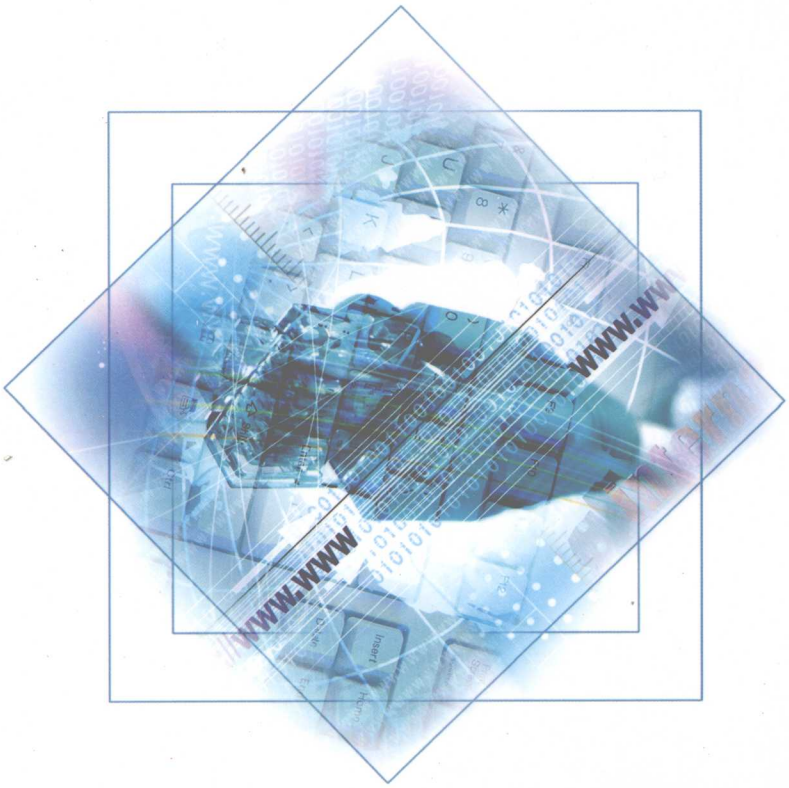
21世纪高等院校计算机系列规划教材

主编 常通义 副主编 孙立功

微型计算机原理及接口技术（第二版）



华中科技大学出版社
http:// www.hustp.com

21 世纪高等学校计算机系列规划教材

微型计算机原理及接口技术

(第二版)

主　编　常通义

副主编　孙立功

编　者　王　磊　程传洛　柳向斌

唐之义　孙向文

主　审　普杰信

华中科技大学出版社

图书在版编目(CIP)数据

微型计算机原理及接口技术(第二版)/常通义　主编
武汉:华中科技大学出版社,2006 年 2 月
ISBN 7-5609-3648-2

Ⅰ.微…
Ⅱ.①常…　②孙…
Ⅲ.微型计算机-接口-技术
Ⅳ.TP364

微型计算机原理及接口技术(第二版)　　常通义　主编

责任编辑:曾　光　　封面设计:刘　卉
责任校对:陈　骏　　责任监印:张正林

出版发行:华中科技大学出版社
武昌喻家山　　邮编:430074　　电话:(027)87557437

录　　排:武汉万卷鸿图科技有限公司
印　　刷:华中科技大学印刷厂

开本:787×1092　1/16　　印张:17.5　　字数:397 000
版次:2006 年 2 月第 2 版　　印次:2006 年 2 月第 3 次印刷　　定价:27.00 元
ISBN 7-5609-3648-2/TP・597

内 容 简 介

本书是为了适应计算机技术发展的形势和非电类专业“微型计算机原理及接口技术”的教学需要而编写的。全书共分九章：计算机基础知识，8086 微处理器，存储器，8086 指令系统，输入／输出和中断，可编程接口芯片，汇编语言程序设计，A/D 与 D/A 转换，总线。各章均附有习题，书后附有部分习题参考答案。

