**LCM12864移植贪吃蛇全部模式**

**第一步**：硬件电路连接

LCM12864数据总线PA0→PA7 GPIO\_OUTPUT

LCM12864控制总线PB4→PB0 GPIO\_OUTPUT

LCM12864两边电源线：外侧GND内测3.3V电源

LCM12864倍压引脚0V→5V可变电压（亮度调节）

风扇驱动PC4 GPIO\_OUTPUT

外部中断按键PC13 控制风扇

八方向键 PD8+PD6→PD0 GPIO\_INPUT 上拉、按下为0

模式选择PC6→PC11 EXTI 上拉、按下为0、下降沿触发

**第二步**：具体硬件电路连接

LCM12864数据总线：

DB0→PA0[CN\_8\_1] DB1→PA1[CN\_8\_2]

DB2→PA2[CN\_10\_34] DB3→PA3[CN\_10\_6]

DB4→PA4[CN\_8\_3] DB5→PA5[CN\_5\_6]

DB6→PA6[CN\_5\_5] DB7→PA7[CN\_5\_4]

LCM12864控制总线：

RS→PB0[CN\_5\_3] RW→PB1[CN\_8\_4]

E→PB2[CN\_10\_22] PSB→PB3[CN\_9\_4]

RST→PB4[CN\_9\_6]

电源线：

LCM12864外侧接GND内测接3.3V电源

风扇驱动接5V和GND 按键一端全部接GND

风扇驱动PC4[CN\_9\_2]

八方向键：

左PD0[CN\_7\_9] 左PD4[CN\_7\_15]

右PD1[CN\_7\_10] 右PD5[CN\_7\_26]

上PD2[CN\_7\_4] 上PD6[CN\_10\_10]

下PD3[CN\_7\_11] 下PD8[CN\_10\_36]

**第三步**：功能规划

复位按键开始新的一局、13号中断按键控制风扇给MPU降温、LCM12864显示界面、扩展电路包括（电源扩展：3.3V、5V、GND、0-5V可变电压调节LCM亮度；模式控制扩展0-9十个模式选择数字按键；2\*4两个四方向控制键）

**第四步**：添加代码

①变量以及函数定义

#include "main.h"

#define u8 unsigned char

#define u16 unsigned int

int last\_num[32][16];

void delay(void);

void lcdwritecmd(unsigned char cmd);

void lcdwritedata(unsigned char dat);

void lcdinit(void);

void clear(void);

void drowpoint(char x,char y,char z);

void lcdsetcursor(unsigned char x, unsigned char y);

void lcdshowstr(unsigned char x, unsigned char y, unsigned char \*str);

const int flg[4][2]={{1,0},{0,1},{0,-1},{-1,0}};//下右左上坐标变化规律

int n=127,m=63; //n和m是蛇的活动范围28行76列+边框=可视化窗口大小

struct xcw{int x,y;}tang[5][15],foodd[2];//结构体tan存储每条蛇的坐标 结构体foodd记录两个食物坐标

int tot[5]; //tot[0123]是四条蛇的长度

int f[5]; //存储蛇的运行状态0123下右左上 搭配flg使用

int score; //得分

int food; //食物个数

int ms=0; //初始--模式选择

int ans\_len=1e9;

int vis[128][64];//记录食物存在点的矩阵 例如(foddd[0].x,foddd[0].y)有食物则vis[foddd[0].x][foddd[0].y]=1

//int choose(); //模式选择，中断代替

int mods(); //对应模式小蛇初始化

void draw(); //显示边框

void rand\_food(); //没有食物时随机投喂

int change(int t); //没有按键按下蛇的运行

int check(int x,int y); //检验某一个点

void machine(int t,int x,int y); //机器人寻路

void result(int t); //积分够了-胜利

void fnd(int t,int x,int y); //撞墙撞蛇--失败

int ads(int x); //取绝对值

int other(int x); //03交换、12交换

int get\_food(int x,int y,int xxx,int yyy); //辅助机器人找寻食物

//void Ending(); //结束

②中断管脚判断以及释放中断标志位

ITStatus EXTI\_GetITStatus(uint32\_t EXTI\_Line) //判断中断管脚

{

ITStatus bitstatus = RESET; //初始位状态0

uint32\_t enablestatus = 0; //初始使能状态0

/\* Check the parameters \*/

assert\_param(IS\_GET\_EXTI\_LINE(EXTI\_Line));

enablestatus = EXTI->IMR1 & EXTI\_Line;

if (((EXTI->FPR1 & EXTI\_Line) != (uint32\_t)RESET) && (enablestatus != (uint32\_t)RESET))

bitstatus = SET;

else

bitstatus = RESET;

return bitstatus;

