**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

学生姓名：\_\_\_\_ 黄瑞轩\_\_\_\_\_

学生学号：\_\_ PB20111686\_\_\_

完成日期：\_\_ 2021.10.19 \_\_

计算机实验教学中心制

2020年09月

**实验题目**

简单组合逻辑电路

**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 HDL基本语法

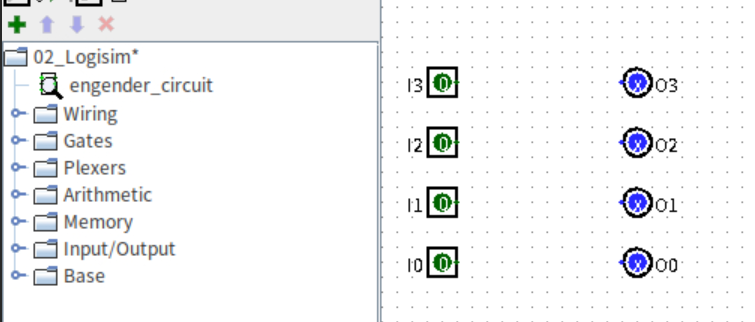
**实验环境**

* PC一台：安装了Linux操作系统的一台虚拟PC
* VLAB实验中心平台（vlab.ustc.edu.cn）
* Logisim Version 2.7.1

**实验过程**

1. 用真值表自动生成电路

* 设计电路一般的做法是：根据真值表画出各输出项的卡诺图；通过卡诺图写出各输出项的逻辑表达式；根据逻辑表达式画出电路图，完成电路设计。
* Logisim具有通过真值表自动生成电路的功能，可以帮助我们完成大部分上述步骤。
* 新建一个电路文件，并按需拖入引脚（输入和输出）。



所需要的功能在“Project”—“Analyze Circuit”—“Table”选项，按照指导手册上要求的真值表修改输出值，再点击“Build Circuit”即可生成。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 输入 | 输出 | | 0001 | 1010 | | 0011 | 0111 | | 1010 | 0011 | | 1011 | 0110 | | 1111 | 0101 | |  |  |

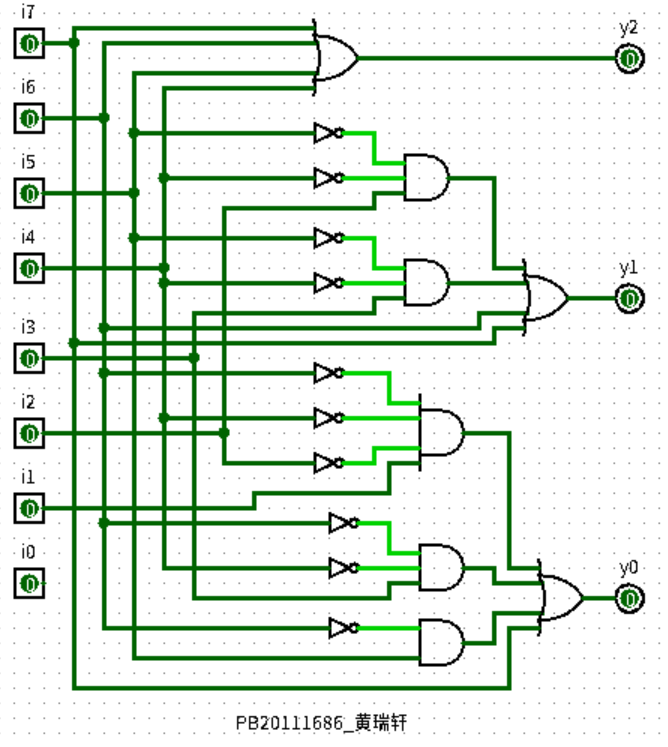
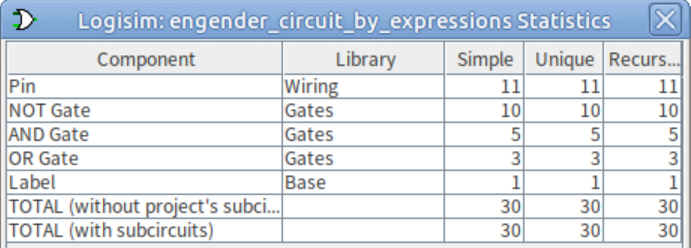
* 用真值表中的测试数据进行输入，验证所得电路的正确性。

1. 用表达式生成电路图

* 真值表条目数与输入项个数呈指数相关，当输入信号数量较多时，编辑真值表也非常麻烦，而Logisim可以直接通过输入的表达式生成电路。
* 选项位置：“Project”—“Analyze Circuit”—“Expression”，填入每个输出信号的表达式，点击“Build Circuit”生成电路。输入指导手册上的示例表达式作为尝试，生成的电路如下右图所示。

|  |  |
| --- | --- |
|  |  |

* 有时候手动输入的表达式不是最简，生成的电路会占用较多逻辑门，可以使用“Minimized”选项卡对表达式进行简化以减少电路使用的逻辑门数量。采用这种方法后的电路如下左图所示，可以看到电路简化了不少。

