

目录

1	多数表决器	2
1.1	分析需求	2
1.2	Logisim, 启动!	2
1.3	测试	3
1.4	总结与心得	3
2	利用 CMOS 晶体管构建两输入或门	4
2.1	分析需求	4
2.2	Logisim, 启动!	4
2.3	测试	5
2.4	总结与心得	5
3	多路选择器及静态冒险检测	6
3.1	需求分析	6
3.2	Logisim, 启动!	6
3.3	冒险测试	7
3.4	总结与心得	7
4	利用传输门实现 2 路选择器	8
4.1	什么是传输门.jpg	8
4.2	Logisim, 启动!	8
4.3	使用组合电路分析功能	8
5	4 选 1 多路选择器	10
5.1	需求分析	10
5.2	Logisim, 启动!	10
6	思考题	10
6.1	将实验中设计的或门作为子电路应用到 2-1MUX-hazard 电路中	10
6.2	修改现有电路设计实现 4 位 4 选 1 多路选择器	11
6.3	设计并实现 4 位二进制数的奇偶校验位生成电路	12

1. 多数表决器

1.1 分析需求

3 输入多数表决器，即输入中含两个或两个以上为真时，输出值为真，否则为假

表 1: 3 输入多数表决器真值表

X	Y	Z	OUTPUT
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

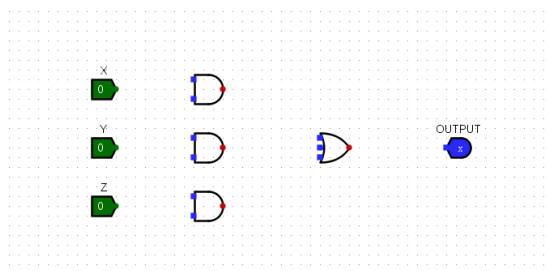
然后我们根据卡诺图可以写出最终化简后的表达式

$$OUTPUT = X \cdot Y + X \cdot Z + Y \cdot Z \quad (1)$$

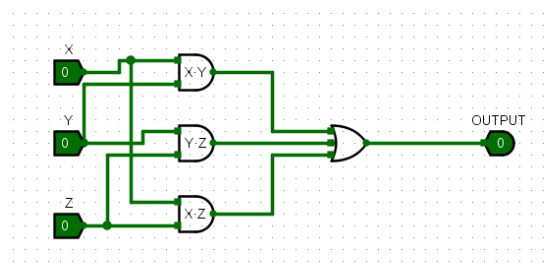
可知，电路第一级为与门，第二级为或门

1.2 Logisim，启动!

先选出需要的各组件，注意或门设置为三输入

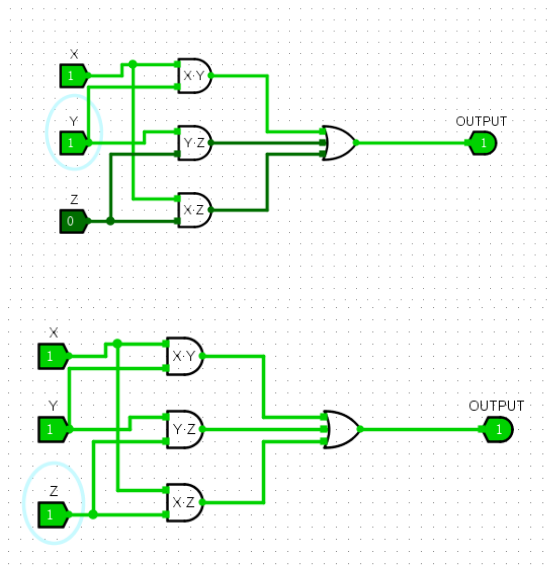


连接电路



1.3 测试

通过对 X, Y, Z 设置不同的输入值, 观察输出, 发现均满足真值表



1.4 总结与心得

这是 DLCO 这门课的第一个实验。花在入门和看说明文档的时间比完成实验本身的时间要久很多。也许这就是“学习成本”?

