# 实验12、寄存器和寄存器传输设计实验报告

姓名： 王祚滨 专业： 信息安全 学号： 3180104933

课程名称： 逻辑与计算机设计基础实验 同组学生姓名： 王国朝、赵卿云

指导老师： 洪奇军 实验地点： 浙江大学紫金港校区东四教学楼509实验日期：2019年 12 月 4 日

# 一、实验目的和要求

1.1掌握寄存器传输电路的工作原理

1.2掌握寄存器传输电路的设计方法

1.3掌握ALU和寄存器传输电路的综合应用

**二、实验内容和原理**

**2.1实验内容：**

任务1：基于ALU的数据传输应用设计

任务2：实现4位加法器应用

**2.2实验原理：**

·寄存器

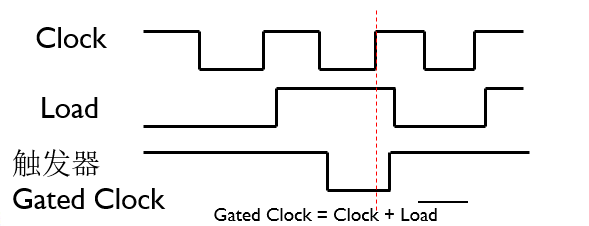
一组二进制存储单元：

一个寄存器可以用于存储一列二进制值，通常用于进行简单数据存储、移动和处理等操作，能存储信息并保存多个时钟周期，能用信号来控制“保存”或“加载”信息

采用门控时钟的寄存器：

如果Load信号为1，允许时钟信号通过，如果为0则阻止时钟信号通过

例如： 对于上升沿触发的边沿触发器或负向脉冲触发的主从触发器：

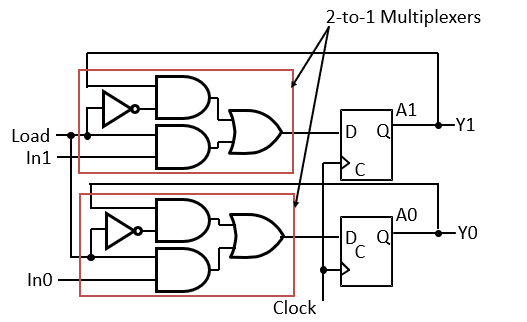


图一

采用Load控制反馈的寄存器

进行有选择地加载寄存器的更可靠方法是：

保证时钟的连续性，且选择性地使用加载控制来改变寄存器的内容。



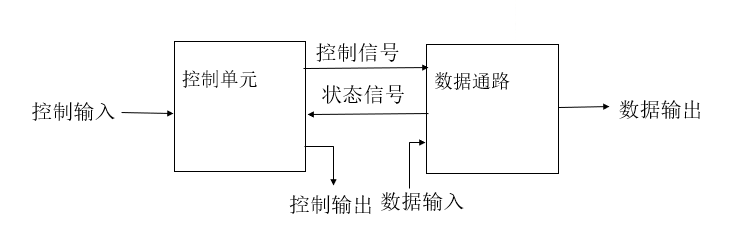
图二

·寄存器传输

寄存器传输：寄存器中数据的传输和处理

三个基本单元：寄存器组、操作、操作控制

基本操作:加载、计数、移位、加法、按位操作等

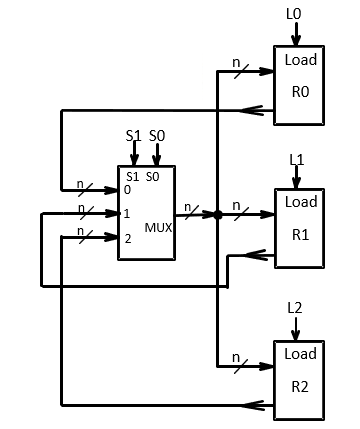


图三

·基于多路选择器总线的寄存器传输

由一个多路选择器驱动的总线可以降低硬件开销

这个结构不能实现多个寄存器相互之间的并行传输操作



图四

·寄存器传输应用设计

1.Mode1：

ALU运算输出控制

2.Mode2：

数据传输控制

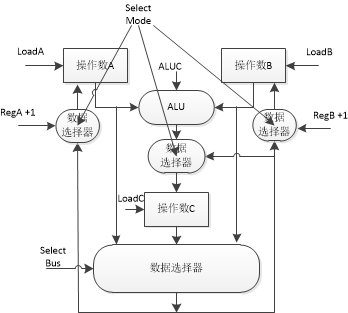


图5 一位加减法器

图五

sw[15]=0 ：Mode0

按键控制输入：sw[2]控制Reg A(自加或自减)，sw[3]控制Reg B，sw[4]对RegC赋值

按键加/减1控制：sw[0]=0加/1：减，对应btn\_out[0]即去抖动后sw[2]，sw[1] =0加/1：减对应btn\_out[1] 即去抖动后的sw[3]，（btn\_out[2]即去抖动后的sw[4]）

ALU运算控制：sw[6:5],00-加，01-减，10-与，11-或。 (RegC是Reg A, Reg B加减与或的结果)

sw[15]=1 Mode1 数据传输控制

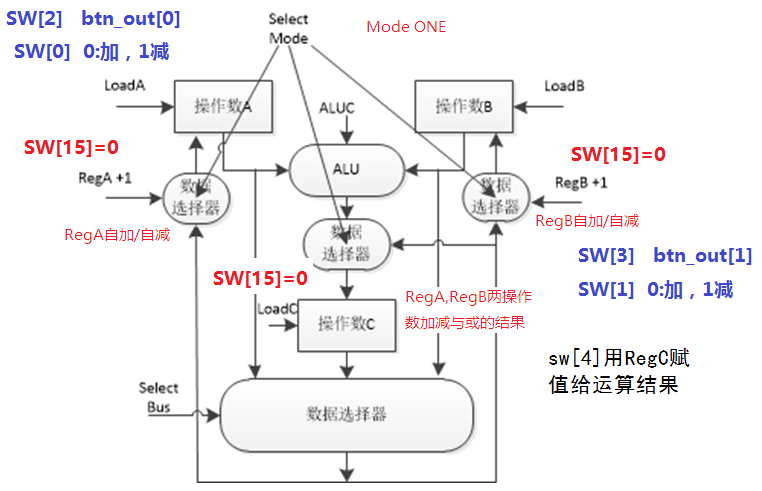
sw[8:7]对应SelectBus：00-选择A，01-选择B，10-选择C

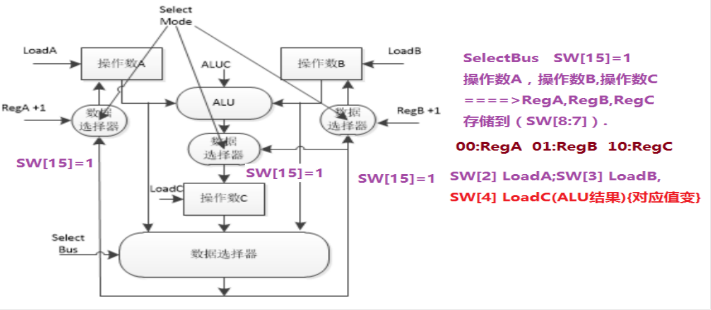
sw[2] LoadA(num[3:0]),sw[3] LoadB(num[7:4])，sw[4] LoadC(ALU结果)

(对SW[8:7]选择出来后的结果加载到哪个寄存器中)

输出 {num[7:0],C,num[11:8]}

AN[0]：Reg B AN[1]：Reg A AN[2]: ALU结果 AN[3]: Reg C





图六

**三、主要仪器设备**

# 1. 装有Xilinx ISE 14.7的计算机 1台

# 2. SWORD开发板

**四、操作方法与实验步骤**

**实验任务：**

基于ALU的数据传输应用设计

**实验步骤：**

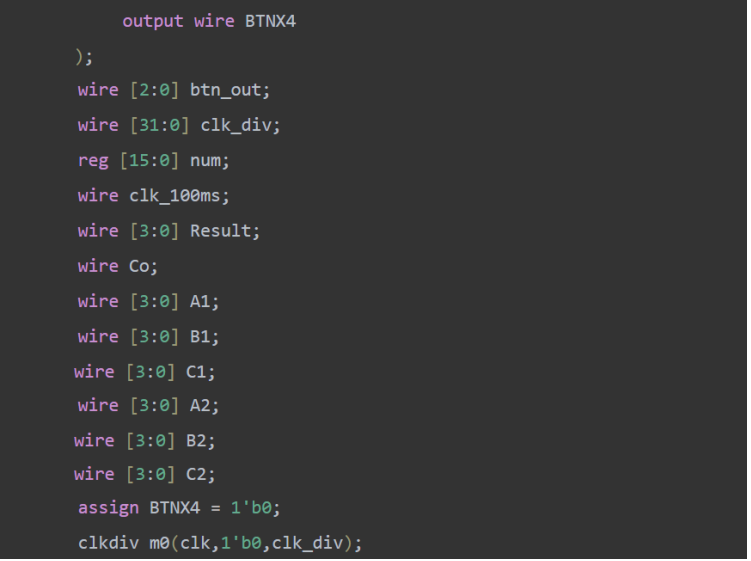
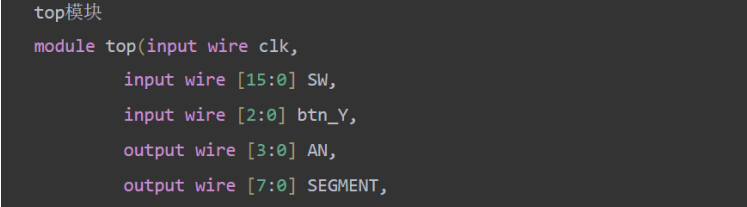
1.新建工程，工程名称用MyALUTrans。

