



21世纪高等院校计算机科学与工程系列教材

北京希望电子出版社 总策划
陈家林 主 编
赵正华 陈裕国 苏文静 编 著

微型计算机 原理与接口技术

 科学出版社
www.sciencep.com



北京希望电子出版社 总策划
陈家林 主 编
赵正华 陈裕国 苏文静 编 著

微型计算机 原理与接口技术



科学出版社
www.sciencep.com

内 容 简 介

本书是“21 世纪高等院校计算机科学与工程系列教材”中的一本，通过详尽的讲解，分析了当前比较流行的接口技术，同时也深入浅出地讲解了微型计算机的基本原理。

全书共 8 章，以 8086/8088 CPU 为例，分析了微处理器的基本结构、指令系统、存储系统和输入/输出接口电路。本书结构新颖，注重基本概念和基本分析方法的学习和运用；例题丰富，配合实验，形式多样，点面结合，举一反三；重点突出，难点分散，循序渐进，通俗易懂。

本书可作为高等院校“微型计算机原理与接口技术”的教材，也可作为计算机培训班的教材和广大计算机爱好者的入门自学参考书。

需要本书或技术支持的读者，请与北京中关村 083 信箱（邮编：100080）发行部联系，电话：010-82702660，82702658，62978181（总机）转 103 或 238 传真：010-82702698 E-mail: yanmc@bhp.com.cn。

图书在版编目（CIP）数据

微型计算机原理与接口技术/陈家林主编；赵正华，陈裕国，苏文静编著.—北京：科学出版社，2004.11

21 世纪高等院校计算机科学与工程系列教材

ISBN 7-03-014062-1

I. 微... II. ①陈...②赵...③陈...④苏... 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 IV. TP36

中国版本图书馆 CIP 数据核字（2004）第 078301 号

责任编辑：曾华 / 责任校对：肖寒

责任印刷：双青 / 封面设计：王翼

科学出版社 出版

北京东黄城根北街16号

邮政编码：100717

<http://www.sciencep.com>

双青印刷厂 印刷

科学出版社发行 各地新华书店经销

*

2004 年 11 月第 一 版 开本：787×1092 1/16

2004 年 11 月第一次印刷 印张：16

印数：1-3000 字数：361 608

定价：24.00 元

21世纪高等院校计算机教材编委会名单

(排名不分先后)

主任:	陈火旺	院士	
副主任:	李仁发	教授	金茂忠 教授 陈 忠 教授 陆卫民 高工
委员:	晏海华	教授	北京航空航天大学
	邵秀丽	副教授	南开大学
	刘振安	教授	中国科技大学
	董玉德	副教授	合肥工业大学
	倪志伟	教授	合肥工业大学
	吕英华	教授	东北师范大学
	杨喜权	副教授	东北师范大学
	朱诗兵	副教授	装备指挥学院
	樊秀梅	副教授	北京理工大学
	徐 安	教授	上海同济大学
	赵 欢	副教授	湖南大学
	胡学钢	教授	合肥工业大学
	林福宗	教授	清华大学
	王家昕	教授	清华大学
	郑 莉	教授	清华大学
	朱淼良	教授	浙江大学
	刁成嘉	副教授	南开大学
	林和平	教授	东北师范大学
	孙铁利	教授	东北师范大学
	温子梅	讲师	广东教育学院
	吕国英	副教授	山西大学
	张广州	讲师	沈阳大学
	何新华	教授	装甲学院
	邱仲潘	副教授	厦门大学
	曾春平	副教授	第二航空学院
	姬东耀	教授	中科院计算所
	赵宏利	教授	装备指挥学院
	喻 飞	博士	浙江大学
	徐建华	总编	北京希望电子出版社
	郑明红	副总编	北京希望电子出版社
	韩素华	编室主任	北京希望电子出版社

总 序

21 世纪挑战与机遇并存, 没有足够的知识储备必将被时代所抛弃。中国 IT 教育产业竞争日趋激烈, 用户需求凸现个性, 行业发展更需要理性。未来五年 IT 行业将以每年 18% 的速度连续增长, 将引发 IT 产业新的发展高潮。实现信息产业大国的目标, 应该依赖教育, 要圆信息产业强国的梦想, 依然要寄托于教育, IT 教育事业任重道远, 其产业也正面临着机遇与挑战。

我国的计算机教学长久以来一直重原理、轻应用。高等院校的计算机教学机制和教材对计算机本身的认识都存在误区。要改革高校计算机教学, 教材改革是重要方面, 用计算机教材的改革促进基础教育的改革势在必行。

