

21世纪重点大学规划教材



北京高等教育精品教材

BEIJING GAODENG JIAOYU JINGPIN JIAOCAI

余春暄 左国玉 等编著

# 80x86/Pentium

## 微机原理及接口技术 第3版



机械工业出版社  
CHINA MACHINE PRESS



附赠光盘

北京高等教育精品教材  
21 世纪重点大学规划教材

# 80x86/Pentium 微机原理 及接口技术

第3版

余春暄 左国玉 等编著

机械工业出版社

本书主要介绍了从 8086 到 Pentium 系列微处理器的结构、特点和相关技术, 寻址方式、指令系统及汇编语言程序设计, 以及微型计算机各组成部分的原理、常用接口技术及其应用。

本书以培养学生应用能力为主要目标, 强调掌握基本知识和基本技术, 以及分析问题和解决问题的方法, 在传统内容的基础上力求反映微型计算机及微处理器的新技术。为了配合教师课堂教学和学生课后学习, 本书配备了用 Authorware、PPT 和 PDF 等软件制作的全国多媒体课件大赛获奖的多媒体辅助教学课件, 利用视频和动画帮助读者理解所学内容。课件中还配有教学大纲和自测练习, 使读者明确整个学习内容, 了解学习效果。

本书既可作为高等院校计算机及相关专业的相关课程的教材, 又可供计算机硬件或软件开发人员参考。

本书配有授课电子课件和部分工具文件, 需要的教师可登录 [www.cmpedu.com](http://www.cmpedu.com) 免费注册, 审核通过后下载, 或联系编辑索取 (QQ: 2966938356, 电话: 010-88379739)。

## 图书在版编目 (CIP) 数据

80x86/Pentium 微机原理及接口技术/余春暄编著. —3 版. —北京: 机械工业出版社, 2015.4

21 世纪重点大学规划教材

ISBN 978-7-111-49627-4

I. ①8… II. ①余… III. ①微型计算机—理论—高等学校—教材②微型计算机—接口—高等学校—教材

中国版本图书馆 CIP 数据核字 (2015) 第 049161 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: 郝建伟 责任编辑: 郝建伟

责任校对: 张艳霞

责任印制: 李 洋

北京瑞德印刷有限公司印刷 (三河市胜利装订厂装订)

2015 年 6 月第 3 版·第 1 次印刷

184mm×260mm·21.25 印张·527 千字

0001—3000 册

标准书号: ISBN 978-7-111-49627-4

ISBN 978-7-89405-784-6

定价: 49.90 元 (含 1CD)

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

电话服务

网络服务

服务咨询热线: (010) 88379833

机工官网: [www.cmpbook.com](http://www.cmpbook.com)

读者购书热线: (010) 88379649

机工官博: [weibo.com/cmp1952](http://weibo.com/cmp1952)

教育服务网: [www.cmpedu.com](http://www.cmpedu.com)

封面无防伪标均为盗版

金书网: [www.golden-book.com](http://www.golden-book.com)

# 出版说明

随着我国信息化建设步伐的逐渐加快,对计算机及相关专业人才的要求越来越高,许多高校都在积极地进行专业教学改革的研究。

加强学科建设,提升科研能力,这是许多高等院校的发展思路。众多重点大学也是以此为基础,进行人才培养。重点大学拥有非常丰富的教学资源 and 一批高学历、高素质、高科研产出的教师队伍,通过多年的科研和教学积累,形成了完善的教学体系,探索出人才培养的新方法,搭建了一流的教学实践平台。同学科建设相匹配的专业教材的建设成为各院校学科建设的重要组成部分,许多教材成为学科建设中的优秀成果。

为了体现以重点建设推动整体发展的战略思想,将重点大学的一些优秀成果和资源与广大师生共同分享,机械工业出版社策划开发了“21世纪重点大学规划教材”。本套教材具有以下特点:

- 1) 由来自于重点大学、重点学科的知名教授、教师编写。
  - 2) 涵盖面较广,涉及计算机各学科领域。
  - 3) 符合高等院校相关学科的课程设置和培养目标,在同类教材中,具有一定的先进性和权威性。
  - 4) 注重教材理论性、科学性和实用性,为学生继续深造学习打下坚实的基础。
  - 5) 实现教材“立体化”建设,为主干课程配备了电子教案、素材和实验实训项目等内容。
- 欢迎广大读者特别是高校教师提出宝贵意见和建议,衷心感谢计算机教育工作者和广大读者的支持与帮助。

机械工业出版社

# 前 言

电子技术和计算机技术的迅猛发展，带来了人类生活、学习和科学研究许多领域的技术革命，使得现代社会和人类生活越来越离不开计算机，计算机知识和应用技能已成为人类知识经济的重要组成部分。无论是从事自动化、电子信息工程、通信工程、计算机应用、电气工程以及智能化仪器仪表等方面的理论研究，还是从事工程实践，都离不开计算机技术，因而“微机原理与接口技术”是电气信息类专业教学的学科基础课之一。

微型计算机原理与应用课程由于知识点多，初学者常感到课程难学、作业难做。本书是在参阅了当前国内外有关微型计算机的大量资料基础上，根据作者多年的教学实践和科学研究的经验编写的，通俗易懂、由浅入深、举一反三，既适合于高等院校计算机及相关专业学习有关课程的教学，又可以作为从事计算机硬件或软件开发与应用工作的工程技术人员的参考资料。本书的主要特点有：

1) 内容上注重结构清晰、重点突出、循序渐进、实例丰富，使初学计算机原理者容易接受。

2) 以目前最为普及的 Intel 80x86/Pentium 系列计算机系统作为背景，详细介绍了微型计算机的组成结构、工作原理、指令系统、接口技术等，为后续课程及计算机应用、开发打下良好的基础。

