

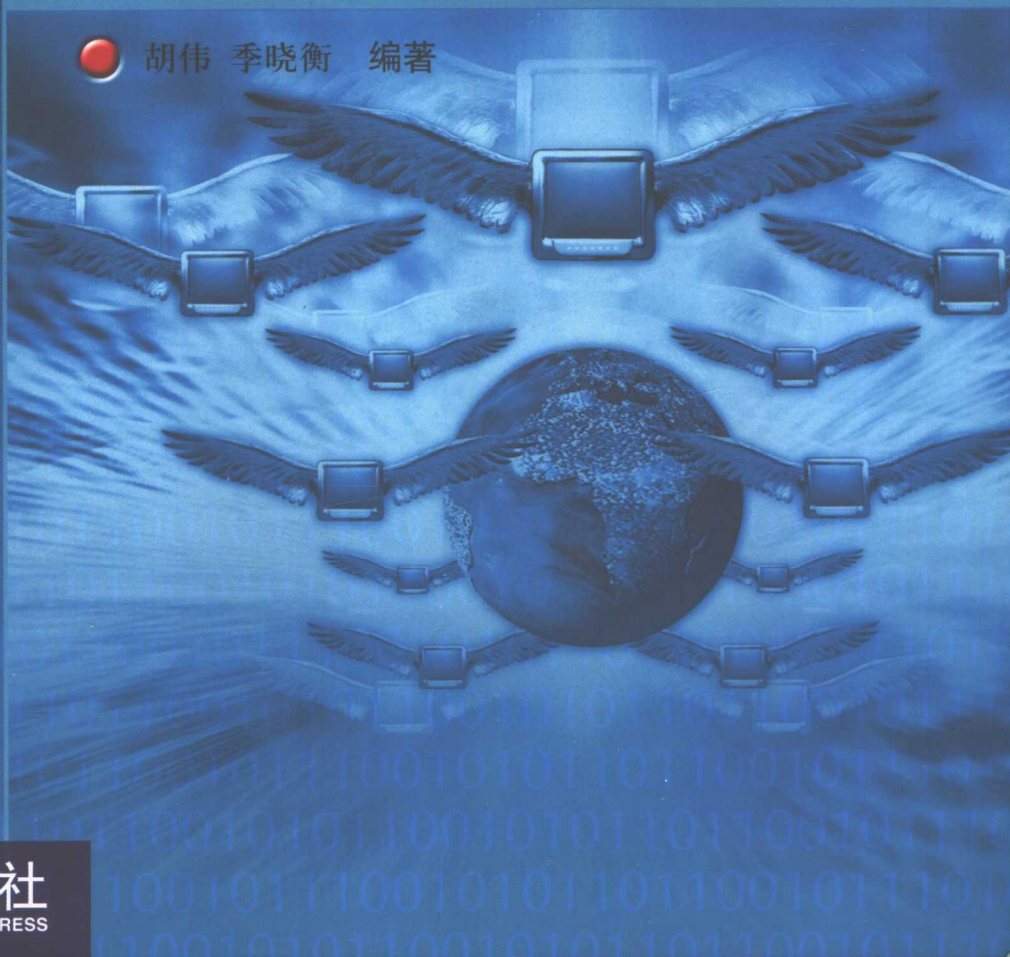
单片机

应用技术系列

单片机C程序设计 及应用实例

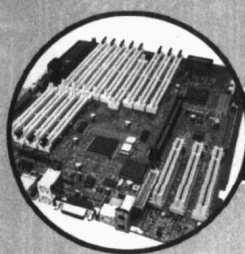


胡伟 季晓衡 编著



人民邮电出版社

POSTS & TELECOMMUNICATIONS PRESS

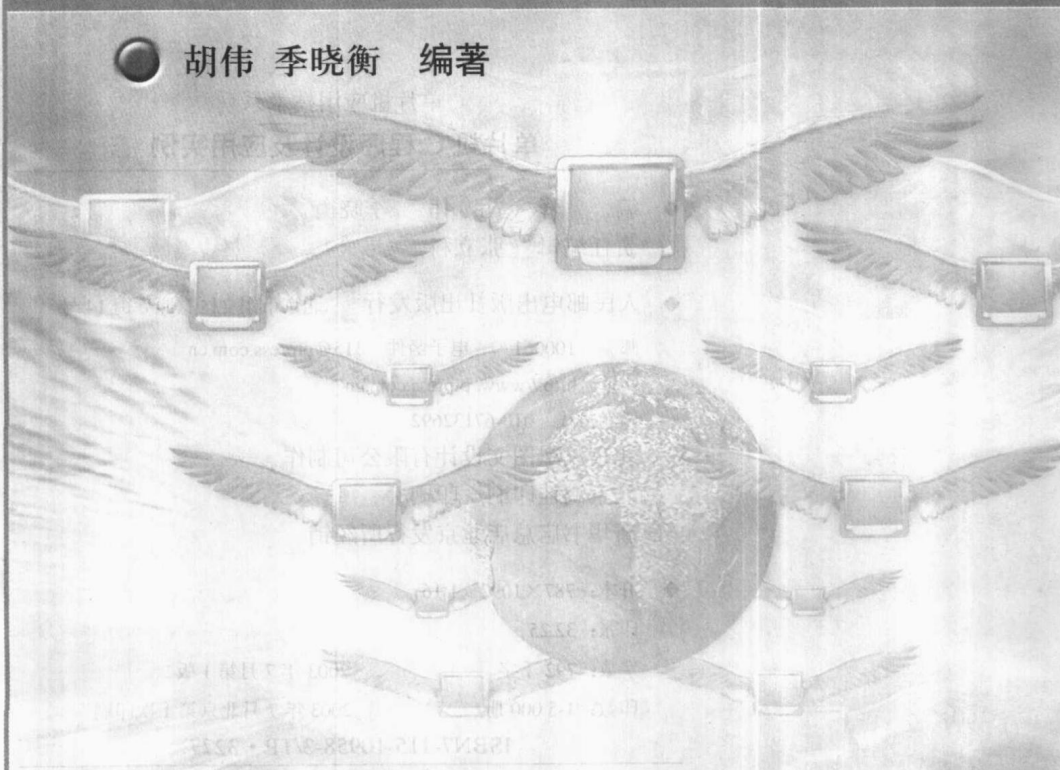


单片机

应用技术系列

单片机C程序设计 及应用实例

● 胡伟 季晓衡 编著



人民邮电出版社

图书在版编目 (CIP) 数据

单片机 C 程序设计及应用实例 / 胡伟编著. —北京: 人民邮电出版社, 2003.7

ISBN 7-115-10958-3

I. 单... II. 胡... III. 单片微型计算机—C 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字 (2003) 第 041138 号

内容提要

随着嵌入式领域的不断发展, 以及 C 语言在嵌入式应用中的不断普及, 嵌入式系统的开发者掌握单片机 C 语言 (C51) 已经成为不可逆转的趋势。KEIL 公司的 IDE C51 开发环境 μ Vision2 是 8051 系列单片机的开发工具。由于其功能强大, 方便易用, 目前已经成为单片机 C 语言开发者最常用的工具之一。本书的主要目的在于:

1. 简单而清晰地介绍单片机的原理、功能及使用方法, 使读者具备 C51 编程时所必需的硬件知识, 对于实际工作中的一般应用不必再去查找其他资料。

2. 根据作者在长期的实际开发中的经验, 通过专门编写的通用、模块化实例来介绍 C51 语言, 使读者尽快了解其语法语义, 并在实际工作中只需要对本书中的实例稍加修改, 就能够在实际工作中使用。

3. 通过实例对 μ Vision2 开发环境和仿真环境进行介绍, 使读者逐渐了解嵌入式系统软件开发的流程。

