21 世纪 CENTURY

高等学校计算机类专业"十一五"规划教材

单片机原理及接口技术

（第二版）

余锡存　曹国华　编著

西安电子科技大学出版社
http://www.xduph.com

面向21世纪高等学校计算机类专业"十一五"规划教材

单片机原理及接口技术

(第二版)

余锡存　曹国华　编著

西安电子科技大学出版社

2007

内 容 简 介

本书首先介绍了微型计算机的基础知识，并以MCS－51系列单片机为核心，系统介绍了单片机的基本结构、指令系统、汇编语言程序设计、系统扩展与接口技术、应用系统设计与开发以及抗干扰技术，最后简要介绍了具有51内核的8位单片机类型与性能。本书配有例题、习题与思考题，便于课堂教学与自学。

本书是高等学校电子类及计算机应用专业的教材，同时也可供非计算机专业、高等职业教育、自学考试和从事微机应用的人员使用。全书内容深入浅出、通俗易懂、注重工程应用。

★ 本书配有电子教案，需要者可来函向我社索取。

图书在版编目(CIP)数据

单片机原理及接口技术/余锡存，曹国华编著. —2版.
—西安：西安电子科技大学出版社，2007.12
面向21世纪高等学校计算机类专业"十一五"规划教材
ISBN 978－7－5606－0870－9

Ⅰ. 单…　Ⅱ. ①余…　②曹…　Ⅲ. ①单片微型计算机—基础理论—高等学校—教材　②单片微型计算机—接口—高等学校—教材　Ⅳ. TP368.1

中国版本图书馆CIP数据核字(2007)第151956号

策　　划　马乐惠
责任编辑　梁家新　马乐惠
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)88242885　88201467　　邮　编　710071
http://www.xduph.com　　E-mail：xdupfxb@pub.xaonline.com
经　　销　新华书店
印刷单位　西安文化彩印厂
版　　次　2007年10月第2版　2007年12月第13次印刷
开　　本　787毫米×1092毫米　1/16　印张14.5
字　　数　342千字
印　　数　60 001～64 000册
定　　价　19.00元
ISBN 978－7－5606－0870－9/TP・0455

XDUP　1141022—13

第二版前言

当代单片机的发展可谓日新月异，其性能不断提高，功能日益增多，应用技术逐渐成熟，应用范围也越来越广。特别是在工业测控、智能仪器和日用家电等领域，单片机已作为必不可少的核心部件，学习与掌握单片机应用技术已成为工程技术人员必须掌握的技术之一。

为适应单片机原理及应用课程教学内容的不断变化，作为高校的单片机教材也应随之不断补充与修订。作者于 2000 年 7 月编写并出版的《单片机原理及接口技术》一书自问世以来，受到了广大读者的厚爱，已多次重印。应读者的要求并根据读者的反馈意见，本人对该书进行了修订。

本次修订，按照求新务实、便于教学的原则，对原书的部分章节进行了补充与订正。全书由南京师范大学电气与自动化工程学院的余锡存与和曹国华共同编写，其中第 1、2、7、8、9 章及附录由余锡存编写，第 3、4、5、6、10 章由曹国华编写，全书由余锡存修订与统稿。

在本书撰写和修订的过程中，参考了大量的国内外文献和教材，在此谨向作者表示衷心的感谢。

编　者

2007 年 9 月

第一版前言

目前，单片机已广泛应用到国民经济建设和日常生活的许多领域，成为测控技术现代化必不可少的重要工具。根据高等院校教学要求，我们总结了多年的教学和实践经验，编写了本书。

本书是高等学校电子类及计算机应用专业的教材，同时也可供非计算机专业、高等职业教育、自学考试和从事微机应用的人员使用。本书力求深入浅出、通俗易懂、注重工程应用。

全书共分10章。首先介绍了微型计算机的基础知识；并以MCS－51系列单片机为基础，系统介绍了单片机的基本结构、指令系统、汇编语言程序设计、系统扩展与接口技术、应用系统设计与开发，以及抗干扰技术，最后还简要介绍了其它系列8位单片机的类型与性能，主要有：Atmel公司的AT89C、Intel公司的8XC51、Philips公司的8XC552等系列。本书配有例题、习题与思考题，便于课堂教学与自学，使读者通过本书的学习，为今后的工作打下坚实的基础。

