Verilog HDL程序设计与仿真 实验报告

专业班级： **通信2101班**

姓名： **罗畅**

学号： U202113940

各芯片代码与测试用例

CD4532

CD4532.v：

```verilog
/*
@author Luo Chang
@UID U202113940
@date 2022/11/23
*/

module CD4532(EI,I,Y,GS,EO);
/*
输入端口：EI使能信号，I为输入的数。
输出端口：Y为对应的二进制数。
GS,EO用于多片的级联：
当输入信号存在有效信号，GS=1,EO=0；
当输入信号全部无效时，GS=0,EO=1；??????；
*/
input EI;
input [7:0] I;
output reg [2:0] Y;
output reg GS,EO;

always @(*)
    begin
        if (EI==0)
            begin
                Y[2:0] = 3'b000;

                GS = 0;
                EO = 0;
            end
        else
            begin
                GS = 1;
                EO = 0;
                casex(I[7:0])
                    8'b1xxx_xxxx : Y[2:0] = 3'b111;
                    8'b01xx_xxxx : Y[2:0] = 3'b110;
                    8'b001x_xxxx : Y[2:0] = 3'b101;
                    8'b0001_xxxx : Y[2:0] = 3'b100;
                    8'b0000_1xxx : Y[2:0] = 3'b011;
                    8'b0000_01xx : Y[2:0] = 3'b010;
                    8'b0000_001x : Y[2:0] = 3'b001;
                    8'b0000_0001 : Y[2:0] = 3'b000;
                    default
                        begin
                            GS = 0;
                            EO = 1;
                            Y[2:0] = 3'b000;
                        end
                endcase
            end
```

```
        end
endmodule
```

Cd4532_tb.v:

```verilog
/*
@author Luo Chang
@UID U202113940
@date 2022/11/23
*/

`timescale 1ns/1ns
module Cd4532_tb;
reg [7:0] I;
reg EI;
wire [2:0] Y;
wire GS,EO;

CD4532 U0(EI,I,Y,GS,EO);

initial $monitor($time,":tI = %b,EI = %b,EO = %b,GS =
%b,Y = %b\n",I,EI,EO,GS,Y);

initial
    begin
//使能信号无效
        EI = 0;
        I=8'b1111_1111;
        #10
        EI = 0;
        I=8'b0111_1111;
        #10
//使能信号有效
        EI = 1;
        I=8'b1111_1111;
        #10
        EI = 1;
        I=8'b0111_1111;
        #10
        EI = 1;
        I=8'b0001_1111;
//停止仿真
        $stop;
    end
endmodule
```
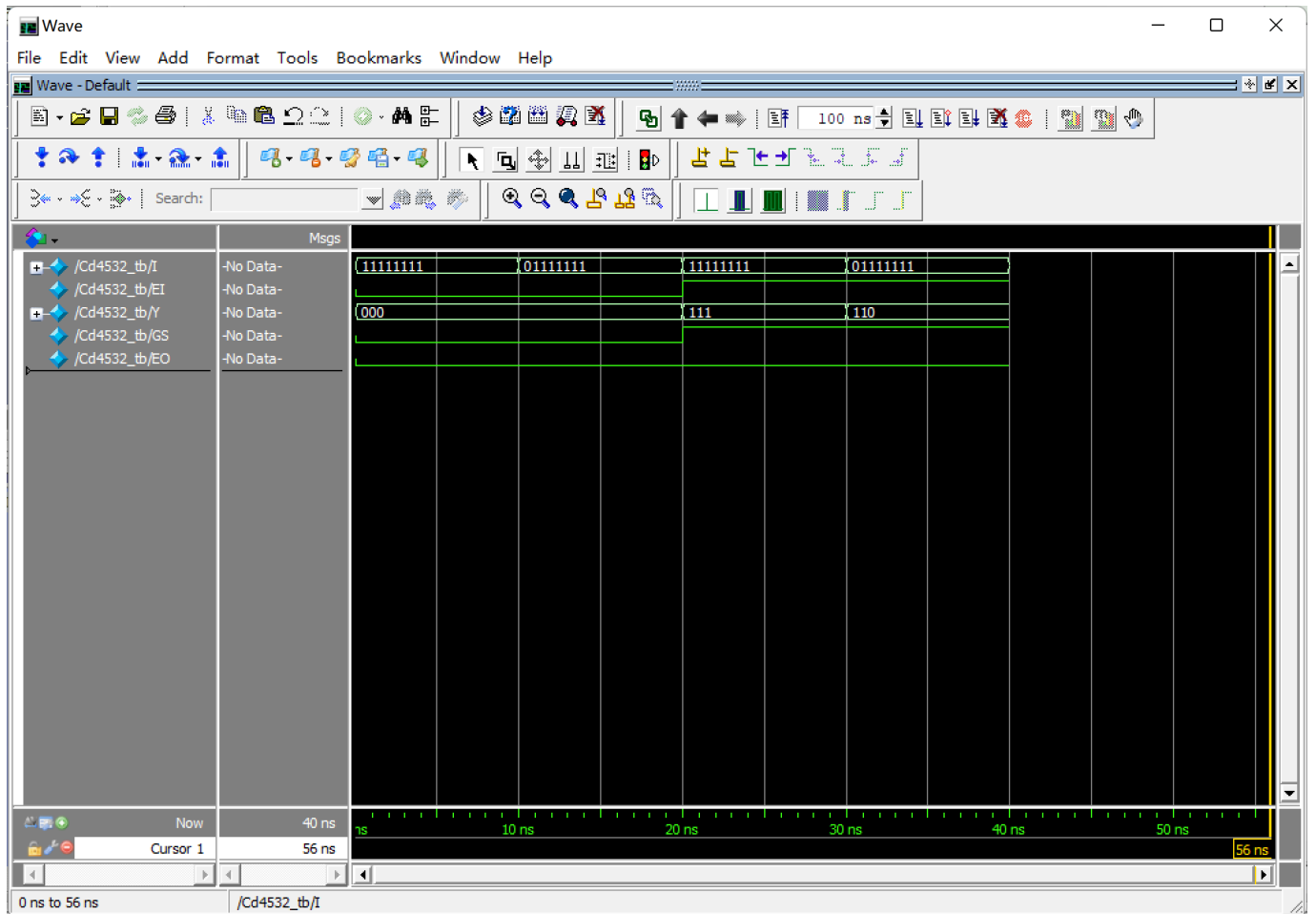
测试波形

