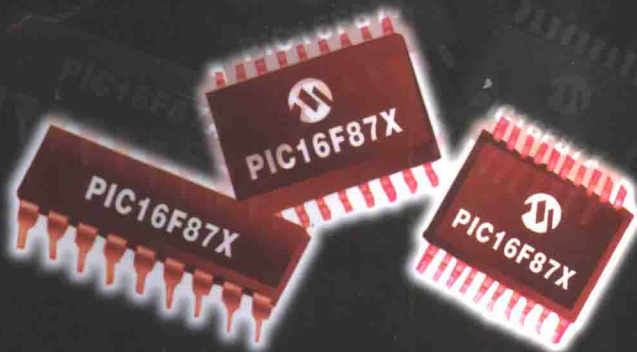




# PIC16F87X 单片机 实用软件与接口技术

## —— C 语言及其应用



刘和平 等编著



北京航空航天大学出版社  
<http://www.buaapress.com.cn>



# PIC16F87X 单片机实用软件与接口技术

## ——C 语言及其应用

刘和平 等编著

北京航空航天大学出版社

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

## 内 容 简 介

本书讨论了 PIC16F87X 系列单片机的 C 语言程序编程方法。书中介绍了大量的 C 语言程序例程,涉及到 PIC16F87X 单片机的各个功能模块的编程应用;给出了应用实例的电路原理图和源程序清单,所有程序均在实验板上调试通过,并配有光盘。

本书与北京航空航天大学出版社出版的《PIC16F87X 单片机实用软件与接口技术——汇编语言及其应用》构成姊妹篇。该姊妹篇是单片机开发者和初学者的很好的参考书,也可作为大学本科学单片机原理及应用课程的实验指导书。

## 图书在版编目(CIP)数据

PIC16F87X 单片机实用软件与接口技术——C 语言及其应用 / 刘和平等编著. —北京:北京航空航天大学出版社, 2002. 4

ISBN 7-81077-169-8

I. P… II. 刘… III. C 语言—程序设计  
IV. TP312

中国版本图书馆 CIP 数据核字(2002)第 019673 号

## PIC16F87X 单片机实用软件与接口技术 ——C 语言及其应用

刘和平 等编著

责任编辑 王 瑛

\*

北京航空航天大学出版社出版发行

北京市海淀区学院路 37 号(100083) 发行部电话:(010)82317024 传真:(010)82328026

http://www.buaapress.com.cn

E-mail:pressell@publica.bj.cninfo.net

河北省涿州市新华印刷厂印装 各地书店经销

\*

开本:787×960 1/16 印张:17 字数:380 千字

2002 年 4 月第 1 版 2002 年 4 月第 1 次印刷 印数:5 000 册

ISBN 7-81077-169-8/TP·093 定价:32.00 元

## 本书编委会

主编：刘和平

编委：冼成瑜 杨利辉 杨立勇  
崔晶 吴俊 黄开长

# 前 言

目前市场上应用广泛的是 8 位单片机。美国微芯公司作为世界上 8 位单片机的第 2 大生产商,推出了 CMOS 8 位 PIC 系列单片机。该系列芯片采用精简指令集(RISC)、哈佛总线结构、2 级流水线取指令方式,具有实用、低价、指令集小、简单易学、低功耗、高速度、体积小、功能强等特点,体现了单片机发展的一种新趋势,深受用户的欢迎,已经逐渐成为世界单片机的新潮流。

本书内容以微芯公司的采用 14 位 RISC 指令集的中级产品 PIC16F87X 为主。由于芯片内含 A/D、内部 EEPROM 存储器、比较输出、捕捉输入、PWM 输出、I<sup>2</sup>C 和 SPI 接口、异步串行通信(USART)接口、LCD 驱动、FLASH 程序存储器读写等许多功能,对初学者来说有一定的难度;而且这方面的参考资料又很少,更没有介绍应用实例和应用程序库的书,给广大的使用者带来了困难。

