浙江大学

本科实验报告

课程名称：数字逻辑设计

姓　　名：颜晗

学　　院：计算机科学与技术学院

系：

专　　业：计算机科学与技术

学　　号：3200105515

指导教师：蔡铭

2021 年　　11　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：综合

实验项目名称：7 段数码管显示译码器设计与应用

学生姓名：颜晗　专业：计算机科学与技术　学号：3200105515.

同组学生姓名：熊儒海、吴俊贤　　指导老师：蔡铭

实验地点：东四 509　　实验日期：2021 年 11 月 1 日

一、实验目的和要求

1.掌握七数码管显示原理且掌握七段码显示译码设计

2.熟悉 Verilog 基础知识，学会使用 Verilog 语言设计硬件模块。

3.进一步熟悉 Xilinx ISE 环境及 SWORD 实验平台

二、实验内容和原理

实验内容：

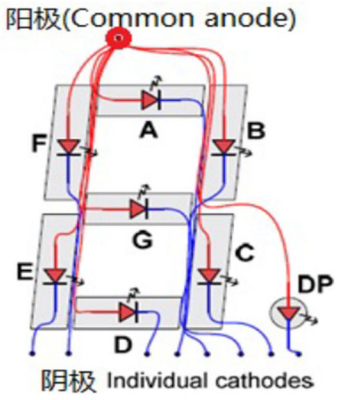
任务一：原理图或 Verilog 设计实现显示译码 MyMC14495 模块；

任务二：用 MyMC14495 模块实现数码管显示。

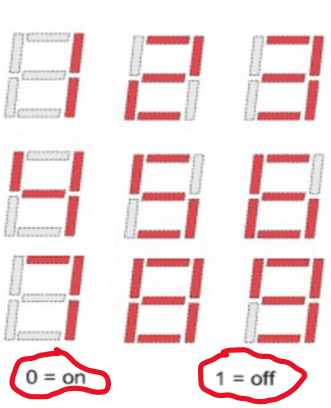
实验原理：

1. LED 数码管构造

由 7+1 个 LED 构成的数字显示器件，每个 LED 显示数字的一段，另一个为小数点。对应标志与控制信号如下：



X	a	b	c	d	e	f	g
0	0	0	0	0	0	0	1
1	1	0	0	1	1	1	1
2	0	0	1	0	0	1	0
3	0	0	0	0	1	1	0
4	1	0	0	1	1	0	0
5	0	1	0	0	1	0	0
6	0	1	0	0	0	0	0
7	0	0	0	1	1	1	1
8	0	0	0	0	0	0	0
9	0	0	0	0	1	0	0
A	0	0	0	1	0	0	0
B	1	1	0	0	0	0	0
C	0	1	1	0	0	0	1
D	1	0	0	0	0	1	0
E	0	1	1	0	0	0	0
F	0	1	1	1	0	0	0



