实验报告 lab04

姓名 林凡琪

学号 211240042

班级 匡亚明学院

邮箱 211240042@smail.nju.edu.cn

实验时间 2022/10/20

实验目的

复习计数器的工作原理，通过介绍几种简单计数器的工作过程和设计方法、以及开发板系统时钟的使用，学习计数器的设计和定时器的工作原理。

实验原理（知识背景）

利用计数器进行时钟信号分频，可以设计出任何我们需要的时钟信号。

实验环境/器材等

Verilog 2022.1

Windows 10

程序代码或流程图

```
22
23
24  module Timer(
25      input CLK100MHZ,
26      input SW,
27      input BTNC,
28      output LED_10,
29      output [7:0] LED,
30      output [7:0] AN,
31      output [7:0] HEX
32      );
33  wire clk_1s, clk_10k;
34  wire rco1, rco2;
35  wire[3:0] q1, q2;
36  clkgen #(1) myls_clk(CLK100MHZ, BTNC, SW, clk_1s);
37  clkgen #(10000) my10k_clk(CLK100MHZ, BTNC,1'b1, clk_10k);
38  counter i1 (
39            .clk(clk_1s), .en(SW), .rst(BTNC), .cnt_limit(4'd10), .Q(q1), .rco(rco1)
40  );
41  counter i2 (
42            .clk(rco1),
43            .en(SW),
44            .rst(BTNC),
45            .cnt_limit(4'd6),
46            .Q(q2),
47            .rco(rco2)
48  );
49  sevenseg my_7seg(clk_10k, 8'h3, {24'b0, q2, q1}, AN, HEX);
50  assign LED[7:0] = {q2, q1};
51  assign LED_10 = clk_1s;
52  endmodule
53
54  module counter(
55      input  clk,
56      input  en,
57      input  rst,
58      input  [3:0] cnt_limit,
59      output reg [3:0] Q,
60      output reg rco
61      );
62
```

顶层设计模块Timer和计数模块counter

Project Summary | lab04.v

H:/DigitalLab/lab04/lab04.srcs/sources_1/new/lab04.v

```
initial
begin
    Q=4'd0;
    rco = 1'b0;
end
always@(posedge clk or posedge rst)
begin
    if(rst)
    begin
        Q<=4'b0;
        rco <= 1'b0;
    end
    else
    if(en)
        begin
            if(Q ==cnt_limit-4'd1)
            begin
                Q<=4'd0;
                rco <= 1'b1;
            end
            else
            begin
                Q <= Q+4'd1;
                rco <= 1'b0;
            end
        end

    end
endmodule



module clkgen(
    input clkin,
    input rst,
    input clken,
    output reg clkout
    //output reg [31:0] clkcount
);
parameter clk_freq=1000;
parameter countlimit=100000000/2/clk_freq-1;
```

counter和clkgen部分

```
parameter clk_freq=1000;
parameter countlimit=100000000/2/clk_freq-1;
reg [31:0] clkcount;
initial begin
clkcount = 32'd0;
clkout = 1'b0;
end
always @ (posedge clkin)

    if(rst)  begin
      clkcount<=0;
      clkout<=1'b0;
      end
     else begin
      if(clken) begin
        if(clkcount>=countlimit)begin
            clkcount<=32'd0;
            clkout<=~clkout;
        end
        else begin
            clkcount<=clkcount+1;
            end
       end
    end
endmodule


module sevenseg(
    input clk,
    input [7:0] en,
    input [31:0] digits,
    output [7:0] an,
    output [7:0] hex
    );
reg [3:0] d;
reg [2:0] s;
wire [7:0] my_an;
initial begin
s = 3'b000;
end
```

clkgen和sevenseg部分

```
always@(s)
    case(s)
        3'd0:d=digits[3:0];
        3'd1:d=digits[7:4];
        3'd2:d=digits[11:8];
        3'd3:d=digits[15:12];
        3'd4:d=digits[19:16];
        3'd5:d=digits[23:20];
        3'd6:d=digits[27:24];
        3'd7:d=digits[31:28];
        default:d=4'd0;
        endcase
always@(posedge clk)
    s <= s + 1;
decode38 dec(s,my_an);
assign an=(en[s]==1'b1)?~my_an:8'hff;
bcd7seg seg(d,hex);

endmodule


module decode38 (
    input [2:0] s,
    output reg [7:0] a
);
always @(s)
        case (s)
          3'b000 : a = 8'b00000001;
          3'b001 : a = 8'b00000010;
          3'b010 : a = 8'b00000100;
          3'b011 : a = 8'b00001000;
          3'b100 : a = 8'b00010000;
          3'b101 : a = 8'b00100000;
          3'b110 : a = 8'b01000000;
          3'b111 : a = 8'b10000000;
       default:  a = 8'b00000000;
       endcase
   //add your code here

endmodule
```