}

void EXTI\_ClearITPendingBit(uint32\_t EXTI\_Line) //释放对应管脚标志位

{

/\* Check the parameters \*/

assert\_param(IS\_EXTI\_LINE(EXTI\_Line));

EXTI->FPR1 = EXTI\_Line;

}

③系统时钟和GPIO初始化声明（自带）

④主函数main非循环部分

HAL\_Init();

SystemClock\_Config();

MX\_GPIO\_Init();

lcdinit();

HAL\_Delay(100);

⑤主函数循环部分

while (1)

{

while(ms==0)

{

lcdshowstr(1,1,"Select Mode");

HAL\_Delay(100);

}

while(ms==1) //1单人模式

{

rand\_food();

if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_4)==0&&f[0]^0) //左右上下

f[0]=3;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_5)==0&&f[0]^3)

f[0]=0;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_6)==0&&f[0]^1)

f[0]=2;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_8)==0&&f[0]^2)

f[0]=1;

else ;

change(0);

}

while(ms==2) //2双人合作

{

rand\_food();

if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_0)==0&&f[0]^0) // 上

f[0]=3;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_1)==0&&f[0]^3) //下

f[0]=0;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_2)==0&&f[0]^1) //左

f[0]=2;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_3)==0&&f[0]^2) //右

f[0]=1;

else ;

if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_4)==0&&f[1]^0) // 上

f[1]=3;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_5)==0&&f[1]^3) //下

f[1]=0;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_6)==0&&f[1]^1) //左

f[1]=2;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_8)==0&&f[1]^2) //右

f[1]=1;

else ;

change(0),change(1);

}

while(ms==3) //3双人对抗

{

rand\_food();

if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_0)==0&&f[0]^0) // 上

f[0]=3;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_1)==0&&f[0]^3) //下

f[0]=0;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_2)==0&&f[0]^1) //左

f[0]=2;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_3)==0&&f[0]^2) //右

f[0]=1;

else ;

if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_4)==0&&f[1]^0) // 上

f[1]=3;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_5)==0&&f[1]^3) //下

f[1]=0;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_6)==0&&f[1]^1) //左

f[1]=2;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_8)==0&&f[1]^2) //右

f[1]=1;

else ;

change(0),change(1);

}

while(ms==4) //4机机对抗抢分

{

rand\_food();

change(2),change(3);

}

while(ms==5) //5人机对抗抢分

{

rand\_food();

if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_4)==0&&f[0]^0) // 上

f[0]=3;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_5)==0&&f[0]^3) //下

f[0]=0;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_6)==0&&f[0]^1) //左

f[0]=2;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_8)==0&&f[0]^2) //右

f[0]=1;

else ;

change(0),change(2);

}

while(ms==6) //6双人对抗抢分

{

rand\_food();

if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_0)==0&&f[0]^0) // 上

f[0]=3;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_1)==0&&f[0]^3) //下

f[0]=0;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_2)==0&&f[0]^1) //左

f[0]=2;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_3)==0&&f[0]^2) //右

f[0]=1;

else ;

if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_4)==0&&f[1]^0) // 上

f[1]=3;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_5)==0&&f[1]^3) //下

f[1]=0;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_6)==0&&f[1]^1) //左

f[1]=2;

else if (HAL\_GPIO\_ReadPin(GPIOD,GPIO\_PIN\_8)==0&&f[1]^2) //右

f[1]=1;

else ;

change(0),change(1);

}

}

⑥系统时钟和GPIO初始化具体函数（自带）

⑦和LCM12864相关函数

void delay ()

{

for(int i=0;i<99;i++)

for(int j=0;j<99;j++)

{}

}

void lcdwritecmd(unsigned char cmd)

{

delay(); // 使能 写 指令

GPIOB->ODR=0x18; //RST=1\PSB=1\E=0\RW=0\RS=0

GPIOA->ODR=cmd;

GPIOB->ODR=0x1C; //RST=1\PSB=1\E=1\RW=0\RS=0

delay();

GPIOB->ODR=0x18; //RST=1\PSB=1\E=0\RW=0\RS=0

}

void lcdwritedata(unsigned char dat)

