

单片机 C 高级语言程序设计及其应用

袁 涛 李月香 杨胜利 编著

北京航空航天大学出版社

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

内 容 简 介

C 高级语言是目前流行的一种计算机高级语言,主要用于单片机和一般微型计算机。本书具体叙述了 C 高级语言,包括变量类型、变量定义、函数、可执行语句、浮点运算、与硬件有关的操作、C 语言与汇编语言的交叉使用,且有大量经过笔者调试的程序实例;还叙述了编译和连接操作,给出了具体包含上机操作在内的完整程序实例,以及程序固化的操作实例;同时对单片机硬件也进行了相应叙述。

本书通俗易懂,很适合初学者使用。本书可作为高等院校和培训班的教材或参考书,也适合从事单片机开发的科技人员和自学读者使用。

图书在版编目(CIP)数据

单片机 C 高级语言程序设计及其应用/袁涛等编.

—北京:北京航空航天大学出版社,2001.10

ISBN 7-81077-117-5

I. 单… II. 袁… III. C 语言-程序设计

IV. TP312

中国版本图书馆 CIP 数据核字(2001)第 071654 号

单片机 C 高级语言程序设计及其应用

袁 涛 李月香 杨胜利 编著

责任编辑 曾昭奇

*

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:82317024 传真:82328026

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

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

北京密云华都印刷厂印装 各地书店经销

*

开本:787×1092 1/16 印张:15 字数:384 千字

2001 年 10 月第 1 版 2002 年 3 月第 2 次印刷 印数:3 001~8 000 册

ISBN 7-81077-117-5/TP·066 定价:18.00 元

前 言

C 高级语言是目前流行的一种计算机高级语言。它主要用于单片机和一般微型计算机,具体有 C++、VC、TC、C-86、C-96、CC78K0 等。它们的区别主要是编译、连接程序不同,从而生成不同的机器代码。例如,C-96 编译、连接程序所生成的是 MCS-96 系列单片机的机器代码,C-86 编译、连接程序所生成的是 8086/8088 机器代码,CC78K0 编译、连接程序所生成的是 NEC 公司 78K0 系列单片机的机器代码。

单片机的真正单片化、低功耗、电源电压允许宽范围波动,这是目前及可预见的未来单片机发展的主流。单片机的单片化,将传统的接口电路基本都集成到了单片机内部,使得电路设计越来越简单。单片化的单片机,由于不使用外部并行总线,抗干扰能力先天就明显优于传统总线扩展式单片机。单片机的单片化必然使得单片机型号明显增多,对开发应用人员来说,应该针对不同情况选择不同型号的单片机,C 高级语言则使此项工作简单易行。

单片机应用广泛,其程度明显超过个人计算机。进行单片机开发工作的人员数量大、分布领域广、技术基础差异大,推广使用 C 高级语言开发单片机是非常必要的。国际上使用 C 语言开发单片机已经比较普遍,中国加入 WTO 后在这方面也必须与国际接轨,才能提高国际竞争力。

C 高级语言程序设计快、可读性好、可靠性高、可移植性好、代码转换质量高。一般情况下,完成同样的任务,使用 C 高级语言比用汇编语言可提高工作效率 5~10 倍,在调试阶段更容易体会到这一点。C 语言程序可维护性好。由于多种原因,包括应用者需求的变化等,需要对系统进行维护,世界软件巨头美国微软公司也不例外,例如 Windows 面市后,到目前为止,仍然在陆续打补丁进行后期维护。越来越多的情况是维护由不同的人进行(不是当初的开发人员),汇编语言难于进行这样的维护,C 语言则容易做到这一点。单片机 C 高级语言的特点是同时兼有高级语言和汇编语言的优点,还能像汇编语言那样直接利用 CPU 的硬件特性进行程序设计,直接操作单片机的硬件和接口。C 高级语言简单、易学,是一种结构化的程序设计语言。一个完整的程序可由一个或多个独立编译的源程序文件组成。C 高级语言目标模块还可以同汇编语言连接组成一个完整程序。目前在单片机应用领域,C 高级语言越来越受到人们的重视。

