硕飞科技
SOFI TECHNOLOGY

ME830 单片机开发实验仪

指 导 教 程

伟纳电子(深圳硕飞科技有限公司)
SHENZHEN SOFI TECHNOLOGY CO., LTD.

TEL: +86-755-84867757
FAX: +86-755-84867941
WEB: WWW.SOFI-TECH.COM
Email: WILLAR@TOM.COM

技术支持网站：www.willar.com(伟纳电子网)
www.mcusj.com(单片机世界)

Publication Release Date: 2009-12
Revision A3

前　言

致用户：

欢迎您加入伟纳(深圳硕飞科技有限公司)用户行列！我们十分感谢您的惠顾！为使您的产品功能得到充分发挥，我们建议您在连接和操作之前，通读一遍使用说明书，请务必了解清楚本产品各功能模块、跳线、开关等的功能和设置方法后再使用，这样便于您掌握系统连接方法和使用要点，有助于您更好的用好这款单片机开发实验仪！

我们对用户手册的编排力求内容全面而又简单易懂，目的是从中您可以获取与您购买的开发系统有关的安装步骤、系统环境、基本操作、软硬件使用方在某些细节上不符，请以最新软件和您购买的开发实验仪实际配置为准。在编写本手册时我们难免会有疏漏甚至错误之处，请您多加包涵并热切欢迎您的指正，但硕飞科技将不对本手册可能出现的错误或疏漏负责。

本开发实验仪随机光盘中有大量的单片机学习资料和配套实验例程供您学习参考，如果您在使用本开发实验仪的过程中遇到问题，可以拨打我们的技术支持电话或发电子邮件联系。另外本公司有专业的技术支持论坛，提供大量以往用户的常见问题和学习经验与您分享，推荐您使用论坛发帖的方式咨询技术问题，可以和众多的用户学习交流。与我们产品或者配套实验例程相关的问题，我们的工程师将会给您细致讲解。您可以用以下两个网址登录：www.willar.com(伟纳电子网)，www.mcusj.com（伟纳单片机世界）！

声明：

本指导教程相关实验例程和技术资料仅供用户在学习中参考，不得用于商业用途。如果需要转载或引用，请保留版权声明和出处。如果您在学习过程中有任何疑问，欢迎到我们的技术支持网站论坛发帖交流！

伟纳电子（深圳硕飞科技有限公司）

目　　录

6.3 扩展实验

附录 1 常见问题解答

第一章 简 介

ME830 单片机开发实验仪是伟纳电子（深圳硕飞科技有限公司）集多年单片机开发工具设计经验，最新研制的具有“实验仪、编程器、仿真器、ISP 下载线”四功能合一的新一代单片机开发实验系统，具有多项领先技术：ME830 单片机开发实验仪摒弃了其他同类产品广泛采用的具有多项缺陷的 SST 公版仿真以及 USB 转串口通讯方案，独家集成了真正 USB2.0 接口的专业仿真器 ICE52，仿真不占资源，可极速单步仿真。配有独立的 POD52 仿真头，不但可以仿真实验仪内部资源，还可以仿真用户目标板，当作独立的仿真器使用，可以完全替代千元级别的专业仿真器。ME830 可以直接支持目前大多数主流 51 系列单片机的烧写和实验，同时也兼容 AVR 系列单片机的烧写和实验。您只需要一台电脑和一套 ME830 即可轻松学习和开发 51/AVR 两种单片机。

ME830 板载丰富的实验硬件资源和接口，各个功能模块各自独立，并对外全部开放和引出 I/O 口线，既可简单地使用短路帽连接成默认电路（方便直观，适合初学者直接使用），也可以取下短路帽后采用杜邦线自由搭建自己的电路，适合有一定经验的用户使用。配合本公司精心编写的大量 C 语言与汇编实验例程，同时有提供专业的论坛技术支持，由本公司工程师提供全程学习指导，可使用户快速掌握单片机原理及其实用接口技术。

ME830 单片机开发实验仪适合单片机初学者、院校学生、技术人员学习单片机使用。专业的仿真和编程下载功能也真正适合单片机工程师开发单片机产品使用，最大限度的避免您将来的重复投资，具有非凡的性价比。

1.1 性能特点

- 实验仪.仿真器.编程器.下载线四合一；
- USB2.0 接口(真正的 USB 口,非 USB 转串口)；
- 直接支持多家公司（如 ATMEL，Winbond，NXP（Philips），SST，DALLAS）的 51 系列单片机的烧写和实验；
- 可以完美支持 AVR 芯片如 ATmega8515 的烧写和实验。用户也可以自制其它 AVR 系列单片机的最小系统，利用 ME830 的 ISP 下载功能及全开放的实验模块进行其它 AVR 单片机的实验，如 ATmega16，Atmega128 等；
- 独家内置 ICE52 专业 51 仿真器，零资源占用,无限制真实仿真（32 个 IO、串口、T2 可完全极速单步仿真，支持全系列标准 8051 芯片仿真）
- 增强版配有独立 POD52 仿真头,可以仿真实验板内部资源,也可以仿真外部目标板,当作独立的仿真器使用；
- 内置 ISP 下载功能,方便对外部目标板下载程序，当作独立的下载编程器使用；
- 首创烧写功能直接嵌入 Keil,学习开发快速高效；
- 拥有功能完善的专业编程控制控制软件 MEFlash，具有专业编程器软件的全部操作功能；
- 丰富的硬件资源,全开放式模块化设计,可以使用固定电路也可以自由组合,可配合任意单片机教程学习；
- 完善的 C 语言,汇编实验例程,配有详细的流程图；
- USB 直接供电与外部电源（5-12V）双重供电选择；
- CPU 控制的高灵敏过载与短路智能保护功能,有效保护实验仪硬件和计算机 USB 口免遭意外损坏；
- 极低的计算机配置要求,具有一个 USB 接口的笔记本或者台式电脑均可使用,支持 Win2000/XP/Visata；
- 蓝色半透明塑料外壳,使用携式方便

图 1.1 ME830 主机外观(尺寸 190x138x38mm)

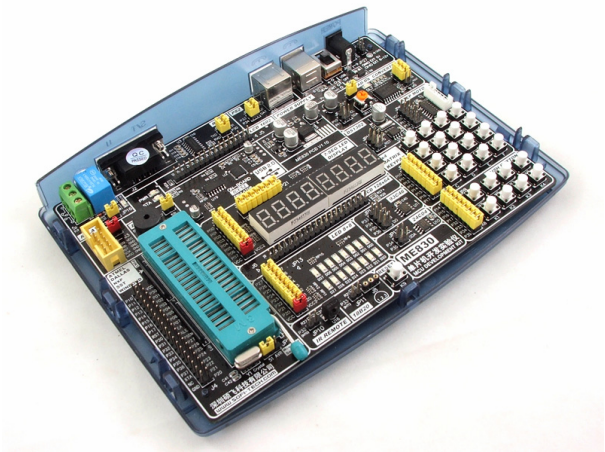


图 1.2 ME830 内部结构

1.2 板载实验硬件资源

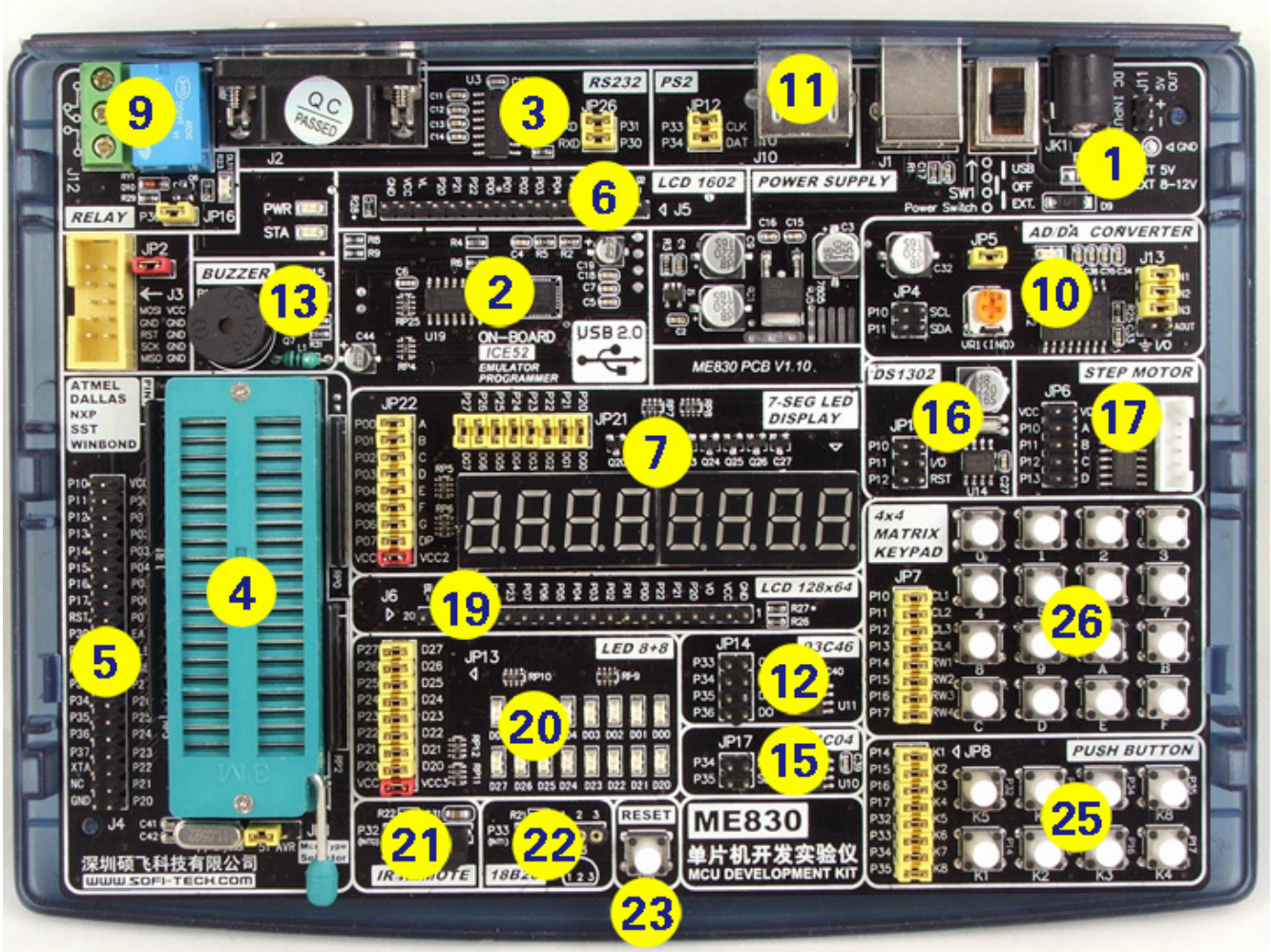


图 1.3

硬件资源列表

编号	说　　明	编号	说　　明
1	电源部分	13	蜂鸣器
2	增强型 ICE52 仿真器/编程器	15	24C04 存储器
3	DB9 串行接口及 MAX232 电平转换器	16	DS1302 实时时钟 RTC
4	锁紧座(用于放置实验芯片或仿真适配头)	17	步进电机驱动模块
5	40PIN 扩展端口	19	128x64 图像液晶模块接口
6	1602 字符型液晶模块接口	20	16 路发光二极管
7	8 位 7 段数码显示器	21	一体化红外线接收器
9	1 路继电器	22	DS18B20 一线温度传感器接口
10	模数/数模转换器	23	系统复位按键
11	PS2 键盘接口	25	8 路直接按键
12	93C46 存储器	26	4x4 矩阵键盘

1.3 产品组成

1.3.1 标准配置

部件名称	普通配置	增强配置	说明
ME830 实验仪主机	一台	一台	普通配置和增强配置主机完全一样
仿真头	SST89E516RD 单仿真芯片	POD52 仿真头	SST 单仿真芯片占用串口，定时器等重要资源；POD52 仿真头不占用任何资源，配合 ME830 可替代千元级的专业级仿真器，适合学习和开发产品使用
AT89S52(DIP40 封装)	一片	一片	实验用单片机
1602 液晶模块	一个	一个	字符液晶显示实验
32 键红外遥控器	一个	一个	红外遥控实验
USB 通讯电缆	一条	一条	仿真、下载通讯用
串口通讯电缆	一条	一条	用于串口通讯实验
10Pin ISP 下载连接线	一条	一条	ISP 下载、仿真头连接
杜邦头实验连接线	24 条	24 条	扩展实验用
短路帽	一包	一包	用于实验硬件连接
用户手册/配套光盘	一本/一张	一本/一张	驱动软件，例程，学习资料

1.3.2 可选配件

- **TFT 彩屏+SD 卡模块**（可以在 ME830 上完成单片机读写 SD 卡实验，单片机驱动 TFT 液晶实验；此模块还带有 DIP28 插座，插上 ATmega88 即可进行 AVR 芯片的开发和实验）
- **12864 图形液晶模块**（带汉字库）
- **DS18B20 温度传感器**
- **ATmega8515L**（用于 AVR 单片机实验），**ATmega88V**（用于 TFT 彩屏+SD 卡模块）
- **步进电机**

1.3.3 产品图片

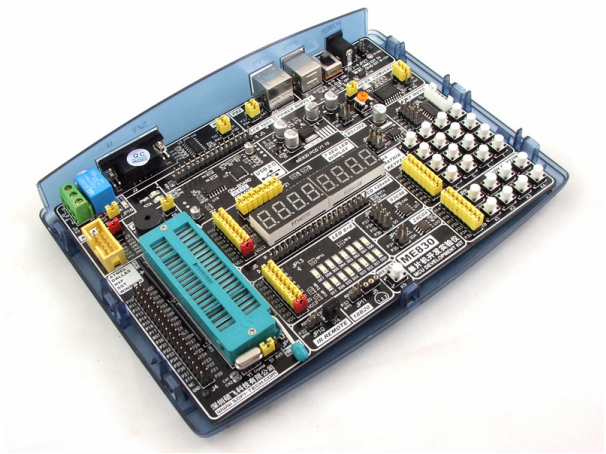


图 1.4 ME830 产品照片

第二章　硬件结构与安装

提示：本章对于使用 ME830 至关重要，务必详细了解 ME830 的硬件结构和各部分的功能后再联机操作。

2.1 硬件结构

ME830 主板电路布局（PCB 版本：V1.1），见图 2.1

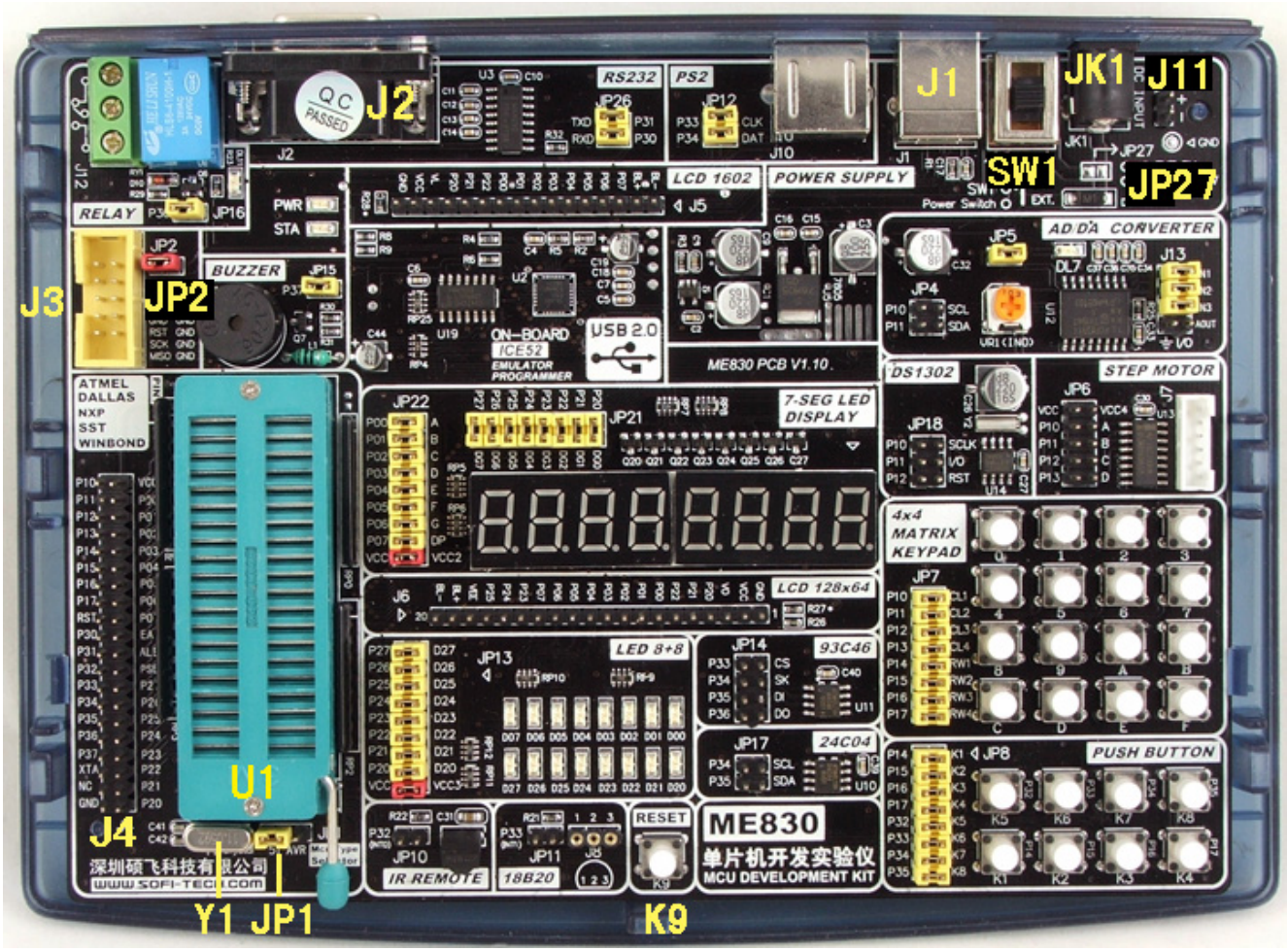


图 2.1

SW1：电源开关
JK1：　辅助电源输入插座
J1：　USB 通讯接口
J3：　ISP 下载接口
U1：　实验 CPU 活动插座
Y1：　晶振插座
K9：　复位按键

JP27：辅助电源选择跳线（焊盘）
J11：5V 电源输出接口
J2：　RS232 串口
JP2：目标供电选择跳线
J4：　40Pin 外扩接口
JP1：实验 CPU 类型（51/AVR）选择跳线

提示：ME830 默认用短路帽  连接实验硬件资源，图 2.1 是默认的短路帽位置，其他没有插短路帽的硬件模块请在做相应实验时插上短路帽：比如做 93C46 的实验，只需在 JP14 跳线上插上 4 个短路帽即可，不用时取下，方便快捷。如果需要按其他线路连接，可以取下短路帽后用配套杜邦线连接。

2.1.1 电源及系统控制

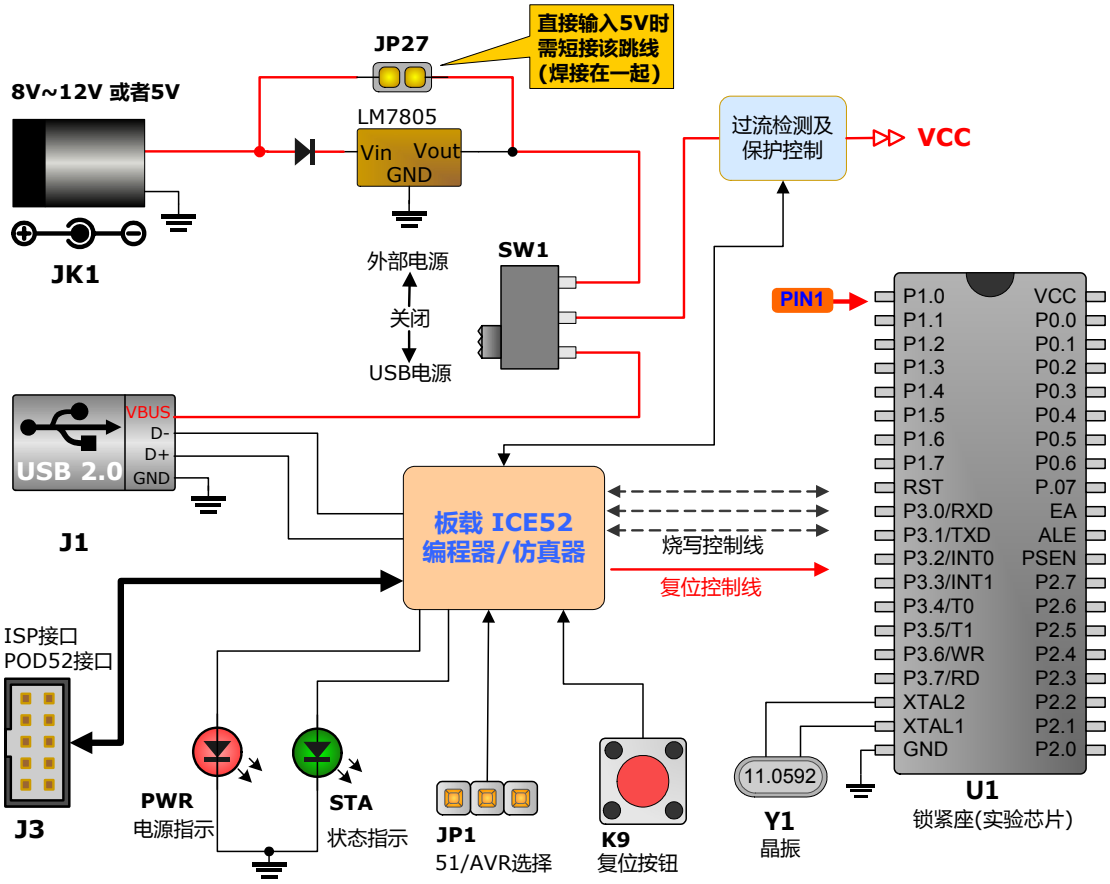


图 2.2

2.1.2 功能模块介绍

名称	标号	功能说明
电源部分	SW1	电源开关，拨到中间关闭整机电源，拨到 USB 端为 USB 供电，拨到“EXT”选择辅助电源供电
	JK1	辅助电源输入插座，插头规格 5.5x2.1,中心为正极，可使用以下两种电源： ● 8-12V 的直流电源：**必须确保 JP27 焊盘断开，否则将会烧坏 ME830** ● 5V 的稳压电源：**需要短接 JP27 焊盘，否则将导致供电不足** ME830 内置有完善的过载短路保护功能，可直接使用 USB 口供电，安全可靠。如果 USB 口供电不足或者需要脱机实验则需要使用辅助电源
	JP27	辅助电源选择跳线（用焊盘代替），默认是断开状态，可使用 8-12V 的外接直流电源。如果使用 5V 的直流电源，请用焊锡短接 JP27 焊盘，注意短路 JP27 后不能使用高于 5V 的电源。
	J11	从 ME830 输出 5V 电源，注意此电源无过载保护，请确保负载电流小于 300mA
仿真/编程/系统控制	J1	USB 接口，ME830 通过此接口与电脑 USB 口连接通讯，并为开发系统提供 5V 工作电源，注意需要保证 USB 口电压为 5V 并有至少 500mA 的负载能力，否则需要外接辅助电源供电。
	J3	仿真/ISP 下载接口，通过连接随机的 ISP 下载线对其他目标系统进行 ISP 在系统编程，当作一台下载编程器使用；或者连接 POD52 仿真头对实验仪的内部资源进行仿真调试，也可以对外部目标板进行仿真调试，当作独立的仿真器使用。
	JP2	目标系统供电跳线,ISP 下载或仿真时可短接此跳线帽由 ME830 向外部目标板提供 5V 电源(目标系统负载电流不能大于 300mA)
	PWR STA	PWR 是电源指示灯，STA 是状态指示灯。正常情况下 PWR 灯长亮，STR 灯只在编程操作时亮。如果 PWR 和 STA 两灯同时闪烁，表明系统出现过载保护了。如插反芯片，接错线等，此时请纠正错误的操作，按复位键 K9 或者重新开启一次电源开关 SW1 即可恢复正常工作

实验部分	U1	40Pin 锁紧插座，实验时插入 AT89S51/52 或其它兼容 40Pin 的 51 系列单片机，也可以插入 AVR 系列单片机如 ATmega8515L 做 AVR 单片机实验。仿真时插入仿真头 POD52。注意需要根据实际使用单片机的类型设置 JP1 跳线，使用 POD52 仿真时 JP1 跳线选择在“51”端。以上芯片的插入方向是芯片缺口方向朝电源开关
	JP1	CPU 类型选择跳线，根据所实验的 CPU 类型(51 或者 AVR)进行相应的设置
	J4	实验外扩接口，此端口连接 U1，可通过此接口引出 CPU 信号扩展外部实验
	Y1	晶振插座，可根据需要更换不同频率的晶振
	K9	复位按钮，用于复位实验芯片
	J2	RS232 串口，通过串口线与电脑 COM 端口连接通讯，用于单片机的串口通讯实验。如果您的计算机没有串口，可以选购一条 USB 转串口线

2.1.3 实验硬件模块连接说明

图 2.3 短路帽

ME830 的实验硬件模块各自独立，并对外全部开放 I/O 口，默认采用短路帽（也称为跳线帽、短路块、短路片，见图 2.3）和单片机的 I/O 端口进行连接，ME830 的主板上已经标识了各模块所连接的 I/O 端口名称，推荐初学者用默认的短路帽连接方式来实验，方便快捷。使用短路帽连接硬件的原则是：默认短接的短路帽可以永久保留不需断开，默认没有插短路帽的实验模块只有在做该模块的实验时才插上，不用时须取下短路帽，避免相互干扰。ME830 默认的短路帽位置请参考图 2.1。

对于有一定基础的用户和开发人员可以根据学习或者项目的需要自定义硬件线路，可以取下短路帽后用随机配送的杜邦头连接线任意组合线路。下面以 Ad/DA 转换模块为例进行说明：ME830 的 AD/DA 模块默认用短路帽连接到单片机的 P1.0，P1.1 这二个 I/O 端口，取下这二个短路帽后就可以用杜邦线连接到 CPU 的其他 I/O 口，见图 2.4 ，图中用 2 芯的杜邦头连接线将 AD/DA 模块改接到 P3.5，P3.6 端口。

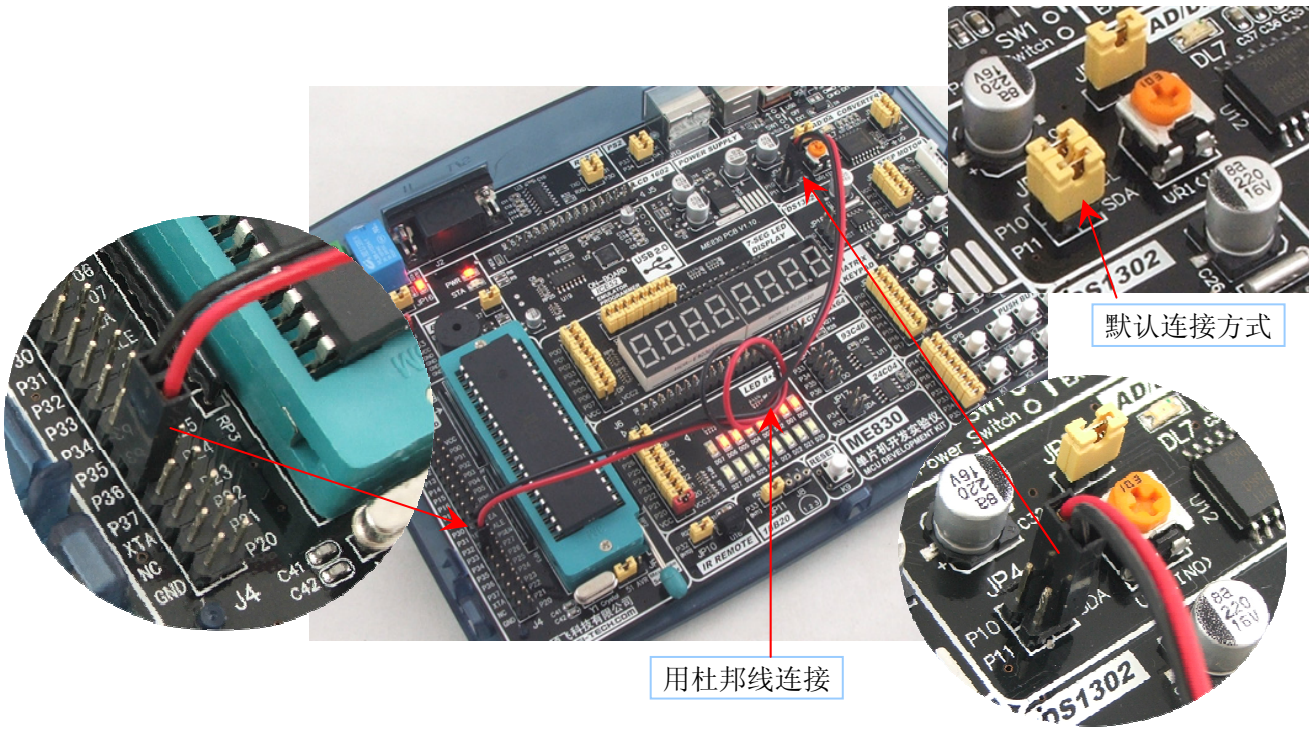


图 2.4 用杜邦连接线自由分配 I/O 端口示意图

提示：既可使用固定线路又可以自由组合线路的资源分配方式，使得 ME830 具有极大的方便性和灵活性。既解决了使用固定线路的局限，也避免了只能使用排线连接需要反复拔插的烦琐。

注意事项：

对于数码管显示模块，不使用时只须断开 JP22 跳线组的 VCC 端短路帽；

对于 LED 显示模块，不使用时只须断开 JP13 跳线组的 VCC 端短路帽；

做 12864LCD 显示实验时，需要断开数码管 JP22 跳线组的 VCC 端短路帽，否则 12864LCD 可能无法正常显示。

2.2 安装

ME830 单片机开发实验仪集成了“ICE52 增强型 51 仿真器/编程器”，支持所有标准 8051/8052 的仿真，并可以对 ATMEL/DALLAS/NXP(PILIPS)/SST/WINBOND 等公司的 51 系列单片机进行 ISP 下载。ICE52 的仿真功能与 Keil μVisison2/μVisison3/μVisison4 配合使用，借助 Keil 软件的强大功能实现 51 单片机的仿真，并具有独特的嵌入到 Keil 的编程下载功能。另外 ICE52 的编程功能还可以使用专业的 MEFlash 编程控制软件，具有专业编程器的所有操作功能。在使用 ME830 之前需进行相应的软件安装与设置。

2.2.1 安装 Keil C51 软件

Keil C51 是德国知名软件公司 Keil（现已并入 ARM 公司）开发的基于 8051 内核的微控制器软件开发平台，是目前开发 8051 内核单片机的主流工具。

1）将产品附带的光盘放入光驱，运行光盘目录 Software 下的 c51v812.exe，软件出现如下图所示的对话框

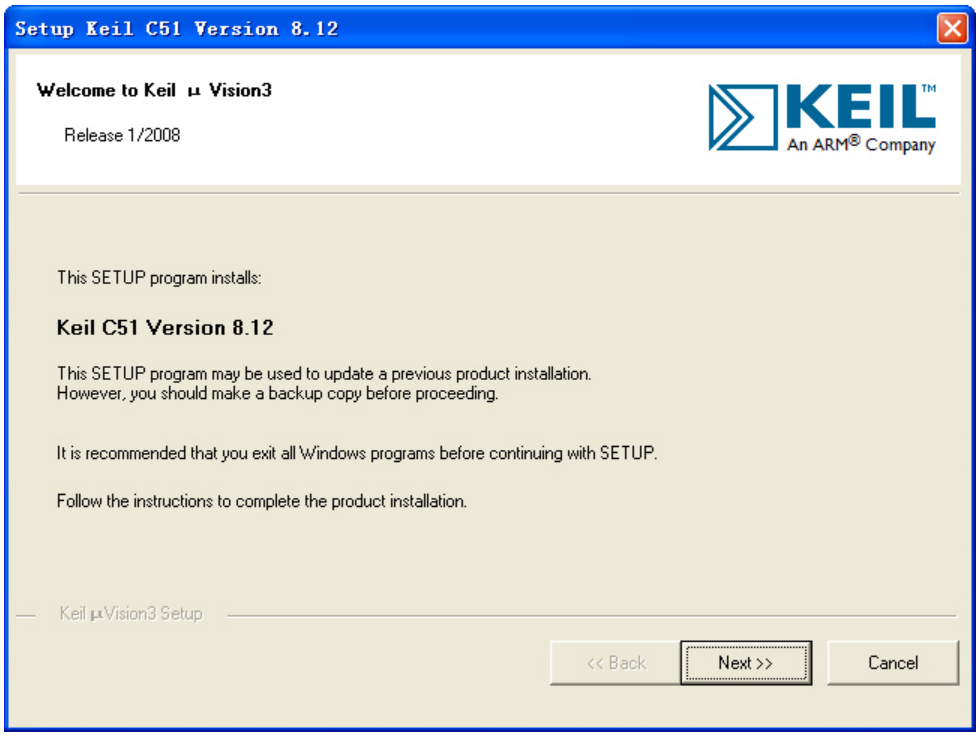


图 2.5 Keil 安装

2）在接下来的几个对话框中，点击 Next 按钮，在提示输入用户名称和公司名时，按需要填写即可

3）安装完成后，按 Finish 结束。

提示： Keil C51 是商业软件，本产品光盘上提供的是 V8.12 评估版软件，有 2K 代码限制。

如果您使用其他版本的 Keil 软件也可以，V7.5 以上版本都适用于 ME830。（V7.5 以下版本未测试，不保证能正常使用）

2.2.2 安装 ICE52 仿真驱动程序

在安装仿真驱动程序之前必须保证电脑上已经安装有 Keil 相应版本。仿真驱动安装步骤如下：

1）运行产品光盘 Software 目录下的 ICE52_DLL_SETUP. exe 软件，如下图所示

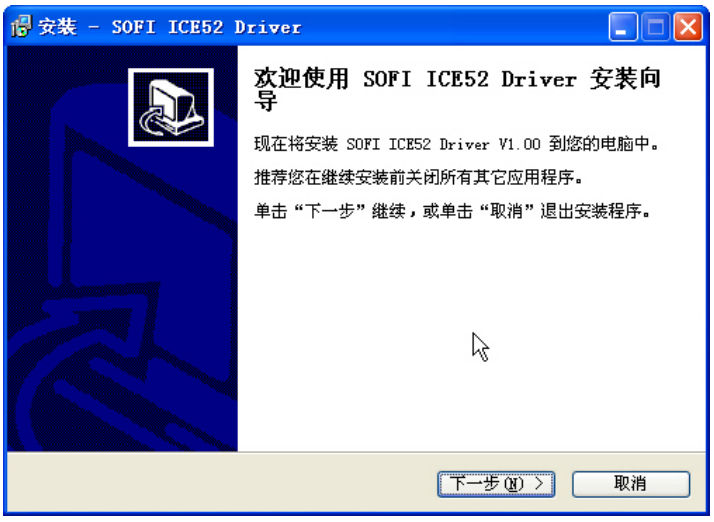


图 2.6 安装 Keil 接口驱动

2）在安装的的第 3 个画面，提示选择语言。可选择简体中文或英文。此选项决定在 Keil 中的仿真/编程对话框使用哪个语言，请根据需要进行选择。如图 2.7 所示：

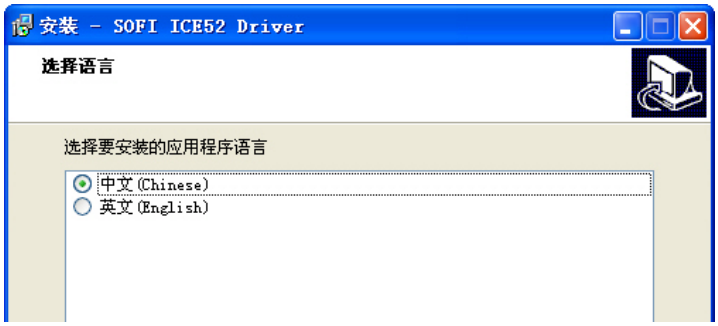


图 2.7 选择 ICE52 使用的语言

3）然后选择 Keil 的安装目录，目录必须正确。如果在选定的目录没有检测到 Keil 软件，软件会给出如图 2.9 所示的提示。

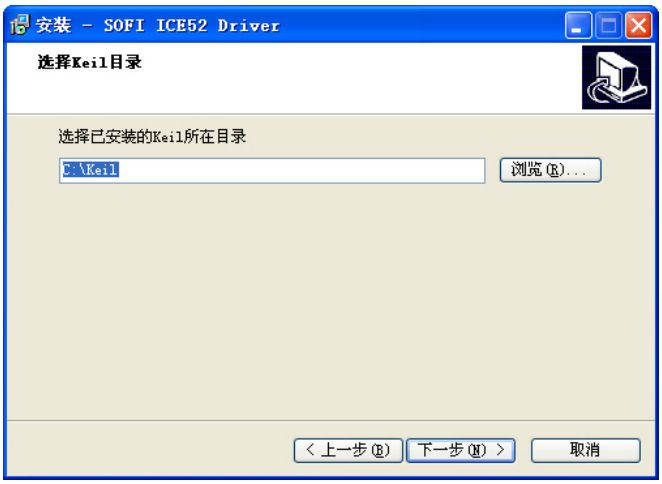


图 2.8 选择 Keil 目录

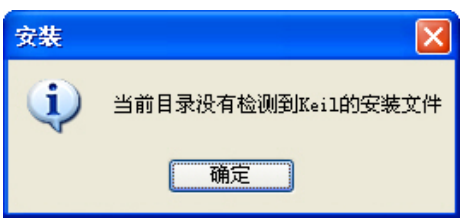


图 2.9 没有检测到 Keil

4）选择正确的 Keil 安装目录之后，便可开始接口程序的安装。

图 2.10 安装

2.2.3 安装 MEFlash 软件

MEFlash 是针对 ME800 系列的单片机开发实验仪而特别开发的一款增强型编程软件，具有类似通用编程器的所有操作功能。通常在开发和学习阶段可以直接使用 ICE52 所具有的独特的集成在 Keil 中的下载功能，直接对芯片进行编程下载，而无需使用该软件。如果需要单独烧写代码文件，或者是对 AVR 系列单片机进行编程才需要使用该软件。

MEFlash 安装程序位于产品光盘的 Software 目录下，文件名为 MEFlash_Setup.exe，运行该程序将出现如下画面，直接点击”下一步”，根据安装软件的提示安装即可。

图 2.11 安装 MEFlash

2.2.4 安装 USB 驱动

ME830 内置的 ICE52 仿真器/编程器采用 USB 接口进行通讯，在使用之前必须安装 USB 驱动程序。安装步骤如下（以 WindowsXP 为例）：

1）用随机 USB 通讯电缆连接 ME830 的 USB 插座和计算机 USB 口（注意：有些台式电脑的前置 USB 接口负载能力很差，可能导致实验仪不能正常供电，最好连接到计算机的后置 USB 端口），开启 ME830 上的电源开关(将 SW1 拨到 USB 端)；电脑会自动检测到新的硬件设备（图 2.12）；

图 2.12

2）操作系统屏幕上弹出“找到新的硬件对话框”，选择“否，暂时不(T)”，进入下一步（图 2.13）；

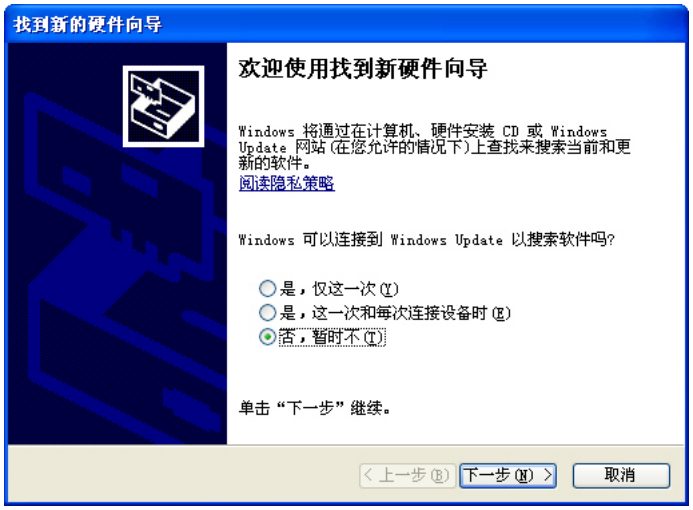


图 2.13

3）接下来选择“从列表或指定位置安装(高级)(S)” 然后点击下一步（图 2.14）；

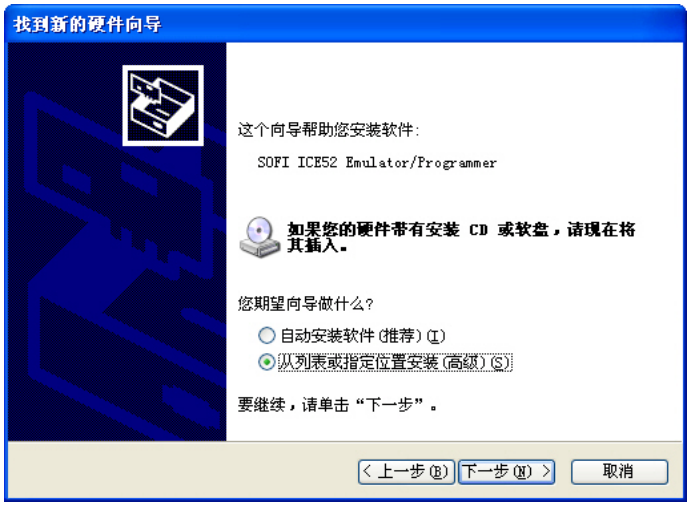


图 2.14

4）选择“在搜索中包括这个位置”，点击浏览，定位到产品光盘 Software\USB_Driver 目录（图 2.15）；

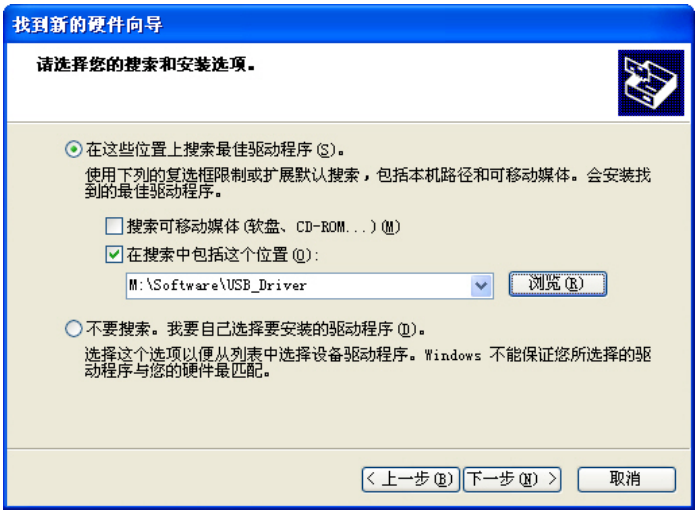


图 2.15

5）系统开始处理驱动程序的安装（图 2.16）；

图 2.16

6）安装过程中可能会提示驱动程序没有通过 Windows 的微标测试，此时请选择”仍然继续”（图 2.17）；

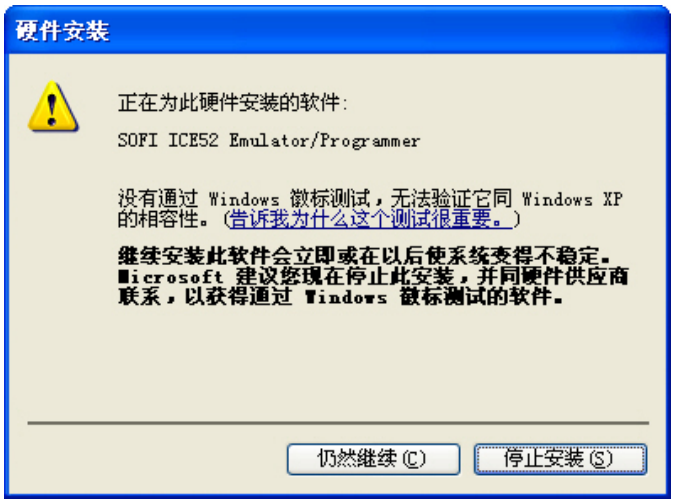


图 2.17

7）弹出“完成找到新硬件向导”对话框，点击“完成”按钮(图 2.18)；

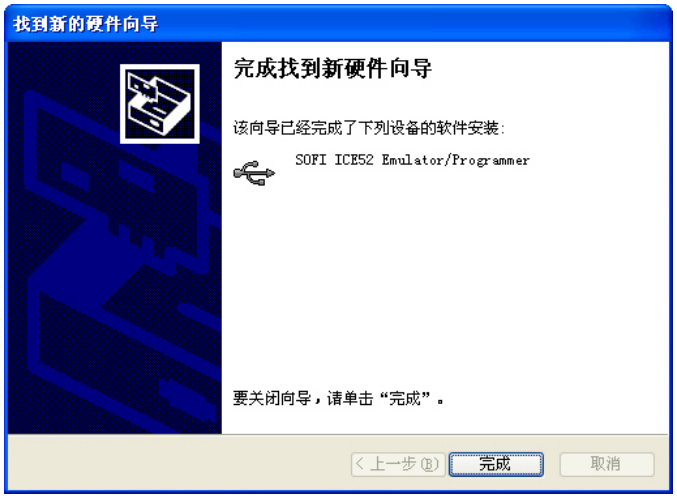


图 2.18

8）右下角弹出对话框“新硬件已安装并可以使用了”，表明 USB 驱动已成功安装。

运行 MEflash 软件，如果能正常联机，就可以正常使用 ME830 单片机开发实验仪了。

第三章 实验功能的使用

ME830 单片机开发实验仪可以直接支持多家主流公司（如 ATMEL，Winbond，NXP（Philips），SST，DALLAS）的 51 系列单片机的烧写和实验，也可以直接支持 AVR 系列芯片如 ATmega8515 的烧写和实验，所谓直接支持就是将单片机芯片插在主机锁紧插座上烧写(也称为编程，就是将程序代码写入单片机芯片的程序存储器中)和实验，如图 3.1 所示（注意芯片的缺口方向应朝向电源开关的方向）。用户也可以自制其它 AVR 系列单片机的最小系统，利用 ME830 的 ISP 下载功以及全开放的实验模块进行其它 AVR 单片机的实验，如 ATmega16，Atmega128 等。对于初学者，建议您先从 51 开始学起。

3.1 快速操作入门 - LED 闪烁实验

下面分别介绍如何在 ME830 上做 51，AVR 二种单片机的实验，二个实验均是使 ME830 上的 16 个发光二极管闪烁点亮；

实验前先按第二章内容安装好相应的软件和驱动程序。

连接好 ME830：将随机 USB 线的扁头端连接至计算机的 USB 接口，方头端连接到 ME830 的 USB 插座上（注意：如果是台式机请连接到计算机背面的 USB 端口，许多计算机的前置 USB 口只能提供很小的工作电流，可能导致 ME830 供电不足，引起工作不正常）；

将电源开关 SW1 拨到 USB 端，电源指示灯“PWR”（红色）亮，表明 ME830 已经正常接通电源，可以做实验了。

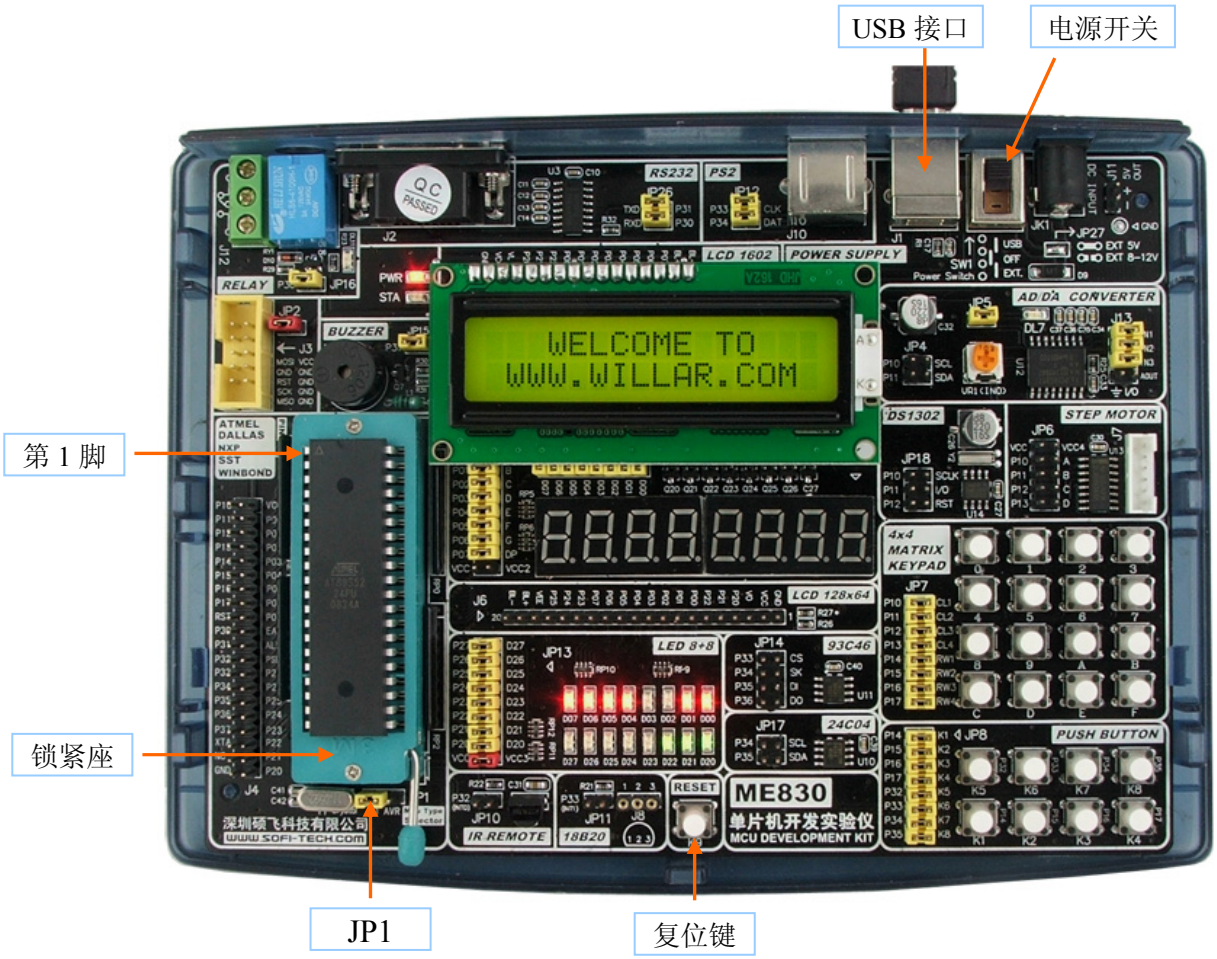


图 3.1

注意：做 LED 显示实验时必须将 1602LCD 拔下，否则 LCD 会干扰 P0 口的 LED，导致 P0 口 LED 显示异常！

3.1.1 51 系列单片机实验

单片机型号：AT89S51 或 AT89S52，DIP40 封装（也可以是其他兼容的 51 系列单片机，具体支持型号见软件列表）

1）将 JP1（MCU 类型选择跳线，位于锁紧插座的手柄旁边）的跳线帽短接在“51”的位置，JP13 的跳线帽全部插上；

2）将 AT89S52 单片机芯片放入 ME830 的锁紧插座，芯片缺口方向如上图所示（如果放反芯片，PWR 和 STA 两个 LED 会同时闪烁报警，请更正插放方向后按复位键即可恢复正常）；

3）启动 MEFlash 软件，正常打开后软件右下角会显示实验仪的型号和连接状态（有关 MEFlash 软件的详细使用说明见后续章节介绍）。如不能联机，请检查 USB 驱动是否正常安装，供电电压是否为正常的 5V；

4）在软件中点击“器件”按钮，选择型号“AT89S52”（型号不能选错）；

5）在软件中点击“加载”按钮，定位到产品光盘 Examples_A51\EX1_LED\LED.hex，点击“打开”，弹出“加载文件”对话框，按默认点击确定即可；

6）在软件中点击“擦除”按钮（注意：芯片擦空后发光二极管都在闪烁是正常现象），再点击“编程”按钮，编程完毕，即可看到 16 个发光二极管都在闪烁了；

如果编程完毕 LED 没有闪烁，请检查：

1）您是否正确加载了烧写文件（必须先选芯片再加载文件）；

2）芯片型号选择是否正确？程序写入后，用校验功能校验一次是否正确？如果校验始终不正确，请确认写入前是否有擦除芯片。检查锁紧插座手柄旁边的晶振 Y1 是否接触不良（为方便用户更换不同频率的晶振，晶振是直接插在一个圆孔插座上的，有可能因运输造成松脱或接触不良，请重新插牢再试）；

3）检查是否正确放置了芯片，方式是先放芯片，再压下锁紧座手柄锁紧；

4）1602LCD 是否已经取下，JP13 跳线组是否全部短接；

5）检查单片机芯片是否损坏，请用新的器件测试；

3.1.2 AVR 系列单片机实验

AVR 单片机是 1997 年由 ATMEL 公司研发出的增强型内置 Flash 的 RISC(Reduced Instruction Set CPU) 精简指令集高速 8 位单片机。AVR 的单片机可以广泛应用于计算机外部设备、工业实时控制、仪器仪表、通讯设备、家用电器等各个领域。

ME830 可以对直接 AVR 系列的 ATmega8515 进行烧写和实验，也可以自制其它 AVR 系列单片机的最小系统，利用 ME830 的 ISP 下载功以及全开放的实验模块进行其它 AVR 单片机的实验，如 ATmega16，Atmega128 等。下面已 ATmega8515L 芯片（DIP40 封装）为例介绍在 ME830 上实验 AVR 单片机的方法：

1）JP1 的跳线帽短接在“AVR”的位置，JP13 的跳线帽全部短接；

2）运行 MEFlash 软件，选择器件“ATmega8515L”；

3）点击配置按钮，弹出“器件配置”对话框，对熔丝位进行配置，Clock sourceand star-uptime 按下图设置，其他按默认设置即可：

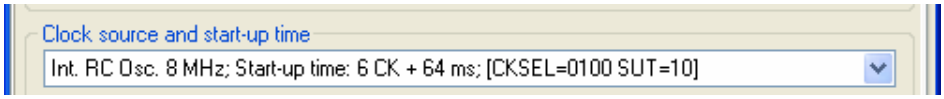


图 3.2

4）加载光盘 Example_AVR\EX1_LED\main.hex 文件

5）点击“自动”按钮，编程完成后，便可以看到 16 个发光二极管都在闪烁了。

特别注意设错熔丝位将锁死实验的芯片：

如将熔丝位选择了外部晶振或外部 RC 振荡，而没有接外部晶振或外部 RC 振荡，或者外接的振荡频率不匹配，导致芯片不能工作，这种情况，需要外挂相应晶体才能再次操作芯片，用户应尽量记起当时设错熔丝的情况，比如错误设置成了外部 3-8M 晶振，那么外挂一个 3-8M 晶振即可进行相应操作。

3.2 单片机开发流程简介

前面的两个实验都需要向单片机写入 Hex 格式机器码文件（当然也可以是 bin 或者其他特定格式），单片机才能运行。简单概括单片机开发过程就是通过相应的开发软件编写源程序，然后利用相应的开发软件编译得到单片机能识别的机器码文件，调试无错后，就可以将最终的机器码文件烧写到单片机，如果程序运行效果达到要求，就完成了单片机的开发过程。

对于 51 系列单片机，目前最主流的开发软件是 Keil C51，它集编辑，编译，仿真于一体，支持汇编,PLM 语言和 C 语言的程序设计，界面友好，易学易用。ME830 光盘中附带的 51 实验范例程序均是在 keil 环境下编写的。

对于 AVR 系列单片机，开发软件比较多，常用的有 AVR Studio，WinAVR（GCCAVR），ICC AVR 等。我们推荐用 C 语言来学习 AVR 单片机，ME830 光盘中提供的 AVR 例程均是 C 语言编写的，开发环境 WinAVR（20080610）

提示：MEFlash 是 ME830 实验仪配套的编程（烧写）软件，具有专业编程器的所有操作功能（读取、擦除、查空、加密、缓冲区编辑、自动编程等），但不能编写和编译源程序，MEFlash 软件的详细使用方法见第五章介绍。

对于 51 系列单片机，我们推荐您使用功能强大的 keil 软件，接下来第四章将以一个简单的 LED 流水灯程序为例来讲解如何用 Keil 开发软件编写和编译程序（生成 Hex 文件），以及如何在 ME830 系列单片机开发实验仪上进行仿真调试、烧写芯片和实验(ME830 具有独特的集成在 keil 中的编程下载功能，可以直接在 keil 中烧写芯片，详见第四章介绍）。

第四章 仿真功能的使用

单片机仿真器是在产品开发阶段用来替代单片机进行软硬件调试的非常有效的开发工具。使用单片机仿真器可以对单片机程序进行单步、断点、全速等手段的调试，在集成开发环境中检查程序运行中单片机 RAM、寄存器内容的变化，观察程序的运行情况。与此同时可以对硬件电路进行实时的调试。使用单片机仿真器可以迅速发现和排除程序中的错误，从而大大缩短单片机开发的周期。

ME830 单片机开发实验仪集成了伟纳（硕飞科技）最新研制的真正USB2.0 接口的仿真器ICE52，自主开发的keil仿真驱动协议，可以与keil开发环境完美结合，在仿真过程中不占用用户的任何资源。ME830 增强版本随机配有独立的POD52 仿真头，不但可以仿真实验仪内部资源，还可以仿真用户目标板，当作独立的仿真器使用，可以完全替代千元级别的专业仿真器。相比其他同类产品广泛采用是具有多项缺陷的SST公版仿真或者落后的USB转串口通讯方案，ICE52 仿真器有多项显著优势，详见最后附录：仿真功能对比。

ICE52 除了具有专业级的仿真功能，还具有编程（也称烧写）和 ISP 下载功能，不仅可以使用 MEFlash 进行编程操作（见第五章介绍），还可以在 Keil 中直接进行烧写芯片，当程序编译完成后，仅需要点击一下 KEIL 功能栏的 FLASH DOWNLOAD 按钮，即可将代码快速下载到单片机中，无需另外打开烧写软件。同类产品中只有 ME830/850 具备此先进功能，将会在学习或开发阶段给您带来极大的便利，大大提高您的学习和开发效率。

本章接下来以一个简单的 LED 流水灯程序为例来讲解用 Keil 开发软件编写和编译程序的过程，并详细讲解在 ME830 单片机开发实验仪上利用内置的 ICE52 专业仿真器进行仿真调试的方法。

使用前必须安装相应的开发软件，如Keil C51，ICE52 仿真驱动程序，USB驱动等，关于软件的安装请参考第二章。ME830 普通版配的是SST89E516 单仿真芯片，仿真方法请参考 附录 2 SST89E516RD仿真芯片的使用。

4.1 ICE52 仿真器的功能特点

- USB2.0 接口(真正的 USB 接口，非 USB 转串口)
- 集成编程下载功能，支持 Keil 的下载编程操作
- 可仿真 ATMEL，WINBOND，DALLAS，INTEL，SST，PHILIPS 等所有兼容 51 与 52 单片机
- 不占用用户资源，完全真实单片机所有端口特性
- 快速代码下载，极速单步操作
- 全新 Keil 仿真接口驱动，与开发环境完美接合
- 支持软件复位操作，无需手动复位硬件，即可进行连续代码下载与仿真操作

POD52 仿真头

- 支持脱机运行
- 微型置入式仿真适配头（POD52），彻底避免传统仿真排线带来的多种不稳定因素
- 可以仿真实验仪内部资源，也可以仿真用户目标板，当作一台独立的仿真器使用
- 支持标准的仿真操作，如全速，单步，跨步运行，断点的设置/禁用/取消，寄存器与变量查看
- 支持运行中暂停(夭折)功能
- 可仿真双 DPTR，PCA，ALE 禁用，SPI 接口，片内 768 字节扩展 RAM 等增强型 51 单片机资源
- 高达 63K 的代码仿真空间，支持外部 64K 扩展 RAM 仿真

4.2 第一个 Keil C51 程序

1）首先在硬盘上建立一个文件夹，比如在 E 盘建立一个名为“Demo”的文件夹，当然也可以是其他名字，为方便的程序的编写和调试，我们将调试过程中产生的文件都放在这个目录中。我们的产品光盘中提供了这个简单的范例程序，见光盘 Demo 目录。

2）启动 Keil C51 软件。您可以通过双击电脑桌面上的“Keil uVision3”快捷方式图标来启动。

3）执行 Keil C51 软件的菜单“Project | New Project…”，弹出一个名为“Create New Project”的对话框。输入工程文件的文件名，我们这里命名为“Demo”，选择你要保存的路径，我们这里保存到刚才建立的“Demo”目录中。

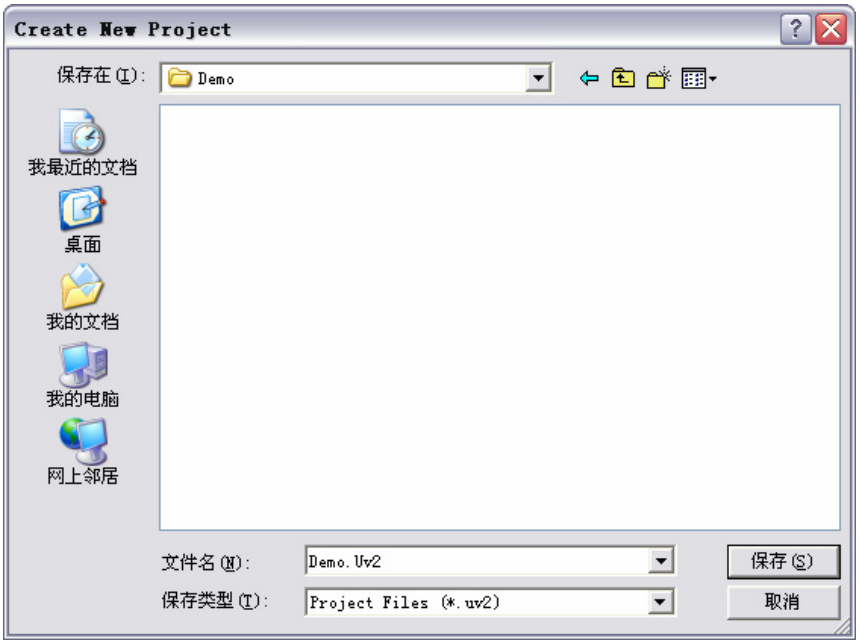


图 4.1

4）紧接着弹出“Options for Target ‘Target 1’。要求你为刚才的项目选择一个 CPU，我们选择 ATMEL 公司的 AT89S52。选择 AT89S52 后，右边一栏是对该单片机的基本说明，然后点击确定。见图 4.2。

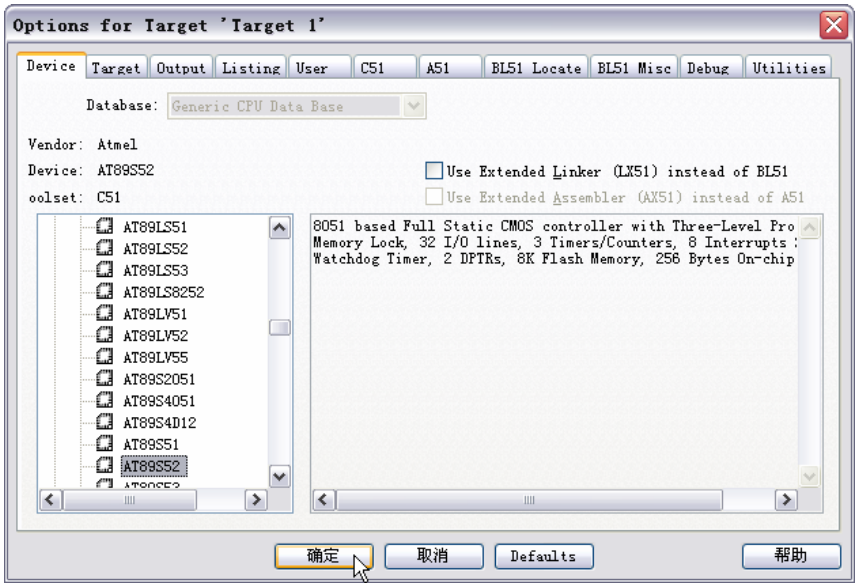


图 4.2

5）接下来弹出一个如图 4.3 所示的对话框。该对话框提示你是否要把标准 8051 的启动代码添加到项目中去。Keil C51 既支持 C 语言编程也支持汇编语言编程。如果打算用汇编语言写程序，则应当选择“(N)”。如果打算用 C 语言写程序，一般也选择“(N)”，但是，如果用到了某些增强功能需要初始化配置时，则可以选择“是(Y)”。在这里，我们选择“ (N)”，即不添加启动代码。

图 4.3

至此，一个空的 Keil C51 项目建立完毕。

6）执行菜单“File | New…”，出现一个名为“Text 1”的文档。 接着执行菜单“File | Save”，弹出一个名为“Save As”的对话框。将文件名改为”LED.ASM”， 然后保存，参见图 4.4。

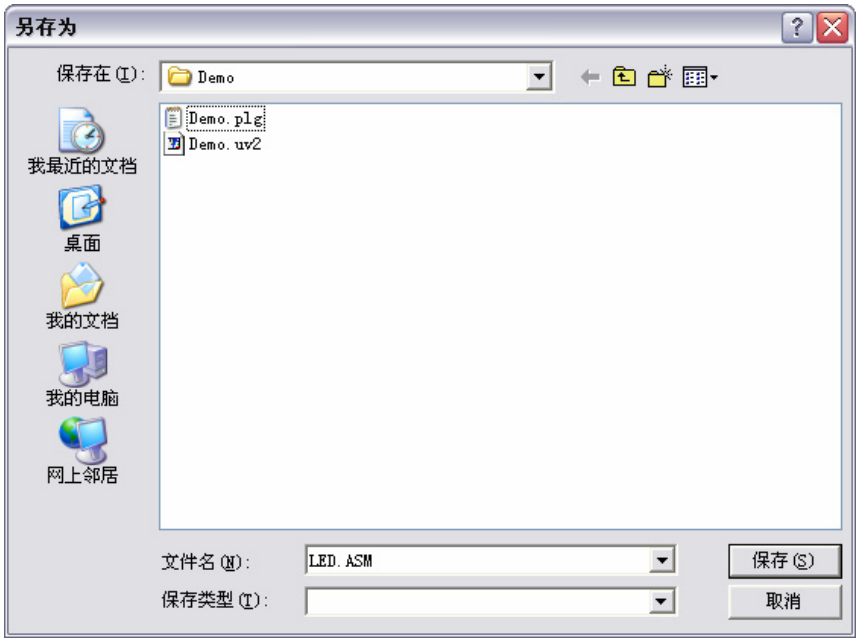


图 4.4

7）添加源程序文件到工程中。现在，一个空的源程序文件“LED.ASM”已经建立，但是这个文件与刚才新建的工程之间并没有什么内在联系。我们需要把它添加到工程中去。单击 Keil C51 软件左边项目工作窗口“Target 1”上的“＋”，将其展开。然后右击“Source Group 1”文件夹，会弹出如图 4.5 所示的选择菜单。单击其中的“Add Files to Group 'Source Group 1'”项，将弹出如图 4.6 所示的对话框。

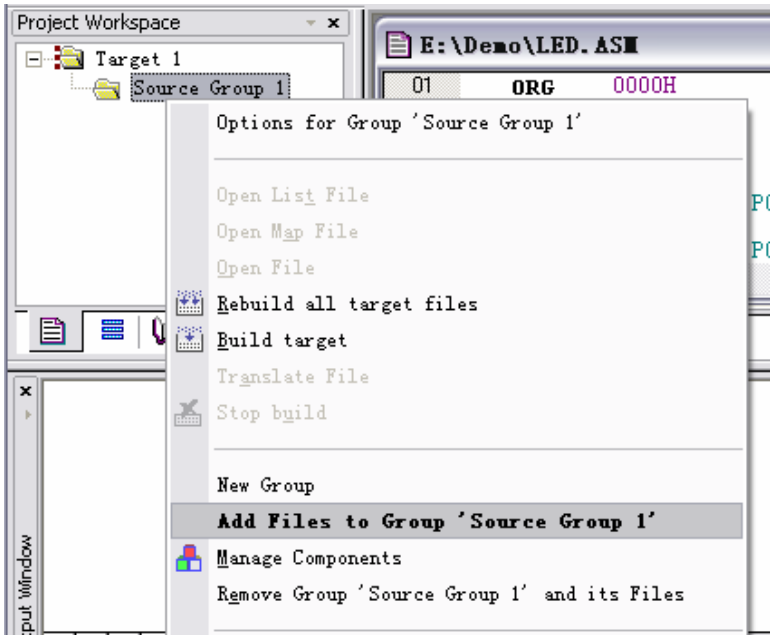


图 4.5

8）先选择文件类型为“Asm Source file (*.s*; *.src; *.a*)”，这时，对话框内将出现刚才保存过的“LED.ASM”。单击文件“LED.ASM”，再按一次“Add”按钮，最后按“Close”按钮。这时，源程序文件“LED.ASM”已经出现在项目工作窗口的“Source Group 1”文件夹内，可以单击左边的“ +”展开查看。

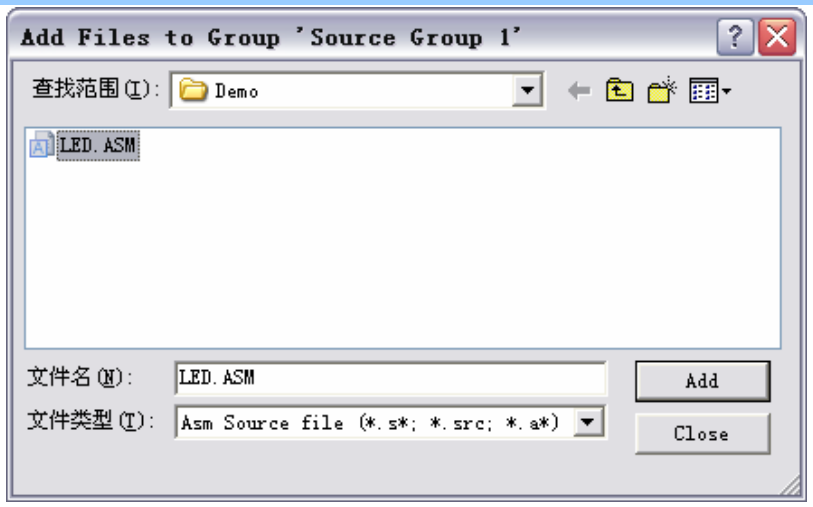


图 4.6

9）现在开始输入源程序。先最大化“LED.ASM”源程序窗口，然后请按程序清单 1.1 输入程序代码（注意输入代码时在英文状态下输入，勿使用中文标点符号，否则编译将会出错），输入完成后，别忘了点击一次保存按钮。

