**重庆三峡学院**

**毕业设计（论文）**

|  |  |  |  |
| --- | --- | --- | --- |
| **题目** | **智能电子秤的设计与实现** | | |
| **学院** | **电子与信息工程** | | |
| **专业** | **电子信息工程** | | |
| **年级** | **2016级电子信息一班** | | |
| **姓名** | **肖增兵** | | |
| **学号** | **201607014103** | | |
| **指导教师** | **孙光壮** | **职称** | **讲师** |

**完成毕业设计（论文）时间 年 月**

目录

[摘要 ........................................................................ I](#_bookmark0)

[Abstract.................................................................... II](#_bookmark1)

[1 绪论 ....................................................................... 1](#_bookmark2)

[1.1 选题的背景及意义 ..................................................... 1](#_bookmark3)

[1.2 本次设计需要完成的内容 ............................................... 1](#_bookmark4)

[2 总设计原理及方案 ........................................................... 3](#_bookmark5)

[2.1 智能电子秤的设计要求 ................................................. 3](#_bookmark6)

[2.2 智能电子秤的工作原理 ................................................. 3](#_bookmark7)

[2.3 智能电子秤原理框图 ................................................... 3](#_bookmark8)

[3 硬件电路设计 ............................................................... 5](#_bookmark9)

[3.1 硬件控制电路模块 ..................................................... 5](#_bookmark10)

[3.2 单片机模块 ........................................................... 5](#_bookmark11)

[3.3 模数（A/D）转换模块 .................................................. 6](#_bookmark12)

[3.4 称重传感器模块 ....................................................... 8](#_bookmark13)

[3.4.1 电阻应变片的结构及工作原理 ..................................... 8](#_bookmark14)

[3.4.2 测量转换电路 ................................................... 9](#_bookmark15)

[3.5 LCD 显示模块 ........................................................ 10](#_bookmark16)

[3.6 超重报警模块 ........................................................ 11](#_bookmark17)

[3.7 复位电路 ............................................................ 11](#_bookmark18)

[3.8 按键处理模块 ........................................................ 12](#_bookmark19)

[4 软件设计 .................................................................. 14](#_bookmark20)

[4.1 软件开发环境 ........................................................ 14](#_bookmark21)

[4.2 主程序流程图 ........................................................ 14](#_bookmark22)

[4.3 显示部分流程图 ...................................................... 15](#_bookmark23)

[4.4 系统按键部分流程图 .................................................. 15](#_bookmark24)

[5 系统调试与调试结果 ........................................................ 17](#_bookmark25)

[5.1 硬件电路的制作 ...................................................... 17](#_bookmark26)

[5.2 硬件电路的调试和结果 ................................................ 17](#_bookmark27)

[5.2.1 硬件电路的调试 .................................................... 17](#_bookmark28)

[5.2.2 硬件电路的调试结果 ................................................ 18](#_bookmark29)

[设计总结.................................................................... 19](#_bookmark30)

[参考文献.................................................................... 20](#_bookmark31)

[致谢 ....................................................................... 21](#_bookmark32)

[附 录 ....................................................................... 22](#_bookmark33) [附录一 智能电子秤原理图 ................................................. 22](#_bookmark34)

[附录二 源程序 ........................................................... 23](#_bookmark35)

智能电子秤设计

# 肖增兵

重庆三峡学院电子与信息工程学院电子信息工程专业 2016 级 重庆万州 404100

摘要：本文设计了以 STC89C52 单片机为控制核心的智能电子秤，它主要由 STC89C52 单片机、HX711 模数转换器、LCD12864 显示器、压力传感器四大部分组成，系统安装了矩阵键盘，用户可以通过矩阵键盘输入项目的价格，系统可以计算总价格的总价格，并实时显示路线的重量和单价。为了防止长时间超负荷负载系统产生压力传感器，当系统超重10公斤时，要添加“超重提示机制”，系统为用户超重发送提示，并减轻重量。

关键词：STC89C52，压力传感器，矩阵键盘，模数转换器，液晶显示

I

**Design of Portable intelligent electronic scale Xiao zeng-bing**

Chongqing Three Gorges University School of Electronics and Information Engineering Electrical Electronic Information Engineering Professional 2014 Chongqing Wanzhou 404100

**Abstract**：With the development of science and technology, electronic devices are becoming

more and more intelligent, and electronic scale is an important tool to weigh things in our life.This paper designed the STC89C52 single-chip microcomputer as the control core of the intelligent electronic scale, it is mainly composed of STC89C52 SCM, HX711 AD converter, LCD12864 display four major part, pressure sensor, the system installed the matrix keyboard, the user can through the matrix keyboard input of the items, system can calculate the total price of the total price, and real-time display route and the weight of the unit price.In order to prevent pressure sensors from being generated by the system with long-term overload, when the system is 10 kg overweight, the "overweight warning mechanism" should be added. The system will send the warning for the user to be overweight and reduce the weight.

**Key words:** STC89C52 , pressure sensor, analog to digital converter， matrix keyboard liquid crystal display

II

# 1 绪论

传统的机械秤具有许多缺点，例如精度低，结构复杂，易老化和成本高。随着社会的发展，对秤的市场需求越来越高，尤其是人体秤，厨房秤和其他便携式小秤。与传统机械秤相比，电子秤具有许多优势。它用压力传感器代替了机械秤的弹簧，从而极大地减小了秤的尺寸和制造难度。传统的表盘被LCD或LED显示屏取代，使外观更加美观。由于集成了微控制器和软件系统，电子秤也具有传统机械秤无与伦比的智能。他可以完成许多功能，例如过载警报，总价计算，数据通信等。

当前市场上的称重工具要么结构复杂，要么操作不可靠，并且成本高昂，并且总体水平不高。一些小企业质量低劣，技术薄弱，设备不齐全，缺乏产品开发能力，产品质量低下徘徊。因此，开发一套具有实用价值的电子秤系统，从技术上克服上述许多缺点，改善电子秤的应用缺陷，具有重要的现实意义。

* 1. 选题的背景及意义

随着社会经济的飞速发展，商品贸易的种类和数量日益增加。长期以来，人们一直对使用传统的机械称重工具称量物品感到不满意。因此，随着科学技术的发展，我们已经开发出了性能更好，秤盘更方便的量具。中国电子秤的研发始于1960年代。原始的电子秤通过机电一体化实现了称重功能。后来，经过数十年的不断研究与开发和改进，伴随着计算机技术和电子技术的发展，电子秤的发展迅速，电子秤朝着数字化，全电子化，智能化，多功能化和高精度化的方向发展。电子秤的称重方法已经从模拟测量逐渐发展为数字测量。由于电子称重工具具有优于传统称重工具的巨大优势，因此传统的机械称重工具逐渐被电子称重工具所取代。在过去的几十年中，中国在电子称重设备的研究和发展方面发展迅速，但就目前而言，中国电子称重仪表行业的发展水平仍远未达到世界一流水平，差距还体现在研发能力弱，制造技术和工艺相对落后，生产设备和测试仪器老化，产品稳定性差，功能不足，环境适应性弱，产品类型不足，智能低，测量精度不高，工作可靠性高。较差的。随着经济的快速发展，商品交易数量急剧增加。我们的对称称重工具需要易于操作，易于识别，可携带，直观显示，方便称重，使用简单，高精度和高度智能化。随着电子技术和数字技术的发展，电子称重工具因其精度更高，智能程度更高，使用更方便快捷，性能可靠，运行稳定，对环境适应性强等优点而被广泛使用。在人们的生活中。它不仅极大地促进了人们的生产生活，而且大大提高了人们的称重效率和商品交易效率。这也促进了社会经济的发展。

# 本次设计需要完成的内容

根据设计要求，智能电子秤的设计需要完成以下工作：首先，明确设计要求，根据设计要求，上网查询相关书籍，收集相关信息，阐明便携式智能电子秤的基本原理。秤，使用知识，结合互联网上收集的书籍和相关信息，缩小设计思路，构建设计框架，设计智能电子秤的示意图，并使用Altium Designer软件进行布局。然后确定组件的类型和数量。根据便携式智能电子秤的要求，实现各种功能，查询相关信息，确定所需的组件，列出电子组件，在电子商务网站上购买相关组件，并使用学到的知识参考相关材料，根据智能电子秤各部分的功能，请使用keil uvision4软件来完成系统软件的编译，校正，调试和刻录。最后，根据设计的示意图，将电子组件合理地布置在铜复合板上并进行焊接。使用万用表检查组件的功能是否正常以及每个电路中是否存在短路或断路。已使用原始电子秤软件和硬件系统正确测试。之后，使用电池为硬件电路供电，并对软件和硬件进行联合调试。在调试过程中，发现问题，并分析和分析问题的原因，直到智能电子秤的所有部分正常工作并符合设计要求为止。

