电 子 科 技 大 学

UNIVERSITY OF ELECTRONIC SCIENCE AND TECHNOLOGY OF CHINA

学 士 学 位 论 文

BACHELOR DISSERTATION

论文题目 数字式秒表设计

学生姓名 李 宁

学 号 2016050201017

学 院 光电科学与工程学院

专 业 光电科学与工程专业

指导教师 刘 曦

指导单位 光电科学与工程学院

2019 年 4 月 9 日

摘 要

本次实验，是基于 FPGA 的秒表设计。秒表是 00:00:00 计数到 59:59:99,使用 VHDL 语言编写秒表的程序，并设计有开始/暂停以及保持/恢复的功能。本试验利用 ISE 进行设计输入和综合，在 modelsim 软件上实现了波形的仿真，将程序下载到芯片 Spartan-3A and 3AN 上，并在七段数码管上实现了秒表的显示，在 FPGA 的按键上实现对秒表的控制功能。

关键字：FPGA，VHDL，ISM，Modelsim，秒表

ABSTRACT

This experiment is a stopwatch design based on FPGA.Stopwatch is counting to 59:59:99, using VHDL language stopwatch program, and design has the function of start/stop and hold/restore.In this experiment, ISE is used for design input and synthesis, waveform simulation is realized on modelsim software, program is downloaded to chip spartan-3a and 3AN, stopwatch display is realized on seven-segment digital tube, and stopwatch control function is realized on FPGA key.

key words: FPGA,VHDL,ISM, ModelsimSecond,chronograph

目录

第1章 引言

本文主要是针对实验任务的要求，基于 XILINX FPGA 芯片 Spartan-3A，进行秒表的制作实验。

实现过程包括 VHDL 文件代码的编写，ISE 综合、Modelsim 仿真、优化、实验板实现等，尤其是在工程当中的分模块构建的思想，将一个大工程分模块化，不仅可以使思路明确，还可以提高调试错误的效率，每一个模块有良好的扩展性，有利于今后的实验学习。

第2章 实验原理

2.1 时间的概念及计时方法

时间是物理学中的七个基本量纲之一，符号 t.在国际单位制(SI)中，时间的额吉本单位是秒，符号 s,在 1967 年召开的第 13 届国际度量衡大会对秒的定义是:鉋 133 原子基态的两个超精细能阶间跳跃对应辐射的 9, 192, 631, 770 个周期的持续时间。这个定义提到的铯原子必须在绝对零度是静止的，而且在地面上的环境是零磁场。在这样的情况下被定义的秒，在天文学上的历书时所定义的秒是等效的。中国古代的计时器有太阳钟和机械钟两种。太阳钟是以太阳的投影和方位来计时的。代表有土圭、圭表、日晷等。而由于地球轨道偏心率以及地球倾角的影响，真太阳时和平太阳时是不一致的，机械钟应运而生，代表有水钟、沙漏等.

时间已经走到 21 世纪，为了更加精确的测量时间，现代人使用了很多方法。最常用的包括有机械时钟和电子时钟。本文介绍的电子秒表便是电子时钟的一种。我们将基于 FPGA 和 EDA 软件来制作-一款秒表，已达到计时的目的。

2.2 VHDL 介绍

2.2.1 VHDL 简介

VHDL 的英文全名是 Very High Speed Integrated Circuit Hardware Description Language,诞生于 1982 年。1987 年底，VHDL 被 IEEE 和美国国防部确认为标准硬件描述语言。

VHDL 主要用于描述数字系统的结构，行为，功能和接口。除了含有许多具有硬件特征的语句外，VHDL 的语言形式和描述风格与句法是十分类似于一般的计算机高级语言。VHDL 的程序结构特点是将项工程设计， 或称设计实体(可以是一个元件，一个电路模块或 一个系统) 分成外部(或称可视部分，及端口)和内部(或称不可视部分)，既涉及实体的内部功能和算法完成部分。在对一个设计实体定 义了外部界面后，-旦其内部开发完成后，其他的设计就可以直接调用这个实体.这种将设计实体分成内外部分的概念是 VHDL 系统设计的基本点。

2.2.2 VHDL 特点

VHDL 语言能够成为标准化的硬件描述语言并获得广泛应用，它自身必然具有很多其他硬件描述语言所不具备的优点。归纳起来，VHDL 语言

主要具有以下优点：

2.2.2.1 VHDL 语言功能强大，设计方式多样

VHDL 语言具有强大的语言结构，只需采用简单明确的 VHDL 语言程序就可以描述十分复杂的硬件电路。同时，它还具有多层次的电路设计描述功能。此外，VHDL 语言能够同时支持同步电路、异步电路和随机电路的设计实现，这是其他硬件描述语言所不能比拟的。VHDL 语言设计方法灵活多样，既支持自顶向下的设计方式，也支持自底向上的设计方法:既支持模块化设计方法，也支持层次化设计方法。

2.2.2.2 VHDL 语言具有强大的硬件描述能力

VHDL 语言具有多层次的电路设计描述功能，既可描述系统级电路，也可以描述门级电路;描述方式既可以采用行为描述、寄存器传输描述或者结构描述，也可以采用三者的混合描述方式。同时，VHDL 语言也支持惯性延迟和传输延迟，这样可以准确地建立硬件电路的模型。VHDL 语言的强大描述能力还体现在它具有丰富的数据类型。VHDL 语言既支持标准定义的数据类型，也支持用户定义的数据类型，这样便会给硬件描述带来较大的自由度。

2.2.2.3 VHDL 语言具有很强的移植能力

VHDL 语言很强的移植能力主要体现在:对于同一个硬件电路的 VHDL 语言描述，它可以从一个模拟器移植到另个模拟器上、 从一个综合器移植到另一个综合器上或者从一个工作平台移植到另一个工作平台上去执行。

2.2.2.4 VHDL 语言的设计描述与器件无关

采用 VHDL 语言描述硬件电路时，设计人员并不需要首先考虑选择进行设计的器件。这样做的好处是可以使设计人员集中精力进行电路设计的优化，而不需要考虑其他的问题。当硬件电路的设计 描述完成以后，VHDL 语言允许采用多种不同的器件结构来实现。

2.2.2.5 VHDL 语言程序易于共享和复用

VHDL 语言采用基于库(ibrary)的设计方法。在设计过程中，设计人员可以建立各种可再次利用的模块，- 一个大规模的硬件电路的设计不可能从门级电路开始步步地进行设计， 而是一些模块的累加。这些模块可以预先设计或者使用以前设计中的存档模块，将这些模块存放在库中，就可以在以后的设计中进行复用。由于 VHDL 语言是一一种描述、模拟、综合、优化和布线的标准硬件描述语言，因此它可以使设计成果在设计人员之间方便地进行交流和共享，从而减小硬件电路设计的工作量，缩短开发周期。

2.3 FPGA 介绍

2.3.1 1FPGA 简介

FPGA (fdereoraabalatetraty) ,即现场可编程门阵列，它是在 PAL、GAL、CPLD 等可编程器件的基础上进一步发展的产物。它是作为专用集成电

路(ASIC)领域中的一一种半定制电路而出现的， 既解决了定制电路的不足，又克服了原有可编程器件门电路数有限的缺点。以硬件描述语言(eio 或 VHDU)所完成的电路设计，可以经过简单的综合与布局，快速的烧录至 FPGA 上进行测试，是现代 C 设计验证的技术主流。这些可编辑元件可以被用来实现一些基本的逻辑门电路(比如 AND、OR. XOR. Nor)或者更复杂些的组合功能比如解码器或数学方程式。在大多数的 FPGA 里面，这些可编辑的元件里也包含记忆元件例如触发器(Flip- flop)或者其他更加完整的记忆块。系统设计师可以根据需要通过可编辑的连接把 FPGA 内部的逻辑块连接起来，就好像-个电路试验板被放在了-个芯片里。一个出厂 后的成品 FPGA 的逻辑块和连接可以按照设计者而改变，所以 FPGA 可以完成所需要的逻辑功能。FPGA 一般来说比 ASIC (专用集成电路)的速度要慢，实现同样的功能比 ASIC 电路面积要大。但是他们也有很多的优点比如可以快速成品，可以被修改来改正程序中的错误和更便宜的造价。厂商也可能会提供便宜的但是编辑能力差的 FPGA。因为这些芯片有比较差的可编辑能力，所以这些设计的开发是在普通的 FPGA 上完成的，然后将设计转移到一个类似于 ASIC 的芯片上。另外-一种方法是用 CPLD (Complex Programmable Logic Device, 复杂可编程逻辑器件)。

2.3.2 FPGA 原理

FPGA 采用了逻辑单元阵列 LCA (Logicoll Array)这样一个概念，内部包括可配置逻辑模块 CLB (Configurable Logic Block) 、输入输出模块 IOB(Input Output Block)和内部连线(Interconnect) 三个部分。现场可编程门阵列(FPGA)是可编程器件，与传统逻辑电路和门阵列(如 PAL, GAL 及 CPLD 器件)相比，FPGA 具有不同的结构。FPGA 利用小型查找表(16X 1RAM)来实现组合逻辑，每个查找表连接到一个 D 触发器的输入端，触发器再来驱动其他逻辑电路或驱动 1/O,由此构成了既可实现组合逻辑功能又可实现时序逻辑功能的基本逻辑单元模块，这些模块间利用金属连线互相连接或连接到/0 模块。FPGA 的逻辑是通过向内部静态存储单元加载编程数据来实现的，存储在存储器单元中的值决定了逻辑单元的逻辑功能以及各模块之间或模块与 1/O 间的联接方式，并最终决定了 FPGA 所能实现的功能，FPGA 允许无限次的编程。

