普通高等教育“十一五”国家级规划教材

国家精品课程教材·国家电工电子教学基地教学成果

高等学校工程创新型『十二五』规划计算机教材

80x86/Pentium微型计算机原理及应用

（第3版）

吴 宁 马旭东 主编

张 颖 周 芳 编著

Engineering Innovation

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

普通高等教育“十一五”国家级规划教材

高等学校工程创新型“十二五”规划计算机教材

国家精品课程教材·国家电工电子教学基地教学成果

80x86/Pentium 微型计算机原理及应用

（第3版）

吴 宁 马旭东 主编
张 颖 周 芳 编著

電子工業出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

本书是普通高等教育“十一五”国家级规划教材和国家精品课程建设成果，以教育部高等学校非计算机专业计算机基础课程“基本要求 V4.0”精神为指导，力求做到“基础性、系统性、实用性和先进性”的统一。全书共 8 章，包括计算机基础、80x86/Pentium 微处理器、80x86/Pentium 指令系统、汇编语言程序设计、半导体存储器、输入/输出和中断、微型机接口技术和微型计算机系统的发展等。本书为任课老师提供电子课件和附录列表。

本书适合作为高校工科各专业微机原理及应用（或微机原理与接口技术）课程教材，也可作为考研参考书和从业人员的参考手册。

图书在版编目（CIP）数据

80x86/Pentium 微型计算机原理及应用 / 吴宁，马旭东主编. —3 版. —北京：电子工业出版社，2011.11
高等学校工程创新型“十二五”规划计算机教材
ISBN 978-7-121-13609-2

Ⅰ. ①8… Ⅱ. ①吴… ②马… Ⅲ. ①微型计算机－高等学校－教材 Ⅳ. ①TP36

中国版本图书馆 CIP 数据核字(2011)第 092204 号

策划编辑：童占梅
责任编辑：童占梅
印　　刷：涿州市京南印刷厂
装　　订：涿州市桃园装订有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编：100036
开　　本：787×1 092　1/16　印张：23.75　字数：601 千字
印　　次：2011 年 11 月第 1 次印刷
印　　数：4000 册　　定价：39.80 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系。联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

致阅读本书的师生朋友

春华秋实，岁月有辛勤付出才美丽；

桃李芬芳，人生看学生成才而快乐！

学习是终生成长的过程，她为我们打开认识世界、自然、社会乃至我们自己的窗口，让我们有勇气、责任、知识和能力来改变现状，创造未来。

人生因学习而改变，世界因你而不同。

——编辑心语

前　言

21世纪的人类社会已进入以知识经济为主导的信息时代，计算机技术与集成电路工艺的迅猛发展，推动了以微处理器性能为表征的新器件、新技术和新方法的不断变革，信息技术产业与新兴电子产品对社会和人类文明的影响越来越大。学习与掌握微型计算机的基本知识与应用技能，已成为人类现代文明的重要组成部分。为了与计算机技术的发展和教学改革的形势相适应，我们对《80x86/Pentium微型计算机原理及应用》教材的前期版本进行了修订，并作为**普通高等教育“十一五”国家级规划教材**出版。

修订版的编写以教育部高等学校非计算机专业计算机基础课程的“基本要求 V4.0”精神为指导，结合国家和省精品课程建设成果及多年的教学实践，深入研究教材内容和课程体系，吸取国内同行师生使用前期版本的反馈意见，并参考了一些国外优秀教材。教材在前期版本的基础上对原章节内容加宽加深，既保持了多年形成的比较成熟的课程体系，又适时地引入了近年来微型计算机中的新器件、新技术和新方法，力求**做到“基础性、系统性、实用性和先进性”的统一**。

根据**工科非计算机专业**学习与应用计算机的特点与教学规律，考虑到 Intel 80x86/Pentium 系列 CPU 以及所构建 PC 系统的兼容性，本次修订版在结构上仍然继承了前版的风格，内容上则做了较大幅度的更新和调整。主要更新与特点总结如下：

（1）结构清晰，循序渐进，重点突出；内容丰富，知识的整体性更好。

（2）以 Intel 80x86/Pentium 系列 CPU 为背景，系统介绍微型计算机的组成结构、工作原理、CPU 功能特点、指令系统、汇编语言程序设计、中断的工作原理、接口技术等。

（3）重点介绍了 Intel 8086/8088 CPU 的结构与特点、基本指令集、外围接口技术的原理与应用方法等，这是我们学习与掌握微型计算机原理与应用的基础。

（4）在有关微处理器的章节中除了个别调整外，重点说明了 80386 CPU 的结构特点、寄存器组织和存储管理，增加了 Pentium CPU 的内部结构、协处理器 80x87、超线程技术、64 位微处理器和多核处理器等内容的介绍。

（5）在“指令系统”与“汇编语言程序设计”两章中，进一步完善了 Intel 80x86/Pentium 系列 CPU 的指令系统，增加了高版本宏汇编伪指令的应用、基于多媒体指令的程序设计方法、浮点运算指令集及其编程、汇编语言与高级语言的接口、保护方式下的编程方法等内容。

（6）对有关存储器的内容进行了部分调整，在保证基本概念完整、更新部分芯片的基础上，增加了双倍速 SDRAM（DDR）存储器、典型快擦写存储器（闪存）的介绍，补充介绍了 32 位处理器及存储器模块的连接应用。

（7）整合了原书的第 6、8 章的内容，系统地介绍了 8086/8088～Pentium 系列 CPU 的 I/O 系统与中断，并结合 32 位 CPU 的特点增加了保护模式下中断与异常的处理过程。

