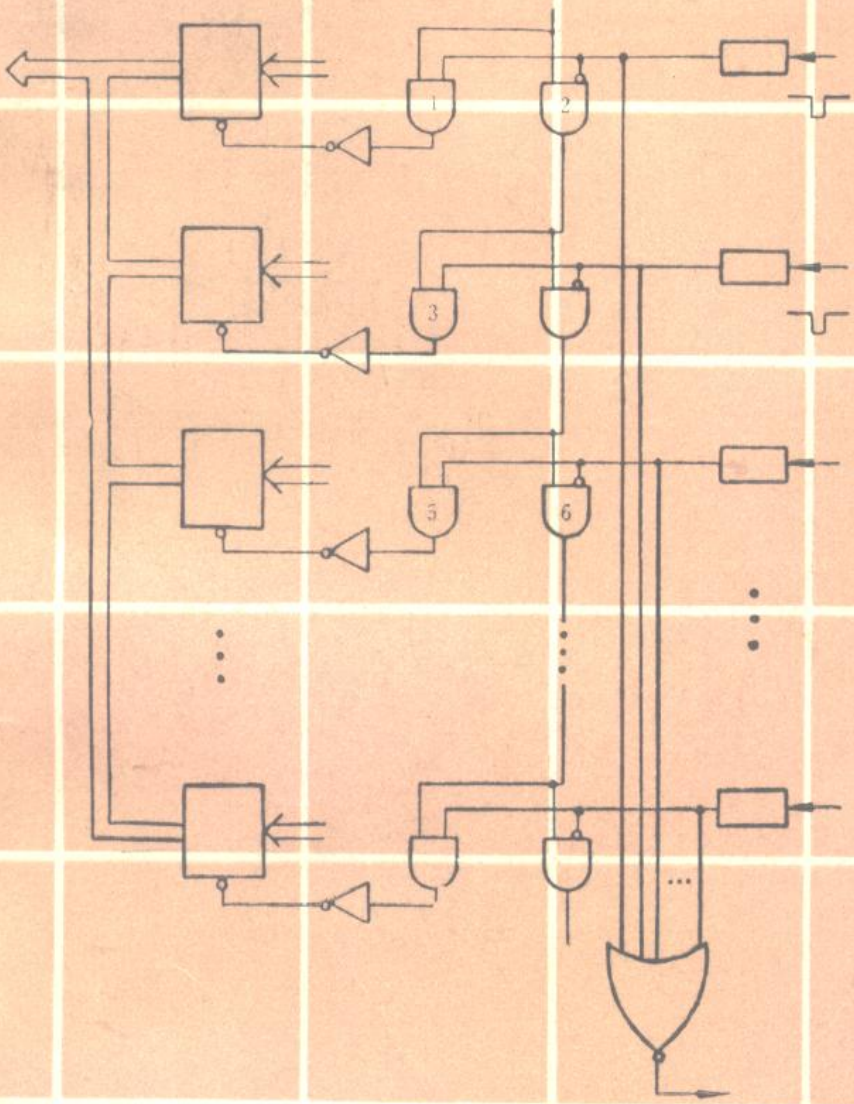
微型计算机及其应用

职工高等学校试用教材



上海科学技术文献出版社

职工高等学校试用教材

微型计算机及其应用

叶祥生　主编

上海科学技术文献出版社

职工高等学校试用教材
微型计算机及其应用
叶祥生　主编

上海科学技术文献出版社出版发行
（上海市武康路2号）
新华书店经销　昆山亭林印刷厂印刷

开本 787×1092 1/16　印张 20　字数 499,000
1987年10月第1版　1987年10月第1次印刷
印数：1—10,500

统一书号：15192·457　定价：4.70元
《科技新书目》120-208

前　言

为适应当前职工高校教学工作的实际需要，上海市高等教育局、上海教育局和江苏省教育厅于一九八四年三月在无锡市召开职工高校教材编写工作会议，会上提出把“精选内容，打好基础，加强实践，利于教学”作为编写职工高校教材的十六字方针。本教材是按会议的决定，参照一九八三年十一月教育部成人教育司在无锡会议上审定通过的全国职工高校“微型计算机原理及其应用”教学大纲，并在总结多年教学实践的基础上，结合成人教育特点编写的。