74X138

74X138.v

```verilog
/*
@author Luo Chang
@UID U202113940
@date 2022/11/23
*/

module _74X138(E1,E2,E3,A,Y);
input E1,E2,E3;
input [2:0] A;
output reg [7:0] Y;

always @(*)
    begin
//The input signal of the enable terminal is valid
        if (E1 == 0 && E2 == 0 && E3 == 1)
            begin
                casex(A[2:0])
                    3'b000 : Y = 8'b0111_1111;
                    3'b001 : Y = 8'b1011_1111;
                    3'b010 : Y = 8'b1101_1111;
                    3'b011 : Y = 8'b1110_1111;
                    3'b100 : Y = 8'b1111_0111;
                    3'b101 : Y = 8'b1111_1011;
                    3'b110 : Y = 8'b1111_1101;
                    3'b111 : Y = 8'b1111_1110;
                endcase
            end
//The input signal of the enable terminal is invalid
        else
            Y = 8'b1111_1111;
    end
endmodule
```

74X138_tb.v:

```
/*
@author Luo Chang
@UID U202113940
@date 2022/11/23
*/

`timescale 1ns/1ns
module _74X138_tb;
reg [2:0] A;
reg E1,E2,E3;
wire [7:0] Y;

_74X138 U0(E1,E2,E3,A,Y);

initial $monitor($time,":A = %b,E1 = %b,E2 = %b,E3 = %b,Y = %b\n",A,E1,E2,E3,Y);

initial
    begin
//The input signal of the enable terminal is invalid
        E1 = 0;
        E2 = 1;
        E3 = 0;
        A=3'b010;
        #10
        E1 = 1;
        E2 = 0;
        A=3'b110;
        #10
//The input signal of the enable terminal is valid
        E1 = 0;
        E2 = 0;
        E3 = 1;
        A=3'b101;
        #10
        E1 = 0;
        E2 = 0;
        E3 = 1;
        A=3'b000;
        #10
        E1 = 0;
        E2 = 0;
        E3 = 1;
        A=3'b111;
        $stop;
    end
endmodule
```
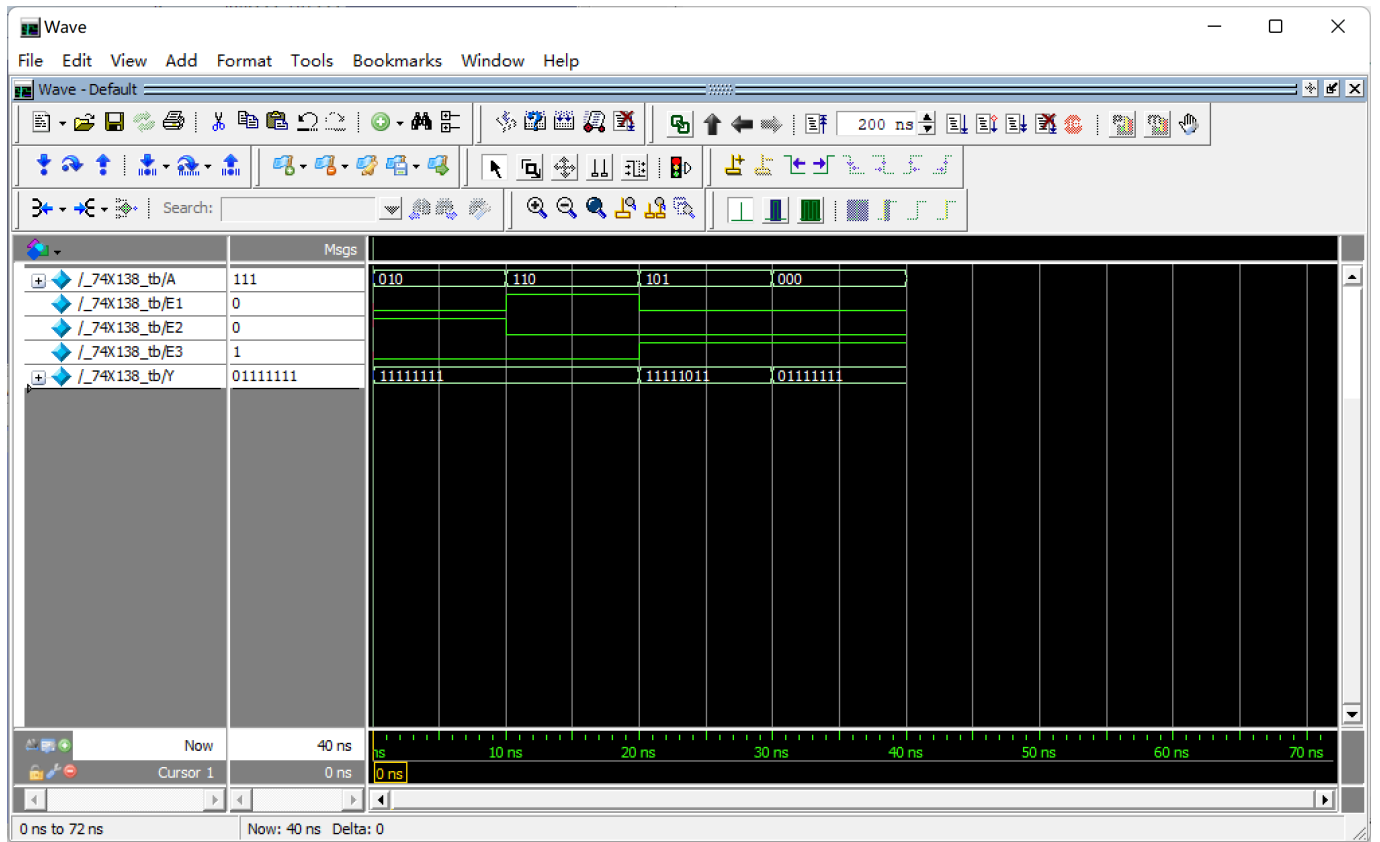
测试波形