（8）从现代微处理器应用角度，调整了接口部分的内容，补充了串行接口与通信的基础知识，精简了 UART 原理和 8250 芯片的具体介绍，增加了 USB 等串行总线的基本原理、DMA 控制器 8237 的介绍，并更新了模拟通道的概念，增加了 AD574 的升级替换芯片的知识。

（9）对原有第 8 章的内容进行了更新。以介绍微型计算机体系结构为主，按序讨论了 IBM PC/AT 和 Pentium 系列微机系统，补充了最新的总线技术（如 AGP 与 PCI Express），并结合

32 位 CPU 的构成特点，介绍了保护模式下多任务管理机制和虚拟 8086 模式，对如何在保护模式下进行多任务切换进行了举例说明。

微型计算机原理及其应用课程是工科电子信息与电气学科等相关专业的重点主干课程，是后继课程学习的纽带和桥梁，是掌握微机软、硬件设计技术的基础，同时也**是后 PC 时代学习、开发和应用 DSP、ARM 及“嵌入式系统”技术的基础。**在微处理器与计算机技术飞速发展和升级换代的进程中，计算机本身的体系结构、基本工作原理并没有改变。基于此，本书仍以 8086/8088 CPU 为切入点，重点讲述 8086/8088 CPU 的构成、寄存器特点、存储器管理方式以及实模式编程技术。在此基础上，介绍 Pentium 系列各处理器的发展与特点，结合扩展的指令系统给出了 Intel 架构 32 位 CPU 的编程特点，以及多媒体和保护模式编程的基本方法。同时，对 CPU 常用外围器件，如半导体存储器、典型可编程接口芯片及其相关的中断技术、接口设计方法和典型控制程序等给予详尽的介绍。**内容组织上遵循“由易及难、循序渐进、宽编窄用”的原则**，叙述上力求做到由浅入深、通俗易懂。

全书共 8 章，第 1～4 章分别介绍微型计算机系统组成、微型计算机的 CPU、汇编语言及其程序设计等基本知识；第 5～8 章介绍微机存储系统、数据传送方式、中断技术、接口芯片及常用外部设备的相关知识及实用技术。

本书由南京航空航天大学和东南大学优秀教学团队联合编写。全书由吴宁统稿，其中，第 1、2 章由张颖编写，第 3、4 章由吴宁编写，第 5、7 章由马旭东编写，第 6、8 章由周芳编写。周磊、葛芬、段丽芬在全书编写过程中给予了许多协助。全书**教学参考学时数为 60～80**，使用时可根据具体情况选择适当的内容。本书还**为任课老师提供电子课件及由于篇幅所限不能在本书提供的附录列表**，需要者请登录**华信教育资源网 http://www.hxedu.com.cn 免费注册下载**。

前版教材《80x86/Pentium 微型计算机原理及应用》自出版以来，**连续 15 次重印，为国内多所重点大学选为本科生教学用书，并列为研究生考试的主要参考书**。借此新版出版之际，对业界同仁的信任与鼓励表示衷心的感谢。由于笔者水平有限，书中难免有错误和不妥之处，请读者批评指正。

编　著　者

目　录

第 1 章　计算机基础 …… 1
1.1　数据、信息、媒体和多媒体 …… 1
1.2　计算机中数值数据信息的表示 …… 2
1.2.1　机器数和真值 …… 2
1.2.2　数的表示方法——原码、反码和补码 …… 3
1.2.3　补码的运算 …… 6
1.2.4　定点数与浮点数 …… 7
1.2.5　BCD 码及其十进制调整 …… 10
1.3　计算机中非数值数据的信息表示 …… 12
1.3.1　西文信息的表示 …… 12
1.3.2　中文信息的表示 …… 13
1.3.3　图、声、像信息的表示 …… 14
1.4　微型计算机基本工作原理 …… 14
1.4.1　微型计算机硬件系统组成 …… 15
1.4.2　微型计算机软件系统 …… 20
1.4.3　微型计算机中指令执行的基本过程 …… 20
1.5　评估计算机性能的主要技术指标 …… 22
1.5.1　CPU 字长 …… 22
1.5.2　内存储器与高速缓存 …… 23
1.5.3　CPU 指令执行时间 …… 23
1.5.4　系统总线的传输速率 …… 24
1.5.5　iCOMP 指数 …… 24
1.5.6　优化的内部结构 …… 24
1.5.7　I/O 设备配备情况 …… 25
1.5.8　软件配备情况 …… 25
习题 1 …… 25
第 2 章　80x86/Pentium 微处理器 …… 27
2.1　80x86/Pentium 微处理器的内部结构 … 27
2.1.1　8086/8088 CPU 基本结构 …… 27
2.1.2　80386 CPU 内部结构 …… 34
2.1.3　80x87 数学协处理器 …… 47
2.1.4　Pentium CPU 内部结构 …… 50
2.1.5　Pentium 系列其他微处理器 …… 55
2.2　微处理器的主要引脚及功能 …… 55
2.2.1　8086/8088 CPU 引脚功能 …… 55
2.2.2　80386 CPU 主要引脚功能 …… 60
2.2.3　Pentium CPU 主要引脚功能 …… 61
2.3　系统总线与典型时序 …… 63
2.3.1　CPU 系统总线及其操作 …… 63
2.3.2　基本总线操作时序 …… 64
2.3.3　特殊总线操作时序 …… 66
2.4　典型 CPU 应用系统 …… 68
2.4.1　8086/8088 支持芯片 …… 68
2.4.2　8086/8088 单 CPU（最小模式）系统 …… 72
2.4.3　8086/8088 多 CPU（最大模式）系统 …… 74
2.5　CPU 的工作模式 …… 76
2.5.1　实地址模式 …… 76
2.5.2　保护模式 …… 76
2.5.3　虚拟 8086 模式 …… 77
2.5.4　系统管理模式 …… 78
2.6　指令流水线与高速缓存 …… 78
2.6.1　指令流水线和动态分支预测 …… 78
2.6.2　片内高速缓存 …… 80
2.7　64 位 CPU 与多核微处理器 …… 81
习题 2 …… 82
第 3 章　80x86/Pentium 指令系统 …… 85
3.1　80x86/Pentium 指令格式 …… 85
3.2　80x86/Pentium 寻址方式 …… 86
3.2.1　寻址方式与有效地址 EA 的概念 …… 86
3.2.2　各种寻址方式 …… 87
3.2.3　存储器寻址时的段约定 …… 90
3.3　8086/8088 CPU 指令系统 …… 90
3.3.1　数据传送类指令 …… 91
3.3.2　算术运算类指令 …… 95
3.3.3　逻辑运算与移位指令 …… 101
3.3.4　串操作指令 …… 104
3.3.5　控制转移类指令 …… 108

