实验一　Quartus II 集成开发环境及 EDA 开发流程

一、实验目的

1. 通过实验掌握 QuartusⅡ集成开发环境的安装；
2. 通过实验掌握 EDA 的开发流程；

二、实验仪器与器件

- 硬件：KONXIN 实验教学系统，PC 机；
- 软件：QuartusⅡ 12.1 开发套件，Windows XP/Windows 7。

三、实验内容

安装 QuartusⅡ，连接下载线，安装下载器驱动，创建工程，编写文件，编译综合，引脚锁定，运行程序，观察实验结果。

四、实验原理及步骤

实验原理：

在本实验平台主板上白色 JTAG PORT 口为程序下载口，通过下载器将程序下载到 FPGA 的 RAM 中。用四位拔码开关作为四位输入二进制数来控制数码管的显示，数码管显示的字符与这四位二进制数相对应。

实验步骤：

1、开机后，同时按 ctrl+shift+alt+F6，若有弹出程序框，则输入密码 123，选择解冻。

（注：**有些电脑重启后会自动还原，需要使用此方法解冻，否则下次开机还得重装软件。安装软件完毕后重新按** ctrl+shift+alt+F6，**选择冻结电脑。**）

2、安装 QuartusⅡ

将安装包拷贝到电脑上 D:\Software 路径下。

（**注意：所有安装包保存路径和软件安装路径都不能有中文。**）

打开\1-Quartus II 12p1_KX\q12.1\12.1_177_acds_windows 的 setup.exe 安装 QuartusⅡ。

打开\1-Quartus II 12p1_KX\q12.1 的 12.1_177_devices_cyclone_max_legacy_windows.exe 安装 devices 库。

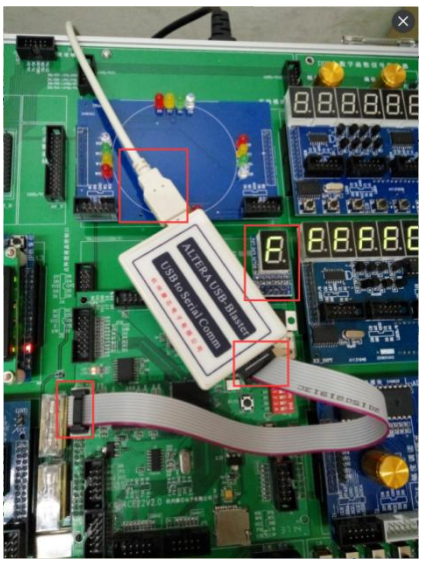
打开 1-Quartus II 12p1_KX\q12.1\12.1\Quartus_12.1_x86 破解器，根据 Crack.docx 文本步骤进行。

