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

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：本文设计是以 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、应变式压力传感器、4＊4 矩阵键盘五大部分组成。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，系统可以计算出总价格，并在液晶显示屏上实时显示出物体的重量和单价。为了防止系统负载，当被称重物超过 10 公斤时，添加了“超重提示机制”，系统为用户超重发送提示。

关键词：STC89C52 单片机，压力传感器，4*4矩阵键盘，HX711模数转换器，LCD12864液晶显示

Design of Portable intelligent electronic scale

Xiao zeng-bing

Chongqing Three Gorges University School of Electronics and Information Engineering Electrical Electronic Information Engineering Professional 2016 Chongqing Wanzhou 404100

Abstract： The design of this article is based on STC89C52 single-chip microcomputer as the core of the control of intelligent electronic scale, which is mainly composed of five parts: STC89C52 single-chip microcomputer, HX711 analog-to-digital converter, LCD12864 display, strain gauge pressure sensor, 4 * 4 matrix keyboard. The system is equipped with a matrix keyboard. The user can input the price of the item through the matrix keyboard. The system can calculate the total price and display the weight and unit price of the object on the LCD screen in real time. In order to prevent the system from loading, when the weight to be weighed exceeds 10 kg, an "overweight reminder mechanism" is added, and the system sends a reminder for the user to be overweight.

Key words: STC89C52 single chip microcomputer, pressure sensor, 4 * 4 matrix keyboard, HX711 analog-to-digital converter, LCD12864 liquid crystal display

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，比如人体秤，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。

1.1 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。

1.2 国内外电子秤的发展现状与趋势

上个世纪四十年代以前，我国的称重仪器都是机械式的。在 1940 时期，开发了机电称重仪器。在 1950 时期，基于称重传感器的电子称重仪表开始出现。自 1980 时期以来，中国通过自己的研究进行了消化吸收和技术改造。传统的机械称重仪表已进入将电子传感器与传感器，微电子技术，计算机技术以及集成技术集成的发展阶段。目前，电子称重仪器因其称量快速，读取方便，在恶劣条件下工作的能力以及易于与计算机技术集成以实现称重技术和过程控制而广泛应用于工矿企业，能源运输和商业领域。自动化。贸易和科学技术等各个领域，随着称重传感器技术和超大规模集成电

路与微处理器的进一步发展，电子称重技术及其应用范围将得到进一步发展，人们越来越受到关注。从各种规格的通用电子秤到大型电子称重系统，从简单的称重和计价到生产过程检测系统的测量控制单元，电子称重仪表的体积和种类繁多。扩大。根据近年来电子称重技术和电子称重仪表的发展以及电子称重仪表市场的需求，电子称重仪表的发展趋势是：小型化，模块化，智能化，集成化；它的技术性能趋向于高速和高精度高可靠性。它的应用趋于全面和结合。

对于全球电子商务市场，各个地区的发展都不平衡，主要为美国，欧盟和亚洲的相互制衡局面。

美国是世界上第一个发展电子商务的国家，也是电子商务发展最成熟的国家。它一直引领着全球电子商务的发展，是全球电子商务的成熟和发达地区。欧盟电子商务的发展起步较美国晚，但发展迅速，已成为全球电子商务的领先地区。亚洲作为电子商务发展的新秀，具有巨大的市场潜力，但近年来的发展速度和份额并不理想，是全球电子商务的可持续发展领域。

1.3 本设计研究的内容

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，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。

本文的结构如下：

第一章 绪论，简要介绍了电子秤的研究背景，研究目的，意义和研究现状。

第二章 总设计原理及方案，本章的主要内容是电子秤编程，首先是整体程序的选择和设计，然后是每个模块（传感器，放大器模块，信号转换模块，电源模块，人机界面）模块）特定的程序显示和设计。

第三章 硬件电路设计，在扩展每个模块的方案中，它引入了简单的功能并将其应用于每种方案中使用的主芯片，并详细说明了此电路设计的具体电路图。

第四章 软件设计，本章主要介绍软件的电子设计，这被称为该设计的主程序流程图和某些模块的子程序图。

第五章 系统调试与调试结果。

最后，对本文的主要工作和结果进行了总结和讨论。

2 总设计原理及方案

2.1 智能电子秤的设计要求

我们选择 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）精度与量程：测量精度为 0.1%，测量误差为±0.010kg，量程范围为 0~10kg；

2）led 警报灯：超出量程最大值 10kg 时光报警功能；

3）LCD 液晶显示功能：开机显示动画，重量显示，单价显示，总价显示；

4）键盘功能：输入单价，重新输入，确认，退格功能。

2.2 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。

2.3 智能电子秤原理框图

本次智能电子秤的基本原理是：将物品放在压力传感器称重平台上时，传感器将物品的重量转换为电压或电流的变化，然后进行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 显示屏会显示物品的重量。 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。