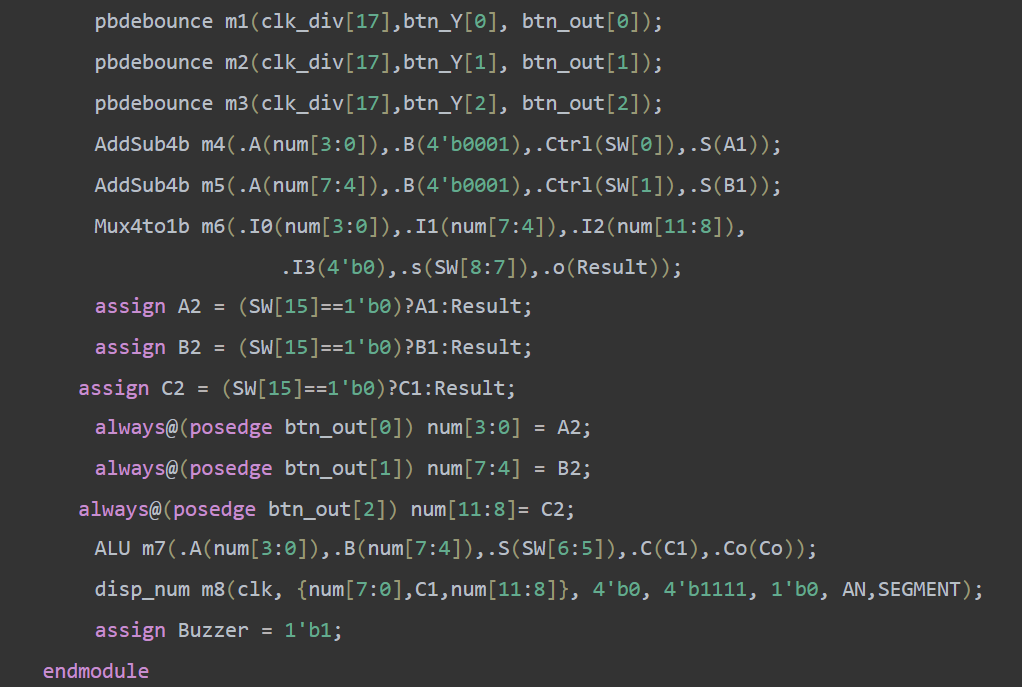
2.Top Level Source Type用HDL

3.添加如下模块：ALU模块，4位4选1模块，防抖动模块，显示模块

4.新建源文件，类型是Verilog，文件名称用Top，右键设为“Set as Top Module”

