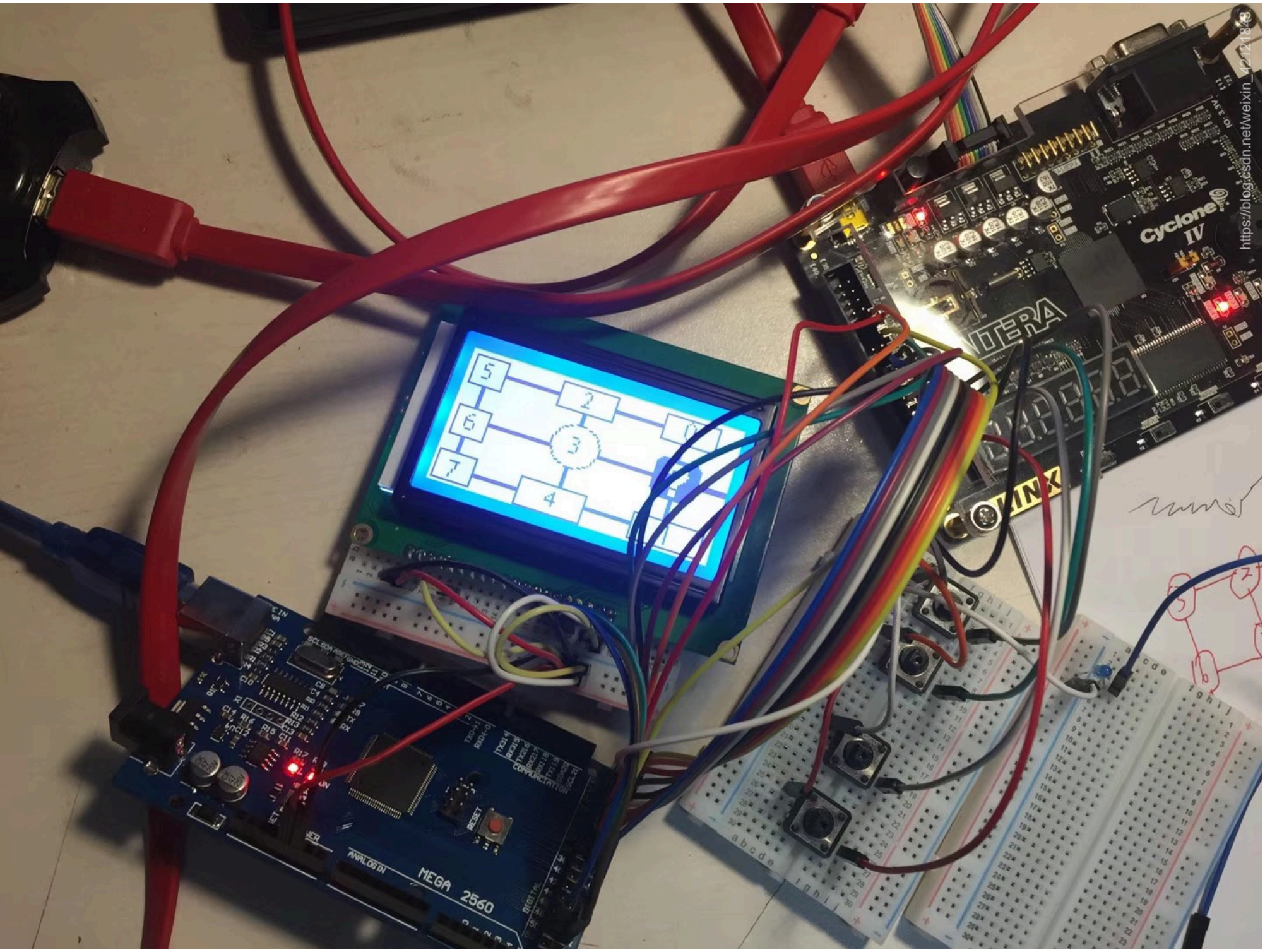
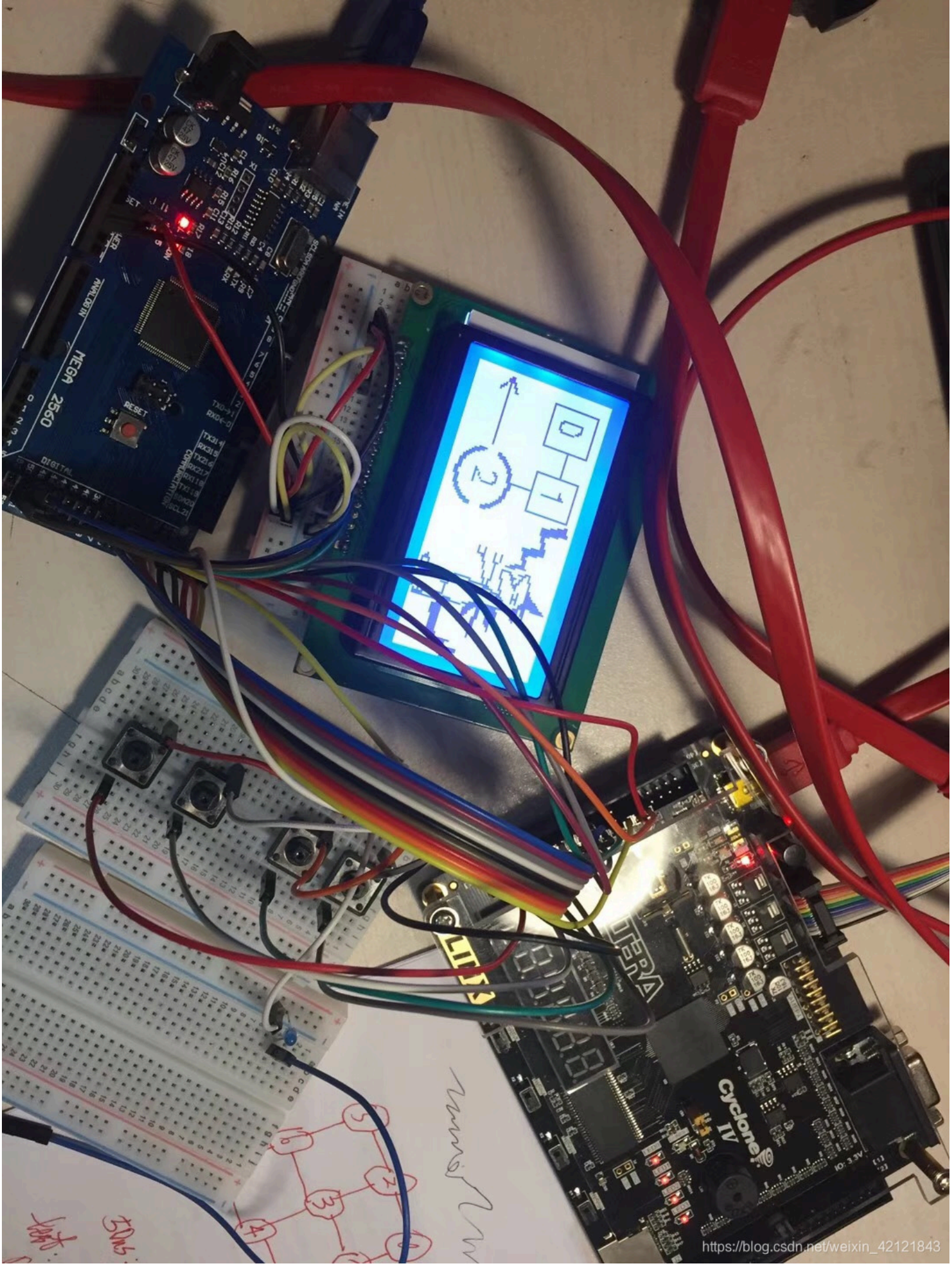
实物图





