《数字电路与数字系统实验》实验报告

第 6 次实验： 计数器和时钟

姓名： 张逸凯

学号： 171840708

院系: 物理 学院

邮箱： 645064582@qq.com

电话： 18051988316

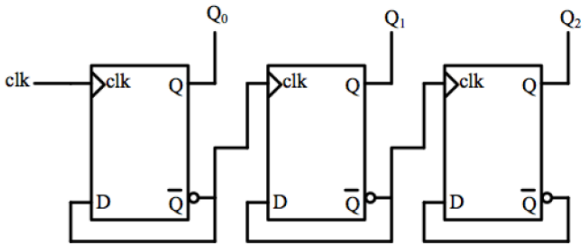
实验时间: 2019 年 4 月 17 日

〇． 预习部分

基本定义

1．加法计数器

利用触发器可以构成简单的计数器。图 5-1是由 3 个上升沿触发的 D 触发器组成的 3 位二进制异步加法计数器，即在每个 Clock 的上升沿，计数器输出 $Q_2Q_1Q_0$ 加 1。



2．减法计数器

利用 D 触发器同样可以构成减法计数器，图 5_3是由 3 个上升沿触发的 D 触发器组成的 3 位二进制异步减法计数器

图 5-4是此 3 位二进制异步减法计数器的状态转移图。

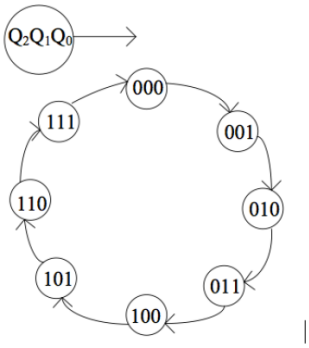


图 5-2: 3 位二进制加法计数器状态图

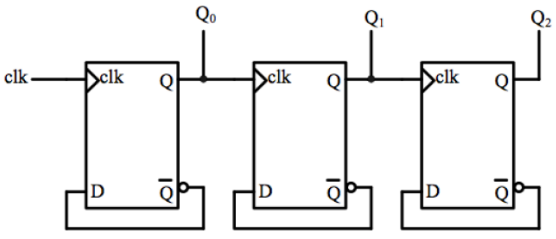


图 5-3: 3 位二进制异步减法计数器

带使能端的 3 位二进制加法计数器代码也很好理解，注意非阻塞语句的使用

表 5-1: 3 位二进制带使能端的加法计数器代码

```
module vminus3(clk,en,out_q);
  input  clk;
  input  en;
  output reg [2:0] out_q;

  always @ (posedge clk)
     if (en)   out_q <= out_q -1;
     else   out_q <= 0;
endmodule
```

开发板上的时钟信号：

DE-10 Standard 开发板为 Cyclone V SOC FPGA 提供了四个频率为 50MHz 的外部输入时钟，这些时钟均可供用户使用。另外还给开放平台上的 HPS 提供了一个 25MHz 的时钟，如图 5-6所示。

后面实验要注意引脚分配问题：掏出手册！

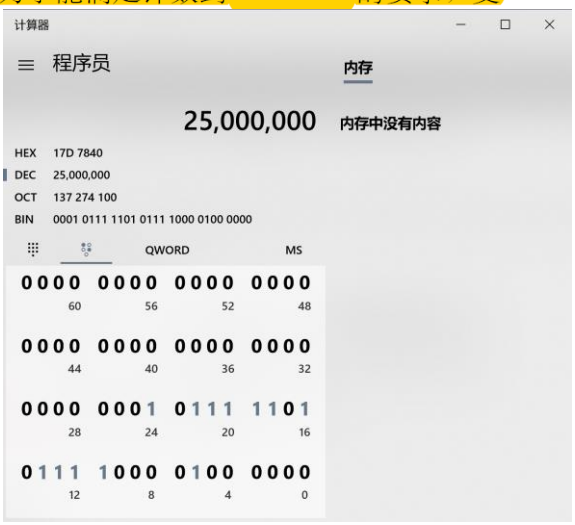
思考题：

表 5-2: 1 秒时钟生成代码

```
always @(posedge clk)
        if(count_clk==25000000)
        begin
                count_clk <=0;
                clk_1s <= ~clk_1s;
        end
        else
                count_clk <= count_clk+1;
```

思考题:

请阅读、理解此段代码，并请思考为了能满足计数到 25000000 的要求，变量“count_clk”的宽度如何设定?



