

**课程实验报告**

**VERILOG HDL语言设计实现数字钟**

|  |  |
| --- | --- |
| 院 系 | 材料科学与工程学院 |
| 专业班级 | 电子封装1801 |
| 姓 名 | 肖玉奇 |
| 学 号 | U201810991 |

2020年 11 月 10 日

**目 录**

[1 实验目的 3](#_Toc56015082)

[2 实验要求 3](#_Toc56015083)

[2.1 基本功能 3](#_Toc56015084)

[2.2 选做功能 3](#_Toc56015085)

[3 实验原理 4](#_Toc56015086)

[3.1 设计方法 4](#_Toc56015087)

[3.2 基础知识 4](#_Toc56015088)

[3.3 流程框图 5](#_Toc56015089)

[3.4 计时模块 5](#_Toc56015090)

[3.5 60进制BCD计数器 6](#_Toc56015091)

[3.6 24进制BCD计数器 6](#_Toc56015092)

[3.7 6位数码管显示 6](#_Toc56015093)

[3.8 24/12小时切换 7](#_Toc56015094)

[4 实验内容 7](#_Toc56015095)

[4.1 报时模块 7](#_Toc56015096)

[4.2 对时模块 8](#_Toc56015097)

[4.3 模24 8](#_Toc56015098)

[4.4 模6 10](#_Toc56015099)

[4.5 模10 10](#_Toc56015100)

[4.6 模60 10](#_Toc56015101)

[4.7 驱动模块 11](#_Toc56015102)

[4.8 1KHz频率产生 11](#_Toc56015103)

[4.9 50MHz频率产生 12](#_Toc56015104)

[4.10 点亮数码管 12](#_Toc56015105)

[4.11 闹钟模块 13](#_Toc56015106)

[4.12 扫描 14](#_Toc56015107)

[4.13 主程序代码 15](#_Toc56015108)

[4.14 \仿真时序图 18](#_Toc56015109)

[5 总结与感想 19](#_Toc56015110)

# 实验目的

1.掌握可编程逻辑器件的应用开发技术——设计输入、编译、仿真和器件编程；

2.熟悉EDA软件使用；

3.掌握VERILOG HDL设计方法；

4.分模块、分层次数字系统设计。

5.使用FPGA板实现多功能数字钟。

通过此次实验，我们将在实践中验证理论知识，不仅是为了巩固课堂上所学知识，更是为了加深我们对EDA技术和VHDL语言的理解;为了让我们自己动手完成从设计输入、逻辑综合、功能仿真、设计实现到实现编程、时序仿真，直到器件的下载测试的整个过程，真切感受利用EDA技术对FPGA进行设计开发的过程，锻炼和提高我们对器件的编程调试能力。

# 实验要求

## 基本功能

（1）能显示小时、分钟、秒钟（小时以24进制,时、分用显示器，秒用LED）；

（2）能调整小时、分钟的时间；

（3）复位。

## 选做功能

（1）任意闹钟；

（2）小时为12/24进制可切换；

（3）报正点数（几点钟LED闪烁几下）；

（4）仿电台报时。

# 实验原理

## 设计方法

课程设计要求我们设计的数字电子钟具有显示时分秒的功能，且能根据

set(设置)按钮进行相应的切换显示很容易想到数码管显示模块，即对set 脉冲进行计数，然后根据计得的数值进行相应的操作。

按照设计内容和要求以及所有的设计思路，综合考虑后，采用元件例化和进

程相结合的方法，设计模块化的结构:顶层设计实体为clock (时钟)模块，其

下又分为:秒脉冲、时分秒、闹钟、报时、数码管显示等模块。每个模块主要使用VHDL语言输入中常用的进程语句、元件例化语句、case 语句、if语句以及赋值语句。

## 基础知识

1.层次化，模块化的设计方法

对于一个复杂的数字系统, 运用层次化设计方法, 使设计课题进一步细化, 分块设计, 条理清晰。另外, 在调试时可采用逆向调试方式, 即从模块调试向总体调试方向开展调试工作, 使设计中出现的问题在模块级就能发现, 及时处理, 这样就会使一个复杂的设计变得容易调试, 缩短了设计时间。

