**现代篇实验报告**

班别 软件工程1班 学号 3119005028 座位号 34 姓名 魏耀辉

## 基本门电路

一、实验目的

1、了解基于Verilog的基本门电路的设计及其验证。

2、熟悉利用EDA工具进行设计及仿真的流程。

3、学习针对实际门电路芯片74HC00、74HC02、74HC04、74HC08、74HC32、74HC86进行VerilogHDL设计的方法。

4、熟悉实验箱的使用和程序下载（烧录）及测试的方法。

二、实验环境及仪器

1、Libero仿真软件。

2、数字逻辑与系统设计实验箱。

3、Actel A3P060 FPGA芯片及Flash Pro5烧录器。

三、实验内容

1、掌握Libero软件的使用方法。

2、进行针对74系列基本门电路的设计，并完成相应的仿真实验。

3、参考教材中相应章节的设计代码、测试平台代码（可自行编程），完成74HC00、74HC02、74HC04、74HC08、74HC32、74HC86相应的设计、综合及仿真。

4、提交针对74HC00、74HC02、74HC04、74HC08、74HC32、74HC86的综合结果，以及相应的仿真结果。

四、实验结果和数据处理

1、所有模块及测试平台代码清单

//74HC00代码-与非

module W\_74HC00(A,B,Y);

input [3:0]A,B;

output [3:0]Y;

assign Y=~(A&B);

endmodule

//74HC00测试平台代码

`timescale 1ns/1ns

module test\_74HC00;

reg[3:0]a,b;

wire[3:0]y;

W\_74HC00 u1(a,b,y);

initial

begin

a=4'b0000;b=4'b0001;

#10 b=b<<1;

#10 b=b<<1;

#10 b=b<<1;

a=4'b1111;b=4'b0001;

#10 b=b<<1;

#10 b=b<<1;

#10 b=b<<1;

end

endmodule

//74HC02代码-或非

module W\_74HC02(A,B,Y);

input[3:0]A,B;

output[3:0]Y;

assign Y=~(A|B);

endmodule

//74HC02测试平台代码

`timescale 1ns/1ns

module test\_74HC02;

reg[3:0]a,b,c;

wire[3:0]y;

W\_74HC02 u1(a,b,y);

initial

begin

a=4'b1111;c=4'b0001;

b=~c;c=c<<1;

#10 b=~c;c=c<<1;

#10 b=~c;c=c<<1;

#10 b=~c;

a=4'b0000;c=4'b0001;

b=~c;c=c<<1;

#10 b=~c;c=c<<1;

#10 b=~c;c=c<<1;

#10 b=~c;

end

endmodule

//74HC04代码-非

module W\_74HC04(A,Y);

input[5:0]A;

output[5:0]Y;

assign Y=~A;

endmodule

//74HC04测试平台代码

`timescale 1ns/1ns;

module test\_74HC04;

reg[5:0]a;

wire[5:0]y;

W\_74HC04 u1(a,y);

initial

begin

a=6'b000001;

#10 a=a<<1;

#10 a=a<<1;

#10 a=a<<1;

#10 a=a<<1;

#10 a=a<<1;

end

endmodule

//74HC08代码-与

module W\_74HC08(A,B,Y);

input[3:0]A,B;

output[3:0]Y;

assign Y=A&B;

endmodule

//74HC08测试平台代码

`timescale 1ns/1ns

module test\_74HC08;

reg[3:0]a,b;

wire[3:0]y;

W\_74HC08 u1(a,b,y);

initial

begin

a=4'b0000;b=4'b0001;

#10 b=b<<1;

#10 b=b<<1;

#10 b=b<<1;

#10 a=4'b1111;b=4'b0001;

#10 b=b<<1;

#10 b=b<<1;

#10 b=b<<1;

end

endmodule

//74HC32代码-或

module W\_74HC32(A,B,Y);

input[3:0]A,B;

output[3:0]Y;

assign Y=A|B;

endmodule

//74HC32测试平台代码

`timescale 1ns/1ns

module test\_74HC32;

reg[3:0]a,b;

wire[3:0]y;

W\_74HC32 u1(a,b,y);

initial

begin

a=4'b0000;b=4'b0001;

#10 b=b<<1;

#10 b=b<<1;

#10 b=b<<1;

#10 a=4'b1111;b=4'b0001;

#10 b=b<<1;

#10 b=b<<1;

#10 b=b<<1;

end

endmodule

//74HC86代码-异或

module W\_74HC86(A,B,Y);

input[3:0]A,B;

output[3:0]Y;

assign Y=A^B;

endmodule

//74HC86测试平台代码

`timescale 1ns/1ns

module test\_74HC86;

reg[3:0]a,b;

wire[3:0]y;

W\_74HC86 u1(a,b,y);

initial

begin

a=4'b0000;b=4'b0001;

#10 b=b<<1;

#10 b=b<<1;

#10 b=b<<1;

#10 a=4'b1111;b=4'b0001;

#10 b=b<<1;

#10 b=b<<1;

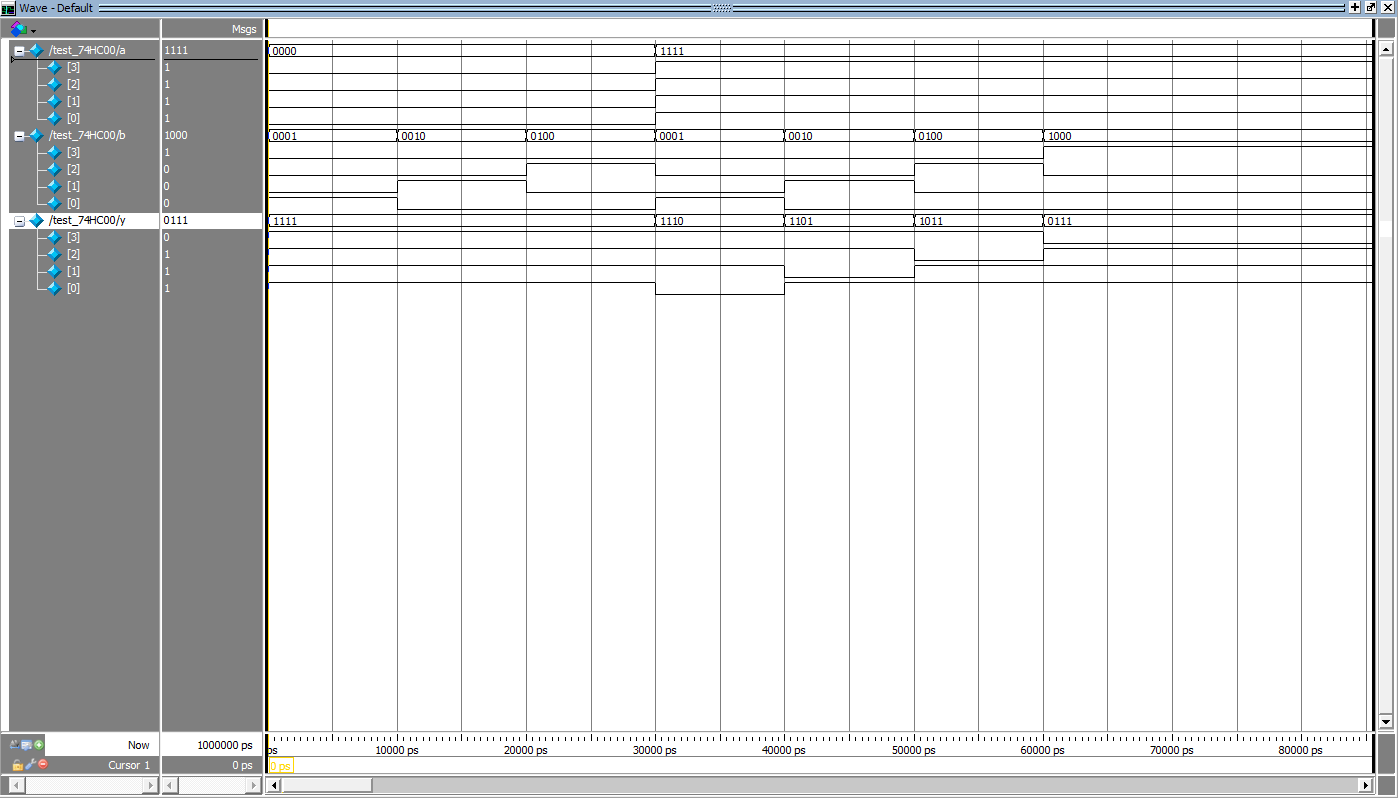
#10 b=b<<1;

end

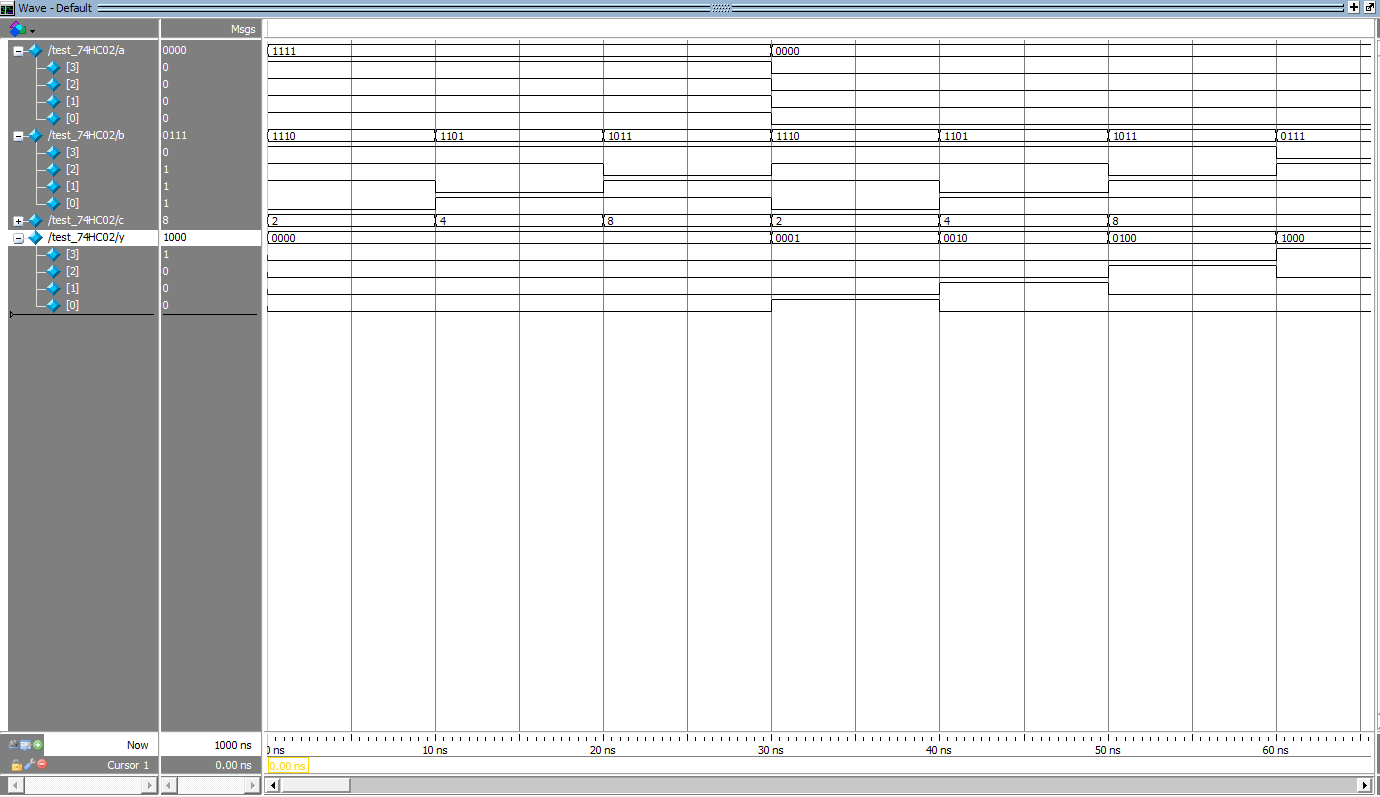
endmodule

1. 第一次仿真结果（**截图，注明对应的门电路**）。（将波形窗口背景设为**白色**，调整窗口至合适大小，使波形能完整显示，对窗口**截图**。后面实验中的仿真使用相同方法处理）