3、连接下载线，并安装驱动。

用十针排线将下载器与板上 JTAG 口（白色）连接，用 USB 线将下载器上的 ALTERA USB-Blaster 口与 PC 连接。

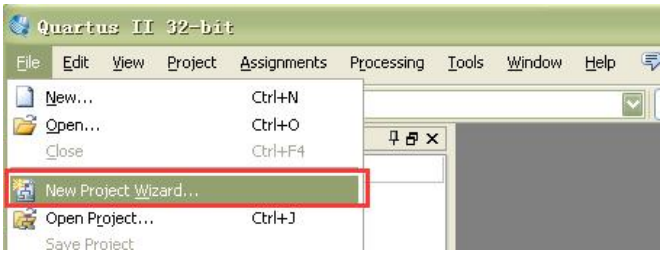
根据附件里文档“USB 下载器驱动路径的指定.doc”来安装下载器驱动。

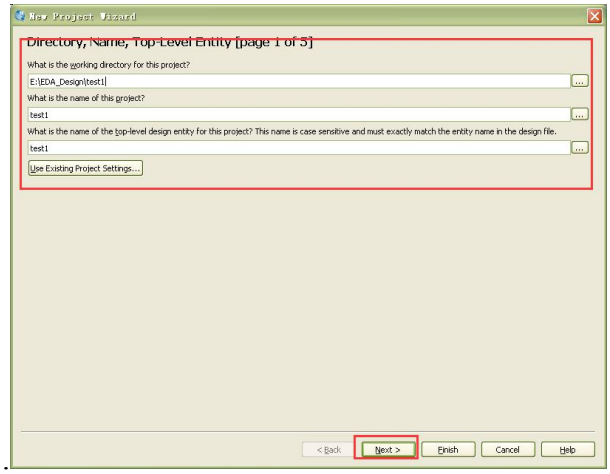
将数码管连接到 JP7 接口。

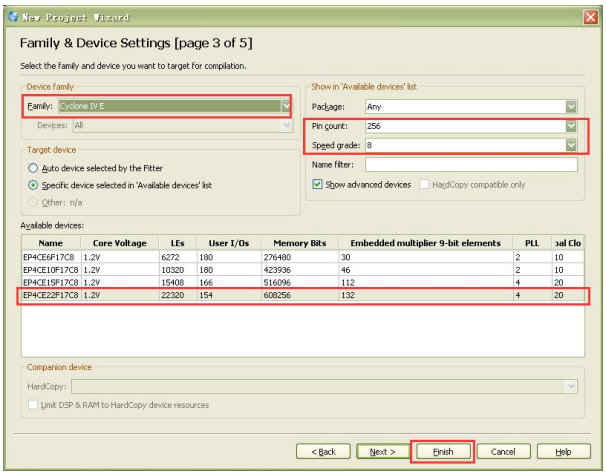


4、创建工程

打开 Quartus，选择 File-New Prioject Wizard，指定工程保存路径，给工程命名，选择芯片型号 EP4CE22F17C8。





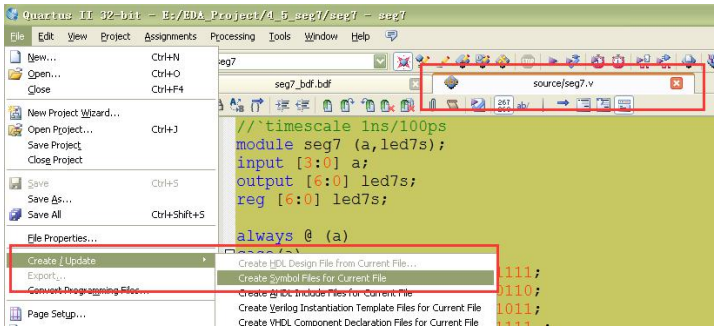


5、编写文件

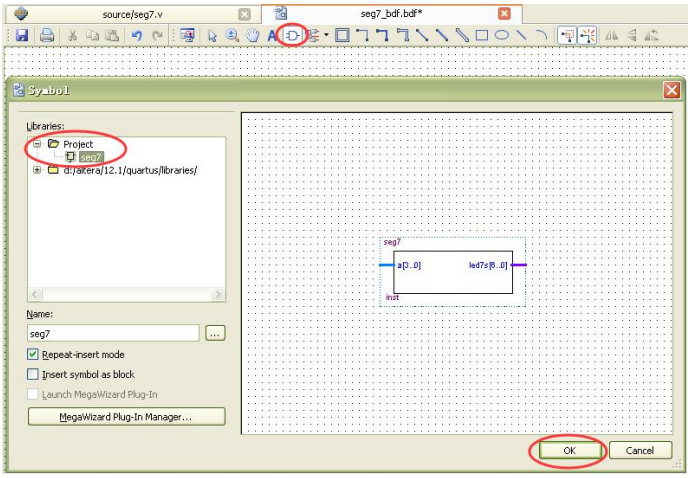
新建 Verilog 文件，File-New-Verilog HDL File 并将如下代码写入文件中：

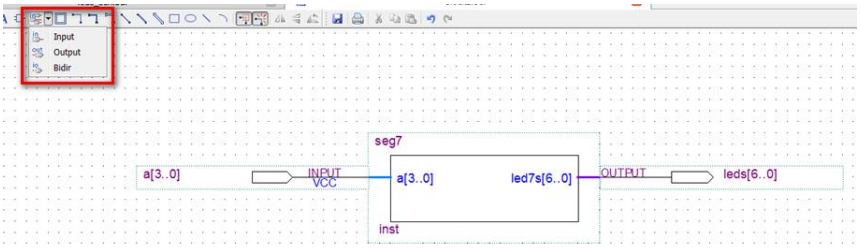
```
module seg7 (a,led7s);
input [3:0] a;
output [6:0] led7s;
reg [6:0] led7s;
always @ (a)
case(a)
4'b0000 : led7s <=  7'b0111111;
4'b0001 : led7s <=  7'b0000110;
4'b0010 : led7s <=  7'b1011011;
4'b0011 : led7s <=  7'b1001111 ;
4'b0100 : led7s <=  7'b1100110 ;
4'b0101 : led7s <=  7'b1101101 ;
4'b0110 : led7s <=  7'b1111101 ;
4'b0111 : led7s <=  7'b0000111 ;
4'b1000 : led7s <=  7'b1111111 ;
4'b1001 : led7s <=  7'b1101111 ;
4'b1010 : led7s <=  7'b1110111 ;
4'b1011 : led7s <=  7'b1111100 ;
4'b1100 : led7s <=  7'b0111001 ;
4'b1101 : led7s <=  7'b1011110 ;
4'b1110 : led7s <=  7'b1111001 ;
4'b1111 : led7s <=  7'b1110001 ;
endcase
endmodule
```

保存文件为 led7s.v，并将该文件生成为一个原件符号，File-Create/Update-Create Symbol Files for Currnet File：

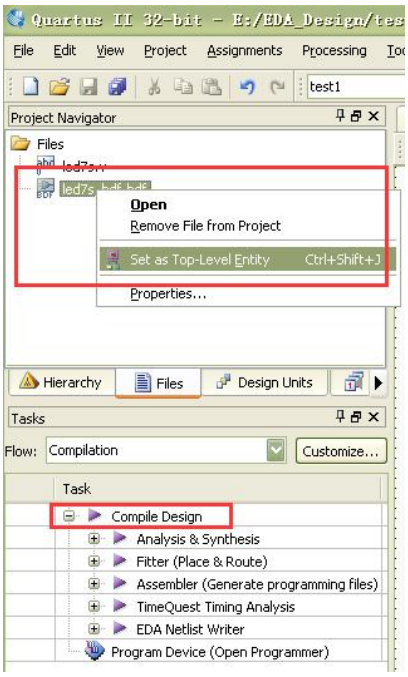


新建原理图文件 New-Block Diagrm/Schematic File，照下图完成原理图绘制，并命名为 seg7_bdf.bdf：(bdf 文件名不要与内部模块名 seg7 相同！)