一本好书, 是人生前进的阶梯; 一套好教材, 就是教学成功的保证。为缓解计算机技术飞速发展与计算机教材滞后落伍的矛盾, 我们通过调查多所院校的师生, 并多次研讨, 根据读者认识规律, 开创出一种全新的方式, 打破过去介绍原理——理论推导——举例说明的模式, 增加实用操作性, 通过上机实验与课上内容结合来增强可读性, 用通俗易懂的语言和例子说明复杂的概念。

本套教材的特点一是“精”, 精选教学内容; 二是“新”, 捕捉最新资讯; 三是“特”, 配备电子课件, 力争达到基础性、先进性、全面性、典型性和可操作性的最大统一。

为保证教材质量, 我们同时聘请了一批学术水平较高的知名专家、教授作为本套教材的主审和编委。全套教材包括必修课教材二十多种, 选修课教材和学习配套用书 10 余种, 基本上涵盖了目前高等院校(含高等职业技术学院、高等专科学校、成人高等学校)计算机科学与技术专业所必修或选修的内容。各种教材编写时既注意到内容上的连贯性, 又保证了教学上的相对独立性。

本套教材在内容的组织上, 大胆汲取当今计算机领域最新技术, 摒弃了传统教材中陈旧过时的内容。这些变化在各本教材中都得到了不同程度的体现。本套教材编写时既参照了教育部有关计算机科学与技术专业的教学要求, 又参考了“程序员考试大纲”和“全国计算机水平等级考试大纲”的内容, 因此既适合作为高等学校计算机科学与技术专业教材, 也可作为计算机等级考试学习用书。

考虑到各校教学特点和计算机设备条件, 我们本着“学以致用”的理念, 在本套教材编写中自始至终贯彻“由浅入深, 理论联系实际”的原则, 以阐明要义为主, 辅之以必要的例题、习题和上机实习, 能够使学生尽快领悟和掌握。

在本套教材编写过程中, 作者们付出了艰辛的劳动, 教材编委会的各位专家、教授进行了认真的审定和悉心的指导。书中参考、借鉴了国内外同类教材和专著, 在此一并表示感谢。

我们希望更多的优秀教师参与到教材建设中来, 真诚希望广大教师、学生与读者朋友在使用本丛书过程中提出宝贵意见和建议。

若有投稿或建议, 请发至本丛书出版者电子邮件: hansuhua@163bj.com

21 世纪高等院校计算机教材编委会

前 言

本教材是在总结多年教学和科研经验的基础上,结合微型计算机技术发展趋势和当前教材改革的需要,在认真研究了大学扩招后本科教学出现的新情况的基础上,为进一步提高教学效果而重新编写的。本教材特别之处在于:

重点突出,通俗易懂,以讲授 8086/8088 CPU 为主,内容力求精练;

难点分散,深入浅出,强调概念和方法的掌握和运用;

例题丰富,形式多样,便于教师点面结合地讲授和学员举一反三地自学。

通过本课程的学习,能够了解微机系统的组成、工作原理,掌握指令的寻址方式和 8086 的指令系统,能用汇编语言进行简单编程,掌握微机的基本接口技术和接口芯片的应用方法,具备设计和使用常用微机接口电路的能力,为后续课程的学习和工程实践打下良好的基础。学习各章节时,需要重点注意的内容如下:

第 1 章微机的组成、发展、工作原理,原码、反码、补码及运算法则,数制及其之间的转换,8086/8088 CPU 的结构、引脚、接线及时序,寄存器结构和存储器组织。

第 2 章指令的格式和分类,指令的寻址方式,指令的使用。

第 3 章汇编语言的编程环境、DOS 与 BIOS 的功能调用,汇编语言的指令格式、伪指令语句、属性操作符,顺序结构、分支结构、循环结构、子程序结构的程序设计方法。

第 4 章存储器的技术指标、分类及层次结构,DRAM 的工作原理,SRAM 的原理及芯片接口方法,存储器的扩展和接口设计。

第 5 章 I/O 端口的功能、编址方式,与 CPU 之间的信息传送方式,简单接口电路的分析、设计、编程和应用,中断的基本原理和过程,8086/8088 的中断系统和可编程中断控制器 8259A 的使用。

第 6 章可编程定时器 8253、可编程并行接口 8255、可编程串行通信接口 8250 的工作原理、编程和应用,RS-232/RS-422/RS-485 串行通信总线的概念,接口电路的工作过程和程序设计。

