

# 数字逻辑与计算机组成实验

---

- 实验名称: lab4 计数器和时钟
- 院系: 计算机科学与技术系
- 学生姓名: 徐简
- 学号: 202220013
- 班级: 数字逻辑与计算机组成实验1班
- 邮箱: [161200063@mail.nju.edu.cn](mailto:161200063@mail.nju.edu.cn)
- 实验时间: 2022 年 3 月 30 日

# 数字逻辑与计算机组成实验Lab4

202220013 徐简 [161200063@smail.nju.edu.cn](mailto:161200063@smail.nju.edu.cn)

## 一, 实验目的

- 复习计数器的相关知识
- 以计数器为基础设计一个闹钟
- 学习Verilog中clk的使用

## 二, 实验原理

利用触发器可以构成简单的计数器。如果在计数器的时钟输入端输入一个固定周期的时钟，那么计数器就变成了定时器。

在实验原理部分，简要的回答一下思考题“为了能满足 0~24999999 的计数要求，变量 “count\_clk”的宽度如何设定？”

- 因为 $2^{24} < 24999999 < 2^{25}$ ，所以count\_clk的宽度至少25位。

## 三, 实验环境/器材

- Quartus
- DE10-Standard开发平台
- FPGA开发板

## 四, 程序代码/流程图

### 计数器

首先构造1s的时钟信号，参考手册代码即可。

```
1 always @(posedge clk)
2   if(count_clk==24999999)
3     begin
4       count_clk <=0;
5       clk_1s <= ~clk_1s;
6     end
7   else
8     count_clk <= count_clk+1;
```

下面介绍一下计数器的设计。输入是时钟，清零，暂停信号，分别用clk，开关实现，输出是暂停标志、结束标志和时间显示，分别用两个LED灯和两组数码管来实现。

因此设计的逻辑如下：

- 初始化信号

```
initial begin
//init clk,sign
end
```

2. 每当时钟上升沿到来，计数加1，根据计数分别设置个位和十位的数码管。这里需要考虑清零和暂停，只需要在不同的always模块分别处理，并进行条件判断即可。

```
// end count
always @ (count or pause)
    if (count == 99 && ~pause)
        sign_end = 1;
    else
        sign_end = 0;
//pause count
always @ (pause)
    if (pause) sign_pause = 1;
    else      sign_pause = 0;
//clear or add
always @ (posedge clk_1s) begin
    if (clear) count <= 0;
    else if (~pause && count == 99) count <= 0;
    else if (~pause) count <= count + 1;
    else count <= count;

    case(count/10)://十位
    case(count%10)://个位
```

## 闹钟

下面基于上面实现的计数器，介绍闹钟的设计，类似的部分就不再重复。

闹钟需要实现的功能有，修改时间，记录设置的闹钟，显示闹钟，显示当前的时间。考虑到开发板的输入有限：

- 使用button作为修改时间的按钮，修改的方式为，修改使能有效时，按hour/min按钮，hour/min加一，按second按钮，second加5。
- 使用Switch作为闹钟时间输入，因为位数不够，使用1位作为使能位，5位表示24小时，4位表示60分钟（以5分钟为单位）。相应的在模块中，用变量记录Switch表示的时间。
- 使用LED表示闹钟时间到，当计数器时间在记录的闹钟时间以及向后三分钟内，LED亮表示闹钟有效。
- 使用六组数码管分别显示时分秒的十位个位。

```
always @ (alarm[9] or count_h or count_min or count_s)
    if (alarm[9]) begin
        alarm_h = alarm[8:4];
        alarm_min = alarm[3:0];
        alarm_min = alarm_min * 5;//单位是5分钟
        if (alarm_h == count_h
            && count_min >= alarm_min
            && count_min < alarm_min + 3)//闹钟持续时间是3分钟
            sign_alarm = 1;
        else
            sign_alarm = 0;
    end else
        sign_alarm = 0;
```

```

always @ (posedge clk_1s) begin
    if (~set_time[3]) begin//when in set mode, time don't change
        if (~set_time[2]) count_h <= (count_h + 1) % 24;
        if (~set_time[1]) count_min <= (count_min + 1) % 60;
        if (~set_time[0]) count_s <= (count_s + 5) % 60;
    end
    else
        begin
            //正常的时间增长
        end
end

