河 南 理 工 大 学

毕业设计（论文）任务书

专业班级＿＿＿＿＿＿＿＿＿＿＿＿ 学生姓名＿＿＿＿＿＿＿＿＿＿＿＿

一、题目＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿

＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿

二、起止日期＿＿＿＿年＿＿＿＿月＿＿＿＿日至＿＿＿＿年＿＿＿＿月＿＿＿＿日

三、主要任务与要求

指 导 教 师＿＿＿＿＿＿ 职称＿＿＿＿＿＿

学 院 领 导＿＿＿＿＿＿签字（盖章）

年　　月　　日

河 南 理 工 大 学

毕业设计（论文）评阅人评语

题目_____

_____

评 阅 人_____ 职称_____

工作单位_____

年　　月　　日

河 南 理 工 大 学

毕业设计（论文）评定书

题目_____

_____

指 导 教 师_____ 职称_____

年　　月　　日

河 南 理 工 大 学

毕业设计（论文）答辩许可证

答辩前向毕业设计答辩委员会（小组）提交了如下资料：

1、设计（论文）说明　　共____页

2、图纸　　共____张

3、指导教师意见　　共____页

4、评阅人意见　　共____页

经审查，_____专业_____班_____同学所提交的毕业设计（论文），符合学校本科生毕业设计（论文）的相关规定，达到毕业设计（论文）任务书的要求，根据学校教学管理的有关规定，同意参加毕业设计（论文）答辩。

指导教师_____签字（盖章）

年　　月　　日

根据审查，准予参加答辩。

答辩委员会主席（组长）_____签字（盖章）

年　　月　　日

河 南 理 工 大 学

毕业设计（论文）答辩委员会（小组）决议

_____学院_____专业_____班_____同学的毕业设计（论文）于_____年_____月____日进行了答辩。

根据学生所提供的毕业设计（论文）材料、指导教师和评阅人意见以及在答辩过程中学生回答问题的情况，毕业设计（论文）答辩委员会（小组）做出如下决议。

一、毕业设计（论文）的总评语

二、毕业设计（论文）的总评成绩：_____

三、答辩组组长签名：

答辩组成员签名：

答辩委员会主席：_____签字（盖章）

年　　月　　日

摘要

人类在自己的历史上从未停止过对安全防护的升级改造。近些年，随着科学技术的进步，人们更多的将自动化和智能化的科技产品应用于安全防护。在这样的背景下，锁具作为居民日常生活不可缺少的安全用具，在人们的生活中扮演着越来越重要的角色。电子密码锁由于具有密码量大，使用方便等优点，因而应用十分广泛。本文在对国内外电子密码锁的发展现状进行研究的基础上，提出了一种适合传统家居的电子密码锁。采用单片机AT89S51 与低功耗CMOS型EEPROM AT24C02 作为主控器件与数据存储器单元，与LCD1602 相结合，实现电子密码锁的基本功能，提高了电子密码锁的安全性、可靠性、实用性，降低了硬件成本，使电子密码锁得到广泛应用。

本文根据系统的设计目标，给出了具体的硬件电路、软件结构以及详细的程序清单，并具体论述了各部分的设计要点，经实际测试，系统各项性能均已达到设计要求。文中设计的电路和控制方法适用于一般的单片机系统设计，硬件和软件也有一定的实用性和通用性。

关键词：AT89S51； AT24C02； 单片机； 密码锁

Abstract

People never stopped upgrading their safety in history. In modern times, with the progress of scientific and technological, people applied more automated and intelligent products to the living safety. In this context, locks, as indispensable equipment in residents' daily life, play a very important role in people's life. Electronic encrypted lock, whose advantage is that the amount of its passwords is enormous. It can be used conveniently.

This paper describes a control system of microcontroller AT89S51 used in encrypted lock. This include the keyboard input and LCD display circuit, memory circuit. As a result, the security, reliability, and the cost of this system is reduced greatly. This system can be applied in many regions. The experiment showed that the system can be used stably and reliably in encrypted lock. The control system is more simple, reliable and flexible.

In this article, according to the design goals of this system, the hardware circuit, and the software are given in this article and each part is also describe in detail. After the test, all the performance have met the designed for safe. Other occasions for high safety requirement can also use this system.

Key words: AT89S51; AT24C02; Microcontroller; Encrypted lock.

目　录

1 绪论

1.1 国内外现状

锁，是居民日常生活中的常用用具，在人们的生活中扮演着重要地角色，任何场所都需要锁具来提供最基本的安全。目前，普通家庭最常用的锁是机械锁，结构简单，使用方便，价格经济实惠。但带来好处的同时也依然暴露了很多缺点：一是机械锁需要依靠钥匙与锁芯的配合来完成工作。锁具庞大的数量导致钥匙不可避免的会发生相似的状况，安全性低。二是钥匙容易复制，不利于众多公共场合使用，比如宾馆等场合。由于人们对锁的安全性，方便性等性能有更多地要求，许多智能锁也相继出现，但这类锁是针对特定的指纹、智能卡，适用于个人或者安全性要求比较高的场合，其成本一般较高，在一定程度上限制了这类锁具的应用和推广。从密码锁的发展现状来看,机械密码锁通常包含复杂的机电功能组件，需借助先进的制造技术与装配工艺，制造周期长、成本高。卡片式密码锁是卡片设备，易磨损，寿命较短，卡片容易复制，不易双向控制。磁卡内存储的信息容易因外界磁场干扰而错乱，以致卡片被磁化无效。指纹密码锁从使用是否方便的角度看，对安装环境和使用者的要求很高。电子密码锁克服了机械式密码锁密码量少、安全性能差的缺点，尤其是的智能电子密码锁；不仅具有电子密码锁的功能，还可引入智能化管理功能，从而使密码锁具有更离的安全性和可靠性。

电子密码锁则具有安全性能高、成本低、功耗低、操作简单等优点。经调查，电子密码锁越来越多的被人们接受，现有国内市场上的电子密码锁有的是通过购买一些产品模块再开发，不具备自主知识产权；有的是自主研发的，但其功耗与成本都比较高，不具备广泛的应用价值。为了克服这些缺点，从经济实用角度出发，采用单片机AT89S51与低功耗CMOS型E^2PROM AT24CO2作为主控器件与数据存储器单元，设计一款可更改密码，具有报警、实时监控功能的电子密码锁。该电子密码锁体积小，易于开发、成本较低，安全性高，具有很高的实用性。

1.2 主要设计任务

设计需要完成的设计任务有：

首先完成硬件电路的设计，包括时钟电路、复位电路、键盘电路、开锁电路、报警电路和存储电路。注意可靠性与抗干扰设计，电路应简洁完善。

完成程序的编写和编译，使系统具备如下功能：开锁，密码输入和设置，密码校验，密码输入错误和报警，锁定键盘，LCD 显示状态等，程序需简洁，模块化，思路应清晰。

完成电路的模拟仿真。

完成实物的焊接。

更重要的是通过这次毕业设计，使我们可以掌握并利用单片机的软件编程技术，能够选择合适的算法，运用所学的电路知识完成基本的电路设计，利用Altium Designer软件绘制电路图，并制作电子密码锁的实物模型并实现所要求的的基本功能，完成整套电子密码锁的设计,从而熟悉一个产品完整的开发流程,增强实践应用能力。

2 电子密码锁的总体方案设计

2.1 电子密码锁系统简介

本系统可以使用两种方案：一是采用数字电路控制，采用双 JK 触发器构成的数字电路作为密码锁的核心控制。采用数字电路方案设计的好处在于设计简单，但控制的准确性和灵活性差。二是采用 AT89S51 为核心的单片机控制方案。选用单片机作为本设计的核心元件，利用单片机丰富的 I/O 口和编程的便捷，以及控制的准确性，实现电子密码锁功能。

目前的电子密码锁普遍使用专用芯片，由于专用芯片的功能固定，而且价格昂贵，几乎不能升级扩展，使用限制很大。为了使系统具备升级扩展能力，本系统选用单片机作为中央处理器，配置适当的外围电路，实现系统的控制功能。

本系统主要由开锁模块、报警模块、外部存储模块和单片机最小系统组成。其主要设计思路为:电子密码锁的初始密码为 123456,用户打开密码锁后,可以进行密码设置，经过两次密码输入，确定新密码。开锁时，用户输入密码，与密码进行比对，密码正确则密码锁打开；密码错误则报警，并累计错误次数，超过 3 次则锁定键盘等。

2.2 电子密码锁系统设计目标和方案

2.2.1 设计目标

电子密码锁系统的设计目标：

1.设计出的产品性能必须可靠、稳定、经济。

2.所设计的产品需要的设计环境相对较低，易容易实现。

3.用户可在密码锁打开的情况下设定密码，密码为 6 位数。

4.在掉电的情况下，原密码保持不变。

5.密码输入错误报警，错误次数累计，超过三次，键盘锁定。

6.应当配置合适的显示器件，具有声音提示与报警功能。

7.核心为单片机控制，直接利用单片机内部资源，系统设计合理规范。

2.2.2 总体方案确定

组成本系统的主要模块为：

1.电源模块。

2.开锁模块。

3.报警模块。

4.显示模块。

5.键盘输入模块。

6.外部存储模块。

7.单片机最小系统。

系统的总体框图如图 2-1 所示：

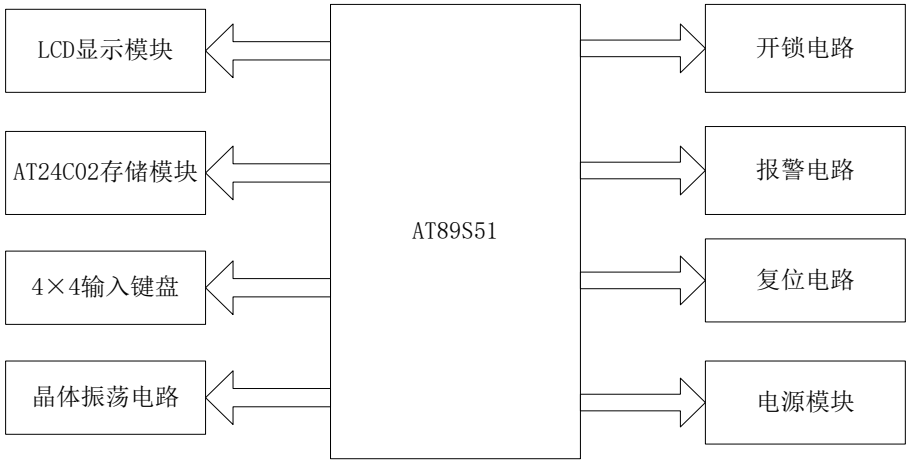


图 2-1 密码锁系统总体框图

由于系统规模不大，电路设计采用了Atmel公司的AT89S51 单片机作为系统的控制器。从成本考虑，采用了一个 4×4 的非编码键盘作为密码的输入电路，键盘的键值扫描口为P1，利用软件来实现键盘的消抖处理。由于功能上要实现密码输入提示，开锁情况显示等功能，本电路采用了SMC1602 液晶显示模块作为显示电路，具有显示信息丰富，功耗低等优点。电路中选用了一块AT24C02 芯片作为密码和开锁信息的存储器。在电路上选用单片机的P2.0 和P2.1 分别与AT24C02 的SDA、SCL相连。开锁电路采用二级三极管电路驱动电磁锁来实现；报警电路采用三级管驱动一个蜂鸣器来实现。

即功能上，其中矩阵键盘电路用于密码的输入和各种功能的实现。LCD显示模块则是显示密码的输入及密码锁的开关状态，以及显示菜单的界面。密码输入正确或者错误决定着单片机控制开锁电路还是报警电路。电源电路提供电压电流保证，从而使系统顺利工作。

根据功能实现要求输入密码错误需要锁定键盘，用程序锁定.根据系统的功能实现要求，采用模块化程序设计。主程序主要实现LCD模块的初始化和存储单元的基本分配，和各个子程序的调用。子程序分为菜单程序、AT24C02 读写子程序、LCD1602 显示程序和键盘扫描程序等。按键共有 16 个，分别代表数字 0～10、确定、上移、下移、删除、退出、密码恢复按键。

3 电子密码锁的硬件结构设计

3.1 硬件系统设计原则

对于系统单片机的设计，一般需要从以下两个方面考虑：

一是单片机与外围设备的连接，即系统配置。选择合适的外围设备，通过接口电路的设计，与单片机电路相连接，交流数据处理信息。在本系统中，需要与单片机进行连接的设备有键盘、LCD液晶显示屏、EEPROM存储器等。

二是系统的扩展设计。假如单片机内部的系统单元无法满足设备所需的功能，可通过外接扩展设备来增加功能模块，主要就是解决单片机与外接扩展设备之间的接口电路的设计。在本系统中，所用单片机完全能够满足设计需要，因此未使用扩展电路。

系统单片机的扩展和配置设计应遵循以下原则：

1.在接口电路的设计上，应选择典型电路，符合单片机的常规接法，注意规范和实用，保证电路可行耐用的基础上，让设计更加简洁，完善。

2.硬件结构应结合软件程序方案一并考虑硬件与软件方案会相互影响。原则是尽量由软件实现软件可以实现的功能，硬件结构应尽可能简化。但选用软件方案时应考虑这些因素，由软件实现要比由硬件实现占用CPU时间，实现功能的时间要长。因此要合理设计才能使系统达到最佳状态。

3.硬件系统设计时，可靠性及抗干扰设计是必须的。包括器件、芯片的选择，滤波，布线等。

4.单片机外接电路较多时，应考虑其驱动能力，驱动能力不足时，系统工作不可靠，解决的办法是增加驱动能力。增设线驱动器或者减少芯片功耗，降低总线负载。

3.2 硬件总体设计

本论文设计的基于单片机的电子密码锁是以AT89S51 作为主控芯片。从成本考虑，采用了一个 4×4 的非编码键盘作为密码的输入电路，键盘的键值扫描口为P1，利用软件来实现键盘的消抖处理。由于功能上要实现密码输入提示，开锁情况显示等功能，本电路采用了SMC1602 液晶显示模块作为显示电路，具有显示信息丰富，功耗低等优点。电路中选用了一块AT24C02 芯片作为密码和开锁信息的存储器。在电路上选用单片机的P2.0 和P2.1 分别与AT24C02 的SDA、SCL相连。开锁电路采用二级三极管电路驱动继电器来实现；报警电路采用三级管驱动一个蜂鸣器来实现。

3.3 CPU及外围电路

3.3.1 CPU的选择

CPU是系统的核心单元，在执行程序中其关键作用，它的优劣直接关系到系统的性能。51 系列单片机，单片机结构相对简单，性能好，控制功能强，价格低廉，应用灵活。

本系统选用Atmel公司生产的 8 位 89S51 单片机作为整个系统控制中心。AT89S51 是美国Atmel公司生产的低功耗，高性能CMOS8 位单片机，片内含 4K的可编程的Flash只读程序存储器，器件采用Atmel公司的高密度、非易失性存储技术生产，兼容标准 8051 指令系统及引脚。它集Flash程序存储器（既可在线编程也可用于传统方法进行编程）及通用 8 位微处理器于单片机芯片中，可灵活应用于各种控制领域。