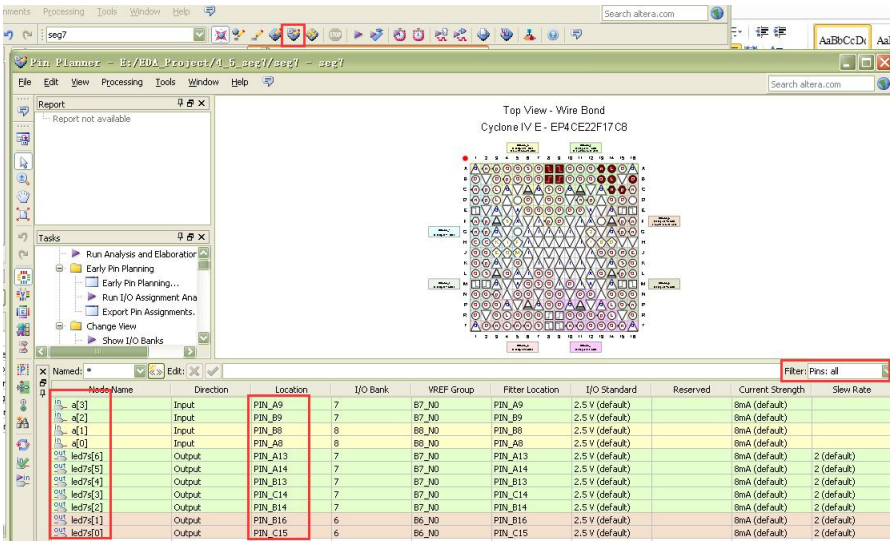


将原理图文件设置为 TOP 层，编译综合文件（Processing-StarCompilation）：



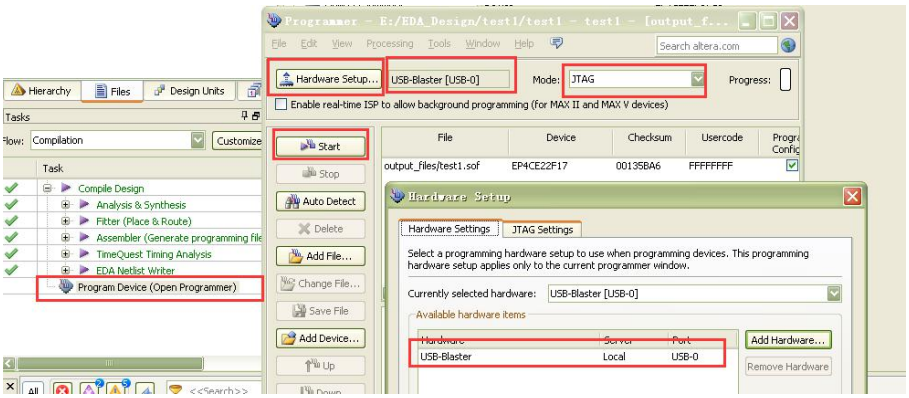
6、引脚锁定

打开 Assignments-Pin Planner，照下图将变量与管脚对应：



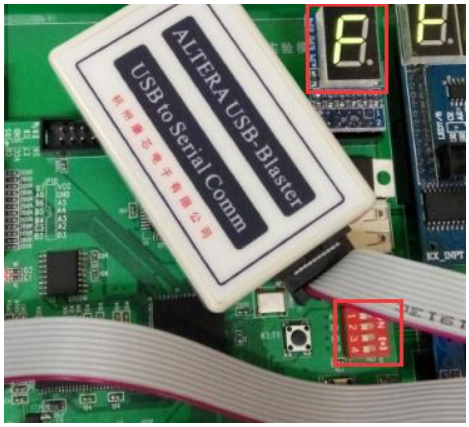
7、下载程序

重新编译后，打开 Tools-Programer，下载程序：



8、观察实验结果

拨动 4 个拔码开关，可以看到数码管显示对应的数字。



五、实验报告

1、用截屏或拍照的方式记录主要实验环节中观察到的现象；

2、写出实验过程中出现的问题及解决方法；

3、附加练习

附加练习 1：用两个数码管（接 JP7 和 JP8）同时显示相同的数字，效果如图“两个数码管同时显示.jpg”所示。(提示：数码管引脚见：“核心板布局.jpg”)；

附加练习 2：将 4 位拔码开关的低两位（二进制）加上高两位（二进制）的和，显示在数码管（接 JP7）上。例如：4 位拔码开关输入为 4'b0110，原来的显示结果为“6”，更改后的结果应为：2'b01 + 2'b10 = “3”。

实验二 Verilog 基本逻辑电路设计

一、实验目的

3. 通过实验掌握 Verilog 基本时序逻辑电路的设计方法；
4. 学习时钟分频电路的设计方法。

二、实验仪器与器件

- 硬件：KONXIN 实验教学系统，PC 机；
- 软件：Quartus II 12.1 开发套件，Windows XP/Windows 7。

三、实验内容

完成一个具有异步复位、同步计数使能和可预置数据的十进制计数器，并将计数结果实时显示于 7 段数码管，同时在计数值为“9”期间，点亮一个 LED 灯。在此基础上，完成具有扩展功能的附加实验。

四、实验原理及步骤

实验原理：

在输入时钟的上升沿计数，时钟、使能、预置控制信号由扩展板引入，复位键、7 段数码管和 LED 灯位于核心板。

实验步骤：

9、创建工程，并编写 Verilog 程序：

```
module CNT10 (clk, rst, en, load, cout, data, led7s, led0);
  input clk, en, rst, load;
  input [3:0] data;
  output cout;
  output [6:0] led7s;
  output led0;
  reg [3:0] dout;
  reg cout;
  reg [6:0] led7s;

  always @(posedge clk or negedge rst)
    begin
      if (!rst) dout <= 0;
      else if (en)
            begin
              if (!load) dout <= data;
              else if (dout < 9) dout <= dout + 1;
                  else dout <= 4'b0000;
            end
    end

  always @(dout)
    if (dout == 4'h9) cout = 1'b1;
    else cout = 1'b0;
```

```
  always @ (dout)
    begin
      case(dout)
        4'b0000 : led7s <=  7'b0111111;
        4'b0001 : led7s <=  7'b0000110;
        4'b0010 : led7s <=  7'b1011011;
        4'b0011 : led7s <=  7'b1001111;
        4'b0100 : led7s <=  7'b1100110;
        4'b0101 : led7s <=  7'b1101101;
        4'b0110 : led7s <=  7'b1111101;
        4'b0111 : led7s <=  7'b0000111;
        4'b1000 : led7s <=  7'b1111111;
        4'b1001 : led7s <=  7'b1101111;
        4'b1010 : led7s <=  7'b1110111;
        4'b1011 : led7s <=  7'b1111100;
        4'b1100 : led7s <=  7'b0111001;
        4'b1101 : led7s <=  7'b1011110;
        4'b1110 : led7s <=  7'b1111001;
        4'b1111 : led7s <=  7'b1110001;
      endcase
    end
endmodule
```

