

**课程设计（论文）任务书**

电气与自动化工程**学院 专业** 电气工程及其自动化 **班级** 电牵2018-5

1. **课程设计（论文）题目 倒车雷达控制电路设计**
2. **课程设计（论文）工作自 2020**  **年 12**  **月 27**  **日起**

**至 12 月** **30** **日止**。

1. **课程设计（论文）的内容和要求**：

**1．程序设计的目的**

(1)综合运用单片机原理及应用相关课程的理论知识和实际应用知识，进行单片机应用系统电路及程序设计，从而使这些知识得到进一步的巩固，加深和发展； (2)熟悉和掌握单片机控制系统的设计方法，汇编语言程序设计及proteus软件的使用；

(3)通过查阅图书资料、以及书写程序设计报告进一步提高综合能力，培养分析问题的能力

**2.程序设计的内容及任务**

(1)测量车前、车后距离；

(2)在数码管上显示距离值；

(3)车前、车后的距离小于0.5m时，系统报警；

**3.课程设计说明书编写要求**

（1）设计说明书用专用稿纸书写或用B5纸打印，要求论述清晰，撰写整洁规范，引用资料应说明出处。插图应该标注图题图序，表格要有表题表序。

（2）说明书内容应包括（装订次序）：题目、目录、正文、设计总结、参考文献、附录等。

（3）报告内容应包括：系统功能确定；方案论证（方案对比，技术经济分析） ;整体设计论述 ;硬件设计（电路接线，元器件说明，硬件资源分配 ）;软件设计 （软件流程，编程思想，程序注释，）系统调试；结论（对设计结果进行总结，收获与体会）;附录（附录包括程序清单（必须加上合理的注释）和符合国标的所设计系统原理图的图纸）。

**学生签名：王子烨**

2020**年** 12 **月** 30 **日**

**课程设计（论文）评审意见**

|  |  |  |  |
| --- | --- | --- | --- |
| **平时表现**  **与态度（10%）** |  | 优（　）、良（　）、中（　）、一般（　）、差（　）； | **10%** |
| **设计成果（30%）** | **课题复杂度** | 优（　）、良（　）、中（　）、一般（　）、差（　）； | **10%** |
| **设计方案** | 优（　）、良（　）、中（　）、一般（　）、差（　）； | **10%** |
| **创新性和性能** | 优（　）、良（　）、中（　）、一般（　）、差（　）； | **10%** |
| **设计说明**  **书质量**  **（30%）** | **整洁度** | 优（　）、良（　）、中（　）、一般（　）、差（　）； | **10%** |
| **规范度** | 优（　）、良（　）、中（　）、一般（　）、差（　）； | **10%** |
| **图纸质量** | 优（　）、良（　）、中（　）、一般（　）、差（　）； | **10%** |
| **答辩**  **（30%）** | **问题分析能力** | 优（　）、良（　）、中（　）、一般（　）、差（　）； | **10%** |
| **回答正确度** | 优（　）、良（　）、中（　）、一般（　）、差（　）； | **10%** |
| **使用工具能力** | 优（　）、良（　）、中（　）、一般（　）、差（　）； | **10%** |

总评成绩：优（ ）、良（ ）、中（ ）、及格（ ）、不及格（ ）

评阅人 职称

年 月 日

**倒车雷达控制电路设计**

**摘要：**这次课设的主要目的自己动手做一个倒车雷达控制电路的设计，可以测量障碍物与车辆前、后距离，当距离小于0.5m时，系统自动报警。

汽车倒车中使用的倒车雷达防撞报警系统即是俗称的倒车雷达，在汽车倒车时，超声波倒车雷达采用超声波测距原理探测汽车尾部离障碍物的距离,是汽车泊车辅助装置。超声波测距是一种非接触式的检测方式。与其它传统的光学检测或者电磁学检测方法相比较,超声波测距不受光线、被测对象颜色等因素的干扰,尤其适用于被测物在烟雾、大量粉尘、电磁干扰或致毒等极端恶劣环境下的测量。倒车时，当汽车尾部探测到障碍物时，倒车雷达就实时动态显示离障碍物的距离，达到设定的安全警告值时，倒车雷达立即发出报警声，以警示驾驶员，辅助驾驶员安全倒车。现在大多数都配置有倒车雷达。倒车雷达电路种类较多，本文介绍基于单片机控制的倒车雷达系统。

此次的倒车雷达控制电路的设计，由超声波发射与回波接收电路、超声波电信号放大电路、单片机控制电路、显示电路和声音报警电路组成,根据超声波测距原理,实现倒车防撞预警。采用通用型单片机作为控制电路,方便功能扩展,充分利用单片机的内部资源,外围元件少、电路简洁、成本低、实用性强。

**关键词：**超声波；倒车雷达；单片机

**目录**

[一、 总体方案论述 6](#_Toc60224840)

[1.1 任务分析 6](#_Toc60224841)

[1.2 方案对比 6](#_Toc60224842)

[1.3 系统总设计 7](#_Toc60224843)

[二、 硬件设计 8](#_Toc60224844)

[2.1 超声波雷达电路 8](#_Toc60224845)

[2.2 报警器电路 10](#_Toc60224846)

[2.3 显示电路 12](#_Toc60224847)

[三、 软件设计 14](#_Toc60224848)

[3.1 软件流程框图 14](#_Toc60224849)

[四、 系统调试结果 18](#_Toc60224850)

[4.1 Keil软件调试 18](#_Toc60224851)

[4.2 Proteus软件仿真过程 18](#_Toc60224852)

[4.3 仿真结果 19](#_Toc60224853)

[五、 总结与体会 21](#_Toc60224854)

[5.1 心得体会 21](#_Toc60224855)