3) 为了配合教师课堂教学和学生课后学习，本教材配备了用 Authorware、PPT 和用 PDF 制作的三款多媒体辅助教学课件，利用众多动画帮助读者理解所学内容。课件中还配有教学大纲和自测练习，使读者明确整个学习内容，了解学习效果。本书还提供部分工具文件，教师可到 [www.cmpedu.com](http://www.cmpedu.com) 免费注册，审核通过后下载。

4) 注重基础，强调理论和实践相结合。重点介绍 8086/8088 处理器及外围接口技术的原理和应用方法，兼顾对计算机技术发展的展示，使读者了解计算机的发展历程，加深理解计算机系统的工作过程，适应计算机技术不断发展和应用不断升级的需求，掌握用计算机解决实际问题的方法。

5) 配套了“80x86 微机原理及接口技术——习题解答与实验指导”一书，从不同角度帮助读者理解和掌握理论教学中的原理和技术，也可作为题库素材。

教材共分为8章。第1章从介绍计算机的组成、微处理器的结构、计算机中数和编码的表示方法入手，建立计算机系统的整体概念，通过处理器执行程序过程的动画演示来了解微处理器的工作过程，之后简单介绍了微机系统中采用的先进技术。第2章以 Intel 80x86 / Pentium 系列芯片的基础 8086/8088 微处理器为切入点，重点介绍 8086/8088 微处理器的内部结构、内部寄存器、工作模式、引脚定义、存储器组织和系统组成，在此基础上对比介绍了 80x86 及 Pentium 系列微处理器的发展和特点。第3~4章介绍了 80x86 / Pentium 系列处理器的指令格式、寻址方式、指令系统、汇编语言程序设计以及调试手段，最后给出了汇编语言与 C/C++ 混合编程的方法。第5章介绍了计算机中常用的半导体存储器的分类与接口设计方法。第6~8章介绍了

计算机接口技术，对计算机接口概念进行了较详细的阐述，通过应用实例介绍了简单接口技术和可编程接口技术。

本书主要由余春暄、左国玉编写和统稿，参与编写的还有施远征、韦燕凤、李展鹏、李锋、彭靖漩、杨洋和吴文茂，邓军完成了PPT版的辅助教学软件。本书在编写过程中参考了大量文献，在参考文献中已尽量列出。但是仍有部分资料因原始出处不详而未被列出，在此深表歉意。本书在编写与出版过程中，得到了机械工业出版社的具体指导与帮助，在此一并表示衷心感谢。

由于编者水平有限，书中难免有错误和不妥之处，敬请读者批评指正。

出版说明

编者

42	处理器	1	1.1 概述
42	2.3.1 微处理器技术	1	1.1.1 微处理器的发展
47	2.3.2 80x86 微处理器	1	1.1.2 常用术语
51	2.3.3 Pentium 系列微处理器	2	1.1.3 正文中使用的符号
53	2.3.4 双微处理器	3	1.2 计算机中数与编码的表示方法
55	2.4 习题思考	3	1.2.1 二进制表示
57	第3章 80x86 及 Pentium 指令系统	3	1.2.2 二进制数表示
57	3.1 指令的结构	5	1.2.3 有符号数的表示
57	3.1.1 指令操作码信息	6	1.2.4 数据的表示
58	3.1.2 指令格式	8	1.3 微处理器结构和工作原理
60	3.2 寻址方式	9	1.3.1 微处理器
60	3.2.1 寻址方式的定义	9	1.3.2 微处理器
60	3.2.2 寻址方式分类	11	1.3.3 微处理器系统
61	3.2.3 微处理器寻址方式	12	1.3.4 微处理器中的指令流
66	3.2.4 目标地址寻址方式	13	1.4 微处理器中采用的先进技术
68	3.3 8086/8088 指令系统	14	1.4.1 流水技术
69	3.3.1 传送类指令	14	1.4.2 微处理器中总线技术
72	3.3.2 算术运算类指令	15	1.4.3 CISC 和 RISC
86	3.3.3 逻辑运算类指令	16	1.4.4 核心技术
88	3.3.4 移位类指令	16	1.5 习题思考
90	3.3.5 转移类指令	17	第2章 80x86 及 Pentium 微处理器
101	3.3.6 串操作类指令	19	2.1 Intel 系列微处理器概述
102	3.3.7 处理器类指令	19	2.2 8086/8088 微处理器
106	3.4 从 80286 到 Pentium 增加指令介绍	21	2.2.1 8086/8088 的寄存器
106	3.4.1 80286 的增强型指令	27	2.2.2 8086/8088 的工作寄存器
109	3.4.2 80286/80486 的增强型指令	31	2.2.3 8086/8088 的时序总线周期
110	3.4.3 Pentium 系列微处理器的增强型指令	32	2.2.4 8086/8088 的存储器组织
114	3.5 习题思考	37	2.2.5 8086/8088 的地址总线
118	第4章 汇编语言程序设计	38	2.2.6 8086/8088 的系统总线
118	4.1 汇编语言简介	38	2.3 80286 到 Pentium 系列微