10、 连线及 FPGA 管脚分配

（1）将实验箱左下角“标准时钟信号源”模块的 1Hz 时钟源飞线到核心板 JP5 插座的 E9（clk）上；

（2）将实验箱右下角“高低电平输出控制开关”模块 J7 插座的 L7 和 L8 分别飞线到核心板 JP6 的 R16(en)和 N16(load)上。

（3）K1：R9 作为复位信号（rst）；

（4）数码管接于 JP7；LED：G2 作为 LED 指示（cout）。

管脚分配如下图所示：

clk	Input	PIN_E9
cout	Output	PIN_G2
data[3]	Input	PIN_A9
data[2]	Input	PIN_B9
data[1]	Input	PIN_B8
data[0]	Input	PIN_A8
en	Input	PIN_N16
led7s[6]	Output	PIN_A13
led7s[5]	Output	PIN_A14
led7s[4]	Output	PIN_B13
led7s[3]	Output	PIN_C14
led7s[2]	Output	PIN_B14
led7s[1]	Output	PIN_B16
led7s[0]	Output	PIN_C15
load	Input	PIN_R16
rst	Input	PIN_R9

11、　　下载程序，观察实验结果。

可以看到数码管显示的数字由 0 到 9 循环增加；分别按下 K1 键、L7 键、L8 键并拔动拔码开关，观察实验现象。

五、实验报告

1、用截屏或拍照的方式记录实验环节中观察到的现象，写出实验过程中出现的问题及解决方法。

2、附加练习

附加练习 1：为本工程增加一个置“9”开关（“高低电平输出控制开关”模块 J7 插座的 L6），使得在 rst 和 load 无效，且 en 有效的情况下，拔动此开关直接将数码管显示的数值设置为 9。

附加练习 2：将时钟信号改接到“标准时钟信号源”模块的 8Hz 时钟源，观察实验现象；设计一个 8 分频的时钟分频电路，对信号源进行分频，使得数码管上计数的速度恢复到原来（1Hz 时）的水平。

实验三　ModelSim 功能仿真

一、实验目的

5. 掌握测试文件的编写及 ModelSim 功能仿真方法；
6. 深入理解 Verilog 模块化设计思想。

二、实验仪器与器件

- 硬件：KONXIN 实验教学系统，PC 机；
- 软件：QuartusⅡ 12.1 开发套件，Windows XP/Windows 7。

三、实验内容

ModelSim 中创建工程，编译测试文件并添加到工程中，运行仿真，观察仿真波形。

四、实验原理及步骤

实验原理：

本实验通过仿真二分频文件，对输入时钟进行二分频（例如：将 20MHz 时钟信号分频为 10MHz 波形信号），通过观察仿真出来的波形来检验是否分频成功。

实验步骤：

1、创建 Modelsim 工程

打开 ModelSim，选择 File>New>Project，填写 Project Name、Project Location，

其余默认。

2、编写二分频文件及其测试文件

选择 Create New File，给文件命名 clk_2.v，Add file as type 选择 Verilog。
编写二分频文件

```
module two_division(rst, clk, clk_2);
  input rst;
  input clk;
  output reg clk_2;

  always@(posedge clk or negedge rst)
    begin
      if (~rst) clk_2 = 0;
      else clk_2 = ~clk_2;
    end
endmodule
```

编写激励文件 clk_2_top.v（File | New |Source | Verilog）

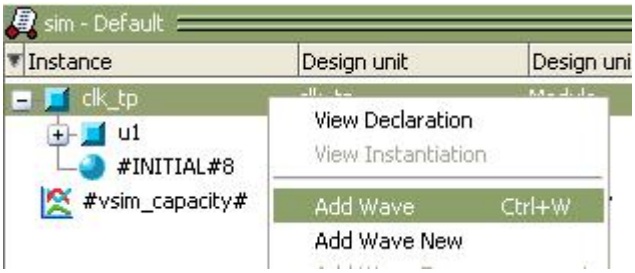
```
`timescale 10ns/1ns
module clk_tp;

integer i;
reg rst_t, clk_t;
wire clk_2_t;


initial
begin
  clk_t = 0; rst_t = 1;

   #10 rst_t = 0;
   #10 rst_t = 1;

//  for (i = 0; i < 256; i = i + 1) #10 clk_t = ~clk_t;
  forever #10 clk_t = ~clk_t;
end

initial  #1000 $stop;

two_division u1(
  .rst(rst_t),
  .clk(clk_t),
  .clk_2(clk_2_t));

endmodule
```