对于一般涉及单片机开发和应用工作的人员,使用汇编语言编制短小程序较容易;但要编写大程序或者多人合作编程,以及编程的后期维护,将是较困难的事情。

针对微芯公司的采用 14 位 RISC 指令集的中级产品 PIC16F87X,微芯公司的第 3 方提供了几种 C 语言开发工具,HI\_TECH PICC 就是其中的一种。本书以 HI\_TECH PICC 为基础,介绍 PIC 中级产品的 C 语言基本知识、软件开发环境、C 语言函数库及 C 语言编程实例。

本书以重庆大学-美国微芯公司 PIC 单片机实验室开发的实验板为对象,以单片机的各种功能模块为线索,通过给出实验板的硬件连接方式和 C 语言编程实例进行讲解,从易到难,循序渐进,逐步深入。

全书共分 15 章。第 1 章,PICC C 语言基础和特点;第 2 章,PICC 的使用;第 3 章,PICC 的库函数;第 4 章,PIC16F877 单片机实验板介绍;第 5 章,PIC16F877 的外围功能模块;第 6 章,模拟量输入与输出;第 7 章,秒表;第 8 章,通用同步/异步通信的应用;第 9 章,PIC16F87X 在 CAN 通信中的应用;第 10 章,利用 CCP 模块设计频率计;第 11 章,交流电压测量;第 12 章,与 PLC 接口的 4 位 LED 数字显

示表;第13章,数控步进直流稳压电源;第14章,单片机控制的电动自行车驱动系统;第15章,液晶显示模块编程。

其中第9,12及14章与本书介绍的实验板联系不太紧密;其余章节都是在实验板的基础上开发出来的,只需少量外围电路即可构成实验系统。

编者还在北京航空航天大学出版社出版了《PIC16F87X 单片机实用软件与接口技术——汇编语言及其应用》、《PIC16F87X 数据手册》和在重庆大学出版社出版了《单片机原理及应用》。它们各有侧重,本书侧重于C语言编程方法和C语言应用编程方面。书中介绍了大量的程序例子,涉及到PIC16F87X单片机的各个功能模块的编程应用;给出了多个应用实例的电路原理图和源程序清单,所有程序均调试通过。本书可作为大学本科学生单片机原理及应用课程的实验指导书,对单片机开发者来说也是一本很好的软件开发参考书。

在成书过程中,得到了电力电子与电力传动系李远树、郑连清、郑群英、巫宣文等老师的大力协助和支持,他们编写了部分章节,并做了校对录入以及实验板制作工作。在此表示感谢。

在这里还要感谢微芯公司提供的大力支持。

限于编者的水平,书中难免存在错误和不当之处,恳请读者批评指正。

编 者

2002年1月于重庆大学

# 目 录

## 第 1 章 PICC C 语言基础和特点

|      |                     |    |
|------|---------------------|----|
| 1.1  | PICC 与 ANSI C 标准的区别 | 1  |
| 1.2  | 处理器支持               | 1  |
| 1.3  | 配置设置                | 2  |
| 1.4  | ID 区域               | 3  |
| 1.5  | EEPROM 数据           | 3  |
| 1.6  | 位指令                 | 3  |
| 1.7  | 支持数据类型              | 4  |
| 1.8  | 绝对变量                | 8  |
| 1.9  | 结构和联合               | 8  |
| 1.10 | ROM 和 RAM 中的字符串     | 10 |
| 1.11 | 常数型和可变量型变量          | 10 |
| 1.12 | ROM 对象的存放及访问        | 11 |
| 1.13 | 特殊类型限定词             | 11 |
| 1.14 | 指 针                 | 12 |
| 1.15 | 工具定义特性              | 14 |
| 1.16 | C 的中断处理             | 14 |
| 1.17 | C 语言和汇编语言的混合编程      | 17 |
| 1.18 | 程序链接                | 20 |
| 1.19 | 函数参数传递              | 20 |
| 1.20 | 函数返回值               | 21 |
| 1.21 | 函数调用规则              | 23 |
| 1.22 | 局部变量                | 23 |
| 1.23 | pragma 伪指令          | 24 |
| 1.24 | 标准 I/O 函数           | 26 |
| 1.25 | MPLAB 的特殊调试信息       | 26 |

