第 2 章　计算机中的数据表示

本章介绍计算机中数据的表示方法，重点处理计算机中常用的数制及其转换、带符号数的表示、字符编码的基本知识。

通过本章的学习，应重点理解和掌握以下内容:

- 计算机中数制的基本概念及其表示
- 不同数制之间的相互转换
- 无符号数和带符号数的表示方法
- ASCII 码和 BCD 码的相关概念及其应用

2.1　计算机中的数制及其转换

数据是一种特殊的信息表达形式，不仅可由人来进行加工处理，而且更适合计算机进行高效率的加工处理、传递及转换。计算机中的数据包括了能够处理的各种数字、文字、图画、声音和图像等。

在计算机内，不论是指令还是数据，都采用了二进制编码形式，包括图形和声音等信息，也必须转换成二进制数的形式才能存入计算机中。

由于计算机的处理对象是各种各样的数据，在使用上，我们把计算机中的数据分为数值型数据和非数值型数据两类，前者指日常生活中接触到的数字类数据，可用来表示数量的多少并进行计算处理；后者指常用的字符型数据以及图画、声音和活动图像等。

2.1.1　数制的基本概念

1. 数的表示

人们在日常生活当中最熟悉、最常用的数是十进制数，它采用 0～9 共 10 个数字符号及其进位来表示数的大小。其相关概念如下：

- 0～9 这些数字符号称为“数码”。
- 全部数码的个数称为“基数”，十进制数的基数为 10。
- 用“逢基数进位”的原则进行计数，称为进位计数制。十进制数的基数是 10，所以其计数原则是“逢十进一”。
- 进位以后的数字，按其所在位置的前后将代表不同的数值，表示各位有不同的“位权”。

例如：十进制数个位的“1”代表 1，即个位的位权是 1；

十进制数十位的“1”代表10，即十位的位权是10；

十进制数百位的“1”代表100，即百位的位权是100；依此类推。

- 位权与基数的关系是：位权的值等于基数的若干次幂。

例如：十进制数 2518.234 可以展开为下面的多项式：

$2518.234=2\times10^3+5\times10^2+1\times10^1+8\times10^0+2\times10^{-1}+3\times10^{-2}+4\times10^{-3}$

式中：10^3、10^2、10^1、10^0、10^{-1}、10^{-2}、10^{-3} 等即为该位的位权，每一位上的数码与该位权的乘积就是该位的数值。

- 任何一种数制表示的数都可以写成按位权展开的多项式之和，其一般形式为：

$$N=d_{n-1}b^{n-1}+d_{n-2}b^{n-2}+d_{n-3}b^{n-3}+\cdots+d_{-m}b^{-m}$$

式中：

n——整数的总位数。

m——小数的总位数。

$d_{\text{下标}}$——表示该位的数码。

b——表示进位制的基数。

$b^{\text{上标}}$——表示该位的位权。

2. 计算机中常用的进位计数制

计算机内部的电子部件有两种工作状态，即电流的“通”与“断”（或电压的“高”与“低”），因此计算机能够直接识别的只是二进制数，这就使得它所处理的数字、字符、图像、声音等信息都是以 1 和 0 组成的二进制数的某种编码。

由于二进制在表达一个数字时，位数太长，不易识别，且容易出错，因此在书写计算机程序时，经常将它们写成对应的十六进制数或人们熟悉的十进制数表示。

在计算机内部可以根据实际情况的需要分别采用二进制数、十进制数和十六进制数。

表 2.1 给出了计算机中常用计数制的基数和数码以及进位借位关系。

表 2.1　计算机中常用计数制的基数和数码以及进位借位关系

计数制形式	基数	采用的数码	进位及借位关系
二进制	2	0、1	逢二进一、借一当二
十进制	10	0、1、2、3、4、5、6、7、8、9	逢十进一、借一当十
十六进制	16	0、1、2、3、4、5、6、7、8、9 A、B、C、D、E、F	逢十六进一、借一当十六

表 2.2 给出了计算机中常用计数制数据的对应关系。

表 2.2　计算机中常用计数制数据的对应关系

十进制数	二进制数	十六进制数	十进制数	二进制数	十六进制数
0	0000	0	8	1000	8
1	0001	1	9	1001	9
2	0010	2	10	1010	A
3	0011	3	11	1011	B
4	0100	4	12	1100	C

续表

十进制数	二进制数	十六进制数	十进制数	二进制数	十六进制数
5	0101	5	13	1101	D
6	0110	6	14	1110	E
7	0111	7	15	1111	F

