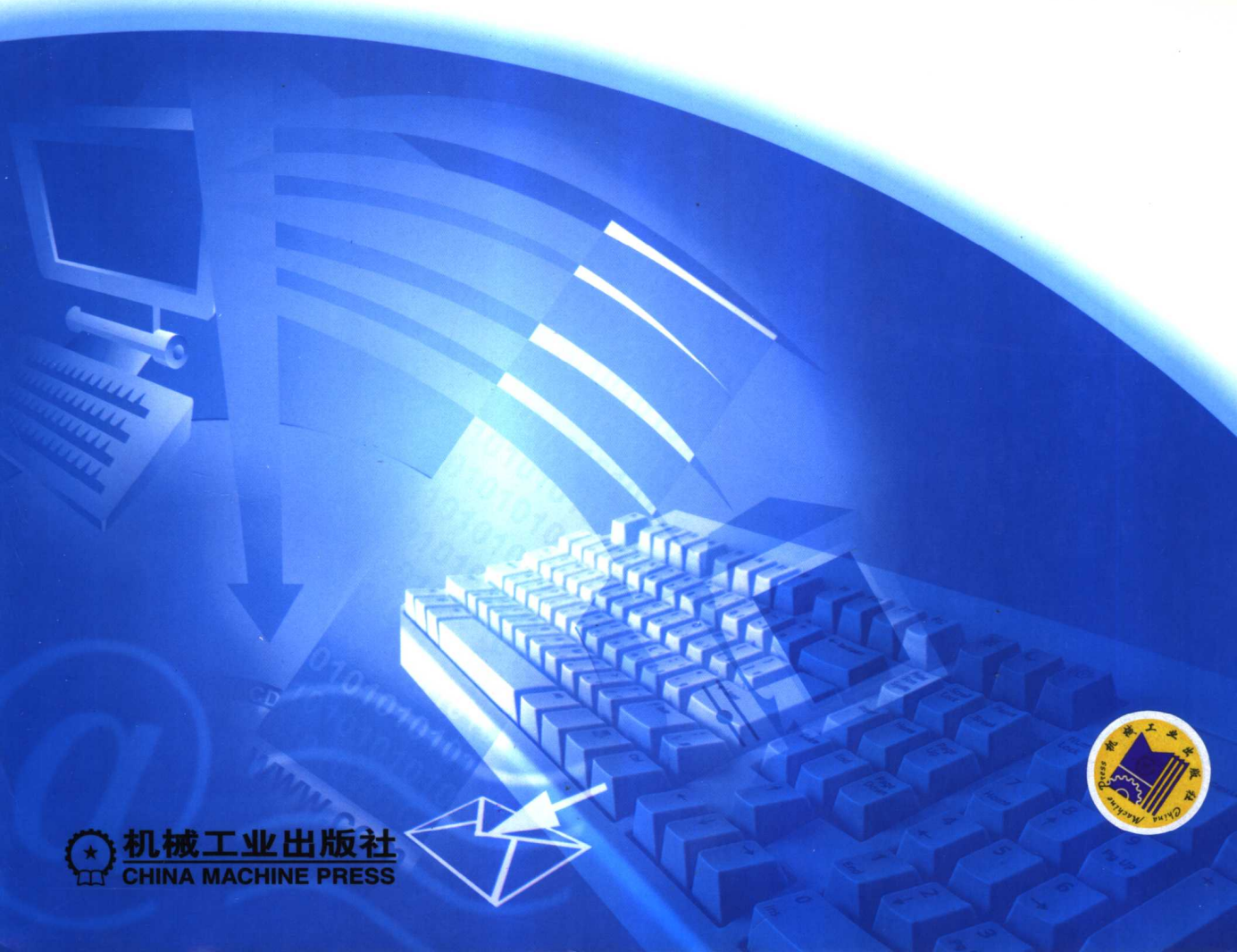
普通高等教育规划教材

80X86汇编语言程序设计

廖 智 主编



机械工业出版社
CHINA MACHINE PRESS

普通高等教育规划教材

80X86汇编语言程序设计

主　编　廖　智
副主编　徐爱芸
参　编　赵鸿宇
主　审　杨路明

机械工业出版社

本书以当前广泛使用的80X86系列微型计算机为背景，系统介绍了汇编语言程序设计的基本理论和方法；以大量实例讲述了如何用汇编语言开发应用程序，以及上机调试和运用汇编语言程序的方法，同时详细介绍了80386后继机型的增强功能。

