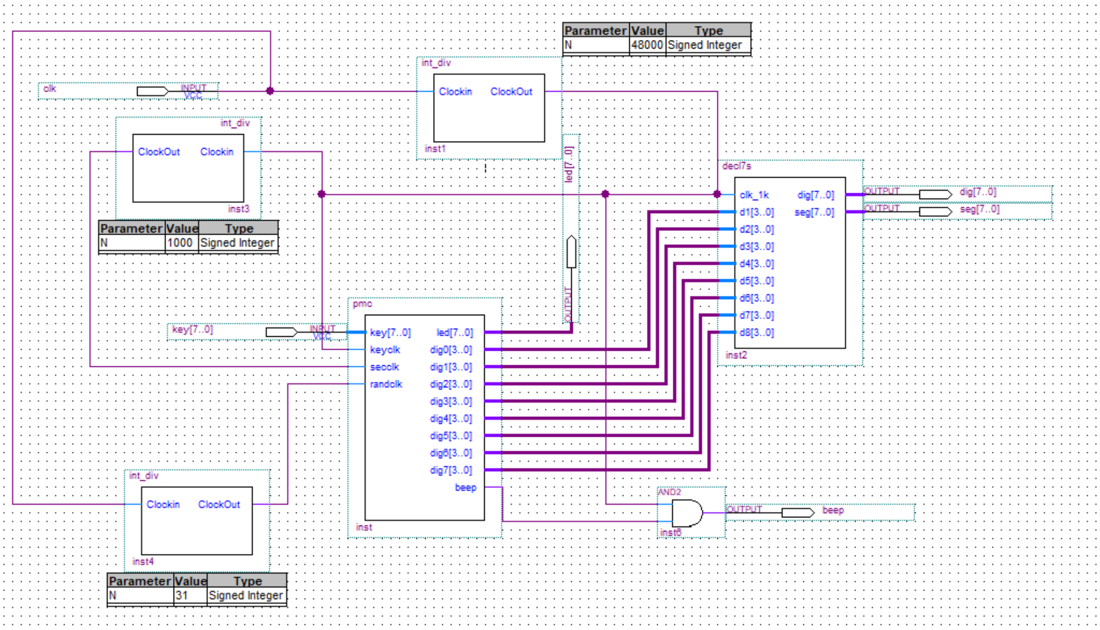
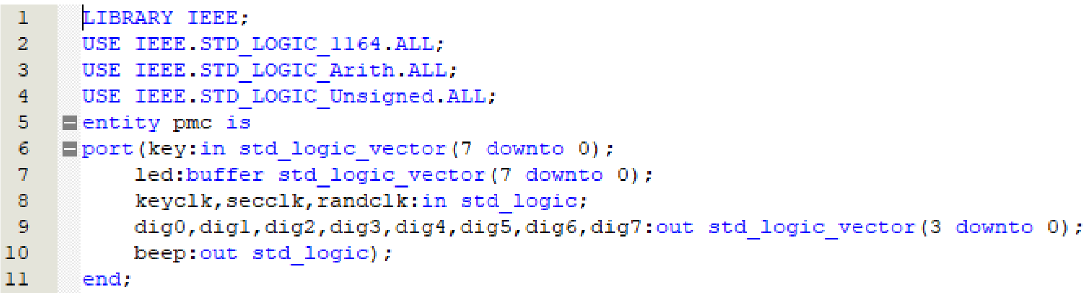
设计过程

（1）设计总体结构：分频器和数码管显示部分使用已有模块，蜂鸣器使用数码管的 1kHz 刷新时钟兼做发声频率，使用一个与门进行控制，主要的功能通过一个新建的模块进行实现。

