**中国科学技术大学计算机学院**

**《数字电路实验》报告**



实验题目：简单组合逻辑电路

学生学号：\_\_PB20111642\_\_\_

完成日期：\_\_2021.10.28\_\_\_

【实验题目】

Logisim 可以用来搭建数字逻辑电路并对其功能进行仿真，但是它的功能仅限于仿真，是对实际电路行为的模拟，而不是真真正正的电路。为了将设计的电路切实地在硬件上运行，我们还需要借助硬件描述语言、综合工具、硬件平台的支持。所以在本次实验中使用 Verilog 语言。

本次实验我们将进一步使用 Logisim 设计组合逻辑电路，并使用 Verilog 语言对设计的电路进行描述。

【实验目的】

熟练掌握 Logisim 的基本用法

进一步熟悉 Logisim 更多功能

用 Logisim 设计组合逻辑电路并进行仿真

初步学习 Verilog 语法

【实验环境】

PC 一台,能流畅的连接校园网

Logisim 仿真工具

vlab.ustc.edu.cn (jre、Logisim 工具以及 Verilog 语法介绍都可在此网站获取)

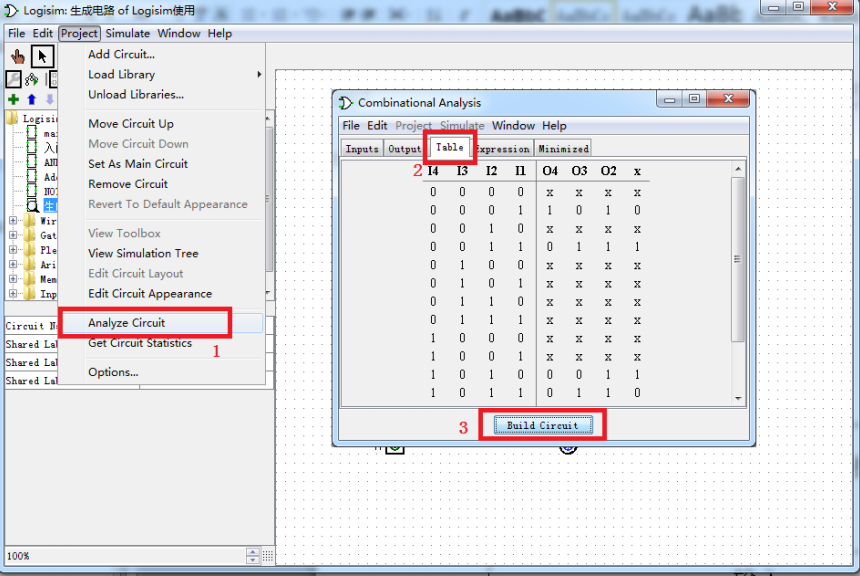
【实验过程】

**第一步：用真值表自动生成电路**

设计电路的一般做法是：1.根据真值表画出各输出项的卡诺图 2.通过卡诺图写出各输出项的逻辑表达式 3.根据逻辑表达式画出电路图，完成电路设计

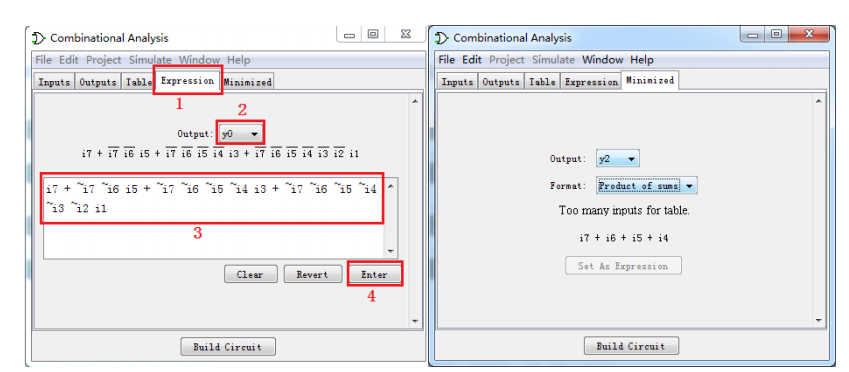
在 Logisim 新建一个电路图然后再电路图中放置输入引脚，有几个输入就放几个引脚，按同样的方式放置输出引脚。放置完毕后，给所有引脚标上标号，并按高低位顺序排列。