[参考文献 38](#_Toc60224856)

# 总体方案论述

### 任务分析

（1）测量障碍物与车辆之间的前、后距离。

（2）在数码管上显示距离值。

（3）当距离值小于0.5M时，

### 方案对比

在设计的过程中，我们产生了如下方案：

**显示模块的论证与选择：**

方案一：用数码管进行显示，外部电路包括三个数码管，三个CD4511芯片，以及一块74139芯片进行片选。

方案二：采用LCD1602进行显示，需要一个LCD1602显示屏，加一个变阻器和若干个电容。

通过比较两个方案，方案一器件较多，连线繁琐，方案二较方案一硬件电路简单。

**主控模块的论证与选择：**

方案一：采用STC12C5A60S2单片机。STC12C5A60S2单片机采用增强型8051内核，速度比普通8051快8到12倍，内部资源丰富，具有高速SPI串行通信接口、2路PWM输出和10位A/D转换器。

方案二：采用STC89C51单片机。STC89C521是一种低功耗、高性能CMOS8位微控制器，具有 8K 在系统可编程Flash 存储器。操作起来简单，便捷，且价格比较便宜。

通过比较两个方案，方案一芯片较贵，方案二的STC89C51的功能完全能满足本设计，因此综合考虑，选择方案二。

### 系统总设计

总体设想：用超声波雷达测量物体与障碍物的距离，然后将距离信息返回给单片机，单片机将此信息用LCD 1602显示出来。如果距离小于0.5M，单片机则控制蜂鸣器报警，当距离越小时，蜂鸣器响的时间越久。



图1 系统框图

# 硬件设计

## 超声波雷达电路

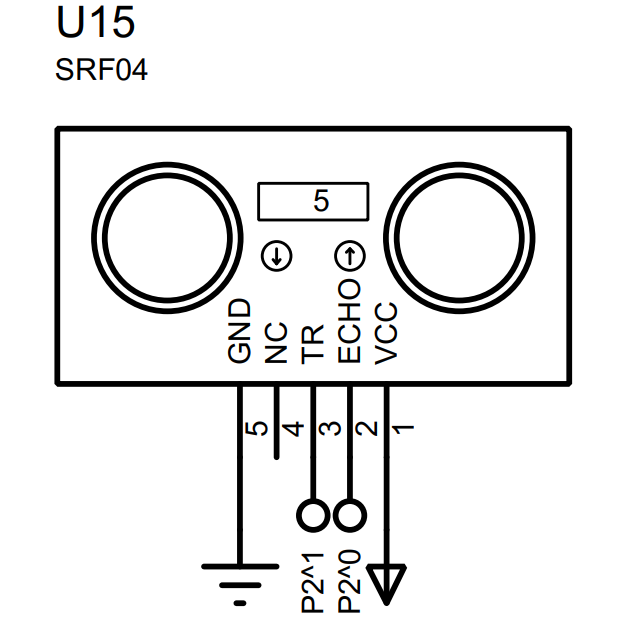


图2 超声波雷达模块仿真图

超声波模块有4个引脚，分别为Vcc、 Trig（控制端）、 Echo（接收端）、 GND；其中VCC、GND接上5V电源， Trig（控制端）控制发出的超声波信号，Echo（接收端）接收反射回来的超声波信号。模块如图：

图3 超声波实物图

控制原理：通过Trig引脚发一个 10US 以上的高电平,就可以在Echo接收口等待高电平输出；一有输出就可以开定时器计时,当此口变为低电平时就可以读定时器的值,此时就为此次测距的时间,方可算出距离.如此不断的周期测,就可以达到你移动测量的值了。

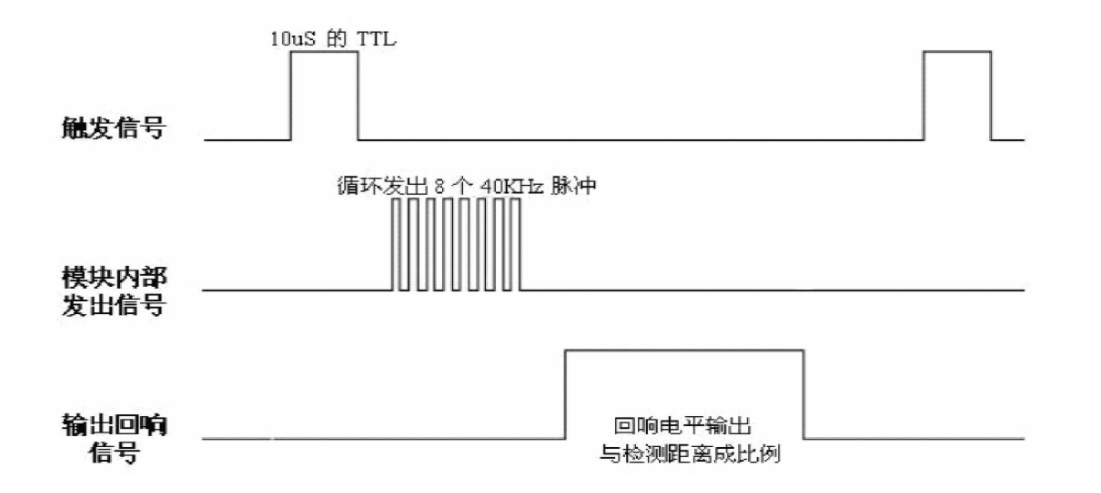


图4 超声波模块工作时序图

工作流程  
a.单片机引脚触发Trig测距，给至少 10us 的高电平信号;  
b.模块自动发送 8 个 40khz 的方波，自动检测是否有信号返回；  
c.有信号返回，通过 IO 输出一高电平，并单片机定时器计算高电平持续的时间;  
d.超声波从发射到返回的时间．  
计算公式：测试距离=(高电平时间\*声速(340M/S))/2;

