数字系统设计作业

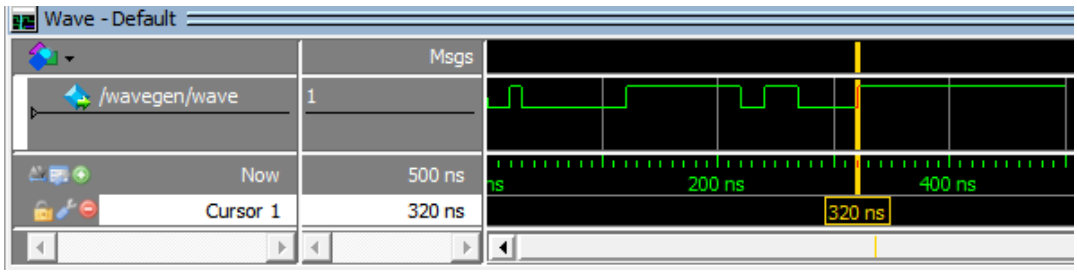
学号：515030910059　姓名：　王靖康　日期：　12 月 25 日

第 1 题：

（1）设计模块（测试模块）

```
1   `timescale 10ns/1ns
2
3   module wavegen(output reg wave);
4
5     initial begin
6       wave = 0;
7       #2 wave = 1;
8       #1 wave = 0;
9       #9 wave = 1;
10      #10 wave = 0;
11      #2 wave = 1;
12      #3 wave = 0;
13      #5 wave = 1;
14    end
15
16    initial begin
17      $monitor("Current Time: %tns,", $time, "<---->wave=%b", wave);
18    end
19
20  endmodule
21
```

（2）测试波形图：



（3）显示输出：

```
VSIM 9> run
# Current Time:                    0ns,<---->wave=0
# Current Time:                   20ns,<---->wave=1
# Current Time:                   30ns,<---->wave=0
# Current Time:                  120ns,<---->wave=1
# Current Time:                  220ns,<---->wave=0
# Current Time:                  240ns,<---->wave=1
# Current Time:                  270ns,<---->wave=0
# Current Time:                  320ns,<---->wave=1
```

（4）设计说明：

由于该题目为模型实现波形生成功能，可以在模块内部测试模块正确性，且题目中只提到一个 wavegen 模块，故将设计模块和测试模块合二为一。

第 2 题：

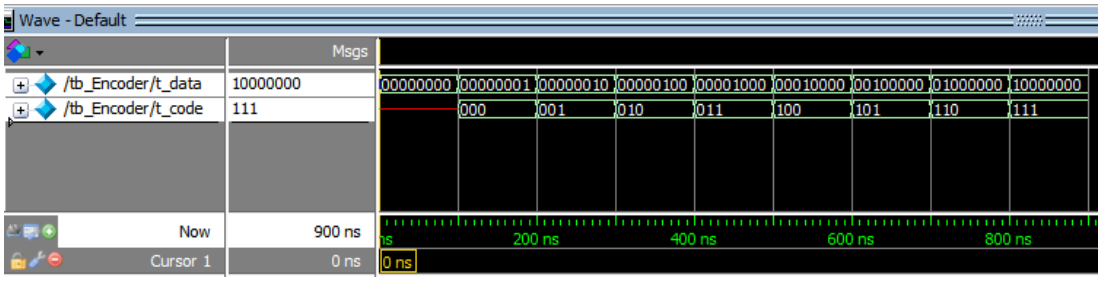
（1）设计模块

```
module Encoder8x3(output reg [2:0] code,
                  input [7:0] data);

always @(*) begin
  case (data)
    8'b0000_0001: code = 0;
    8'b0000_0010: code = 1;
    8'b0000_0100: code = 2;
    8'b0000_1000: code = 3;
    8'b0001_0000: code = 4;
    8'b0010_0000: code = 5;
    8'b0100_0000: code = 6;
    8'b1000_0000: code = 7;
    default: code = 3'bx;
  endcase
end

endmodule
```

（2）测试模块

```
`include "Encoder8x3.v"
`timescale 10ns/1ns

module tb_Encoder;
  reg [7:0] t_data;
  wire [2:0] t_code;

  Encoder8x3 test(.data(t_data), .code(t_code));

  initial begin
    t_data = 0;
    #10 t_data = 8'b0000_0001;
    #10 t_data = 8'b0000_0010;
    #10 t_data = 8'b0000_0100;
    #10 t_data = 8'b0000_1000;
    #10 t_data = 8'b0001_0000;
    #10 t_data = 8'b0010_0000;
    #10 t_data = 8'b0100_0000;
    #10 t_data = 8'b1000_0000;
    #10 $stop;
  end

  initial begin
    $monitor("Current time %tns, ", $time, "<----> t_data:%b, t_code:%b", t_data, t_code);
  end
endmodule
```

（3）测试波形图



（4）显示输出

```
VSIM 14> run
# Current time                  100ns, <----> t_data:00000001, t_code:000
# Current time                  200ns, <----> t_data:00000010, t_code:001
# Current time                  300ns, <----> t_data:00000100, t_code:010
# Current time                  400ns, <----> t_data:00001000, t_code:011
# Current time                  500ns, <----> t_data:00010000, t_code:100
# Current time                  600ns, <----> t_data:00100000, t_code:101
# Current time                  700ns, <----> t_data:01000000, t_code:110
# Current time                  800ns, <----> t_data:10000000, t_code:111
# Break in Module tb_Encoder at C:/Software/modeltech64_10.1c/workspace/Assignments/tb_Encoder8x3.v
```

第 3 题：

（1）设计模块

```
module mux2x1(output dout,
              input sel,
              input [1:0] din);

bufif1 b2(dout, din[1], sel);
bufif0 b1(dout, din[0], sel);

endmodule
```

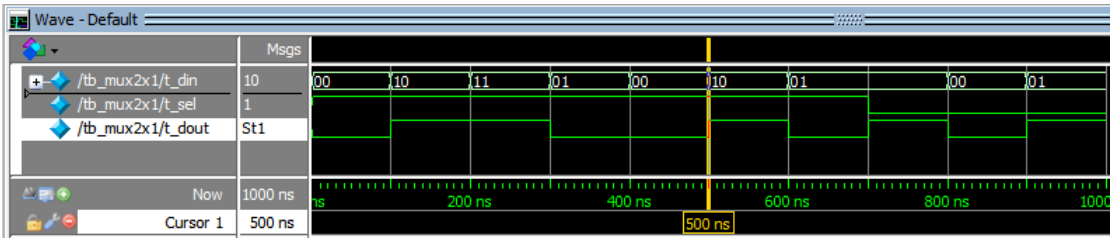
```
`include "mux2x1.v"

module mux4x1(output dout,
              input [1:0] sel,
              input [3:0] din);

  wire dout1;
  wire dout2;

  mux2x1 m1_mux2x1(.dout(dout1),
                   .sel(sel[0]),
                   .din(din[3:2]));

  mux2x1 m2_mux2x1(.dout(dout2),
                   .sel(sel[0]),
                   .din(din[1:0]));

  assign dout = (sel[1] & dout1) | (~sel[1] & dout2);


endmodule
```

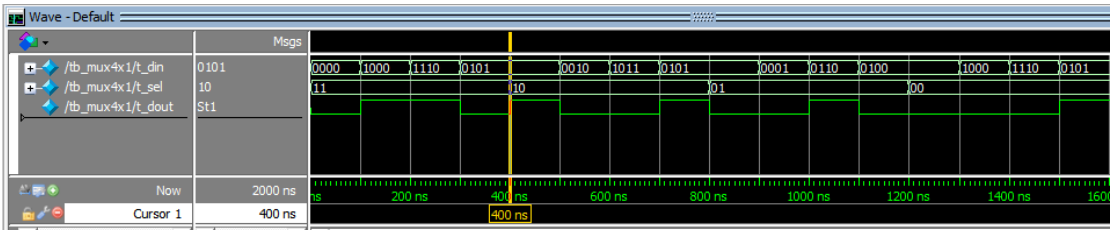
（2）测试模块

```
`timescale 10ns/1ns
`include "mux2x1.v"

module tb_mux2x1;
  reg [1:0] t_din;
  reg t_sel;
  wire t_dout;

  mux2x1 m_mux2x1(.dout(t_dout),
                  .sel(t_sel),
                  .din(t_din));

  initial begin
    t_sel = 1'b1; t_din = 2'b00;
    #10 t_din = 2'b10;
    #10 t_din = 2'b11;
    #10 t_din = 2'b01;
    #10 t_din = 2'b00;
    #10 t_din = 2'b10;
    #10 t_din = 2'b01;
    #10 t_sel = 1'b0;
    #10 t_din = 2'b00;
    #10 t_din = 2'b01;
  end
endmodule

```

```
`timescale 10ns/1ns
`include "mux4x1.v"

module tb_mux4x1;
  reg [3:0] t_din;
  reg [1:0] t_sel;
  wire t_dout;

  mux4x1 m_mux4x1(.dout(t_dout),
                  .sel(t_sel),
                  .din(t_din));

  initial begin
    t_sel = 2'b11; t_din = 4'b0000;
    #10 t_din = 4'b1000;
    #10 t_din = 4'b1110;
    #10 t_din = 4'b0101;
    #10 t_sel = 2'b10;
    #10 t_din = 4'b0010;
    #10 t_din = 4'b1011;
    #10 t_din = 4'b0101;
    #10 t_sel = 2'b01;
    #10 t_din = 4'b0001;
    #10 t_din = 4'b0110;
    #10 t_din = 4'b0100;
    #10 t_sel = 2'b00;
    #10 t_din = 4'b1000;
    #10 t_din = 4'b1110;
    #10 t_din = 4'b0101;
  end
endmodule
```