第 7 章 D/A 转换器和 A/D 转换器的工作原理和常用 8 位 D/A 和 A/D 的使用方法。

第 8 章汇编程序的设计调试过程,常用软件工具的使用方法,基本实验。

建议本课程按 76 学时安排,其中教学课时为 68 学时,实验课时为 8 学时。

本书第 1、2、3 章由苏文静编撰,第 4、5 章由赵正华编撰,第 6、7 章由陈家林编撰,第 8 章由陈裕国编撰。全书由杨宁、陈家林策划,陈家林组织编写和校对,杨宁负责审核。

在本书写作过程中许多同事和同行热情地提出不少建议,特别是谈宏华、杨志方两位老师给予了极大的帮助和支持,在此我们表示衷心的感谢。

由于编者水平有限,编写时间仓促,书中错误和不当之处在所难免,敬请各位读者和专家批评指正,以便再版时及时修正。

编 者

目 录

第1章 微型计算机基础1	2.1.8 隐含寻址..... 37
1.1 计算机中的数制.....1	2.2 8086 指令系统..... 38
1.1.1 常用计数制.....1	2.2.1 数据传送类..... 38
1.1.2 各种数制之间的转换.....2	2.2.2 算术运算类..... 42
1.2 二进制数的运算.....4	2.2.3 逻辑运算和移位类..... 45
1.2.1 无符号数和有符号数的表示.....4	2.2.4 串操作类..... 47
1.2.2 算术运算.....5	2.2.5 控制转移类..... 50
1.2.3 逻辑运算.....7	2.2.6 处理器控制类..... 54
1.3 二进制编码.....8	习题..... 55
1.3.1 二进制编码的概念.....8	第3章 汇编语言程序设计 57
1.3.2 十进制数的二进制编码.....8	3.1 汇编语言程序概述..... 57
1.3.3 字符的二进制编码.....9	3.1.1 汇编语言程序的特点..... 57
1.4 微型计算机的基本结构.....10	3.1.2 关于汇编语言程序设计..... 57
1.4.1 微型计算机的组成.....10	3.1.3 汇编语言程序的运行..... 58
1.4.2 微型计算机的概念结构.....12	3.2 8088 汇编语言和程序框架..... 59
1.4.3 微型机的工作过程.....16	3.2.1 8088 汇编语言的数据项..... 59
1.5 8088/8086 微处理器.....19	3.2.2 8088 汇编语言伪指令..... 61
1.5.1 8088/8086 的外部引脚.....19	3.2.3 8088 汇编语言程序的框架..... 63
1.5.2 8088/8086 的功能结构.....22	3.3 DOS 系统功能调用..... 65
1.5.3 8088/8086 的编程结构.....23	3.3.1 DOS 系统功能调用..... 65
1.5.4 存储器及 I/O 组织.....25	3.3.2 DOS 系统功能调用的方法..... 65
1.6 8088 系统总线.....26	3.3.3 DOS 系统功能调用示例..... 65
1.6.1 8088 系统总线.....26	3.4 汇编语言程序设计基础..... 66
1.6.2 8088 的工作时序.....29	3.4.1 基本程序结构..... 66
习题.....32	3.4.2 模块化程序设计方法简介..... 76
第2章 8088/8086 指令系统34	习题..... 77
2.1 8086 的寻址方式.....34	第4章 存储器与存储器扩展 80
2.1.1 立即寻址.....34	4.1 概述..... 80
2.1.2 直接寻址.....35	4.1.1 存储器的一般概念..... 80
2.1.3 寄存器寻址.....35	4.1.2 存储器芯片的主要性能指标..... 80
2.1.4 寄存器间接寻址.....35	4.1.3 存储器的分类..... 81
2.1.5 寄存器相对寻址.....36	4.1.4 内存的基本组成..... 82
2.1.6 基址—变址寻址.....37	4.2 随机存储器..... 83
2.1.7 基址—变址—相对寻址.....37	4.2.1 静态随机存储器 SRAM..... 83