sevenseg和decode38

```
module bcd7seg(
      input  [3:0] b,
      output reg [7:0] h
      );
     always @(*)
        case (b)
          4'd9 : h = 8'b10010000;
          4'd8 : h = 8'b10000000;
          4'd7 : h = 8'b11111000;
          4'd6 : h = 8'b10000010;
          4'd5 : h = 8'b10010010;
          4'd4 : h = 8'b10011001;
          4'd3 : h = 8'b10110000;
          4'd2 : h = 8'b10100100;
          4'd1 : h = 8'b11111001;
          4'd0 : h = 8'b11000000;
       default:  h = 8'b11111111;
       endcase

endmodule
```

bcd7seg模块

```
## Clock signal
set_property -dict { PACKAGE_PIN E3    IOSTANDARD LVCMOS33 } [get_ports { CLK100MHZ }]; #IO_L12P_T1_MRCC_35 Sch=clk100mhz
#create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports {CLK100MHZ}];


##Switches
set_property -dict { PACKAGE_PIN J15   IOSTANDARD LVCMOS33 } [get_ports { SW }]; #IO_L24N_T3_RS0_15 Sch=sw[0]
#set_property -dict { PACKAGE_PIN L16   IOSTANDARD LVCMOS33 } [get_ports { SW[1] }]; #IO_L3N_T0_DQS_EMCCLK_14 Sch=sw[1]
#set_property -dict { PACKAGE_PIN M13   IOSTANDARD LVCMOS33 } [get_ports { SW[2] }]; #IO_L6N_T0_D08_VREF_14 Sch=sw[2]
#set_property -dict { PACKAGE_PIN R15   IOSTANDARD LVCMOS33 } [get_ports { SW[3] }]; #IO_L13N_T2_MRCC_14 Sch=sw[3]
#set_property -dict { PACKAGE_PIN R17   IOSTANDARD LVCMOS33 } [get_ports { SW[4] }]; #IO_L12N_T1_MRCC_14 Sch=sw[4]
#set_property -dict { PACKAGE_PIN T18   IOSTANDARD LVCMOS33 } [get_ports { SW[5] }]; #IO_L7N_T1_D10_14 Sch=sw[5]
#set_property -dict { PACKAGE_PIN U18   IOSTANDARD LVCMOS33 } [get_ports { SW[6] }]; #IO_L17N_T2_A13_D29_14 Sch=sw[6]
#set_property -dict { PACKAGE_PIN R13   IOSTANDARD LVCMOS33 } [get_ports { SW[7] }]; #IO_L5N_T0_D07_14 Sch=sw[7]
#set_property -dict { PACKAGE_PIN T8    IOSTANDARD LVCMOS18 } [get_ports { SW[8] }]; #IO_L24N_T3_34 Sch=sw[8]
#set_property -dict { PACKAGE_PIN U8    IOSTANDARD LVCMOS18 } [get_ports { SW[9] }]; #IO_25_34 Sch=sw[9]
#set_property -dict { PACKAGE_PIN R16   IOSTANDARD LVCMOS33 } [get_ports { SW[10] }]; #IO_L15P_T2_DQS_RDWR_B_14 Sch=sw[10]
#set_property -dict { PACKAGE_PIN T13   IOSTANDARD LVCMOS33 } [get_ports { SW[11] }]; #IO_L23P_T3_A03_D19_14 Sch=sw[11]
#set_property -dict { PACKAGE_PIN H6    IOSTANDARD LVCMOS33 } [get_ports { SW[12] }]; #IO_L24P_T3_35 Sch=sw[12]
#set_property -dict { PACKAGE_PIN U12   IOSTANDARD LVCMOS33 } [get_ports { SW[13] }]; #IO_L20P_T3_A08_D24_14 Sch=sw[13]
#set_property -dict { PACKAGE_PIN U11   IOSTANDARD LVCMOS33 } [get_ports { SW[14] }]; #IO_L19N_T3_A09_D25_VREF_14 Sch=sw[14]
#set_property -dict { PACKAGE_PIN V10   IOSTANDARD LVCMOS33 } [get_ports { SW[15] }]; #IO_L21P_T3_DQS_14 Sch=sw[15]

## LEDs
set_property -dict { PACKAGE_PIN H17   IOSTANDARD LVCMOS33 } [get_ports { LED[0] }]; #IO_L18P_T2_A24_15 Sch=led[0]
set_property -dict { PACKAGE_PIN K15   IOSTANDARD LVCMOS33 } [get_ports { LED[1] }]; #IO_L24P_T3_RS1_15 Sch=led[1]
set_property -dict { PACKAGE_PIN J13   IOSTANDARD LVCMOS33 } [get_ports { LED[2] }]; #IO_L17N_T2_A25_15 Sch=led[2]
set_property -dict { PACKAGE_PIN N14   IOSTANDARD LVCMOS33 } [get_ports { LED[3] }]; #IO_L8P_T1_D11_14 Sch=led[3]
set_property -dict { PACKAGE_PIN R18   IOSTANDARD LVCMOS33 } [get_ports { LED[4] }]; #IO_L7P_T1_D09_14 Sch=led[4]
set_property -dict { PACKAGE_PIN V17   IOSTANDARD LVCMOS33 } [get_ports { LED[5] }]; #IO_L18N_T2_A11_D27_14 Sch=led[5]
set_property -dict { PACKAGE_PIN U17   IOSTANDARD LVCMOS33 } [get_ports { LED[6] }]; #IO_L17P_T2_A14_D30_14 Sch=led[6]
set_property -dict { PACKAGE_PIN U16   IOSTANDARD LVCMOS33 } [get_ports { LED[7] }]; #IO_L18P_T2_A12_D28_14 Sch=led[7]
#set_property -dict { PACKAGE_PIN T15   IOSTANDARD LVCMOS33 } [get_ports { LED[9] }]; #IO_L14N_T2_SRCC_14 Sch=led[9]
set_property -dict { PACKAGE_PIN U14   IOSTANDARD LVCMOS33 } [get_ports { LED_10 }]; #IO_L22P_T3_A05_D21_14 Sch=led[10]
#set_property -dict { PACKAGE_PIN T16   IOSTANDARD LVCMOS33 } [get_ports { LED[11] }]; #IO_L15N_T2_DQS_DOUT_CSO_B_14 Sch=led[11]
#set_property -dict { PACKAGE_PIN V15   IOSTANDARD LVCMOS33 } [get_ports { LED[12] }]; #IO_L16P_T2_CSI_B_14 Sch=led[12]
```

