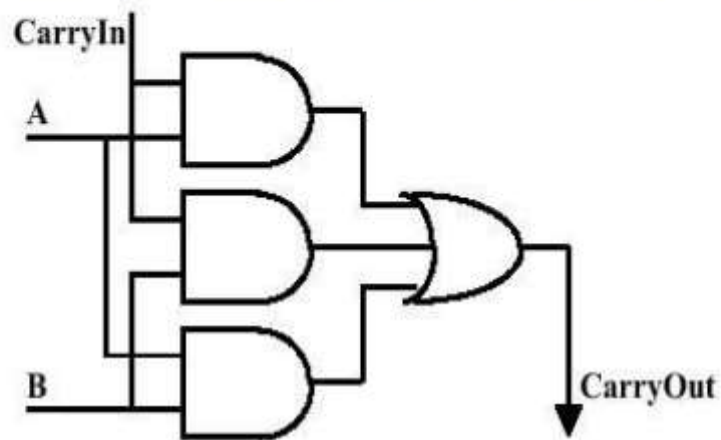
$$F = X \oplus Y \oplus C_{in}$$

$$P = X \oplus Y$$

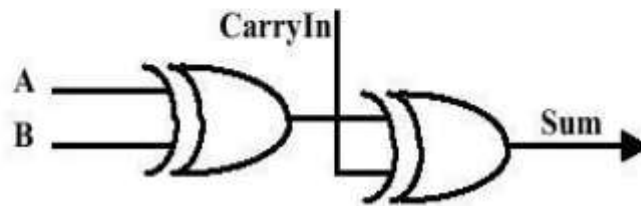
$$G = X \& Y$$

电路图:

$$\text{CarryOut} = B \& \text{CarryIn} \mid A \& \text{CarryIn} \mid A \& B$$

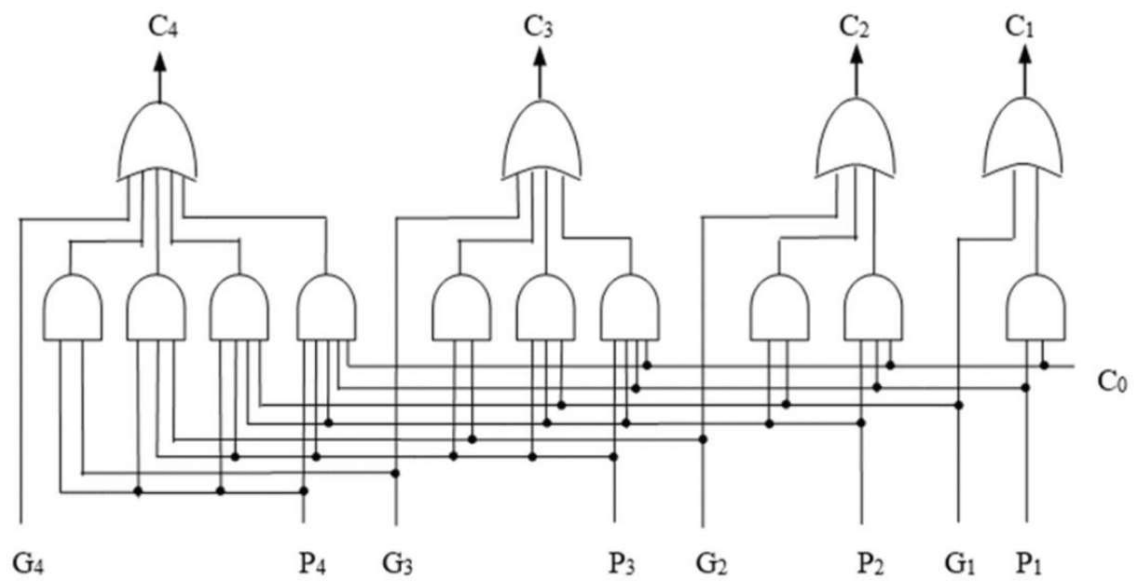


$$\text{Sum} = A \text{ XOR } B \text{ XOR } \text{CarryIn}$$



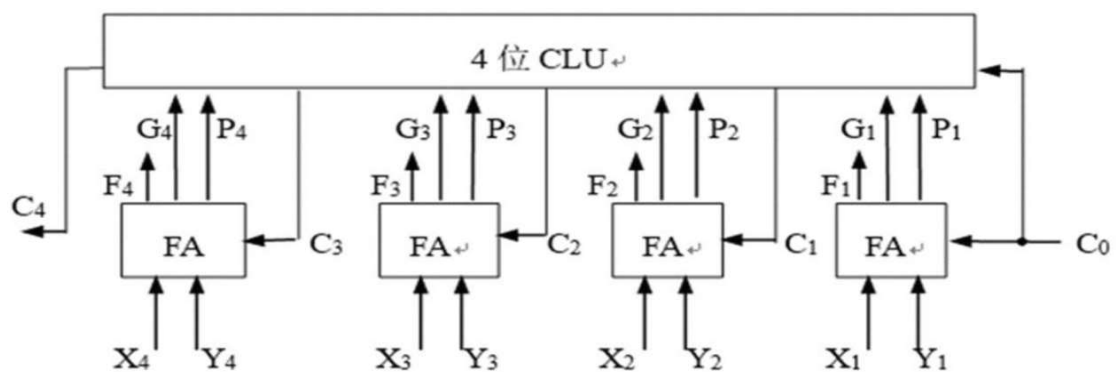
先行进位部件 (CLU): 输入为 4 位进位传递信号 P1, P2, P3, P4, 4 位进位生成信号 G1, G2, G3, G4, 和一位进位输入 Cin (同 C0); 其输出为四位进位信号 C1, C2, C3, C4.

电路图:



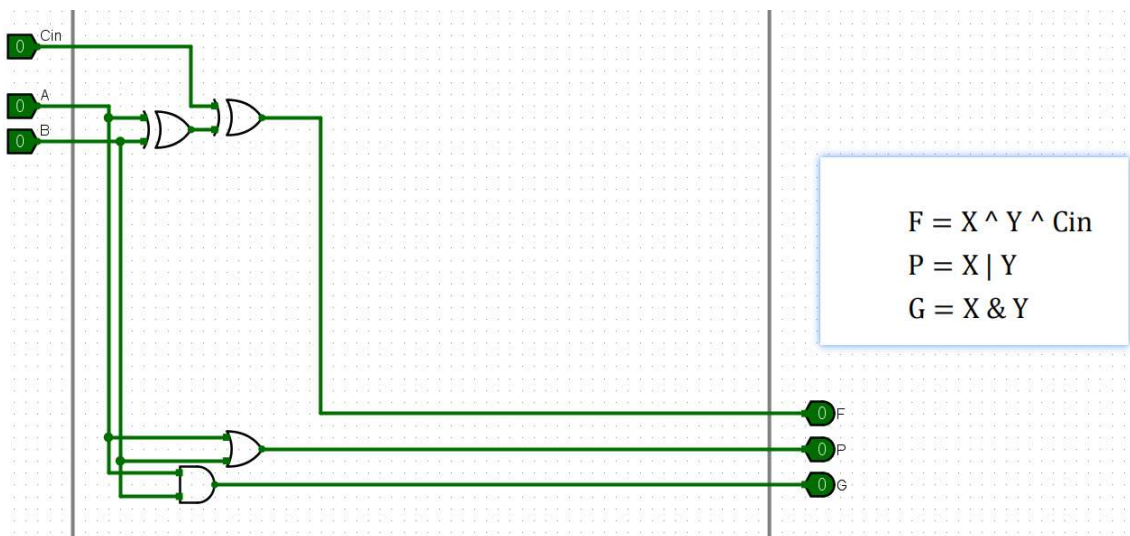
4 位 CLA: 由 4 个 FA 和一个 4 位 CLU 组成, 实现四位的加法功能.