2. 共阳（阴）控制

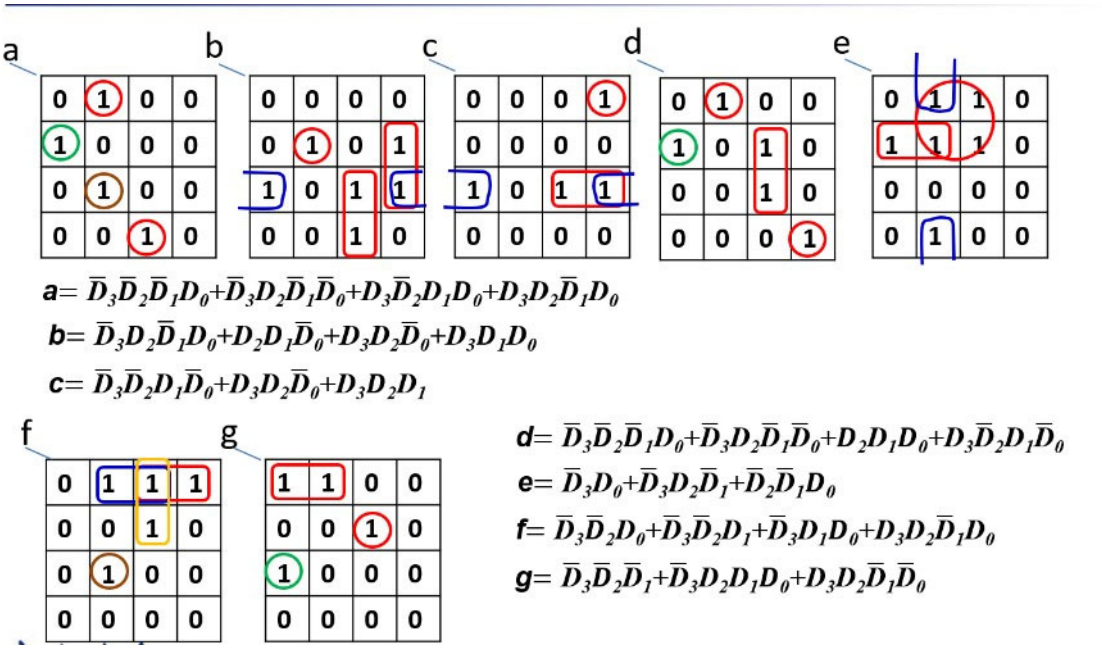
LED 的正极(负极)连在一起，另一端作为点亮的控制。

共阳：正极连在一起，负极＝0，点亮。

共阴：负极连在一起，正极＝1，点亮

3. 控制原理

通过 4-16 译码器与七段数码管的一一对应来实现数字的显示。经卡诺图优化后容易得到真值表并以此设计电路或代码。



$a = \bar{D}_3\bar{D}_2\bar{D}_1D_0+\bar{D}_3D_2\bar{D}_1\bar{D}_0+D_3\bar{D}_2D_1D_0+D_3D_2\bar{D}_1D_0$

$b = \bar{D}_3D_2\bar{D}_1D_0+D_2D_1\bar{D}_0+D_3D_2\bar{D}_0+D_3D_1D_0$

$c = \bar{D}_3\bar{D}_2D_1\bar{D}_0+D_3D_2\bar{D}_0+D_3D_2D_1$

$d = \bar{D}_3\bar{D}_2\bar{D}_1D_0+\bar{D}_3D_2\bar{D}_1\bar{D}_0+D_2D_1D_0+D_3\bar{D}_2D_1\bar{D}_0$

$e = \bar{D}_3D_0+\bar{D}_3D_2\bar{D}_1+\bar{D}_2\bar{D}_1D_0$

$f = \bar{D}_3\bar{D}_2D_0+\bar{D}_3\bar{D}_2D_1+\bar{D}_3D_1D_0+D_3D_2\bar{D}_1D_0$

