

中国科学技术大学计算机学院

《数字电路实验》报告



实验题目：编码器 and 译码器

学生姓名：柯志伟

学生学号：PB20061338

完成日期：2021 年 11 月 2 号

计算机实验教学中心制

2020 年 09 月

【实验题目】

编码器和译码器

【实验目的】

1. 熟悉并使用 Verilog 硬件描述语言描述电路的结构并构建电路
2. 熟悉普通编码器、优先编码器、普通译码器、七段译码器的电路结构与具体实现
3. 使用 Vivado 平台完成电路仿真，查看 RTL 分析和综合后电路图并查看综合后电路资源使用情况
4. 使用 Nexys4 开发板下载 Verilog 代码，实现编码器与译码器，熟悉使用开发板实现电路的流程

【实验环境】

PC 机
Vivado 平台
Nexys4 开发板

【实验过程】

1. 使用模块化设计，用 Verilog 语言实现普通编码器，优先编码器，普通译码器，7 段译码器的电路

(1) 首先实现普通编码器，优先编码器，普通译码器，7 段译码器的模块，具体代码如下：

```
1  module CommonEncoder (                                     //8421 码的10—4编码器
2      input wire [9:0] In ,
3      output reg [3:0] Out
4  );
5      always @(*)
6          begin
7              case (In)
8                  10'b0000_0000_01 : Out = 4'b0000;
9                  10'b0000_0000_10 : Out = 4'b0001;
10                 10'b0000_0001_00 : Out = 4'b0010;
11                 10'b0000_0010_00 : Out = 4'b0011;
12                 10'b0000_0100_00 : Out = 4'b0100;
13                 10'b0000_1000_00 : Out = 4'b0101;
14                 10'b0001_0000_00 : Out = 4'b0110;
15                 10'b0010_0000_00 : Out = 4'b0111;
16                 10'b0100_0000_00 : Out = 4'b1000;
17                 10'b1000_0000_00 : Out = 4'b1001;
18                 default : Out = 4'b1111;                    // 无效输入
19             endcase
```

```
20         end
21     endmodule
```

```
1     module PriorityEncoder(                                // 余3循环码的优先编码器
2     input wire [9:0] In ,
3     output reg [3:0] Out
4     );
5     always @(*) begin
6         if(In[9]) Out =4'b1010;
7         else if(In[8]) Out =4'b1110;
8         else if(In[7]) Out =4'b1111;
9         else if(In[6]) Out =4'b1101;
10        else if(In[5]) Out =4'b1100;
11        else if(In[4]) Out =4'b0100;
12        else if(In[3]) Out =4'b0101;
13        else if(In[2]) Out =4'b0111;
14        else if(In[1]) Out =4'b0110;
15        else if(In[0]) Out =4'b0010;
16        else Out = 4'b1111;                                // 无效输入
17    end
18 endmodule
```

```
1     module CommonDecoder(                                // 4-10 普通译码器
2     input wire [3:0] In ,
3     output reg [9:0] Out
4     );
5     always @(*)
6     begin
7         case(In)
8             4'b0000: Out = 10'b0000_0000_01;
9             4'b0001: Out = 10'b0000_0000_10;
10            4'b0010: Out = 10'b0000_0001_00;
11            4'b0011: Out = 10'b0000_0010_00;
12            4'b0100: Out = 10'b0000_0100_00;
13            4'b0101: Out = 10'b0000_1000_00;
14            4'b0110: Out = 10'b0001_0000_00;
15            4'b0111: Out = 10'b0010_0000_00;
16            4'b1000: Out = 10'b0100_0000_00;
17            4'b1001: Out = 10'b1000_0000_00;
18            default : Out = 10'b0000_0000_00;              // 无效输入
19        endcase
20    end
21 endmodule
```

```

1      module Decoder7Seg(           //7段译码管
2  input wire [3:0] In ,
3  output reg [6:0] Out
4      );
5  always @ (*)
6      begin
7      case(In)
8          4'b0000: Out = 7'b0000_001;
9          4'b0001: Out = 7'b1001_111;
10         4'b0010: Out = 7'b0010_010;
11         4'b0011: Out = 7'b0000_110;
12         4'b0100: Out = 7'b1001_100;
13         4'b0101: Out = 7'b0100_100;
14         4'b0110: Out = 7'b0100_000;
15         4'b0111: Out = 7'b0001_111;
16         4'b1000: Out = 7'b0000_000;
17         4'b1001: Out = 7'b0000_100;
18         default : Out = 7'b1111_111;
19     endcase
20     end
21 endmodule

