单片机原理及接口技术

朱定华　编著

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
URL: http://www.phei.com.cn

单片机原理及接口技术

朱定华　编著

電子工業出版社
Publishing House of Electronics Industry
北京·BEIJING

内 容 简 介

《单片机原理及接口技术》以 MCS-51 单片机为背景机，系统地介绍了单片机的原理及接口技术。主要内容包括计算机基础知识、汇编语言程序设计、MCS-51 单片机的内部接口、MCS-51 单片机的扩展方法、半导体存储器、常用可编程接口芯片、A/D 和 D/A 转换芯片等。本书内容精练，实例丰富。其中大量的接口电路和程序是作者多年在科研和教学中反复提炼得来的，因而本书应用性很强。本书内容系统全面，论述深入浅出，循序渐进。可作为大专院校和高职、高专、成人本科等高等教育汇编语言程序设计、微机原理及应用或微机原理及接口技术等课程的教学用书。也可以供从事电子技术、计算机应用与开发的科研人员和工程技术人员学习参考，还适于初学者自学使用。

图书在版编目(CIP)数据

单片机原理及接口技术/朱定华编著. - 北京：电子工业出版社，2001.4
ISBN 7-5053-6583-5

Ⅰ.单… Ⅱ.朱… Ⅲ.①单片微型计算机 - 理论 - 高等学校：技术学校 - 教材 ②单片微型计算机 - 接口 - 高等学校：技术学校 - 教材 Ⅳ.TP368.1

中国版本图书馆 CIP 数据核字(2001)第 16702 号

书　　名：单片机原理及接口技术
编 著 者：朱定华
责任编辑：束传政　卢先河
特约编辑：朱　宇
排版制作：电子工业出版社计算机排版室
印 刷 者：北京东光印刷厂
装 订 者：三河市金马印装有限公司
出版发行：电子工业出版社　URL：http://www.phei.com.cn
　　北京市海淀区万寿路 173 信箱　邮编 100036
经　　销：各地新华书店
开　　本：787×1 092　1/16　印张：12.75　字数：320 千字
版　　次：2001 年 4 月第 1 版　2001 年 6 月第 2 次印刷
书　　号：ISBN 7-5053-6583-5
　　TN·1440
印　　数：2 000 册　　定价：18.00 元

前　言

当今微型计算机技术发展形成两大分支，一是以微处理器(Micro Processor Unit)为核心所构成的通用微机系统，主要用于科学计算、数据处理、图形图像处理、数据库管理、人工智能、数字模拟与仿真等领域。另一大分支是微控制器(Micro Controller Unit)，俗称单片机。单片机主要用于工业测控，如家用电器、计算机外围设备、工业智能化仪表、机器人、生产过程的自动控制、农业、化工、军事、航空航天等领域。面对这样"势均力敌"的两大分支，高等院校的微机原理课程究竟应选用何种机型为背景机来组织教学，也出现了分歧。

作为21世纪的工科大学生，不仅要熟练地使用通用微机进行各种数据处理，还要把计算机技术运用到本专业领域或相关领域，即具有"开发"能力。新世纪的工科大专院校的大学生既要掌握通用微机，又要掌握单片机，所以有些学校在学习以通用微机为背景机的微机原理课程后，又开设单片机及接口技术选修课。单片机和通用微机作为微型机发展的两大分支，其基本结构、工作原理、控制思路及实现方法都非常类似。有了一个做基础，再学另一个就很容易了。选用80x86通用微机和MCS-51系列单片机为背景机组织微机原理课程的教学都是可行的。

8位单片机应用于控制目的时，功能已足够强大，已能满足控制领域中多数场合的要求，虽然近十年来也发展出16位和32位产品，但在目前乃至今后相当长的时间内仍将以8位机为主。选用芯片巨人Intel公司的MCS-51系列8位单片机作为微机原理课程的主要内容，既可满足教学内容稳定、实验设备成熟便宜，又不失其先进性与实用性。因此，本书是以Intel公司的MCS-51系列单片机为背景机的微机原理及接口技术课程的教学用书 。