本书第1、2、7、8、9章由余锡存同志编写，第3、4、5、6、10章由曹国华同志编写，本书由上海理工大学的唐俊杰老师主审。

由于编者水平有限，时间仓促，加之单片机技术日新月异，书中存在的不当之处，敬请读者指正。

编　者

1999年11月于南京

目　录

第 1 章

微型计算机基础

1.1　计算机中的数制及相互转换

在日常生活中人们最熟悉的是十进制数，但在计算机中，采用二进制数“0”和“1”可以很方便地表示机内的数据与信息。在编程时，为了便于阅读和书写，人们还常用八进制数或十六进制数来表示二进制数。

1.1.1　进位计数制

按进位原则进行计数的方法，称为进位计数制。十进制数有两个主要特点：

(1) 有 10 个不同的数字符号：0、1、2、…、9；

(2) 低位向高位进位的规律是“逢十进一”。

因此，同一个数字符号在不同的数位所代表的数值是不同的。如 555.5 中 4 个 5 分别代表 500、50、5 和 0.5，这个数可以写成

$$555.5=5\times10^{2}+5\times10^{1}+5\times10^{0}+5\times10^{-1}$$

式中的 10 称为十进制的基数，10^2、10^1、10^0、10^{-1}称为各数位的权。

任意一个十进制数 N 都可以表示成按权展开的多项式：

$$\begin{aligned}N&=d_{n-1}\times10^{n-1}+d_{n-2}\times10^{n-2}+\cdots+d_0\times10^0+d_{-1}\times10^{-1}+\cdots+d_{-m}\times10^{-m}\\&=\sum_{i=-m}^{n-1}d_i\times10^i\end{aligned}$$

其中，d_i 是 0～9 共 10 个数字中的任意一个，m 是小数点右边的位数，n 是小数点左边的位数，i 是数位的序数。例如，543.21 可表示为

$$543.21=5\times10^{2}+4\times10^{1}+3\times10^{0}+2\times10^{-1}+1\times10^{-2}$$

一般而言，对于用 R 进制表示的数 N，可以按权展开为

$$\begin{aligned}N&=a_{n-1}\times R^{n-1}+a_{n-2}\times R^{n-2}+\cdots+a_0\times R^0+a_{-1}\times R^{-1}+\cdots+a_{-m}\times R^{-m}\\&=\sum_{i=-m}^{n-1}a_i\times R^i\end{aligned}$$

式中，a_i是 0、1、…、R－1 中的任一个，m、n 是正整数，R 是基数。在 R 进制中，每个数

字所表示的值是该数字与它相应的权 R^i 的乘积，计数原则是“逢 R 进一”。

1. 二进制数

当 R=2 时，称为二进位计数制，简称二进制。在二进制数中，只有两个不同数码：0 和 1，进位规律为“逢二进一”。任何一个数 N，可用二进制表示为

$$N = a_{n-1}\times 2^{n-1} + a_{n-2}\times 2^{n-2} + \cdots + a_0\times 2^0 + a_{-1}\times 2^{-1} + \cdots + a_{-m}\times 2^{-m}$$
$$= \sum_{i=-m}^{n-1} a_i \times 2^i$$

例如，二进制数 1011.01 可表示为

$$(1011.01)_2 = 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 1\times 2^0 + 0\times 2^{-1} + 1\times 2^{-2}$$

2. 八进制数

当 R=8 时，称为八进制。在八进制中，有 0、1、2、…、7 共 8 个不同的数码，采用“逢八进一”的原则进行计数。如 $(503)_8$ 可表示为

$$(503)_8 = 5\times 8^2 + 0\times 8^1 + 3\times 8^0$$

3. 十六进制

当 R=16 时，称为十六进制。在十六进制中，有 0、1、2、…、9、A、B、C、D、E、F 共 16 个不同的数码，进位方法是“逢十六进一”。

例如，$(3A8.0D)_{16}$ 可表示为

$$(3A8.0D)_{16} = 3\times 16^2 + 10\times 16^1 + 8\times 16^0 + 0\times 16^{-1} + 13\times 16^{-2}$$

