

浙江大学

本科实验报告

课程名称:	数字逻辑设计
姓 名:	王浩雄
学 院:	竺可桢学院
系:	混合班
专 业:	计算机科学与技术
学 号:	3230106032
指导教师:	马德

2025 年 3 月 20 日

浙江大学实验报告

课程名称：____数字逻辑设计____实验类型：____
实验项目名称：____变量译码器设计与应用____
学生姓名：____王浩雄____专业：____混合班____学号：____3230106032____
同组学生姓名：____无____指导老师：____马德____
实验地点：____紫金港东 4-509____实验日期：____2025 年 3 月 20 日____

一、实验目的和要求

- ① 掌握变量译码器的逻辑构成和逻辑功能；
- ② 用变量译码器实现组合函数；
- ③ 采用原理图设计电路模块；
- ④ 进一步熟悉 ISE 平台及下载实验平台物理验证。

二、实验内容和原理

1、实验背景

- ① **译码器**：将一种输入编码转换成另一种编码的电路，即将给定的代码进行“翻译”并转换成指定的状态或输出信号（脉冲或电平）；



- ② **74LS138**：一款广泛使用的 3-8 译码器集成电路，属于 TTL 逻辑系列。它通过 3 个二进制输入（A、B、C）的组合，将信号转换为 8 个独立的低电平有效输出（Y0-Y7），每个输出对应一种输入状态。此外，74LS138 还配备了 3 个使能端（G1、G2A、G2B），用于控制译码器的工作状态。当使能端有效时，输入信号决定哪个输出为低电平，其余输出保持高电平；若使能端无效，则所有输出均为高电平。74LS138 在数字电路中常用于地址译码、存储器选择和多路复用等场景。

2、实验内容

- ① 原理图设计实现 74LS138 译码器模块
- ② 用 74LS138 译码器实现楼道灯控制

三、 主要仪器设备

- ① 装有 ISE 14.7 的计算机 1 台
- ② SWORD 开发板 1 套

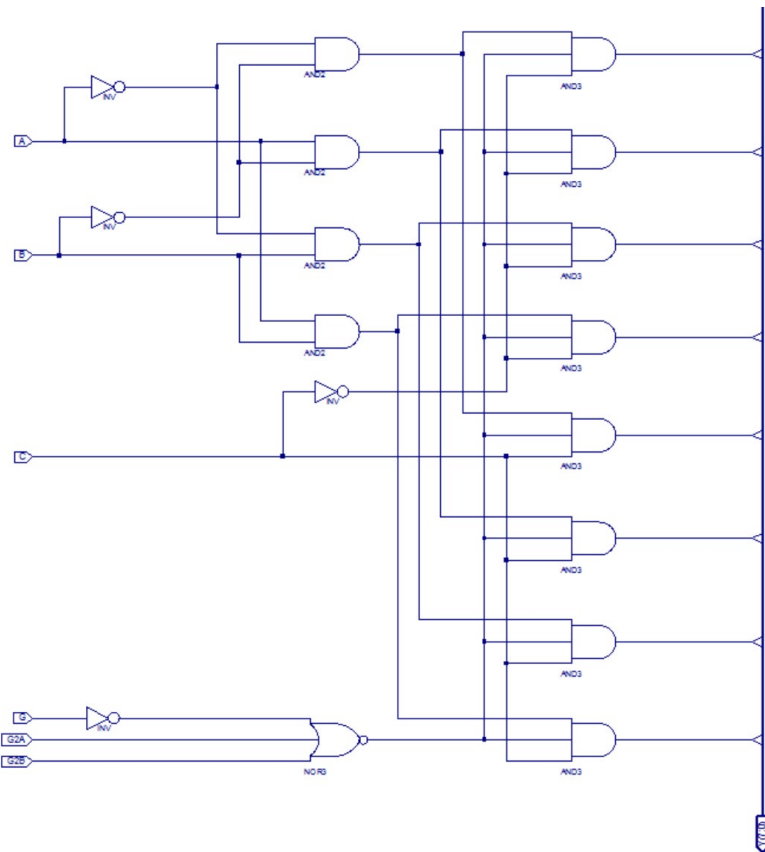
四、 操作方法与实验步骤

1、原理图设计实现 74LS138 译码器模块

- ① 结合设计要求，给出本器件的真值表如下：

输入		输出							
使能	变量								
GG _{2A} G _{2B}	CBA	Y ₀	Y ₁	Y ₂	Y ₃	Y ₄	Y ₅	Y ₆	Y ₇
X11	XXX	1	1	1	1	1	1	1	1
0XX	XXX	1	1	1	1	1	1	1	1
100	000	0	1	1	1	1	1	1	1
100	001	1	0	1	1	1	1	1	1
100	010	1	1	0	1	1	1	1	1
100	011	1	1	1	0	1	1	1	1
100	100	1	1	1	1	0	1	1	1
100	101	1	1	1	1	1	0	1	1
100	110	1	1	1	1	1	1	0	1
100	111	1	1	1	1	1	1	1	0

