基于单片机的超市自动存包柜

集美大学诚毅学院信息工程系

自动化专业　2012届　陈道乐 201241055104

摘要　本文详细介绍了自动化存包柜的设计方法，本设计分为电路设计部分与程序设计部分。主要功能有：当顾客接近锁存柜时随机生成时钟自动生成4位密码，当顾客需要取出物品时需要将生成的密码输入即可取出物品，并且有3次容错的机会。当3次机会用完时系统就判断当前锁存柜处于不安全的状态将自动锁死系统，只有进入管理员模式才能将系统解锁，管理员模式可以将自己的管理员密码重置，并且在管理员模式下可以解锁指定柜子，默认密码为8888。使用的主要电路模块有：AT89c52、1602 、4\*4键盘、HC-SR04距离传感器、LED显示矩阵。

关键词 单片机 电子密码锁 存包柜 自动化 超声波测距

**Automatic lockers microcontroller-based supermarket**

Chen Daole

201241055104，Automation Major,2016

Dept. of Information Engineering,

Chengyi University College of Jimei University

**Abstract：**This paper introduces the design method of automated storage bag ark, include circuit design program design part: features when customers close to latch ark randomly generated four clock automatically generated password, when customers need to take out the items need to be generated password can remove items, and four fault tolerance of the opportunity, when opportunity out of four think latches ark is in a state of insecurity of automatic lock system, only the system into the administrator mode to unlock, can put your own administrator password reset administrator mode, And in the administrator mode can unlock the designated cabinet,the default password is 8888. The main components are: AT89c51, 4 \* 4 keyboard HC - 1602 SR04 distance sensor, simulation package cabinet LED display.

**Key words:** Microcontrollers Digital code lock Locker automation Ultrasonic distance measurement

目 录

[引 言 3](#_Toc421263590)

[第1章 绪论 3](#_Toc421263591)

[1.1 超市自动存包概述 3](#_Toc421263595)

[1.2 设计任务及要求 3](#_Toc421263592)

[1.3 设计目的 3](#_Toc421263595)

[第2章 硬件设计 3](#_Toc421263597)

[2.1 自动存包柜的工作原理 3](#_Toc421263598)

[2.2 主控芯片介绍 3](#_Toc421263601)

[2.3 显示模块1602芯片介绍 3](#_Toc421263601)

[2.4 HC-SR04超声波测距芯片介绍 3](#_Toc421263601)

[2.5 4X4键盘电路设计 3](#_Toc421263601)

[2.6 控制电路设计 3](#_Toc421263601)

[2.7 显示电路设计 3](#_Toc421263601)

[2.8 测距电路设计 3](#_Toc421263601)

[2.9 模拟存储柜LED板 3](#_Toc421263601)

[第3章 软件设计 3](#_Toc421263602)

[3.1 语言选择 3](#_Toc421263603)

[3.2 程序设计 3](#_Toc421263604)

[3.2.1 程序模块 3](#_Toc421263604)

[3.2.2 系统程序流程图 3](#_Toc421263604)

[3.2.3 四位随机密码生成详解 3](#_Toc421263604)

[第4章 仿真调试 3](#_Toc421263605)

[4.1 仿真软件简介及仿真流程 3](#_Toc421263606)

[4.1.1 程序编写软件keil 3](#_Toc421263606)

[4.1.2 软硬件仿真Protues及流程 3](#_Toc421263606)

[4.2 仿真过程 3](#_Toc421263607)

[第5章 使用说明 3](#_Toc421263605)

[5.1 用户模式 3](#_Toc421263605)

[5.2 管理员模式 3](#_Toc421263605)

[结 论 3](#_Toc421263614)

[致 谢 3](#_Toc421263615)

[参考文献 3](#_Toc421263616)

[附录1： 3](#_Toc421263617)

[附录1.1 电路仿真图 3](#_Toc421263618)

[附录1.2 超市自动存储柜源程序 3](#_Toc421263619)

# 引 言

随着世界自动化产业的不断发展，自动化技术将不断普及民用生活的各个角落。在中国，存包柜的发展基本随着超市的发展而发展，愈来愈多的购物商城和超市如雨后春笋一样在城镇和农村拔地而起，顾客数量随之快速增长，存包的发展由最初的人工存包到机械存包，在相当长的一段时间内主导着寄存市场，但是不论大型的，类似“万达购物广场”，还是小型的农村超市，都需要在顾客购物时，进行存包的服务，以免顾客的物品和商品混淆。

在超市，购物商城，私营商店等等，人流量大的地方，为顾客提供物品寄存的服务就成了必须，如果用人工不但费时，效率低，还需特定人员负责，浪费人力物力，现在是21世纪，科技发达，可以使用单片机这种，廉价便捷的CPU制作存包系统，大大节省成本。

但是不同的超市购买不同的设备，遇到设备故障时，维修不便，且购买新的设备价格昂贵，存储柜不能随各个季度的人流量，进行数量的加减，能保证顾客用上存储柜。超市自动存包柜的设计的优势在于:设备价格便宜，操作人性化，节省时间，免去的随身携带钥匙的不便，维修方便，模块化便于升级，安全。

超市自动存包柜的设计，主旨在于免去顾客多余操作的同时提高安全性。节约存储柜占用的土地面积，友好的用户的界面。

# 第1章 绪论

**1.1.1超市自动存包柜概述**

超市自动存包柜的设计，主旨在于免去顾客多余操作的同时提高安全性。节约存储柜占用的土地面积，友好的用户的界面。 基于AT89c52的单片机设计的超市存储柜，不但价格便宜，容易维护。目前市面上流行的自动存包柜主要有：

（1） 机械存包柜（使用机械锁，安全低）

（2） 机设条码式存包柜（价格昂贵，不环保）

（3） 指纹式电子柜 （价格昂贵，识别率低）

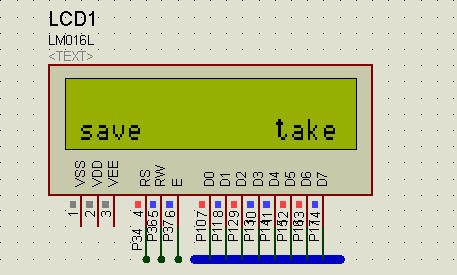
（4） 非接触IC卡存包柜（需要携带IC卡）

在公共场合如此需要安全便捷的存储柜，就需要我们设计出安全廉价，便捷易维修的存储柜以保证人民群众便捷的生活，因此，研制一款自动存包柜的设备就赋予了重大的意义，本设计的超市自动存包柜能满足人民生活的基本需要。

**1.1.2 设计任务及要求**

设计以单片机为控制核心,利用4X4矩阵键盘、LED/LCD显示屏、开关，实现如下功能：按照柜子编号管理10个空柜，系统根据已存包的柜子编号随意产生3位取包密码；用户输入取包密码打开柜子，系统具有3次容错功能和管理员解锁功能。

图1.1 交互界面



本设计的用户交互界面，如图1.1所示，顾客只需对‘take’进行操作并且输入密码即可获得存放的物品，‘save’部分由系统自动检测顾客和生成密码，并且在身设计中添加了当手的距离在6cm内时密码的显示保持，手的位置离开6cm后自动清除密码显示回到主界面。

**1.1.3 设计目的**

随着自动化产业发展和普及，超市数量不断增长，为了满足顾客存放物品的需求，就必须出现一个兼顾安全和廉价的设计，剔除多余的操作，传统的机械存储柜，需要随身携带钥匙，容易丢失，并且安全性也并不高，为了解决这个问题，本设计使用了经典的89c5X系列的单片机，既能保持稳定，又能在价格方面得到保证。

# 第2章 硬件设计

## 2.1 自动存包柜的工作原理

本设计使用AT89C52作为整个设计的核心，主要外部模块分为：输入部分，输出部分，输入：测距模块HC-SR04，4\*4键盘模块，输出：显示模块1602，模拟存储柜LED模块，主控芯片处理输入的信息，判断当前处于什么状态，核对内部存储的数据，处理前一个状态，输出下一个状态，当处理输入完毕后，跳转至实时监控输入部分。