表 1.1 列出了二、八、十、十六进制数之间的对应关系。

表 1.1 各种进位制的对应关系

十进制	二进制	八进制	十六进制	十进制	二进制	八进制	十六进制
0	0	0	0	9	1001	11	9
1	1	1	1	10	1010	12	A
2	10	2	2	11	1011	13	B
3	11	3	3	12	1100	14	C
4	100	4	4	13	1101	15	D
5	101	5	5	14	1110	16	E
6	110	6	6	15	1111	17	F
7	111	7	7	16	10000	20	10
8	1000	10	8				

1.1.2 不同进制间的相互转换

1. 二、八、十六进制转换成十进制

根据各进制的定义表示方式，按权展开相加，即可将二进制数、八进制数、十六进制数转换成十进制数。

【例 1】 将$(10.101)_2$，$(46.12)_8$，$(2D.A4)_{16}$转换为十进制数。

$$(10.101)_2=1\times2^1+0\times2^0+1\times2^{-1}+0\times2^{-2}+1\times2^{-3}=2.625$$

$$(46.12)_8=4\times8^1+6\times8^0+1\times8^{-1}+2\times8^{-2}=38.156\,25$$

$$(2D.A4)_{16}=2\times16^1+13\times16^0+10\times16^{-1}+4\times16^{-2}=45.640\,62$$

2. 十进制数转换成二、八、十六进制数

任意十进制数 N 转换成 R 进制数，需将整数部分和小数部分分开，采用不同方法分别进行转换，然后用小数点将这两部分连接起来。

(1) 整数部分：除基取余法。

分别用基数 R 不断地去除 N 的整数，直到商为零为止，每次所得的余数依次排列即为相应进制的数码。最初得到的为最低有效数字，最后得到的为最高有效数字。

【例 2】 将$(168)_{10}$转换成二、八、十六进制数。

		余数	
2	168		
2	84	… 0	↑最低位
2	42	… 0	
2	21	… 0	
2	10	… 1	
2	5	… 0	
2	2	… 1	
2	1	… 0	
	0	… 1	最高位

$(168)_{10}=(10101000)_2$

		余数
8	168	
8	21	… 0
8	2	… 5
	0	… 2

$(168)_{10}=(250)_8$

		余数
16	168	
16	10	… 8
	0	… A

$(168)_{10}=(A8)_{16}$

(2) 小数部分：乘基取整法。

分别用基数 R(R=2，8 或 16)不断地去乘 N 的小数，直到积的小数部分为零(或直到所要求的位数)为止，每次乘得的整数依次排列即为相应进制的数码。最初得到的为最高有效数字，最后得到的为最低有效数字。

【例 3】 将$(0.645)_{10}$转换成二、八、十六进制数(用小数点后五位表示)。

整数	0.645
	× 2
1 …	1.290
	0.29
	× 2
0 …	0.58
	0.58
	× 2
1 …	1.16
	0.16
	× 2
0 …	0.32
	× 2
0 …	0.64

整数	0.645
	× 8
5 …	5.160
	0.16
	× 8
1 …	1.28
	0.28
	× 8
2 …	2.24
	0.24
	× 8
1 …	1.92
	0.92
	× 8
7 …	7.36

整数	0.645
	× 16
A …	10.320
	0.32
	× 16
5 …	5.12
	0.12
	× 16
1 …	1.92
	0.92
	× 16
E …	14.72
	0.72
	× 16
B …	11.52

故：$(0.645)_{10}=(0.10100)_2=(0.51217)_8=(0.A51EB)_{16}$

【例 4】 将$(168.645)_{10}$转换成二、八、十六进制数。

根据例 2、例 3 可得

$$(168.645)_{10}=(10101000.10100)_2=(250.51217)_8=(A8.A51EB)_{16}$$

3. 二进制与八进制之间的相互转换

由于 $2^3=8$，故可采用“合三为一”的原则，即从小数点开始分别向左、右两边各以 3 位为一组进行二—八换算：若不足 3 位的以 0 补足，便可将二进制数转换为八进制数。反之，采用“一分为三”的原则，每位八进制数用三位二进制数表示，就可将八进制数转换为二进制数。

【例 5】 将$(101011.01101)_2$ 转换为八进制数。

$$\begin{array}{ccccc} 101 & 011 & . & 011 & 010 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 5 & 3 & . & 3 & 2 \end{array}$$

即 $(101011.01101)_2=(53.32)_8$

【例 6】 将$(123.45)_8$ 转换成二进制数。

$$\begin{array}{cccccc} 1 & 2 & 3 & . & 4 & 5 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 001 & 010 & 011 & . & 100 & 101 \end{array}$$