本教材共七章。前五章主要讲述微型计算机的基本原理、微处理器、存贮器、汇编语言程序设计、输入输出和接口技术。由于本课程的主要目的是“如何使用微型计算机”，因此在编写本教材时着重从应用的角度来分析和解决问题，如在硬件部分，不是详尽地分析微型计算机的内部结构和各功能部件电路，而是着重了解和掌握各功能部件的“外”特性，如 CPU 的引脚功能，存贮芯片和接口芯片的使用方法，以便实现存贮器、接口和微型计算机系统的硬件设计。在软件部分则是着重介绍汇编语言的程序设计技巧，使读者具有一定的编程能力。编写第六章的目的则是使读者对微型计算机系统有一个整体的认识，而第七章则是通过一些简单的应用实例为读者提供微机应用入门原理。

本课程的参考教学时数为 80～100 学时。由于本课程实践性很强，在教学中应加强实验和实践环节，有条件的学校可开设“课程设计”。

本课程应在“数字电子技术基础”课程后开设，并应有 A/D、D/A 转换原理知识。

本教材是为职工高校电子类有关专业的“微型计算机及其应用”课程而编写的，但也可作为具有数字电路知识的工程技术人员自学之用。

本教材由叶祥生主编，第一章由钱敏皓编写，第二章由王家乐、叶祥生编写，第三、五、七章由叶祥生编写，第四、六章由杨浩樑编写。

本教材由合肥工业大学张奠成教授主审，在本书编写过程中，张奠成教授始终给予积极的支持和帮助，提出了不少宝贵的意见，在此深表感谢。

在本书的编写和试用过程中得到上海第二工业大学计算机系、自动控制系、应用电子系各有关教研室教师的支持，陈益康教授等教师在试用本教材的过程中提出了不少有益的修改意见，陈淑萍同志绘制了本书的插图，在此一并表示感谢。

由于编者学识浅薄，时间仓促，书中定有不少错误和缺点，恳切希望读者予以指正。

编　者

1985.3

目　录

第一章 基础知识

§1-1 概 述

电子数字计算机(简称计算机)是一种能高速、自动地进行算术、逻辑运算和信息处理的工具。目前,它的应用已深入到国防、科研、工农业生产及生活等领域,并把其作为衡量一个国家现代化水平的重要标志。

一、微型计算机发展概况

从计算机发展过程来看,它已经历了四个阶段。第一代计算机(1946 年至 1956 年),其器件主要是电子管。第一台电子计算机"ENIAC",用了 18000 多只电子管,1500 多只继电器,重约 30 吨,耗电约 140 千瓦,占地 30 平方米,运算速度为 5000 次/秒,用机器语言和汇编语言编程。第二代计算机(1956 年至 1964 年),其器件主要是晶体管,它以磁芯存贮器为主存贮器,辅助存贮器已开始采用磁盘,运算速度为每秒数十万次,开始使用程序设计语言和操作系统。第三代计算机(1965 年至 1970 年),其器件主要采用中小规模集成电路,运算速度提高到每秒数百万次,而且操作系统得到进一步发展,建立了数据库及数据库管理系统,软件功能大大提高。从七十年代初进入了计算机的第四代,其器件主要是大规模和超大规模集成电路,其特点是硬件与软件有更多的结合,为了提高计算机的功能,产生了计算机网络。当前,已进入以人工智能为主要特征的第五代计算机的研制阶段。

