華南理工大學

South China University of Technology

课程设计报告书

基于VHDL的万年历系统设计

目录

一、选题背景

1.1 验收标准：

设计一个万年历系统，通过数码管显示。

（1）数码管显示时间信息，包括年、月、日、时、分、秒。

（2）数码管开始显示时分秒，通过按键切换显示年月日。

（3）按键设置时间，通过选择时间单位，增加的方式设置。

（4）具有闹钟功能，闹钟设置方式同时间设置方式。

（5）有走马灯功能。

1.2 设计思想：

这次设计是基于 VHDL 语言进行的，设计过程中，采用了模块化的思想，实现各个功能的设计。根据验收标准，把万年历分成计时模块、显示模块、时间设置模块、闹钟模块。

万年历有计时模式和闹钟模式，通过按键切换。在计时模式下，计时模块利用了 VHDL 分频计数原理，根据年月日时分秒设计不同进制的计数器进行计时进位。由于只有 8 个数码管，无法显示所有信息，所以显示模块中把年月日和时分秒分开显示，通过按键切换。时间设置模块需要 2 个按键进行控制，一个按键选择当前显示的时间中的一位，另一个按键改变时间。闹钟模式是计时模式的延伸，通过两个按键设置小时和分钟，一个按键复位，完成特定闹钟设置。在闹钟模式下，计时照常进行，完成闹钟设置后，到达设定时间，蜂鸣器会发出警报声，有流水灯提醒。

由于设计中多处使用到按键，所以加上延时去抖代码。在整体设计过程中，多处需要选择判断，可以通过 VHDL 中的 case 语句或 if 语句来实现。

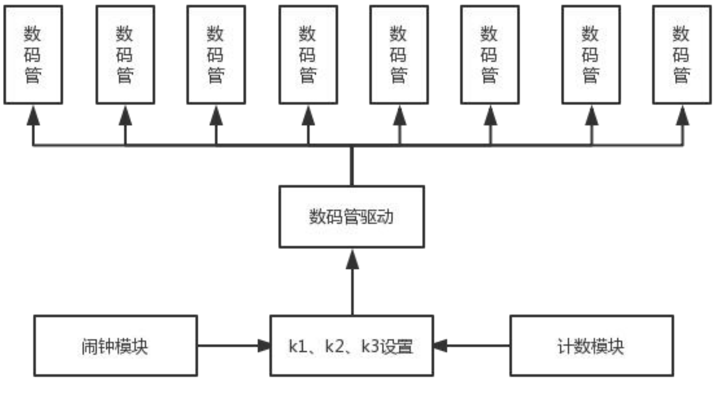

图 1 计时器运行流程图

二、方案论证(设计理念)

本次设计利用 VHDL 分频计数原理，对时间进行计数，根据秒到年不同进制进行设计，时间相对更精确。设计过程中利用 VHDL 中的 case 语句和 if 语句进行选择，逻辑清晰，代码可读性强，有利于后期维护。显示模块由于数码管数量限制，采用分开显示的方式，

是切实可行的。同时，由于使用了按键，加上延时去抖的模块，使之更稳定，减少使用误操作的概率。外接蜂鸣器模块，用于闹钟的提醒。走马灯功能也用在闹钟提醒里。

三、过程论述

3.1 整体设计

3.1.1 设计思路

万年历有计时模式和闹钟模式，通过按键 k0 进行切换，两种模式下，计时模块都照常进行。通过 led 灯 1、2、3 来显示当前状态。

计时模块根据时间不同包括了分频、秒、分、时、日、月、年模块。分频器采用系统 50MHz 信号，每记录 50M 次输出，即输出 1 秒。秒、分模块采用 60 进制计数器，时采用 24 进制计数器，月采用 12 进制计数器，年分高位和低位采用 100 进制计数器。而日模块，根据大月（1、3、5、7、8、10、12）和小月（4、6、9、11）和闰年二月和平年二月分别采用 31 进制、30 进制、29 进制、28 进制计数器，通过一个选择器对年份、月份信息判断选择相应计数器。同时除年高位外每个计数器都有进位输出，以实现低位时间单位向高一级的时间单位进位。

时间设置功能，按键 k1 选择要改变的时间单位，按键 k2 加 1，以实现设置时间。设置功能只改变年月日时分五个时间六个状态（年分高低位），通过 led 灯 1、2、3 来表示当前状态。

受硬件设备所限，在显示模块中，开始显示时分秒，按下 k3，显示年月日。

闹钟模块中，通过按键来控制时间，对 k1、k2、k3 按键实现复用，k1、k2 分别设置小时、分钟，k3 复位，完成设置后按 k0 退出。当系统时间和闹钟设置时间一致，闹钟会发出提醒。闹钟模块外接蜂鸣器，同时用 led 灯 4、5、6 作为流水灯，用于提醒。

3.1.2 整体流程图

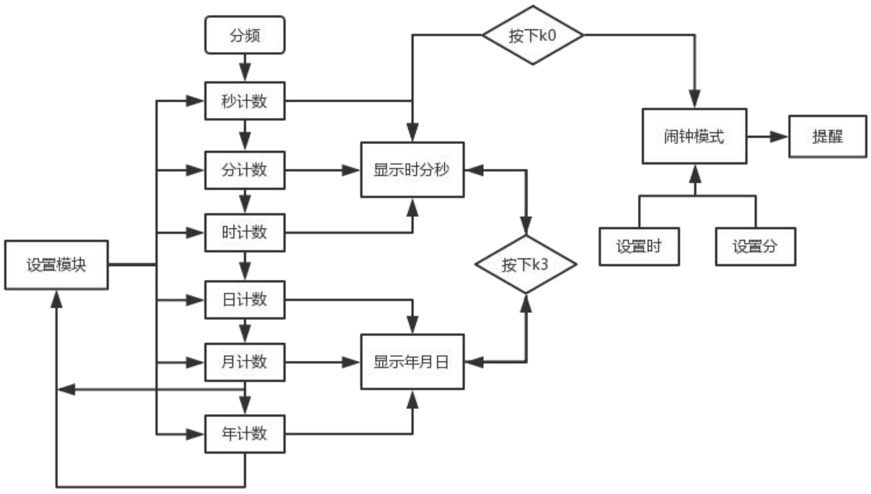

图 2 整体流程图

3.2 各模块设计分析

3.2.1 计数器模块

（1）分频模块

系统采用 50MHz 的信号，分频器每记录脉冲 50M 次输出，实现输出周期为 1 秒的信号，作为秒计数器的输入。

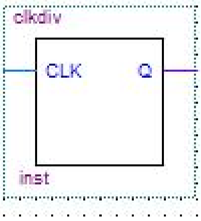

图 3 分频模块图

代码如下：

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY clkdiv IS
PORT(CLK:IN STD_LOGIC;Q:OUT STD_LOGIC);
END clkdiv;
ARCHITECTURE BEHAV OF clkdiv IS
BEGIN
PROCESS(CLK)
VARIABLE TIME:INTEGER RANGE 0 TO 50000000;
BEGIN
    IF RISING_EDGE(CLK) THEN
        TIME:=TIME+1;
        IF TIME=25000000 THEN
                Q<='1';
        ELSIF TIME=50000000 THEN
            Q<='0';
            TIME:=0;
        END IF;
    END IF;
END PROCESS;
END BEHAV;
```

（2）秒计数器模块

秒计数器为60进制计数器，clk有时钟输入时，计数器计数，q口输出为当前计数，输出到显示模块，c为进位输出，当计数到达60时，计数置零，同时c口输出。

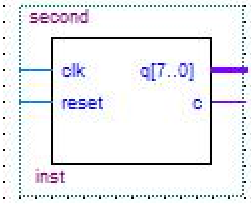

图4 秒计数器模块图

代码如下：

```
Library ieee;
Use ieee.std_logic_1164.all;
Use ieee.std_logic_unsigned.all;
Entity second is
Port(clk,reset:in std_logic;
    q:out std_logic_vector(7 downto 0);
    c:out std_logic);
end entity;
architecture behav of second is
begin
process(clk)
variable r:std_logic_vector(7 downto 0);
begin
      if reset='0' then r:="00000000";
      elsif (rising_edge(clk)) then
         r:=r+1;c<='0';
         if r(3 downto 0)="1010"  then
            r(3 downto 0):="0000";
            if r(7 downto 4)/="0101" then
               r(7 downto 4):=r(7 downto 4)+1;
            elsif r(7 downto 4)="0101"then
               r(7 downto 4):="0000";c<='1';
            end if;
         end if;
      end if;
      q<=r;
```

```
end process;
end behav;
```

（3）分计数器模块

分计数器也为60进制计数器，同秒计数器。

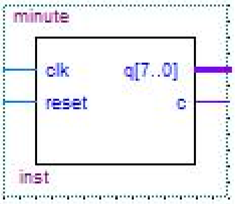

图 5 分计数器模块图

代码如下：

```
Library ieee;
Use ieee.std_logic_1164.all;
Use ieee.std_logic_unsigned.all;
Entity minute is
Port(clk,reset:in std_logic;
    q:out std_logic_vector(7 downto 0);
    c:out std_logic);
end entity;
architecture behav of minute is
begin
process(clk)
variable r:std_logic_vector(7 downto 0);
begin
      if reset=’0’ then r:=″00000000″;
      elsif (rising_edge(clk)) then
          r:=r+1;c<=’0’;
          if r(3 downto 0)=″1010″  then
              r(3 downto 0):=″0000″;
              if r(7 downto 4)/=″0101″ then
                  r(7 downto 4):=r(7 downto 4)+1;
              elsif r(7 downto 4)=″0101″then
                  r(7 downto 4):=″0000″;c<=’1’;
              end if;
          end if;
```

```
        end if;
        q<=r;
end process;
end behav;
```

（4）时计数器模块

时计数器为 24 进制计数器，其它同秒计数器。

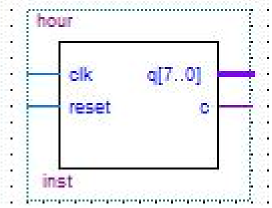

图 6 时计数器模块图

代码如下：

```
Library ieee;
Use ieee.std_logic_1164.all;
Use ieee.std_logic_unsigned.all;
Entity hour is
Port(clk,reset:in std_logic;
    q:out std_logic_vector(7 downto 0);
    c:out std_logic);
end entity;
architecture behav of hour is
begin
process(clk,reset)
variable r:std_logic_vector(7 downto 0);
begin
        if reset=’0’ then r:=″00000000″;
        elsif (rising_edge(clk)) then
            r:=r+1;c<=’0’;
            if r(3 downto 0)=4 and r(7 downto 4)=2 then
                r(3 downto 0):=″0000″;r(7 downto 4):=″0000″;c<=’1’;
            elsif r(3 downto 0)=10  then
                r(3 downto 0):=″0000″;r(7 downto 4):=r(7 downto 4)+1;
            end if;
        end if;
        q<=r;
```