3. 计数制的书写规则

为了区分各种计数制的数据，经常采用如下的方法进行书写表达。

（1）在数字后面加写相应的英文字母作为标识。

B（Binary）：表示二进制数，二进制数的 100 可写成 100B。

D（Decimal）：表示十进制数，十进制数的 100 可写成 100D，通常其后缀 D 可省略。

H（Hexadecimal）：表示十六进制数，十六进制数 100 可写成 100H。

（2）在括号外面加数字下标。

$(1011)_2$：表示二进制数的 1011。

$(2468)_{10}$：表示十进制数的 2468。

$(2DF2)_{16}$：表示十六进制数的 2DF2。

2.1.2 数制之间的转换

1. 十进制数转换为二进制数

一个十进制数通常由整数部分和小数部分组成，这两部分的转换规则是不相同的，在实际应用当中，整数部分与小数部分要分别进行转换。

（1）十进制整数转换为二进制整数。

十进制整数转换为二进制整数的方法：用基数 2 连续去除该十进制整数，直至商等于“0”为止，然后逆序排列余数，可得到与该十进制整数相应的二进制整数各位的系数值。

【例 2.1】将十进制整数 105 转换为二进制整数。

采用“除 2 倒取余”的方法，过程如下：

```
2 | 105
  2 | 52      余数为1
    2 | 26    余数为0
      2 | 13  余数为0
        2 | 6 余数为1
          2 | 3   余数为0
            2 | 1 余数为1
              0   余数为1
```

所以，$105=(1101001)_2$。

（2）十进制小数转化为二进制小数。

十进制小数转换为二进制小数的方法：用基数 2 连续去乘以该十进制小数，直至乘积的小数部分等于“0”，然后顺序排列每次乘积的整数部分，可得到与该十进制小数相应的二进制

小数各位的系数。

【例 2.2】将十进制小数 0.8125 转换为二进制小数。

采用“乘 2 顺取整”的方法，过程如下：

0.8125×2=1.625　　取整数位 1
0.625×2=1.25　　取整数位 1
0.25×2=0.5　　取整数位 0
0.5×2=1.0　　取整数位 1

所以，$0.8125=(0.1101)_2$。

如果出现乘积的小数部分一直不为“0”，则可以根据精度的要求截取一定的位数。

2. 十进制数转换为十六进制数

同理，十进制数转换为十六进制数时，可参照十进制数转换为二进制数的对应方法来处理。

（1）十进制整数转换为十六进制整数。

十进制整数转换为十六进制整数的方法：采用基数 16 连续去除该十进制整数，直至商等于“0”为止，然后逆序排列所得到的余数，可得到与该十进制整数相应的十六进制整数各位的系数。

【例 2.3】将十进制整数 2347 转换为十六进制整数。

采用“除 16 倒取余”的方法，过程如下：

16 | 2347
16 | 146　　余数为11（十六进制数为B）
16 | 9　　余数为2
0　　余数为9

所以，$2347=(92B)_{16}$。

（2）十进制小数转换为十六进制小数。

十进制小数转换为十六进制小数的方法：连续用基数 16 去乘以该十进制小数，直至乘积的小数部分等于“0”，然后顺序排列每次乘积的整数部分，可得到与该十进制小数相应的十六进制小数各位的系数。

【例 2.4】将十进制小数 0.5432 转换为十六进制小数。

采用“乘 16 顺取整”的方法，过程如下：

0.5432×16=8.6912　　取整数位 8
0.6912×16=11.0592　　取整数位 11（十六进制数为 B）
0.0592×16=0.9472　　取整数位 0
0.9472×16=15.1552　　取整数位 15（十六进制数为 F）

取数据的计算精度为小数点后 4 位数。

所以，$0.5432=(0.8B0F)_{16}$。

3. 二进制数、十六进制数转换为十进制数

二进制数、十六进制数转换为十进制数的时候，按照“位权展开求和”的方法可得到其结果。

（1）二进制数转换为十进制数。

二进制数转换为十进制数时，用其各位所对应的系数 1（系数为 0 时可以不必计算）来乘

以基数为 2 的相应位权，可得到与二进制数相应的十进制数。

【例 2.5】将二进制数$(1011001.101)_2$转换为十进制数。

按“位权展开求和”，过程如下：

$$(1011001.101)_2 = 1\times2^6+1\times2^4+1\times2^3+1\times2^0+1\times2^{-1}+1\times2^{-3}$$
$$=64+16+8+1+0.5+0.125$$
$$=89.625$$