目 录

摘 要

关键词：FPGA， 时钟分频，异步时序
本设计基于FPGA（ALTERA cyloneII EP4CE6F17C8N）和Arduino（Mega2560）两种控制器，FPGA实现功能，Arduino实现屏幕显示。
定义时钟模块，为room模块和maze模块提供clk时钟。提供随机数。
定义room模块，实现重置，时钟，方向，和判断是否拿到剑，s[n]表示到达哪个房间，s3表示进入maze模块。用case语句实现走动和位置的记录，最终通过结合随机数和是否拿到sword来判断能否打败龙。
定义maze模块，制作了一个八个房间的迷宫，剑藏在随机数所确定的房间里,如果走到的房间恰好为随机数表示的房间，那么v=1，表明拿到了剑。

一、总体设计方案

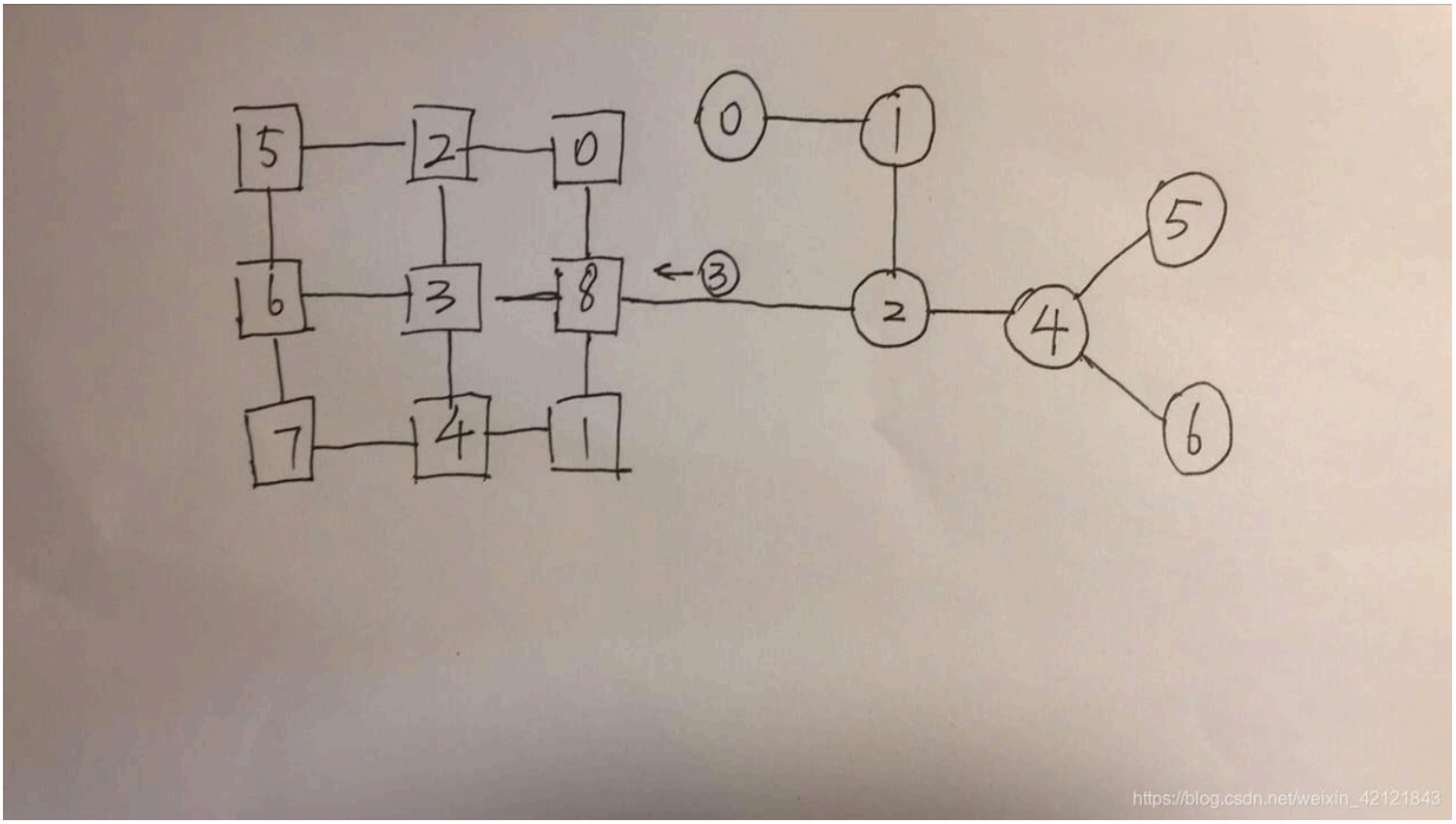
1.1 总体功能描述
有6个房间和1把斩首剑，以及一个包含9个房间的迷宫（剑在迷宫内）。开始于一个嘈杂的洞穴（the Cave of Cacophony）。要想获得胜利，必须要先通过曲折的隧道（the Twisty Tunnel）和湍急的河流（the Rapid River）；然后你需要进入迷宫寻找随机放置在一个房间的斩首剑（the Vorpal Sword）。这把剑将保护你通过后面危机四伏的龙穴（the Dragon Den），安全地到达胜利穹顶（the Victory Vault）赢得胜利。如果你进入龙穴而没有斩首剑，你会被凶残的暴龙撕裂分食，尸骨被扔进阴森恐怖的墓地（the Grievous Graveyard）最终输掉游戏。而如果你有斩首剑，还要根据随机数来判定你是否能取得胜利。
新功能：