随着大规模和超大规模集成电路工艺的发展,计算机已进入微型化的时代。1971 年美国 INTEL 公司研制成功四位微处理器 4004,不久又推出 4040 微处理器,它们虽均是四位微处理器,但如果配上相应的存贮器、输入输出接口和外部设备就能组成四位微型计算机,1972 年该公司又研制出八位微处理器 8008,以上这些微处理器称为第一代微处理器。1973 年 INTEL 公司研制出八位微处理器 8080,这标志着微处理器进入了第二代。在此之后,许多半导体制造厂商相继推出了各种不同型号的八位微处理器,如 MOTOROLA 公司的 M6800 等。1976 年美国 ZILOG 公司研制出 Z-80 八位微处理器,这是功能最强的八位微处理器,1976 年以后,各半导体制造厂商又陆续向市场推出单片微型计算机芯片,这些芯片上包含了处理器、存贮器和输入输出接口,有的甚至包含了 A/D 和 D/A 接口,并出现了一位微处理器。1978 年以后,微处理器进入了第三代,不少厂商一方面把性能较好的小型机微型化,另一方面研制了功能强的十六位微处理器,如 INTEL 公司的 8086 和 8088,MOTOROLA 公司的 68000,ZILOG 公司的 Z-8000 等。近年来,不仅研制成功了三十二位微处理器,而且还正在研制功能更强的新型微处理器。尽管微处理器和微型计算机的发展历史只有十几年,但其发展的速度却是惊人的,平均三年左右就换代一次。今后,随着半导体制造技术和计算机技术的发展,微型计算机一定会得到进一步的发展和应用。

二、微型计算机的特点

电子数字计算机的主要特点是：

(1) 运算速度快　一台每秒运算一百万次的计算机，其一分钟完成的工作量，相当于一个人用算盘运算几十年的工作量。特别是一些计算量大，时间要求又短的计算，使用计算机的意义就更大，如气象数据的计算、运载火箭的发射及其运行轨迹的修正的计算等。

(2) 计算精确　从原理上讲，计算机的精度可以不受限制，但是对于某一台计算机，其精度是确定的。字长愈长，精度愈高。

(3) 自动化程度高　计算机具有记忆能力，它能把原始数据、计算程序(即计算步骤)事先存贮起来，于是，执行计算程序的过程，就是按事先安排的步骤自动计算的过程，计算机还有逻辑判断及自动选择的能力，所以在计算机正常工作过程中能摆脱"人"的参与。

(4) 通用性强　一台计算机既可以用于数值计算、数据处理、自动控制，还可以用于辅助设计、人工智能等方面，因而，具有很强的通用性。

此外，微型计算机还具有本身的特点：

(1) 体积小、价廉　由于微型机的器件是大规模、超大规模集成电路，体积小是显而易见的，随着集成电路集成度的提高，体积会更小；另外，价格便宜。

(2) 可靠性高　微型机采用大规模、超大规模集成电路，其各部件之间连接接头及连线大大减少，因而焊接点减少，故障几率也随之减小；另外，由于集成技术的提高，芯片损坏率下降，微机的失效率已减小到 10^{-8}/小时。

(3) 使用简单、容易掌握　采用大规模或超大规模集成电路技术，就能将计算机的功能块集中在一块或几块芯片上，给微机系统的设计、应用及维修带来了方便。

(4) 功耗低　微型机和功能相近的小型机相比，功耗低了很多，这一特点不仅符合节能要求，而且使元、器件受热影响减小，延长了使用寿命。

由于上述特点，再加上微型机的配套芯片种类多，功能日趋齐全，结构简单灵活，对环境也无过高要求，因此，微型机得到广泛应用。

§1-2 计算机中数的表示方法

一、进位计数制

进位计数制是一种计数的方法，有二进制、八进制、十进制、十二进制、十六进制等。人们习惯使用十进制计数法，而计算机采用二进制计数法。由于二进制数位数较长，书写时易出错，又不易记忆，所以，在编写程序时，往往用八进制数或十六进制数来代替二进制数的书写。

(一)计算机中常用的几种数制

1. 十进制数

区别某种计数制的性质是通过它的基数(又称底数)来鉴别的。基数就是用来表示数值大小的数字符号的个数。十进制是用 0～9 十个数字符号表示数值的大小，所以十进制的基数为 10，其进位是逢十进一。在一个十进制数中，每个数字符号在不同的位置，它所表示的

值是该位置所处的"权"乘以这个数符。

"权"是数的某一位置所表示的值。例如：

十进制数：5　5　5·5　2　5

各位的权：10^2　10^1　10^0　10^{-1}　10^{-2}　10^{-3}

上式中从高位开始，第一个5在百位(即10^2为该位的权)，它表示500；第二个5在十位(10^1为该位的权)，它表示50；第三个5在个位(10^0为该位的权)，它表示5；第四个5在十分之一位，第五个5在千分之一位，它们的"权"分别为10^{-1}和10^{-3}，它们的数值分别表示为0.5及0.005。