（2）十六进制数转换为十进制数。

十六进制数转换为十进制数时，用其各位所对应的系数来乘以基数为 16 的相应位权，可得到与十六进制数相应的十进制数。

【例 2.6】将十六进制数$(2D7.A)_{16}$转换为十进制数。

按“位权展开求和”，过程如下：

$$(2D7.A)_{16} = 2\times16^2+13\times16^1+7\times16^0+10\times16^{-1}$$
$$=512+208+7+0.625$$
$$=727.625$$

4. 二进制数与十六进制数之间的转换

因为$16=2^4$，所以 1 位十六进制数相当于 4 位二进制数。从十六进制数转换为二进制数时，只要将每位十六进制数用 4 位二进制数表示即可；而从二进制数转换为十六进制数时，先要从小数点开始分别向左或向右将每 4 位二进制数分成一组，不足 4 位的要补 0，然后将每 4 位二进制数用一位十六进制数表示即可。

（1）十六进制数转换为二进制数的方法是“一分为四”。

【例 2.7】将十六进制数$(32A8.C69)_{16}$转换为二进制数。

将每一位十六进制数用 4 位二进制数表示，过程如下：

3	2	A	8.	C	6	9
↓	↓	↓	↓	↓	↓	↓
0011	0010	1001	1000.	1100	0110	1001

所以，$(31A8.C69)_{16}=(11001010011000.110001101001)_2$。

（2）二进制数转化为十六进制数的方法是“四合一”。

【例 2.8】将二进制数$(1110110010110.010101101)_2$转换为十六进制数。

从小数点开始分别向左或向右将每 4 位二进制数分成一组，过程如下：

0001	1101	1001	0110.	0101	0110	1000
↓	↓	↓	↓	↓	↓	↓
1	D	9	6.	5	3	8

所以，$(1110110010110.010101101)_2=(1D96.538)_8$。

为方便大家的理解和使用，我们将计数制之间的转换方法总结如表 2.3 所示。

表 2.3 计数制之间的转换方法

转换要求	转换方法
十进制整数转换为二进制和十六进制整数	分别用基数 2、16 连续除该十进制整数，至商等于“0”为止，然后逆序排列所得余数

续表

转换要求	转换方法
十进制小数转化为二进制和十六进制小数	连续用基数 2、16 乘该十进制小数，至乘积小数部分等于“0”，然后顺序排列所得乘积整数
二进制、十六进制数转换为十进制数	用各位所对应的系数和基数按“位权求和”的方法可得转换结果
二进制数转换为十六进制数	从小数点开始分别向左或向右将每 4 位二进制数分成 1 组，不足位数补 0，每组用 1 位十六进制数表示
十六进制数转换为二进制数	从小数点开始分别向左或向右将每位十六进制数用 4 位二进制数表示

2.2 计算机中数值数据的表示

2.2.1 基本概念

在计算机内部表示二进制数的方法通常称为数值编码。把一个数及其符号在机器中的表示加以数值化，这样的数称为机器数。机器数所代表的数称为该机器数的真值。

要全面完整地表示一个机器数，应考虑以下 3 个因素：

- 机器数的范围
- 机器数的符号
- 机器数中小数点的位置

1. 机器数的范围

通常机器数的范围由计算机的硬件决定。

当使用 8 位寄存器时，字长为 8 位，所以一个无符号整数的最大值是：

11111111B=255，此时机器数的范围是 0～255。

当使用 16 位寄存器时，字长为 16 位，所以一个无符号整数的最大值是：

1111111111111111B=FFFFH=65535，此时机器数的范围是 0～65535。

2. 机器数的符号

在算术运算中，数据是有正有负的，这类数据称为带符号数。

为了在计算机中正确地表示带符号数，通常规定每个字长的最高位为符号位，并用“0”表示正数，用“1”表示负数。

字长为 8 位二进制数时，D_7 为符号位；字长为 16 位二进制数时，D_{15} 为符号位。

例如，在一个 8 位字长的计算机中，带符号数据的格式如下：

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0							

正数

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1							

负数

其中，最高位 D_7 是符号位，其余 D_6～D_0 为数值位，这样把符号数字化并和数值位一起

编码的方法很好地解决了带符号数的表示及其计算问题。

这类编码方法常用的有原码、反码、补码 3 种。

3. 机器数中小数点的位置

在机器中，小数点的位置通常有定点和浮点两种表示方法。

（1）定点表示。

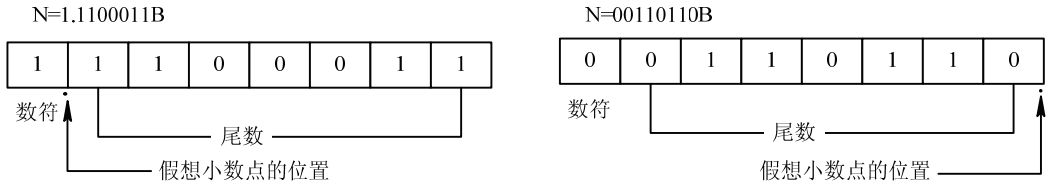
若规定计算机中的数其小数点位置固定不变时，该表示方法称为数的定点表示，该数称为“定点数”。