# 总设计原理及方案

* 1. 智能电子秤的设计要求

通过上面每个模块的介绍，我们最终选择STC89C52作为本设计的主控制芯片，使用应变计压力传感器和HX711模块收集重量信息，转换为实际重量并由LCD12864实时显示，并增加了4个\* 4矩阵键盘，用户可以通过键盘输入单价，系统会根据单价和重量自动计算总价，还可以执行剥皮功能。 如果称量的重量超过10Kg的范围，显示屏将显示“ overweight”（超重），并且板上的指示灯将点亮以提示。本次设计实现的主要功能如下：

1）电子秤的量程范围为 0~10kg；

2）去皮功能；

3）测量精度为 0.1%，测量误差为±0.010kg；

4）超出量程最大值 10kg 时光报警功能；

5) 重量显示功能。

# 智能电子秤的工作原理

根据本设计的要求，进行以下设计：本设计使用STC89C52 MCU作为整个系统的核心控制单元，按下电源开关，然后通过独立的按钮设置电子秤的量程（0〜10kg） 。当它在电子秤的称重平台上时，机械变形会导致电阻发生变化，从而产生压电效应。利用这一原理，称重传感器可以将称量物品的重量转换成可以用相关仪器测量的电压变化。电压信号进入放大电路，经过放大，滤波，然后输入到模块（A / D）。转换器通过模数（A / D）转换器将其从模拟信号转换为数字信号，然后将其发送到单片机进行控制和解码。最后，被称量物品的重量以数字形式显示在LCD上。另外，当被称量物品的重量在（0〜10kg）范围内时，智能电子秤正常工作，超重报警系统不工作，即LED不闪烁；当被称量物品的重量超过电子秤的最大范围时。当该值是10kg时，超重警报系统起作用，即LED闪烁。

# 智能电子秤原理框图

智能电子秤的基本原理是：将物品放在称重平台上时，称重传感器将物品的重量转换为电压或电流的变化，可以通过相关仪器进行测量。 通过放大器电路放大电压或电流后，HX711模块1（A / D）将其从模拟量转换为数字量，然后输入到单片机进行处理。 如果物品的重量大于电子秤的最大范围10kg，单片机将发出指令使超重报警系统发出报警； 如果物品的重量在电子秤的范围内（0〜10kg），则LCD会显示物品的重量。 根据上述智能电子秤的一般原理，结合所学知识并参考相关资料，设计了以下框图所示的方案。



图 2-1 智能电子秤原理图

# 硬件电路设计

* 1. 硬件控制电路模块

本次设计的便携式智能电子秤共有称重传感器、AD 转换、单片机处理、超重报警、按键处理、LCD 显示六大模块。

# 单片机模块

根据设计要求，本设计选用STC89C52单片机作为核心控制单元。STC89C52单片机是[STC](https://baike.baidu.com/item/STC)公司生产的单片机。在单片机的生产过程中，采用了CMOS工艺和高密度、高速技术。因此，SYC89C52单片机具有低功耗的特点。STC89C52具有外部中断、定时器、时钟输出和计数器等功能。

（一）STC89C52 单片机组成部分：

1、8 位中央处理单元。

2、512 字节的内部数据存储器

3、8k 片内程序存储器。

4、3 个 16 位定时/计数器。

5、32 个双向输入/输出（I/O）口。

6、5 个两级中断结构。

7、1 个全双工串行通信口。

8、时钟振荡电路。

（二）89C52 单片机引脚如图 3-1 所示：



图 3-1 STC89C52 单片机引脚图

表 3-1 STC89C52 单片机引脚功能表

|  |  |
| --- | --- |
| 引脚符号 | 引脚功能说明 |
| VCC | 主电源输入引脚，接﹢5V 直流电源 |
| GND | 接地端 |
| RST | 复位输入引脚，保持两个机器周期的高电平时动作 |
| P0 口 | 共有 8 位双向 I/O 口，可作为低 8 位地址/数据复用 |
| P1 口 | 共有 8 位双向 I/O 口。作为输入口使用时，对 pP1 口写入“1” |
| P2 口 | 8 位双向 I/O 口，其内部接有上拉电阻。可作为高 8 位地址/数据复用 |
| P3 口 | 共有 8 位双向 I/O 口，其内部有上拉电阻。此端口可用作特殊功能 |
| XTAL1 | 反向振荡放大器的输入及内部时钟工作电路的输入端 |
| XTAL2 | 反向振荡输出端 |
| PSEN | 外部程序存储选通信号输出端 |
| EA/VPP | 外部程序访问允许端 |

# （A/D）转换模块

A/D转换部分是整个设计的关键。如果这部分处理不好，整个设计就毫无意义。目前，世界上的adc有多种类型，包括传统的并行、逐次逼近和积分型adc，以及近年来新开发的sigma-delta和流水线adc。每种ADC都有各自的优缺点，可以满足不同的具体应用要求。A/D 转换的性能指标有：量化误差、分辨率、转换速度、偏移误差、满刻度误差、线性度。

HX711是一款专为高精度电子秤设计的24位a/D转换芯片。与同类芯片相比，该芯片集成了其它同类芯片所需的外围电路，如稳压电源、片上时钟振荡器等，具有集成度高、响应速度快、抗干扰能力强等优点，降低了规模成本，提高了性能提高了秤的可靠性。该芯片与后端MCU芯片的接口和编程非常简单。所有控制信号都由管脚驱动，无需对芯片内部寄存器进行编程。输入选择开关可任意选择a或B通道，并与内部低噪声可编程放大器相连。通道a的可编程增益为128或64，对应的全差分输入信号幅度分别为±20毫伏或±40毫伏。通道B是用于系统参数检测的固定64增益。芯片内提供的电源可以直接向芯片内的外部传感器和A/D转换器供电，系统板上不需要额外的模拟电源。芯片中的时钟振荡器不需要任何外部设备。开机自动复位功能简化了开机初始化过程。芯片引脚图如图3.2所示。

（一）HX711 芯片引脚图：

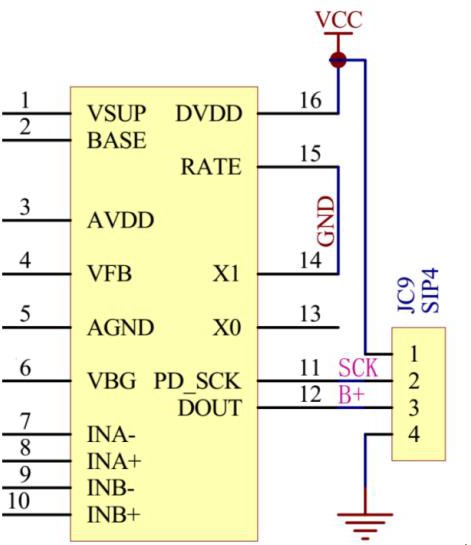


图 3-2 HX711 引脚功能图

如上图 3-2 所示，HX711AD 转换芯片的工作原理是：AD 转换芯片的外部电路将从称重传感器直流电桥输出的模拟信号进行滤波，再由 AD 转换芯片的模拟输入通道将模拟信号进行

128 倍的增益，然后采样为 24bit 的数字信号输入到单片机。

表 3-2 HX711AD 转换器引脚功能表

|  |  |  |
| --- | --- | --- |
| 引脚标号 | 引脚符号 | 引脚功能说明 |
| 1 | VSUP | 稳压电路供电电源端口 稳压供电电压为 2.6 ~ 5.5V |
| 2 | BASE | 模拟输出端口 用作稳压电路控制输出 |
| 3 | AVDD | 模拟电源输入端，输入电压为: 2.6 ~ 5.5V |
| 4 | VFB | 模拟输入端 稳压电路控制输入（不用稳压电路时应接地） |
| 5 | AGND | 模拟地端 |
| 6 | VBG | 模拟输出端 参考电源输出 |
| 7 | INNA | 模拟输入端 通道 A 负输入端 |
| 8 | INPA | 模拟输入端、通道 A 正输入端 |
| 9 | INNB | 模拟输入端、通道 B 负输入端 |
| 10 | INPB | 模拟输入、通道 B 正输入端 |
| 11 | PD-SCK | 数字输入断电控制（高电平有效）和串口时钟输入 |
| 12 | DOUT | 数字输出、 串口数据输出 |
| 13 | XO | 数字输入、输出，晶振输入 |
| 14 | XI | 数字输入、外部时钟或晶振输入端 |
| 15 | RATE | 数字输入、输出数据速率控制端 |
| 16 | DVDD | 电源，供电标准为: 2.6 ~ 5.5V |