1、设有九宫格式迷宫，而宝剑藏于其中，需要通过寻找来获得宝剑，增加了游戏的趣味性。

2、采用液晶屏显示，显示实时位置，并且会有如“YOU WIN”等游戏输赢结果的提示和暴龙凶恶的画面，增加游戏的可读性和趣味性。

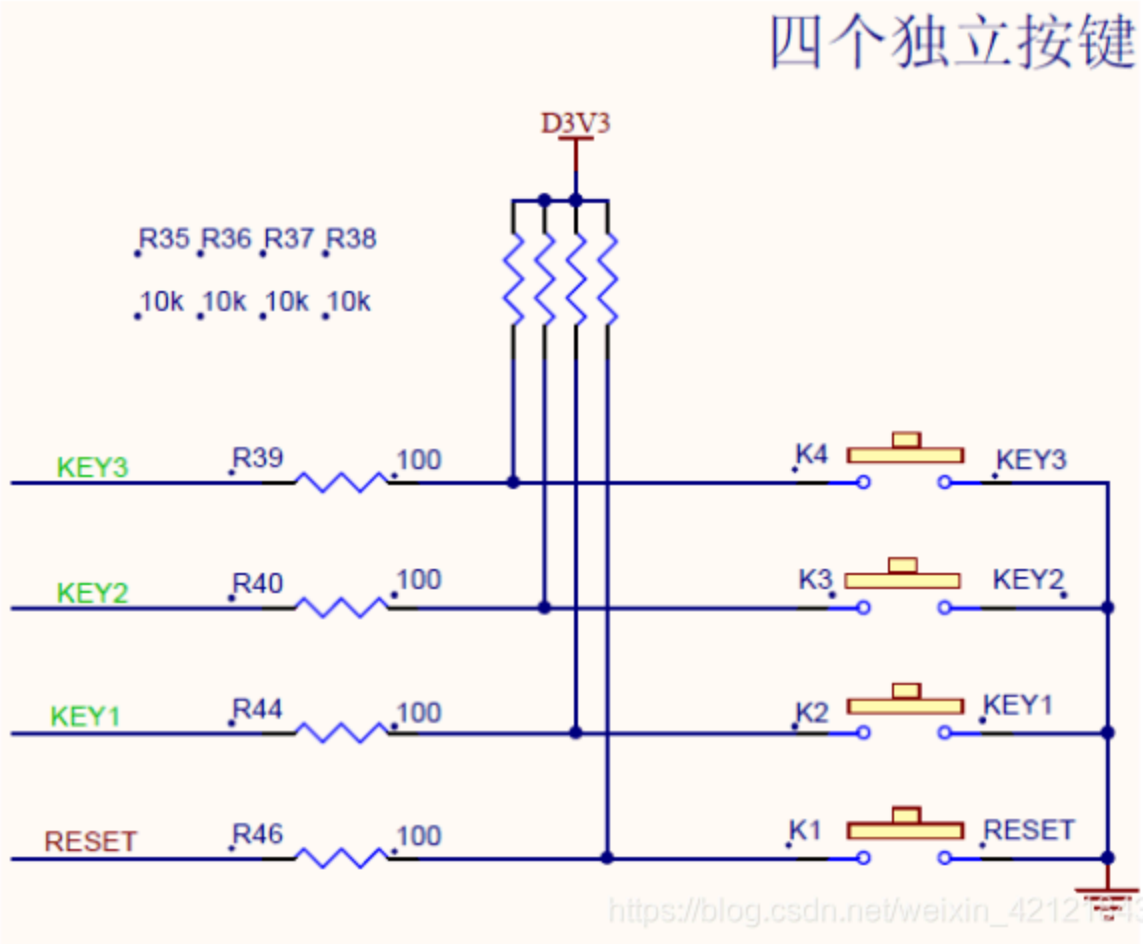
1.2 设计原理

（1）整体示意图（游戏地图）

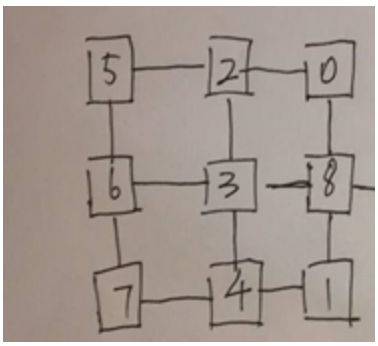


（2）子功能模块：

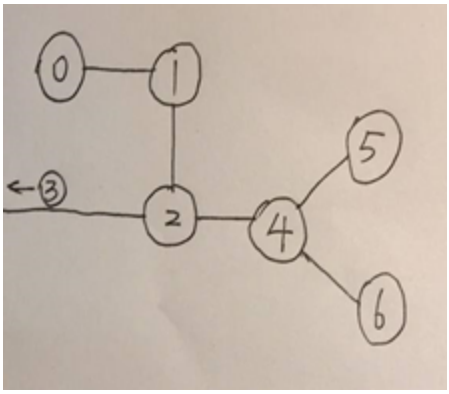
①独立按键模块：



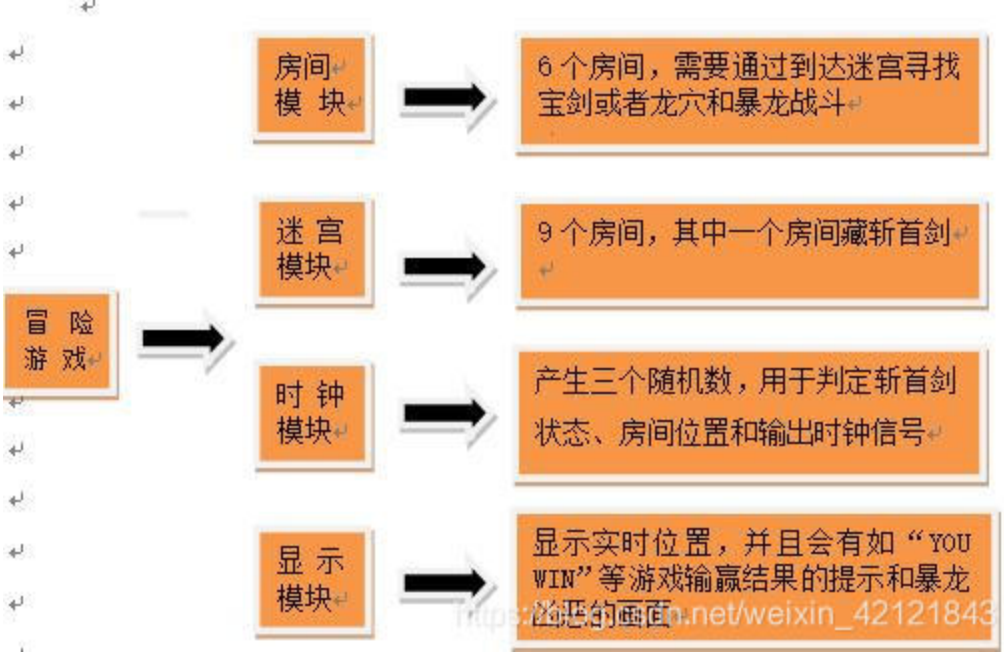
②Maze模块：



3、Room模块：



逻辑抽象



（4）状态转换图

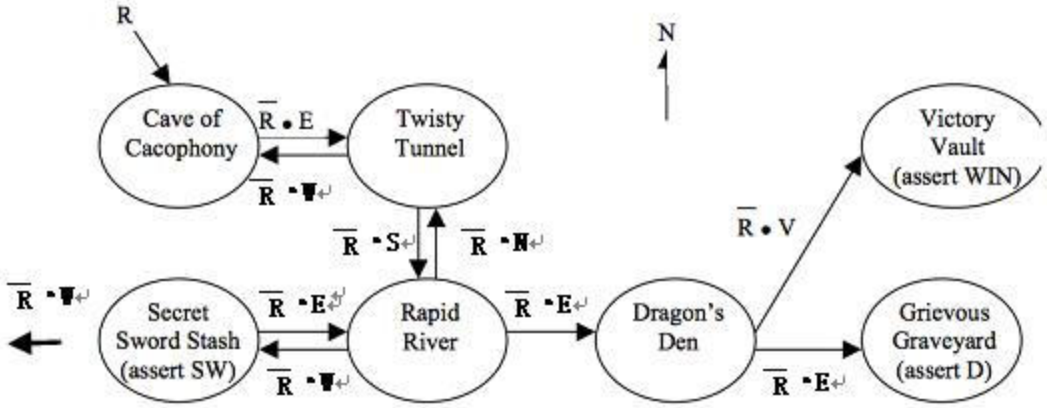


图 1　Room FSM 的状态转换图

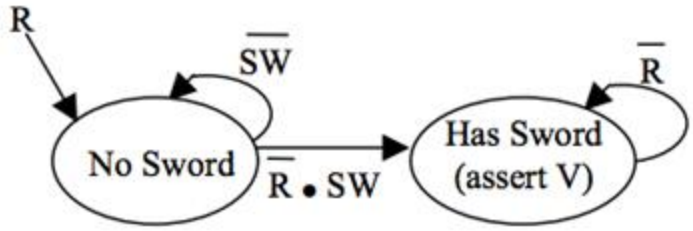


图 2 Sword FSM 的状态转换图

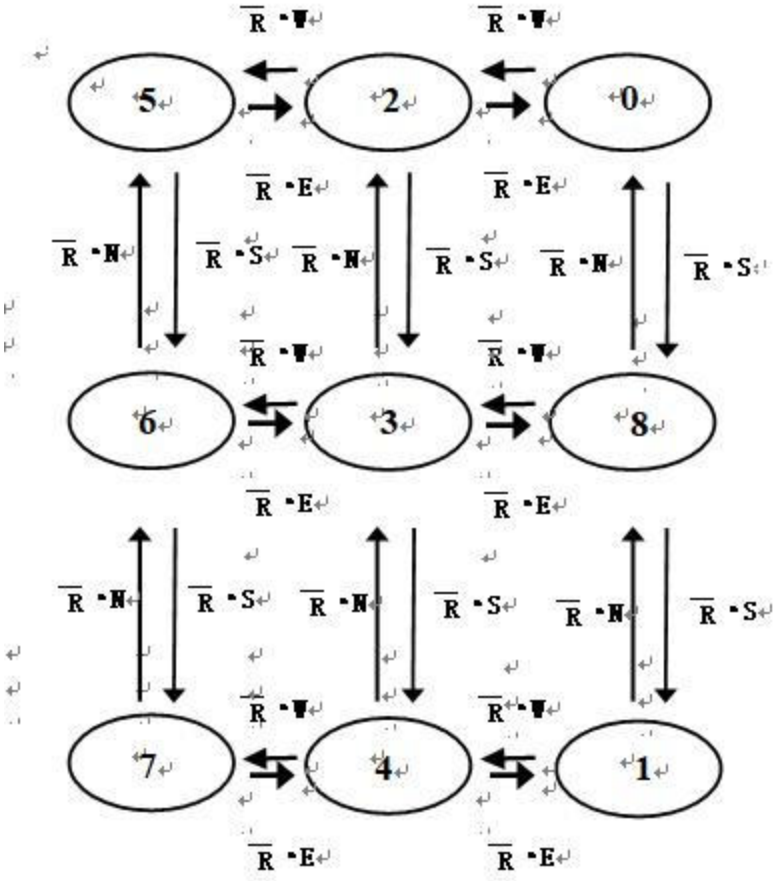


图 3 Maze FSM 的状态转换图

1.3 Verilog HDL程序代码设计及功能介绍

1、独立按键模块：

（1）功能：四个按健各代表“N”“S”“W”“E”中的一个方向，实现“东”“西”“南”“北”的位置移动。