即 $(123.45)_8=(1010011.100101)$

4. 二进制数与十六进制数之间的转换

由于 $2^4=16$，故可采用“合四为一”的原则，从小数点开始分别向左、右两边各以 4 位为一组进行二—十六换算；若不足 4 位以 0 补足，即可将二进制数转换为十六进制数。反之，采用“一分为四”的原则，每位十六进制数用 4 位二进制数表示，便可将十六进制数转换为二进制数。

【例 7】 将$(110101.011)_2$ 转换为十六进制数。

$$\begin{array}{cccc} 0011 & 0101 & . & 0110 \\ \downarrow & \downarrow & \downarrow & \downarrow \\ 3 & 5 & . & 6 \end{array}$$

即 $(110101.011)_2=(35.6)_{16}$

【例 8】 将$(4A5B.6C)_{16}$转换为二进制数。

$$\begin{array}{ccccccc} 4 & A & 5 & B & . & 6 & C \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 0100 & 1010 & 0101 & 1011 & . & 0110 & 1100 \end{array}$$

即 $(4A5B.6C)_{16}=(100101001011011.011011)_2$

在程序设计中，为了区分不同进制的数，通常在数的后面加字母作为标注。其中，字母 B(Binary)表示二进制数；字母 Q(Octal，用字母 Q 而不用 O 主要是为区别数字 0)表示八进制数；字母 D(Decimal)或不加字母表示十进制数；字母 H(Hexadecimal)表示十六进制数。如 1101B、57Q、512D、3AH 等。

1.2　二进制数的运算

1.2.1　二进制数的算术运算

二进制数只有 0 和 1 两个数字，其算术运算较为简单，加、减法遵循“逢二进一”、“借一当二”的原则。

1. 加法运算

规则：0+0=0；0+1=1；1+0=1；1+1=10(有进位)

【例 1】 求 1001B+1011B。

```
被加数    1001
  加数+   1011
──────────────
  进位   10010
    和   10100
```

即　1001B+1011B=10100B

2. 减法运算

规则：0－0=0；1－1=0；1－0=1；0－1=1(有借位)

【例 2】 求 1100B－111B。

```
被减数    1100
  减数－   111
──────────────
  借位    0110
    差    0101
```

即　1100B－111B=101B

3. 乘法运算

规则：0×0=0；0×1=1×0=0；1×1=1

【例 3】 求 1011B×1101B。

```
被乘数       1011
  乘数      ×1101
─────────────────
             1011
            0000
           1011
       +  1011
─────────────────
积       10001111
```

即　1011B×1101B=10001111B

4. 除法运算

规则：0/1=0；1/1=1

【例 4】 求 10100101B/1111B。

$$
\begin{array}{r}
1011\\
1111\,\overline{)\,10100101}\\
\underline{1111}\\
1011\\
\underline{0000}\\
10110\\
\underline{1111}\\
1111\\
\underline{1111}\\
0
\end{array}
$$

