中国科学院大学
University of Chinese Academy of Sciences

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

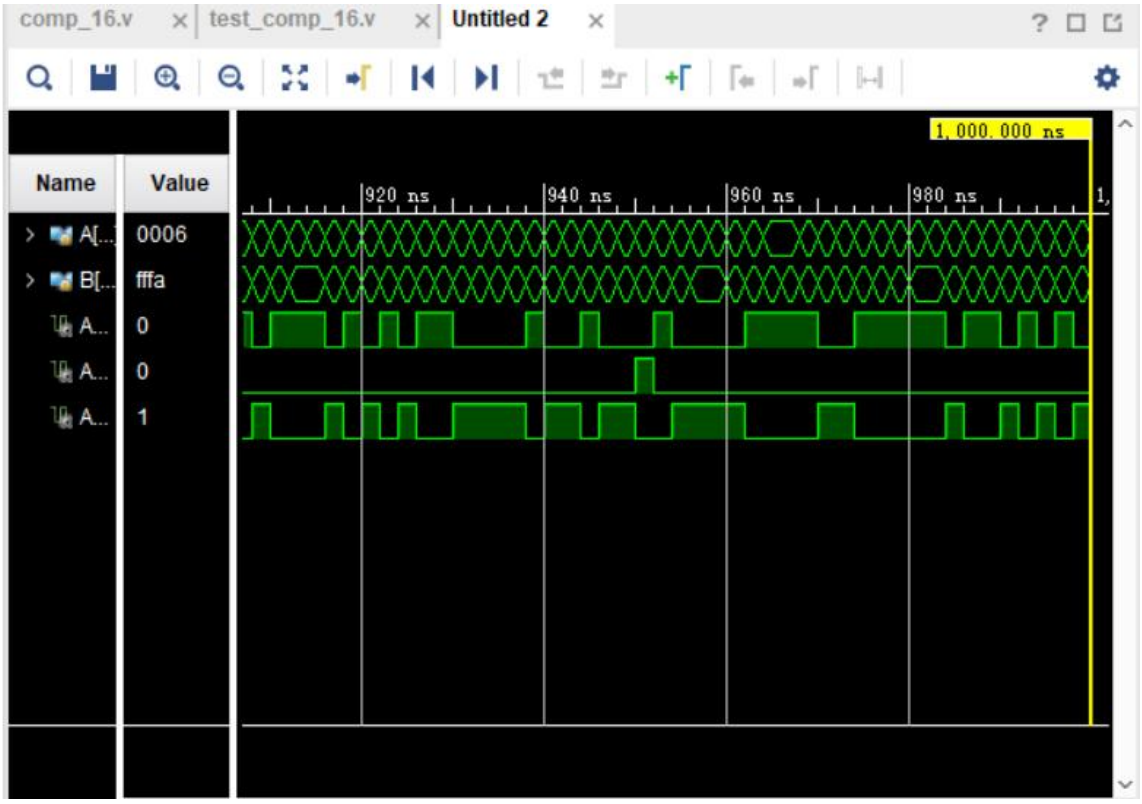
1、 熟悉 verilog 编程、调试
2、 熟悉简单比较器的工作原理
3、 通过简单模块例化、连线实现复杂的数字电路

