中等专业学校教材

微型计算机原理及应用

上海市化工学校　余龙山　主编

化学工业出版社

中等专业学校教材

微型计算机原理及应用

上海市化工学校　　余龙山 主编

化学工业出版社
·北　京·

前　言

本教材是根据化工中专电类课程教材编审委员会1987年4月制定的《微机原理及应用》教学大纲编写的，本书可作为中等专业学校仪电类专业的试用教材，也可作为广大科技人员的自学课本和参考用书。

本教材力求具有中专特色，在总结有关教材经验和教学实践的基础上，结合培养应用性、工艺性中等专业人材的需要，教材编写贯彻“分散难点、循序渐进、软硬件结合、面向应用”的原则，把微型计算机的基本原理和基本应用作为总线贯穿于整个教材之中。教材以Z80微处理器为编写背景，内容包括：计算机的数制和编码，微型计算机的基本组成和工作原理，Z80指令系统、汇编语言程序设计，半导体存储器，输入输出和中断技术，接口电路，A/D、D/A转换器，监控程序和应用举例等。为了适应当前单片机应用的日益普及，教材编入了有关单片机的内容，根据大纲精神，监控程序和单片机章节可作为学生的选读内容。

本教材由上海化工学校余龙山主编，王建萍参编。余龙山编写绪论，第一、二、五、六、七、九、十、十一章并负责定稿。王建萍编写三、四、八章。安徽化工学校张新广担任本教材的主审。

参加本教材审稿的有：辽宁石油化工学校张仁仲、高广仁，兰州化工学校贺平，武汉化工学校张亚武、李少华，湖南化工学校徐润德，吉林化工学校柳东根，上海化工学校童迺涛、赵家庆。主审和参审对全书进行了全面而详细的审阅，对书稿中存在的缺点和不妥之处提出了宝贵的意见和建议。泸州化工学校张蜀刚同志对编写提纲和书稿提出了详尽的书面意见。在此一并表示衷心的感谢和敬意。

由于编者水平有限，书中难免存在不少缺点和错误，恳请使用本教材的师生和读者批评指正。

编者

1991年10月

目　录

诸　论

自1946年世界上第一台电子计算机ENIAC问世以来，计算机科学一直以日新月异的势头向前发展。纵观半个世纪电子计算机的发展史，大致经历了电子管计算机、晶体管计算机、集成电路计算机和大规模集成电路计算机的四代演变，并开始研制具有知识处理、推理解题、智能接口为特点的第五代计算机。

电子计算机的每一代演变与电子元器件的更新换代直接相关。七十年代以来，由于微电子技术的迅猛发展，特别是大规模集成电路和超大规模集成电路芯片的出现，导致了以微处理器为中心的微型计算机的迅速发展。1971年微处理器微计算机的问世标志着第四代计算机的诞生。以后每隔2～4年微型机就要更新一代。若以微处理器的字长位和功能划分，至今已发展了四代产品。

第一代为1971～1973年的4位微处理器和低档8位微处理器。如美国Intel公司的4004、Intel 8008。软件主要采用机器语言及简单的汇编语言。

第二代为1974～1977年的8位微处理器和微型计算机。其间有1974～1975年的Intel 8080、MotoRola的MC6800和1976～1977年的Intel8085及Zilog公司的Z80，软件除汇编语言外，还配有BASIC、FORTRAN等多种语言。它们主要用于事务管理，工业控制和智能仪表。

第三代为1978～1980年的16位处理器和微计算机，如Intel8086，Z8000，MC68000。从各种性能指标评价，它们已经达到和超过中低档小型机（如PDP11/45)的水平。

第四代为1981年以后的32位微处理器和微计算机。如Intel公司的IAPX432，HP公司及贝尔研究所的MP32等。现在32位微型机的功能已足以同高档小型机匹敌，大有取代中、小型机之势。

当前微处理器与微计算机的发展趋势主要有以下几个方面：

（1）发展高性能的16位和32位微处理器，使微处理的功能能与高档小型机匹敌。

（2）发展专用和通用的单片微计算机，它把CPU，RAM，ROM/EPROM,I/O接口，定时器/计数器集成在一片芯片上，这使得微计算机的体积大大缩小，使得它特别适用于仪器仪表的智能化。