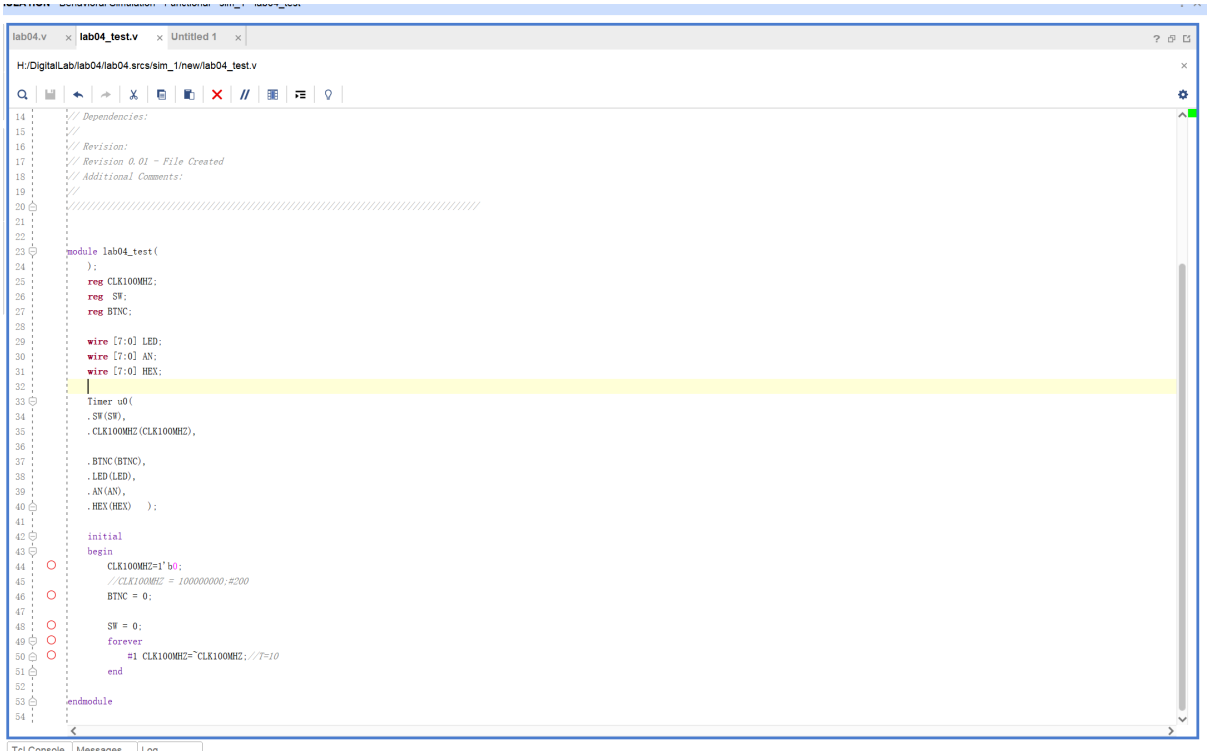
缺省文件1/2

```
##7 segment display
set_property -dict { PACKAGE_PIN T10   IOSTANDARD LVCMOS33 } [get_ports { HEX[0] }]; #IO_L24N_T3_A00_D16_14 Sch=ca
set_property -dict { PACKAGE_PIN R10   IOSTANDARD LVCMOS33 } [get_ports { HEX[1] }]; #IO_25_14 Sch=cb
set_property -dict { PACKAGE_PIN K16   IOSTANDARD LVCMOS33 } [get_ports { HEX[2] }]; #IO_25_15 Sch=cc
set_property -dict { PACKAGE_PIN K13   IOSTANDARD LVCMOS33 } [get_ports { HEX[3] }]; #IO_L17P_T2_A26_15 Sch=cd
set_property -dict { PACKAGE_PIN P15   IOSTANDARD LVCMOS33 } [get_ports { HEX[4] }]; #IO_L13P_T2_MRCC_14 Sch=ce
set_property -dict { PACKAGE_PIN T11   IOSTANDARD LVCMOS33 } [get_ports { HEX[5] }]; #IO_L19P_T3_A10_D26_14 Sch=cf
set_property -dict { PACKAGE_PIN L18   IOSTANDARD LVCMOS33 } [get_ports { HEX[6] }]; #IO_L4P_T0_D04_14 Sch=cg
set_property -dict { PACKAGE_PIN H15   IOSTANDARD LVCMOS33 } [get_ports { HEX[7] }]; #IO_L19N_T3_A21_VREF_15 Sch=dp
set_property -dict { PACKAGE_PIN J17   IOSTANDARD LVCMOS33 } [get_ports { AN[0] }]; #IO_L23P_T3_FOE_B_15 Sch=an[0]
set_property -dict { PACKAGE_PIN J18   IOSTANDARD LVCMOS33 } [get_ports { AN[1] }]; #IO_L23N_T3_FWE_B_15 Sch=an[1]
set_property -dict { PACKAGE_PIN T9    IOSTANDARD LVCMOS33 } [get_ports { AN[2] }]; #IO_L24P_T3_A01_D17_14 Sch=an[2]