4.2.2 动态随机存储器 DRAM.....	87	6.1.3 8253 的 6 种工作方式.....	155
4.3 只读存储器.....	91	6.1.4 8253 的应用.....	160
4.3.1 可擦除的可编程只读 存储器 EPROM.....	91	6.2 可编程并行接口 8255.....	162
4.3.2 E ² PRO.....	96	6.2.1 8255 的外部引脚及内部结构.....	163
4.4 存储器芯片扩展及其与 CPU 的连接.....	100	6.2.2 8255 的控制字.....	165
4.4.1 存储器芯片与 CPU 的连接.....	100	6.2.3 8255 的工作方式.....	166
4.4.2 存储器芯片的扩展.....	104	6.2.4 8255 应用举例.....	171
习题.....	108	6.3 可编程串行通信接口 8250.....	175
第 5 章 输入/输出和中断技术.....	109	6.3.1 串行通信基本概念.....	175
5.1 输入/输出接口.....	109	6.3.2 RS-232-C 串行通信接口.....	177
5.1.1 I/O 接口的任务与功能.....	109	6.3.3 可编程串行通信接口 8250.....	178
5.1.2 I/O 接口的编址方式.....	110	习题.....	187
5.1.3 I/O 端口地址的译码.....	111	第 7 章 数/模和模/数转换.....	189
5.1.4 I/O 数据的传送方式.....	112	7.1 数/模转换器.....	189
5.2 简单接口电路.....	113	7.1.1 数/模转换器的基本原理.....	189
5.2.1 接口电路的基本构成.....	113	7.1.2 数/模转换器的主要技术指标.....	192
5.2.2 三态门缓冲器接口芯片.....	114	7.1.3 典型的数/模转换器芯片及其 与系统的连接.....	193
5.2.3 锁存器接口芯片.....	116	7.2 模/数转换器.....	198
5.2.4 简单接口的应用举例.....	117	7.2.1 模/数转换器的工作原理及 技术指标.....	198
5.3 输入/输出的控制方式.....	118	7.2.2 典型的模/数转换器芯片 ADC0809.....	200
5.3.1 无条件传送方式.....	118	习题.....	204
5.3.2 程序查询传送方式.....	119	第 8 章 实验.....	206
5.3.3 程序中断传送方式.....	121	8.1 汇编软件使用与基本编程实验.....	206
5.3.4 直接数据通道传送方式.....	121	8.1.1 实验目的.....	206
5.4 中断技术.....	123	8.1.2 实验环境.....	206
5.4.1 中断的基本概念.....	123	8.1.3 上机步骤.....	206
5.4.2 8086/8088 的中断系统.....	129	8.1.4 实验程序代码.....	211
5.5 可编程中断控制器 8259A.....	134	8.2 汇编语言程序设计综合实验.....	212
5.5.1 8259A 的引线及内部结构.....	134	8.2.1 实验目的.....	212
5.5.2 8259A 的工作过程.....	136	8.2.2 实验条件.....	212
5.5.3 8259A 的工作方式.....	137	8.2.3 实验内容.....	212
5.5.4 8259A 的初始化编程.....	141	8.2.4 编程提示.....	212
习题.....	150	8.2.5 上机步骤.....	212
第 6 章 常用数字接口电路.....	152	8.2.6 调试提示.....	213
6.1 可编程定时计数器 8253.....	152	8.2.7 实验程序与框图.....	214
6.1.1 8253 的外部引脚和内部结构.....	152		
6.1.2 8253 的控制字.....	154		

8.3 并行接口实验	216	8.4.2 实验设备.....	223
8.3.1 实验目的	216	8.4.3 实验内容.....	223
8.3.2 实验设备	216	附录 A ASCII 码表	228
8.3.3 实验内容	217	附录 B 8086/8088 指令简表	230
8.3.4 实验程序与框图	220	附录 C DOS 系统功能调用简表	234
8.4 A/D 与 D/A 转换实验	223	附录 D 教学大纲及说明	239
8.4.1 实验目的	223		

第 1 章 微型计算机基础

当今时代, 微型计算机已成为帮助我们认识和改造世界的重要工具。那么现实世界各种信息如何在计算机中表示和处理、计算机如何辅助我们完成各种任务呢? 通过本章的学习, 可以了解到数和字符在计算机中的表示方法; 二进制数的运算; 微型计算机的基本结构和工作原理; 8088CPU 的内部结构、外部引脚和工作时序。

1.1 计算机中的数制

1.1.1 常用计数制

1. 数制的概念

数制是人们利用符号来计数的科学方法。

2. 十进制

日常生活中, 人们习惯于使用十进制数。它由 0~9 共 10 个符号构成, 如 883。十进制数的计数规则是逢 10 进 1。

但同一符号在不同的位置代表的值是不同的, 如 883 百位上的 8 表示 800, 十位上的 8 表示 80, 所以十进制数可按位展开成式 1-1:

$$\begin{aligned}(D)_{10} &= D_{n-1} \times 10^{n-1} + D_{n-2} \times 10^{n-2} + \cdots D_1 \times 10^1 + D_0 \times 10^0 + D_{-1} \times 10^{-1} + \cdots D_{-m} \times 10^{-m} \\ &= \sum_{i=-m}^{n-1} D_i \times 10^i\end{aligned}\quad (1-1)$$