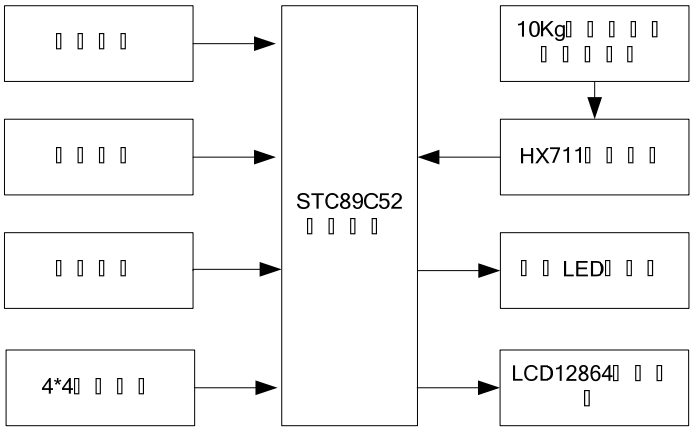


图 2-1 智能电子秤原理图

3 硬件电路设计

3.1 硬件控制电路模块

本次设计的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，本章主要介绍该设计中电路各部分的设计原理。 通过每个模块的功能描述，了解其工作原理及其在设计中的作用。

3.2 STC89C51 单片机模块的概述

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 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）STC89C52 单片机引脚如图 3-1 所示：

