实验 2　16 位比较器实验

班级：01　　学号：2021k8009925006　　姓名：冯浩瀚

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

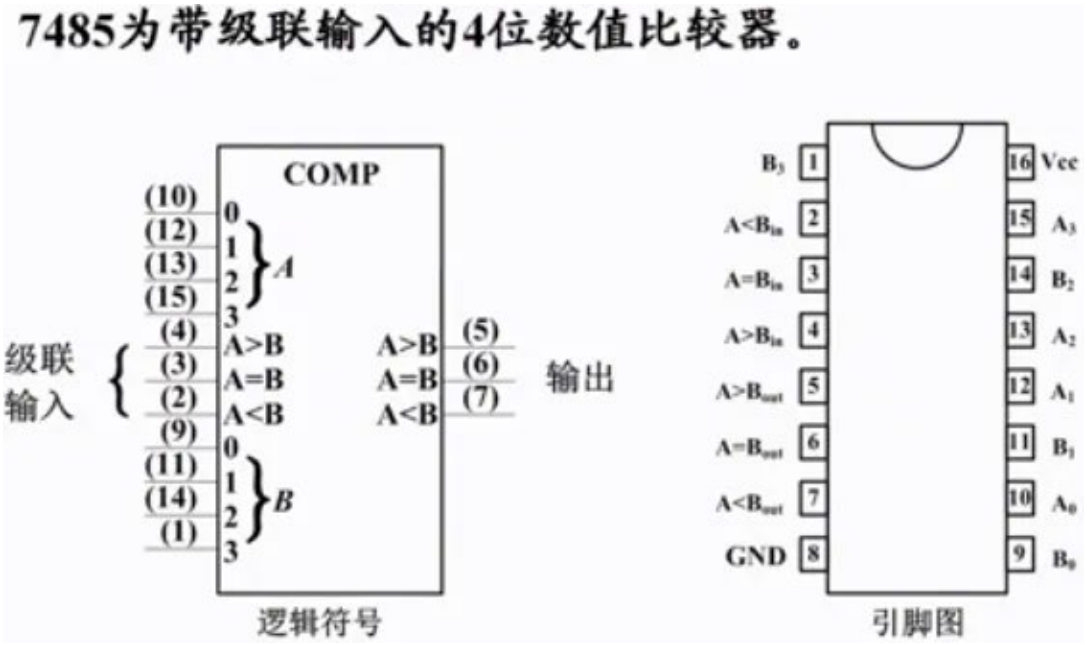
实验环境:

1、操作系统：Windows 11

2、软件及版本：Vivado 2017.4

原理说明

1. 4 位比较器



7485功能表

比较输入				级联输入			输出		
A_3 B_3	A_2 B_2	A_1 B_1	A_0 B_0	$I_{(A>B)}$	$I_{(A<B)}$	$I_{(A=B)}$	$Y_{(A>B)}$	$Y_{(A<B)}$	$Y_{(A=B)}$
$A_3>B_3$	×	×	×	×	×	×	1	0	0
$A_3<B_3$	×	×	×	×	×	×	0	1	0
$A_3=B_3$	$A_2>B_2$	×	×	×	×	×	1	0	0
$A_3=B_3$	$A_2<B_2$	×	×	×	×	×	0	1	0
$A_3=B_3$	$A_2=B_2$	$A_1>B_1$	×	×	×	×	1	0	0
$A_3=B_3$	$A_2=B_2$	$A_1<B_1$	×	×	×	×	0	1	0
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0>B_0$	×	×	×	1	0	0
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0<B_0$	×	×	×	0	1	0
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	1	0	0	1	0	0
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	0	1	0	0	1	0
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	×	×	1	0	0	1
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	1	1	0	0	0	0
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	0	0	0	1	1	0

√

在具体的 Verilog 语言编写 comp_4 时可采用条件语句，当低位比较结果为大于或小于时直接输出大于或小于；当低位比较结果为等于时将输入的 2 个四位数进行比较大小并输出结果。

