南京郵電大學
Nanjing University of Posts and Telecommunications

电工电子实验报告

课程名称：电工电子实验（二）

实验项目：可编程器件的应用及数字系统设计

学　　院：贝尔英才学院

班　　级：

学　　号：

姓　　名：

学　　期：2022-2023学年第1学期

可编程器件的应用及数字系统设计

一、实验目的

1.使用ISE软件完成组合逻辑设计的输人并仿真。
2.掌握Testbench中组合逻辑测试文件的写法。
3.下载并测试实现的逻辑功能。

二、预习要求

1.复习组合逻辑电路的Verilog HDL建模。
2.熟悉7段译码器的功能表。
3.根据实验内容，思考测试电路的要求，画出测试激励的时序图。

三、实验内容

设计并实现共阴极7段译码器的逻辑功能。
1.设计分析。
段译码器结构如图 7-27 所示。
(1）确定输人/输出的端口个数：共阴极7段译码器输人4位；输出（高电平有效）7位。端口结构图如图7-28 所示。
(2）真值表如表7-10所示。

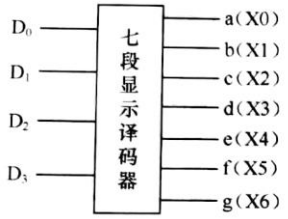
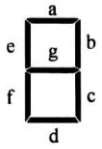


图 7-27 7 段译码器结构



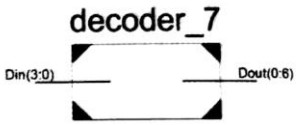


图 7-28 7 段译码器端口结构图

表 7-10 共阴极 7 段译码器真值表

D_3	D_2	D_1	D_0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	1	1	1	0	1	1	1
1	0	1	1	0	0	1	1	1	1	1
1	1	0	0	0	0	0	1	1	0	1
1	1	0	1	0	1	1	1	1	0	1
1	1	1	0	1	0	0	1	1	1	1
1	1	1	1	1	0	0	0	1	1	1

2.设计实现。
（1）新建工程。

双击桌面上的ISE14.7图标，启动ISE软件(也可通过菜单启动)。每次开ISE都会默认恢复最近使用过的工程界面。第一次使用时，由于还没有历史记录，所以工程管理区显示空白。单击“File”/“New Proiect”，启动新建工程向导，在弹出的对话框中输人工程名称decode_7,并指定工程路径。

单击“Next”按钮进人下一对话框，选择实验箱上所用的集成电路型号及综合仿真工具。这里选用的集成电路型号为xC3S50TQ144。选择Verilog HDL作为默认的硬件描达语言。

单击“Next”按钮进人设置确认对话框，确认无误后，单击“Finish’”按钮，完成新建工程操作。

(2）设计输人和设计仿真。

在工程管理区任意位置单击鼠标右键，在弹出的菜单中选择"New Source“命令，单击"Verilog Module"，并输人文件名 decode_ 7。

单击“Next”按钮进人模块定义对话框，输入（Din）为4 位数据，输出(Dout）为7位数据也可以略过这一步，在源程序中进行添加。

单击“Next”按钮进入新建文件汇总对话框，确认无误后，单击“Fnish”按钮，完成新建文件操作。

通过以上操作，ISE会自动生成一个Verilog模块的代码框架，并且在源代码编辑区打开。简单的注释、实体和端口定义已经自动生成，接下来的工作就是在此基础上设计待实现的编辑功能。参考代码如下。

```
module decode_7(Din,Dout);
    input [3:0] Din;              //定义输入端口
    output reg [6:0] Dout;        //定义输出端口，设置为寄存器型变量
    always@( Din )                //电路模块功能描述
    begin                         //块语句开始
        case(Din)
                4'b0000: Dout <= 7'b1111110 ;          //0
                4'b0001: Dout <= 7'b0110000 ;          //1
                4'b0010: Dout <= 7'b1101101 ;          //2
                4'b0011: Dout <= 7'b1111001 ;          //3
                4'b0100: Dout <= 7'b0110011 ;          //4
                4'b0101: Dout <= 7'b1011011 ;          //5
                4'b0110: Dout <= 7'b1011111 ;          //6
                4'b0111: Dout <= 7'b1110000 ;          //7
                4'b1000: Dout <= 7'b1111111 ;          //8
                4'b1001: Dout <= 7'b1111011 ;          //9
                4'b1010: Dout <= 7'b1110111 ;          //A
                4'b1011: Dout <= 7'b0011111 ;          //B
                4'b1100: Dout <= 7'b0001101 ;          //C
                4'b1101: Dout <= 7'b0111101 ;          //D
                4'b1110: Dout <= 7'b1001111 ;          //E
                4'b1111: Dout <= 7'b1000111 ;          //F
              default: Dout <= 7'b0000000;
        endcase
    end
endmodule
```