本书的主要读者对象是非电类专业的本科生，也可作为其他专业的本科生和工程技术人员的参考书以及自学用书。

前　言

电子计算机是信息技术的基础，是20世纪最卓越的成就之一。目前，计算机在科学计算、数据处理、过程控制、计算机辅助设计和制造(CAD／CAM)、人工智能方面的应用日益广泛深入，显示了旺盛的生命力和更为广阔的应用前景。本课程以8086为主讲机型，介绍微机的基本原理和应用问题。这里的“应用”主要不是指用于计算，而是指用于组成一个检测、控制等实时系统来解决科研生产中有关问题。一个计算机系统由硬件和软件两部分组成。系统中由计算机和其他实际电路元件等组成的组合体称为硬件。系统硬件的工作是由程序控制的，控制程序就是系统的软件。本课程的内容包括计算机的硬件设置和软件编制两部分内容。这些内容是计算机应用的基础。

本书是为了适应对非电专业学生进行16位微型机教学的需要，根据编者多年的“微型计算机原理及应用”教学经验，在试用的基础上再总结提高编写而成的。编写本书的指导思想和本书的特点是：立足应用，注重基本原理、基本概念和基本方法，精选内容、突出重点、叙述扼要、深入浅出。

全书共分九章，第一、二、七章由常通义编写，第三、四章由孙立功编写，第五章由王磊编写，第六章和附录由程传洛编写，第八、九章由唐之义编写。全书由常通义担任主编，孙立功担任副主编，普杰信教授担任主审。田葳、刘珊中、许春媚对书稿进行了认真审阅并提出了许多宝贵的意见和建议，赵海霞、阎保定参加了试用教材的编写工作，在此一并表示衷心感谢。

由于编者水平所限，书中难免有错误和不妥之处，敬请使用本书的师生和读者批评指正。

编　者

2003年6月于河南科技大学

再 版 前 言

本书第一版得到了广大读者，特别是使用它的教师和同学们的支持与关心，提出了许多宝贵的意见和建议。第二版根据第一版的使用情况以及上述意见和建议做了以下工作：(1)对第一章的内容进行了增删和调整；(2)充实了第七章的内容，增加了习题，并对书中所用程序在 DOS 下进行了精心调试；(3)增加了一个附录(宏汇编程序 MASM 和连接程序 LINK 的使用)；(4)增加了部分习题参考答案；(5)对第一版由于时间紧迫产生的有关问题予以订正。

第二版保留了第一版的主导思想和特点，相信本版更能满足非电专业的教学要求。本书第一、二章由常通义编写，第三、四章由孙立功编写，第五章由王磊编写，第六章及附录 A 由程传洛编写，第七章由柳向斌编写，第八、九章由唐之义编写，部分习题参考答案及附录 B 由孙向文编写。全书由常通义担任主编，孙立功担任副主编，普杰信教授担任主审。田葳、刘珊中、何谷慧参加了审稿。许春媚为本书出版提出了不少宝贵的意见和建议，并做了许多细致工作，对此表示衷心感谢。

由于编者水平有限，书中难免有不妥之处，希望读者，特别是使用本书的教师和同学提出批评和宝贵意见，以便不断完善和提高。

编　者

2005 年 7 月

目　录

第一章　计算机基础知识

计算机的基本功能是进行数值运算、逻辑运算和数据处理。计算机是一个复杂的数字逻辑系统，由若干基本部件组成。本章首先介绍计算机进行数值运算时采用的数制和实现基本运算的电路，其次介绍总线的概念与信息传输原理，接着介绍组成计算机的基本逻辑部件，最后通过一个计算机模型介绍各逻辑部件之间的联系和基本工作原理。本章内容是本课程的基础。