通过计算器我们可以发现 25,000,000 需要 25 来存，这里 count_clk 的宽度应该就是表成二进制数的时候的长度？为了避免溢出，count_clk 必须是 25 位的.

开始实验部分

〇． 实验目的

1．复习计数器的工作原理．

2．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。

3．学习FPGA开发平台上时钟源的使用，并结合计数器的设计方法学习定时器的设计．

4．在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。

一． 实验原理（ 知识背景，结合理论课总结 ）

- **基本定义：**

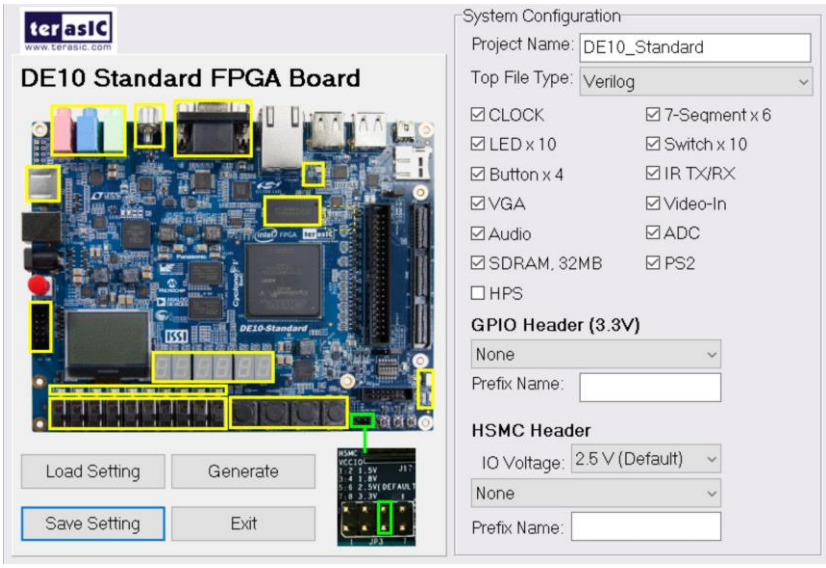
先设计一个分频器 **注意 PDF 里的代码提示：**

利用开发板上的频率为 50MHz 的时钟，请先设计一个分频器，其输入为 50MHz 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

这个有了 PPT 的提示，不难实现：

```
15      always @ (posedge clk)
16      begin
17          if(count_clk == 25000000)
18          begin
19              count_clk = 0;
20              clk_1s = ~clk_1s;
21          end
22          else
23              count_clk = count_clk + 1;
24      end
```

本次实验用了 System Builder，引脚分配全自动(滑稽)，希望不要出什么岔子..



计数器相关：

计数器是数字系统中用的较多的基本逻辑器件，它的基本功能是统计时钟脉冲的个数，即实现计数操作，它也可用与分频、定时、产生节拍脉冲和脉冲序列等。本次实验是利用 FPGA 开发板上提供的外部输入时钟，形成一个自己的周期一定的时钟，再利用这个时钟来进行接下来的实验。

二． 实验设备环境

硬件器材：FPGA 开发板．

软件平台：Qaurtus 开发平台．

三． 实验步骤 / 过程（设计思路、设计代码、测试代码、仿真结果和硬件实现等的截图代码等）

- **设计思路：**

基于 PPT 例子（和查找资料）

```
module Timer(clk, en, stop, clear, endone, seg0, seg1);
   input clk;
   input en;
   input stop;
   input clear;
   output reg endone;
   output reg [6:0] seg0;
   output reg [6:0] seg1;
   reg [3:0] h = 0;
   reg [3:0] l = 0;
   reg [6:0] counter = 0;
   reg clk_1s = 0;
   reg [24:0] count_clk = 0;

   always @ (posedge clk)
   begin
      if(count_clk == 25000000)
      begin
         count_clk = 0;
         clk_1s = ~clk_1s;
      end
      else
         count_clk = count_clk + 1;
   end

   always @ (posedge clk_1s)
   begin
   if (en)
   begin
      endone = 0;
      if (clear)
      begin
         counter = 0;
         l = 0;
         h = 0;
      end

      else if (stop)
      begin
         counter = counter;
         l = l;
         h = h;
```

```
43          end
44
45          else
46          begin
47              if (counter > 98)
48              begin
49                  endone = counter % 2;
50                  counter = counter + 1;
51              end
52              else
53              begin
54              counter = counter + 1;
55              end
56
57              if (counter < 100)
58              begin
59                  l = counter % 10;
60                  h = ((counter - counter % 10) / 10);
61              end
62              else
63              begin
64                  l = l;
65                  h = h;
66              end
67          end
68
69          case(h)
81          case(l)
93      end
94
95      else if (!en)
96      begin
97          counter = 0;
98          l = 0;
99          h = 0;
100         seg0 = 7'b1000000;
101         seg1 = 7'b1000000;
102     end
103 end
104 endmodule
```

