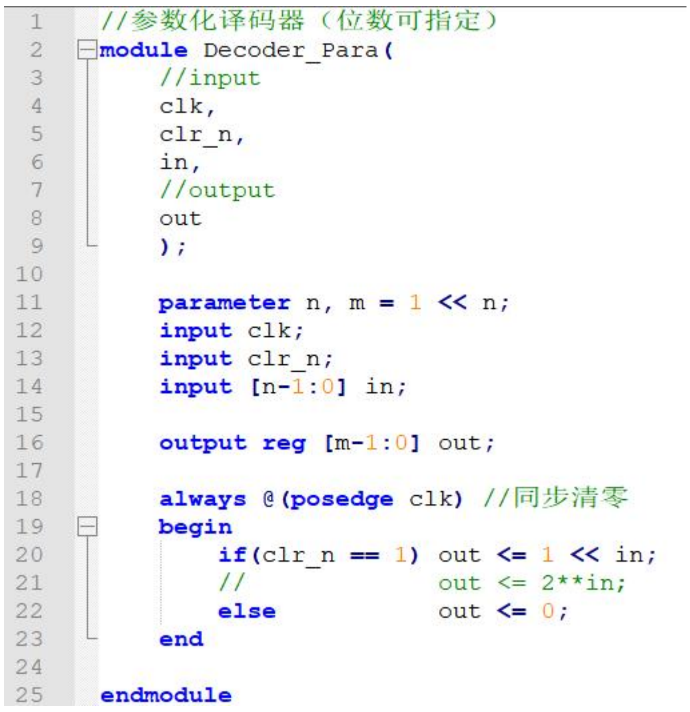
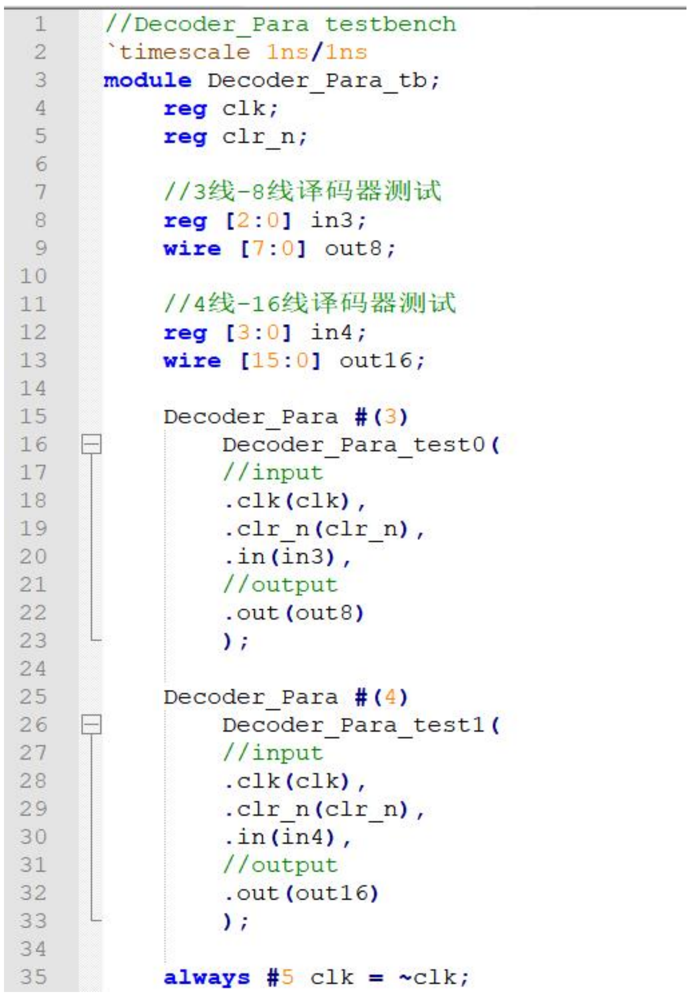
一．参数化译码器

1. 模块功能代码：

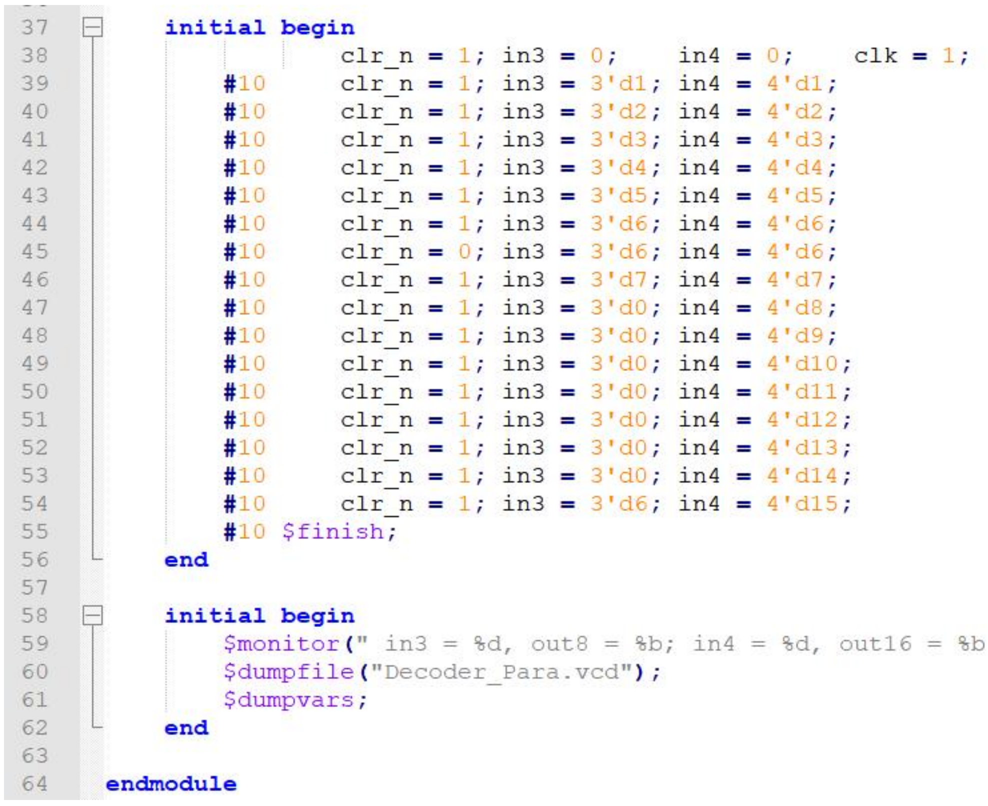


```
//参数化译码器（位数可指定）
module Decoder_Para(
    //input
    clk,
    clr_n,
    in,
    //output
    out
    );

    parameter n, m = 1 << n;
    input clk;
    input clr_n;
    input [n-1:0] in;

    output reg [m-1:0] out;

    always @(posedge clk) //同步清零
    begin
        if(clr_n == 1) out <= 1 << in;
        //             out <= 2**in;
        else           out <= 0;
    end

endmodule
```

