



21st CENTURY

实用规划教材

21世纪全国高职高专
电子信息系列实用规划教材

微机原理 及接口技术

主 编 王用伦 张开成

副主编 屈芳升 张小义



北京大学出版社
PEKING UNIVERSITY PRESS

21 世纪全国高职高专电子信息系列实用规划教材

微机原理及接口技术

主 编 王用伦 张开成
副主编 屈芳升 张小义
参 编 亢娟娜 陈昕志 李 纯



北京大学出版社
PEKING UNIVERSITY PRESS

内 容 简 介

本书是根据高等职业技术学院的教学要求编写的。在介绍计算机基础知识的基础上,以 Intel 8086 微处理器为重点,以 Intel 80X86 微处理器组成的 PC 系列微机为背景,结合微机发展的新知识、新技术,系统地介绍了微机的组成原理、接口技术和典型应用。

全书共分 11 章,首先介绍了计算机基础知识,然后分别介绍了微型计算机结构、指令系统、汇编语言程序设计、存储器结构、微机常用总线系统、中断系统等微机组成原理。并在此理论基础上,对并行接口技术,串行接口技术,中断和定时计数、显示器、键盘接口技术进行了介绍。为了帮助读者理解和掌握所学知识,每章后都有思考与练习题。同时,还结合理论知识给出了实训,帮助读者提高实际动手能力。

本书可作为高职高专教育相关专业微机原理与接口技术课程的教材,也可以作为本科生学习微机原理与接口技术的参考书,还可以作为从事微机关、硬件工作的工程技术人员的参考书。

图书在版编目(CIP)数据

微机原理及接口技术/王用伦,张开成主编. —北京:北京大学出版社, 2009.4

(21 世纪全国高职高专电子信息系列实用规划教材)

ISBN 978-7-301-12385-0

I. 微… II. ①王…②张… III. ①微型计算机—理论—高等学校:技术学校—教材②微型计算机—接口—高等学校:技术学校—教材 IV.TP36

中国版本图书馆 CIP 数据核字(2007)第 083347 号

书 名: 微机原理及接口技术

著作责任者: 王用伦 张开成 主编

策划编辑: 赖 青

责任编辑: 魏红梅

标准书号: ISBN 978-7-301-12385-0/TM · 0009

出 版 者: 北京大学出版社

地 址: 北京市海淀区成府路 205 号 100871

网 址: <http://www.pup.cn> <http://www.pup6.com>

电 话: 邮购部 62752015 发行部 62750672 编辑部 62750667 出版部 62754962

电子邮箱: pup_6@163.com

印 刷 者: 世界知识印刷厂

发 行 者: 北京大学出版社

经 销 者: 新华书店

787 毫米×1092 毫米 16 开本 18.75 印张 433 千字

2009 年 4 月第 1 版 2009 年 4 月第 1 次印刷

定 价: 29.00 元

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有 侵权必究

举报电话: 010-62752024

电子邮箱: fd@pup.pku.edu.cn

前 言

本书是根据高等职业技术学院的教学要求编写的, 突出了“基本理论够用为度, 应用为重点”的高职教育特点, 结合微机发展的新技术、新知识, 注重学生动手能力的培养。

本书以 Intel 8086 微处理器为重点, 以 Intel 80X86 微处理器组成的 PC 系列微机为背景, 结合微机发展的新知识、新技术, 系统地介绍了微机的组成原理、接口技术和典型应用。全书共分为 11 章。第 1 章介绍了微型计算机基础知识; 第 2 章以 Intel 8086 微处理器为重点, 对 Intel 80X86 微处理器及微机结构进行了介绍; 第 3 章介绍了 8086 指令系统; 第 4 章介绍了汇编语言程序设计; 第 5 章介绍了微机的存储器结构, 其中包括一些存储新技术; 第 6 章重点介绍了目前常用的总线技术; 第 7 章介绍了微型计算机中断及定时/计数技术; 第 8 章介绍了微型计算机并行接口技术; 第 9 章介绍了微型计算机串行接口技术; 第 10 章介绍了微型计算机显示器及键盘接口技术; 第 11 章为综合实训, 通过软件仿真和温度控制实训项目, 提高学生对微机原理和接口技术的实际应用能力。本书的前半部分主要介绍微机的组成原理, 后半部分主要介绍微机的接口技术。接口技术部分主要围绕各种可编程芯片及其应用进行介绍, 既对硬件接口的具体连接进行讲解, 又结合应用进行了软件编程。为了帮助读者理解和掌握所学知识, 每章后面都有思考与练习题。