第一节　数　　制

数制是人们利用符号来计数的科学方法。日常生活和通常的科学计算中均采用十进制来计数，在计算机中则使用二进制、八进制、十六进制和二-十进制来计数。

一、数制的基与权

某种数制所使用的计数符号（数码）的个数叫做这种数制的基数，简称基；表示基本单位的数符“1”在某个数位上所表示的数值叫该数位的权。下面介绍上述各数制的基与权。

1. 十进制

十进制用 0、1、2、3、4、5、6、7、8、9 十个符号计数，它的基是 10。计数单位“1”在不同数位代表不同的数值，如图 1-1 所示。“1”在不同数位上所表示的数值有所不同，但都是 10 的整数指数幂，即各数位的位权都是 10 的整数指数幂，例如，个位的位权是 10^0（1），十位的位权是 10^1（10），十分位的位权是 10^{-1}（0.1）等。

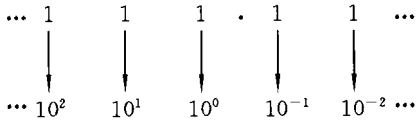


图 1-1　十进制的位权

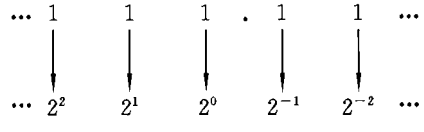


图 1-2　二进制的位权

2. 二进制

二进制采用“0”和“1”两个字符计数，其基为 2。与十进制相似，它的各数位位权也都是基（2）的整数指数幂。

如图 1-2 所示，小数点向左第一位的权是 2^0（1），第二位的权是 2^1（2）……小数点向右各位的权依次为 $2^{-1}(\frac{1}{2})$，$2^{-2}(\frac{1}{4})$……

3. 八进制

它采用0、1、2、3、4、5、6、7八个字符计数，其基是8，各位的权由小数点向左依次为8^0、8^1、8^2……由小数点向右依次为8^{-1}、8^{-2}……

4. 十六进制

它采用0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F十六个数符计数，其基为16。各位的位权从小数点往左依次为16^0、16^1、16^2……小数点向右依次为16^{-1}、16^{-2}……

5. 二-十进制

这是一种用四位二进制数码（BCD码）来表示十进制数字的计数制。它的本质是十进制，只是十个数符分别采用“0000”、“0001”、“0010”、“0011”、“0100”、“0101”、“0110”、“0111”、“1000”和“1001”十个四位二进制数表示。

二、数制之间的相互转换

二进制、十进制、十六进制可分别用字母B、D、H表示，如二进制数1101可写作1101B，十进制数43可写作43D（D一般可省略），十六进制数A2可写作A2H。

1. 将十进制数转换为二进制数

【例1-1】 将十进制整数13转换为二进制数。

解 将十进制整数转换为二进制数采用除2取余法，即将13除以2得商取出余数，再将商除以2得商取出余数，如此辗转相除直至商0取余为止，如图1-3所示。

然后将最后一个余数作高位，第一个余数为最低位依次排列即得所求的二进制数。由此可得

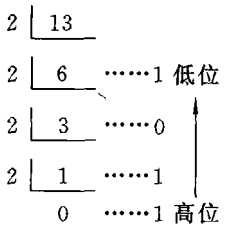


图1-3 除2取余法

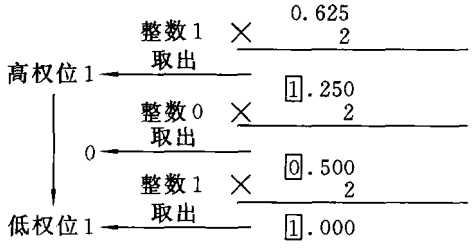


图1-4 乘2取整法

13D=1101B

【例1-2】 将十进制小数0.625转化为二进制数。

解 将十进制小数转换为二进制小数采用乘2取整法，即将纯小数乘2取出积的整数，再将积的小数部分乘2，取出积的整数……依次乘2取整直至积的小数部分为0（或得到适当的二进制的小数位数）为止。最后将最先取得的整数作高位，依次将所取乘积整数部分排列到小数点后就可以了，如图1-4所示。由此可得