② 新建 Schematic 文件，结合设计要求和真值表，绘制电路连接图如下：



③ 由电路连接图生成的硬件描述代码如下：

```
module decoder_3_8(C, B, A, G, G2A, G2B, Y);
input wire A, B, C, G, G2A, G2B;
output wire [7:0] Y;

not node_0_0(A_n, A),
   node_0_1(B_n, B),
   node_0_2(C_n, C),
   node_0_3(G_n, G);

and node_1_0(D0, B_n, A_n),
   node_1_1(D1, B_n, A ),
   node_1_2(D2, B, A_n),
   node_1_3(D3, B, A );

nor node_1_4(EN, G_n, G2A, G2B);

nand node_2_0(Y[0], EN, D0, C_n),
      node_2_1(Y[1], EN, D1, C_n),
      node_2_2(Y[2], EN, D2, C_n),
```

```

        node_2_3(Y[3], EN, D3, C_n),
        node_2_4(Y[4], EN, D0, C ),
        node_2_5(Y[5], EN, D1, C ),
        node_2_6(Y[6], EN, D2, C ),
        node_2_7(Y[7], EN, D3, C );

endmodule

```

④ 新建仿真激励文件 sim.v 如下：

```

module sim;
    // Inputs
    reg C;
    reg B;
    reg A;
    reg G;
    reg G2A;
    reg G2B;
    // Outputs
    wire [7:0] Y;
    // Instantiate the Unit Under Test (UUT)
    decoder_3_8 uut (.C(C),.B(B),.A(A),.G(G),.G2A(G2A),.G2B(G2B),.Y(Y));

    integer i;
    initial begin
        // Initialize Input
        C = 0; B = 0; A = 0; G = 1; G2A = 0; G2B = 0;
        // Wait for global reset to finish
        #50;

        for (i=0; i<=7;i=i+1) begin
            {C,B,A} = i;
            #50;
        end

        assign G = 0;
        assign G2A = 0;
        assign G2B = 0;
        #50;
        assign G = 1;
        assign G2A = 1;
        assign G2B = 0;
        #50;
        assign G = 1;
        assign G2A = 0;
    end
endmodule

```

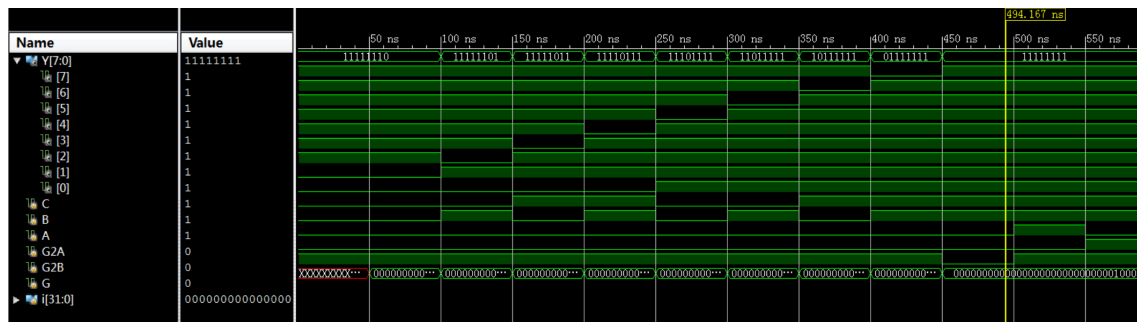
```

        assign G2B = 1;
        #50;
    end
endmodule

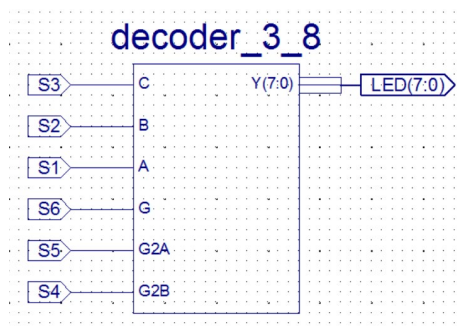
```