通过上述三点的综合, 使读者了解完整的嵌入式系统的 C 语言开发, 并对硬件有一定的了解。本书突破了传统的将单片机的软硬件的学习截然割裂的做法, 相信对读者会有所帮助。

总之, 对于 C51 编程的人员来说, 本书将是不可多得的助手, 而对于需要进行其他类型的嵌入式 C 语言开发人员来说, 本书亦极具参考价值, 是一本很好的参考书。

单片机应用技术系列

单片机 C 程序设计及应用实例

◆ 编 著 胡 伟 季晓衡
责任编辑 张立科

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号

邮编 100061 电子函件 315@ptpress.com.cn

网址 <http://www.ptpress.com.cn>

读者热线 010-67132692

北京汉魂图文设计有限公司制作

北京鸿佳印刷厂印刷

新华书店总店北京发行所经销

◆ 开本: 787×1092 1/16

印张: 32.25

字数: 792 千字

2003 年 7 月第 1 版

印数: 1-5 000 册

2003 年 7 月北京第 1 次印刷

ISBN7-115-10958-3/TP · 3277

定价: 47.00 元

本书如有印装质量问题, 请与本社联系 电话: (010) 67129223

前 言

1. 本书特点

C 语言已成为当前举世公认的高效简洁而又贴近硬件的编程语言之一。将 C 语言向单片机 8051 上的移植,始于 20 世纪 80 年代的中后期,经过十几年的努力,C 语言终于成为专业化的单片机上的实用高级语言。过去长时间困扰人们的“C 语言产生代码太长、运行速度太慢,不适合单片机使用”的致命缺点已经基本被克服。而且,由于开发速度、软件质量、结构化、可维护性方面的原因,使 C 语言渐渐有取代汇编的趋势。所以,不管对于新进入这一领域的开发者来说还是对于有多年单片机开发经验的人来说,学习 C51 已经成为一个比较紧迫的任务。本书就是在这样的背景下写成的。