通常，任意一个数N可表示为：

$$N=\pm[a_{n-1}\cdot R^{n-1}+a_{n-2}\cdot R^{n-2}+\cdots+a_1\cdot R^1+a_0\cdot R^0+a_{-1}\cdot R^{-1}+\cdots+a_{-m}\cdot R^{-m}]$$

$$=\pm\sum_{i=-m}^{n-1}[a_i\cdot R^i]$$

式中，m、n为幂指数，均为正整数，n为整数部分的位数，m为小数部分的位数；a_i为系数，对于基数为10的十进制数的系数a_i，可在0，1，2，…，9十个数字符号中任意取数；R_i称为基数，表示进位制，若R等于10，即为十进制。

所以，任何一个十进制数可表示为：

$$N=\pm[a_{n-1}\cdot 10^{n-1}+a_{n-2}\cdot 10^{n-2}+\cdots+a_0\cdot 10^0+a_{-1}\cdot 10^{-1}+\cdots+a_{-m}\cdot 10^{-m}]$$

$$=\pm\sum_{i=-m}^{n-1}[a_i\cdot 10^i]$$

例如：$555.525=5\times10^2+5\times10^1+5\times10^0+5\times10^{-1}+2\times10^{-2}+5\times10^{-3}$

2．二进制数

二进制数只有0和1两个数字符号，它的基数为2，进位是逢二进一。

二进制数的一般表示式为：

$$N=\pm[a_{n-1}\cdot 2^{n-1}+a_{n-2}\cdot 2^{n-2}+\cdots+a_0\cdot 2^0+a_{-1}\cdot 2^{-1}+\cdots+a_{-m}\cdot 2^{-m}]$$

$$=\pm\sum_{i=-m}^{n-1}[a_i\cdot 2^i]$$

其中，a_i为系数，只能为0或1。

十进制数1，在二进制中也写成1；十进制数2，在二进制中就要向高位进一，故写成10；以此类推，3——11，4——100，5——101，6——110，7——111，8——1000，9——1001，等等。可见，不同位置上的二进制数字，其值也是由"权"决定的。

例如，二进制数：1　0　1　1．0　1　1

各位的权：2^3　2^2　2^1　2^0　2^{-1}　2^{-2}　2^{-3}

二进制数的优点：

(1) 二进制数只有1和0两个数字符号，这两个数字符号可以表示为电路的接通和断开、电压的高和低、灯的亮和暗、物体的有和无等等，而且，这些物理状态很容易实现。

(2) 二进制数只有两种状态，出错几率小，可靠性高。

(3) 二进制数运算简单。

二进制数运算公式简单。以加法、乘法为例：

0+0=0	0×0=0
0+1=1	0×1=0
1+0=1	1×0=0
1+1=0（向高位进1）	1×1=1

即加法四条，乘法四条。

例1 二进制数加法：

```
  110110   ——被加数
  101010   ——加数
+ 11110    ——进位
---------
 1100000   ——和
```

例2 二进制乘法：

```
    1101    ——被乘数
  ×1011     ——乘数
---------
    1101  ┐
    1101  │
    0000  ├——部分积
   1101   ┘
---------
10001111    ——积
```

(4) 二进制数可节省存贮空间。

由于二进制数具备了这些优点，计算机就采用了二进制数字系统。

3. 八进制数

八进制是用0～7八个数字符号来表示数值的大小，它的基数为8，其进位是逢八进一。和十进制数、二进制数一样，不同位置上的八进制数字，其值也是由"权"决定的。

例如，八进制数：1 0 5 3 7 . 5 2 4

各位的权：8^4 8^3 8^2 8^1 8^0 8^{-1} 8^{-2} 8^{-3}

4. 十六进制数

十六进制是用0～9以及A～F十六个不同的数字符号来表示数值的大小，它的基数为16，其进位是逢十六进一。同样，不同位置上的十六进制数字，其值也是由其对应的"权"决定的。