2. 模块测试代码：由于译码器的参数可指定，则测试模块选择 3-8 线和 4-16 线两种进行测试，通过参数重载调用模块。



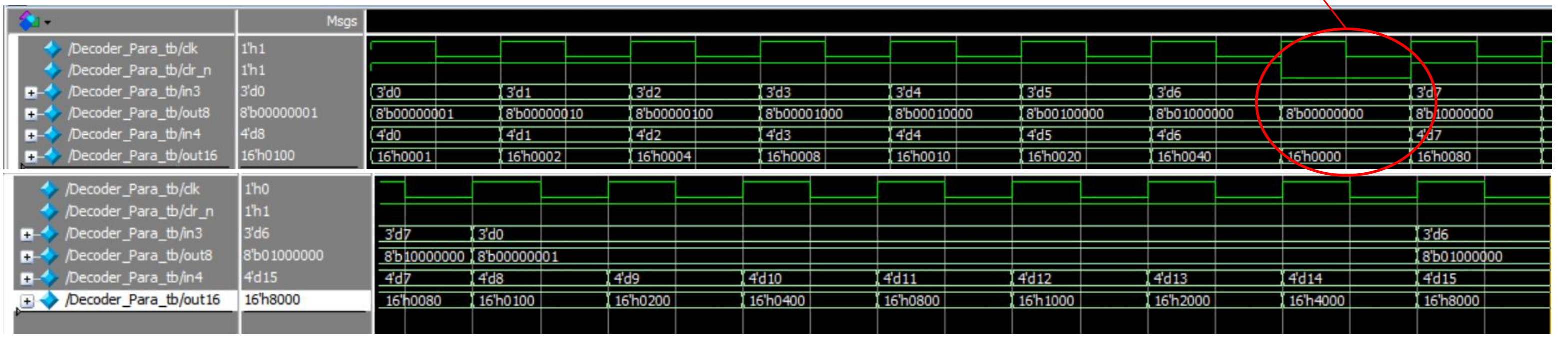
```
//Decoder_Para testbench
`timescale 1ns/1ns
module Decoder_Para_tb;
    reg clk;
    reg clr_n;

    //3线-8线译码器测试
    reg [2:0] in3;
    wire [7:0] out8;

    //4线-16线译码器测试
    reg [3:0] in4;
    wire [15:0] out16;

    Decoder_Para #(3)
        Decoder_Para_test0(
        //input
        .clk(clk),
        .clr_n(clr_n),
        .in(in3),
        //output
        .out(out8)
        );

    Decoder_Para #(4)
        Decoder_Para_test1(
        //input
        .clk(clk),
        .clr_n(clr_n),
        .in(in4),
        //output
        .out(out16)
        );

    always #5 clk = ~clk;
```



```verilog
    initial begin
                clr_n = 1; in3 = 0;    in4 = 0;    clk = 1;
        #10     clr_n = 1; in3 = 3'd1; in4 = 4'd1;
        #10     clr_n = 1; in3 = 3'd2; in4 = 4'd2;
        #10     clr_n = 1; in3 = 3'd3; in4 = 4'd3;
        #10     clr_n = 1; in3 = 3'd4; in4 = 4'd4;
        #10     clr_n = 1; in3 = 3'd5; in4 = 4'd5;
        #10     clr_n = 1; in3 = 3'd6; in4 = 4'd6;
        #10     clr_n = 0; in3 = 3'd6; in4 = 4'd6;
        #10     clr_n = 1; in3 = 3'd7; in4 = 4'd7;
        #10     clr_n = 1; in3 = 3'd0; in4 = 4'd8;
        #10     clr_n = 1; in3 = 3'd0; in4 = 4'd9;
        #10     clr_n = 1; in3 = 3'd0; in4 = 4'd10;
        #10     clr_n = 1; in3 = 3'd0; in4 = 4'd11;
        #10     clr_n = 1; in3 = 3'd0; in4 = 4'd12;
        #10     clr_n = 1; in3 = 3'd0; in4 = 4'd13;
        #10     clr_n = 1; in3 = 3'd0; in4 = 4'd14;
        #10     clr_n = 1; in3 = 3'd6; in4 = 4'd15;
        #10 $finish;
    end

    initial begin
        $monitor(" in3 = %d, out8 = %b; in4 = %d, out16 = %b ", in3, out8, in4, out16);
        $dumpfile("Decoder_Para.vcd");
        $dumpvars;
    end

endmodule
```