本书主要特色在于介绍 C51 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 C51 开发与仿真环境的使用,透彻分析了 C51 的语义语法及高级的连接定位功能,以及开发过程中可能存在的问题和难点,并进一步介绍了 C51 综合实例的开发。通过对这些编程案例和技巧的学习,可以让读者清楚地感受到作者的开发意图,并且能将这些经验同读者自己的开发实践结合起来。

本书的另一个特点在于突破了传统的软硬件截然割裂的方法,使读者对嵌入式系统的开发有一个整体的了解,并且在进行连接定位等与硬件关系紧密的操作时不至于晕头转向。相信本书的这个特点会节省读者进入嵌入式 C 领域的时间,同时能更清楚认识应用系统开发的过程,深入理解 C51 的编译机制。

2. 本书的结构和内容

本书是一本专门讲解 C51 编程的书,并以“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 程序编写、函数和类库支持、连接定位以及在实际系统中的具体运用。

本书全书总共分为 9 章,首先介绍了 C51 的应用,然后介绍开发、仿真系统 μ Vision2,接着介绍 C 语言的语义语法,以及在具体的一款典型 51 单片机上的应用,随后介绍编译连接及库函数的知识,最后介绍了几个实际应用的综合实例。

本书各章的内容具体是这样安排的:

“第 1 章 引言”,本章主要介绍了 C51 在嵌入式领域的应用,并对 C51 语言和汇编语言作了深入的比较。

“第 2 章 Windows 集成开发环境 μ Vision2”,本章主要介绍了开发和软件仿真环境 KEIL 公司的 μ Vision2 (一般习惯上被称作 KEIL51) 的使用方法,包括项目的创建、参数的设置以及不同参数设置以后在实际系统中的表现等。本章的另一个重点是介绍了 μ Vision2 的强大的软件仿真功能的使用,软件仿真函数的编写。可以说,程序在开发过程中如果通过了仿真的测试,开发任务的绝大部分就完成了,而软件仿真与硬件仿真相比,由于其简单性和方便性一直为人们所重视。所以本章的仿真部分对开发来说是相当重要的内容。

“第3章 C语言基础”，本章介绍了C语言的基本语义语法，并编写了很多通过串口作输入/输出的实例。本章实例中包含有若干排序及数据结构的算法，读者在实际工作中可以直接使用。

“第4章 使用C语言对AT89C51编程”，本章通过一款比较流行的单片机的C语言开发来帮助读者学习与硬件紧密相关的操作。当然，本章对AT89C51这一款芯片做了简单而明了的介绍。

“第5章 预处理器”，本章主要讨论了宏定义等内容。

“第6章 C51编译器”，本章介绍了C51编译过程中的控制指令，以及KEIL公司的C51编译器支持的特殊的扩展关键字的含义及其使用方法。另外，本章介绍了C51的一些稍加修改即可使用的启动配置及其修改方法。最后介绍了一些常用的库函数。

“第7章 目标文件的连接定位与转换”，本章介绍了对最终烧录到单片机上的程序来说相当重要的定位连接的工作。这一步做的好坏会对程序的占用空间及执行效率有一定的影响。

“第8章 硬件仿真”，本章介绍了利用其他公司提供的硬件仿真器在系统现场仿真的步骤及方法，这一步对于单片机不是很熟练的开发来说，是在程序烧录到ROM之前要做的一步重要工作。

“第9章 C51的编程技巧与实例”，本章介绍了若干个比较复杂的项目实例。这些实例都比较典型，并且有代表性，稍加修改就可以应用到实际的项目中。

本书为立志掌握C51编程技术的人提供了一条有效的捷径，对于不同层次的嵌入式C语言编程人员来说，本书都极具参考价值，是一本不可多得的参考书。

另外，本书涉及到的源代码，读者可以到<http://www.cs-book.com>页面上相应的位置下载。

由于本书所涉及的内容比较广泛，对C51的介绍也比较深入，以及作者水平所限，尽管对本书中所涉及的内容一再推敲和仔细调试，仍有可能出现错误和纰漏，希望广大读者批评指正。

