**推箱子的串口实现**

**第一步：**创建工程

“连接”→“LPUART1”→模式“异步”

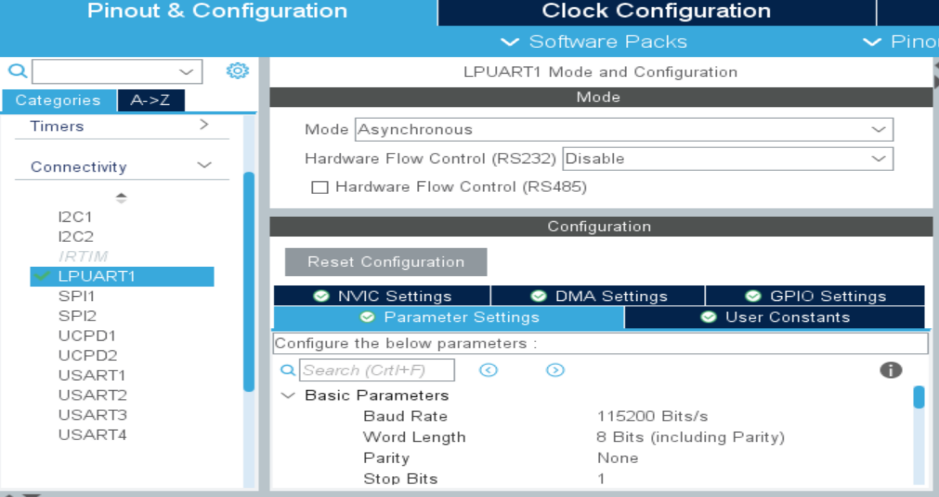


图7.4 Pinout & Configuration标签页面

如图所示，设置完成之后，他将默认选择PC0和PC1管脚。

然后，将其分别映射在PA2与PA3管脚。

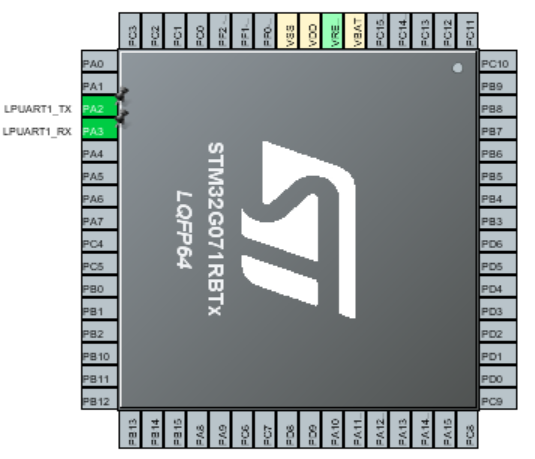


图7.6 重新分配完的串口引脚分配(视图90︒旋转)

