

数字电路与数字系统实验

实验二 8-3 优先编码器

姓名： 你猜

学号： 你猜

班级： 你猜

邮箱： 你猜

实验时间： 你猜

一、实验目的

学习译码器、编码器和优先编码器的原理和实现方法，通过 Verilog 语言设计和实现这三种器件，并通过 FPGA 上的七段数码管显示输出的数字。

二、实验原理

编码器与译码器的功能相反，通常把来自于 2^n 条输入线的信息编码转换为 n 位二进制码。二进制编码器每次输入的 2^n 位信号中只能有一位为 1，其余均为 0。

而优先编码器允许同时在几个输入端都有输入信号，按输入信号排定的优先顺序，对信号中优先权最高的一个进行编码。即：

没有输入有效，则输出 IDLE 有效。

采用 HDL (如 Verilog) 中的语言结构说明优先编码器，相当地简单和自然，但是，为了更易于理解，先来看看采用逻辑方程的说明。为了写出优先编码器输出的逻辑方程，我们首先定义 8 个中间变量 $H_0 \sim H_7$ ，当且仅当 I_n 是值为 1 的输入中优先级最高的， H_n 才为 1：

$$\begin{aligned} H_7 &= I_7 \\ H_6 &= I_6 \cdot I_7' \\ H_5 &= I_5 \cdot I_6' \cdot I_7' \\ &\dots \\ H_0 &= I_0 \cdot I_1' \cdot I_2' \cdot I_3' \cdot I_4' \cdot I_5' \cdot I_6' \cdot I_7' \end{aligned}$$

注意，这是因为这些信号定义为任何时间最多只能有一个有效。采用这些符号，输出 $A_2 \sim A_0$ 的等式类似于简单二进制编码器的等式：

$$\begin{aligned} A_2 &= H_4 + H_5 + H_6 + H_7 \\ A_1 &= H_2 + H_3 + H_6 + H_7 \\ A_0 &= H_1 + H_3 + H_5 + H_7 \end{aligned}$$

如果没有输入为 1，那么输出 IDLE 为 1：

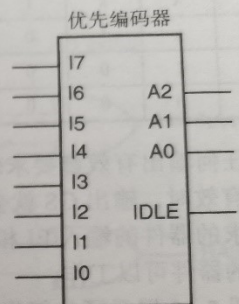
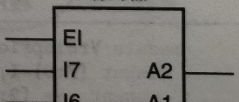
$$\begin{aligned} IDLE &= (I_0 + I_1 + I_2 + I_3 + I_4 + I_5 + I_6 + I_7)' \\ &= I_0' \cdot I_1' \cdot I_2' \cdot I_3' \cdot I_4' \cdot I_5' \cdot I_6' \cdot I_7' \end{aligned}$$


图 7-11 通用 8 输入优先编码器的逻辑符号



级联优先编码器

8-3 优先编码器真值表：

输入									输出		
E1	I0	I1	I2	I3	I4	I5	I6	I7	A2	A1	A0
0	x	x	x	x	x	x	x	x	0	0	0
1	x	x	x	x	x	x	x	1	1	1	1
1	x	x	x	x	x	x	1	0	1	1	0
1	x	x	x	x	1	0	0	0	1	0	1
1	x	x	x	x	1	0	0	0	1	0	0
1	x	x	x	1	0	0	0	0	0	1	1
1	x	x	1	0	0	0	0	0	0	1	0
1	x	1	0	0	0	0	0	0	0	0	1
1	1	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0