本书全面地介绍了MCS-51单片机的结构原理和应用技术，全书共分8章。第1章介绍微型计算机的基础知识，包括计算机中的数制和编码、逻辑单元和逻辑部件，微型计算机的结构和工作原理以及MCS-51单片机的结构和主要系列产品的特性。第2章介绍MCS-51单片机常用的汇编指令和伪指令以及指令的时序。第3章介绍汇编语言程序设计的基本技术。通过第2章和第3章的学习，使读者能更透彻地了解汇编语言程序设计，为编程应用打下基础。第4章介绍MCS-51单片机的内部接口电路，包括中断系统、定时器、并行口和串行口，同时还介绍了计算机间的通信。第5章介绍了MCS-51单片机的最小应用系统和扩展技术。第6章介绍存储器及其与微型计算机的接口技术。第7章介绍常用可编程接口芯片的功能与应用。第8章介绍A/D和D/A转换器与微型计算机的接口与应用。

本书内容精练、实例丰富，每章后均附有思考题与习题。编写本书时，注意了理论和实践相结合，力求做到既有一定的理论基础，又能运用理论解决实际问题；既掌握一定的先进技术，又着眼于为当前的应用服务。

本课程的参考学时数为64学时。与本书配套的教材有《单片机原理及接口技术学习辅导》和《单片机原理及接口技术实验》。

参加本书编写的还有朱悦、杜德军(武汉钢铁设计院)、程世平、蒙文武、林卫、王红和崔厚孝等。

由于作者水平有限，书中难免存在一些缺点和错误，殷切希望读者批评指正。

朱定华

2001.3.28

于武昌喻家山

目　录

第 1 章　微型计算机的基础知识

1.1　计算机中的数和编码

1.1.1　计算机中的数制

计算机最早是作为一种计算工具出现的，所以它的最基本的功能是对数进行加工和处理。数在机器中是以器件的物理状态来表示的。一个具有两种不同的稳定状态且能相互转换的器件就可以用来表示 1 位(bit)二进制数。二进制数有运算简单、便于物理实现、节省设备等优点，所以目前在计算机中数几乎都是采用二进制表示。但是，二进制数书写起来太长，且不便阅读和记忆。目前大部分微型机是 8 位、16 位或 32 位的，都是 4 的整数倍，而 4 位二进制数即是 1 位十六进制数，所以微型机广泛采用十六进制数来缩写二进制数。十六进制数用 0～9、A～F 共 16 个数码表示十进制数 0～15。1 个 8 位的二进制数用 2 位十六进制数表示，1 个 16 位的二进制数用 4 位十六进制数表示等。这样书写方便，且便于阅读和记忆。然而，人们最熟悉、最常用的是十进制数。为此，需要熟练地掌握十进制数、二进制数和十六进制数间的相互转换。它们之间的关系如表 1-1 所示。

表 1-1　十进制数、二进制数及十六进制数对照表

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

为了区别上述 3 种数制，可在数的右下角注明数制，或者在数的后面加一字母。如 B(binary)表示二进制数制，D(decimal)或不带字母表示十进制数制，H(hexadecimal)表示十六进制数制。

1. 二进制数和十六进制数间的相互转换

根据表 1-1 所示的对应关系即可实现它们之间的转换。

二进制整数转换为十六进制数，方法是从右(最低位)向左将二进制数分为每 4 位 1 组，每组用 1 位十六进制数表示，左边不足 4 位时应在左边加 0，以凑成 4 位 1 组。如：

1 1111 1100 0111B → 0001 1111 1100 0111B＝1FC7H

十六进制数转换为二进制数，只需用 4 位二进制数代替 1 位十六进制数即可。如：

3AB9H＝0011 1010 1011 1001B

2. 十六进制数和十进制数间的相互转换

十六进制数转换为十进制数十分简单，只需将十六进制数按权展开相加即可。如：

1F3DH＝(4096×1)＋(256×15)＋(16×3)＋(1×13)＝4096＋3840＋48＋13＝7997

十进制整数转换为十六进制数可用除 16 取余法，即用 16 不断地除待转换的十进制数，直至商等于 0 为止。将所得的各次余数，依倒序排列，即可得到所转换的十六进制数。如：

$$
\begin{array}{r|l}
16 & 38947 \quad 3 \\
16 & 2434 \quad 2 \\
16 & 152 \quad 8 \\
16 & 9 \quad 9 \\
 & 0
\end{array}
$$