空气中的声速是每秒340米。一微妙时间声波传输的距离是340×0.000001=0.00034米=0.034厘米，除以2就是所测离:0.017=1.7/100。 65.536ms=65536μs最大探测距离是65536×0.017=1114厘米=11.14米。

## 报警器电路

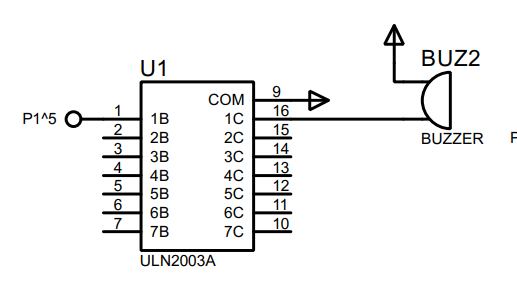


图5 报警器电路仿真图

报警器电路是由ULN2003芯片与蜂鸣器组成。

ULN2003是大电流驱动阵列,多用于单片机、智能仪表、PLC、数字量输出卡等控制电路中。可直接驱动继电器等负载。　ULN2003 是高压大电流达林顿晶体管阵列系列产品，具有电流增益高、工作电压高、温度范围宽、带负载能力强等特点，适应于各类要求高速大功率驱动的系统。

由于以上特点，将其作为驱动蜂鸣器的芯片。

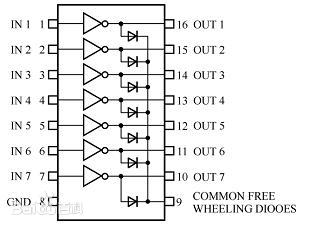
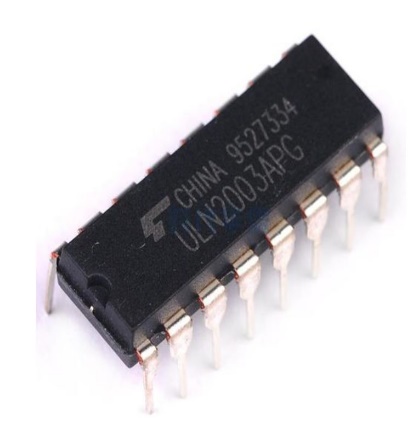


图6 芯片引脚图

图7 芯片实物图

ULN2003 引脚功能：

引脚 1至7：CPU 脉冲输入端，端口对应一个信号输出端。

引脚 8：接地。

引脚 9：该脚是内部 7 个续流二极管负极的公共端，各二极管的正极分别接 各达林顿管的集电极。用于感性负载时，该脚接负载电源正极，实现续流作用。如果 该脚接地，实际上就是达林顿管的集电极对地接通。

引脚 10至16：脉冲信号输出端，对应相应引脚信号输入端。

蜂鸣器选用的是有源蜂鸣器。有源蜂鸣器是靠压电效应的原理来发声的，压电材料，一般常见的是各种压电陶瓷。 这种材料的特别之处在于，当电压作用于压电材料时，就会随电压和频率的变化产生机械变形。另一方面，当振动压电陶瓷时，则会产生电荷。就是说这种材料能把机械变形和电荷相互转化，压电式蜂鸣器里面的起振片，就是一种压电陶瓷。如上所述，要让它振动，除了压电陶瓷本身，还需要适当大小和频率变化的电压作用于压电陶瓷。有源蜂鸣器内部带有多谐振荡器，可以产生 1.5—2.5kHZ 的电压信号。 由此有源蜂鸣器才能发声。

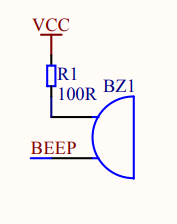
 

图9 蜂鸣器实物图

图8 蜂鸣器仿真图

通过P1^5引脚输入一个频率一定的方波，使蜂鸣器发出比较尖锐的声音，提醒驾驶员距离已经小于0.5m。通过改变方波的持续时间来控制声音的持续时间。当障碍物与车的距离越短，蜂鸣器的声音持续时间也越久。

## 显示电路

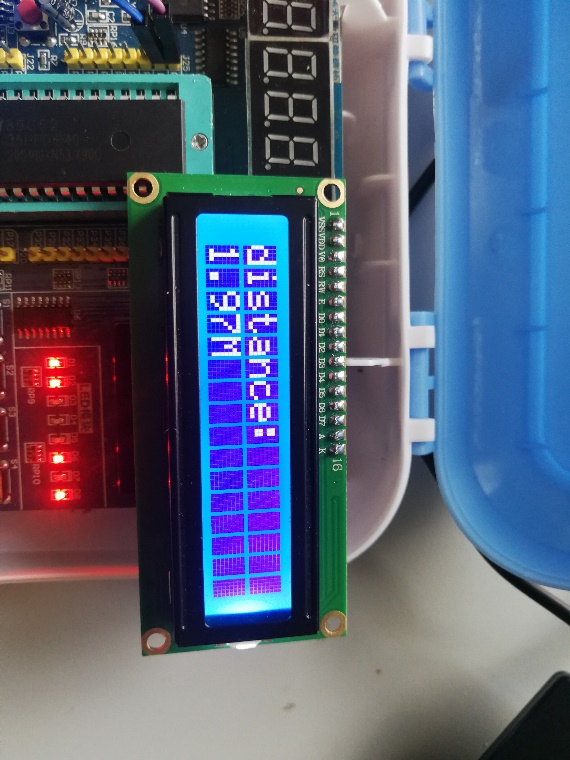


图10 lcd1602实物图

显示部分采用的是LCD1602，所谓的1602是指显示的时候，有2行内容每行有16个字符。1602的主要作用就是显示字母、数字、符号的。

LCD1602的8根数据线和3根控制线E，RS和R/W与单片机相连后即可正常工作。一般应用中只须往LCD1602中写入命令和数据，因此，可将LCD1602的R/W读/写选择控制端直接接地，这样可节省1根数据线。VO引脚是液晶对比度调试端，通常连接一个10kΩ的电位器即可实现对比度的调整；也可采用将一个适当大小的电阻从该引脚接地的方法进行调整，不过电阻的大小应通过调试决定。

