

21 世纪高等学校计算机基础实用规划教材

微机原理、汇编 与接口技术 (第二版)

朱定华 编著

清华大学出版社

21 世纪高等学校计算机**基础**实用规划教材

微机原理、汇编与接口技术

(第二版)

朱定华 编著

清华大学出版社
北京

内 容 简 介

本书系统地介绍了 80x86 微型计算机的原理、汇编语言程序设计与接口技术,主要内容包括计算机基础知识、汇编语言与汇编程序、程序设计技术、总线、存储器及其接口、输入与输出技术、中断技术和常用可编程接口芯片等。

本书内容精练、实例丰富,其中大量的接口电路和程序是作者多年来在科研和教学中反复提炼得来的,因而本书应用性很强,可作为大专院校和高职高专成人高等教育“汇编语言程序设计”、“微机原理及应用”、“接口技术”等课程的教学用书,也可以供从事电子技术、计算机应用与开发的科研人员和工程技术人员学习参考,还适于初学者自学使用。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

微机原理、汇编与接口技术/朱定华编著. —2 版. —北京:清华大学出版社,2010.1

(21 世纪高等学校计算机基础实用规划教材)

ISBN 978-7-302-21143-3

I. 微… II. 朱… III. ①微型计算机—理论—高等学校—教材 ②汇编语言—程序设计—高等学校—教材 ③微型计算机—接口—高等学校—教材 IV. TP36 TP313

中国版本图书馆 CIP 数据核字(2009)第 175765 号

责任编辑:魏江江 薛 阳

责任校对:白 蕾

责任印制:何 芊

出版发行:清华大学出版社

地 址:北京清华大学学研大厦 A 座

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

邮 编:100084

社 总 机:010-62770175

邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

印 刷 者:北京国马印刷厂

装 订 者:北京市密云县京文制本装订厂

经 销:全国新华书店

开 本:185×260 印 张:20.75 字 数:499 千字

版 次:2005 年 6 月第 1 版 2010 年 1 月第 2 版

印 次:2010 年 1 月第 1 次印刷

印 数:19001~23000

定 价:29.00 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题,请与清华大学出版社出版部联系调换。联系电话:(010)62770177 转 3103 产品编号:033703-01

出版说明

随着我国改革开放的进一步深化,高等教育也得到了快速发展,各地高校紧密结合地方经济建设发展需要,科学运用市场调节机制,加大了使用信息科学等现代科学技术提升、改造传统学科专业的投入力度,通过教育改革合理调整和配置了教育资源,优化了传统学科专业,积极为地方经济建设输送人才,为我国经济社会的快速、健康和可持续发展以及高等教育自身的改革发展做出了巨大贡献。但是,高等教育质量还需要进一步提高以适应经济社会发展的需要,不少高校的专业设置和结构不尽合理,教师队伍整体素质亟待提高,人才培养模式、教学内容和方法需要进一步转变,学生的实践能力和创新精神亟待加强。

教育部一直十分重视高等教育质量工作。2007年1月,教育部下发了《关于实施高等学校本科教学质量与教学改革工程的意见》,计划实施“高等学校本科教学质量与教学改革工程(简称‘质量工程’)”,通过专业结构调整、课程教材建设、实践教学改革、教学团队建设等多项内容,进一步深化高等学校教学改革,提高人才培养的能力和水平,更好地满足经济社会发展对高素质人才的需要。在贯彻和落实教育部“质量工程”的过程中,各地高校发挥师资力量强、办学经验丰富、教学资源充裕等优势,对其特色专业及特色课程(群)加以规划、整理和总结,更新教学内容、改革课程体系,建设了一大批内容新、体系新、方法新、手段新的特色课程。在此基础上,经教育部相关教学指导委员会专家的指导和建议,清华大学出版社在多个领域精选各高校的特色课程,分别规划出版系列教材,以配合“质量工程”的实施,满足各高校教学质量和教学改革的需要。