实现基于ALU的数据传输应用设计

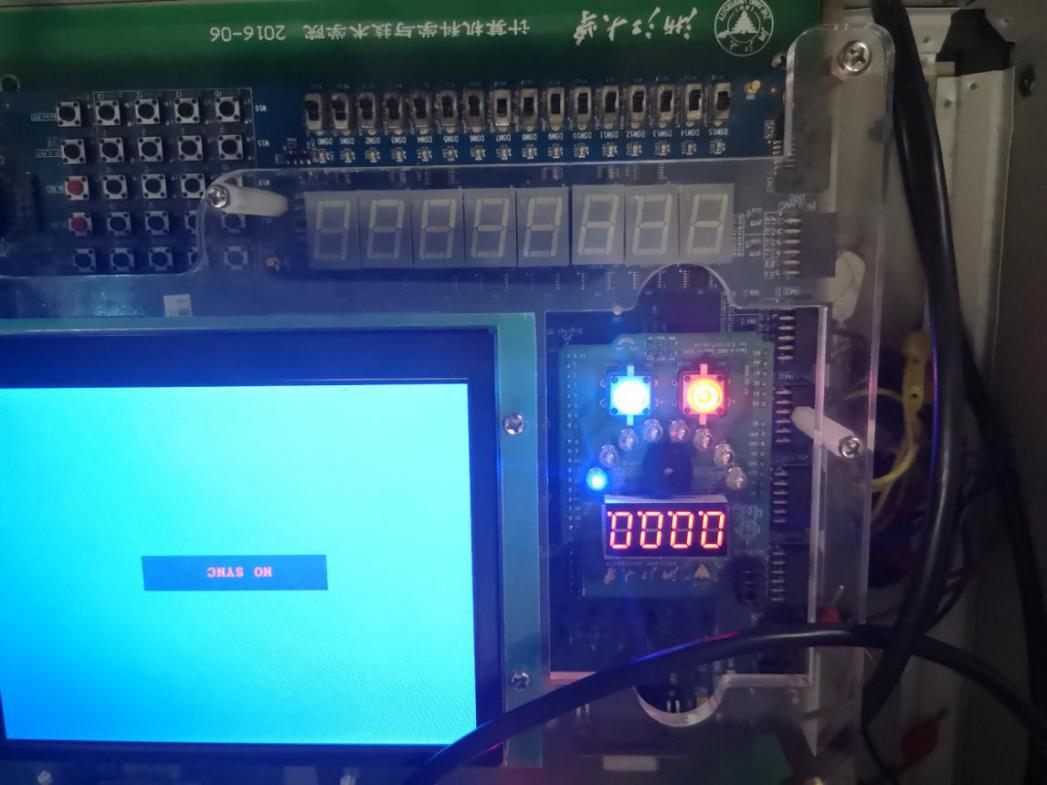




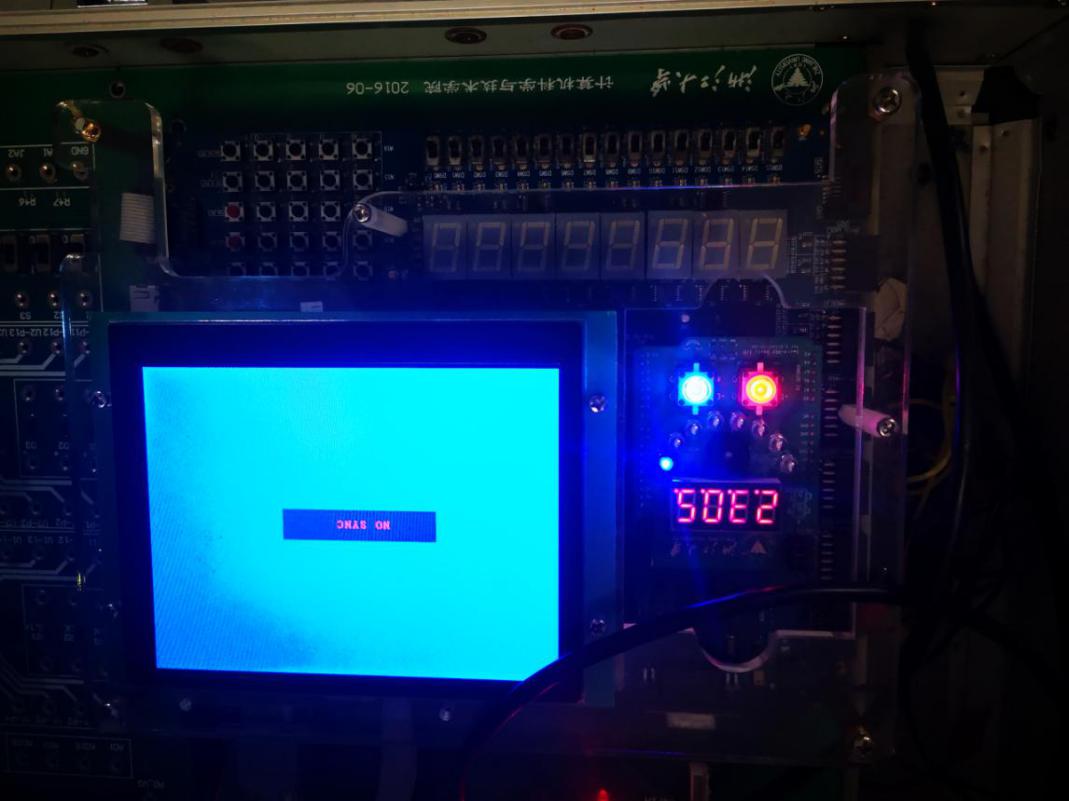
# 

# 

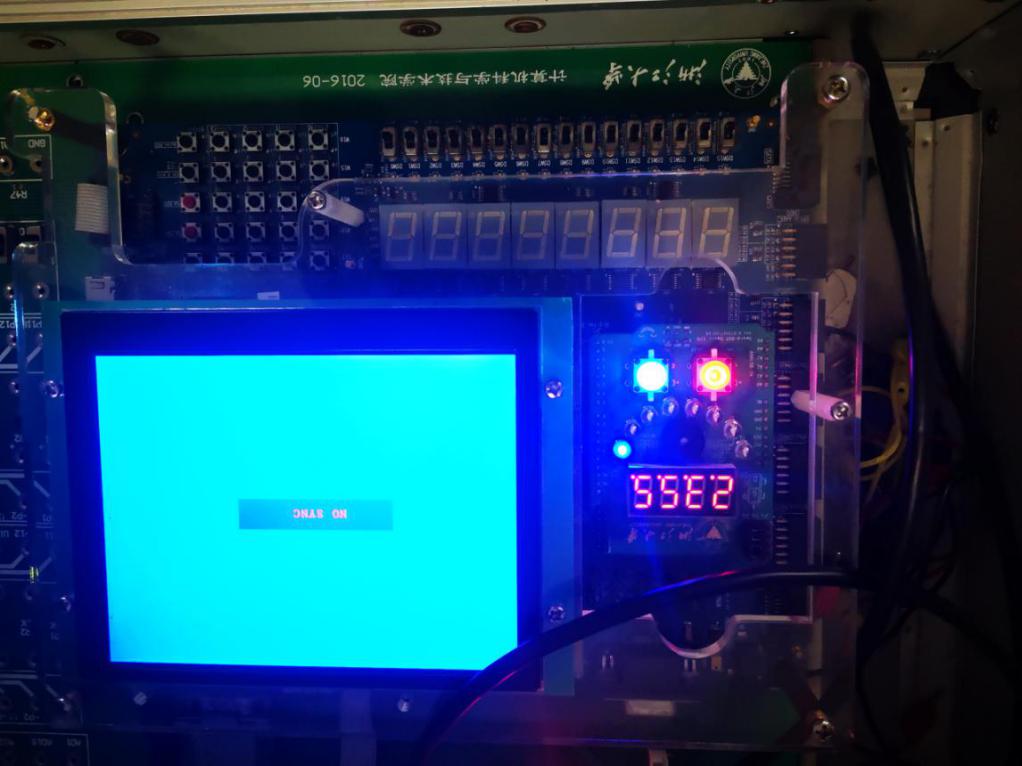
1. **实验结果与分析**

****

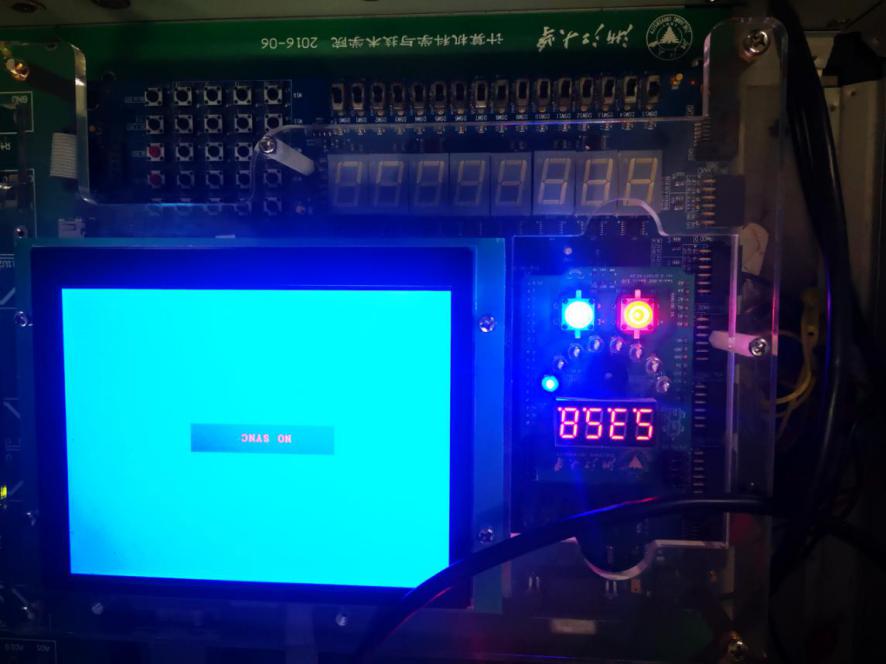
初态：都为0



在第一种模式下，进行加法运算，2+3=5，C没有进行赋值



将第四位赋值给第三位，因此C得到5



切换模式，将第三位赋值给第一位，因此第一位为5，同时result变为8，即5+3的结果

**六、讨论、心得**

此次实验，通过对ALU传输来了解了寄存器的设计，由于对verilog语法还不算熟悉，因此踩到了很多坑，但总算最终还是将结果展现了出来，并且还完成了按钮实现的任务。

通过对以往组件的调用让我对之前各个组件的功能有了更为清晰的认知，但由于在做实验时对寄存器还不够了解，老师在上课时还没有讲到这一部分，因此回去后还需要继续巩固寄存器的相关知识。