二、实验环境

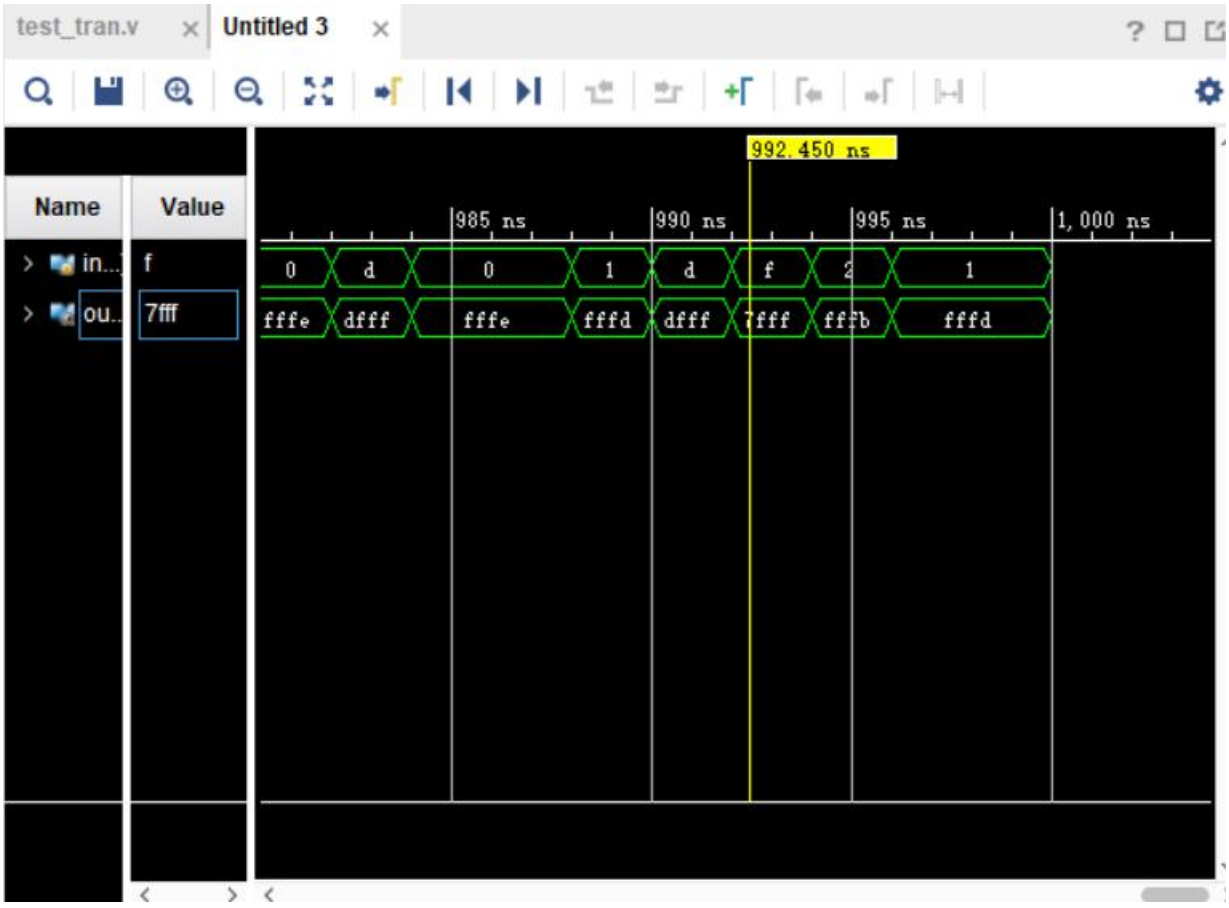
本次实验我采用的是 vivado 2017.4 版本。

三、调试过程

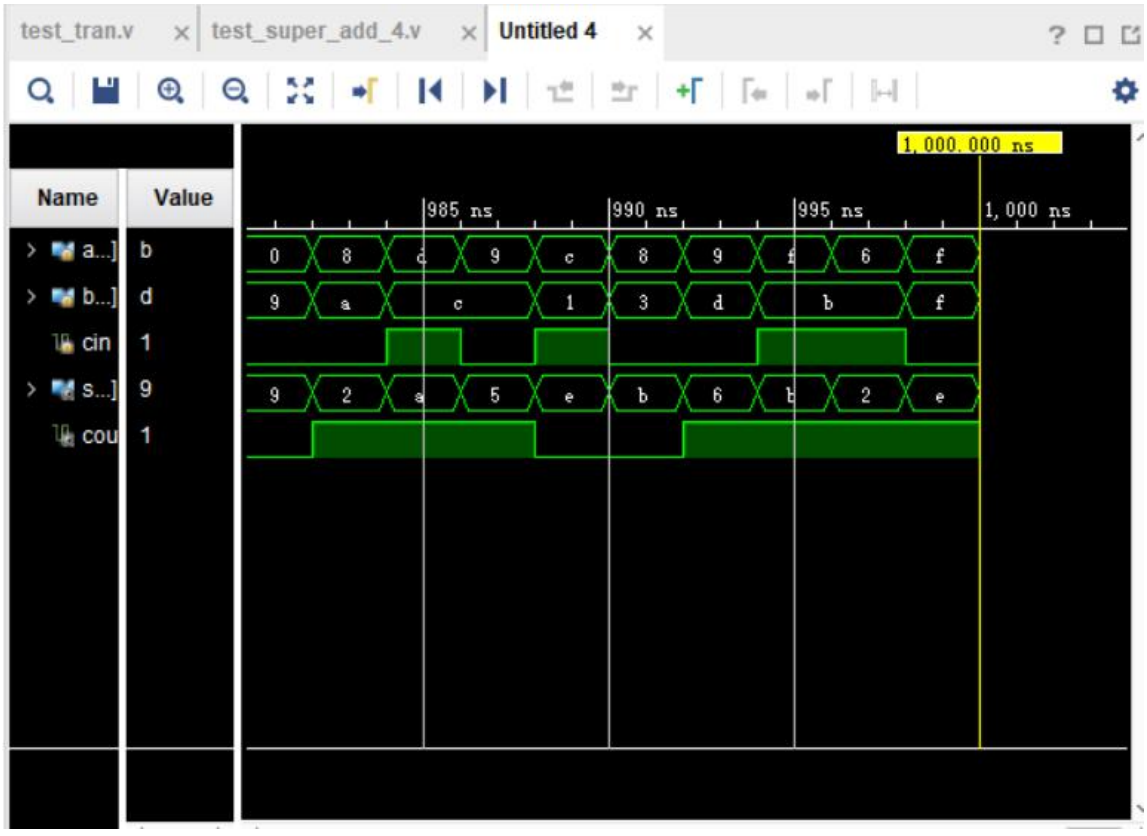
16bit 比较器



4-16 线译码器


test_tran.v
Untitled 3
992.450 ns
Name
Value
985 ns
990 ns
995 ns
1,000 ns
in...
f
ou...
7fff
0
d
0
1
d
f
1
fffe
dfff
fffe
fffd
dfff
7fff
fffb
fffd

4 位超前进位加法器


test_tran.v
test_super_add_4.v
Untitled 4
1,000.000 ns
Name
Value
985 ns
990 ns
995 ns
1,000 ns
a...
b
b...
d
cin
1
s...
9
cou
1
0
8
d
9
c
8
9
f
6
f
9
a
c
1
3
d
b
f
9
2
a
5
e
b
6
b
2
e

四、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。

五、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
```