（3）发展多处理机系统和局部网络。多处理机系统具有高运算能力，高可靠性和高利用率。应用局部网络可以实现微机的互相通信和资源共享。

（4）发展和充实微型计算机的外围接口电路。

微型计算机不仅具有普通计算机所具有的特点。如运算速度快、精度高、自动化，具有记忆和逻辑判别能力，同时还具有价格低、体积小、重量轻、功耗低、可靠性高以及方便灵活、通用性强等特点。由于微型机的这些特点，使它深入到过去计算机无法深入的应用领域。主要体现如下：

（1）微型机促进和加速了仪器仪表产品的更新换代。由于许多仪器仪表装入了微型机（特别是单片机)，使仪器仪表的产品结构发生了根本的变革，使一大批产品朝着数字化、智能化多功能方向发展。

（2）微型机使生产过程的自动控制由过去的集中控制发展成为多台微机的集中分散控制，将生产过程分段由微机进行监视和控制，这些微机可以是单板机、单片机及微机系统。而各微机之间又用小型机或高档微机来集中指挥。它比集中系统具有更好的灵活性和可靠性。

随着16位和32位高档微机的研制成功和投入使用，微型机的应用已广泛地渗透到工农业、国防科研、文教卫生、家用电气等各个行业。应用领域已由传统的科学计算、数据处理、实时控制扩大到计算机辅助制造（CAM）、计算机辅助设计（CAD）、计算机辅助教学（CAI）、人工智能、办公室自动化等。并且日益显示出它强大的生命力。

本书从应用的角度出发，并结合Z80微处理器介绍微计算机的基本原理及应用。在叙述方法上采用分散疑难概念、贯彻循序渐进、注重软硬结合、强调系统意识。课程重点是讲述微型机的基本组成和工作原理、汇编语言程序设计、输入输出及中断系统、接口电路及其应用，并对单片机的组成和原理作简单介绍。通过本教材的学习，使读者掌握微型机的基本工作原理和基本应用方法，为今后从事计算机的应用和深入学习其它类型的微处理器和微计算机（如高档16位/32位微机、单片机等）打下扎实的基础。

第一章　计算机运算基础

各种类型计算机的处理对象都是数。因此，在学习微型计算机原理之前，首先对计算机中数的表示方法及运算规则作一简要介绍。

第一节　计算机中的数制

一、进位制数

按进位的方法进行计数，称进位制数。

在日常生活中，人们最熟悉最常用的是十进制数，还有十二进制数、六十进制数等。在微型计算机中常用的数有二进制数，十六进制数等。

我们把进位制数中所使用的数码个数称为“基数”；进位制数每一位所具有的值称为“权”。某位的数码和该位的“权”的乘积就是该位数值的大小。“权”和“基数”反映了各种进位制数的特点。

下面就计算机中常用的数制作简要介绍。

（一）十进制数

十进制数的基数为“10”，它有十个不同的数码；0、1、2、3、4、……9，且逢10进位。它各位的权是以10为底的幂。

例如：十进制数：$\underset{\downarrow}{7}\quad\underset{\downarrow}{9}\quad\underset{\downarrow}{2}\cdot\underset{\downarrow}{5}\quad\underset{\downarrow}{4}$

各位的权：$10^2\ \ 10^1\ \ 10^0\cdot 10^{-1}\ \ 10^{-2}$

这个数可表示为：

$$792.54=7\times10^2+9\times10^1+2\times10^0+5\times10^{-1}+4\times10^{-2}$$

一般地说，任意一个十进数N都可表示为：

$$\begin{aligned}N=\pm(&K_{n-1}\times10^{n-1}+K_{n-2}\times10^{n-2}+\cdots\cdots+K_1\times10^1\\&+K_0\times10^0+K_{-1}\times10^{-1}+\cdots\cdots+K_{-m}\times10^{-m})\\=&\sum_{i=-m}^{n-1}(K_i\times10^i)\end{aligned}\qquad(1)$$

