541

P36-43
F-61

高等学校计算机基础教育教材精选

微型计算机原理与接口技术

冯博琴　主编
吴　宁　陈文革　程向前　编著

清华大学出版社

(京)新登字 158 号

内 容 简 介

本书主要以 8086/8088 CPU 为例，全面剖析微处理器的基本结构、指令系统、存储系统与接口电路，介绍了新型 CPU 技术、主板技术和总线及存储技术。

本书例题丰富，形式多样，全部实例都有详细的分析和注释；并根据作者多年从事教学、科研的经验和体会，突出重点，循序渐进，力求通俗易懂，尤其针对学生学习时普遍反映的难点问题进行了详细讨论，收到了很好的课堂效果。

本书是普通高等院校非计算机专业本科学生的教材，也可作为成人高等教育的培训教材，还可供广大科技人员自学参考。

书　名：微型计算机原理与接口技术
作　者：冯博琴　主编
出版者：清华大学出版社(北京清华大学学研大厦，邮编 100084)
　　　　http://www.tup.tsinghua.edu.cn
印刷者：北京清华园胶印厂
发行者：新华书店总店北京发行所
开　本：787×1092 1/16　印张：27　字数：635 千字
版　次：2002 年 2 月第 1 版　2002 年 2 月第 1 次印刷
书　号：ISBN 7-302-05272-7/TP・3098
印　数：0001～8000
定　价：29.00 元

前言

本书是非计算机专业学生学习“微型计算机原理与接口技术”课程的通用教材，主要以8086/8088CPU为例，分析了微处理器的基本结构、指令系统、存储系统及输入输出接口电路。同时，为适应形势的发展，还介绍了部分新型CPU技术、主板技术和总线构成等。本书主要特点是：

(1) 对所举全部实例都有详细的分析和注释。例如在汇编语言程序设计部分，读者经常反映入门困难，本书通过对每段程序添加详细解释，使读者能够较为容易地理解和掌握汇编语言程序设计的思想。

(2) 内容新颖。如本书在第2章中专门介绍了当前部分新型微处理机的性能，使读者能够对8086/8088以后的CPU技术的发展和概况有一定的了解。另外，用一定篇幅介绍了总线的概念、分类常用系统总线标准，以及多媒体技术。在存储系统部分，增加了高速缓存(cache)、虚拟存储的基本概念和存储原理。

(3) 例题丰富，形式多样。本书以面向应用为主，在例题、接口电路等的选择上，尽量考虑与实际的工程应用相结合，插入了大量的电路连接图、结构图、时序图和详细的分析说明。

(4) 常用外设介绍。较为系统地介绍了微型机常用的外部设备。

(5) 循序渐进，易于理解。考虑到本书读者主要为非计算机专业学生，在进入这门课的学习之前并不具备计算机组成和结构方面的知识，是初次涉及计算机的内部结构，所以本书在内容次序的安排上注意由浅入深，突出重点；在文字叙述上，力求通俗易懂。同时，在编写中加入了作者多年从事教学、科研的经验和体会。

通过本书的学习并结合上机实践，可使读者对微型计算机系统的组成和工作原理有初步了解，具备一定的汇编语言程序设计能力，并能够开发简单外部设备的应用控制系统。因此，本书不仅作为课堂用教材，还能对学生以后的工作有一定指导作用。

本书由冯博琴教授策划并任主编，由吴宁、陈文革、程向前编写。其中第1、3、4、6以及第5章的第1、2、3、4、6节由吴宁编写，第2章和第5章部分内容由陈文革编写，第7章由程向前编写。

由于计算机技术的发展日新月异，新技术层出不穷，加之时间仓促，编者水平有限，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

2001年12月于西安

第1章 基础知识

1.1 概 述

计算机技术是20世纪发展最快的技术之一，自1946年第一台计算机问世以来，它的发展可谓一日千里。在五十多年的历史中，已经历了由电子管计算机、晶体管计算机、集成电路计算机到大规模、超大规模集成电路计算机这样四代的更替。而目前，已有了第五代“非冯·诺依曼”计算机和第六代“神经”计算机的研制计划。

计算机按其性能、价格和体积等的不同，可分为巨型机、大型机、中型机、小型机、微型机和单片机六大类。

微型计算机诞生于20世纪70年代，由于它体积小，价格低，尤其是日益提高的性能价格比，使其迅速在各行各业乃至家庭得到了广泛的应用。现在一台微型机的处理能力，不仅早已超过了20世纪50年代初期占地上百平方米、重量数千吨、功耗几十千瓦的电子管计算机，而且大大超过了20多年前、造价数十万美元的晶体管数字计算机系统。