# 目 录

## 出版说明

## 前言

第 1 章 计算机基础	1	处理器	42
1.1 概述	1	2.3.1 微处理器相关技术概述	42
1.1.1 微计算机的发展	1	2.3.2 80x86 微处理器	47
1.1.2 常用术语	2	2.3.3 Pentium 系列微处理器	51
1.1.3 正文中使用的符号	3	2.3.4 双核微处理器	53
1.2 计算机中数与编码的表示方法	3	2.4 习题与思考	55
1.2.1 进制表示	3	第 3 章 80x86 及 Pentium 指令系统	57
1.2.2 定点和浮点表示	5	3.1 指令的结构	57
1.2.3 有符号数的表示	6	3.1.1 指令提供的信息	57
1.2.4 编码的表示	8	3.1.2 指令格式	58
1.3 微型计算机结构及工作原理	9	3.2 寻址方式	60
1.3.1 微处理器	9	3.2.1 寻址方式的定义	60
1.3.2 微型计算机	11	3.2.2 寻址方式分类	60
1.3.3 微型计算机系统	12	3.2.3 数据型操作数寻址方式	61
1.3.4 计算机中的指令执行过程	13	3.2.4 目标地址寻址方式	66
1.4 微机系统中采用的先进技术	14	3.3 8086/8088 指令系统	68
1.4.1 流水线技术	14	3.3.1 传送类指令	69
1.4.2 高速缓冲存储技术	15	3.3.2 算术运算类指令	75
1.4.3 CISC 和 RISC	16	3.3.3 逻辑运算类指令	86
1.4.4 多核心技术	16	3.3.4 位移类指令	88
1.5 习题与思考	17	3.3.5 转移类指令	90
第 2 章 80x86 及 Pentium 微处理器	19	3.3.6 串操作类指令	101
2.1 Intel 系列微处理器概述	19	3.3.7 处理器类指令	105
2.2 8086/8088 微处理器	21	3.4 从 80286 到 Pentium 增加	
2.2.1 8086/8088 的编程结构	21	指令介绍	106
2.2.2 8086/8088 的工作模式与引脚定义	27	3.4.1 80286 的增强与增加指令	106
2.2.3 8086/8088 的时序与总线周期	31	3.4.2 80386/80486 的增强与增加指令	109
2.2.4 8086/8088 的存储器组织	35	3.4.3 Pentium 系列处理器的增加指令	110
2.2.5 8086/8088 的堆栈组织	37	3.5 习题与思考	114
2.2.6 8086/8088 的系统组织	38	第 4 章 汇编语言程序设计	118
2.3 80286 到 Pentium 系列微		4.1 编程语言概述	118

4.1.1	计算机语言的分类	118	5.1	计算机存储器概述	178
4.1.2	MASM 汇编语言	119	5.1.1	微型计算机中存储器的分类	178
4.2	伪指令	121	5.1.2	半导体存储器的分类	179
4.2.1	汇编语言中数、符号、表达式的描述规范	122	5.1.3	半导体存储器的性能指标	181
4.2.2	处理器定义伪指令	125	5.1.4	计算机系统中常见的半导体存储器	181
4.2.3	模式定义伪指令	126	5.1.5	存储卡技术	183
4.2.4	段定义伪指令	126	5.2	半导体存储器结构与工作原理	185
4.2.5	数据定义伪指令	130	5.2.1	半导体存储器的基本组成	186
4.2.6	符号定义伪指令	132	5.2.2	随机存储器 (RAM)	187
4.2.7	类型定义伪指令	133	5.2.3	只读存储器 (ROM)	193
4.2.8	过程定义伪指令	134	5.2.4	非易失读/写存储器	195
4.2.9	程序计数器与定位伪指令	135	5.3	半导体存储器接口设计	200
4.2.10	条件汇编伪指令	135	5.3.1	存储芯片的选择	200
4.2.11	记录与结构伪指令	136	5.3.2	存储器的地址分配	201
4.2.12	模块定义伪指令	140	5.3.3	存储器的地址译码	202
4.3	宏指令	141	5.3.4	存储器与 CPU 的信号连接	205
4.3.1	宏指令定义	141	5.3.5	存储器接口设计举例	206
4.3.2	宏指令的应用	142	5.4	80x86 存储器技术	210
4.3.3	宏指令与子程序的区别	143	5.4.1	虚拟存储器简介	210
4.4	BIOS 和 DOS 的功能调用	143	5.4.2	80x86 中的 ROM 重复和影子 RAM	211
4.4.1	BIOS 调用	144	5.5	习题与思考	211
4.4.2	DOS 软中断	144	第 6 章	微型计算机接口技术	214
4.4.3	DOS 系统功能调用	144	6.1	微型计算机接口结构与功能	214
4.5	汇编语言程序设计与调试	146	6.1.1	接口的基本结构	214
4.5.1	汇编语言程序设计步骤	146	6.1.2	接口的功能	215
4.5.2	MASM 汇编语言的调试方法	147	6.1.3	80x86 PC 系统中的 I/O 地址映射	217
4.6	汇编语言程序设计基本方法	148	6.2	微处理器与外设数据传输控制方式	218
4.6.1	顺序结构程序	148	6.2.1	直接程序传输	218
4.6.2	分支结构程序	151	6.2.2	查询程序传输	218
4.6.3	循环结构程序	156	6.2.3	中断传输	219
4.6.4	子程序调用结构程序	161	6.2.4	DMA 传输	219
4.6.5	综合应用程序设计举例	168	6.3	微型计算机的中断系统	221
4.6.6	80x86 应用程序设计举例	169	6.3.1	中断控制方式的优点	221
4.7	汇编语言与 C/C++ 语言的混合编程	170	6.3.2	与中断有关的术语	221
4.7.1	内嵌模块方法	171	6.3.3	中断过程	222
4.7.2	外调模块方法	171	6.3.4	80x86 中断系统	226
4.8	习题与思考	174	6.4	微型计算机功能扩展总线	
第 5 章	半导体存储器及其接口技术	178			