```
end process;
end behav;
```

（5）日计数器模块

当有时钟输入 clk 口时，日计数器要根据 sel 输入判断日计数器的进制，包括大月 31 进制、小月 30 进制、闰年二月 29 进制、平年二月 28 进制。计数方式同上。

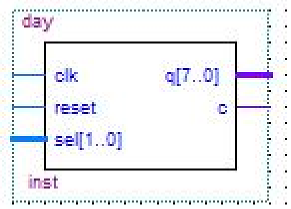

图 7 日计数器模块

代码如下：

```
Library ieee;
Use ieee.std_logic_1164.all;
Use ieee.std_logic_unsigned.all;
Entity day is
Port(clk,reset:in std_logic;
    sel:in std_logic_vector(1 downto 0);
    q:out std_logic_vector(7 downto 0);
    c:out std_logic);
end entity;
architecture behav of day is
begin
process(clk,reset)
variable r:std_logic_vector(7 downto 0);
begin
        if reset='0' then r:="00000001";
        elsif (rising_edge(clk)) then
            r:=r+1;c<='0';
            case sel is
            when"00"=>
                if r(3 downto 0)=2 and r(7 downto 4)=3 then
                    r(3 downto 0):="0001";r(7 downto 4):="0000";c<='1';
                elsif r(3 downto 0)=10  then
                    r(3 downto 0):="0000";r(7 downto 4):=r(7 downto 4)+1;
                end if;
            when"01"=>
                if r(3 downto 0)=1 and r(7 downto 4)=3 then
                    r(3 downto 0):="0001";r(7 downto 4):="0000";c<='1';
                elsif r(3 downto 0)=10  then
                    r(3 downto 0):="0000";r(7 downto 4):=r(7 downto 4)+1;
                end if;
            when"10"=>
                if r(3 downto 0)=10 and r(7 downto 4)=2 then
                    r(3 downto 0):="0001";r(7 downto 4):="0000";c<='1';
                elsif r(3 downto 0)=10  then
                    r(3 downto 0):="0000";r(7 downto 4):=r(7 downto 4)+1;
                end if;
            when"11"=>
                if r(3 downto 0)=9 and r(7 downto 4)=2 then
                    r(3 downto 0):="0001";r(7 downto 4):="0000";c<='1';
                elsif r(3 downto 0)=10  then
                    r(3 downto 0):="0000";r(7 downto 4):=r(7 downto 4)+1;
                end if;
            when others=>null;
            end case;
        end if;
        q<=r;
end process;
end behav;
```

（6）月计数器模块

时钟到来时，月计数器根据 run 输入判断是否为闰年，然后根据计数信息，通过 sel 把信息发给日计数器。计数方式同上。

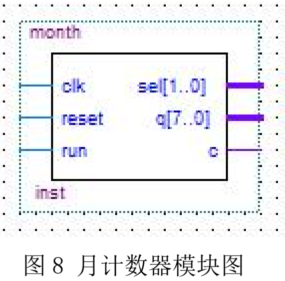

图 8 月计数器模块图

代码如下：

```
Library ieee;
Use ieee.std_logic_1164.all;
Use ieee.std_logic_unsigned.all;
Entity month is
Port(clk,reset:in std_logic;
    run:in std_logic;--runnian wei 1
    sel:out std_logic_vector(1 downto 0);
     q:out std_logic_vector(7 downto 0);
     c:out std_logic);
end entity;
architecture behav of month is
begin
process(clk,run,reset)
variable r:std_logic_vector(7 downto 0);
begin
        if reset=’0’ then r:=″00000001″;
        elsif (rising_edge(clk)) then
            r:=r+1;c<=’0’;
            if r(3 downto 0)=3 and r(7 downto 4)=1 then
                r(3 downto 0):=″0001″;r(7 downto 4):=″0000″;c<=’1’;
            elsif r(3 downto 0)=10  then
                r(3 downto 0):=″0000″;r(7 downto 4):=r(7 downto 4)+1;
            end if;
        end if;
        case r is
            when ″00000001″=>sel<=″00″;
            when ″00000010″=>sel<=″11″;
                if(run=’1’)then sel<=″10″;
                end if;
            when ″00000011″=>sel<=″00″;
            when ″00000100″=>sel<=″01″;
            when ″00000101″=>sel<=″00″;
            when ″00000110″=>sel<=″01″;
            when ″00000111″=>sel<=″00″;
            when ″00001000″=>sel<=″00″;
```

```
           when "00001001"=>sel<="01";
           when "00001010"=>sel<="00";
           when "00001011"=>sel<="01";
           when "00001100"=>sel<="00";
           when others=>null;
       end case;
       q<=r;
end process;
end behav;
```

（7）年计数器模块

年计数器分为置年高位和低位两种模式，由输入 uph 口控制。根据自身计数判断是否为闰年，通过 run 口输入到月计数器中。当处于置低位模式，每次计数加 1，当处于置高位模式，每次计数加 100。其它计数方式同上。

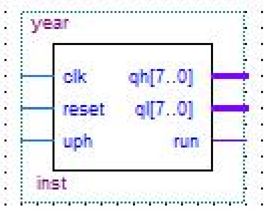

图 9 年计数模块

代码如下：

```
Library ieee;
Use ieee.std_logic_1164.all;
Use ieee.std_logic_unsigned.all;
Entity year is
Port(clk,reset,uph:in std_logic;
    qh:out std_logic_vector(7 downto 0);
    ql:out std_logic_vector(7 downto 0);
    run:out std_logic);
end entity;
architecture behav of year is
begin
process(clk,reset)
variable r:std_logic_vector(15 downto 0);
variable a:integer range 1 to 4;
variable b:integer range 1 to 100;
variable c:integer range 1 to 400;
begin
        if reset=’0’ then r:=″0010000000000000″;
        elsif (rising_edge(clk)) then
            a:=a+1;b:=b+1;c:=c+1;run<=’0’;
            if uph=’1’then r(11 downto 8):=r(11 downto 8)+1;
                if r(11 downto 8)=″1010″ then
                    r(11 downto 8):=″0000″;
                    r(15 downto 12):=r(15 downto 12)+1;
                end if;
            else r:=r+1;
            end if;
            if a=4 then run<=’1’;a:=1;
            end if;
            if b=100 then run<=’0’;b:=1;
            end if;
            if c=100 then run<=’1’;c:=1;
            end if;
            if r(3 downto 0)=″1010″  then
                r(3 downto 0):=″0000″;
                r(7 downto 4):=r(7 downto 4)+1;
                if r(7 downto 4)=″1010″ then
                    r(7 downto 4):=″0000″;
                    r(15 downto 8):=r(15 downto 8)+1;
                    if r(11 downto 8)=″1010″ then
                        r(11 downto 8):=″0000″;
                        r(15 downto 12):=r(15 downto 12)+1;
                    end if;
                end if;
            end if;
            if r=9999 then
                r:=″0000000000000000″;
            end if;

        end if;
        qh<=r(15 downto 8);
        ql<=r(7 downto 0);
end process;
end behav;
```

3.2.2 设置模块

设置模块通过 case 语句实现。k1 选择输入模式，包括正常显示、置分、置时、置日、置月、置年高低位其中模式，通过 led 灯 1、2、3 亮灭表示当前所处状态。选择好时间后，每按下 k2 一次就加 1，以完成时间的设置。

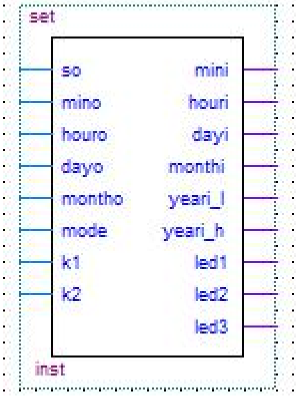

图 10 设置模块图

代码如下：

```
Library ieee;
Use ieee.std_logic_1164.all;
Use ieee.std_logic_unsigned.all;
Entity set is
Port( so,mino,houro,dayo,montho:in std_logic;
        mode:in std_logic;
        k1,k2:in std_logic;
        mini,houri,dayi,monthi,yeari_l,yeari_h:out std_logic;
        led1,led2,led3:out std_logic);
end entity;
architecture behav of set is
    signal r:std_logic_vector(2 downto 0);
begin
        process(mode,k1,k2,so,mino,houro,dayo,montho)
        begin
            case mode is
                when '0'=>
                    if k1'event and k1='1' then
                        r<=r+'1';
```

```
                           if r=6 then
                               r<="000";
                           end if;
                       end if;
                       yeari_l<=’0’;
                       yeari_h<=’0’;
                       case r is
                           when"000"=>mini<=so;houri<=mino;dayi<=      houro;monthi<=
dayo;yeari_l<= montho;
                                  led1<=’0’;led2<=’0’;led3<=’0’;
                           when"001"=>mini<=k2;houri<=’0’; dayi<=’0’;   monthi<=’0’;
                                  led1<=’0’;led2<=’0’;led3<=’1’;
                           when"010"=>mini<=so;houri<=k2;  dayi<=’0’;   monthi<=’0’;
                                  led1<=’0’;led2<=’1’;led3<=’0’;
                           when"011"=>mini<=so;houri<=mino;dayi<=k2;    monthi<=’0’;
                                  led1<=’0’;led2<=’1’;led3<=’1’;
                           when"100"=>mini<=so;houri<=mino;dayi<= houro;monthi<=k2;
                                  led1<=’1’;led2<=’0’;led3<=’0’;
                           when"101"=>mini<=so;houri<=mino;dayi<=
houro;monthi<=dayo;yeari_l<=k2;
                                  led1<=’1’;led2<=’0’;led3<=’1’;
                           when"110"=>mini<=so;houri<=mino;dayi<=
houro;monthi<=dayo;yeari_l<=k2;yeari_h<=’1’;
                                  led1<=’1’;led2<=’1’;led3<=’0’;

                           when others=>null;
                           end case;
                   when ’1’=>
                       r<="000";
                       mini<=so;
                       houri<=mino;
                       dayi<= houro;
                       monthi<= dayo;
                       yeari_l<= montho;
                       led1<=’1’;
                       led2<=’1’;
```