设计思路：

先将 clk 接到 FPGA 提供的 50MHz 外部输入时钟，所以 clk_1s 每隔 0.5s 就会变化一次，所以 clk_1s 的变化周期是一秒钟，利用这个思想也可以实现其它类似的定时器了。

我们可以设置一个变量，每当这个一秒的时钟来临一次以后就将这个变量加一，再将计数器的值显示到七段管上，即可完成 99 计数器的设计了。

需要暂停就是冻结当前时刻，这个很简单不要变就可以了：

```
else if (stop)
begin
    counter = counter;
    l = l;
    h = h;
end
```

当 count 大于 98 的时候我们设置一个输出，作为 LED 随着时钟信号闪烁就可以了，其他时候：

```
if (counter < 100)
begin
    l = counter % 10;
    h = ((counter - counter % 10) / 10);
end
else
begin
    l = l;
    h = h;
end
```

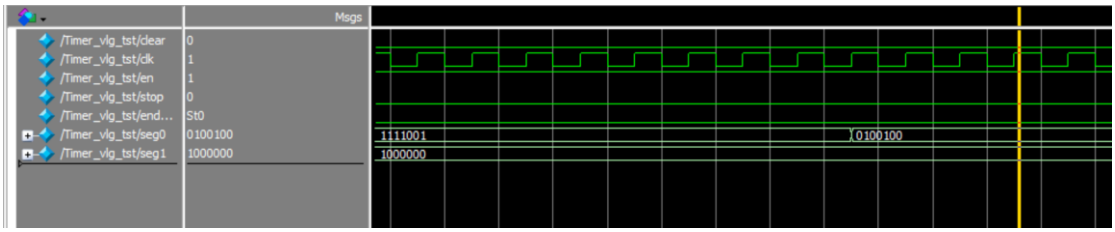
把每一位取出来， 然后在七段数码管上输出就可以了， 代码隐藏部分是七段数码管的简单输出.

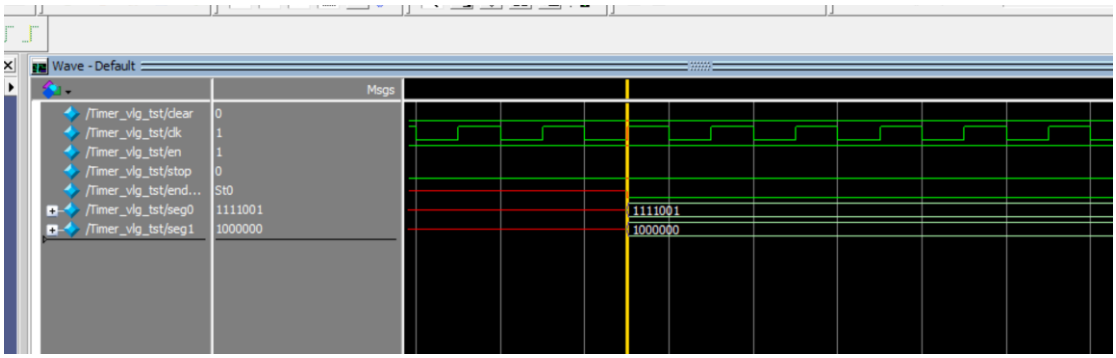
激励代码：

```
// assign statements (if any)
Timer i1 (
// port map - connection between master ports and signals/registers
    .clear(clear),
    .clk(clk),
    .en(en),
    .endone(endone),
    .seg0(seg0),
    .seg1(seg1),
    .stop(stop)
);
initial
begin
    en = 1; clk = 0; clear = 0; stop = 0; #20;
    en = 1;          clear = 0; stop = 1; #20;
    en = 1;          clear = 1; stop = 0; #20;
    en = 0;          clear = 0; stop = 0; #20;
end
always
begin
    #1 clk = ~clk;
end
endmodule
```

