计算机系列
全国高职计算机专业教材
院士教授、企业资深从业人员、职教一线教师共同打造
◎顾问 张效祥 院士　◎总主编 邱玉辉 教授

微型计算机原理

张开成 主编

西南师范大学出版社

全国高职计算机专业教材

院士教授、企业资深从业人员、职教一线教师共同打造

◎ 顾问 张效祥 院士 ◎ 总主编 邱玉辉 教授

微型计算机原理

西南师范大学出版社

图书在版编目(CIP)数据

微型计算机原理/张开成主编. —重庆:西南师范大学出版社,2006.7

(全国高职计算机专业教材/邱玉辉主编)

ISBN 7-5621-3665-3

Ⅰ.微… Ⅱ.张… Ⅲ.微型计算机-高等学校:技术学校-教材 Ⅳ.TP36

中国版本图书馆 CIP 数据核字(2006)第 070426 号

全国高职计算机专业教材

顾　　问:张效祥 院士

总 主 编:邱玉辉 教授

总 策 划:周安平　李远毅

执行策划:周　松　张浩宇

微型计算机原理

张开成　主编

责任编辑:李　俊

封面设计:唐小慧　西　西

出版发行:西南师范大学出版社

(重庆·北碚　邮编 400715

网址:http://www.xscbs.com)

印 刷 者:重庆诚凤印务有限公司

开　　本:787mm×1092mm　1/16

印　　张:17.5

字　　数:448 千字

版　　次:2006 年 8 月　第 1 版

印　　次:2006 年 8 月　第 1 次印刷

书　　号:ISBN 7-5621-3665-3/TP·71

定　　价:26.50 元

《全国高职计算机专业教材》编委会联系方式

联 系 人:周　松　张浩宇

电　　话:023-68254356　13908317565　13883206497

地　　址:重庆市北碚区西南师范大学出版社内

邮　　编:400715

E-mail:qggzjsjjc@yahoo.com.cn

《全国高职计算机专业教材》总编委会

总编委会顾问

张效祥 中国科学院院士、著名计算机专家、“两弹一星”功臣

总编委会主任

邱玉辉 西南大学人工智能研究所所长、教授、博士生导师

总编委会副主任

黄国兴	华东师范大学软件学院	院长、教授
王能忠	四川托普信息技术职业学院	院长、教授
张为群	西南大学计算机与信息科学学院	院长、教授
汪林林	重庆邮电大学软件学院	原院长、教授
李吉桂	华南师范大学计算机科学系	原系主任、教授
张　杰	西北大学软件职业技术学院	院长、教授
徐爱容	重庆电子职业技术学院计算机系	主任、教授

丛书总序

CONGSHU ZONGXU

总主编 邱玉辉

高等职业教育是我国高等教育体系的重要组成部分。近年来，国家高度重视职业教育，并为推动我国职业教育跨越式发展，颁发了《国务院关于大力推进职业教育改革与发展的决定》，提出了将高等职业教育学制逐步由目前的三年改为两年的改革方向。

教材是提高教育质量的关键之一。信息产业部电子教育中心调查后认为，现在使用的教材多数是普通高校本科教材的压缩和简化，偏重理论知识的介绍，而案例教学、项目教学的内容极少，实用技能的训练更是不足，课程内容滞后于专业技术的更新与发展，与社会需求和行业发展相脱节，从而导致学生分析问题和解决问题的能力，特别是职业能力较弱，毕业的学生很少能直接顶岗工作。

为落实国家大力发展职业技术教育的重大决策和解决目前缺乏面向两年学制的高职计算机专业系列教材的问题，我们组织开发了这套《全国高职计算机专业教材》。

这套教材由我国著名计算机专家、“两弹一星”功臣张效祥院士担任顾问，并得到中央教育科学研究所的大力支持。其编写指导思想是：需求牵引，改革驱动，理论适度，着眼技术，立足实用，培养能力。我们通过总结当前职业教育专家教学改革的最新研究成果，紧紧依靠高职院校从事计算机教育的一线教师，以培养技能型紧缺人才为目标，让学生明白Why，知道What，重点学会How。把理论与实践融为一体，既考虑了每门课程本身的科学性，又兼顾了课程间的联系与衔接。全套教材具有重点突出，针对性强；结构清晰，循序渐进；模块结构，易教易学等特点。此外，我们还将为教材配备包含教参和习题解答等内容的光盘，供教师参考和学生自学。

