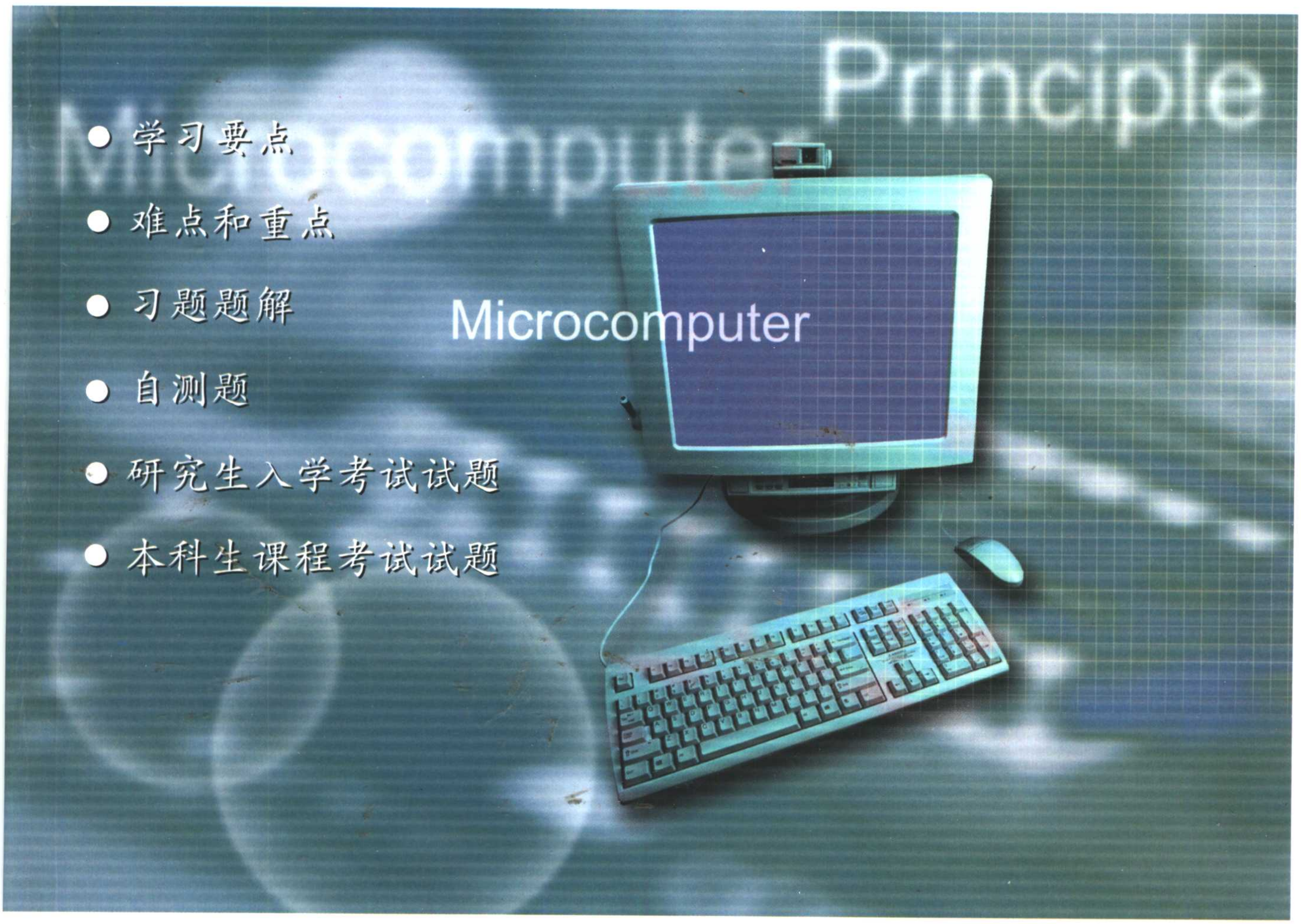
21世纪

高等学校电子信息类系列教材

《微型计算机原理（第四版）》学习指导

■ 乔瑞萍　欧文　编



- 学习要点
- 难点和重点
- 习题题解
- 自测题
- 研究生入学考试试题
- 本科生课程考试试题

西安电子科技大学出版社
http://www.xduph.com

☆ 21世纪高等学校电子信息类系列教材

《微型计算机原理(第四版)》
学 习 指 导

乔瑞萍　欧文　编著

西安电子科技大学出版社
2002

内 容 简 介

本书作为教材《微型计算机原理(第四版)》(姚燕南、薛钧义主编，西安电子科技大学出版社出版)一书的学习指导，对1～8章作了要点、难点和重点分析，并补充了微处理器外部结构和总线操作时序部分的内容(第9章)。通过分析，帮助学生加深对课本的理解，并对书中的大部分习题和思考题作了解答。此外，还增添了一些自测题，以供学生检查对知识点掌握的程度。

本书也可作为高等学校非计算机专业微型计算机原理(16/32位机)课程的教学参考与学习指导书。

图书在版编目(CIP)数据

《微型计算机原理(第四版)》学习指导/乔瑞萍等编著.
—西安：西安电子科技大学出版社，2002.10
21世纪高等学校通信类系列教材
ISBN 7-5606-1169-9

Ⅰ.微… Ⅱ.乔… Ⅲ.微型计算机—高等学校—教学参考资料 Ⅳ.TP36

中国版本图书馆CIP数据核字(2002)第056371号

策　　划　夏大平
责任编辑　雷鸿俊
出版发行　西安电子科技大学出版社(西安市太白南路2号)
电　　话　(029)8227828　邮　编　710071
http://www.xduph.com　E-mail:xdupfxb@pub.xaonline.com