电路图:

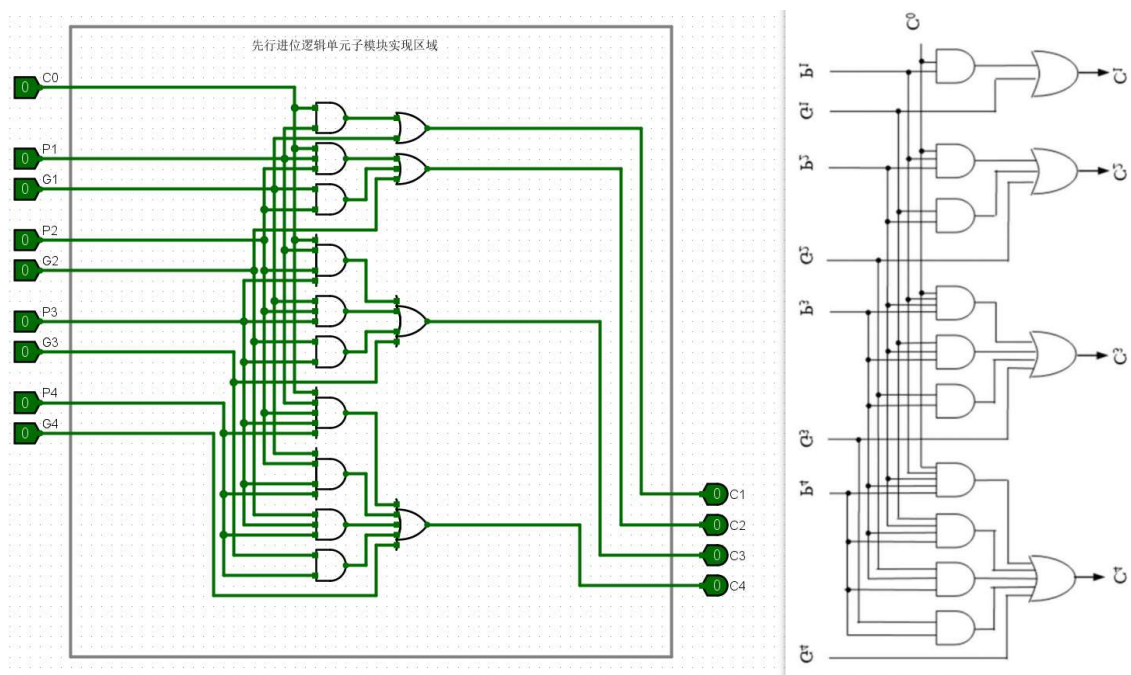


(b) 实验步骤

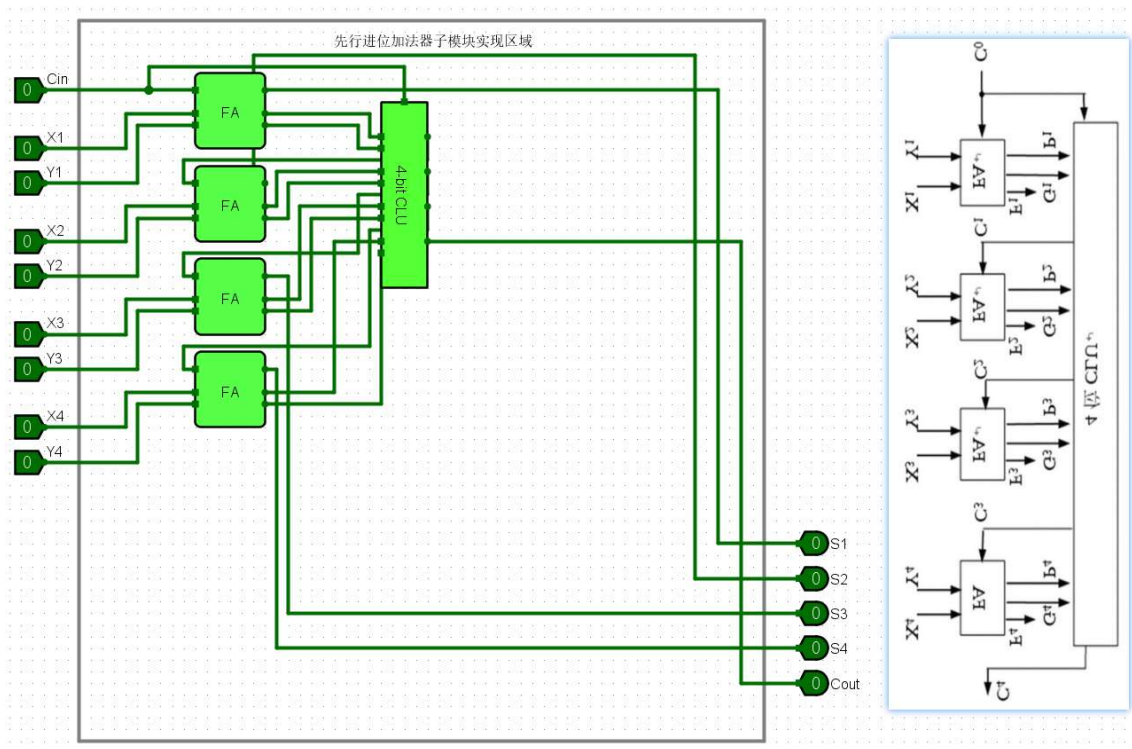
1. 先将 1 位全加器根据逻辑表达式搭建完毕.



2. 再制作 CLU 模块.