由于 C 高级语言具有可移植性好的特点,所以书中以何种单片机为例就显得不那么重要了。本书虽然以 NEC 单片机为例,如果将来需要移植到其他单片机上并不困难。NEC 公司针对 78K0 系列单片机推出的 CC78K0 C 语言正是单片机领域的典型代表,使得人们可以更加方便地开发具有优良性能的 NEC 单片机应用系统。NEC 公司 8 位单片机真正实现了单片化、低功耗(其待机电流可低至 $0.05\sim 0.01\mu\text{A}$,片内具有 $8\sim 60\text{K}$ 字节的程序存储器(ROM 或 Flash)空间、 $512\sim 5120$ 字节的片内数据存储器(RAM)空间、定时计数器、多路 10 位或 8 位 A/D 和 D/A 转换、丰富的串行接口 UART 和 I²C 等,片上还有看门狗定时器和多达 22 级的中断源等)和电源电压范围宽(一般允许在 $1.8\sim 5.5\text{V}$ 波动)。书中以 C 高级语言为主进行叙述。为了避免重复,对 NEC 单片机 C 语言 CC78K0 着重叙述它们与标准 C 语言不同的部

分。书中不仅叙述了 C 语言本身,还详细叙述了如何进行编译、连接,包括编译、连接控制项的使用,这对如何用好 C 高级语言是不可忽视的内容。

78K0 系列各型号单片机 C 编译器相同,只是器件文件不同而已。所以能使用 C 语言熟练开发一个型号单片机后,再开发该系列中其他几十个型号单片机就很容易。

本书的实例基本是以 NEC 公司 780058 单片机为例,笔者也使用 C 语言在 78F0034 单片机(具有 10 位 A/D,Flash 存储程序)上开发了多个项目,还使用 C 语言在低价格的 78K0S 系列单片机 78F9116 单片机(具有 10 位 A/D,Flash 存储程序)上开发了多个项目。确实体验了 C 语言可移植性好、可维护性好、开发效率高的特点。笔者甚至仅使用 C 编译器和程序固化器就完成了某个项目的开发。

为了照顾过去熟悉 8051 单片机的使用者,NEC 公司 78K0 系列单片机按照缺省设置,则与 8051 单片机没有什么区别,甚至寄存器可以使用与 8051 相同的名称;同时又可以使用多种设置选项,选择使用单片机芯片内部丰富多彩的其他各种功能。

嵌入式系统中使用的嵌入式操作系统绝大部分都是以 C 高级语言为基础的。嵌入式操作系统很多是直接提供源码给使用者,这些主要是以 C 高级语言形式提供的。因而掌握开发单片机的 C 高级语言就是一件重要且必要的工作。

本书适用于 C 高级语言的初学者,书中不仅对 C 语言进行了叙述,还有大量程序实例,且经过笔者实际调试,对单片机硬件也进行了相应叙述。对于已掌握一定基础的人,可以先阅读附录和第五章,然后根据自己的程度,选择阅读书中内容。书中还有具体的上机实际操作实例,可以边学习边操作。本书既适用于工程技术人员自学,也适用于各工科大、专院校和中等专业学校的教材及培训班教材。本书由清华大学自动化系袁涛、山西大学计算机系李月香、清华大学杨胜利执笔。

在成书过程中,得到了 NEC 公司、香港日电电子有限公司、首钢日电电子公司、清华-NEC 单片机及嵌入式系统研究与培训中心的大力支持。同时,得到翁绍捷、木合塔尔、陈潇炜、焦钢、吴章群、杨大炜、汤岚岚、万水明等同志的协助。在此,向以上单位和同志表示衷心感谢。