74HC4511

_74HC4511.v

```verilog
/*
@author Luo Chang
@UID U202113940
@date 2022/11/24
*/

// 74HCT4511 is a BCD to 7-segment latch/decoder/driver

module _74HC4511(LE,BL,LT,D,Q);
input LE,BL,LT;
input [3:0] D;
output reg [6:0] Q;

always @(*)
    begin
        if (LE == 0 && !BL == 1 && !LT == 1)
            begin
                casex(D[3:0])
                    4'b0000 : Q = 7'b1111_110;
                    4'b0001 : Q = 7'b0110_000;
                    4'b0010 : Q = 7'b1101_101;
                    4'b0011 : Q = 7'b1111_001;
                    4'b0100 : Q = 7'b0110_011;
                    4'b0101 : Q = 7'b1011_011;
                    4'b0110 : Q = 7'b0011_111;
                    4'b0111 : Q = 7'b1110_000;
                    4'b1000 : Q = 7'b1111_111;
                    4'b1001 : Q = 7'b1111_011;
                    default : Q = 7'b0000_000;
                endcase
            end
        else if (!LT == 0)
            Q = 7'b1111_111;
        else if (!BL == 0 && !LT == 1)
            Q = 7'b0000_000;
        else
            Q <= Q;
    end
endmodule
```

_74HC4511_tb.v

```verilog
/*
@author Luo Chang
@UID U202113940
@date 2022/11/24
*/

`timescale 1ns/1ns

module _74HC4511_tb;

reg [3:0] D;
reg LE,BL,LT;
wire [6:0] Q;

_74HC4511 U0(LE,BL,LT,D,Q);

initial $monitor($time,"D = %b,Q = %b,LE = %b,BL = %b,LT = %b",D,Q,LE,BL,LT);

initial
    begin
// 74HC4511 on test
        LE = 1;
        BL = 0;
        LT = 1;
        D = 4'b1101;
        #10
// 74HC4511 turn off
        LE = 0;
        BL = 1;
        LT = 0;
        D = 4'b1011;
        #10
// 74HC4511 turn on
        LE = 0;
        BL = 0;
        LT = 0;
        D = 4'b0000;
        #10
        D = 4'b1001;
        #10
        D = 4'b0110;
        #10
        D = 4'b1010;
        #10
// 74HC4511 latched
        LE = 1;
        BL = 0;
        LT = 0;
        #15
        $stop;
```
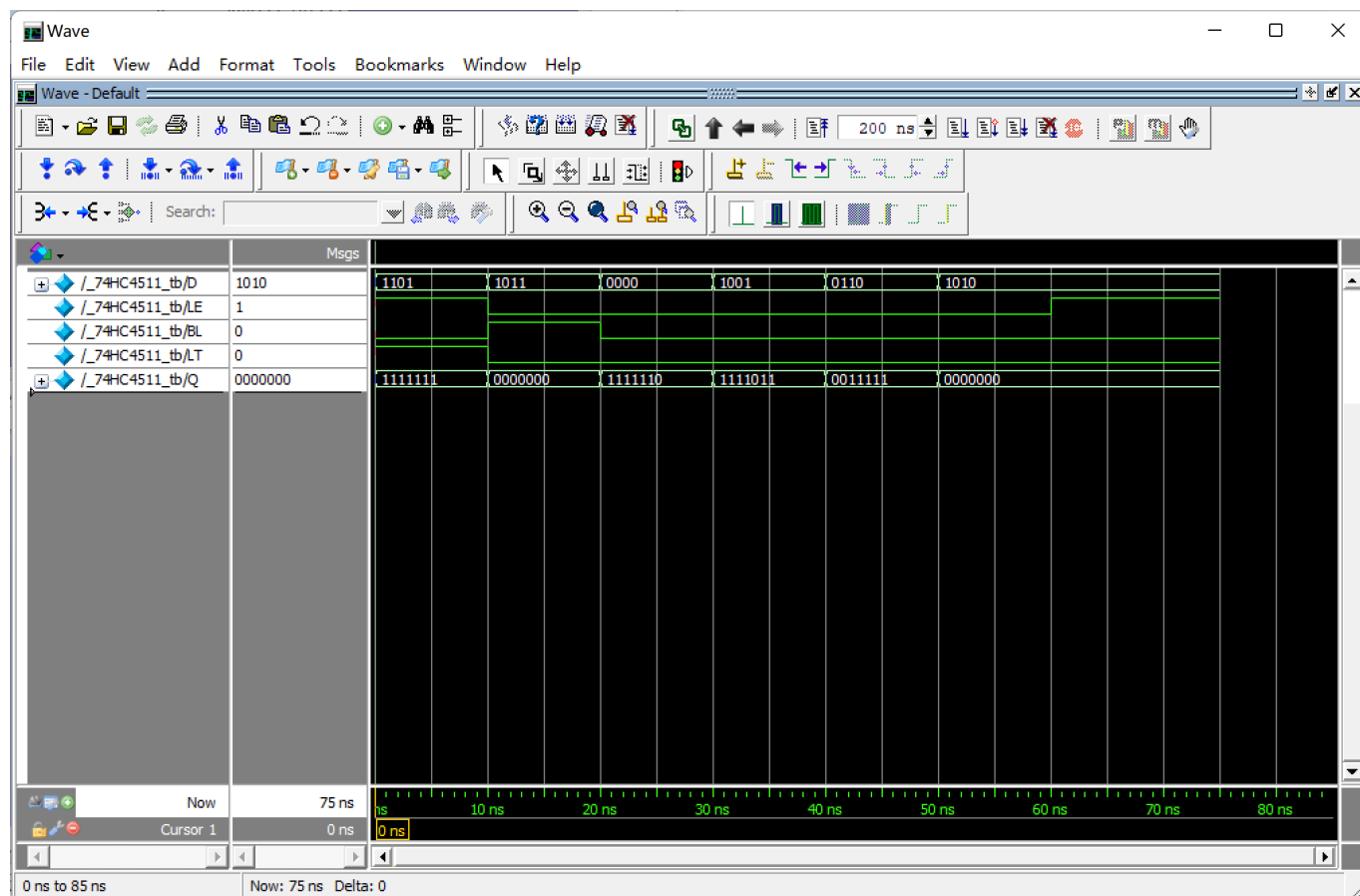
```
        end
    endmodule
```

测试波形

74HC151

_74HC151.v

```verilog
/*
@author Luo Chang
@UID U202113940
@date 2022/11/25
*/

module _74HC151(D,S,E,Y,Y_);
input E;
input [7:0] D;
input [2:0] S;
output reg Y,Y_;
always @(*)
    begin
        if (! E ==1)
            begin
                Y = 0;
                Y_ = 1;
            end
        else
            begin
                casex (S[2:0])
                    3'b000 : Y = D[0];
                    3'b001 : Y = D[1];
                    3'b010 : Y = D[2];
                    3'b011 : Y = D[3];
                    3'b100 : Y = D[4];
                    3'b101 : Y = D[5];
                    3'b110 : Y = D[6];
                    3'b111 : Y = D[7];
                endcase
                Y_ = ~Y;
            end
    end
endmodule
```