（2）设计过程：通过按键实现输入和状态转化，设计按键在电路板的四个方向，操控更加方便

2、clock_random模块：

（1）功能：①产生了两个随机数，一个是0/1，判定拿到斩首剑与暴龙战斗是否能胜利。1代表胜利，0代表失败。一个是000-111，产生斩首剑存放的房间号码。

②输出7Hz的主时钟信号，避免按一次独立按键检测到多个信号。

（2）设计过程：通过时钟产生随机数，通过三个时钟类似计数器产生000-111的信号，时间相等，因此概

率也相等，在需要判定的那一刻取数，便实现了随机的效果。主时钟信号是多次调整的结果，频率太低容易检测不到，太高容易检测到多次。

3、单片机Arduino的LCD12864显示程序：

（1）功能：显示实时位置，并且会有如“YOU WIN”等游戏输赢结果的提示和暴龙凶恶的画面。

（2）设计过程：因用LED难以辨别状态，因此使用arduino的mega2560实现显示功能，通信方式为并行，通过I/O口检测高低电平显示不同的图像文字。

4、maze模块：

（1）功能：有9个房间，其中有一个房间藏着斩首剑，需要游戏者去寻找。

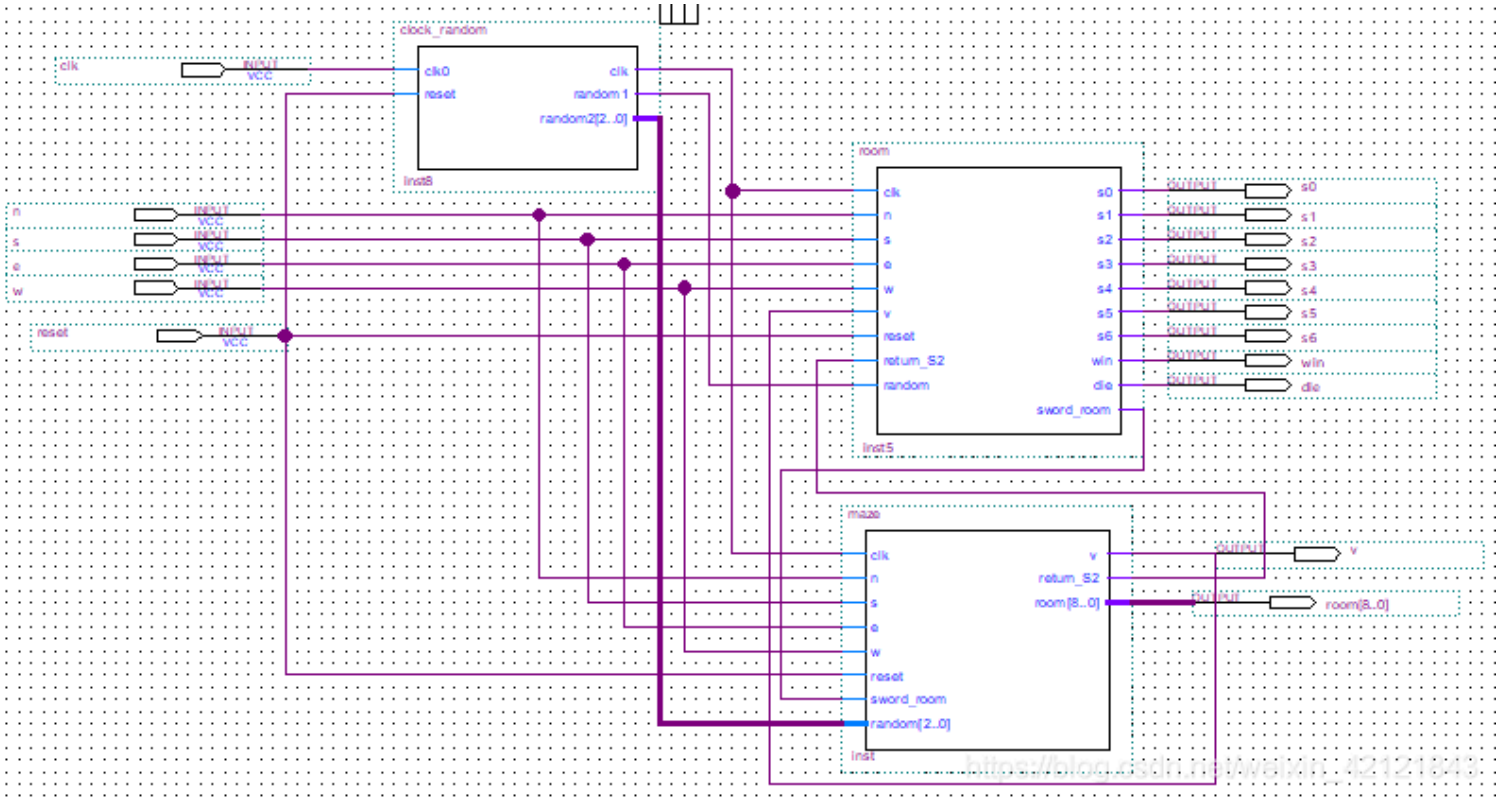
（2）设计过程：这是拓展部分，宝剑随机藏于0-7号房间，由东南西北四个按键控制，另外实验改进加入了一个类似锁存器的功能在模块中，以便于在进入房间一刻把随机数锁定。Maze模块属于状态3，与主模块room之间用sword_room和return_S2连接