HX711AD 转换芯片输出信号的计算：

本次设计的智能电子秤量程为 0~10kg, 灵敏度为 1.0mV/V，称重传感器的供电电压为

4.3V，称重传感器称重恰好为量程最大值 10kg 时的输出电压（满量程电压）为：

*V*full-scale =*VS* \* *KS* (3-1)

如式子 3-1 所示，*V*full-scale 为称重传感器的满量程电压，*VS* 为称重传感器的供电电压，

*KS* 为称重传感器的灵敏度系数。满量程电压*V*full-scale ：

4.3m*V* \*1.0m*V* / *V*  4.3m*V* (3-2) 称重传感器输出的电压模拟信号输入到 HX711AD 转换芯片，HX711 内部的增益通道将这

一放大，然后采样为 24bit 的数字信号输入到单片机。

HX711 对模拟信号的增益倍数为 128 倍，增益后的电压大小为：

4.3m*V* \*128  550.4m*V* (3-3) 满量程情况下，HX711 模数转换（采样）后输出的数值为：

0.554\*224 / 4.3≈2147483 (3-4)

假设被称量物品的重量为 Wkg，称量该物品时 AD 转换器器输出的电压为 X，则称重传感器输出的电压为：

*W*kg\*4.3m*V* /10kg=0.43*W*m*V* (3-5)

经过 AD 转换后输出的数字信号大小为：

0.43*W*mV\*128\*224 / 4.3≈21478.65*W* (3-6)

*X* =21478.65 /100=2147.4865*W* (3-7)

*W* =*X* / 2147.4865kg≈2.15g (3-8)

# 称重传感器模块

电阻应变式压力传感器静态全桥如图3.2.1所示。

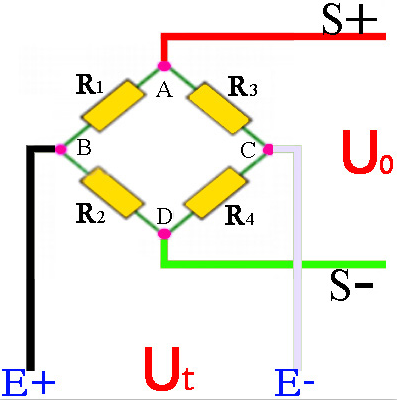


图3.2.1 电阻应变式压力传感器静态全桥

电阻应变传感器是一种利用电阻应变效应将各种机械量转换成电信号的结构传感器。电阻应变传感器核心元件的工作原理是基于材料的电阻应变效应。电阻应变片可单独作为传感器使用，也可作为敏感元件与弹性元件结合形成机械量传感器。

导体电阻随机械变形而变化的现象称为电阻应变效应。用电阻应变计将机械应变信号转换成△R/R后，由于应变和相应电阻的变化很小，很难直接准确地测量，且不便于处理。因此，应变片△R/R的变化应通过转换电路转换为电压或电流的变化。它的转换电路经常被用来测量电桥。

直流电桥的特点是信号不受各元件和导体的分布电感和电容的影响，具有较强的抗干扰能力。但由于机械应变输出信号小，要求采用高增益、高稳定度的放大器进行放大。

应变计传感器具有以下特性：

（1） 应变计可以制成各种机械传感器。

（2） 高分辨率、高灵敏度、高精度。

（3） 它结构轻巧，对试件影响小，对复杂环境适应性强，可用于高温、高压、强磁场等特殊环境，具有良好的频率响应。

（4） 商品化，使用方便，便于实现远程自动测量。

通过对压力传感器和电阻应变传感器的对比分析，最终选择了第二种方案。要求称重范围为0-10kg，满量程测量误差不大于0.005kg。考虑到称重平台的自重、振动和冲击部件，为避免传感器因超重而损坏，传感器量程必须大于额定重量10kg。选用电阻应变式压力传感器，测量范围为10kg，精度为0.01%，满足系统精度要求。

# LCD12864 液晶显示模块

显示模块采用LCD12864液晶显示电路的设计。128X64带汉字库是一种点阵式图形液晶显示模块，具有4位/8位并行、2线或3线串行接口，包含国标一级和二级简体汉字库，显示分辨率为128×64，有8192个16×16点汉字和128个16\*内置8位ASCII字符，采用灵活的界面模式和简单方便的操作指令模块，可形成中文人机交互图形界面。可显示8×4行16×16点阵汉字，也可完成图形显示。低电压和低功耗是该系统的另一个显著特点。与同类图形点阵液晶显示模块相比，该模块构成的液晶显示方案在硬件电路结构或显示程序上都要简单得多，且价格略低于同类点阵图形液晶显示模块。LCD12864的原理图如图3.5所示。



图 3-5 LCD1280 液晶模块引脚功能图

# 超重报警模块

根据智能电子秤的设计需要，设计出超重报警模块，其电路如图 3-6 所示。超重报警模块的作用是：当智能电子秤称量物品重量时，如果被称量物品的重量超过智能电子秤量程上限，即大于 10kg 时，发出指令使蜂鸣器发出声音和二极管闪烁来提示使用者，以避免因被称量物品超过量程太多造成电子秤损伤或毁坏。



图 3-6 超重报警模块

如上图 3-6 所示，超重报警模块由一个 2kΩ的定值电阻、一个发光二极管、一个 9102 三极管、一个蜂鸣器组成，由单片机的 I/O 口 P1.1 口控制其工作情况。其工作原理是：当智能电子秤称量物品重量时，单片机将接收到的当前物品的重量值与设定的最大量程值进行比较，若被称量物品的重量大于电子秤最大量程 10kg，则单片机将调用超重报警子程序， 并经 I/O 口 P1.1 发出相关指令，向超重报警模块的三极管的基极输出低电平信号，使得三极管导通，蜂鸣器和发光二极管与电源的正负极形成回路，以发出声音和灯光闪烁的方式向使用者发出超重警报。若电子秤未称量物品或被称量物品的重量小于量程最大值 10kg 时， 单片机经 P1.1I/O 口向三极管基极输出高电平信号，三极管处于截止状态，此时超重报警系统处于开路状态，不发出警报。

# 复位电路

智能电子秤在每次使用时，都需要对单片机进行复位（初始化），这样才能使单片机系统的各部件及电子秤的各功能部件从初始状态开始工作。本次设计采用了按键复位的复位方式，如图 3-7 所示。



图 3-7 按键复位电路图

如上图 3-7 所示，复位电路由一个独立按键、一个 10kΩ电阻、一个 10μF 的电容组成。复位电路工作地基本原理时：使用按键复位时，复位电路中的电容会给单片机的 RST 引脚一个短暂的高电平信号，这一信号会随着 Vcc 端的﹢5v 电源对电容的充电过程而逐渐减弱， 由此可见，RST 端的高电平持续时间的长短是由电容的充电时间决定的。因此，RST 端的高电平信号必须要维持足够长的时间（两个机器周期及以上）才能保证系统能够可靠地复位。复位的条件是：一、单片机系统处于正常工作状态[且振荡器](https://baike.sogou.com/v472849.htm)处于稳定状态。二、复位信号必须是持续两个[机器周期](https://baike.sogou.com/v502820.htm)及以上的高电平。满足以上两点，单片机就可以响应并且使系统复位。

# 矩阵键盘设计模块

当操作中需要较多的按键时，为了减少单片机的I/O口占用，通常采用矩阵式排列，即矩阵键盘。在矩阵键盘中，每一条水平线和垂直线在交点处不是直接连接的，而是通过一个键连接的。这样，一个端口（如P3端口）就可以形成4\*4=16个键，这是直接用端口线做键盘的两倍，而且线越多，区别就越明显。例如，添加另一行可以形成一个有20个键的键盘，而直接使用端口行只能再生成一个键（9个键）。因此，当键盘所需按键数较大时，采用矩阵法制作键盘是合理的。矩阵键盘的电路图如图3.8所示。



图 3-8 矩阵按键输入模块

\*4矩阵键盘的16个按键对应的功能表如3-1所示。

表3-1 矩阵键盘对应功能表

|  |  |  |  |
| --- | --- | --- | --- |
| 7 | 8 | 9 | 输入单价 |
| 4 | 5 | 6 | 去皮 |
| 1 | 2 | 3 | 重新输入 |
| . | 0 | 退格 | 确定 |

# 软件设计

* 1. 软件开发环境

该设计使用KeilμVision4进行编程。 Keil C51是由Keil Software在美国生产的与51系列兼容的单芯片C语言软件开发系统。 与汇编语言相比，C语言在功能，结构，可读性和可维护性方面具有明显的优势，因此易于学习和使用。 Keil提供了完整的开发解决方案，包括C编译器，宏程序集，链接器，库管理和强大的仿真调试器等。这些部件通过集成的开发环境（uVision）组合在一起。 运行Keil软件需要WIN98，NT，WIN2000和WINXP等操作系统。 如果您使用C语言编程，那么Keil几乎是您的最佳选择。 即使您不使用C而是仅使用汇编语言，其易于使用的集成环境和强大的软件仿真调试工具也可以使您事半功倍。

