南京大学

数电实验二：译码器和编码器

课程名称：数字逻辑与计算机组成实验

姓名：孙文博

学号：201830210

班级：数电一班

邮箱：201830210@smail.nju.edu.cn

实验时间：2022.3.8

一、 实验目的

1. 复习数字电路中译码器和编码器的相关知识

2. 掌握译码器和编码器的工作原理

3. 熟悉 Verilog 语言中 for 循环的使用

4. 设计一个 8-3 优先级编码器并通过七段数码管显示

二、 实验环境

设计\编译环境：Quartus (Quartus Prime 17.1) Lite Edition

开发平台：DE10-Standard

FPGA 芯片：Cyclone II 5CSXFC6D6

三、 实验原理

译码器是组合逻辑电路中的一个重要器件，它可以将某一输入信息转换为某一特定输出，通常是多路输入/输出电路，输入编码和输出编码之间存在着一一对应的映射关系，即每个输入编码产生唯一一个不同于其他的输出编码；编码器则是一种与译码器功能相反的逻辑电路，编码器的输出编码比其输入编码位数少，它把二进制码按一定的规律编排，例如 8421 码、格雷码等，使每组代码具有特定的含义（代表某个数或控制信号）。

常用的二进制译码器是一个有 n 路输入和 $m = 2^n$ 路输出的逻辑电路。译码器有一个使能信号 En，当 En=0 时，无论输入是多少，译码器都没有有效值输出；当 En=1 时，输入的值决定了输出信号的值。

在二进制码中，最常用的输出编码是 m = 2^n 位中取 1 位编码，即任何时刻，m 位输出编码中只能有 1 位有效数字为 1，其余各位都为 0，这样的二进制编码也被称为独热编码（one-hot encoded）。

四、 实验过程

本次实验中我们需要实现一个 8-3 优先编码器并将其在七段数码管上显示，实验步骤如下：

1. 基于 for 循环语句实现 8-3 优先编码器

首先我们要熟悉 Verilog 语法中的 for 循环语句，其格式如下：

```
for ( initial_assignment; condition; step_assignment )
```

图表 1：Verilog 中的 for 循环语句

其中初始赋值 initial_assignment 给出循环变量的初始值；condition 条件表达式指定循环在什么情况下必须结束，只要条件为真，循环中的语句就执行，它的值必须为常数；step_assignment 给出要修改的赋值，通常为增加或减少循环变量计数。值得注意的是，与 C 语言语法不同，这里的变量自增不能简写为 i++格式，而要使用完整的 i = i+1，否则编译会报错。

仿照实验手册我们可以写出 8-3 优先编码器的 Verilog 代码：

```
module exp_2(x,en,y);
    input [7:0] x;
    input en;
    output reg [2:0] y;
    integer i;
    always @(x or en) begin
        if(en) begin
            y=0;
            for(i=0;i<=7;i=i+1)
             if(x[i]==1)
               y=i;
        end
        else y=0;
    end
endmodule
```