其中K_i表示第i位数码，可以是0～9十个数字符号中的任意一个，由具体的N来确定。m、n为正整数，n为小数点左边的数位；m为小数点右边的数位，10^{n-1}、10^{n-2}、10^0、10^{-1}、……10^{-m}为十进制数各位的“权”。而10为十进制数的基数，表示逢十进一。

对于任意进位制数，基数可用正整数R来表示。这时，数N可表示为：

$$N=\pm\sum_{i=-m}^{n-1}K_iR^i\qquad(2)$$

式中m、n均为正整数；K_i则是0、1……$(R-1)$中的任一个，R是基数，采用“逢R进一”的原则进行计数，各位的权为R^i。

（二）二进制数

二进制数的基数为“2”，它具有二个不同的数码；0和1，且逢2进位。二进制数各位的权是以2为底的幂。

例如　二进制数：$1\ 1\ 1\ 0\ 1 \cdot 1\ 1$

各位的权　$2^4\ 2^3\ 2^2\ 2^1\ 2^0 \cdot 2^{-1}\ 2^{-2}$

根据式2，这个数可写成：

$$(11101.11)_2=1\times2^4+1\times2^3+1\times2^2+0\times2^1+1\times2^0+1\times2^{-1}+1\times2^{-2}$$

$$=16+8+4+0+1+0.5+0.25$$

$$=(29.75)_{10}$$

它是逢2进位的。如

$$(10)_2+(11)_2=(101)_2$$

$$(101)_2-(11)_2=(10)_2$$

（三）十六进制数

十六进制数的基数为“16”，它具有十六个不同的数码；0、1、2……9、A、B、C、D、E、F。其中A、B、C、D、E、F分别代表10、11、12、13、14、15，且逢16进位。16进制数各位的权是以16为底的幂。

根据式(2)，16进制数A5.8可展开成下式：

$$(A5.8)=A\times16^1+5\times16^0+8\times16^{-1}=160+5+0.5=(165.5)_{10}$$

它逢16进位，如16进制数39＋98＝D1，100－39＝C7。

以上三种数制的表示方法如表1-1。

表 1-1　10、2、16进制数对照表

十进制	二进制数	十六进制	十进制	二进制数	十六进制
0	0000	0	9	1001	9
1	0001	1	10	1010	A
2	0010	2	11	1011	B
3	0011	3	12	1100	C
4	0100	4	13	1101	D
5	0101	5	14	1110	E
6	0110	6	15	1111	F
7	0111	7	16	10000	10
8	1000	8			

为了区别所表示数的数制，一般有二种方法：一是在数的括号右下角用数字注明数制。如二进制数1101.1可写成$(1101.1)_2$、十六进制数A5.7可写成$(A5.7)_{16}$

另一种表示法可以在数字后面跟一个英文字母。通常用B(Binary)表示二进制数，H(Hexad ecimal)表示十六进制数。所以上面二个数也可写成：1101.1B，A5.7H。十进制数一般可省略不写。

（四）二进制数的特点

（1）状态简单，只要具有两个状态的元件都可用来表示为二进制数的0和1，所以物质基础广泛，容易实现。

（2）运算规则简单，使计算机中的运算线路大为简化。

（3）可用布尔代数这一数学工具对计算机电路进行设计和分析，便于对计算机结构的

优化。

由于它具有上述特点，使得二进制数特别适用于计算机。但由于二进制数书写冗长，阅读不便，为此人们常用十六进制数来书写。此外人们习惯于十进制数，这就要解决不同数制之间的互相转换问题。

二、各种数制的转换

（一）二进制数与十进制数的转换

1．二进制数转换为十进制数

转换方法是按“权”展开然后相加。

例如 $(11001.101)_2=1\times2^4+1\times2^3+0\times2^2+0\times2^1+1\times2^0+1\times2^{-1}+0\times2^{-2}+1\times2^{-3}$
$=(25.625)_2$。