$g = \bar{D}_3\bar{D}_2\bar{D}_1+\bar{D}_3D_2D_1D_0+D_3D_2\bar{D}_1\bar{D}_0$

4. 多位七段数码管显示原理

4.1 静态显示：每个 7 段码对应一个显示译码电。

4.2 动态扫描显示：时分复用显示

利用人眼视觉残留，一个 7 段码译码电路分时为每个 7 段码提供译码。

4.3 控制时序

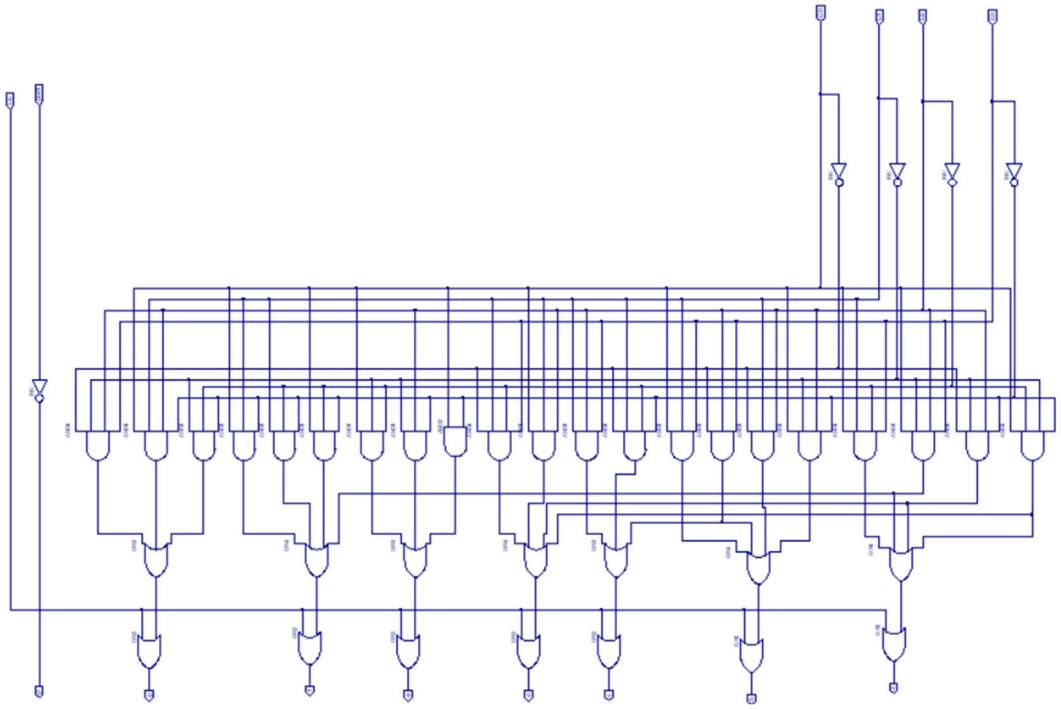
用定时计数信号控制公共极，分时输出对应七段码的显示信号：动态扫描。

注：实验板上的数码管有四组，实际上每次只能显示一组即一个数字，但是只要依次接入四组信号，LED 灯闪烁够快，就可以利用人的视觉残留表现出显示四个数字的效果。

三、实验过程和数据记录

任务一：原理图/Verilog 设计实现显示译码 MyMC14495 模块

1. 新建工程与源文件（schematic 即原理图设计，Verilog Module 即 Verilog 代码设计）
2. 画原理图或代码，原理图与代码如下：



注意：设计代码时要确认 LED 赋值的顺序。

```
module MyMC14495(
    input [3:0]D,
    input LE,
    input Point,
    output reg [6:0]LED,
    output reg p
    );

always @(*) begin
   if(LE==0) begin
      LED<=8'b1111111;
   end
   else begin
      case (D)
         4'h0: LED <= 7'b1000000;//0000001;
         4'h1: LED <= 7'b1111001;//1001111;
         4'h2: LED <= 7'b0100100;//0010010;
         4'h3: LED <= 7'b0110000;//0000110;
         4'h4: LED <= 7'b0011001;//1001100;
         4'h5: LED <= 7'b0010010;//0100100;
         4'h6: LED <= 7'b0000010;//0100000;
         4'h7: LED <= 7'b1111000;//0001111;
         4'h8: LED <= 7'b0000000;//0000000;
         4'h9: LED <= 7'b0010000;//0000100;
         4'hA: LED <= 7'b0001000;//0001000;
         4'hB: LED <= 7'b0000011;//1100000;
         4'hC: LED <= 7'b1000110;//1000010;
         4'hD: LED <= 7'b0100001;//0110000;
         4'hE: LED <= 7'b0000110;//0111000;
         4'hF: LED <= 7'b0001110;//0111000;
      endcase
   end
   p <= Point;
end
```

3. 检查设计是否有误，如果使用原理图设计，可查看 Verilog HDL 代码进行学习。
4. 对设计好的 Mc14495 模块进行仿真（原理图设计和 Verilog 设计的激励

