微型计算机原理及应用

刘彦儒　主编

机械工业出版社

微型计算机原理及应用

主　编　刘彦儒
副主编　肖　莹　毛徐辛
编　者　（排名不分先后）
苏　珣　李宝对　王　宁　蔡夕忠
李东瀛　王玉海　张力军　刘　哲
审　稿　脱介慈　陶　砂

机械工业出版社

本书以8086微处理器为背景阐述16位微处理器的结构、工作原理、指令系统、汇编语言程序设计、中断技术及接口技术；讲解以8086微处理器为核心构成的微型计算机的组成、工作原理和典型应用；较系统地介绍了32位微型计算机的软、硬件结构，适应当前微机发展的需要。本书适于职业教育使用，亦可供从事计算机应用工作的工程技术人员参考。

图书在版编目(CIP)数据

微型计算机原理及应用/刘彦儒主编．—北京：机械工业出版社，1998.7

ISBN 7-111-06608-1

Ⅰ.微… Ⅱ.刘… Ⅲ.微型计算机-基本知识 Ⅳ.TP36

中国版本图书馆CIP数据核字(98)第18510号

出 版 人：马九荣(北京市百万庄大街22号 邮政编码100037)
责任编辑：王冠宇 版式设计：李 悦 责任校对：姚培新
封面设计：陈 伟 责任印制：侯新民
机械工业出版社京丰印刷厂印刷·新华书店北京发行所发行
1998年7月第1版·1998年7月第1次印刷
787mm×1092mm 1/16·18.5印张·456千字
0 001—4 000册
定价：25.00元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

前　言

人类社会进入 20 世纪 70 年代以后，由电子计算技术和大规模、超大规模集成电路相结合而诞生的微型计算机，因其具有体积小、重量轻、功耗低、结构灵活、工作可靠和价格便宜等优点，发展迅速，应用广泛，已成为当代新技术革命的主要标志之一。

微型计算机技术发展速度很快，产品更新尤为迅速。字长从 4 位、8 位、16 位发展到 32 位，用了不到 20 年的时间。其应用已渗透到国民经济的各个领域，从科学计算、信息处理、办公事务管理到生产过程控制、仪器仪表制造，从民用产品到工业、军事各个领域，微型计算机正以其神奇的力量创造着人类的未来。

诞生于 80 年代的 IBM- PC 计算机（美国国际商业机器公司），以 16 位微处理器 Intel 8086/8088 为核心，有多种类型的“扩充件”可供选择，可以方便地加接各种外部设备，特别配备了极其丰富的系统软件和大量的应用软件，因此其用户也是最多的。近年来，32 位微处理器有了很大发展，其设计采用了小型机乃至大型机的结构和设计思想，但它们都兼容 16 位机。正是因为如此，职业教育的“微型计算机原理及应用”课程的教学内容应以 16 位微处理器来组织。但由于历史原因，职业教育教学中相当多的学校还以 8 位微处理器为中心来组织教学，远远落后于技术和应用的发展。鉴于以上情况，编者考虑了各类职业教育的实际和可能，认为编写一本实用的教材是非常必要的。

根据职业教育的特点，结合职业教育对计算机应用的要求，编写本书时，在各章节内容的安排及叙述方式、方法上都体现出了基本概念清晰、实用性强、以应用为目的的特点，叙述力求深入浅出、通俗易懂，尽量多举实例，尤其第八章，以实际工程应用为例单独组成一章，成为本书的特点。

本书在编写时，考虑了职业教育教学中《微型计算机原理及应用》的学时数，选编了一些实用性强的附录，每章都配有适量的习题与思考题。编者总结多年从事职业教育教学和实际工作的经验，参考大量的文献和资料，编写了这本适于职业教育的微机教材。本书各章节相对独立，在学时的安排上可以灵活取舍，以适应不同类型职业教育的需求。

考虑各校教学的实际，本书以 16 位微处理器为主要教学内容，也适当介绍以 80386 和 80486 为核心的 32 位微处理器的内容，这些内容都是依据职业教育的实际需要和教学实践来安排的。

本书适用于职业技术教育的计算机、自动化等专业使用，也可供工程技术人员参考。

本书由刘彦儒、肖莹、毛徐辛、苏珣、李宝对、王宁、蔡夕忠、李东瀛、王玉海、张力军、刘哲编写。刘彦儒主编，肖莹、毛徐辛副主编。脱介慈、陶砂审稿。杜西宁、李京、薛银安、朱海波、张军跃、周和生、白松岩等同志参加了编选习题、选定实验、上机验证等工作。在编写过程中，蒋湘若、陶砂、刘连青等同志提出许多宝贵意见和建议，在此表示感谢。由于水平所限，书中难免会有疏漏和不妥之处，敬请读者给予批评指正。