本书共分八章，内容由浅入深，循序渐进，力求遵循面向应用、重视实践的原则，着重培养学生动手能力和思维方法。

本书是应用型本科计算机科学与技术专业规划教材，适用于一般工科院校计算机及相关专业的本科教学使用，也可作为计算机工作者学习汇编语言的自学参考书。

图书在版编目（CIP）数据

80X86汇编语言程序设计/廖智主编 .—北京：机械工业出版社，2004.8
普通高等教育规划教材
ISBN 7-111-14598-4

Ⅰ.8... Ⅱ.廖 ... Ⅲ.汇编语言－程序设计－高等学校－教材
Ⅳ.TP313

中国版本图书馆CIP数据核字（2004）第052295号

机械工业出版社（北京市百万庄大街22号 邮政编码100037）
策划编辑：王小东 责任编辑：冷 彬 版式设计：霍永明
责任校对：贾卫东 封面设计：饶 薇 责任印制：李 妍
北京机工印刷厂印刷·新华书店北京发行所发行
2004年7月第1版第1次印刷
787mm×1092mm 1/16·13.75印张·340千字
定价：20.00元

计算机科学与技术专业分委员会委员名单

普通高等教育应用型人才培养规划教材
编审委员会名单

序

工程科学技术在推动人类文明的进步中一直起着发动机的作用。随着知识经济时代的到来，科学技术突飞猛进，国际竞争日趋激烈。特别是随着经济全球化发展和我国加入 WTO，世界制造业将逐步向我国转移。有人认为，我国将成为世界的“制造中心”。有鉴于此，工程教育的发展也因此面临着新的机遇和挑战。

迄今为止，我国高等工程教育已为经济战线培养了数百万专门人才，为经济的发展作出了巨大的贡献。但据 IMD1998 年的调查，我国“人才市场上是否有充足的合格工程师”指标排名世界第 36 位，与我国科技人员总数排名世界第一形成很大的反差。这说明符合企业需要的工程技术人员特别是工程应用型技术人才市场供给不足。在此形势下，国家教育部近年来批准组建了一批以培养工程应用型本科人才为主的高等院校，并于 2001、2002 年两次举办了“应用型本科人才培养模式研讨会”，对工程应用型本科教育的办学思想和发展定位作了初步探讨。本系列教材就是在这种形势下组织编写的，以适应经济、社会发展对工程教育的新要求，满足高素质、强能力的工程应用型本科人才培养的需要。

航天工程的先驱、美国加州理工学院的冯·卡门教授有句名言：“科学家研究已有的世界，工程师创造未有的世界。”科学在于探索客观世界中存在的客观规律，所以科学强调分析，强调结论的惟一性。工程是人们综合应用科学（包括自然科学、技术科学和社会科学）理论和技术手段去改造客观世界的实践活动，所以它强调综合，强调方案优缺点的比较并做出论证和判断。这就是科学与工程的主要不同之处。这也就要求我们对工程应用型人才的培养和对科学研究型人才的培养应实施不同的培养方案，采用不同的培养模式，采用具有不同特点的教材。然而，我国目前的工程教育没有注意到这一点，而是：①过分侧重工程科学（分析）方面，轻视了工程实际训练方面，重理论，轻实践，没有足够的工程实践训练，工程教育的“学术化”倾向形成了“课题训练”的偏软现象，导致学生动手能力差。②人才培养模式、规格比较单一，课程结构不合理，知识面过窄，导致知识结构单一，所学知识中有一些内容已陈旧，交叉学科、信息学科的内容知之甚少，人文社会科学知识薄弱，学生创新能力不强。③教材单一，注重工程的科学分析，轻视工程实践能力的培养；注重理论知识的传授，轻视学生个性特别是创新精神的培养；注重教材的系统性和完整性，造成课程方面的相互重复、脱节等现象；缺乏工程应用背景，存在内容陈旧的现象。④老师缺乏工程实践经验，自身缺乏“工程训练”。⑤工程教育在实践中与经济、产业的联系不密切。要使我国工程教育适应经济、社会的发展，培养更多优秀的工程技术人才，我们必须努力改革。