当 clr_n 为低电平时，在时钟上升沿两个译码器都被清零。

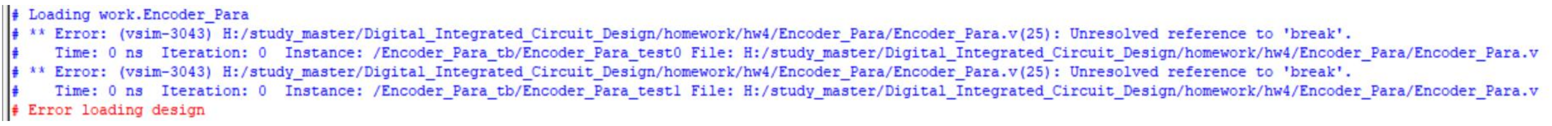
3. ModelSim 仿真结果及分析：



由测试用例可以看出，分别采用了列举全部情况的方式输入测试激励，由波形可知结果正确。

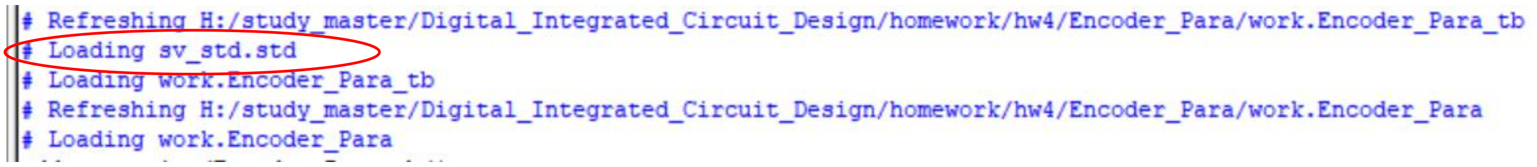
二．参数化（优先）编码器

1. 模块功能代码：值得注意的是，优先编码器采用了 break 语句退出 for 循环，则只能将文件保存为 systemverilog 格式（.sv），否则用 ModelSim 仿真时的加载会产生以下错误。

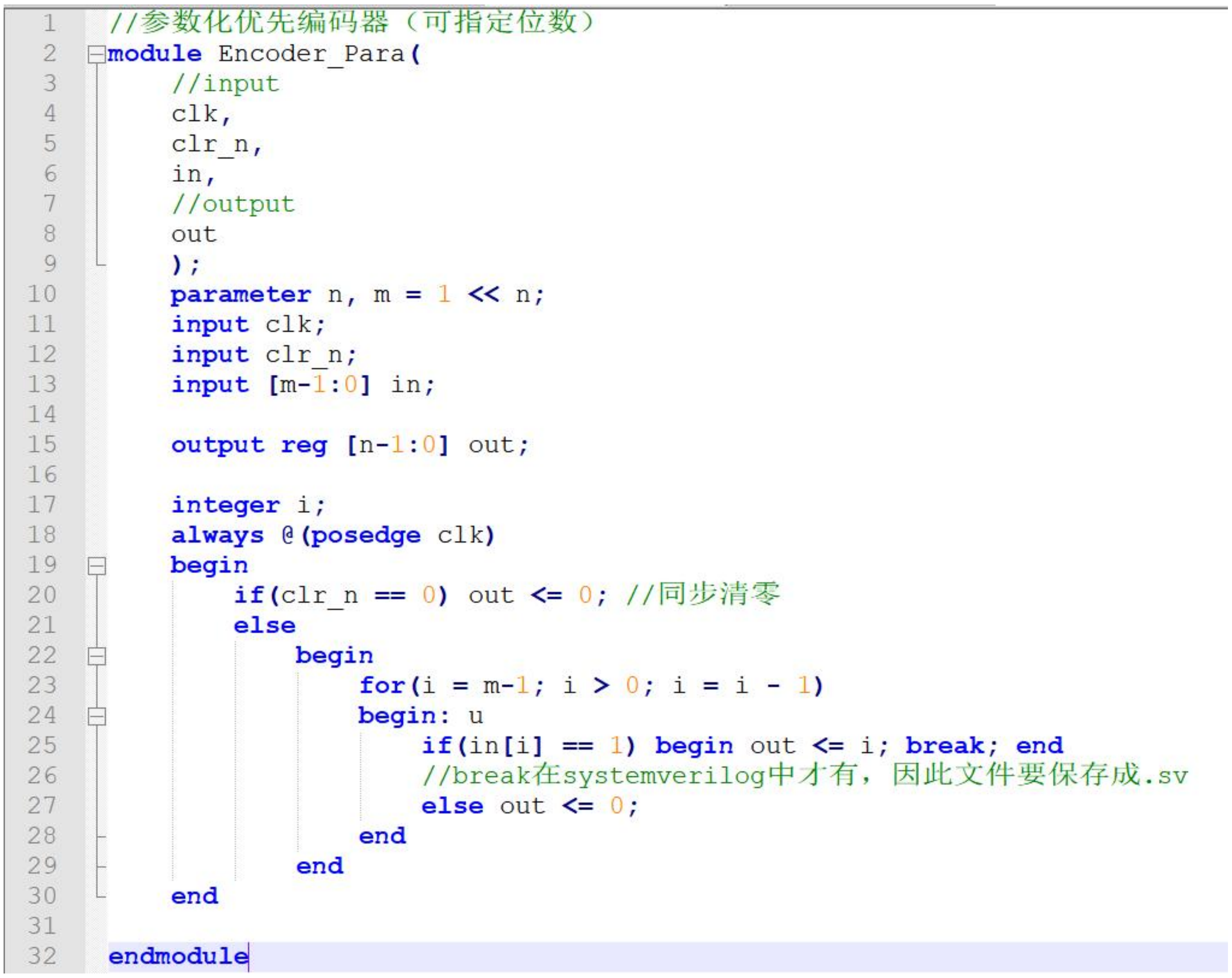


```
# Loading work.Encoder_Para
# ** Error: (vsim-3043) H:/study_master/Digital_Integrated_Circuit_Design/homework/hw4/Encoder_Para/Encoder_Para.v(25): Unresolved reference to 'break'.
#    Time: 0 ns  Iteration: 0  Instance: /Encoder_Para_tb/Encoder_Para_test0 File: H:/study_master/Digital_Integrated_Circuit_Design/homework/hw4/Encoder_Para/Encoder_Para.v
# ** Error: (vsim-3043) H:/study_master/Digital_Integrated_Circuit_Design/homework/hw4/Encoder_Para/Encoder_Para.v(25): Unresolved reference to 'break'.
#    Time: 0 ns  Iteration: 0  Instance: /Encoder_Para_tb/Encoder_Para_test1 File: H:/study_master/Digital_Integrated_Circuit_Design/homework/hw4/Encoder_Para/Encoder_Para.v
# Error loading design
```