5、Room模块：

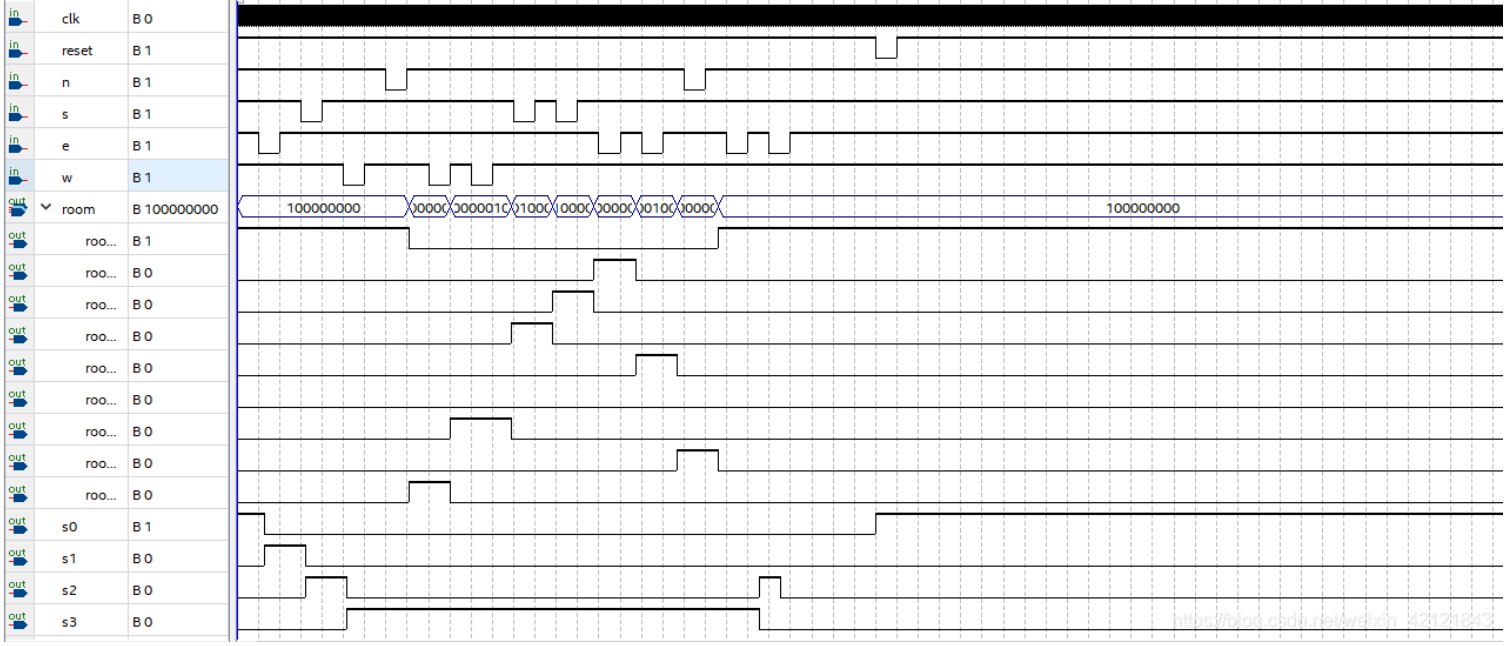
（1）功能：有 6个房间，需要游戏者通过来可能到达迷宫寻找斩首剑以及前往龙穴战斗。

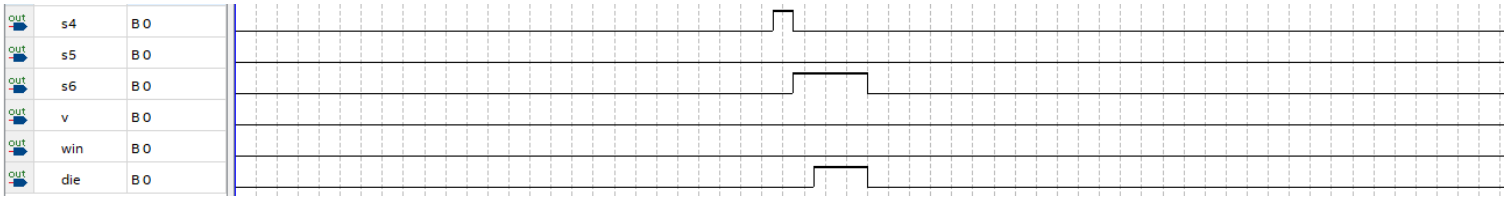
（2）设计过程：按照游戏要求设计，共7个状态，由东南西北四个按键控制。为避免进入S3后按键输入出现混乱，使用了if语句，在进入S3后room模块主功能停止作用。