2.3.3 FPGA 特点

2.3.3.1 **采用 FPGA 设计 ASIC 电路(专用集成电路)，用户不需要投片生产，就能得到合用的芯片.**

2.3.3.2 **FPGA 可做其它全定制或半定制 ASIC 电路的中试样片。**

2.3.3.3 **FPGA 内部有丰富的触发器和 1/0 引脚。**

2.3.3.4 **FPGA 是 ASIC 电路中设计周期最短、开发费用最低、风险最小的器件之-”。**

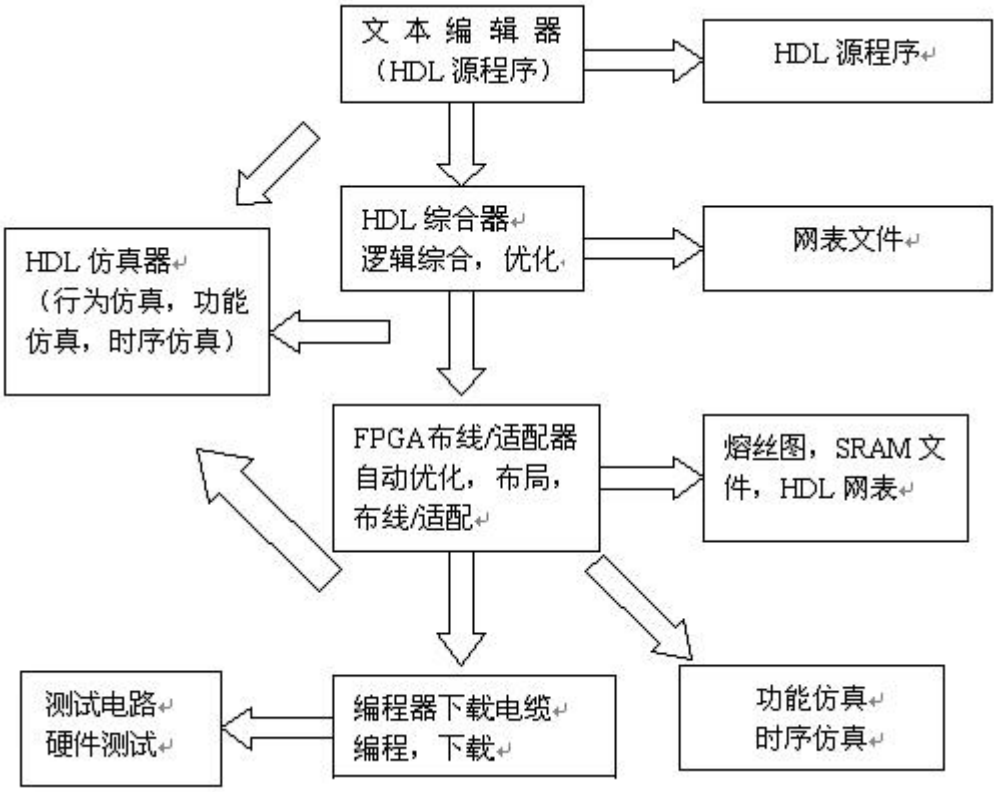
2.3.3.5 **FPGA 采用高速 CMOS 工艺功耗低，可以与 CMOS、TlL 电平兼容。**

可以说，FPGA 芯片是小批量系统提高系统集成度、可靠性的最佳选择之一。

FPGA 是由存放在片内 RAM 中的程序来设置其工作状态的，因此，工作时需要对片内的 RAM 进行编程。用户可以根据不同的配置模式，采用不同的编程方式。

加电时，FPGA 芯片将 EPROM 中数据读入片内编程 RAM 中，配置完成后，FPGA 进入工作状态。掉电后，FPGA 恢复成白片，内部逻辑关系消失，因此，FPGA 能够反复使用。FPGA 的编程无须专用的 FPGA 编程器，只须用通用的 EPROM、PROM 编程器即可。当需要修改 FPGA 功能时，只需换--片 EPROM 即可。这样，同一片 FPGA,不同的编程数据，可以产生不同的电路功能。因此，FPGA 的使用非常灵活。

2.4 基于 FPGA 的 VHDL 设计流程



2.4.1文本编辑

用任何文本编辑器都可以进行，通常 VHDL 文件保存为 vhd 文件，Verilog 文件保存为 v 文件。

2.4.2 使用编译工具编译源文件

HDL 的编译器有很多，ACTIVE 公司，MODELSIM 公司，SYNPLICITY 公司，SYNOPSYS 公司，VERIBEST 公司等都有自己的编译器。

2.4.3 逻辑综合

将源文件调入逻辑综合软件进行综合。综合的目的是在于将设计的源文件由语言转换为实际的电路。但是此时还没有在芯片中形成真正的电路。这一步的最终目的是生成门电路级的网表(Netlist)。

2.4.4 布局布线

将第 3 步生成的网表文件调入 PLD 厂家提供的软件中进行布线，即把设计好的逻辑安放到 CPLD / FPGA 内。这一步的目的是生成用于下载(编程 Programming)的编程文件。在这一步，将用到第 3 步生成的网表，并根据

CPLD／FPGA 厂商的器件容量，结构等进行布局、布线。这就好像在设计 PCB 时的布局布线一样。先将各个设计中的门根据网表的内容和器件的结构放在器件的特定部位。然后，在根据网表中提供的各门的连接，把各个门的输入输出连接起来。最后，生成一个供编程的文件。这一步同时还会加一些时序信息(Timing)到你的设计项目中去，以便于你做后仿真。

2.4.5 仿真

利用在布局布线中获得的精确参数，用仿真软件验证电路的时序。(也叫布局布线仿真或时序仿真)。这一步主要是为了确定你的设计在经过布局布线之后，是不是还满足你的设计要求。

2.4.6 编程下载

如果前几步都没有发生错误，并且符合设计要求，这一步就可以将由适配器等产生的配置或下载文件通过编程器或下载电缆下载到目标芯片中。

2.4.7 硬件测试

硬件测试的目的是为了在更真实的环境中检验 HDL 设计的运行情况，特别是对于 HDL 程序设计上不是十分规范，语义上含有一定歧义的程序。

第3章 开发环境

3.1 软件：

3.1.1 ISE 介绍

ISE 的主要功能包括设计输入、综合、仿真、实现和下载，涵盖了可编程逻辑器件开发的全过程，从功能上讲，完成 CPLD/PFCA 的设计流程无需借助任何第三方 EDA 软件。下面简婴说明各功能的作用:设计输入，IsE 提供的设计输入工具包括用于 HDL 代码输入和查看报告的 SE 文本编辑器(The text ISE Editor ,用于原理图编辑的工具 ECS (The Engineering Capture System)，用于生成 IP Core 的 CoreGenerator,用于状态机设计的 State CAD 以及用于约束文件编辑的 Constraint Editor 等。

综合: ISE 的综合工具不但包含了 Xilinx 自身提供的综合工具 XST, 同时还可以内嵌 Mentor Graphics 公司的 Leonardo Spectrum 和 Synplicity 公司的 Syplif,实现无缝链接。

仿真:ISE 本身自带了一个具有图形化波形编辑功能的仿真工具 HDL Bencher,同时又提供了使用 Model Tech 公司的 Modelsim 进行仿真的接口。实现:此功能包括了翻译、映射、布局布线等，还具备时序分析、管脚指定以及增量设计等高级功能。

下载:下载功能包括了 Bit Gen, 用于将布局布线后的设计文件转换为位流文件， 还包括了 IMPACT,功能是进行芯片配置和通信， 控制将程序烧写到 FPGA 芯片中去。

3.1.2 Modelsim 介绍

3.1.2.1 Modelsim 简介

Mentor 公司的 ModelSim 是业界最优秀的 HDL 语言仿真软件，它能提供友好的仿真环境，是业界唯的单 内核支持 VHDL 和 Verilog 混合仿真的仿真器。它采用直接优化的编译技术、Tcl/Tk 技术、和单一内核仿真技术，编译仿真速度快，编译的代码与平台无关，便于保护 IR 核，个性化的图形界面和用户接口，为用户加快调错提供强有力的手段，是 FPGA/ASIC 设计的首选仿真软件。

3.1.2.2 Modelsim 的特点

1）、RTL 和门级优化，本地编译结构，编译仿真速度快，跨平台跨版本仿真；

2）、单内核 VHDL 和 Verilog 混合仿真；

3）、源代码模版和助手，项目管理；

4）、集成了性能分析、波形比较、代码覆盖、数据流 ChaseX、Signal Spy、虚拟对象 Virtual 0bject、Memory 窗口、Assertion 窗口、源码窗口显示信号值、信号条件断点等众多调试功能；

5）、C 和 Tcl/Tk 接口，C 调试；

6）、对 SystemC 的直接支持，和 HDL 任意混合；

7）、支持 SystemVerilog 的设计功能；

8）、对系统级描述语言的最全面支持，System Verilog, System C, PSL；

9）、ASIC Sign off。

10）、可以单独或同时 进行行为(behaviora1) RTL 级、和门级(gate-level)的代码。

3.2 硬件

3.2.1 电子科技大学电子实验中心 FPGA 实验电路板

3. 2. 1. 1 实验电路板原理图

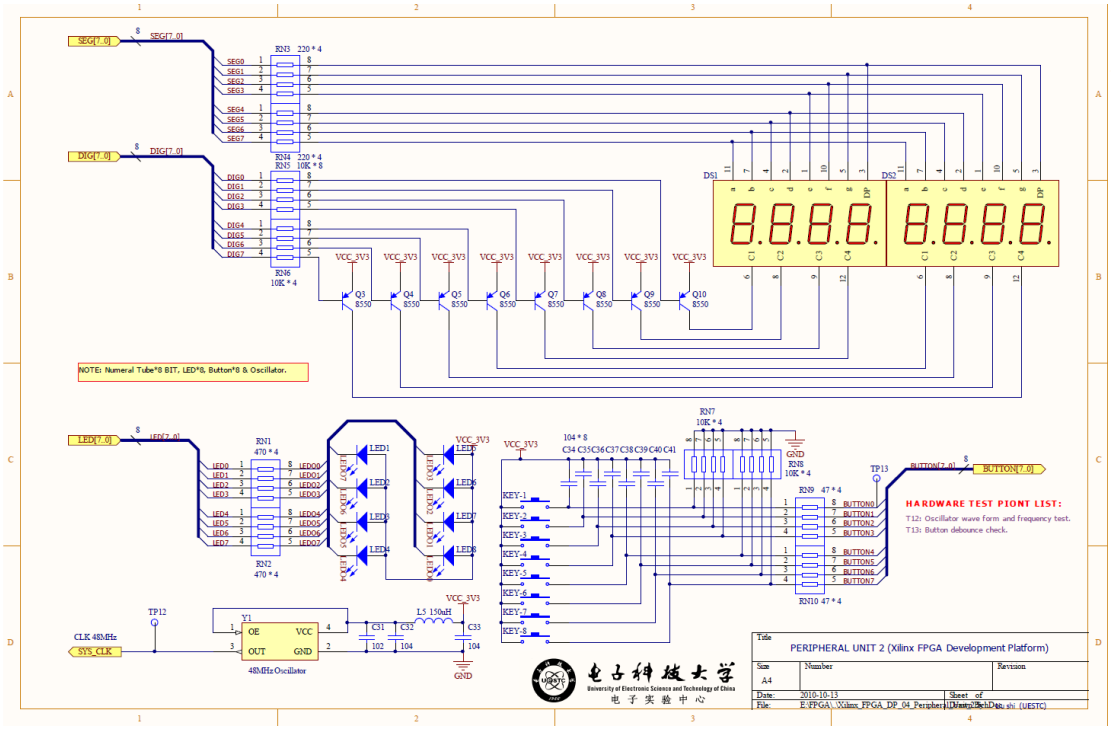


图 1 实验电路板原理图

根据电路板原理图,我们可以看到电路板的时钟为 48Mhz,故如果我

们要使用内部时钟,我们必须要对其分频。

实验电路板的按键输入并没有硬件上的消抖动,故如果我们想要输入一个时钟的高电平是很困难的,所以如果使用按键,就必须有消抖动的考虑。

实验电路板上的八位数码管为共阳数码管，当输入低电平时数码管亮，但是由于数码管共用 abcdef,dp 八位段选，我们要想让其显示不同的字符必须进行位选的考虑，具体实现根据实际情况确定位选显示频率。