电路设计概况如下图2.1.1：

图 2.1.1 自动存包柜设计方案

模拟存储柜

LED板

1604

4\*4键盘

HC-SR04

89C52

**2.2 主控芯片介绍**

单片机全称单片微型计算机（Single Chip Micorcomputer）又称微控制器MCU（Micro Controller Uint），是计算机的基本功能微型化，使之集成在一块芯片上的微机。STC89C52RCzhuyao用于控制，目前主流的单片机仍然是8位的MCU占据主流，只有在精度要求特别高的场合如图像处理，高性能电机控制，才采用16位或32位的MCU。本设计使用8位的MUC，单片机的特点主要有：体积小，功能全，功耗低，性能价格比高运行速度快，抗干扰能力强，详细引脚图如图 2.2.1 所示

图 2.2.1 STC89C51RC引脚图



**2.3显示模块1602介绍**

液晶显示器以其微功耗、体积小、显示内容丰富、超薄轻巧的诸多优点， 在袖珍式仪表和低功耗应用系统中得到越来越广泛的应用。字符型液晶模块是一种用 5x7 点阵图形来显示字符的液晶 显示器，根据显示的容量可以分为 1 行 16 个字、2 行 16 个字、2 行 20 个字等等，本设计使用的是 2 行 16 个字的 1602 液晶模块，1602 液晶模块内部的字符发生存储器（CGROM)已经存储了 160 个不 同的点阵字符图形，如表 1 所示，这些字符有：阿拉伯数字、英文字母 的大小写、常用的符号、和日文假名等，每一个字符都有一个固定的代 码，比如大写的英文字母“A”的代码是 01000001B（41H），显示时模 块把地址 41H 中的点阵字符图形显示出来，就能看到字母“A。

一.参数设置流程：

1.初始化设置：

显示模式设置，详见表格2.3.3

显示开/关及光标设置，详见表格2.3.4

二.数据控制：

控制器内部设有一个数据地址指针，用户通过他们来访问内部全部80字节ROM。

1.数据指针设置，详见表格2.3.5

2.读数据

3.写数据

4.其他设置详见表格2.3.6

表2.3.1 主要技术参数

|  |  |
| --- | --- |
| 显示容量： | 16X2个字符 |
| 芯片工作电压： | 4.5至5.5 |
| 工作电流： | 2.0mA（5.0） |
| 模块最佳工作电压： | 5.0 |
| 字符尺寸： | 2.95X4.35（WXH）mm |

表2.3.2 接口型号说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 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（H/L） | 12 | D5 | Data I/O |
| 5 | R/W | 读/写选择端（H/L） | 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 |

表2.3.3初始化模式设置指令说明

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 指令码 | | | | | | | | 功能 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 设置16X2显示，5X7点阵，8位数据接口 |

表2.3.4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=不显示光标  B=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一，且光标加一  N=0 当读或写一个字符后地址指针减一，且光标减一  S=1 当写一个字符，整个屏幕显示左移（N=1）或右移（N=0），以得到光标不移动的效果  S=0 当写一个字符，整个屏幕显示不移动 |

表2.3.5数据指针设置

|  |  |
| --- | --- |
| 指令码 | 功能 |
| 80H+地址码（0 - 27H，40H - 67H） | 设置数据地址指针 |

表2.3.6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： 1.数据指针清零 |

## 2.3 HC-SR04超声波测距芯片介绍

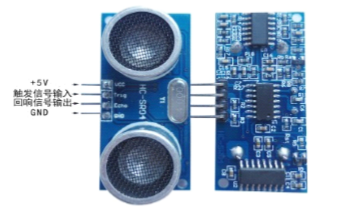
1.产品特点：

HC-SR04超声波测距模块可提供2cm-400cm 的非接触式距离感测功能，测距精度可达高到 3毫米；模块包括超声波发射器、接收器与控制电路。

1.基本工作原理： (1)采用 IO 口 TRIG 触发测距，给最少 10us 的高电平信呈。 (2)模块自动发送 8 个 40khz 的方波，自动检测是否有信号返回； (3)有信号返回，通过 IO 口 ECHO 输出一个高电平，高电平持续的时间就是超声 波从发射到返回的时间。测试距离=(高电平时间\*声速(340M/S))/2;

1. 实物图：

3.电器参数



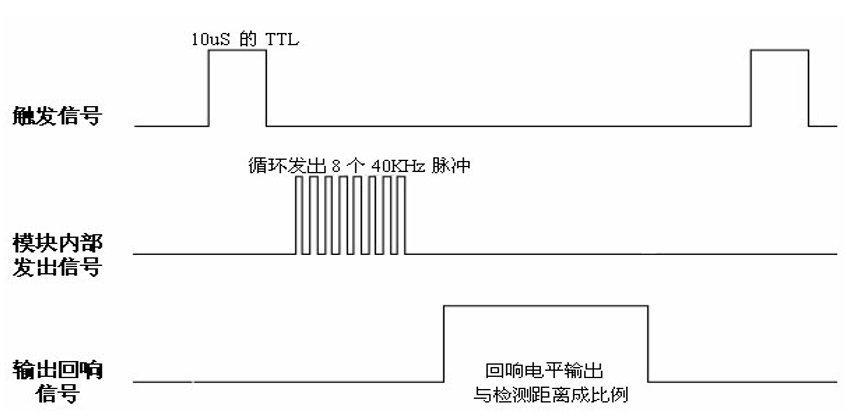
如右图接线，VCC 供 5V 电源，GND 为地线， TRIG 触发控制信号输 入，ECHO 回响信号输出

等四个接口端。

图3-1

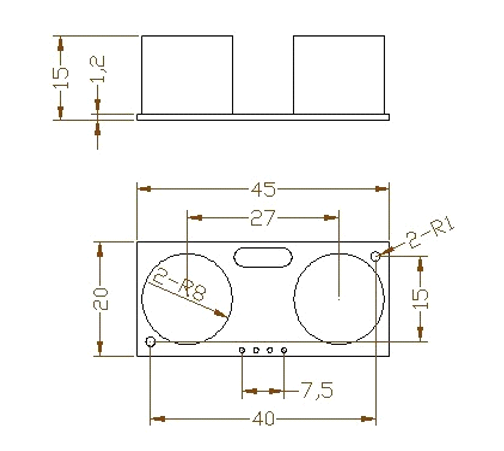
|  |  |
| --- | --- |
| 电器参数 | HC-SR04超声波模块 |
| 工作电压 | DC 5V |
| 工作电流 | 15mA |
| 工作频率 | 40kHz |
| 最远射程 | 4m |
| 最近射程 | 2cm |
| 测量角度 | 15度 |
| 输入触发信号 | 10us的TTL脉冲 |
| 输出回响信号 | 输出TTL电平信号，与射程成比例 |
| 尺寸规格 | 45\*20\*15mm |

4.超声波时序图



超声波时序图

5.实物规格



该模块在本设中为后期添加，前期主要使用一颗按键模拟用户操作，主要原因是应为在本设计使用的仿真软件中没有该模块的仿真器件，所以在添加该模块时全部使用实物调试，在距离调试中也是为了符合用户的使用规律，进行了一步一步的测试，的出在6cm距离内最为何时，同时也满足了调试的需要。

## 2.4 4X4键盘电路设计

1.4X4矩阵键盘的工作原理

矩阵键盘又称行列式键盘，它是用4条I/O线作为列线组成的键盘。在行线和列线的每个交叉点上，设置一个按键，这样键盘中按键的个数是4X4。这种行列式键盘结构能够有效的提高单片机系统中的I/O的利用率。