顶层模块视图如下：

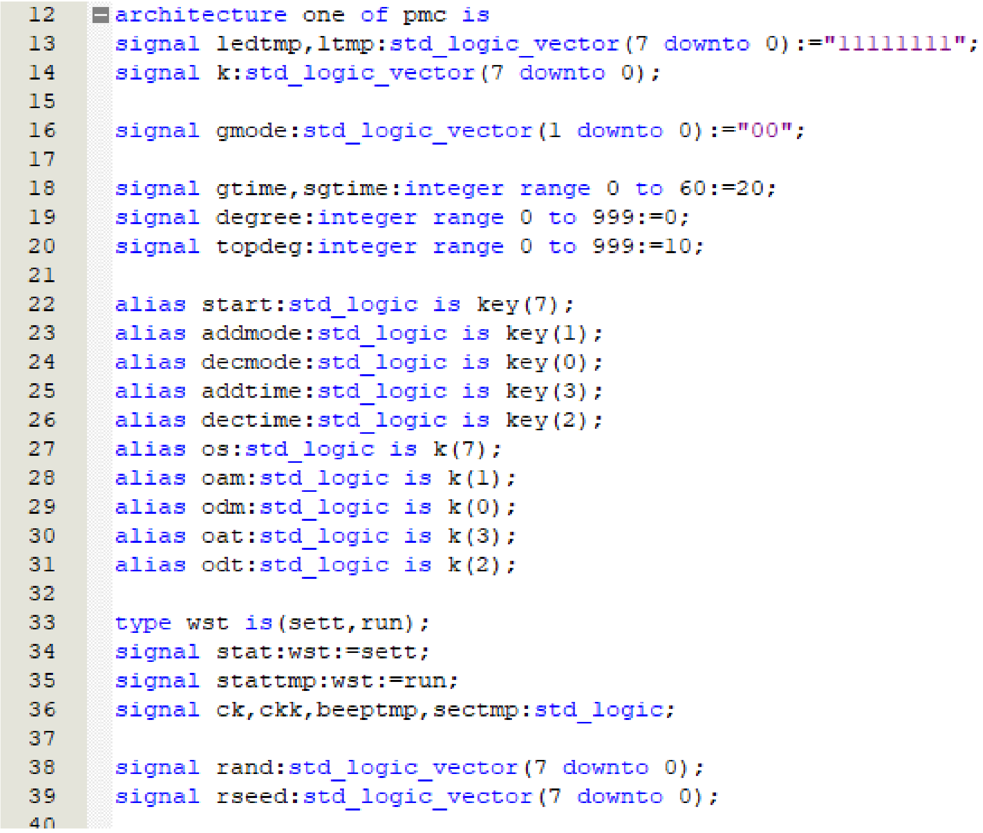


（2）根据所需模块端口新建一个实体



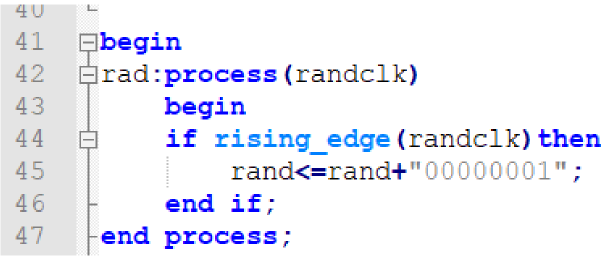
```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_Arith.ALL;
USE IEEE.STD_LOGIC_Unsigned.ALL;
entity pmc is
port(key:in std_logic_vector(7 downto 0);
    led:buffer std_logic_vector(7 downto 0);
    keyclk,secclk,randclk:in std_logic;
    dig0,dig1,dig2,dig3,dig4,dig5,dig6,dig7:out std_logic_vector(3 downto 0);
    beep:out std_logic);
end;
```