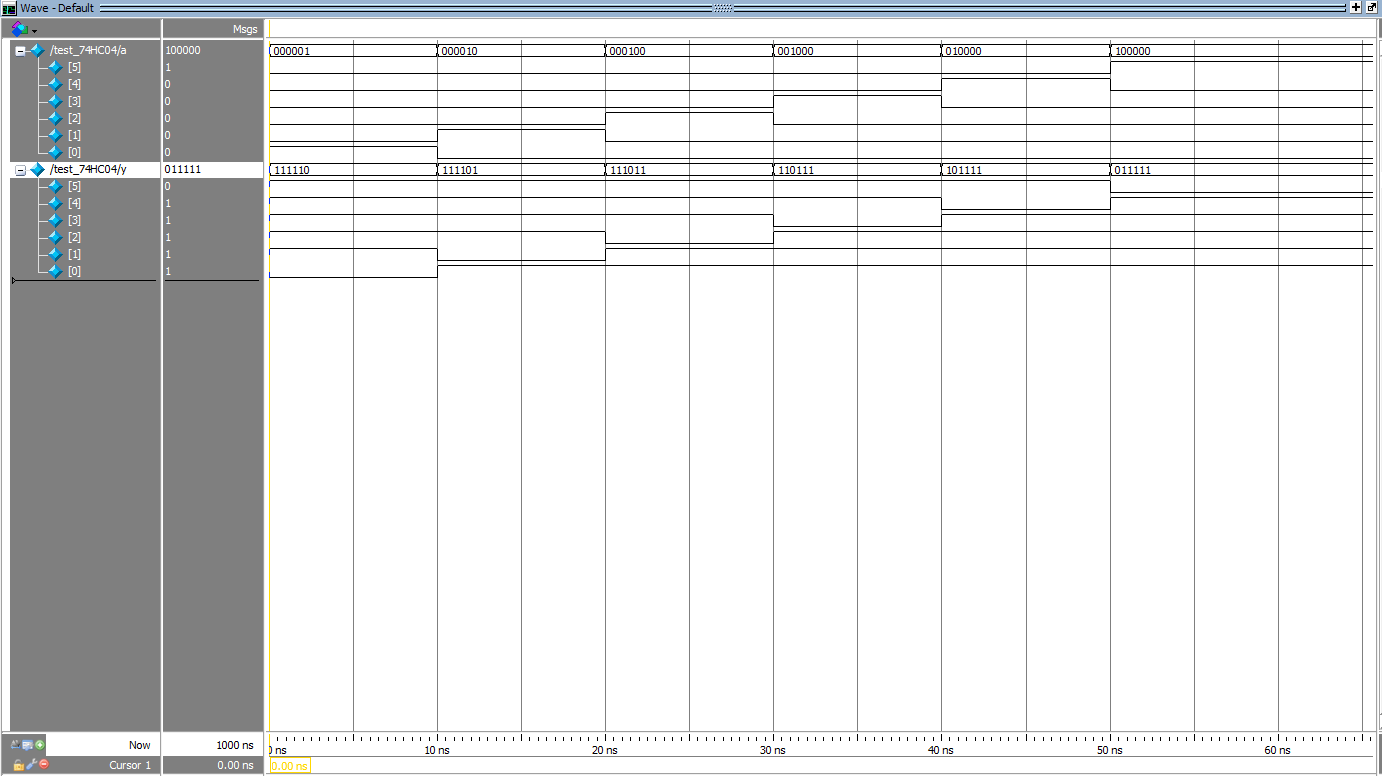
//74HC00-与非门：



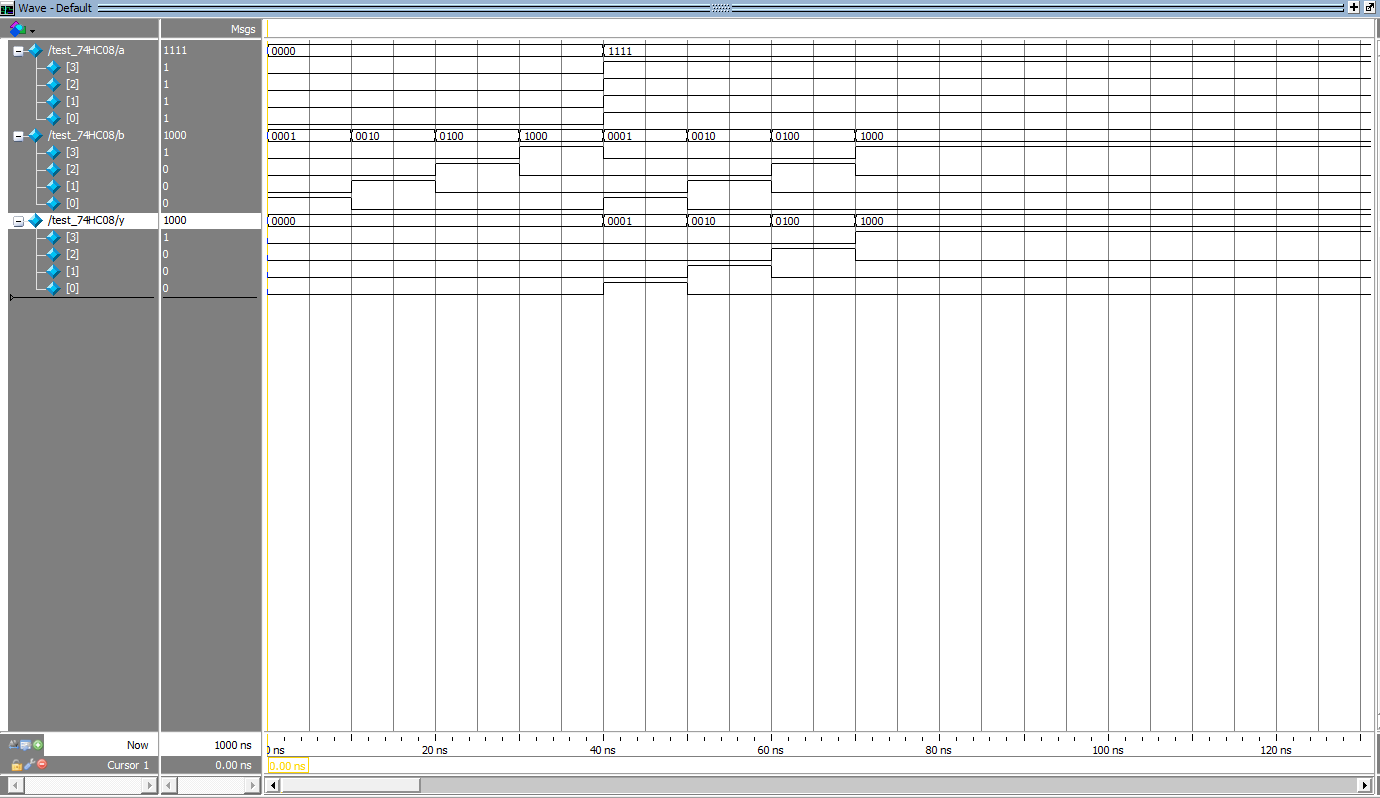
//74HC02-或非门：



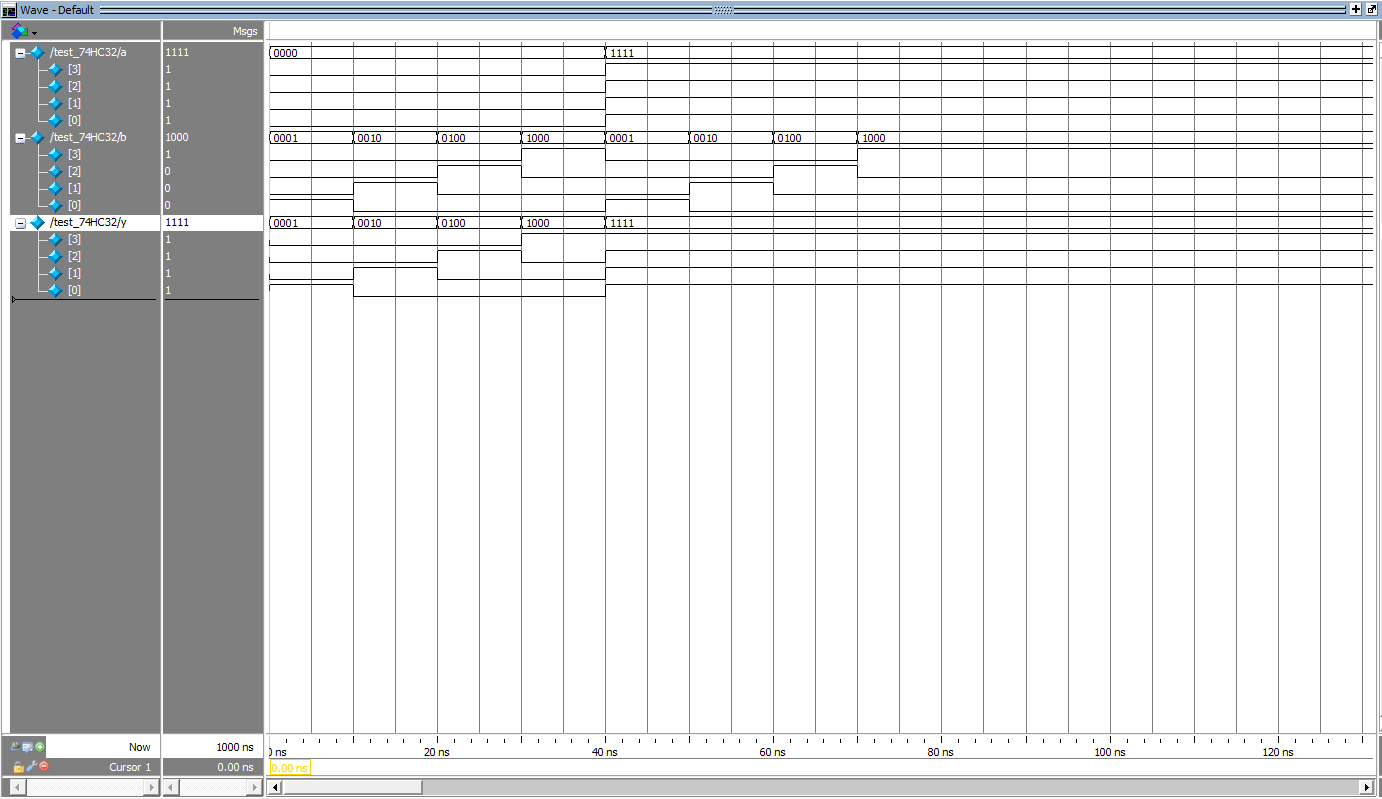
//74HC04-非门：



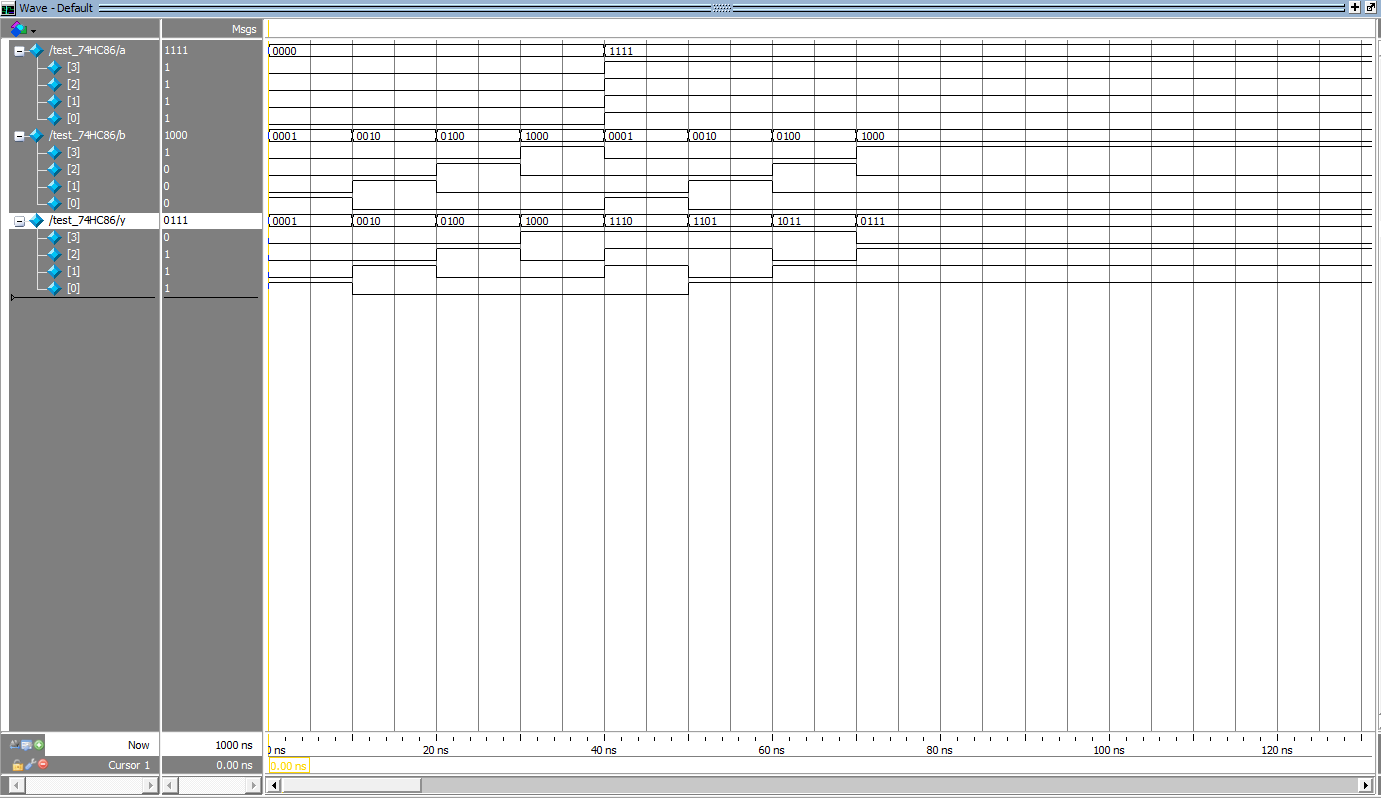
//74HC08-与门：



//74HC32-或门：

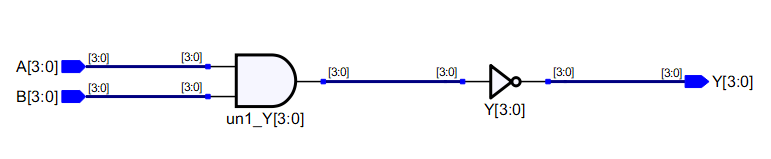


//74HC86-异或门：

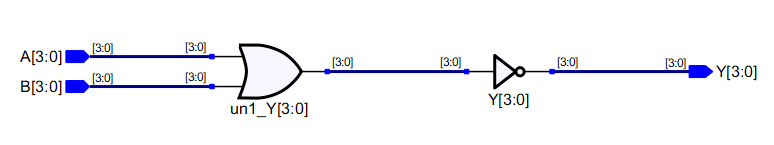


3、综合结果（**截图，注明对应的门电路**）。（将相关窗口调至合适大小，使RTL图能完整显示，对窗口截图，后面实验中的综合使用相同方法处理）

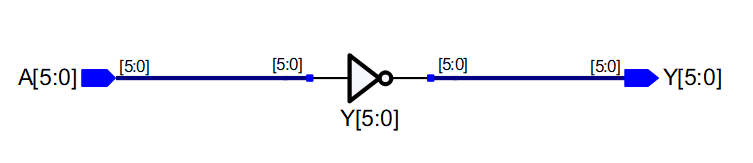
//74HC00-与非门：



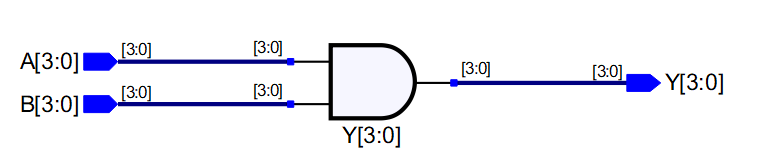
//74HC02-或非门：



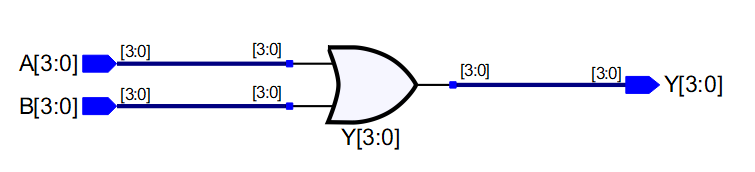
//74HC04-非门：



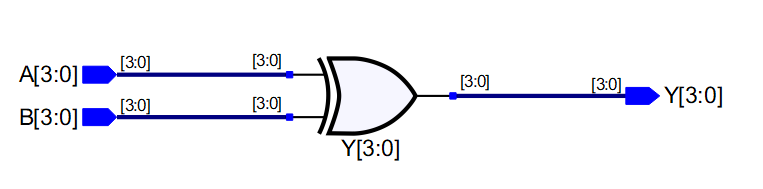
//74HC08-与门：



//74HC32-或门：

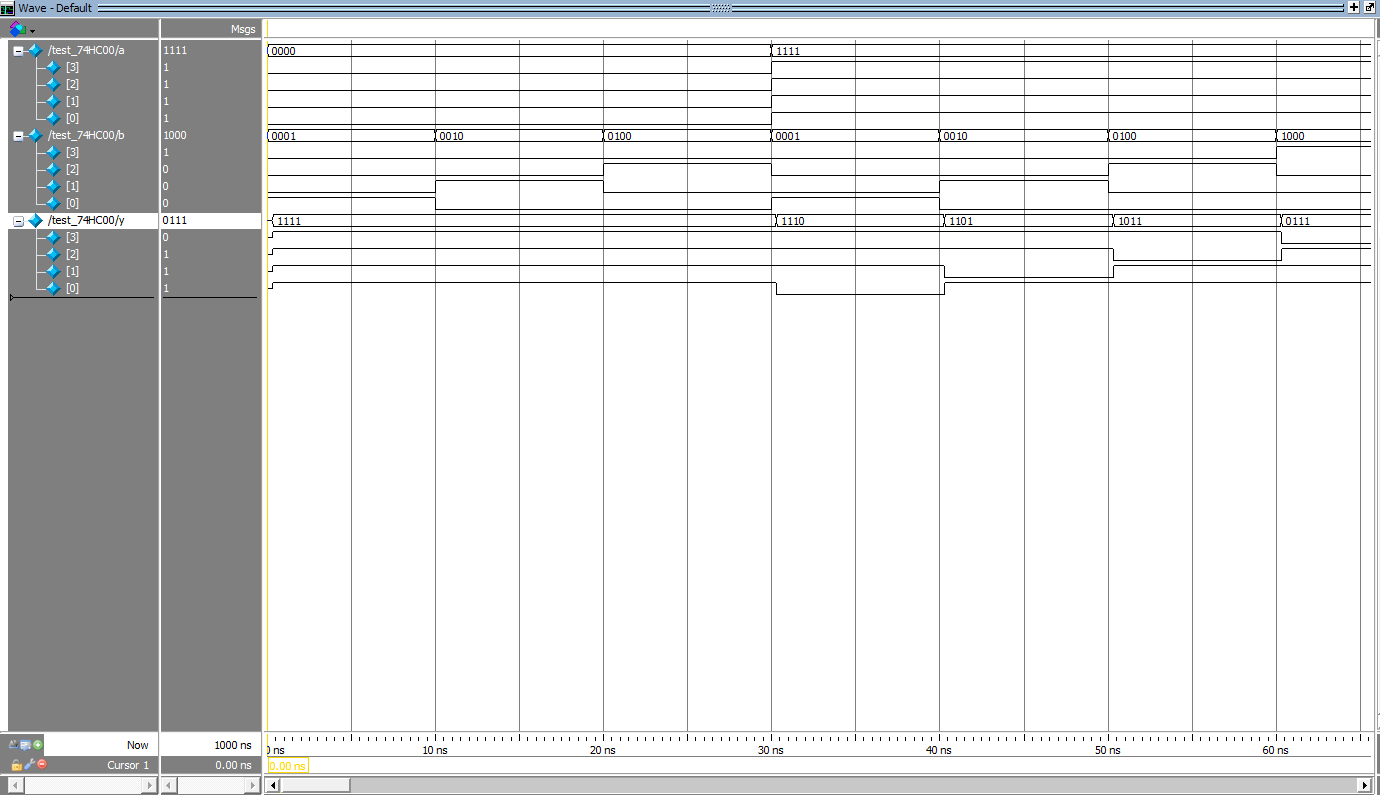


//74HC86-异或门：

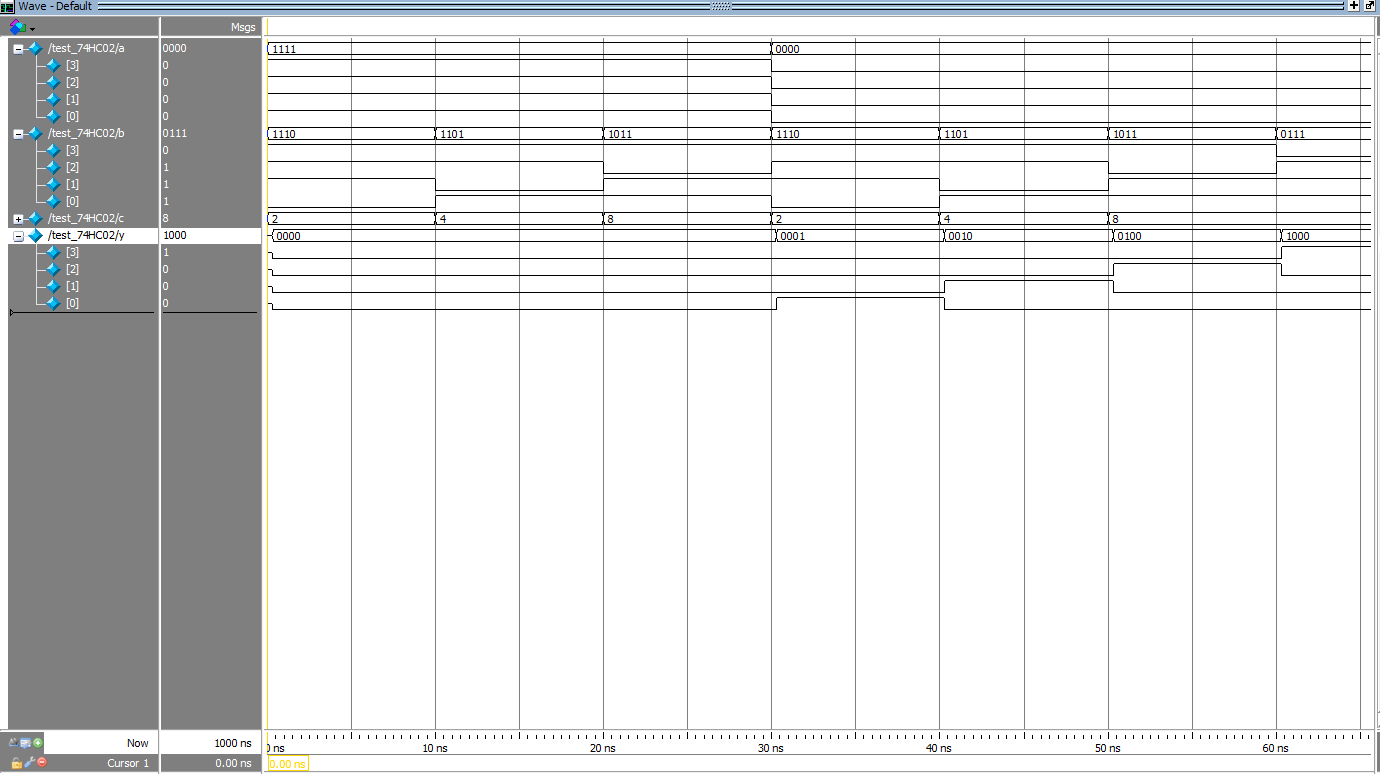


4、第二次仿真结果（综合后）（**截图，注明对应的门电路**）。回答输出信号是否有延迟，延迟时间约为多少？

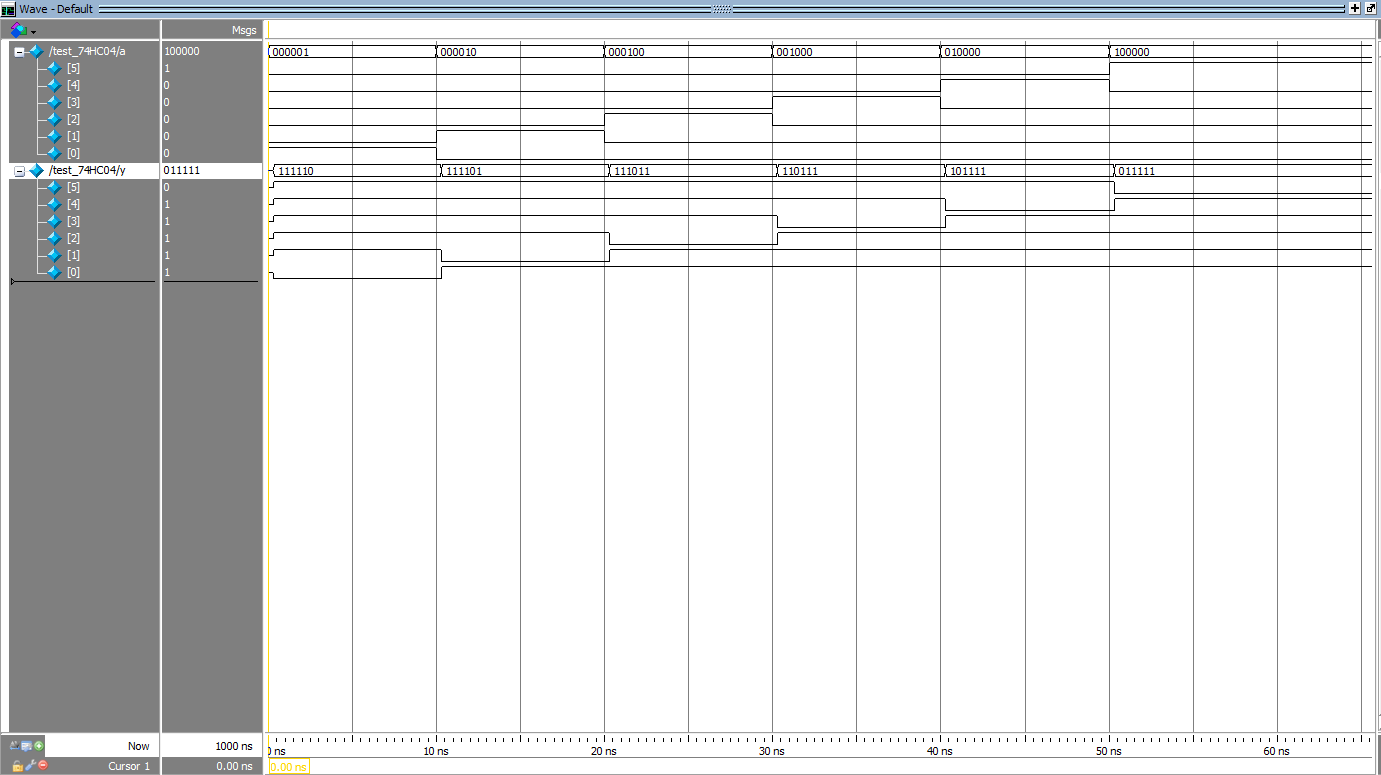
//74HC00-与非门：有延时，延时大概为290ps。



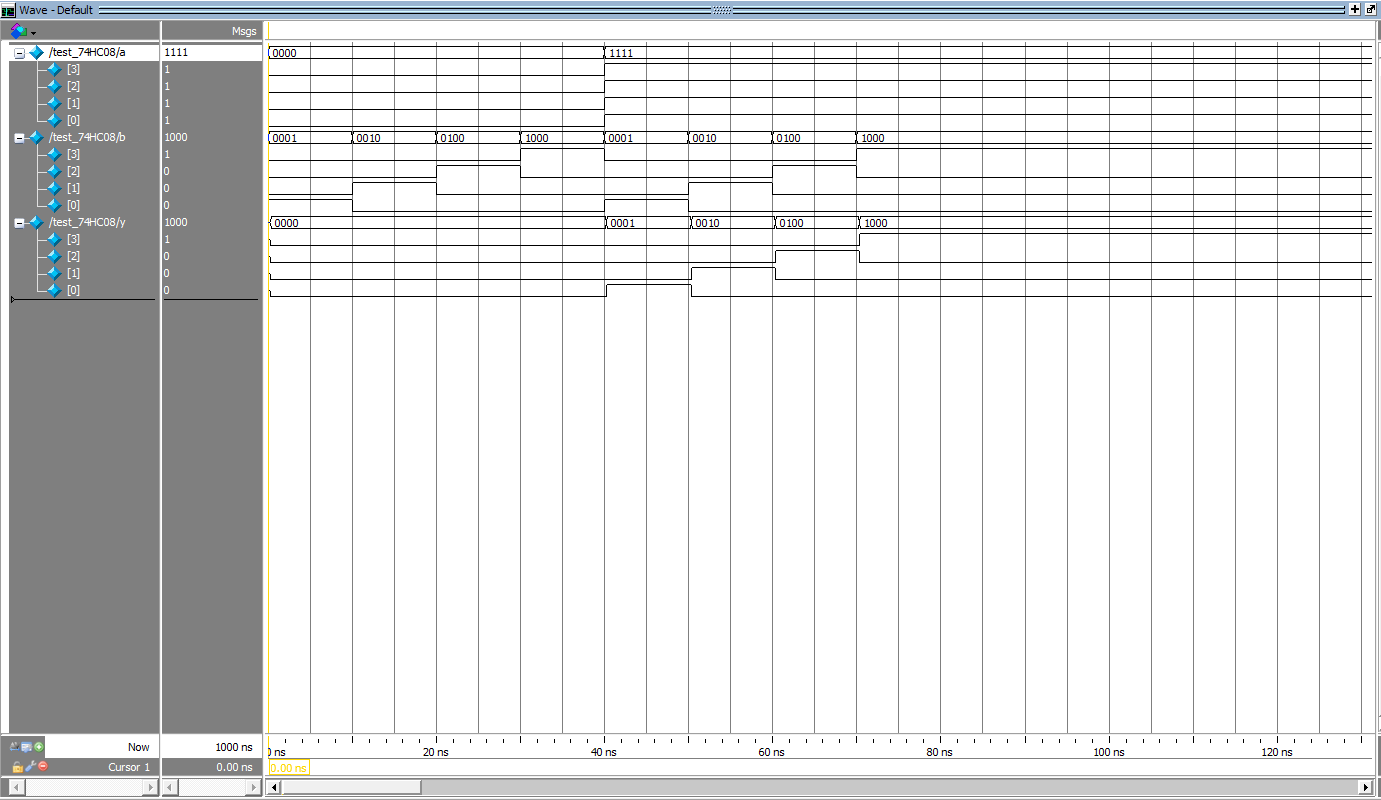
//74HC02-或非门：有延时，延时大概为300ps。



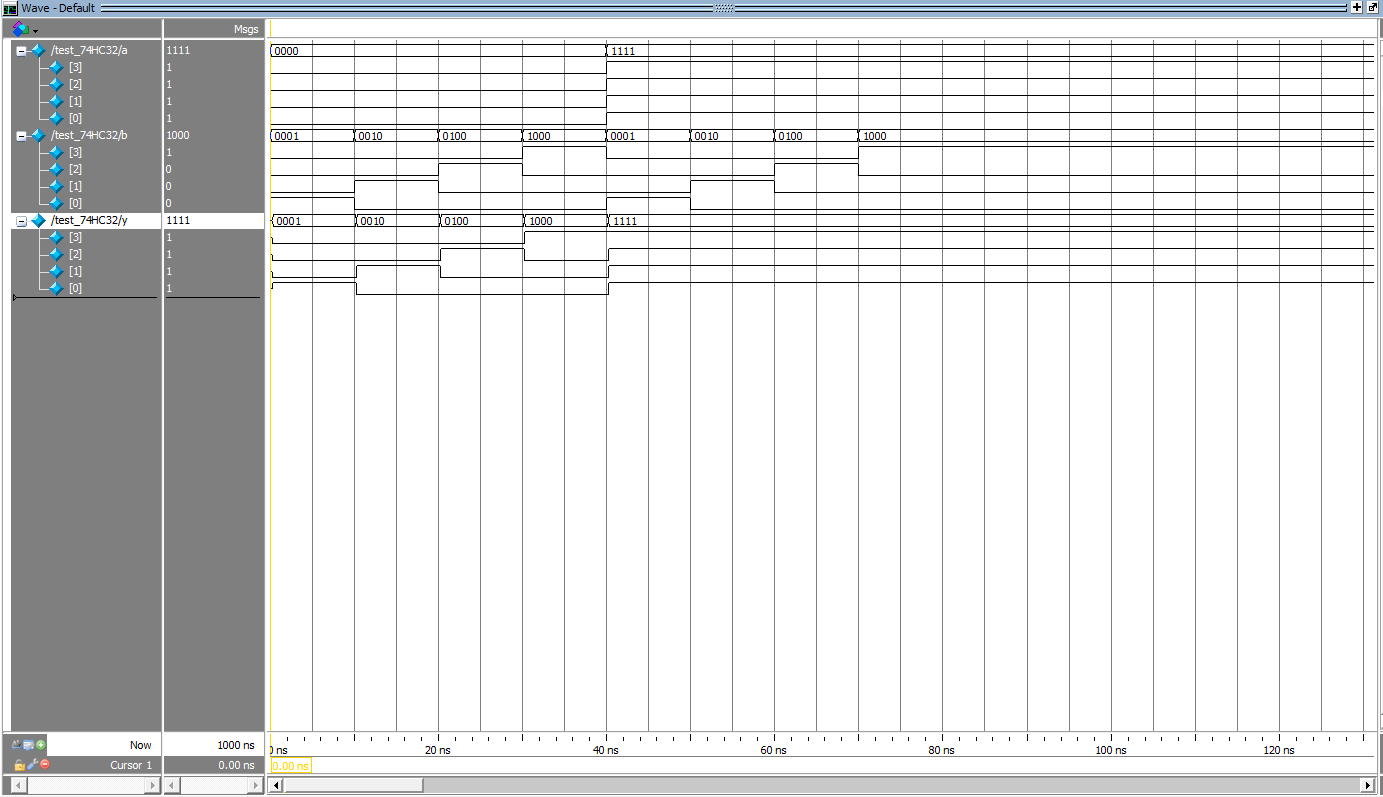
//74HC04-非门：有延时，延时大概为300ps。



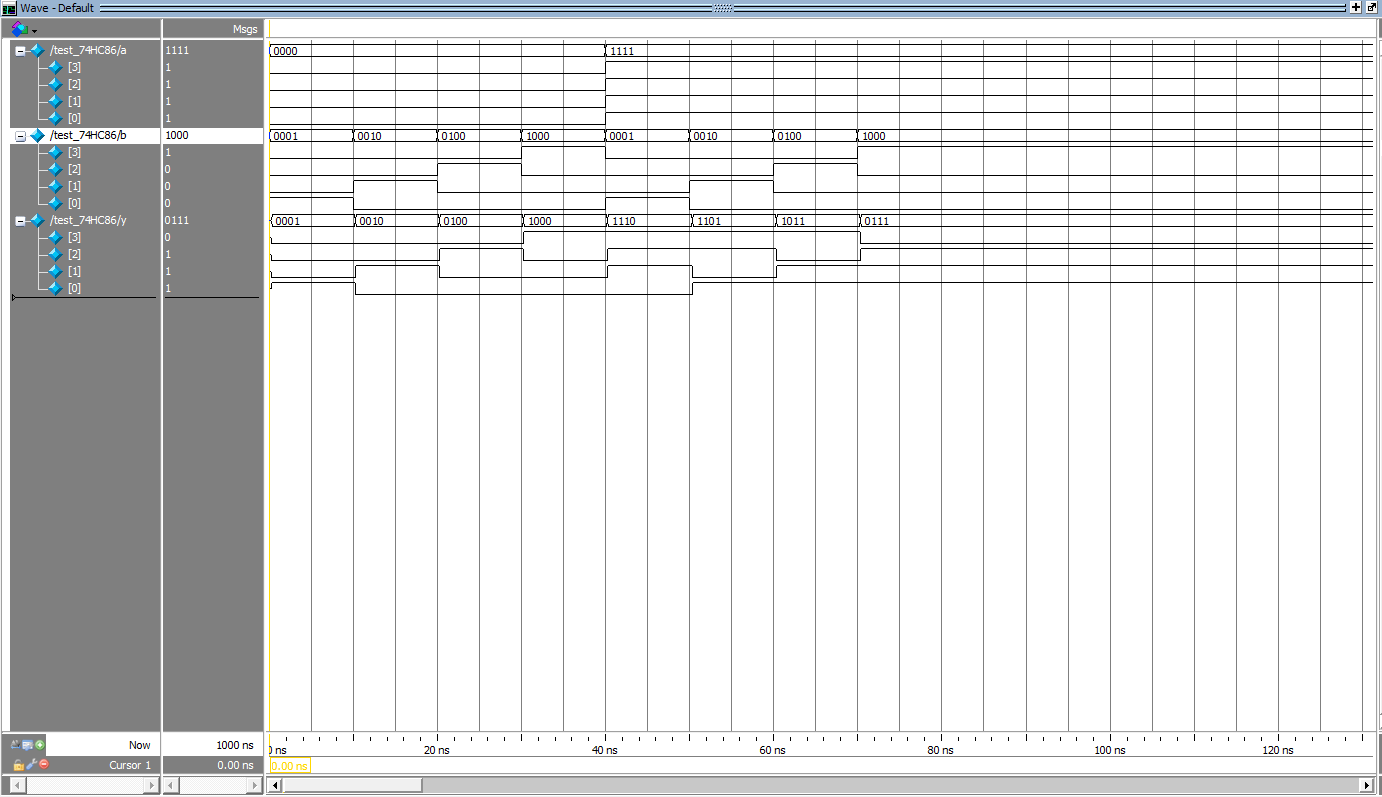
//74HC08-与门：有延时，约为290ps，无竞争冒险。



//74HC32-或门：有延时，延时大概为300ps。

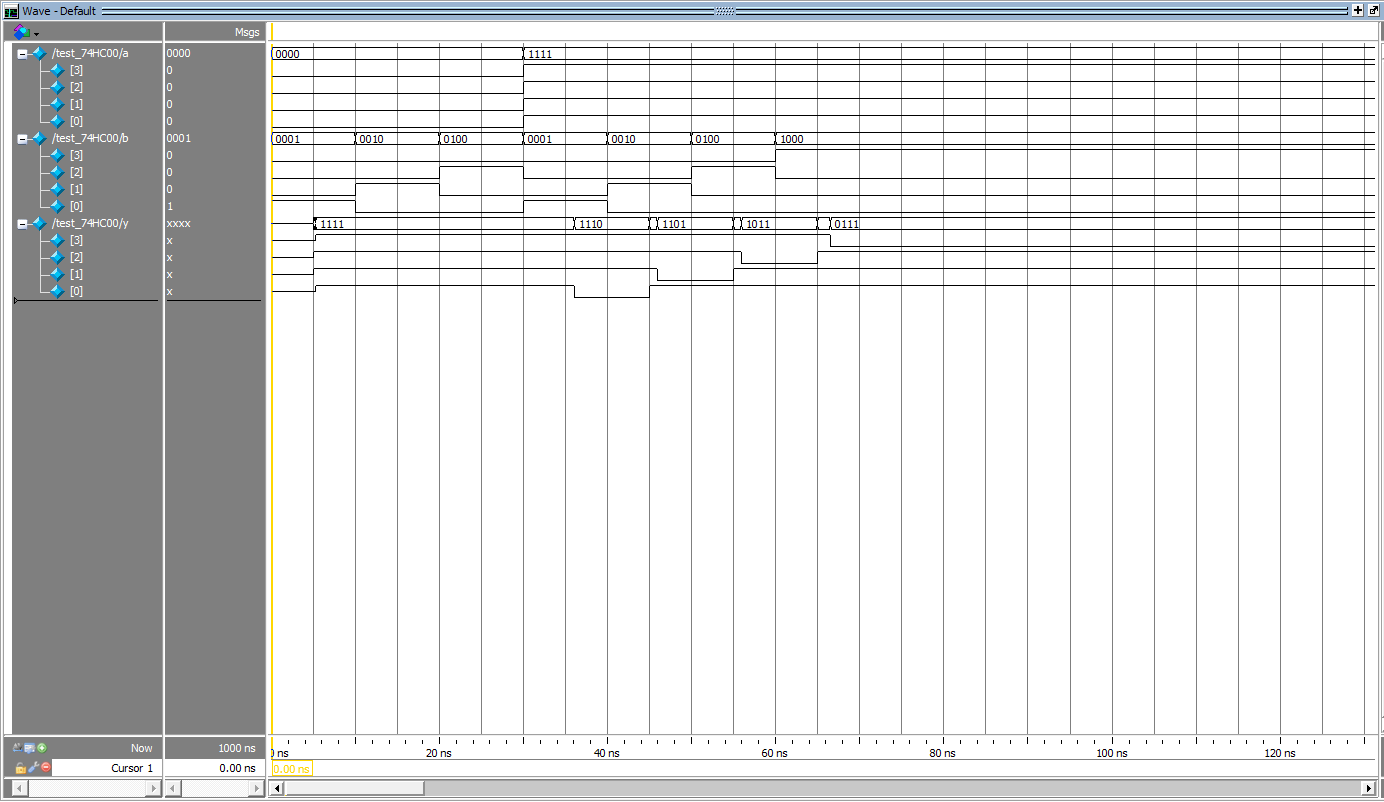


//74HC86-异或门：有延时，延时大概为300ps。



1. 第三次仿真结果（布局布线后）（**截图，注明对应的门电路**）。回答输出信号是否有延迟，延迟时间约为多少？分析是否有出现竞争冒险。

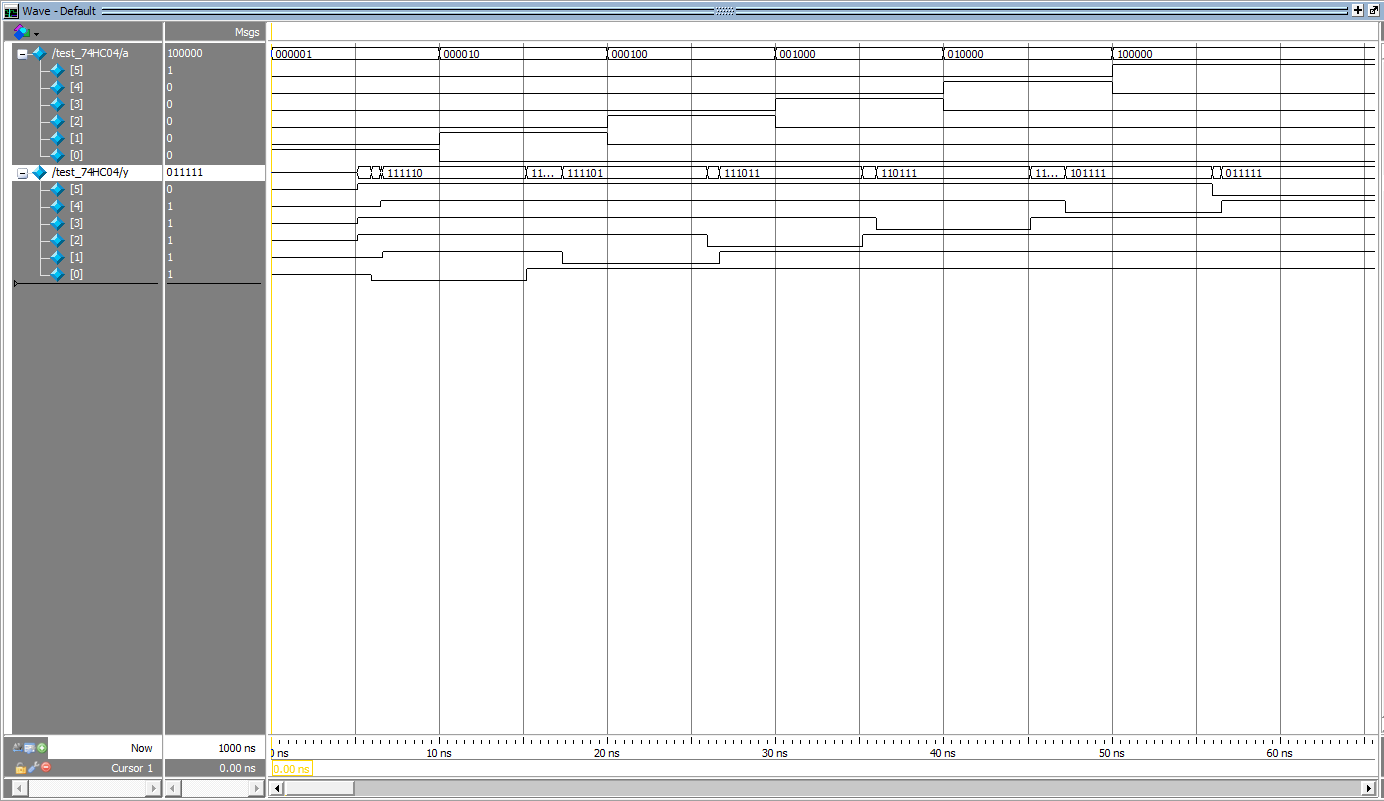
//74HC00-与非门：有延时，延时大概为7820ps。无竞争冒险。



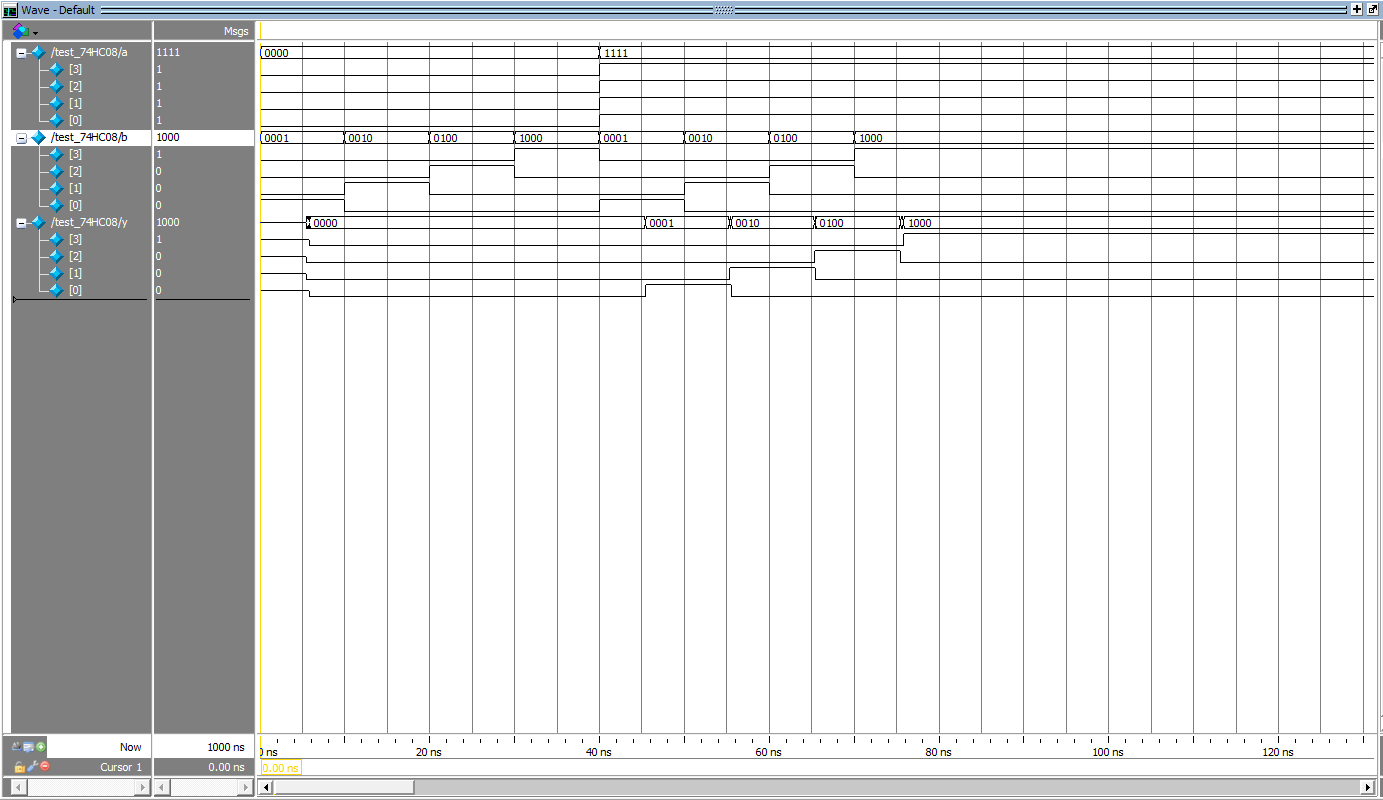
## //74HC02-或非门：有延时，延时大概为7241ps。无竞争冒险。

## 布局布线后

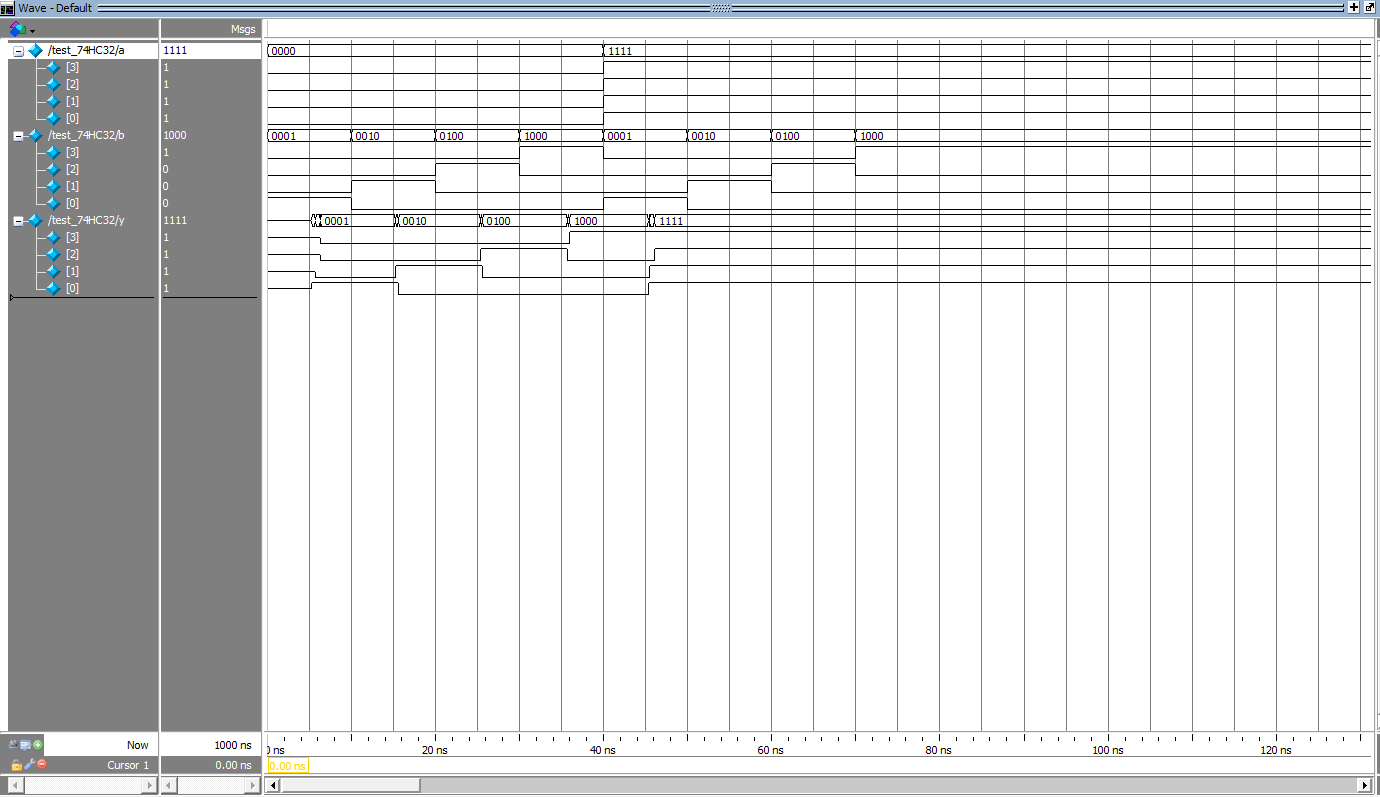
//74HC04-非门：有延时，延时大概为6859ps。无竞争冒险。



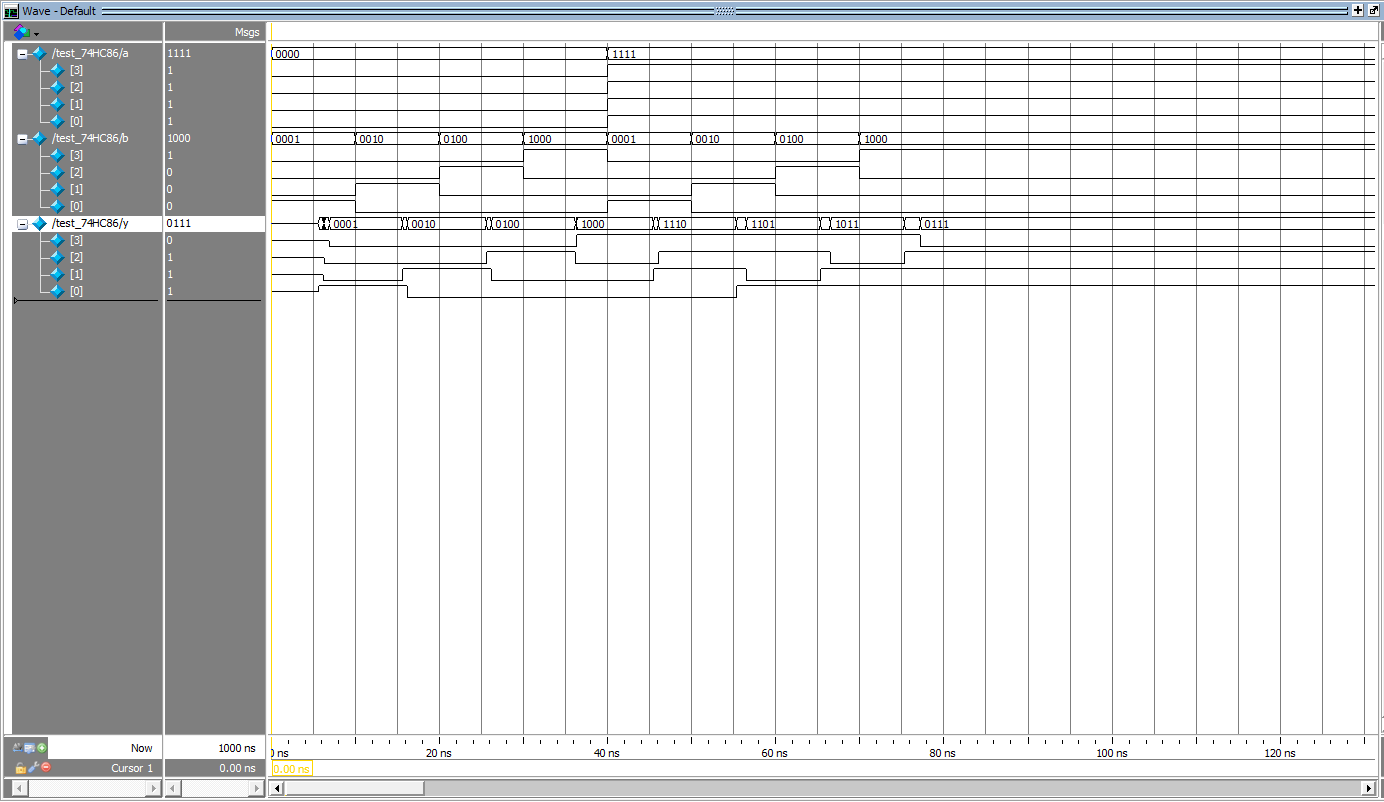
//74HC08-与门：有延时，约为7520ps，无竞争冒险。



//74HC32-或门：有延时，延时大概为7190ps。无竞争冒险。



//74HC86-异或门：有延时，延时大概为9330ps，无竞争冒险。



## 组合逻辑电路

一、实验目的

1、了解基于Verilog的组合逻辑电路的设计及其验证。

2、熟悉利用EDA工具进行设计及仿真的流程。

3、熟悉实验箱的使用和程序下载（烧录）及测试的方法。

4、学习针对实际组合逻辑电路芯片74HC148、74HC138、74HC153、74HC85、74HC283、74HC4511进行VerilogHDL设计的方法。

二、实验环境及仪器

1、Libero仿真软件。

2、数字逻辑与系统设计实验箱。

3、Actel A3P060 FPGA芯片及Flash Pro5烧录器。

三、实验内容

1、掌握Libero软件的使用方法。

2、进行针对74系列基本组合逻辑电路的设计，并完成相应的仿真实验。

3、参考教材中相应章节的设计代码、测试平台代码（可自行编程），完成74HC148、74HC138、74HC153、74HC85、74HC283、74HC4511相应的设计、综合及仿真。

4、74HC85测试平台的测试数据要求：进行比较的A、B两数，分别为本人学号的末两位，如“89”，则A数为“1000”，B数为“1001”。若两数相等，需考虑级联输入（级联输入的各种取值情况均需包括）；若两数不等，则需增加一对取值情况，验证A、B相等时的比较结果。

5、74HC4511设计成扩展型的，即能显示数字0~9、字母a~f。

6、提交针对74HC148、74HC138、74HC153、74HC85、74HC283、74HC4511的综合结果，以及相应的仿真结果。

四、实验结果和数据处理

1、所有模块及测试平台代码清单

//74HC148代码

module W\_74HC148(DataIn,EO,DataOut,EI,GS);

input [7:0]DataIn;

input EI;

output EO,GS;

output [2:0]DataOut;

reg [2:0]DataOut;

reg EO,GS;

integer I;

always @(DataIn or EI)

begin

if(EI)

begin

DataOut=7;

EO=1;GS=1;

end

else if(DataIn==8'b11111111)

begin

DataOut=7;

EO=0;GS=1;

end

else

for(I=0;I<8;I=I+1)

begin

if(~DataIn[I])

begin

DataOut=~I;

EO=1;GS=0;

end

end

end

endmodule

//74HC148测试平台代码

`timescale 1ns/10ps

module test\_74HC148;

reg EI;

reg [7:0]DataIn;

wire [2:0]DataOut;

wire EO,GS;

initial

begin

EI=0;

DataIn='b11111110;

repeat(9)

#20 DataIn=DataIn<<1;

end

W\_74HC148 u1(DataIn,EO,DataOut,EI,GS);

endmodule

//74HC138代码

module W\_74HC138(DataIn,Enable,Eq);

input [2:0]DataIn;

input [3:1]Enable;

output [7:0]Eq;

reg [7:0]Eq;

wire [2:0]DataIn;

integer I;

always@(DataIn or Enable)