本书在编写过程中, 力求既注重微机基本知识与典型应用的结合, 又注重基本知识与最新知识的联系, 尽量把最新技术介绍给读者。同时, 还结合每章的理论知识给出了具体的实训项目, 提出了具体要求, 帮助学生提高实际动手能力。最后, 通过综合实训帮助学生建立起微机应用的整体概念, 提高分析解决实际问题的能力。

本书建议授课的总学时为 50, 各章内容及学时分配见下表。

章 节	授课学时	章 节	授课学时
第 1 章 微型计算机系统概述	4	第 6 章 总线	4
第 2 章 微型计算机结构	6	第 7 章 微型计算机中断系统及定时/计数器应用	4
第 3 章 指令系统	6	第 8 章 微型计算机 I/O 接口技术及应用	4
第 4 章 汇编语言程序设计	10	第 9 章 微型计算机串行接口技术及应用	4
第 5 章 存储器结构	4	第 10 章 显示器及键盘接口技术	4

实训项目可结合各学校的实训条件和学习内容, 另外安排学时进行。

本书第 1 章、第 6 章由重庆航天职业技术学院的张开成编写, 第 2 章、第 11 章由重庆航天职业技术学院的王用伦编写, 第 3 章由甘肃畜牧工程职业技术学院的张小义编写, 第 4 章由甘肃畜牧工程职业技术学院的亢娟娜编写, 第 5 章、第 9 章由河南职业技术学院的陈昕志编写, 第 7 章、第 8 章由河南职业技术学院的屈芳升编写, 第 10 章由重庆航天职业技术学院的李纯编写。本书由王用伦、张开成任主编, 由王用伦负责统稿。

本书参考了有关微机原理与接口技术方面的大量书刊资料, 引用了部分参考文献的内

容，在此谨向这些书刊资料的作者表示衷心的感谢！

由于编者的认识和专业水平有限，加之时间比较仓促，书中不足之处在所难免，敬请广大读者批评指正。

编 者
2009年1月

目 录

第 1 章 微型计算机系统概述	1	2.3.1 80X86 系列微处理器概述	37
1.1 微型计算机的发展史	1	2.3.2 80486 微处理器简介	38
1.2 计算机中的数和编码	2	2.4 奔腾(Pentium)系列微处理器	41
1.2.1 计算机中的数制	2	2.4.1 奔腾(Pentium)微处理器	41
1.2.2 符号数的表示	3	2.4.2 Pentium Pro 微处理器	44
1.2.3 二进制数的运算	5	2.4.3 MMX Pentium 微处理器	44
1.2.4 二进制编码	8	2.4.4 Pentium II 微处理器	45
1.3 微型计算机系统	11	2.4.5 Pentium III 微处理器	46
1.3.1 微机系统组成	11	2.4.6 Pentium 4 微处理器	47
1.3.2 微型计算机系统的主要技术 指标	12	本章小结	48
1.4 微型计算机的工作过程	14	思考与练习题	48
1.4.1 指令与程序的执行	14	实训 微处理器的认识与理解	49
1.4.2 程序执行过程举例	15	第 3 章 指令系统	50
本章小结	16	3.1 指令与指令系统	50
思考与练习题	17	3.2 8086 指令系统的寻址方式	50
第 2 章 微型计算机的结构	18	3.2.1 指令的基本格式	50
2.1 微型计算机的基本结构	18	3.2.2 8086 指令系统的基本寻址 方式	51
2.1.1 微处理器	18	3.3 8086 指令系统	55
2.1.2 存储器	18	3.3.1 传送类指令	55
2.1.3 输入/输出接口电路	19	3.3.2 算术运算类指令	61
2.1.4 总线	19	3.3.3 位操作指令	71
2.2 8086/8088 微处理器	19	3.3.4 串操作类指令	76
2.2.1 8086/8088 微处理器的 内部结构	20	3.3.5 循环和转移指令	81
2.2.2 8086/8088 微处理器的 寄存器结构	22	3.3.6 处理器控制指令	88
2.2.3 8086 微处理器的引脚及其 功能	25	本章小结	91
2.2.4 8086 微处理器的工作模式	27	思考与练习题	91
2.2.5 8086 微处理器的总线周期	30	实训 8086 基本指令训练	94
2.2.6 8086 系统的存储器管理	33	第 4 章 汇编语言程序设计	100
2.3 80X86 系列微处理器	37	4.1 汇编语言	100
		4.1.1 汇编语言的基本概念	100
		4.1.2 汇编语言源程序的格式	100