微处理器是微型机的核心芯片，简称μP或MP(micro processor)。它将计算机中的运算器和控制器集成在一片硅片上，也称为中央处理单元，即CPU(central processing unit)。它是20世纪70年代人类重要的创新之一，在不到30年的时间中，获得了极快的发展，其集成度和性能，几乎每两年就提高一倍。

微处理器和微型计算机的发展历史是和大规模集成电路的发展分不开的。20世纪60年代初期的硅平面管工艺和二极管晶体管逻辑电路的发展，使得在1963年、1964年有了小规模集成电路(small scale integration，SSI)的出现，之后的金属氧化物半导体(metal oxide semiconductor，MOS)工艺，又使集成度提高了一大步。到60年代后期，在一片几平方毫米的硅片上，已可集成几千个晶体管，这就出现了大规模集成电路(large scale integration，LSI)。LSI器件体积小、功耗低、可靠性高，为微处理器的生产打下了基础。现代最新型的集成电路已可在单个芯片上集成上千万个晶体管，线宽小于0.13μm，工作频率超过2GHz。

到目前为止，微处理器的发展过程大致可分为六代。

1. 第一代微处理器

第一代微处理器为4位或低档8位微处理器，其发展大约从1971到1973年。其主

要代表是美国 Intel 公司在 1971 年研制成功的 4004 微处理器和于次年推出的 8008 微处理器。

4004 是一种 4 位微处理器，可进行 4 位二进制的并行运算，拥有 45 条指令，速度为 0.05MIPS(million instructions per second，每秒百万条指令)。4004 的功能极其有限，主要用于计算器、电动打字机、照相机、台秤、电视机等家用电器上，使这些电器设备具有智能化，从而提高它们的性能。4004 本来是作为高级袖珍计算器而设计的，一般不适用于通用计算机，后经改进，成为可用于微型机的 8008 型微处理器。

8008 是世界上第一种 8 位的微处理器，与 4004 相比，它可一次处理 8 位二进制数据，其寻址空间扩大为 16K 字节，并且扩充了指令系统(达到 48 条)。

第一代微处理器的指令系统比较简单，运算能力较弱，速度也比较慢(基本指令执行时间为 10 到 20μs)，但价格低廉。其软件主要使用机器语言和简单的汇编语言。

2. 第二代 8 位微处理器

1973 至 1978 年间，各公司开始推出了第二代微处理器。首先在 1973 年，Intel 公司在 8008 的基础上推出了另一种 8 位微处理器 Intel 8080。这是一个划时代的产品，因为它是第一个真正实用的微处理器。它的存储器寻址空间增加到 64K 字节，并扩充了指令集，指令执行速度达到 0.5MIPS，比 8008 快 10 倍。另外，它使 CPU 外部电路的设计变得更加容易且成本降低。

这个时期推出的微处理器除了 Intel 公司的 8080 外，还有 Intel 公司的 8085，Motorola 公司的 MC6800 系列，以及 Zilog 公司的 Z80 等。第二代微处理器与第一代相比，其集成度提高了 1～4 倍，运算速度提高了 10～15 倍，指令系统相对比较完善，已具备典型的计算机体系结构及中断、直接存储器存取(DMA)等功能。软件方面除汇编语言外，还可使用如 BASIC、FORTRAN 等高级语言。至于后期的以 8080A/8085A、Z80、MC6502 等 CPU 芯片为核心的具有磁盘和各种外设的微型计算机，还可配上简单的操作系统，如 CP/M(control program/monitor)。

3. 第三代 16 位微处理器

1977 年前后，超大规模集成电路(VLSI)研制成功，在一片硅片上可集成 1 万个以上的晶体管，这为研制 16 位微处理器创造了必要的条件。1978 年，Intel 公司率先推出 16 位微处理器 8086，这是第一种第三代微处理器。同时，为了方便原 8 位机用户，Intel 公司又很快推出了一种准 16 位微处理器 8088。它的内部结构为 16 位，但外部数据总线是 8 位，其指令系统与 8086 完全兼容。采用 8088 的 IBM PC、PC/XT 准 16 位计算机，以其体系结构的开放性和较高的性能价格比而很快占领了市场。

在 Intel 公司推出 8086、8088 CPU 之后，各公司也相继推出了同类的产品，有 Motorola 公司的 MC68000 和 Zilog 公司的 Z8000 等。