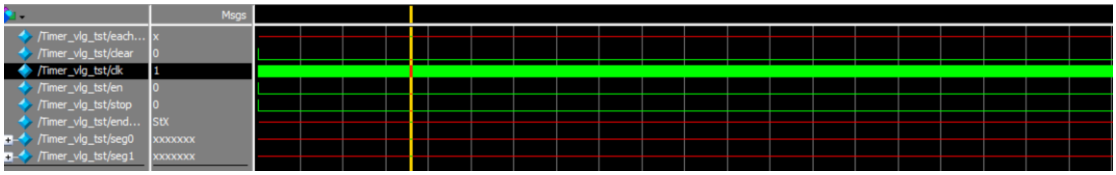
因为我为了避免激励代码出现 Bug，所以都是每次测一个 if 之后就删改了，这是后来还原出来的.

ModelSim 仿真波形：





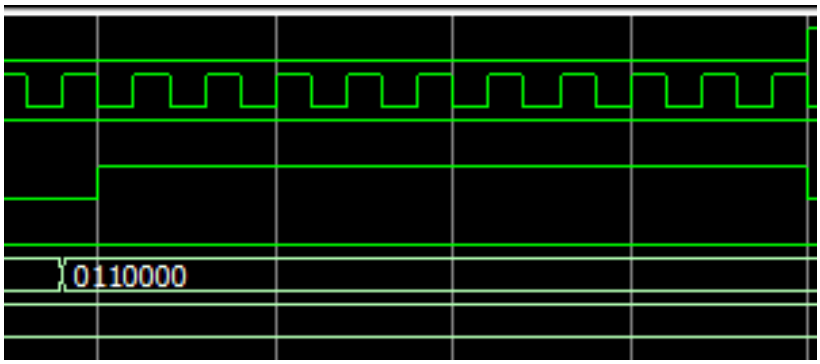
这是第一次测的，显然太密了不好调：



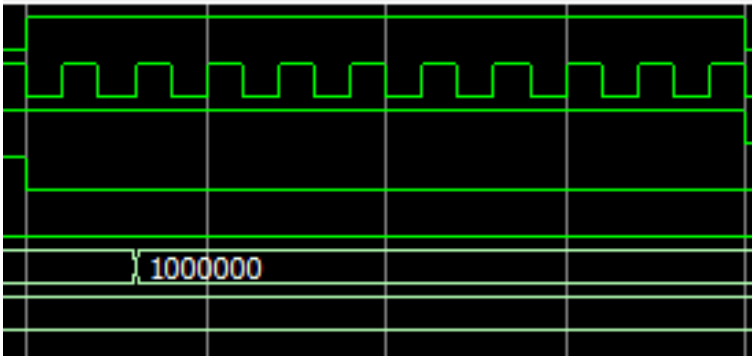
很容易迷失在刚开始还没赋值的那个 `if` 里面.

所以这里有一个小技巧，先不用 25,000,000，改小一点，写入开发板的时候再改大.