这里引出两个概念: 基、权。

数制所使用符号的个数称为基。符号所在位的单位值称为该位的权, 它等于 (基)ⁱ, i 代表符号所在位。十进制数的基为 10, 第 i 位上的权值为 10ⁱ。式 (1-1) 表示的就是各符号与其所在位权值乘积的代数 and, 也称其为权表达式。

十进制数可用后缀 D 标识, 如 883D。此后缀也可不写, 如 883 表示十进制数 883。

3. 二进制

在数字计算机中, 使用的是二进制。二进制数由 0、1 两个符号构成, 如 00111100。二进制的计数规则是逢 2 进 1。

二进制数的基为 2, 各位的权值为 2ⁱ。它的权表达式如式 1-2:

$$\begin{aligned}(B)_2 &= B_{n-1} \times 2^{n-1} + B_{n-2} \times 2^{n-2} + \cdots B_1 \times 2^1 + B_0 \times 2^0 + B_{-1} \times 2^{-1} + \cdots B_{-m} \times 2^{-m} \\ &= \sum_{i=-m}^{n-1} B_i \times 2^i\end{aligned}\quad (1-2)$$

二进制数用后缀 B 标识, 如 00111100B。

计算机中采用二进制的因素如下所示：

(1) 二进制只有 0 和 1 两个状态，显然制造具有两种状态的电子器件要比制造具有 10 种特定状态的器件容易得多，如图 1-1 所示。而且由于状态简单，其工作更可靠，传输也不容易出错。

(2) 二进制运算规则简单，可以使运算器的结构简化，并简化控制机构。

(3) 0、1 数码与逻辑代数变量值 0 与 1 吻合，所以二进制还可以使计算机方便地进行逻辑运算。

(4) 二进制和十进制数之间的关系也不复杂。

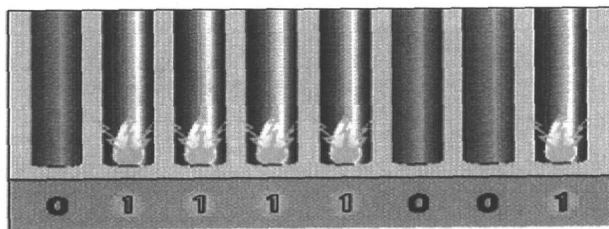


图 1-1 计算机中可用电路有/无电脉冲表示 1/0

4. 其他常用进制

用二进制数表示一个较大的数时，既冗长又难以记忆，为了阅读和书写方便，有时也采用十六进制。

十六进制数由 0~9，A~F 16 个符号构成，如 5A7C。它的计数规则是逢 16 进 1。

十六进制数的基为 16，各位的权值为 16^i 。它的权表达式如式 1-3：

$$\begin{aligned}(H)_{16} &= H_{n-1} \times 16^{n-1} + H_{n-2} \times 16^{n-2} + \cdots + H_1 \times 16^1 + H_0 \times 16^0 + H_{-1} \times 16^{-1} + \cdots + H_{-m} \times 16^{-m} \\ &= \sum_{i=-m}^{n-1} H_i \times 16^i\end{aligned}\quad (1-3)$$

十六进制数用后缀 H 标识，如 5A7CH。

1.1.2 各种数制之间的转换

既然存在多种数制，必然产生各种数制之间的转换问题，如图 1-2 所示。

1. 非十进制数转换成十进制数

方法：按相应的权表达式展开，再按十进制运算规则求和。

【例 1-1】将二进制数 1010.11B 转换成十进制数。

$$\begin{aligned}(1010.11)_2 &= 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} \\ &= 8 + 0 + 2 + 0 + 0.5 + 0.25 = (10.75)_{10}\end{aligned}$$

【例 1-2】将十六进制数 2AE.4H 转换成十进制数。

$$\begin{aligned}(2AE.4)_{16} &= 2 \times 16^2 + A \times 16^1 + E \times 16^0 + 4 \times 16^{-1} \\ &= 2 \times 256 + 10 \times 16 + 14 \times 1 + 4/16 \\ &= (686.25)_{10}\end{aligned}$$

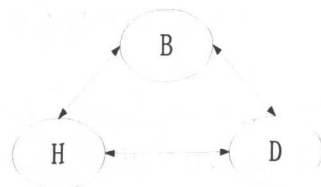


图 1-2 数制转换

2. 十进制数转换成非十进制数

方法：十进制数转换为非十进制数分成两个部分，整数部分和小数部分，即分开转换后再以小数点为结合点组合起来。

整数部分：除基取余，直至商为 0，余数按先后顺序从低位到高位排列。

小数部分：乘基取整，直至达到所需精度或小数部分为 0，整数按先后顺序从高位到低位排列。

【例 1-3】将十进制数 112.25 转换成二进制数。

整数部分	余数		整数	小数部分
2 112	0 ← 最低位		最高位 → 0	0.25 * 2
2 56	0			.5 * 2
2 28	0		1	.0
2 14	0			
2 7	1			
2 3	1			
2 1	1			
0		1110000		