总之，这套教材经过长期策划，精心打造，认真审读，终于问世了。它倾注了编写教师、总编委会以及出版社的大量心血。如果它能够对我们的高职计算机教育有所助益，那么我们的目的就达到了。

前言

QIANYAN

随着计算机科学技术的进步和发展，微型计算机的应用已经渗透到各个领域。微型计算机原理是学习和使用微型计算机的必备基础，是设计和开发各类微机应用系统的关键。本书正是为高职/高专各专业师生及一般科技人员学习微型计算机而编写。

本书以当前应用极为广泛的 PC 系列微机为背景，从系统角度出发，在讲清原理的基础上，强调实际应用。在内容的安排上，够用为度，难度适中。既注重基础知识的掌握和实用技能的培养，又体现新技术的发展；硬件部分着重说明 CPU 与接口芯片的功能及应用，软件部分强调与硬件结合；内容系统、循序渐进、由浅入深。

本书共分八章，主要内容如下：

第一章 讲述微型计算机的发展史、计算机中的数和编码以及微型计算机系统的组成和基本工作过程。

第二章 讲述 8086 微处理器的编程结构、外部特性和操作时序。

第三章 讲述 8086 指令系统及应用编程。

第四章 讲述内存储器的基本结构和功能，同时也简要介绍外存储器的组成和作用。

第五章 讲述微机输入输出系统以及微机系统中常用的几种总线标准。

第六章 讲述中断技术，并简要介绍 8259A 中断控制器的应用。

第七章 讲述 8255A 并行接口芯片、8251A 串行接口芯片及 8253A 定时/计数芯片的结构、功能和应用。

第八章 简要介绍微机系统的设计。

本课程建议采用两种学时：长学时为 80 学时，其中：第一章：10 学时，第二章：12 学时，第三章：12 学时，第四章：10 学时，第五章：6 学时，第六章：10 学时，第七章：16 学时，第八章：4 学时；短学时为 60 学时，只讲述前六章。

本书是一本实用性较强的专业基础课教材，适合于作高职/高专计算机专业《微机原理及应用》和《微机原理及接口技术》等课程的教材。

本书由张开成主编。第一、二、三、七章由张开成编写，第四、八

章由钟文龙编写，第五章由余辉编写，第六章由叶沿飞编写，全书由张开成统稿、定稿。

限于编者的水平，且时间仓促，书中难免有错误和不妥之外，恳请读者不吝赐教、指正。

编者

2006 年 3 月

内容简介

本书以 INTEL 系列微处理器为对象，主要介绍微型计算机的体系结构、工作过程、8086 指令系统及汇编语言程序设计、存储器、输入输出系统、中断技术、接口技术、微机应用系统设计，并重点阐述微机的基本工作原理及在工程实践中的应用。

本书内容简明扼要、深入浅出、通俗易懂，融入作者从事多年教学及工程实践应用的体会和经验。通过本书学习，将为微机应用打下坚实的基础。本书可作为高职/高专各专业的教学用书，也可作为一般工程技术人员从事技术工作的参考用书。

目 录

第一章　微机系统概述

【学习要求】

通过本章的学习，使学生对微型计算机有一个整体认识，掌握微型计算机中信息的表示方法及有关基本概念，了解微型计算机的系统结构和硬件组成，理解微型计算机的基本工作过程，为学习后续章节奠定坚实基础。

【主要内容】

本章主要介绍微型计算机的发展史，计算机中的数和编码，微型计算机的基本概念和体系结构以及微型计算机的工作过程。

第一节　微型计算机的发展史

以半导体集成电路为中心的微电子技术的进步，使计算机向着微型化、高性能、低成本的方向迅猛发展。自1946年第一台电子计算机问世以来，计算机已经历了从电子管、晶体管、集成电路到大规模和超大规模集成电路的发展演变。计算机通常按其体积、性能和价格可分为巨型机、大型机、中型机、小型机和微型机五类。由于微型机具有体积小、重量轻、功耗低、价格便宜、结构灵活等特点，从而得到了广泛的普及和应用。