{

delay(); // 使能 写 数据

GPIOB->ODR=0x19; //RST=1\PSB=1\E=0\RW=0\RS=1

GPIOA->ODR=dat;

GPIOB->ODR=0x1D; //RST=1\PSB=1\E=1\RW=0\RS=1

delay();

GPIOB->ODR=0x19; //RST=1\PSB=1\E=0\RW=0\RS=1

}

void lcdinit()

{

lcdwritecmd(0x34); //E 使能 RS1数据0指令 RW1读0写

lcdwritecmd(0x30);

lcdwritecmd(0x0C);

lcdwritecmd(0x01);

}

void clear(void)

{

for(int i=0;i<=32;i++) // 64行 不用一个一个像素清除了

for(int j=0;j<=18;j++) // 18大列

{

lcdwritecmd(0x34); //打开扩展指令集

lcdwritecmd(0x80 + i - 1);

lcdwritecmd(0x80 + j);

lcdwritedata(0x00);

lcdwritedata(0x00);

lcdwritecmd(0x36); //打开绘图显示

lcdwritecmd(0x30); //回到基本指令集

}

}

void drowpoint(char x,char y,char z) //像素输出

{

char i = 0;

char j = 0;

int num = 0x8000; //16位输出高位在左

lcdwritecmd(0x34); //打开扩展指令集

if( y <= 32 )

{

i += y; // i=y -1 i的范围32以下

j = (x - 1) / 16; //大列 j的范围16以下

x = ( x%16>0? (x%16):16 ) - 1 ;

num = num >> x;

}

else

{

i = y - 32; //i=y-32 -1

j = ( (x - 1) / 16 ) + 8;

x = ( x%16>0? (x%16):16 ) - 1 ;

num = num >> x;

}

if(z>0)

num = last\_num[i][j] | num;

else

{

num=~num;

num=last\_num[i][j] & num;

}

lcdwritecmd(0x80 + i - 1);

lcdwritecmd(0x80 + j);

lcdwritedata(num / 256);

lcdwritedata(num % 256);

lcdwritecmd(0x36); //打开绘图显示

lcdwritecmd(0x30); //回到基本指令集

last\_num[i][j] = num;

}

void lcdsetcursor(unsigned char x, unsigned char y) //字符定位

{

unsigned char address;

if(y==0)

address=0x80+x;

else if(y==1)

address=0x90+x;

else if(y==2)

address=0x88+x;

else

address=0x98+x;

lcdwritecmd(address|0x80);

}

void lcdshowstr(unsigned char x, unsigned char y, unsigned char \*str) //字符输出

{

lcdsetcursor(x,y);

while((\*str)!='\0')

{

lcdwritedata(\*str);

str++;

}

}

void EXTI4\_15\_IRQHandler(void) // 中断控制模式

{

if (EXTI\_GetITStatus(0x2000) != 0x00) // 0x2000==GPIO\_PIN\_13

{

EXTI\_ClearITPendingBit(0x2000); //Clear interrupt flag bit

HAL\_GPIO\_TogglePin(GPIOC,GPIO\_PIN\_4);

}

else

{

lcdshowstr(1,1," ");

clear();

score=0;

food=0;

draw();

}

if (EXTI\_GetITStatus(0x0040) != 0x00) // 0x0040==EXTI\_PIN\_6

{

EXTI\_ClearITPendingBit(0x0040);

ms=1;

}

else if (EXTI\_GetITStatus(0x0080) != 0x00) // 0x0080==EXTI\_PIN\_7

{

EXTI\_ClearITPendingBit(0x0080);

ms=2;

}

else if (EXTI\_GetITStatus(0x0100) != 0x00) // 0x0100==EXTI\_PIN\_8

{

EXTI\_ClearITPendingBit(0x0100);

ms=3;

}

else if (EXTI\_GetITStatus(0x0200) != 0x00) // 0x0200==EXTI\_PIN\_9

{

EXTI\_ClearITPendingBit(0x0200);

ms=4;

}

else if (EXTI\_GetITStatus(0x0400) != 0x00) // 0x0400==EXTI\_PIN\_10

{

EXTI\_ClearITPendingBit(0x0400);

ms=5;

}

else if (EXTI\_GetITStatus(0x0800) != 0x00) // 0x0800==EXTI\_PIN\_11

{

EXTI\_ClearITPendingBit(0x0800);

ms=6;