作 者

目 录

第 1 章 引言.....	1
1.1 8 位单片机在嵌入式系统中的应用	1
1.2 C 语言在嵌入式系统开发中的应用	1
1.2.1 C 语言与汇编语言的优势对比	2
1.2.2 C51 与 ANSI-C 的主要区别	3
1.2.3 编写高效的 C51 程序及优化程序	3
第 2 章 Windows 集成开发环境 μ Vision2	6
2.1 启动 μ Vision2	6
2.2 创建程序	7
2.2.1 创建项目	8
2.2.2 项目目标和文件组	13
2.2.3 在 Project Window 中查看文件和组的属性	14
2.2.4 Target 对话框简介	14
2.2.5 地址复用技术 —— Code banking	14
2.2.6 μ Vision2 功能	15
2.2.7 代码优化	21
2.2.8 一些技巧	24
2.3 CPU 仿真	26
2.3.1 μ Vision2 调试器	26
2.3.2 调试命令	35
2.3.3 存储器空间	37
2.3.4 表述 (Expressions)	38
2.3.5 μ Vision2 和 C 的不同	49
2.3.6 一些技巧	49
2.4 调试函数	51
2.4.1 创建函数	51
2.4.2 激活函数	52
2.4.3 函数分类	53
2.4.4 调试函数和 C 的不同	60
第 3 章 C 语言基础	62
3.1 基本概念	62
3.1.1 入门	62
3.1.2 变量与算术表达式	65
3.1.3 for 语句	70
3.1.4 符号常量	72

3.2 数据类型、运算符和表达式	73
3.2.1 C 语言的数据类型	73
3.2.2 常量与变量	74
3.2.3 整型数据	75
3.2.4 实型数据	77
3.2.5 字符型数据	78
3.2.6 运算符	83
3.2.7 表达式	88
3.3 程序控制语句	91
3.3.1 程序的 3 种基本结构	91
3.3.2 条件控制语句	91
3.3.3 程序应用举例	99
3.4 循环控制语句	100
3.4.1 while 语句	100
3.4.2 do...while 语句	103
3.4.3 for 语句	104
3.4.4 break 与 continue 语句	106
3.4.5 程序应用举例	108
3.5 函数与程序结构	110
3.5.1 函数的基本知识	111
3.5.2 返回非整数值的函数	115
3.5.3 外部变量	117
3.5.4 作用域规则	118
3.5.5 头文件	119
3.5.6 静态变量	119
3.5.7 寄存器变量	120
3.5.8 分程序结构	120
3.5.9 初始化	121
3.5.10 递归	122
3.6 数组	124
3.6.1 一维数组	124
3.6.2 二维数组	127
3.6.3 多维数组	131
3.6.4 数组的初始化	132
3.7 指针	134
3.7.1 指针与指针变量	134
3.7.2 指针变量的定义与引用	134
3.7.3 指针运算符与指针表达式	136
3.7.4 指针与数组	139
3.7.5 指针的地址分配	154

3.7.6 指针数组	156
3.7.7 指向指针的指针	163
3.8 结构体与共用体	167
3.8.1 结构体类型变量的定义和引用	167
3.8.2 结构体数组的定义和引用	171
3.8.3 结构体指针的定义和引用	178
3.8.4 共用体	182
3.9 小结	186
第4章 使用C语言对AT89C51编程	187
4.1 使用89C51芯片硬件资源的例程序	187
4.1.1 CPU结构	188
4.1.2 存储器、特殊功能寄存器及位地址空间	196
4.1.3 中断系统	200
4.1.4 输入/输出(I/O)口的使用	212
4.1.5 定时/计数器的使用	216
4.1.6 串口通信程序的编制	222
4.1.7 节电工作方式的程序	234
4.1.8 单步的程序	235
4.2 含51核的系列微处理器简介	236
4.2.1 AT89C51	236
4.2.2 DS80C320	237
4.2.3 其他芯片	238
4.3 一个完整的使用多种资源的例程序	239
4.3.1 项目需求	239
4.3.2 步进电机背景知识	239
4.3.3 解决方案	242
第5章 预处理器	262
5.1 宏定义	262
5.1.1 不带参数的宏定义	262
5.1.2 带参数的宏定义	263
5.2 文件包含	263
5.3 条件编译	264
5.4 其他预处理命令	266
5.4.1 字符串化的操作	266
5.4.2 连接符	267
5.4.3 预定义好的常量	267
第6章 C51编译器	269
6.1 C51编译器控制指令详解	272

6.1.1	源控制指令	272
6.1.2	列表控制指令	276
6.1.3	目标控制指令	282
6.2	C51 扩展的关键字	296
6.2.1	内存区域	296
6.2.2	数据变量分类	297
6.2.3	存储模式分类	300
6.2.4	变量定位到绝对地址	301
6.2.5	指针	302
6.2.6	函数	313
6.3	C51 与其他语言的混合编程	325
6.3.1	C51 编译器格式规范	325
6.3.2	C51 与汇编语言的混合编程	330
6.4	C51 的高级配置文件	336
6.4.1	STARTUP.A51	336
6.4.2	START751.A51	344
6.4.3	INIT.A51	346
6.4.4	INIT751.A51	356
6.5	C51 的库函数	360
6.5.1	字符函数 CTYPE.H	360
6.5.2	一般 I/O 函数 STDIO.H	368
6.5.3	字符串函数 STRING.H	377
6.5.4	标准函数 STDLIB.H	385
6.5.5	数学函数 MATH.H	388
6.5.6	绝对地址访问 ABSACC.H	390
6.5.7	内部函数 INTRINS.H	390
6.5.8	变量参数表 STDARG.H	392
6.5.9	全程跳转 SETJMP.H	393
6.5.10	访问 SFR 和 SFR_bit 地址 REGxxx.H	395
第 7 章	目标文件的连接定位与转换	396
7.1	连接定位器	396
7.2	连接/定位控制指令	397
7.3	μVision2 中连接定位器的使用	398
第 8 章	硬件仿真	408
8.1	伟福仿真器系统及其使用方法	408
8.1.1	安装伟福仿真调试系统	408
8.1.2	使用伟福系列仿真器系统	409
8.2	万利 MedWin 仿真系统简介	420