如图2.4.1所示电路仿真图

F

E

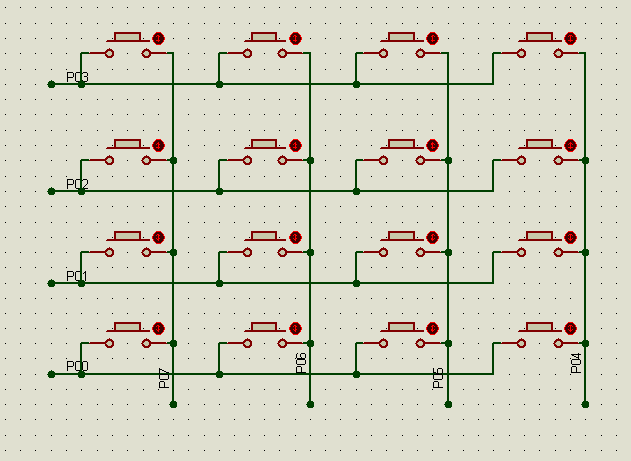
D

C

B

A

9

0

8

7

6

5

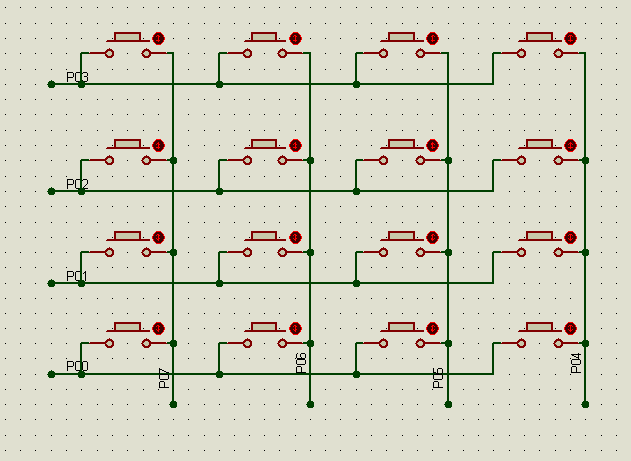
4

3

2

1

图2.4.1 4X4键盘电路仿真图



本设计使用该电路设计主要是因为该设计能节省引脚的数量，每一个按键按下后将触发唯一的信号，端口进行扫描，为了能够快速的检测到信号，本设计不仅提高了晶振，而且为了防止抖动带来的错误信号，本设计在程序做了判断，本设计在后期添加了HC-SR04模块，为了保证原有程序整体设计思路不变，所以在键盘扫描中直接添加对HC-SR04模块的程序，截取部分代码如下：

#include "reg52.h"

#include "SR04.h"

#include "timer.h"

short int checkkey(void)

{

unsigned char n;

P2=0xf0;

n=P2;

if(n!=0xf0)

return 1;

else

return 0;

}

int key()

{

int NO=0;

int sign=1;

unsigned char i,j;

while(sign)//从此处开始添加了了对HC-SR04模块的处理

{

if(sr04() == 1)//距离在6厘米内

{

return 100; // 返回标志信号

}

if(checkkey()==1)

{

delay(1);

if(checkkey()==1)

P2=0xf0;

i=P2;

P2=0x0f;

j=P2;

i=i|j;

switch(i)

{

case 0x7e:NO=0,sign=0;break;

case 0xbe:NO=1,sign=0;break;

case 0xde:NO=2,sign=0;break;

case 0xee:NO=3,sign=0;break;

case 0x7d:NO=4,sign=0;break;

case 0xbd:NO=5,sign=0;break;

case 0xdd:NO=6,sign=0;break;

case 0xed:NO=7,sign=0;break;

case 0x7b:NO=8,sign=0;break;

case 0xbb:NO=9,sign=0;break;

case 0xdb:NO=10,sign=0;break;

case 0xeb:NO=11,sign=0;break;

case 0x77:NO=12,sign=0;break;

case 0xb7:NO=13,sign=0;break;

case 0xd7:NO=14,sign=0;break;

case 0xe7:NO=15,sign=0;break;

}

while(checkkey()==1); //等待按键弹起

}

}

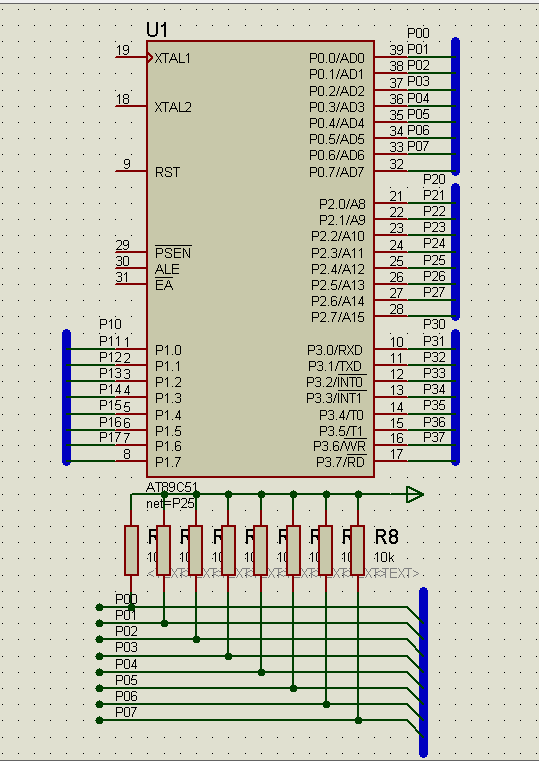
return NO;

}

## 2.6 控制电路设计

控制电路I/O口分配如图2.6.1所示，在P0口添加上拉电阻，增加驱动能力，使其能够驱动LED灯，另外我们也使用P3.0和P3.1口驱动LED，本设计使用P1口驱动1602显示器的数据口，RS对应P3.4，RW对应P3.6，E对应P3.7，使用P2口扫描键盘以获取键值信息，本设计中使用HC-SR04超声波测距模块其中trig对应P3.2，echo对应P3.3。

图 2.6.1 控制电路仿真图



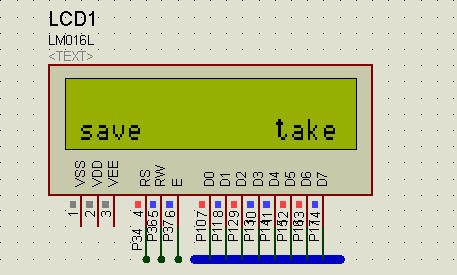
## 2.7 显示电路设计

本实验中使用了1602作为本设计的显示器，如图2.7.1所示VSS接地，VDD接电源，VEE本实验串接了3k电阻调节对比度，外接设备正极统一接线到电源正极，对地端统一接线到电源负极，详见图 2.7.1

图2.7.1

显示器作为向用户展示该设计的操作流程肩负着重要的功能，界面简介易懂是本设计的重要理念。

图2.7.1



本节截取部分代码介绍：

void write\_com(uchar com) //命令子函数command

{

lcdrs=0;//接受指令

P1=com; //给P0口送指令码

delay(5);

lcden=1; //E变为高电平

delay(5);//在此延迟期间，将指令送入

lcden=0; //E变为低电平，高脉冲结束

}

void write\_data(uchar date) //数据子函数data

{

lcdrs=1; //接受数据

P1=date; //给P0口送数据

delay(5);

lcden=1; //E变为高电平

delay(5); //在此延迟期间，将数据送入

lcden=0; //E变为低电平，高脉冲结束

}

void init() //初始函数

{

lcdrw=0;

lcden=0; **//**E的初始值为低电平，后面的程序给高电平从而满足E为高脉冲

write\_com(0x38); //写入显示模式指令

write\_com(0x0e); //显示是否打开以及光标的设置

write\_com(0x06); //地址指针的加减和整屏是否移动

write\_com(0x01); //清屏指令

write\_com(0x80+0x00); //显示字符的初始位置

}

void write\_lcd(uint offset,uchar tab[])

{

uint num;

write\_com(0x80+offset); //显示字符的初始位置

for(num=0;num<16;num++)

{

write\_data(tab[num]);

delay(20);

}

}

上述程序主要描述了如何在初始化中写入控制码和写入数据，由于单片机的使用8位并行数据传输，在本程序中也是用unsigned char 类型数据，该数据类型也是8位，在程序中对P1口输出。