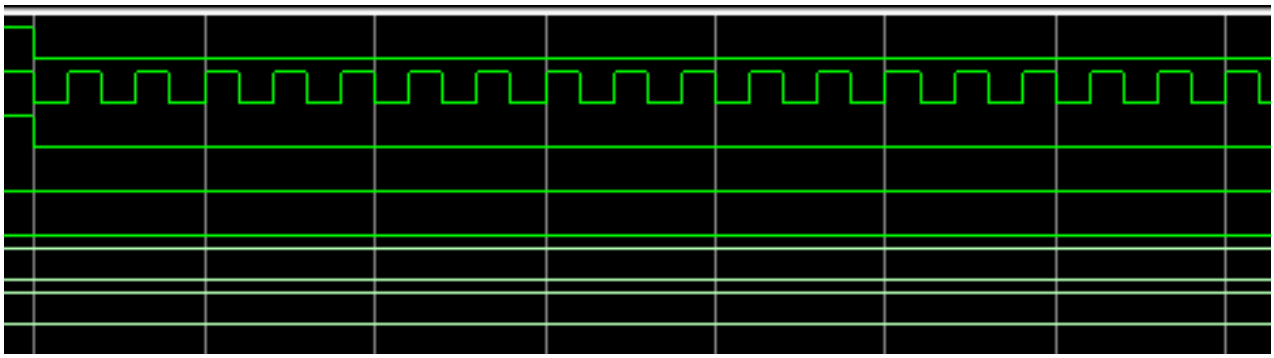
下面具体看一下：



这一小段是 stop 的状态，在时钟有效时，计数器依然没有计数，所以计数器是停下来了。



这一段是清零端有效时的状态，当清零端有效时，计数器被清零，所以对应 7 段显示管的 seg0 和 seg1 都编码为 1000000 了。



这是使能端无效时的状态，这个时候 seg1 和 seg0 的编码都是 0 的编码，就一直不会再变化了。

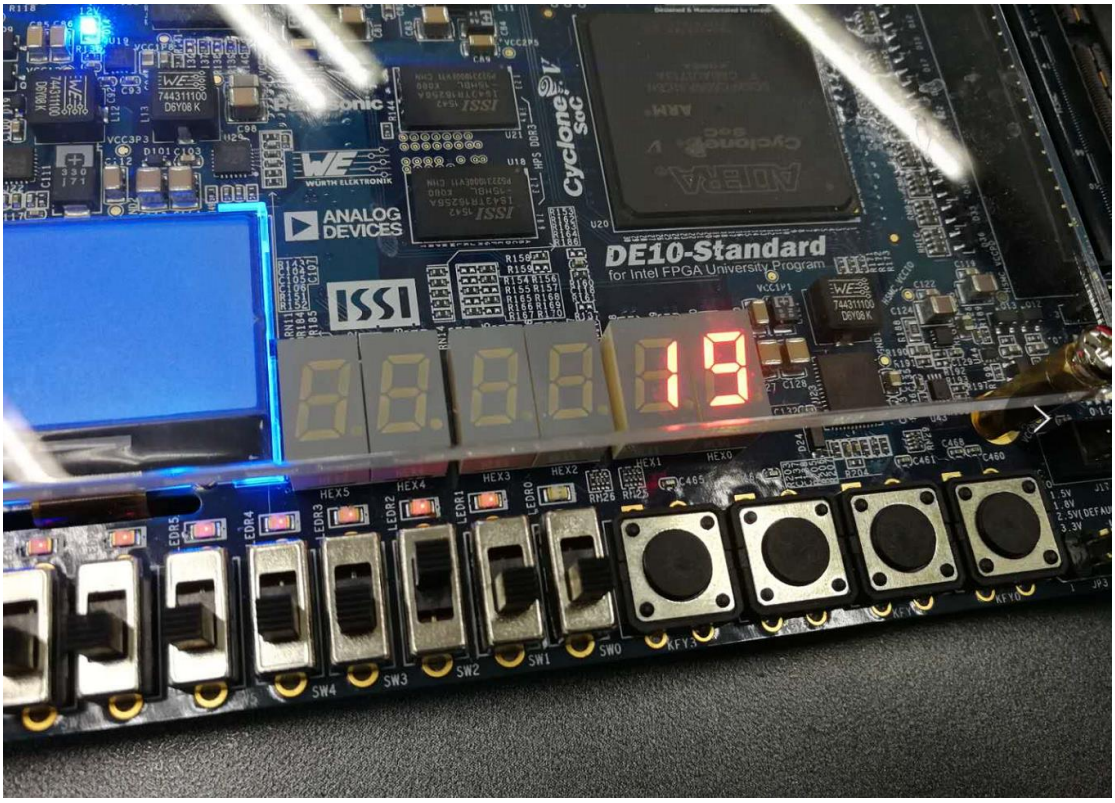
➢ 硬件实现：

引脚分配：

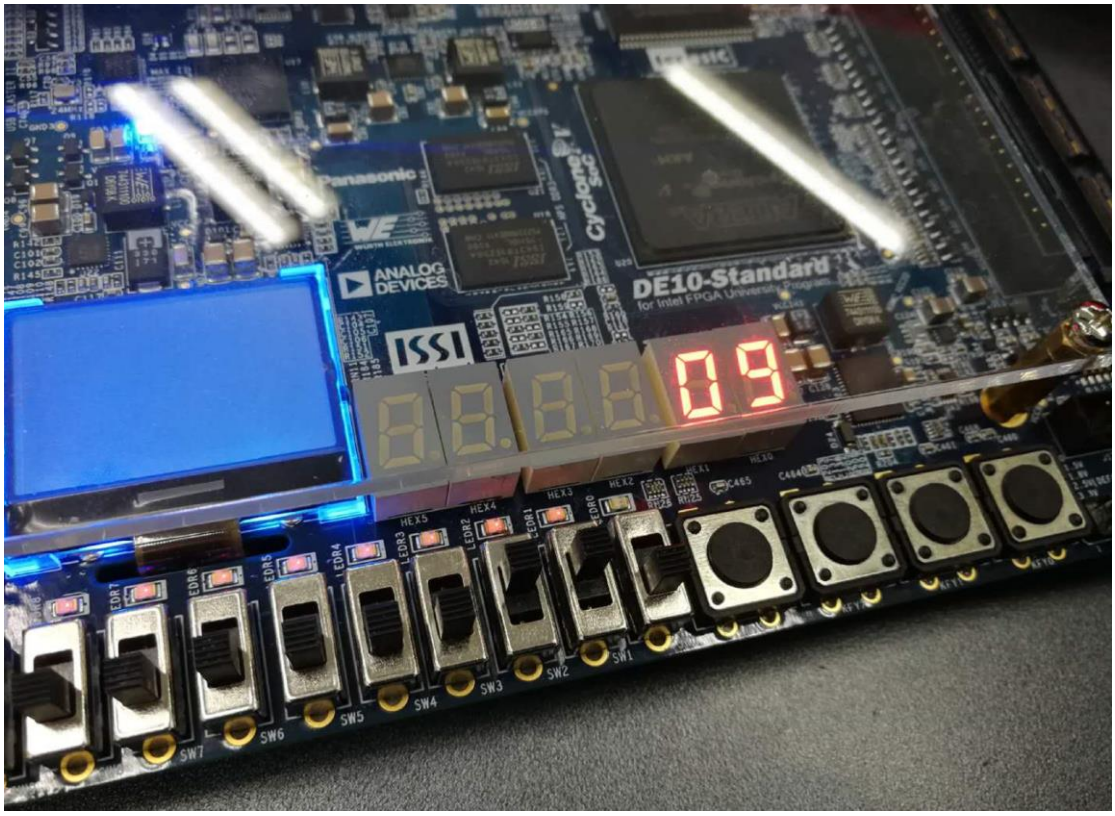
Named: * Edit:

Node Name	Direction	Location	O Ban	EF Grc	Stand	eserve	nt Str	ew
clear	Input	PIN_AB30	5B	B...0	2.5...lt)		12...t)	
clk	Input	PIN_AF14	3B	B...0	2.5...lt)		12...t)	
en	Input	PIN_AB28	5B	B...0	2.5...lt)		12...t)	
endone	Output	PIN_AA24	5A	B...0	2.5...lt)		12...t)	1 (
seg0[6]	Output	PIN_AH18	4A	B...0	2.5...lt)		12...t)	1 (
seg0[5]	Output	PIN_AG18	4A	B...0	2.5...lt)		12...t)	1 (