由于编者水平有限且时间仓促,书中难免存在缺点和错误,恳请读者批评指正。

编 者

2001.6

于清华大学自动化系

目 录

上 篇

第 1 章 C 高级语言基本知识

1.1 概 述	3
1.1.1 C 高级语言及其特点	3
1.1.2 使用 C 高级语言的必要性	4
1.2 字符集和标识符	4
1.2.1 字符集	4
1.2.2 标识符和保留字	5
1.2.3 符号、分界符和空符号的作用	5
1.2.4 注 释	6
1.3 常 量	6
1.3.1 整型常量	6
1.3.2 实型常量	7
1.3.3 字符常量和字符串常量	7
1.4 变量及变量定义	7
1.4.1 整型变量	7
1.4.2 实型变量	8
1.4.3 字符变量	8
1.4.4 各种类型变量的数值范围	8
1.5 运算、表达式及规则	8
1.5.1 运算对象	8
1.5.2 表达式	9
1.5.3 算术运算及其表达式	9
1.5.4 关系运算及其表达式	10
1.5.5 逻辑运算及其表达式	10
1.5.6 表达式计算	11
1.6 数组和结构	12
1.6.1 数 组	12
1.6.2 数组元素的引用	13
1.6.3 结 构	13
1.6.4 结构体数组和结构体内数组	14

第 2 章 C 可执行语句

2.1 赋值语句和赋值表达式	15
----------------------	----

2.1.1	赋值语句	15
2.1.2	赋值表达式	15
2.2	循环控制语句	16
2.2.1	while 语句	16
2.2.2	do-while 语句	16
2.2.3	for 语句	17
2.3	条件(if)语句	17
2.3.1	if 语句	17
2.3.2	嵌套 if 语句	18
2.4	语句标号和 goto 语句	18

第 3 章 函 数

3.1	概 述	20
3.2	函数定义的一般形式	20
3.2.1	函数参数	21
3.2.2	函数的返回值	21
3.3	函数的调用	22

第 4 章 指 针

4.1	指针的概念	23
4.2	指针与数组	24

第 5 章 扩展功能

5.1	CC78K0 编译系统扩展的保留字	27
5.2	存储器	28
5.3	#pragma 指令	28
5.4	callt 函数(callt/___callt)	29
5.5	寄存器变量	30
5.6	saddr 存储区的使用	30
5.7	特殊功能寄存器(SFR)的使用	32
5.8	位变量的使用	32
5.9	在 C 语言中使用汇编语句	34
5.10	中断功能的使用	35
5.11	开中断函数 EI()和关中断函数 DI()	37
5.12	CPU 控制指令	38
5.13	绝对地址存取指令	39
5.14	位段的声明	40
5.15	二进制常数	42
5.16	循环移位	42

5.17	乘法函数	43
5.18	除法函数	44
5.19	BCD 操作函数	45
5.20	寄存器直接存取功能	47
5.21	callf 函数(callf/___callf)	49
5.22	noauto 函数	50
5.23	norec 函数	51
5.24	中断函数限定符(___interrupt、___interrupt_brk)的使用	52

第 6 章 编译与连接

6.1	编译控制项的设置	54
6.1.1	编译预处理选项(1. Preprocess)	54
6.1.2	存储器中的变量分配(2. Memory Arrange)	55
6.1.3	程序优化(3. Optimize)	56
6.1.4	调试和输出(4. Debug/Output)	57
6.1.5	编译选项的其他控制(5. Others)	58
6.2	连接控制项的设置	59

第 7 章 程序设计实例

7.1	样本程序 1(计算一组数据的均值、残差、方差、标准差)	60
7.2	样本程序 2(分类程序)	66

附 录

附录 A	saddr 存储区的符号名	69
A.1	通常模式	69
A.2	静态模式	70
附录 B	程序限制	71

下 篇

第 1 章 μ PD780058 单片机介绍

1.1	NEC 单片机简介	75
1.2	μ PD780058 单片机	75
1.2.1	μ PD780058 单片机的性能	75
1.2.2	引脚配置、功能及注意事项	76
1.2.3	功能结构	78
1.2.4	μ PD780058 单片机的应用举例	84