修改文件格式后，加载成功，且出现了一行关于 systemverilog 的说明。

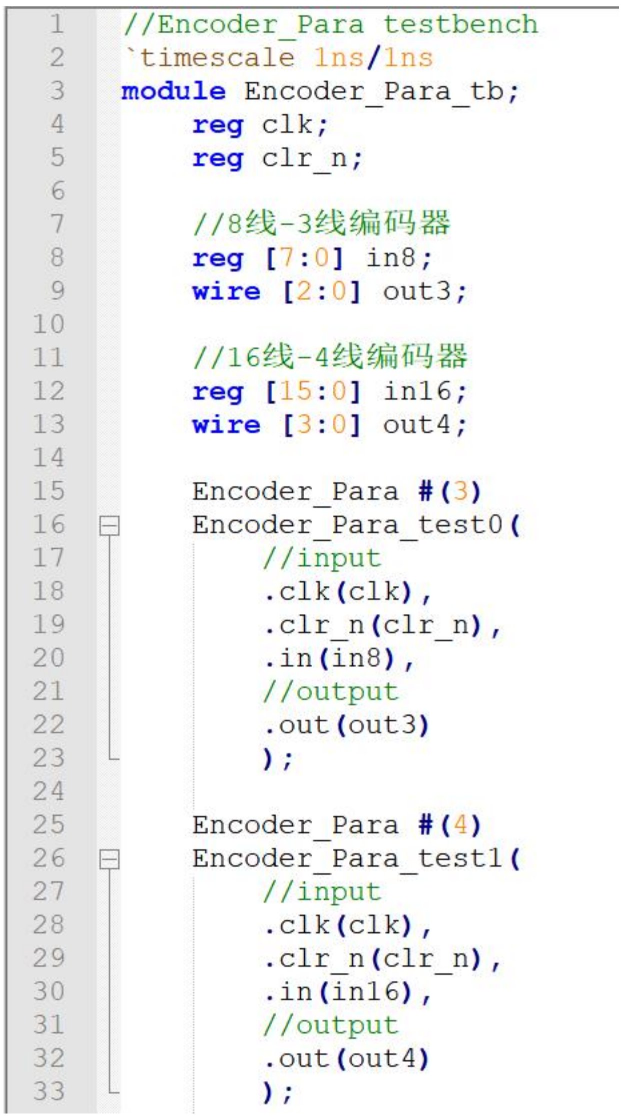


```
# Refreshing H:/study_master/Digital_Integrated_Circuit_Design/homework/hw4/Encoder_Para/work.Encoder_Para_tb
# Loading sv_std.std
# Loading work.Encoder_Para_tb
# Refreshing H:/study_master/Digital_Integrated_Circuit_Design/homework/hw4/Encoder_Para/work.Encoder_Para
# Loading work.Encoder_Para
```