通过本次实验，完整的一个流程让我对大程序的实现充满信心。

# 实验13、计数器、定时器设计与应用实验报告

姓名： 王祚滨 专业： 信息安全 学号： 3180104933

课程名称： 逻辑与计算机设计基础实验 同组学生姓名： 王国朝、赵卿云

指导老师： 洪奇军 实验地点： 浙江大学紫金港校区东四教学楼509实验日期：2019年 12 月 4 日

# 一、实验目的和要求

1. 掌握同步四位二进制计数器74LS161的工作原理和设计方法

2.掌握时钟/定时器的工作原理与设计方法

**二、实验内容和原理**

2.1实验内容：

任务1：采用行为描述设计同步四位二进制计数器74LS161

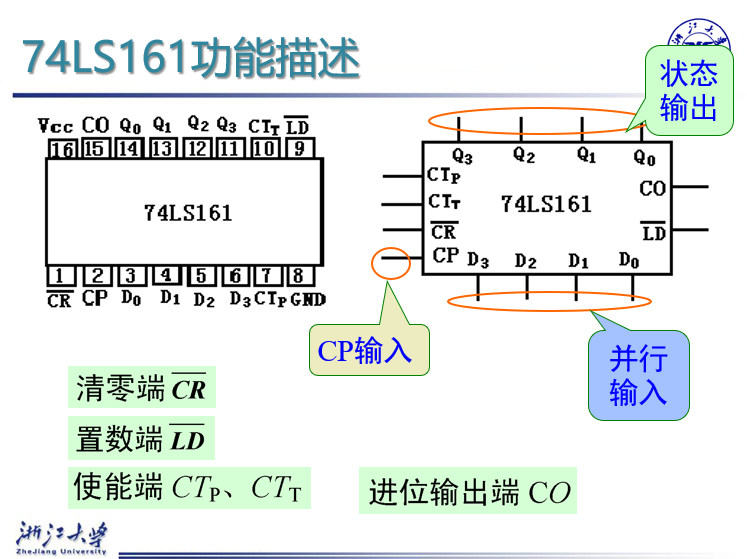
任务2：基于74LS161设计时钟应用

2.2实验原理：

**2.2.1同步四位二进制计数器74LS161**

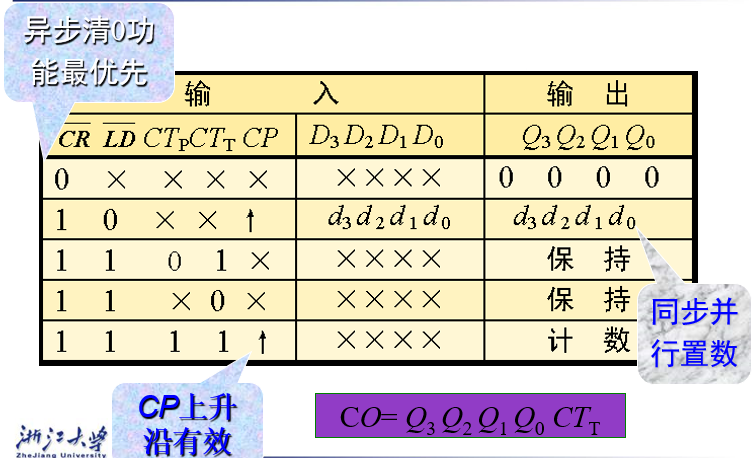
74LS161是常用的四位二进制可预置的同步加法计数器

可灵活运用在各种数字电路，实现分频器等很多重要的功能



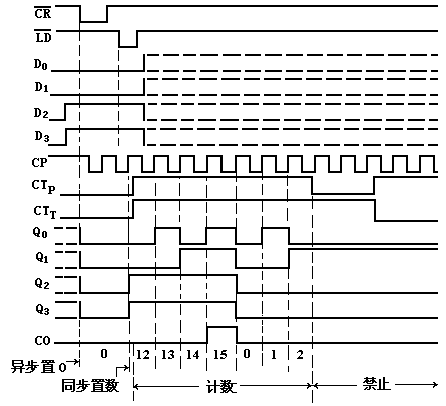
**图一** 74LS161

**74LS161功能表：**



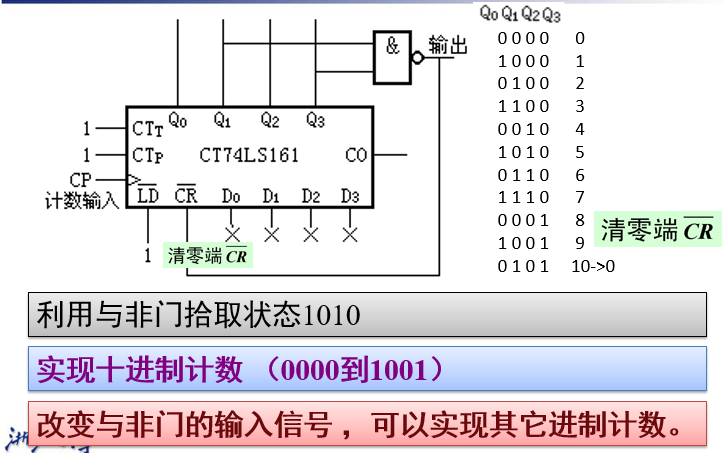
**图二 74LS161功能表**

**74LS161时序图：**

****

**图三 74LS161时序图**

**2.2.2 实现十进制计数器**



**图四 十进制计数器**

# 三、主要仪器设备

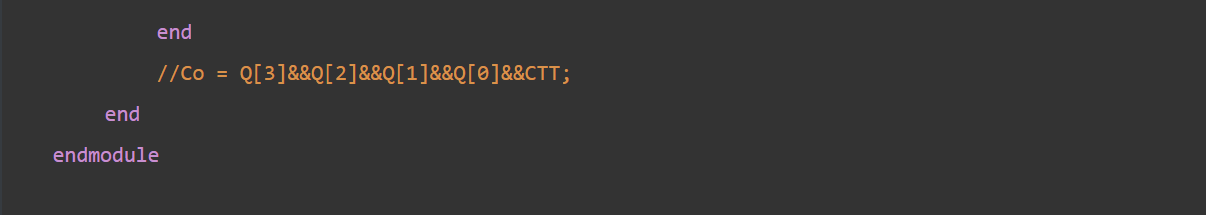
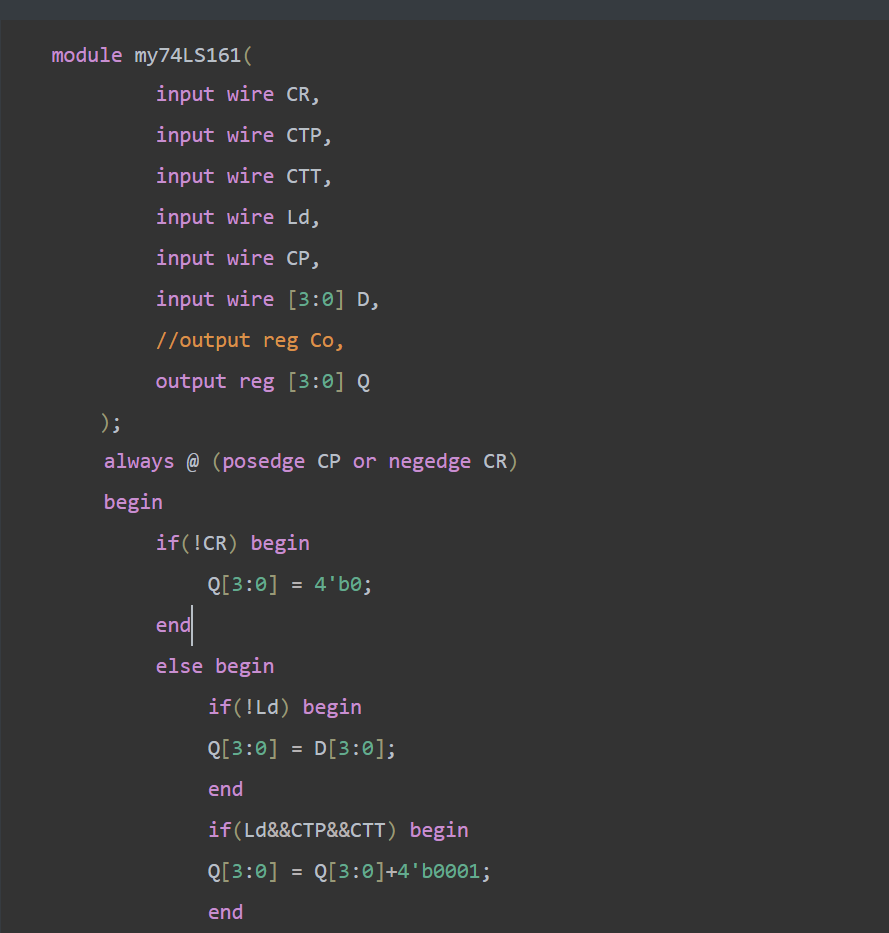
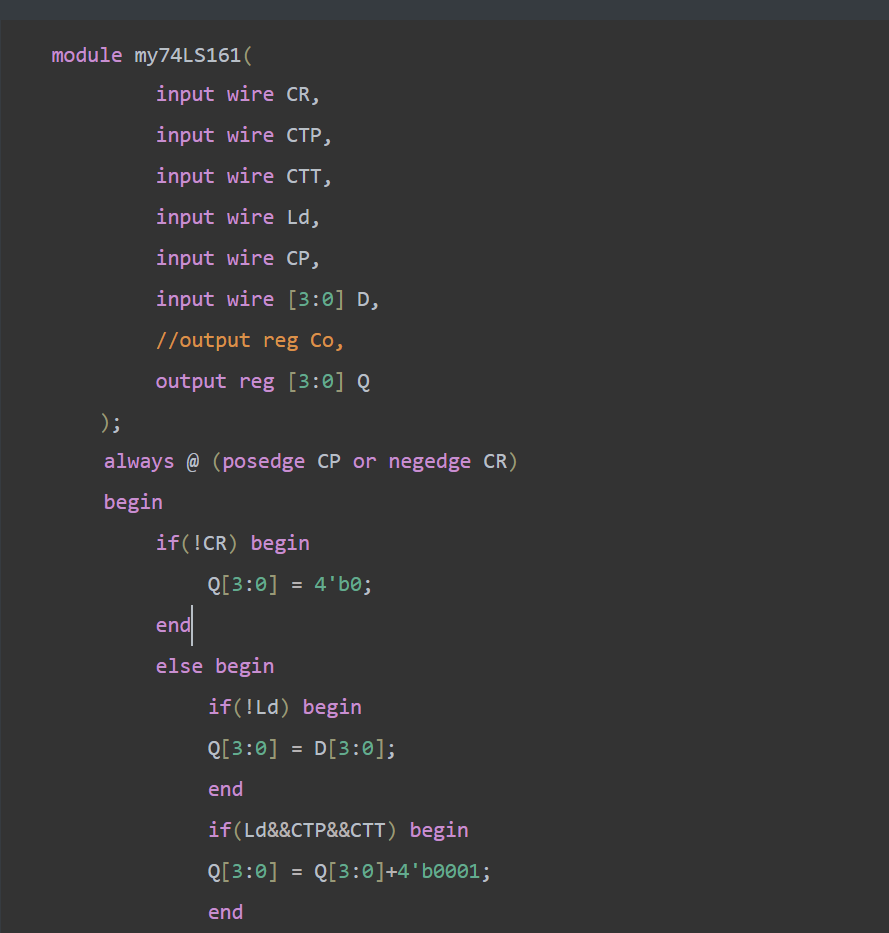
# 1. 装有Xilinx ISE 14.7的计算机 1台