# 主程序流程图

主函数void main（）是程序的入口函数，完整的程序必须包含此函数。 在此功能开始时，需要先初始化MCU和某些外围设备，然后才能正常使用它们。 初始化并重新分配一些变量。 初始化后，进入死循环。 如果不进入死循环，程序将退出一次。 如果添加无限循环程序，它将继续循环以实现实时检测和执行的目的。 在设计主程序时，应注意，在主函数中放置过多代码是不合适的。 特定代码通常由函数封装，然后在主函数中调用，因此也可以轻松读取和修改。 具体流程图如下面的4.1所示。



图 4-1 主程序流程图

# LCD12864显示函数流程图

仅需严格按照制造商的时序要求对LCD12864显示器进行编程即可完成显示。 LCD12864的液晶显示器首先需要通过命令写入要显示位置的地址，然后按顺序写入数据。 在写入地址后显示第一个内容后，该地址将自动加一。 函数名称LCD12864\_display\_string（uchar x，uchar y，uchar \* s），参数为x，y，\* s，其中x，y表示LCD屏幕上的位置坐标，\* s为要显示的字符数组。 该软件根据输入的位置坐标计算地址。 显示功能的流程图在4.2中显示。



图 4-2 显示部分流程图

# 矩阵键盘检测函数的流程图

程序中矩阵键盘的具体测试方法如下（流程图中键代表P3口）。

（1） 首先，将键盘上所有的p3.0-p3.3线设置为低电平，然后检查p3.5-p3.7线是否有低电平现象。如果一列中有一个低电平，它将证明该列中的四个键之一已按下。如果线路中没有低电平，则没有按键。

（2） 当确认按下某个键时，将进一步确定按下哪个键。方法是：将四行p3.0-p3.3按顺序设置为低电平，即当一行低时，另一行保持在高电平。然后，通过确定当某条线是低电平时，如果第一步中获得的列是低电平，则可以确定该线与第一步中获得的列相交的键是按下的键。矩阵键盘检测功能流程图见4.4。



# 超重报警部分流程图

图 4-3 按键部分流程图

智能电子秤超重报警部分流程图如下图 4-4 所示

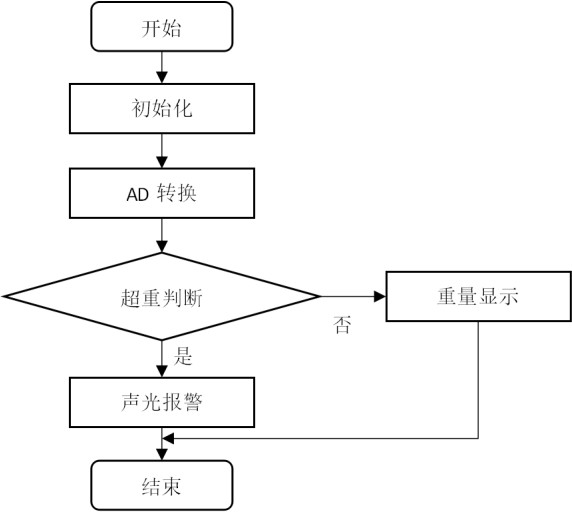


图 4-4 超重报警部分流程图

超重报警部分大致的工作方式是：称重传感器将物品重量转换为可测量的电压变化量， 然后 AD 转换器将这一模拟量转换为数字量送入单片机，单片机将物品重量值与设定的量程进行比较，如果物品重量超出智能电子秤量程的最大值，则单片机会调用超重报警子程序并发出指令，使超重报警器发出超重警报。

# 系统调试与调试结果

* 1. 硬件电路的制作

根据设计的电路图，将各元器件合理地排布于电路板上，再根据电路图进行手工焊接。实物图如图 5-1 所示。

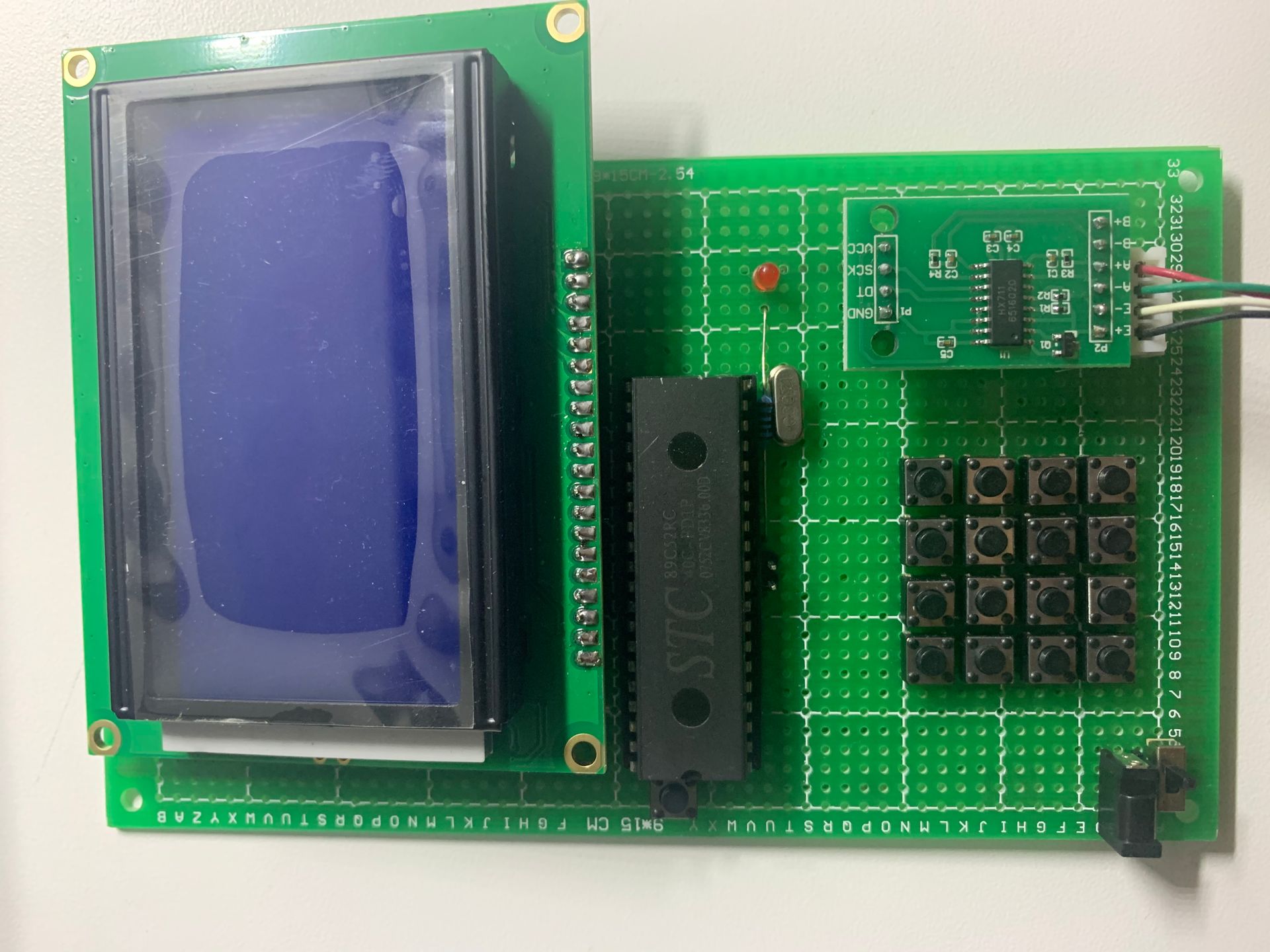


图 5-2 系统电路板实物图

# 硬件电路的调试和结果

* + 1. 硬件电路的调试

1、硬件部分各元器件焊接完毕后，用万用表对各元器件及各段电路进行分别测量，检测是否有元器件功能不正常以及电路是否有不符合设计要求的短路、开路等情况，更换不能正常工作的元器件并修改不正确的电路。

2、对编写的程序进行检查、修改，最终检查无误后，将 STC89C52 单片机装入开发板， 并且将程序烧录进单片机，然后将单片机取下插入电子秤的单片机座上。

3、万用表检测电路完全无误后，将三节 1.5V 的 5 号干电池正确装入电池槽，然后打开电源开关，蜂鸣器发出一声“滴”声并伴随着发光二级管闪烁一下后，LCD1602 液晶显示器显示的各项参数均正常。