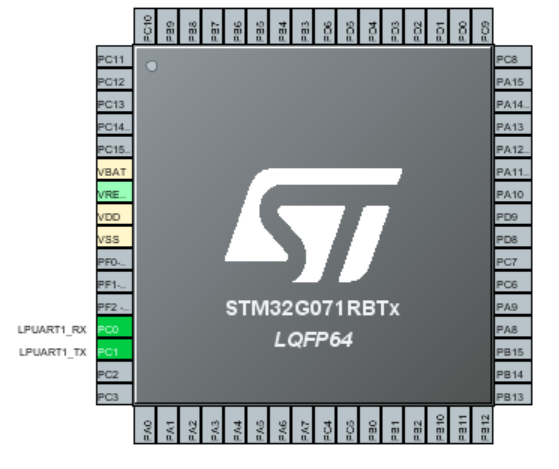


图7.5 默认的串口引脚分配

接下来配置时钟：在“时钟配置”选项卡下。

以最大时钟64MHz频率运行微控制器，即HCLK=64；届时，软件将自动为LPUART1配置为64Mhz。

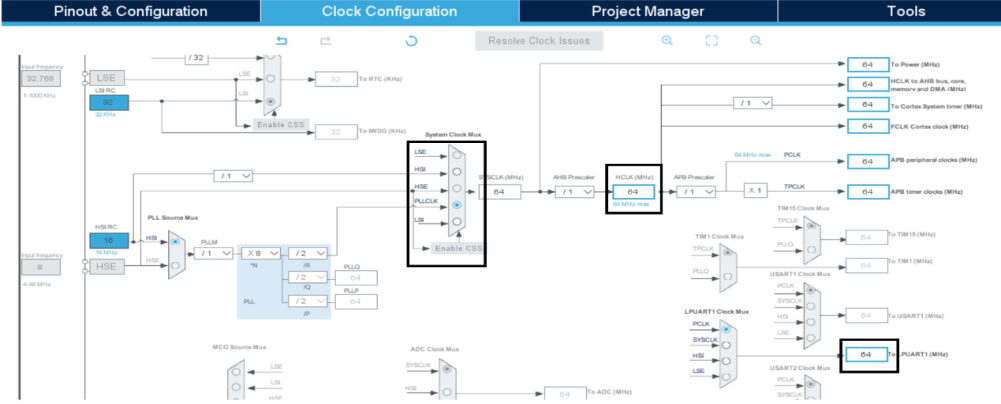


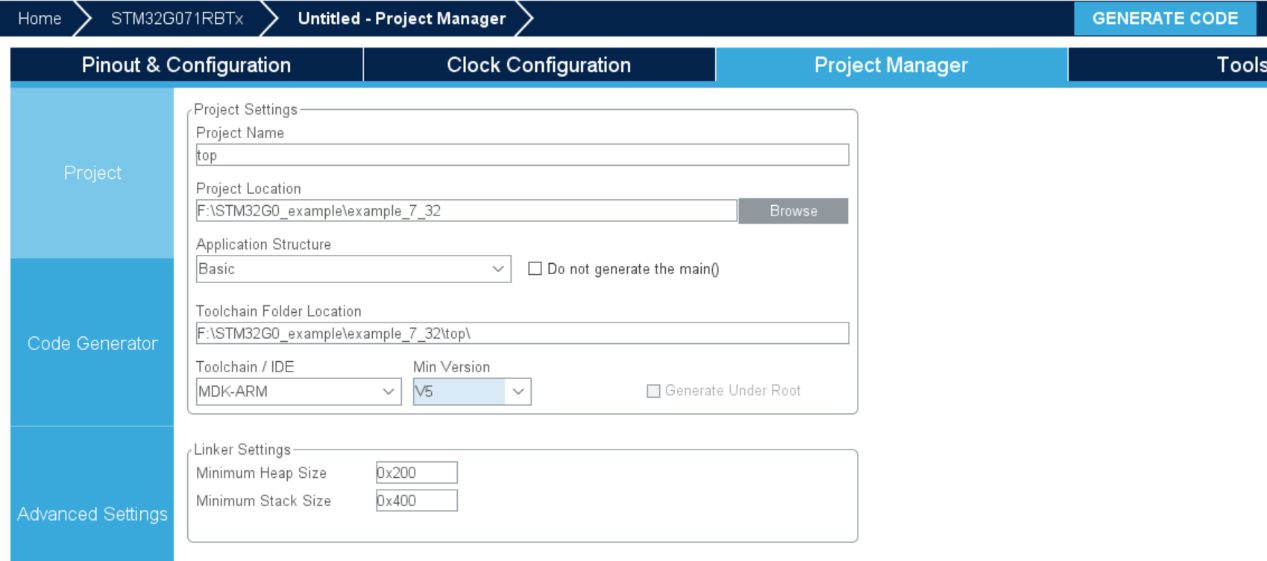
图7.7 Clock Configuration界面

接下来配置LPUART1：

波特率11520 字长8

奇偶校检无 停止位“1”

