

**数 字 电 路**

**实验报告**

**班级：**教221

**组号：----**

**姓名：**唐嘉良

**学号：**2020K8009907032

**实验名称：**16位比较器实验

**2021年10月26日**

**一、实验目的**

1、 熟悉 verilog 编程、调试

2、 熟悉简单比较器的工作原理

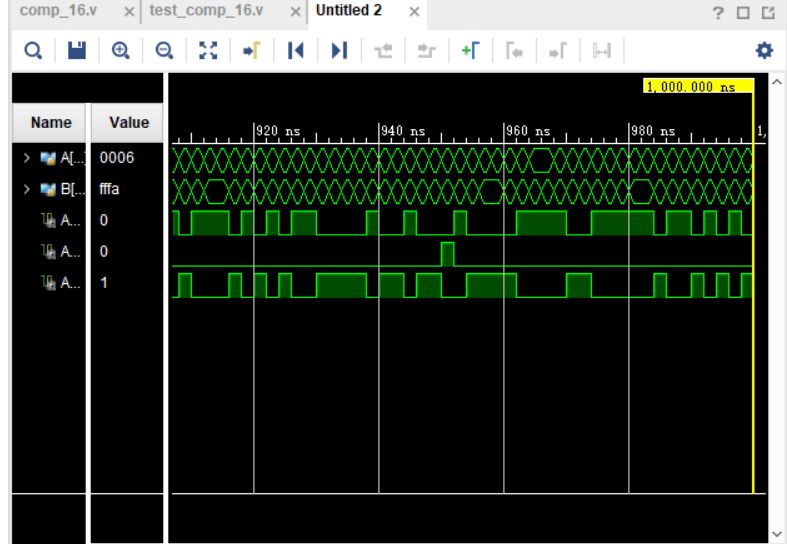
3、 通过简单模块例化、连线实现复杂的数字电路

1. **实验环境**

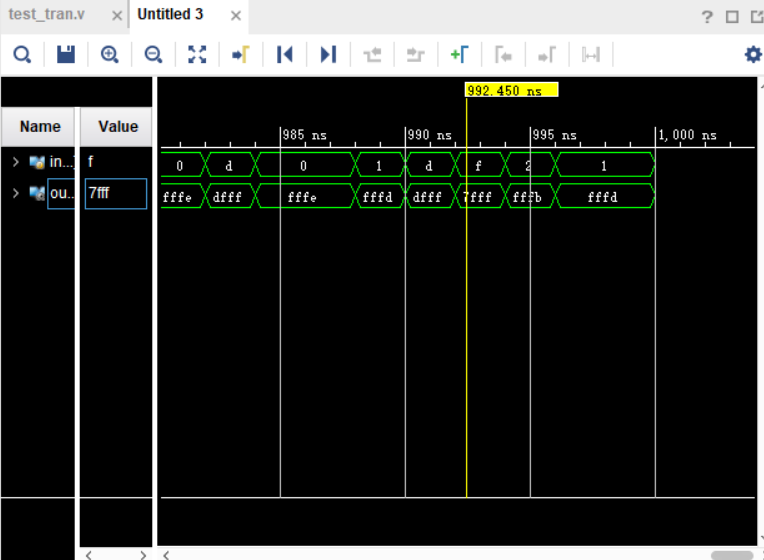
本次实验我采用的是vivado 2017.4版本。

1. **调试过程**

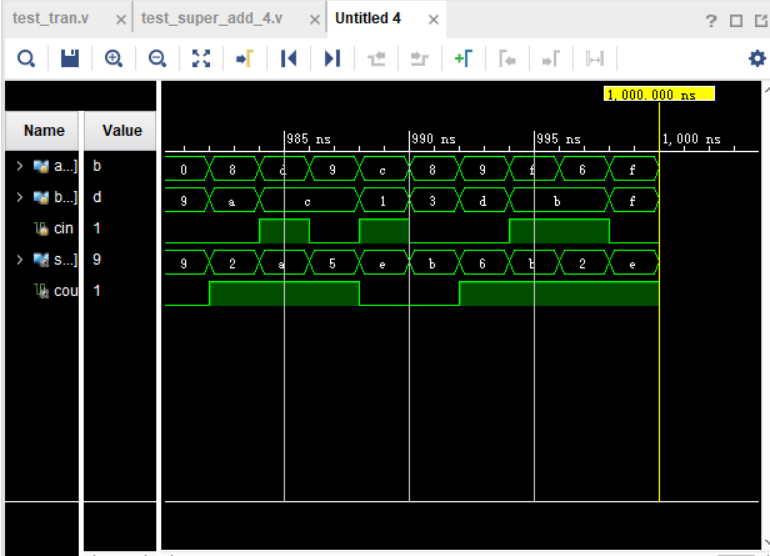
16bit比较器



4-16线译码器



4位超前进位加法器



1. **实验总结**

在此次实验中，我更加熟悉vivado平台的操作流程，现在能够创建激励文件并进行调试。同时，通过构建16bit比较器、4-16译码器和4bit超前进位加法器，我对Verilog语言的掌握程度大大提升，能够更加熟练自如地构建并调试某些常见组合电路模块。

