个人形象照

李想 3180103542

实验 5—变量译码器设计与应用实验报告

姓名：李想　　学号：3180103542　　专业：软件工程

课程名称：逻辑与计算机设计基础实验　　同组学生姓名：　无

实验时间：2019-10-16　　实验地点：紫金港东 4-509　　指导老师：　洪奇军

一、实验目的和要求

1.1 掌握变量译码器的的逻辑构成和逻辑功能。

1.2 用变量译码器实现组合函数

1.3 掌握变量译码器的典型应用（地址译码的具体方法）

1.4 了解存储器编址的概念

1.5 采用原理图设计电路模块

1.6 进一步熟悉 ISE 平台及下载实验平台物理实验

二、实验内容和原理

2.1 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

2.2 实验原理

1、译码器定义及分类

译码器是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）。

译码可分为：**变量译码**、**显示译码**。

变量译码一般是将一种**较少位输入变为较多位输出**的器件，如 2n 译码和 8421BCD 码译码。

显示译码主要进行 **2 进制数显示成 10 进制或 16 进制数的转换**，可分为驱动 LED 和 LCD 两类

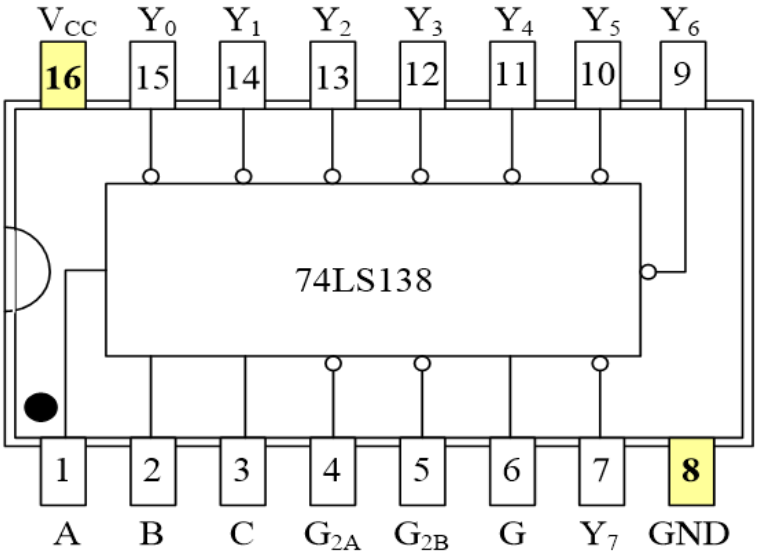
变量译码器是一个将 n 个输入变为 2n 个最小项输出的多输出端的组合逻辑电路。n 通常在 2~64 之间。

图表 1 变量译码器工作原理

2、变量译码器——74LS138

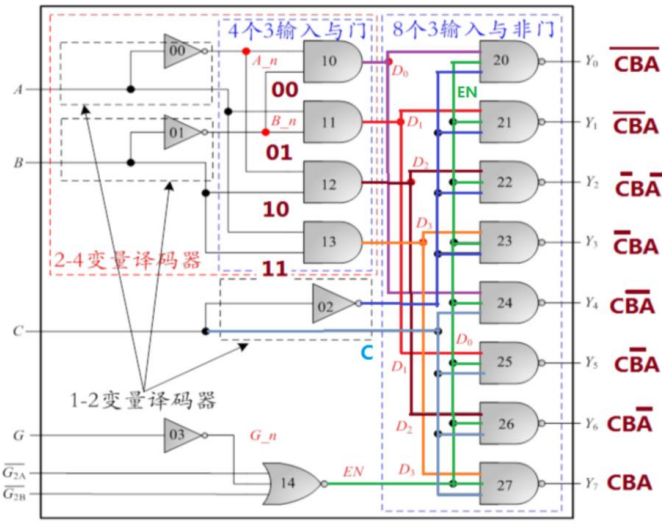
输入		译码器输出（低电平有效）							
使能	变量								
$GG_{2A}G_{2B}$	CBA	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
×11	×××	1	1	1	1	1	1	1	1
0××	×××	1	1	1	1	1	1	1	1
100	000	0	1	1	1	1	1	1	1
100	001	1	0	1	1	1	1	1	1
100	010	1	1	0	1	1	1	1	1
100	011	1	1	1	0	1	1	1	1
100	100	1	1	1	1	0	1	1	1
100	101	1	1	1	1	1	0	1	1
100	110	1	1	1	1	1	1	0	1
100	111	1	1	1	1	1	1	1	0

图表 2 74LS138 变量译码器功能表



图表 3 74LS138 变量译码器引脚

带 3 个使能端的 3-8 译码器的逻辑结构由三级门电路构成，输出低电平有效



图表 4 74LS138 连线明细图

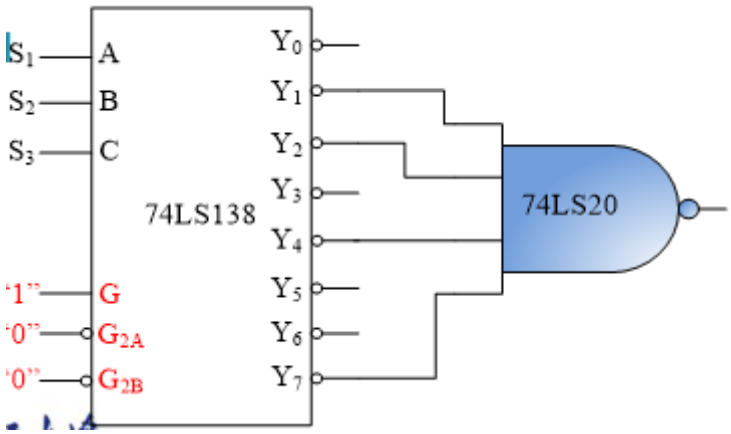
3、用变量译码器实现组合函数（本次实验中为楼道灯的控制）

变量译码器的输出对应所有输入变量的最小项组合，如果将函数转换成最小项和的形式，则可以用变量译码器实现函数的组合电路:

楼道灯输入输出关系　F =　001　+ 010　+ 100　+111

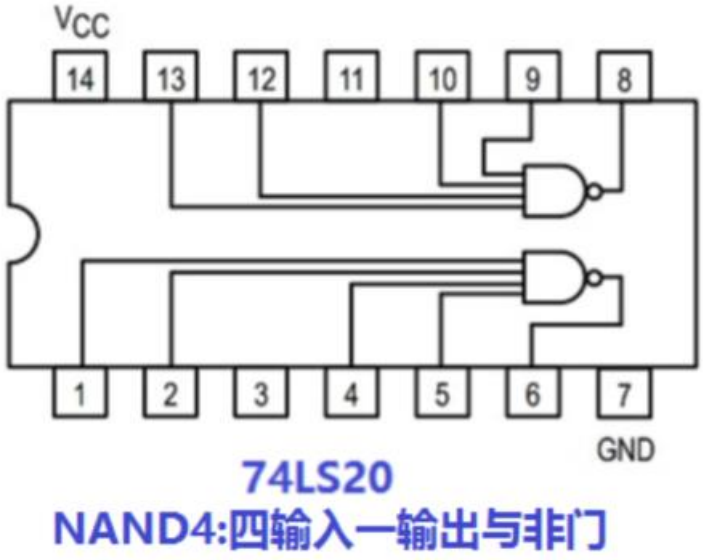
S_3	S_2	S_1	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