# 2. SWORD开发板

# 四、操作方法与实验步骤

4.1采用行为描述设计同步四位二进制计数器74LS161

将注释去掉即可获得带Co的计数器



4.2 数字时钟

设计一个数字钟，使用60进制和24进制计数器，实现24小时内时间的实时显示。

数字钟的初值通过初始化语句来实现，用数码管前两位显示小时的十位和个位，后两位显示分钟的十位和个位。

过程：

1.新建工程，工程名称用MyClock。

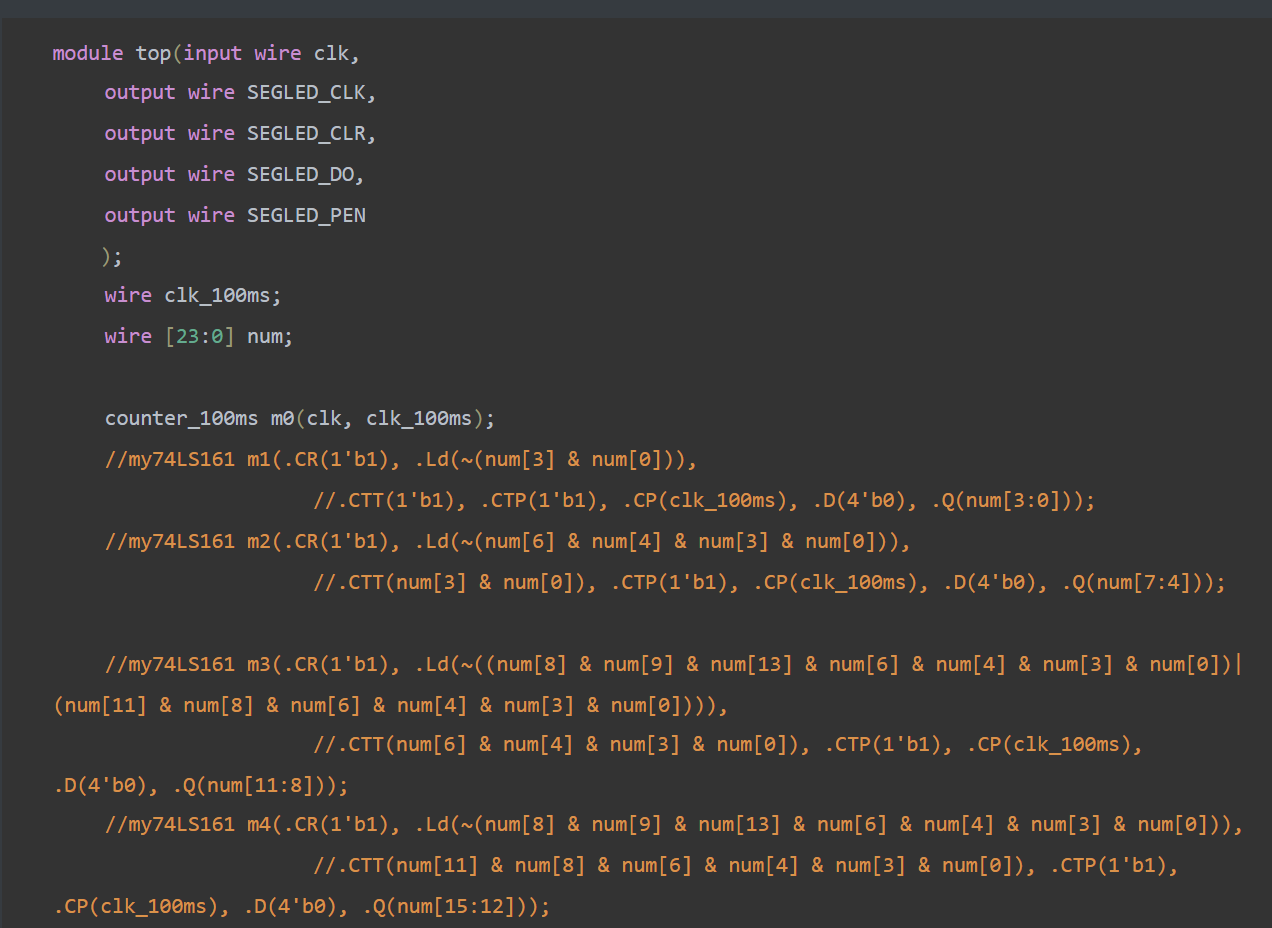
Top Level Source Type用HDL

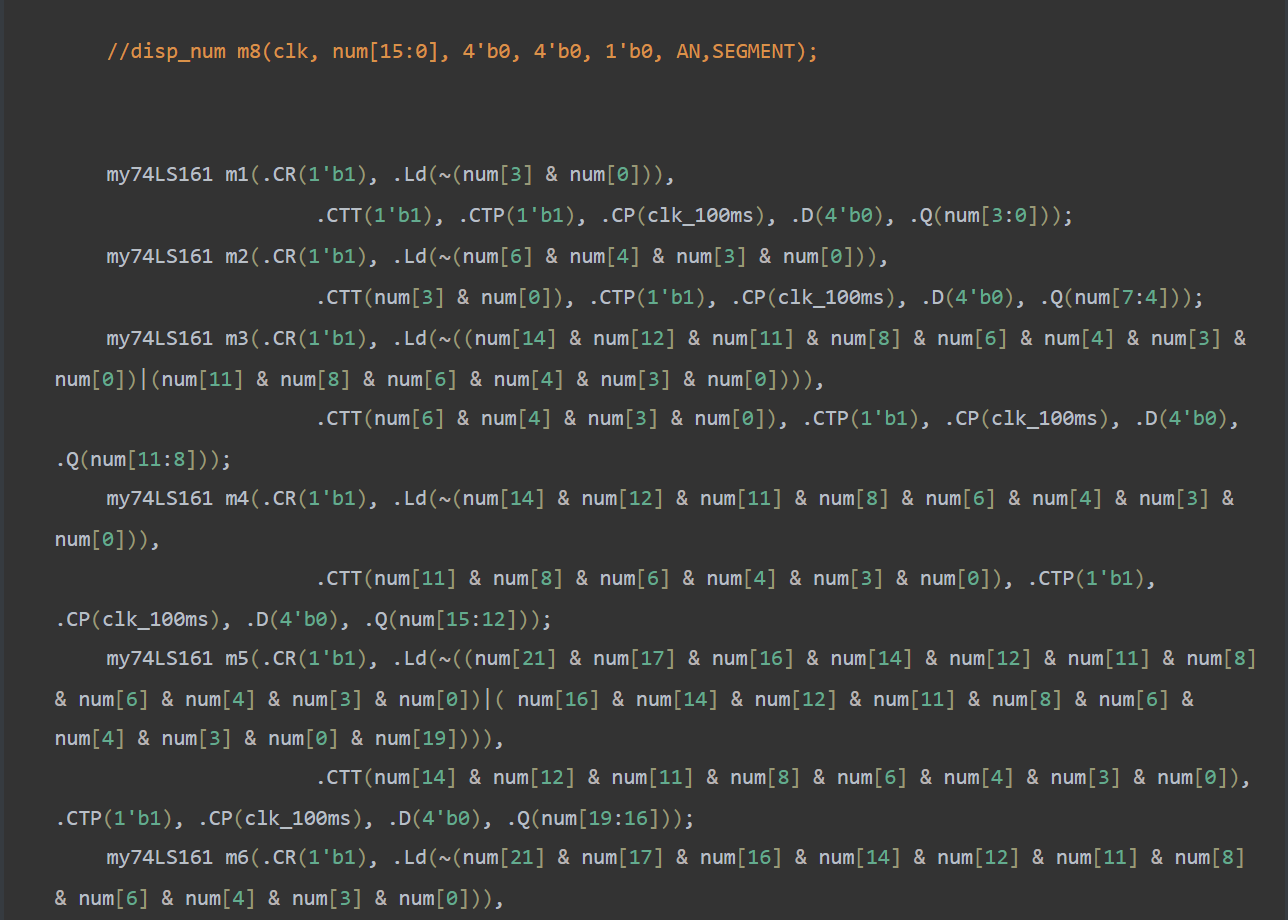
用结构化描述设计

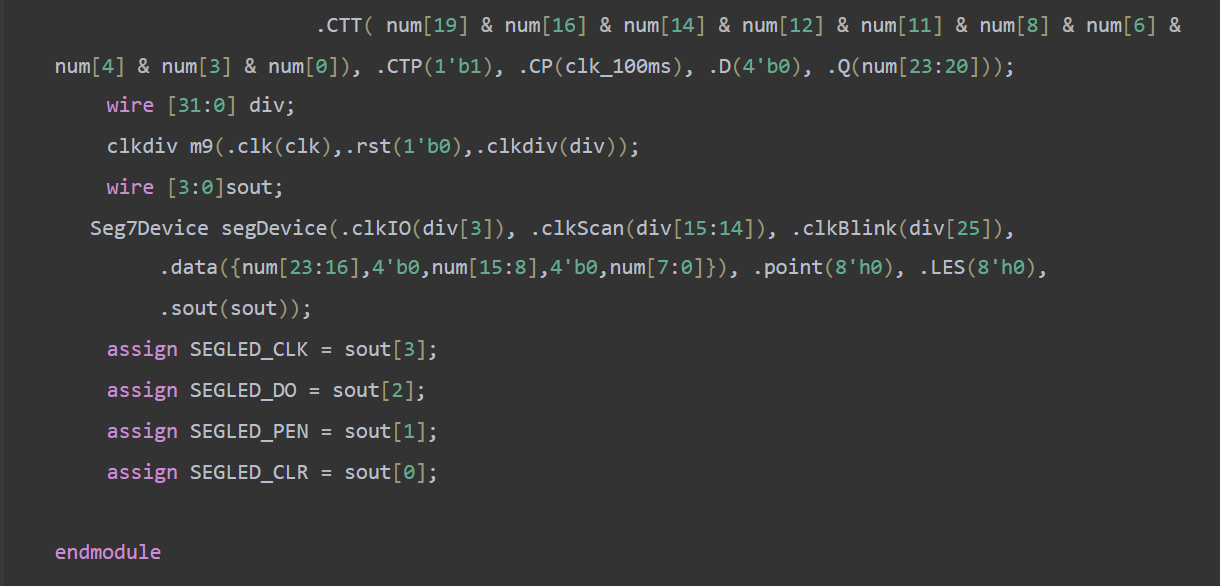
调用My74LS161

调用分频模块，用100ms作为分的驱动时钟

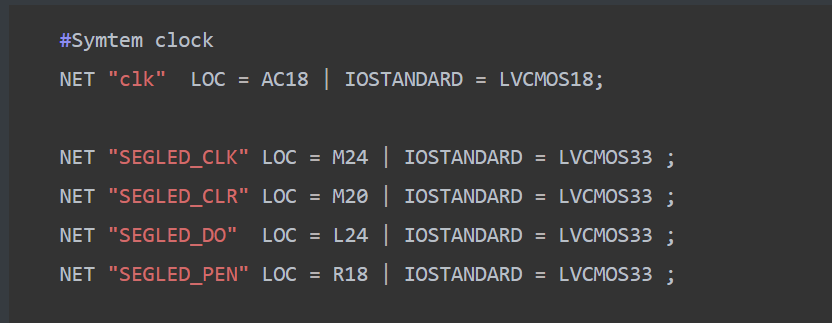
调用显示模块







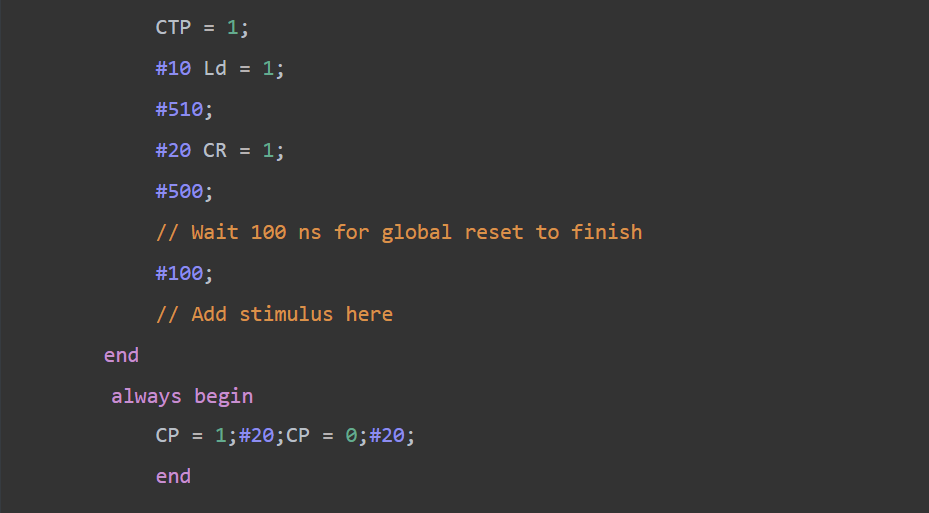
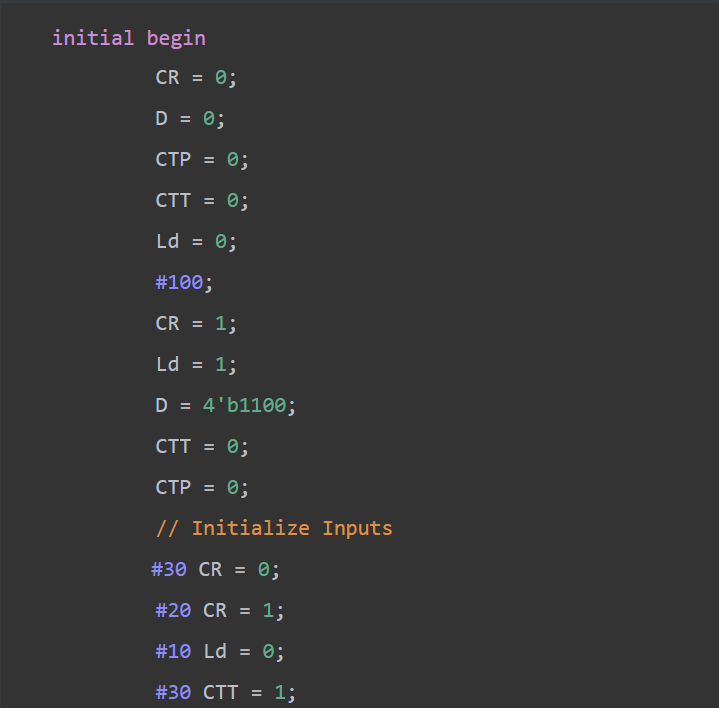
top代码

