



(本书附光盘1张)



徐爱钧 彭秀华 编著

单片机高级语言 **C51** Windows 环境编程与应用



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

www.phei.com.cn

单片机高级语言 C51 Windows 环境编程与应用

徐爱钧 彭秀华 编著

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

本书以 Keil 公司最新版本的 C51 编译器为核心,阐述了在 Windows 环境下采用单片机高级语言 C51 进行应用系统程序设计的基本原理和方法,详细介绍了 Keil C51 的 Windows 集成开发环境 μ Vision51 和软件仿真器 dScope51 的强大功能和具体使用方法。在 Windows 集成开发环境 μ Vision51 中,可以完成从源程序编写、编译、连接定位到目标文件的仿真调试等的全部工作。dScope51 是一种基于 Windows 的多窗口软件仿真器,它可以在完全没有 8051 单片机硬件的情况下模拟调试各种应用程序,即使对单片机内部特殊集成功能的编程,也可以找到对应的模拟窗口。本书对最新版本 C51 编译器新增加的控制命令作了全面介绍;给出了最新 C51 运行库函数及其应用范例;对 Keil C51 的各种应用工具,如 BL51 连接定位器、A51 宏汇编器、LIB51 库管理程序以及 OH51 符号转换程序等都作了详细介绍;还介绍了单片机实时多任务操作系统 RTX51 及其子集 RTX51 TINY 的具体功能与应用方法。

本书强调先进性和实用性。全书各章列举了大量程序实例,并附赠一张配套光盘。因此本书既适合广大从事单片机应用系统开发研制的工程技术人员的阅读,也可以作为高等工科院校相应专业大学生或研究生的教学用书。

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,翻版必究。

图书在版编目(CIP)数据

单片机高级语言 C51 Windows 环境编程与应用/徐爱钧,彭秀华编著. —北京:电子工业出版社,2001.7
ISBN 7-5053-6756-0

I 单... II. ①徐... ②彭... III. 单片微型计算机-程序设计 IV. TP368.1

中国版本图书馆 CIP 数据核字(2001) 第 035301 号

书 名: 单片机高级语言 C51 Windows 环境编程与应用

编 著 者: 徐爱钧 彭秀华

策划编辑: 李新社

责任编辑: 朱沐红

排版制作: 电子工业出版社计算机排版室监制

印 刷 者: 北京四季青印刷厂

装 订 者: 河北省涿州桃园装订厂

出版发行: 电子工业出版社 URL: <http://www.phei.com.cn>

北京市海淀区万寿路 173 信箱 邮编 100036

经 销: 各地新华书店

开 本: 787×1092 1/16 印张:33 字数:844 千字 附光盘:1 张

版 次: 2001 年 7 月第 1 版 2001 年 7 月第 1 次印刷

书 号: ISBN 7-5053-6756-0
TP·3786

印 数: 8000 册 定价: 48.00 元(含光盘)

凡购买电子工业出版社的图书,如有缺页、倒页、脱页、所附磁盘或光盘有问题者,请向购买书店调换。

若书店售缺,请与本社发行部联系调换。电话 68279077

前 言

8051 单片机在工业测量控制领域内的应用极为广泛。由于 Intel 公司将 8051 CPU 内核向全世界各大半导体公司进行推广,目前已有 Philips、Siemens、Dallas、OKI、Advance Micro Device、Atmel 等公司生产 100 多种型号的 8051 系列单片机。因此,可以说 8051 单片机已经成为事实上的工业标准。在早期研制单片机应用系统时,大多以汇编语言作为软件工具。汇编语言不是一种结构化的程序设计语言,它的编写效率很低。随着单片机硬件性能的提高(目前 8051 最高时钟频率可达 40MHz 以上),在编写单片机应用系统程序时,更看重程序本身的编写效率。为了适应这种要求,许多软件公司都致力于单片机高级语言编译器的开发研究。其中,德国 Keil 公司在开发单片机 C 语言编译器方面取得了相当大的成功,从编译器、调试器、实时操作系统到集成开发环境,全面支持 8051、251、166 等单片机主流产品及其众多的派生系列。

Keil 公司的 C51 编译器 V4.0 版本以下的产品曾通过美国 Franklin 公司在市场上已销售了多年,例如传入到我国并得到广泛使用的 Franklin C51 V3.2 和 V3.96 版本等。近年来,Keil 公司的产品不断升级,已经推出 V5.0 以上版本的 C51 编译器和 Windows 集成开发环境,主要包括 C51 编译器、A51 宏汇编器、BL51 连接定位器等应用工具及 μ Vision51 集成编译环境、dScope51 软件仿真器等基于 Windows 的开发平台。

C51 编译器的好坏直接影响着生成代码的效率、大小和可靠性。V5.0 以上版本的 Keil C51 编译器具有高效、可靠的特点,代表着这一领域的工业标准。其主要特点有:支持九种基本数据类型,变量可存放在 bit、data、bdata、idata、xdata 和 pdata 等不同类型的存储器空间;支持用 C 语言编写中断服务函数,保留源程序中所有符号信息以方便调试;支持位寻址对象;支持 AMD 和 Dallas 公司 80320 等单片机的双数据指针和 Philips 公司 80751 等单片机的指令集以及 Infineon(Siemens)公司 80C517 单片机中的算术单元和多重数据指针。与 V4.0 以前版本的 C51 编译器相比,Keil C51 V5.0 以上版本的性能有了长足的进步。无论在长整型数运算中还是浮点型数运算测试中,新版本的运行时间都比老版本减少 40%。在提高效率的同时,Keil C51 V5.0 以上版本还加强了代码的优化;提供由低到高的 7 级普通优化和针对 8051 的特殊优化措施;尽可能地利用工作寄存器来存储变量和函数参数,减少重复无用的代码,支持数据覆盖。这些都有效地降低了代码长度,提高了程序的可靠性。