经　　销　新华书店
印　　刷　陕西画报社印刷厂
版　　次　2002年10月第1版　2002年10月第1次印刷
开　　本　787毫米×1092毫米　1/16　印张12.25
字　　数　284千字
印　　数　1～6 000册
定　　价　14.00元
ISBN 7-5606-1169-9/TP·0601

XDUP 1440001—1

前　　言

随着计算机技术的不断发展，“微机原理与接口技术”这门课程的教材也在不断更新，《微型计算机原理（第四版）》一书作为电子工业部“九五”重点规划教材深受广大教师和学生的喜爱，已陆续重印多次。第四版以 80x86 为样板机，与第三版相比增加了不少 80386/80486 CPU 的新知识，因而难度也相应增加。为了配合《微型计算机原理（第四版）》的教学，我们编写了这本学习指导书，书中每章都从要点、难点、重点、例题分析、习题解答等多方面入手。首先根据我们多年的教学实践进行归纳总结提高后，对本书各章节的要点、难点和重点做了详细的分析，并对典型例题及习题的解题过程给出了示范，目的在于帮助学生深入掌握基本概念，并教会学生解题的思路，启发和鼓励学生要多作习题，加深对教学内容的理解；同时，考虑到“微机原理与接口技术”是一门实践性很强的课程，要求学生不仅会作习题，还应该能熟练上机解决实际问题。为此，本学习指导书的多数程序都在高版本 MASM 6.11 汇编下上机调试通过，目的在于启迪学生思维，教会学生亲自动手的能力，激发学生的学习兴趣，使学生牢固掌握教材内容，并学以致用。为了拓宽学生学习的思路，我们收集了大量习题及试题，精选后给出了一些自测题，学生在认真学习各章内容并扎实掌握好习题、例题的解题思路后，这些自测题是不难得到答案的。