任何一个二进制数 N 都可以表示为：$N=\pm 2^{\pm P}\times S$。

式中 S 为数 N 的尾数，表示该数的全部有效数字；2 为计数制底数，2 前面的±号是尾数符号；P 为数 N 的阶码，指明小数点实际位置，2 的右上方的±号是阶码的符号。若阶码 P 固定不变，则小数点位置是固定的。

定点数有两种约定：

- 取阶码 P=0，把小数点固定在尾数的最高位之前，称为定点小数，格式如图 2-1（a）所示。
- 取阶码 P=n（n 为二进制尾数的位数），把小数点约定在尾数最末位之后，称为定点整数，格式如图 2-1（b）所示。



（a）定点小数表示　　（b）定点整数表示

图 2-1　定点数的表示方法

计算机中小数点的位置是假想位置，厂家在机器设计时将数的表示形式约定好，计算机中各种部件及运算线路均按约定的形式进行设计。机器数的字长确定后，其数值的表示范围即可确定。

（2）浮点表示。

将十进制数用指数形式表示如下：

$$562.98=0.56298\times 10^{3}-0.000034=-0.34\times 10^{-4}$$

可以看出：在原数字中无论小数点前后各有几位数，它们都可以用一个纯小数与 10 的整数次幂的乘积来表示，这就是浮点数的表示方法。

对于一个二进制数，当其阶码 P 不固定时，数的小数点实际位置将根据阶码值相对浮动，该数称为“浮点数”。

浮点数在机器中的编码排列如下：

阶符	阶码 P	尾符	尾数 S

阶码 P 表示数的实际小数点相对机器中约定小数点位置的浮动方向，如阶符为负，则实际小数点在约定小数点的左边，反之在右边，其位置由阶码值来确定，而尾数符号则代表了浮

点数的符号。

浮点数中的阶符和阶码指明了小数点的位置，小数点随着阶码 P 的符号和大小而浮动。可见，浮点数可以表示的数值范围要比定点数大，这也是它的主要可取之处。

2.2.2 带符号数的原码、反码、补码表示

1. 原码

正数的符号位为“0”，负数的符号位为“1”，其他位按照一般的方法来表示数的绝对值，用这样的表示方法得到的就是该数的原码。

【例 2.9】当机器字长为 8 位二进制数时：

X=＋1011011　　$[X]_{原码}$=01011011

Y=＋1011011　　$[Y]_{原码}$=11011011

$[+1]_{原码}$=00000001　　$[-1]_{原码}$=10000001

$[+127]_{原码}$=01111111　　$[-127]_{原码}$=11111111

在二进制数的原码表示中，“0”的表示有正负之分：

$[+0]_{原码}$=00000000　　$[-0]_{原码}$=10000000

原码表示的整数范围是$-(2^{n-1}-1)$～$+(2^{n-1}-1)$，其中 n 为机器字长。

则：8 位二进制原码表示的整数范围是－127～＋127；

16 位二进制原码表示的整数范围是－32767～＋32767。

两个符号相异但绝对值相同的数的原码，除了符号位以外，其他位的表示都是一样的。数的原码表示简单直观，而且与其真值转换方便。

但是，如果有两个符号相异的数要进行相加或两个同符号数相减，就要做减法运算。做减法运算会产生借位的问题，很不方便。为了将加法运算和减法运算统一起来，以加快运算速度，就引进了数的反码和补码表示。