在工程管理区的View栏选中“Simulation”，在该区任意位置单击鼠标右键，在弹出的菜单中选择“New Source”，在类型中选择"Verilog Test Fixture”，输入测试文件名,单击“Next”按钮。这时,工程中所有的模块名都会显示出来,选择要进行测试的模块decoder_ 7。关联后

生成的测试文件中会自动加入对源文件的实例化代码。单击“Next”按钮,在弹出的对话框中确认信息无误后单击“Finish”按钮。

ISim自动生成了基本的信号并对被测模块做了实例化,并生成了测试代码框架。因为本例为组合逻辑,没有时钟脉冲,所以要把!生成的测试代码框架中和时钟脉冲相关的进程删去,并加入描述激励信号的代码。

```
`timescale 1ns / 1ps
module decode_7_test;
    //声明输入变量
    reg [3:0] Din;
    //声明输出变量
    wire [6:0] Dout;
    //调用设计块
    decode_7 uut (
        .Din(Din),
        .Dout(Dout)
    );
    initial begin
        //初始化输入变量
        Din = 0;
        #50 Din = 4'b0000;
        #50 Din = 4'b0001;
        #50 Din = 4'b0010;
        #50 Din = 4'b0011;
        #50 Din = 4'b0100;
        #50 Din = 4'b0101;
        #50 Din = 4'b0110;
        #50 Din = 4'b0111;
        #50 Din = 4'b1000;
        #50 Din = 4'b1001;
        #50 Din = 4'b1010;
        #50 Din = 4'b1011;
        #50 Din = 4'b1100;
        #50 Din = 4'b1101;
        #50 Din = 4'b1110;
        #50 Din = 4'b1111;
    end
endmodule
```

测试程序也可设计出4位模16计数器，产生Din管脚的输入激励信号，测试代码如下。

```
'timescale 1ns / 1ps
module decode_7_test;
    //声明输入变量
    reg [3:0] Din;
    reg clk;
    //声明输出变量
    wire [6:0] Dout;
    //调用设计块
    decode_7 uut (
        .Din(Din),
        .Dout(Dout)
    );
    initial begin
        //初始化输入变量
        Din = 0;
        clk=0;
        #100;
    end
always @(posedge clk)              //检测clk上升沿
        begin
           if (Din == 4'b1001)     //计数到“1001”即9，则清零
                Din = 4'b0000;
             else
                Din = Din + 4'b0001;
        end
always #20 clk = ~clk;             //产生时钟脉冲
endmodule
```

在工程管理区的View栏中选中“Simulation”，在下方的下拉列表中选中行为仿真（Behavioral）。
在过程管理区中单击展开“ISim Simulator′”，双击“Behavioral Check Syntax”对Testbench进行语法检查，如有语法错误，则修改错误并继续进行语法检查。通过Behavioral Check Syntax后，双击第二个选项“Simulate Behavioral Model”，启动行为仿真。双击“Simulate BehavioralModel”后，仿真器ISim自动打开。
仿真波形如图7-29所示。