其主要性能参数：

1.与MCS51 产品指令系统完全兼容

2.4K字节在线系统编程（ISP）Flash闪速存储器

3.1000 次擦写周期

4.4.0～5.5V工作电压范围

5.全静态工作模式：0Hz～33MHz

6.三级程序加密锁

7.一个片内振荡频率为 1.2～12MHz的振荡器及时钟电路

8.128B的片内 RAM

9.32 个可编程I/O口线

10.2 个 16 位定时/计数器

11.6 个中断源

12.全双工串行UART通道

13.低功耗空闲和掉电模式

AT89S51 采用 40 引脚双列直插封装形式，内部由CPU，4KB的片内程序ROM，256B的RAM，2 个 16B的定时/计数器T0、T1，4 个 8B的I/O端口：P0、P1、P2、P3，一个全双工串行通信口组成。P0、P1、P2、P3 这些端口都是双向的，每个端口位均包含：两个三态输入缓冲器、一个输出锁存器及一个场效应管驱动器，其中P0 端口还有一个上拉场效应管。

由于它们都属于地址号可被 8 整除的特殊功能寄存器，故可通过位寻址或直接寻址方式对其进行按位或字节型的I/O操作。

其引脚图如图 3-1 所示：

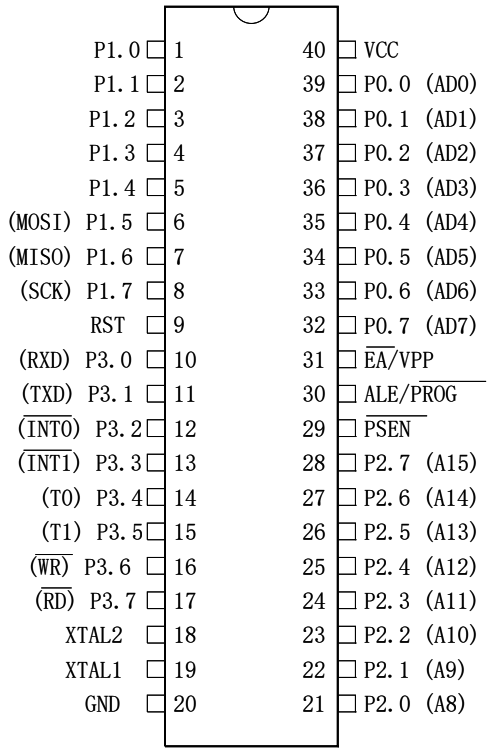


图 3-1 AT89S51 引脚图

VCC：电源电压。

GND：接地。

P0：P0 口是一组 8 位漏极开路型双向I/O口，也就是地址/数据总线复用口。作为输出口时，每位能驱动 8 个TTL逻辑门电路，对端口写“1”可作为高阻抗输入端使用。在访问外部数据存储器或程序存储器时，这组口线分时转换地址和数据总线复用，在访问期间激活内部上拉电阻。在Flash编程时，P0 口接受指令字节，在程序校验时输出指令字节，此时需要接上拉电阻。

P1：P1 口是一个带有内部上拉电阻的 8 位双向I/O口，P1 的输出缓冲级可驱动 4 个TTL逻辑门电路。对端口写“1”，通过内部的上拉电阻把端口拉到高电平，此时可作为输入口使用。Flash编程和校验期间，P1 接收低 8 位地址。

P2：P2 口是一个带有内部上拉电阻的 8 位双向I/O口，P2 的输出缓冲级可驱动 4 个TTL逻辑门电路。对端口写“1”，通过内部的上拉电阻把端口拉到高电平，此时可做输出口做输入口使用时，因为内部存在上拉电阻，某个引脚被外部信号拉低时会输出一个电流。

在访问外部程序存储器或16位地址的外部数据存储器时,P2送出高8位地址数据。

在访问 8 位地址的外部数据存储器时，P2 口线上的内容，在整个访问期间不改变。

Flash编程或校验时，P2 也接收高位地址和其他控制信号。

P3：P3 口是一组带有内部上拉电阻的 8 位双向I/O口.P3 口输出缓冲级可驱动 4 个TTL逻辑门电路。对P3 口写入“1”时，它们被内部上拉电阻拉高并可作为输出端口。作输入端时，被外部拉低的P3 口将用上拉电阻输出电流。P3 口除作为I/O口线，还具有第二功能。

第二功能如表 3-1 所示：

表 3- 1 P3 口第二功能表

引脚	第二功能
P3.0	RXT串行口接收端
P3.1	TXD串行口发送端
P3.2	INT0 外部中断 0 请求线
P3.3	INT1 外部中断 1 请求线
P3.4	T0 定时器/计数器 0 输入线
P3.5	T1 定时器/计数器 1 输入线
P3.6	WR写外部数据存储器控制信号
P3.7	RD读外部数据存储器控制信号

RST/V_{PD}：该引脚为双引脚功能。

复位信号输入端，高电平有效。当单片机工作时，RST引脚出现两个机器周期以上的高电平使单片机复位。

VCC掉电后，此引脚可接备用电源，低功耗条件下保持内部RAM中的数据不丢失。

$\overline{ALE}/\overline{PROG}$：该引脚为双引脚功能。

地址锁存允许。在系统扩展时，该信号的下跳沿将由P0 口发出的低 8 位地址信号进行锁存，并保证此时锁存信息是稳定的地址信息。在不访问片外存储器时，ALE引脚上也输出频率为时钟振荡频率的 1/6 的周期性信号。

对有片内EPROM的单片机进行编程时，脉冲信号由该引脚引入。

PSEN：片外程序存储器读选通信号。

取指令操作期间，PSEN的操作频率为振荡频率的 1/6；以通过P0 口读入指令，在访问外部数据存储器时该信号无效。

$\overline{EA}$/VPP：该引脚为双引脚功能。

一为片外程序存储器选择信号，当EA=0，选择片外程序存储器。对无片内程序存储器的单片机此引脚必须接地。EA=1 时，单片机访问片内程序存储器。

在对 8071 单片机片内EEPROM编程期间，此引脚引入+21V编程电源VPP。

XTAL1、XTAL2：接外部晶体的一个引脚，需要采用外部时钟信号时，注意CHMOS单片机与HMOS单片机中的XTAL1、XTAL2 引脚接法有所不同。

3.3.2 晶振电路

单片机的时钟信号用来提供单片机内各种微操作的时间基准。时钟电路用于产生单片机工作所需的时钟信号，时序是指令执行中各信号之间的相互关系。单片机本身就如同一个复杂的同步时序电路，为了保证同步工作方式的实现，电路应在唯一的时钟信号控制下严格的按时序进行工作。

51 系列单片机的时钟信号通常有两种产生方式：内部时钟方式和外部时钟振荡方式。在引脚XTAL1 和XTAL2 外接晶体振荡器，就构成了自激振荡器，并产生震荡时钟脉冲。晶振选择 6MHz、12MHz或 24MHz。电容器C1、C2 起稳定振荡频率、快速起振的作用。电容值为 5～30pF。外部振荡方式是把已有的时钟信号引入单片机内，该方式适用于多片单片机同时工作，以使各单片机的时钟同步。内部振荡的方式所得的信号比较稳定，因此本系统中采用内部振荡方式。如图 3-2 所示：

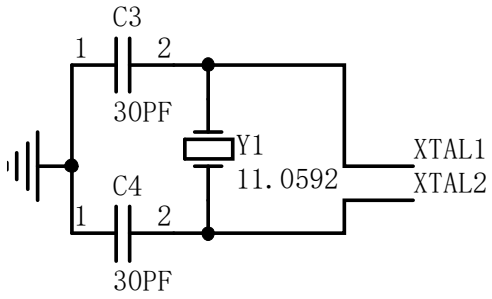


图 3-2 时钟电路

3.3.3 复位电路

复位电路是单片机的初始化操作，其主要功能是使单片机从 0000H单元开始执行程序。除了进入系统的正常初始化以外，当由于程序运行出错或操作错误使系统处于死锁状态时，为摆脱困境也需按复位键以重新启动。51 系列单片机的RST引脚为复位引脚，只要在RST引脚上引入一个至少保持两个机器周期的高电平，单片机就完成一次复位。如果RST持续为高电平，单片机就处于循环复位状态，而无法执行程序，因此要求单片机复位后能脱离复位状态。本系统采用开关复位方式。开关复位同样具有上电复位的功能。上电后，由于电容充电，使RST持续一段高电平时间。当单片机已在运行之中时，按下复位键也能使RST持续一段时间的高电平，从而实现上电且开关复位的操作。

电路图如图 3-3 所示：

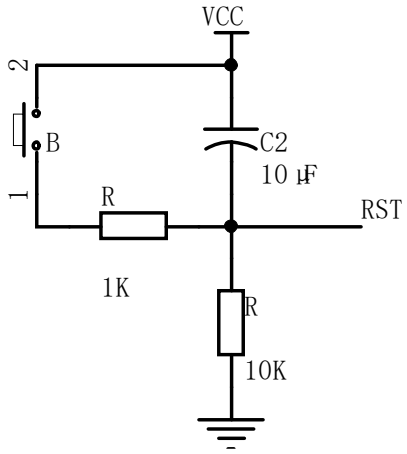


图 3-3 复位电路

3.4 外部存储电路

由于单片机内部存储的容量有限，又因为本设计所要存储的数据大于单片机内部的存储容量，因此需要对外扩充一个存储器。外部存储器的作用是当系统突然掉电时用户先前存储的数据不会丢失，一旦再次通电，可继续进行操作。在本系统中，我们采用的是EEPROM存储器，即电擦除可编程只读存储器，型号为AT24C02。EEPROM是近年来被广泛使用的一种只读存储器，它能在系统中进行在线改写，并能在掉电的情况下保存数据，符合系统设计的要求。本设计中，用户设定的密码存储于EEPROM AT24C02 中，AT24C02 与单片机之间采用I^2C总线通讯方式。

I^2C是Philips公司提出的串行通信接口规范，使用I^2C总线时需要注意以下基本概念：

1.发送器：发送数据到I^2C总线的器件。

2.接收器：从I^2C总线接收数据的器件。

3.主机：初始化发送、产生时钟信号及负责终止发送的器件。本设计即单片机为主机。

4.从机：具有I^2C总线唯一地址，可被主机寻址的器件，本设计从机为EEPROM器件。

I^2C总线主要用于IC器件之间的二线制同步通信，他通过两根线（串行时钟线SCL和串行数据线SDA）便能实现总线上各器件的同步数据传送。I^2C总线可以极为方便的构成多机系统和外围器件扩展系统。总线用软件寻址来识别每个器件。其典型的系统接线图如图 3-4 所示。

由图可知，I^2C总线是一个多主机总线，即总线上可以有一个或多个主机，总线运行由主机控制。I^2C总线接口为开漏或开集电极输出，需要加上拉电阻。连接到I^2C总线上的所有单片机、外围器件都有一个唯一的地址。利用I^2C总线进行传输的过程中，所有状态

都将生成相对应的状态码，系统中的主机则根据这些状态码自动地进行总线管理。用户只需要在程序中装入标准处理模块，根据数据操作要求完成I²C总线的初始化，启动I²C总线就能自动完成规定的数据传送操作。

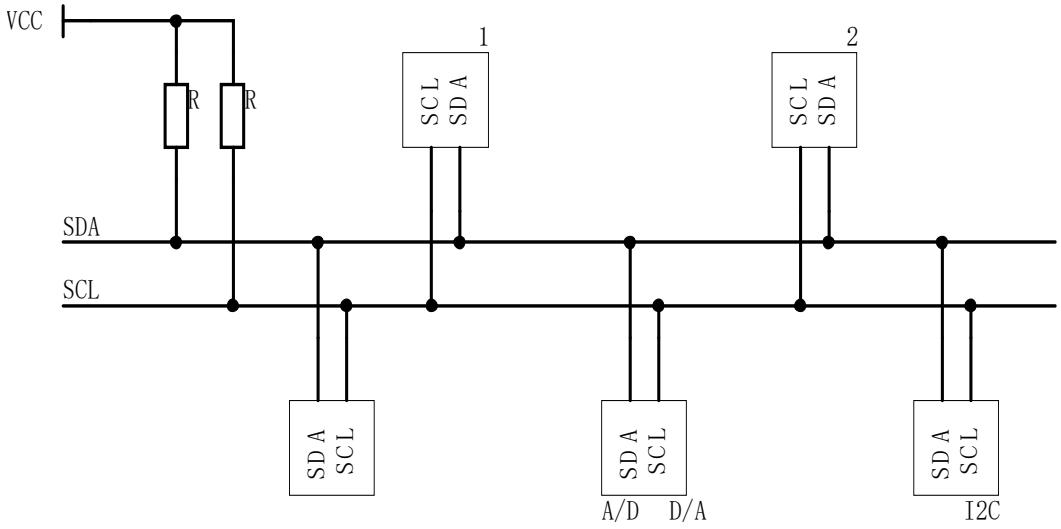


图 3-4 I²C 系统接线图

I²C总线数据传送的相关规定及技术要点如下所述。

1.数据位有效性规定

进行I²C总线数据传送时，SDA线上数据在SCL线为高电平期间必须保持稳定，只有在SCL时钟信号为低电平期间，数据线SDA上的高电平或低电平状态才允许变化。

2.起始和终止信号

I²C总线的起始信号和终止信号定义如下：

SCL线为高电平期间，SDA线由高电平向低电平跳变表示起始信号。

SCL线为高电平期间，SDA线由低电平向高电平跳变表示终止信号。

起始和终止信号定义中，SDA的变化均在SCL为高电平期间发生，这样的变化被I²C总线认为是非有效数据位，是启/停控制信号位。即SCL时钟信号为低电平期间才允许SDA线数据状态变化。

接收器接收到一字节数据后，如果要完成一些其他工作，如处理内部中断服务等，可能无法立即接收下一字节，此时接收器件可将SCL线拉至低电平，从而使发送器处于等待状态，接收器准备好接受下一字节时，再释放SCL线使之为高电平，继续数据接收。

3.字节传送与应答

启动I²C总线后，所传输的每字节必须均为 8 位长度，高位优先传送，每字节后面跟随一位“应答”位，故一帧数据共有 9 位。

如果从机由于某种原因不能应答主机，它必须将SDA线置于高电平。

此时主机读取到从机的非应答信号时可产生一个终止信号，结束总线数据传送。主机接收从机数据的最后一个字节之前的每一字节均需要向从机发送应答，当主机接收到从机数据的最后一个字节时，它要向从机发出一个非应答信号以便从机结束传送，从机随后释放SDA线，以便允许主机产生终止信号。