begin

if(Enable[1]==1&&Enable[2]==1&&Enable[3]==0)

for(I=0;I<=7;I=I+1)

if(DataIn==I)

Eq[I]=0;

else

Eq[I]=1;

else

Eq=0;

end

endmodule

//74HC138测试平台代码

`timescale 1ns/10ps

module testbench;

reg[2:0]in;

reg [3:1]enable;

wire[7:0]eq;

W\_74HC138 u1(in,enable,eq);

initial

begin

in=0;

repeat(20)

#20 in=$random;

end

initial

begin

enable[1]=0;enable[2]=0;enable[3]=1;

#40 enable[1]=1;enable[2]=1;enable[3]=0;

end

endmodule

//74HC153代码

module W\_74HC153(D0,D1,D2,D3,Sel0,Sel1,Result);

input D0,D1,D2,D3,Sel0,Sel1;

output Result;

reg Result;

always@(D0 or D1 or D2 or D3 or Sel0 or Sel1)

begin

case({Sel0,Sel1})

0:Result=D0;

1:Result=D1;

2:Result=D2;

3:Result=D3;

default:Result=1'bx;

endcase

end

endmodule

//74HC153测试平台代码

`timescale 1ns/1ps;

module test\_74HC153;

reg D0,D1,D2,D3,Sel0,Sel1;

wire Result;

W\_74HC153 u1(D0,D1,D2,D3,Sel0,Sel1,Result);

initial

begin

D0=0;D1=0;D2=0;D3=0;

Sel1=0;Sel0=0;

#100 D0=1;D1=0;D2=0;D3=1;

#100 Sel1=0;Sel0=1;

#100 Sel1=1;Sel0=0;

#100 Sel1=1;Sel0=1;

#100;

end

endmodule

//74HC85代码

module W\_74HC85(A,B,Q);

input[3:0]A,B;

output[2:0]Q;

reg[2:0]Q;

always@(A or B)

begin

if(A>B)

Q=3'b100;

else if(A<B)

Q=3'b001;

else if(A==B)

Q=3'b010;

end

endmodule

//74HC85测试平台代码

`timescale 1ns/10ps

module test\_74HC85;

reg[3:0]a;

reg[3:0]b;

wire[2:0]q;

integer I;

W\_74HC85 u1(a,b,q);

initial

begin

a=0;b=0;

#5 a=4'b0010;b=4'b1000;

#5 a=4'b0100;b=4'b0100;

end

initial

$monitor("time=%t,A=%b,B=%b,q=%b",$time,a,b,q);

endmodule

//74HC283代码

module W\_74HC283(DataA,DataB,Cin,Sum,Cout);

input [3:0]DataA,DataB;

input Cin;

output [3:0]Sum;

output Cout;

reg[3:0]Sum;

reg Cout;

always@(DataA or DataB or Cin)

{Cout,Sum}=DataA+DataB+Cin;

endmodule

//74HC283测试平台代码

`timescale 1ns/1ns;

module test\_74HC283;

reg[3:0]a,b;

reg in;

wire [3:0]sum;

wire out;

integer I;

W\_74HC283 u1(a,b,in,sum,out);

task accumulate;

begin

a=4'b0100;

for(I=0;I<16;I=I+1)

begin

b=I;

#5;

end

end

endtask

initial

begin

in=0;accumulate();

in=1;accumulate();

end

initial $monitor("time=%t,a=%b,b=%b,in=%b,sum=%b,out=%b",$time,a,b,in,out,sum);

endmodule

//74HC4511代码

module W\_74HC4511(IN,Seg,LT,BI,LE);

input LT,BI,LE;

input[3:0]IN;

output[7:0]Seg;

reg[7:0]Seg;

always@(IN or LT or BI or LE)

begin

if(!LT)Seg=8'b11111111;

else if(!BI)Seg=8'b00000000;

else if(!LE)

begin

case(IN)

4'd1:Seg=8'b00000110;

4'd2:Seg=8'b01011011;

4'd3:Seg=8'b01001111;

4'd4:Seg=8'b01100110;

4'd5:Seg=8'b01101101;

4'd6:Seg=8'b01111101;

4'd7:Seg=8'b00000111;

4'd8:Seg=8'b01111111;

4'd9:Seg=8'b01101111;

4'd10:Seg=8'b01110111;

4'd11:Seg=8'b01111100;

4'd12:Seg=8'b00111001;

4'd13:Seg=8'b01011110;

4'd14:Seg=8'b01111001;

4'd15:Seg=8'b01110001;

default:Seg=7'b0000000;

endcase

end

else Seg=7'b0000000;

end

endmodule

//74HC4511测试平台代码

module test\_74HC4511;

reg[3:0]IN;

wire[7:0]Seg;

reg LT,BI,LE;

W\_74HC4511 u1(IN,Seg,LT,BI,LE);

initial

begin

LT=1;BI=1;LE=0;IN=0 ;

#30 IN=4'b0001;

#30 IN=4'b1000;

#30 IN=4'b0111;

#30 IN=4'b1010;

#30 IN=4'b0101;

#30 LE=1;

#30 BI=0;

#30 LT=0;

#20;

end

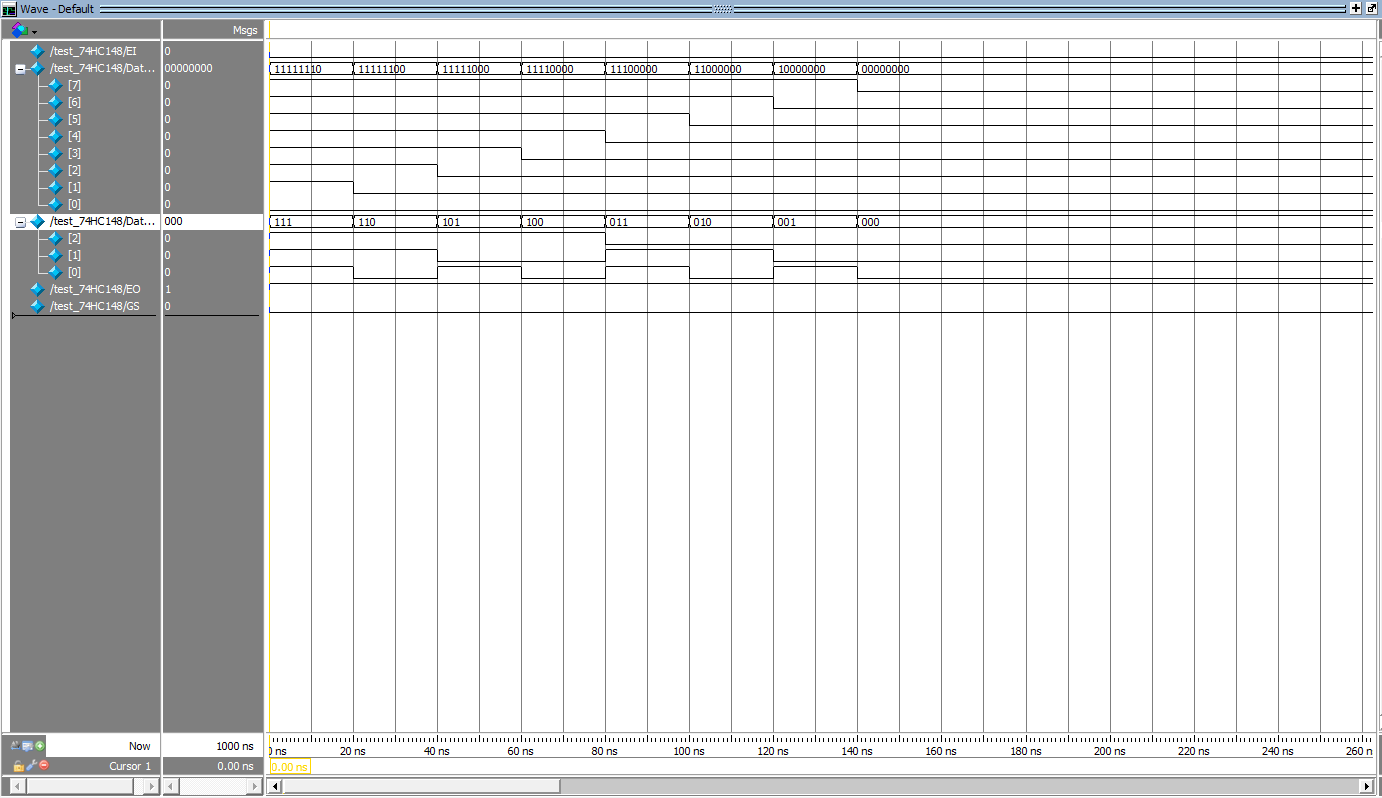
initial

$monitor("time=%t,IN=%b,Seg=%b,LT=%b,BI=%b,LE=%b",$time,IN,Seg,LT,BI,LE);