第4章 电子秒表的设计与实现

4.1 设计任务

4.1.1 输入输出

4.1.2

1、秒表的计时范围为 00’00’’00~59’59’’99。

2、有两个按钮开关 Start/Stop 和 Split/Reset,控制秒表的启动、停止、分段、复位。

4.1.3 状态解释

1、在秒表已经被复位的情况下，按下“Start/Stop”键秒表正常运行；

2、在秒表正常运行的情况下，如果按下“Start/Stop”键，则秒表暂停计时；

3、再次按下该键，秒表继续计时。

4、在秒表正常运行的情况下，如果按下“Split/Reset,”键，显示停止在按键时的时间，但秒表仍然在计时；

5、再次按下该键，秒表恢复正常显示。

6、在秒表暂停计时的情况下按下“Split/Reset,”键，秒表复位归零。

4.2 实验条件

利用 EDA 软件（ISE、modelsim）和 VHDL 语言在 EEC-FPGA 实验板上完成秒表的设计与实现，实验板原理如图所在项目开始设计时，首先要确定系统的需求并发展出针对这些需求的计划，设计方案以及系统需求必须按照设计要求并根据客观条件来确定。

4.3 系统需求和解决方案计划

按照秒表的设计要求以及具备的客观条件，在 FPGA 上设计的电路包括下面这些组成部分：

4.3.1 分频器

对晶体振荡器产生的时钟信号进行分频，产生时间基准信号。

4.3.2 计数器

对时间基准脉冲进行计数，完成计时功能。

4.3.3 数据锁存器

锁存数据，使显示保持稳定。

4.3.4 控制器

控制计数器的运行、暂停以及复位。

产生锁存器的使能信号。

4.3.5 显示模块

包括扫描计数器、译码器和数据选择器。

控制 8 个数码管以扫描方式显示计时结果。

4.3.6 按键消抖电路

消除按键输入信号抖动的影响，输出单脉冲。

4.4 设计电路图

4.4.1 电路图

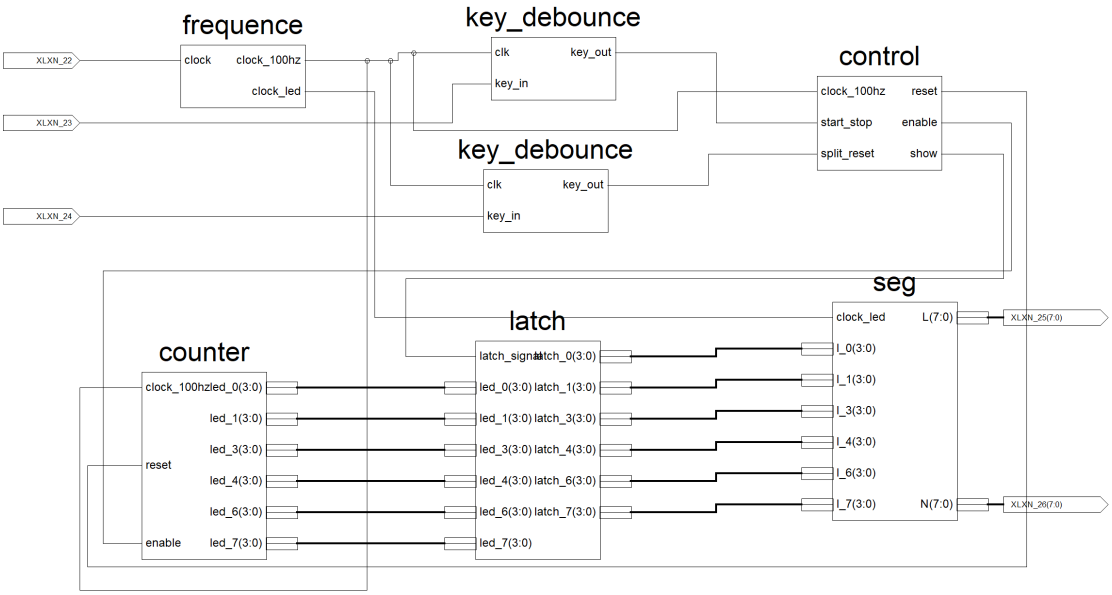


图 2 顶层电路图

4.4.2 模块功能叙述

48Mhz 时钟输入 frequence 模块输出计数时钟 100hz 和显示时钟 1khz，外部输入 start/stop,split/reset 经过消抖电路输入控制电路，由控制电路产生复位、使能、锁存显示的信号，分别控制计数器计数、清零、暂停和锁存器锁存，计数器输出的六位秒表数字输入锁存器然后由锁存器锁存输出，经过译码之后产生片选和段选信号，驱动数码管显示。

4.5 模块设计及仿真结果

4.5.1 分频器

4. 5. 1. 1 介绍

首先对 48Mhz 信号分频，分出 1khz 的片选信号信号，然后在此基础上分出 100hz 的计数信号，节省触发器。

4.5.1.2 仿真结果

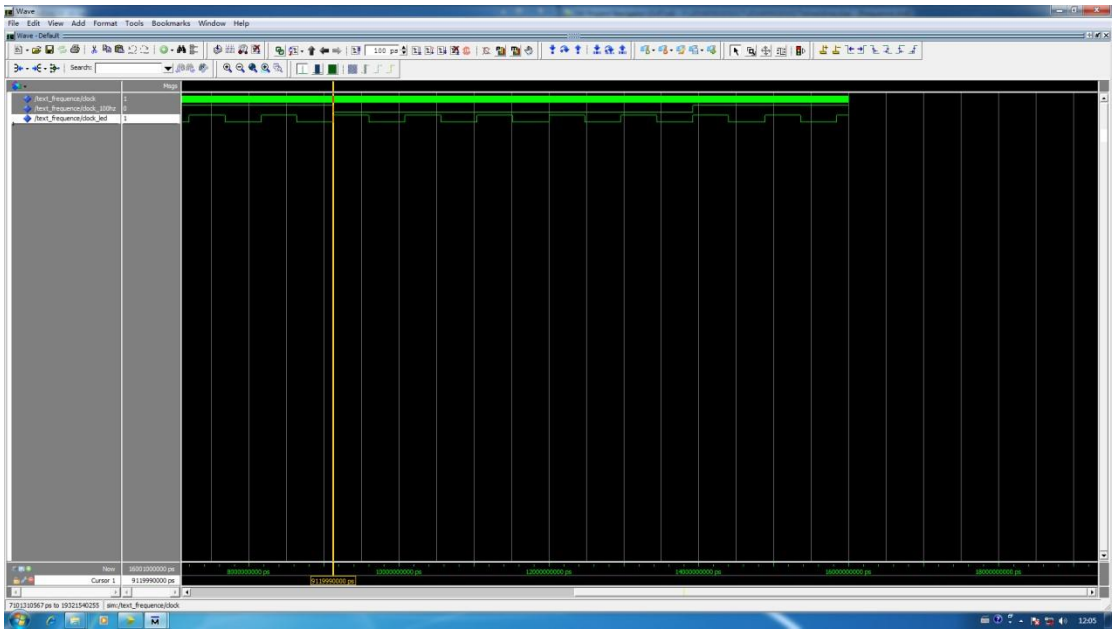


图 3 分频器仿真结果

4.5.2 计数器

4.5.2.1 介绍

根据设计要求，我们可以将计数模块分为十进制计数器 counter_10 和六进制计数器 counter_6，再用顶层计数器模块 counter 按顺序调用即可产生六位计数信号。