（3）测试波形图





第 4 题：

（1）设计模块

```
module comb_str(output Y,
                input A, B, C, D);
  wire [4:0] tmp;
  or U3(tmp[0], A, D);
  not U2(tmp[1], tmp[0]);
  not U1(tmp[2], D);
  and U4(tmp[3], B, C, tmp[2]);
  and U5(Y, tmp[3], tmp[1]);

endmodule
```

```
module comb_dataflow(output Y,
                     input A, B, C, D);
  wire [4:0] tmp;
  assign tmp[0] =  A | D;
  assign tmp[1] =  ~tmp[0];
  assign tmp[2] = ~D;
  assign tmp[3] =  B & C & tmp[2];
  assign Y = tmp[3] & tmp[1];

endmodule
```

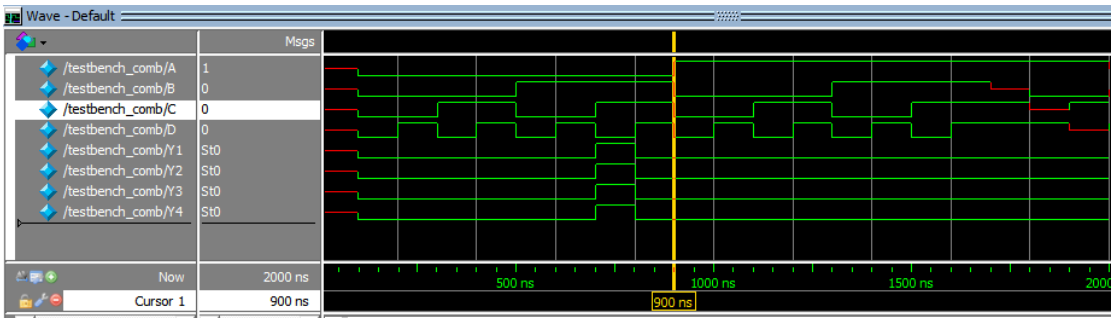
```
module comb_behavior(output Y,
                     input A, B, C, D);
  reg [4:0] tmp;
  reg Y_tmp;
  always @(*) begin
    tmp[0] =  A | D;
    tmp[1] =  ~tmp[0];
    tmp[2] = ~D;
    tmp[3] =  B & C & tmp[2];
    Y_tmp = tmp[3] & tmp[1];
  end
  assign Y = Y_tmp;
endmodule
```

```
primitive comb(output Y,
               input A, B, C, D);

table
  // A  B  C  D  :  Y;
     0  0  0  0  :  0;
     0  0  0  1  :  0;
     0  0  1  0  :  0;
     0  0  1  1  :  0;
     0  1  0  0  :  0;
     0  1  0  1  :  0;
     0  1  1  0  :  1;
     0  1  1  1  :  0;
     1  0  0  0  :  0;
     1  0  0  1  :  0;
     1  0  1  0  :  0;
     1  0  1  1  :  0;
     1  1  0  0  :  0;
     1  1  0  1  :  0;
     1  1  1  0  :  0;
     1  1  1  1  :  0;
endtable

endprimitive

module comb_prim(output Y,
                 input A, B, C, D);

  comb (Y, A, B, C, D);

endmodule
```

（2）测试模块

```
`timescale 10ns/1ns
`include "comb_dataflow.v"
`include "comb_str.v"
`include "comb_behavior.v"
`include "comb_prim.v"

module testbench_comb;
  reg A, B, C, D;
  wire Y1, Y2, Y3, Y4;

  comb_dataflow test1(.Y(Y1), .A(A), .B(B), .C(C), .D(D));
  comb_str test2(.Y(Y2), .A(A), .B(B), .C(C), .D(D));
  comb_behavior test3(.Y(Y3), .A(A), .B(B), .C(C), .D(D));
  comb_prim test4(.Y(Y4), .A(A), .B(B), .C(C), .D(D));

  initial begin
    #10 {A, B, C, D} = 4'b0000;
    #10 {A, B, C, D} = 4'b0001;
    #10 {A, B, C, D} = 4'b0010;
    #10 {A, B, C, D} = 4'b0011;
    #10 {A, B, C, D} = 4'b0100;
    #10 {A, B, C, D} = 4'b0101;
    #10 {A, B, C, D} = 4'b0110;
    #10 {A, B, C, D} = 4'b0111;
    #10 {A, B, C, D} = 4'b1000;
    #10 {A, B, C, D} = 4'b1001;
    #10 {A, B, C, D} = 4'b1010;
    #10 {A, B, C, D} = 4'b1011;
    #10 {A, B, C, D} = 4'b1100;
    #10 {A, B, C, D} = 4'b1101;
    #10 {A, B, C, D} = 4'b1110;
    #10 {A, B, C, D} = 4'b1111;
    #10 {A, B, C, D} = 4'b1x11;
    #10 {A, B, C, D} = 4'b10x1;
    #10 {A, B, C, D} = 4'b101x;
    #10 {A, B, C, D} = 4'bxx11;
    #10 {A, B, C, D} = 4'bxxx1;
    #10 {A, B, C, D} = 4'bxxxx;
  end

  initial begin
    $monitor("Current time: %tns", $time, "<---->A = %b, B = %b, C = %b, D = %b, Y1 = %b, Y2 = %b, Y3 = %b",
             A, B, C, D, Y1, Y2, Y3);
  end
endmodule
```

（3）测试波形图



（4）显示输出

```
# Current time:                    0ns<----->A = x, B = x, C = x, D = x, Y1 = x, Y2 = x, Y3 = x
# Current time:                  100ns<----->A = 0, B = 0, C = 0, D = 0, Y1 = 0, Y2 = 0, Y3 = 0
# Current time:                  200ns<----->A = 0, B = 0, C = 0, D = 1, Y1 = 0, Y2 = 0, Y3 = 0
# Current time:                  300ns<----->A = 0, B = 0, C = 1, D = 0, Y1 = 0, Y2 = 0, Y3 = 0
# Current time:                  400ns<----->A = 0, B = 0, C = 1, D = 1, Y1 = 0, Y2 = 0, Y3 = 0
# Current time:                  500ns<----->A = 0, B = 1, C = 0, D = 0, Y1 = 0, Y2 = 0, Y3 = 0
# Current time:                  600ns<----->A = 0, B = 1, C = 0, D = 1, Y1 = 0, Y2 = 0, Y3 = 0
# Current time:                  700ns<----->A = 0, B = 1, C = 1, D = 0, Y1 = 1, Y2 = 1, Y3 = 1
# Current time:                  800ns<----->A = 0, B = 1, C = 1, D = 1, Y1 = 0, Y2 = 0, Y3 = 0
# Current time:                  900ns<----->A = 1, B = 0, C = 0, D = 0, Y1 = 0, Y2 = 0, Y3 = 0
# Current time:                 1000ns<----->A = 1, B = 0, C = 0, D = 1, Y1 = 0, Y2 = 0, Y3 = 0
# Current time:                 1100ns<----->A = 1, B = 0, C = 1, D = 0, Y1 = 0, Y2 = 0, Y3 = 0
# Current time:                 1200ns<----->A = 1, B = 0, C = 1, D = 1, Y1 = 0, Y2 = 0, Y3 = 0
# Current time:                 1300ns<----->A = 1, B = 1, C = 0, D = 0, Y1 = 0, Y2 = 0, Y3 = 0
# Current time:                 1400ns<----->A = 1, B = 1, C = 0, D = 1, Y1 = 0, Y2 = 0, Y3 = 0
# Current time:                 1500ns<----->A = 1, B = 1, C = 1, D = 0, Y1 = 0, Y2 = 0, Y3 = 0
# Current time:                 1600ns<----->A = 1, B = 1, C = 1, D = 1, Y1 = 0, Y2 = 0, Y3 = 0
# Current time:                 1700ns<----->A = 1, B = x, C = 1, D = 1, Y1 = 0, Y2 = 0, Y3 = 0
# Current time:                 1800ns<----->A = 1, B = 0, C = x, D = 1, Y1 = 0, Y2 = 0, Y3 = 0
# Current time:                 1900ns<----->A = 1, B = 0, C = 1, D = x, Y1 = 0, Y2 = 0, Y3 = 0
```

（5）设计说明

四种设计方式均保证了相同的输出，特别是存在 x 时，均保证了只有输入全为 x，输出才为 x，其余情况均为 0。(除 UDP 方式，以为其不考虑 x 的情况，只考虑 0、1 二值的情况)

第 5 题

（1）设计模块

```
1   module comb_Y1(output Y,
2                  input A, B, C);
3
4   wire Y1, Y2, Y3;
5
6     assign Y1 = (~A & B & C);
7     assign Y2 = (~A & ~B & C);
8     assign Y3 = (A & ~B);
9
10    assign Y = Y1 | Y2 | Y3;
11
12  endmodule
13
```

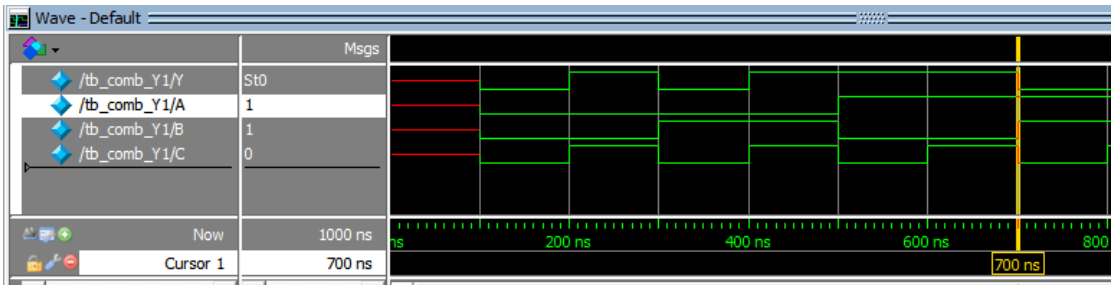
```
module comb_Y2(output Y,
               input A, B, C, D);

  assign Y = (~A & B) | (A & B & ~C) | (A & ~B & C & D);

endmodule
```