如图 3-5、3-6、3-7 给出了I^2C总线协议信号，包括起始信号、终止信号、应答、非应答信号及部分相关时序。

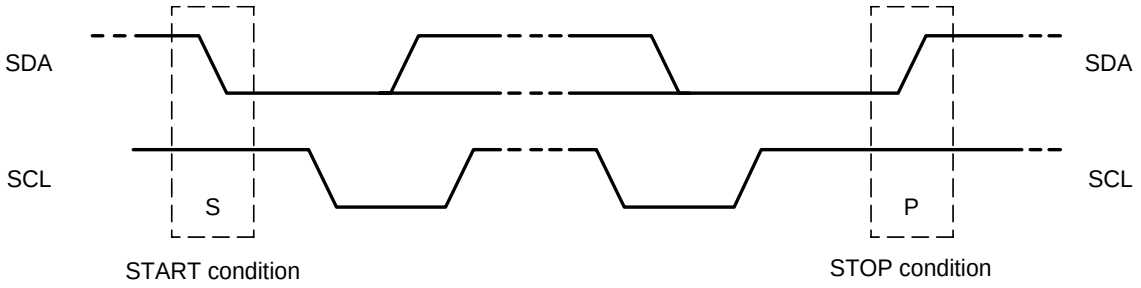


图 3-5 起始和停止条件

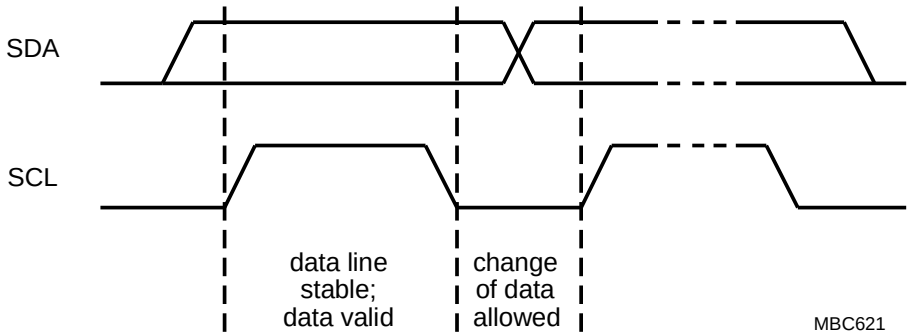


图 3-6 I^2C 总线的位传输

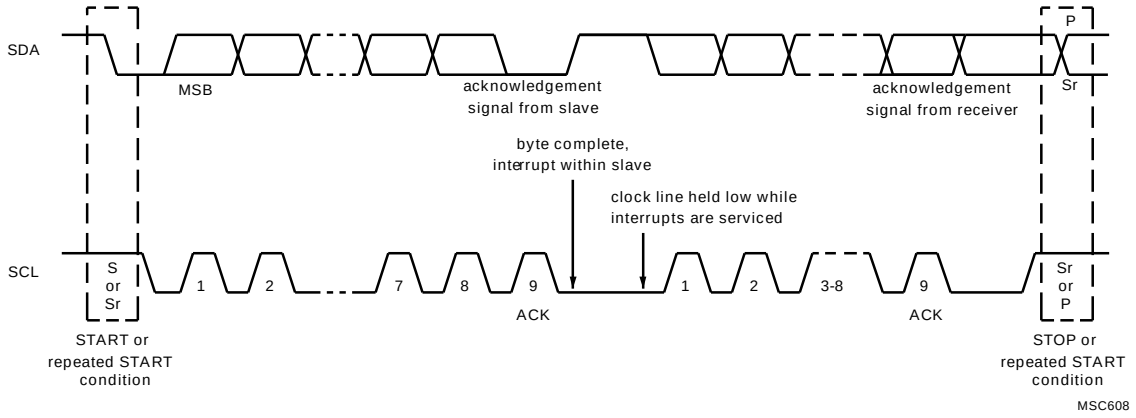


图 3-7 I^2C 总线的数据传输

信号传输的基本流程是：

1. 主控器首先发出系统起始信号，然后发出所需被控器的地址及 1 字节的读写位。

2. 被控器接收到信号后，反馈应答信号。

3. 主控器收到应答信号后，或向被控器发送数据，或从被控器接收数据。

4.主控器最后发出停止信号。

带I²C总线接口的EEPROM是单片机应用系统中应用较为广泛的一类I²C存储器器件。其优点是体积小、功耗低、占用I/O口线少，性价比高。AT24C系列的EEPROM，具有单电源供电，工作电压范围宽（1.8～5.5V）；低功耗CMOS技术（100kHz（2.5V）和 400kHz（5V）兼容），自定时写周期（包含自动擦除）、硬件写保护等特点。AT24C02 存储容量为 256B。存储器硬件电路连接如图 3-8 所示：

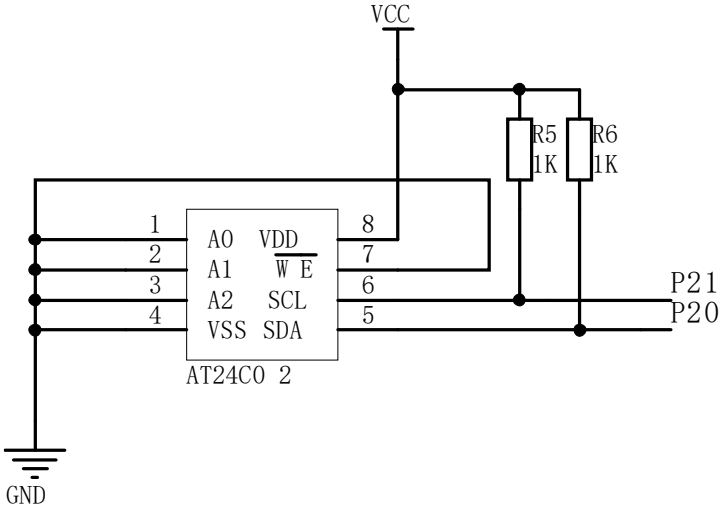


图 3-8 AT24C02 与单片机的连接图

图中，AT24C02 的A0、A1、A2 脚是三条地址线，用于确定芯片的硬件地址。在电路图连接中都接地。VCC接+5V电源，VSS接地。SDA为串行数据输入/输出，数据通过这条双向I²C总线串行传送，在电路图连接中和单片机的P2.0 连接。SCL为串行时钟输入线，在电路图连接中和单片机的P2.1 连接。

8051 单片机读/写I²C接口AT24C02 时，需要用软件模拟I²C串行时钟信号和操作时序，I²C的启动、停止、读写等操作均由“IIC总线通用宏及函数”提供，解读这些模块代码时可对照I²C的各项操作时序图。如图 3-9～3-13 是AT24CX系列的操作时序图。在对AT24CX进行读写程序编程时需要对照时序图进行编程。分别为字节写、多字节及页写、从当前地址读字节、从任意指定地址读字节、顺序读取。

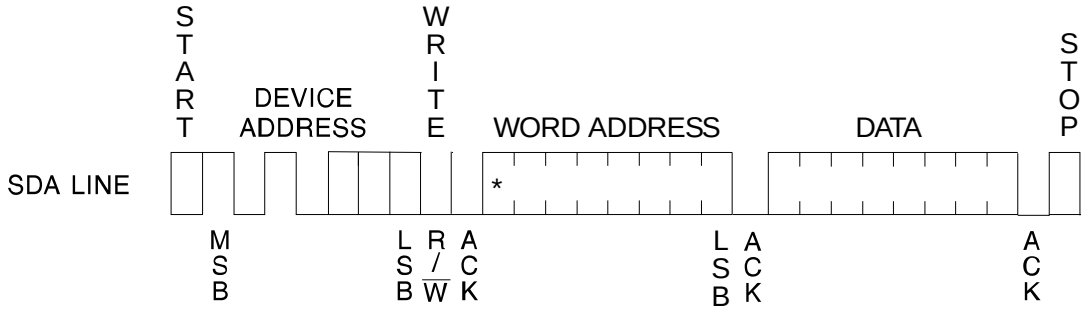


图 3-9 字节写

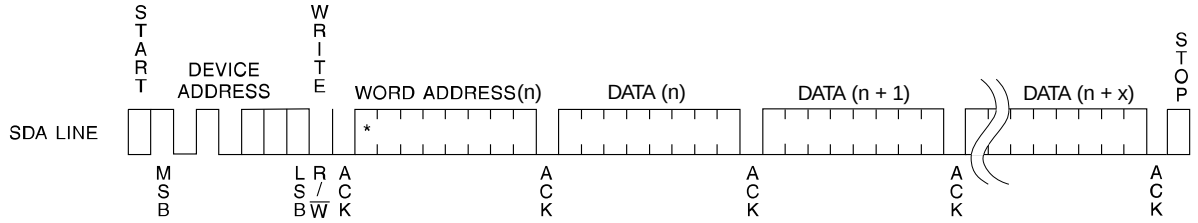


图 3-10 多字节及页写

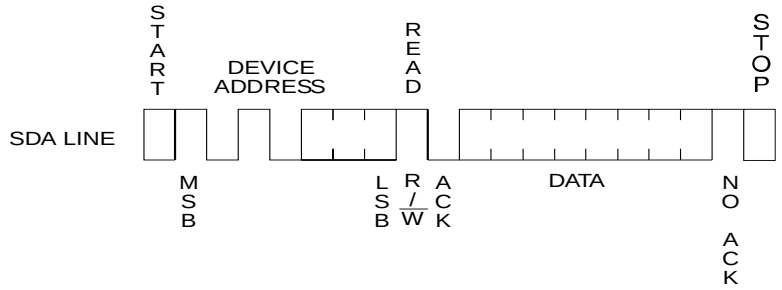


图 3-11 从当前地址读字节

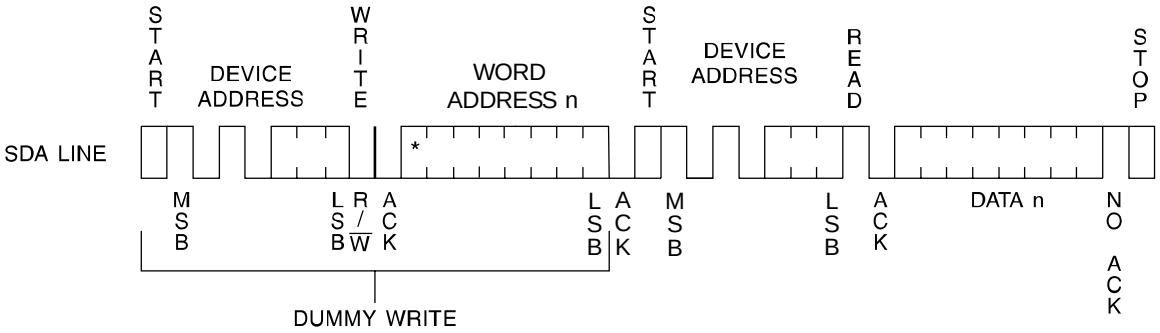


图 3-12 从任意指定地址读字节

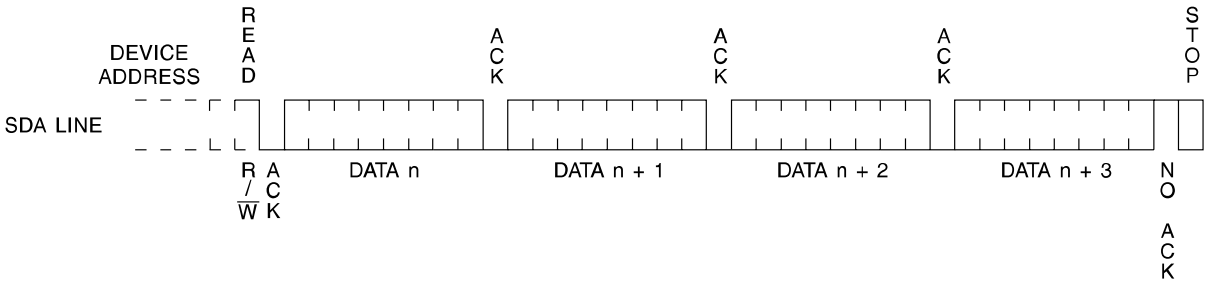


图 3-13 顺序读取

编写程序可以以此为准。

3.5 电源电路

如图 3-14，该电路是经过变压器到 9V 的交流电，再由桥式整流，变为直流，通过稳压管 7805 稳压得到 5V 电压 VCC。图中桥式整流电路将交流电转变为直流电,电容起滤波的作用。

本系统中也可直接与有 5V电压的电源口相连，如图 3-15。

本系统由于条件限制,采用了 USB 口直接与 5V 电源相连。

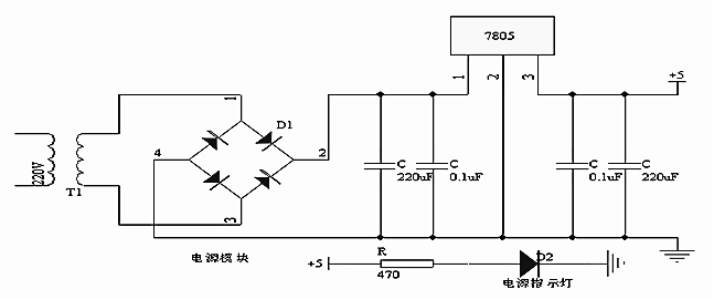


图 3-14 电源电路 a

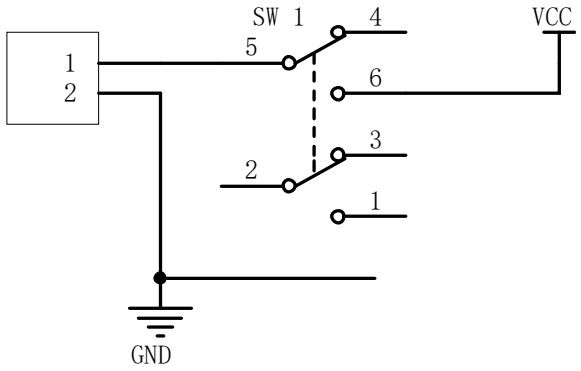


图 3-15 电源电路 b

3.6 开锁电路

电子锁电路中最重要的部分就是开锁机构电路，通过单片机送给开锁执行机构电路，电路驱动电磁锁吸合，从而达到开锁的目的。当用户密码输入正确，单片机便输出开门信号，送到开锁驱动电路，然后驱动电磁锁，达到开门的目的。其实际电路图如图 3-16 所示。

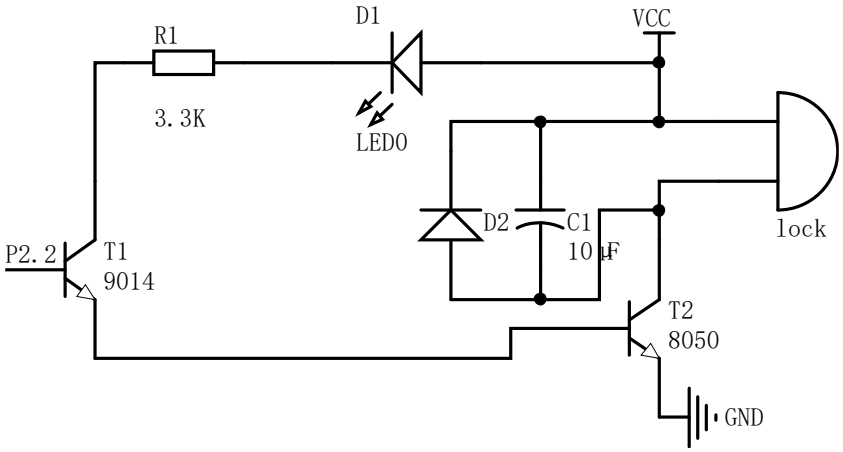


图 3-16 开锁电路

开锁机构由驱动电路和执行开锁电路两级组成。由D1、R1、T1 组成驱动电路，T1 可以选择普通小功率三极管如 9014、9018 可以满足要求，D1 作为开锁的提示。由D2、C1、