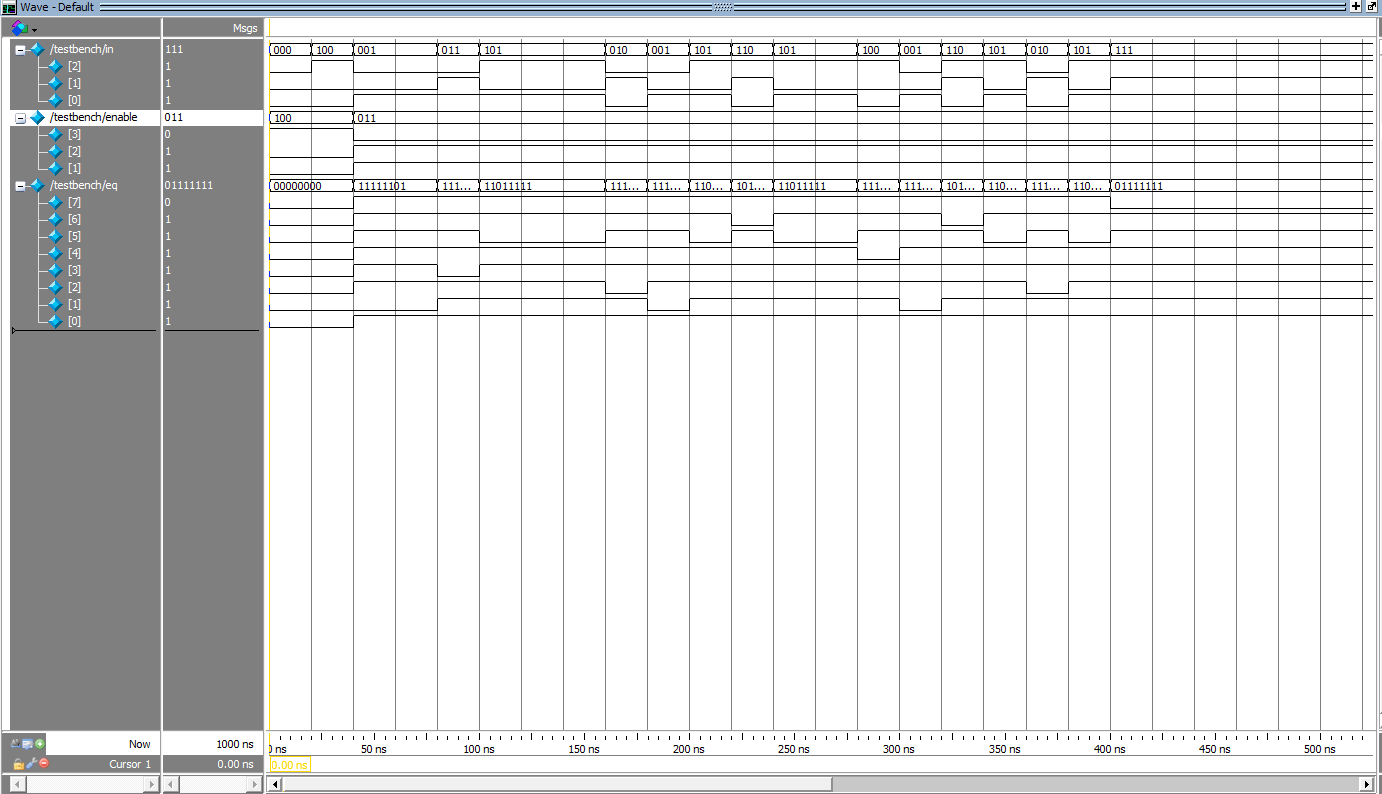
endmodule

2、第一次仿真结果（**截图，注明对应的模块**）

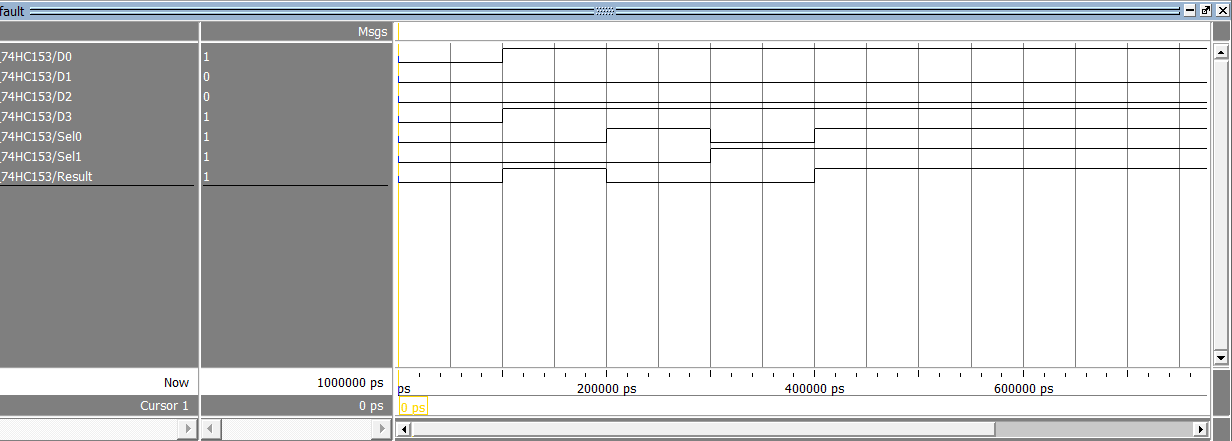
//74HC148：



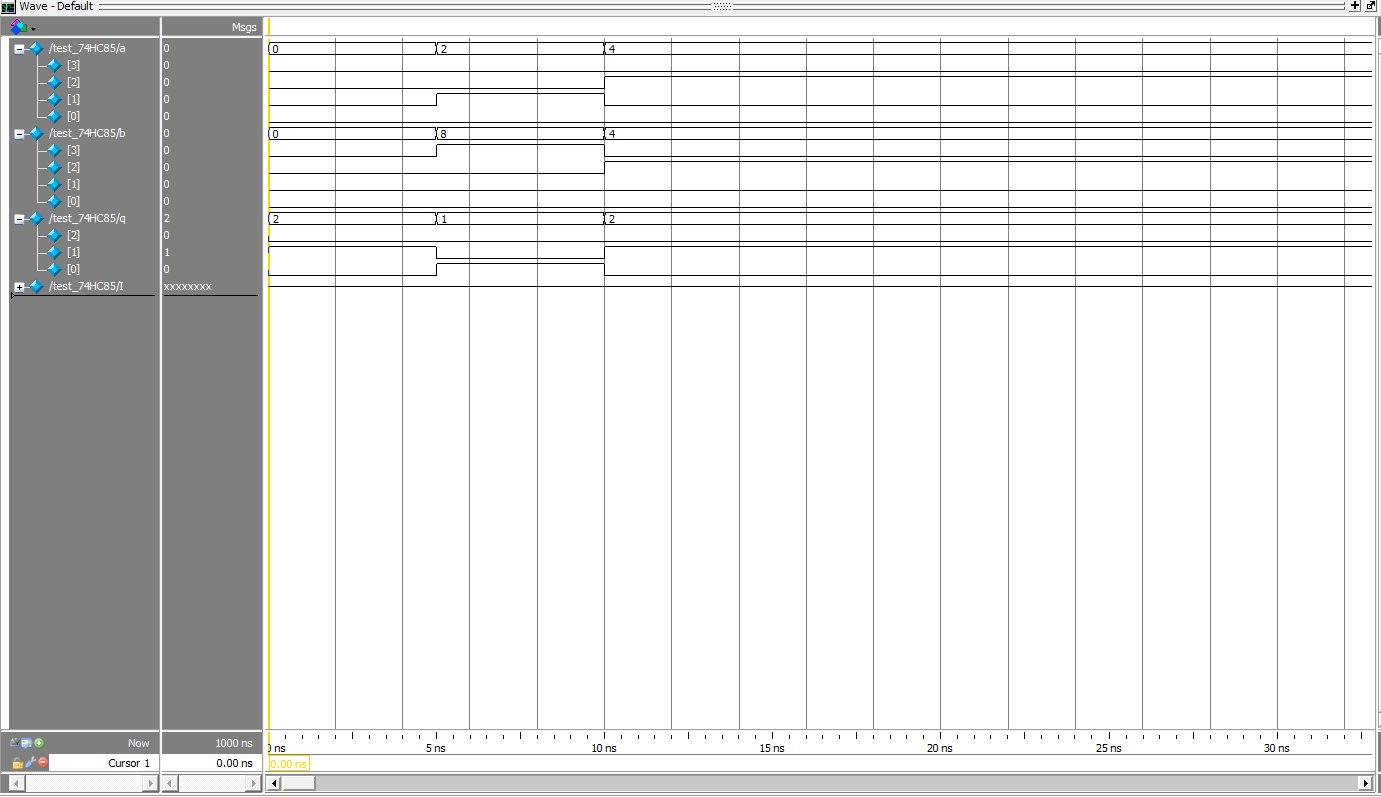
//74HC138：



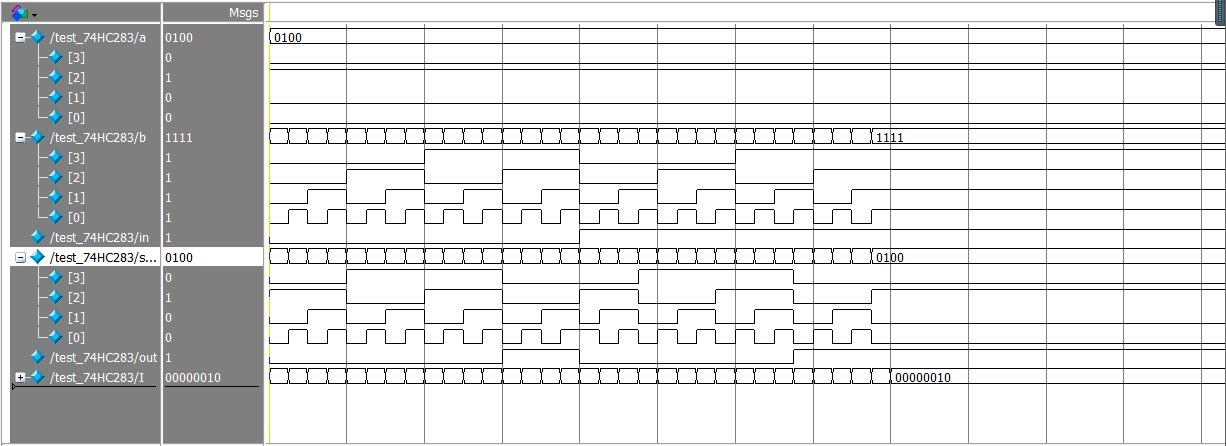
//74HC153：



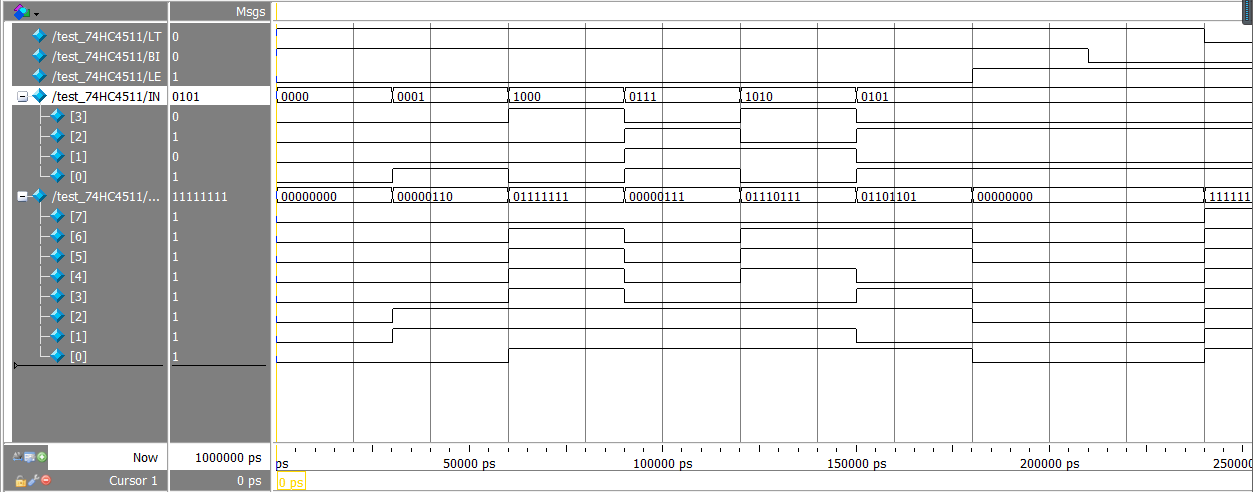
//74HC85：



//74HC283：

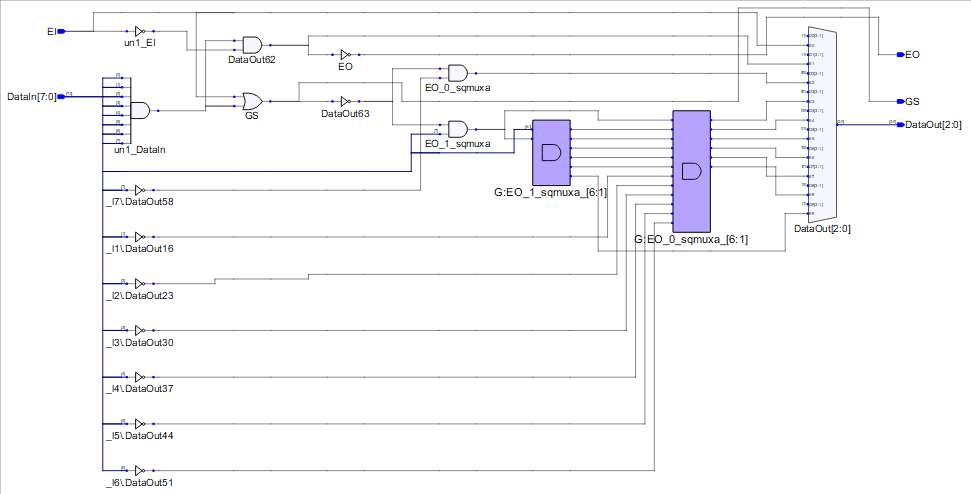


//74HC4511：

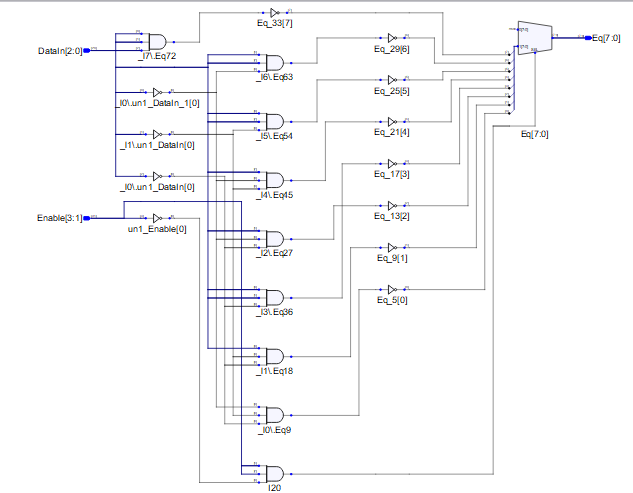


3、综合结果（**截图，注明对应的模块**）

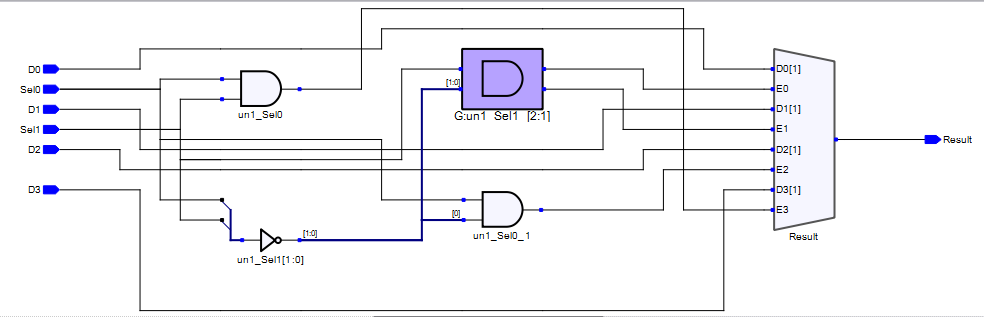
//74HC148：

****

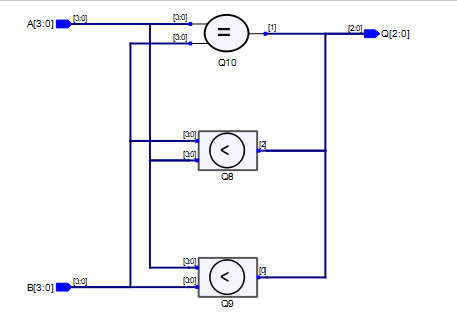
//74HC138：

****

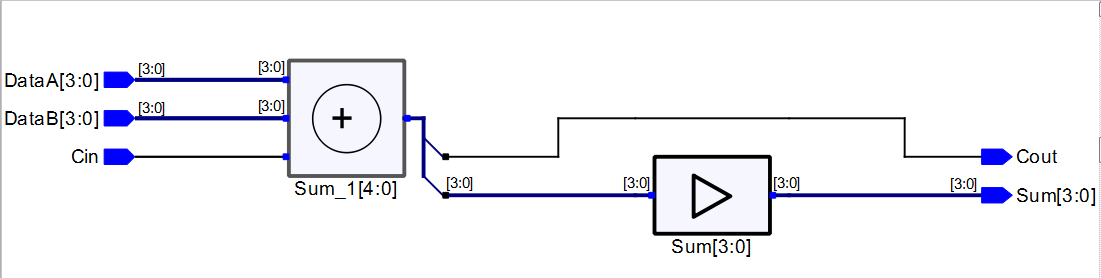
//74HC153：

****

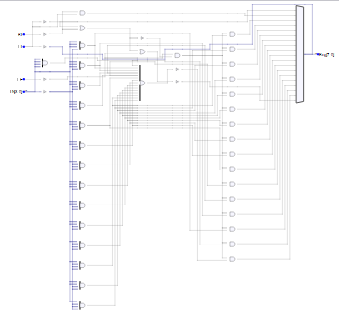
//74HC85：

****

//74HC283：

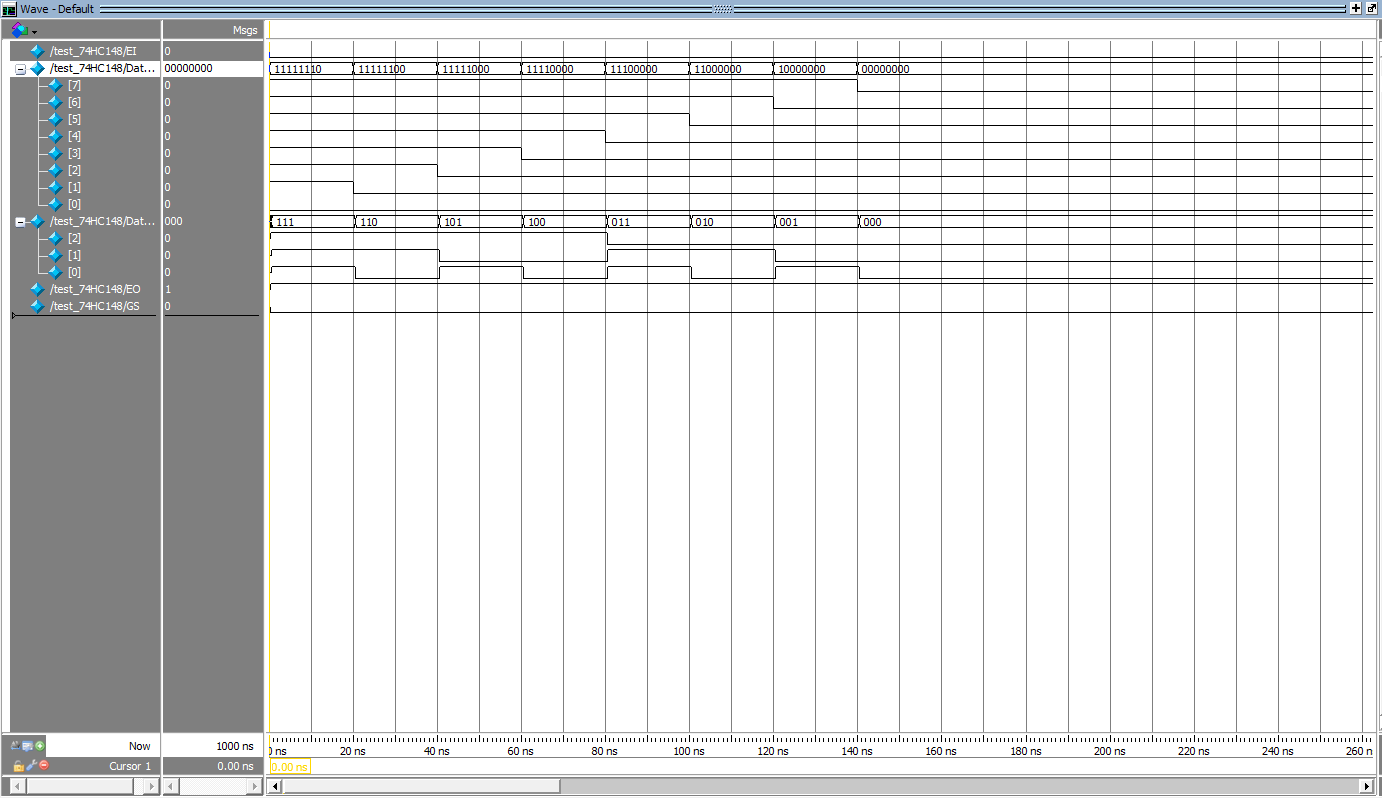
****

//74HC4511：

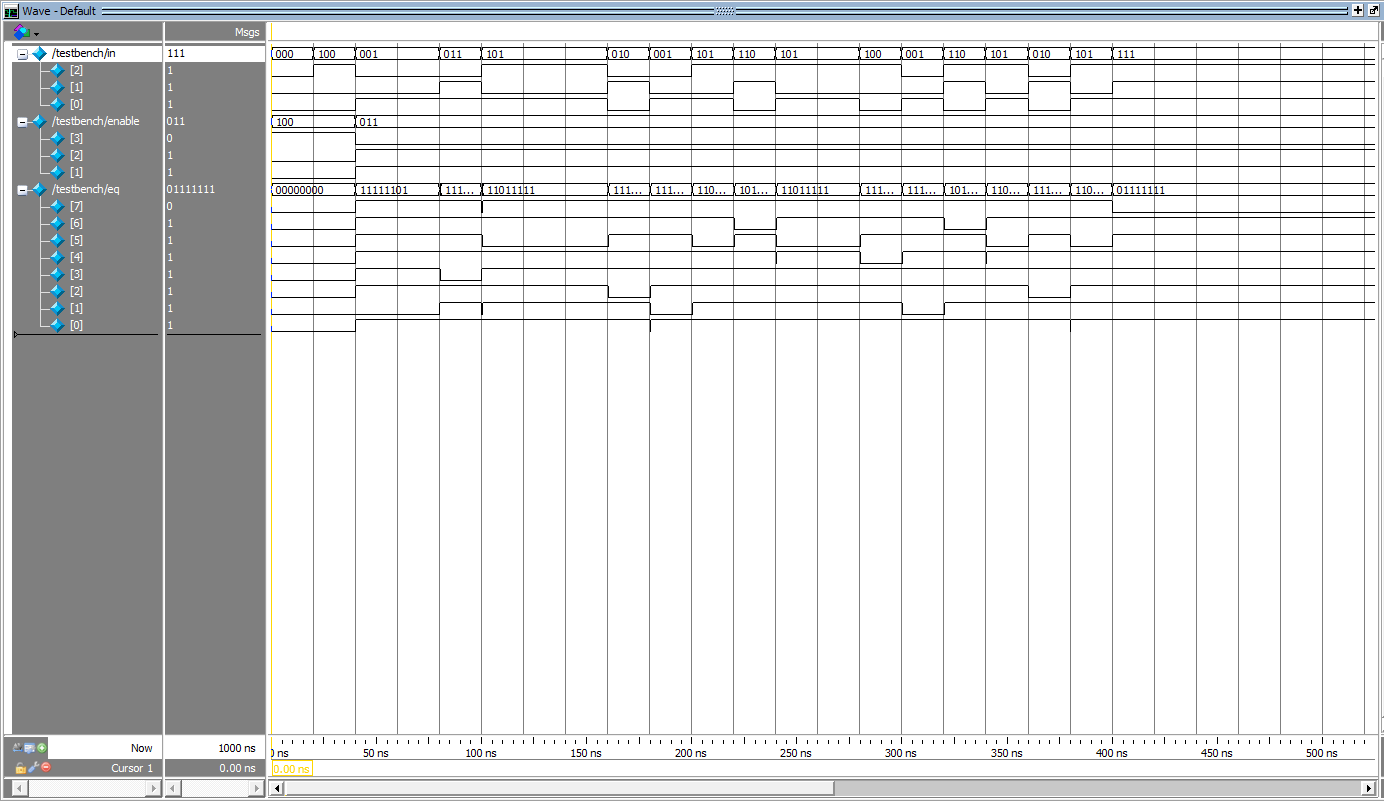
****

1. 第二次仿真结果（综合后）（**截图，注明对应的模块**）。回答输出信号是否有延迟，延迟时间约为多少？

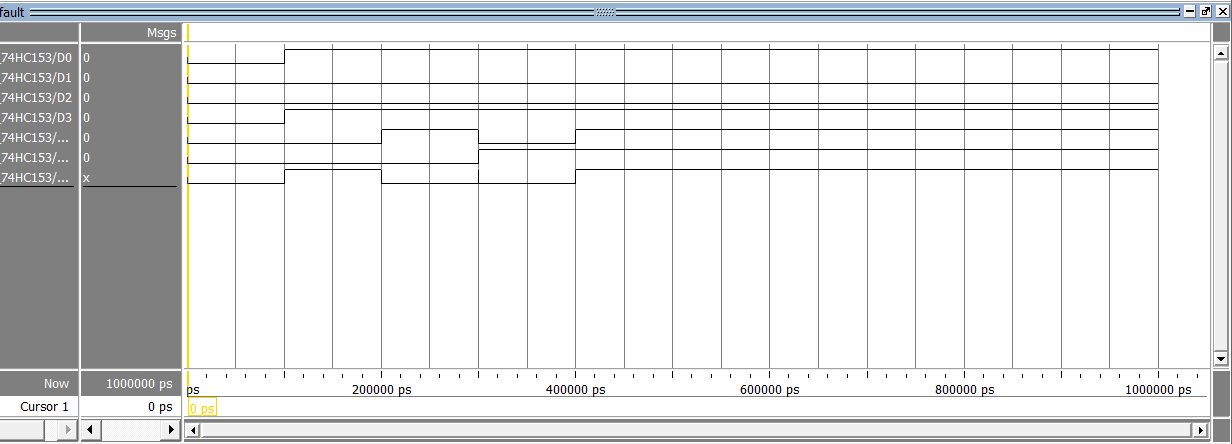
//74HC148：信号有延迟约500ps，出现竞争冒险



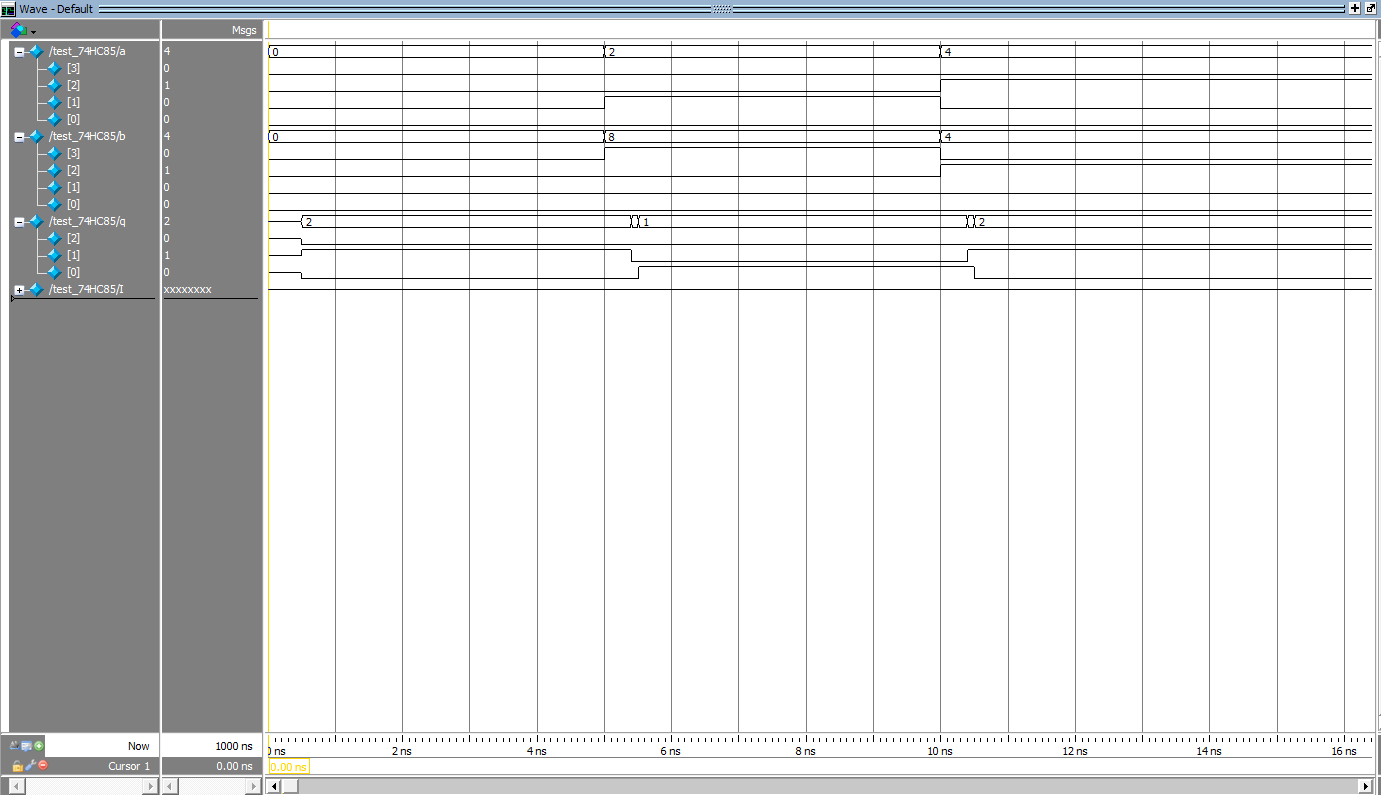
//74HC138：有延时，延时大概为300ps，有竞争冒险。



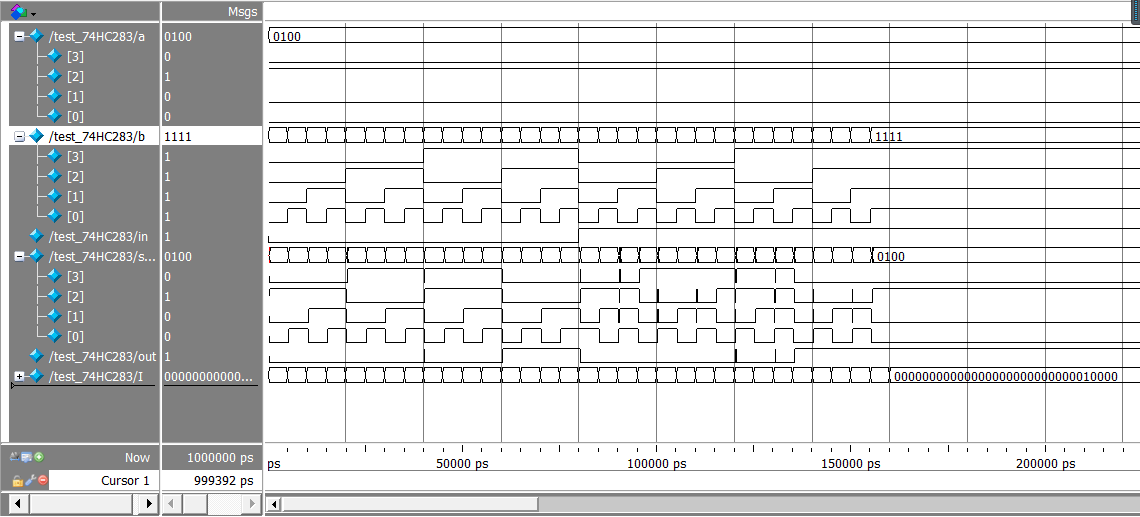
//74HC153：有延时，延时约为560ps, 有竞争冒险。



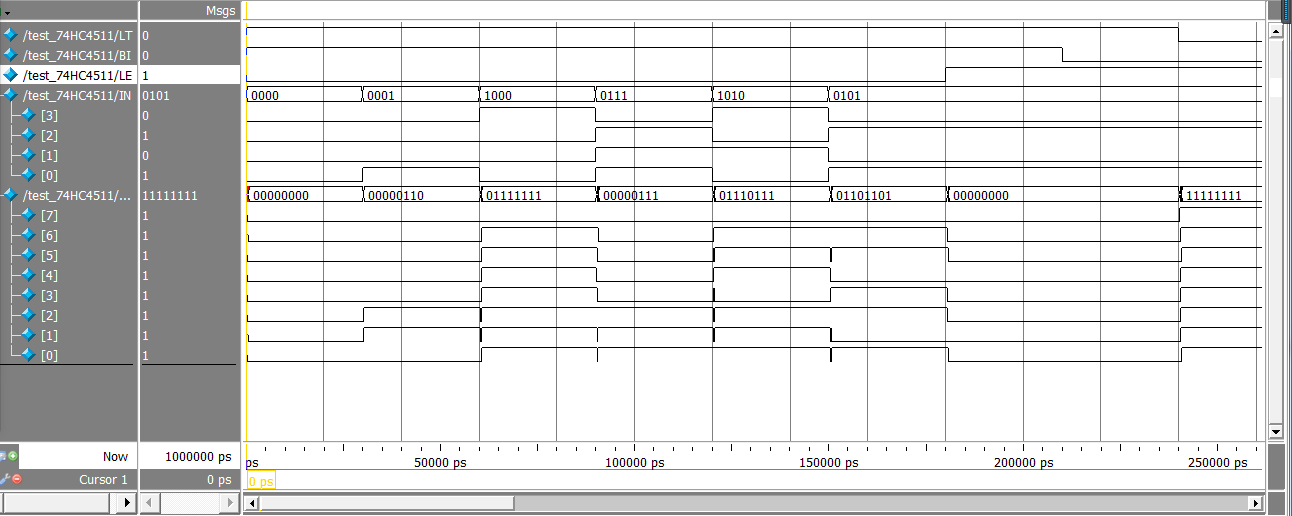
//74HC85：信号有延迟约371ps，出现竞争冒险。



//74HC283：有延时，延时大概300ps。

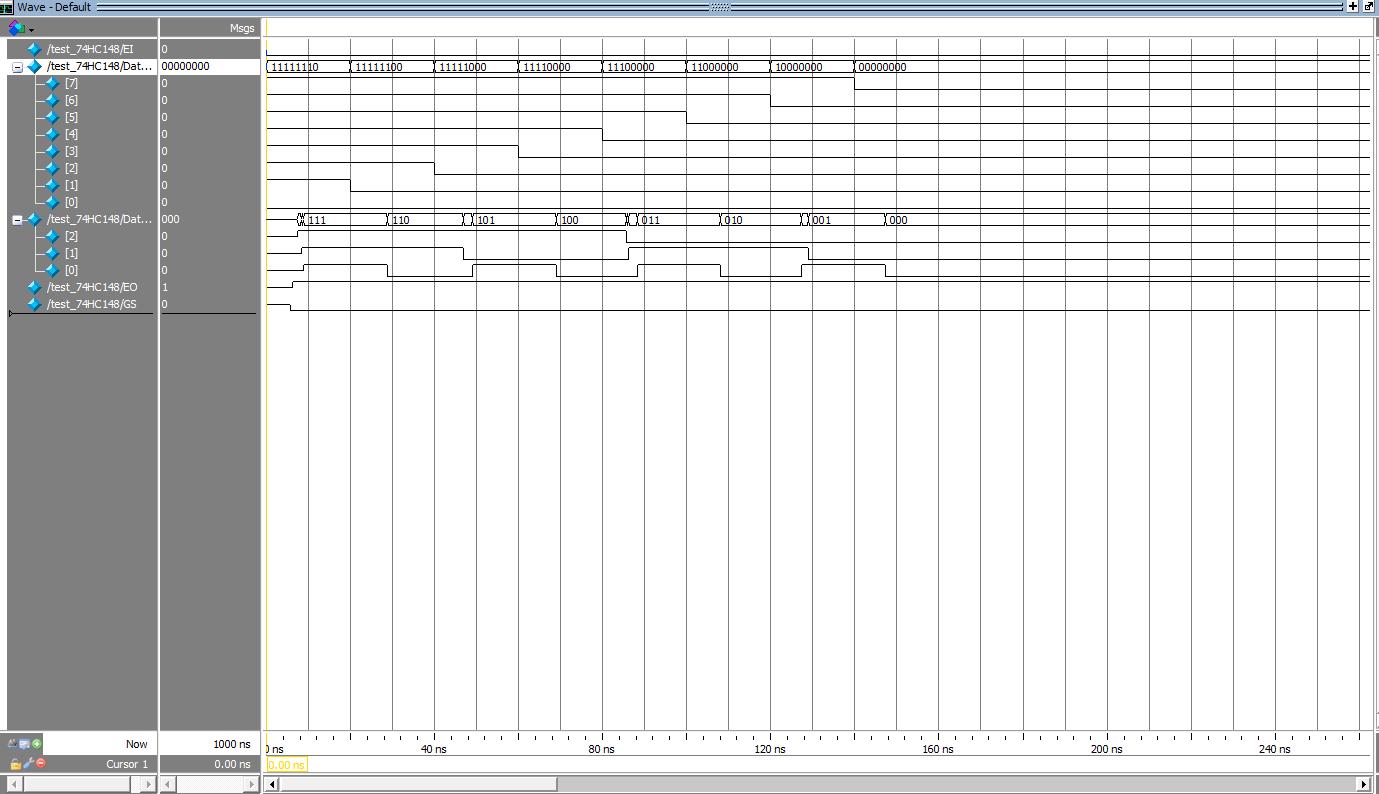


//74HC4511：信号有延迟约600ps，出现竞争冒险。

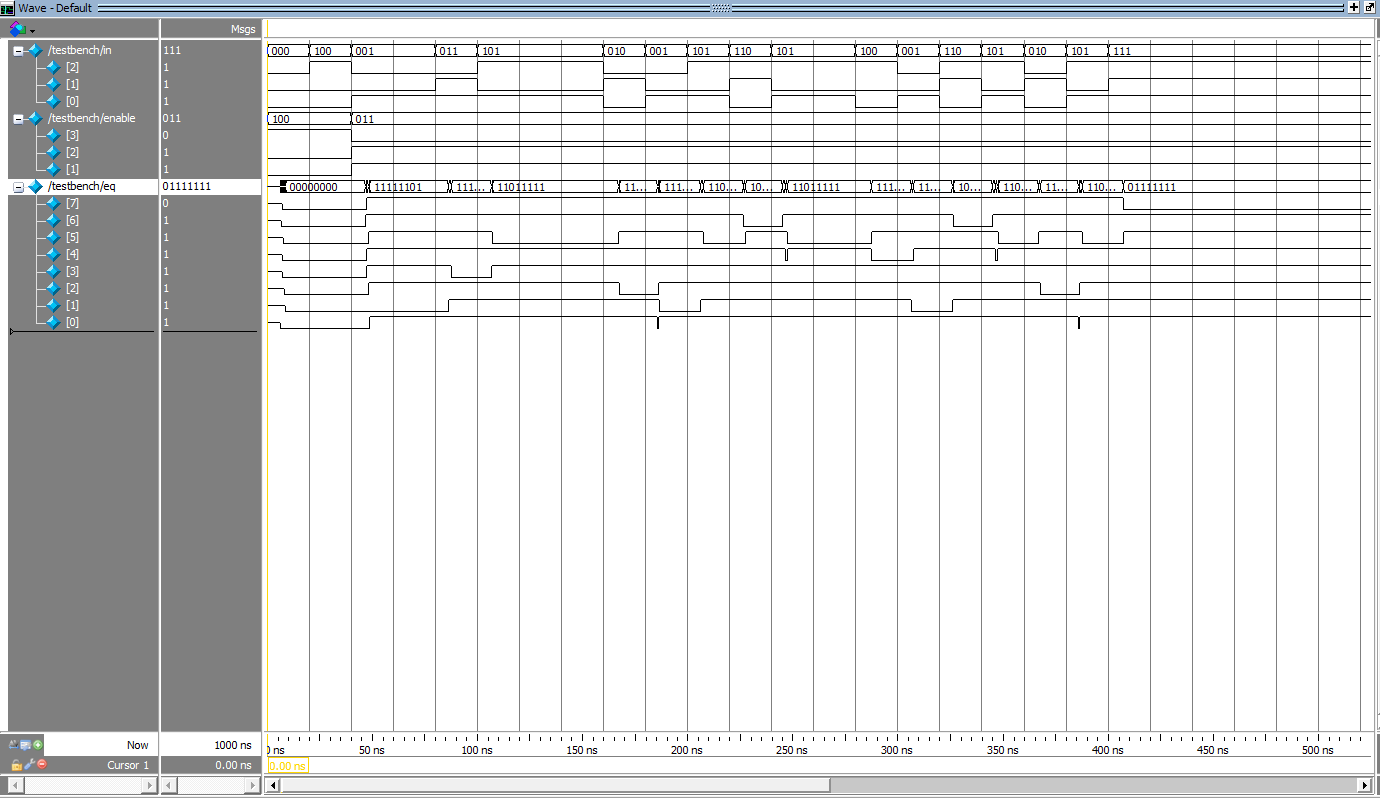


1. 第三次仿真结果（布局布线后）（**截图，注明对应的模块**）。回答输出信号是否有延迟，延迟时间约为多少？分析是否有出现竞争冒险。

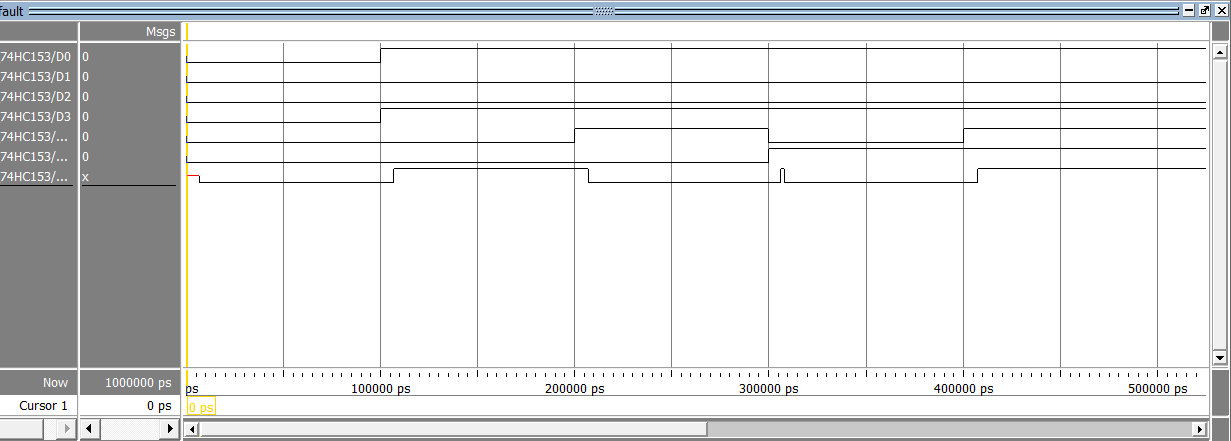
//74HC148：信号有延迟约1300ps，无竞争冒险



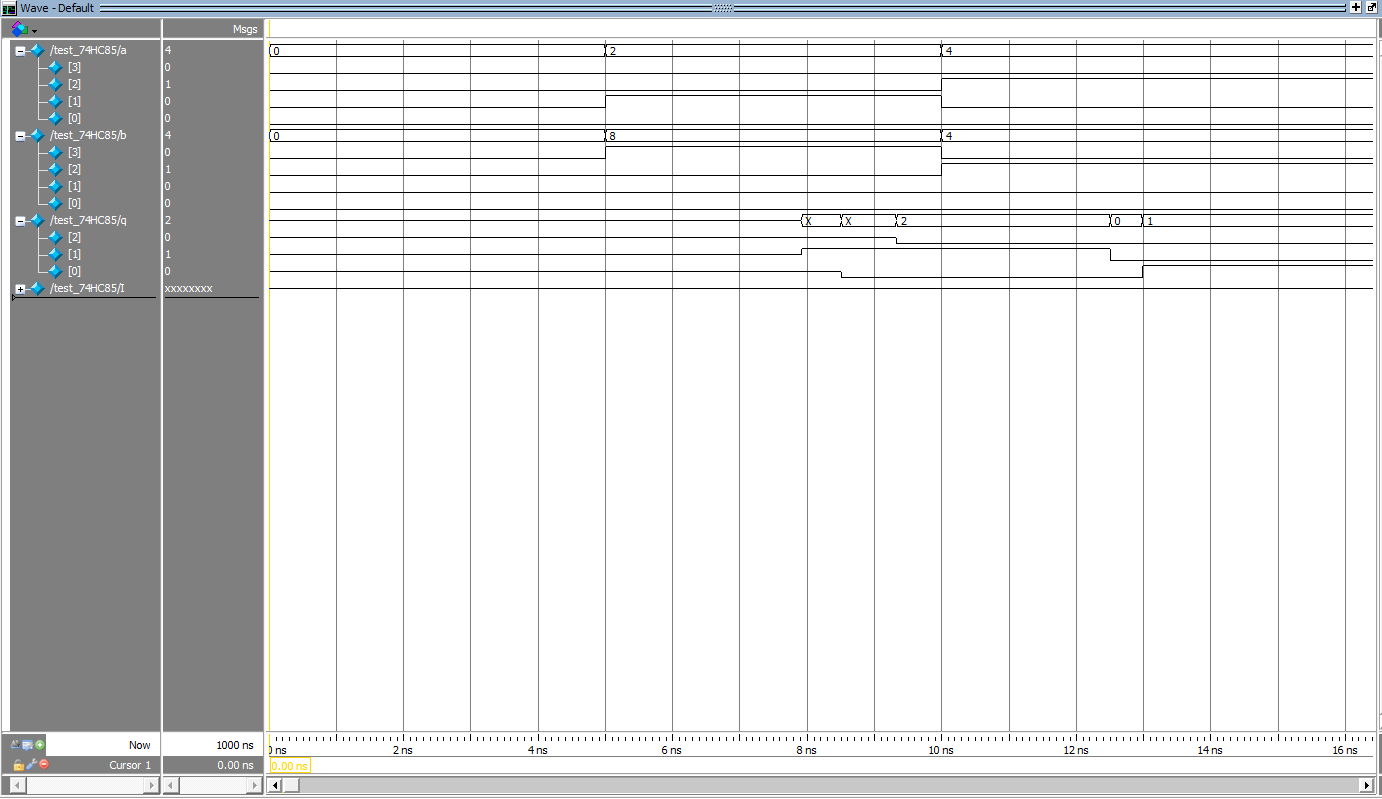
//74HC138：有延时，延时大概为4453ps，无竞争冒险 。



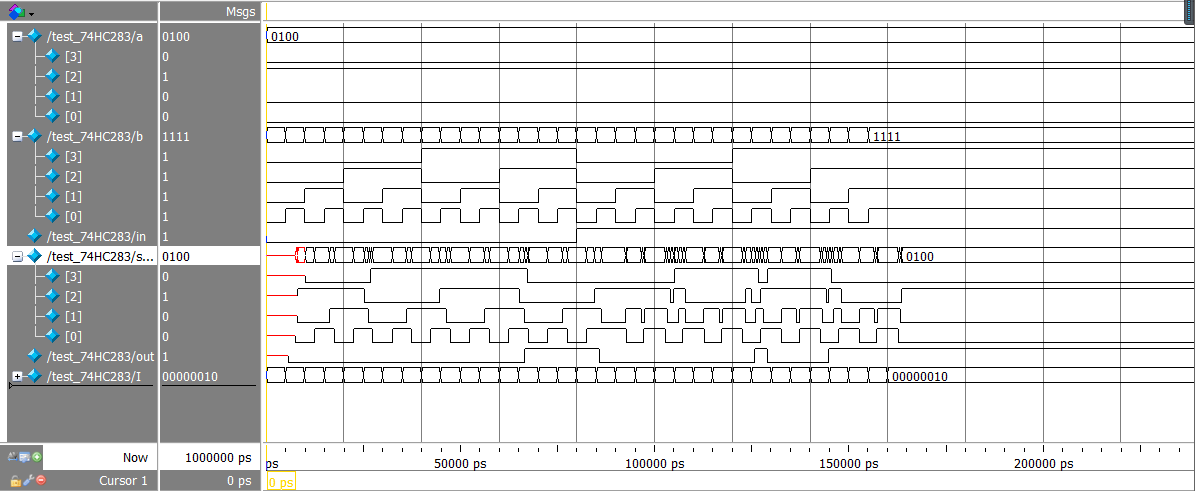
//74HC153：有延时，延时约10000ps, 有竞争冒险。



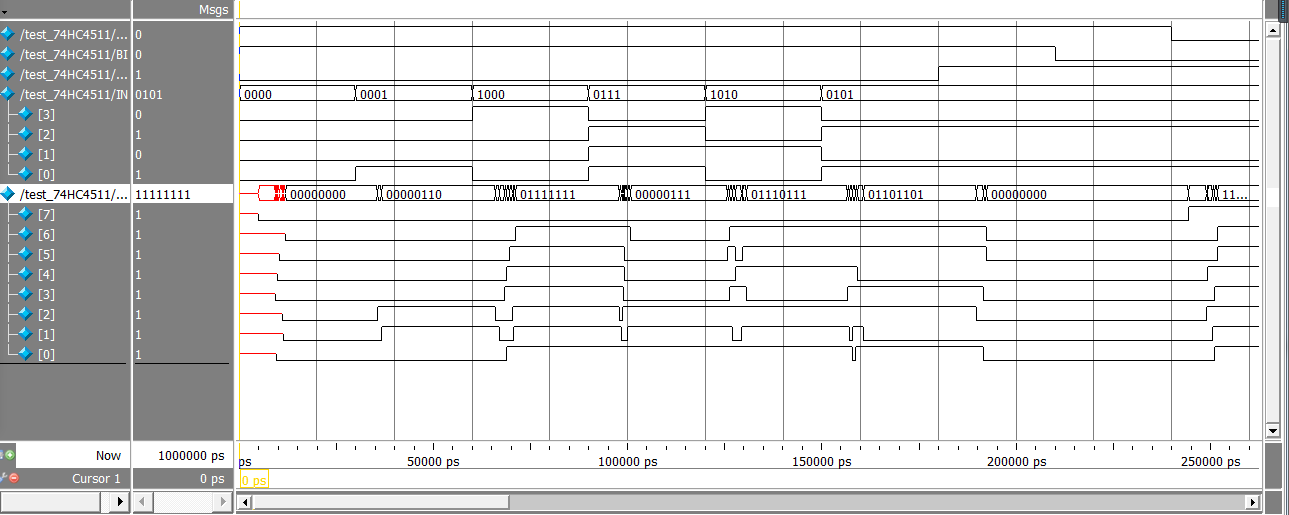
//74HC85：信号有延迟约550ps，无竞争冒险。



//74HC283：有延时，大概为470ps，无竞争冒险。



//74HC4511：信号有延迟约10416ps，无竞争冒险。



## 时序逻辑电路

一、实验目的

1、了解基于Verilog的时序逻辑电路的设计及其验证。

2、熟悉利用EDA工具进行设计及仿真的流程。

3、熟悉实验箱的使用和程序下载（烧录）及测试的方法。

4、学习针对实际时序逻辑电路芯片74HC74、74HC112、74HC194、74HC161进行VerilogHDL设计的方法。

二、实验环境及仪器

1、Libero仿真软件。

2、数字逻辑与系统设计实验箱。

3、Actel A3P060 FPGA芯片及Flash Pro5烧录器。

三、实验内容

1、熟练掌握Libero软件的使用方法。

2、进行针对74系列时序逻辑电路的设计，并完成相应的仿真实验。

3、参考教材中相应章节的设计代码、测试平台代码（可自行编程），完成74HC74、74HC112、74HC161、74HC194相应的设计、综合及仿真。

4、提交针对74HC74、74HC112、74HC161、74HC194的综合结果，以及相应的仿真结果。

四、实验结果和数据处理

1、所有模块及测试平台代码清单

//74HC74代码

module W\_74HC74(S,R,CLK,D,Q,Qn);

input S,R,CLK,D;

output Q,Qn;

reg Q,Qn;

always@(posedge CLK or negedge S or negedge R)

case({S,R})

0:{Q,Qn}<=2'b11;

1:{Q,Qn}<=2'b10;

2:{Q,Qn}<=2'b01;

default:{Q,Qn}<={D,~D};

endcase

endmodule

//74HC74测试平台代码

`timescale 1ns/100ps

module test\_74HC74;

reg s,r,clk,d;

wire q,qn;

W\_74HC74 u1(s,r,clk,d,q,qn);

task clock;

repeat(10)

begin

d=0;#3;d=1;#3;

end

endtask

always

begin

clk=0;#2;clk=1;#2;

end

initial

begin

s=0;r=1;clock;

#5 s=1;r=0;clock;

#5 s=0;r=0;clock;

#5 s=1;r=1;clock;

end

endmodule

//74HC112代码

module W\_74HC112(S,R,CLK,J,K,Q,Qn);

input S,R,CLK,J,K;

output Q,Qn;

reg Q,temp1,temp2;

assign Qn=~Q;

always@(negedge CLK)

begin

if(S&&R)

begin