组织编写本套系列教材，目的在于改革传统的高等工程教育教材，建设一套富有特色、有利于应用型人才培养的本科教材，满足工程应用型人才培养的要求。

本套系列教材的建设原则是：

1. 保证基础，确保后劲

科技的发展，要求工程技术人员必须具备终生学习的能力。为此，从内容安排上，保证学生有较厚实的基础，满足本科教学的基本要求，使学生日后具有较强的发展后劲。

2. 突出特色，强化应用

围绕培养目标，以工程应用为背景，通过理论与工程实际相结合，构建工程应用型本科教育系列教材特色。本套系列教材的内容、结构遵循如下9字方针：知识新、结构新、重应用。教材内容的要求概括为："精"、"新"、"广"、"用"。"精"指在融会贯通教学内容的基础上，挑选出最基本的内容、方法及典型应用；"新"指将本学科前沿的新进展和有关的技术进步新成果、新应用等纳入教学内容，以适应科学技术发展的需要。妥善处理好传统内容的继承与现代内容的引进。用现代的思想、观点和方法重新认识基础内容和引入现代科技的新内容，并将这些按新的教学系统重新组织："广"指在保持本学科基本体系下，处理好与相邻以及交叉学科的关系；"用"指注重理论与实际融会贯通，特别是注入工程意识，包括经济、质量、环境等诸多因素对工程的影响。

3. 抓住重点，合理配套

工程应用型本科教育系列教材的重点是专业课（专业基础课、专业课）教材的建设，并做好与理论课教材建设同步的实践教材的建设，力争做好与之配套的电子教材的建设。

4. 精选编者，确保质量

遴选一批既具有丰富的工程实践经验，又具有丰富的教学实践经验的教师担任编写任务，以确保教材质量。

我们相信，本套系列教材的出版，对我国工程应用型人才培养质量的提高，必将产生积极作用，会为我国经济建设和社会发展作出一定的贡献。

机械工业出版社颇具魄力和眼光，高瞻远瞩，及时提出并组织编写这套系列教材，他们为编好这套系列教材做了认真细致的工作，并为该套系列教材的出版提供了许多有利的条件，在此深表衷心感谢！

编委会主任
湖南工程学院院长　　刘国荣教授

前　言

“汇编语言程序设计”是计算机专业的重要专业基础课，是从事计算机研究与应用，特别是软件研究的基础，是计算机专业必修的核心课程之一。该课程从系统软件和应用软件设计的角度出发，以目前使用最为广泛的80X86系列微型计算机为例，系统地介绍了汇编语言的基本概念、基本原理和程序设计的常用方法与技术，并通过具体实例，叙述了用计算机解决实际问题的全过程，同时还介绍了在80X86机上调试运行汇编语言程序的方法及其最新集成软件CmasmW的使用。

本书作为应用型本科计算机科学与技术专业规划教材之一，是以着重培养学生动手能力和思维方法为目的编写的。全书在内容的选取、概念的引入、文字的叙述以及例题和习题的选择等方面，都遵循面向应用、重视实践的原则。

全书共分九章。第一章介绍了学习80X86汇编语言程序设计所需的基础知识；第二章详细介绍了七种寻址方式及指令系统的格式、功能及使用方法；第三章主要介绍了伪指令、汇编语言程序结构和汇编语言程序的上机过程；第四章先介绍了常用的DOS功能调用，然后，系统地介绍了顺序、分支和循环程序设计的基本方法及技巧；第五章重点介绍了子程序设计的基本方法及技巧；第六章主要介绍了输入/输出的程序设计和中断程序设计的概念和方法，以及ROM BIOS和DOS中断调用的调用方法，其中还特别介绍了“鼠标中断调用”；第七章全面介绍了高级汇编语言技术：宏汇编、重复汇编、条件汇编、多模块程序设计以及汇编语言与高级语言的连接；第八章简介了80386后继机型的增强功能。第九章的实验部分具体规定了各次实验的目的要求、实验内容以及实验思考题，以方便读者自学和加强实践环节。