4.5.2.2 仿真结果

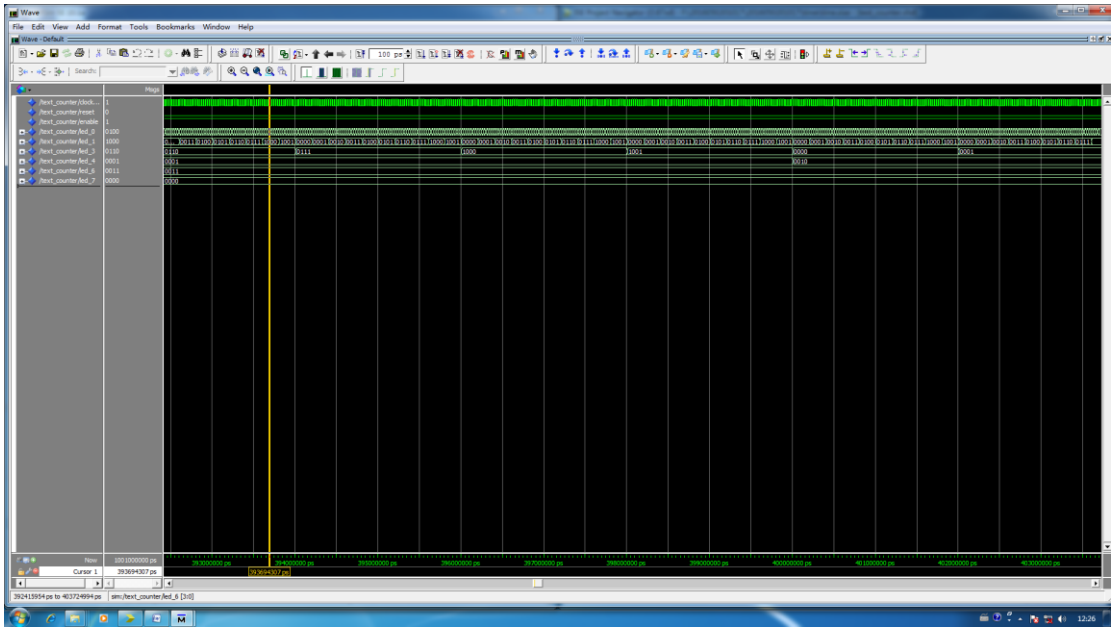


图 4 计数器仿真结果

4.5.3 数据锁存器

4.5.3.1 介绍

根据 ISE 对 VHDL 语言中不完整的 if 语句综合成为锁存器，仅需要用一个 if 语句即可产生锁存。

4.5.3.2 仿真结果

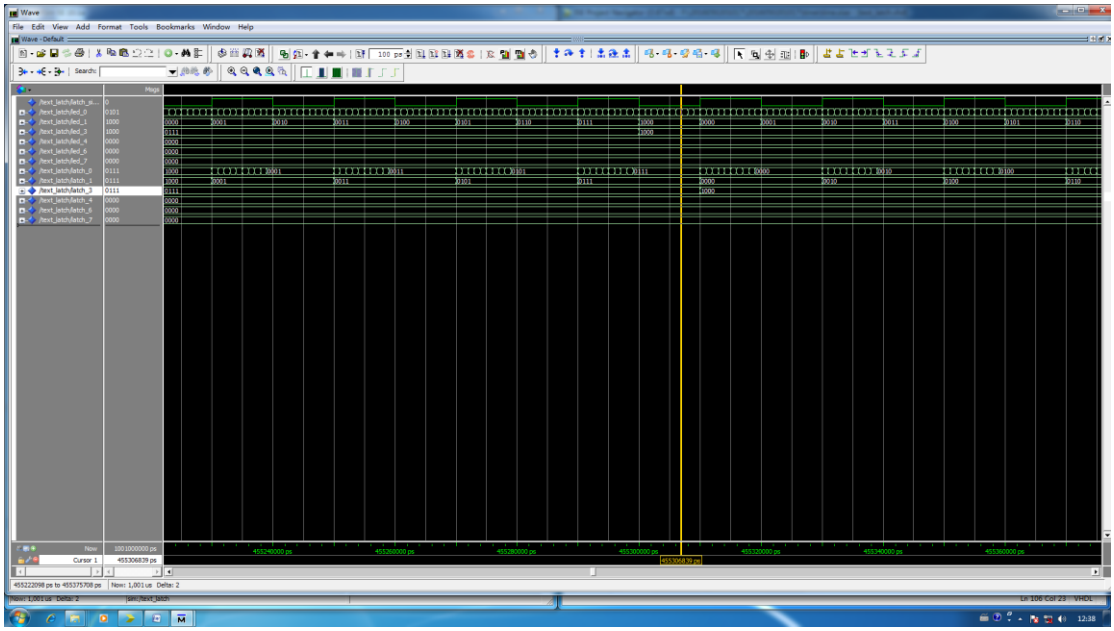


图 5 锁存器仿真结果

4.5.4 控制器

4.5.4.1 介绍

控制器是整个设计的核心，严格按照状态图来设计。

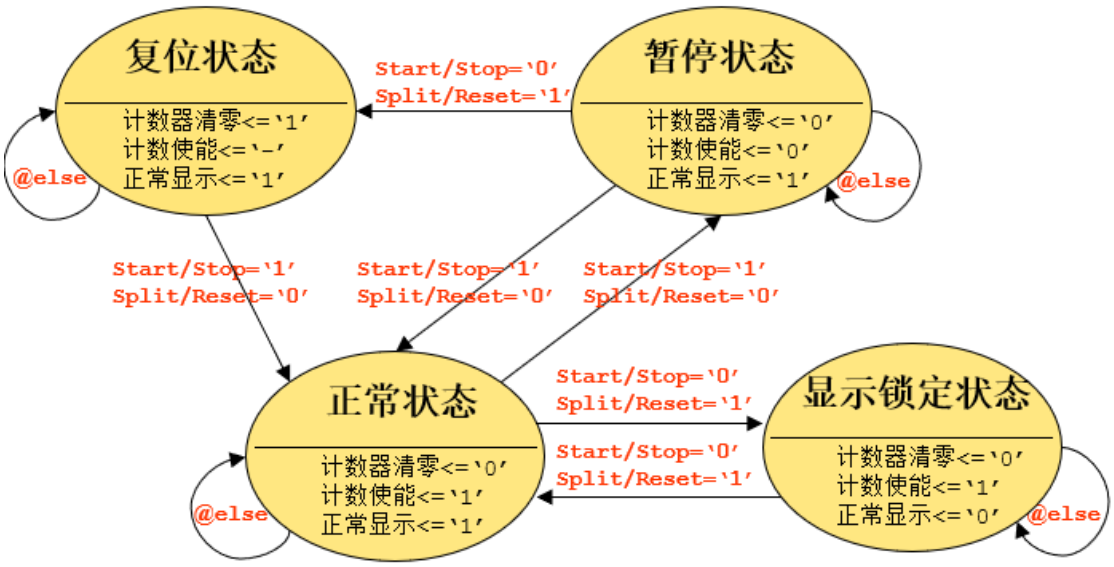


图 6 状态图

根据状态图可画出状态表如下：

A：正常状态；B 暂停状态；C 复位状态；D 显示锁定状态

表格 4-1

	start/stop		split/reset		reset	enable	show
Q	00	01	11	10			
A	A	D	A	B	0	1	1
B	B	C	B	A	0	0	1
C	C	C	C	A	1	0	1
D	D	A	D	D	0	1	0
	Q*						

根据状态表可用 case 语句决定下一状态（snext），用 with selet 决定输出（reset,enable,show）。

4.5.4.2 仿真结果

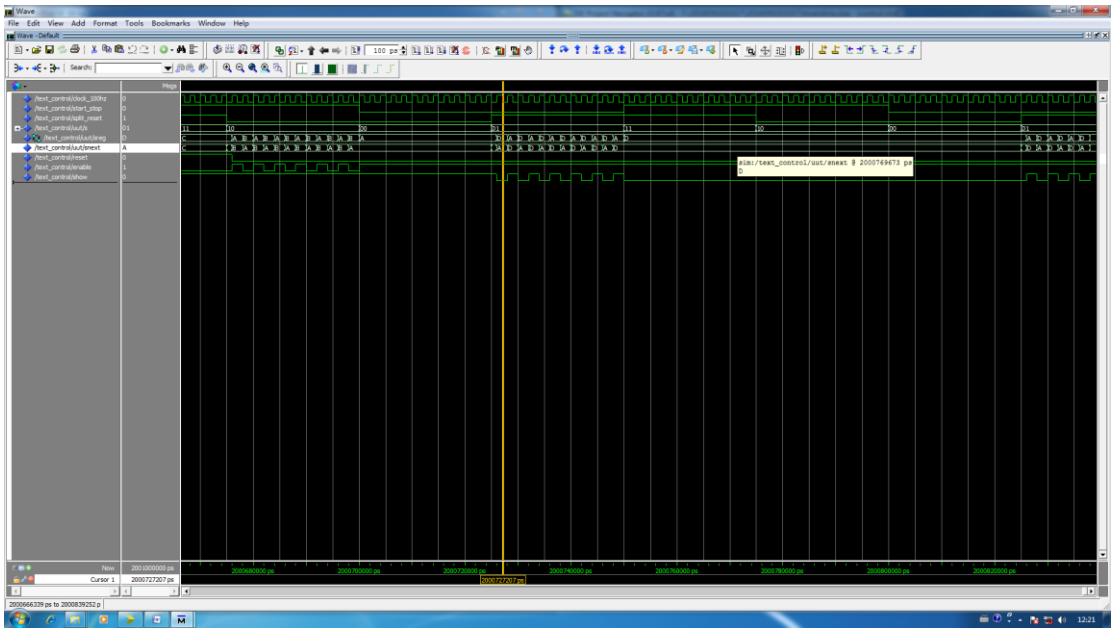


图 7 控制器仿真结果

4.5.5 显示模块

4.5.5.1 介绍

显示模块分为显示时钟 cloch_led 的片选和译码模块，其中译码部分调用 bcd.vhd 文件进行 0~9 数字的显示，片选部分用八个状态循环来控制八个数码管的亮灭。