## 2.8 测距电路设计

本设计使用HC-SR04超声波测距模块，模块详细参数请参见第二章2.4节，本设计根据模块测得的参数判断物体的距离，从而判断是否生成四位密码。因protues中无该器件的仿真模块故本设计在添加该模块时使用实物调试。

详细代码操作：

void init\_sr04()//初始化器件，并初始化定时器

{

trig=0;

EA=1;

ET1=1;

ET0=1;

TMOD=0x10;

TH1=0;

TL1=0;

s=0;

TR1=0;

delay(1000);

}

unsigned int location()//启动超声波测距

{

init\_sr04();

trig=1;

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

trig=0;

while(!echo);

TR1=1;

while(echo);

TR1=0;

sj=TH1\*256+TL1;//计算时间

s=sj\*340/2/1000;//计算距离

return s;

}

int sr04()//判断距离是否符合要求

{

int k;

k=location();

if(k<60)

{

return 1;

}

else

{

return 0;

}

}

void time1() interrupt 3

{

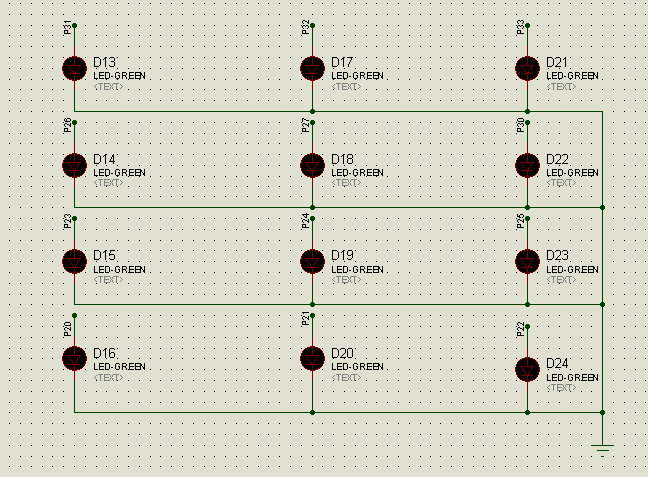
}

上述程序主要使用定时器计算声波折射回来的时间，但是本设计中定时器0已经被使用，所以本设计为了不影响随机数的生成，直接使用另一个定时器，为了尽量简化在主程序中的代码量，本设计直接将判断距离的算法封装在sr04()函数中。

## 2.9 模拟存储柜LED板

本设计在实物设计阶段，发现电压过高故在实物中为每个LED串接一个1K的电阻，为满足在仿真中并没有添加故特此标明。仿真图如图

图2.8.1



在本设计中，当红灯亮起时视为处于空闲状态可以存储物品，但是一旦有顾客进行存放物品的操作的时候，就会对该面板中的一个灯进行处理并视为该柜处于忙状态。

# 第3章 软件设计

## 3.1 语言选择

单片机主流编程语言有C与汇编，现对于汇编C语言具有更具有自然语言的特点，程序容易编写，使用汇编编写比较复杂，代码多，但是他也有C不具有的优势，就是可以自己定义空间，把每个单元都用到极致。对于本设计而言，选择C语言具有明显的优势，本设计需要设计交互的流程故使用C语言。

