



普通高等教育“十一五”国家级规划教材

◎ 电子信息科学与工程类专业 规划教材

# 单片机的C语言程序 设计与应用

## ——基于Proteus仿真（第2版）

◎ 姜志海 赵艳雷 陈 松 编著



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

<http://www.phei.com.cn>

普通高等教育“十一五”国家级规划教材  
电子信息科学与工程类专业规划教材

# 单片机的 C 语言程序设计与应用

## ——基于 Proteus 仿真(第 2 版)

姜志海 赵艳雷 陈 松 编著

电子工业出版社  
Publishing House of Electronics Industry  
北京·BEIJING

## 内 容 简 介

本书是普通高等教育“十一五”国家级规划教材。本书以标准 C 语言和 MCS-51 单片机为基础, 全面系统地介绍单片机的 C 语言程序设计与应用的基本问题, 是一本重在应用、兼顾理论的实用教程。主要内容包括: 单片机的 C 语言概述、Keil 与 Proteus 应用基础、C51 语言程序设计基础、中断系统与定时/计数器及串行口的 C51 语言编程、键盘/显示的 C51 编程、模拟量接口技术的 C51 编程。本书包含大量单片机的 C 语言设计实例, 均可在 Keil 和 Proteus 软件平台上直接运行, 并提供电子课件、程序代码、仿真电路图和习题参考答案。

本书可作为高等学校电气与电子信息专业和计算机专业的本专科教材, 也可供从事单片机应用与产品开发工作的工程技术人员学习参考。

未经许可, 不得以任何方式复制或抄袭本书之部分或全部内容。  
版权所有, 侵权必究。

### 图书在版编目(CIP)数据

单片机的 C 语言程序设计与应用 / 姜志海, 赵艳雷, 陈松编著. —2 版. —北京: 电子工业出版社, 2011.7  
电子信息科学与工程类专业规划教材

ISBN 978-7-121-13826-3

I. ①单… II. ①姜… ②赵… ③陈… III. ①单片微型计算机—C 语言—程序设计—高等学校—教材  
IV. ①TP368.1②TP312

中国版本图书馆 CIP 数据核字 (2011) 第 110602 号

策划编辑: 王羽佳

责任编辑: 王羽佳 特约编辑: 王 崧

印 刷: 涿州市京南印刷厂

装 订: 涿州市桃园装订有限公司

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱 邮编: 100036

开 本: 787×1092 1/16 印张: 15.25 字数: 407 千字

印 次: 2011 年 7 月第 1 次印刷

印 数: 4000 册 定价: 29.00 元

凡所购买电子工业出版社图书有缺损问题, 请向购买书店调换。若书店售缺, 请与本社发行部联系, 联系及邮购电话: (010) 88254888。

质量投诉请发邮件至 [zltz@phei.com.cn](mailto:zltz@phei.com.cn), 盗版侵权举报请发邮件至 [dbqq@phei.com.cn](mailto:dbqq@phei.com.cn)。

服务热线: (010) 88258888。

# 前 言

单片机在嵌入式系统应用中占据很重要的地位。MCS-51 单片机在工业测量、控制领域中得到了广泛的应用。随着国内单片机开发工具研制水平的提高，现在的单片机仿真器普遍支持 C 语言程序的调试，为单片机使用 C 语言提供了便利条件，C 语言已成为举世公认的高效简洁而又贴近硬件的编程语言之一。

本书的主要特色在于，在介绍单片机的 C 语言设计的过程中，清晰地说明了单片机的所有功能，并对每项功能给出了实例代码，同时详细介绍了单片机的 C 语言开发与仿真环境的使用，透彻分析了单片机的 C 语言的语法和语义，以及开发过程中可能存在的问题和难点。

本书的另一个特点是突破了传统的软、硬件截然断裂的方法，使读者对嵌入式系统的开发有一个整体的了解。相信本书的这一特点会节省读者进入嵌入式 C 语言领域的时间，同时能够更清楚地认识应用系统开发的过程，深入理解单片机的 C 语言编程机制。

本书从构思和选材上，注意了尽量符合单片机应用系统的发展要求，并在 Proteus 软件平台上实现本书中所述的实例，保证系统设计原理和方法上的先进性。全书共分 7 章，主要内容包括：单片机的 C 语言概述、Keil 和 Proteus 应用基础、C51 语言程序设计基础、MCS-51 单片机中断系统与定时/计数器的 C51 编程、MCS-51 单片机串行口的 C51 编程、MCS-51 单片机键盘/显示器的 C51 编程、MCS-51 单片机模拟量接口的 C51 编程。每章都通过本章小结和习题来巩固所学的知识。全书通过大量实例阐述了单片机的 C 语言设计原理和技巧，所有实例均可在 Keil 和 Proteus 软件平台上运行。

本书提供有配套的电子课件、程序代码、仿真电路图和习题参考答案，请读者登录华信教育资源网 (<http://www.hxedu.com.cn>) 注册下载。

本书是一本专门讲解单片机的 C 语言编程的书，以由浅入深、相互贯穿、重点突出、文字叙述与典型代码实例相结合为原则，向每一位单片机、嵌入式爱好者和开发者全面介绍 C51 语言程序的编写。本书内容系统全面，论述深入浅出，循序渐进，可作为高等学校“单片机 C 语言设计与应用”课程的教学用书，也可以供从事单片机应用与产品开发工作的工程技术人员学习参考。