4.2 汇编语言的语句	101	思考与练习题	167
4.2.1 指令语句	102	实训 存储器扩展	167
4.2.2 伪指令语句	106	第 6 章 总线	170
4.2.3 宏指令语句	112	6.1 总线概述	170
4.3 汇编语言程序设计	114	6.1.1 总线的类别	170
4.3.1 程序设计的基本方法	114	6.1.2 总线的优点	171
4.3.2 顺序程序设计	115	6.2 系统总线	171
4.3.3 分支程序设计	117	6.2.1 ISA 总线	172
4.3.4 循环程序设计	121	6.2.2 STD 总线	173
4.3.5 子程序设计	126	6.2.3 PCI 总线	173
4.3.6 DOS 系统功能调用	130	6.2.4 AGP 总线	175
4.4 汇编语言程序的上机过程及调试	134	6.3 外部总线	177
4.4.1 编辑汇编语言源程序	134	6.3.1 RS232C 总线	178
4.4.2 汇编语言源程序	135	6.3.2 SCSI 总线	182
4.4.3 连接程序	136	6.3.3 USB 总线	182
4.4.4 程序的执行	137	6.3.4 串行总线 IEEE-1394	185
本章小结	137	本章小结	188
思考与练习题	137	思考与练习题	188
实训 程序设计	141	第 7 章 微型计算机中断系统及定时/	
第 5 章 存储器结构	145	计数器应用	189
5.1 半导体存储器	145	7.1 中断系统概述	189
5.1.1 存储器概述	145	7.1.1 中断的基本概念	189
5.1.2 半导体存储器的分类	146	7.1.2 中断源的分类	189
5.2 随机存储器	147	7.1.3 中断优先级和中断向量表	191
5.2.1 静态 RAM(SRAM)	147	7.1.4 中断处理过程	192
5.2.2 动态 RAM(DRAM)	150	7.2 可编程中断控制器 8259A 及其	
5.2.3 几种新型 RAM 技术	152	应用	195
5.3 只读存储器	153	7.2.1 8259A 的引脚与内部结构	195
5.3.1 ROM 的分类	153	7.2.2 8259A 的编程及应用	198
5.3.2 ROM 的应用	154	7.3 8253 可编程定时/计数器及其应用	206
5.4 存储器与 CPU 的连接	155	7.3.1 8253 可编程定时/计数器	206
5.4.1 连接中应考虑的问题	155	7.3.2 8253 的编程和工作方式	208
5.4.2 存储器与 CPU 的连接	156	7.3.3 8253 的应用	210
5.5 PC 的存储器结构	161	本章小结	212
5.5.1 PC 存储系统的层次结构	161	思考与练习题	213
5.5.2 内存管理	162	实训 微型计算机计数器/定时器	213
5.5.3 DIMM 内存部件	163		
本章小结	166		