2. 利用 CMOS 晶体管构建两输入或门

2.1 分析需求

本题要求我们用晶体管构建两输入或门，容易联想到或门可用或非门和非门级联成或门。参考原理图：

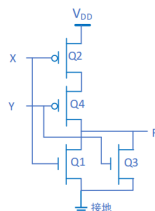


图 1.6 或非门原理图

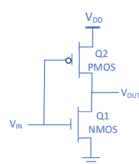
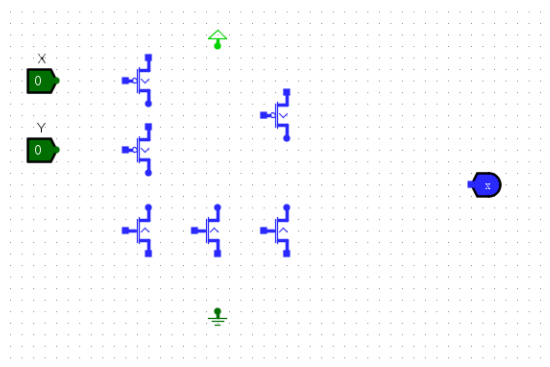


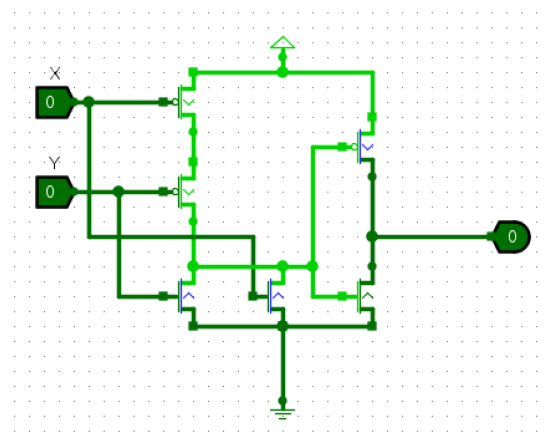
图 1.7 非门原理图

2.2 Logisim，启动！

准备好所需要的两个输入端，一个输出端，以及所需要的晶体管



连接电路

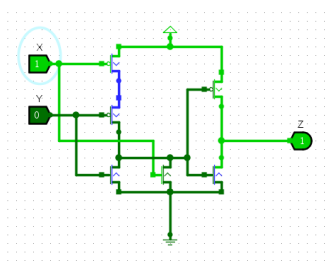


这里有一点小插曲，因为一开始没注意 NMOS 的方向导致电路总是报错，发现这个问题并修改之后电路可正常运行。

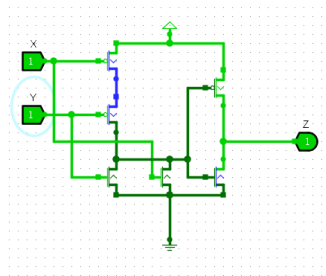
2.3 测试

表 2: 或门真值表

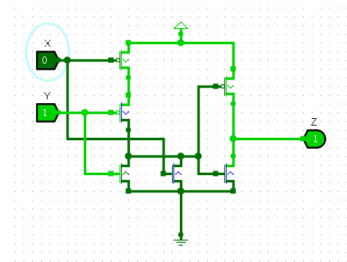
X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1



(a) 图 1



(b) 图 2



(c) 图 3

图 1: 测试结果

将仿真结果与真值表对照，符合预期。

2.4 总结与心得

注意细节!! (比如 CMOS 的方向)