seg0[4]	Output	PIN_AH17	4A	B...0	2.5...lt)		12...t)	1 (
seg0[3]	Output	PIN_AG16	4A	B...0	2.5...lt)		12...t)	1 (
seg0[2]	Output	PIN_AG17	4A	B...0	2.5...lt)		12...t)	1 (
seg0[1]	Output	PIN_V18	4A	B...0	2.5...lt)		12...t)	1 (
seg0[0]	Output	PIN_W17	4A	B...0	2.5...lt)		12...t)	1 (
seg1[6]	Output	PIN_V17	4A	B...0	2.5...lt)		12...t)	1 (
seg1[5]	Output	PIN_AE17	4A	B...0	2.5...lt)		12...t)	1 (
seg1[4]	Output	PIN_AE18	4A	B...0	2.5...lt)		12...t)	1 (
seg1[3]	Output	PIN_AD17	4A	B...0	2.5...lt)		12...t)	1 (
seg1[2]	Output	PIN_AE16	4A	B...0	2.5...lt)		12...t)	1 (
seg1[1]	Output	PIN_V16	4A	B...0	2.5...lt)		12...t)	1 (
seg1[0]	Output	PIN_AF16	4A	B...0	2.5...lt)		12...t)	1 (
stop	Input	PIN_Y27	5B	B...0	2.5...lt)		12...t)	
<<new node>>								