第 8 章 微型计算机 I/O 接口技术及应用216	9.2.2 8251A 的编程..... 242
8.1 微型计算机 I/O 接口.....216	9.2.3 8251A 的应用..... 244
8.1.1 I/O 接口.....216	本章小结..... 247
8.1.2 I/O 端口的编址方法.....217	思考与练习题..... 247
8.1.3 I/O 接口数据的传送方式.....218	实训 串行接口..... 248
8.2 可编程并行 I/O 接口芯片 8255A 及其应用.....221	第 10 章 显示器及键盘接口技术 251
8.2.1 并行接口的基本概念.....221	10.1 显示器及其接口..... 251
8.2.2 8255A 的结构和功能.....222	10.1.1 LED 显示器及其接口..... 251
8.2.3 8255A 的控制字和工作方式.....223	10.1.2 LCD 显示器及其接口..... 256
8.2.4 8255A 的编程及应用.....229	10.2 键盘及其接口..... 260
本章小结.....231	10.2.1 独立式键盘及其接口..... 261
思考与练习题.....231	10.2.2 矩阵式键盘及其接口..... 261
实训 并行接口.....232	10.3 可编程键盘/显示器接口 8279..... 264
第 9 章 微型计算机串行接口技术及应用234	10.3.1 8279 的结构..... 265
9.1 串行通信的基本概念.....234	10.3.2 8279 的引脚定义..... 266
9.1.1 串行通信的连接方式.....234	10.3.3 8279 的编程..... 268
9.1.2 信号的调制与解调.....235	10.3.4 8279 的应用..... 271
9.1.3 同步与异步通信方式.....235	本章小结..... 273
9.1.4 波特率与收/发时钟.....237	思考题及习题..... 273
9.2 可编程串行接口芯片 8251A.....237	实训 键盘扫描显示..... 274
9.2.1 8251A 引脚及内部结构.....238	第 11 章 综合实训 275
	11.1 软件仿真——应用程序的建立与调试..... 275
	11.2 空调温度的控制..... 277
	参考文献 289

第 1 章 微型计算机系统概述

【教学提示】 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、体系结构以及工作过程。

【教学目标】 了解计算机的发展史，理解微型计算机的系统结构、硬件组成及基本工作过程，掌握计算机中的数和编码方法。

【教学重点】 微型计算机的系统结构、硬件组成及基本工作过程，计算机中的数和编码方法。

1.1 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、性能和价格分为巨型机、大型机、中型机、小型机和微型机 5 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，因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 年代初第一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就提高一倍，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 $4.2\text{mm} \times 3.2\text{mm}$ 的硅片上集成了 2 250 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 微处理器为基础，再配以相应的 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，微处理器的发展就经历了以下几个时期。

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 2 000 个晶体管/片，采用 PMOS 工艺， $10\mu\text{m}$ 光刻技术，时钟频率仅 1MHz，平均执行指令时间约为 $20\mu\text{s}$ 。

第二代包括 1974—1978 年 Intel 公司推出的 8080/8085、Zilog 公司推出的 Z80、Motorola 公司的 MC6800/6802 等。这一代高档 8 位微处理器的特点是采用 NMOS 工艺，集成度达到 8 400 个晶体管/片以上，时钟频率为 $2 \sim 4\text{MHz}$ ，平均执行指令时间约为 $1 \sim 2\mu\text{s}$ 。这时的微处理器设计和生产技术已相当成熟，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 等。它们采用了 HMOS 工艺, 集成度为 20 000~60 000 个晶体管/片, 时钟频率为 4~8MHz, 平均执行指令时间为 0.5 μ s。

第四代(1980—1992 年)以 32 位微型机为代表。典型产品有 Intel 的 80386/80486、Motorola 的 MC68020、Zilog 的 Z80000 等。这一代微处理器的时钟频率为 16~100MHz, 集成度为 10^5 个晶体管/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 $5.0 \times 10^6 \sim 9.3 \times 10^6$ 个晶体管/片, 时钟频率达 150~300MHz。