1.4总体电路图



二、功能仿真及分析





分析：

向东：到达s1向南：到达s2

向西：到达maze,s3=1

向北：到达room0，v=0未拿到剑

向西：到达room2，v=0未拿到剑

向西：到达room5，v=0未拿到剑

向南：到达room6，v=0未拿到剑

向南：到达room7，v=0未拿到剑

向东：到达room4，v=0未拿到剑

向东：到达room1，v=0未拿到剑

向北：到达maze起点，v=0未拿到剑

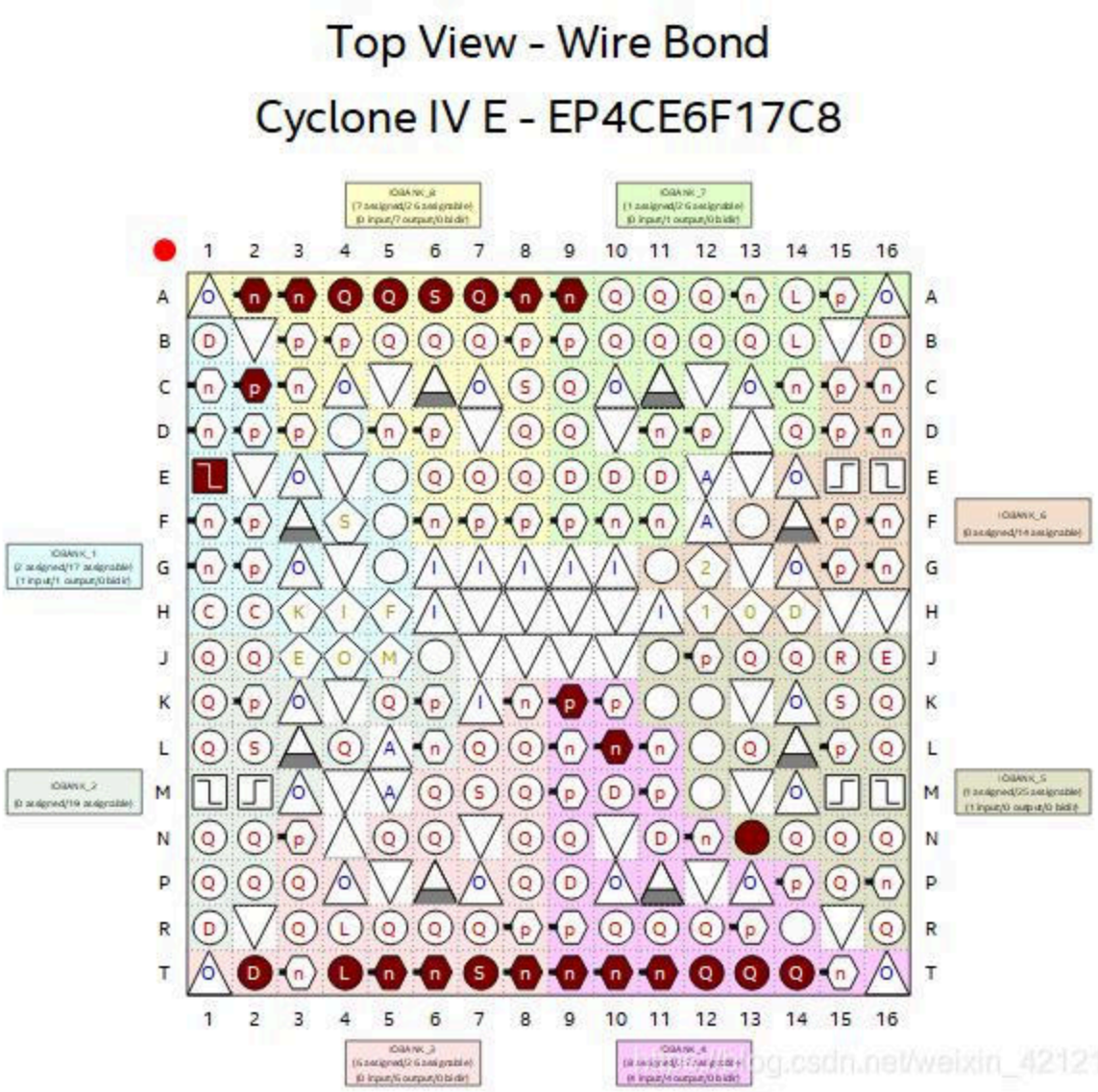
向东：到达s2

向东：到达龙穴，没有剑，die=1战败死亡

reset置零，回到起点。

三、功能测试及分析

1、引脚分配

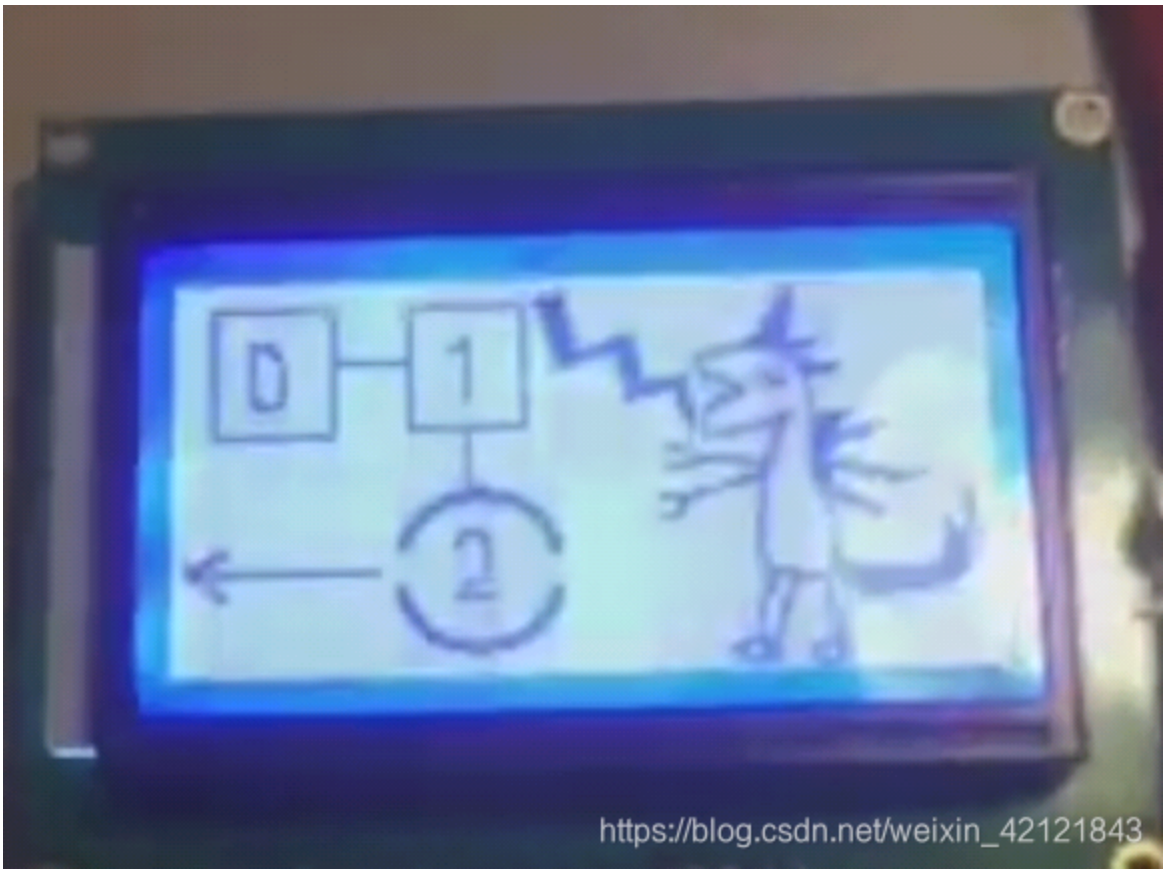


in	clk	Input	PIN_E1
out	die	Output	PIN_L10
in	e	Input	PIN_T12
in	n	Input	PIN_T14
in	reset	Input	PIN_N13
out	room[8]	Output	PIN_C2
out	room[7]	Output	PIN_A2
out	room[6]	Output	PIN_A3
out	room[5]	Output	PIN_A4
out	room[4]	Output	PIN_A5
out	room[3]	Output	PIN_A6
out	room[2]	Output	PIN_A7
out	room[1]	Output	PIN_A8
out	room[0]	Output	PIN_A9
in	s	Input	PIN_T13
out	s0	Output	PIN_T4
out	s1	Output	PIN_T5
out	s2	Output	PIN_T6
out	s3	Output	PIN_T7
out	s4	Output	PIN_T8

out s5	Output	PIN_T9
out s6	Output	PIN_T10
out v	Output	PIN_K9
in w	Input	PIN_T11
out win	Output	PIN_T2

2、测试照片

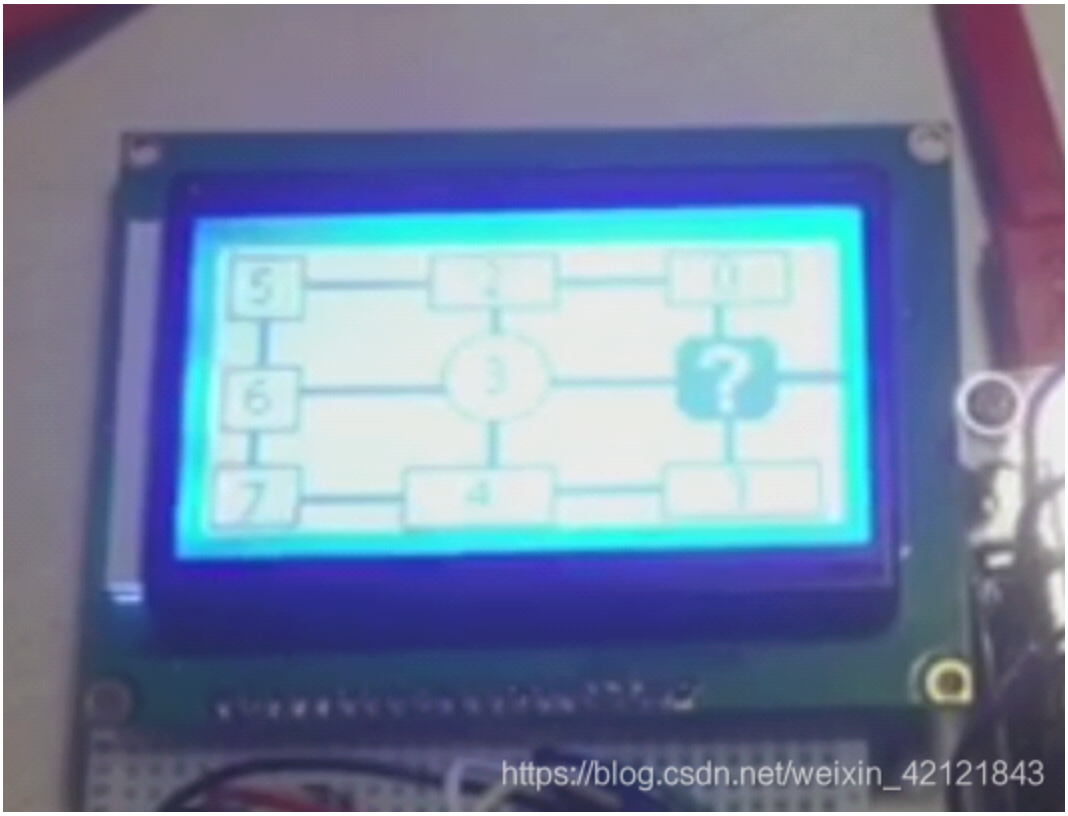
①初始界面：显示嘈杂的洞穴（0号位置），曲折的隧道（1号位置）湍急的河流（2号位置），以及暴龙形象，呼应冒险游戏的主题，渲染紧张的游戏氛围。