_74HC151_tb.v

```
/*
@author Luo Chang
@UID U202113940
@date 2022/11/24
*/

`timescale 1ns/1ns

module _74HC151_tb;

reg [2:0] S;
reg E;
reg [7:0] D;

wire Y,Y_;

_74HC151 U0(D,S,E,Y,Y_);

initial $monitor($time,"E = %b,S = %b,D = %b,Y = %b,Y_ = %b",E,S,D,Y,Y_);

initial
    begin
// enabled terminal invalid
        E = 0;
        D = 8'b1101_1110;
        S = 3'b101;
        #10
        D = 8'b1001_0110;
        S = 3'b111;
        #10
// enabled terminal valid
        E = 1;
        D = 8'b0000_0001;
        S = 3'b000;
        #10
        D = 8'b1110_1111;
        S = 3'b100;
        #10
        D = 8'b0111_1111;
        S = 3'b111;
        #10
        D = 8'b0000_0010;
        S = 3'b001;
        #10
        $stop;
    end
endmodule
```
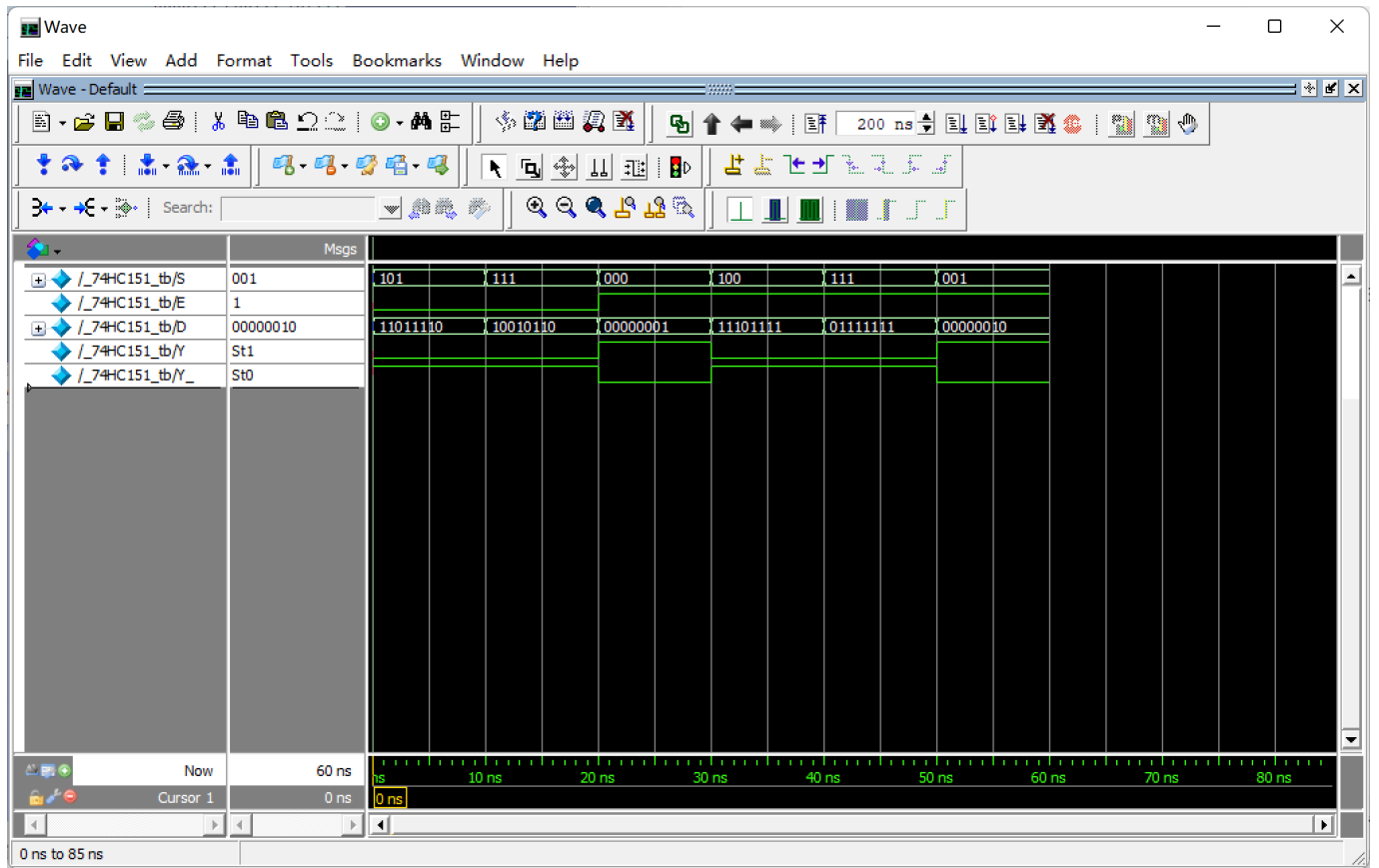
测试波形

74HC85

_74HC85.v

```
/*
@author Luo Chang
@UID U202113940
@date 2022/11/25
*/

module _74HC85(A,B,I_g,I_s,I_e,F_g,F_s,F_e);
// A, B are the signals we need to compare
input [3:0] A,B;
/*
I_g means I_greater(A>B), I_s means I_smaller(A<B), I_e means I_equal(A==B).
They're used for cascading
*/
input I_g,I_s,I_e;
// F are similar with I
output reg F_g,F_s,F_e;

always @(*)
    begin
// compare highest bit
        if (A[3] > B[3])
            begin
                F_g = 1;
                F_s = 0;
                F_e = 0;
            end
        else if (A[3] < B[3])
            begin
                F_g = 0;
                F_s = 1;
                F_e = 0;
            end
        else
// compare 2nd bit
            if (A[2] > B[2])
                begin
                    F_g = 1;
                    F_s = 0;
                    F_e = 0;
                end
            else if (A[2] < B[2])
                begin
                    F_g = 0;
                    F_s = 1;
                    F_e = 0;
                end
            else
// compare 3rd bit
                if (A[1] > B[1])
                    begin
```

```verilog
                    F_g = 1;
                    F_s = 0;
                    F_e = 0;
                end
        else if (A[1] < B[1])
            begin
                F_g = 0;
                F_s = 1;
                F_e = 0;
            end
        else
// compare lowest bit
            if (A[0] > B[0])
                begin
                    F_g = 1;
                    F_s = 0;
                    F_e = 0;
                end
            else if (A[0] < B[0])
                begin
                    F_g = 0;
                    F_s = 1;
                    F_e = 0;
                end
            else
// 4 bits are all the same, compare input I
                if (I_e == 1)
// I_e
                    begin
                        F_g = 0;
                        F_s = 0;
                        F_e = 1;
                    end
// I_g valid
                else if (I_g == 1 && I_s == 0)
                    begin
                        F_g = 1;
                        F_s = 0;
                        F_e = 0;
                    end
// I_s valid
                else if (I_g == 0 && I_s == 1)
                    begin
                        F_g = 0;
                        F_s = 1;
                        F_e = 0;
                    end
// invalid I input
                else if (I_g == I_s)
                    begin
                        F_g = ~I_g;
```

```verilog
                    F_s = ~I_s;
                    F_e = 0;
                end

    end
endmodule
```