注意：仿真文件和被测文件中所有 reg 型变量必须给出明确的初值。

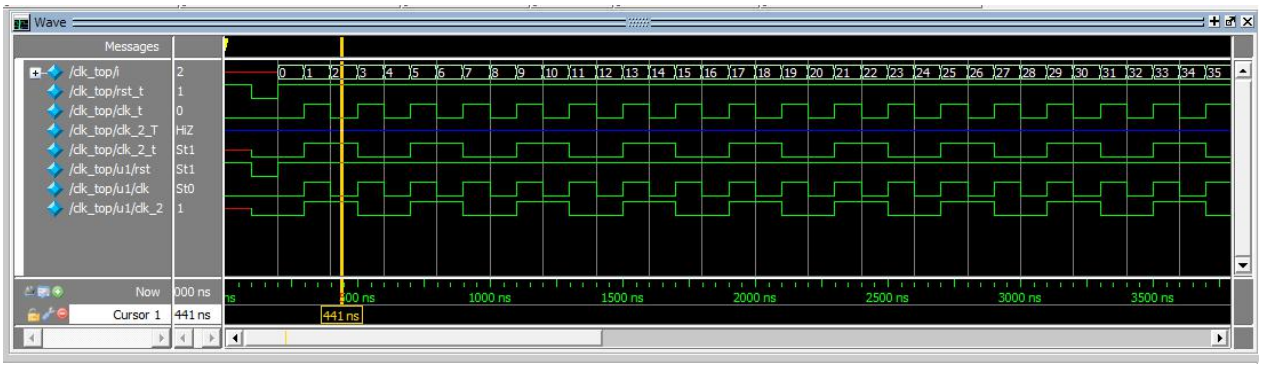
编译所有文件 Compile All，Status 显示对号时则编译成功。

3、进行仿真

选择 Simulate，在 work 目录下选择顶层模块 clk_top，去掉勾选的 Enable optimization，然后如下图方式添加波形。



4、观察结果



可见分频后的信号频率为原来的 1/2。同时观察变量 i 及其它信号的变化情况。

五、实验报告

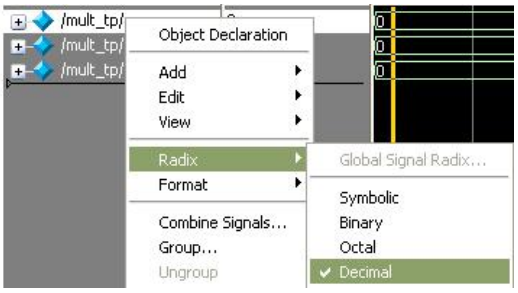
1、用截屏或拍照的方式记录主要实验环节中观察到的现象；

2、写出实验过程中出现的问题及解决方法；

六、附加练习

1、编写两个数 a、b 的乘法运算，结果为 c，其中 a、b 为小于 100 的数。仿真下面这几组数：a=2, b=3；a=99, b=99；a=12, b=13；a=2, b=33

（将显示数据格式设置为十进制便于观察结果）



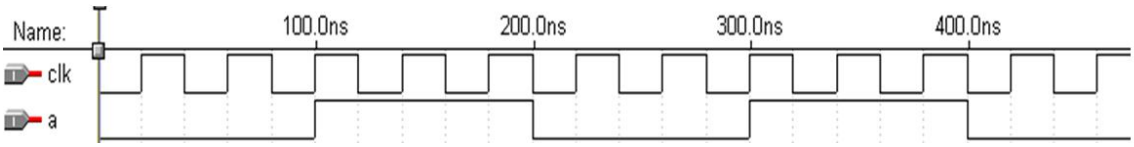
观察仿真波形看结果是否正确。

2、本课程课件第 5 讲中的讲到了阻塞赋值与非阻塞赋值。设计测试代码，仿真课件中的两个实例。

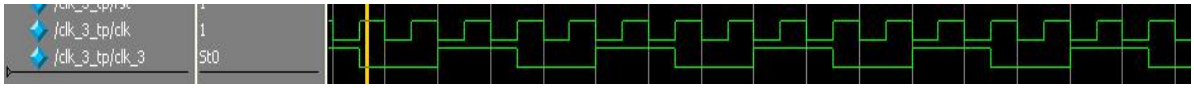
阻塞赋值与非阻塞赋值实例代码如下：

```
module    non_block(c,b,a,clk);
output    c, b;
input    clk, a;
reg    c, b;
always @(posedge clk)
    begin
            b <= a;
            c <= b;
    end
endmodule


module    block(c,b,a,clk);
output    c, b;
input    clk, a;
reg    c, b;
always @(posedge clk)
    begin
            b = a;
            c = b;
    end
endmodule
```

输入信号 a 变化情况如下图所示：



3、（选做）编写一个 3 分频文件，并编写其激励文件，仿真观察结果是否正确。仿真结果如下图：



【提示】实现占空比为 50%的奇数分频的方法：
两个 always 模块，分别在上升沿和下降沿触发计数，计数到（N-1）/2 输出时钟翻转，然后计数到（N-1）再次进行翻转，得到两个占空比非 50%的奇数 n 分频时钟。两个占空比非 50%的 n 分频时钟相或运算，得到占空比为 50%的奇数 n 分频时钟。

实验四　状态机控制串/并转换&数码静态显示电路设计

一、实验目的

7. 通过实验理解数码管扫描显示的原理；
8. 了解使用 IP 核生成电路模块（锁相环）的方法；
9. 掌握 Verilog 状态机编程方法；

- 掌握二进制数串/并或并/串转换的实现方法。

二、实验仪器与器件

- 硬件：KONXIN 实验教学系统，PC 机；
- 软件：QuartusⅡ 12.1 开发套件，Windows XP/Windows 7。

三、实验内容

- 1、编写 Verilog 状态机程序，实现 4 位数码扫描电路。
- 2、实现 8 位输入数据的并/串转换。

四、实验原理及步骤

实验原理：

图 1 所示的是 4 位数码扫描显示电路，其中每个数码管的 8 个段 h、g、f、e、d、c、b、a（h 是小数点）都分别连在一起，4 个数码管分别由 4 个选通信号 k1～k4 来选择。被选择的数码管显示数据，其余关闭。例如，在某一时刻，若选通信号 k3 为高电平，其余选通信号为低电平，这时仅 k3 对应的数码管显示来自段信号端的数据，而其他 3 个数码管呈现关闭状态。根据这种电路结构，如果使 4 个选通信号 k1～k4 轮流被选通，并在段信号输入口输入对应数码管要显示的数据，那么随着选通信号的扫变，就能实现 4 位数码管扫描显示。

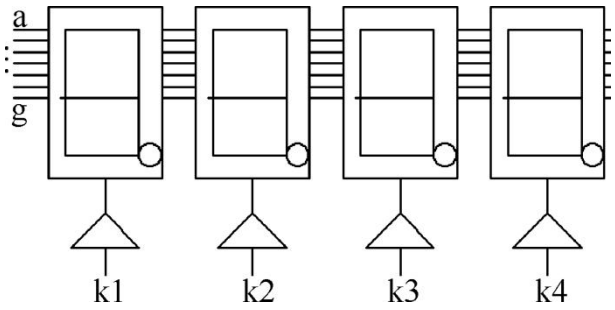


图 1　4 位数码扫描显示电路。

实验步骤：

12、　用 IP 核生成器 MegaWizard 产生锁相环文件 CLK_2K。

选择 Tools---MegaWizard Plug-In Manager，选择 Create a new custom megafunction variation，在 I/O 目录下选择 ALTPLL，右侧中间提示框补上输出文件名称 CLK_2K，选择参数使 C0 输出频率为 2KHZ。注意：输入时钟信号选用板载的 20MHz 晶振，连接于 FPGA 的 E1 管脚。