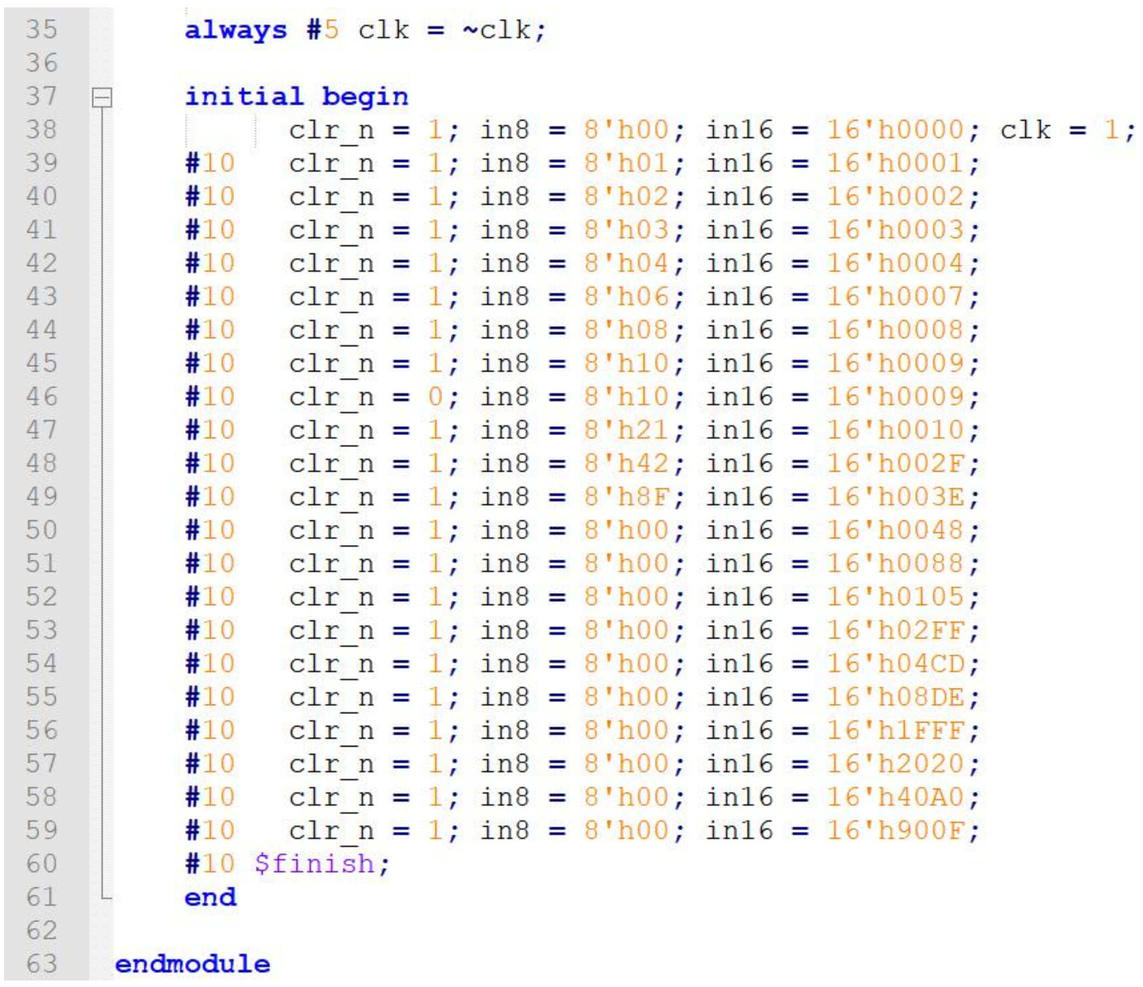


```
//参数化优先编码器（可指定位数）
module Encoder_Para(
    //input
    clk,
    clr_n,
    in,
    //output
    out
    );
    parameter n, m = 1 << n;
    input clk;
    input clr_n;
    input [m-1:0] in;

    output reg [n-1:0] out;

    integer i;
    always @(posedge clk)
    begin
        if(clr_n == 0) out <= 0; //同步清零
        else
            begin
                for(i = m-1; i > 0; i = i - 1)
                begin: u
                    if(in[i] == 1) begin out <= i; break; end
                    //break在systemverilog中才有，因此文件要保存成.sv
                    else out <= 0;
                end
            end
    end

endmodule
```

2. 模块测试代码：由于编码器的参数可指定，则测试模块选择 8-3 线和 16-4 线两种进行测试，通过参数重载调用模块。

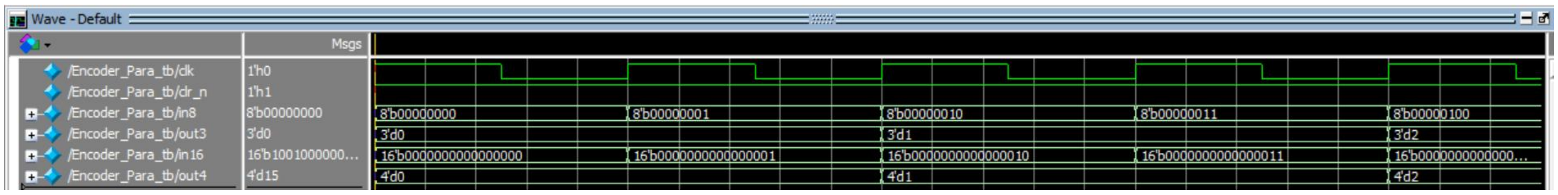


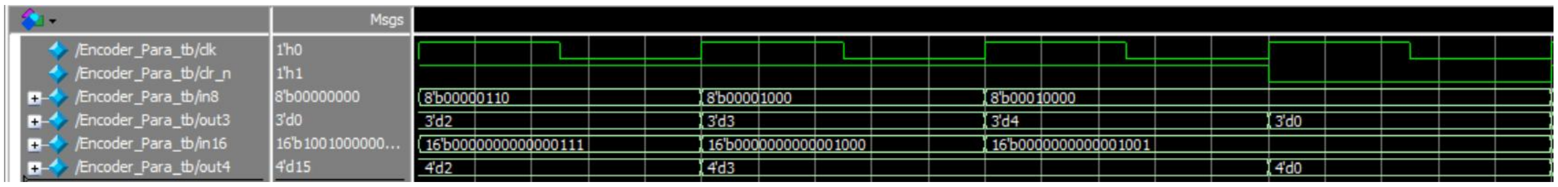
```verilog
//Encoder_Para testbench
`timescale 1ns/1ns
module Encoder_Para_tb;
    reg clk;
    reg clr_n;

    //8线-3线编码器
    reg [7:0] in8;
    wire [2:0] out3;

    //16线-4线编码器
    reg [15:0] in16;
    wire [3:0] out4;

    Encoder_Para #(3)
    Encoder_Para_test0(
        //input
        .clk(clk),
        .clr_n(clr_n),
        .in(in8),
        //output
        .out(out3)
        );

    Encoder_Para #(4)
    Encoder_Para_test1(
        //input
        .clk(clk),
        .clr_n(clr_n),
        .in(in16),
        //output
        .out(out4)
        );
```