和接口标准	227	8.2 并行通信接口 8255A	266
6.4.1 一些常用的总线术语	227	8.2.1 8255A 内部结构与引脚定义	266
6.4.2 总线的分类	228	8.2.2 8255A 的工作方式及其初始化设置	268
6.4.3 80x86 系列微机中常用的总线 和接口标准	228	8.2.3 8255A 各工作方式的功能特点 说明	269
6.5 习题与思考	235	8.2.4 8255A 应用举例	273
<b>第 7 章 简单接口电路设计</b>	<b>236</b>	8.3 串行通信接口 8250/8251	280
7.1 接口电路概述	236	8.3.1 串行通信接口技术的概念	281
7.1.1 数据锁存器	236	8.3.2 可编程异步通信接口 8250/16450	285
7.1.2 数据缓冲器	236	8.3.3 可编程串行通信接口芯片 Intel 8251A	294
7.2 开关量输出接口设计	237	8.4 可编程中断控制器 8259A	302
7.2.1 单个开关量输出接口	237	8.4.1 8259A 概述	302
7.2.2 多个开关量输出接口	238	8.4.2 8259A 的引脚特性	303
7.2.3 数码显示接口	238	8.4.3 8259A 内部结构	303
7.3 开关量输入接口设计	241	8.4.4 8259A 的初始化编程	305
7.3.1 单个开关量输入接口	241	8.4.5 8259A 的工作编程	308
7.3.2 多个开关量输入接口	241	8.4.6 8259A 应用举例	309
7.3.3 键盘接口	241	8.5 DMA 控制器 8237A	315
7.4 D/A 转换接口	244	8.5.1 8237A 的功能及引脚特性	315
7.4.1 D/A 转换原理	245	8.5.2 8237A 内部寄存器及读写操作	317
7.4.2 D/A 转换器技术参数	246	8.5.3 8237A 初始化编程	319
7.4.3 DAC 0832 介绍	246	8.5.4 8237A 应用举例	319
7.5 A/D 转换接口	249	8.6 习题与思考	321
7.5.1 A/D 转换原理	249	<b>附录</b>	<b>326</b>
7.5.2 A/D 转换器的主要技术指标	252	附录 A 7 位 ASCII 码编码表	326
7.5.3 ADC 0809 介绍	253	附录 B DEBUG 的常用命令	327
7.6 A/D 和 D/A 转换接口应 注意的问题	254	附录 C 多媒体辅助教学软件说明	328
7.7 习题与思考	256	附录 D 逻辑符号对照表	329
<b>第 8 章 可编程接口技术</b>	<b>257</b>	附录 E 8086/8088 常用指令 简单列表	330
8.1 可编程计数器 8253/8254	257	附录 F 常用汇编语言伪指令 简单列表	331
8.1.1 8253/8254 外部特点与功能	257	附录 G TD 的常用命令	331
8.1.2 8253/8254 内部结构与工作原理	258	<b>参考文献</b>	<b>332</b>
8.1.3 8253/8254 的控制字与初始化 编程	259		
8.1.4 8253/8254 的工作方式	260		
8.1.5 8253/8254 应用举例	263		

# 第1章 计算机基础

计算机技术是 20 世纪最杰出的科技成果之一，它极大地改变了人类社会的生活、学习和工作的方式。本章介绍计算机的发展背景、体系结构、常用术语以及计算机中数和编码的表示方法，使读者了解计算机系统的整体概念和计算机的基本工作过程，为后续学习打下基础。

## 1.1 概述

### 1.1.1 微计算机的发展

“计算”是人类生活中最重要的活动之一。随着人类社会的发展与进步，计算量越来越大，而且越来越复杂，促使人们不断推出各种各样的计算工具，如我国唐宋时期出现了算盘，后来科学家又发明了计算尺、机械式计算器等。随着电子技术的发展，在 1946 年，世界上第一台电子数字积分式计算机（Electronic Numerical Integrator And Calculator, ENIAC）诞生了。著名数学家冯·诺伊曼（von Neumann）在 ENIAC 计算机的设计制造期间，首次提出了“存储程序”的概念，从那时开始，这个概念一直沿用至今。因此，人们把按照这一概念制造的计算机称作冯·诺伊曼计算机（von Neumann Machine）。冯·诺伊曼计算机的核心是：

- 1) 指令和数据用二进制数表示。
- 2) 程序预存在存储器中，在执行时会将指令自动地逐条取出并分析执行。
- 3) 计算机的硬件由运算器、控制器、存储器、输入设备和输出设备 5 部分组成。

计算机的发展可分为：

第一代（1946 年~1958 年）真空管计算机。真空管体积大，并且非常耗电，例如 ENIAC 的功率为 130 000W。

第二代（1958 年~1964 年）晶体管计算机。

第三代（1964 年~1971 年）集成电路计算机。

第四代（1971 年至今）超大规模集成电路计算机。

目前正在向第五代人工智能计算机方向突破，其主要目标是希望实现更高程度上模拟人脑的思维功能。

现在，人们广泛使用的计算机是第四代计算机，它的发展是以微处理器的发展为基础的。微处理器的更新速度很快，几乎每两年集成度翻一番，每 2~4 年更新换代一次。可以看到：

1971 年~1973 年，第一代微处理器主要为 4 位或低档 8 位微处理器，其指令系统比较简单，运算能力差、速度慢。

1974 年~1978 年，第二代中高档 8 位微处理器，它比第一代微处理器有了较多的改进，集成度提高 1~4 倍，运算速度提高 10~15 倍，指令系统相对比较完善，已具有典型的计算机体系结构以及中断、存储器直接读取（DMA）功能。

1978 年~1981 年推出第三代 16 位微处理器，从各项性能指标看，第三代微处理器比第二代微处理器的性能提高了很多，已达到或超过原来的中低档小型机的水平。

1985年, Intel公司推出了32位微处理芯片80386。80386有两种结构: 80386 SX和80386 DX。

1990年, Intel公司在80386的基础上研制出新一代32位微处理芯片80486, 其性能比80386大大提高, 并采用了精简指令集计算(Reduced Instruction Set Computing, RISC)设计思想。

1993年3月, Intel公司推出了第五代微处理芯片Pentium(简称P5), 它的外部数据总线为64位, 内部总线为32位, 工作频率为66MHz。次年, Intel公司又推出了第二代Pentium, 在体系结构上采用了RISC技术, 可以说它是CISC和RISC技术相结合的产物。

1995年2月, Intel公司发布代号为Pentium Pro(简称P6)的新一代微处理器产品。

1998年到1999年, Intel公司又推出了Pentium的改进型, 即Pentium II和Pentium III。

2001年底, Intel公司又推出了代号为Northwood的Pentium 4, 很快成为主流高端32位CPU市场的佼佼者, 特别是在多媒体应用领域, 更具有突出的表现。同时, 与Intel公司进行竞争的AMD公司, 也不断推出新型微处理器。