答案：结果为 $(11000.01)_2$ 或 $11000.01B$

【例 1-4】将十进制数 301.6875 转换成十六进制数。

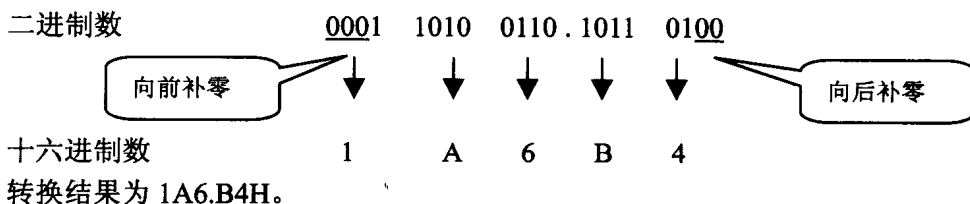
整数部分	余数		整数	小数部分
16 301	13 ← 最低位		最高位 → 11	0.6875 * 16
16 18	2			.0
16 1	1			
0				
		$(13)_{10} = (D)_{16}$		$(11)_{10} = (B)_{16}$
		12D		.B

答案：结果为 $(12D.B)_{16}$ 或 $12D.BH$

3. 二进制数与十六进制数互换

方法：因为 $2^4=16$ ，所以以小数点为分界点，向左向右每 4 位二进制位与一位十六进制位对应。

【例 1-5】将二进制数 $110100110.101101B$ 转换成十六进制数。



1.2 二进制数的运算

1.2.1 无符号数和有符号数的表示

在日常生活中，我们会遇到两种类型的数：无符号数，如地址号；有符号数，比如年收益。计算机里对无符号数和有符号数的表示方法是有差别的。

1. 无符号数的表示

无符号数是将二进制数的所有位都用来表示无符号数的数值大小。

n 位二进制数可表示的无符号数的范围为： $0 \leq X \leq 2^n - 1$ 。如 8 位二进制数表示的无符号数的范围为 0~255，00000000B 表示 0，11111111B 表示 255。

2. 有符号数的表示

有符号数存在正、负号。所以一般用二进制数的最高位代表符号，该位也称符号位，其余位表示其数值部分。符号位为 0 代表正、符号位为 1 代表负。

数在机器中的表示称为机器数，其对应的实际值称为该机器数对应的真值。

有符号数在机器中有 3 种表示方法：原码、反码、补码。

(1) 原码。

表示方法：无论数的正负，机器数的数值部分为真值的绝对值。

n 位原码可表示数的范围为： $-(2^{n-1} - 1) \sim +(2^{n-1} - 1)$ 。

【例 1-6】 $[+42]_{\text{原}} = 0\ 0101010$ $[-42]_{\text{原}} = 1\ 0101010$

【例 1-7】 $[+0]_{\text{原}} = 0\ 0000000$ $[-0]_{\text{原}} = 1\ 0000000$

注意： $[X]_{\text{原}}$ 表示真值 X 的原码。本节中的机器数均用 8 位二进制数表示。

(2) 反码。

表示方法：正数同原码，即数值部分为真值的绝对值。负数的数值部分为真值绝对值按位取反。

n 位反码可表示数的范围为： $-(2^{n-1} - 1) \sim +(2^{n-1} - 1)$ 。

【例 1-8】 $[+42]_{\text{反}} = 0\ 0101010$ $[-42]_{\text{反}} = 1\ 1010101$

【例 1-9】 $[+0]_{\text{反}} = 0\ 0000000$ $[-0]_{\text{反}} = 1\ 1111111$

注意： $[X]_{\text{反}}$ 表示真值 X 的反码。

(3) 补码。

表示方法：正数同原码，即数值部分为真值的绝对值。负数的数值部分为真值绝对值按位取反再加 1。

n 位反码可表示数的范围为： $-2^{n-1} \sim +(2^{n-1} - 1)$ 。

【例 1-10】 $[+42]_{\text{补}} = 0\ 0101010$ $[-42]_{\text{补}} = 1\ 1010110$

【例 1-11】 $[0]_{\text{补}} = 0\ 0000000$ $[-128]_{\text{补}} = 1\ 0000000$

注意: $[X]_{\text{补}}$ 表示真值 X 的补码。

表 1-1 列出了 8 位和 16 位二进制数原码、反码和补码所能表示的有符号数的范围。

表 1-1 8/16 位二进制数可表示的有符号数范围

	8 位	16 位
原码	- 127 ~ +127	- 32767 ~ +32767
反码	- 127 ~ +127	- 32767 ~ +32767
补码	- 128 ~ +127	- 32768 ~ +32767

原码表示法简单易懂, 与真值转换方便, 为什么还要引入反码和补码的概念呢?