代码不同），检查波形图。

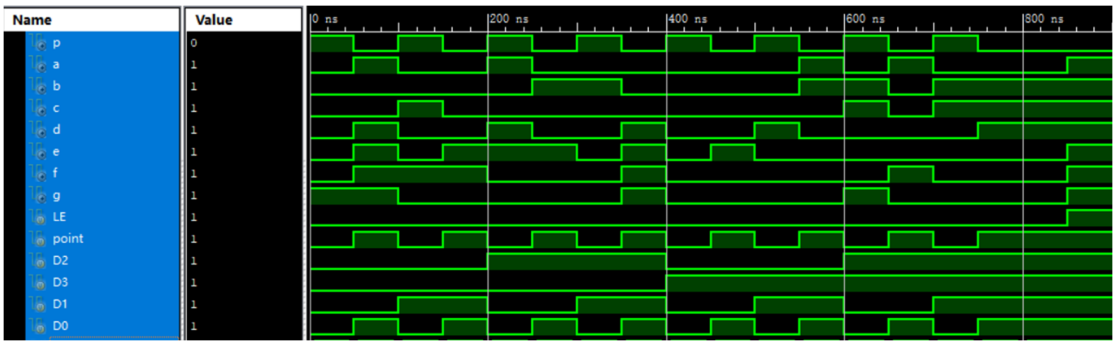
原理图激励代码：

```
integer i;
initial begin
        D3 = 0;
        D2 = 0;
        D1 = 0;
        D0 = 0;
        LE = 0;
        point = 0;
        for (i=0; i<=15;i=i+1) begin
                {D3,D2,D1,D0}=i;
                point = i;
                #50;
        end

        #50;
        LE = 1;
end
```

Verilog 激励代码：

```
module MC14495Test();
        // Inputs
        reg [3:0] D;
        reg LE;
        reg Point;
        // Outputs
        wire [6:0] LED;
        wire p;
        MC14495 uut (
                .D(D),
                .LE(LE),
                .Point(Point),
                .LED(LED),
                .p(p));
        integer i;
        initial begin
            D = 4'b0000;
            LE = 1;
            Point = 0;
            for (i=0; i<=15;i=i+1) begin
                D = i;
                Point = i;
                #50;
            end
            #50;
            LE = 0;
        end
endmodule
```