16 位微处理器比 8 位微处理器的集成度提高了约一个数量级，功能也大大增强，这表现在如下几个方面：

• 数据总线的位数由 8 位增加到 16 位，大幅度提高数据处理能力。

- 地址总线的位数由 16 位增加到 20 位以上。增强了计算机的存储器寻址范围。
- 时钟频率提高到 5MHz～40MHz。使系统运算速度大为提高，基本指令执行时间约 0.15μs。同时 CPU 的内部结构也有很大的改进，如 Intel 8086 内部采用流水线结构，设置了 6 字节的指令预取队列。因此，处理速度明显加快。CPU 内部的通用寄存器增多，从而减少了对存储器的访问频度。而且大多数通用寄存器都可以在算术运算指令中作为累加器使用。
- 扩充了指令系统，并且指令功能也大大增强。如指令系统中增加了乘法和除法指令，各种指令数量达上万条。寻址方式也丰富了，如 MC68000 具有多达 14 种寻址方式。由于指令系统中指令的数量多，复杂程度高，这类微处理器通常被称为 CISC(complex instruction set computer，复杂指令系统计算机)结构的微处理器。
- 可处理多种数据类型。有二进制位、压缩 BCD 码、非压缩 BCD 码、字节、字、双字、字串等。
- 中断功能增强。
- 具有构成多微处理器系统的能力。
- 配备有较强的系统软件。

16 位微处理器比 8 位微处理器有更大的寻址空间、更强的运算能力、更快的处理速度和更完善的指令系统。所以，16 位微处理器已能够替代部分小型机的功能。特别是在单任务、单用户的系统中，8086 等 16 位微处理器更是得到了广泛的应用。

1982 年，Intel 公司又推出 16 位高级微处理器 80286。它具有多任务系统所必需的任务转换功能、存储器管理能力和多种保护功能。同一年，Motorola 公司也推出了同类型的 MC68010。这两种微处理器的数据总线虽然仍是 16 位的，但地址总线增加到 24 位，其存储器直接寻址能力可达 16MB。时钟频率提高到 5MHz～25MHz。在 20 世纪 80 年代中、后期至 1991 年初，80286 一直是个人计算机的主流 CPU。

4. 第四代 32 位高档微处理器

1985 年，Intel 公司推出第四代微处理器 80386。它是一种与 8086 向上兼容的 32 位超级微处理器，具有 32 位的数据线，32 位的地址线，存储器直接寻址能力可达 4GB，每一个任务具有 64TB 的逻辑存储空间。其执行速度达到 3～4MIPS。同一时期推出的 32 位微处理器中，还有 Motorola 公司的 MC68020、贝尔实验室的 Bellmac-32A，National Semiconductor 公司的 16032 和 NEC 的 V70 等。32 位微处理器的出现，使微处理器开始进入一个崭新的时代。32 位微处理器无论从结构、功能、应用范围等方面看，可以说是小型机的微型化。这时 32 位微处理器组成的微型机已接近 20 世纪 80 年代小型机的水平。

随着集成电路工艺水平的进一步提高，1989 年，Intel 公司又推出性能更高的 32 位微处理器 80486，它在芯片上集成约 120 万个晶体管，是 80386 的 4 倍。80486 由 3 个部件组成：一个 80386 体系结构的主处理器，64 位的内部数据总线，一个与 80387 兼容的数字协处理器和一个 8KB 容量的高速缓冲存储器，并采用了 RISC(reduction instruction set computer，精简指令系统计算机)技术、与 RAM 进行高速数据交换的突发总线等先进技术。这些新技术的采用，使 80486 在同等时钟频率下的处理速度要比 80386 快 2 到 4 倍。

同期推出的产品还有 Motorola 公司的 MC68030 的后继换代产品 MC68040，NEC 公司的 V70 的后继换代产品 V80。这是 3 种典型的 CISC 体系结构的 32 位高档微处理器。

5. 第五代 32 位高档微处理器

1993 年，Intel 公司推出了 32 位微处理器 Pentium(中文译名为奔腾)。它集成了 330 万个晶体管，内部采用 4 级超标量结构，数据线 64 位，地址线 36 位。工作频率为 60/66MHz，处理速度达 110MIPS。由于第一代 Pentium 采用 0.8μm 工艺技术和 5V 电源驱动，使得芯片尺寸较大，成本过高；另外其功耗达 15W，使系统散热成为问题。在 1994 年 3 月，Intel 推出了第二代 Pentium(以 P54C 代称)，P54C 采用 0.6μm 工艺和 3.3V 电源，功耗仅为 4W，而且可在不需要时自动关闭浮点单元，散热问题基本得以解决。P54C 的主时钟为 100MHz 和 90MHz 两种。在常规配置下，P54C 系统的处理能力比原奔腾系统高出了 40%、是 486DX/66MHz 系统性能的两倍。在体系结构上，Pentium 在内核中采用了 RISC 技术，可以说它是 CISC 与 RISC 技术相结合的产物。