0.625D=0.101B

【例 1-3】 将 13.625D 转换为二进制数。

解　将既有整数部分又有小数部分的十进制数转换成二进制数时，先用除 2 取余法转换整数部分，再用乘 2 取整法转换小数部分，最后将两部分拼合起来。这样可得

$$13.625D=1101.101B$$

2. 将二进制数转换为十进制数

采用将二进制数按权展开成多项式和的形式再求和的方法，即将各位数字乘以所在位的权再相加的方法进行转换。

【例 1-4】 将 101011B 转换为十进制数。

解　$101011B=1\times2^0+1\times2^1+0\times2^2+1\times2^3+0\times2^4+1\times2^5=43D$。

3. 二进制数与十六进制数间的相互转换

以小数点为基准向左向右每四位划为一节，最后不足四位的补 0 凑成四位，然后写出各节二进制数值的十六进制码，即可将二进制数转换为十六进制数。如

$$0011\ 1100.1110\ B=3C.EH$$

反之，将一个十六进制数的各位数字用四位二进制数来表示就可把十六进制数转换为二进制数。如

$$2A.5H=0010\ 1010.0101\ B$$

第二节　计算机的逻辑运算和数值运算

一、逻辑运算

计算机可以对二进制数进行“与”、“或”、“非”、“异或”等逻辑运算。逻辑运算是按位进行的。

设 A=10100111，B=11000001，则有如下逻辑运算。

1. “与”运算

A 和 B 的“与”运算如图 1-5 所示，即

$$Y=A\cdot B=10000001$$

2. “或”运算

A 和 B 的“或”运算如图 1-6 所示，即

$$Y=A+B=11100111$$

3. “非”运算

A 和 B 的“非”运算，即

$$Y=\overline{A}=01011000,\ F=\overline{B}=00111110$$

4. “异或”运算

A 和 B 的“异或”运算如图 1-7 所示，即

$$Y=A \oplus B=01100110$$

A: 10100111
∧) B: 11000001
10000001

图 1-5 “与”运算

A: 10100111
∨) B: 11000001
11100111

图 1-6 “或”运算

A: 10100111
⊕ B: 11000001
01100110

图 1-7 “异或”运算

二、数值运算

计算机进行数值运算时采用二进制。

1. 加法运算

二进制加法运算具有逢 2 进 1 的特点。两个一位二进制数加法运算的法则是：0＋0=0，0＋1=1，1＋0=1，1＋1=10。

（1）半加器。

设 A、B 为两个一位二进制数，如用 S 表示它们相加的本位和，C 表示进位值，可得出 A、B 求和运算的真值表（见表 1-1）。由表 1-1 可得

$$S=\overline{A}B+A\overline{B}, \qquad C=A \cdot B$$

根据以上逻辑式可画出如图 1-8 所示的电路，这就是半加器。

表 1-1 半加器真值表

A	B	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

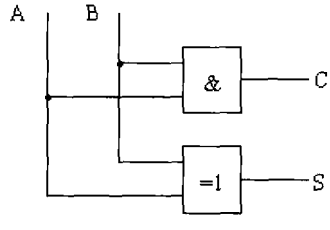


图 1-8 半加器

（2）全加器。

设 A、B 分别为 n 位二进制数，$A=A_{n-1}\cdots A_i\cdots A_0$，$B=B_{n-1}\cdots B_i\cdots B_0$。对 A、B 进行求和运算时，要从最低位开始按位相加，除最低位外，其余各位按位相加时必须考虑其相邻低位相加后向本位的进位，然后得出本位和和向高位的进位。例如对第 i 位求和时，其低位相加向本位的进位为 C_i，用 S_i 表示本位和，C_{i+1} 表示本位相加向高位的进位，可得出如表 1-2 所示的真值表。根据真值表可得 S_i 和 C_{i+1} 与 A_i、B_i、C_i 之间的逻辑关系式：$S_i=A_i \oplus B_i \oplus C_i$，$C_{i+1}=B_i \cdot C_i + A_i \cdot C_i + A_i \cdot B_i$，反映这种逻辑关系的电路如图 1-9 所示，这就是全加器。