在不断完善32位CPU系列的同时, Intel公司和AMD公司在开发第7代CPU即64位CPU方面展开了激烈的竞争, 并采取了不同的策略。Intel公司在开发64位的Itanium(安腾)时, 放弃一直沿用的x86架构, 而在IA-64架构的体系中采用了所谓的显式并行指令计算(Explicitly Parallel Instruction Computing, EPIC)核心技术, 保持了技术上的优势。而AMD公司在其开发64位CPU K8 SledgeHammer(大锤)时, 则采取了更为平滑的过渡方式, 尽管它在运行64位软件时速度不及Intel公司的Itanium, 但由于它注重增强同IA-32指令的兼容性, 使其在执行IA-32软件时速度明显高于Itanium。

## 1.1.2 常用术语

1) 位(bit): 是计算机所能表示的最小的数据单位。每一位只能有两种状态: “0”或“1”。bit是Binary Digit的缩写。

2) 字节(Byte): 一个8位二进制数称为一个字节, 在计算机中的基本存储单元内容用字节表示。

3) 字(Word): 是微处理器内部数据传输、处理的基本单位。目前PC常将2个字节定义为一个字, 即一个字为16位二进制数。一个双字为32位二进制数。

4) 指令: 用二进制代码组成, 规定微处理器进行某种操作的命令。即由“0”和“1”编码组成的代码, 微处理器每一个操作均对应一组唯一的编码。

5) 指令系统: 是指一台计算机所能识别的全部指令。不同类型的微处理器, 其指令系统不同, 一般不能兼容。如IBM机、苹果机、各种单片机的指令系统均不相同。

6) 程序: 指令的有序集合。即为使计算机完成某种工作而编制的一系列指令。计算机将按照这一系列指令一步一步地工作。而这一系列指令就组成程序。

7) 存储器: 用于存储数据和程序。而数据和程序均用二进制编码表示。我们把存储器看成很多的单元, 每个基本存储单元存储8位二进制编码(称为一个字节), 并且均有自己的编号, 这个编号称为该存储单元的地址。

8) 存储容量单位:

1B=8bit(称为一个字节)

1KB=1024B=1024×8bit

1MB=1024KB=1024×1024B

1GB=1024MB=1024×1024×1024B

1TB=1024GB=1024×1024×1024×1024B

9) 地址: 指存储单元或 I/O 接口的编号。每个存储单元或 I/O 接口均有自己的唯一的物理编号, 即称为物理地址。

### 1.1.3 正文中使用的符号

R	寄存器	M	存储器
R8	8 位寄存器 (字节寄存器)	M8	8 位存储器 (字节单元)
R16	16 位通用寄存器 (字寄存器)	M16	16 位存储器 (字单元)
R32	32 位通用寄存器 (双字寄存器)	M32	32 位存储器 (双字单元)
src	源操作数	mem16	16 位存储器单元的逻辑地址
dest	目标操作数	mem32	32 位存储器单元的逻辑地址
n <sub>8</sub>	8 位立即数	EA	有效地址
n <sub>16</sub>	16 位立即数	disp	位移量
n <sub>32</sub>	32 位立即数	disp8	位移量为 8 位二进制数补码
n <sub>64</sub>	64 位立即数	disp16	位移量为 16 位二进制数补码

## 1.2 计算机中数与编码的表示方法

计算机的工作就是处理数据或编码, 那么, 计算机是如何描述数和编码的? 根据冯·诺伊曼计算机的概念, 计算机中采用二进制数表达。因此, 计算机中无论是数还是编码均用二进制数描述。但是为了描述方便, 在编程中常采用不同的进制来描述数据。例如, 为了书写方便且不易出错, 常用十六进制数, 为顾及日常生活的习惯常使用十进制数 (BCD 码), 在大型机、巨型机中还使用八进制数等。本书中常用的数制有: 二进制数、十进制数、十六进制数。了解二、十、十六进制数的表达方式及相互转换的方法是非常重要的。除此之外, 还应了解计算机中如何描述有符号数和编码。

### 1.2.1 进制表示

数制即为表示一个数的格式, 目前均以进位制来描述, 即每位的数值为该位的数字乘以该位的权值, 每位之间均为一个进位, 即

$$N_r = d_{n-1} \times r^{n-1} + d_{n-2} \times r^{n-2} + \dots + d_1 \times r^1 + d_0 \times r^0 + d_{-1} \times r^{-1} + \dots + d_{-m} \times r^{-m}$$

其中,  $r$  为进位值;  $n$  为整数位数;  $m$  为小数位数;  $d_i$  为第  $i$  位的数字 ( $r$  个数字中之一)。

1) 十进制数, 即  $r=10$ , 为逢十进一, 借一当十, 需要 10 个数字符号 (0~9), 第  $i$  位以  $10^i$  为权, 如  $N_{10} = (683.74)_{10} = 683.74D = 6 \times 10^2 + 8 \times 10^1 + 3 \times 10^0 + 7 \times 10^{-1} + 4 \times 10^{-2} = 683.74$ 。

2) 二进制数, 即  $r=2$ , 为逢二进一, 借一当二, 需要两个数字符号 0 和 1, 第  $i$  位以  $2^i$  为权, 如  $N_2 = (10001110.11)_2 = 10001110.11B = 1 \times 2^7 + 0 \times 2^6 + 0 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} = (571.75)_{10}$ 。

3) 十六进制数, 即  $r=16$ , 为逢十六进一, 借一当十六, 需要 16 个数字符号 0~9 及 A~F, 第  $i$  位以  $16^i$  为权, 如  $N_{16} = (5A2.F1)_{16} = 5A2.F1H = 5 \times 16^2 + A \times 16^1 + 2 \times 16^0 + F \times 16^{-1} + 1 \times 16^{-2} = (1442.941)_{10}$ 。