在菜单栏的“Project”选项卡中找到“Analyze Circuit”选项， 并选中。在弹出的窗口中选择“Table”选项，按照前面的真值表修 改输出值（鼠标点击输出信号对应的叉号就可修改），最后点击“Build Circuit”便可生成电路（弹出的对话框都选择“是”）。

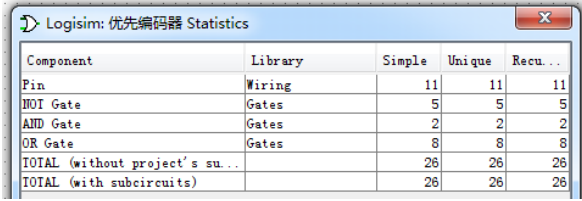


**第二步：用表达式生成电路图**

我们可以在 Logisim 中直接输入表达式生成电路，“Project” --> “Analyze Circuit”的弹出窗口中选“Expression”选项， 填入每个输出信号的表达式。最后点击“Build Circuit”生成电路。 有时候手动输入的表达式并不是最简形式，最终生成的电路也会占用 较多的逻辑门，我们可以借助“Minimized”选项卡对表达式进行简化，进而减少电路使用的逻辑门数量，电路输入信号不多的情况下，该窗口还能显示卡诺图。



我们还可以通过“Project”--> “Get Circuit Statistics”选项统计电路的基本信息。



Logisim 的自动生成电路功能，能为用户带来便利，节省大量时间，但也有一点小小的不足，其输入输出信号必须是单 bit 位宽，对于多 bit 位宽的输入信号并不支持，需要将其拆分成多个单 bit 信号才可以。

**第三步：Verilog HDL 语法入门**

Verilog 模块的最基本结构。

*module 模块名(*

*输入端口声明,*

*输出端口声明);*

*内部信号声明<可选>;*

*逻辑描述（模块主体）*

*endmodule*

每个模块都是以关键字 module 开头，以 endmodule 结束。module 后面是模块名，括号内是输入输出信号的声明（任何一个有实际功能的电路都应该有输入输出，但也存在例外，比如后续讲到对电路进行仿真时，其仿真激励文件一般就没有输入输出信号）。如果模块功能较复杂的话，可能会用到一些中间信号，那就要在模块内部声明，此例中没有用到，所以没有声明。逻辑描述部分是每个模块的主体，用于描述该电路的行为特性，其语法还算简单，相信读者可以很容易就能看懂。这里用到了一个非常重要的关键字“assign”，该关键字放在逻辑表达式之前，用于表明后面是一条连续赋值语句，一般来说，对组合逻辑的赋值都可以使用该关键字实现。

此外，同很多编程语言一样，Verilog 中也可以有注释，单行注释以“//”开始，多行注释则使用“/\* *注释内容* \*/”，注释内容仅仅是为了增加代码可读性，不会对代码功能产生影响。

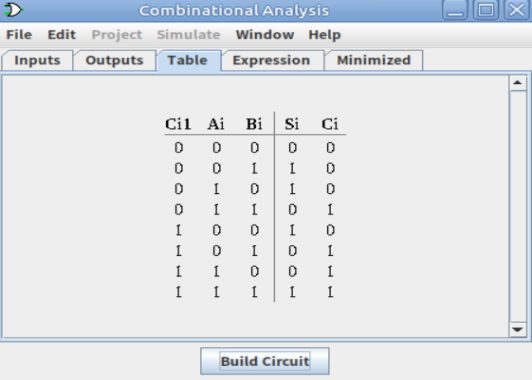
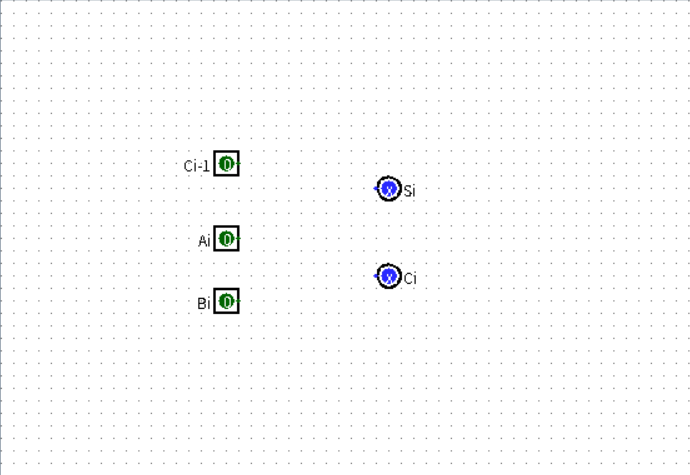
一段代码中有两条连续赋值语句，它们的顺序交换并不会对电路产生影响，换句话说，他们是位置无关的，并不是前面的先执行后面的后执行。