本书第 1、4、5 章由姜志海编写，第 6、7 章由赵艳雷编写，第 2、3 章由陈松编写，全书由姜志海负责整理、统稿。

本书在编写过程中得到了许多专家和同行的大力支持与热情帮助，他们对本书提出了许多建设性的建议和意见，在此一并表示衷心的感谢。

鉴于编者的水平有限，加之新的编程技术不断涌现，书中难免有不完善之处，恳请广大读者批评指正。反馈信息请发送至 [wyj@phei.com.cn](mailto:wyj@phei.com.cn)。

作 者  
2011 年 7 月

# 目 录

|   |      |
|---|------|
| 第 1 章 单片机的 C 语言概述 .....                   | (1)  |
| 1.1 C 语言与 MCS-51 单片机编程 .....              | (1)  |
| 1.1.1 单片机的 C 语言的特点 .....                  | (1)  |
| 1.1.2 单片机的 C 语言和标准 C 语言的比较 .....          | (2)  |
| 1.1.3 单片机的 C 语言与汇编语言的优势对比 .....           | (2)  |
| 1.2 C51 程序 .....                          | (6)  |
| 1.2.1 C51 的程序结构 .....                     | (6)  |
| 1.2.2 C51 编程规范及注意事项 .....                 | (7)  |
| 1.2.3 C51 的标识符与关键字 .....                  | (8)  |
| 1.3 C51 编程举例 .....                        | (10) |
| 本章小结 .....                                | (14) |
| 习题 .....                                  | (14) |
| 第 2 章 $\mu$ Vision2 与 Proteus 使用基础 .....  | (15) |
| 2.1 $\mu$ Vision2 集成开发环境 .....            | (15) |
| 2.1.1 File、Edit 和 View 菜单 .....           | (16) |
| 2.1.2 Project、Debug 和 Flash 菜单 .....      | (18) |
| 2.1.3 Peripherals、Tools 和 Window 菜单 ..... | (19) |
| 2.2 用 $\mu$ Vision2 建立与调试工程 .....         | (20) |
| 2.2.1 工程创建、设置、编译与链接 .....                 | (20) |
| 2.2.2 用 $\mu$ Vision2 调试工程 .....          | (25) |
| 2.2.3 C51 程序调试举例说明——HELLO.C .....         | (28) |
| 2.2.4 Keil C51 的调试技巧及举例 .....             | (30) |
| 2.3 Proteus 快速入门 .....                    | (34) |
| 2.3.1 集成 Proteus ISIS 环境 .....            | (34) |
| 2.3.2 电路原理图设计 .....                       | (38) |
| 2.4 Proteus 仿真工具介绍 .....                  | (43) |
| 2.4.1 探针 .....                            | (43) |
| 2.4.2 虚拟仪器 .....                          | (44) |
| 2.4.3 信号发生器 .....                         | (44) |
| 2.4.4 仿真图表 .....                          | (45) |
| 2.5 Proteus 软件中的 C51 程序运行与调试 .....        | (45) |
| 2.5.1 驱动的安装 .....                         | (45) |

|              |  |              |
|--------------|--|--------------|
| 2.5.2        | Keil 和 Proteus 的配置 .....                   | (46)         |
| 2.5.3        | Keil 和 Proteus 的调试过程 .....                 | (47)         |
| 本章小结         | .....                                      | (47)         |
| 习题           | .....                                      | (47)         |
| <b>第 3 章</b> | <b>单片机 C51 语言程序设计基础 .....</b>              | <b>(48)</b>  |
| 3.1          | C51 语言的数据 .....                            | (48)         |
| 3.1.1        | 数据类型 .....                                 | (48)         |
| 3.1.2        | 常量与变量 .....                                | (48)         |
| 3.1.3        | 数据存储类型 .....                               | (52)         |
| 3.1.4        | 局部变量与全局变量 .....                            | (56)         |
| 3.2          | C51 语言对单片机主要资源的控制 .....                    | (58)         |
| 3.2.1        | 特殊功能寄存器的 C51 语言定义 .....                    | (58)         |
| 3.2.2        | 绝对地址的访问 .....                              | (61)         |
| 3.2.3        | 位变量的 C51 语言定义 .....                        | (62)         |
| 3.3          | C51 语言的基本运算与流程控制语句 .....                   | (63)         |
| 3.3.1        | 基本运算 .....                                 | (63)         |
| 3.3.2        | 选择(分支)控制语句 .....                           | (68)         |
| 3.3.3        | 循环结构流程控制语句 .....                           | (73)         |
| 3.3.4        | break 语句、continue 语句和 goto 语句 .....        | (77)         |
| 3.4          | C51 语言的数组与指针 .....                         | (78)         |
| 3.4.1        | 数组 .....                                   | (79)         |
| 3.4.2        | 指针 .....                                   | (82)         |
| 3.5          | C51 语言函数 .....                             | (85)         |
| 3.5.1        | 函数的基本问题 .....                              | (85)         |
| 3.5.2        | 中断函数 .....                                 | (91)         |
| 3.5.3        | C51 语言的库函数 .....                           | (91)         |
| 3.6          | C51 的预处理命令及汇编语句的嵌入 .....                   | (94)         |
| 3.6.1        | 文件包含 .....                                 | (94)         |
| 3.6.2        | 宏定义指令 .....                                | (95)         |
| 3.6.3        | 条件编译 .....                                 | (96)         |
| 3.6.4        | C51 中汇编语句的嵌入 .....                         | (98)         |
| 本章小结         | .....                                      | (99)         |
| 习题           | .....                                      | (99)         |
| <b>第 4 章</b> | <b>MCS-51 系列单片机中断与定时/计数器的 C51 编程 .....</b> | <b>(100)</b> |
| 4.1          | 中断系统基本知识 .....                             | (100)        |
| 4.1.1        | 中断系统结构与中断控制 .....                          | (100)        |