4) 各进制之间相互转换规则：一个数可以用不同的进制表示，各进制间转换过程中，整数部分和小数部分的处理方法不同，其转换规则为：

$$\left. \begin{array}{l} \text{十进制数} \rightarrow \text{二进制:} \\ \text{十进制数} \rightarrow \text{十六进制:} \\ \left. \begin{array}{l} \text{二进制数} \rightarrow \text{十进制} \\ \text{十六进制数} \rightarrow \text{十进制} \end{array} \right\} \end{array} \right\} \begin{array}{l} \text{整数部分为将数据N的整数部分连续除2取余, 直至商为0} \\ \text{小数部分为将数据N的小数部分连续乘2取整, 直至满足精度要求} \\ \text{整数部分为将数据N的整数部分连续除16取余, 直至商为0} \\ \text{小数部分为将数据N的小数部分连续乘16取整, 直至满足精度要求} \end{array}$$

按权展开, 即:  $N = \sum_{i=-n}^m d_i \cdot r^i$ 。

其中,  $n$  为整数位数;  $m$  为小数位数;  $r$  为进制值 (如十六进制数, 则  $r=16$ );  $d_i$  为第  $i$  位上的数字 (如十六进制数  $\rightarrow$  十进制数时,  $d_i$  为 0 到 15 中的某个数字)。

【例 1-1】 将十进制数 87.65 用 8 位二进制整数和 4 位二进制小数表示。

解： 整数部分：

小数部分：

2   87	余数	2   0.65	整数	
2   43	1	x 2	1	高位
2   21	1	x 2	0	
2   10	1	x 2	1	
2   5	0	x 2	0	低位
2   2	1	x 2	1	
2   1	0	x 2	0	
0	1	x 2	0	

所以  $87.65D = 01010111.1010B$ , 注意位数不足时, 要补 0, 如加粗的 0。

【例 1-2】 将十进制数 3587.658 转换成 4 位十六进制整数和 2 位十六进制小数。

解： 整数部分：

小数部分：

16   3587	余数	x) 0.658	整数	
16   224	3	x) 16	A	高位
16   14	0	+ 3168	8	低位
0	E	+ 528	8	

即  $3587.658D = 0E03.A8H$

【例 1-3】 将十六进制数 57.AH 转换成十进制数。

解：  $57.AH = 5 \times 16^1 + 7 \times 16^0 + A \times 16^{-1} = 80 + 7 + 0.625 = 87.625$

注意：在汇编语言程序中十进制数的标记 D 可以省略不写，但十六进制数的标记 H 和二

进制数的标记 B 不能省略。

## 1.2.2 定点和浮点表示

在实际应用中，要处理的数据多数是带有小数点的，而整数是小数的特殊情况。然而计算机只能识别 0、1 二进制数，不会识别小数点。对具有小数部分的数，在计算机中有两种表示方法：定点数和浮点数。

1) 定点数：即小数点固定的数。或定义为纯整数，或定义为纯小数，如图 1-1 所示。

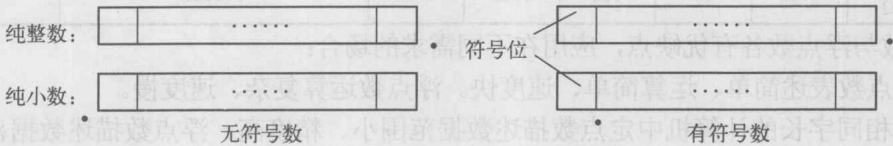


图 1-1 定点数格式示意

2) 浮点数：又称为实数，即小数点浮动的数，此类数实际上是用指数形式表示，如  $\pm L \cdot 2^{\pm C}$ ，其中 C 称为阶数（即指数），其位数多少决定数据表示的范围大小；L 称为尾数，其位数多少决定数据的精度。用定点数表示浮点数的格式如图 1-2 所示。阶符和尾符分别为阶数和尾数的符号，0 表示“+”号，1 表示“-”号。

【例 1-4】用国际标准 (IEEE-754) 的 32 位浮点数（单精度浮点数）表示 876.5 和 -0.125。

分析：目前，大多数编译器都支持国际标准 (IEEE-754) 的格式。其 32 位浮点数的格式要求为，尾数占 24 位，其中 23 位 ( $D_{22} \sim D_0$ ) 用绝对值表示有效数据，1 位 ( $D_{31}$ ) 为符号位，阶码占 8 位 ( $D_{30} \sim D_{23}$ )，如图 1-3 所示。

阶符	阶数	尾符	尾数
----	----	----	----

图 1-2 用定点数表示浮点数的格式

31	30	23	22	0
符号	阶码 P (8 位)	尾数 L (23 位)		

图 1-3 国际标准 (IEEE-754) 的 32 位浮点数格式

如果按照尾数归一化格式要求，其中二进制数尾数中整数部分一定为 1，故尾数只写小数部分，则有  $0.5 \leq |L| < 1$ ， $P = 7FH \pm$  阶数（阶符为“+”则加，阶符为“-”则减）。所以有：

$$876.5 = 1101101100.100B = 1.10110110010000000000000B \times 2^{1001B}$$

$$\text{尾数 } L = 10110110010000000000000B$$

$$\text{阶码 } P = 0111\ 1111B + 1001B = 10001000B = 88H$$

$$876.5 \text{ 的浮点数表示为 } = 0\ 10001000\ 10110110010000000000000B = 445B2000H$$

$$-0.125 = -0.0010000B = -1.00000000000000000000000B \times 2^{-11B}$$

$$\text{尾数 } L = 00000000000000000000000B$$

$$\text{阶码 } P = 0111\ 1111B - 11B = 01111100B = 7CH$$

$$-0.125 \text{ 的浮点数表示为 } = 1\ 01111100\ 00000000000000000000000B = BE000000H$$

从例 1-4 可以看出，用浮点表示数据比较复杂，运算也复杂（需要阶数对齐）。一般由专用的浮点运算部件（FPU）实现浮点运算，Pentium 处理器开始将浮点运算部件（FPU）集成在微处理器内部，用专门的浮点运算指令集实现浮点数据处理操作。表 1-1 列出了 IEEE754 三种浮点数的格式参数。双精度浮点数的阶码  $P = 3FFH \pm$  阶数，长双精度浮点数的阶码  $P = 3FFFH \pm$  阶数。