8.2.1	MedWin 仿真系统的安装	420
8.2.2	MedWin 仿真系统的使用	421
8.2.3	MedWin 仿真系统的特点	422
8.3	一个简单程序调试的例子	423
8.3.1	单片机电路图	423
8.3.2	测试程序源代码	424
8.3.3	运行	425
8.4	小结	426
第 9 章	C51 的工程开发案例与技巧	427
9.1	C51 应用程序设计的基本方法	427
9.1.1	简介	427
9.1.2	C51 编译器	427
9.2	C51 系统设计所需要的一些知识	429
9.2.1	硬件以及电路的知识	429
9.2.2	软件以及编程语言的知识	432
9.3	C51 系统设计所需要的一些设备	433
9.4	C51 系统设计所需要注意的一些问题	434
9.4.1	单片机资源的分配	434
9.4.2	单片机的寻址	435
9.4.3	C51 函数的返回值	435
9.4.4	单片机的看门狗功能	435
9.4.5	单片机的外设	435
9.4.6	单片机的功耗	436
9.5	有关 C51 的一些问题	436
9.6	应用案例 1 —— 外部 ROM 的实现	439
9.6.1	简单的电路	439
9.6.2	电路设计的背景及功能	439
9.6.3	电路的设计	440
9.6.4	电路的 C51 程序代码	441
9.7	应用案例 2 —— 键盘和发光数码管显示	441
9.7.1	电路设计的背景及功能	441
9.7.2	电路的设计	442
9.7.3	键盘扫描电路的 C51 程序代码	443
9.7.4	电路的改进 —— 键盘的消抖动程序	446
9.7.5	电路的显示部分 —— LED 数码管电路	451
9.8	应用案例 3 —— 秒表程序的实现	455
9.8.1	电路设计的背景及功能	455
9.8.2	电路的设计	455
9.8.3	电路的 C51 程序代码	456

9.9 应用案例 4 —— 24 小时时钟的实现	462
9.9.1 电路设计的背景及功能	462
9.9.2 电路的设计	462
9.9.3 电路的 C51 程序代码	463
9.10 应用案例 5 —— 多机串行通信的实现	470
9.10.1 电路设计的背景及功能	470
9.10.2 主机端电路的设计	470
9.10.3 主机端电路的 C51 程序代码	471
9.10.4 从机端电路的设计	477
9.10.5 从机端电路的 C51 程序代码	478
9.11 应用案例 6 —— A/D、D/A 转换器使用	478
9.11.1 电路设计的背景及功能	478
9.11.2 电路的设计	479
9.11.3 电路的 C51 程序代码	481
9.12 应用案例 7 —— 与计算机的互连	485
9.12.1 电路设计的背景及功能	485
9.12.2 电路的设计	485
9.12.3 电路的 C51 程序代码	486
9.12.4 计算机端的 Visual C++ 程序代码	487

第 1 章 引言

1.1 8 位单片机在嵌入式系统中的应用

嵌入式系统是与通用计算机相对的，完成某些相对简单或者某些特定功能的 CPU 系统。与从 8 位机迅速向 16 位、32 位、64 位过渡的通用计算机相比，嵌入式系统有其功能的要求和成本的考虑，这决定了嵌入式系统内高、中、低端系统共存的局面。在低端系统中，嵌入式 8 位单片机从 20 世纪 70 年代初期诞生至今，虽历经从单片微型计算机到微控制器、MPU 和 SoC 的变迁，但它仍然是嵌入式低端应用的主要机型，而且在未来相当长的时间里，仍会保持这个势头。这是因为嵌入式系统和通用计算机系统有完全不同的应用特性，因而它们走向了完全不同的技术发展道路。

嵌入式系统嵌入到对象体系中，并在对象环境下运行。与对象领域相关的操作主要是对外界物理参数进行采集、处理，对外界对象实现控制，并与操作者进行人机交互等。而对象领域中物理参数的采集与处理、外部对象的控制以及人机交互所要求的响应速度有限，而且不会随时间变化，在 8 位单片机能基本满足其响应速度要求后，数据宽度就不是技术发展的主要矛盾了。因此 8 位单片机不会被淘汰，其技术发展方向转为最大限度地满足对象的采集、控制、可靠性和低功耗等品质要求。