编者

1998 年 6 月

目　录

第一章　计算机基础知识

内 容 提 要

作为计算机的基础知识，这一章分四节，为初学者介绍了计算机中数的表示方法及运算，一般微型计算机的结构组成，微型计算机的基本工作原理和评价计算机性能的主要技术指标。

通过这一章的学习，能够使读者了解有关微型计算机的一些基本概念，了解微型计算机的硬件结构和基本工作原理，帮助读者对微型计算机形成一个总体的认识，为深入学习和研究以后章节的内容奠定必要的基础。

第一节　概　　述

一个完整的微型计算机系统由硬件系统和软件系统两部分组成。硬件系统是指组成微型计算机实体的机械和电子部件。软件系统是指为了运行、管理和维护微型计算机而编制的各种程序的集合。硬件系统是组成微型计算机的物质基础，软件系统是指挥硬件系统发挥其功能的命令信息。我们可以用图 1-1 简略地描述一个完整的微型计算机系统。图中示出的硬件系统和软件系统所分别包括的部分是这一节主要介绍的内容。

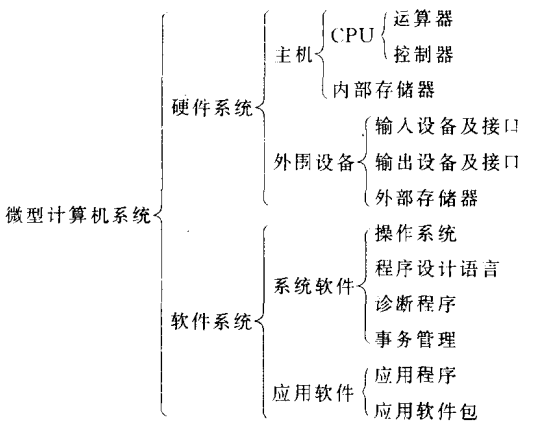


图 1-1　微型计算机系统示意图

一、硬件系统

微型计算机的机种和型号较多，但其基本结构都是由五部分组成，即：运算器、控制器、存储器、输入设备和输出设备，通常称为计算机的五大部件。下面就分别介绍这五大部件的基本功能和它们之间的相互联系。

1. 存储器

存储器的功能是用来存放程序和数据。程序是计算机操作的依据；数据是计算机操作的对象，它们都作为信息以二进制数据的形式存放在存储器中。

设在微型计算机主机内部的存储器称为内部存储器，简称内存。设在微型计算机主机之外的存储器称为外部存储器，简称外存。内存存储信息的容量是有限的，内存容量的大小是计算机的重要性能指标之一。外存因在主机之外（如磁盘、光盘等），所以理论上讲，外存的容量是无限的。

（1）内部存储器

内存由若干半导体存储器芯片组成，每个芯片都包含若干个存储单元，每个存储单元存放一个 8 位的二进制数据。一般将一个 8 位二进制数记作一个字节（Byte）。内存中可以存放

数据的总数称为内存容量或存储空间，通常以字节为单位。一千个字节，即 $2^{10}=1024$ 个字节记作 1KB；2^{20}个字节记作 1MB，而 1GB 是 1024MB 字节。内存单元按某种顺序的编号称为单元地址，CPU 通过地址识别不同的内存单元对其进行操作。如图 1-2 所示，访问内存时，先由地址译码器接收地址总线上送来的地址，经译码后找到相应的单元，再由读/写控制电路按读写命令完成读出或写入的操作，被读出或写入的数据通过数据总线实现和 CPU 之间的传送。

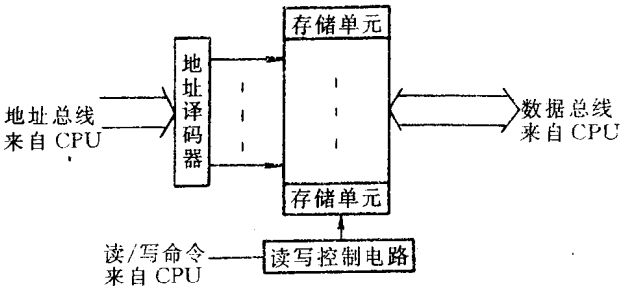


图 1-2　内存单元示意图

内存可以根据它的工作方式分为随机存储器（RAM）和只读存储器（ROM）两种。RAM 可以供用户随机地读出或写入信息，而 ROM 却只允许用户读出信息，不允许写入信息。因此，RAM 中通常存放的是用户程序和运算结果。而 ROM 中一般存放不需要经常改变的信息。

（2）外部存储器

最常用的外部存储器有软磁盘、硬磁盘和光盘。

软磁盘、硬磁盘和光盘存储器在使用时都要通过驱动器与主机接口。驱动器的主要作用是：将主机的命令翻译成控制驱动器的各种信号；实现磁盘与主机之间传送信息的串并转换；为主机提供驱动器的各种状态信息。