|              |  |              |
|--------------|--|--------------|
| 4.1.2        | 中断优先级与中断请求的撤除 .....                    | (102)        |
| 4.1.3        | 中断函数的结构形式 .....                        | (103)        |
| 4.2          | 外部中断源的 C51 编程 .....                    | (103)        |
| 4.2.1        | 外部中断源初始化 .....                         | (103)        |
| 4.2.2        | 外部中断源的 C51 编程示例 .....                  | (104)        |
| 4.3          | 定时/计数器基本知识 .....                       | (107)        |
| 4.3.1        | 定时/计数器的结构与控制 .....                     | (107)        |
| 4.3.2        | 工作方式与工作模式 .....                        | (108)        |
| 4.3.3        | 定时/计数器的初始化 .....                       | (111)        |
| 4.4          | 定时/计数器不同工作模式下的 C51 编程 .....            | (113)        |
| 4.4.1        | 工作模式 1 的 C51 编程 .....                  | (114)        |
| 4.4.2        | 工作模式 2 的 C51 编程 .....                  | (115)        |
| 4.4.3        | 工作模式 3 的 C51 编程 .....                  | (115)        |
| 4.5          | 中断与定时/计数器综合编程示例 .....                  | (116)        |
|              | 本章小结 .....                             | (133)        |
|              | 习题 .....                               | (134)        |
| <b>第 5 章</b> | <b>MCS-51 系列单片机串行口的 C51 编程 .....</b>   | <b>(135)</b> |
| 5.1          | 串行口基础知识 .....                          | (135)        |
| 5.1.1        | 串行口的结构 .....                           | (135)        |
| 5.1.2        | 有关的寄存器 .....                           | (135)        |
| 5.1.3        | 串行口的工作方式 .....                         | (137)        |
| 5.2          | 不同工作方式下的 C51 编程 .....                  | (138)        |
| 5.2.1        | 工作方式 0 的 C51 程序设计 .....                | (138)        |
| 5.2.2        | 工作方式 1 的 C51 程序设计 .....                | (142)        |
| 5.2.3        | 工作方式 2 的 C51 程序设计 .....                | (146)        |
| 5.2.4        | 工作方式 3 的 C51 程序设计 .....                | (150)        |
| 5.3          | 串行口的 C51 编程应用示例 .....                  | (154)        |
|              | 本章小结 .....                             | (165)        |
|              | 习题 .....                               | (166)        |
| <b>第 6 章</b> | <b>MCS-51 系列单片机键盘/显示的 C51 编程 .....</b> | <b>(167)</b> |
| 6.1          | 七段数码显示器的 C51 编程 .....                  | (167)        |
| 6.1.1        | 七段 LED 显示器基础知识 .....                   | (167)        |
| 6.1.2        | 七段数码静态显示的 C51 编程 .....                 | (169)        |
| 6.1.3        | 七段数码动态显示的 C51 编程 .....                 | (171)        |
| 6.1.4        | 串行口控制的 LED 显示的 C51 编程 .....            | (176)        |
| 6.2          | 键盘接口的 C51 编程 .....                     | (177)        |