4.5.5.2 仿真结果

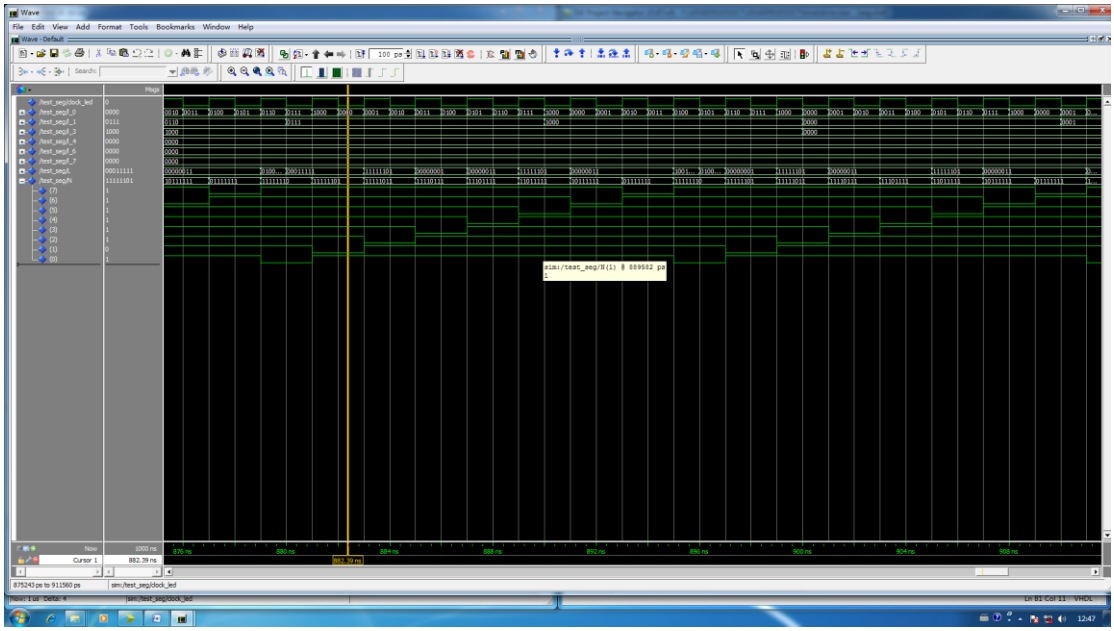


图 8 显示器仿真结果

4.5.6 按键消抖电路

4.5.6.1 介绍

用已有的按键消抖电路的 VHDL 文件代码。

4.6 实现结果

经过综合、仿真、下载到 FPGA 里之后，实验板上很好地实现了所有的功能，如下图：

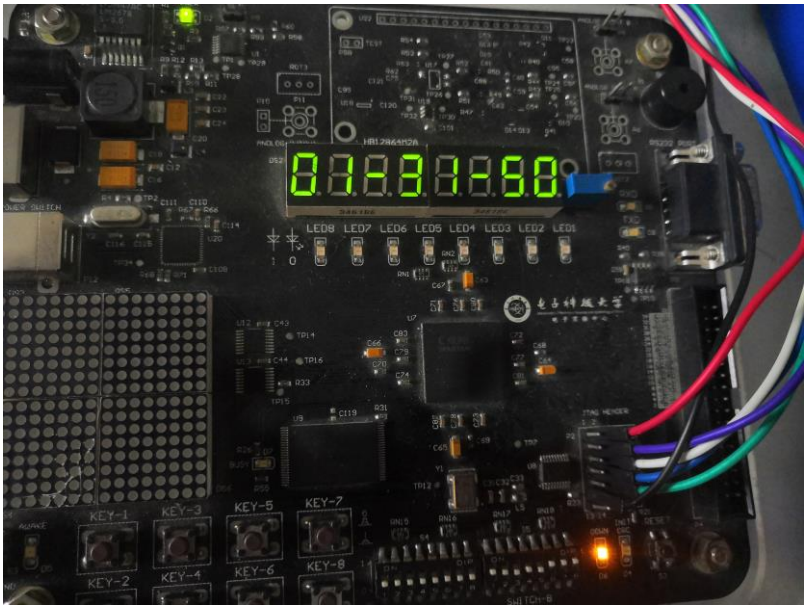


图 9 实现结果

第5章 结论

本次实验严格根据基于 FPGA 的 VHDL 设计流程，从顶层出发，由状态图、状态表开始将整个工程分为数个模块，分模块化编程，高效率完成了实验要求，但是由于对优化上下功夫不够，导致我编写的的分频文件有些瑕疵；对 if 语句和 case 语句的逻辑综合后的电路结果理解不深刻，控制文件的状态转化部分出现优先级的问题，我仍需要大量的练习来巩固我的学习成果。

参考文献

【1】数字设计原理与实践（原书第四版） 林生 葛红 金京林 译 机械工业出版社 2016 年 3 月第 7 次印刷。
【2】FPGA/CPLD 设计与实践教程 沈莉丽 卢家凰 张志立 编 中国电力出版社出版 2017 年 1 月第一版。
【3】Verilog 经典教程 夏雨闻 著 北京航空航天大学出版社
【4】VHDL 数字电路设计教程 乔庐峰 王志功 等译 电子工业出版社 2009 年 12 月第 5 次印刷

致谢

非常感谢黄晓辉老师对我理解状态机设计的帮助以及在实现过程中出现问题的讲解，感谢刘曦老师对我在 VHDL 语法和工程优化方面的指导，让我对 VHDL 语言的综合规则有更深刻的理解。

附录（程序代码）

附录一：源文件

顶层模块：time.vhd

```
----------------------------------------------------------------------------
-------------
-- Company:
-- Engineer:
--
-- Create Date:    13:43:10 03/31/2019
-- Design Name:
-- Module Name:    time - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
---- Revision 0.01 - File Created
---- Additional Comments:
----
--Device utilization summary:
-----------------------------
--
--Selected Device : 3s200aft256-5
--
-- Number of Slices:                      89  out of   1792     4%
-- Number of Slice Flip Flops:            85  out of   3584     2%
-- Number of 4 input LUTs:               162  out of   3584     4%
-- Number of IOs:                        19
-- Number of bonded IOBs:                 19  out of    195     9%
-- Number of GCLKs:                       3  out of     24    12%
--
----------------------------------------------------------------------------
---------------
```

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity time is
    Port ( clock : in  STD_LOGIC;
           start_stop : in  STD_LOGIC;
           split_reset : in  STD_LOGIC;
           L : out  STD_LOGIC_VECTOR (7 downto 0);
           N : out  STD_LOGIC_VECTOR (7 downto 0));
end time;

architecture Behavioral of time is
---------------------------------------------------
component frequence is
    Port ( clock : in  STD_LOGIC;
           clock_100hz : out  STD_LOGIC;
              clock_led : out  STD_LOGIC
              );
end component frequence;
---------------------------------------------------
component key_debounce is
    Port ( clk : in  STD_LOGIC;
           key_in : in  STD_LOGIC;
           key_out : out  STD_LOGIC);
end component key_debounce;
---------------------------------------------------
component control is
    Port ( clock_100hz : in std_logic;
            start_stop : in  STD_LOGIC;
           split_reset : in  STD_LOGIC;
           reset : out  STD_LOGIC;
           enable : out  STD_LOGIC;
           show : out  STD_LOGIC);
```

```
end component control;
-------------------------------------------------
component counter is
    Port ( clock_100hz : in  STD_LOGIC;
             reset : in std_logic;
             enable : in std_logic;
           led_0 : out  STD_LOGIC_VECTOR (3 downto 0);
           led_1 : out  STD_LOGIC_VECTOR (3 downto 0);
           led_3 : out  STD_LOGIC_VECTOR (3 downto 0);
           led_4 : out  STD_LOGIC_VECTOR (3 downto 0);
           led_6 : out  STD_LOGIC_VECTOR (3 downto 0);
           led_7 : out  STD_LOGIC_VECTOR (3 downto 0));
end component counter;
-------------------------------------------------
component latch is
    Port ( latch_signal : in  STD_LOGIC;
           led_0 : in  STD_LOGIC_VECTOR (3 downto 0);
           led_1 : in  STD_LOGIC_VECTOR (3 downto 0);
           led_3 : in  STD_LOGIC_VECTOR (3 downto 0);
           led_4 : in  STD_LOGIC_VECTOR (3 downto 0);
           led_6 : in  STD_LOGIC_VECTOR (3 downto 0);
           led_7 : in  STD_LOGIC_VECTOR (3 downto 0);
           latch_0 : out  STD_LOGIC_VECTOR (3 downto 0);
           latch_1 : out  STD_LOGIC_VECTOR (3 downto 0);
           latch_3 : out  STD_LOGIC_VECTOR (3 downto 0);
           latch_4 : out  STD_LOGIC_VECTOR (3 downto 0);
           latch_6 : out  STD_LOGIC_VECTOR (3 downto 0);
           latch_7 : out  STD_LOGIC_VECTOR (3 downto 0));
end component latch;
-------------------------------------------------
component seg is
    Port ( clock_led : in  STD_LOGIC;
           l_0 : in  STD_LOGIC_VECTOR (3 downto 0);
           l_1 : in  STD_LOGIC_VECTOR (3 downto 0);
           l_3 : in  STD_LOGIC_VECTOR (3 downto 0);
           l_4 : in  STD_LOGIC_VECTOR (3 downto 0);
           l_6 : in  STD_LOGIC_VECTOR (3 downto 0);
           l_7 : in  STD_LOGIC_VECTOR (3 downto 0);
           L : out  STD_LOGIC_VECTOR (7 downto 0);
           N : out  STD_LOGIC_VECTOR (7 downto 0));
end component seg;
-------------------------------------------------
-----------------------------------------------------
-------------------------------------------------
```

```
------------------------------------------------------
------------------------------------------------------
------------------------------------------------------
signal
clock_t,start_stop_t,split_reset_t,clock_100hz_t,clock_led_t,start_st
op_key_out_t,split_reset_key_out_t,reset_t,
       enable_t,show_t:std_logic;
signal
led_0_t,led_1_t,led_3_t,led_4_t,led_6_t,led_7_t,latch_0_t,latch_1_t,l
atch_3_t,latch_4_t,
    latch_6_t,latch_7_t:std_logic_vector(3 downto 0);

signal L_t,N_t:std_logic_vector(7 downto 0);
begin
    clock_t<=clock;
    start_stop_t<=start_stop;
    split_reset_t<=split_reset;
    T1:frequence port map (clock_t,clock_100hz_t,clock_led_t);
    T2:key_debounce port map
(clock_100hz_t,start_stop_t,start_stop_key_out_t);
    T3:key_debounce port map
(clock_100hz_t,split_reset_t,split_reset_key_out_t);
    T4:control port map
(clock_100hz_t,start_stop_key_out_t,split_reset_key_out_t,reset_t,ena
ble_t,show_t);
    T5:counter port map
(clock_100hz_t,reset_t,enable_t,led_0_t,led_1_t,led_3_t,led_4_t,led_6
_t,led_7_t);
    T6:latch port map
(show_t,led_0_t,led_1_t,led_3_t,led_4_t,led_6_t,led_7_t,latch_0_t,lat
ch_1_t,latch_3_t,latch_4_t,latch_6_t,
       latch_7_t);
    T7:seg port map
(clock_led_t,latch_0_t,latch_1_t,latch_3_t,latch_4_t,latch_6_t,latch_
7_t,L_t,N_t);
    L<=L_t;
    N<=N_t;

end Behavioral;
```

分频器：frequence.vhd

```
----------------------------------------------------------------------------
-------------
-- Company:
-- Engineer:
--
-- Create Date:    13:07:14 03/31/2019
-- Design Name:
-- Module Name:    frequence - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;


-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity frequence is
    Port ( clock : in  STD_LOGIC;
           clock_100hz : out  STD_LOGIC;
              clock_led : out  STD_LOGIC
              );
end frequence;
```

```
architecture Behavioral of frequence is
signal counter : std_logic_vector(3 downto 0) := (others => '0');
signal counter1 : std_logic_vector(14 downto 0) := (others => '0');
signal s,s1 : std_logic:='0';
begin

process(clock)--1khz
    begin
    if rising_edge(clock) then
        counter1<=counter1 +1;
        if counter1=23999 then
            s1<=not s1;
            counter1<=(others => '0');
        end if;
    end if;
end process;

process(s1)--100hz
    begin
    if rising_edge(s1) then
        counter<=counter +1;
        if counter=4 then
            s<=not s;
            counter<=(others => '0');
        end if;
    end if;
end process;


clock_100hz<=s;
clock_led<=s1;


end Behavioral;
```

计数器：counter.vhd

```
----------------------------------------------------------------------------
-------------
-- Company:
-- Engineer:
--
```