程序代码（光盘 Demo 目录中有此范例程序）：

```
        ORG    0000H
        LJMP    MAIN
        ORG    0050H
MAIN:
        MOV    A,#0FEH          ;赋初始值
LOOP:
        MOV      P0,A           ;初试点亮 P0.0 LED
        RL       A              ;左移
        LCALL    DELAY
        LJMP     LOOP
DELAY:                          ;延时子程序
       MOV         R7, #250
L1:
       MOV         R6, #250
L2:
       DJNZ   R6, L2
       DJNZ   R7, L1
       RET                      ;返回主程序

       END
```

程序清单 1.1

10）点击工具栏“Options for target”按钮，图 4.7 箭头所示

图 4.7

这时会弹出“Options for Target ‘Target 1’”对话框，如图 4.8

图 4.8

“Xtal”：定义 CPU 时钟，填写我们实际使用的晶振频率，假设是 11.0592M 的晶振，在“Xtal”后边框中填入“11.0592”。下面依次是编译的存储模式，程序空间大小等设置，均使用默认值即可。

点击 Output 选项，选中“Create Hxe File”（必须选中此项目，否则不能生成 HEX 文件），见图 4.9

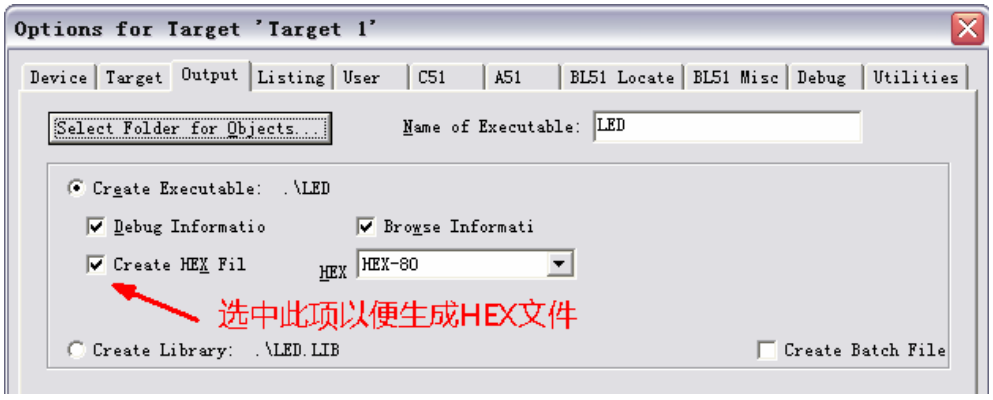


图 4.9

11）单击工具栏的按钮  编译当前源程序。编译结果会显示在输出窗口内。如果是“0 Error(s) , 0 Warning(s).”，就表示程序没有问题了（至少是在语法上不存在问题了）参见图 4.10。如果存在错误或警告，请仔细检查您的程序是否与程序清单 1.1 一致。修改后，再编译，直到通过为止。

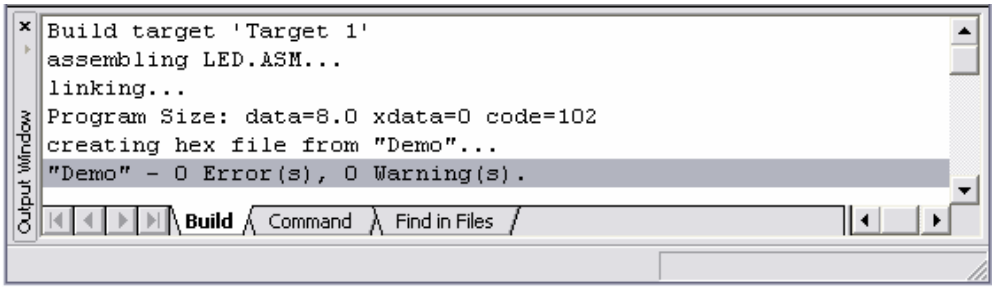


图 4.10

看看 Demo 目录，是否多了一个 LED.HEX 文件，这个就是程序编译后得到的烧写代码。

至此我们的第一个 Keil C51 程序已经完成，接下来的章节将讲述如何对该程序进行仿真以及将该程序下载到实验芯片中去运行。

4.3 仿真调试

4.3.1 仿真器的硬件连接

1）仿真内部资源

用随机的 10PIN 电缆将仿真器适配头（POD52）连接到 ME830 的 ICE/ISP 接口（J3），将仿真头插入到目标板的锁紧座上锁紧，用 USB 电缆连接好计算机与 ME830，开启 ME830 的电源开关(将开关拨到 USB 一边)。如下图所示:

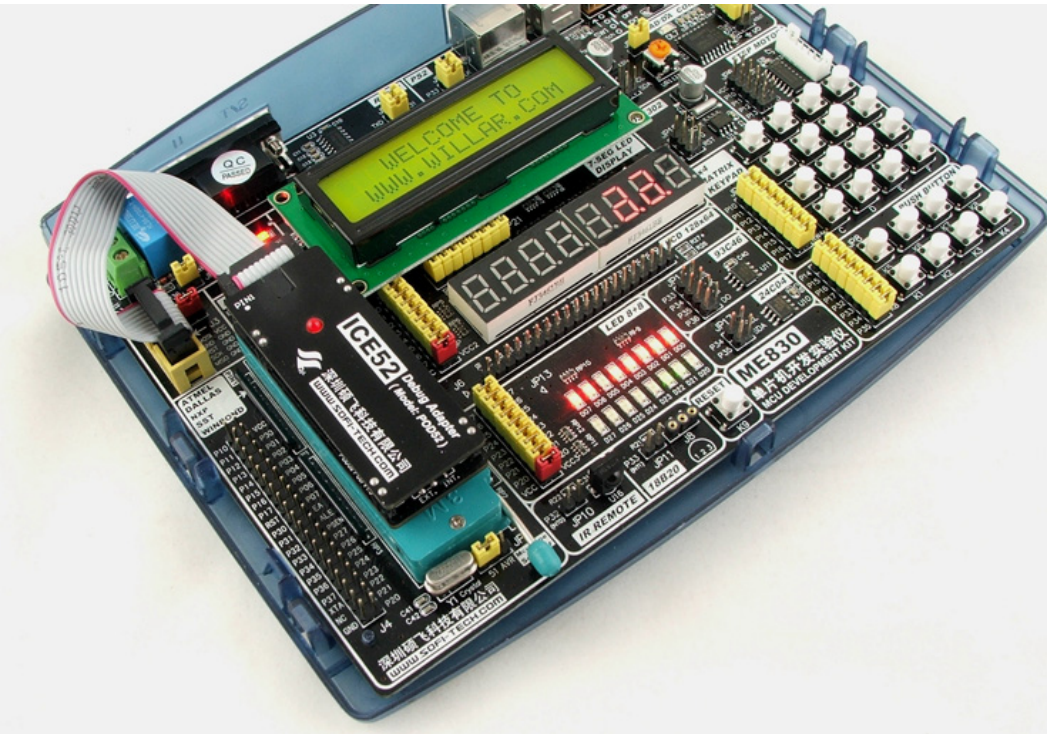


图 4.11　仿真头连接示意图

仿真头 POD52 上有一个晶振选择开关，用于选择 POD52 是使用内置晶振还是用户板上的晶振。将开关设置到 INT 选择 POD52 内置的 11.0592MHz 晶振。拨到 EXT 端选择 ME830 实验 CPU 的或者是用户板上的晶振。

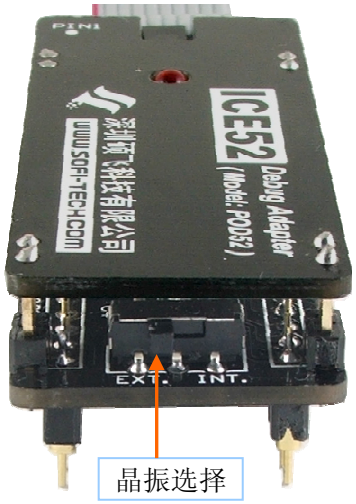


图 4.12　仿真头晶振设置

2）仿真外部目标板

ME830 内置的 ICE52 仿真器支持对外部目标板的仿真，可以作为一台独立的仿真器使用，可以完全替代专业的仿真器调试产品。直接将仿真头插入到外部目标板的 CPU 插座上即可，如图 4.13 所示。

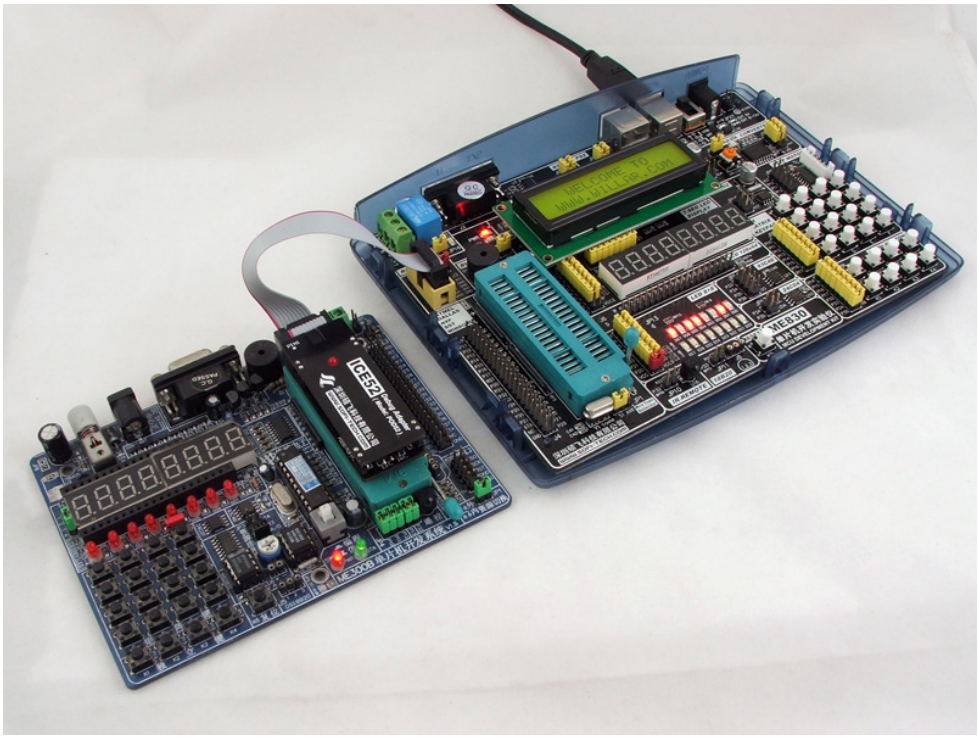


图 4.13　仿真外部目标板

对外部仿真时，短接 ME830 实验仪上 JP2(VTG)跳线，可对外部目标板进行供电。

4.3.2 仿真器的软件设置

1）紧接着 4.1 章节建立的 Keil C51 程序（也可以打开一个已经建立好的其他 Project 文件来调试）

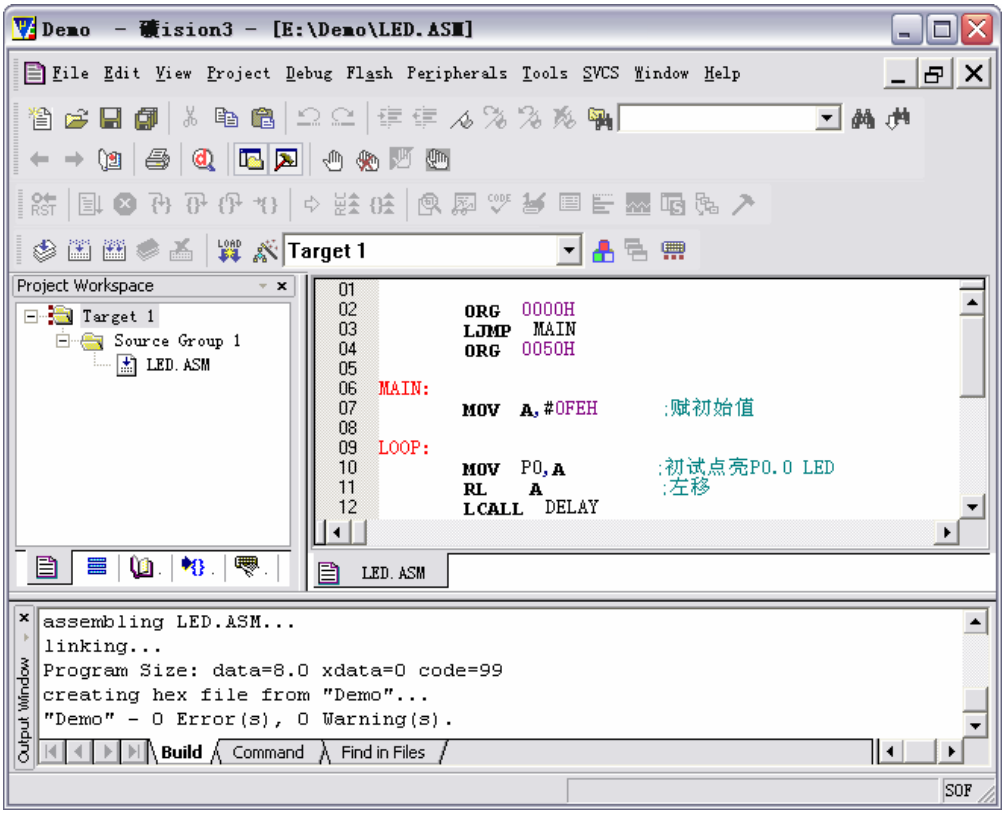


图 4.14 Keil 开发环境

2）点击 Keil 的工具栏按钮  ，在弹出的对话框中选择“Debug”选项页。按图 4.15 中的设置步骤进行

设置。在下拉框中如果没有发现 SOFI ICE52 Emulator/Programmer 选项，可能是没有安装 ICE52 的仿真驱动程序，请参考第二章的仿真驱动程序安装部分。其他设置请保持与图 4.15 一致.

图 4.15 Keil 仿真设置

3）接着点击 Debug 选项页中右上角 Settings 按钮，软件会弹出 ICE52 的仿真设置对话框，如图 4.16 所示。对于 ICE52 的仿真部分只有唯一选项，即是否开启“暂停功能”。当开启该功能之后，用户不可以使用代码区域中 0x003b 处的三个字节，这个三个字节需要被仿真器占用。你需要在程序中进行相应设置，具体方法见 4.3.3 章节 3）暂停功能的介绍。光盘中所有示例程序已经对 0x003b 处做了保留，因此我们可以勾选该功能。

图 4.16 ICE52 设置

回到“Options for Target…”，按“确定”按钮，完成设置。

4）进行完以上设置后，编译该项目，检测编译结果。编译无误后，便可进行仿真功能操作。点击工具栏按钮  启动调试模式，代码便开始下载到仿真头中，下载完成后，Keil 的信息输出窗口显示如图 4.17 所示。

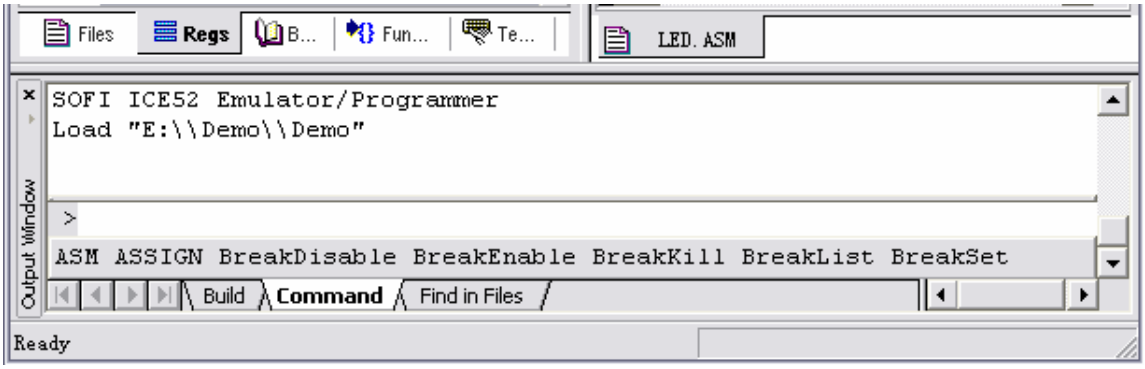


图 4.17　仿真界面

接下来就可以点击 Debug 菜单下的仿真命令或者相应的工具栏按钮，即可进行仿真操作. 包括全速运行，单步运行，跨步运行，断点的设置/取消等等。观察/修改变量值、观察/修改存储器数据等操作。具体的仿真调试方法见第四章。

如果 Keil 弹出如图 4.18 所示信息框，标明仿真器与仿真头通讯失败！请按提示信息的内容检查。另外请参照前面图 4.15 查是否选择了正确的 Keil 驱动，如果 USB 驱动未正确安装，也会导致连接失败。

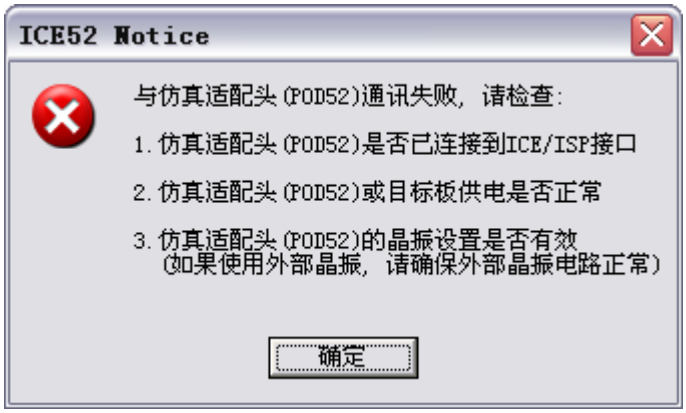


图 4.18

4.3.3 调试程序

1）断点设置与取消

ICE52 最大可以支持 19 个固定断点(即地址断点)和一个临时断点(用于跨步运行)。断点的设置与取消操作比较简单，仅需要将光标移到相应的源代码行，然后点击的 Debug 菜单中的 Insert/Remove Breakpoint 或者按一下键盘的 F9 键即可。在断点使能后，在当前源代码行的前面会显示一个红色的方块标志，如下图所示.

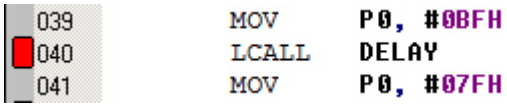


图 4.19　断点设置

用户设置的地址断点不可以超过 19 个，如果断点超过该限制，在开始全速运行或跨步运行时，Keil 会在 Command 窗口中显示如下图所示的提示信息:

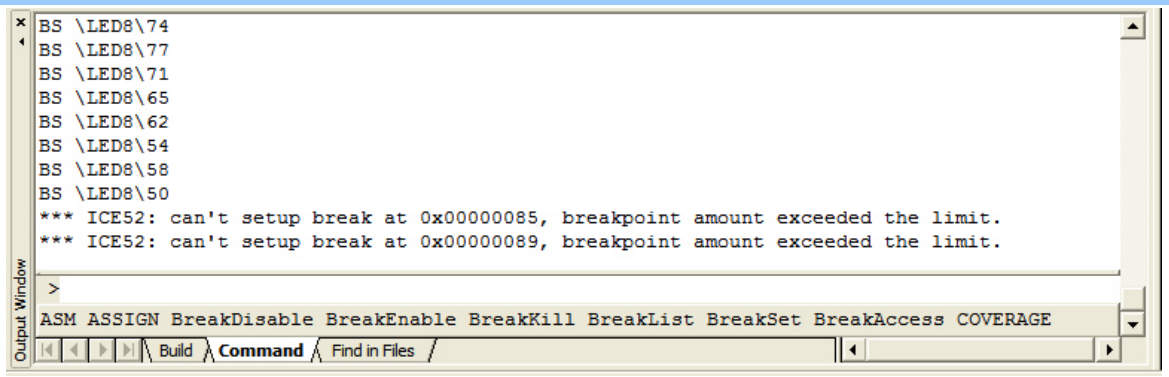


图 4.20 错误提示

超过数量的断点将自动被禁用，禁用的断点会显示成灰白色，如下图所示：

图 4.21 禁用的断点

2）仿真运行

进入仿真模式之后，可以通过菜单 Debug 已经相应的工具栏按钮执行相应的运行操作。

Debug 菜单	工具栏按钮	功能
Run		全速运行
Step		单步运行
Step Over		跨步运行
Run to Cursor line		运行到光标所在行
Reset CPU		复位
Stop Running		暂停(需开启暂停功能)

3）暂停功能

暂停功能也叫夭折功能，在仿真器全速运行过程中，可以点击 Keil 工具栏的按钮，或者是菜单【Debug】→【Stop Running】将仿真器暂停下载，暂停后可以再次点击全速/单步运行和跨步运行。

此功能是一个可选项，需在 ICE52 对话框中进行设置，只有勾选此选项后，该功能才有效。

开启该功能后， 仿真器需要占用代码空间的 003BH 的三字节空间，用户的代码不可以使用该区域。对于 C 语言程序可以使用如下的代码将该空间进行保留。

```
char code reserve [3] _at_ 0x3b;          // 保留三字节
```

对于汇编语言， 可以直接将主程序跳过该区域， 如将主程序定位在 0050H 之后，代码如下：

```
        ORG 0000H
        LJMP    MAIN

        ORG 0050H
MAIN:
        ;主程序开始
```

该功能借助 51 的单片机的内部的中断功能实现，所以在用户的代码中不可以关闭中断功能（例如将 EA 设为了 0），否则仿真器运行后，将无法暂停。软件会显示如下所示的提示：

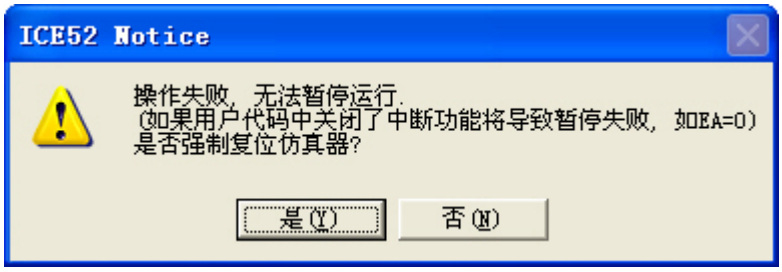


图 4.22 ICE52 Notice

4）仿真扩展 RAM

仿真监控 CPU 是一个增强型的 51 单片机，片内已带有 768 字节的扩展 RAM，在默认模式(上电复位后)，MOVX 指令对地址范围 000H~2FFH 的读写操作，将直接访问片内的扩展 RAM，当地址超过 2FFH 时，才会访问片外扩展的 RAM。

如果需要直接访问片外 RAM，可以将片内的扩展 RAM 禁用，通过设置 AUXR 寄存器的 EXTRAM 位为 1 即可。

AUXR 寄存器 EXTRAM 位	MOVX 访问地址 00H~2FFH	MOVX 访问地址 300H~FFFFH
EXTRAM=0 (芯片默认模式)	访问片内扩展 RAM	访问片外扩展 RAM
EXTRAM=1	访问片外扩展到 RAM	

5）脱机运行

POD52 仿真头支持脱机运行，直接将 POD52 仿真头插入目标板的 CPU 插座，仿真头便会自动运行上一次的调试过的程序。

脱机运行的程序也可以通过 Keil 的 Flash Download 功能重新下载. 在 ICE52 的设置对话框中的下载编程设置区内选中“下载到仿真适配头 (POD52)”，然后执行 Keil 的 Flash Download 命令，Keil 自动将程序下载到仿真适配头的 CPU 内，然后开始全速运行，运行效果与一个真实的 CPU 完全相同。

6）退出仿真

点击工具栏按钮 即可停止调试，退出仿真状态！

4.3.4 在 keil 中下载运行

ME830 单片机开发实验仪具有独特的集成在 Keil 软件的下载功能，在学习和开发阶段，除可以直接在 Keil 开发环境中完成源程序的编写、编译、仿真等过程，还可以在 keil 中烧录 51 系列单片机芯片，无需在开发软件和烧写软件间来回切换，可大大提高学习和开发效率。本节将介绍如何在 keil 中实现对芯片的下载编程。

注意：ME830 实验仪在 keil 中只支持 51 系列单片机的编程下载。如果要烧写 AVR 系列单片机，请使用配套的 MEFlash 编程控制软件（见第五章内容）。

在 keil 中烧写芯片必须在已经建立的 Project 文件中进行，另外必须关闭 MEFlash 软件。下面以我们前面 4.2 章节建立的 KEIL C51 程序为例，介绍在 Keil 中烧写一片 AT89S52 芯片的步骤。

1）点击工具栏按钮 ，然后点击 Unitilies 选项页，按图 4.23 中的设置步骤进行设置。

图 4.23 Keil 编程设置

2）选中 Use Target Driver for Flash Programming 选项，在列表框中选择 SOFI ICE52 Emulator/Programmer。然后点击 Settings 按钮，打开 ICE52 的仿真/编程设置对话框，如图 4.24 所示。在对话框的下载编程选项中分别选择合适的器件厂商和芯片型号。我们现在是对 ME830 锁紧座上的 AT89S52 编程，选择型号 AT89S52。如果我们需要对外部目标板上的芯片下载程序，请选择带有“@ISP”后缀的型号。外部 ISP 的使用方法见第五章介绍。

图 4.24 ICE52 设置

3）将产品附带的 AT89S52 芯片放置到 ME830 的锁紧座上，如图 4.25 所示。芯片缺口朝向串口座的方向，注意不要插反。

另外注意跳线的设置必须正确：将 JP1（MCU 类型选择跳线，位于锁紧座的手柄下方）的跳线帽短接在“51”的位置，JP13 的跳线帽全部插上，因我们现在实验的是流水灯程序，不能插上 1602LCD，否则会造成干扰。

图 4.25　实验芯片放置

4）点击 Keil 工具栏的  按钮，或者是菜单【Flash】→【Download】即可将程序烧写开发板的实验芯片(AT89S52)内。Keil 的窗口内会显示如图 4.26 所示的内容。下载完毕，目标板上单片机便开始运行写入的程序（注意程序必须正常，如果不能正常运行，可以直接使用光盘中的范例程序）。

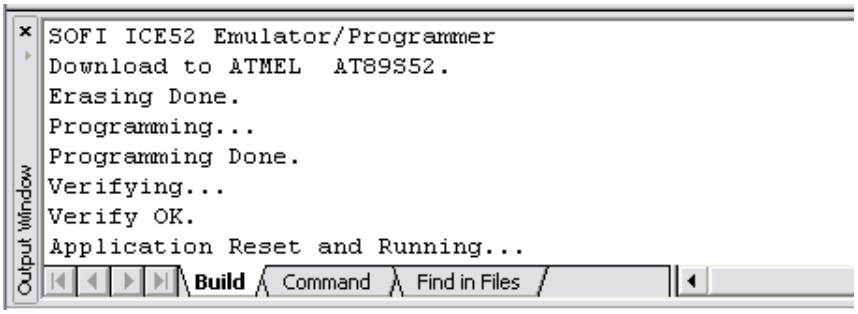


图 4.26 Keil 编程信息

如果下载失败，keil 会弹出以下错误提示框，见图 4.27

图 4.27

下载失败的原因可能有：

芯片损坏，芯片接触不良，锁紧座晶振接触不良，同时开启了 MEFlash 软件(在 keil 中下载时必须关闭 MEFlash 软件)。如果是 SST，NXP(Philips)，winbond 单片机，还需要写入 ISP 固件后才能在 ME830 中使用，见第五章的内容。

第五章 编程/ISP 下载功能的使用

ME830 单片机开发实验仪集成的 USB2.0 接口的仿真/编程器 ICE52，除了具有独特的嵌入到 keil 的烧写功能外（可以直接在 keil 环境下点击点击 Keil 工具栏的  按钮，便可将代码下载到单片机芯片中运行，非常方便快捷，具体使用方法见前面 4.3.4 章节），如配合硕飞科技专门开发的编程控制软件 MEFlash ，可以对实验仪锁紧座上的单片机编程，也可以通过预留的 ISP 接口对用户目标板进行 ISP 下载编程，支持 51 系列和 AVR 系列单片机的编程，具有专业编程器的所有操作功能（读取、擦除、查空、加密、缓冲区编辑、自动编程等）。此功能将在产品开发过程中给您带来极大的便利。

本章将详细介绍利用 MEFlash 软件进行编程（烧写）和对外部目标板进行 ISP 下载的方法：

5.1 支持器件列表

器件厂商	器 件 型 号			
ATMEL	AT89S51	AT89S52	AT89S53	AT89S8252
	AT89S51@ISP	AT89S52@ISP	AT89S53@ISP	AT89S8252@ISP
	ATmega8515	ATmega8525L	ATmega8515@ISP	ATmega8515@ISP
	ATmega8@ISP	ATmega16@ISP	ATmega32@ISP	ATmega64@ISP
	ATmega128@ISP	ATmega48@ISP	ATmega88@ISP	
	ATmega168@ISP	ATmega8L@ISP	ATmega16L@ISP	
	ATmega32L@ISP	ATmega64L@ISP	ATmega128L@ISP	
	ATmega48L@ISP	ATmega88L@ISP	ATmega168L@ISP	
	ATmega48V@ISP	ATmega88V@ISP	ATmega168V@ISP	
DALLAS(MAXIM)	DS89C430	DS89C440	DS89C450	
NXP(Philips)	P89V51RB2	P89V51RC2	P89V51RD2	
SST	SST89E52	SST89E54	SST89E58	SST89E516 SST89E564
WINBOND	W78E58B	W78E516B		

说明:

1）上表中的蓝色部分的器件为 AVR 系列的单片机，该部分芯片不能在 Keil 中进行下载(Keil 仅支持 51 系列单片机)，必须使用 MEFlash 专业编程软件来进行烧写；

2）带@ISP 后缀的器件采用 ME830 主板上的 ISP 接口进行下载(即外部 ISP 编程)，不能直接放在主板的锁紧座上烧写。同样，不带@ISP 后缀的器件只能放在主板的锁紧上烧写，不能通过 ISP 接口进行烧写；

3）新的 NXP(Philips)、SST 系列芯片出厂时已经内置了 ISP 固件，可以直接在 ME830 中编程和下载。如果是旧的的或者已经被擦除了 ISP 固件的芯片需要先用其他编程器写入 ISP 固件才能在 ME830 上使用（重写 ISP 固件的方法见后续章节的说明）；

4）WINBOND 系列单片机（W78E58B，W78E516B）出厂没有固化 ISP 固件，同样需要预先写入 ISP 固件才能在 ME830 上使用（写入 ISP 固件的方法见后续章节的说明）；

5.2 MEFlash 软件功能介绍

1）软件的主要特点

- 中文/英文双语支持,可自由随时切换
- 编程操作完善，包含擦除，查空，校验，编程(写入)，读取，读写熔丝位，加密等等
- 支持自动编程操作，一步完成多项操作，可设置操作内容.
- 自动检测文件修改并重加载
- 支持跳过无效数据，以加快烧写速度
- 可选的编程区域设置(对于有多个存储区的芯片)
- 完善的缓冲编辑功能，支持键盘输入修改，支持复制/填充以及逻辑运算等等
- 最近器件列表
- 最近文件列表

2）软件主界面

以 ME800 系列控制软件 V1.00 版为例，连机正常运行后软件主界面如下:

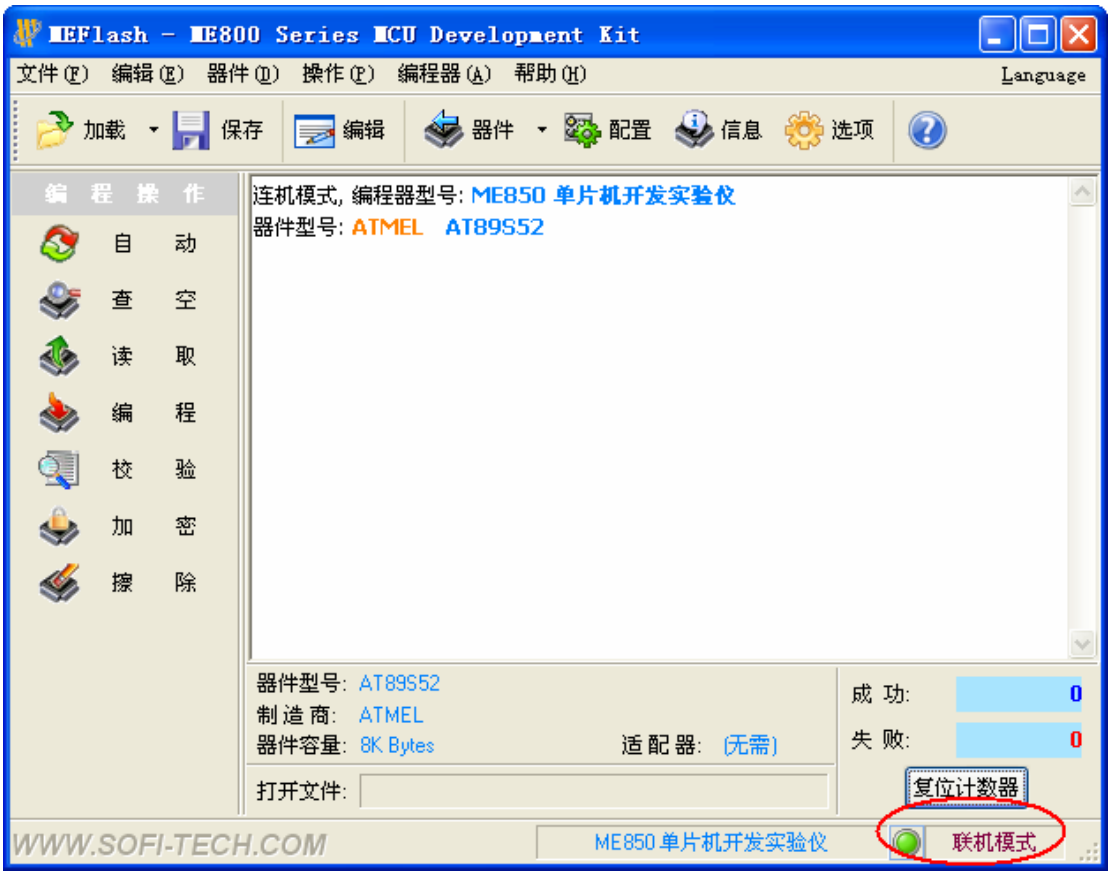


图 5.1 软件主界面

主界面从上到下，从左至右分别是：主菜单栏（上一）、工具栏（上二）、快捷工具栏（左）、操作信息窗口（右）、器件信息窗口和统计表窗口（右下），状态栏（最下）。

MEFlash 软件的大部分功能可以通过单击“快捷工具栏”按钮实现，少部分功能则需要使用菜单操作。部分主要菜单和工具栏按钮功能介绍如下：

主菜单栏：

最后一个菜单 Language 用于设置软件语言，可选择英文和简体中文

工具栏：

编辑 点击此按钮打开“缓冲区编辑”窗口，缓冲区用于保存从文件加载的数据，或从芯片读取的数据。数据缓冲区可以进行编辑。

信息 器件信息除了包含器件的基本信息，例如:厂商,型号，容量,引脚数,适配器等等,还有芯片的放置说明图。对于 ISP 芯片则包含其 ISP 下载连接图。

选项 点击此按钮打开“操作选项”窗口，如下图（不同的芯片有所不同）。可以设置是否自动加载修改过的文件、检查器件 ID、忽略 0xFF 数据段，另外可以设置自动编程选项。

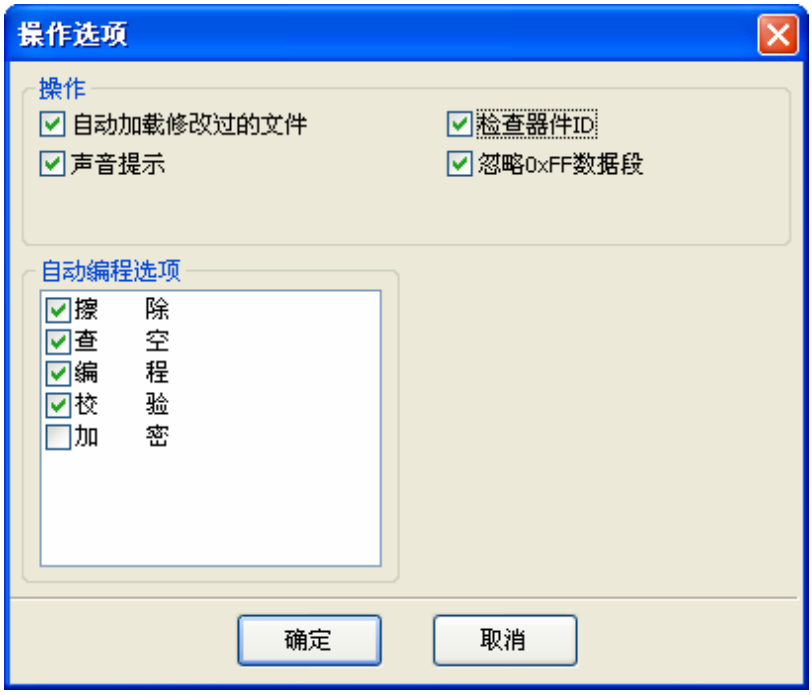


图 5.2 操作选项

提示： 1）选择忽略 0xFF 数据断可以加快烧写的速度
2）执行“自动”编程功能时必须设置自动编程选项

5.3 烧录器件的方法

ME830 可以用以下二种方式对器件进行编程（也称烧写或烧录）：

5.3.1 使用 ME830 主机直接烧录

芯片直接插在 ME830 的主机锁紧插座上烧录，此方式无需其他附件，可支持 40pin 的 ATMEL，Winbond，NXP（Philips），SST，DALLAS）的 51 系列以及部分 AVR 单片机，具体支持型号见软件列表。

1） 硬件准备

在确保安装好 USB 驱动和 MEFlash 软件之后，用 USB 电缆连接好 ME830 实验仪，此时实验仪上的红色电源指示灯 PWR 应点亮，绿色状态指示灯应熄灭。

放置芯片到实验仪的锁紧座上，注意放置方向：芯片的第 1 脚应对应锁紧座标识有 PIN1 的地方。

2） 选择器件

单击“器件”按钮，弹出“选择器件”窗口。注意：软件中器件尾缀不含“@”的表示直接用 ME830 主机烧录，器件尾缀是“@ISP”的表示该器件需要通过 ISP 下载接口下载。请根据需要找到对应芯片型号或直接输入器件型号名搜索，快速选定器件。

3）将数据装入缓冲区

选择好芯片型号后再将待烧写的数据装入软件缓冲区，数据装入缓冲区有两个途径：

- **从文件加载**

选择主菜单“文件”下的“加载文件”或单击工具栏“加载”按钮，在弹出的文件选择对话框中选择准备烧写的文件，然后点击“打开”，随后弹出“加载文件”对话框，可通过此对话框选择相应的文件类型和起始地址，没有特殊要求一般按默认即可。

- **从母片读取**

选择器件后，放置好母片，在快捷工具栏中单击“读取”按钮，即可将芯片中的数据复制到缓冲区，这些数据可以存盘共日后使用。注意：有些器件没有读出功能，或者该器件被加密，就无法从母片中读出数据。

4）配置选项设置

选择主菜单“器件”下的“器件配置”或单击工具栏“配置”按钮，弹出“器件配置”对话框，可以对器件的加密位和配置字（Dev.Config）进行设置。对于有配置字的芯片如 AVR 单片机必须在烧录前进行配置字设置，才可以保证烧录后的芯片可以在用户的目标系统上工作。

5）编程

正确放置好待烧录芯片后，操作步骤如下：

- 查空（检查芯片的存储单元是否为空，如果芯片是新的，一般可略过）
- 擦除（如果芯片不是空的，须擦除一次。如是空的，可略过）
- 编程（此步骤将程序写入芯片的程序存储器）
- 校验（这一步是必须的，只有校验成功，才可以认为芯片烧录无错误）
- 写熔丝位（如果是 AVR 芯片可能需要写熔丝位操作，需先根据需要正确配置熔丝位）
- 如果芯片需要加密，在校验之后执行“加密”，需要正确设置配置选项才有效

以上步骤可以设置好自动编程选项后，单击“自动”按钮或按“F5”一次完成所有操作

5.3.2 ISP 下载功能的使用

ME830 预留了一个用于连接用户板的 ISP 接口 J3,使得 ME830 单片机开发实验仪还具有 ISP 编程器的功能。可以通过随机的 10Pin 的 ISP 下载连接头对用户的其他目标板进行在系统编程,可以支持 AT89S，AT89LS，大部分 AVR 系列的可 ISP 器件的下载。

所谓 ISP，即（In-System Programming）在系统可编程，指电路板上的空白器件可以编程写入最终用户代码， 而不需要从电路板上取下器件，已经编程的器件也可以用 ISP 方式擦除或再编程。此功能无论是对于实验学习还是产品调试都是非常的方便，特别是重新烧写电路板上的贴片芯片，ISP 目前是不二之选。

1） ISP 下载线的连接

ME830 主板上的 ISP 下载接口引出方式如图 5.3 所示（元件面俯视图）。

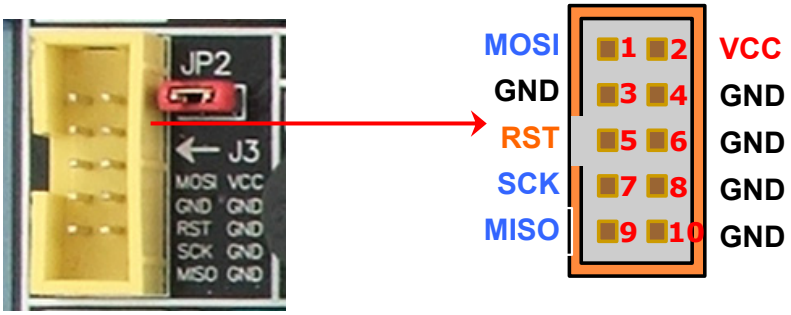


图 5.3 ME830 主板 ISP 接口定义

你只需在目标板上预留一个双排 5pin 的排针作为 ISP 下载接口，将目标板上单片机的 ISP 相关的 6 个管脚（MOSI、SCK、RESET、MISO、VCC、GND，管脚定义请参考相应单片机的 Datasheet）连接到预留的 ISP 下载接口。确认目标板的 ISP 接口连接无误，即可通过随机 10Pin 的 ISP 连接线连接到 ME830 的 ISP 下载接口如下图所示连接起来。JP2 跳线用于选择是否给目标板供电，短接后，ME830 向目标板提供 5V 电源，最大电流 300mA。如果短接 JP2 后，ME830 上的 PWR 和 STA 两个 LED 同时闪烁，说明是目标板引起 ME830 过载保护了。请检查目标板是否短路，ISP 接口接线是否错误。如果目标板本身工作电流较大，则不能通过 ME830 向目标板供电，请取下 JP2 上的跳线帽后，给目标板单独供电。

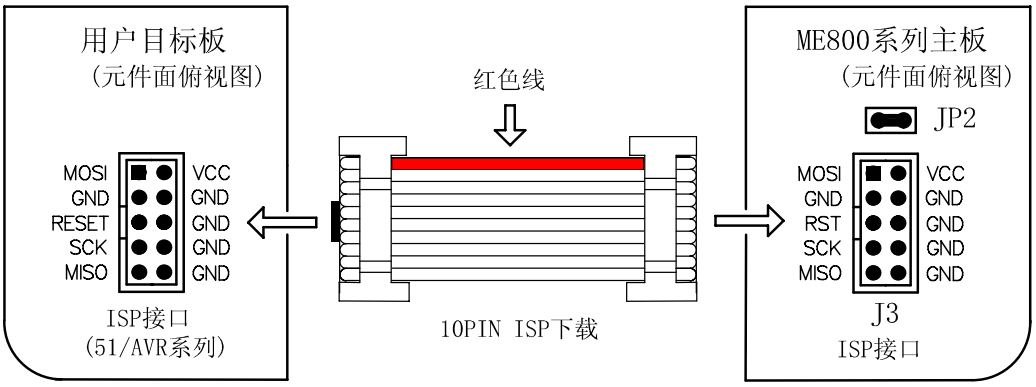


图 5.4 ISP 下载连接示意图

下图是 ME830 对外部目标板（目标芯片是 AT89S52）的下载的实物照片：

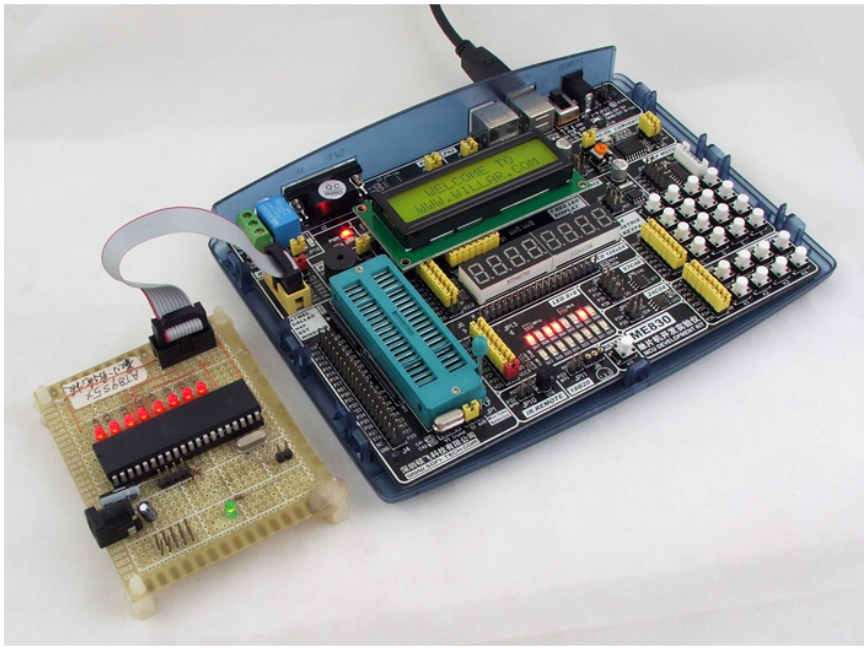


图 5.5 ISP 下载实物照片

2）下载操作

连接无误，便可进行 ISP 下载编程操作了。软件操作和在 ME830 主机上烧写方法唯一不同的是需要在器件型号中需选择“xxx@ISP”，即器件型号尾缀必须是含“@ISP”的，表示该器件使用 ISP 方式烧写。烧写步骤请参考 5.2.1 章的内容。

提示： 1）用户目标板必须能正常运行，否则将不能正常进行 ISP 下载，MEFlash 软件会提示“初始化器件失败”。
2）对外部目标板下载时主机锁紧座上不能放芯片
3）软件中选择的器件型号必须带有“@ISP 后缀”

5.4 器件编程特殊说明

ME830 实验仪支持多个厂商的单片机,所有的单片机均使用单片机的 ISP 功能进行编程和下载,所以在 ME830 上使用这些芯片必需保证其 ISP 功能正常。部分芯片的 ISP 功能是固有的，例如 ATMEL 公司的 AT89S51/S52，这些芯片在 ME830 上可以直接使用，无需任何预先设置。但是有些芯片的 ISP 功能可能需要进行预先设置，才能保证在 ME830 上正确使用。

5.4.1 ATMEL 单片机

ATMEL 单片机主要指 AT89S51/S52/S53/S8252/S8253 等芯片，该系列可放在 ME830 的主板锁紧座上进行烧写，也可以通过 ME830 主板的 ISP 接口对用户的目标板进行下载(即外部 ISP 编程)。

无论是放在锁紧座上，还是对用户的目标板进行下载，芯片都必须外接晶振，范围 3~24MHz。ME830 主板上默认的晶振是 11.0592MHz(出厂配置，用户可以根据需要更换)

在对用户目标板的芯片进行下载时，还必须保证芯片的有可靠的供电，供电可以是用户的目标板自供电也可以通过 ME830 的 ISP 接口进行供电。在使用 ME830 的 ISP 接口进行供电时，需短接 ME830 主板上的 JP2 跳线。

5.4.2 SST 单片机

该系列单片机包含两个程序存储区，分别为 BLOCK0 用户程序区，和 BLOCK1 启动代码程序区。要在 ME830 系列开发板(实验仪)上实现此类芯片的编程下载操作，BLOCK1 必须内置有 ISP 启动下载代码，并且单片机的配置位 CS0/CS1(SST89E516 和 SST89E516 只有 CS0)必须进行相应的设置，使单片机复位后从 BLOCK1 启动。

该芯片在出厂时，通常已经烧录有 BSL 并配置好相应的配置位。即芯片如果是全新没有使用过的，可以直接在开发板(实验仪)上使用。

如果是已经使用过的芯片，其 BSL 和配置位很有可能已经被破坏，此时必须使用其他商用编程器进行恢复。产品光盘中有该 BSL 的代码文件，位于 BOOTLOADER\目录下，文件名为 F51MBLL5.BIN。将该代码烧录到 BLOCK1 区，然后设定正确的配置位即可。设置方式请参考下表

BSL 代码及配置位设置表

芯片型号	BSL 文件	缓冲区开始地址 (BLOCK1 映射地址)	配置位设置
SST89E52	F51MBLL5.BIN	0xE000	SC0=1，CS1=1
SST89E54	F51MBLL5.BIN	0xE000	SC0=1，CS1=1
SST89E58	F51MBLL5.BIN	0xE000	SC0=1，CS1=1
SST89E516/564	F51MBLL5.BIN	0x10000	SC0=1

说明:

- 在使用其他编程器加载 FM51LL11.BIN 文件时，需要设置缓存区开始地址，即上表中的 BLOCK1 映射地址。
- 配置位未编程时(即擦除后)等于 1，对该配置位执行编程操作后等于 0。
- SST 系列芯片在出厂时已烧录好 BSL 和相应的配置位，可以直接在开发板(实验仪)上使用。
- ME830 实验仪只能实现对 BLOCK0 区的烧写，不可以烧写 BLOCK1。

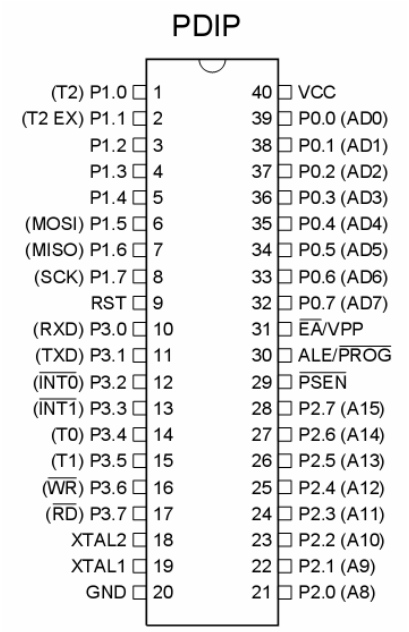
5.4.3 WINBOND 单片机

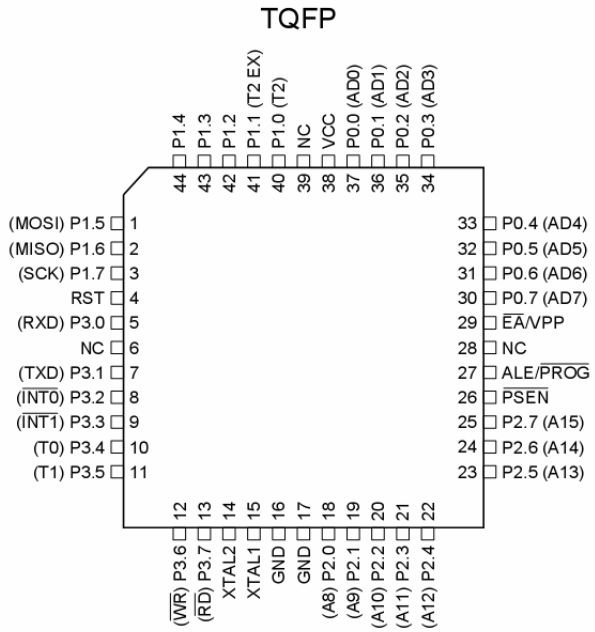
此类芯片与 SST 的单片机类似，也包含两个存储区。一个为 APROM，一个为 LDROM。APROM 为用户代码存储区，LDROM 是启动代码存储区。LDROM 必须预先烧录相应的启动下载代码，才可以在 ME830 实验仪上使用。

WINBOND 单片机的启动代码位于产品光盘的 BOOTLOADER\目录下，文件名为 WB_ISP.HEX。烧录该文件必须使用其他的商业并行编程器。在加载该代码时，请注意缓冲区的偏移量设置，必须保证 WB_ISP.hex 正确的调入到 LDROM 开启区。

5.5 AT89S51/52 管脚定义图

下图分别是 AT89S51/52 的管脚定义示意图，来源于厂家的 Datasheet，其他类型单片机请参考相应的 Datasheet





从上图可知，AT89S51/52 ISP 相关管脚定义如下表：

ISP 接口名称	直插（DIP）	贴片（TQFP）
MOSI	P1.5(6 脚)	P1.5(1 脚)
SCK	P1.7(8 脚)	P1.7(3 脚)
RESET	(9 脚)	(4 脚)
MISO	P1.6(7 脚)	P1.6(2 脚)
VCC	(40 脚)	(38 脚)
GND	(20 脚)	(16，17 脚)

第六章 实验指导

6.1 基础实验

实验一 LED 闪烁

发光二极管是半导体二极管的一种，可以把电能转化成光能。常简写为 LED（light emitting diode）。

发光二极管与普通二极管一样也具有单向导电性。当给发光二极管加上正向电压（大于 LED 的正向压降）就会发光，当给发光二极管加上负向电压就不会发光。

发光二极管的发光亮度与通过的工作电流成正比，一般情况下，LED 的正向工作电流在 10mA 左右，若电流过大时会损坏 LED，因此使用时必须串联限流电阻以控制通过管子的电流。限流电阻 R 可用下式计算：

R＝（E－UF）／IF

式中 E 为电源电压，UF 为 LED 的正向压降，IF 为 LED 的一般工作电流。

普通发光二极管的正向饱和压降为 1.4～2.1V，正向工作电流为 5～20mA.

LED 广泛应用于各种电子电路、家电、仪表等设备中、做电源或电平指示。

1. 实验任务

P0、P2 端口的 LED 亮 300ms，灭 300ms，如此循环。

发光二极管在不停地一亮一灭，间隔时间为 300ms，形成闪烁效果。

2. 实验电路

实验 CPU 电路

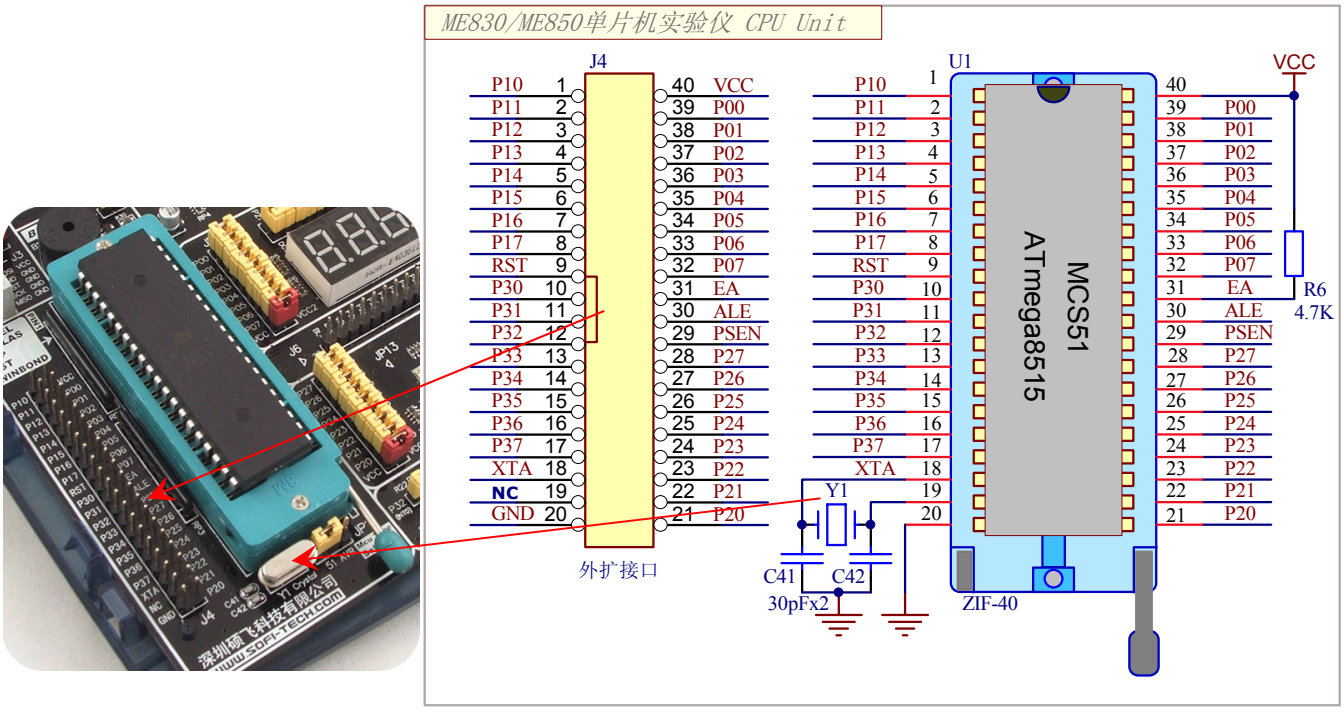


图 6.1 实验 CPU 电路

实验 CPU 插在 40Pin 活动锁紧座 U1 上（图 6.1），可方便更换不同型号的 51 或 AVR 单片机进行烧写或实验，芯片在锁紧座上的方向是芯片缺口背向手柄。注意：实验插座上只能放置和 89C51 管脚兼容的 51 系列单片机（如 AT89S521/52），同样只能放置和 8515 管脚兼容的 AVR 系列单片机（如 ATmega8515L）。

Y1 是可以更换的晶振，J4 是外扩接口，引出 CPU 信号扩展各自模块，比如选配的 TFT 彩屏/SD 卡模块。

实验 CPU 的复位电平“MCU-RST”来自系统控制芯片，兼容 51、AVR 单片机的正常复位，按下实验仪上的复

位键 K9，实验 CPU 即得到复位信号。

注意：本指导教程原理图和 ME830 电路板中 CPU 的 I/O 口标识省掉了小数点，如 P10 表示实际端口 P1.0，P20 表示实际端口 P2.0。

提示：本指导教程中原理图按单元绘制，各硬件单元均通过短路帽和实验 CPU 连接，图中网络标号名称相同的表示电气关系是连接在一起的。如图 6.2 的 LED 显示电路连接到 CPU 的 P0 和 P2 口。整机原理图见光盘。

LED 显示电路

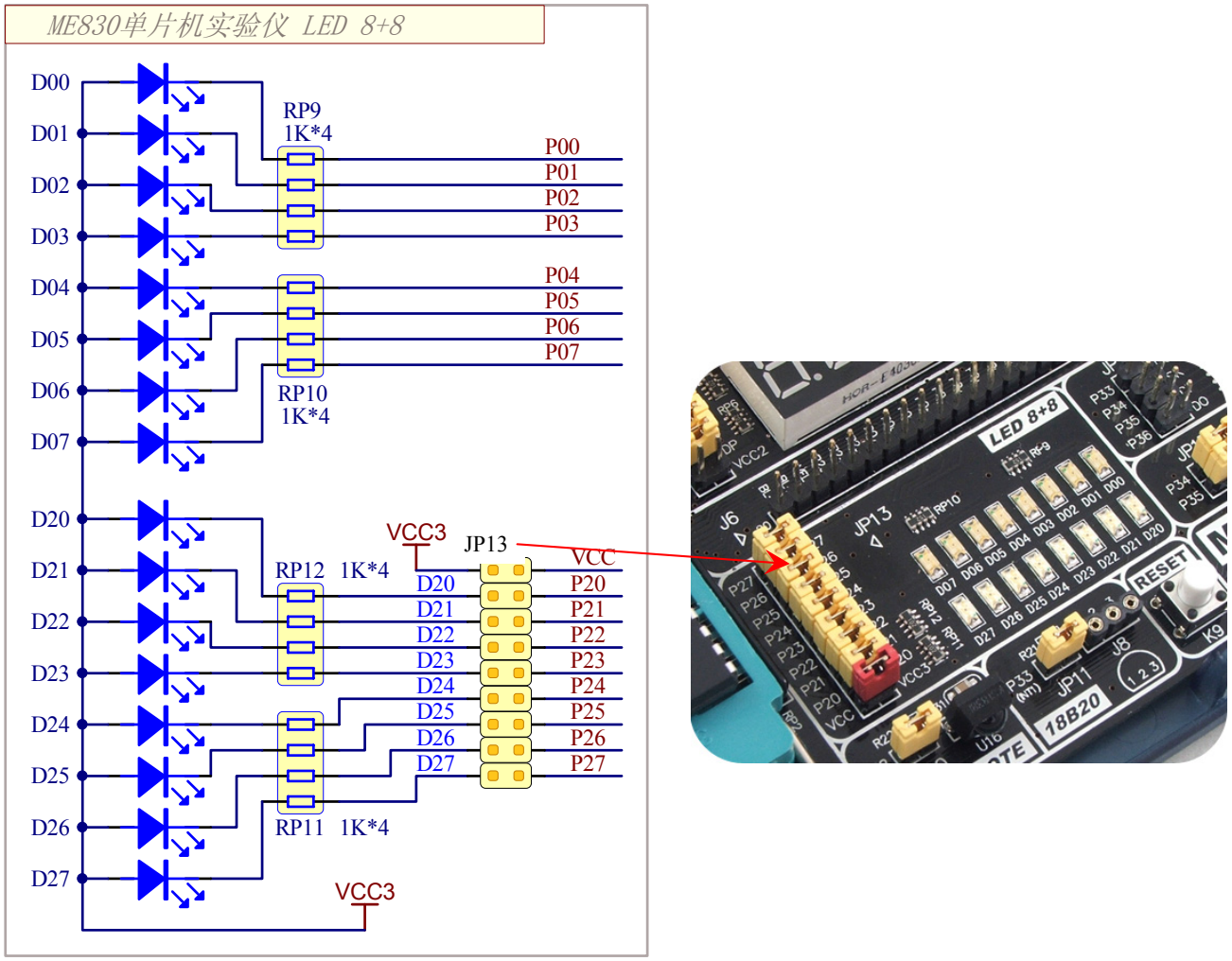


图 6.2 LED 显示电路

RP9、RP10、RP11、RP12 为发光二极管的限流电阻。

用 P0 和 P2 端口作输出口，每个端口接 8 位用作逻辑电平显示的发光二极管。

D00～D07 为发红色光的发光二极管，D20～D27 为发绿色光的发光二极管。共有 16 个发光二极管，ME830 上使用的是贴片封装的。

当 P0 和 P2 端口输出低电平时，D00～D07 和 D20～D27 正向导通发光。

当 P0 和 P2 端口输出高电平时，D00～D07 和 D20～D27 截止不发光。

3. 实验步骤

将 JP13 的 9 个短接子全部用短接帽短接，使 D20～D27 与 P2 端口接通，VCC 向发光二极管模块供电；

4. 程序流程图

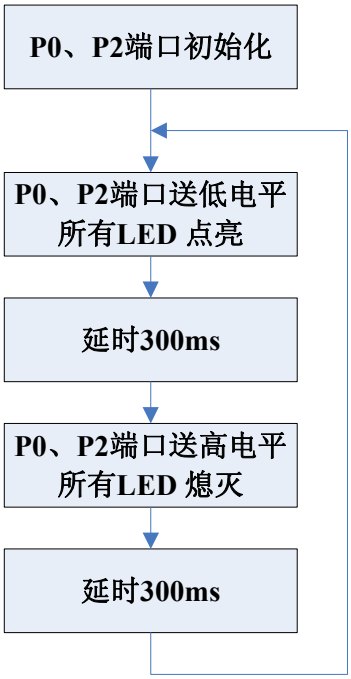


图 6.3　EX1_LED 流程图

5. 汇编源程序

（光盘：Example_A51\EX1_LED）

```
;*********************************************************************
;*                                                                   *
;* ME830 单片机开发实验仪演示程序 - LED 点亮与熄灭                    *
;*                                                                   *
;* P0、P2 端口 16 位 LED 显示                                        *
;*                                                                   *
;* 版本:  V1.0 (2008/06/11)                                          *
;* 作者:  gguoqing (Email: gguoqing@willar.com)                      *
;* 网站:  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界)*
;* 邮箱:  willar@tom.com                                             *
;*                                                                   *
;*【版权】Copyright(C) 深圳硕飞科技有限公司  All Rights Reserved    *
;*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息!         *
;*                                                                   *
;*********************************************************************

        ORG   0000H
        AJMP  MAIN
        ORG   0050H

MAIN:
        MOV   P0,#0FFH          ;端口初始化
        MOV   P2,#0FFH
LOOP:
        MOV   P0,#00H           ;LED 显示
        MOV   P2,#00H
        ACALL  DELAY            ;延时 300ms

        MOV   P0,#0FFH          ;关闭 LED 显示
        MOV   P2,#0FFH
        ACALL  DELAY            ;延时 300ms

        AJMP  LOOP

;--------------------------------------------
; 延时子程序
; 延时 300ms (11.0592MHz)
;--------------------------------------------
DELAY:
        MOV   R5,#3
DEL1:
```

```
        MOV   R6,#200
DEL2:
        MOV   R7,#230
DEL3:
        DJNZ  R7,DEL3          ;第一层循环
        DJNZ  R6,DEL2          ;第二层循环
        DJNZ  R5,DEL1          ;第三层循环
        RET

        END                        ;结束
```

6. C 语言源程序

（光盘：Example_C51\EX1_LED）

```
/*********************************************************************
*                                                                    *
* ME830 单片机开发实验仪演示程序 - LED 点亮与熄灭                    *
*                                                                    *
*  P0、P2 端口 8 位 LED 显示                                         *
*                                                                    *
* 版本：  V1.0 (2008/06/11)                                          *
* 作者：  gguoqing (Email: gguoqing@willar.com)                      *
* 网站：  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界)  *
* 邮箱：  willar@tom.com                                             *
*                                                                    *
*【版权】Copyright(C)深圳硕飞科技有限公司  All Rights Reserved      *
*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！        *
*                                                                    *
*********************************************************************/

#include <reg52.h>
char code reserve [3] _at_ 0x3b;  //保留 0x3b 开始的 3 个字节

/*------------------------------------------------------
  延时函数
------------------------------------------------------*/
void delayms(unsigned int ms)
{
  unsigned char k;
  while(ms--)
  {
    for(k = 0; k < 114; k++);
  }
}

/*------------------------------------------------------
  主函数
------------------------------------------------------*/
void main()
{
   P0 = 0xff;                  //初始化端口
   P2 = 0xff;

   while(1)
   {
     P0 = 0x00;                //LED 显示
     P2 = 0x00;
     delayms(300);             //延时 300ms

     P0 = 0xff;                //关闭 LED 显示
     P2 = 0xff;
     delayms(300);             //延时 300ms
   }
}
```

实验二 LED 流水灯

1. 实验任务

P0、P2 端口的 LED 先从从右至左方向依次点亮，再从左至右方向依次点亮，如此循环形成流水灯效果。

2. 实验线路

（见图 6.1）

3. 实验步骤

将 JP13 的 9 个短接子全部用短接帽短接，使 D20～D27 与 P2 端口接通，VCC 向发光二极管模块供电；

4. 程序流程图

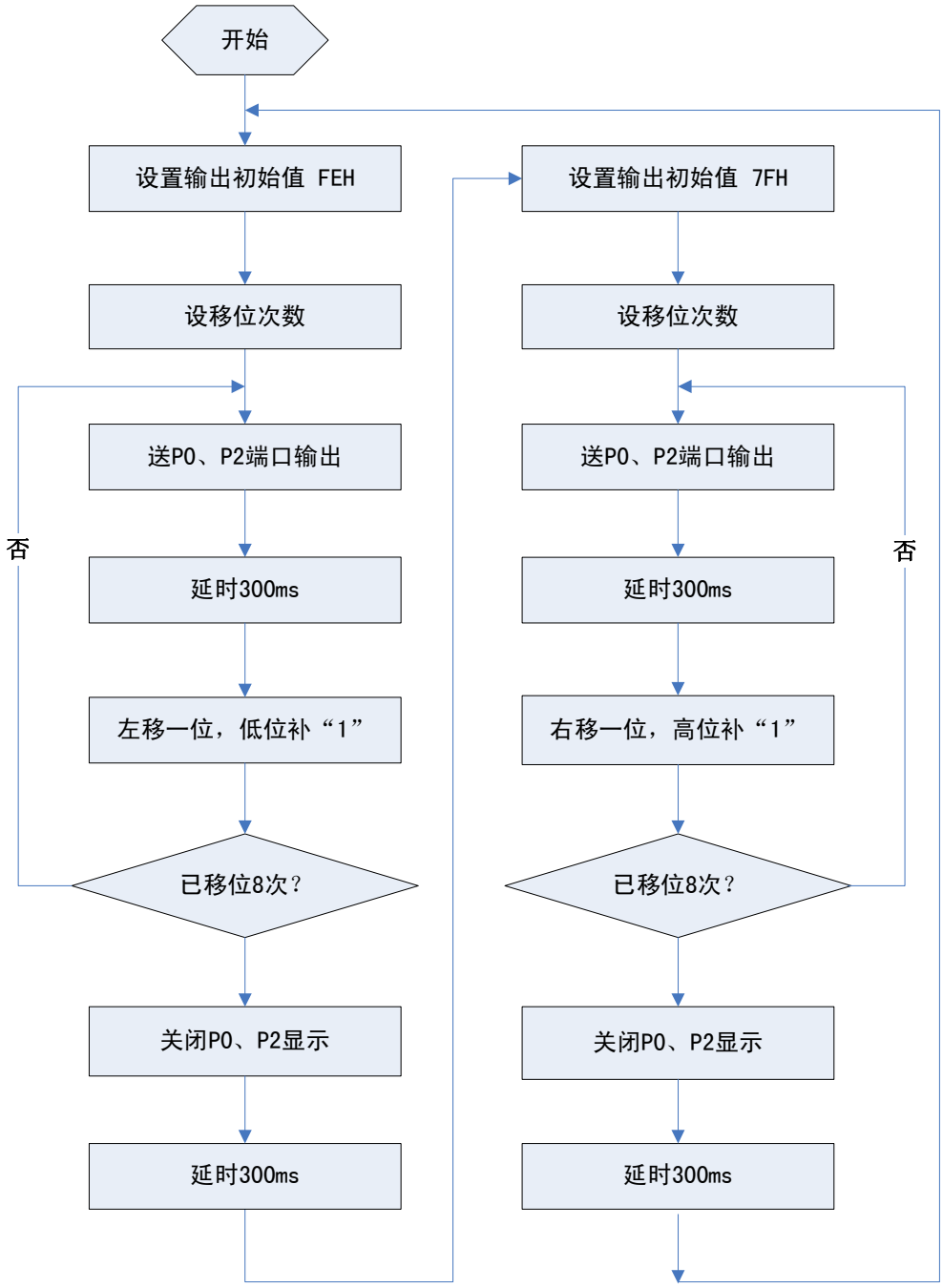


图 6.4 EX2_LEDX8 流程图

5. 汇编源程序

（光盘： Example_A51\EX2_LEDX8）

```
;*********************************************************************
;*                                                                   *
;* ME830 单片机开发系统演示程序 - LED 左右移动流水灯                 *
;*                                                                   *
;* P0、P2 端口 8 位 LED 显示                                         *
;*                                                                   *
;* 版本： V1.0 (2008/06/11)                                          *
;* 作者： gguoqing (Email: gguoqing@willar.com)                      *
;* 网站： www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界)*
;* 邮箱： willar@tom.com                                             *
;*                                                                   *
;*【版权】Copyright(C) 深圳硕飞科技有限公司  All Rights Reserved    *
;*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！       *
;*                                                                   *
;*********************************************************************
        ORG   0000H
        AJMP  MAIN
        ORG   0050H
;-------------------------------------------------
MAIN:
        MOV   P0,#0FFH          ;端口初始化
        MOV   P2,#0FFH
LOOP:
        MOV   A,#0FEH           ;赋初始值
        MOV   R0,#08H           ;移动次数
LOOPL:                          ;左移显示
        MOV   P0,A              ;送数显示
        MOV   P2,A
        RL    A                 ;左移一位
        ACALL  DELAY            ;延时 300ms
        DJNZ  R0,LOOPL          ;是否左移 8 次?

        MOV   P0,#0FFH          ;关闭显示
        MOV   P2,#0FFH
        ACALL  DELAY            ;延时 300ms

        MOV   A,#7FH            ;赋初始值
        MOV   R0,#08H           ;移动次数
LOOPR:                          ;右移显示
        MOV   P0,A              ;送数显示
        MOV   P2,A
        RR    A                 ;右移一位
        ACALL  DELAY            ;延时 300ms
        DJNZ  R0,LOOPR          ;是否右移 8 次?

        MOV   P0,#0FFH          ;关闭显示
        MOV   P2,#0FFH
        ACALL  DELAY            ;延时 300ms

        AJMP  LOOP

;-------------------------------------------------
; 延时子程序
; 延时 300ms (11.0592MHz)
;-------------------------------------------------
DELAY:
        MOV   R5,#3
DEL1:
        MOV   R6,#200
DEL2:
        MOV   R7,#230
DEL3:
        DJNZ  R7,DEL3           ;第一层循环
        DJNZ  R6,DEL2           ;第二层循环
        DJNZ  R5,DEL1           ;第三层循环
        RET
;-------------------------------------------------
        END             ;结束
;-------------------------------------------------
```

