



**2024-2025学年第一学期课程设计 说明书**

**学院： 信息工程学院**

**课程： 单片机课程设计**

**题目： 密码锁设计**

**姓名： 赵文晨**

**学号： ？？？？？？**

**专业班级： 电子221**

**指导教师： 向菲**

**课程设计任务书**

课程设计名称 单片机课程设计 学生姓名 赵文晨 专业班级 电子221

设计题目 密码锁设计

1. 课程设计目的

1、综合运用单片机技术，独立完成一个课题的设计，考察运用所学知识，解决实际问题的能力；

2、结合理论知识，考察阅读参考资料、文献、手册的能力；

3、进一步熟悉单片机技术的开发流程，掌握器件选型、电路匹配、SFR设定、程序编辑调试、代码下载验证等环节的实现方法和应用技巧；

4、锻炼撰写研究报告、研究论文的能力；

5、通过本实践环节，培养科学和严谨的工作作风。

1. 设计内容、技术条件和要求

1、设计密码锁控制电路，具有上锁键和开锁键；

2、上锁前先按上锁键，自设密码（3位），开锁时先按开锁键，然后输入上锁时设定的密码开锁；

3、发光二极管的亮灭代表开锁和上锁；

4、密码错误报警，三次输入密码错误锁定密码锁。

三、时间进度安排

第1周： (1) 完成设计准备，确定实施方案；

(2) 完成电路的硬件设计；

(3) 完成控制程序设计流程。

第2周： (1) 完成控制程序编辑、编译及调试，生成可下载代码；

1. 完成文件至器件的下载，并进行功能验证；

(3) 撰写设计说明书。

四、主要参考文献

[1] 王守中. 51单片机开发入门与典型实例[M]. 北京:人民邮电出版社,2007

[2] 彭为, 黄科. 单片机典型系统设计[M]. 北京:电子工业出版社,2006

[3] 求是科技. 单片机典型外围器件及应用实例[M]. 北京:人民邮电出版社,2006

[4] 戴佳. 51单片机C语言应用程序设计[M]. 北京:电子工业大学出版社,2006

指导教师签字： 2024年12月6日

**密码锁设计**

前　言

在当今社会，密码锁作为一种便捷且高效的安全防护手段，已广泛应用于各类场合，从家庭住宅到商业场所，乃至高端电子设备中，均可见其身影。本项目旨在利用单片机为核心控制器，结合键盘输入、显示模块、存储单元及执行机构等外围设备，设计并实现一个功能完善的电子密码锁系统。通过这一实践过程，我们不仅能够学习单片机的基本编程、接口设计、电路搭建等关键技术，还能深入探索密码学基础、数据加密与验证算法等高级主题，为将来的专业学习和职业发展打下坚实的基础。

第1章 题目及要求

1.1 课程设计题目

密码锁设计

1.2 主要技术指标要求

1、设计密码锁控制电路，具有上锁键和开锁键；

2、上锁前先按上锁键，自设密码（3位），开锁时先按开锁键，然后输入上锁时设定的密码开锁；

3、发光二极管的亮灭代表开锁和上锁；

4、密码错误报警，三次输入密码错误锁定密码锁。

第2章 自主拓展功能介绍

1.密码位数增加至6位。

2.扩展了更改密码程序，只有开锁状态才可以更改密码。

3.增加了电磁门锁外设，可以同步密码锁程序的开锁和上锁。

4.加上了语音控制芯片，设置了开锁语音“主人你回来啦，我好想你呀”，密码第一次错误语音“你不是我的主人，你是个坏蛋”，密码第二次错误语音“你不要过来，你不要过来呀”，密码锁锁定歌曲【奇迹再现】，通过功放喇叭播放声音。

第3章 设计电路及硬件介绍

**3.0 引脚接线**

3.0.1清翔51开发板：

数码管使用P0，LED使用P1，矩阵键盘使用P3，蜂鸣器P2.3,数码管位选P2.7,数 码管段选P2.6。

3.0.2外设拓展：

语音模块SPK+接音响正极，SPK-接音响负极，DC-5V接单片机5V0引脚，GND接单 片机GND引脚，RX接单片机P3.1引脚。继电器输入端VCC接单片机3V3引脚，GND 接单片机GND引脚，IN接单片机P1.0引脚，继电器输出端COM接单片机5V0引脚， NC接电磁门锁正极，电磁门锁负极接单片机GND引脚。