例如，十六进制数：3 4 A 0 E. 1 F C

各位的权：$(16)^4$ $(16)^3$ $(16)^2$ $(16)^1$ $(16)^0$ $(16)^{-1}$ $(16)^{-2}$ $(16)^{-3}$

十进制数、二进制数、八进制数以及十六进制数之间的关系如表1-1所示。从表中可见，四位二进制数用一位十六进制数表示即可。因而，在微型计算机中，为了方便，经常用十六进制数来写地址、数据和指令代码。

注意：在计算机中为了区分不同数制的数，通常在数字后面加字母B(Binary)表示二进制数，如1011B，数字后面为Q表示八进制数，如47Q；数字后面不加字母或加字母

表 1-1　几种常用数制的对照

十进制数	二进制数	八进制数	十六进制数
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10
17	10001	21	11
18	10010	22	12
19	10011	23	13
20	10100	24	14

D(Decimal)表示十进制数，如 57；数字后面加字母 H(Hexdecimal)表示十六进制数，如 3AH。

(二)各数制之间的转换

1．二进制数转换成十进制数

根据二进制数的定义，将欲转换的二进制数按权展开并相加即得相应的十进制数。例如：

$$\begin{aligned}110.011\text{B}&=1\times2^2+1\times2^1+0\times2^0+0\times2^{-1}+1\times2^{-2}+1\times2^{-3}\\&=6.375\end{aligned}$$

2．十进制数转换成二进制数

(1) 十进制整数转换成二进制整数

十进制整数 M 用二进制整数表示时可写成：

$$\begin{aligned}M_{10}&=(a_{n-1}a_{n-2}\cdots a_1a_0)_2\\&=a_{n-1}\cdot2^{n-1}+a_{n-2}\cdot2^{n-2}+\cdots+a_1\cdot2^1+a_0\cdot2^0\end{aligned}$$

只要找到系数 a_{n-1}, a_{n-2}, …, a_1, a_0 的值，就得到相应的二进制数，这些系数不是 1 就是 0。由上面多项式可知，等号右边除 a_0 项外，其余各项都能被 2 整除，故用 2 去除十进制数 M，得到的余数必为 a_0；同理，用 2 去除 $\frac{M}{2}$，得到的余数即为 a_1，用这样的方法一直做下去，直到商为 0 为止，就可得到 a_{n-1}, a_{n-2}, …, a_1, a_0 的值。

例如，把十进制整数 67 转换成二进制数；

$$
\begin{array}{r|l}
2 & 67 \\
\hline
2 & 33 \quad \cdots\cdots 余\ 1 = a_0 \uparrow \\
\hline
2 & 16 \quad \cdots\cdots 余\ 1 = a_1 \\
\hline
2 & 8 \quad \cdots\cdots 余\ 0 = a_2 \\
\hline
2 & 4 \quad \cdots\cdots 余\ 0 = a_3 \\
\hline
2 & 2 \quad \cdots\cdots 余\ 0 = a_4 \\
\hline
2 & 1 \quad \cdots\cdots 余\ 0 = a_5 \\
\hline
 & 0 \quad \cdots\cdots 余\ 1 = a_6
\end{array}
$$

$\therefore\ (67)_{10} = (1000011)_2$

为了便于记忆，我们把十进制整数转换成二进制整数的方法归纳为一句话："除 2 倒取余"。

(2) 十进制小数转换成二进制小数

若把十进制小数 0.8125 展开成二进制小数，其形式为：

$$(0.8125)_{10} = (0.a_{-1}a_{-2}\cdots a_{-m})_2 = a_{-1}\cdot 2^{-1} + a_{-2}\cdot 2^{-2} + \cdots + a_{-m}\cdot 2^{-m})_2$$

从此多项式可以看出，若等号两边同时乘以 2，则可得到整数 a_{-1} 的值。即

$$1.6250 = a_{-1} + a_{-2}\cdot 2^{-1} + a_{-3}\cdot 2^{-2} + \cdots + a_{-m}\cdot 2^{-m+1}$$