```
                    led3<='1';
                end case;
            end process;
        end behav;
```

3.2.3 显示切换模块

由于只有 8 个数码管，年月日和时分秒要分开显示，通过按键 k3 切换显示。

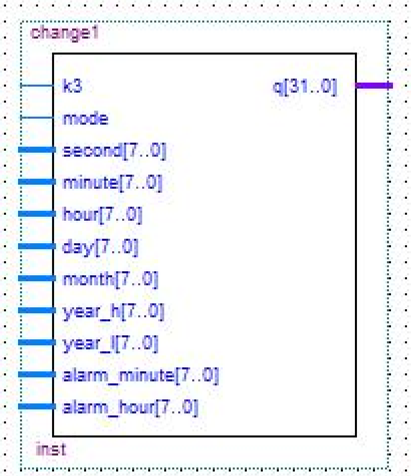

图 11 显示切换模块

代码如下：

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;
entity change1 is
port(k3:in std_logic;
    mode:in std_logic;
     second:in std_logic_vector(7 downto 0);
    minute:in std_logic_vector(7 downto 0);
    hour:in std_logic_vector(7 downto 0);
    day:in std_logic_vector(7 downto 0);
    month:in std_logic_vector(7 downto 0);
    year_h:in std_logic_vector(7 downto 0);
    year_l:in std_logic_vector(7 downto 0);
    alarm_minute:in std_logic_vector(7 downto 0);
    alarm_hour:in std_logic_vector(7 downto 0);
     q:out std_logic_vector(31 downto 0));
end entity;
```

```
    architecture behav of change1 is
    signal key_value:std_logic_vector(1 downto 0);
    begin
        process(k3)
        begin
            if(k3’event and k3=’1’) then
                   key_value<=key_value+’1’;
                   if key_value=″01″ then
                   key_value<=″00″;
                   end if;
            end if;
        end process;

process(mode,key_value,second,minute,hour,day,month,year_h,year_l,alarm_hour,alarm_mi
nute)
        begin
        case mode is
            when ’0’=>
            if key_value=″00″ then
                q<=″00000000″&hour&minute&second;
            else
                q<=year_h&year_l&month&day;
            end if;
            when ’1’=>
                q<=″00000000″&alarm_hour&alarm_minute&″00000000″;
        end case;
        end process;
    end behav;
```

3.2.4 数码管驱动模块

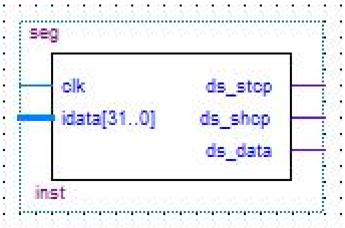

图 12 数码管驱动模块

代码如下：

```
LIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;
use IEEE.STD_LOGIC_unsigned.all;
ENTITY seg IS
  PORT(clk:IN std_logic;
       idata:IN std_logic_vector(31 downto 0);
       ds_stcp,ds_shcp,ds_data:OUT std_logic);
END entity;
ARCHITECTURE behav OF seg IS
   signal b,clk2,ds_stcpr,ds_shcpr,ds_datar:std_logic;
   signal cnt: std_logic_vector(11 DOWNTO 0);
   signal seg_duan :std_logic_vector(7 downto 0);
   signal seg_wei :std_logic_vector(7 downto 0);
   signal seg_num:std_logic_vector(3 downto 0);
   signal sel:std_logic_vector(2 downto 0);
   --signal idata: std_logic_vector(31 DOWNTO 0);

   CONSTANT SEG_NUM0:std_logic_vector(7 downto 0):=″11000000″;
   CONSTANT SEG_NUM1:std_logic_vector(7 downto 0):=″11111001″;
   CONSTANT SEG_NUM2:std_logic_vector(7 downto 0):=″10100100″;
   CONSTANT SEG_NUM3:std_logic_vector(7 downto 0):=″10110000″;
   CONSTANT SEG_NUM4:std_logic_vector(7 downto 0):=″10011001″;
   CONSTANT SEG_NUM5:std_logic_vector(7 downto 0):=″10010010″;
   CONSTANT SEG_NUM6:std_logic_vector(7 downto 0):=″10000010″;
   CONSTANT SEG_NUM7:std_logic_vector(7 downto 0):=″11111000″;
   CONSTANT SEG_NUM8:std_logic_vector(7 downto 0):=″10000000″;
   CONSTANT SEG_NUM9:std_logic_vector(7 downto 0):=″10010000″;

   CONSTANT SEG_WE0:std_logic_vector(7 downto 0):=″11111110″;
   CONSTANT SEG_WE1:std_logic_vector(7 downto 0):=″11111101″;
   CONSTANT SEG_WE2:std_logic_vector(7 downto 0):=″11111011″;
   CONSTANT SEG_WE3:std_logic_vector(7 downto 0):=″11110111″;
   CONSTANT SEG_WE4:std_logic_vector(7 downto 0):=″11101111″;
   CONSTANT SEG_WE5:std_logic_vector(7 downto 0):=″11011111″;
   CONSTANT SEG_WE6:std_logic_vector(7 downto 0):=″10111111″;
   CONSTANT SEG_WE7:std_logic_vector(7 downto 0):="01111111";


BEGIN
   process(clk)
   begin
   if(clk’event and clk=’1’) then
    clk2<=’0’;
    clk2<=not clk2;
   end if;
   end process;
   process(clk2,sel,seg_num)
   begin
   if(clk2’event and clk2=’1’) then
     cnt<=cnt+1;

    if(        (cnt >= x″002″ and cnt < x″022″) or (cnt >= x″024″ and cnt < x″044″)
           or (cnt >= x″046″ and cnt < x″066″) or (cnt >= x″068″ and cnt < x″088″)
           or (cnt >= x″08a″ and cnt < x″0aa″) or (cnt >= x″0ac″ and cnt < x″0cc″)
           or (cnt >= x″0ce″ and cnt < x″0ee″) or (cnt >= x″0f0″ and cnt < x″110″)
           or (cnt >= x″112″ and cnt < x″132″) or (cnt >= x″134″ and cnt < x″154″)
           or (cnt >= x″156″ and cnt < x″176″) or (cnt >= x″178″ and cnt < x″198″)
           or (cnt >= x″19a″ and cnt < x″1ba″) or (cnt >= x″1bc″ and cnt < x″1dc″)
           or (cnt >= x″1de″ and cnt < x″1fe″)

           )
        then
        ds_shcpr<=not ds_shcpr ;
        else ds_shcpr<=’0’;
        end if;
        if(  (cnt = x″001″) or (cnt = x″023″)
            or (cnt = x″045″) or (cnt = x″067″)
            or (cnt = x″089″) or (cnt = x″0ab″)
             or (cnt = x″0cd″) or (cnt = x″0ef″)
            or (cnt = x″111″) or (cnt = x″133″)
            or (cnt = x″155″) or (cnt = x″177″)
```

```
          or (cnt = x"199") or (cnt = x"1bb")
          or (cnt = x"1dd") or (cnt = x"199")
          ) then
          ds_stcpr<='0';
          end if;
        if(  (cnt = x"022") or (cnt = x"044")
          or (cnt = x"066") or (cnt = x"088")
          or (cnt = x"0aa") or (cnt = x"0cc")
           or (cnt = x"0ee") or (cnt = x"110")
          or (cnt = x"132") or (cnt = x"154")
          or (cnt = x"176") or (cnt = x"198")
          or (cnt = x"1ba") or (cnt = x"1dc")
          or (cnt = x"1fe")
          ) then
          ds_stcpr<='1';
          end if;

if (cnt=x"200")then
 cnt<=x"000";
 end if;
end if;
sel<=cnt(8 downto 6);
case sel is
 when "000" => seg_wei<=seg_we0;seg_num<=idata(3 downto 0);
 when "001" => seg_wei<=seg_we1;seg_num<=idata(7 downto 4);
 when "010" => seg_wei<=seg_we2;seg_num<=idata(11 downto 8 );
 when "011" => seg_wei<=seg_we3;seg_num<=idata(15 downto 12);
 when "100" => seg_wei<=seg_we4;seg_num<=idata(19 downto 16);
 when "101" => seg_wei<=seg_we5;seg_num<=idata(23 downto 20);
 when "110" => seg_wei<=seg_we6;seg_num<=idata(27 downto 24);
 when "111" => seg_wei<=seg_we7;seg_num<=idata(31 downto 28);
 when others=>NULL;
 end case;
 case seg_num is
 when "0000" =>seg_duan <= SEG_NUM0;
 when "0001" =>seg_duan <= SEG_NUM1;
```

```
        when "0010" =>seg_duan <= SEG_NUM2;
        when "0011" =>seg_duan <= SEG_NUM3;
        when "0100" =>seg_duan <= SEG_NUM4;
        when "0101" =>seg_duan <= SEG_NUM5;
        when "0110" =>seg_duan <= SEG_NUM6;
        when "0111" =>seg_duan <= SEG_NUM7;
        when "1000" =>seg_duan <= SEG_NUM8;
        when "1001" =>seg_duan <= SEG_NUM9;
        when others=>NULL;
        end case;
        end process;
        process(cnt)
        begin
             case cnt is
              when x"002"|x"046"|x"08a"|x"0ce"|x"112"|x"156"|x"19a"|x"1de"  => ds_datar
<= seg_duan(7);
              when x"004"|x"048"|x"08c"|x"0d0"|x"114"|x"158"|x"19c"|x"1e0"  => ds_datar
<= seg_duan(6);
              when x"006"|x"04a"|x"08e"|x"0d2"|x"116"|x"15a"|x"19e"|x"1e2"  => ds_datar
<= seg_duan(5);
              when x"008"|x"04c"|x"090"|x"0d4"|x"118"|x"15c"|x"1a0"|x"1e4"  => ds_datar
<= seg_duan(4);
              when x"00a"|x"04e"|x"092"|x"0d6"|x"11a"|x"15e"|x"1a2"|x"1e6"  => ds_datar
<= seg_duan(3);
              when x"00c"|x"050"|x"094"|x"0d8"|x"11c"|x"160"|x"1a4"|x"1e8"  => ds_datar
<= seg_duan(2);
              when x"00e"|x"052"|x"096"|x"0da"|x"11e"|x"162"|x"1a6"|x"1ea"  => ds_datar
<= seg_duan(1);
              when x"010"|x"054"|x"098"|x"0dc"|x"120"|x"164"|x"1a8"|x"1ec"  => ds_datar
<= seg_duan(0);
              when x"012"|x"056"|x"09a"|x"0de"|x"122"|x"166"|x"1aa"|x"1ee"  => ds_datar
<= seg_wei(0);
              when x"014"|x"058"|x"09c"|x"0e0"|x"124"|x"168"|x"1ac"|x"1f0"  => ds_datar
<= seg_wei(1);
              when x"016"|x"05a"|x"09e"|x"0e2"|x"126"|x"16a"|x"1ae"|x"1f2"  => ds_datar
<= seg_wei(2);
```

```
             when x″018″|x″05c″|x″0a0″|x″0e4″|x″128″|x″16c″|x″1b0″|x″1f4″  => ds_datar <= seg_wei(3);
             when x″01a″|x″05e″|x″0a2″|x″0e6″|x″12a″|x″16e″|x″1b2″|x″1f6″  => ds_datar <= seg_wei(4);
             when x″01c″|x″060″|x″0a4″|x″0e8″|x″12c″|x″170″|x″1b4″|x″1f8″  => ds_datar <= seg_wei(5);
             when x″01e″|x″062″|x″0a6″|x″0ea″|x″12e″|x″172″|x″1b6″|x″1fa″  => ds_datar <= seg_wei(6);
             when x″020″|x″064″|x″0a8″|x″0ec″|x″130″|x″174″|x″1b8″|x″1fc″  => ds_datar <= seg_wei(7);

             when x″024″|x″068″|x″0ac″|x″0f0″|x″134″|x″178″|x″1bc″  => ds_datar <=’1’;
             when x″026″|x″06a″|x″0ae″|x″0f2″|x″136″|x″17a″|x″1be″  => ds_datar <=’1’;
             when x″028″|x″06c″|x″0b0″|x″0f4″|x″138″|x″17c″|x″1c0″  => ds_datar <=’1’;
             when x″02a″|x″06e″|x″0b2″|x″0f6″|x″13a″|x″17e″|x″1c2″  => ds_datar <=’1’;
             when x″02c″|x″070″|x″0b4″|x″0f8″|x″13c″|x″180″|x″1c4″  => ds_datar <=’1’;
             when x″02e″|x″072″|x″0b6″|x″0fa″|x″13e″|x″182″|x″1c6″  => ds_datar <=’1’;
             when x″030″|x″074″|x″0b8″|x″0fc″|x″140″|x″184″|x″1c8″  => ds_datar <=’1’;
             when x″032″|x″076″|x″0ba″|x″0fe″|x″142″|x″186″|x″1ca″  => ds_datar <=’1’;
             when x″034″|x″078″|x″0bc″|x″100″|x″144″|x″188″|x″1cc″  => ds_datar <=’1’;
             when x″036″|x″07a″|x″0be″|x″102″|x″146″|x″18a″|x″1ce″  => ds_datar <=’1’;
             when x″038″|x″07c″|x″0c0″|x″104″|x″148″|x″18c″|x″1d0″  => ds_datar <=’1’;
             when x″03a″|x″07e″|x″0c2″|x″106″|x″14a″|x″18e″|x″1d2″  => ds_datar <=’1’;
             when x″03c″|x″080″|x″0c4″|x″108″|x″14c″|x″190″|x″1d4″  => ds_datar <=’1’;
             when x″03e″|x″082″|x″0c6″|x″10a″|x″14e″|x″192″|x″1d6″  => ds_datar <=’1’;
             when x″040″|x″084″|x″0c8″|x″10c″|x″150″|x″194″|x″1d8″  => ds_datar <=’1’;
             when x″042″|x″086″|x″0ca″|x″10e″|x″152″|x″196″|x″1da″  => ds_datar <=’1’;
             when others=>ds_datar<=’0’;
             end case;
       end process;
                  ds_stcp<=ds_stcpr;
                  ds_shcp<=ds_shcpr;
                  ds_data<=ds_datar;
    end behav;
```