本书第 1～6 章由乔瑞萍编写，第 7～9 章和附录由欧文编写。为了便于学习，增添的第 9 章内容可放在第 6 章之后学习。

西安交通大学姚燕南教授在本书整个编写过程中倾注了大量心血，提出了许多珍贵建议并做了全面、细致的审校，在此表示最诚挚的谢意。

由于编者水平有限，书中难免有错，恳请读者批评指正。

编　著
2002 年 3 月
于西安交通大学

目　　录

第 1 章　微型计算机中的数据类型

1.1 学 习 要 点

▲ 常用数据类型

▲ 数的进位制表示约定

▲ 整数运算

▲ 数学协处理器的数据格式

1.1.1 常用数据类型

80x86 系列微机中，常用的数据类型为：有(无)符号整数、BCD 码(8421 码)、ASCII 码、浮点数等。

1. 数据在内存储器中的存储方式

1) 几个常用概念

(1) 字长：计算机字所含二进制位数。计算机字也就是作为一个整体被一次传送或运算最多的二进制数位。它和计算机能够处理二进制信息的位数是两个概念。如 32 位和 32 位数相加，用 8 位机需加 4 次，用 16 位机需加 2 次，用 32 位机加 1 次即可。字长是对某一型号的机器而言的。

(2) 字节：1 个字节(Byte)＝8 位(bit)。

(3) 字：1 个字(Word)＝2 个字节＝16 位。

(4) 双字：1 个双字(Double Word)＝2 个字＝4 个字节＝32 位。

(5) $1\ K=2^{10}=1024$。

(6) $1\ M=2^{20}$。

(7) $1\ G=2^{30}$。

2) 存储规则

多字节数据的存储采取高位字节在地址号高的单元中，低位字节在地址号低的单元中的规则。数据在内存中常以字节为单位进行存储，即一个字节占用内存的一个地址。

2. 数的进位制表示约定

1) 计算机技术中常用的进位制数

(1) 数字：阿拉伯数字 0～9。

(2) 位置计数法(positional notation)

设待表示的数为 N，则 X 进位制 N 的表达式为

$$(N)_X=\sum_{i=-m}^{n-1}a_iX^i,\quad \text{等式右边计算结果是十进制数真值}$$

其中：

X：基数，若取 2、16、10，则得二进制、十六进制、十进制的表达式；

a_i：系数，可在 0～$X-1$ 中任意取值，十六进制系数范围是 0～15，其中 10～15 用 A～F 表示；

X^i：权，每个数位所具有的位值；

n：整数位数；

m：小数位数。

采用汇编语言编写程序时，数制约定：

① 二进制加后缀 B(Binary)。

② 十六进制加后缀 H(Hexadecimal)。

③ 十进制加后缀 D(Decimal)或不加。

(3) 特点：

① 可以少数的数字表示很大的数。

② 划分数位，按位累计，由低到高进位。

③ 相同的数码在不同的位代表不同的数值，例：666。

数字和位置计数法构成了数(Number)。

The number 58042 contains five digits.

2) 数制之间的转换

(1) 其他进制数⟶十进制：加权法，直接按 $(N)_X = \sum_{i=-m}^{n-1} a_i X^i$ 计算。

(2) 十进制⟶二进制：降幂法，除基数取余法(整数)，乘基数取余法(小数)。

(3) 十六进制⟶二进制：十六进制是一种很重要的短格式计数法，它把二进制数每 4 位分成一组，分别用 0～9 和 A～F 来表示 0000～1111；反之，十六进制数的每一位用 4 位二进制表示，就是相应的二进制数。

3. 整数

整数分为有符号数和无符号数两种。

无符号数：数的绝对值，即字长的每一位都为数值位。

有符号数：字长的最高位是符号位，其余各位为数值位。

1) 有符号数的表示方法

有符号数的表示有原码、补码、反码及移码等 4 种。以下分别介绍它们的定义。

① 原码定义：假定字长为 n，则

$$[X]_{原} = \begin{cases} X & 0 \leqslant X < 2^{n-1}；X \text{ 为正数，MSD} = 0 \\ 2^{n-1} - X & -2^{n-1} < X \leqslant 0；X \text{ 为负数，MSD} = 1 \end{cases}$$

说明：最高位 MSD 为符号位，其余 $n-1$ 位为数值位。

用途：浮点数的有效数字常用原码表示，二进制乘除运算时也多采用原码表示。

② 补码定义：假定字长为 n，则

$$[X]_{补} = \begin{cases} X & 0 \leqslant X < 2^{n-1}；X \text{ 为正数} \\ 2^n + X & -2^{n-1} \leqslant X < 0；X \text{ 为负数} \end{cases}$$