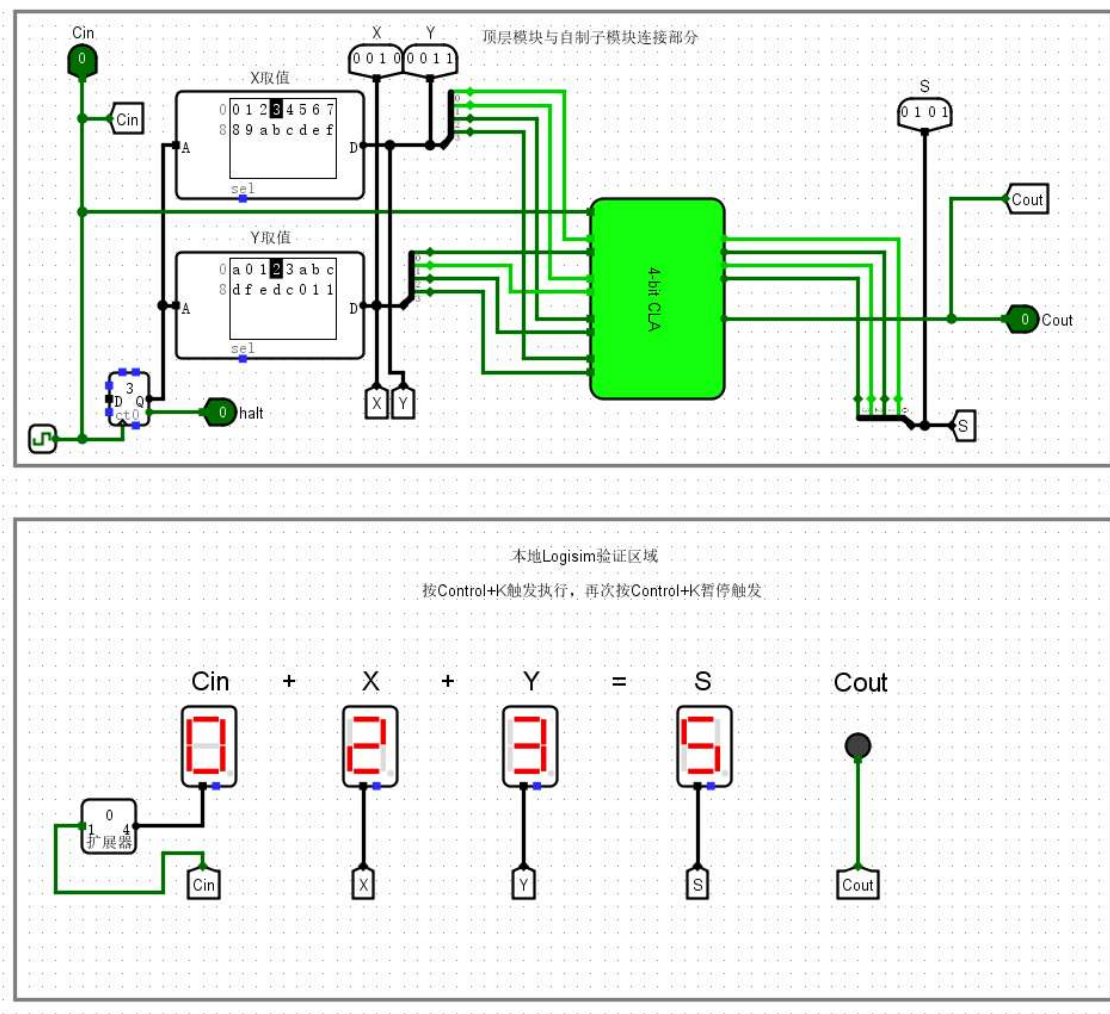
3. 基于 FA 和 CLU 制作 CLA.



(c) 仿真验证

按下 Ctrl + K 开始本地仿真, 结果良好.

最后结果如下:



(d) 实验结果

通过了头歌平台的验证。

1/1 全部通过

测试集1

消耗内存226.67MB 代码执行时长: 6.93秒

预期输出					实际输出				
0	1010	0000	1010	0	0	1010	0000	1010	0
1	0000	0001	0010	0	1	0000	0001	0010	0
0	0000	0001	0001	0	0	0000	0001	0001	0
1	0001	0010	0100	0	1	0001	0010	0100	0

2. 16 位先行进位加法器

(a) 实验原理

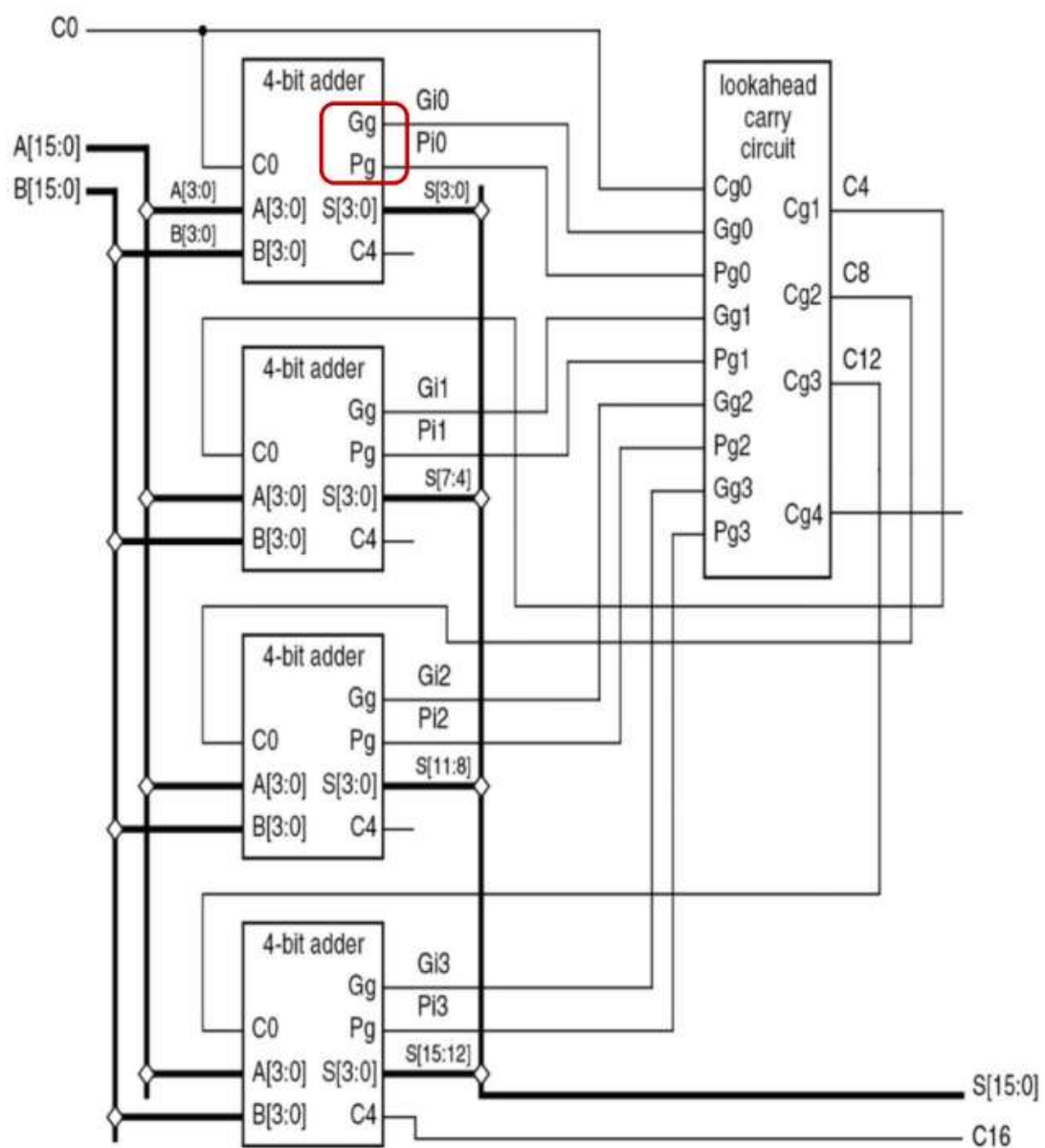
组间先行进位函数逻辑单元: 其输入为 4 位进位传递信号 P_1, P_2, P_3, P_4 和 4 位进位生成信号 G_1, G_2, G_3, G_4 , 输出为组间进位传递信号 P_g 和组间进位生成信号 G_g .