随着现代通信技术的发展，智能化系统对 DSP 需求的增长，提高单片机运算速度是一个重要研究方向。8 位单片机在不扩展数据总线的情况下提高运行速度仍有潜力可挖。例如，采用 RISC 结构实现并行流水线作业，CISC 结构的 C8051F 采用 CIP-8051 结构，可以使单周期指令速度提高到原 8051 的 12 倍。

鉴于嵌入式低端应用对象的有限响应要求、嵌入式系统低端应用的巨大市场以及 8 位机具有的速度潜力，可以预期在未来相当长的时间内，8 位机仍然是嵌入式应用中的主流机型。

随着半导体技术的发展，8 位单片机在 CPU 结构、CPU 外围、功能外围、外围接口和集成开发环境方面都会得到迅速的发展。

1.2 C 语言在嵌入式系统开发中的应用

随着市场竞争的日趋激烈，电子工程师需要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，使得代码的规范性、模块化的要求越来越高，以方便多个工程师以软件工程的形式进行协同开发。汇编语言作为传统的嵌入式系统的编程语言，已经不能满足这样的实际需要了。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开发工具，并得以广泛应用。1985 年出现了针对

6-53168/1.3

8051 的 C51 编译器,而其他流行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 系列都有功能强大的 C 语言编译系统以及丰富的 C 语言库函数。

1.2.1 C 语言与汇编语言的优势对比

在国内,汇编语言仍然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系统的习惯性,使 C 语言在不少地方遭到冷落。诚然,优秀程序员写出的汇编语言程序的确有执行效率高的优点,但其可移植性和可读性差,使其开发出来的产品在维护和功能升级方面都有极大的困难,从而导致整个系统的可靠性和可维护性也比较差。而使用 C 语言进行嵌入式系统的开发,有着汇编语言编程所不可比拟的优势,介绍如下。

1. 编程调试灵活方便

C 语言编程灵活,同时,当前几乎所有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 语言编译系统编译出来的代码效率只比直接使用汇编低 20%左右,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能够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,尽量不做或者少做改动,以减少重复劳动。如果使用 C 语言开发,数据交换可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不经修改,直接被其他项目所用,这样就可以很好地利用已有的大量 C 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

由于不同系列的嵌入式系统的 C 语言编译工具都是以 ANSI-C 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,只需将部分与硬件相关的地方和编译连接的参数进行适当修改,就可方便移植到另外一种系列上,例如,C51 下编写的程序通过改写头文件以及少量的程序行,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达到平台的无关性。

5. 便于项目维护管理

用 C 语言开发的代码便于开发小组计划项目、灵活管理、分工合作以及后期维护,基本上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、可靠性以及可升级性。

1.2.2 C51 与 ANSI-C 的主要区别

不同系列的嵌入式系统的 C 编译器根据它所对应的不同芯片系列，有其各自的特点，在这里，以 KEIL 公司的针对 51 系列的 KEILC51 编译器为例，简要说明它与 ANSI-C 的主要区别，因为了解嵌入式系统的 C 编译器与标准 ANSI-C 的区别是用 C 编译器系统进行嵌入式系统开发的前提条件。

不同的嵌入式 C 编译系统之所以与 ANSI-C 有不同的地方，主要是由于它们所针对的硬件系统不同。国内开发人员所熟悉的 51 系列单片机，有着最丰富的 C 编译系统，其中最为出色的当属 KEIL（也就是 FRANKLING，但 FRANKLING 只相当于 KEIL 的早期产品，它是 KEIL 公司在美国销售时曾使用的一个品牌）。

(1) 从头文件来说，51 系列有不同的厂家，不同的系列产品，如仅 ATMEL 公司就有大家所熟悉的 89c2051、89c51、89c52 以及大家不是很熟悉的 89s8252 等系列产品。它们都是基于 51 系列的芯片，唯一不同之处在于内部资源如定时器、中断、I/O 等数量以及功能的不同。为了实现这些功能，只需要将相应的功能寄存器的头文件加载在程序内就可实现它们所指定的不同功能。因此，KEILC51 系列头文件集中体现了各系列芯片的不同功能。

(2) 从数据类型来说，由于 8051 系列器件包含位操作空间和丰富的位操作指令，因此嵌入式 C 与 ANSI-C 相比要多一种位类型，使得它能如同汇编语言一样，能够灵活地进行位指令操作。

(3) 从数据存储类型来说，8051 系列有片内、片外程序存储器，片内、片外数据存储器，在片内程序存储器还分直接寻址区和间接寻址区，它分别对应 code、data、xdata、idata 以及根据 51 系列特点而设定的 pdata 类型，使用不同存储器将会影响程序执行的效率。在编写 C51 程序时，最好指定变量的存储类型，这样将有利于提高程序执行效率（此问题将在后面专门讲述）。与 ANSI-C 稍有不同，它只分 SMALL、COMPACT、LARGE 模式，不同的模式对应不同的硬件系统和不同的编译结果。