T2 组成执行开锁电路，其中D2、C1 是为了消除电磁锁可能产生的反向高电压以及可能产生的电磁干扰从而起到保护电路的作用。T2 可选用中功率的三极管如 8050，电磁锁的选用要视情况而定，但是吸合力要足够且有一定的余量。当单片机输入开门信号时驱动电路T1 导通从而D1 发光提示开锁，同时驱动T2，T2 导通执行开锁。

3.7 报警电路

报警电路采用三极管驱动一个蜂鸣器来实现，报警驱动电路的信号由P3.3 输出。报警电路由蜂鸣器及外围电路组成，加电后不发声，当密码输入错误发出报警声。与二级管相连的33KΩ电阻起保护作用，防止电流过大烧毁蜂鸣器。三极管T3 相当于一个开关，单片机提供高电平，三极管导通，蜂鸣器发声。 电路如图 3-17 所示：

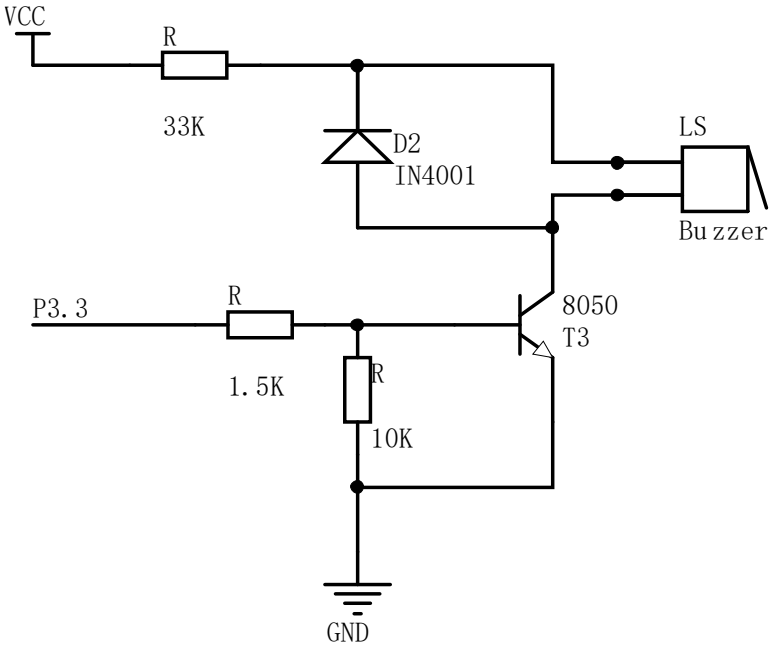


图 3-17 报警电路

3.8 键盘电路

键盘是用户与单片机交互的重要手段，用于向CPU输入运行参数和控制系统的运行状态。键盘电路形式分为直接编码输入键盘和矩阵键盘，前者接口电路简单，一般应用于需要少量按键的控制系统，后者因占用IO引脚数少，常被按键较多的控制系统采用。

本系统所有的密码设置及输入都需要由键盘输入，且由于按键较多，所以采用 4×4 矩阵键盘形式。如图 3-18 所示：

矩阵键盘中，行线、列线分别连接到按键开关的两侧。将列线赋值高电平，或通过上拉电阻接到+5V上。无按键按下时，列线处于高电平状态；当有按键按下时，行、列线

导通，列线电平将由与它相连的行电平决定，将行线、列线信号配合起来做适当处理，可唯一确定闭合键所在的位置。由于要实现的功能众多，输入的六位密码也需要数字键输入，因此无法设定单独的功能按键，只能设置确定、退出、上移、下移等按键，以及数字按键。程序设置菜单，选择相应的功能。

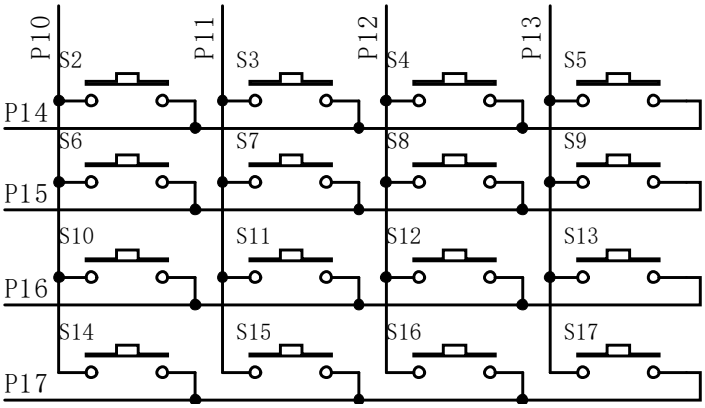


图 3-18 矩阵键盘电路

3.9 显示电路

在日常生活中，我们对液晶显示器并不陌生。液晶显示模块已作为很多电子产品的通过器件，如在计算器、万用表、电子表及很多家用电子产品中都可以看到，显示的主要是数字、专用符号和图形。在单片机的人机交流界面中，一般的输出方式有以下几种：发光管、LED数码管、液晶显示器。发光管和LED数码管比较常用，软硬件都比较简单，本系统采用的是LCD1602 作为输出显示器件。在单片机系统中应用液晶显示器作为输出器件有以下几个优点：

1.显示质量高。由于液晶显示器每个点在收到信号后会一直保持色彩和亮度不变，恒定发光，不像阴极射线管显示器（CRT）那样需要不断刷新亮点。因此，液晶显示器画质高而且不会闪烁。

2.数字式接口。液晶显示器都是数字式的，和单片机系统的接口更加简单可靠，操作更加方便。

3.体积小、重量轻。液晶显示器通过显示屏上的电极控制液晶分子状态来达到显示目的，在重量上比相同显示面积的传统显示器要轻得多。

4.功耗低。相对而言，液晶显示器的功耗主要消耗在其内部的电极和驱动IC上，因而耗电量比其它显示器要少得多。

显然液晶显示模块体积小、功耗低、显示内容丰富、超薄轻巧等优点，使其在低功耗应用系统中得到广泛应用。目前字符型液晶显示模块已是单片机应用设计中最常用的

信息显示器件。LCD1602 液晶是一款很常用的字符液晶。可显示 2 行，每行 16 个字符。采用单+5V电源供电，外围电路简单，价格便宜，具有很高的性价比。

LCD1602 有个明显的缺点就是显示内容虽然已经很丰富了,但然存在局限.字库里默认只有数字、字母等,中文汉字虽然可以显示，但非常麻烦，需要取模等，并且无法显示复杂的汉字。本设计LCD1602 完全可以满足功能。显示的信息以英文显示,如输入密码界面显示“PASSWORD INPUT”。

尺寸如图 3-19 所示：

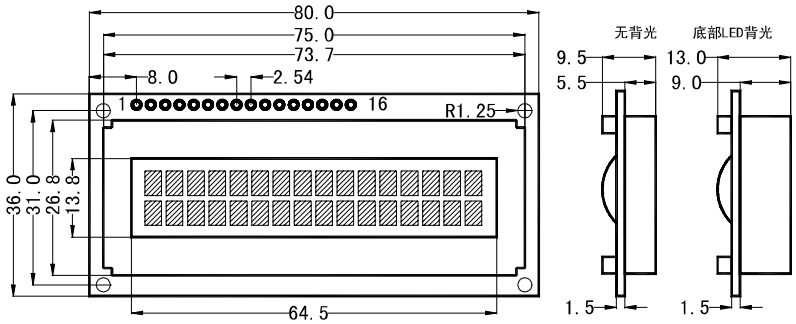


图 3-19 LCD1602 外形尺寸

主要技术参数如表 3-2 所示：

表 3-2 LCD1602 主要技术参数

显示容量	16×2 个字符
芯片工作电压	4.5～5.5V
工作电流	2.0mA（5.0V）
模块最佳工作电压	5.0V
字符尺寸	2.95×4.35mm

LCD1602 采用标准 14 脚或 16 脚接口，各引脚功能如表 3-3 所示。

表 3-3 LCD1602 引脚及功能

编号	符号	引脚说明	编号	符号	引脚说明
1	VSS	电源地	9	D2	DATA I/O
2	VDD	电源正极	10	D3	DATA I/O
3	VL	液晶显示偏压信号	11	D4	DATA I/O
4	RS	数据/命令选择端	12	D5	DATA I/O
5	R/W	读/写选择端	13	D6	DATA I/O
6	E	使能信号	14	D7	DATA I/O
7	D0	DATA I/O	15	BLA	背光源正极
8	D1	DATA I/O	16	BLK	背光源负极

VL为液晶显示器对比度调整端，接正电源时对比度最弱，接地电源时对比度最高。若对比度过高会产生鬼影，使用时可以通过一只 10KΩ电阻来调整对比度。RS为寄存器选择端，RS为高电平时选择数据寄存器，为低电平时选择指令寄存器。RW为读写信号线，为高电平时进行读操作，为低电平时进行写操作。当RS和RW同为低电平时可以显示指令或者显示地址。当RS为低电平、RW为高电平时可以读忙信号，当RS为高电平、RW为低电平时可以写入数据。E为使能端，当E由高电平跳变到低电平时，液晶模块执行命令。D0～D8 为位双向数据线。

控制器接口说明：

1.基本操作时序。

读状态：输入：RS=L，RW=H，E=H　　输出：D0～D7=状态字

写指令：输入：RS=RW=L，D0～D7=指令码，E=高脉冲　　输出：无

读数据：输入：RS=H，RW=H，E=H　　输出：D0～D7=数据

写数据：输入：RS=H，RW=L，D0～D7=数据，E=高脉冲　　输出：无

2.状态字说明。

表 3-3 状态字说明

STA7	STA6	STA5	STA4	STA3	STA2	STA1	STA0
D7	D6	D5	D4	D3	D2	D1	D0

STA0～STA6	当前数据地址指针的数值	
STA7	读写操作使能	1：禁止　0：允许

注：对控制器每次进行读写操作之前，都必须进行读写检测，确保STA7 为 0。

3.RAM地址映射图。

控制器内部带有 80×8 位（80 字节）的RAM缓冲区，对应关系如图 3-20 所示。

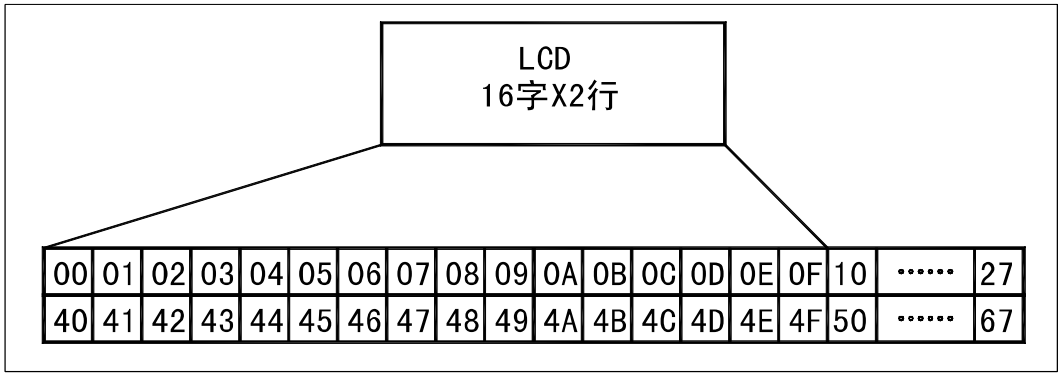


图 3-20 RAM 地址映射图

4.指令说明。

显示模式设置：

表 3-4 显示模式设置

指令码								功能
0	0	1	1	1	0	0	0	设置 16×2 显示，5×7 点阵，8 位数据接口

显示开/关及光标设置：

表 3-5 显示开/关及光标设置

指令码								功能
0	0	0	0	1	D	C	B	D=1 开显示； D=0 关显示 C=1 显示光标；C=0 不显示光标 B=1 光标闪烁；B=0 光标不显示
0	0	0	0	0	1	N	S	N=1 当读或写一个字符后地址指针加 1，且光标加 1。 N=0 当读或写一个字符后地址指针减 1，且光标减 1。 S=1 当写一个字符，整屏显示左移（N=1）或右移（N=0），以得到光标不移动而屏幕移动的效果。 S=0 当写一个字符，整屏显示不移动

数据控制：

控制器内部设有一个数据地址指针，用户可以通过它们来访问内部的全部 80 字节 RAM。

数据指针设置：

表 3-6 数据指针设置

指令码	功能
80H+地址码（0～27H，40～67H）	设置数据地址指针

其他设置：

表 3-7 其他设置

指令码	功能
01H	显示清屏：1.数据指针清零 2.所有显示清零
02H	显示回车：数据指针清零

控制器接口时序说明：

1. 读操作时序

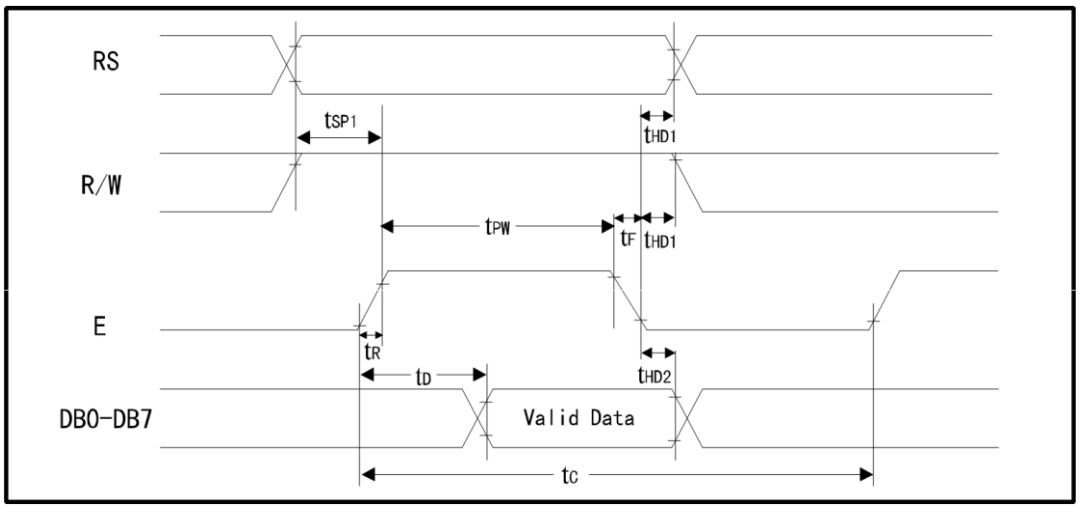


图 3-21 读操作时序图

2.写操作时序

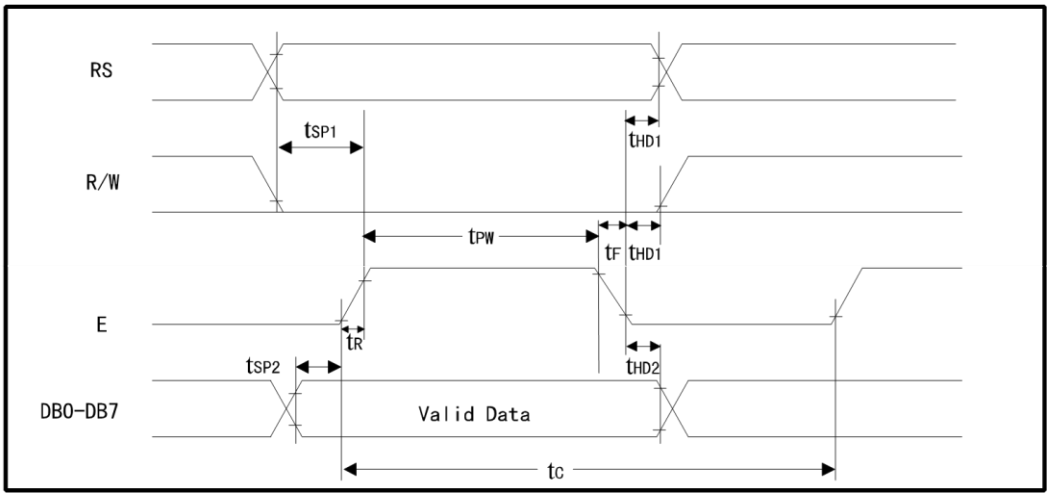


图 3-22 写操作时序图

本系统液晶显示电路如图 3-23 所示。其实现的主要功能：完成密码的输入、修改、开锁、关锁等功能的显示。V0 端口可以接滑动变阻器进行对比度调节，这里接 2KΩ电阻，可以进行正常显示，但无法调节。