2. 16 位比较器

将 4 个 4 位比较器串行级联实现 16 位比较器。在具体的 Verilog 语言编写时可采用调用 module comp_4 的方法实现。

接口定义:

1. 4 位比较器

Input 信号			Output 信号	
名称	说明	位宽	名称	位宽
A	用于比较	4	out_A_G_B	1
B	用于比较	4	out_A_E_B	1
in_A_G_B	低位比较结果	1	out_A_L_B	1
in_A_E_B	低位比较结果	1		
in_A_L_B	低位比较结果	1		

2. 16 位比较器

Input 信号			Output 信号	
名称	说明	位宽	名称	位宽
A	用于比较	4	out_A_G_B	1
B	用于比较	4	out_A_E_B	1
			out_A_L_B	1

调试过程及结果波形:

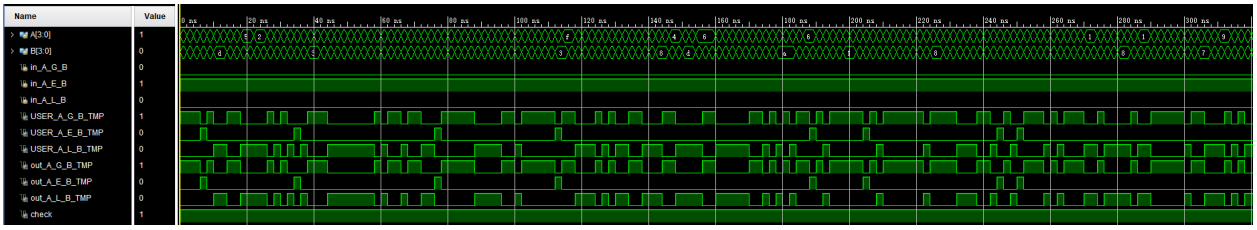
1. 编写 comp_16 时采用的信号名称为 out_A_G_B、out_A_E_B、out_A_L_B, 而老师给出的仿真激励文件的模板中使用 A_G_B、A_E_B、A_L_B, 仔细观察后调试成功。
2. 模块例化的时候接口顺序出错，仔细观察后调试成功，建议以后使用命名端口连接法，不需要保证顺序相同，只需要保证端口口名字与外部信号匹配（例如下图，来自 https://www.runoob.com/w3cnote/verilog-generate.html）

实例

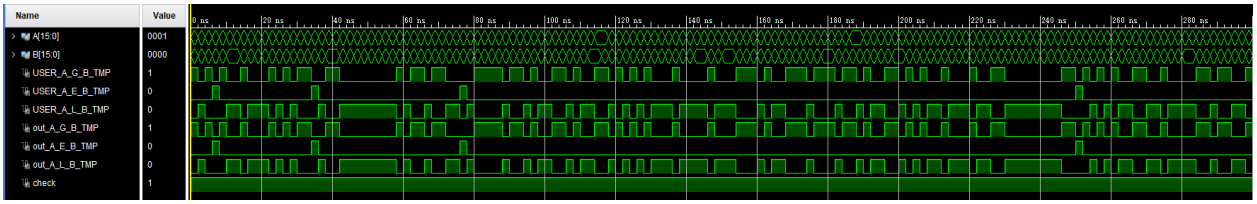
```
full_adder1  u_adder0(
    .Ai     (a[0]),
    .Bi     (b[0]),
    .Ci     (c==1'b1 ? 1'b0 : 1'b1),
    .So     (so_bit0),
    .Co     (co_temp[0]));
```

结果波形：

1. 4 位比较器



2. 16 位比较器



实验总结:

本实验中练习编写了 4 位比较器，并利用模块例化实现 16 位比较器。通过实验学习了用简单的模块例化、连线实现更复杂的数字电路。

源代码:

1. 4 位比较器：

```
`timescale 1ns / 1ps
```

```
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 2022/10/20 10:16:22
// Design Name:
// Module Name: comp_4
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


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 reg out_A_G_B,
    output reg out_A_E_B,
    output reg out_A_L_B
    );
    always@ (A or B or in_A_G_B or in_A_E_B or in_A_L_B)
    begin
        if(in_A_G_B == 1)
        begin
            out_A_G_B <= 1;
            out_A_E_B <= 0;
            out_A_L_B <= 0;
        end
        else if(in_A_L_B == 1)
        begin
            out_A_G_B <= 0;
            out_A_E_B <= 0;
            out_A_L_B <= 1;
```

```
        end
        else if(in_A_E_B == 1)
        begin
            if(A > B)
            begin
                out_A_G_B <= 1;
                out_A_E_B <= 0;
                out_A_L_B <= 0;
            end
            else if(A == B)
            begin
                out_A_G_B <= 0;
                out_A_E_B <= 1;
                out_A_L_B <= 0;
            end
            else if(A < B)
            begin
                out_A_G_B <= 0;
                out_A_E_B <= 0;
                out_A_L_B <= 1;
            end
        end
    end

endmodule
```

2. 16 位比较器源码：

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 2022/10/20 11:17:42
// Design Name:
// Module Name: comp_16
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
```

```
//
//////////////////////////////////////////////////////////////////////////////////
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 2022/10/20 10:16:22
// Design Name:
// Module Name: comp_4
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module comp_4(
    input in_A_G_B,
    input in_A_E_B,
    input in_A_L_B,
    input [3:0] A,
    input [3:0] B,
    output reg out_A_G_B,
    output reg out_A_E_B,
    output reg out_A_L_B
    );
    always@ (A or B or in_A_G_B or in_A_E_B or in_A_L_B)
    begin
        if(in_A_G_B == 1)
        begin
            out_A_G_B <= 1;
            out_A_E_B <= 0;
            out_A_L_B <= 0;
        end
        else if(in_A_L_B == 1)
        begin
```

```
                out_A_G_B <= 0;
                out_A_E_B <= 0;
                out_A_L_B <= 1;
            end
            else
            begin
                if(A > B)
                begin
                    out_A_G_B <= 1;
                    out_A_E_B <= 0;
                    out_A_L_B <= 0;
                end
                else if(A == B)
                begin
                    out_A_G_B <= 0;
                    out_A_E_B <= 1;
                    out_A_L_B <= 0;
                end
                else if(A < B)
                begin
                    out_A_G_B <= 0;
                    out_A_E_B <= 0;
                    out_A_L_B <= 1;
                end
            end
        end

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 [2:0] G, E, L;
        wire [3:0] AC3,BC3,AC2,BC2,AC1,BC1,AC0,BC0;
        wire in_A_G_B = 0;
        wire in_A_E_B = 1;
        wire in_A_L_B = 0;
        genvar k;
        for(k=15; k>=12; k=k-1)
        begin
```

```
        assign    AC3[k-12] = A[k];
        assign    BC3[k-12] = B[k];
    end
    for(k=11; k>=8; k=k-1)
    begin
        assign    AC2[k-8] = A[k];
        assign    BC2[k-8] = B[k];
    end
    for(k=7; k>=4; k=k-1)
    begin
        assign    AC1[k-4] = A[k];
        assign    BC1[k-4] = B[k];
    end
      for(k=3; k>=0; k=k-1)
    begin
        assign    AC0[k] = A[k];
        assign    BC0[k] = B[k];
    end
    comp_4 C3(in_A_G_B, in_A_E_B, in_A_L_B, AC3, BC3, G[0], E[0], L[0]);
    comp_4 C2(G[0], E[0], L[0], AC2, BC2, G[1], E[1], L[1]);
    comp_4 C1(G[1], E[1], L[1], AC1, BC1, G[2], E[2], L[2]);
    comp_4 C0(G[2], E[2], L[2], AC0, BC0, A_G_B, A_E_B, A_L_B);

endmodule
```