汇编语言程序设计是一门实践性很强的课程，它需要复杂的脑力劳动，还可以培养学生的创造性和动手能力。为了帮助读者更好地掌握汇编语言程序设计的技巧，书中结合应用安排了丰富的例题和习题。读者可以通过这些例题学习一些规律，并且要认真完成习题，同时做到多编程序，多上机实践，这样，才能真正掌握程序设计的方法与技巧。

本书由廖智担任主编，负责全书的总体规划和统稿工作。其中，赵鸿宇编写第一、二章；廖智编写第三～五章，第六章的部分，第九章的部分（实验一～四）和附录；徐爱芸编写第七、八章，第六章的部分和第九章的部分（实验

五～八)。中南大学的杨路明教授在百忙中审阅了全部书稿，并提出了宝贵的修改意见，在此表示衷心的感谢。

由于编者水平有限，书中不妥或错误之处在所难免，殷切希望广大读者批评指正。

编　者

目　录

第一章　基础知识

本章介绍学习汇编语言程序设计所必须具备的基本知识，主要包括汇编语言的基本概念、计算机中数据和字符的表示以及80X86计算机系统的存储器组织和寄存器组。通过本章的学习，应该了解汇编语言的概念和特点、计算机中数据和字符的常用表示方法，熟练掌握80X86 CPU的寄存器组以及存储器的分段，掌握几个常用标志的含义及判断方法。

第一节　汇编语言简介

一、汇编语言

计算机程序设计语言通常分为三类：机器语言（Machine Language）、汇编语言（Assembly Language）和高级语言（High Level Language）。

汇编语言是一种面向机器的程序设计语言，和机器语言一样，与机器密切相关，是一种低级语言。

我们知道，计算机的功能是记忆、传输和加工二进制信息，能够直接识别并处理由0、1组成的二进制代码。机器指令是指用二进制编码的指令，指示计算机要进行的操作及操作对象。二进制编码的指令集合以及指令的使用规则，构成了计算机惟一能够识别的语言——机器语言。这种二进制编码形式的机器语言不仅复杂难记，而且还依赖于具体的机型，于是人们将机器指令符号化，用直观、便于记忆的英文缩写符号来表示机器指令，这些符号被称作助记符（Mnemonic）。例如，MOV表示传送指令，XCHG表示交换指令，SUB表示减法指令等。

用助记符描述的指令称作汇编格式指令或符号指令，通常简称为指令。指令的集合及其程序设计规则便构成了汇编语言，用汇编语言编写的程序就是汇编语言源程序。通常，在80X86系列微型计算机上，汇编语言源程序存放在以．ASM为扩展名的磁盘文件中。

例如：利用汇编语言编写出将数6和数7相加的程序为：

```
MOV   AL，6
ADD   AL，7
HLT
```

可见，用汇编语言编写的程序比较简洁易读。但是计算机并不识别助记符，指令必须用机器码来表示，同样，数字也只能用二进制数或十六进制数表示，即需要将指令和数据翻译成机器能识别的二进制代码。这样，上述程序对应的二进制代码如下：

第一条指令　1011 0000（MOV AL，n）

　　0000 0110（n＝6）

第二条指令　0000 0100（ADD AL，n）

0000 0111（n = 7）

第三条指令 1111 0100（HLT）

显然，用汇编语言编写的程序要比机器代码更容易理解，但其仅仅是机器语言的符号化表示，每条汇编语言指令均应对应惟一的机器指令，因而与机器语言并没有本质区别，只是在直观和记忆方面有了改进。

虽然汇编语言较机器语言有了很大改进，但并无本质上的飞跃。人们迫切希望有一种接近自然语言或数学表达形式的程序设计语言，使程序设计工作能避开与机器硬件相关的细节，而着重于解决问题的算法本身，于是便产生了高级语言，例如 PASCAL、FORTRAN、BASIC、C 等。高级语言是面向问题的，在程序设计的简易性与可移植性等方面有了质的提高，但是，用高级语言编写的源程序必须经过编译和连接，将其转换为可执行程序或借助于解释程序方可运行。