2. 运算器

运算器的功能是完成数据的算术运算和逻辑运算。通常，运算器由运算逻辑部件 ALU（Arithmetic Logic Unit）和寄存器组组成。如图 1-3 所示，ALU 是以全加器为基础，再配置移位、比较等其它控制逻辑构成的组合电路。能完成加、减、移位、比较等各种算术和逻辑运算。寄存器组由多个寄存器组成，用于存放运算操作数。在不同的机型中，寄存器的个数、名称和用途各不相同，但大致都可分为通用寄存器和专用寄存器两种。通用寄存器一般供用户调度各操作数时使用，而专用寄存器各自有专门的功用。

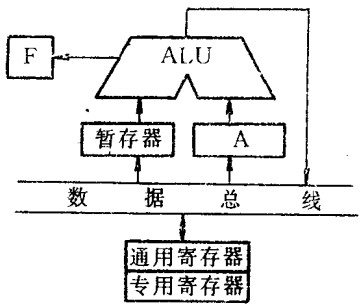


图 1-3　运算器简图

在寄存器组中有三个功能特殊的寄存器 A、F 和 IP。寄存器 A 称为累加器，用于存放一个参加运算的数据和运算后的结果。寄存器 F 称为标志寄存器，它用来专门记录每次运算结果的一些特征。如：结果是否为“0”，是否有进（借）位和溢出发生等。寄存器 IP 称为指令指针寄存器，用以存放预取指令的地址。CPU 根据 IP 中存放的地址到内存单元中取出指令后进行指令所规定的操作。

3. 控制器

控制器是全机的指挥中心，通常由数据寄存器（在 8086/8088 微处理器中为指令队列）、指令译码器、定时和逻辑控制电路组成。

控制器将指令从存储器中取出后存入数据寄存器，再经指令译码器译码后产生一系列操作命令，通过定时和控制电路发向各个部件，指挥各部件彼此配合，协调动作，在一定时间

内有序地完成规定的操作，即完成指令的执行过程。

在上述结构中，人们往往把内存、运算器和控制器合在一起称为主机；在主机中又常常把运算器和控制器合在一起称之为中央处理单元，即 CPU (Central Processing Unit)。在现代计算机技术中，将 CPU 的各部件集成在一块很小的半导体芯片上称为微处理器。

4. 输入设备

输入设备的功能是将待处理的信息传递到内存中。常见的输入信息有数字、字母、文字、图形、图象、声音等多种形式，而送入计算机的只有一种形式即二进制数据。因此，输入设备是变换输入信息形式的部件。它将各种不同形式的信息转换成二进制数据送入内存。常用的输入设备有键盘、鼠标器、光笔、扫描仪、模/数转换器等。

5. 输出设备

输出设备的功能是将计算机运算结果的二进制信息转换成其它设备能识别的形式后输出。常用的输出设备有打印机、显示器、数/模转换器等。

输入/输出设备（简称 I/O 设备）也常被人们统称为计算机的外围设备。它们与主机之间的连接都要通过专门的电路来完成。

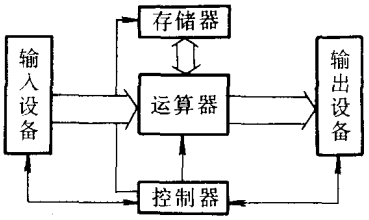


图 1-4 微机硬件系统结构示意图

上述五大部件之间的相互联系如图 1-4 所示：

二、软件系统

微型计算机的软件系统按照功能可以分为两大类：系统软件和应用软件。

1. 系统软件

系统软件的功能是对计算机系统进行管理、调度、监视和服务。其目的是方便用户，合理调用微机系统资源，扩充系统功能。系统软件包括操作系统、程序设计语言、诊断程序和事务管理。

2. 应用软件

应用软件包括用户程序和应用软件包。用户程序是用户为解决某个具体问题而编制的程序。应用软件包是为用户解决同类问题所提供的通用的处理程序。

第二节 计算机中数的表示方法及运算

人们在日常生活中使用最多的是十进位计数制，但在计算机内部因为只能采用电位的高低来表示数码 1 和 0，所以计算机只能使用二进位计数制。由于二进制数码冗长，书写和阅读都不方便，故我们在编程时常采用十六进制来表示。所以，用二进制数如何表示数值数据和非数值数据，二进制数与十进制数和十六进制数之间如何转换，以及二进制数的运算方法是本节研究的主要问题。