6. C 语言源程序

（光盘： Example_C51\EX2_LEDX8）

```
/*********************************************************************
*                                                                    *
* ME830 单片机开发实验仪演示程序 - LED 左右移动流水灯                *
*                                                                    *
*  P0、P2 端口 8 位 LED 显示                                         *
*                                                                    *
* 版本： V1.0 (2008/06/11)                                           *
* 作者： gguoqing (Email: gguoqing@willar.com)                       *
* 网站： www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界)   *
* 邮箱： willar@tom.com                                              *
*【版权】Copyright(C)伟纳电子 www.willar.com  All Rights Reserved   *
*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！        *
*********************************************************************/
#include <reg52.h>
#include <intrins.h>
unsigned char  scan_num;
char code reserve [3] _at_ 0x3b;  //保留 0x3b 开始的 3 个字节

/*-------------------------------------------------------------
  延时函数
-------------------------------------------------------------*/
void delayms(unsigned int ms)
{
  unsigned char k;
  while(ms--)
  {
    for(k = 0; k < 114; k++);
  }
}

/*-------------------------------------------------------------
  主函数
-------------------------------------------------------------*/
void main(void)
{
   unsigned char i;
   P0 = 0xff;                 //初始化端口
   P2 = 0xff;

   while(1)
   {
     scan_num = 0xfe;         //扫描初始值

     for(i = 0;i < 8;i++)     //左移显示
      {
       P0 = scan_num;         //送显示
       P2 = scan_num;
       scan_num<<=1;          //左移一位
       scan_num|=0x01;        //最低位补"1"
       delayms(300);          //延时 300ms
     }
     P0 = 0xff;               //关闭 LED 显示
     P2 = 0xff;
     delayms(300);            //延时 300ms

     scan_num = 0x7f;         //扫描初始值

     for(i = 0;i < 8;i++)     //右移显示
     {
        P0 = scan_num;         //送显示
       P2 = scan_num;
       scan_num >>=1;         //右移一位
       scan_num|=0x80;        //最高位补"1"
       delayms(300);          //延时 300ms
     }
     P0 = 0xff;               //关闭 LED 显示
     P2 = 0xff;
     delayms(300);            //延时 300ms
   }
}
```

实验三　继电器控制

继电器是一种电子控制器件，它具有控制系统（输入回路）和被控制系统（输出回路），通常应用于自动控制电路中，它实际上是用较小的电流去控制较大电流的一种“自动开关”。

1. 实验任务

用按键控制继电器的工作状态：

K1- 吸合键，K2- 释放键

按 K1 键，继电器吸合，DL11 灯亮。

按 K2 键，继电器释放，DL11 灯灭。

2. 实验线路

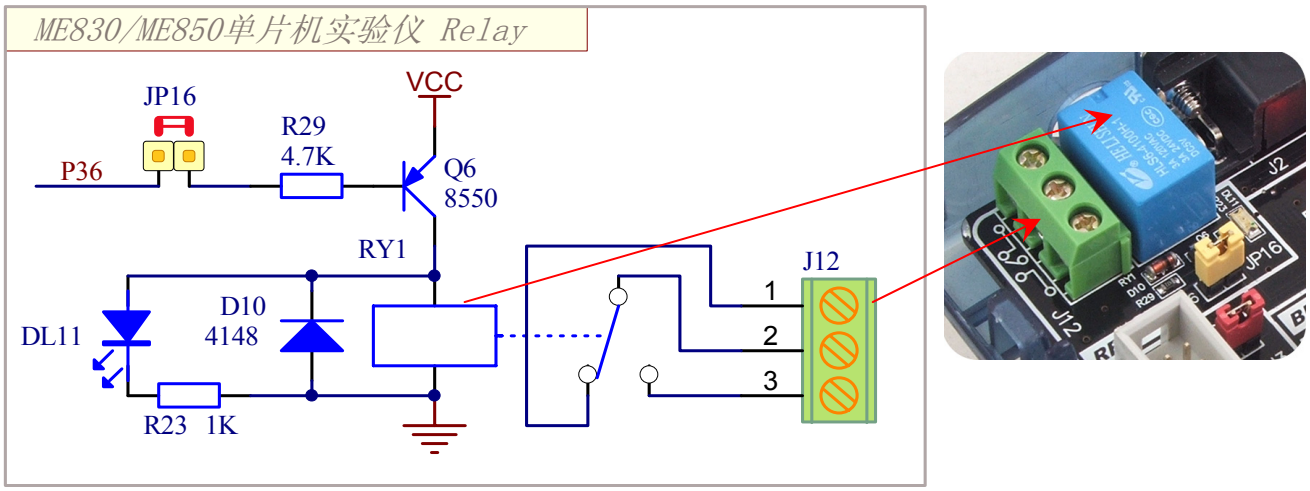


图 6.5　继电器驱动电路

因为继电器稳定吸合时，线圈的工作电流已超出 C51 芯片 I/O 端口的驱动能力，所以考虑使用三极管来驱动继电器。因继电器线圈是电感性负载，在驱动电路中还要加相应的保护措施才行。

实际的继电器控制电路如图 6.5 所示，当 P36 引脚输出“0”时，三极管 Q6 导通，继电器 RY1 吸合，发光二极管 DL11 点亮；当 P36 引脚输出“1”时，三极管 Q6 截止，继电器 RY1 释放，发光二极管 DL11 熄灭。

3. 实验步骤

短接 JP16 短接子，使继电器接口电路使能；

将 JP8 的 8 个短接子全部用短接帽短接，使独立按键与相应的端口接通。

4. 程序流程图

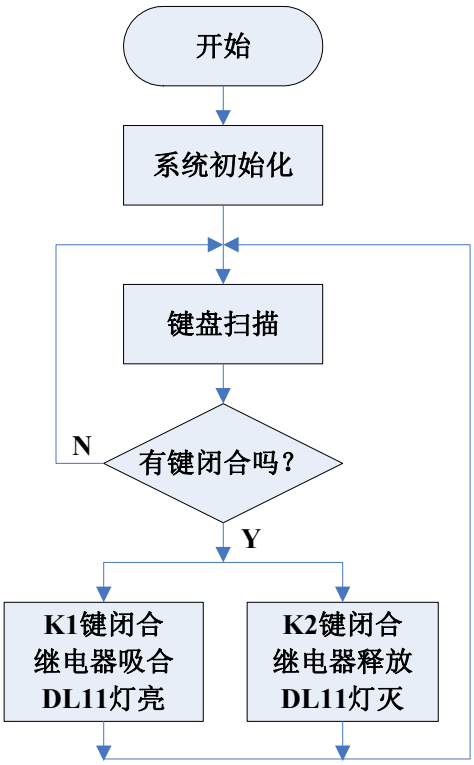


图 6.6　EX3_RELAY 流程图

5. 汇编源程序

（光盘：Example_A51\EX3_RELAY）

```
;*********************************************************************
;*                                                                   *
;* ME830 单片机开发系统演示程序 -  键控制继电器                      *
;*                                                                   *
;* K1- 吸合键，K2- 释放键                                            *
;*                                                                   *
;* 版本：  V1.0 (2008/08/16)                                         *
;* 作者：  gguoqing (Email: gguoqing@willar.com)                     *
;* 网站：  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界) *
;* 邮箱：  willar@tom.com                                            *
;*                                                                   *
;*【版权】Copyright(C) 深圳硕飞科技有限公司  All Rights Reserved    *
;*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息!         *
;*                                                                   *
;*********************************************************************

        KEY_NEW  EQU  40H
        KEY_OLD  EQU  41H

        K1      BIT  P1.4      ;
        K2      BIT  P1.5      ;
        RELAY   BIT  P3^6          ;继电器控制线

;---------------------------------------------------------
        ORG  0000H
        AJMP  MAIN
        ORG  0050H
;---------------------------------------------------------

; 主程序

;---------------------------------------------------------
MAIN:
         MOV  SP,#60H          ;设置栈指针
         MOV  P0,#0FFH             ;
         MOV  P2,#0FFH             ;
         MOV  KEY_OLD,#03H         ;初始键比较值
```

```
KEY_CHK:                            ;循环检测按键是否按下
        ACALL SCAN_KEY              ;输入按键状态
        XRL   A,KEY_OLD             ;查按键值是否改变
        JZ   KEY_CHK                ;若无键被按,则跳回 KEY_CHK

        ACALL   DELAY               ;延时去抖
        ACALL   SCAN_KEY            ;再次检查按键值
        XRL   A,KEY_OLD
        JZ   KEY_CHK

        MOV     KEY_OLD,KEY_NEW     ;保存按键状态
        ACALL PROC_KEY
        AJMP   KEY_CHK

;-------------------------------------------------------

; 扫描按键子程序
; 返回值:    A --- 按键状态

;-------------------------------------------------------
SCAN_KEY:
        CLR   A
        MOV   C,K1
        MOV   ACC.0,C
        MOV   C,K2
        MOV   ACC.1,C
        MOV   KEY_NEW,A             ;无键按下 key_new=03H
        RET
;-------------------------------------------------------

; 按键处理子程序

;-------------------------------------------------------
PROC_KEY:
        MOV   A,KEY_NEW
        JNB   ACC.0,PROC_K1          ;K1 键按下
        JNB   ACC.1,PROC_K2          ;K2 键按下
         RET
PROC_K1:                             ;按键 K1 处理程序
        CLR   RELAY                 ;继电器吸合
        RET

PROC_K2:                            ;按键 K2 处理程序
        SETB   RELAY                    ;继电器释放
        RET

;-------------------------------------------------------

; 延时子程序(10MS)

;-------------------------------------------------------
DELAY:
        MOV   R6,#10
DEL1:
        MOV     R7,#185
DEL2:
        NOP
        NOP
        NOP
        DJNZ   R7,DEL2
        DJNZ   R6,DEL1
        RET
;-------------------------------------------------------
        END      ;结束
;-------------------------------------------------------
```

6. C 语言源程序

（光盘：Example_C51\EX3_RELAY）

```
/*********************************************************************
 *                                                                   *
 * ME830 单片机开发实验仪演示程序 -  键控制继电器                    *
 *                                                                   *
 * K1- 吸合键，K2- 释放键                                            *
 *                                                                   *
 * 版本：  V1.0 (2008/08/16)                                         *
 * 作者：  gguoqing (Email: gguoqing@willar.com)                     *
 * 网站：  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界) *
 * 邮箱：  willar@tom.com                                            *
 *                                                                   *
 *【版权】Copyright(C)伟纳电子 www.willar.com  All Rights Reserved  *
 *【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！       *
 *                                                                   *
 *********************************************************************/
#include <reg52.h>
char code reserve[3]_at_ 0x3b; //保留 0x3b 开始的 3 个字节
sbit K1 = P1 ^ 4;
sbit K2 = P1 ^ 5;
sbit relay = P3 ^ 6;
unsigned char key_new, key_old;
/*-------------------------------------------------------
延时函数
-------------------------------------------------------*/
void delayms(unsigned int ms)
{
  unsigned char k;
  while (ms--)
  {
    for (k = 0; k < 114; k++);
  }
}
/*-------------------------------------------------------
扫描健盘函数
-------------------------------------------------------*/
unsigned char scan_key()
{
  key_new = 0x00;
  key_new |= K2;
  key_new <<= 1;                    //左移 1 位
  key_new |= K1;
  return key_new;                   //无键按下 key_new=0x03。
}
/*-------------------------------------------------------
主函数
-------------------------------------------------------*/
void main(void)
{
  P0 = 0xff;                        //初始化端口
  P2 = 0xff;
  P1 = 0xf0;                        //置 P1 高四位为输入
  key_old = 0x03;                   //初始键比较值
  relay = 1;                        //继电器释放

  while (1)
  {
    scan_key();
    if (key_new != key_old)
    {
      delayms(10);                  //延时去抖动
      scan_key();                   //再次判健是否按下
      if (key_new != key_old)
      {
        key_old = key_new;           //保存按键状态
        if ((key_new &0x01) == 0)     //K1 键按下
          relay = 0;                 //继电器吸合

        if ((key_new &0x02) == 0)    //K2 键按下
          relay = 1;                  //继电器释放
      }
    }
  }
}
```

实验四　蜂鸣器控制

蜂鸣器是一种一体化结构的电子讯响器，广泛应用于电子产品中作发声报警。

蜂鸣器有两类：一类是压电式，一类是电磁式。

因为 ME830 上使用是电磁式蜂鸣器，所以只对电磁式蜂鸣器进行相应的介绍。

电磁式蜂鸣器有两种类型：有源蜂鸣器和无源蜂鸣器。有源蜂鸣器内部带振荡源，无源蜂鸣器内部不带振荡源。**这里所说的“源”不是指“电源”，而是指“振荡源”。**

有源蜂鸣器和无源蜂鸣器的主要差别是对输入信号的要求不一样，有源蜂鸣器工作的理想信号是直流电，无源蜂鸣器工作的理想信号是方波。无源蜂鸣器接直流电是不会工作的。

1. 实验任务

蜂鸣器响 300ms，P0 端口的 D00 和 D07 点亮。

蜂鸣器停 300ms，P0 端口的 D00 和 D07 熄灭。

蜂鸣器在不停地一响一停，发光二极管也在不停地一亮一灭，间隔时间为 300ms，形成声光报警效果。

2. 实验线路

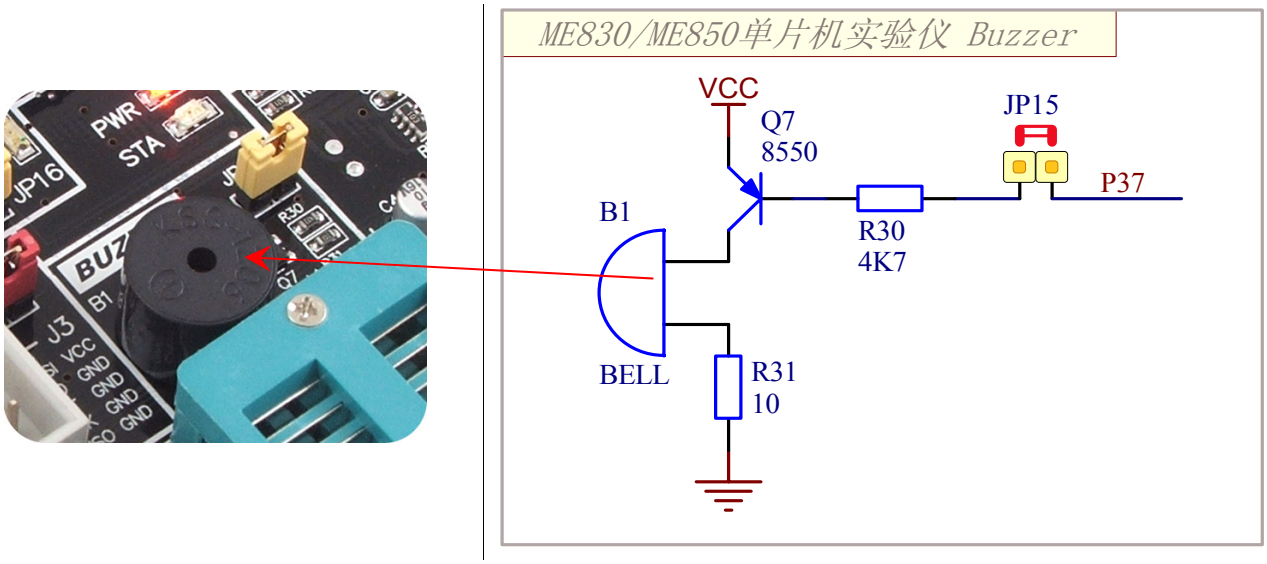


图 6.7　蜂鸣器实验电路

由于蜂鸣器工作时电流比较大，单片机的 I/O 端口输出电流比较小，无法直接进行驱动，需要加一个 PNP 型三极管 8550 进行隔离放大。

单片机的 P3.7 引脚通过限流电阻 R30 与三极管 Q7 的基极相接，蜂鸣器 B1 和电阻 R31 串接在三极管 Q7 的集电极回路中。

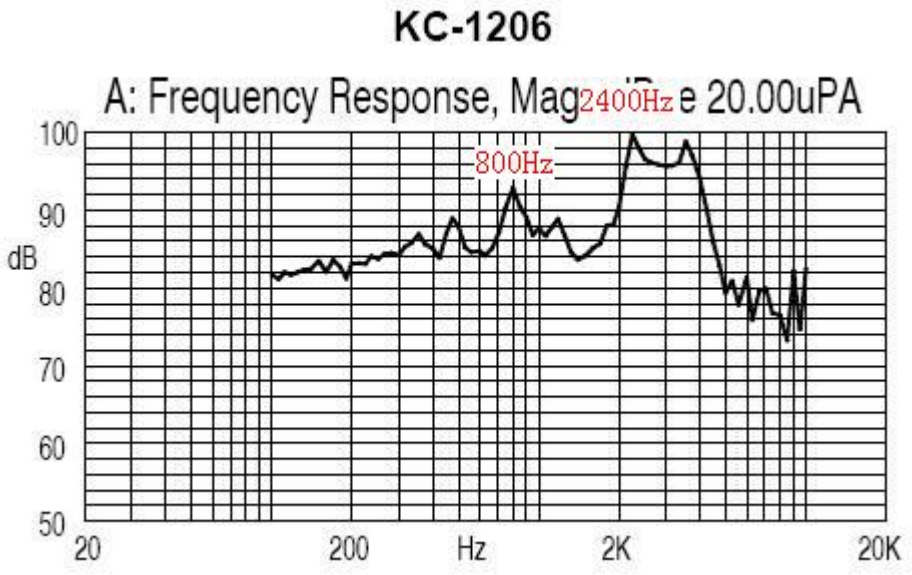
三极管 Q7 的基极经限流电阻 R30 后由单片机的 P3.7 引脚控制，当 P3.7 输出高电平时，三极管 Q7 截止，没有电流流过蜂鸣器内部线圈，蜂鸣器不发声；当 P3.7 输出低电平时，三极管 Q7 导通，有电流流过蜂鸣器内部线圈，蜂鸣器发出声。所以让 P3.7 引脚不断地输出方波，三极管 Q7 就会不断地导通和截止，使无源蜂鸣器发出声音。因此，可以通过程序控制 P3.7 引脚电平来使蜂鸣器发音与关闭。

程序中改变单片机 P3.7 引脚的输出频率，就可以调整蜂鸣器的音调，可产生各种不同音调的声音。改变 P3.7 引脚的输出电平的占空比，则可以控制蜂鸣器的声音大小。

ME830 使用是无源蜂鸣器，型号：KC-1206，参数如下：

Model Nunber：	KC-1206
Rated Voltage 电压(V)	5V
Operating Voltage 工作电压 (V)	4-8V

Max.Rated Current 最大电流 (mA)	40mA
Coil Resistance (DC)直流电阻 (Ω)	42±4Ω
Coil impedance(AC) 交流阻抗 (Ω)	80Ω
Min. Sound Pressure Level 最小声压(dB/cm)	85db/10cm
Resonant Frequency 谐振频率 (Hz)	2400
Perating Temperature 工作温度 (℃)	-20 - +60
Weight 重量 (g)	2



频率响应图

从频率响应图可以看到 KC-1206 蜂鸣器在 2400 Hz 处有一个最高峰，这个峰就是 KC-1206 蜂鸣器额定频率（2400 Hz）。当外加的方波信号与其额定频率相同时，产生共振，输出声音最大。另外在频率 800 Hz 处也有一个高峰，当外加的方波信号与其峰频率相同时，输出声音也将比较大。因这两处峰的频率值不同，所以蜂鸣器发出声音的音调也不相同。

3. 实验步骤

短接 JP15 短接子，使蜂鸣器接口电路使能。

将 JP13 的 VCC－VCC3 短接子用短接帽短接，VCC 向发光二极管模块供电。

4. 程序流程图

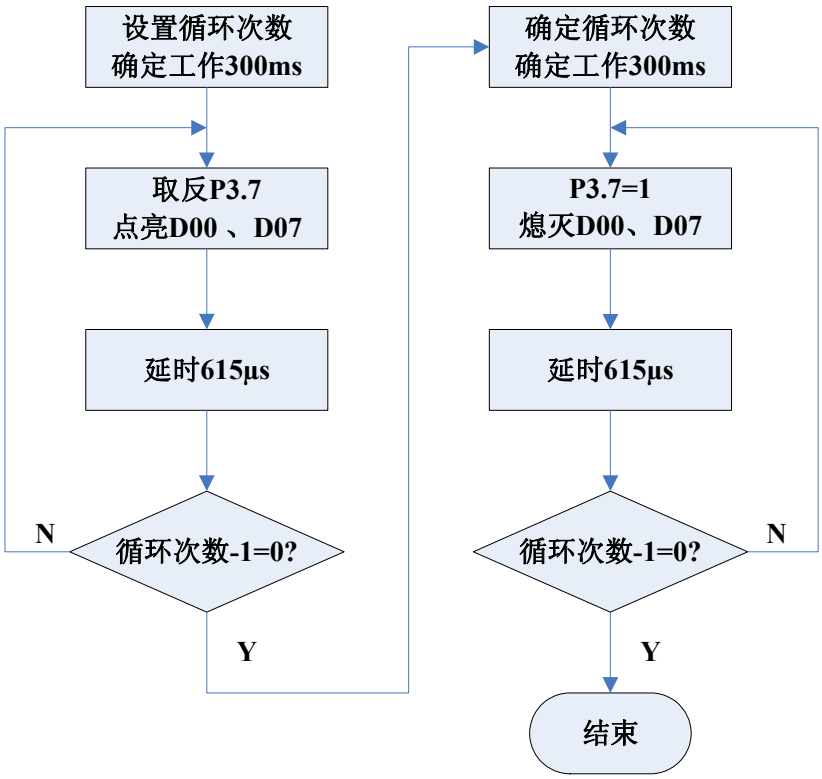


图 6.8　蜂鸣器实验流程图

5. 汇编源程序

（Example_A51\EX4_BELL）

```
;*********************************************************************
;*                                                                   *
;* ME830 单片机开发系统演示程序 -  滴滴倒车声                        *
;*                                                                   *
;* 蜂鸣器输出                                                        *
;*                                                                   *
;* 版本:  V1.1 (2008/07/04)                                          *
;* 作者:  gguoqing (Email: gguoqing@willar.com)                      *
;* 网站:  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界)*
;* 邮箱:  willar@tom.com                                             *
;*                                                                   *
;*【版权】Copyright(C) 深圳硕飞科技有限公司  All Rights Reserved    *
;*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息!         *
;*                                                                   *
;*********************************************************************
        BEEP  BIT  P3.7

        ORG 0000H
        AJMP  MAIN
        ORG  0050H

MAIN:
        MOV  SP,#60H          ;初始化
        MOV  P0,#0FFH
        MOV  P2,#0FFH
LOOP1:
        MOV  R2,#10           ;响 300ms
LOOP2:
        MOV  R3,#49
LOOP3:
        CPL  BEEP             ;输出频率 800Hz
        MOV  P0,#7EH          ;点亮 D00 和 D07
        ACALL  DELAY          ;延时 615us
        DJNZ R3,LOOP3
        DJNZ R2,LOOP2

        MOV  R2,#10           ;关 300ms
LOOP4:
```

```
        MOV  R3,#49
LOOP5:
        SETB  BEEP              ;关闭蜂鸣器
        MOV  P0,#0FFH           ;关闭显示
        ACALL DELAY             ;延时 615us
        DJNZ R3,LOOP5
        DJNZ R2,LOOP4

        SJMP LOOP1

;-------------------------------------------------------
; 延时 615us
;-------------------------------------------------------
DELAY:
        MOV R7,#189
DEL:
        NOP
        DJNZ R7,DEL
        RET

        END                 ;结束

;-------------------------------------------------------
;信号产生的方法
;800Hz 信号周期为 1230us
;615-49-10=301350us=301ms
;-------------------------------------------------------
```

6. C 语言源程序

（光盘：Example_C51\EX4_BELL）

```
/*********************************************************************
 *                                                                   *
 * ME830 单片机开发实验仪演示程序 -  滴滴倒车声                      *
 *                                                                   *
 * 蜂鸣器输出                                                        *
 *                                                                   *
 * 版本：  V1.1 (2008/07/04)                                         *
 * 作者：  gguoqing (Email: gguoqing@willar.com)                     *
 * 网站：  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界) *
 * 邮箱：  willar@tom.com                                            *
 *                                                                   *
 *【版权】Copyright(C)伟纳电子 www.willar.com  All Rights Reserved  *
 *【声明】此程序仅用于学习与参考，引用请注明版权和作者信息!         *
 *                                                                   *
 *********************************************************************/

#include <reg52.h>
#include <intrins.h>

sbit BEEP = P3 ^ 7;
char code reserve[3]_at_ 0x3b; //保留 0x3b 开始的 3 个字节

/-------------------------------------------------------
310us 延时函数
晶振：11.0592MHz
 -------------------------------------------------------/
void delay(void)
{
  unsigned char i;
  for (i = 143; i > 0; i--)     ;
}

/-------------------------------------------------------
主函数
 -------------------------------------------------------/
void main(void)
{
  unsigned int j;
  P0 = 0xff; //端口初始化
  P1 = 0xff;
  P2 = 0xff;
```

```
    while (1)
    {
      for (j = 490; j > 0; j--)
      //响 300ms
      {
        BEEP = ~BEEP; //输出频率 800Hz
        P0 = 0x7E; //点亮 D00 和 D07
        delay(); //延时 310us
        delay(); //延时 310us
      }

      for (j = 490; j > 0; j--)
      //关 300ms
      {
        BEEP = 1; //关闭蜂鸣器
        P0 = 0xff; //关闭显示
        delay(); //延时 310us
        delay(); //延时 310us
      }
    }
  }
```

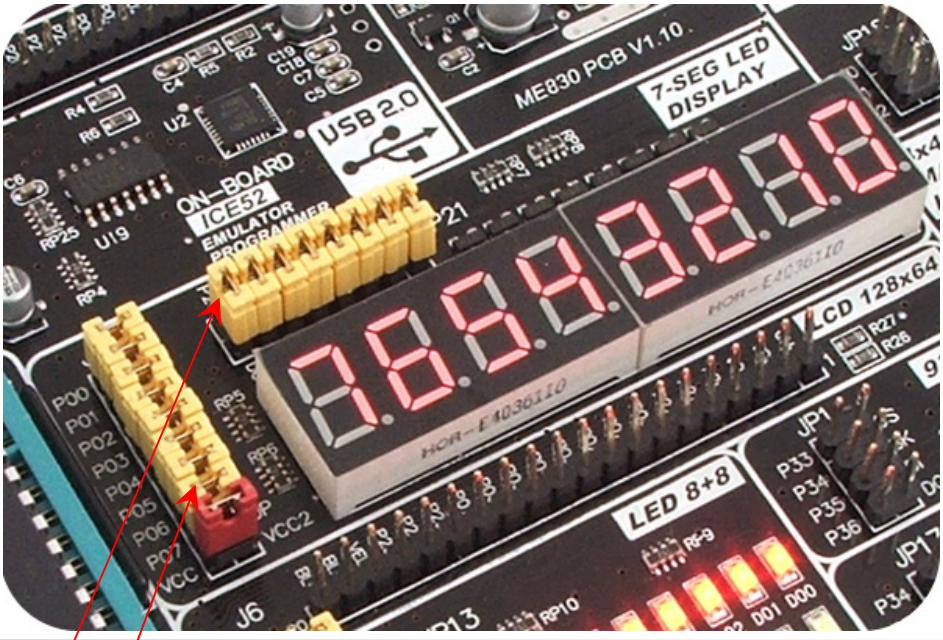
实验五　数码管显示 0-7

数码管显示输出是单片机系统中最常用的一种显示输出，主要用于单片机控制中的数据输出和状态信息显示。

1. 实验任务

先将“0-7”数码管的段码值写入显示存储器中，使 8 位数码管从右至左显示 0－7 。

2. 实验线路



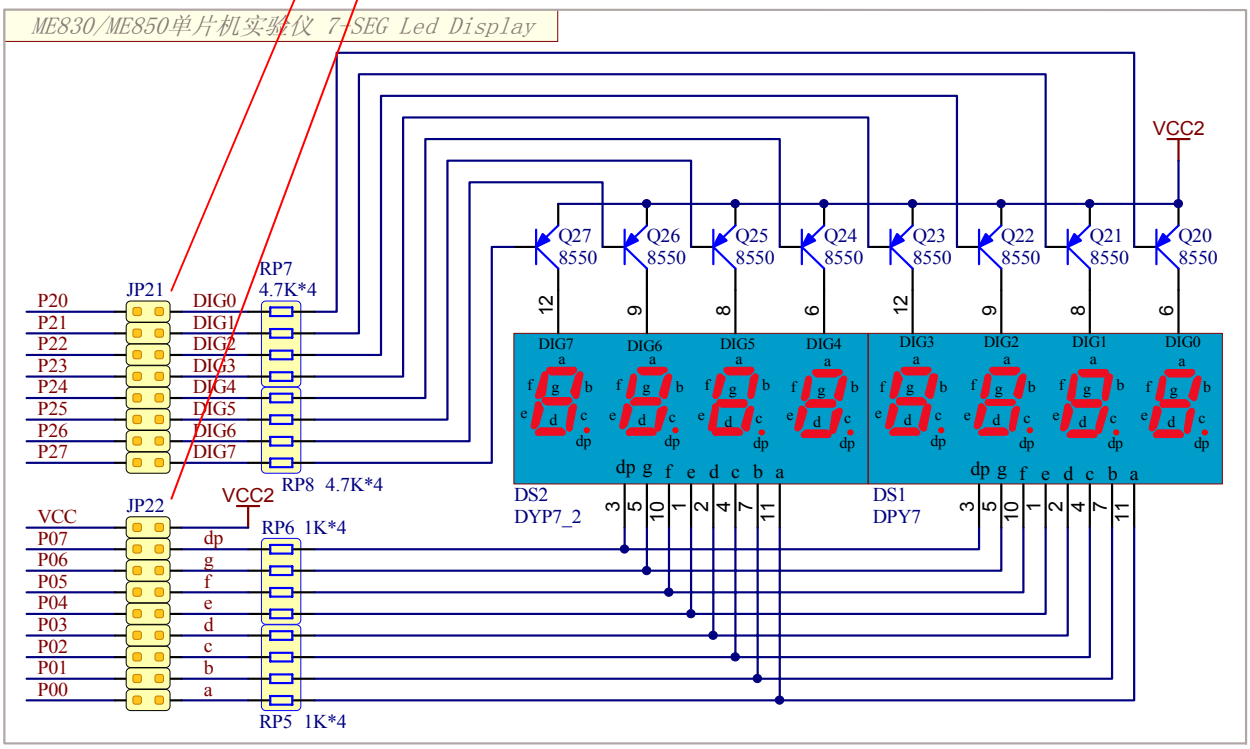


图 6.9　数码管显示电路

ME830 采用 8 位数码管动态扫描显示，可以简化硬件电路、方便软件编程和减少电源的功耗。具体的电路原理图如图 6.9 所示。

8 位数码管动态扫描显示,就是将 8 个数码管的 8 个相同的段线并接在一起，通过限流电阻 RP5、RP6 接到 AT89S52 的 P0 口，由 P0 口控制字段输出(低电平有效)。而各位共阳极数码管的 COM 由 AT89S52 的 P2 口(低电平有效)通过限流电阻 RP7、RP8 控制 Q20－Q27 来实现 8 位数码管的位输出控制(高电平有效)。

2.1 数码管动态扫描原理简介

从数码管动态扫描显示电路的原理可知，对于 8 位数码管动态扫描显示需要由两组信号来控制：**一组是字段输出口输出的字形代码，用来控制显示的字形，称为段码；另一组是位输出口输出的控制信号，用来选择第几位数码管工作，称为位码。**

由于各位数码管的段线并联，段码的输出对各位数码管来说都是相同的。因此，在同一时刻如果各位数码管的位选线都处于选通状态的话，8 位数码管将显示相同的字符。若要各位数码管能够显示出与本位相应的字符，就必须采用扫描显示方式。即在某一时刻，只让某一位的位选线处于导通状态，而其它各位的位选线处于关闭状态。同时，段线上输出相应位要显示字符的字型码。这样在同一时刻，只有选通的那一位显示出字符，而其它各位则是熄灭的，如此循环下去，就可以使各位数码管显示出将要显示的字符。

虽然这些字符是在不同时刻出现的，而且同一时刻，只有一位显示，其它各位熄灭，但由于数码管具有余辉特性和人眼有视觉暂留现象，只要每位数码管显示间隔足够短，给人眼的视觉印象就会是连续稳定地显示。

数码管不同位显示的时间间隔可以通过调整延时程序的延时长短来完成。数码管显示的时间间隔也能够确定数码管显示时的亮度，若显示的时间间隔长，显示时数码管的亮度将亮些，若显示的时间间隔短，显示时数码管的亮度将暗些。若显示的时间间隔过长的话，数码管显示时将产生闪烁现象。所以，在调整显示的时间间隔时，即要考虑到显示时数码管的亮度，又要数码管显示时不产生闪烁现象。

2.2 数码管组成结构

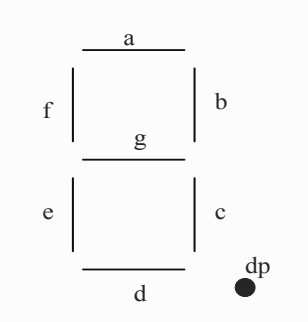


图 6.10　数码管结构图

数码管内部是由 7 个条形的发光二极管和右下方一个圆形的发光二极管组成，这样一共有 8 个段线，恰好适用于 8 位的并行系统。数码管按内部连接方式分为共阴极数码管和共阳极数码管两种。

（1）共阴极数码管

共阴极数码管是将所有发光二极管的阴极接在一起作为公共端 COM，当公共端接低电平时，某一段阳极上的电平为“1”时，该段点亮，电平为“0”时，该段熄灭。

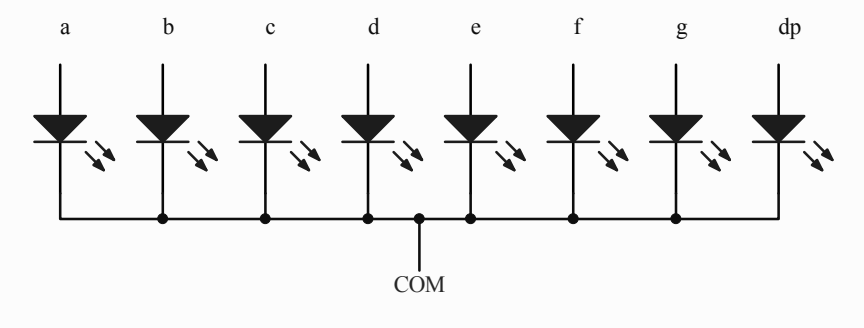


图 6.11 共阴数码管结构图

（2）共阳极数码管

共阳极数码管是将所有发光二极管的阳极接在一起作为公共端 COM，当公共端接高电平时，某一段阴极上的电平为“0”时，该段点亮，电平为“1”时，该段熄灭。

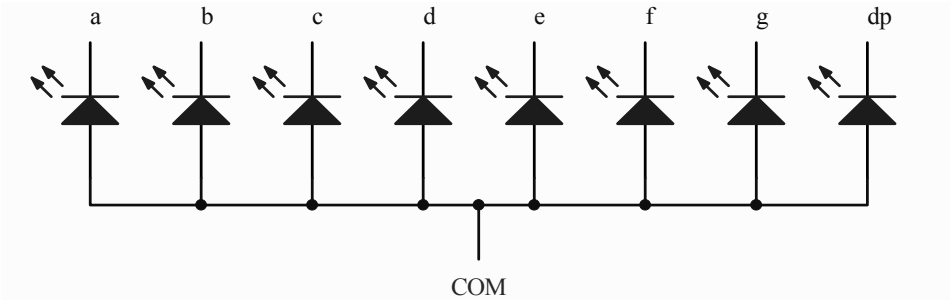


图 6.12 共阳数码管结构图

（3）共阳极数码管的字型代码表

字型	dp	g	f	e	d	c	b	a	段码
0	1	1	0	0	0	0	0	0	0C0H
1	1	1	1	1	1	0	0	1	0F9H
2	1	0	1	0	0	1	0	0	0A4H
3	1	0	1	1	0	0	0	0	0B0H
4	1	0	0	1	1	0	0	1	99H
5	1	0	0	1	0	0	1	0	92H
6	1	0	0	0	0	0	1	0	82H
7	1	1	1	1	1	0	0	0	0F8H
8	1	0	0	0	0	0	0	0	80H
9	1	0	0	1	0	0	0	0	90H
a	1	0	0	0	1	0	0	0	88H
b	1	0	0	0	0	0	1	1	83H
c	1	1	0	0	0	1	1	0	0C6H
d	1	0	1	0	0	0	0	1	0A1H
E	1	0	0	0	0	1	1	0	86H
f	1	0	0	0	1	1	1	0	8EH

举例：

如果你想让图 1 最右边的数码管显示“0”的话，首先将段码“0C0H”送达 P0 口，然后将 P2.0 清为低电平。当 P2.0 为低电平时，三极管 Q20 导通，其该位数码管的公共阳极接至＋5V，于是该位数码管就显示“0”。

```
MOV   P0,#0C0H      ;送段码到 P0 口
MOV   P2,#0FEH      ;清 P2.0 为低电平
```

3. 实验步骤

将 JP21 的 8 个短接子用短接帽短接，使数码管的位控制线与 P2 端口接通。

将 JP22 的 9 个短接子用短接帽短接，使数码管的数据线与 P0 端口接通，并使 VCC 向数码管接口电路供电。

4. 程序设计

数码管显示程序的编程方法

1）先准备好要显示的数据，放入相应的显示存储单元中。

2）根据要使用的数码管的具体位置来确定扫描初值和扫描方向。

3）根据使用数码管的个数来确定扫描的位数。

4）查表将要显示的数据转换为能使数码管正确显示相对应的段码。

5）分时送段码和位码，数码管开始循环显示。

5. 程序流程图

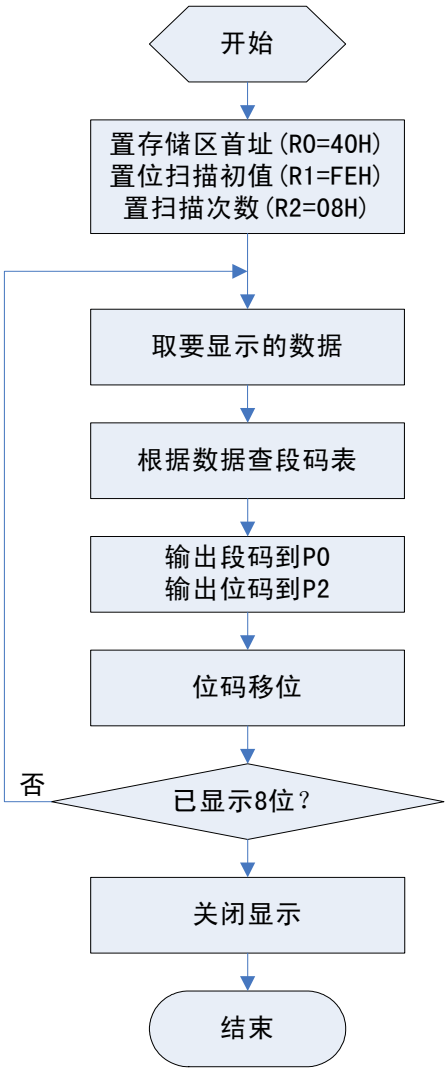


图 6.13　EX5_7SEG 流程图

6. 汇编源程序

（光盘：Example_A51\EX5_7SEG）

```
;*********************************************************************
;* ME830 单片机开发系统演示程序 - 8 位数码管显示                      *
;* 8 位数码管从右至左显示 0-7                                         *
;*                                                                   *
;* 版本:  V1.0 (2008/07/20)                                          *
;* 作者:  gguoqing (Email: gguoqing@willar.com)                      *
;* 网站:  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界)*
;* 邮箱:  willar@tom.com                                             *
;*                                                                   *
;*【版权】Copyright(C) 深圳硕飞科技有限公司  All Rights Reserved    *
;*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息!         *
;*********************************************************************
```

```
        DISSTART  EQU  40H     ;显示单元首地址
        LED_DATA  EQU  P0      ;数码管数据口定义
;-------------------------------------------------------------
        ORG 0000H
        AJMP MAIN
        ORG 0050H
;-------------------------------------------------------------
; 主程序
;-------------------------------------------------------------
MAIN:
        MOV  SP,#60H
        MOV  P0,#0FFH          ;初始化
        MOV  P2,#0FFH
        MOV  R2,#08H           ;8 组数据

        MOV  R0,#DISSTART      ;显示存储单元首地址
        MOV  R1,#00H
MAIN1:
        MOV  A,R1
        MOV  @R0,A             ;将 0-7 分别存入显示存储单元
        INC  R1
        INC  R0
        DJNZ  R2,MAIN1         ;
LOOP:
        ACALL  PLAY            ;循环显示
        SJMP   LOOP

;-------------------------------------------------------------
; 显示子程序
;-------------------------------------------------------------
PLAY:
        MOV R0,#DISSTART       ;获得显示单元首地址
        MOV R1,#0FEH           ;位码初始值
        MOV R2,#08H            ;有 8 位数码管显示
DISP1:
        MOV A,@R0              ;取要显示的数据
        MOV DPTR,#TAB_NU       ;置段码表首址
        MOVC A,@A+DPTR         ;根据数据查段码表
        MOV LED_DATA,A         ;段码输出
        MOV P2,R1              ;位码输出
        MOV A,R1               ;准备下一次显示的位码
        RL A
        MOV R1,A               ;保存位码
        INC R0                 ;取下一个显存单元地址
        ACALL DELAY            ;调用延时
        DJNZ R2,DISP1          ;8 位数码管是否显示完
        MOV  P2,#0FFH          ;关闭显示
        RET                    ;显示完成，返回

;-------------------------------------------------------------
;1MS 延时子程序
;-------------------------------------------------------------
DELAY:
        MOV  R6,#5
DEL1:
        MOV  R7,#93
DEL2:
        DJNZ  R7,DEL2          ;第一层循环
        DJNZ  R6,DEL1          ;第二层循环
        RET

;-------------------------------------------------------------
; 段码表
;-------------------------------------------------------------
TAB_NU:
    DB 0C0H,0F9H,0A4H,0B0H,099H,092H,082H,0F8H,080H
    DB 090H,088H,083H,0C6H,0A1H,086H,08EH,0FFH

;-------------------------------------------------------------
        END                    ;结束
;-------------------------------------------------------------
```

7. C 语言源程序

（光盘：Example_C51\EX5_7SEG）

```
/*********************************************************************
 *                                                                   *
 * ME830 单片机开发实验仪演示程序 - 8 位数码管显示                  *
 *                                                                   *
 * 8 位数码管从右至左显示 0-7                                        *
 *                                                                   *
 * 版本:  V1.0 (2008/07/20)                                          *
 * 作者:  gguoqing (Email: gguoqing@willar.com)                      *
 * 网站:  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界) *
 * 邮箱:  willar@tom.com                                             *
 *                                                                   *
 *【版权】Copyright(C)伟纳电子 www.willar.com  All Rights Reserved  *
 *【声明】此程序仅用于学习与参考，引用请注明版权和作者信息!        *
 *                                                                   *
 *********************************************************************/
#include <reg52.h>
#include <intrins.h>

char code reserve[3]_at_ 0x3b; //保留 0x3b 开始的 3 个字节

unsigned char code display[] =
{
  0xC0, 0xF9, 0xA4, 0xB0, 0x99, 0x92, 0x82, 0xF8, 0x80, 0x90
};

/*-------------------------------------------------------
 延时子程序
 -------------------------------------------------------*/
void delayms( unsigned int ms )
{
  unsigned char k;
  while (ms--)
  {
    for (k = 0; k < 114; k++)
      ;
  }
}

/*-------------------------------------------------------
 显示函数
 -------------------------------------------------------*/
void main(void)
{
  unsigned char k, shift;

  P0 = 0xff;                    //端口初始化
  P2 = 0xff;

  while (1)
  {
    shift = 0xfe;               //位扫描初值
    P2 = 0xff;                  //关闭显示
    for (k = 0; k < 8; k++)
    {
      P0 = display[k];          //送段码
      P2 = shift;               //送位码
      shift = _crol_(shift, 1); //左移一位，修改位码
      delayms(1);               //延时 1ms
    }
  }
}

 /*-------------------------------------------------------*/
```

实验六　独立按键识别

按键（轻触开关）是一种广泛应用于各种电子设备的元件，比如我们最常用的电视机面板控制按钮，遥控器按钮。其实就是一个常开的开关，按下后两个触点接触形成通路状态，松开时形成开路状态。

1. 实验任务

当有键按下，对应的 LED 灯亮。

K1 － K8 对应 P0 端口的 LED　D00 － D07

K1 键按下后，D00 亮。

……

K8 键按下后，D07 亮

在确认有按键按下时，蜂鸣器会响一声。

2. 实验电路

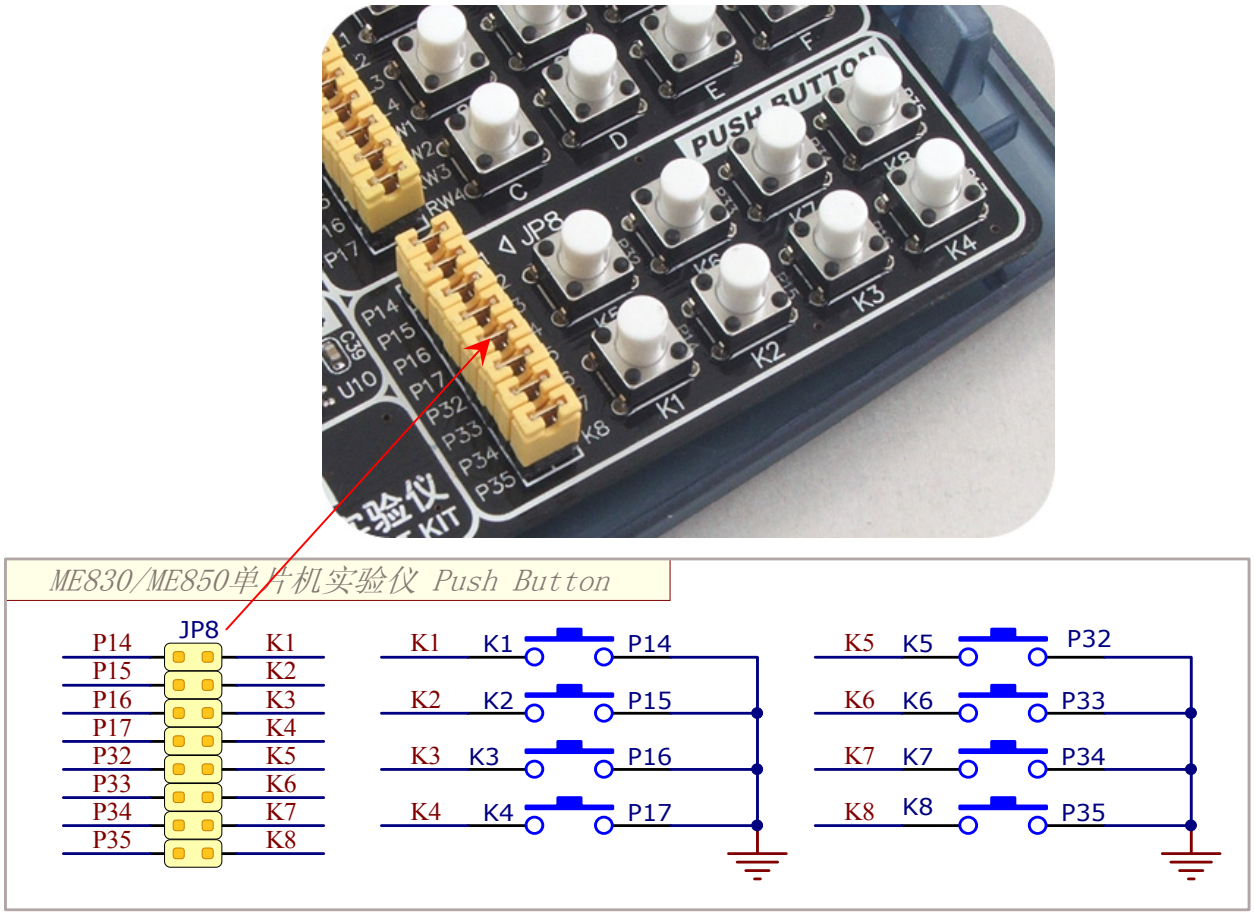


图 6.14　ME830 独立按键

ME830 上共有 8 个独立按键 K1－K8 ，分别通过短接跳线组 JP8 连接到 P14-P17、P32－P35 端口 LED 显示电路已经在前面介绍，见图 6.1

3. 实验步骤

将 JP8 的 8 个短接子全部用短接帽短接，使独立按键与相应的端口接通；

将 JP13 的 VCC-VCC3 短接子用短接帽短接，使 VCC 向发光二极管 D00-D07 供电；

将 JP15 短接子用短接帽短接，使蜂鸣器接口电路工作使能；

如果 PS2 接口插了键盘，请拔下键盘插头或者取下 JP12 上的短接帽。

4. 程序流程图

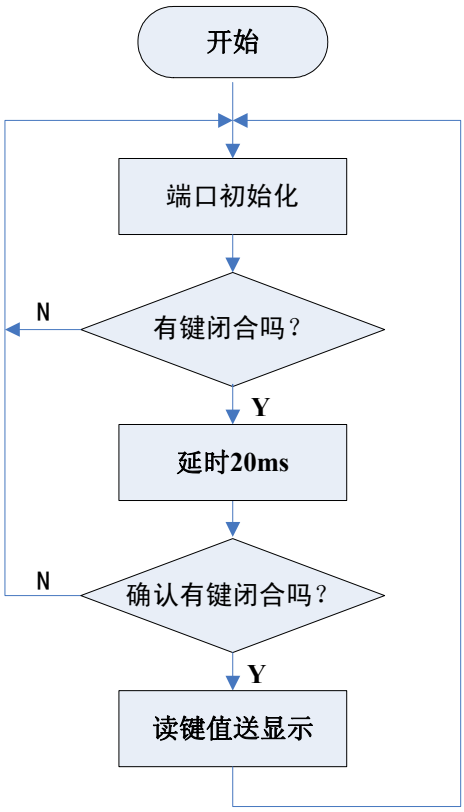


图 6.16　EX6_KEY 流程图

5. 汇编源程序

（光盘：Example_A51\EX6_KEY）

```
;*********************************************************************
;*                                                                   *
;* ME830 单片机开发系统演示程序  K1-K8 键状态指示                    *
;*                                                                   *
;* 有键按下，对应的 LED 灯亮                                         *
;*                                                                   *
;* 版本：  V1.0 (2008/09/30)                                         *
;* 作者：  gguoqing (Email: gguoqing@willar.com)                     *
;* 网站：  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界) *
;* 邮箱：  willar@tom.com                                            *
;*                                                                   *
;*【版权】Copyright(C) 深圳硕飞科技有限公司  All Rights Reserved    *
;*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！       *
;*                                                                   *
;*********************************************************************
        BEEP   BIT    P3.7
;-------------------------------------------------------------
        ORG   0000H
        AJMP   MAIN
        ORG   0050H
;-------------------------------------------------------------
MAIN:
        MOV   SP,#60H
        MOV   P2,#0FFH
LOOP0:
        MOV   P0,#0FFH          ;关闭所有 LED 显示
        MOV   P1,#0FFH          ;为输入状态
        MOV   P3,#0FFH

        ACALL   SCANKEY         ;第一次判键
        CJNE   A,#0FFH,LOOP1
        AJMP   LOOP0
LOOP1:
        MOV   R5,#2             ;延时 20ms
        ACALL   DELAY
```

```
        ACALL   SCANKEY          ;第二次判键
        CJNE    A,#0FFH,LOOP2
        AJMP    LOOP0

LOOP2:
        MOV   P0,A               ;键值送显示
          ACALL   BEEP_BL
        AJMP    LOOP0

;---------------------------------------------------------

; 键盘扫描子程序

;---------------------------------------------------------
SCANKEY:
        MOV   P1,#0FFH          ;为输入状态
        MOV   P3,#0FFH

        MOV   R0,P1             ;读 P1 端口
        MOV   R1,P3             ;读 P3 端口

        MOV   A,R0
        ANL   A,#0F0H           ;保留高四位
        SWAP   A                    ;低四位与高四位数据互换
        MOV   R0,A              ;低四位为有效位(K1-K4)

        MOV   A,R1
        RL    A                  ;左移两次
        RL    A
        ANL   A,#0F0H           ;高四位为有效位(K5-K8)
        ORL   A,R0              ;重新组合，键值保存在 A
        RET

;---------------------------------------------------------
;
;蜂鸣器响一声子程序
;
;---------------------------------------------------------
BEEP_BL:
          MOV   R6,#200
BL1:
          ACALL   BL2
          CPL   BEEP                ;蜂鸣器取反产生驱动脉冲
          DJNZ   R6,BL1
          SETB   BEEP               ;关闭蜂鸣器
          MOV   R5,#15              ;延时 150ms,防止键连击
          ACALL   DELAY
          RET
BL2:
          MOV   R7,#220
BL3:
          NOP
          DJNZ   R7,BL3
          RET

;---------------------------------------------------------

; 延时子程序

;---------------------------------------------------------
DELAY:                              ;延时 R5×10MS
          MOV   R6,#50
DEL1:
          MOV   R7,#93
DEL2:
          DJNZ   R7,DEL2
          DJNZ   R6,DEL1
          DJNZ   R5,DELAY
          RET

;---------------------------------------------------------
          END                     ;结束
;---------------------------------------------------------
```

6. C 语言源程序

（光盘：Example_C51\EX6_KEY）

```
/*********************************************************************
*                                                                    *
* ME830 单片机开发实验仪演示程序 - K1-K8 键状态指示                  *
*                                                                    *
* 有键按下，对应的 LED 灯亮                                          *
*                                                                    *
* 版本：  V1.0 (2008/09/29)                                          *
* 作者：  gguoqing (Email: gguoqing@willar.com)                      *
* 网站：  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界) *
* 邮箱：  willar@tom.com                                             *
*                                                                    *
*【版权】Copyright(C)伟纳电子 www.willar.com  All Rights Reserved  *
*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！        *
*                                                                    *
*********************************************************************/

#include <reg52.h>

char code reserve[3]_at_ 0x3b; //保留 0x3b 开始的 3 个字节

sbit BEEP = P3 ^ 7;

/*******************************************************

延时子函数

*******************************************************/
void delayms(unsigned int ms)
{
  unsigned int t;
  while (ms--)
  {
    for (t = 0; t < 114; t++)
      ;
  }
}

/*******************************************************

x*0.14MS 延时子函数

*******************************************************/
void delayus(unsigned char x)
{
  unsigned char i;
  while (x--)
  {
    for (i = 0; i < 14; i++)
    {
      ;
    }
  }
}

/*******************************************************

蜂鸣器驱动子函数

*******************************************************/
void beep()
{
  unsigned char i;
  for (i = 0; i < 200; i++)
  {
    delayus(6);
    BEEP = !BEEP; //BEEP 取反
  }
  BEEP = 1; //关闭蜂鸣器
  delayms(150); //延时
}
```

```
/*************************************************************

键盘扫描子函数

 *************************************************************/
unsigned char scankey()
{
  unsigned char keynum, keynum1, keynum2;
  P1 = 0xff; //为输入状态
  P3 = 0xff; //为输入状态

  keynum1 = P1; //读 P1 端口
  keynum2 = P3; //读 P3 端口

  keynum1 = (keynum1 &0xf0) >> 4; //低四位为有效位(K1-K4)
  keynum2 = (keynum2 &0x3c) << 2; //高四位为有效位(K5-K8)

  keynum = keynum1 | keynum2; //组合

  return (keynum);
}

/*************************************************************

主函数

 *************************************************************/
void  main()
{
  unsigned char key;
  P0 = 0xff; //关闭 LED 显示
  P2 = 0xff;

  while (1)
  {
    key = scankey(); //第一次判断
    if (key != 0xff)
    {
      delayms(20); //延时 20ms
      key = scankey(); //第二次判断
      if (key != 0xff)
      {
        P0 = key; //键值送显示
        beep();
      }
    }
    P0 = 0xff; //关闭 LED 显示
  }
}

 /*************************************************************/
```

实验七　外部中断

1. 实验任务

利用单片机的外部中断功能进行计数，然后将计数值输出到数码管上显示。

K5 键 — 计数值加 1（外部中断 0）

K6 键 — 计数值减 1（外部中断 1）

3 位数码管显示，最大计数值 255。

2. 实验线路

见图 6.9，图 6.14

3. 实验步骤

将 JP21 的 8 个短接子全部用短接帽短接，使 DG0～DG7 与 P2 端口接通。

将 JP22 的 9 个短接子全部用短接帽短接，使 A～DP 与 P0 端口接通，VCC 向数码管模块供电。

将 JP8 的 8 个短接子全部用短接帽短接，使独立按键与相应的端口接通。

4. 程序流程图

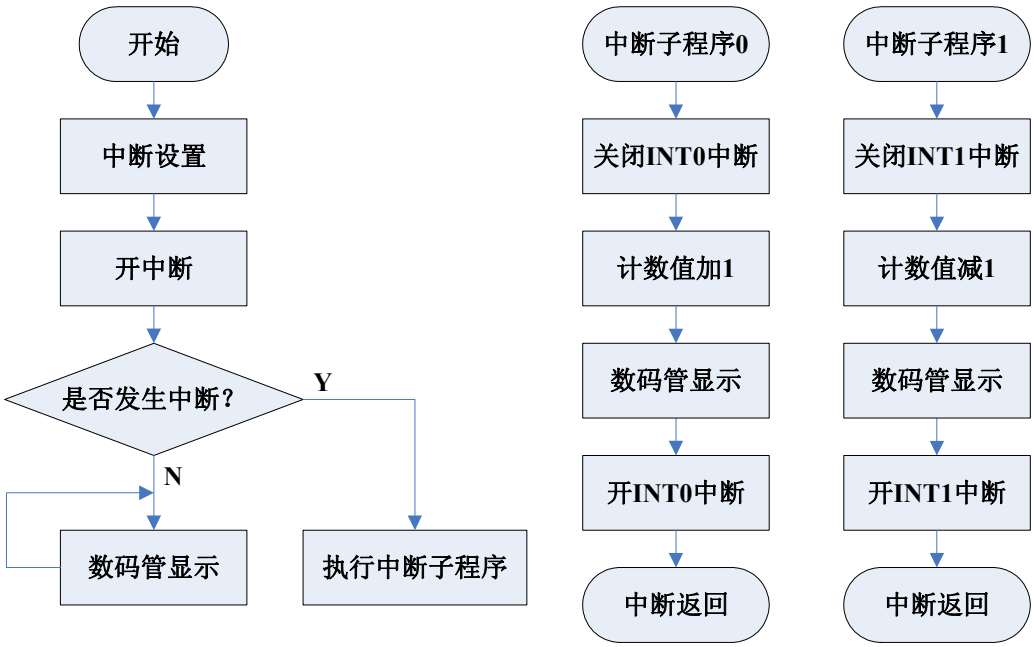


图 6.17　EX7_KEY_INT 流程图

5. 汇编源程序

（光盘：Example_A51\EX7_KEY_INT）

```
;*********************************************************************
;*                                                                   *
;* ME830 单片机开发系统演示程序 - INT0 INT1 中断计数                 *
;*                                                                   *
;* 3 位数码管显示(最大显示 255)                                      *
;*                                                                   *
;* 版本:  V1.0 (2008/08/20)                                          *
;* 作者:  gguoqing (Email: gguoqing@willar.com)                      *
;* 网站:  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界)*
;* 邮箱:  willar@tom.com                                             *
;*                                                                   *
;*【版权】Copyright(C) 深圳硕飞科技有限公司  All Rights Reserved    *
;*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息!         *
;*                                                                   *
;*********************************************************************

        K5    BIT   P3.2
        K6    BIT   P3.3

        DISSTART  EQU   40H     ;显示单元首地址
        LED_DATA  EQU   P0      ;数码管数据口定义
        COUNT     EQU   30H      ;计数单元

;*********************************************************************

        ORG   0000H
        AJMP   MAIN
        ORG   0003H
        AJMP INT0_EX0
        ORG   0013H
        AJMP INT1_EX1
        ORG   0050H

;*********************************************************

; 主程序

;*********************************************************
MAIN:
        MOV   SP,#60H
        MOV   P0,#0FFH
        MOV   P2,#0FFH
        MOV   COUNT,#00H        ;计数单元清零

        MOV   R0,#DISSTART
CLR1:
        MOV   @R0,#00H          ;清显存单元
        INC   R0
        CJNE   R0,#DISSTART+3,CLR1

        CLR    IT0              ;INT0 为电平触发
;       SETB   IT0              ;INT0 为下降沿触发
        CLR    IT1              ;INT1 为电平触发
;       SETB   IT1              ;INT1 为下降沿触发
        SETB   EA
        SETB   EX0
         SETB   EX1
MAIN1:
        ACALL   CONVT
        ACALL   PLAY
        AJMP    MAIN1

;*********************************************************

; INT0 外部中断服务子程序  (加计数)

;*********************************************************
INT0_EX0:
        PUSH   ACC              ;入栈保护
        PUSH   PSW
        SETB   RS0              ;更换寄存器组
```

```
        CLR   RS1
        CLR   EX0           ;关闭 INT0 中断
        INC   COUNT         ;计数值加 1

        MOV   R4,#70
EX0_PLAY:
        MOV A,COUNT         ;用显示程序进行延时
        ACALL CONVT
        ACALL  PLAY
        DJNZ  R4,EX0_PLAY

        SETB  EX0           ;开启 INT0 中断
        POP   PSW           ;出栈
        POP   ACC
        RETI

;*********************************************************

; INT1 外部中断服务子程序  (减计数)

;*********************************************************
INT1_EX1:
        PUSH  ACC           ;入栈保护
        PUSH  PSW
        SETB  RS0           ;更换寄存器组
        CLR   RS1
        CLR   EX1           ;关闭 INT1 中断
        DEC   COUNT         ;计数值减 1

        MOV   R4,#70
EX1_PLAY:
        MOV A,COUNT         ;用显示程序进行延时
        ACALL CONVT
        ACALL  PLAY
        DJNZ  R4,EX1_PLAY

        SETB  EX1           ;开启 INT1 中断
        POP   PSW           ;出栈
        POP   ACC
        RETI

;*********************************************************

;数据转换 (HEX TO BCD)

;*********************************************************
CONVT:
        MOV   A,COUNT         ;计数值处理
        MOV   B,#100
        DIV   AB
        MOV   DISSTART+2,A    ;百位存放在 DISSTART+2
        MOV   A,#10
        XCH   A,B
        DIV   AB
        MOV   DISSTART+1,A    ;十位存放在 DISSTART+1
        MOV   DISSTART,B      ;个位存放在 DISSTART

        MOV   A,DISSTART+2      ;高位为 0，不显示
        CJNE   A,#00H,CONVT1
        MOV   DISSTART+2,#0AH
        MOV   A,DISSTART+1
        CJNE   A,#00H,CONVT1
        MOV   DISSTART+1,#0AH
CONVT1:
        RET

;*********************************************************

; 数码管显示子程序

;*********************************************************
PLAY:
        MOV R0,#DISSTART       ;获得显示单元首地址
        MOV R1,#0FEH           ;从第一个数码管开始
```

```
        MOV R2,#03H             ;共显示 3 位数码管
DISP1:
        MOV A,@R0               ;获得当前位地址
        MOV DPTR,#TAB_NU        ;获得表头
        MOVC A,@A+DPTR          ;查表获得显示数据
        MOV LED_DATA,A          ;送段码
        MOV P2,R1               ;送位码
        MOV A,R1                ;准备下一位的位码
        RL A
        MOV R1,A
        INC R0                  ;取下一个显存单元地址
        ACALL DELAY1MS          ;延时 1 MS
        DJNZ R2,DISP1           ;重复显示下一个
        MOV  P2,#0FFH           ;关闭显示
        RET                     ;显示完成，返回

;*********************************************************

;延时子程序

;*********************************************************
DELAY1MS:
        MOV  R6,#5
DEL1:
        MOV  R7,#93
        DJNZ  R7,$
        DJNZ  R6,DEL1
        RET

;*********************************************************

TAB_NU:
    DB 0C0H,0F9H,0A4H,0B0H,099H,092H,082H,0F8H
    DB 080H,090H,0FFH

;*********************************************************

        END             ;结束

;*********************************************************
```

6. C 语言源程序

见光盘 Example_C51\EX7_KEY_INT

```
/*********************************************************************
 *                                                                   *
 * ME830 单片机开发实验仪演示程序 - INT0 INT1 中断计数                *
 *                                                                   *
 *  51 单片机外部中断功能演示，通过连接在 INT0 和 INT1 的按键 K5 与 K6 实现  *
 *  程序对外部中断次数进行计数，计数值显示在 7 段数码管上             *
 *                                                                   *
 *   INT0(K5) : 计数值递增                                           *
 *   INT1(K6) : 计数值递减                                           *
 *                                                                   *
 * 版本:  V1.0 (2008/09/16)                                          *
 * 作者:  gguoqing (Email: gguoqing@willar.com)                      *
 * 网站:  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界)    *
 * 邮箱:  willar@tom.com                                             *
 *                                                                   *
 *【版权】Copyright(C)伟纳电子 www.willar.com  All Rights Reserved    *
 *【声明】此程序仅用于学习与参考，引用请注明版权和作者信息!           *
 *                                                                   *
 *********************************************************************/

#include < reg52.h >
#include <intrins.h>

unsigned char code LEDData[] =
{
  0xC0, 0xF9, 0xA4, 0xB0, 0x99, 0x92, 0x82, 0xF8, 0x80, 0x90, 0xff
};
//段码
```

```
unsigned char data display[3]; //显示缓存单元

unsigned char code scan_bit[8] =
{
  0xfe, 0xfd, 0xfb, 0xf7, 0xef, 0xdf, 0xbf, 0x7f
};
//位码
unsigned char count; //计数单元

char code reserve[3]_at_ 0x3b; //保留 0x3b 开始的 3 个字节

/*******************************************************
 *                                                     *
 * 延时函数                                            *
 *                                                     *
 *******************************************************/
void delayms(unsigned int ms)
{
  unsigned char k;
  while (ms--)
  {
    for (k = 0; k < 114; k++)
      ;
  }
}

/*******************************************************

数据处理与显示函数

 *******************************************************/
void disp_count()
{
  unsigned char n, temp;

  temp = count;

  for (n = 0; n <= 1; n++)
  //数据处理
  {
    display[n] = temp % 10;
    temp = temp / 10;
  }
  display[2] = temp; //百位数据

  for (n = 2; n > 0; n--)
  //高位为 0，不显示
  {
    if (display[n] == 0)
      display[n] = 0x0a;
    else
      break;
  }

  for (n = 0; n < 3; n++)
  {
    P0 = LEDData[display[n]]; //显示段码
    P2 = scan_bit[n]; //输出位码
    delayms(1);
    P2 = 0xff; //关闭显示
  }
}

/*******************************************************
 *                                                     *
 * 主程序                                              *
 *                                                     *
 *******************************************************/
void main(void)
{
  P0 = 0xff;
  P1 = 0xff;
  P2 = 0xff;
```

```
    IT0 = 0; //低电平触发
    //   IT0=1;              //下降沿触发
    IT1 = 0; //低电平触发
    //   IT1=1;              //下降沿触发

    EA = 1; //总中断允许
    EX1 = 1; //开启 INT1 中断
    EX0 = 1; //开启 INT0 中断

    while (1)
    {
      disp_count(); //数码管显示
    }
}

/*********************************************************
 *                                                       *
 * INT0 中断函数   (加计数)                               *
 *                                                       *
 *********************************************************/
void INT0_ISR(void)interrupt 0
{
  unsigned char x;

  EX0 = 0; //关闭 INT0 中断
  count++; //计数值加 1

  for (x = 0; x < 70; x++)
  //用显示程序进行延时
  {
    disp_count();
  }
  EX0 = 1; //开启 INT0 中断
}

/*********************************************************
 *                                                       *
 * INT1 中断函数   (减计数)                               *
 *                                                       *
 *********************************************************/
void INT1_ISR(void)interrupt 2
{
  unsigned char x;

  EX1 = 0; //关闭 INT1 中断
  count--; //计数值减 1

  for (x = 0; x < 70; x++)
  //用显示程序进行延时
  {
    disp_count();
  }
  EX1 = 1; //开启 INT1 中断
}

/*********************************************************/
```