## 第 2 章 PICC 的使用

- 2.1 生成单源文件项目 ..... 27
- 2.2 生成多源文件项目 ..... 37

## 第 3 章 PICC 的库函数

- 3.1 ABS 函数 ..... 44
- 3.2 ACOS 函数 ..... 45
- 3.3 ASCTIME 函数 ..... 45
- 3.4 ASIN 函数 ..... 47
- 3.5 ATAN 函数 ..... 47
- 3.6 ATAN2 函数 ..... 48
- 3.7 ATOF 函数 ..... 49
- 3.8 ATOI 函数 ..... 49
- 3.9 ATOL 函数 ..... 50
- 3.10 CEIL 函数 ..... 51
- 3.11 COS 函数 ..... 51
- 3.12 COSH, SINH, TANH 函数 ..... 52
- 3.13 CTIME 函数 ..... 53
- 3.14 DI, EI 函数 ..... 53
- 3.15 DIV 函数 ..... 54
- 3.16 EEPROM\_READ, EEPROM\_WRITE 函数 ..... 55
- 3.17 EVAL\_POLY 函数 ..... 56
- 3.18 EXP 函数 ..... 56
- 3.19 FABS 函数 ..... 57
- 3.20 FLOOR 函数 ..... 57
- 3.21 FREXP 函数 ..... 58
- 3.22 GET\_CAL\_DATA 函数 ..... 59
- 3.23 GMTIME 函数 ..... 59
- 3.24 ISALNUM, ISALPHA, ISDIGIT, ISLOWER 等函数 ..... 60
- 3.25 KBHIT 函数 ..... 62
- 3.26 LDEXP 函数 ..... 63
- 3.27 LDIV 函数 ..... 63
- 3.28 LOCALTIME 函数 ..... 64

|      |                                          |    |
|------|------------------------------------------|----|
| 3.29 | LOG, LOG10 函数 .....                      | 65 |
| 3.30 | MEMCHR 函数 .....                          | 66 |
| 3.31 | MEMCMP 函数 .....                          | 67 |
| 3.32 | MEMCPY 函数 .....                          | 68 |
| 3.33 | MEMMOVE 函数 .....                         | 69 |
| 3.34 | MEMSET 函数 .....                          | 69 |
| 3.35 | MODF 函数 .....                            | 70 |
| 3.36 | PERSIST_CHECK, PERSIST_VALIDATE 函数 ..... | 71 |
| 3.37 | POW 函数 .....                             | 72 |
| 3.38 | PRINTF 函数 .....                          | 72 |
| 3.39 | RAND 函数 .....                            | 74 |
| 3.40 | SIN 函数 .....                             | 75 |
| 3.41 | SPRINTF 函数 .....                         | 76 |
| 3.42 | SQRT 函数 .....                            | 76 |
| 3.43 | SRAND 函数 .....                           | 77 |
| 3.44 | STRCAT 函数 .....                          | 78 |
| 3.45 | STRCHR, STRICHR 函数 .....                 | 79 |
| 3.46 | STRCMP, STRICMP 函数 .....                 | 80 |
| 3.47 | STRCPY 函数 .....                          | 81 |
| 3.48 | STRCSPN 函数 .....                         | 82 |
| 3.49 | STRLEN 函数 .....                          | 82 |
| 3.50 | STRNCAT 函数 .....                         | 83 |
| 3.51 | STRNCMP, STRNICMP 函数 .....               | 84 |
| 3.52 | STRNCPY 函数 .....                         | 85 |
| 3.53 | STRPBRK 函数 .....                         | 86 |
| 3.54 | STRRCHR, STRRICHR 函数 .....               | 87 |
| 3.55 | STRSPN 函数 .....                          | 88 |
| 3.56 | STRSTR, STRISTR 函数 .....                 | 88 |
| 3.57 | STRTOK 函数 .....                          | 89 |
| 3.58 | TAN 函数 .....                             | 90 |
| 3.59 | TIME 函数 .....                            | 91 |
| 3.60 | TOLOWER, TOUPPER, TOASCII 函数 .....       | 91 |
| 3.61 | VA_START, VA_ARG, VA_END 函数 .....        | 92 |
| 3.62 | XTOI 函数 .....                            | 93 |