3.3.6 处理器控制类指令……115
3.4 80x86/Pentium CPU 指令系统……116
3.4.1 80286 CPU 的增强与增加指令……116
3.4.2 80386 CPU 的增强与增加指令……118
3.4.3 80486 CPU 增加的指令……121
3.4.4 Pentium 系列 CPU 增加的指令……121
3.5 80x87 浮点运算指令……124
3.5.1 80x87 的数据类型与格式……125
3.5.2 浮点寄存器……125
3.5.3 80x87 指令简介……125
习题 3……126
第 4 章 汇编语言程序设计……132
4.1 程序设计语言概述……132
4.2 汇编语言的程序结构与语句格式……133
4.2.1 汇编语言源程序的框架结构……133
4.2.2 汇编语言的语句……135
4.3 汇编语言的伪指令……138
4.3.1 基本伪指令语句……139
4.3.2 80x86/Pentium CPU 扩展伪指令……152
4.4 汇编语言程序设计方法……155
4.4.1 程序设计的基本过程……155
4.4.2 顺序结构程序设计……156
4.4.3 分支结构程序设计……157
4.4.4 循环结构程序设计……161
4.4.5 子程序设计与调用技术……165
4.5 模块化程序设计技术……173
4.5.1 模块化程序设计的特点与规范……174
4.5.2 程序中模块间的关系……174
4.5.3 模块化程序设计举例……175
4.6 综合应用程序设计举例……177
4.6.1 16 位实模式程序设计……177
4.6.2 基于 32 位指令的实模式程序设计……181
4.6.3 基于多媒体指令的实模式程序设计……183
4.6.4 保护模式程序设计……184
4.6.5 浮点指令程序设计……188
4.7 汇编语言与 C/C++语言混合编程……189
4.7.1 内嵌模块方法……189
4.7.2 多模块混合编程……190
习题 4……192
第 5 章 半导体存储器……196
5.1 概述……196
5.1.1 半导体存储器的分类……197
5.1.2 存储原理与地址译码……198
5.1.3 主要性能指标……200
5.2 随机存取存储器（RAM）……201
5.2.1 静态 RAM（SRAM）……201
5.2.2 动态 RAM（DRAM）……204
5.2.3 随机存取存储器 RAM 的应用……206
5.3 只读存储器（ROM）……209
5.3.1 掩膜 ROM 和 PROM……209
5.3.2 EPROM（可擦除的 PROM）……210
5.4 存储器连接与扩充应用……215
5.4.1 存储器芯片选择……215
5.4.2 存储器容量扩充……217
5.4.3 RAM 存储模块……218
5.5 CPU 与存储器的典型连接……221
5.5.1 8086/8088 CPU 的典型存储器连接……221
5.5.2 80386/Pentium CPU 的典型存储器连接……223
5.6 微机系统的内存结构……225
5.6.1 分级存储结构……225
5.6.2 高速缓存 Cache……226
5.6.3 虚拟存储器与段页结构……226
习题 5……227
第 6 章 输入/输出和中断……229
6.1 输入/输出及接口……229
6.1.1 I/O 信息的组成……229
6.1.2 I/O 接口概述……229
6.1.3 I/O 端口的编址……230
6.1.4 简单的 I/O 接口……233
6.2 输入/输出的传送方式……234