13、　编写分频文件，将 2KHZ 分为 1HZ(CLK2K_1HZ.v)。

```
//clk 2KHZ to 1HZ
module CLK2K_1HZ(clk_2k,clk_1hz);
input  clk_2k;
output clk_1hz;

reg clk_t=0;
reg [10:0] i=0;
always@(posedge clk_2k )
begin
if(i>=999) begin
 clk_t<=~clk_t;
 i<=0; end
else i=i+1; end
assign clk_1hz=clk_t;
endmodule
```

14、　编写主文件，数码管动态扫描显示（ScanLed.v）。

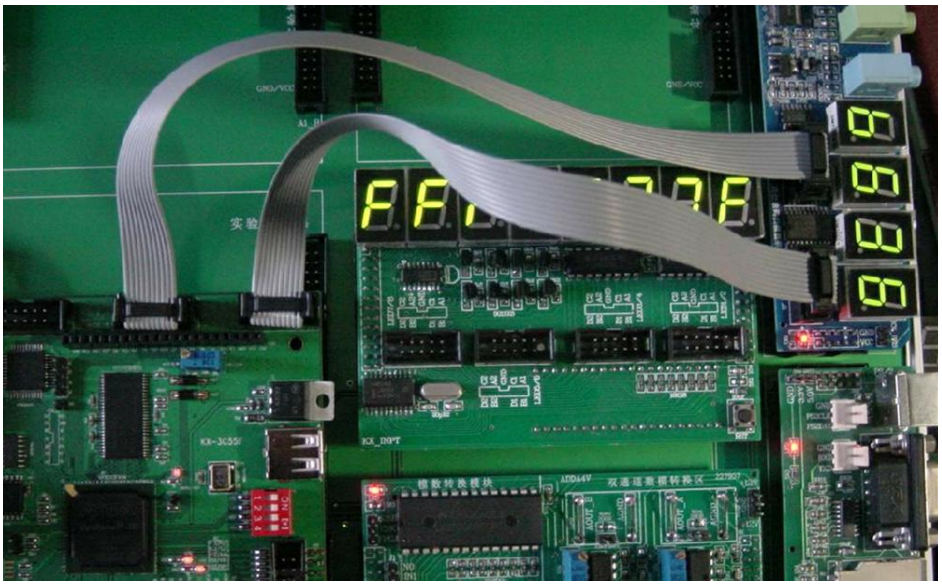
```
module   ScanLed(rst,clk,cs,led);
input rst,clk;
output   reg[3:0] cs;
output   reg[7:0] led;
reg   [3:0] a=1,b=2,c=3,d=4,y=0;
reg   [1:0] state=2'b00;
parameter   s0=2'b00,s1=2'b01,s2=2'b10,s3=2'b11;
//四个状态，每个状态只有一个数码管显示。
always@  (posedge clk or negedge rst)
begin if(!rst) state=0;
else  begin
case(state)
   s0 :  begin
      cs    <=4'b0001;
      y     <=b;
      state <=s1;    end
   s1 :  begin
      cs    <=4'b0010;
      y     <=c;
      state <=s2;    end
   s2 :  begin
      cs    <=4'b0100;
      y     <=d;
      state <=s3;    end
   s3 :  begin
      cs    <=4'b1000;
      y     <=a;
      state <=s0;    end
endcase
```

```
case(y)      //LED display char
0  :  led   <= 8'h3f;
1  :  led   <= 8'h06;
2  :  led   <= 8'h5b;
3  :  led   <= 8'h4f;
4  :  led   <= 8'h66;
5  :  led   <= 8'h6d;
6  :  led   <= 8'h7d;
7  :  led   <= 8'h07;
8  :  led   <= 8'h7f;
9  :  led   <= 8'h6f;
10 :  led   <= 8'h77;
11 :  led   <= 8'h7c;
12 :  led   <= 8'h39;
13 :  led   <= 8'h5e;
14 :  led   <= 8'h79;
15 :  led   <= 8'h71;
endcase
end   end

//调用锁相环、分频文件。
wire  c0,clk_1hz;
CLK_2K       u1(.inclk0(clk), //调用锁相环，将系统频率分为2KHZ
                .c0(c0));
CLK2K_1HZ    u2(.clk_2k(c0),//调用分频文件，分频为1HZ
                .clk_1hz(clk_1hz));
endmodule
```

15、　编译成功后，锁定引脚。

Node Name	Direction	Location
clk	Input	PIN_E1
cs[3]	Output	PIN_C9
cs[2]	Output	PIN_A10
cs[1]	Output	PIN_B11
cs[0]	Output	PIN_B12
led[7]	Output	PIN_B13
led[6]	Output	PIN_A13
led[5]	Output	PIN_C14
led[4]	Output	PIN_A14
led[3]	Output	PIN_B14
led[2]	Output	PIN_A15
led[1]	Output	PIN_B16
led[0]	Output	PIN_C15
rst	Input	PIN_R9