表 1-2　全加器真值表

A_i	B_i	C_i	S_i	C_{i+1}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

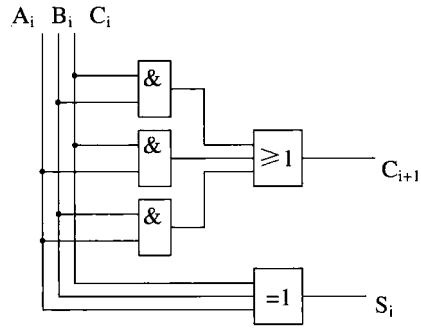
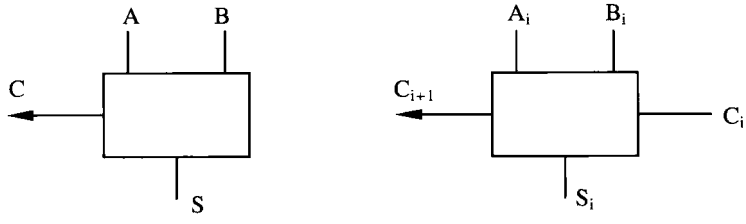


图 1-9　全加器

半加器和全加器的电路符号如图 1-10 所示。



（a）半加器　　（b）全加器

图 1-10　半加器和全加器的电路符号

图 1-11 所示的为一个由全加器和半加器组成的四位二进制加法电路。应该注意，若该加法电路的加数、被加数、和数都是四位，则向 4 权位的进位将自动丢失。

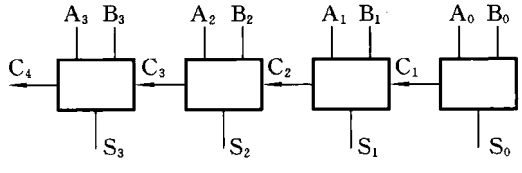


图 1-11　四位二进制加法电路

2. 减法运算——补码加法

数的正、负通常是分别用符号“+”、“-”表示的。在计算机中，数据是以二进制形式存放在存储单元内的，每一位或者是“0”或者是“1”，为了表示数的正、负，将一个数分成两部分，一部分表示数值叫数值位，另一部分表示数的正、负叫符号位。一般情况下，符号位是一位，放在数值位之前并指定用“0”表示正，用“1”表示负，即将符号数值化。例如：将+1010011B 表示为 01010011，将-1010011B 表示为 11010011。

n 位二进制数中最高位表示数的符号，其余 n-1 位表示数值，这样的数叫带符号数。符号数值化了的数也叫“机器数”。由于机器数的最高位表示符号，所以其形式值不一定等于真实数值，如机器数 11010011 的形式值是 211，而实际值是- 83。机器数对应的实际数值称为它的“真值”。