图表 2：8-3 优先编码器

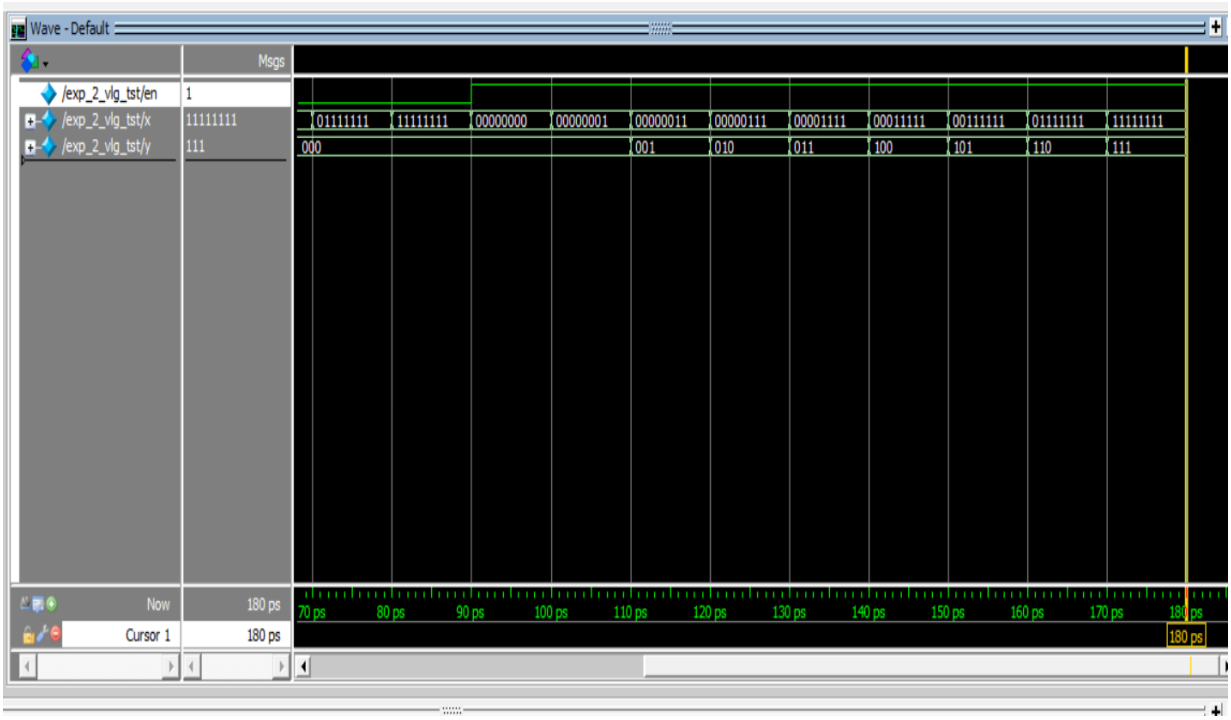
其中使用 for 循环的原理是，由低到高依次检测输入 X 的各位数值，若某一位数值为 1 则更新 Y 的值为当前位；若之后有更高位的值为 1 则再次更新 Y 的值，从而实现“优先级”的要求。输入 en 为使能端，en 为 0 的时候 Y 恒为 0，en 为 1 的时候编码器才工作。

2. ModelSim 仿真检测

接着为以上代码添加 testbench 测试文件，在 ModelSim 软件中进行仿真模拟：

```
begin
    en=1'b0;
    x=8'b00000000;#10;
    x=8'b00000001;#10;
    x=8'b00000011;#10;
    x=8'b00000111;#10;
    x=8'b00001111;#10;
    x=8'b00011111;#10;
    x=8'b00111111;#10;
    x=8'b01111111;#10;
    x=8'b11111111;#10;

    en=0'b1;
    x=8'b00000000;#10;
    x=8'b00000001;#10;
    x=8'b00000011;#10;
    x=8'b00000111;#10;
    x=8'b00001111;#10;
    x=8'b00011111;#10;
    x=8'b00111111;#10;
    x=8'b01111111;#10;
    x=8'b11111111;#10;

end
endmodule
```

图表 3：Testbench 测试文件

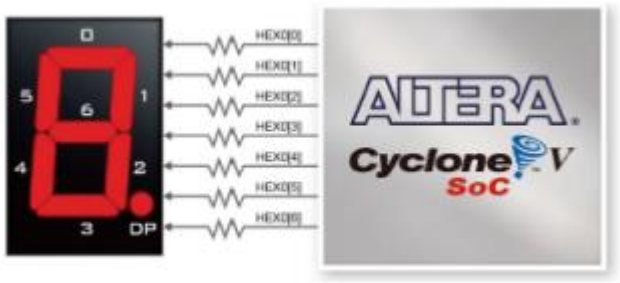


图表 4：ModelSim 仿真测试

由以上测试看出，当输入 X 中含有 1 的位数的升高，Y 取含 1 最高位的位数，实现了独热码的转换。

3. 连接七段数码管

七段 LED 数码管是一种常见的显示元件，用于显示数值。DE10-Standard 开发板上的数码管连接如图所示，我们可以使用一个 wire[6:0] LED 变量控制七段数码管，其中 1 表示暗，0 表示亮，数位是从低到高(这里 debug 了好久,因为这个设定不符合常理啊😭)。