_74HC85_tb.v

```verilog
/*
@author Luo Chang
@UID U202113940
@date 2022/11/25
*/

`timescale 1ns/1ns

module _74HC85_tb;

reg [3:0] A,B;
reg I_g,I_s,I_e;

wire F_g,F_s,F_e;

_74HC85 U0(A,B,I_g,I_s,I_e,F_g,F_s,F_e);

initial $monitor($time,"A = %b,B = %b,I_g = %b,I_s = %b,I_e = %b,F_g = %b,F_s = %b,F_e
= %b",A,B,I_g,I_s,I_e,F_g,F_s,F_e);

initial
    begin
// I invalid
        A = 4'b1000;
        B = 4'b0100;
          #10
            A = 4'b0100;
        B = 4'b1000;
        #10
        A = 4'b0110;
        B = 4'b0111;
        #10
// I valid(A == B)
        A = 4'b1101;
        B = 4'b1101;
        I_g = 0;
        I_s = 0;
        I_e = 1;
        #10
        A = 4'b1001;
        B = 4'b1001;
        I_g = 1;
        I_s = 0;
        I_e = 0;
        #10
        A = 4'b0001;
        B = 4'b0001;
        I_g = 0;
        I_s = 1;
        I_e = 0;
```
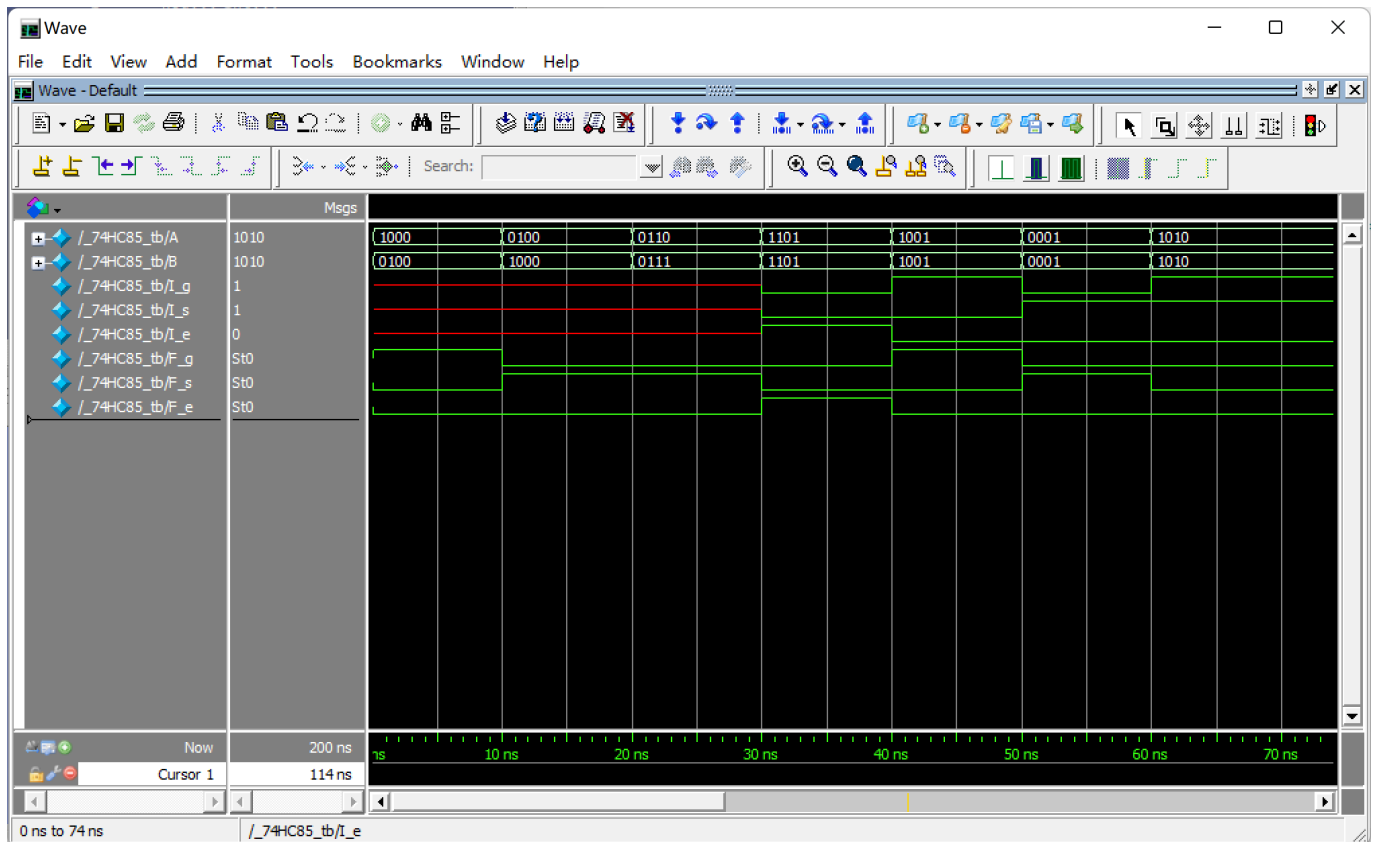
```
        #10
        A = 4'b1010;
        B = 4'b1010;
        I_g = 1;
        I_s = 1;
        I_e = 0;
    end
endmodule
```