（3）所需的信号定义与别名定义



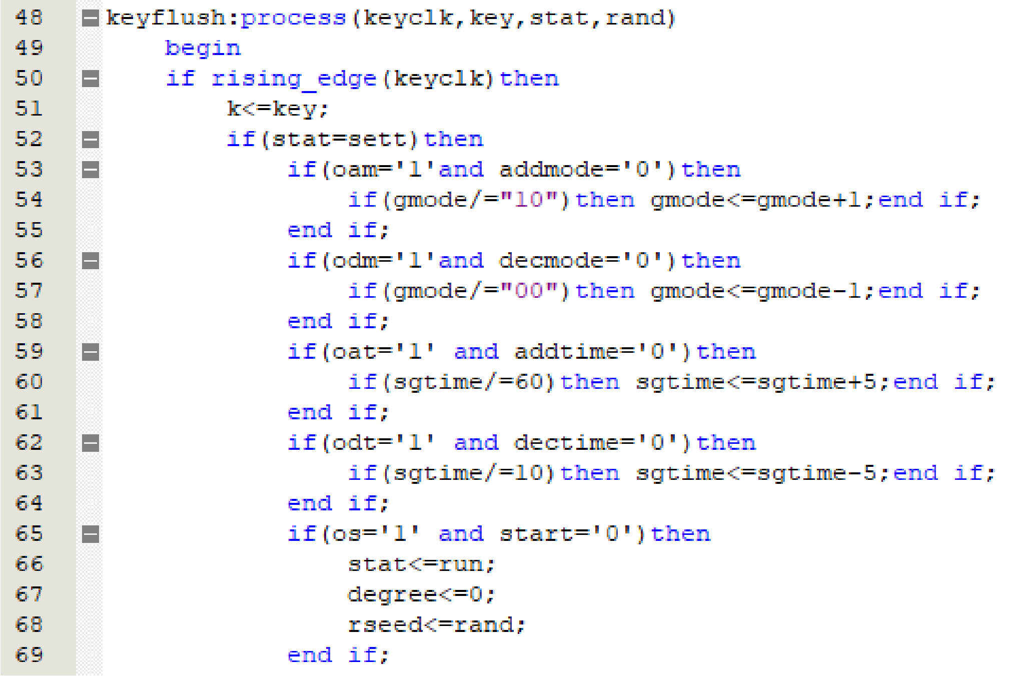
```
architecture one of pmc is
signal ledtmp,ltmp:std_logic_vector(7 downto 0):="11111111";
signal k:std_logic_vector(7 downto 0);

signal gmode:std_logic_vector(1 downto 0):="00";

signal gtime,sgtime:integer range 0 to 60:=20;
signal degree:integer range 0 to 999:=0;
signal topdeg:integer range 0 to 999:=10;

alias start:std_logic is key(7);
alias addmode:std_logic is key(1);
alias decmode:std_logic is key(0);
alias addtime:std_logic is key(3);
alias dectime:std_logic is key(2);
alias os:std_logic is k(7);
alias oam:std_logic is k(1);
alias odm:std_logic is k(0);
alias oat:std_logic is k(3);
alias odt:std_logic is k(2);

type wst is(sett,run);
signal stat:wst:=sett;
signal stattmp:wst:=run;
signal ck,ckk,beeptmp,sectmp:std_logic;

signal rand:std_logic_vector(7 downto 0);
signal rseed:std_logic_vector(7 downto 0);
```

（3）随机数计数器，用作随机数的生成源



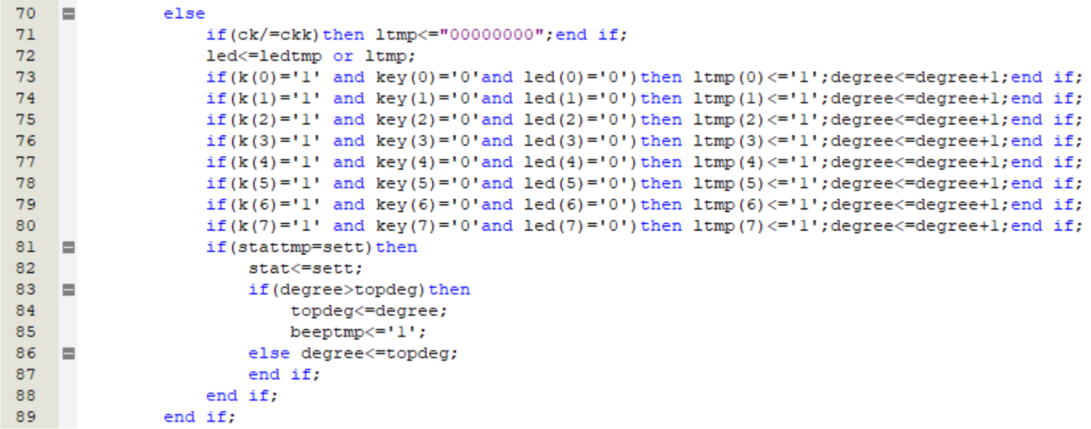
```
begin
rad:process(randclk)
    begin
    if rising_edge(randclk) then
        rand<=rand+"00000001";
    end if;
end process;
```