本系列教材立足于计算机公共课程领域,以公共基础课为主、专业基础课为辅,横向满足高校多层次教学的需要。在规划过程中体现了如下一些基本原则和特点。

(1) 面向多层次、多学科专业,强调计算机在各专业中的应用。教材内容坚持基本理论适度,反映各层次对基本理论和原理的需求,同时加强实践和应用环节。

(2) 反映教学需要,促进教学发展。教材要适应多样化的教学需要,正确把握教学内容和课程体系的改革方向,在选择教材内容和编写体系时注意体现素质教育、创新能力与实践能力的培养,为学生的知识、能力、素质协调发展创造条件。

(3) 实施精品战略,突出重点,保证质量。规划教材把重点放在公共基础课和专业基础课的教材建设上;特别注意选择并安排一部分原来基础比较好的优秀教材或讲义修订再版,逐步形成精品教材;提倡并鼓励编写体现教学质量和教学改革成果的教材。

(4) 主张一纲多本,合理配套。基础课和专业基础课教材配套,同一门课程有针对不同层次、面向不同专业的多本具有各自内容特点的教材。处理好教材统一性与多样化,基本教材与辅助教材、教学参考书,文字教材与软件教材的关系,实现教材系列资源配套。

前 言

本书第一版自 2005 年出版以来,受到诸多兄弟院校师生及广大读者的关注,我们深表感谢。

通过多年来的教学实践,尤其是近 3 年来课程改革的经验,我们对教材内容和课程体系进行了深入的研究,并作了修改和更新。随着电子技术和微型计算机技术的迅猛发展,从 8086 开始,80286、80386、80486、Pentium 等系列微处理器不断推出,本书在第一版的基础上对原章节加宽加深,既保持了多年形成的比较成熟的课程体系,又适当地介绍了微型计算机中的新器件、新技术和新方法。

本书通过对 8086 到 Pentium 微处理器和以 80x86 为 CPU 的微型计算机的硬件和软件的分析,阐明微型计算机的组成原理、汇编语言程序设计以及存储器、输入输出接口芯片与微处理器的接口方法,为学习者在微处理器和微型计算机的应用上打下坚实的基础。

本书包括汇编语言程序设计和接口技术两部分内容。汇编语言程序设计是微机应用系统的系统软件和应用软件的设计基础,接口技术是微机应用系统硬件组成的设计基础。本书内容较全面,实例丰富。书中的程序和接口电路的设计包含了作者多年来在科研和教学中积累的经验 and 技巧。学习微型计算机的汇编语言程序设计和接口技术必须理论联系实际。本书在介绍基本概念的同时,列举了大量典型而有意义的例题和习题。这些例题和习题,无论是汇编程序还是接口电路都在 80x86 系列微型计算机系统上调试通过。

80x86 为用户提供了实地址方式、虚地址保护方式和虚拟 8086 方式 3 种工作方式,但从编程角度看,仅提供了实地址方式和虚地址保护方式两种工作方式。就编程而言,这两种工作方式并无实质上的区别,而且使用实地址方式已可解决应用程序所面向的大量问题,所以本书有关汇编语言程序设计的讨论只限于 DOS 环境下(MASM 5.0)的实地址方式。

本书内容精练,实用性强。每章后均附有习题与思考题。本书注意理论和实践相结合,力求做到既有一定的理论基础,又能运用理论解决实际问题;既掌握一定的先进技术,又着眼于当前的应用服务。

本教材的参考学时数为 80 学时(不含实验)。学时数较少的学校或专业可以不讲授第 2 章和第 3 章中的以下内容:地址传送指令、查表转换指令、BIOS、串处理程序设计和宏功能程序设计等,本书后面没有使用这些内容,还可以只选讲实地址方式的有关内容。

本书主要由朱定华编写,参加本书编写工作的人员还有周斌、蔡红娟、蔡苗、翟晟、黄松、吕建才、陈艳、林卫、程萍、张德芳、李志文、林威等。

计算机的发展日新月异,教学改革任重道远,限于我们的水平和能力,不妥之处在所难免,恳请读者批评指正,以便我们今后不断改进。

编 者

2009 年 10 月于武昌

(5) 依靠专家,择优选用。在制定教材规划时依靠各课程专家在调查研究本课程教材建设现状的基础上提出规划选题。在落实主编人选时,要引入竞争机制,通过申报、评审确定主题。书稿完成后要认真实行审稿程序,确保出书质量。