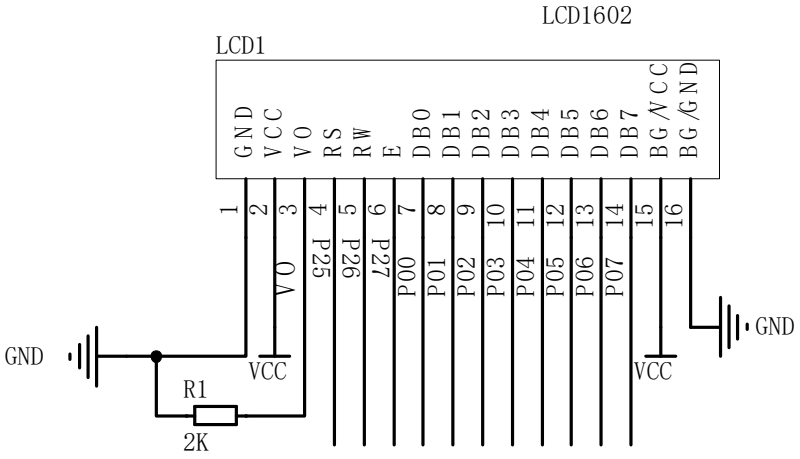


图 3-23 LCD1602 与单片机接线图

4 电子密码锁的软件设计

4.1 软件设计遵循的原则

应用系统中的应用软件是根据系统功能要求设计的，应尽可能地实现系统的各种功能。每一个合格的应用系统软件都应该具有以下原则：

1. 软件结构清晰、简洁、流程合理。

2. 各功能程序实现模块化、子程序化，以便于调试、链接和移植修改。

3. 程序存储区和数据存储区规划合理，这样能节约内存容量，操作又方便。

4. 运行状态实现标识化管理。各个功能程序状态、结果及要求都设置运行状态标志以便查询，程序的转移、运行和控制都可通过状态标志条件来控制。

5. 经过调试修改过的程序应进行规范化，去除修改痕迹，以便于交流和借鉴，方便后面软件的模块化和标准化。

6. 实现软件的抗干扰设计，提高单片机系统应用可靠性。

4.2 AT24C02 读写驱动程序

本系统由AT89S51 的P2.0 和P2.1 来分别模拟SDA和SCL来实现与EEPROM24C02C的I^2C通讯。在AT24C02EEPROM的读写过程中，必须先确定带操作器件的地址，AT24C02 的EEPROM器件地址的固定部分为 1010，A2、A1、A0 三个引脚的不同状态可确定 3 位编码，由此形成的 7 位编码即为该器件的地址码，其格式如下，其中R/W为数据传送方向。

表 4-1 AT24C02 器件的地址组成

1	0	1	0	A2	A1	A0	R/W

主机需对器件进行读写操作时，其操作过程如下：

1. 发送起始信号S（SCL高电平时，SDA产生负跳变）。

2. 发送该器件的 7 位地址码和写方向位“0”，发送完后释放SDA，并在SCL线上产生第 9 个时钟信号，这会触发被选中的存储器器件再确认是自己的地址后，通过将SDA置为低电平来表示对接收到的地址的确认，单片机收到该确认信号后可进行数据的传送。如果接收方没能将SDA置为低电平，主机就会中断传输，而采取适当的错误处理措施。

3. 读写操作

读操作：读操作有两种可能的操作方式，即对当前地址和指定起始地址的读操作。

对当前地址的数据读操作。在主机收到目标器件的确认信号后，逐个读取数据。数据地址按当前存储器地址指针逐个递增。当最后一个字节数据读完之后，主机返回“确认非”信号。

对指定起始地址的数据读操作。在主机收到目标器件的确认信号后，发出一个字节的存储区首地址，待被确认后，主机要重复一次起始信号并发出器件地址和读方向位，收到器件的接收确认后，就可以读出数据字节。当最后一个字节数据读完后，主机应返回以“非确认”信号。

写操作：写操作有两种基本方式，即字节写和页写。

字节写。在主机收到目标器件的确认信号后，将依次发送，一个字节的存储区首地址和待存储的一个字节数据。

页写。在主机收到目标器件的确认信号后，将发送一个字节的存储区首地址。然后逐个发送各数据字节。

在对总线进行操作时，每发送一个字节后都要等待接收方的确认。

当要读或写的数据传送完后，主机应发送结束信号P（SCL高电平时，SDA产生正跳变）以结束读或写操作。读写部分程序代码如下：

```
void write_24c02_8(uchar n,uchar add,uchar *p)
{
    uchar i;
    EA = 0;
    start();
    write_I2C(0xa0);
    ack();
    write_I2C(add);
    for(i=0;i<n;i++)
    {
        ack();
        write_I2C(*p);
        p++;
    }
    no_ack();
    stop();
```

```
    EA = 1;
}
```

数据通信流程图如图 4-1、4-2 所示。

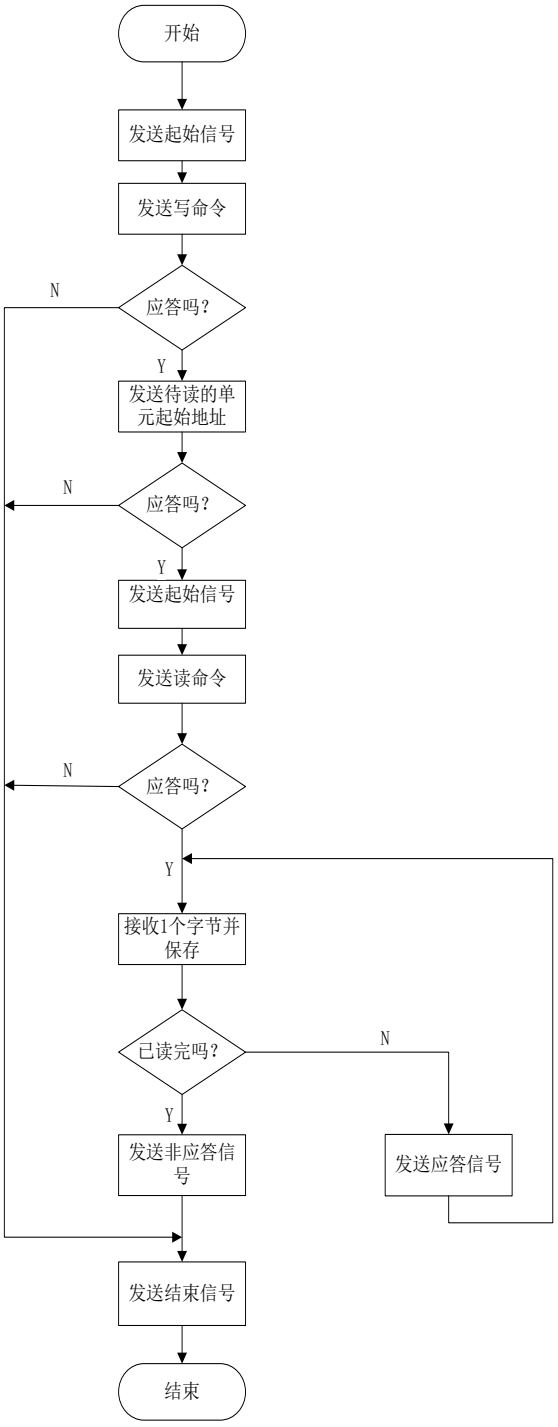


图 4-1 读流程图

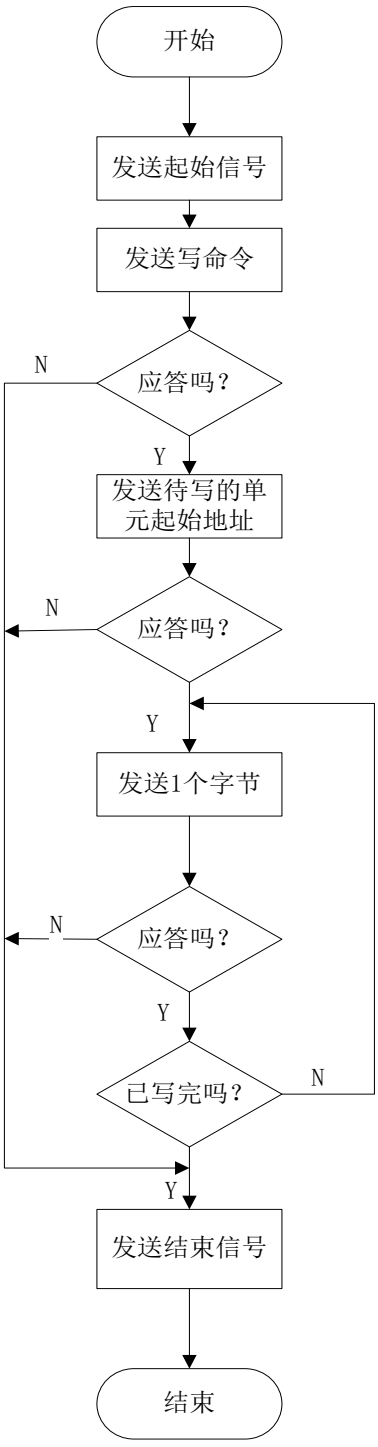


图 4-2 写流程图

4.3 密码输入设置程序

密码输入设置程序流程图如图 4-3 所示。

电子密码锁开机后，出现菜单界面，有两个选项，一是开锁菜单，二是密码修改菜单。选择开锁菜单，输入密码正确，则密码锁打开。选择密码修改菜单，输入初始密码正确，则进入密码修改页面，两次输入新密码，设置新密码成功。

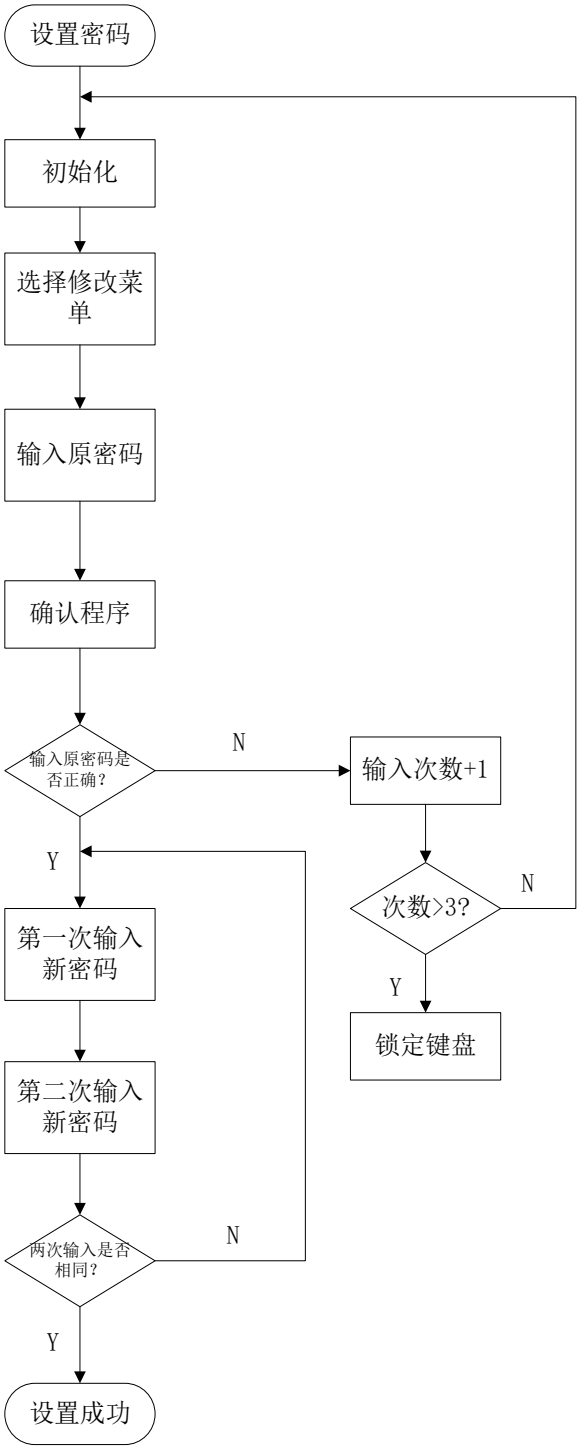


图 4-3 密码输入设置程序流程图

选择密码修改菜单后，程序由主程序进入密码修改程序，首先进行清屏处理，同时输出输入密码提示信息，用户通过键盘输入相应的密码。程序此时接受密码都会以“*”显示在显示屏上，不做其他处理，目的在于防止有人非法套用密码。当用户输入正确密