一、进位计数制及其相互转换

1. 进位计数制

大家都知道，十进位计数制 (Decimal) 是用十个不同的数码（即 0、1、2、…8、9）在不同的数位上按位权（同样的数码在不同的位上可以表示不同的数值，这称之为位权）来表示数的值。它的特征是“逢十进位”，每位的位权是以 10 为底的幂。任意一个十进制数 N_D 可

以表示为

$$N_D=\sum_{i=-m}^{n-1}D_i\times 10^i \tag{1-1}$$

其中，角标 D 表示十进制数，m 表示小数位的位数，n 表示整数位的位数，D_i 为十进制数字符号 0～9。例如，数值为 248.25 的十进制数可以按上式展开表示为

$$248.25=2\times10^2+4\times10^1+8\times10^0+2\times10^{-1}+5\times10^{-2}$$

与十进制数类似，二进位计数制（Binary）是用两个不同的数码（0、1）在不同的数位上按位权表示数的值。其特征是“逢二进位”，每位的位权是以 2 为底的幂。任意一个二进制数 N_B 可以表示为

$$N_B=\sum_{i=-m}^{n-1}B_i\times 2^i \tag{1-2}$$

其中角标 B 表示二进制数，B_i 为二进制数字符号 0 或 1。例如，数值为 248.25 的二进制数 11111000.01B 可表示为

$$11111000.01B=1\times2^7+1\times2^6+1\times2^5+1\times2^4+1\times2^3+$$
$$0\times2^2+0\times2^1+0\times2^0+0\times2^{-1}+1\times2^{-2}$$

同理，十六进制（Hexadecimal）是用十六个不同的数码（即 0、1、2、3…8、9、A、B、C、D、E、F）在不同的数位上按位权表示数的值。它的特征是“逢十六进位”，每位的位权是以 16 为底的幂。任意一个十六进制数 N_H 可以表示为

$$N_H=\sum_{i=-m}^{n-1}H_i\times 16^i \tag{1-3}$$

其中角标 H 表示十六进制数，H_i 为十六进制数字符号 0～F。例如数值为 248.25 的十六进制数为 0F8.4H 可以按上式展开表示为（按十六进制数的书写规定，若数的最高位是字母，应在字母前加"0"示意）

$$0F8.4H=15\times16^1+8\times16^0+4\times16^{-1}$$

在书写时，不同的数制要加以注明。常用的标记方法有两种，一种是把数加上括号，在括号的右下角标注数制代号，例如：$(37)_{16}$、$(1010)_2$、$(101)_{10}$等；另一种是用英文字母标记，加在数的后面，用 B 表示二进制数，用 D 表示十进制数，用 H 表示十六进制数。例如：37H、1010B、101D 等。其中，十进制数中的 D 可以省略不写。在本书中采用英文字母标记的方法。

2. 不同进位计数制之间的转换

（1）十进制数转换成二进制数

1）十进制整数转换成二进制整数　这种转换有几种方法，这里只介绍最常用的“除 2 取余法”。

“除 2 取余法”即用 2 不断地去除要转换的十进制数，直到商为 0。然后将所得的各次余数以最后得到的余数为最高位，依次排列起来就得到对应的二进制数。

例如：将 215 转换成二进制数。

215÷2＝107……余 1
107÷2＝53…… 余 1
53÷2＝26……余 1
26÷2＝13……余 0
13÷2＝6…… 余 1
6÷2＝3……余 0
3÷2＝1……余 1
1÷2＝0……余 1

转换结果：215＝11010111B

2）十进制小数转换成二进制小数　这里只介绍“乘 2 取整法”。

“乘 2 取整法”即用 2 不断地去乘所要转换的十进制数，直至小数部分等于零或满足要求的精度为止。把每次乘积的整数部分以最初得到的数为最高位，依次排列起来即得到结果。

例如：将 0.625 转换成二进制数。

0.625
× 2
1.250……整数部分为 1
× 2
0.5……整数部分为 0
× 2
1.0……整数部分为 1

转换结果：0.625＝0.101B。

要强调说明的是：在乘 2 运算过程中产生的整数位不继续参加运算。对于具有整数和小数两部分的十进制数，要分别将整数和小数部分转换成二进制数，然后合并起来即得到结果。

（2）二进制数转换成十进制数

二进制数转换成十进制数只要按位权展开相加即可。

例如：将 11111000.01B 转换成十进制数。

$$\begin{aligned}11111000.01\text{B} &= 1\times2^7+1\times2^6+1\times2^5+1\times2^4+1\times2^3+\\ &\quad 0\times2^2+0\times2^1+0\times2^0+0\times2^{-1}+1\times2^{-2}\\ &= 248.25\text{D}\end{aligned}$$