说明：模为 2^n，正数的原码与补码表示是相同的。

③ 反码定义：假定字长为 n，则

$$[X]_{反} = \begin{cases} X & 0 \leqslant X < 2^{n-1}；X 为正整 \\ 2^n - 1 + X & -2^{n-1} \leqslant X < 0；X 为负数 \end{cases}$$

说明：当 X 为正数时，$[X]_{原}=[X]_{补}=[X]_{反}=0X_{n-2}X_{n-3}\cdots X_1X_0$；当 X 为负数时的反码等于其原码除符号位不变(等于 1)外，其余各位全部变反。

用途：求反逻辑运算，完成一定的计算或控制功能。

④ 移码定义：假定字长为 n，则

$$[X]_{移}=2^{n-1}-1+X \quad -2^{n-1}<X\leqslant 2^{n-1}$$

说明：移码是在数的真值上加一个偏移量形成的，偏移量为 $2^{n-1}-1$。

用途：用作 A/D 和 D/A 转换器的双极性编码，也可用在浮点数的阶码表示中。

2) 补码的运算

(1) 补码的求法：

① 按定义求补码。

② 将$[X]_{原}$除符号位以外，其余各位按位取反，最低位加 1。

③ 将$[X]_{原}$从其最低位起到出现第一个 1 以前(包括第一个 1)的原码中的数字不变，以后逐位取反，符号位不变。

④ 已知$[X]_{原}$，求$[X]_{补}$。其方法为：

若符号位为 0，则$[X]_{补}=[X]_{原}$(X 为正数)；

若符号位为 1，则$[[X]_{补}]_{补}=[X]_{原}$(X 为负数)。

⑤ 变补：已知$[Y]_{补}$求$[-Y]_{补}$。其方法是将$[Y]_{补}$连同符号位一起求反加“1”而得。

(2) 补码的运算：

① 符号位与数值位一同参加运算。

② 有符号数的运算的所有数(包括结果)均为补码形式。

③ 运算规则：

“加”：$[X+Y]_{补}= [X]_{补}+[Y]_{补}$

“减”：$[X-Y]_{补}= [X]_{补}-[Y]_{补}$

$[X-Y]_{补}= [X]_{补}+[-Y]_{补}$

(3) 补码的特点：

① 数的范围：若字长为 n，则$-2^{n-1}\leqslant X<2^{n-1}$。

② “0”的表示唯一。

③ 减运算是变补相加。

3) 有符号数的溢出(Overflow)问题

(1) 定义：运算值超出其表示范围。即参加运算的数(初/中/终)值超过了计算机允许的表示范围($-2^{n-1}\leqslant X<2^{n+1}$)，产生错误结果，称溢出。对溢出的处理为

$$溢出处理-\begin{cases}停机 \\ 检查程序转入 \text{ INTO}\end{cases}$$

(2) 溢出条件：同号数相加/异号数相减，才可能产生溢出。

C_s：表示符号位进位，如有进位，$C_s=1$，否则，$C_s=0$。

C_p：表示数值最高位(字长次高位) 进位，如有进位，$C_p=1$，否则，$C_p=0$。

(3) 溢出判别：采用双高位判别法，即

逻辑关系：$V=C_s\oplus C_p$

$$
\text{溢出标志}\begin{cases} V=1 & \text{有溢出}\begin{cases} C_s=0, & C_p=1 \\ C_s=1, & C_p=0 \end{cases} \\ V=0 & \text{无溢出}\begin{cases} C_s=0, & C_p=0 \\ C_s=1, & C_p=1 \end{cases} \end{cases}
$$