3.2.5 闹钟模块

（1）闹钟设置模块

mode 有输入时，系统进入闹钟模式，k1、k2、k3 按键功能改变。闹钟模式下，k1 设置小时，k2 设置分钟，k3 置零完成。

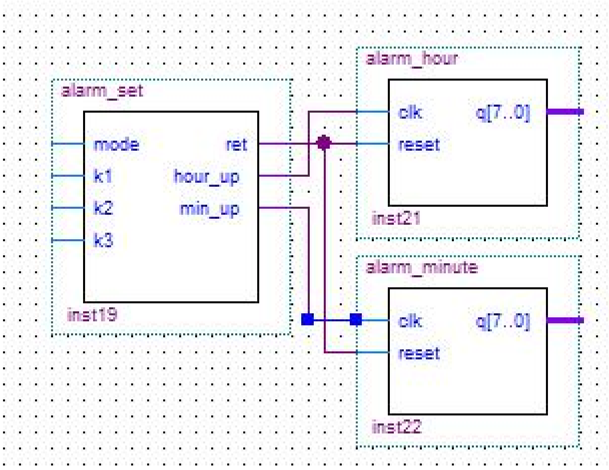

图 13 闹钟设置模块

代码如下：

```
Library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
Entity alarm_set is
Port( mode:in std_logic;
      k1,k2,k3:in std_logic;
      ret:out std_logic;
      hour_up,min_up:out std_logic);
end entity;
architecture behav of alarm_set is
begin
      process(mode,k1,k2,k3)
      begin
      case mode is
         when ’1’ =>
               if k1=’1’ then
                  hour_up<=’1’;
               else hour_up<=’0’;
               end if;
               if k2=’1’ then
```

```
                     min_up<=’1’;
                 else min_up<=’0’;
                 end if;
                 if k3=’1’ then
                     ret<=’1’;
                 else ret<=’0’;
                 end if;
          when ’0’ =>
              hour_up<=’0’;
              min_up<=’0’;
              ret<=’0’;
       end case;
       end process;
   end behav;

Library ieee;
Use ieee.std_logic_1164.all;
Use ieee.std_logic_unsigned.all;
Entity alarm_hour is
Port(clk,reset:in std_logic;
    q:out std_logic_vector(7 downto 0));
end entity;
architecture behav of alarm_hour is
begin
process(clk)
variable r:std_logic_vector(7 downto 0);
begin
       if reset=’1’ then r:=″00000000″;
       elsif (rising_edge(clk)) then
          r:=r+1;
          if r(3 downto 0)=4 and r(7 downto 4)=2 then
              r(3 downto 0):=″0000″;r(7 downto 4):=″0000″;
          elsif r(3 downto 0)=9  then
              r(3 downto 0):=″0000″;r(7 downto 4):=r(7 downto 4)+1;
          end if;
       end if;
```

```
      q<=r;
end process;
end behav;

Library ieee;
Use ieee.std_logic_1164.all;
Use ieee.std_logic_unsigned.all;
Entity alarm_minute is
Port(clk,reset:in std_logic;
    q:out std_logic_vector(7 downto 0));
end entity;
architecture behav of alarm_minute is
begin
process(clk)
variable r:std_logic_vector(7 downto 0);
begin
      if reset='1' then r:="00000000";
      elsif (rising_edge(clk)) then
         r:=r+1;
         if r(3 downto 0)="1010"  then
            r(3 downto 0):="0000";
            if r(7 downto 4)/="0101" then
               r(7 downto 4):=r(7 downto 4)+1;
            elsif r(7 downto 4)="0101"then
               r(7 downto 4):="0000";
            end if;
         end if;
      end if;
      q<=r;
end process;
end behav;
```

（2）闹钟提醒模块

当计时模式下的时间和闹钟设置的时间一致时，闹钟提醒用户。

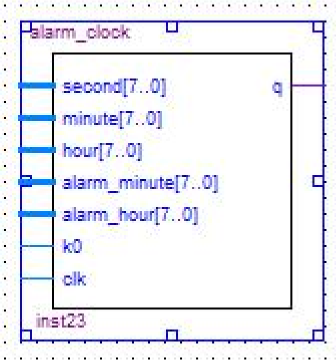

图 14 闹钟提醒模块

代码如下：

```
brary ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;
Entity alarm_clock is
Port( second:in std_logic_vector(7 downto 0);
       minute:in std_logic_vector(7 downto 0);
       hour:in std_logic_vector(7 downto 0);
        alarm_minute:in std_logic_vector(7 downto 0);
       alarm_hour:in std_logic_vector(7 downto 0);
        k0:in std_logic;
       clk:in std_logic;
       q:out std_logic);
end entity;
architecture behav of alarm_clock is
begin
   process(clk)
   begin
   if(clk’event and clk=’1’) then
   if(k0=’1’) then
            q<=’0’;
   elsif(hour=alarm_hour) then
       if(minute=alarm_minute) then
          if(second=″0000000″) then
              q<=’1’;
          end if;
       end if;
   end if;
   end if;
   end process;
end behav;
```

3.2.6 流水灯模块

当时钟到来时，led 灯 4、5、6 轮流点亮，形成流水灯，用于闹钟的提醒。

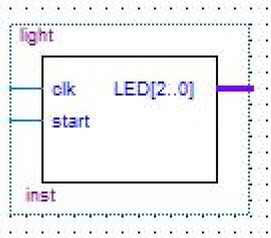

图 14 流水灯模块

代码如下：

```
LIBRARY IEEE;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_unsigned.all;
ENTITY light IS
  PORT(clk,start:IN std_logic;
      LED:OUT std_logic_vector(2 DOWNTO 0));
END light;
ARCHITECTURE Behav OF light IS
signal i:std_logic_vector(1 DOWNTO 0);
signal c: std_logic;
signal count:integer range 0 to 5000000;
BEGIN

process(clk)
   begin
   if (rising_edge(clk))then
          count<=count+1;
          if count=2500000 then
              c<='1';
          elsif count=5000000 then
              c<='0';
              count<=0;
```

```
         end if ;
     end if;
  end process;

    PROCESS(c,start)
    BEGIN
     IF(start='0')THEN
         LED<="000";i<="00";
     ELSIF(c'EVENT AND c='1')THEN
         IF(i=2)THEN
             i<="00";
         ELSE
             i<=i+'1';
         END IF;
         CASE i IS
             WHEN "00"=>LED<="110";
             WHEN "01"=>LED<="101";
             WHEN "10"=>LED<="011";
             WHEN others=>LED<="111";
         END CASE;
     END IF;
    END PROCESS;
  END Behav;
```

