# 课程设计--棋钟实验报告

学生姓名：周子涵 学号2018011218014

一 项目要求

自行设计一个棋钟，功能如下：

1）棋类比赛中每步棋有时间限制。假设A、B两位棋手比赛，当A棋手落子后按下A键，此时B棋手的秒表开始倒计时，A棋手的秒表恢复为倒计时初始值；当B棋手落子后按下B键，A棋手的秒表开始倒计时，B棋手的秒表恢复为倒计时初始值。

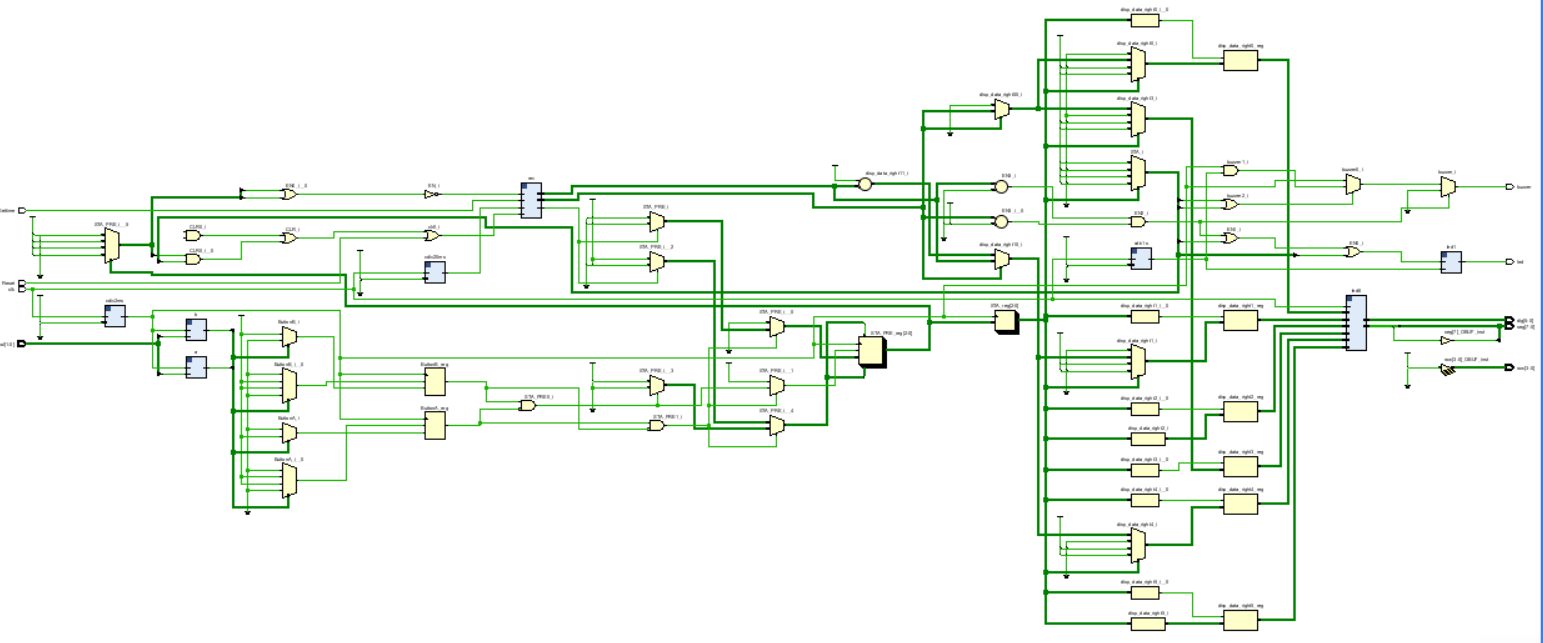
2）倒计时初始值可设置为10秒或30秒两档。

3）当倒计时至5秒内时，蜂鸣器每秒响一下或led灯 闪烁1下。

4）当某选手超时判负时，蜂鸣器长响，led灯持续闪烁，数码管显示出获胜的棋手编号（A或B）。

5）增加其他功能可自由发挥，如能实现有加分。

二 实现方案框图



三 实验情况描述

1. 通过计数器分频实现了1s的数码管变动，实现了倒计时功能
2. 通过适当的分频实现了按键消抖
3. 设置开关来进行10s档和30s档的切换
4. 将数码管显示与蜂鸣器联系起来，当数码管小于5的时候蜂鸣器开始响，实现了蜂鸣器提醒的功能
5. 实现了按钮复位的功能