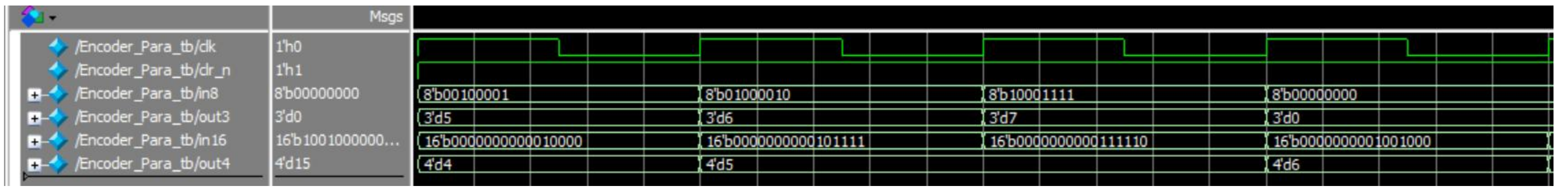


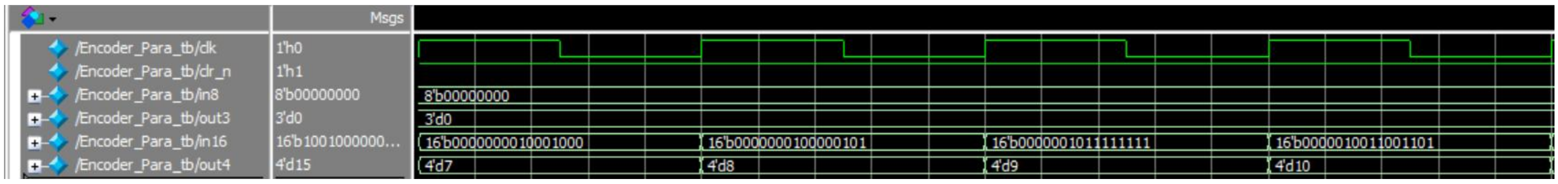
```verilog
    always #5 clk = ~clk;

    initial begin
          clr_n = 1; in8 = 8'h00; in16 = 16'h0000; clk = 1;
    #10   clr_n = 1; in8 = 8'h01; in16 = 16'h0001;
    #10   clr_n = 1; in8 = 8'h02; in16 = 16'h0002;
    #10   clr_n = 1; in8 = 8'h03; in16 = 16'h0003;
    #10   clr_n = 1; in8 = 8'h04; in16 = 16'h0004;
    #10   clr_n = 1; in8 = 8'h06; in16 = 16'h0007;
    #10   clr_n = 1; in8 = 8'h08; in16 = 16'h0008;
    #10   clr_n = 1; in8 = 8'h10; in16 = 16'h0009;
    #10   clr_n = 0; in8 = 8'h10; in16 = 16'h0009;
    #10   clr_n = 1; in8 = 8'h21; in16 = 16'h0010;
    #10   clr_n = 1; in8 = 8'h42; in16 = 16'h002F;
    #10   clr_n = 1; in8 = 8'h8F; in16 = 16'h003E;
    #10   clr_n = 1; in8 = 8'h00; in16 = 16'h0048;
    #10   clr_n = 1; in8 = 8'h00; in16 = 16'h0088;
    #10   clr_n = 1; in8 = 8'h00; in16 = 16'h0105;
    #10   clr_n = 1; in8 = 8'h00; in16 = 16'h02FF;
    #10   clr_n = 1; in8 = 8'h00; in16 = 16'h04CD;
    #10   clr_n = 1; in8 = 8'h00; in16 = 16'h08DE;
    #10   clr_n = 1; in8 = 8'h00; in16 = 16'h1FFF;
    #10   clr_n = 1; in8 = 8'h00; in16 = 16'h2020;
    #10   clr_n = 1; in8 = 8'h00; in16 = 16'h40A0;
    #10   clr_n = 1; in8 = 8'h00; in16 = 16'h900F;
    #10 $finish;
    end

endmodule
```

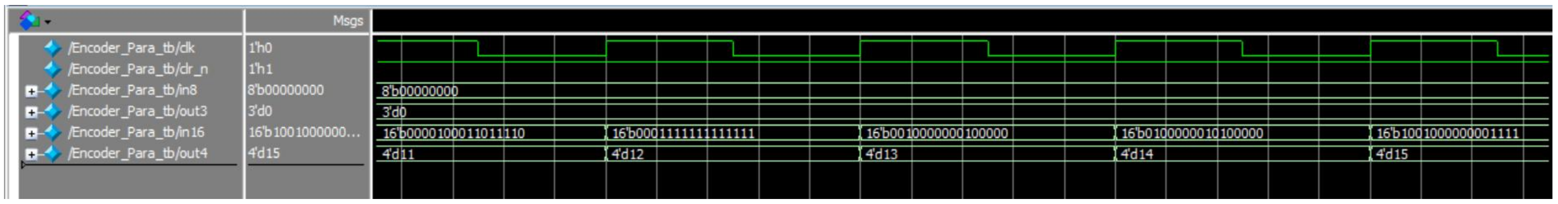
3. ModelSim 仿真结果及分析：




Msgs
/Encoder_Para_tb/clk 1'h0
/Encoder_Para_tb/clr_n 1'h1
/Encoder_Para_tb/in8 8'b00000000
/Encoder_Para_tb/out3 3'd0
/Encoder_Para_tb/in16 16'b1001000000...
/Encoder_Para_tb/out4 4'd15
8'b00000110
8'b00001000
8'b00010000
3'd2
3'd3
3'd4
3'd0
16'b0000000000000111
16'b0000000000001000
16'b0000000000001001
4'd2
4'd3
4'd0


Msgs
/Encoder_Para_tb/clk 1'h0
/Encoder_Para_tb/clr_n 1'h1
/Encoder_Para_tb/in8 8'b00000000
/Encoder_Para_tb/out3 3'd0
/Encoder_Para_tb/in16 16'b1001000000...
/Encoder_Para_tb/out4 4'd15
8'b00100001
8'b01000010
8'b10001111
8'b00000000
3'd5
3'd6
3'd7
3'd0
16'b0000000000010000
16'b0000000000101111
16'b0000000000111110
16'b0000000001001000
4'd4
4'd5
4'd6