为了方便对机器数进行算术运算，提高运算速度，人们采用多种方法将符号位与数值位一起编码，常用的编码有原码、补码和反码。

（1）机器数的原码。

若机器数的数值位表示数的绝对值，称之为机器数的原码，简称原码。

例如：+1001001B 的八位二进制原码是 01001001B，－1001001B 的八位二进制原码是 11001001B。

（2）机器数的反码。

n 位二进制数 X 反码的定义如下

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

上式说明，正数的反码就是它自身，负数的反码符号位为 1，数值位是其对应的数值位按位求反。

例如：$[+1001001]_{反}=01001001$，$[-1001001]_{反}=10110110$。

（3）机器数的补码。

在计算机中引入补码是为了达到下列两个目的：

① 使符号位能和数值位一起参加运算，从而简化运算规则，节省运算时间；

② 使减法运算转化为加法运算，这样用加法器就既可以进行加法运算又可以进行减法运算，从而简化计算机中运算器的硬件线路。

下面给出补码的定义，说明补码的求法，并举例说明用补码加法运算代替减法运算的问题。

在计算机运算时数据的位数是有限的。字长为 n 的两数相加时，如果最高位产生了进位，该进位就被丢掉。这个被丢掉的进位值叫做模，这里模等于 2^n。如图 1-11 所示，加法电路加数、被加数、和数都是四位二进制数，加法运算由 3 权位向 4 权位的进位值（2^4），即模将被丢掉。

n 位二进制机器数 x（最高位是符号位，其余 n－1 位是数值位）的模是 2^n，其补码定义为

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

例如，机器数（原码）*X*= 01001001 （真值为+1001001），由于 X＞0，所以

$[X]_{补}=X=$ 0 1 0 0 1 0 0 1

↑—— 补码符号位

再如机器数（原码）Y = 11001001（真值为 －1001001），由于 X＜0，所以

$$[Y]_{补}=2^8+Y=2^8-|Y|=2^7+(2^7-|Y|)$$

$$=10000000+0110111=10110111$$

1 0 1 1 0 1 1 1

↑——补码符号位

【例 1-5】 已知 A=－1110000B，B =－0001111B ，试分别求出它们的补码和补码的和。

解 因为 A 和 B 都是负数，所以

$[A]_{补} = 2^7 + (2^7 + A) = 10000000B + 0010000B = 10010000B$

$[B]_{补} = 2^7 + (2^7 + B) = 10000000B + 1110001B = 11110001B$

$[A]_{补} + [B]_{补} = 10010000B + 11110001B = 1\ \ 1\ 0000001B$

丢掉 ←　↑　→ 符号位

由前面所举例子可知，按照定义求补码，一个正数的补码符号位为 0，数值部分就是该正数本身，即正数的补码就是其原码；负数的补码符号位为 1，数值部分是该负数原码的数值部分按位求反再在末位加 1，即负数的补码就是其反码加 1。

需要指出的是，－128 的补码为 10000000B，它不存在八位的原码和反码。“+0”和“－0”的八位补码都是 00000000B。八位二进制补码表示的数值范围是：－128D～127D。补码运算的结果仍为补码。如果已知机器数的补码求其原码，只要将补码再求一次补就可以了，即机器数的原码就是其补码的补码 。

【例 1-6】 已知 X= +1101001B，Y= +1001001B，试计算 Z = X－Y 之值。

解 （1）按照减法运算规则计算，即

Z = X－Y = 1101001B—1001001B = + 0100000B = +32D

（2）采用补码（8 位）加法计算。令 Y′ = －Y，$[Y']_{补}$ =10110111B，$[X]_{补}$ = 01101001B，Z = X－Y = X＋Y′，则