表 1-1 几种浮点数据的特点

数据类型	总位数	字节数	符号位	阶码位数	尾数位数	偏移值 (十六进制数)	偏移值 (十进制数)
单精度浮点数 (float)	32位	4	1位	8位	23位	7FH	+127
双精度浮点数 (double)	64位	8	1位	11位	52位	3FFH	+1023
长双精度浮点数 (long double)	80位	10	1位	15位	64位	3FFFH	+16383

定点数与浮点数各有优缺点，应用在不同需求的场合：

- 1) 定点数表述简单、运算简单、速度快。浮点数运算复杂、速度慢。
- 2) 在相同字长的计算机中定点数描述数据范围小、精度高，浮点数描述数据范围大、精度低。

### 1.2.3 有符号数的表示

计算机也可以处理有符号数，那么计算机是如何描述、处理有符号数的？计算机表示有符号数是将符号数值化，即将二进制数的最高位定义为符号位，用“0”表示“+”号，用“1”表示“-”号；无符号数是有符号数的特例，可以看成正数。

为了计算方便，引入了原码、反码、补码的概念。同一数的原码、反码和补码，是从不同的角度表示这个数。

#### 1. 原码

原码是最高位为符号位（0 表示正数，1 表示负数），其余位均为数值位的二进制数，所表示的数据范围为： $1-2^{n-1} \sim 2^{n-1}-1$ 。如 8 位二进制数（ $n=8$ ），所表示的数据范围为： $1-2^{8-1} \sim 2^{8-1}-1$ ，即  $-127 \sim +127$ 。

例如： $[+56]_{原} = 00111000B = 38H$        $[-56]_{原} = 10111000B = B8H$

0 的原码有两种表示：

$[+0]_{原} = 00000000B = 00H$        $[-0]_{原} = 10000000B = 80H$

#### 2. 反码

正数的反码与其原码相同。负数的反码等于原码中符号位不变，其他各位取反。反码表示的数据范围为： $1-2^{n-1} \sim 2^{n-1}-1$ 。如十六位二进制数（ $n=16$ ），所表示的数据范围为： $1-2^{16-1} \sim 2^{16-1}-1$ ，即  $-32767 \sim +32767$ 。

例如： $[+56]_{反} = [+56]_{原} = 00111000B = 38H$        $[-56]_{反} = 11000111B = C7H$

0 的反码有两种表示：

$[+0]_{反} = [+0]_{原} = 00000000B = 00H$        $[-0]_{反} = 11111111B = FFH$

#### 3. 补码

正数的补码与其原码相同。负数的补码等于原码中符号位不变，其他各位取反后再加一。所表示的数据范围为： $-2^{n-1} \sim 2^{n-1}-1$ 。如 8 位二进制数（ $n=8$ ），所表示的数据范围为： $-2^{8-1} \sim 2^{8-1}-1$ ，即  $-128 \sim +127$ 。

例如： $[+56]_{补} = [+56]_{原} = 00111000B = 38H$        $[-56]_{补} = 11001000B = C8H$

而 0 的补码只有一种表示：

$[+0]_{补} = [+0]_{原} = 00000000B = 00H$        $[-0]_{补} = 11111111B + 1 = 00H$

#### 4. 运算规则

$$[X]_{\text{原}} = [[X]_{\text{补}}]_{\text{补}}$$

$$[X]_{\text{原}} = [[X]_{\text{反}}]_{\text{反}}$$

$$[X \pm Y]_{\text{补}} = [X]_{\text{补}} \pm [Y]_{\text{补}}$$

$$[X \pm Y]_{\text{补}} = [X]_{\text{补}} + [\pm Y]_{\text{补}}$$

【例 1-5】 写出-111 和+97 的原码、反码和补码。

$$[-111]_{\text{原}} = 11101111\text{B} \quad [-111]_{\text{反}} = 10010000\text{B} \quad [-111]_{\text{补}} = 10010001\text{B}$$

$$[+97]_{\text{原}} = [+97]_{\text{反}} = [+97]_{\text{补}} = 01100001\text{B}$$

【例 1-6】 C2H 是某十进制数的补码，请问该数真值是多少？

$$X = [[X]_{\text{补}}]_{\text{补}} = [C2H]_{\text{补}} = [11000010\text{B}]_{\text{补}} = 10111110\text{B} = -62$$

【例 1-7】  $[X]_{\text{补}} = 2\text{AH}$ ， $[Y]_{\text{补}} = \text{B6H}$ ，试问  $X+Y=?$

方法一：

$$X+Y = [[X+Y]_{\text{补}}]_{\text{补}} = [[X]_{\text{补}} + [Y]_{\text{补}}]_{\text{补}} = [2\text{AH} + \text{B6H}]_{\text{补}} = [\text{E0H}]_{\text{补}} = 10100000\text{B} = -32$$

方法二：

$$X+Y = [[X]_{\text{补}}]_{\text{补}} + [[Y]_{\text{补}}]_{\text{补}} = [2\text{AH}]_{\text{补}} + [\text{B6H}]_{\text{补}} = 2\text{AH} + \text{CAH} = 42 + (-74) = -32$$

注意以下几点：

1) 互补的概念不是计算机中特有的，日常生活中也存在。例如，在时间上 11 点和差 1 小时 12 点表示的是同一个时刻。也就是说，11 和-1 在 12 进制时间表示上是互补的。