case({J,K})

2'd0:temp1<=temp1;

2'd1:temp1<=0;

2'd2:temp1<=1;

2'd3:temp1<=~Q;

endcase

end

end

always@(S,R)

begin

case({S,R})

2'd0:temp2<=1;

2'd1:temp2<=1;

2'd2:temp2<=0;

2'd3:temp2<=temp2;

endcase

end

always@(temp1,temp2)

begin

if(S&&R)

Q<=temp1;

else

Q<=temp2;

end

endmodule

//74HC112测试平台代码

`timescale 1ns/1ns

module test\_74HC112;

reg clk,s,r,j,k;

wire q,qn;

always #33 clk=~clk;

W\_74HC112 u1(s,r,clk,j,k,q,qn);

initial

begin

clk=0;r=0;s=0;j=0;k=0;

#100 r=0;s=1;

#100 r=1;s=0;

#100 r=0;s=1;

#100 r=1;s=0;

#100 r=1;s=1;

#100 r=0;s=1;

#100 r=1;s=0;

#100 r=1;s=1;

#100 r=0;s=0;

#100 $finish;

end

initial $monitor("time=%t,clk=%b,j=%b,k=%b,q=%b,qn=%b",$time,clk,j,k,q,qn);

endmodule

//74HC161代码

module W\_74HC161(MR\_N,CP,CEP,CET,PE\_N,D,Q,TC);

input MR\_N,CP,CEP,CET,PE\_N;

input[3:0]D;

output [3:0]Q;

reg[3:0]Q;

output TC;

always@(negedge MR\_N or negedge CP)

if(!MR\_N)

Q=0;

else if(CEP&CET&PE\_N)

Q=Q+1;

else if(!PE\_N)

Q=D;

assign TC=&{CET,Q};

endmodule

//74HC161测试平台代码

`timescale 1ns/1ns

module test\_74HC161;

reg mr,cp,cep,cet,pe;

reg[3:0]d;

wire[3:0]q;

wire tc;

W\_74HC161 u1(mr,cp,cep,cet,pe,d,q,tc);

initial

begin

mr=1;

#100 mr=0;

#20 mr=1;

#200 mr=0;

#22 mr=1;

end

initial

begin

cp=0;

repeat(200)

#5 cp=~cp;

end

initial

begin

pe=1;

#152 pe=0;

#22 pe=1;

#200 pe=0;

#20 pe=1;

#200 pe=0;

#30 pe=1;

end

initial

begin

cep=1;

#222 cep=0;

#40 cep=1;

end

initial

begin

cet=1;

#333 cet=0;

#30 cet=1;

end

initial

begin

d=0;

repeat(20)

#41 d=$random;

end

initial $monitor("time=%t,mr=%b,cp=%b,cep=%b,cet=%b,pe=%b,d=%b,q=%b,tc=%b",$time,mr,cp,cep,cet,pe,d,q,tc);

endmodule

//74HC194代码

module W\_74HC194(MR\_N,S,D,CLK,IN,OUT);

input MR\_N,CLK;

input[1:0]S;

input[1:0]D;

input[0:3]IN;

output reg [0:3]OUT;

always@(posedge CLK or negedge MR\_N)

if(~MR\_N)

OUT<=0;

else

case(S)

2'b00:OUT<=OUT;

2'b01:if(D[1]) OUT<={1'b1,OUT[0:2]};

else OUT<=OUT>>1;

2'b10:if(D[0]) OUT<={OUT[1:3],1'b1};

else OUT<=OUT<<1;

default: OUT<=IN;

endcase

endmodule

//74HC194测试平台代码

`timescale 1ns/1ns

module test\_74HC194;

reg mr,clk;

reg[1:0]s;

reg[1:0]d;

reg[0:3]in;

wire[0:3]out;

W\_74HC194 u1(mr,s,d,clk,in,out);

always

#5 clk=~clk;

task s\_clock;

begin

d=0;

repeat(4)

#10 d=d+1;

end

endtask

task clock;

begin

s=2'b11;s\_clock;

s=2'b00;s\_clock;

s=2'b01;s\_clock;

s=2'b10;s\_clock;

end

endtask

initial

begin

clk=1;in=4'b0110;clock;

end

initial

begin

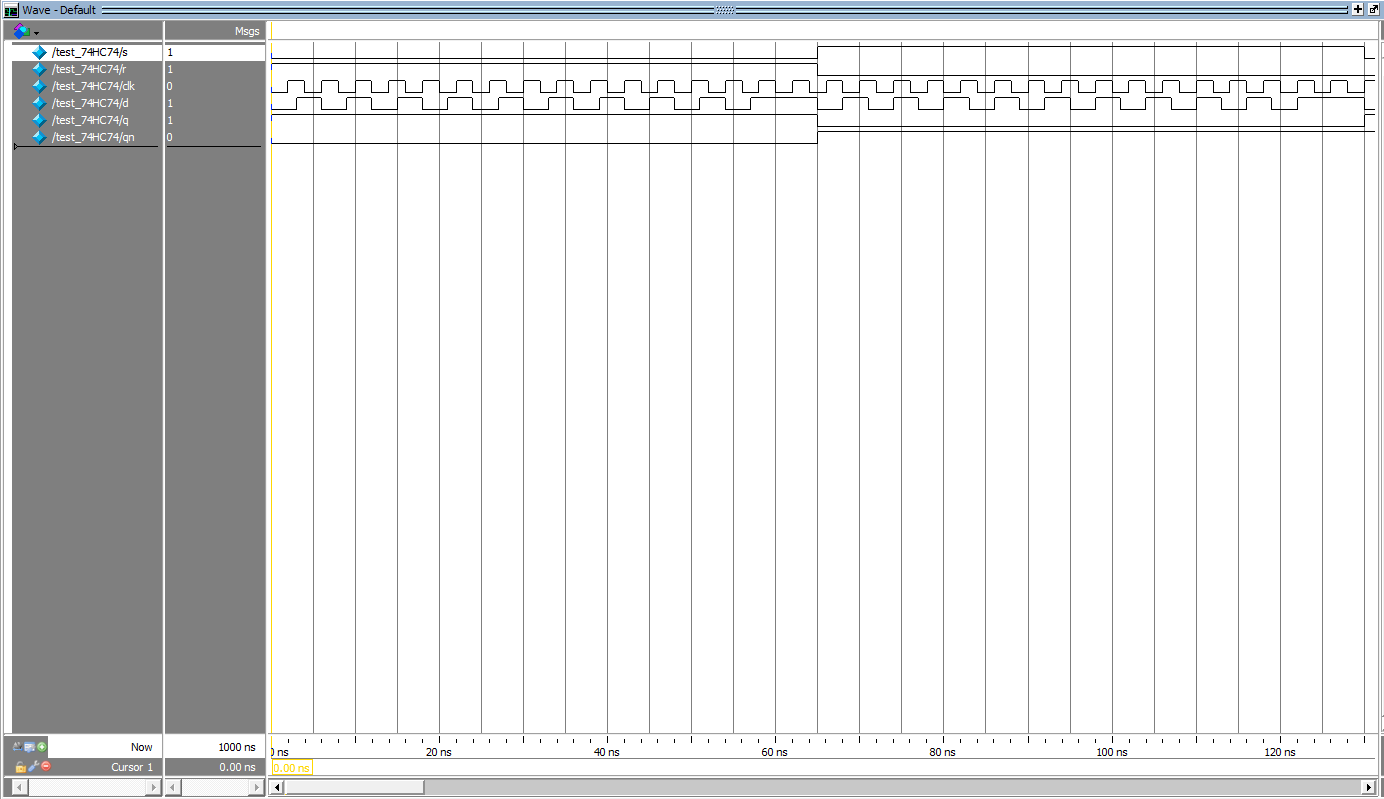
mr=1;#5 mr=0;#10mr=1;

end

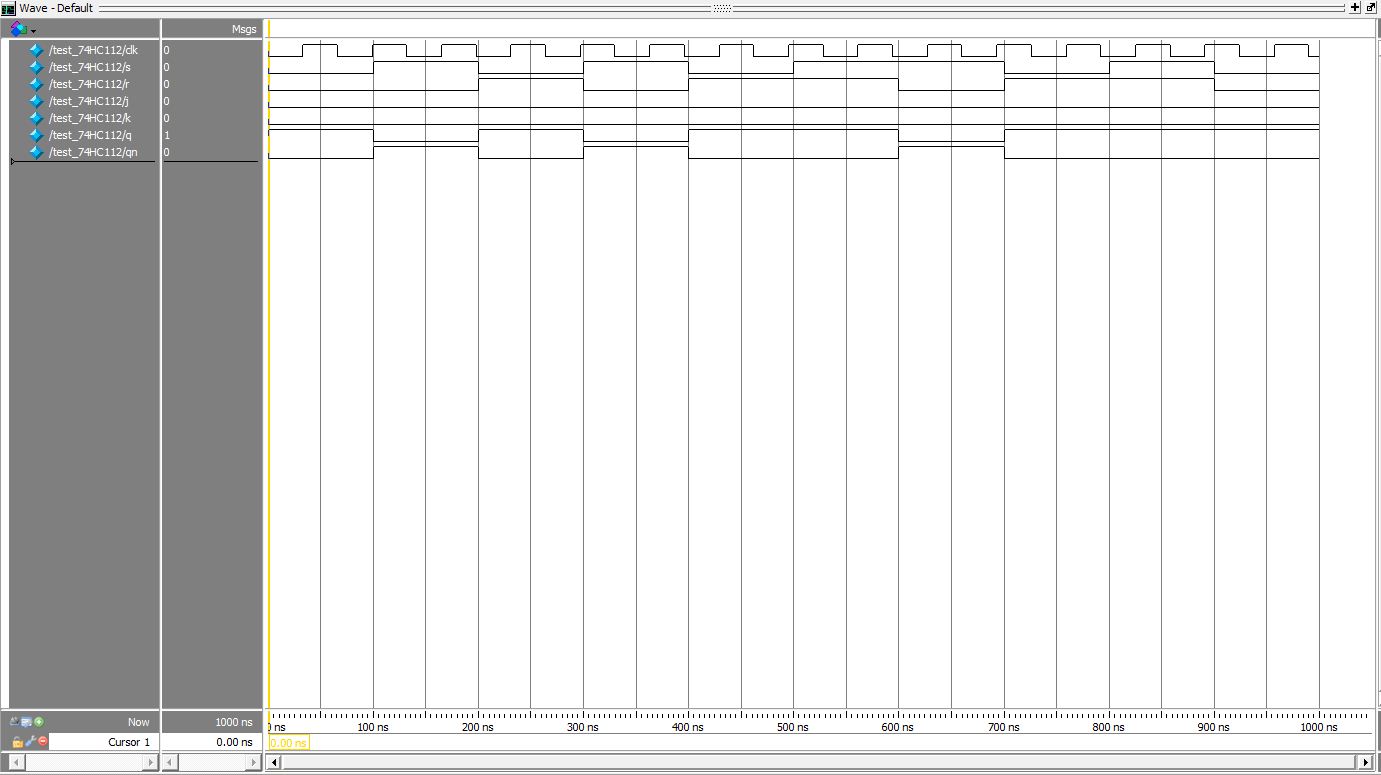
endmodule

2、第一次仿真结果（**截图，注明对应的模块**）

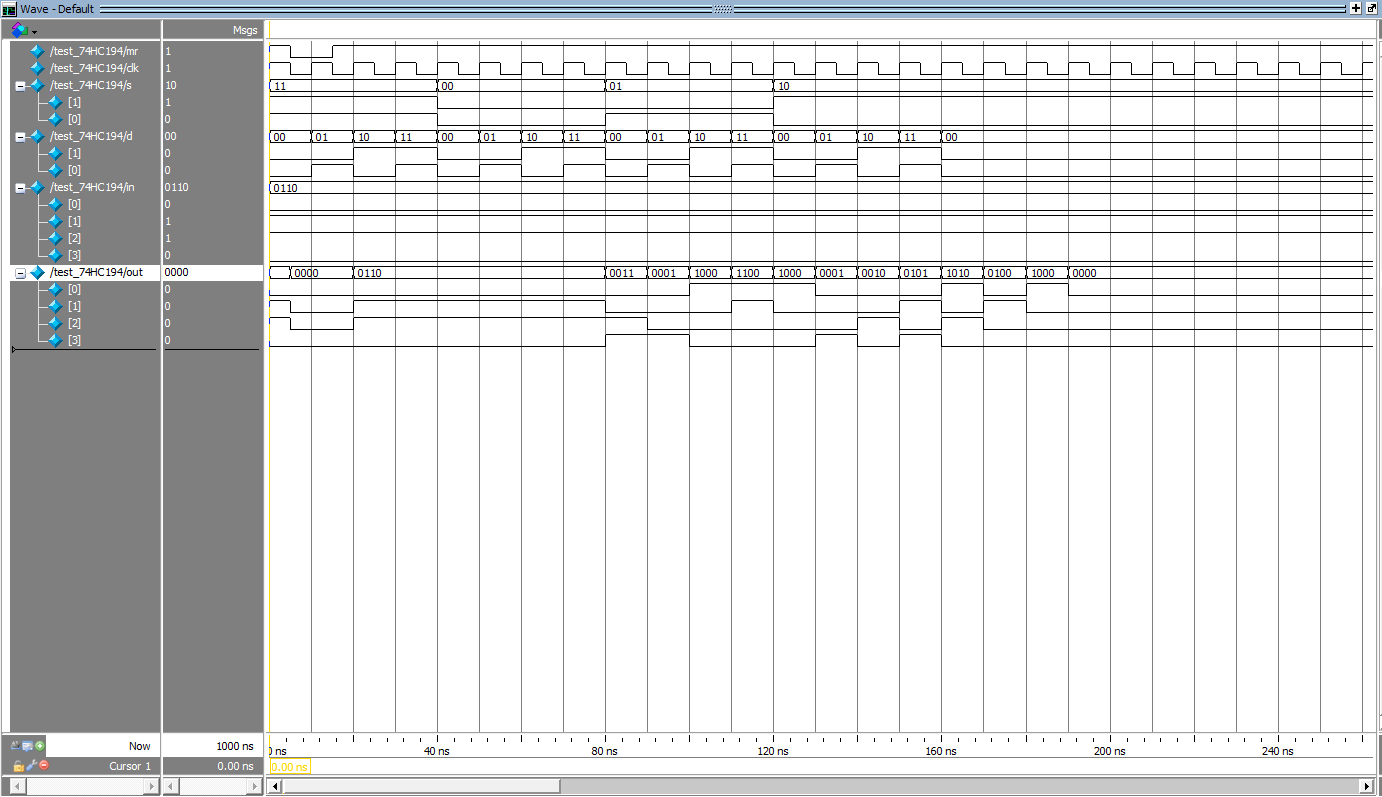
//74HC74:



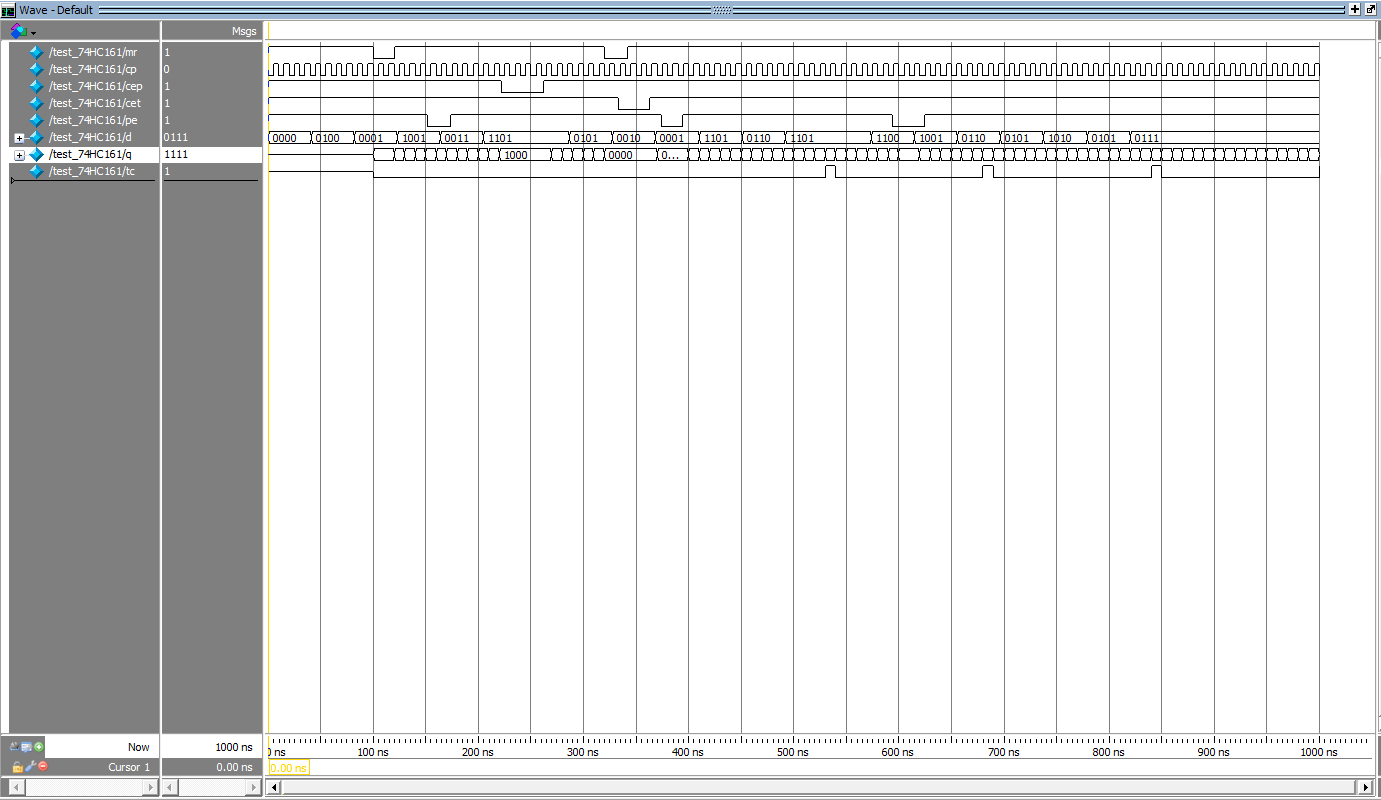
//74HC112:



//74HC194:

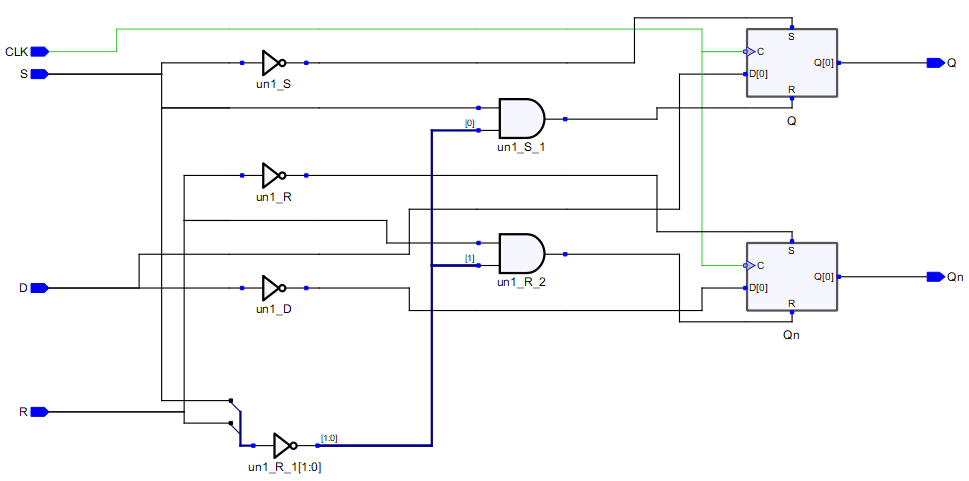


//74HC161:

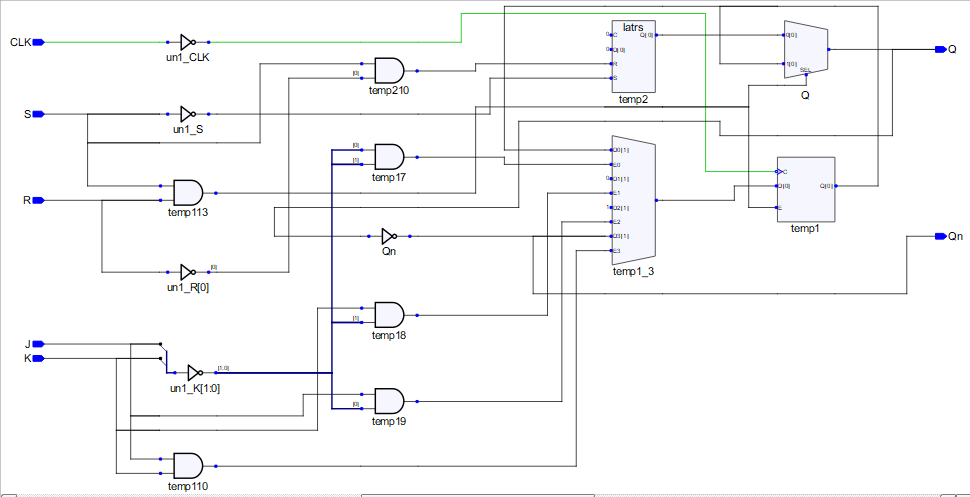


3、综合结果（**截图，注明对应的模块**）

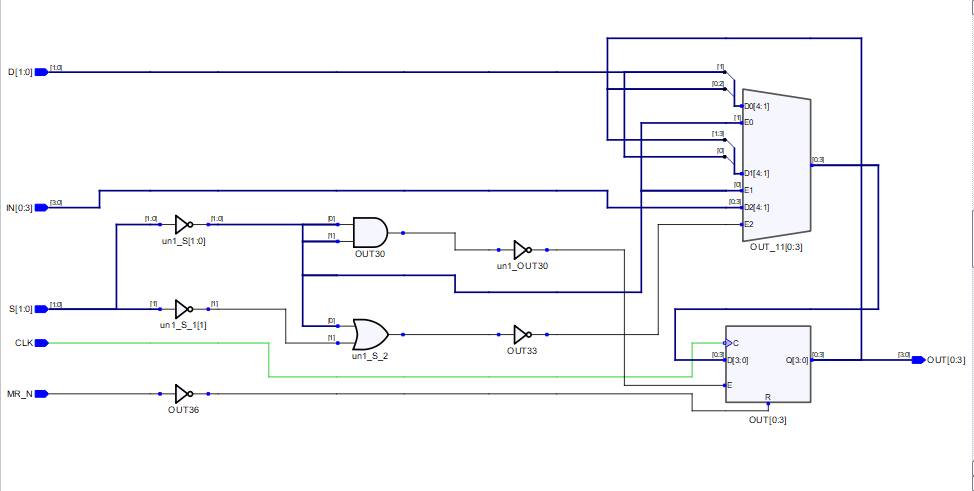
//74HC74:



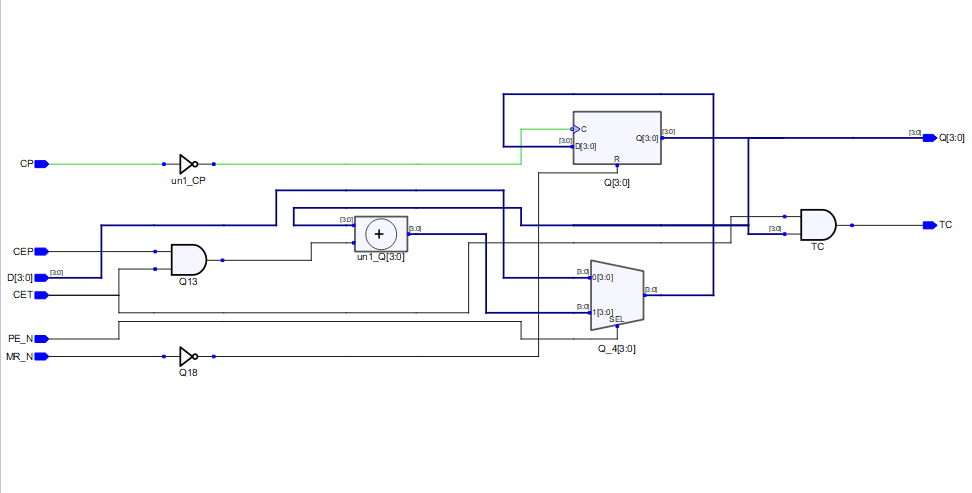
//74HC112:



//74HC194:

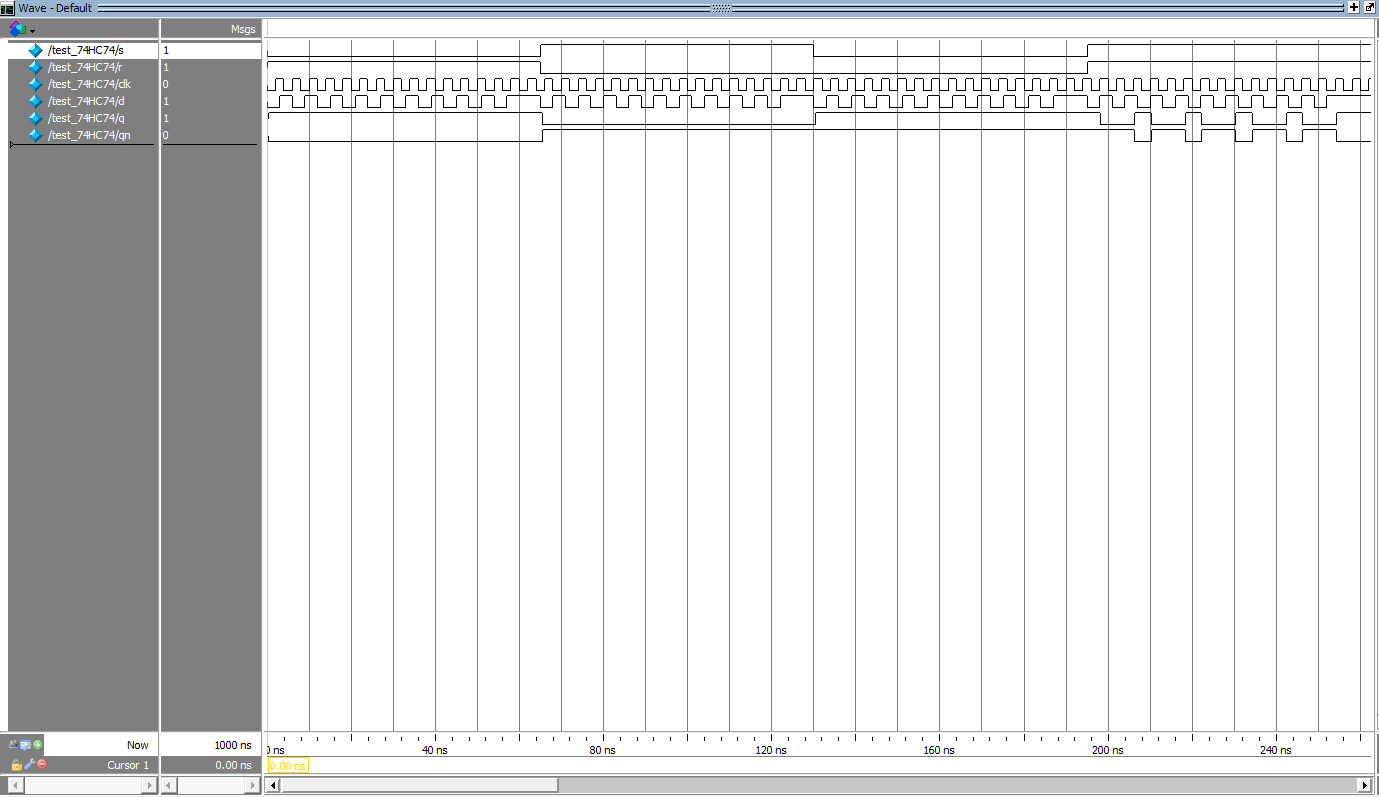


//74HC161:

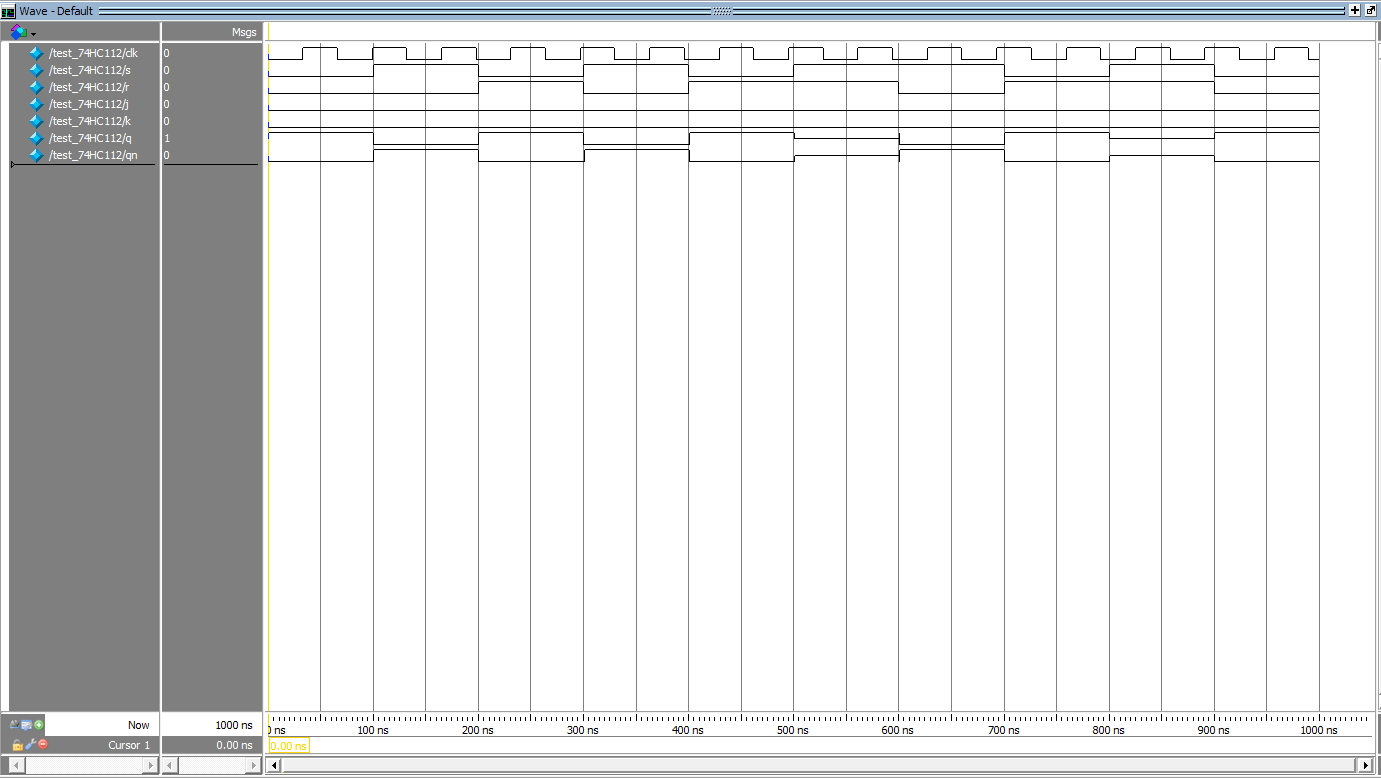


4、第二次仿真结果（综合后）（**截图，注明对应的模块**）。

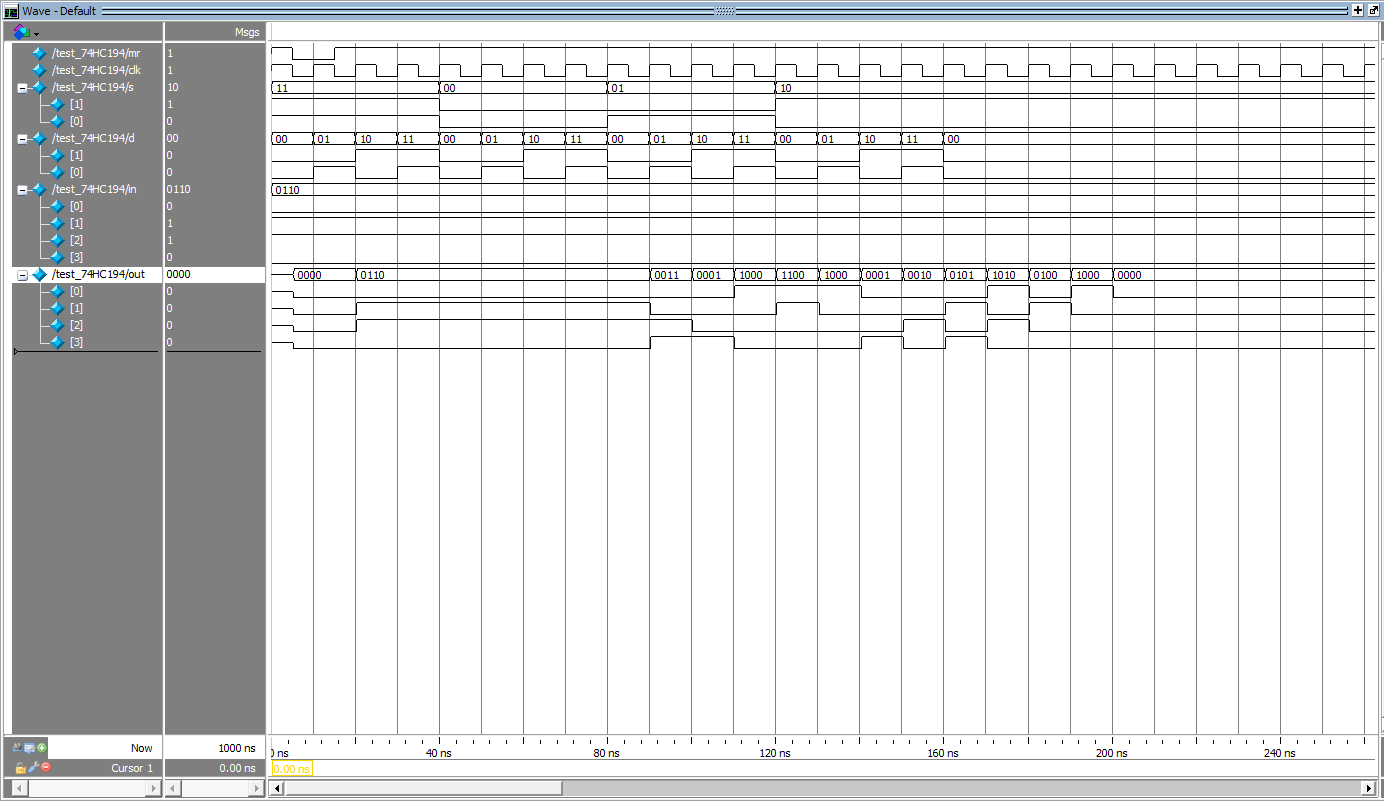
//74HC74:有延时，延时大概为1000ps。



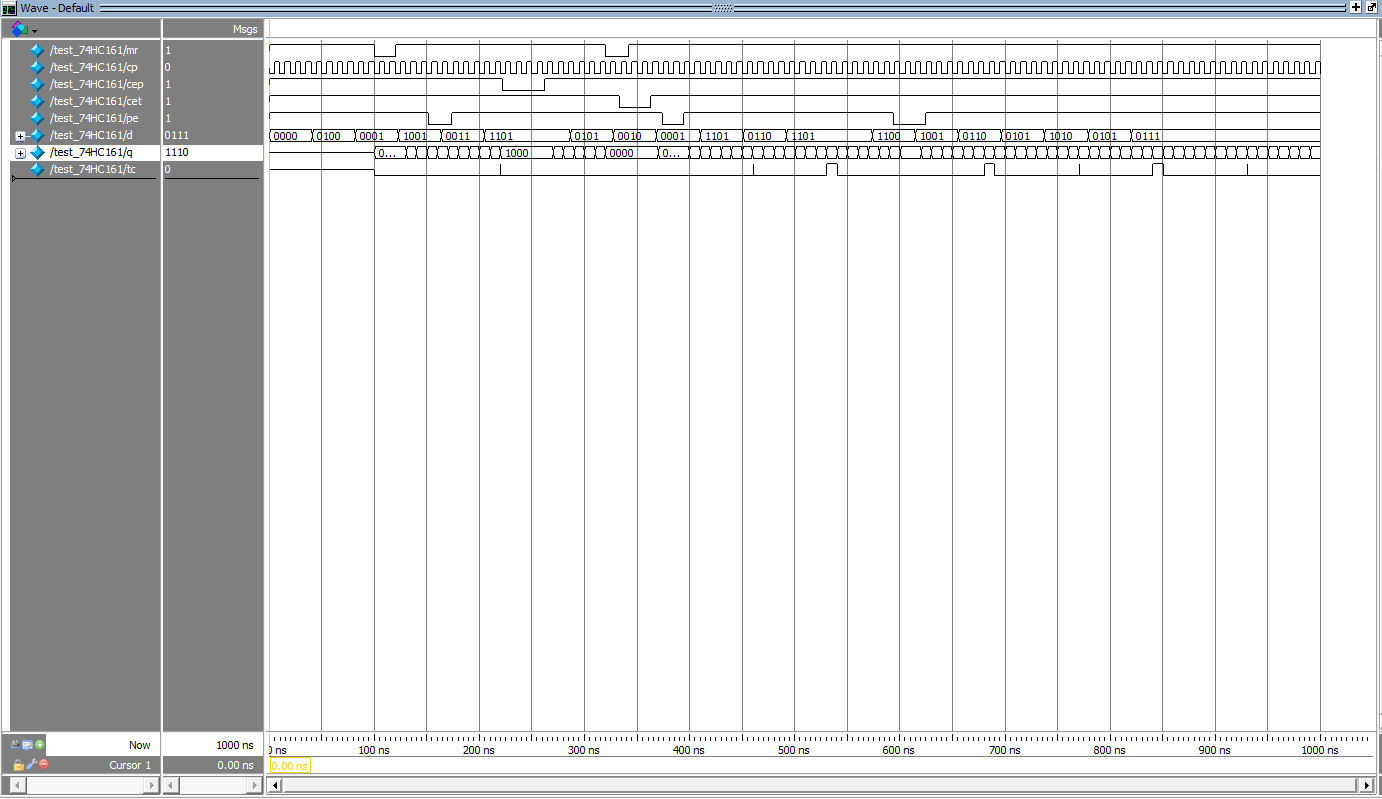
//74HC112:有延时，延时约为400ps。



//74HC194:有延时，延时时间大概为300ps。

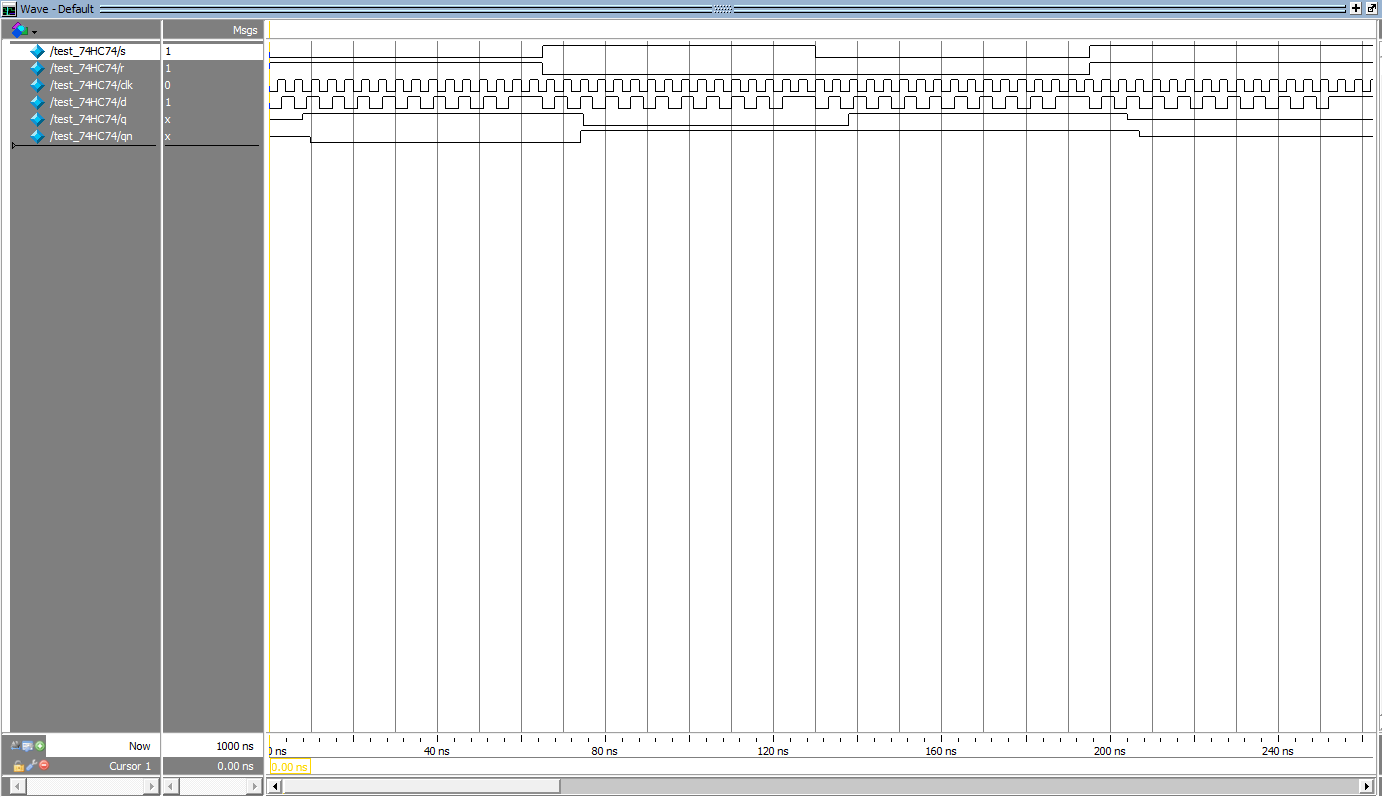


//74HC161:信号有延迟 约400ps。

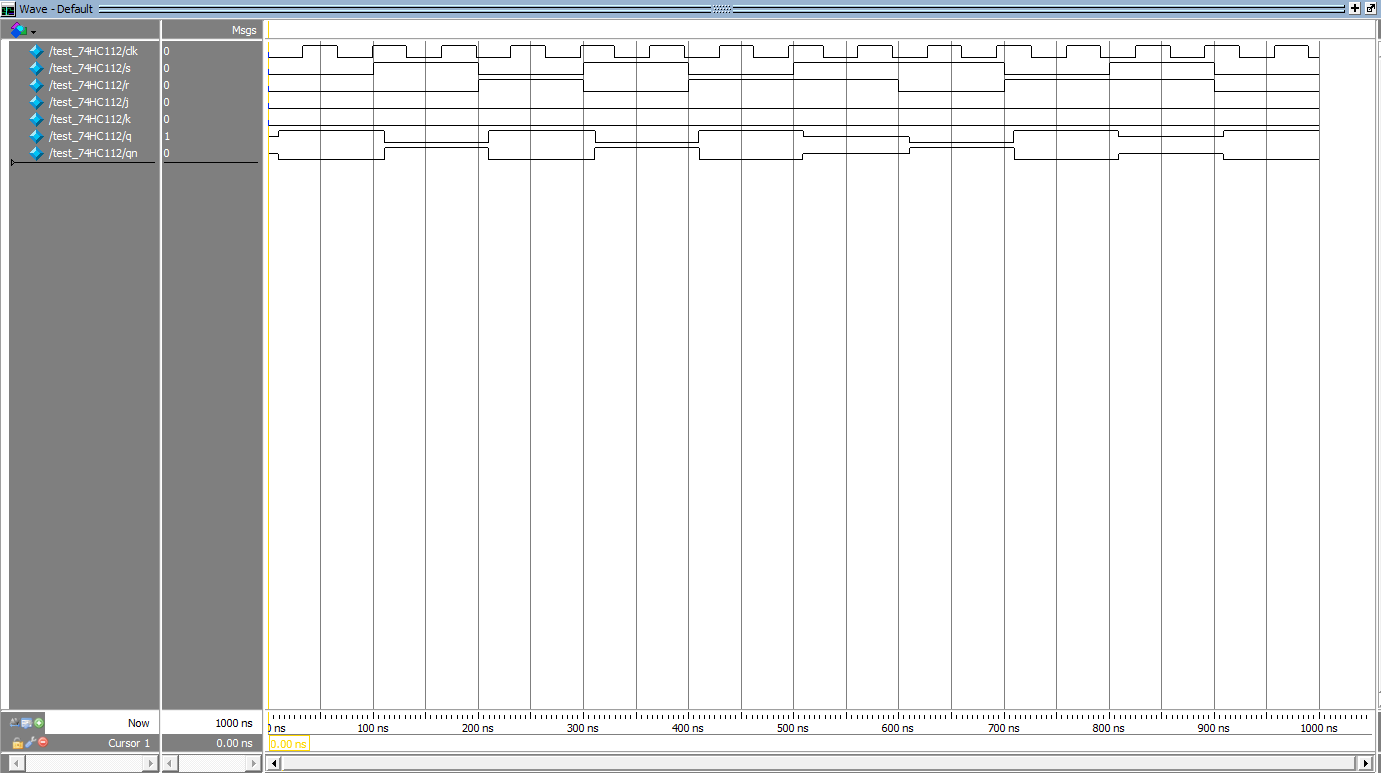


1. 第三次仿真结果（布局布线后）（**截图，注明对应的模块**）。

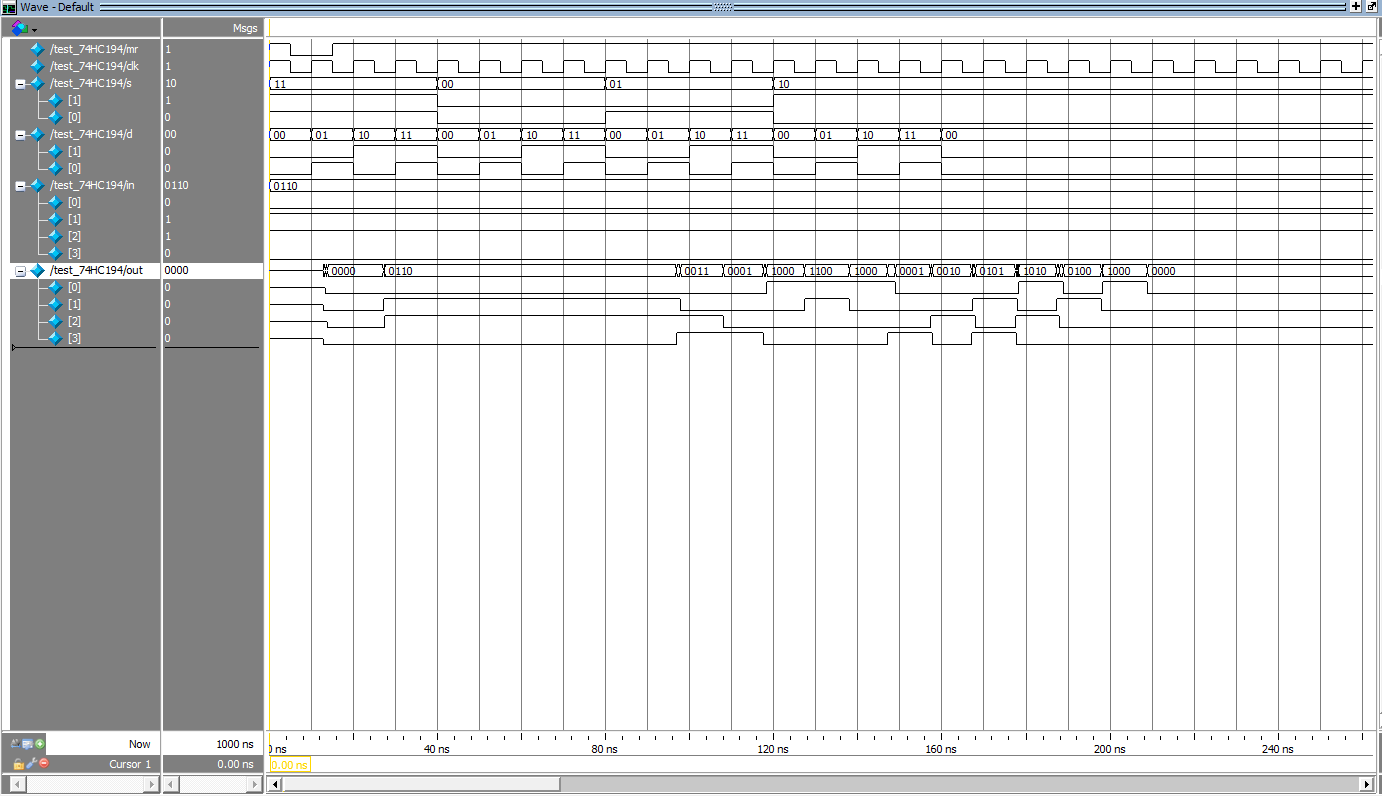
//74HC74:有延时，延时大概为1000ps。无竞争冒险。



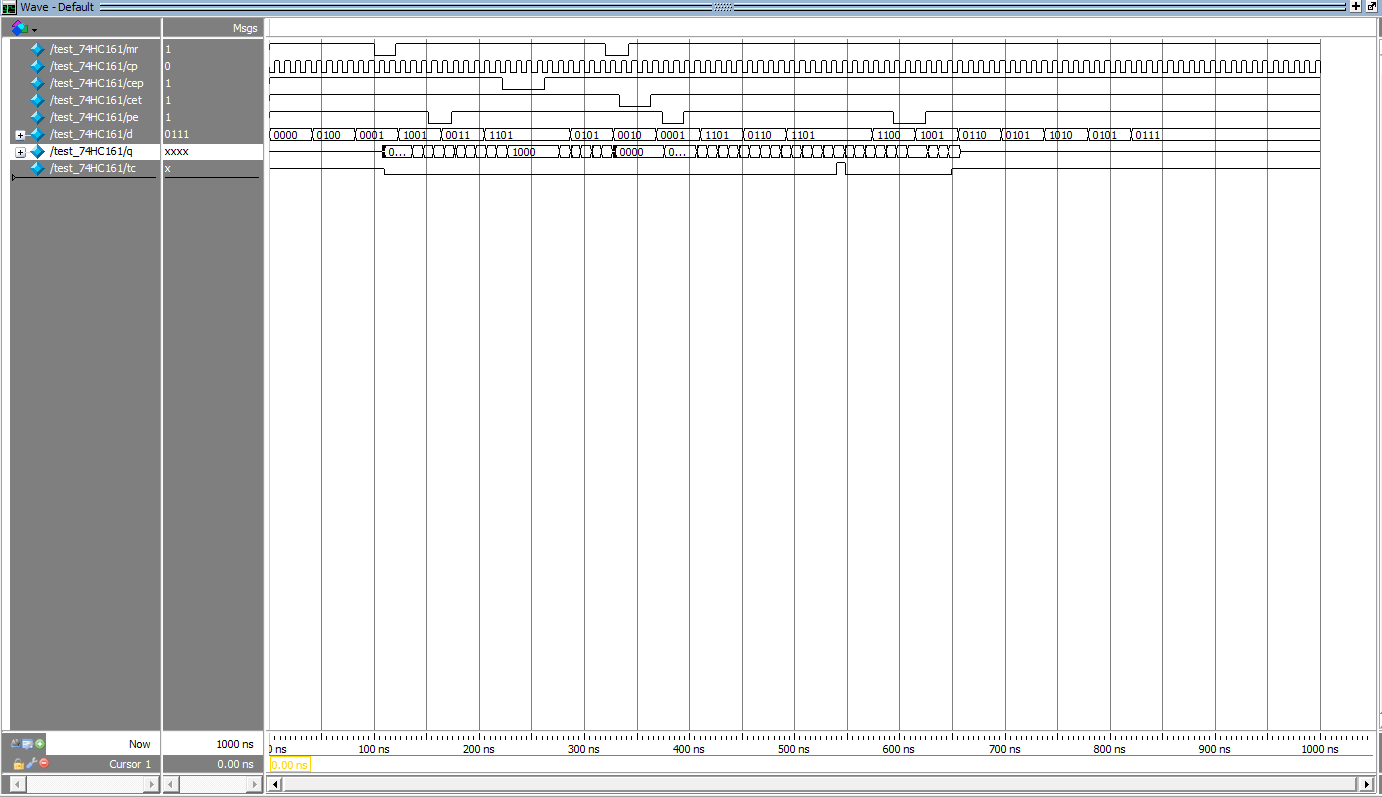
//74HC112:有延时，延时约为900ps，无竞争冒险。



//74HC194:有延时，延时时间大概为670ps，没有竞争冒险



//74HC161:信号有延迟，约436ps，无竞争冒险。



## 综合实验

一、实验目的

1、进一步熟悉利用EDA工具进行设计及仿真的流程。

2、熟悉利用EDA工具中的图形化设计界面进行综合设计。

3、熟悉芯片烧录的流程及步骤。

4、掌握分析问题、解决问题的综合能力，通过EDA工具设计出能解决实际问题的电路。

二、实验环境

1、Libero仿真软件。

2、数字逻辑与系统设计实验箱。

3、Actel A3P060 FPGA芯片及Flash Pro5烧录器。

三、实验内容

完成以下综合实验1-7，可选做8-11，并按要求完成实验报告。

综合实验1——编码器扩展实验：当按下小于等于9的按键后，显示数码管显示数字，当按下大于9的按键后，显示数码管不显示数字。若同时按下几个按键，优先级别的顺序是9到0。本实验需要两个编码器74HC148、一个数码显示译码器74HC4511、一个共阴极8段显示数码管LN3461Ax和一个数值比较器74HC85。利用Libero SmartDesign图形化设计工具，采用图文混合设计方法进行设计。