内部信号声明，关键字 wire 表明声明的信号为线网类型，对于这种信号类型，可以简单的理解为电路中的导线，可以通过assign 关键字进行赋值的信号都是这种类型，wire类型verilog中的默认类型，凡是没有明确声明类型的信号，都被当作 wire 类型处理。

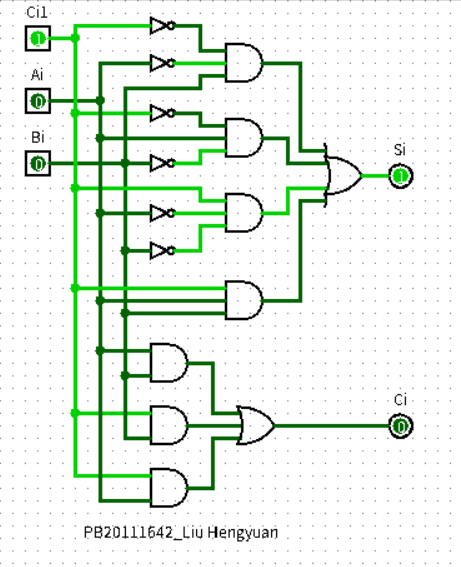
模块调用在 Verilog 中也非常重要，在电路较复杂时，我们需要将其分解成若干个子电路，最后再将子电路整合，或者复用第三方以及自己之前设计的功能模块时，都需要用到模块调用。