图7.8 设置输出工程的参数



**第二步：**生成代码

**第三步：**编写函数

**①**串口的重定向：

#include "stdio.h"

#include "string.h"

#define unit8\_t unsigned char

#define PUTCHAR\_PROTOTYPE int fputc(int ch,FILE \*f) //函数定义

#define GETCHAR\_PROTOTYPE int fgetc(FILE \*f)

#define BACKSPACE\_PROTOTYPE int \_backspace(FILE \*f)

PUTCHAR\_PROTOTYPE //重定向 fputc()函数

{

HAL\_UART\_Transmit(&hlpuart1,(unit8\_t\*) &ch,1,0xFFFF); //调用串口发送函数

return ch; //返回发送的字符

}

GETCHAR\_PROTOTYPE //重定向 fgetc()函数

{

uint8\_t value; //定义无符号字符型变量 value

while((LPUART1->ISR & 0x00000020)==0){} //判断串口是否接收到字符

value=(uint8\_t)LPUART1->RDR; //读取串口接收到的字符

HAL\_UART\_Transmit(&hlpuart1,(uint8\_t \*)&value,1,0x1000); //回显接收到的字符

return value; //返回接收到的值 value

}

BACKSPACE\_PROTOTYPE //重定向\_\_backspace 函数

{

return 0;

}

**②**推箱子运行逻辑相关函数定义与内容

void level\_pushbox(int k); //关卡地图初始化

void input\_pushbox(int ch); //输入按键判断逻辑

void supplement\_pushbox(); //目标点补充函数

void success\_pushbox(); //通关之后运行

void output\_pushbox(); //推箱子显示输出函数

void level\_pushbox(int GK) //关卡地图初始化

{

switch(GK)

{

case 1:

{

strcpy(a[0]," ### ");

strcpy(a[1]," #\*# ");

strcpy(a[2]," # # ");

strcpy(a[3],"####O###### ");

strcpy(a[4],"#\* OM O \*# ");

strcpy(a[5],"#####O##### ");

strcpy(a[6]," # # ");

strcpy(a[7]," #\*# ");

strcpy(a[8]," ### ");

strcpy(a[9]," ");

strcpy(a[10],"关卡 1/5 ");

strcpy(a[11]," ");

strcpy(a[12]," ");

strcpy(a[13]," ");

strcpy(a[14]," ");

Player\_Coordinates\_Y=4,Player\_Coordinates\_X=5;

Box1\_Destination\_Y=1,Box1\_Destination\_X=4;

Box2\_Destination\_Y=4,Box2\_Destination\_X=1;

Box3\_Destination\_Y=4,Box3\_Destination\_X=9;

Box4\_Destination\_Y=7,Box4\_Destination\_X=5;

}

break;

case 2:

{

strcpy(a[0],"###### ");

strcpy(a[1],"#\* # ");

strcpy(a[2],"### # ");

strcpy(a[3],"# O ###### ");

strcpy(a[4],"#\* OM O \*# ");

strcpy(a[5],"#####O##### ");

strcpy(a[6]," # # ");

strcpy(a[7]," #\*# ");

strcpy(a[8]," ### ");

strcpy(a[9]," ");

strcpy(a[10],"关卡 2/5 ");

strcpy(a[11]," ");

strcpy(a[12]," ");

strcpy(a[13]," ");

strcpy(a[14]," ");

Player\_Coordinates\_Y=4,Player\_Coordinates\_X=5;

Box1\_Destination\_Y=1,Box1\_Destination\_X=1;

Box2\_Destination\_Y=4,Box2\_Destination\_X=1;

Box3\_Destination\_Y=4,Box3\_Destination\_X=9;

Box4\_Destination\_Y=7,Box4\_Destination\_X=5;

}

break;

case 3:

{

strcpy(a[0]," #### ");

strcpy(a[1]," # # ");

strcpy(a[2]," # # ");

strcpy(a[3]," #M # ");

strcpy(a[4],"### ###### ");

strcpy(a[5],"# O O\*# ");

strcpy(a[6],"# O\* ### ");

strcpy(a[7],"#####\* O\*# ");

strcpy(a[8]," ###### ");

strcpy(a[9]," ");

strcpy(a[10],"关卡 3/5 ");

strcpy(a[11]," ");

strcpy(a[12]," ");

strcpy(a[13]," ");

strcpy(a[14]," ");

Player\_Coordinates\_Y=3,Player\_Coordinates\_X=3;

Box1\_Destination\_Y=6,Box1\_Destination\_X=3;

Box2\_Destination\_Y=5,Box2\_Destination\_X=8;

Box3\_Destination\_Y=7,Box3\_Destination\_X=5;

Box4\_Destination\_Y=7,Box4\_Destination\_X=8;

}

break;

case 4:

{

strcpy(a[0]," ######## ");

strcpy(a[1]," # ### ");

strcpy(a[2],"##O### # ");

strcpy(a[3],"#M O O # ");

strcpy(a[4],"# \*\*# O ## ");

strcpy(a[5],"##\*\*# # ");

strcpy(a[6]," ######## ");

strcpy(a[7]," ");

strcpy(a[8]," ");

strcpy(a[9]," ");

strcpy(a[10],"关卡 4/5 ");

strcpy(a[11]," ");

strcpy(a[12]," ");

strcpy(a[13]," ");

strcpy(a[14]," ");

Player\_Coordinates\_Y=3,Player\_Coordinates\_X=1;

Box1\_Destination\_Y=4,Box1\_Destination\_X=2;

Box2\_Destination\_Y=4,Box2\_Destination\_X=3;

Box3\_Destination\_Y=5,Box3\_Destination\_X=2;

Box4\_Destination\_Y=5,Box4\_Destination\_X=3;

}

break;

case 5:

{

strcpy(a[0]," #### ");

strcpy(a[1]," #\*\*# ");

strcpy(a[2]," ## \*## ");

strcpy(a[3]," # O\*# ");

strcpy(a[4],"## O ## ");

strcpy(a[5],"# #OO # ");

strcpy(a[6],"# M # ");

strcpy(a[7],"######## ");

strcpy(a[8]," ");

strcpy(a[9]," ");

strcpy(a[10],"关卡 5/5 ");

strcpy(a[11]," ");

strcpy(a[12]," ");

strcpy(a[13]," ");

strcpy(a[14]," ");

Player\_Coordinates\_Y=6,Player\_Coordinates\_X=3;

Box1\_Destination\_Y=1,Box1\_Destination\_X=3;

Box2\_Destination\_Y=1,Box2\_Destination\_X=4;

Box3\_Destination\_Y=2,Box3\_Destination\_X=4;

Box4\_Destination\_Y=3,Box4\_Destination\_X=5;

}

break;

case 0:

{

strcpy(a[0]," 通关攻略 ");

strcpy(a[1],"关卡 1/5 : ");

strcpy(a[2]," SS WW DDD AAAAAA DD WW ");

strcpy(a[3],"关卡 2/5 : ");

strcpy(a[4]," SS WW DDD AAAAAA DWW DWAA ");

strcpy(a[5],"关卡 3/5 : ");

strcpy(a[6]," SSSDDDWDA SSDAWWAASAW AASDDDWDS AAWWWD WWASSSS ");

strcpy(a[7],"关卡 4/5 : ");

strcpy(a[8]," DDDD SSDDWWA WWAAAA SSASDWDS WAWWDDDD SSAAADDD ");

strcpy(a[9]," WWAAAA SSASD WDDDDD SASAW DWAAA DDDWW AAAA SSS ");

strcpy(a[10]," WDDDDDD WAS AAAAA WWDDDD SDS AAAA DDDWW AAAASS");

strcpy(a[11],"关卡 5/5 : ");

strcpy(a[12]," AWW DDWW SSAASS DDWWW SSSDD WASAAA WWWDDD SAAA");

strcpy(a[13]," SSDD WW DWA SAWW SSASS DDDWW ");

strcpy(a[14]," ");