二、汇编语言的特点

尽管汇编语言比较复杂难学、不易理解，而且与具体机型的硬件结构密切相关，可移植性差，开发效率较低，但是与高级语言相比仍具有很多优势：

1）用汇编语言容易得到高时空效率的程序。由于汇编语言本质上就是机器语言，可以直接、有效地控制计算机硬件，因而与高级语言相比，容易得到运行速度快、执行代码短、占用内存空间少的高时空效率的目标程序。当然，随着计算机运行速度的提高和内存容量的增加，有些人认为汇编语言的这一优势将不再突出。

2）用汇编语言可以实现高级语言难以胜任甚至不能完成的任务。汇编语言具有直接和简捷的特点，用它编制程序能精确地描述算法，充分发挥计算机硬件的功能。在过程控制、多媒体接口、通信等方面的程序设计时，用汇编语言来得直接方便，执行速度快，效率高。

是否采用汇编语言编写程序，要看具体的应用场合，在软件开发时间及软件质量方面进行权衡和选择。一般来说，某些对执行时间和存储容量要求较高的程序采用汇编语言编写，如实时控制系统、智能化仪器仪表及高性能软件等。

第二节 计算机中的数制

一、二进制数

多位数码中每一位的构成方法以及从低位到高位的进位规则，我们称为数制。十进制（Decimal）是日常生活和学习中最常使用的进位计数制。在十进制数中，有 0、1、…、9 十个数码，所以计数的基数是 10。超过 9 的数必须用多位数表示，其中低位和相邻高位之间的关系是“逢十进一”，故称为十进制。例如：

$$145.87 = 1\times10^{2} + 4\times10^{1} + 5\times10^{0} + 8\times10^{-1} + 7\times10^{-2}$$

所以任意一个十进制数 D 均可表示为每位数字乘以其权所得到的乘积之和，即

$$D = \sum(K_i \times 10^i) \qquad (1\text{-}1)$$

其中，K_i 是第 i 位的系数，它可以是 0～9 这十个数码中的任何一个，10^i 称为该位数字的权。

为了便于存储及计算的物理实现，现代计算机系统采用二进制（Binary）来表示数据。在二进制数中，只有0和1两个数码，所以计数基数为2，低位和相邻高位间的进位关系是“逢二进一”，根据式（1-1），任何一个二进制数均可表示为

$$D = \sum(K_i \times 2^i) \tag{1-2}$$

例如：$(101.11)_2 = 1\times2^2 + 0\times2^1 + 1\times2^0 + 1\times2^{-1} + 1\times2^{-2} = (5.75)_{10}$

其中下标2和10表示括号里的数是二进制和十进制数，有时也用B(Binary)和D(Decimal)代替2和10这两个下标。

将十进制数的整数部分转换成二进制数通常采用“除2取余法”，它是将整数部分逐次被2除，依次记下余数，直到商为0，第一个余数为二进制整数的最低位，最后一个余数为最高位；将十进制数的小数部分转换成二进制数采用“乘2取整法”，它是将小数部分连续乘以2，取乘数的整数部分作为二进制的小数，第一次取整得到的数为二进制小数的最高位。若一个十进制数既有整数部分，又有小数部分，则可将整数部分和小数部分分别转换成对应的二进制整数和小数，再将转换结果合并在一起，就得到了该十进制数转换的完整结果。

例1-1 将十进制数 $(107.625)_{10}$ 转换成二进制数。

1）整数部分转换

余 数

2 | 107	…… 1	…… K_0	最低位
2 | 53	…… 1	…… K_1	↑
2 | 26	…… 0	…… K_2	↑
2 | 13	…… 1	…… K_3	↑
2 | 6	…… 0	…… K_4	↑
2 | 3	…… 1	…… K_5	↑
2 | 1	…… 1	…… K_6	最高位
0			