2. 反码

对于一个带符号的数来说，正数的反码与其原码相同，负数的反码为其原码除符号位以外的各位按位取反。

【例 2.10】当机器字长为 8 位二进制数时：

X=＋1011011　　$[X]_{原码}$=01011011　　$[X]_{反码}$=01011011

Y=－1011011　　$[Y]_{原码}$=11011011　　$[Y]_{反码}$=10100100

$[+1]_{反码}$=00000001　　$[-1]_{反码}$=11111110

$[+127]_{反码}$=01111111　　$[-127]_{反码}$=10000000

可以看出，负数的反码与负数的原码有很大的区别，反码通常用作求补码过程中的中间形式。

反码表示的整数范围与原码相同。

数据“0”在二进制数的反码表示中有以下形式：

$[+0]_{反码}$=$[+0]_{原码}$=00000000　　$[-0]_{反码}$=11111111

3. 补码

正数的补码与其原码相同，负数的补码为其反码在最低位加 1。

【例 2.11】X=＋1011011　　$[X]_{原码}$=01011011　　$[X]_{补码}$=01011011

Y=－1011011　[Y]原码=11011011　[Y]反码=10100100　[Y]补码=10100101

[＋1]补码=00000001　[－1]补码=11111111

[＋127]补码=01111111　[－127]补码=10000001

在二进制数的补码表示中，“0”的表示是唯一的。

即：[＋0]补码=[－0]补码=00000000

采用补码的目的是为使符号位作为数参加运算，解决将减法转换为加法运算的问题，并简化计算机控制线路，提高运算速度。在很多计算机系统中都采用补码来表示带符号的数。

由于计算机中存储数据的字节数是有限制的，所以能存储的带符号数也有一定的范围。

补码表示的整数范围是-2^{n-1}～$+(2^{n-1}-1)$，其中 n 为机器字长。

则：8 位二进制补码表示的整数范围是－128～＋127；

16位二进制补码表示的整数范围是－32768～＋32767。

当运算结果超出这个范围时，就不能正确表示数了，此时称为溢出。

对于 8 位字长的二进制数，其原码、反码、补码的对应关系如表 2.4 所示。

表 2.4　8 位二进制数原码、反码、补码的对应关系

二进制数	无符号数	带符号数		
		原码	反码	补码
00000000	0	＋0	＋0	＋0
00000001	1	＋1	＋1	＋1
…	…	…	…	…
01111111	127	＋127	＋127	＋127
10000000	128	－0	－127	－128
…	…	…	…	…
11111110	254	－126	－1	－2
11111111	255	－127	－0	－1

4. 补码与真值之间的转换

已知某机器数的真值可通过补码的定义将其转换为补码。已知正数的补码，其真值等于补码本身；已知负数的补码，可除符号位以外将补码的有效值按位求反后在末位加 1，即可得该负数补码对应的真值。

【例 2.12】 已知 X＝－43，采用 8 位二进制表示出 X 的原码、反码和补码。

解：给定数据为负数，将其转换为二进制数为 X＝－101011B

按照相关转换方法可得 X 的 8 位原码、反码和补码表示：

[X]原码=10101011B

[X]反码=11010100B

[X]补码=[X]反码＋1

=11010101B

【例 2.13】给定[X]补码=01011001B，求其真值 X。

解：由于给定[X]补码的符号位是“0”，代表该数是正数，则其真值为：

X=＋1011001B

$=+(1\times2^6+1\times2^4+1\times2^3+1\times2^0)$

=＋(64＋16＋8＋1)

=＋89

【例 2.14】给定$[X]_{补码}$=11011001B，求其真值 X。

解：由于给定$[X]_{补码}$的符号位是“1”，代表该数是负数，则其真值为：

$X=-([1011001]_{求反}+1)B$

=－(0100110＋1)B

=－0100111B

$=-(1\times2^5+1\times2^2+1\times2^1+1\times2^0)$

=－(32＋4＋2＋1)

=－39

2.2.3 带符号数的加减运算与数据溢出判断

1. 带符号数的加减运算

由于补码运算比较简单，而且负数用相应补码表示后，可以将减法运算转换为加法运算。所以，一般计算机中只设置加法器，减法运算是通过适当求补处理后再进行相加来实现。

给定两个带符号数 X、Y：

进行补码加法运算时：$[X+Y]_{补码}=[X]_{补码}+[Y]_{补码}$

进行补码减法运算时：$[X-Y]_{补码}=[X]_{补码}-[Y]_{补码}=[X]_{补码}+[-Y]_{补码}$

【例 2.15】已知 X=－110011B，Y=＋10101B，求 X＋Y=?

解：给定数据中 X 为负数，Y 为正数，根据补码运算规则有：

$[X]_{原码}$=10110011B，$[X]_{补码}=[X]_{反码}+1$=11001101B

$[Y]_{补码}=[Y]_{原码}$=00010101B

所以，$[X+Y]_{补码}=[X]_{补码}+[Y]_{补码}$

=11001101B＋00010101B

=11100010B

可见，X＋Y 的补码为 11101010B，其中符号位为“1”表示该题的结果为负数，即 X＋Y=－30，这是由于 X=－51，Y=＋21，两者相加结果为负数，这与十进制的运算结果相同。

2. 数据溢出判断

运算后得到的结果若超过计算机所能表示的数值范围称为数据溢出。如 8 位带符号数取值范围是－128～＋127，当 X±Y＜－128 或 X±Y＞127 时产生溢出，将导致错误结果。