16、　按下图连接硬件电路，编译成功后下载，观看显示结果。



五、实验报告

a) 用截屏或拍照的方式记录主要实验环节中观察到的现象，写出实验过程中出现的问题及解决方法。

b) 附加练习

（1）用上述四个数码管实现电子表的功能，即前两位数码管显示分，后两位显示秒。提示：用 1HZ 作为秒的输入时钟，编写 60 进制计数器。

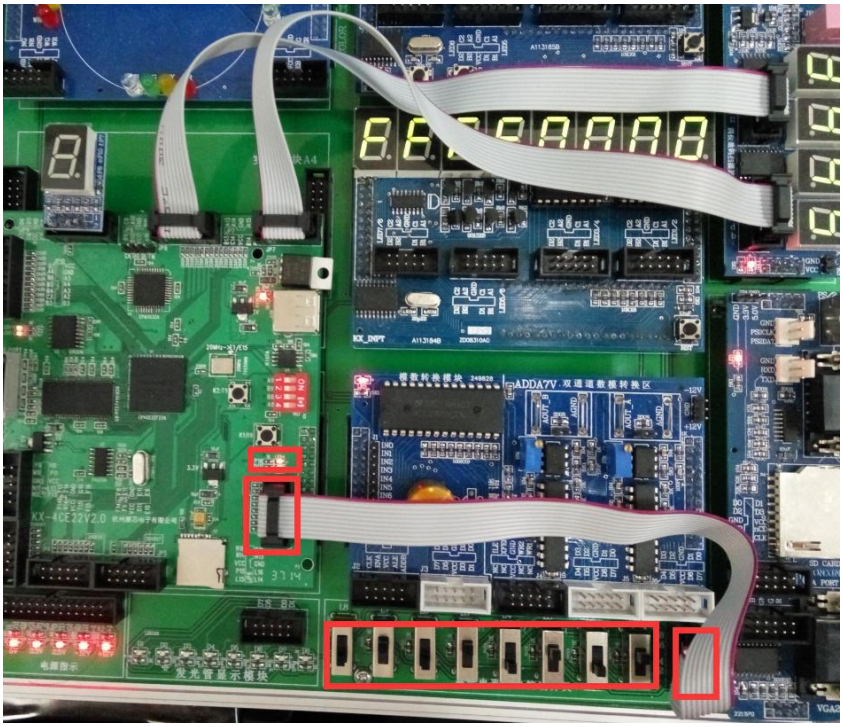
（2）8 位 2 进制数并/串转换。用八位拨码开关并行输入 8 位二进制数，按复位键后，串行输出这 8 位二进制数来控制 LED，输出 1 时亮，输出 0 时灭，并将这 8 个数中 0 和 1 的个数分别显示在数码管上。

提示：如下图中红圈所示，用八位拨码开关作为输入，LED（FPGA 对应引脚 P16）作为输出显示。按键开关 K2(FPGA 对应引脚 T9)作为并串转换启动信号，K2 按下为低电平复位。

参考代码：

```
//并行输入串行输出
  module      p2s(di, do, clk, ld);
      input     [7:0]di;
      input       clk,ld;
      output    reg do;
      reg [6:0] t;
     always@(posedge clk or negedge ld)
     begin
```

```
            if(~ld)    {t, do} <= di;
            else       {t, do} <= {1'b0, t};
        end
    endmodule
```



与此相对应，阅读并理解以下串并转换程序：

```
module Serial_Par_Con(clk, reset, enable, din, dout)
   input   clk,   reset,   enable,   din;
   output [3:0] dout;
   always @ (posedge clk)
     begin
        if(reset)       dout <= 4’h0;
        else   if(enable)  dout <= { dout, din};
                 else dout <= dout;
     end
endmodule
```

实验五　字符液晶显示器驱动控制电路设计

一、实验目的

10. 通过实验了解 LCD 的显示原理；

11. 掌握通过状态机驱动字符型 LCD 的方法。

二、实验仪器与器件

- 硬件：KONXIN 实验教学系统，PC 机；
- 软件：QuartusⅡ 12.1 开发套件，Windows XP/Windows 7。

三、实验内容

编写 Verilog 程序，用状态机实现对 2004 字符型液晶的驱动。

四、实验原理：

本试验使用的 2004 型 LCD（Liquid Crystal Display）是字符型液晶，可以显示 4 行，每行可以显示 20 个字符。

LCD 模块中内置了 CGROM、CGRAM、DDRAM 三个存储器：

CGROM 中存储了标准字符的字符代码（字模），是液晶屏出厂时固化在控制芯片中的，用户不能改变其中的存储内容，只能读取调用，包含有标准的 ASCII 码、日文字符和希腊文字符（不支持标准中文字符）。

CGRAM 是控制芯片留给用户，用以存储用户自己设计的字符代码的存储器。

DDRAM 是与屏幕显示区域有对应关系的存储器。用户在 DDRAM 指定地址中写入想要显示的字符所对应的字符代码，就会在 2004 型 LCD 对应的区域显示出该字符。DDRAM 地址分配(16 进制)如下：

第 1 行：0x80---93
第 2 行：0xC0---D3
第 3 行：0x94---A7
第 4 行：0xD4---E7

LCD 的接口信号：

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
VSS	VCC	VEE	RS	R/W	E	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7	LED+	LED-

其中：

RS：　　RS=0,指令代码输入；RS=1,数据输入。
R/W：　　RW=0,写数据；RW=1,读数据。
DB0-DB7：数据线。
E：　　使能信号，一般为 1。
VSS、VCC、LED+、LED-、VEE：电源及对比度控制。