从上表中可以看出，I7 优先级最高，然后是 I6，…，最后是 I0。

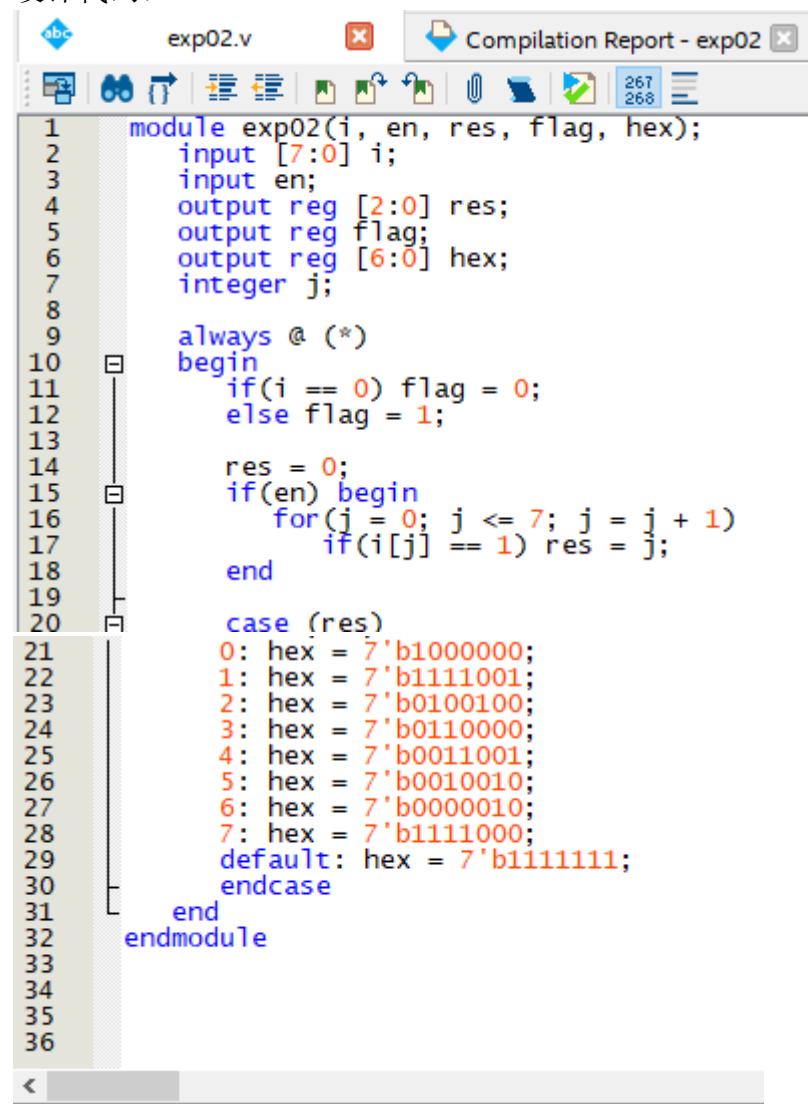
三、实验环境

Quartus 18.1、FPGA 开发板

四、实验过程

设计思路：可以参考实验手册中 4-2 优先编码器的代码，使用 for 循环来实现 8-3 优先编码器。同时，设置一个指示位，当输入全为 0 时，指示位也为 0，否则为 1。并使用 case 语句来判断输出所对应的七段数码管的显示。

设计代码：

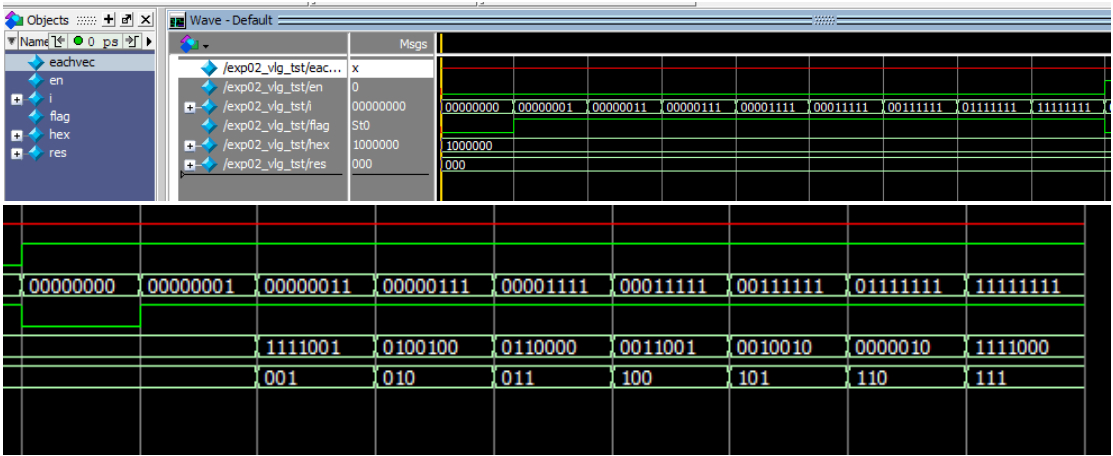


```
1 module exp02(i, en, res, flag, hex);
2     input [7:0] i;
3     input en;
4     output reg [2:0] res;
5     output reg flag;
6     output reg [6:0] hex;
7     integer j;
8
9     always @ (*)
10    begin
11        if(i == 0) flag = 0;
12        else flag = 1;
13
14        res = 0;
15        if(en) begin
16            for(j = 0; j <= 7; j = j + 1)
17                if(i[j] == 1) res = j;
18        end
19
20        case (res)
21            0: hex = 7'b1000000;
22            1: hex = 7'b1111001;
23            2: hex = 7'b0100100;
24            3: hex = 7'b0110000;
25            4: hex = 7'b0011001;
26            5: hex = 7'b0010010;
27            6: hex = 7'b0000010;
28            7: hex = 7'b1111000;
29            default: hex = 7'b1111111;
30        endcase
31    end
32 endmodule
33
34
35
36
```