|              |   |              |
|--------------|---|--------------|
| 6.2.1        | 键盘的基础知识 .....                           | (177)        |
| 6.2.2        | 独立式键盘接口的 C51 编程 .....                   | (178)        |
| 6.2.3        | 矩阵键盘的 C51 编程 .....                      | (180)        |
| 6.3          | 点阵 LED 显示器接口的 C51 编程 .....              | (185)        |
| 6.3.1        | 点阵 LED 显示器基础知识 .....                    | (185)        |
| 6.3.2        | 一个 5×7 点阵显示一个字符的 C51 编程 .....           | (186)        |
| 6.3.3        | 一个 8×8 点阵显示字符串的 C51 编程 .....            | (187)        |
| 6.3.4        | 两个 8×8 点阵显示字符串的 C51 编程 .....            | (189)        |
| 6.4          | 液晶显示器的 C51 编程 .....                     | (190)        |
| 6.4.1        | LCD 基础知识 .....                          | (190)        |
| 6.4.2        | 字符型 LCD1602 应用的 C51 编程 .....            | (192)        |
|              | 本章小结 .....                              | (200)        |
|              | 习题 .....                                | (200)        |
| <b>第 7 章</b> | <b>MCS-51 系列单片机模拟量接口的 C51 编程 .....</b>  | <b>(201)</b> |
| 7.1          | 8 位并行 D/A 转换器 DAC0832 的 C51 编程 .....    | (201)        |
| 7.1.1        | DAC0832 基础知识 .....                      | (201)        |
| 7.1.2        | DAC0832 单缓冲和双缓冲的 C51 编程 .....           | (202)        |
| 7.1.3        | DAC0832 输出各种波形的 C51 编程 .....            | (206)        |
| 7.2          | 12 位并行 D/A 芯片 DAC1208 的 C51 编程 .....    | (210)        |
| 7.2.1        | 12 位并行 DAC1208 简介 .....                 | (210)        |
| 7.2.2        | DAC1208 的 C51 编程 .....                  | (211)        |
| 7.3          | 8 位 A/D 转换器 ADC0808/0809 的 C51 编程 ..... | (212)        |
| 7.3.1        | ADC0808/0809 简介 .....                   | (212)        |
| 7.3.2        | ADC0809 的 C51 编程 .....                  | (213)        |
| 7.4          | 12 位串行 A/D 转换器 TLC2543 的 C51 编程 .....   | (215)        |
| 7.4.1        | TLC2543 简介 .....                        | (216)        |
| 7.4.2        | TLC2543 的 C51 编程 .....                  | (217)        |
|              | 本章小结 .....                              | (220)        |
|              | 习题 .....                                | (220)        |
| <b>附录 A</b>  | <b>MCS-51 系列单片机简介 .....</b>             | <b>(221)</b> |
| <b>附录 B</b>  | <b>8255A 简介 .....</b>                   | <b>(228)</b> |
| <b>附录 C</b>  | <b>ASCII 码字符表 .....</b>                 | <b>(232)</b> |
| <b>附录 D</b>  | <b>Proteus 常用分离器件名称 .....</b>           | <b>(233)</b> |
| <b>参考文献</b>  | <b>.....</b>                            | <b>(234)</b> |



# 第 1 章 单片机的 C 语言概述

我们都知道，在单片机应用系统开发过程中，软件编程占有非常重要的地位。尤其是随着单片机技术的发展，嵌入式系统的推广和应用，硬件的集成化程度越来越高，同时对软件编程的要求也越来越高。这就要求单片机开发人员能在短时间内编写出执行效率高、运行可靠的代码。同时，由于实际系统的日趋复杂，对使用代码的规范性、模块化的要求越来越高，要方便多个工程师以软件工程的形式进行协同开发。在这种形势下，仅靠单片机在推广应用的初期使用的汇编语言来进行软件开发，是远远不够的。C 语言是近年来在国内外普遍使用的一种程序设计语言。C 语言能直接对计算机硬件进行操作，既有高级语言的特点，又有汇编语言的特点，因此在单片机应用系统开发过程中得到了非常广泛的应用。

在单片机应用系统设计与开发过程中，只要简单地熟悉相应单片机的硬件结构，利用 C 语言作为编程语言，就可以大大缩短开发周期。本章主要对单片机的 C 语言的基本问题进行概括的说明。

## 1.1 C 语言与 MCS-51 单片机编程

嵌入式单片机在开发过程中的编程语言主要有汇编语言和 C 语言。汇编语言作为传统的嵌入式系统的编程语言，已经不能满足实际需要了，而 C 语言的结构化和高效性满足了这样的需要，成为电子工程师在进行嵌入式系统编程时的首选语言，并得以广泛应用。尤其是 C 语言编译系统的发展，更加促进了 C 语言的应用。1985 年出现了针对 8051 单片机的 C51 编译器，进而又出现了其他流行的嵌入式处理器系统，如 196 系列、PIC 系列、MOTORAL 系列、MSP430 系列、AD 公司和 TI 公司的 DSP 系列的 C 语言编译系统，以及丰富的 C 语言库函数。本书主要讨论 8 位嵌入式单片机——MCS-51 单片机及其派生产品的 C 语言编程问题，简称 C51 的程序设计。

### 1.1.1 单片机的 C 语言的特点

单片机的 C 语言的特点主要体现在以下几个方面：

- ① 无须了解机器硬件及其指令系统，只需初步了解 MCS-51 的存储器结构；
- ② C51 语言能方便地管理内部寄存器的分配、不同存储器的寻址和数据类型等细节问题，但对硬件控制有限，而汇编语言可以完全控制硬件资源；
- ③ C51 语言在小应用程序中产生的代码量大，执行速度慢，但在较大的程序中代码效率高；
- ④ C51 语言程序由若干函数组成，具有良好的模块化结构，便于改进和扩充；
- ⑤ C51 语言程序具有良好的可读性和可维护性，而汇编语言在大应用程序开发中，开发难度增加，可读性差；
- ⑥ C51 语言有丰富的库函数，可以大大减少用户的编程量，显著缩短编程与调试时间，大大提高软件开发效率；