2. 自顶向下---从系统级开始把系统划分为基本单元，然后再把每个基本单元划分为下一层次的基本单元，一直这样做下去直到可以直接用 元件库中的元件来实现为止。

3. 自下而上---是一种传统的设计方法，从存在的基本单元出发，设计树最末枝上的单元要么是已经制造出的单元要么是其他项目已开发好的单元或者是可外购得到的单元，逐级叠加，逐模块叠加，直至实现功能。

4. VERILOG 层次化设计中调用底层模块的方法：

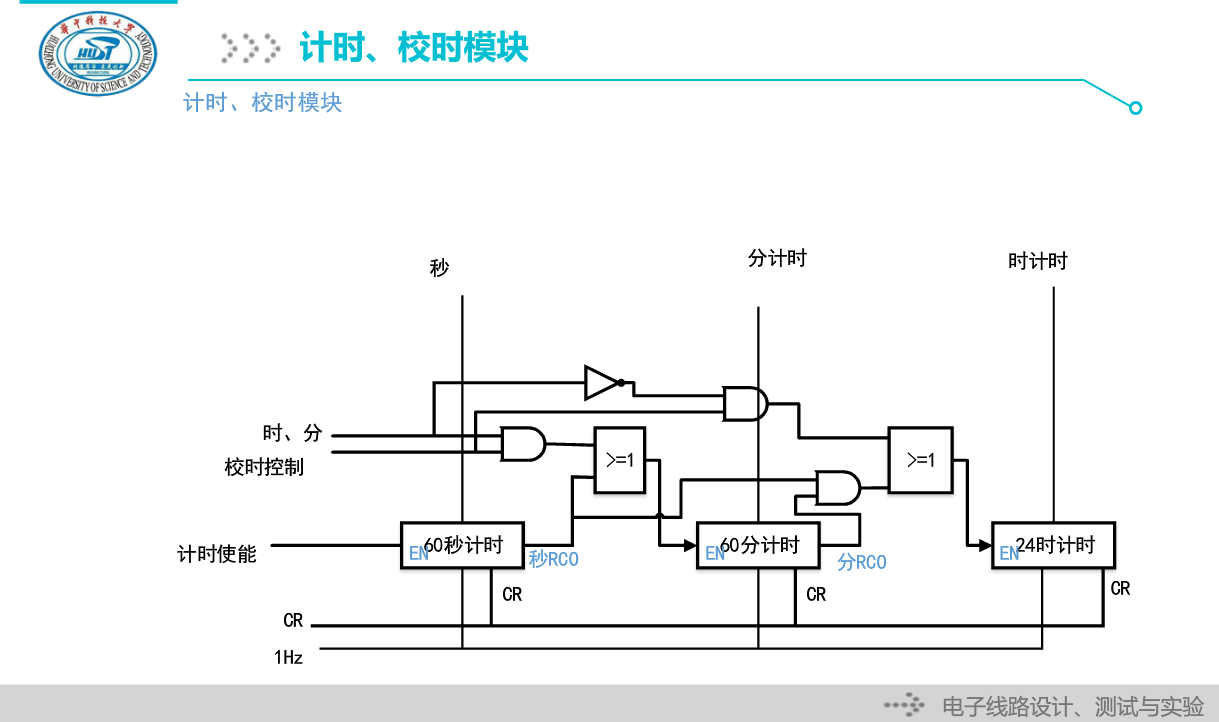
基本方式： 模块名 调用名（端口名表项）

调用方式一：位置对应调用方式；注意位置要严格对应；

调用方式二：端口名对应调用方式；注意端口名要保持一致。

## 流程框图

## 计时模块



## 60进制BCD计数器

## 24进制BCD计数器

## 6位数码管显示

## 24/12小时切换

# 实验内容

## 报时模块

module baoshi(clk,nCR,minuteh,minutel,hourh,hourl,rock );

input nCR;