繁荣教材出版事业,提高教材质量的关键是教师。建立一支高水平教材编写梯队才能保证教材的编写质量和建设力度,希望有志于教材建设的教师能够加入到我们的编写队伍中来。

21 世纪高等学校计算机基础实用规划教材

联系人:魏江江 weijj@tup.tsinghua.edu.cn

目 录

第 1 章 微型计算机基础	1
1.1 计算机中的数和编码	1
1.1.1 计算机中的数制	1
1.1.2 符号数的表示法	2
1.1.3 二进制数的加减运算	5
1.1.4 二进制数的逻辑运算与逻辑电路	7
1.1.5 二进制编码	9
1.1.6 BCD 数的加减运算	11
1.2 微型计算机的结构和工作原理	12
1.2.1 微型计算机常用的术语	12
1.2.2 微型计算机的基本结构	12
1.2.3 计算机的工作原理	16
1.3 80x86 微处理器	16
1.3.1 8086/8088	16
1.3.2 80286	18
1.3.3 80386	18
1.3.4 80486	18
1.3.5 Pentium	19
1.3.6 Itanium	22
1.4 80x86 的寄存器	23
1.4.1 8086/8088 的寄存器	23
1.4.2 80286 的寄存器	26
1.4.3 80386 的寄存器	26
1.4.4 80486 的寄存器	27
1.4.5 Pentium 的寄存器	27
1.5 80x86 的工作方式与存储器物理地址的生成	34
1.5.1 80x86 的工作方式	34
1.5.2 80x86 存储器的分段和物理地址的生成	35
习题与思考题	38

第2章 汇编语言与汇编程序	41
2.1 符号指令中的表达式	41
2.1.1 常量	42
2.1.2 标号	42
2.1.3 变量	43
2.2 符号指令的寻址方式	45
2.2.1 寄存器寻址	45
2.2.2 立即寻址	46
2.2.3 直接寻址	46
2.2.4 间接寻址	47
2.2.5 基址寻址	47
2.2.6 变址寻址	48
2.2.7 基址变址寻址	48
2.2.8 比例变址寻址(80386 及其后继微处理器可用)	49
2.2.9 基址比例变址寻址(80386 及其后继微处理器可用)	49
2.2.10 存储器寻址及存储器寻址中段基址或段选择字的确定	49
2.3 常用指令	51
2.3.1 数据传送类指令	51
2.3.2 加减运算指令	57
2.3.3 逻辑运算指令	60
2.3.4 移位指令	62
2.3.5 位搜索(扫描 bit scan)指令(80386 及其后继微处理器可用)	65
2.3.6 位测试(bit tests)指令(80386 及其后继微处理器可用)	65
2.3.7 指令应用举例	66
2.4 常用伪指令	70
2.4.1 过程的定义	70
2.4.2 微处理器选择伪指令	70
2.4.3 段的定义	71
2.4.4 汇编地址计数器	72
2.4.5 段寄存器的假定	72
2.4.6 源程序的结束	73
2.4.7 宏汇编源程序的格式	73
2.5 常用系统功能调用和 BIOS	75
2.5.1 系统功能调用	75
2.5.2 常用系统功能调用应用举例	77
2.5.3 BIOS	81
习题与思考题	83