## 第 4 章 PIC16F877 单片机实验板介绍

|                      |    |
|----------------------|----|
| 4.1 实验板功能介绍 .....    | 96 |
| 4.2 实验板的硬件布局 .....   | 98 |
| 4.3 测试点及主要器件介绍 ..... | 99 |

## 第 5 章 PIC16F877 的外围功能模块

|                                            |     |
|--------------------------------------------|-----|
| 5.1 输入/输出端口 .....                          | 103 |
| 5.2 利用 MSSP 模块的 SPI 方式实现与 LED 数码显示接口 ..... | 104 |
| 5.3 利用 I/O 直接扩展键盘 .....                    | 111 |
| 5.4 利用 PORTB 端口的电平变化中断实现键盘功能 .....         | 116 |
| 5.5 利用 MSSP 模块的 SPI 方式扩展并行输入端口 .....       | 120 |
| 5.6 CCP 模块的 PWM 波形产生方法 .....               | 123 |
| 5.7 监视定时器的应用 .....                         | 126 |
| 5.8 休眠工作方式与其激活 .....                       | 130 |

## 第 6 章 模拟量输入与输出

|                                                     |     |
|-----------------------------------------------------|-----|
| 6.1 A/D 转换的应用 .....                                 | 133 |
| 6.2 MSSP 模块的 I <sup>2</sup> C 总线方式扩展串行 D/A 芯片 ..... | 139 |

## 第 7 章 秒 表

|                |     |
|----------------|-----|
| 7.1 工作原理 ..... | 146 |
| 7.2 程序设计 ..... | 147 |

## 第 8 章 通用同步/异步通信的应用

|                         |     |
|-------------------------|-----|
| 8.1 USART 的波特率发生器 ..... | 155 |
| 8.2 USART 的异步工作方式 ..... | 156 |
| 8.3 USART 的同步主控方式 ..... | 157 |
| 8.4 USART 的同步从动方式 ..... | 158 |
| 8.5 单片机双机异步通信 .....     | 159 |
| 8.6 单片机双机同步通信 .....     | 163 |
| 8.7 单片机与 PC 机通信 .....   | 168 |

**第 9 章 PIC16F87X 在 CAN 通信中的应用**

|              |     |
|--------------|-----|
| 9.1 CAN 通信原理 | 171 |
| 9.2 硬件电路     | 177 |
| 9.3 软件清单     | 184 |

**第 10 章 利用 CCP 模块设计频率计**

|                      |     |
|----------------------|-----|
| 10.1 CCP 模块的捕捉工作方式简介 | 193 |
| 10.2 设计要求            | 194 |
| 10.3 硬件原理图           | 195 |
| 10.4 设计与测试原理         | 195 |
| 10.5 程序设计            | 196 |

**第 11 章 交流电压测量**

|                 |     |
|-----------------|-----|
| 11.1 模拟输入电路     | 204 |
| 11.2 数据处理原理     | 205 |
| 11.3 程序流程图及程序清单 | 206 |

**第 12 章 与 PLC 接口的 4 位 LED 数字显示表**

|                 |     |
|-----------------|-----|
| 12.1 数显表头硬件电路原理 | 211 |
| 12.2 数显表头软件设计思路 | 213 |
| 12.3 程序流程图      | 213 |
| 12.4 程序清单       | 214 |

**第 13 章 数控步进直流稳压电源**

|             |     |
|-------------|-----|
| 13.1 电路原理图  | 221 |
| 13.2 系统工作原理 | 222 |
| 13.3 程序设计   | 223 |

**第 14 章 单片机控制的电动自行车驱动系统**

|                        |     |
|------------------------|-----|
| 14.1 单片机控制的电动自行车驱动系统简介 | 231 |
| 14.2 无刷直流电动机的工作原理      | 231 |
| 14.3 控制系统结构设计          | 232 |
| 14.4 控制系统软件设计          | 233 |