⑤ 运行仿真，得到如下的仿真波形。结合真值表（上文已给出）进行分析，该波形与预期相符，表明原理图绘制的正确性。

（注：由于显示问题，下图的变量名与波形没有对齐）



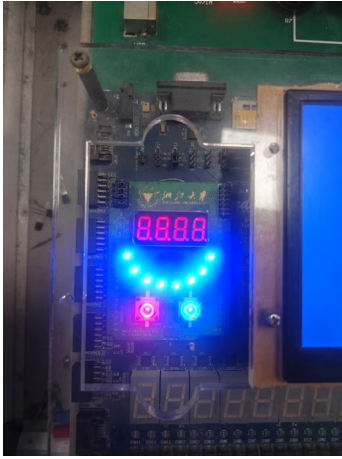
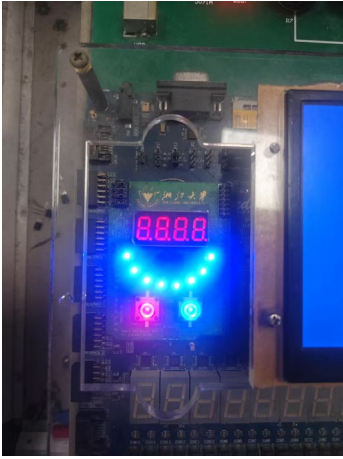
⑥ 新建项目，新建 Schematic 文件，绘制用拨盘开关控制模块的输入，用 LED(7:0) 作为模块的输出，验证模块的功能。

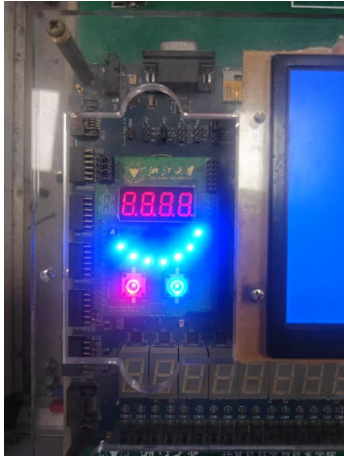

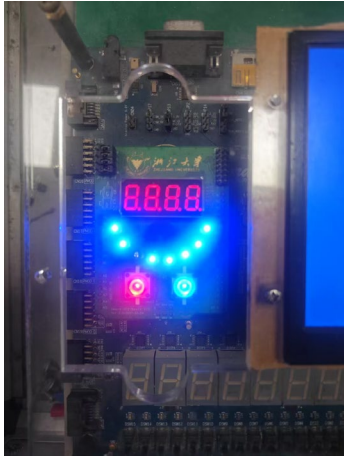

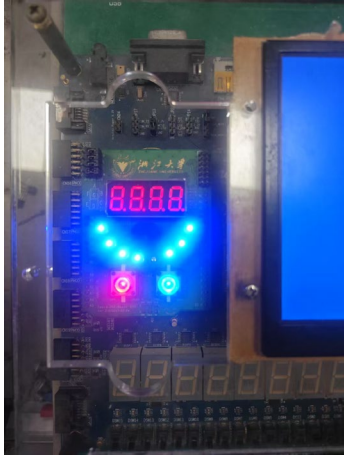
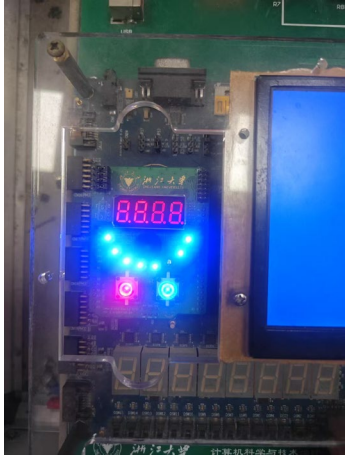