所以，$a_{-1}=1$。等号两边再同时乘以 2，可得到 a_{-2}，这样一直继续下去，直到小数部分为 0，或者达到精度要求为止。

例如，把十进制小数 0.8125 转换成二进制数：

$$
\begin{array}{l}
\quad 0.8125 \\
\times \quad\ 2 \\
\hline
\quad 1.6250 \quad \cdots\cdots 整数\ 1 = a_{-1} \\
\quad 0.625 \\
\times \quad\ 2 \\
\hline
\quad 1.250 \quad \cdots\cdots 整数\ 1 = a_{-2} \\
\quad 0.25 \\
\times \quad\ 2 \\
\hline
\quad 0.50 \quad \cdots\cdots 整数\ 0 = a_{-3} \\
\times \quad\ 2 \\
\hline
\quad 1.0 \quad \cdots\cdots 整数\ 1 = a_{-4} \downarrow
\end{array}
$$

$\therefore\ (0.8125)_{10} = (0.1101)_2$

把上面十进制小数转换成二进制小数的方法归纳成一句话："乘 2 顺取整"。

如果是整数与小数混合的十进制数，则根据上述方法分别求之。

3．二进制数与八进制数相互转换

二进制数转换成八进制数的方法是：以二进制数的小数点为中心，向两边分成三位一组，若最后不足三位，则以无效零补足。

例如，二进制数：11101.11

分成三位一组：$\boxed{0}11\ 101.\ 11\boxed{0}$

八进制数：3 5 6

$\therefore\ (11101.11)_2=(35.6)_8$

显然，八进制数要还原成二进制数，只须把每一位八进制数以三位二进制数表示即可。

例如，

八进制数：3 5 . 6

每位以三位二进制数表示：011 101. 110

二进制数：11101.11

$\therefore\ (35.6)_8=(11101.11)_2$

4．二进制数与十六进制数的转换

二进制数转换成十六进制数的方法为：以二进制数的小数点为中心，向两边分成四位一组，不足四位，以无效零补足。

例如，

二进制数：111110.10101

分成四位一组：0011 1110.1010 1000

十六进制数：3 E . A 8

$\therefore\ (111110.10101)_2=(3E.A8)_{16}$

十六进制数转换成二进制数的方法为：把一位十六进制数以四位二进制数表示即可。

例如，

十六进制数：3 E . A 8

每位以四位二进制数表示：0011 1110.1010 1000

二进制数：111110.10101

$\therefore\ (3E.A8)_{16}=(111110.10101)_2$

二、码制——计算机中正、负数的表示方法

我们在讲述进位计数制时，已提及计算机采用的二进制数的表示法，但是表示一个数，除了数的值外，还必须确定该数的正、负，也就是说，数总是由数符(数的符号)及尾数两部分组成，如

+1011101

数符——↑　↑——尾数

数的正、负号在计算机中也以二进制数字表示，以“0”表示正号，以“1”表示负号。通常，把符号位放在最高位。

以“0”或“1”为数符的二进制数称为机器数，其数值称为机器数的真值。

例如，

机器数	真值
$X_1=01010100$	$X_1=+1010100$
$X_2=11010100$	$X_2=-1010100$

在计算机中，表示一个带符号的二进制数常有以下三种表示方法，即原码表示法、反码表示法和补码表示法。由于反码在运算中对计算机的结构有特殊要求，不常采用，在这里引出反码仅是为了能方便地求得补码。所以，这里只着重介绍原码和补码。

(一)原码

1．原码表示法

用最高位(符号位)表示数的符号，而用其余各位表示数的尾数，叫做原码表示法。

原码和真值之间具有以下关系：

1° 正数的原码表示

$$X=+1010100,\ [X]_{原}=01010100$$

2° 负数的原码表示

$$X=-1010100,\ [X]_{原}=11010100$$

3° 零的原码表示

正零的原码表示：

$$X=+0000000,\ [X]_{原}=00000000$$

负零的原码表示：

$$X=-0000000,\ [X]_{原}=10000000$$

综上所述，

$$[X]_{原}=\begin{cases}X & \text{当 } X>0 \text{ 或 } X=+0 \text{ 时}\\ 2^{n-1}-X & \text{当 } X<0 \text{ 或 } X=-0 \text{ 时}(n \text{ 为字长})\end{cases}$$