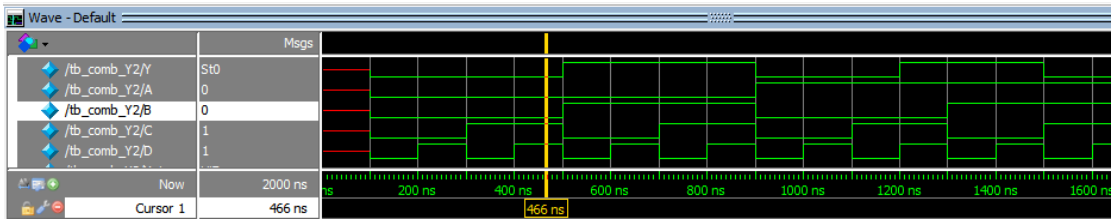
（2）测试模块

```
`timescale 10ns/1ns
`include "comb_Y2.v"

module tb_comb_Y2;

  wire Y;
  reg A, B, C, D;
  wire Y;

  comb_Y2 m_comb_Y2(Y, A, B, C, D);
  initial begin
    #10 {A, B, C, D} = 4'b0000;
    #10 {A, B, C, D} = 4'b0001;
    #10 {A, B, C, D} = 4'b0010;
    #10 {A, B, C, D} = 4'b0011;
    #10 {A, B, C, D} = 4'b0100;
    #10 {A, B, C, D} = 4'b0101;
    #10 {A, B, C, D} = 4'b0110;
    #10 {A, B, C, D} = 4'b0111;
    #10 {A, B, C, D} = 4'b1000;
    #10 {A, B, C, D} = 4'b1001;
    #10 {A, B, C, D} = 4'b1010;
    #10 {A, B, C, D} = 4'b1011;
    #10 {A, B, C, D} = 4'b1100;
    #10 {A, B, C, D} = 4'b1101;
    #10 {A, B, C, D} = 4'b1110;
    #10 {A, B, C, D} = 4'b1111;
  end

  initial begin
    $monitor("Current time: %tns", $time, "<---->A = %b, B = %b, C = %b, D = %b, Y2 = %b",
             A, B, C, D, Y);
  end

endmodule
```

```
`timescale 10ns/1ns
`include "comb_Y1.v"

module tb_comb_Y1;

  wire Y;
  reg A, B, C;

  comb_Y1 m_comb_Y1(.Y(Y), .A(A), .B(B), .C(C));

  initial begin
    #10 {A, B, C} = 3'b000;
    #10 {A, B, C} = 3'b001;
    #10 {A, B, C} = 3'b010;
    #10 {A, B, C} = 3'b011;
    #10 {A, B, C} = 3'b100;
    #10 {A, B, C} = 3'b101;
    #10 {A, B, C} = 3'b110;
    #10 {A, B, C} = 3'b111;
  end

  initial begin
    $monitor("Current time: %tns", $time, "<---->A = %b, B = %b, C = %b, Y1 = %b",
             A, B, C, Y);
  end

endmodule
```

（3）测试波形图





（4）显示输出

```
VSIM 35> run
# Current time:                  0ns<---->A = x, B = x, C = x, Y1 = x
# Current time:                100ns<---->A = 0, B = 0, C = 0, Y1 = 0
# Current time:                200ns<---->A = 0, B = 0, C = 1, Y1 = 1
# Current time:                300ns<---->A = 0, B = 1, C = 0, Y1 = 0
# Current time:                400ns<---->A = 0, B = 1, C = 1, Y1 = 1
# Current time:                500ns<---->A = 1, B = 0, C = 0, Y1 = 1
# Current time:                600ns<---->A = 1, B = 0, C = 1, Y1 = 1
# Current time:                700ns<---->A = 1, B = 1, C = 0, Y1 = 0
# Current time:                800ns<---->A = 1, B = 1, C = 1, Y1 = 0

VSIM 45> run
# Current time:                  0ns<---->A = x, B = x, C = x, D = x, Y2 = x
# Current time:                100ns<---->A = 0, B = 0, C = 0, D = 0, Y2 = 0
# Current time:                200ns<---->A = 0, B = 0, C = 0, D = 1, Y2 = 0
# Current time:                300ns<---->A = 0, B = 0, C = 1, D = 0, Y2 = 0
# Current time:                400ns<---->A = 0, B = 0, C = 1, D = 1, Y2 = 0
# Current time:                500ns<---->A = 0, B = 1, C = 0, D = 0, Y2 = 1
# Current time:                600ns<---->A = 0, B = 1, C = 0, D = 1, Y2 = 1
# Current time:                700ns<---->A = 0, B = 1, C = 1, D = 0, Y2 = 1
# Current time:                800ns<---->A = 0, B = 1, C = 1, D = 1, Y2 = 1
# Current time:                900ns<---->A = 1, B = 0, C = 0, D = 0, Y2 = 0
# Current time:               1000ns<---->A = 1, B = 0, C = 0, D = 1, Y2 = 0
# Current time:               1100ns<---->A = 1, B = 0, C = 1, D = 0, Y2 = 0
# Current time:               1200ns<---->A = 1, B = 0, C = 1, D = 1, Y2 = 1
# Current time:               1300ns<---->A = 1, B = 1, C = 0, D = 0, Y2 = 1
# Current time:               1400ns<---->A = 1, B = 1, C = 0, D = 1, Y2 = 1
# Current time:               1500ns<---->A = 1, B = 1, C = 1, D = 0, Y2 = 0
# Current time:               1600ns<---->A = 1, B = 1, C = 1, D = 1, Y2 = 0
```

第 6 题：

（1）设计模块

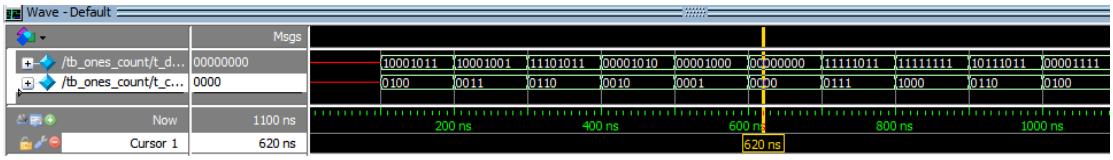
```
module ones_count(output reg [3:0] count,
                  input [7:0] dat_in);
parameter BITS = 8;
integer i;

always @(dat_in) begin

i = 0;

count = 0;
  while(i < BITS) begin
    if (dat_in[i] == 1'b1)
      count = count + 1;
    i = i + 1;
  end
end

endmodule
```

```
`timescale 10ns/1ns
`include "ones_count.v"

module tb_ones_count;
  reg [7:0] t_dat_in;
  wire [3:0] t_count;

  ones_count m_ones_count(.dat_in(t_dat_in),
                          .count(t_count));

  initial begin
    #10 t_dat_in = 8'b1000_1011;
    #10 t_dat_in = 8'b1000_1001;
    #10 t_dat_in = 8'b1110_1011;
    #10 t_dat_in = 8'b0000_1010;
    #10 t_dat_in = 8'b0000_1000;
    #10 t_dat_in = 8'b0000_0000;
    #10 t_dat_in = 8'b1111_1011;
    #10 t_dat_in = 8'b1111_1111;
    #10 t_dat_in = 8'b1011_1011;
    #10 t_dat_in = 8'b0000_1111;
  end

  initial begin
    $monitor("Current time: %tns", $time, "<---->count = %b, dat_in = %b", t_dat_in, t_count);
  end

endmodule
```

（2）测试模块

```
`timescale 10ns/1ns
`include "ones_count.v"

module tb_ones_count;
  reg [7:0] t_dat_in;
  wire [3:0] t_count;

  ones_count m_ones_count(.dat_in(t_dat_in),
                          .count(t_count));

  initial begin
    #10 t_dat_in = 8'b1000_1011;
    #10 t_dat_in = 8'b1000_1001;
    #10 t_dat_in = 8'b1110_1011;
    #10 t_dat_in = 8'b0000_1010;
    #10 t_dat_in = 8'b0000_1000;
    #10 t_dat_in = 8'b0000_0000;
    #10 t_dat_in = 8'b1111_1011;
    #10 t_dat_in = 8'b1111_1111;
    #10 t_dat_in = 8'b1011_1011;
    #10 t_dat_in = 8'b0000_1111;
  end

  initial begin
    $monitor("Current time: %tns", $time, "<---->count = %b, dat_in = %b", t_dat_in, t_count);
  end

endmodule
```

（3）测试波形图



（4）显示输出

```
VSIM 53> run
# Current time:                    0ns<---->count = xxxxxxxx, dat_in = xxxx
# Current time:                  100ns<---->count = 10001011, dat_in = 0100
# Current time:                  200ns<---->count = 10001001, dat_in = 0011
# Current time:                  300ns<---->count = 11101011, dat_in = 0110
# Current time:                  400ns<---->count = 00001010, dat_in = 0010
# Current time:                  500ns<---->count = 00001000, dat_in = 0001
# Current time:                  600ns<---->count = 00000000, dat_in = 0000
# Current time:                  700ns<---->count = 11111011, dat_in = 0111
# Current time:                  800ns<---->count = 11111111, dat_in = 1000
# Current time:                  900ns<---->count = 10111011, dat_in = 0110
# Current time:                 1000ns<---->count = 00001111, dat_in = 0100
```