4）无符号数的运算

无符号数与带符号数加、减法是一样的，正负号取决于减法结果，CF=0 为正，CF=1 为负。

注意：（1）溢出判别范围：$0\sim2^n-1$；

（2）溢出条件：CF=1。

5）算术移位

二进制数在寄存器或存储器中进行算术移位时，左移一位将引起倍增，右移一位将引起倍减（减半）。

（1）对于正数，左移或右移时空位都补以 0。

（2）补码表示的负数，左移时最低位补以 0，右移时最高位补以 1。

（3）反码表示的负数，左移和右移时，最低位和最高位均补以 1。

4．字符串、位及位串

（1）字符串是 80x86 系列微处理机处理的数据类型之一。字符串包括字节串、字串和双字串。

操作：串传送、比较、查找、插入。

（2）80386/80486 微机中有位串操作，一个位串最长为 2^{32} 位。其中：

字节地址＝位偏移量÷8；

位在所在字节中的位置＝位偏移量÷8 取余数。

用途：设置状态。如：打印机忙（1），闲（0）。

5．BCD 码

BCD（Binary Coded Decimal）码是一种二进制编码的十进制数，最常用的是 8421 码。

1）特点

因人们习惯于十进制，但计算机仅识别二进制，为解决此矛盾采用 BCD 码。BCD 码的主要特点如下：

（1）逢十进位。

（2）4 位二进制数表示一位十进制数。

因 $2^4=16$，仅取其中 10 种组合，分别代表 0～9 这 10 个数字。为了便于记忆和比较直观，常用 8421BCD 码。其中 8、4、2、1 分别是 4 位二进制数的位权值 2^3、2^2、2^1、2^0。8421BCD 码具有如下特点：

① 按自然二进制规律易转换（D ↔ B）。

$$
\begin{array}{cc} 0 & 0000 \\ \wr & \wr \\ 9 & 1001 \end{array}
$$

例：$(25.4)_{10}=(00100101.0100)_{BCD}$

② 各位数分别为 8、4、2、1。

③ 具有奇偶性（通信传输中易实现判断）：

奇：尾数是 0；

偶：尾数是 1。

说明：同一个 8 位二进制数，当认为它表示的是 BCD 时，其数值与 8 位二进制的不同。例如，对于 00011000B，它表示的二进制值为 24，而其 BCD 值则为 18。

2）基本格式

（1）组合式 BCD 码：两位 BCD 数占一字节。其基本格式为

高	低
1	8
0001	1000

（2）分离式 BCD 码：两位 BCD 数占两字节中的低四位。其基本格式为

$$\underbrace{\text{XXXX}}_{\text{高四位无关}}0001 \qquad \underbrace{\text{XXXX}}_{\text{高四位无关}}1000$$

80387 支持的压缩（组合）BCD 数长度为 80 位二进制数。80 位二进制可表示为 20 位 BCD 数，但 80387 只用了 18 位 BCD 数。这与高级语言相符合。最高 8 位中最高一位为符号位，其余 7 位不用。

3）BCD 码的运算

这里以组合 BCD 码为例介绍 BCD 码的运算（加/减）。

（1）不能将 BCD 码直接交计算机运算（即当作二进制数）。因为 BCD 码为十进制，逢十进/借位，用 4 位二进制表示一个 BCD 码时逢十六进/借位，所以必须加、减 6 进行“修正”才能得正确结果。

（2）修正方法：加、减 6 修正。

① 加法：(a) BCD 对应位相加，结果小于等于 9，不修正（无进位）；

(b) BCD 对应位相加，结果大于 9 小于 16，加 6 修正；

(c) BCD 对应位相加，相加有进位，结果大于等于 16，加 6 修正；

(d) 两高位对应位加低位进位结果处理同(a)～(c)。

② 减法：因两个 BCD 进行减法运算时，当低位向高位有借位时“借一作十六”，而本应“借一作十”，多了“6”，故减 6 修正，规则及方法同加法。

一般计算机均有十进制调整指令。使用时跟在二进制运算指令之后修正为正确结果。

6. ASCII 码

计算机中的字符数据用 ASCII 码表示，一个字符在存储器中占用一个字节（8 位二进制码）。用 7 位 ASCII 码可表示 128 种字符，其中包括 32 个控制符、10 个数字（0～9）、52 个大/小写英文字母和 34 个专用符号（运算符、标点、括号等）。例如，数字 0 的 ASCII 码为 30H，9 的 ASCII 码为 39H，字母 A 的 ASCII 码为 41H，等等。

