

Lab1-2 Report

译码管设计：将 0-15 的二进制表示放入数组 decoded 中，LE 为总开关，若 LE 为 0，则输出 data,若 LE 为 1，则输出全 1，意即关闭所有的数码管。而 p 表示小数点是否显示，在 LE 为 0 且 point 有输入时，显示小数点。

```
testbench.v Mux4T1_32.v SegDecoder.v X Gen
src > lab1-2 > submit > SegDecoder.v
18 assign {a, b, c, d, e, f, g} = ~LE ? ma[data] : 7'b1111111;
19 assign p = (LE == 0) ? ~point : 1'b1;
20 reg [7:0] decoded[0:15];
21
22 initial begin
23     // 初始化十六进制数字到7段LED的映射
24     // 由于是负逻辑，所以亮为0，灭为1
25     decoded[0] = 8'b0000001; // 0
26     decoded[1] = 8'b1001111; // 1
27     decoded[2] = 8'b0010010; // 2
28     decoded[3] = 8'b0000110; // 3
29     decoded[4] = 8'b1001100; // 4
30     decoded[5] = 8'b0100100; // 5
31     decoded[6] = 8'b0100000; // 6
32     decoded[7] = 8'b0001111; // 7
33     decoded[8] = 8'b0000000; // 8
34     decoded[9] = 8'b0000100; // 9
35     decoded[10] = 8'b0001000; // A
36     decoded[11] = 8'b1100000; // b
37     decoded[12] = 8'b0110001; // C
38     decoded[13] = 8'b1000010; // d
39     decoded[14] = 8'b0110000; // E
```

复合多路选择器及语法分析比较：

1.? 语法 特点：简洁，而且便于书写，同时逻辑性强。

```
assign O = S[1] ?
(S[0]?I3:I2) :(S[0]?I1:I0);
```

2.index 语法 特点：对于所有结果，均将其存入数组 I 中，后根据输入的 S 即可直接从数组中取得结果。优点为较为直观，缺点是较为占用内存空间。

```
wire [31:0] I [3:0];

assign I[0] = I0;

assign I[1] = I1;

assign I[2] = I2;
```

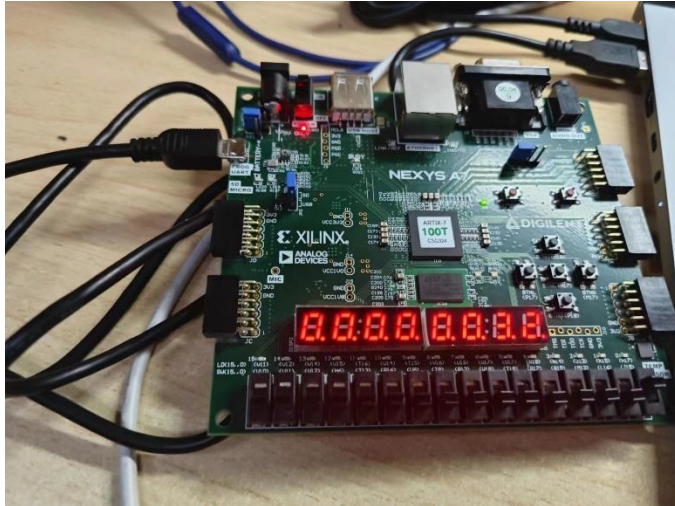
```
assign I[3] = I3;  
  
assign 0 = I[S];
```

3.if-else 语法：特点：较为符合 c 语言逻辑，更为贴近选择器的逻辑
但缺点是代码量较大，且 verilog 对 if-else 不太友好

```
always@(*)begin  
    if(S[1])begin  
        if(S[0]) 0<=I3;  
        else 0<=I2;  
    end else begin  
        if(S[0]) 0<=I1;  
        else 0<=I0;  
    end  
end
```

4.case 语法:与前一种语法较为贴近，也较为直观。

```
always@(*)begin  
    case(S)  
        2'b00:0<=I0;  
        2'b01:0<=I1;  
        2'b10:0<=I2;  
        2'b11:0<=I3;  
    endcase  
end
```

Testbench.v for 循环展开:

```
for(i=0;i<8;i=i+1)begin

    data=i[3:0];

    #5;

end

/*将 data 赋值为 0000-0111(对应十进制 0-7)

data = 4'b0000

#5

data = 4'b0001

#5

data = 4'b0010

#5

data = 4'b0011

#5

data = 4'b0100

#5

data = 4'b0101

#5

data = 4'b0110
```

```
#5  
  
data = 4'b0111  
  
#5  
  
*/
```

剩下 8-15 部分同理

For 语句即通过 i 的累加, i[3:0]赋值给 data, 实现对 data 的赋值。此处 i[3:0]可视为 data 的下标, 即十进制下 0-15.

多种多路选择器语法比较:

- 1.? 语法: 条件运算符适用于简单的条件选择, 尤其是在赋值语句中。
- 2.if-else 语法: if-else 适用于条件较为复杂的情况, 可分别处理每一个情况, 逻辑性较强
- 3.case 语法: 使用在多个选择的分支当中, 基于一个表达式的多个值做选择
- 4.index 语法: 直接访问某个特定选择, 查找效率较高

最喜欢的语法: ?语法 所需代码最简洁, 工作量最小