即　　10100101B/1111B=1011B

1.2.2 二进制数的逻辑运算

1. "与"运算

"与"运算是实现"必须都有，否则就没有"这种逻辑关系的一种运算。运算符为"・"，其运算规则如下：

$$0\cdot 0=0,\ 0\cdot 1=1\cdot 0=0,\ 1\cdot 1=1$$

【例 5】 若 X=1011B，Y=1001B，求 X・Y。

$$
\begin{array}{rr}
 & 1011\\
\cdot & 1001\\
\hline
 & 1001
\end{array}
$$

即　　X・Y=1001B

2. "或"运算

"或"运算是实现"只要其中之一有，就有"这种逻辑关系的一种运算，其运算符为"+"。"或"运算规则如下：

$$0+0=0,\ 0+1=1+0=1,\ 1+1=1$$

【例 6】 若 X=10101B，Y=01101B，求 X+Y。

$$
\begin{array}{rr}
 & 10101\\
+ & 01101\\
\hline
 & 11101
\end{array}
$$

即　　X+Y=11101B

3. "非"运算

"非"运算是实现"求反"这种逻辑的一种运算，如变量 A 的"非"运算记作 $\overline{A}$。其运算规则如下：

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

【例 7】 若 A＝10101B，求 $\overline{A}$。

$\overline{A}=\overline{10101}B=01010B$

4. “异或”运算

“异或”运算是实现“必须不同，否则就没有”这种逻辑的一种运算，运算符为“⊕”。其运算规则是：

$$0\oplus0=0，0\oplus1=1，1\oplus0=1，1\oplus1=0$$

【例 8】 若 X＝1010B，Y＝0110B，求 X⊕Y。

$$\begin{array}{r} 1010 \\ \oplus\ \ 0110 \\ \hline 1100 \end{array}$$

即 $X\oplus Y=1100B$

1.3 带符号数的表示

1.3.1 机器数及真值

计算机在数的运算中，不可避免地会遇到正数和负数，那么正负符号如何表示呢？由于计算机只能识别 0 和 1，因此，我们将一个二进制数的最高位用作符号位来表示这个数的正负。规定符号位用“0”表示正，用“1”表示负。例如，X＝－1101010B，Y＝＋1101010B，则 X 表示为：11101010B，Y 表示为 01101010B。

可见，一个二进制数连同符号位在内作为一个数，称为机器数，如 11101010B。而一般书写形式的数，称为该机器数的真值，如－1101010B。计算机中机器数的表示方法有三种，即原码、反码和补码。

1.3.2 数的码制

1. 原码

当正数的符号位用 0 表示，负数的符号位用 1 表示，数值部分用真值的绝对值来表示的二进制机器数称为原码，用$[X]_原$表示，设 X 为整数。

若 $X=+X_{n-2}X_{n-3}\cdots X_1X_0$，则$[X]_原=0X_{n-2}X_{n-3}\cdots X_1X_0=X$；

若 $X=-X_{n-2}X_{n-3}\cdots X_1X_0$，则$[X]_原=1X_{n-2}X_{n-3}\cdots X_1X_0=2^{n-1}-X$。

其中，X 为 n－1 位二进制数，X_{n-2}、X_{n-3}、…、X_1、X_0 为二进制数 0 或 1。例如＋115 和－115 在计算机中(设机器数的位数是 8)其原码可分别表示为

$$[+115]_原=01110011B；[-115]_原=11110011B$$

可见，真值 X 与原码$[X]_原$的关系为

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

值得注意的是，由于$[+0]_原=00000000B$，而$[-0]_原=10000000B$，所以数 0 的原码不唯一。

8 位二进制原码能表示的范围是：－127～＋127。

2. 反码

一个正数的反码，等于该数的原码；一个负数的反码，由它的正数的原码按位取反形成。反码用$[X]_{反}$表示。

若 $X=-X_{n-2}X_{n-3}\cdots X_1X_0$，则$[X]_{反}=1\,\overline{X_{n-2}X_{n-3}\cdots X_1X_0}$。例如：X＝＋103，则$[X]_{反}=[X]_{原}=01100111B$；X＝－103，$[X]_{原}=11100111B$，则$[X]_{反}=10011000B$。

可见，

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

注意：

(1) 8 位二进制反码能表示数的范围为：－127～＋127。

(2) 在反码中，＋0 与－0 的表示方法不同。

3. 补码

在讨论补码之前，先介绍模(mod)的概念。

“模”是指一个计量系统的计数量程。如，时钟的模为 12。任何有模的计量器，均可化减法为加法运算。仍以时钟为例，设当前时钟指向 11 点，而准确时间为 7 点，调整时间的方法有两种，一种是时钟倒拨 4 小时，即 11－4＝7；另一种是时钟正拨 8 小时，即 11＋8＝12＋7＝7。由此可见，在以 12 为模的系统中，加 8 和减 4 的效果是一样的，即

$$-4=+8\pmod{12}$$

下面引进补码表示法。对于 n 位计算机来说，数 X 的补码定义为

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

即正数的补码就是它本身，负数的补码是真值与模数相加而得。

例如，n＝8 时，

$$[+75]_{补}=01001001B$$

$$[-73]_{补}=10000000\,B-01001001B=10110111B$$

$$[0]_{补}=[+0]_{补}=[-0]_{补}=00000000B$$

可见，数 0 的补码表示是唯一的。在用补码定义求负数补码的过程中，由于做减法不方便，一般该法不用。负数补码的求法：用原码求反码，再在数值末位加 1，即：$[X]_{补}=[X]_{反}+1$。例如：$[-30]_{补}=[-30]_{反}+1=\overline{[+30]_{原}}+1=11100001+1=11100010B$。8 位二进制补码能表示的范围为：－128 ～＋127，若超过此范围，则为溢出。