激励代码：

```
18 );
19 initial
20 begin
21 // code that executes only once
22 // insert code here --> begin
23 en = 1'b0; i = 8'b00000000; #20;
24 i = 8'b00000001; #20;
25 i = 8'b00000011; #20;
26 i = 8'b00000111; #20;
27 i = 8'b00001111; #20;
28 i = 8'b00011111; #20;
29 i = 8'b00111111; #20;
30 i = 8'b01111111; #20;
31 i = 8'b11111111; #20;
32
33 en = 1'b1; i = 8'b00000000; #20;
34 i = 8'b00000001; #20;
35 i = 8'b00000011; #20;
36 i = 8'b00000111; #20;
37 i = 8'b00001111; #20;
38 i = 8'b00011111; #20;
39 i = 8'b00111111; #20;
40 i = 8'b01111111; #20;
41 i = 8'b11111111; #20;
42
43 // --> end
44 $display("Running testbench");
45 end
46 always
47 // optional sensitivity list
48 // @(event1 or event2 or ... eventn)
```

ModelSim 仿真：

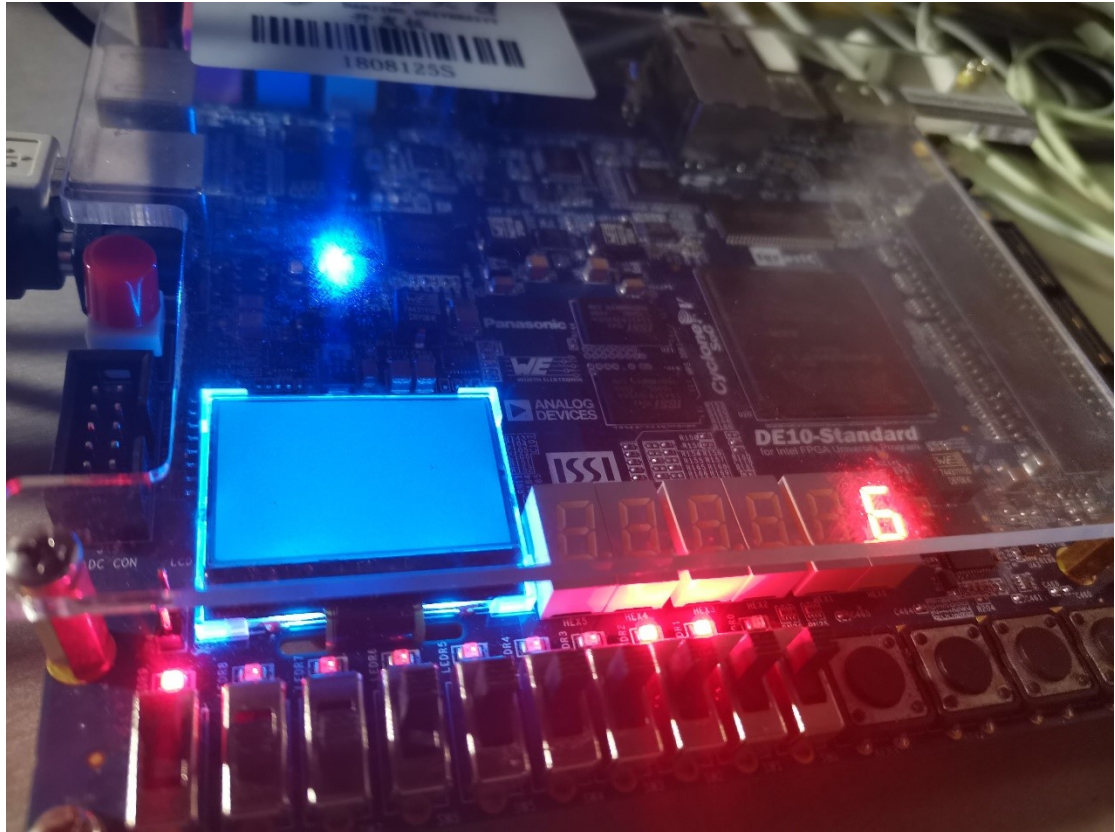


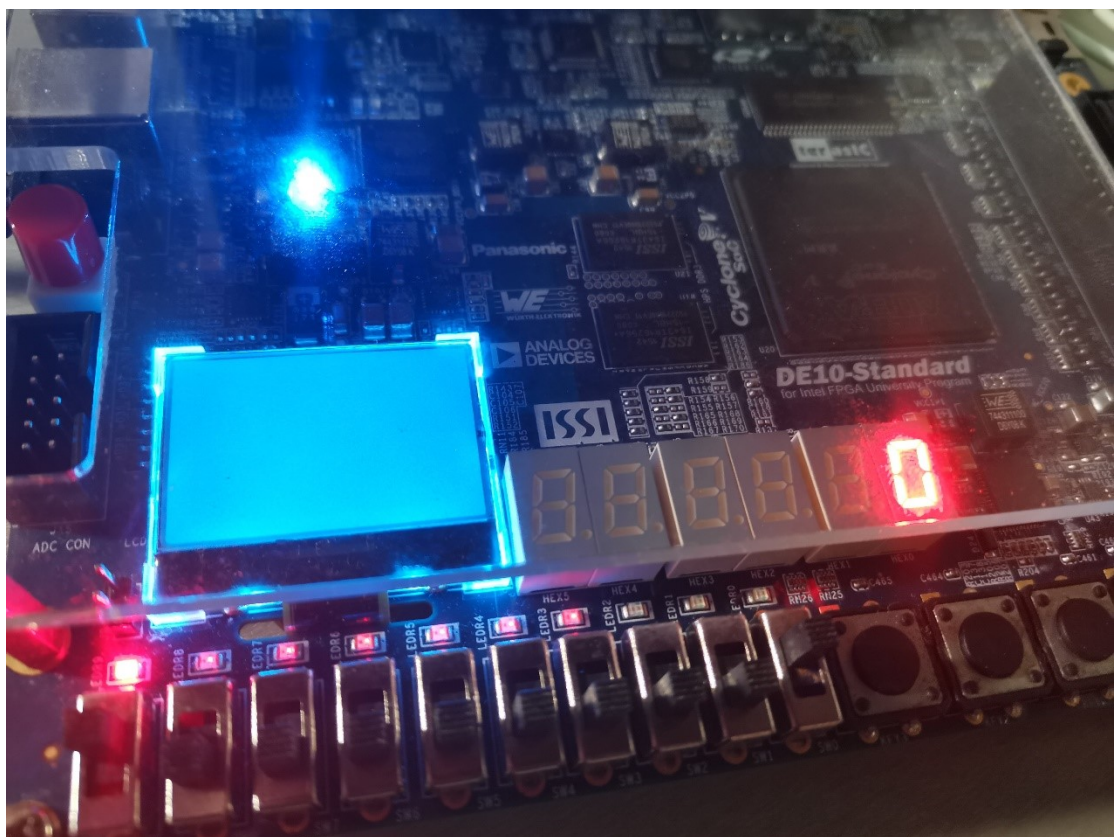
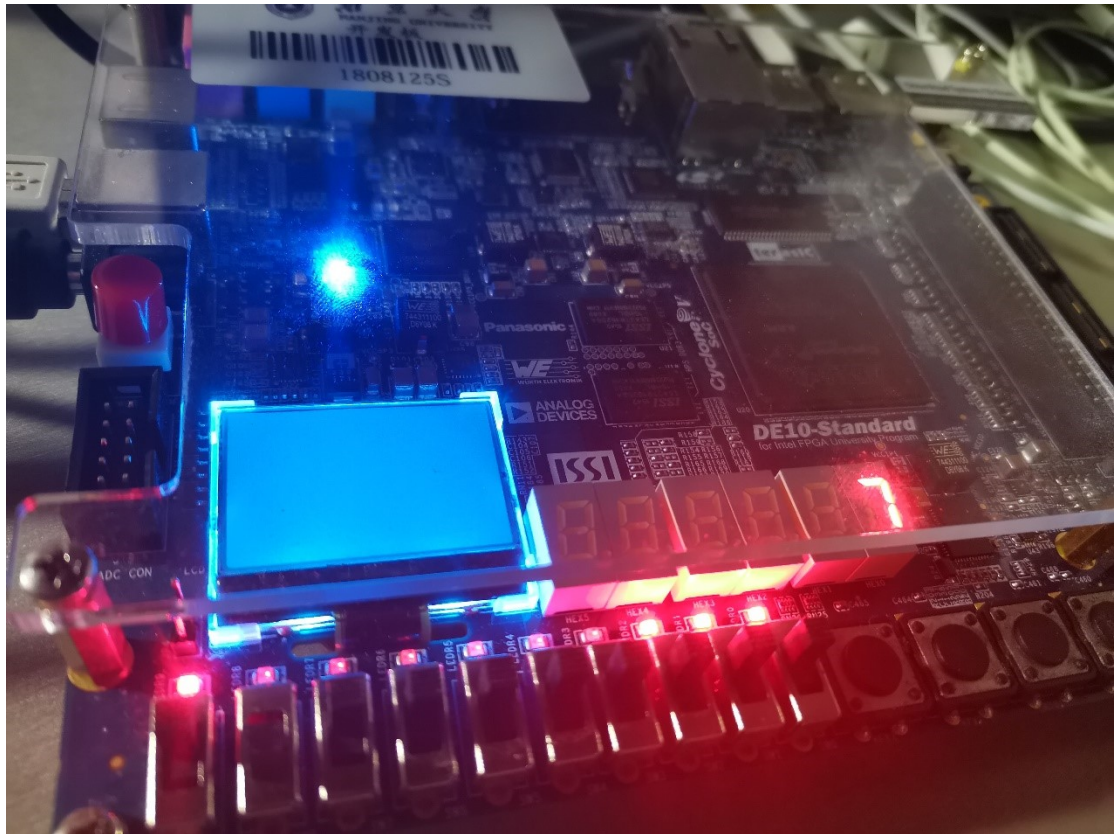
引脚分配：