（3）十进制数转换成十六进制数

这种转换与十进制转换成二进制数方法相似，即采用“除 16 取余法”。

例如：将 3901 转换成十六进制数。

16 | 3901
16 | 243……余 13，写作 D
16 | 15……余 3，写作 3
0……余 15，写作 F

转换结果：3901＝0F3DH

十进制小数转换成十六进制小数的方法，读者可以参考十进制与二进制小数转换的方法自行演算“乘 16 取整法”。

(4) 十六进制转换成十进制数

这种转换与二进制转换成十进制的方法相似，即把十六进制数按位权展开相加即可。

例如：$0F3DH=15\times16^2+3\times16^1+13\times16^0=3901D$

(5) 二进制数与十六进制数的转换

例如：将 1101011.100101B 转换成十六进制数。

方法：01101011. 10010100

6 B . 9 4

转换结果：1101011.100101B=6B.94H

从上例看出，将二进制数转换成十六进制数的方法是：从小数点开始，分别向左向右每四位一组，在最高位和最低位处，不足四位的用零补足，然后分别把每组数用十六进制数表示，并按序相连。

同理，将一个十六进制数转换成二进制数的方法是：把十六进制的每一位用四位二进制数表示，然后将各位按序相连。

例如：将 3A.B5H 转换成二进制数

3 A . B 5

00111010. 10110101

转换结果：3A.B5H=111010.10110101B

二、二进制数的运算

1. 基本运算

(1) 算术运算

1) 加法 二进制数加法运算的规则为

0+0=0

0+1=1

1+0=1

1+1=0 (进位 1)

例：

```
  1010B
+1001B
 10011B
```

2) 减法 二进制数减法运算的规则为

0−0=0

1−0=1

1−1=0

0−1=1 (有借位)

例：

```
  1110B
−1001B
  0101B
```

3) 乘法 二进制数乘法运算的规则为

0×0=0

0×1=0

1×0=0

1×1=1

例：

```
   1110B
×1001B
   1110
  0000
 0000
+1110
1111110B
```

4）除法　二进制除法是乘法的逆运算，运算规则与十进制相似。

例如：

$$\begin{array}{r} 111\text{B} \\ 101\text{B}\overline{)100011\text{B}} \\ \underline{-101} \\ 0111 \\ \underline{-101} \\ 0101 \\ \underline{-101} \\ 0 \end{array}$$

（2）逻辑运算

1）“与”运算（AND）　逻辑“与”运算与算术运算中的乘法运算相似，所以也称逻辑乘，运算符为“·”或“∧”。其运算规则为：“0”和任何数相“与”其结果为“0”。只有“1”和“1”相“与”结果为“1”。

2）“或”运算（OR）　逻辑“或”运算与算术运算中的加法运算相似，因此也称逻辑加。运算符为“+”或“∨”。其运算规则为：“1”和任何数进行“或”运算其结果为“1”。只有“0”和“0”相“或”结果为“0”。

3）“非”运算（NOT）　逻辑“非”运算实际上是将逻辑变量取反。可用逻辑变量上的“—”表示取原来状态的相反状态。

4）“异或”运算（XOR）　“异或”运算的规则是：两个逻辑变量的取值不同时，运算结果为“1”；两个逻辑变量的取值相同时，运算结果为“0”。运算符为“⊕”或“∀”。

以上四种逻辑运算都是按位进行的。

2. 带符号数的运算

在数学运算中，表示一个数的正负，可以在数的前面冠以正负号。而在计算机中，数的正负号也只能用“1”和“0”两个数码来表示。例如，在字长为8位的二进制数中，把它的最高位规定为符号位，“1”表示负，“0”表示正，其余各位为数值位，用来表示数的值，这样的数称为带符号数。

同一个二进制数，例如10001010B，如果作为无符号数其值为138，如果作为带符号数其值为−10。

（1）带符号数的表示方法

为了运算方便，带符号数有三种表示法：原码表示法、反码表示法和补码表示法。

1）原码表示法　正数的符号位为“0”，负数的符号位为“1”，其余为数值位，这种表示数的方法就称原码表示法。数 X 的原码记作 $[X]_{原}$。我们可以把原码定义为

$$[X]_{原}=\begin{cases} X & 0\leqslant X\leqslant 2^{n-1}-1 \\ 2^{n-1}-X & -(2^{n-1}-1)\leqslant X<0 \end{cases} \tag{1-4}$$

“n”为包括符号位和数值位的总位数。

例如：当 $X=+10$ 和 $X=-10$ 时

$[+10]_{原}=00001010\text{B}$

$[-10]_{原}=10001010\text{B}$

由此可见，用原码表示数的范围是 $-(2^{n-1}-1)\sim+(2^{n-1}-1)$。8位（$n=8$）二进制数原码

的表示范围是－127～＋127，16 位(n=16)二进制数原码的表示范围是－32767～＋32767。

2）反码表示法　正数的反码与其原码相同。负数的反码为原码的数值位按位取反（注意：符号位不变）。数 X 的反码记作 $[X]_{反}$。我们可以把反码定义为

$$[X]_{反}=\begin{cases}X & 0\leqslant X\leqslant 2^{n-1}-1\\ (2^n-1)+X & -(2^{n-1}-1)\leqslant X<0\end{cases} \tag{1-5}$$