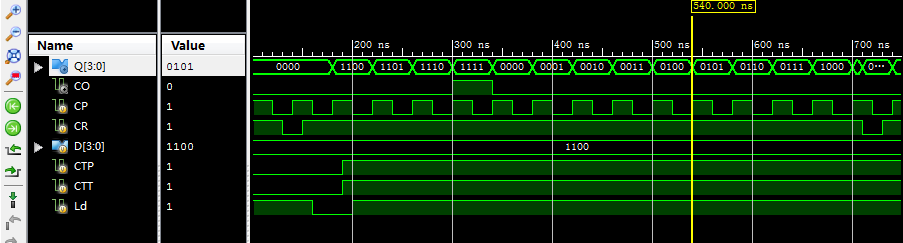


UCF定义

# 五、实验结果与分析

5.1 74LS161仿真

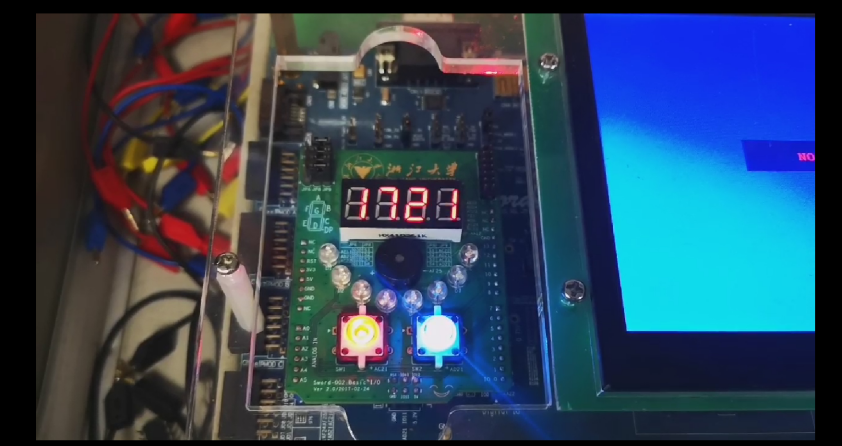




仿真图

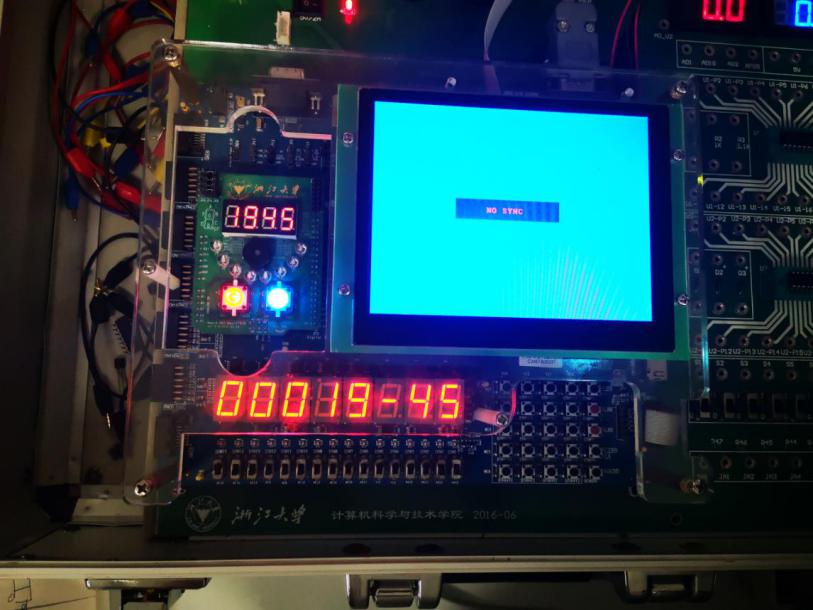
基于74LS161设计时钟应用

结果图：

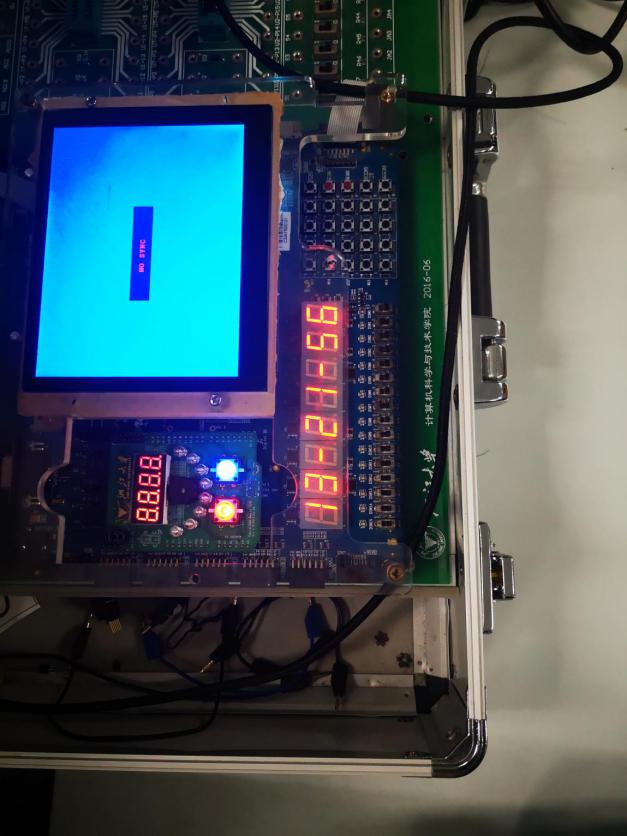


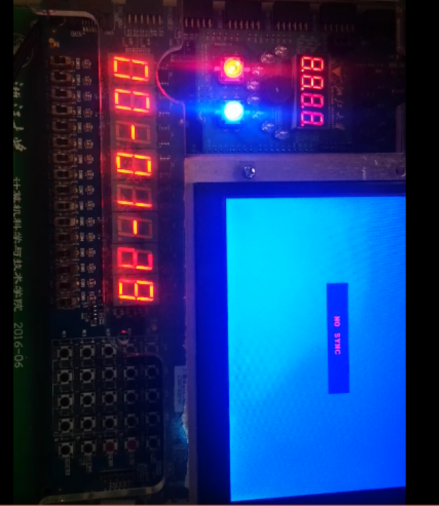
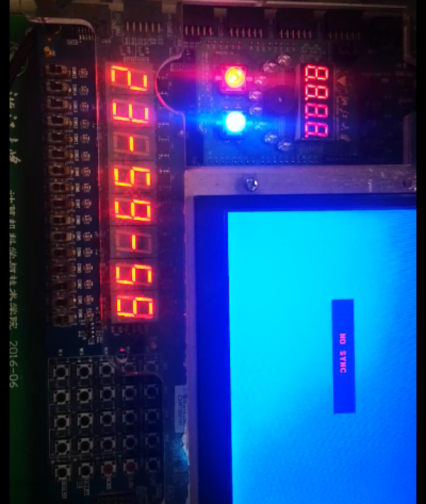
最初版





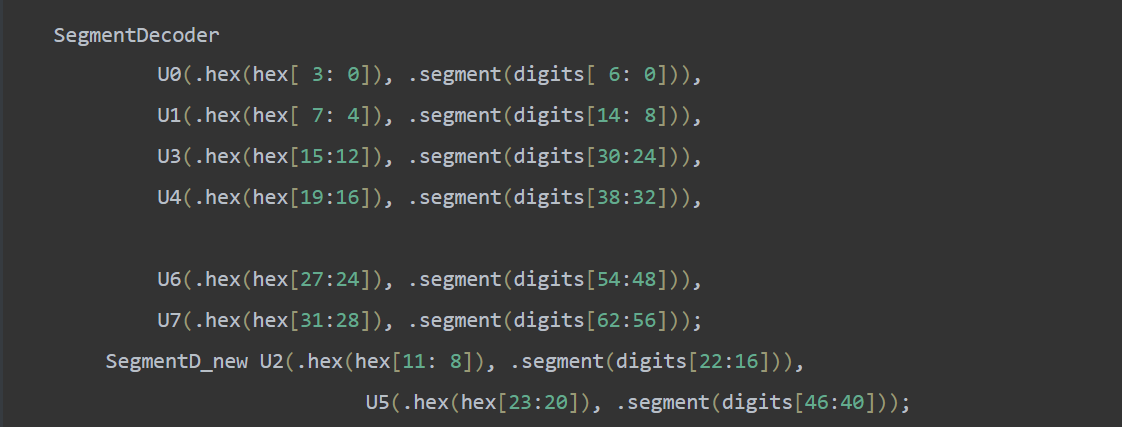
加分版-1



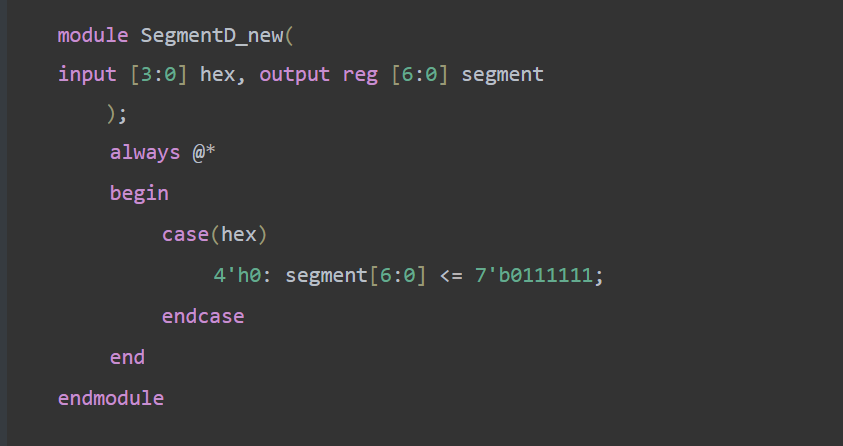


加分版-2

加分版实现方法，将decoder改写



新增SegmentD\_new器件



其定义如下，

因此可以在对应位置显示两个横杠

# 六、讨论、心得

这一次实验基础部分较为容易，由于课下对VERILOG语言资料进行了很长时间的整理，因此上手起来非常容易，但在上板子的时候还是遇到了很多问题，比如在最初的24进制时遇到3就进位，比如第一位狂闪等等问题，最后梳理出正确的逻辑，再配上合适的框架，成功将其显示。

加分项难处理在要对decoder进行改写，最后成功找到对应位置，将其赋值为0时的decode进行了改写，从而实现了横杠的显示，但不足的是top代码过于冗长，应该想出更优秀的解决方法来避免重复书写。