input clk;

input [3:0] minuteh,minutel;

input [3:0] hourh,hourl;

output reg rock;

integer i=0;

reg en;

always@(posedge clk or negedge nCR)

begin

if(~nCR)

begin

rock<=0;

end

else if((minuteh==4'd5)&&(minutel==4'd9)) //到达报时时间段

begin

i=0; //对i进行置零

en<=1; //允许报时

end

else if((i<(hourh\*20+hourl\*2))&&(en)) //当i<整点数字以及允许报时时(en==1),开始报时

begin

rock<=~rock;

rock<=~rock;

i=i+1;

end

else if(i==(hourh\*20+hourl\*2)) //当i==整点数字时，允许信号置零

en<=0;

else

rock<=0;

end

endmodule

## 对时模块

module comparetime(CP, Set\_Hour, Set\_Minute, Hour, Minute, Time);

input [7:0] Set\_Hour, Set\_Minute, Hour, Minute;

input CP;

output reg Time;

always @(posedge CP)

begin

if((Set\_Hour == Hour) && (Set\_Minute == Minute))

Time<= 1;

else

Time<= 0;

end

endmodule

## 模24

module count24\_1(CntH,CntL,nCR,EN,CP,change);

input nCR,CP,EN,change;

output reg [3:0] CntH;

output reg [3:0] CntL;

always@(posedge CP,negedge nCR)

begin

if(nCR==0)

begin {CntH,CntL}<=8'b00000000; end

else if(change==0) //\*change=0时，进入24小时制\*\*\*\*\*\*\*\*\*\*\*

begin

if(EN==0)

begin {CntH,CntL}<={CntH,CntL}; end

else if((CntH>2)||(CntL>9)||((CntH==2)&&(CntL>=3)))

begin {CntH,CntL}<=8'b00000000; end

else if((CntH==2)&&(CntL<3))

begin CntH<=CntH;CntL<=CntL+1'b1;end

else if(CntL==9)

begin CntH<=CntH+1'b1;CntL<=4'b0000; end

else

begin CntH<=CntH;CntL<=CntL+1'b1; end

end

/\*else if(change==1)

begin

if((CntH>=1)&&(CntL>2))

begin tim<=1; CntH<=CntH-1'b1; CntL<=CntL-2'b10;end

//将24小时制的下午时间转换为12小时制，并显示下午

else if(EN==0)

begin {CntH,CntL}<={CntH,CntL}; end if((CntH>4'b0001)||(CntL>4'b1001)||((CntH==4'b0001)&&(CntL>=4'b0010)))

begin {CntH,CntL}<=8'b00000001; tim<=~tim; end

// led灯上午下午进行转变

else if((CntH==1)&&(CntL<1))

begin CntH<=CntH;CntL<=CntL+1'b1;end

else if(CntL==9)

begin CntH<=CntH+1'b1;CntL<=4'b0000; end

else

begin CntH<=CntH;CntL<=CntL+1'b1; end

end

end\*/

else if(change==1)//\*change=1时，进入12小时制\*\*\*\*\*\*\*\*\*

begin

if(CntH>=4'b0001&&CntL>4'b0010) begin CntH<=CntH-1'b1; CntL<=CntL-2'b10; end

else if(~EN) {CntH,CntL}<={CntH,CntL};

else

begin if((CntH>4'b0001)||(CntL>4'b1001)||((CntH==4'b0001)&&(CntL>=4'b0010)))

begin {CntH,CntL}<=8'h01; end

else if((CntH==4'b0001)&&(CntL<4'b0001))

begin CntH<=CntH; CntL<=CntL+1'b1; end

else if(CntL==4'b1001)

begin CntH<=CntH+1'b1; CntL<=4'b0000; end

else

begin CntH<=CntH; CntL<=CntL+1'b1; end

end

end//12进制小时计数完成

end

endmodule

## 模6

module count6(Q,nCR,EN,CP);

input EN,CP,nCR;

output reg [3:0] Q;

always@(posedge CP,negedge nCR)