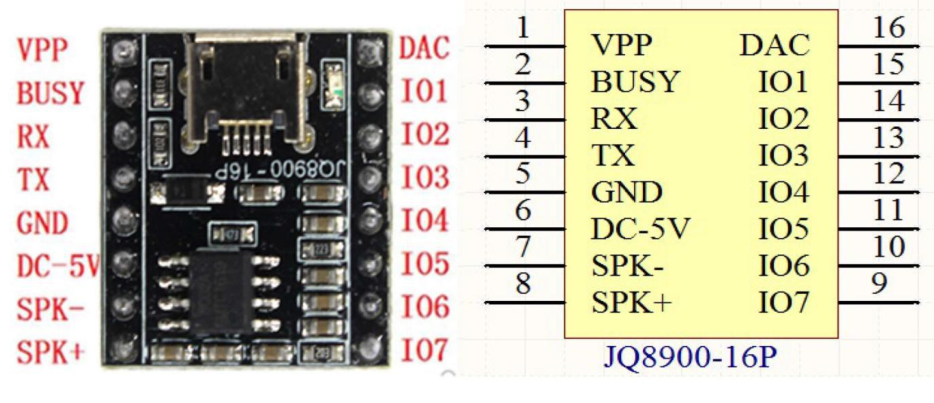
**3.1 JQ8900-16P语音模块**

**3.1.1 模块简介**

JQ8900-16P选用的是SOC方案，集成了一个16位的MCU,以及一个专门针对音频解码的ADSP（Audio Digital Signal Processing），ADSP解码器是指音频数字信号处理解码器，它通常用于处理音频数据，将其从压缩格式解码成原始音频数据，以便播放。ADSP 解码器可以是硬件实现的，也可以是软件实现的。采用硬解码的方式，更加保证了系统的稳定性和音质。小巧尺寸更加满足嵌入其它产品的需求。且模块内部含有一个FLASH存储器，用于对语音音频进行存储。开发板利用串口UART1与JQ8900-16P模块进行通信，可通过发送协议约定的通信指令控制语音模块利用SPI总线读写FLASH的方式播报指定的语音音频。

**3.1.2 硬件参数**

**3.1.3管脚说明**

**3.1.4 两线串口通信控制**

芯片通信采用全双工串口通信，波特率9600，停止位1位，检验位N。

部分控制指令：  
 指定播放1-5首

AA 07 02 00 01 B4

AA 07 02 00 02 B5

AA 07 02 00 03 B6

AA 07 02 00 04 B7

AA 07 02 00 05 B8

音量减 AA 15 00 BF

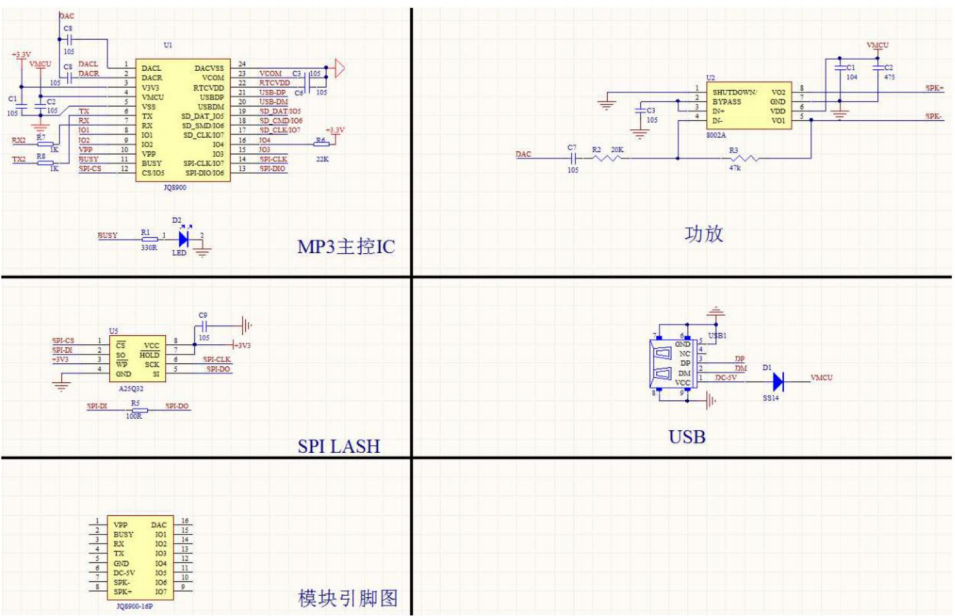
音量加 AA 14 00 BE

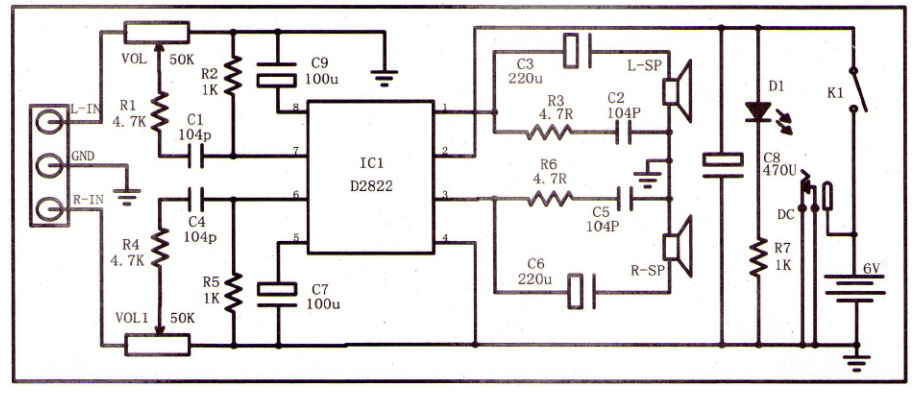
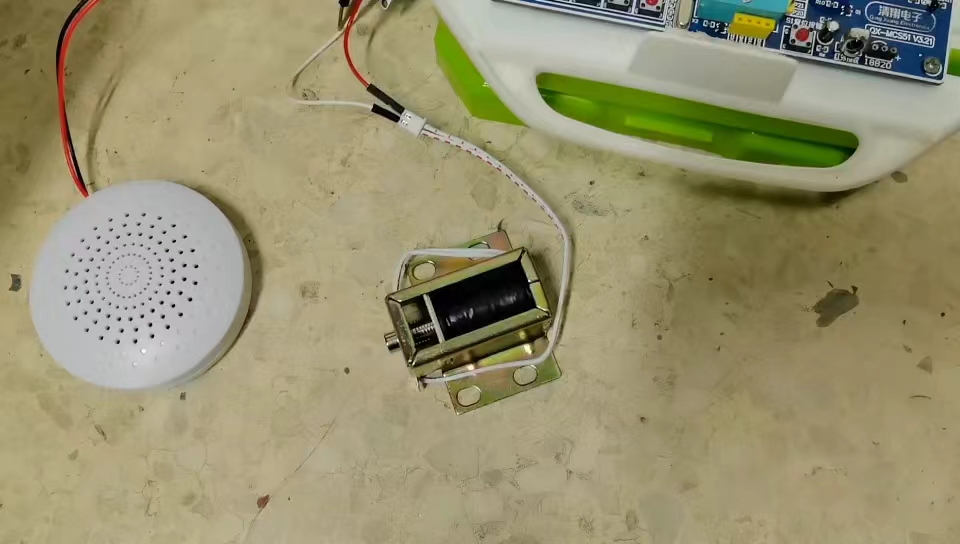
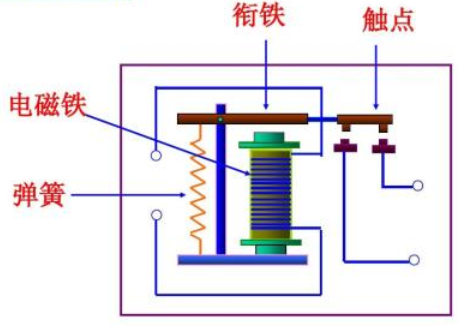
停止 AA 04 00 AE