第 3 章 程序设计的基本技术	88
3.1 顺序程序设计	88
3.1.1 乘除法指令	88
3.1.2 BCD 数调整指令	91
3.1.3 顺序程序设计举例	96
3.2 分支程序设计	99
3.2.1 条件转移指令	99
3.2.2 无条件转移指令	101
3.2.3 分支程序设计举例	102
3.3 循环程序设计	106
3.3.1 循环程序的基本结构	107
3.3.2 重复控制指令	108
3.3.3 单重循环程序设计举例	109
3.3.4 多重循环程序设计举例	122
3.4 串处理程序设计	127
3.4.1 方向标志置位和清除指令	127
3.4.2 串操作指令	128
3.4.3 重复前缀	129
3.4.4 串操作程序设计举例	129
3.5 子程序设计	135
3.5.1 子程序的概念	135
3.5.2 子程序的调用指令与返回指令	137
3.5.3 子程序及其调用程序设计举例	138
3.6 宏功能程序设计	148
3.6.1 宏指令	149
3.6.2 条件汇编与宏库的使用	152
3.6.3 宏功能程序设计举例	153
习题与思考题	155
第 4 章 总线	161
4.1 总线概述	161
4.1.1 总线分类	161
4.1.2 总线操作	162
4.2 8086/8088 的 CPU 总线	162
4.2.1 8086/8088 的引线及功能	162
4.2.2 8088 的 CPU 系统	166
4.2.3 8088 的时序	171

4.3	Pentium 的 CPU 总线	177
4.4	局部总线	181
4.4.1	ISA 局部总线	181
4.4.2	PCI 局部总线	183
4.5	通用外部总线	188
4.5.1	IDE 总线	188
4.5.2	SCSI 总线	188
4.5.3	通用串行总线 USB	189
4.5.4	视频接口 AGP	190
4.6	Pentium 微型计算机系统	190
4.6.1	北桥-南桥式控制芯片组与微型计算机系统	191
4.6.2	MCH-ICH 集中式控制芯片组与微型计算机系统	191
	习题与思考题	193
第 5 章	存储器及其接口	194
5.1	存储器概述	194
5.1.1	存储器的类型	194
5.1.2	存储器的性能指标	194
5.1.3	存储器的分级结构	195
5.2	常用的存储器芯片	196
5.2.1	半导体存储器芯片的结构	196
5.2.2	只读存储器 ROM	196
5.2.3	随机读写存储器 RAM	198
5.3	存储器与 CPU 的接口	202
5.3.1	存储器芯片与地址总线的连接	203
5.3.2	存储器芯片与数据总线的连接	204
5.3.3	存储器芯片与控制总线的连接	204
5.3.4	连接举例	204
	习题与思考题	208
第 6 章	输入输出和接口技术	209
6.1	接口的基本概念	209
6.1.1	接口的功能	209
6.1.2	接口控制原理	210
6.1.3	接口控制信号	212
6.2	I/O 指令和 I/O 地址译码	213
6.2.1	标准的 I/O 寻址方式	213
6.2.2	存储器映像 I/O 寻址方式	213

6.2.3	输入输出指令	214
6.2.4	I/O 接口的端口地址译码	214
6.3	数字通道接口	216
6.3.1	数据输出寄存器	217
6.3.2	数据输入三态缓冲器	217
6.3.3	三态缓冲寄存器	218
6.3.4	寄存器和缓冲器接口中的应用	218
6.3.5	打印机适配器	225
6.4	模拟通道接口	229
6.4.1	数模转换器 DAC 及其与微型计算机的接口	230
6.4.2	模数转换器 ADC 及其与微型计算机的接口	236
	习题与思考题	241
第 7 章	中断技术	244
7.1	中断和中断系统	244
7.1.1	中断的概念	244
7.1.2	中断请求与控制 and 中断源的识别	244
7.1.3	中断系统的功能	245
7.1.4	CPU 对外部可屏蔽中断的响应及中断过程	246
7.2	中断控制器 8259A	246
7.2.1	8259A 的组成和接口信号	246
7.2.2	8259A 处理中断的过程	248
7.2.3	8259A 的级联连接	248
7.2.4	8259A 的命令字	248
7.3	80x86 微处理器的中断指令和 80x86 微型计算机的中断系统	251
7.3.1	中断指令	251
7.3.2	外部中断	252
7.3.3	内部中断	253
7.3.4	中断类型码及中断种类	253
7.3.5	中断向量表和中断描述符表	254
7.3.6	中断响应和处理过程	255
7.4	实地址方式下的可屏蔽中断服务程序的设计	256
7.4.1	中断服务程序入口地址的装入	256
7.4.2	中断屏蔽与中断结束的处理	257
7.4.3	中断服务程序设计举例	257
	习题与思考题	263
第 8 章	常用可编程接口芯片	264
8.1	可编程并行接口 8255	264