第 7 题

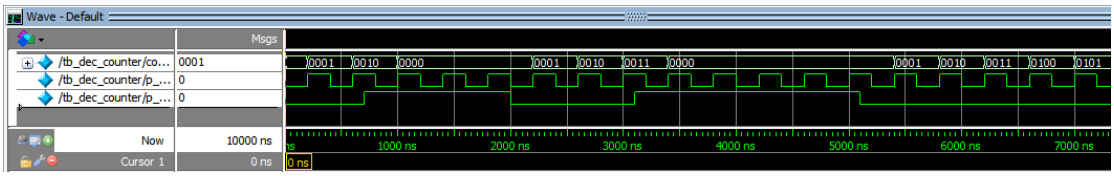
（1）设计模块

```
module dec_counter(output [3:0] count,
                   input clk, reset);

  reg [3:0] cnt = 0;

  always @(posedge clk) begin
    if (clk & ~reset) begin
      if (cnt == 4'b1010) cnt = 0;
      else cnt = cnt + 1;
    end
    else begin
        cnt = 0;
    end
  end

  assign count = cnt;

endmodule
```

（2）测试模块

```
`timescale 10ns/1ns
`include "dec_counter.v"
`define CLOCK_CYCLE 20

module tb_dec_counter;

  reg p_clk_in, p_rst;
  wire [3:0] count;

  dec_counter m_dec_counter(count, p_clk_in, p_rst);

  initial begin
    p_clk_in = 1'b0;
    forever #`CLOCK_CYCLE p_clk_in = ~p_clk_in;
  end

  initial begin
    p_rst = 1'b0;

    #70 p_rst = 1'b1;
    #130 p_rst = 1'b0;
    #110 p_rst = 1;
    #200 p_rst = 0;
    #1000 $stop;
  end

  initial begin
    $monitor("Current time: %tns", $time, "<---->count = %b, p_clk_in = %b, p_rst = %b", count, p_clk_in, p_rst);
  end

endmodule
```

（3）测试波形图



（4）显示输出

```
# Current time:                  4000ns<----->count = 0000, p_clk_in = 0, p_rst = 1
# Current time:                  4200ns<----->count = 0000, p_clk_in = 1, p_rst = 1
# Current time:                  4400ns<----->count = 0000, p_clk_in = 0, p_rst = 1
# Current time:                  4600ns<----->count = 0000, p_clk_in = 1, p_rst = 1
# Current time:                  4800ns<----->count = 0000, p_clk_in = 0, p_rst = 1
# Current time:                  5000ns<----->count = 0000, p_clk_in = 1, p_rst = 1
# Current time:                  5100ns<----->count = 0000, p_clk_in = 1, p_rst = 0
# Current time:                  5200ns<----->count = 0000, p_clk_in = 0, p_rst = 0
# Current time:                  5400ns<----->count = 0001, p_clk_in = 1, p_rst = 0
# Current time:                  5600ns<----->count = 0001, p_clk_in = 0, p_rst = 0
# Current time:                  5800ns<----->count = 0010, p_clk_in = 1, p_rst = 0
# Current time:                  6000ns<----->count = 0010, p_clk_in = 0, p_rst = 0
# Current time:                  6200ns<----->count = 0011, p_clk_in = 1, p_rst = 0
```

第 8 题

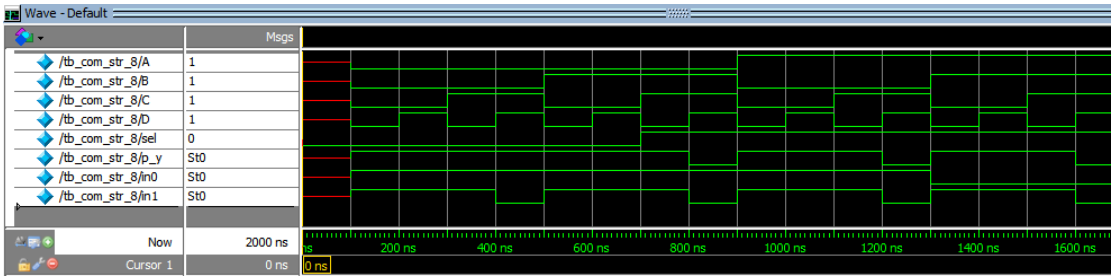
（1）设计模块

```
1   module comb_str(output y,
2                   input sel, A, B, C, D);
3
4     wire [3:0] tmp;
5     nand (tmp[0], A, B);
6     nand (tmp[1], C, D);
7     and (tmp[2], ~sel, tmp[0]);
8     and (tmp[3], sel, tmp[1]);
9     or (y, tmp[2], tmp[3]);
10
11  endmodule
```

（2）测试模块

```
1   `timescale 10ns/1ns
2   `include "ones_count.v"
3
4   module tb_ones_count;
5     reg [7:0] t_dat_in;
6     wire [3:0] t_count;
7
8     ones_count m_ones_count(.dat_in(t_dat_in),
9                             .count(t_count));
10
11    initial begin
12      #10 t_dat_in = 8'b1000_1011;
13      #10 t_dat_in = 8'b1000_1001;
14      #10 t_dat_in = 8'b1110_1011;
15      #10 t_dat_in = 8'b0000_1010;
16      #10 t_dat_in = 8'b0000_1000;
17      #10 t_dat_in = 8'b0000_0000;
18      #10 t_dat_in = 8'b1111_1011;
19      #10 t_dat_in = 8'b1111_1111;
20      #10 t_dat_in = 8'b1011_1011;
21      #10 t_dat_in = 8'b0000_1111;
22    end
23
24    initial begin
25      $monitor("Current time: %tns", $time, "<---->count = %b, dat_in = %b", t_dat_in, t_count);
26    end
27
28  endmodule
```

（3）测试波形图



（4）显示输出

```
VSIM 59> run
# Current time:                 0ns<---->A = x, B = x, C = x, D = x, sel 0, y = x
# Current time:               100ns<---->A = 0, B = 0, C = 0, D = 0, sel 0, y = 1
# Current time:               200ns<---->A = 0, B = 0, C = 0, D = 1, sel 0, y = 1
# Current time:               300ns<---->A = 0, B = 0, C = 1, D = 0, sel 0, y = 1
# Current time:               400ns<---->A = 0, B = 0, C = 1, D = 1, sel 0, y = 1
# Current time:               500ns<---->A = 0, B = 1, C = 0, D = 0, sel 0, y = 1
# Current time:               600ns<---->A = 0, B = 1, C = 0, D = 1, sel 0, y = 1
# Current time:               700ns<---->A = 0, B = 1, C = 1, D = 0, sel 1, y = 1
# Current time:               800ns<---->A = 0, B = 1, C = 1, D = 1, sel 1, y = 0
# Current time:               900ns<---->A = 1, B = 0, C = 0, D = 0, sel 1, y = 1
# Current time:              1000ns<---->A = 1, B = 0, C = 0, D = 1, sel 1, y = 1
# Current time:              1100ns<---->A = 1, B = 0, C = 1, D = 0, sel 1, y = 1
# Current time:              1200ns<---->A = 1, B = 0, C = 1, D = 1, sel 1, y = 0
# Current time:              1300ns<---->A = 1, B = 1, C = 0, D = 0, sel 1, y = 1
# Current time:              1400ns<---->A = 1, B = 1, C = 0, D = 1, sel 1, y = 1
# Current time:              1500ns<---->A = 1, B = 1, C = 1, D = 0, sel 1, y = 1
# Current time:              1600ns<---->A = 1, B = 1, C = 1, D = 1, sel 1, y = 0
```

第 9 题

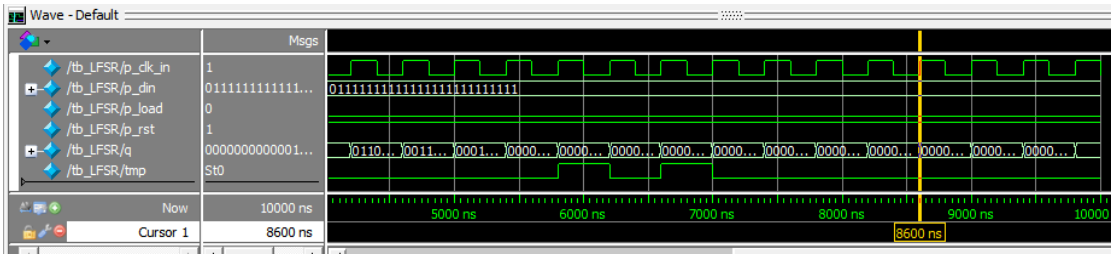
（1）设计模块

```
module LFSR(output reg [1:26] q,
            input clk,
            input rst_n,
            input load,
            input [1:26] din);

  reg out;
  always @(posedge clk) begin
    if (~rst_n)
      q <= 26'b0;
    else begin
      if (load)
        q <= (!din) ? din : 26'b1;
      else begin
        if (q == 26'b0)
          q <= 26'b1;
        else begin
          q[10:26] <= q[9:25];
          q[9] <= q[8] ^ q[26];
          q[8] <= q[7] ^ q[26];
          q[3:7] <= q[2:6];
          q[2] <= q[1] ^ q[26];
          q[1] <= q[26];
        end
      end
    end
  end

endmodule
```