μ Vision51 是一种基于 Windows 的集成开发环境,从源程序编写、编译、连接定位一直到目标文件的仿真调试等都可在其中完成。在 μ Vision51 中,可以很方便地对 C51 编译器、A51 汇编器、BL51 连接定位器的各种控制命令和工作环境进行设置。在创建一个工作项目后,只要点击 Build All 按钮就可完成项目中所有文件的编译连接工作,生成 OMF51 或 Intel HEX 格式的绝对目标代码文件。

dScope51 是一种基于 Windows 的多窗口软件仿真器,可以在完全没有 8051 硬件的情况下调试各种应用程序,即使对单片机内部特殊集成功能的编程,也可以找到对应的模拟窗口。dScope51 支持 8051 系列单片机众多的派生种类,如 80751、80410、80320、80552、80517 等。这些芯片的驱动程序以 CPU 动态链接库(DLL)的形式存在。向 dScope51 装入 CPU 动态驱动库文件后,主菜单中会生成相应的 Peripherals 选项,并针对不同单片机的驱动程序,提供它们

特有的内部特殊集成功能接口仿真。例如,对应 80517. DLL 就有 A/D 转换器、多重数据指针、算术运算单元等项内容,这给调试工作带来极大的方便。dScope51 还可以通过目标监控程序与用户硬件系统接口,直接对用户目标硬件系统进行调试,从而节省了用户购买昂贵硬件仿真器的费用。

本书作者曾以基于 DOS 环境的 V3.96 版本 Franklin C51 编译器为核心编写了《单片机高级语言 C51 应用程序设计》一书;出版后受到广大读者的欢迎,已经连续 5 次重印。现在, Franklin 公司已经不再作为 Keil 公司的 OEM,而 Keil C51 仍在不断发展。为提高编译效率和改进功能,新版本的 Keil C51 编译器增加了许多新的控制命令,如 FLOATFUZZY、WARNINGLEVEL 等;数据的处理方法也有重大改进,如浮点数运算已经可以实现全再入处理。在中断服务函数中进行浮点数运算时,不再需要调用 fpsave 和 fprestore 库函数进行浮点数保护;浮点数的存储格式也进行了改进,使浮点数处理效率更高。更为突出的是,Keil 公司推出了基于 Windows 环境的 C51 开发平台。众所周知,Windows 环境要比 DOS 环境先进得多,在 Windows 环境下,各种编译工具的使用和仿真调试过程都要方便得多。为使广大喜爱 C51 的单片机用户能够及时了解和掌握最新版本 C51 编译器的诸多优点以及在 Windows 环境下的编程和应用,我们根据 Keil 公司最新原文资料以及实际应用经验总结新编写了本书。全书共分 8 章。第 1 章概述了 8051 单片机的存储器结构、C51 语言特点以及在 Windows 集成开发环境下进行 C51 程序设计和编译调试的基本方法。第 2 章阐述了 C51 程序设计的基本语法。第 3 章阐述了最新版本 Keil C51 的 Windows 集成开发环境,详细介绍了 C51 开发平台 μ Vision51 和 dScope51 的各种功能和具体使用方法。第 4 章阐述了 Keil C51 编译器,详细介绍了最新版本 C51 编译器的各种编译控制命令、数据调用协议,给出了 Keil C51 的全部库函数及其应用范例。第 5 章阐述了 A51 宏汇编器,详细介绍了各种符号与表达式、汇编伪指令、宏处理以及汇编控制命令。第 6 章阐述了目标文件的连接和转换,详细介绍了连接定位器 BL51 的各种连接控制命令、符号转换程序 OH51、库管理程序 LIB51 的使用方法。第 7 章阐述了 RTX51 实时多任务操作系统及其子集 RTX51 TINY 的具体使用方法。第 8 章阐述了在 Windows 环境下进行 C51 应用程序设计的技巧。通过具体实例详细介绍了 μ Vision51 和 dScope51 工作环境、控制命令设置、目标程序的仿真调试方法;讨论了许多 C51 实际应用中的常见问题和解决方法;给出了若干个采用 C51 进行应用系统程序设计的例子。

为帮助读者更好地学习在 Windows 环境下进行 C51 应用程序设计,本书附带一张配套光盘。其中包含本书各章所有的范例程序,Keil 公司提供的全功能 Windows 版本 C51 评估软件包,大量与 8051 单片机及其开发工具有关的各种信息;其内容涉及各个半导体厂商关于 8051 单片机硬件产品数据手册和应用笔记、开发工具的使用、应用程序的开发等,对于读者学习使用 Keil C51 或进行 8051 单片机应用系统开发都有极大的帮助。另外,我们还制作了一种能与本书介绍的 dScope51 开发平台配套使用的 C51 硬件评估板,可以将用户程序下载到评估板中直接进行应用目标系统的仿真调试,以节省购买专用仿真器的费用,评估板请见封底。