```

## 五, 实验步骤/过程

- 使用DE10软件建立工程，这样可以自动进行引脚分配
- 编写计数器和闹钟的代码，并在项目中实例化
- 进行测试
- 上板验证

## 六, 测试方法

- 仿真：为了测试的方便，将count\_clk的周期从24999999，改成了5。简单了测试了一下暂停和清零能否正常工作，发现没问题就上板进行详细的功能测试了。

```

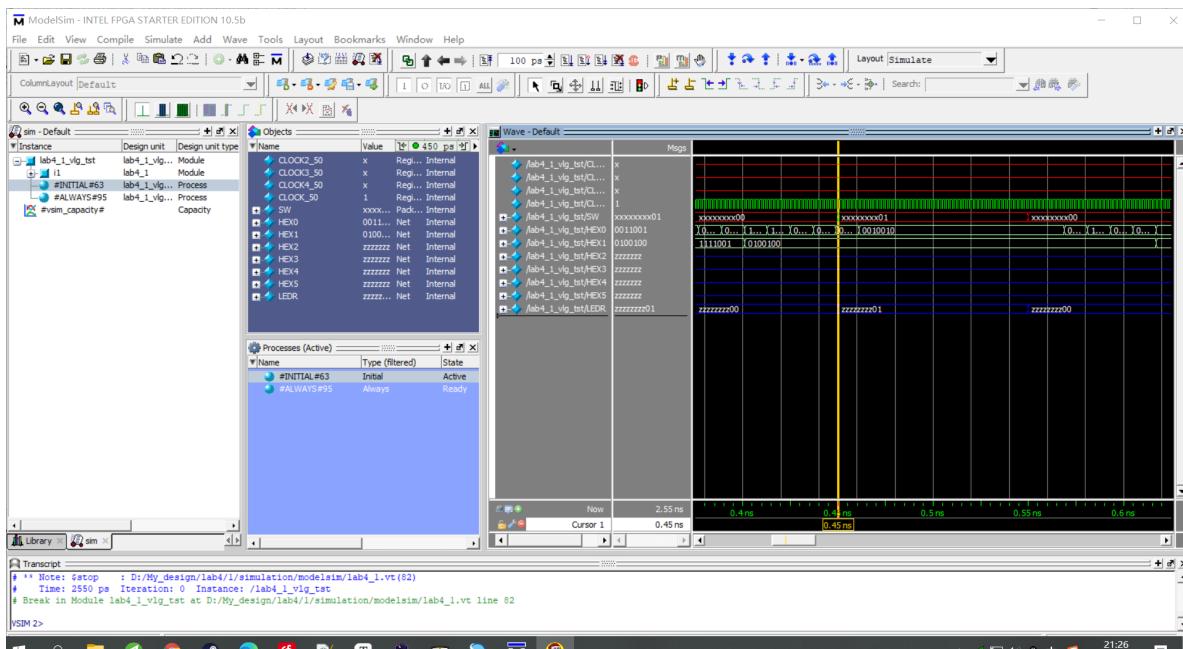
begin
    CLOCK_50 = 0;
    SW[0] = 0; // pause
    SW[1] = 0; // clear
    #100;
    SW[1] = 1;
    #100;
    SW[1] = 0;
    #100;
    SW[0] = 1;
    #100;
    SW[0] = 0;
    #2000;
    $stop;
$display("Running testbench");
end