实验八　矩阵键盘识别

1. 实验任务

1 位数码管显示矩阵键盘的按下键的键值。

开机时，数码管初始显示"-"，当键按下时，数码管显示按下键的键值，蜂鸣器响一声。

2. 实验电路

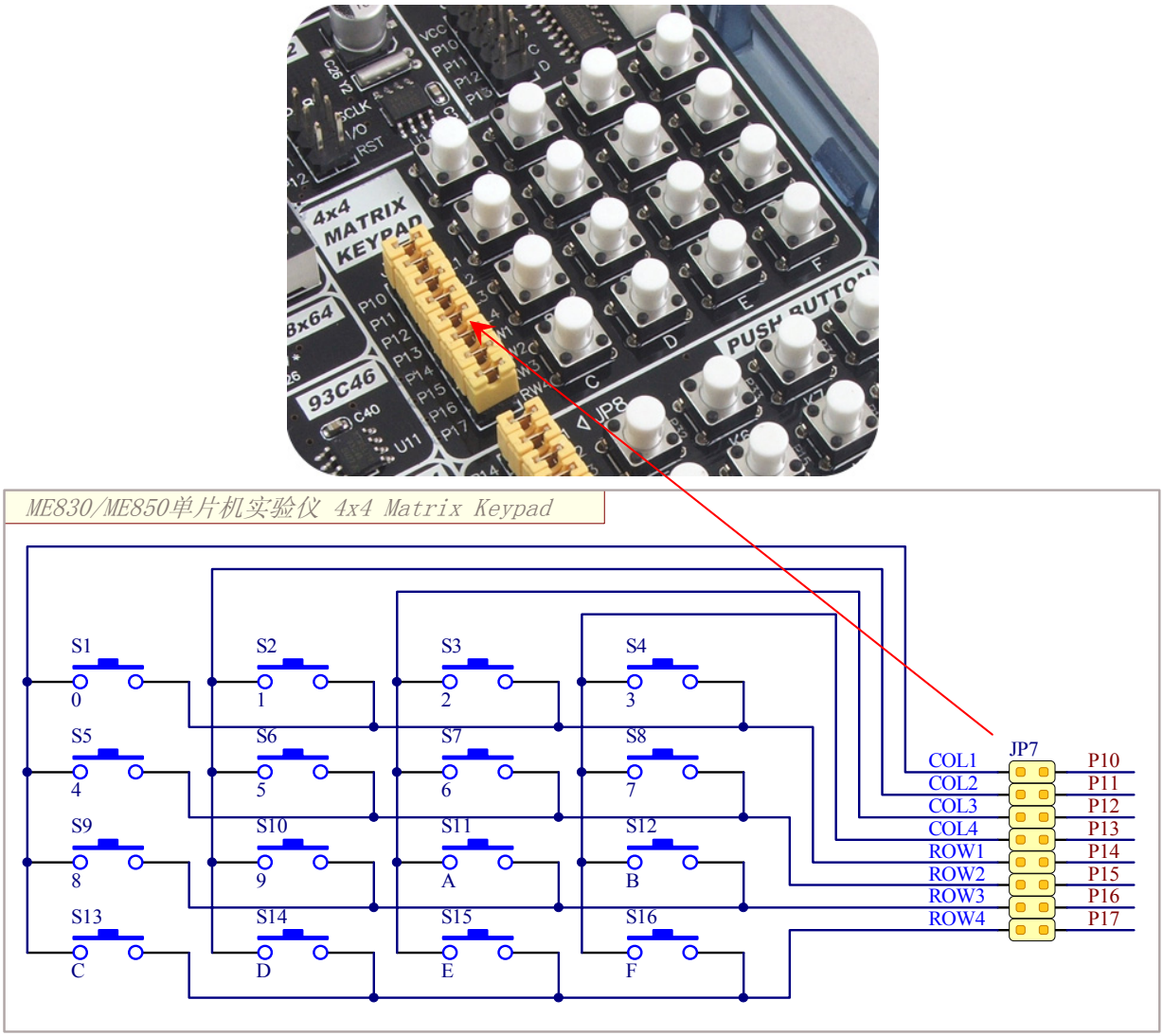


图 6.18　矩阵键盘电路

数码管部分电路请回顾前面有关章节！

4×4 矩阵键盘的编程方法：

（1）先读取键盘的状态，得到按键的特征编码。

先从 P1 口的高四位输出低电平，低四位输出高电平，从 P1 口的低四位读取键盘状态。再从 P1 口的低四位输出低电平，高四位输出高电平，从 P1 口的高四位读取键盘状态。将两次读取结果组合起来就可以得到当前按键的特征编码。使用上述方法我们得到 16 个键的特征编码。

举例说明如何得到按键的特征编码：

假设“1”键被按下，找其按键的特征编码。

从 P1 口的高四位输出低电平，即 P1.4—P1.7 为输出口。低四位输出高电平，即 P1.0—P1.3 为输入口。读 P1 口的低四位状态为“1101”，其值为“0DH”。

再从 P1 口的高四位输出高电平，即 P1.4—P1.7 为输入口。低四位输出低电平，即 P10—P13 为输出口，读 P1 口的高四位状态为“1110”，其值为“E0H”。

将两次读出的 P0 口状态值进行逻辑或运算就得到其按键的特征编码为“EDH”。

用同样的方法可以得到其它 15 个按键的特征编码。

（2）根据按键的特征编码，查表得到按键的顺序编码。

将用上述方法得到的 16 个按键的特征编码按图 1.5 按键编号排列的顺序排成一张特征编码与顺序编码的对应关系表，然后用当前读得的特征编码来查表，当表中有该特征编码时，它所在的位置就是对应的顺序编码。

（3）矩阵键盘键值查找程序的具体编程

1、识别键盘有无按键按下，若无键按下返回。

2、如果有键按下，找出具体的按键值（顺序码）。

3. 实验步骤

将 JP21 的 8 个短接子全部用短接帽短接，使 DG0～DG7 与 P2 端口接通。

将 JP22 的 9 个短接子全部用短接帽短接，使 A～DP 与 P0 端口接通，VCC 向数码管模块供电。

将 JP7 的 8 个短接子全部用短接帽短接，使矩阵按键与 P1 端口接通。

将 JP6（步进电机）短接子上的短接帽全部取掉，否则矩阵键盘将不能正常工作。

4. 程序流程图

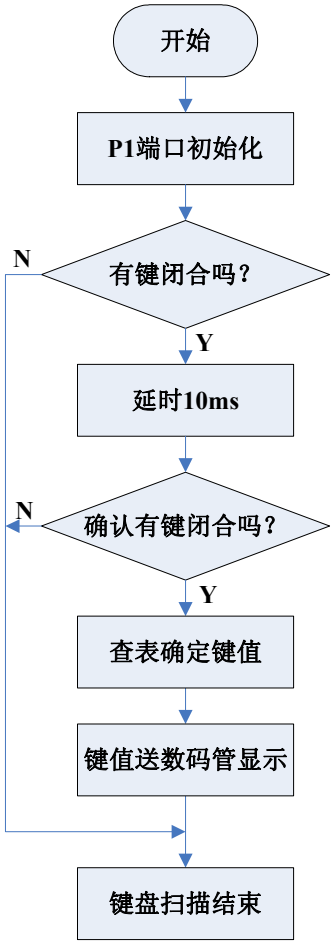


图 6.19　EX8_KEY_4X4 流程图

5. 汇编源程序

（光盘： Example_A51\EX8_KEY_4X4）

```
;*********************************************************************
;* ME830 单片机开发系统演示程序 - 4x4 键盘键值显示                    *
;*                                                                   *
;* 一位数码管显示                                                    *
;*                                                                   *
;* 版本： V1.1 (2008/07/02)                                          *
;* 作者： gguoqing (Email: gguoqing@willar.com)                      *
;* 网站： www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界)*
;* 邮箱： willar@tom.com                                             *
;*                                                                   *
;*【版权】Copyright(C) 深圳硕飞科技有限公司  All Rights Reserved    *
;*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！        *
;*                                                                   *
;*********************************************************************
;
;矩阵键盘定义：
;P1.0-P1.3 为列线,P1.4-P1.7 为行线;

;*************************************************

        BEEP    BIT  P3.7
        KEYNUM  EQU  30H

        ORG  0000H
        AJMP  MAIN
        ORG  0050H

;*************************************************
; 主程序
;*************************************************
MAIN:
        MOV  SP,#60H
        MOV  KEYNUM,#10H        ;开机时显示"-"
        ACALL  KEY_PLAY
LOOP:
        ACALL  KEY_SCAN
        AJMP  LOOP

;*************************************************
;矩阵键盘键值查找程序
;键值存入 30H 单元
;*************************************************
KEY_SCAN:
        MOV  P1,#0F0H           ;置列线为 0，行线为 1
        MOV  A,P1               ;读入 P1 口状态
        ANL  A,#0F0H            ;保留高 4 位
        MOV  B,A                ;保存数据
        MOV  P1,#0FH            ;置列线为 1，行线为 0
        MOV  A,P1               ;读入 P1 口状态
        ANL  A,#0FH             ;保留低 4 位
        ORL  A,B                ;高四位与低四位重新组合
        CJNE  A,#0FFH,KEY_IN1   ;0FFH 为末按键
        AJMP  KEY_END
KEY_IN1:
        MOV  B,A                ;保存键值
        MOV  DPTR,#KEYTABLE     ;置键编码表首址
        MOV  R3,#0FFH           ;
KEY_IN2:
        INC  R3                 ;查表次数加 1
        MOV  A,R3
        MOVC  A,@A+DPTR         ;取出键码
        CJNE  A,B,KEY_IN3       ;比较
        MOV  A,R3               ;找到，取次数值
        MOV  KEYNUM,A           ;送显示单元
        ACALL  KEY_PLAY         ;显示键值
        ACALL  BEEP_BL          ;蜂鸣器响一声
        AJMP  KEY_END
KEY_IN3:
        CJNE  A,#00H,KEY_IN2   ;继续查 ;00H 为结束码
KEY_END:
        RET
```

```
;*************************************************************
; 键编码表
;*************************************************************
KEYTABLE:

    DB   0EEH,0EDH,0EBH,0E7H,0DEH
    DB   0DDH,0DBH,0D7H,0BEH,0BDH
    DB   0BBH,0B7H,07EH,07DH,07BH
    DB   077H,00H                    ;00H 为结束码

;*************************************************************
;;蜂鸣器响一声子程序;
;*************************************************************
BEEP_BL:
        MOV   R6,#200
  BL1:
        ACALL   BL2
        CPL   BEEP              ;蜂鸣器取反产生驱动脉冲
        DJNZ   R6,BL1
        SETB   BEEP             ;关闭蜂鸣器
        MOV   R5,#25
        ACALL   DELAY
        RET
  BL2:
        MOV   R7,#220
  BL3:
        NOP
        DJNZ   R7,BL3
        RET

;*************************************************************

; 延时子程序

;*************************************************************
DELAY:                          ;延时 R5×10MS
        MOV   R6,#50
DEL1:
        MOV   R7,#100
        DJNZ   R7,$
        DJNZ   R6,DEL1
        DJNZ   R5,DELAY
        RET

;*************************************************************

;键值显示子程序

;*************************************************************
KEY_PLAY:
        MOV   A,KEYNUM          ;要显示的数据
        MOV   DPTR,#TABLE       ;置段码表地址
        MOVC   A,@A+DPTR        ;查显示数据段码
        MOV   P0,A              ;输出段码至 P0
        CLR   P2.0              ;第一个数码管亮
        RET

;*************************************************************

; 数码管段码表

;*************************************************************
TABLE:
 DB   0C0H,0F9H,0A4H,0B0H,99H,92H,82H,0F8H
 DB   80H,90H,88h,83h,0c6h,0a1h,86h,8eh,0BFH   ;0－F,-

;*************************************************************
            END                 ;结束
;*************************************************************
```

6. C 语言源程序

（光盘： Example_C51\EX8_KEY_4X4）

```
/*********************************************************************
 *                                                                   *
 * ME830 单片机开发实验仪演示程序 - 4x4 键盘键值显示                 *
 *                                                                   *
 * 一位数码管显示                                                    *
 *                                                                   *
 * 版本： V1.2 (2008/07/01)                                          *
 * 作者： gguoqing (gguoqing@willar.com)                             *
 * 网站： www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界) *
 * 邮箱： willar@tom.com                                             *
 *                                                                   *
 *【版权】Copyright(C)伟纳电子 www.willar.com  All Rights Reserved  *
 *【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！       *
 *                                                                   *
 *********************************************************************
 *                                                                   *
 * 功能：                                                            *
 * 根据扫描键盘返回的键值编码查键值编码表，从而得到键值送数码管显示 *
 * 开机时，数码管显示“－”。                                         *
 * 当键按下时，数码管显示按下键的键值，蜂鸣器响一声。               *
 *                                                                   *
 *********************************************************************/

#include <reg52.h>
#include <intrins.h>

sbit BEEP = P3 ^ 7; //蜂鸣器驱动线

unsigned char key;

unsigned char code disp_code[] =
{
  //显示码数组
  0xc0, 0xf9, 0xa4, 0xb0, 0x99, 0x92, 0x82, 0xf8, 0x80, 0x90, 0x88,
  0x83, 0xc6, 0xa1, 0x86, 0x8e, 0xbf
};

unsigned char code key_code[] =
{
  //键编码数组
  0xee, 0xed, 0xeb, 0xe7, 0xde, 0xdd, 0xdb, 0xd7, 0xbe, 0xbd, 0xbb,
  0xb7, 0x7e, 0x7d, 0x7b, 0x77
};

char code reserve[3]_at_ 0x3b; //保留 0x3b 开始的 3 个字节

/*********************************************************

延时子函数

 *********************************************************/
void delayms(unsigned int ms)
{
  unsigned char t;
  while (ms--)
  {
    for (t = 0; t < 114; t++)
      ;
  }
}

/*********************************************************

x*0.14MS 延时子函数

 *********************************************************/
void delayus(unsigned char x)
{
  unsigned char i;
  while (x--)
  {
```

```
    for (i = 0; i < 14; i++)
    {
      ;
    }
  }
}

/*************************************************************

蜂鸣器驱动子函数

 *************************************************************/
void beep()
{
  unsigned char i;
  for (i = 0; i < 250; i++)
  {
    delayus(6);
    BEEP = !BEEP; //BEEP 取反
  }
  BEEP = 1; //关闭蜂鸣器
  delayms(150); //延时
}

/*************************************************************

键盘扫描子函数

 *************************************************************/
unsigned char keyscan()
{
  unsigned char scan1, scan2, keycode, j;

  P1 = 0xf0;
  scan1 = P1;
  if (scan1 != 0xf0)
  //判键是否按下
  {
    delayms(10); //延时 10ms
    scan1 = P1;
    if (scan1 != 0xf0)
    //二次判键是否按下
    {
      P1 = 0x0f;
      scan2 = P1;
      keycode = scan1 | scan2; //组合成键扫描编码

      for (j = 0; j < 16; j++)
      {
        if (keycode == key_code[j])
        //查表得键值
        {
          key = j;
          return (key); //返回有效键值
        }
      }
    }
  }
  else
    P1 = 0xff;

  return (key = 16); //返回无效码
}

/*************************************************************

主函数

 *************************************************************/
void main(void)
{
  P0 = 0xbf; //数码管初始显示"-"
  P2 = 0xfe;
  P1 = 0xff;
```

```
    while (1)
    {
      keyscan();
      if (key < 16)
      //有效键值
      {
        P0 = disp_code[key]; //显示键值
        beep(); //蜂鸣器响一声
      }
    }
}

/*********************************************************/
```

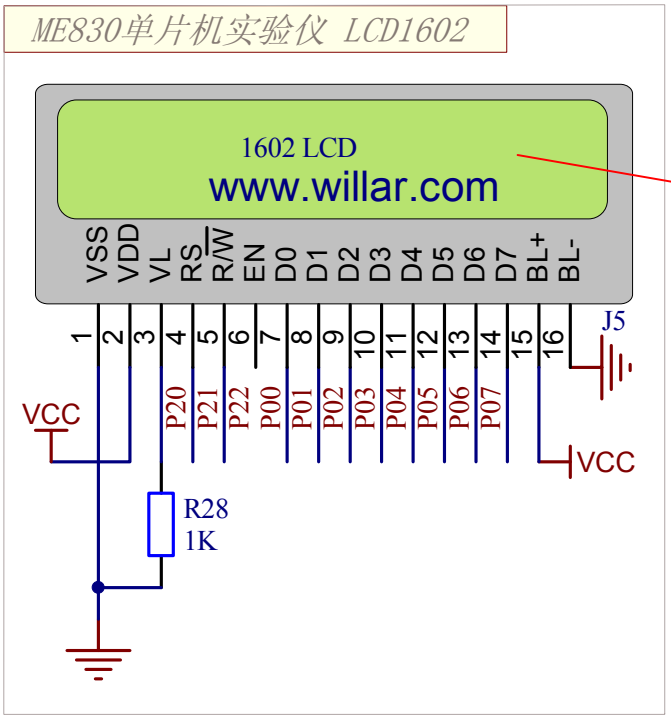
实验九　1602 LCD 显示

1. 实验任务

在 1602LCD 上逐字移出显示两组字符串信息，如此循环。

WELCOME TO WWW.WILLAR.COM ME830　MCU DEVELOPMENT KIT

2. 实验电路



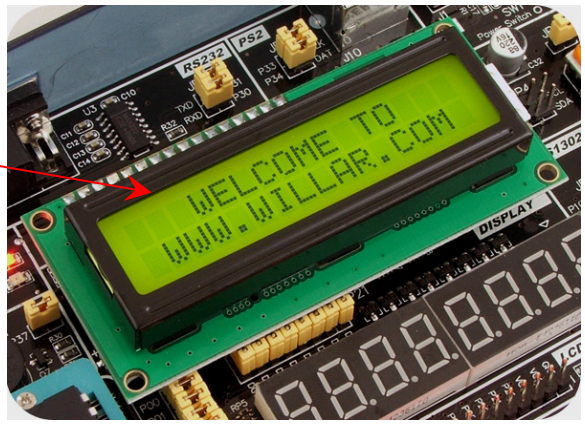


图 6.20　LCD1602 接口原理图

3. 实验步骤

将 1602LCD 模块插在 J5 接口上，注意屏幕方向不要插反

断开数码管的供电跳线帽（JP22 跳线组 Vcc 端的跳线帽）

4. 程序流程图

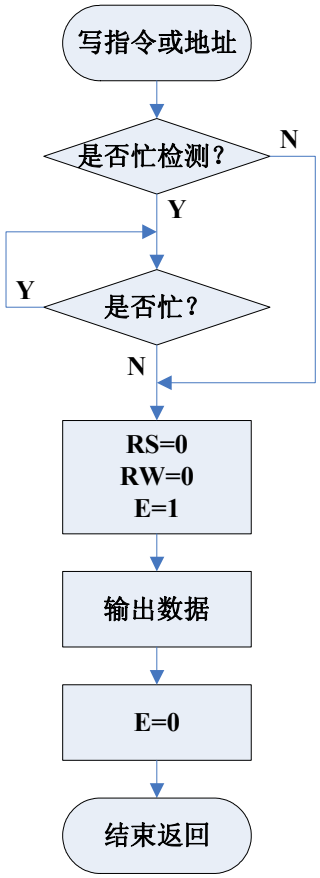


图 6.21 写指令或地址流程图

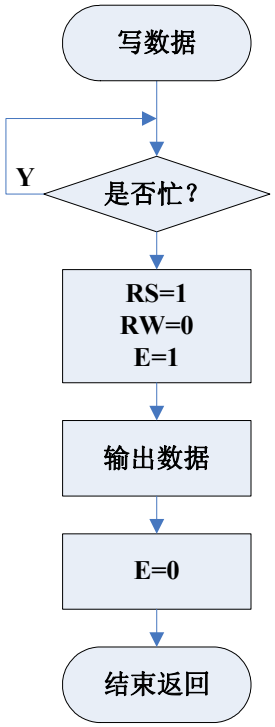


图 6.22 写数据流程图

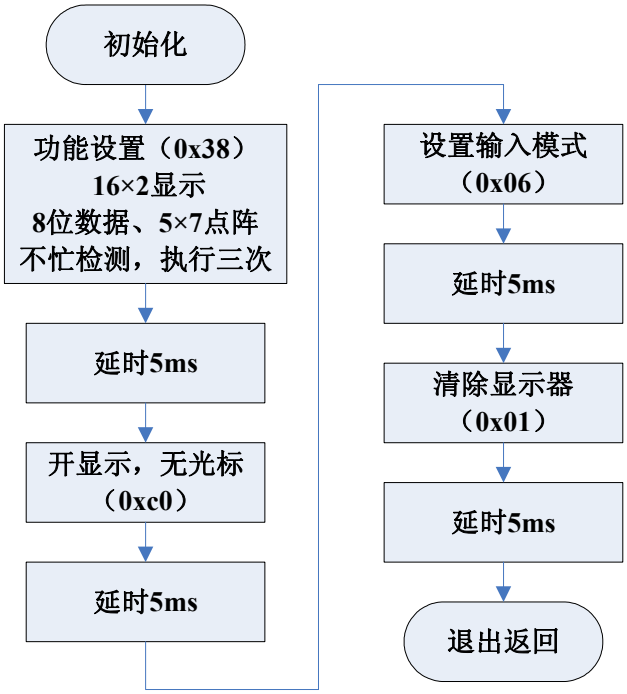


图 6.23 初始化流程图

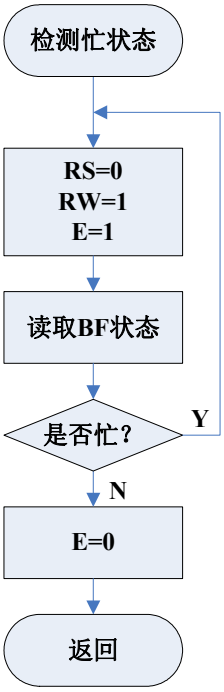


图 6.24　忙检测流程图

5. 汇编源程序

（光盘： Example_A51\EX9_LCD1602）

```
/*********************************************************************
 *                                                                   *
 * ME830 单片机开发实验仪演示程序 - LCD1602 显示                      *
 *                                                                   *
 * 版本:  V1.0 (2008/08/12)                                          *
 * 作者:  gguoqing (gguoqing@willar.com)                             *
 * 网站:  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界) *
 * 邮箱:  willar@tom.com                                             *
 *                                                                   *
 *【版权】Copyright(C)伟纳电子 www.willar.com  All Rights Reserved  *
 *【声明】此程序仅用于学习与参考，引用请注明版权和作者信息!         *
 *                                                                   *
 *********************************************************************/

#include <reg52.h>
#include <intrins.h>

//#define uchar unsigned char
//#define uint  unsigned int

#define DATA_PORT  P0

sbit LCD_RS = P2 ^ 0;
sbit LCD_RW = P2 ^ 1;
sbit LCD_EN = P2 ^ 2;

unsigned char code cdis1[] =
{
  "   WELCOME TO   "
};
unsigned char code cdis2[] =
{
  " WWW.WILLAR.COM "
};

unsigned char code cdis3[] =
{
  "   ME830  MCU   "
};
unsigned char code cdis4[] =
```

```
{
  "DEVELOPMENT KIT "
};

char code reserve[3]_at_ 0x3b; //保留 0x3b 开始的 3 个字节

/*************************************************************

us 延时子程序  (4.34us)

 *************************************************************/
void delayNOP()
{
  _nop_();
  _nop_();
  _nop_();
  _nop_();
}

/*************************************************************

ms 延时子程序

 *************************************************************/
void delayms(unsigned int ms)
{
  unsigned char k;
  while (ms--)
  {
    for (k = 0; k < 114; k++)
      ;
  }
}

/*************************************************************

检查 LCD 忙状态

lcd_busy 为 1 时，忙，等待。
lcd-busy 为 0 时,闲，可写指令与数据。

 *************************************************************/
void lcd_busy()
{
  bit busy;
  busy = 1;
  while (busy)
  {
    LCD_RS = 0;
    LCD_RW = 1;
    LCD_EN = 1;
    busy = (bit)(DATA_PORT &0x80);
    delayNOP();
  }
  LCD_EN = 0;
}

/*************************************************************

写指令数据到 LCD

RS=L，RW=L，EN 下降沿执行写操作。D0-D7=指令码。
Check=1，进行忙检测。

 *************************************************************/
void lcd_wcmd(unsigned char cmd, bit Check)
{
  if (Check)
  lcd_busy();        //进行忙检测

  LCD_RS = 0;
  LCD_RW = 0;
  LCD_EN = 1;
  DATA_PORT = cmd;
```

```
  delayNOP();
  LCD_EN = 0;
}

/*********************************************************

写显示数据到 LCD

RS=H，RW=L，EN 下降沿执行写操作。D0-D7=数据。

 *********************************************************/
void lcd_wdat(unsigned char dat)
{
  lcd_busy();          //进行忙检测
  LCD_RS = 1;
  LCD_RW = 0;
  LCD_EN = 1;
  DATA_PORT = dat;
  delayNOP();
  LCD_EN = 0;
}

/*********************************************************

LCD 初始化设定

 *********************************************************/
void lcd_init()
{
  delayms(15);
  lcd_wcmd(0x38, 0); //16*2 显示，5*7 点阵，8 位数据
  delayms(5);
  lcd_wcmd(0x38, 0); //不进行忙检测，强制执行三次。
  delayms(5);
  lcd_wcmd(0x38, 0);
  delayms(5);

  lcd_wcmd(0x38, 1); //进行忙检测
  delayms(5);
  lcd_wcmd(0x0c, 1); //显示开，关光标
  delayms(5);
  lcd_wcmd(0x06, 1); //移动光标
  delayms(5);
  lcd_wcmd(0x01, 1); //清除 LCD 的显示内容
  delayms(5);
}

/*********************************************************

设定显示位置

 *********************************************************/
void lcd_pos(unsigned char xpos, unsigned char ypos)
{
  if (ypos == 0x01)
    lcd_wcmd((xpos | 0x80), 1);
  if (ypos == 0x02)
    lcd_wcmd((xpos | 0xc0), 1);
}

/*********************************************************

写字符串子函数

 *********************************************************/
void wr_string(unsigned char str[])
{
  unsigned char num = 0;

  while (str[num])
  {
    lcd_wdat(str[num++]);
    delayms(150);
  }
```

```
}

/*********************************************************

主函数

 *********************************************************/
void main()
{
  P0 = 0xff; //置 P0 口
  P2 = 0xff; //置 P2 口
  delayms(100); //上电延时

  lcd_init(); //初始化 LCD

  while (1)
  {
    lcd_pos(0, 1); //第一行显示
    wr_string(cdis1);

    lcd_pos(0, 2); //第二行显示
    wr_string(cdis2);

    delayms(2000); //停留 2000ms

    lcd_wcmd(0x01, 1); //清除 LCD 的显示内容
    delayms(5);

    lcd_pos(0, 1); //第一行显示
    wr_string(cdis3);

    lcd_pos(0, 2); //第二行显示
    wr_string(cdis4);

    delayms(2000); //停留 2000ms

    lcd_wcmd(0x01, 1); //清除 LCD 的显示内容
    delayms(5);
  }
}

 /*********************************************************/
```

6. C 语言源程序

（光盘： Example_C51\EX9_LCD1602）

```
/*********************************************************************
 *                                                                   *
 * ME830 单片机开发实验仪演示程序 - LCD1602 显示                     *
 *                                                                   *
 * 版本： V1.0 (2008/08/12)                                          *
 * 作者： gguoqing (gguoqing@willar.com)                             *
 * 网站： www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界) *
 * 邮箱： willar@tom.com                                             *
 *                                                                   *
 *【版权】Copyright(C)伟纳电子 www.willar.com  All Rights Reserved  *
 *【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！       *
 *                                                                   *
 *********************************************************************/

#include <reg52.h>
#include <intrins.h>

//#define uchar unsigned char
//#define uint  unsigned int

#define DATA_PORT  P0

sbit LCD_RS = P2 ^ 0;
sbit LCD_RW = P2 ^ 1;
sbit LCD_EN = P2 ^ 2;

unsigned char code cdis1[] =
```

```
{
  "   WELCOME TO   "
};
unsigned char code cdis2[] =
{
  " WWW.WILLAR.COM "
};

unsigned char code cdis3[] =
{
  "   ME830  MCU   "
};
unsigned char code cdis4[] =
{
  "DEVELOPMENT KIT "
};

char code reserve[3]_at_ 0x3b; //保留 0x3b 开始的 3 个字节

/*************************************************************

us 延时子程序  (4.34us)

 *************************************************************/
void delayNOP()
{
  _nop_();
  _nop_();
  _nop_();
  _nop_();
}

/*************************************************************

ms 延时子程序

 *************************************************************/
void delayms(unsigned int ms)
{
  unsigned char k;
  while (ms--)
  {
    for (k = 0; k < 114; k++)
      ;
  }
}

/*************************************************************

检查 LCD 忙状态

lcd_busy 为 1 时，忙，等待。
lcd-busy 为 0 时,闲，可写指令与数据。

 *************************************************************/
void lcd_busy()
{
  bit busy;
  busy = 1;
  while (busy)
  {
    LCD_RS = 0;
    LCD_RW = 1;
    LCD_EN = 1;
    busy = (bit)(DATA_PORT &0x80);
    delayNOP();
  }
  LCD_EN = 0;
}

/*************************************************************

写指令数据到 LCD
```

```
RS=L，RW=L，EN 下降沿执行写操作。D0-D7=指令码。
Check=1，进行忙检测。

 ************************************************************/
void lcd_wcmd(unsigned char cmd, bit Check)
{
  if (Check)
  lcd_busy();        //进行忙检测

  LCD_RS = 0;
  LCD_RW = 0;
  LCD_EN = 1;
  DATA_PORT = cmd;
  delayNOP();
  LCD_EN = 0;
}

/************************************************************

写显示数据到 LCD

RS=H，RW=L，EN 下降沿执行写操作。D0-D7=数据。

 ************************************************************/
void lcd_wdat(unsigned char dat)
{
  lcd_busy();          //进行忙检测
  LCD_RS = 1;
  LCD_RW = 0;
  LCD_EN = 1;
  DATA_PORT = dat;
  delayNOP();
  LCD_EN = 0;
}

/************************************************************

LCD 初始化设定

 ************************************************************/
void lcd_init()
{
  delayms(15);
  lcd_wcmd(0x38, 0); //16*2 显示，5*7 点阵，8 位数据
  delayms(5);
  lcd_wcmd(0x38, 0); //不进行忙检测，强制执行三次。
  delayms(5);
  lcd_wcmd(0x38, 0);
  delayms(5);

  lcd_wcmd(0x38, 1); //进行忙检测
  delayms(5);
  lcd_wcmd(0x0c, 1); //显示开，关光标
  delayms(5);
  lcd_wcmd(0x06, 1); //移动光标
  delayms(5);
  lcd_wcmd(0x01, 1); //清除 LCD 的显示内容
  delayms(5);
}

/************************************************************

设定显示位置

 ************************************************************/
void lcd_pos(unsigned char xpos, unsigned char ypos)
{
  if (ypos == 0x01)
    lcd_wcmd((xpos | 0x80), 1);
  if (ypos == 0x02)
    lcd_wcmd((xpos | 0xc0), 1);
}

/************************************************************
```

写字符串子函数

```
 ********************************************************/
void wr_string(unsigned char str[])
{
  unsigned char num = 0;

  while (str[num])
  {
    lcd_wdat(str[num++]);
    delayms(150);
  }
}

/********************************************************

主函数

 ********************************************************/
void main()
{
  P0 = 0xff; //置 P0 口
  P2 = 0xff; //置 P2 口
  delayms(100); //上电延时

  lcd_init(); //初始化 LCD

  while (1)
  {
    lcd_pos(0, 1); //第一行显示
    wr_string(cdis1);

    lcd_pos(0, 2); //第二行显示
    wr_string(cdis2);

    delayms(2000); //停留 2000ms

    lcd_wcmd(0x01, 1); //清除 LCD 的显示内容
    delayms(5);

    lcd_pos(0, 1); //第一行显示
    wr_string(cdis3);

    lcd_pos(0, 2); //第二行显示
    wr_string(cdis4);

    delayms(2000); //停留 2000ms

    lcd_wcmd(0x01, 1); //清除 LCD 的显示内容
    delayms(5);
  }
}

 /********************************************************/
```

实验十 12864 LCD 显示

1. 实验任务

在 12864LCD 上逐字移出显示四组字符串信息和 2 张图片信息，如此循环。

硕飞科技伟纳电子
WWW.WILLAR.COM
ME830_开发实验仪
TEL:077584867757

2. 实验电路

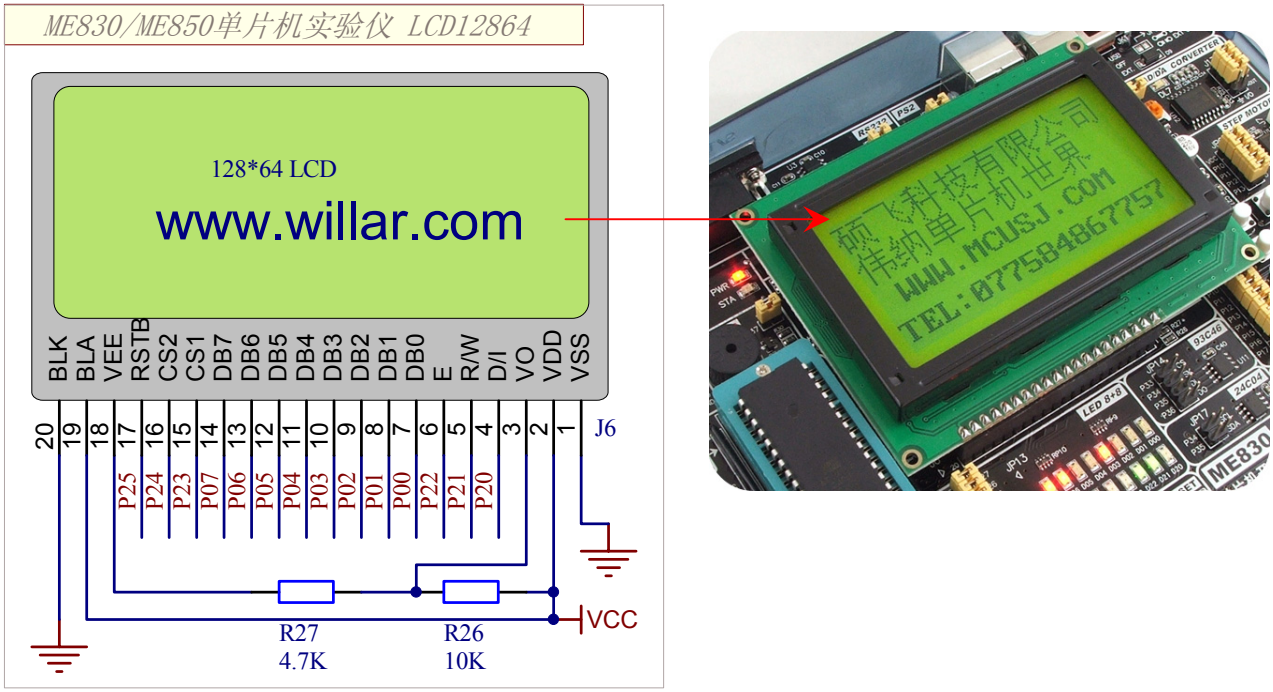


图 6.25 LCD12864 接口原理图

3. 实验步骤

将 12864 液晶模块插到 J6 插座上（注意液晶模块所用的控制器必须是 ST7920 或者兼容的，否则将不能正常显示，因光盘中的例程是针对 ST7920 编写的）；

断开数码管的供电跳线帽（JP22 跳线组 Vcc 端的跳线帽）。

4. 程序流程图

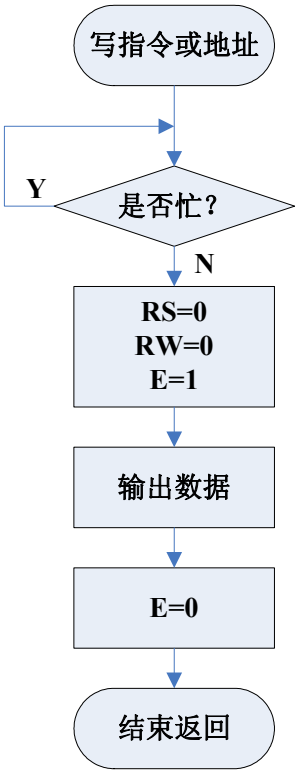


图 6.26 写指令或地址流程图

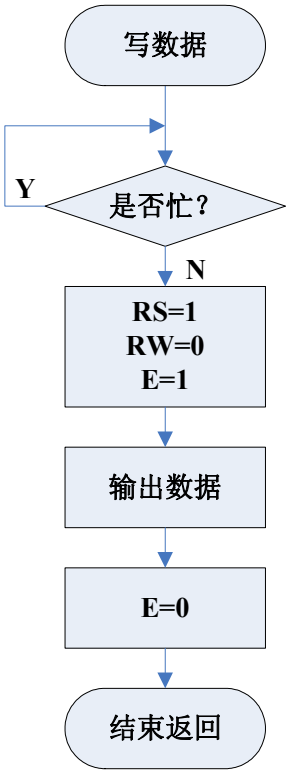


图 6.27 写数据流程图

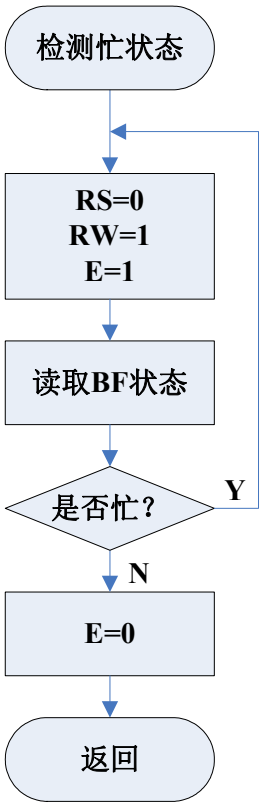


图 6.28 忙检测流程图

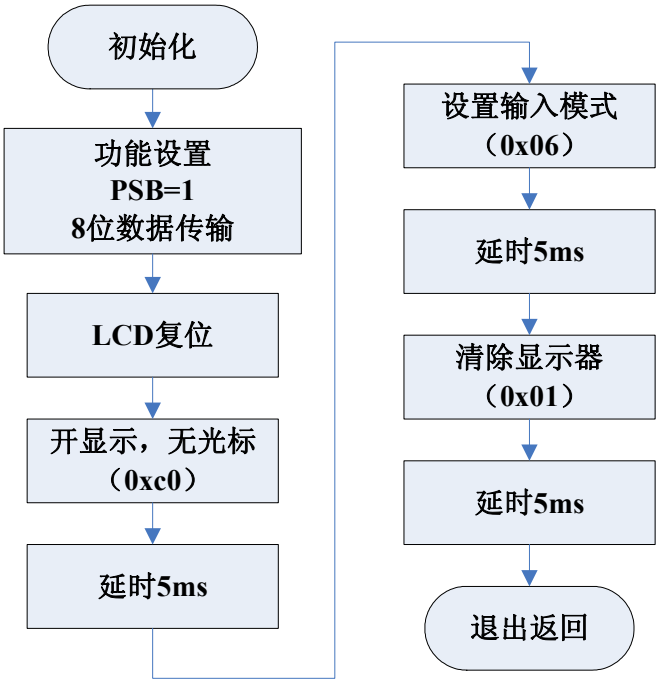


图 6.29　LCD 初始化流程图

5. 汇编源程序

（见光盘： Example_A51\EX10_LCD12864）

6. C 语言源程序

（见光盘： Example_C51\EX10_LCD12864）

提示：12864LCD 显示图片所需要的数据使用“zimo221”软件提取，光盘 Tool 目录下提供了此软件。光盘“Example_A51\EX10_LCD12864” 目录和“Example_C51\EX10_LCD12864”目录均有此软件的使用说明。

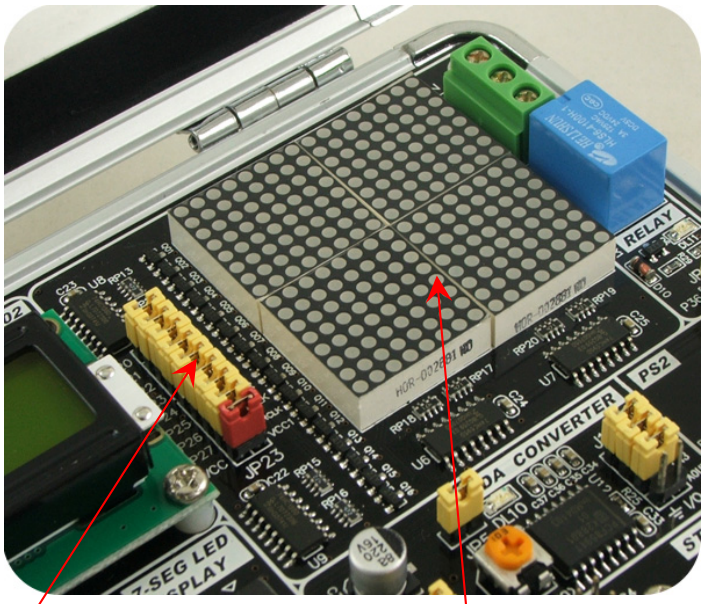
实验十一 16x16 LED 点阵显示

说明：ME830 无此模块（ME850 有此模块）!!!

1. 实验任务

16X16 点阵逐字显示“硕飞科技伟纳电子”，如此循环。

2. 实验电路



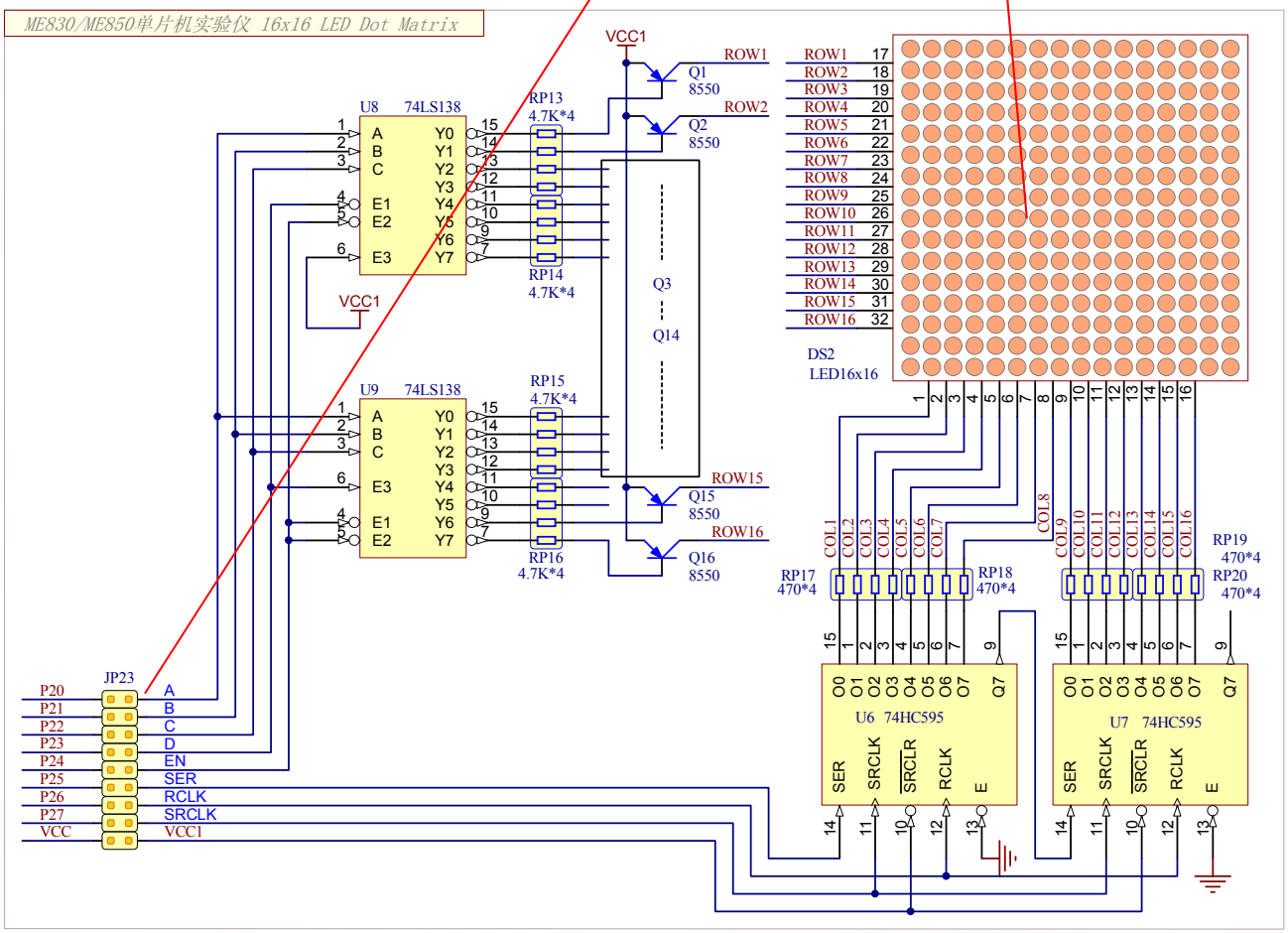


图 6.29 16x16LED 点阵原理图

3. 实验步骤

将 JP23 的 9 个短接子全部用短接帽短接，使点阵接口电路控制端与 P2 端口接通，VCC 向点阵模块供电。

4. 程序流程图

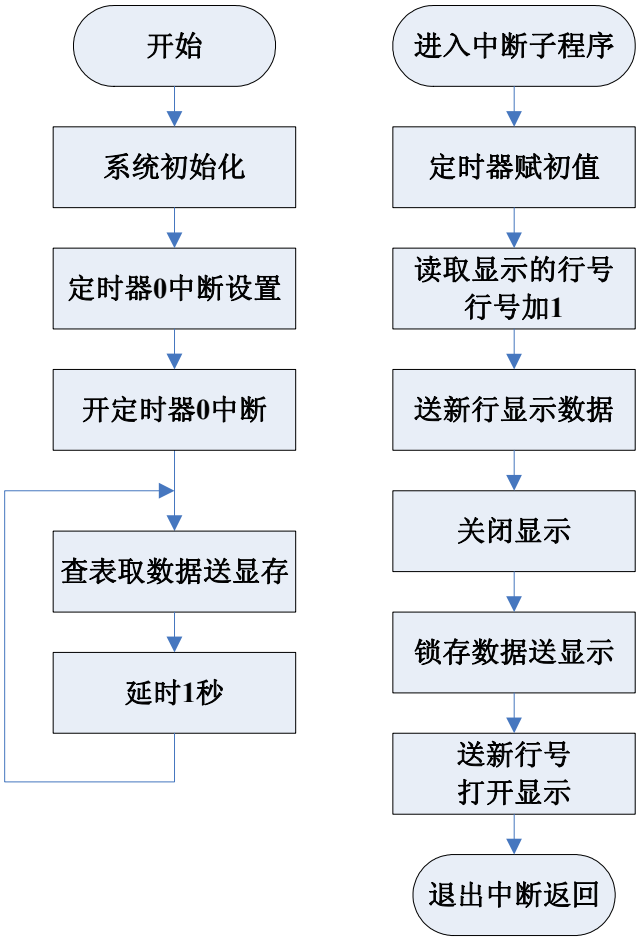


图 6.30 EX11_LED16X16 流程图

5. 汇编源程序

（光盘： Example_A51\EX11_LED16X16）

```
;*********************************************************************
;*                                                                   *
;* ME830 单片机开发系统演示程序 - 显示汉字 硕飞科技伟纳电子          *
;*                                                                   *
;* 16×16LED 点阵显示                                                *
;*                                                                   *
;* 版本:  V1.0 (2008/10/05)                                          *
;* 作者:  gguoqing (gguoqing@willar.com)                             *
;* 网站:  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界) *
;* 邮箱:  willar@tom.com                                             *
;*                                                                   *
;*【版权】Copyright(C) 深圳硕飞科技有限公司  All Rights Reserved    *
;*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息!         *
;*                                                                   *
;*********************************************************************

        SDATA_595   EQU  P2.5   ;串行数据输入
        SCLK_595    EQU  P2.7   ;移位时钟脉冲
        RCK_595     EQU  P2.6   ;输出锁存器控制脉冲

        G_74138     EQU  P2.4   ;显示允许控制信号端口

         ROW_END     BIT  20H.0  ;行扫描完毕标志

        COUN_COMP  EQU  30H   ;计数比较值单元

;*************************************************************
```

```
          ORG    0000H
          AJMP   MAIN
          ORG    000BH
          AJMP   TIMER0

;*********************************************************
MAIN:
          MOV   SP,#70H

          MOV   A,#0FFH
          MOV   P1,A
          MOV   P2,A
          MOV   P3,A
          MOV   P0,A
            CLR   ROW_END
          CLR   RCK_595

          MOV   TMOD,#01H             ;设置定时器 0 工作在定时方式 1
          MOV   TH0,#0FCH            ;1ms 定时常数
          MOV   TL0,#18H
          MOV   IE,#82H               ;允许总中断，允许定时器 0 中断

            MOV   DPTR,#TAB
            MOV   R0,#00H
MAIN1:
          LCALL   DISPLAY
          AJMP    MAIN1

;*********************************************************

; 显示子程序

; R0 — 取数据地址高位 DPH       R1 — 取数据地址低位 DPL
; R2 — 行扫描地址                R3 — 循环显示次数
; B  — 暂存 R1 的过程数据

;*********************************************************
DISPLAY:
          MOV   B,#00H                ;查表偏址暂存（从 00 开始）
            MOV   R1,B
            MOV   COUN_COMP,#00H

D_LOOP:
          MOV   R3,#5AH               ;控制移动速度
D_LOOP1:
          MOV   R2,#00H               ;第 0 行开始
            MOV   R1,B
          SETB   TR0                  ;开扫描（每次一帧）
D_LOOP2:
          JBC   ROW_END,D_LOOP3       ;标志为 1 扫描一帧结束
          AJMP   D_LOOP2
D_LOOP3:
          DJNZ   R3,D_LOOP1           ;一帧重复显示（控制移动速度）

          MOV   A,R1
            MOV   B,A
            CJNE   A,COUN_COMP,D_LOOP   ;8 个字是否显示完?

            MOV   R1,#00H               ;低位计数单元清零
            INC   R0                     ;调整高位计数单元
            MOV   A,R0
            CJNE   A,#2,D_LOOP4

            MOV   R0,#00H              ;高位计数单元清零
            MOV   DPTR,#TAB
            AJMP   LOOPEND
D_LOOP4:
          INC   DPH                  ;调整取码指针高位,取第 9 个字开始地址
            MOV   COUN_COMP,#32           ;1 个汉字 32 个字节数据
            AJMP D_LOOP
LOOPEND:
          RET
```

```
;*********************************************************

; T0 中断扫描子程序

; R1 — 取数据地址;   R2 — 行扫描地址

;*********************************************************
TIMER0:
        PUSH  ACC

        MOV   TH0,#0FCH         ;1ms 定时常数
        MOV   TL0,#18H

        INC   R1                ;取行右边字节偏址
        MOV   A,R1
        MOVC  A,@A+DPTR         ;取行右边字节数据
        LCALL  WR_595

        DEC   R1                ;取行左边字节偏址
        MOV   A,R1
        MOVC  A,@A+DPTR         ;取行左边字节数据
        LCALL  WR_595

        SETB  G_74138           ;关行显示，准备刷新
        NOP
        NOP
        SETB  RCK_595           ;产生上升沿，数据打入输出端
        NOP
        NOP
        CLR   RCK_595           ;恢复低电平

        MOV   A,R2              ;
        MOV   P2,A              ;行输出
        CLR   G_74138           ;开行显示

        INC   R1
        INC   R1                ;下一行数据地址

        INC   R2                ;修改显示行地址
        MOV   A,R2
        ANL   A,#0FH
        JNZ   T0_END            ;一帧扫描是否完毕?
        SETB   ROW_END          ;一帧扫描完,置标记
         CLR   TR0               ;一帧扫描完，关扫描
T0_END:
        POP   ACC
        RETI

;*********************************************************

;移位寄存器接收数据子程序

;*********************************************************
WR_595:
         MOV R4,#08H
WR_LOOP:
         RLC A
         MOV SDATA_595,C
         SETB SCLK_595          ;上升沿发生移位
         NOP
         NOP
         CLR SCLK_595
         DJNZ R4,WR_LOOP
         RET

;*********************************************************
TAB:
;黑屏
DB   0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH
DB   0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH
DB   0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH
DB   0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH

;硕
```

```
    DB   0F7H,0FBH,002H,001H,0EFH,0DFH,0EFH,0BBH
    DB   0DEH,001H,0DAH,0FBH,0C0H,0DBH,09AH,0DBH
    DB   05AH,0DBH,0DAH,0DBH,0DAH,0DBH,0DAH,0DBH
    DB   0C3H,0DFH,0DBH,0A7H,0FFH,07BH,0FCH,0FDH
    ;飞
    DB   0FFH,0DFH,000H,00FH,0FFH,0DFH,0FFH,0DBH
    DB   0FFH,0D3H,0FFH,0CFH,0FFH,0DFH,0FFH,0CFH
    DB   0FFH,0D3H,0FFH,0DBH,0FFH,0DFH,0FFH,0DFH
    DB   0FFH,0EFH,0FFH,0EDH,0FFH,0F5H,0FFH,0FBH
    ;科
    DB   0FBH,0EFH,0F1H,0EFH,007H,06FH,0F7H,0AFH
    DB   0F7H,0EFH,001H,06FH,0F7H,0AFH,0E3H,0EBH
    DB   0E5H,0E1H,0D6H,00FH,0D7H,0EFH,0B7H,0EFH
    DB   077H,0EFH,0F7H,0EFH,0F7H,0EFH,0F7H,0EFH
    ;技
    DB   0EFH,0BFH,0EFH,0BFH,0EFH,0B7H,0ECH,003H
    DB   003H,0BFH,0EFH,0BFH,0EFH,0BFH,0ECH,007H
    DB   0E5H,0F7H,0CEH,0EFH,02EH,0EFH,0EFH,05FH
    DB   0EFH,0BFH,0EFH,04FH,0AEH,0F1H,0D9H,0FBH


    ;伟
    DB   0F7H,0BFH,0F7H,0BFH,0F7H,0BBH,0E8H,001H
    DB   0EFH,0BFH,0CFH,0B7H,0A8H,003H,06FH,0BFH
    DB   0EFH,0BBH,0E0H,001H,0EFH,0BBH,0EFH,0BBH
    DB   0EFH,0ABH,0EFH,0B7H,0EFH,0BFH,0EFH,0BFH

    ;纳
    DB   0EFH,0BFH,0EFH,0BFH,0DFH,0BBH,0DCH,001H
    DB   0B5H,0BBH,005H,0BBH,0EDH,0BBH,0DDH,0BBH
    DB   0BDH,05BH,005H,06BH,0FCH,0EBH,0FDH,0FBH
    DB   0E5H,0FBH,01DH,0FBH,0BDH,0EBH,0FDH,0F7H

    ;电
    DB   0FDH,0FFH,0FDH,0FFH,0FDH,0EFH,080H,007H
    DB   0BDH,0EFH,0BDH,0EFH,080H,00FH,0BDH,0EFH
    DB   0BDH,0EFH,080H,00FH,0BDH,0EFH,0FDH,0FFH
    DB   0FDH,0FBH,0FDH,0FBH,0FEH,003H,0FFH,0FFH

    ;子
    DB   0FFH,0FFH,0C0H,00FH,0FFH,0EFH,0FFH,0DFH
    DB   0FFH,0BFH,0FEH,07FH,0FEH,0FBH,000H,001H
    DB   0FEH,0FFH,0FEH,0FFH,0FEH,0FFH,0FEH,0FFH
    DB   0FEH,0FFH,0FEH,0FFH,0FAH,0FFH,0FDH,0FFH

    ;*********************************************************

            END             ;结束

    ;*********************************************************
```

6. C 语言源程序

（光盘： Example_C51\EX11_LED16X16）

```
    /*********************************************************************
     *                                                                   *
     * ME830 单片机开发实验仪演示程序 - 显示汉字-硕飞科技伟纳电子          *
     *                                                                   *
     * 16×16LED 点阵显示                                                 *
     *                                                                   *
     * 版本：  V1.0 (2008/08/20)                                         *
     * 作者：  gguoqing (gguoqing@willar.com)                            *
     * 网站：  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界) *
     * 邮箱：  willar@tom.com                                            *
     *                                                                   *
     *【版权】Copyright(C)伟纳电子 www.willar.com  All Rights Reserved  *
     *【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！        *
     *                                                                   *
     *********************************************************************/

    #include <reg52.h>         //51 芯片管脚定义头文件
    #include <intrins.h>       //内部包含延时函数 _nop_();
```

```
#define BLKN 2              //列锁存器数

sbit G_74138 = P2 ^ 4;    //显示允许控制信号端口
sbit SDATA_595 = P2 ^ 5; //串行数据输入
sbit RCK_595 = P2 ^ 6;    //输出锁存器控制脉冲
sbit SCLK_595 = P2 ^ 7;   //移位时钟脉冲

char code reserve[3]_at_ 0x3b; //保留 0x3b 开始的 3 个字节

unsigned char data dispram[32]; //显示缓存
unsigned char temp;

unsigned char code Bmp[][32] =
{
  {
    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
  }
  ,  //黑屏

  {
    0xF7, 0xFB, 0x02, 0x01, 0xEF, 0xDF, 0xEF, 0xBB, 0xDE, 0x01, 0xDA, 0xFB,
    0xC0, 0xDB, 0x9A, 0xDB, 0x5A, 0xDB, 0xDA, 0xDB, 0xDA, 0xDB, 0xDA, 0xDB,
    0xC3, 0xDF, 0xDB, 0xA7, 0xFF, 0x7B, 0xFC, 0xFD
  }
  ,  //硕

  {
    0xFF, 0xDF, 0x00, 0x0F, 0xFF, 0xDF, 0xFF, 0xDB, 0xFF, 0xD3, 0xFF, 0xCF,
    0xFF, 0xDF, 0xFF, 0xCF, 0xFF, 0xD3, 0xFF, 0xDB, 0xFF, 0xDF, 0xFF, 0xDF,
    0xFF, 0xEF, 0xFF, 0xED, 0xFF, 0xF5, 0xFF, 0xFB
  }
  ,  //飞

  {
    0xFB, 0xEF, 0xF1, 0xEF, 0x07, 0x6F, 0xF7, 0xAF, 0xF7, 0xEF, 0x01, 0x6F,
    0xF7, 0xAF, 0xE3, 0xEB, 0xE5, 0xE1, 0xD6, 0x0F, 0xD7, 0xEF, 0xB7, 0xEF,
    0x77, 0xEF, 0xF7, 0xEF, 0xF7, 0xEF, 0xF7, 0xEF
  }
  ,  //科

  {
    0xEF, 0xBF, 0xEF, 0xBF, 0xEF, 0xB7, 0xEC, 0x03, 0x03, 0xBF, 0xEF, 0xBF,
    0xEF, 0xBF, 0xEC, 0x07, 0xE5, 0xF7, 0xCE, 0xEF, 0x2E, 0xEF, 0xEF, 0x5F,
    0xEF, 0xBF, 0xEF, 0x4F, 0xAE, 0xF1, 0xD9, 0xFB
  }
  ,  //技

  {
    0xf7, 0xbf, 0xf7, 0xbf, 0xf7, 0xbb, 0xe8, 0x1, 0xef, 0xbf, 0xcf, 0xb7, 0xa8,
    0x3, 0x6f, 0xbf, 0xef, 0xbb, 0xe0, 0x1, 0xef, 0xbb, 0xef, 0xbb, 0xef,
    0xab, 0xef, 0xb7, 0xef, 0xbf, 0xef, 0xbf
  }
  ,  //伟

  {
    0xef, 0xbf, 0xef, 0xbf, 0xdf, 0xbb, 0xdc, 0x1, 0xb5, 0xbb, 0x5, 0xbb, 0xed,
    0xbb, 0xdd, 0xbb, 0xbd, 0x5b, 0x5, 0x6b, 0xfc, 0xeb, 0xfd, 0xfb, 0xe5,
    0xfb, 0x1d, 0xfb, 0xbd, 0xeb, 0xfd, 0xf7
  }
  ,  //纳

  {
    0xfd, 0xff, 0xfd, 0xff, 0xfd, 0xef, 0x80, 0x7, 0xbd, 0xef, 0xbd, 0xef, 0x80,
    0xf, 0xbd, 0xef, 0xbd, 0xef, 0x80, 0xf, 0xbd, 0xef, 0xfd, 0xff, 0xfd,
    0xfb, 0xfd, 0xfb, 0xfe, 0x3, 0xff, 0xff
  }
  ,  //电

  {
    0xff, 0xff, 0xc0, 0xf, 0xff, 0xef, 0xff, 0xdf, 0xff, 0xbf, 0xfe, 0x7f, 0xfe,
    0xfb, 0x00, 0x01, 0xfe, 0xff, 0xfe, 0xff, 0xfe, 0xff, 0xfe, 0xff, 0xfe,
    0xff, 0xfe, 0xff, 0xfa, 0xff, 0xfd, 0xff
```

```
  } //子
};

/*************************************************************

延时子程序

 *************************************************************/
void delayms(unsigned int ms)
{
  unsigned char k;
  while (ms--)
  {
    for (k = 0; k < 114; k++)
      ;
  }
}

/*************************************************************

主函数

 *************************************************************/
void main(void)
{
  unsigned char i, k;

  RCK_595 = 1;
  SCLK_595 = 0;
  P2 = 0xF0;      //行号端口清零并关闭显示

  TMOD = 0x01;    //定时器 T0 工作方式 1
  TH0 = 0xFC;     //1ms 定时常数
  TL0 = 0x66;

  IE = 0x82;     //允许定时器 T0 中断
  TR0 = 1;       //启动定时器 T0

  while (1)
  {
    for (k = 0; k < 9; k++)
    //显示"硕飞科技伟纳电子"
    {
      for (i = 0; i < 32; i++)
      //一个汉字数据有 32 个字节
      {
        dispram[i] = Bmp[k][i]; //数据存入显存单元
      }
      delayms(1000);             //确定字显示时间
    }
  }
}

/*************************************************************

将显示数据送入 74HC595 内部移位寄存器

 *************************************************************/
void WR_595(void)
{
  unsigned char x;
  for (x = 0; x < 8; x++)
  {
    temp = temp << 1; //数据左移一位，最高位送入 CY
    SDATA_595 = CY;

    SCLK_595 = 1;      //上升沿发生移位
    _nop_();
    _nop_();
    SCLK_595 = 0;
  }
}

/*************************************************************
```

```
中断服务函数

 ************************************************************/
void leddisplay(void) interrupt 1
{
  unsigned char row, j = BLKN;

  TH0 = 0xFC; //1ms 定时常数
  TL0 = 0x66;

  row = P2; //读取当前显示的行号
  row = ++row &0x0f; //行号加 1，屏蔽高 4 位

  G_74138 = 1; //关闭显示
  temp = dispram[row *BLKN + 1]; //先写行右边字节
  WR_595();
  temp = dispram[row *BLKN]; //后写行左边字节
  WR_595();

  P2 &= 0xf0; //行号端口清零
  RCK_595 = 1; //上升沿将数据送到输出锁存器
  P2 |= row; //写入行号
  RCK_595 = 0;
  G_74138 = 0; //打开显示
}

 /************************************************************/
```

提示：16x16LED 点阵显示汉字所需要的数据使用“16x16 字模”软件提取，光盘 Tool 目录下提供了此软件。光盘“Example_A51\EX11_LED16X16” 目录和“Example_C51\EX11_LED16X16”目录均有此软件的使用说明。

实验十二　RS232 串口通信

1. 实验任务

先通过串口向计算机发送中英文字符串和字符。

英文字符串： welcome to www.willar.com

中文字符串： 硕飞科技－伟纳电子

然后从机等待接收主机发送来的数据，当从机接收到主机发送来的数据后，将此数据再发送回主机。

2. 实验线路

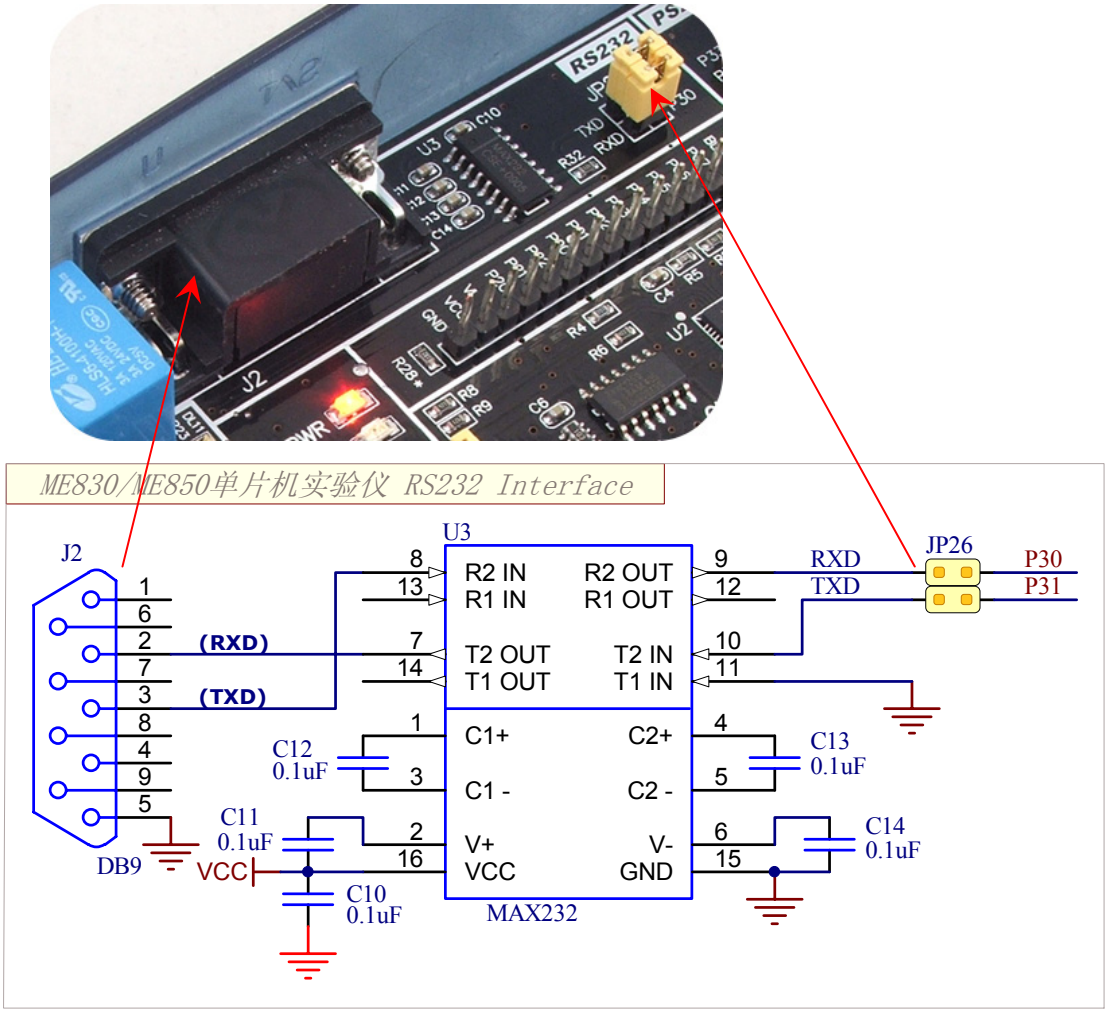


图 6.31　RS232 串口通信电路

3. 实验步骤

短接 JP26 短接子，使芯片的串行端口（RXD-P3.0、TXD-P3.1）与 RS232 接口芯片 MAX232C 连接。

上位机使用伟纳编写的“串口 TT”（com TT）串口调试程序。

1）“串口 TT”参数设定：

端口号：COM1（实际使用的端口号）　波特率：9600

数据位：8　校验位：None　停止位：1

2）将接收信息框（左上信息框）显示模式均设置为文本模式。

3）将发送信息框（左下信息框）显示模式均设置为文本模式。

图 6.32　串口 TT 设置

4. 程序流程图

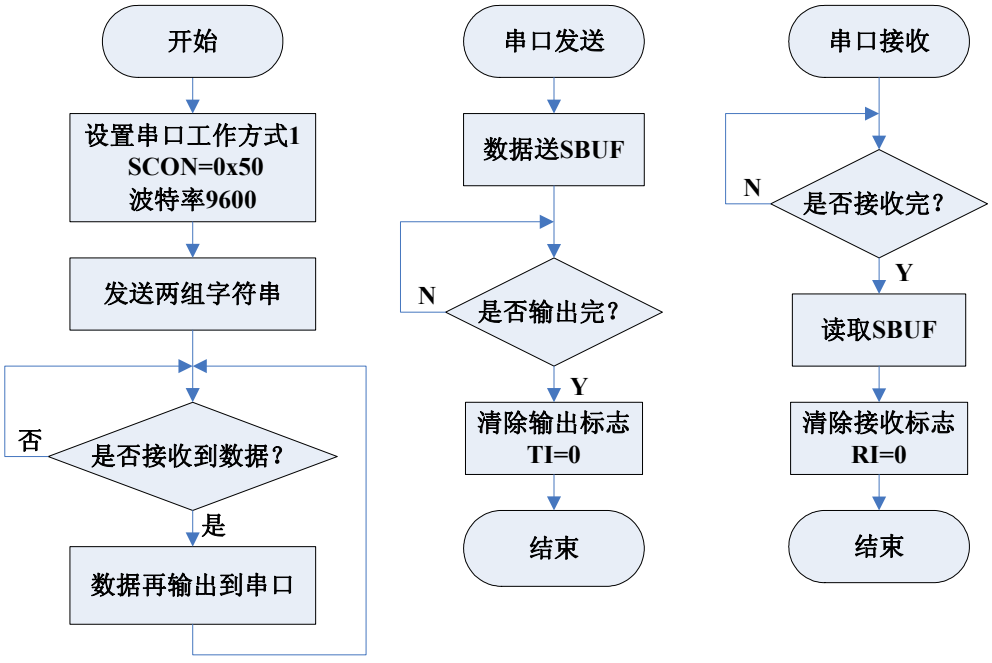


图 6.33　EX12_UART 流程图

5. 汇编源程序

（光盘： Example_A51\EX12_UART）

```
;*********************************************************************
;*                                                                   *
;* ME830 单片机开发系统演示程序 - 串行通迅                           *
;*                                                                   *
;* 工作芯片：AT89S52       晶振频率：11.0592MHz                      *
;*                                                                   *
;* 版本： V1.0 (2008/08/17)                                          *
;* 作者： gguoqing (Email: gguoqing@willar.com)                      *
;* 网站： www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界)*
;* 邮箱： willar@tom.com                                             *
;*                                                                   *
;*【版权】Copyright(C) 深圳硕飞科技有限公司  All Rights Reserved    *
;*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！       *
;*                                                                   *
;*********************************************************************
;*                                                                   *
;* 描述：                                                            *
;*                                                                   *
;* 1、单片机发送字符串给主机                                         *
;* 2、将接收的数据再发回主机                                         *
;*                                                                   *
;*********************************************************************/

        ORG   0000H
        AJMP  MAIN
        ORG   0050H

;*******************************************************

;  主程序

;*******************************************************
MAIN:
        MOV   SP,#60H
        MOV   P0,#0FFH
        MOV   P2,#0FFH

        MOV   TMOD,#20H    ; 定时器 1 工作于 8 位自动重载模式,用于产生波特率
        MOV   TH1,#0FDH
        MOV   TL1,#0FDH    ; 波特率 9600

        MOV   SCON,#50H    ; 设定串行口工作方式 1,接收使能
        ANL   PCON,#00H    ; 波特率不倍增

        SETB  EA           ; 允许总中断
        SETB  TR1          ; 启动定时器 1

         MOV  R5,#100
        ACALL  DELAY

        MOV   DPTR,#TAB_ENG     ;字串表格地址
        ACALL SEND_STRING

        MOV   R5,#100
        ACALL  DELAY
        MOV   DPTR,#TAB_CHS     ;字串表格地址
        ACALL SEND_STRING

        MOV   R5,#100
        ACALL  DELAY
        MOV   A,#'0'            ;发送字符“0”
        ACALL  TXD_CHAR
        MOV   A,#'K'            ;发送字符“K”
        ACALL  TXD_CHAR
        MOV   A,#0AH            ;换行
        ACALL  TXD_CHAR
LOOP:
        ACALL  RXD_CHAR         ;接收数据
         ACALL  TXD_CHAR            ;发送数据
        AJMP   LOOP
```

```
;*********************************************************

; 发送数据子程序

;*********************************************************
TXD_CHAR:
        MOV   SBUF,A     ;发送数据
        JNB   TI,$       ;等特数据传送完毕
        CLR   TI         ;清除中断标志
        RET

;*********************************************************

; 接收数据子程序

;*********************************************************
RXD_CHAR:
        JNB   RI,$       ;等特数据接收完毕
        MOV   A,SBUF     ;接收数据
        CLR   RI         ;清除中断标志
        RET

;*********************************************************

;发送字符串子程序

;*********************************************************
SEND_STRING:
        CLR   A
        MOVC   A,@A+DPTR
        JZ   S_END              ;查到 00H 时,表示字串结束
        ACALL   TXD_CHAR
        INC DPTR                ;下一字符
        SJMP SEND_STRING
S_END:
        RET

;*********************************************************

; 延时 10MS 子程序

;*********************************************************
DELAY:
        MOV   R6,#50
DEL1:
        MOV   R7,#93
DEL2:
        DJNZ   R7,DEL2
        DJNZ   R6,DEL1
        DJNZ   R5,DELAY
        RET

;*********************************************************
TAB_ENG:
     DB  "welcome to www.willar.com "
     DB  0AH        ;换行
     DB  00H

TAB_CHS:
     DB  "硕飞科技－伟纳电子 "
     DB  0AH        ;换行
     DB  00H

;*********************************************************

         END          ;结束

;*********************************************************
```

6. C 语言源程序

（光盘： Example_C51\EX12_UART）

```
/*********************************************************************
 *                                                                   *
 * ME830 单片机开发实验仪演示程序 - 串行通迅                          *
 *                                                                   *
 * 工作芯片：AT89S52       晶振频率：11.0592MHz                      *
 *                                                                   *
 * 版本： V1.0 (2008/09/01)                                          *
 * 作者： gguoqing (gguoqing@willar.com)                             *
 * 网站： www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界)  *
 * 邮箱： willar@tom.com                                             *
 *                                                                   *
 *【版权】Copyright(C)伟纳电子 www.willar.com  All Rights Reserved  *
 *【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！       *
 *                                                                   *
 *********************************************************************
 *                                                                   *
 * 描述：                                                            *
 *                                                                   *
 * 1、单片机发送字符串给主机                                         *
 * 2、将接收的数据再发回主机                                         *
 *                                                                   *
 *********************************************************************/

#include <reg52.h>
#include <intrins.h>

unsigned char code str1[] = " welcome to www.willar.com  \n ";

unsigned char code str2[] = " 硕飞科技－伟纳电子 \n ";

unsigned char code reserve[3]_at_ 0x3b; //保留 0x3b 开始的 3 个字节

/*********************************************************

延时子程序

 *********************************************************/
void delayms(unsigned int ms)
{
  unsigned char k;
  while (ms--)
  {
    for (k = 0; k < 114; k++)
      ;
  }
}

/*********************************************************

发送数据子函数

 *********************************************************/
void txdata(unsigned char dat)
{
  SBUF = dat; //发送数据
  while (!TI)
    ;
  //等待数据发送完中断
  TI = 0; //清中断标志
}

/*********************************************************

接收数据子函数

 *********************************************************/
unsigned char rxdata()
{
  unsigned char dat;

  while (!RI)
```

```
    ;
  //等待数据接收完
  dat = SBUF; //接收数据
  RI = 0; //清中断标志
  return (dat);
}

/*******************************************************

传送字符串函数

*******************************************************/
void send_str(unsigned char str[])
{
  unsigned char i = 0;

  while (str[i] != '\0')
  {
    SBUF = str[i++];
    while (!TI)
      ;
    //等待数据传送完毕
    TI = 0; //清中断标志
  }
}

/*******************************************************

主函数

 *******************************************************/
void main(void)
{
  unsigned char buff;
  P0 = 0xff;
  P2 = 0xff;

  SCON = 0x50; //设定串口工作方式 1，接收使能
  PCON = 0x00; //波特率不倍增

  TMOD = 0x20; //定时器 1 工作于 8 位自动重载模式，用于产生波特率
  EA = 1;
  TL1 = 0xfd;
  TH1 = 0xfd; //波特率 9600
  TR1 = 1;

  delayms(100);

  send_str(str1); //发送英文字符串
  delayms(1000);

  send_str(str2); //发送中文字符串
  delayms(1000);

  txdata('O');
  txdata('K');
  txdata('\n'); //换行
  delayms(1000);

  while (1)
  {
    buff = rxdata(); //接收数据
    txdata(buff);    //发送数据
  }
}

 /*******************************************************/
```

实验十三　74HC164 串转并实验

说明：ME830 无此模块（ME850 有此模块）!!!