所以，$(107)_{10} = (K_6\ K_5\ K_4\ K_3\ K_2\ K_1\ K_0)_2 = (1101011)_2$

2）小数部分转换

$0.625 \times 2 = 1.250$　　整数部分为1，$K_{-1} = 1$

$0.250 \times 2 = 0.500$　　整数部分为0，$K_{-2} = 0$

$0.500 \times 2 = 1.000$　　整数部分为1，$K_{-3} = 1$

所以，$(0.625)_{10} = (0.K_{-1}K_{-2}K_{-3})_2 = (0.101)_2$

由此可得十进制数107.625对应的二进制数为1101011.101，即

$$(107.625)_{10} = (1101011.101)_2$$

二、十六进制数

1. 十六进制数

在计算机中，所有运算和判断都是通过采用二进制数来进行的，从数据到指令、从地址到内容均是用二进制数来表示。但是二进制数阅读、书写起来很不方便，为了便于人们对二进制数的描述，应该选择一种易于与二进制数相互转换的数制，在计算机应用中，常用的有八进制数（Octal）和十六进制数（Hexadecimal），这里主要介绍十六进制数。

十六进制数以 16 为基数，每一位有十六个不同的数码，分别用 0 ~ 9、A、B、C、D、E、F 表示，低位和相邻高位间的进位关系是“逢十六进一”，根据式 (1-1)，任何一个十六进制数均可表示为

$$D = \sum (K_i \times 16^i) \tag{1-3}$$

例如：$(2AD.7F)_{16} = 2 \times 16^2 + 10 \times 16^1 + 13 \times 16^0 + 7 \times 16^{-1} + 15 \times 16^{-2}$

$= (685.4960937)_{10}$

式中的下脚标 16 表示括号里的数是十六进制数，有时也用 H(Hexadecimal)代替这个脚标。

十进制转换成十六进制数时，同样地，整数部分可采用“除 16 取余法”，小数部分采用“乘 16 取整法”，这里不再赘述。

由于十六进制数的基数 $16 = 2^4$，故每位十六进制数由 4 位二进制数构成。因此，二进制数转换为十六进制数的方法是：整数部分从低位开始，每 4 位二进制数为一组，最后不足 4 位的，则在最高位加 0 补足 4 位；小数部分从高位开始，每 4 位二进制数为一组，最后不足 4 位的，在低位加 0 补足 4 位，然后用对应的十六进制数来代替，再按顺序写出对应的十六进制数。

例 1-2 将二进制数$(10011111011.111011)_2$转换成十六进制数。

0100	1111	1011	.	1110	1100
↓	↓	↓		↓	↓
4	F	B	.	E	C

所以，$(10011111011.111011)_2 = (4FB.EC)_{16}$

十六进制数转换成二进制数时，将每位十六进制数用 4 位二进制数来代替，再按原来的顺序排列起来便得到了相应的二进制数。

例 1-3 将十六进制数$(3BE5.97D)_{16}$转换成二进制数。

3	B	E	5	.	9	7	D
↓	↓	↓	↓		↓	↓	↓
0011	1011	1110	0101	.	1001	0111	1101

所以，$(3BE5.97D)_{16} = (11101111100101.100101111101)_2$

2.十六进制数的运算

十六进制数的加减运算可以采用先把该十六进制数转换成十进制数，经过计算后再把结果换成为十六进制数的方法，但这样做比较繁琐。其实，只要按照逢十六进一的规则，直接用十六进制数来计算也是很方便的。

例 1-4

05C3H	3D25H
+3D25H	−05C3H
42E8H	3762H

十六进制数的乘、除运算，可以先将十六进制数转换为十进制数进行乘、除运算，然后再将结果转换为十六进制数。这里不再具体说明。

第三节 计算机中数和字符的表示

一、数值表示

计算机中的数是用二进制来表示的，数的符号也是用二进制表示的。在机器中，把一个数连同其符号在内数值化表示的数称为机器数，而它的数值称为机器数的真值。通常一个数的最高位为符号位，正数用 0 表示，负数用 1 表示，例如：

真值 X = +1011011　　其对应的机器数为 01011011；

真值 X = -1011011　　其对应的机器数为 11011011。

机器数可以用不同的编码来表示，常用的有原码、补码和反码表示方法。上例的机器数就是用原码表示的。多数机器采用补码表示法，80X86 机也是这样，所以我们这里只介绍补码表示法。