综合实验2——译码器扩展实验：设计一个电路，通过改变输入，令显示数码管的4个数位轮流显示数字。本实验需要一个3-8译码器74HC138、一个数码显示译码器74HC4511、一个共阴极8段显示数码管LN3461Ax。将译码器74HC138的输入显示在数码管LN3461Ax上，并利用译码器74HC138的输出控制数码显示译码器74HC4511的工作（、或LE中任一个）。

综合实验3——有符号比较器实验：设计一个电路，比较两个8位有符号数的大小，判定是否满足大于等于的关系。方法：直接利用Libero工具提供的比较器IP核，实现一个有符号比较器。

综合实验4——4位有符号二进制加法器。

综合实验5——二——十进制码转换电路：设计一个能实现8位二进制码转换为12位8421BCD码的电路。

综合实验6——利用状态机实现一个简单自动售货机控制电路（顶层结构框图如图7-23）。该电路有两个投币口（1元和5角），商品2元一件，不设找零。In[0]表示投入5角，In[1]表示投入1元，D\_out表示是否提供货品。分别用Mealy和Moore型实现。

综合实验7——与寄存器结合的有限状态机：将寄存器逻辑（利用时钟信号同步进行赋值）与Mealy或Moore状态机组合起来，可以得出两种解决方案。

综合实验8——跑马灯：设计要求：共8个LED灯连成一排，用以下3种模式来显示，模式选择使用两个按键进行控制。

① 模式1：先点亮奇数灯，即1、3、5、7灯亮，然后偶数灯，即2、4、6、8灯亮，依次循环，灯亮的时间按时钟信号的二分频设计。

② 模式2：按照1、2、3、4、5、6、7、8的顺序依次点亮所有灯；然后再按1、2、3、4、5、6、7、8的顺序依次熄灭所有灯，间隔时间按时钟信号的八分频设计。

③ 模式3：按照1/8、2/7、3/6、4/5的顺序依次点亮所有灯，每次同时点亮两个灯；然后再按1/8、2/7、3/6、4/5的顺序熄灭相应灯，每次同时熄灭两个灯，灯亮的时间按时钟信号的四分频设计。

④ 模式4：自行设计。

综合实验9——四位数码管扫描显示电路的设计

设计要求：共4个数码管，连成一排，要求可以显示其中任意一个数码管。具体要求如下：

① 依次选通4个数码管，并让每个数码管显示不同的值。

② 要求能在实验箱上演示出数码管的动态显示过程。必须使得4个选通信号COM1、COM2、COM3、COM4轮流被单独选通，同时，在段信号输入口加上希望在对应数码管上显示的数据，这样随着选通信号的变化，才能实现扫描显示的目的。

综合实验10——交通灯控制器

设计要求：实现一个常见的十字路口交通灯控制功能。一个十字路口的交通灯一般分为两个方向，每个方向具有红灯、绿灯和黄灯三种。实现一个常见的十字路口交通灯控制功能，具体要求如下：

① 十字路口包含A、B两个方向的车道。A方向放行一分钟（绿55秒，黄5秒），同时B方向禁行（红60秒）；然后A方向禁行1分钟（红60秒），同时B方向放行（绿55秒，黄5秒），示意图如图5-56所示。依此类推，循环往复。