图表 5：DE10-Standard 数码管连接

加入数码管输入后的代码如下：

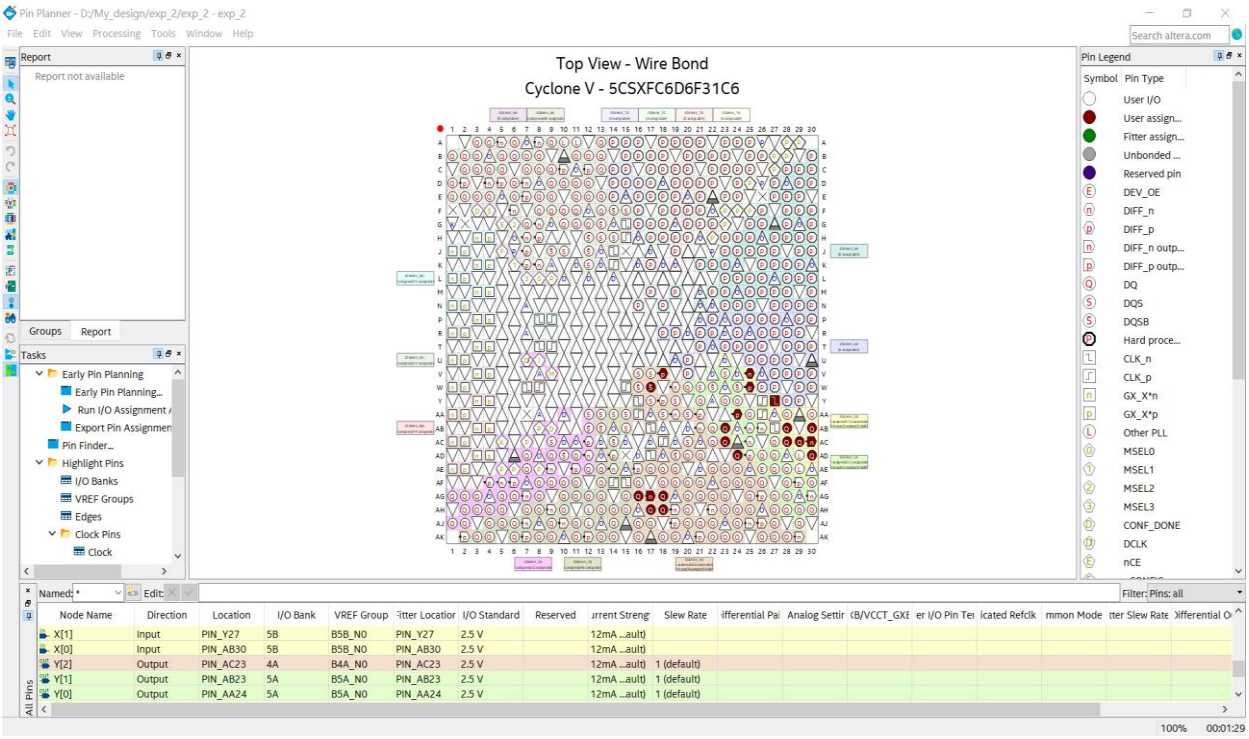
```
module exp_2(
    input  [7:0] X,
    input  en,
    output reg valid,
    output [6:0] F
    );

     integer i;
    reg [2:0] y;
    reg [6:0] f;
    always @(*) begin
       y=0;valid=0;
       if(en==0) begin
          f=7'b1111111;
          valid=0;
       end

       else begin
          for(i=0;i<=7;i=i+1)
          if(X[i]==1) begin
             y=i;
             valid=1;
          end
          case(y)
          3'b000: begin
             if(valid==0)
                f=7'b1111111;
             else
                f=7'b1000000;
          end
          3'b001: f=7'b1111001;
          3'b010: f=7'b0100100;
          3'b011: f=7'b0110000;
          3'b100: f=7'b0011001;
          3'b101: f=7'b0010010;
          3'b110: f=7'b0000010;
          3'b111: f=7'b1111000;
       endcase
       end
    end
    assign F=f;
```