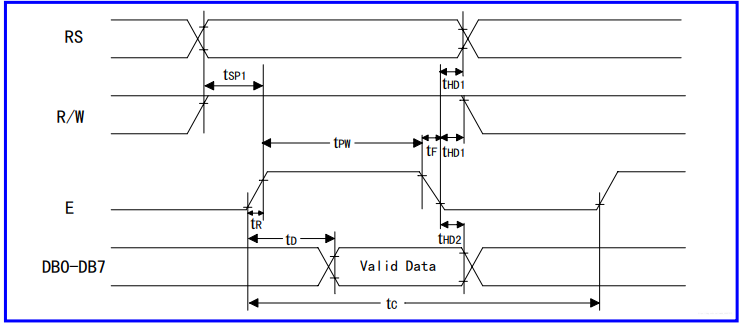


图11 lcd1602读操作时序图

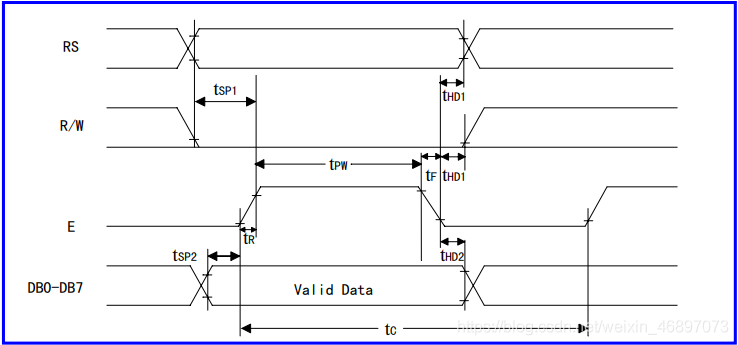


图12 lcd1602写操作时序图

# 软件设计

## 软件流程框图



图13 程序流程框图

**主函数：**

void main(void)

{

TMOD=0x01; //设T0为方式1，GATE=1；

TH0=0;

TL0=0;

ET0=1; //允许T0中断

EA=1; //开启总中断

LcdInit();

LcdShowStr(0,0,PuZh);

while(1)

{

StartModule();

while(!Echo); //当RX为零时等待

TR0=1; //开启计数

while(Echo); //当RX为1计数并等待

TR0=0; //关闭计数

Conut(); //计算

if(S<50)

voice();

delayms(80);

}

}

**计数函数：**

void Conut(void)

{

time=TH0\*256+TL0;

TH0=0;

TL0=0;

S=(time\*1.7)/100; //算出来是CM

if((S>=700)||flag==1) //超出测量范围显示“-”

{

flag=0;

DisplayOneChar(0, 1, ASCII[11]);

DisplayOneChar(1, 1, ASCII[10]); //显示点

DisplayOneChar(2, 1, ASCII[11]);

DisplayOneChar(3, 1, ASCII[11]);

DisplayOneChar(4, 1, ASCII[12]); //显示M

}

else

{

disbuff[0]=S%1000/100;

disbuff[1]=S%1000%100/10;

disbuff[2]=S%1000%10 %10;

DisplayOneChar(0, 1, ASCII[disbuff[0]]);

DisplayOneChar(1, 1, ASCII[10]); //显示点

DisplayOneChar(2, 1, ASCII[disbuff[1]]);

DisplayOneChar(3, 1, ASCII[disbuff[2]]);

DisplayOneChar(4, 1, ASCII[12]); //显示M

}

}

**LCD显示控制：**

//按指定位置显示一个字符

void DisplayOneChar(unsigned char X, unsigned char Y, unsigned char DData)

{

Y &= 0x1;

X &= 0xF; //限制X不能大于15，Y不能大于1

if (Y) X |= 0x40; //当要显示第二行时地址码+0x40;

X |= 0x80; //算出指令码

LcdWriteCom(X); //发命令字

LcdWriteData(DData); //发数据

}

void LcdShowStr(unsigned char x,unsigned char y,unsigned char \*str) //显示字符串

{

LcdSetCursor(x,y); //当前字符的坐标

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

{

LcdWriteData(\*str++);

}

}

# 系统调试结果

## Keil软件调试

程序按照模块进行调试，先将每个模块的程序编辑出来，然后再进行调试，如果各个模块都没有错误时，再进行整合，最后再总体进行调试。

## Proteus软件仿真过程

1. 元器件清单（包含实物所用元器件）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 名称 | 型号 | 数目/个 | 用途 |
| 1 | 单片机 | AT89C51 | 2 | 控制核心 |
| 2 | 晶振 | 11.0592MHZ | 2 | 晶振电路 |
| 3 | 电容 | 22pF | 4 | 晶振电路 |
| 4 | 电解电容 | 10μF | 2 | 复位电路 |
| 5 | 排阻 | 10K | 2 | 上拉电阻 |
| 6 | 电阻 | 500欧 | 2 | 分压 |
| 7 | 数码管 | 七段共阴 | 2 | 显示 |
| 8 | Led灯 | Led | 4 | 显示 |
| 9 | 按键 | Button | 2 | 控制 |
| 11 | 底座 | 单片机底座 | 1 | 安装芯片 |