（4）通过按键对游戏模式与游戏时间进行设置，以及开始游戏按钮的实现



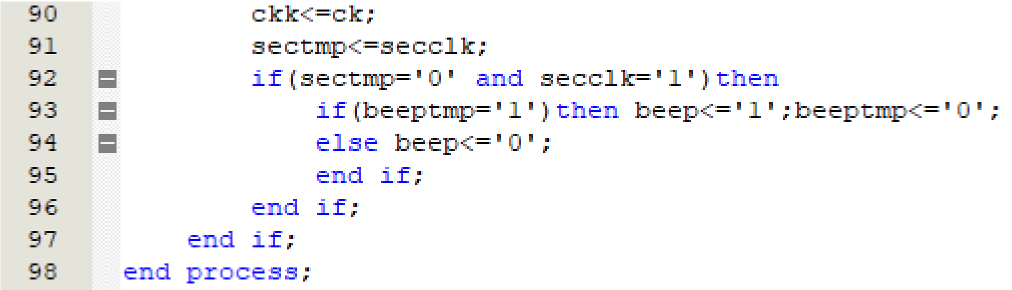
```vhdl
keyflush:process(keyclk,key,stat,rand)
    begin
    if rising_edge(keyclk)then
        k<=key;
        if(stat=sett)then
            if(oam='1'and addmode='0')then
                if(gmode/="10")then gmode<=gmode+1;end if;
            end if;
            if(odm='1'and decmode='0')then
                if(gmode/="00")then gmode<=gmode-1;end if;
            end if;
            if(oat='1' and addtime='0')then
                if(sgtime/=60)then sgtime<=sgtime+5;end if;
            end if;
            if(odt='1' and dectime='0')then
                if(sgtime/=10)then sgtime<=sgtime-5;end if;
            end if;
            if(os='1' and start='0')then
                stat<=run;
                degree<=0;
                rseed<=rand;
            end if;
```

（5）实现游戏中按钮按下的反馈以及游戏结束后判断得分是否打破记录



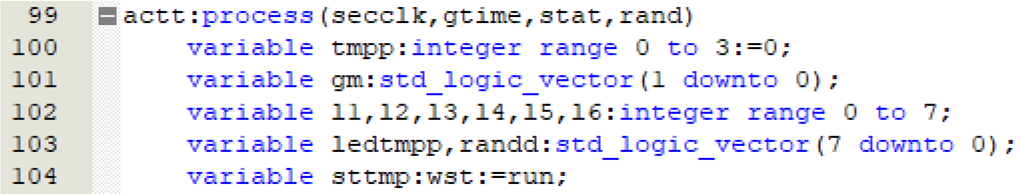
```vhdl
        else
            if(ck/=ckk)then ltmp<="00000000";end if;
            led<=ledtmp or ltmp;
            if(k(0)='1' and key(0)='0'and led(0)='0')then ltmp(0)<='1';degree<=degree+1;end if;
            if(k(1)='1' and key(1)='0'and led(1)='0')then ltmp(1)<='1';degree<=degree+1;end if;
            if(k(2)='1' and key(2)='0'and led(2)='0')then ltmp(2)<='1';degree<=degree+1;end if;
            if(k(3)='1' and key(3)='0'and led(3)='0')then ltmp(3)<='1';degree<=degree+1;end if;
            if(k(4)='1' and key(4)='0'and led(4)='0')then ltmp(4)<='1';degree<=degree+1;end if;
            if(k(5)='1' and key(5)='0'and led(5)='0')then ltmp(5)<='1';degree<=degree+1;end if;
            if(k(6)='1' and key(6)='0'and led(6)='0')then ltmp(6)<='1';degree<=degree+1;end if;
            if(k(7)='1' and key(7)='0'and led(7)='0')then ltmp(7)<='1';degree<=degree+1;end if;
            if(stattmp=sett)then
                stat<=sett;
                if(degree>topdeg)then
                    topdeg<=degree;
                    beeptmp<='1';
                else degree<=topdeg;
                end if;
            end if;
        end if;
```

