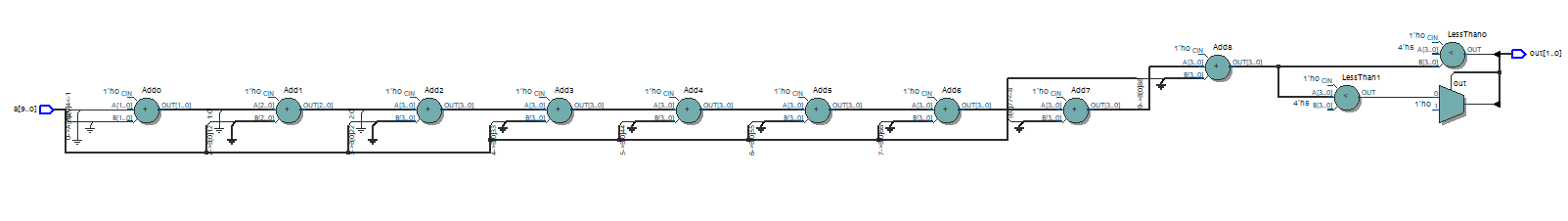
1. **十人裁决器**

**设计思路：**

十人裁决器，当多于5人，输出为10，等于5人则为00，小于5人则为01；同时可以用sum统计十人的1的数目，进行判定。

**原理图：**



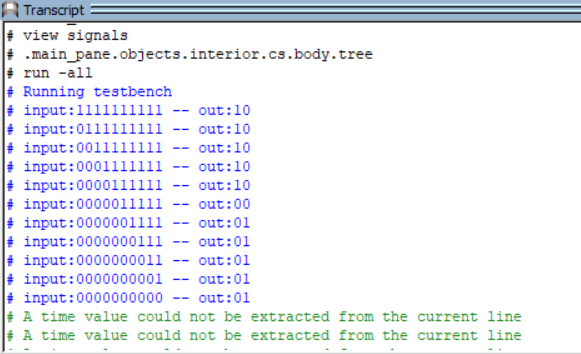
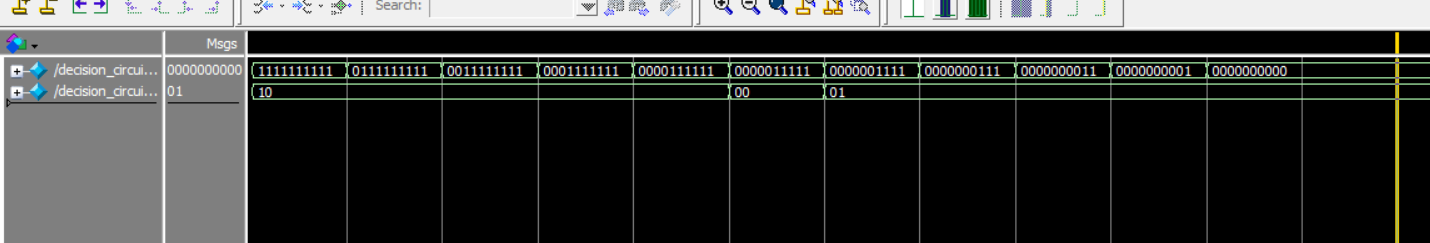
**功能模块代码：**

module decision\_circuit\_10(a,out);  
input[9:0] a;  
output out;  
reg[1:0] out;  
reg[3:0] sum;  
always@(a,sum,out)  
begin  
sum=a[0]+a[1]+a[2]+a[3]+a[4]+a[5]+a[6]+a[7]+a[8]+a[9];  
if(sum>5) out=2&apos;b10;  
else if(sum<5) out=2&apos;b01;  
else out=2&apos;b00;  
end  
endmodule