同时期推出的第五代微处理器还有 IBM、Apple 和 Motorola 3 家联盟的 PowerPC (这是一种完全的 RISC 微处理器)，以及 AMD 公司的 K5 和 Cyrix 公司的 M1 等。

表 1-1　Intel 主要 CPU 芯片一览表

代	发表年份	字长(位)	型号	线宽(μm)	晶体管数(万个)	时钟频率(MHz)	速度(MIPS)
一	1971 1972	4 8	4004 8008	50	0.2 0.3	<1	0.05
二	1974	8	8080	20	0.5	2～4	0.5
三	1978 1982	16	8086 80286	2～3	2.9 13.4	4.77～10 8～16	<1 1～2
四	1985 1989	32	80386 80486	1～2	27.5 120	16～33 25～66	6～12 20～40
五	1993	32	Pentium	0.6～0.8	330	60～200	100～200
六	1995 1996 1997 1999 2001	32	Pentium Pro Pentium MMX Pentium Ⅱ Pentium Ⅲ Pentium Ⅳ	0.6 0.6 0.35 0.13～0.25 0.13～0.18	550 450 750 850 1000	133～200 166～233 233～450 450～1200 1300～2400	>300
七	有计划未发表	64	Itanium	0.13	CPU:2.5K cache:30K	800(20 条指令/时钟周期)	>3000

注：Pentium 具有 64 位数据总线，但仅有 32 位地址总线，所以它仍称其为 32 位的微处理器。

6. 第六代 Pentium 微处理器

1996 年 Intel 公司将它的第六代微处理器正式命名为 Pentium Pro，该处理器的集成电路采用了 0.35μm 的工艺，时钟频率为 200MHz，运算速度达 200MIPS。

1998 年到 2001 年，Intel 又进一步推出了一系列 Pentium Pro 的改进型微处理器、Pentium Ⅱ和 Pentium Ⅲ。其他公司类似的产品还有 AMD 的 K7。这些 CPU 的集成度已高达近 1 千万个晶体管，时钟频率也达到了 1GHz 以上。目前正在进入市场的是 Pentium 4 系列，其 CPU 集成度达 2 千 5 百万个晶体管，工作频率达 2GHz 以上。

表 1-1 列出了 Intel 公司第一代到第六代 CPU 的发展情况（表中也列出了即将上市的 Itanium 新一代微处理器）。

1.2 计算机中的数制

在日常生活中，人们习惯于使用十进制数来进行记数和计算。但对计算机来说，它只能识别由“0”和“1”构成的二进制代码，也就是说计算机中的数是用二进制表示的。但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，或适应某些特殊场合的需要，在计算机中有时也采用十六进制数和十进制数。所以，在学习计算机原理之前，首先需要了解和掌握这 3 种常用记数制及其相互间的转换。

1.2.1 常用记数制

1. 十进制数

十进制数中有 0～9 十个数字符号，无论数的大小，都可用这十个符号的组合来表示。任何一个十进制数 D，都可用权展开式表示为

$$\begin{aligned}(D)_{10} &= D_{n-1}\times 10^{n-1} + D_{n-2}\times 10^{n-2} + \cdots + D_1\times 10^1 + D_0\times 10^0 + D_{-1}\times 10^{-1} \\ &\quad + \cdots + D_{-m}\times 10^{-m} \\ &= \sum_{i=-m}^{n-1} D_i \times 10^i \end{aligned} \tag{1.1}$$

其中，D_i 是 D 的第 i 位的数码，可以是 0～9 十个符号中的任何一个，n 和 m 为正整数，n 表示小数点左边的位数，m 表示小数点右边的位数，10 为基数，10^i 称为十进制的权。

【例 1-1】 十进制数 3256.87 可表示为

$$(3256.87)_{10} = 3\times 10^3 + 2\times 10^2 + 5\times 10^1 + 6\times 10^0 + 8\times 10^{-1} + 7\times 10^{-2}$$