码按下确定键后，程序进行密码比较，在当输入没有超过三次的情况下，即可实现开锁和密码修改。当非法输入超过三次，程序发出警报信号，并锁定键盘。

4.4 键盘开锁报警程序

开锁流程图如图 4-4 所示。

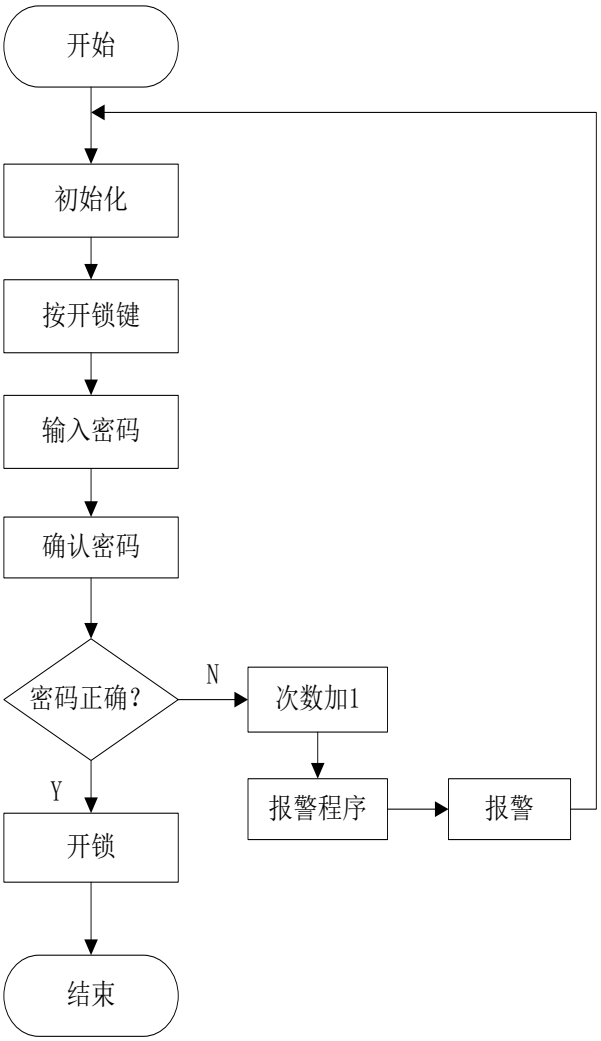


图 4-4 开锁程序流程图

4.5 键盘扫描程序

键盘管理程序的内容包括：

1. 判断有无按键按下。
2. 键盘扫描取得闭合键的行、列信息，去抖动。
3. 根据键的行列信息得到键特征值。
4. 根据键的特征值查表得到键代号。

5. 判断闭合键是否释放，如果没释放则继续等待；若释放则去抖动。

6. 根据键代号去执行该按键所对应的处理程序。

在编程扫描方式下，键盘扫描子程序的流程图如图 4-5 所示。

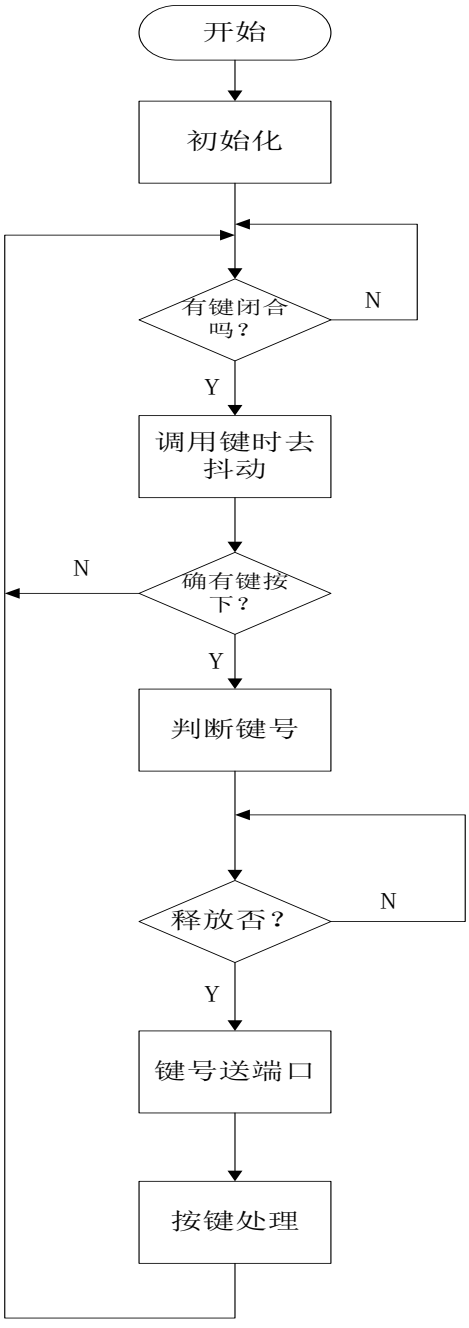


图 4-5 键盘扫描程序流程图

4.6 系统主程序

根据系统的功能实现要求，采用模块化程序设计。主程序主要实现LCD模块的初始化和存储单元的基本分配，和各个子程序的调用。

系统主程序流程图如图 4-6 所示。

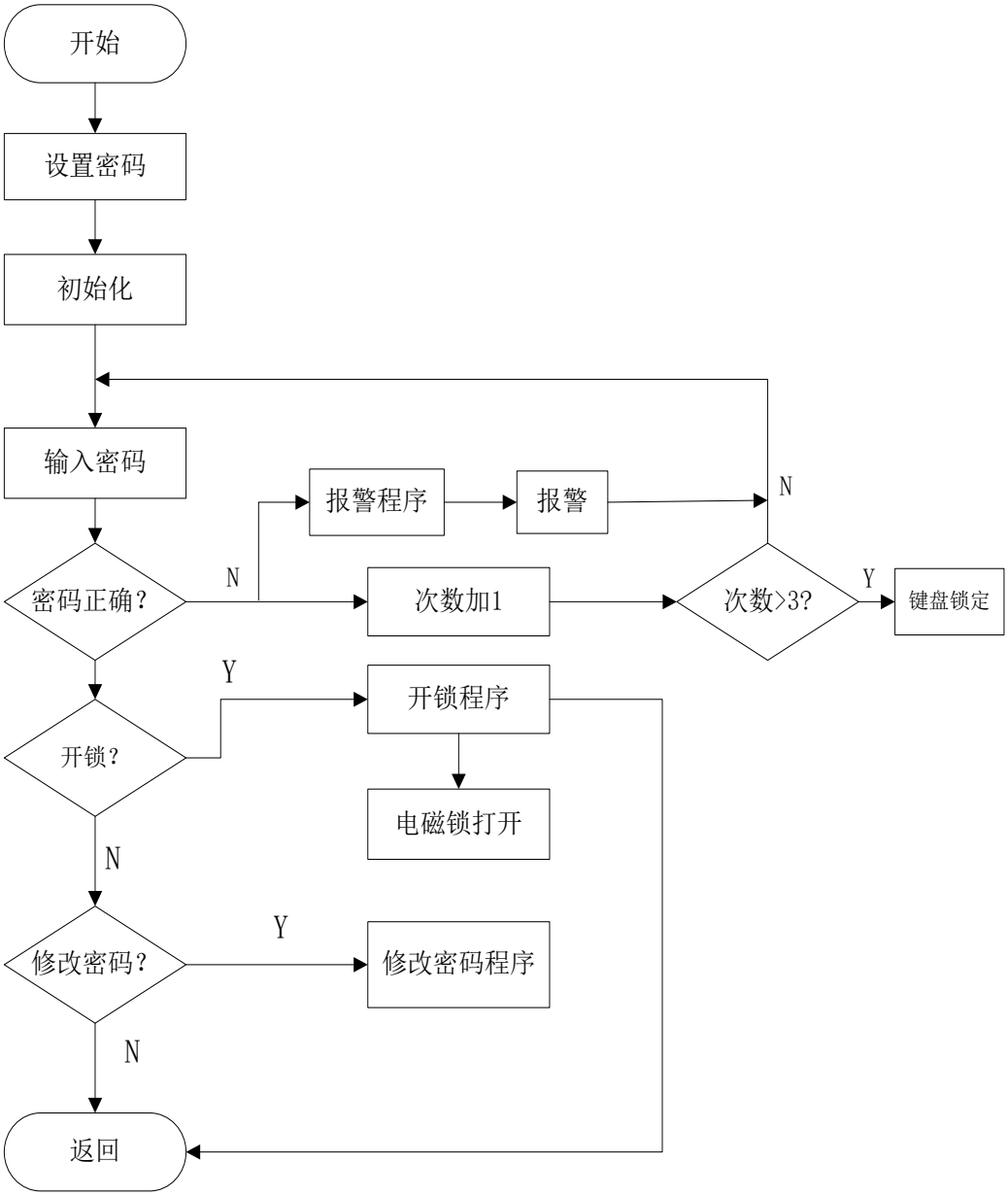


图 4-6 系统主程序流程图

5 电子密码锁系统的仿真与测试

5.1 仿真综述

单片机控制系统的调试包括硬件调试和软件调试。硬件调试即根据设计的原理电路做好电子密码锁硬件电路，然后便可以进入硬件调试阶段。调试工作的主要任务是排除硬件故障，包括设计错误和工艺性故障。由于条件限制，硬件调试主要是进行脱机检查：用万用表笔逐步按照逻辑图检查各器件的电源及各引脚的连接是否正确，检查数据总线、地址总线和控制总线是否短路。为保护芯片，对各底座的电位进行检查，确定无误后再插入芯片检查。

软件调试是使用Keil进行编程，生成HEX文件，在Proteus中进行仿真。

本系统采用Proteus软件进行仿真。Proteus支持众多 51 架构的单片机芯片，可以进行编辑、编译及程序仿真，还支持汇编和C语言程序设计，在调试程序和软件仿真方面有很广泛的应用。Proteus可以仿真单片机及外围电路，在仿真时不仅能观察到程序执行时单片机寄存器和存储器等内容的变化，而且可以直观的看到外围电路的工作情况，非常接近实际电路应用。Proteus软件提供大量的仿真设备和元件，还具备各种调试工具和调试信号。因此用Proteus实现单片机及外围电路仿真很方便，比如模拟电路仿真、数字电路仿真、单片机外围电路仿真、I^2C调试器、SPI调试器、键盘和LCD系统仿真等。

Proteus仿真具有以下特点：

1. 全部满足单片机软件仿真系统的标准，并在同类产品中具有明显的优点。该软件的单片机仿真库里有 51 系列、PIC系列、AVR系列等，并支持大量的存储器和外围芯片。

2. 仿真基于Spice 3F5，具备进行模拟分析、数字分析、数字信号分析等电路分析。

3. 提供虚拟示波器、信号发生器、逻辑分析仪、电表等虚拟仪表使用。

4. 能绘制原理图及PCB图。

5.2 仿真过程

5.2.1 软件程序调试

本系统程序是基于Keil μVision4 的单片机C程序调试。

Keil C51 是美国Keil Software公司出品的 51 系列兼容单片机C语言软件开发系统，与汇编相比，C语言在功能性、结构性、可读性和可维护性上有明显的优势。Keil提供了

包括C编程器、宏汇编、连接器、库管理和一个功能强大的仿真调试器等在内的完整开发方案，通过一个集成开发环境（μVision）将这些部分组合在一起。

软件程序调试时，首先对源程序进行编译，源程序编译通过，表明语法正确，却并不能保证该程序能够正确运行。还需要对其逻辑功能进行调试。Keil软件具有很强的软件仿真功能。程序调试时，程序可以单步执行或连续执行。连续执行是指一条指令执行完后接着连续立即执行下一条，中间不停止。这样程序执行的速度很快。可以看到程序执行的总体效果。即最终结果是正确或错误。但如果程序有错，则难以确认具体出错地方。单步执行指每执行完一条指令后即停止。等待命令执行下一行程序，此时可以观察该条指令执行后得到的实际结果。对比分析是否与预期结果一致。借此可以找到程序中的错误原因所在。这种方式的缺点是需要时间长，排查错误效率很低。尤其当程序很大时。因此，对于用户确认没有错误的程序段可以采用连续运行。对怀疑出错或容易出错的地方则单步执行。对比排查错误，这样调试的效率将会高一些。

程序开发过程中通常情况下需要综合运用单步执行、连续执行、设置断点、观察变量等各种程序调试方法，并且不断积累总结调试经验。这种程序调试能力也是衡量软件开发者水平高低的标志之一。

5.2.2 硬件Proteus的仿真

运行Proteus的ISIS后，首先选择所需类型的元器件，然后添加元件，将仿真原理图绘制完成。电路图绘制完成后，添加由Keil生成的HEX文件，完成程序添加工作，便可以进行系统仿真。Proteus进行的是一种交互式仿真，在仿真中可以对键盘按键，控制按钮等进行操作，系统会对相应的操作进行反映。

首先接通电源开机界面如图 5-1 所示。

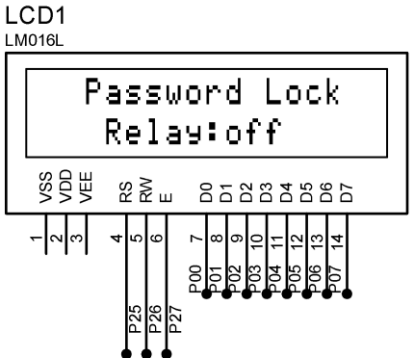


图 5-1 开机界面

按下确认键，显示两个菜单：一是开锁菜单，二是密码修改菜单。菜单界面如图 5-2 所示。

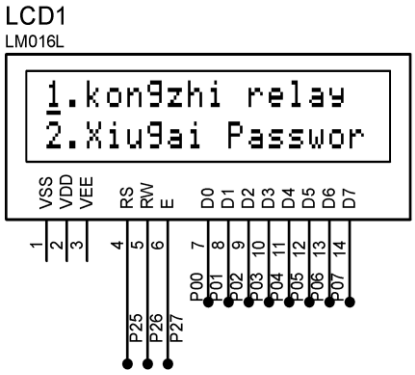


图 5-2 菜单界面

选择开锁菜单，密码输入界面如图 5-3 所示。输入密码正确，则密码锁打开。输入密码错误，报警并显示错误次数。显示错误次数如图 5-4 所示，超过 3 次则键盘锁定。需要通过复位键恢复。

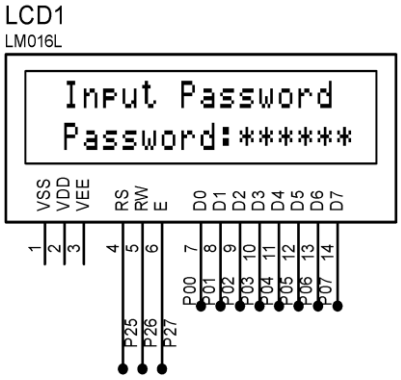


图 5-3 密码输入界面

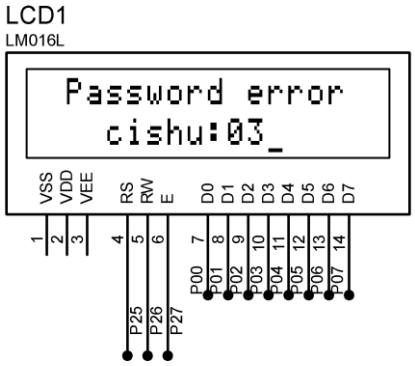


图 5-4 显示错误次数界面

选择密码修改菜单，首先需要输入原始密码，输入正确则进入密码修改界面，如图 5-5 所示。

两次输入新密码，相同，按确认键，则新密码设置成功。

输入错误则显示密码输入错误，显示错误次数，错误次数超过 3 次，则锁定键盘。锁定时间为 1 分钟，时间到则蜂鸣器响 3 声，键盘锁定解除，可以继续输入密码开启密码锁，或修改密码。