波形图：两者相同，只是要将 LED[6:0]与 a~g 对应起来。



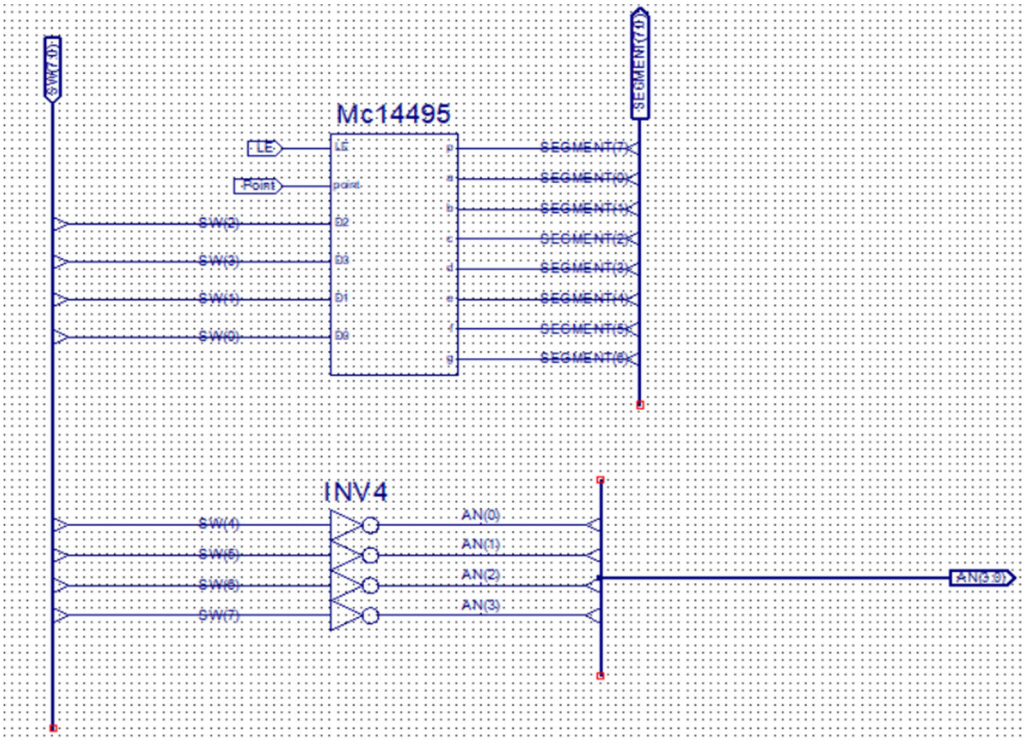
5. 生成对应的模块文件待下一任务使用。

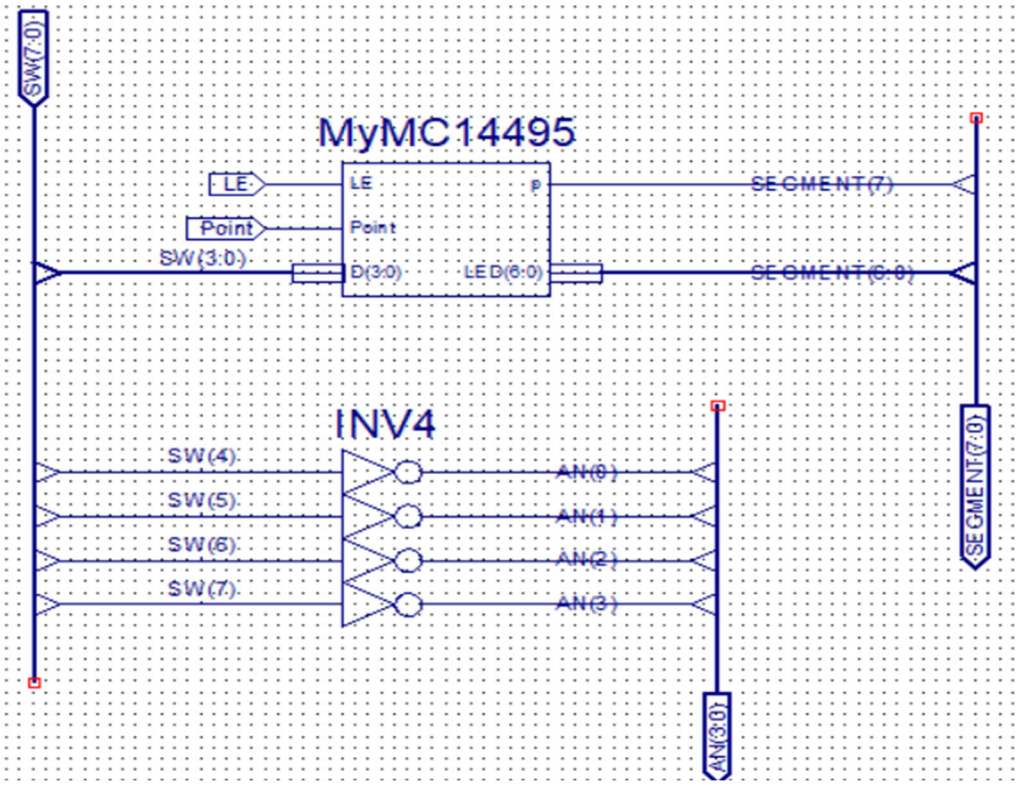
任务二：

1. 新建工程和 schematic 源文件。
2. 将任务一设计地 Mc14495 模块复制到该工程中（即.sym 和原理图的.vf 或

Verilog 的.v 文件）

3. 设计并绘制原理图，原理图如下（Mc14495 的设计方式不同，形式也有不同，原理图也有些许不同，注意导线的对应关系）





4. 创建引脚分配文件，文件内容如下：

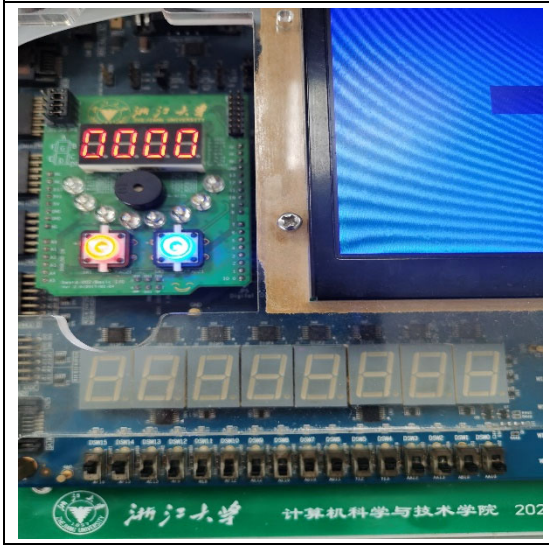
```
NET "SW[0]"          LOC = AA10   | IOSTANDARD = LVCMOS15;
NET "SW[1]"          LOC = AB10   | IOSTANDARD = LVCMOS15;
NET "SW[2]"          LOC = AA13   | IOSTANDARD = LVCMOS15;
NET "SW[3]"          LOC = AA12   | IOSTANDARD = LVCMOS15;

NET "SW[4]"          LOC = Y13    | IOSTANDARD = LVCMOS15; #AN0
NET "SW[5]"          LOC = Y12    | IOSTANDARD = LVCMOS15; #AN1
NET "SW[6]"          LOC = AD11   | IOSTANDARD = LVCMOS15; #AN2
NET "SW[7]"          LOC = AD10   | IOSTANDARD = LVCMOS15; #AN3