}

else ;

if (EXTI\_GetITStatus(0x2000) == 0x00)

mods(); //对应模式小蛇初始化

}

⑧和贪吃蛇控制逻辑相关函数

int mods() //对应模式小蛇初始化

{

if(ms!=4)

{

tang[0][1].x=2;

tang[0][1].y=5;

tang[0][2].x=2;

tang[0][2].y=4;

tang[0][tot[0]=3].x=2;

tang[0][tot[0]=3].y=3;

f[0]=1; //玩家1初始方向 向右

for(int i=1;i<=tot[0];i++)

drowpoint(tang[0][i].x,tang[0][i].y,1);

}

if(ms==2||ms==3||ms==6)

{

tang[1][1].x=n;

tang[1][1].y=m-3;

tang[1][2].x=n;

tang[1][1].y=m-2;

tang[1][tot[1]=3].x=n;

tang[1][tot[1]=3].y=m-1;

f[1]=2; //电脑1初始方向 向左

for(int i=1;i<=tot[1];i++)

drowpoint(tang[1][i].x,tang[1][i].y,1);

}

if(ms==5||ms==4)

{ //电脑1

tang[2][1].x=n;

tang[2][1].y=m-3;

tang[2][2].x=n;

tang[2][1].y=m-2;

tang[2][tot[2]=3].x=n;

tang[2][tot[2]=3].y=m-1;

f[2]=2; //电脑1初始方向 向左

for(int i=1;i<=tot[2];i++)

drowpoint(tang[2][i].x,tang[2][i].y,1);

}

if(ms==4)

{

tang[3][1].x=2;

tang[3][1].y=5;

tang[3][2].x=2;

tang[3][2].y=4;

tang[3][tot[3]=3].x=2;

tang[3][tot[3]=3].y=3;

f[3]=1; //电脑2初始方向 向右

for(int i=1;i<=tot[3];i++)

drowpoint(tang[3][i].x,tang[3][i].y,1);

}

}

void draw() //显示边框

{

for(int i=1;i<=n+1;i++)

{

drowpoint(i,1,1); //\*\*\*\*

drowpoint(i,m+1,1);

}

for(int i=1;i<=m+1;i++)

{

drowpoint(1,i,1);

drowpoint(n+1,i,1);

}

}

void rand\_food() //没有食物时随机投喂

{

if(!vis[foodd[0].x][foodd[0].y]) //如果0号食物不见了

{

int x=rand()%(n+1),y=(rand()%m/2)\*2; //随机产生xy坐标y为偶数

while(!check(x,y))

x=rand()%(n+1),y=(rand()%m/2)\*2; //检验随机投喂的点，是否需要重新投喂

drowpoint(x,y,1);

food++;

vis[x][y]=1;

foodd[0].x=x;

foodd[0].y=y;

}

if(!vis[foodd[1].x][foodd[1].y]) //第二个食物--同上

{

int x=rand()%n,y=rand()%m;

while(!check(x,y))

x=rand()%n,y=(rand()%m);

drowpoint(x,y,1);

food++;

vis[x][y]=1;

foodd[1].x=x;

foodd[1].y=y;

}

}

int change(int t) //没有按键按下蛇的运行--第t条蛇

{

int x=tang[t][1].x,y=tang[t][1].y; //x和y是蛇头位置的xy

if(t==2||t==3) //2号和3号蛇是电脑

machine(t,x,y); //机器人寻路

x+=flg[f[t]][0],y+=flg[f[t]][1]; //蛇头(x,y)按规律变化

for(int i=tot[t];i;i--) //蛇的身子位置数据库刷新

{

tang[t][i+1]=tang[t][i];

drowpoint(tang[t][i].x,tang[t][i].y,1); //身子显示刷新

}

drowpoint(tang[t][tot[t]+1].x,tang[t][tot[t]+1].y,0); //原来的蛇尾消失

if(vis[x][y]) //如果吃掉一个食物的话

{

vis[x][y]=0,score+=(t==0||t==1),food--;

if(++tot[t]>=10) //有一方吃够分

result(t);

drowpoint(tang[t][tot[t]].x,tang[t][tot[t]].y,1); //尾巴处+1

}

fnd(t,x,y); //一方撞墙撞蛇则失败

tang[t][1].x=x; //把蛇头位置反馈到全局变量中

tang[t][1].y=y;

drowpoint(tang[t][1].x,tang[t][1].y,1); //显示蛇头

}

int check(int x,int y) //检验某一个点

{

if(x<1||x>n||y<2||y>m) //边界外返回0

return 0;

int t=1;

if(ms!=4)

for(int i=1;i<=tot[0];i++) //在蛇身上返回0

if(x==tang[0][i].x&&y==tang[0][i].y)