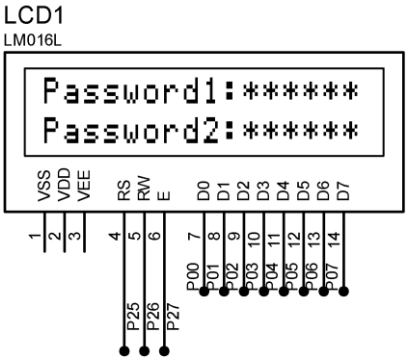


图 5-5 重置密码界面

密码设置成功，则为如下界面，如图 5-6 所示。

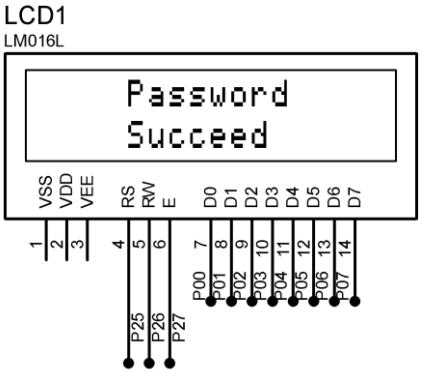


图 5-6 重置密码成功界面

密码设置失败，则为如下界面，如图 5-7 所示。

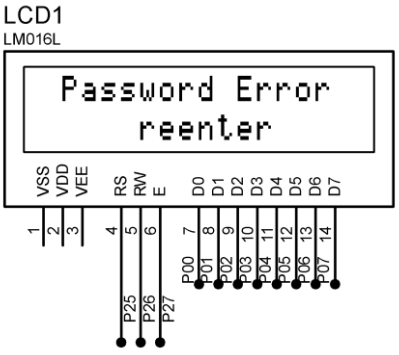


图 5-7 重置密码失败界面

在进行仿真调试时，一般会采取Keil与Proteus联调，联调时，先运行Proteus的ISIS，绘制好电路原理图，然后在Keil集成开发环境下，将应用程序编译好，单击Debug下的Start/Stop Debug Session后，Keil会将机器码自动加载到Proteus ISIS单片机模型中去，不用再次手动绑定HEX文件。在Keil中进行程序的单步、过程调试时，对应Proteus中的电路则会按照程序的语句做出相应的动作。

5.3 电子密码锁实物图

电子密码锁实物图如图 5-8 所示，经测试各项功能均可使用。

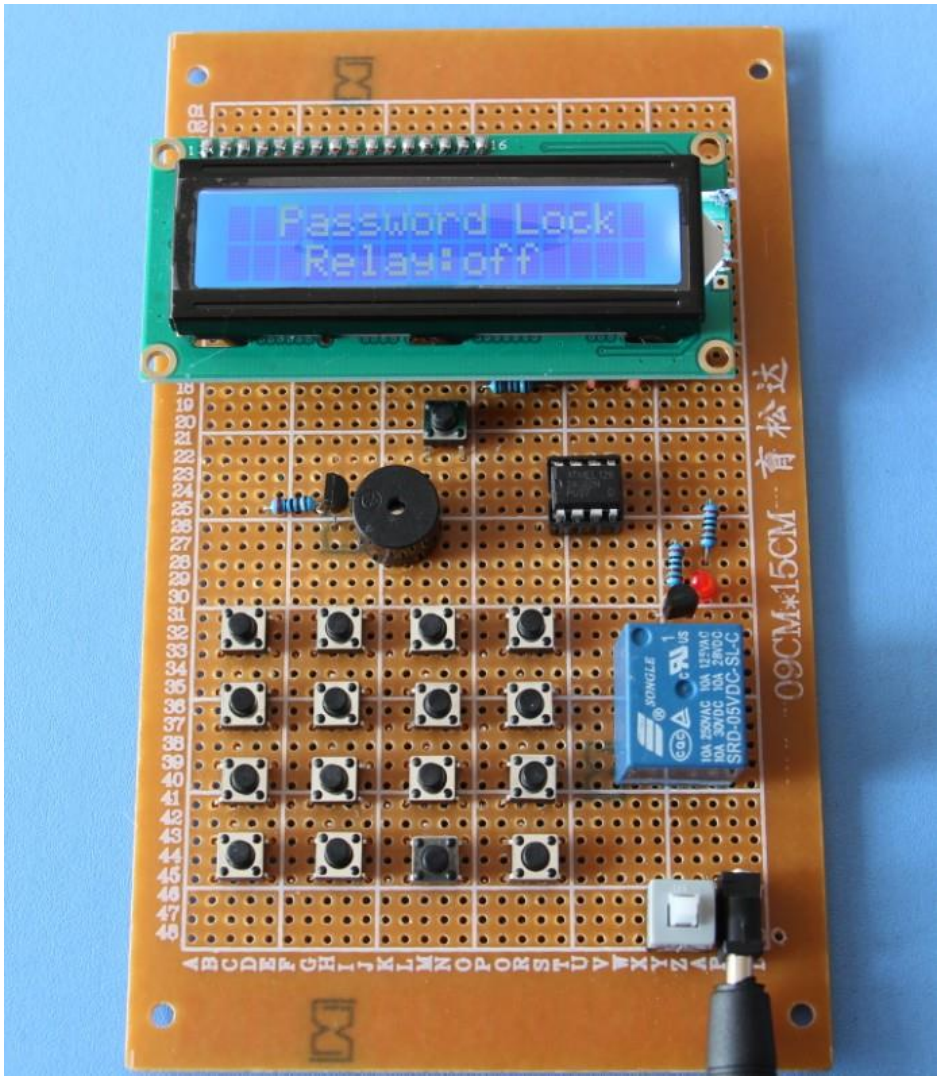


图 5-8 电子密码锁实物图

结束语

本设计对当前广泛使用的，与居民生活息息相关的锁具进行了研究与探讨，通过查阅文献，比较国内外市场上出现的电子密码锁的产品性能和设计方法，提出了基于单片机的电子密码锁设计，并对其进行了详细的系统硬件和软件设计分析，最终完成了产品实物的制作，并进行了验证和调试。各项性能和指标都达到了设计目标要求。

设计基于单片机的电子密码锁的工作过程中重点考虑了单片机控制系统的使用与分析。采用EEPROM AT24C02 来实现系统的密码存储工作，AT24C02 与单片机采用I2C总线通讯方式。系统具有体积小、制造成本低、功耗低、稳定性高等优点，能满足居民现有日常工作生活安全所需。本设计主要有以下几个方面：

1. 完成了系统的硬件电路设计。

2. 完成了系统的软件设计。

3. 完成了系统软、硬件的调试。

4. 完成了系统实物的制作。

由于时间上的限制，加上本人能力有限，本设计仍然存在一些问题需要解决：

1. 系统的安全性需要进一步加强，避免密码的丢失和破解这些问题上还需要解决。

2. 电子密码锁还可以添加更多功能，比如LCD显示模块还可以添加万年历功能，可以添加遥控通信等功能，更具有实用性，方便人们生活。

3. 应该注重改善锁具本身的便捷性和美观性。在功能完善的前提下，注意外观锁的设计和完善，以满足用户需要，适应市场需求。

同样，基于单片机密码锁系统的缺点也很明显：

1. 利用单片机设计需要外围电路较多，安全性差，密码易破解。

2. 利用单片机设计的电路逻辑性较差。

3. 电路结构复杂，需要利用单独的存储器才能完成。

密码锁产品虽然品种丰富，制作精良，但改进和更新还任重而道远，需要更深的理论知识。本设计具有较强的社会实践性，文中提到的设计思路和控制措施，乃至软件编程思想完全适用于很多类似的控制系统，具有一定的参考价值。

参考文献

[1]黄勤. 单片机原理及应用[M]. 北京：清华大学出版社，2010.9

[2]彭伟. 单片机 C 语言程序设计 100 例：基于 8051+Proteus 仿真[M]. 北京：电子工业出版社，2012.10

[3]艾永乐，付子义. 模拟电子技术基础[M]. 北京：中国电力出版社，2008

[4]贾宗璞，许合利. C 语言程序设计[M]. 北京：人民邮电出版社，2010,9

[5]郑晓霞. 基于 AT89S51 单片机实验开发系统设计[硕士学位论文]. 内蒙古大学，2009

[6]张云，周明辉，周海林等. 基于 AT89S51 的多功能电子密码锁设计[J]. 电子设计工程.2010,18（6），2010.6

[7]王慧军. 基于 AT89S51 单片机控制的电子密码锁设计[J]. 装备制造技术 2010，（5）66-67

[8]林嘉. 基于 89S52 的 LCD1602 程序设计[J]. 电脑知识与技术，第 8 卷第 26 期，2012.9

[9]卢旭锦. 基于 Keil C 的 AT24C02 串行 E2PROM 的编程[J]. 现代电子技术，2007,（8） 154-160

[10]杨勇，王为民.MCS-51 系列单片机结构化程序设计探讨[J]. 电子技术应用，2007,（7） 24-26

致谢

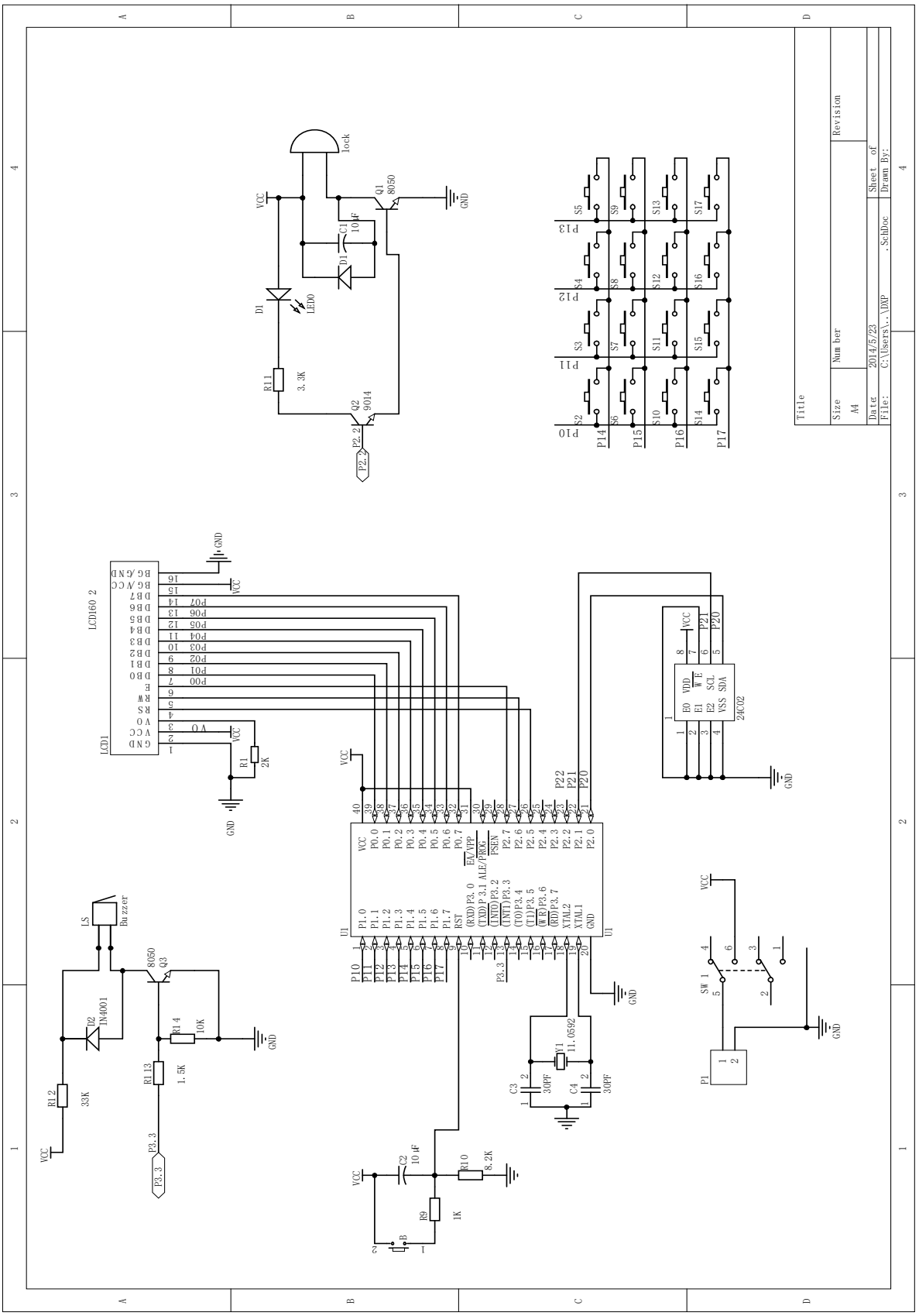
在本科学习生活的四年时间里，很多人给予了我巨大的帮助和鼓励，使我能够顺利完成课程和学位论文的撰写，在这里对一直支持我的人们致以衷心的感谢！

首先，要感谢指导老师，在老师严谨的治学态度和勤恳的工作精神的熏陶下，我努力克服学习中遇到的各种困难。当毕业设计中遇到难以解决的问题时，卜老师总是能够百忙之中抽出时间为我指点和讲解，给与我悉心的指导和帮助。这段经历必将成为我人生中的一笔宝贵财富，使我受益匪浅。