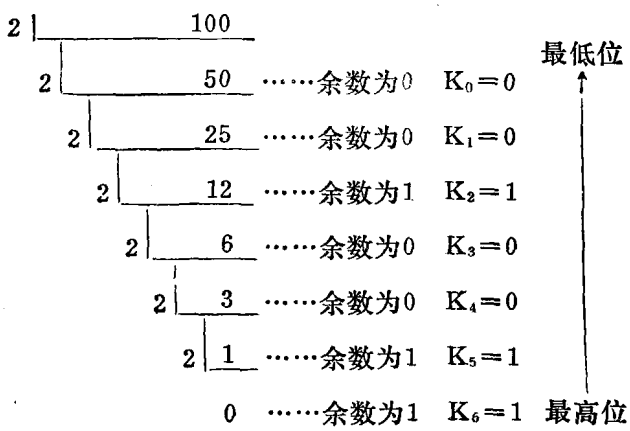
2．十进制数转换为二进制数

十进制数整数部分和小数部分的转换方法是不同的。

（1）整数部分的转换

整数部分的转换采用“除2取余法”，即用2不断地去除要转换的十进制整数，直至商为0。将所得的各次余数，以最后余数为最高位，依次排列，即得到所转换的二进制数。

例如 十进制数100转换为二进制数的过程如下：

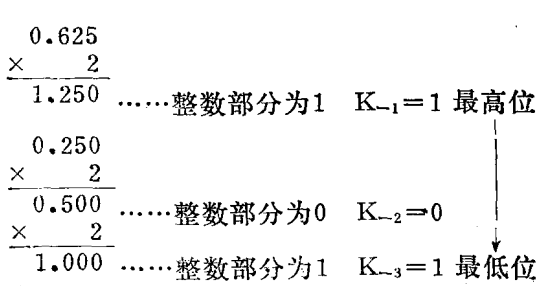


所以$(100)_{10}=(1100100)_2$。

（2）小数部分的转换

小数部分的转换采用“乘2取整法”，即用2不断地去乘所要转换的十进制小数部分，直到小数部分等于零或满足所要求的精度为止，把每次乘积的整数部分，以最初整数为最高位，依次排列，即得到所转换的二进制数。

例如 求0.625的二进制数



所以$(0.625)_{10}=(0.101)_2$

十进制小数在转换时，整个转换过程有时会无限地进行下去，这时可根据精度要求，选取适当的位数。

对于既有整数部分又有小数部分的十进制数，只要按上述方法分别转换，然后合并起来即可。

例如：十进制数75.75转换结果为：

$$(75.75)_{10}=(1001011.11)_2$$

（二）十六进制数与十进制数的转换

转换方法同前面二进制数与十进制数之间转换类似。区别仅在于把基数 2 换成16即可，这里不再赘述。

（三）二进制数与十六进制数之间的转换

由于数16与数 2 之间的关系为$16=2^4$。因此一位十六进制数对应四位二进制数，根据这个关系，对于二进制整数的转换，只要从小数点开始向左依次每四位二进制数用一位十六进制数表示。若最左边不足四位应在左边添 0 补足四位；对于二进制小数的转换，只要从小数点开始，向右依次每四位二进制数用一位十六进制数表示，若最右边不足四位的，应在右边添 0 补足四位。