即 $38947=9823\mathrm{H}$

1.1.2 符号数的表示法

1. 机器数与真值

二进制数与十进制数一样有正负之分。在计算机中，常用数的符号和数值部分一起编码的方法表示符号数，较常用的有原码、反码和补码表示法。这几种表示法都将数的符号数码化。通常正号用"0"表示，负号用"1"表示。为了区分一般书写时表示的数和机器中编码表示的数，我们称前者为真值，后者为机器数，即数值连同符号数码"0"或"1"一起作为一个数就称为机器数，而它的数值连同符号"+"或"-"称为机器数的真值。把机器数的符号位也当做数值的数，就是无符号数。

为了表示方便，常把 8 位二进制数称为字节，把 16 位二进制数称为字，把 32 位二进制数称为双字。对于机器数应将其用字节、字或双字表示，所以只有 8 位、16 位或 32 位机器数的最高位才是符号位。

2. 原码

按上所述，数值用其绝对值，正数的符号位用 0 表示，负数的符号位用 1 表示，这样表示的数称为原码。如：

$$X_1=105=+1101001\mathrm{B} \qquad [X_1]_{原}=01101001\mathrm{B}$$

$$X_2=-105=-1101001\mathrm{B} \qquad [X_2]_{原}=11101001\mathrm{B}$$

其中，最高位为符号位，后面 7 位是数值。用原码表示时，+105 和 -105 的数值部分相同而符号位相反。

原码表示简单易懂，而且与真值的转换方便。但若是两个异号数相加，或两个同号数相减，就要做减法。为了把减运算转换为加运算从而简化计算机的结构，引进了反码和补码。

3. 反码

正数的反码与原码相同，负数的反码为它的绝对值（即与其绝对值相等的正数）按位取反（连同符号位）。如：

$$X_1=105=+1101001\mathrm{B} \qquad [X_1]_{反}=01101001\mathrm{B}$$

$$X_2=-105=-1101001\mathrm{B} \qquad [X_2]_{反}=10010110\mathrm{B}$$

4. 补码

正数的补码与原码相同；负数的补码为与它的绝对值相等的正数的补数。把一个数连同符号位按位取反后再加 1，可以得到该数的补数。如：

$$X_1=105=+1101001\mathrm{B} \qquad [X_1]_{补}=01101001\mathrm{B}$$

$$X_2=-105=-1101001\mathrm{B} \qquad [X_2]_{补}=10010111\mathrm{B}$$

求补数还可以直接求，方法是从最低位向最高位扫描，保留直至第一个"1"的所有位，以后各位按位取反。负数的补码可以由其正数求补得到。根据两数互为补数的原理，对补码表示的负数求补就可以得到其正数，即可得该负数的绝对值。如：

$$[-105]_{补}=10010111\mathrm{B}$$

对其求补，从右向左扫描，第一位就是1，故只保留该位，对其左面的七位均求反得：01101001，即补码表示的机器数97H的真值是－69H＝－105。

一个用补码表示的机器数，若最高位为0，则其余几位即为此数的绝对值；若最高位为1，其余几位不是此数的绝对值，而需将该数求补，才得到它的绝对值。

当数采用补码表示时，就可以把减法转换为加法。例如：

$$X=64-10=64+(-10)$$

$$[X]_{补}=[64]_{补}+[-10]_{补}$$

$$[64]_{补}=40H=0100\ 0000B$$

$$[10]_{补}=0AH=0000\ 1010B$$

$$[-10]_{补}=1111\ 0110B$$

做减法运算过程如下：

$$\begin{array}{r} 0100\ 0000 \\ -\ 0000\ 1010 \\ \hline 0011\ 0110 \end{array}$$

用补码相加过程如下：

$$\begin{array}{r} 0100\ 0000 \\ +\ 1111\ 0110 \\ \hline 10011\ 0110 \end{array}$$