逻辑表达式为:

$$G_g = G_4 + P_4 \cdot G_3 + P_4 \cdot P_3 \cdot G_2 + P_4 \cdot P_3 \cdot P_2 \cdot P_1$$
$$P_g = P_4 \cdot P_3 \cdot P_2 \cdot P_1$$

16 位先行进位加法器: 通过 4 位 CLA 和组间先行进位函数逻辑单元组合形成.

电路原理图:

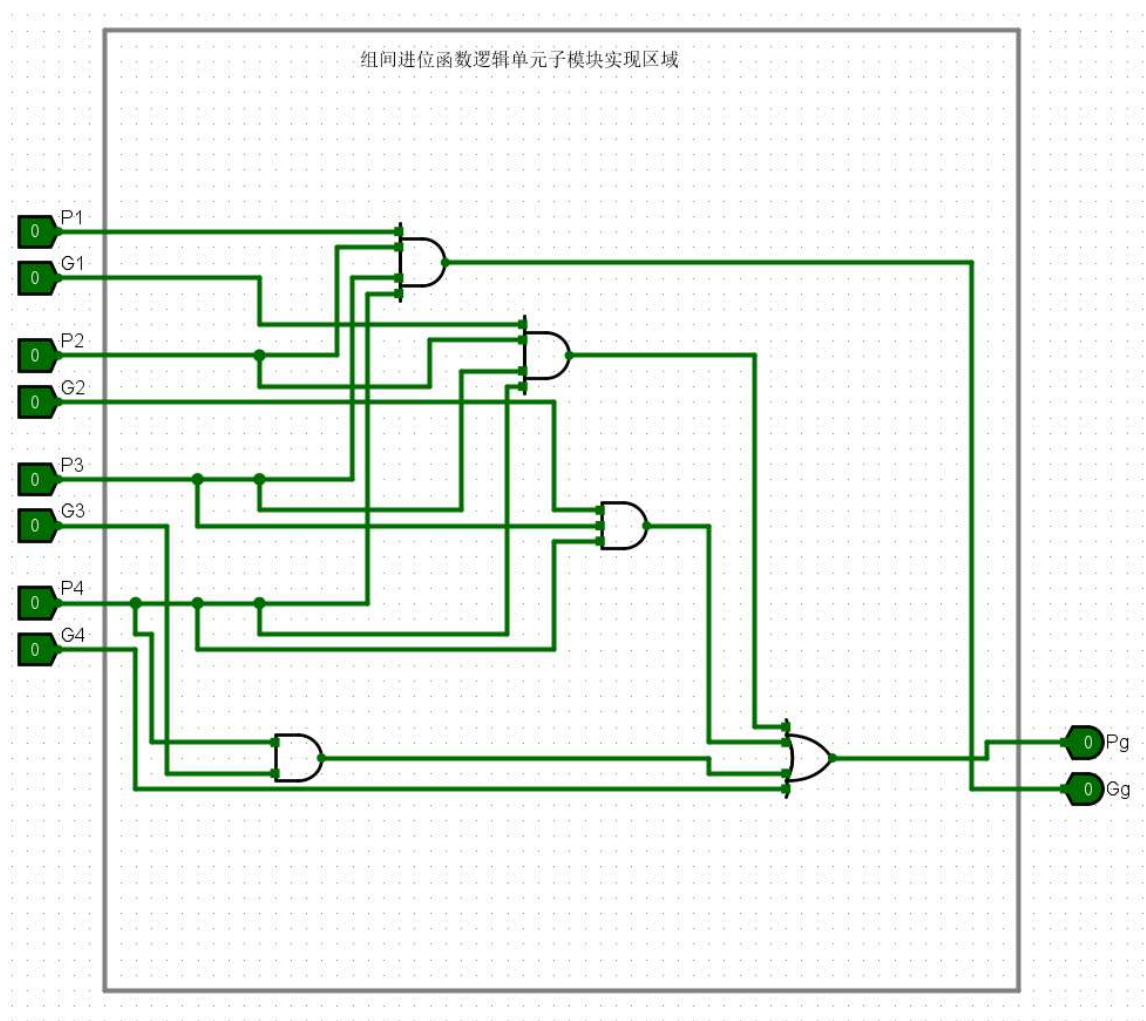


(b) 实验步骤

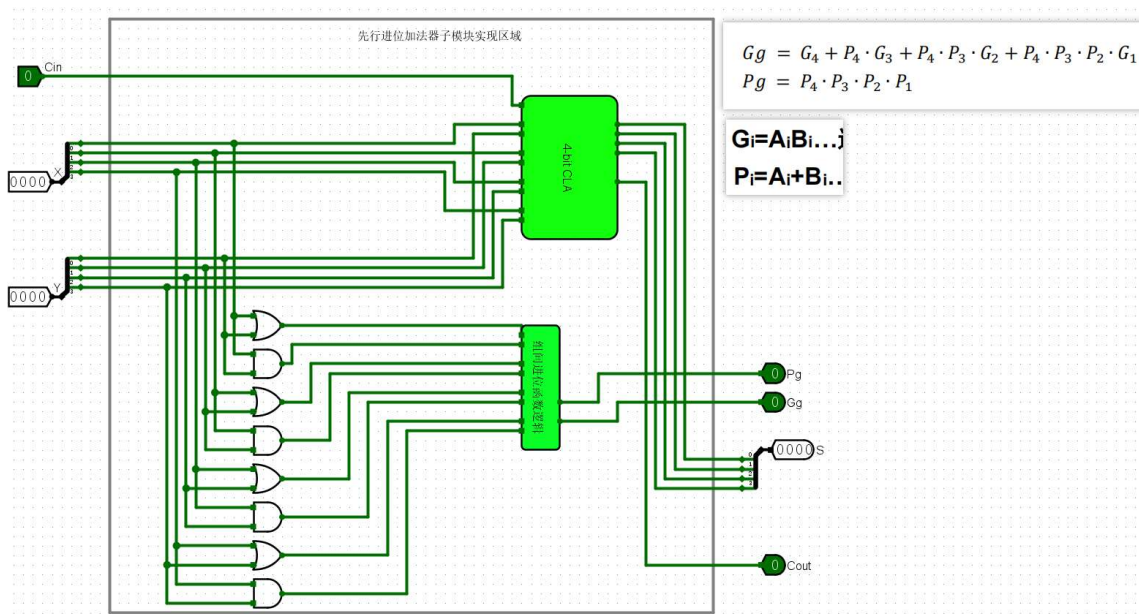
1. 引入 `exp4-1.circ` 的组件.