4、操作三个独立按键，将电子秤量程设定为 0~10kg，然后将砝码放置于电子秤称台上，此时电子秤显示器示值与砝码本身标定的重量值一致，取下砝码，然后按下复位按键，电子秤显示器重量示值归“0”。将估计重量值大于 10kg 的物品慢慢放置于秤台上，当被称量物品施加于电子秤的重量刚刚超过 10kg 时，电子秤蜂鸣器发出报警声，电子秤超重报警系统工作正常。

# 硬件电路的调试结果

在对硬件电路进行调试时，使用智能电子秤对以下物品进行测量，得出的结果如下表

5-1 所示。

表 5-1 物品重量称量结果表

|  |  |  |
| --- | --- | --- |
| 物品名称 | 标准重量（g） | 智能电子秤示值（kg） |
| 5g 砝码 | 5 | 0.005 |
| 10g 砝码 | 10 | 0.010 |
| 20g 砝码 | 20 | 0.020 |
| 50g 砝码 | 50 | 0.050 |
| 一元硬币 | 6.05 | 0.006 |
| 5kg 杠铃片 | 5000 | 4.998 |
| 10kg 杠铃片 | 10000 | 9.995 |
| 20kg 杠铃片 | 20000 | 超重报警 |

如表 5-1 所示，根据称量物品重量的数据可知，便携式智能电子秤称量物品的测量精度、量程以及测量误差均符合设计要求，电子秤各功能模块均能正常工作，智能电子秤的设计达到预期效果。

# 设计总结

随着集成电路和计算机技术的快速发展，电子机器整体水平发生了巨大的变化，传统机构逐渐被智能机器取代，智能机器的核心部件还是单片机，与高价格相比，实现了广泛的应用和发展，智能机加快发展，传感器作为对象信息，在测量控制系统的入口逐渐受到人们的重视，因此要充分掌握相关智能机、微型控制器、传感器和各部分之间的关系，才能满足要求。

最终，在设计过程中遇到了很多困难，但是在数据检索和教师的帮助下，问题得到了解决，通过自己的实践增强实践能力，通过实际工程的设计，我可以看出书中所学的知识和实际运用的差异。单身更进一步了解了微型计算机系统的设计。

# 参考文献

[1]胡向东. 传感器与检测技术[M]. 机械工业出版社，2013.9.

[2]谢维成. 单片机原理及 C51 程序设计[M]. 清华大学出版社，2014.1.

[3]王祁. 智能仪器设计基础[M]. 机械工业出版社，2015.2.

[4]谭浩强. C 语言设计教程[M]. 清华大学出版社，2013.8.

[5]朱巍. 微机原理及接口技术[M]. 人民邮电出版社，2016.1.

[6]马岚. 数字集成电路[M]. 电子工业出版社，2017.1.

[7]李阳. 电子与自动化类毕业设计指导[M]. 中国电力出版社，2016.5.

[8]韦建英.徐安静 模拟电子技术[M]. 华中科技大学出版社，2013.2.

[9]刘理云. 嵌入式单片机开发与应用[M]. 北京理工大学出版社，2016.1.

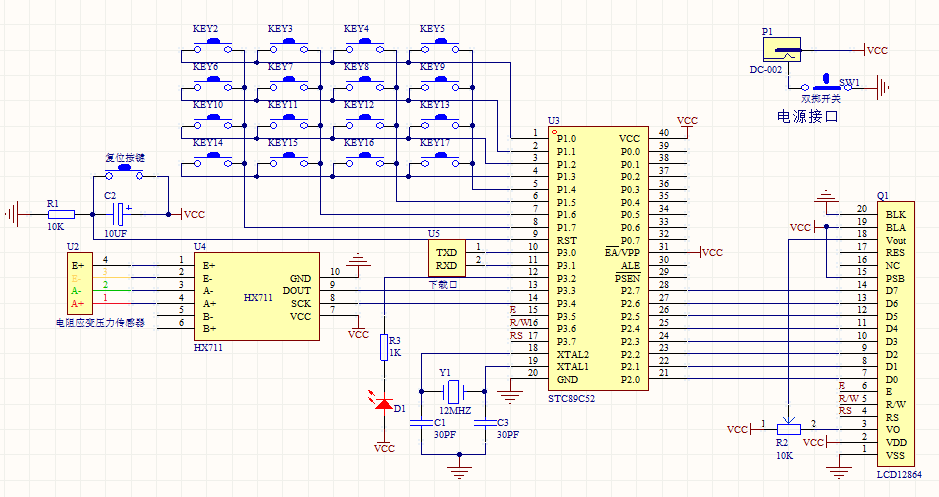
[10]魏芬. 基于 proteus 的单片机实验与课程设计[M]. 清华大学出版社，215.3.

# 致谢

毕业设计是对大学所学部分知识的一次综合检验与应用，也是对四年的大学学习所交出的一份答卷与总结，毕业设计完成的情况体现的不仅是我们在大学学期期间对相关知识的学习情况、掌握情况，也体现了我们将所学知识用于实际应用的能力。感谢指导老师聂文亮在本次设计的过程中不辞辛劳地予以帮助，导师的循循善诱、孜孜教导使学生受益匪浅并感佩于心。在此次设计过程当中，每当遇到有不懂的问题时，老师和同学都会在百忙之中抽出时间为我悉心讲解，和我一起探讨、分析并解决问题，指出我设计上的误区和不足，帮我改进设计中的欠缺点，使我能及时地发现问题、改正错误，把设计顺利的进行下去，在此也表示最衷心的感谢

附录

# 附录一 智能电子秤原理图



附录二 源程序

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

HX711头文件

实现功能：HX711的控制显示

补充说明：

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

#ifndef \_HX711\_H\_

#define \_HX711\_H\_

#include <reg52.h>

#define uchar unsigned char

#define uint unsigned int

#define ulong unsigned long

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*HX711引脚定义\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

sbit ADDO = P3^3;

sbit ADSK = P3^4;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*HX711函数定义\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

ulong ReadCount(void);

ulong fil1();

ulong fil2();

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*HX711变量定义\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

ulong init\_val; //存储零点重量

ulong value; //存储AD数值

//ulong weight; //存储实际重量

uint ii;

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

函数名称:ulong ReadCount(void)

函数作用:读取AD值

参数说明:

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

ulong ReadCount(void)

{

unsigned long Count;

unsigned char i;

ii=0;

ADSK=0; //使能AD（ PD\_SCK 置低）

ADDO=1;

Count=0;

while(ADDO&&ii<8000) //AD转换未结束则等待，否则开始读取

ii++;

for (i=0;i<24;i++)

{

ADSK=1; //PD\_SCK 置高（发送脉冲）

Count=Count<<1; //下降沿来时变量Count左移一位，右侧补零

ADSK=0; //PD\_SCK 置低

if(ADDO) Count++;

}

ADSK=1;

Count=Count^0x800000; //第25个脉冲下降沿来时，转换数据

ADSK=0;

return(Count);

}

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

函数名称:ulong fil()

函数作用:读取AD值

参数说明:

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

ulong fil()

{

uchar i;

ulong val=0; //记录采集值

for(i=0;i<3;i++) //循环采集5次

val+=ReadCount();

return val/3; //返回 中间数值 的数据

}

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

函数名称:ulong fil2()

函数作用:读取AD值。这个用于获取初值，增加采集次数和时间间隔使的更精确

初始都不准，那后面怎么测都会不准

参数说明:

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

ulong fil2()

{

uchar i,j;

ulong temp;

ulong val[9];

for(i=0;i<9;i++)

{

val[i]=ReadCount();

LCD12864\_delay(3);

}

for(i=0;i<9;i++)

{

for(j=0;j<9-i;j++)

{

if(val[j]>val[j+1])//把>改成<就是从小到大

{

temp=val[j];

val[j]=val[j+1];

val[j+1]=temp;

}

}

}

return val[4];