**测试代码：**

`timescale 1 ps/ 1 ps  
module decision\_circuit\_10\_vlg\_tst();  
reg [9:0] a;                                               
wire[1:0] out;  
decision\_circuit\_10 i1 (    
.a(a),  
.out(out)  
);  
initial                                                  
begin  
$display("Running testbench");   
#0 a[0] = 1; a[1] = 1; a[2] = 1; a[3] = 1; a[4] = 1; a[5] = 1; a[6] = 1;a[7]=1;a[8]=1;a[9]=1;  
#1 $display("input:%b -- out:%b", a, out);  
#0 a[0] = 1; a[1] = 1; a[2] = 1; a[3] = 1; a[4] = 1; a[5] = 1; a[6] = 1;a[7]=1;a[8]=1;a[9]=0;  
#1 $display("input:%b -- out:%b", a, out);  
#0 a[0] = 1; a[1] = 1; a[2] = 1; a[3] = 1; a[4] = 1; a[5] = 1; a[6] = 1;a[7]=1;a[8]=0;a[9]=0;  
#1 $display("input:%b -- out:%b", a, out);  
#0 a[0] = 1; a[1] = 1; a[2] = 1; a[3] = 1; a[4] = 1; a[5] = 1; a[6] = 1;a[7]=0;a[8]=0;a[9]=0;  
#1 $display("input:%b -- out:%b", a, out);  
#0 a[0] = 1; a[1] = 1; a[2] = 1; a[3] = 1; a[4] = 1; a[5] = 1; a[6] = 0;a[7]=0;a[8]=0;a[9]=0;  
#1 $display("input:%b -- out:%b", a, out);  
#0 a[0] = 1; a[1] = 1; a[2] = 1; a[3] = 1; a[4] = 1; a[5] = 0; a[6] = 0;a[7]=0;a[8]=0;a[9]=0;  
#1 $display("input:%b -- out:%b", a, out);  
#0 a[0] = 1; a[1] = 1; a[2] = 1; a[3] = 1; a[4] = 0; a[5] = 0; a[6] = 0;a[7]=0;a[8]=0;a[9]=0;  
#1 $display("input:%b -- out:%b", a, out);  
#0 a[0] = 1; a[1] = 1; a[2] = 1; a[3] = 0; a[4] = 0; a[5] = 0; a[6] = 0;a[7]=0;a[8]=0;a[9]=0;           
#1 $display("input:%b -- out:%b", a, out);  
#0 a[0] = 1; a[1] = 1; a[2] = 0; a[3] = 0; a[4] = 0; a[5] = 0; a[6] = 0;a[7]=0;a[8]=0;a[9]=0;           
#1 $display("input:%b -- out:%b", a, out);  
#0 a[0] = 1; a[1] = 0; a[2] = 0; a[3] = 0; a[4] = 0; a[5] = 0; a[6] = 0;a[7]=0;a[8]=0;a[9]=0;           
#1 $display("input:%b -- out:%b", a, out);  
#0 a[0] = 0; a[1] = 0; a[2] = 0; a[3] = 0; a[4] = 0; a[5] = 0; a[6] = 0;a[7]=0;a[8]=0;a[9]=0;           
#1 $display("input:%b -- out:%b", a, out);  
#9;  
end                                                                                           
endmodule

**运行结果图：**



**遇到的问题：**

最开始的时候由于module的名字与创建工程名字不一样出现了3warning和1error，后将名字改对后进行运行。

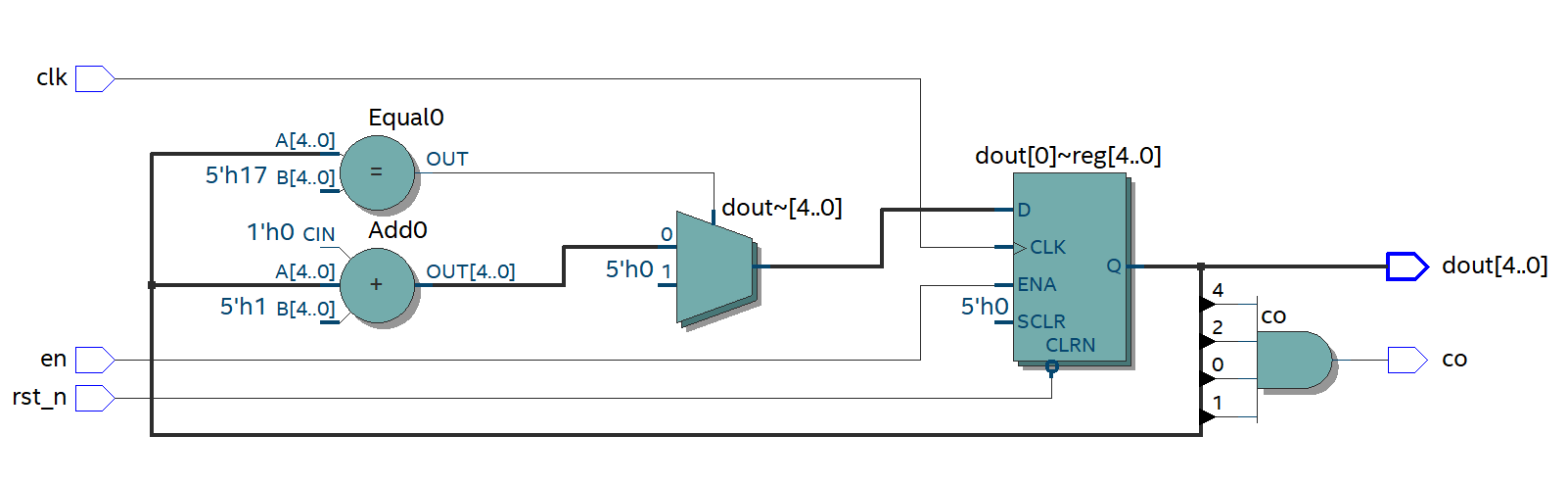
Out的格式一开始只有两位0，1，无法表示三种状态，后将out改为2位输入，产生四种选项。