* 通过“Project”—“Get Circuit Statistics”可以统计电路的基本信息。上面这个电路的统计信息如上右图所示。
* 注意：Logisim的自动生成电路功能，其输入输出信号必须是单bit位宽，不支持多bit位宽的输入信号，要用到后者时需要将其拆分成多个单bit信号。

1. Verilog HDL语法入门

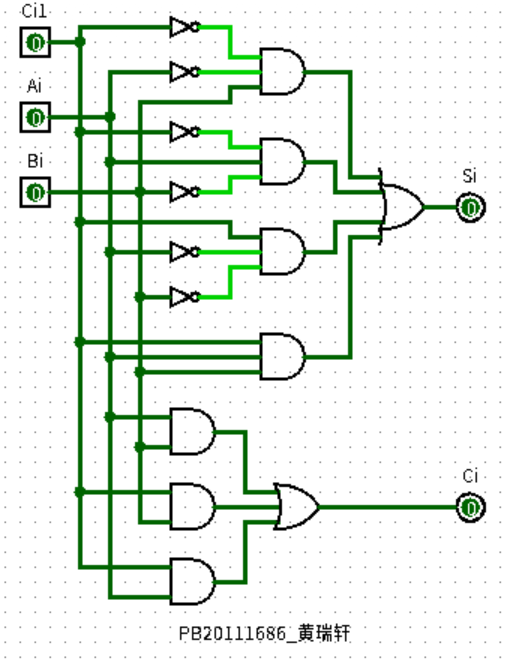
* Verilog中有一些关键字，这些关键字有特定含义和专门的用途，不能用来作为信号名或模块名。
* 每个模块都以module开头，以endmodule结束。module后面是模块名，括号内是输入输出信号的声明。
* 如果模块功能较复杂的话，可能会用到一些中间信号，需要在模块内部声明。
* Verilog中也可以注释，单行注释以“//”开始，多行注释则使用“/\* 注释内容 \*/”。
* 两条连续赋值语句的顺序并不会对电路产生影响，并不是前面的先执行后面的后执行。
* wire声明的信号为线网类型，可以通过assign进行赋值的信号都是这种类型，wire类型是默认类型，凡没有明确声明类型的信号都被当作wire处理。

|  |  |
| --- | --- |
| 电路结构 | Verilog HDL代码 |
|  | module test(input in, output out, out\_n);  assign out = in;  assign out\_n = ~in;  endmodule |
|  | module add(input a, b, output sum, cout);  assign {cout, sum} = a + b;  endmodule |
|  | module full\_add(input a, b, cin, output sum, cout);  wire s, carry1, carry2;  add add1(.a(a), .b(b), .sum(s), .cout(carry1));  add add2(.a(s), .b(cin), .sum(sum), .cout(carry2));  assign cout = carry1 | carry2;  endmodule |

**实验练习**

**【题目1】**

* 利用实验过程(1.)中的方法生成的电路图如下。



**【题目2】**

* 根据实验指导手册上的真值表列出表达式如下：

Y7 = ~(G1 ~G2 ~G3 A2 A1 A0)

Y6 = ~(G1 ~G2 ~G3 A2 A1 ~A0)

Y5 = ~(G1 ~G2 ~G3 A2 ~A1 A0)

Y4 = ~(G1 ~G2 ~G3 A2 ~A1 ~A0)

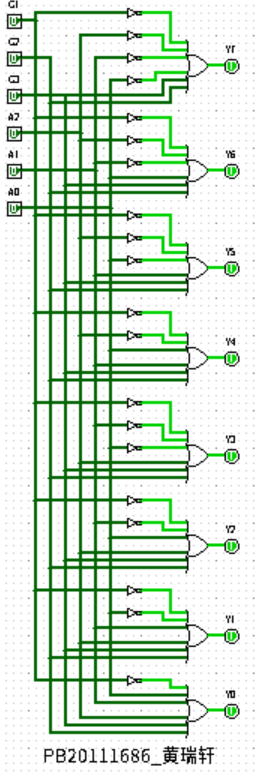
Y3 = ~(G1 ~G2 ~G3 ~A2 A1 A0)

Y2 = ~(G1 ~G2 ~G3 ~A2 A1 ~A0)

Y1 = ~(G1 ~G2 ~G3 ~A2 ~A1 A0)

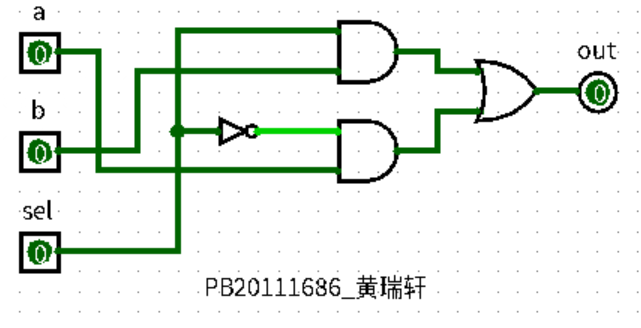
Y0 = ~(G1 ~G2 ~G3 ~A2 ~A1 ~A0)