8.1.1	8255 的组成与接口信号	264
8.1.2	8255 的工作方式与控制字	266
8.1.3	3 种工作方式的功能	269
8.2	可编程计数器/定时器 8253	275
8.2.1	8253 的组成与接口信号	275
8.2.2	计数器的工作方式及其与输入输出的关系	277
8.2.3	8253 的控制字和初始化编程	278
8.2.4	8253 的应用	280
8.3	串行通信与异步通信控制器 8250 的应用	285
8.3.1	80x86 微型计算机的串行口	285
8.3.2	异步通信控制器 8250	287
8.3.3	8250 与微型计算机及 RS-232 接口信号的连接	294
8.3.4	异步串行通信程序设计	295
8.3.5	80x86 微型计算机之间的通信	297
8.3.6	80x86 微型计算机与 MCS-51 单片机之间的通信	299
	习题与思考题	302
附录 A	80x86 指令系统表	304
附录 B	80x86 指令按字母顺序查找表	309
附录 C	80x86 算术逻辑运算指令对状态标志位的影响	314

1.1 计算机中的数和编码

1.1.1 计算机中的数制

计算机最早是作为一种计算工具出现的,所以它的最基本的功能是对数进行加工和处理。数在机器中是以器件的物理状态来表示的。一个具有两种不同的稳定状态且能相互转换的器件就可以用来表示 1 位二进制数。二进制数有运算简单、便于物理实现、节省设备等优点,所以目前在计算机中数几乎全是采用二进制表示的。但是二进制数书写起来太长,且不利于阅读和记忆,而 4 位二进制数有 16 个不同的状态 0000~1111,即是 1 位十六进制数,所以微型计算机中的二进制数都采用十六进制数来缩写。十六进制数用 0~9 和 A~F 共 16 个数码表示 4 位二进制数 0000~1111,这 16 个二进制数 0000~1111 的大小就是十进制数 0~15。1 个 8 位的二进制数用 2 位十六进制数表示,1 个 16 位的二进制数用 4 位十六进制数表示等。这样书写方便,又便于阅读和记忆,且转换方便,因此常用十六进制数来缩写二进制数。然而人们最熟悉、最常用的是十进制数。为此,要熟练地掌握十进制数、二进制数和十六进制数间的相互转换。它们之间的关系如表 1-1 所列。

表 1-1 十进制数、二进制数及十六进制数对照表

十进制数	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
二进制数	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
十六进制数	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

为了区别十进制数、二进制数及十六进制数 3 种数制,可在数的右下角注明数制,或者在数的后面加一字母。如 B(binary)表示二进制数制;D(decimal)或不带字母表示十进制数制;H(hexadecimal)表示十六进制数制。

1. 二进制数和十六进制数整数间的相互转换

根据表 1-1 所示的对应关系即可实现它们之间的转换。

二进制整数转换为十六进制数,其方法是从右(最低位)向左将二进制数分组;每 4 位为 1 组,最后一组若不足 4 位则在其左边添加 0,以凑成 4 位 1 组,每组用 1 位十六进制数表示。如

11111110001111B → 1 1111 1100 0111B → 0001 1111 1100 0111B = 1FC7H

十六进制数转换为二进制数,只需用 4 位二进制数代替 1 位十六进制数即可。如

$$3AB9H = 0011\ 1010\ 1011\ 1001B$$

2. 十六进制数和十进制数整数间的相互转换

十六进制数转换为十进制数十分简单,只需将十六进制数按权展开相加即可。如

$$\begin{aligned} 1F3DH &= 16^3 \times 1 + 16^2 \times 15 + 16^1 \times 3 + 16^0 \times 13 \\ &= 4096 \times 1 + 256 \times 15 + 16 \times 3 + 1 \times 13 \\ &= 4096 + 3840 + 48 + 13 = 7997 \end{aligned}$$