微型计算机的发展史也就是微处理器的发展史，自从20世纪70年代初第1个微处理器诞生以来，微处理器的性能和集成度几乎每两年提高一倍，而价格却降低了一个数量级。

1971 年 10 月，美国 Intel 公司首先推出 4004 微处理器，采用 PMOS 技术，在 4.2mm × 3.2mm 的硅片上集成了 2250 个晶体管，可进行 4 位二进制的并行处理。这是实现 4 位并行运算的单片处理器，所有元件都集成在一片 MOS 大规模集成电路芯片上。以 4004 为基础，再配以相应 RAM、ROM 和 I/O 接口芯片等，就构成了 MCS－4 微型计算机。

从 20 世纪 70 年代初至今仅 30 多年的时间，微处理器的发展经历了以下时期。

第一代从 1971 年开始，是 4 位微处理器和低档 8 位微处理器的时期。典型产品有 Intel 的 4004(4 位)、4040(8 位)和 8008(8 位)，其集成度为 2000 管/片，采用 P－MOS 工艺，10μm 光刻技术，时钟频率仅 1MHz，平均执行指令时间约为 20μs。

第二代包括 1974 年～1978 年 Intel 公司推出的 8080/8085、Zilog 公司推出的 Z80、Motorola 公司的 MC6800/6802 等。这一代 8 位高档的微处理器的特点是采用 N－MOS 工艺，集成度达 8400 管/片以上，时钟频率为 2MHz～4MHz，平均执行指令时间约为 1μs～2μs。这时的微处理器的设计和生产技术相当成熟，配套的各种器件也很齐备。以后的微处理器在提高集成度，提高功能和速度，增加外围电路的功能和种类上发展很快。

第三代以 16 位微处理器的出现为代表，时间为 1978 年。典型产品有 Intel 的 8086、Zilog 的 Z8000 和 Motorola 的 MC68000。它们采用了 H－MOS 工艺，集成度为 20000～60000 管/片，时钟频率为 4MHz～8MHz，平均执行指令时间为 0.5μs。

第四代微型计算机(1980 年～1992 年)以 32 位微型机为代表。典型的微处理器产品有 80386/80486、MC68020、Z80000 等。时钟频率为 16MHz～100MHz，集成度为 105 管/片。

1993 年 Intel 公司推出的 Pentium 微处理器，宣布了第五代微处理器的诞生。这种 Pentium 微处理器芯片采用了全新的体系结构。芯片的集成度高达 $5.0 \times 10^6 \sim 9.3 \times 10^6$ 管/片。时钟频率达 150MHz～300MHz。

1995 年 Pentium Ⅱ问世，1999 年 Pentium Ⅲ诞生。目前市场上的主流产品已由 PentiumⅣ取代了 PentiumⅢ。随着集成电路工艺和计算机科学技术的迅猛发展，更高性能的微处理器将不断推出，微型计算机的发展速度是不可估量的，微型计算机的应用越来越广泛。

第二节 计算机中的数和编码

一、计算机中的数制

(一)常用进位计数制

1. 十进制

十进制数人们最熟悉,但机器实现起来困难。

2. 二进制

二进制数由于每位只需两个状态"0"或"1",机器实现容易,因而二进制是数字系统唯一认识的代码,但二进制书写太长。

3. 八进制

因为 $2^3=8$,因而三位二进制数可用一位八进制数表示。

4. 十六进制

因为 $2^4=16$,因而四位二进制数可用一位十六进制数表示。

在计算机应用系统中,二进制主要用于机器内部的数据处理,八进制和十六进制主要用于书写程序,十进制主要用于运算最终结果的输出。

(二)数制转换

1. 非十进制数转换成十进制数

不同数制之间的转换方法有若干种,把非十进制数转换成十进制数采用按权展开相加法。具体步骤是:首先把非十进制数写成按权展开的多项式,然后按十进制数的计数规则求和。

2. 十进制数转换成二进制数

对于既有整数部分又有小数部分的十进制数转换成二进制数,首先要把整数部分和小数部分分开转换,再把两者的转换结果相加。具体方法介绍如下。

(1)整数转换

方法:连续除以2取余数,直到商为0,并按照和运算过程相反的顺序把各个余数排列起来,即为二进制整数。

(2)小数转换

方法:连续乘2取整数,并按照和运算过程相同的顺序把各个整数排列起来,即为二进制小数。

3. 二进制数转换成八进制数或十六进制数

二进制数转换成八进制数(或十六进制数)时,其整数部分和小数部分可以同时进行转换。其方法是:以二进制数的小数点为起点,分别向左、向右,每三位(或四位)分一组。对于小数部分,最低位一组不足三位(或四位)时,必须在有效位右边补0,使其足位。然后把每一组二进制数转换成八进制(或十六进制)数,并保持原顺序。对于整数部分,最高位一组不足位时,可在有效位的左边补0,也可以不补。