3. 多路选择器及静态冒险检测

3.1 需求分析

实验要求给出的逻辑公式为：

$$Y = D0 \cdot \neg S + D1 \cdot S \quad (2)$$

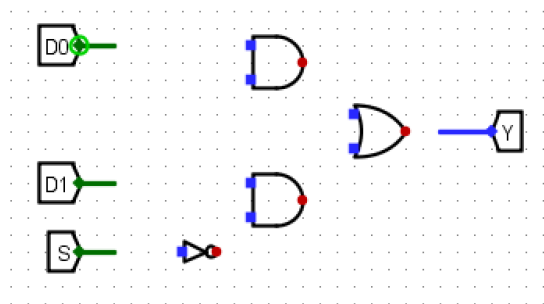
根据逻辑公式给出真值表：

表 3: 多路选择器真值表

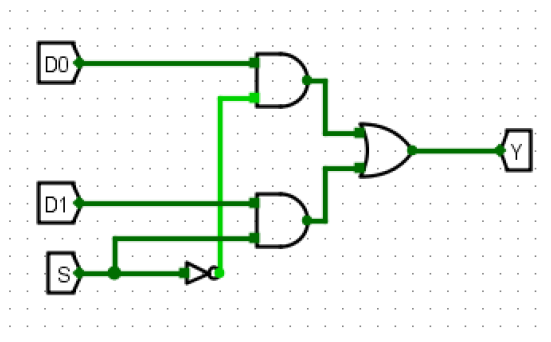
D0	D1	S	OUTPUT
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

3.2 Logisim，启动！

可知，我们需要三个输入引脚 D0，D1，S，需要两个与门，一个非门和一个或门。准备好的基础部件如图：

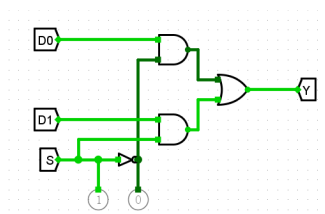


连接好电路

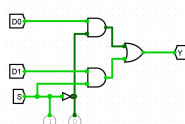
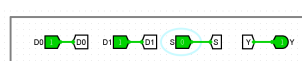


3.3 冒险测试

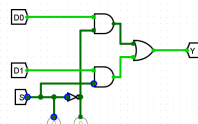
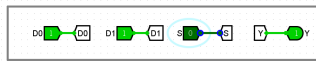
由实验指南有，我们应该在非门两端设置探针，并设置 $D0 = 1$, $D1 = 1$, $S = 1$, 观察结果：



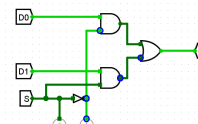
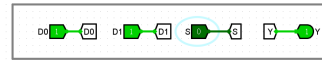
进行单步仿真：



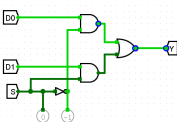
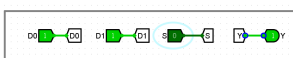
(a) 图 1