## 第 15 章 液晶显示模块编程

|            |                                    |     |
|------------|------------------------------------|-----|
| 15.1       | PIC16F877 与 MG-12232 的硬件接口电路 ..... | 244 |
| 15.2       | 软件编程 .....                         | 246 |
| 15.3       | 液晶显示屏的结构 .....                     | 250 |
| 15.4       | 程序清单 .....                         | 250 |
| 参考文献 ..... |                                    | 260 |

# 第 1 章 PICC C 语言基础和特点

PICC 具有许多特殊的性质,并且进行了 C 语言的扩展,从而可以更轻松地完成编程任务。本章提供了编辑器选项和一些有用的特点,读完本章之后,可以做到:

- 配置 I/O 接口程序,从而可利用<stdio. h>程序对硬件进行操作。
- 用 C 语言建立中断。
- 利用 C 语言进行 I/O 口编程。
- 通过内嵌或外部汇编语言程序建立起 C 和汇编之间的联系。

## 1.1 PICC 与 ANSI C 标准的区别

PICC 与 ANSI C 标准的区别在于函数的递归调用。这是因为 PIC 受硬件的限制,没有堆栈,内存的数量也有限;所以不支持递归调用。

## 1.2 处理器支持

PICC 支持的处理器范围如表 1.1 所列。只需在 LIB 目录下编写 picinfo. ini 文件,其他处理器就可以加进来。picinfo. ini 文件分为低等、中等及高等 3 部分,用户定义的处理器可以放到文件的最后。文件的开头部分解释怎样定义处理器。

表 1.1 PICC 支持的处理器范围

| Baseline Processors | Midrange Processors |           |           | High - End Processors |
|---------------------|---------------------|-----------|-----------|-----------------------|
| PIC12C508           | PIC12C671           | PIC16C711 | PIC16F84  | PIC17C42              |
| PIC12C508A          | PIC12C672           | PIC16C712 | PIC16F84A | PIC17C42A             |
| PIC12C509           | PIC12CE673          | PIC16C715 | PIC16F627 | PIC17C43              |
| PIC12C509A          | PIC12CE674          | PIC16C716 | PIC16F628 | PIC17C44              |
| PIC12CE518          | PIC14000            | PIC16C717 | PIC16F870 | PIC17C752             |
| PIC12CE519          | PIC16C554           | PIC16C72  | PIC16F871 | PIC17C756             |
| PIC12CR509A         | PIC16C554A          | PIC16C72A | PIC16F872 | PIC17C756A            |
| PIC16C505           | PIC16C556           | PIC16C73  | PIC16F873 | PIC17C762             |
| PIC16C52            | PIC16C556A          | PIC16C73A | PIC16F874 | PIC17C766             |
| PIC16C54            | PIC16C558           | PIC16C73B | PIC16F876 | PIC17CR42             |
| PIC16C54A           | PIC16C558A          | PIC16C74  | PIC16F877 | PIC17CR43             |

续表 1.1

| Baseline Processors | Midrange Processors |            |  | High - End Processors |
|---------------------|---------------------|------------|--|-----------------------|
| PIC16C54B           | PIC16C61            | PIC16C74A  |  |                       |
| PIC16C54C           | PIC16C62            | PIC16C74B  |  |                       |
| PIC16C55            | PIC16C62A           | PIC16C745  |  |                       |
| PIC16C55A           | PIC16C62B           | PIC16C76   |  |                       |
| PIC16C56            | PIC16C620           | PIC16C77   |  |                       |
| PIC16C56A           | PIC16C620A          | PIC16C770  |  |                       |
| PIC16C57            | PIC16C621           | PIC16C771  |  |                       |
| PIC16C57C           | PIC16C621A          | PIC16C773  |  |                       |
| PIC16C58            | PIC16C622           | PIC16C774  |  |                       |
| PIC16C58A           | PIC16C622A          | PIC16C765  |  |                       |
| PIC16C58B           | PIC16C63            | PIC16C84   |  |                       |
| PIC16CR54A          | PIC16C63A           | PIC16C923  |  |                       |
| PIC16CR54B          | PIC16C64            | PIC16C924  |  |                       |
| PIC16CR54C          | PIC16C64A           | PIC16C99   |  |                       |
| PIC16CR56A          | PIC16C641           | PIC16CE623 |  |                       |
| PIC16CR57B          | PIC16C642           | PIC16CE624 |  |                       |
| PIC16CR57C          | PIC16C65            | PIC16CE625 |  |                       |
| PIC16CR58A          | PIC16C65A           | PIC16CR62  |  |                       |
| PIC16CR58B          | PIC16C65B           | PIC16CR63  |  |                       |
| PIC16HV540          | PIC16C66            | PIC16CR64  |  |                       |
|                     | PIC16C661           | PIC16CR65  |  |                       |
|                     | PIC16C662           | PIC16CR72  |  |                       |
|                     | PIC16C67            | PIC16CR83  |  |                       |
|                     | PIC16C71            | PIC16CR84  |  |                       |
|                     | PIC167C710          | PIC16F83   |  |                       |