$$[Z]_{补} = [X]_{补} + [Y']_{补} = 01101001B + 10110111B = 00100000B$$

因为$[Z]_{补}$符号位为 0，所以 Z 是一个正数，它的原码与补码相同，即

$$[Z]_{原} = [Z]_{补} = 00100000B，\ Z = +0100000B = +32D$$

本例中两种方法计算的结果相同。这说明用补码加法运算可以代替减法运算，即若 X>0，Y>0，则 Z =X－Y 的运算可用下式进行：

$$[Z]_{补}=[X]_{补}+[-Y]_{补}$$

亦即减法运算的结果（差的补码）等于被减数的补码与减数对应的负数的补码相加运算的结果（补码）。因此，计算机中的数据在一般情况下用补码表示。

需要说明的是，在进行补码运算时，符号位也要一起参与运算，符号位运算向高位的进位被丢掉。例 1-6 的补码运算就是这样进行的。

【例 1-7】 设 X = + 1000001， Y = +1100000 ，试用补码（8 位）运算求 Z = X + Y。

解

$$[X]_{补} = 01000001B，\ [Y]_{补} = 01100000B$$

$$[Z]_{补} = [X]_{补} + [Y]_{补} = 10100001B$$

注意，这里 X 和 Y 是两个正数，其和也应是正数。但 $[Z]_{补}=[X]_{补}+[Y]_{补}=10100001$，从和的补码来看，和却变成了负数，显然出现了错误，这是为什么呢？原因就是 Z=X+Y=+129D 超出了 8 位补码所能表示的数值范围，即出现了溢出。计算机判断溢出常用的方法是：监测补码运算的进位情况，如果在运算过程中符号位和数值最高位都没有进位或者数值最高位运算向符号位有进位，符号位运算也出现了进位（如例 1-6），都不会溢出；而如果运算时只有数值最高位运算向符号位有进位或只有符号位运算出现了进位就有溢出（这里 $[Z]_{补}=[X]_{补}+[Y]_{补}=01000001+01000000=10100001$，就是只有数值最高位运算向符号位有进位，而符号位运算没有进位）。溢出会使运算结果出错，应注意避免出现溢出错误。

【例 1-8】 已知$[X]_{原}$=10101010，$[Y]_{原}$ = 01100110，用补码运算求 Z =X+Y 和 Z' = X－Y，并判断有无溢出错误。

解 $[X]_{补}$= 11010110B，$[Y]_{补}$ = 01100110B，$[-Y]_{补}$= 10011010，所以

$$[Z]_{补} = [X]_{补}+[Y]_{补} = 11010110B + 01100110B = 00111100$$

由于运算过程中符号位位和数值最高位都有进位，所以没有溢出。$[Z]_{补}$符号位为 0，故 Z 为正数，且

$$[Z]_{原} = [Z]_{补} = 00111100，\ Z = +0111100B = +60D$$

$$[Z']_{补} = [X]_{补}+[-Y]_{补} = 11010110B + 10011010B = 01110000B$$

由于运算过程中只有符号位运算出现了进位，所以发生了溢出。显然，若 X=－42D，Y=102D，那么 Z = X－Y =－42D－102D =－144D 超出了 8 位二进制补码所能表示的数值范围。

第三节 信息交换码

计算机与外部设备（如打印机、显示器等）交换的信息除了数字以外还有字符（如 A、B、C、a、b、c、!、+、# 等），控制信号（如“标题开始”、“正文结束”、“换行”、“纸尽”等）等。这些信息也必须用二进制代码的形式去传输。因此需要约定上述信息的代码。这种约定的代码叫信息交换码。常用的信息交换码有标准化的 ASCII 码、EBCDIC 码和汉字编码。

一、ASCII 码

ASCII 码是美国信息交换标准委员会制定的一种七位二进制码，因为是七位编码，所以可表示 2^7（128）种码字。ASCII 码是使用最广泛的一种信息交换码：字符编码有 52 个表示英文字母的大小写；还有标点符号、空格、括号等 33 个；各种控制码共 33 个。表 1-3 是 ASCII 码编码表。

表 1-3 ASCII 码编码表

LSD \ MSD		0 000	1 001	2 010	3 011	4 100	5 101	6 110	7 111
0	0000	NUL	DLE	SP	0	@	P	、	p
1	0001	SOH	DC1	!	1	A	Q	a	q
2	0010	STX	DC2	″	2	B	R	b	r
3	0011	ETX	DC3	#	3	C	S	c	s
4	0100	EOT	DC4	$	4	D	T	d	t
5	0101	ENG	NAK	%	5	E	U	e	u
6	0110	ACK	SYN	&	6	F	V	f	v
7	0111	BEL	ETB	′	7	G	W	g	w
8	1000	BS	CAN	(	8	H	X	h	x