```

(2) 接着使用以上模块实现综合的顶层模块，代码如下：

```

1      module Lab01_top(           // 普通编码器、普通译码器、七段译码器的组合
2  input wire [9:0] In ,
3  output wire [9:0] Out1 ,
4  output wire [6:0] Out2
5      );
6      wire [3:0] temp;
7      CommonEncoder cer (In ,temp);
8      CommonDecoder cdr (temp,Out1);
9      Decoder7Seg d7s (temp,Out2);
10 endmodule

```

```

1      module Lab01_top(           // 优先编码器、普通译码器、七段译码器的组合
2  input wire [9:0] In ,
3  output wire [9:0] Out1 ,
4  output wire [6:0] Out2
5      );
6      wire [3:0] temp;
7      PriorityEncoder per (In ,temp);

```

```

8         CommonDecoder cdr (temp, Out1);
9         Decoder7Seg d7s (temp, Out2);
10     endmodule
11 endmodule

```

2. 完成电路仿真和下载测试

(1) 创建测试文件，代码如下：

```

1     module testbench ();
2         reg [9:0] x;
3         wire [9:0] y;
4         wire [6:0] n; //your defination for Seven-segment digital tube
5         Lab01_top lab01(x, y, n) ;//your top mudule
6
7         initial begin
8             //for ordinary encoder
9             x = 10'b00_0000_0001;
10            #10 x = 10'b00_0000_0010;
11            #10 x = 10'b00_0000_0100;
12            #10 x = 10'b00_0000_1000;
13            #10 x = 10'b00_0001_0000;
14            #10 x = 10'b00_0010_0000;
15            #10 x = 10'b00_0100_0000;
16            #10 x = 10'b00_1000_0000;
17            #10 x = 10'b01_0000_0000;
18            #10 x = 10'b10_0000_0000;
19            //for priority encoder
20            #10 x = 10'b00_0000_0001;
21            #10 x = 10'b00_0000_0011;
22            #10 x = 10'b00_0000_0111;
23            #10 x = 10'b00_0000_1111;
24            #10 x = 10'b00_0001_1111;
25            #10 x = 10'b00_0011_1111;
26            #10 x = 10'b00_0111_1111;
27            #10 x = 10'b00_1111_1111;
28            #10 x = 10'b01_1111_1111;
29            #10 x = 10'b11_1111_1111;
30        end
31    endmodule