暂停 AA 03 00 AD

结束播放 AA 10 00 BA

**3.1.5 原理图**

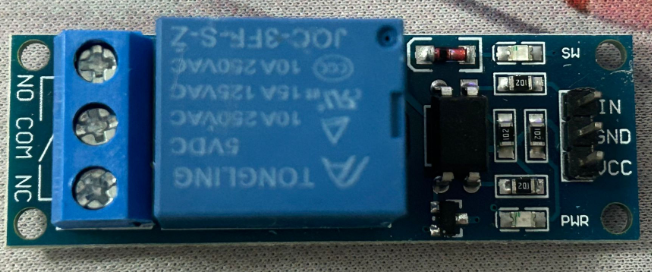
**3.2 音响原理图**(实际只有单声道)

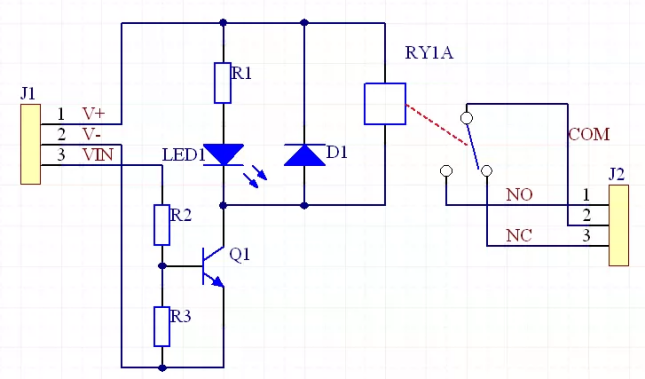
**3.3 电磁门锁**

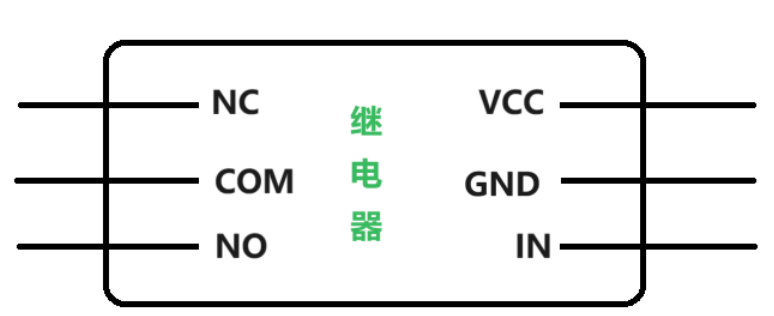
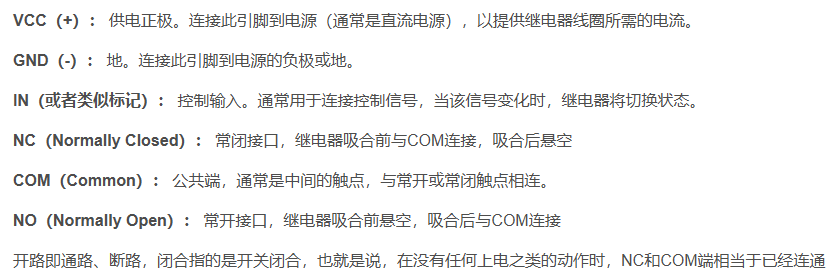
**3.4 继电器**

