数字逻辑与计算机组成实验

（时序逻辑电路设计）lab04：

计数器和时钟

姓名：郑凯琳

学号：205220025

邮箱：205220025@smail.nju.edu.cn

（一） 实验目的

在 Nexys A7-100T 开发板上实现一个计时器，在七段数码管上直接以十 进制显示。

利用开发板上的频率为 100MHz 的时钟，请先设计一个分频器，其输入为 100MHz 的时钟，输出为一个频率为 1Hz，周期为 1 秒的时钟信号。再用这个新的频率为 1Hz 的时钟信号作为你设计的时钟信号，进行计数。

要求此计时器有开始、暂停和清零功能，要求从 00 计数到 59，计数值到 60 后重新从零开始计数。在数码管上用两位数字显示。

可以在计时结束的时候让某一个发光二极管闪烁一个时钟周期，提示计时结束。

（二） 实验原理

计数器是数字电路中广泛使用的逻辑部件，是时序逻辑电路中最重要的逻辑部件之一。

1. 功能：
   - 对输入脉冲的个数进行计数
   - 分频、定时、产生节拍脉冲等
2. 分类：
   - 按功能：加法计数器、减法计数器、既具有加法又有减法的可逆计数器
   - 按计数进制：二进制计数器、十进制计数器、任意进制计数器

（三） 实验环境/器材等

硬件器材：Nexys A7-100T 开发板
软件平台：Vivado 开发平台

（四） 实验过程

<u>数字抽象</u>

1. 输入：
   - clk ：时钟信号，与分频器的输出时钟信号连接
   - en ：使能端
   - stop ：暂停信号
   - reset ：清零信号
2. 输出：
   - endone ：一轮计数结束的标志（为 1，持续 1s）
   - [7:0] AN ：七段 LED 数码管
   - [6:0] hex ：数码管上的 LED

设计思路 & 设计代码

1. 设置变量并初始化为 0

```
reg [3:0] h;
reg [3:0] l;
reg [6:0] counter;
reg clk_1s;
reg [31:0]count_clk;

initial
begin
h = 0;
l = 0;
counter = 0;
clk_1s = 0;
count_clk = 0;
end
```

- h : 十位
- l : 个位
- counter : 计数的真值

2. 生成 1 秒时钟

```
always @ (posedge clk)
    if(count_clk == 49999999)
    begin
        count_clk <= 0;
        clk_1s <= ~clk_1s;
    end
    else
        count_clk <= count_clk + 1;
```

3. 对每次 clk 上升沿时进行判断

```
    if(!en)
    begin
        h = 0;
        l = 0;
        counter = 0;
    end
    else
    begin
        endone = 0;

        if(reset)
        begin
            h = 0;
            l = 0;
            counter = 0;
        end

        else if(stop)
        begin
            h = h;
            l = l;
            counter = counter;
        end
        else
        begin
            if(counter == 59)
            begin
                endone = counter % 2;
                counter = 0;
            end
            else
                counter = counter + 1;

            if(counter < 60)
            begin
                l = counter % 10;
                h = (counter - (counter % 10)) / 10;
            end
            else
            begin
                l = l;
                h = h;
            end
        end
    end
end
```

采用模 60 运算来计算 counter 以达到在 60 内循环计数的效果。

4. 数码管显示

```
always @ (*)
begin
if(clk_1ms)
begin
    AN = 8'b11111101;
    case(h)
              4'b0000: hex = 7'b1000000;
              4'b0001: hex = 7'b1111001;
              4'b0010: hex = 7'b0100100;
              4'b0011: hex = 7'b0110000;
              4'b0100: hex = 7'b0011001;
              4'b0101: hex = 7'b0010010;
              4'b0110: hex = 7'b0000010;
              4'b0111: hex = 7'b1111000;
              4'b1000: hex = 7'b0000000;
              4'b1001: hex = 7'b0010000;
              endcase
end
else
begin
    AN = 8'b11111110;
              case(l)
              4'b0000: hex = 7'b1000000;
              4'b0001: hex = 7'b1111001;
              4'b0010: hex = 7'b0100100;
              4'b0011: hex = 7'b0110000;
              4'b0100: hex = 7'b0011001;
              4'b0101: hex = 7'b0010010;
              4'b0110: hex = 7'b0000010;
              4'b0111: hex = 7'b1111000;
              4'b1000: hex = 7'b0000000;
              4'b1001: hex = 7'b0010000;
              endcase
end
end
```