（6）实现得分打破记录后，蜂鸣器的鸣叫



```vhdl
        ckk<=ck;
        sectmp<=secclk;
        if(sectmp='0' and secclk='1')then
            if(beeptmp='1')then beep<='1';beeptmp<='0';
            else beep<='0';
            end if;
        end if;
    end if;
end process;
```

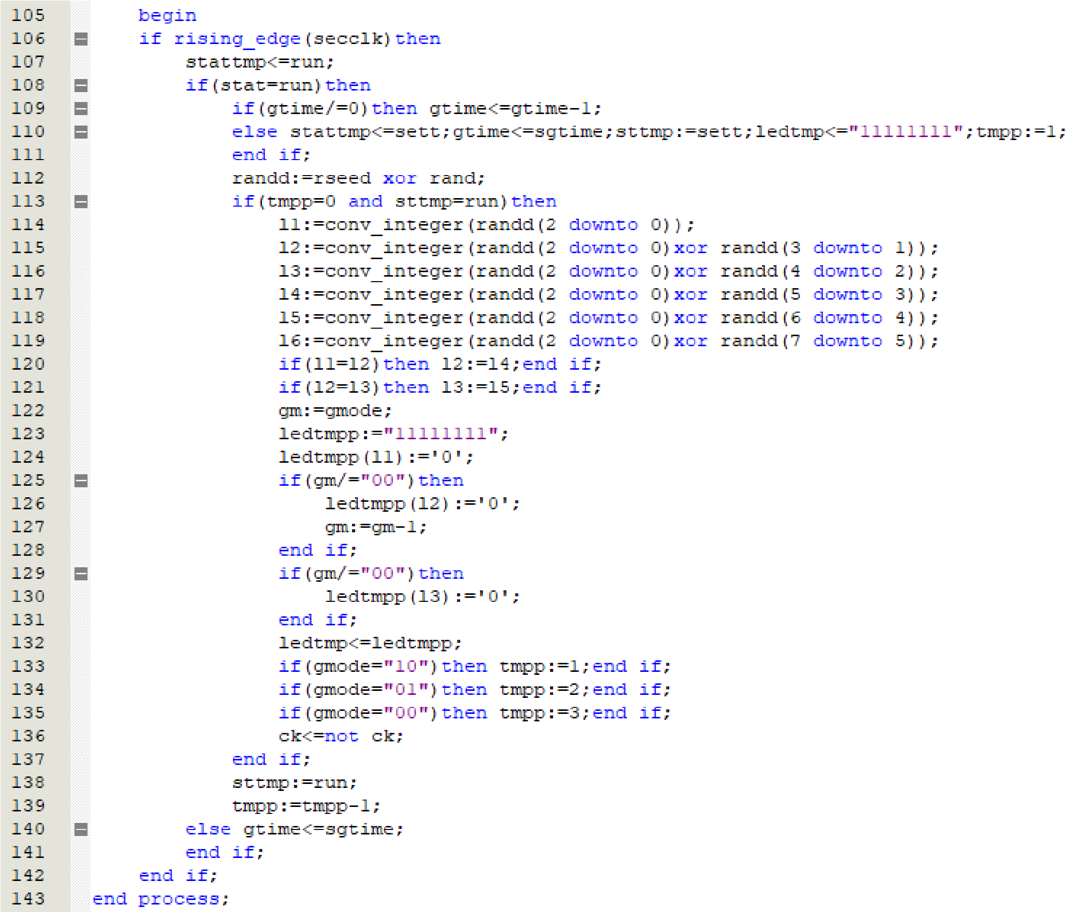
（7）变量定义



```vhdl
actt:process(secclk,gtime,stat,rand)
    variable tmpp:integer range 0 to 3:=0;
    variable gm:std_logic_vector(1 downto 0);
    variable l1,l2,l3,l4,l5,l6:integer range 0 to 7;
    variable ledtmpp,randd:std_logic_vector(7 downto 0);
    variable sttmp:wst:=run;
```