1995 年 Pentium II 问世, 1999 年 Pentium III 诞生。目前市场上的主流产品已由 Pentium 4 取代了 Pentium III。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不可估量, 其应用也越来越广泛。

1.2 计算机中的数和编码

1.2.1 计算机中的数制

1. 常用进位计数制

1) 十进制

十进制数是人们最熟悉的, 但机器实现起来困难。

2) 二进制

由于二进制数每位只需两个状态“0”或“1”, 机器实现容易, 因而二进制数是数字系统唯一认识的代码, 但其书写太长。

3) 八进制

$2^3=8$, 因而 3 位二进制数可用一位八进制数表示。

4) 十六进制

$2^4=16$, 因而 4 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, 十进制主要用于输出运算的最终结果。

为了使书写的进位计数制数便于区分, 十进制数用 D 表示或省略, 二进制数用 B 表示, 八进制数用 Q 表示, 十六进制数用 H 表示。

2. 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

既有整数部分又有小数部分的十进制数转换成二进制数, 首先要将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：连续乘以 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。方法是：以二进制数的小数点为起点，分别向左、向右将每 3 位(或 4 位)分成一组。对于小数部分，最低位一组不足 3 位(或 4 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数(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

八进制数(或十六进制数)转换成二进制数时，只需把八进制数(或十六进制数)的每一位数码分别转换成 3 位(或 4 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。

1.2.2 符号数的表示

1. 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。机器数应用字节、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为带符号数的原码表示法，所表示的数称原码。

$$y_1 = +127 = +1111111B \quad [y_1]_{\text{原}} = 01111111B$$

$$y_2 = -127 = -1111111B \quad [y_2]_{\text{原}} = 11111111B$$

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 0FFH~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 0FFFFH~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”。

$$y1=+127=+11111111B \quad [y1]_{\text{反}}=01111111B$$

$$y2=-127=-11111111B \quad [y2]_{\text{反}}=10000000B$$

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”。

$$y1=+127=+11111111B \quad [y1]_{\text{补}}=01111111B$$

$$y2=-127=-11111111B \quad [y2]_{\text{补}}=10000001B$$

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。

2. 原码、反码、补码和真值的相互转换关系

前面已经讨论了符号数的真值表示和在机器中的表示，下面总结机器数和真值的相互转换关系。

正数的原码、反码、补码相同，即用正号“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。

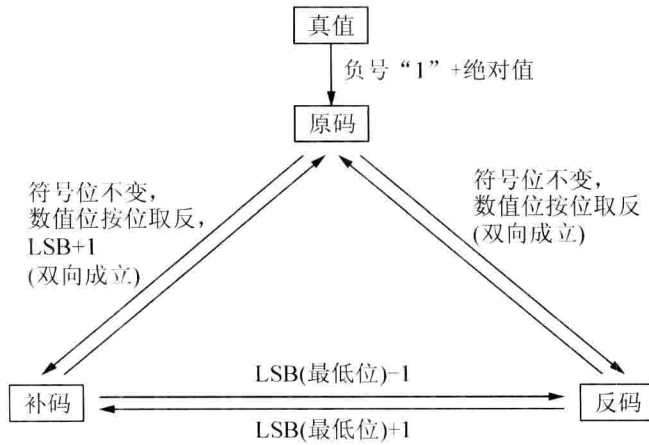


图 1.1 负数的原码、反码、补码和真值的相互转换关系

下面以负数情况为例来进行分析，以说明符号数的机器表示和真值的相互转换关系。

1) 已知原码，求补码

【例 1.1】 已知某数 X 的原码为 10110100B，试求其补码。

【解】 由[X]_原=10110100B，知 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。

$$\begin{array}{r}
 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0 \quad \text{原码} \\
 \downarrow\downarrow\downarrow\downarrow\downarrow\downarrow\downarrow \\
 1\ 1\ 0\ 0\ 1\ 0\ 1\ 1 \quad \text{符号位不变, 数值位取反} \\
 + \qquad\qquad\qquad 1 \quad +1 \\
 \hline
 1\ 1\ 0\ 0\ 1\ 1\ 0\ 0 \quad \text{补码} \\
 \therefore [X]_{\text{补}}=11001100\text{B}
 \end{array}$$