十进制整数转换为十六进制数可用除 16 取余法,即用 16 不断地去除待转换的十进制数,直至商等于 0 为止。将所得的各次余数,依倒序排列,即可得到所转换的十六进制数。如将 38947 转换为十六进制数,其方法及算式如下:

$$\begin{array}{r} 16 \overline{) 38947} \quad 3 \\ 16 \overline{) 2434} \quad 2 \\ 16 \overline{) 152} \quad 8 \\ 16 \overline{) 9} \quad 9 \\ 0 \end{array}$$

即 $38947 = 9823H$ 。

3. 二进制数和十进制数整数间的相互转换

把一个十进制数转换为二进制数,可以先把该数转换为十六进制数,然后再转换为二进制数,这样可以减少计算次数;反之,要把一个二进制数转换为十进制数,也可以采用同样的办法。若使用 2^n (2^n 的二进制数等于 1 后跟 n 个 0) 和十六进制数、十进制数的对应关系(如表 1-2 所示)以及个别十进制整数和十六进制数的对应关系(如 $50 = 32H$, $80 = 64H$, $100 = 64H$ 等),则转换起来更为方便。如

$$38947 = 32768 + 4096 + 2048 + 32 + 3 = 8000H + 1000H + 800H + 20H + 3H = 9823H$$

$$1F3DH = 2000H - (80H + 40H + 3H) = 8192 - (128 + 64 + 3) = 7997$$

表 1-2 部分二进制数与十六进制数和十进制数的对应关系

2^n	2^5	2^6	2^7	2^8	2^9	2^{10}	2^{11}	2^{12}	2^{13}	2^{14}	2^{15}	2^{16}	2^{20}	2^{30}	2^{40}	2^{50}	2^{60}
十六进制数	20	40	80	100	200	400	800	1000	2000	4000	8000	10000					
十进制数	32	64	128	256	512	1024	2048	4096	8192	16384	32768	65536					
常用的缩写						1K	2K	4K	8K	16K	32K	64K	1M	1G	1T	1P	1E

1.1.2 符号数的表示法

1. 机器数与真值

二进制数与十进制数一样有正负之分。在计算机中,常用数的符号和数值部分一起编码的方法表示符号数。常用的有原码、反码和补码表示法。这几种表示法都将数的符号数码化。通常正号用“0”表示,负号用“1”表示。为了区分一般书写时表示的数和机器中编码表示的数,我们称前者为真值,后者为机器数,即数值连同符号数码“0”或“1”一起作为一个数就称为机器数,而它的数值连同符号“+”或“-”称为机器数的真值。把机器数的符号位也当作数值的数,就是无符号数。

为了表示方便,常把 8 位二进制数称为字节,把 16 位二进制数称为字,把 32 位二进制

数称为双字。对于机器数应将其用字节、字或双字表示,所以只有 8 位、16 位或 32 位机器数的最高位才是符号位。

2. 原码

按上所述,数值用其绝对值,正数的符号位用 0 表示,负数的符号位用 1 表示,这样表示的数就称为原码。如

$$X_1 = 105 = +0110\ 1001B \quad [X_1]_{\text{原}} = 0110\ 1001B$$

$$X_2 = -105 = -0110\ 1001B \quad [X_2]_{\text{原}} = 1110\ 1001B$$

其中最高位为符号,后面 7 位是数值。用原码表示时,+105 和 -105 的数值部分相同而符号位相反。

原码表示简单易懂,而且与真值的转换方便。但若是两个异号数相加,或两个同号数相减,就要做减法。为了把减运算转换为加运算,从而简化计算机的结构,就引进了反码和补码。

3. 反码

正数的反码与原码一样,符号位为 0,其余位为其数值;负数的反码为它的绝对值(即与其绝对值相等的正数)按位(连同符号位)取反。如

$$X_1 = 105 = +0110\ 1001B \quad [X_1]_{\text{反}} = 0110\ 1001B$$

$$X_2 = -105 = -0110\ 1001B \quad [X_2]_{\text{反}} = 1001\ 0110B$$

4. 补码

正数的补码与原码一样,符号位为 0,其余位为其数值;负数的补码为它的绝对值(即与该负数的绝对值相等的正数)的补数。把一个数连同符号位按位取反再加 1,可以得到该数的补数。如