另外，常采用最高位（即奇偶校验位）来检测数据传输、存储过程中的错误。

1.1.2 数学协处理器的数据格式

1. 80387 支持的数据类型

80386 的数学协处理器 80387 主要任务为支持浮点运算，可支持 7 种数据类型，其中又分整型数(4 种)和实型数(3 种)两大类。

$$\text{整型数}\begin{cases}\text{字整数(16 位)}\\ \text{短整数(32 位)}\\ \text{长整数(64 位)}\\ \text{BCD 数(80 位)}\end{cases}\qquad\qquad \text{实型数}\begin{cases}\text{短实数(32 位)}\\ \text{长实数(64 位)}\\ \text{临时实数(80 位)}\end{cases}$$

2. 实型数格式

(1) 定点数：小数点在数中的位置不变。

浮点数：小数点位置随阶码移动。

实型数：以浮点数表示的带小数点的数。

(2) 形式：

$$N=\pm S\cdot 2^{J}$$

其中，N 为二进制数；S 为有效数字或尾数；J 为指数或阶码。

(3) 格式：

S	指数	有效数字

符号位 $\begin{cases}0\text{—正}\\ 1\text{—负}\end{cases}$

其中，指数通常采用移码表示。

此外，80387 规定，任何实型数只能用如下格式表示：

1. ×××…×× · 2^n(×表示 0 或 1)

这说明实型数的小数点前仅有一位，且该位永远为 1。

(4) 浮点数的特点：

① 相同字长下，浮点数表示数值范围大得多。

② 采用浮点数运算，需要“对阶”(使阶码相等的操作称为对阶)，编程麻烦。

③ 结果要用规格化标准形式表示(高级语言中的指数形式)。

1.2 难点和重点

本章的难点与重点在于掌握数制转换及补码运算。

1. 数制转换

数制转换重点要掌握十进制转换为二进制的方法，其转换一般用降幂法。降幂法的步骤为：首先写出要转换的十进制数，其次写出所有小于此数的各位二进制权值，然后用要转换的十进制数减去与它最相近的二进制权值，如够减则减去并在相应位记以 1，如不够减则在相应位记以 0 并跳过此位。如此不断反复，直到该数为 0 为止。

例 1.1 $N=23.625\text{D}$，用降幂法将其转换为二进制，并将其表示成短实型数形式。

解：第一步：将 N 转换为二进制形式。

小于 23D 整数部分的二进制权为

	2^4	2^3	2^2	2^1	2^0	
	16	8	4	2	1	
对应的二进制数是	1	0	1	1	1	
	(16+	0+	4	+2	+1	=23D)

小于 0.625 小数部分的二进制权为

	2^{-1}	2^{-2}	2^{-3}	
	0.5	0.25	0.125	
对应的二进制数是	1	0	1	
	(0.5+	0+	0.125	=0.625D)

所以 $N=23.625\text{D}=10111.101\text{B}$。

第二步：将 N 表示成短实型数形式。

$$\text{N}=10111.101\text{B}=1.0111101\cdot 2^4$$

短实型数格式为

31	23	0
$S(1)$	指数(8)	有效数字(23)

其中：

S：0（正数）；

指数(用移码表示)：00000100B+01111111B=10000011B；

有效数字(用原码表示)：011110100…00(隐藏了 1，小数点前仅有一位，该位永远为 1)。

2. 补码的加减运算

补码加减运算的特点是符号位与数值位一同参加运算。作减法时，可将减数变补与被减数相加来实现。运算时要注意字长、数值范围及溢出判别。一般只有在同号数相加或异号数相减时，才可能产生溢出。