}

#endif

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

LCD12864头文件

实现功能：LCD12864的控制显示

补充说明：

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

#ifndef \_LCD12864\_H\_

#define \_LCD12864\_H\_

#include <reg52.h>

#include<intrins.h>

#define uchar unsigned char

#define uint unsigned int

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*LCD12864引脚定义\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define LCD P2 //并行数据口D0~D7

sbit RS =P3^7; //数据/命令选择 引脚

sbit RW =P3^6; //读/写选择 引脚

sbit E =P3^5; //使能信号 引脚

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*LCD12864函数定义\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void LCD12864\_delay(uint x); //LCD12864延时

void write\_com(uchar com); //LCD12864写命令

void write\_data(uchar dat); //LCD12864写数据

void LCD12864\_display\_string(uchar x,uchar y,uchar \*s); //在第y行，x+1列开始显示字符串

void LCD12864\_image3216(uchar x,uchar y,uchar code \*pPicture);//LCD12864在横坐标x（0~7），纵坐标y（1~4）开始显示一个32\*16的图片

void LCD12864\_clear3216(uchar x,uchar y); //LCD12864在横坐标x（0~7），纵坐标y（1~4）开始清除一个32\*16的图片

void LCD12864\_clear12864(); //LCD12864清除整个屏幕的画板

void LCD12864\_init(void); //LCD12864初始化函数

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*LCD12864变量定义\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

uchar code num12864[]=

{

0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFE,0x3F,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xFF,0x3E,0x73,0xE4,0xF8,0x03,0xE7,0xE0,0x08,0x01,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xE0,0x03,0x00,0x40,0x09,0x33,0x00,0x7F,0x8E,0x01,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xFC,0x13,0xC8,0x4C,0xF9,0x33,0x26,0x7F,0x18,0x03,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xE0,0x06,0x08,0x00,0x18,0x03,0x00,0x7E,0x3E,0x47,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xF1,0x9F,0x08,0x00,0x19,0x33,0x26,0x40,0x04,0x01,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xF9,0x8F,0x00,0x40,0xF8,0x03,0x26,0x7E,0x78,0x27,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xF0,0x0F,0x00,0x48,0xF9,0x33,0x00,0x3E,0x78,0x67,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xF2,0x27,0x39,0xC8,0x71,0x33,0x27,0x3E,0x7E,0x67,0x87,0xF0,0xFE,0x1F,0xFF,

0xFF,0xE4,0xF2,0x00,0x03,0x03,0x03,0xE0,0x78,0x7E,0x67,0x87,0xF0,0xFE,0x1F,0xFF,

0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x87,0xF0,0xFE,0x1F,0xFF,

0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,

0xF4,0xBF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xF8,0x3F,0xFF,0xFB,0xFF,0xFF,

0xF8,0x7F,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xF8,0x03,0xFF,0x79,0xFF,0xFF,

0xC0,0x05,0xFF,0xFF,0xFF,0xFC,0xFF,0xFF,0xFF,0xFF,0xFF,0xF3,0xEE,0x39,0xFF,0xFF,

0xC0,0x05,0xFF,0xFF,0xFF,0xFC,0xFF,0xFF,0xFF,0xFF,0xFF,0xF3,0xEE,0x39,0xFF,0xFF,

0xF8,0x7F,0xFF,0xFF,0xFF,0xE1,0xFC,0xFF,0xFF,0xFE,0x7F,0xF3,0xC8,0x09,0xFF,0xFF,

0xF4,0x3F,0xFF,0xFF,0xFF,0x81,0xE3,0xFB,0xFF,0xFF,0x8F,0xFF,0x00,0x81,0xFF,0xFF,

0xF4,0x3F,0xFF,0xFF,0xFF,0x81,0xE3,0xFB,0xFF,0xFF,0x8F,0xFF,0x00,0x81,0xFF,0xFF,

0xCC,0xDF,0xFF,0xFF,0xFC,0x03,0x03,0xC3,0xFE,0xFF,0xF3,0xFE,0xF1,0xC3,0xFF,0xFF,

0xFC,0xFE,0xFF,0x0F,0xF8,0x00,0x03,0x06,0x7F,0x0F,0xFC,0x7F,0xF1,0xFB,0xFF,0xFF,

0xFF,0xFF,0x3E,0x07,0xE0,0x00,0x00,0x04,0x5F,0x80,0xFE,0x1F,0xF7,0xFF,0xE7,0xFF,

0xFF,0xFF,0x3E,0x07,0xE0,0x00,0x00,0x04,0x5F,0x80,0xFE,0x1F,0xF7,0xFF,0xE7,0xFF,

0xFC,0xFD,0x3C,0xE3,0xC0,0x00,0x07,0x88,0x41,0xF0,0x0F,0x83,0xFF,0xFF,0xBF,0xFF,

0xFF,0xFE,0x1D,0x13,0xC0,0x00,0x00,0x40,0x40,0x1E,0x01,0xC1,0xFE,0x3F,0x9B,0xFF,

0xFF,0xFE,0x1D,0x13,0xC0,0x00,0x00,0x40,0x40,0x1E,0x01,0xC1,0xFE,0x3F,0x9B,0xFF,

0xFF,0xFA,0x1C,0x6B,0x80,0x00,0x00,0x01,0xC0,0x00,0x00,0x20,0xF8,0x0F,0x87,0xFF,

0xFF,0xFC,0x0C,0x68,0x00,0x00,0x00,0x01,0xC0,0x78,0x00,0x00,0x30,0xC6,0x1D,0xFF,

0xFF,0xE4,0x02,0x10,0x00,0x00,0x00,0x02,0x40,0x80,0x00,0x00,0x31,0x36,0x1B,0xFF,

0xFF,0xE4,0x02,0x10,0x00,0x00,0x00,0x02,0x40,0x80,0x00,0x00,0x31,0x36,0x1B,0xFF,

0xFF,0xF8,0x03,0x00,0x00,0x03,0xF8,0x02,0x20,0x00,0x00,0x00,0x16,0x84,0x02,0xFF,

0xFF,0xD8,0x03,0x80,0x00,0x1F,0xFC,0x04,0x20,0x00,0x00,0x00,0x06,0x84,0x01,0xFF,

0xFF,0xE0,0x0F,0xE0,0x00,0x00,0x1F,0x08,0x10,0x07,0xF8,0x00,0x01,0x08,0x00,0xFF,

0xFF,0xE0,0x0F,0xE0,0x00,0x00,0x1F,0x08,0x10,0x07,0xF8,0x00,0x01,0x08,0x00,0xFF,

0xFF,0xA0,0x1F,0x60,0x00,0x00,0x78,0x30,0x10,0x8F,0xFE,0x00,0x00,0x3E,0x01,0xFF,

0xFF,0xC0,0x00,0x60,0x00,0x00,0x70,0x40,0x0C,0x7F,0x00,0x00,0x00,0x7F,0x81,0x3F,

0xFF,0xC0,0x00,0x60,0x00,0x00,0x70,0x40,0x0C,0x7F,0x00,0x00,0x00,0x7F,0x81,0x3F,

0xFF,0x40,0x00,0x80,0x00,0x00,0x10,0x80,0x02,0x07,0x00,0x00,0x00,0x40,0x40,0xFF,

0xFF,0x80,0x1F,0x00,0x01,0x8C,0xE0,0x80,0x01,0x07,0x80,0x00,0x00,0x40,0x00,0xEF,

0xFF,0xC0,0xE1,0x00,0x00,0x42,0x03,0x00,0x00,0x81,0x80,0x00,0x00,0x40,0x00,0xC7,

0xFF,0xC0,0xE1,0x00,0x00,0x42,0x03,0x00,0x00,0x81,0x80,0x00,0x00,0x40,0x00,0xC7,

0xFF,0x41,0x20,0x80,0x01,0xA1,0x03,0x00,0x00,0x61,0x00,0x00,0x00,0x70,0x00,0xEF,

0xFC,0x02,0x10,0x80,0x00,0x40,0x03,0x00,0x00,0x60,0xC4,0x20,0x00,0x48,0x00,0xFF,

0xFC,0x02,0x10,0x80,0x00,0x40,0x03,0x00,0x00,0x60,0xC4,0x20,0x00,0x48,0x00,0xFF,

0xFF,0x02,0x0C,0x60,0xE0,0x00,0x03,0x00,0x00,0x10,0x08,0x40,0x1F,0x80,0x00,0x2F,

0xFF,0x84,0x0C,0x10,0x9C,0x00,0x04,0x00,0x00,0x10,0x31,0xA0,0x28,0xC2,0x00,0x3F,

0xFB,0x04,0x02,0x0F,0x1B,0xC0,0x18,0x00,0x00,0x10,0x00,0x40,0xC6,0x31,0x80,0x2F,

0xFB,0x04,0x02,0x0F,0x1B,0xC0,0x18,0x00,0x00,0x10,0x00,0x40,0xC6,0x31,0x80,0x2F,

0xFF,0x04,0x02,0x00,0x18,0x7F,0xE0,0x00,0x00,0x10,0x00,0x01,0xC1,0x00,0x40,0x1F,

0xFB,0x84,0x02,0x00,0x18,0x40,0x00,0x00,0x00,0x08,0x00,0x0E,0xC1,0x00,0x00,0x37,

0xFB,0x84,0x02,0x00,0x18,0x40,0x00,0x00,0x00,0x08,0x00,0x0E,0xC1,0x00,0x00,0x37,

0xFF,0x04,0x02,0x00,0x18,0x40,0x00,0x00,

};

uchar code ASI[]=

{

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x1F,0xE0,0x40,0x00,

0x00,0x00,0x40,0x00,0x00,0x00,0x5D,0x80,0x7F,0xF8,0x89,0x00,0x08,0x80,0x8A,0x1B,//