$$X_1 = 105 = +0110\ 1001B \quad [X_1]_{\text{补}} = 0110\ 1001B$$

$$X_2 = -105 = -0110\ 1001B \quad [X_2]_{\text{补}} = 1001\ 0111B$$

求补数还可以直接求,方法是从最低位向最高位扫描,保留直至第一个“1”的所有位,以后各位按位取反。负数的补码可以由与其绝对值相等的正数求补得到。根据两数互为补数的原理,对补码表示的负数求补就可以得到该负数的绝对值。如

$$[-105]_{\text{补}} = 1001\ 0111B = 97H$$

对其求补,从右向左扫描,第一位就是 1,故只保留该位,对其左面的七位均求反得: 0110 1001,即补码表示的机器数 97H 的真值是 -69H (= -105)。

一个用补码表示的机器数,若最高位为 0,则其余几位即为此数的绝对值;若最高位为 1,则其余几位不是此数的绝对值,把该数(连同符号位)求补,才得到它的绝对值。

当数采用补码表示时,就可以把减法转换为加法。例如

$$64 - 10 = 64 + (-10)$$

$$[64]_{\text{补}} = 40H = 0100\ 0000B$$

$$[10]_{\text{补}} = 0AH = 0000\ 1010B$$

$$[-10]_{\text{补}} = 1111\ 0110B$$

做减法运算过程如下:

$$\begin{array}{r} 0100\ 0000 \\ -0000\ 1010 \\ \hline 0011\ 0110 \end{array}$$

用补码相加过程如下：

$$\begin{array}{r} 0100\ 0000 \\ +1111\ 0110 \\ \hline 1\ 0011\ 0110 \\ \text{↑进位自然丢失} \end{array}$$

结果相同，其真值为：54(36H=48+6)。

最高位的进位是自然丢失的，故做减法与用补码相加的结果是相同的。因此，在微型机中，凡是符号数一律是用补码表示的。一定要记住运算的结果也是用补码表示的。如

$$\begin{aligned} 34-68 &= 34+(-68) \\ 34 &= 22\text{H} = 0010\ 0010\text{B} \\ 68 &= 44\text{H} = 0100\ 0100\text{B} \\ -68 &= 1011\ 1100\text{B} \end{aligned}$$

做减运算过程如下：

$$\begin{array}{r} 0010\ 0010 \\ -0100\ 0100 \\ \hline 1\ 1101\ 1110 \\ \text{↑借位自然丢失} \end{array}$$

用补码相加过程如下：

$$\begin{array}{r} 0010\ 0010 \\ +1011\ 1100 \\ \hline 1101\ 1110 \end{array}$$

结果相同。因为符号位为 1，所以结果为负数。对其求补，得其真值：-0010 0010B，即为 -34(-22H)。

由上面两个例子还可以看出，当数采用补码表示后，两个正数相减，若无借位，化为补码相加就会有进位；若有借位，化作补码相加就不会有进位。

5. 8 位二进制数的范围

8 位二进制数，将其看作无符号数和符号数，它所表示的数的大小是不同的。为加深读者的印象，将其列于表 1-3 中。

表 1-3 8 位二进制数(2 位十六进制数)的大小

8 位二进制数	2 位十六进制数	无符号数	原码数	反码数	补码数
0000 0000	00	0	0	0	0
0000 0001	01	1	1	1	1
0000 0010	02	2	2	2	2
⋮	⋮	⋮	⋮	⋮	⋮
0111 1101	7D	125	125	125	125
0111 1110	7E	126	126	126	126
0111 1111	7F	127	127	127	127
1000 0000	80	128	-0	-127	-128
1000 0001	81	129	-1	-126	-127
1000 0010	82	130	-2	-125	-126
⋮	⋮	⋮	⋮	⋮	⋮
1111 1101	FD	253	-125	-2	-3
1111 1110	FE	254	-126	-1	-2
1111 1111	FF	255	-127	-0	-1