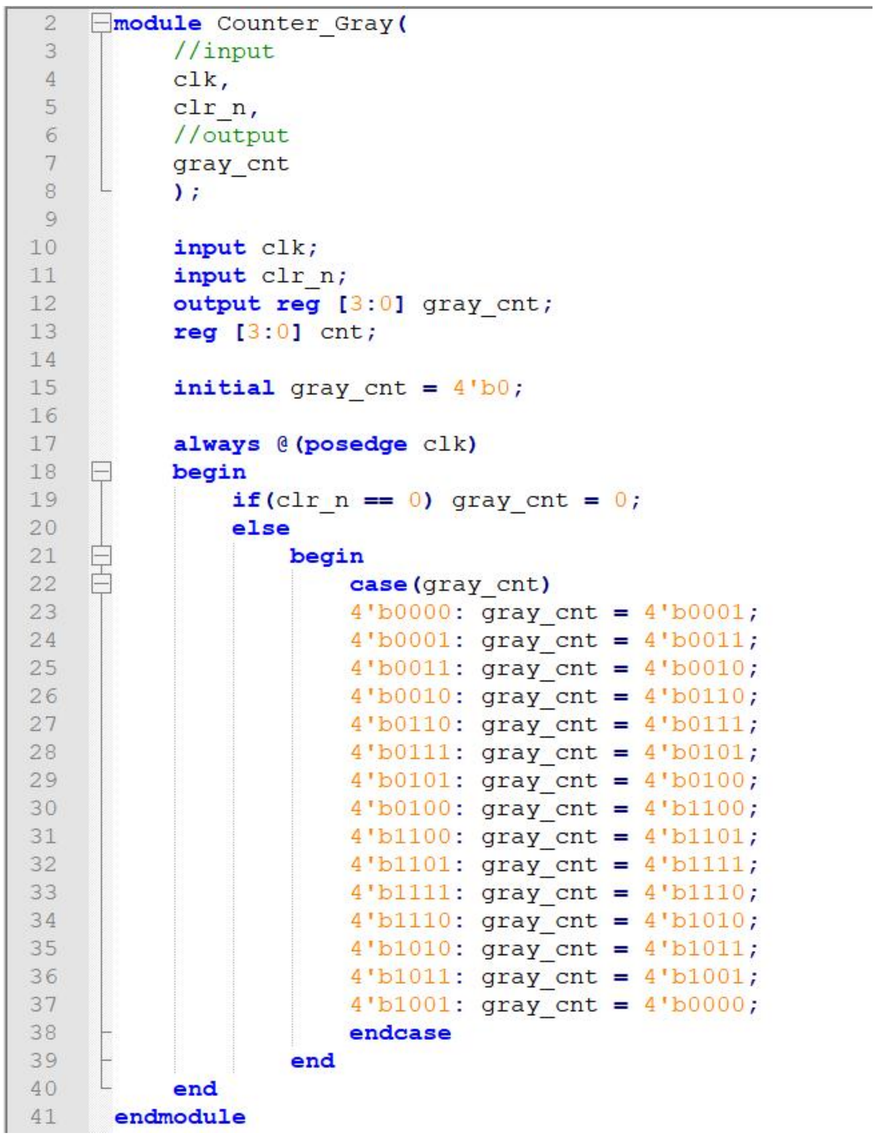

Msgs
/Encoder_Para_tb/clk 1'h0
/Encoder_Para_tb/clr_n 1'h1
/Encoder_Para_tb/in8 8'b00000000
/Encoder_Para_tb/out3 3'd0
/Encoder_Para_tb/in16 16'b1001000000...
/Encoder_Para_tb/out4 4'd15
8'b00000000
3'd0
16'b0000000010001000
16'b0000000100000101
16'b0000001011111111
16'b0000010011001101
4'd7
4'd8
4'd9
4'd10


Msgs
/Encoder_Para_tb/clk 1'h0
/Encoder_Para_tb/clr_n 1'h1
/Encoder_Para_tb/in8 8'b00000000
/Encoder_Para_tb/out3 3'd0
/Encoder_Para_tb/in16 16'b1001000000...
/Encoder_Para_tb/out4 4'd15
8'b00000000
3'd0
16'b0000100011011110
16'b0001111111111111
16'b0010000000100000
16'b0100000010100000
16'b1001000000001111
4'd11
4'd12
4'd13
4'd14
4'd15