⑦ 使用汇编语言编制的程序,当机型改变时,无法直接移植使用,而 C 语言程序是面向用户的程序设计语言,能在不同类型的机器上运行,可移植性好。

### 1.1.2 单片机的 C 语言和标准 C 语言的比较

标准 C 语言,或称为 ANSI C 语言。单片机的 C 语言和标准 C 语言之间有许多相同的地方,但也有其自身的一些特点。不同的嵌入式 C 语言编译系统之所以与 ANSI C 语言有不同的地方,主要是由于它们所针对的硬件系统不同,对于 MCS-51 系列单片机,称为 C51 语言。C51 语言与标准 C 语言的不同点主要体现在以下几方面。

#### (1) 库函数

标准 C 语言定义的库函数是按照通用微型计算机来定义的,而 C51 语言中的库函数是按 MCS-51 单片机的应用情况来定义的。

#### (2) 数据类型

在 C51 语言中增加了几种针对 MCS-51 单片机的特有数据类型。例如, MCS-51 系列单片机包含位操作空间和丰富的位操作指令,因此, C51 语言与 ANSI C 语言相比多了一种位类型,从而使其能同汇编语言一样,灵活地进行位指令操作。

#### (3) 变量的存储模式

C51 语言中变量的存储模式与 MCS-51 单片机的存储器紧密相关。从数据存储类型上, MCS-51 系列单片机有片内、片外程序存储器,片内、片外数据存储器。在片内程序存储器中,又有直接寻址区和间接寻址区之分,其分别对应 code、data、xdata、idata,以及根据 MCS-51 系列单片机特点而设定的 pdata 类型。使用不同存储器将会影响程序执行的效率,不同的模式对应不同的硬件系统和不同的编译结果。但 ANSI C 语言对存储模式要求不高。

#### (4) 输入/输出

C51 语言中的输入/输出是通过 MCS-51 串行口来完成的,输入/输出指令执行前必须对串行口进行初始化。

#### (5) 函数使用

C51 语言中有专门的中断函数。

### 1.1.3 单片机的 C 语言与汇编语言的优势对比

在国内,汇编语言在单片机开发过程中是比较流行的开发工具。长期以来对编译效率的偏见,以及不少程序员对使用汇编语言开发硬件环境的习惯性,使得 C 语言在很多地方遭到冷落。优秀的程序员写出的汇编语言程序的确有执行效率高的优点,但汇编语言其可移植性和可读性差的特点,使得使用其开发出来的产品在维护和功能升级时确有极大的困难,从而导致整个系统的可靠性和可维护性比较差。而使用 C 语言进行嵌入式系统的开发,有着汇编语言不可比拟的优势。

#### (1) 编程调试灵活方便

C 语言编程灵活,当前几乎所有的嵌入式系统都有相应的 C 语言级别的仿真调试系统,调试十分方便。

#### (2) 生成的代码编译效率高

当前较好的 C 编译系统编译出来的代码效率只比直接使用汇编语言低 20%,如果使用优化编译选项甚至可以更低。

### (3) 模块化开发

目前的软硬件开发都向模块化、可复用性的目标集中。不管是硬件还是软件，都希望其有比较通用的接口，在以后的开发中如果需要通过实现相同或者相近的功能，就可以直接使用以前开发过的模块，尽量不做或少做改动，以减少重复劳动。如果使用C语言开发，那么数据交换可以方便地通过约定实现，有利于多人协同进行大项目的合作开发。同时C语言的模块化开发方式使开发出来的程序模块可以不经修改而直接被其他项目使用，这样就可以很好地利用已有的大量C语言程序资源与丰富的库函数，从而最大程度地实现资源共享。

### (4) 可移植性好

由于不同系列嵌入式系统的C语言编译工具都是以标准C语言作为基础进行开发的，因此一种C语言环境下所编写的C语言程序，只需要将部分与硬件相关的地方和编译链接的参数进行适当修改，就可以方便地移植到另一种系列上。例如，在C51下编写的程序通过改写头文件及少量的程序行，就可以方便地移植到196或PIC系列上。也就是说，基于C语言环境下的嵌入式系统能基本实现平台的无关性。

### (5) 便于项目的维护

用C语言开发的代码便于开发小组计划项目、灵活管理、分工合作及后期维护，基本可以杜绝因开发人员变化而给项目进度、后期维护或升级所带来的影响，从而保证整个系统的品质、可靠性及可升级性。

下面通过一个例子对汇编语言和C语言进行比较。

**【例 1-1】** 将外部数据存储器的000BH和000CH单元的内容相互交换。

用汇编语言编程，源程序如下：

```
ORG      0000H
MOV      DPTR,#000BH
MOVX     A,@DPTR           ;将000BH内容读入A
MOV      R7,A             ;暂存000BH内容
INC      DPTR
MOVX     A,@DPTR           ;将000CH内容读入A
MOV      DPTR,#000BH
MOVX     @DPTR,A
INC      DPTR
MOV      A,R7
MOVX     @DPTR,A
SJMP     $
END
```

用C语言编程，C51源程序如下（注：C51语言对地址的指示可以采用指针变量，也可以引用头文件absacc.h作为绝对地址访问。下面的程序采用绝对地址访问方法）：

```
#include<absacc.h>

void main(void)
{
    char c;
    c=XBYTE[11];
    XBYTE[11]=XBYTE[12];
    XBYTE[12]=c;
    while(1);
}
```