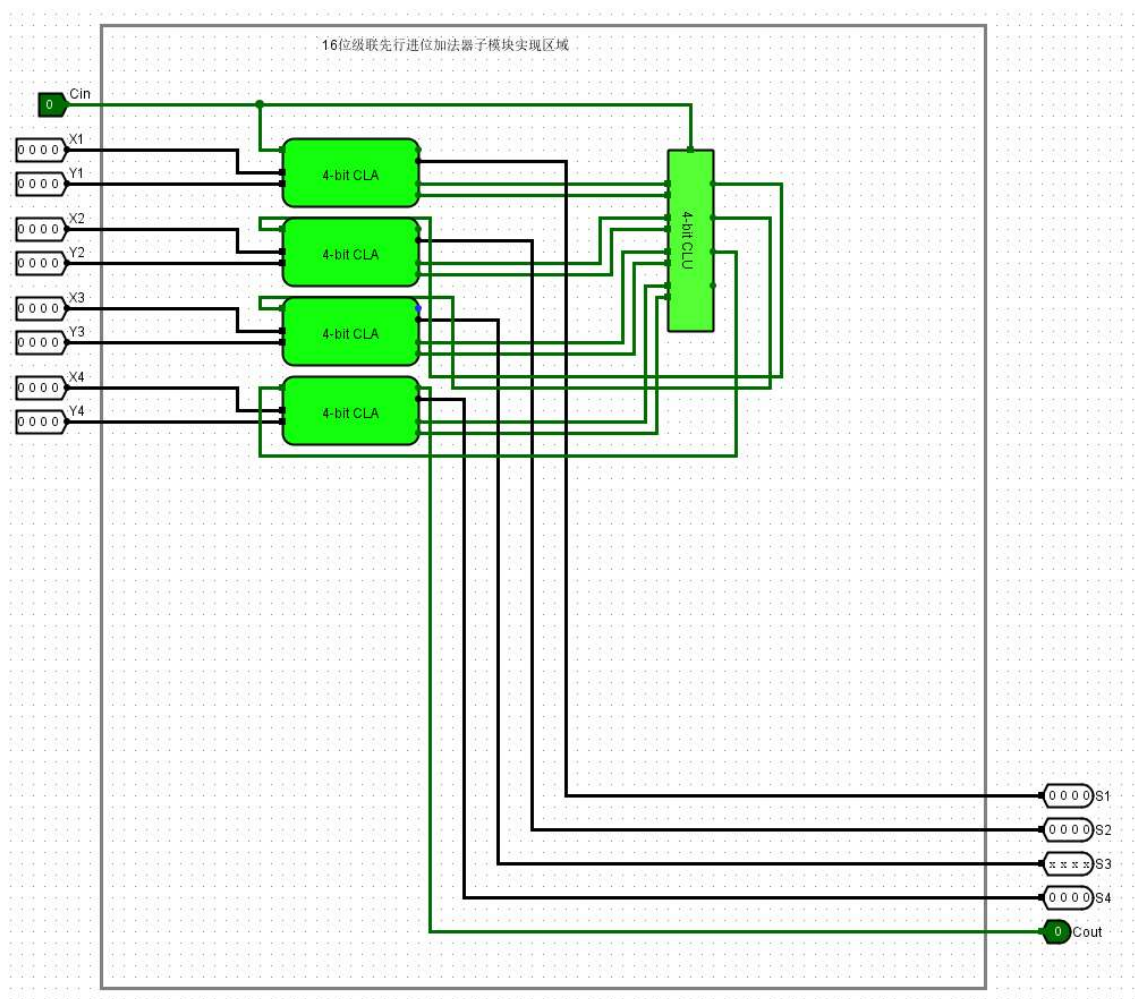
2. 根据逻辑表达式组建组间先行进位函数逻辑单元.



3. 修改 CLA 为可级联的 4 位 CLA.