测试用例由列举法列出了产生所有输出的结果，由波形图可以看出，功能设计正确。

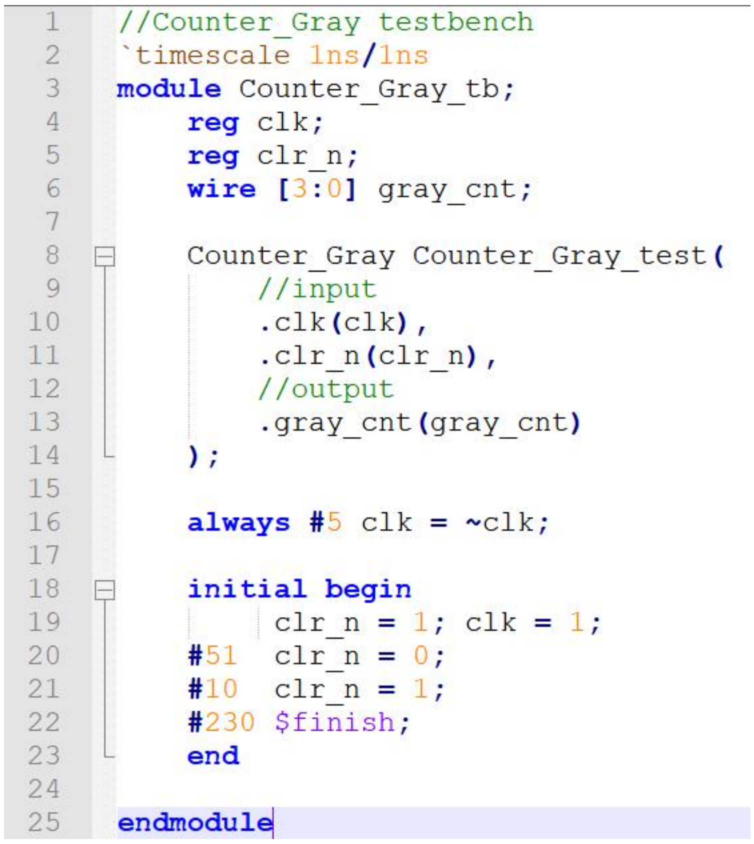
三．格雷码计数器

1. 模块功能代码：利用状态转换的方法实现。



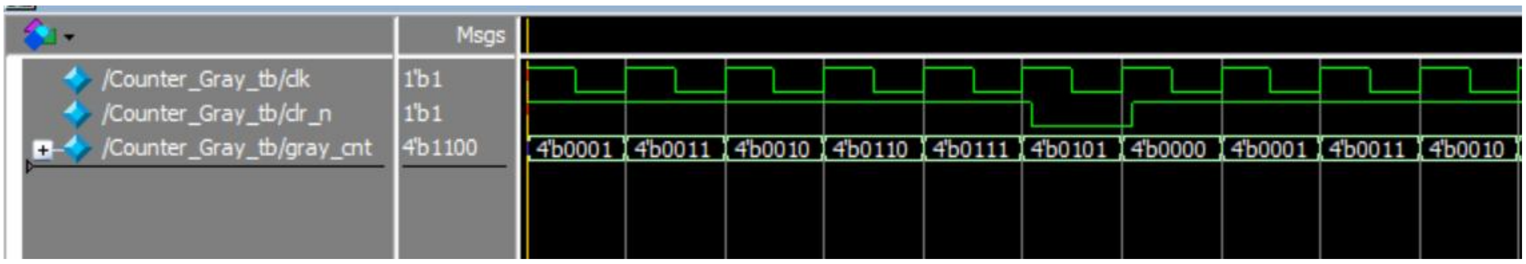
```verilog
module Counter_Gray(
    //input
    clk,
    clr_n,
    //output
    gray_cnt
    );

    input clk;
    input clr_n;
    output reg [3:0] gray_cnt;
    reg [3:0] cnt;

    initial gray_cnt = 4'b0;

    always @(posedge clk)
    begin
        if(clr_n == 0) gray_cnt = 0;
        else
            begin
                case(gray_cnt)
                4'b0000: gray_cnt = 4'b0001;
                4'b0001: gray_cnt = 4'b0011;
                4'b0011: gray_cnt = 4'b0010;
                4'b0010: gray_cnt = 4'b0110;
                4'b0110: gray_cnt = 4'b0111;
                4'b0111: gray_cnt = 4'b0101;
                4'b0101: gray_cnt = 4'b0100;
                4'b0100: gray_cnt = 4'b1100;
                4'b1100: gray_cnt = 4'b1101;
                4'b1101: gray_cnt = 4'b1111;
                4'b1111: gray_cnt = 4'b1110;
                4'b1110: gray_cnt = 4'b1010;
                4'b1010: gray_cnt = 4'b1011;
                4'b1011: gray_cnt = 4'b1001;
                4'b1001: gray_cnt = 4'b0000;
                endcase
            end
    end
endmodule
```

2. 模块测试代码：

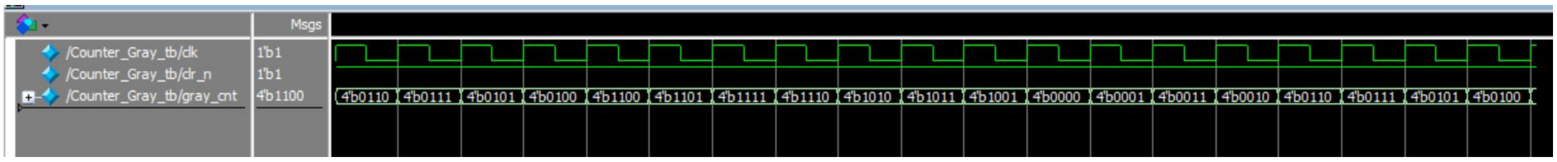


```verilog
//Counter_Gray testbench
`timescale 1ns/1ns
module Counter_Gray_tb;
    reg clk;
    reg clr_n;
    wire [3:0] gray_cnt;

    Counter_Gray Counter_Gray_test(
        //input
        .clk(clk),
        .clr_n(clr_n),
        //output
        .gray_cnt(gray_cnt)
    );

    always #5 clk = ~clk;

    initial begin
            clr_n = 1; clk = 1;
    #51     clr_n = 0;
    #10     clr_n = 1;
    #230 $finish;
    end

endmodule
```

3. ModelSim 仿真结果及分析：



```
/Counter_Gray_tb/clk       1'b1
/Counter_Gray_tb/clr_n     1'b1
/Counter_Gray_tb/gray_cnt  4'b1100   4'b0001 | 4'b0011 | 4'b0010 | 4'b0110 | 4'b0111 | 4'b0101 | 4'b0000 | 4'b0001 | 4'b0011 | 4'b0010
```



```
/Counter_Gray_tb/clk       1'b1
/Counter_Gray_tb/clr_n     1'b1
/Counter_Gray_tb/gray_cnt  4'b1100   4'b0110 | 4'b0111 | 4'b0101 | 4'b0100 | 4'b1100 | 4'b1101 | 4'b1111 | 4'b1110 | 4'b1010 | 4'b1011 | 4'b1001 | 4'b0000 | 4'b0001 | 4'b0011 | 4'b0010 | 4'b0110 | 4'b0111 | 4'b0101 | 4'b0100
```

由仿真结果可以看出，四位格雷码计数器设计正确，当 clr_n 为低电平时，在时钟上升沿清零。