上面的程序经过编译,生成的反汇编程序如下:

```

0x0000 020013 LJMP     STARTUP1(C:0013)           ;跳转
0x0003 90000B MOV     DPTR,#0x000B
0x0006 E0      MOVX   A,@DPTR
0x0007 FF      MOV     R7,A
0x0008 A3      INC     DPTR
0x0009 E0      MOVX   A,@DPTR
0x000A 90000B MOV     DPTR,#0x000B
0x000D F0      MOVX   @DPTR,A
0x000E A3      INC     DPTR
0x000F EF      MOV     A,R7
0x0010 F0      MOVX   @DPTR,A
0x0011 80FE    SJMP   C:0011
0x0013 787F    MOV     R0,#0x7F           ;以下是清零部分
0x0015 E4      CLR     A
0x0016 F6      MOV     @R0,A
0x0017 D8FD    DJNZ   R0,IDATALOOP(C:0016)
0x0019 758107 MOV     SP(0x81),#0x07
0x001C 020003 LJMP   main(C:0003)

```

对照 C 语言编写的源程序与反汇编程序,可以看出:

① 进入 C 语言程序后,首先将 RAM 地址为 7FH 开始的 128 个单元清零,然后置 SP 为 07,因此如果要对内部 RAM 置初值,那么一定是在执行了一条 C 语言语句之后;

② 对于 C 语言程序设定的变量,C51 编译器自行安排寄存器或存储器作为参数传递区,通常在 R0~R7(一组或两组,根据参数多少而定),因此如果对具体地址置数,则应避免 R0~R7 这些地址;

③ 如果不特别指定变量的存储类型,那么变量通常被安排在内部 RAM 区。

下面再给出几个 C 语言和汇编语言对照的例子。

**【例 1-2】** 二进制数转换成十进制数(BCD 码)。将累加器 A 中给定的二进制数,转换成 3 个十进制数(BCD 码),并存入 Result 开始的 3 个单元。

汇编语言源程序如下:

```

Result      EQU     20H
             ORG     0000H
             LJMP   START
START:       MOV     SP,#60H           ;主程序
             MOV     A,#123
             LCALL  BINTOBCD
             SJMP   $
BINTOBCD:   MOV     B,#100           ;设置转换子程序
             DIV    AB
             MOV    Result,A         ;除以 100 得百位数
             MOV    A,B
             MOV    B,#10
             DIV    AB
             MOV    Result+1,A      ;余数除以 10 得十位数

```

```

MOV     Result+2,B           ;余数为个位数
RET
END

```

调试结果:

片内 RAM 20H、21H、22H 中的数值分别为 01H、02H、03H。

用 C 语言编程, C51 源程序如下:

```

void main(void)
{
    unsigned char Result[3];
    unsigned char Number;
    Number=123;
    Result[0]=Number/100;           //除以 100 得百位数
    Result[1]=(Number%100)/10;     //余数除以 10 得十位数
    Result[2]=Number%10;          //余数为个位数
    while(1);                       //等待暂停
}

```

调试结果:

片内 RAM 07H 中的数据为 7BH, 08H、09H、0AH 中的数据分别为 01H、02H、03H。

**【例 1-3】** 二进制数转换成 ASCII 码程序。将累加器 A 中的内容分为两个 ASCII 码, 并存入 Result 开始的两个单元。

汇编语言源程序如下:

```

Result      EQU      20H
              ORG      0000H
              LJMP     START
START:       MOV      SP, #40H
              MOV      A, #00011010B
              LCALL   BINTOHEX
              SJMP    $
BINTOHEX:   MOV      DPTR, #ASCIIITAB
              MOV      B, A
              SWAP    A
              ANL     A, #0FH           ;取 A 的高 4 位
              MOVC   A, @A+DPTR
              MOV     Result, A
              MOV     A, B
              ANL     A, #0FH           ;取 A 的低 4 位
              MOVC   A, @A+DPTR
              MOV     Result+1, A
              RET
ASCIIITAB:  DB      '0123456789ABCDEF'
              END

```

调试结果:

片内 RAM 20H、21H 中的数据分别为 31H、41H。

用 C 语言编程, C51 源程序如下:

```

code unsigned char ASCIIITAB[16]="0123456789ABCDEF";
void main(void)

```

```

{   unsigned char Result[2];
    unsigned char Number;
    Number=0x1a;
    Result[0]=ASCIITAB[Number/16];           //高 4 位
    Result[1]=ASCIITAB[Number&0x0f];       //低 4 位
    while(1);
}

```

调试结果:

片内 RAM07H 中的数据为 1AH, 08H、09H 中的数据分别为 31H、41H。

## 1.2 C51 程序

### 1.2.1 C51 的程序结构

单片机 C51 语言继承了 C 语言的特点, 其程序结构与一般 C 语言的程序结构没有差别。C51 源程序文件的扩展名为“.c”, 如 test.c、function.c 等。每个 C51 源程序包含一个名为 main() 的主函数, C51 程序的执行总是从 main() 函数开始的。当主函数的所有语句执行完毕, 则程序执行结束。例 1-4 是一个典型的 C51 源程序的例子。