②显示初始位置：嘈杂的洞穴（0号位置）

③经过曲折的隧道（1号位置）湍急的河流（2号位置）到达迷宫入口（entrance），即迷宫里的初始位置（8号房间）

④到达“？”房间（8号房间）后，显示迷宫地图。斩首剑藏于其中一个房间。



⑤去不同房间寻找宝剑，当前位于迷宫的3号房间

⑥获得斩首剑，即将返回前往龙穴

⑦返回到湍急的河流,下一个地点即为龙穴

⑧拿到斩首剑，却因为暴龙状态极佳，而无法与之匹敌，最终死亡（“you die”）

⑨游戏失败的画面

⑩游戏者再接再厉，最终拿到斩首剑赢得胜利

⑪游戏胜利的欢乐画面

四、结论

4.1系统特点及存在的问题

系统特点

模块之间接口完美，逻辑清晰。

利用同步时序电路，明确状态的变化，可以清楚的观察游戏者所处的位置。

存在的问题

迷宫部分实现了最基本的走动和随机数功能，还可以拓展其丰富性，另外，打龙的时候用随机数的概率来获得胜利比较单调，可以再设计一个能为玩家所控制的打龙模块。迷宫部分实现了最基本的走动和随机数功能，还可以拓展其丰富性，另外，打龙的时候用随机数的概率来获得胜利比较单调，可以再设计一个能为玩家所控制的打龙模块。

4.2学习体会

学到了各个模块之间如何实现接口的信息反馈，认识到模块的可扩展性尤为重要。熟悉了各种语句比如if语句case语句。学会如何表示状态的转移。认识到正确的仿真对测试结果的重要性。

五、附件（代码）

FPGA部分 (Verilog HDL)：

Room模块：

```
module room(
input clk,n,s,e,w,v,reset,return_S2,random,
output reg s0,s1,s2,s3,s4,s5,s6,win,die,sword_room
);
always@(posedge clk or negedge reset)
begin
if(reset==1'b0)
{s0,s1,s2,s3,s4,s5,s6,win,die,sword_room}<=10'b1000_0000_00;
else if(return_S2&&s3==1)
        begin
        {s0,s1,s2,s3,s4,s5,s6,sword_room}<=8'b0010_0000;
        end
else if(sword_room==0)
        begin
        case ({s0,s1,s2,s3,s4,s5,s6})
        7'b1000_000:if(e==0) //s0
                                                {s0,s1,s2,s3,s4,s5,s6}<=7'b0100_000;
        7'b0100_000:begin//s1
                                        if(s==0)
                                                {s0,s1,s2,s3,s4,s5,s6}<=7'b0010_000;
                                        else if(w==0)
```

```
                                        {s0,s1,s2,s3,s4,s5,s6}<=7'b1000_000;
                                    end
        7'b0010_000:begin//s2
                                        if(n==0)
                                        {s0,s1,s2,s3,s4,s5,s6}<=7'b0100_000;
                                        else if(w==0)
                                        {s0,s1,s2,s3,s4,s5,s6}<=7'b0001_000;//s3
                                        else if(e==0)
                                        {s0,s1,s2,s3,s4,s5,s6}<=7'b0000_100;//s4
                                        end
        7'b0001_000:begin//s3
                                    sword_room<=1;
                                    end
        7'b0000_100:begin//s4
                                        if(v==0)
                                        begin
                                                    {s0,s1,s2,s3,s4,s5,s6}<=7'b
                                        end
                                                    else
                                                            begin
                                                            if(random==0)
                                                            {s0,s1,s2,s3,s4,s5,
                                                            else
                                                            {s0,s1,s2,s3,s4,s5,
                                                            end
                                    end

        7'b0000_010:begin//s5
                                    win<=1;
                                    end
        7'b0000_001:begin//s6
                                    die<=1;
                                    end
        default:{s0,s1,s2,s3,s4,s5,s6}<=7'b1000_000;

endcase

end
end
endmodule
```

maze模块：

```
module maze(
input clk,n,s,e,w,reset,sword_room,
input[2:0]random,
output reg v,
output reg return_S2,
output reg [8:0] room
);
reg[3:0]state=4'b1000;
reg flag=0;
reg [2:0]locked=3'b111;
always@(posedge clk or negedge reset)
begin
                                if(reset==1'b0)
                                begin
                                state<=4'b1000;
                                flag<=0;
                                locked[2:0]<=3'b111;
                                v<=0;
                                return_S2<=0;
                                room<=9'b100_000_000;
                                end
                                else if(sword_room)

                                                begin
                                                        if(flag==0)
                                                        begin
                                                        locked[2:0]<=random[2:0];
                                                        flag<=1;
                                                        end
                                                case(state)
                                                4'b0000:begin
                                                room<=9'b000_000_001;//room0
                                                        if(w==0)state<=4'b0010;//2
                                                        else if(s==0)state<=4'b1000;//8
                                                        else state<=4'b000;
                                                        end
                                                4'b0001:begin
                                                room<=9'b000_000_010;//room1
                                                        if(w==0)state<=4'b0100;
                                                        else if(n==0)state<=4'b1000;
                                                        else state<=4'b0001;
                                                        end
                                                4'b0010:begin
                                                room<=9'b000_000_100;//room2
                                                        if(s==0)state<=4'b0011;
                                                        else if(w==0)state<=4'b0101;
```

```
        else if(e==0)state<=4'b0000;
        else state<=4'b0010;
        end
4'b0011:begin
room<=9'b000_001_000;//room3
        if(e==0)state<=4'b1000;
        else if(w==0)state<=4'b0110;
        else if(n==0)state<=4'b0010;
        else if(s==0)state<=4'b0100;
        else state<=4'b0011;
        end
4'b0100:begin
room<=9'b000_010_000;//room4
        if(e==0)state<=4'b0001;
        else if(n==0)state<=4'b0011;
        else if(w==0)state<=4'b0111;
        else state<=4'b0100;
        end
4'b0101:
        begin
        room<=9'b000_100_000;
        if(e==0)state<=4'b0010;
        else if(s==0)state<=4'b0110;
        else state<=4'b0101;
        end
4'b0110:begin
room<=9'b001_000_000;
        if(e==0)state<=4'b0011;
        else if(s==0)state<=4'b0111;
        else if(n==0)state<=4'b0101;
        else state<=4'b0110;
        end
4'b0111:begin
room<=9'b010_000_000;
        if(e==0)state<=4'b0100;
        else if(n==0)state<=4'b0110;
        else state<=4'b0111;
        end
4'b1000:begin
room<=9'b100_000_000;//room8
        if(e==0)  return_S2<=1;
        else if(n==0)state<=4'b0000;
        else if(s==0)state<=4'b0001;
        else if(w==0)state<=4'b0011;
        else state<=4'b1000;
        end
```

```
                                                endcase
                                                if(state[2:0]==locked[2:0])
                                                v<=1;
                                                end

        end

endmodule
```

clock_random模块：

```
module clock_random(
input clk0,reset,
output reg clk,random1,
output reg [2:0] random2
);
reg clk1,clk2,clk3;
//test:
/*
parameter COUNT_TIME=2000_0000;//room and maze,clk
parameter COUNT_TIME1=2000_0000;//clk1
parameter COUNT_TIME2=1000_0000;//clk2
parameter COUNT_TIME3=500_0000;//clk3
parameter CLK_FREQ = 100_00_0000; //clock frequency
*/

parameter COUNT_TIME=7;//room and maze
parameter COUNT_TIME1=100;
parameter COUNT_TIME2=50;
parameter COUNT_TIME3=200;
parameter CLK_FREQ = 5000_0000; //clock frequency

parameter NUM_COUNT=CLK_FREQ/COUNT_TIME;
parameter NUM_COUNT1=CLK_FREQ/COUNT_TIME1;
parameter NUM_COUNT2=CLK_FREQ/COUNT_TIME2;
parameter NUM_COUNT3=CLK_FREQ/COUNT_TIME3;
reg[31:0] scan_timer,scan_timer1,scan_timer2,scan_timer3;  //scan time counter
always@(posedge clk0 or negedge reset)
begin
        if(reset == 1'b0)
        begin
                scan_timer<=32'd0;
                scan_timer1 <= 32'd0;
```

```
        scan_timer2<= 32'd0;
        scan_timer3<= 32'd0;
end
else if(scan_timer>=NUM_COUNT)
begin
                scan_timer<=32'd0;
                clk=~clk;
end
else if(scan_timer1>=NUM_COUNT1)
begin
                scan_timer1<=32'd0;
                clk1=~clk1;
end
else if(scan_timer2>=NUM_COUNT2)
begin
                scan_timer2<=32'd0;
                clk2=~clk2;
end
else if(scan_timer3>=NUM_COUNT3)
begin
                scan_timer3<=32'd0;
                clk3=~clk3;
end
else
        begin
                scan_timer<=scan_timer+32'd1;
                scan_timer1 <= scan_timer1 + 32'd1;
                scan_timer2 <= scan_timer2 + 32'd1;
                scan_timer3 <= scan_timer3 + 32'd1;


        if(clk1==clk2)
        random1<=1;
        else
        random1<=0;
        //random2:0-7,一定要写成b不能是d!
        case({clk1,clk2,clk3})
                3'b000:random2<=3'b000;
                3'b001:random2<=3'b001;
                3'b010:random2<=3'b010;
                3'b011:random2<=3'b011;
                3'b100:random2<=3'b100;
                3'b101:random2<=3'b101;
                3'b110:random2<=3'b110;
                3'b111:random2<=3'b111;
                default:random2<=0;
```

```
                    endcase

                end

end
endmodule
```