## 3.2 程序设计

3.2.1程序模块

开/关柜

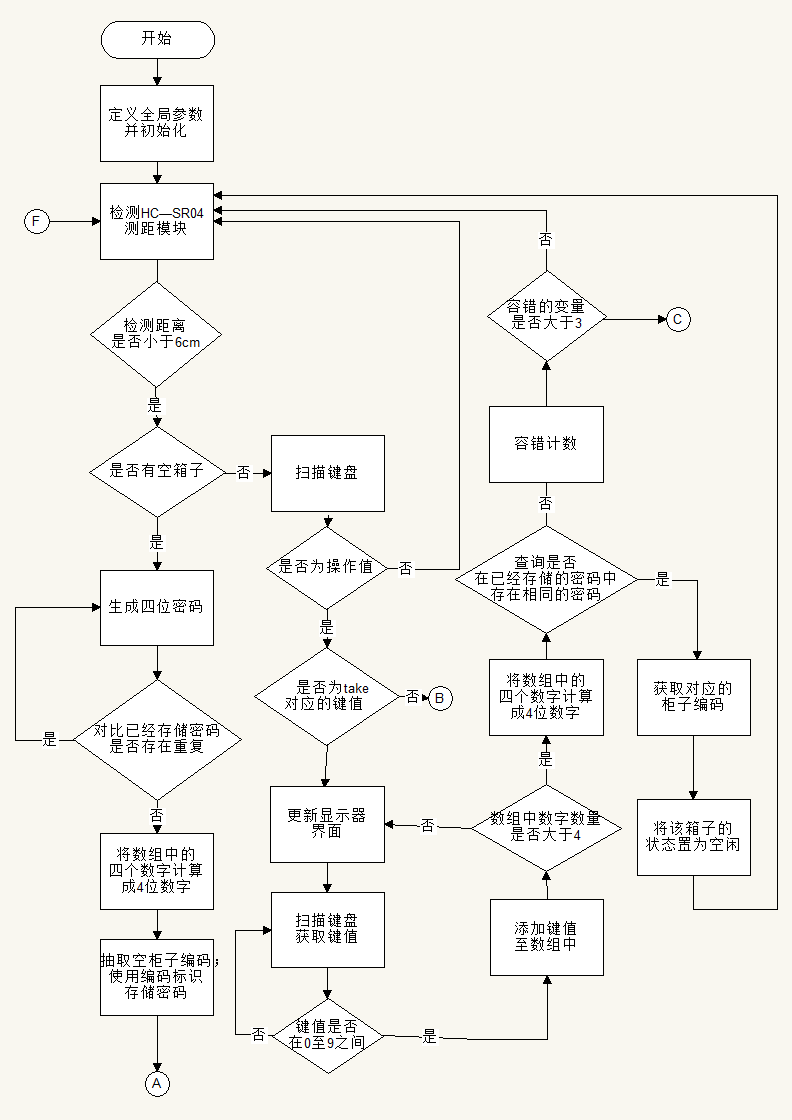
显示

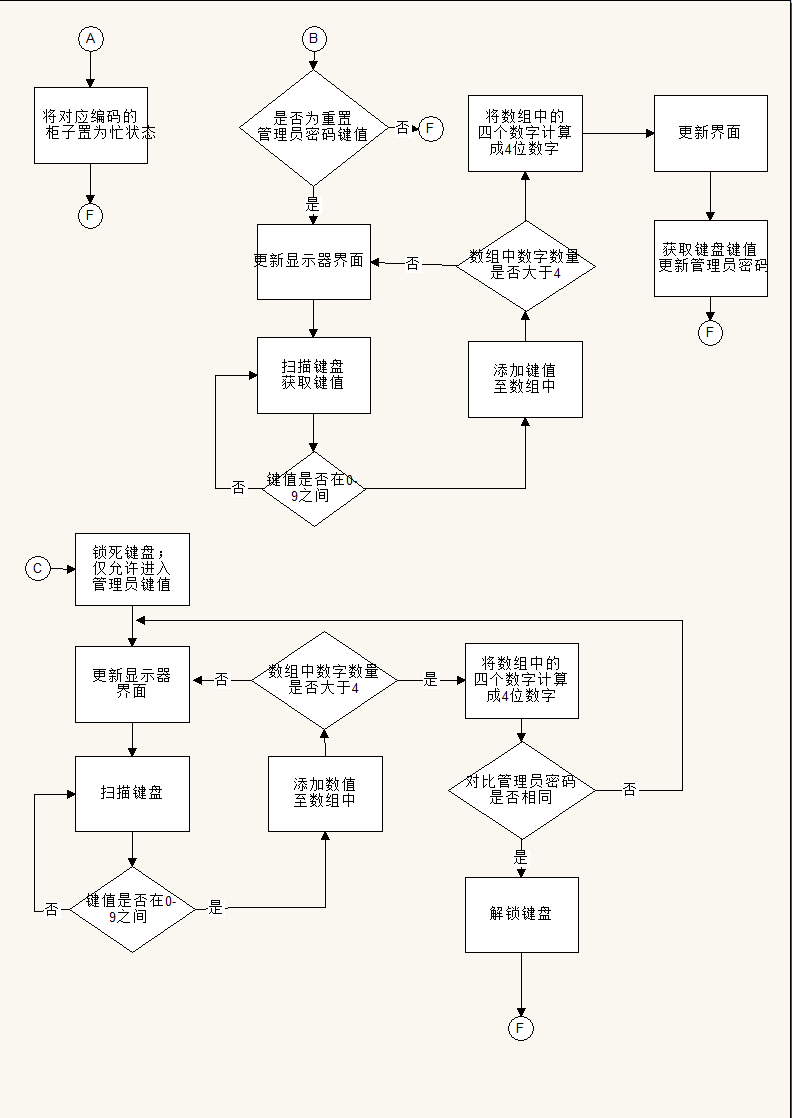
键盘

测距

全局变量定义，参数初始化

3.2.2 系统程序流程图





3.2.2程序代码

详见附录1.2

3.2.3 四位密码生成详解

本程序在初始化过程中就设置了定时器的参数和启动定时器，在进入存放模式之前获取定时器的TH0和TL0中的值，为了检测是否有重复的密码生成所以在提取密码后会对比原先存放数组中的密码是否有重复，如果有重复的密码将重新获取定时器参数和提取密码。

详细程序代码：

#include<reg52.h>

#include "1602.h"

#include "keyboard.h"

#include "LED.h"

#include "SR04.h"

#define uchar unsigned char

#define uint unsigned char

extern unsigned char number[10];

extern uint NO1,NO2;

extern unsigned int boxes[];

int save2()

{

uint temp;

uint iter;

uint userkey\_temp;

uint clock\_num[4];

int i;

for(iter = 0 ; iter<BNUM ; iter++)

{

if(boxes[iter] == 0)

{

restart:clock\_num[0] = NO1%10;

clock\_num[1] = NO1%9;

clock\_num[2] = NO2%10;

clock\_num[3] = NO2%9;

write\_com(0x01);

write\_lcd(0x00,"key: ");

temp =clock\_num[0];

disp\_lcd(0x40,number[temp]);

temp =clock\_num[1];

disp\_lcd(0x41,number[temp]);

temp =clock\_num[2];

disp\_lcd(0x42,number[temp]);

temp =clock\_num[3];

disp\_lcd(0x43,number[temp]);

userkey\_temp = clock\_num[0]\*1000+clock\_num[1]\*100+clock\_num[2]\*10+clock\_num[3];

for(i = 0 ; i<BNUM ; i++) //查询密码是否有相同的柜子

{

if(boxes[i] == userkey\_temp)

{

NO1 = TH0;

NO2 = TL0;

goto restart;

}

}

boxes[iter] = userkey\_temp;

select\_led(iter);

goto L;

}

}

L: while(1)

{

if(location()>80) //返回

{

return 1;

}

}

}

# 第4章 仿真调试

## 4.1 仿真软件简介及仿真流程

4.1.1 程序编写软件keil

本实验使用keil uVision4进行程序编写，Keil C51是美国Keil Software公司出品的51系列兼容[单片机](http://baike.baidu.com/view/1012.htm" \t "http://baike.baidu.com/item/_blank)C语言软件开发系统，与汇编相比，C语言在功能上、结构性、可读性、可维护性上有明显的优势，因而易学易用。Keil提供了包括C[编译器](http://baike.baidu.com/view/487018.htm" \t "http://baike.baidu.com/item/_blank)、宏汇编、链接器、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](http://baike.baidu.com/view/14867.htm" \t "http://baike.baidu.com/item/_blank)（μVision）将这些部分组合在一起。本设计软件使用C语言编程，正好发挥keil的优势，故本设计使用keil进行程序编写。

4.1.2 软硬件仿真protues及流程

[Proteus](https://zh.wikipedia.org/w/index.php?title=Proteus&action=edit&redlink=1" \o "Proteus（页面不存在）)软件是初学者入门的首选软件，它是Labcenter Electronics公司的一款商业版电路设计与仿真软件。内包括ISIS、ARES等软件模块。ARES模块主要用来完成PCB的设计，ISIS模块用来完成电路原理图的布图与仿真。Proteus的软件仿真基于VSM技术，它与其他软件最大的不同也是最大的优势就在于能仿真大量的单片机芯片，比如MCS-51系列、PIC系列等等，甚至ARM处理器，以及单片机外围电路，比如键盘、LED、LCD等等。该软件还附带了一些案例和丰富的帮助文件。基于这些优势本设计使用protues进行硬件设计和软硬件仿真。

设计流程如图4.1.2

完成调试

图4.1.2

不符合/符合要求

软硬联合

调试

根据电路进行软件编写

硬件电路设计

确立目标

## 4.2 仿真过程

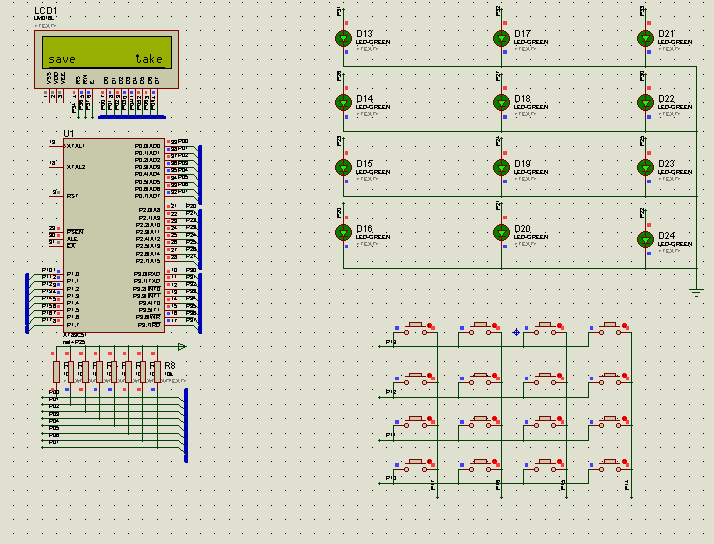
4.2.1 仿真过程操步骤如下：

1. 使用Protue绘制电路图（由于protues没有HC-SR04模块所以在仿真阶段剔除该功能，使用实物直接进行调试）。
2. 使用Keil将程序编译生成hex烧写文件。
3. 在Protues中的89c51芯片装载该烧写文件。
4. 运行仿真，执行相应的操作查看操作是否与设计的流程符合，查找问题所在，对程序进行优化和修改。

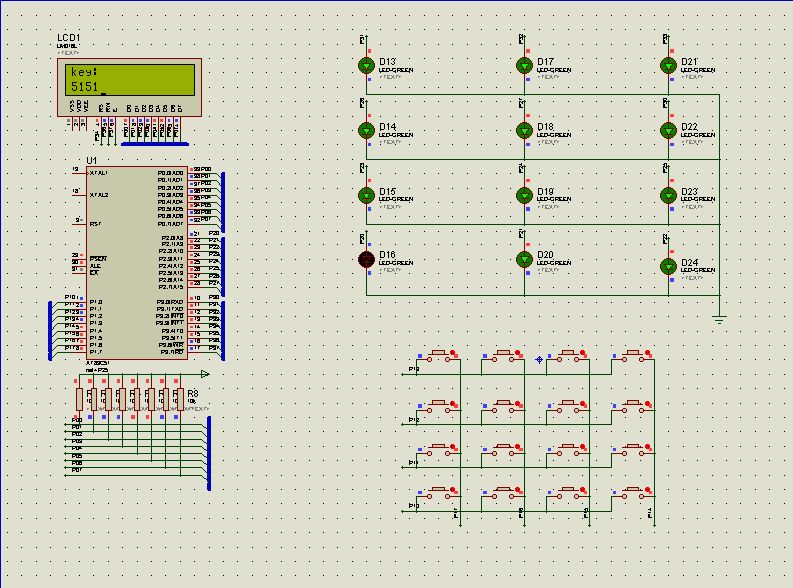
注：本设计中实物的烧写程序不能在仿真中使用，仿真的烧写程序需剔除HC-SR04模块的的程序。