设计补码表示法的目的是:

(1) 使符号位能和有效数值部分一起参加数值的运算, 从而简化运算规则, 节省运算时间。

(2) 使减法运算转化成加法运算, 从而进一步简化计算机中运算器的线路设计。

(3) 设计反码表示法的目的则是为了引出补码表示法。

计算机中基本的运算有两种: 算术运算(加、减、乘、除)和逻辑运算(与、或、非、异或等)。下面逐一介绍计算机中这两种运算所采用的运算规则。

1.2.2 算术运算

1. 加/减法运算

规则: $[X+Y]_{\text{补}} = [X]_{\text{补}} + [Y]_{\text{补}}$, $[X-Y]_{\text{补}} = [X]_{\text{补}} + [-Y]_{\text{补}}$

特点:

(1) 采用补码形式完成加、减法。

(2) 可实现将减法转换为加法, 所以只用设置加法器就可完成加、减运算。

(3) 符号位被当作数的一部分参加运算。

(4) 因为计算机字长有限, n 位计算机是以 2^n 为模进行加法, 最高位若产生进位, 则自然丢失。

【例 1-12】计算 $64 - 10$ (用 8 位二进制数表示)。

若用原码做减法:

$$64 - 10 = 01000000\text{B} - 00001010\text{B} = 00110110\text{B}$$

若采用补码运算规则:

$$[64 - 10]_{\text{补}} = [64]_{\text{补}} + [-10]_{\text{补}} = 01000000\text{B} + 11110110\text{B} = \boxed{1}00110110\text{B}$$

在字长为 8 位的机器中, 从第 7 位的进位是自然丢失的, 故用原码做减法与用补码做加法的结果是相同的。

2. 关于溢出

由于计算机的字长有一定的限制, 所以它只能表示一定范围内的数, 当运算结果超出这个表达范围时, 便产生溢出。

计算机是怎样进行溢出判断的呢？令数值部分最高位向符号位的进位为 C_s ，符号位向更高位的进位为 C_{s+1} ，若 C_s 和 C_{s+1} 相异，则运算发生溢出。计算机用异或电路即可实现溢出判断。

运算结果若发生溢出，则会产生溢出中断，CPU 将对其进行处理。

【例 1-13】 $01111111 (+127) + 00000101 (+5) = 10000100 (-124)$ ，

$C_s=1$ ， $C_{s+1}=0$ ， C_s 和 C_{s+1} 相异，发生溢出，运算结果错误。

【例 1-14】 $10000100 (-124) + 11111011 (-5) = 01111111 (+127)$ ，

$C_s=0$ ， $C_{s+1}=1$ ， C_s 和 C_{s+1} 相异，发生溢出，运算结果错误。

3. 乘/除法运算

实现乘、除运算，一般有两种方法：一种是用程序来实现，另一种是采用硬件直接完成（即由执行乘/除法指令实现），但是两种方法运算的基本原理是一致的。

在乘除运算中，由于同号相乘（除）为正，异号相乘（除）为负，因此在计算机中对二进制数符号位异或，即可确定乘积（或商）的符号。所以乘除运算一般采用原码形式，即数值部分相乘（除），得乘积（商）的数值部分，符号位做异或运算，得乘积（商）的符号位。

下面以具体的例子来说明计算机中数值部分相乘除的运算规则。

(1) 乘法运算。

$$\begin{array}{r}
 1100 \quad \text{被乘数} \\
 \times 1001 \quad \text{乘数} \\
 \hline
 1100 \\
 0000 \quad \text{0 或被移位的被乘数} \\
 0000 \\
 1100 \\
 \hline
 1101100 \quad \text{乘积}
 \end{array}$$