#### 【例 1-4】 C51 源程序示例

```

#include<reg52.h>           //预处理命令, reg52.h 是一个库文件
#include<stdio.h>          //stdio.h 是一个库文件
void Function1(void);      //自定义函数 Function1 声明
unsigned int ch;           //全局变量声明

void main(void)            //主函数
{   SCON=0x50;              //SCON:模式 1, 8bit 异步串行口通信
    TMOD=0x20;              //TMOD:定时器 1 为模式 2, 8bit 自动装载方式
    TH1=221;                //TH1:1200bit/s 的装载值, 16MHz
    TR1=1;                  //TR1:timer1 运行
    TI=1;                   //TI:设置为 1, 以发送第一个字节
                                //以上 5 条语句是为串行口调试设置的

    while(ch<=5)
    {   Function1();         //调用自定义函数
        printf("char=%d\n", ch); //程序语句
    }
    while(1);
}

void Function1(void)       //自定义函数 Function1
{   unsigned char ps;      //自定义函数内部变量声明
    ps=1;
    ch=ch+ps;
}

```

在串行口窗口 Serial#1 中, 输出结果:

```
char=1
char=2
char=3
char=4
char=5
char=6
```

从上面的例子可以看出，一个典型的 C51 源程序包含预处理命令、自定义函数声明、主函数 `main()` 和自定义函数。这几部分与 C 语言的程序结构完全类似，各部分的功能如下。

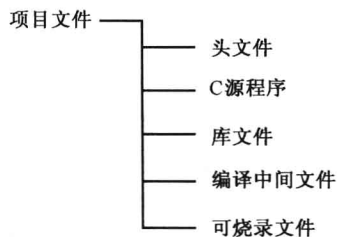
① 预处理命令部分常用 `#include` 命令来包含一些程序中用到的头文件。这些头文件中包含了一些库函数，以及其他函数的声明与定义。

② 自定义函数声明部分用来声明源程序中自定义的函数。

③ 主函数 `main()` 是整个 C51 程序的入口。不论 `main()` 函数位于程序代码中的哪个位置，C51 程序总是从 `main()` 函数开始执行的。

④ 自定义函数部分是 C51 源程序中用到的自定义函数的函数体。

除了扩展名为 “.c” 的源程序文件外，C51 程序还支持扩展名为 “.h” 的头文件以及扩展名为 “.lib” 的库文件等。在一般的编译系统中，通常以项目（工程）结构来管理复杂的 C51 程序文件。例如 Keil  $\mu$ Vision2 编译环境中，整个项目结构如图 1-1 所示。



在这里，整个项目由项目文件管理，项目文件扩展名为 “.UV2”。整个工程项目中可以包含如下几类文件。

图 1-1 Keil  $\mu$ Vision2 项目结构示意图

① 头文件用来包含一些库函数，系统变量声明并将不同的 C 文件连接起来。

② C 源文件是 C51 程序的主要部分，用来实现特定的功能。C 源文件可以有一个，也可以按照不同的功能分成多个，但所有这些 C 源文件中有且仅有一个可以包含一个 `main()` 主函数。

③ 库文件是实现特定功能的函数库，供 C 源文件调用。

④ 编译中间文件是源程序在编译链接过程中生成的中间文件，其中包含了文件编译调试的信息。

⑤ 可烧录文件是编译系统生成的可以烧录到单片机内部供执行的文件，类似于 “.exe” 可执行文件。在 C51 语言中，一般扩展名为 “.hex” 或者 “.bin” 等。

在这些文件中，C 源文件是必需的，其他文件可以根据用户实际的需要而使用。

## 1.2.2 C51 编程规范及注意事项

在学习任何一种编程语言时，按照一定的规范培养良好的编程习惯很重要。良好的编程规范可以帮助开发人员理清思路、方便整理代码，同时也便于他人阅读、理解，以促进代码的交流。在进行 C51 语言程序设计时，应该注意以下几方面的编程规范。

### 1. 注释

任何编程语言都支持注释语句。注释语句只对代码起到功能描述的作用，在实际的编译链接过程中不起作用。在 C51 语言中可以通过两种方式来表示注释内容。

(1) 用“//”开头来注释一行

如“//变量声明”。本方法简明、方便，“//”符号可以在一行的开始，这样整行都表示注释内容；“//”也可以在某行的执行语句后面，“//”符号后面的内容是对该语句的注释内容。

(2) 用“/\*”符号开头，并以“\*/”符号结束

如用“/\*声明整型变量 ch\*/”。本方法灵活多变，可以注释多行，如

```
/*
*****
/*      Function.c: 使用 C51 编译器的自定义功能函数库      */
*****
*/
```

用户也可以在程序语句的内部进行注释，示例如下：

```
printf("ch=%d\n", /*整型变量 ch*/ch);
```

一个好的 C51 源程序应该添加必要的注释内容。这样，可以增加程序的可读性，方便日后修改或者与他人的代码交流。注释内容一般包括程序的功能、实现方式、自定义函数的功能描述、语句的功能等。