3.2.7 延时去抖模块

延时去抖模块根据按下按键时长判断此次输入是否有效，保证按键输入的有效性。

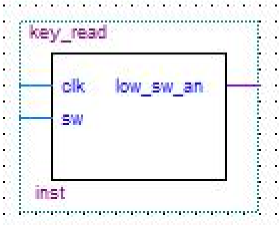

图 15 延时去抖模块

代码如下：

```
LIBRARY IEEE ;
USE IEEE.std_logic_1164.all ;
USE IEEE.std_logic_unsigned.all ;
ENTITY key_read IS
```

```
PORT (clk : IN std_logic ;
sw :IN std_logic ;
low_sw_an:out std_logic);
END ENTITY ;
ARCHITECTURE behav OF key_read IS
SIGNAL key_rst : std_logic ;
SIGNAL key_rst_an : std_logic ;
SIGNAL key_rst_r :std_logic ;
SIGNAL low_sw : std_logic ;
SIGNAL low_sw_r :std_logic ;
SIGNAL cnt : std_logic_vector (19 downto 0) ;
BEGIN
PROCESS (clk)
BEGIN
IF  clk’event and clk = ’1’  THEN
key_rst <= sw;
END IF ;
END PROCESS ;
PROCESS (clk)
BEGIN
IF clk’event and clk = ’1’ THEN
key_rst_r <= key_rst ;
END IF ;
END PROCESS ;
key_rst_an <= key_rst_r AND NOT key_rst ;
PROCESS (clk )
BEGIN
IF clk’event and clk = ’1’ THEN
IF key_rst_an = ’0’
THEN cnt <= cnt + ’1’ ;
 ELSE cnt <= (OTHERS => ’0’) ;
 END IF ;
END IF ;
END PROCESS ;
PROCESS (clk)
BEGIN
```

```
IF clk'event and clk = '1' THEN
IF cnt = "11111111111111111111" THEN
low_sw <= sw;
ELSE NULL ;
 END IF ;
END IF ;
END PROCESS ;
PROCESS (clk )
BEGIN
IF clk'event and clk = '1' THEN
low_sw_r <= low_sw ;
END IF ;
END PROCESS ;
low_sw_an <= low_sw_r AND NOT low_sw ;
END behav;
```