NET "point"          LOC = AF13   | IOSTANDARD = LVCMOS15 ;#SW[14]
NET "LE"             LOC = AF10   | IOSTANDARD = LVCMOS15 ;#SW[15]

NET "SEGMENT[0]"     LOC = AB22     | IOSTANDARD = LVCMOS33 ;#a
NET "SEGMENT[1]"     LOC = AD24     | IOSTANDARD = LVCMOS33 ;#b
NET "SEGMENT[2]"     LOC = AD23     | IOSTANDARD = LVCMOS33 ;
NET "SEGMENT[3]"     LOC = Y21      | IOSTANDARD = LVCMOS33 ;
NET "SEGMENT[4]"     LOC = W20      | IOSTANDARD = LVCMOS33 ;
NET "SEGMENT[5]"     LOC = AC24     | IOSTANDARD = LVCMOS33 ;
NET "SEGMENT[6]"     LOC = AC23     | IOSTANDARD = LVCMOS33 ;#g
NET "SEGMENT[7]"     LOC = AA22     | IOSTANDARD = LVCMOS33 ;#point

NET "AN[0]"          LOC = AD21     | IOSTANDARD = LVCMOS33 ;
NET "AN[1]"          LOC = AC21     | IOSTANDARD = LVCMOS33 ;
NET "AN[2]"          LOC = AB21     | IOSTANDARD = LVCMOS33 ;
NET "AN[3]"          LOC = AC22     | IOSTANDARD = LVCMOS33 ;
```