2)  $[\text{正数}]_{\text{原}} = [\text{正数}]_{\text{反}} = [\text{正数}]_{\text{补}}$ ， $[\text{负数}]_{\text{原}} \neq [\text{负数}]_{\text{反}} \neq [\text{负数}]_{\text{补}}$ 。

3) 采用补码运算的计算机称为补码机，目前我们见到的计算机均为补码机。

4) 在补码运算时，要注意溢出问题，它与进位不同。所谓溢出就是运算的结果超出了所能表示的范围，使得数据侵占了符号位。判断溢出的方法有如下两种。

- 双进位法：2 个进位位分别为次高位向最高位的进位和最高位向进位位的进位。如果两个进位均有或均无，则无溢出。如果两个进位中 1 个有进位而另 1 个无进位，则一定有溢出。
- 符号判断法：同号相减无溢出，同号相加时结果符号与加数符号相反有溢出，相同则无溢出。异号相加无溢出，异号相减时结果符号与减数符号相同有溢出，相反则无溢出。

【例 1-8】 用 8 位二进制补码完成  $(-23)+(-123)$  的运算。

	$[-23]_{\text{补}} = [10010111\text{B}]_{\text{补}} = 11101001\text{B}$
+)	$[-123]_{\text{补}} = [11111011\text{B}]_{\text{补}} = 10000101\text{B}$
	<hr style="border: 0.5px solid black;"/>
	$[+110]_{\text{补}} <= [01101110\text{B}]_{\text{补}} <= 01101110\text{B}$

从上面运算式中可以看出，次高位向最高位无进位，而最高位向进位位有进位，所以运算结果有溢出。从另一个角度来看，两个负数相加，结果为正数，其符号与减数的符号相反，所以运算结果有溢出。也就是  $(-23)+(-123) = -146 \neq +110$ ，运算结果不正确，这是因为运算结果有溢出，也就是运算结果的数据位超出了所能表示的范围，侵占了符号位。

正确的解决方法是扩大数据位数，即用 16 位二进制补码完成  $(-23)+(-123)$  的运算。

	$[-23]_{\text{补}} = [10000000\ 00010111\text{B}]_{\text{补}} = 11111111\ 11101001\text{B}$
+)	$[-123]_{\text{补}} = [10000000\ 01111011\text{B}]_{\text{补}} = 11111111\ 10000101\text{B}$
	<hr style="border: 0.5px solid black;"/>
	$[-146]_{\text{补}} <= [10000000\ 10010010\text{B}]_{\text{补}} <= 11111111\ 01101110\text{B}$



上式中次高位向最高位有进位，最高位向进位位也有进位，所以运算结果没有溢出。

【例 1-9】 用 8 位二进制补码完成 25-100 的运算。

$$\begin{array}{r}
 [+25]_{\text{补}} \Rightarrow [00011001\text{B}]_{\text{补}} \Rightarrow 00011001\text{B} \\
 -) [+100]_{\text{补}} \Rightarrow [01100100\text{B}]_{\text{补}} \Rightarrow 01100100\text{B} \\
 \hline
 [-75]_{\text{补}} \Leftarrow [11001011\text{B}]_{\text{补}} \Leftarrow 10110101\text{B}
 \end{array}$$

因为两个正数相减，所以运算结果一定无溢出。另外，从上面运算式中可以看出，次高位向最高位有借位，最高位向进位位也有借位，用双进位位判断溢出的方法，结果仍为无溢出。验证： $(+25) - (+100) = -75$ ，运算结果正确。

值得注意的是，对于有符号数运算一定要用补码运算，而不能用原码运算。为什么？请读者思考。

## 1.2.4 编码的表示

计算机除了可以处理数据，还可处理字符、图形，用特定的编码表示符号。编码种类有很多，如电报码、格雷码、BCD 码、ASCII 码、汉字编码等。在这里只介绍常用的 BCD 码、ASCII 码和汉字编码。

### 1. 二进制数编码的十进制数（BCD 码，Binary Coded Decimal）

计算机只能识别和处理二进制数，而日常生活中更为常用的是十进制数。计算机中可用二进制数编码表示十进制数称作 BCD 码，也就是用四位二进制数表示一位十进制数，它主要有两种表示形式：

1) 非压缩型 BCD 码：用一个字节表示一位十进制数，高四位清零。

2) 压缩型 BCD 码：用一个字节表示两位十进制数。

注意：

1) 计算机并不能真正识别 BCD 码，只是程序员明白所要处理数据的性质。

2) 有些指令对 BCD 码的表示形式有要求，如 80x86 的十进制调整指令 AAA、AAS、AAM 和 AAD 只对非压缩型 BCD 码进行操作，而 DAA 和 DAS，只对压缩型 BCD 码进行操作（参见第 3 章）。

表 1-2 给出了十进制数与非压缩型 BCD 码和压缩型 BCD 码的对照。

表 1-2 十进制数与非压缩型和压缩型 BCD 码的对照表

十进制数	非压缩型BCD码 (B)	压缩型BCD码 (B)
12	0000 0001 0000 0010	0001 0010
285	0000 0010 0000 1000 0000 0101	0000 0010 1000 0101
1470	0000 0001 0000 0100 0000 0111 0000 0000	0001 0100 0111 0000

## 2. ASCII 码

计算机中字符也必须采用二进制编码形式表示。编码形式有多种，目前广泛采用美国信息交换标准代码，即 ASCII (American Standard Code for Information Interchange) 码，也就是用 7 位二进制数对字符进行编码，附录 A 给出了 ASCII 码与字符对照表。其中共有  $2^7=128$  个字符，包括 94 个可视字符和 34 个控制字符。如数字符“0”~“9”对应的编码为 30H~39H，小写英文字母“a”~“z”对应的编码为 61H~7AH，大写英文字母“A”~“Z”对应的编码为 41H~5AH，回车符 CR 为 0DH，换行符 LF 为 0AH，“\$”符为 24H，以及“+”、“-”、“\*”、“/”、“%”等字符（参见附录 A）。