四 完整程序

1. top模块：

module top(

input *clk*,Settime,Reset,

input [1:0]*col*,

output [3:0] *row*,

output [7:0] *seg*,

output [5:0] *dig*,

output *led*,buzzer

    );

 assign row=1110;

//clk

wire clk\_2ms,clk\_1s,clk\_20ms;

clk\_div cdiv2ms(clk,clk\_2ms,50000);

clk\_div cdiv1s(clk,clk\_1s,25000000);

clk\_div cdiv20ms(clk,clk\_20ms,500000);

reg ButtonA,ButtonB;

//cnt

wire [3:0]Q\_10,Q\_3;

wire cy\_10,EN,CLR;

revcounter rev(clk\_20ms,CLR|Reset,EN,Q\_10,Q\_3,cy\_10,Settime);//30?10

wire R,S;

 ajxd  a(col[0],clk\_2ms,R);

 ajxd  b(col[1],clk\_2ms,S);

always@(posedge clk\_2ms)

begin

    case({R,S})

    2'b00: begin

           ButtonA<=ButtonA;

           ButtonB<=ButtonB;

            end

    2'b01: begin

           ButtonA<=0;

           ButtonB<=1;

            end

    2'b10: begin

           ButtonA<=1;

           ButtonB<=0;

            end

     2'b11: begin

           ButtonA<=ButtonA;

           ButtonB<=ButtonB;

            end

     endcase

end

//FSM

reg [2:0]STA,STA\_PRE;

parameter A\_WAIT=3'b000;

parameter B\_WAIT=3'b001;

parameter A\_VIC=3'b100;

parameter B\_VIC=3'b101;

parameter Secure=3'b111;

always@(posedge clk\_2ms)

begin

if(ButtonA&(~ButtonB))

  begin  if(cy\_10)

        STA\_PRE<=B\_VIC;

    else STA\_PRE<=A\_WAIT;

  end

else if(ButtonB&(~ButtonA))

  begin

    if(cy\_10)

        STA\_PRE<=A\_VIC;

    else STA\_PRE<=B\_WAIT;

   end

end

always@(posedge clk\_2ms)

begin

    STA<=STA\_PRE;

end

assign CLR=((STA\_PRE==A\_WAIT)&&(STA==B\_WAIT))||((STA\_PRE==B\_WAIT)&&(STA==A\_WAIT));

assign EN=(~((STA\_PRE==A\_VIC)||(STA\_PRE==B\_VIC)));

//display

reg [3:0]disp\_data\_right0,disp\_data\_right1,disp\_data\_right2,disp\_data\_right3,disp\_data\_right4,disp\_data\_right5;

always@(STA)

begin

case (STA)

    A\_WAIT:  begin

               disp\_data\_right0=0;

               disp\_data\_right1=0;

               disp\_data\_right2=4'hb;

               disp\_data\_right3=(Q\_10==10)?0:Q\_10;

               disp\_data\_right4=(Q\_10==10)?(Q\_3+1):Q\_3;

               disp\_data\_right5=4'ha;

              end

    B\_WAIT:  begin

               disp\_data\_right3=0;

               disp\_data\_right4=0;

               disp\_data\_right2=4'hb;

               disp\_data\_right0=(Q\_10==10)?0:Q\_10;

               disp\_data\_right1=(Q\_10==10)?(Q\_3+1):Q\_3;

               disp\_data\_right5=4'ha;

              end

    A\_VIC:  begin

               disp\_data\_right3=4'ha;

               disp\_data\_right4=4'ha;

               disp\_data\_right2=4'ha;

               disp\_data\_right0=4'ha;

               disp\_data\_right1=4'ha;

               disp\_data\_right5=4'ha;

              end

    B\_VIC:  begin

               disp\_data\_right3=4'hb;

               disp\_data\_right4=4'hb;

               disp\_data\_right2=4'hb;

               disp\_data\_right0=4'hb;

               disp\_data\_right1=4'hb;

               disp\_data\_right5=4'hb;

              end

endcase

end

dynamic\_led6 led6(

disp\_data\_right0,

disp\_data\_right1,

disp\_data\_right2,

disp\_data\_right3,

disp\_data\_right4,

disp\_data\_right5,

 clk,

seg,

dig

    );

//led

led\_toggle led1(

 clk\_1s,

((Q\_3==0)&&(Q\_10<=5))||(STA==A\_VIC)||(STA==B\_VIC),

led

    );

//buzzer

assign buzzer=((Q\_3==0)&&(Q\_10<=5))?((STA==A\_VIC)||(STA==B\_VIC)?clk\_2ms:clk\_2ms&clk\_1s):0;

endmodule