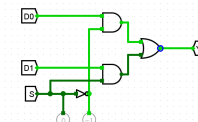
(b) 图 2



(c) 图 3



(d) 图 4



(e) 图 5

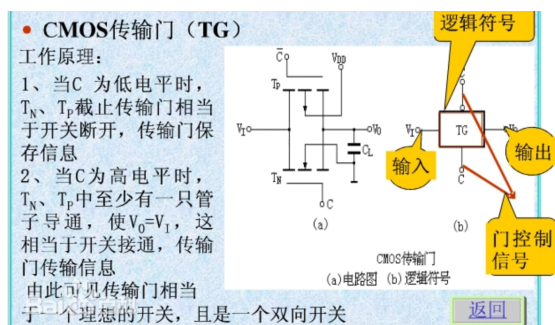
图 2: 测试结果

3.4 总结与心得

单步仿真类似 C++ 的断点调试逐步执行，探针可观察变量当前的值

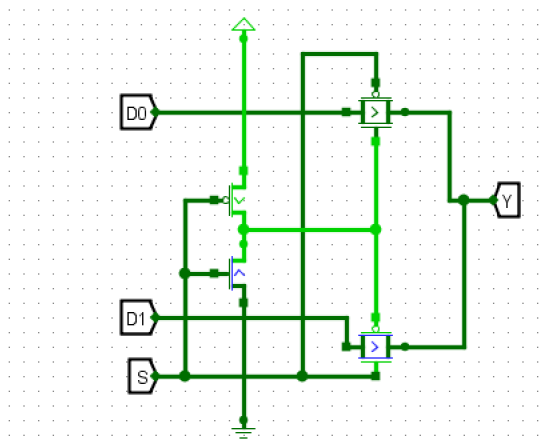
4. 利用传输门实现 2 路选择器

4.1 什么是传输门.jpg



4.2 Logisim, 启动!

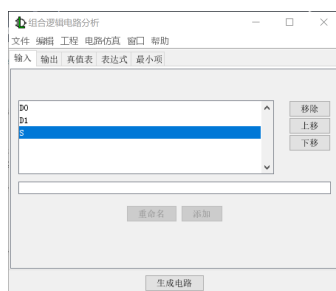
根据实验要求连接好电路图:



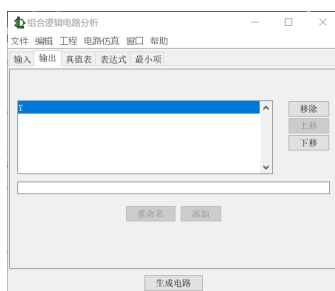
经测试, 所实现的电路满足要求

4.3 使用组合电路分析功能

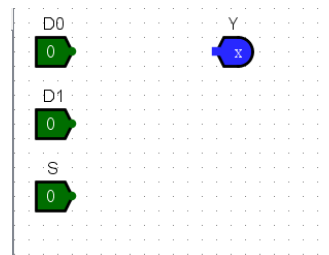
• 设置输入输出



(a) 图 1



(b) 图 2



(c) 图 3

图 3: 设置输入输出

• 设置真值表

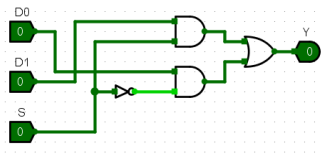
组合逻辑电路分析

文件 编辑 工程 电路仿真 窗口 帮助

输入 输出 真值表 表达式 最小项

D0	D1	S	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

生成电路



• 输入逻辑表达式

组合逻辑电路分析

文件 编辑 工程 电路仿真 窗口 帮助

输入 输出 真值表 表达式 最小项

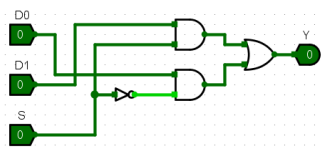
输出: Y

D1 S + D0 S

D1 S + D0 S

清空 恢复 输入

生成电路



• 设置最小项列表

组合逻辑电路分析

文件 编辑 工程 电路仿真 窗口 帮助

输入 输出 真值表 表达式 最小项

输出: Y

格式: 积之和形式

D1, S

D0	D1	S	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

D1 S + D0 S

设为表达式

生成电路

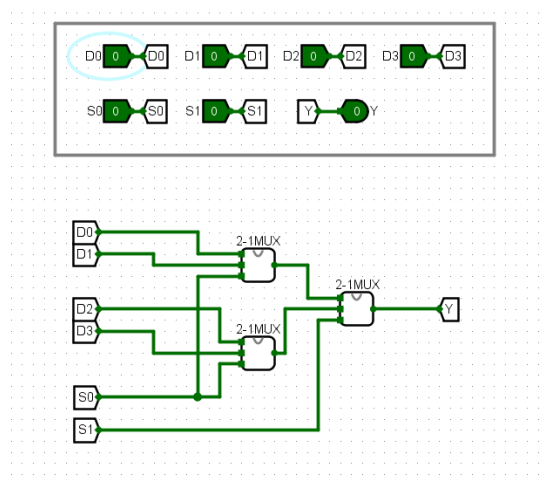
5. 4 选 1 多路选择器

5.1 需求分析

要求我们用 3 中实现的 2 选 1 多路选择器级联来实现 4 选 1 多路选择器。可知，输入端我们需要 4 个 D_i 引脚，两个 S_i 引脚用于发挥选择作用。