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 $[X]_{\text{补}}=11101110\text{B}$, 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。

$$\begin{array}{r}
 1\ 1\ 1\ 0\ 1\ 1\ 1\ 0 \quad \text{补码} \\
 \downarrow\downarrow\downarrow\downarrow\downarrow\downarrow\downarrow \\
 1\ 0\ 0\ 1\ 0\ 0\ 0\ 1 \quad \text{符号位不变, 数值位取反} \\
 + \qquad\qquad\qquad 1 \quad +1 \\
 \hline
 1\ 0\ 0\ 1\ 0\ 0\ 1\ 0 \quad \text{原码} \\
 \therefore [X]_{\text{原码}}=10010010\text{B}
 \end{array}$$

3. 求补

已知 $[X]_{\text{补}}$, 求 $[-X]_{\text{补}}$ 的过程称为求补。所谓求补, 就是将 $[X]_{\text{补}}$ 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 $[-X]_{\text{补}}$ 。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 $[+97]_{\text{补}}=01100001\text{B}$

求 $[-97]_{\text{补}}$ 的方法是:

$$\begin{array}{r}
 0\ 1\ 1\ 0\ 0\ 0\ 0\ 1 \quad [+97]_{\text{补}} \\
 \downarrow\downarrow\downarrow\downarrow\downarrow\downarrow\downarrow \\
 1\ 0\ 0\ 1\ 1\ 1\ 1\ 0 \quad \text{逐位取反} \\
 + \qquad\qquad\qquad 1 \quad +1 \\
 \hline
 1\ 0\ 0\ 1\ 1\ 1\ 1\ 1 \quad [-97]_{\text{补}} \\
 \therefore [+97]_{\text{补}}=01100001\text{B}, [-97]_{\text{补}}=10011111\text{B}
 \end{array}$$

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 $[X]_{\text{补}}=10101011\text{B}$, 知 X 为负数, 故符号位不变, 数值部分按“求反加 1”法得到 $[X]_{\text{原}}$ 。

$$[X]_{\text{原}}=11010101\text{B}, X=-1010101\text{B}, X=-85$$

1.2.3 二进制数的运算

1. 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 $0 \sim (2^n - 1)$ 。

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。例如：

$$\begin{array}{r}
 127+150=7FH+96H \\
 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1 \\
 +\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 0 \\
 \hline
 1\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 1=115H=256+16+5=277 \\
 \uparrow \\
 \text{进位}
 \end{array}$$

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。例如：

$$\begin{array}{r}
 128-10=80H-0AH \\
 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \\
 -\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 0 \\
 \hline
 0\ 1\ 1\ 1\ 0\ 1\ 1\ 0=76H=112+6=118
 \end{array}$$

反过来相减，即 $10-128$ ，运算过程如下：

$$\begin{array}{r}
 0\ 0\ 0\ 0\ 1\ 0\ 1\ 0 \\
 -\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \\
 \hline
 1\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 0=-01110110=-118 \\
 \uparrow \\
 \text{借位}
 \end{array}$$