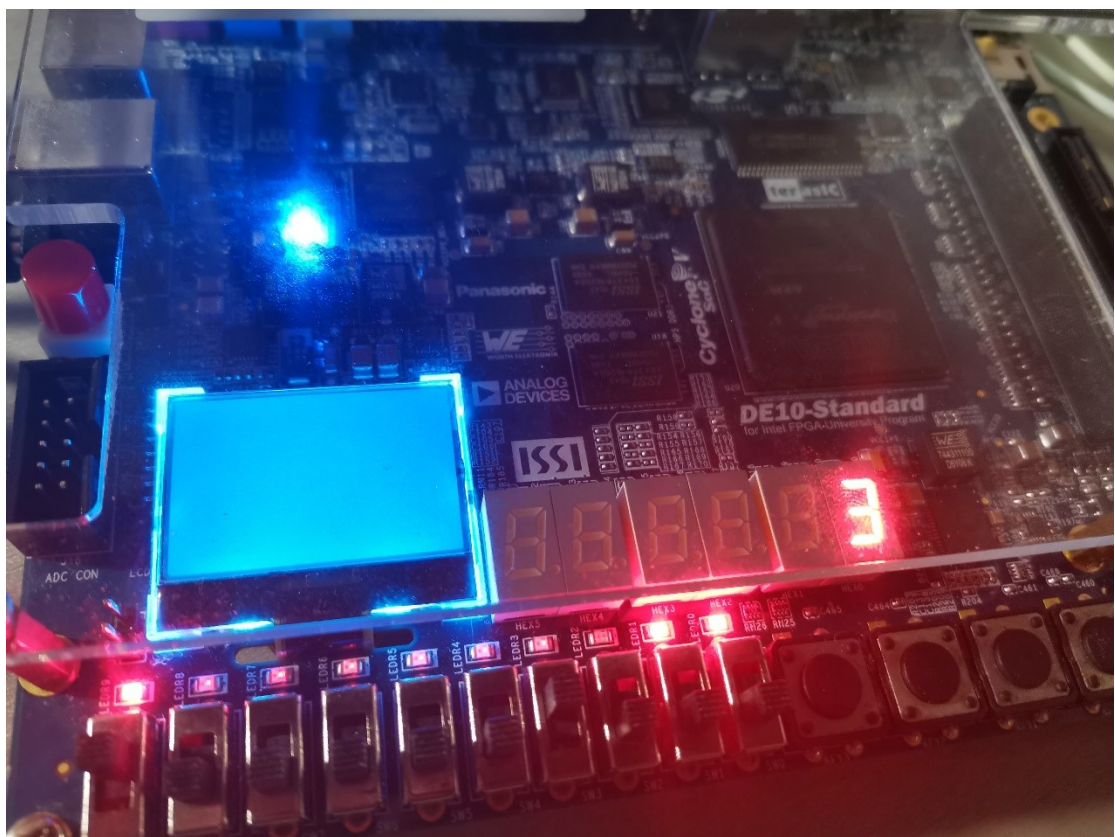
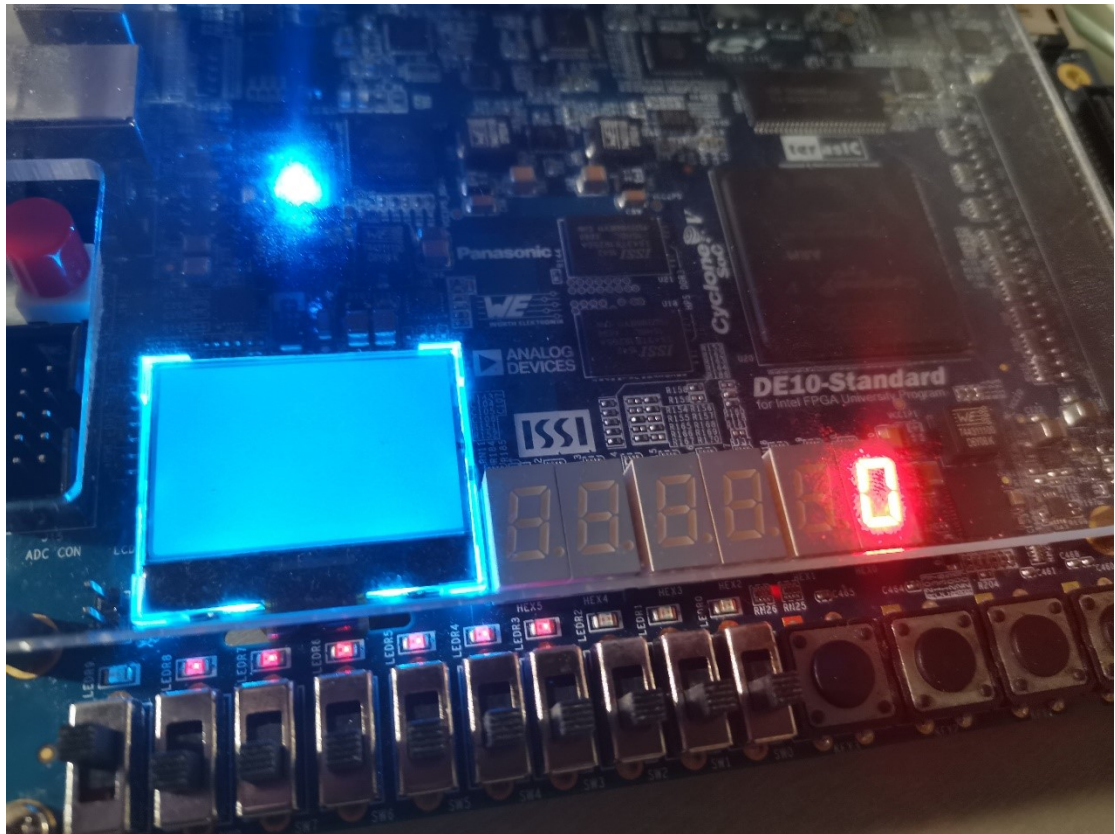
Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location	I/O Standard	Reserved	Current Strength	Slew Rate
en	Input	PIN_AA30	5B	B5B_NO	PIN_AA30	2.5 V		12mA (default)	
flag	Output	PIN_AC22	4A	B4A_NO	PIN_AC22	2.5 V		12mA (default)	1 (default)
hex[6]	Output	PIN_AH18	4A	B4A_NO	PIN_AH18	2.5 V		12mA (default)	1 (default)
hex[5]	Output	PIN_AG18	4A	B4A_NO	PIN_AG18	2.5 V		12mA (default)	1 (default)
hex[4]	Output	PIN_AH17	4A	B4A_NO	PIN_AH17	2.5 V		12mA (default)	1 (default)
hex[3]	Output	PIN_AG16	4A	B4A_NO	PIN_AG16	2.5 V		12mA (default)	1 (default)
hex[2]	Output	PIN_AG17	4A	B4A_NO	PIN_AG17	2.5 V		12mA (default)	1 (default)
hex[1]	Output	PIN_V18	4A	B4A_NO	PIN_V18	2.5 V		12mA (default)	1 (default)
hex[0]	Output	PIN_W17	4A	B4A_NO	PIN_W17	2.5 V		12mA (default)	1 (default)
i[7]	Input	PIN_AD30	5B	B5B_NO	PIN_AD30	2.5 V		12mA (default)	
i[6]	Input	PIN_AC28	5B	B5B_NO	PIN_AC28	2.5 V		12mA (default)	
i[5]	Input	PIN_V25	5B	B5B_NO	PIN_V25	2.5 V		12mA (default)	