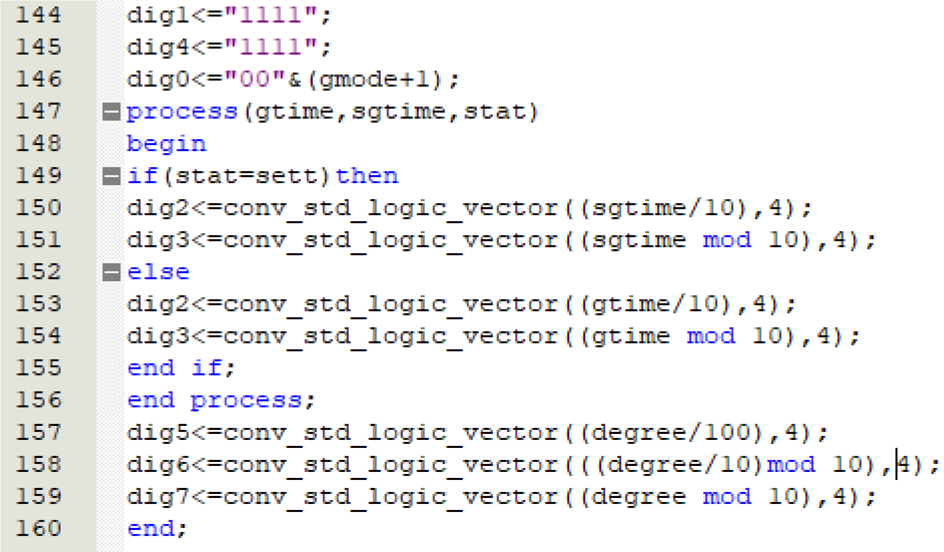
（8）通过开始游戏按键按下时所产生的随机数种子，与随机数计数器当前值进行异或运算，

再依据所得结果及游戏模式点亮相应 LED 灯，同时进行游戏时间的计时



```
    begin
    if rising_edge(secclk)then
        stattmp<=run;
        if(stat=run)then
            if(gtime/=0)then gtime<=gtime-1;
            else stattmp<=sett;gtime<=sgtime;sttmp:=sett;ledtmp<="11111111";tmpp:=1;
            end if;
            randd:=rseed xor rand;
            if(tmpp=0 and sttmp=run)then
                l1:=conv_integer(randd(2 downto 0));
                l2:=conv_integer(randd(2 downto 0)xor randd(3 downto 1));
                l3:=conv_integer(randd(2 downto 0)xor randd(4 downto 2));
                l4:=conv_integer(randd(2 downto 0)xor randd(5 downto 3));
                l5:=conv_integer(randd(2 downto 0)xor randd(6 downto 4));
                l6:=conv_integer(randd(2 downto 0)xor randd(7 downto 5));
                if(l1=l2)then l2:=l4;end if;
                if(l2=l3)then l3:=l5;end if;
                gm:=gmode;
                ledtmpp:="11111111";
                ledtmpp(l1):='0';
                if(gm/="00")then
                    ledtmpp(l2):='0';
                    gm:=gm-1;
                end if;
                if(gm/="00")then
                    ledtmpp(l3):='0';
                end if;
                ledtmp<=ledtmpp;
                if(gmode="10")then tmpp:=1;end if;
                if(gmode="01")then tmpp:=2;end if;
                if(gmode="00")then tmpp:=3;end if;
                ck<=not ck;
            end if;
            sttmp:=run;
            tmpp:=tmpp-1;
        else gtime<=sgtime;
        end if;
    end if;
end process;
```

（9）进程外部数码管显示输出部分



```
dig1<="1111";
dig4<="1111";
dig0<="00"&(gmode+1);
process(gtime,sgtime,stat)
begin
if(stat=sett)then
dig2<=conv_std_logic_vector((sgtime/10),4);
dig3<=conv_std_logic_vector((sgtime mod 10),4);
else
dig2<=conv_std_logic_vector((gtime/10),4);
dig3<=conv_std_logic_vector((gtime mod 10),4);
end if;
end process;
dig5<=conv_std_logic_vector((degree/100),4);
dig6<=conv_std_logic_vector(((degree/10)mod 10),4);
dig7<=conv_std_logic_vector((degree mod 10),4);
end;
```