set_property -dict { PACKAGE_PIN J14   IOSTANDARD LVCMOS33 } [get_ports { AN[3] }]; #IO_L19P_T3_A22_15 Sch=an[3]
set_property -dict { PACKAGE_PIN P14   IOSTANDARD LVCMOS33 } [get_ports { AN[4] }]; #IO_L8N_T1_D12_14 Sch=an[4]
set_property -dict { PACKAGE_PIN T14   IOSTANDARD LVCMOS33 } [get_ports { AN[5] }]; #IO_L14P_T2_SRCC_14 Sch=an[5]
set_property -dict { PACKAGE_PIN K2    IOSTANDARD LVCMOS33 } [get_ports { AN[6] }]; #IO_L23P_T3_35 Sch=an[6]
set_property -dict { PACKAGE_PIN U13   IOSTANDARD LVCMOS33 } [get_ports { AN[7] }]; #IO_L23N_T3_A02_D18_14 Sch=an[7]

##CPU Reset Button
#set_property -dict { PACKAGE_PIN C12   IOSTANDARD LVCMOS33 } [get_ports { CPU_RESETN }]; #IO_L3P_T0_DQS_AD1P_15 Sch=cpu_resetn

##Buttons
set_property -dict { PACKAGE_PIN N17   IOSTANDARD LVCMOS33 } [get_ports { BTNC }]; #IO_L9P_T1_DQS_14 Sch=btnc
#set_property -dict { PACKAGE_PIN M18   IOSTANDARD LVCMOS33 } [get_ports { BTNU }]; #IO_L4N_T0_D05_14 Sch=btnu
#set_property -dict { PACKAGE_PIN P17   IOSTANDARD LVCMOS33 } [get_ports { BTNL }]; #IO_L12P_T1_MRCC_14 Sch=btnl
#set_property -dict { PACKAGE_PIN M17   IOSTANDARD LVCMOS33 } [get_ports { BTNR }]; #IO_L10N_T1_D15_14 Sch=btnr
#set_property -dict { PACKAGE_PIN P18   IOSTANDARD LVCMOS33 } [get_ports { BTND }]; #IO_L9N_T1_DQS_D13_14 Sch=btnd
```