本书在编写出版过程中得到电子工业出版社的热情支持和帮助,在此表示衷心感谢。由于作者水平有限,书中难免会有错误和不妥之处,恳请广大读者批评指正。

作 者

目 录

第 1 章 8051 单片机与 C51 语言的基础	1
1.1 8051 单片机的基本结构与存储器组织	1
1.2 C51 语言的特点	4
1.2.1 概述	4
1.2.2 C51 的基本程序结构	6
1.2.3 Windows 环境下的编程和调试	8
第 2 章 C51 程序设计基础	14
2.1 标识符与关键字	14
2.2 C51 程序设计的基本语法	16
2.2.1 数据类型	16
2.2.2 常量	18
2.2.3 变量及其存储模式	19
2.2.4 用 typedef 重新定义数据类型	23
2.2.5 运算符与表达式	24
2.2.6 基本语句	35
2.3 函数	46
2.3.1 函数的定义	46
2.3.2 函数的调用	48
2.3.3 函数的递归调用与再入函数	53
2.3.4 中断服务函数与寄存器组定义	54
2.3.5 函数变量的存储方式	58
2.3.6 函数的参数和局部变量的存储器模式	66
2.4 数组与指针	66
2.4.1 数组的定义与引用	67
2.4.2 字符数组	68
2.4.3 数组作为函数的参数	71
2.4.4 指针的概念	73
2.4.5 数组的指针	84
2.4.6 指针的地址计算	86
2.4.7 函数型指针	88
2.4.8 返回指针型数据的函数	90
2.4.9 指针数组与指针型指针	91
2.4.10 抽象型指针	97
2.5 结构与联合	100
2.5.1 结构变量的定义与引用	100
2.5.2 结构变量的初值	103

2.5.3	结构数组	103
2.5.4	结构型指针	104
2.5.5	结构与函数	106
2.5.6	联合变量的定义与引用	109
2.6	预处理器	113
2.6.1	宏定义	114
2.6.2	文件包含	118
2.6.3	条件编译	118
2.6.4	其他预处理命令	120
第 3 章	Windows 集成开发环境	122
3.1	文件管理编译环境 μ Vision51	122
3.2	μ Vision51 的下拉菜单	123
3.2.1	File 菜单	124
3.2.2	Edit 菜单	124
3.2.3	Project 菜单	126
3.2.4	Run 菜单	128
3.2.5	Options 菜单	129
3.2.6	Tools 菜单	138
3.2.7	Window 菜单	138
3.2.8	Help 菜单	139
3.3	软件仿真器 dScope51	139
3.3.1	File 菜单	139
3.3.2	View 菜单	141
3.3.3	Setup 菜单	149
3.3.4	Peripherals 菜单	155
3.3.5	Help 菜单	157
3.4	CPU 动态驱动库文件	157
3.4.1	8051.DLL/8052.DLL	157
3.4.2	8051Fx.DLL	158
3.4.3	80515.DLL/80515A.DLL	158
3.4.4	80517.DLL/80517A.DLL	159
3.4.5	80552.DLL	160
3.4.6	80751.DLL	161
3.4.7	80781.DLL	162
3.4.8	80410.DLL	162
3.4.9	80320.DLL	162
3.5	dScope51 的命令	163
3.5.1	显示和更新存储器内容命令	163
3.5.2	用户程序执行控制命令	168
3.5.3	断点管理命令	170

3.5.4	其他通用命令	173
3.6	dScope51 的表达式	180
3.6.1	表达式的组成	180
3.6.2	dScope51 表达式与 C 语言表达式之间的差别	185
3.6.3	dScope51 表达式应用示例	185
3.7	dScope51 的函数	189
3.7.1	用户函数	189
3.7.2	内部函数	192
3.7.3	信号函数	195
3.7.4	dScope51 函数与 C51 函数的差别	197
第 4 章	C51 编译器	198
4.1	C51 编译器控制命令详解	200
4.1.1	源控制命令	200
4.1.2	列表控制命令	200
4.1.3	目标控制命令	206
4.2	C51 编译器的数据调用协议	224
4.2.1	数据在内存中的存储格式	224
4.2.2	目标代码的段管理	225
4.2.3	再入函数的栈结构	227
4.2.4	与汇编语言程序的接口	228
4.2.5	与 PL/M51 语言程序的接口	241
4.2.6	配置文件	241
4.2.7	C51 编译器的限制	250
4.3	C51 的库函数	251
4.3.1	字符函数 CTYPE.H	252
4.3.2	一般 I/O 函数 STDIO.H	258
4.3.3	字符串函数 STRING.H	266
4.3.4	标准函数 STDLIB.H	273
4.3.5	数学函数 MATH.H	277
4.3.6	绝对地址访问 ABSACC.H	284
4.3.7	内部函数 INTRINS.H	285
4.3.8	变量参数表 STDARG.H	287
4.3.9	全程跳转 SETJMP.H	288
4.3.10	访问 SFR 和 SFR_bit 地址的 REGxxx.H	289
4.4	C51 编译器的特殊支持	290
4.4.1	具有双数据指针的 AMD/DALLAS 单片机	290
4.4.2	具有高速算术处理器的 SIEMENS 单片机	290
4.4.3	具有 2KB 片内 ROM 的 PHILIPS 单片机	296
第 5 章	宏汇编器 A51	301
5.1	符号与表达式	303