4.2.2 仿真结果

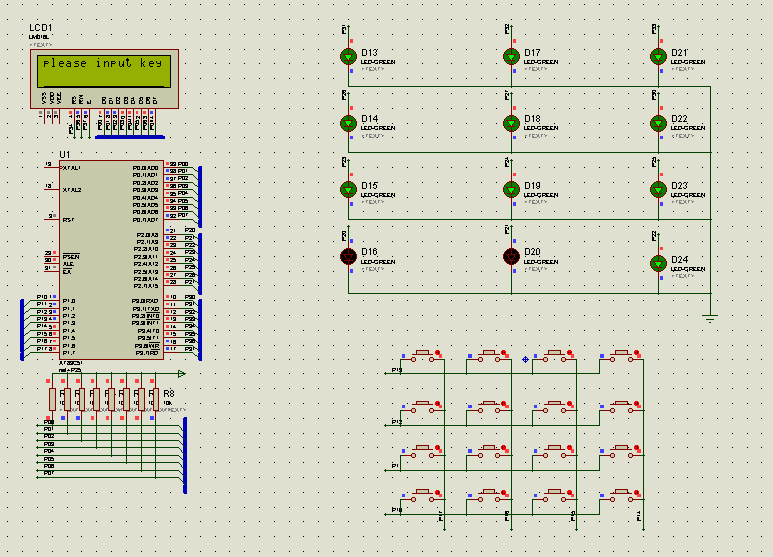
1. 系统仿真



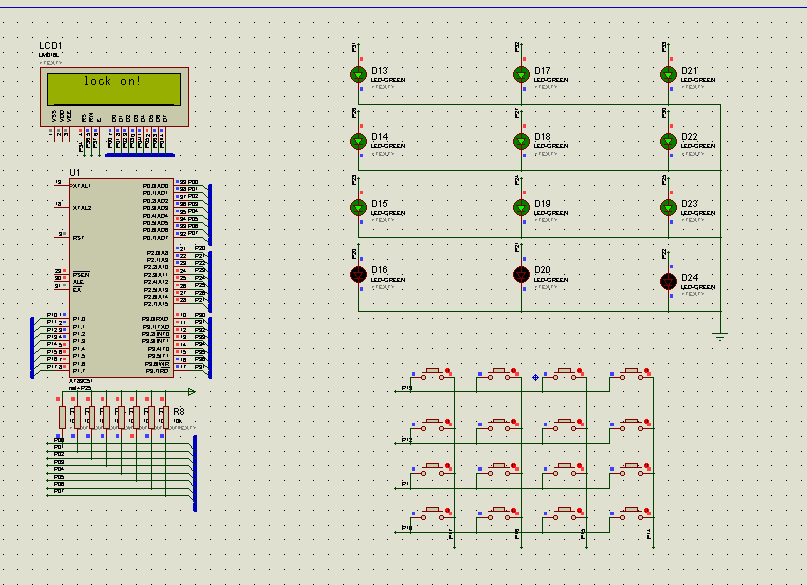
（2）获取四位密码



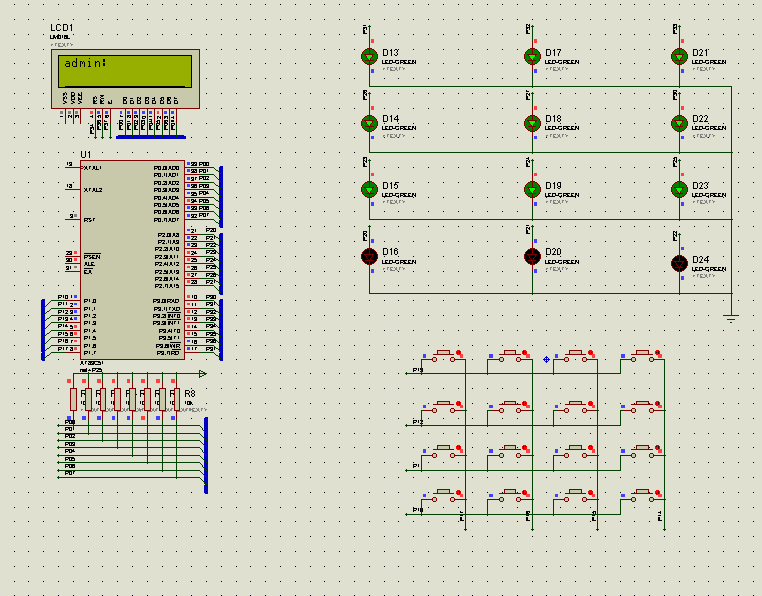
（3）输入四位密码



（4）锁定状态



1. 管理员模式



（6）后期在实物中添加管理员密码重置



# 第5章 使用介绍

5.1 用户模式

在用户模式下视为面向顾客的操作，只有两个操作：取出和存放，就目前而言进行添加HC-SR04模块后取消了使用点击按键生成密码的操作改用挥手生成密码，进一步简化操作，相对与存放操作，取出的操作为了保证安全性依然使用输入四位密码的操作，特别说明取出按钮的键值为B，原有的存放按键键值A已经无效，为了避免未来进一步添加功能，目前保留该按键并不在硬件上剔除。

5.2管理员模式

在管理员模式下视为面向管理员，本设计在设计时考虑到安全问题并没有设计一键解锁所有箱子而是在管理员模式下可以解锁指定编号的箱子，在管理员模式下可以解锁因为顾客不正确输入3次错误密码而造成的锁死，本设计进入到管理员模式是需要4为密码，初始的管理员密码为8888，为了安全起见本设计无论在何时都可以重置密码，特别说明解锁指定编号的柜子的键值为D，重置管理员密码的键值为E，在锁死状态下进入解锁界面的键值为F。

# 

# 结 论

通过本次的自动超市存储柜的设计，了解到单片机的许多知识，不论在电路设计或是软件设计都有了深刻的了解，虽然在设计的地方遇到一些器件无法在电脑上仿真，只能通过不断的实物调试获得结果，正因如此获取了许多实战的经验，本设计在设计电路方面尽量简单，而在程序方面拟补不足，由于HC-SR04模块的添加导致的键盘扫描的速度减低，本设计尽量在硬件方面提高晶振，在程序方面进行优化，虽然提高不少，但还是能感到延迟。整体设计部分，主要先完成了存储和取物这两个基本的功能，其中最重要的是4为密码的生成问题，第一次是尝试直接使用函数生成密码但是发现容易发现规律，并且在每次重启单片机后与重启单片机前生成的代码的规律是相同的，所以本设计从新的角度审视为题，最终使用了提取计时器中的值作为参数的方法，每一次获取的密码都为随机，从这个问题中获得了不少的知识。

# 致 谢

感谢我的夏梦导师，他们严谨细致、一丝不苟的作风一直是我工作、学习中的榜样;他们循循善诱的教导和不拘一格的思路给予我无尽的启迪。

感谢我的老师，这片论文的每个实验细节和每个数据，都离不开你的细心指导。

感谢我的室友们，从遥远的家来到这个陌生的城市里，是你们和我共同维系着彼此之间兄弟般的感情，维系着寝室那份家的融洽。四年了，仿佛就在昨天。四年里，我们没有红过脸，没有吵过嘴，没有发生上大学前所担心的任何不开心的事情。只是今后大家就难得再聚在一起吃每年元旦那顿饭了吧，没关系，各奔前程，大家珍重。我们在一起的日子，我会记一辈子的。