例如：当 X＝＋10 和 X＝－10 时

$[+10]_{反}=[+10]_{原}=00001010B$

$[-10]_{反}=11110101B$

由此可见，反码表示数的范围与原码相同。

3）补码表示法　正数的补码与原码相同。负数的补码为其反码在最低位加“1”（符号位不变）。数 X 的补码记作 $[X]_{补}$。我们可以把补码定义为

$$[X]_{补}=\begin{cases}X & 0\leqslant X\leqslant 2^{n-1}-1\\ 2^n+X & -2^{n-1}\leqslant X<0\end{cases} \tag{1-6}$$

例如：当 X＝＋10 和 X＝－10 时

$[+10]_{补}=[+10]_{原}=00001010B$

$[-10]_{补}=[-10]_{反}+1=11110110B$

由此可见，补码表示数的范围是：-2^{n-1}～＋($2^{n-1}-1$)。8 位(n=8)二进制补码表示数的范围是－128～＋127，16 位(n=16)二进制补码表示数的范围是－32768～＋32767。

（2）补码运算

1）补码加法　对于任意数 X、Y，补码加法的法则是

$$[X]_{补}+[Y]_{补}=[X+Y]_{补} \tag{1-7}$$

我们用三个例子来验证上式的正确性。

已知：$[+34]_{补}=00100010B$

$[-34]_{补}=11011110B$

$[+55]_{补}=00110111B$

$[-55]_{补}=11001001B$

例 1：00100010＝$[+34]_{补}$

＋）00110111＝$[+55]_{补}$

01011001＝$[+89]_{补}$

例 2：11011110＝$[-34]_{补}$

＋）11001001＝$[-55]_{补}$

1 10100111＝$[-89]_{补}$

例 3：00110111＝$[+55]_{补}$

＋）11011110＝$[-34]_{补}$

1 00010101＝$[+21]_{补}$

从上例可以看出，不管两个相加的数是正数还是负数，只要直接用它们的补码运算（包括符号位），当结果不超出补码的表示范围时，运算结果都是正确的。后两例中最高位向更高位的进位，由于字长的限制自然丢失，不会影响结果的正确性。

2）补码减法　对任意数 X、Y，补码减法的法则是：

$$[X]_{补}-[Y]_{补}=[X]_{补}+[-Y]_{补}=[X-Y]_{补} \tag{1-8}$$

我们仍用三个例子来验证上式的正确性。

求：＋34－(－55)，55－34，－55－(－34)的值

$00100010=[+34]_{补}$	$00110111=[+55]_{补}$	$11001001=[-55]_{补}$
$+)00110111=[+55]_{补}$	$+)11011110=[-34]_{补}$	$+)00100010=[+34]_{补}$
$01011001=[+89]_{补}$	$\boxed{1}\ 00010101=[+21]_{补}$	$11101011=[-21]_{补}$

可见，无论减数和被减数是正数还是负数，只要将减数 Y 的补码 $[Y]_{补}$ 转换成 $[-Y]_{补}$，然后与被减数做加法，当结果不超出补码的表示范围时，就可以得到正确的运算结果。这里要特别强调的是：将 $[Y]_{补}$ 转换成 $[-Y]_{补}$ 的方法是将 $[Y]_{补}$ 连同符号位一起取反后加 1 就得到 $[-Y]_{补}$。

由上述分析可以看出，采用补码运算使计算机中数的运算得到简化。第一，把减法变为加法运算，从而省略了减法器。第二，无符号数和带符号数的加法可以用同一加法器完成（因为补码运算中符号位不需要单独处理）。

3）溢出　采用补码运算时，若结果的数值超出了补码能表示的范围，计算结果会发生错误。我们称这种情况为溢出。

请看下面两例：已知$[+124]_{补}=01111100B$

$[-124]_{补}=10000100B$

$[-21]_{补}=11101011B$

①求	-124	$10000100=[-124]_{补}$	②求	$+124$	$01111100=[+124]_{补}$
	$+)-21$	$+)11101011=[-21]_{补}$		$-)-21$	$+)00010101=[+21]_{补}$
	-145	$\boxed{1}\ 01101111=[+110]_{补}$		$+145$	$10010001=[-111]_{补}$

从以上两例看到，补码运算的结果是错误的。这是因为运算结果超出了补码的表示范围，运算发生溢出。溢出与进位的自然丢失是不同的概念，进位的自然丢失不影响结果的正确性，而溢出使运算结果必然错误。