图表 5　楼道灯输入输出真值表



图表 6 楼道灯实现组合电路

上图中的 74LS20 的功效等同于 4 输入 1 输出的与非门



图表 7 74LS20 内部原理图

注：图表 1-7 来源洪老师课件

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

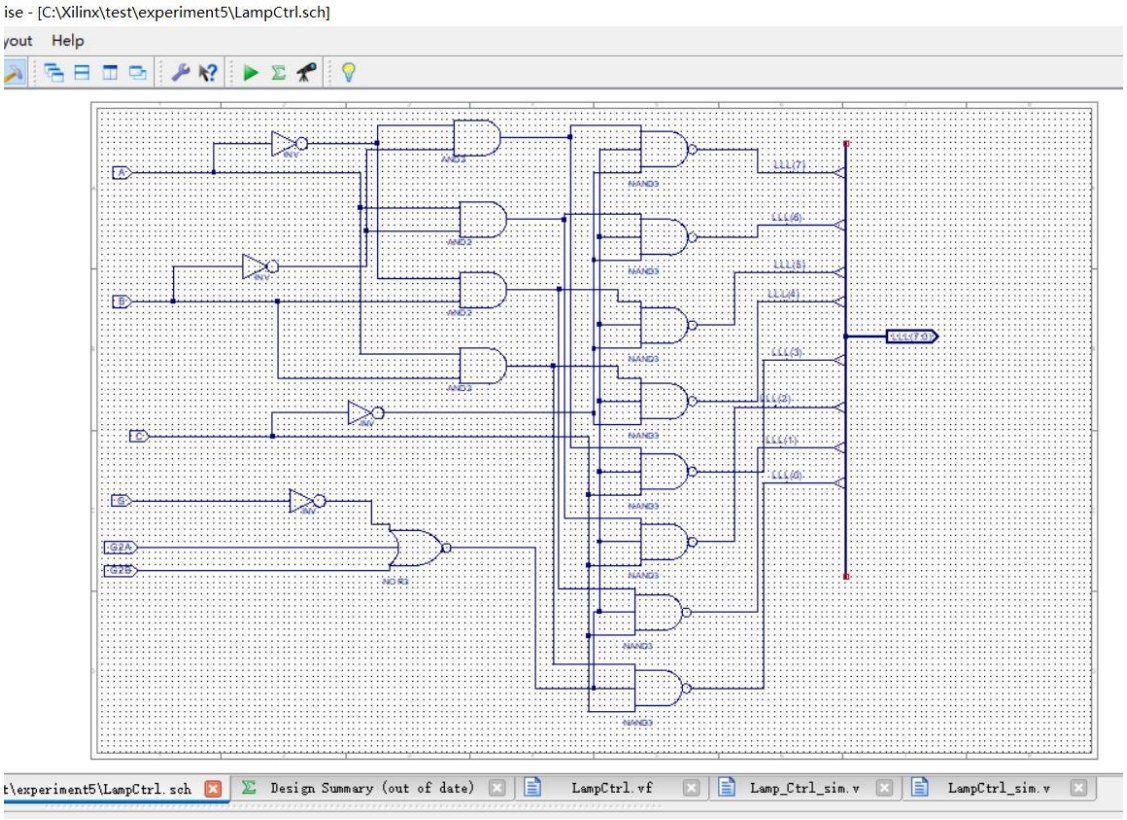
4.1 设计实现 74LS138 译码器模块

1、新建工程，工程名称用 D_74LS138_SCH。

2、新建 Schematic 源文件，文件名称用 D_74LS138。

3、用原理图方式进行设计

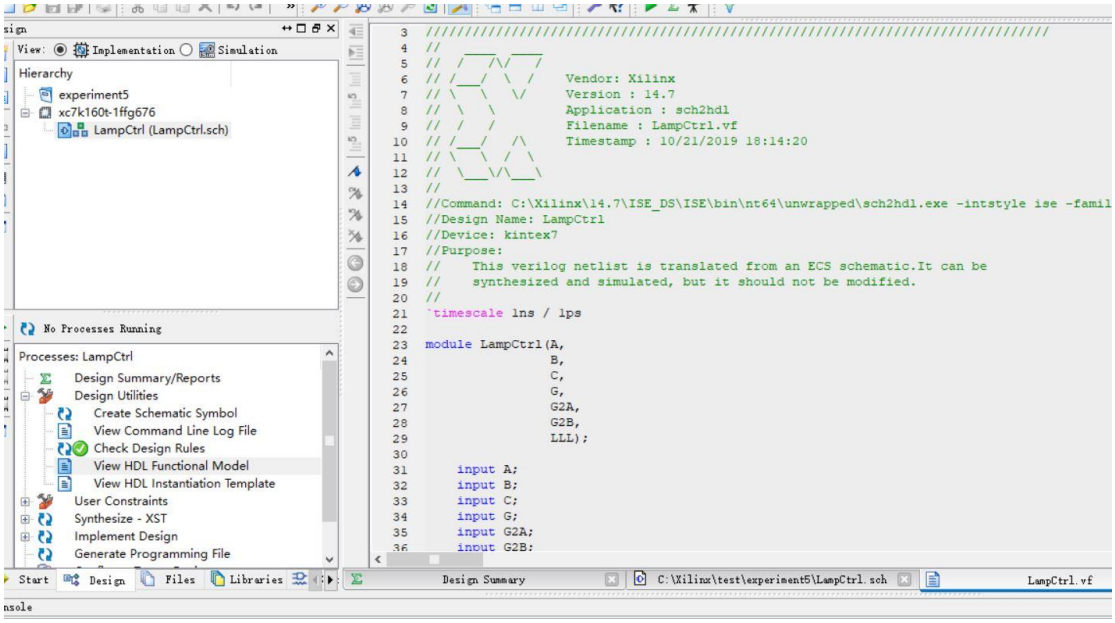
以下图片为实验中绘制的 74LS138 原理图



图表 8 74LS138 原理图

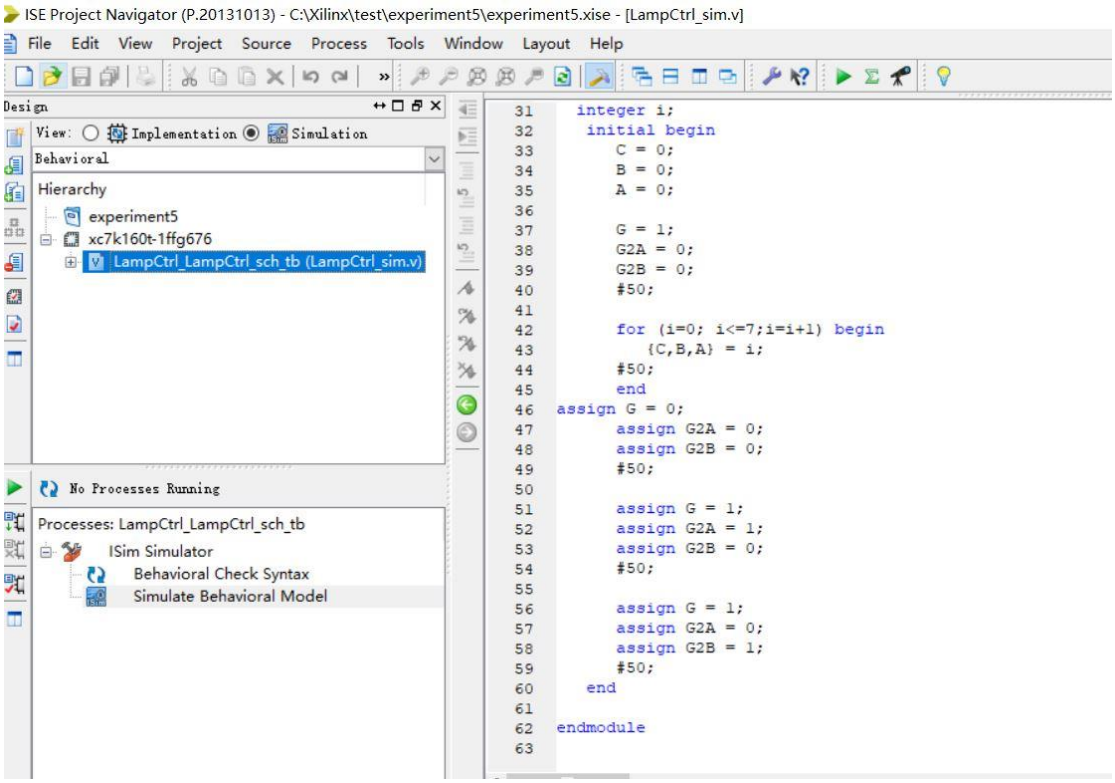
4、Check Design Rules，检查错误

5、View HDL Functional Model，查看并学习 Verilog HDL 代码

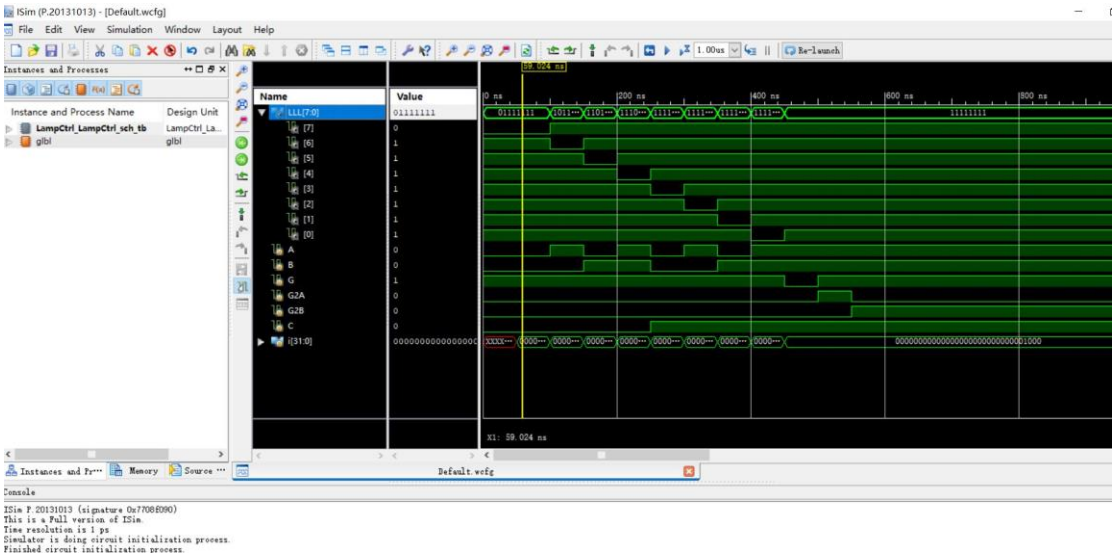


图表 9 ISE 中检查错误与查看代码

6、输入激励代码，对模块进行仿真



图表 10 仿真操作及对应的激励代码



图表 11 仿真激励波形图

从上述波形图可以看出，C，B，A 的值从 000 变化到 111，对应的 Yi（0<i<7）依次变为有效的 0 值，故符合预期，实现了 74LS138 的功能。

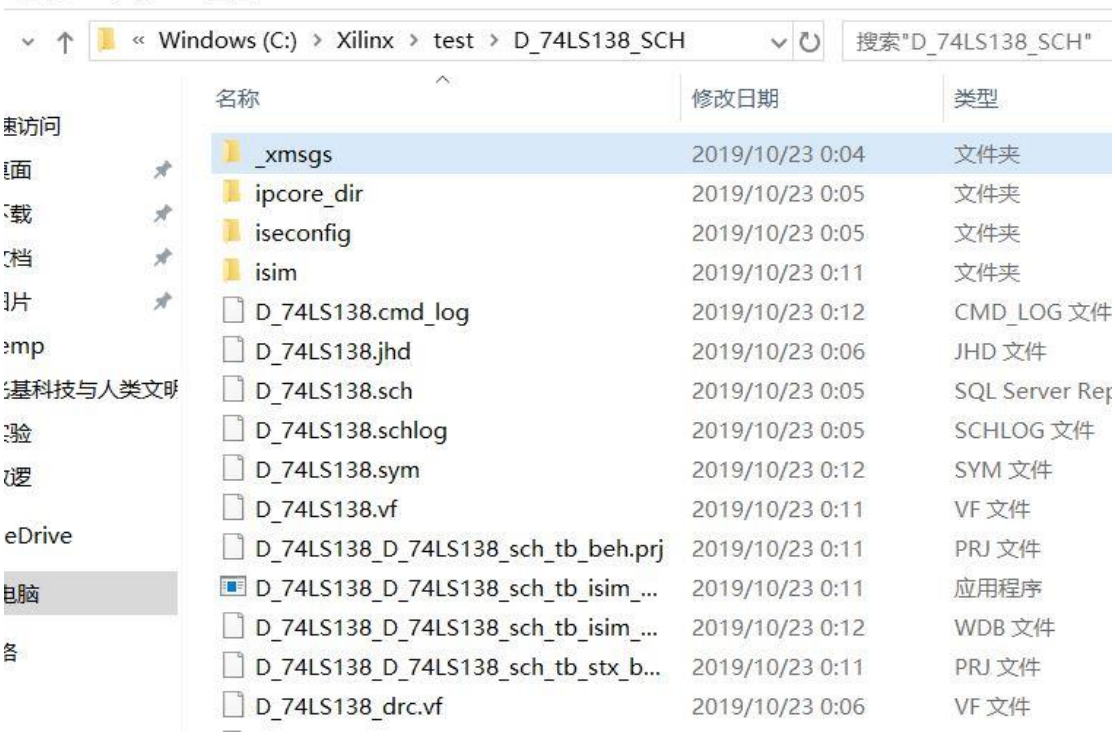
7、生成逻辑符号图和 VF 文件

点击 Create Schematic Symbol，状态栏出现如下文字

```
Command Line: sch2sym -intstyle ise -family kintex7 -refsym LampCtrl C:/Xilinx/test/experiment5/LampCtrl.sch C:/Xilir

Process "Create Schematic Symbol" completed successfully
```

图表 12 Create Schematic Symbol 结果

在工程的根目录找到.sym 与.vf 文件



图表 13 生成的文件

4.2 验证 74LS138

1、新建工程“D_74LS138_Test”。（第二个工程）

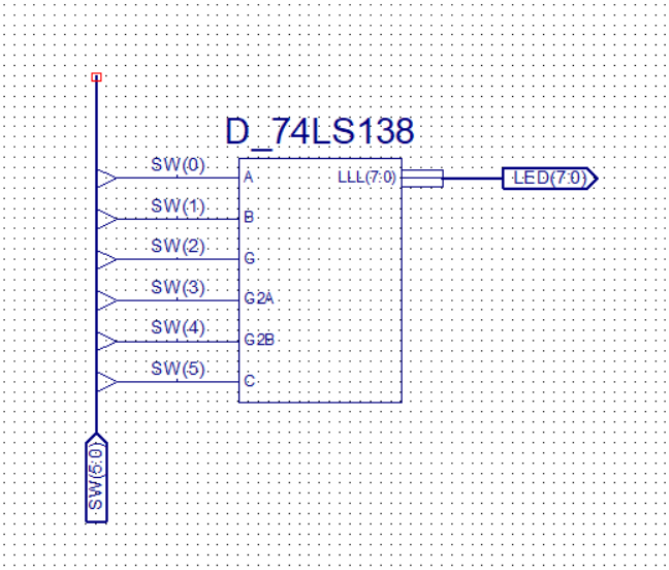
2、新建 Schematic 文件“D_74LS138_Test”。

3、复制 D_74LS138.sym 和.vf 到工程目录。（Add new copy of source）

在 symbols 框里的第一个元件，就是 D_74LS138

4、按照原理图设计楼道灯控制逻辑电路，连线图如下图所示

（注：老师的 PPT 中 SW 是 output，好像不太对）



图表 14 D_74LS138_Test 原理图

5、引脚约束

```
1
2  NET"SW[0]"LOC=AA13 | IOSTANDARD=LVCMOS15;#电压说明
3  NET"SW[1]"LOC=AB10 | IOSTANDARD=LVCMOS15;
4  NET"SW[2]"LOC=AA10 | IOSTANDARD=LVCMOS15;
5  NET"SW[3]"LOC=Y12  | IOSTANDARD=LVCMOS15;
6  NET"SW[4]"LOC=Y13  | IOSTANDARD=LVCMOS15;
7  NET"SW[5]"LOC=AA12 | IOSTANDARD=LVCMOS15;
8
9  NET"LED[0]"LOC=W23  | IOSTANDARD=LVCMOS33 ;#D1
10 NET"LED[1]"LOC=AB26 | IOSTANDARD=LVCMOS33 ;#D2
11 NET"LED[2]"LOC=Y25  | IOSTANDARD=LVCMOS33 ;#D3
12 NET"LED[3]"LOC=AA23 | IOSTANDARD=LVCMOS33 ;#D4
13 NET"LED[4]"LOC=Y23  | IOSTANDARD=LVCMOS33 ;#D5
14 NET"LED[5]"LOC=Y22  | IOSTANDARD=LVCMOS33 ;#D6
15 NET"LED[6]"LOC=AE21 | IOSTANDARD=LVCMOS33 ;#D7
16 NET"LED[7]"LOC=AF24  | IOSTANDARD=LVCMOS33 ;#D8
17
```

图表 15 ucf 文件代码

6、pinout report

Number	Name	Usage	Name		Standard
AA12	SW<5>	IOB	IO_L16N_T2_33	INPUT	LVCMO...
AA13	SW<0>	IOB	IO_L16P_T2_33	INPUT	LVCMO...
AE17	SW<1>	IOB	IO_L1P_T0_32	INPUT	LVCMO...
AE18	SW<2>	IOB	IO_L3P_T0_DQS_32	INPUT	LVCMO...
AF14	SW<3>	IOB	IO_L2P_T0_32	INPUT	LVCMO...
AF15	SW<4>	IOB	IO_L2N_T0_32	INPUT	LVCMO...
AD15	LED<3>	IOB	IO_L4P_T0_32	OUTPUT	LVCMO...
AE15	LED<4>	IOB	IO_L4N_T0_32	OUTPUT	LVCMO...
AF17	LED<1>	IOB	IO_L1N_T0_32	OUTPUT	LVCMO...
AF18	LED<5>	IOB	IO_L3N_T0_DQS_32	OUTPUT	LVCMO...
AF19	LED<7>	IOB	IO_L5P_T0_32	OUTPUT	LVCMO...
AF20	LED<6>	IOB	IO_L5N_T0_32	OUTPUT	LVCMO...
V13	LED<2>	IOB	IO_0_VRN_32	OUTPUT	LVCMO...
W23	LED<0>	IOB33	IO_L8P_T1_12	OUTPUT	LVCMO...