begin

if(nCR==0)Q<=4'b0000;

else if(EN==0)Q<=Q;

else if(Q==4'b0101) Q<=4'b0000;

else Q<=Q+1'b1;

end

endmodule

## 模10

module count10(Q,nCR,EN,CP);

input EN,nCR,CP;

output reg [3:0]Q;

always@(posedge CP,negedge nCR)

begin

if(nCR==0)Q<=4'b0000;//清零

else if(EN==0)Q<=Q;

else if(Q==4'b1001)Q<=4'b0000;

else Q<=Q+1'b1;

end

endmodule

## 模60

//模60计数器模块

module counter60(clk, rst\_n, en, dout, co);

input clk, rst\_n, en;

output co;

output [7:0] dout;

wire co10\_1, co10, co6;

wire [3:0] dout10, dout6;

count10 U1(.clk(clk), .rst\_n(rst\_n), .en(en), .dout(dout10), .co(co10\_1));

count6 U2(.clk(clk), .rst\_n(rst\_n), .en(co10), .dout(dout6), .co(co6));

and U3(co, co10, co6);

and U4(co10, en, co10\_1);

assign dout = {dout6, dout10};

endmodule

## 驱动模块

module divider(CR,clk\_in,clk\_out );

parameter count=25;

parameter infreq=50000000;

parameter outfreq=1000;//修改为1

input clk\_in,CR;

output clk\_out;

reg clk\_out;

reg[count-1:0]counter;

initial begin

counter=0;

clk\_out=0;

end

always@(posedge clk\_in or negedge CR)

begin

if(CR==0)

begin

counter<=0;

clk\_out<=0;

end

else begin

if(counter<infreq/(2\*outfreq))

counter<=counter+1;

else begin

counter<=0;

clk\_out<=~clk\_out;

end

end

end

endmodule

## 1KHz频率产生

module divider1Hz(CP,nCR,clk );

input CP,nCR;

reg [29:0]count2;

output reg clk;

always @(posedge CP)//产生1Hz的频率

begin

if(nCR==0) begin clk=0;count2<=30'd00;end

else if(count2==30'd25000000) begin count2<=30'd00; clk=~clk; end

else count2<=count2+1'b1;

end

endmodule

## 50MHz频率产生

module divider(clk1,clr,clk2 );

parameter n=25;

parameter infreq=50000000;

parameter outfreq=1000;

input clk\_in,CR;

output clk\_out;

reg clk\_out;

reg[count-1:0]counter;

initial begin

counter=0;

clk\_out=0;

end

always@(posedge clk\_in or negedge CR)begin

if(CR==0)begin

counter<=0;

clk\_out<=0;

end

else begin

if(counter<infreq/(2\*outfreq))

counter<=counter+1;

else begin

counter<=0;

clk\_out<=~clk\_out;

end

end

end

endmodule

## 点亮数码管

module lut7(out,in );

input [3:0] in;

output reg [6:0] out;

always@(in)

begin

case(in)

4'h1:out=7'b111\_1001;

4'h2:out=7'b010\_0100;

4'h3:out=7'b011\_0000;

4'h4:out=7'b001\_1001;

4'h5:out=7'b001\_0010;

4'h6:out=7'b000\_0010;

4'h7:out=7'b111\_1000;

4'h8:out=7'b000\_0000;

4'h9:out=7'b001\_0000;

4'h0:out=7'b100\_0000;

default:out=7'b1000000;

endcase

end

endmodule

## 闹钟模块

module naozhong(clk\_2Hz,nCR,naozhong\_swh,,baochi\_swh,Adj\_Hour,Adj\_Min,hourl,hourh,minutel,minuteh,CntH,CntL,CntHm,CntLm,alarm);

input clk\_2Hz,nCR;

input naozhong\_swh,baochi\_swh; //闹钟开关

input Adj\_Hour,Adj\_Min;

input [3:0]hourl,hourh; //时间比较

input [3:0]minutel,minuteh;

output reg[3:0]CntH,CntL;

output reg[3:0]CntHm,CntLm;

