



● **新世纪** 网络课程教材

微型计算机 原理及应用

黄道君 刘丹华 杨西惠



高等教育出版社

[http:// www.hep.com.cn](http://www.hep.com.cn)

[http:// www.hep.edu.cn](http://www.hep.edu.cn)

内 容 提 要

本书是为配合教育部新世纪网络课程建设项目——“微型计算机原理及应用”教学而编写的配套教材。内容共分12章。第一章简要介绍学习和应用微型计算机必须具备的数学和数字逻辑电路基础知识。第二~六章以IBM PC/XT系统机为例,阐述8086/8088微处理器的基本原理、系统总线形成、指令系统、汇编语言编程、存储器组织、基本数据传递方式及常用接口电路芯片,其目的在于引导读者对典型16位微机建立起一个完整的认识。第七章从改善16位微机性能出发,简要介绍了总线裁决器8289、数字协处理器8087、输入/输出处理器8089、微处理器的升级芯片80286、80386、80486及高速缓冲存储器Cache,其目的一方面是扩展读者对相关新技术的视野,另一方面是帮助读者由典型16位微机概念过渡到32位微机系统概念。第八章则通过对Pentium机的介绍,简要描述现代32位微机系统的基本结构和原理,使读者对现代32位微型机有一个比较全面的认识和理解。第九章,以MS-DOS为基础介绍磁盘组织与文件系统、基本I/O调用功能,这些功能对提高计算机应用系统的设计效率具有重要意义。第十章介绍单片微型机的原理。第十一章介绍D/A和A/D转换器原理。第十二章对微型计算机应用系统的一般设计进行了讨论。

本书概念解释力求通俗、严谨,内容全面而不显繁杂,以应用系统为主线,图文讲解密切结合。

本书可以作为普通高等学校电子信息及相关专业本科生的教材,也可供从事微机应用开发的工程技术人员和技术爱好者参考使用。

图书在版编目(CIP)数据

微型计算机原理及应用/黄道君,刘丹华,杨西惠编著. —北京:
高等教育出版社,2004.3

ISBN 7-04-014146-9

I. 微… II. ①黄…②刘…③杨… III. 微型计算机-高等学校-教材

IV. TP36

中国版本图书馆 CIP 数据核字(2004)第 005888 号

出版发行 高等教育出版社

社 址 北京市西城区德外大街4号

邮政编码 100011

总 机 010-82028899

购书热线 010-64054588

免费咨询 800-810-0598

网 址 <http://www.hep.edu.cn>

<http://www.hep.com.cn>

经 销 新华书店北京发行所

印 刷 潮河印业有限公司

开 本 787×1092 1/16

印 张 30.75

字 数 640 000

版 次 2004年3月第1版

印 次 2004年3月第1次印刷

定 价 35.00元

本书如有缺页、倒页、脱页等质量问题,请到所购图书销售部门联系调换。

版权所有 侵权必究

前 言

2000年10月作者承接了国家教育部远程教育中心的“微型计算机原理及应用”网络课件制作任务。经过课题组8名同志近2年的齐心努力,“微型计算机原理及应用”网络课件已发布于高等教育出版社远程教育网站(WWW. HEP. EDU. CN)的网络课堂中。

在随后应用该网络课件的教学实践中,同学们的普遍反映是:内容丰富,将基本计算机原理同现代微型计算机技术紧密结合,适用于多门课程(如,微型计算机技术及应用、微型计算机原理与程序设计、微型计算机原理及接口技术、单片机等)的学习;课件的成功之处是应用“动画”、“图形”及“简明的文字”,把微型计算机芯片内部和芯片之间的抽象关系形象生动地表达出来,使人一看即懂,便于理解,便于记忆。但许多同学提出,希望能提供一本基于网络课件、内容更为详细的文字教材。为此,我们编写了本书。

本书具有以下特色:

(1) 吸取了国内相当一部分名校名师的同类教材中的精华。

(2) 以16位的PC/XT机为例,全面地介绍了微型机的基本原理。在此基础上,运用渐进比较方式系统地介绍了现代微型机中各种新技术、新概念,很自然地将16位微型机的概念提升到32位微型机的概念。

(3) 注重创新和应用。在基本原理部分强调了16位微型机的基本概念和创新,从16位微型机的性能提升角度引出了现代微型机的多种新技术和新概念,全书贯穿了部件如何设计、系统如何设计、应用系统如何构成3部分内容。

本书作者身处自动化领域,多年从事微机原理及其应用类课程教学实践,深感微型计算机技术日新月异的发展对教学改革所产生的压力:如何在有限的授课时间内,一方面把微型计算机的基本原理讲透,另一方面又能使学生对现代微型计算机技术有一个基本了解,从而高水平地把计算机技术应用到不同的领域。基于这种思想,本书的内容安排如下:

(1) 本书起点以初涉计算机硬件世界的读者入手,在第一章中较为详细地介绍了计算机中常用的数制知识和数字逻辑基础,以便于读者更好地理解计算机中基本部件的组成和原理。