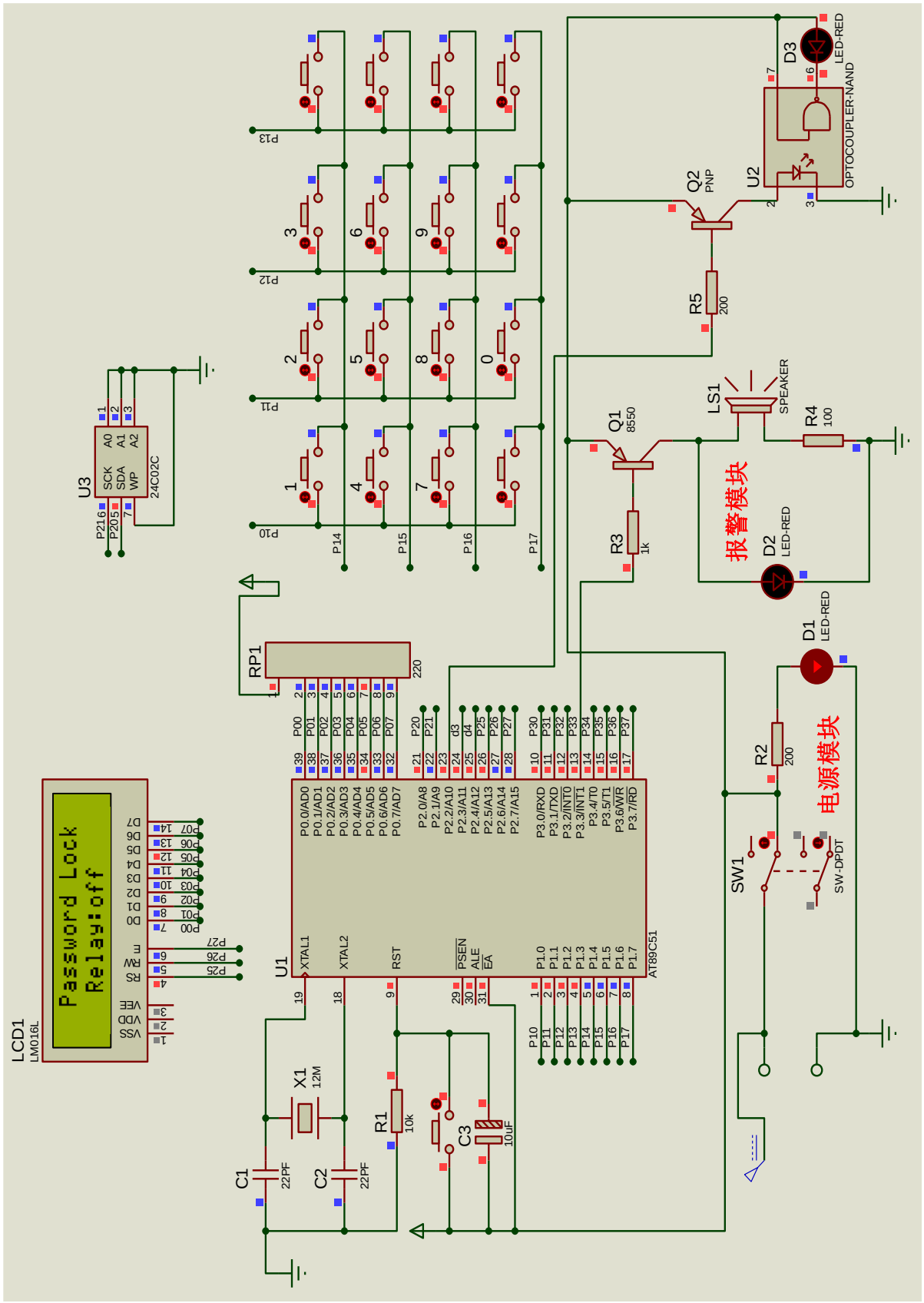
其次要感谢同学们，学习中遇到难题时，与他们一起讨论总能够打开思路，使人豁然开朗；生活中他们也给予我很多帮助，在此向他们表示感谢。还要感谢我的父母四年来对我的支持，使我能够全身心的投入到学习和研究之中，为以后学习和工作打好基础。

最后感谢河南理工大学四年来对我的培养和教育，是学校为我们配备了良好的师资和优越的实验环境，感谢学校为我们所做的努力。

附录 1 系统硬件结构图


Lock
Q1
8050
GND
VCC
C1
10μF
D1
D1
LED0
R11
3.3K
Q2
9014
P2.2
P2.2
S5
S9
S13
S17
S4
S8
S12
S16
S3
S7
S11
S15
S2
S6
S10
S14
P13
P12
P11
P10
P14
P15
P16
P17
Title
Size
A4
Number
Revision
Date
2014/5/23
File:
C:\Users\...\DNP
.SchDoc
Sheet of
Drawn By:
LCD160 2
LCD1
R1
2K
U1
P0.0
P0.1
P0.2
P0.3
P0.4
P0.5
P0.6
P0.7
P2.7
P2.6
P2.5
P2.4
P2.3
P2.2
P2.1
P2.0
P22
P21
P20
24C02
LS
Buzzer
8050
Q3
D2
IN4001
R12
33K
R13
1.5K
R14
10K
P3.3
C2
10 μF
R10
8.2K
R9
1K
Y1
11.0592
C3
30PF
C4
30PF
SW 1
P1

附录 2 系统仿真电路图


Password Lock
Relay:off
LCD1
LM016L
U1
AT89C51
U2
OPTOCOUPLER-NAND
U3
24C02C
D1
LED-RED
D2
LED-RED
D3
LED-RED
Q1
8550
Q2
PNP
LS1
SPEAKER
R1
10k
R2
200
R3
1k
R4
100
R5
200
RP1
220
X1
12M
C1
22PF
C2
22PF
C3
10uF
SW1
SW-DPDT
报警模块
电源模块

附录 3 部分程序清单

1. AT24C02 读写驱动程序

```
#ifndef _I2C_H_
#define _I2C_H_
sbit scl = P2^1;
sbit sda = P2^0;
void start()
{
    sda = 1;
    scl = 1;
    sda = 0;
    scl = 0;
}
void stop()
{
    sda = 0;
    scl = 1;
    sda = 1;
    scl = 0;
}
void ack()
{
    uchar i;
    sda = 1;
    scl = 1;
    while((i < 100) & (sda == 1))  i++;
    scl =0;
}
void ack_zhuji()
```

```
{
    sda = 0;
    scl = 1;
    scl = 0;
    sda = 1;
}
void no_ack()
{
    sda = 1;
    scl = 1;
    scl = 0;
}
void write_I2C(uchar dat)
{
    uchar i;
    for(i=0;i<8;i++)
    {
        scl = 0;
        dat <<= 1 ;
        sda = CY;
        scl = 1;
    }
    scl = 0;
}
uchar read_I2C()
{
    uchar i,value;
    for(i=0;i<8;i++)
    {
        scl = 1;
        value <<= 1;
```

```
        if(sda == 1)
            value |= 0x01;
        scl = 0;
    }
    return value;
}
void write_24c02(uchar add,uchar dat)
{
    start();
    write_I2C(0xa0);
    ack();
    write_I2C(add);
    ack();
    write_I2C(dat);
    no_ack();
    stop();
}
uchar read_24c02(uchar add)
{
    uchar value;
    start();
    write_I2C(0xa0);
    ack();
    write_I2C(add);
    ack();
    start();
    write_I2C(0xa1);
    ack();
    value = read_I2C();
    no_ack();
    stop();
```

```
    return value;
}
void write_24c02_8(uchar n,uchar add,uchar *p)
{
    uchar i;
    EA = 0;
    start();
    write_I2C(0xa0);
    ack();
    write_I2C(add);
    for(i=0;i<n;i++)
    {
        ack();
        write_I2C(*p);
        p++;
    }
    no_ack();
    stop();
    EA = 1;
}
void read_24c02_8(uchar n,uchar add,uchar *p)
{
    uchar i;
    EA = 0;
    start();
    write_I2C(0xa0);
    ack();
    write_I2C(add);
    ack();
    start();
    write_I2C(0xa1);
```

```
    ack();
    for(i=0;i<n;i++)
    {
        p[i] = read_I2C();
        if(i == n-1)
        {
            no_ack();
            stop();
        }
        else
            ack_zhuji();
    }
    EA = 1;
}
#endif
```

2. 新密码输入设置部分程序

```
void menu_2_1_1_init()
{
    menu_3 = 1; menu_4 = 0;
    key_break = menu_2_init;
    key_up_down = null;
    key_shuzi = menu_2_1_1_shuzi_n;
    key_delete = menu_2_1_1_delete_n;
    menu_i = 0;
    smg_i = 6;
    for(i=0;i<6;i++)
    {
        if(password_bj[i] == password[i])
        {
            flag_password = 1;
```

```
        }
        else
        {
            flag_beep_en = 1;
            init_menu();
            flag_password = 0;
            flag_password_cichu2 ++;
            write_string(1,0," Password error ");
            write_string(2,0,"      cishu:            ");
            write_sfm2(2,9,flag_password_cichu2);
            for(i=0;i<6;i++)
            {
                beep = ~beep;
                delay_1ms(200);
            }
            beep = 1;
            delay_1ms(500);
            menu_2_1_init();
            break;
        }
    }
    if(flag_password == 1)
    {
        flag_password_cichu2 = 0;
        write_string(1,0,"Password1:          ");
        write_string(2,0,"Password2:          ");
        lcd1602_guanbiao(1,10);
        key_shuzi = menu_2_1_1_shuzi_n;
        key_break = menu_2_init;
         key_up_down = null;
        key_delete    = menu_2_1_1_delete_n;
```

```
            clear_shuzu(password_bj);
            clear_shuzu(dis_smg);
            dis_smg[0] = 0xc0;
             menu_i = 0;
        }
}
void menu_2_1_2_init()
{
        menu_3 = 2; menu_4 = 0;
        key_break = menu_2_init;
        key_delete = menu_2_1_2_delete_n;
        key_shuzi = menu_2_1_2_shuzi_n;
        key_up_down = null;
        flag_password_cichu2 = 0;
        clear_shuzu(password_xg);
        clear_shuzu(dis_smg);
        lcd1602_guanbiao(1,10 + 0x40);
         menu_i = 0;
}
void menu_2_1_3_init()
{
        menu_3 = 3; menu_4 = 0;
        key_break = menu_2_init;
        key_up_down = menu_1_1_1_n;
        key_shuzi = null_1;
        key_delete = null;
        menu_i = 0;
        for(i=0;i<6;i++)
        {
            if(password_bj[i] == password_xg[i])
            {
```

```
            flag_password = 1;
        }
        else
        {
            flag_beep_en = 1;
            init_menu();
            flag_password = 0;
            write_string(1,0," Password Error ");
            write_string(2,0,"        reenter       ");
            delay_1ms(300);
            for(i=0;i<6;i++)
            {
                beep = ~beep;
                delay_1ms(300);
            }
            beep = 1;
            delay_1ms(500);
            menu_2_init();
            break;
        }
    }
    if(flag_password == 1)
    {
        for(i=0;i<6;i++)
        {
            write_string(1,0,"       Password       ");
            write_string(2,0,"       Succeed        ");
            password[i] = password_bj[i];
        }
        write_24c02_8(6,0,password);
        delay_1ms(300);
```

```
            beep = 0;
            delay_1ms(1000);
            beep = 1;
            key_shuzi = menu_2_1_1_shuzi_n;
            key_break = menu_2_init;
             key_up_down = null;
            key_delete    = menu_1_1_delete_n;
            clear_shuzu(password_bj);
            clear_shuzu(dis_smg);
            menu_2_init();
        }
}
void menu_2_1_1_fun()
{
        switch(menu_3)
        {
              case 0:
                    menu_2_1_1_init();
                    break;
              case 1:     beep = 0;
                   menu_2_1_2_init();
                    break;
              case 2:
                     beep = 0;
                   menu_2_1_3_init();
                    break;
        }
}
#endif
```

3. 系统主程序

```
#include <reg52.h>
#define uchar unsigned char
#define uint    unsigned int
#include "lcd1602.h"
#include "I2C.h"
uchar value,i;
uchar flag_lj_en;
uchar flag_lj_en_value;
sbit relay = P2^2;
sbit beep    = P3^3;
uchar smg_i;
uchar dis_smg[6];
uchar password[6]={6,5,4,3,2,1};
uchar password_bj[6]={1,2,3,4,5,6};
uchar code password_r[6] = {6,5,4,3,2,1} ;
uchar password_xg[6];
uchar flag_password;
uchar flag_password_cichu1;
uchar flag_password_cichu2;
bit flag_200ms=1;
bit flag_beep_en;
bit flag_relay_en;
uchar key_can;
#include "I2C.h"
void delay_100us(uchar z)
{
    uchar x,y;
    for(x=0;x<z;x++)
        for(y=0;y<30;y++);
}
void delay_1ms(uint q)
```

```
{
    uint i,j;
    for(i=0;i<q;i++)
        for(j=0;j<120;j++);
}
void key()
{
    static uchar key_new = 0, key_l;
    key_can = 20;
    P1 = 0x0f;
    if((P1 & 0x0f) != 0x0f)
    {
        delay_1ms(1);
        if(((P1 & 0x0f) != 0x0f) && (key_new == 1))
        {
            key_new = 0;
            key_l = (P1 | 0xf0);
            P1 = key_l;
            switch(P1)
            {
                case 0xee:    key_can = 1;    break;
                case 0xde:    key_can = 4;    break;
                case 0xbe:    key_can = 7;    break;
                case 0x7e:    key_can = 10;    break;

                case 0xed:    key_can = 2;    break;
                case 0xdd:    key_can = 5;    break;
                case 0xbd:    key_can = 8;    break;
                case 0x7d:    key_can = 0;    break;

                case 0xeb:    key_can = 3;    break;
```

```
                case 0xdb:    key_can = 6;    break;
                case 0xbb:    key_can = 9;    break;
                case 0x7b:    key_can = 11;    break;

                case 0xe7:    key_can = 15;    break;
                case 0xd7:    key_can = 14;    break;
                case 0xb7:    key_can = 13;    break;
                case 0x77:    key_can = 12;    break;
            }
        }
    }
    else
    {
        key_new = 1;
        flag_lj_en = 0;
    }
}
void password_return()
{
    if(flag_lj_en == 1)
    {
        flag_lj_en_value ++;
        if(flag_lj_en_value > 13)
        {
            flag_lj_en_value = 0;
            flag_lj_en = 0;
            write_24c02_8(6,0,password_r);
            beep = 0;
            delay_1ms(200);
            beep = 1;
            read_24c02_8(6,0,password);
```

```
            }
        }
}
void clear_shuzu(uchar *p)
{
    for(i=0;i<6;i++)
        p[i] = ' ';
}
void time_init()
{
    EA    = 1;
    TMOD = 0X01;
    ET0   = 1;
    TR0   = 1;
}
#include "menu.h"
void menu_dis()
{
    if(menu_1 == 0)
    {
        if(relay == 1)
            write_string(2,0,"    Relay:off     ");
        else
            write_string(2,0,"    Relay:open    ");
    }
}
void password_chushifa()
{
    scl = 0;
    value = read_24c02(10) ;
    if(value != 99)
```

```
    {
        value = 99;
        beep = 0;
        write_24c02(10,value);
        delay_1ms(200);
        write_24c02_8(6,0,password_r);
        delay_1ms(200);
        read_24c02_8(6,0,password);
        beep = 1;
    }
}
void main()
{
    static uint value ;
    beep = 0;
    delay_1ms(150);
    P0 = P1 = P2 = P3 = 0xff;
    password_chushifa();
    time_init();
    init_menu();
    read_24c02_8(6,0,password);
    init_1602();
    init_1602_dis_csf();
    while(1)
    {
        if(flag_password_cichu1 < 3)
        {
            key();
            if(key_can < 20)
            {
                key_with();
```

```
            }
        }
        if(flag_200ms == 1)
        {
            flag_200ms = 0;
            menu_dis();
            if(flag_password_cichu1 >= 3)
            {
                value ++;
                if(value >= 5 * 60)
                {
                    value = 0;
                    flag_password_cichu1 = 0;
                    for(i=0;i<6;i++)
                    {
                        beep = ~beep;
                        delay_1ms(150);
                    }
                }
            }
            password_return();
        }
        delay_1ms(1);
    }
}
void time0_int() interrupt 1
{
    static uchar value;
    TH0 = 0x3c;
    TL0 = 0xb0;
     value ++;
```

```
    if(value % 4 == 0)
    {
        flag_200ms = 1;
    }
}
```