2．*原码运算*

(1) 加、减法运算　两原码数进行加、减法运算时，首先判别两数的符号是否相同。如果两数符号相同，则两数的尾数进行加法运算，其结果的符号就是参加运算的数的符号。如果参加运算的两数为异号，则进行减法运算。绝对值大的数减去绝对值小的数，就是结果的尾数，结果的数符和两数中绝对值大的那个数相同。

(2) 乘、除法运算　乘、除法用原码运算比较方便。计算机做原码乘、除法运算和十进制数乘、除法一样，对数符的运算和对尾数的求积(或商)运算是分别进行的。对尾数的求积(或商)通常是分解为一系列加(或减)法及移位操作来实现的，而对积(或商)的符号是按一定规则得到的，见表 1-2。

表 1-2

被乘数(被除数)符号	乘数(除数)符号	积(商)符号
0	0	0
1	1	0
0	1	1
1	0	1

这种数符之间的关系，在数字逻辑中称半加(或称异或)的逻辑关系，很易实现。最后把分别求得的积或商的尾数及其符号拼在一起，得到一个完整的结果。

(二)反码表示法

用反码表示一个数时，正数的反码等于其原码；负数的反码是把该数的尾数各位取反，然后在符号位上放 1。例如：

$$X=+1001010,\quad [X]_{反}=01001010$$
$$X=-1001010,\quad [X]_{反}=10110101$$
$$X=+0000000,\quad [X]_{反}=00000000$$
$$X=-0000000,\quad [X]_{反}=11111111$$

(三)补码表示法

原码表示法，对乘、除法运算是较简单的，只要对尾数和数符分别运算即可，但是对加减

法运算就较为复杂。譬如，两数相加，首先要判两数是否同号，若同号，则进行加法，否则就做减法；如果是做减法，还得比较两数绝对值的大小，且记下绝对值大的数符，然后大数减去小数，差值的符号与绝对值大的数的符号一致。不难看出，为了实现原码加减法，硬件结构相应地就比较复杂，所以，在微型计算机中，为了简化硬件结构通常是用补码实现加减法运算。

采用补码后，减法可以由补码的加法实现。

设有一台时钟，指针指在 10 时，而标准时间是 6 时，要拨准时钟有两种办法：

其一是倒拨四时，即 $10-4=6$

其二是顺拨八时，即 $10+8=\underset{\text{自动丢失}}{12}+6=6$

两式得到相同的结果，说明$(10-4)$可以用$(10+8)$代替。这个自动丢失的数称为模(mod)。模和某数 X(如 -4)相加所得的数，如$[12+(-4)=8]$称为该数的补码。即

$$[X]_{补}=模+X$$

所以，$$10-4=10+[-4]_{补}=10+(12+(-4))=10+8=6 \pmod{12}$$

由此可见，有模的数，其减法可以化做加法。计算机内的数是有模的数，其小数的模为 2，整数的模为 2^n(n 为字长)。

在定点机中，如果机器的字长为 n 位，那么它的模就是 2^n。这是一个 $n+1$ 位的数 $\overbrace{10000\cdots0}^{n\text{位}}$，由于机器只能表示 n 位数，因此数 2^n 在机器中仅能以 n 个 0 来表示，而该数最左边的数字 1 就自动舍弃了。由此可见，2^n 和 0 在机器中的表示形式是完全一致的。

1．补码表示法

1° 正数的补码表示

$$X=+1010100,\ [X]_{补}=模+X=2^8+1010100=01010100=X \pmod{2^8}$$

$\therefore\ [X]_{补}=01010100$

由此可见，正数的补码等于原码。

2° 负数的补码表示

$$X=-1010100,\ [X]_{补}=模+X=2^8-1010100=10101100 \pmod{2^8}$$

$\therefore\ [X]_{补}=10101100$

在计算机中，引出二进制数的补码表示，是为了化减法为加法，从而使运算大大简化。但是，为了求出负数的补码还是用了减法，在机器中实现不太方便。一种简便使用的方法是：对于负数的补码表示，只需求出该负数的反码，然后在末位加一。这在机器的硬件中容易实现。下面仍以 -1010100 为例：