表1 元器件清单表

1. 根据原理图安置芯片及连线。原理图见附录B。
2. 接入代码：在Keil 运行代码，代码运行无误会生成HEX文件，然后再proteus上双基单片机模型，会弹出一个如下对话框。点击文件按钮，选择所需要的程序文件（. HEX）。选择合适的工作频率即可确认,然后运行调试。

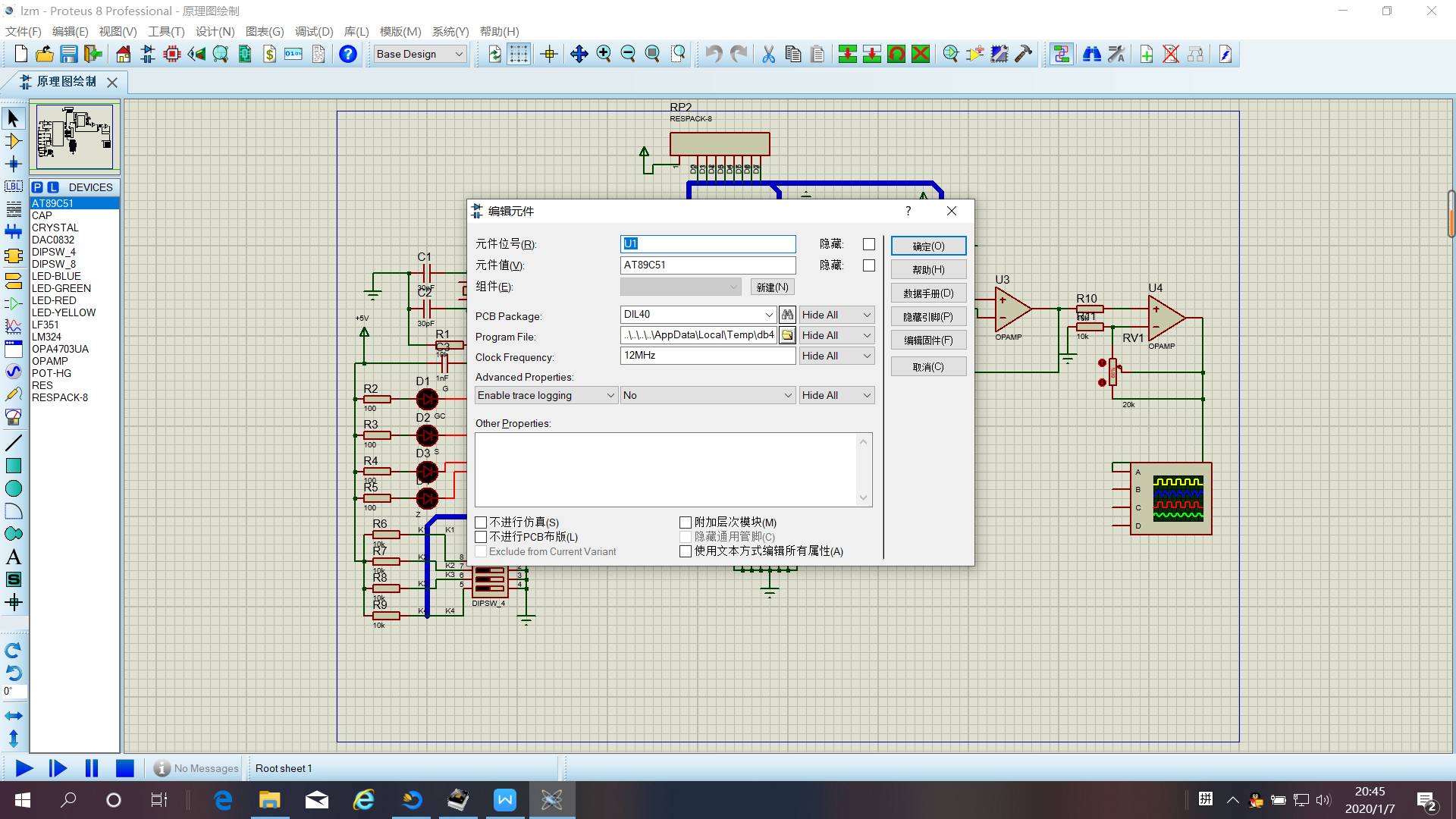


图14 程序载入图

## 仿真结果

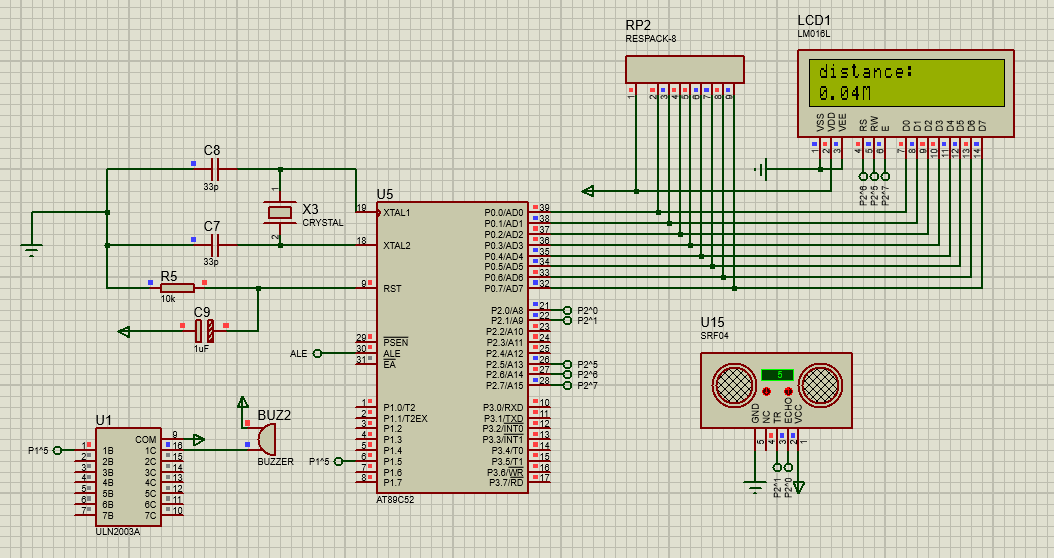


图15 总体仿真图

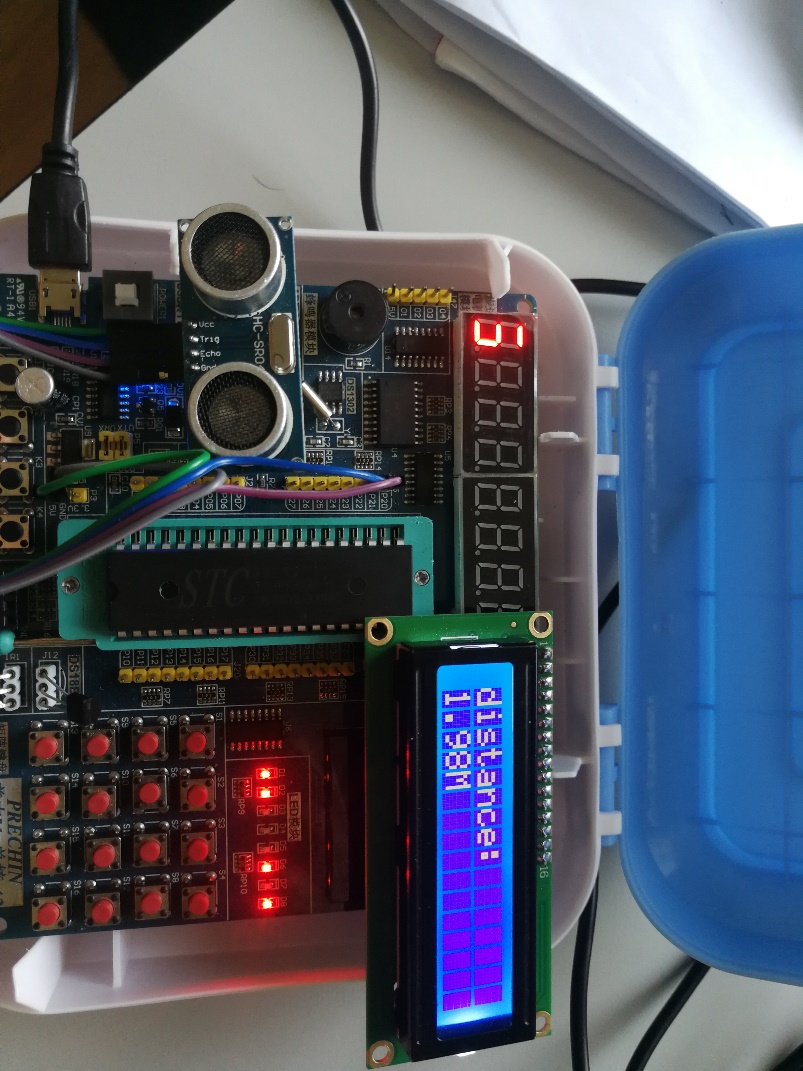
****

图16 总体实物图

# 总结与体会

## 心得体会

作为一名电气专业的大三学生，我觉得做单片机课程设计是十分有意义的，而且是十分必要的。在已度过的大学时间里，我们大多数接触的是专业课。我们在课堂上掌握的仅仅是专业课的理论知识，如何去锻炼我们的实践能力?如何把我们所学的专业基础课理论知识运用到实践中去呢?我想做类似的课程设计就为我们提供了良好的实践平台。

在做本次课程设计的过程中，我感触最深的当属查阅大量的设计资料了。为了让自己的设计更加完善，查阅这方面的设计资料是十分必要的，同时也是必不可少的。我们是在做单片机课程设计，但我们不是艺术家，他们可以抛开实际尽情在幻想的世界里翱翔，而我们一切都要有据可依，有理可寻，不切实际的构想永远只能是构想，永远无法升级为设计。