```

(2) 进行电路仿真，结果如下：

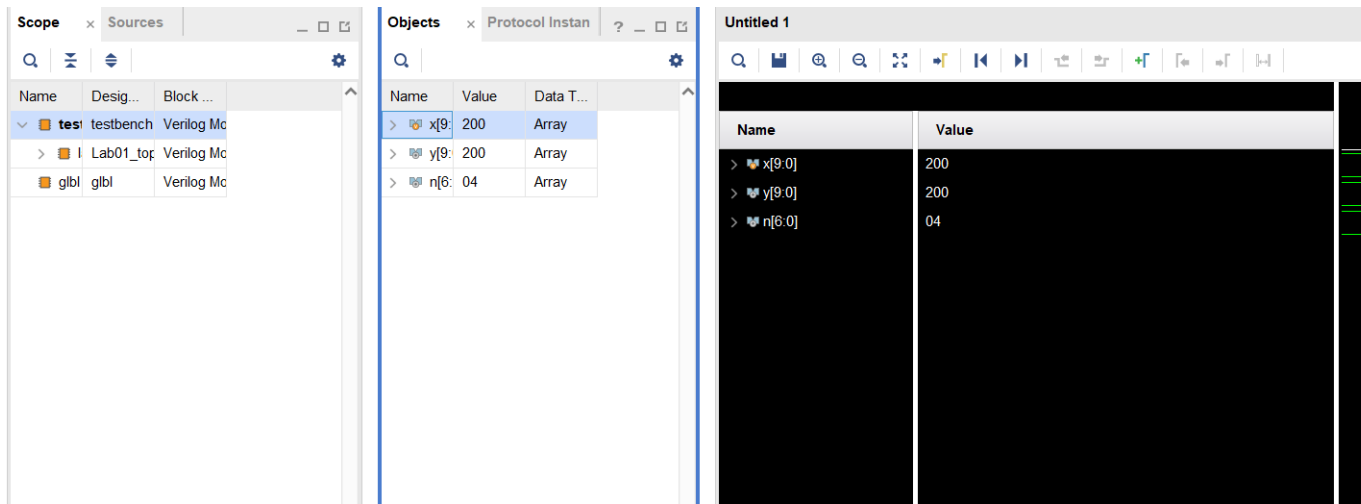


图 1: 普通编码器、普通译码器、七段译码器电路组合的仿真

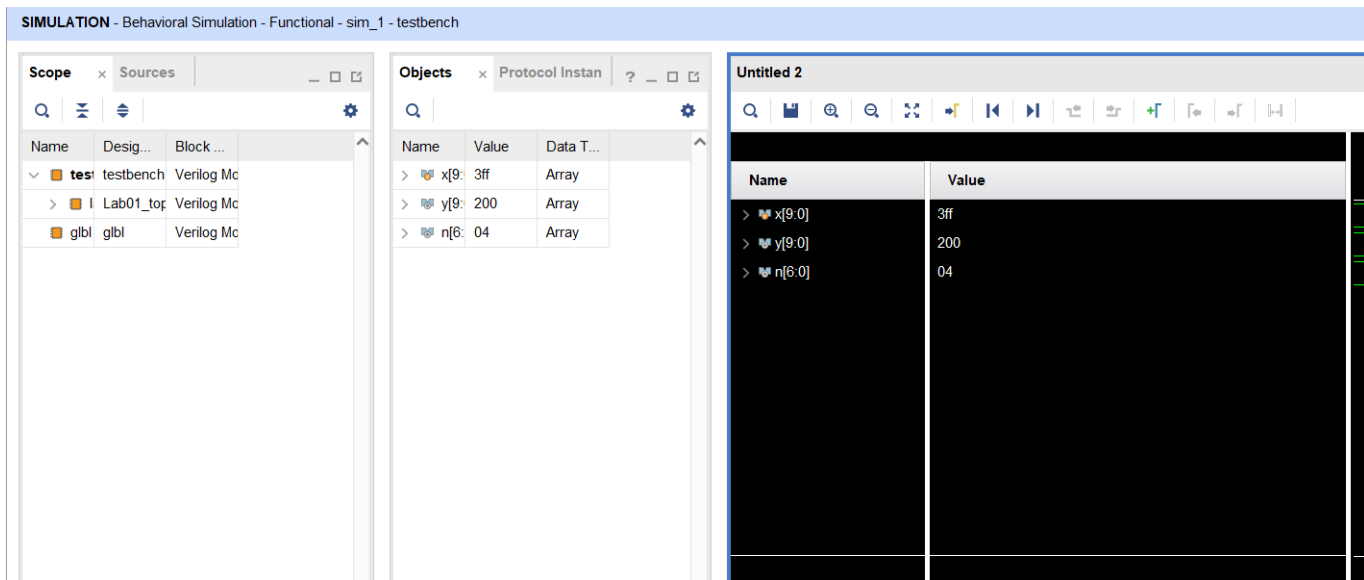


图 2: 优先编码器、普通译码器、七段译码器电路组合的仿真

(3) 添加约束文件，经综合实现并生成比特流文件，下载到开发板测试，结果如下：

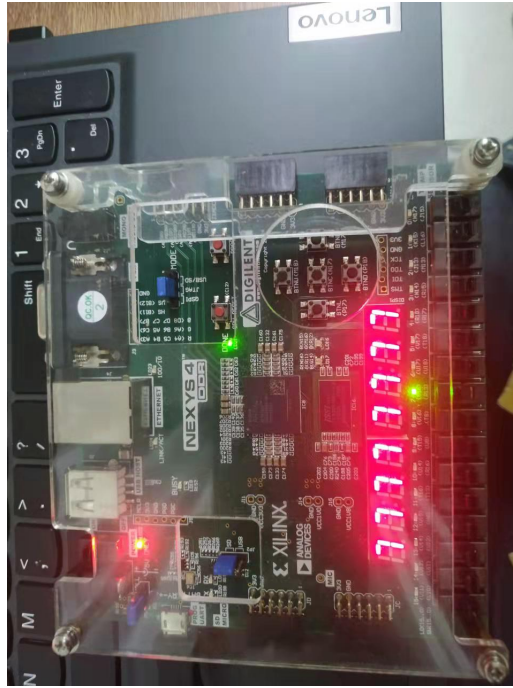


图 3: 普通编码器、普通译码器、七段译码器组合的下载测试

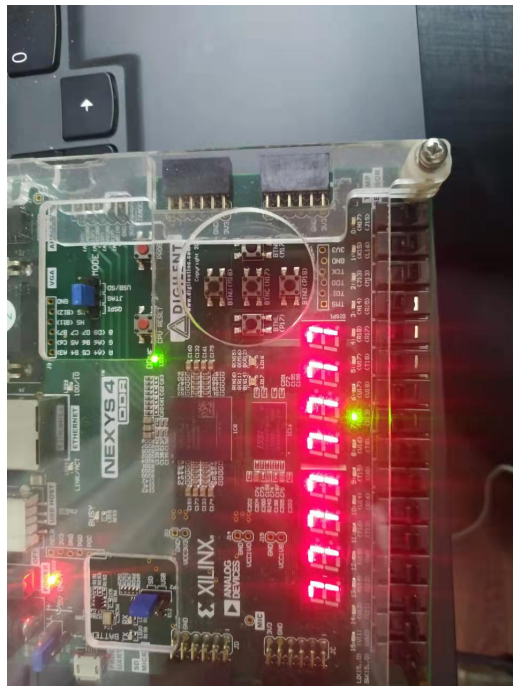


图 4: 优先编码器、普通译码器、七段译码器组合的下载测试

3. 查看 RTL 分析和综合后电路图

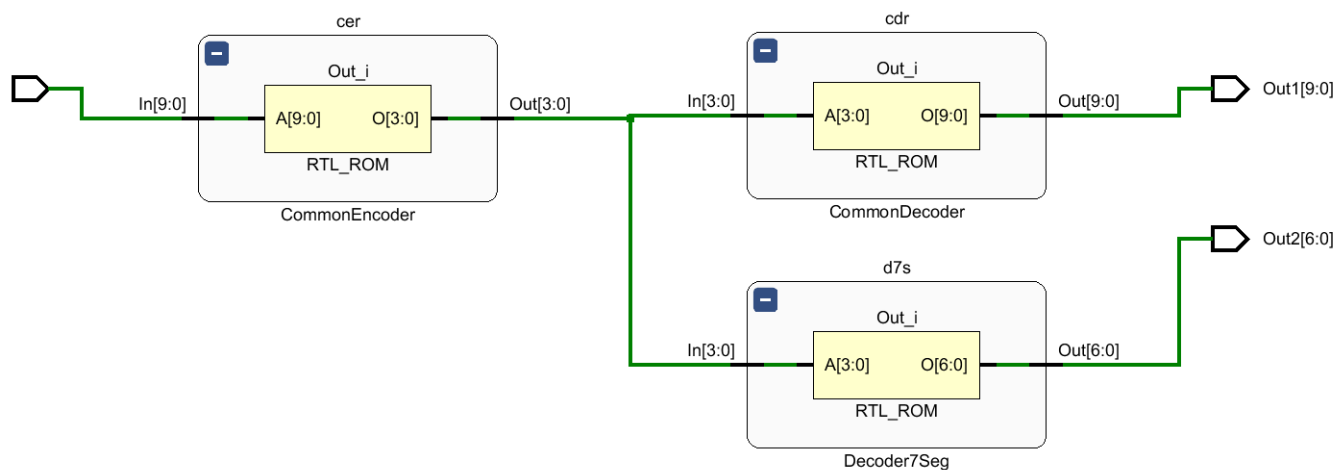


图 5: 普通编码器、普通译码器、七段译码器组合的电路图

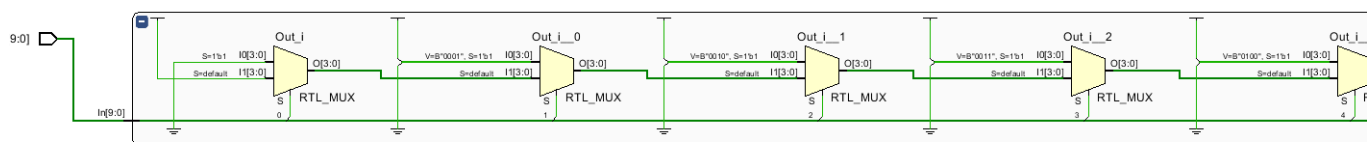


图 6: 优先编码器、普通译码器、七段译码器组合的电路图

4. 查看综合后电路资源使用情况

Hierarchy			
Name	1	Slice LUTs (63400)	Bonded IOB (210)
Lab01_top		30	27

图 7: 普通编码器、普通译码器、七段译码器组合的电路资源使用情况

Name	1	Slice LUTs (63400)	Slice (15850)	LUT as Logic (63400)	Bonded IPADs (2)
Lab01_top		20	7	20	27

图 8: 优先编码器、普通译码器、七段译码器组合的电路资源使用情况

【总结与思考】

总结：

通过此次实验，熟悉了 Vivado 平台的基本使用，Verilog 语言描述电路，大致了解了使用开发板实现电路的流程

思考：

1. 在此次实验中，发现在对 Verilog 语言描述电路过程中，自己对 Verilog 语言描述电路的一些注意事项不清楚，导致产生的电路图与设想中不同
2. 在使用 Vivado 过程中，对其中的一些报错信息，警告信息不太理解，需要加强这方面的经验