我们可以根据运算数最高位和次高位的进位状态来判断是否有溢出发生。如果用 C_Y 表示最高位向更高位的进位状态，C_P 表示次高位向最高位的进位状态，1 表示有进位，0 表示无进位。用 V 表示溢出，$V=1$ 表示有溢出，$V=0$ 表示无溢出，则运算结果是否发生溢出可用下式判断：$V=C_Y\oplus C_P$ 即：最高位和次高位同时有进位或同时无进位时无溢出，两位中只有一位发生进位时有溢出。

3. 定点数与浮点数

在前面的讨论中我们没有涉及小数及小数点在机器中如何表示的问题。而实际上计算机处理的数据中大部分是带有小数的。因此，在计算机中采用两种方法表示数据，一种是定点表示法，另一种是浮点表示法。

（1）定点表示法

定点表示法是将小数点的位置固定不变，因而不必再使用记号来表示小数点，只将其按共同的约定隐含在数位间就可以了。

一般来说，小数点可以固定在任何位置。最常采用的两种约定是：

1）约定小数点隐含在最低数值位之后。这使得所有数值位表示的数为纯整数，称为定点整数。

2）约定小数点隐含在最高数值位之前。这使得所有数值位表示的数为纯小数，称为定点小数。

我们在前面讨论的所有 n 位二进制数，实际上是采用了第一种约定。

n 位原码定点整数表示数的范围我们在前面已经分析过，即：$-(2^{n-1}-1)\leqslant X\leqslant 2^{n-1}-1$。$n$ 位原码定点小数表示数的最大值为：$1-2^{-(n-1)}$；最小值为：$-(1-2^{-(n-1)})$，所以 n 位原码定点小数表示数的范围是：$-(1-2^{-(n-1)})\sim 1-2^{-(n-1)}$。因此定点数表示法虽然简单、直观，但表示数的范围小。

(2) 浮点表示法

为了扩大计算机表示数的范围，还可以采用数的浮点表示法，即把任意一个数通过移动小数点位置再乘以相应比例因子的办法表示。例如大家熟悉的：$248.25=0.24825\times 10^{3}=24825\times 10^{-2}$，我们可以把任意一个二进制数 N_B 表示成：

$$N_B=S\times 2^{J} \tag{1-9}$$

其中：S 称做 N_B 的尾数，是数值的有效数字部分。J 称做 N_B 的阶码，是有符号的整数。一般情况下，S 应取纯小数的形式。

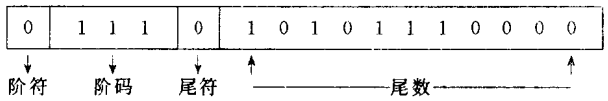
例如：1010111B 应写做 $0.1010111B\times 2^{7}$

101.00111B 应写做 $0.10100111B\times 2^{3}$

0.01010111B 应写做 $0.1010111B\times 2^{-1}$

浮点数在机器中的表示方法是：对于一个 n 位字长的机器来说，高 h 位用来表示浮点数的阶码，其中最高位表示阶符，0 表示正，1 表示负，另外 $h-1$ 位表示阶数。低 m 位用来表示浮点数的尾数，其中最高位表示尾符，其它 $m-1$ 位表示尾数的值。($h+m=n$)

例如：$0.1010111B\times 2^{7}$ 表示在字长 $n=16$ 的机器中为：



显然，当机器的字长一定时，用于表示阶码的位越多，则能够表示数的范围越大。而用于表示尾数的位数越多，则所能表示数的精度越高。因此，究竟阶码和尾数分别用多少位，要根据数的表示范围和精度要求合理分配。

有浮点数运算过程中，为了使有效数字不致丢失，要求尾数的最高位为非 0 数码，也就是对浮点数进行规格化。对于原码来说，要求尾数的最高位为 1，即尾数的形式为 0.1XX…X。“X” 表示任意值。如果是补码，若为正数与原码的要求相同。若为负数，则要求尾数的最高位必须是 0，即尾数的形式为 1.0XX…X。

例如，若 $[X]_{原}=0.000101B$，其规格化后应表示为：$[X]_{原}=0.101B\times 2^{-3}$

若 $[X]_{补}=1.000101B$，其规格化后应表示为：$[X]_{补}=1.0101B\times 2^{-2}$

三、BCD 码和文字符号代码

1. BCD 码（二—十进制编码）

BCD 码 (Binary Coded Decimal) 又称二—十进制码。它是用二进制的形式表示十进制数的一种编码。编码的方法是：以四位二进制数为一组，取 0000～1001 共 9 种状态对应表示十进制数码 0～9，每组表示十进制数的一位。