其次，在这次课程设计中，我们运用到了以前所学的专业课知识，如：CAD制图、汇编语言、模拟和数字电路知识等。虽然过去从未独立应用过它们，但在学习的过程中带着问题去学我发现效率很高，这是我做这次课程设计的又一收获。后，要做好一个课程设计，就必须做到：在设计程序之前，对所用单片机的内部结构有一个系统的了解，知道该单片机内有哪些资源;要有一个清晰的思路和一个完整的的软件流程图;在设计程序时，不能妄想一次就将整个程序设计好，反复修改、不断改进是程序设计的必经之路;

要养成注释程序的好习惯，一个程序的完美与否不仅仅是实现功能，而应该让人一看就能明白你的思路，这样也为资料的保存和交流提供了方便;在设计课程过程中遇到问题是很正常德，但我们应该将每次遇到的问题记录下来，并分析清楚，以免下次再碰到同样的问题的课程设计结束了，但是从中学到的知识会让我受益终身。

发现、提出、分析、解决问题和实践能力的提高都会受益于我在以后的学习、工作和生活中。设计过程，好比是我们人类成长的历程，常有一些不如意，但毕竟这是第一次做，难免会遇到各种各样的问题。在设计的过程中发现了自己的不足之处，对以前所学过的知识理解得不够深刻，掌握得不够牢固。