6.2.1 程序控制的输入/输出 …… 234
6.2.2 中断控制的输入/输出 …… 237
6.2.3 直接数据通道传送 …… 238
6.3 中断技术 …… 239
6.3.1 中断的基本概念 …… 239
6.3.2 中断优先权 …… 241
6.4 80x86/Pentium 中断系统 …… 243
6.4.1 中断结构 …… 243
6.4.2 中断向量表 …… 245
6.4.3 中断响应过程 …… 246
6.4.4 80386/80486/Pentium CPU 中断系统 …… 248
6.5 8259A 可编程中断控制器 …… 251
6.5.1 8259A 芯片的内部结构与引脚 …… 251
6.5.2 8259A 芯片的工作过程及工作方式 …… 253
6.5.3 8259A 命令字 …… 255
6.5.4 8259A 芯片应用举例 …… 260
6.6 中断程序设计 …… 263
6.6.1 设计方法 …… 263
6.6.2 中断程序设计举例 …… 265
习题 6 …… 270
第 7 章 微型机接口技术 …… 273
7.1 概述 …… 273
7.2 可编程定时/计数器 …… 274
7.2.1 概述 …… 274
7.2.2 可编程定时/计数器 8253 …… 275
7.2.3 可编程定时/计数器 8254 …… 282
7.3 可编程并行接口 …… 282
7.3.1 可编程并行接口芯片 8255A … 283
7.3.2 并行打印机接口应用 …… 290
7.3.3 键盘和显示器接口 …… 293
7.4 串行接口与串行通信 …… 298
7.4.1 串行通信的基本概念 …… 298
7.4.2 可编程串行通信接口 8251A … 304
7.4.3 可编程异步通信接口 INS8250 …… 311
7.4.4 通用串行总线 USB …… 311
7.4.5 I^2C 与 SPI 串行总线 …… 314
7.5 DMA 控制器接口 …… 315
7.5.1 8237A 芯片的基本功能和引脚特性 …… 316
7.5.2 8237A 芯片内部寄存器与编程 …… 318
7.5.3 8237A 应用与编程 …… 320
7.6 模拟量输入/输出接口 …… 322
7.6.1 概述 …… 322
7.6.2 并行和串行 D/A 转换器 …… 323
7.6.3 并行和串行 A/D 转换器 …… 330
习题 7 …… 337
第 8 章 微型计算机系统的发展 …… 340
8.1 微型计算机体系结构 …… 340
8.1.1 IBM PC/AT 微机系统 …… 340
8.1.2 80386、80486 微机系统 …… 341
8.1.3 Pentium 及以上微机系统 …… 342
8.2 系统外部总线 …… 344
8.2.1 ISA 总线 …… 344
8.2.2 PCI 局部总线 …… 345
8.2.3 AGP 总线 …… 346
8.2.4 PCI Express 总线 …… 347
8.3 网络接口与网络协议 …… 348
8.3.1 网络基本知识 …… 348
8.3.2 网络层次结构 …… 349
8.3.3 网络适配器 …… 350
8.3.4 802.3 协议 …… 352
8.4 80x86 的多任务保护 …… 353
8.4.1 保护机制与保护检查 …… 353
8.4.2 任务管理的概念 …… 356
8.4.3 控制转移 …… 357
8.4.4 虚拟 8086 模式与保护模式之间的切换 …… 360
8.4.5 多任务切换程序设计举例 …… 361
习题 8 …… 368
参考文献 …… 369

第 1 章　计算机基础

微型计算机是性能提高最快、应用最广泛的数字计算机。通用微型计算机配有比较完善的系统软件和外围设备，一般可用于数值计算、信息处理和智能控制等诸多方面。本章主要介绍数字计算机数据处理的基础、计算机系统的基本构成以及主要的技术指标。

1.1　数据、信息、媒体和多媒体

微型计算机最基本的用途是信息处理，本节将介绍数据、信息和媒体等相关概念。

1．数据

国际标准化组织（ISO）对数据的定义是：“数据是对事实、概念或指令的一种特殊表达形式，这种特殊的表达形式可以用人工的方法或者用自动化的装置进行通信、翻译转换或者进行加工处理。”根据这一定义，通常意义下的数字、文字、图形、图像、声音、活动图像（视频）等都可认为是数据，因为人们可以对它们进行各种人工方式的处理，如通信、加工、转换等。

对计算机而言，数字、文字、图形、图像、声音、活动图像等都不能直接由它进行处理，它们必须采用“特殊的表达形式”才能由计算机进行通信、转移或加工处理。这种特殊的表达形式就是二进制编码形式。

通常，在计算机内部又将数据分为数值型数据和非数值型数据。数值型数据是指日常生活中经常接触到的数字类数据，它主要用来表示数量的多少，可以比较大小；而 ISO 定义中其他的数据统称为非数值型数据。在非数值型数据中又有一类最常用的数据，称为字符型数据，它可以方便地表示文字信息，供人们直接阅读和理解。其他的非数值型数据则主要用来表示图形、图像、声音和活动图像。

用计算机进行数据处理指对数据进行加工、转换、存储、分类、排序和计算的过程。数据处理的目的是从原始数据或基础数据生成或转移得到对使用者有用的数据。

2．信息

根据 ISO 的定义可以通俗地认为：信息是对人有用的数据，这些数据可能影响人们的行为与决策。

计算机信息处理，实质上就是由计算机进行数据处理的过程。也就是说，通过数据的采集和输入，有效地把数据组织到计算机中，由计算机系统对数据进行相应的存储、转换、合并、分类、计算、汇总等操作。经过计算机对数据的加工处理，向人们提供有用的信息，这一过程就是信息处理。更通俗地说，信息处理的本质就是数据处理，数据处理的主要目的是获取有用的信息。

3．媒体

媒体又称媒介、媒质，英文是 medium（单数）和 media（复数），指承载信息的载体。根据国际电信联盟（ITU）下属的国际电报电话咨询委员会（CCITT）的定义，与计算机信息处理有关的媒体有下列 5 种：感觉媒体、表示媒体（二进制编码）、存储媒体（存储器）、表现媒体（输入或输出设备）、传输媒体。