5.2 Logisim，启动！

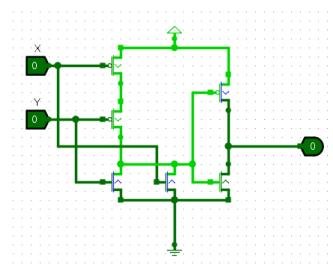
在 4-1MUX 中引入已经成功实现的 2-1MUX，并且对电路进行连接



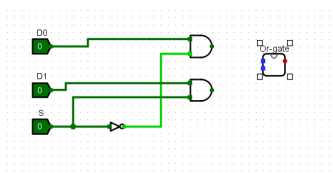
6. 思考题

6.1 将实验中设计的或门作为子电路应用到 2-1MUX-hazard 电路中

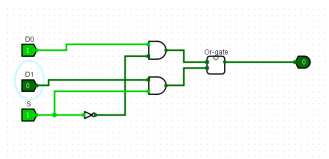
实现的或门



将或门引入 2-1MUX-hazard 电路

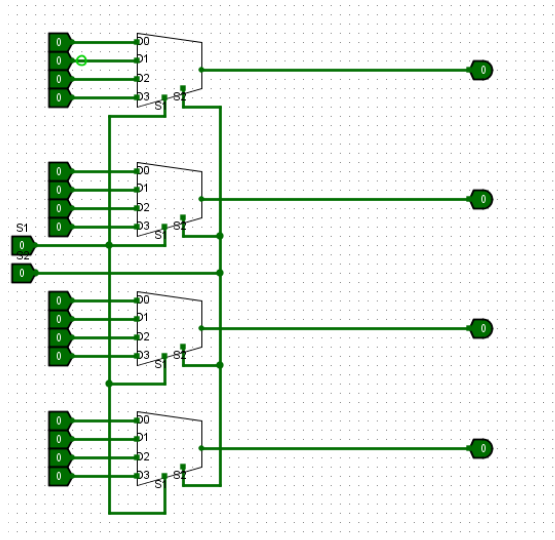


利用自己实现的或门代替提供的或门，并测试，电路工作正常



6.2 修改现有电路设计实现 4 位 4 选 1 多路选择器

- **思考：**通过已经实现的四路选择器来实现，每个选择器输出四位中的一位



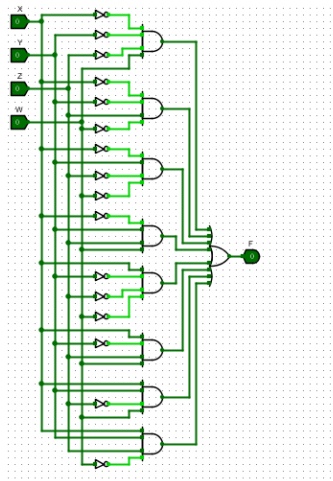
- **试错：**一开始试图直接通过修改位宽来实现，但是遇到了位宽不匹配的问题。于是转而采用这种使用四个选择器，每个选择器控制一位的输出的做法。

6.3 设计并实现 4 位二进制数的奇偶校验位生成电路

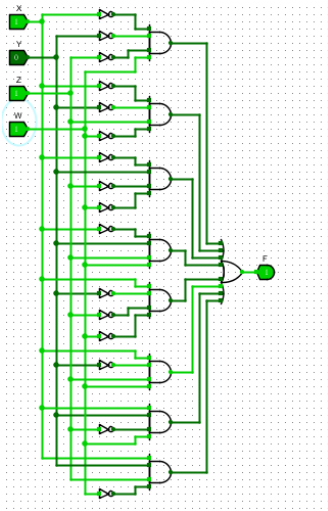
- 奇校验器 思考：有四位输入，当输入四位中含奇数个 1 时输出 1，含偶数个 1 时输出 0，于是我们可以画出真值表

XYZW	F
0000	0
0001	1
0010	1
0011	0
0100	1
0101	0
0110	0
0111	1
1000	1
1001	0
1010	0
1011	1
1100	0
1101	1
1110	1
1111	0

电路图如图



经过仿真测试，满足要求



- 偶校验器同理，此处不再赘述。