X：	-1010100
$[X]_{反}$：	10101011
末位加 1：	1
$[X]_{补}$：	10101100

3° 零的补码表示

正零的补码表示：

$$X=+0000000,\ [X]_{补}=2^8+0000000=00000000$$

负零的补码表示

$$X=-0000000,\ [X]_{补}=2^8-0000000=00000000$$

所以补码中零只有一种表示，无正负零之分。

综上所述，
$$[X]_{补}=\begin{cases}X & \text{当 } X\geqslant 0 \text{ 时}\\ 2^n+X & \text{当 } X<0 \text{ 时}(n \text{ 为字长})\end{cases}$$

2．补码运算

补码运算有如下一个重要公式：

$$[X+Y]_{补}=[X]_{补}+[Y]_{补}\quad(\mathrm{mod}\,2^n)$$

式中 $-2^{n-1}\leqslant X<2^{n-1}$，$-2^{n-1}\leqslant Y<2^{n-1}$，且 $-2^{n-1}\leqslant X+Y<2^{n-1}$

这个公式表明，采用补码形式表示数时，两数相加运算中不必考虑数的符号，而只要将符号位当作数值部分同样参与运算，其结果即为"和"的真值的补码。

分四种情况证明：

1° 当 $2^{n-1}>X\geqslant 0$，$2^{n-1}>Y\geqslant 0$，且 $2^{n-1}>(X+Y)\geqslant 0$ 时，由补码定义可知：

$$[X]_{补}=X,\ [Y]_{补}=Y$$

则有
$$[X+Y]_{补}=X+Y=[X]_{补}+[Y]_{补}\quad(\mathrm{mod}\,2^n)$$

2° 当 $2^{n-1}>X\geqslant 0$，$0>Y\geqslant -2^{n-1}$，且 $2^{n-1}>X+Y\geqslant 0$ 时，由补码定义可知：

$$[X]_{补}=X$$
$$[Y]_{补}=2^n+Y$$
$$[X+Y]_{补}=X+Y$$

所以，
$$[X]_{补}+[Y]_{补}=X+2^n+Y=2^n+(X+Y)=X+Y=[X+Y]_{补}\quad(\mathrm{mod}\,2^n)$$

3° 当 $2^{n-1}>X\geqslant 0$，$0>Y\geqslant -2^{n-1}$，且 $0\geqslant(X+Y)\geqslant -2^{n-1}$ 时，由补码定义可知：

$$[X]_{补}=X$$
$$[Y]_{补}=2^n+Y$$
$$[X+Y]_{补}=2^n+(X+Y)$$

所以，
$$[X]_{补}+[Y]_{补}=X+2^n+Y=2^n+(X+Y)=[X+Y]_{补}\quad(\mathrm{mod}\,2^n)$$

4° 当 $0>X\geqslant -2^{n-1}$，$0>Y\geqslant -2^{n-1}$，且 $0>(X+Y)\geqslant -2^{n-1}$ 时，由补码定义可知：

$$[X]_{补}=2^n+X$$
$$[Y]_{补}=2^n+Y$$
$$[X+Y]_{补}=2^n+(X+Y)$$

所以，
$$[X]_{补}+[Y]_{补}=2^n+X+2^n+Y=2^n+(X+Y)=[X+Y]_{补}\quad(\mathrm{mod}\,2^n)$$

总之，引进补码概念后，加减法运算中，减法可以用加法来实现，即用求"和"代替求"差"，数的符号位可以与数值部分一样参与运算，并且补码的"和"等于"和"的补码，运算简单方便，因此，补码被广泛用于微型计算机的算术运算中。

两个补码表示的数相加，其结果可能会超过数的表示范围，在计算机中称溢出。如何判断有否溢出呢？可以按下述规则判断：

1° 将两个补码表示的数相加，观察符号位是否有进位输入（从低位进位到符号位）或进位输出（从符号位产生一进位输出）。

2° 如没有进位发生，或同时产生进位输入和进位输出，其结果不会溢出，则结果正确；如果只产生一次进位（只有进位输入或只有进位输出），其结果必然溢出，则得到不正确结