### 1.3 配置设置

PIC 处理器的配置可以用 `_config` 宏进行设置,形式如下:

```
#include <pic.h>
```

```
__CONFIG( x);
```

其中 `x` 是配置语句。在头文件中适当地定义了不同的标号,可以利用这些标号使用芯片的某一特性。下面是一个 PIC16C5X 的例子:

```
__CONFIG(WDTDIS & XT & UNPROTECT);
```

这条语句的作用是关闭看门狗定时器,使用 XT 晶振和不保护程序代码。在下载程序之前,一定要先检查 `__CONFIG` 宏中所有的配置位以及头文件是否正确;也可以通过 ICD 的“OPTION”对话框配置。

注意,所有的个人选项都被加到一起,所有未被宏选择的位将保持不编程状态。应该在下载时,保证所有选择的位都正确,从而使各部分合理地工作。更详细的资料可以查看有关的PIC数据手册。

宏\_\_CONFIG不产生可执行代码,使用时可以将其放在函数的外部。

## 1.4 ID 区域

一些PIC芯片在可编程存储区间外,还有一些其他区域。这些区域可以用来存储下载信息,例如ID号。宏\_\_IDLOC可以用来将所需数据放到这一区域,它使用的格式如下:

```
#include <pic.h>
__IDLOC(x);
```

这里x为一串十六进制数,指向相应的ID区域。其中只有低4位ID数被下载编程;因此,对于

```
__IDLOC(15F0);
```

将有4位十进制数进入ID区域,即1,5,15(F)及0。最基本的ID区域的确定,是由指令idloc完成的。它将自动根据所选择的芯片安排适当的地址。

## 1.5 EEPROM 数据

有些芯片支持外部EEPROM数据。这可以通过宏\_\_EEPROM\_DATA以十六进制文件的形式初始化EEPROM数据。用法如下:

```
#include <pic.h>
__EEPROM_DATA(0, 1, 2, 3, 4, 5, 6, 7);
```

这个宏需要8个参数,分别表示8个数据值。每个数值为1B。不需要使用的数据则应该用参数0代替。这个宏可以被多次调用,从而定义所要求数量的EEPROM数据。宏应该放到所有函数的外部。每个数据存储时,都占用2B,并且第一个字节为0,存储为十六进制的形式。

这个宏在程序运行时,不能将数据写入EEPROM存储区;在下载编程时,宏EEPROM\_READ(),EEPROM\_WRITE()及其他这类宏可以用来读或者写EEPROM。

## 1.6 位指令

有时,PICC要用到PIC的位操作指令。例如,当需要位取反操作以及要改变一个整型数中的某一位时,编译器将检测屏蔽值,以决定是否可以采用一条位指令实现同样的功能。

```
int foo;
foo |= 0x40;
```

将产生下面的指令：

```
bsf _foo,6
```

为了将一个整型数中的某一位置 1 或清 0, 可以使用下面的宏定义：

```
#define bitset(var, bitno) ((var) |= 1 << (bitno))
#define bitclr(var, bitno) ((var) &= ~(1 << (bitno)))
```