5.2	汇编伪指令	305
5.2.1	符号定义指令	305
5.2.2	保留和初始化存储器空间的指令	308
5.2.3	控制程序连接的指令	309
5.2.4	控制汇编状态的指令	310
5.2.5	段选择指令	311
5.3	宏处理	312
5.3.1	宏定义	312
5.3.2	宏调用	315
5.3.3	应用实例	316
5.4	汇编控制命令	319
5.4.1	首要控制命令	320
5.4.2	一般控制命令	323
5.4.3	条件汇编命令	324
第 6 章	目标文件的连接与转换	327
6.1	连接定位器 BL51	327
6.2	连接定位控制命令详解	331
6.2.1	列表控制命令	331
6.2.2	连接控制命令	332
6.2.3	定位控制命令	333
6.2.4	高级语言控制命令	336
6.2.5	分组控制命令	355
6.2.6	BL51 的分组配置	358
6.2.7	RTX51 控制命令	389
6.3	符号转换程序	389
6.3.1	Intel HEX 文件转换程序 OH51	389
6.3.2	分组目标文件转换程序 OC51	390
6.4	库管理器 LIB51	391
第 7 章	RTX51 实时多任务操作系统	393
7.1	RTX51 一般介绍	393
7.2	RTX51 技术参数	397
7.3	使用 RTX51 TINY 的要求和限定	399
7.4	RTX51 TINY 的任务管理	400
7.5	RTX51 TINY 的配置	401
7.6	RTX51 TINY 的系统函数	403
7.7	RTX51 TINY 的系统调试	408
第 8 章	Windows 环境下 C51 应用编程技巧与实例	421
8.1	Windows 集成开发环境的基本应用	421
8.1.1	工作环境与控制命令设置	421
8.1.2	编写 C51 应用程序的基本原则	430

8.2	C51 应用中的一些常见问题与解决方法	432
8.2.1	C51 程序设计中容易出错的地方	432
8.2.2	有关 C51 的若干实际应用技巧	434
8.3	8051 单片机串行接口扩展应用编程	439
8.3.1	用 8051 串行口扩展的矩阵键盘	439
8.3.2	利用 8051 串行口实现多机通信	441
8.3.3	串行接口 5 位 LED 驱动器 MC14489 的应用	443
8.4	软件模拟 I ² C 总线的 C51 读写程序	447
8.4.1	I ² C 总线简介	447
8.4.2	I ² C 总线通用读写程序	450
8.5	8051 单片机并行接口扩展应用编程	457
8.5.1	打印输出接口及其驱动程序	457
8.5.2	D/A 及 A/D 转换接口及其驱动程序	460
8.5.3	用可编程芯片 8155 实现 I/O 接口扩展	464
8.6	80C552 单片机中 A/D 转换器的应用	479
8.6.1	80C552 单片机简介	479
8.6.2	80C552 内部 ADC 的应用	481
8.6.3	使用 80C552 内部 ADC 时印刷电路板的设计要点	484
8.6.4	使用 80C552 内部 ADC 的 C51 驱动程序	486
8.7	87C752 单片机在气流量测量中的应用	492
8.7.1	87C752 单片机简介	492
8.7.2	气流量测量仪表的硬件设计	497
8.7.3	气流量测量仪表的软件设计	499
附录 A	Keil C51 与 ANSI C 的差别	510
附录 B	Keil C51 不同版本的差别	511
附录 C	代码优化	514
附录 D	关于配套光盘及硬件评估板	517
参考文献		518

第 1 章 8051 单片机与 C51 语言的基础

1.1 8051 单片机的基本结构与存储器组织

8051 单片机最早由 Intel 公司推出。它在一块超大规模集成电路芯片上同时集成了 CPU、ROM、RAM 以及 TIMER/COUNTER 等部件,具有 64KB 的寻址能力,使用者只需外接少量的接口电路就可组成自己的专用微处理器系统。目前市场上 8051 单片机的硬件支持芯片及软件应用程序也十分丰富;除了 Intel 公司之外, Philips、Siemens、ADM、Fujitsu、OKI、ATMEL 等公司都推出了以 8051 为核心的单片机。新一代的 8051 单片机集成度更高,在片内集成了更多的功能部件,如 A/D、PWM、PCA、WDT 以及高速 I/O 接口等,在工业测量控制领域内获得极为广泛的应用。因此有人指出 8051 单片机已成为事实上的工业标准。目前已有多个厂家生产不同型号的 8051 单片机。它们各有特点,但其基本内核相同,指令系统也完全兼容。图 1.1 所示为 8051 基本内核的结构框图。