}

break;

}

}

void input\_pushbox(int ch) //输入按键判断逻辑

{

switch(ch) //按键判断逻辑

{

case 8:

if(a[Player\_Coordinates\_Y-1][Player\_Coordinates\_X]!='#') //上边不是墙

{

if(a[Player\_Coordinates\_Y-1][Player\_Coordinates\_X]!='O'&&a[Player\_Coordinates\_Y-1][Player\_Coordinates\_X]!='@') //上边不是箱子

{

a[Player\_Coordinates\_Y][Player\_Coordinates\_X]=' '; //当前位置人物消失

Player\_Coordinates\_Y--; //人物上移

a[Player\_Coordinates\_Y][Player\_Coordinates\_X]='M'; //人物出现

}

else if

(a[Player\_Coordinates\_Y-2][Player\_Coordinates\_X]!='#'&&a[Player\_Coordinates\_Y-2][Player\_Coordinates\_X]!='@'&&a[Player\_Coordinates\_Y-2][Player\_Coordinates\_X]!='O')

{ //上边是箱子 上上不是墙也不是箱子

a[Player\_Coordinates\_Y][Player\_Coordinates\_X]=' '; //同

Player\_Coordinates\_Y--;

a[Player\_Coordinates\_Y][Player\_Coordinates\_X]='M'; //上

a[Player\_Coordinates\_Y-1][Player\_Coordinates\_X]='O'; //箱子出现在人上方 推箱操作成功

if(((Player\_Coordinates\_Y-1==Box1\_Destination\_Y)&&(Player\_Coordinates\_X==Box1\_Destination\_X))

||((Player\_Coordinates\_Y-1==Box2\_Destination\_Y)&&(Player\_Coordinates\_X==Box2\_Destination\_X)) ||((Player\_Coordinates\_Y-1==Box3\_Destination\_Y)&&(Player\_Coordinates\_X==Box3\_Destination\_X))

||((Player\_Coordinates\_Y-1==Box4\_Destination\_Y)&&(Player\_Coordinates\_X==Box4\_Destination\_X)))

a[Player\_Coordinates\_Y-1][Player\_Coordinates\_X]='@';//判断箱子是否在目标点

}

}

break;

case 4:

if(a[Player\_Coordinates\_Y][Player\_Coordinates\_X-1]!='#')

{

if(a[Player\_Coordinates\_Y][Player\_Coordinates\_X-1]!='O'&&a[Player\_Coordinates\_Y][Player\_Coordinates\_X-1]!='@')

{

a[Player\_Coordinates\_Y][Player\_Coordinates\_X]=' ';

Player\_Coordinates\_X--;

a[Player\_Coordinates\_Y][Player\_Coordinates\_X]='M';

}

else if(a[Player\_Coordinates\_Y][Player\_Coordinates\_X-2]!='#'&&a[Player\_Coordinates\_Y][Player\_Coordinates\_X-2]!='@'&&a[Player\_Coordinates\_Y][Player\_Coordinates\_X-2]!='O')

{

a[Player\_Coordinates\_Y][Player\_Coordinates\_X]=' ';

Player\_Coordinates\_X--;

a[Player\_Coordinates\_Y][Player\_Coordinates\_X]='M';

a[Player\_Coordinates\_Y][Player\_Coordinates\_X-1]='O';

if(((Player\_Coordinates\_Y==Box1\_Destination\_Y)&&(Player\_Coordinates\_X-1==Box1\_Destination\_X))

||((Player\_Coordinates\_Y==Box2\_Destination\_Y)&&(Player\_Coordinates\_X-1==Box2\_Destination\_X))

||((Player\_Coordinates\_Y==Box3\_Destination\_Y)&&(Player\_Coordinates\_X-1==Box3\_Destination\_X))

||((Player\_Coordinates\_Y==Box4\_Destination\_Y)&&(Player\_Coordinates\_X-1==Box4\_Destination\_X)))

a[Player\_Coordinates\_Y][Player\_Coordinates\_X-1]='@';