# 实验14、移位寄存器设计与应用实验报告

姓名： 王祚滨 专业： 信息安全 学号： 3180104933

课程名称： 逻辑与计算机设计基础实验 同组学生姓名： 王国朝、赵卿云

指导老师： 洪奇军 实验地点： 浙江大学紫金港校区东四教学楼509实验日期：2019年 12 月 18 日

# 一、实验目的和要求

1.掌握支持并行输入的移位寄存器的工作原理

2.掌握支持并行输入的移位寄存器的设计方法

**二、实验内容和原理**

2.1实验内容：

任务1：设计8位带并行输入的右移移位寄存器

任务2：设计主板LED灯驱动模块

任务3：设计主板七段数码管驱动模块

2.2实验原理：

2.2.1 移位寄存器

每来一个时钟脉冲，寄存器中的数据按顺序向左或向右移动一位；

必须采用主从触发器或边沿触发器，不能采用锁存器；

数据移动方式：左移、右移、循环移位。

数据输入输出方式：

串行输入，串行输出

串行输入，并行输出

并行输入，串行输出

2.2.2 串行输入右移移位寄存器

使用D触发器构成串行输入的右移移位寄存器：



图一 右移移位寄存器

循环右移移位寄存器：



图二 循环右移移位寄存器

2.2.3 带并行输入的右移移位寄存器



图三 带并行输入的右移移位寄存器

2.2.4并行－串行转换器



图四 并行－串行转换器

# 三、主要仪器设备

# 1. 装有Xilinx ISE 14.7的计算机 1台

# 2. SWORD开发板

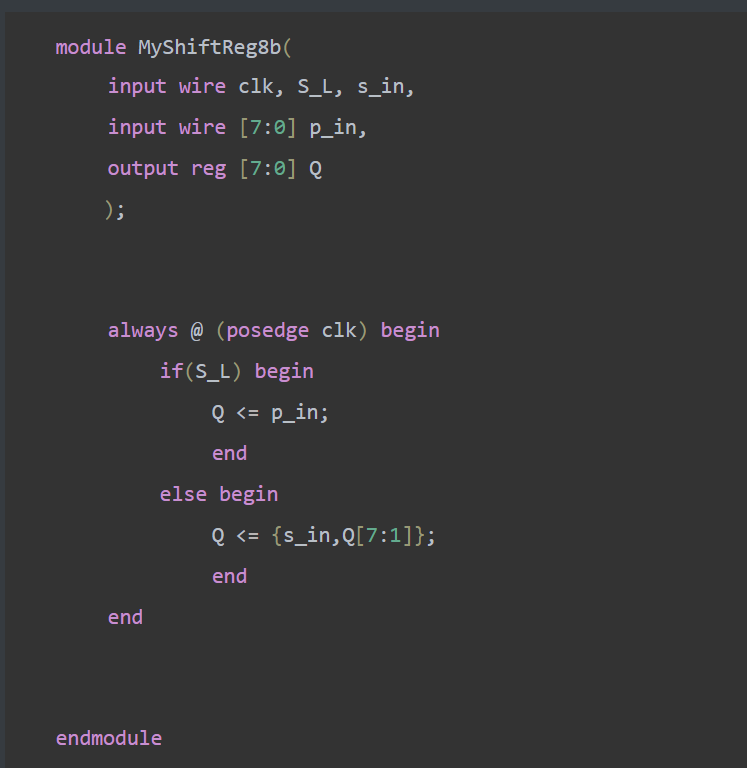
# 四、操作方法与实验步骤

**4.1 设计****8位带并行输入的右移移位寄存器**

新建工程，工程名称用ShfitReg8b\_dy。

Top Level Source Type用HDL

用结构化描述设计。



**4.2：****设计跑马灯应用**

新建工程，工程名称用MyMarquee。

Top Level Source Type用HDL

用结构化描述设计

调用ShfitReg8b

调用分频模块，用1s作为移位寄存器驱动时钟

调用显示模块

调用CreateNumber模块

用sw[0]和sw[1]作为regA和regB的按键自增控制输入

sw[2]=1，并行输入，将{RegA,RegB}赋给移位寄存器

sw[2]=0，串行/循环右移移位

sw[4]作为移位寄存器的模式选择：

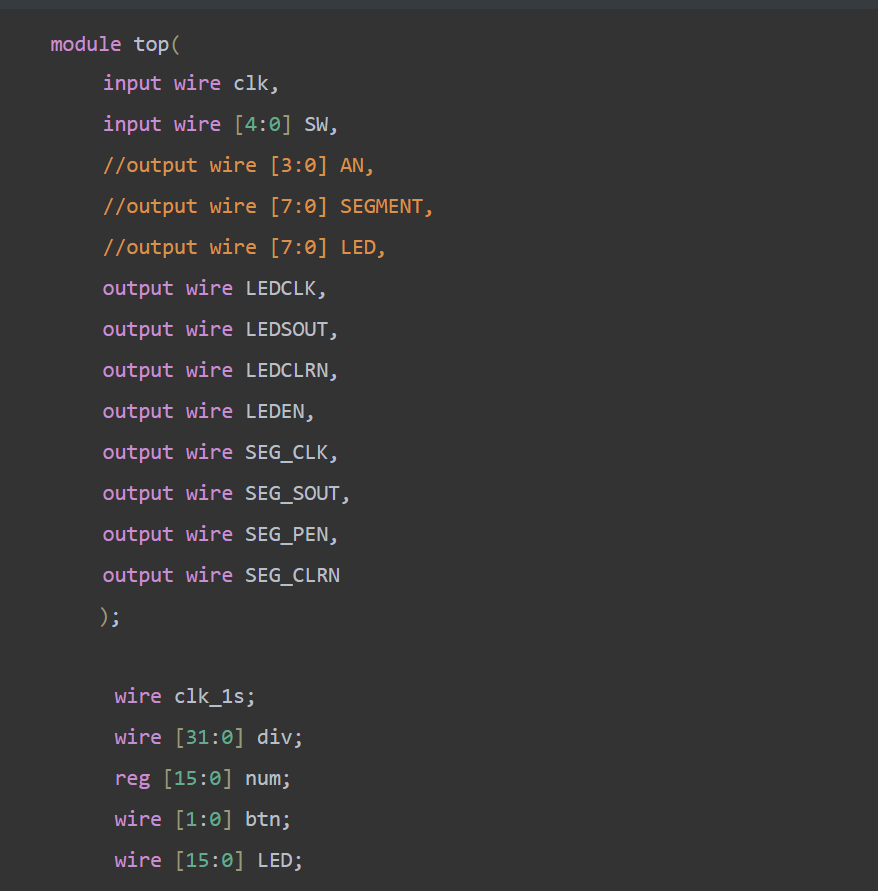
sw[4]=0，串行右移，串行输入值为sw[3]