4. 八进制数或十六进制数转换成二进制数

八进制(或十六进制)数转换成二进制数时,只要把八进制(或十六进制)数的每一位数码分别转换成三位(或四位)的二进制数,并保持原排序即可。整数最高位一组左边的0,及小数最低位一组右边的0,可以省略。

二、符号数的表示

(一)机器数与真值

二进制数与十进制数一样有正有负。在计算机中,常把数的符号和数值部分一起编码后表示带符号数。常用的带符号数编码方法有原码、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,称前者为真值,后者为机器数,即数值连同符号数码"0"或"1"一起作为一个数称为机器数,而它的数值连同符号"+"或"-"称为机器数的真值。

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,因此只有8位、16位或32位机器数的最高位才是符号位。

1. 原码

如上所述,数值部分用该数的绝对值,符号部分用"0"表示正数,用"1"表示负数,这样表示数的方法称带符号数的原码表示法,所表示的数称为原码。

$y_1 = +127 = +1111111B$　　$[y_1]$原码 $= 01111111B$

$y_2 = -127 = -1111111B$　　$[y_2]$原码 $= 11111111B$

其中最高位为符号,后面7位是数值。用原码表示时,+127和-127的数值部分相同而符号位相反。

8位整数原码表示的数值范围为FFH~7FH,即-127~+127。原码数00H和80H的数值部分相同,符号位相反,它们分别为+0和-0。16位整数原码表示的数值范围为FFFFH~7FFFH,即-32767~+32767。原码数0000H和8000H的数值部分相同,符号位

相反，它们分别为 +0 和 -0。

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。为简化计算机的结构把减法运算转换为加法运算，因而引入了反码和补码。

2. 反码

正数的反码与原码相同；负数的反码为它的原码的数值部分按位取反，而符号位不变，即仍为“1”。

$y_1 = +127 = +1111111B$　　$[y_1]$反码 $= 01111111B$

$y_2 = -127 = -1111111B$　　$[y_2]$反码 $= 10000000B$

3. 补码

正数的补码与原码相同；负数的补码为它的原码的数值部分按位取反，最末位再加1，而符号位不变，即仍为“1”。

$y_1 = +127 = +1111111B$　　$[y_1]$补码 $= 01111111B$

$y_2 = -127 = -1111111B$　　$[y_2]$补码 $= 10000001B$

8 位补码数表示的数值范围为 80H ~ 7FH，即 -128 ~ +127。16 位补码表示的数值范围为 8000H ~ 7FFFH，即 -32768 ~ +32767。