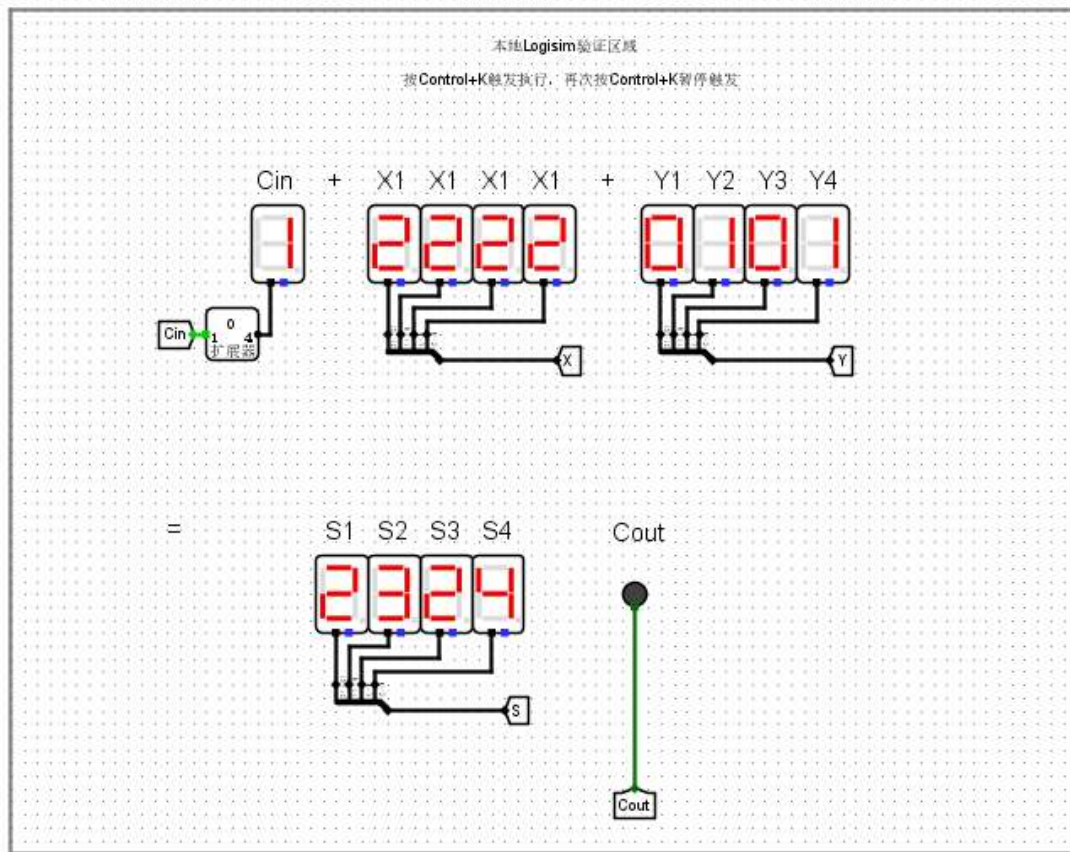
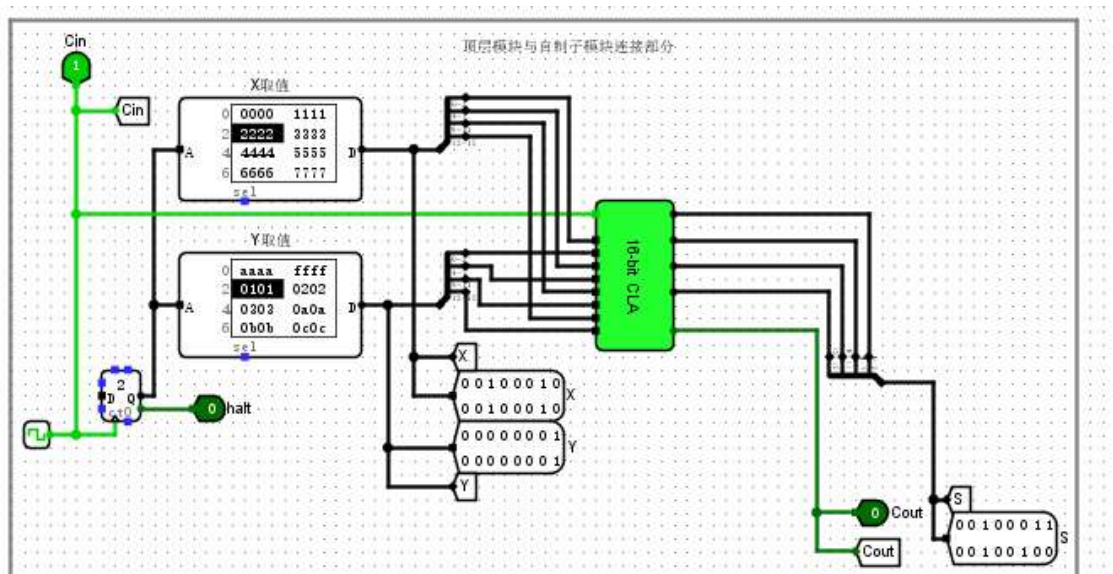


4. 使用 4 位 CLA 搭建 16 位先行进位加法器.



(c) 仿真实证

按下 Ctrl + K 开始本地仿真, 结果良好.

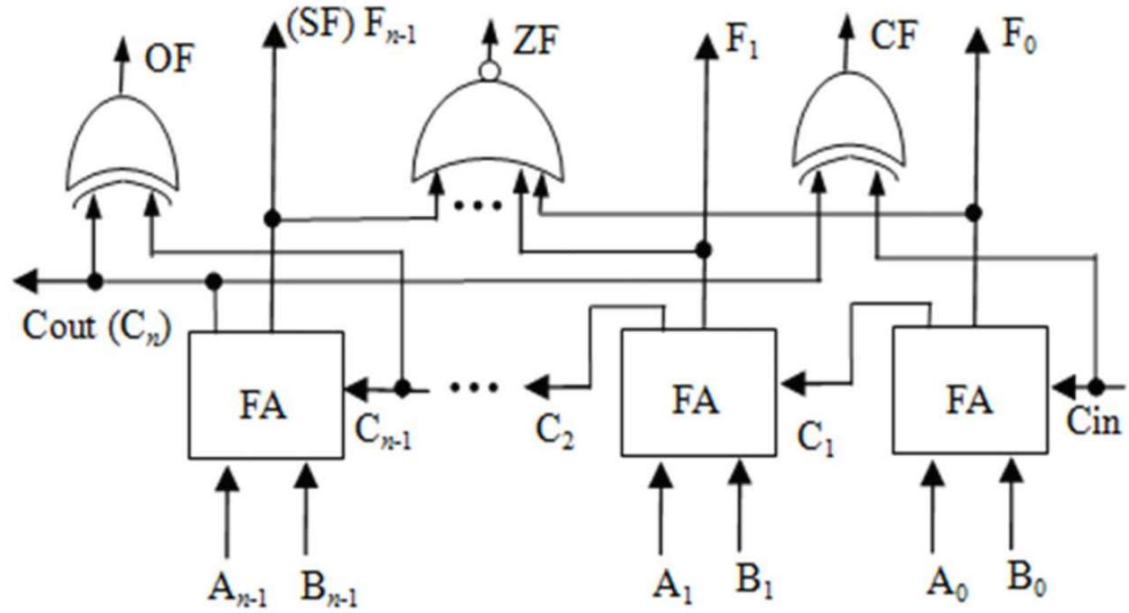


3. 算术逻辑部件 (ALU)

(a) 实验原理

4 位带标志位加法器: 输入为两个 4 位操作数 $X_1, X_2, X_3, X_4, Y_1, Y_2, Y_3, Y_4$ (由低位到高位, 同图中 $A_0 \sim A_3, B_0 \sim B_3$), 进位 C_{in} ; 输出为 4 位计算结果 S (同图中 F), 最高位进位 C_{out} , 溢出标志位 OF , 符号标志位 SF , 零标志位 ZF , 进/借位标志位 CF .