通过这次课程设计我也发现了自身存在的不足之处，虽然感觉理论上已经掌握，但在运用到实践的过程中仍有意想不到的困惑，经过一番努力才得以解决。这也激发了我今后努力学习的兴趣，我想这将对我以后的学习产生积极的影响。

通过这次设计，我懂得了学习的重要性，了解到理论知识与实践相结合的重要意义，学会了坚持、耐心和努力，这将为自己今后的学习和工作做出了最好的榜样。我觉得作为一名自动化专业的学生，单片机的课程设计是很有意义的。更重要的是如何把自己平时所学的东西应用到实际中。

虽然自己对于这门课懂的并不多，很多基础的东西都还没有很好的掌握，觉得很难，也没有很有效的办法通过自身去理解，但是靠着这一个多礼拜的学习，在同学的帮助和讲解下，渐渐对这门课逐渐产生了些许的兴趣，自己开始主动学习并逐步从基础慢慢开始弄懂它。

我认为这个收获应该说是相当大的。觉得课程设计反映的是一个从理论到实际应用的过程，但是更远一点可以联系到以后毕业之后从学校转到踏上社会的一个过程。小组人员的配合﹑相处，以及自身的动脑和努力，都是以后工作中需要的。

# 参考文献

* 1. 徐春辉，陈忠斌，章海亮.单片微型计算机原理及应用.北京：电子工业出版社，2017.8
  2. 蔡明文，冯先成.单片机课程设计.武汉：华中科技大学出版社，2007.3
  3. 杨居义.单片机课程设计指导.北京：清华大学出版社，2009.9
  4. 林立，张俊亮.单片机原理及应用：基于Proteus和Keil C.北京：电子工业出版社，2013
  5. 楼然苗，李光飞.单片机课程设计指导.北京：北京航空航天大学出版社，2012.1
  6. 周润景，张丽娜，丁莉.基于PROTEUS的电路及单片机设计与仿真.北京：北京航空航天大学出版社，2009.12

**附录**

#include<reg51.h>

#include<intrins.h>

#include"lcd.h"

sbit Trig = P2^1;

sbit Echo = P2^0;

sbit beep=P1^5;

unsigned char PuZh[]="distance: ";

unsigned char code ASCII[15] = {'0','1','2','3','4','5','6','7','8','9','.','-','M'};

static unsigned char DisNum = 0; //显示用指针

unsigned int time=0;

unsigned long S=0;

bit flag =0;

unsigned char disbuff[4] ={ 0,0,0,0,};

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

\* 函 数 名 : main

\* 函数功能 : 主函数

\* 输 入 : 无

\* 输 出 : 无

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

void Conut(void)

{

time=TH0\*256+TL0;

TH0=0;

TL0=0;

S=(time\*1.7)/100; //算出来是CM

if((S>=700)||flag==1) //超出测量范围显示“-”

{

flag=0;

DisplayOneChar(0, 1, ASCII[11]);

DisplayOneChar(1, 1, ASCII[10]); //显示点

DisplayOneChar(2, 1, ASCII[11]);

DisplayOneChar(3, 1, ASCII[11]);

DisplayOneChar(4, 1, ASCII[12]); //显示M

}

else

{

disbuff[0]=S%1000/100;

disbuff[1]=S%1000%100/10;

disbuff[2]=S%1000%10 %10;

DisplayOneChar(0, 1, ASCII[disbuff[0]]);

DisplayOneChar(1, 1, ASCII[10]); //显示点

DisplayOneChar(2, 1, ASCII[disbuff[1]]);

DisplayOneChar(3, 1, ASCII[disbuff[2]]);

DisplayOneChar(4, 1, ASCII[12]); //显示M

}

}

void zd0() interrupt 1 //T0中断用来计数器溢出,超过测距范围

{

flag=1; //中断溢出标志

}

void StartModule() //启动模块

{

Trig=1; //启动一次模块

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

Trig=0;

}

void delayms(unsigned int ms)

{

unsigned char i=100,j;

for(;ms;ms--)

{

while(--i)

{

j=10;

while(--j);

}

}

}

void delay(unsigned int i)

{

while(i--);

}

void voice()

{

unsigned int i;

i=(50-S)\*100;

while(i--)

{

beep=~beep;

delay(50);

}

}

void main(void)

{

TMOD=0x01; //设T0为方式1，GATE=1；

TH0=0;

TL0=0;

ET0=1; //允许T0中断

EA=1; //开启总中断

LcdInit();

LcdShowStr(0,0,PuZh);

while(1)

{

StartModule();

while(!Echo); //当RX为零时等待

TR0=1; //开启计数

while(Echo); //当RX为1计数并等待

TR0=0; //关闭计数

Conut(); //计算

if(S<50)

voice();

delayms(80);

}

}

#include"lcd.h"

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

\* 函 数 名 : Lcd1602\_Delay1ms

\* 函数功能 : 延时函数，延时1ms

\* 输 入 : c

\* 输 出 : 无

\* 说 名 : 该函数是在12MHZ晶振下，12分频单片机的延时。

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

void Lcd1602\_Delay1ms(uint c) //误差 0us

{

uchar a,b;

for (; c>0; c--)

{

for (b=199;b>0;b--)

{

for(a=1;a>0;a--);