output reg alarm;

always@(posedge clk\_2Hz)

begin

if((naozhong\_swh)&&(!baochi\_swh))

begin

if(nCR==0)

begin alarm=0;{CntHm,CntLm}<=8'b00000000;{CntH,CntL}<=8'b00000000; end //清零

else if(Adj\_Hour) //设定小时

begin

if((CntH>2)||(CntL>9)||((CntH==2)&&(CntL>=3)))

begin {CntH,CntL}<=8'b00000000; end

else if((CntH==2)&&(CntL<3))

begin CntH<=CntH;CntL<=CntL+1'b1;end

else if(CntL==9)

begin CntH<=CntH+1'b1;CntL<=4'b0000; end

else

begin CntH<=CntH;CntL<=CntL+1'b1; end

end

else if(Adj\_Min) //设定分钟

begin

if((CntHm>5)||(CntLm>9)||((CntHm==5)&&(CntLm>=9)))

begin {CntHm,CntLm}<=8'b00000000; end

else if((CntHm==5)&&(CntLm<9))

begin CntHm<=CntHm;CntLm<=CntLm+1'b1;end

else if(CntLm==9)

begin CntHm<=CntHm+1'b1;CntLm<=4'b0000; end

else

begin CntHm<=CntHm;CntLm<=CntLm+1'b1; end

end

end

end

//显示闹钟

always@(posedge clk\_2Hz)

begin

if((CntH == hourh)&&(CntL == hourl)&&(CntHm==minuteh)&&(CntLm==minutel))

begin

alarm <= 'd1;

end

else if(baochi\_swh)

begin

alarm <= 'd0;

end

else

begin

alarm <= 'd0;

end

end

endmodule

## 扫描

module scan(CP,nCR,clk1

);

input CP,nCR;

output reg clk1;

reg [29:0]count1;

always @(posedge CP)

begin

if(~nCR) count1<=30'd00;

else if(count1==30'd100000) begin count1<=30'd00; clk1=~clk1; end

else count1<=count1+1'b1;

end

endmodule

## 主程序代码

module top(Clock,Hour,Minute,Second,CP,nCR,EN,Adj\_Min,Adj\_Hour,out,Light,clk,in,change,tim,rock,q,w,e,r,t,Set\_Alarm,Set\_Min,Set\_Hour);

input CP; //输入的时钟信号

input nCR;

input EN,change; //change为12/24进制转换开关

input Adj\_Min; //校分控制

input Adj\_Hour; //校时控制

// \*\*

input Set\_Alarm;

input Set\_Min, Set\_Hour;

output Clock;

wire [7:0] C\_Hour, C\_Minute;

wire C\_MinL\_EN, C\_MinH\_EN;

wire C\_Hour\_EN;

//\*\*

output [7:0] Hour,Minute,Second;

output [3:0] Light,in; //light控制数码管，in控制显示数字

output [6:0] out; //控制led灯

output clk,tim; //clk为分频时钟信号，频率为1Hz

//

output rock;

//

output reg q,w,e,r,t;

wire [7:0] Hour,Minute,Second;

reg clk; //clk为1Hz的频率,clk为2Hz的频率

reg [29:0] count,count1; //分频需要的计数器

reg [1:0] scan; //扫描时需要的计数器

reg [3:0] Light,in;

reg [6:0] out;

reg clk1; //为分频后的扫描频率

supply1 Vdd; //vdd恒等于1

wire MinL\_EN,MinH\_EN,Hour\_EN;

//

//60进制秒计数器

count10 U1(Second[3:0],nCR,EN,clk); //秒个位

count6 U2(Second[7:4],nCR,(Second[3:0]==4'h9),clk); //秒十位

//60进制分钟计数器

count10 U3(Minute[3:0],nCR,MinL\_EN,clk);

count6 U4(Minute[7:4],nCR,MinH\_EN,clk);