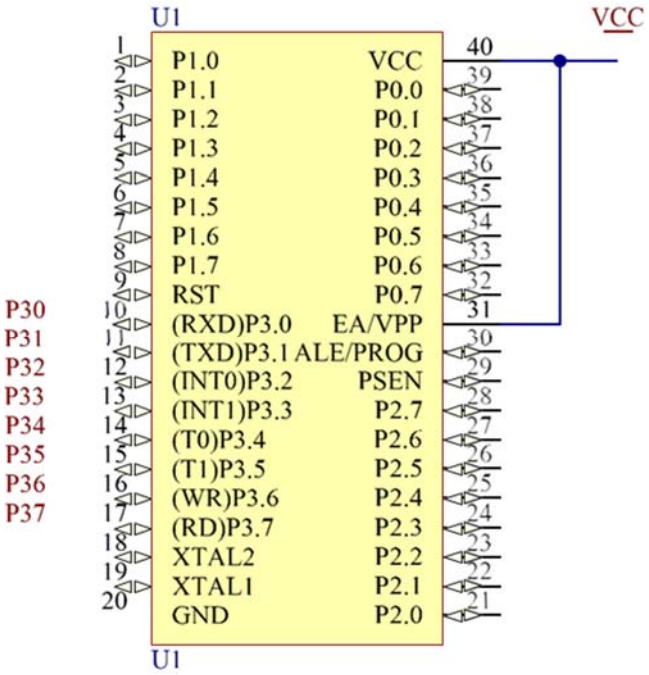


图 3-1 STC89C52 单片机引脚图

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

3.3 （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：

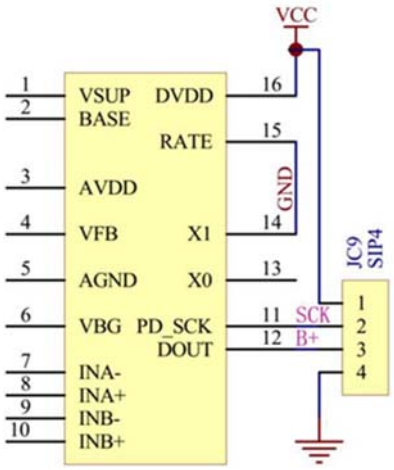


图 3-2 HX711 引脚功能图

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

3.4 称重传感器模块

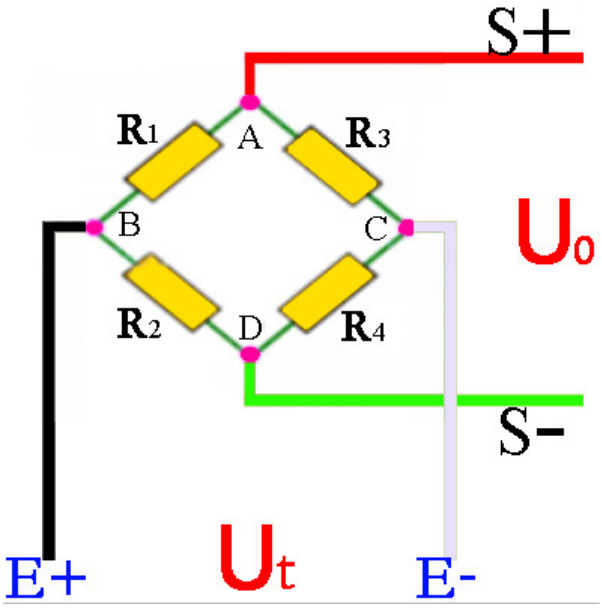


图3.3 电阻应变式压力传感器静态全桥

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，如图 3.3 所示。

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。

3.5 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。

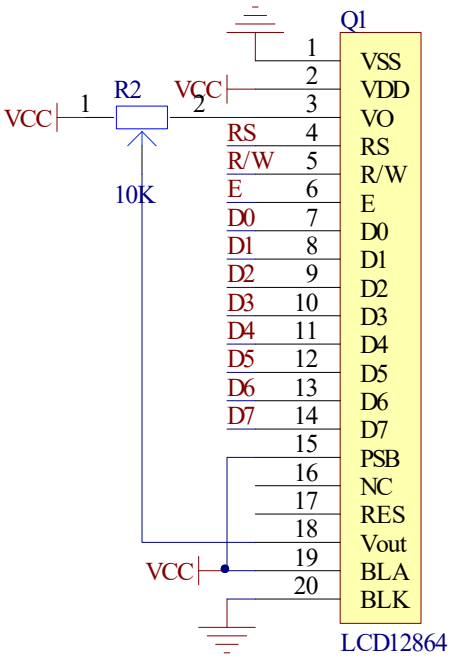


图 3-5 LCD1280 液晶模块引脚功能图

3.6 超重报警模块

根据智能电子秤的设计需求，设计了一个超重报警模块，其电路如图 3-6 所示。 超重报警模块的功能是：称重智能电子秤的重量时，如果被称量物品的重量超过智能电子秤范围的上限，即大于 10kg，则发出命令进行 二极管闪烁，提醒用户避免超过称重范围的称重物品引起电子秤损坏或破坏。

3.7 复位电路

智能电子秤每次使用时都需要复位（初始化）单片机，以使单片机系统的组件和电子秤的功能组件可以从初始状态开始工作。 本设计采用按键复位的复位方法，如图 3-7 所示。

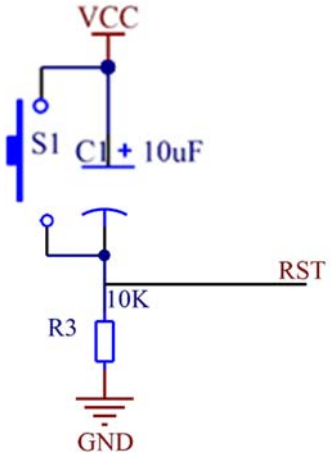


图 3-7 按键复位电路图

如上图 3-7 所示，复位电路由一个独立的按钮，一个 10kΩ 电阻和一个 10μF 电容器组成。复位电路工作位置的基本原理：使用按键复位时，复位电路中的电容器将向微控制器的 RST 引脚提供短时高电平信号。它逐渐变弱，这表明 RST 端子上的高电平持续时间取决于电容器的充电时间。因此，必须将 RST 端子上的高电平信号保持足够长的时间（两个机器周期或更长时间），以确保可以可靠地重置系统。 复位的条件是：1.微控制器系统处于正常工作状态，振荡器处于稳定状态。2.复位信号必须在两个或两个以上的机器周期内处于高电平。 满足以上两点，微控制器可以响应并重置系统。