3.3 整机电路图

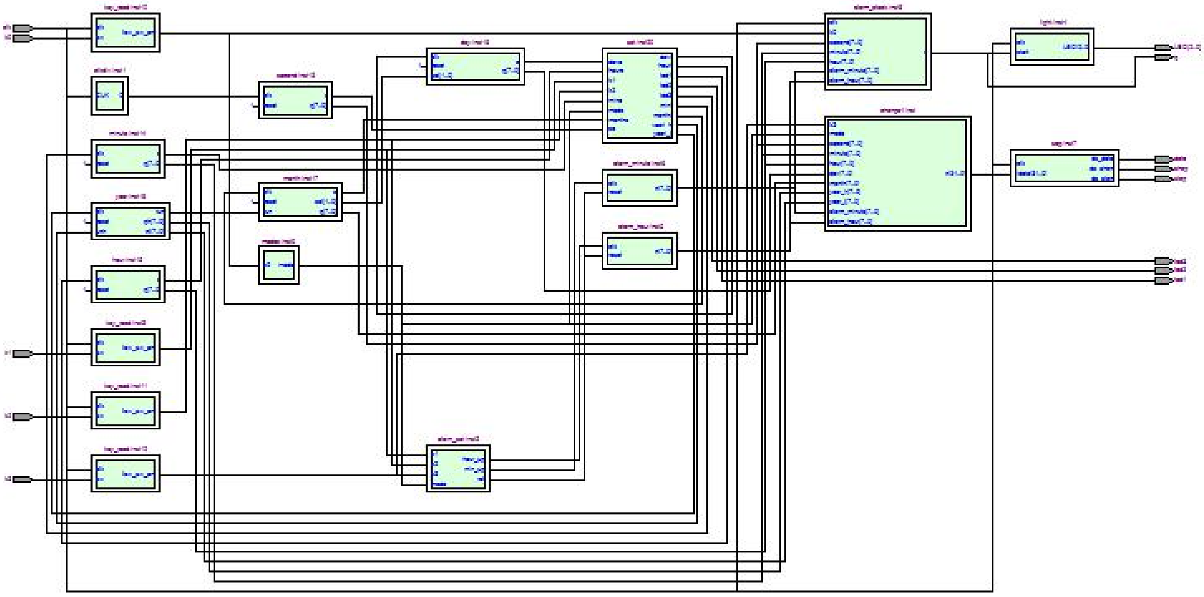

图 16 整机电路图

四、结果分析

4.1 仿真结果及分析

4.1.1 秒计数器模块：

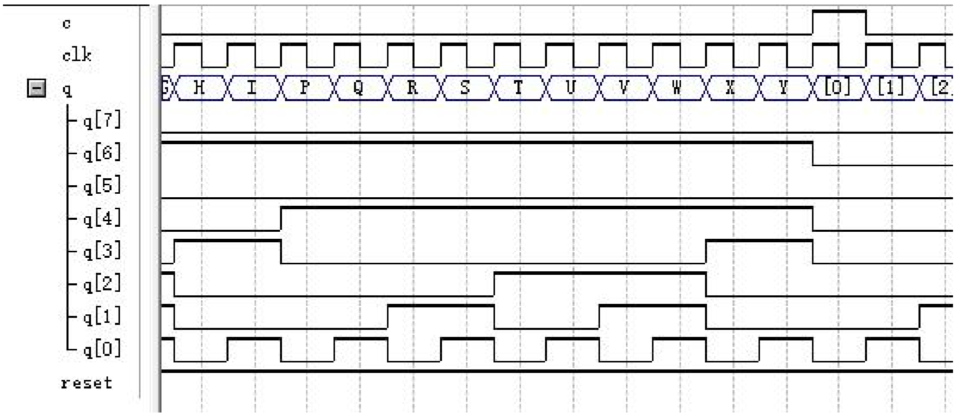

图 17 秒计数器模块仿真图

由上图可以看出，在秒计数器模块中每计数 60 次产生一个进位，设计正确。

4.1.2 分计数器模块：

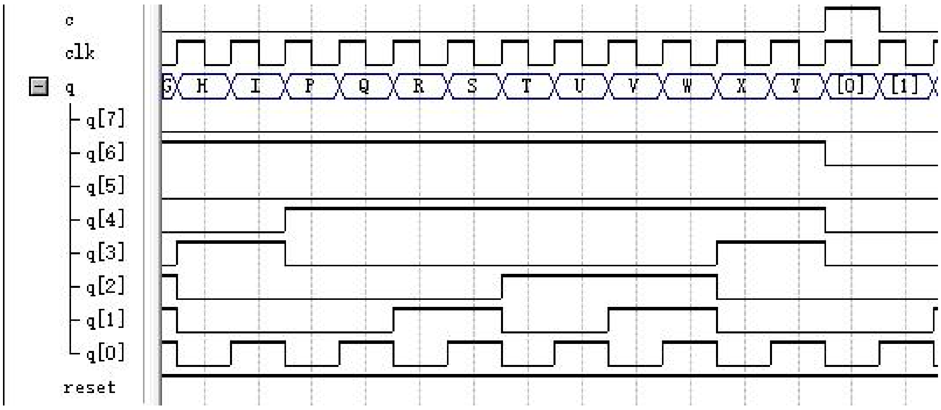

图 18 分计数器模块仿真图

由上图可以看出，在分计数器模块中每计数 60 次产生一个进位，设计正确。

4.1.3 时计数器模块：

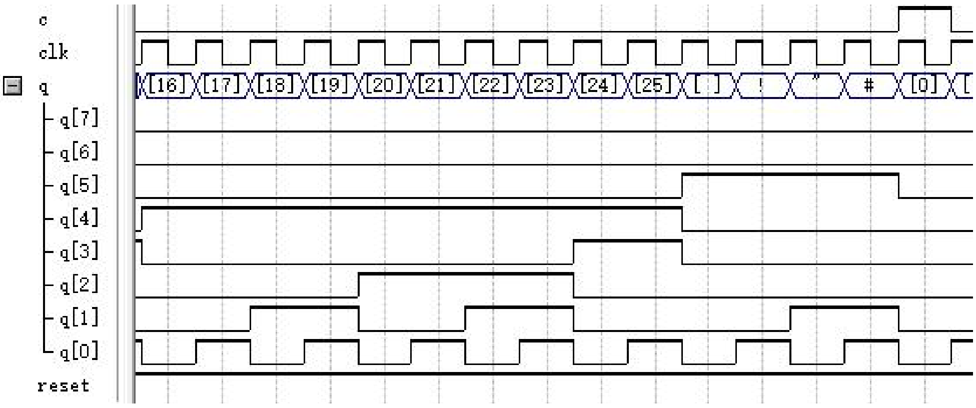

图 19 时计数器模块仿真图

由上图可以看出，在时计数器模块中每计数 24 次产生一个进位，设计正确。

4.1.4 日计数器模块

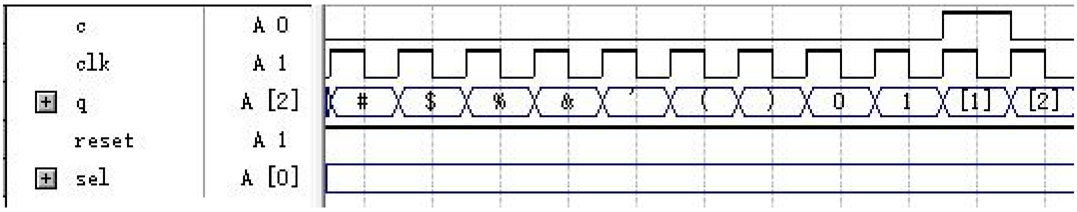

图 20 日计数器模板仿真图（大月）

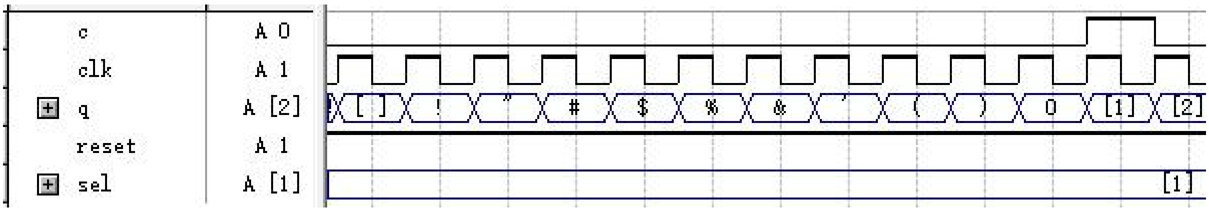

图 21 日计数器模板仿真图（小月）

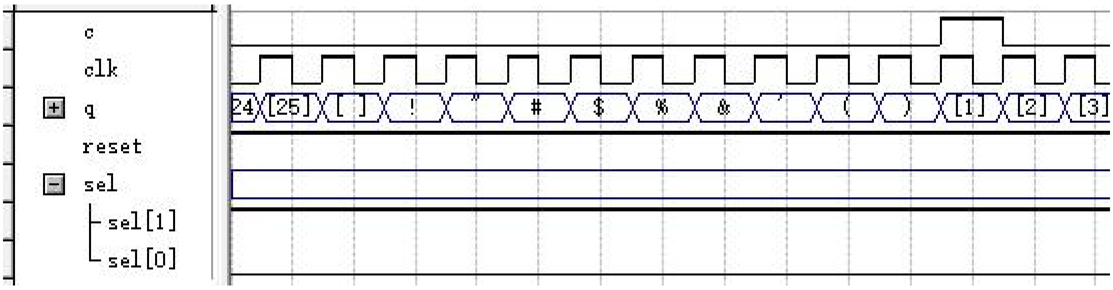

图 21 日计数器模板仿真图（闰年 2 月）

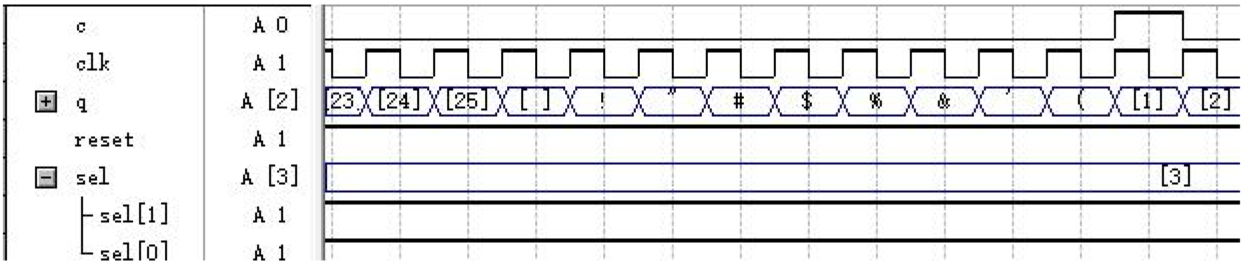

图 23 日计数器模板仿真图（平年 2 月）

由上面仿真图可以看出，根据不同 sel 输入，日计数器有不同的进制，来表示大月、小月、闰年 2 月、平年 2 月的天数。

4.1.5 月计数器模块

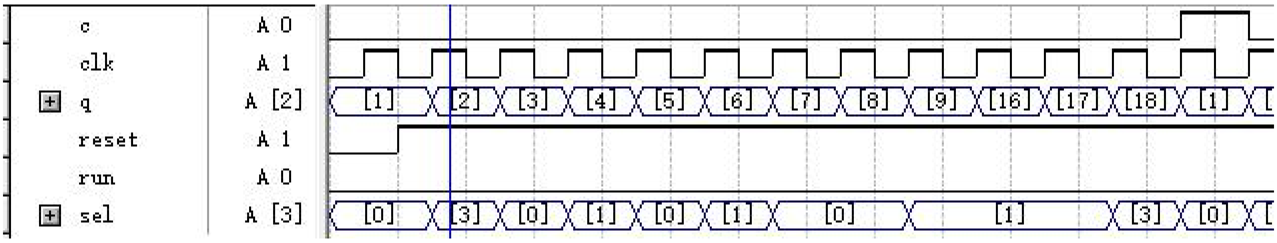

图 24 月计数器模板仿真图（平年）

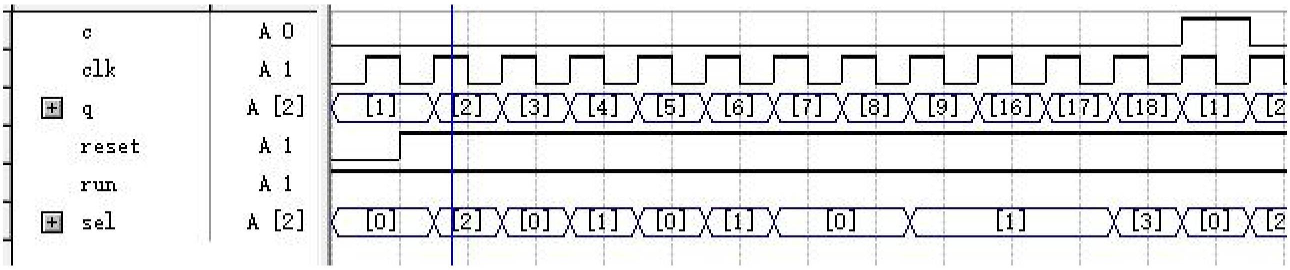