感谢我的家人，焉得谖草，言树之背，养育之恩，无以回报，你们永远健康快乐是我最大的心愿。

在论文即将完成之际，我的心情无法平静，从开始进入课题到论文的顺利完成，有多少可敬的师长、同学、朋友给了我无言的帮助，在这里请接受我诚挚的谢意。

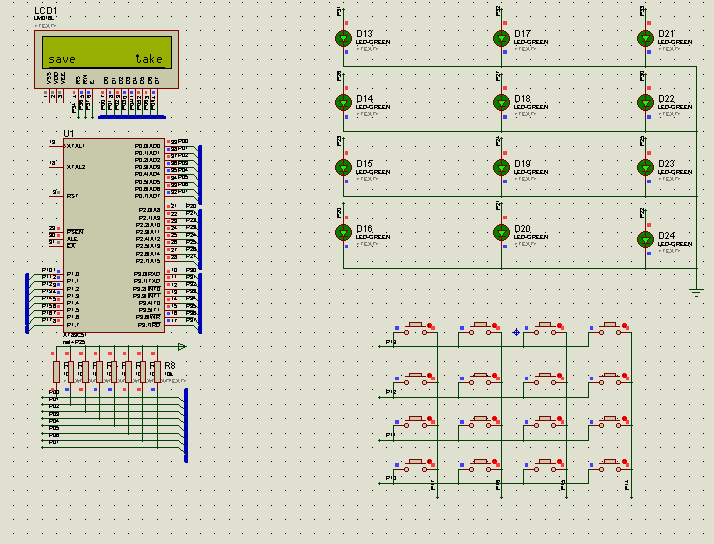
# 参考文献

[1] 张毅刚，彭喜元，彭宇编著. 单片原理及应用（第二版）.北京： 高等教育出版社，2010

[2] 徐爱钧，徐阳著. Kell C51 单片机高级语言应用编程与实践.北京：电子工业出版社，2013

# 附录1：参考文献的格式规范

## 附录1 电路仿真图



## 

## 附录2 超市自动存储柜源程序

/\*定义全局变量，和管脚定义\*/

#ifndef \_\_REG52\_H\_\_

#define \_\_REG52\_H\_\_

#define BNUM 9

/\* BYTE Registers \*/

sfr P0 = 0x80;

sfr P1 = 0x90;

sfr P2 = 0xA0;

sfr P3 = 0xB0;

sfr PSW = 0xD0;

sfr ACC = 0xE0;

sfr B = 0xF0;

sfr SP = 0x81;

sfr DPL = 0x82;

sfr DPH = 0x83;

sfr PCON = 0x87;

sfr TCON = 0x88;

sfr TMOD = 0x89;

sfr TL0 = 0x8A;

sfr TL1 = 0x8B;

sfr TH0 = 0x8C;

sfr TH1 = 0x8D;

sfr IE = 0xA8;

sfr IP = 0xB8;

sfr SCON = 0x98;

sfr SBUF = 0x99;

/\* 8052 Extensions \*/

sfr T2CON = 0xC8;

sfr RCAP2L = 0xCA;

sfr RCAP2H = 0xCB;

sfr TL2 = 0xCC;

sfr TH2 = 0xCD;

/\* BIT Registers \*/

/\* PSW \*/

sbit CY = PSW^7;

sbit AC = PSW^6;

sbit F0 = PSW^5;

sbit RS1 = PSW^4;

sbit RS0 = PSW^3;

sbit OV = PSW^2;

sbit P = PSW^0; //8052 only

/\* TCON \*/

sbit TF1 = TCON^7;

sbit TR1 = TCON^6;

sbit TF0 = TCON^5;

sbit TR0 = TCON^4;

sbit IE1 = TCON^3;

sbit IT1 = TCON^2;

sbit IE0 = TCON^1;

sbit IT0 = TCON^0;

/\* IE \*/

sbit EA = IE^7;

sbit ET2 = IE^5; //8052 only

sbit ES = IE^4;

sbit ET1 = IE^3;

sbit EX1 = IE^2;

sbit ET0 = IE^1;

sbit EX0 = IE^0;

/\* IP \*/

sbit PT2 = IP^5;

sbit PS = IP^4;

sbit PT1 = IP^3;

sbit PX1 = IP^2;

sbit PT0 = IP^1;

sbit PX0 = IP^0;

/\* P3 \*/

sbit RD = P3^7;

sbit WR = P3^6;

sbit T1 = P3^5;

sbit T0 = P3^4;

sbit INT1 = P3^3;

sbit INT0 = P3^2;

sbit TXD = P3^1;

sbit RXD = P3^0;

/\* SCON \*/

sbit SM0 = SCON^7;

sbit SM1 = SCON^6;

sbit SM2 = SCON^5;

sbit REN = SCON^4;

sbit TB8 = SCON^3;

sbit RB8 = SCON^2;

sbit TI = SCON^1;

sbit RI = SCON^0;

/\* P1 \*/

sbit T2EX = P1^1; // 8052 only

sbit T2 = P1^0; // 8052 only

/\* T2CON \*/

sbit TF2 = T2CON^7;

sbit EXF2 = T2CON^6;

sbit RCLK = T2CON^5;

sbit TCLK = T2CON^4;

sbit EXEN2 = T2CON^3;

sbit TR2 = T2CON^2;

sbit C\_T2 = T2CON^1;

sbit CP\_RL2 = T2CON^0;

/\*定义led管脚\*/

sbit led0 = P0^0;

sbit led1 = P0^1;

sbit led2 = P0^2;

sbit led3 = P0^3;

sbit led4 = P0^4;

sbit led5 = P0^5;

sbit led6 = P0^6;

sbit led7 = P3^0;

sbit led8 = P3^1;

#endif

/\*主程序\*/

#include<reg52.h>

#include"1602.h"

#include"keyboard.h"

#include"save.h"

#include"take.h"

#include"admin.h"

#define uchar unsigned char

#define uint unsigned char

int admin = 8888;

uchar number[] ="0123456789";

uint NO1,NO2;

unsigned int boxes[BNUM]={0,0};

int lock;

int main()

{

int num;

int count = 0;

TMOD=0x01;

TH0=(65536-50000)/256;

TL0=(65536-50000)%256;

EA=1;

ET0=1; //开中断

TR0=1;

init();

P0=0xff;

st: lock=0;

while(1)

{

write\_com(0x01);

write\_lcd(0x40,"save take");

num = key();

if(num == 100) //存放模式

{

NO1 = TH0;

NO2 = TL0;

save2();

}

if(num == 11) //取出模式

{

take2();

}

if(num ==13) //管理员制定打开柜子

{

write\_com(0x01);

write\_lcd(0x00,"admin: ");

if(admin\_key() == 1)

{

write\_com(0x01);

admin\_control();

}

}

if(num == 14)//重置管理员密码

{

write\_com(0x01);

write\_lcd(0x00,"admin: ");

if(admin\_key() == 1)

{

write\_com(0x01);

admin\_resetkey();

}

}

if(lock>4)

{

lk:write\_com(0x01);

write\_lcd(0x00," lock on! ");

while(1)

{

if(key() == 13)

{

write\_lcd(0x00,"admin: ");

if(admin\_key() == 1)

{

lock = 0;

goto st;

}

goto lk;

}

}

}

}

return 0;

}

void timer0() interrupt 1 //中断函数

{

TH0=(65536-50000)/256;

TL0=(65536-50000)%256;

}

/\*1602控制文件\*/

#include<reg52.h>

#include"timer.h"

#define uchar unsigned char

#define uint unsigned char

sbit lcden=P3^7;

sbit lcdrw=P3^6;

sbit lcdrs=P3^4;

void write\_com(uchar com) //命令子函数command

{

lcdrs=0;//接受指令

P1=com; //给P0口送指令码

delay(5);

lcden=1; //E变为高电平

delay(5);//在此延迟期间，将指令送入

lcden=0; //E变为低电平，高脉冲结束

}

void write\_data(uchar date) //数据子函数data

{

lcdrs=1; //接受数据

P1=date; //给P0口送数据

delay(5);

lcden=1; //E变为高电平

delay(5); //在此延迟期间，将数据送入

lcden=0; //E变为低电平，高脉冲结束

}

void init() //初始函数

{

lcdrw=0;

lcden=0; //E的初始值为低电平，后面的程序给高电平从而满足E为高脉冲

write\_com(0x38); //写入显示模式指令

write\_com(0x0e); //显示是否打开以及光标的设置

write\_com(0x06); //地址指针的加减和整屏是否移动

write\_com(0x01); //清屏指令

write\_com(0x80+0x00); //显示字符的初始位置

}

void write\_lcd(uint offset,uchar tab[])