```

- 上板测试：写入fpga开发板，根据功能测试输入输出的情况

## 七, 实验结果

软件仿真

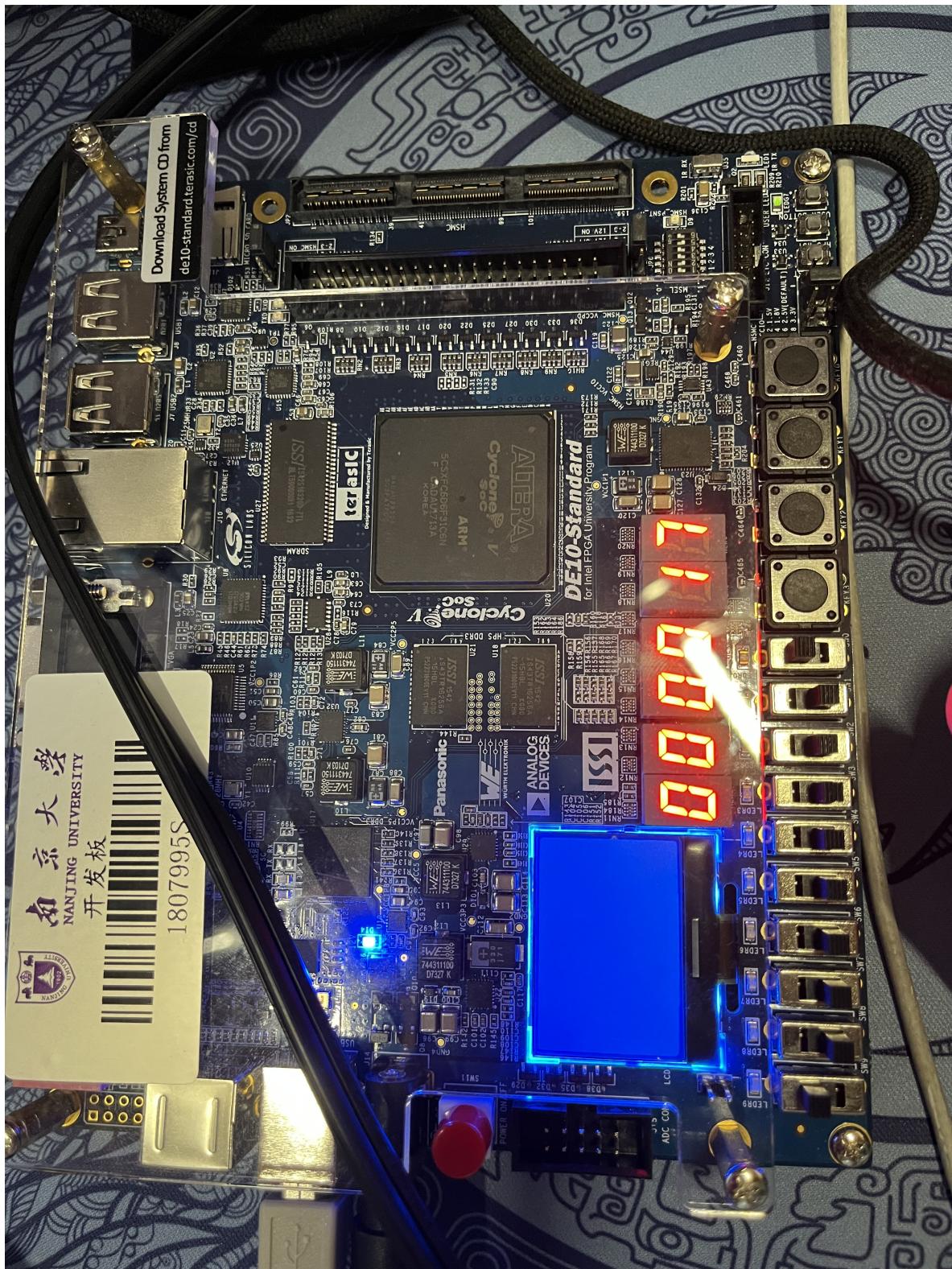


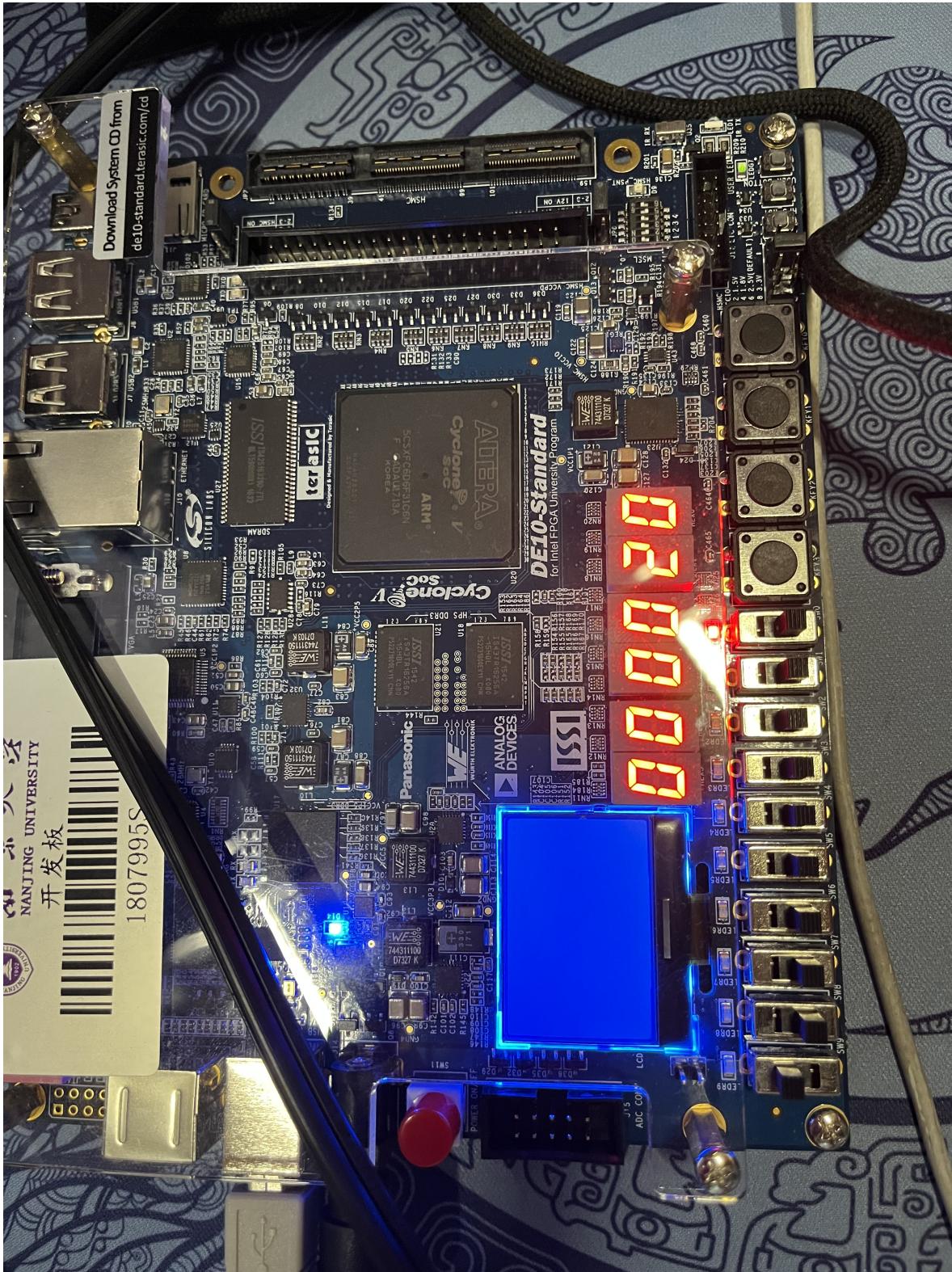
FPGA上板 (视频只上传了计数器, 闹钟演示太长, 文件太大了orz)

计数器



闹钟





## 八, 遇到的问题和解决办法

- 应该选择非阻塞赋值
- 在最开始的时候，要进行变量初始化操作

## 九, 实验得到的启示

## 十, 意见和建议

- 分而治之，并复用原先的时钟以及数码管实现