【实验练习】

**题目一：**

根据真值表，按照第一步指示进行操作。 

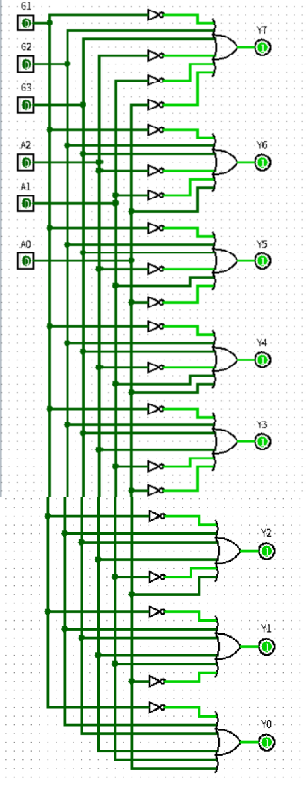
得到的电路图如下：



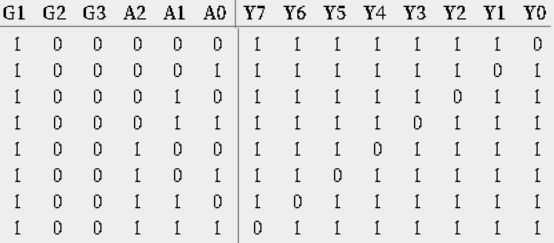
经检验，该图符合题目要求。

**题目二：**

根据真值表，按照第二步指示进行操作。得到电路图如下：

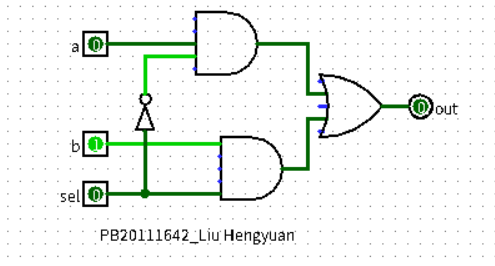


根据构建电路图后生成的真值表，验证电路图正确。



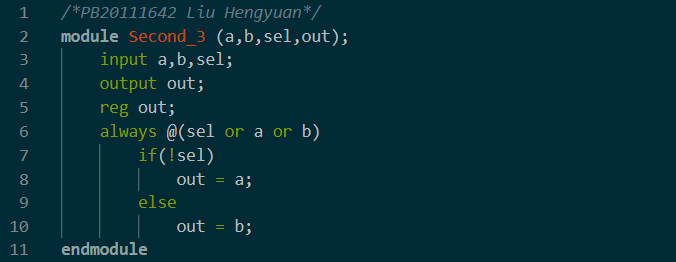
**题目三：**

构建一个二选一选择器，如图所示。



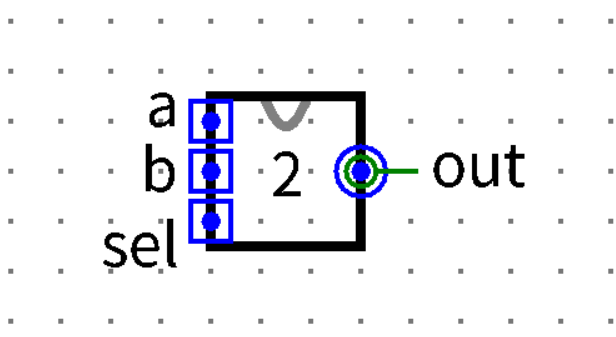
（当sel = 0时，选择a）

代码如下：

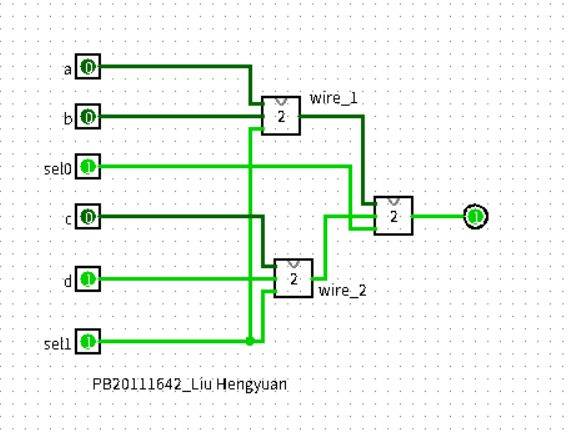


**题目四：**

例化二选一选择器并封装后，得到如图所示封装电路。



用二选一选择器构建四选一选择器，电路图如下。



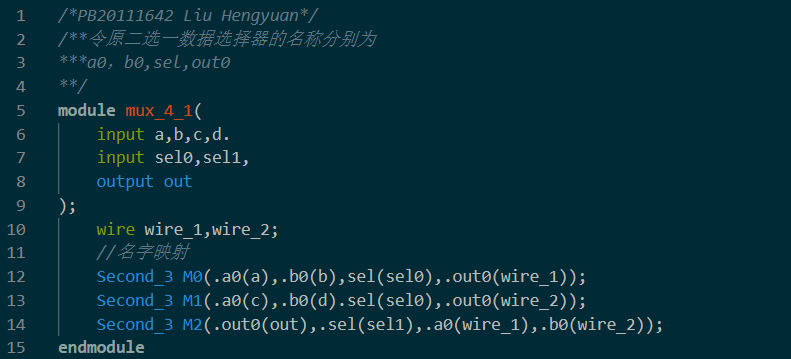