5. 检查所有文件无误后，上板查看实验结果。

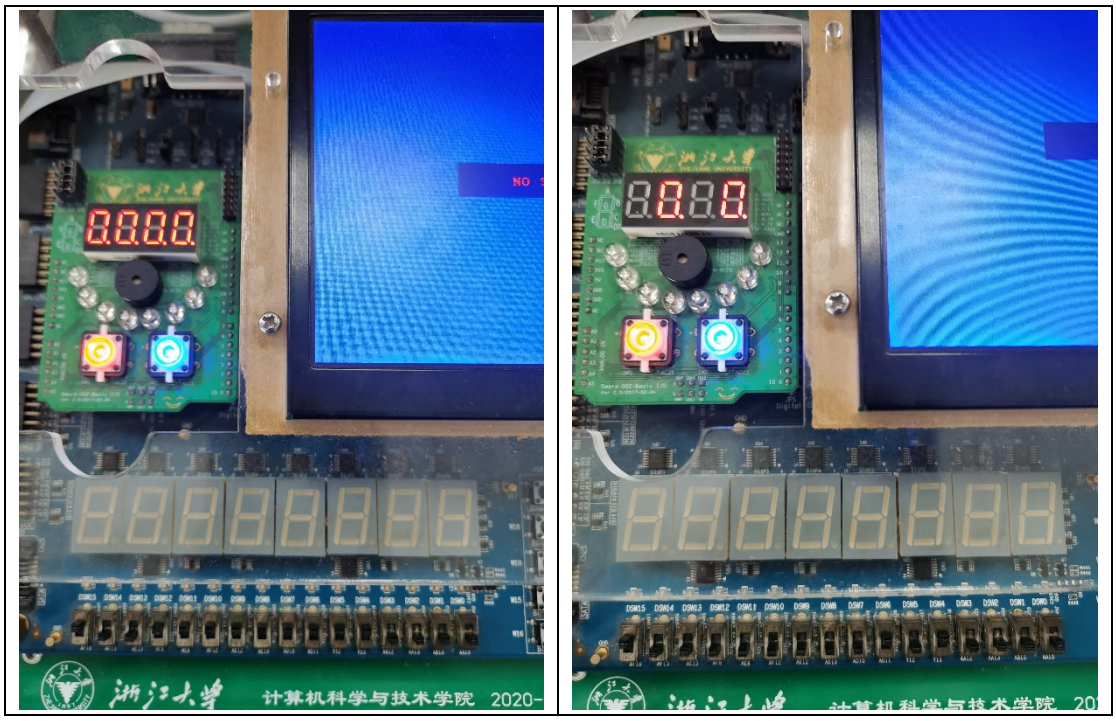
四、实验结果分析

经过上板测试后，无论是原理图还是 Verilog 设计都较好地完成了数码管地控制显示任务，取得了较好的结果。由于实验时对实验板和 Verilog 了解不足，导致两种设计方法的控制逻辑有所不同，但最终效果是相同的，以下是 Verilog 设计的结果。

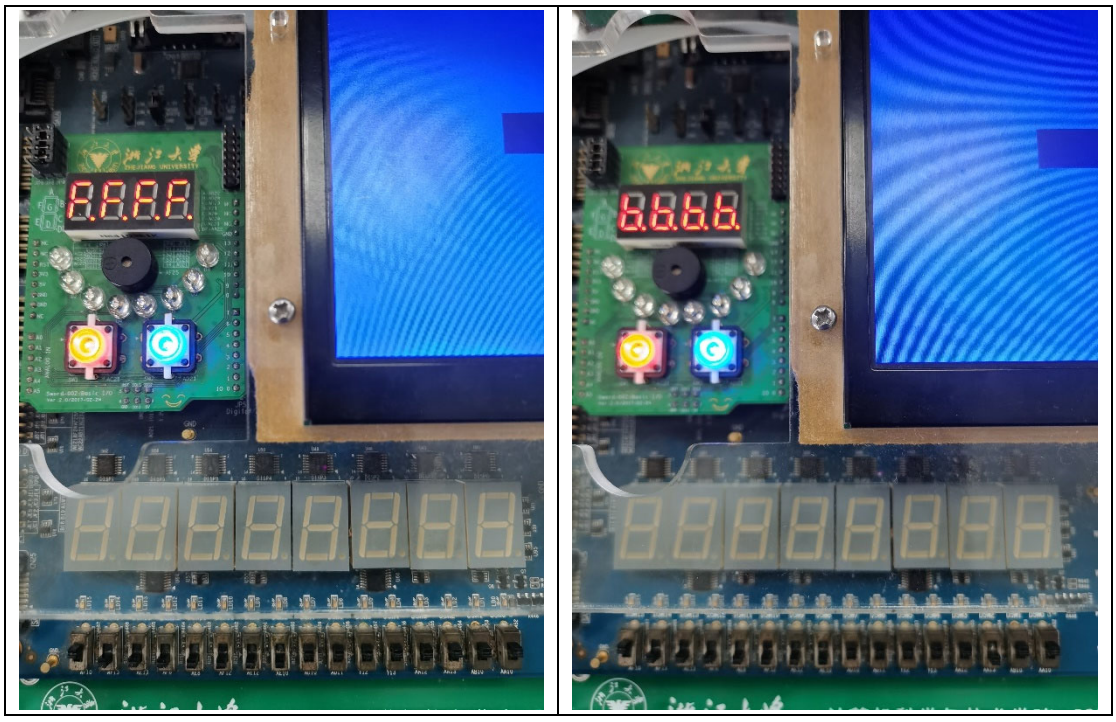
1. LE 输入控制，当 LE 为 0 时，LED 灯不显示。当 LE 为 1 且分别控制各组灯的开关也为 1 时，灯亮。