下图为 LCD 的指令表和字符代码表：

指令	指令码										说明	指令周期
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		f_{osc}=250kHz
清屏	0	0	0	0	0	0	0	0	0	1	清除屏幕，置 AC 为 0，光标回位。	1.64ms
光标返回	0	0	0	0	0	0	0	0	1	*	DDRAM 地址为 0，显示回原位，DDRAM 内容不变.	1.64ms
设置输入方式	0	0	0	0	0	0	0	1	I/D	S	设置光标移动方向并指定显示是否移动。	40μs
显示开关	0	0	0	0	0	0	1	D	C	B	设置显示开或关 D、光标开关 C、光标所在字符闪烁 B.	40μs
移位	0	0	0	0	0	1	S/C	R/L	*	*	移动光标及整体显示，同时不改变 DDRAM 内容。	40μs
功能设置	0	0	0	0	1	DL	N	F	*	*	设置接口数据位数 DL、显示行数 L、字符字体 F。	40μs
CGRAM 地址设置	0	0	0	1	ACG						设置 CGRAM 地址。设置后发送接收数据。	40μs
DDRAM 地址设置	0	0	1	ADD							设置 DDRAM 地址。设置后发送接收数据。	40μs
忙标志/读地址计数器	0	1	BF	AC							读忙标志 BF 标志正在执行内部操作并读地址计数器内容。	0μs
CGRAM/DDRAM 数据写	1	0	写数据								从 CGRAM 或 DDRAM 写数据。	40μs
CGRAM/DDRAM 数据读	1	1	读数据								从 CGRAM 或 DDRAM 读数据。	40μs
	I/D=1：增量方式；I/D=0：减量方式 S=1：移位 S/C=1：显示移位；S/C=0：光标移位 R/L=1：右移；R/L=0：左移 DL=1：8 位；DL=0：4 位 N=1：2 行；N=0：1 行 F=1：5 x 10 字体　F=0：5 x 7 字体 BF=1：执行内部操作；BF=0 可接收指令										DDRAM：显示数据 RAM CGRAM：字符发生器 RAM ACG：CGRAM 地址 ADD：DDRAM 地址及光标地址 AC：地址计数器，用于 DDRAM 和 CGRAM	执行周期主频改变而改变。例如当 fcp 或 fosc=270KHZ 时：40μs x 250/270=37μs

High 4BIT / LOW 4BIT	MSB 0000	0010	0011	0100	0101	0110	0111
LSB xxxx0000	RAM (1)		0	@	P	`	p
xxxx0001	(2)	!	1	A	Q	a	q
xxxx0010	(3)	“	2	B	R	b	r
xxxx0011	(4)	#	3	C	S	c	s
xxxx0100	(5)	$	4	D	T	d	t
xxxx0101	(6)	%	5	E	U	e	u
xxxx0110	(7)	&	6	F	V	f	v
xxxx0111	(8)	’	7	G	W	g	w
xxxx1000	(1)	(	8	H	X	h	x
xxxx1001	(2)	)	9	I	Y	i	y
xxxx1010	(3)	*	:	J	Z	j	z
xxxx1011	(4)	+	;	K	[	k	{
xxxx1100	(5)	,	<	L	¥	l	|
xxxx1101	(6)	-	=	M	]	m	}
xxxx1110	(7)	.	>	N	^	n	→
xxxx1111	(8)	/	?	O	_	o	←

例如：若要在 LCD 屏在第一行第二个字符的位置显示一个字母 X，查表得 X 的 ASCII 码为 0x58。首先选中 DDRAM 中第一行第二个字符的位置地址，根据 DDRAM 地址分配和指令表可知：RS=0，RW=0，DB[7:0]=8’h81。然后给 DDRAM 写入想要显示字符的值，根据指令表可知 RS=1，RW=0，DB[7:0]=8’h58。这样，就可在第一行第二个字符的位置显示一个字符 X。

需要注意的是，LCD 在显示前先要进行初始化和显示模式的设置。本实验的初始化程序是：将屏幕清空、光标复位、设置显示模式为自动右移（可以连续显示字符，只设定一次显示位置）。

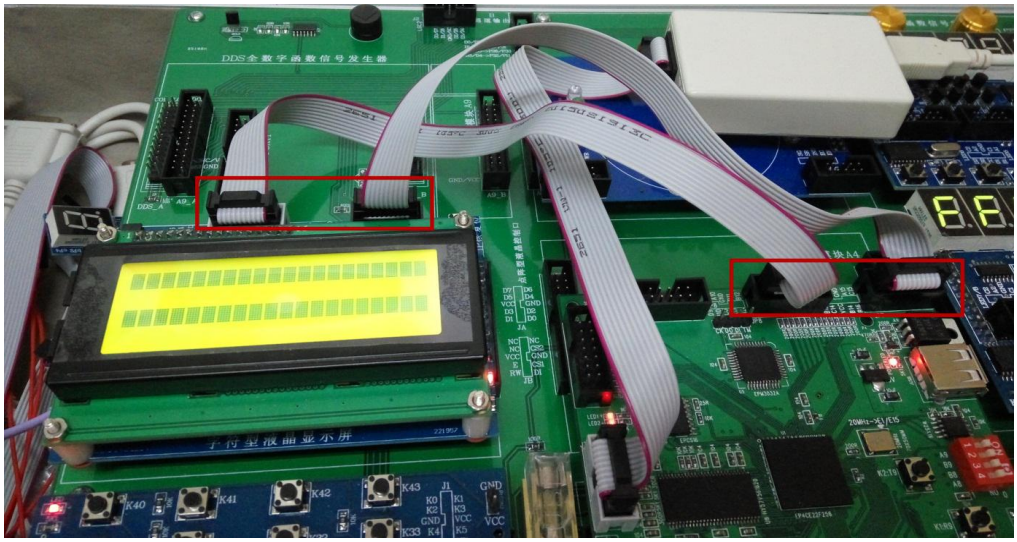
另外，用户自定义字符数据和待显示的字符串均可在独立的文本件中编译并存储，初始化时**将文本文件读入到 memory 存储器中即可使用**。将文件读取存储器到中，可用以下系统命令实现：

$readmemh(“filename”, mem_name);

五、实验步骤及分析报告

17、 按下图所示连线，时钟选用 256Hz（飞线到 E9 管脚）。编译整个工程并下载到实验箱，记录观察到的实验现象。

18、 分析状态机的状态转移关系，并画出状态转移图。



六、附加练习

1、将“SZU”加一空格显示在第一行字母后面。

2、修改状态机，将第 4 行内容显示到原来的第 3 行上，第 4 行不显示内容。