3.8 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。

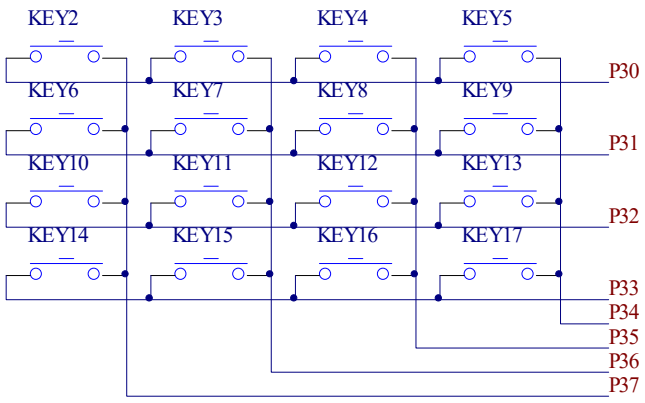


图 3-8 矩阵按键输入模块

每个按键相对应应的功能表如 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

4 软件设计

4.1 软件开发环境

该设计使用 Keil μVision4 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你事半功倍。

4.2 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。

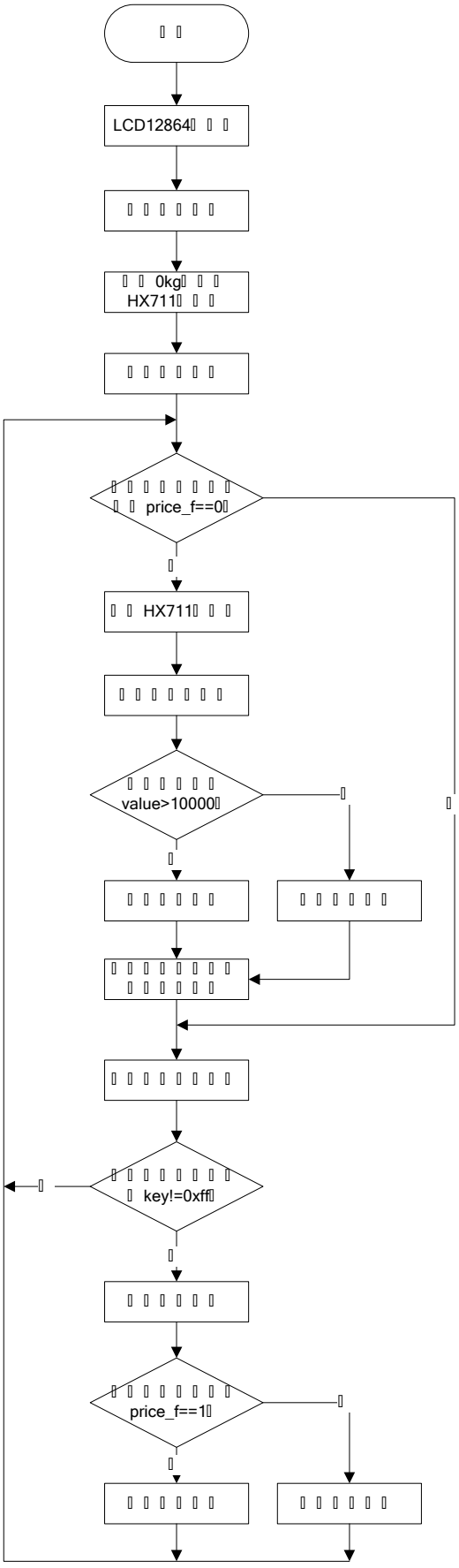

LCD12864
0kg
HX711
price_f==0
HX711
value>10000
key!=0xff
price_f==1

图 4-1 主程序流程图

4.3 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。

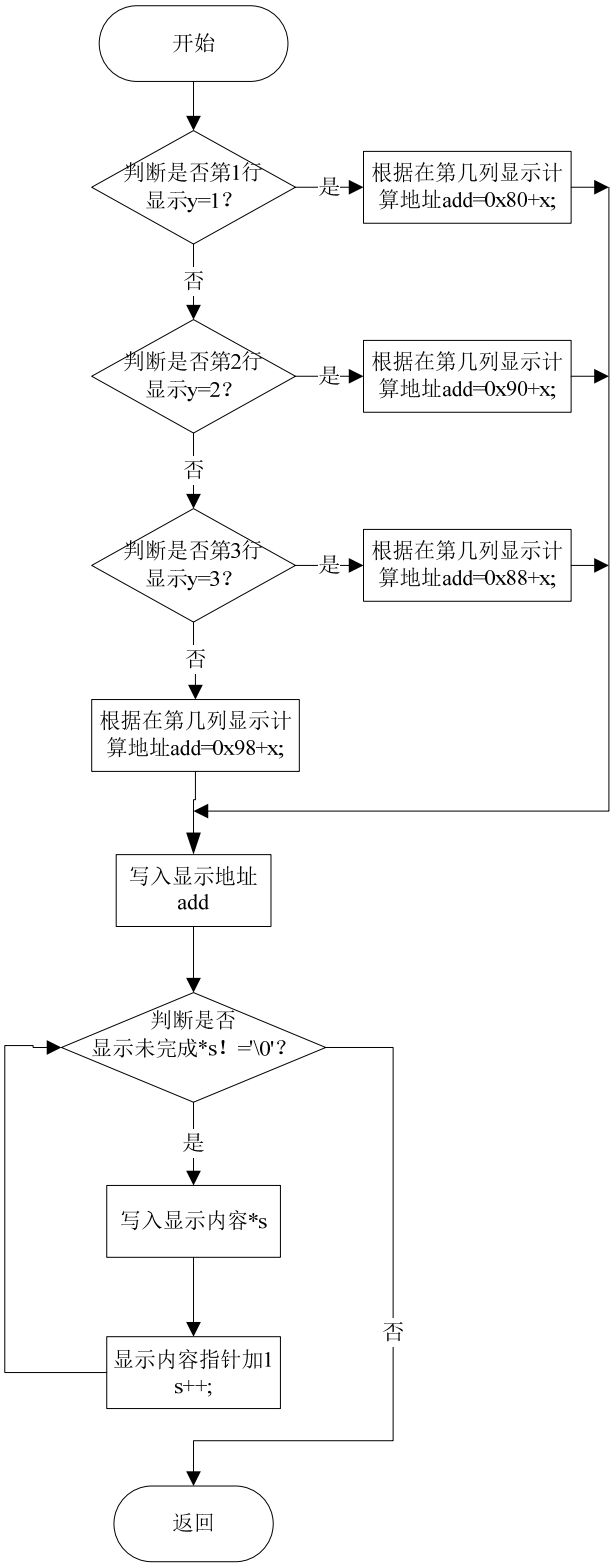


图 4-2 显示部分流程图

4.4 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 KEY 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。