1. **二十四计数器**

**设计思路：**

模24计数器，从零开始计数，计数到23时（10111），产生一个进位信号1，同时计数状态清零。

**原理图：**



**功能模块代码：**

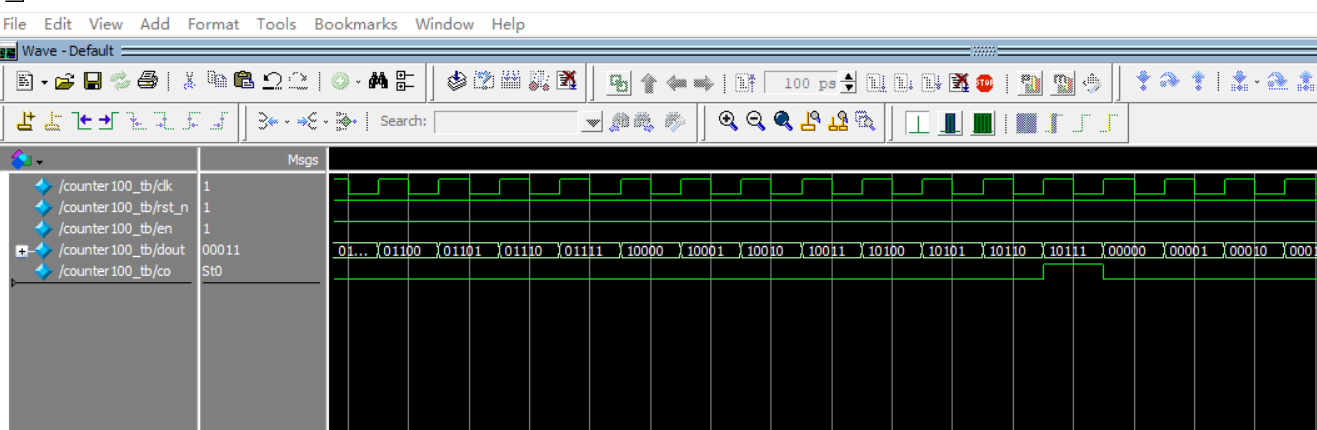
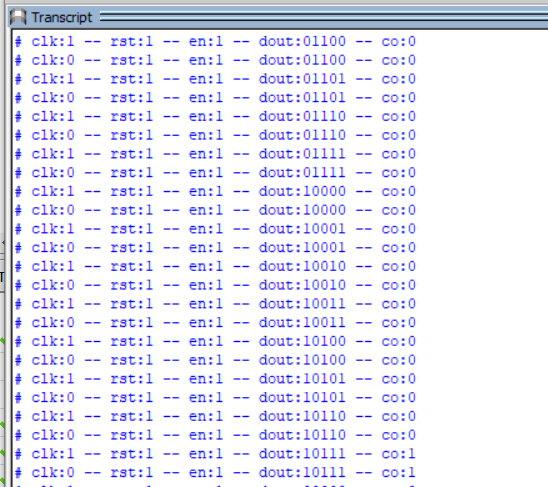
module counter100(clk, rst\_n, en, dout, co);  
input clk, rst\_n, en;  
output[4:0] dout;  
reg [4:0] dout;  
output co;  
always@(posedge clk or negedge rst\_n)  
begin  
if(!rst\_n)  
dout <= 5&apos;b00000;         
else if(en)  
if(dout == 5&apos;b10111)      
dout <= 5&apos;b00000;  
else  
dout <= dout + 1&apos;b1;   
else  
dout <= dout;  
end  
assign co = dout[0]&dout[1]&dout[2]&dout[4];    
endmodule

**测试模块代码：**

`timescale 1ns/1ps  
module counter100\_tb;  
reg clk, rst\_n, en;  
wire[4:0] dout;  
wire co;  
always  
begin  
#1 clk = ~clk;

#2 $display("clk:%b -- rst:%b -- en:%b -- dout:%b -- co:%b", clk, rst\_n,en,dout,co);  
end  
initial  
begin  
clk = 1&apos;b0;  
rst\_n = 1&apos;b1;  
en = 1&apos;b0;  
#2 rst\_n = 1&apos;b0;  
#2 rst\_n = 1&apos;b1; en = 1&apos;b1;      
end  
counter100 u1(.clk(clk), .rst\_n(rst\_n), .en(en), .dout(dout), .co(co));  
endmodule

**运行图：**



**遇到问题：**

刚开始想当然以为24位就是11000，后来发现这样就变成了25位计数，后改成了10111。

对于reg内的顺序理解错误，起初的时候10111的顺序理解成了assign co = dout[0]&dout[2]&dout[3]&dout[4];没出现进位脉冲。

应当改为assign co = dout[0]&dout[1]&dout[2]&dout[4];

对于reg的理解还叫浅薄。