原理：电磁感应

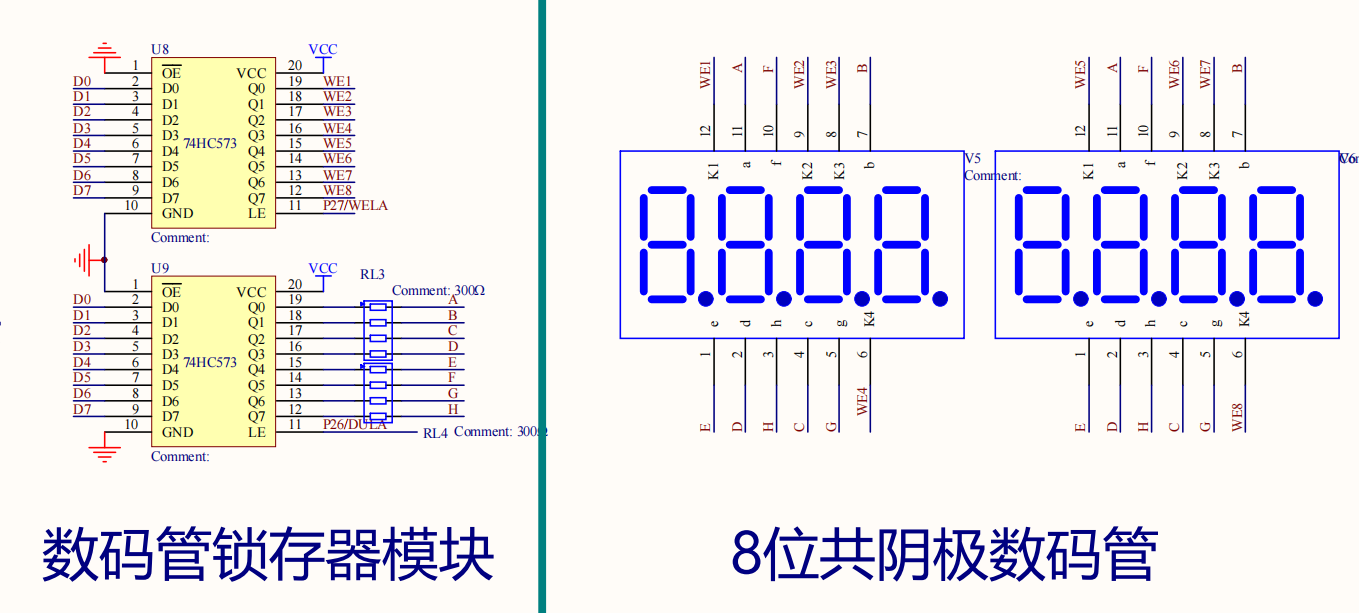
通过继电器可以实现小电流控制大电流

**3.4.1原理图**

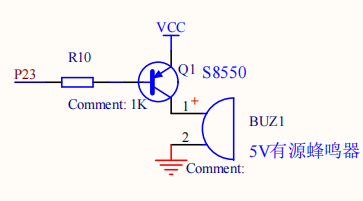
3.4.2引脚图



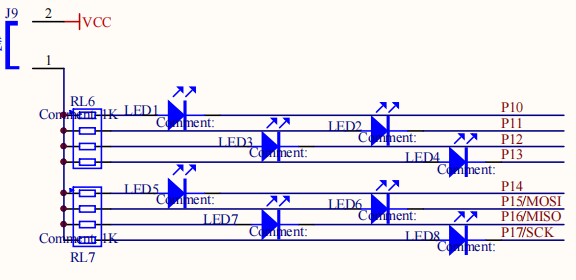
3.5 数码管原理图



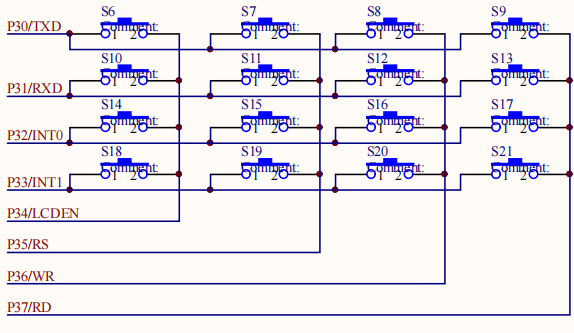
3.6 蜂鸣器原理图



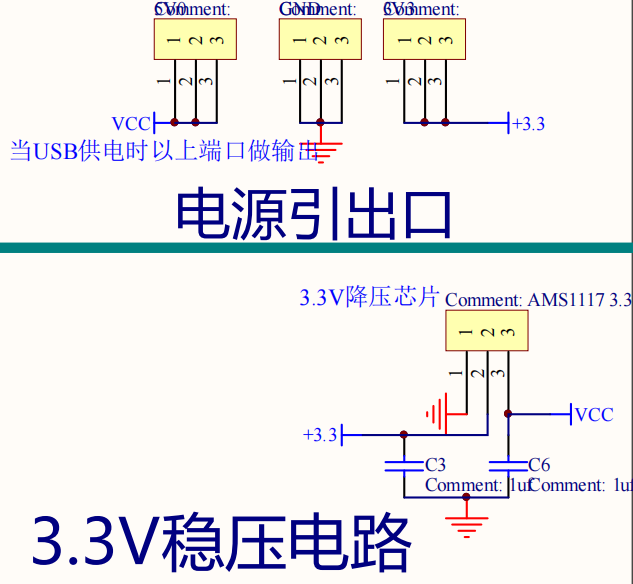