* 利用实验过程(2.)中的方法生成的电路图如下。



**【题目3】**

* 按照题目要求设计的电路如下：



* Verilog代码如下：

module sel2\_1(input a, b, sel, output out);

assign out = (~sel & a) | (sel & b);

endmodule

**【题目4】**

* 所需要的四选一选择器的Verilog代码如下：

module sel4\_1(input a, b, c, d, sel0, sel1, output out);

wire o1, o2;

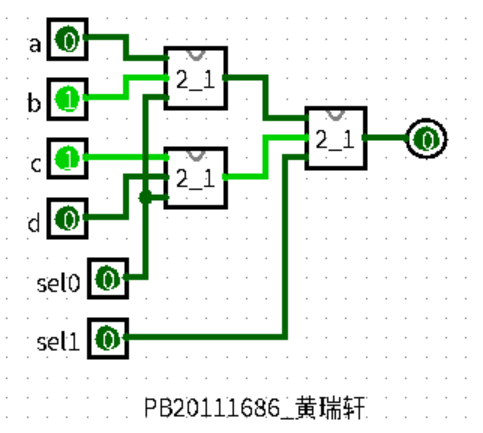
sel2\_1 S1(o1, a, b, sel0);

sel2\_1 S2(o2, c, d, sel0);

sel2\_1 S3(out, o1, o2, sel1);

endmodule

* 电路图如下：



**【题目5】**

* 根据真值表得到的逻辑式为：

y2 = i4 + i5 + i6 + i7

y1 = ~i5 ~i4 i2 + ~i5 ~i4 i3 + i6 + i7

y0 = ~i6 ~i4 ~i2 i1 + ~i6 ~i4 i3 + ~i6 i5 + i7

* Verilog代码为：

module encode8(input i7, i6, i5, i4, i3, i2, i1, i0, output y2, y1, y0);

assign y2 = i4 | i5 | i6 | i7;

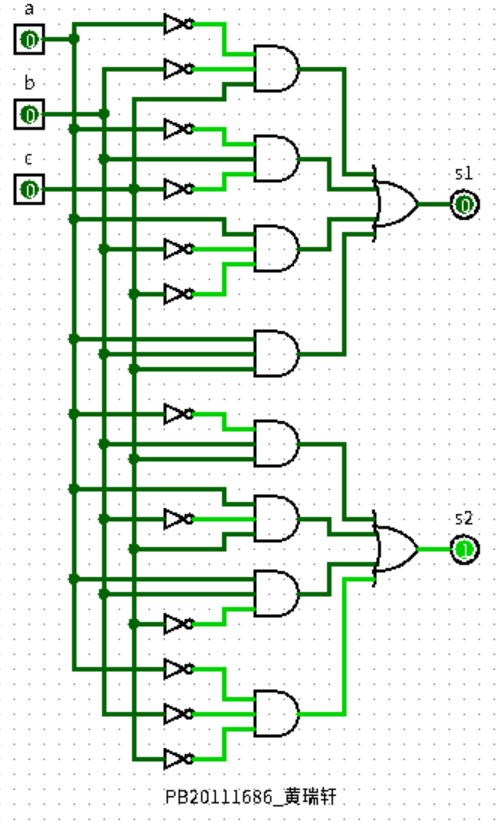
assign y1 = ~i5 & ~i4 & i2 | ~i5 & ~i4 & i3 | i6 | i7;

assign y0 = ~i6 & ~i4 & ~i2 & i1 | ~i6 & ~i4 & i3 | ~i6 & i5 | i7;

endmodule

**【题目6】**

* 其功能为：s1表示a, b, c三者作为单bit二进制数相加后最低位是否为1，s1 = 1表示是；s2表示a, b, c三者作为单bit二进制数相加后最低位是否为0，s2 = 1表示是。
* 对应的电路图如下：



**总结与思考**

* 本次实验使我了解了利用Logisim将真值表和逻辑表达式转换为电路的方法，减轻了未来使用其进行作业的任务量。知道了统计电路信息和查阅真值表的方法，减轻了检查电路正确性的工作量。并且学习了Verilog HDL语言的书写与使用。
* 本次实验练习均可以反复利用自动生成电路的功能，在第一次实验的基础上简化了设计电路的过程，使得练习做起来难度不大。
* 本次实验任务量不多，练习题难度逐渐递进，综合地应用了所学知识，使得实验不显得繁琐。关于本次实验任务量的建议将写在下一部分。
* 改进建议：

1. Verilog HDL语言的介绍和练习已经在理论课上完成，不需要再在实验中单独设置一个部分再来讲解，建议删除实验过程中“Verilog HDL语法入门”这一部分。
2. 题目6的功能分析部分与理论课重合，生成电路的部分与前面习题重合，徒增任务量，建议去掉或改成其他习题。