图表 16 pinout report

7、Generate Programming file 设计实现并检查约束结果

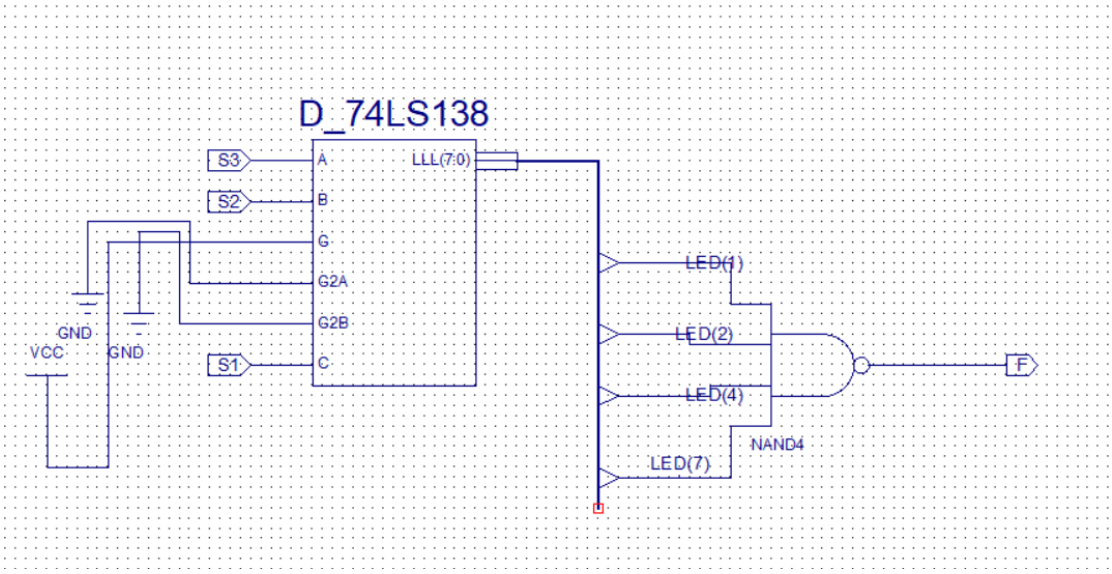
4.3 实现楼道灯控制

1 新建工程 LampCtrl138。

2 复制 D_74LS138.sym 和.vf 文件到工程目录。

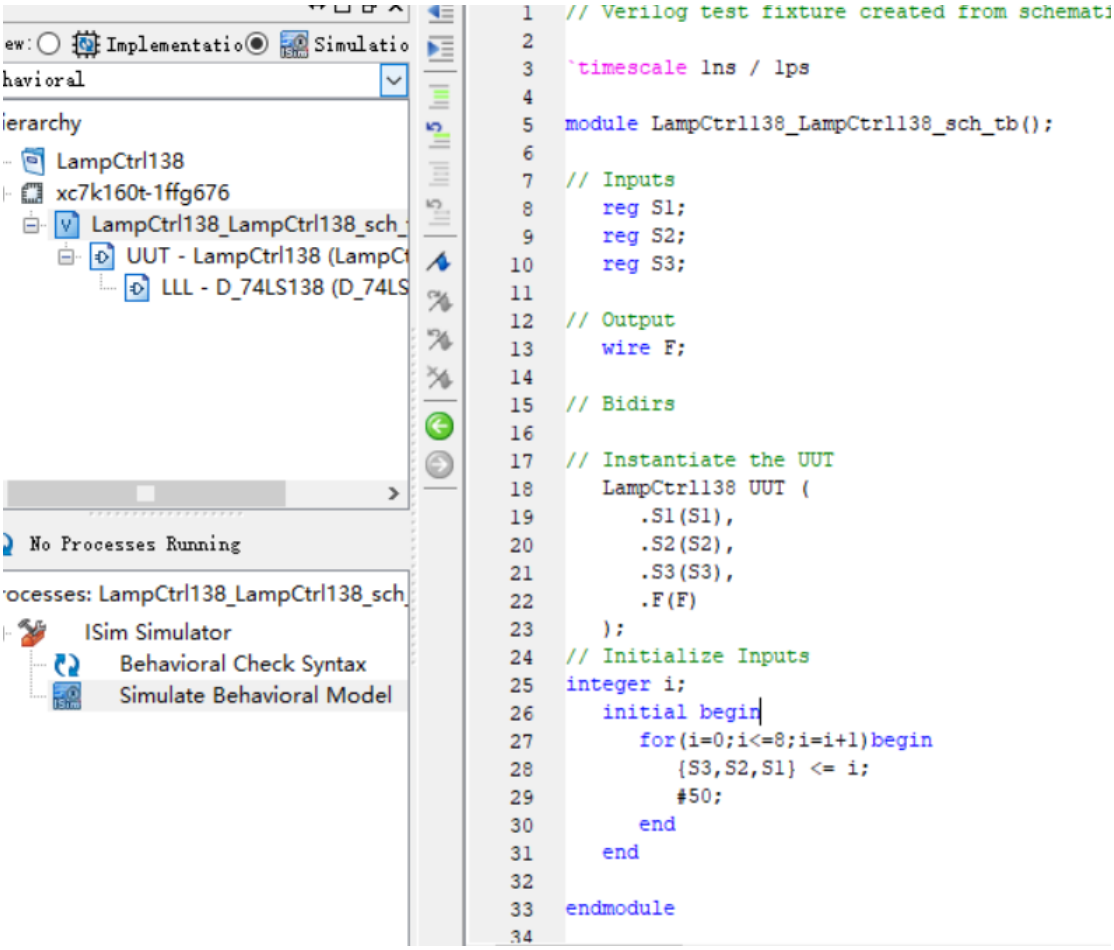
只有加入到了工程目录中，在设计实现楼道灯控制时，才可以直接使用之前完成的 74LS138 元件进行设计。

3 按照原理图设计楼道灯控制逻辑电路，连线图如下图所示：

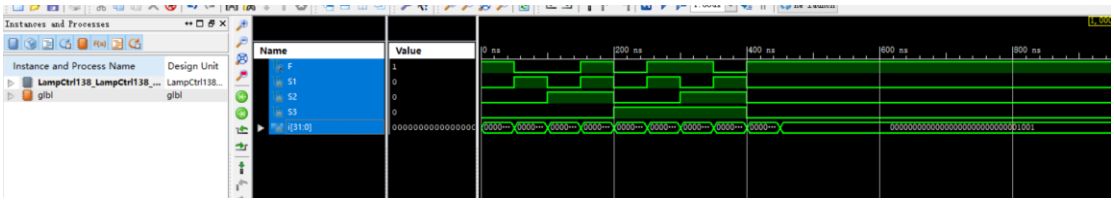


图表 17 楼道灯控制原理图

4、仿真



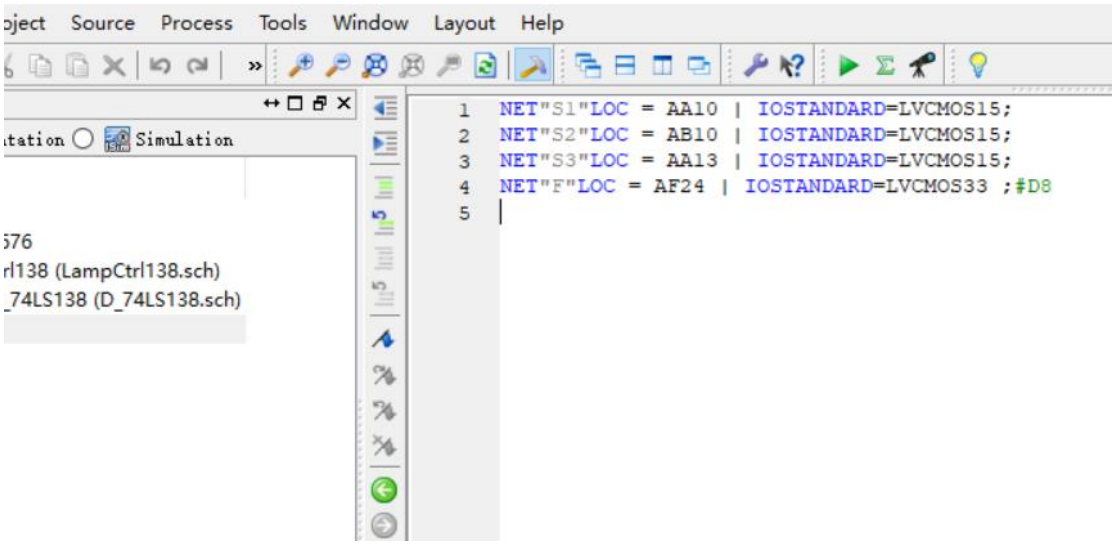
图表 18 仿真代码

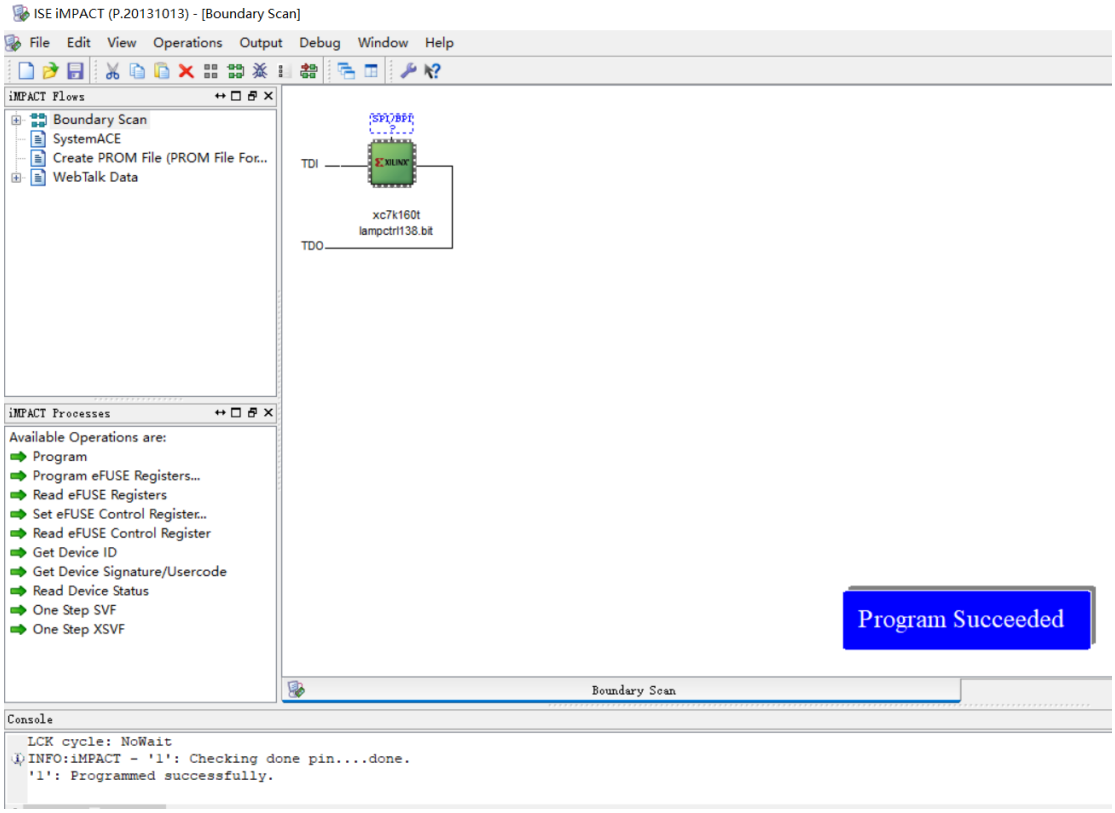


图表 19 仿真激励波形图

可以看到，在仿真激励时，只有一个 S1、S2、S3 中全部为 0 时，F 输出 1，LED 灯亮；只有一个为 1 时，F 输出 0，LED 灭；有两个为 1 时，F 输出 1，LED 灯亮；全部为 1 时，F 输出位 0，LED 灯灭；总体符合楼道灯的功能要求，但可以看出是反逻辑的，其实只需在最后的与非门后再加上一个非门，即为正逻辑，但同样可以将开关向上视为关闭开关，同样可以符合正逻辑，故不更改原先的电路图。