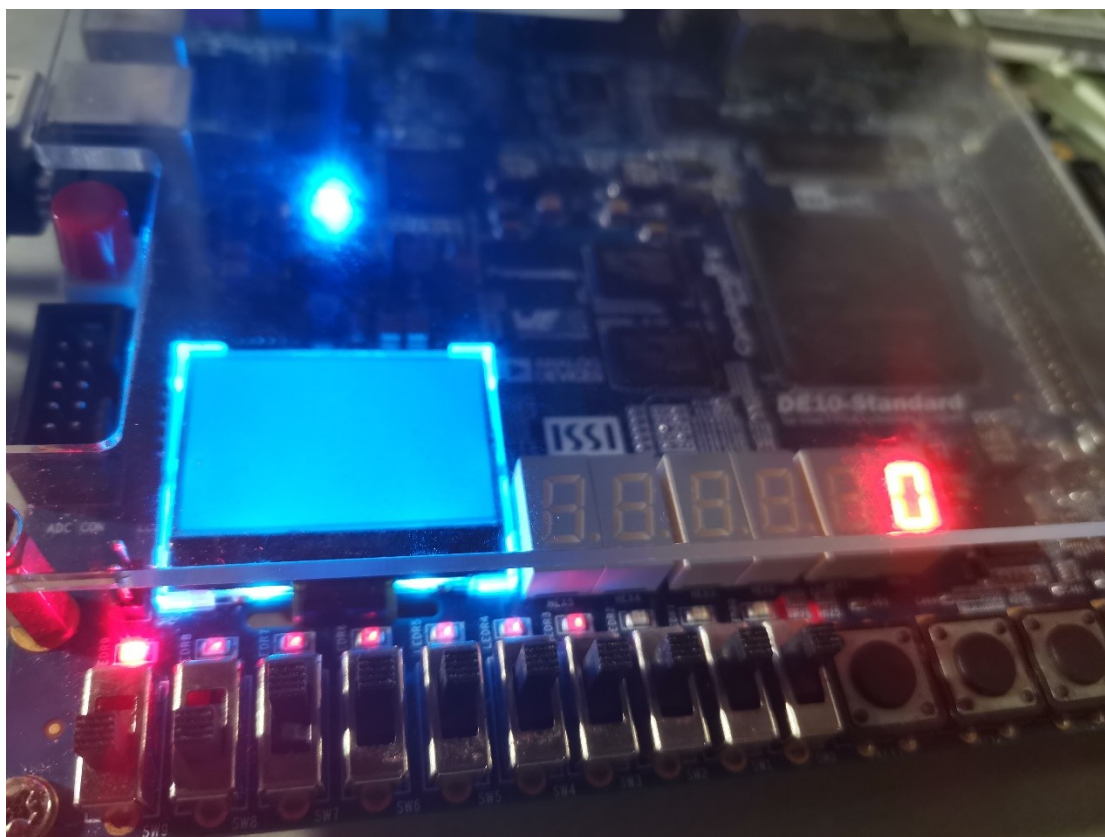
in	i[4]	Input	PIN_W25	5B	B5B_NO	PIN_W25	2.5 V		12mA (default)	
in	i[3]	Input	PIN_AC30	5B	B5B_NO	PIN_AC30	2.5 V		12mA (default)	
in	i[2]	Input	PIN_AB28	5B	B5B_NO	PIN_AB28	2.5 V		12mA (default)	
in	i[1]	Input	PIN_Y27	5B	B5B_NO	PIN_Y27	2.5 V		12mA (default)	
in	i[0]	Input	PIN_AB30	5B	B5B_NO	PIN_AB30	2.5 V		12mA (default)	
out	res[2]	Output	PIN_AC23	4A	B4A_NO	PIN_AC23	2.5 V		12mA (default)	1 (default)
out	res[1]	Output	PIN_AB23	5A	B5A_NO	PIN_AB23	2.5 V		12mA (default)	1 (default)
out	res[0]	Output	PIN_AA24	5A	B5A_NO	PIN_AA24	2.5 V		12mA (default)	1 (default)
<<new node>>										