第 2 章 CPU 结构和存储器空间

2.1	存储器空间	85
-----	-------------	----

2.1.1	存储器结构	85
2.1.2	内部程序空间	86
2.1.3	内部数据空间	86
2.1.4	内部存储器直接地址读/写操作举例	87
2.2	处理器寄存器	87
2.2.1	控制寄存器(PC、PSW、SP)	87
2.2.2	通用寄存器	88
2.2.3	通用寄存器的使用举例	89
2.2.4	特殊功能寄存器	89
2.2.5	特殊功能寄存器的使用举例	92
2.3	外部器件扩展	93
2.3.1	功 能	93
2.3.2	功能结构框图	93
2.3.3	控制寄存器(MM、IMS)	94
2.3.4	外部扩展存储器及 I/O 器件的存取操作举例	95

第 3 章 输入/输出(I/O)端口

3.1	端口功能	96
3.2	端口配置	98
3.2.1	端口 0(P0)	98
3.2.2	端口 1(P1)	98
3.2.3	端口 2(P2)	98
3.2.4	端口 3(P3)	99
3.2.5	端口 4(P4)	99
3.2.6	端口 5(P5)	99
3.2.7	端口 6(P6)	99
3.2.8	端口 7(P7)	99
3.2.9	端口 12(P12)	99
3.2.10	端口 13(P13)	100
3.3	端口功能控制寄存器	100
3.3.1	端口模式寄存器(PM0~PM3、PM5~PM7、PM12、PM13)	100
3.3.2	上拉电阻选择寄存器(PUOL 和 PUOH)	101
3.3.3	存储器扩展模式寄存器(MM)	101
3.4	I/O 端口功能的使用	102

第 4 章 时钟发生电路

4.1	时钟发生器的功能	103
4.1.1	主系统时钟发生器	103
4.1.2	副系统时钟发生器	103

4.2 时钟发生器的配置	103
4.2.1 时钟发生器的配置	103
4.2.2 时钟发生器的电路框图	103
4.3 时钟发生器控制寄存器	104
4.3.1 振荡模式选择寄存器(OSMS)	104
4.3.2 处理器时钟控制寄存器(PCC)	104
4.4 系统时钟振荡器	105
4.4.1 主系统时钟振荡器	105
4.4.2 副系统时钟振荡器	106
4.5 时钟发生器使用举例	106

第5章 多功能定时器/计数器

5.1 μ PD780058 单片机内部定时器/计数器概述	107
5.2 16 位定时器/计数器	108
5.2.1 16 位定时器/计数器的功能	108
5.2.2 16 位定时器/计数器的配置	108
5.2.3 16 位定时器/计数器电路框图	109
5.2.4 16 位定时器/计数器输出控制电路框图	109
5.2.5 寄存器(CR00、CR01、TM0)	110
5.2.6 控制寄存器(TCL0、TMC0、CRC0、TOC0、PM3、INTM0、SCS)	111
5.2.7 16 位定时器/计数器的使用	115
5.3 8 位定时器/计数器	125
5.3.1 8 位定时器/计数器模式的功能	125
5.3.2 8 位定时器/计数器的配置	125
5.3.3 8 位定时器/计数器电路框图	125
5.3.4 8 位定时器/计数器输出控制电路框图	125
5.3.5 寄存器(CR10、CR20、TM1、TM2)	125
5.3.6 控制寄存器(TCL1、TMC1、TOC1、PM3)	127
5.3.7 8 位定时器/计数器的使用	130
5.4 钟表定时器	138
5.4.1 钟表定时器的功能	138
5.4.2 钟表定时器的配置	139
5.4.3 钟表定时器电路框图	139
5.4.4 控制寄存器(TCL2、TMC2)	139
5.4.5 钟表定时器的使用	141
5.5 看门狗定时器	143
5.5.1 看门狗定时器的功能	143
5.5.2 看门狗定时器的配置	144
5.5.3 看门狗定时器电路框图	144
5.5.4 控制寄存器(TCL2、WDTM)	144