(2) 本书第二章首先以一节的篇幅介绍了一般微型计算机的硬件和软件组成,引出了基本微型机和微型计算机系统的概念。在随后的几节中以8086/8088 CPU为例,介绍了CPU芯片的内部组成、引脚功能、总线形成及CPU对内存和I/O的管理。这种由一般到特例的介绍,目的是引导读者建立起PC/XT 16位机的具体结构框架。

(3) 第三~六章分别对PC/XT 16位机的指令系统、汇编语言程序设计、内部存储器、

CPU 同外部世界的数据传递方式及 I/O 接口技术进行了详细的分析和讨论,目的是引导读者通过 16 位微型机的学习,全面掌握微型计算机的基本原理。

(4) 第七章从提高 PC/XT 系统机性能角度出发,分别讨论了数字协处理器(8087)、总线裁决器(8289)、I/O 处理器(8089)、CPU 的升级(80x86)、总线的升级(ISA、PCI)和存储系统等一系列新概念、新技术。其目的是引导读者对现代微型机技术发展有一个概括性的了解。

(5) 在第八章中简要介绍了 Pentium 机的内部编程结构、段变换与分页概念及保护概念,目的是为读者进一步深入学习和使用现代微型机打下一个牢固的基础。

(6) 本书第九~十二章从应用角度出发,对微型计算机的系统启动、常用调用程序(这是微机应用开发中经常用到的)、单片微型计算机、A/D 及 D/A 转换、应用系统的设计方法进行了介绍。

本书由黄道君教授负责策划并承担大纲拟定和大部分章节的书稿撰写。刘丹华老师编写了部分书稿并负责全书的录入和图表绘制,并为各章选择了习题。在同“微型计算机原理及应用”网络课件协调方面,杨西惠老师也做了大量工作。

本书编写过程中,作者已受聘于西北农林科技大学信息工程学院计算机科学与技术系任教。本书内容的教学实践是在西北农林科技大学计算机科学与技术系进行的。本书能够出版是与农林科技大学信息工程学院的领导和同学的支持是分不开的,本人在此顺致衷心的感谢。

由于作者水平有限,虽竭尽全力,难免存在不足和疏漏,欢迎计算机专家、教授和广大读者提出意见。

黄道君

2003 年 11 月于(杨陵)

西北农林科技大学信息工程学院

目 录

第一章 计算机中的数制与逻辑预备	第二章 微型计算机组成与原理
知识	§ 2.1 微型计算机的组成与原理概述
§ 1.1 数与数制	2.1.1 微型计算机的硬件组成
1.1.1 数的表示方法与数制	2.1.2 微型计算机的软件组成
1.1.2 数制之间的转换	2.1.3 微型计算机的工作过程
§ 1.2 二进制数的运算规则	§ 2.2 8086/8088 微处理器
1.2.1 二进制数的算术运算	2.2.1 8086/8088 微处理器的功能
1.2.2 二进制数的逻辑运算	结构
§ 1.3 有符号数的表示方法及运算	2.2.2 8086/8088 引脚的功能定义
法则	2.2.3 8086/8088 对存储器的管理
1.3.1 有符号数的表示方法	2.2.4 8086/8088 对 I/O 端口的管理
1.3.2 补码运算时的溢出	§ 2.3 8086/8088 系统总线
§ 1.4 实数的表示方法	2.3.1 最小方式下系统总线结构
1.4.1 定点表示法	2.3.2 最大方式下系统总线结构
1.4.2 浮点表示法	§ 2.4 PC/XT 系统和 PC 扩展总线标准
§ 1.5 码与码制	2.4.1 PC/XT 系统机的组成
1.5.1 8421 BCD 码	2.4.2 PC/XT 扩展总线标准
1.5.2 ASCII 码	§ 2.5 8086/8088 系统总线时序
§ 1.6 基本逻辑电路	2.5.1 总线周期的概念
1.6.1 基本逻辑门电路	2.5.2 总线时序
1.6.2 基本触发器逻辑电路	§ 2.6 典型例题精解
§ 1.7 常用组合逻辑电路	§ 2.7 本章小结
1.7.1 加法器逻辑电路	习题二
1.7.2 算术逻辑运算电路	第三章 指令与汇编
1.7.3 译码器电路	§ 3.1 寻址方式和指令编码格式
1.7.4 数据选择器电路	3.1.1 数据寻址方式
§ 1.8 常用时序逻辑电路	3.1.2 转移地址寻址方式
1.8.1 计数器电路	3.1.3 指令的编码格式
1.8.2 寄存器电路	§ 3.2 8086/8088 指令系统
§ 1.9 典型例题精解	3.2.1 数据传递类指令
§ 1.10 本章小结	3.2.2 算术运算类指令
习题一	3.2.3 逻辑运算指令