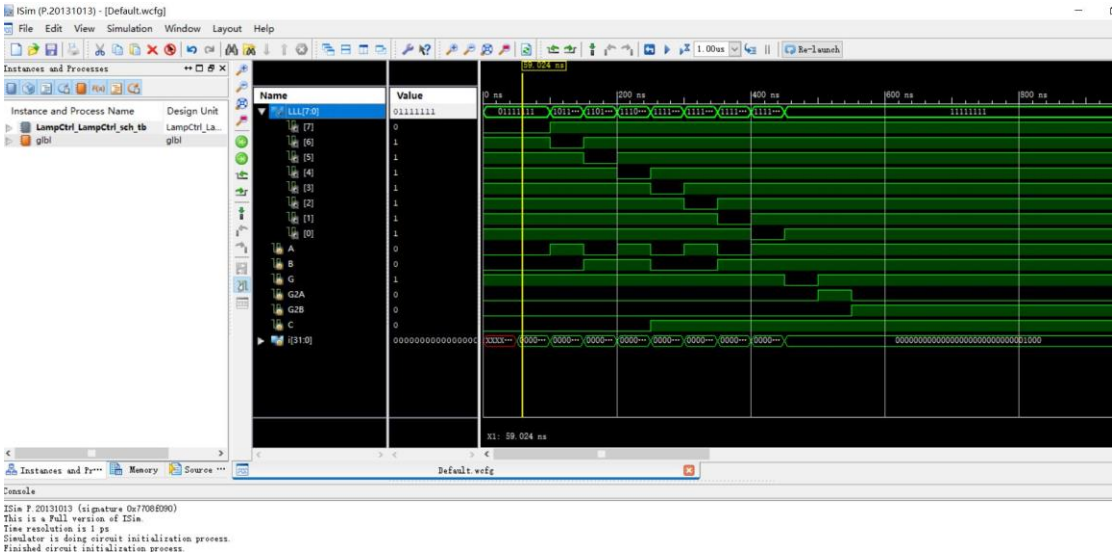
5、下载验证





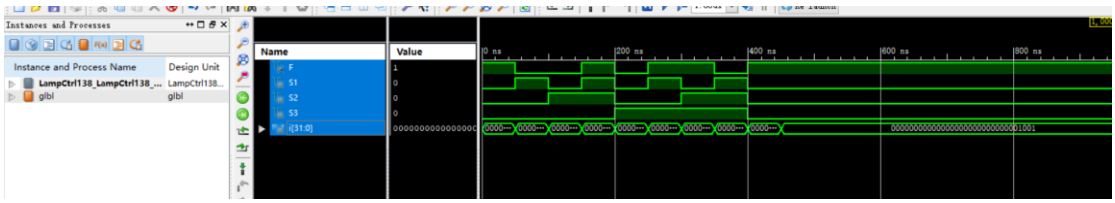
五、实验结果与分析

一、74LS138 仿真激励波形图

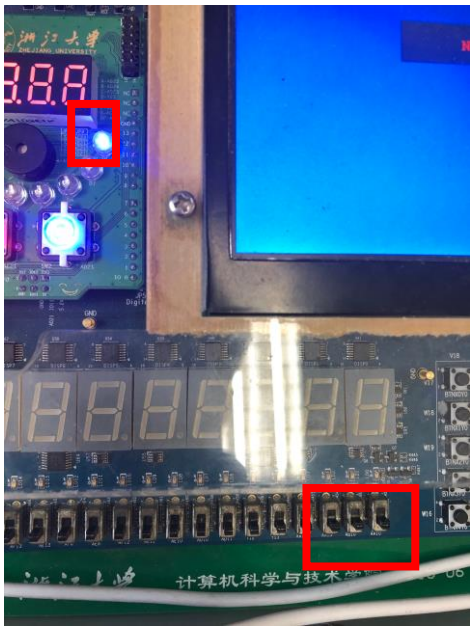


二、楼道灯控制

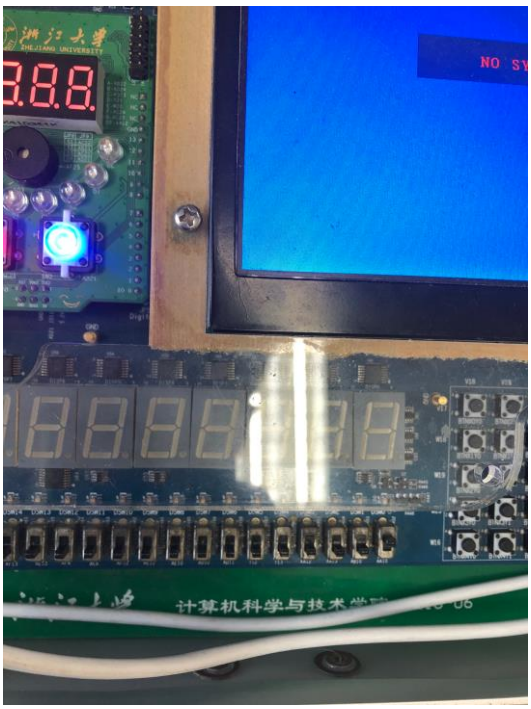
1、仿真激励波形图



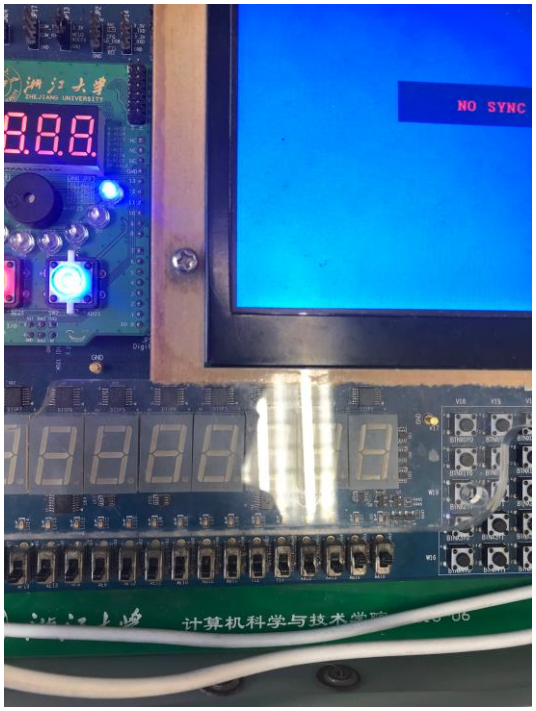
2、当三个开关都向下，灯亮



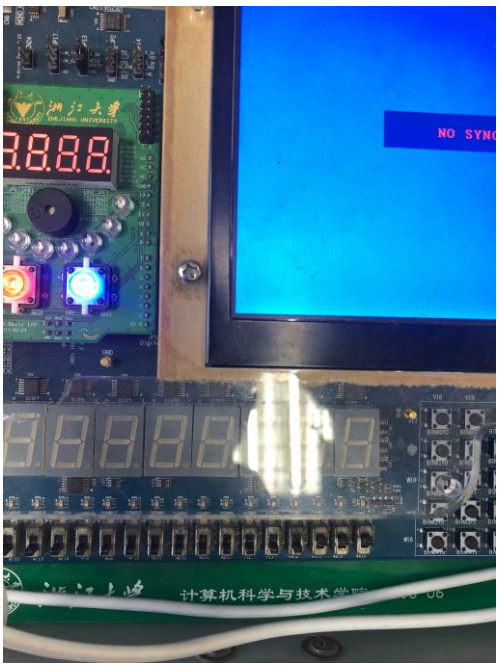
3、当一个开关向下，两个开关向上灯灭



4、当两个开关向下，一个开关向上，灯亮



5、当三个开关向下，灯灭。



分析：

对波形图的分析在实验步骤中给出

总体而言可以实现楼道灯的功能，可以每个开关单独控制灯的亮灭，也可以不同开关共同作用达到控制效果

但很明显的是，开关向上本应意味着打开，但实现的效果与之成反逻辑的状态，因此在实验中，开关向上被视作关闭。

想要将之转换为正逻辑的方法很简单，即在实现电路的与非门后再加上一个非门即可

上述方案同样可以实现楼道灯的控制。

六、讨论、心得

这是第二次使用 ise 这个软件，总体来说也比第一次熟练了不少，在模仿为主的同时，也慢慢开始理解了每一部为什么要这么做，比如 ucf 的引脚约束，可以明白哪些是输入哪些是输出，怎么样用 SWORD 板上不同的开关及灯来实现等等。同时也慢慢可以通过实验现象分析原理图的改进方式，如楼道灯控制的实验中发现了激励波形图中是反逻辑的，因此联想到了在最后一个与非门后再加入一个非门使得总体变为正逻辑（重复度较高，并未截图上传）。

总体来说这次实验让我对 ise 的使用更加熟练了，也在绘制原理图上更加快速，希望下一步可以熟悉相关代码的编写，因为在我看来画的再快不如代码敲的快啊哈哈哈。

实验 6—7 段数码管显示译码器设计与应用

姓名：李想　　学号：3180103542　　专业：软件工程

课程名称：逻辑与计算机设计基础实验　　同组学生姓名：　无

实验时间：2019-10-23　　实验地点：紫金港东 4-509　　指导老师：　洪奇军

一、实验目的和要求

1. 掌握七数码管显示原理

2. 掌握七段码显示译码设计

3. 进一步熟悉 Xilinx ISE 环境及 SWORD 实验平台

二、实验内容和原理

2.1 实验内容

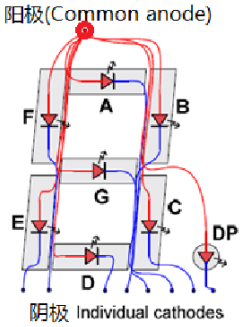
1 原理图设计实现显示译码 MyMC14495 模块

2 用 MyMC14495 模块实现数码管显示

2.2 实验原理

1、7 段数码管原理

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。



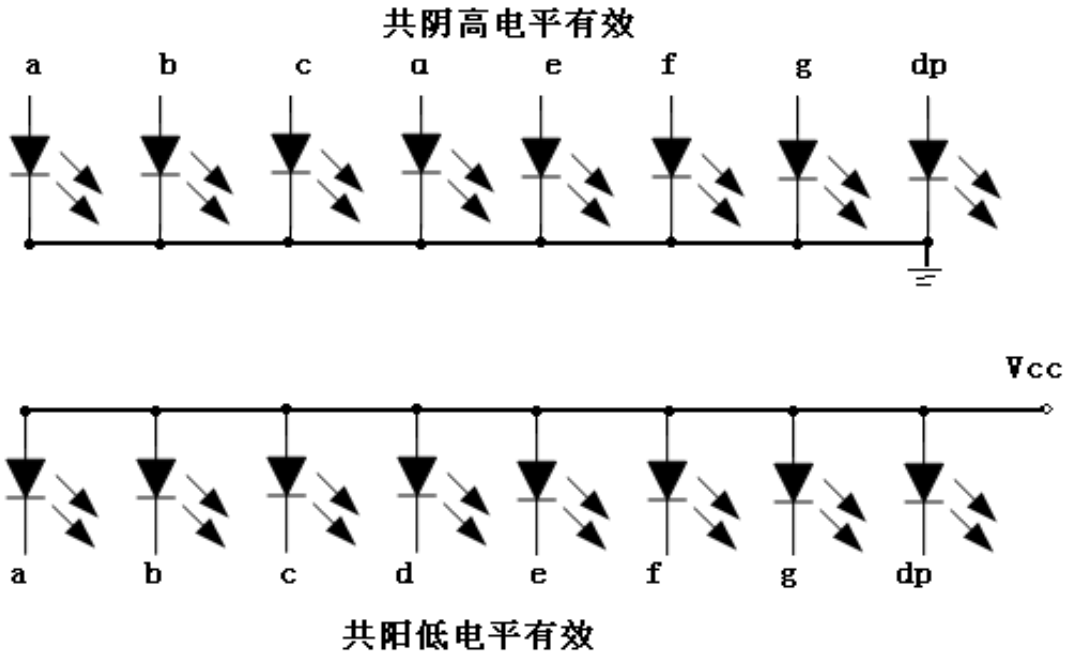
图表 1　7 段数码管原理

2、数码管控制原理

LED 的正极(负极)连在一起，另一端作为点亮的控制，分为共阴和共阳两种类型：

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



图表 2 共阴共阳示意图

3、显示原理

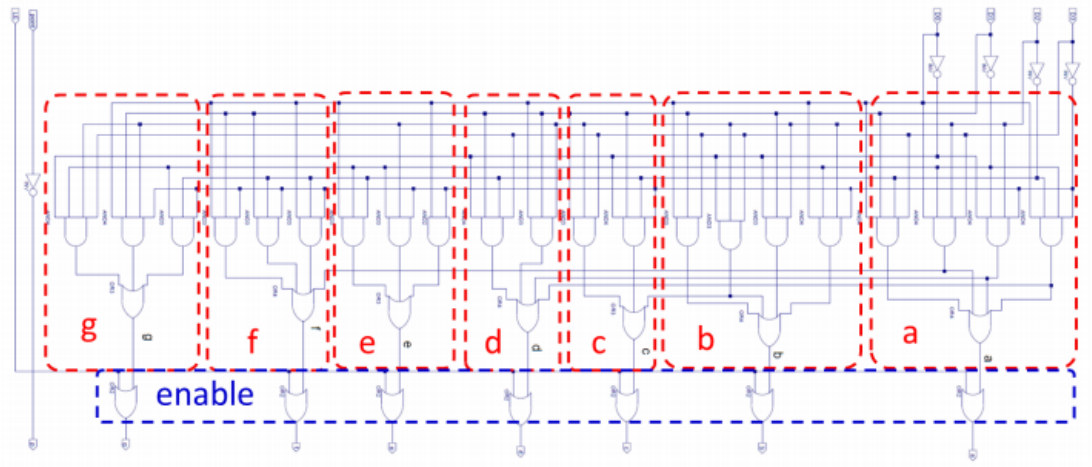
通过 4-16 译码器与七段数码管的一一对应来实现数字的显示。其对应关系如下:

Hex	$D_3D_2D_1D_0$	BI/LE	a	b	c	d	e	f	g	p
0	0 0 0 0	1	0	0	0	0	0	0	1	p
1	0 0 0 1	1	1	0	0	1	1	1	1	p
2	0 0 1 0	1	0	0	1	0	0	1	0	p
3	0 0 1 1	1	0	0	0	0	1	1	0	p
4	0 1 0 0	1	1	0	0	1	1	0	0	p
5	0 1 0 1	1	0	1	0	0	1	0	0	p
6	0 1 1 0	1	0	1	0	0	0	0	0	p
7	0 1 1 1	1	0	0	0	1	1	1	1	p
8	1 0 0 0	1	0	0	0	0	0	0	0	P
9	1 0 0 1	1	0	0	0	0	1	0	0	P
A	1 0 1 0	1	0	0	0	1	0	0	0	P
B	1 0 1 1	1	1	1	0	0	0	0	0	P
C	1 1 0 0	1	0	1	1	0	0	0	1	P
D	1 1 0 1	1	1	0	0	0	0	1	0	P
E	1 1 1 0	1	0	1	1	0	0	0	0	P
F	1 1 1 1	1	0	1	1	1	0	0	0	P
X	x x x x	0	1	1	1	1	1	1	1	1

图表 3 7 段数码管真值表示意图

4、电路连接

通过对真值表中每一个变量进行卡诺图化简，可得到它们各自的逻辑表达式，将其连接成图如下:



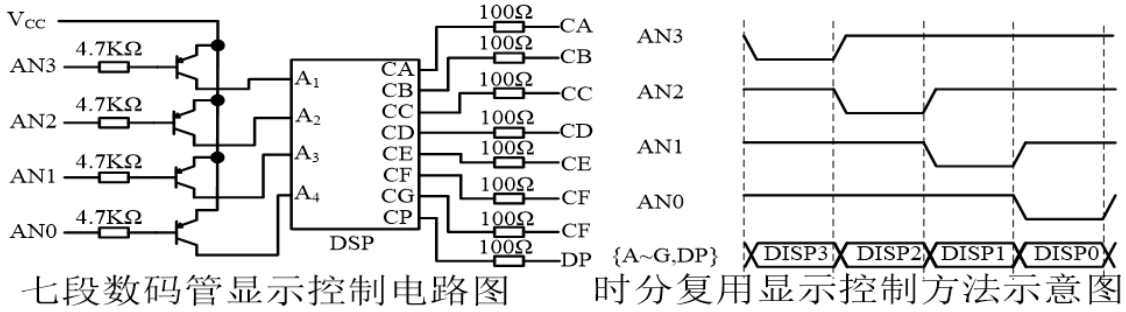
图表 4 数码管工作原理连线示意图

5、多位七段数码管显示原理

静态显示为每个 7 段码对应一个显示译码电路，而动态扫描显示（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，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 4 位七段码结构。

6、时分控制示意

通过动态扫描，低电平与输入显示对应，分时送 a~g，p，用序列信号控制。



图表 5 时分复用原理

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

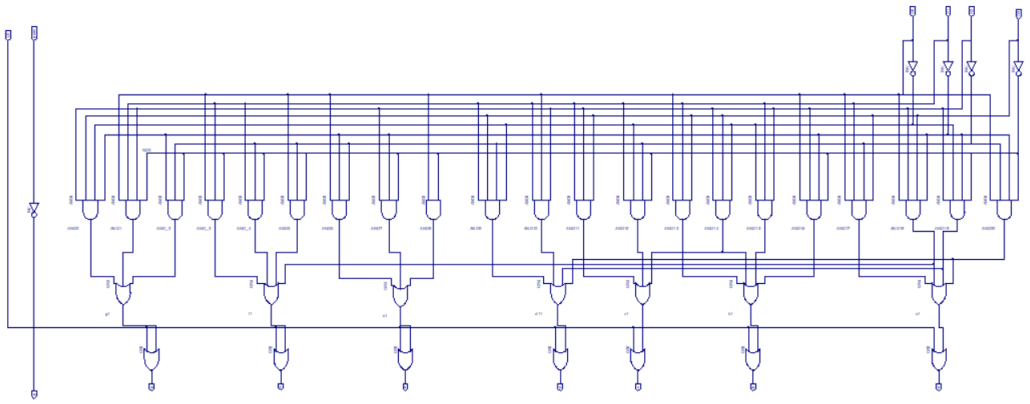
四、操作方法与实验步骤

4.1 设计实现 MyMC14495

1、新建工程，工程名称用 MyMC14495。

2、新建源文件，文件名称用 MyMC14495。

3、用原理图方式进行设计



图表 6 MC14496 原理图

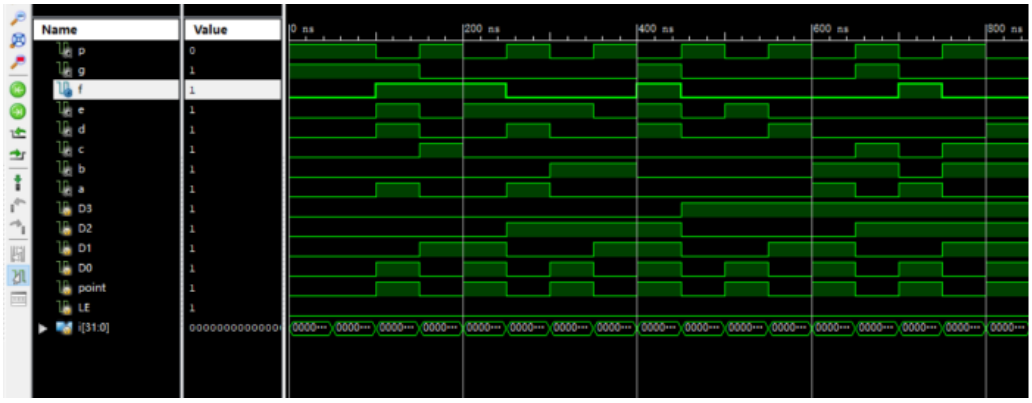
4、Check Design Rules，检查错误

5、View HDL Functional Model，查看并学习 Verilog HDL 代码

6、Simulation 仿真

对 MyMC14495 模块进行仿真，在 initial 部分修改代码如下：

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

end
#50
LE=1;
end
//    `endif endmodule
```



图表 7 MyMC14495 仿真结果

分析：从结果可以看出 D0-D3 均为低电平时，a-g 中仅有 g 为 1，设计中采用共阳数码管设计，因此仅有 g 为暗，其余均为亮，故显示为 0，以此类推 D0-D3 分别表示 0-F 时，7 端数码管依次显示对应数字。p 为输入 point 的反，用于控制小数点。

7、生成.sym 和.vf 文件

MyMC14495_sch.sym	2019/10/23 16:37	SYM 文件	4 KB
MyMC14495_sch.vf	2019/10/23 16:37	VF 文件	7 KB

图表 8 生成的.sym 和.vf 文件

4.2 实现数码管显示

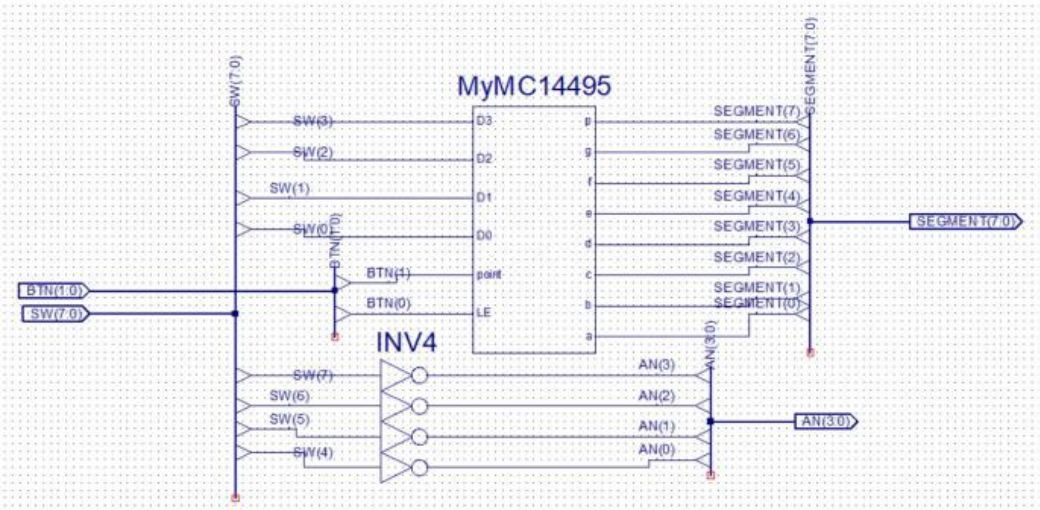
1、新建工程 DispNumber_sch

2、新建 schematic 文件 DispNumber_sch

3、添加 MyMC14495 元件

复制 MyMC14495.sym 和.vf 到工程根目录，并将.vf 添加到工程中。

4、利用原理图方式实现数码管显示



图表 8 数码管显示原理图

5、下载验证

UCF 引脚定义

新建文本文档 (1).txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
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;
NET "SW[5]" LOC = Y12  | IOSTANDARD = LVCMOS15;
NET "SW[6]" LOC = AD11  | IOSTANDARD = LVCMOS15;
NET "SW[7]" LOC = AD10  | IOSTANDARD = LVCMOS15;

NET "BTN[0]" LOC = AF13  | IOSTANDARD = LVCMOS15 ;#SW[14]
NET "BTN[1]" 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 ;#poir

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 ;
```