//产生分钟计数器使能信号。Adj\_Min=1，校正分钟；Adj\_Min=0，分钟正常计时

assign MinL\_EN=Adj\_Min?Vdd:(Second==8'h59);

assign MinH\_EN=(Adj\_Min&&(Minute[3:0]==4'h9))||((Minute[3:0]==4'h9)&&(Second==8'h59));

//产生小时计数器使能信号。Adj\_Hour=1，校正小时；Adj\_Hour=0，小时正常计时

assign Hour\_EN=Adj\_Hour?Vdd:((Minute==8'h59)&&(Second==8'h59));

//小时计数器

count24 U5(Hour[7:4],Hour[3:0],nCR,Hour\_EN,clk,change,tim);

//整点报时\*\*\*\*\*\*\*\*\*\*\*\*\*

baoshi U9(clk,nCR,Minute[7:4],Minute[3:0],Hour[7:4],Hour[3:0],rock);

//闹钟\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

count10 M2 (C\_Minute[3:0], nCR, C\_MinL\_EN, clk);

count6 M3 (C\_Minute[7:4], nCR, C\_MinH\_EN, clk);

assign C\_MinL\_EN = (Set\_Min && Set\_Alarm);

assign C\_MinH\_EN = (Set\_Min && (C\_Minute[3:0] == 4'h9)&&Set\_Alarm);

count24\_1 H1(C\_Hour[7:4],C\_Hour[3:0],nCR,C\_Hour\_EN,clk,change);

assign C\_Hour\_EN=(Set\_Alarm&&Set\_Hour)?Vdd:(Set\_Alarm&&C\_Minute ==8'h59&&Set\_Min);

comparetime C1 (clk,C\_Hour,C\_Minute,Hour,Minute, Time);

assign Clock = Time;

//分频

always @(posedge CP)

begin

if(nCR==0) begin t<=0; q<=0;w<=0;e<=0;r<=0;clk<=0; count<=30'd00; end

else if(count==30'd25000000) begin count<=30'd00; clk<=~clk; end

else count<=count+1'b1;

end

//扫描

/\*scan U7(CP,nCR,clk1);\*/

always @(posedge CP)

begin

if(~nCR) begin clk1<=0; count1<=30'd00; end

else if(count1==30'd100000) begin count1<=30'd00; clk1<=~clk1; end

else count1<=count1+1'b1;

end

//扫描计数器

always @(posedge clk1)

begin

if(~nCR) scan<=2'b00;

else if(scan==2'b11) scan<=2'b00;

else scan<=scan+1'b1;

end

always @(scan[1:0])

begin

case(scan[1:0])

2'b00: Light<=4'b0111;

2'b01: Light<=4'b1011;

2'b10: Light<=4'b1101;

2'b11: Light<=4'b1110;

endcase

end

//显示数字

always @ (scan[1:0] or Hour or Minute or C\_Minute or C\_Hour or Set\_Alarm)

begin

if(Set\_Alarm==0) //显示时钟

begin

case(scan[1:0])

2'b00: in<=Hour[7:4];

2'b01: in<=Hour[3:0];

2'b10: in<=Minute[7:4];

2'b11: in<=Minute[3:0];

endcase

end

else //设置时钟信号

begin

case(scan[1:0])

2'b00: in<=C\_Hour[7:4];

2'b01: in<=C\_Hour[3:0];

2'b10: in<=C\_Minute[7:4];

2'b11: in<=C\_Minute[3:0];

endcase

end

end

/\*//译码

lut7 U8(out,in);\*/

always @(in)

begin

case(in)