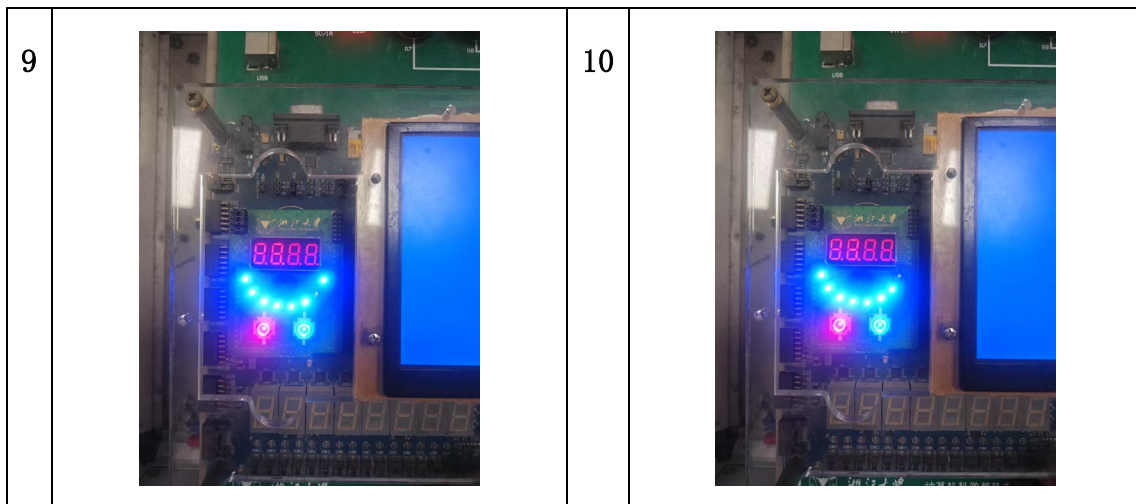
⑦ 下板，按真值表（上文已给出）进行验证。得到如下的结果。该结果与真值表和仿真波形相符。

测试组别	输入信号：信号名称（对应的拨盘开关号）						熄灭的 LED 灯编号
	A (S[0])	B (S[1])	C (S[2])	G2B (S[3])	G2A (S[4])	G (S[5])	
1	0	0	0	0	0	0	全亮
2	0	0	0	1	1	1	全亮
3	0	0	0	0	0	1	1
4	1	0	0	0	0	1	2
5	0	1	0	0	0	1	3
6	1	1	0	0	0	1	4
7	0	0	1	0	0	1	5
8	1	0	1	0	0	1	6
9	0	1	1	0	0	1	7
10	1	1	1	0	0	1	8
...	（其它测试组别均与预期相符，此处省略）						