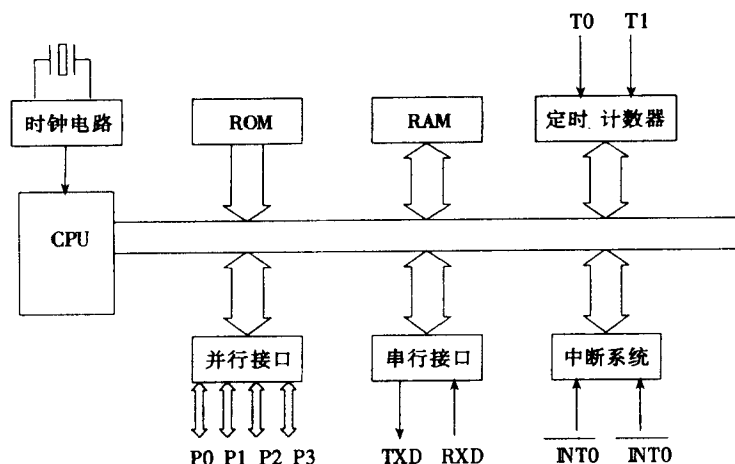


图 1.1 8051 基本内核的结构框图

每个 8051 基本内核都包括用于执行各种指令和运算处理的中央处理器 CPU,用以存放可以读写数据的内部程序存储器 RAM;用以存放程序指令或某些常数表格的内部程序存储器 ROM,4 个 8 位的并行 I/O 接口 P0、P1、P2 和 P3(每个口都可以用来输入或者输出),用来作外部事件计数器或内部定时的 2 个定时器/计数器,具有 5 个中断源的中断系统(2 个外部中断、2 个定时器中断、1 个串行口中断,采用 2 个优先级的嵌套中断结构,可实现 2 级中断服务程序嵌套,每一个中断源都可用程序规定为高优先级中断或低优先级中断),1 个串行接口电路(用于异步接收发送器),内部时钟电路(但晶体和微调电容需要外接,振荡频率最高达 40MHz)。以上各部分通过内部总线相连接。在很多情况下,单片机还要和外部设备或外部存储器相连接。连接方式采用三总线(地址、数据、控制)方式,但在 8051 单片机中,没有单独

的地址总线 and 数据总线,而是与通用并行 I/O 接口中的 P0 接口及 P2 接口公用。P0 口分时作为低 8 位地址线和 8 位数据线;P2 口则作为高 8 位地址线使用,所以是 16 条地址线和 8 条数据线。但是,一定要建立一个明确的概念,即单片机内的地址线 and 数据线都不是独立的总线,而是与并行 I/O 接口公用的,这是 8051 单片机结构上的一个特点。

图 1.2 所示为 8051 单片机的存储器结构组织图。在物理上,它有 4 个存储器空间:片内程序存储器 ROM、片外程序存储器 ROM、片内数据存储器 RAM 和片外数据存储器 RAM。从使用者的角度看,它有 3 个存储器地址空间:称为 CODE 空间的片内外统一的 64KB 的 ROM;128 字节(对 51 子系列)或 256 字节(对 52 系列)的片内 RAM,其中低 128 字节称为 DATA 空间,高 128 字节称为 IDATA 空间;称为 XDATA 空间的 64KB 的外部 RAM,在 XDATA 空间,还可以利用 P2 口实现分页寻址,称为 PDATA 空间。在访问这几个不同的存储器空间时,应采用不同形式的指令。此外,8051 单片机专门提供 1 个引脚“EA”来区分片内 ROM 和片外 ROM。当 EA 引脚接高电平时,单片机从片内 ROM 中读取指令;而当指令地址超过片内 ROM 空间范围(4KB~8KB)后,就自动地转向片外 ROM 读取指令;当 EA 引脚接低电平时,所有的取指令操作均对片外 ROM 进行。程序存储器的某些地址单元保留给系统使用:0000H~0002H 单元是所有执行程序入口地址复位后,CPU 总是从 0000H 地址开始执行程序;0003H~0002AH 单元均匀地分为 5 段,用于 5 个中断服务程序的入口,产生某个中断时,将自动进入其对应入口地址开始执行中断服务程序。外部数据存储器 RAM 也有 64KB 的寻址区,在地址上与 ROM 重叠。8051 单片机通过不同的信号来选通外部 ROM 或外部 RAM。当从外部 ROM 中取指令时,采用引脚“PSEN”作为选通信号;而从外部 RAM 中读写数据时,则采用引脚“RD”和“WR”来作为选通信号;因此,不会因地址重叠而发生混乱。在某些特殊应用场合,需要执行存放在数据存储器 RAM 内的程序。这时可将“PSEN”和“RD”做逻辑“与”操作,将外部 ROM 和外部 RAM 空间合并,形成“Von Neumann”空间,“与”得到的结果产生一个低电平有效的读选通信号,用于选通合并的“Von Neumann”空间。