```
-- Create Date:    13:20:56 03/31/2019
-- Design Name:
-- Module Name:    counter - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity counter is
    Port ( clock_100hz : in  STD_LOGIC;
              reset : in std_logic;
              enable : in std_logic;
           led_0 : out  STD_LOGIC_VECTOR (3 downto 0);
           led_1 : out  STD_LOGIC_VECTOR (3 downto 0);
           led_3 : out  STD_LOGIC_VECTOR (3 downto 0);
           led_4 : out  STD_LOGIC_VECTOR (3 downto 0);
           led_6 : out  STD_LOGIC_VECTOR (3 downto 0);
           led_7 : out  STD_LOGIC_VECTOR (3 downto 0));
end counter;

architecture Behavioral of counter is
```

```
-------------------------------------
component counter_10 is
    Port ( clock_one : in  STD_LOGIC;
           clear : in  STD_LOGIC;
           c_in : in  STD_LOGIC;
           c_out : out  STD_LOGIC;
           s : out  STD_LOGIC_VECTOR (3 downto 0));
end component counter_10;
-----------------------------------------
component counter_6 is
    Port ( clock_one : in  STD_LOGIC;
           clear : in  STD_LOGIC;
           c_in : in  STD_LOGIC;
           c_out : out  STD_LOGIC;
           s : out  STD_LOGIC_VECTOR (3 downto 0));
end component counter_6;
-------------------------------------------
signal c_out_0,c_out_1,c_out_3,c_out_4,
               c_out_6,c_out_7: std_logic;--½øÎ»Êä³ö
signal reset_text:std_logic;

begin
reset_text<='1' when (reset='1' or c_out_7='1') else '0';
C0:counter_10 port map(clock_100hz,reset_text,enable,c_out_0,led_0);
C1:counter_10 port map(clock_100hz,reset_text,c_out_0,c_out_1,led_1);
C3:counter_10 port map(clock_100hz,reset_text,c_out_1,c_out_3,led_3);
C4:counter_6 port map(clock_100hz,reset_text,c_out_3,c_out_4,led_4);
C6:counter_10 port map(clock_100hz,reset_text,c_out_4,c_out_6,led_6);
C7:counter_6 port map(clock_100hz,reset_text,c_out_6,c_out_7,led_7);



end Behavioral;
```

十进制计数器：counter_10.vhd

```
----------------------------------------------------------------------------
-------------
-- Company:
-- Engineer:
--
-- Create Date:    14:53:53 03/31/2019
-- Design Name:
-- Module Name:    counter_10 - Behavioral
-- Project Name:
```

```
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity counter_10 is
    Port ( clock_one : in  STD_LOGIC;
           clear : in  STD_LOGIC;
           c_in : in  STD_LOGIC;
           c_out : out  STD_LOGIC;
           s : out  STD_LOGIC_VECTOR (3 downto 0));
end counter_10;

architecture Behavioral of counter_10 is
signal counter: std_logic_vector(3 downto 0);

begin
process(clear,clock_one)
begin
  if clear='1' then
    counter<="0000";
  else if clock_one'event and clock_one='1' then
       if c_in='1' then
```

```
            if counter<"1001" then
                counter<=counter+1;
             else
                counter<="0000";
             end if;
          else
          null;
          end if;
       end if;
  end if;
end process;
s<=counter;
c_out<='1' when c_in='1' and counter="1001" else '0';

end Behavioral;
```

六进制计数器：counter_6.vhd

```
----------------------------------------------------------------------------
-------------
-- Company:
-- Engineer:
--
-- Create Date:    14:57:28 03/31/2019
-- Design Name:
-- Module Name:    counter_6 - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
```

```
-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity counter_6 is
    Port ( clock_one : in  STD_LOGIC;
           clear : in  STD_LOGIC;
           c_in : in  STD_LOGIC;
           c_out : out  STD_LOGIC;
           s : out  STD_LOGIC_VECTOR (3 downto 0));
end counter_6;

architecture Behavioral of counter_6 is
signal counter: std_logic_vector(3 downto 0):=(others=>'0');
begin
process(clear,clock_one)
begin
  if clear='1' then
    counter<="0000";
  else if clock_one'event and clock_one='1' then
       if c_in='1' then
           if counter<"0101" then
              counter<=counter+1;
            else
              counter<="0000";
            end if;
         else
         null;
         end if;
      end if;
  end if;
end process;
s<=counter;
c_out<='1' when c_in='1' and counter="0101" else '0';

end Behavioral;
```

数据锁存器：latch.vhd

```
----------------------------------------------------------------------------
-------------
-- Company:
-- Engineer:
--
-- Create Date:    13:41:25 03/31/2019
-- Design Name:
-- Module Name:    latch - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity latch is
    Port ( latch_signal : in  STD_LOGIC;
           led_0 : in  STD_LOGIC_VECTOR (3 downto 0);
           led_1 : in  STD_LOGIC_VECTOR (3 downto 0);
           led_3 : in  STD_LOGIC_VECTOR (3 downto 0);
           led_4 : in  STD_LOGIC_VECTOR (3 downto 0);
           led_6 : in  STD_LOGIC_VECTOR (3 downto 0);
```

```
          led_7 : in  STD_LOGIC_VECTOR (3 downto 0);
          latch_0 : out  STD_LOGIC_VECTOR (3 downto 0);
          latch_1 : out  STD_LOGIC_VECTOR (3 downto 0);
          latch_3 : out  STD_LOGIC_VECTOR (3 downto 0);
          latch_4 : out  STD_LOGIC_VECTOR (3 downto 0);
          latch_6 : out  STD_LOGIC_VECTOR (3 downto 0);
          latch_7 : out  STD_LOGIC_VECTOR (3 downto 0));
end latch;

architecture Behavioral of latch is
begin
process(latch_signal,led_0,led_1,led_3,led_4,led_6,led_7)
    begin
    if latch_signal='1' then
        latch_0<=led_0;
        latch_1<=led_1;
        latch_3<=led_3;
        latch_4<=led_4;
        latch_6<=led_6;
        latch_7<=led_7;
    end if;
end process;
end Behavioral;
```

控制器：control.vhd

```
----------------------------------------------------------------------------
-------------
-- Company:
-- Engineer:
--
-- Create Date:    13:23:28 03/31/2019
-- Design Name:
-- Module Name:    control - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
```

```
-- Additional Comments:
--
----------------------------------------------------------------------------
-------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity control is
    Port ( clock_100hz : in std_logic;
              start_stop : in  STD_LOGIC;
           split_reset : in  STD_LOGIC;
           reset : out  STD_LOGIC;
           enable : out  STD_LOGIC;
           show : out  STD_LOGIC);
end control;

architecture Behavioral of control is
type state_type is (A,B,C,D);
signal sreg,snext: state_type :=A;
signal s: std_logic_vector(1 downto 0);

begin
    s(0)<=split_reset;
    s(1)<=start_stop;
process(clock_100hz)
    begin
    if rising_edge(clock_100hz) then
        sreg <=snext;
    end if;
    end process;

process(sreg,s)
    begin
```

```
        case sreg is
            when A => case s is
                              when   "00"   => snext <=A;
                              when   "01"   => snext <=D;
                              when   "11"   => snext <=A;
                              when   "10"   => snext <=B;
                              when others   => snext <=A;
                          end case;
            when B => case s is
                              when   "00"   => snext <=B;
                              when   "01"   => snext <=C;
                              when   "11"   => snext <=A;
                              when   "10"   => snext <=A;
                              when others   => snext <=B;
                          end case;
            when C => case s is
                              when   "00"   => snext <=C;
                              when   "01"   => snext <=C;
                              when   "11"   => snext <=C;
                              when   "10"   => snext <=A;
                              when others   => snext <=C;
                          end case;
            when D => case s is
                              when   "00"   => snext <=D;
                              when   "01"   => snext <=A;
                              when   "11"   => snext <=D;
                              when   "10"   => snext <=D;
                              when others   => snext <=D;
                          end case;
            when others   => snext<=A;
        end case;
end process;

with sreg select
    reset <= '0'  when   A,
             '0' when   B,
             '1' when   C,
             '0' when   D;

with sreg select
    enable <= '1' when   A,
              '0'   when   B,
              '0'   when   C,
              '1'   when   D;
```

```
with sreg select
    show <=  '1'   when   A,
             '1' when   B,
             '1' when   C,
             '0' when   D;

end Behavioral;
```

显示模块：seg.vhd

```
----------------------------------------------------------------------------
-------------
-- Company:
-- Engineer:
--
-- Create Date:    13:36:58 03/31/2019
-- Design Name:
-- Module Name:    seg - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
```

```
--library UNISIM;
--use UNISIM.VComponents.all;

entity seg is
    Port ( clock_led : in  STD_LOGIC;
           l_0 : in  STD_LOGIC_VECTOR (3 downto 0);
           l_1 : in  STD_LOGIC_VECTOR (3 downto 0);
           l_3 : in  STD_LOGIC_VECTOR (3 downto 0);
           l_4 : in  STD_LOGIC_VECTOR (3 downto 0);
           l_6 : in  STD_LOGIC_VECTOR (3 downto 0);
           l_7 : in  STD_LOGIC_VECTOR (3 downto 0);
           L : out  STD_LOGIC_VECTOR (7 downto 0);
           N : out  STD_LOGIC_VECTOR (7 downto 0));
end seg;

architecture Behavioral of seg is
-----------------------------------------------
component bcd is
    Port ( A : out  STD_LOGIC_VECTOR (7 downto 0);
           S : in  STD_LOGIC_VECTOR (3 downto 0));
end component bcd;
----------------------------------------------
type state_type is (A1,B1,C1,D1,E1,F1,G1,H1);
signal sreg,snext: state_type;
signal A,B,C,D,E,F,G,H:  STD_LOGIC_vector (7 downto 0);
signal s0,s1,s3,s4,s6,s7 :std_logic_vector (3 downto 0);
------------------------------------------
begin
process(clock_led)
begin
  if clock_led'event and clock_led='1' then
    sreg <=snext;
  end if;
end process;

process(sreg)  --ˊ㎜ˇ 1kHZ
  begin
       case sreg is
           when A1    =>  snext <=B1;
           when B1    =>  snext <=C1;
           when C1    =>  snext <=D1;
           when D1    =>  snext <=E1;
            when E1    =>  snext <=F1;
            when F1    =>  snext <=G1;
```