测试照片（节选）：

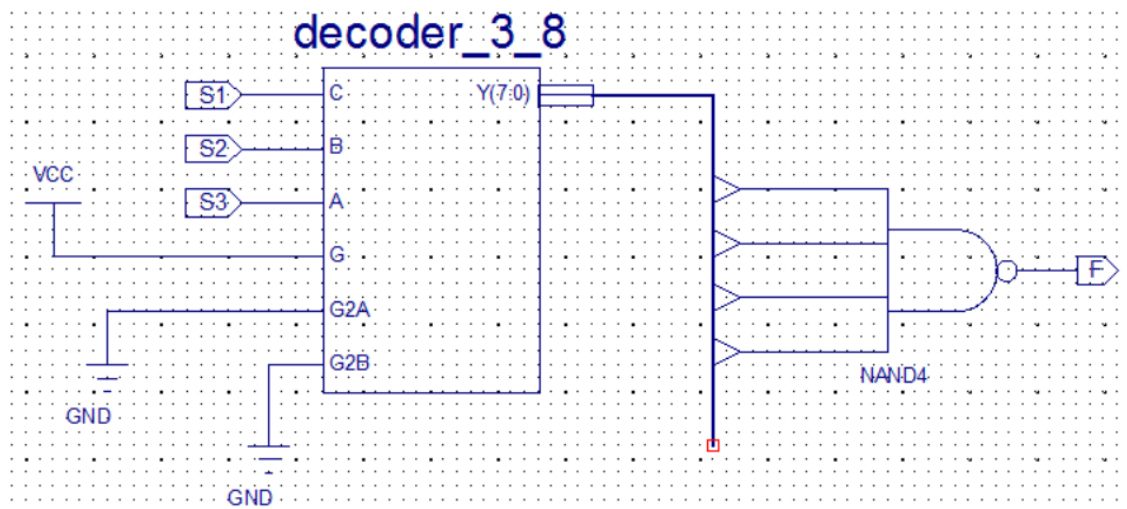
1		2	
---	---	---	--

3		4	
5		6	
7		8	

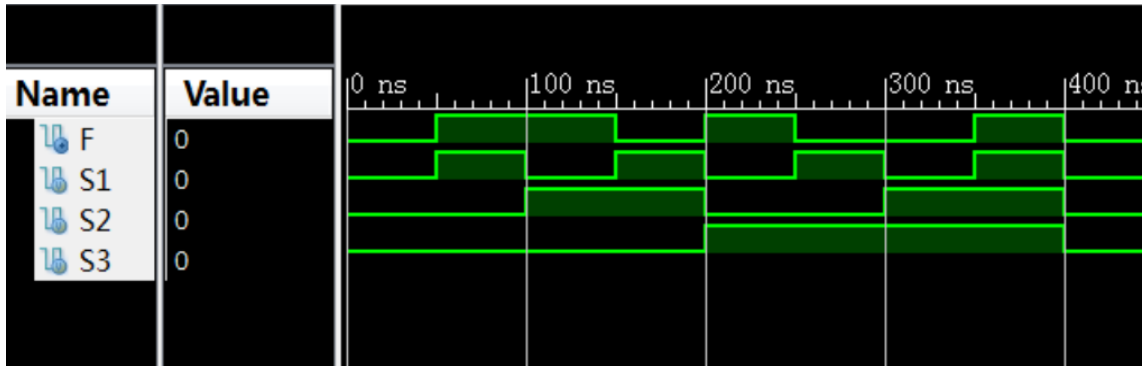


2、用 74LS138 译码器实现楼道灯控制

① 新建 Schematic 文件, 绘制电路连接图如下。其中, 接入 NAND4 的信号为 Y[1]、Y[2]、Y[4]、Y[7]。



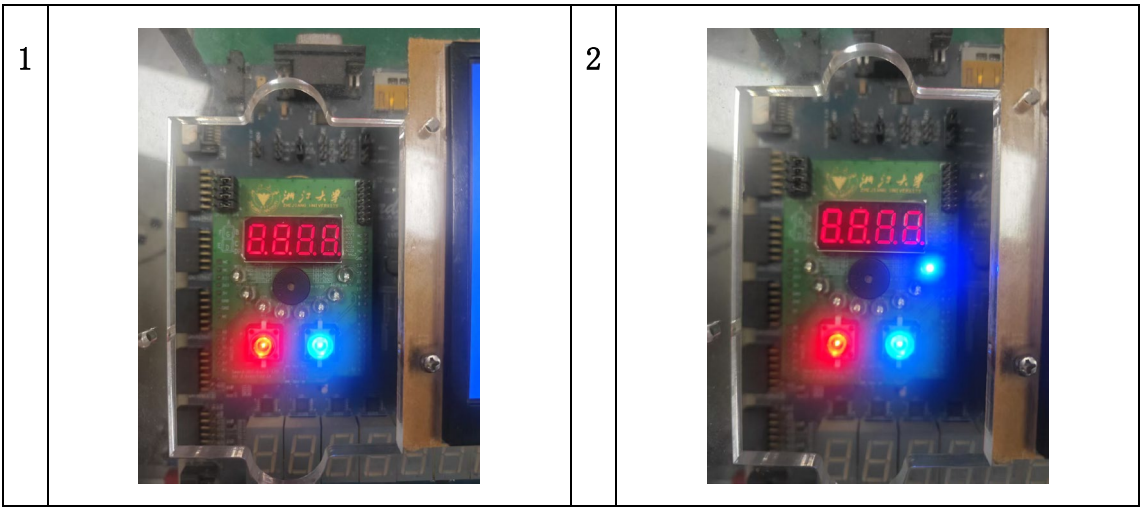
② 使用 Lab 4 的仿真激励文件, 运行仿真, 得到如下的仿真波形。该波形与预期相符, 其规律可概括为: 任意切换奇数次开关都会改变灯的开关状态, 而任意切换偶数次开关不会改变灯的开关状态。



③ 下板，按真值表（上文已给出）进行验证。得到如下的结果。该结果与真值表和仿真波形相符。

测试组别	开关 1	开关 2	开关 3	LED
1	断	断	断	灭
2	通	断	断	亮
3	通	通	断	灭
4	通	通	通	亮
...	(其它测试组别均与预期相符，此处省略)			

测试照片（节选）：



五、 实验结果与分析

（请见上方分析）

六、 讨论、心得

在本次实验中，我使用画图实现了 74LS138 译码器的设计，并成功通过仿真和上板验证的方式验证其正确性。同时，我在实验中初步接触了模块化设计的思想，将 74LS138 译码器封装为一个独立模块，并在其他工程中调用该模块，从而简化了开发流程，提高了设计的复用性和效率。