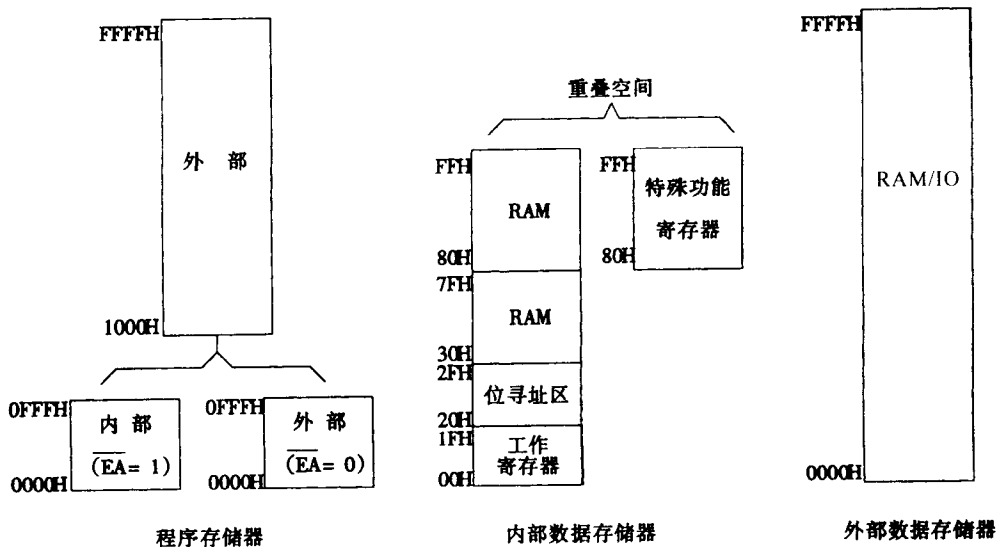


图 1.2 8051 单片机的存储器组织图

8051 单片机的片内 RAM 虽然字节数不多,但却起着十分重要的作用。图 1.3 所示为 8051 单片机片内 RAM 中的 DATA 区地址范围。其中,低 128 个字节(DATA 区)中 00H~1FH 范围为工作寄存器区,它又分为 4 组,每组都有 8 个工作寄存器 R0~R7。可以通过编程来确定当前的工作寄存器组。组与组之间不会相互混淆,从而给中断或子程序调用带来了极大的方便。在 20H~2FH 范围中,每个存储器单元的每一位都可以用位处理指令直接操作。该段地址范围称为位寻址区(BDATA 区),其中每一位称为 1 个 bit (比特)。30H~7FH 范围为通用 RAM 区,用于存放一些临时数据。由此可见,DATA 空间范围是十分有限的,51 子系列单片机仅有上述低 128 个字节。52 子系列单片机增加了高 128 个字节的片内 RAM,地址范围为 80H~FFH,称为 IDATA 区,该区只能采用间接寻址方式进行访问。与 IDATA 空间重叠的高 128 个字节(地址范围 80H~FFH)称为特殊功能寄存器区(SFR)。有些特殊功能寄存器是可以位寻址的,其可寻址位称为 sbit。各种型号单片机内部集成功能的操作都是通过特殊功能寄存器来实现的。表 1-1 所示为 8051 单片机特殊功能寄存器一览表,表中带 * 号的为可位寻址的特殊功能寄存器。8051 单片机的堆栈操作必须在片内 RAM 中进行。由于片内 RAM 地址空间有限,而且还包括上述工作寄存器区和位寻址区,因此,堆栈的安排必须十分谨慎,不要发生堆栈溢出而导致程序不能正常运行的情况。

RAM 地址	MSB								LSB
7FH									127
2FH	7F	7E	7D	7C	7B	7A	79	78	47
2EH	77	76	75	74	73	72	71	70	46
2DH	6F	6E	6D	6C	6B	6A	69	68	45
2CH	67	66	65	64	63	62	61	60	44
2BH	5F	5E	5D	5C	5B	5A	59	58	43
2AH	57	56	55	54	53	52	51	50	42
29H	4F	4E	4D	4C	4B	4A	49	48	41
28H	47	46	45	44	43	42	41	40	40
27H	3F	3E	3D	3C	3B	3A	39	38	39
26H	37	36	35	34	33	32	31	30	38
25H	2F	2E	2D	2C	2B	2A	29	28	37
24H	27	26	25	24	23	22	21	20	36
23H	1F	1E	1D	1C	1B	1A	19	18	35
22H	17	16	15	14	13	12	11	10	34
21H	0F	0E	0D	0C	0B	0A	09	08	33
20H	07	06	05	04	03	02	01	00	32
1FH	工作寄存器3区								31
18H									24
17H	工作寄存器2区								23
10H									16
0FH	工作寄存器1区								15
08H									8
07H	工作寄存器0区								7
00H									0

图 1.3 8051 片内 RAM 地址范围

表 1-1 8051 单片机特殊功能寄存器一览表

符 号	地 址	说 明
* ACC	E0H	累加器
* B	F0H	乘法寄存器
* PSW	D0H	程序状态字寄存器
SP	81H	堆栈指针寄存器
DPL	82H	数据存储器指针 低 8 位
DPH	83H	数据存储器指针 高 8 位
* IE	A8H	中断允许寄存器
* IP	B8H	中断优先寄存器

(续表)

符 号	地 址	说 明
* P0	80H	并行 I/O 接口 P0
* P1	90H	并行 I/O 接口 P1
* P2	A0H	并行 I/O 接口 P2
* P3	B0H	并行 I/O 接口 P3
PCON	87H	电源控制与波特率选择寄存器
* SCON	98H	串行口控制寄存器
SBUF	99H	串行口缓冲寄存器
* TCON	88H	定时器方式选择寄存器
TMOD	89H	定时器方式选择寄存器
TL0	8AH	定时器 0 初值寄存器 低 8 位
TH0	8BH	定时器 0 初值寄存器 高 8 位
TL1	8CH	定时器 1 初值寄存器 低 8 位
TH1	8DH	定时器 1 初值寄存器 高 8 位

8051 单片机的指令系统有 7 种寻址方式,可寻址 64KB 的程序存储器空间和 64KB 的数据存储器空间;它共有 111 条指令,其中包括乘、除指令和位操作指令。Keil C51 编译器实际上就是将用 C 语言编写的用户程序编译成为 8051 单片机的机器指令,再通过连接定位器将机器代码安排到合适的存储器空间位置上去执行。

1.2 C51 语言的特点

1.2.1 概述

C 语言是一种通用的计算机程序设计语言,在国际上十分流行。它既可用于编写计算机的系统程序,也可用来编写一般的应用程序。以前计算机的系统软件主要用汇编语言编写,单片机应用系统更是如此。由于汇编语言程序的可读性和可移植性都较差,采用汇编语言编写单片机应用程序不但周期长,而且调试和排错也比较困难。为了提高编制单片机应用程序的效率,改善程序的可读性和可移植性,采用高级语言无疑是一种最好的选择。C 语言既具有一般高级语言的特点,又能直接对计算机的硬件进行操作,表达和运算能力也较强,许多以前只能采用汇编语言来解决的问题现在都可以改用 C 语言来解决。德国 Keil Software 公司多年来致力于单片机 C 语言编译器的研究。该公司开发的 Keil C51 是一种专为 8051 单片机设计的高效率 C 语言编译器,符合 ANSI 标准,生成的程序代码运行速度极高,所需要的存储器空间极小,完全可以和汇编语言相媲美。