1.4 定点数和浮点数

计算机中的数，既有整数也有小数，但在计算机中小数并不以单独的信息存放。为了确定小数点的位置，通常采用两种方法表示：定点法和浮点法。

1. 定点法

定点法中约定所有数据的小数点隐含在某个固定位置。对于纯小数，小数点固定在数

符与数值之间；对于整数，则把小数点固定在数值部分的最后面，其格式为

纯小数表示：数符. 尾数

数符	尾数

. 小数点

整数表示：数符尾数.

数符	尾数

. 小数点

其中，数符用来表示数的正负，正数为 0，负数为 1；尾数是指某数本身的数值部分。

定点法所能表示的数值范围很有限，当计算机采用定点法处理较大数值范围的运算时，很容易产生溢出。因此，为了扩大数的表示范围和精度，时常采用浮点表示。

2. 浮点法

浮点法中，数据的小数点位置不是固定不变的，而是可浮动的。因此，可将任意一个二进制数 N 表示成

$$N = \pm M \cdot 2^{\pm E}$$

其中，M 为尾数，为纯二进制小数，E 称为阶码。可见，一个浮点数有阶码和尾数两部分，且都带有表示正负的阶符与数符，其格式为

阶符	阶码 E	数符	尾数 M

设阶码 E 的位数为 m 位，尾数 M 的位数为 n 位，则浮点数 N 的取值范围为

$$2^{-n}2^{-2^m+1} \leqslant |N| \leqslant (1-2^{-n})2^{2^m-1}$$

为了提高精度，发挥尾数有效位的最大作用，还规定尾数数字部分原码的最高位为 1，叫做规格化表示法。如 0.000101 表示为：$2^{-3} \times 0.101$

1.5　BCD 码和 ASCII 码

1.5.1　BCD 码

人们习惯使用十进制数，为使计算机能识别、存储十进制数，并能直接使用十进制数进行运算，就需要对十进制数进行编码。将十进制数表示为二进制编码的形式，称为二—十进制编码，即 BCD(Binary Coded Decimal)码。

1 位十进制数有 0～9 共 10 个不同数码，至少需要由 4 位二进制数表示。4 位二进制数有 16 种组合，取其 10 种组合分别代表 10 个十进制数码。最常用的方法是 8421BCD 码，其中 8、4、2、1 分别为 4 位二进制数的位权值。表 1.2 给出了十进制数和 8421BCD 码的对应关系。

表 1.2　8421BCD 编码表

十进制数	8421BCD 码	十进制数	8421BCD 码
0	0000	5	0101
1	0001	6	0110
2	0010	7	0111
3	0011	8	1000
4	0100	9	1001

【例 1】 写出 69.25 的 BCD 码。

根据表 1.2，可直接写出相应的 BCD 码：

$$69.25 = (01101001.00100101)_{BCD}$$

1.5.2 ASCII 码

目前国际上比较通用的是 1963 年美国标准学会 ANSI 制定的美国国家信息交换标准字符码(American Standard Code for Information Interchange)，简称 ASCII 码。它的编码如表 1.3 所示，从表中可见，ASCII 码采用 7 位二进制编码，它包括 26 个大写英文字母；26 个小写英文字母；10 个数字 0～9；32 个通用控制符号；34 个专用符号，共 128 个字符。

表 1.3 ASCII 码 表

行	列	0	1	2	3	4	5	6	7
	MSB 位 654 LSB 位 3210	000	001	010	011	100	101	110	111
0	0000	NUL	DLE	SP	0	@	P	`	p
1	0001	SOH	DC$_1$	!	1	A	Q	a	q
2	0010	STX	DC$_2$	″	2	B	R	b	r
3	0011	ETX	DC$_3$	#	3	C	S	c	s
4	0100	EOT	DC$_4$	$	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	HS	/	?	O	←	o	DEL

如果要确定一个数字、字母或符号的 ASCII 码，可以先在表 1.3 中找到这个字符，然后将字符所在行与列所对应的二进制数连接起来(列对应的 3 位在前，行对应的 4 位在后)，所得到的 7 位二进制代码即为该字符的 ASCII 码。如，大写字母 W 的 ASCII 码为 1010111B(57H)。