缺省文件2/2

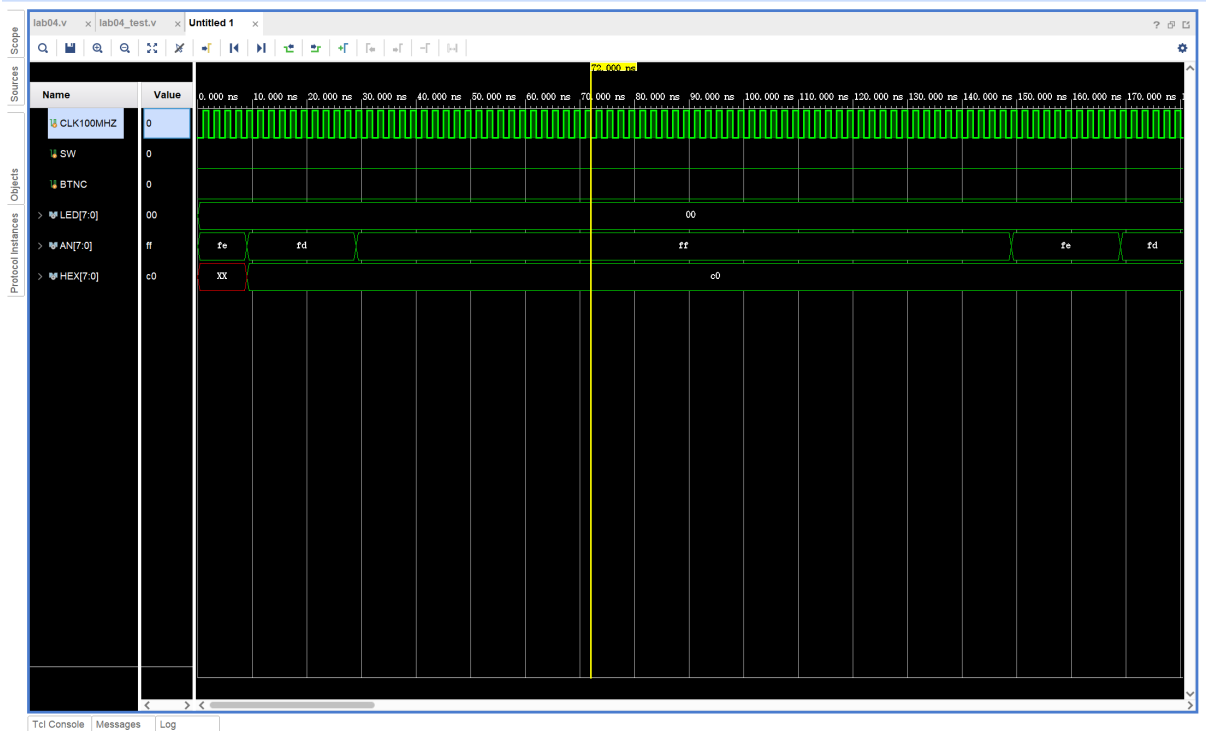


测试激励代码

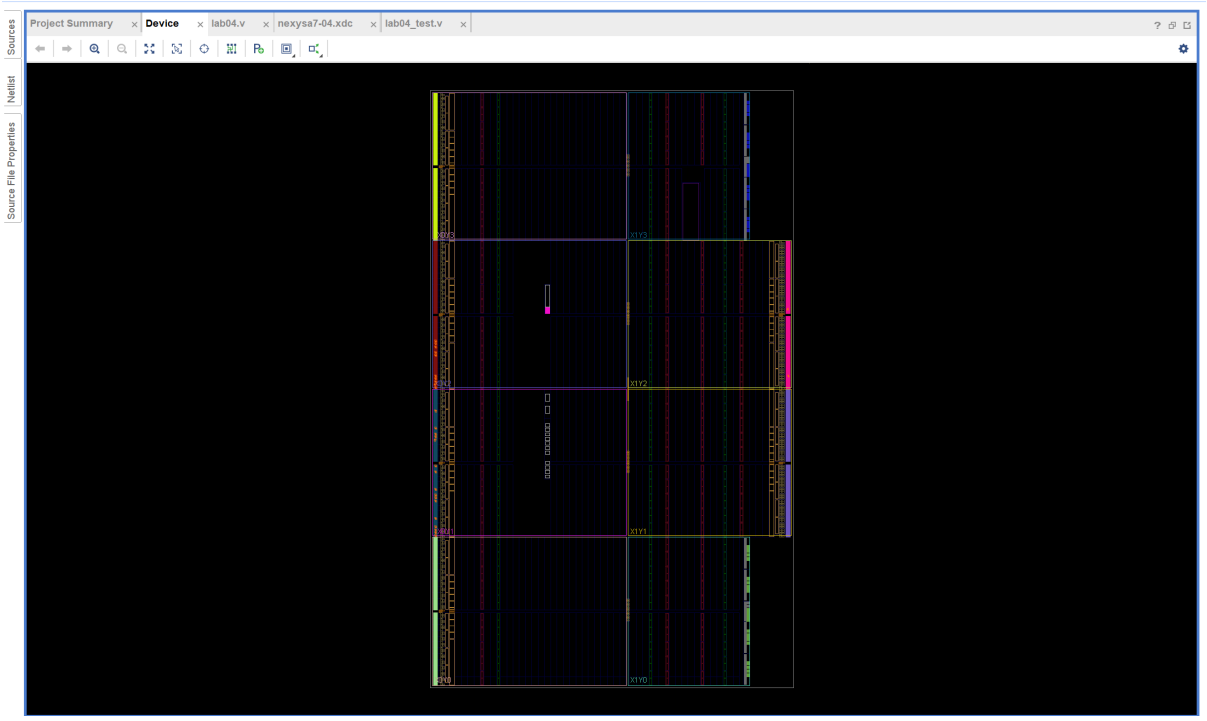
实验步骤/过程