图 25 月计数器模板仿真图（闰年）

由上面仿真图可以看出，月计数器为 12 进制计数器，同时，不同月份的 sel 输出不同，run 输入影响了 2 月份 sel 输出，所以设计正确。

4.1.6 年计数器模块

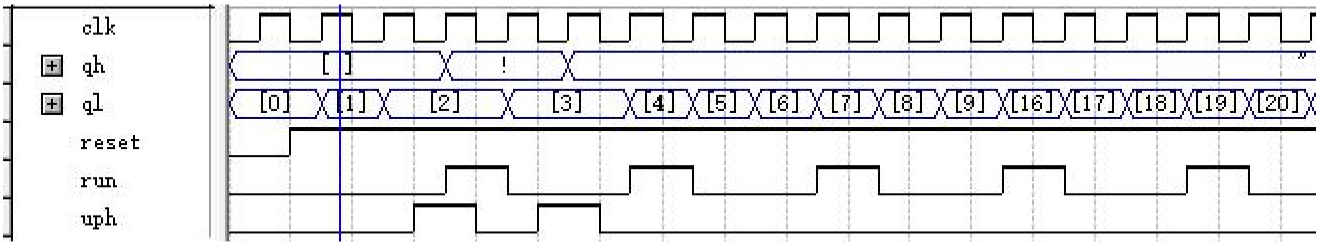

图 26 年计数器模板仿真图

由上图可以看出，uph 控制年高位还是年低位加 1，当年低位满 100 会向年高位进 1，设计正确。

4.1.7 设置模块

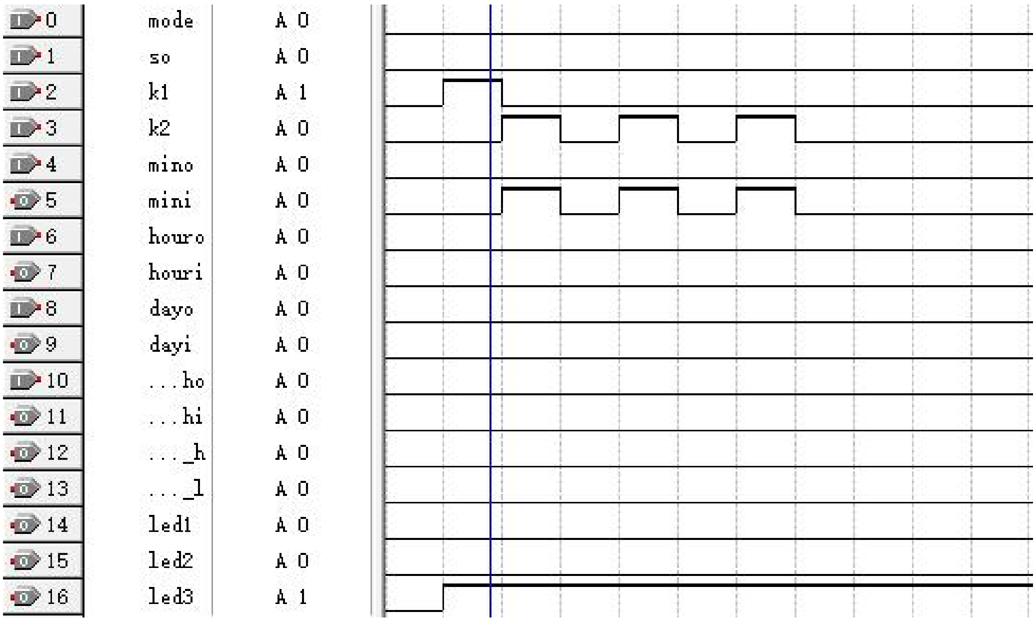

图 27 设置分钟仿真图

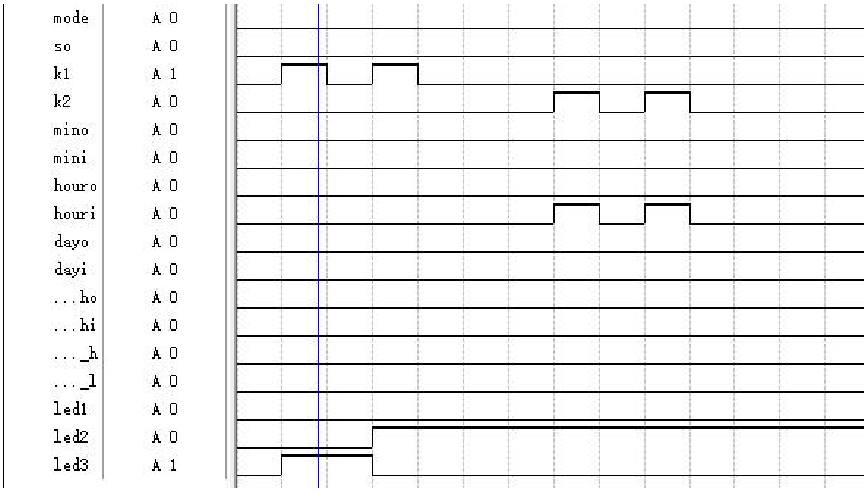

图 28 设置小时仿真图

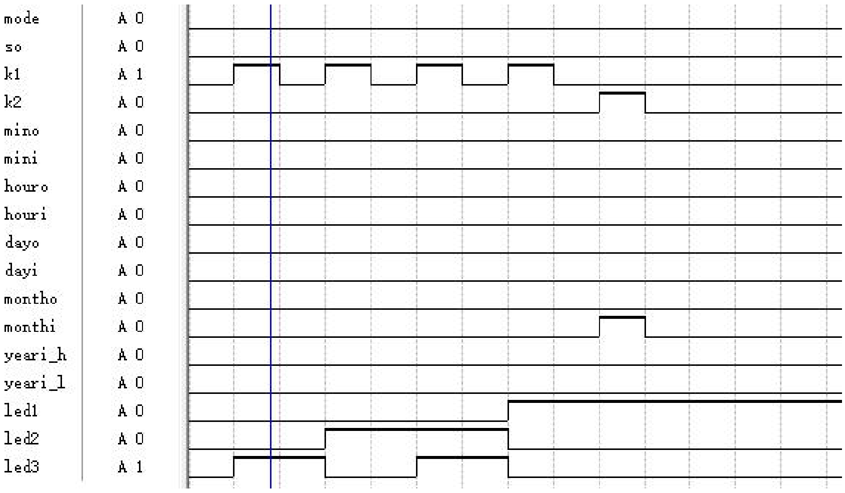

图 29 设置月份仿真图

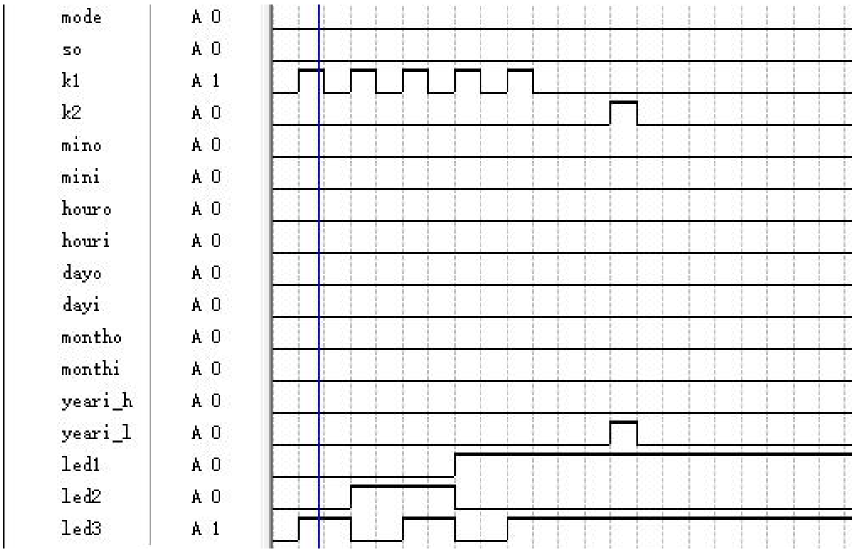

图 30 设置年低位仿真图

由上面的仿真图可以看出，k1 能选择设置的时间，k2 增加控制的时间，同时 led 也会根据设置的时间而改变，设计成功。

4.1.8 闹钟模块

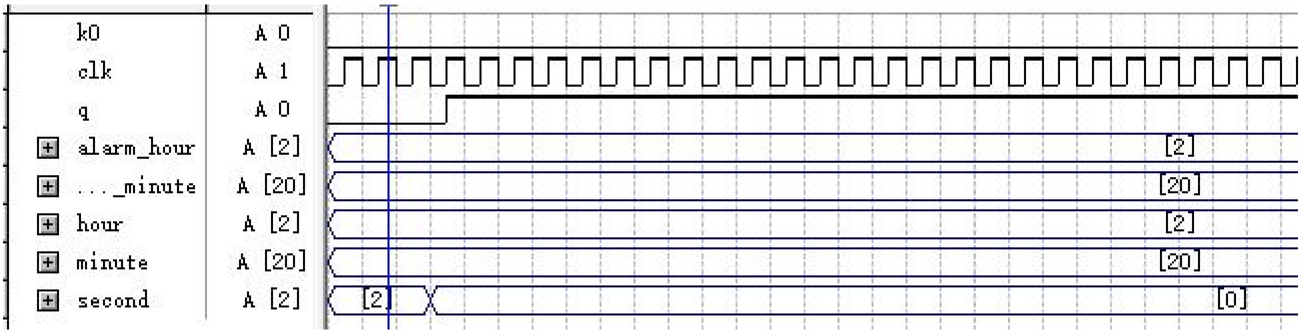

图 31 闹钟模式仿真图

由上面仿真图可以看出，当时钟上升沿到来，闹钟设定时间和计数时间相同时，闹钟模块有输出，设计正确。

4.2 实验结果

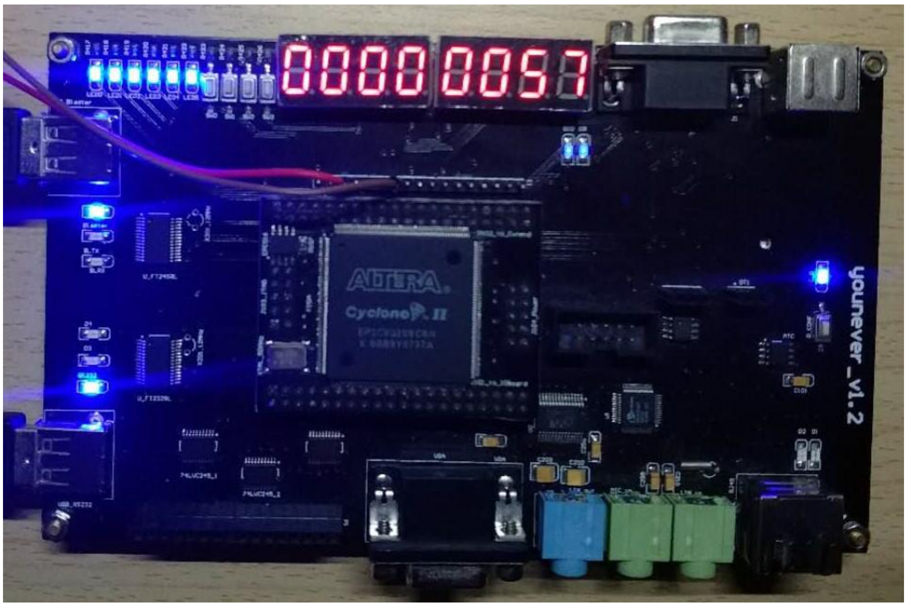

图 32 时分秒计时模式图

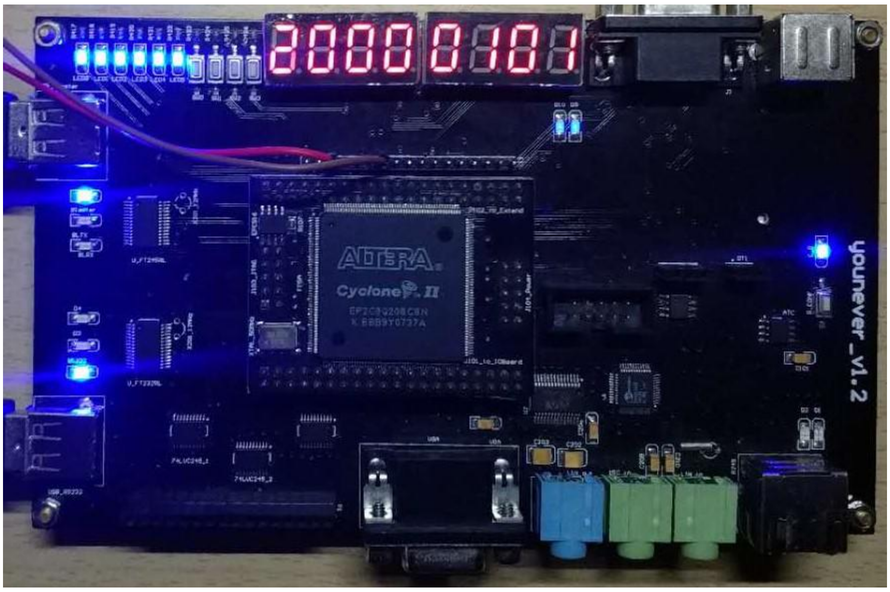

图 33 年月日计时模式图

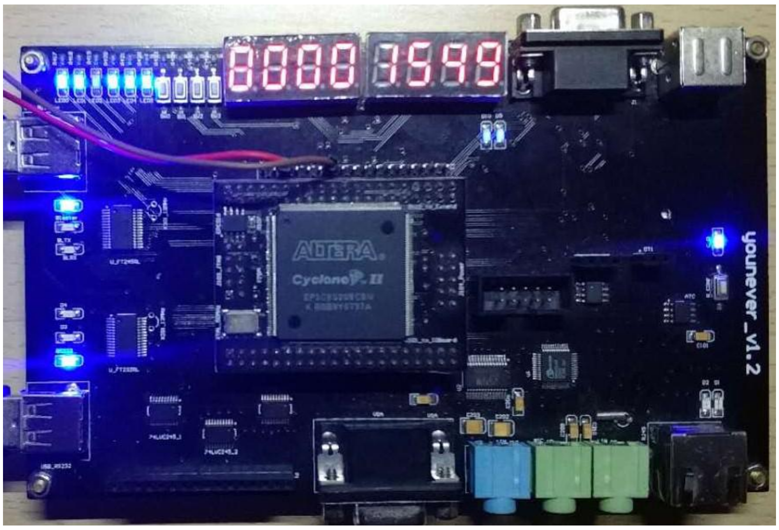

图 34 设置分钟图

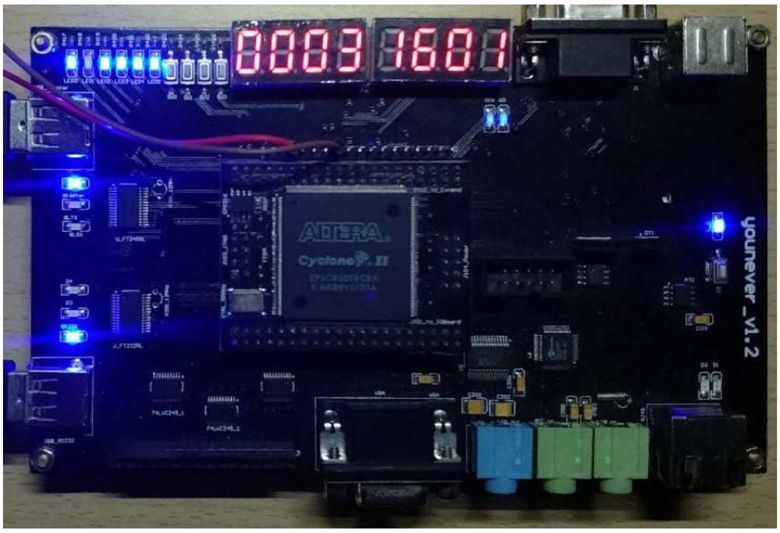