4'd0:out<=7'b0000001; //0

4'd1:out<=7'b1001111; //1

4'd2:out<=7'b0010010; //2

4'd3:out<=7'b0000110; //3

4'd4:out<=7'b1001100; //4

4'd5:out<=7'b0100100; //5

4'd6:out<=7'b0100000; //6

4'd7:out<=7'b0001111; //7

4'd8:out<=7'b0000000; //8

4'd9:out<=7'b0000100; //9

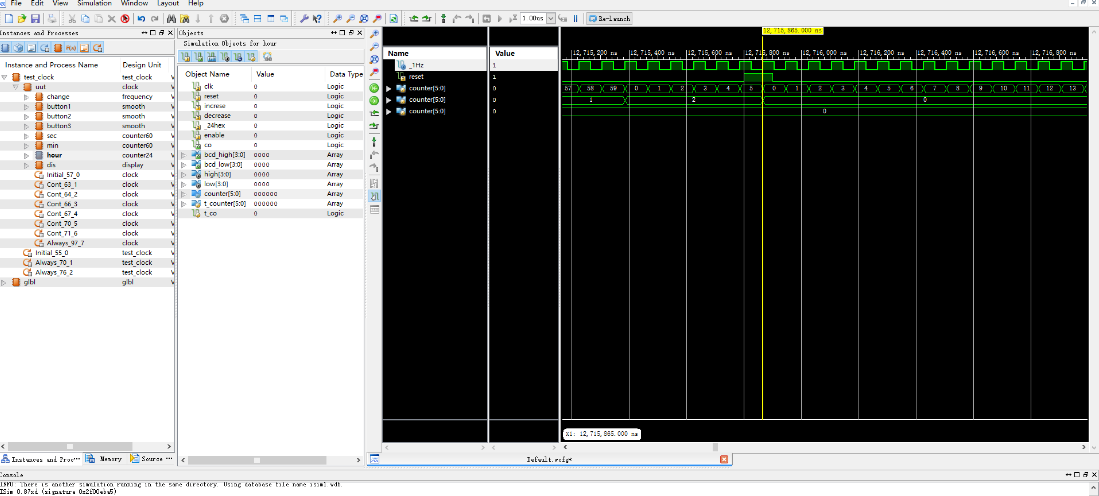
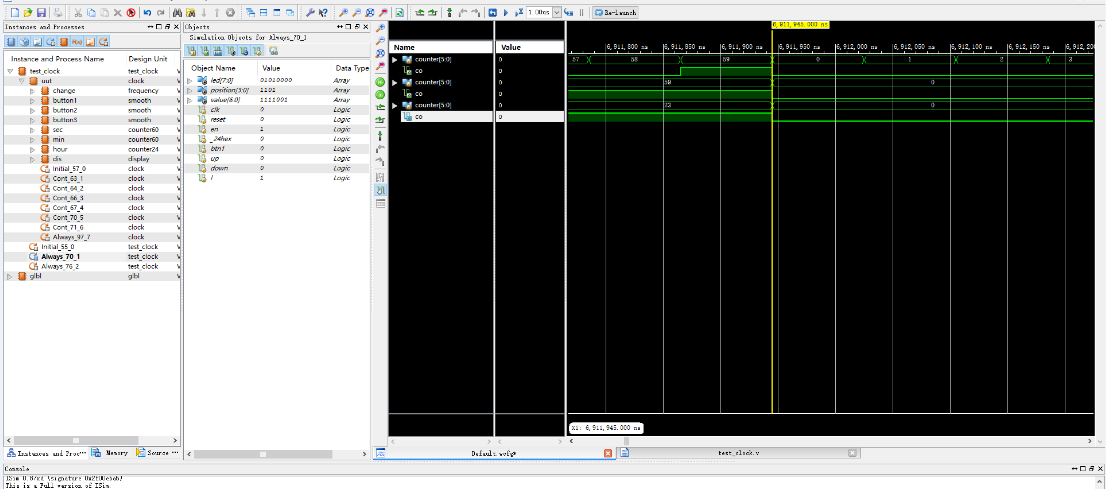
default:out<=7'b0000001;

endcase

end

endmodule

## \仿真时序图



# 总结与感想

在本实验中，我参考并仿照案例代码完成了基础功能，并且加入了LED灯计秒的功能，同时优化了按键布局。整个程序较之前的程序更加庞大，功能也更加复杂，但是拆分模块之后容易分部实现。主要的难点在整合部分，即主程序的编写。

通过此实验，我更加深入了解了VHDL语言的语法规则，并且练习了各种调用底层模块的方法，并且实现了简单的数字钟，整个实验还是较为成功的。