源程序代码如上

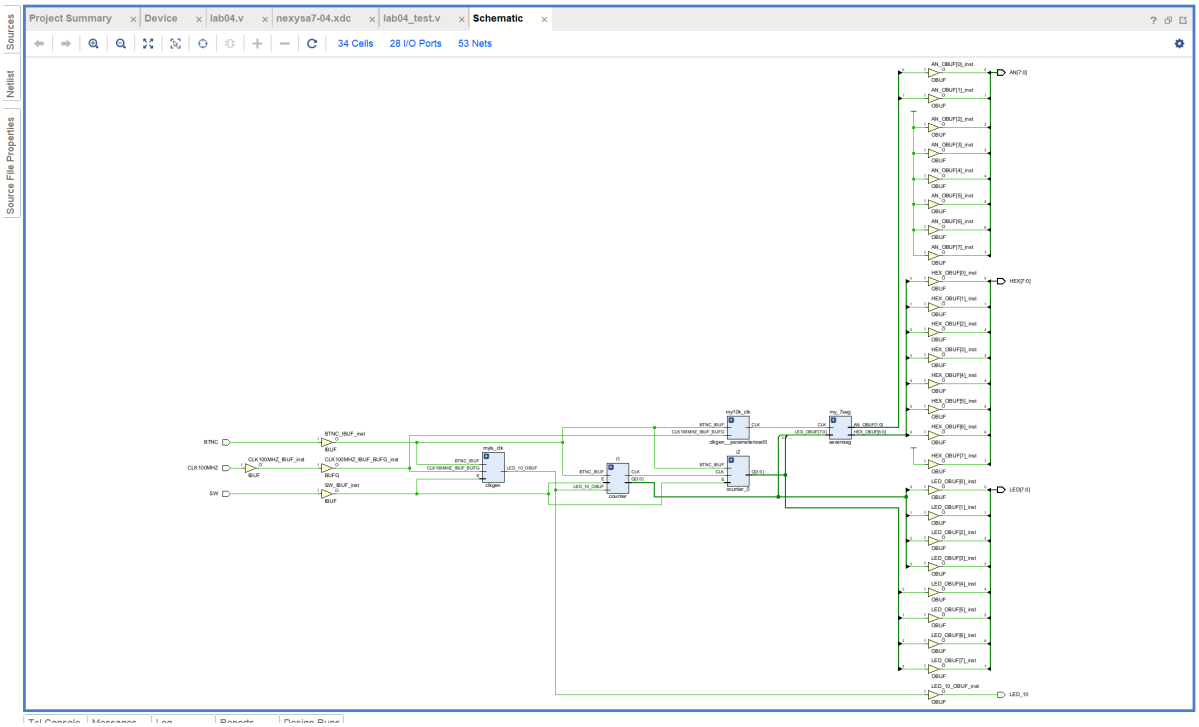
仿真模拟



Run Synthesis后

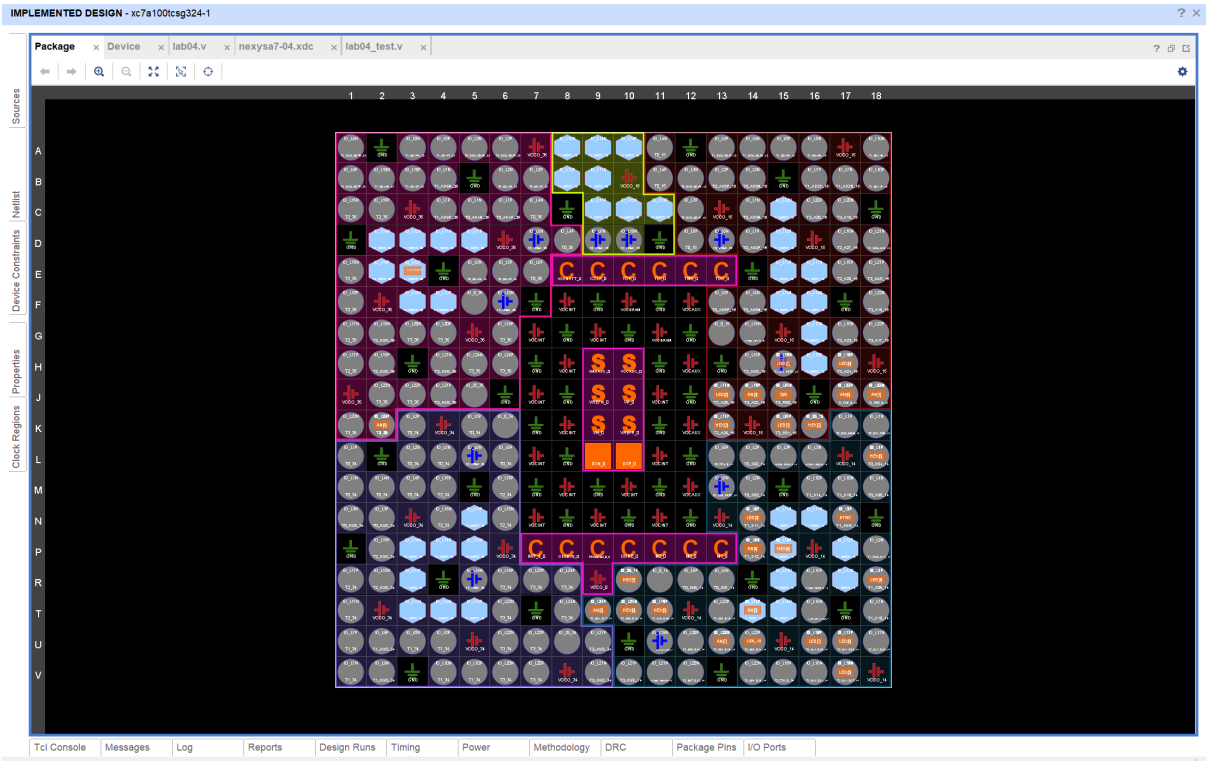