为了执行与上面一样的操作, 也可以采用如下的位设置宏：

```
bitset(foo, 6);
```

## 1.7 支持数据类型

PICC 编辑器支持基本的 1, 2 及 4 B 数据。所有多字节数据遵守的格式为低字节在前、高字节在后的原则, 即一个数中低位字节存储在地址低的存储单元中。表 1.2 列出了 PICC 支持的数据类型及对应的大小和数学类型。其中, 字符型的缺省值为无符号型; 如果 PICC 采用了 SIGNED\_CHAR 选项, 则为有符号型。Double 型变量的缺省值为 24 位; 如果选择 PIC-D32, 则为 32 位。

表 1.2 PICC 数据类型

| 类 型            | 大小/bit | 数学类型     |
|----------------|--------|----------|
| Bit            | 1      | 二进制      |
| Char           | 8      | 符号或无符号整型 |
| Unsigned char  | 8      | 无符号整型    |
| Short          | 16     | 符号整型     |
| Unsigned short | 16     | 无符号整型    |
| Int            | 16     | 符号整型     |
| Unsigned int   | 16     | 无符号整型    |
| Long           | 32     | 符号整型     |
| Unsigned long  | 32     | 无符号整型    |
| Float          | 24     | 实 型      |
| double         | 24     | 实 型      |

### 1.7.1 数进制及常数

整数格式具有不同的数进制。PICC 支持 ANSI 标准码制, 同样的, C 代码也支持二进制常数。不同格式表示方法如表 1.3 所列。其指明二进制和十六进制的字母有时不敏感, 就像用字母指明十六进制数一样。

表 1.3 数据格式

| 进 制  | 格 式          | 例 子        |
|------|--------------|------------|
| 二进制  | ob+数值或 OB+数值 | 0b10101100 |
| 八进制  | 0+数值         | 0234       |
| 十进制  | 数 值          | 432        |
| 十六进制 | 0x+数值或 0X+数值 | 0x2F       |

一个整型常量的类型应该是可以装得下这个值、使之不溢出的、最小的数据类型。在常数后面加 L 或 l, 可以使常数的数据类型为有符号或无符号长整型; 如加后缀 u 或 U, 则可使常数为无符号数据类型; 如同时加 l 或 L 和 u 或 U, 则表示无符号长整型。浮点常数加后缀 f 或 F 时, 为浮点数据类型; 否则, 认为是双精度数据类型。在双精度数据类型后加后缀 l 或 L, 则表示 PICC 中的双精度数据类型。

字符常数应加单引号(注意: 此处的单引号为西文符号, 不能用中文符号)表示, 如 'a'。字符常数的数据类型为字符型。PICC 不支持多字节字符常数。

字符串常数或常量需要加双引号(注意: 此处的双引号为西文符号, 不能用中文符号)表示, 例如 "hello world"。字符常量的数据类型为字符常量指针类型(const char \*), 被存储在 ROM 中。如指定字符串常量为非常量字符指针, 编辑器将产生一个警告。例如:

```
char * cp = "one";           // "one" in ROM, 产生警告
const char * ccp = "two";    // "two" in ROM
char ca[] = "two";          // "two" 与上面不同
```

(注意: HI\_TECH PICC 的注释可以用 "//" 号, 也可以用 "/\* 注释内容 \*/"。 ) 一个由字符串数组初始化的非常数数组, 如上面最后一句所示, 将产生一个 RAM 数组, 这个数组在程序开始运行前, 被字符串常数 "two" (从 ROM 中复制而来) 初始化; 而其他方法表示的字符串常数将产生一个未命名的常数数组, 由直接访问 ROM 获得。

除像上面最后一条语句所示的字符串将在 RAM 中被初始化外, 对具有相同字母顺序的字符串, PICC 将采用相同的存储空间和标号; 对于邻近的字符串(也就是 2 个字符串之间只有一个空格), 编辑器将自动将它们连接起来。也就有:

```
const char * cp = "hello world";
```

编辑器将使指针指向字符串 "hello world"。

### 1.7.2 位数据类型

PICC 允许利用 bit 指令定义位变量。一个变量被定义为位变量, 例如:

```
static bit init_flag;
```

它将被分配在可寻址位区域 rbit\_n (n 表示存储区的序号), 并且只在这个模式或函数下有效; 当在所有的函数外部使用下面的语句进行位定义时:

```
bit init_flag;
```