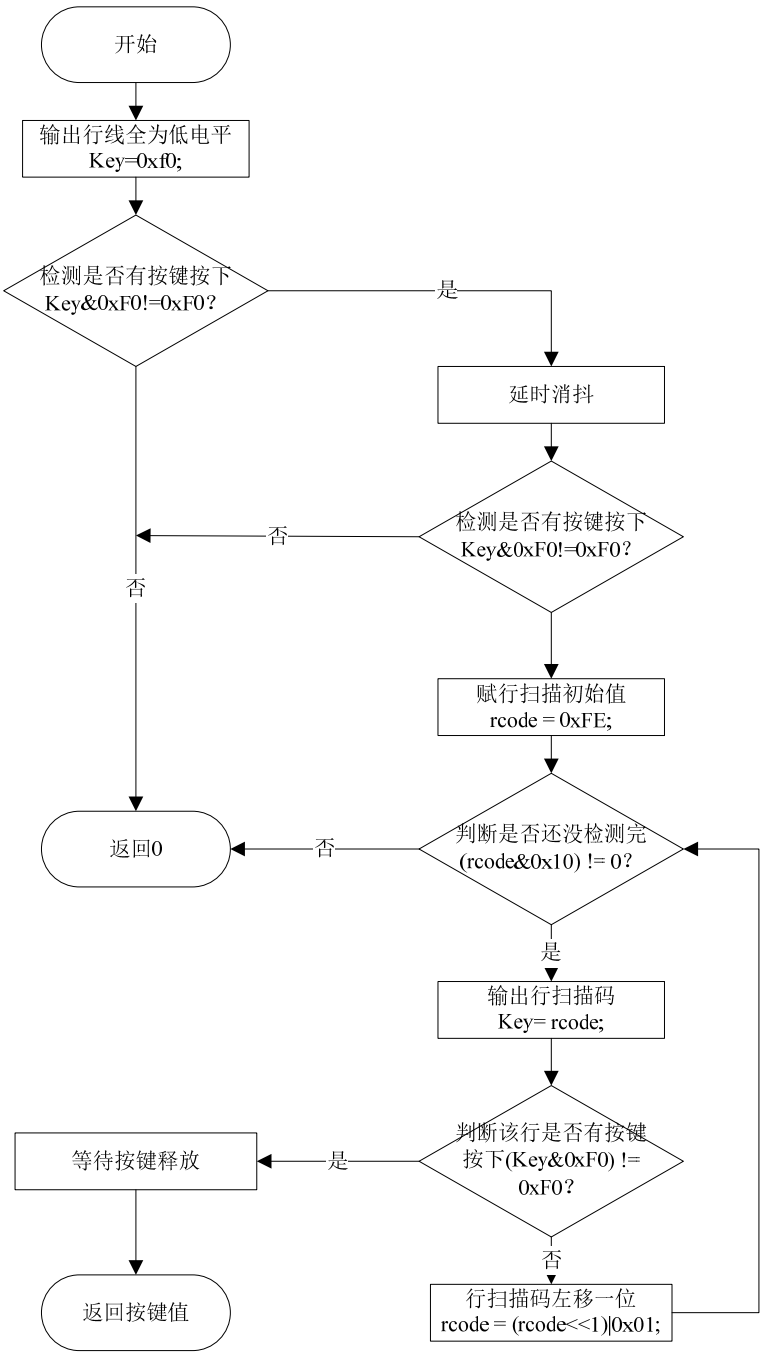


图 4-3 按键部分流程图

4.5 超重报警部分流程图

智能电子秤超重报警部分流程图如下图 4-4 所示

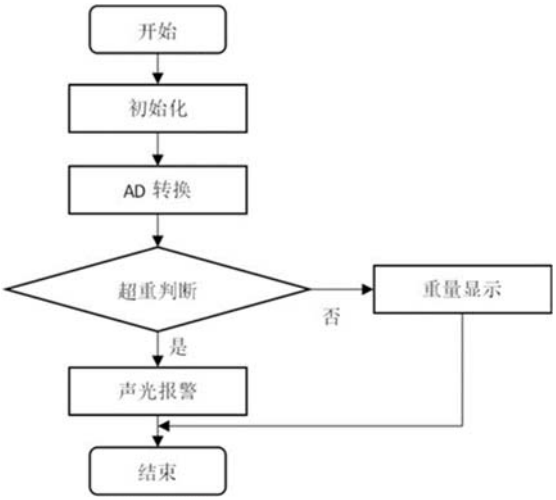


图 4-4 超重报警部分流程图

超重警报部分的工作方法是：称重传感器将物品的重量转换为可测量的电压变化，然后 AD 转换器将该模拟量转换为数字量并将其发送到微控制器。 比较范围。 如果物品的重量超过智能电子秤范围的最大值，单片机将调用超重警报子程序并发出指令，使 LED 灯亮起。

5 系统调试与调试结果

5.1 硬件电路的制作

根据设计的原理图，将各元器件排布于电路板上，再根据原理图进行焊接。实物图如图 5-1 所示。

图 5-2 系统电路板实物图

5.2 硬件电路的调试和结果

5.2.1 硬件电路的调试及结果

1.焊接完硬件部分的所有组件后，使用万用表分别测量每个组件和电路的每个部分，以检测组件是否有异常功能，以及电路是否存在短路和断路不符合设计要求。正常工作并修改错误电路的组件。

2.检查并修改编写的程序。最终检查正确后，将 STC89C52 单片机安装到开发板上，将程序刻

录到 MCU 中，然后将 MCU 移到电子秤的单片机基座中。

3.万用表检测电路完全正确后，连接电源并打开开关，LCD12864 液晶显示屏上显示的参数正常。

4.操作矩阵按钮，等待片刻，归零，设置单价，然后将其放置在电子秤的称重平台上。此时，电子秤显示器的显示值与重量本身校准的重量值一致。然后按复位按钮，电子秤显示器的重量显示值返回到“0”。将估计重量大于 10kg 的物品缓慢放在称重平台上。当电子秤上被称重的物品的重量刚刚超过 10kg 时，led 灯将亮起，电子秤的超重警报系统将正常工作。

硬件电路测试时，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，根据称重物品的数据，便携式智能电子秤的称重物品的测量精度，量程和测量误差均满足设计要求，其功能模块如下： 电子秤可以正常工作。 设计达到预期的效果。