{

uint num;

write\_com(0x80+offset); //显示字符的初始位置

for(num=0;num<16;num++)

{

write\_data(tab[num]);

delay(20);

}

}

void disp\_lcd(uint offset,uchar c)

{

write\_com(0x80+offset); //显示字符的初始位置

write\_data(c);//些数据

}

/\*HC-SR04 \*/

#include"reg52.h"

#include"timer.h"

#include<intrins.h>

bit cuowu;

sbit trig=P3^2;

sbit echo=P3^3;

unsigned int s;

unsigned long sj;

void init\_sr04()

{

trig=0;

EA=1;

ET1=1;

ET0=1;

TMOD=0x10;

TH1=0;

TL1=0;

s=0;

TR1=0;

delay(1000);

}

unsigned int location()

{

init\_sr04();

trig=1;

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

trig=0;

while(!echo);

TR1=1;

while(echo);

TR1=0;

sj=TH1\*256+TL1;

s=sj\*340/2/1000;

return s;

}

int sr04()

{

int k;

k=location();

if(k<60)

{

return 1;

}

else

{

return 0;

}

}

void time1() interrupt 3

{

}

/\*键盘\*/

#include "reg52.h"

#include "SR04.h"

#include "timer.h"

short int checkkey(void)

{

unsigned char n;

P2=0xf0;

n=P2;

if(n!=0xf0)

return 1;

else

return 0;

}

int key()

{

int NO=0;

int sign=1;

unsigned char i,j;

while(sign)

{

if(sr04() == 1)//距离在6厘米内

{

return 100; // 返回标志信号

}

if(checkkey()==1)

{

delay(1);

if(checkkey()==1)

P2=0xf0;

i=P2;

P2=0x0f;

j=P2;

i=i|j;

switch(i)

{

case 0x7e:NO=0,sign=0;break;

case 0xbe:NO=1,sign=0;break;

case 0xde:NO=2,sign=0;break;

case 0xee:NO=3,sign=0;break;

case 0x7d:NO=4,sign=0;break;

case 0xbd:NO=5,sign=0;break;

case 0xdd:NO=6,sign=0;break;

case 0xed:NO=7,sign=0;break;

case 0x7b:NO=8,sign=0;break;

case 0xbb:NO=9,sign=0;break;

case 0xdb:NO=10,sign=0;break;

case 0xeb:NO=11,sign=0;break;

case 0x77:NO=12,sign=0;break;

case 0xb7:NO=13,sign=0;break;

case 0xd7:NO=14,sign=0;break;

case 0xe7:NO=15,sign=0;break;

}

while(checkkey()==1); //等待按键弹起

}

}

return NO;

}

/\*管理员模式\*/

#include"keyboard.h"

#include"1602.h"

#include"LED.h"

extern unsigned char number[];

extern unsigned int boxes[];

extern int admin;

int admin\_key() //管理员密码校验程序

{

int iter;

int i;

int temp[4];

int key\_temp;

/\* key\_temp=key();

disp\_lcd(0x00,number[key\_temp]);

disp\_lcd(0x01,number[admin]);

while(1);\*/ //测试

lp3:write\_lcd(0x40," ");

for( iter = 0 ; iter < 4 ; iter++ )

{

i = key();

if(i == 12)

{

return 0;

} else if(i > 9)

{

goto lp3;

}

temp[iter] = i;

disp\_lcd(0x40+iter,number[i]);

}

key\_temp = temp[0]\*1000+temp[1]\*100+temp[2]\*10+temp[0];

if(key\_temp == admin)

{

return 1;

}

else

{

return 0;

}

}

void admin\_control()//管理员控制

{ int num,x,i;

lp2:

num = 0;

write\_lcd(0x00,"please input num");

for(i=0 ; i<2 ; i++)

{

x = key();

if(x > 9)

{

goto lp2;

}

if(x== 12)

{

goto end;

}

disp\_lcd(0x40+i,number[x]);

if(i == 0)

{

num += x\*10;

}

else

{

num += x;

}

if(num>12)

{

write\_lcd(0x40," ");

num =0;

i = -1;

}

}

open\_led(num);

boxes[num]=0;

end: write\_com(0x01);

}

void admin\_resetkey()//管理员密码重置

{

int iter;

int i;

int temp[4];

write\_lcd(0x00,"reset\_adminkey: ");

lp3:write\_lcd(0x40," ");

for( iter = 0 ; iter < 4 ; iter++ )

{

i = key();

if(i > 9)

{

goto lp3;

}else if(i == 12)

{

goto over;

}

temp[iter] = i;

disp\_lcd(0x40+iter,number[i]);

}

admin = temp[0]\*1000+temp[1]\*100+temp[2]\*10+temp[0];

over:;

}

/\*取出物品\*/

#include"keyboard.h"

#include"1602.h"

#include"LED.h"

#include"timer.h"

extern unsigned int boxes[12];

extern unsigned char number[10];

extern lock;

void take2()

{

int err=0;

int iter = 0;

int temp[4] = {0};

int num[4]={0};

int i;

write\_com(0x01);

write\_lcd(0x00,"please input key");

num[0] = key();

i = num[0];

disp\_lcd(0x40,number[i]);

num[1] = key();

i = num[1];

disp\_lcd(0x41,number[i]);

num[2] = key();

i = num[2];

disp\_lcd(0x42,number[i]);

num[3] = key();

i = num[3];

disp\_lcd(0x43,number[i]);

for(iter = 0 ; iter<12 ; iter++)

{

temp[0] = boxes[iter]/1000;

temp[1] = (boxes[iter]%1000)/100;

temp[2] = (boxes[iter]%100)/10;

temp[3] = boxes[iter]%10;

if(temp[0] == num[0] && temp[1] == num[1] && temp[2] == num[2] && temp[3] == num[3] )

{

open\_led(iter);

boxes[iter]=0;

lock = 0;

goto end;

}

}

lock++;

end: ;

}

/\*存物品\*/

#include<reg52.h>

#include "1602.h"

#include "keyboard.h"

#include "LED.h"

#include "SR04.h"

#define uchar unsigned char

#define uint unsigned char

extern unsigned char number[10];

extern uint NO1,NO2;

extern unsigned int boxes[];

int save2()

{

uint temp;

uint iter;

uint userkey\_temp;

uint clock\_num[4];

int i;

for(iter = 0 ; iter<BNUM ; iter++)

{

if(boxes[iter] == 0)

{

restart:clock\_num[0] = NO1%10;

clock\_num[1] = NO1%9;

clock\_num[2] = NO2%10;

clock\_num[3] = NO2%9;

write\_com(0x01);

write\_lcd(0x00,"key: ");

temp =clock\_num[0];

disp\_lcd(0x40,number[temp]);

temp =clock\_num[1];

disp\_lcd(0x41,number[temp]);

temp =clock\_num[2];

disp\_lcd(0x42,number[temp]);

temp =clock\_num[3];

disp\_lcd(0x43,number[temp]);

userkey\_temp = clock\_num[0]\*1000+clock\_num[1]\*100+clock\_num[2]\*10+clock\_num[3];

for(i = 0 ; i<BNUM ; i++) //查询密码是否有相同的柜子

{

if(boxes[i] == userkey\_temp)

{

NO1 = TH0;

NO2 = TL0;

goto restart;

}

}

boxes[iter] = userkey\_temp;

select\_led(iter);

goto L;

}

}

L: while(1)

{

if(location()>80) //返回

{

return 1;

}

}

}

/\*延迟控制\*/

#define uint unsigned int

#define uchar unsigned char

void delay(uint z) //延时函数

{

uint x,y;

for(x=z;x>0;x--)

for(y=114;y>0;y--);

}