电路图如下:



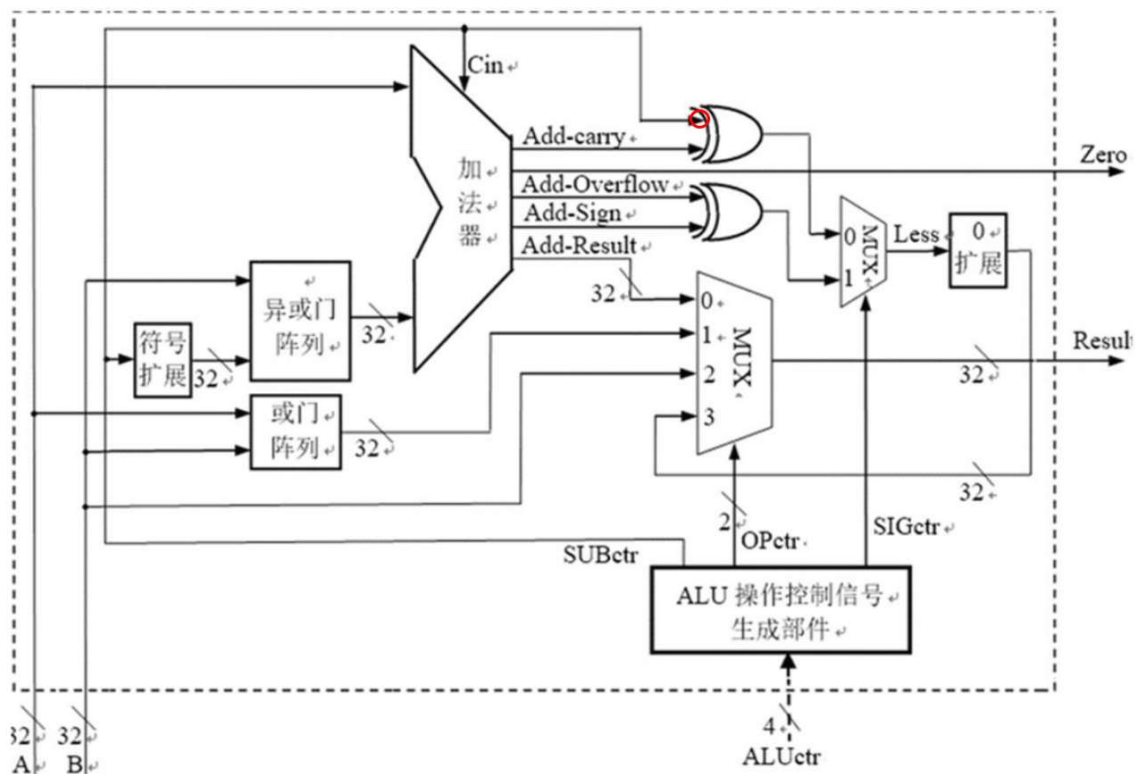
带标志加法器的逻辑电路

ALU: 有一个操作控制端 ($ALUop$), 用来决定 ALU 所执行的处理功能. $ALUop$ 的位数 k 决定了操作的种类例如, 当位数 k 为 3 时, ALU 最多只有 $2^3 = 8$ 种操作. 这里我们支持 6 种操作 ($add, slt, sltu, sub, or, srcB$)

操作对应的表格:

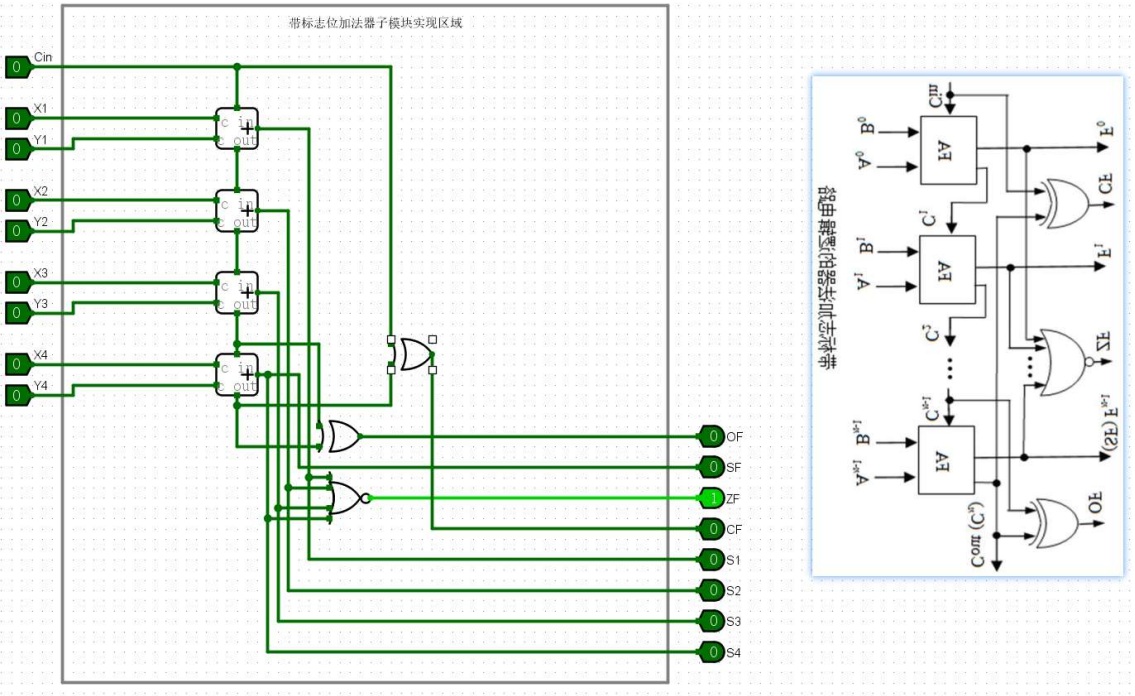
ALUctr<3:0>	操作类型	SUBctr	SIGctr	OPctr<1:0>	OPctr 的含义
0 0 0 0	add	0	×	0 0	选择加法器的结果输出
0 0 0 1	(未用)				
0 0 1 0	slt	1	1	1 1	使用减法做有符号整数的 比较大小, 如果 X 小于 Y 置位输出
0 0 1 1	sltu	1	0	1 1	使用减法做无符号整数的 比较大小, 如果 X 小于 Y 置位输出
0 1 0 0	(未用)				
0 1 0 1	(未用)				
0 1 1 0	or	×	×	0 1	选择按位或结果输出
0 1 1 1	(未用)				
1 0 0 0	sub	1	×	0 0	选择加法器的结果输出
其余	(未用)				
1 1 1 1	srcB	×	×	1 0	选择操作数 B 直接输出

电路图如下:

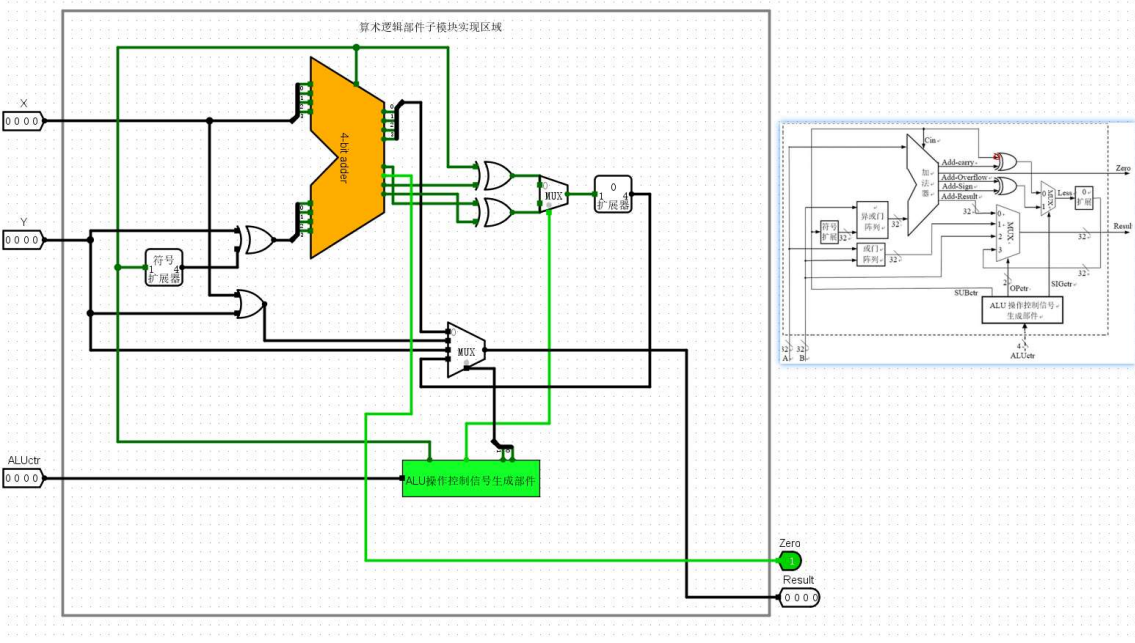


(b) 实验步骤

1. 借助 Logisim 自带的 FA 构建 4 位 带标志位加法器.

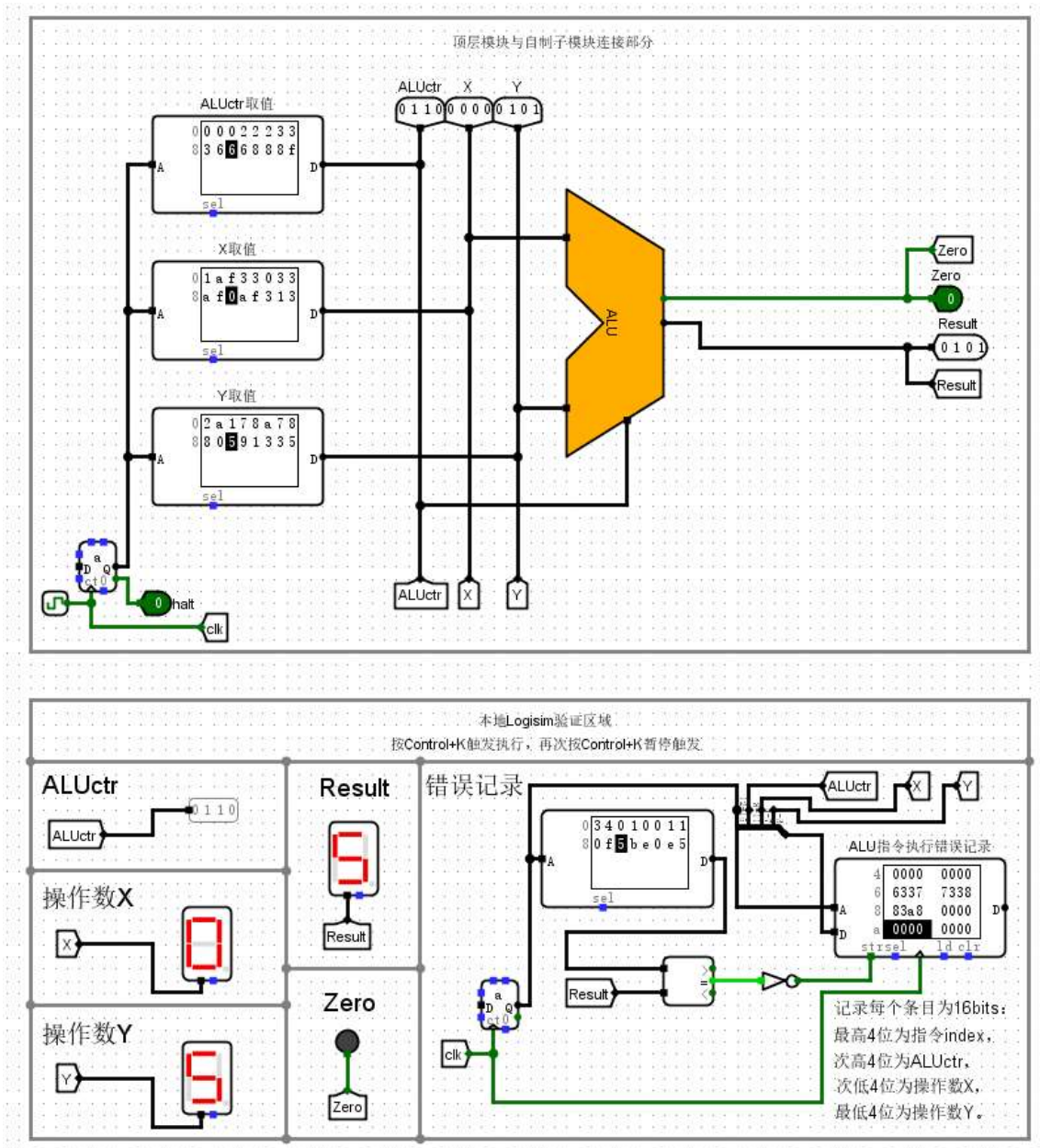


2. 基于带标志位加法器实现 ALU.



(c) 仿真验证

按下 Ctrl + K 开始本地仿真, 结果良好.



(d) 实验结果

通过了头歌平台的验证.

1/1 全部通过

测试集1

消耗内存91.5MB 代码执行时长: 1.99秒

—— 预期输出 ——					—— 实际输出 ——				
0000	0001	0010	0	0011	0000	0001	0010	0	0011
0000	1010	1010	0	0100	0000	1010	1010	0	0100
0000	1111	0001	1	0000	0000	1111	0001	1	0000
0010	0011	0111	0	0001	0010	0011	0111	0	0001
0010	0011	1000	0	0000	0010	0011	1000	0	0000
0010	0000	1010	0	0000	0010	0000	1010	0	0000
0011	0011	0111	0	0001	0011	0011	0111	0	0001

四, 实验中遇到的问题和解决方法

1. 对于 Logisim 器件不熟悉

问题: 一开始不清楚异或门阵列和或门阵列要怎么实现.

解决办法: 只需要将门电路的数据位宽调为 4 即可.

问题: 分线器使用不当.

解决方法: 分线端口数和位宽均调为 4 即可, 这代表着 4 位编码器或 4 位译码器.

2. 未看清楚题目

题目中的红圈没看明白是什么意思, 实际上是取非的意思.