测试代码

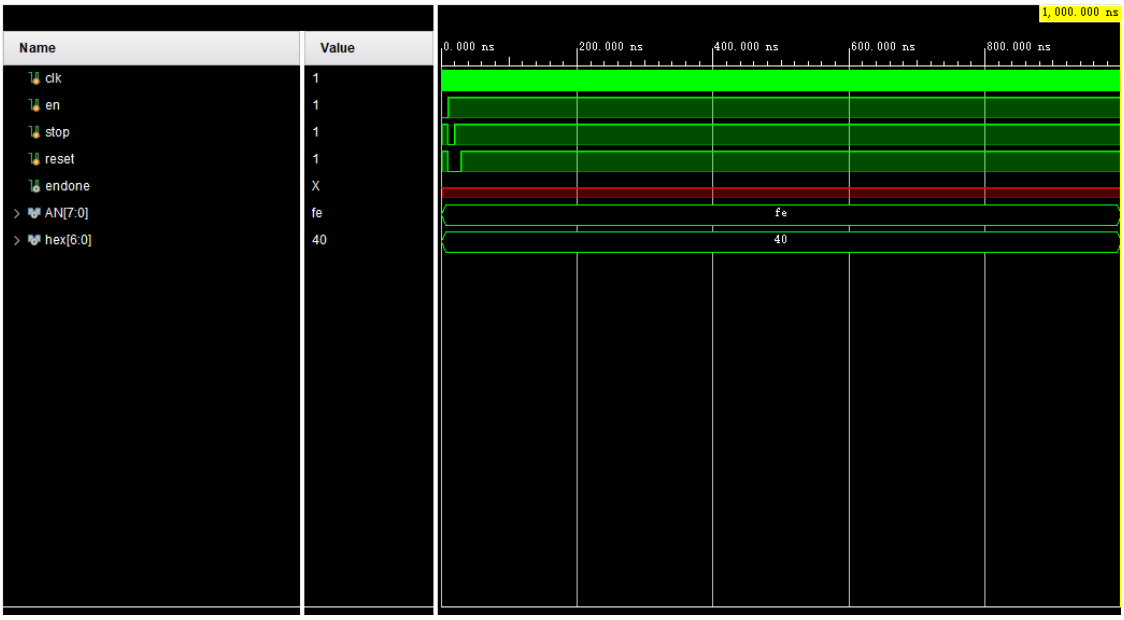
```
lab04_timer t1(
   .clk(clk),
   .en(en),
   .stop(stop),
   .reset(reset),
   .endone(endone),
   .AN(AN),
   .hex(hex));

initial
begin
   clk = 0;
   en = 0; stop = 1; reset = 1; #10;
   en = 1; stop = 0; reset = 0; #10;
   en = 1; stop = 1; reset = 0; #10;
   en = 1; stop = 1; reset = 1; #10;
$display("Running testbench");
end

always
begin
   clk = ~clk; #1;
end
```

硬件实现（引脚分配)

```
## Clock signal
set_property -dict { PACKAGE_PIN E3    IOSTANDARD LVCMOS33 } [get_ports { clk }]; #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 { en }]; #IO_L24N_T3_RS0_15 Sch=sw[0]
set_property -dict { PACKAGE_PIN L16   IOSTANDARD LVCMOS33 } [get_ports { stop }]; #IO_L3N_T0_DQS_EMCCLK_14 Sch=sw[1]
set_property -dict { PACKAGE_PIN M13   IOSTANDARD LVCMOS33 } [get_ports { reset }]; #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 { endone }]; #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 V16   IOSTANDARD LVCMOS33 } [get_ports { LED[8] }]; #IO_L16N_T2_A15_D31_14 Sch=led[8]
#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]
#set_property -dict { PACKAGE_PIN V14   IOSTANDARD LVCMOS33 } [get_ports { LED[13] }]; #IO_L22N_T3_A04_D20_14 Sch=led[13]
#set_property -dict { PACKAGE_PIN V12   IOSTANDARD LVCMOS33 } [get_ports { LED[14] }]; #IO_L20N_T3_A07_D23_14 Sch=led[14]
#set_property -dict { PACKAGE_PIN V11   IOSTANDARD LVCMOS33 } [get_ports { LED[15] }]; #IO_L21N_T3_DQS_A06_D22_14 Sch=led[15]

## 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 { DP }]; #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]
```

（五） 实验结果

仿真结果:



（六） 实验中遇到的问题及解决方法

问题：只有个位数码管在计数，十位数码管没有计数（无法同时显示两位数）

解决方法：动态显示

如果轮流的时间足够短，也就是两个数码管切换的足够快，根据人眼的视觉残留原理，看起来，十位和个位两个字符就是同时在两个数码管上被显示的了。

生成 1ms 时钟:

```
reg clk_1ms;
reg [31:0]count_clk2;

always @ (posedge clk)
if(count_clk2 == 4999)
begin
    count_clk2 <= 0;
    clk_1ms <= ~clk_1ms;
end
else
    count_clk2 <= count_clk2 + 1;
```