测试波形

74HC283

_74HC283.v

```verilog
/*
@author Luo Chang
@UID U202113940
@date 2022/11/26
*/

module _74HC283(C_IN,A,B,C_OUT,S);
input C_IN;
input [3:0] A,B;
output reg C_OUT;
output reg [3:0] S;

wire [3:0] X,Y,C;

assign X = A ^ B;
assign Y = A & B;
assign C[0] = Y[0] | (X[0] & C_IN);

genvar i;
for (i = 1;i < 4;i = i + 1)
    assign C[i] = Y[i] | (X[i] & C[i - 1]);
assign C_OUT = C[3];

assign S[0] = X[0] ^ C_IN;
for (i = 1;i < 4;i = i + 1)
    assign S[i] = X[i] ^ C[i -1];
endmodule
```

_74HC283_tb.v

```verilog
/*
@author Luo Chang
@UID U202113940
@date 2022/11/26
*/


`timescale 1ns/1ns
module _74HC283_tb;

reg C_IN;
reg [3:0] A,B;
wire C_OUT;
wire [3:0] S;

_74HC283 U0(C_IN,A,B,C_OUT,S);

initial $monitor($time,"C_IN = %b,A = %b,B = %b,C_OUT = %b,S = %b,",C_IN,A,B,C_OUT,S);

initial
    begin
// C_IN = 0
        C_IN = 0;
        A = 4'b1101;
        B = 4'b0011;
        #5
        C_IN = 0;
        A = 4'b0000;
        B = 4'b0000;
        #5
        C_IN = 0;
        A = 4'b1111;
        B = 4'b1111;
        #5
// C_IN = 1
        C_IN = 1;
        A = 4'b1011;
        B = 4'b0011;
        #5
        C_IN = 1;
        A = 4'b0000;
        B = 4'b0000;
        #5
// egde test
        C_IN = 1;
        A = 4'b1111;
        B = 4'b1111;
        #5
        $stop;
    end
endmodule
```
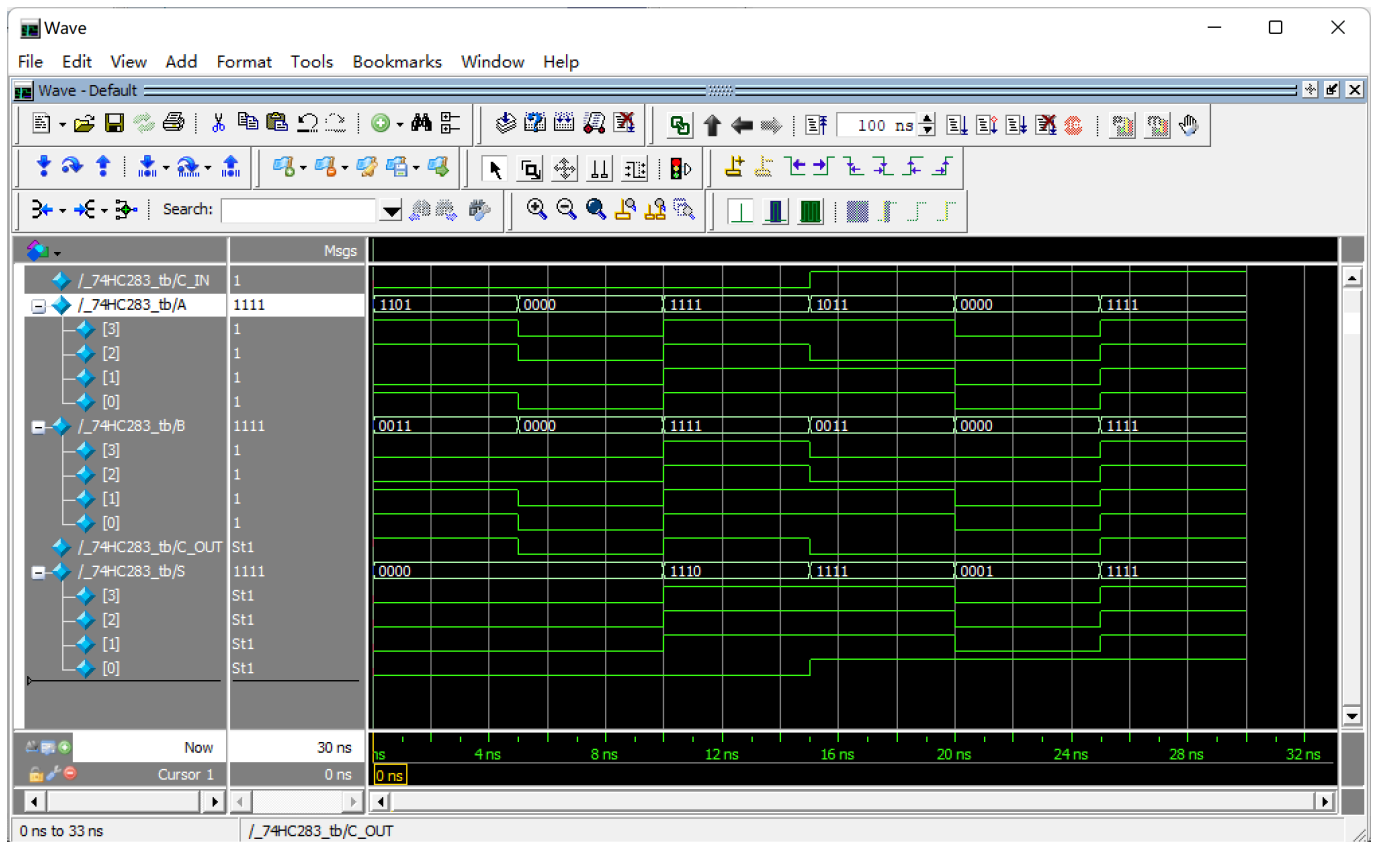
测试波形

74HC194

_74HC194.v

```verilog
/*
@author Luo Chang
@UID U202113940
@date 2022/11/26
*/

module _74HC194(CR,CP,D_SR,D_SL,S,D_I,Q);
/*
CR: asyn clear
CP: clock signal
D_SR: right shift serial data input
D_SL: left shift serial data input
D_I: parallel data input

S: control signal,
when S == 00,output stays;
     S == 01,output right shift;
     S == 10,output left shift;
     S == 11,sync parallel transponse

Q: output signal

sensitive signal: CP's rising edge or CR's droping edge
*/
input CR,CP,D_SR,D_SL;
input [1:0] S;
input [3:0] D_I;
output reg [3:0] Q;

always @(posedge CP or negedge CR)
    begin
        if (CR == 1) Q <= 4'b0000;
        else
            casex (S[1:0])
                2'b00 : Q <= Q;
                2'b01 : Q <= {Q[2:0],D_SR};
                2'b10 : Q <= {D_SL,Q[3:1]};
                2'b11 : Q <= D_I;
            endcase
    end
endmodule
```