Keil 公司的 C51 编译器 V4.0 版本以下的产品曾通过美国 Franklin 公司销售多年,例如传入到中国国内并得到广泛使用的基于 DOS 环境的 Franklin C51V3.2 和 V3.96 版本等。近年来,Keil 公司的产品不断升级,已经开发出基于 Windows 环境的 C51 编译器 V5.50 以上版本。高版本 Keil C51 编译器具有更多的优点,使用也更为方便。Keil C51 以软件包的形式向用户提供,主要包括 C51 交叉编译器、A51 宏汇编器、BL51 连接定位器等一系列工具和基于 Windows 集成编译环境的 μ Vision51、软件仿真器 dScope51 等开发平台。 μ Vision51 是一种集成化的文件管理编译环境,其中集成了文件编辑处理、编译连接、项目(project)管理、窗口和工具引用以及工作环境路径设置等多种功能。在 μ Vision51 中完成一个项目文件后,只要点击

Make All 按钮就可完成所有的编译工作,生成绝对目标代码。如果还想得到 Intel 的 HEX 格式文件,只需要在 Options 菜单的 Make 选项中选择“Run OH51 Object Hex Converter”即可。dScope51 是 Windows 环境下的多窗口软件仿真调试器,可以单独应用,也可以通过 μ Vision51 进行调用。dScope51 具有十分强大的仿真调试功能,支持软件模拟仿真和用户目标板调试两种工作方式。在软件模拟仿真方式下,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;与用户目标板相连时,可以直接进行硬件系统调试,使用户可以节省购买昂贵硬件仿真器的费用。

V5.50 以上版本的 Keil C51 具有如下一些主要特点:

- 支持 9 种基本数据类型,其中包括 32 位长的浮点类型;
- 变量可存放在不同类型的存储空间中,如 bit, data, bdata, idata, xdata 和 pdata;
- 支持直接采用 C 语言编写的 8051 单片机的中断服务函数;
- 充分利用 8051 工作寄存器组;
- 可以保留源程序中的所有符号、类型信息,方便调试;
- 支持位寻址操作;
- 支持 Dallas 公司 80C320/520/530 等单片机的双数据指针;
- 支持 Philips 公司 8XC750/751/752 等单片机的指令集;
- 支持 Siemens (Infineon)公司 80C517 单片机的算术单元和多重数据指针。

与 4.0 以前版本的 C51 相比,新版本的 Keil C51 无论对长整型数的运算还是浮点型运算都有了长足的进步,同时还降低了存储器开销和数据量。在提高效率的同时,Keil C51 还加强了代码的优化,提供由低到高的 7 级普通优化和针对 8051 的特殊优化措施,尽可能地利用寄存器来存储变量和函数参数,减少重复无用的代码、表达式,支持数据覆盖等方案都有效地节省了系统资源,提高了程序的可靠性。

C51 编译器可以实现对 8051 系列所有资源的操作,利用关键字“sfr”和“sbit”很容易实现特殊功能寄存器 SFR 的操作。变量可旋转到任一个地址空间。用关键字“_at_”可以把变量放入指定地址的存储器空间;利用存储模式关键字“LARGE”、“COMPACT”和“SMALL”可以决定变量的存储类型。分组连接定位器 BL51 支持多个分组代码区,从而允许用户程序突破 64KB 的限制。C51 允许用户使用 C 语言编写中断服务程序,以更加高效地实现堆栈快速操作和工作寄存器区切换。利用关键字“reentrant”可定义函数的再入属性,多任务、中断或非中断的函数代码要求必须具备可再入功能。C51 提供了灵活高效的指针操作,通用指针用 3 个字节来存储存储器类型及目标地址,可以在 8051 的任意存储区内存取任何变量。基于存储器的指针在声明时指定了存储器类型,使之指向某一特定的存储区域。基于存储器指针的存储只需 1~2 个字节,因此,指针存取非常迅速。

C51 具有丰富的函数库,包含 100 多种功能函数,其中大多数是可再入函数,为用户提供极大的方便。C51 程序可以实现与汇编语言程序的接口,两者相互之间的调用十分方便。Keil 公司还提供一种多任务实时操作系统 RTX51。RTX51 可以支持 C 语言和汇编语言程序、代码区分组的应用程序、标准任务(用一个共享寄存器区)、快速任务(用一个专用寄存器区)、CAN 总线、邮箱(为内部任务通信)、存储池(memory pools)等操作,对于单片机系统,可以采用简化后的内核 RTX51 TINY。

尽管 C51 具有很多的优点,但和其他任何一种程序设计语言一样,也有其自身的缺点,如不能自动检查数组的边界,各种运算符的优先级别太多,某些运算符具有多种用途等。但总

地来说, C51 的优点远远超过了它的缺点。经验表明, 程序设计人员一旦学会使用 C51 之后, 就会对它爱不释手。

1.2.2 C51 的基本程序结构

C 语言源程序文件的扩展名为“.c”, 如 Timer.c, EX1_2.c 等。C 语言源程序由若干个函数单元组成, 每个函数都是完成某个特殊任务的子程序段。一个 C 语言程序必须有而且只能有一个名为 main() 的函数。它是一个特殊的函数, 也称为该程序的主函数, 程序的执行都是从 main() 函数开始的。我们先来看一个简单的程序例子。