例如：十进制数 89.7 用 BCD 码可表示为 10001001.0111 (BCD)

BCD 码在运算时要遵循逢十进一的原则。因此，两个 BCD 码相加时要按四位一组逐组相

加，一组的和大于 9 时要做加 6 的修正。两个 BCD 码相减时，组与组之间若有借位则要做减 6 的修正。

例如：求两个 BCD 码 01011000 与 00110111 的和。

则：

01011000	被加数 58
+)00110111	加数 37
10001111	个位组和大于 9
+) 0110	加 6 修正
10010101	和 95

求两个 BCD 码 10000101 与 00101000 的差。

则：

10000101	被减数 85
−)00101000	减数 28
01011101	组间有借位
−) 0110	减 6 修正
01010111	差 57

2. 文字符号代码（ASCII 码）

ASCII 码（American Standard Code for Information Interchange）即美国信息交换标准代码。这是一种七位二进制编码，用来表示 128 个计算机常用的字符，如“回车”（CR）、“换行”（LF）、各种标点符号、大小写英文字母等。因为 8 位二进制数为一个字节，所以 ASCII 码在使用时，常在最高位用“0”补足或加奇偶校验位。ASCII 码表详见附录 A。

第三节　微型计算机的基本工作原理

对于操作者来讲，使用计算机工作的过程可以粗略地描述为以下几个步骤：

①根据所要解决的问题编制程序。

②通过输入设备（如键盘）将程序存入内存。

③执行程序。

④通过输出设备（如显示器或打印机）输出运行结果。

上述步骤中的第③步，即程序被启动后在计算机内部的运行过程是自动的，对操作者来说是不可见的。为了让读者初步了解计算机是怎样完成这个自动过程的，我们在这一节中通过一条指令的执行简略地进行描述。

计算机的工作过程，实质上是计算机的硬件设备在系统软件的支持下，逐条执行指令的过程。在计算机执行指令的过程中，CPU 总是有规则地重复以下工作步骤：

①从存储器中取出一条指令。

②分析指令的操作码。

③取操作数。

④执行该指令要求的操作动作。

⑤存储或输出结果。

我们将上述步骤划分为三个阶段，即取指令阶段、指令译码阶段和指令执行阶段。因此，计算机在执行程序时，始终是周而复始地重复这三个过程，直至程序结束。

第四节　评估计算机的主要技术指标

评估一台微型计算机的性能有许多技术指标，在这一节中，我们列出最主要的几项技术指标。

一、字长

微机的字长是指在CPU内部同时并行传递和运算的数据的位数。各类微机的字长不同，有4位、8位、16位、32位等。字长直接影响运算精度和运算速度。字长越长，运算精度越高。经推算，若要保证K位十进制数的运算精度，字长至少要取3.3×K位。另外，一个16位的数，对8位机来说，需要进行两次传送处理，而对16位机则只需一次。这就是字长长的机器在处理速度上所具有的优越性。

字长是由微处理器对外数据通路的数据总线条数和微处理器内部结构决定的。数据总线条数和内部结构均为16位的计算机称为16位机。而对外数据总线只有8条，内部结构为16位的微处理器称为准16位机。

二、主频（时钟周期）

主频是指CPU在单位时间内平均要“动作”的次数，以兆赫（MHz）为单位。主频越高，微机的运算速度越快。

三、访存空间

访存空间是指微处理器所能访问的内存单元总数，是由地址总线的条数决定的。8位微处理器的地址总线有16条，可寻址的内存单元2^{16}=65536个，也称64KB。16位微处理器有20条地址线，访存空间为2^{20}=1048576=1024KB，也称1MB。高档微机的访存空间可达1GB=1024MB。

四、指令数

一台微机可以有几十到几百种指令，每种指令可以指挥计算机完成一种操作，机器能完成的操作种类越多，即指令数越多，该微机系统的功能就越强。

五、基本指令执行时间（运算速度）

微机工作的过程是执行指令的过程，所以微机执行一条指令所用的时间长短反映了工作速度。但各种指令的执行时间是不一样的。为了衡量微处理器的速度，选择寄存器加法指令作为基本指令，它的执行时间就作为基本指令执行时间。基本指令执行时间与其所用的时钟周期数和时钟周期有关。基本指令执行时间越短，微处理器的运算速度越快。

六、是否能构成多处理器系统

是否能构成多处理器系统是指微处理器是否有协处理器接口，是否能连接协处理器将主处理器的某些任务分由协处理器去完成，使整个系统的功能上百倍地增加。16位以上的微处理器一般具有这项功能。

习　题

1. 将下列数按数值大小排列①01101001B　②（01101001）BCD　③233H　④10010101B　⑤96H　⑥233

2. 将下列十进制数转换为二进制（八位）、十六进制数和BCD码。