可通过参加运算的两数和运算结果的符号位来判断是否产生溢出，如果两个正数相加得到的结果为负数或者两个负数相加得到的结果为正数，则产生溢出。

【例 2.16】 已知两个带符号数 X=01001001B，Y=01101010B，用补码运算求 X＋Y 的结果，并判断其是否会产生溢出。

解：给定为两个正数，按照补码运算规则可得：

$[X]_{补码}$=01001001B，$[Y]_{补码}$=01010101B

$[X+Y]_{补码}=[X]_{补码}+[Y]_{补码}$=01001001B＋01101010B=10110011B

运算结果 10110011B 的符号位为 1，表示 X＋Y 的值为负数。两个正数相加得到负数显然是错误的，出错的原因是由于 X＋Y=73＋106=179>127，超出 8 位带符号数的取值范围，产生溢出。

当两数异号时，相加的结果只会变小，所以不会产生溢出。运算结果产生溢出时，计算机会自动进行判断，为使用户知道带符号数算术运算的结果是否产生了溢出，专门在 CPU 的标志寄存器中设置了溢出标志 OF。当 OF=“1”时表示运算结果产生了溢出，OF=“0”时表示运算结果未溢出。

2.3 字符编码

计算机除了用于数值计算之外，还要进行大量的文字信息处理，也就是要对表达各种文字信息的符号进行加工。例如，计算机和外设的键盘、显示器、打印机之间的通信都是采用字符方式输入/输出的。

字符在机器里也必须用二进制数来表示，但是这种二进制数是按照特定规则编码表示的。计算机为了识别和区分这些符号，采用了以下方法：

- 使用由若干位组成的二进制数去代表一个符号。
- 一个二进制数只能与一个符号唯一对应，即符号集内所有的二进制数不能相同。

这样，二进制数的位数自然取决于符号集的规模。

例如：128 个符号的符号集，需要 7 位二进制数；

256 个符号的符号集，需要 8 位的二进制数。

这就是所谓的字符编码，由此可以看出：计算机解决任何问题都是建立在编码技术上的。目前最通用的两种字符编码分别是美国信息交换标准代码（ASCII 码）和二—十进制编码（BCD 码）。

2.3.1 美国信息交换标准代码（ASCII 码）

ASCII（American Standard Code for Information Interchange）码是美国信息交换标准代码的简称，用于给西文字符编码，包括英文字母的大小写、数字、专用字符、控制字符等。

这种编码由 7 位二进制数组合而成，可以表示 128 种字符，目前在国际上广泛流行。

ASCII 码的编码内容如表 2.5 所示。

表 2.5　7 位 ASCII 码编码表

低 4 位代码		高 3 位代码							
		0	1	2	3	4	5	6	7
		000	001	010	011	100	101	110	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

续表

低 4 位代码		高 3 位代码							
		0	1	2	3	4	5	6	7
		000	001	010	011	100	101	110	111
4	0100	EOT	DC4	$	4	D	T	d	t
5	0101	ENQ	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
9	1001	HT	EM	）	9	I	Y	i	y
A	1010	LF	SUB	*	：	J	Z	j	z
B	1011	VT	ESC	+	；	K	[	k	{
C	1100	FF	FS	，	＜	L	\	l	|
D	1101	CR	GS	—	=	M	]	m	}
E	1110	SO	RS	.	＞	N	↑	n	～
F	1111	SI	US	/	?	O	←	o	DEL

ASCII 码的特点分析如下：

（1）每个字符的 7 位以高 3 位和低 4 位二进制数组合而成 ASCII 码，采用十六进制数来表示。

如换行“LF”的 ASCII 码是 0AH，回车“CR”的 ASCII 码是 0DH。

数码 0～9 的 ASCII 码是 30H～39H（可见去掉高 4 位，即减去 30H 就是 BCD 码的表示）。

大写字母 A～Z 的 ASCII 码是 41H～5AH，小写字母 a～z 的 ASCII 码是 61H～7AH（可见大小写字母之间 ASCII 码值相差 20H，两者之间的转换容易实现）。

（2）128 个字符的功能可分为 94 个信息码和 34 个功能码。

信息码包括 10 个阿拉伯数字、52 个英文大小写字母、32 个专用符号等，可供书写程序和描述命令之用，能够显示和打印出来。

功能码在计算机系统中起各种控制作用，可提供传输控制、格式控制、设备控制、信息分隔控制及其他控制等，这些控制符只表示某种特定操作，不能显示和打印。

功能码的含义如表 2.6 所示。

表 2.6　ASCII 编码表中功能码的含义

字符	操作功能	字符	操作功能	字符	操作功能
NUL	空	FF	走纸控制	CAN	作废
SOH	标题开始	CR	回车	EM	纸尽
STX	正文结束	SO	移位输出	SUB	减