（当sel0 =sel1 = 0时，选择a

当sel0 = 0，sel1 = 1时，选择b

当sel0 = 1，sel1 = 0时，选择c

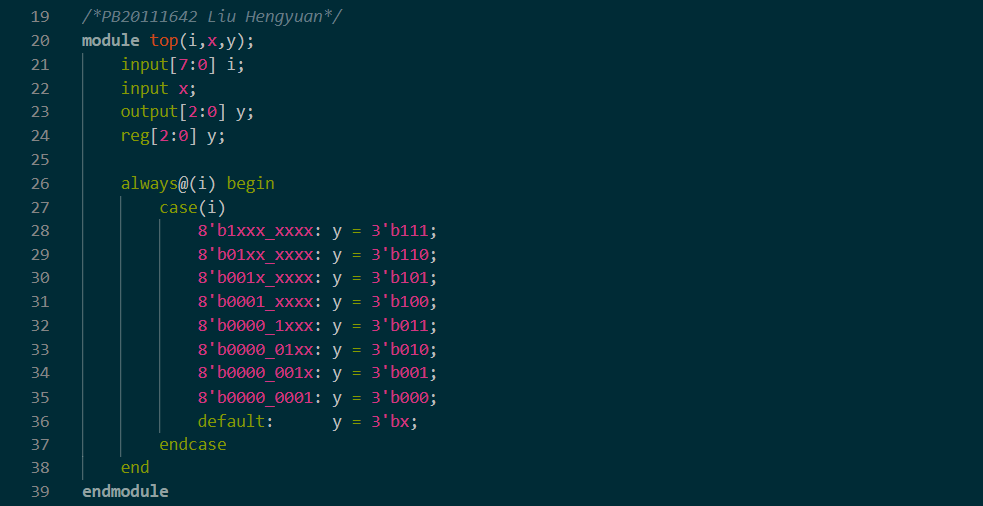
当sel0 = sel1 = 1时，选择d）

代码如下：



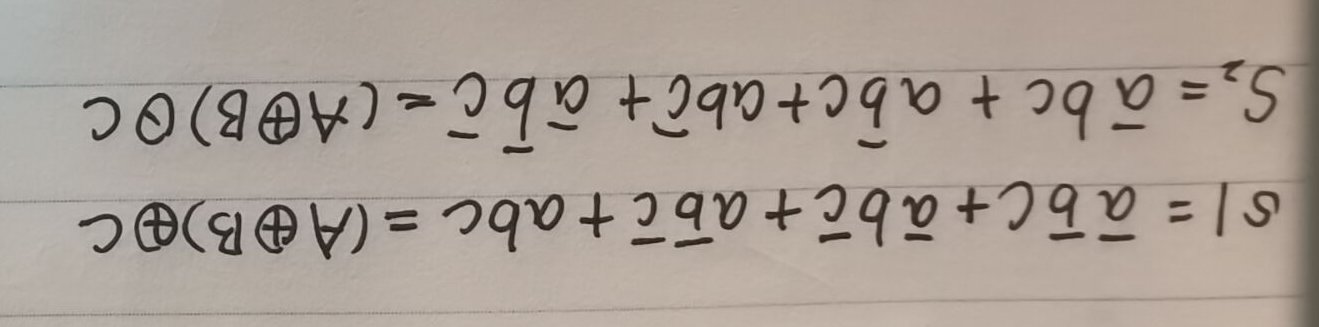
**题目五：**

根据八位优先编码器真值表，编写代码如下：

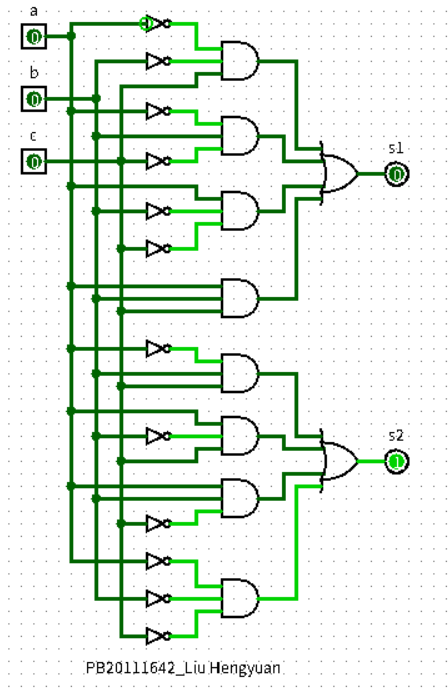


**题目六：**

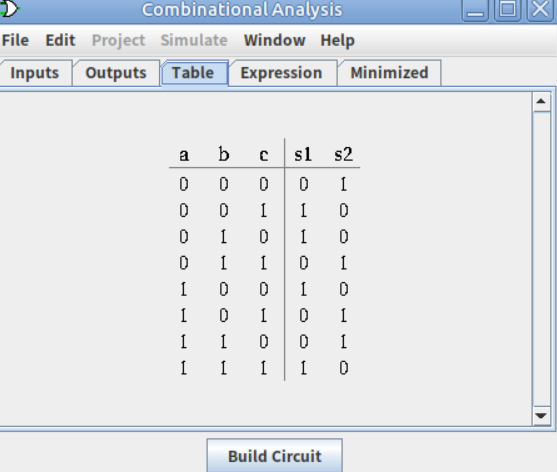
根据代码，列出逻辑表达式：

**

画出电路图如下。（运用方法表达式生成电路图）



通过分析真值表可得：



当输入为偶数个1时，s2为1，s1为0；当输入为奇数个1时，s1为1，s2为0。

【总结与思考】

通过本次实验，我学到了初步的Verilog编写方法，并成功地进行了一些基础电路的实践。对于Logisim以及Verilog的操作更加熟练。本次任务难度适中，任务量尚可。