例 1.2 设字长为 8 位，$X=11001011\text{B}$，$Y=01001111\text{B}$。

(1) 若将两数视为带符号位数，求 $X-Y$，并判别结果是否溢出。

(2) 若将两数视为不带符号位数，求 $X+Y$，并判别结果是否溢出。

解：(1) 因为计算机处理的均为补码数，所以

$[X]_{原}=[[X]_{补}]_{补}=[11001011\text{B}]_{补}=10110101\text{B}=-(2^5+2^4+2^2+2^0)\text{D}=-53\text{D}$(负数)

$[Y]_{原}=[Y]_{补}=01001111\text{B}=+(2^6+2^3+2^2+2^1+2^0)\text{D}=+79\text{D}$(正数)

$[-Y]_{补}=10110001\text{B}$(将$[Y]_{补}$连同符号位一起求反加“1”而得)

$X-Y=[X]_{补}-[Y]_{补}=[X]_{补}+[-Y]_{补}$

$[X]_{补}-[Y]_{补}$：

$$
\begin{array}{rll}
 & 11001011B & [-53]_{补} \\
- & 01001111B & [+79]_{补} \\
\hline
 & 01111100B & [+124]_{补}
\end{array}
$$

$C_s=0$，$C_p=1$，$OF=C_s \oplus C_p=1$，负数减正数，运算结果为正，无效。

因$(-53)-(+79)=-132<-128$，超出负数范围。

$[X]_{补}+[-Y]_{补}$：

$$
\begin{array}{rll}
 & 11001011B & [-53]_{补} \\
+ & 10110001B & [-79]_{补} \\
\hline
 & 01111100B & [+124]_{补}
\end{array}
$$

$C_s=1$，$C_p=0$，$OF=C_s \oplus C_p=1$，负数加负数，运算结果为正，无效。

因$(-53)+(-79)=-132<-128$，超出负数范围。

(2) 因为 X、Y 为不带符号数，字长的每一位都为数值位，所以

$X=11001011B=2^7+2^6+2^3+2^1+2^0=203D$

$Y=01001111B=79D$

$X+Y$：

$$
\begin{array}{rll}
 & 11001011B & 203 \\
+ & 01001111B & 79 \\
\hline
 & \boxed{1}00011010B & 26
\end{array}
$$

丢弃

上式中各补码数被视为无符号数，即 11001011B＝203D，01001111B ＝79D，显然 203D＋79D≠26D，运算结果无效。这是因为 $203+79>2^8-1=255$，CF＝1，超出无符号数的范围。

注意：有、无符号数补码真值的区别，这是由用户程序来实现的。有符号数的溢出是由双高位来判别，即 $OF=C_s \oplus C_p$，而无符号数的溢出则由 CF 来决定。

1.3 习题题解

2. 设机器字长为 6 位，写出下列各数的原码、补码、反码和移码：

10101　　11111　　10000

－10101　　－11111　　－10000

解：题中部分数的求解结果见表 1.1。

表 1.1

真值	原码	补码	反码
10101	010101	010101	010101
－10101	110101	101011	101010
－10000	110000	110000	101111

5. 设机器字长为 8 位，最高位为符号位，试对下列各算式进行二进制补码运算：

(1) 16+6=?　　　　(2) 8+18=?

(3) 9+(-7)=?　　　　(4) -25+6=?

(5) 8-18=?　　　　(6) 9-(-7)=?

(7) 16-6=?　　　　(8) -25-6=?

解：(2) 设 $X=8$，$Y=18$。因为 X，Y 为正数，所以

$[X]_{原}=[X]_{补}=00001000B$

$[Y]_{原}=[Y]_{补}=00010010B$

$$\begin{array}{rll} & 00001000B & [8]_{补} \\ + & 00010010B & [18]_{补} \\ \hline & 00011010B & [26]_{补} \end{array}$$

$C_s=0$，$C_p=0$，$OF=C_s \oplus C_p=0$，无溢出。

(4) 设 $X=-25$，$Y=6$。因为 X 为负数，Y 为正数，所以

$[X]_{原}=10011001B$，$[X]_{补}=11100111B$

$[Y]_{原}=[Y]_{补}=00000110B$

$$\begin{array}{rll} & 11100111B & [-25]_{补} \\ + & 00000110B & [6]_{补} \\ \hline & 11101101B & [-19]_{补} \end{array}$$