4．多媒体

多媒体技术中的多媒体，指多种感觉媒体。所谓多媒体技术，是指能够交互式地综合处理

多种不同感觉媒体（语言、音乐、文字、数值、图形、图像、活动图像，其中至少包含声音或活动图像）的信息处理技术。具有这种功能的计算机就是多媒体计算机，具有这种能力的通信系统就是多媒体通信系统，能够有效地存储、管理、检索多种感觉媒体的数据库系统就是多媒体数据库系统。多媒体技术的重点是使计算机能很好地处理声音信息及（动态）视、听特性的媒体，如视频、全活动景像等。多媒体信息的处理涉及大量的数据，所以大容量数据存储、数据压缩和解压缩技术也是多媒体技术的重要发展方向之一。

多媒体技术的发展，使计算机更便利地进入了人类生活的各个领域，促进了全新的信息产品制造业与信息服务业的繁荣兴旺，促使人与计算机之间建立起更默契、更融洽的新型关系。

1.2 计算机中数值数据信息的表示

1.2.1 机器数和真值

计算机在本质上只能识别 0、1 表示的二进制数码。为了表示正数和负数，专门选择一位二进制数来表示数的符号，该位为 0，表示正号；该位为 1，表示负号，通常选择最高位作为符号位。

这就是说，数的符号在计算机中也数值化了。一个数在机器（计算机）中的表示形式称为机器数，而数本身的实际值叫做真值（机器数真值）。真值可以用二进制数表示，也可用十进制数表示，但根据习惯，常用十进制数表示，如图 1-1 所示。

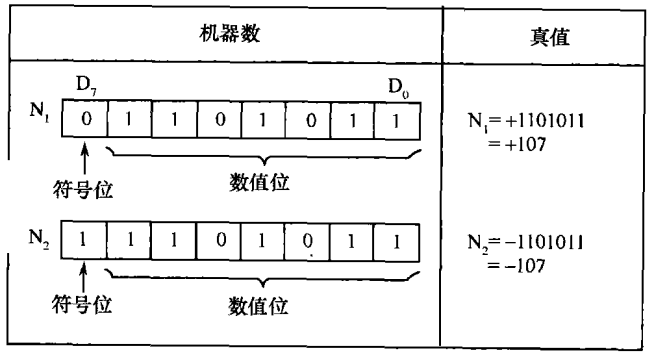


图 1-1 机器数真值

机器数有如下特点：

（1）机器数的正、负号是数值化的。

（2）机器数所能表示的数的范围受到机器（计算机）字长的限制。

那么什么是计算机字长呢?我们先来解释几个计算机中的常用术语。

① 位（bit）：是计算机所能表示的最小数据单位，它只有两种状态 0 和 1。要想表示更大的数，就得把更多的位组合起来作为一个整体，每增加一位，所能表示的数就增大一倍。

② 字节（Byte）：一个 8 位二进制数称为 1 字节，它是计算机处理数据的基本单位（B）。在计算机中，其存储器容量大小常以字节数的多少来度量。常用的 4 种度量单位是 KB、MB、GB 和 TB，其大小分别为：

$1KB=2^{10}B=1024B$

$1MB=2^{10}\times 2^{10}B=1024KB$

$1GB=2^{10}\times2^{10}\times2^{10}\ B=1024MB$

$1TB=2^{10}\times2^{10}\times2^{10}\times2^{10}\ B=1024GB$

③ 字（Word）：作为单位一般指 16 位二进制数，但也把计算机处理数据时，处理器通过数据总线一次存取、加工和传送的数据位数称为字，这时字通常由 1 字节或若干字节组成。

④ 字长（Word Length）：指处理器的二进制位数。8 位微处理器的字长为 8 位；16 位微处理器的字由 2 字节构成；32 位微处理器的字则由 4 字节构成。字长是衡量计算机性能的一个重要标志，字长越长，性能越强，精度越高。

（3）小数点不能直接标出，需要按一定方式约定小数点的位置。

需要说明的是，在计算机系统开发和应用中会涉及二进制（Binary，简写 B）、八进制（Octal，O 或 Q）、十进制（Decimal，D）和十六进制（Hexdecimal，H）等数制，其中十六进制最常用、最方便，它与二进制的关系简明、直接，并用 0～9，A～F（H）代表 0～15（D）。

1.2.2 数的表示方法——原码、反码和补码

为妥善解决符号数值化的表示与运算问题，设定了机器数 x 的三种不同编码形式，即原码、反码和补码。分别记作$[x]_{原}$、$[x]_{反}$和$[x]_{补}$。

1. 原码

设 $x=x_1x_2\cdots x_{n-1}$，其中 x_i 为一位二进制数，$i=1,2,\cdots,(n-1)$。n 位二进制数

$$[x]_{原}=\begin{cases}0x_1x_2\ldots x_{n-1} & 当x\geqslant 0\\ 1x_1x_2\ldots x_{n-1} & 当x\leqslant 0\end{cases}$$

一个数的原码，就是数值部分为绝对值，加上用 0 和 1 分别表示数的符号+和−的机器数。

【例 1-1】 $x_1=67=+1000011$，$[x_1]_{原}=01000011$；$x_2=-67=-1000011$，$[x_2]_{原}=11000011$。

在原码表示法中，根据定义，数 0 的原码有两种不同形式（设字长为 8 位）：

$$[+0]_{原}=00000000，[-0]_{原}=10000000$$

原码表示简单易懂，而且与真值的转换方便。但原码表示的数不便于计算机运算，因为在两原码数运算时，首先要判断它们的符号，然后再决定用加法还是用减法，导致机器的结构相应地复杂化或增加机器的运算时间。为解决上述问题，引入反码和补码表示法。