```
            when G1       =>  snext <=H1;
            when H1       =>   snext <=A1;
          when others => snext <=A1;
       end case;
end process;
  s0<=l_0;
  s1<=l_1;
  s3<=l_3;
  s4<=l_4;
  s6<=l_6;
  s7<=l_7;
  C<="11111101";
  F<="11111101";
  U0:bcd port map (A,s0);
  U1:bcd port map (B,s1);
  U3:bcd port map (D,s3);
  U4:bcd port map (E,s4);
  U6:bcd port map (G,s6);
  U7:bcd port map (H,s7);

  with sreg select
  L <=  A   when A1,
            B   when B1,
            C   when C1,
            D   when D1,
            E   when E1,
            F   when F1,
            G   when G1,
            H   when H1;

  with sreg select
  N <=  "11111110" when A1,
            "11111101" when B1,
            "11111011" when C1,
            "11110111" when D1,
            "11101111" when E1,
            "11011111" when F1,
            "10111111" when G1,
            "01111111" when H1;
end Behavioral;
```

译码器：bcd. vhd

```
--------------------------------------------------------------------
```

```
----------
-- Company:
-- Engineer:
--
-- Create Date:    13:26:16 03/31/2019
-- Design Name:
-- Module Name:    bcd - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
----------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity bcd is
    Port ( A : out  STD_LOGIC_VECTOR (7 downto 0);
           S : in  STD_LOGIC_VECTOR (3 downto 0));
end bcd;

architecture Behavioral of bcd is

begin
process(S)
    begin
```

```
        case S is        --abcdefg
            when "0000" => A <= "00000011";
            when "0001" => A <= "10011111";
            when "0010" => A <= "00100101";
            when "0011" => A <= "00001101";
            when "0100" => A <= "10011001";
            when "0101" => A <= "01001001";
            when "0110" => A <= "01000001";
            when "0111" => A <= "00011111";
            when "1000" => A <= "00000001";
            when "1001" => A <= "00001001";
            when "1010" => A <= "11111101";
            when others => A <= "00000011";
        end case;
end process;

end Behavioral;
```

消抖电路：key_debounce.vhd

```
----------------------------------------------------------------------------
-------------
-- Company:
-- Engineer:
--
-- Create Date:    12:57:27 03/31/2019
-- Design Name:
-- Module Name:    key_debounce - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
```

```
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity key_debounce is
    Port ( clk : in  STD_LOGIC;
           key_in : in  STD_LOGIC;
           key_out : out  STD_LOGIC);
end key_debounce;

architecture Behavioral of key_debounce is
    signal cnt  : std_logic_vector(2 downto 0);
    signal k1,k2 : std_logic;

begin
p1:process(key_in,clk)
    begin
        if key_in = '1' then
            cnt <= "000";
        elsif falling_edge(clk) then
            if cnt /= 7 then
                cnt <= cnt + 1;
            end if;
        end if;
end process;

p2:process(clk)
    begin
        if falling_edge(clk) then
            if cnt /= 7 then
                k1 <= '0';
            else
                k1 <= '1';
            end if;
            k2 <=  k1;
        end if;
```

```
end process;

p3: key_out <= (not k1)  and k2;
end Behavioral;
```

配置文件：time.ucf

```
# PlanAhead Generated physical constraints

NET "L[0]" LOC = C9;
NET "L[1]" LOC = A9;
NET "L[2]" LOC = B10;
NET "L[3]" LOC = A10;
NET "L[4]" LOC = C10;
NET "L[5]" LOC = C11;
NET "L[6]" LOC = A11;
NET "L[7]" LOC = B12;
NET "N[0]" LOC = N13;
NET "N[1]" LOC = M13;
NET "N[2]" LOC = L13;
NET "N[3]" LOC = K13;
NET "N[4]" LOC = J13;
NET "N[5]" LOC = J12;
NET "N[6]" LOC = H13;
NET "N[7]" LOC = G13;
NET "clock" LOC = N9;
NET "split_reset" LOC = E6;
NET "start_stop" LOC = H7;
```

附录二：仿真文件

分频器：text_frequence.vhd

```
----------------------------------------------------------------------------
-----------
-- Company:
-- Engineer:
--
-- Create Date:   11:11:10 04/02/2019
-- Design Name:
```

```
-- Module Name:   F:/2016050201017/time/text_frequence.vhd
-- Project Name:  time
-- Target Device:
-- Tool versions:
-- Description:
--
-- VHDL Test Bench Created by ISE for module: frequence
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
-- Notes:
-- This testbench has been automatically generated using types std_logic
and
-- std_logic_vector for the ports of the unit under test.  Xilinx
recommends
-- that these types always be used for the top-level I/O of a design in
order
-- to guarantee that the testbench will bind correctly to the
post-implementation
-- simulation model.
--------------------------------------------------------------------------
-----------
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--USE ieee.numeric_std.ALL;

ENTITY text_frequence IS
END text_frequence;

ARCHITECTURE behavior OF text_frequence IS

    -- Component Declaration for the Unit Under Test (UUT)

    COMPONENT frequence
    PORT(
         clock : IN  std_logic;
         clock_100hz : OUT  std_logic;
```

```
        clock_led : OUT  std_logic
       );
    END COMPONENT;


   --Inputs
   signal clock : std_logic := '0';

    --Outputs
   signal clock_100hz : std_logic;
   signal clock_led : std_logic;


BEGIN

    -- Instantiate the Unit Under Test (UUT)
   uut: frequence PORT MAP (
         clock => clock,
         clock_100hz => clock_100hz,
         clock_led => clock_led
       );

   -- Clock process definitions
    process
        begin
            clock <= '0';
            wait for 10 ns;
            clock <= '1';
            wait for 10 ns;
    end process;

END;
```

计数器：text_counter.vhd

```
----------------------------------------------------------------------------------
-----------
-- Company:
-- Engineer:
--
-- Create Date:   17:51:07 03/31/2019
-- Design Name:
-- Module Name:   F:/2016050201017/time/text_counter.vhd
```

```
-- Project Name:  time
-- Target Device:
-- Tool versions:
-- Description:
--
-- VHDL Test Bench Created by ISE for module: counter
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
-- Notes:
-- This testbench has been automatically generated using types std_logic
and
-- std_logic_vector for the ports of the unit under test.  Xilinx
recommends
-- that these types always be used for the top-level I/O of a design in
order
-- to guarantee that the testbench will bind correctly to the
post-implementation
-- simulation model.
--------------------------------------------------------------------------
-----------
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--USE ieee.numeric_std.ALL;

ENTITY text_counter IS
END text_counter;

ARCHITECTURE behavior OF text_counter IS

    -- Component Declaration for the Unit Under Test (UUT)

    COMPONENT counter
    PORT(
         clock_100hz : IN  std_logic;
         reset : IN  std_logic;
         enable : IN  std_logic;
```

```
        led_0 : OUT  std_logic_vector(3 downto 0);
        led_1 : OUT  std_logic_vector(3 downto 0);
        led_3 : OUT  std_logic_vector(3 downto 0);
        led_4 : OUT  std_logic_vector(3 downto 0);
        led_6 : OUT  std_logic_vector(3 downto 0);
        led_7 : OUT  std_logic_vector(3 downto 0)
       );
    END COMPONENT;


   --Inputs
    signal clock_100hz : std_logic;
   signal reset :  std_logic;
   signal enable :  std_logic;
    --Outputs
   signal led_0 : std_logic_vector(3 downto 0);
   signal led_1 : std_logic_vector(3 downto 0);
   signal led_3 : std_logic_vector(3 downto 0);
   signal led_4 : std_logic_vector(3 downto 0);
   signal led_6 : std_logic_vector(3 downto 0);
   signal led_7 : std_logic_vector(3 downto 0);



BEGIN

    -- Instantiate the Unit Under Test (UUT)
   uut: counter PORT MAP (
         clock_100hz => clock_100hz,
         reset => reset,
         enable => enable,
         led_0 => led_0,
         led_1 => led_1,
         led_3 => led_3,
         led_4 => led_4,
         led_6 => led_6,
         led_7 => led_7
       );
    process
        begin
            clock_100hz<='0';
            wait for 10 ns;
            clock_100hz<='1';
            wait for 10 ns;
```

```
    end process;

    process
        begin
            reset<='1';
            wait for 10 ns;
            reset<='0';
            wait for 100 ms;
    end process;

    process
        begin
            enable<='1';
            wait for 100 ms;
            enable<='0';
            wait for 10 ns;
    end process;

END;
```

数据锁存器：text_latch.vhd

```
--------------------------------------------------------------------------------
-----------
-- Company:
-- Engineer:
--
-- Create Date:   12:27:37 04/14/2019
-- Design Name:
-- Module Name:   F:/2016050201017/2016050201017/time/text_latch.vhd
-- Project Name:  time
-- Target Device:
-- Tool versions:
-- Description:
--
-- VHDL Test Bench Created by ISE for module: latch
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
```

```
-- Notes:
-- This testbench has been automatically generated using types std_logic
and
-- std_logic_vector for the ports of the unit under test.  Xilinx
recommends
-- that these types always be used for the top-level I/O of a design in
order
-- to guarantee that the testbench will bind correctly to the
post-implementation
-- simulation model.
--------------------------------------------------------------------------
-----------
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--USE ieee.numeric_std.ALL;

ENTITY text_latch IS
END text_latch;

ARCHITECTURE behavior OF text_latch IS

    -- Component Declaration for the Unit Under Test (UUT)

    COMPONENT latch
    PORT(
         latch_signal : IN  std_logic;
         led_0 : IN  std_logic_vector(3 downto 0);
         led_1 : IN  std_logic_vector(3 downto 0);
         led_3 : IN  std_logic_vector(3 downto 0);
         led_4 : IN  std_logic_vector(3 downto 0);
         led_6 : IN  std_logic_vector(3 downto 0);
         led_7 : IN  std_logic_vector(3 downto 0);
         latch_0 : OUT  std_logic_vector(3 downto 0);
         latch_1 : OUT  std_logic_vector(3 downto 0);
         latch_3 : OUT  std_logic_vector(3 downto 0);
         latch_4 : OUT  std_logic_vector(3 downto 0);
         latch_6 : OUT  std_logic_vector(3 downto 0);
         latch_7 : OUT  std_logic_vector(3 downto 0)
        );
    END COMPONENT;
```