3.7 LED原理图



3.8 矩阵键盘原理图

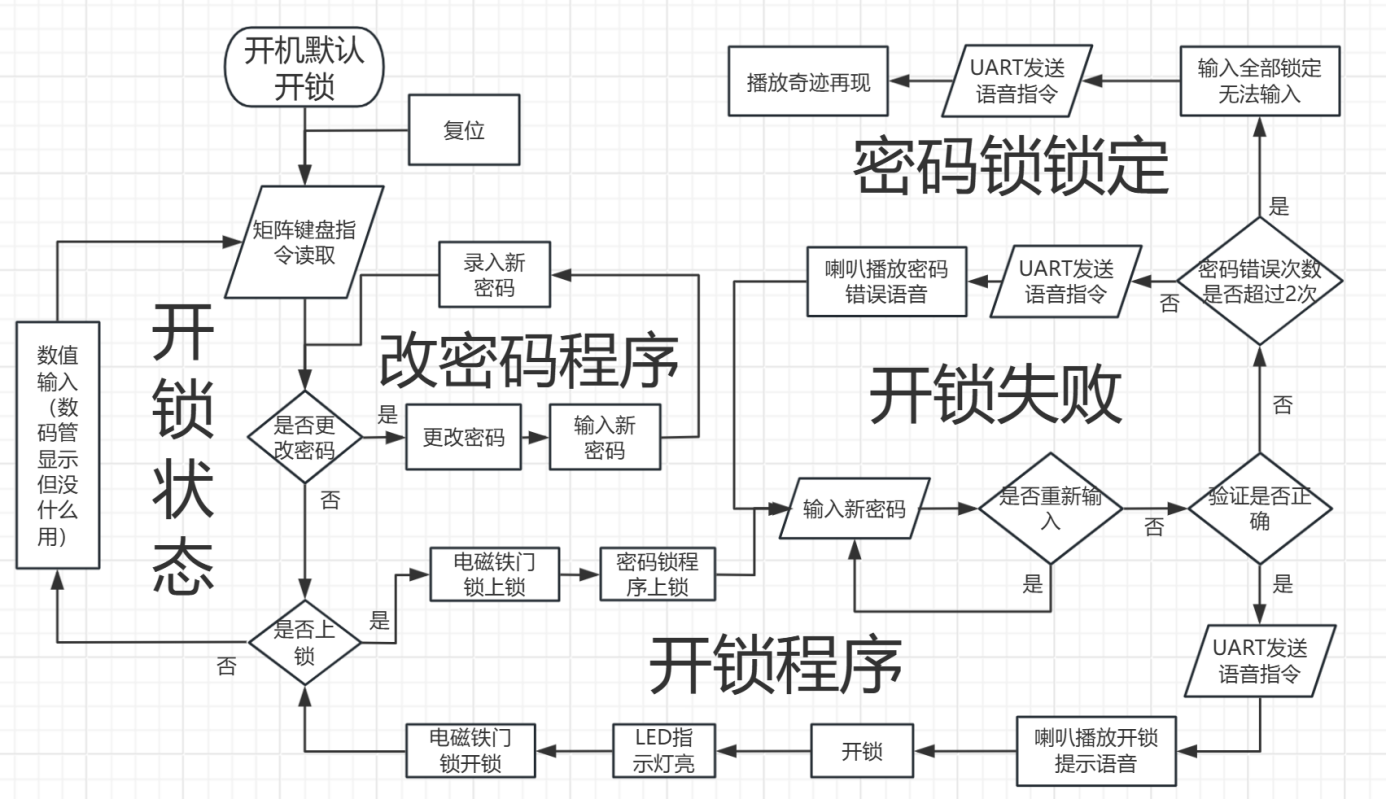


3.9 电源引出口



第4章 软件流程介绍

4.1 总体程序流程

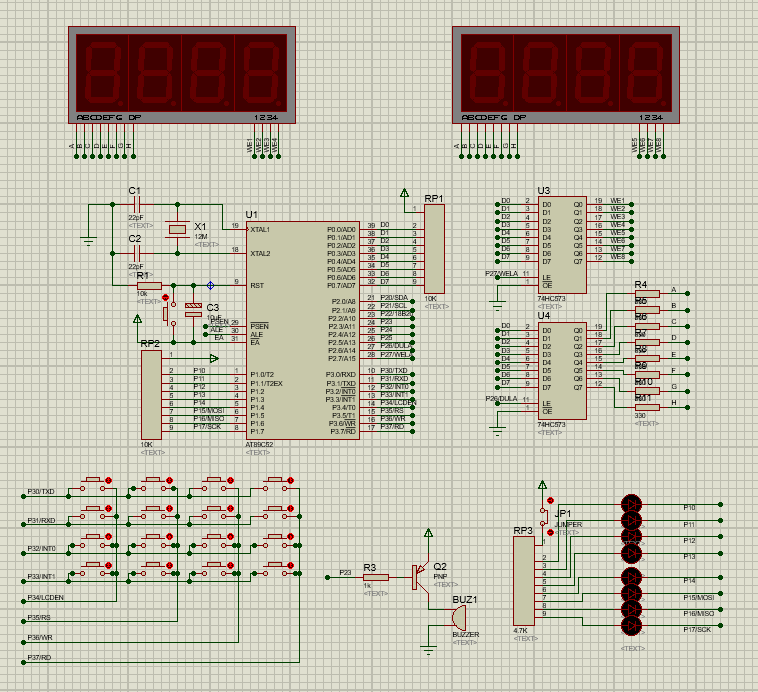


注意： 1.只有开锁状态才能改密码程序

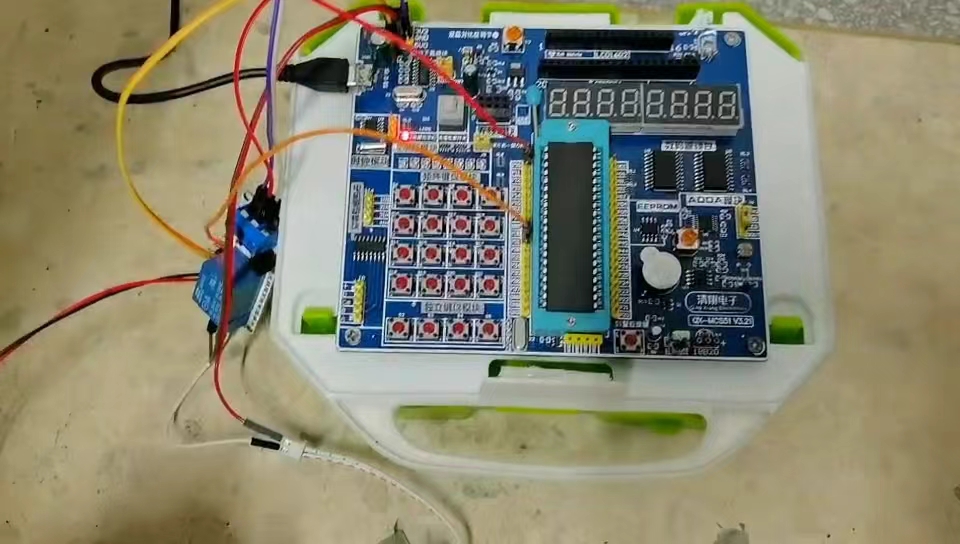
2.是否上锁指的是上锁键是否被按下，开锁状态（改密码和锁定除外）不管程 序到哪里下按上锁键可直接上锁，上锁状态就是程序已经上锁。

