

SHANGHAI JIAO TONG UNIVERSITY

实验报告

EXPERIMENT REPORT



基于 Verilog 和 FPGA/CPLD 的多功能秒表设计

学生姓名: <u>XXX</u>

学生学号: ____XXX_____

班 级: ____XXX

目录

<u> </u>	1
实验仪器	
实验任务	
实验过程	
Verilog 部分	
Quartus 部分	
实验总结	_

实验目的

- 1. 初步掌握利用 Verilog 硬件描述语言进行逻辑功能设计的原理和方法。
- 2. 理解和掌握运用大规模可编程逻辑器件进行逻辑设计的原理和方法。
- 3. 理解硬件实现方法中的并行性, 联系软件实现方法中的并发性。
- 4. 理解硬件和软件是相辅相成、并在设计和应用方法上的优势互补的特点。
- 5. 本实验学习积累的 Verilog 硬件描述语言和对 FPGA/CPLD 的编程操作,是进行后续《计算机组成原理》部分课程实验,设计实现计算机逻辑的基础。

实验仪器

- 1. DE1-SOC Cyclone V 实验板 1 套
- 2. Altera Quartus 13.1 软件

实验任务

- 1. 运用 Verilog 硬件描述语言,基于 DE1-SOC 实验板,设计实现一个具有较多功能的计时秒表。
- 2. 要求将 6 个数码管设计为具有"分: 秒: 毫秒"显示, 按键的控制动作有: "计时复位"、"计数/暂停"、"显示暂停/显示继续"等。功能能够满足马拉松或长跑运动员的计时需要。
- 3. 利用示波器观察按键的抖动,设计按键电路的消抖方法。
- 4. 在实验报告中详细报告自己的设计过程、步骤及 Verilog 代码。

实验过程

Verilog 部分

首先,根据实验任务的要求,并且参考图 1 中的秒表逻辑设计,我需要在已经给出的部分 Verilog 代码的基础上完成硬件逻辑设计。

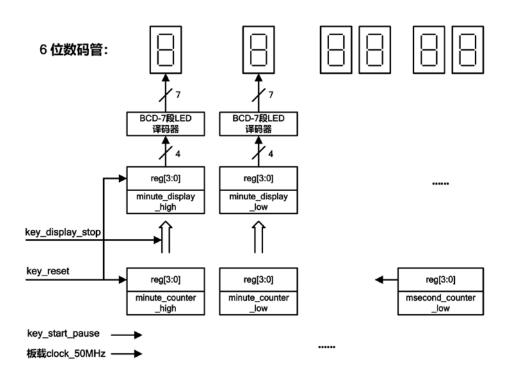


图 1: 实验中秒表参考逻辑设计

在每个时钟信号的上升沿到来时,触发秒表的控制逻辑:最基本的,如果秒表处于运行状态(即 start 置为 1),那么每 500000 个时钟周期(时钟频率是 50MHz,周期为 20ns)后,秒表时间递增 0.01s,同时还需处理好时间单位间的进位。这一部分是秒表的基础部分,大致的 Verilog 代码如下:(其中的 led3 置为 1 表示计时溢出,超出了 60 分钟)

```
if(start == 1) begin
  counter_50M = counter_50M + 1;
  if(counter_50M == CONST_50M) begin
    counter_50M = 0;
    msecond_counter_low = msecond_counter_low + 1;
    if(msecond_counter_low == 10) begin
      msecond_counter_low = 0;
      msecond_counter_high = msecond_counter_high + 1;
      if(msecond_counter_high == 10) begin
        msecond_counter_high = 0;
        second_counter_low = second_counter_low + 1;
        if(second_counter_low == 10) begin
          second_counter_low = 0;
          second_counter_high = second_counter_high + 1;
          if(second_counter_high == 6) begin
             second_counter_high = 0;
             minute_counter_low = minute_counter_low + 1;
             if(minute_counter_low == 10) begin
```

在对秒表时间的计数器递增后,如果秒表此时处于正常显示状态 (即 display 置为 1),则还需要让秒表显示与计数器保持同步: (此处的 counter_reset == DELAY_TIME 用于复位按键的消抖)

```
if(display == 1 || counter_reset == DELAY_TIME) begin
  msecond_display_low = msecond_counter_low;
  msecond_display_high = msecond_counter_high;
  second_display_low = second_counter_low;
  second_display_high = second_counter_high;
  minute_display_low = minute_counter_low;
  minute_display_high = minute_counter_high;
end
```

而为了实现按键对于秒表运行状态的控制,以及按键的消抖,我采取的方式是: 当一个时钟上升沿到来时,如果按键被按下(对应的输入变量为 0),则对应的计数器加一,否则计数器清零,若计数器达到了一定的值,即按键被连续一定时间按下,则认为用户真正按下了这个按键并改变秒表的运行状态。对于 start/pause 按键,所做的操作是将秒表时间计数器是否暂停累加;对于 display/stop 按键,所做的操作是设置秒表显示是否与计数器保持同步;对于 reset 按键,则是将秒表的计数器和显示都归零。这一部分的 Verilog 代码如下:

```
if(key_reset == 0)
    counter_reset = counter_reset + 1;
else
    counter_reset = 0;
if(key_start_pause == 0)
    counter_start = counter_start + 1;
else
    counter_start = 0;
if(key_display_stop == 0)
    counter_display = counter_display + 1;
else
    counter_display = 0;
```

```
if(counter_start == DELAY_TIME)
      start = ~start;
    if(counter_display == DELAY_TIME)
      display = \sim display;
    if(counter_reset == DELAY_TIME) begin
      counter_50M = 0;
      msecond counter low = 0;
      msecond_counter_high = 0;
      second_counter_low = 0;
      second_counter_high = 0;
      minute_counter_low = 0;
      minute_counter_high = 0;
    end
    为了方便判断秒表的运行状态, 我设置了 3 个 LED 指示灯的输出, 分别用于指示秒表
是否在计时, 是否同步显示, 以及是否计时溢出(此 LED 指示灯见秒表计数器的基础部分)。
   led0 = start;
   led1 = display;
    额外地,为了确保秒表运行状态正确,对用到的变量进行了初始化:
    initial begin
      start = 0;
      display = 1;
      led0 = start;
      led1 = display;
      led3 = 0;
      counter_reset = 0;
      counter_start = 0;
      counter_display = 0;
      counter_50M = 0;
      msecond_counter_low = 0;
      msecond_counter_high = 0;
      second_counter_low = 0;
      second_counter_high = 0;
      minute_counter_low = 0;
      minute_counter_high = 0;
      msecond_display_low = 0;
      msecond_display_high = 0;
      second_display_low = 0;
      second_display_high = 0;
      minute_display_low = 0;
      minute_display_high = 0;
    end
```