1. 程序功能

利用 TTL 芯片 74HC164 进行串口转并口实验。

采用串口方式 0 传输数据，使 74HC164 的输出端所连接的 8 个发光二极管 DL0 - DL7 从右至左轮流点亮。

2. 实验线路

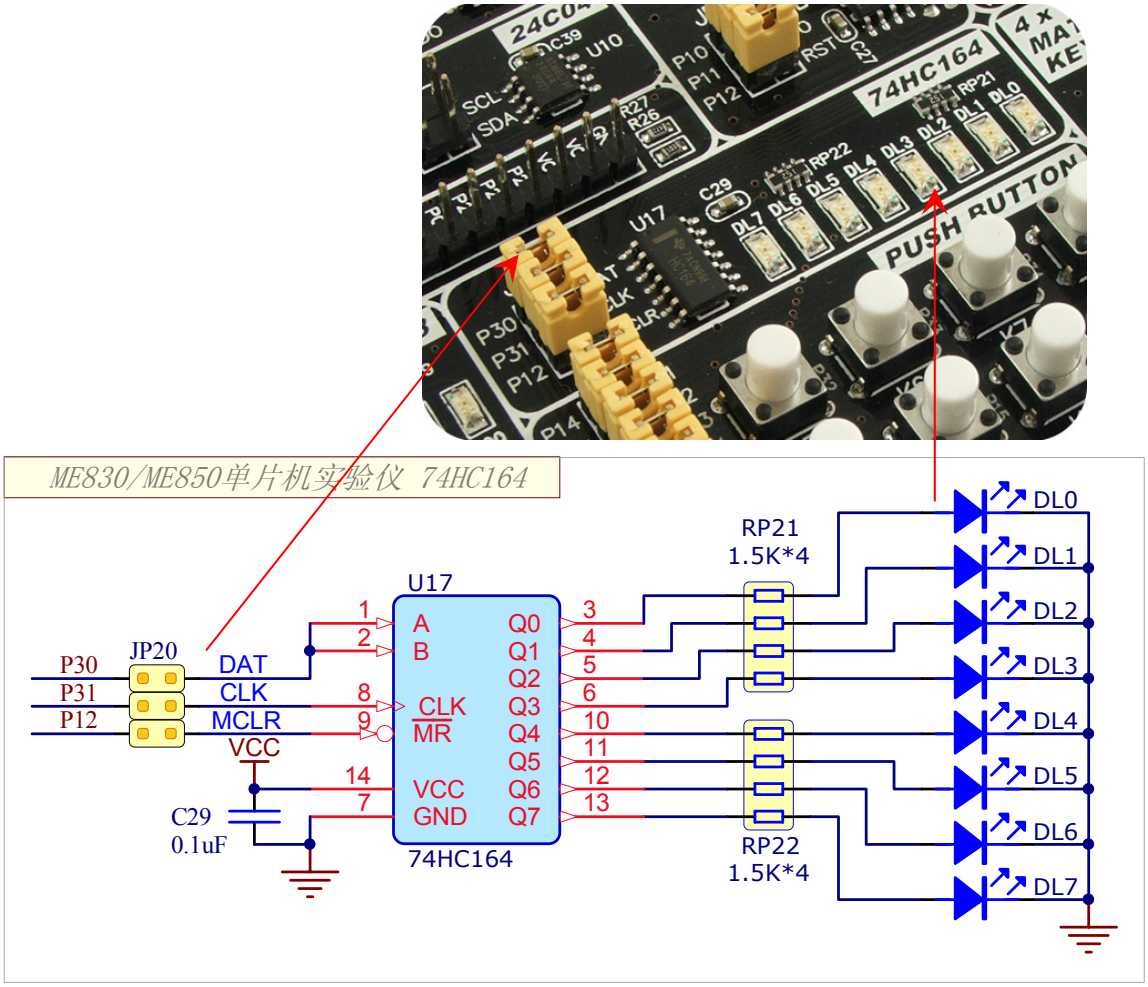


图 6.34　74HC164 原理图

3. 实验步骤

短接 JP20 短接子，使 74HC164 的相应引脚与串行端口（RXD-P3.0、TXD-P3.1）连接；

将 JP6（步进电机）、JP26（MAX232）、JP19（74HC165）短接子上的短接帽全部取掉。

4. 程序流程图

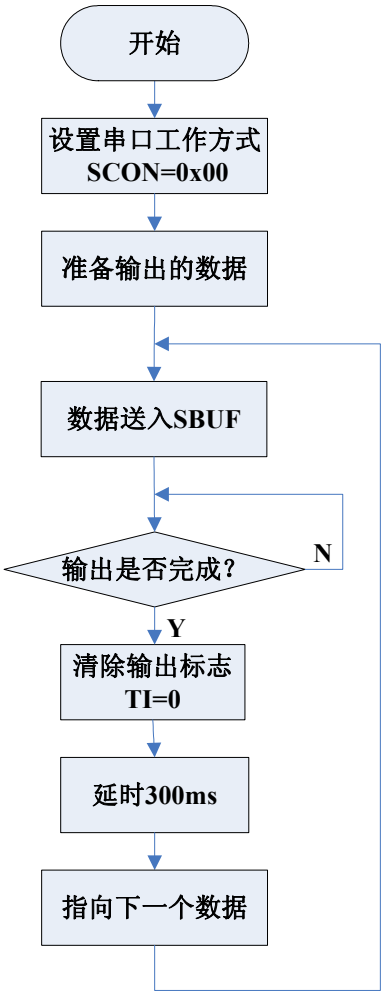


图 6.35　EX13_HC164 流程图

5. 汇编源程序

（光盘： Example_A51\EX13_HC164）

```
;*********************************************************************
;*                                                                   *
;* ME830 单片机开发系统演示程序 - 74HC164 串入转并出演示程序        *
;*                                                                   *
;* 版本:  V1.0 (2008/08/06)                                          *
;* 作者:  gguoqing (gguoqing@willar.com)                              *
;* 网站:  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界)*
;* 邮箱:  willar@tom.com                                             *
;*                                                                   *
;*【版权】Copyright(C) 深圳硕飞科技有限公司  All Rights Reserved    *
;*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息!         *
;*                                                                   *
;*********************************************************************
;*                                                                   *
;* 74HC164 是一个串行输入并行输出的移位寄存器，用于扩展并行输出口。 *
;* 编程采用串行口方式。                                              *
;* 功能:                                                             *
;* 8 个发光二极管流水方式显示                                         *
;*                                                                   *
;*********************************************************************

        DATA_164  BIT  P3.0     ;RXD
        CLK_164   BIT  P3.1     ;TXD
        MR_164    BIT  P1.2     ;MCLR

;***********************************************************

        ORG  0000H
        AJMP  MAIN
```

```
        ORG   0050H

;*********************************************************
; 主程序
;*********************************************************
MAIN:
        MOV   SP,#60H

        MOV   SCON,#00H      ;设置串行口工作方式 0，发送

        CLR   MR_164         ;清 164
        ACALL   DELAY1MS
        SETB   MR_164
MAIN1:
        MOV   R0,#80H        ;赋显示初值
        MOV   R2,#08H        ;8 个发光二极管
MAIN2:
        ACALL   WR_Byte
        ACALL   DELAY        ;延时 300ms
        MOV   A,R0           ;准备下一个显示数据
        RR    A               ;右移一位
        MOV   R0,A           ;保存新数据
        DJNZ   R2,MAIN2

        CLR   MR_164         ;清 164
        ACALL   DELAY1MS
        SETB   MR_164

        ACALL   DELAY        ;延时 300ms

        AJMP   MAIN1

;*********************************************************
; 发送数据子程序
;*********************************************************
WR_Byte:
        MOV   A,R0           ;取数据
        MOV   SBUF,A         ;开始串行输出
W_WAIT:
        JNB   TI,W_WAIT      ;判断数据输出是否完毕
        CLR   TI             ;发送完毕，清中断
        RET

;*********************************************************
;延时子程序     (300ms)
;*********************************************************
DELAY:
        MOV   R5,#03
DEL1:
        MOV   R6,#200
DEL2:
        MOV   R7,#230
DEL3:
        DJNZ   R7,DEL3
        DJNZ   R6,DEL2
        DJNZ   R5,DEL1
        RET

;*********************************************************
;延时子程序     (1ms)
;*********************************************************
DELAY1MS:
        MOV   R6,#2
DEL4:
        MOV   R7,#230
DEL5:
        DJNZ   R7,DEL5
        DJNZ   R6,DEL4
        RET

;*********************************************************
        END                  ;结束
;*********************************************************
```

6. C 语言源程序

（光盘： Example_C51\EX13_HC164）

```
/*********************************************************************
 *                                                                   *
 * ME830 单片机开发实验仪演示程序 - 74HC164 串入转并出演示程序       *
 *                                                                   *
 * 版本： V1.0 (2008/08/06)                                          *
 * 作者： gguoqing (gguoqing@willar.com)                             *
 * 网站： www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界) *
 * 邮箱： willar@tom.com                                             *
 *                                                                   *
 *【版权】Copyright(C)伟纳电子 www.willar.com  All Rights Reserved  *
 *【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！       *
 *                                                                   *
 *********************************************************************
 *                                                                   *
 * 74HC164 是一个串行输入并行输出的移位寄存器，用于扩展并行输出口。  *
 * 编程采用串行口方式。                                              *
 * 功能：                                                            *
 * 8 个发光二极管以流水方式显示。                                    *
 *                                                                   *
 *********************************************************************/

#include <reg52.h>           //51 芯片管脚定义头文件
#include <intrins.h>        //内部包含延时函数 _nop_();

sbit data_164 = P3 ^ 0;
sbit clk_164 = P3 ^ 1;
sbit mr_164 = P1 ^ 2;

char code reserve[3]_at_ 0x3b; //保留 0x3b 开始的 3 个字节

/*********************************************************

延时 t 毫秒
11.0592MHz 时钟，延时约 1ms

 *********************************************************/
void delayms(unsigned int t)
{
  unsigned char k;
  while (t--)
  {
    for (k = 0; k < 114; k++)
      ;
  }
}

/*********************************************************

发送数据子函数

 *********************************************************/
void wr_byte(unsigned char num)
{
  SBUF = num; //发送数据
  while (!TI)
    ;
  //等待数据输出完毕
  TI = 0; //发送完毕，清中断标志
}

/*********************************************************

主函数

 *********************************************************/
void main(void)
{
  unsigned char n, temp;

  SCON = 0x00; //设置串行口工作方式 0，发送
```

```
    mr_164 = 0; //清 164
    delayms(1);
    mr_164 = 1;

    while (1)
    {
      temp = 0x80; //赋显示初值
      for (n = 0; n < 8; n++)
      {
        wr_byte(temp); //写数据，送显示
        delayms(300);
        temp >>= 1; //准备下一个显示数据
        //  temp=temp|0x80;     //最高位置 1
      }
      wr_byte(0x00); //关闭显示
      delayms(300);
    }
}

/************************************************************

如果 temp=temp|0x80，显示效果变为逐个点亮 8 个 LED。

 ************************************************************/
```

实验十四　74HC165 并转串

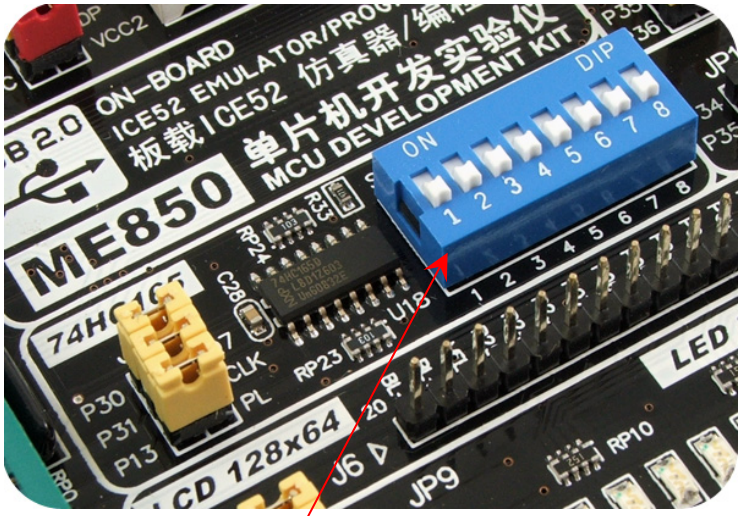
说明：ME830 无此模块（ME850 有此模块）!!!

1. 程序功能

利用 TTL 芯片 74HC165 进行并口转串口实验。

采用串口方式 0 接收数据，读取 74HC165 并行输入口的 8 个拨指开关状态，每种状态读取 2 次，保存第一次读取值并输出到 P2 端口 D20－D27 显示，第二次读取的值与第一次读取值进行比较，如果两次读取值相同输出到 P0 端口 D00－D07 显示，否则关闭 P0 端口 D00－D07 显示。

2. 实验线路



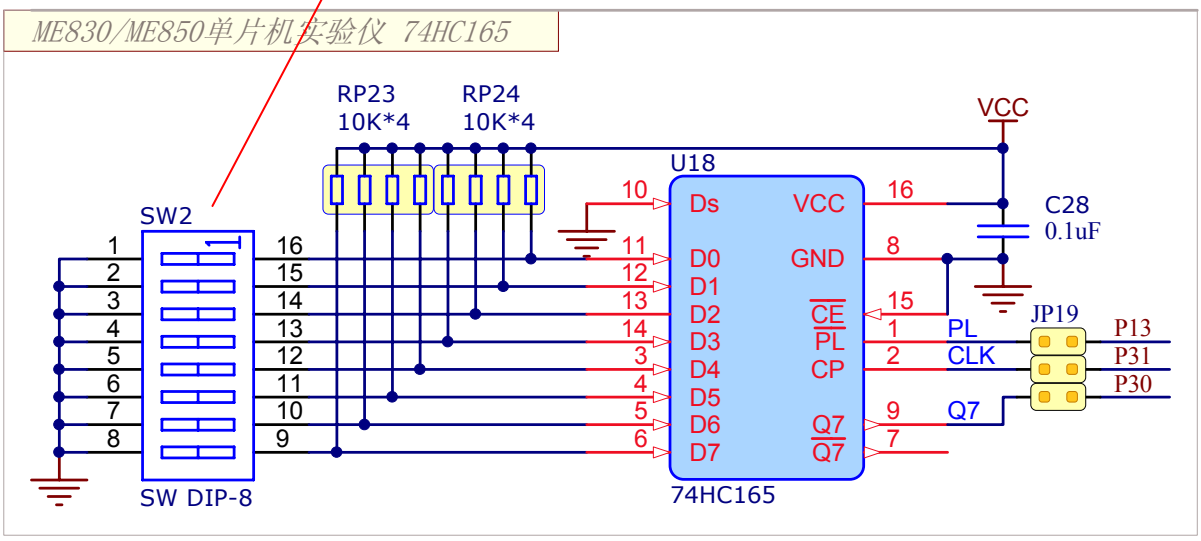


图 6.36　74HC165 原理图

3. 实验步骤

短接 JP19 短接子，使 74HC165 的相应引脚与 CPU 串行端口（RXD-P3.0、TXD-P3.1）连接。

将 JP13 的 9 个短接子全部用短接帽短接，使 D20～D27 与 P2 端口接通，VCC 向发光二极管模块供电。

将 JP6（步进电机）、JP26（MAX232C）、JP20（74HC164） 短接子上的短接帽全部取掉。

4. 程序流程图

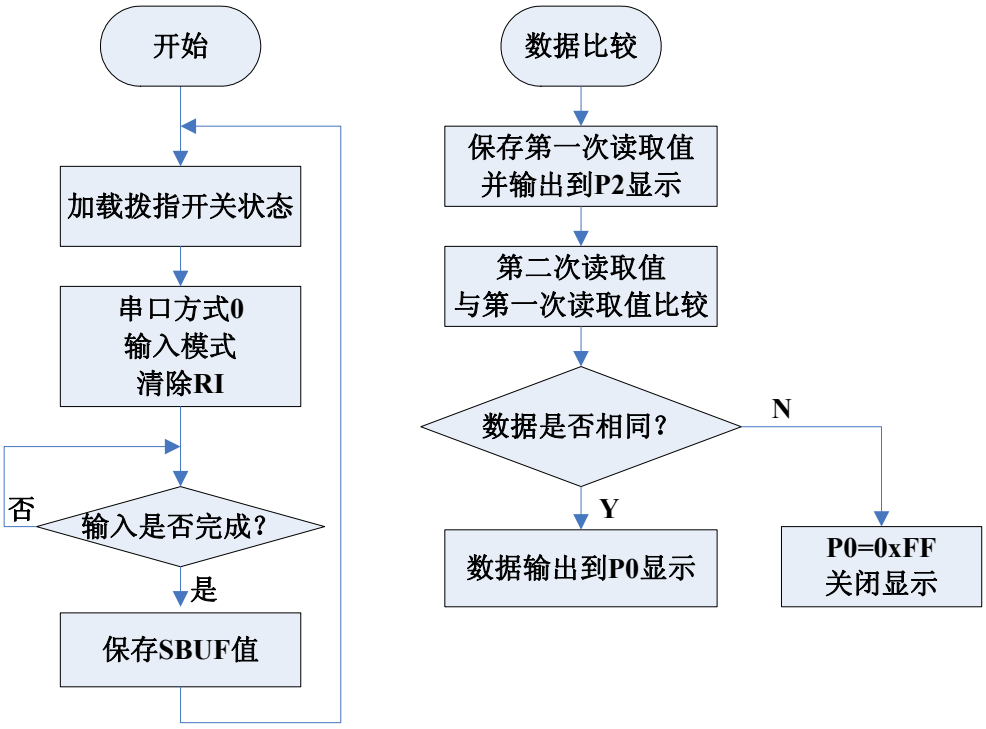


图 6.37　EX14_HC165 流程图

5. 汇编源程序

（光盘： Example_A51\EX14_HC165）

```
;*********************************************************************
;*                                                                   *
;* ME830 单片机开发系统演示程序 - 并入转串出演示程序                  *
;*                                                                   *
;*                                                                   *
;* 版本:  V1.0 (2008/08/16)                                          *
;* 作者:  gguoqing                                                   *
;* 网站:  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界)*
;* 邮箱:  willar@tom.com                                             *
;*                                                                   *
;*【版权】Copyright(C) 深圳硕飞科技有限公司  All Rights Reserved    *
;*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息!         *
;*                                                                   *
;*********************************************************************
;*                                                                   *
;* 74LS165 是一个并行输入串行输出的移位寄存器，用于扩展并行输入口   *
;* 编程采用串行口方式，                                              *
;* 功能:                                                             *
;* 读两次数据，然后进行比较，如果相同则送 ME500 的 D00-D07 显示。     *
;*                                                                   *
;*********************************************************************

        DATA_165  BIT  P3.0
        CLK_165   BIT  P3.1
        LD_165    BIT  P1.3

;*************************************************************

        ORG  0000H
        AJMP  MAIN
        ORG  0050H

;*************************************************************

; 主程序

;*************************************************************
MAIN:
```

```
        MOV   SP,#6FH
        MOV   P0,#0FFH
        MOV   P2,#0FFH
MAIN1:
        ACALL RD_BYTE          ;第一次读数据
        MOV   B,A
        MOV   P2,A             ;送 P2 显示
        ACALL   DELAY10MS      ;延时
        ACALL RD_BYTE          ;第二次读数据
        CJNE   A,B,MAIN1       ;两次读的数据比较
        MOV   P0,A             ;相同，送 P0 显示
        AJMP   MAIN1

;*********************************************************

; 读数据子程序

;*********************************************************
RD_BYTE:
        CLR  LD_165            ;置入并行数据
        NOP
        NOP
        SETB LD_165            ;开始串行移位输出

        MOV   SCON,#10H        ;工作方式 0，接收数据，清除 RI
R_WAIT:
        JNB   RI, R_WAIT       ;等待 RI 串行输入中断
        MOV   A,SBUF           ;读取数据
        RET

;*********************************************************

;10MS 延时子程序

;*********************************************************
DELAY10MS:
        MOV   R6,#20
DEL1:
        MOV   R7,#250
        DJNZ   R7,$
        DJNZ   R6,DEL1
        RET

;*********************************************************

        END

;*********************************************************

;    ; CLR   RI                ;清 RI 标志,启动 RX 控制器的开始信号

; MOV   SCON,#11H          ;设置串行口 mode0,REN=1 程序接收使能
                             ;RI=1 停止 RX 控制器的开始信号
```

6. C 语言源程序

（光盘： Example_C51\EX14_HC165）

```
/*****************************************************************
 *                                                               *
 * ME830 单片机开发实验仪演示程序 - 并入转串出演示程序                *
 *                                                               *
 * 版本： V1.0 (2008/08/07)                                      *
 * 作者： gguoqing (gguoqing@willar.com)                         *
 * 网站： www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界) *
 * 邮箱： willar@tom.com                                         *
 *                                                               *
 *【版权】Copyright(C)伟纳电子 www.willar.com  All Rights Reserved *
 *【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！       *
 *                                                               *
 *****************************************************************
 *                                                               *
 * 74LS165 是一个并行输入串行输出的移位寄存器，用于扩展并行输入口。  *
 * 编程采用串行口方式，                                          *
 * 功能:                                                         *
 * 读两次数据，然后进行比较，如果相同则送 ME830 的 D00-D07 显示。
 *****************************************************************/

#include <reg52.h>           //51 芯片管脚定义头文件
#include <intrins.h>        //内部包含延时函数 _nop_();

sbit DATA_165 = P3 ^ 0; //串口数据
sbit CLK_165  = P3 ^ 1; //串口时钟
sbit LD_165   = P1 ^ 3; //用 P1^3 控制 SH/LD 管脚

char code reserve[3]_at_ 0x3b; //保留 0x3b 开始的 3 个字节

/*************************************************************

延时 t 毫秒
11.0592MHz 时钟，延时约 1ms

 *************************************************************/
void delayms(unsigned int t)
{
  unsigned char k;
  while (t--)
  {
    for (k = 0; k < 114; k++)
    {
      ;
    }
  }
}

/*************************************************************

读数据子函数

 *************************************************************/
unsigned char ReadByte(void)
{
  unsigned char RD_buf;

  LD_165 = 0; //锁存并行输入数据
  delayms(1);
  LD_165 = 1; //开始串行移位输出

  SCON = 0x10; //串行口 mode0，接收数据，清除 RI
  while (!RI)
    ;
  //等待 RI 串行输入中断
  RD_buf = SBUF; //读取数据

  return (RD_buf); //返回所读取的数据
}

/*************************************************************
```

```
主函数

 ************************************************************/
void main(void)
{
  unsigned char temp1, temp2;
  P0 = 0xff;
  P2 = 0xff;

  while (1)
  {
    temp1 = ReadByte(); //第一次读数据
    P2 = temp1; //送 P2 显示
    delayms(10);
    temp2 = ReadByte(); //第二次读数据
    if (temp1 == temp2)
    //两次读的数据比较
      P0 = temp2;
    //相同，送 P0 显示
    else
      P0 = 0xff;
  }
}

//************************************************************
```

实验十五　步进电机控制

步进电机是一种将电脉冲信号变换成相应的角位移的机电执行元件。控制步进电机的输入脉冲数量、频率及电机各项绕组的接通顺序，可以得到各种需要的运行特性。尤其与数字设备配套时，体现了更大的优越性，因此广泛应用于数字控制系统中。

1. 实验任务

采用单双八拍工作方式，控制步进电机正（顺时针）反（逆时针）方向转动。

步进电机正转 360 度（一圈），反转 180 度（半圈），如此循环。

2. 实验线路

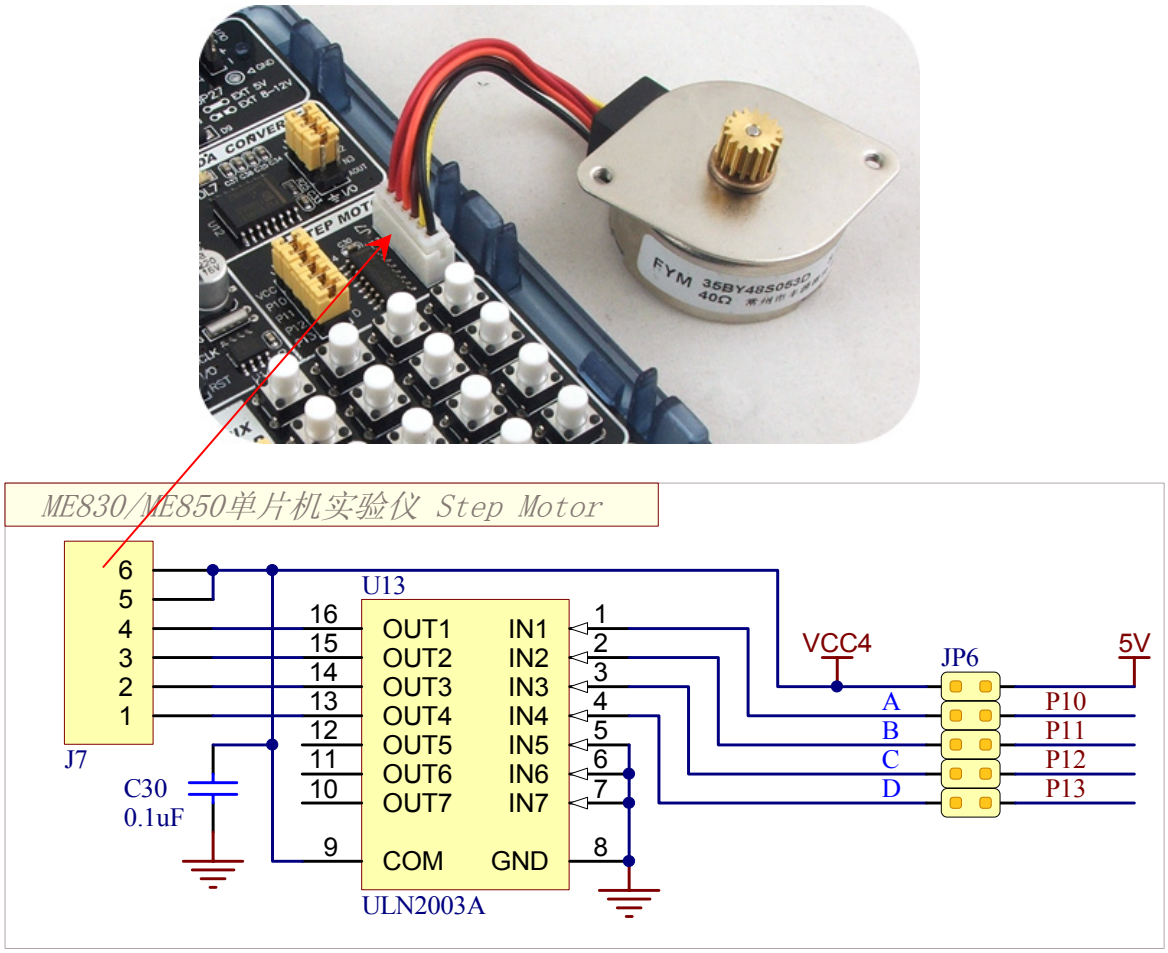


图 6.38　步进电机驱动电路

单片机的 P1.0-P1.3 输出的脉冲信号经 JP6 送到 ULN2003 的 IN1-IN4 输入端，经 ULN2003 放大和倒相后的输出脉冲信号通过 J7 来驱动步进电机作相应的操作。ULN2003 的 COM 端和步进电机的 COM1、COM2 连接到 VCC。步进电机内部原理图如图 6.39 所示。

例如：当单片机的 P1.0 输出高电平时，ULN2003 的 IN1 输入端则为高电平，经过 ULN2003 放大和倒相后在 OUT1 输出端输出低电平，使步进电机的 A 相得电旋转一个步距角。

2.1 ME830 板载步进电机性能指标

2 相 6 线式步进电机

步距角 7.5 度

工作电压>12V（实验时也可以用 5V 供电，只是力矩变小）

额定静力矩＞240g/cm
动力矩＞80g/cm
外形：φ35×15mm

2.2 步进电机内部电路

步进电机电路结构则如图 6.39 所示，包含两组带有中间抽头的线圈，A-COM1-C 为一组，B-COM2-D 为另一组。整个电机共有六条线。

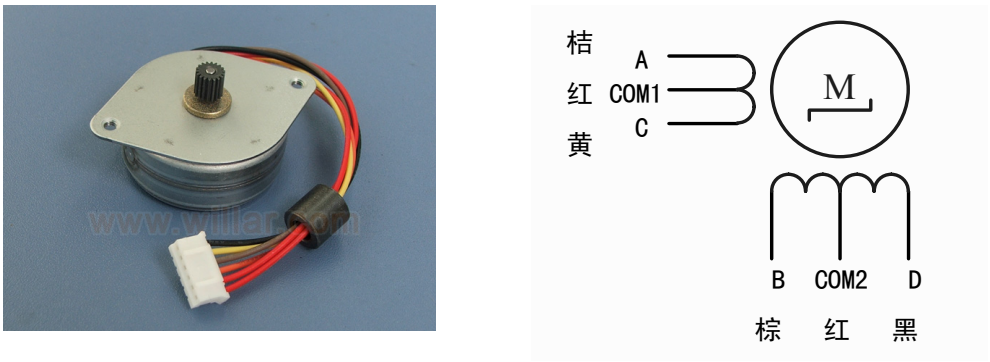


图 6.39　步进电机内部电路

3. 实验步骤

将 JP6 短接子全部用短接帽短接，使步进电机驱动芯片的输入脚与 P1 端口接通，VCC 向步进电机模块供电。

注意：不做步进电机实验时，请将 JP6 短接子上的短接帽全部取掉，否则将影响其它使用 P1.0-P1.3 端口模块的正常工作。

4. 程序流程图

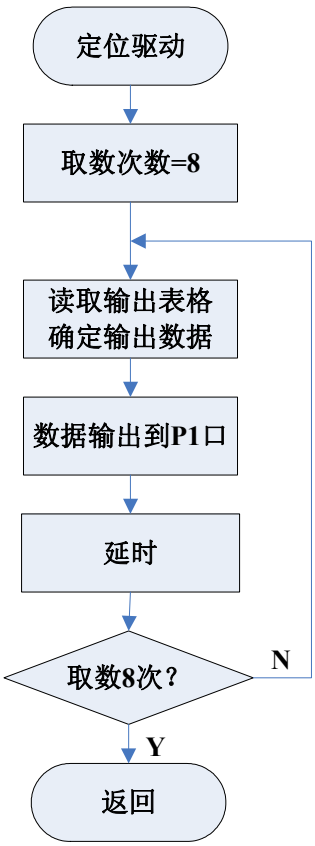


图 6.40　定位子程序流程图

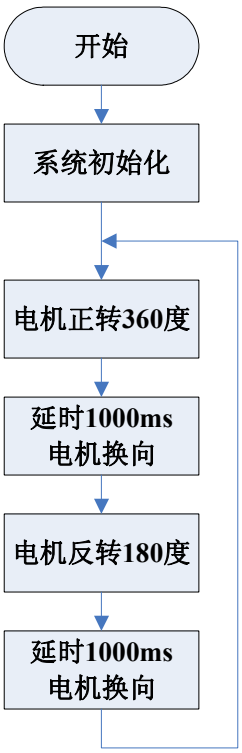


图 6.41　主程序流程图

5. 汇编源程序

（光盘： Example_A51\EX15_Motor）

```
;*****************************************************************
;*                                                               *
;* ME830 单片机开发系统演示程序 - 步进电机正反转                 *
;*                                                               *
;* 版本： V1.0 (2008/08/20)                                      *
;* 作者： gguoqing (Email: gguoqing@willar.com)                  *
;* 网站： www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界)*
;* 邮箱： willar@tom.com                                         *
;*                                                               *
;*【版权】Copyright(C) 深圳硕飞科技有限公司  All Rights Reserved  *
;*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！     *
;*                                                               *
;*****************************************************************
;*                                                               *
;* 步进电机步进角为 7.5 度。                                     *
;*                                                               *
;* 单双八拍工作方式：                                            *
;* A-AB-B-BC-C-CD-D-DA (即一个脉冲,转 3.75 度)                    *
;*                                                               *
;*****************************************************************

            ORG 0000H
            AJMP MAIN
            ORG 0050H

;*********************************************************
MAIN:
            MOV  P0,#0FFH        ;端口初始化
            MOV  P2,#0FFH
            MOV  P1,#0F0H
LOOP:
            MOV  R3,#12          ;正转 360 度，30*12=360 度
FFW:
            ACALL  MOTOR_F
            DJNZ R3,FFW

            MOV P1,#0F0H         ;使步进电机掉电
            ACALL DELAY1         ;延时 2s

            MOV  R3,#6           ;反转 180 度，30*6=180 度
REV:
            ACALL  MOTOR_R
            DJNZ R3,REV

            MOV P1,#0F0H        ;使步进电机掉电
            ACALL DELAY1        ;延时 2s
            AJMP  LOOP

;*********************************************************
; 步进电机正转 30 度子程序
; 一个脉冲转 3.75 度,8 个脉冲转 30 度。
;*********************************************************
MOTOR_F:
            MOV  R0,#00H
FFW1:
            MOV A,R0
            MOV DPTR,#TABLE_F
            MOVC A,@A+DPTR
            MOV P1,A
            ACALL DELAY
            INC R0
            CJNE  R0,#08H,FFW1
            RET

;*********************************************************
; 步进电机反转 30 度子程序
; 一个脉冲转 3.75 度,8 个脉冲转 30 度。
;*********************************************************
MOTOR_R:
```

```
            MOV R0,#00H
REV1:
            MOV A,R0
            MOV DPTR,#TABLE_R
            MOVC A,@A+DPTR
            MOV P1,A
            ACALL DELAY
            INC R0
            CJNE  R0,#08H,REV1
            RET

;*************************************************************
; 延时子程序 (步进电机的转速)
;*************************************************************
DELAY:
            MOV R7,#14
DEL1:
            MOV R6,#230
DEL2:
            DJNZ R6,DEL2
            DJNZ R7,DEL1
            RET

;*************************************************************

; 2s 延时子程序

;*************************************************************
DELAY1:
            MOV   R5,#20
DEL3:
            MOV   R7,#200
DEL4:
            MOV   R6,#230
            DJNZ  R6,$
            DJNZ  R7,DEL4
            DJNZ  R5,DEL3
            RET

;*************************************************************

;单双八拍工作方式编码

;*************************************************************
TABLE_F:
      DB  0F1H,0F3H,0F2H,0F6H,0F4H,0FCH,0F8H,0F9H    ;正转表
      DB  00      ;正转结束
TABLE_R:
      DB  0F9H,0F8H,0FCH,0F4H,0F6H,0F2H,0F3H,0F1H    ;反转表
      DB  00      ;反转结束

;*************************************************************

        END         ;结束

;*************************************************************
```

6. C 语言源程序

（光盘： Example_C51\EX15_Motor）

```
/*****************************************************************
 *                                                               *
 * ME830 单片机开发实验仪演示程序 一步进电机正反转实验            *
 *                                                               *
 * 版本： V1.0 (2008/08/20)                                      *
 * 作者： gguoqing (gguoqing@willar.com)                         *
 * 网站： www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界) *
 * 邮箱： willar@tom.com                                         *
 *                                                               *
 *【版权】Copyright(C)伟纳电子 www.willar.com  All Rights Reserved *
 *【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！     *
 *                                                               *
 *****************************************************************
 *                                                               *
 * 步进电机步进角为 7.5 度。                                     *
 *                                                               *
 * 单双八拍工作方式：                                            *
 * A-AB-B-BC-C-CD-D-DA (即一个脉冲,转 3.75 度)                    *
 *                                                               *
 *****************************************************************/
#include <reg52.h>        //51 芯片管脚定义头文件
#include <intrins.h>     //内部包含延时函数 _nop_();

unsigned char code FFW[8] =
{
  0xf1, 0xf3, 0xf2, 0xf6, 0xf4, 0xfc, 0xf8, 0xf9
};
unsigned char code REV[8] =
{
  0xf9, 0xf8, 0xfc, 0xf4, 0xf6, 0xf2, 0xf3, 0xf1
};

char code reserve[3]_at_ 0x3b; //保留 0x3b 开始的 3 个字节

/*************************************************************

延时 t 毫秒
11.0592MHz 时钟，延时约 1ms

 *************************************************************/
void delayms(unsigned int t)
{
  unsigned char k;
  while (t--)
  {
    for (k = 0; k < 114; k++)
      ;
  }
}

/*************************************************************

步进电机正转(转 n*30 度)

 *************************************************************/
void motor_ffw(unsigned int n)
{
  unsigned char i;
  unsigned int j;
  for (j = 0; j < n; j++)
  //转 n*30 度
  {
    for (i = 0; i < 8; i++)
    //一个周期转 30 度
    {
      P1 = FFW[i]; //取数据
      delayms(8); //调节转速
    }
  }
  P1 = 0xf0; //使步进电机掉电
}
```

```
/*************************************************************

步进电机反转(转 n*30 度)

 *************************************************************/
void motor_rev(unsigned int n)
{
  unsigned char i;
  unsigned int j;
  for (j = 0; j < n; j++)
  //转 n*30 度
  {
    for (i = 0; i < 8; i++)
    //一个周期转 30 度
    {
      P1 = REV[i]; //取数据
      delayms(8); //调节转速
    }
  }
  P1 = 0xf0; //使步进电机掉电
}

/*************************************************************

主程序

 *************************************************************/
void main(void)
{
  P1 = 0xf0; //端口初始化
  P0 = 0xff;
  P2 = 0xff;
  while (1)
  {
    motor_ffw(12); //电机正转 360 度
    delayms(1000); //换向延时
    motor_rev(6); //电机反转 180 度
    delayms(1000); //换向延时
  }
}
 /*************************************************************/
```

实验十六　NE555 计数实验

说明：ME830 无此模块（ME850 有此模块）!!!

1. 实验任务

利用 C51 单片机的 T0、T1 定时计数器对 NE555 产生的脉冲信号进行频率计数，频率测量结果通过 LCD1602 显示出来。

信号的频率定义为在 1 秒内信号的周期数，设置 C51 单片机的定时/计数器 0 工作在定时方式，用于实现 1S 定时；设置定时/计数器 1 工作在计数方式，实现对外部时钟脉冲个数的测量。1S 定时结束时，定时/计数器 1 中的计数值即为信号的频率。

2. 实验线路

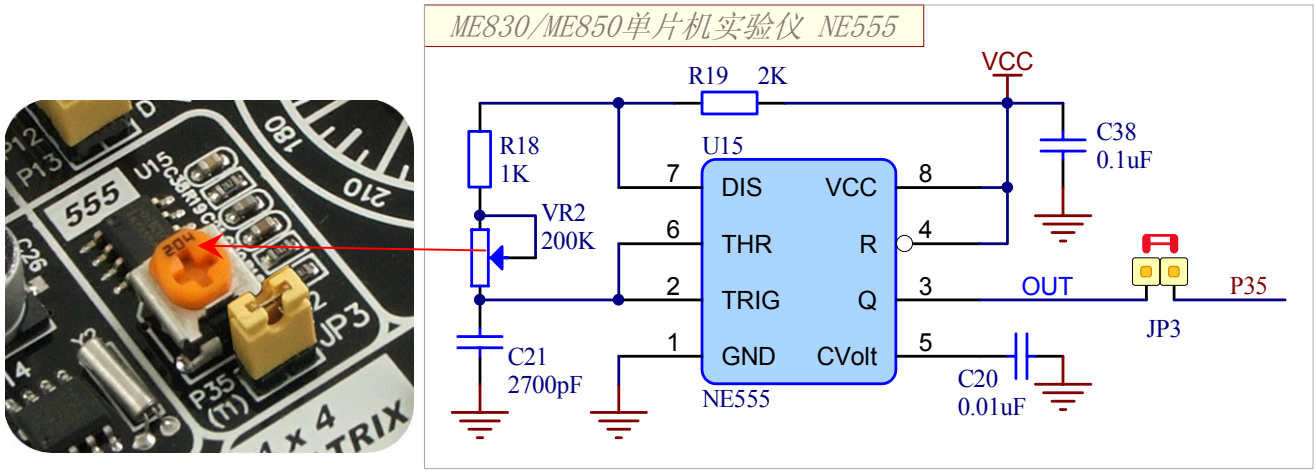


图 6.42 NE555 多谐振荡器电路

3. 实验步骤

将 1602LCD 模块插在 J5 接口上；

将 JP3 短接子短接，使 NE555 的输出脉冲信号与 P3.5（T1）连接；

将 JP17（24C04）短接子上的短接帽取掉，否则无法计频。

4. 程序流程图

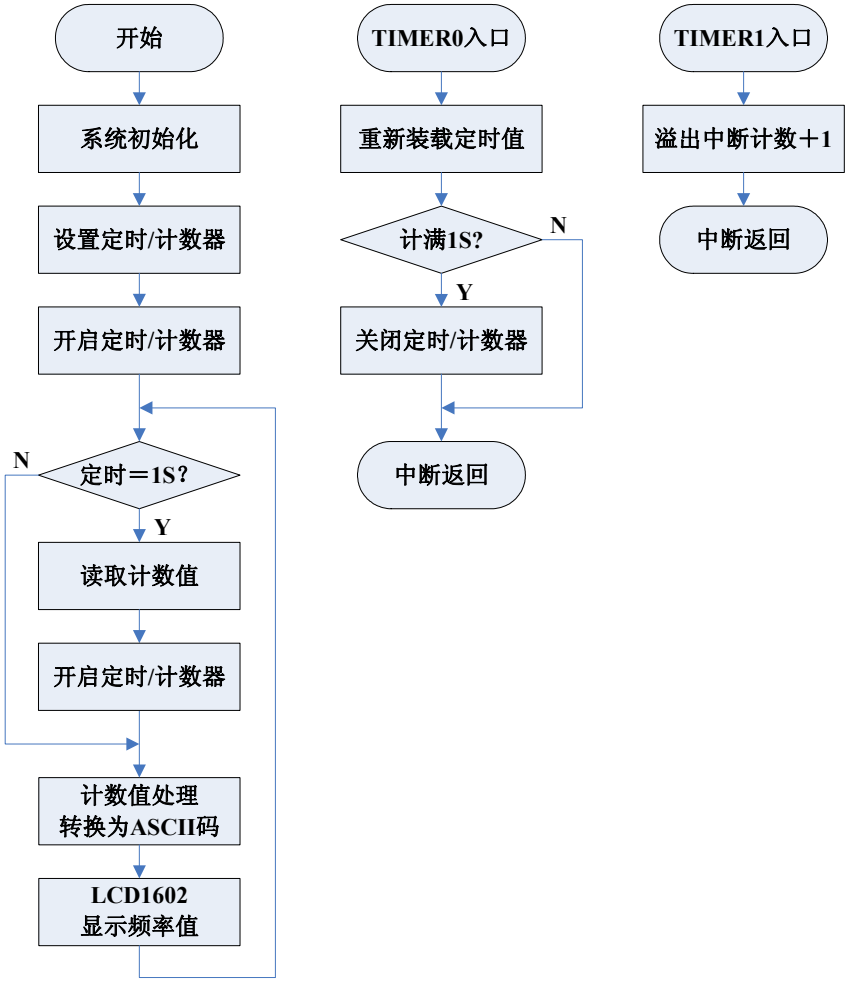


图 6.43　频率计程序流程图

5. 汇编源程序

（光盘： Example_A51\EX16_NE555）

```
;*********************************************************************
;*                                                                   *
;* ME830 单片机开发系统演示程序 -  频率计                            *
;*                                                                   *
;* LCD1602 显示                                                      *
;*                                                                   *
;* 晶振：12MHz                                                       *
;*                                                                   *
;* 版本：  V1.0 (2008/08/22)                                         *
;* 作者：  gguoqing (Email: gguoqing@willar.com)                     *
;* 网站：  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界)*
;* 邮箱：  willar@tom.com                                            *
;*                                                                   *
;*【版权】Copyright(C) 深圳硕飞科技有限公司  All Rights Reserved    *
;*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！       *
;*                                                                   *
;*********************************************************************

        LCD_RS  BIT  P2.0       ;LCD 控制管脚定义
        LCD_RW  BIT  P2.1
        LCD_EN  BIT  P2.2
        BUSY    BIT  P0.7

        DATAPORT EQU  P0        ;定义 LCD 的数据端口

        LCD_X   EQU  3FH        ;LCD 地址变量

        BUSY_CHECK  BIT  20H.0
```

```
TIMER_H   EQU   30H       ;定时器高位字节单元
TIMER_L   EQU   31H       ;定时器低位字节单元
TIMCOUNT  EQU   32H       ;50ms 中断计数

INT_G     EQU   35H       ;中断计数缓冲单元高地址
  INT_H     EQU   34H       ;中断计数缓冲单元中地址
  INT_L     EQU   33H       ;中断计数缓冲单元低地址

T_S       EQU   36H       ;数据显示低位
T_M       EQU   37H       ;数据显示中位
T_H       EQU   38H       ;数据显示高位
T_G       EQU   39H       ;数据显示最高位

TEMP_H    EQU   3AH
TEMP_L    EQU   3BH

;*********************************************************

      ORG 0000H
      AJMP MAIN             ;主程序开始
      ORG 000BH
      AJMP TIMER_INT        ;定时器 T0 中断服务程序
      ORG  001BH
      AJMP  TIMER1          ;定时器 T1 中断服务程序
      ORG  0050H

;*********************************************************
MAIN:
      MOV  SP,#60H          ;设置 SP 指针
      ACALL PRO_SET         ;程序初始化
      ACALL LCD_INIT        ;LCD 初始化
MAIN1:
      MOV  B,#00H
      MOV  DPTR,#INFO1      ;指针指到信息 1 首地址
      ACALL  W_STRING1
      MOV  B,#00H
      MOV  DPTR,#INFO2      ;指针指到信息 2 首地址
      ACALL  W_STRING2

      ACALL  TIMER_SET      ;定时器初始设定
MAIN2:
      ACALL  SBIN_SBCD
      ACALL  PLAY
      AJMP  MAIN2

;*********************************************************

INFO1:  DB  "    CYMOMETER   ",0   ;LCD 第一行显示信息
INFO2:  DB  "FREQ:        HZ ",0   ;LCD 第二行显示信息

;*********************************************************

;初始化程序

;*********************************************************
PRO_SET:
      MOV P0,#0FFH          ;端口初始化
      MOV P1,#0FFH
      MOV P2,#0FFH

      MOV A,#00H            ;寄存器初始化
      MOV B,#00H
      MOV 2AH,A

      MOV  INT_H,A
      MOV  INT_L,A
      MOV  INT_G,A
      MOV  T_S,A
      MOV  T_H,A
      MOV  T_M,A
      MOV  T_G,A
      MOV TIMCOUNT,A

      MOV TIMER_H,#04CH     ;定时 50 MS
```

```
        MOV TIMER_L,#10H
        SETB  P3.5              ;P3.5 端口置为输入状态
        RET                     ;T1(TIMER1 的外部输入脚)

;*********************************************************

; 定时器设置

; 设置定时器 0 工作在定时方式 1,定时器 1 工作在计数方式 1

;*********************************************************
TIMER_SET:
        MOV TMOD,#51H
        MOV TH0,TIMER_H         ;设置定时初值高位
        MOV TL0,TIMER_L         ;设置定时初值低位
        MOV  TH1,#00H           ;清 T1 计数器
        MOV  TL1,#00H
        MOV  IE,#8AH            ;开中断总允许, 允许 T0 溢出中断
        SETB  PT1               ;TIMER1 中断优先
        SETB  TR1
        SETB  TR0               ;定时器开始工作
        RET

;*********************************************************

;检查 LCD 忙状态

;busy 为 1 时, 忙, 等待。busy 为 0 时,闲, 可写指令与数据

;*********************************************************
LCD_BUSY:
        MOV  DATAPORT,#0FFH
BUSY_1:
        CLR  LCD_RS
        SETB LCD_RW
        CLR  LCD_EN
        NOP
        SETB LCD_EN
        JB  BUSY,BUSY_1
        CLR  LCD_EN
        RET

;*********************************************************

;LCD 写命令子程序

;LCD_RS=L,LCD_RW=L,D0-D7=指令码, E=高脉冲

;若 BUSY_CHECK =1,进行忙检测

;*********************************************************
WCOM:
        JNB  BUSY_CHECK,WCOM_1
        ACALL  LCD_BUSY
WCOM_1:
        MOV   DATAPORT,A        ;写入指令
        CLR   LCD_RS
        CLR   LCD_RW
        NOP
        SETB  LCD_EN
        NOP
        CLR   LCD_EN
        RET

;*********************************************************

;LCD 写数据子程序

;LCD_RS=H,LCD_RW=L,D0-D7=数据码, E=高脉冲

;*********************************************************
WDATA:
        ACALL  LCD_BUSY
        MOV  DATAPORT,A         ;写入数据
```

```
        SETB   LCD_RS
        CLR    LCD_RW
        NOP
        SETB   LCD_EN
        NOP
        CLR    LCD_EN
        RET

;*********************************************************

; 在 LCD 第一行的指定显示位置

;*********************************************************
SET_X1:
        MOV   A,LCD_X
        ORL   A,#80H
        ACALL   WCOM
        RET

;*********************************************************

; 在 LCD 第二行的指定显示位置

;*********************************************************
SET_X2:
        MOV   A,LCD_X
        ORL   A,#0C0H
        ACALL   WCOM
        RET

;*********************************************************

; 写字符串子程序 1

;*********************************************************
W_STRING1:
        MOV   A,#80H          ;设置 LCD 的第一行地址
        ORL   A,B
        ACALL   WCOM          ;写入命令
        ACALL   FILL_CHAR
        RET

;*********************************************************

; 写字符串子程序 2

;*********************************************************
W_STRING2:
        MOV   A,#0C0H          ;设置 LCD 的第二行地址
        ORL   A,B
        ACALL   WCOM           ;写入命令
        ACALL   FILL_CHAR
        RET

;*********************************************************

; 写入字符子程序

;*********************************************************
FILL_CHAR:
        CLR   A                     ;填入字符
        MOVC   A,@A+DPTR            ;由字符区取出字符
        CJNE   A,#0,F_CHAR          ;判断是否为结束码
        RET
F_CHAR:
        ACALL   WDATA               ;写入数据
        INC   DPTR                  ;指针加 1
        AJMP   FILL_CHAR            ;继续填入字符
        RET

;*********************************************************

; LCD 初始化子程序
```

```
;*********************************************************
LCD_INIT:
        CLR   BUSY_CHECK     ;不进行忙检测
        MOV   A,#38H         ;强制执行 3 次
        ACALL   WCOM
        ACALL   DELAY1
        MOV   A,#38H         ;双列显示，字形 5*7 点阵
        ACALL   WCOM
        ACALL   DELAY1
        MOV   A,#38H         ;双列显示，字形 5*7 点阵
        ACALL   WCOM
        ACALL   DELAY1

        SETB   BUSY_CHECK    ;进行忙检测
        MOV   A,#0CH         ;开显示，不显示光标。
        ACALL   WCOM
        ACALL   DELAY1
        MOV   A,#06H         ;
        ACALL   WCOM
        ACALL   DELAY1
        MOV   A,#01H         ;清除 LCD 显示屏
        ACALL   WCOM
        ACALL   DELAY1
        RET

;*********************************************************

; 延时 5MS 子程序

;*********************************************************
DELAY1:
        MOV   R6,#25
DEL3:
        MOV   R7,#93
DEL4:
        DJNZ   R7,DEL4
        DJNZ   R6,DEL3
        RET

;*********************************************************

;定时器 0 中断服务程序

;*********************************************************
TIMER_INT:
        CLR TR0              ;关闭定时器
        PUSH   ACC
        MOV TL0,TIMER_L      ;重新赋初值
        MOV TH0,TIMER_H                ;
        INC TIMCOUNT         ;定时 50msj 计数单位
        MOV A,TIMCOUNT       ;查看数量值
        CPL   P1.4           ;产生自测信号
        CJNE A,#20,T_END     ;如果没有到 1S 返回

        CLR   TR1            ;关闭计数器 T1
        MOV TIMCOUNT,#00H    ;到 1S 则清零
        MOV INT_L,TL1        ;取出计数值低位
        MOV INT_H,TH1        ;取出计数值高位
        MOV INT_G,2AH        ;取出溢出计数值位
        MOV   TH1,#00H
        MOV   TL1,#00H
        MOV   2AH,#00H
        SETB   TR1
T_END:
        POP   ACC
        SETB TR0             ;重新开始定时操作
        RETI

;*********************************************************

;T1 计数器中断服务子程序

;计 T1 计数器溢出次数
```

```
;*********************************************************
TIMER1:
        INC   2AH
        RETI

;*********************************************************

; 频率值显示子程序

;*********************************************************
PLAY:
        MOV   A,#0C6H           ;确定显示首地址
        ACALL   WCOM

        MOV   A,T_H
        ACALL   CONV
        MOV   A,TEMP_H
        ACALL   WDATA
        MOV   A,TEMP_L
        ACALL   WDATA

        MOV   A,T_M
        ACALL   CONV
        MOV   A,TEMP_H
        ACALL   WDATA
        MOV   A,TEMP_L
        ACALL   WDATA

        MOV   A,T_S
        ACALL   CONV
        MOV   A,TEMP_H
        ACALL   WDATA
        MOV   A,TEMP_L
        ACALL   WDATA
        RET

;*********************************************************

; 数据转换子程序

; 输入：  A-待转换数据(压缩 BCD 码)
; 输出：  TEMP_H - ASCII 码高位，TEMP_L - ASCII 码低位

;*********************************************************
CONV:
        MOV   B,#16             ;压缩 BCD 码分离
        DIV   AB
        ADD   A,#30H            ;转换为 ASCII 码
        MOV   TEMP_H,A          ;保存数据
        MOV   A,B
        ADD   A,#30H            ;转换为 ASCII 码
        MOV   TEMP_L,A          ;保存数据
        RET

;*********************************************************

;三字节二进制整数转换成四字节 BCD 码子程序

;二进制数从低位到高位分别存放在 INT_L、INT_H、INT_G 单元中
;BCD 码从低位到高位分别存放在 T_S、T_M、T_H、T_G 单元中

;*********************************************************
SBIN_SBCD:
        CLR A                   ;清累加器
        MOV T_G,A
        MOV T_H,A               ;清除出口单元，准备转换
        MOV T_M,A
        MOV T_S,A

        MOV   R7,INT_L          ;设置二进制数起始地址
        MOV   R6,INT_H
        MOV   R5,INT_G
        MOV   R2,#24   ;循环次数 3(字节)*8(bit)
        CLR   C
```

```
SBIN_SBCD1:
        MOV   A,R7          ;低位
        RLC   A
        MOV   R7,A

        MOV   A,R6          ;次高位
        RLC   A
        MOV   R6,A

        MOV   A,R5          ;高位
        RLC   A
        MOV   R5,A

        MOV A,T_S           ;得到低位数据
        ADDC A,T_S          ;累加
        DA A                ;十进制调整
        MOV T_S,A           ;保存数据结果

        MOV A,T_M           ;得到第二位数据
        ADDC A,T_M          ;累加
        DA A                ;十进制调整
        MOV T_M,A           ;保存数据结果

        MOV A,T_H           ;得到第三位
        ADDC A,T_H          ;累加
        DA  A               ;十进制调整
        MOV T_H,A           ;保存数据结果

        MOV A,T_G           ;得到第四位
        ADDC A,T_G          ;累加
        DA  A               ;十进制调整
        MOV T_G,A           ;保存数据结果

        DJNZ R2,SBIN_SBCD1       ;

        RET

;*********************************************************

        END                 ;告诉编译器本程序到此结束。

;*********************************************************
```

6. C 语言源程序

（光盘： Example_C51\EX16_NE555）

```
/*********************************************************************
 *                                                                   *
 * ME830 单片机开发实验仪演示程序 -    频率计                        *
 *                                                                   *
 * LCD1602 显示                                                      *
 *                                                                   *
 * 时间：  2008/09/17                                                *
 * 作者：  gguoqing (gguoqing@willar.com)                            *
 * 网站：  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界) *
 * 邮箱：  willar@tom.com                                            *
 *                                                                   *
 *【版权】Copyright(C)伟纳电子 www.willar.com  All Rights Reserved  *
 *【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！       *
 *                                                                   *
 *********************************************************************/

#include <reg52.h>
#include <intrins.h>

sbit BEEP = P3 ^ 7; //蜂鸣器

unsigned char code cdis1[] =
{
  "    CYMOMETER   "
};
unsigned char code cdis2[] =
```

```
{
  "FREQ:        Hz "
};

sbit LCD_RS = P2 ^ 0;
sbit LCD_RW = P2 ^ 1;
sbit LCD_EN = P2 ^ 2;

bit sec = 0;
unsigned char msec = 0, Hdata = 0, Ldata = 0, Count = 0;
unsigned long temp = 0;
unsigned char data display[] =
{
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};

char code reserve[3]_at_ 0x3b; //保留 0x3b 开始的 3 个字节

/*********************************************************

us 延时函数    (4.34us)

 *********************************************************/
void delayNOP()
{
  _nop_();
  _nop_();
  _nop_();
  _nop_();
}

/*********************************************************

ms 延时函数

 *********************************************************/
void delayms(unsigned int ms)
{
  unsigned char n;
  while (ms--)
  {
    for (n = 0; n < 114; n++)
      ;
  }
}

/*********************************************************
 *                                                       *
 * 检查 LCD 忙状态                                       *
 * lcd_busy 为 1 时，忙，等待。                          *
 * lcd-busy 为 0 时,闲，可写指令与数据。                 *
 *                                                       *
 *********************************************************/
bit lcd_busy()
{
  bit result;
  LCD_RS = 0;
  LCD_RW = 1;
  LCD_EN = 1;
  delayNOP();
  result = (bit)(P0 &0x80);
  LCD_EN = 0;
  return (result);
}

/*********************************************************
 *                                                       *
 *写指令数据到 LCD                                       *
 *RS=L，RW=L，E=高脉冲，D0-D7=指令码。                   *
 *                                                       *
 *********************************************************/
void lcd_wcmd(unsigned char cmd)
{
  while (lcd_busy())
```

```
    ;
  LCD_RS = 0;
  LCD_RW = 0;
  LCD_EN = 1;
  P0 = cmd;
  delayNOP();
  LCD_EN = 0;
}

/*************************************************************
 *                                                           *
 *写显示数据到 LCD                                            *
 *RS=H, RW=L, E=高脉冲, D0-D7=数据。                          *
 *                                                           *
 *************************************************************/
void lcd_wdat(unsigned char dat)
{
  while (lcd_busy())
    ;
  LCD_RS = 1;
  LCD_RW = 0;
  LCD_EN = 1;
  P0 = dat;
  delayNOP();
  LCD_EN = 0;
}

/*************************************************************
 *                                                           *
 *  LCD 初始化设定                                           *
 *                                                           *
 *************************************************************/
void lcd_init()
{
  delayms(15);

  lcd_wcmd(0x38); //16*2 显示, 5*7 点阵, 8 位数据
  delayms(5);
  lcd_wcmd(0x38);
  delayms(5);
  lcd_wcmd(0x38);
  delayms(5);

  lcd_wcmd(0x0c); //开显示, 不显示光标
  delayms(5);
  lcd_wcmd(0x06); //
  delayms(5);
  lcd_wcmd(0x01); //清除 LCD 的显示内容
  delayms(5);
}

/*************************************************************
 *                                                           *
 *  设定显示位置                                             *
 *                                                           *
 *************************************************************/

void lcd_pos(unsigned char pos)
{
  lcd_wcmd(pos | 0x80); //数据指针=80+地址变量
}

/*************************************************************
 *                                                           *
 * 显示函数                                                  *
 *                                                           *
 *************************************************************/
void play()
{
  unsigned char n;

  for (n = 0; n <= 4; n++)
  //数据转换
  {
```

```
    display[n] = temp % 10+0x30;
    temp = temp / 10;
  }
  display[5] = temp + 0x30;

  for (n = 5; n > 0; n--)
  //高位为"0"不显示
  {
    if (display[n] == 0x30)
      display[n] = 0x20;
    else
      break;
  }

  lcd_pos(0x46); //显示实际频率值
  for (n = 5; n != 0xff; n--)
    lcd_wdat(display[n]);
}

/*********************************************************
 *                                                       *
 * 主函数                                                *
 *                                                       *
 *********************************************************/
void main()
{
  unsigned char m;
  unsigned long frq_num;

  P3 = 0xff;

  lcd_init();
  lcd_pos(0x00); //设置显示位置为第一行
  for (m = 0; m < 16; m++)
    lcd_wdat(cdis1[m]);
  //显示字符
  lcd_pos(0x40); //设置显示位置为第二行
  for (m = 0; m < 16; m++)
    lcd_wdat(cdis2[m]);
  //显示字符

  TMOD = 0x51; //定时器 0 工作在定时方式
  //定时器 1 工作在计数方式
  TH0 = 0x4c; //50ms 定时
  TL0 = 0x00;
  TH1 = 0x00; //计数初值
  TL1 = 0x00;
  ET0 = 1; //使能 TIMER0 中断
  ET1 = 1; //使能 TIMER1 中断
  EA = 1; //允许中断
  PT1 = 1; //定义 TIMER1 中断优先
  TR0 = 1;
  TR1 = 1;

  while (1)
  {
    if (sec)
    {
      Hdata = TH1; //取计数值
      Ldata = TL1;
      frq_num = ((Count *65535+Hdata * 256+Ldata) *108 / 100);
      TH1 = 0;
      TL1 = 0;
      sec = 0;
      Count = 0;
      TR1 = 1;
      TR0 = 1;
    }
    temp = frq_num;
    play();
  }
}

/*********************************************************
```

```
 *                                                          *
 * Time0 中断函数                                            *
 *                                                          *
 ************************************************************/
void Time0()interrupt 1
{
  TH0 = 0x4c; //50ms 定时
  TL0 = 0x00;
  msec++;
  if (msec == 20)
  //50*20=1S
  {
    TR0 = 0; //关闭 TIMER0
    TR1 = 0; //关闭 TIMER1
    msec = 0;
    sec = 1; //置秒标记位
  }
}

/************************************************************
 *                                                          *
 * Time1 中断函数                                            *
 *                                                          *
 ************************************************************/
void Time1()interrupt 3
{
  Count++;
}

/************************************************************/
```

实验十七　93C46 读写实验

1. 实验任务

先从地址 0x00 开始连续写入 8 个数据"0-7"，然后再从地址 0x00 开始读出所写入的 8 个数据，存到 8 个显示存储区中，然后再由 8 位数码管显示。写读操作成功后，8 位数码管从右至左依次显示 0-7。

