2211290 姚知言 Verilog作业4

1.请自行设计一个模块，完成统计32位2进制数0和1出现的次数。

count.v设计：

`timescale 1ns / 1ps

module count(

input [31:0] innum,

output [5:0] cnt0,

设计思路：

模块有一个输入和两个输出，输入为32位2进制数innum，也是我们要分析的数串。

输出是两个6位2进制数cnt0和cnt1，分别为0出现的次数和1出现的次数。

使用6位2进制数以保证能够保证0-32的所有数。

设计一个always块，每当任意数值发生变化时执行一次，根据当时的innum遍历每一位，统计1的个数，更新寄存器c1，并以此更新cnt0和cnt1。

output [5:0] cnt1

);

reg [5:0] c1,i;

always @(\*)

begin

c1 = 0;

for (i = 0; i < 32; i = i + 1)

begin

if (innum[i] == 1'b1)

c1 = c1 + 1;

end

end

assign cnt0 = 6'd32 - c1;

assign cnt1 = c1;

endmodule

testbench.v设计：

`timescale 1ns / 1ps

module testbench();

reg [31:0]tnum;

设计思路：

寄存器tnum初始化为0，每5秒用random生成32位随机二进制数将其刷新，调用count模块，计算cnt0，cnt1，存入tcnt0和tcnt1中，进行验证。

wire [5:0] tcnt0,tcnt1;

count mycnt(tnum,tcnt0,tcnt1);

initial begin

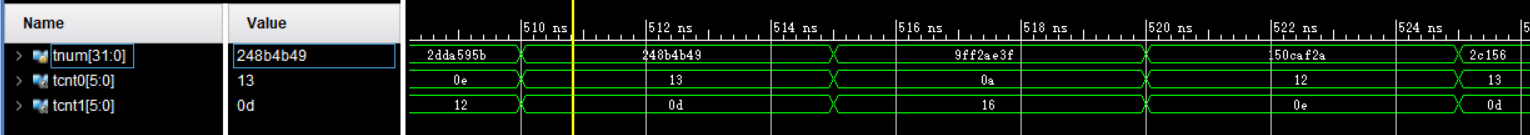
tnum = 32'b0;

end

always #5 tnum=$random;

endmodule

测试结果：



任举两例分析：

510ns：248b4b49 => 0010 0100 1000 1011 0100 1011 0100 1001

共有19（0x13）个0，13（0x0d）个1。

515ns：9ff2ae3f => 1001 1111 1111 0010 1010 1110 0011 1111

共有10（0x0a）个0，22（0x16）个1。通过验证。

2.请使用always块语句，实现一个十分频器，divider10( input clk\_in,input reset, output count, output clk\_out)。其功能可以理解为将时钟降频为原来的10分之一（思路：对clk\_in进行count计数，count取值0~9，count数到5时，clk\_out由1变0，count数到10时自动归零同时clk\_out由0变1）。

divider10.v设计：

`timescale 1ns / 1ps

module divider10(

input clk\_in,

input reset,

设计思路：

每个clk\_in上升沿执行always语句块，若reset为1，count和clk\_out清零。否则，count+1%10，同时若count为4或9翻转clkout(clk\_out会和count的5/0一起到来)。

output reg [3:0] count,

output reg clk\_out

);

always @(posedge clk\_in) begin

if (reset) begin

count <= 4'd0;

clk\_out <= 1'b0;

end else begin

if (count == 4'd4 || count == 4'd9) begin

clk\_out <= ~clk\_out;

end

count <= (count + 1'd1) % 4'd10 ;

end

end

endmodule

testbench.v设计：

`timescale 1ns / 1ps

module testbench();

reg tclkin,treset;

wire tclkout;

wire [3:0] tcnt;

divider10 mydiv(tclkin,treset,tcnt,tclkout);

设计思路：

设计为treset先设为1，2周期后设为0，tclkin设为0，这样保证更好的冷启动。

随后每1周期翻转tclkin，每37周期翻转treset，进行验证。

initial begin

tclkin = 1'b0;

treset = 1'b1;

#2;

treset = 1'b0;

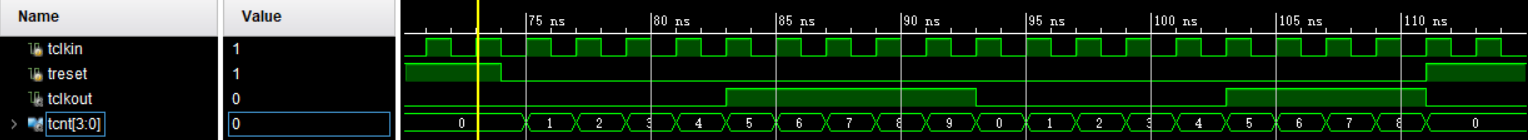
end

always #1 tclkin = ~tclkin;

always #37 treset = ~treset;

endmodule

测试结果：



reset验证：reset为1的时候，tclkout为0，tcnt置0，不受tclkin影响。reset为0的时候，当tcnt数到5的时候tclkout置为1，数到10(0)的时候tclk置为0。图中74ns treset置0，随每ns翻转可以看到tcnt正常计数，到5和0翻转，直到111ns treset置1清零，通过验证。