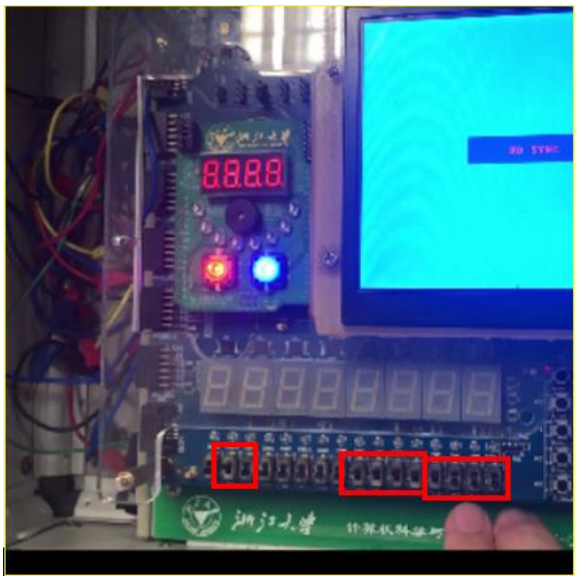
图表 9 UCF 引脚约束

五、实验结果与分析

1、MyMC14495 波形图

具体图形以及分析在实验步骤中给出

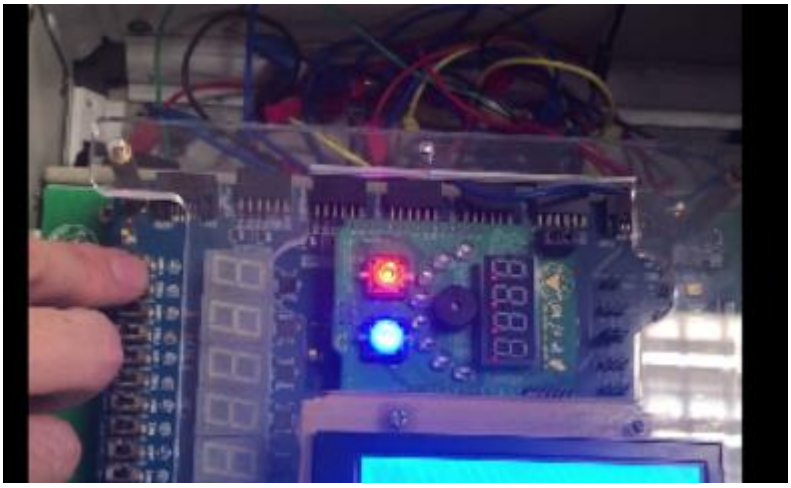
2、数码管显示



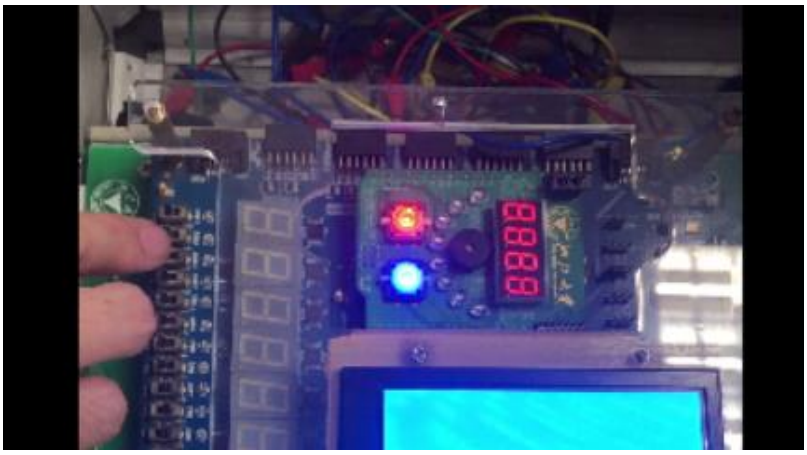
图表 10 实验结果数码管控制

如上图所示，最左边方框中的开关从左至右控制 point 及 LE，中间的方框中的四个开关负责数码管控制，最右边的四个开关控制数字大小，按照二进制规律。

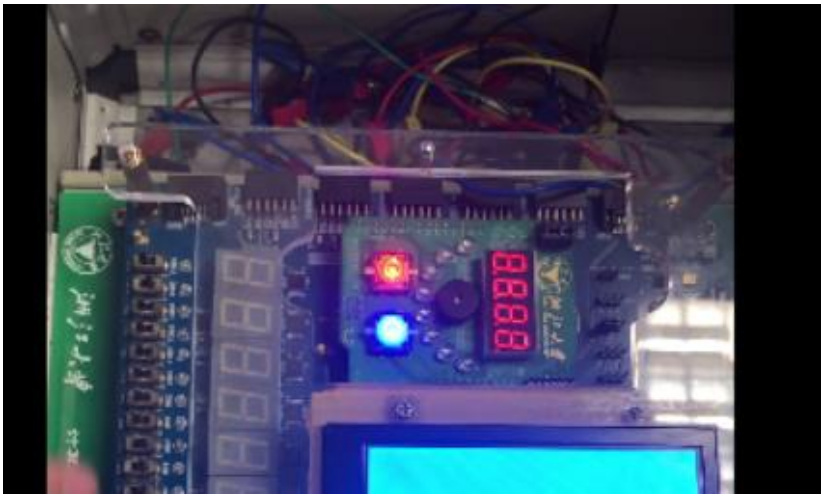
以下图片为部分实验结果图：



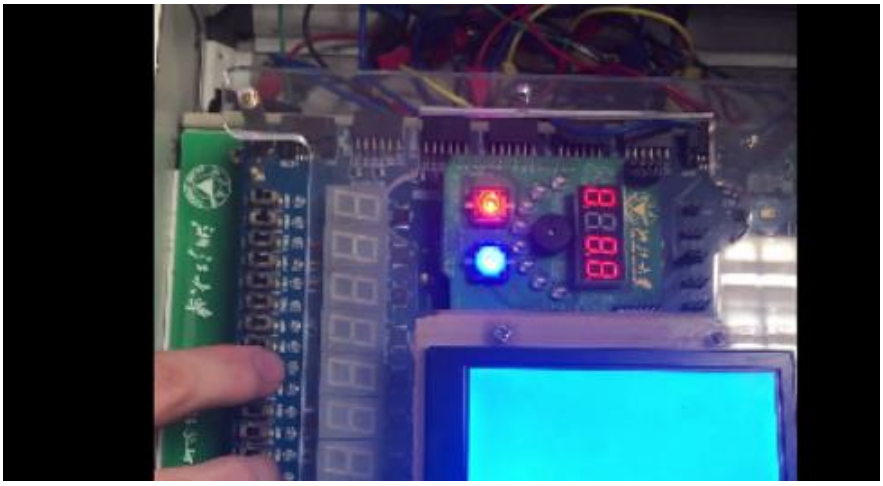
图表 11 打开 LE，打开 point 仅小数点部分亮



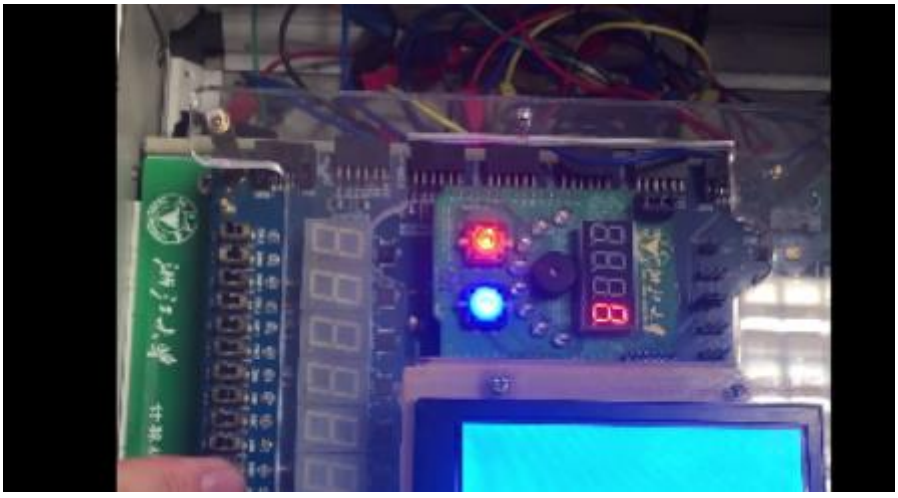
图表 12 关闭 LE，关闭 point 仅数字部分点亮



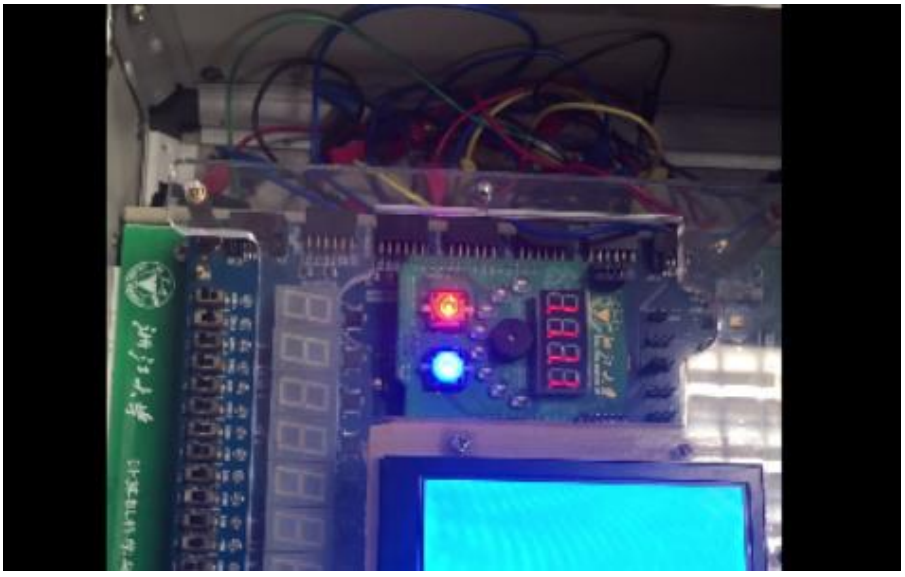
图表 13 关闭 LE，打开 point 数字部分及小数点部分均亮



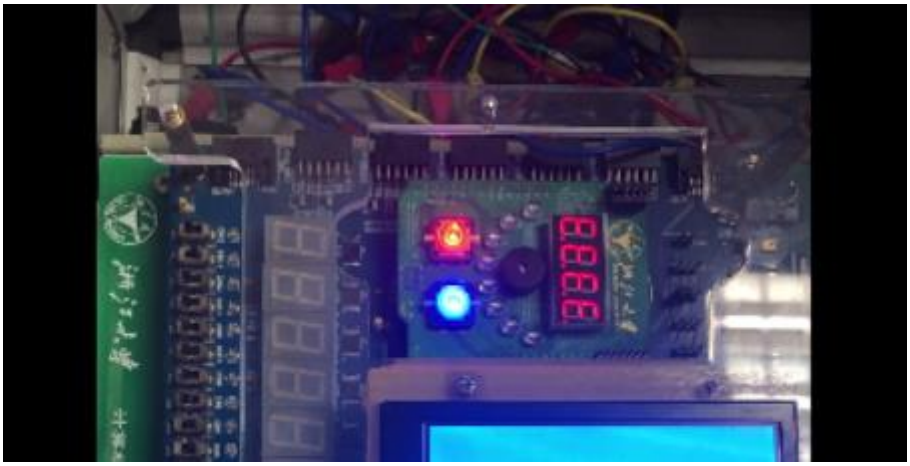
图表 14 各个数码管的单独控制（1）



图表 15 各个数码管的单独控制（2）



图表 16 各个数码管显示 1



图表 17 各个数码管显示 E

经实验验证，四个数码管可以表示 0-9、a-e 的各个数字，且可以分别亮暗。

六、讨论、心得

这次实验思路本身较为简单，但是绘制 MC14495 的原理图时，由于电路较为复杂，因此花去了很多时间，感到了原理图的不便性，初次绘制时常常会出错。但是，绘制到一半时，停下来，理解了 LE，以及“先与后或”的这样一种模式之后，就发现了许多重复的模块，唯一需要注意的就是各个最小项的连线，这样一来绘制也快了许多。