（2）测试模块

```
`timescale 10ns/1ns
`include "LFSR.v"
`define CLOCK_CYCLE 20

module tb_LFSR;
  reg p_clk_in, p_rst, p_load;
  reg [1:26] p_din;
  wire [1:26] q;

  wire tmp;

  LFSR m_LFSR(q, p_clk_in, p_rst, p_load, p_din);

  initial begin
    p_clk_in = 1'b0;
    forever #`CLOCK_CYCLE p_clk_in = ~p_clk_in;
  end

  assign tmp = q[26] ^ q[8] ^ q[7] ^ q[1];

  initial begin
    p_din = 26'h3ffff_ff; p_load = 1;
    #40 p_din = 26'h2ffff_ff; p_load = 0;
    #150 p_din = 26'h1ffff_ff; p_load = 1;
    #180 p_load = 0;
  end

  initial begin
    p_rst = 1'b1;

    #70 p_rst = 1'b0;
    #130 p_rst = 1'b0;
    #110 p_rst = 1;
    #1000 $stop;
  end

  initial begin
    $monitor("Current time: %tns", $time, "<---->q = %b, p_din = %b, p_load = %b, p_rst = %b, tmp = %b",
             q, p_din, p_load, p_rst, tmp);
  end
endmodule
```

（3）测试波形图



（4）显示输出

```
# Current time:                0ns<---->q = xxxxxxxxxxxxxxxxxxxxxxxxx, p_din = 11111111111111111111111111, p_load = 1, p_rst = 1, tmp = x
# Current time:              200ns<---->q = 00000000000000000000000001, p_din = 11111111111111111111111111, p_load = 1, p_rst = 1, tmp = 1
# Current time:              400ns<---->q = 00000000000000000000000001, p_din = 10111111111111111111111111, p_load = 0, p_rst = 1, tmp = 1
# Current time:              600ns<---->q = 11000001100000000000000000, p_din = 10111111111111111111111111, p_load = 0, p_rst = 1, tmp = 0
# Current time:              700ns<---->q = 11000001100000000000000000, p_din = 10111111111111111111111111, p_load = 0, p_rst = 0, tmp = 0
# Current time:             1000ns<---->q = 00000000000000000000000000, p_din = 10111111111111111111111111, p_load = 0, p_rst = 0, tmp = 0
# Current time:             1900ns<---->q = 00000000000000000000000000, p_din = 01111111111111111111111111, p_load = 1, p_rst = 0, tmp = 0
# Current time:             3100ns<---->q = 00000000000000000000000000, p_din = 01111111111111111111111111, p_load = 1, p_rst = 1, tmp = 0
# Current time:             3400ns<---->q = 00000000000000000000000001, p_din = 01111111111111111111111111, p_load = 1, p_rst = 1, tmp = 1
# Current time:             3700ns<---->q = 00000000000000000000000001, p_din = 01111111111111111111111111, p_load = 0, p_rst = 1, tmp = 1
# Current time:             3800ns<---->q = 11000001100000000000000000, p_din = 01111111111111111111111111, p_load = 0, p_rst = 1, tmp = 0
# Current time:             4200ns<---->q = 01100000110000000000000000, p_din = 01111111111111111111111111, p_load = 0, p_rst = 1, tmp = 0
# Current time:             4600ns<---->q = 00110000011000000000000000, p_din = 01111111111111111111111111, p_load = 0, p_rst = 1, tmp = 0
# Current time:             5000ns<---->q = 00011000001100000000000000, p_din = 01111111111111111111111111, p_load = 0, p_rst = 1, tmp = 0
# Current time:             5400ns<---->q = 00001100000110000000000000, p_din = 01111111111111111111111111, p_load = 0, p_rst = 1, tmp = 0
# Current time:             5800ns<---->q = 00000110000011000000000000, p_din = 01111111111111111111111111, p_load = 0, p_rst = 1, tmp = 1
# Current time:             6200ns<---->q = 00000011000001100000000000, p_din = 01111111111111111111111111, p_load = 0, p_rst = 1, tmp = 0
# Current time:             6600ns<---->q = 00000001100000110000000000, p_din = 01111111111111111111111111, p_load = 0, p_rst = 1, tmp = 1
# Current time:             7000ns<---->q = 00000000110000011000000000, p_din = 01111111111111111111111111, p_load = 0, p_rst = 1, tmp = 0
```

（5）设计说明

第 10 题

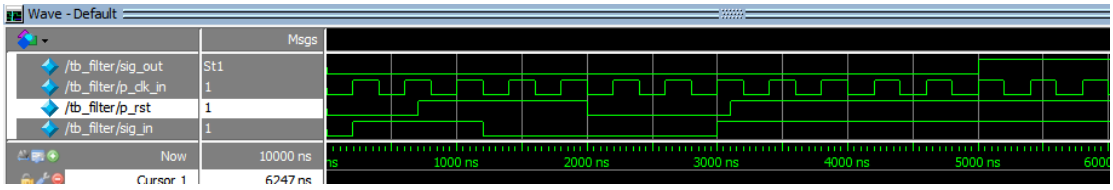
（1）设计模块

```
1   module filter(output reg sig_out,
2                 input clock,
3                 input reset,
4                 input sig_in);
5
6     reg [3:0] q;
7     wire J, K;
8
9     always @(posedge clock or negedge reset) begin
10      if (!reset) q[0] <= 0;
11      else q[0] <= sig_in;
12    end
13
14    always @(posedge clock or negedge reset) begin
15      if (!reset) q[1] <= 0;
16      else q[1] <= q[0];
17    end
18
19    always @(posedge clock or negedge reset) begin
20      if (!reset) q[2] <= 0;
21      else q[2] <= q[1];
22    end
23
24    always @(posedge clock or negedge reset) begin
25      if (!reset) q[3] <= 0;
26      else q[3] <= q[2];
27    end
28
29    assign J = q[1] & q[2] & q[3];
30    assign K = ~q[1] & ~q[2] & ~q[3];
31
32    always @(posedge clock or negedge reset) begin
33      if (!reset) sig_out <= 0;
34      else begin
35        sig_out <= J & ~sig_out | ~K & sig_out;
36      end
37    end
```

（2）测试模块

```
1   `timescale 10ns/1ns
2   `include "filter.v"
3   `define CLOCK_CYCLE 20
4
5   module tb_filter;
6
7     wire sig_out;
8     reg p_clk_in, p_rst, sig_in;
9
10    filter m_filter(sig_out, p_clk_in, p_rst, sig_in);
11
12    initial begin
13      sig_in = 0;
14      #20 sig_in = 1;
15      #100 sig_in = 0;
16      #180 sig_in = 1;
17    end
18
19    initial begin
20      p_clk_in = 1'b0;
21      forever #`CLOCK_CYCLE p_clk_in = ~p_clk_in;
22    end
23
24    initial begin
25      p_rst = 1'b0;
26
27      #70 p_rst = 1'b1;
28      #130 p_rst = 1'b0;
29      #110 p_rst = 1;
30      #1000 $stop;
31    end
32
33    initial begin
34      $monitor("Current time: %tns", $time, "<---->clk = %b, reset = %b, sig_in = %b, sig_out = %b, q = %b, J = %b, K = %b",
35                p_clk_in, p_rst, sig_in, sig_out, m_filter.q, m_filter.J, m_filter.K);
36    end
37
38  endmodule
```

（3）测试波形图



（4）显示输出

```
# Current time:              1600ns<---->clk = 0, reset = 1, sig_in = 0, sig_out = 0, q = 0010, J = 0, K = 0
# Current time:              1800ns<---->clk = 1, reset = 1, sig_in = 0, sig_out = 0, q = 0100, J = 0, K = 0
# Current time:              2000ns<---->clk = 0, reset = 0, sig_in = 0, sig_out = 0, q = 0000, J = 0, K = 1
# Current time:              2200ns<---->clk = 1, reset = 0, sig_in = 0, sig_out = 0, q = 0000, J = 0, K = 1
# Current time:              2400ns<---->clk = 0, reset = 0, sig_in = 0, sig_out = 0, q = 0000, J = 0, K = 1
# Current time:              2600ns<---->clk = 1, reset = 0, sig_in = 0, sig_out = 0, q = 0000, J = 0, K = 1
# Current time:              2800ns<---->clk = 0, reset = 0, sig_in = 0, sig_out = 0, q = 0000, J = 0, K = 1
# Current time:              3000ns<---->clk = 1, reset = 0, sig_in = 1, sig_out = 0, q = 0000, J = 0, K = 1
# Current time:              3100ns<---->clk = 1, reset = 1, sig_in = 1, sig_out = 0, q = 0000, J = 0, K = 1
# Current time:              3200ns<---->clk = 0, reset = 1, sig_in = 1, sig_out = 0, q = 0000, J = 0, K = 1
# Current time:              3400ns<---->clk = 1, reset = 1, sig_in = 1, sig_out = 0, q = 0001, J = 0, K = 1
# Current time:              3600ns<---->clk = 0, reset = 1, sig_in = 1, sig_out = 0, q = 0001, J = 0, K = 1
# Current time:              3800ns<---->clk = 1, reset = 1, sig_in = 1, sig_out = 0, q = 0011, J = 0, K = 0
# Current time:              4000ns<---->clk = 0, reset = 1, sig_in = 1, sig_out = 0, q = 0011, J = 0, K = 0
# Current time:              4200ns<---->clk = 1, reset = 1, sig_in = 1, sig_out = 0, q = 0111, J = 0, K = 0
# Current time:              4400ns<---->clk = 0, reset = 1, sig_in = 1, sig_out = 0, q = 0111, J = 0, K = 0
```

第 11 题

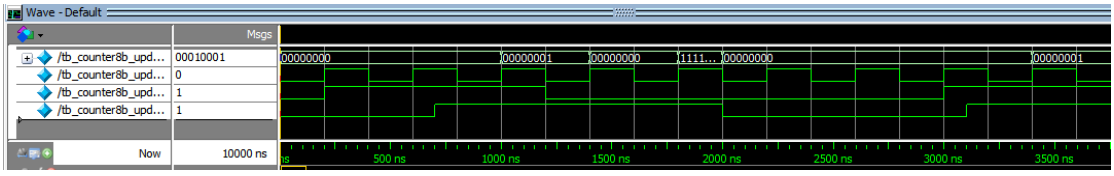
（1）设计模块

```
module counter8b_updown(output reg [7:0] count,
                        input clk,
                        input reset,
                        input dir);

  always @(posedge clk or negedge reset) begin
    if (~reset) begin
      count = 0;
    end
    else begin
      if (dir == 1) count = count + 1;
      else count = count - 1;
    end
  end
endmodule
```