图 7-29 7段译码器仿真波形

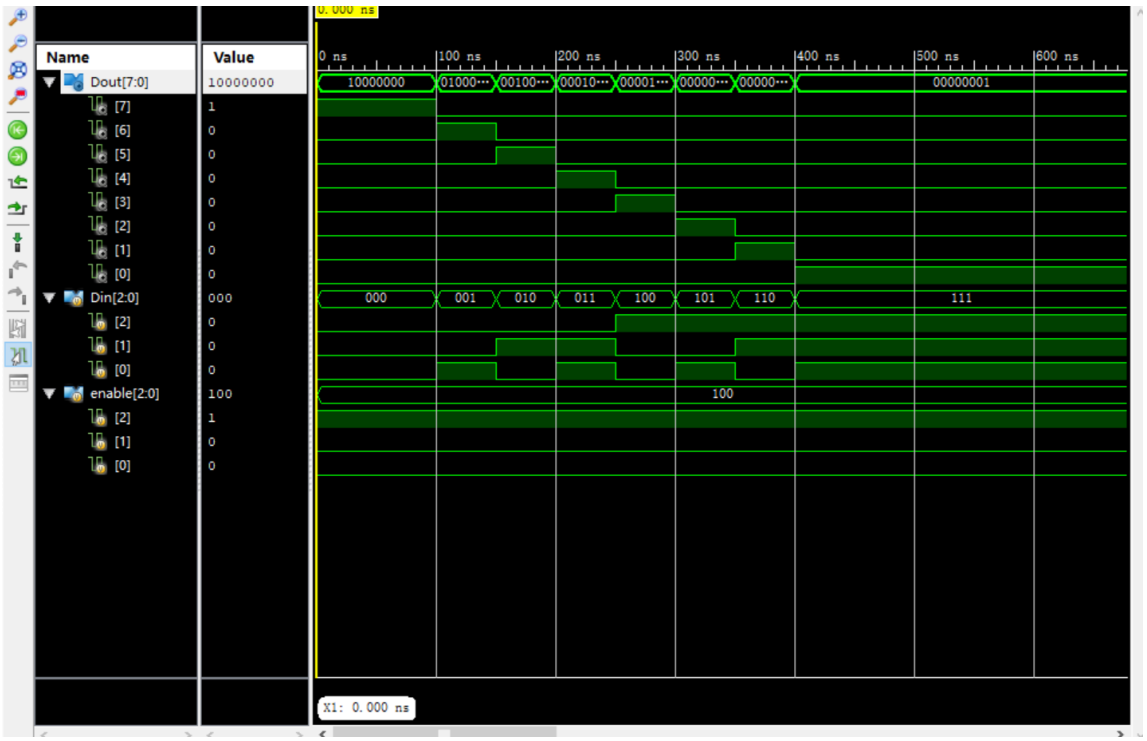
四、实验要求

用Verilog HDL设计一个3-8线译码器，完成设计模块、Testbench，并保存仿真波形。要求 Testbench 能够覆盖所有的输人组合、下载到FPGA、完成硬件调测并实现电路功能。

（1）设计代码及测试代码

```
module fixed;

   // Inputs
   reg [2:0] Din;
   reg [2:0] enable;

   // Outputs
   wire [7:0] Dout;

   // Instantiate the Unit Under Test (UUT)
   yimaqi uut (
      .Din(Din),
      .Dout(Dout),
      .enable(enable)
   );

   initial begin
      // Initialize Inputs
      Din = 0;
      enable[2] = 1;
      enable[1] = 0;
      enable[0] = 0;
      #50 Din =3'b000;
      #50 Din =3'b001;
      #50 Din =3'b010;
      #50 Din =3'b011;
      #50 Din =3'b100;
      #50 Din =3'b101;
      #50 Din =3'b110;
      #50 Din =3'b111;
      // Wait 100 ns for global reset to finish
      #1000;

      // Add stimulus here

   end

endmodule
module yimaqi(Din,Dout,enable );
 input [2:0] Din;
 input [2:0] enable;
 output reg [7:0] Dout;
 always @(enable or Din)
    if(enable[2]&(~enable[1])&(~enable[0]))
      case(Din)
      3'b000:Dout <= 8'b10000000;
      3'b001:Dout <= 8'b01000000;
      3'b010:Dout <= 8'b00100000;
      3'b011:Dout <= 8'b00010000;
      3'b100:Dout <= 8'b00001000;
      3'b101:Dout <= 8'b00000100;
      3'b110:Dout <= 8'b00000010;
      3'b111:Dout <= 8'b00000001;
      endcase
      else Dout<=8'b00000000;
 endmodule
```

（2）仿真截图



五、实验总结

通过本次实验我学会了使用ISE软件完成组合逻辑设计的输人并仿真。掌握了Testbench中组合逻辑测试文件的写法。能下载并测试实现的逻辑功能。