3.2.4 移位指令和循环移位指令	(98)	4.2.3 静态 RAM6264 简介	(152)
3.2.5 标志操作指令和微处理器 控制指令	(100)	§ 4.3 动态随机读/写存储器 DRAM2164 简介	(154)
3.2.6 转移指令	(101)	§ 4.4 只读存储器	(157)
3.2.7 循环控制指令	(103)	4.4.1 2764EPROM 简介	(157)
3.2.8 调用与返回指令	(103)	4.4.2 EEPROM 简介	(158)
3.2.9 字符串操作指令	(105)	4.4.3 闪速(FLASH)EEPROM	(160)
3.2.10 输入/输出指令	(107)	§ 4.5 微型计算机内部存储器电路 设计	(163)
3.2.11 软中断指令和中断返回指令 ..	(108)	§ 4.6 外部存储器简介	(166)
§ 3.3 8086/8088 汇编语言的基本概念 ..	(109)	§ 4.7 典型例题精解	(171)
3.3.1 汇编语言中的助记符	(109)	§ 4.8 本章小结	(171)
3.3.2 汇编语句的格式	(110)	习题四	(172)
3.3.3 常数与表达式	(110)	第五章 数据传递方式	(174)
3.3.4 标号与变量	(111)	§ 5.1 程序控制数据传递方式	(174)
3.3.5 属性操作符	(113)	5.1.1 无条件直接数据传递方式	(174)
§ 3.4 伪指令	(114)	5.1.2 查询式传递方式	(175)
3.4.1 通用伪指令	(114)	§ 5.2 中断控制传递方式	(176)
3.4.2 升级的伪指令	(123)	5.2.1 中断与相关概念	(176)
§ 3.5 汇编语言的程序设计	(124)	5.2.2 中断工作过程	(178)
3.5.1 程序设计步骤	(124)	§ 5.3 8259 中断控制器	(182)
3.5.2 程序设计的基本方法	(125)	5.3.1 引脚定义	(182)
§ 3.6 举例	(128)	5.3.2 内部功能结构及初始化	(182)
§ 3.7 汇编、链接及调试	(136)	5.3.3 关于嵌套和优先级排序的 进一步说明	(187)
3.7.1 编辑程序	(136)	5.3.4 应用中的问题	(189)
3.7.2 汇编程序软件的用法和 汇编过程	(138)	5.3.5 寻址与连接	(190)
3.7.3 链接	(141)	5.3.6 应用举例	(190)
3.7.4 调试	(142)	§ 5.4 直接存储器访问方式	(192)
§ 3.8 典型例题精解	(144)	§ 5.5 8237 DMA 控制器	(194)
§ 3.9 本章小结	(145)	5.5.1 引脚定义	(194)
习题三	(147)	5.5.2 8237 内部结构与功能	(196)
第四章 微型计算机中的存储器	(149)	5.5.3 8237 应用举例	(201)
§ 4.1 概述	(149)	§ 5.6 典型例题精解	(204)
4.1.1 存储器的分类	(149)	§ 5.7 本章小结	(205)
4.1.2 存储器性能指标	(149)	习题五	(206)
§ 4.2 静态随机读/写存储器	(150)	第六章 微型计算机接口技术	(207)
4.2.1 一位静态 RAM 原理	(150)	§ 6.1 微型计算机接口技术概述	(207)
4.2.2 静态 RAM 矩阵原理	(151)		

§ 6.2 并行接口和并行接口设计举例	(208)	§ 7.5 微处理器的升级	(298)
§ 6.3 通用并行接口芯片 8255A	(214)	7.5.1 80186	(298)
6.3.1 引脚定义	(215)	7.5.2 80286	(303)
6.3.2 8255A 的工作原理	(215)	7.5.3 80386	(312)
6.3.3 8255A 工作方式	(218)	7.5.4 80486	(321)
6.3.4 应用举例	(225)	§ 7.6 微机系统总线标准的升级	(323)
§ 6.4 定时器/计数器芯片 8253	(226)	§ 7.7 高速缓冲存储器(Cache)的应用	(328)
6.4.1 引脚功能定义	(227)	§ 7.8 典型例题精解	(331)
6.4.2 8253 内部功能结构	(227)	§ 7.9 本章小结	(332)
6.4.3 控制字编程	(229)	习题七	(333)
6.4.4 工作方式	(230)	第八章 Pentium 微处理器	(335)
6.4.5 应用举例	(236)	§ 8.1 Pentium 微处理器的引脚与结构	(335)
§ 6.5 串行通信接口 8250 和 8251	(238)	§ 8.2 Pentium 的内部寄存器	(337)
6.5.1 串行通信术语	(239)	§ 8.3 段描述符和段选择器	(340)
6.5.2 8250 引脚定义	(243)	§ 8.4 工作模式	(343)
6.5.3 8250 内部结构与寄存器地址	(245)	§ 8.5 中断与异常	(347)
6.5.4 8250 初始化、通信编程和 通信流量控制	(251)	§ 8.6 保护机构	(352)
6.5.5 中断功能的实现	(254)	§ 8.7 程序转移	(354)
6.5.6 8251 的编程模型	(255)	§ 8.8 任务切换	(359)
6.5.7 8251 编程过程	(259)	8.8.1 任务状态段	(359)
§ 6.6 CRT 与 CRTC(6845)	(261)	8.8.2 任务切换实现的四种方法	(360)
6.6.1 CRT 显示器的工作原理	(261)	§ 8.9 Pentium 系列机的新发展	(360)
6.6.2 显示器控制系统	(266)	§ 8.10 本章小结	(361)
6.6.3 可编程 CRTC(6845)	(268)	习题八	(362)
6.6.4 MC6845 在彩色显示器接口板中 的应用	(274)	第九章 MS-DOS 操作系统、文件管理 和 I/O 驱动程序	(363)
6.6.5 视频标准	(279)	§ 9.1 微型计算机的操作系统	(363)
§ 6.7 典型例题精解	(279)	9.1.1 MS-DOS 的结构	(363)
§ 6.8 本章小结	(282)	9.1.2 MS-DOS 的引导过程与内存 分配	(365)
习题六	(283)	§ 9.2 MS-DOS 的命令	(368)
第七章 系统性能改善	(284)	9.2.1 内部命令	(368)
§ 7.1 微型计算机改善性能可能的 途径	(284)	9.2.2 外部命令	(369)
§ 7.2 多处理器系统与总线裁决 器 8289	(285)	9.2.3 命令的识别和执行	(370)
§ 7.3 数字协处理器 8087	(289)	§ 9.3 磁盘组织与文件系统	(371)
§ 7.4 输入/输出处理机 8089	(294)	9.3.1 磁盘结构	(371)
		9.3.2 文件和文件名	(372)
		9.3.3 目录	(373)