（2）测试模块

```
`timescale 10ns/1ns
`include "counter8b_updown.v"
`define CLOCK_CYCLE 20

module tb_counter8b_updown;

  wire [7:0] count;
  reg p_clk_in, p_rst, p_dir;

  counter8b_updown m_counter_updown(count, p_clk_in, p_rst, p_dir);

  initial begin
    p_dir = 0;
    #20 p_dir = 1;
    #100 p_dir = 0;
    #180 p_dir = 1;
  end

  initial begin
    p_clk_in = 1'b0;
    forever #`CLOCK_CYCLE p_clk_in = ~p_clk_in;
  end

  initial begin
    p_rst = 1'b0;

    #70 p_rst = 1'b1;
    #130 p_rst = 1'b0;
    #110 p_rst = 1;
    #1000 $stop;
  end

  initial begin
    $monitor("Current time: %tns", $time, "<---->clk = %b, reset = %b, p_dir = %b, count = %b",
             p_clk_in, p_rst, p_dir, count);
  end
```

（3）测试波形图



（4）显示输出

```
# Current time:                1000ns<---->clk = 1, reset = 1, p_dir = 1, count = 00000001
# Current time:                1200ns<---->clk = 0, reset = 1, p_dir = 0, count = 00000001
# Current time:                1400ns<---->clk = 1, reset = 1, p_dir = 0, count = 00000000
# Current time:                1600ns<---->clk = 0, reset = 1, p_dir = 0, count = 00000000
# Current time:                1800ns<---->clk = 1, reset = 1, p_dir = 0, count = 11111111
```

```
# Current time:                2800ns<---->clk = 0, reset = 0, p_dir = 0, count = 00000000
# Current time:                3000ns<---->clk = 1, reset = 0, p_dir = 1, count = 00000000
# Current time:                3100ns<---->clk = 1, reset = 1, p_dir = 1, count = 00000000
# Current time:                3200ns<---->clk = 0, reset = 1, p_dir = 1, count = 00000000
# Current time:                3400ns<---->clk = 1, reset = 1, p_dir = 1, count = 00000001
# Current time:                3600ns<---->clk = 0, reset = 1, p_dir = 1, count = 00000001
# Current time:                3800ns<---->clk = 1, reset = 1, p_dir = 1, count = 00000010
```

第 12 题

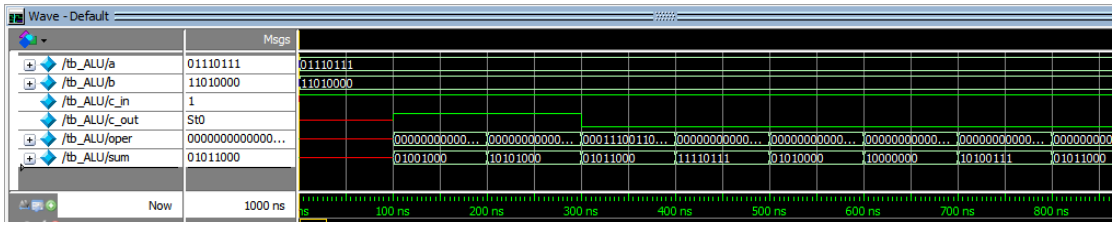
（1）设计模块

```
module ALU(output reg c_out,
           output reg [7:0] sum,
           input reg [8*10+1:0]oper,
           input [7:0] a, b,
           input c_in);

  always @(oper, a, b) begin
    case (oper)
      "and":          begin tmp = a + b + c_in; end
      "subtract":     begin tmp = a + ~b + c_in; end
      "subtract_a":   begin tmp = b + ~a + ~c_in;  end
      "or_ab":        {c_out, sum} = {1'b0, a | b};
      "and_ab":       {c_out, sum} = {1'b0, a & b};
      "not_ab":       {c_out, sum} = {1'b0, ~a & b};
      "exor":         {c_out, sum} = {1'b0, a ^ b};
      "exnor":        {c_out, sum} = {1'b0, a ~^ b};
    endcase
  end

endmodule

```

（2）测试模块

```
`timescale 10ns/1ns
`include "ALU.v"

module tb_ALU;

  wire [7:0] sum;
  wire c_out;
  reg [7:0] a, b;
  reg c_in;
  reg [8*10+1:0] oper;

  ALU m_ALU(c_out, sum, oper, a, b, c_in);


  initial begin
    a = 8'b0111_0111;
    b = 8'b1101_0000;
    c_in = 1;
  end

  initial begin
    #10 oper = "and";
    #10 oper = "subtract";
    #10 oper = "subtract_a";
    #10 oper = "or_ab";
    #10 oper = "and_ab";
    #10 oper = "not_ab";
    #10 oper = "exor";
    #10 oper = "exnor";
  end


  initial begin
    $monitor("Current time: %tns", $time, "<---->sum = %b, c_out = %b, oper = %s, a = %b, b = %b, c_in = %b, ",
             sum, c_out, oper, a, b, c_in);
  end

endmodule
```

（3）测试波形图



（4）显示输出

```
SIM 81> run
 Current time:                  0ns<---->sum = xxxxxxxx, c_out = x, oper =           , a = 01110111, b = 11010000, c_in = 1,
 Current time:                100ns<---->sum = 01001000, c_out = 1, oper =         and, a = 01110111, b = 11010000, c_in = 1,
 Current time:                200ns<---->sum = 10101000, c_out = 1, oper =    subtract, a = 01110111, b = 11010000, c_in = 1,
 Current time:                300ns<---->sum = 01011000, c_out = 0, oper =  subtract_a, a = 01110111, b = 11010000, c_in = 1,
 Current time:                400ns<---->sum = 11110111, c_out = 0, oper =       or_ab, a = 01110111, b = 11010000, c_in = 1,
 Current time:                500ns<---->sum = 01010000, c_out = 0, oper =      and_ab, a = 01110111, b = 11010000, c_in = 1,
 Current time:                600ns<---->sum = 10000000, c_out = 0, oper =      not_ab, a = 01110111, b = 11010000, c_in = 1,
 Current time:                700ns<---->sum = 10100111, c_out = 0, oper =        exor, a = 01110111, b = 11010000, c_in = 1,
 Current time:                800ns<---->sum = 01011000, c_out = 0, oper =       exnor, a = 01110111, b = 11010000, c_in = 1,
```

（5）设计说明

本题采用 reg 保存字符串信息，利用 case 语句，从而实现算术逻辑单元（ALU）的基本单元，如全加器，全减器，异或等。

第 13 题

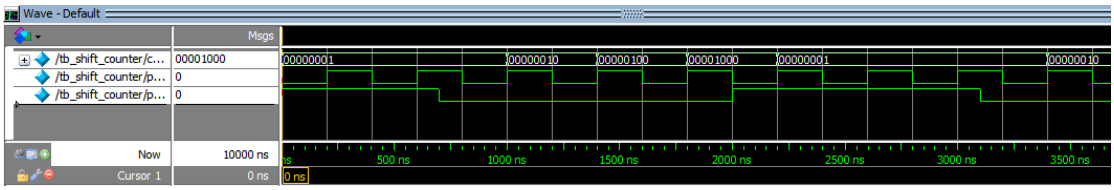
（1）设计模块

```
1   module shift_counter(output reg [7:0] count,
2                        input clk,
3                        input reset);
4
5     reg LEFT = 1'b1;
6
7     initial begin
8       count = 1;
9     end
10
11    always @(posedge clk) begin
12      if (reset) begin
13        LEFT = 1'b1;
14        count = 1;
15      end
16      else begin
17        if (LEFT) begin
18          count <= {count[6:0], 1'b0};
19          if (count == 8'b0100_0000) LEFT = 1'b0;
20        end
21        else begin
22          count <= {1'b0, count[7:1]};
23          if (count == 8'b0000_0010) LEFT = 1'b1;
24        end
25      end
26    end
27
28  endmodule
```

（2）测试模块

```
1   `timescale 10ns/1ns
2   `include "shift_counter.v"
3   `define CLOCK_CYCLE 20
4
5   module tb_shift_counter;
6
7     reg p_clk_in, p_rst;
8     wire [7:0] count;
9
10    shift_counter m_shift_counter(count, p_clk_in, p_rst);
11
12    initial begin
13      p_clk_in = 1'b0;
14      forever #`CLOCK_CYCLE p_clk_in = ~p_clk_in;
15    end
16
17    initial begin
18      p_rst = 1'b1;
19      #70 p_rst = 1'b0;
20      #130 p_rst = 1'b1;
21      #110 p_rst = 0;
22
23      #1000 $stop;
24    end
25
26    initial begin
27      $monitor("Current time: %tns", $time, "<---->clk = %b, reset = %b, count = %b",
28               p_clk_in, p_rst, count);
29    end
30
31  endmodule
```