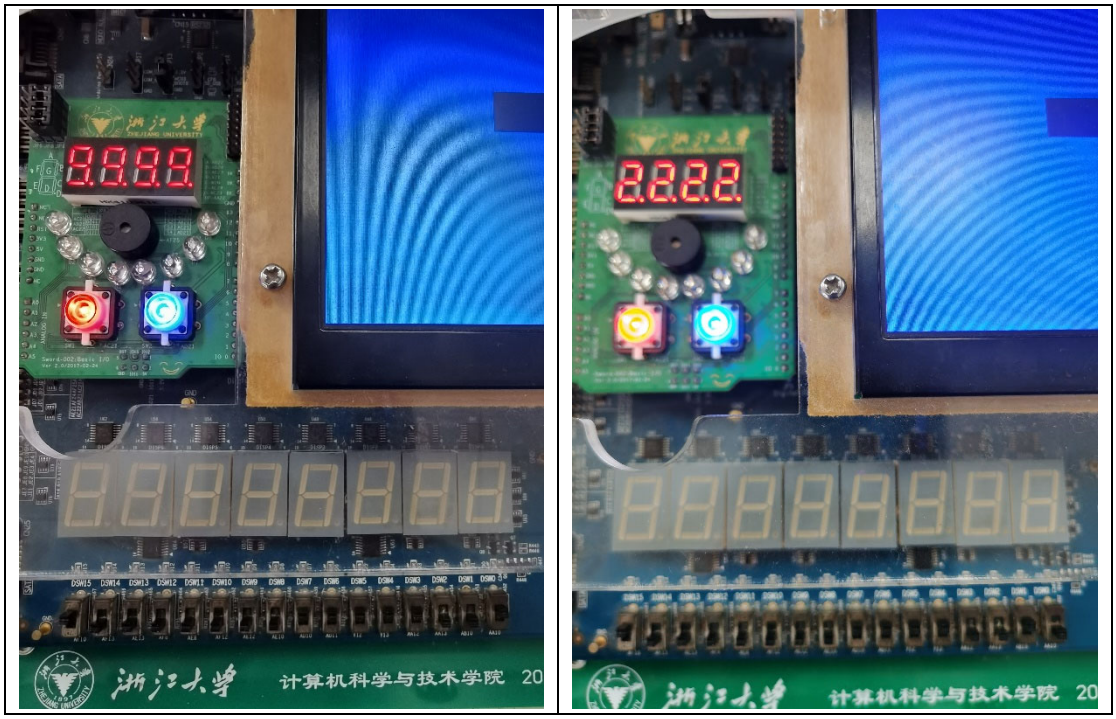
LE=1	LE=0
	

2. 数字的部分显示与点的显示



3. 数字控制





五、实验总结与反思

本次实验我们又设计了一种硬件模块，当然最重要的是第一次实践使用了 Verilog 代码，对比原理图和 Verilog 代码很容易就会发现 Verilog 的简洁性，减少了大量时间。当然 Verilog 代码在设计时也需要注意 Bug 问题，且由于并不直观可能更加难以找出错误。本次设计时由于多位变量赋值规则不了解，最初设计的电路显示的数字为镜像的，另外在输入不同情况时也容易串行导致错误，找问题时也花了不少时间。在设计时一定要确认电路的逻辑，认真检查代码。