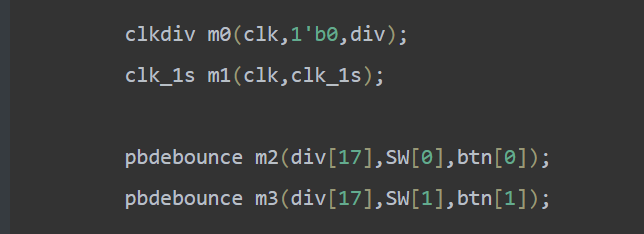
sw[4]=1，循环右移

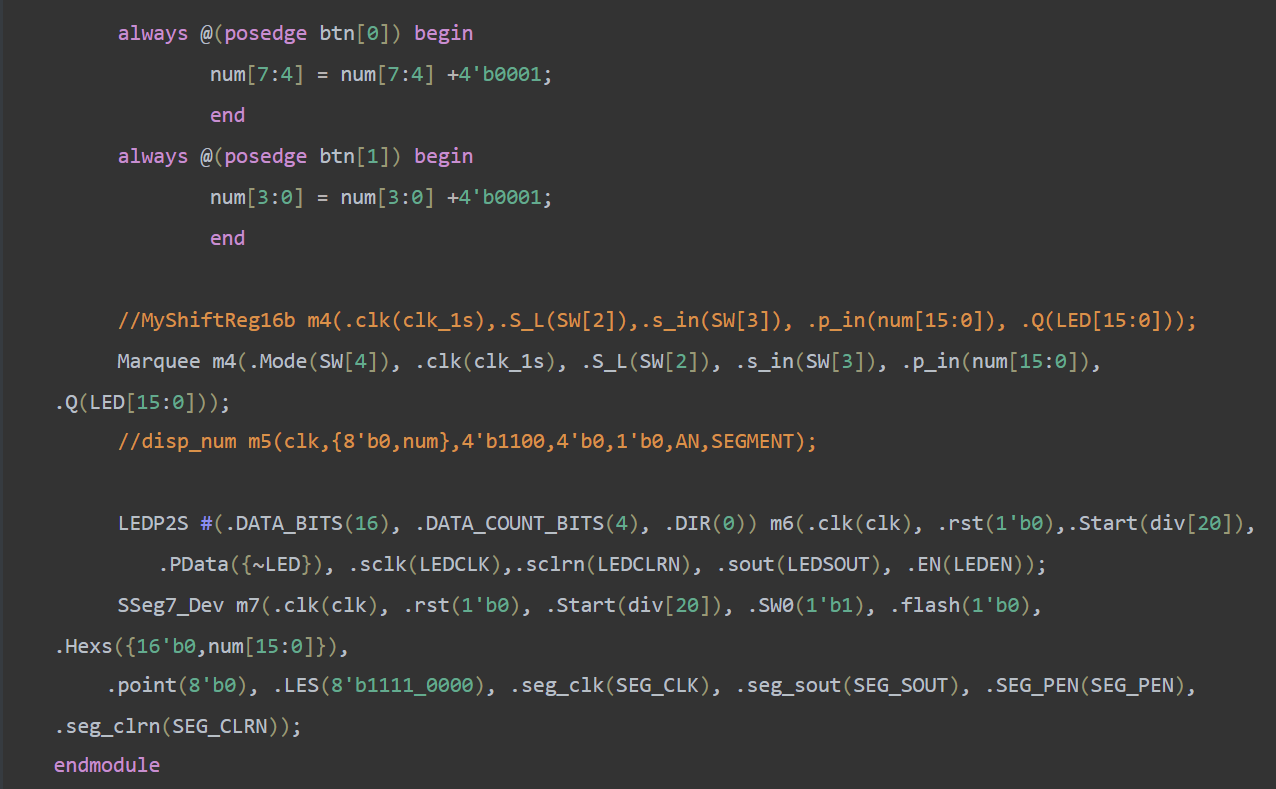
8位的移位寄存器的值用LED灯表示。

注：在这里由于改成了下方16位的七段数码管以及LED灯，故将原8位的表示注释掉

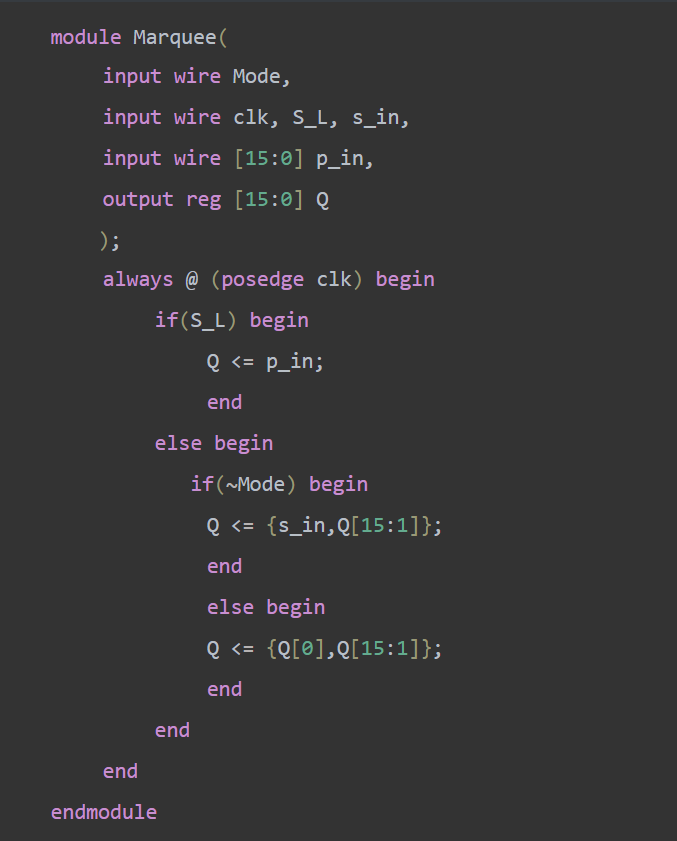
Verilog 代码：





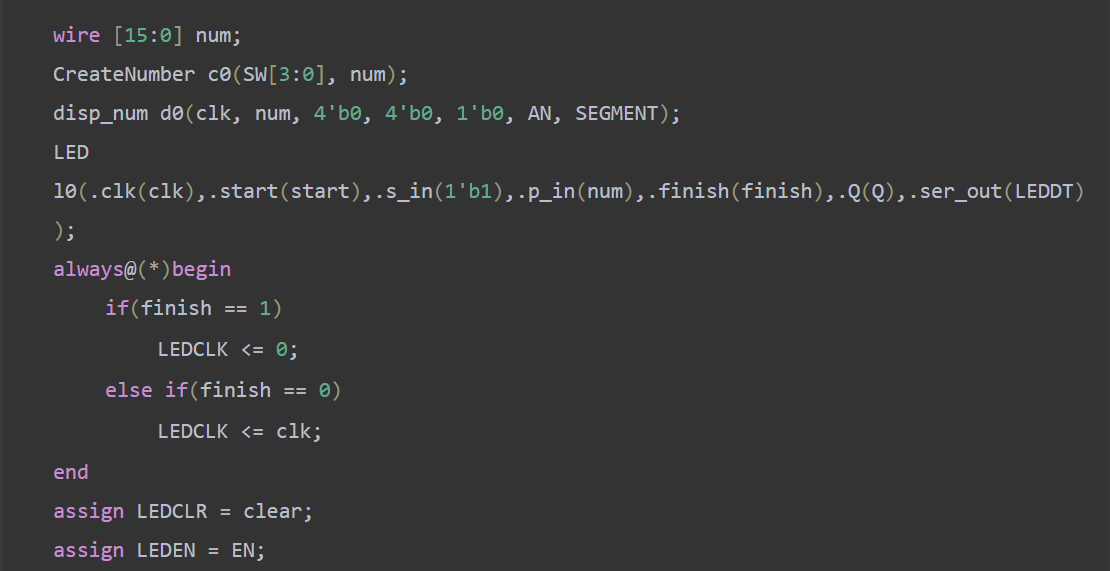


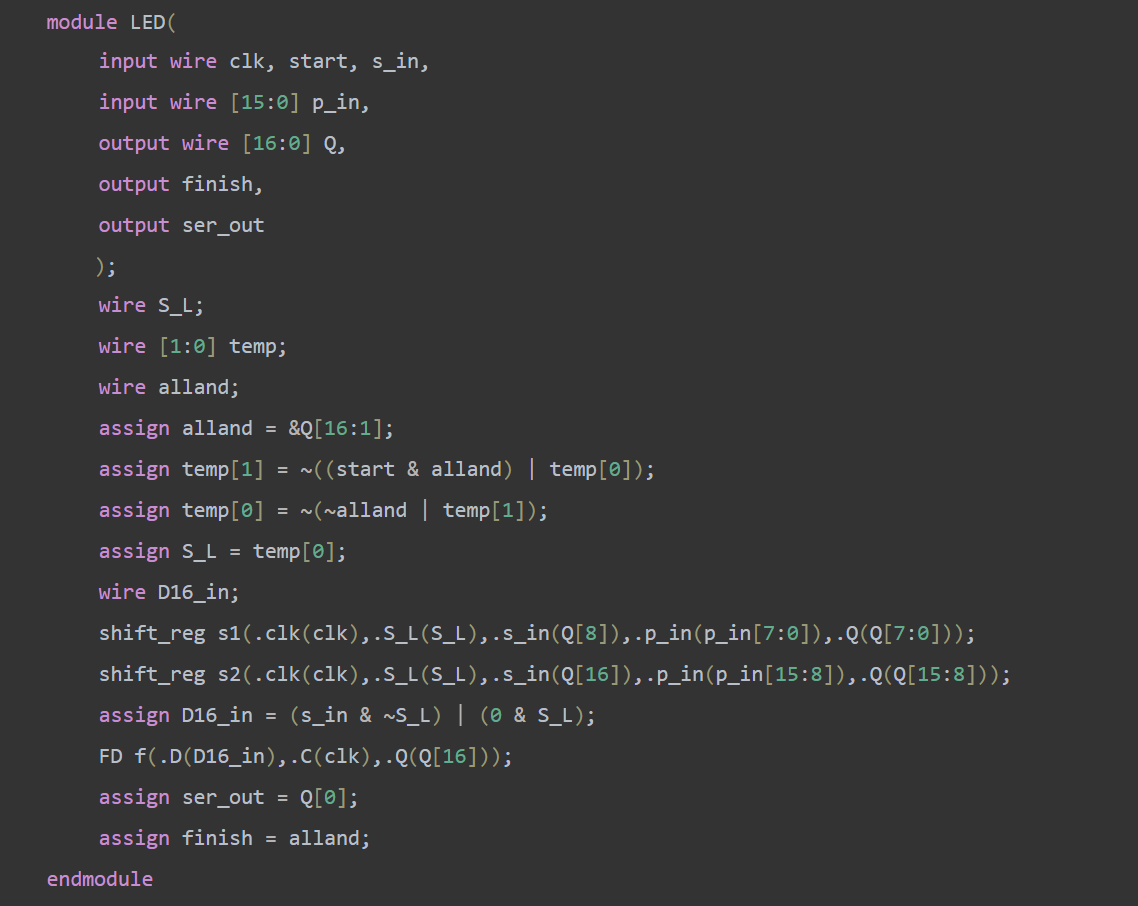
其中调用Marquee部分：



**LED驱动：**

verilog代码如下：

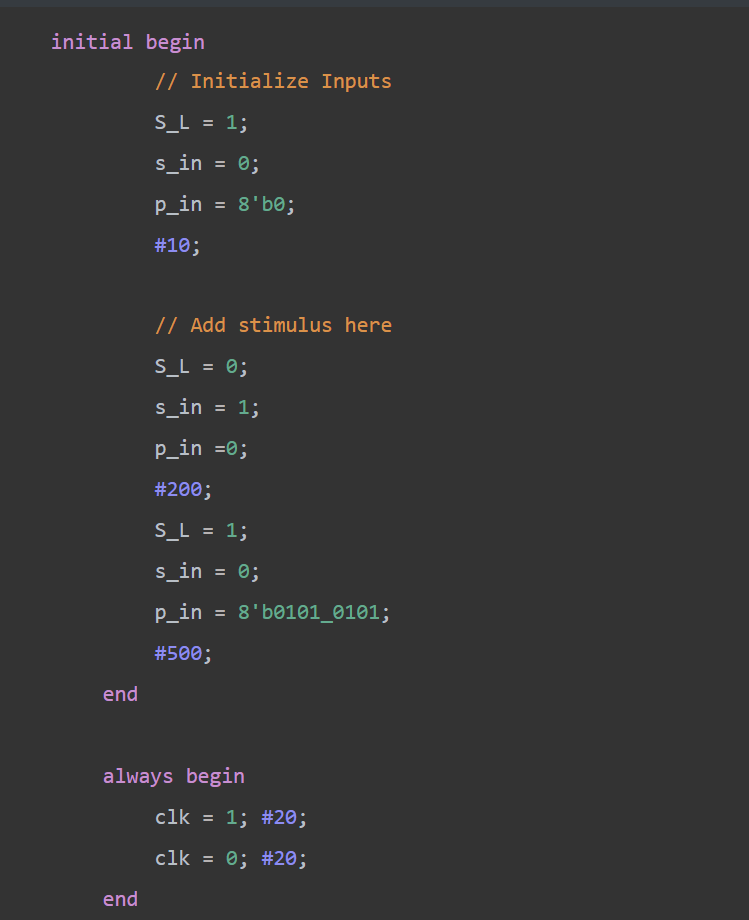


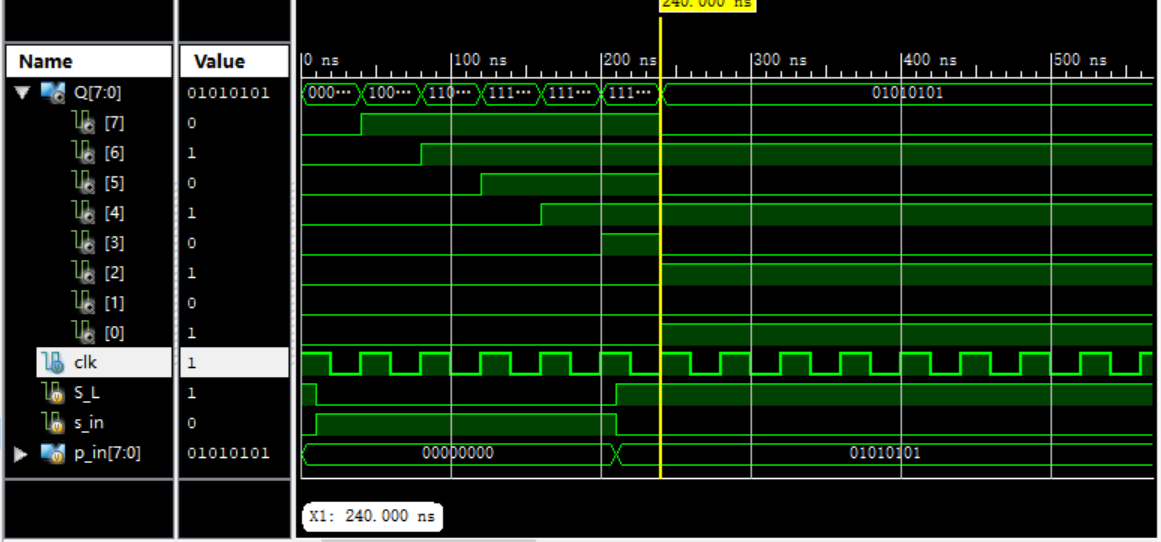


# 五、实验结果与分析

5.1 8位带并行输入的右移移位寄存器：

波形仿真





图七 波形仿真

前240ns是串行加载，每个正向加载“1”

时钟脉冲，每个寄存器在接下来的几个时钟周期右移一位。

240ns是并行加载。在时钟脉冲的上升沿加载“01010101”。

5.2 设计跑马灯应用





5.3LED驱动：



全暗



全亮

# 六、讨论、心得

作为最后一个实验，有着其一定的难度，在写shiftReg时，没有按照ppt中用and or等方式实现，而是采用更通俗易懂的代码，经过验证后也能产生正确的结果。另外，在实验过程中先是采用两位数和上面的数码管，LED灯，摸清逻辑后，更换到下面的数码管和led灯，实现了4位的跑马灯应用。此次实验，让我们有了更多的手段应对大作业，在写这份报告时大作业其实也已经在进行中了，确实用到了之前做过的组件，因此，我又重新温习了一下自己做过的实验，从中又吸取到了不同的感受。