2. 二进制数

二进制数的每一位只取 0 和 1 两个数字符号，其计数规律遵循逢二进一的法则。一个二进制数 B 可用其权展开式表示为

$$(B)_2 = B_{n-1}\times 2^{n-1} + B_{n-2}\times 2^{n-2} + \cdots + B_0\times 2^0 + B_{-1}\times 2^{-1} + \cdots + B_{-m}\times 2^{-m}$$
$$= \sum_{i=-m}^{n-1} B_i \times 2^i \qquad (1.2)$$

其中，B_i只能取 1 或 0，2 为基数，2^i为二进制的权，m、n 的含义与十进制表达式相同。为与其他进位记数制相区别，一个二进制数通常用下标 2 表示。

【例 1-2】 二进制数 1010.11 可表示为

$$(1010.11)_2 = 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 0\times 2^0 + 1\times 2^{-1} + 1\times 2^{-2}$$

3. 十六进制数

十六进制数共有 16 个数字符号，0～9 及 A～F，逢十六进一。一个十六进制数 H 也可用权展开式表示为

$$(H)_{16} = H_{n-1}\times 16^{n-1} + H_{n-2}\times 16^{n-2} + \cdots + H_0\times 16^0 + H_{-1}\times 16^{-1} + \cdots + H_{-m}\times 16^{-m}$$
$$= \sum_{i=-m}^{n-1} H_i \times 16^i \qquad (1.3)$$

这里，H_i的取值在 0～F 的范围内，16 为基数，16^i为十六进制数的权；m、n 的含义与上相同。十六进制数通常用下标 16 表示。

【例 1-3】 十六进制数 2AE.4H 可表示为

$$(2AE.4)_{16} = 2\times 16^2 + A\times 16^1 + E\times 16^0 + 4\times 16^{-1}$$

二进制数与十六进制数之间存在有一种特殊关系，即 $2^4=16$，也就是说一位十六进制数恰好可用四位二进制数来表示，且它们之间的关系是惟一的。所以，在计算机应用中，虽然机器只能识别二进制数，但在数字的表达上更广泛地采用十六进制数。

计算机中常用的二进制数、十六进制数和十进制数之间的关系如表 1-2 所示。

表 1-2 数制对照表

十进制数	二进制数	十六进制数
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

4. 其他进制数

除以上介绍的二、十和十六进制 3 种常用的进位记数制外，计算机中还可能用到八进制数，有兴趣的读者可自行将其记数及表示方法进行归纳，这里就不再详细介绍了。下面给出任一进位制数的权展开式的一般形式。

一般地，对任意一个 K 进制数 S 都可表示为

$$(S)_k = S_{n-1} \times K^{n-1} + S_{n-2} \times K^{n-2} + \cdots + S_0 \times K^0 + S_{-1} \times K^{-1} + \cdots + S_{-m} \times K^{-m}$$

$$= \sum_{i=-m}^{n-1} S_i \times K^i \tag{1.4}$$

其中，S_i 是 S 的第 $\boldsymbol{i}$ 位的数码，可以是所选定的 K 个符号中的任何一个；$\boldsymbol{n}$ 和 $\boldsymbol{m}$ 的含义同上，K 为基数，K^i 称为 K 进制数的权。

除了用基数作为下标来表示数的进制外，还可以在数的后面加上字母 B、H、D 来分别表示二进制数、十六进制数和十进制数，如 11000101B、2C0FH、1300D 等。在不至于混淆时，十进制数后面的 D 也可以省略。

1.2.2 各种数制之间的转换

人们习惯的是十进制数，计算机采用的是二进制数，人们编写程序又多采用十六进制数，因此必然会产生不同进位记数制之间进行转换的问题。

1. 非十进制数到十进制数的转换

非十进制数转换为十进制数的方法比较简单，只要将它们按相应的权表达式展开，再按十进制运算规则求和，即可得到它们对应的十进制数。

【例 1-4】 将二进制数 1101.101 转换为十进制数。

解：根据二进制数的权展开式，有

$$(1101.101)_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3}$$
$$= (13.625)_{10}$$

【例 1-5】 将十六进制数 64.CH 转换为十进制数。

解：根据十六进制数的权展开式，有

$$\begin{aligned}(64.\mathrm{C})_{16} &= 6 \times 16^1 + 4 \times 16^0 + \mathrm{C} \times 16^{-1} \\ &= 6 \times 16^1 + 4 \times 16^0 + 12 \times 16^{-1} \\ &= (100.75)_{10}\end{aligned}$$