2. 实验线路

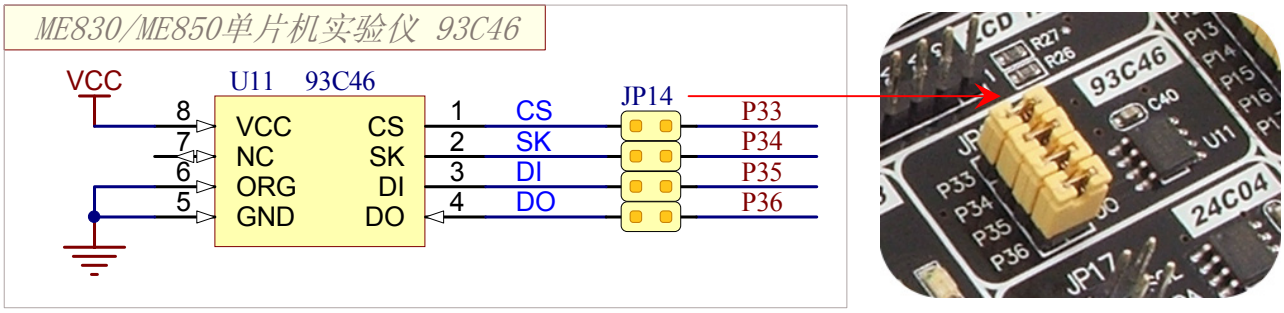


图 6.44　93C46 原理图

3. 实验步骤

将 JP21 的 8 个短接子全部用短接帽短接，使 DG0～DG7 与 P2 端口接通。

将 JP22 的 9 个短接子全部用短接帽短接，使 A～DP 与 P0 端口接通，VCC 向数码管模块供电。

将 JP14 的 4 个短接子全部用短接帽短接，使芯片管脚与 P3.3～P3.6 端口接通。

4. 程序流程图

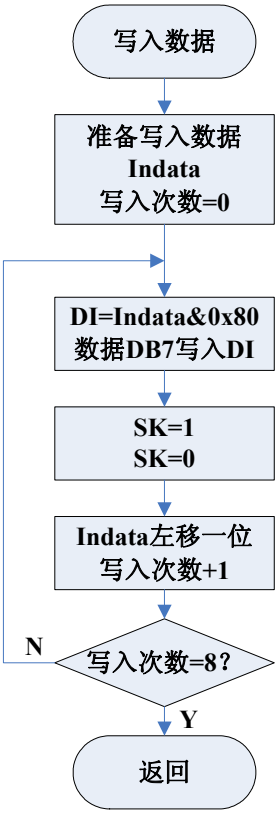


图 6.45　写 1 字节数据流程图

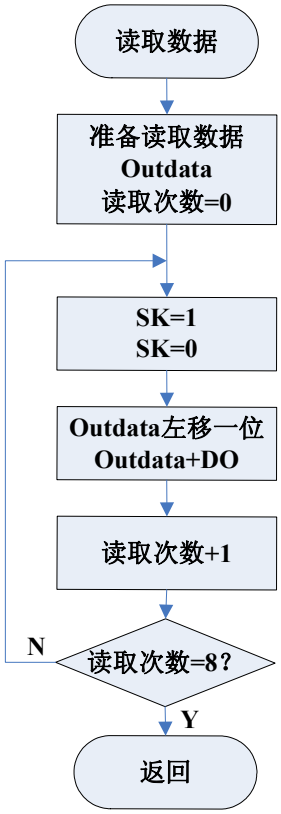


图 6.46　读 1 字节数据流程图

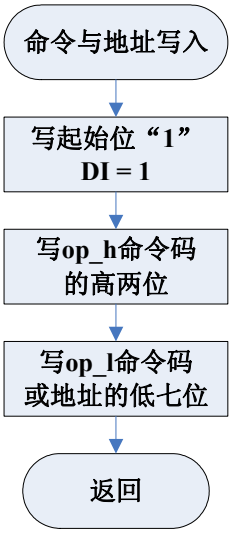


图 6.47　命令与地址写入流程图

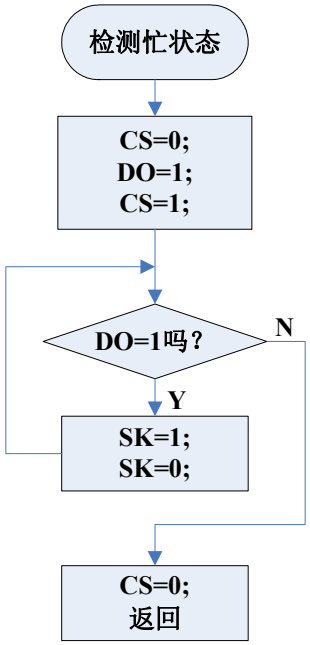


图 6.48　忙检测流程图

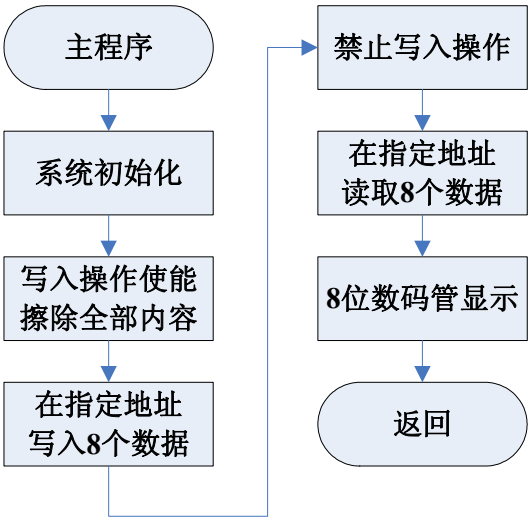


图 6.49　主程序流程图

5. 汇编源程序

（光盘： Example_A51\EX17_93C46）

```
;*********************************************************************
;*                                                                   *
;* ME830 单片机开发系统演示程序 - 读写 93C46                         *
;*                                                                   *
;* 8 位数码管显示                                                    *
;*                                                                   *
;* 版本:  V1.0 (2008/09/20)                                          *
;* 作者:  gguoqing (Email: gguoqing@willar.com)                      *
;* 网站:  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界)  *
;* 邮箱:  willar@tom.com                                             *
;*                                                                   *
;*【版权】Copyright(C) 深圳硕飞科技有限公司  All Rights Reserved     *
;*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息!          *
;*                                                                   *
;*********************************************************************
;*                                                                   *
;* 将演示数据先写入 93C46 芯片内，再将其数据读出送数码管显示。       *
;*                                                                   *
```

```
;* ORG=0  8 位数据存储器结构                                          *
;*                                                                    *
;* 注意：在擦除或写入数据之前，必须先写入 EWEN 指令。                 *
;*                                                                    *
;*********************************************************************/

        CS   BIT  P3.3
        SK   BIT  P3.4
        DI   BIT  P3.5
        DO   BIT  P3.6

        ADDR       EQU  30H
        INDATA     EQU  31H
        DIS_BUFF   EQU  40H

       OP_EWEN_H   EQU  00H   ; 00          write enable
       OP_EWEN_L   EQU  60H   ; 11X XXXX    write enable

       OP_EWDS_H   EQU  00H   ; 00          disable
       OP_EWDS_L   EQU  00H   ; 00X XXXX    disable

       OP_WRITE_H  EQU  40H   ; 01 A6-A0    write data
       OP_READ_H   EQU  80H   ; 10 A6-A0    read data
       OP_ERASE_H  EQU  0c0H  ; 11 A6-A0    erase a word

       OP_ERAL_H   EQU  00H   ; 00          erase all
       OP_ERAL_L   EQU  40H   ; 10X XXXX    erase all

       OP_WRAL_H   EQU  00H   ; 00          write all
       OP_WRAL_L   EQU  20H   ; 01X XXXX    write all

;*************************************************************

        ORG  0000H
        AJMP  MAIN
        ORG  0050H

;*************************************************************

; 主程序

;*************************************************************
MAIN:
        MOV   SP,#60H              ;设置堆栈

        CLR   CS                   ;芯片初始化
        CLR   SK
        SETB  DI
        SETB  DO

        ACALL EWEN                 ;使能写入操作
        ACALL ERASE              ;擦除全部内容

        CLR   A
        MOV   ADDR,A               ;开始写入地址为 00H
WRITE_LP:
        MOV   A,ADDR               ;写入的数据为地址值
        MOV   DPTR,#TABLE
        MOVC  A,@A+DPTR
        MOV  R7,A                  ;要写入的数据保存到 R7
        ACALL WRITE
        INC   ADDR                 ;修改地址
        MOV  R4,ADDR
        CJNE  R4,#08H,WRITE_LP     ;8 个数据是否写完?

        ACALL EWDS                 ;禁止写入操作

        MOV   R0,#DIS_BUFF         ;显存单元首地址
        CLR   A
        MOV   ADDR,A               ;开始读取地址为 00H
READ_LP:
        ACALL READ
        MOV   A,R7
        MOV   @R0,A                ;读出的数据存入相应的显存单元
```

```
          INC   R0                    ;修改显存单元地址
          INC    ADDR                 ;修改地址
          MOV   R4,ADDR
          CJNE   R4,#08H,READ_LP      ;8 个数据是否读完?
LEDOUT1:
          MOV   R0,#DIS_BUFF          ;显存单元首地址
          MOV   R4,#08H               ;8 位数码管
          MOV   A,#0FEH               ;位码初始值
LEDOUT2:
          MOV   P0,@R0                ;段码输出
          MOV   P2,A                  ;位码输出
          INC   R0
          RL    A
          ACALL   DELAY1MS
          DJNZ   R4,LEDOUT2           ;8 位数码管是否显示完毕?
          MOV   P2,#0FFH              ;关闭数码管显示
          SJMP   LEDOUT1

;*************************************************************

; addr 处写入数据子程序

;*************************************************************
WRITE:
          MOV   indata,R7          ;写入数据转移
          MOV   R5,ADDR
          MOV   R7,#OP_WRITE_H     ;40H
          ACALL INOP               ;写入操作码和地址
          MOV   R7,indata          ;写入数据
          ACALL SHIN
          ACALL BUSY               ;忙检测
          RET

;*************************************************************

; 读取 addr 处的数据子程序

;*************************************************************
READ :
          MOV   R5,ADDR
          MOV   R7,#OP_READ_H      ;80H
          ACALL INOP               ;写入操作码和地址
          ACALL SHOUT              ;读出数据
          CLR    CS
          RET

;*************************************************************

; 写使能子程序

;*************************************************************
EWEN:
          MOV   R5,#OP_EWEN_L      ;60H
          MOV   R7,#OP_EWEN_H      ;00H
          ACALL INOP
          CLR    CS
          RET

;*************************************************************

; 写禁止子程序

;*************************************************************
EWDS:
          MOV   R5,#OP_EWDS_L      ;00H
          MOV   R7,#OP_EWDS_H      ;00H
          ACALL INOP
          CLR    CS
          RET

;*************************************************************

; 擦除子程序
```

```
;*********************************************************
ERASE:
          MOV   R5,#0P_ERAL_L       ;40H
          MOV   R7,#0P_ERAL_H       ;00H
          ACALL INOP
          ACALL  BUSY
          RET

/*********************************************************

  忙检测子函数

  DO=0 表示芯片仍在编程中
  DO=1 表示芯片完成数据写入

*********************************************************/
BUSY:
         CLR   CS
         SETB  DO                ;置接收位为 1
         NOP
         NOP
         SETB  CS
BUSY1:
         JNB   DO,BUSY1          ;DO=0 在编程中
         CLR   CS
         RET

;*********************************************************

 ;命令与地址写入子程序

 ;R7 为指令码的高两位
 ;R5 为指令码的低 7 位或 7 位地址

;*********************************************************
INOP:
          CLR    SK
          SETB   DI              ;写起始位"1"
          SETB   CS         ;选中该芯片
          NOP
          NOP
          SETB   SK              ;上升沿将起始位送入
          NOP
          NOP
          CLR    SK         ;开始位结束

          MOV    A,R7            ;写操作码高位
          RLC    A
          MOV    DI,C            ;移入指令码高位
          SETB   SK
          RLC    A
          CLR    SK
          MOV    DI,C            ;移入指令码次高位
          SETB   SK
          NOP
          NOP
          CLR    SK

          MOV    A,R5            ;写操作码低位或地址数据
          RLC    A
          MOV    R5,A            ;抛弃最高位
          CLR    A
          MOV    R7,A            ;计数单元清零
INOP_LP:
          MOV    A,R5            ;写 7 位操作码低位或地址数据
          RLC    A
          MOV    R5,A
          MOV    DI,C            ;写一位
          SETB   SK              ;上升沿将数据送入
          NOP
          NOP
          CLR    SK
          INC    R7
          CJNE   R7,#07H,INOP_LP   ;7 位数据是否写完?
```

```
          SETB   DI
          RET

;*********************************************************

; 写入数据子程序

; R7_要写入的数据，R6_计数单元

;*********************************************************
SHIN :
          CLR    A                ;清相关寄存器
          MOV    R6,A
          MOV    A,R7             ;要写入的数据在 R7 中
SHIN_LP:
          RLC    A
          MOV    DI,C             ;写一位
          SETB   SK
          NOP
          NOP
          CLR    SK
          INC    R6
          CJNE   R6,#08H,SHIN_LP
          SETB   DI
          RET

;*********************************************************

; 读出数据子函数

; R7_读出的数据保存单元，R6_计数单元

;*********************************************************
SHOUT:
          CLR    A                ;清相关寄存器
          MOV    R6,A
SHOUT_LP:
          SETB   SK
          NOP
          NOP
          CLR    SK
          MOV    C,DO             ;读一位
          RLC    A
          INC    R6
          CJNE   R6,#08H,SHOUT_LP
          MOV    R7,A             ;读出的数据保存在 R7 中
          RET

;*********************************************************

;延时 1ms 子程序

;*********************************************************
DELAY1MS:
          MOV R7,#2
 DL3:
          MOV R6,#230
 DL4:
          DJNZ R6,DL4
          DJNZ R7,DL3
          RET

;*********************************************************

TABLE:     DB   0C0H,0F9H,0A4H,0B0H,99H,92H,82H,0F8H,80H

;TABLE1:   DB   7EH,0BDH,0DBH,0E7H,0DBH,0BDH,7EH,0FFH

;*********************************************************

          END          ;结束

;*********************************************************
```

6. C 语言源程序

（光盘： Example_C51\EX17_93C46）

```
/*****************************************************************
 *                                                               *
 * ME830 单片机开发实验仪演示程序 - 读写 93C46                    *
 *                                                               *
 * 8 位数码管显示                                                *
 *                                                               *
 * 版本： V1.0 (2008/09/20)                                      *
 * 作者： gguoqing (gguoqing@willar.com)                         *
 * 网站： www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界) *
 * 邮箱： willar@tom.com                                         *
 *                                                               *
 *【版权】Copyright(C)伟纳电子 www.willar.com  All Rights Reserved *
 *【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！     *
 *                                                               *
 *****************************************************************
 *                                                               *
 * 将演示数据先写入 93C46 芯片内，再将其数据读出送数码管显示。   *
 *                                                               *
 * ORG=0  8 位数据存储器结构                                     *
 *                                                               *
 * 注意：在擦除或写入数据之前，必须先写入 EWEN 指令。            *
 *                                                               *
 *****************************************************************/

#include <reg52.h>
#include <intrins.h>

//define OP code
#define OP_EWEN_H       0x00 // 00             write enable
#define OP_EWEN_L       0x60 // 11X XXXX       write enable

#define OP_EWDS_H       0x00 // 00             disable
#define OP_EWDS_L       0x00 // 00X XXXX       disable

#define OP_WRITE_H      0x40 // 01 A6-A0       write data
#define OP_READ_H       0x80 // 10 A6-A0       read data
#define OP_ERASE_H      0xc0 // 11 A6-A0       erase a word

#define OP_ERAL_H       0x00 // 00             erase all
#define OP_ERAL_L       0x40 // 10X XXXX       erase all

#define OP_WRAL_H       0x00 // 00             write all
#define OP_WRAL_L       0x20 // 01X XXXX       write all

//define pin
sbit CS = P3 ^ 3;
sbit SK = P3 ^ 4;
sbit DI = P3 ^ 5;
sbit DO = P3 ^ 6;

unsigned char code dis_code[] =
{
  0xc0, 0xf9, 0xa4, 0xb0, 0x99, 0x92, 0x82, 0xf8, 0x80
};
unsigned char data display[8];

char code reserve[3]_at_ 0x3b; //保留 0x3b 开始的 3 个字节

/*********************************************************

延时子程序

 *********************************************************/
void delayms(unsigned int ms)
{
  unsigned char i;
  while (ms--)
  {
    for (i = 0; i < 114; i++)
      ;
  }
```

```
}

/*********************************************************

命令与地址写入子函数

写入 op_h 的高两位和 op_l 的低 7 位
op_h 为指令码的高两位
op_l 为指令码的低 7 位或 7 位地址

 *********************************************************/
void inop(unsigned char op_h, unsigned char op_l)
{
  unsigned char i;

  /*********************************************

  写起始位

   *********************************************/
  SK = 0;
  DI = 1; //写起始位"1"
  CS = 1; //选中该芯片
  _nop_();
  _nop_();
  SK = 1; //上升沿将起始位送入
  _nop_();
  _nop_();
  SK = 0; // 开始位结束

  /*********************************************

  写 op_h 命令码

  写入指令码高位和次高位

   *********************************************/
  for (i = 0; i < 2; i++)
  {
    DI = (bit)(op_h &0x80);
    SK = 1;
    op_h <<= 1;
    SK = 0;
  }

  /*********************************************

  写 op_l 命令码或地址

  从次高位开始写入

   *********************************************/
  for (i = 0; i < 7; i++)
  {
    DI = (bit)(op_l &0x40); //从次高位开始写入
    SK = 1;
    op_l <<= 1;
    SK = 0;
  }
  DI = 1;
}

/*********************************************************

写入数据子函数

 *********************************************************/
void shin(unsigned char indata)
{
  unsigned char i;
  for (i = 0; i < 8; i++)
  {
    DI = (bit)(indata &0x80); //从高位开始写入
    SK = 1;
```

```
    indata <<= 1;
    SK = 0;
  }
  DI = 1;
}

/*********************************************************

读出数据子函数

 *********************************************************/
unsigned char shout(void)
{
  unsigned char i, out_data;
  for (i = 0; i < 8; i++)
  {
    SK = 1;
    out_data <<= 1; //从高位开始读出
    SK = 0;
    out_data |= DO;
  }
  return (out_data);
}

/*********************************************************

忙检测子函数

DO=0 表示芯片仍在编程中
DO=1 表示芯片完成数据写入

 *********************************************************/
void busy()
{
  CS = 0;
  DO = 1; //置接收端为 1
  _nop_();
  CS = 1;
  while (DO == 0)
  //DO=0 在编程中
  {
    SK = 1;
    SK = 0;
  }
  CS = 0;
}

/*********************************************************

写使能子函数

写入命令码和地址后,CS 产生一个低电平脉冲启动自定时编程.

 *********************************************************/
void ewen()
{
  inop(OP_EWEN_H, OP_EWEN_L);
  CS = 0;
}

/*********************************************************

写禁止子函数

写入命令码和地址后,CS 产生一个低电平脉冲启动自定时编程.

 *********************************************************/
void ewds()
{
  inop(OP_EWDS_H, OP_EWDS_L);
  CS = 0;
}

/*********************************************************
```

整片擦除子函数

```
 ********************************************************/
void eral()
{
  inop(OP_ERAL_H, OP_ERAL_L);
  CS = 0;
  busy(); //忙检测
}

/********************************************************

擦除指定地址子函数

 ********************************************************/
/*
void erase(uchar  addr)
{
inop(OP_ERASE_H, addr);
CS=0;
busy();              //忙检测
}
 */
/********************************************************

整片写入子函数

 ********************************************************/
/*
void  wral(uchar  indata)
{
eral();                      //擦除全部内容
inop(OP_WRAL_H, OP_WRAL_L);
shin(indata);          //写入数据
CS=0;
busy();                   //忙检测
}
 */
/********************************************************

写入数据 indata 到 addr

 ********************************************************/
void write(unsigned char addr, unsigned char indata)
{
  inop(OP_WRITE_H, addr); //写入指令和地址
  shin(indata); //写入数据
  busy(); //忙检测
}

/********************************************************

读取 addr 处的数据

 ********************************************************/
unsigned char read(unsigned char addr)
{
  unsigned char out_data;
  inop(OP_READ_H, addr); //写入指令和地址
  out_data = shout(); //数据读出
  CS = 0;
  return (out_data);
}

/********************************************************

主函数

 ********************************************************/
void main(void)
{
  unsigned char i, shift;
  CS = 0; //初始化端口
```

```
    SK = 0;
    DI = 1;
    DO = 1;

    ewen(); //使能写入操作
    eral(); //擦除全部内容

    for (i = 0; i < 8; i++)
    //写入显示代码到 AT93C46
      write(i, dis_code[i]);

    ewds(); //禁止写入操作

    for (i = 0; i < 8; i++)
      display[i] = read(i);
    //读取 AT93C46 内容

    while (1)
    {
      shift = 0xfe;
      P2 = 0xff;
      for (i = 0; i < 8; i++)
      {
        P0 = display[i];
        P2 = shift;
        shift = _crol_(shift, 1);
        delayms(1); //固定方式显示
        // delayms(400);          //跑马灯方式显示
      }
    }
  }

  /*********************************************************/
```

实验十八　24C04 读写实验

1. 实验任务

先从地址 0x00 开始连续写入 8 个数据"0-7"，然后再从地址 0x00 开始读出所写入的 8 个数据，存到 8 个显示存储区中，然后再由 8 位数码管显示。

写读操作成功后，8 位数码管从右至左依次显示 0-7。

2. 实验线路

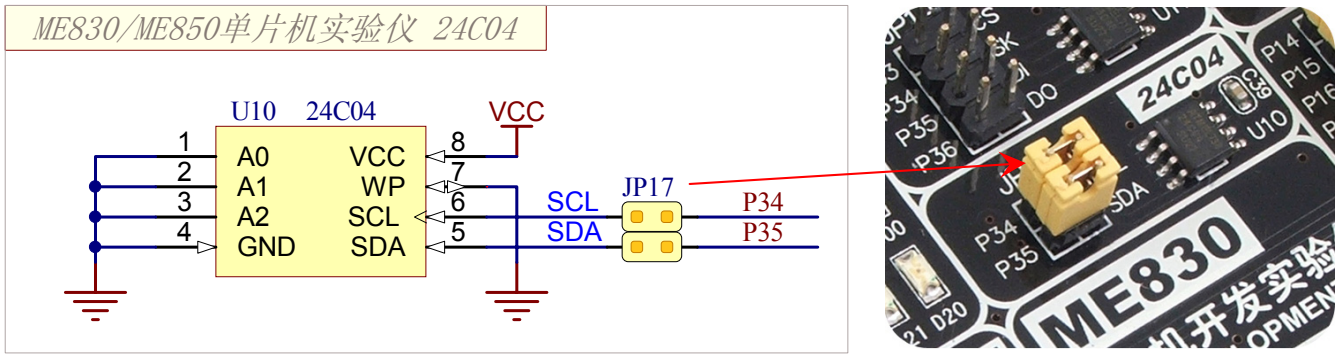


图 6.50　主程序流程图

3. 实验步骤

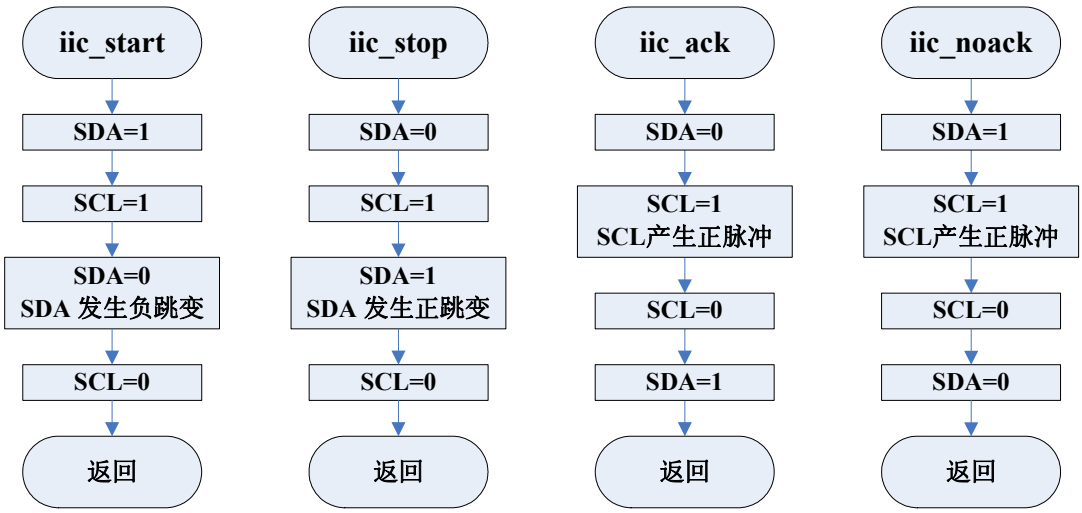
将 JP21 的 8 个短接子全部用短接帽短接，使 DG0～DG7 与 P2 端口接通。

将 JP22 的 9 个短接子全部用短接帽短接，使 A～DP 与 P0 端口接通，VCC 向数码管模块供电。

将 JP17 的 2 个短接子全部用短接帽短接，使芯片管脚与 P3.4～P3.5 端口接通。

备注：因为 C51 单片机芯片上没有专用的 I^2C 接口引脚，所以程序只能是采用模拟 I^2C 接口数据传输方式来编写。

4. 程序流程图



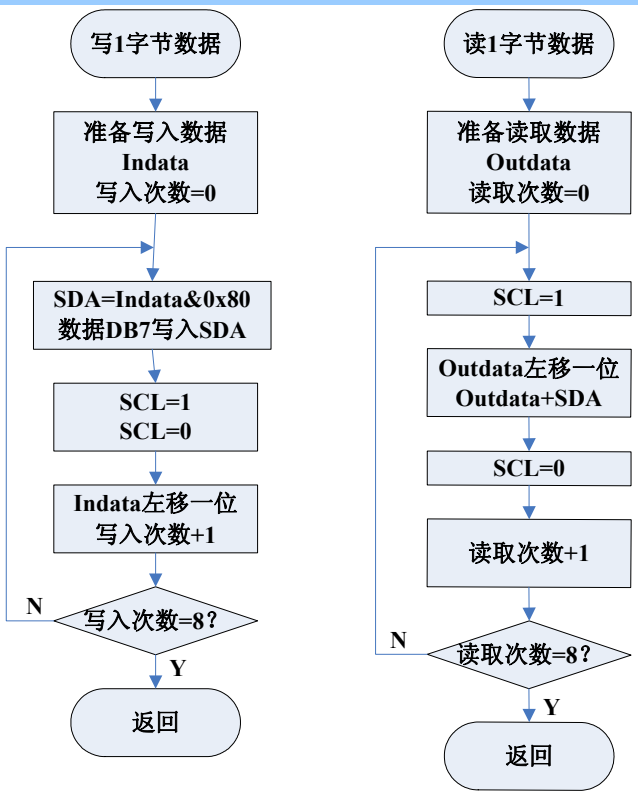


图 6.51　IIC 基础程序流程图

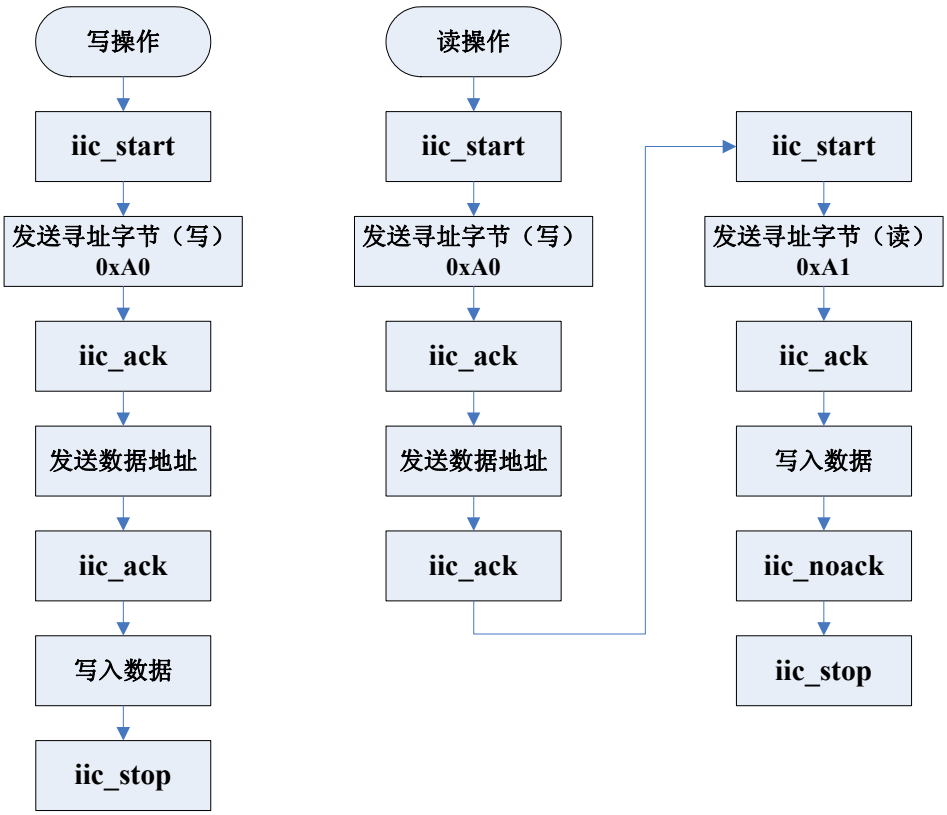


图 6.52　单字节读写操作流程图

5. 汇编源程序

（光盘： Example_A51\EX18_24C04）

```
;*********************************************************************
;*                                                                   *
;* ME830 单片机开发系统演示程序 - 24C04 读写                          *
;*                                                                   *
;* 8 位数码管显示显示                                                 *
;*                                                                   *
;* 版本： V1.0 (2008/08/20)                                          *
;* 作者： gguoqing (Email: gguoqing@willar.com)                      *
;* 网站： www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界)  *
;* 邮箱： willar@tom.com                                             *
;*                                                                   *
;*【版权】Copyright(C) 深圳硕飞科技有限公司  All Rights Reserved     *
;*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！        *
;*                                                                   *
;*********************************************************************
;*                                                                   *
;* 将数据先写入 24C04 芯片内，再将其数据逐个读出送数码管显示。       *
;*                                                                   *
;*********************************************************************

          SDA BIT P3.5           ;定义 24C04 数据线
          SCL BIT P3.4           ;定义 24C04 时钟线

          DISSTART  EQU  40H     ;显示单元首地址
          LED_DATA  EQU  P0      ;数码管数据口定义

;*************************************************************

          ORG 0000H
          AJMP MAIN
          ORG 0050H

;*************************************************************
MAIN:
         MOV  SP,#60H
         MOV  P0,#0FFH
         ACALL  WRITE_DATA     ;写数据入 24C04
         MOV R4,#04H           ;延时约 20ms
         ACALL DELAY_5MS       ;从 24C04 中读出数据
         ACALL  READ_DATA
LOOP:
         ACALL PLAY            ;显示
         AJMP  LOOP

;*************************************************************

;写 N 字节数据子程序
;查表写数据入 24C02

;*************************************************************
WRITE_DATA:
         MOV R0,#00H           ;数据写入首地址
         MOV R1,#8             ;共写入 8 个字节的数据
         MOV DPTR,#TAB_NU      ;表头首地址
WR_LOOP:
         CLR A
         MOVC A,@A+DPTR
         MOV B,A
         ACALL WRITE_BYTE      ;将查表结果写入 24C02
         INC R0                ;地址+1
         INC DPTR              ;数据指针+1
         DJNZ R1,WR_LOOP       ;8 个数写入完毕?
         RET

;*************************************************************

;读 N 字节数据子程序
;从 24C02 读出数据

;*************************************************************
READ_DATA:
```

```
        MOV R0,#00H          ;设定读取的初始地址
        MOV R3,#8            ;设定读取个数
        MOV R1,#DISSTART
RD_LOOP:
        ACALL READ_BYTE      ;读 EEPROM
        ACALL STOP
        MOV @R1,A             ;存储读出的数据
        INC R1
        INC R0                ;地址+1
        MOV R4,#04H           ;延时约 20ms
        ACALL DELAY_5MS
        DJNZ R3,RD_LOOP
        RET

;*******************************************************

;写操作子程序
;输入参数：R0---要写入的地址，B---要写入的数据

;*******************************************************
WRITE_BYTE:
       ACALL START

       MOV A,#0A0H
       ACALL  SENDBYTE
       ACALL  WAITACK

       MOV A,R0
       ACALL SENDBYTE
       ACALL  WAITACK

       MOV A,B
       ACALL SENDBYTE
       ACALL  WAITACK
       ACALL STOP

       MOV R4,#1           ;每写入 1 个字节，延时若干 MS
       ACALL DELAY_5MS
       RET

;*******************************************************

;读操作子程序
;输入参数：R0---要读的字节地址,
;输出参数：A---结果

;*******************************************************
READ_BYTE:
       ACALL START
       MOV A,#0A0H
       ACALL SENDBYTE
       ACALL  WAITACK

       MOV A,R0
       ACALL SENDBYTE
       ACALL  WAITACK

       ACALL START
       MOV A,#0A1H
       ACALL SENDBYTE
       ACALL  WAITACK
       ACALL  RCVBYTE
       RET

;*******************************************************

;从 IIC 总线上接收一个字节数据
;出口参数：A---接收数据存放在 A 中

;*******************************************************
RCVBYTE:
       MOV R7,#08          ;一个字节共接收 8 位数据
       CLR  A
       SETB  SDA           ;释放 SDA 数据线
```

```
R_BYTE:
        CLR  SCL
        ACALL DELAY_5US
        SETB SCL            ;启动一个时钟周期，读总线
        ACALL DELAY_5US
        MOV C,SDA           ;将 SDA 状态读入 C
        RLC A               ;结果移入 A
        SETB  SDA           ;释放 SDA 数据线
        DJNZ R7,R_BYTE      ;判断 8 位数据是否接收完全?
        RET

;*********************************************************

;向 IIC 总线发送一个字节数据
;入口参数：A---待发送数据存放在 A 中

;*********************************************************
SENDBYTE:
        MOV R7,#08
S_BYTE:
        RLC A
        MOV SDA,C
        SETB SCL
        ACALL DELAY_5US
        CLR SCL
        DJNZ R7,S_BYTE    ;8 位发送完毕?
        RET

;*********************************************************

;等待应答信号
;等待从机返回一个响应信号

;*********************************************************
WAITACK:
        CLR  SCL
        SETB SDA          ;释放 SDA 信号线
        ACALL DELAY_5US
        SETB SCL
        ACALL DELAY_5US
        MOV C,SDA
        JC  WAITACK       ;SDA 为低电平，返回了响应信号
        CLR  SDA
        CLR  SCL
        RET

;*********************************************************

;启动信号子程序

;*********************************************************
START:
        SETB SDA
        SETB SCL
        ACALL DELAY_5US
        CLR SDA
        ACALL DELAY_5US
        CLR SCL
        RET

;*********************************************************

;停止信号子程序

;*********************************************************
STOP:
        CLR SDA
        NOP
        SETB SCL
        ACALL DELAY_5US
        SETB SDA
        ACALL DELAY_5US
        CLR  SCL
        CLR  SDA
```

```
        RET

;*********************************************************

;延时 5US 子程序

;*********************************************************
DELAY_5US:
        NOP
        NOP
        NOP
        NOP
        RET

;*********************************************************

;延时 5MS 子程序
;输入参数：R4---R4*5MS

;*********************************************************
DELAY_5MS:
       MOV R6,#10
DE_LP:
       MOV R5,#250
       DJNZ R5,$
       DJNZ R6,DE_LP
       DJNZ R4,DELAY_5MS
       RET

;*********************************************************

; 显示子程序

;*********************************************************
PLAY:
        MOV R0,#DISSTART        ;获得显示单元首地址
        MOV R1,#0FEH            ;位码初始值
        MOV R2,#08H             ;8 位数码管显示
DISP1:
        MOV A,@R0               ;获得当前位的段码
        MOV LED_DATA,A          ;输出段码
        MOV P2,R1               ;输出位码
        MOV A,R1                ;调整位码
        RL A
        MOV R1,A                ;保存位码
        INC R0                  ;取下一个显存单元地址
        ACALL DELAY2MS          ;延时 2 MS
        DJNZ R2,DISP1           ;8 位数码管是否显示完毕
        MOV  P2,#0FFH           ;关闭显示
        RET                     ;显示完成，返回

;*********************************************************

;延时子程序

;*********************************************************
DELAY2MS:
        MOV  R6,#10
DEL1:
        MOV  R7,#93
        DJNZ  R7,$
        DJNZ  R6,DEL1
        RET

;*********************************************************
TAB_NU:
    DB 0C0H,0F9H,0A4H,0B0H,099H,092H,082H,0F8H
    DB 080H,090H,0FFH,088H,083H,0C6H,0A1H,086H,08EH

;*********************************************************

        END          ;结束
;*********************************************************
```

6. C 语言源程序

（光盘： Example_C51\EX18_24C04）

```
/*****************************************************************
 *                                                               *
 * ME830 单片机开发实验仪演示程序 - 24C04 读写                    *
 *                                                               *
 * 8 位数码管显示                                                *
 *                                                               *
 * 时间： 2008/08/20                                             *
 * 作者： gguoqing (gguoqing@willar.com)                         *
 * 网站： www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界) *
 * 邮箱： willar@tom.com                                         *
 *                                                               *
 *【版权】Copyright(C)伟纳电子 www.willar.com  All Rights Reserved *
 *【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！     *
 *                                                               *
 *****************************************************************
 *                                                               *
 * 将 0-7 显示数据先写入 24C04 芯片，再将其数据逐个读出送数码管显示。 *
 *                                                               *
 *****************************************************************/

#include <reg52.h>
#include <intrins.h>

#define OP_WRITE 0xa0          // 器件地址以及写入操作
#define OP_READ  0xa1          // 器件地址以及读取操作

unsigned char data display[] =
{
  0xc0, 0xc0, 0xc0, 0xc0, 0xc0, 0xc0, 0xc0, 0xc0
};

unsigned char code sendbuf[] =
{
  0xc0, 0xF9, 0xA4, 0xB0, 0x99, 0x92, 0x82, 0xF8, 0x80, 0x90
};

sbit SDA = P3 ^ 5;
sbit SCL = P3 ^ 4;

char code reserve[3]_at_ 0x3b; //保留 0x3b 开始的 3 个字节

/*************************************************************

延时子程序   (4.34us)

 *************************************************************/
void delayNOP(void)
{
  _nop_();
  _nop_();
  _nop_();
  _nop_();
}

/*************************************************************

延时子程序

 *************************************************************/
void delayms(unsigned int ms)
{
  unsigned char k;
  while (ms--)
  {
    for (k = 0; k < 114; k++)
      ;
  }
}

/*************************************************************
```

```
启动子程序

在 SCL 高电平期间 SDA 发生负跳变

 *********************************************************/
void iic_start()
{
  SDA = 1;
  SCL = 1;
  delayNOP();
  SDA = 0;
  delayNOP();
  SCL = 0;
}

/*********************************************************

停止子函数

在 SCL 高电平期间 SDA 发生正跳变

 *********************************************************/
void iic_stop()
{
  SDA = 0;
  SCL = 1;
  delayNOP();
  SDA = 1;
  delayNOP();
  SCL = 0;
}

/*********************************************************

IIC 初始化子程序

 *********************************************************/
void iic_init()
{
  SCL = 0;
  iic_stop();
}

/*********************************************************

发送应答位子函数

在 SDA 低电平期间 SCL 发生一个正脉冲

 *********************************************************/
void iic_ack()
{
  SDA = 0;
  SCL = 1;
  delayNOP();
  SCL = 0;
  SDA = 1;
}

/*********************************************************

发送非应答位子函数

在 SDA 高电平期间 SCL 发生一个正脉冲

 *********************************************************/
void iic_noack()
{
  SDA = 1;
  SCL = 1;
  delayNOP();
  SCL = 0;
  SDA = 0;
}
```

```
/*************************************************************

应答位检测子函数

 *************************************************************/
/*
bit iic_testack()
{
bit ack_bit;

SDA = 1;           //置 SDA 为输入方式
SCL = 1;
delayNOP();
ack_bit = SDA;
SCL = 0;
delayNOP();

return ack_bit;  //返回 AT24C04 应答位
}
 */
/*************************************************************

读一个字节子程序

从 AT24C04 读数据到 MCU

 *************************************************************/
/*
unsigned char readbyte()
{
unsigned char i,read_data;

for(i = 0; i < 8; i++)
{
read_data <<= 1;
if(iic_testack())
read_data++;
}
return(read_data);
}
 */

/*************************************************************

读一个字节子程序

从 AT24C04 读数据到 MCU

 *************************************************************/
unsigned char readbyte()
{
  unsigned char i, read_data;
  SDA = 1; //置 SDA 为输入方式
  for (i = 0; i < 8; i++)
  {
    SCL = 1; //使 SDA 数据有效
    read_data <<= 1; //调整接收位
    if (SDA)
    //读 SDA
      read_data++;
    SCL = 0; //继续接收数据
  }
  return (read_data);
}


/*************************************************************

发送一个字节子程序

从 MCU 移出数据到 AT24C04

 *************************************************************/
```

```
void writebyte(unsigned char write_data)
{
  unsigned char i;

  for (i = 0; i < 8; i++)
  // 循环移入 8 个位
  {
    SDA = (bit)(write_data &0x80); //将发送位送入 SDA 数据线
    SCL = 1;
    delayNOP();
    SCL = 0; //SDA 数据线上数据变化
    write_data <<= 1; //调整发送位
  }
}

/*********************************************************

在指定地址 addr 处写入 N 个数据

 *********************************************************/
void write_nbyte(unsigned char addr, unsigned char n)
{
  unsigned char x;
  iic_start();
  writebyte(OP_WRITE); //写 0xa0
  iic_ack();
  writebyte(addr); //写存储地址
  iic_ack();
  while (n--)
  {
    writebyte(sendbuf[x++]); //写数据
    iic_ack();
    delayms(1);
  }
  iic_stop(); //发送结束
}

/*********************************************************

在指定地址 addr 处读出 N 个数据

 *********************************************************/
void read_nbyte(unsigned char addr, unsigned char n)
{
  unsigned char x = 0;

  iic_start();

  writebyte(OP_WRITE); //写 0xa0
  iic_ack();
  writebyte(addr); //写读取地址
  iic_ack();
  iic_start();
  writebyte(OP_READ); //写 0xa1
  iic_ack();

  while (n--)
  {
    display[x++] = readbyte(); //读出数据写入相应显存单元
    iic_ack(); //发送应答位
    delayms(1);
  }
  iic_noack(); //发送非应答位
  iic_stop(); //发送结束
}

/*********************************************************

主函数

 *********************************************************/
void main(void)
{
  unsigned char k, shift;
```

```
    iic_init();

    write_nbyte(0, 8); //写入显示代码到 AT24C04
    delayms(100); //延时 100ms，等待芯片自动编程完毕
    read_nbyte(0, 8); //从 AT24C04 读出显示代码

    while (1)
    {
      shift = 0xfe; //位码初始值
      P2 = 0xff; //关闭显示
      for (k = 0; k < 8; k++)
      {
        P0 = display[k]; //段码输出
        P2 = shift; //位码输出
        shift = _crol_(shift, 1); //修改位码
        delayms(1); //延时 1ms
      }
    }
}

 /*************************************************************/
```

实验十九　PCF8591 A/D 转换实验

PCF8591 是具有 I2C 总线接口的 8 位 A/D 及 D/A 转换器。有 4 路 A/D 转换输入，1 路 D/A 模拟输出。它既可以作 A/D 转换也可以作 D/A 转换。A/D 转换为逐次比较型。

I2C 总线是 Philips 公司推出的串行总线，整个系统仅靠数据线（SDA）和时钟线（SCL）实现完善的全双工数据传输，即 CPU 与各个外围器件仅靠这两条线实现信息交换。I2C 总线系统与传统的并行总线系统相比具有结构简单、可维护性好、易实现系统扩展、易实现模块化标准化设计、可靠性高等优点。

在一个完整的单片机系统中，A/D 转换芯片往往是必不可少的。PCF8591 是一种具有 I2C 总线接口的 A/D 转换芯片。在与 CPU 的信息传输过程中仅靠时钟线 SCL 和数据线 SDA 就可以实现。

1. 实验任务

4 个 A/D 输入通道：IN0 与电位器 VR1 连接用于测量电位器的输出电压。

IN1、IN2 和 IN3 可测量通过 J13 输入的外接电压。（输入电压要求<=5V）。

D/A 输出：将 IN0 的 A/D 转换值送 D/A 输出，DL10 随 D/A 输出值的大小改变亮度。

LCD1602 同时显示四通道 A/D 转换值，并将 IN0 通道的 A/D 转换值送 D/A 输出。

2. 实验线路

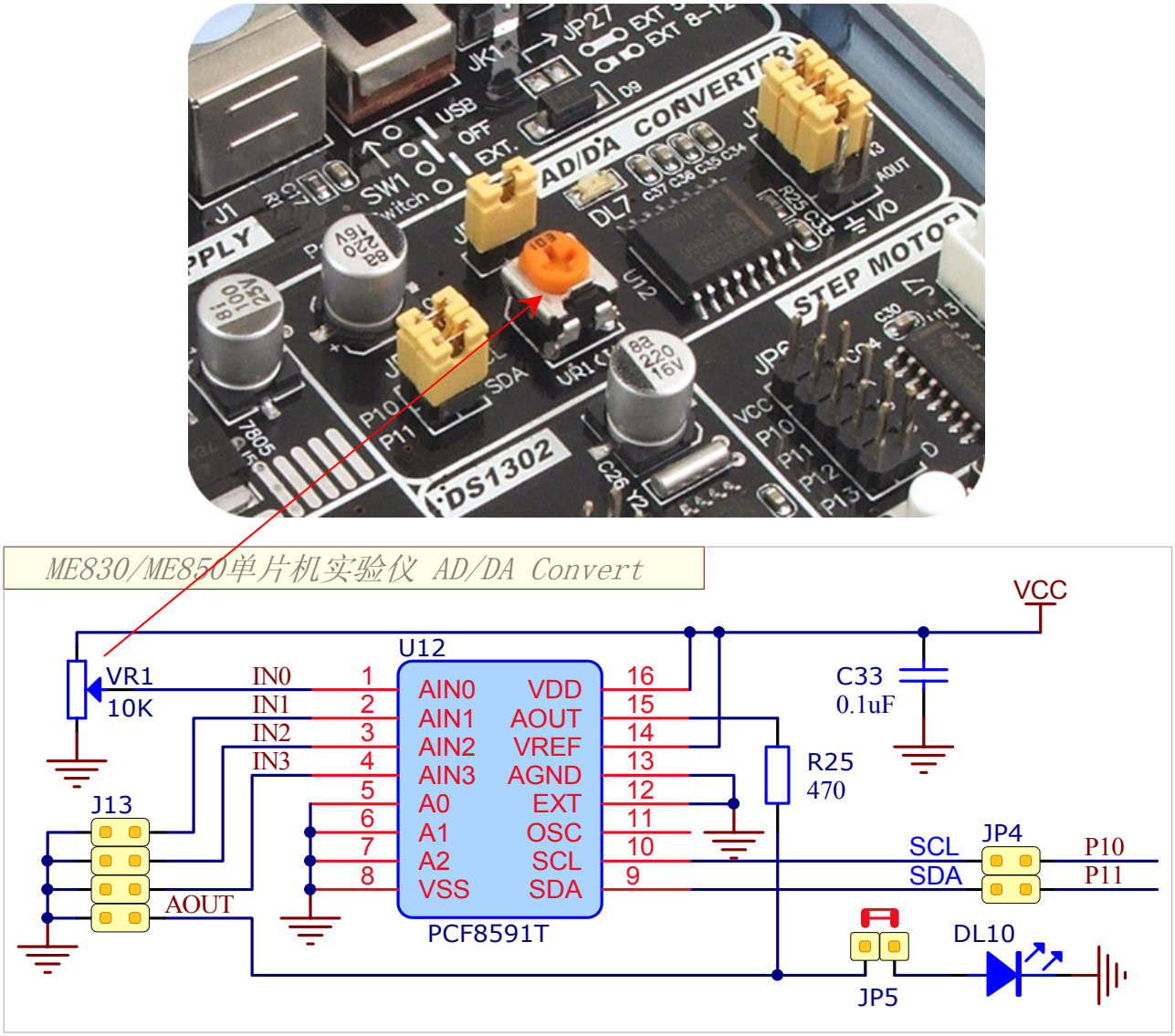


图 6.53　ADC/DAC(PCF8591)原理图

3. 实验步骤

将 1602LCD 模块插在 J5 接口上；

将 JP4 的短接子短接，使芯片管脚与 P1.0～P1.1 端口接通；

将 JP5 的短接子短接，使 DL10（LED）与芯片 D/A 输出连接；

注意：如果不使用 IN1、IN2、IN3 做外接电压输入时，请将 J13 上相应通道的短接子短接，短接后的相应通道输入端与地连接，避免引入干扰。

4. 程序流程图

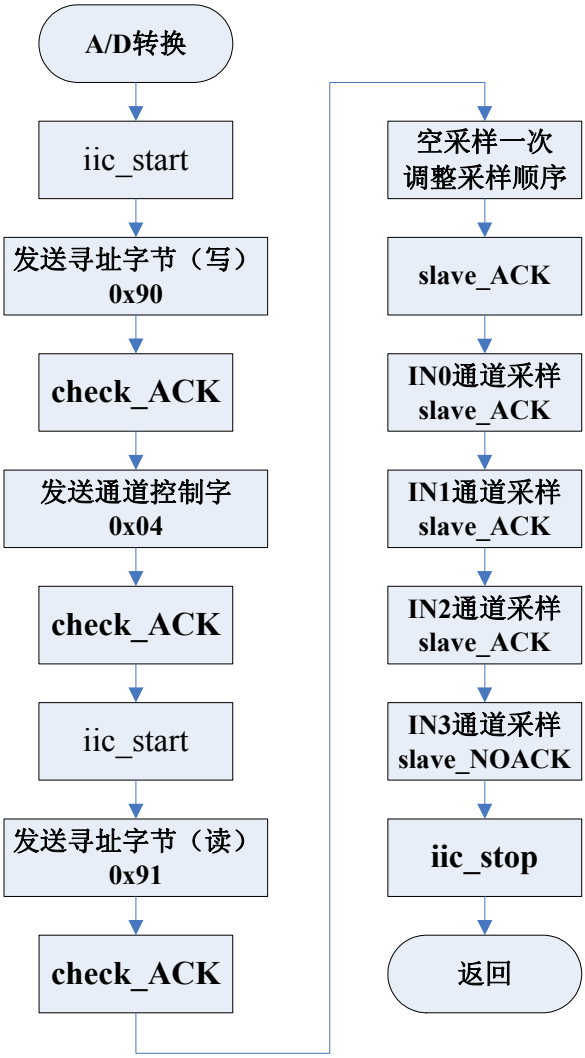
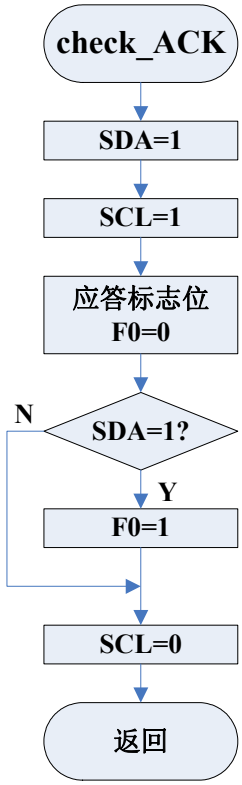


图 6.54　A/D 转换程序流程图



应答位检查

图 6.55　应答位检查程序流程图

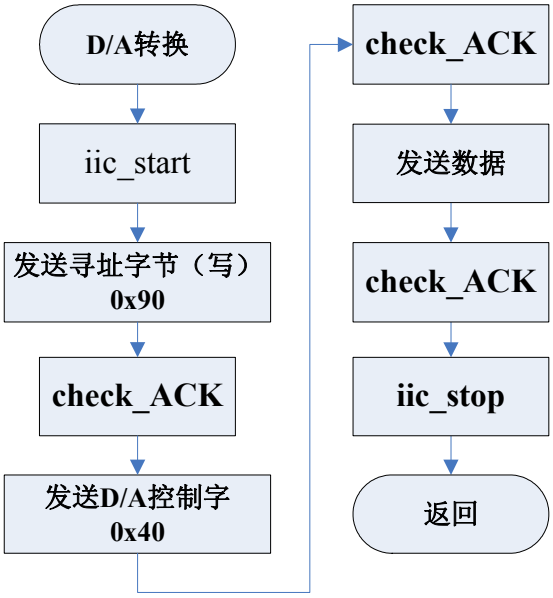


图 6.56　D/A 转换程序流程图

IIC 的基本子程序的流程图参阅 EX17_24C04 程序流程图。

5. 汇编源程序

（见光盘： Example_A51\EX19_ADC）

6. C 语言源程序

（见光盘： Example_C51\EX19_ADC）

实验二十　PCF8591 D/A 转换实验

1. 实验任务

利用 D/A 输出功能，通过输入不同的数据方式，可产生各种波形的输出脉冲。

通过按键 K2 选择输出波形的类形，K3 键启动，K4 键停止。

使用 LCD1602 显示菜单及程序运行过程。

可以使用 DL10（LED）来显示输出脉冲，也可以将示波器探头接在 JP5 的引脚上进行测试和观察。

2. 实验线路

见图 6.53

3. 实验步骤

将 1602LCD 模块插在 J5 接口上；

将 JP4 的短接子短接，使芯片管脚与 P1.0～P1.1 端口接通。

将 JP5 的短接子短接，使 DL10 与芯片 D/A 输出连接，用 LED 显示。

如果使用示波器观测时，请将 JP5 短接子上的短接帽取掉，否则波形失真变形。

4. 程序流程图

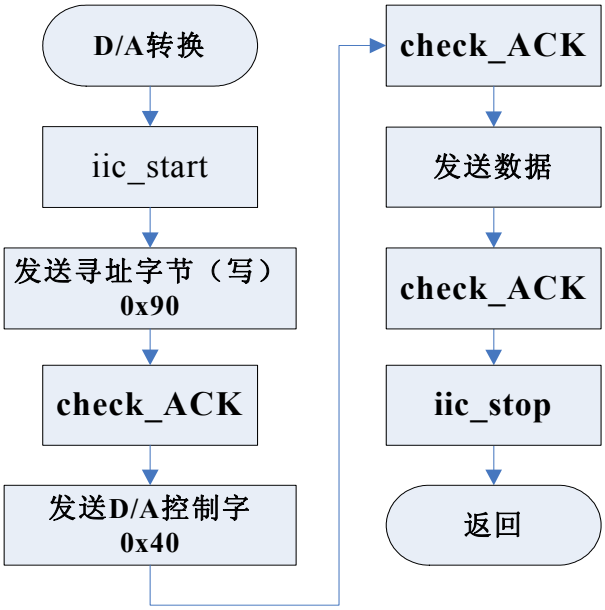


图 6.57　D/A 转换程序流程图

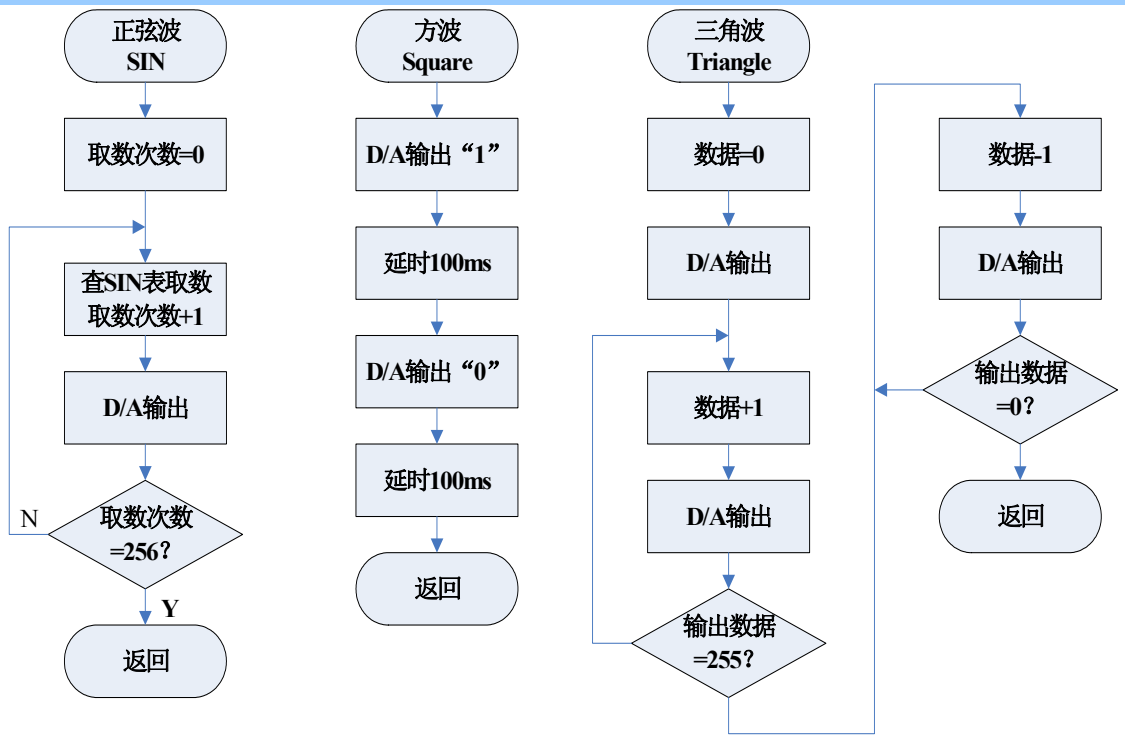


图 6.58　波形输出程序流程图

5. 汇编源程序

（见光盘： Example_A51\EX20_DAC）

6. C 语言源程序

（见光盘： Example_C51\EX20_DAC）

实验二十一　DS1302 实时时钟

DS1302 是美国 DALLAS 公司推出的一种高性能、低功耗、带 RAM 的实时时钟电路，它可以对年、月、日、周日、时、分、秒进行计时，具有闰年补偿功能，工作电压为 2.5V～5.5V。采用三线接口与 CPU 进行同步通信，并可采用突发方式一次传送多个字节的时钟信号或 RAM 数据。DS1302 内部有一个 31×8 的用于临时性存放数据的 RAM 寄存器。DS1302 增加了主电源/后背电源双电源引脚，同时提供了对后背电源进行涓细电流充电的能力。

1. 实验任务

DS1302 时钟芯片包括实时时钟/日历，提供秒、分、时、日、周、月和年等信息。

用 LCD1602 显示从 DS1302 读出的 年、月、日、星期、时、分、秒 的实时值。

同时按下 K1 和 K4 键将由程序预设的日期和时间数据写入 DS1302 芯片内。

2. 实验线路

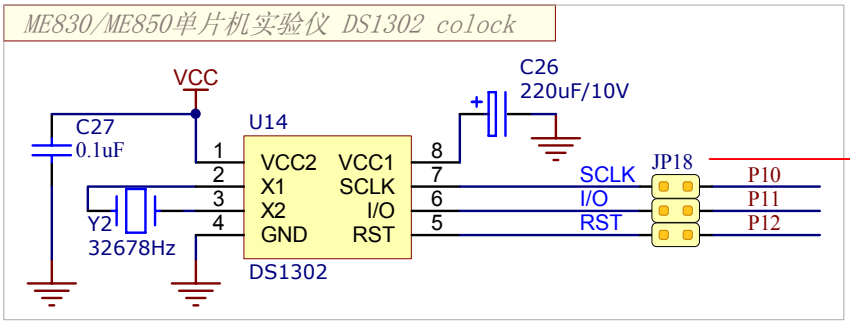


图 6.59　DS1302 原理图

3. 实验步骤

将 1602LCD 模块插在 J5 接口上；

将 JP18 短接子短接，使芯片管脚与 P1.0～P1.2 端口接通。

4. 程序流程图

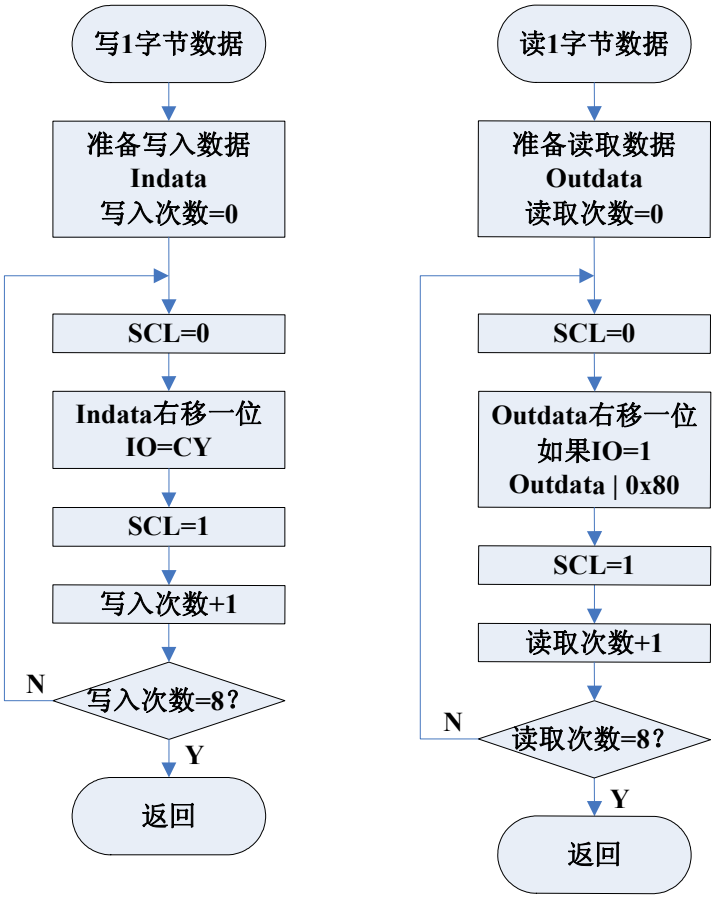


图 6.60　读写 1 字节子程序流程图

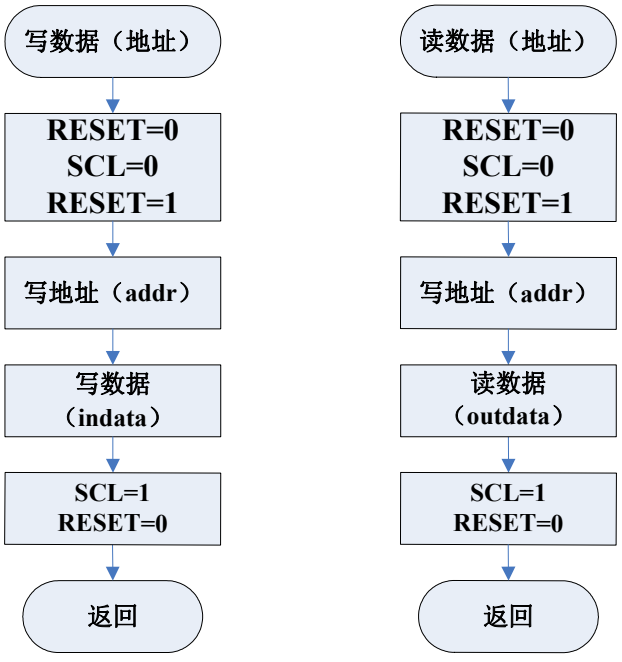


图 6.61　固定地址读写 1 字节程序流程图

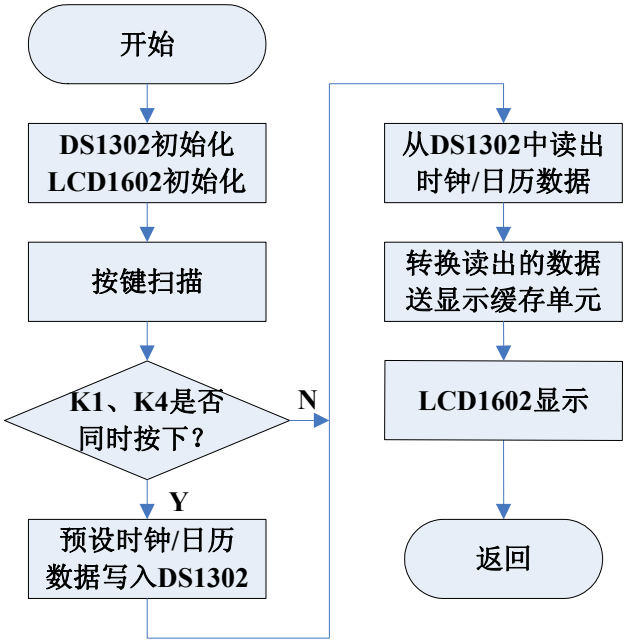


图 6.62　主程序流程图

5. 汇编源程序

（见光盘： Example_A51\EX21_DS1302）

6. C 语言源程序

（见光盘： Example_C51\EX21_DS1302）

实验二十二　DS18B20 数字温度传感器

DS18B20是Dallas公司生产的单总线(One-wire)数字化温度传感器，它采用单根信号线传输数据，而且数据传输是双向的。它能直接读出被测温度，因此可以通过简单的编程实现温度显示与温度控制。

DS18B20具有如下特性：

- 测温范围： -55 ～ 125 ℃
- 转换精度：9 ～ 12 位（包括符号位），可编程决定转换精度的位数。
- 测温分辨率：9 位精度为 0.5 ℃，12 位精度为 0.0625 ℃。
- 转换时间：9 位精度为 93.75 ms，12 位精度为 750ms。
- 具有非易失性，上、下报警设定功能。