2. 反码

设 $x=x_1x_2\cdots x_{n-1}$，其中 x_i 为一位二进制数，$i=1,2,\cdots,(n-1)$。定义

$$[x]_{反}=\begin{cases}0x_1x_2\ldots x_{n-1} & 当x\geqslant 0\\ 1\overline{x}_1\overline{x}_2\ldots\overline{x}_{n-1} & 当x\leqslant 0\end{cases}$$

式中，

$$\overline{x}_i=\begin{cases}0 & 当x_i=1\\ 1 & 当x_i=0\end{cases}$$

根据定义，正数的反码与原码相同；对于负数，反码的符号位为 1，其余位为数值位按位取反。

【例 1-2】 $x_1=83=+1010011$，$[x_1]_{反}=01010011$；$x_2=-83=-1010011$，$[x_2]_{反}=10101100$。

在反码表示法中，根据定义，数 0 的反码也有两种不同形式（设字长为 8 位）：

$$[+0]_{反}=00000000，[-0]_{反}=11111111$$

3. 补码

引入补码的概念，一方面是为了解决原码、反码存在的一些问题，另一方面是为了将计算机中的加、减运算简化为单纯的加法运算，这种运算上的特性与补码本身的定义有关。

（1）同余的概念和补码

设有两个数 a=17，b=27，若用 10 去除 a 和 b，则它们的余数均为 7，我们称 17 和 27 在以 10 为模时是同余的，并记作 17=27 (mod 10)。

或者说，17 和 27 在以 10 为模时是相等的。此处的模是一个计量系统所能表示的最大量程（或一个计量单位称为模或模数）。

由同余概念，不难得出

$$a+M=a \pmod{M} \quad , \quad a+2M=a \pmod{M}$$

因此当 a 为负数时，如 a=-4，在以 10 为模时，有

$$-4+10=-4 \pmod{10} \quad 即 \quad 6=-4 \pmod{10}$$

上式说明，在以 10 为模时，-4 与+6 是相等的。我们称+6 为-4 的补码，或者说+6 与-4 对模 10 来说互为补数。有了补码的概念，就可将减法转化为加法（加补码）来进行。如：

$$7-4=7+6 \pmod{10}$$

该式说明，在以 10 为模时，7 减 4 可以通过 7 加-4 的补码 6 来进行，而所得结果是相同的（只要在加补码 6 时，将所产生的进位舍弃即可，这正是以 10 为模的意思）。

现在把同余和补码的概念推广到二进制数。设计算机字长为 n 位，那么其模为 2^n，即

$$\underbrace{1\overbrace{0\ 0\cdots 0}^{n\text{个}\ 0}}_{n+1\text{位}}$$

这就是说，字长 n 位的机器，2^n 在机器中仅能以 n 个 0 表示。或者说，2^n 和 0 在机器中表示的形式是一样的。

（2）补码求法

设 $x=x_1x_2\cdots x_{n-1}$，其中 x_i 为一位二进制数，i=1，2，…，(n-1)。定义

$$[x]_{补}=\begin{cases} 0x_1x_2\ldots x_{n-1} & 当 x \geqslant 0 \\ 1\overline{x}_1\overline{x}_2\ldots\overline{x}_{n-1}+1 & 当 x \leqslant 0 \end{cases}$$

或写成

$$[x]_{补}=\begin{cases} x & 当 0 \leqslant x \leqslant 2^{n-1} \\ 2^n+x & 当 -2^{n-1} \leqslant x \leqslant 0 \end{cases} \pmod{2^n}$$

① 正数的补码。$[x]_{补}=[x]_{原}=[x]_{反}$ 。

【例 1-3】 $[+127]_{原}=[+127]_{反}=[+127]_{补}=\underbrace{01111111}_{字长为8位}$。

② 负数的补码。先求负数的反码，然后在反码最低一位上加 1 就得到该负数的补码。

还可以用形式上更简便的方法求得负数 x 的补码：符号位为 1，将原数值中最右边一个 1 及其后面的 0 保持不变，而最右一个 1 以左的各位按位取反。

【例 1-4】 x_1=−0011000，$[x_1]_{补}$=11101000；x_2=−1000000，$[x_2]_{补}$=11000000。

根据定义，数 0 的补码仅有一种形式：$[+0]_{补}=[-0]_{补}=\underbrace{00\cdots0}_{n位}$。

若已知一个负数的补码，再取一次补，则 $\{[x]_{补}\}_{补}=[x]_{原}$。

【例 1-5】 设$[x]_{补}=10010111$，求 x 的真值。

$$\begin{array}{r} 11101000 \\ +\quad\quad 1 \\ \hline 11101001 \end{array}$$

所以，$[x]_{原}=11101001$，$x=-1101001=-105$。

当已知 8 位二进数补码符号位为 1 时，表示该数为负数。此时要注意，其余几位不是该二进制的数值，一定要把它们按位取反，且在最低位加 1，才是该二进制数的值（真值）。

十进制数及其对应的 8 位二进制数的原码、反码和补码表示如表 1-1 所示。

表 1-1 十进制数及其对应的 8 位二进制数的表示方法

十进制数	二进制数	原码	反码	补码
+0	+0000000	00000000	00000000	00000000
+1	+0000001	00000001	00000001	00000001
+2	+0000010	00000010	00000010	00000010
…	…	…	…	…
+126	+1111110	01111110	01111110	01111110
+127	+1111111	01111111	01111111	01111111
-0	-0000000	10000000	11111111	00000000
-1	-0000001	10000001	11111110	11111111
-2	-0000010	10000010	11111101	11111110
…	…	…	…	…
-126	-1111110	11111110	10000001	10000010
-127	-1111111	11111111	10000000	10000001
-128	-1000000	无法表示	无法表示	10000000