9.3.4 目录项和目录项表	(373)	§ 10.6 本章小结	(423)
9.3.5 程序段前缀	(375)	习题十	(423)
9.3.6 文件控制块	(376)	第十一章 D/A 转换器与 A/D 转换器	(425)
9.3.7 文件分配表	(378)	§ 11.1 概述	(425)
§ 9.4 DOS 功能调用	(379)	§ 11.2 D/A 转换器及其应用	(426)
§ 9.5 BIOS 中断调用	(380)	11.2.1 基本原理和指标	(426)
§ 9.6 文件读写	(385)	11.2.2 DAC0832 举例	(427)
§ 9.7 本章小结	(385)	§ 11.3 A/D 转换器及其应用	(430)
习题九	(386)	11.3.1 基本原理	(430)
第十章 单片机原理	(388)	11.3.2 技术指标	(432)
§ 10.1 MCS-51 系列机的结构与引脚	(388)	11.3.3 AD574 举例	(433)
10.1.1 内部结构	(389)	11.3.4 ADC0809 举例	(437)
10.1.2 外部引脚	(390)	§ 11.4 本章小结	(440)
10.1.3 时钟与复位电路	(390)	习题十一	(440)
§ 10.2 MCS-51 系列机存储器配置	(392)	第十二章 应用系统	(441)
§ 10.3 MCS-51 的指令系统与汇编	(396)	§ 12.1 微型计算机的应用领域	(441)
10.3.1 寻址方式	(396)	§ 12.2 应用系统的一般构成	(443)
10.3.2 指令系统	(397)	12.2.1 硬件系统	(443)
10.3.3 汇编语言程序	(401)	12.2.2 软件系统	(444)
§ 10.4 MCS-51 的接口与开发	(403)	§ 12.3 应用系统的设计原则和要求	(445)
10.4.1 4 个端口介绍	(403)	§ 12.4 应用系统的基本设计内容	
10.4.2 端口控制命令	(405)	和步骤	(447)
10.4.3 输入/输出举例	(406)	§ 12.5 应用系统的系统集成	(451)
10.4.4 总线形成电路	(407)	§ 12.6 本章小结	(452)
10.4.5 程序存储器扩展	(408)	习题十二	(453)
10.4.6 数据存储器扩展	(408)	附录	(454)
10.4.7 较大存储器空间扩展	(409)	附录 A ASCII 码编码表	(454)
10.4.8 I/O 口扩展	(410)	附录 B 8087 的指令系统	(455)
§ 10.5 MCS-51 的中断、定时与串行		附录 C 8089 的指令系统	(461)
接口	(411)	附录 D DOS 功能调用总表	(464)
10.5.1 中断系统	(411)	附录 E 部分小规模集成电路国家标准	
10.5.2 定时器/计数器系统	(415)	与国外流行表示对照	(482)
10.5.3 串行接口	(419)	参考文献	(483)

第一章 计算机中的数制与逻辑预备知识

计算机是同数打交道的机器,其最基本的功能是进行数据的计算和处理加工。数在计算机中是以器件的物理状态来表示的,为了方便和可靠,在计算机中采用了二进制数字系统,即计算机中要处理的所有数都用二进制数字系统来表示,所有的字母、符号也都要用二进制编码来表示。本章将介绍计算机中数制和码制、各数制运算规则以及数的表示方法等。正确掌握本章是深入学习计算机原理和应用计算机的基础。

§ 1.1 数与数制

1.1.1 数的表示方法与数制

数是客观事物的数量在人们头脑中的反映。数可以用不同的计数制来表示,生活中人类习惯采用十进制计数制,但在计算机中经常采用二进制计数,有时也用八进制计数或十六进制计数。不同的计数制其书写形式和读法不同,但能表达同一个数量。