2. 十进制数转换为非十进制数

(1) 十进制数转换为二进制数

十进制数整数和小数部分应分别进行转换。整数部分转换为二进制数时采用“除 2 取余”的方法。即连续除以 2 并取余数作为结果，直至商为 0，得到的余数从低位到高位

依次排列即得到转换后二进制数的整数部分;对小数部分,则用“乘 2 取整”的方法。即对小数部分连续用 2 乘,以最先得到的乘积的整数部分为最高位,直至达到所要求的精度或小数部分为零为止(可以看出,转换的结果的整数和小数部分是从小数点开始分别向高位和向低位逐步扩展)。

【例 1-6】 将十进制数 112.25 转换为等值的二进制数。

解:

整数部分	小数部分
112/2＝56 …… 余数＝0(最低位)	0.25×2＝0.5 …… 整数＝0(最高位)
56/2＝28 …… 余数＝0	0.5×2＝1.0 …… 整数＝1
28/2＝14 …… 余数＝0	
14/2＝7 …… 余数＝0	
7/2＝3 …… 余数＝1	
3/2＝1 …… 余数＝1	
1/2＝0 …… 余数＝1	

从而得到转换结果$(112.25)_{10}=(1110000.01)_2$

(2) 十进制数转换为十六进制数

与十进制数转换为二进制数的方法类似,整数部分按“除 16 取余”的方法进行,而小数部分按“乘 16 取整”的方法进行。

【例 1-7】 将十进制数 301.6875 转换为等值的十六进制数。

解:

整数部分	小数部分
301/16＝18 …… 余数＝D	0.6875×16＝11.0000 …… 整数
	$=(11)_{10}=(B)_{16}$
18/16＝1 …… 余数＝2	
1/16＝0 …… 余数＝1	

所以有 $(301.6875)_{10}=(12D.B)_{16}$。

也可将十进制数先转换为二进制数,再转换为十六进制数。下边将会看到,后者的转换是非常方便的。

3. 二进制数与十六进制数之间的转换

由于 $2^4=16$,故一位十六进制数能够表示的数值恰好相当于四位二进制数能够表示的数值,这就使十六进制数与二进制数间的相互转换变得非常容易。

将二进制数转换为十六进制数的方法是:从小数点开始分别向左和向右把整数和小数部分每四位分为一组。若整数最高位的一组不足 4 位,则在其左边补零;若小数最低位的一组不足 4 位,则在其右边补零。然后将每组二进制数用对应的十六进制数代替,则得到转换结果。

【例 1-8】 将二进制数 110100110.101101B 转换为十六进制数。

解：

二进制数	0001	1010	0110.	1011	0100
	↓	↓	↓	↓	↓
十六进制数	1	A	6.	B	4

所以有$(110100110.101101)_2=(1A6.B4)_{16}$。

十六进制数转换为二进制数的方法与上述过程相反，即用四位二进制代码取代对应的一位十六进制数。

【例 1-9】 将十六进制数 2A8F.6DH 转换为二进制数。

解：

十六进制数	2	A	8	F.	6	D
	↓	↓	↓	↓	↓	↓
二进制数	0010	1010	1000	1111.	0110	1101

所以有$(2A8F.6D)_{16}=(0010101010001111.01101101)_2$

1.3 无符号二进制数的算术运算和逻辑运算

1.3.1 二进制的算术运算

由于二进制数中只有 0 和 1 两个数，故其运算规则比十进制要简单得多。

1. 加法运算

二进制的加法运算遵循如下法则：

0+0=0　　0+1=1　　1+0=1　　1+1=0（有进位）

【例 1-10】 计算 10110110B+01101100B =(?)B

解：

```
进  位      1 11111000
被加数        10110110
加  数 +      01101100
----------------------
            1 00100010
```

可得 10110110B+01101100B = 100100010B

2. 减法运算

二进制数的减法遵循如下法则：

0-0=0　　1-0=1　　1-1=0　　0-1=1（有借位）

【例 1-11】 计算 11000100B-00100101B = (?)B

解：

借　位	01111110
被减数	11000100
减　数 －	00100101
	10011111

可得 11000100B－00100101B ＝ 10011111B

3. 乘法运算

二进制数的乘法法则如下：

0×0=0　　0×1=0　　1×0=0　　1×1=1

即仅当两个 1 相乘时结果为 1，否则结果为 0。所以，二进制数的乘法是非常简单的。若乘数位为 1，就将被乘数照抄加于中间结果，若乘数位为 0，则加 0 于中间结果，只是在相加时要将每次中间结果的最后一位与相应的乘数位对齐。