Device



电路图

Run Implementation后



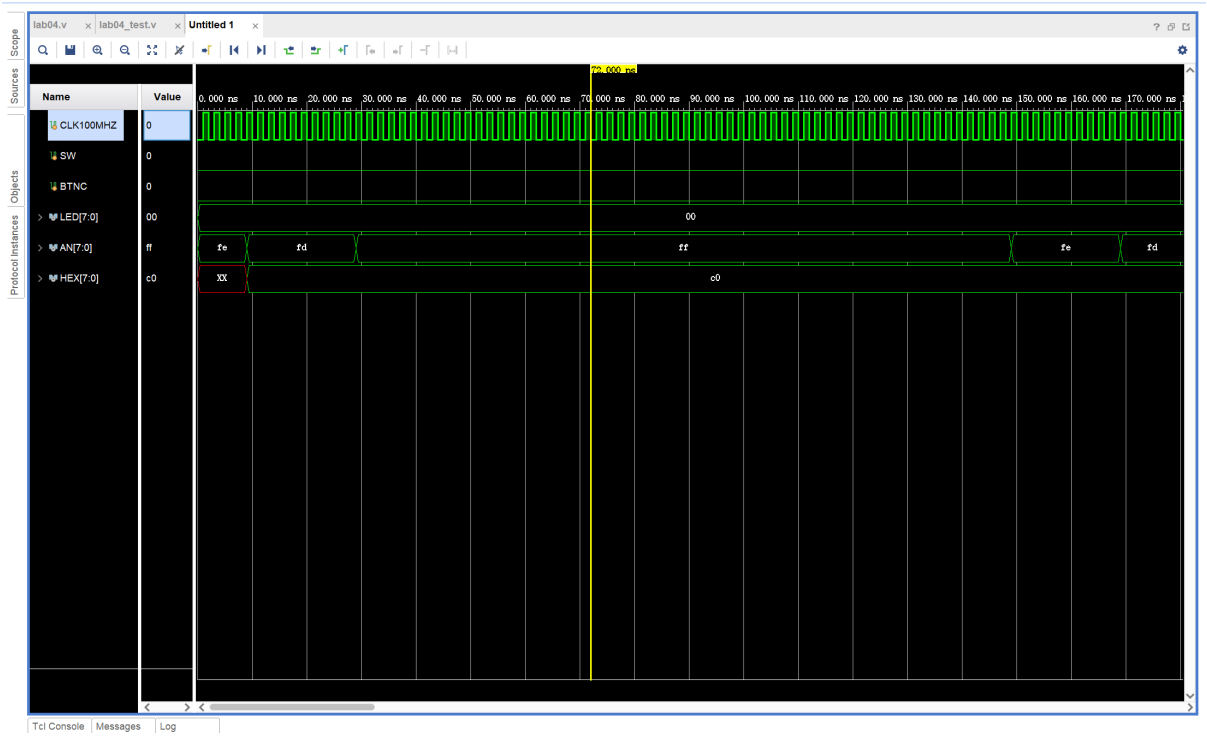
Package

生成二进制流文件，上板（验收可见）

测试方法

用testbench并且上版测试

实验结果



仿真模拟

其他见上

实验中遇到的问题及解决办法

模拟总是失败，全是XXXX，即使初始化了也没有用。

模拟时长限制导致需要更改分频

实验得到的启示

多提问

意见和建议