(4) 从数据运算操作、程序控制语句以及函数的使用上来讲，它们几乎没有什么明显的不同，在函数的使用上，由于嵌入式系统的资源有限，它的编译系统不允许太多的程序嵌套。C 语言丰富的库函数对程序开发提供了很大的帮助，但它的库函数和 ANSI-C 也有一些不同之处。首先从编译的角度来说，由于 51 系列是 8 位机，扩展 16 位字符不被 C51 所支持，其次，ANSI-C 所具备的递归特性不被 C51 支持，在 C51 中，要使用递归特性，必须用 REENTRANT 进行声明才能使用。

(5) KEIL C51 与标准 ANSI-C 从库函数的方面来说有很大的不同。由于部分库函数不适合于嵌入式处理系统，因此它们被排除 KEIL C51 之外，如字符屏幕和图形函数。有一些库函数可以继续使用，但这些库函数都是厂家针对硬件特点做出的相应开发，它们与 ANSI-C 的构成及用法都有很大不同，如 printf 和 scanf。在 ANSI-C 中这两个函数通常用于屏幕打印和接收字符，而在 KEILC51 中，它们则主要用于串行数据的收发。

1.2.3 编写高效的 C51 程序及优化程序

帮助读者尽快熟悉 C51 并写出有良好结构和高效率的程序就是本书的目标之一。尽管嵌入式 C 语言是一种强大而方便的开发工具，但开发人员如果要快速编出高效而易于维护的嵌

入式系统程序，就必须对 C 语言编程有较为透彻的掌握。不仅如此，优秀的程序员还应该对实际电子硬件系统有深入的理解。因此在学习嵌入式 C 之前，了解汇编语言编程和硬件结构是十分必要的。下面简单介绍一下开发的要点，更具体的内容需要读者在长期的工作中慢慢体会。

1. 良好的、规范化的编程风格

以软件工程的方式进行总体程序设计安排，形成良好的、规范化的编程风格，对高效地编出易于维护的嵌入式系统程序有至关重要的作用，图 1-1 给出了较为合理的工作流程框图。

正规、一致的编程风格十分重要，建议使用“The C Programming Language”一书中使用的书写风格，对变量命名采用大部分 Windows 程序员所采纳的 Hungarian 命名法则，再加上模块化的编程方式，详尽的程序说明文档，这样可以促进高效编程的实现，并适应多人分工协作，也有益于项目开发的规范化以及后期维护。同时，这样写出的程序逻辑清楚，代码简单，注释明白，也便于重新利用。

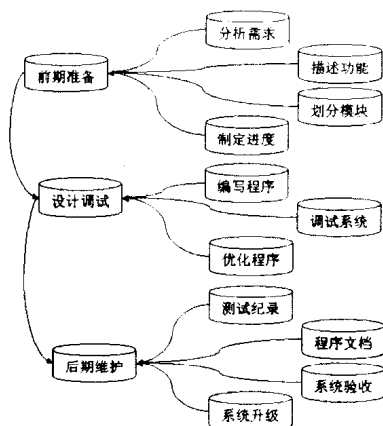


图 1-1 合理的工作流程框图

2. 灵活选择变量的存储类型，是提高程序运行效率的重要途径

嵌入式系统资源有限，它有各种有限的存储器资源，对需进行位操作的变量，其存储类型为 `bdata`，直接寻址存储器类型为 `data`，间接寻址存储器类型为 `idata`（间接寻址存储器也可访问直接寻址存储器区），外部寻址存储器类型为 `xdata`，当对不同的存储器类型进行操作时，尽管反映在 C 语言中相同，但是编译后的代码执行效率各不相同，内部存储器中直接寻址空间和间接寻址空间也不相同。

为了提高执行效率，对存储器类型的设定，应该根据以下原则：只要条件满足，尽量先使用内部直接寻址存储器（`data`），其次设定为间接寻址存储器（`idata`）。在内部存储器数量不够的情况下，才使用外部存储器。而且在外部存储器中，优先选择 `pdata`，最后才是 `xdata`。同时，在内部和外部存储器共同使用的情况下，要合理分配存储器，对经常使用和频繁计算的数据，应该使用内部存储器，其他的则使用外部存储器。要根据它们的数量进行分配，在例程中，内部数据存储器数量充足时，如果将所有变量都声明为直接寻址存储类型，将使程序访问数据存储器所用的时间最少，从而提高程序的运行效率。

3. 合理设置变量类型以及设置运算模式可以减少代码量，提高程序执行效率

由于 51 系列是 8 位机，它只能够直接处理 8 位无符号数的运算，而处理其他类型的数据结构则需通过额外的算法来实现，因此，在对变量进行类型设置时，要尽可能选用无符号的字符类型，少用有符号以及多字节类型，因此，程序设计中，都尽量采用无符号数以提高运算速度，以此避免进行多余运算。