图表 6：加入七段数码管后的优先编码器

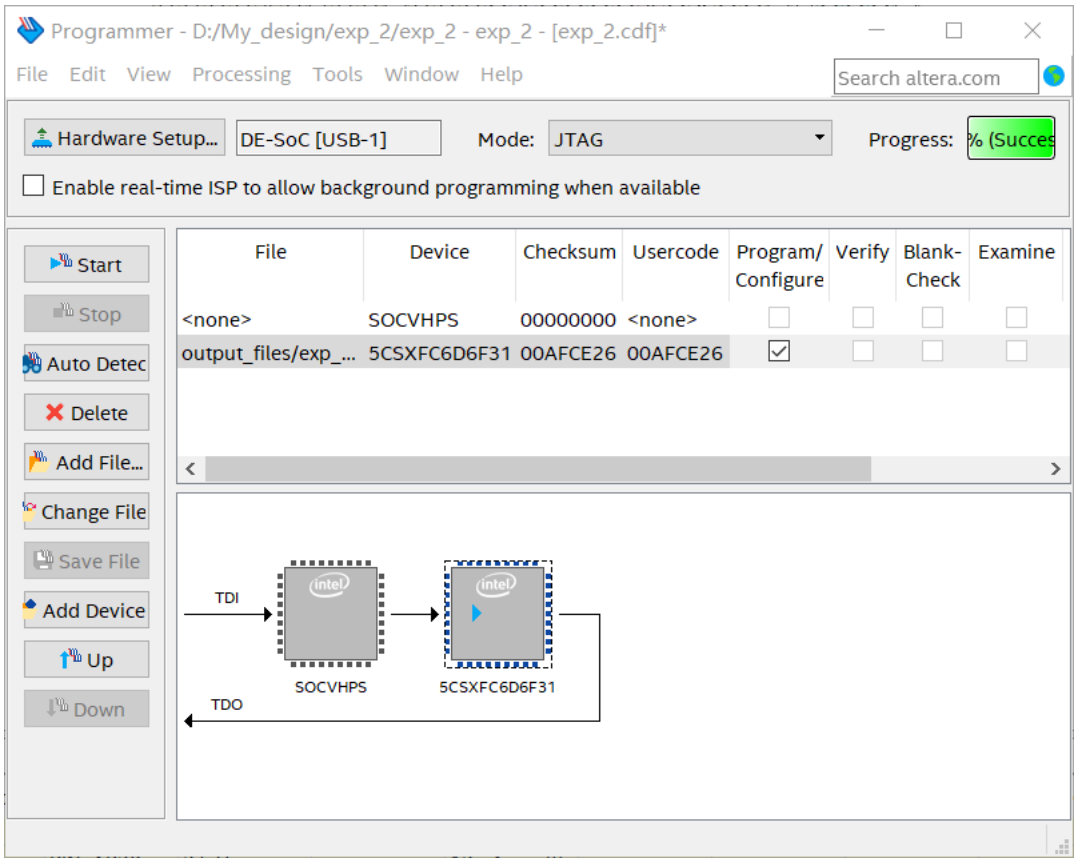
五、 实验结果

1. 上板验收

对我们写好的代码进行编译处理，分配好引脚并将烧录好的文件下载到 DE10-Standard 开发板上验证：

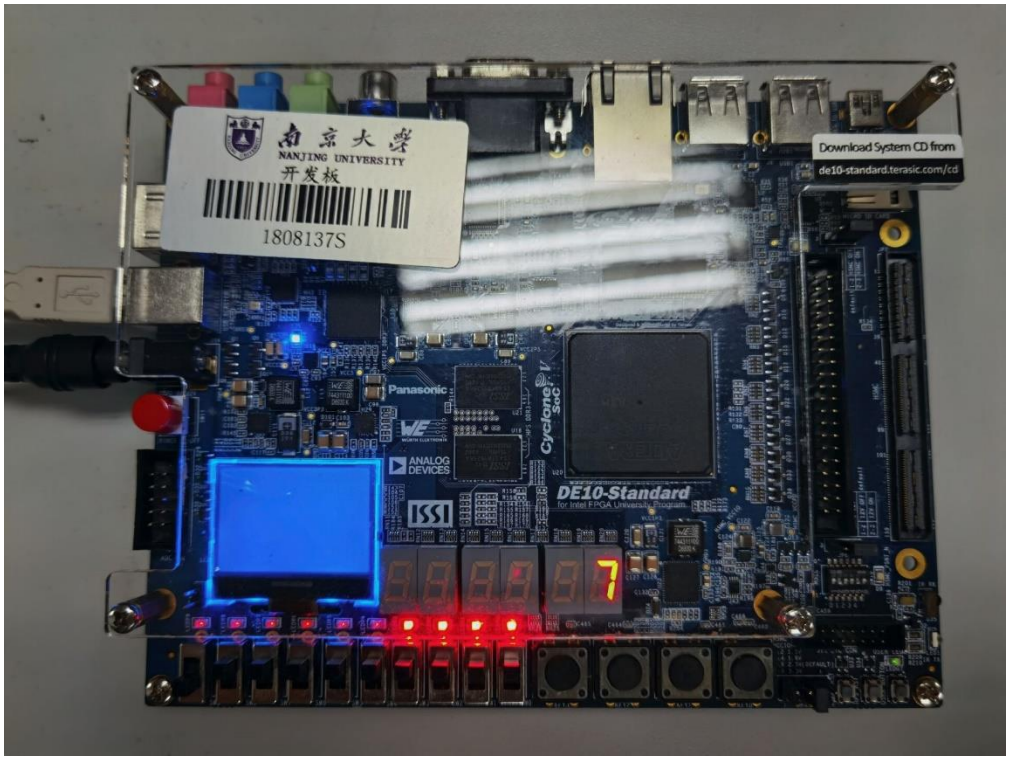


图表 7：引脚分配



图表 8：配置芯片

最后上板验收，实验成功！



图表 9：上板验收

2. 思考题：casex 与 casez 语句

casez 与 casex 语句是 case 语句的两种变体，下表给出这三种语句的区别：

case	0	1	x	z
0	1	0	0	0
1	0	1	0	0
x	0	0	1	0
z	0	0	0	1

casez	0	1	x	z
0	1	0	0	1
1	0	1	0	1
x	0	0	1	1
z	1	1	1	1

casex	0	1	x	z
0	1	0	1	1
1	0	1	1	1
x	1	1	1	1
z	1	1	1	1

图表 10：casex 语句，casez 语句和 case 语句的不同

在 case 语句中，敏感表达式中与各项值之间的比较是一种全等比较，每一位都相同才认为匹配；在 casez 语句中，如果分支表达式某些位的值为高阻 z，那么对这些位的比较就会忽略，不予考虑，而只关注其他位的比较结果；在 casex 语句中，则把这种处理方式进一步扩展到对 x 的处理，即如果比较双方有一方的某些位的值是 z 或 x，那么这些位的比较就不予考虑。

用 casex 语句实现的 8-3 优先级编码器如下：

```
casex(X)    //逻辑值x表示输入信号中那些位可以为任意值
  8'b0000_0001: Y <= 3'b000;

  8'b0000_001x: Y <= 3'b001;

  8'b0000_01xx: Y <= 3'b010;

  8'b0000_1xxx: Y <= 3'b011;

  8'b0001_xxxx: Y <= 3'b100;

  8'b001x_xxxx: Y <= 3'b101;

  8'b01xx_xxxx: Y <= 3'b110;

  8'b1xxx_xxxx: Y <= 3'b111;

  default:begin
            Y <= 3'b000;
            end  //处理编码器输入信号无效的情况
  endcase
```

图表 11：casex 语句实现

六、 总结与反思

本次实验中通过熟悉译码器和编码器的原理，实现一个 8-3 优先级编码器，同时在开发板上使用七段数码管显示结果，加深了我们对数字电路基础及 Quartus 仿真软件验收流程的理解，在实验过程中虽然有一些卡顿 (比如七段数码管的编码容易写反顺序)，但总体来说还是比较顺利的，希望接下来的实验可以再接再厉！💪