5.3 软件系统的制作

5.3.1 软件系统的调试及结果

测试所需的工具：KEIL 软件，系统硬件，PL2303 下载器等。

系统软件由 KEIL 软件编写，.HEX 文件由编写的程序生成，然后通过 PL2303 下载器下载到微控制器。通过观察整个系统的运行状态，然后反复修改和调试该程序，最终获得了一个完美的程序。

系统软件调试主要遇到以下问题：

（1）LCD12864 显示花屏。

解决方案：本设计中使用 LCD12864 字体显示和图形显示。首先，只要显示一个字体，就会自动清除字体显示的内容和图形显示的内容。结果并非如此。未清除屏幕后，图形内容将重叠并导致出现花屏现象。通过检查 LCD 手册，可以发现字体显示和图形显示不同，以后可以修改程序。操作字体显示时，首先清除图形显示，反之亦然。 。

（1）有一种读取矩阵键盘按键的方法。在程序中，您需要将读取的每个键值与每个键对应起来，并赋予其特定的功能。如果直接直接手动计算键值，则似乎可以比较工作量。较大，可能会导致计算错误，从而导致大量时间浪费在调试上。

解决方案：因为此设计中有一个显示设备，所以可以将获得的键值直接显示在显示设备上，这样，当按下每个键时，相应的键值代码会显示在显示屏上，然后依次记录下来。即，每个按钮的设计功能以统一的方式执行。这样可以节省大量时间并确保正确性。

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.

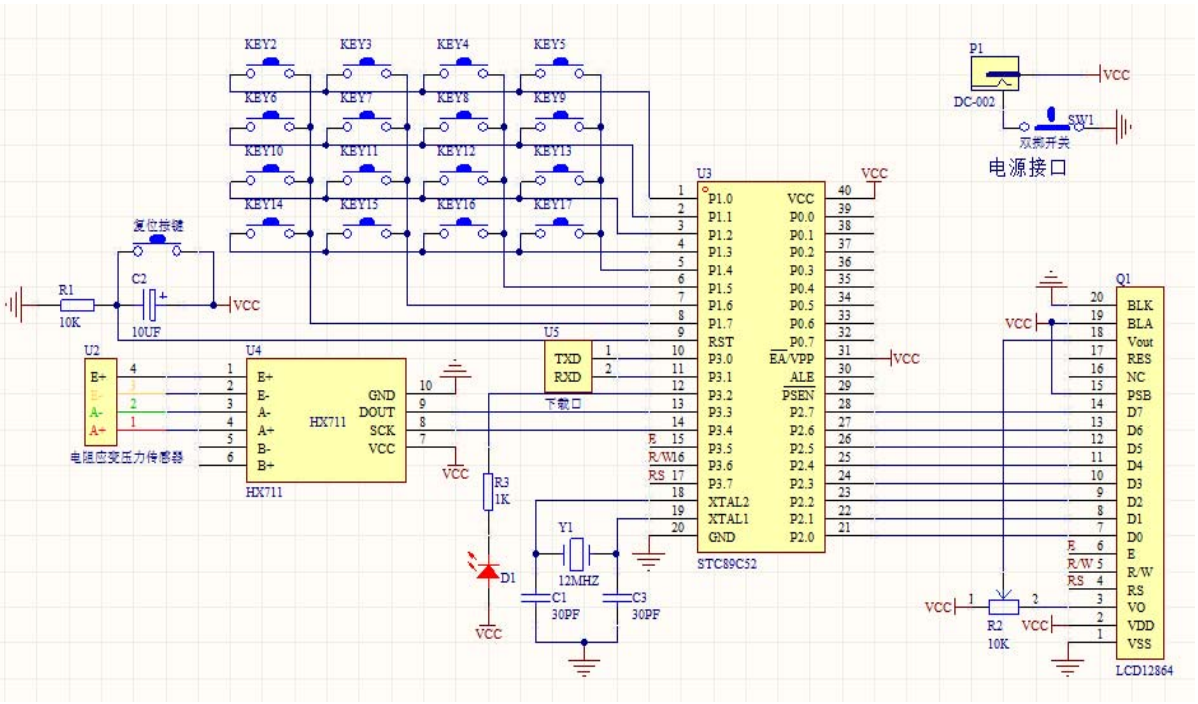
[10]魏芬. 基于 proteus 的单片机实验与课程设计[M]. 清华大学出版社，215.3.

致谢

本论文是在指导老师及相关材料的指导下完成的。本毕业设计是自己四年大学生涯的总结，是一项综合的应用实践。毕业设计也是过去四年对自己的考验。也可以将其视为填补空白的泄漏检查。不会的知识点会通过导师需求帮助，或者查阅相关的资料，也能检验自己的实践能力，因此感谢讲师的帮助。老师平日里工作繁多，但我做毕业设计的每个阶段，从选题到查阅资料，论文提纲的确定，中期论文的修改，后期论文格式调整等各个环节中都给予了我悉心的指导。后面论文的格式。老师提供了许多的帮助。我要对老师表示由衷的感谢和高度的敬意。同时，该毕业论文的撰写也得到了一些学生的热情帮助。感谢在整个毕业设计过程中与我紧密合作的同学们以及以前在各个方面都帮助过我的合作伙伴。在此，再一次衷心感谢帮助我的老师和同学。