}

}

break;

case 2:

if(a[Player\_Coordinates\_Y+1][Player\_Coordinates\_X]!='#')

{

if(a[Player\_Coordinates\_Y+1][Player\_Coordinates\_X]!='O'&&a[Player\_Coordinates\_Y+1][Player\_Coordinates\_X]!='@')

{

a[Player\_Coordinates\_Y][Player\_Coordinates\_X]=' ';

Player\_Coordinates\_Y++;

a[Player\_Coordinates\_Y][Player\_Coordinates\_X]='M';

} else if(a[Player\_Coordinates\_Y+2][Player\_Coordinates\_X]!='#'&&a[Player\_Coordinates\_Y+2][Player\_Coordinates\_X]!='@'&&a[Player\_Coordinates\_Y+2][Player\_Coordinates\_X]!='O')

{

a[Player\_Coordinates\_Y][Player\_Coordinates\_X]=' ';

Player\_Coordinates\_Y++;

a[Player\_Coordinates\_Y][Player\_Coordinates\_X]='M';

a[Player\_Coordinates\_Y+1][Player\_Coordinates\_X]='O';

if(((Player\_Coordinates\_Y+1==Box1\_Destination\_Y)&&(Player\_Coordinates\_X==Box1\_Destination\_X))

||((Player\_Coordinates\_Y+1==Box2\_Destination\_Y)&&(Player\_Coordinates\_X==Box2\_Destination\_X))

||((Player\_Coordinates\_Y+1==Box3\_Destination\_Y)&&(Player\_Coordinates\_X==Box3\_Destination\_X))

||((Player\_Coordinates\_Y+1==Box4\_Destination\_Y)&&(Player\_Coordinates\_X==Box4\_Destination\_X)))

a[Player\_Coordinates\_Y+1][Player\_Coordinates\_X]='@';

}

}

break;

case 6:

if(a[Player\_Coordinates\_Y][Player\_Coordinates\_X+1]!='#')

{ if(a[Player\_Coordinates\_Y][Player\_Coordinates\_X+1]!='O'&&a[Player\_Coordinates\_Y][Player\_Coordinates\_X+1]!='@')

{

a[Player\_Coordinates\_Y][Player\_Coordinates\_X]=' ';

Player\_Coordinates\_X++;

a[Player\_Coordinates\_Y][Player\_Coordinates\_X]='M';

}

else If

(a[Player\_Coordinates\_Y][Player\_Coordinates\_X+2]!='#'&&a[Player\_Coordinates\_Y][Player\_Coordinates\_X+2]!='@'&&a[Player\_Coordinates\_Y][Player\_Coordinates\_X+2]!='O')

{

a[Player\_Coordinates\_Y][Player\_Coordinates\_X]=' ';

Player\_Coordinates\_X++;

a[Player\_Coordinates\_Y][Player\_Coordinates\_X]='M';

a[Player\_Coordinates\_Y][Player\_Coordinates\_X+1]='O'; if(((Player\_Coordinates\_Y==Box1\_Destination\_Y)&&(Player\_Coordinates\_X+1==Box1\_Destination\_X))

||((Player\_Coordinates\_Y==Box2\_Destination\_Y)&&(Player\_Coordinates\_X+1==Box2\_Destination\_X))

||((Player\_Coordinates\_Y==Box3\_Destination\_Y)&&(Player\_Coordinates\_X+1==Box3\_Destination\_X))

||((Player\_Coordinates\_Y==Box4\_Destination\_Y)&&(Player\_Coordinates\_X+1==Box4\_Destination\_X)))

a[Player\_Coordinates\_Y][Player\_Coordinates\_X+1]='@';