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。

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：

$$[x \pm y]_{补} = [x]_{补} + [\pm y]_{补}$$

补码加法运算规则如下。

- ① 参加运算的数均为补码。
- ② 符号位和数值位一起参加运算，符号位向前的进位自然丢掉，若不发生溢出，留下的结果一定是正确的。
- ③ 运算结果为补码。

④ 若运算结果为负数的补码，若未发生溢出，应转换为原码，这样才能知道结果的真值。

【例 1.5】 用 8 位补码完成下列计算。

$$(1) 63-127 \qquad (2) -67-15$$

【解】

$$\begin{array}{r}
 (1) \quad [63]_{\text{补}}: \quad 00111111 \\
 \quad [-127]_{\text{补}}: + 10000001 \\
 \hline
 \quad [补]: \quad 11000000 \\
 \quad [原]: \quad 11000000 \\
 \quad \therefore 63-127=-64
 \end{array}
 \qquad
 \begin{array}{r}
 [-67]_{\text{补}}: \quad 10111101 \\
 [-15]_{\text{补}}: + 11110001 \\
 \hline
 [1]10101110[补] \\
 \quad 11010010[原] \\
 \quad \therefore -65-15=-82
 \end{array}$$

(2) 补码加减法运算的溢出判断。

两个补码加数相加时，在什么情况下才有可能发生溢出呢？很显然，只有当它们都是正数或负数，补码相加时才有可能发生溢出。

【例 1.6】 两个带符号数 0100001B(+65)和 0100011B(+67)相加如下。

$$\begin{array}{r}
 01000001 \\
 + 01000011 \\
 \hline
 10000100
 \end{array}$$

例 1.6 中是两个正数相加，但相加结果却是一个负数，显然这个结果是错误的，出现这种情况的原因就在于这两个加数相加的结果超过了 8 位二进制带符号数补码所能表示的范围(-128~+127)。

【例 1.7】 再来看两个负数补码 10001000B(-120)和 11101110B(-18)的相加情况。

$$\begin{array}{r}
 10001000 \\
 + 11101110 \\
 \hline
 \boxed{1}01110110
 \end{array}$$

由于规定用 8 位二进制数来表示带符号数，故忽略作为进位位的第 9 位。按 8 位二进制数来解释这两个符号数的相加，其结果为一个正数，很明显，结果是错误的。错误的原因如上所述。

以上两种情况叫做补码运算的溢出。当两个同符号的数据相加时，如果相加的结果超过了微处理器所能表示的数值范围，就将发生溢出，其结果就是错误的。因此，在微处理器中设有专门的电路用以判断运算结果是否产生溢出，并以某种标志告诉人们这次运算的结果是否存在溢出。只要溢出没有发生，运算的结果总是正确的。这个在 8086 微处理器中是用 OF 来标志的。若运算结果发生溢出，则置 OF 为 1；否则 OF 为 0。

在机器中如何判溢出呢？如何置 OF 标志位呢？

设符号位向进位位的进位为 C_V ，数值部分向符号位的进位为 C_S ，则有无溢出的判别式如下。

$$OF=C_V \oplus C_S$$

式中：OF=1 表示有溢出；OF=0 表示无溢出。

【例 1.8】 再来看 $105+50$ 、 $-105-50$ 和 $-50-5$ 的运算结果有无溢出。

$\begin{array}{r} (1) \quad 01101001 \\ + \quad 00110010 \\ \hline 10011011 \\ C_Y=0, C_S=1 \\ OF=0\oplus 1=1, \text{有溢出。} \end{array}$	$\begin{array}{r} (2) \quad 10010111 \\ + \quad 11001110 \\ \hline 101100101 \\ C_Y=1, C_S=0 \\ OF=1\oplus 0=1, \text{有溢出。} \end{array}$	$\begin{array}{r} (3) \quad 11001110 \\ + \quad 11111011 \\ \hline 111001001 \\ C_Y=1, C_S=1 \\ OF=1\oplus 1=0, \text{无溢出。} \end{array}$
---	--	--