}

}

}

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

\* 函 数 名 : LcdWriteCom

\* 函数功能 : 向LCD写入一个字节的命令

\* 输 入 : com

\* 输 出 : 无

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

#ifndef LCD1602\_4PINS //当没有定义这个LCD1602\_4PINS时

void LcdWriteCom(uchar com) //写入命令

{

LCD1602\_E = 0; //使能

LCD1602\_RS = 0; //选择发送命令

LCD1602\_RW = 0; //选择写入

LCD1602\_DATAPINS = com; //放入命令

Lcd1602\_Delay1ms(1); //等待数据稳定

LCD1602\_E = 1; //写入时序

Lcd1602\_Delay1ms(5); //保持时间

LCD1602\_E = 0;

}

#else

void LcdWriteCom(uchar com) //写入命令

{

LCD1602\_E = 0; //使能清零

LCD1602\_RS = 0; //选择写入命令

LCD1602\_RW = 0; //选择写入

LCD1602\_DATAPINS = com; //由于4位的接线是接到P0口的高四位，所以传送高四位不用改

Lcd1602\_Delay1ms(1);

LCD1602\_E = 1; //写入时序

Lcd1602\_Delay1ms(5);

LCD1602\_E = 0;

// Lcd1602\_Delay1ms(1);

LCD1602\_DATAPINS = com << 4; //发送低四位

Lcd1602\_Delay1ms(1);

LCD1602\_E = 1; //写入时序

Lcd1602\_Delay1ms(5);

LCD1602\_E = 0;

}

#endif

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

\* 函 数 名 : LcdWriteData

\* 函数功能 : 向LCD写入一个字节的数据

\* 输 入 : dat

\* 输 出 : 无

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

#ifndef LCD1602\_4PINS

void LcdWriteData(uchar dat) //写入数据

{

LCD1602\_E = 0; //使能清零

LCD1602\_RS = 1; //选择输入数据

LCD1602\_RW = 0; //选择写入

LCD1602\_DATAPINS = dat; //写入数据

Lcd1602\_Delay1ms(1);

LCD1602\_E = 1; //写入时序

Lcd1602\_Delay1ms(5); //保持时间

LCD1602\_E = 0;

}

#else

void LcdWriteData(uchar dat) //写入数据

{

LCD1602\_E = 0; //使能清零

LCD1602\_RS = 1; //选择写入数据

LCD1602\_RW = 0; //选择写入

LCD1602\_DATAPINS = dat; //由于4位的接线是接到P0口的高四位，所以传送高四位不用改

Lcd1602\_Delay1ms(1);

LCD1602\_E = 1; //写入时序

Lcd1602\_Delay1ms(5);

LCD1602\_E = 0;

LCD1602\_DATAPINS = dat << 4; //写入低四位

Lcd1602\_Delay1ms(1);

LCD1602\_E = 1; //写入时序

Lcd1602\_Delay1ms(5);

LCD1602\_E = 0;

}

#endif

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

\* 函 数 名 : LcdInit()

\* 函数功能 : 初始化LCD屏

\* 输 入 : 无

\* 输 出 : 无

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

#ifndef LCD1602\_4PINS

void LcdInit() //LCD初始化子程序

{

LcdWriteCom(0x38); //开显示

LcdWriteCom(0x0c); //开显示不显示光标

LcdWriteCom(0x06); //写一个指针加1

LcdWriteCom(0x01); //清屏

LcdWriteCom(0x80); //设置数据指针起点

}

#else

void LcdInit() //LCD初始化子程序

{

LcdWriteCom(0x32); //将8位总线转为4位总线

LcdWriteCom(0x28); //在四位线下的初始化

LcdWriteCom(0x0c); //开显示不显示光标

LcdWriteCom(0x06); //写一个指针加1

LcdWriteCom(0x01); //清屏

LcdWriteCom(0x80); //设置数据指针起点

}

#endif

void LcdSetCursor(unsigned char x,unsigned char y) //坐标显示

{

unsigned char addr;

if(y == 0)

addr = 0x00 + x;

else

addr = 0x40 + x;

LcdWriteCom(addr|0x80);

}

//按指定位置显示一个字符

void DisplayOneChar(unsigned char X, unsigned char Y, unsigned char DData)

{

Y &= 0x1;

X &= 0xF; //限制X不能大于15，Y不能大于1

if (Y) X |= 0x40; //当要显示第二行时地址码+0x40;

X |= 0x80; //算出指令码

LcdWriteCom(X); //发命令字

LcdWriteData(DData); //发数据

}

void LcdShowStr(unsigned char x,unsigned char y,unsigned char \*str) //显示字符串

{

LcdSetCursor(x,y); //当前字符的坐标

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

{

LcdWriteData(\*str++);

}

}

#ifndef \_\_LCD\_H\_

#define \_\_LCD\_H\_

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

当使用的是4位数据传输的时候定义，

使用8位取消这个定义

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

//#define LCD1602\_4PINS

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

包含头文件

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

#include<reg51.h>

//---重定义关键词---//

#ifndef uchar

#define uchar unsigned char

#endif

#ifndef uint

#define uint unsigned int

#endif

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

PIN口定义

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

#define LCD1602\_DATAPINS P0

sbit LCD1602\_E=P2^7;

sbit LCD1602\_RW=P2^5;

sbit LCD1602\_RS=P2^6;

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

函数声明

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

/\*在51单片机12MHZ时钟下的延时函数\*/

void Lcd1602\_Delay1ms(uint c); //误差 0us

/\*LCD1602写入8位命令子函数\*/

void LcdWriteCom(uchar com);

/\*LCD1602写入8位数据子函数\*/

void LcdWriteData(uchar dat) ;

/\*LCD1602初始化子程序\*/

void LcdInit();

void LcdSetCursor(unsigned char x,unsigned char y); //坐标显示

void LcdShowStr(unsigned char x,unsigned char y,unsigned char \*str); //显示字符串

void DisplayOneChar(unsigned char X, unsigned char Y, unsigned char DData);

#end