5.5.5 看门狗定时器的使用	145
-----------------------	-----

第 6 章 时钟、蜂鸣器输出控制电路和实时输出端口

6.1 时钟输出控制电路	149
6.1.1 时钟输出控制电路的功能	149
6.1.2 时钟输出控制电路的配置	149
6.1.3 时钟输出控制电路框图	149
6.1.4 控制寄存器(TCL0、PM3)	150
6.1.5 时钟输出控制电路的使用	150
6.2 蜂鸣器输出控制电路	151
6.2.1 蜂鸣器输出控制电路的功能	151
6.2.2 蜂鸣器输出控制电路的配置	151
6.2.3 蜂鸣器输出控制电路框图	151
6.2.4 控制寄存器(TCL2、PM3)	151
6.2.5 蜂鸣器输出控制电路的使用	152
6.3 实时输出端口	153
6.3.1 实时输出端口的功能	153
6.3.2 实时输出端口的配置	153
6.3.3 实时输出端口电路框图	153
6.3.4 实时输出端口缓冲寄存器(RTBL 和 RTBH)	154
6.3.5 实时输出端口控制寄存器(RTPM、RTPC、PM12)	154
6.3.6 实时输出端口的使用	155

第 7 章 模数、数模转换器

7.1 模数(A/D)转换器	156
7.1.1 A/D 转换器的功能	156
7.1.2 A/D 转换器的配置	156
7.1.3 A/D 转换器电路框图	157
7.1.4 A/D 转换控制寄存器(ADM、ADIS、INTM1)	157
7.1.5 A/D 转换器的使用	159
7.2 数模(D/A)转换器	160
7.2.1 D/A 转换器功能	160
7.2.2 D/A 转换器的配置	161
7.2.3 D/A 转换电路框图	161
7.2.4 D/A 转换器模式寄存器(DAM)	161
7.2.5 D/A 转换器的使用	162

第 8 章 异步串行通信接口

8.1 串行通信接口通道 2	164
----------------------	-----

8.1.1	串行通信接口通道 2 的电路框图	164
8.1.2	串行通信接口通道 2 的功能	165
8.1.3	串行通信接口通道 2 的配置	166
8.1.4	寄存器(TXS、RXS、RXB)	166
8.1.5	控制寄存器(CSIM2、ASIM、ASIS、BRGC、SIPS)	166
8.1.6	收信/发信波特率时钟的产生	169
8.2	通信操作	171
8.2.1	数据格式	171
8.2.2	校验类型和操作	171
8.2.3	数据发送	171
8.2.4	数据接收	172
8.2.5	UART 模式操作时的注意事项	172
8.3	异步串行通信接口 UART 的使用	173

第 9 章 中断功能和检测功能

9.1	中断功能类型	177
9.1.1	非屏蔽中断(NMI)	177
9.1.2	可屏蔽中断(INT)	177
9.1.3	软件中断	177
9.1.4	检测输入	177
9.2	中断源及配置	177
9.2.1	中断源	177
9.2.2	中断功能的基本配置	178
9.3	中断功能控制寄存器	179
9.3.1	中断请求标志寄存器(IF0L、IF0H、IF1L)	180
9.3.2	中断屏蔽标志寄存器(MK0L、MK0H、MK1L)	181
9.3.3	中断优先顺序指定标志寄存器(PR0L、PR0H、PR1L)	181
9.3.4	外部中断模式寄存器(INTM0、INTM1)	181
9.3.5	采样时钟选择寄存器(SCS)	182
9.3.6	程序状态字(PSW)	184
9.4	中断服务操作	184
9.4.1	非屏蔽中断请求响应操作	184
9.4.2	可屏蔽中断请求响应操作	185
9.4.3	软件中断响应操作	186
9.4.4	中断保留	186
9.5	可屏蔽中断请求响应举例	186
9.6	测试功能	188
9.6.1	测试功能	188
9.6.2	测试功能控制寄存器(IF1L、MK1L、KRM)	188

9.6.3 测试输入信号响应操作	189
------------------------	-----

第 10 章 后备功能及复位功能

10.1 后备功能及配置	190
10.1.1 后备功能	190
10.1.2 后备功能控制寄存器 (OSTS)	190
10.2 后备功能操作	191
10.2.1 HALT 模式	191
10.2.2 STOP 模式	193
10.3 后备功能使用举例	194
10.4 复位功能	196
10.4.1 产生复位动作的操作	196
10.4.2 复位后各硬件部分的状态	196
10.4.3 复位功能电路	198
10.4.4 各种复位信号产生的复位时序	199