}

}

break;

default:

printf("\r\n 违规操作！！！ \r\n");; //按下其他按键不响应

break;

}

}

void supplement\_pushbox() //目标点补充函数

{

if(a[Box1\_Destination\_Y][Box1\_Destination\_X]==' ')

a[Box1\_Destination\_Y][Box1\_Destination\_X]='\*';

if(a[Box2\_Destination\_Y][Box2\_Destination\_X]==' ')

a[Box2\_Destination\_Y][Box2\_Destination\_X]='\*';

if(a[Box3\_Destination\_Y][Box3\_Destination\_X]==' ')

a[Box3\_Destination\_Y][Box3\_Destination\_X]='\*';

if(a[Box4\_Destination\_Y][Box4\_Destination\_X]==' ')

a[Box4\_Destination\_Y][Box4\_Destination\_X]='\*';

}

void success\_pushbox() //通关之后运行

{

printf("\r\n！！！恭喜通关！！！\r\n");

printf("\r\n 密码是 19991015 \r\n");

for (float y = 1.3f; y > -1.0f; y -= 0.1f) //画爱心图

{

for (float x = -1.5f; x < 1.5f; x += 0.05f)

{

float a = x \* x + y \* y - 1;

putchar(a \* a \* a - x \* x \* y \* y \* y <= 0.0f ? '\*' : ' ');

}

printf("\r\n");

}

}

void output\_pushbox() //推箱子显示输出函数

{

int i;

int j;

printf("\r\n");

for(i=0;i<15;i++)

{

for(j=0;j<50;j++)

{

printf("%c",a[i][j]);

}

HAL\_Delay(20); //必须有延时

printf("\r\n");

}

}

**第四步**：添加主函数

①相关初始化

HAL\_Init();

SystemClock\_Config();

MX\_GPIO\_Init();

MX\_LPUART1\_UART\_Init();

②判断是否显示通关攻略

printf("\r\n 请输入密码查看通关攻略: \r\n");

scanf("%d",&PSWD);

HAL\_Delay(100);

printf("\r\n Receive PSWD=%d",PSWD);

if(PSWD==19991015) //判断密码是否正确

{

level\_pushbox(0);

output\_pushbox();

}

else

printf("\r\n 密码错误 \r\n");

③判断是否解锁全部关卡

printf("\r\n 请输入密码解锁所有关卡: \r\n");

scanf("%d",&PSWD);

HAL\_Delay(100);

printf("\r\n Receive PSWD=%d",PSWD);

if(PSWD==19991015) //判断密码是否正确

{

p=5;

printf("\r\n 密码正确，所有关卡已解锁！！！ \r\n");

}

else

printf("\r\n 密码错误，解锁失败。 \r\n");

④判断是否预览通关画面

printf("\r\n 请输入密码预览通关结果: \r\n");

scanf("%d",&PSWD);

HAL\_Delay(100);

printf("\r\n Receive PSWD=%d",PSWD);

if(PSWD==19991015) //判断密码是否正确

success\_pushbox();

else

printf("\r\n 密码错误 \r\n");

⑤关卡选择部分

do

{

printf("请选择关卡(1-5): \r\n");

scanf("%d",&GK);

HAL\_Delay(50);

printf("\r\n");

HAL\_Delay(50);

if (GK<1||GK>5)

printf("关卡选择错误 \r\n");

else if(GK>p)

printf("禁止白嫖 \r\n");

}while(GK<1||GK>5||GK>p);