（二）原码、反码、补码和真值的相互转换关系

前面我们已经讨论了符号数的真值表示和在机器中的表示，下面我们总结一下机器数和真值的相互转换关系。

正数的原码、反码、补码相同，即正号“0”加上绝对值的数值位表示。

负数的原码、反码、补码和真值的相互转换关系，如图 1.1 所示。

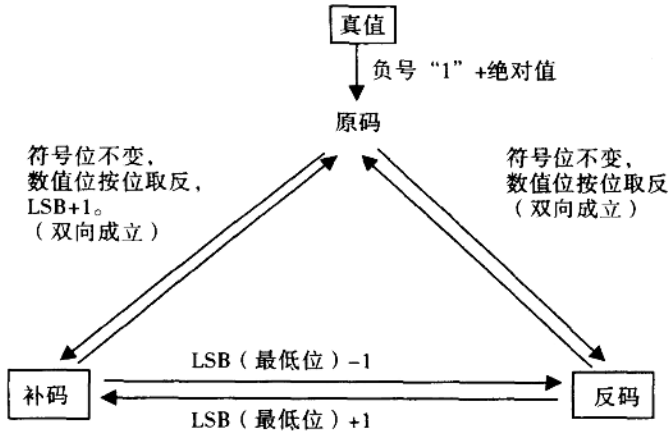

图 1.1　负数的原码、反码、补码和真值的相互转换关系

下面我们举例说明符号数的机器表示和真值的相互转换关系。以负数情况为例进行分析。

1. 已知原码，求补码

【例 1.1】已知某数 X 的原码为 10110100B，试求 X 的补码。

【解】由[X]原 =10110100B，X 为负数。求其补码表示时，符号位不变，数值部分按位取反，再在末位加 1。

```
  1 0 1 1 0 1 0 1    原码
  ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓
  1 1 0 0 1 0 1 0    符号位不变，数值位取反
+               1    +1
-----------------
  1 1 0 0 1 0 1 1    原码
```

∴ [X]补 = 11001011B

2. 已知补码，求原码

【例 1.2】已知某数 X 的补码为 11101110B，试求其原码。

【解】由[X]补 = 11101110B，知 X 为负数。求其原码表示时，符号位不变，数值部分按位求反后，再在末位加 1。

```
  1 1 1 0 1 1 1 0    原码
  ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓
  1 0 0 1 0 0 0 1    符号位不变，数值位取反
+               1    +1
-----------------
  1 0 0 1 0 0 1 0    原码
```

∴ [X]原码 = 10010010B

3. 求补

已知[X]补，求[-X]补的过程称为求补。所谓求补，就是将[X]补的所有位(包括符号位)一起逐位取反，然后在末位加 1，即可得到 -X 的补码，亦即[-X]补。不管 X 是正数还是负数，都应按此方法操作。

【例 1.3】试求 +97、-97 的补码。

【解】+97 = 1100001，于是[+97]补 = 01100001B

求[-97]补的方法是：

```
[97]补=0 1 1 0 0 0 0 1    [+97]补
       ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓
       1 0 0 1 1 1 1 0    逐位取反
+                     1    +1
-----------------------
       1 0 0 1 1 1 1 1    [-97]补
```

∴ [+97]补 = 01100001B，[-97]补 = 10011111B

4. 已知补码，求对应的十进制数。

【例 1.4】已知某数 X 的补码为 10101011B，试求其对应的十进制数。

【解】由[X]补 = 10101011B，知 X 为负数，故符号位不变，数值部分按“求反加 1”法得

到[X]原：

[X]原 = 11010101B，X = -1010101B，X = -85

三、二进制数的运算

（一）二进制数的加减运算

计算机把机器数均当作无符号数进行运算，即符号位也参与运算。运算的结果要根据运算结果的符号，运算有无进位（借位）和溢出等来判别。计算机中设置有这些标志位，标志位的值由运算结果自动设定。

1. 无符号数的运算

无符号数实际上是指参加运算的数均为正数，且全部数位都用于表示数值。n 位无符号二进制数能表示的数值范围为 0 ~ (2^n - 1)

两个无符号数相加，由于两个加数均为正数，因此其和也是正数。当和超过其位数所允许表示的数值范围时，就向更高位进位。如：

127 + 150 = 7FH + 96H

```
   0 1 1 1 1 1 1 1
+  1 0 0 1 0 1 1 0
-----------------------
1 0 0 0 1 0 1 0 1=115H=256+16+5=277
↑
└── 进位
```

两个无符号数相减

被减数大于或等于减数，无借位，结果为正；被减数小于减数，有借位，结果为负。如：

128 - 10 = 80H - 0AH

```
   1 0 0 0 0 0 0 0
-  0 0 0 0 1 0 1 0
-----------------------
   0 1 1 1 0 1 1 0=76H=112+6=118
```

反过来相减，即 10 - 128，运算过程如下：

```
   0 0 0 0 1 0 1 0
-  1 0 0 0 0 0 0 0
-----------------------
1 1 0 0 0 1 0 1 0=-01110110=-118
↑
└── 借位
```

由此可见，对无符号数进行减法运算，其结果的符号用借位标志位来判别：若 CF = 0，则无借位，结果一定为正；若 CF = 1，则有借位，结果一定为负，对 8 位数值位求补便可求得它的绝对值。

2. 符号数的加法运算和溢出判断

（1）补码的加减法运算

由于符号数在引入补码表示法以后，补码的减法运算可以转换为加法运算来进行。其运算法则为：