$C_s=0$，$C_p=0$，$OF=C_s \oplus C_p=0$，无溢出。

(6) 设 $X=9$，$Y=-7$。因为 X 为正数，Y 为负数，所以

$[X]_{原}=[X]_{补}=00001001B$

$[Y]_{原}=10000111B$，$[Y]_{补}=11111001B$

$[-Y]=00000111B$

$$\begin{array}{rll} & 00001001B & [9]_{补} \\ + & 00000111B & [7]_{补} \\ \hline & 00010000B & [16]_{补} \end{array}$$

$C_s=0$，$C_p=0$，$OF=C_s \oplus C_p=0$，无溢出。

(8) 设 $X=-25$，$Y=-6$。因为 X 为负数，Y 为正数，所以

$[X]_{原}=10011001B$，$[X]_{补}=11100111B$

$[Y]_{原}=[Y]_{补}=00000110B$

$[-Y]_{补}=11111010B$

$$\begin{array}{rll} & 11100111B & [-25]_{补} \\ + & 11111010B & [-6]_{补} \\ \hline & 11100001B & [-31]_{补} \end{array}$$

$C_s=1$，$C_p=1$，$OF=C_s \oplus C_p=0$，无溢出。

6. 设机器字长为 8 位，最高位为符号位，试用“双高位”法判别下述二进制运算有没有溢出产生。若有，是正溢出还是负溢出？

(1) 43+8=？　　　　(2) −52+7=？

(3) 50+84=？　　　　(4) 72−8=？

(5) −33+(−37)=？　　(6) −90+(−70)=？

解：(2) 设 $X=-52$，$Y=7$。因为 X 为负数，Y 为正数，所以

$[X]_{原}=10110100B$，$[X]_{补}=11001100B$

$[Y]_{原}=[Y]_{补}=00000111B$

$$\begin{array}{rll} & 11001100B & [-52]_{补} \\ + & 00000111B & [+7]_{补} \\ \hline & 11010011B & [-45]_{补} \end{array}$$

$C_s=0$，$C_p=0$，$OF=C_s \oplus C_p=0$，无溢出。

(4) 设 $X=72$，$Y=8$。因为 X、Y 为正数，所以

$[X]_{原}=[X]_{补}=01001000B$

$[Y]_{原}=[Y]_{补}=00001000B$

$[-Y]_{补}=11111000B$

$$\begin{array}{rll} & 01001000B & [+72]_{补} \\ + & 11111000B & [-8]_{补} \\ \hline & 01000000B & [+64]_{补} \end{array}$$

$C_s=1$，$C_p=1$，$OF=C_s \oplus C_p=0$，无溢出。

(6) 设 $X=-90$，$Y=-70$。因为 X、Y 为负数，所以

$[X]_{原}=11011010B$，$[X]_{补}=10100110B$

$[Y]_{原}=11000110B$，$[Y]_{补}=10111010B$

$$\begin{array}{rll} & 10100110B & [-90]_{补} \\ + & 10111010B & [-70]_{补} \\ \hline & 01100000B & [+96]_{补} \end{array}$$

$C_s=1$，$C_p=0$，$OF=C_s \oplus C_p=1$，两负数相加，结果为正，负溢出。

8. 已知位 b_i 及 b_j 在位串中的地址(位偏移量)分别为 92 和 −88，试求它们各自在位串中的字节地址及其在所在字节中的位置。

解：b_i 在 m+11 字节中的第 4 位；b_j 在 m−11 字节中的第 0 位。

9. 将下列十进制数变为 8421BCD 码：

(1) 8609；　　　　(2) 5324。

解：(2) 5324=0101 0011 0010 0100B，写为 5324H

10. 将下列 8421BCD 码表示成十进制数和二进制数：

(1) 01111001B；　　(2) 10000011B。

解：(2) 10000011B 即 83H，将其表示成十进制数为 83，表示成二进制数为 01010011B。