1. **源代码**

**16bit比较器**

module comp\_4(

input [3:0] A,

input [3:0] B,

input in\_A\_G\_B,

input in\_A\_E\_B,

input in\_A\_L\_B,

output out\_A\_G\_B,

output out\_A\_E\_B,

output out\_A\_L\_B

);

assign out\_A\_G\_B = (A[3]&(~B[3])) | (~(A[3]^B[3])&A[2]&(~B[2])) | (~(A[3]^B[3])&~(A[2]^B[2])&A[1]&(~B[1])) | (~(A[3]^B[3])&~(A[2]^B[2])&~(A[1]^B[1])&A[0]&(~B[0])) | (~(A[3]^B[3])&~(A[2]^B[2])&~(A[1]^B[1])&~(A[0]^B[0])&in\_A\_G\_B);

assign out\_A\_L\_B = ((~A[3])&B[3]) | (~(A[3]^B[3])&(~A[2])&B[2]) | (~(A[3]^B[3])&~(A[2]^B[2])&(~A[1])&B[1]) | (~(A[3]^B[3])&~(A[2]^B[2])&~(A[1]^B[1])&(~A[0])&B[0]) | (~(A[3]^B[3])&~(A[2]^B[2])&~(A[1]^B[1])&~(A[0]^B[0])&in\_A\_L\_B);

assign out\_A\_E\_B = ~(A[3]^B[3])&~(A[2]^B[2])&~(A[1]^B[1])&~(A[0]^B[0])&in\_A\_E\_B;

endmodule

module comp\_16(

input [15:0] A,

input [15:0] B,

output A\_G\_B,

output A\_E\_B,

output A\_L\_B);

wire [4:0] temp\_A\_G\_B;

wire [4:0] temp\_A\_E\_B;

wire [4:0] temp\_A\_L\_B;

assign temp\_A\_G\_B[0]=0;

assign temp\_A\_E\_B[0]=1;

assign temp\_A\_L\_B[0]=0;

comp\_4 u0(A[3:0],B[3:0],temp\_A\_G\_B[0],temp\_A\_E\_B[0],temp\_A\_L\_B[0],temp\_A\_G\_B[1],temp\_A\_E\_B[1],temp\_A\_L\_B[1]);

comp\_4 u1(A[7:4],B[7:4],temp\_A\_G\_B[1],temp\_A\_E\_B[1],temp\_A\_L\_B[1],temp\_A\_G\_B[2],temp\_A\_E\_B[2],temp\_A\_L\_B[2]);

comp\_4 u2(A[11:8],B[11:8],temp\_A\_G\_B[2],temp\_A\_E\_B[2],temp\_A\_L\_B[2],temp\_A\_G\_B[3],temp\_A\_E\_B[3],temp\_A\_L\_B[3]);

comp\_4 u3(A[15:12],B[15:12],temp\_A\_G\_B[3],temp\_A\_E\_B[3],temp\_A\_L\_B[3],temp\_A\_G\_B[4],temp\_A\_E\_B[4],temp\_A\_L\_B[4]);

assign A\_G\_B = temp\_A\_G\_B[4];

assign A\_E\_B = temp\_A\_E\_B[4];

assign A\_L\_B = temp\_A\_L\_B[4];

endmodule

**激励文件**

module test\_comp\_16(

);

reg [15:0] A;

reg [15:0] B;

wire A\_G\_B;

wire A\_E\_B;

wire A\_L\_B;

comp\_16 u0(A,B,A\_G\_B,A\_E\_B,A\_L\_B);

initial begin

A=16'b1;

B=16'b0;

end

always begin

#2;

A=$random()%16;

B=$random()%16;

end

endmodule

**4-16译码器**

module tran\_4\_16(

input [3:0] in,

output [15:0] out

);

reg [15:0] out;

always @ (in) begin

case(in)

4'b0000:out=~16'b0000\_0000\_0000\_0001;

4'b0001:out=~16'b0000\_0000\_0000\_0010;

4'b0010:out=~16'b0000\_0000\_0000\_0100;

4'b0011:out=~16'b0000\_0000\_0000\_1000;

4'b0100:out=~16'b0000\_0000\_0001\_0000;

4'b0101:out=~16'b0000\_0000\_0010\_0000;

4'b0110:out=~16'b0000\_0000\_0100\_0000;

4'b0111:out=~16'b0000\_0000\_1000\_0000;

4'b1000:out=~16'b0000\_0001\_0000\_0000;

4'b1001:out=~16'b0000\_0010\_0000\_0000;

4'b1010:out=~16'b0000\_0100\_0000\_0000;

4'b1011:out=~16'b0000\_1000\_0000\_0000;

4'b1100:out=~16'b0001\_0000\_0000\_0000;

4'b1101:out=~16'b0010\_0000\_0000\_0000;

4'b1110:out=~16'b0100\_0000\_0000\_0000;

4'b1111:out=~16'b1000\_0000\_0000\_0000;

endcase

end

endmodule

**激励文件**

module test\_tran(

);

reg [3:0] in;

wire [15:0] out;

tran\_4\_16 u0(in,out);

initial begin

in = 4'b0001;

end

always begin

#2;

in = $random() % 4;

end

Endmodule

**4bit超前进位加法器**

module add\_super\_4(

input [3:0] a,

input [3:0] b,

input cin,

output [3:0] s,

output cout

);

wire [4:0] g,p,c;

assign c[0]=cin;

assign p = a | b;

assign g = a & b;

assign c[1] = g[0]|(p[0]&c[0]);

assign c[2] = g[1]|(p[1]&(g[0]|(p[0]&c[0])));

assign c[3] = g[2]|(p[2]&(g[1]|(p[1]&(g[0]|(p[0]&c[0])))));

assign c[4] = g[3]|(p[3]&(g[2]|(p[2]&(g[1]|(p[1]&(g[0]|(p[0]&c[0])))))));

assign s = a^b^c [3:0];

assign cout = c[4];

endmodule

**激励文件**

module test\_super\_add\_4(

);

reg [3:0] a,b;

reg cin;

wire [3:0] s;

wire cout;

add\_super\_4 add\_super(

a,b,cin,s,cout);

initial begin

a = 4'b1000;

b = 4'b0111;

cin = 0;

end

always begin

#2;

a = $random() %16;

b = $random() %16;

cin = $random() %2;

end

endmodule