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,0x
FF,
        0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFE,0x3F,0xFF,0xFF,0xFF,0xFF,0x
FF,
        0xFF,0xFF,0x3E,0x73,0xE4,0xF8,0x03,0xE7,0xE0,0x08,0x01,0xFF,0xFF,0xFF,0xFF,0x
FF,
        0xFF,0xE0,0x03,0x00,0x40,0x09,0x33,0x00,0x7F,0x8E,0x01,0xFF,0xFF,0xFF,0xFF,0x
FF,
        0xFF,0xFC,0x13,0xC8,0x4C,0xF9,0x33,0x26,0x7F,0x18,0x03,0xFF,0xFF,0xFF,0xFF,0x
FF,
        0xFF,0xE0,0x06,0x08,0x00,0x18,0x03,0x00,0x7E,0x3E,0x47,0xFF,0xFF,0xFF,0xFF,0x
FF,
        0xFF,0xF1,0x9F,0x08,0x00,0x19,0x33,0x26,0x40,0x04,0x01,0xFF,0xFF,0xFF,0xFF,0x
FF,
        0xFF,0xF9,0x8F,0x00,0x40,0xF8,0x03,0x26,0x7E,0x78,0x27,0xFF,0xFF,0xFF,0xFF,0x
FF,
        0xFF,0xF0,0x0F,0x00,0x48,0xF9,0x33,0x00,0x3E,0x78,0x67,0xFF,0xFF,0xFF,0xFF,0x
FF,
        0xFF,0xF2,0x27,0x39,0xC8,0x71,0x33,0x27,0x3E,0x7E,0x67,0x87,0xF0,0xFE,0x1F,0x
FF,
        0xFF,0xE4,0xF2,0x00,0x03,0x03,0x03,0xE0,0x78,0x7E,0x67,0x87,0xF0,0xFE,0x1F,0x
FF,
        0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x87,0xF0,0xFE,0x1F,0x
FF,
        0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x
FF,
        0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x
FF,
        0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x
FF,
        0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x
FF,
        0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x
FF,
        0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x
FF,
        0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x
FF,
```

```
0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x
FF,
0xF4,0xBF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xF8,0x3F,0xFF,0xFB,0xFF,0x
FF,
0xF8,0x7F,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xF8,0x03,0xFF,0x79,0xFF,0x
FF,
0xC0,0x05,0xFF,0xFF,0xFF,0xFC,0xFF,0xFF,0xFF,0xFF,0xFF,0xF3,0xEE,0x39,0xFF,0x
FF,
0xC0,0x05,0xFF,0xFF,0xFF,0xFC,0xFF,0xFF,0xFF,0xFF,0xFF,0xF3,0xEE,0x39,0xFF,0x
FF,
0xF8,0x7F,0xFF,0xFF,0xFF,0xE1,0xFC,0xFF,0xFF,0xFE,0x7F,0xF3,0xC8,0x09,0xFF,0x
FF,
0xF4,0x3F,0xFF,0xFF,0xFF,0x81,0xE3,0xFB,0xFF,0xFF,0x8F,0xFF,0x00,0x81,0xFF,0x
FF,
0xF4,0x3F,0xFF,0xFF,0xFF,0x81,0xE3,0xFB,0xFF,0xFF,0x8F,0xFF,0x00,0x81,0xFF,0x
FF,
0xCC,0xDF,0xFF,0xFF,0xFC,0x03,0x03,0xC3,0xFE,0xFF,0xF3,0xFE,0xF1,0xC3,0xFF,0x
FF,
0xFC,0xFE,0xFF,0x0F,0xF8,0x00,0x03,0x06,0x7F,0x0F,0xFC,0x7F,0xF1,0xFB,0xFF,0x
FF,
0xFF,0xFF,0x3E,0x07,0xE0,0x00,0x00,0x04,0x5F,0x80,0xFE,0x1F,0xF7,0xFF,0xE7,0x
FF,
0xFF,0xFF,0x3E,0x07,0xE0,0x00,0x00,0x04,0x5F,0x80,0xFE,0x1F,0xF7,0xFF,0xE7,0x
FF,
0xFC,0xFD,0x3C,0xE3,0xC0,0x00,0x07,0x88,0x41,0xF0,0x0F,0x83,0xFF,0xFF,0xBF,0x
FF,
0xFF,0xFE,0x1D,0x13,0xC0,0x00,0x00,0x40,0x40,0x1E,0x01,0xC1,0xFE,0x3F,0x9B,0x
FF,
0xFF,0xFE,0x1D,0x13,0xC0,0x00,0x00,0x40,0x40,0x1E,0x01,0xC1,0xFE,0x3F,0x9B,0x
FF,
0xFF,0xFA,0x1C,0x6B,0x80,0x00,0x00,0x01,0xC0,0x00,0x00,0x20,0xF8,0x0F,0x87,0x
FF,
0xFF,0xFC,0x0C,0x68,0x00,0x00,0x00,0x01,0xC0,0x78,0x00,0x00,0x30,0xC6,0x1D,0x
FF,
0xFF,0xE4,0x02,0x10,0x00,0x00,0x00,0x02,0x40,0x80,0x00,0x00,0x31,0x36,0x1B,0x
FF,
0xFF,0xE4,0x02,0x10,0x00,0x00,0x00,0x02,0x40,0x80,0x00,0x00,0x31,0x36,0x1B,0x
FF,
```