② 实现正常的倒计时功能，用两组数码管作为A和B两个方向的倒计时显示。

③ 系统时钟1KHz。

综合实验11——键盘扫描器和编码器

设计要求：

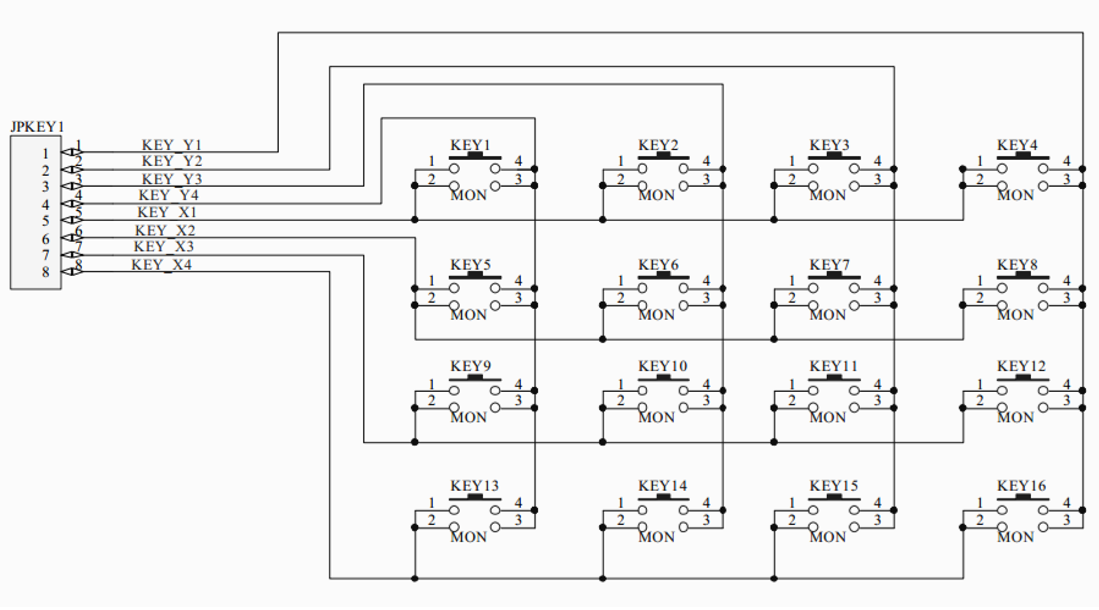
① 检测是否有按键按下；

② 确定按键的位置；

③ 生成唯一按键编码；

④ 在数码管上显示相应的按键内容。

原理图如下：

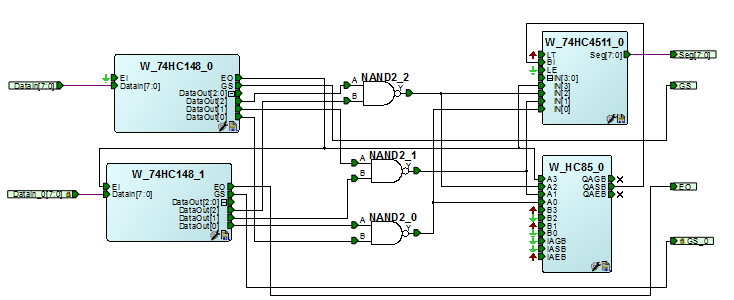


四、实验结果和数据处理

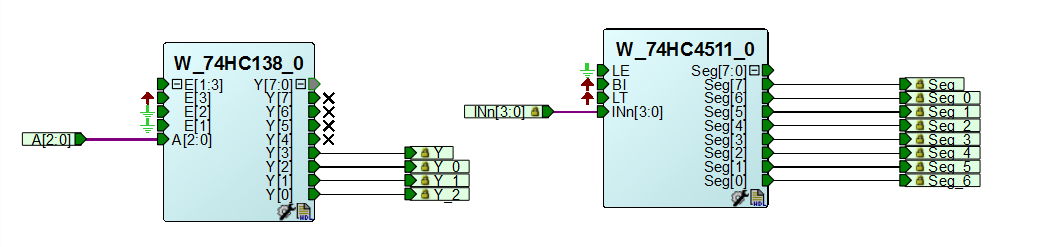
每个实验均要记录以下内容。

（1）SmartDesign的连线图

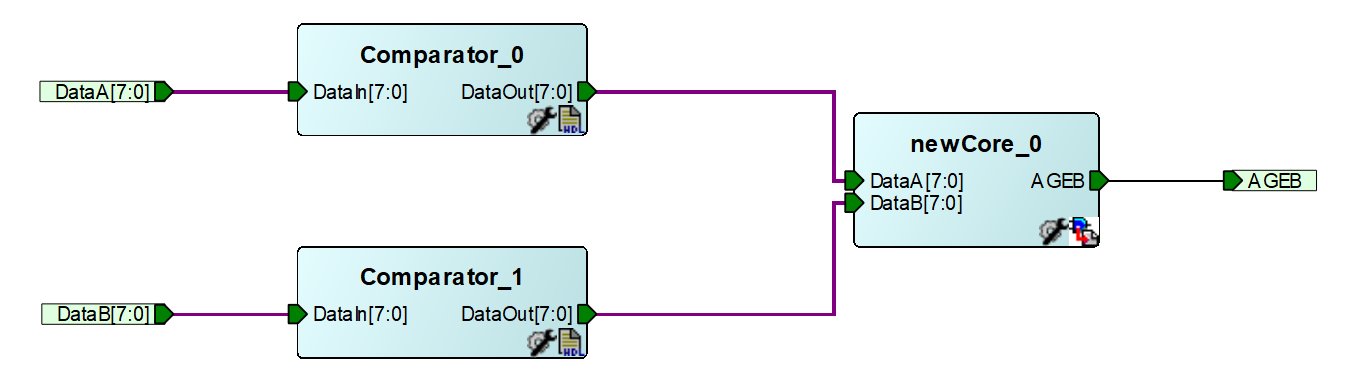
//综合实验一：



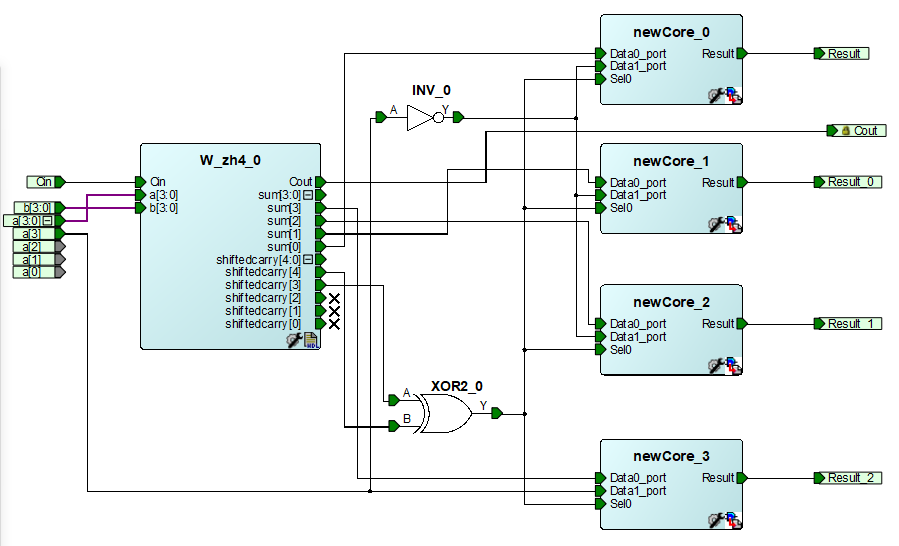
//综合实验二：



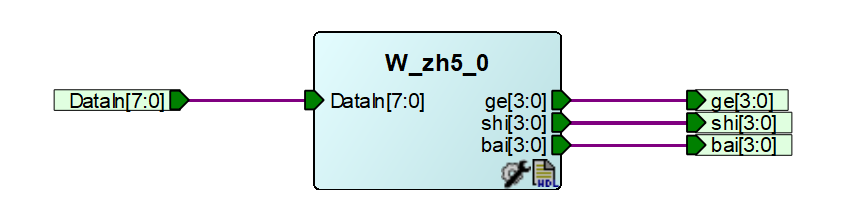
//综合实验三：



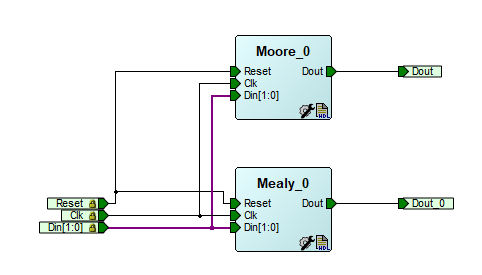
//综合实验四：



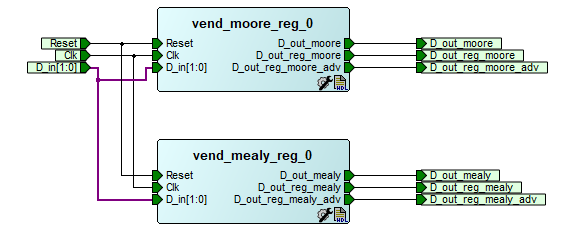
//综合实验五：



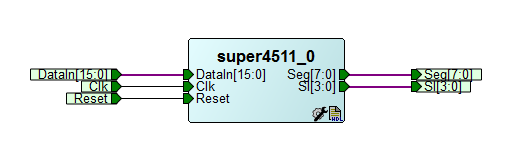
//综合实验六：



//综合实验七：

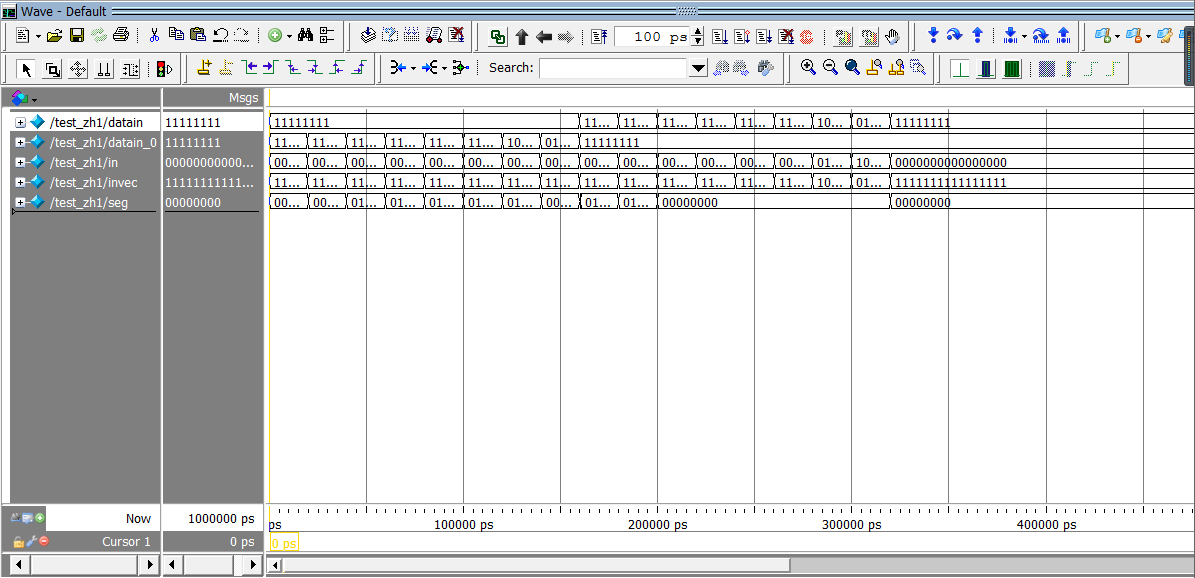


//综合实验九：

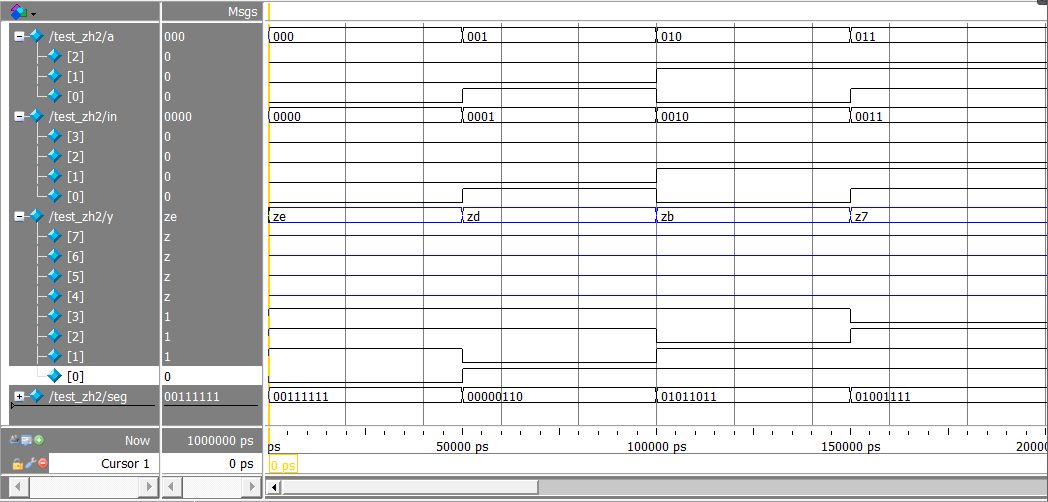


（2）功能仿真波形图

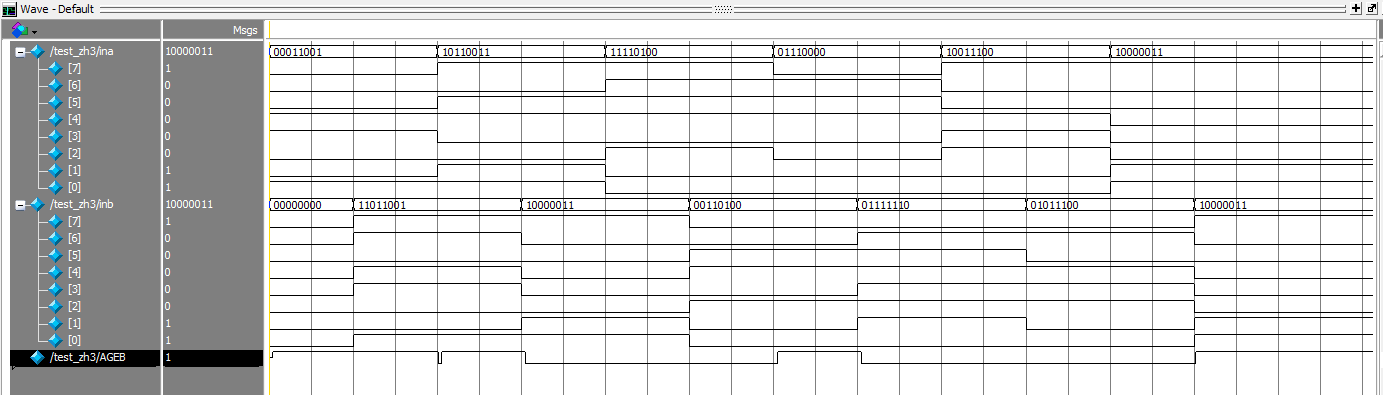
//综合实验一：



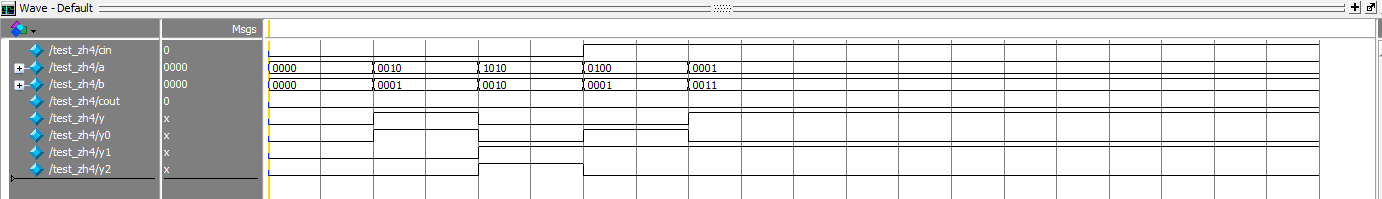
//综合实验二：



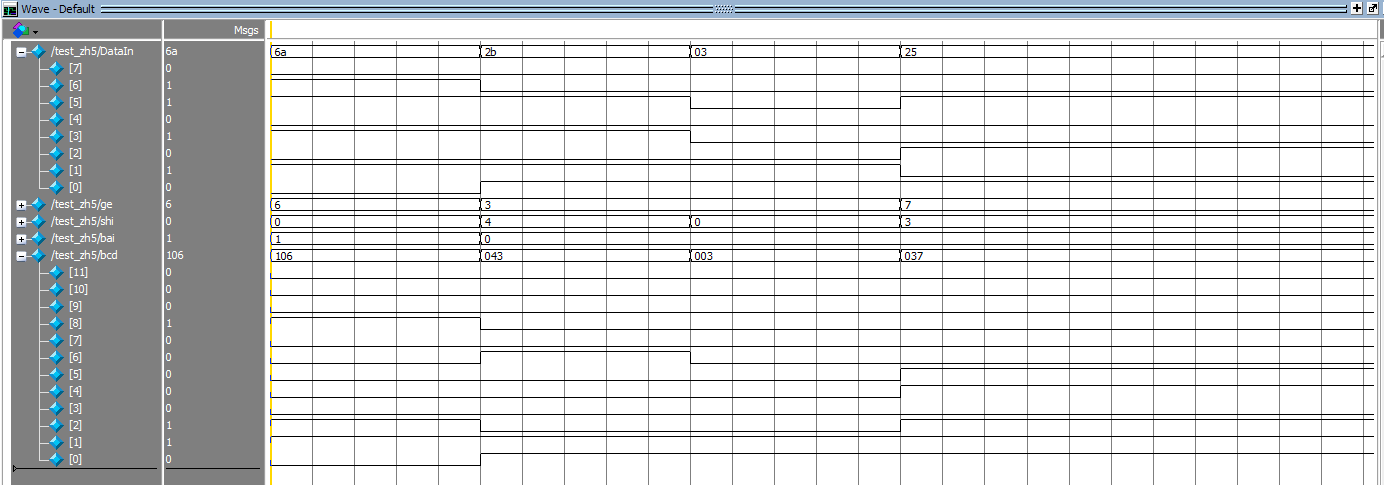
//综合实验三：



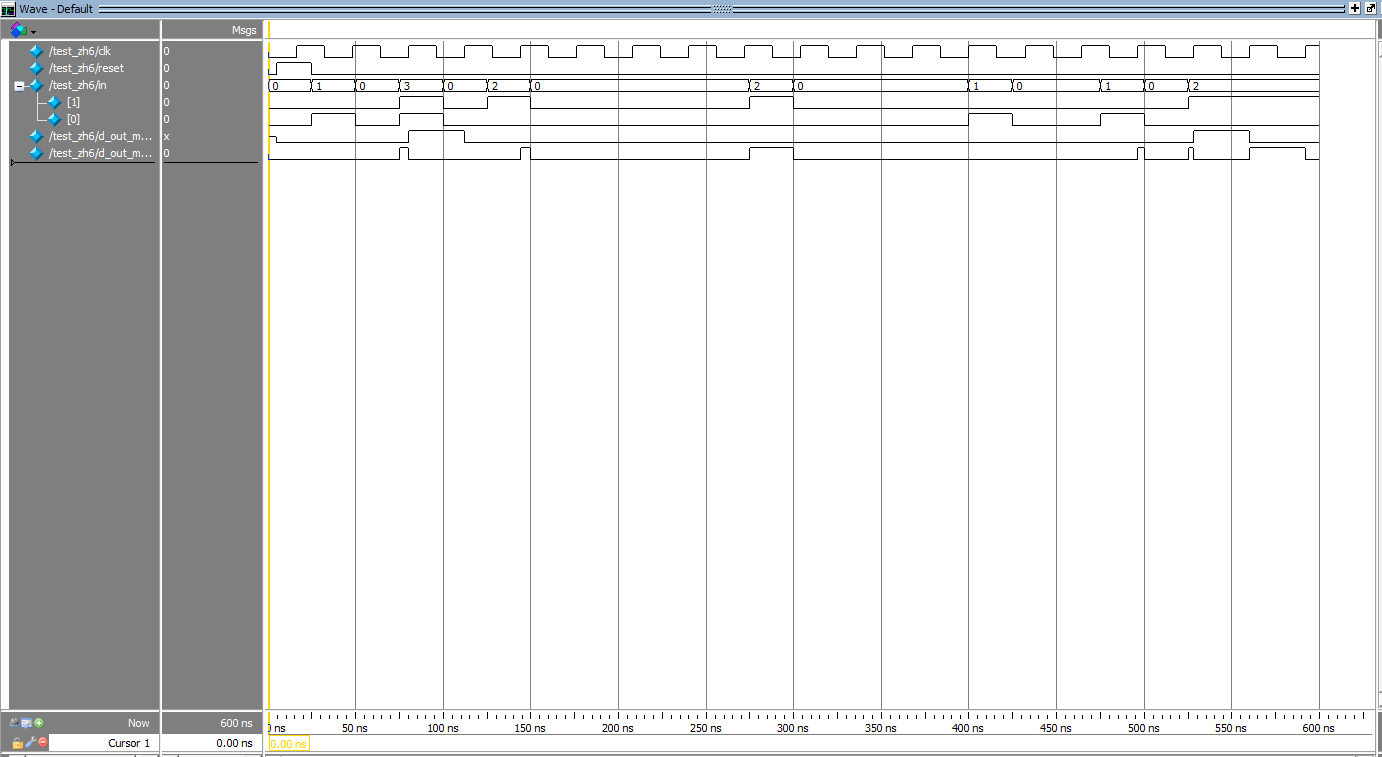
//综合实验四：



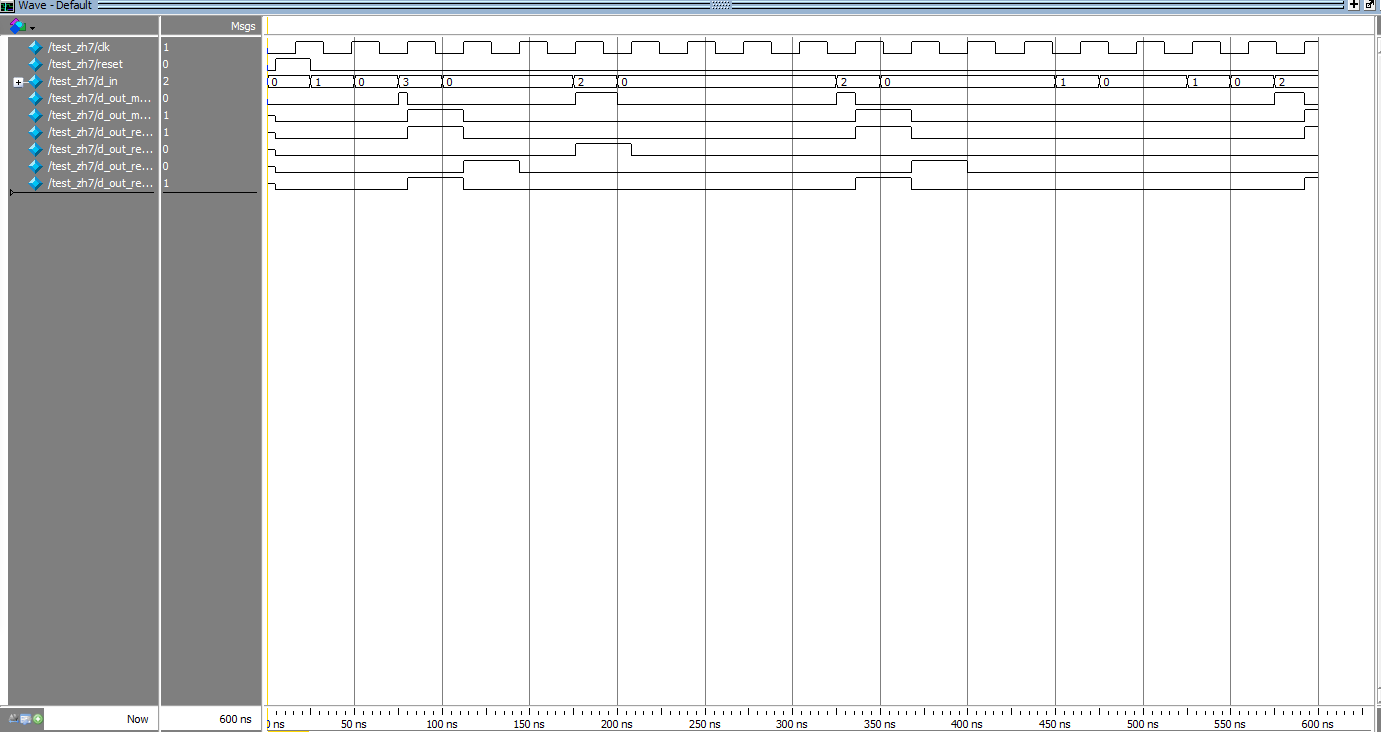
//综合实验五：



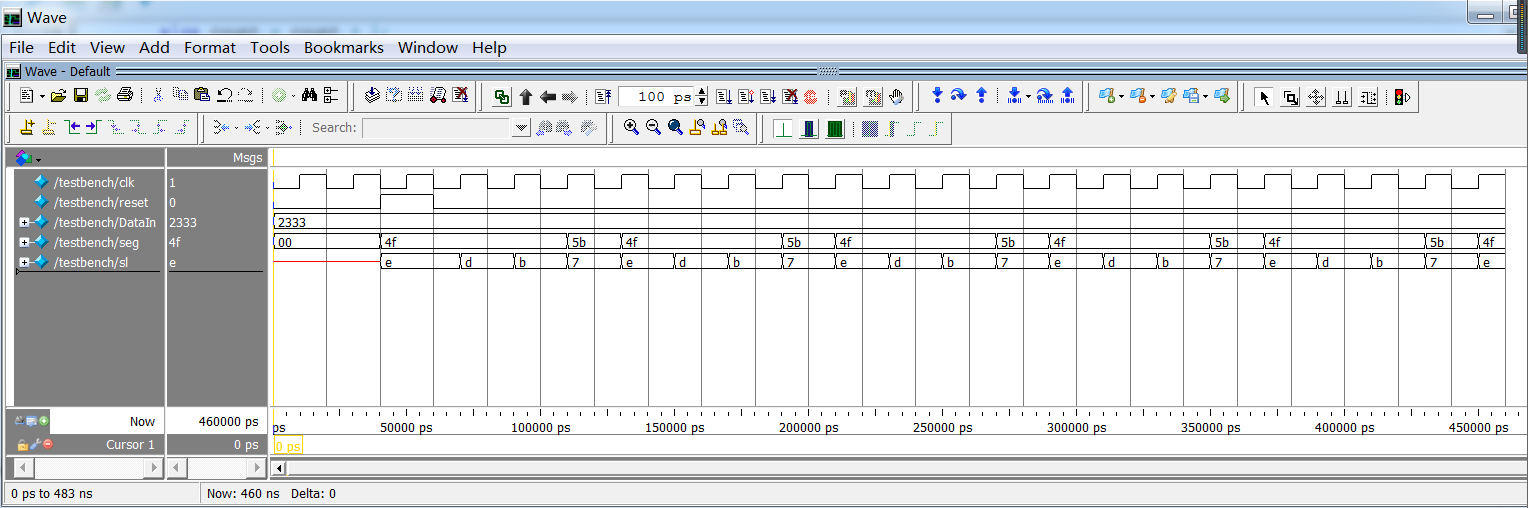
//综合实验六：



//综合实验七：

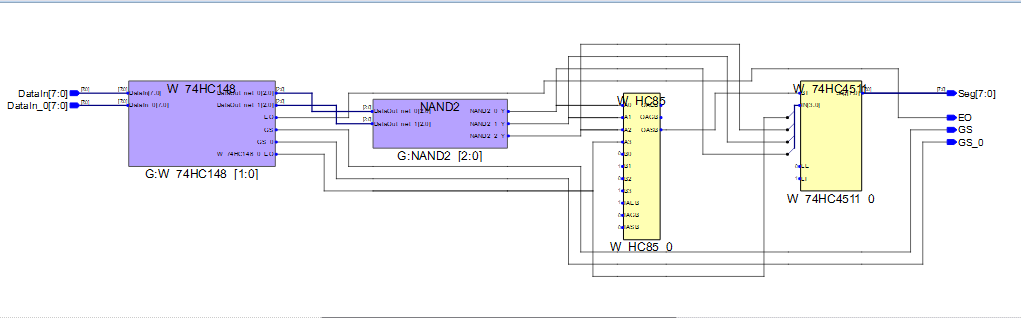


//综合实验九：

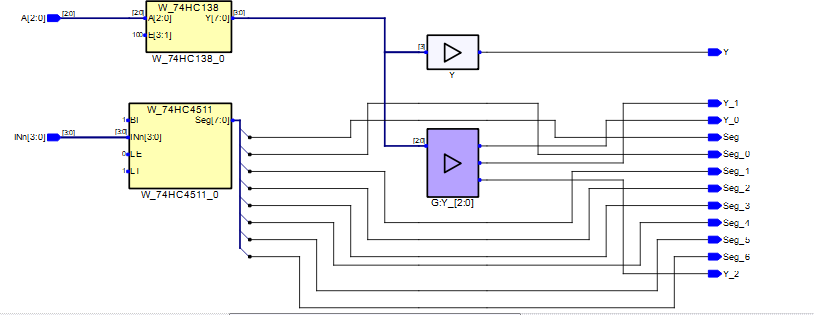


（3）综合结果RTL图

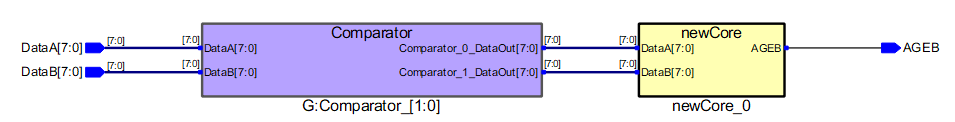
//综合实验一：



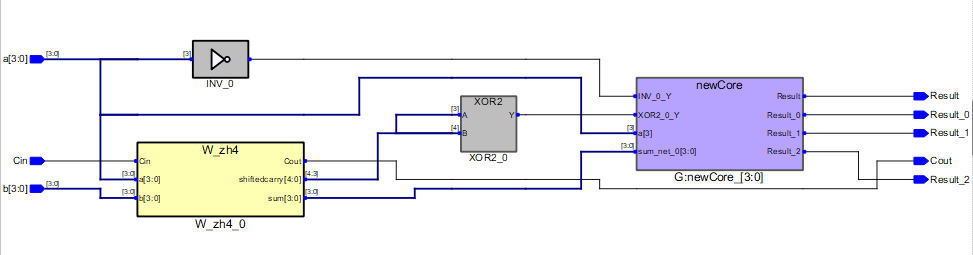
//综合实验二：



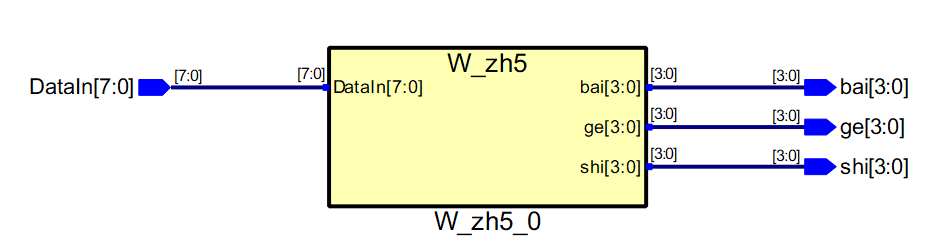
//综合实验三：



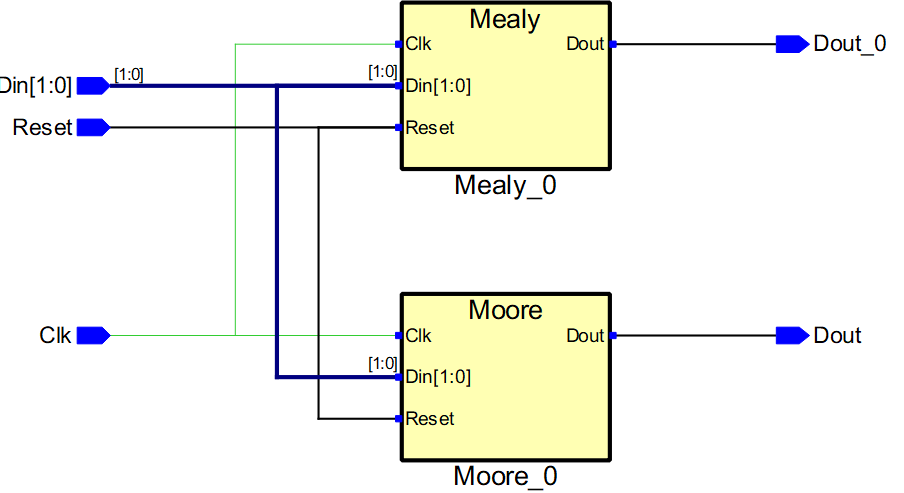
//综合实验四：



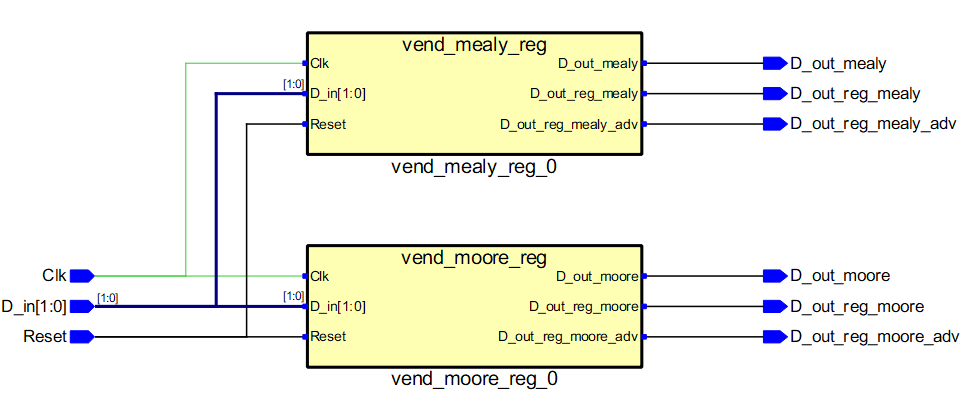
//综合实验五：



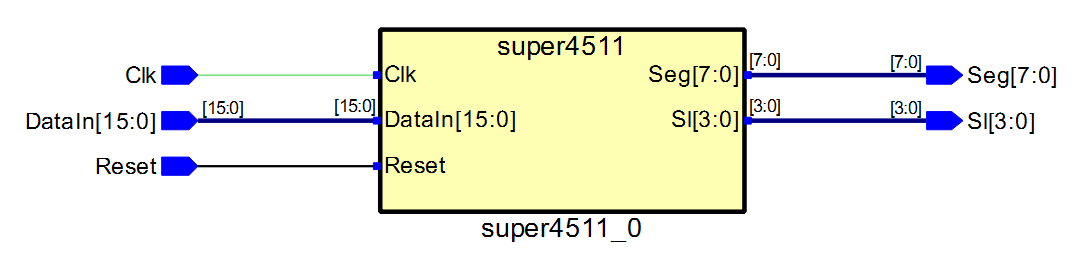
//综合实验六：



//综合实验七：

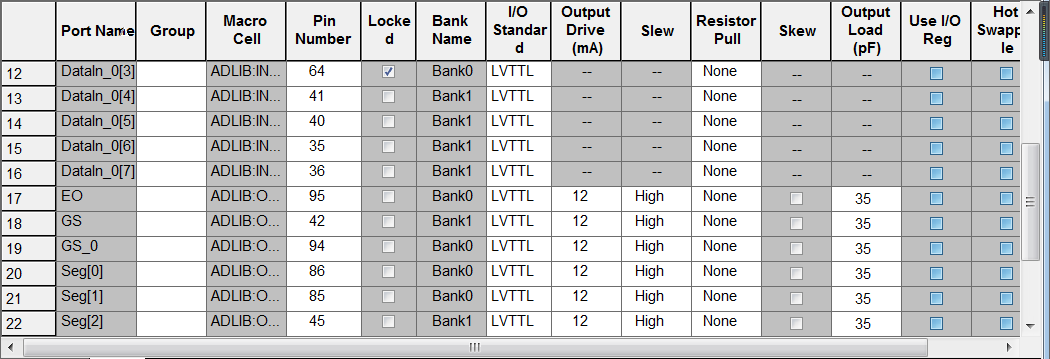
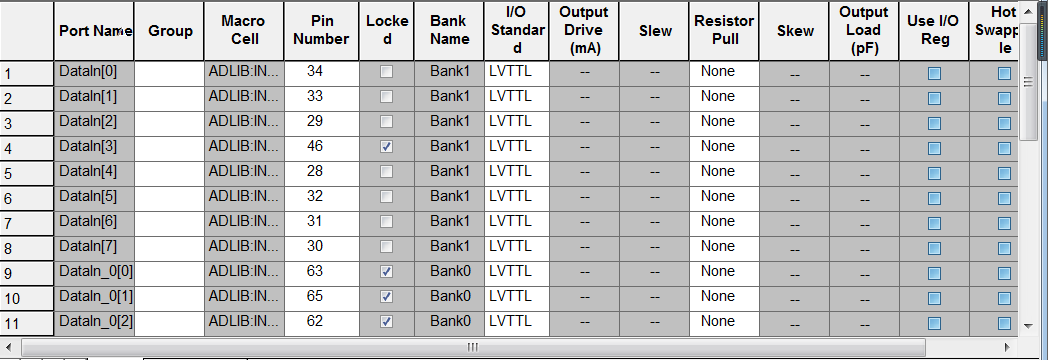
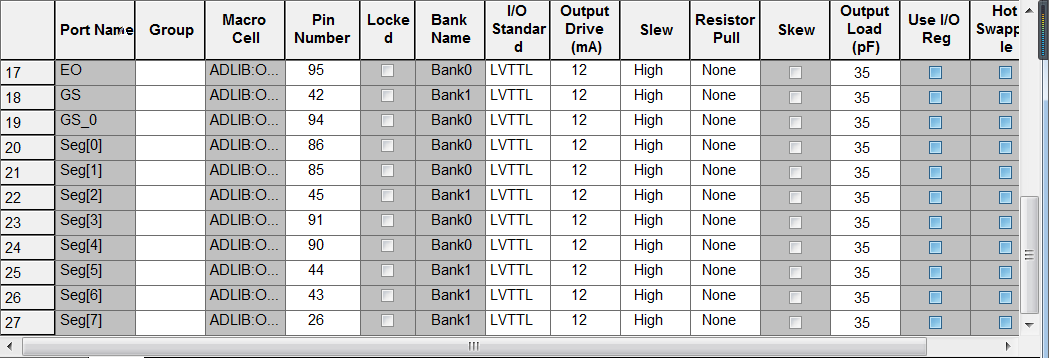


//综合实验九：

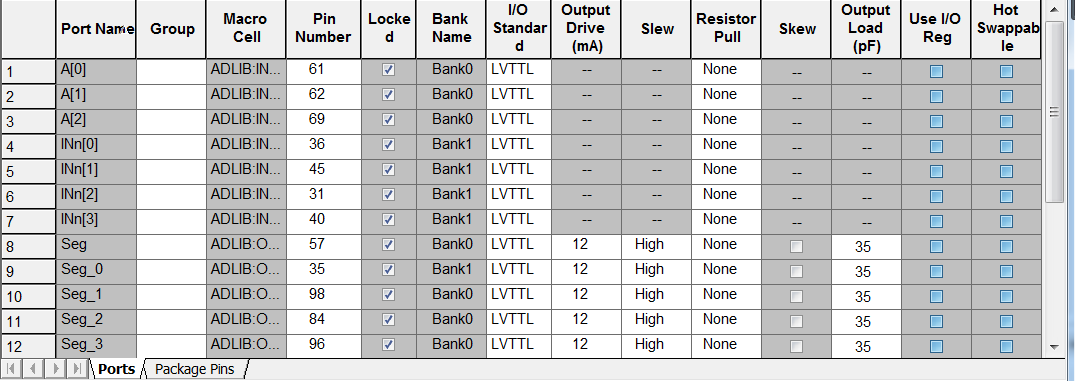


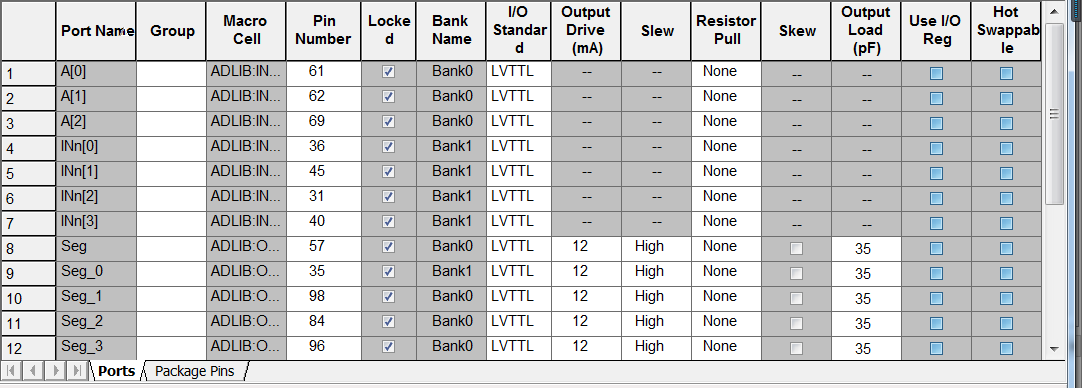
（4）引脚分配I/O Attribute Editor截图

//综合实验一：

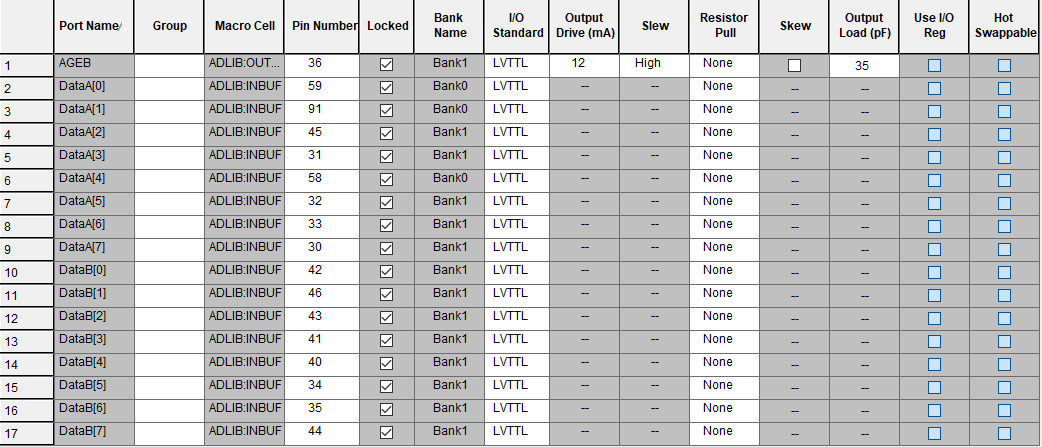
 

//综合实验二：

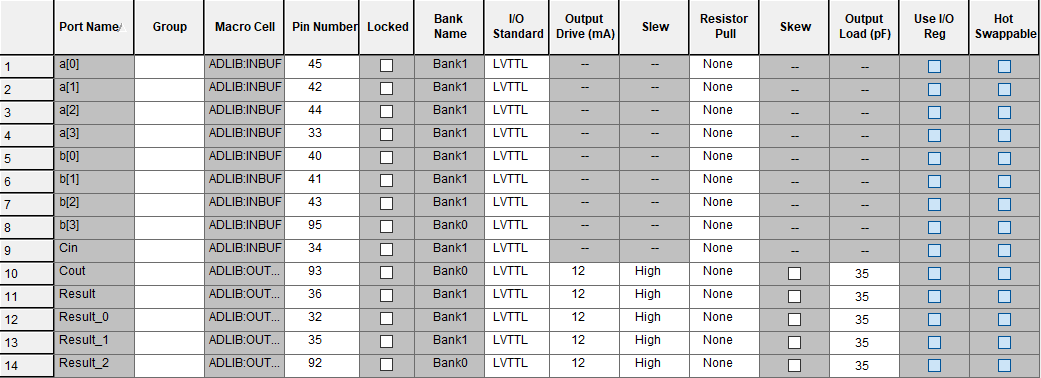




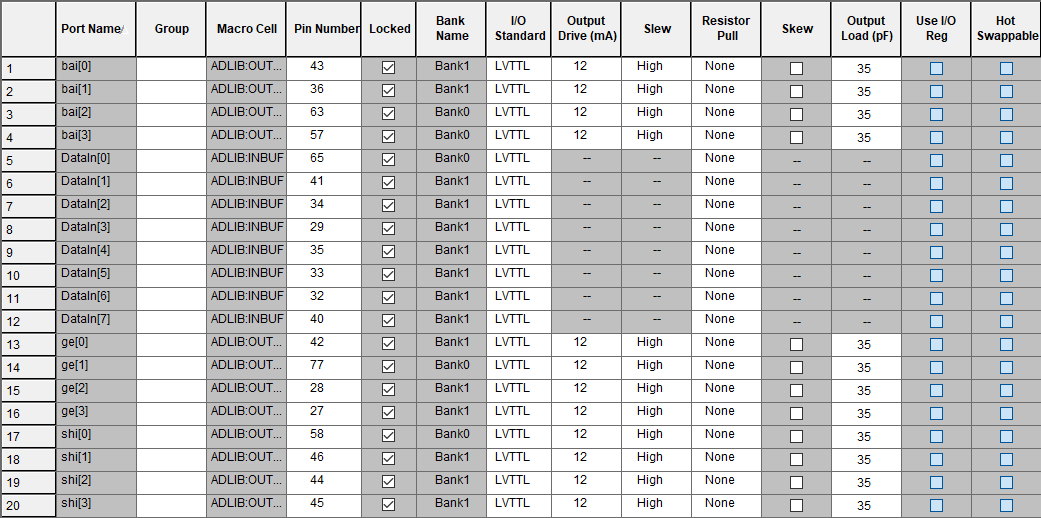
//综合实验三：



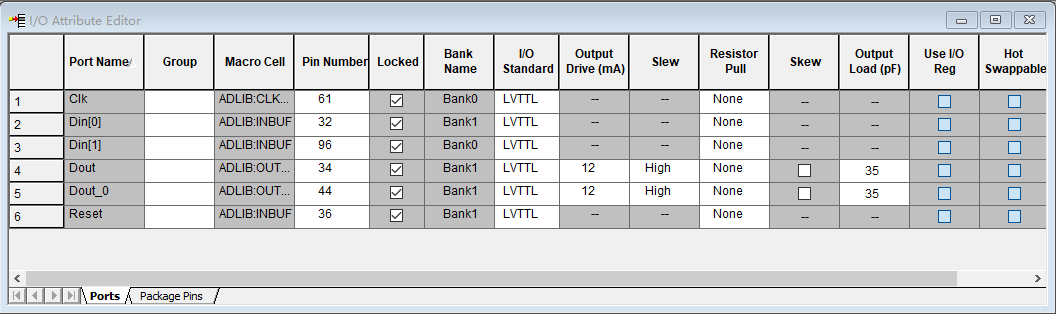
//综合实验四：



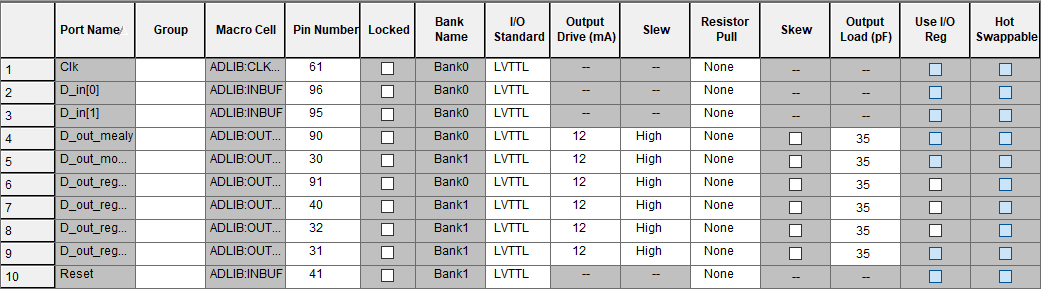
//综合实验五：



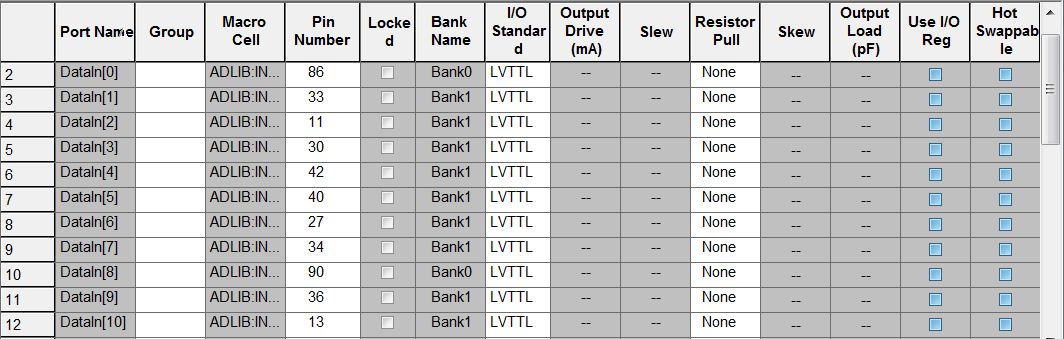
//综合实验六：

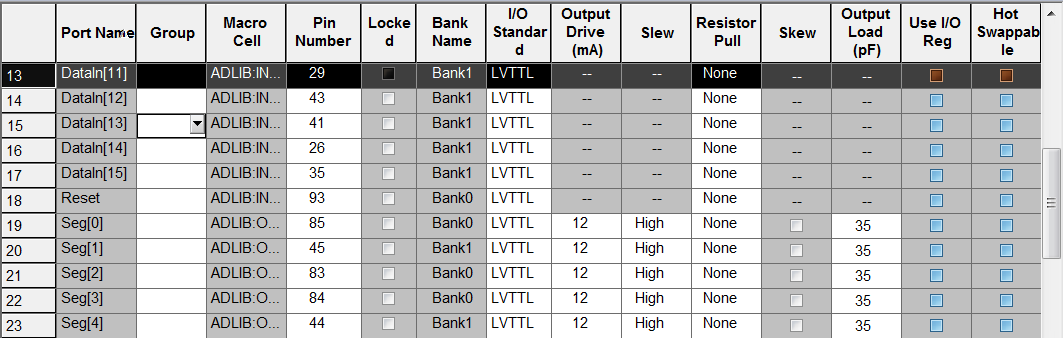


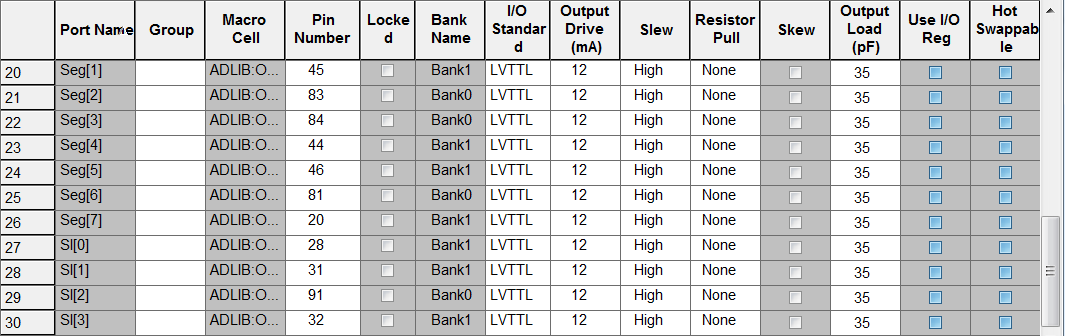
//综合实验七：



//综合实验九：





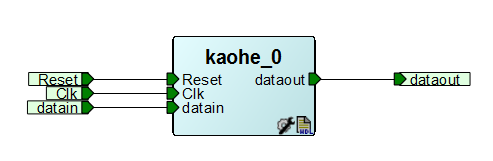


（5）记录实测结果：已在课堂上检查

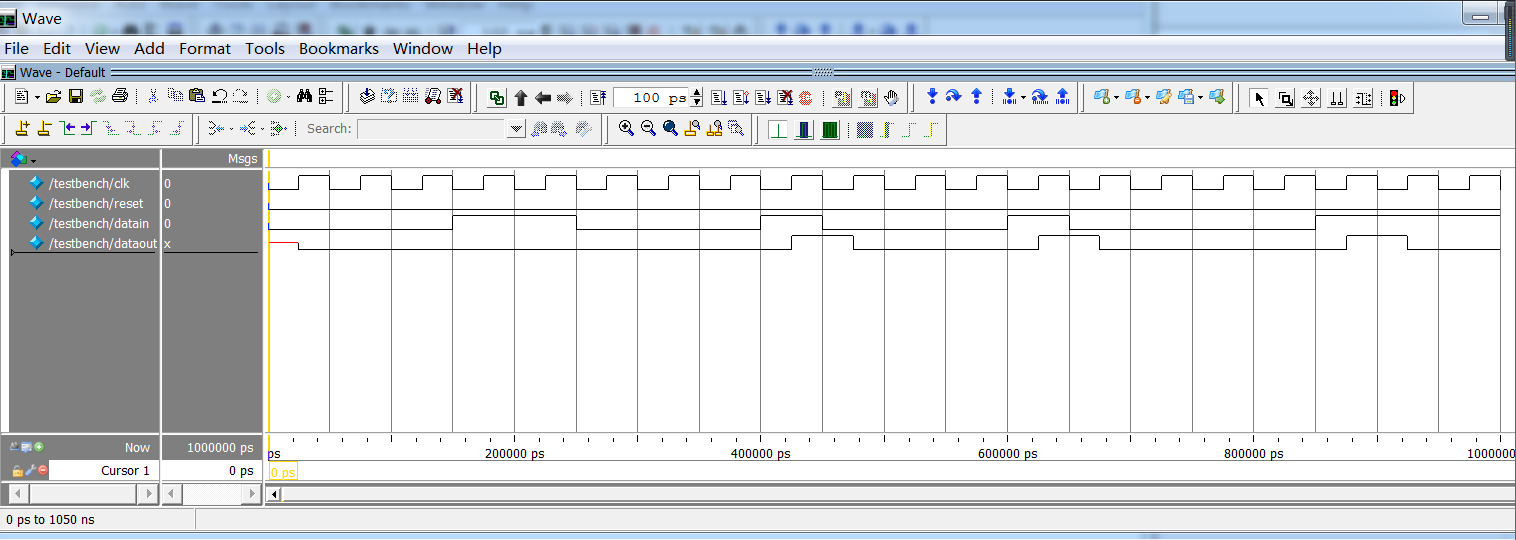
大考核

请在120分钟内按老师现场布置的题目要求，在Libero中完成设计并烧录实测，将设计及运行结果记录下来。

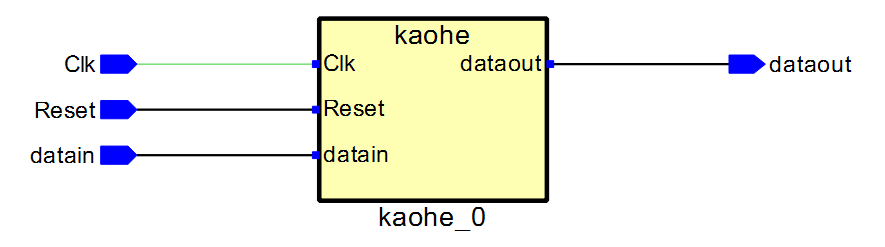
SmartDesign:



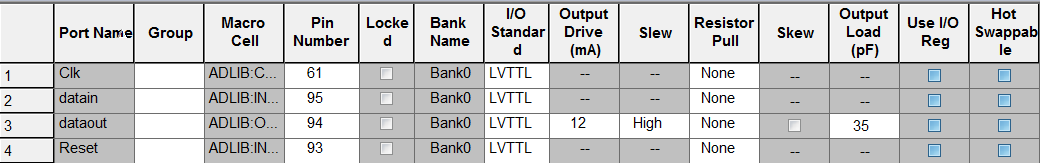
仿真波形图：



RTL：



引脚图：



## 附录：子箱26位开关及LED灯引脚分配表

如果单独使用子箱，由于子箱中提供输入输出的26位开关及LED灯，已在内部与FPGA芯片相关引脚连接，所以在引脚分配时必须按照下表进行。若不需要用这26位开关及LED灯，则无需遵照。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **26开关** | **开关座编号** | **物理引脚** | **26灯** | **物理引脚** |
| KEY1 | U2 | 15 | LED1 | 2 |
| KEY2 | 11 | LED2 | 3 |
| KEY3 | 10 | LED3 | 4 |
| KEY4 | 8 | LED4 | 46 |
| KEY5 | 7 | LED5 | 45 |
| KEY6 | 6 | LED6 | 44 |
| KEY7 | 5 | LED7 | 43 |
| KEY8 | 57 | LED8 | 42 |
| KEY1 | U3 | 58 | LED9 | 41 |
| KEY2 | 59 | LED10 | 40 |
| KEY3 | 60 | LED11 | 36 |
| KEY4 | 61 | LED12 | 35 |
| KEY5 | 62 | LED13 | 34 |
| KEY6 | 63 | LED14 | 33 |
| KEY7 | 64 | LED15 | 32 |
| KEY8 | 65 | LED16 | 31 |
| KEY1 | U6 | 69 | LED17 | 30 |
| KEY2 | 70 | LED18 | 29 |
| KEY3 | 71 | LED19 | 28 |
| KEY4 | 72 | LED20 | 27 |
| KEY5 | 73 | LED21 | 26 |
| KEY6 | 76 | LED22 | 23 |
| KEY7 | 77 | LED23 | 22 |
| KEY8 | 78 | LED24 | 21 |
| KEY1 | U7 | 79 | LED25 | 20 |
| KEY2 | 80 | LED26 | 19 |