例如：二进制数1110111010101.110010111可以用上述方法转换为十六进制数。

(000)1	1101	1101	0101	·	1100	1011	1(000)
↓	↓	↓	↓		↓	↓	↓
1	D	D	5	·	C	B	8

十六进制数转换为二进制数的方法只需四位二进制数代换一位十六进制数即可。

例如：十六进制数2BA · 3C

2	B	A	·	3	C
↓	↓	↓		↓	↓
0010	1011	1010	·	0011	1100

即$(2BA\cdot3C)_{16}=(001010111010.00111100)_2$

目前，在微型机中普遍采用十六进制数，其原因是十六进制与二进制数之间的转换方便，同时书写较短，便于阅读。

三、二进制数的运算规则

（一）算术运算规则

二进制数只有0、1两个数码，它的运算规则比十进制数的运算规则简单得多。

1．加法规则

（1）0＋0＝0

（2）1＋0＝1

（3）0＋1＝1

（4）1＋1＝10

2．减法规则

（1）0－0＝0

（2）1－0＝1

（3）1－1＝0

（4）0－1＝1向相邻高位借1当作2。

3．乘法规则

（1）0×0＝0

（2）0×1＝0

（3）1×0＝0

（4）1×1＝1

除了1×1外，其余乘积均为0，显然它比十进制数的“九九”乘法规则简单得多。

4．除法规则

它是乘法的逆运算，规则与十进制除法类似，这里不再赘述。

（二）逻辑运算规则

这里介绍四种逻辑运算。

1．或运算（逻辑加），运算符为“+”或“∨”

（1）0∨0＝0

（2）1∨0＝1

（3）0∨1＝1

（4）1∨1＝1

2．与运算（逻辑乘），运算符为“·”或“∧”

（1）0∧0＝0

（2）0∧1＝0

（3）1∧0＝0

（4）1∧1＝1

3．非运算（逻辑否定）

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

4．异或运算（按位加），运算符为“⊕”

（1）0⊕0＝0

（2）0⊕1＝1

（3）1⊕0＝1

（4）1⊕1＝0

由此可知，按位加是不考虑进位的加法运算。

第二节　计算机中带符号数的表示法及其运算

对于一个不带正负符号的数，称无符号数。它将字长的所有位均用于表示数值的大小。一个n位字长的数据可用来表示2^n个正整数。例如，一个8位数据可表示的数值范围为：

00000000B——11111111B

即：0——255共256个数值

通常，数还有正、负之别，并用符号“+”、“－”来表示，称为带符号数。在电子计算机中，带符号数又如何表示呢？这将是本节要讨论的问题。

一、机器数与真数

在计算机中，数的正负符号与数一起存放在寄存器中，因此数的符号在计算机中也“数

字化”了。通常规定在数的前面增设一位符号位，并规定正号用“0”表示，负号用“1”表示。

若以8位字长的存数单元为例，设有数

$N_1 = +1001010$

$N_2 = -1001010$

则N_1和N_2在计算机中表示形式为：

N_1'：01001010；十进制数为+74。

N_2'：11001010；十进制数为-74，而不是193。

在计算机中，把正、负号已数字化了的数称为机器数，而把原来带正负号的数称为该机器数的真数。上面例子01001010和11001010为机器数，而+1001010和-1001010是对应机器数的真数。

在计算机中，机器数有三种表示形式，即原码、补码、反码。

二、原码、反码和补码

在介绍原码、反码和补码之前，先介绍模的概念和性质。我们把一个数字系统的量程或一个计量器的容量称为“模”或“模数”，记为M或modM。在计量过程中它会自动丢失。

如钟表系统的模为12，当时钟转到12时，时钟又从零开始重新计时，数12就自动丢失。又如三位十进制计数器，它的表示范围从000到999，若在999时再加1，则为1000，此时1000这个数就自动丢失，在计数器中又从000开始重新计数，所以1000是此计数器的模。又如对四位二进制计数器的模为2^4即10000，当1111再加1时，计数器的值为0000，2^4就自动丢失。即2^4和0000在四位二进制计数器中表示是一样的。同理，对于n位二进制计数器，它的模为2^n。2^n和0在n位二进制计数器中表示的形式是一样的。

（一）原码

当符号位为“0”表示正数，符号位为“1”表示负数，其余各位表示尾数，这种表示法称原码表示法。前面介绍的机器数N_1'和N_2'就是原码表示法。

1．对于正数的原码表示

例 1　X＝+1001011。

$$[X]_{原}=\underbrace{0}_{\text{符号位}}\underbrace{1001011}_{\text{尾数}}$$

2．对于负数的原码表示

例 2　X＝-1001011

$$[X]_{原}=\underbrace{1}_{\text{符号位}}\underbrace{1001011}_{\text{尾数}}$$

3．对于0的原码表示

“0”在原码中的表示法有二种

$$[+0]_{原}=000\cdots\cdots0$$

$$[-0]_{原}=100\cdots\cdots0$$

计算机遇到这两种情况都当作0来处理。

对于8位二进制原码的数值范围为+127～-127，即：

$$[+127]_{原}=01111111$$

$$[-127]_{原}=11111111$$

由上述原码的表示形式，可将字长为n位二进制整数（包括符号位）X的原码定义为：

$$[X]_{原}\begin{cases}2^n+X & 当\quad 0\leqslant X<2^{n-1}\\ 2^{n-1}-X & 当\quad -2^{n-1}<X\leqslant 0\end{cases}\quad(\mathrm{mod}2^n)$$