续表

字符	操作功能	字符	操作功能	字符	操作功能
ETX	本文结束	SI	移位输入	ESC	换码
EOT	传输结束	DLE	数据链换码	FS	文字分隔符
ENQ	询问	DC1	设备控制 1	GS	组分隔符
ACK	承认	DC2	设备控制 2	RS	记录分隔符
BEL	报警符	DC3	设备控制 3	US	单元分隔符
BS	退格	DC4	设备控制 4	SP	空格
HT	横向列表	NAK	否定	DEL	删除
LF	换行	SYN	空转同步		
VT	垂直制表	ETB	信息组传输结束		

34 个功能码可分成以下 5 种处理功能：

- 传输控制类字符，如 SOH、STX、ETX 等。
- 格式控制类字符，如 BS、LF、CR 等。
- 设备控制类字符，如 DC1、DC2、DC3 等。
- 信息分隔类控制字符，如 FS、RS、US 等。
- 其他控制字符，如 NUL、BEL、ESC 等。

（3）由于微型计算机基本存储单元是一个字节（byte），即 8 位二进制数，表达 ASCII 码时也采用 8 位，最高位 D_7 通常作为“0”。进行数据通信时，最高位 D_7 通常作为奇偶校验位，用来检验代码在存储和传送过程中是否发生错误。

奇校验含义：包括校验位在内的 8 位二进制码中所有“1”的个数为奇数。如字符“A”的 ASCII 码是 41H（1000001B），加奇校验位时“A”的 ASCII 码为 C1H（11000001B）。

偶校验含义：包括校验位在内的 8 位二进制码中所有“1”的个数为偶数。如字符“A”加偶校验时 ASCII 码依然是 41H（01000001B）。

随着信息技术的发展，为了扩大计算机处理信息的范围，IBM 公司又将 ASCII 码的位数增加了一位，由原来的 7 位变为用 8 位二进制数构成一个字符编码，共有 256 个符号。扩展后的 ASCII 码除原有的 128 个字符外，又增加了一些常用的科学符号和表格线条等。

2.3.2 二—十进制编码——BCD 码

将一个十进制数在计算机中采用二进制编码来表示，称为 BCD（Binary-Coded Decimal）码，即“二—十进制编码”。

常用的 BCD 码是 8421-BCD 编码，采用 4 位二进制数来表示一位十进制数，自左至右每一个二进制位对应的位权是 8、4、2、1。

应该指出的是，4 位二进制数有 0000～1111 共 16 种状态，而十进制数 0～9 只取 0000～1001 的 10 种状态，其余 6 种不用。

8421-BCD 编码如表 2.7 所示。

表 2.7　8421-BCD 编码表

十进制数	8421-BCD 编码	十进制数	8421-BCD 编码
0	0000	8	1000
1	0001	9	1001
2	0010	10	0001　0000
3	0011	11	0001　0001
4	0100	12	0001　0010
5	0101	13	0001　0011
6	0110	14	0001　0100
7	0111	15	0001　0101

通常，BCD 码有两种形式，即压缩 BCD 码和非压缩 BCD 码。

1. 压缩 BCD 码

表 2.7 所示的 BCD 码为压缩 BCD 码（或称组合 BCD 码），其特点是采用 4 位二进制数来表示一位十进制数，即一个字节表示两位十进制数。如十进制数 57 的压缩 BCD 码为 01010111B。

压缩 BCD 码的每一位数采用 4 位二进制数来表示，即一个字节表示两位十进制数。

例如，二进制数 10001001B，采用压缩 BCD 码表示为十进制数 89。

2. 非压缩 BCD 码

非压缩 BCD 码（或称非组合 BCD 码）表示特点是采用 8 位二进制数来表示一位十进制数，即一个字节表示 1 位十进制数，而且只用每个字节的低 4 位来表示 0～9，高 4 位设定为 0。例如，十进制数 89D，采用非压缩 BCD 码表示为二进制数是 00001000 00001001B。

BCD 码与十进制数之间转换很容易实现，如压缩 BCD 码为 1001 0101 0011.0010 0111，其十进制数值为 953.27。

BCD 码可直观地表达十进制数，也容易实现与 ASCII 码的相互转换，便于数据的输入和输出。

计算机中的各种信息均表示为二进制数据。但是二进制数据书写比较冗长、容易出错，实际使用中，人们多采用十进制和十六进制来表示数据。各类数制之间的相互转换有特定的规律可循。