0x08,0x80,0x8A,0x26,0x08,0x81,0x0C,0x26,0x08,0x81,0x0A,0x26,0x08,0x81,0x09,0x1E,

0x10,0x8A,0x09,0x06,0x20,0x8A,0x1C,0xA6,0xC0,0x78,0x00,0x1C,0x00,0x00,0x00,0x00,//单价.BMP0

};

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

函数名称:void LCD12864\_delay(uint x)

函数作用:LCD12864延时函数

参数说明:

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

void LCD12864\_delay(uint x)

{

uint j,i;

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

{

for(i=0;i<120;i++);

}

}

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

函数名称:uchar Lcd\_CheckBusy(void)

函数作用:LCD12864读忙函数

参数说明:

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

uchar Lcd\_CheckBusy(void)

{

unsigned char Busy;

LCD=0xff;

RS=0;

RW=1;

E=1;

\_nop\_();

Busy=LCD&0x80;

E=0;

return Busy;

}

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

函数名称:void write\_com(uchar com)

函数作用:LCD12864写命令

参数说明:com为LCD12864指令，参考手册

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

void write\_com(uchar com)

{

while(Lcd\_CheckBusy());

RS=0;

RW=0;

E=0;

\_nop\_();

\_nop\_();

LCD=com;

\_nop\_();

\_nop\_();

E=1;

\_nop\_();

\_nop\_();

E=0;

}

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

函数名称:void write\_data(uchar dat)

函数作用:LCD12864写数据

参数说明:dat为所写内容

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

void write\_data(uchar dat)

{

while(Lcd\_CheckBusy());

RS=1;

RW=0;

E=0;

\_nop\_();

\_nop\_();

LCD=dat;

E=1;

\_nop\_();

\_nop\_();

E=0;

}

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

函数名称:void LCD12864\_display\_string(uchar x,uchar y,uchar \*s)

函数作用:LCD12864显示字符

参数说明:在横坐标x（0~7），纵坐标y（1~4）显示字符串\*s

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

void LCD12864\_display\_string(uchar x,uchar y,uchar \*s)

{

uchar add; //存储显示位置

if(y==1) //在第1行显示

add=0x80+x;

else

if(y==2) //在第2行显示

add=0x90+x;

else

if(y==3) //在第3行显示

add=0x88+x;

else

if(y==4) //在第4行显示

add=0x98+x;

write\_com(add); //先写显示地址

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

{

write\_data(\*s);

s++;

LCD12864\_delay(1);

}

}

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

函数名称:void LCD12864\_image3216(uchar x,uchar y,uchar code \*pPicture)

函数作用:LCD12864显示一张32\*16像素图片

参数说明:在横坐标x（0~7），纵坐标y（1~4）开始显示一个32\*16的图片

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

void LCD12864\_image3216(uchar x,uchar y,uchar code \*pPicture)

{

uchar add,i,j;

if(y%2==1)

add=0x80;

else

if(y%2==0)

add=0x90;

write\_com( 0x34 ) ;

write\_com( 0x36 ) ;

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

{

write\_com(add+i) ;

if(y>2)

write\_com(0x88+x);

else

write\_com(0x80+x);

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

{

write\_data(pPicture[i\*4+j\*2] );

write\_data(pPicture[i\*4+j\*2+1] );

}

}

write\_com( 0x30 ) ;

}

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

函数名称:void LCD12864\_clear3216(uchar x,uchar y)

函数作用:LCD12864清除一张32\*16像素图片

参数说明:在横坐标x（0~7），纵坐标y（1~4）开始清除一个32\*16的图片

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

void LCD12864\_clear3216(uchar x,uchar y)

{

uchar add,i,j;

if(y%2==1)

add=0x80;

else

if(y%2==0)

add=0x90;

write\_com( 0x34 ) ;

write\_com( 0x36 ) ;

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

{

write\_com(add+i) ;

if(y>2)

write\_com(0x88+x);

else

write\_com(0x80+x);

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

{

write\_data(0x00);

write\_data(0x00);

}

}

write\_com( 0x30 ) ;

}

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

函数名称:void LCD12864\_image12864( uchar code \*pPicture )

函数作用:LCD12864显示一张128\*64的图片

参数说明:输入参数为字模数组，可以用图片取模软件获得

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

void LCD12864\_image12864( uchar code \*pPicture )

{

unsigned char i,j;

write\_com( 0x34 ) ;

write\_com( 0x36 ) ;

for(i=0;i<32;i++)

{

write\_com( 0x80+i);

write\_com( 0x80 );

for(j=0;j<8;j++)

{

write\_data( ~pPicture[i\*16+j\*2] );

write\_data( ~pPicture[i\*16+j\*2+1] );

}

}

for(i=0;i<32;i++)

{

write\_com( 0x80+i) ;

write\_com( 0x88 );

for(j=0;j<8;j++)

{

write\_data( ~pPicture[32\*16+i\*16+j\*2] );

write\_data( ~pPicture[32\*16+i\*16+j\*2+1] );

}

}

write\_com( 0x30 ) ;

}

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

函数名称:void LCD12864\_clear12864()

函数作用:LCD12864清除整个屏幕的画板

参数说明:

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

void LCD12864\_clear12864()

{

unsigned char i,j;

write\_com( 0x34 ) ;

write\_com( 0x36 ) ;

for(i=0;i<32;i++)

{

write\_com( 0x80+i) ;

write\_com( 0x80 );

for(j=0;j<8;j++)

{

write\_data(0x00 );

write\_data(0x00 );

}

}

for(i=0;i<32;i++)

{

write\_com( 0x80+i) ;

write\_com( 0x88 );

for(j=0;j<8;j++)

{

write\_data(0x00 );

write\_data(0x00 );

}

}

write\_com( 0x30 ) ;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*LCD12864初始化\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void LCD12864\_init(void)

{

write\_com(0x30); //选择基本指令集

write\_com(0x30); //选择8bit数据流

write\_com(0x0c); //开显示(无游标、不反白)

write\_com(0x01); //清除显示，并且设定地址指针为00H

write\_com(0x06); //指定在资料的读取及写入时，设定游标的移动方向及指定显示的移位，光标从右向左加1位移动

}

#endif

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

矩阵键盘头文件

实现功能：矩阵键盘的控制

补充说明：

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

#ifndef \_KEY\_H\_

#define \_KEY\_H\_

#include<reg52.h>

#define uchar unsigned char

#define uint unsigned int

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*矩阵键盘引脚定义\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define Key P1

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*矩阵键盘函数声明\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

uchar jiema(unsigned char key); //解码函数，输入按键编码，返回按键位置

void delay(); //延时函数

uchar keyscan(void); //按键查询函数，返回矩阵键盘位置

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

函数名称:uchar jiema(unsigned char key)

函数作用:转换按键码为1~16的数字

参数说明:返回按下的按键位置

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

uchar jiema(unsigned char key)

{

uchar n;

switch(key)

{

case 0x11: n= '.'; break;

case 0x21: n= '0'; break;

case 0x41: n= 'T'; break;

case 0x81: n= '='; break;

case 0x12: n= '1'; break;

case 0x22: n= '2'; break;

case 0x42: n= '3'; break;

case 0x82: n= 'C'; break;

case 0x14: n= '4'; break;

case 0x24: n= '5'; break;

case 0x44: n= '6'; break;

case 0x84: n= 'Q'; break;

case 0x18: n= '7'; break;

case 0x28: n= '8'; break;

case 0x48: n= '9'; break;

case 0x88: n= 'D'; break;

default: break;

}

return n;

}

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

函数名称:void delay()

函数作用:延时函数

参数说明:

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

void delay() //延时子程序

{

uchar m;

for (m = 5; m > 0; m--);

}

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

函数名称:uchar Keyscan(void)

函数作用:进行按键扫描

参数说明:返回按键值，=0时表示没有按键按下

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

uchar keyscan(void) //按键扫描程序 P1.0--P1.3为行线 P1.4--P1.7为列线

{

unsigned char rcode, ccode;

Key = 0xF0; // 发全0行扫描码，列线输入

if((Key&0xF0) != 0xF0) // 若有键按下

{

delay();// 延时去抖动

if((Key&0xF0) != 0xF0)

{

rcode = 0xFE; // 逐行扫描初值

while((rcode&0x10) != 0)