（3）测试波形图



（4）显示输出

```
VSIM 85> run
# Current time:                    0ns<---->clk = 0, reset = 1, count = 00000001
# Current time:                  200ns<---->clk = 1, reset = 1, count = 00000001
# Current time:                  400ns<---->clk = 0, reset = 1, count = 00000001
# Current time:                  600ns<---->clk = 1, reset = 1, count = 00000001
# Current time:                  700ns<---->clk = 1, reset = 0, count = 00000001
# Current time:                  800ns<---->clk = 0, reset = 0, count = 00000001
# Current time:                 1000ns<---->clk = 1, reset = 0, count = 00000010
# Current time:                 1200ns<---->clk = 0, reset = 0, count = 00000010
# Current time:                 1400ns<---->clk = 1, reset = 0, count = 00000100
# Current time:                 1600ns<---->clk = 0, reset = 0, count = 00000100
# Current time:                 1800ns<---->clk = 1, reset = 0, count = 00001000
# Current time:                 2000ns<---->clk = 0, reset = 1, count = 00001000
# Current time:                 2200ns<---->clk = 1, reset = 1, count = 00000001
# Current time:                 2400ns<---->clk = 0, reset = 1, count = 00000001
# Current time:                 2600ns<---->clk = 1, reset = 1, count = 00000001
# Current time:                 2800ns<---->clk = 0, reset = 1, count = 00000001
# Current time:                 3000ns<---->clk = 1, reset = 1, count = 00000001
# Current time:                 3100ns<---->clk = 1, reset = 0, count = 00000001
# Current time:                 3200ns<---->clk = 0, reset = 0, count = 00000001
# Current time:                 3400ns<---->clk = 1, reset = 0, count = 00000010
```

第 14 题

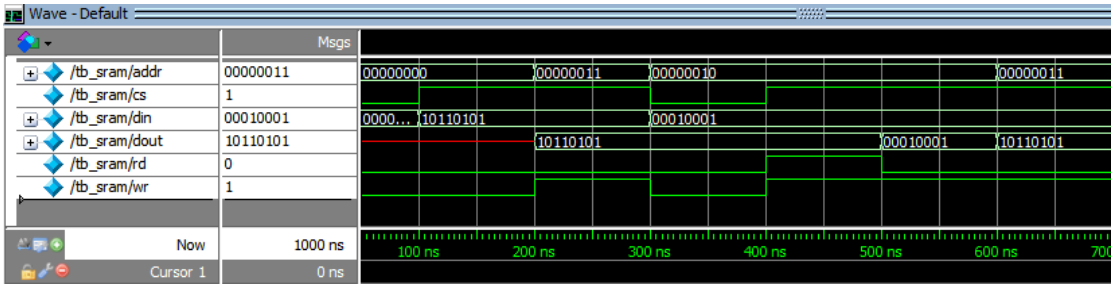
（1）设计模块

```
module sram(output [7:0] dout,
            input [7:0] din, addr,
            input wr, rd, cs);

  reg [7:0] mem [255:0];

  always @(posedge wr) begin
    if (cs == 1) begin
      mem[addr] = din;
    end
  end

  assign dout = (cs & (~rd)) ? mem[addr] : dout;


endmodule
```

（2）测试模块

```
`timescale 10ns/1ns
`include "sram.v"
`define CLOCK_CYCLE 20

module tb_sram;

  wire [7:0] dout;
  reg [7:0] din, addr;
  reg wr, rd, cs;

  sram sram(dout, din, addr, wr, rd, cs);

  initial begin
    cs = 0; rd = 0; wr = 0; din = 0; addr =0;
    #10 cs = 1; din = 8'b1011_0101;
    #10 wr = 1; addr = 8'b11;
    #10 cs = 0; addr = 8'b10; wr = 0; din = 8'b0001_0001;
    #10 cs = 1; wr = 1; rd = 1;
    #10 rd = 0;
    #10 addr = 3;
  end

  initial begin
    $monitor("Current time: %tns", $time, "<---->dout = %b, din = %b, addr = %b, wr = %b, rd = %b, cs = %b",
             dout, din, addr, wr, rd, cs);
  end

endmodule
```

（3）测试波形图



（4）显示输出

```
VSIM 91> run
# Current time:                 0ns<---->dout = xxxxxxxx, din = 00000000, addr = 00000000, wr = 0, rd = 0, cs = 0
# Current time:               100ns<---->dout = xxxxxxxx, din = 10110101, addr = 00000000, wr = 0, rd = 0, cs = 1
# Current time:               200ns<---->dout = 10110101, din = 10110101, addr = 00000011, wr = 1, rd = 0, cs = 1
# Current time:               300ns<---->dout = 10110101, din = 00010001, addr = 00000010, wr = 0, rd = 0, cs = 0
# Current time:               400ns<---->dout = 10110101, din = 00010001, addr = 00000010, wr = 1, rd = 1, cs = 1
# Current time:               500ns<---->dout = 00010001, din = 00010001, addr = 00000010, wr = 1, rd = 0, cs = 1
# Current time:               600ns<---->dout = 10110101, din = 00010001, addr = 00000011, wr = 1, rd = 0, cs = 1
```

第 15 题

（1）设计模块

```
module seq_detect_1101(output reg flag,
                       input din, clk, rst_n);

    localparam IDLE = 5'b0_0001, A = 5'b0_0010, B = 5'b0_0100,
                 C = 5'b0_1000, D = 5'b1_0000;
    reg [4:0] state;

    always @(negedge clk) begin
      if (!rst_n) begin
        flag <= 1'b0;
        state <= IDLE;
      end
      else begin
        flag <= (state == D)? 1'b1 : 1'b0;
        case (state)
          IDLE:   state <= (din)? A : IDLE;
          A:      state <= (din)? B : IDLE;
          B:      state <= (din)? B : C;
          C:      state <= (din)? D : IDLE;
          D:      state <= (din)? B : IDLE;
          default: state <= IDLE;
        endcase
      end
    end

endmodule
```

```
module seq_detect_0110(output reg flag,
                       input din, clk, rst_n);

    localparam IDLE = 5'b0_0001, A = 5'b0_0010, B = 5'b0_0100,
                 C = 5'b0_1000, D = 5'b1_0000;
    reg [4:0] state;

    always @(negedge clk) begin
      if (!rst_n) begin
        flag <= 1'b0;
        state <= IDLE;
      end
      else begin
        flag <= (state == D)? 1'b1: 1'b0;
        case (state)
          IDLE:   state <= (din)? IDLE : A;
          A:      state <= (din)? B : A;
          B:      state <= (din)? C : A;
          C:      state <= (din)? IDLE : D;
          D:      state <= (din)? B : A;
          default: state <= IDLE;
        endcase
      end
    end

endmodule
```

```
module seq_detect(output reg flag,
                  input din, clk, rst_n);

    wire flag1, flag2;
    seq_detect_1101 detect1(flag1, din, clk, rst_n);
    seq_detect_0110 detect2(flag2, din, clk, rst_n);

    always @(*) begin
      flag = flag1 | flag2;
    end
endmodule
```

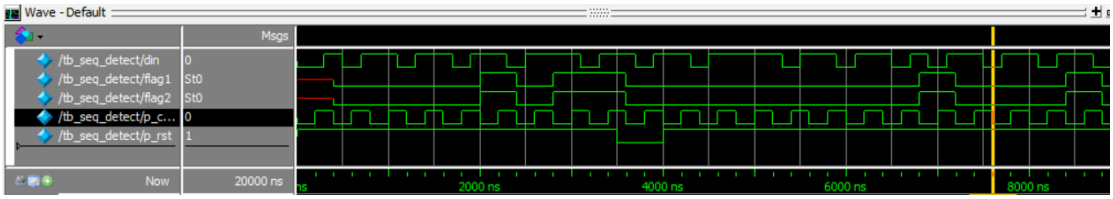
(将两个状态机合并为一个)

```
module seq_detect2(output reg flag,
                   input din, clk, rst_n);

    localparam IDLE = 9'b0_0000_0001, A = 9'b0_0000_0010, B = 9'b0_0000_0100,
               C = 9'b0_0000_1000, D = 9'b0_0001_0000, E = 9'b0_0010_0000,
               F = 9'b0_0100_0000, G = 9'b0_1000_0000, H = 9'b1_0000_0000;
    reg [8:0] state;

    always @(negedge clk) begin
      if (!rst_n) begin
        flag <= 1'b0;
        state <= IDLE;
      end
      else begin
        flag <= ((state == D) | (state == H))? 1'b1 : 1'b0;
        case (state)
          IDLE:   state <= (din)? A : E;
          A:      state <= (din)? B : E;
          B:      state <= (din)? B : C;
          C:      state <= (din)? D : E;
          D:      state <= (din)? G : E;
          E:      state <= (din)? F : E;
          F:      state <= (din)? G : E;
          G:      state <= (din)? B : H;
          H:      state <= (din)? D : E;
          default: state <= IDLE;
        endcase
      end
    end

endmodule
```

（2）测试模块

```
`timescale 10ns/1ns
`include "seq_detect.v"
`define CLOCK_CYCLE 20

module tb_seq_detect;

  wire flag;
  reg p_clk_in, p_rst, din;

  seq_detect m_seq_detect(flag, din, p_clk_in, p_rst);

  initial begin
    din = 0;
    #30 din = 1;#20 din = 0;#20 din = 1;
    #20 din = 1;#20 din = 0;#20 din = 1;
    #20 din = 1;#20 din = 0;#20 din = 1;
    #20 din = 1;#20 din = 0;#20 din = 1;
    #20 din = 1;#20 din = 1;#20 din = 0;
    #20 din = 1;#20 din = 1;#20 din = 0;
    #20 din = 1;#20 din = 1;#20 din = 0;
    #20 din = 1;#20 din = 1;#20 din = 1;
    #20 din = 1;#20 din = 0;#20 din = 1;
    #20 din = 1;#20 din = 0;#20 din = 1;
    #20 din = 1;#20 din = 0;#20 din = 1;
  end

  initial begin
    p_clk_in = 1'b0;
    forever #`CLOCK_CYCLE p_clk_in = ~p_clk_in;
  end

  initial begin
    p_rst = 1;
    #350 p_rst = 0;
    #50 p_rst = 1;
  end

  initial begin
    $monitor("Current time: %tns", $time, "<---->clk = %b, reset = %b, din = %b, flag = %b, flag1 = %b, flag2 = %b",
             p_clk_in, p_rst, din, flag, m_seq_detect.flag1, m_seq_detect.flag2);
  end