## 2. 命名

在进行程序设计时，经常需要自定义一些函数或变量。一般来说，只要符合 C51 命名规范即可通过编译。但是，为了便于源程序的理解和交流，在进行命名时应注意以下几点。

① 自定义函数或者变量的名称最好能反映该函数或变量的功能与用途。因此，需要选用有意义的单词或者字母组合来表示。例如，MAX 表示最大值、MIN 表示最小值等。

② 变量名通常加上表示数据类型的前缀，例如 ucSendData 的前缀 uc 表示 unsigned char。

③ 在命名时不要和系统保留的标识符及关键字产生冲突或者歧义。

## 3. 格式

为了程序阅读方便，在进行 C51 程序设计时，在程序结构及语句书写格式方面应注意以下几点。

① 虽然 C51 语言对 main() 函数放置的位置没有限定，但为了程序阅读的方便，最好把它放在所有自定义函数的前面，即依次为头文件声明、自定义函数及全局变量声明、main() 函数、自定义函数。

② C51 语句可以写在一行上，也可以写在多行上。为了程序理解的方便，最好将每条语句单独写在一行上，并加以注释。有时，共同执行某个功能的几条相连的语句可以放置在一行上。

③ 对于源程序文件不同结构部分之间要留有空行。例如，头文件声明、自定义函数声明、main() 函数，以及自定义函数之间均要空一行，来明显区分不同的结构。

④ 对于 if、while 等块结构语句中的“{”和“}”，要配对对齐，以便于程序阅读时能够理解该结构的起始和结束。

⑤ 源代码安排时，可以通过适当的空格和制表位来实现代码对齐。

以上是一些常用的编程规范，读者可以参考借鉴。

### 1.2.3 C51 的标识符与关键字

标识符和关键字是一种编程语言最基本的组成部分，C51 语言同样支持自定义的标识符及系统保留的关键字。在进行 C51 程序设计时，需要了解标识符和关键字的使用规则。



## 1. 标识符

标识符常用来声明某个对象的名称，如变量和常量的声明、数组和结构的声明、自定义函数的声明，以及数据类型的声明等。示例如下：

```
int count;
void Function1();
```

在上面的例子中，`count` 为整型变量的标识符，`Function1` 为自定义函数的标识符。

在 C51 语言中，标识符可以由字母、数字（0~9）或者下画线“\_”组成，最多可支持 32 个字符。C51 标识符的第一个字符必须是字母或者下画线“\_”，例如 `unt`、`ch_1` 等都是正确的标识符，而 `5count` 则是错误的标识符。另外，C51 的标识符区分大小写，例如 `count` 和 `COUNT` 代表两个不同的标识符。使用标识符时应注意以下几点。

① 在命名 C51 标识符时，需要能够清楚地表达其功能含义，这样有助于阅读和理解源程序。

② C51 的标识符原则上可以使用下画线开头，但有些编译系统的专用标识符或者预定义项是以下画线开头的。为了程序的兼容性和可移植性，建议一般不使用下画线开头来命名标识符。

③ 尽量不要使用过长的标识符，以便于使用和程序理解方便。

④ 自定义的 C51 标识符不能使用 C51 语言保留的关键字，也不能和用户已使用的函数名或 C51 库函数同名。例如 `char` 是关键字，所以不能作为标识符使用。

## 2. 关键字

关键字是 C51 语言的重要组成部分，是 C51 编译器已定义保留的专用特殊标识符，有时也称为保留字。这些关键字通常有固定的名称和功能，如 `int`、`if`、`for`、`do`、`while`、`case` 等。C51 语言中常用的关键字如表 1-1 所示。

表 1-1 C51 语言中常用的关键字

| 类别                | 关键字                   | 类型                        | 用途说明                             |
|-------------------|-----------------------|---------------------------|----------------------------------|
| ANSI C 标准关键字      | <code>auto</code>     | 存储种类说明                    | 常用于声明局部变量，默认值为此类型                |
|                   | <code>break</code>    | 程序语句                      | 无条件退出循环程序最内层循环                   |
|                   | <code>case</code>     | 程序语句                      | <code>switch</code> 选择语句中的选择项    |
|                   | <code>char</code>     | 数据类型说明                    | 单字节整型数据或字符型数据                    |
|                   | <code>const</code>    | 存储类型说明                    | 定义不可更改的常量值                       |
|                   | <code>continue</code> | 程序语句                      | 中断本次循环，并转向下一次循环                  |
|                   | <code>default</code>  | 程序语句                      | <code>switch</code> 选择语句中的默认选择项  |
|                   | <code>do</code>       | 程序语句                      | 用以构成 <code>do...while</code> 循环  |
|                   | <code>double</code>   | 数据类型说明                    | 声明双精度浮点型数据                       |
|                   | <code>else</code>     | 程序语句                      | 用以构成 <code>if...else</code> 选择结构 |
|                   | <code>enum</code>     | 数据类型说明                    | 枚举                               |
|                   | <code>extern</code>   | 存储种类说明                    | 在其他程序模块中说明了的全局变量                 |
|                   | <code>float</code>    | 数据类型说明                    | 定义单精度浮点型数据                       |
|                   | <code>for</code>      | 程序语句                      | 构成 <code>for</code> 循环语句         |
| <code>goto</code> | 程序语句                  | 构成 <code>goto</code> 转移语句 |                                  |