_74HC194_tb.v

```
/*
@author Luo Chang
@UID U202113940
@date 2022/11/26
*/

`timescale 1ns/1ns

module _74HC194_tb;
reg CR,CP,D_SR,D_SL;
reg [1:0] S;
reg [3:0] D_I;
wire [3:0] Q;

_74HC194 U0(CR,CP,D_SR,D_SL,S,D_I,Q);

initial $monitor($time,"D_SR = %b,D_SL = %b,S = %b,D = %b,Q = %b",D_SR,D_SL,S,D_I,Q);

initial
    CP = 0;
    always #5 CP = ~CP;

initial
    begin
// clear
        CR = 1;
        S = 2'b11;
        D_I = 4'b1001;
        D_SR = 1;
        D_SL = 0;
        #20
// set as 1111
        CR = 0;
        S = 2'b11;
        D_I = 4'b1111;
        D_SR = 0;
        D_SL = 0;
        #20
// right shift
        CR = 0;
        S = 2'b01;
        D_I = 4'b1111;
        D_SR = 0;
        D_SL = 1;
        #20
// right shift again
        CR = 0;
        S = 2'b01;
        D_I = 4'b1111;
        D_SR = 1;
```

```
            D_SL = 0;
            #20
// clear
            CR = 1;
            S = 2'b11;
            D_I = 4'b1001;
            D_SR = 1;
            D_SL = 0;
            #20
// set as 0000
            CR = 0;
            S = 2'b11;
            D_I = 4'b0000;
            D_SR = 0;
            D_SL = 1;
            #20
// left shift
            CR = 0;
            S = 2'b10;
            D_I = 4'b0000;
            D_SR = 0;
            D_SL = 1;
            #20
// left shift again
            CR = 0;
            S = 2'b10;
            D_I = 4'b1111;
            D_SR = 1;
            D_SL = 0;
            #20
// stay
            CR = 0;
            S = 2'b00;
            D_I = 4'b0000;
            D_SR = 1;
            D_SL = 1;
            #20
            $stop;
        end

endmodule
```
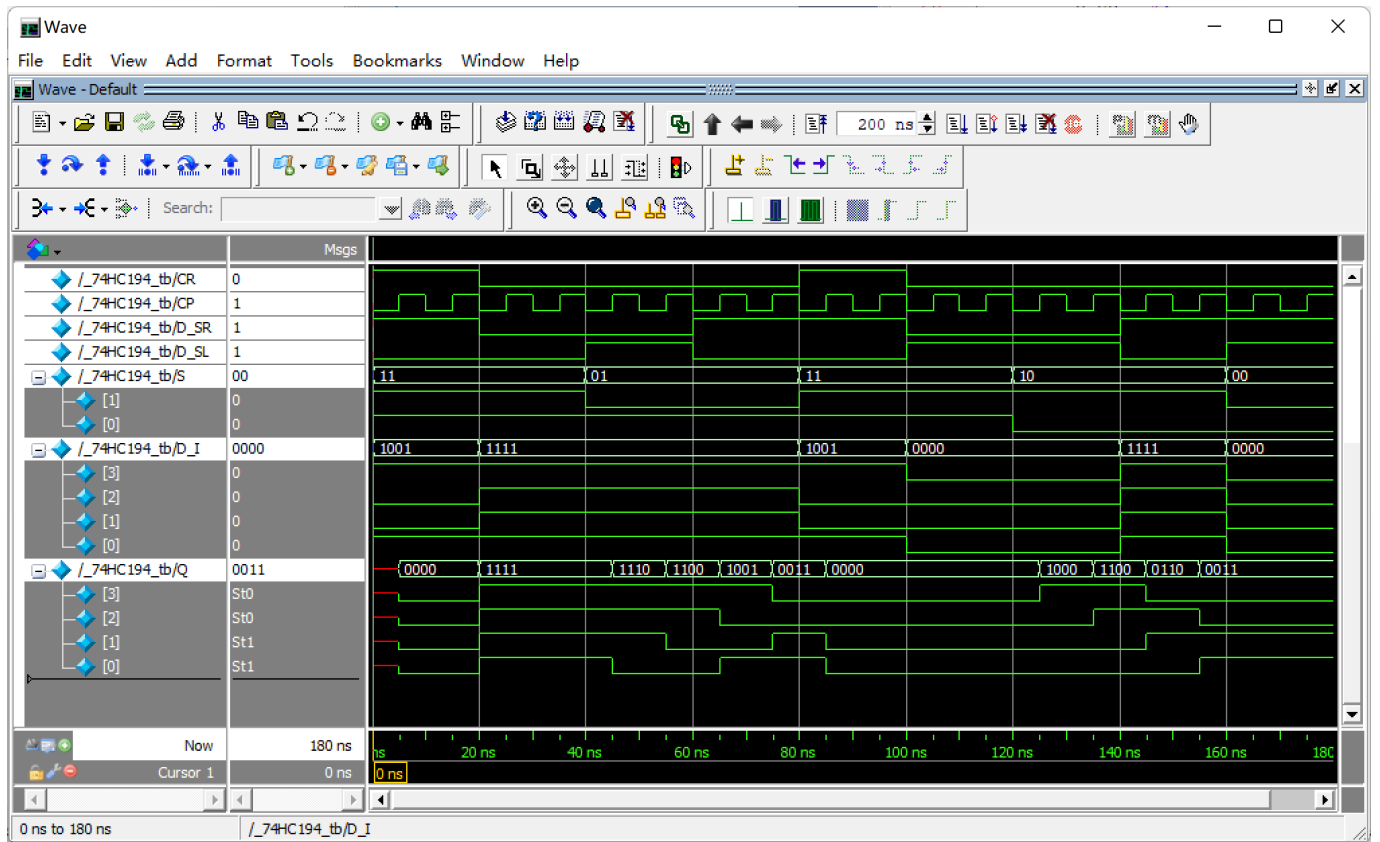
测试波形