不同的计数制采用不同的基本计数符号,例如,十进制使用 0、1、2、3、…、9 十个符号,二进制仅使用 0、1 两个符号,通常把这些符号称为数符。而将不同的计数制使用的数符个数称为计数制的基数,以 X 表示,对十进制计数,其基数 $X = 10$,而对二进制计数,其基数 $X = 2$ 。

利用基本数符仅能表示小于基数的整数,要表示一个一般的数(既有整数部分又有小数部分),必须借助于位值法则。所谓位值法则,即用小数点分隔整数和小数,左边为整数,右边为小数。整数位由右向左依次为第 0 位、第 1 位、…、第 $n-1$ 位,小数部分由左向右依次为 -1 位、 -2 位、 -3 位、…、 $-m$ 位,这些序号称为位序号。不同位置有不同的权值,权值既同计数制的基数有关又同位序号有关。位序号为 i 的权值记为 X^i ,若位序号为 i 的数符为 a_i ,则表示的数值为 $a_i \times X^i$ 。因此任何一种计数制都可以用下面形式表示一个一般的数:

$$a_{n-1} \cdots a_3 a_2 a_1 a_0 . a_{-1} a_{-2} \cdots a_{-m}$$

式中 a_{n-1} 到 a_{-m} 是位于不同位的数符,不同计数制其位权值不同。对于基数为 X 的计数制,一个 n 位整数和 m 位小数可表示的实际数值 N 为

$$(N)_X = \sum_{i=0}^{n-1} a_i X^i + \sum_{j=-m}^{-1} a_j X^j = \sum_{i=-m}^{n-1} a_i X^i$$

其中: i 表示数的某一位; a_i 表示第 i 位的数字, 它可以为 $0 \sim X-1$ 中的任一数字; m 和 n 为正整数。

下面将介绍四种常用的计数制。

1. 十进制计数制

对于十进制数, 基数 $X=10$, 其整数位权值由右向左依次为个、十、百、千、..., 而小数位由左向右依次为十分之一、百分之一、..., 很容易被识别。

对于 n 位整数 m 位小数的任意十进制数 N , 可用多项式表示如下:

$$\begin{aligned} N_{10} &= a_{n-1} \cdot 10^{n-1} + a_{n-2} \cdot 10^{n-2} + \cdots + a_1 \cdot 10^1 + a_0 \cdot 10^0 + a_{-1} \cdot 10^{-1} + a_{-2} \cdot 10^{-2} \\ &\quad + \cdots + a_{-(m-1)} \cdot 10^{-(m-1)} + a_{-m} \cdot 10^{-m} \\ &= \sum_{i=-m}^{n-1} a_i \cdot 10^i \end{aligned}$$

例 1.1.1

$$(1\ 392.67)_{10} = 1 \times 10^3 + 3 \times 10^2 + 9 \times 10^1 + 2 \times 10^0 + 6 \times 10^{-1} + 7 \times 10^{-2}$$

2. 二进制计数制

计算机中采用二进制数的优点是物理实现容易且运算特别简单。

二进制数的基数 $X=2$, 其整数位权值依次为 $2^0, 2^1, 2^2, \dots$, 小数位的权值为 $2^{-1}, 2^{-2}, \dots$ 。

对于 n 位整数 m 位小数的任意二进制数 N , 可用多项式表示如下:

$$\begin{aligned} N_2 &= a_{n-1} \cdot 2^{n-1} + a_{n-2} \cdot 2^{n-2} + \cdots + a_1 \cdot 2^1 + a_0 \cdot 2^0 + a_{-1} \cdot 2^{-1} + a_{-2} \cdot 2^{-2} \\ &\quad + \cdots + a_{-(m-1)} \cdot 2^{-(m-1)} + a_{-m} \cdot 2^{-m} \\ &= \sum_{i=-m}^{n-1} a_i \cdot 2^i \end{aligned}$$

例 1.1.2

$$(110101)_2 = 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

$$(0.110101)_2 = 1 \times 2^{-1} + 1 \times 2^{-2} + 0 \times 2^{-3} + 1 \times 2^{-4} + 0 \times 2^{-5} + 1 \times 2^{-6}$$

二进制数常以尾部加一个字母 B 表示, 如: $(10101)_2$ 常以 10101B 表示。

3. 八进制计数制

二进制数虽在计算机中普遍应用, 但书写和阅读却较繁, 易出错。因此计算机技术中常采用八进制或十六进制对二进制数进行书写。所谓八进制数, 即采用 $0, 1, 2, 3, 4, 5, 6, 7$ 这 8 个数符, 并运用位值法则表示的数, 由于基数为 8, 则个位的权值为 8^0 , 十位的权值为 8^1 , 依此类推。