第 11 章 综合应用举例

11.1 开关量的输入/输出	200
11.2 电子日历	202

附 录

附录 A 开发工具及使用步骤	210
A.1 建立仿真环境的操作步骤	210
A.2 注意事项	210
A.3 编译及仿真调试的操作步骤	210
附录 B 编译及仿真调试软件的安装	211
B.1 软件安装过程	211
B.2 注意事项	211
附录 C FlashMASTER 编程器的使用方法	212
C.1 FlashMASTER 软件的安装	212
C.2 FlashMASTER 编程器与微机、单片机板的连接	212
C.3 编 程	214
C.4 说 明	216
附录 D 汇编语言指令系统	217
D.1 操作表达式、描述方法和指令操作表	217
D.2 操作栏的说明	225
D.3 标志位栏的说明	226

上

篇

第 1 章 C 高级语言基本知识

1.1 概 述

1.1.1 C 高级语言及其特点

C 语言作为高级语言,它更接近和体现人的设计思想。C 语言不仅具有一般高级语言的特点,而且还能像汇编语言那样直接利用 CPU 的硬件特性进行程序设计。因而,与其他高级语言相比,功能更多,用途更广泛,在 8 位、16 位、32 位单片机应用领域都受到人们的普遍重视。广泛用于单片机和微处理器的系统软件和应用软件的开发。

C 高级语言的特点主要体现在以下几个方面:

(1) 简单、易学。C 高级语言的语句可分为两类:一类是说明语句,用于说明变量和函数(函数类似于其他语言中的子程序,但功能更强);另一类是可执行语句,如赋值语句、条件语句等。

(2) 可读性好。C 高级语言属于结构化语言,其程序是块式结构,层次清晰,便于理解和阅读。一个完整的程序可由多个单独编译的源程序文件组成,每个源程序文件可由预编译命令和多个函数组成,函数可以互相调用和嵌套。

(3) 占用内存容量小,运行速度快。若程序长度接近或超过 2K 字节时,其占用内存大小和运行速度甚至可能优于一般人直接用汇编语言编写的程序。

(4) 可靠性高。用 C 高级语言编写的程序容易达到正确的目的。

(5) 可维护性好。既便于修改和增添,有利于将来扩充和开发,又便于发现程序中的错误。

(6) 能够使用与实际问题更接近的数据类型和数据结构。例如,布尔变量,字符,数组,结构体,有符号和无符号整数,浮点数,位操作等。

(7) 变量作用域的概念和规则,增强了程序编写的灵活性。例如,可以多人共同编制一个程序,而不必担心是否使用了相同的标识符。

(8) 程序设计速度快,开发成本低(投入人力、财力少),周期短,效益高。这是因为一条 C 语句相当于多条汇编语句,且程序出错的可能性小。

(9) C 高级语言可与汇编语言程序连接生成一组目标码。

(10) C 高级语言程序库可由用户增加和删改。

C 高级语言到机器代码的转换主要由编译程序完成。经过编译后生成可重定位的机器代码文件,最终由连接程序将多个模块连接定位。可重定位的目标模块(机器代码)可由 C 高级语言或汇编语言生成。编译程序提供一个输出清单、错误信息、一定数量的控制功能,以帮助进行程序开发和调试。C 编译程序的主要特点如下:

(1) 结构化编程。

(2) 兼容性好。可与汇编语言生成的目标模块相连接。

- (3) 支持多种数据类型及逻辑、算术、关系等多种运算。
- (4) 支持多种数据结构,如数组和结构体等。
- (5) 支持中断管理。
- (6) 丰富的编译控制,这增加了程序编译的灵活性。这些控制包括:

- ① 优化。
- ② 条件编译。
- ③ 存储器中变量的分配。
- ④ 符号交叉引用。
- ⑤ 选择列出 C 高级语言程序对应的汇编语言程序。

- (7) 有多种代码自动优化功能。

(8) 详细的错误检查。C 编译程序发现程序错误时,则提供很详细的错误信息。这对初学者是非常有益的。

1.1.2 使用 C 高级语言的必要性

C 高级语言,尤其是用于开发单片机的 C 语言,其优越性已在微机开发中明显体现出来了。从 20 世纪 80 年代中期开始,国际上单片机应用就进入了一个新时代,尤其是近年,单片机的速度、数据处理能力、存储器容量明显增大,我国目前也在逐步跟上国际微机应用发展的步伐。在诸多单片机中,NEC 单片机以其功能强、单片化、使用电压范围宽和低功耗占有一定的优势。在 NEC 78K0 系列单片机中,仅基本指令就有 63 条,完成同样的功能,有多种指令可选,这使我们编写程序时觉得很灵活,但也有不利因素,即在程序较长时,我们很难靠人工来充分利用各种指令的功能,很难使整个程序水平达到最优。对此,C 高级语言可以很好地予以解决。当今国际上科技发展速度加快,要想跟上世界科技发展速度,只靠增加工作时间是不行的,必须使用先进的技术和手段。在微机开发应用领域更是如此。对此,使用 C 高级语言开发微机(尤其是单片机),可大大缩短开发周期。例如,国内某单位,使用 C 及相应开发装置,3 个多月的时间完成了软件、硬件设计和调试(包含 35K 字节程序及显示、键盘、A/D、输出和 CPU 板等),并送到了现场运行,且学习、掌握 C 高级语言和 NEC 单片机也在这 3 个多月时间之内。若使用汇编语言,想如此迅速完成这样的课题,几乎是不可能的。

使用 C 高级语言的工作效率高,其生成的机器代码质量也是高水平的。程序越长,使用 C 高级语言的优越性越明显。使用 C 语言可显著增加程序可靠性。这一点与其他高级语言一致。

1.2 字符集和标识符

1.2.1 字符集

C 语言程序是由字符组成的。合法的字符分为如下几类。

26 个大写英文字母:

A B C D E F G H I J K L M
N O P Q R S T U V W X Y Z

26 个小写英文字母:

a b c d e f g h i j k l m

n o p q r s t u v w x y z

数字: 0 1 2 3 4 5 6 7 8 9

其他: ! " # % & ' () * + , - . / :

; < = > ? [\] ^ _ { | } ~

这些字符的名称和作用,将在后面章节中陆续介绍。如果一个 C 程序包含了上述全部字符以外的任何字符,编译程序将视为错误。

1.2.2 标识符和保留字

标识符是用来标识变量名、符号常量名、函数名、数组名、类型名、文件名等的有效字符序列。一个标识符最长可有 31 个字符。每个标识符的第一个字符必须是英文字母或下划线,其余字符可以是字母、数字或下划线。

保留字不能用作标识符,它们已有确定的意义,实际上是 C 语言的一部分。例如,do,if,for,auto,EOF 等。应注意,EOF 作为 C 的一条特殊语句,表示一个源程序文件(可单独编译的程序块)的结束。但对编译结果不产生影响。

标准 C 语言的保留字如下:

auto	break	case	char	const	continue	
default	do	double	else	enum	extern	for
float	goto	if	int	long	register	return
short	signed	sizeof	static	struct	switch	
typedef	union	unsigned	void	volatile	while	

在使用 CC78K0 编译器时,还有以下的保留字:

callt/___callt	callt 函数声明
callf/___callf	callf 函数声明
sreg/___sreg	sreg 变量声明
noauto	noauto 函数声明
norec/___leaf	norec 函数声明
bit	bit 型变量声明
boolean/___boolean	boolean 型变量声明
___interrupt	硬中断函数调用
___interrupt_brk	软中断函数调用
___banked1 ___banked15	bank 函数
___asm	asm 语句
___rtos_interrupt	RTOS 中断处理

1.2.3 符号、分界符和空符号的作用

C 语句是由符号组成的,每个符号属于下列种类之一。

(1) 标识符。