实验结果：









五、 实验中遇到的问题及解决办法

- 1、没有认真读实验手册关于数码管的部分，没有注意到数码管是共阳极，即引脚输出逻辑 0 被点亮，导致数码管显示错误。经过后来对实验手册的认真阅读，解决了这个问题。
- 2、注意 begin、end 和 case、endcase 的配对使用，不要老是忘记写 end 和 endcase!!!

六、 启示

从老师发的《从算法设计到硬线逻辑的实现》第三章可以看到，casez 语句用来处理不考虑高阻值 z 的比较过程，casex 语句则将高阻值 z 和不定值都视为不必关心的情况。下面是 case、casez、casex 语句的真值表：

case	0	1	x	z	casez	0	1	x	z	casex	0	1	x	z
0	1	0	0	0	0	1	0	0	1	0	1	0	1	1
1	0	1	0	0	1	0	1	0	1	1	0	1	1	1
x	0	0	1	0	x	0	0	1	1	x	1	1	1	1
z	0	0	0	1	z	1	1	1	1	z	1	1	1	1

所以可以使用 casex 来实现优先编码器，代码如下：


```

res = 0;
if(en) //begin
    //for(j = 0; j <= 7; j = j + 1)
    //if(i[j] == 1) res = j;
//end
    casex (i)
        8'b1??????: res = 3'b111;
        8'b01?????: res = 3'b110;
        8'b001?????: res = 3'b101;
        8'b0001????: res = 3'b100;
        8'b00001??? : res = 3'b011;
        8'b000001?? : res = 3'b010;
        8'b0000001? : res = 3'b001;
        default: res = 3'b000;
    endcase

```

七、 意见与建议

虽然之前我并没有上过数字电路这门课，但是实验指南前面的两个例子非常好，由浅入深，帮助我学习和完成了这次实验。