对于 n 位整数 m 位小数的任意八进制数 N , 可用多项式表示如下:

$$\begin{aligned} N_8 &= a_{n-1} \cdot 8^{n-1} + a_{n-2} \cdot 8^{n-2} + \cdots + a_1 \cdot 8^1 + a_0 \cdot 8^0 + a_{-1} \cdot 8^{-1} + a_{-2} \cdot 8^{-2} \\ &\quad + \cdots + a_{-(m-1)} \cdot 8^{-(m-1)} + a_{-m} \cdot 8^{-m} \end{aligned}$$

$$= \sum_{i=-m}^{n-1} a_i \cdot 8^i$$

八进制数以尾部加一个字母 Q 表示,如: $(123)_8 = 123Q$ 。

4. 十六进制计数制

十六进制数是用 0、1、2、3、4、5、6、7、8、9 和 A、B、C、D、E、F 这 16 个数符并运用位值法则表示的数。基数为 16,个位的权值为 16^0 ,十位的权值为 16^1 ,依此类推。

对于 n 位整数 m 位小数的任意十六进制数 N ,可用多项式表示如下:

$$\begin{aligned} N_{16} &= a_{n-1} \cdot 16^{n-1} + a_{n-2} \cdot 16^{n-2} + \cdots + a_1 \cdot 16^1 + a_0 \cdot 16^0 + a_{-1} \cdot 16^{-1} + a_{-2} \cdot 16^{-2} \\ &\quad + \cdots + a_{-(m-1)} \cdot 16^{-(m-1)} + a_{-m} \cdot 16^{-m} \\ &= \sum_{i=-m}^{n-1} a_i \cdot 16^i \end{aligned}$$

十六进制数以尾部加一个字母 H 表示,如: $(13A)_{16} = 13AH$ 。

注意:实际表示时,一个十六进制数如果最高位数字是字母(A~F),则字母前面必须加一个 0,以便与变量名相区别。例如 A32H 若被视为一个数,书写时必须写成 0A32H。

1.1.2 数制之间的转换

十进制数是人们生活中习惯使用的计数法。当要把生活中的数输入计算机时,就存在着如何把十进制数转化为二进制数或其他进制数的问题。下面列出上述四种计数制相互之间的转换方法。

1. 十进制整数同任意进制整数之间的转换

设 N 为要转换的十进制整数,它相应的任意进制整数共有 n 位,即

$$N = a_{n-1}X^{n-1} + a_{n-2}X^{n-2} + \cdots + a_1X^1 + a_0X^0$$

将十进制数 N 除以要求转化为相应数制的基数 X ,得商为 Q ,所得余数即为所求数值的最低位;将前一步商再除以基数 X ,得新商 Q' ,余数即第二位,依次连除下去,直至商为 0,其余数即为最高位。这种方法称为除基取余法。

例 1.1.3 将 17 转化为二进制数。

转换过程如下:

$$\begin{array}{rcl} 2 & \overline{) 17} & \cdots \cdots \text{余数为 } 1 = a_0 \text{ 最低位} \\ 2 & \overline{) 8} & \cdots \cdots \text{余数为 } 0 = a_1 \\ 2 & \overline{) 4} & \cdots \cdots \text{余数为 } 0 = a_2 \\ 2 & \overline{) 2} & \cdots \cdots \text{余数为 } 0 = a_3 \\ 2 & \overline{) 1} & \cdots \cdots \text{余数为 } 1 = a_4 \text{ 最高位} \\ & 0 & \\ & 17=10001B & \end{array}$$

例 1.1.4 将十进制数 301 转换为十六进制数。

转换过程如下：

$$\begin{array}{r|l}
 16 & 301 \quad \cdots \cdots \text{余数为 } D = a_0 \text{ 最低位} \\
 16 & 18 \quad \cdots \cdots \text{余数为 } 2 = a_1 \\
 16 & 1 \quad \cdots \cdots \text{余数为 } 1 = a_2 \text{ 最高位} \\
 & 0
 \end{array}$$

$$301 = 12DH$$

2. 十进制小数同任意进制小数之间的转换

采用乘 2 取整法,即用 2 不断地去乘要转换的十进制数,直到小数部分为 0 或满足所要求的精度为止。把每次乘积的整数部分(不参加下次乘),以初整数为最高位(没有整数的取 0),依次排列,即得所转换的二进制小数。

例 1.1.5 将 0.6875 转换为二进制小数。

$$\begin{array}{l}
 0.6875 \\
 \times 2 \\
 \hline
 \text{取整数} \quad 1.3750 \\
 \text{作为最高位} \quad \times 2 \\
 \hline
 \text{取整数 } 0 \\
 \text{作为次高位} \quad 0.7500 \\
 \quad \times 2 \\
 \hline
 \text{取整数 } 1 \\
 \text{作为第二次高位} \quad 1.5000 \\
 \quad \times 2 \\
 \hline
 \text{乘积小数全为 } 0, \\
 \text{则整数 } 1 \text{ 作为最低位} \quad 1.0000
 \end{array}$$

$$0.6875 = 0.1011B$$

例 1.1.6 将 0.65625 转换为十六进制小数。

$$\begin{array}{l}
 0.65625 \\
 \times 16 \\
 \hline
 \text{高位} \quad 10.50000 \\
 \quad \times 16 \\
 \hline
 \text{低位} \quad 8.00000
 \end{array}$$

$$0.65625 = 0.A8H$$

注意:

(1) 整数和小数部分必须分别转换。

$$25D = 11001B; 1101B = 13D; 0.25D = 0.01000B; 0.1101B = 0.8125D$$

(2) 一个二进制小数可完全准确转化为十进制小数, 但一个十进制小数不一定能准确转换为二进制小数。这即是说, 并非所有的有限位十进制小数都能用有限位的二进制小数表示。因为, 若乘积的小数部分始终不为 0, 说明相对应的 X 进制小数为无限小数, 这时, 乘到能满足计算机精度要求即可。

3. 八进制、十六进制数同二进制数间的转换

二进制数的两个数符 0 和 1, 可用电路的两种状态分别表示, 二者必居其一, 非 0 即 1, 实现起来极为方便, 因此计算机中的数, 均以二进制形式出现。计算机中的八进制和十六进制数并不是用具有 8 个(或 16 个)独立状态的电路来构成一位八进制(或十六进制)计数的, 而是把 3 位二进制数视为一个八进制位, 将 4 位二进制数视为一个十六进制位。因此要将一个八进制数转变为二进制数, 只需将八进制数按位依次变为 3 位二进制数即可。

例 1.1.7 $467Q = 100110111B$

$$\begin{array}{cccc} 4 & 6 & 7 & Q \\ 100 & 110 & 111 & B \\ 467Q = 100110111B \end{array}$$

若需将一个二进制整数转换为八进制整数, 只要从最低位开始, 每 3 位分为一组, 不够 3 位在左面以 0 补足 3 位, 然后将每组二进制数分别用相应的八进制数表示即可。

例 1.1.8 $010\ 111\ 000\ 101B = 2705Q$

$$\begin{array}{cccc} 010 & 111 & 000 & 101 & B \\ 2 & 7 & 0 & 5 & Q \\ 010111000101B = 2705Q \end{array}$$

若需将一个二进制小数转换为八进制小数, 则要从高位开始分组, 不足 3 位在右面以 0 补足, 然后把每组二进制数写成相应八进制数即可。

例 1.1.9 $0.100\ 101\ 011\ 110B = 0.4536Q$

$$\begin{array}{cccc} 0.100 & 101 & 011 & 110B \\ 0.4 & 5 & 3 & 6Q \\ 0.10010101110B = 0.4536Q \end{array}$$

至于十六进制数同二进制数之间的转换, 同八进制数完全类似, 差异仅仅在于十六进制数 1 位对应二进制数的 4 位, 分组要按 4 位一组。

§ 1.2 二进制数的运算规则

1.2.1 二进制数的算术运算

二进制计数制不仅物理实现容易,而且运算方法也较十进制计数制大为简单。只要注意到二进制数加法是“逢二进一”和减法是“借一作二”的特点,仿照十进制数算术运算的方法,很容易理解和完成二进制数的算术运算。

例 1.2.1 $10110101\text{B} + 1111\text{B}$

被加数		1 0 1 1 0 1 0 1
加数		0 0 0 0 1 1 1 1
进位	+	1 1 1 1 1 1
和		1 1 0 0 0 1 0 0

$$10110101\text{B} + 1111\text{B} = 11000100\text{B}$$

例 1.2.2 $11000100\text{B} - 100101\text{B}$

被减数		1 1 0 0 0 1 0 0
减数		0 0 1 0 0 1 0 1
借位	-	1 1 1 1 1 1
差		1 0 0 1 1 1 1 1

$$11000100\text{B} - 100101\text{B} = 10011111\text{B}$$

例 1.2.3 $1101\text{B} \times 1011\text{B}$

被乘数		1 1 0 1
乘数	×	1 0 1 1
		1 1 0 1
		1 1 0 1
		0 0 0 0
	+	1 1 0 1
积		1 0 0 0 1 1 1 1

$$1101\text{B} \times 1011\text{B} = 10001111\text{B}$$

例 1.2.4 $100110\text{B} \div 110\text{B}$

$$\begin{array}{r}
 000110 \\
 110 \overline{) 100110} \\
 \underline{110} \\
 0111 \\
 \underline{110} \\
 10
 \end{array}$$

$$100110B \div 110B = 110B \text{ 余 } 10B$$

1.2.2 二进制数的逻辑运算

二进制数的逻辑运算常用的有“与”、“或”、“异或”和“非”四种。在组成表达式时,可分别用符号“AND”、“OR”、“XOR”和“NOT”作为运算符,有时也用“ $\cdot$ ”、“ $+$ ”、“ $\oplus$ ”作为“与”、“或”、“异或”的运算符,用数字上面加横线“ $\bar{}$ ”表示对该数进行非运算。

二进制数的逻辑运算只能按位进行运算,不存在进位问题,这是逻辑运算同算术运算的区别所在。

1 位二进制数运算规则如下:

“与”运算的规则为

$$1 \cdot 0 = 0$$

$$0 \cdot 1 = 0$$

$$0 \cdot 0 = 0$$

$$1 \cdot 1 = 1$$

“或”运算的规则为

$$0 + 0 = 0$$

$$1 + 0 = 1$$

$$0 + 1 = 1$$

$$1 + 1 = 1$$

“异或”运算的规则为

$$0 \oplus 0 = 0$$

$$1 \oplus 0 = 1$$

$$0 \oplus 1 = 1$$

$$1 \oplus 1 = 0$$

“非”运算的规则为

$$\bar{0} = 1$$

$$\bar{1} = 0$$

例 1.2.5 二进制数 10010111 和 00111000 的“与”、“或”及“异或”运算。

$$\begin{array}{r}
 1\ 0\ 0\ 1\ 0\ 1\ 1\ 1 \\
 \text{与)}\quad 0\ 0\ 1\ 1\ 1\ 0\ 0\ 0 \\
 \hline
 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0
 \end{array}$$

$$\begin{array}{r}
 1\ 0\ 0\ 1\ 0\ 1\ 1\ 1 \\
 \text{或)}\quad 0\ 0\ 1\ 1\ 1\ 0\ 0\ 0 \\
 \hline
 1\ 0\ 1\ 1\ 1\ 1\ 1\ 1
 \end{array}$$

$$\begin{array}{r}
 1\ 0\ 0\ 1\ 0\ 1\ 1\ 1 \\
 \text{异或)}\quad 0\ 0\ 1\ 1\ 1\ 0\ 0\ 0 \\
 \hline
 1\ 0\ 1\ 0\ 1\ 1\ 1\ 1
 \end{array}$$

例 1.2.6 二进制数 10010110 的“非”运算。

运算结果为

0 1 1 0 1 0 0 1

§ 1.3 有符号数的表示方法及运算法则

1.3.1 有符号数的表示方法

1. 数的符号表示法

为了表示二进制数的符号,一般方法是牺牲二进制数位中的最高位,以 0 代表“+”二进制数,1 代表“-”二进制数。8 位有符号数的符号位为 D7 位,如图 1.3.1 所示。

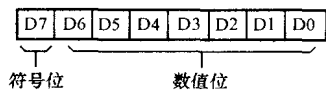


图 1.3.1 8 位有符号数表示

16 位有符号数的符号位为 D15 位,如图 1.3.2 所示。

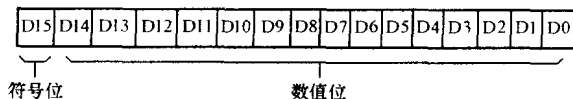


图 1.3.2 16 位有符号数表示

这样,数的符号在机器中就数字化了。

2. 有符号数的原码表示

直接应用前述的符号规则和二进制计数规则写出的 0、1 码串称为数的原码,以 $[X]_{\text{原}}$ 表示。

$$[X]_{\text{原}} = \begin{cases} 0 & X_{n-1}X_{n-2}\cdots X_1X_0 \quad \text{当 } X \geq 0 \\ 1 & X_{n-1}X_{n-2}\cdots X_1X_0 \quad \text{当 } X \leq 0 \end{cases}$$

其最高位为符号位,其余 $n-1$ 位按二进制规则作为数值位。

例 1.3.1 若 $X_1 = +1010101$, 则 $[X_1]_{\text{原}} = 01010101$;

若 $X_2 = -1010101$, 则 $[X_2]_{\text{原}} = 11010101$ 。

8 位二进制原码能表示的数的范围是 $-127 \sim +127$, 简单易理解, 与真值之间的转换也很方便。但运算麻烦, 进行数值加、减运算时, 必须考虑数的符号和绝对值大小。另外, 有符号数采用原码表示时, 数 0 有两种表示形式: $[+0]_{\text{原}} = 00000000\text{B}$; $[-0]_{\text{原}} = 10000000\text{B}$ 。

3. 有符号数的补码表示

(1) 补码的定义

计算机技术中, 广泛采用补码表示有符号数, 补码的概念源于同余概念:

$$a + NK = a \pmod{K}$$

其中 K 是模, N 为任意整数。

也就是说, 在模的意义下, 数 a 与该数本身加上其模的整数倍之和相等。在数 a 的无数个 $a + NK$ 同余数中, 称 N 为 1 的同余数为补数。即

$$[a]_{\text{补}} = a + K \pmod{k} = \begin{cases} a & 0 \leq a < k \\ K - |a| & -k < a < 0 \end{cases}$$

在计算机中, 数据的位数即字长总是有限的。假设字长为 n 位, 当两数相加求和时, 如果 n 位的最高位产生进位就会丢掉, 这正是在模的意义下相加的概念。

当数 X 用 n 位二进制数表示时, 1 位为符号位, $(n-1)$ 位为数值位, 它的模 K 为 2^n 。数 X 的补码记为 $[X]_{\text{补}}$, 补码的定义为

$$[X]_{\text{补}} = \begin{cases} X & 0 \leq X < 2^{n-1} \\ 2^n + X & -2^{n-1} \leq X < 0 \end{cases}$$

由定义可见, 正数补码与其原码相同, 只有负数才有求补的问题。所以严格的说, “补码表示法”应称为“负数的补码表示法”。

(2) 补码的求法

按定义求补码, 必须做一次减法, 计算不方便, 很少使用。通常采用的求补方法是将负数原码的符号位保持不变, 其余各位按位取反再在最低位加 1。

例 1.3.2 若 $X = -1010101\text{B}$, 则