【例 1.1】 求两个输入数据中的较大者

```
# include <stdio.h>                                /* 预处理命令 */
# include <reg51.h>

char max (char x, char y);                          /* 功能函数 max 及其形式参数说明 */

main() {                                             /* 主函数 */
    char a, A, c;                                   /* 主函数的内部变量类型说明 */
    SCON = 0x52;                                    /* 8051 单片机串行口初始化 */
    TMOD = 0x20;
    TCON = 0x69;
    TH1 = 0x0F3;
    scanf ("%c %c", &a, &A);                      /* 输入变量 a 和 b 的值 */
    c = max (a, A);                                  /* 调用 max 函数 */
    printf ( " \n max = %c \n ", c);               /* 输出变量 c 的值 */
}                                                    /* 主程序结束 */

char max (char x , char y) {                       /* 定义 max 函数, x,y 为形式参数 */
    if ( x > y ) return (x);                        /* 将计算得到的最大值返回到调用处 */
    else return(y);
}                                                    /* max 函数结束 */
```

在本例程序的开始处使用了预处理命令 # include, 它告诉编译器在编译时将头文件 stdio.h 和 reg51.h 读入后一起编译。头文件 stdio.h 中包括了对标准输入输出函数的说明; 头文件 reg51.h 中包括了对 8051 单片机特殊功能寄存器的说明。

程序中, main 是主函数名, 要执行的内容称为主函数体, 主函数体用花括号“{}”围起来。函数体中包含若干条将被执行的程序语句, 每条语句都必须以分号“;”为结束符。为了使程序便于阅读和理解, 可以给程序加上一些注释。C 语言的注释部分由符号“/*”开始, 由符号“*/”结束, 它们之间的内容即为注释。注释内容可在一行写完, 也可以分成几行来写。注释部分不参加编译, 编译时, 注释的内容不产生可执行代码。注释在程序中的作用是很重要的, 一个好的程序设计者应该在程序中使用足够的注释来说明整个程序的功能、有关算法和注意事项等。需要注意的是, C 语言中的注释不能嵌套, 即在“/*”和“*/”之间不允许再次出现“/*”和“*/”。

一般情况下, 一个 C 语言程序除了必须有一个主函数之外, 还可能若干个其他的功能

函数。本程序中除了 `main()` 函数之外,还用到了一个功能函数 `max()`。函数 `max()` 的作用是求出变量 `x` 和 `y` 中较大者,并通过 `return` 语句将它的值返回到 `main()` 函数的调用处。变量 `x` 和 `y` 在函数 `max()` 中是一种形式变量,实际值是通过 `main()` 函数中的调用语句传送过来的。此外,ANSI C 标准规定函数必须要“先说明,后调用”,因此,在调用函数 `max()` 之前必须先进行说明。函数是 C 语言程序的基本单位,函数调用类似于子程序调用,用户可以根据实际需要编制出各种不同用途的功能函数。可以说,C 语言是函数式的语言,利用 C 语言的这一特点,可以很容易实现结构化的程序设计。

C51 编译器提供了十分丰富的库函数,本例在 `main()` 函数中调用了库函数 `scanf()` 和 `printf()` 来实现变量的输入和输出。C 语言本身没有输入输出功能,输入输出需要通过函数调用来实现。需要注意的是,C51 提供的输入输出库函数是通过 8051 单片机的串行口来实现的。因此,在调用库函数 `scanf()` 和 `printf()` 之前,必须先对 8051 单片机的串行口进行初始化。但是对于单片机应用系统来说,由于具体要求的不同,应用系统的输入输出方式也多种多样,不可能一律采用串口作输入和输出。应该根据实际需要,由应用系统的研制人员自己来编写满足特定需要的输入输出函数,这一点对于单片机应用系统的开发研制人员来说是十分重要的。另外我们在程序中还可以看到,小写字母 `a` 和大写字母 `A` 分别是两种不同的变量。C 语言规定,同一个字母由于其大小写的不同可以代表两个不同的变量;例如,`SCON` 和 `scn` 在 C 语言程序中会被认为是两个完全不同的变量,这也是 C 语言的一个特点。一般的习惯是在普通情况下采用小写字母,对于一些具有特殊意义的变量或常数采用大写字母,如本例中,8051 单片机特殊功能寄存器 `SCON`、`TMOD`、`TCON` 和 `TH1` 等均采用了大写字母。

从以上说明可以看到,C 语言程序一般具有如下的结构:

预处理命令	<code># include < ></code>
函数定义	<code>char fun1 ();</code> <code>float fun2();</code>
主函数	<code>main() {</code>
函数体	<code>...</code> <code>}</code>
功能函数 1	<code>fun1() {</code>
函数体	<code>...</code> <code>}</code>
功能函数 2	<code>fun2() {</code>
函数体	<code>...</code> <code>}</code> <code>...</code>

C 语言程序的开始部分通常是预处理命令,如上面程序中的 `# include` 命令。这个预处理命令通知编译器在对用户程序进行编译时,将所需要的头文件(`*.h`)读入后,再一起进行编译。一般在头文件中包含程序编译时的一些必要的信息。通常 C 语言编译器会提供若干个不同用途的头文件。头文件的读入是在对程序进行编译时才完成的。此外还有其他一些预处理命令,将在第 2 章中详细介绍。

一个 C 语言程序至少应包含一个主函数 `main()`,也可以包含一个 `main()` 函数和若干个其他的函数。函数之间可以相互调用,但 `main()` 函数只能调用其他的函数,而不能