```
   --Inputs
   signal latch_signal : std_logic := '0';
   signal led_0 : std_logic_vector(3 downto 0) := (others => '0');
   signal led_1 : std_logic_vector(3 downto 0) := (others => '0');
   signal led_3 : std_logic_vector(3 downto 0) := (others => '0');
   signal led_4 : std_logic_vector(3 downto 0) := (others => '0');
   signal led_6 : std_logic_vector(3 downto 0) := (others => '0');
   signal led_7 : std_logic_vector(3 downto 0) := (others => '0');

    --Outputs
   signal latch_0 : std_logic_vector(3 downto 0);
   signal latch_1 : std_logic_vector(3 downto 0);
   signal latch_3 : std_logic_vector(3 downto 0);
   signal latch_4 : std_logic_vector(3 downto 0);
   signal latch_6 : std_logic_vector(3 downto 0);
   signal latch_7 : std_logic_vector(3 downto 0);
   -- No clocks detected in port list. Replace <clock> below with
   -- appropriate port name


BEGIN

    -- Instantiate the Unit Under Test (UUT)
   uut: latch PORT MAP (
         latch_signal => latch_signal,
         led_0 => led_0,
         led_1 => led_1,
         led_3 => led_3,
         led_4 => led_4,
         led_6 => led_6,
         led_7 => led_7,
         latch_0 => latch_0,
         latch_1 => latch_1,
         latch_3 => latch_3,
         latch_4 => latch_4,
         latch_6 => latch_6,
         latch_7 => latch_7
       );

   -- Clock process definitions
    process
       begin
           latch_signal<='0';
```

```
        wait for 10 ns;
        latch_signal<='1';
        wait for 10 ns;
end process;

process
    begin
        led_0<="0000";
        wait for 1 ns;
        led_0<="0001";
        wait for 1 ns;
        led_0<="0010";
        wait for 1 ns;
        led_0<="0011";
        wait for 1 ns;
        led_0<="0100";
        wait for 1 ns;
        led_0<="0101";
        wait for 1 ns;
        led_0<="0110";
        wait for 1 ns;
        led_0<="0111";
        wait for 1 ns;
        led_0<="1000";
        wait for 1 ns;
end process;

process
    begin
        led_1<="0000";
        wait for 10 ns;
        led_1<="0001";
        wait for 10 ns;
        led_1<="0010";
        wait for 10 ns;
        led_1<="0011";
        wait for 10 ns;
        led_1<="0100";
        wait for 10 ns;
        led_1<="0101";
        wait for 10 ns;
        led_1<="0110";
        wait for 10 ns;
        led_1<="0111";
```

```
        wait for 10 ns;
        led_1<="1000";
        wait for 10 ns;
end process;

process
    begin
        led_3<="0000";
        wait for 100 ns;
        led_3<="0001";
        wait for 100 ns;
        led_3<="0010";
        wait for 100 ns;
        led_3<="0011";
        wait for 100 ns;
        led_3<="0100";
        wait for 100 ns;
        led_3<="0101";
        wait for 100 ns;
        led_3<="0110";
        wait for 100 ns;
        led_3<="0111";
        wait for 100 ns;
        led_3<="1000";
        wait for 100 ns;
end process;

process
    begin
        led_4<="0000";
        wait for 1 ms;
        led_4<="0001";
        wait for 1 ms;
end process;

process
    begin
        led_6<="0000";
        wait for 1 ms;
        led_6<="0001";
        wait for 1 ms;
end process;

process
```

```
        begin
            led_7<="0000";
            wait for 1 ms;
            led_7<="0001";
            wait for 1 ms;
    end process;

END;
```

控制器：text_control.vhd

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date:   12:08:04 04/14/2019
-- Design Name:
-- Module Name:   F:/2016050201017/2016050201017/time/text_control.vhd
-- Project Name:  time
-- Target Device:
-- Tool versions:
-- Description:
--
-- VHDL Test Bench Created by ISE for module: control
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
-- Notes:
-- This testbench has been automatically generated using types std_logic
and
-- std_logic_vector for the ports of the unit under test.  Xilinx
recommends
-- that these types always be used for the top-level I/O of a design in
order
-- to guarantee that the testbench will bind correctly to the
post-implementation
-- simulation model.
----------------------------------------------------------------------------------
```

```
----------
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--USE ieee.numeric_std.ALL;

ENTITY text_control IS
END text_control;

ARCHITECTURE behavior OF text_control IS

    -- Component Declaration for the Unit Under Test (UUT)

    COMPONENT control
    PORT(
         clock_100hz : IN  std_logic;
         start_stop : IN  std_logic;
         split_reset : IN  std_logic;
         reset : OUT  std_logic;
         enable : OUT  std_logic;
         show : OUT  std_logic
        );
    END COMPONENT;


   --Inputs
   signal clock_100hz : std_logic := '0';
   signal start_stop : std_logic := '0';
   signal split_reset : std_logic := '0';

    --Outputs
   signal reset : std_logic;
   signal enable : std_logic;
   signal show : std_logic;


BEGIN

    -- Instantiate the Unit Under Test (UUT)
   uut: control PORT MAP (
          clock_100hz => clock_100hz,
          start_stop => start_stop,
```

```
        split_reset => split_reset,
        reset => reset,
        enable => enable,
        show => show
      );

  -- Clock process definitions
   process
      begin
          clock_100hz<='1';
          wait for 1 ns;
          clock_100hz<='0';
          wait for 1 ns;
   end process;

   process
      begin
          start_stop<='0';
          wait for 50 ns;
          start_stop<='1';
          wait for 50 ns;
   end process;

   process
      begin
          split_reset<='0';
          wait for 25 ns;
          split_reset<='1';
          wait for 50 ns;
          split_reset<='0';
          wait for 25 ns;
   end process;

END;
```

显示模块：text_seg.vhd

```
----------------------------------------------------------------------------------
-----------
-- Company:
-- Engineer:
--
-- Create Date:   12:40:20 04/14/2019
```

```
-- Design Name:
-- Module Name:   F:/2016050201017/2016050201017/time/test_seg.vhd
-- Project Name:  time
-- Target Device:
-- Tool versions:
-- Description:
--
-- VHDL Test Bench Created by ISE for module: seg
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
-- Notes:
-- This testbench has been automatically generated using types std_logic
and
-- std_logic_vector for the ports of the unit under test.  Xilinx
recommends
-- that these types always be used for the top-level I/O of a design in
order
-- to guarantee that the testbench will bind correctly to the
post-implementation
-- simulation model.
--------------------------------------------------------------------------
-----------
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--USE ieee.numeric_std.ALL;

ENTITY test_seg IS
END test_seg;

ARCHITECTURE behavior OF test_seg IS

    -- Component Declaration for the Unit Under Test (UUT)

    COMPONENT seg
    PORT(
         clock_led : IN  std_logic;
```

```
        l_0 : IN  std_logic_vector(3 downto 0);
        l_1 : IN  std_logic_vector(3 downto 0);
        l_3 : IN  std_logic_vector(3 downto 0);
        l_4 : IN  std_logic_vector(3 downto 0);
        l_6 : IN  std_logic_vector(3 downto 0);
        l_7 : IN  std_logic_vector(3 downto 0);
        L : OUT  std_logic_vector(7 downto 0);
        N : OUT  std_logic_vector(7 downto 0)
       );
    END COMPONENT;


   --Inputs
   signal clock_led : std_logic := '0';
   signal l_0 : std_logic_vector(3 downto 0) := (others => '0');
   signal l_1 : std_logic_vector(3 downto 0) := (others => '0');
   signal l_3 : std_logic_vector(3 downto 0) := (others => '0');
   signal l_4 : std_logic_vector(3 downto 0) := (others => '0');
   signal l_6 : std_logic_vector(3 downto 0) := (others => '0');
   signal l_7 : std_logic_vector(3 downto 0) := (others => '0');

    --Outputs
   signal L : std_logic_vector(7 downto 0);
   signal N : std_logic_vector(7 downto 0);

   -- Clock period definitions

BEGIN

    -- Instantiate the Unit Under Test (UUT)
   uut: seg PORT MAP (
         clock_led => clock_led,
         l_0 => l_0,
         l_1 => l_1,
         l_3 => l_3,
         l_4 => l_4,
         l_6 => l_6,
         l_7 => l_7,
         L => L,
         N => N
       );

   -- Clock process definitions
    process
```

```
    begin
        clock_led <= '0';
        wait for 1 ns;
        clock_led <= '1';
        wait for 1 ns;
end process;

process
    begin
        l_0<="0000";
        wait for 1 ns;
        l_0<="0001";
        wait for 1 ns;
        l_0<="0010";
        wait for 1 ns;
        l_0<="0011";
        wait for 1 ns;
        l_0<="0100";
        wait for 1 ns;
        l_0<="0101";
        wait for 1 ns;
        l_0<="0110";
        wait for 1 ns;
        l_0<="0111";
        wait for 1 ns;
        l_0<="1000";
        wait for 1 ns;
end process;

process
    begin
        l_1<="0000";
        wait for 10 ns;
        l_1<="0001";
        wait for 10 ns;
        l_1<="0010";
        wait for 10 ns;
        l_1<="0011";
        wait for 10 ns;
        l_1<="0100";
        wait for 10 ns;
        l_1<="0101";
        wait for 10 ns;
        l_1<="0110";
```

```
        wait for 10 ns;
        l_1<="0111";
        wait for 10 ns;
        l_1<="1000";
        wait for 10 ns;
end process;

process
    begin
        l_3<="0000";
        wait for 100 ns;
        l_3<="0001";
        wait for 100 ns;
        l_3<="0010";
        wait for 100 ns;
        l_3<="0011";
        wait for 100 ns;
        l_3<="0100";
        wait for 100 ns;
        l_3<="0101";
        wait for 100 ns;
        l_3<="0110";
        wait for 100 ns;
        l_3<="0111";
        wait for 100 ns;
        l_3<="1000";
        wait for 100 ns;
end process;

process
    begin
        l_4<="0000";
        wait for 1 ms;
        l_4<="0001";
        wait for 1 ms;
end process;

process
    begin
        l_6<="0000";
        wait for 1 ms;
        l_6<="0001";
        wait for 1 ms;
end process;
```

```
    process
        begin
            l_7<="0000";
            wait for 1 ms;
            l_7<="0001";
            wait for 1 ms;
    end process;
END;
```