```
0xFF, 0xF8, 0x03, 0x00, 0x00, 0x03, 0xF8, 0x02, 0x20, 0x00, 0x00, 0x00, 0x16, 0x84, 0x02, 0x
FF,
0xFF, 0xD8, 0x03, 0x80, 0x00, 0x1F, 0xFC, 0x04, 0x20, 0x00, 0x00, 0x00, 0x06, 0x84, 0x01, 0x
FF,
0xFF, 0xE0, 0x0F, 0xE0, 0x00, 0x00, 0x1F, 0x08, 0x10, 0x07, 0xF8, 0x00, 0x01, 0x08, 0x00, 0x
FF,
0xFF, 0xE0, 0x0F, 0xE0, 0x00, 0x00, 0x1F, 0x08, 0x10, 0x07, 0xF8, 0x00, 0x01, 0x08, 0x00, 0x
FF,
0xFF, 0xA0, 0x1F, 0x60, 0x00, 0x00, 0x78, 0x30, 0x10, 0x8F, 0xFE, 0x00, 0x00, 0x3E, 0x01, 0x
FF,
0xFF, 0xC0, 0x00, 0x60, 0x00, 0x00, 0x70, 0x40, 0x0C, 0x7F, 0x00, 0x00, 0x00, 0x7F, 0x81, 0x
3F,
0xFF, 0xC0, 0x00, 0x60, 0x00, 0x00, 0x70, 0x40, 0x0C, 0x7F, 0x00, 0x00, 0x00, 0x7F, 0x81, 0x
3F,
0xFF, 0x40, 0x00, 0x80, 0x00, 0x00, 0x10, 0x80, 0x02, 0x07, 0x00, 0x00, 0x00, 0x40, 0x40, 0x
FF,
0xFF, 0x80, 0x1F, 0x00, 0x01, 0x8C, 0xE0, 0x80, 0x01, 0x07, 0x80, 0x00, 0x00, 0x40, 0x00, 0x
EF,
0xFF, 0xC0, 0xE1, 0x00, 0x00, 0x42, 0x03, 0x00, 0x00, 0x81, 0x80, 0x00, 0x00, 0x40, 0x00, 0x
C7,
0xFF, 0xC0, 0xE1, 0x00, 0x00, 0x42, 0x03, 0x00, 0x00, 0x81, 0x80, 0x00, 0x00, 0x40, 0x00, 0x
C7,
0xFF, 0x41, 0x20, 0x80, 0x01, 0xA1, 0x03, 0x00, 0x00, 0x61, 0x00, 0x00, 0x00, 0x70, 0x00, 0x
EF,
0xFC, 0x02, 0x10, 0x80, 0x00, 0x40, 0x03, 0x00, 0x00, 0x60, 0xC4, 0x20, 0x00, 0x48, 0x00, 0x
FF,
0xFC, 0x02, 0x10, 0x80, 0x00, 0x40, 0x03, 0x00, 0x00, 0x60, 0xC4, 0x20, 0x00, 0x48, 0x00, 0x
FF,
0xFF, 0x02, 0x0C, 0x60, 0xE0, 0x00, 0x03, 0x00, 0x00, 0x10, 0x08, 0x40, 0x1F, 0x80, 0x00, 0x
2F,
0xFF, 0x84, 0x0C, 0x10, 0x9C, 0x00, 0x04, 0x00, 0x00, 0x10, 0x31, 0xA0, 0x28, 0xC2, 0x00, 0x
3F,
0xFB, 0x04, 0x02, 0x0F, 0x1B, 0xC0, 0x18, 0x00, 0x00, 0x10, 0x00, 0x40, 0xC6, 0x31, 0x80, 0x
2F,
0xFB, 0x04, 0x02, 0x0F, 0x1B, 0xC0, 0x18, 0x00, 0x00, 0x10, 0x00, 0x40, 0xC6, 0x31, 0x80, 0x
2F,
0xFF, 0x04, 0x02, 0x00, 0x18, 0x7F, 0xE0, 0x00, 0x00, 0x10, 0x00, 0x01, 0xC1, 0x00, 0x40, 0x
1F,
```

```
        0xFB,0x84,0x02,0x00,0x18,0x40,0x00,0x00,0x00,0x08,0x00,0x0E,0xC1,0x00,0x00,0x
37,
        0xFB,0x84,0x02,0x00,0x18,0x40,0x00,0x00,0x00,0x08,0x00,0x0E,0xC1,0x00,0x00,0x
37,
        0xFF,0x04,0x02,0x00,0x18,0x40,0x00,0x00,
    };

    uchar code ASI[]=
    {
        0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x1F,0xE0,0x40,0x
00,
        0x00,0x00,0x40,0x00,0x00,0x00,0x5D,0x80,0x7F,0xF8,0x89,0x00,0x08,0x80,0x8A,0x
1B,//
        0x08,0x80,0x8A,0x26,0x08,0x81,0x0C,0x26,0x08,0x81,0x0A,0x26,0x08,0x81,0x09,0x
1E,
        0x10,0x8A,0x09,0x06,0x20,0x8A,0x1C,0xA6,0xC0,0x78,0x00,0x1C,0x00,0x00,0x00,0x
00,//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】
```

```
                }
            }
        }
    }
```