1. 实验任务

DS18B20 检测温度，用 6 位数码管显示实际温度值和符号。

当检测到 DS18B20 不存在或有问题时，蜂鸣器将报警，数码管黑屏。

2. 实验线路

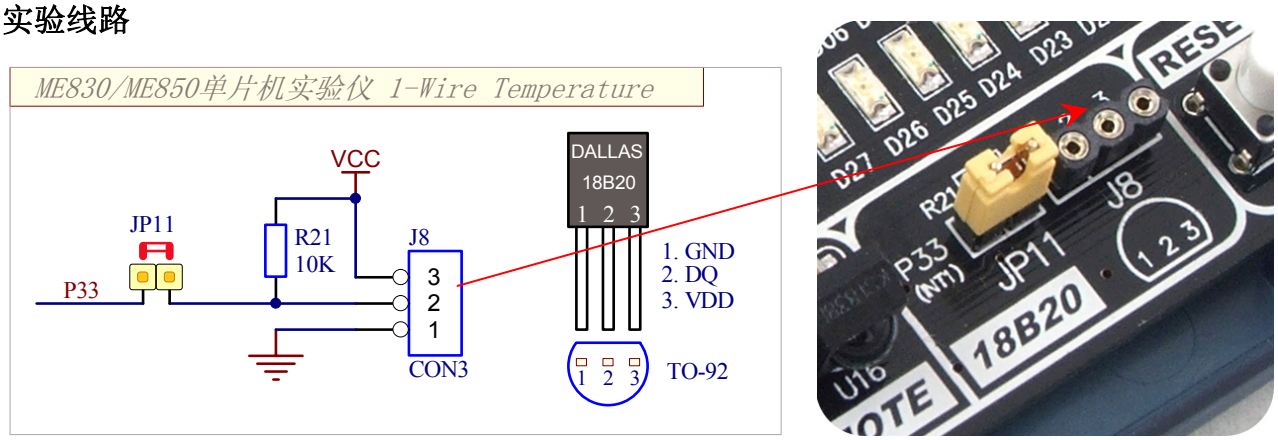


图 6.63　DS18B20 接口原理图

3. 实验步骤

将 JP21 的 8 个短接子全部用短接帽短接，使 DG0～DG7 与 P2 端口接通。

将 JP22 的 9 个短接子全部用短接帽短接，使 A～DP 与 P0 端口接通，VCC 向数码管模块供电。

将 JP11 的短接子用短接帽短接，使 DS18B20 的数据线与 P3.3 端口接通。

4. 程序流程图

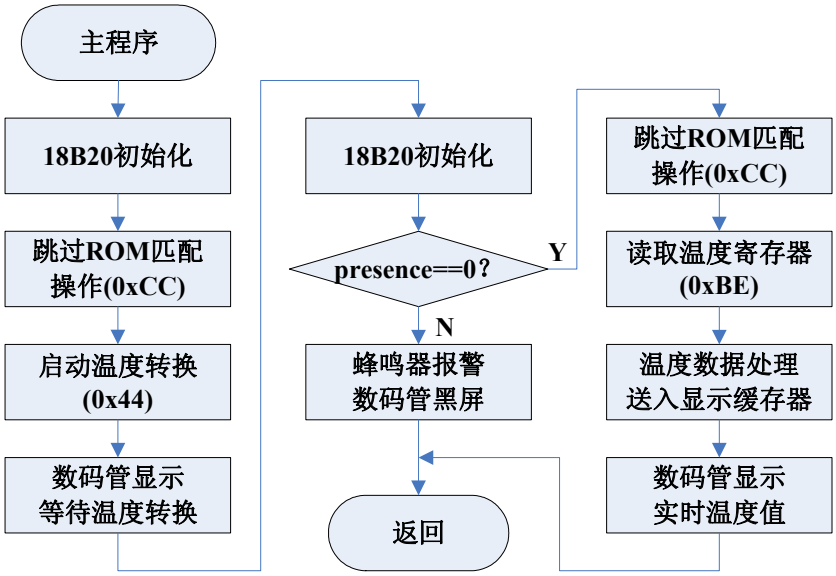


图 6.64　主程序流程图

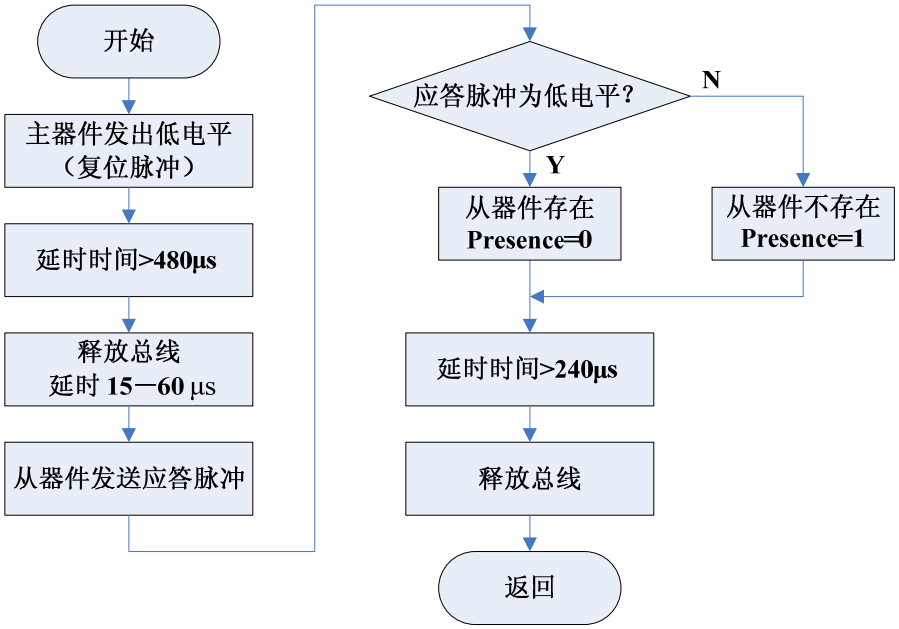


图 6.65　初始化子程序流程图

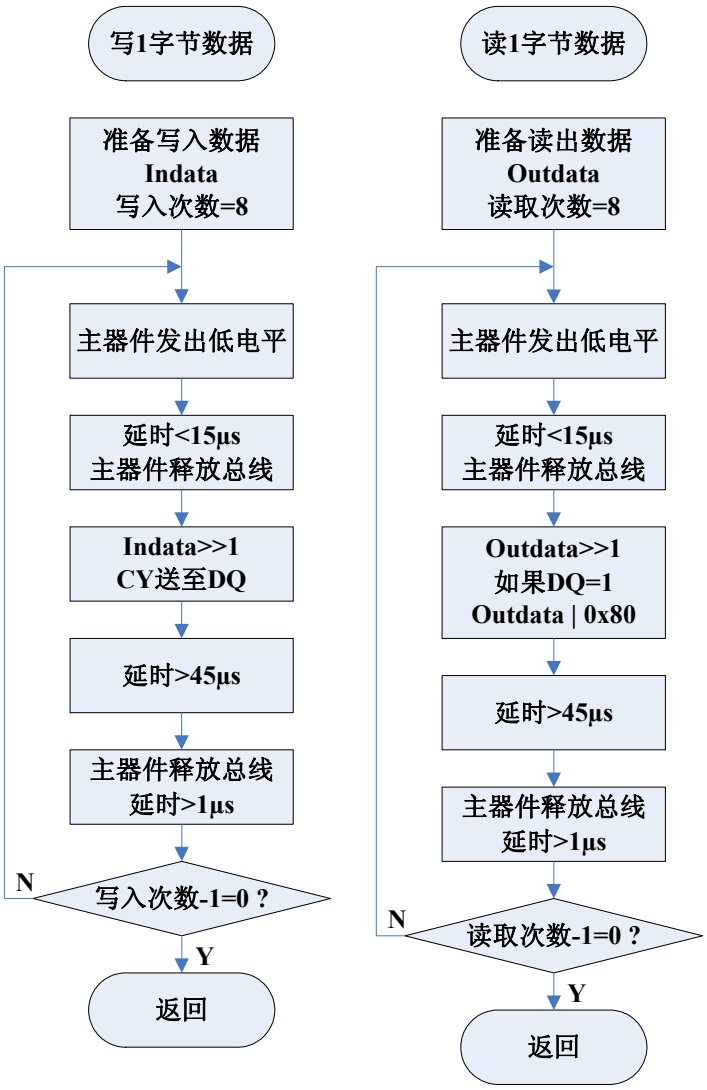


图 6.66　字节读写子程序流程图

5. 汇编源程序

（光盘： Example_A51\EX22_DS18B20）

```
;*********************************************************************
;*                                                                   *
;* ME830 单片机开发系统演示程序 - DS18B20 温度显示                   *
;*                                                                   *
;* 6 位数码管显示                                                    *
;*                                                                   *
;* 版本：  V1.0 (2008/08/24)                                         *
;* 作者：  gguoqing (Email: gguoqing@willar.com)                     *
;* 网站：  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界)*
;* 邮箱：  willar@tom.com                                            *
;*                                                                   *
;*【版权】Copyright(C) 深圳硕飞科技有限公司  All Rights Reserved    *
;*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！       *
;*                                                                   *
;*********************************************************************

        TEMPL       EQU  30H
        TEMPH       EQU  31H

        LEDPLAY     EQU  40H        ;数码管显示缓存单元

        BEEP        BIT   P3.7      ;蜂鸣器控制线
        DATA_LINE   BIT   P3.3      ;DS18B20 DQ 线

        PRESENCE    BIT   20H.0     ;DS18B20 存在标志位
        FLAG        BIT   20H.1     ;温度符号标志位
```

```
;*********************************************************

        ORG 0000H
        AJMP  MAIN
        ORG  0050H

;*********************************************************
MAIN:
        MOV  SP,#60H              ;设置堆栈指针
        MOV  P0,#0FFH
        MOV  P2,#0FFH

        SETB  PRESENCE
        SETB  FLAG
        SETB  BEEP

        MOV  LEDPLAY,#0BH         ;赋初始化显示字符 C
        MOV  LEDPLAY+1,#0CH       ;显示 °
        MOV  LEDPLAY+2,#0DH       ;显示 -
        MOV  LEDPLAY+3,#0DH
        MOV  LEDPLAY+4,#0DH
        MOV  LEDPLAY+5,#0DH
LOOP:
        ACALL RESET               ;复位与检测 DS18B20
        JNB PRESENCE,LOOP1
        MOV  P2,#0FFH             ;关闭数码管显示
        ACALL  BEEP_BL            ;DS18B20 错误，报警
        AJMP  LOOP
LOOP1:
        MOV A,#0CCH               ;跳过 ROM 匹配
        ACALL WRITE
        MOV A,#044H               ;发出温度转换命令
        ACALL WRITE

        MOV R4,#130               ;利用显示程序延时
LOOP2:
        ACALL  TEMP_PLAER         ;延时>750ms
        DJNZ  R4,LOOP2           ;等待温度转换完成

        ACALL RESET              ;复位与检测 DS18B20
        JB PRESENCE,LOOP
        MOV A,#0CCH               ;跳过 ROM 匹配
        ACALL WRITE
        MOV A,#0BEH               ;发出读温度命令
        ACALL WRITE

        ACALL  READ_TEMP          ;读温度数据
        ACALL  CONVTEMP           ;温度数据处理
        MOV R4,#30
LOOP3:
        ACALL  TEMP_PLAER         ;显示实时温度
        DJNZ  R4,LOOP3

        AJMP  LOOP                ;返回 LOOP

;*********************************************************

;DS18B20 复位与检测子程序

;PRESENCE=0 OK, PRESENCE=1 ERROR

;*********************************************************
RESET:
        SETB DATA_LINE            ;释放数据线
        NOP
        NOP
        CLR DATA_LINE             ;主机发出复位低脉冲

        MOV B,#64H                ;延时 600us
        MOV R1,#03H
RESET1:
        DJNZ B, RESET1
        MOV B,#64H
```

```
        DJNZ R1,RESET1

        SETB DATA_LINE          ;释放数据线
        NOP
        MOV B,#22H
RESET2:
        DJNZ B,RESET2           ;延时 64us
        JNB DATA_LINE,RESET3    ;检测 DS18B20 应答位
        AJMP RESET4
RESET3:
        MOV  B,#210             ;延时 420us
        DJNZ  B,$
        JNB DATA_LINE,RESET4    ;判 DS1820 释放数据线
        CLR  PRESENCE           ;清标志位,表示 DS1820 存在
        AJMP RESET5
RESET4:
        SETB PRESENCE           ;置标志位,表示 DS1820 不存在
RESET5:
        SETB DATA_LINE          ;释放数据线
        RET

;********************************************************

; 写数据子程序

;********************************************************
WRITE:
        MOV R2,#8               ;一个字节 8 个 bit
        CLR C
WR1:
        CLR DATA_LINE           ;开始写时,数据线要处于低电平
        MOV B,#05
        DJNZ B,$                ;延时 10us
        RRC A                   ;把一个 BIT 环移给 C
        MOV DATA_LINE,C         ;写入一个 BIT
        MOV B,#23
        DJNZ B,$                ;延时 46us
        SETB DATA_LINE          ;释放总线,表示此次位写操作完成
        DJNZ R2,WR1             ;写入下一个 BIT
        RET

;********************************************************

; 读数据子程序

;********************************************************
READ:
        MOV R2,#8               ;一个字节 8 个 bit
        SETB DATA_LINE
        CLR C
RD1:
        CLR DATA_LINE           ;清数据线为低
        NOP                       ;
        NOP
        SETB DATA_LINE          ;释放数据线
        NOP
        NOP
        MOV C,DATA_LINE         ;送数据位至 C
        RRC A                   ;把读得的位值移入 A
        MOV B,#23
        DJNZ B,$                ;延时 46us
        SETB DATA_LINE          ;释放总线,表示此次位读操作完成
        DJNZ R2,RD1             ;读下一个 BIT
        RET

;********************************************************

;读温度子程序

;从 DS18B20 中读出温度低位、高位数据
;存入 TEMPL 开始的 2 个存储单元

;********************************************************
READ_TEMP:
```

```
          MOV R4,#2
          MOV R1,#TEMPL
RE00:
          ACALL  READ
          MOV @R1,A
          INC R1
          DJNZ R4,RE00
          RET

;*********************************************************

;  温度数据处理子程序

;*********************************************************
CONVTEMP:
         MOV  A,TEMPH              ;判温度是否零下
         ANL  A,#80H
         JZ  TEMPC1                ;温度零上转
         CLR  C
         MOV  A,TEMPL              ;二进制数求补（双字节）
         CPL  A                    ;取反
         ADD  A,#01H               ;加 1
         MOV  TEMPL,A
         MOV  A,TEMPH
         CPL  A
         ADDC  A,#00H
         MOV  TEMPH,A
         CLR  FLAG                 ;温度是零下,清温度符号标志位
         SJMP  TEMPC11
TEMPC1:
         SETB  FLAG                ;温度是零上,置温度符号标志位
TEMPC11:
         MOV  A,TEMPL
         ANL  A,#0FH               ;取小数部分数据
         MOV  DPTR,#TEMPDOTTAB      ;小数表首地址
         MOVC  A,@A+DPTR
         MOV  LEDPLAY+2,A          ;小数部分存入

         MOV  A,TEMPL              ;整数部分
         ANL  A,#0F0H              ;取 TEMPL 的高四位
         SWAP  A
         MOV  TEMPL,A
         MOV  A,TEMPH              ;取 TEMPH 的低四位
         ANL  A,#0FH
         SWAP  A
         ORL  A,TEMPL              ;组合后的值存入 A

         LCALL  HEX2BCD1           ;转换为 BCD 码

         MOV  TEMPL,A
         ANL  A,#0FH
         MOV  LEDPLAY+3,A          ;个位数存入

         MOV  A,TEMPL
         SWAP  A
         ANL  A,#0FH
         MOV  LEDPLAY+4,A      ;十位数存入

         MOV  A,R7                  ;R7 - 百位数
         JZ  TEMPC12               ;如果百位数为"0"转
         ANL  A,#0FH
         MOV  LEDPLAY+5,A          ;百位数存入
         AJMP  TEMPC15
TEMPC12:
         JNB  FLAG,TEMPC13
         MOV  LEDPLAY+5,#0AH       ;百位为"0"不显示
         MOV  A,LEDPLAY+4
         JNZ  TEMPC15
         MOV  LEDPLAY+4,#0AH       ;十位为"0"不显示
         AJMP TEMPC15
TEMPC13:
         MOV  A,  LEDPLAY+4
         JZ  TEMPC14               ;如果十位数为"0"转
         MOV  LEDPLAY+5,#0DH       ;显示"-"符号
```

```
        AJMP   TEMPC15
TEMPC14:
        MOV   LEDPLAY+5,#0AH      ;不显示
     MOV   LEDPLAY+4,#0DH      ;显示"-"符号
TEMPC15:
        RET

;*********************************************************

;  温度小数部分码表

;*********************************************************
TEMPDOTTAB:
  DB    00H,01H,01H,02H,03H,03H,04H,04H
  DB    05H,06H,06H,07H,08H,08H,09H,09H

;*********************************************************

; 温度显示子程序

;*********************************************************
TEMP_PLAER:
        MOV   R0,#LEDPLAY           ;指向显示缓存首址
        MOV   R5,#0FEH              ;位码初值
PLAY:
        MOV   A,@R0                 ;取显示数据到 A
        MOV   DPTR,#TABLE           ;取段码表地址
        MOVC   A,@A+DPTR           ;查显示数据对应段码
        MOV   P0,A                  ;输出段码
        MOV   A,R5
        MOV   P2,A                  ;输出位码
        JB    ACC.3,LOOP5           ;小数点处理
        CLR   P0.7
LOOP5:
        ACALL   DL_MS               ;延时 1ms
        INC   R0                    ;指向下一个显存地址
        MOV   A,R5
        JNB   ACC.5,ENDOUT          ;ACC.5=0 时,一次显示结束
        RL   A                      ;修改位码
        MOV   R5,A                  ;存入 R5 中
        AJMP   PLAY                 ;返回 PLAY 循环
ENDOUT:
        MOV   P2,#0FFH              ;关闭显示
        RET

;*********************************************************

; 数码管段码表

;*********************************************************
TABLE:
 DB   0C0H,0F9H,0A4H,0B0H,99H,92H,82H,0F8H
;    “0”  “1” “2” “3” “4”“5”“6”“7”
 DB   80H,90H,0FFH,0C6H,9CH,0BFH,0C7H,89H
;     “8”“9”“灭”   C       "-"   "L" "H"

;*********************************************************

; 1MS 延时程序

; LED 显示程序用

;*********************************************************
DL_MS:
        MOV   R6,#2
DL1:
        MOV   R7,#230
DL2:
        DJNZ   R7,DL2
        DJNZ   R6,DL1
        RET

;*********************************************************
```

```
;单字节十六进制转 BCD

;输入：(A)- 待转换数据
;输出:R7-百位,A-十位和个位(压缩 BCD 码)

;*************************************************************
HEX2BCD1:
        PUSH   B
        MOV    B,#64H           ;除数为 100
        DIV    AB
        MOV    R7,A             ;百位数据存入 R7
        MOV    A,#0AH           ;除数为 10
        XCH    A,B              ;A 与 B 数据交换
        DIV    AB
        SWAP   A                   ;高低 4 位数据交换
        ORL    A,B              ;十位和个位数据存入 A
        POP    B
        RET

;*************************************************************

;蜂鸣器驱动子程序

;*************************************************************
BEEP_BL:
        MOV    R6,#180
 BL1:
        ACALL   DEX1
        CPL    BEEP             ;对 P3.7 取反
        DJNZ   R6,BL1
        MOV    R5,#10
        ACALL   DELAY
        RET

;*************************************************************

; 650us 延时子程序

; 蜂鸣器驱动程序用

;*************************************************************
DEX1:
        MOV    R7,#200
DEX2:
        NOP
        DJNZ   R7,DEX2
        RET

;*************************************************************

; (R5)*10MS    延时子程序

;*************************************************************
DELAY:
        MOV    R6,#10
DEL1:
        MOV    R7,#230
DEL2:
        DJNZ   R7,DEL2
        DJNZ   R6,DEL1
        DJNZ   R5,DELAY
        RET

;*************************************************************

        END             ;结束

;*************************************************************
```

6. C 语言源程序

（光盘： Example_C51\EX22_DS18B20）

```
/*****************************************************************
 *                                                               *
 * ME830 单片机开发实验仪演示程序 - DS18B20 温度显示             *
 *                                                               *
 * 6 数码管显示                                                  *
 *                                                               *
 * 版本： V1.0 (2008/08/23)                                      *
 * 作者： gguoqing (Email: gguoqing@willar.com)                  *
 * 网站： www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界) *
 * 邮箱： willar@tom.com                                         *
 *                                                               *
 *【版权】Copyright(C)伟纳电子 www.willar.com  All Rights Reserved *
 *【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！     *
 *                                                               *
 *****************************************************************/

#include <reg52.h>
#include <intrins.h>

sbit DQ = P3 ^ 3; //定义 DS18B20 端口 DQ
sbit BEEP = P3 ^ 7; //定义蜂鸣器控制端口

bit presence;

unsigned char code LEDData[] =
{
  0xC0, 0xF9, 0xA4, 0xB0, 0x99, 0x92, 0x82,
  0xF8, 0x80, 0x90, 0xff, 0xc6, 0x9c, 0xbf
};

unsigned char data temp_data[2];

unsigned char data display[7] =
{
  0x0b, 0x0c, 0x0d, 0x0d, 0x0d, 0x0d, 0x0d
};

char code reserve[3]_at_ 0x3b; //保留 0x3b 开始的 3 个字节

/*******************************************************

us 延时函数  (8*1.085)*num

 *******************************************************/
void Delay(unsigned int num) //延时函数
{
  while (--num)
    ;
}

/*******************************************************

延时子程序

 *******************************************************/
void delayms(unsigned int ms)
{
  unsigned char k;
  while (ms--)
  {
    for (k = 0; k < 114; k++)
      ;
  }
}

/*******************************************************

蜂鸣器驱动子函数

 *******************************************************/
void beep()
```

```
{
  unsigned char i;
  for (i = 0; i < 180; i++)
  {
    Delay(80);
    BEEP = !BEEP; //BEEP 取反
  }
  BEEP = 1; //关闭蜂鸣器
  delayms(100);
}

/*************************************************************

DS18B20 初始化

presence=0  OK  presence=1  ERROR

 *************************************************************/
unsigned char Init_DS18B20(void)
{
  DQ = 0; //单片机发出低电平复位信号
  Delay(60); //延时>480us
  DQ = 1; //释放数据线
  Delay(8); //延时>64us,等待应答

  presence = DQ; //接收应答信号
  Delay(50); //延时>400us,等待数据线出现高电平
  DQ = 1; //释放数据线

  return (presence); //返回 presence 信号
}

/*************************************************************

读一个字节数据

 *************************************************************/
unsigned char ReadOneChar(void)
{
  unsigned char i = 0;
  unsigned char dat = 0;

  DQ = 1;
  for (i = 0; i < 8; i++)
  //一个字节 8 个 bit
  {
    DQ = 0; //给低脉冲信号
    dat >>= 1;
    DQ = 1; //释放总线
    _nop_();
    _nop_();
    if (DQ)
    //读总线电平状态
      dat |= 0x80;
    //最高位置 1
    Delay(6); //延时>45us
    DQ = 1; //释放总线,表示此次读操作完成
  }

  return (dat); //返回所读得数据
}

/*************************************************************

写一个字节数据

 *************************************************************/
void WriteOneChar(unsigned char dat)
{
  unsigned char i = 0;
  for (i = 0; i < 8; i++)
  //一个字节 8 个 bit
  {
    DQ = 0; //给低脉冲信号
```

```
    Delay(1); //延时<15us
    dat >>= 1; //数据右移一位,最低位移入 CY
    DQ = CY; //写 1bit 数据
    Delay(6); //延时>45us
    DQ = 1; //释放总线,表示此次写操作完成
  }
}

/*************************************************************

温度数据转换子程序

 *************************************************************/
void Temperature_conver()
{
  unsigned char minus = 0;

  //  display[0]=0x0b;                //显示 C
  //  display[1]=0x0c;                //显示 °

  if (temp_data[1] > 127)
  //温度为负值
  {
    temp_data[0] = (~temp_data[0]) + 1; //取反加一，将补码变成原码
    if ((~temp_data[0]) >= 0xff)
      temp_data[1] = (~temp_data[1]) + 1;
    else
      temp_data[1] = ~temp_data[1];
    minus = 1; //温度为负值标志
  }

  display[6] = temp_data[0] &0x0f; //取小数位数据
  display[2] = (display[6] *10) / 16; //保留一位小数

  display[6] = ((temp_data[0] &0xf0) >> 4) | ((temp_data[1] &0x0f) << 4);
    //取整数
  display[5] = display[6] / 100; //百位
  display[4] = (display[6] % 100) / 10; //十位
  display[3] = display[6] % 10; //个位

  if (!display[5])
  //高位为 0,不显示
  {
    display[5] = 0x0a;
    if (!display[4])
    //次高位为 0,不显示
      display[4] = 0x0a;
  }

  if (minus)
  {
    display[5] = 0x0d; //显示负号
  }
}

/*************************************************************

数码管显示子函数

 *************************************************************/
void ledplay()
{
  unsigned char n, shift;

  shift = 0xfe; //位码初值

  for (n = 0; n < 6; n++)
  //6 位数码管显示
  {
    if (n == 3)
      P0 = (LEDData[display[n]]) &0x7f;
    //加小数点显示
    else
      P0 = LEDData[display[n]];
```

```
    //输出段码

    P2 = shift; //输出位码
    shift = (shift << 1) | 0x01; //修改位码
    delayms(1);
  }
  P2 = 0xff; //关闭显示
  delayms(1);
}

/*************************************************************

主函数

 *************************************************************/
void main(void)
{
  unsigned char m;

  P0 = 0xff;
  P2 = 0xff;

  while (1)
  {

    Init_DS18B20();
    if (presence == 0)
    {
      WriteOneChar(0xCC); //跳过 ROM 匹配操作
      WriteOneChar(0x44); //启动温度转换

      for (m = 0; m < 120; m++)
      //数码管初始化显示
        ledplay();
      //等待数据转换完成
    }

    Init_DS18B20();
    if (presence == 0)
    {
      WriteOneChar(0xCC); //跳过 ROM 匹配操作
      WriteOneChar(0xBE); //读取温度寄存器

      temp_data[0] = ReadOneChar(); //温度低 8 位
      temp_data[1] = ReadOneChar(); //温度高 8 位
      Temperature_conver(); //数据转换
      for (m = 0; m < 120; m++)
        ledplay();
      //温度显示
    }
    else
    {
      beep(); //蜂鸣器报警
      P2 = 0xff; //关闭显示
    }
  }
}

 /*************************************************************/
```

实验二十三　红外遥控解码实验

现在大部分家电产品都具有遥控功能，如电视机、影碟机、空调、电风扇及音响等。它们都是采用红外线遥控方式来控制，红外遥控器上的每个按键都具有特定的遥控功能。

ME830 单片机开发实验仪集成有一路一体化红外接收头，并配有定制的红外发射器，能够做红外接收与解码实验。

1. 实验任务

红外一体化接收头接收到红外遥控发射器所发射的信号，并将此信号进行整形和反相送入单片机 P3.2 端口。经过软件译码，将译码结果（按键代码）送数码管显示。

2. 实验线路

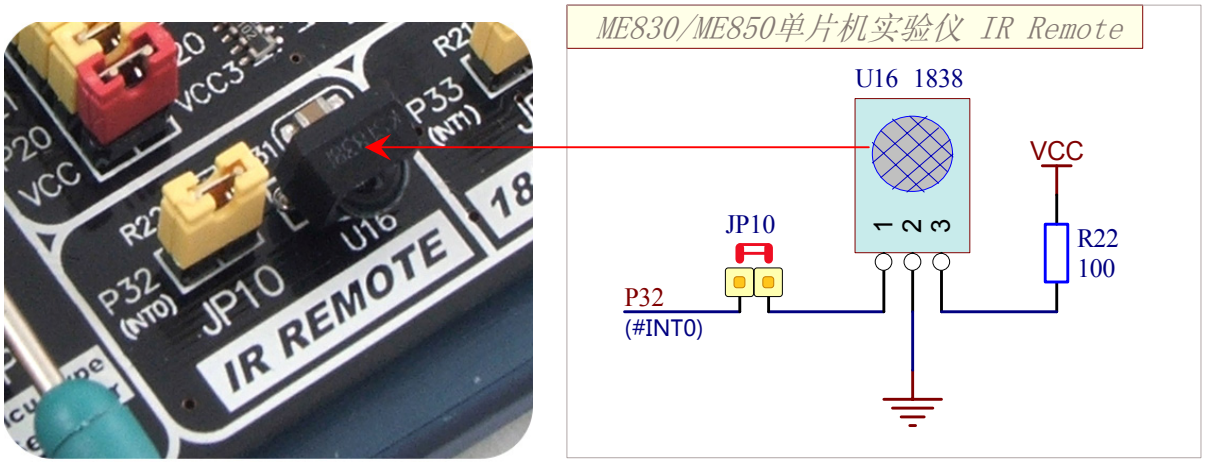


图 6.67　红外接收接口电路

ME830 选用 T1838 一体化红外接收头，接收来自红外遥控器的红外遥控信号。T1838 集成红外接收二极管、放大、解调、整形等电路在同一封装上。T1838 负责红外遥控信号的解调，将调制在 38kHz 上的红外脉冲信号解调并倒相后输入到单片机的 P3.2（INT0）引脚，由单片机进行高电平与低电平宽度的测量（脉冲宽度调制解码）。

T1838 的输出端通过 JP10 与 AT89S52 的 P3.2（INT0）连接，既可以使用中断方式也可以使用查询方式来编程。应用电路如图 6.67 所示。

ME830 配套的红外遥控器采用 DT9122D 芯片制作（兼容 HT6222、SC6122），共有 32 个功能键，在每个按键上标有功能码和此键的数据代码，如图 6.68 所示。当红外遥控器按键按下后，即有规律地将遥控编码发出，所按的键不同，遥控编码也不同。

图 6.68 红外遥控发射器

3. 实验步骤

将 JP21 的 8 个短接子全部用短接帽短接，使 DG0～DG7 与 P2 端口接通。

将 JP22 的 9 个短接子全部用短接帽短接，使 A～DP 与 P0 端口接通，VCC 向数码管模块供电。

将 JP10 的短接子用短接帽短接，使红外接收头 U16 的数据线与 P3.2 端口接通。

第一次使用遥控器要取下电池盖下的隔离胶片！

4. 程序流程图

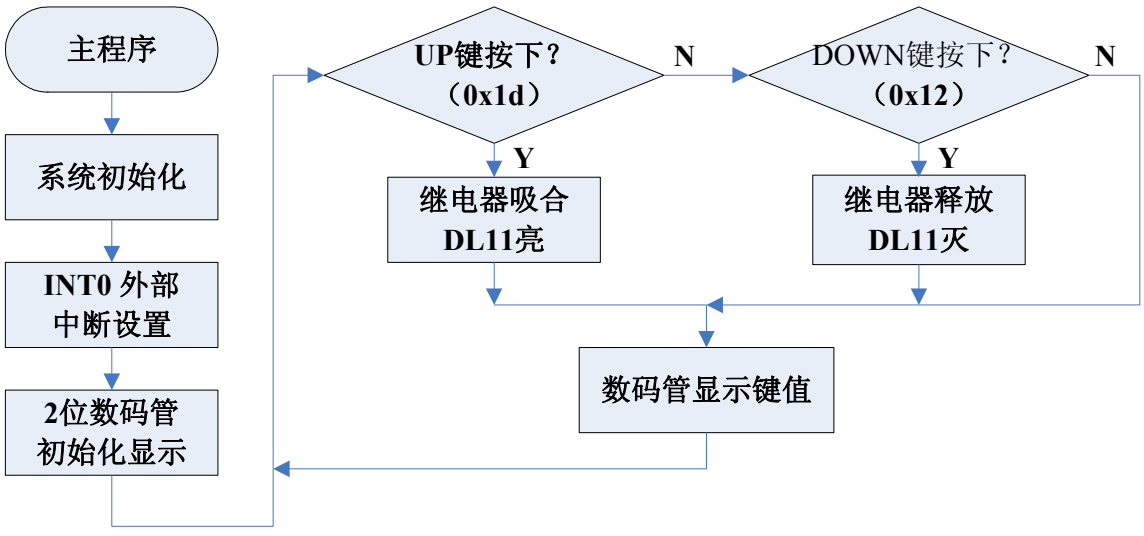


图 6.69 主程序流程图

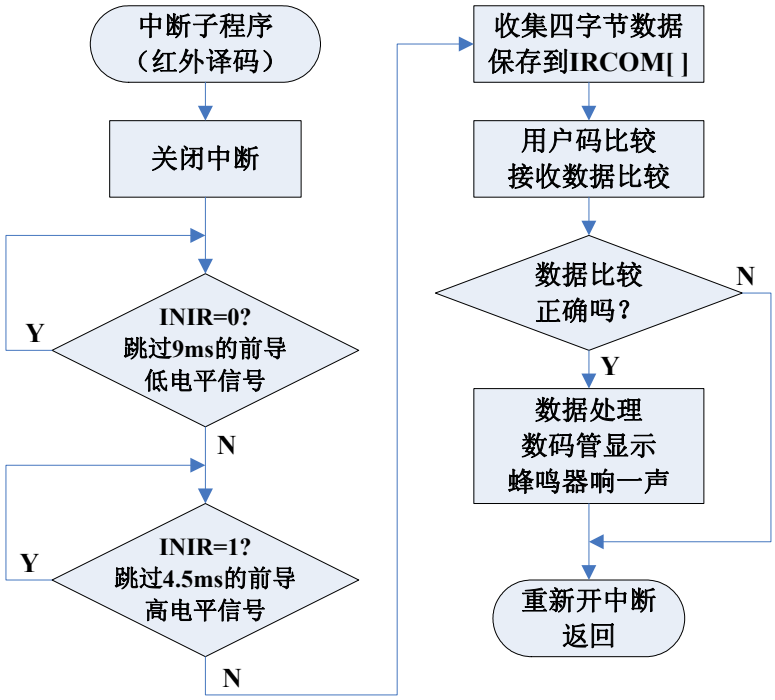


图 6.70　中断子程序流程图

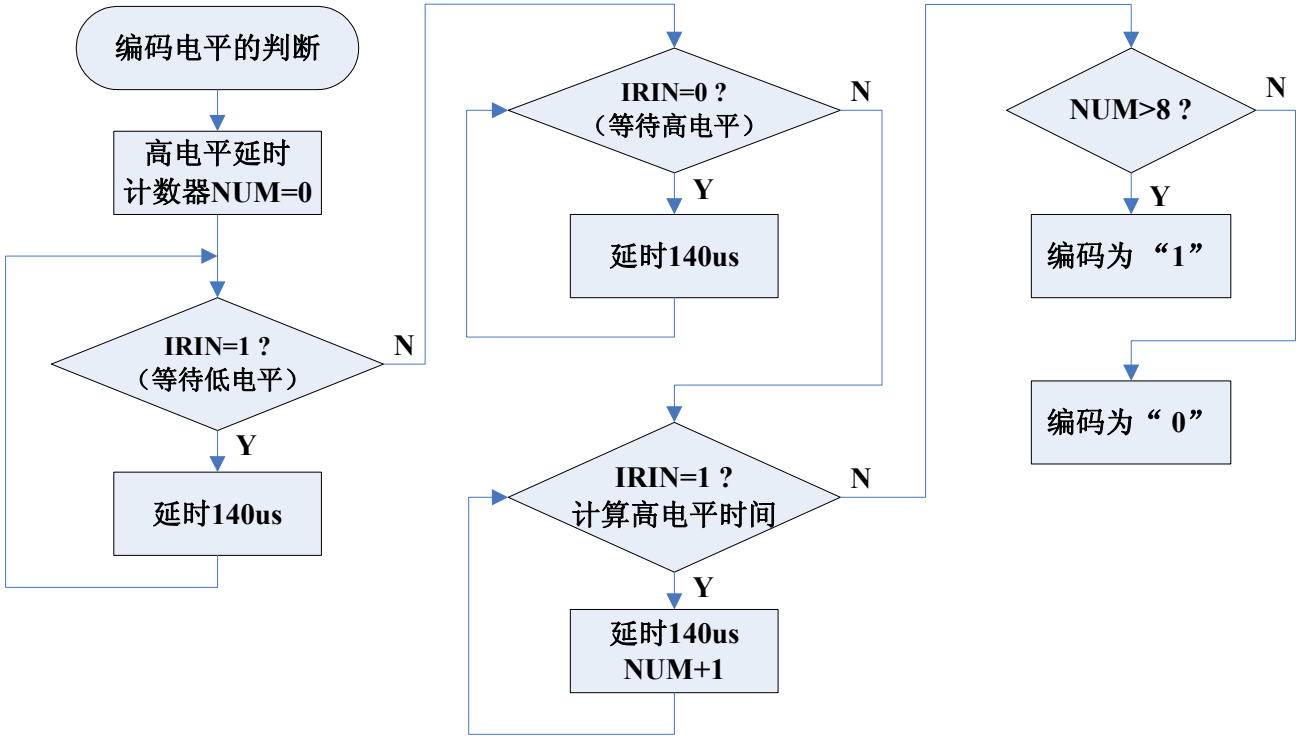


图 6.71　判断编码电平流程图

5. 汇编源程序

（光盘： Example_A51\EX23_IR）

```
/;*****************************************************************************
;* ME830 单片机开发系统演示程序 - 遥控键值解码                                  *
;*                                                                            *
;* 2 位数码管显示                                                             *
;*                                                                            *
;* 版本:  V1.0(  2008-08-20   )                                               *
;* 作者:  gguoqing                                                            *
;* 网站:  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界)           *
;* 邮箱:  willar@tom.com                                                      *
;*                                                                            *
;*【版权】Copyright(C) 深圳硕飞科技有限公司  All Rights Reserved              *
;*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息!                   *
;*****************************************************************************
;*                                                                            *
;*  描述:                                                                     *
;*        ME830  遥控键值读取器                                               *
;*        数码管显示, P0 口为数码管的数据口                                   *
;*                                                                            *
;*        UP 键按下，继电器吸合。DOWN 键按下，继电器关闭。                    *
;*                                                                            *
;*****************************************************************************

      IRCOM  EQU  30H       ;30H-33H IR 使用

      IRIN   EQU  P3.2
      BEEP   EQU  P3.7
      RELAY  EQU  P3.6

;*************************************************************

        ORG 0000H
        AJMP  MAIN
        ORG 0003H          ;外部中断 INT0 入口地址
        AJMP  IR_IN
        ORG  0050H

;*************************************************************
MAIN:
         MOV   SP,#60H
         MOV   A,#00H
         MOV   R0,#IRCOM
LOOP0:
         MOV   @R0,A           ;30H-37H 单元清零
         INC   R0
         CJNE  R0,#IRCOM+8,LOOP0

          MOV  IE,#81H          ;允许总中断中断,使能 INT0 外部中断
         MOV  TCON,#01H        ;触发方式为脉冲负边沿触发

         SETB  IRIN
         SETB  BEEP
         SETB  RELAY
         MOV   IRCOM+5,#10H         ;显示"-"
         MOV   IRCOM+6,#10H
         ACALL   IR_SHOW

LOOP1:
         ACALL   IR_SHOW
         MOV  A,IRCOM+2
         CJNE   A,#1DH,LOOP2     ;UP 键按下
         CLR    RELAY            ;继电器吸合
LOOP2:
         CJNE   A,#12H,LOOP3     ;DOWN 键按下
         SETB   RELAY            ;继电器关闭
LOOP3:
         AJMP    LOOP1

;*************************************************************

;键值显示
```

```
;*********************************************************
IR_SHOW:
          MOV    A,IRCOM+5            ;取显示数据到 A
          MOV    DPTR,#TAB            ;取段码表地址
          MOVC   A,@A+DPTR           ;查显示数据对应段码
          MOV    P0,A                 ;段码放入 P0 口
          CLR    P2.0
          SETB   P2.1
          ACALL   DELAY1
          MOV    A,IRCOM+6            ;取显示数据到 A
          MOV    DPTR,#TAB            ;取段码表地址
          MOVC   A,@A+DPTR           ;查显示数据对应段码
          MOV    P0,A                 ;段码放入 P0 口
          CLR    P2.1
          SETB   P2.0
          ACALL   DELAY1
          MOV   P2,#0FFH
          RET

;*********************************************************
TAB:
 DB   0C0H,0F9H,0A4H,0B0H,99H,92H,82H,0F8H
 DB   80H,90H,88h,83h,0c6h,0a1h,86h,8eh,0bfh  ;0－F,-

;*********************************************************

; IR 译码子程序
; R0---存放 IR 数据
; R2---高电平宽度的计数值
; R3--- 一字节需接收 8 位计数

;*********************************************************
IR_IN:
          CLR EA                      ;暂时关闭 CPU 的所有中断请求
          PUSH   ACC
          PUSH   PSW
          SETB   PSW.3                ;选择工作寄存器组 1
          CLR    PSW.4
          MOV   R0,#IRCOM
          MOV   R4,#30
IR_IN1:
          ACALL   DELAY              ;延时 4ms，去干扰再确认 IR 信号是否出现
          DJNZ   R4,IR_IN1
          JB   IRIN,IR_END           ;如果 IRIN＝1 没有 IR 信号出现，则退出
WAIT_H:
          JB   IRIN,WAIT_L           ;等 IR 变为高电平避开 9ms 低电平引导脉冲
          ACALL   DELAY
          AJMP    WAIT_H
WAIT_L:
          JNB   IRIN,WAIT_H1         ;等 IR 变为低电平避开 4.5ms 高电平引导脉冲
          ACALL   DELAY
          AJMP   WAIT_L
WAIT_H1:
          MOV   R3,#0                   ;8 位数清为 0
WAIT_L1:
          JNB   IRIN,WAIT_H2            ;等 IR 变为低电平
          ACALL   DELAY
          AJMP   WAIT_L1
WAIT_H2:
          JB   IRIN,IR_COUN             ;等 IR 变为高电平
          ACALL   DELAY
          AJMP   WAIT_H2
IR_COUN:
          MOV   R2,#0                   ;对高电平进行 0.14ms 计数
IR_COUN1:
          ACALL   DELAY
          JB   IRIN,  IR_COUN2          ;等 IR 变为高电平
                                        ;IR=0，检查 R2 中的计数值
          MOV   A,#8
          CLR   C                       ;清借位标志
          SUBB   A,R2                   ;判断高低位
                                        ;若 C=0，则解码为“0″
                                        ;若 C=1，则解码为“1″
          MOV   A,@R0                   ;取出原先的数据
```

```
            RRC   A                         ;将借位标志 C 移入 A
            MOV   @R0,A                     ;处理完一位,将数据写入
            INC   R3                        ;接收字节计数加 1
            CJNE   R3,#8,WAIT_L1             ;需处理完 8 位
            MOV   R3,#0
            INC   R0                        ;存放 IR 数据值加 1
            CJNE   R0,#IRCOM+4,WAIT_L1      ;收集到 4 字节了
            AJMP   IR_COMP
IR_COUN2:
            INC   R2
            CJNE   R2,#15,IR_COUN1          ;0.14ms 计数过长，则自动离开
IR_END:
            POP   PSW
            POP   ACC
            SETB   EA
            RETI

IR_COMP:
            MOV A,IRCOM+3
            CPL A                           ;比较所接收的数据
            CJNE A,IRCOM+2,COMP_END         ;如果不等表示接收数据错误,放弃。

            MOV A,IRCOM                     ;效验用户码是否为 "00"
            JNZ   COMP_END

            MOV   A,IRCOM+2                 ;键码数据处理
            ANL   A,#0FH
            MOV   IRCOM+5,A                 ;送个位显示单元
            MOV   A,IRCOM+2
            ANL   A,#0F0H
            SWAP   A
            MOV   IRCOM+6,A                 ;送十位显示单元
            ACALL   IR_SHOW                 ;显示键值
            ACALL   BEEP_BL                 ;蜂鸣器鸣响表示解码成功

COMP_END:
            AJMP   IR_END

;*************************************************************

; 0.14MS 延时子程序
; IR 解码使用

;*************************************************************
DELAY:
            MOV   R6,#2
DEL1:
            MOV   R7,#32
DEL2:
            DJNZ   R7,DEL2
            DJNZ   R6,DEL1
            RET

;*************************************************************

;蜂鸣器驱动子程序

;*************************************************************
BEEP_BL:
            MOV   R6,#100
BL1:
            CALL   DEX1
            CPL   BEEP
            DJNZ   R6,BL1
            RET
DEX1:
            MOV   R7,#220
DEX2:
            NOP
            DJNZ   R7,DEX2
            RET

;*************************************************************
DELAY1:                         ;数码管延时 4MS
```

```
        MOV   R6,#20
  DL2:
        MOV   R7,#100
        DJNZ  R7,$
        DJNZ  R6,DL2
        RET

;*********************************************************

         END              ;结束

;*********************************************************

; IRCOM[0] --- 存放用户码      00H
; IRCOM[1] --- 存放用户反码    FFH
; IRCOM[2] --- 存放数据码
; IRCOM[3] --- 存放数据反码

;================================

;DT9122D 遥控器（伟纳电子）

;******  红外遥控器键值表  ******

;  10      03      01      06
;  09      1D      1F      0D
;  19      1B      11      15
;  17      12      16      4C
;  40      48      04      00
;  02      05      54      4D
;  0A      1E      0E      1A
;  1C      14      0F      0C

;================================
```

6. C 语言源程序

（光盘： Example_C51\EX23_IR）

```
/*********************************************************************
 *                                                                   *
 * ME830 单片机开发实验仪演示程序 - 红外遥控器键值显示               *
 *                                                                   *
 * 2 位数码管显示                                                    *
 *                                                                   *
 * 版本:  V1.0 (2008/08/20)                                          *
 * 作者:  gguoqing (Email: gguoqing@willar.com)                      *
 * 网站:  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界) *
 * 邮箱:  willar@tom.com                                             *
 *                                                                   *
 *【版权】Copyright(C)伟纳电子 www.willar.com  All Rights Reserved  *
 *【声明】此程序仅用于学习与参考，引用请注明版权和作者信息!         *
 *                                                                   *
 *********************************************************************/

#include <reg52.h>
#include <intrins.h>

sbit IRIN = P3 ^ 2; //红外接收器数据线
sbit BEEP = P3 ^ 7; //蜂鸣器驱动线
sbit RELAY = P3 ^ 6; //继电器驱动线

unsigned char IRCOM[] =
{
  0x00, 0x00, 0x00, 0x00, 0x10, 0x10
};

unsigned char code table[] =
{
  0xc0, 0xf9, 0xa4, 0xb0, 0x99, 0x92, 0x82, 0xf8,   //0,1,2,3,4,5,6,7
  0x80, 0x90, 0x88, 0x83, 0xc6, 0xa1, 0x86, 0x8e, 0xbf //8,9,A,B,C,D,E,F,-
};
```

```
char code reserve[3]_at_ 0x3b; //保留 0x3b 开始的 3 个字节

/*************************************************************

ms 延时子函数

 *************************************************************/
void delayms(unsigned int ms)
{
  unsigned char k;
  while (ms--)
  {
    for (k = 0; k < 114; k++)
      ;
  }
}

/*************************************************************

us 延时子函数

 *************************************************************/
void delay(unsigned char x) //x*0.14MS
{
  unsigned char i;
  while (x--)
  {
    for (i = 0; i < 14; i++)
      ;
  }
}

/*************************************************************

蜂鸣器驱动子函数

 *************************************************************/
void beep()
{
  unsigned char i;
  for (i = 0; i < 100; i++)
  {
    BEEP = !BEEP; //BEEP 取反
    delay(6);
  }
  BEEP = 1; //关闭蜂鸣器
}

/*************************************************************

显示函数

 *************************************************************/
void play()
{
  P0 = (table[IRCOM[4]]); //个位
  P2 = 0xfe;
  delayms(1);
  P0 = (table[IRCOM[5]]); //十位
  P2 = 0xfd;
  delayms(1);
  P2 = 0xff; //关闭显示
  delayms(1);
}

/*************************************************************

主函数

 *************************************************************/
void main(void)
{
  P0 = 0xff; //I/O 口初始化
  P2 = 0xff;
```

```
    IRIN = 1;
    BEEP = 1;
    RELAY = 1;

    IE = 0x81; //允许总中断中断,使能 INT0 外部中断
    TCON = 0x01; //触发方式为脉冲负边沿触发

    play();

    while (1)
    {
      if (IRCOM[2] == 0x1d)
      //UP 键
        RELAY = 0;
      //继电器吸合
      if (IRCOM[2] == 0x12)
      //DOWN 键
        RELAY = 1;
      //继电器关闭

      play(); //显示
    }
}

/************************************************************

INT0 中断服务子函数

 ************************************************************/
void IR_IN()interrupt 0
{
  unsigned char j, k, Num = 0;

  EX0 = 0; //关闭 INT0 中断
  delay(15); //延时
  if (IRIN == 1)
  //再确认 IR 信号是否出现
  {
    EX0 = 1; //开 INT0 中断
    return ; //退出
  }

  while (!IRIN)
  //等 IR 变为高电平，跳过 9ms 的前导低电平信号。
  {
    delay(1);
  }

  while (IRIN)
  //等 IR 变为低电平，跳过 4.5ms 的前导高电平信号。
  {
    delay(1);
  }

  for (j = 0; j < 4; j++)
  //收集四组数据
  {
    for (k = 0; k < 8; k++)
    //每组数据有 8 位
    {
      while (IRIN)
      //等 IR 变为低电平
      {
        delay(1);
      }
      while (!IRIN)
      //等 IR 变为高电平
      {
        delay(1);
      }
      while (IRIN)
      //计算 IR 高电平时长
      {
        delay(1);
```

```
            Num++;
            if (Num >= 15)
            {
              EX0 = 1; //0.14ms 计数过长自动离开。
              return ;
            }
          } //高电平计数完毕
          IRCOM[j] = IRCOM[j] >> 1; //数据最高位补“0”
          if (Num >= 8)
            IRCOM[j] = IRCOM[j] | 0x80;
          //数据最高位补“1”
          Num = 0;
        } //end for k
      } //end for j

      if (IRCOM[0] != 0x00)
      //比较用户码
      {
        EX0 = 1; //开 INT0 中断
        return ; //退出
      }
      if (IRCOM[2] != ~IRCOM[3])
      //接收数据是否正确
      {
        EX0 = 1; //开 INT0 中断
        return ; //退出
      }

      IRCOM[4] = IRCOM[2] &0x0F; //取键码的低四位
      IRCOM[5] = IRCOM[2] >> 4; //右移 4 次，高四位变为低四位

      play(); //显示键码
      beep(); //蜂鸣器响一声
      EX0 = 1; //重新开 INT0 中断
    }

    /********************************************************
    ;================================

    ;DT9122D 遥控器（伟纳电子）

    ;******  红外遥控器键值表  ******

    ;  10     03      01      06
    ;  09     1D      1F      0D
    ;  19     1B      11      15
    ;  17     12      16      4C
    ;  40     48      04      00
    ;  02     05      54      4D
    ;  0A     1E      0E      1A
    ;  1C     14      0F      0C

    ;================================

    IRCOM[0] --- 存放用户码    00H
    IRCOM[1] --- 存放用户反码  ffH
    IRCOM[2] --- 存放数据码
    IRCOM[3] --- 存放数据反码

     ********************************************************/
```

实验二十四　PS2 键盘解码实验

1. 实验任务

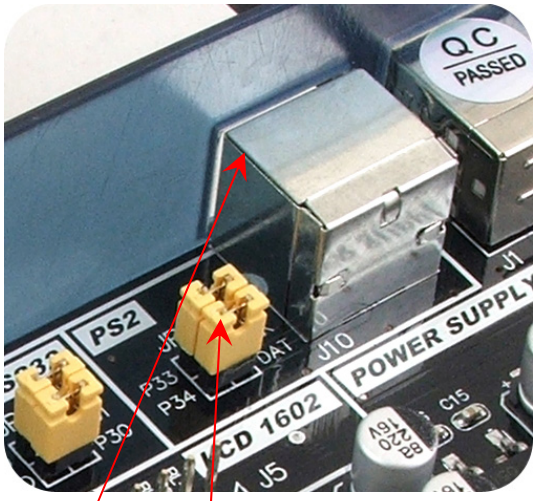
开机上电时，首先程序向连接在 PS2 接口的 PC 键盘发出复位指令，PC 键盘被复位，你可以看到 PC 键盘上的 3 个 LED 灯闪亮一下，1 位数码管初始显示“－”。

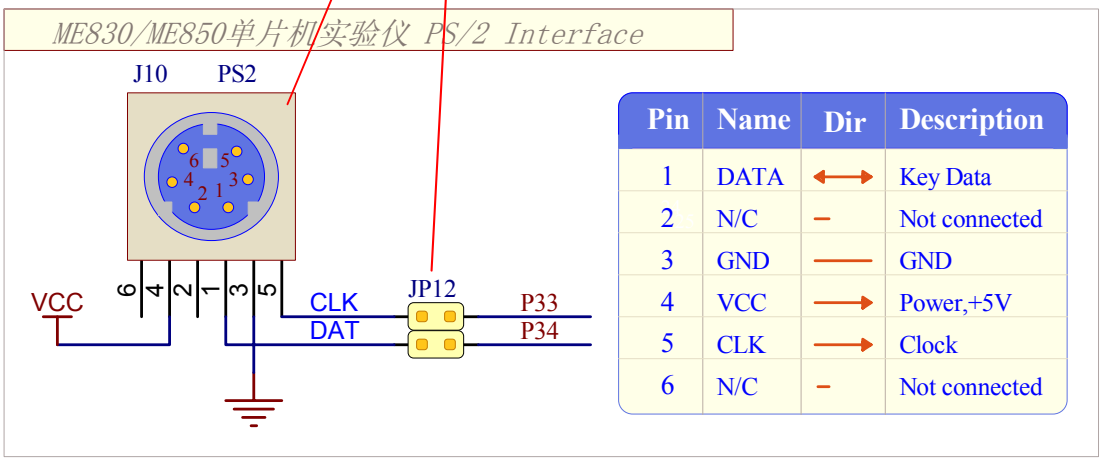
程序定义了 PC 键盘的 1-f 数字键、Esc、NumLock 功能键和右边小键盘中的 0-9 数字键。

右边小键盘中的 0-9 数字键是否有效由 NumLock 功能键控制，按下 NumLock 功能键，NumLock 灯亮，右边小键盘中的 0-9 数字键有效；按下 NumLock 功能键，NumLock 灯灭，右边小键盘中的 0-9 数字键失效。按下 Esc 键，向 PC 键盘发送复位命令。

按下被定义而且有效的 PC 键盘的按键时，1 位数码管显示其键值，蜂鸣器响一声。
按下 NumLock 功能键，数码管显示“n”，蜂鸣器响一声。
按下没有被定义或被定义但无效的 PC 键盘的按键时数码管显示“－”。

2. 实验线路





Pin	Name	Dir	Description
1	DATA	⟷	Key Data
2	N/C	–	Not connected
3	GND	—	GND
4	VCC	→	Power,+5V
5	CLK	→	Clock
6	N/C	–	Not connected

图 6.72　PS2 键盘接口电路

3. 实验步骤

将 JP21 的 8 个短接子全部用短接帽短接，使 DG0～DG7 与 P2 端口接通。
将 JP22 的 9 个短接子全部用短接帽短接，使 A～DP 与 P0 端口接通，VCC 向数码管模块供电。
将 JP12 的短接子用短接帽短接，使 PS/2 接口线与 P3.3 和 P3.4 端口接通。
将 PC 键盘插入 J10，使 PS/2 接口使能。
将 JP11 的短接子（DS18B20）上的短接帽取掉，以免产生干扰。

4. 程序流程图

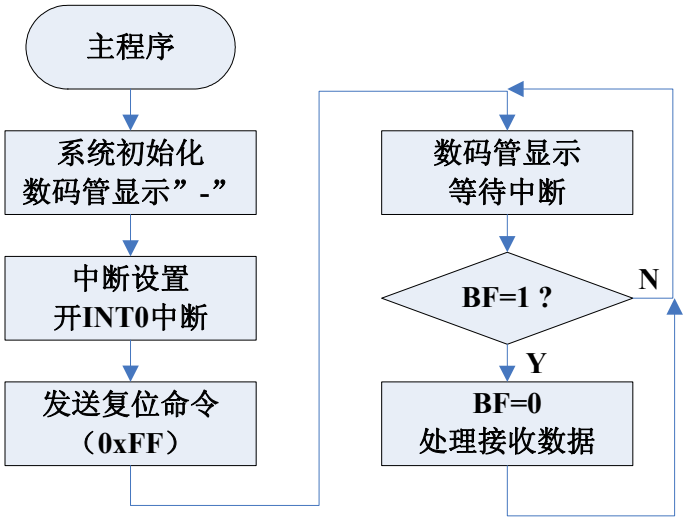


图 6.73　主程序流程图

数据接收编程说明：

PS2 协议采用的传送数据帧由 1 位起始位（0）、8 位数据位、1 位奇偶校验位和 1 位停止位（1）构成。数据发送时低位在前，高位在后。

当 IntNum＝0 时，　起始位（起始位始终为“0”）。
当 IntNum＝9 时，　奇偶校验位
当 IntNum＝10 时，停止位（停止位始终为“1”）

因上述 3 个数据位在数据接收时不考虑进行相应的处理，所以将其丢弃。

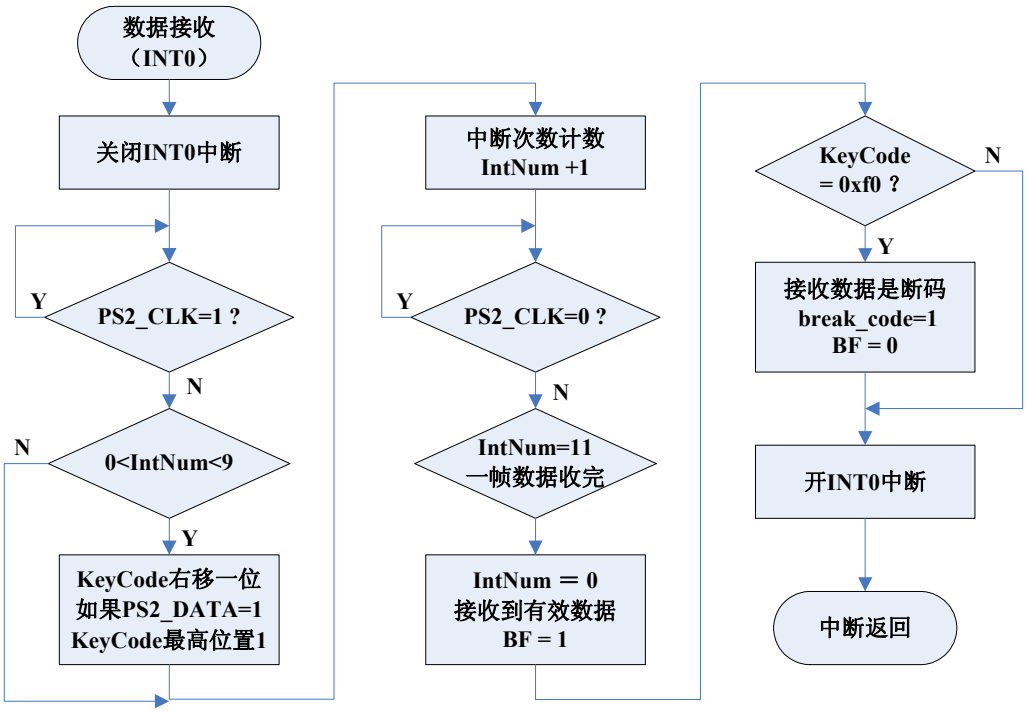


图 6.74　接收数据流程图

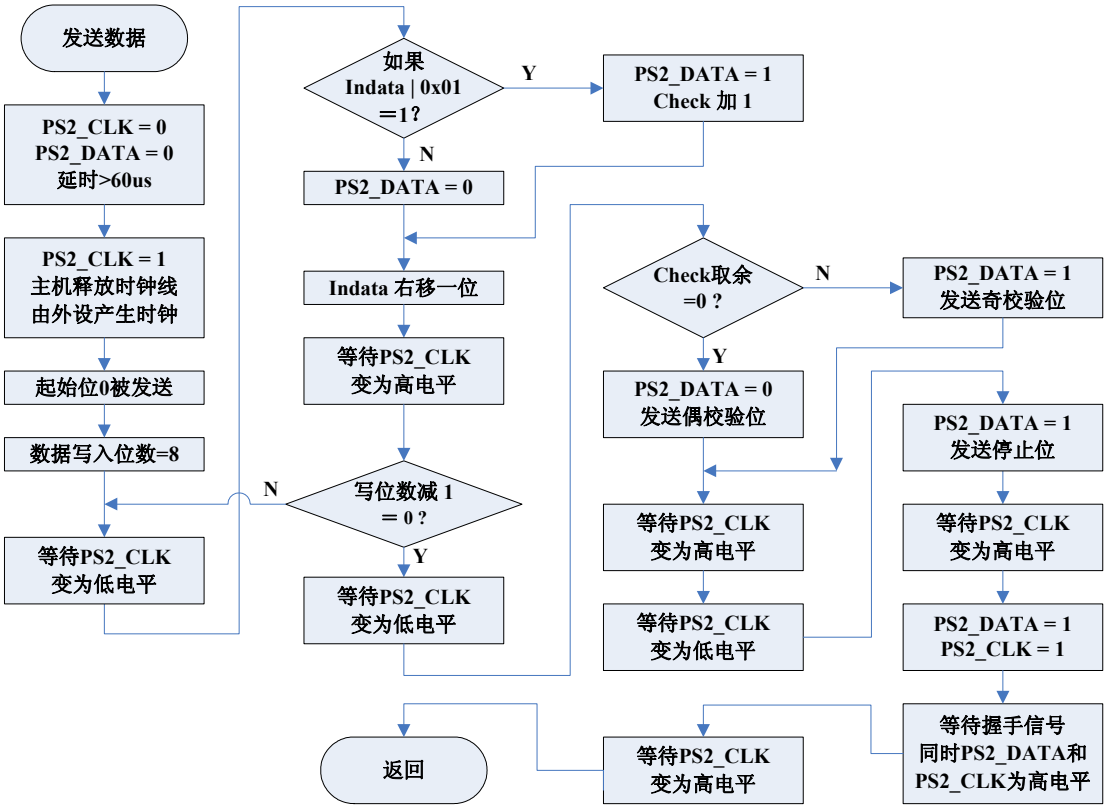


图 6.75　发送数据流程图

5. 汇编源程序

（光盘： Example_A51\EX24_PS2）

```
;*********************************************************************
;*                                                                   *
;* ME830 单片机开发系统演示程序 - PS2 键值显示                        *
;*                                                                   *
;* 1 位数码管显示键码                                                 *
;*                                                                   *
;* 版本:  V1.0 (2008/09/23)                                          *
;* 作者:  gguoqing (Email: gguoqing@willar.com)                      *
;* 网站:  www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界)*
;* 邮箱:  willar@tom.com                                             *
;*                                                                   *
;*【版权】Copyright(C) 深圳硕飞科技有限公司  All Rights Reserved    *
;*【声明】此程序仅用于学习与参考，引用请注明版权和作者信息!         *
;*                                                                   *
;*********************************************************************

          PS2_CLK   EQU  P3.3
          PS2_DATA  EQU  P3.4

          KEY_DATA  EQU  30H
          ESC_DATA  EQU  31H
          NUM_DATA  EQU  32H
          S_DATA    EQU  33H
          TEMP      EQU  34H

          FLAG      BIT  20H.0
          NumLock   BIT  20H.1
          SEND_OK   BIT  20H.2
          BREAK_C   BIT  20H.4    ;断码标志
          BEEP      BIT  P3.7

;*******************************************************

           ORG  0000H
```

```
            JMP   MAIN
            ORG   0013H
            JMP   EXT1
            ORG   0050H

;*********************************************************
MAIN:
            MOV   SP,#60H
            MOV   P0,#0FFH
            MOV   P2,#0FEH
            MOV   P3,#0FFH

            CLR   IT1             ;外部中断 1 为低电平触发
            SETB  EA              ;开总中断
            SETB  EX1             ;开外部中断 1

            MOV   R0,#00H
            MOV   R1,#00H
            MOV   R3,#00H
            CLR   NumLock
            SETB  PS2_DATA
            SETB  PS2_CLK

            MOV   S_DATA,#0FFH      ;PS2 键盘复位命令
            ACALL   SEND_DATA
            ACALL   DELAY
            MOV   ESC_DATA,#00H
            MOV   NUM_DATA,#00H
            MOV   P0,#0BFH            ;显示'-'

DISP:
            MOV   P0,A

            MOV   R0,NUM_DATA
            CJNE  R0,#77H,NUM2        ;NUM 键码
            MOV   NUM_DATA,#00H
            CPL   NumLock

            MOV   S_DATA,#0EDH        ;PS2 键盘命令
            ACALL   SEND_DATA

            JNB   NumLock,NUM1
            MOV   S_DATA,#02H         ;点亮 NumLock
            ACALL   SEND_DATA
              MOV   P0,#0C8H            ;显示"n"
            ACALL  BEEP_BL
              MOV   A,#0C8H             ;显示"n"
            AJMP   DISP
NUM1:
            MOV   S_DATA,#00H         ;关闭 NumLock
            ACALL   SEND_DATA
            MOV   P0,#0C8H            ;显示"n"
            ACALL  BEEP_BL
              MOV   A,#0C8H             ;显示"n"
            AJMP   DISP
NUM2:
            MOV   R0,ESC_DATA
            CJNE  R0,#76H,DISP_END    ;ESC 键码
            MOV   ESC_DATA,#00H
            MOV   P0,#0BFH            ;显示'-'
              MOV   S_DATA,#0FFH        ;PS2 键盘复位命令
            ACALL   SEND_DATA
              ACALL  BEEP_BL
            MOV   A,#0BFH             ;显示'-'
DISP_END:
            AJMP   DISP
;---------------------------------------------------------

;根据 PS2 的键值来查找代码，并取得顺序码。
;再根据顺序码来查找显示码

;---------------------------------------------------------
PS2KEY_IN:
            MOV   B,A
```

```
            MOV   DPTR,#TABLE_D
            MOV   R3,#0FFH
KEY_IN1:
            INC   R3
            MOV   A,R3
            MOVC   A,@A+DPTR
            CJNE   A,B,KEY_IN3
            MOV   A,R3                  ;找到，取顺序码
            CLR   C
            SUBB   A,#10H               ;不够减，C=1
            JC   KEY_IN2
            JNB   NumLock,KEY_END
KEY_IN2:
            MOV   A,R3
            MOV   DPTR,#TAB_NU          ;根据顺序码来查找显示码
            MOVC   A,@A+DPTR
             MOV   P0,A                  ;显示键值
            ACALL   BEEP_BL             ;转换成功，蜂鸣器响一声
            RET
KEY_IN3:
            CJNE   A,#0FFH,KEY_IN1      ;未完，继续查
KEY_END:
            CLR   BREAK_C
            MOV   A,#0BFH               ;没有定义的键显示'-'
            RET

;************************************************************

;外部中断子程序      (接收数据)

;************************************************************
EXT1:
            CLR   EX1
            CJNE   R1,#00H,IN_LOOP      ;跳过第一位启动位
            AJMP   IN_LOOP3
IN_LOOP:
            CJNE   R1,#09H,IN_LOOP1     ;2-9 位为数据
IN_LOOP1:
            JNC   IN_LOOP3              ;大于或等于 9，转。
            RR   A                      ;
            JB   PS2_DATA,IN_LOOP2      ;判数据是“1”，还是“0”
            ANL   A,#7FH
            AJMP   IN_LOOP3
IN_LOOP2:
            ORL   A,#80H
IN_LOOP3:
            INC   R1                    ;中断计数
            JNB   PS2_CLK,$             ;等待 PS2_CLK 变高
IN_LOOP4:
            CJNE   R1,#0BH,IN_LOOP5     ;一桢数据是否读完?
IN_LOOP5:
            JNC   IN_LOOP6              ;大于或等于 11，转。
            AJMP   EXT1_END
IN_LOOP6:
            CJNE   A,#0F0H,IN_LOOP7     ;断码是否开始
            SETB   BREAK_C              ;置断码标志
            MOV   R1,#00H                ;中断计数清零
            JMP   EXT1_END
IN_LOOP7:
            CJNE   A,#77H,IN_LOOP8      ;NUMLOCK
            JB   BREAK_C,IN_LOOP7A
            MOV   NUM_DATA,A
IN_LOOP7A:
            CLR   BREAK_C
            MOV   R1,#00H               ;中断计数清零
            AJMP   EXT1_END
IN_LOOP8:
            CJNE   A,#76H,IN_LOOP9      ;ESC
            JB   BREAK_C,IN_LOOP8A
            MOV   ESC_DATA,A
IN_LOOP8A:
            CLR   BREAK_C
            MOV   R1,#00H                ;中断计数清零
            AJMP   EXT1_END
```

```
IN_LOOP9:
            MOV   R1,#00H               ;中断计数清零
            ACALL   PS2KEY_IN
EXT1_END:
            SETB   EX1
            RETI

;*********************************************************

; 发送的数据子程序

;*********************************************************
SEND_DATA:
            CLR   EA
            MOV   A,S_DATA          ;要发送的数据入 A
            JB PSW.0,SET1
            SETB   FLAG             ;P=0 偶数个 1,FLAG=1
            AJMP    SEND_DATA0
SET1:
            CLR   FLAG              ;P=1 奇数个 1,FLAG=0
SEND_DATA0:
            CLR   PS2_CLK           ;请求发送数据
            SETB   PS2_DATA
            ACALL   DELAY120US      ;拉低 PS2_CLK 线 120us

            CLR   PS2_DATA          ;起始位（低电平）
            SETB   PS2_CLK          ;释放时钟线

            MOV   R1,#08H
SEND_DATA1:
            JB   PS2_CLK,$          ;等待 PS2 拉低时钟
            RRC A
            MOV PS2_DATA,C
            JNB   PS2_CLK,$          ;等待 PS2 拉高时钟
            DJNZ R1,SEND_DATA1      ;循环 8 次送 8 个 bit

            JB   PS2_CLK,$           ;等待 PS2 拉低时钟

            JNB   FLAG,SEND_DATA2
            SETB PS2_DATA           ;发送 1 效验位
            AJMP   SEND_DATA3
SEND_DATA2:
            CLR   PS2_DATA          ;发送 0 效验位
SEND_DATA3:
            JNB   PS2_CLK,$          ;等待 PS2 拉高时钟
            JB   PS2_CLK,$           ;等待 PS2 拉低时钟
            SETB PS2_DATA           ;发送结束位
            JNB   PS2_CLK,$          ;等待 PS2 拉高时钟

            SETB PS2_DATA
            SETB PS2_CLK
            JB   PS2_CLK,$           ;等待 PS2 拉低时钟
            MOV   C,PS2_DATA         ;读 ACK 信号,低电平表示成功
            JNB   PS2_CLK,$          ;等待 PS2 拉高时钟
            JNB   PS2_DATA,$

            JC   SEND_ERROR
            SETB   SEND_OK          ;数据发送成功
            AJMP SEND_DATA_END
SEND_ERROR:
            CLR   SEND_OK           ;数据发送错误
SEND_DATA_END:
            SETB   EA
            RET

;*********************************************************

; 延时子程序

;*********************************************************
DELAY120US:
            MOV   R7,#60
            DJNZ   R7,$
            RET
```

```
;*************************************************************

;蜂鸣器驱动子程序

;*************************************************************
BEEP_BL:
            MOV   R6,#200
  BL1:
            ACALL   DEX1
            CPL   BEEP
            DJNZ   R6,BL1
            ACALL  DELAY
            RET

;*************************************************************

; 680US 延时子程序

;*************************************************************
DEX1:
            MOV   R7,#210
DEX2:
            NOP
            DJNZ   R7,DEX2
            RET

;*************************************************************

; 100ms 延时子程序

;*************************************************************
DELAY:
            MOV   R6,#200
DEL1:
            MOV   R7,#230
            DJNZ   R7,$
            DJNZ   R6,DEL1
            RET

;*************************************************************

;键值表

;*************************************************************
TABLE_D:

DB 45H,16H,1EH,26H,25H,2EH,36H,3DH,3EH,46H   ;0-9
DB 1CH,32H,21H,23H,24H,2BH                   ;a-f
DB 70H,69H,72H,7AH,6BH,73H,74H,6CH,75H,7DH   ;右边数字键 0-9
DB 0FFH

;*************************************************************



;*************************************************************
TAB_NU:
    DB 0C0H,0F9H,0A4H,0B0H,099H,092H,082H,0F8H
    DB 080H,090H,088H,083H,0C6H,0A1H,086H,08EH

    DB 0C0H,0F9H,0A4H,0B0H,099H,092H,082H,0F8H
    DB 080H,090H,088H,083H,0C6H,0A1H,086H,08EH

    DB  0FFH

;*************************************************************

       END              ;结束

;*************************************************************
```

6. C 语言源程序

（光盘： Example_C51\EX24_PS2）

```
/*********************************************************************
 *                                                                   *
 * ME830 单片机开发实验仪演示程序 - PS2 键值显示                      *
 *                                                                   *
 * 1 位数码管显示                                                    *
 *                                                                   *
 * 版本： V1.0 (2008/09/15)                                          *
 * 作者： gguoqing (Email: gguoqing@willar.com)                      *
 * 网站： www.sofi-tech.com(硕飞科技) www.mcusj.com(伟纳单片机世界) *
 * 邮箱： willar@tom.com                                             *
 *                                                                   *
 *【版权】Copyright(C)伟纳电子 www.willar.com  All Rights Reserved  *
 *【声明】此程序仅用于学习与参考，引用请注明版权和作者信息！       *
 *                                                                   *
 *********************************************************************/

#include <reg52.h>
#include <intrins.h>

sbit PS2_DATA = P3 ^ 4; //PS2 数据线
sbit PS2_CLK = P3 ^ 3; //PS2 时钟线
sbit BEEP = P3 ^ 7; //蜂鸣器驱动线

bit numlock = 1; //NumLock 标志
bit break_code = 1; //断码标志
bit BF = 0; //标识是否有字符被收到

#define NUMLOCK_CODE    0x77   //NumLock 键码
#define ESC_CODE        0x76   //ESC 键码

unsigned char TEMP, TEMP1, y;

unsigned char KeyCode; //键盘键值码

unsigned char code TABLE_D[] =
{
  //键值码

  0x45, 0x16, 0x1e, 0x26, 0x25, 0x2e, 0x36, 0x3d, 0x3e, 0x46,  //0-9
  0x1c, 0x32, 0x21, 0x23, 0x24, 0x2b,  //a-f
  0x70, 0x69, 0x72, 0x7a, 0x6b, 0x73, 0x74, 0x6c, 0x75, 0x7d, 0xff
}; //右边数字键 0-9

unsigned char code TABLE_NUM[] =
{
  //显示码

  0xC0, 0xF9, 0xA4, 0xB0, 0x99, 0x92, 0x82, 0xF8, 0x80, 0x90, 0x88, 0x83, 0xC6,
    0xA1, 0x86, 0x8E, 0xC0, 0xF9, 0xA4, 0xB0, 0x99, 0x92, 0x82, 0xF8, 0x80,
    0x90, 0xff
};

unsigned char IntNum = 0; //中断次数计数

char code reserve[3]_at_ 0x3b; //保留 0x3b 开始的 3 个字节

/*******************************************************

x*0.14ms 延时函数

 *******************************************************/
void delay(unsigned int x)
{
  unsigned char i;
  while (x--)
  {
    for (i = 0; i < 14; i++)
      ;
  }
}
```

```
/*************************************************************

延时子程序

 *************************************************************/
void delayms(unsigned int ms)
{
  unsigned char k;
  while (ms--)
  {
    for (k = 0; k < 114; k++)
      ;
  }
}

/*************************************************************

蜂鸣器驱动子函数

 *************************************************************/
void beep()
{
  unsigned char i;
  for (i = 0; i < 200; i++)
  {
    delay(6);
    BEEP = !BEEP; //BEEP 取反
  }
  BEEP = 1; //关闭蜂鸣器
  delayms(100); //延时 100ms
}

/*************************************************************

函数:SEND_DATA()
功能:向键盘发送命令
输入:命令

 *************************************************************/
void SEND_DATA(unsigned char orderByte)
{
  unsigned char k;
  unsigned char check = 0;

  EA = 0; //关闭总中断,发送命令到键盘

  PS2_CLK = 0; //首先拉低时钟
  PS2_DATA = 0; //拉低数据线
  for (k = 0xff; k != 0; k--)
    ;
  //延时，抑制键盘发送

  PS2_DATA = 0; //拉低数据，发送起始位
  PS2_CLK = 1; //置高时钟，释放此线
  for (k = 8; k != 0; k--)
  // 发送八位数据，循环八次
  {
    while (PS2_CLK)
      ;
    //等待 PS2 拉低时钟
    if (orderByte &0x01)
    {
      PS2_DATA = 1; // 根据低位设定输出数据
      check++; // 如果输出一个 1，效验记录数据加一
    }
    else
      PS2_DATA = 0;

    orderByte >>= 1; // 命令字调整
    while (!PS2_CLK)
      ;
    // 输出脉冲
  }
  while (PS2_CLK)
```

```
    ;
  //等待 PS2 拉低时钟
  if (check % 2)
  {
    // 如果数据输出过偶数个 1
    PS2_DATA = 0; // 效验数据位置 0
  }
  else
  {
    PS2_DATA = 1; // 否则数据位置 1
  }
  while (!PS2_CLK)
    ;
  //发送效验位
  while (PS2_CLK)
    ;
  PS2_DATA = 1;
  while (!PS2_CLK)
    ;
  // 发送终止位
  PS2_DATA = 1;
  PS2_CLK = 1;
  while ((PS2_CLK) | (PS2_DATA))
    ;
  // 等待 ACK 握手信号
  while (!PS2_CLK)
    ;
  //等待 scl 变高
  EA = 1; //开总中断
}

/*********************************************************

键码变换为显示码子函数

 *********************************************************/
void PS2KEY_NUM()
{
  if (break_code)
  //断码标志
  {
    break_code = 0; //清断码标志
    return ; //退出
  }

  if (KeyCode == ESC_CODE)
  //Esc 键码
  {
    KeyCode = 0;
    SEND_DATA(0xff); //键盘复位
    beep();
    return ; //退出
  }

  if ((KeyCode == NUMLOCK_CODE) &numlock)
  //NumLock 键码
  {
    numlock = 0;
    KeyCode = 0;

    SEND_DATA(0xED); //点亮 NumLock
    SEND_DATA(0x02);

    TEMP1 = 0xc8; //显示'n'
    P0 = TEMP1;
    beep();
    return ; //退出
  }
  else
  {
    if ((KeyCode == NUMLOCK_CODE) &(!numlock))
    {
      numlock = 1;
      KeyCode = 0;
```

```
        SEND_DATA(0xED); //熄灭 NumLock
        SEND_DATA(0x00);
        TEMP1 = 0xc8; //显示'n'
        P0 = TEMP1;
        beep();
        return ;
      }
    }

    for (y = 0; y < 27; y++)
    //键值变换为顺序码
    {
      if (KeyCode == TABLE_D[y])
      //查表比较
      {
        if ((y > 15) &(numlock))
        //NumLock 失效
        {
          KeyCode = 0x00;
          return ;
        }
        else
        {
          TEMP1 = TABLE_NUM[y]; //查表取数
          P0 = TEMP1; //送显示
          beep(); //查找有效，蜂鸣器响一声
          KeyCode = 0x00;
          return ;
        }
      }
      else
      //没有定义的键
      {
        TEMP1 = 0xbf; //显示'-'
        P0 = TEMP1;
      }
    }
    BF = 0;
}

/*********************************************************

主函数

 *********************************************************/
void main(void)
{
  TEMP1 = 0xbf; //显示'-'
  P0 = TEMP1;
  P2 = 0xfe;

  IT1 = 0; //外部中断 1 为低电平触发
  EA = 1; //开总中断
  EX1 = 1; //开外部中断 1
  SEND_DATA(0xff); //键盘复位

  while (1)
  {
    P0 = TEMP1; //显示数据

    if (BF)
    {
      BF = 0;
      PS2KEY_NUM();
    }
  }
}

/*********************************************************

外部中断函数

 *********************************************************/
void ReadPS2()interrupt 2
```

```
{
  EX1 = 0;
  while (PS2_CLK)
    ;
  if ((IntNum > 0) && (IntNum < 9))
  //跳过起始位
  {
    KeyCode = KeyCode >> 1; //因键盘数据是低>>高
    if (PS2_DATA)
      KeyCode = KeyCode | 0x80;
    //当键盘数据线为 1 时为 1 到最高位
  }
  PS2_CLK = 1;
  IntNum++;
  while (!PS2_CLK)
    ;
  //等待 PS/2CLK 拉高

  if (IntNum == 11)
  //当中断 11 次后表示一帧数据收完
  {
    IntNum = 0; //清变量准备下一次接收
    BF = 1;

    if (KeyCode == 0xf0)
    //当收到 0xF0，表示是断码
    {
      break_code = 1;
      BF = 0; //不进行数据处理
    }
  }
  EX1 = 1; //开外部中断 1
}

/*************************************************************/
```