计算机内部将一个数及其符号数值化表示的方法称为机器数。表示一个完整的机器数需要考虑数的范围、符号和小数点的位置。带符号数在计算机中有原码、反码和补码三种表示方法。引入补码的目的是为了使数字化后的符号位能作为数参加运算，并将减法运算转换为加法运算，简化了机器数的运算。如果机器数中的小数点位置固定不变称为“定点数”，小数点位置可以浮动时称为“浮点数”。计算机中参加运算的数若超过计算机所能表示的数值范围称为溢出。可根据两数的符号位或运算结果标志位来判断结果是否产生溢出。

描述特定字符和信息等也需要用二进制进行编码，目前普遍采用的是美国信息交换标准代码（ASCII 码）和二—十进制编码（BCD 码）。ASCII 码用一个字节中的 7 位对字符进行编码，可表示 128 种字符，最高位是奇偶校验位，用以判别数码传送是否正确。BCD 码专门解决用二进制数表示十进制数的问题。

本章着重介绍了计算机中数据的表示方法，分析了二、十、十六进制数的相关概念及各类数制间相互转换的方法、无符号数和带符号数的机器内部表示、字符编码等。通过学习，要掌握计算机内部的信息处理方法和特点，熟悉各类数制之间的相互转换，理解无符号数和带符号数的表示方法，掌握字符的 ASCII 码和 BCD 码及其应用。

一、选择题

1．当机器数采用（　　）方式时，零的表示形式是唯一的。

A．原码　　B．补码　　C．反码　　D．真值

2．带符号数在计算机中通常采用（　　）来表示。

A．原码　　B．反码　　C．补码　　D．BCD 码

3．在 8 位二进制数中，采用补码表示时其数的真值范围是（　　）。

A．－127～＋127　　B．－127～＋128

C．－128～＋127　　D．－128～＋128

4．已知某数为－128，其机器数为 10000000B，则其机内采用的是（　　）表示。

A．原码　　B．反码　　C．补码　　D．真值

5．若十六进制数为 AC.BH，则其十进制数为（　　）。

A．254.54　　B．2763　　C．172.6875　　D．172.625

6．计算机内数据产生溢出是指（　　）。

A．运算结果为无穷大

B．运算结果超出一个字数据所能表示的范围

C．运算结果超出该指令指定存储单元的数据范围

D．运算结果超出计算机内存单元所能表示的数据范围

7．大写字母“B”的 ASCII 码是（　　）。

A．41H　　B．42H　　C．61H　　D．62H

8．某数在计算机中用压缩 BCD 码表示为 1001 0011，其真值为（　　）。

A．10010011B　　B．93H　　C．93　　D．147

二、填空题

1．任何进位计数制都包含基数和位权两个基本要素，二进制的基数为_____，其中第 i 位的位权为_____。

2．计算机中的数有_____和_____两种表示方法，前者的特点是_____，后者的特点是_____。

3．计算机中带符号的数在运算处理时通常采用_____表示，其好处在于_____。

4．计算机中参加运算的数及运算结果都应在_____范围内，如参加运算的数及运算结果_____，称为数据溢出。

5．已知某数为 61H，若为无符号数其真值为_____；若为带符号数其真值为_____；若为 ASCII 码其值代表_____；若为 BCD 码其值代表_____。

6．ASCII 码可以表示_____种字符，其中起控制作用的称为_____；供书写程序和描述命令使用的称为_____。

7．BCD 码是一种_____表示方法，按照表示形式可分为_____和_____两种表现形式。

三、判断题

（　　）1．在计算机中，数据的表示范围不受计算机字长的限制。

（　　）2．计算机中带符号数采用补码表示的目的是为了简化机器数的运算。

（　　）3．计算机在运算中产生了数据溢出，其原因是运算过程中最高位产生了进位。

（　　）4．“0”的原码和反码各有不同表示，而“0”的补码表示是唯一的。

（　　）5．计算机键盘输入的各类符号在计算机内部均表示为 ASCII 码。

四、数制转换题

1．将下列十进制数分别转化为二进制数、十六进制数和压缩 BCD 码。

（1）26　　（2）79.15　　（3）125　　（4）228

2．将下列二进制数或十六进制数分别转化为十进制数。

（1）10110110B　　（2）10100101B　　（3）A8H　　（4）B5.62H

3．写出下列带符号十进制数的原码、反码、补码表示（采用 8 位二进制数）。

（1）＋26　　（2）＋75　　（3）－38　　（4）－119

4．已知下列补码求其真值。

（1）97H　　（2）3FH　　（3）8B4CH　　（4）3C2AH

5．按照字符所对应的 ASCII 码值，查表写出下列字符的 ASCII 码。

K、R、good、*、$、ESC、LF、CR