【例 1-12】 求两个二进制数 1100B 与 1001B 的乘积。

解法一：按照十进制的乘法过程有

1100	被乘数
× 1001	乘　数
1100	部分积
0000	
0000	
1100	
1101100	乘　积

可得 1100B×1001B ＝ 1101100B。

解法二：采用移位加的方法，则有

乘数		被乘数	部分积
1 0 0 1		1100	0000
	乘数为 1，加被乘数到部分积上，并		1100
	将被乘数左移 1 位	11000	
	乘数为 0，不加被乘数，被乘数左移 1 位	110000	
	乘数为 0，不加被乘数，被乘数左移 1 位	1100000	
	乘积为 1，加左移后的被乘数到部分积上		1100
			+1100000
			1101100

即可得 1100B×1001B ＝ 1101100B。

可以看出计算结果与方法一相同。由此可见，二进制的乘法运算可以转换为加法和移位的运算。事实上，在计算机中乘法运算就是这样做的。每左移一位，相当于乘以 2，

而左移 n 位就相当于乘以 2^n。

4. 除法运算

除法是乘法的逆运算。所以二进制数的除法运算也可转换为减法和右移运算。每右移一位相当于除以 2，右移 n 位就相当于除以 2^n。

1.3.2 无符号数的表示范围

1. 无符号二进制数的表示范围

一个 n 位的无符号二进制数 X，它可表示的数的范围为

$$0 \leqslant X \leqslant 2^n - 1$$

比如一个 8 位的二进制数，即 $n=8$，其表示范围为 $0 \sim 2^8-1$，即 00H～FFH(0～255)。若运算结果超出数的可表示范围，则会产生溢出，结果将不正确。

【例 1-13】 计算 10110111B+0100110B=(?)B

解：

$$\begin{array}{r} 10110111 \\ +\ 01001101 \\ \hline 1\quad 00000100 \end{array}$$

由上式可得，上面两个 8 位二进制数相加的结果为 9 位，超出了 8 位数的表示范围。若仅取 8 位字长(00000100B)，结果就显然是错误的，这种情况就称为溢出。事实上，$(10110111)_2=(183)_{10}$，$(0100110)_2=(77)_{10}$，则 183+77=260，大于 8 位二进制数所能表示的最大值 255，所以最高位的进位(代表了 256)给丢失了，这样最后的结果就是 260－256=4，即 00000100B。

2. 无符号二进制数的溢出判断

令无符号二进制数加法(或减法)中最高有效位 D_i 的进(借)位为 C_i，则两个无符号二进制数相加(或相减)时，若最高有效位 D_i 产生进位(或相减有借位)，即 $C_i=1$，则产生溢出。比如在例 1-13 中，两个 8 位无符号二进制数相加，最高有效位(即 D_7 位)产生了进位 C_7，结果就出现溢出。

1.3.3 二进制数的逻辑运算

算术运算是将一个二进制数的所有位作为一个整体来考虑的，而逻辑运算则是对二进制数按位进行操作，这意味着逻辑运算没有进借位。基本逻辑运算包括"与"、"或"、"非"及"异或"4 种运算。

1. "与"运算

"与"运算的规则是按位相"与"。"与"运算符一般用符号"∧"表示。其规则为

$1 \wedge 1=1 \qquad 1 \wedge 0=0 \qquad 0 \wedge 1=0 \qquad 0 \wedge 0=0$

即参加“与”操作的两位中只要有一位为 0，则“与”的结果就为 0，仅当两位均为 1 时，其结果才为 1。(试比较二进制数的“与”运算规则和乘法运算规则，思考一下二者之间的相同之处和不同之处)。

【例 1-14】 计算 10110110B∧10010011B=(?)B

解：

$$\begin{array}{r} 10110110 \\ \wedge\, 10010011 \\ \hline 10010010 \end{array}$$

即 10110110B∧10010011B = 10010010B。

2. “或”运算

“或”运算的规则为按位相“或”。“或”运算符一般用符号“∨”表示。其规则为

$0 \vee 0=0 \qquad 0 \vee 1=1 \qquad 1 \vee 0=1 \qquad 1 \vee 1=1$

即参加“或”操作的两位中只要有一位为 1，则“或”的结果就为 1，仅当两位均为 0 时，其结果才为 0。试比较二进制数的“或”运算规则和加法运算规则，思考一下二者之间的相同之处和不同之处。