实验 7—多路选择器设计和应用

姓名：李想　　学号：3180103542　　专业：软件工程

课程名称：逻辑与计算机设计基础实验　　同组学生姓名：　无

实验时间：2019-10-23　　实验地点：紫金港东 4-509　　指导老师：　洪奇军

一、实验目的和要求

1、掌握数据选择器的工作原理和逻辑功能

2、掌握数据选择器的使用方法

3、掌握 4 位数码管扫描显示方法

4、掌握 4 位数码管显示应用—记分板设计

二、实验内容和原理

2.1 实验内容

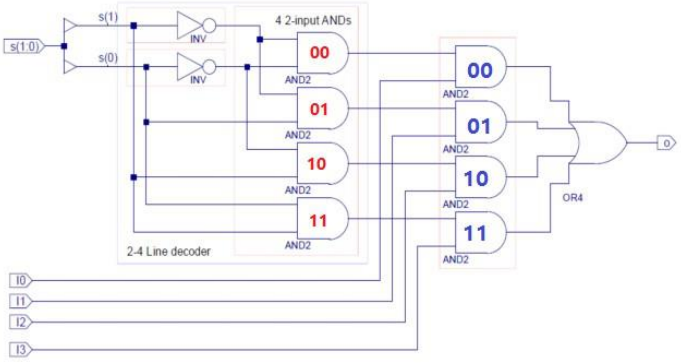
1、数据选择器设计

2、记分板设计

2.2 实验原理

1、四选一多路选择器

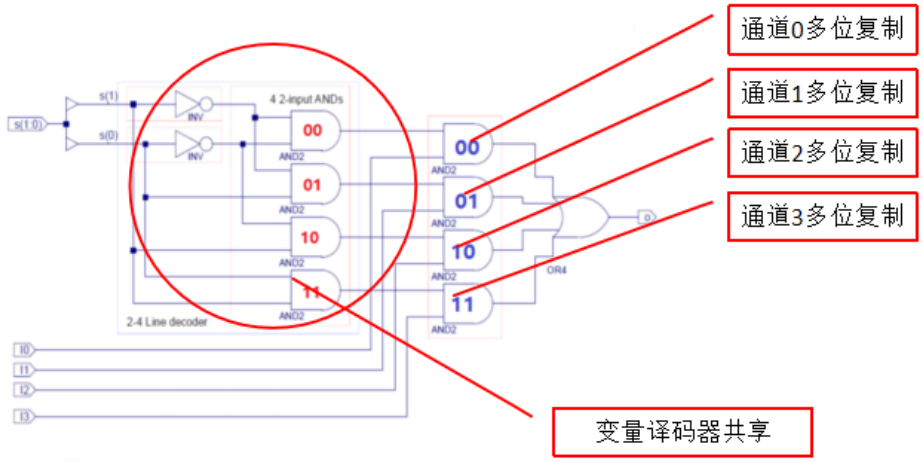
根据事件简化真值表，通过输出判断控制信号，将表达式化简成最小项与或结构。



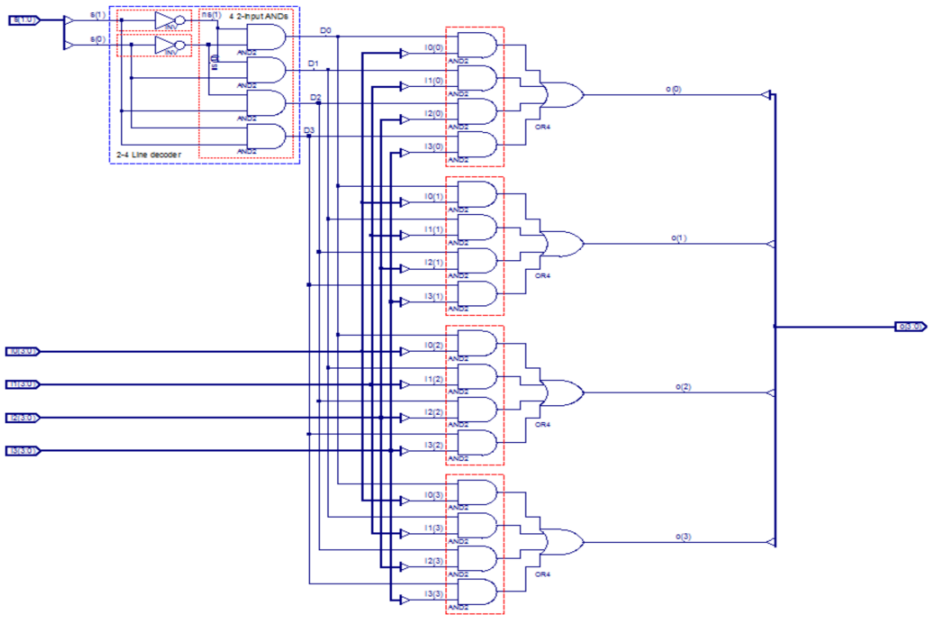
图表 1 多路选择器原理图

2、四位四选一多路选择器

控制结构不变，每路输出向量化



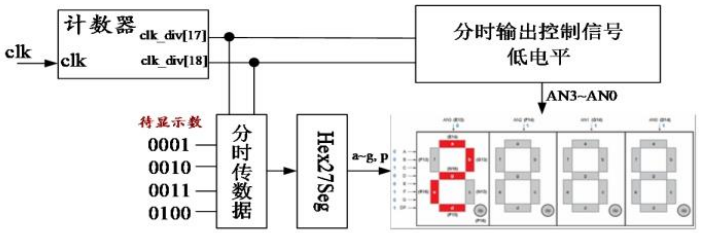
图表 2 扩展至 4 位的方式



图表 3 4 位四选一扩展：MUX4to1b4

3、动态扫描显示

扫描信号来自计数器，**将时序电路转化为组合电路**，由板载时钟 clk(100MHz)作为计数器时钟，分频后输入到数据选择器的控制端，作为数码管的扫描信号。计数器的分频系数要适当，眼睛舒适即可。

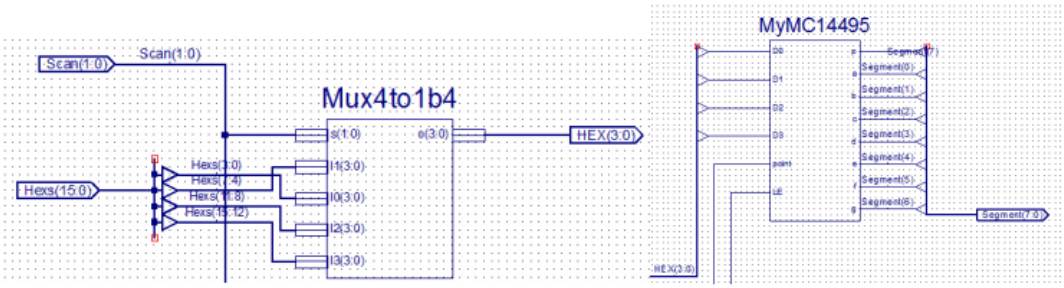


图表 4 动态扫描显示原理

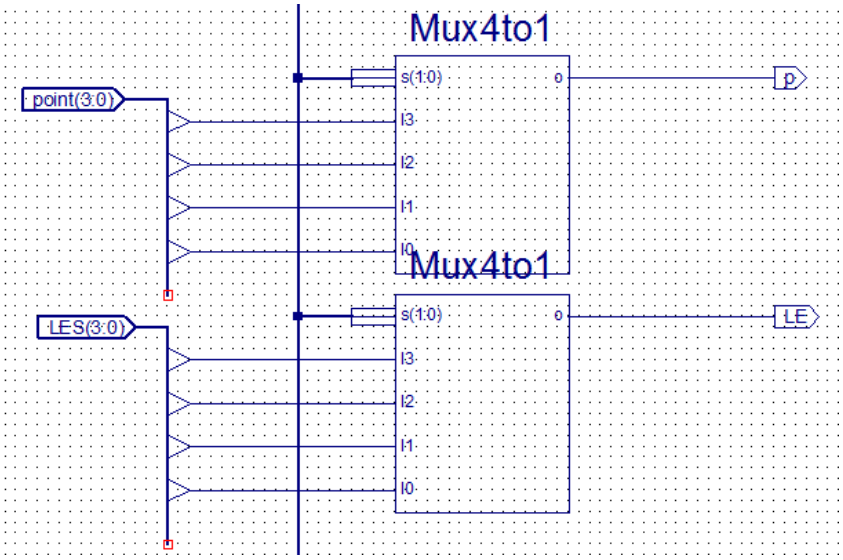
4、多路选择器应用：4 位七段显示扫描控制

实现原理：

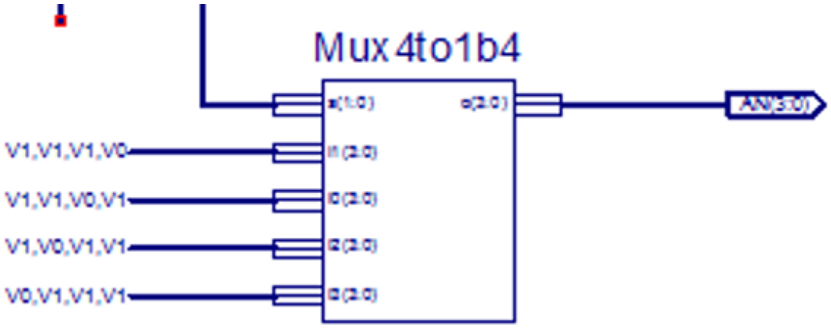
Scan 输入扫描控制信号，Hexs 输入需要显示的 4 个 4 位二进制数，多路选择器输出需要显示的 1 个 4 位二进制数，MyMC14495 输出当前显示的数码管的七段码



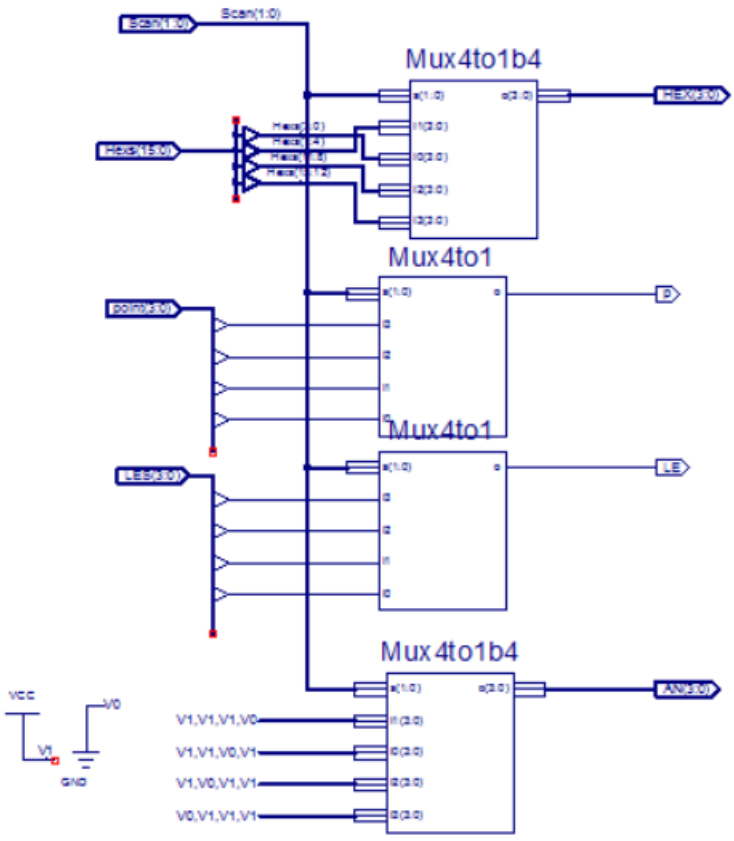
图表 5 数字显示



图表 6 小数点控制及数码管消隐控制



图表 7 位选择电路（低电平有效）



图表 8 displaysync 最终电路原理图

5、时钟计数分频器

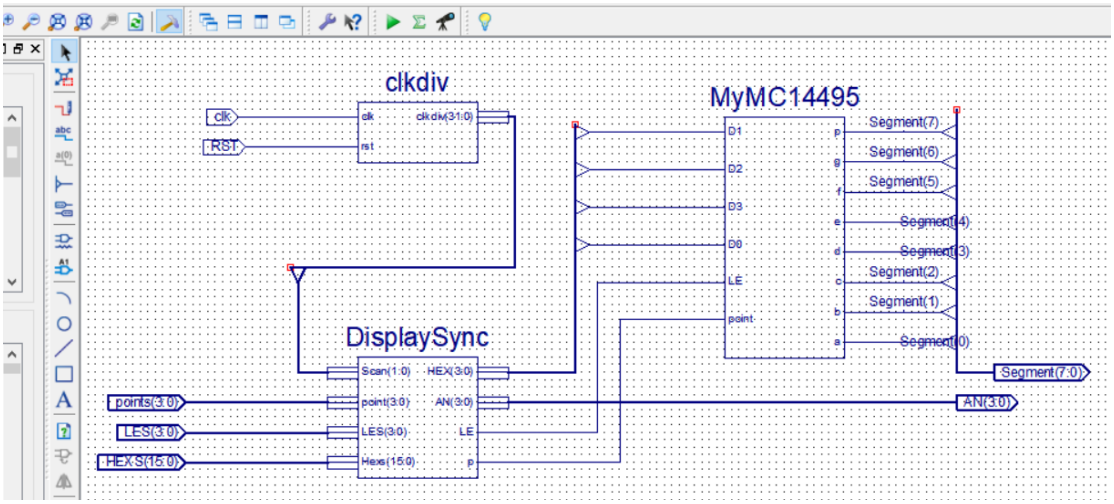
一个 32 位时钟计数分频器可输出 2-232 分频信号，可用于一般非同步类时钟信号，其延时较高，要求不高的时钟也可以用，在本实验中多位七段显示器动态扫描即可用。

```
module  clkdiv(input clk,
               input rst,
               output reg[31:0]clkdiv
               );

// Clock divider-时钟分频器

   always @ (posedge clk or posedge rst) begin
      if (rst) clkdiv <= 0;
      else clkdiv <= clkdiv + 1'b1;
   end

endmodule
```