74LVC161

_74LVC161.v

```verilog
/*
@author Luo Chang
@UID U202113940
@date 2022/11/26
*/

module _74LVC161(CR,PE,CEP,CET,CP,D,Q,TC);
/*
CR: clear
PE: preset
CEP,CET: enabled terminal
D: preset data
Q: output
TC: carry bit
*/
input CR,PE,CEP,CET,CP;
input [3:0] D;
output reg [3:0] Q;
output reg TC;

always @(posedge CP or negedge CR)
    begin
        if (!CR == 0)
            begin
                Q = 4'b0000;
                TC = 0;
            end
        else
            if (!PE == 0) Q <= D;
            else
                casex({CEP,CET})
                    2'b0x : Q <= Q;
                    2'bx0 :
                        begin
                            Q <= Q;
                            TC = 0;
                        end
                    2'b11 :
                        begin
                            Q = Q + 1;
                            TC = (Q == 4'b1111);
                        end
                endcase
    end
endmodule
```

_74LVC161_tb.v

```verilog
/*
@author Luo Chang
@UID U202113940
@date 2022/11/26
*/


`timescale 1ns/1ns

module _74LVC161_tb;

reg CR,PE,CEP,CET,CP;
reg [3:0] D;
wire [3:0] Q;
wire TC;


_74LVC161 U0(CR,PE,CEP,CET,CP,D,Q,TC);

initial $monitor($time,"CR = %b,PE = %b,CEP = %b,CET = %b,D = %b,Q = %b,TC =
%b",CR,PE,CEP,CET,CP,D,Q,TC);

initial
    CP = 0;
    always #5 CP = ~CP;
initial
    begin
//clear
        CR = 1;
        PE = 1;
        CEP = 0;
        CET = 1;
        D = 4'b1101;
        #20
// preset as 1001
        CR = 0;
        PE = 1;
        CEP = 1;
        CET = 1;
        D = 4'b1001;
        #20
// count
        CR = 0;
        PE = 0;
        CEP = 1;
        CET = 1;
        D = 4'b0000;
        #50
// stay
        CR = 0;
        PE = 0;
        CEP = 1;
```

```
            CET = 0;
            #10
// count
            CR = 0;
            PE = 0;
            CEP = 1;
            CET = 1;
            D = 4'b0000;
            #10
//stay and carry bit
            CR = 0;
            PE = 0;
            CEP = 0;
            CET = 1;
            #15
            $stop;
        end
    endmodule
```
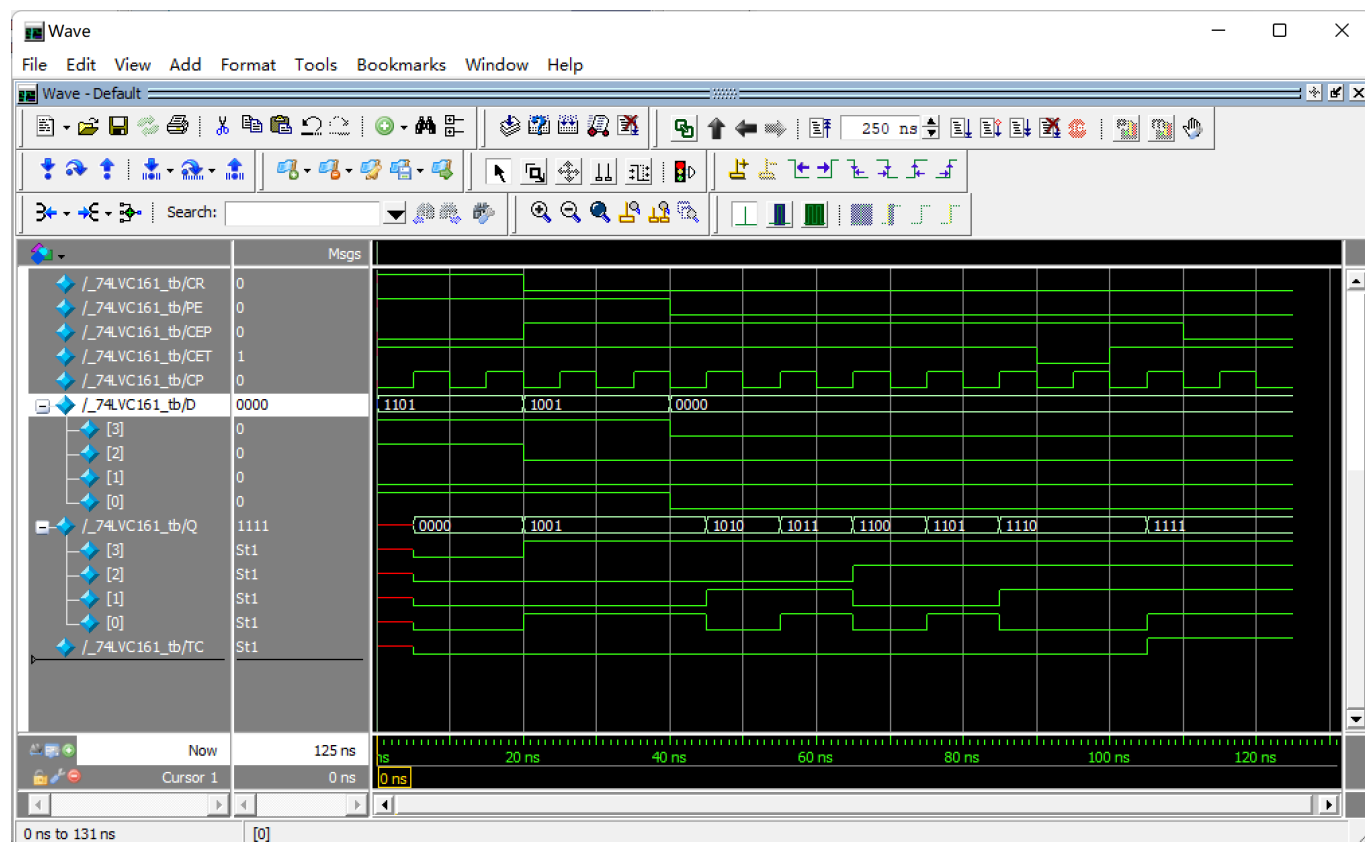
测试波形

扩展功能

74X139和74X138构建5-32译码器

两片74LS151连接成一个16选1数据选择器

篮球24秒计时器

两片74LS151连接成一个16选1数据选择器

篮球24秒计时器