【例 1-15】 计算 11011001B∨10010110B=(?)B

解：

$$\begin{array}{r} 11011001 \\ \vee\, 10010110 \\ \hline 11011111 \end{array}$$

即 11011001B∨10010110B = 11011111B。

3. “非”运算

“非”运算的规则为按位取反，即 1 的“非”为 0，而 0 的“非”为 1。其运算符为一上横线(¯)。

$\overline{1}=0 \qquad \overline{0}=1$

【例 1-16】

解：只要对 11011001 按位求反即可

$\overline{11011001B}=00100110B$

4. “异或”运算

“异或”运算的规则是相异为 1，相同为 0。即进行“异或”操作的两个二进制位不相同时，结果就为 1；两位相同时，结果为 0。“异或”运算符用符号⊕表示。

$0 \oplus 0=0 \qquad 1 \oplus 1=0 \qquad 0 \oplus 1=1 \qquad 1 \oplus 0=1$

【例 1-17】 计算 11010011B⊕10100110B=(?)B

解：

$$\begin{array}{r} 11010011 \\ \oplus 10100110 \\ \hline 01110101 \end{array}$$

即 $11010011B \oplus 10100110B = 01110101B$。

试比较二进制数的“异或”运算规则和减法运算规则，思考一下二者之间的相同之处和不同之处。

1.3.4 基本逻辑门及常用逻辑部件

本小节介绍几种在后面的章节中将要用到的最常用的计算机基本逻辑部件，对已经学过数字电路的读者，可跳过本节。对这些逻辑部件，我们也仅是从应用的角度出发，只关心它们的逻辑功能和外部引线连接，而不关心其内部的电路构成。

1. 与门(AND gate)

与门是对多个逻辑变量进行“与”运算的门电路。对两个逻辑变量 A 和 B 的“与”操作，其结果 Y 可表示为

$$Y = A \land B$$

它们的关系也可从表 1-3 所示的真值表中清楚地了解到。即仅在输入 A 和 B 均为 1 时，输出 Y 才为 1，A 和 B 中只要一个为 0，则 Y 就等于 0。从电路的角度来说，若采用正逻辑，则仅当与门的输入 A 和 B 都是高电平时，输出 Y 才是高电平，否则 Y 就输出低电平。

在电路连接上，与门常用图 1-1 所示的逻辑符号表示。

表 1-3 与门的真值表

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

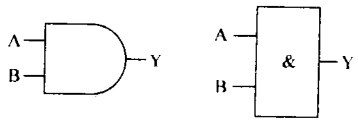

图 1-1 与门的逻辑符号

2. 或门(OR gate)

或门是对多个逻辑变量进行“或”运算的门电路。对两个逻辑变量 A 和 B 的“或”操作，它们的逻辑关系可用下边的运算表达式表示

$$Y = A \lor B$$

即两个输入变量 A 和 B 中任意一个为 1，输出 Y 就为 1；仅当 A 和 B 都为 0 时 Y 才为 0。从电路的角度来说，当或门的输入 A 和 B 只要有一个是高电平，输出 Y 就为高电平，否则 Y 就输出低电平。

或门的逻辑符号见图 1-2，其真值表见表 1-4。

表 1-4　或门的真值表

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

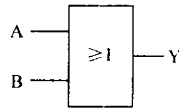

图 1-2　或门的逻辑符号

3. 非门(NOT gate)

非门又称为反相器，是对单一逻辑变量进行“非”运算的门电路，其输入变量 A 与输出变量 Y 之间的关系可用下式表示

$$Y=\overline{A}$$

非运算也称求反运算，变量 A 上的上划线“¯”在数字电路中表示反相之意。非门的逻辑符号见图 1-3，其真值表见表 1-5。

表 1-5　非门的真值表

A	Y
0	1
1	0

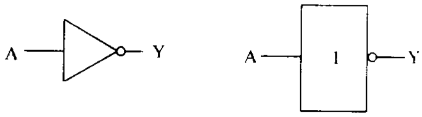

图 1-3　非门的逻辑符号

4. 与非门(NAND gate)

与非门是“与”门与“非”门的结合。若输入变量为 A 和 B，则先对输入 A 和 B 进行“与”运算，再对结果进行“非”运算，运算表达式为

$$Y=\overline{A \wedge B}$$

与非门的逻辑符号见图 1-4，图中逻辑符号图中的小圆圈表示“非”(本书将始终采用这种表示方法)。与非门的真值表见表 1-6。

表 1-6　与非门的真值表

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

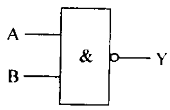

图 1-4　与非门的逻辑符号

5. 或非门(NOR gate)

和与非门类似，或非门是“或”门与“非”门的结合。即先对输入 A 和 B 进行“或”运算，再对其结果进行“非”运算，其运算表达式为