图 35 设置小时图

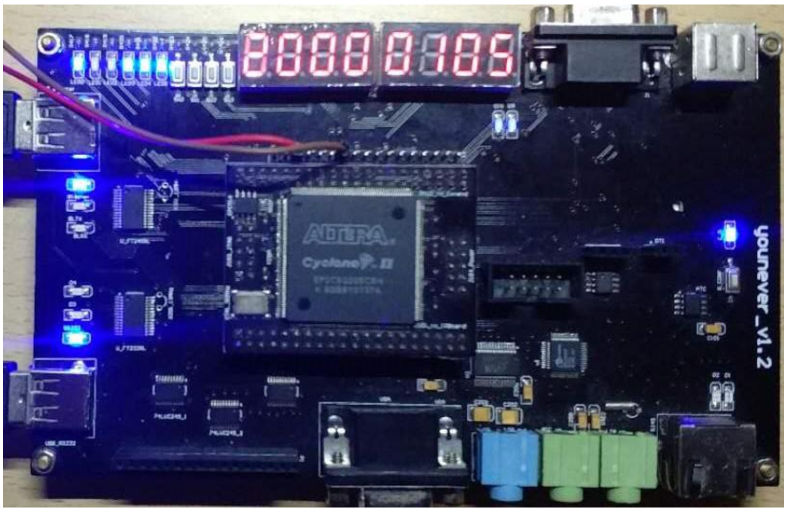

图 36 设置日图

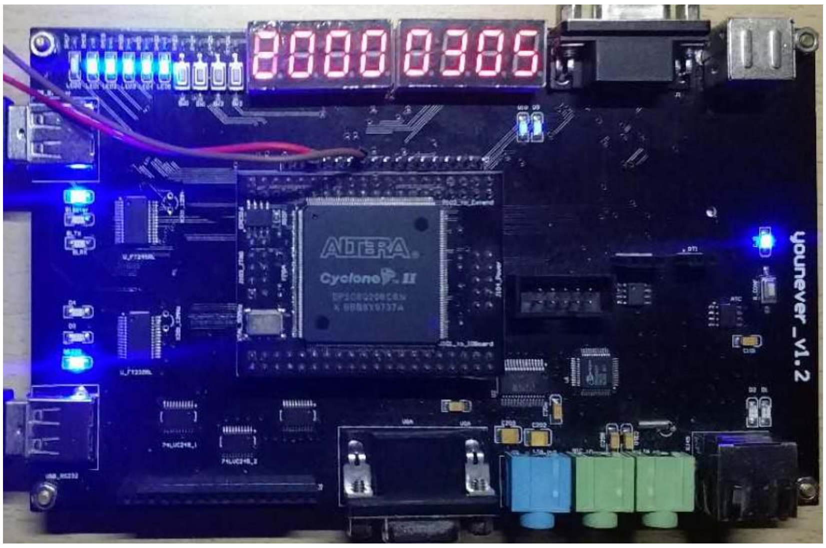

图 37 设置月份图

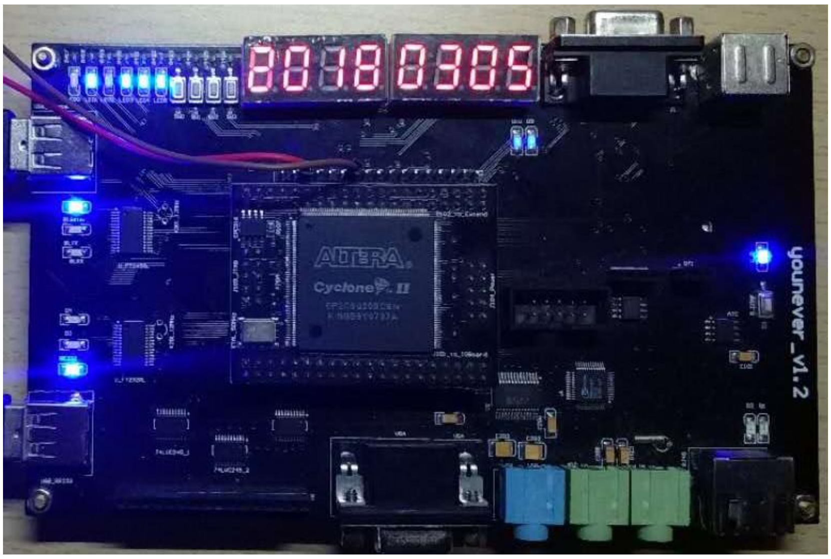

图 38 设置年低位图

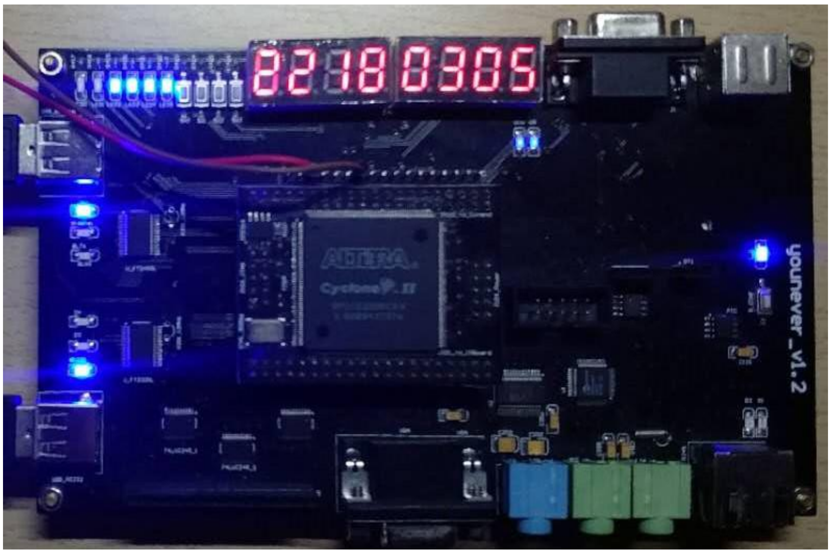

图 39 设置年高位图

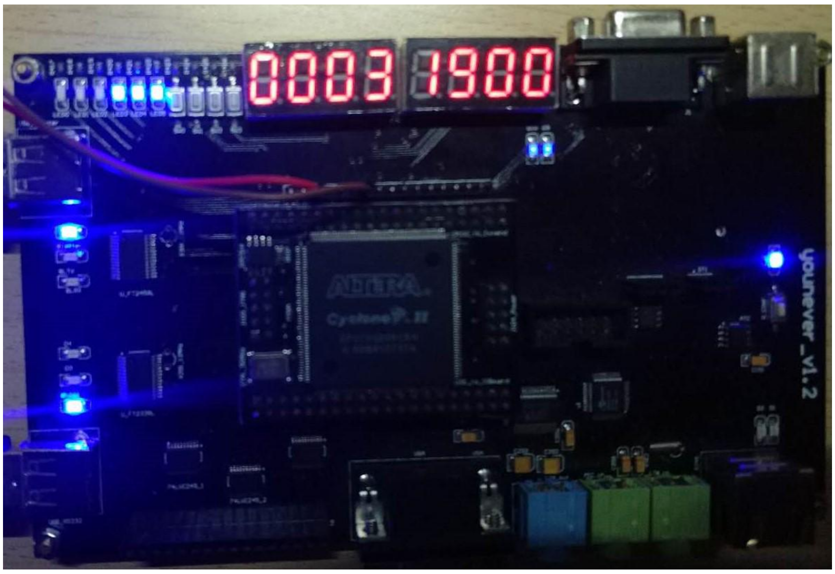

图 40 设置闹钟图

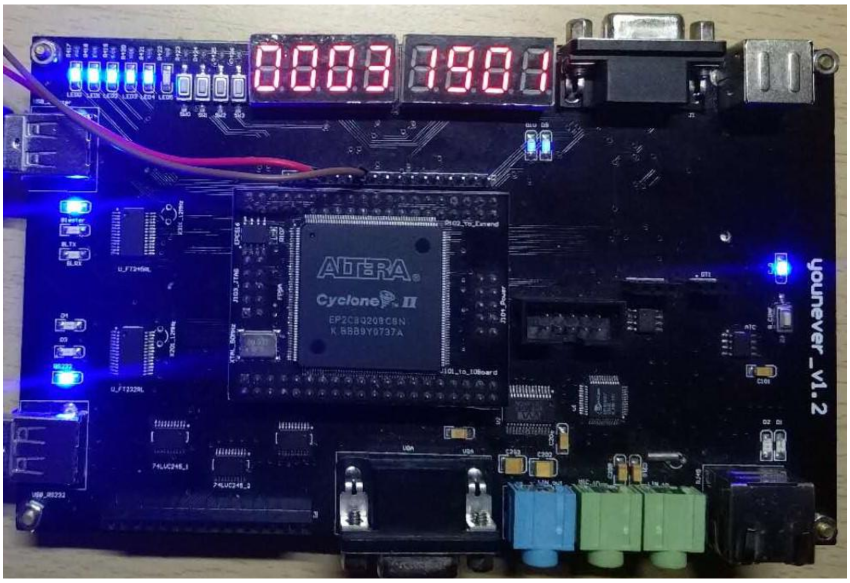

图 41 闹钟提醒图

五、课程设计总结

通过这次的课程设计我们受益良多，学习到了很多在课堂上没有发现的知识。因为编程是动手实践的学科，通过这样的课程设计，不仅有助于我们巩固课堂上学习到的 VHDL 的理论知识，同时也锻炼了我们把学习到的理论知识用到生活中的能力。

这次万年历的设计，采用了模块化的设计思想，所以才设计过程中，整体思路清晰，后期修改代码，增加新的模块也相对容易，让我们体会到这种编程方式的优点。同时，VHDL 这种软硬件相结合的编程方式，有利于自顶向下地进行设计，而不用考虑太多底层的实现，提高了我们开发的效率。

在测试过程中，天数计数器上出现了比较多的问题。因为存在 4 种情况，在用计数是很容易混淆出错。我们的一个错误就是，闰年有 29 天，因为计数用的变量是 16 位的，而显示的是 10 进制的数，在设计 29 进制计数器时，把判断的阈值设为 30，但实际情况不会出现 0，所以无法在计数到 30 时置零进位。判断的阈值应为 3A。

在设计的过程中，闹钟的时间判定模块原本采用时、分与闹钟的时、分触发进程，结果闹钟会在设定的时间响之后依旧会再响两次。之后改用系统时钟上升沿触发便解决了问题。