与十进制数乘法运算类似，二进制乘法是由乘数的每一位和被乘数相乘，因为乘数每一位的值只可能是 0 和 1，所以每次相乘得到的或者是 0 或者是移位的被乘数。

不过，在计算机中并不是列出所有移位的被乘数再相加，而是采用将每一次得到的移位的被乘数和部分积相加的办法做乘法。用这种方法再次重复上述例子，如表 1-2 所示。

表 1-2 用部分积的方法做乘法

乘数	运算说明	被乘数	部分积
1 0 0 1		1100	0000
1	乘数为 1，加被乘数到部分积上，并将被乘数左移一位	11000	1100
0	乘数为 0，不加被乘数，被乘数左移一位	110000	1100
0	乘数为 0，不加被乘数，被乘数左移一位	1100000	1100
1	乘数为 1，加左移后的被乘数到部分积上，被乘数左移一位	11000000	1101100
即：1100B*1001B=1101100B		移位	加法

由上表可见，在微型机中乘法运算可转换成加法和移位运算。在计算机里进行移位可采用移位器。

(2) 除法运算。

二进制数的除法与十进制数的除法也是类似的。首先从被除数的最高位开始检查，并定出需要大于或等于除数的位数，找到这个位时商记为1，并把选定的被除数减除数，然后把被除数的下一位移到余数上。如果余数不能减去除数（不够减），则商上0，把被除数的再下一位移到余数上；若余数够减，则商上1，余数减去除数，然后把被除数的下一位移到余数上，继续求商的过程。

$$\begin{array}{r}
 000111 \\
 101 \overline{)100011} \\
 \underline{-101} \\
 0111 \\
 \underline{-101} \\
 101 \\
 \underline{-101} \\
 0
 \end{array}$$

以上运算过程在计算机中实现，可归纳为如下步骤：

步骤1，余数（被除数）左移移位。

步骤2，余数大于除数吗？如果余数大于除数，则商上1，且余数-除数=新余数；如果余数小于除数，则商上0，且余数=新余数。

重复上述规律，每次进行移位、比较、上商，对应n位被除数，则重复进行n次运算。上例过程如表1-3所示。在微型机中除法运算则可转换成减法和移位运算。

表1-3 除法操作示例

被除数 1 0 0 0 1 1	除数 101	商	余数
1	1<101, 新余数=1	0	1
10	10<101, 新余数=10	00	10
100	100<101, 新余数=100	000	100
1000	1000>101, 新余数=1000-101=11	0001	11
111	111>101, 新余数=111-101=10	00011	10
101	101>=101, 新余数=101-101=0	000111	0

1.2.3 逻辑运算

逻辑运算是按位进行的，每一位的运算都是独立的。

1. 与运算 (AND, ∧)

规则：对应位上的状态为全“1”时，该位上与运算结果才为“1”，否则为“0”。

【例1-15】 $11000011 \wedge 00101001 = 00000001$

2. 或运算 (OR, ∨)

规则：对应位上的状态为全“0”时，该位上与运算结果才为“0”，否则为“1”。

【例1-16】 $11000011 \vee 00101001 = 11101001$

3. 非运算 (NOT, $\neg$)

规则：若某位状态为“1”，则该位非运算结果为“0”；若某位状态为“0”，则该位非运算结果为“1”。

【例 1-17】 $\overline{11000011} = 00111100$

4. 异或运算 (XOR, $\oplus$)

规则：对应位上的状态相异时，该位上异或运算结果才为“1”，否则为“0”。

【例 1-18】 $11000011 \oplus 00101001 = 11101010$

1.3 二进制编码

1.3.1 二进制编码的概念

我们希望计算机不仅能按二进制计数，还应能方便地处理在日常生活中广泛存在的十进制数、字符、文字、声音、图像等信息，但计算机只能识别二进制数，所以可采用若干位 0、1 的组合来对前述各种信息进行编码，这就称为二进制编码。

1.3.2 十进制数的二进制编码

要对十进制数进行二进制编码是因为我们在日常生活中普遍采用的就是十进制数，所以希望计算机能比较直观地存储十进制数，并能实现按十进制运算规则进行的计算。

1. 8421BCD 码

用二进制编码表示的十进制数，称为二—十进制码，简称 BCD 码 (Binary Coded Decimal)。

BCD 码的编码方案有多种，最常用的一种是 8421BCD 码。它是用 4 位比特对一位十进制数进行编码 (如表 1-4 所示)。这 4 个比特都有特定的权，从左至右分别为：8、4、2、1，故称其为 8421 码。

表 1-4 十进制数的 8421BCD 码

十进制数	8421 码	十进制数	8421 码
0	0000	5	0101
1	0001	6	0110
2	0010	7	0111
3	0011	8	1000
4	0100	9	1001

2. 8421BCD 码与十进制、二进制数的转换

十进制数与 8421BCD 码按表 1-4 对应转换即可。

二进制数与 8421BCD 码以十进制数作为中介即可进行转换。

【例 1-19】试把十进制数 321.57 写成 8421 码的形式。