Quartus 部分

在利用 Verilog 完成了硬件控制逻辑的描述之后, 还需要在 Quartus 中进行布线、编译、烧录等步骤。

首先,从 Verilog 文件创建 Symbol 文件,得到了 stop_watch 的 symbol,将其加入到 top-level design 的 bdf 文件中,并连接合适的输入输出组件。如图 2 所示:

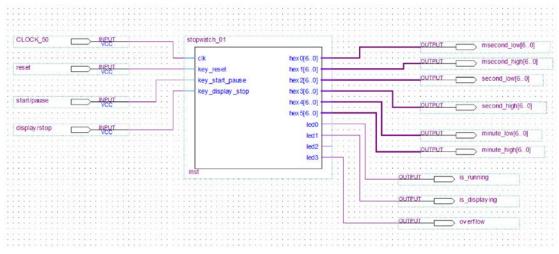


图 2 BDF 文件

在 Analysis & Elaboration 通过后,为输入输出设置合适的 DE1-SOC 开发板上引脚,如图 3 所示:

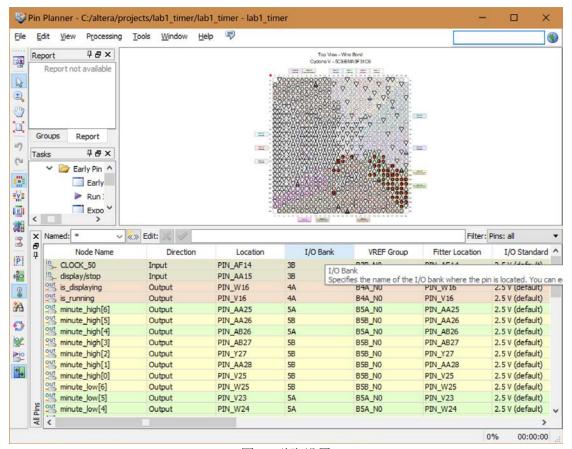


图 3 引脚设置

然后进行 Compilation,通过之后得到 sof 文件,即可用 Programmer 写到开发板上,如图 4 所示:

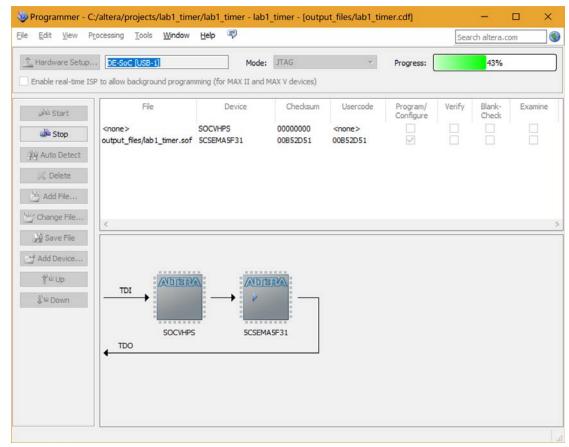
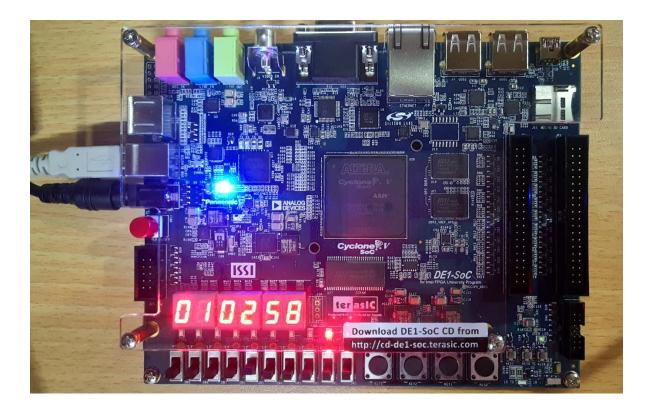


图 4 Programmer 页面

最终, 可以在开发板上观察实际运行效果, 如图 5 所示:



实验总结

本次实验是计算机组成课程中的第一次实验,实现了一个功能较为完备的秒表。通过本次实验,我学习了 Verilog 语言的基本用法以及使用 DE1-SOC 开发板和 Quartus 软件进行 FPGA 设计的基本流程。

关于实验内容,比如按键消抖,这让我了解到了一些以往的学习中并未设计的工程实践内容,更加深入地理解了硬件设计的复杂性与工程性,收益颇丰。而利用 FPGA 开发板进行设计,更是加深了我对计算机硬件工作原理的认识。