图表 9 时钟模块 Verilog 代码

6、设计 disp_num 显示模块

通过结合以上几个设计好的模块，我们可以实现 4 位七段数码管的扫描显示。



图表 10 disp_num 显示模块原理图

7、设计按键数据输入模块

使用行为描述设计，其功能为四个按键，各按一下，4 个 4 位 2 进制数分别加 1。

```
34
35  module CreateNumber(
36      input wire[3:0] btn,
37      output reg[15:0] num
38      );
39      wire[3:0] A,B,C,D;
40
41      initial num<=16'b1010_1011_1100_1101; //display "AbC
42
43      assign A=num[3:0]+4'd1;
44      assign B=num[7:4]+4'd1;
45      assign C=num[11:8]+4'd1;
46      assign D=num[15:12]+4'd1;
47
48      always@(posedge btn[0]) num[3:0]<=A;
49      always@(posedge btn[1]) num[7:4]<=B;
50      always@(posedge btn[2]) num[11:8]<=C;
51      always@(posedge btn[3]) num[15:12]<=D;
52
53  endmodule
54
```

图表 11 CreateNumber 模块 Verilog 代码

三、主要仪器设备

1、装有 ISE 14.7 的计算机　　1 台

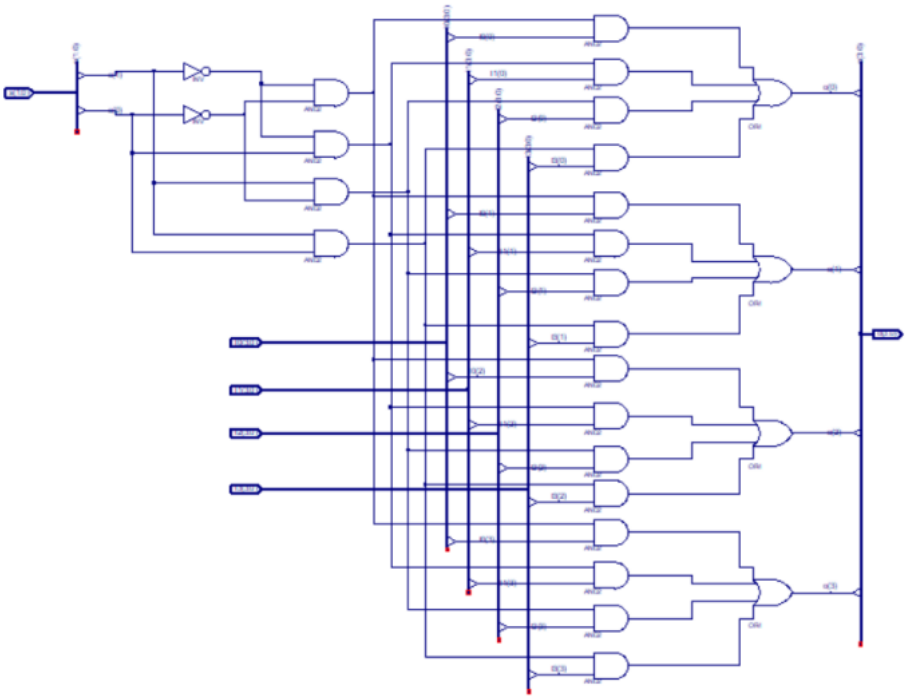
2、SWORD 开发板　　1 套

四、操作方法与实验步骤

1、数据选择器设计

1 新建工程：工程名称用 Mux4to1b4_sch。

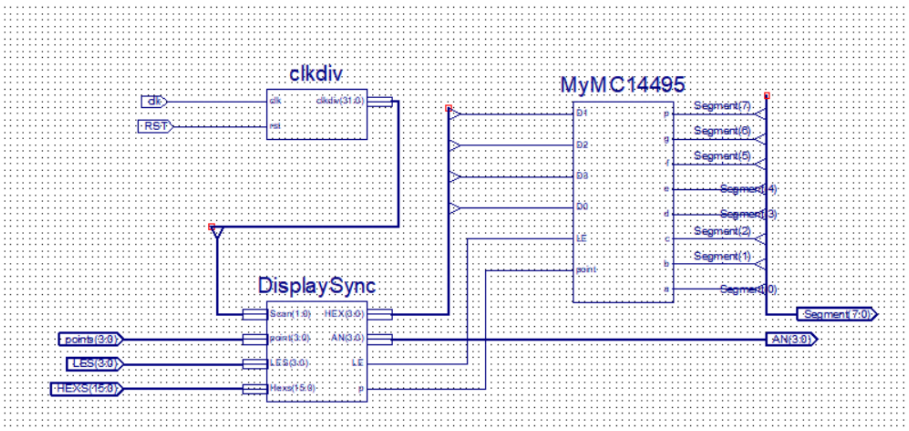
2 新建源文件，类型是 Schematic，文件名称用 Mux4to14b。



图表 12 Mux4to1b4 电路连线图

2、记分板应用设计

1 新建工程，工程名称用 ScoreBoard。
2 新建文件，文件名称用 DisplaySync.sch，利用原理图方式设计。
3 根据原理设计通用计数分频模块
4 新建源文件，类型是 Schematic，文件名称用 disp_num
5 原理图方式进行设计显示模块



图表 13 动态显示计分板

6 新建源文件 top，并右键设为“Top Module”
该文件调用其他模块，并第一个执行。

```
//////////////////////////////////////////////////////////////////////////////////
module top(input wire clk,
    input wire [7:0] SW,
    input wire [3:0] btn,
    output wire [3:0] AN,
    output wire [7:0] SEGMENT
);
    wire [15:0] num;

    CreateNumber c0(btn,num);

    disp_num d0(clk, num, SW[7:4], SW[3:0], 1'b0, AN, SEGMENT);

endmodule
```

图表 14 top 文件

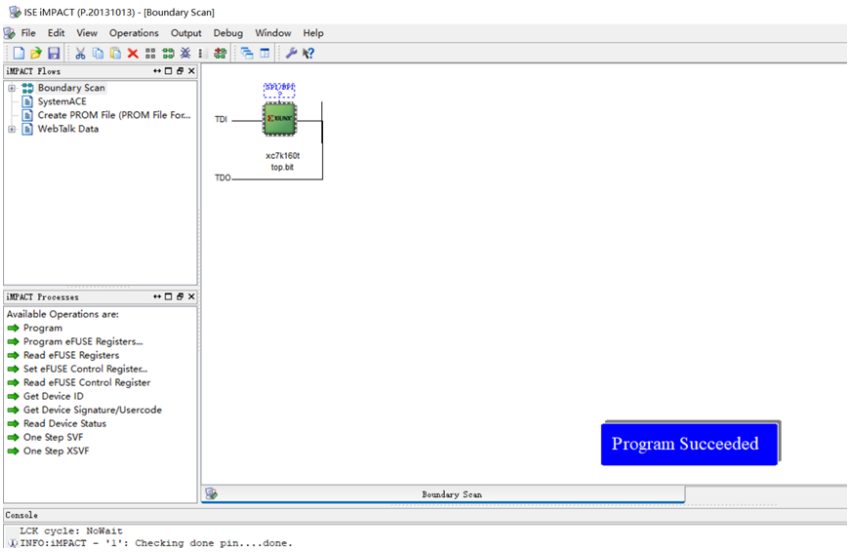
注：CreateNumber 模块在实验原理部分给出

7 UCF 引脚定义

```
NET "clk" LOC = AC18 | IOSTANDARD = LVCMOS18 ;
NET "SW[0]" LOC = Y13 | IOSTANDARD = LVCMOS15 ;
NET "SW[1]" LOC = Y12 | IOSTANDARD = LVCMOS15 ;
NET "SW[2]" LOC = AD11 | IOSTANDARD = LVCMOS15 ;
NET "SW[3]" LOC = AD10 | IOSTANDARD = LVCMOS15 ;
NET "SW[4]" LOC = AA10 | IOSTANDARD = LVCMOS15 ;
NET "SW[5]" LOC = AB10 | IOSTANDARD = LVCMOS15 ;
NET "SW[6]" LOC = AA13 | IOSTANDARD = LVCMOS15 ;
NET "SW[7]" LOC = AA12 | IOSTANDARD = LVCMOS15 ;

NET "btn[0]" LOC = AF8 | IOSTANDARD = LVCMOS15 ;
NET "btn[0]" clock_dedicated_route = false;
NET "btn[3]" LOC = AE13 | IOSTANDARD = LVCMOS15 ;
NET "btn[3]" clock_dedicated_route = false;
NET "btn[1]" LOC = AF13 | IOSTANDARD = LVCMOS15 ;
NET "btn[1]" clock_dedicated_route = false;
NET "btn[2]" LOC = AF10 | IOSTANDARD = LVCMOS15 ;
NET "btn[2]" clock_dedicated_route = false;

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

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
```

图表 15 ucf 引脚定义

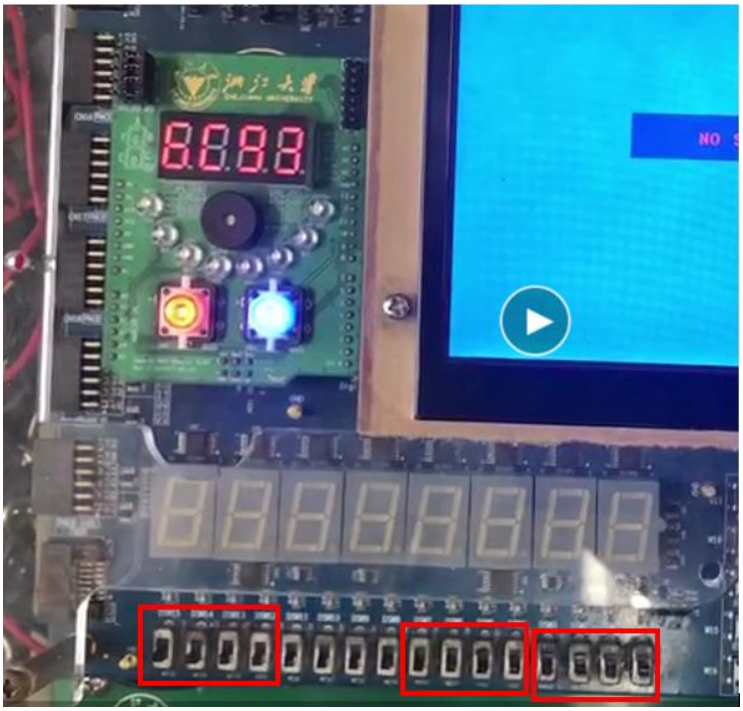
8 下载验证



图表 16 下载成功

五、实验结果与分析

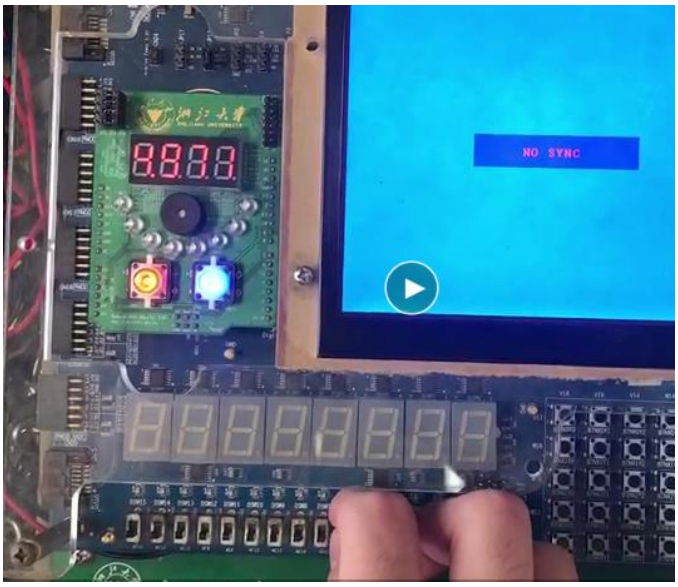
计分板应用设计



图表 17 开关对应功能

如上图所示，最左边的四个开关分别控制四个数字的加 1，中间方框中的四个开关控制小数点的显示，最右边的四个开关控制数字的显示。

图表 18 控制部分数字显示



图表 19 控制部分小数点显示

图表 20 控制某一数字加一（1）

图表 21 控制某一数字加一（2）

经验证，可以让任意一个数码管单独实现加一操作以及数字开关、小数点开关等操作。且数字可以从 0-f 变化。

六、讨论、心得

这次实验相较于前几次较为复杂，利用许多模块间的组合实现了功能，而这实际上是一种自顶向下的设计模式，先将大的任务拆分为若干模块，再调用模块来实现最终效果。其中，时分复用让每个数码管单独控制的操作比较让我印象深刻，实际上是通过很多时间内依次显示各个数码管的数字，利用人眼的暂留效应实现的，在我看来是一种非常巧妙的方法，这次实验做了比较长的时间，一直从下午做到晚上，但相应的也有比较大的收获。