小结：

（1）有符号数三种编码的最高位都是符号位。符号位为 0，表示真值为正数，其余位为真值；符号位为 1，表示真值为负数，其余位除原码外，不再是真值。对于反码，只需按位取反，便是真值；对于补码，还需按位取反加 1，才是真值。

（2）对正数，三种编码是一样的，即$[x]_{原}=[x]_{反}=[x]_{补}$。对于负数，三种编码就不同了。所以说，原码、反码和补码的实质是用来解决负数在机器中的表示而引入的编码方法。

（3）8 位二进制数原码、反码和补码所能表示的数值范围不完全相同。它们分别是-127～+127，-127～+127 和-128～+127。其中对 0 的表示也不尽相同，原码和反码有两种表示方法，补码只有一种表示方法。

（4）当计算机采用不同的码制时，运算器和控制器的结构将不同。采用原码形式的计算机称原码机。类似地，有反码机和补码机。小型计算机和微型计算机大多为补码机（实际上原码机与反码机已不采用了）。

1.2.3 补码的运算

从以上讨论可知，在微型计算机中，有符号数以补码的形式在机器中存在和运算。这主要是因为补码的加、减运算比原码简单；符号位与数值位一起参加运算，并能自动获得正确结果。

设 x 和 y 是两个正数，可以证明，两个数和的补码等于两个数补码的和：

$$[x+y]_{补}=2^n+(x+y)=(2^n+x)+(2^n+y)=[x]_{补}+[y]_{补}$$

同样也可以证明，该两数差的补码等于被减数的补码与减数负值的补码（或称求补）之和。

$$[x-y]_{补}=2^n+(x-y)=2^n+x+2^n+(-y)=[x]_{补}+[-y]_{补}$$

上式说明，在补码运算中，两数差的运算可简化为单纯的加法运算。

x 的补码直接由补码定义可得到，而$[-y]_{补}$可通过对$[y]_{补}$“连同符号位在内一起变反加 1”得到（称为“求补”）。

【例 1-6】 $[y]_{补}$=00000100，$[-y]_{补}$=11111100。

【例 1-7】 计算 $x-y$。x，y 均为正数，且 $x>y$。设 x=122，y=37，字长 n=8。

解：

十进制计算	二进制补码计算
122	01111010 = $[x]_{补}$
− 37	+ 11011011 = $[-y]_{补}$
85	101010101

进位自动舍去 ↑ ——— 符号位为 0，表示正数

求得真值为正数$(01010101)_2$=85。

【例 1-8】 计算 $x-y$。x，y 均为正数，且 $x<y$。设 x=64，y=65，字长 n=8。

解：

十进制计算	二进制补码计算	
64	01000000 = $[x]_{补}$	
− 65	+ 10111111 = $[-y]_{补}$	
−1	11111111	符号位为 1，表示负数

求真值：$(11111111)_{补}=\{(11111111)_{补}\}_{补}=-(00000001)=-1$。

【例 1-9】 计算 $x+y$。x，y 均为正数。设 x=64，y=65，字长 n=8。

解：

十进制计算	二进制补码计算	
64	01000000 = $[x]_{补}$	
+ 65	+ 01000001 = $[y]_{补}$	
129	10000001	符号位为 1，表示负数

此时，两个正数相加得出负数，显然是错误的。这种情况称为“溢出”。由表 1-1 知，8 位计算机中，由于最高位为符号位，剩下的数值位只有 7 位，表示数的范围是−128～+127。当两个正数相加其和大于 127 或两个负数相减其绝对值之和大于 128 时，就产生了“溢出”，致使结果出错。推广到字长 n 位的符号数，最高位为符号位，$n-1$ 位表示数值。能表示的最大值为 $2^{n-1}-1$（即 $n-1$ 个 1）。当运算结果超过此值，就会产生“溢出”。

小结：

（1）补码运算时，参加运算的两个数均为补码，结果也是补码，需转换才能得到真值。

（2）可以将减法变为加法运算，以简化硬件设计。

（3）运算时：① 符号位与数值位一起参加运算；② 符号位产生的进位可舍弃；③ 要保

证运算结果不超过补码所能表示的数的最大范围，否则将产生“溢出”错误。为此，在计算机中设有专门电路用以判断运算结果是否产生溢出，并设置溢出标志告知本次运算的结果的状态。

（4）无符号数和有符号数的加法运算可用同一电路完成。

1.2.4 定点数与浮点数

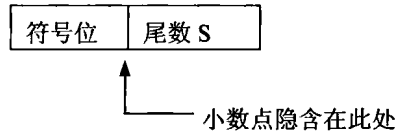
在计算机中，不仅要处理整数运算，而且也要处理小数运算，如何处理小数点位置是十分重要的，通常用定点法和浮点法来表示小数点的位置。