在运算时，可以进行定点运算的尽量进行定点运算，避免进行浮点运算。如乘 2 或除 2，就可以使用移位操作来代替。这样不仅可以减少代码量，同时，还能提高程序执行效率。

4. 灵活设置变量，高效利用存储器，提高程序执行效率

嵌入式系统资源有限，而 C 语言编程通常是采用模块化方法，因此在 C 嵌入式编程中，如何实现高效的数据传输对提高程序执行效率十分关键。

通常，C 程序中，子程序模块中与其他变量无关的变量尽可能使用局部变量，对整个程序都要使用的变量将其设置为全局变量更合适，这样，子程序模块可以直接申明要使用的变量为外部变量即可。

其次要结合 C 语言的特点进行灵活的数据传输。灵活使用 C 语言中所特有的指针、结构、联合等数据类型，可以提高编程效率，也可以方便数据传输。主程序和子程序传递数据量不能过多，变量也要尽量少，否则将影响执行效率。子程序模块和主程序模块都定义了相同的数据类型，在进行数据传输时，主程序模块和子程序模块定义相同的数据类型，在进行数据传输时，只需将指针传送到子程序模块，这样，既可以使不同的程序有很好的独立性和封装性，又能实现不同程序间数据的灵活高效传输。这样做可使不同的开发人员开发独立性很强的通用子程序模块。

5. 将汇编程序嵌入嵌入式 C 程序中，完成实时响应或大运算量的任务

对一些实时性或者运算能力要求很高的程序，如中断程序处理、数据采集程序、实时控制程序以及一些实时的带符号或多位运算，都建议将汇编语言嵌入 C 程序中进行处理。

在 KEILC51 的 C 编译系统中，C 程序能够与汇编程序实现方便灵活的接口，C 程序中调用汇编十分方便灵活，二者之间调用的主要难度在于数据的准确传输。汇编与 C 中实现数据传输可通过两种方式，一是利用工作寄存器进行数据传送，这种方式安全，但根据传送数据类型的不同，只能传送 1~3 个参数。另外一种方案是指定特定的数据区，二者在特定的工作区内进行数据传输，这种方式，其传递数据量可自行控制，但不安全，需要开发人员小心控制。对于特定的编辑器，可以参考其 C 语言的函数编译出来的汇编程序，按照 C 语言函数的格式来写汇编，在 C 中可以直接当作带参数的函数来调用。

6. 利用丰富的库函数，可以大大提高编程效率

在 KEILC51 中，开发厂家提供了许多常用的库函数，主要是数学函数、内存分配等标准函数供编程人员使用。开发人员可灵活的使用这些函数，以提高开发效率。

第2章 Windows 集成开发环境 μ Vision2

KEIL 51 是德国 KEIL 公司开发的单片机 C 语言编译器, 其前身是 FRANKLIN C51, 现在的最新版本 V6 功能已经相当不错, 特别是兼容 ANSI C 后又增加很多与硬件密切相关的编译特性, 使得在 8051 系列单片机上开发应用程序更为方便和快捷。 μ Vision2 是一种集成化的文件管理编译环境, 本书的编译环境使用 KEIL 51, 集成了文件编辑处理、编译链接、项目管理、窗口、工具引用和软件仿真调试等多种功能, 是相当强大的 C51 开发工具。在 μ Vision2 的仿真功能中, 有两种仿真模式: 软件模拟方式和目标板调试方式。在软件模拟方式下, 不需要任何 8051 单片机硬件即可完成用户程序仿真调试, 极大地提高了用户程序开发效率。在目标板调试方式下, 用户可以将程序装到自己的 8051 单片机系统板上, 利用 8051 的串口与 PC 机进行通信来实现用户程序的实时在线仿真。

在本书中, 不对 KEIL51 和 μ Vision2 两个术语做严格的区分, 一般来说, 都指的是 μ Vision2 集成开发环境。本章首先横向地简单介绍 μ Vision2 中各个菜单栏的作用, 然后再通过创建程序的流程和调试的流程来详细介绍各菜单的使用以及仿真功能的应用。

2.1 启动 μ Vision2

μ Vision2 是 KEIL 公司关于 8051 系列 MCU 的开发工具, 可以用来编译 C 源码、汇编源程序、连接和重定位目标文件和库文件、创建 HEX 文件、调试目标程序等。 μ Vision2 的启动界面如图 2-1 所示, 几秒钟后, 主画面中间的版本提示将会消失, 由文本编辑窗口所代替。如果上次退出 μ Vision2 时没有关闭文件, 那么将会恢复显示上次文件的编辑窗口状态, 否则要重新打开文件。



图 2-1 μ Vision2 的启动界面