```

（3）测试波形图



（4）显示输出

```
# Current time:                600ns<---->clk = 1, reset = 1, din = 0, flag1 = 0, flag2 = 0, state = 000000010
# Current time:                700ns<---->clk = 1, reset = 1, din = 1, flag1 = 0, flag2 = 0, state = 000000010
# Current time:                800ns<---->clk = 0, reset = 1, din = 1, flag1 = 0, flag2 = 0, state = 000000100
# Current time:               1000ns<---->clk = 1, reset = 1, din = 1, flag1 = 0, flag2 = 0, state = 000000100
# Current time:               1100ns<---->clk = 1, reset = 1, din = 0, flag1 = 0, flag2 = 0, state = 000000100
# Current time:               1200ns<---->clk = 0, reset = 1, din = 0, flag1 = 0, flag2 = 0, state = 000001000
# Current time:               1300ns<---->clk = 0, reset = 1, din = 1, flag1 = 0, flag2 = 0, state = 000001000
# Current time:               1400ns<---->clk = 1, reset = 1, din = 1, flag1 = 0, flag2 = 0, state = 000001000
# Current time:               1600ns<---->clk = 0, reset = 1, din = 1, flag1 = 0, flag2 = 0, state = 000010000
# Current time:               1700ns<---->clk = 0, reset = 1, din = 0, flag1 = 0, flag2 = 0, state = 000010000
# Current time:               1800ns<---->clk = 1, reset = 1, din = 0, flag1 = 0, flag2 = 0, state = 000010000
# Current time:               1900ns<---->clk = 1, reset = 1, din = 1, flag1 = 0, flag2 = 0, state = 000010000
# Current time:               2000ns<---->clk = 0, reset = 1, din = 1, flag1 = 1, flag2 = 1, state = 010000000
# Current time:               2200ns<---->clk = 1, reset = 1, din = 1, flag1 = 1, flag2 = 1, state = 010000000
# Current time:               2300ns<---->clk = 1, reset = 1, din = 0, flag1 = 1, flag2 = 1, state = 010000000
# Current time:               2400ns<---->clk = 0, reset = 1, din = 0, flag1 = 0, flag2 = 0, state = 100000000
# Current time:               2500ns<---->clk = 0, reset = 1, din = 1, flag1 = 0, flag2 = 0, state = 100000000
# Current time:               2600ns<---->clk = 1, reset = 1, din = 1, flag1 = 0, flag2 = 0, state = 100000000
# Current time:               2800ns<---->clk = 0, reset = 1, din = 1, flag1 = 1, flag2 = 1, state = 000010000
```

（5）设计说明

本题先采用了 moore 机设计方式，设计了两个序列探测器，后将二者的结果合并。另一种设计方法为只设计一个状态机，直接输出探测结果（moore 机，状态数目为 4+4+1=9）。对比测试波形图以及输出结果可知二种设计结果完全相同。

第 16 题

（1）设计模块

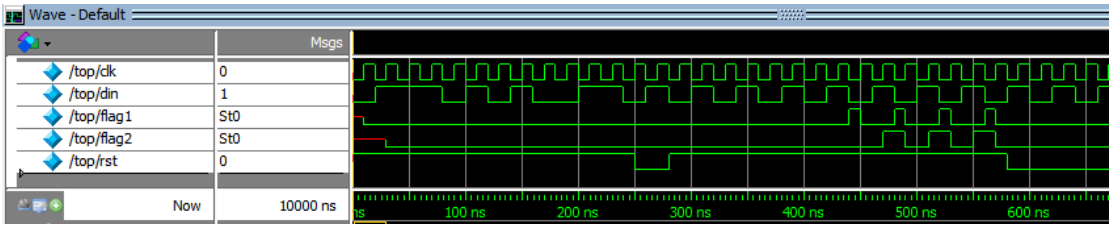
```
module mealy(output flag,
             input din, clk, rst);

    localparam IDLE = 8'b0000_0001, A = 8'b0000_0010, B = 8'b0000_0100,
                  C = 8'b0000_1000, D = 8'b0001_0000, E = 8'b0010_0000,
                  F = 8'b0100_0000, G = 8'b1000_0000;
    reg [7:0] p_state, n_state;

    always @(posedge clk or negedge rst) begin
      if (!rst) p_state <= IDLE;
      else p_state <= n_state;
    end

    always @(*) begin
      case (p_state)
          IDLE:   n_state = (din)? A : IDLE;
          A:      n_state = (din)? A : B;
          B:      n_state = (din)? C : IDLE;
          C:      n_state = (din)? A : D;
          D:      n_state = (din)? E : IDLE;
          E:      n_state = (din)? A : F;
          F:      n_state = (din)? G : IDLE;
          G:      n_state = (din)? A : F;
          default: n_state = IDLE;
      endcase
    end

    assign flag = ((p_state == G) && (din == 1'b0)) ? 1'b1 : 1'b0;
endmodule
```

```
module moore(output reg flag,
             input din, clk, rst);


    localparam IDLE = 9'b0000_0000_1, A = 9'b0000_0001_0, B = 9'b0000_0010_0,
                  C = 9'b0000_0100_0, D = 9'b0000_1000_0, E = 9'b0001_0000_0,
                  F = 9'b0010_0000_0, G = 9'b0100_0000_0, H = 9'b1000_0000_0;
    reg [8:0] state;



    always @(posedge clk or negedge rst) begin
      if (!rst) begin
        flag <= 1'b0;
        state <= IDLE;
      end
      else begin
        flag <= (state == H)? 1'b1: 1'b0;
        case (state)
          IDLE:   state <= (din)? A : IDLE;
          A:      state <= (din)? A : B;
          B:      state <= (din)? C : IDLE;
          C:      state <= (din)? A : D;
          D:      state <= (din)? E : IDLE;
          E:      state <= (din)? A : F;
          F:      state <= (din)? G : IDLE;
          G:      state <= (din)? A : H;
          H:      state <= (din)? G : IDLE;
          default: state <= IDLE;
        endcase
      end
    end

endmodule
```

（2）测试模块

```
`timescale 10ns/1ns
`include "mealy.v"
`include "moore.v"
`define CLOCK_CYCLE 1

module top;
  wire flag1, flag2;
  reg din, clk, rst;

  mealy m_mealy(flag1, din, clk, rst);
  moore m_moore(flag2, din, clk, rst);

  initial begin
    clk = 1'b0;
    forever #`CLOCK_CYCLE clk = ~clk;
  end

  initial begin
    din = 0;
    #2 din = 1;#2 din = 1;#2 din = 1;#2 din = 0;
    #2 din = 1;#2 din = 0;#2 din = 1;#2 din = 0;
    #2 din = 0;#2 din = 1;#2 din = 1;#2 din = 0;
    #2 din = 1;#2 din = 0;#2 din = 1;#2 din = 0;
    #2 din = 1;#2 din = 0;#2 din = 1;#2 din = 0;
    #2 din = 1;#2 din = 0;#2 din = 1;#2 din = 0;
    #2 din = 1;#2 din = 0;#2 din = 1;#2 din = 0;
    #2 din = 1;
  end

  initial begin
    rst = 1'b1;
    #25 rst = 1'b0;
    #3 rst = 1'b1;
    #30 rst = 1'b0;
  end

  initial begin
    $monitor("Current time: %tns", $time, "<----> flag1 = %b, flag2 = %b, din = %b, clk = %b, rst = %b",
             flag1, flag2, din, clk, rst);
  end

endmodule
```

（3）测试波形图



（4）显示输出

```
VSIM 103> run
# Current time:                  0ns<----> flag1 = x, flag2 = x, din = 0, clk = 0, rst = 1
# Current time:                 10ns<----> flag1 = 0, flag2 = x, din = 0, clk = 1, rst = 1
# Current time:                 20ns<----> flag1 = 0, flag2 = x, din = 1, clk = 0, rst = 1
# Current time:                 30ns<----> flag1 = 0, flag2 = 0, din = 1, clk = 1, rst = 1
# Current time:                 40ns<----> flag1 = 0, flag2 = 0, din = 1, clk = 0, rst = 1
# Current time:                 50ns<----> flag1 = 0, flag2 = 0, din = 1, clk = 1, rst = 1
# Current time:                 60ns<----> flag1 = 0, flag2 = 0, din = 1, clk = 0, rst = 1
# Current time:                 70ns<----> flag1 = 0, flag2 = 0, din = 1, clk = 1, rst = 1
# Current time:                 80ns<----> flag1 = 0, flag2 = 0, din = 0, clk = 0, rst = 1
# Current time:                 90ns<----> flag1 = 0, flag2 = 0, din = 0, clk = 1, rst = 1
```

（5）设计说明

由仿真结果可知，moore 机和 mealy 机在序列探测问题上的不同，包括两个方面。第一，moore 机状态比 mealy 机状态多一，等于序列长度加一。第二，moore 机比 mealy 机滞后一个周期。

由波形图可知，二种设计方式均达到了设计目的，能实现功能，但波形有差别。这是由于 mealy 机采用了 assign 语句计算 flag 使得结果无需等待时钟边沿到来。(若将 assign 语句换为过程语句放入 always 中，则波形高电平长度相同）另外，图中 moore 机比 mealy 少一个高电平，这是由于最后引入复位的缘故（而 mealy 机比 moore 早一个周期，故不会受到影响）。