单片机Arduino的LCD12864显示程序(C语言):

```
#include <LCD12864RSPI.h>
#define AR_SIZE( a ) sizeof( a ) / sizeof( a[0] )
#define S0 22
#define S1 24
#define S2 26
#define S3 28
#define S4 30
//#define S5 32
//#define S6 34
#define V 32
#define WIN 34
#define DIE 36
#define ROOM0 23
#define ROOM1 25
#define ROOM2 27
#define ROOM3 29
#define ROOM4 31
#define ROOM5 33
#define ROOM6 35
#define ROOM7 37
#define ROOM8 39
//unsigned char show0[]={0xBC,0xAB,0xBF,0xCD,0xB9,0xA4,0xB7,0xBB};//极客工坊
const unsigned char show1[] ="make by Chan,Jin,Shou";
unsigned char  win_pic[]=
//以下省略几十页图片矩阵
const unsigned char s0[] ="Your position is 0";
const unsigned char s1[] ="Your position is 1";
const unsigned char s2[] ="Your position is 2";
const unsigned char s3[] ="You're in the sword_room";
const unsigned char s4[] ="You're fighting with dragon";
const unsigned char win[]="you WIN";
const unsigned char die[]="you DIE";
const unsigned char v[]="get the sword";
const unsigned char room8[] ="You're in entrance";
```

```
const unsigned char room7[] ="You're in room7";
const unsigned char room6[] ="You're in room6";
const unsigned char room5[] ="You're in room5";
const unsigned char room4[] ="You're in room4";
const unsigned char room3[] ="You're in room3";
const unsigned char room2[] ="You're in room2";
const unsigned char room1[] ="You're in room1";
const unsigned char room0[] ="You're in room0";
void setup()
{
LCDA.Initialise(); // 屏幕初始化
delay(100);
for(int i=22;i<=36;i=i+2)
pinMode(i,INPUT);
for(int i=23;i<=41;i=i+2)
pinMode(i,INPUT);
LCDA.DisplayString(0,0,show1,AR_SIZE(show1));
delay(1000);
LCDA.CLEAR();
delay(50);
}

void loop()
{

 if((digitalRead(S3)==0)&&digitalRead(WIN)==0&&digitalRead(DIE)==0)
  {
     LCDA.DrawFullScreen(main_map);
     delay(1000);
     if(digitalRead(S0))
      {
        LCDA.CLEAR();
        LCDA.DisplayString(2,1,s0,AR_SIZE(s0));
        delay(500);
       }
      if(digitalRead(S1))
      {
        LCDA.CLEAR();
        LCDA.DisplayString(2,1,s1,AR_SIZE(s1));
        delay(500);
       }
       if(digitalRead(S2))
      {
        LCDA.CLEAR();
        LCDA.DisplayString(2,1,s2,AR_SIZE(s2));
        delay(500);
```

```
      }
      if(digitalRead(S4))
     {
       LCDA.CLEAR();
       LCDA.DisplayString(2,1,s4,AR_SIZE(s4));
       delay(500);
      }

       LCDA.CLEAR();
       delay(50);
 }
if(digitalRead(S3)==1)
 {
   LCDA.DrawFullScreen(sword_room_map);
   delay(1000);
   if(digitalRead(ROOM0))
   {
     LCDA.CLEAR();
     LCDA.DisplayString(2,1,room0,AR_SIZE(room0));
     delay(500);
         if(digitalRead(V))
        {
            LCDA.CLEAR();
            LCDA.DisplayString(2,1,v,AR_SIZE(v));
             delay(500);
         }
     }

    if(digitalRead(ROOM1))
   {
     LCDA.CLEAR();
     LCDA.DisplayString(2,1,room1,AR_SIZE(room1));
     delay(500);
          if(digitalRead(V))
        {
            LCDA.CLEAR();
            LCDA.DisplayString(2,1,v,AR_SIZE(v));
             delay(500);
         }
     }

     if(digitalRead(ROOM2))
   {
     LCDA.CLEAR();
     LCDA.DisplayString(2,1,room2,AR_SIZE(room2));
     delay(500);
```

```
        if(digitalRead(V))
     {
         LCDA.CLEAR();
         LCDA.DisplayString(2,1,v,AR_SIZE(v));
          delay(500);
       }
  }

  if(digitalRead(ROOM3))
{
  LCDA.CLEAR();
  LCDA.DisplayString(2,1,room3,AR_SIZE(room3));
  delay(500);
       if(digitalRead(V))
     {
         LCDA.CLEAR();
         LCDA.DisplayString(2,1,v,AR_SIZE(v));
          delay(500);
       }
  }

  if(digitalRead(ROOM4))
{
  LCDA.CLEAR();
  LCDA.DisplayString(2,1,room4,AR_SIZE(room4));
  delay(500);
       if(digitalRead(V))
     {
         LCDA.CLEAR();
         LCDA.DisplayString(2,1,v,AR_SIZE(v));
          delay(500);
       }
  }

  if(digitalRead(ROOM5))
{
  LCDA.CLEAR();
  LCDA.DisplayString(2,1,room5,AR_SIZE(room5));
  delay(500);
       if(digitalRead(V))
     {
         LCDA.CLEAR();
         LCDA.DisplayString(2,1,v,AR_SIZE(v));
          delay(500);
       }
  }
```

```
    if(digitalRead(ROOM6))
  {
    LCDA.CLEAR();
    LCDA.DisplayString(2,1,room6,AR_SIZE(room6));
    delay(500);
         if(digitalRead(V))
       {
           LCDA.CLEAR();
           LCDA.DisplayString(2,1,v,AR_SIZE(v));
            delay(500);
         }
    }

    if(digitalRead(ROOM7))
  {
    LCDA.CLEAR();
    LCDA.DisplayString(2,1,room7,AR_SIZE(room7));
    delay(500);
         if(digitalRead(V))
       {
           LCDA.CLEAR();
           LCDA.DisplayString(2,1,v,AR_SIZE(v));
            delay(500);
         }
    }
          if(digitalRead(ROOM8))
  {
    LCDA.CLEAR();
    LCDA.DisplayString(2,1,room8,AR_SIZE(room8));
    delay(500);
         if(digitalRead(V))
       {
           LCDA.CLEAR();
           LCDA.DisplayString(2,1,v,AR_SIZE(v));
            delay(500);
         }
    }


  }
if(digitalRead(WIN)==1||digitalRead(DIE)==1)
  {
  if(digitalRead(WIN))
    {
      LCDA.CLEAR();
```

```
        LCDA.DisplayString(2,1,win,AR_SIZE(win));
        delay(500);
        LCDA.DrawFullScreen(win_pic);
        delay(1000);
       }
       if(digitalRead(DIE))
      {
        LCDA.CLEAR();
        LCDA.DisplayString(2,1,die,AR_SIZE(die));
        delay(500);
         LCDA.DrawFullScreen(die_pic);
        delay(1000);
       }
    }

    LCDA.CLEAR();
        delay(50);
}
```