↑进位自然丢失

最高位的进位是自然丢失的，故做减法与用补码相加的结果是相同的。因此，在微型机中，凡是符号数一律是用补码表示的。一定要记住运算的结果也是用补码表示的，如：

$$34-68=34+(-68)$$

$$34=22H=0010\ 0010B$$

$$68=44H=0100\ 0100B$$

$$-68=1011\ 1100B$$

做减运算过程如下：

$$\begin{array}{r} 0010\ 0010 \\ -\ 0100\ 0100 \\ \hline 11101\ 1110 \end{array}$$

↑借位自然丢失

用补码相加过程如下：

$$\begin{array}{r} 0010\ 0010 \\ +\ 1011\ 1100 \\ \hline 1101\ 1110 \end{array}$$

结果相同。因为符号位为1，所以结果为负数。对其求补，得其真值为－00100010B，即－34（－22H）。

由上面两个例子还可以看出，当数采用补码表示后，两个正数相减，若无借位，化为补码相加就会有进位；若有借位，化作补码相加就不会有进位。

1.1.3 二进制数的算术运算

计算机把机器数均当做无符号数进行运算，即符号位也参与运算。运算的结果要根据运算结果的符号，运算有无进（借）位和溢出等来判别。计算机中设置有这些标志位，标志位的值由

运算结果自动设定。

1. 无符号数的运算

无符号数实际上是指参加运算的数均为正数，且整个数位全部用于表示数值。n 位无符号二进制数的范围为 $0\sim(2^n-1)$。

1）两个无符号数相加，由于两个加数均为正数，因此其和也是正数。当和超过其位数所允许的范围时，就向更高位进位。如：

$$127+160=7FH+0A0H$$

为了区分数字和符号，写字母开头的十六进制数，前面应添加一个 0。

$$\begin{array}{r} 0111\ 1111 \\ +\ 1010\ 0000 \\ \hline 10001\ 1111 \end{array} \quad =11FH=256+16+15=287$$

↑进位

2）两个无符号数相减，被减数大于或等于减数，无借位，结果为正；被减数小于减数，有借位，结果为负。如：

$$192-10=0C0H-0AH$$

$$\begin{array}{r} 1100\ 0000 \\ -\ 0000\ 1010 \\ \hline 1011\ 0110 \end{array} \quad =B6H=176+6=182$$

反过来相减，即 10－192，运算过程如下：

$$\begin{array}{r} 0000\ 1010 \\ -\ 1100\ 0000 \\ \hline 10100\ 1010 \end{array} \quad =-10110110B=-B6H=-182$$

↑借位

由此可见，对无符号数进行减法运算，其结果的符号用进位来判别：CF＝0（无借位），结果为正；CF＝1（有借位）结果为负（对 8 位数值位求补得到它的绝对值）。

2. 符号数的运算

n 位二进制数，除去一位符号位，还有 $n-1$ 位表示数值，所能表示的补码的范围为 $-2^{n-1}\sim(2^{n-1}-1)$。如果运算结果超过此范围就会产生溢出。如：

$$105+50=69H+32H$$

$$\begin{array}{r} 0110\ 1001 \\ +\ 0011\ 0010 \\ \hline 1001\ 1011 \end{array} \quad =9BH=155\text{ 或}=-65H=-101$$

若把结果视为无符号数，为 155，结果是正确的。若将此结果视为符号数，其符号位为 1，结果为－101，这显然是错误的。其原因是，和数 155 大于 8 位符号数所能表示的补码数的最大值 127，使数值部分占据了符号位的位置，产生了溢出，从而导致结果错误。又如：

$$-105-50=-155$$

$$\begin{array}{r} 1001\ 0111 \\ +\ 1100\ 1110 \\ \hline 10110\ 0101 \end{array}$$

↑进位

两个负数相加，和应为负数，而结果 01100101B 却为正数，这显然是错误的。其原因是，和数－155 小于 8 位符号数所能表示的补码数的最小值－128，也产生了溢出。若不将第 7 位（从第 0 位开始计）0 看作符号位，也看作数值而将进位看作数的符号，结果为－10011011B＝