1. 仿真及实测图

5.1仿真图



5.2实测图





第6章 遇到困难及解决方法

6.1 没错误但是keil无法生成hex文件

写好的程序用keil，编译0错误0警告，组建却报错，但是程序并没有逻辑性的错误。

\*\*\* ERROR L107: ADDRESS SPACE OVERFLOW  
 SPACE: DATA  
 SEGMENT: ?DT?\_WRITEONECHAR?  
 LENGTH: 0002H

**解决办法：**

一般keil在small模式。small 模式下未指存储类型的变量默认为data型，即直接寻址，只能访问低 128 个字节，但这 128 个字节也不是全为你程序所有，寄存器 R0-R7必须映射到低RAM，要占去 8 个字节，如果使用寄存组切换，占用的更多。可以通过删除程序中的无关变量使data减小到128以下，或者把keil编译模式small更改成large，这样data上限可以超过128，就不会报错了。

6.2电磁铁门锁无法工作

单片机IO口驱动能力有限，一般20－25mA，多个IO口加起来总电流有限制，根据厂家和封装不同而变化，一般不能超300mA,无法直接带动电磁铁门锁。

**解决办法：**

电磁铁门锁本质上就是一个电磁线圈，工作电流较大，借助电磁继电器，将电磁继电器的输入端接单片机IO口，通过输出端来控制电磁铁门锁的通电与断电，相当于把继电器当成一个开关，实现小电流控制大电流。

6.3程序经常自动复位，数码管乱码闪烁

整个密码锁系统包括单片机和外设（继电器，电磁门锁，语音控制芯片和功放喇叭），所需的供电均由一根USB数据线提供，他们一起工作时电流比较大，总电流上限较低，使得供给单片机的电流较小，无法支撑单片机执行程序，就会出现程序执行不稳定导致复位以及数码管乱码闪烁。

**解决办法：**

将系统供电的USB口更换为手机充电适配器的USB口，而不是电脑USB口，手机充电适配器输入端为220V~50Hz交流电，输出端电压虽然还是5V,但是输出电流上限更高，电流上限更高可以保证单片机工作电流供给足够，程序以及各种模块外设才能稳定工作。

6.4改密码要改程序

实际应用的密码锁，客户想要改密码还要重新下载程序，这是非常不方便的，市场竞争力太低。

**解决办法：**

将原来程序设定的默认密码更改为一个可以通过外设更改的数组，当触发更改 密码程序时，可以用新的数组覆盖原来的密码数组，实现更改密码的目的。

设计总结

1.我认为目前我做的51密码锁跟市场上的密码锁相比有一个不足点，就是一个密码锁并非只能有一个密码，可以是多个密码，或者是伪装密码，不同密码权限不一样，不同的人用不同的密码，就算改密码也只是更改自己的密码，不影响其他密码，甚至可以有记录功能，什么时间哪个密码开了锁，这是可以提升完善的地方。

2.我感觉51单片机性能不怎么好，首先5它只有一个定时器，1个I2C,1个SPI,1个UART，我这个功能要用，另一个功能就用不了，能扩展的外设太少了。比如数码管和1602LCD显示屏共用P0端口，这就意味着我不能同时使用它们。我扩展了1个语音模块，用的是UART通信，我再想扩展功能（比如视觉模块），如果还是UART和单片机通信，那我就没地方用了了。其次51晶振频率太低，我想扩展WS2812编程灯带，但是WS2812对时序要求比较严格，需要到ns级别，51单片机太吃力了，带不动。

3.还是有收获的，锻炼了我的编程能力和实际应用能力，各个模块之间的配合使用。

参考文献

[1]李全利,单片机原理及接口技术[M1.高等教育山版社，2003

[2]PROTEL99SE电路设计与制板[M],机械工业出版社，2007

[3]杨将新，李华军，刘到骏等.单片机程序设计及应用(从基础到实践).电子工业出版社

[4]魏立峰，王宝兴,单片机原理与应用技术.北京:北京大学出版社，2006:11-13

[5]刘鳀，孙春亮,单片机C语言入门,北京:人民邮电出版社:2010:11-48

[6]李建波,STC89C52单片机的电子密码锁,现代电子技术，2008，Vo1.286

[7]王为清，程国刚,单片机KeilCx51应用开发技术,北京:人民邮电出版社，2007;11-70

附 录

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

【编写时间】： 2024年12月16日

\* 【作 者】： 电子221赵文晨

\* 【版 本】： 2.1

\* 【实验平台】： QX-MCS51 单片机开发板

\* 【外部晶振】： 11.0592mhz

\* 【主控芯片】： STC89C52

\* 【编译环境】： Keil μVisio4

\* 【程序功能】： 密码锁

\* 【使用说明】：

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

功能键

S6---S15数字键0-9 S16---更改密码

S17---更改密码完毕后确认 S18---重试密码、重新设定

S19---关闭密码锁 S20---验证密码

初始密码：000000 密码位数：6位

注意：掉电后，所设密码会丢失，重新上点时，密码恢复为原始的000000。与P1相连的8位发光LED点亮代表锁被打开；熄灭代表锁被锁上

程序功能：

1、开锁：

下载程序后，直接按六次S6（即代表数字0），8位LED亮，锁被打开，输入密码时，六位数码管依次显示小横杠。

2、更改密码：只有当开锁（LED亮）后，该功能方可使用。首先按下更改密码键S16，然后设置相应密码，此时六位数码管会显示设置密码对应的数字。最后设置完六位后，按下S17确认密码更改，此后新密码即生效。