6.2 综合实验

实验二十五　PWM 控制 LED 灯渐亮渐灭

1. 实验任务

利用定时器控制产生占空比可变的 PWM 波
按 K1，PWM 值增加，则占空比减小,P0 口的 8 个 LED 灯渐暗
按 K2，PWM 值减小，则占空比增加,P0 口的 8 个 LED 灯渐亮
当 PWM 值增加到最大值或减小到最小值时，蜂鸣器将报警

2. 实验线路

本实验涉及到的实验硬件模块有 LED 显示（图 6.2）、独立按键（图 6.14）、蜂鸣器（图 6.7）

3. 实验步骤

将相应的实验硬件模块的短接子插上跳线帽使之与实验 CPU 的 I/O 端口连接（LED 显示部分的 JP13，独立按键部分的 JP8，蜂鸣器部分的 JP15）。

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX25_PWM CONTROL LED）

6. C 语言源程序

（见光盘： Example_C51\EX25_PWM CONTROL LED）

实验二十六　数码管左移显示

1. 实验任务

8 位数码管左移显示“1-8”

```
                                ;开机时，数码管黑屏。
                              1 ;逐字移入
                            1 2
                          1 2 3
                        1 2 3 4
                      1 2 3 4 5
                    1 2 3 4 5 6
                  1 2 3 4 5 6 7
1 2 3 4 5 6 7 8   ;停留 2s
2 3 4 5 6 7 8     ;逐字移出
3 4 5 6 7 8
4 5 6 7 8
5 6 7 8
6 7 8
7 8
8
                                ;黑屏
```

2. 实验线路

见图 6.9

3. 实验步骤

将 JP21 的 8 个短接子用短接帽短接，使数码管的位控制线与 P2 端口接通。

将 JP22 的 9 个短接子用短接帽短接，使数码管的数据线与 P0 端口接通，并使 VCC 向数码管接口电路供电。

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX26_7SEG Move L）

6. C 语言源程序

（见光盘： Example_C51\EX26_7SEG Move L）

实验二十七　数码管右移显示

1. 实验任务

8 位数码管右移显示“1-8”

```
                                                ;黑屏，开机时，数码管黑屏
1                                               ;逐字移入
2  1
3  2  1
4  3  2  1
5  4  3  2  1
6  5  4  3  2  1
7  6  5  4  3  2  1
8  7  6  5  4  3  2  1                         ;停留 2s
   8  7  6  5  4  3  2                         ;逐字移出
      8  7  6  5  4  3
         8  7  6  5  4
            8  7  6  5
               8  7  6
                  8  7
                     8
                                                ;黑屏
```

2. 实验线路

见图 6.9

3. 实验步骤

将 JP21 的 8 个短接子用短接帽短接，使数码管的位控制线与 P2 端口接通。

将 JP22 的 9 个短接子用短接帽短接，使数码管的数据线与 P0 端口接通，并使 VCC 向数码管接口电路供电。

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX27_7SEG Move R）

6. C 语言源程序

（见光盘： Example_C51\EX27_7SEG Move R）

实验二十八　数码管左右移动显示

1. 实验任务

8 位数码管左右移动显示“1-8”

```
                                                ;开机时，数码管黑屏。先左移：(从右向左移动)
                                              1 ;逐字移入
                                            1 2
                                          1 2 3
                                        1 2 3 4
                                      1 2 3 4 5
                                    1 2 3 4 5 6
                                  1 2 3 4 5 6 7
1 2 3 4 5 6 7 8                                 ;停留 2s
2 3 4 5 6 7 8                                   ;逐字移出
3 4 5 6 7 8
4 5 6 7 8
5 6 7 8
6 7 8
7 8
8                                               ;黑屏
                                                ;黑屏，开始右移：(从左向右移动)
1                                               ;逐字移入
2 1
3 2 1
4 3 2 1
5 4 3 2 1
6 5 4 3 2 1
7 6 5 4 3 2 1
8 7 6 5 4 3 2 1                                 ;停留 2s
  8 7 6 5 4 3 2                                 ;逐字移出
    8 7 6 5 4 3
      8 7 6 5 4
        8 7 6 5
          8 7 6
            8 7
              8
                                                ;黑屏
```

2. 实验线路

见图 6.9

3. 实验步骤

将 JP21 的 8 个短接子用短接帽短接，使数码管的位控制线与 P2 端口接通。

将 JP22 的 9 个短接子用短接帽短接，使数码管的数据线与 P0 端口接通，并使 VCC 向数码管接口电路供电。

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX28_7SEG Move LR）

6. C 语言源程序

（见光盘： Example_C51\EX28_7SEG Move LR）

实验二十九 数码管字幕显示

1. 实验任务

从右向左移动显示：--HELLO—

2. 实验线路

见图 6.9

3. 实验步骤

将 JP21 的 8 个短接子用短接帽短接，使数码管的位控制线与 P2 端口接通。

将 JP22 的 9 个短接子用短接帽短接，使数码管的数据线与 P0 端口接通，并使 VCC 向数码管接口电路供电。

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX29_7SEG Caption）

6. C 语言源程序

（见光盘： Example_C51\EX29_7SEG Caption）

实验三十　LCD12864 并口 4 位数据传输方式显示

1. 实验任务

LCD12864 并口方式可以分为 8 位和 4 位数据传输方式，8 位传输方式是用 D0-D7 数据线来传送控制命令及数据。4 位传输方式是用 D4-D7 数据线来传送控制命令及数据。使用 4 位数据线传输时，需要分两次来传送，先送出高 4 位数据，再送出低 4 位数据。可以节省单片机的 4 根端口线。

2. 实验线路

见图 6.25

3. 实验步骤

由于数码管，LED 都占用了 P0 或者 P2 口，进行 12864LCD 实验时需要进行如下设置，否则 12864LCD 可能不能正常显示：

断开数码管部分 JP22 跳线 Vcc 端的短路帽；

断开 LED 部分 JP13 跳线 Vcc 端短路帽；

将 12864LCD 模块插在 ME830 的 J6 插针上，通电后，LED 的背光应点亮；

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX30_LCD12864 4BIT）

6. C 语言源程序

（见光盘： Example_C51\EX30_LCD12864 4BIT）

实验三十一　LCD12864 串口传输方式显示

1. 实验任务

学习 12864LCD 的串口传输方式显示。

2. 实验线路

见图 6.25

3. 实验步骤

由于数码管，LED 都占用了 P0 或者 P2 口，进行 12864LCD 实验时需要进行如下设置，否则 12864LCD 可能不能正常显示：

断开数码管部分 JP22 跳线 Vcc 端的短路帽；

断开 LED 部分 JP13 跳线 Vcc 端短路帽；

将 12864LCD 模块插在 ME830 的 J6 插针上，通电后，LED 的背光应点亮；

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX31_LCD12864 Serial）

6. C 语言源程序

（见光盘： Example_C51\EX31_LCD12864 Serial）

实验三十二　蜂鸣器模拟枪声

1. 实验任务

模拟枪声演示程序，由 K1 键控制发声。

2. 实验线路

见图 6.7，图 6.14

3. 实验步骤

短接 JP15 短接子，使蜂鸣器接口电路使能；

将 JP8 的 8 个短接子全部用短接帽短接，使独立按键与相应的端口接通。

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX32_Buzzer Gun Sound）

6. C 语言源程序

（见光盘： Example_C51\EX32_Buzzer Gun Sound）

实验三十三　蜂鸣器模拟救护车警报声

1. 实验任务

单片机控制蜂鸣器模拟救护车警报声

信号产生的方法：

1）用 245us 延时程序作为时间基准，产生频率约为 2KHz 的音频信号。

2）用 325us 延时程序作为时间基准，产生频率约为 1.5KHz 的音频信号。

输出方式：

1）用 P3.7 端口输出 2KHz 音频信号，持续时间大约 1 秒。

2）用 P3.7 端口输出 1.5KHz 音频信号，持续时间大约 1 秒。

交替执行上述两个过程,形成救护车警报声。

2. 实验线路

见图 6.7

3. 实验步骤

短接 JP15 短接子，使蜂鸣器接口电路使能；

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX33_Buzzer Ambulance）

6. C 语言源程序

（见光盘： Example_C51\EX33_Buzzer Ambulance）

实验三十四　蜂鸣器模拟消防车警报声

1. 实验任务

产生由低到高,又由高到低的声音,就如同消防车的报警声音

2. 实验线路

见图 6.7

3. 实验步骤

短接 JP15 短接子，使蜂鸣器接口电路使能；

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX34_Buzzer Fire Engine）

6. C 语言源程序

（见光盘： Example_C51\EX34_Buzzer Fire Engine）

实验三十五　0-99 秒循环计时

1. 实验任务

0-99 秒循环计时，2 位数码管显示

2. 实验线路

见图 6.9

3. 实验步骤

将 JP21 的 8 个短接子用短接帽短接，使数码管的位控制线与 P2 端口接通。

将 JP22 的 9 个短接子用短接帽短接，使数码管的数据线与 P0 端口接通，并使 VCC 向数码管接口电路供电。

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX35_0-99 Count）

6. C 语言源程序

（见光盘： Example_C51\EX35_0-99 Count）

实验三十六　0-99 秒倒计时定时控制器

1. 实验任务

0-99 秒倒计时定时控制器，采用 4 位数码管显示，倒计时方式进行定时控制（显示剩余时间）。

L　0　　;初始状态显示

H　99　　;运行状态显示

按键功能定义：

K1 --- 定时时间设置－UP 键

K2 --- 定时时间设置－DOWN 键

K3 --- 停止定时键

K4 --- 启动定时键

每按一下有效键，蜂鸣器响一声。

L --- 定时停止状态标志符

H --- 定时运行状态标志符

使用方法：

根据定时需要，先用设置键对定时时间进行设置。

设置好定时时间后，按 K4 键启动定时控制运行；

在定时控制运行中，按 K3 键可停止定时控制运行；

当定时时间为“0”时，按 K4 键也无法启动定时控制运行；

当设置值倒计数减为“0”时，系统停止定时控制运行，并自动将定时值送控制系统，为下一次运行做准备；

在运行状态下，K1、K2、K4 键失效；

在非运行状态下，K3 键失效。

控制功能：

定时控制运行时，继电器吸合(DL11 亮)。

定时控制停止时，继电器释放(DL11 灭)。

2. 实验线路

分别见图 6.9，图 6.7，图 6.5，图 6.14

或参照光盘整机原理图

3. 实验步骤

将以下硬件模块的 I/O 选择跳线分别用跳线帽短接，使相关硬件模块与单片机 I/O 口连通：

JP21：数码管的位控制线；

JP22：数码管的数据线，并使 VCC 向数码管电路供电；

JP15：蜂鸣器端口选择跳线；

JP16：继电器端口选择跳线；

JP8： 独立按键端口选择跳线；

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX36_0-99 Count Down）

6. C 语言源程序

（见光盘： Example_C51\EX36_0-99 Count Down）

实验三十七 8 位数码管显示秒表

1. 实验任务

8 位数码管显示秒表：

K1 — 启动 / 暂停 功能键

K4 — 复位键

在计时工作运行状态下，K4 键失效

数码管显示单位为：分、秒、10 毫秒。

2. 实验线路

分别见图 6.9，图 6.7，图 6.14

或参照光盘整机原理图

3. 实验步骤

将以下硬件模块的 I/O 选择跳线分别用跳线帽短接，使相关硬件模块与单片机 I/O 口连通：

JP21：数码管的位控制线；

JP22：数码管的数据线，并使 VCC 向数码管电路供电；

JP15：蜂鸣器端口选择跳线；

JP8： 独立按键端口选择跳线；

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX37_STOPWATCH 7SEG）

6. C 语言源程序

（见光盘： Example_C51\EX37_STOPWATCH 7SEG）

实验三十八 1602 液晶显示秒表

1. 实验任务

1602LCD 显示秒表：

K1 — 启动 / 暂停 功能键

K4 — 复位键

在计时工作运行状态下，K4 键失效

LCD1602 显示单位为：时、分、秒、10 毫秒，还可以显示 K1 键按下的次数。

2. 实验线路

分别见图 6.7，图 6.14，图 6.20

或参照光盘整机原理图

3. 实验步骤

将 1602LCD 模块插在 J5 接口上；

将以下硬件模块的 I/O 选择跳线分别用跳线帽短接，使相关硬件模块与单片机 I/O 口连通：

JP15：蜂鸣器端口选择跳线；

JP8： 独立按键端口选择跳线；

将数码管 JP22 跳线组 Vcc 端的跳线帽断开，避免数码管电路干扰 1602LCD。

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX38_STOPWATCH LCD1602）

6. C 语言源程序

（见光盘： Example_C51\EX38_STOPWATCH LCD1602）

实验三十九　8 位数码管显示简易时钟

1. 实验任务

8 位数码管显示简易时钟，通过程序预设开始时间。

2. 实验线路

见图 6.9

3. 实验步骤

将以下硬件模块的 I/O 选择跳线分别用跳线帽短接，使相关硬件模块与单片机 I/O 口连通：

JP21：数码管的位控制线；

JP22：数码管的数据线，并使 VCC 向数码管电路供电；

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX39_SIMPLE CLOCK 7SEG）

6. C 语言源程序

（见光盘： Example_C51\ EX39_SIMPLE CLOCK 7SEG）

实验四十　1602 液晶显示简易时钟

1. 实验任务

1602 液晶显示简易时钟，通过程序预设开始时间。

2. 实验线路

见图 6.20

3. 实验步骤

将 1602LCD 模块插在 J5 接口上；

将数码管 JP22 跳线组 Vcc 端的跳线帽断开，避免数码管电路干扰 1602LCD。

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX40_SIMPLE CLOCK LCD1602）

6. C 语言源程序

（见光盘： Example_C51\EX40_SIMPLE CLOCK LCD1602）

实验四十一　8 位数码管显示通用时钟

1. 实验任务

8 位数码管显示通用时钟，可以用按键行时间设置，开始运行与显示。

K4 ———— 功能选择键，选择顺序： 时、分、秒、退出，被选中的修改项闪动显示。

K1 ———— 加键

K2 ———— 减键

2. 实验线路

见图 6.7，图 6.9，图 6.14

或参考光盘整机原理图

3. 实验步骤

将以下硬件模块的 I/O 选择跳线分别用跳线帽短接，使相关硬件模块与单片机 I/O 口连通：

JP21：数码管的位控制线；

JP22：数码管的数据线，并使 VCC 向数码管电路供电；

JP15：蜂鸣器端口选择跳线；

JP8： 独立按键端口选择跳线；

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX41_UNIVERSAL CLOCK 7SEG）

6. C 语言源程序

（见光盘： Example_C51\EX41_UNIVERSAL CLOCK 7SEG）

实验四十二　1602 液晶显示通用时钟

1. 实验任务

1602 液晶显示通用时钟，可以用按键行时间设置，开始运行与显示。

K4 ———— 功能选择键，选择顺序： 时、分、秒、退出，被选中的修改项闪动显示。

K1 ———— 加键

K2 ———— 减键

2. 实验线路

见图 6.7，图 6.14，图 6.20。或参考光盘整机原理图

3. 实验步骤

将 1602LCD 模块插在 J5 接口上；

将以下硬件模块的 I/O 选择跳线分别用跳线帽短接，使相关硬件模块与单片机 I/O 口连通：

JP15：蜂鸣器端口选择跳线；

JP8： 独立按键端口选择跳线；

将数码管 JP22 跳线组 Vcc 端的跳线帽断开，避免数码管电路干扰 1602LCD。

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX42_UNIVERSAL CLOCK LCD1602）

6. C 语言源程序

（见光盘： Example_C51\EX42_UNIVERSAL CLOCK LCD1602）

实验四十三　8 位数码管显示闹钟

1. 实验任务

8 位数码管显示多功能闹钟，可实行定时闹铃，整点报时，定时控制的功能。

开机时 8 位数码管显示：

23－59－50　程序预设时间

时钟运行 10 秒后：

00－00－00　整点 24 点，蜂鸣器响 24 声进行整点报点

时钟再运行 30 秒后：

00－00－30　到定时起闹时间，蜂鸣器发出机械闹钟声

00－01－00　蜂鸣器停止发声

功能键的应用：

K1：设定定时起闹时间

K2：查看定时起闹时间

K3：定时控制开关

K4：设定实时时间

1）设定实时时间

K4：时间设定选择健

K2：增加键

K3：减少键

通过按 K4 健来选择修改项，被选中的修改项将会闪动

选择顺序：时、分、秒，设定完后，再按 K4 健，退出

2）查看定时起闹时间

K2：查看定时起闹时间

按一次 K2 健，显示起闹时间：00－00－30

再按一次 K2 健,退出

3）设定定时起闹时间

K1：定时时间设定选择健

K2：增加键

K3：减少键

选择顺序：时、分、秒。

通过按 K1 健来选择修改项，被选中的修改项将会闪动，设定完后，再按 K1 健，退出。

4）定时输出控制

K3：定时输出控制

当定时起闹时间与实时时间一致时，蜂鸣器发出机械闹钟声，数码管开始闪烁显示（开始闹时）

数码管闪烁显示 30 秒后，自动恢复正常显示（停止闹时）

在数码管闪烁显示期间，可以按 K3 中止数码管闪烁显示和蜂鸣器发声（中止闹时）

整点报数功能：

根据整点值，蜂鸣器响“整点值”次。

蜂鸣器的功能：

每当有键按下，蜂鸣器将响一声。

整点报数。

继电器的功能：

闹时有效时，继电器吸合，否则继电器释放。

2. 实验线路

见图 6.5，图 6.7，图 6.9，图 6.14

或参考光盘整机原理图

3. 实验步骤

将以下硬件模块的 I/O 选择跳线分别用跳线帽短接，使相关硬件模块与单片机 I/O 口连通：

JP21：数码管的位控制线；

JP22：数码管的数据线，并使 VCC 向数码管电路供电；

JP15：蜂鸣器端口选择跳线；

JP16：继电器端口选择跳线；

JP8： 独立按键端口选择跳线；

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX43_ ALARM CLOCK 7SEG）

6. C 语言源程序

（见光盘： Example_C51\EX43_ ALARM CLOCK 7SEG）

实验四十四　1602 液晶显示闹钟

1. 实验任务

1602 液晶显示多功能闹钟，可实行定时闹铃，整点报时，定时控制的功能。

开机时 LCD1602 显示：

ALARM　CLOCK
TIME: 23－59－50　　程序预设时间

时钟运行 10 秒后：

LARM　CLOCK
TIME: 00:00:00　　整点 24 点，蜂鸣器响 24 声进行整点报点

时钟再运行 30 秒后：

ALARM　CLOCK
TIME: 00:00:30　　到定时起闹时间，蜂鸣器发出机械闹钟声
ALARM　CLOCK
TIME: 00:01:00　　蜂鸣器停止发声

功能键的应用：

K1: 设定定时起闹时间
K2: 查看定时起闹时间
K3: 定时功能控制开关
K4: 设定实时时间

1) 设定实时时间

K4: 时间设定选择健
K2: 增加键
K3: 减少键

SET REAL_TIME
TIME: 23:59:50

通过按 K4 健来选择修改项，被选中的修改项将会闪动
选择顺序：时、分、秒，设定完后，再按 K4 健，退出

2) 查看定时起闹时间

K2: 查看定时起闹时间
按一次 K2 健，显示起闹时间

LOOK ALARM_TIME
TIME: 00－00－30

再按一次 K2 健,退出

3) 设定定时起闹时间

K1: 定时时间设定选择健
K2: 增加键
K3: 减少键

SET ALARM_TIME
TIME: 00:00:30

选择顺序：时、分、秒，

通过按 K1 健来选择修改项，被选中的修改项将会闪动，设定完后。再按 K1 健，退出。

4）定时控制

K3：定时控制

可选择定时起闹功能使能与失效，LCD1602 第 2 行 15 列有小喇叭显示，当小喇叭显示时，起闹功能使能。当无小喇叭显示时，起闹功能失效。

当定时起闹时间与实时时间一致时，蜂鸣器发出机械闹钟声（30 秒），在定时起闹期间，可以按 K3 中止定时和蜂鸣器发声（中止闹时）。

整点报数功能：

根据整点值，蜂鸣器响“整点值”次

蜂鸣器的功能：

每当有键按下，蜂鸣器将响一声

整点报数

继电器的功能：

闹时有效时，继电器吸合，否则继电器释放。

2. 实验线路

见图 6.5，图 6.7，图 6.14，图 6.20

或参考光盘整机原理图

3. 实验步骤

将 1602LCD 模块插在 J5 接口上；

将以下硬件模块的 I/O 选择跳线分别用跳线帽短接，使相关硬件模块与单片机 I/O 口连通：

JP15：蜂鸣器端口选择跳线；

JP16：继电器端口选择跳线；

JP8： 独立按键端口选择跳线；

将数码管 JP22 跳线组 Vcc 端的跳线帽断开，避免数码管电路干扰 1602LCD。

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX44_ALARM CLOCK LCD1602）

6. C 语言源程序

（见光盘： Example_C51\EX44_ALARM CLOCK LCD1602）

实验四十五　DS18B20 温度检测与控制(数码管显示)

1. 实验任务

DS18B20 温度检测与控制，8 位数码管显示。

数码管显示格式：

第 1 和第 2 位数码管显示温度符号℃。

第 3 至第第 6 位数码管显示温度值。

第 7 位显示报警状态符号“L，H”

开机检测 DS18B20 状态：

DS18B20 正常，则显示： 18.8 ℃ （实际温度值）

DS18B20 不正常，则黑屏、蜂鸣器一直响。

按键功能说明：

1) 查看温度报警值：

K1 → 进入查看温度报警值 TH 状态（第一次）:

H 28℃

TH: 28 高位报警值

K1 → 进入查看温度报警值 TL 状态（第二次）:

L 20℃

TL: 20 低位报警值

K1 → 退出查看温度报警值状态（第三次）。

2) 设定温度报警值：

K3 → 进入设定温度报警值 TH 状态（第一次）:

H 28 （设定值闪动）

K3 → 进入设定温度报警值 TL 状态（第二次）:

L 20 （设定值闪动）

K3 → 返回（第三次）

设定过程： K1 →加键 （UP）， K2 →减键 （DOWN），可快速调节。

将设定的温度报警值自动存入 DS18B20 的 EEROM 中，可永久保存，每次开机时自动从 DS18B20 的 EEROM 读出温度报警值。

报警功能说明：

1) 当实际温度大于 TH 的设定值时，第 7 位数码管显示 H ，关闭继电器，表示超温；

2）当实际温度小于 TL 的设定值时，第 7 位数码管显示 L；

3）当实际温度小于 TH 的设定值时，继电器吸合，DL11 (LED) 点亮。

2. 实验线路

见图 6.5，图 6.7，图 6.9，图 6.14，图 6.63

或参考光盘整机原理图

3. 实验步骤

将以下硬件模块的 I/O 选择跳线分别用跳线帽短接，使相关硬件模块与单片机 I/O 口连通：

JP21：数码管的位控制线；
JP22：数码管的数据线，并使 VCC 向数码管电路供电；
JP15：蜂鸣器端口选择跳线；
JP16：继电器端口选择跳线；
JP8： 独立按键端口选择跳线；

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX45_DS18B20 7SEG）

6. C 语言源程序

（见光盘： Example_C51\EX45_DS18B20 7SEG）

实验四十六 DS18B20 温度检测与控制(1602 液晶显示)

1. 实验任务

DS18B20 温度检测与控制，1602 液晶显示。

开机程序检测 DS18B20 状态：

DS18B20 如果不正常时，则 LCD 显示：

DS18B20 正常时，程序先读取 DS18B20 的 ROMCORD 码，送 LCD 显示：

延时 2 秒后，LCD1602 显示：

上图中 LCD1602 的第一行显示的小喇叭标志为温度报警时允许蜂鸣器响的标志。

TEMP: 21.3℃ ——— 实际温度显示值
TH: 24 TL: 19 ——— 设定的温度高、低报警值

按键功能：

1) 查看 DS18B20 的 ROMCORD 码值

K1 → 进入查看 DS18B20 的 ROMCORD 码值
K2 → 退出查看

2) 设定温度报警值

K3 → 选择键（根据按键的次数来确定）
第一次→ 设定温度报警 TH 值
第二次→ 设定温度报警 TL 值
第三次→ 退出设定状态
K1 → 设定值加(UP)
K2 → 设定值减(DOWN)
在设定中以闪动方式显示设定值。

蜂鸣器控制

K4 → 控制蜂鸣器在报警时是否响
LCD1602 有蜂鸣器响的标志显示，则响。LCD1602 无蜂鸣器响的标志显示，则不响。

报警功能

当实际温度大于 TH 的设定值时，在显示 TH 数值的后面有小喇叭闪动显示，蜂鸣器报警。
当实际温度小于 TL 的设定值时，在显示 TL 数值的后面有小喇叭闪动显示，蜂鸣器报警。

2. 实验线路

见图 6.5，图 6.7，图 6.14，图 6.20
或参考光盘整机原理图

3. 实验步骤

将 1602LCD 模块插在 J5 接口上；
将以下硬件模块的 I/O 选择跳线分别用跳线帽短接，使相关硬件模块与单片机 I/O 口连通：
JP15：蜂鸣器端口选择跳线；
JP16：继电器端口选择跳线；
JP8： 独立按键端口选择跳线；
将数码管 JP22 跳线组 Vcc 端的跳线帽断开，避免数码管电路干扰 1602LCD。

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\ EX46_DS18B20 LCD1602）

6. C 语言源程序

（见光盘： Example_C51\ EX46_DS18B20 LCD1602）

实验四十七　步进电机加减速运行

1. 实验任务

步进电机三种运行状态控制：
步进电机启动时，转速由慢到快逐渐加速，发光二极管 D00 点亮；
步进电机匀速运行，发光二极管 D01 点亮；
步进电机由快到慢逐渐减速，发光二极管 D02 点亮。
K1 键启动步进电机运行。

2. 实验线路

见图 6.2，图 6.14，图 6.38
或参考光盘整机原理图

3. 实验步骤

将以下硬件模块的 I/O 选择跳线分别用跳线帽短接，使相关硬件模块与单片机 I/O 口连通：
JP13：LED 端口选择跳线；
JP8：独立按键端口选择跳线；

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX47_STEP MOTOR Variable）

6. C 语言源程序

（见光盘： Example_C51\EX47_STEP MOTOR Variable）

实验四十八　键控步进电机加减速运行

1. 实验任务

键控步进电机加减速运行：
按下 K1，转速逐步减速，达到最低速时发光二极管 D00 点亮；
按下 K2，转速逐步加速，达到最高速时发光二极管 D01 点亮。

2. 实验线路

见图 6.2，图 6.14，图 6.38
或参考光盘整机原理图

3. 实验步骤

将以下硬件模块的 I/O 选择跳线分别用跳线帽短接，使相关硬件模块与单片机 I/O 口连通：
JP13：LED 端口选择跳线；
JP8：独立按键端口选择跳线；

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX48_STEP MOTOR KEY）

6. C 语言源程序

（见光盘： Example_C51\EX48_STEP MOTOR KEY）

实验四十九　红外遥控步进电机

1. 实验任务

使用红外遥控器对步进电机的转动方向和速度进行控制，1602 液晶显示红外遥控器键值编码。

遥控器的 5 个有效按键：

UP　键按下步进电机正转，显示 >>>>

DOWM 键按下步进电机反转，显示 <<<<

K1　键按下步进电机停止转动，显示 STOP

＋　键减少延时时间，步进电机加速

－　键增加延时时间，步进电机减速

注意：步进电机转动时请先按 K1 停止后，再切换转动方向。

2. 实验线路

见图 6.7，图 6.20，图 6.66

或参考光盘整机原理图

3. 实验步骤

将 1602LCD 模块插在 J5 接口上；

将以下硬件模块的 I/O 选择跳线分别用跳线帽短接，使相关硬件模块与单片机 I/O 口连通：

JP15：蜂鸣器端口选择跳线；

JP10：红外接收头端口选择跳线；

将数码管 JP22 跳线组 Vcc 端的跳线帽断开，避免数码管电路干扰 1602LCD。

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX49_STEP MOTOR IR）

6. C 语言源程序

（见光盘： Example_C51\EX49_STEP MOTOR IR）

实验五十　电子密码锁

1. 实验任务

开机后，LCD1602 显示如下：

等待你输入密码，输入密码方法：

1）按“F”键启动进入输入密码程序

按住“F”键(3 秒以上)进入输入密码状态，LCD1602 显示：

2）使用矩阵键盘输入密码

在输入密码状态下，0-9 数字键为有效键。有时间、次数限制功能，不给别人试探的机会。有三次输入密码机会，每次限制在 10 秒内完成。

密码输入正确后，LCD1602 显示：

继电器吸合，表示锁打开。

密码输入错误后，LCD1602 显示：

键入密码有误或每次输入密码时间超过 10 秒，被认为是密码输入错误。

当 3 次输入都错误时，程序将返回起始状态。

在密码输入正确的情况下，程序进入看密码和修改密码状态。

看密码功能：

按 “A” 键进入看密码状态 LCD1602 显示：

按“E” 键退出看密码状态。

重新设置密码功能：

按 “B” 键进入重新设置密码状态 LCD1602 显示：

在输入新密码时，如果输入有误，可按“C”删除后，重新输入；
按“E”确认后，程序退出修改密码状态；
按“D”键或等待 10 秒后程序退出修改密码和看密码状态，回到起始状态。

程序内定密码为：987654
送电开机时，只要输入内定密码便可开门，这样可预防停电后再送电时无密码可用。

2. 实验线路

见图 6.5，图 6.7，图 6.18，图 6.20
或参考光盘整机原理图

3. 实验步骤

将 1602LCD 模块插在 J5 接口上；
将以下硬件模块的 I/O 选择跳线分别用跳线帽短接，使相关硬件模块与单片机 I/O 口连通：
JP15：蜂鸣器端口选择跳线；
JP16：继电器端口选择跳线；
JP7： 矩阵按键端口选择跳线；
将数码管 JP22 跳线组 Vcc 端的跳线帽断开，避免数码管电路干扰 1602LCD。

4. 程序流程图

流程图略

5. 汇编源程序

（见光盘： Example_A51\EX50_Electronic Code Lock）

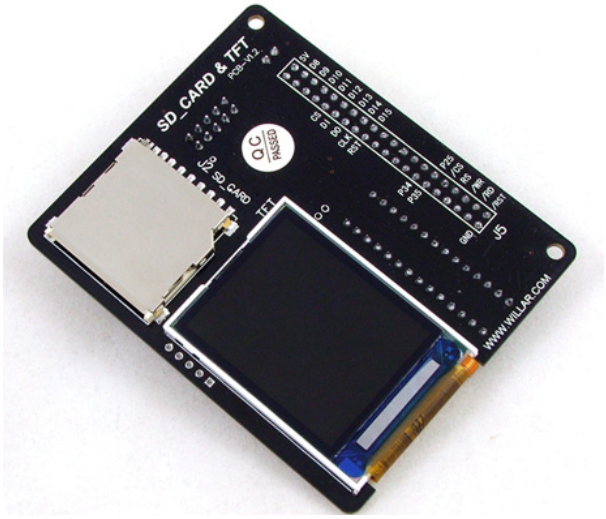
6. C 语言源程序

（见光盘： Example_C51\EX50_Electronic Code Lock）

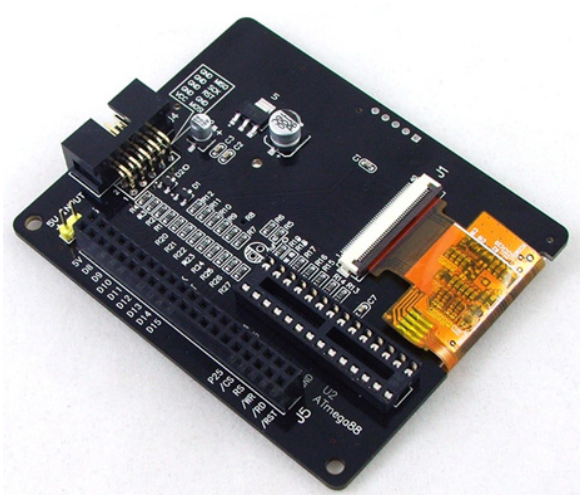
6.3 扩展实验

“TFT 彩屏+SD 卡模块”是硕飞科技(伟纳电子)为 ME830/850 单片机开发实验仪开发的一款多功能扩展模块。板载标准 SD/MMC 卡座、1.8 寸 TFT 真彩液晶屏、3.3V 稳压器，另外带有 28Pin DIP 插座，带有 ISP 下载接口，插上 ATmega88V 单片机芯片后即可当作一台独立的 AVR 开发板使用。此模块可以完成单片机读写 SD 卡实验、单片机驱动 TFT 真彩液晶实验，制作数码相框等。如果和 ME830/850 单片机开发实验仪配套使用，可以完成更多丰富的实验：如 TFT 彩屏显示多功能时钟(带温度显示)、TFT 彩屏显示红外遥控步进电机、TFT 彩屏显示矩阵键盘识别等，TTF 真彩屏的显示效果远非 12864 黑白液晶可比。我们为本模块编写了大量的实验例程，详见光盘 Examples_TFT+SD 文件夹。

TFT 彩屏+SD 卡模块使用说明如下：



TFT 彩屏+SD 卡模块正面图



TFT 彩屏+SD 卡模块背面图

性能特点

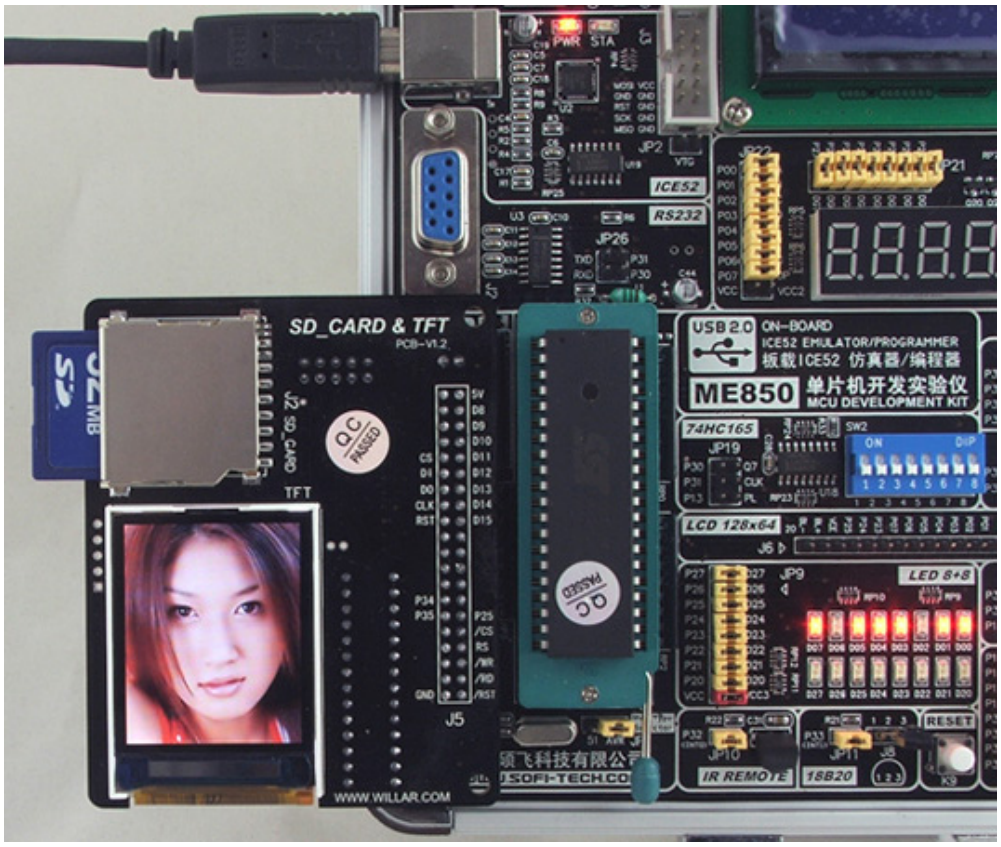
●板载标准 SD/MMC 卡座一个；
●板载 1.8 寸，128x160 真彩液晶屏一块；
●板载 28Pin DIP 插座一个，可以插上 ATmega88V（此芯片为选配件）作为 AVR 开发板使用；预留有标准 ISP 下载接口。
●预留有 40Pin 插座，可以直接插在 ME830/850 的扩展接口上进行实验；
●提供丰富的实验例程：如 SD 卡读写实验，AVR 单片机实现数码相框功能；TFT 液晶显示数字温度表，TFT 彩屏显示多功能时钟，TFT 彩屏显示红外遥控步进电机.....

模块的使用与演示

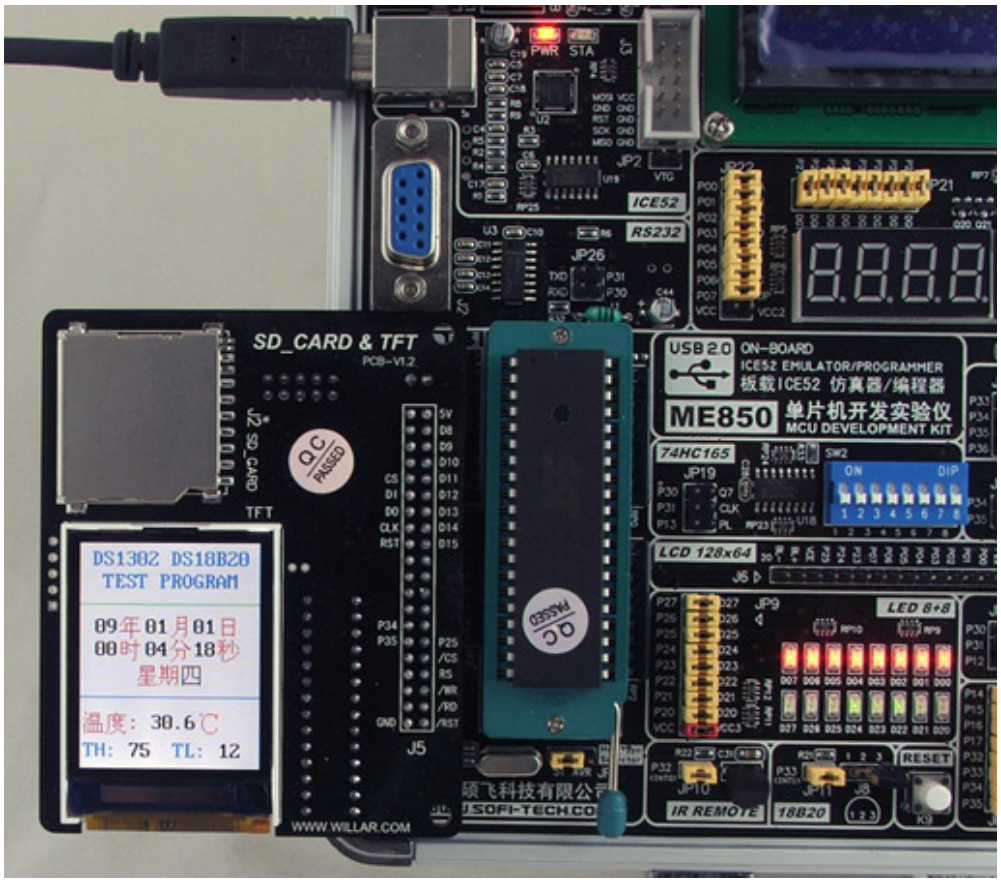
1. 51 单片机实验

首选需要检查“TFT+SD 卡模块”背面的 DIP28Pin 插座（U2）上是否插有 ATmega88V 芯片，如果有需要先取下，否则就不能进行 51 单片机的实验。然后在关闭电源的情况下将“TFT+SD 卡模块”背面的“J5 插座”对准 ME830/850 的扩展接口 J4 插入，如下图所示：

51 单片机驱动 TFT 实验

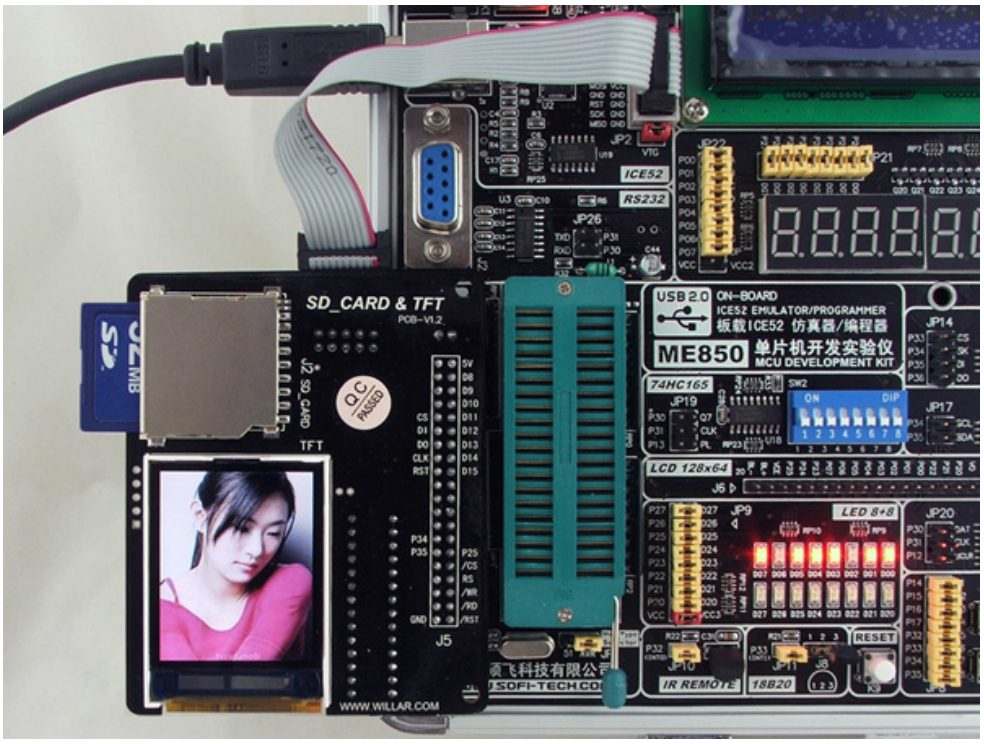


TFT 显示多功能数字钟实验



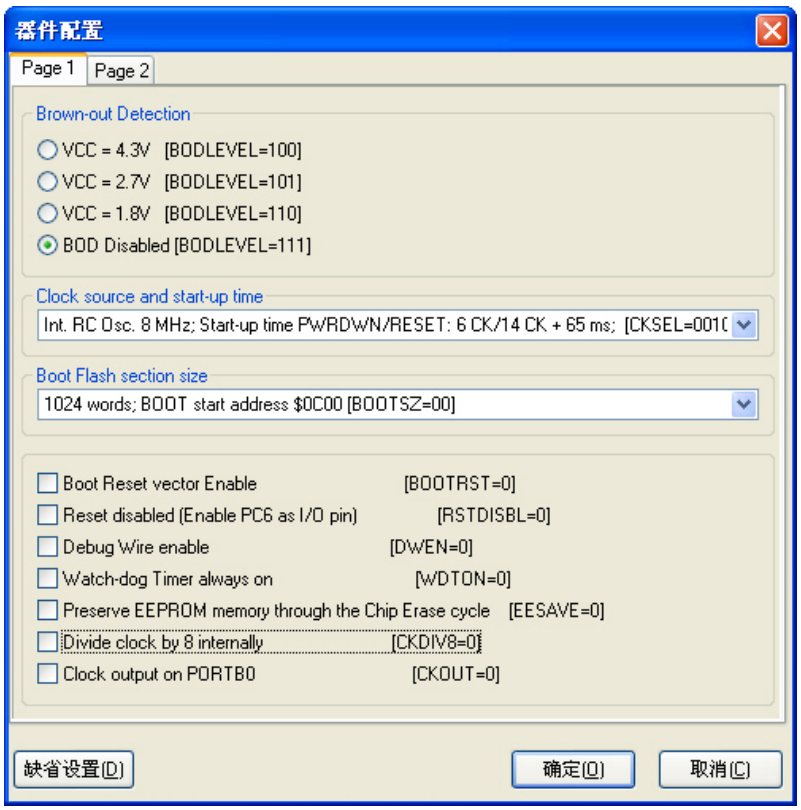
2. AVR 单片机实验

在模块的 U2 插座上插上一片 ATmega88V 单片机芯片（DIP40 封装），将 ISP 线连接一端连接到 ME830/850 的 JP2 接口，另外一端连接到模块的 J4 接口，然后将模块插到 ME830/850 主机的扩展接口 J4 上，将 ME830/850 锁紧座下方的 JP1 跳线切换到“AVR”位置，如下图



烧写 AVR 芯片的方法与注意事项：

- 使用MEFlash软件，启动软件后先选择芯片型号 ATmega88V@ISP（@ISP表示用ISP方式烧写）
- 配置熔丝位，点击菜单“配置”，按下图设置即可（注意时钟源"Clock source and start-up time"需按下图选择内部 RC 振荡方式“Int. RC Osc. ...”，千万不要勾选“Reset disabled...”项，否则芯片将被锁死而不能用 ISP 方式烧写，必须用高压编程器恢复熔丝位）



- 操作选项设置，如下图：

- 点击"自动"按钮即可（熔丝位写入后如果不需要重新配置可以不用每次都写）

郑重声明

在编写 ME830/ME850 实验程序和相关功能介绍过程中，参阅了大量书籍和资料，首先向书籍和资料的作者表示衷心感谢。

由于时间仓促，加之本人的理解能力和编写水平有限，难免出现一些错误和不妥之处，请 ME830 用户多多见谅并批评指正。

ME830/ME850 相关实验例程和技术资料仅供用户在学习中参考，不得用于商业用途。如果需要转载或引用，请保留版权声明和出处。如果您在学习过程中有任何疑问，欢迎到我们的技术支持网站论坛发帖交流！

伟纳电子（深圳硕飞科技有限公司） Gguoqing
公司网站： www.sofi-tech.com
技术支持： www.mcusj.com （单片机世界）
www.willar.com （伟纳电子网）
2009-12-2

附录 1　常见问题解答

1. 通电后实验仪上的红灯（PWR）和绿灯（STA）同时闪是什么原因?

ME830/850 内置有 CPU 控制的过载短路保护功能，如果因放反芯片或者其他错误操作导致负载电流大于 500mA，保护电路将立即启动切断整机供电，同时 PWR 和 STA 两个 LED 闪烁报警，排除错误后按复位键即可恢复正常，可以有效的保护计算机 USB 接口和实验仪不会因意外而损坏。

2. 为何有时烧写的芯片不能运行?

烧写的芯片不能运行，通常有以下几个原因：

1)程序问题或者加载的文件不对。如果源代码程序有问题，那么烧写芯片肯定是不能正常工作的。

2)目标板硬件问题：单片机正常工作必须具备相应的硬件条件，例如外部电源，晶振的连接，复位电路等等。

3)编程操作步骤错误（如先读取后编程），正确的操作步骤：擦除->查空->编程->校验。

3. 为何有时做数码管、液晶、LED 显示实验的时候不能正常工作?

数码管、1602LCD，12864LCD，16 路 LED 这四个模块默认共用了 P0 口和 P2 口，如果同时使用可能会相互干扰。因此做其中某个模块实验的时候，最好将另外的三个模块断开或禁用。

比如做 1602LCD 或者 12864LCD 显示实验，要将数码管旁边的 JP22 跳线组的 VCC 端短路帽取下，否则液晶可能不能正常工作。同样在做数码管显示实验的时候，为避免 1602LCD 对数码管产生干扰，对于 ME830 直接取下液晶模块即可。ME850 已经固定安装了 1602LCD，只需将 1602LCD 上端的 JP24 跳线短接在 OFF 端就可以了（使用 1602LCD 时须短接在 ON 端）。

如果您需要同时使用 2 组显示模块，可以用杜邦线另外搭配线路。

4. 如何设置实验仪上的跳线?

一般按第二章 ME830/850 主板电路布局图的默认位置插上跳线帽即可。使用跳线帽连接硬件的原则是：默认短接的跳线帽可以永久保留不需断开，默认没有插短路帽的实验模块只有在做该模块的实验时才插上，不用时须取下短路帽，避免相互干扰。在 ME830/850 的电路布局图中： 表示已经插了跳线帽， 表示没有插跳线帽。

特别注意由于 1602LCD 是固定安装在 ME850 主板上的，任何时候不做 1602LCD 实验时一定要将 JP24 的短路帽短接在 OFF 位置来禁用 1602LCD，否则 1602LCD 可能干扰其他使用 P0，P2 口的实验模块；

对于数码管显示模块，不使用时只须断开 JP22 跳线组的 VCC 端跳线；

对于 16x16LED 点阵显示模块、不使用时只须断开 JP23 跳线组的 VCC 端跳线；

对于 LED 显示模块、不使用时只须断开 JP13 跳线组的 VCC 端跳线；

如果您需要自己定义硬件连接，可以取下实验模块的短接跳线帽，然后根据需要用杜邦线连接即可。

5. 烧写时提示器件 ID 错误是什么原因?

每一个器件都有其相应的 ID 信息，软件会通过该特性来检查芯片的型号是否正确。如果芯片的型号选择错误，将会出现此提示。还有一种可能是芯片已经损坏。

6. 烧写时提示器件初始化错误是什么原因?

在对部分器件(例如 AVR 系列单片机)进行烧写时，软件会首先对器件进行初始化操作， 如果初始化失败，软件会给出此提示。初始化失败的原因主要有：

1）AVR 芯片的熔丝位配置错误。AVR 系列芯片有一些熔丝位，如果设置不当将导致该芯片不能采用 ME830 系列开发实验仪进行 ISP 编程操作。这种情况下只能通过其他并行编程器将该熔丝位进行恢复，导致不能进行编程操作的熔丝位包括：

- ISP 编程使能熔丝位

● RESET 端口被设置为 IO 模式

● 时钟熔丝位被配置位不匹配的模式，例如选择了外部时钟源或者外部 RC 震荡，而单片机外部又没有相应的电路或电子元件.

2）ISP 连接错误. 在进行外部 ISP 编程操作，必须保证 ISP 的连线正确.

3）芯片有问题

4） 芯片没有内置相应的 ISP 启动代码，在对 SST/WINBOND/NXP 等公司的器件进行编程时，芯片内部必须首先内置有相应的 ISP 启动代码，否则芯片将不能进行编程操作。详细内容请参考光盘电子版使用手册。

7. 什么是 ISP ？

ISP 是 In System Programming 的简称，即在系统中编程。这种编程方式无需将芯片从线路板上取下来，仅仅需要连接几条的烧写控制线即可，用户在设计目标板时，预留相应的 ISP 接口，即可随时通过 ISP 更新单片机内的程序。

ME830 支持 ISP 编程操作，使用方法见第五章的介绍。注意：在进行 ISP 下载时，选择的器件型号必须带有@ISP 后缀。

8. 为什么 C 语言中有些代码行不能设置断点?

C 语言具有优化功能，如果当前的代码被优化掉了，则在此行代码前是不能设置断点的。

9. 使用 MEFlash 软件加载文件时自动关闭是什么原因？

有些精简版或克隆版的 XP 系统容易出现此问题，解决方法如下：

点击 开始——运行——输入“gpedit.msc”回车 打开组策略

然后在组策略中按下面的方法设置：

组策略→“本地计算机”策略→用户配置→管理模板→任务栏和[开始]菜单→不要保留最近打开文档的记录（如设置为“已启用”则出问题，设置为“未配置”则正常运行）

10. 仿真器连接失败是什么原因？

先检查仿真驱动是否正确按照，仿真软件是否正确设置，仿真头是否正常连接，请参照前面“4.3 仿真调试”章节检查设置。

需要注意的是：使用仿真功能时需要关闭 MEFlash 软件，另外如果是仿真外部目标板，复位电路的电容不能过大，最好不大于 10uF，否则可能造成仿真头连接超时！

仿真时请关闭 MEFlash 软件。

11. 单步仿真延时程序很慢是什么原因？

由于 Keil 仿真机制限制，在单步运行一条 C 语句(该 C 语言可能包含多条汇编指令)时，Keil 实际会执行多个单步运行，如果在这些指令中存在循环，那么 Keil 将连续不断的执行单步，导致单步运行会较慢。因此对于 C 语言的指令延时程序，请在下一条指令设置断点，直接跳过，不要使用单步运行。

12. 仿真器有什么作用？ME830/850 内置的 ICE52 仿真器有什么特点？

单片机仿真器是在产品开发阶段用来替代单片机进行软硬件调试的非常有效的开发工具。使用单片机仿真器可以对单片机程序进行单步、断点、全速等手段的调试，在集成开发环境中检查程序运行中单片机 RAM、寄存器内容的变化，观察程序的运行情况。与此同时可以对硬件电路进行实时的调试。使用单片机仿真器可以迅速发现和排除程序中的错误，从而大大缩短单片机开发的周期。

ME830/850 集成有伟纳电子（深圳硕飞科技有限公司）最新研发的增强型 USB2.0 接口 51 单片机仿真器 ICE52，仿真不占用用户任何资源，可极速单步，完全真实仿真标准 8051/8052 单片机的所有功能，可仿真部分单片机的增强型功能。

目前所见到的其他 51 系列开发板全部采用 SST 公版仿真方案,即直接使用一片价值十元左右的 SST 单片机配合 keil 软件来调试，具有本质的缺陷：如占用串口、定时器 T2 和 8 字节堆栈，单步速度缓慢，对于真正开

发产品没有太大价值。

仿真功能对比表

对比项	ICE52 仿真器	其他千元级以下仿真器	SST 公版简易仿真器
串口占用	不占用	不占用	占用
定时器占用	不占用	不占用	占用
堆栈占用	2 字节	2 字节	9 字节
脱机运行	支持(自动处理)	支持(需手动设置)	不支持
Keil 驱动协议(注 2)	SFICE52.DLL(硕飞自主开发)	MON51.DLL	MON51.DLL
通讯接口	USB 2.0(真正的 USB 接口,速度快,无需任何设置操作)	串口或 USB 转串口 (速度慢, 需手动设置端口号和波特率)	
复位再运行(注 3)	支持	不支持(需再次下载)	
夭折功能(运行中暂停)	支持	支持	支持(但占用串口中断向量,并影响全速运行实效)
断点数量	19+1	10	9+1
单步运行速度(注 1)	极速(<0.1s @11MHz)	100ms 左右	超慢
仿真空间	63K	63K	63K
Keil Flash Download(注 4)	支持 (支持 ATMEL 的 51 系列芯片)	不支持	不支持
ISP 下载(注 4)	ICE52B 可支持(支持 AT89S 及 AVR 系列单片机的 ISP 下载编程)	不支持	不支持

注 1: 运行速度基于目标晶振 11.0591MHz 进行测试，视不同的晶振以及具体情况会稍有不同。

注 2: 硕飞科技自主开发的 Keil 仿真驱动协议,功能相比于 MON51.DLL 有较大的突破，例如：更改通讯接口为真正的 USB 口,支持 Keil Flash Download 功能，支持软件复位，增加断点数量等等。

注 3: 复位再运行是指在仿真器运行过程中，用户按下复位按钮之后，可以再次进行全速/单步/跨步运行操作。其他业余仿真器必须重新下载代码(再次下载代码需要在 KEIL 中退出仿真状态之后，再进入，比较麻烦)，否则无法正常运行。

注 4: ICE52B 仿真器集成有独特的 ISP 下载功能，可以直接在 keil 中实现对 AT89S 系列芯片进行 ISP 下载编程，此功能通过点击 Keil 的 Flash Download 按钮执行，此功能在开发阶段非常方便和实用，无需在开发软件和烧写软件之间频繁切换。如果配合硕飞专门开发的 MeFlash 软件，可以支持 AT89S 系列和 AVR 系列单片机的 ISP 下载。注意 ICE52A 仿真器没有 ISP 下载功能。

13. ME830/ME850 各有什么特点？作为初学者选择哪一款比较合适？

ME830 和ME830 同属于ME800 系列，采用相同的系统架构，可支持实验和及开发的芯片型号数量完全一样。不同之处总结如下（具体请参考选型指南：http://www.willar.com/shop_cast.asp?action=view&id=19）

硬件资源：

相比 ME830，ME830 增加了 74HC164 串转并、74HC165 并转串、16x16LED 点阵显示模块、NE555 实验电路、板载步进电机（ME830 只集成有步进电机驱动电路，步进电机为选配）。ME830 拥有更为宽松的实验环境和资源，更加适合产品开发使用。

ME830 普通配置和 ME830 增强配置的差别在于增强配置搭配的是 POD52 专用仿真头（仿真不占用资源，可极速单步仿真，仿真功能和速度可媲美千元级以上的专业仿真器，特别适合开发产品使用），而 ME830 普通配置是搭配的 SST 单仿真芯片（仿真占用串口、8 字节堆栈和 T2 定时器，单步速度缓慢，只对于开发产品没有使用价值）

ME830 增强配置相比 ME830 标准配置多了 12864LCD(代字库)和 DS18B20 温度传感器，其他功能和配置完全一样！

ME830 和 ME830 都适合初学者使用。如果您只是用于一般的学习，可以选用 ME830 普通配置，如果您希望

将来能更加方便的用于产品开发，可以选用 ME830 增强配置或者选用功能最强的 ME830。ME830/850 光是内置的 ICE52 仿真器价值在 400 元以上，性价比是同类产品无可比拟的！

14. ME830/850 所采用的全开放式模块化设计有什么特点和优势？

ME830/850 所采用的全开放式模块化设计是我们在吸取多年的单片机实验仪（开发板）设计经验后，最终采用的一种方式：各硬件模块各自独立，并全部引出 I/O 口线，默认用短路帽进行连接，各模块和 CPU 的 I/O 口连接关系在 PCB 上有清晰的标识，在演示产品配套的上百个实验例程只需要拔插极少数短路帽，可以不看电路图就可以实验，对于初学者入门学习极其方便快捷。对于有经验的用户和开发人员，可以根据需要拔下短路帽后用随机附送的杜邦线任意组合线路。

目前绝大多数板子都是用的固定线路（有些板子还采用了电子开关来切换硬件资源，我们早期的 ME300B 就有采用过 4066 来切换资源，实践证明是不可取的，对于初学者不容易看明白硬件连接关系），固定线路的局限使得硬件可能束缚软件，不利于学习，更不适合开发。也有少数板子是模块化设计，但只能用连接线连接到 CPU 的 I/O 口，做任何一个实验都得反复拔插连接线，比较烦琐，如果做一个综合的实验，就需要用较多的连接线，太多的连接线飞在板上将会阻碍视线和操作。

所以相比其他开发板，ME830/850 这种科学的设计既避免了只能使用固定线路的局限，也解决了只能使用连接线的烦琐。全开放式设计使 ME830/ME850 可以配合任何单片机教材使用，也特别适合开发产品使用。

15. 如何得到技术支持以及可以得到哪些技术支持？

一些初次使用中的问题可以在技术支持网站（www.mcusj.com 单片机世界，或者 www.willar.com 伟纳电子网）论坛相关版块搜索，常见问题大多有解答。如果没有找到答案，建议您将所遇到的问题在论坛发帖提出，发帖时详细描述所遇到的问题，必要时配以图示，我们有安排工程师负责解答论坛技术问题，也有很多的热心网友和您一起学习交流。您也可以通过Email、电话联系。由于电话中不容易说明白问题，非紧急情况下建议通过电话之外的其他途径联系。

我们可提供与 ME830 产品、实验例程相关的技术咨询，但不负责用户项目和实验的技术支持。

附录 2 SST89E516RD 仿真芯片的使用

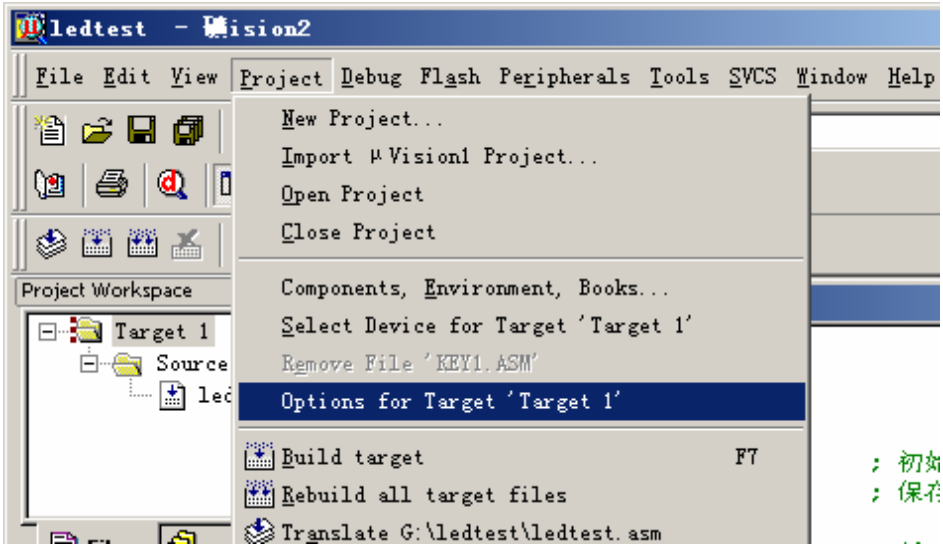
ME830 普通配置不含 POD52 仿真头，配送一片 SST89E516RD 单芯片，此芯片已经写入 SST 公司公版的的仿真监控程序，可以配合 keil 软件在 ME830 上进行硬件仿真调试仿真。SST 单仿真芯片相比 POD52 专业仿真头，具有较多的缺陷：SST 公版仿真芯片单步调试时要占用串口和 8 字节的 idata 堆栈，而且单步速度缓慢，没有暂停功能，没有脱机运行功能，只能设置 10 个断点，只适用于一般的学习和实验。如果要调试比较大的程序或者是开发产品使用，请使用 POD52 仿真头。

硬件设置方法:连接好 ME830 的串口线和 USB 线(此时 USB 线仅用于给 ME830 供电),将仿真芯片 SST89E516RD 插在 ME830 的锁紧座上锁紧，注意芯片缺口方向应朝串口座。打开电源开关，PWR 灯亮，则硬件设置完成。

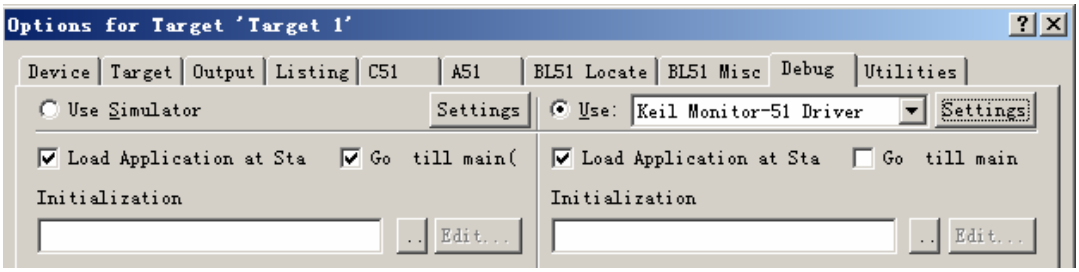
Keil 软件设置步骤如下：

1)打开一个要调试的 keil Project 文件(比如我们前面 4.2 章节建立的流水灯程序,在 Keil 菜单“Project”下点击“Open Project”，在弹出的“Select Project File”对话框中找到“E:\Demo”，点击打开“Demo.Uv2”，你也可以直接打开其他 Project 文件，ME830 光盘中全部例程均提供了 keil Project 文件，可以复制到你的电脑硬盘后使用。如果要建立自己的 keil project 文件，可以参考 4.2 章节：第一个 Keil C51 程序)。

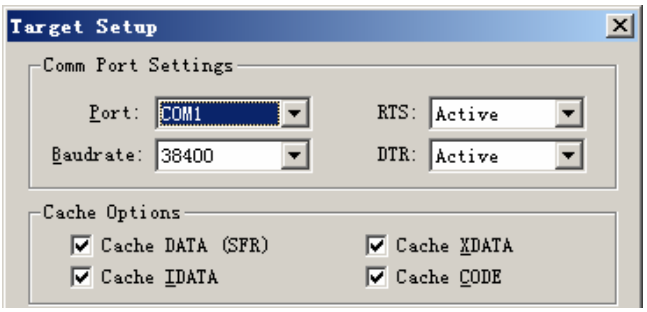
2）在菜单“Project”下面点击“Options for Target'Target 1'”；



3）在弹出的选项窗口中选择“Dubug”页，选择“Use:Keil Monitor-51 Driver”，选中“Load Application at Startup”，其他默认

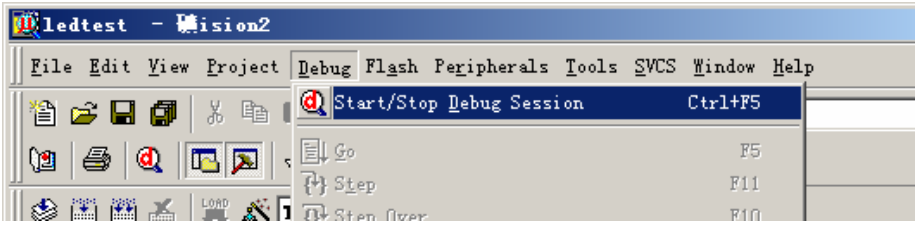


点击“Keil Monitor-51 Driver”右边的“Settings”按钮，将弹出如下所示的设置对话框。选择 ME830 所连接的串口（如选择错误将不能正常联机，如果你使用了 USB 转串口线，请在设备管理器中查看虚拟的串口端口号），设置波特率为 38400，其他可默认。

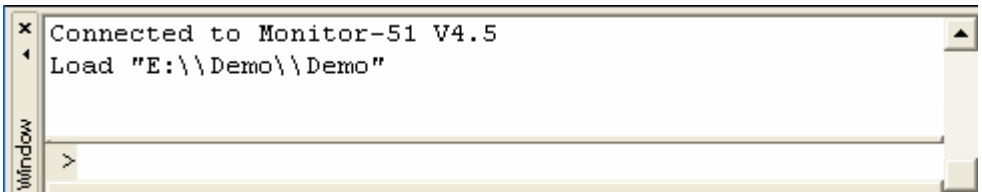


回到 Project－Options for Target，按确认按钮，完成设置。

4)开始仿真调试,先按一下 ME830 上的复位按钮,确保 ME830 复位,再点击菜单“Debug”下的“Start/Stop Debug Session”即可开始仿真调试。



当联机正确后，在 uVision 的 Output Window 将显示：Connected to Monitor-51 Vx.x(注：其中 Vx.x 为仿真监控程序版本号)，同时程序将通过串口下载到仿真模块内。



此时你就可以按单步按钮让程序单步执行，并可查看变量的变化、寄存器的变化。点击 Run 按钮，程序就进入全速运行状态，你可看到 P0 口和 P2 口的 16 个 LED 开始闪烁了。要退出全速运行状态，按一下 ME830 上的复位按钮 K9 即可。

注意：由于仿真调试需要使用串口，仿真调试时请将其它占用串口的软件如串口调试器关闭。