{t=0;break;}

if(ms==2||ms==3||ms==6)

for(int i=1;i<=tot[1];i++)

if(x==tang[1][i].x&&y==tang[1][i].y)

{t=0;break;}

if(ms==5||ms==4)

for(int i=1;i<=tot[2];i++)

if(x==tang[2][i].x&&y==tang[2][i].y)

{t=0;break;}

if(ms==4)

for(int i=1;i<=tot[3];i++)

if(x==tang[3][i].x&&y==tang[3][i].y)

{t=0;break;}

return t; //都不在这些点返回1

}

void machine(int t,int x,int y) //机器人寻路

{

int tt=0;

int foodid,minn=1e9,newf=f[t];

if((get\_food(x,y,foodd[0].x,foodd[0].y)<=get\_food(x,y,foodd[1].x,foodd[1].y)&&vis[foodd[0].x][foodd[0].y])||!vis[foodd[1].x][foodd[1].y])

foodid=0;

else

foodid=1;

for(int i=0;i<4;i++)

if(f[t]^other(i))

{

if(check(x+flg[i][0],y+flg[i][1]))

{

int now=get\_food(x+flg[i][0],y+flg[i][1],foodd[foodid].x,foodd[foodid].y);

if(now<minn)

newf=i,tt=1,minn=now;

if(!tt)

newf=i;

}

}

f[t]=newf;

}

void result(int t) //积分够了--胜利

{

if(ms==5) //人机对抗-抢分

if(t==0)

lcdshowstr(1,1,"Win!!!");

else

lcdshowstr(1,1,"Lose!!!");

else if(ms==4) //机机对抗-抢分

if(t==2)

lcdshowstr(1,1,"Robote1 Wine (Enough)");

else

lcdshowstr(1,1,"Robote2 Wine (Enough)");

else if(ms==6) //对抗模式-抢分

if(t==1)

lcdshowstr(1,1,"Player1 Wine (Enough)");

else

lcdshowstr(1,1,"Player2 Wine (Enough)");

else ;

while(1); //结束

}

void fnd(int t,int x,int y)

{

if(check(x,y)) return; //check(x,y)为0则相撞

if(ms==4) //机机对抗-抢分

if(t==3)

lcdshowstr(1,1,"Robot2 Die");

else

lcdshowstr(1,1,"Robot1 Die");

else if(ms==3||ms==6) //对抗模式 对抗模式-抢分

if(t==1)

lcdshowstr(1,1,"Player2 Die");

else

lcdshowstr(1,1,"Player1 Die");

else

{

if(t==2)

lcdshowstr(1,1,"Win!!!");

else

lcdshowstr(1,1,"Lose!!!");

//printf("\n 玩家分数:%d\n",score);

score=0;

}

while(1); //结束

}

int ads(int x) //取绝对值

{

return x<0?-x:x; //x是否<0,如果x<0的话就执行:前面的语句，如果x不小于0的话就执行:后面的语句

}

int other(int x) //03交换、12交换

{

if (x==0) return 3;

else if(x==1) return 2;

else if(x==2) return 1;

else return 0;

}

int get\_food(int x,int y,int xxx,int yyy) //辅助机器人找寻食物

{

return ads(x-xxx)+ads(y-yyy);

}

**第五步**：功能展示



**第六步**：总结展望

按键用GPIO\_INPUT的循环扫描效率相对不是很高（硬件反应时间长、代码冗余程度高），可以考虑一下换成中断的（但是需要考虑中断个数的占用问题）。中断占用情况：中断按键占用管脚PC13即13号中断，中断PC6→PC11占用了6→11号中断，4-15号中断集中在一起，可用的还有4、5、12、14、15号中断不够8个。方案一，考虑其他的单独中断。方案二，将模式选择设置为GPIO\_INPUT，放在while(ms==0)循环内改变ms的值，缺点是不能中途转换模式。（优先考虑方案二）

新增功能：可以在游戏结束后，延时一定时间，自动进入低功耗模式，可通过中断唤醒。