{

Key = rcode; // 输出行扫描码

if((Key&0xF0) != 0xF0) // 本行有键按下

{

ccode = (Key&0xF0)|0x0F;

do{;}

while((Key&0xF0) != 0xF0); //等待键释放

return jiema((~rcode) + (~ccode)); // 返回键编码

}

else

rcode = (rcode<<1)|0x01; // 行扫描码左移一位

}

}

}

return 0xff; // 无键按下，返回值为0

}

#endif

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

电子秤

补充说明：

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

#include<reg52.h>//头文件

#include<lcd12864.H>

#include<HX711.h>

#include<KEY.h>

#include<math.h>

#include<stdio.h>

#define MAX 3 //宏定义，单价最大输入3位数

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*引脚定义\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

sbit led=P3^2; //超重指示灯

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*变量定义\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

uchar key; //存储按键值

double price=0; //存储零时单价

double prices=0; //存储最终单价

bit price\_f=0; //记录输入单价标志位

bit price\_w=0; //记录输入单价完成标志位

uchar price\_z=0; //存储总价

bit dian\_f=0; //小数单价标志位

bit dian\_w=0; //小数单价标志位

uchar n=0; //记录当前输入了几位数

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

函数名称:void fixed\_display()

函数作用:固定显示函数

参数说明:

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

void fixed\_display()

{

LCD12864\_display\_string(2,1,"电子秤"); //2：表示第3列，1：表示第1行，所有关于显示都一样

LCD12864\_display\_string(0,2,"重量:"); //0：表示第1列，2：表示第2行，所有关于显示都一样

LCD12864\_display\_string(0,3,"单价: 0");

LCD12864\_display\_string(0,4,"总价:");

LCD12864\_display\_string(7,2,"g");

LCD12864\_image3216(6,3,ASI); //6：表示第7列，3：表示第3行，显示【元/Kg】

LCD12864\_display\_string(7,4,"元");

}

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

函数名称:void chuli\_num(uchar keys)

函数作用:在输入单价的时候按下数字按键处理函数

参数说明:

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

void chuli\_num(uchar keys)

{

if((price\_f==1&&price\_w==0&&n<MAX)||(dian\_f==1&&dian\_w==0))//判断是否为输入单价状态，并且输入未满最大整数位数或者小数未满一位，才可以继续输入

{

if(dian\_f==0) //判断是否为整数

{

price=price\*10+keys;//单价计算

n++; //输入的位数加1

if(n==MAX) //判断是否输入完成

price\_w=1; //是的话标记输入完成

}

else //否者为。小数

{

price=price+(float)keys/10;//单价计算 12+ 1/10=12.1

dian\_w=1; //标记小数输入完成。因为单价最低也就0.1元，所以只能输入一位小数

price\_w=1; //标记单价输入完成

}

}

}

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

函数名称:void chuli()

函数作用:按键处理函数

参数说明:

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

void chuli()

{

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*按下数字键1~9键\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if(key=='1'||key=='2'||key=='3'||key=='4'||key=='5'||key=='6'||key=='7'||key=='8'||key=='9')//判断当前按下是否为0~9的数字键

chuli\_num(key-0x30);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*按下‘0’键\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if(key=='0')

{

if((price\_f==1&&price!=0&&price\_w==0&&n<MAX)||(dian\_f==1&&dian\_w==0))//判断是否为输入单价状态，并且输入未满最大整数位或者小数未满一位，才可以继续输入

{

if(dian\_f==0) //判断是否为整数

{

price=price\*10+0; //单价计算

n++; //输入位数加1

if(n==MAX) //判断是否输入四位完成

price\_w=1; //是，标记输入完成

}

else //小数

{

price=price+0.0; //单价计算

dian\_w=1; //标记小数输入完成。

price\_w=1; //标记单价输入完成

}

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*按下‘.’键\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if(key=='.')

{

if(price\_f==1&&dian\_f==0) //标记位当前单价是具有小数

dian\_f=1;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*按下‘去皮’键\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if(key=='Q')

{

if(price\_f!=1) //重新获取初值，0kg对应的AD值

init\_val=fil();

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*按下‘单价’键\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if(key=='D')

{

price\_f=1; //标记位单价输入状态

LCD12864\_display\_string(3,3," ");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*按下‘重输’键\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if(key=='C')

{

if(price\_f==1)

{

price\_f=1; //清除所以数据，重新输入单价

price=0;

n=0;

price\_w=0;

dian\_f=0;

dian\_w=0;

LCD12864\_display\_string(3,3," ");

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*按下‘退格’键\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if(key=='T') //13

{

if(price\_f==1&&price!=0) //判断是否为输入状态，并且当前输入的单价不为0采集必要进行退格处理

{

if(dian\_f==1) //小数

{

price\_w=0;

dian\_w=0;

dian\_f=0;

price=(ulong)price;

LCD12864\_display\_string(3,3," ");

}

else //整数

{

n--; //输入的位数减1

price\_w=0;

price=(ulong)price/10;//单价计算

LCD12864\_display\_string(3,3," ");

}

}

else

if(price\_f==0)

{

prices=0;

//清空显示合计后的价格

LCD12864\_display\_string(3,3," ");

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*按下‘确认’键\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if(key=='=')

{

if(price\_f==1) //按下确定键后，将输入的单价记录下来

{

prices=price; //记录单价

price\_f=0; //以下清除所有变量

price=0;

n=0;

price\_w=0;

dian\_f=0;

dian\_w=0;

}

}

}

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

函数名称:void main()

函数作用:主函数

参数说明:

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

void main()

{

uchar weight\_s[8]; //存储重量转换为字符串

uchar price\_s[4]="0"; //存储单价转换为字符串

uchar z\_s[8];

uchar wei,i;

LCD12864\_init();

/\*LCD12864\_display\_string(1,2,"初始化.");

init\_val=fil2();

LCD12864\_display\_string(1,2,"初始化..");

init\_val=(init\_val+fil2())/2;

LCD12864\_display\_string(1,2,"初始化...");

init\_val=(init\_val+fil2())/2;\*/

write\_com(0x01); //清除屏幕显示

LCD12864\_image12864(num12864); //开机显示启动画面

init\_val=fil2(); //读取0kg初始值

while(ii>=8000) //判断模块是否插接好

{

LCD12864\_clear12864(); //清除开机画面

LCD12864\_display\_string(2,1,"☆警告☆");

LCD12864\_display\_string(1,2,"未检测到模块");

LCD12864\_display\_string(0,3,"关闭电源后检测下");

LCD12864\_display\_string(0,4,"HX711 是否插接好");

}

init\_val=(init\_val+fil2())/2; //读取0kg初始值，这里多次采集取平均值，为了提高测量精度

init\_val=(init\_val+fil2())/2;

init\_val=(init\_val+fil2())/2;

init\_val=fil2();

init\_val=(init\_val+fil2())/2;

init\_val=(init\_val+fil2())/2;

init\_val=fil2();

init\_val=(init\_val+fil2())/2;

LCD12864\_clear12864(); //清除开机画面

fixed\_display(); //显示固定内容

while(1) //死循环

{

if(price\_f==0) //非单价输入状态

{

value=fil(); //采集实际AD

if(value<init\_val)

value=(init\_val-value)/400.03;//转换成实际重量，其中最后一个数字是指1g所占的AD值

//value=(init\_val-value)/41.220;//转换成实际重量，其中最后一个数字是指1g所占的AD值

else

{

value=0; //重量=0

led=1; //关闭蜂鸣器

}

if(value<=10000) //判断是否超重

//if(value<=100000)

{

led=1; //关闭超重警示

//显示实际重量

wei=sprintf(weight\_s,"%ld",(ulong)value);

//wei=sprintf(weight\_s,"%0.1f",(double)value/10);

for(i=wei/2+3;i<7;i++)

LCD12864\_display\_string(i,2," ");

LCD12864\_display\_string(3,2,weight\_s);

//显示总价

wei=sprintf(z\_s,"%0.1f",(double)value/1000\*prices);

//wei=sprintf(z\_s,"%0.1f",(double)value/10000\*prices);

for(i=wei/2+3;i<7;i++)

LCD12864\_display\_string(i,4," ");

LCD12864\_display\_string(3,4,z\_s);

}

else //超重

{

led=0; //指示灯警示

LCD12864\_display\_string(3,2,"超重 ");//显示“超重”提示

}

}

key=keyscan();//获取按键返回值

if(key!=0xff) //判断是否有按键按下

{

chuli(); //按键处理

if(price\_f==1) //单价输入

{

if(dian\_f==1)

{

sprintf(price\_s,"%0.1f",price); //将价格转换成字符，有小数

LCD12864\_display\_string(3,3,price\_s); //显示价格

LCD12864\_clear3216(6,3);

}

else

{

sprintf(price\_s,"%ld",(ulong)price); //将价格转换成字符，整数

LCD12864\_display\_string(3,3,price\_s); //显示价格

LCD12864\_clear3216(6,3);

}

}

else

{

sprintf(price\_s,"%0.1f",prices); //将价格转换成字符，最终的价格

LCD12864\_display\_string(3,3,price\_s); //显示价格

LCD12864\_image3216(6,3,ASI); //显示【元/Kg】

}

}

}

}