（二）反码

1．正数的反码与正数的原码相同

2．负数的反码除符号位仍为1外，其余各位均取反

例 3　X＝－1001011

$[X]_{反}=10110100$

3．“0”的反码表示不是唯一的，它有$[+0]_{反}$和$[-0]_{反}$二种表示法

$$[+0]_{反}=000\cdots\cdots0$$

$$[-0]_{反}=111\cdots\cdots1$$

由上所述，可将字长为n位二进制整数（包括符号位）X的反码定义为：

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

（三）补码

原码表示简单易懂，与真数的转换方便。原码在执行乘除运算时，只要将尾数进行乘除操作，而商或积的符号为两个原码符号的异或结果，所以显得十分方便。但遇到两个异号数相加时，计算机的运算就变得复杂，这时计算机首先要比较两数绝对值的大小，然后进行减法运算，将绝对值大者减绝对值小者，最后决定差值的符号。计算机做这些工作势必增加硬件电路的复杂程度，而且减低了加减运算的速度。为了简化硬件设备，提高运算速度，人们采用了另一种机器数表示法，即补码表示法。

补码表示法可以把负数转化为正数，使减法转换为加法，从而使正负数的加减运算转化为单纯的正数相加运算，这样就解决了原码表示法在加减运算中所遇到的上述问题。

1．同余的概念和补码

如果有两个整数a和b，被某一正整数M相除，所得余数相等时，则称a和b对模M是同余的。

例如，a＝16，b＝26，若模为10；则a、b除以10，余数均为6，我们称16和26在以10为模时是同余的，并记为：

$$16=26(\mathrm{mod}10)$$

即16和26在以10为模时，它们是相等的。由同余的概念不难得出：

$$a+M=a(\mathrm{mod}\ M)$$

$$a+2M=a(\mathrm{mod}\ M)$$

因此当a为负数时，如a＝－7，在以10为模时有：

$$-7+10=-7(\mathrm{mod}\ 10)$$

$$3=-7(\mathrm{mod}\ 10)$$

即在以10为模时，－7与＋3是相等的。我们称＋3是－7的补码，或者说＋3与－7对模来说互为补数。同理，＋6是－4的补码，＋8是－2的补码等等。由此可以得出补码的定义为：