3、重试密码：当输入密码时，密码输错后按下键S18，可重新输入六位密码。 当设置密码时，设置中途想更改密码，也可按下此键重新设置。

4、关闭密码锁：按下S19即可将打开的密码锁关闭。确认密码更改---按S19关闭密码锁---输入新的密码打开密码锁

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include<reg52.h>

#define uchar unsigned char

#define uint unsigned int

#define jingzhen 11059200UL /\* 使用22.1184MHz晶振 \*/

#define botelv 9600UL /\* 波特率设置为9600 \*/

uchar old1,old2,old3,old4,old5,old6; //原始密码000000

uchar new1,new2,new3,new4,new5,new6; //每次MCU采集到的密码输入

uchar a=16,b=16,c=16,d=16,e=16,f=16; //送入数码管显示的变量

uchar wei,key,temp,confirm;

uchar Number\_of\_errors=0,i=0;

uint ii;

unsigned char sound1[4] = {0xAA, 0x15, 0x00, 0xBF}; // 音量-

unsigned char sound2[4] = {0xAA, 0x14, 0x00, 0xBF}; // 音量+

unsigned char end[4] = {0xAA, 0x10, 0x00, 0xBA}; // 音量+end

unsigned char song1[6] = {0xAA, 0x07, 0x02, 0x00, 0x01, 0xB4}; // 第1首

unsigned char song2[6] = {0xAA, 0x07, 0x02, 0x00, 0x02, 0xB5}; // 第2首

unsigned char song3[6] = {0xAA, 0x07, 0x02, 0x00, 0x03, 0xB6}; // 第3首

unsigned char song4[6] = {0xAA, 0x07, 0x02, 0x00, 0x04, 0xB7}; // 第4首

unsigned char song5[6] = {0xAA, 0x07, 0x02, 0x00, 0x05, 0xB8}; // 第5首

unsigned char song6[6] = {0xAA, 0x07, 0x02, 0x00, 0x06, 0xB9}; // 第6首

unsigned char song7[6] = {0xAA, 0x07, 0x02, 0x00, 0x07, 0xBA}; // 第7首

unsigned char song8[6] = {0xAA, 0x07, 0x02, 0x00, 0x08, 0xBB}; // 第8首

volatile unsigned char sending;

volatile unsigned char index = 0; // 当前发送的索引

bit allow,genggai,ok,wanbi,retry,close; //各个状态位

sbit S2=P3^0;

sbit S3=P3^1;

sbit S4=P3^2;

sbit S5=P3^3;

sbit dula=P2^6;

sbit wela=P2^7;

sbit beep=P2^3;

void init(void) // 串口初始化

{

EA=0; // 暂时关闭中断

TMOD &= 0x0F; // 定时器1工作在模式2

TMOD |= 0x20; // 定时器1工作在模式2，自动重装

SCON=0x50; // 串口工作在模式1

TH1=256-jingzhen/(botelv\*12\*16); // 计算定时器重装载值

TL1=256-jingzhen/(botelv\*12\*16);

PCON|=0x80; // 波特率加倍

ES=1; // 使能串口中断

TR1=1; // 启动定时器1

REN=1; // 允许接收

EA=1; // 使能全局中断

}

void send(unsigned char d) // 发送一个字节的数据

{

SBUF=d; // 将数据写入串口缓冲区

sending=1; // 设置发送标志

while(sending); // 等待发送完成

}

void uart(void) interrupt 4 // 串口中断服务程序

{

if(RI) // 接收中断

{

RI=0; // 清除接收中断标志

}

else // 发送中断

{

TI=0; // 清除发送中断标志

sending=0; // 清除发送标志

}

}

unsigned char code table[]=

{0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,

0x6f,0x77,0x7c,0x39,0x5e,0x79,0x71,0x00,0x40};

void Init(void); //初始化函数

void delay(unsigned char i)

{

uchar j,k;

for(j=i;j>0;j--)

for(k=125;k>0;k--);

}

void display(uchar a,uchar b,uchar c,uchar d,uchar e,uchar f)

{

dula=0;

P0=table[a];

dula=1;

dula=0;

wela=0;

P0=0xfe;

wela=1;

wela=0;

delay(5);

P0=table[b];

dula=1;

dula=0;

P0=0xfd;

wela=1;

wela=0;

delay(5);

P0=table[c];

dula=1;

dula=0;

P0=0xfb;

wela=1;

wela=0;

delay(5);

P0=table[d];

dula=1;

dula=0;

P0=0xf7;

wela=1;

wela=0;

delay(5);

P0=table[e];

dula=1;

dula=0;

P0=0xef;

wela=1;

wela=0;

delay(5);

P0=table[f];

dula=1;

dula=0;

P0=0xdf;

wela=1;

wela=0;

delay(5);

}

void keyscan()

{

{

P3=0xfe;

temp=P3;

temp=temp&0xf0;

if(temp!=0xf0)