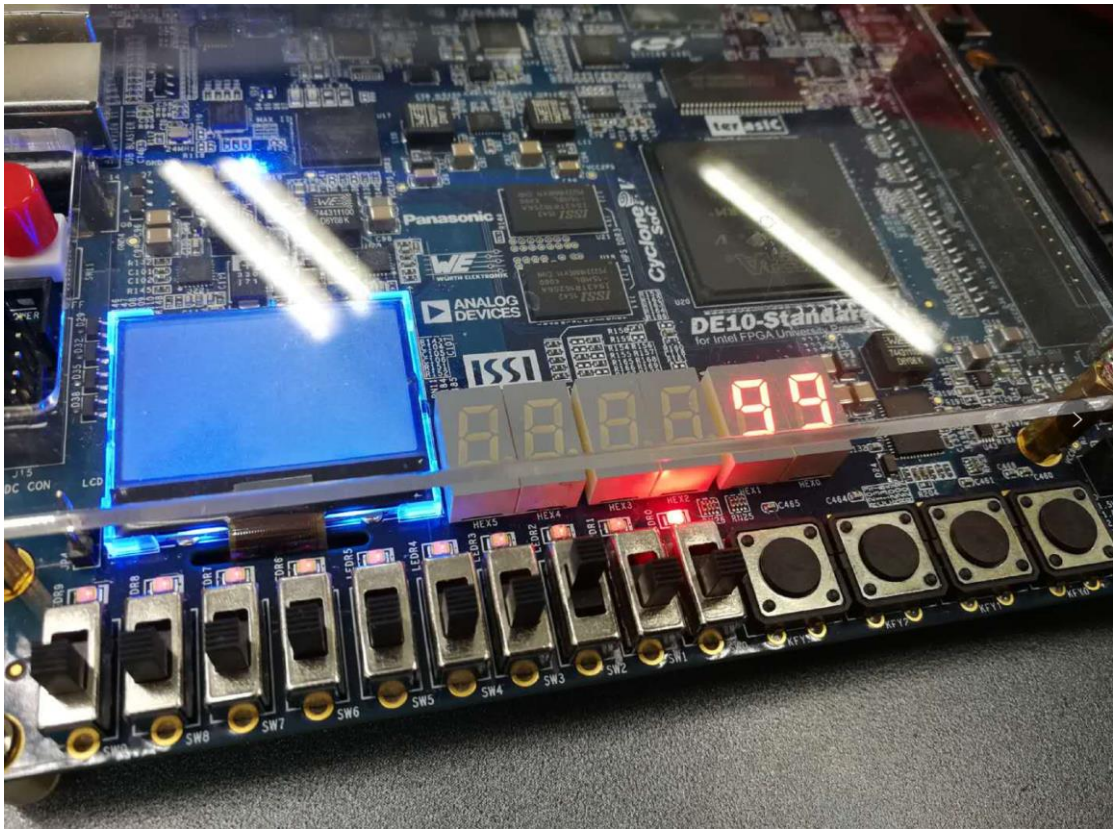
➢ 开发板实现


DE10-Standard
for Intel FPGA University Program
ANALOG DEVICES
ISSI
Panasonic

计时中...


DE10-Standard
ANALOG DEVICES
ISSI
Panasonic

计时完成 LED 灯闪烁


Panasonic
ANALOG DEVICES
ISSI
terasic
DE10-Standard
for Intel FPGA University Program
99
SW1
SW2
SW3
SW4
SW5
SW6
SW7
SW8

四． 实验中遇到的问题及解决方案（请具体的描述问题和解决方法）

a) 连接模块端口的中间变量使用了 reg 类型，使编译出错，解决方案：改用 wire 类型，而不应该是 reg 类型。

b) 第一次使用模块实例化，出现了一些技术性的 bug，解决方案：查找相关资料，一步一步解决问题。

c) **always @ (posedge clk or negedge clr_n)**报错，解决方案：在 always 里面首先 **if(!clr_n),**如此，便不会报这个错误了。

五． 实验得到的启示（积极思考）

在代码量有一定的增大的时候，写代码之前一定要先进行提前规划，将整体框架先想好了，再开始写代码，这样效率会高很多，最关键的是不容易出错。

同时在出现小错误的情况下，要慢慢查找，不能急，这样反而能很快的定位出错位置。注意测试代码也会有 bug…

面对自己陌生的知识以及实验过程，不要急于动手操作，着急只会使实验变得更加复杂，应该好好查阅资料，了解大概的框架之后，再动手去实验。

六． 意见和建议等

这次都挺好的，实现的东西也很有意思！哈哈感觉不错．