2. 二进制数的逻辑运算

1) 逻辑非

逻辑非亦称“求反”。对二进制数进行逻辑非运算，就是按位求它的反。

2) 逻辑与

对两个二进制数进行逻辑与，就是按位求它们的逻辑积。逻辑与又称逻辑“乘”，常用记号“ \wedge ”或“ \cdot ”来表示。

3) 逻辑加

对两个二进制数进行逻辑加，就是按位求它们的“或”，所以逻辑加又称“逻辑或”，常用记号“ \vee ”或“ $+$ ”来表示。

4) 逻辑异或

对两个二进制数进行逻辑异或，就是按位求它们的模 2 和，所以逻辑异或又称“按位加”，常用“ \oplus ”来表示。

注意：按位加与普通整数加法的区别是它仅按位相加，不产生进位。

1.2.4 二进制编码

计算机中除了处理数值信息外，还要处理大量的字符数据，如字母、符号、汉字、图像、语音等。在计算机中常用的字符编码有 BCD(Binary Coded Decimal)码、ASCII 码、格雷码、奇偶校验码等，这些编码在机器内部都是用二进制表示的，所以统称为二进制编码。

1. BCD 码

二进制编码的十进制码(BCD 码)有压缩和非压缩两种存储形式。压缩的 BCD 码用半个字节存放一位十进制数，即一个字节存放两位十进制数；而非压缩的 BCD 码则以一个字节存放一位十进制数。BCD 码在指令中是常用的一种编码。

所谓 BCD 码就是由 4 位二进制数表示一位十进制数，十进制数只有 10 个基数符号，而 4 位二进制数有 16 种取值组合，这样就出现了多种编码方案。下面仅介绍 8421 码、余 3 码和格雷码。这 3 种编码与十进制数的对应关系见表 1-1。

表 1-1 BCD 码与十进制数的对应关系

十进制数据	8421 码	余 3 码	格雷 码
0	0000	0011	0000
1	0001	0100	0001
2	0010	0101	0011
3	0011	0110	0010
4	0100	0111	0110
5	0101	1000	0111
6	0110	1001	0101
7	0111	1010	0100
8	1000	1011	1100
9	1001	1100	1101

1) 8421 码

8421 码是一种最简单的二进制自然编码，它以 4 位二进制数的前 10 个代码分别对应十进制数的 10 个数码，1010~1111 为无效编码。这种编码按权求和，和就是对应的十进制数。这就是说，编码中的每位仍保留着二进制数所具有的位权，而且 4 位代码从左向右的位权依次为 8、4、2、1，8421 码就是由此而命名的。

如十进制数 72 的压缩 BCD 码用 8421 码可表示为 01110010；十进制数 54 对应的编码为 01010100。

2) 余 3 码

余 3 码是无权码，它具有良好的代码校验性。这种编码转换成十进制数后，每个代码的值比相应的十进制数多 3。例如，表 1-1 中的十进制数 4，它对应的二进制代码是 0111=7；十进制数 9 对应的二进制代码是 1100=12，它们都比相应的十进制数大 3。因此，称这种编码为余 3 码。

3) 格雷码

格雷码也是一种无权的 BCD 编码形式，其特点是相邻的两位代码仅有一个码元变化。这种编码的抗干扰能力强，常用于计算机控制机床的角编码器、轴位编码等。

2. ASCII 码

对各个字母和符号编制的代码叫字符代码。字符代码的种类繁多，目前在计算机和数字通信系统中被广泛采用的是 ASCII 码(American Standard Code for Information Interchange, 美国信息交换标准代码)，其编码表见表 1-2。

在表 1-2 中读码时，先读列码 $B_7B_6B_5$ ，再读行码 $B_4B_3B_2B_1$ 即为某字符的 7 位 ASCII 码。例如字母 A 的列码是 100，行码是 0001，因此 A 的 7 位 ASCII 码是 1000001，即 41H。注意表中最左边一列的 A, B, ..., F 是十六进制数的 6 个数码。

目前，ASCII 码有 7 位和 8 位两种字符编码形式。常用的是 7 位 ASCII 码，它包括 26 个大写和小写英文字母、10 个数字以及一些专用字符。7 位编码的 ASCII 码，实际也采用 8 位二进制(一个字节)，但最高位置 0 或用作校验，故最多可表示 128 个字符(即 $2^7=128$)。