正数的补码是其本身，即其最高位为符号位，用 0 表示正，其余位为数值位。

例如：假设机器字长为 8 位，则$[+0]_{补}=00000000$，$[+127]_{补}=01111111$。

负数的补码是对其正数各位求反，末位（最低位）加 1 后形成的。

例如：假设机器字长为 16 位，求 -373 的补码。计算过程如下：

+373 二进制表示为	0000	0001	0111	0101
各位求反后为	1111	1110	1000	1010
末位加 1 后为	1111	1110	1000	1011
用十六进制数表示为	F	E	8	B

即 $[-373]_{补}=FE8BH$

我们把这种对一个机器数补码按位求反后在末位加 1 的运算称为求补运算，可以证明补码具有以下特性：

1) $$[X]_{补} \xleftrightarrow{\text{求补}} [-X]_{补}$$

例如：在 8 位二进制数表示下，

$$[X]_{补}=0000\quad 1010B$$

按位求反后得　1111　0101

末位加 1 后得　1111　0110

此数正是$[-X]_{补}=11110110B$

2)$[X+Y]_{补}=[X]_{补}+[Y]_{补}$。

3)$[X-Y]_{补}=[X]_{补}+[-Y]_{补}$。

当负数采用补码表示时，就可以将减法运算转换为加法运算了。

例如：设机器字长为 8 位，X = 64、Y = 10，用补码运算方法求 X - Y。

$$X=1000000B$$

$$[X]_{补}=01000000B$$

$$Y=0001010B$$

$$[Y]_{补} = 00001010B$$

$$[-Y]_{补} = 11110110B$$

于是

$$\begin{array}{r} 01000000 \\ -\ 00001010 \\ \hline 00110110 \end{array} \qquad \begin{array}{r} 01000000 \\ +\ 11110110 \\ \hline (进位)\ 1\ \ 00110110 \end{array}$$

由于机器字长的限制，从最高有效位向高位的进位或借位是自然丢失的，故做减法运算与用补码做加法运算的结果是相同的，同时计算机将把这一信息保留在标志寄存器中。

所以，$[X-Y]_{补} = [X]_{补} + [-Y]_{补} = 00110110B$

则　　$X - Y = +0110110B$

二进制补码表示是基于固定长度的。换句话说，一个有符号数在不同位数下，其二进制补码表示可能是不同的。例如，-1 的 8 位二进制数补码表示是 FFH，-1 的 16 位二进制数补码表示是 FFFFH，可见差异之大。

有时，需要将一个二进制数从较少位数扩展到较多位数，如从 8 位扩展到 16 位，或者从 16 位扩展到 32 位等，并保持在特定意义下的取值不变，这就是所谓的符号扩展与零扩展。对于用补码表示的数，正数的符号扩展应该在前面补 0，而负数的符号扩展则应该在前面补 1。例如，我们已经知道机器字长为 8 位，X = 46，则

$[X]_{补} = 00101110B$，$[-X]_{补} = 11010010B$

如果把它们从 8 位扩展到 16 位，则

$[X]_{补} = 0000000000101110B = 002EH$

$[-X]_{补} = 1111111111010010B = FFD2H$

在有些情况下，计算机要处理的数全是正数，此时再保留符号位就没有意义了。我们通常把最高有效位也作为数值处理，这样的数称为无符号数。无符号整数既可以表示纯数字，又可表示存储单元的地址。

零扩展是将 0 添入扩展的每一位，使得在无符号数意义下取值不变。

例如：00100110B = 26H

如果对它进行 16 位零扩展，则为

$$0000000000100110B = 0026H$$

二、字符表示

除了数值以外，人们有时还需要用计算机处理字符或字符串。例如，从键盘输入的信息或打印输出的信息都是以字符方式输入输出的。字符一般分成两大类：一类为可打印的图形字符，它包括：52 个大小写英文字母，0～9 十个数字和 34 个专用符号（如 +、-、*、/、=、%、…）；另一类是作为命令的控制字符，它包括 32 个命令，例如，回车（CR）、换行（LF）、换页（FF）、送毕（EOT）等等。要注意的是，控制字符在不同输出设备上可能会执行不同的操作，没有非常规范的标准。

这些计算机不能直接识别的字符，只有采用二进制的编码方式后才能识别。80X86 机采用目前最常用的美国信息交换标准代码 ASCII（American Standard Code for Information Inter-