$$[X]_{补}=模+X$$

由于引进了补码，我们就可以将减法转化为加法（加补码），例如：

$$7-4=7+6(\mathrm{mod}10)$$

即在以10为模时，7减4可以通过7加－4的补码6来运算，所得的结果是一样的。只是在加补码时所产生的进位“10”舍弃即可，这也正是以10为模的含意。

由补码定义可知，为了求得补码仍然用了减法运算，似乎还没避免减法运算。然而在计算机中求二进制数的补码可以用其它简单的方法而不用减法。

2．模为2的补码

假定字长为n（包括符号位）位的带符号二进制整数，它的模就是2^n，也称以2为模。根据补码的定义得到整数补码的表示式为：

$$[X]_{补}=2^n+X\ (\mathrm{mod}2^n)$$

由定义可知

（1）正数的补码与正数的原码表示相同

例 4　$X=+1001011$

则$[X]_{补}=2^8+01001011=01001011$(模自动丢失)。

（2）对于负数的补码表示

例 5　$X=-1001011$

$$\text{则}[X]_{补}=2^8-1001011$$
$$=10110101$$

（3）对于“0”的补码表示

$$[+0]_{补}=000\cdots\cdots0$$
$$[-0]_{补}=2^n-0$$
$$=000\cdots\cdots0$$

因此，对于补码而言$[+0]_{补}=[-0]_{补}$即0的补码只有一种表示方式。

（4）当$X=-2^{n-1}$时，仍可用$[X]_{补}=2^n+X$求得；

$$[-2^{n-1}]_{补}=2^n-2^{n-1}=2^{n-1}$$

这样，补码的取值范围扩大为：$-2^{n-1}\leqslant X<2^{n-1}$。

若字长为8位的计算机中，补码所能表示的范围为：

$+127\sim-128$即：

$$01111111\sim10000000$$

比较$[X]_{原}$、$[X]_{反}$、$[X]_{补}$三个定义表达式可知；当$X>0$时，$[X]_{原}=[X]_{反}=[X]_{补}$。当$X<0$时，$[X]_{补}=2^n+X=\underbrace{(2^n-1)+X}_{[X]_{反}}+1=[X]_{反}+1$。

由此可知，对于满足$-2^{n-1}\leqslant X<0$的负数，$[X]_{补}$为$[X]_{原}$除符号位外各位求反加1，即X的反码加1。这样，在计算机中求补码时就可避免减法运算。

例 6

$[X]_{原}=1\ 0\ 0\ 0\ 1\ 0\ 1\ 0$

↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓

$1\ 1\ 1\ 1\ 0\ 1\ 0\ 1$　除符号位外各位变反最低位

$+\qquad\qquad\qquad\ 1$　加1

$[X]_{补}=1\ 1\ 1\ 1\ 0\ 1\ 1\ 0$

另外，若已知$[X]_{补}$，可以对$[X]_{补}$再求一次补，就得到$[X]_{原}$。表示为：

$$[[X]_{补}]_{求补}=[X]_{原}$$

例 7　$[X]_{补}=1\ 1\ 1\ 1\ 0\ 1\ 1\ 0$

$[X]_{原}=1\ 0\ 0\ 0\ 1\ 0\ 1\ 0$

从以上分析得到，8 位二进制数码，表示为无符号数为0～255；表示为原码为－127～＋127；表示为反码为－127～＋127；表示为补码为－128～＋127。如表1-2。

表 1-2　数的表示方法小结

8位二进制数码	无符号数	原　码	反　码	补　码
00000000	0	+0	+0	+0
00000001	1	+1	+1	+1
00000010	2	+2	+2	+2
⋮	⋮	⋮	⋮	⋮
01111101	125	+125	+125	+125
01111110	126	+126	+126	+126
01111111	127	+127	+127	+127
10000000	128	−0	−127	−128
10000001	129	−1	−126	−127
⋮	⋮	⋮	⋮	⋮
11111110	254	−126	−1	−2
11111111	255	−127	−0	−1

三、补码的加减运算

计算机中的补码运算是对带符号数运算而言，补码运算具有以下特点：

1．凡参加运算的数一律用补码表示，其运算结果也用补码表示。若结果的符号为0，表示正数；若结果的符号为1；表示负数，得到的是补码。

2．补码的符号位与数值部分一起参加运算，并自动获得结果（包括符号和数值部分）

补码加减法运算的一般公式是：

（1）补码加法：$[X]_{补}+[Y]_{补}=[X+Y]_{补}$

（2）补码减法：$[X]_{补}+[-Y]_{补}=[X-Y]_{补}$

其中，$[-Y]_{补}=[[Y]_{补}]_{变补}$。变补就是连同符号位一起求反加1，它与前面介绍的由补码求原码的表示式$[[X]_{补}]_{求补}$不同。

例 8　已知X＝＋0010001，Y＝－0000111，进行补码加法运算：

$[X]_{补}=$　00010001　　＋17的补码

$+[Y]_{补}=$　11111001　　－7的补码

$[X+Y]_{补}$ 1　00001010　　＋10的补码

进位自动丢失　　符号位

符号位为0，表示和为正数，正数的补码即为原码。结果，X＋Y＝＋0001010(＋10)

例 9　已知X＝－0111000，Y＝－0010001用补码进行减法运算

$[X]_{原}=10111000$

$[X]_{补}=11001000$

$[Y]_{原}=10010001$