－155，结果就是正确的。

因此，应当注意溢出与进位及补码运算中的进位或借位丢失间的区别，即

1）进位或借位是指无符号数运算结果的最高位向更高位进位或借位。通常多位二进制数将其拆成二部分或三部分或更多部分进行运算时，数的低位部分均无符号位，只有最高部分的最高位才为符号位。运算时，低位部分向高位部分进位或借位。由此可知，进位主要用于无符号数的运算，这与溢出主要用于符号数的运算是有区别的。

2）溢出与补码运算中的进位丢失也应加以区别，如：

$$-50-5=-55$$

$$\begin{array}{r} 1100\ 1110 \\ -\ 1111\ 1011 \\ \hline 11100\ 1001 \end{array} \quad =-00110111\mathrm{B}=-55$$

↑进位

两个负数相加，结果为负数是正确的。这里虽然出现了补码运算中产生的进位，但由于和数并未超出 8 位二进制数－128～127 的范围，因此无溢出。那么，如何来判别有无溢出呢？

设符号位向进位位的进位为 C_y，数值部分向符号位的进位为 C_s，则溢出

$$O = C_y \oplus C_s$$

$O=1$，有溢出；$O=0$，无溢出。

下面用 M、N 两数相加来证明。设 M_s 和 N_s 为两个加数的符号位，R_s 为结果的符号位，则表 1-2 所列的真值表。由真值表得逻辑表达式：

$$O = \overline{C}_s C_y + C_s \overline{C}_y = C_s \oplus C_y$$

表 1-2　符号、进位、溢出的真值表

M_s	N_s	R_s	C_s	C_y	O
0	0	0	0	0	0
0	0	1	1	0	1
0	1	0	1	1	0
0	1	1	0	0	0
1	0	0	1	1	0
1	0	1	0	0	0
1	1	0	0	1	1
1	1	1	1	1	0

再来看 105＋50、－105－50 和－50－5 三个运算有无溢出。

$$\begin{array}{r} 0110\ 1001 \\ +\ 0011\ 0010 \\ \hline 1001\ 1011 \end{array}$$

$C_y=0 \quad C_s=1$

$O=0\oplus1=1$　有溢出

$$\begin{array}{r} 1001\ 0111 \\ +\ 1100\ 1110 \\ \hline 10110\ 0101 \end{array}$$

$C_y=1 \quad C_s=0$

$O=1\oplus0=1$　有溢出

$$\begin{array}{r} 1100\ 1110 \\ +\ 1111\ 1011 \\ \hline 11100\ 1001 \end{array}$$

$C_y=1 \quad C_s=1$

$O=1\oplus1=0$　无溢出

1.1.4　二进制数的逻辑运算与逻辑电路

计算机除了可进行基本的算术运算外，还可对两个或一个无符号二进制数进行逻辑运算。计算机中的逻辑运算，主要是指“逻辑非”、“逻辑乘”、“逻辑加”和“逻辑异或”等四种基本运算。下面分别介绍这四种基本逻辑运算及实现这些运算的逻辑电路。

1. 逻辑非

逻辑非也称"求反"。对某二进制数进行逻辑非运算，就是按位求它的反，常用变量上方加上划线表示。

例 1.1 $A=01100001$，$B=11001011$，求 $\overline{A}$、$\overline{B}$。

解 $\overline{A}=10011110$　　$\overline{B}=00110100$

实现逻辑非运算的电路称为非门，又称反相器。它只有一个输入和一个输出。它的国标符号和国外符号如图 1-1 所示。

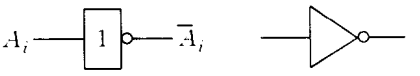

图 1-1　非门的符号表示

2. 逻辑乘

对两个二进制数进行逻辑乘，就是按位求它们的"与"，所以逻辑乘又称"逻辑与"，常用记号"∧"或"·"表示。1 位二进制数逻辑乘的规则为

$$0\wedge 0=0 \quad 0\wedge 1=0 \quad 1\wedge 0=0 \quad 1\wedge 1=1$$

例 1.2 $A=01100001$，$B=11001011$，求 $A\wedge B$

解

$$\begin{array}{r} 0110\ 0001 \\ \wedge\ 1100\ 1011 \\ \hline 0100\ 0001 \end{array}$$