（1）定点表示法。就是小数点位置在数中固定不变。例如：

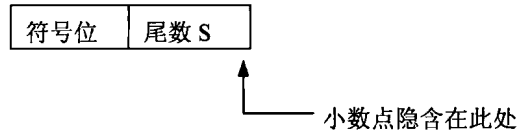
1.100101，　0.100101，　110.0101

一般来说，小数点位置固定在哪个位置上并无限制，但为了使用简便，在计算机中有两种常用的定点数。

① 定点纯小数。把小数点固定在最高数值位左边，在小数点左边设有一位符号位，小数点本身不占位，其格式为：

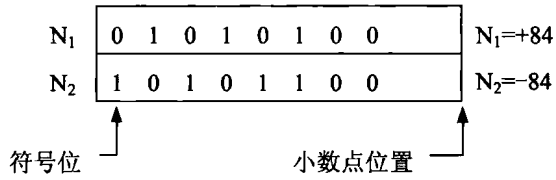


② 定点纯整数。把小数点固定在最低数值位右边，最高位仍为符号位，小数点本身不占位，其格式为：

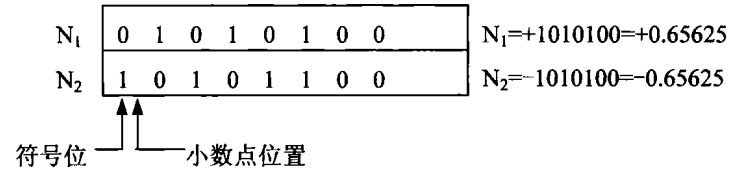


定点数的这两种表示法，在计算机中均有采用。但对一台机器而言，采用哪种方法，需事先约定。

【例 1-10】 有如下两个 8 位二进制数：



【例 1-11】 有例 1-10 中同样两个数，但小数点位置不同，则有



从上面两个例子可以看出，定点整数和定点小数在格式上毫无差别，这是因为定点数的小数点是隐含的，但它们的真值却不相同。此外，相同位数、不同码制的定点数表示的数的范围也不相同。

同一台计算机可处理的整数类型往往有很多种，以目前广泛使用的 Pentium 机为例，除了无符号整数之外，它还有三种不同类型的整数，如表 1-2 所示。

表 1-2　Pentium 处理器的三种整数类型

整数类型	数值范围	精　度	格　式
16 位整数	-32768～32767	二进制 16 位	16 个二进制位，补码表示
短整数	-2^{31}～$2^{31}-1$	二进制 32 位	32 个二进制位，补码表示
长整数	-2^{63}～$2^{63}-1$	二进制 64 位	64 个二进制位，补码表示

由于定点数是将小数点的位置固定，因此运算起来很不方便。一方面，它要求对所有原始数据要用比例因子化成小数或整数，算出结果又要用比例因子折算成真值；另一方面，这种方法所表示的数的范围小、精度低。所以有必要引入另一表示方法——浮点表示法。

（2）浮点表示法。为了在位数有限的前提下，尽量扩大数的表示范围，同时又保持数的有效精度，计算机往往采用浮点数表示数值。在浮点表示初期，把一个数通过改变小数点位置表示成 2 的 p 次幂和绝对值小于 1 的数 S 相乘的形式：

$$N = 2^p S$$

式中，N 称为浮点数或实数；S 是 N 的尾数，它是数值的有效数字部分，通常用带符号的定点小数表示，一般用原码表示；2 是 N 的底数，是该进位计数制的基数，在计算机中不出现，是隐含的；p 是指数，称为阶码，通常为带符号整数，一般用补码表示。阶码 p 的大小决定了数的范围，尾数 S 的长短则规定了数的有效数字的位数（精度）。在计算机中，p 和 S 均为二进制数。

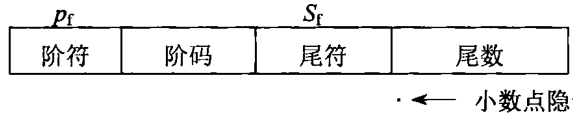
【例 1-12】　试将以下数表示成浮点数形式。

$1011.1101\text{B} = (0.10111101\text{B})\times 2^4 = (0.10111101)\times 2^{100}\text{B}$

$0.00101101\text{B} = (0.101101\text{B})\times 2^{-2} = (0.101101)\times 2^{-10}\text{B}$

$-111001010\text{B} = (-0.111001010\text{B})\times 2^9 = (-0.111001010)\times 2^{1001}\text{B}$

浮点数在机器中的一种表示形式如下：



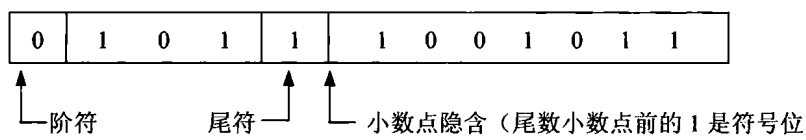
也就是说，若要在机器中表示一个浮点数，阶码和尾数要分别表示，且都有自己的符号位。

通常，用一位二进制数 p_f 表示阶码的符号位。当 p_f=0 时，表示阶码为正；当 p_f= 1 时，表示阶码为负。同样，用一位二进制数 S_f 表示尾数的符号位。当 S_f= 0 时，尾数为正；当 S_f= 1 时，尾数为负。

若浮点数中 $1/2 \leqslant S < 1$，则称该浮点数为规格化了的浮点数。

【例 1-13】　$(-18.75)_{10}=(-10010.11)_2=(-0.1001011)\times 2^{+101}$。

假定尾数用 8 位二进制数表示，阶码用 4 位二进制数表示，且均含符号位，尾数用原码表示，则有



或表示成$(1.1001011)\times(10)^{0101}$。

【例 1-14】　$(0.078125)_{10}=(0.000101)_2=(0.101)\times 2^{-11}$。