{

delay(10);

if(temp!=0xf0)

{

temp=P3;

switch(temp)

{

case 0xee:

key=0;

wei++;

break;

case 0xde:

key=1;

wei++;

break;

case 0xbe:

key=2;

wei++;

break;

case 0x7e:

key=3;

wei++;

break;

}

while(temp!=0xf0)

{

temp=P3;

temp=temp&0xf0;

beep=0;

}

beep=1;

}

}

P3=0xfd;

temp=P3;

temp=temp&0xf0;

if(temp!=0xf0)

{

delay(10);

if(temp!=0xf0)

{

temp=P3;

switch(temp)

{

case 0xed:

key=4;

wei++;

break;

case 0xdd:

key=5;

wei++;

break;

case 0xbd:

key=6;

wei++;

break;

case 0x7d:

key=7;

wei++;

break;

}

while(temp!=0xf0)

{

temp=P3;

temp=temp&0xf0;

beep=0;

}

beep=1;

}

}

P3=0xfb;

temp=P3;

temp=temp&0xf0;

if(temp!=0xf0)

{

delay(10);

if(temp!=0xf0)

{

temp=P3;

switch(temp)

{

case 0xeb:

key=8;

wei++;

break;

case 0xdb:

key=9;

wei++;

break;

case 0xbb:

genggai=1;

wei=0;

break;

case 0x7b:

if(allow)

ok=1;

break;

}

while(temp!=0xf0)

{

temp=P3;

temp=temp&0xf0;

beep=0;

}

beep=1;

}

}

P3=0xf7;

temp=P3;

temp=temp&0xf0;

if(temp!=0xf0)

{

delay(10);

if(temp!=0xf0)

{

temp=P3;

switch(temp)

{

case 0xe7:

retry=1;

break;

case 0xd7:

close=1;

break;

case 0xb7:

confirm=1;

break;

}

while(temp!=0xf0)

{

temp=P3;

temp=temp&0xf0;

beep=0;

}

beep=1;

}

}

}

}

void shumima() //对按键采集来的数据进行分配

{

if(!wanbi)

{

switch(wei)

{

case 1:new1=key;

if(!allow) a=17;

else a=key; break;

case 2:new2=key;

if(a==17) b=17;

else b=key; break;

case 3:new3=key;

if(a==17) c=17;

else c=key; break;

case 4:new4=key;

if(a==17) d=17;

else d=key; break;

case 5:new5=key;

if(a==17) e=17;

else e=key; break;

case 6:new6=key;

if(a==17) f=17;

else f=key;

wanbi=1; break;

}

}

}

void yanzheng() //验证密码是否正确

{

if(wanbi) //只有当六位密码均输入完毕后方进行验证

{

if((new1==old1)&(new2==old2)&(new3==old3)&(new4==old4)&(new5==old5)&(new6==old6))

allow=1; //当输入的密码正确，会得到allowe置一

}

}

void main()

{

init();

allow=1;

P1=0x00;

for(ii=0;ii<=10;ii++)// 降低音量

{

index = 0; // 重置索引

while(index < 4) // 发送四个字节的数据

{

send(sound1[index]);

index++;

delay(10);

}

}

while(1)

{

keyscan();

shumima();

if(confirm) //验证

{

confirm=0;

ok=0; wei=0;

yanzheng();

if(allow) //验证完后，若allow为1，则开锁

{

P1=0x00;//

index = 0; // 重置索引

while(index < 6)

{

send(song1[index]); // 第1歌

index++;

}

if(!genggai)

{wanbi=0;}

Number\_of\_errors=0;

}

else if(!allow)

{

if(!genggai)

{wanbi=0;}

Number\_of\_errors++;

if(Number\_of\_errors==1)

{

index = 0; // 重置索引

while(index < 6)

{

send(song2[index]); // 第2歌

index++;

}

}

else if(Number\_of\_errors==2)

{

index = 0; // 重置索引

while(index < 6)

{

send(song3[index]); // 第3歌

index++;

}

}

else if(Number\_of\_errors>=3)//bo fang yin yue

{

index = 0; // 重置索引

while(index < 6)

{

send(song4[index]); // 第4歌

index++;

}

}

}

}

if(genggai) //当S16更改密码键被按下，genggai会被置一

{

if(allow) //若已经把锁打开，才有更改密码的权限

{

while(!wanbi) //当新的六位密码没有设定完，则一直在这里循环

{

keyscan();

shumima();

if(retry|close) //而当探测到重试键S18或者关闭密码锁键S19被按下时，则跳出

{ wanbi=1;

break;

}

display(a,b,c,d,e,f);

}

}

}

if(ok) //更改密码时，当所有六位新密码均被按下时，可以按下此键，结束密码更改

{ //其他时间按下此键无效

ok=0; wei=0;

genggai=0;

old1=new1;old2=new2;old3=new3; //此时，旧的密码将被代替

old4=new4;old5=new5;old6=new6;

a=16;b=16;c=16;d=16;e=16;f=16;

}

if(retry) //当重试按键S18被按下，retry会被置位

{

retry=0; wei=0;wanbi=0;

a=16;b=16;c=16;d=16;e=16;f=16;

new1=0;new2=0;new3=0;new4=0;new5=0;new6=0;

}

if(close) //当关闭密码锁按键被按下，close会被置位

{

close=0;genggai=0;//所有变量均被清零。

wei=0; wanbi=0;

allow=0;

P1=0xff;

a=16;b=16;c=16;d=16;e=16;f=16;

new1=0;new2=0;new3=0;new4=0;new5=0;new6=0;

}

display(a,b,c,d,e,f); //实时显示

}

}