⑥游戏运行过程

while(GK>=1&&GK<=5) //游戏执行

{

level\_pushbox(GK); //关卡初始化函数

output\_pushbox(); //初始状态显示

while(1) //运行模块

{

int ch;

printf("请执行操作：");

scanf("%d",&ch);

input\_pushbox(ch); //输入按键判断逻辑

supplement\_pushbox(); //目标点补充函数

output\_pushbox(); //状态显示刷新

if(a[Box1\_Destination\_Y][Box1\_Destination\_X]=='@'&&a[Box2\_Destination\_Y][Box2\_Destination\_X]=='@' &&a[Box3\_Destination\_Y][Box3\_Destination\_X]=='@'&&

a[Box4\_Destination\_Y][Box4\_Destination\_X]=='@')

{

GK++;

if(GK>=p) p=GK; //选关限制----即检查点

HAL\_Delay(150);

break; //确认获胜则跳出while(1)循环

}

}

}

⑦通关之后的奖励机制

success\_pushbox();//通关之后运行

**第四步：**编译下载程序

**第五步：**打开串口助手运行程序

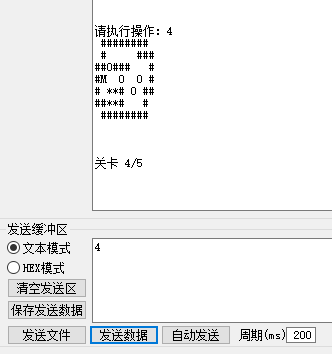
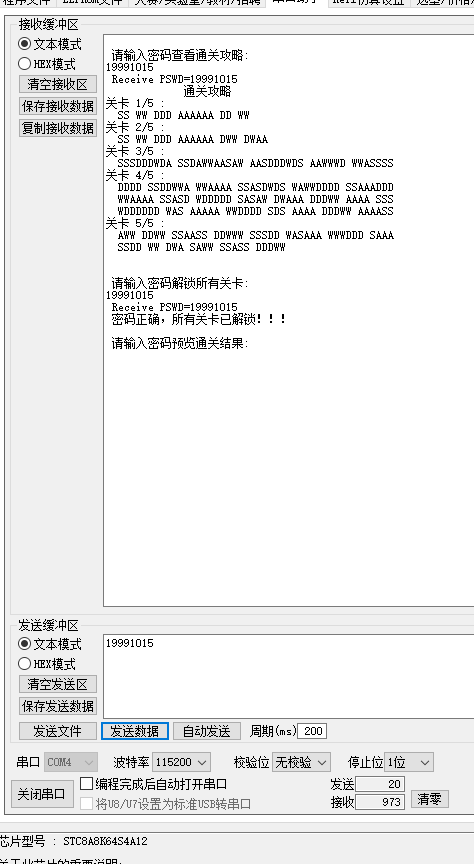
①串口设：选择合适串口（在设备管理器中查看端口）；波特率“115200”；校检位“无校检”；停止位“1位”。

②点击打开串口。

③按下开发板的“复位”按键。

④根据提示向串口发送数据即可。

**第六步：**结果分享



**第七步：**问题及其解决总结

①跳转函数 longjmp不能用——用do-while语句和if组合将其代替。

②不能远距离跳跃，例如子函数某个位置跳转到主函数某个位置——用顺序结构搭配少许循环、分支指令代替。适当改变其功能。

③重定位后的字符输入有问题，容易卡死——方向键的实现用数字“8462”代替字符“WASD”。

④原密码用字符串来判断——现在用double类型数字代替[优点：代码更简单/缺点：密码不能包含数字以外其他字符]

⑤误输入字符卡死——输出提示“违规操作”，提示用户“非程序BUG”按下复位键。

⑥输入输出不理想，赋值之后无更改——在变量定义前加关键字“volatile”，它是一种类型修饰符，用它声明的类型变量表示可以被某些编译器未知的因素更改，比如：操作系统、硬件或者其它线程等。

⑦【注意】数字输入的时候，发送完一个数字，必须以回车键结束。

**第八步：**展望

字符输入问题的解决：回想之前做过的单片机STC8A8K64S4A12串口通信中字符输入是没问题的。仔细回顾，并移植到ARM即可。