$\therefore A\wedge B=0100\ 0001$

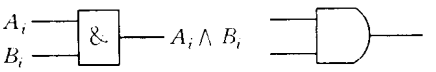

图 1-2　与门的符号表示

实现逻辑乘运算的电路称为与门，2 个输入与门的国标符号和国外符号如图 1-2 所示。

3. 逻辑加

对两个二进制数进行逻辑加，就是按位求它们的"或"，所以逻辑加又称"逻辑或"，常用记号"∨"或"+"来表示。1 位二进制数逻辑加的规则为

$$0\vee 0=0 \quad 0\vee 1=1 \quad 1\vee 0=1 \quad 1\vee 1=1$$

例 1.3 $A=01100001$，$B=11001011$，求 $A\vee B$

解

$$\begin{array}{r} 0110\ 0001 \\ \vee\ 1100\ 1011 \\ \hline 1110\ 1011 \end{array}$$

$\therefore A\vee B=11101011$

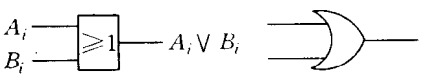

图 1-3　或门的符号表示

实现逻辑加运算的电路称为或门，2 个输入或门的国标符号和国外符号如图 1-3 所示。

4. 逻辑异或

对两个二进制数进行逻辑异或，就是按位求它们的模 2 和，所以逻辑异或又称"按位加"，常用符号"⊕"来表示。1 位二进制数的逻辑异或运算规则为

$$0\oplus 0=0 \quad 0\oplus 1=1 \quad 1\oplus 0=1 \quad 1\oplus 1=0$$

例 1.4 $A=01100001$，$B=11001011$，求 $A\oplus B$。

解

$$\begin{array}{r} 0110\ 0001 \\ \oplus\ 1100\ 1011 \\ \hline 1010\ 1010 \end{array}$$

$\therefore A\oplus B=1010\ 1010$

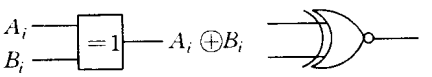

图 1-4　异或门的符号表示

注意：按位加与普通整数加法的区别是它不考虑低位产生的进位。

实现逻辑异或运算的电路称为异或门，2 输入异或门的国标符号和国外符号如图 1-4 所示。

异或门的特点是，只有当输入的两个变量相异时，输出为高（“1”），否则输出为低（“0”）。

5. 正逻辑与负逻辑

逻辑电路实现的逻辑关系，可用高电平表示逻辑 1，用低电平表示逻辑 0，在这种规定下的逻辑关系称为正逻辑。事实上，还有另一种规定：用低电平表示逻辑 1，用高电平表示逻辑 0，在这种规定之下的逻辑关系称为负逻辑。对于同一个逻辑电路，由于采用的逻辑制度不同，使它实现的逻辑关系也就不同。

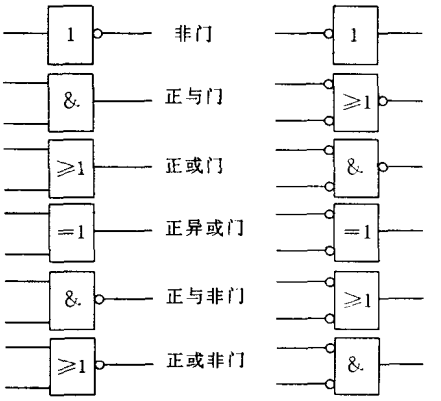

图 1-5　正负逻辑门电路的符号表示

逻辑门的正逻辑和负逻辑形式如图 1-5 所示。在图中我们看到，每一种逻辑门都可以用两种逻辑符号来表示。在电路的输入和输出端上的小圆圈可以看成是逻辑运算中的“非”，所以称作反相圈。在逻辑电路中使用反相圈是为了强调此处的逻辑关系是负逻辑。分析“正与门”和“正或门”的输入输出关系可以看出，正与门就是负或门，正或门就是负与门。计算机中常使用负逻辑，即信号多数是低电平有效，所以正逻辑的“或门”往往都表示成负逻辑的“与门”形式，就是为了表示这里是低电平的“与”操作，即输入同时为低电平时，输出为低电平。

1.1.5　二进制编码

如上所述，计算机中数是用二进制表示的，而计算机又应能识别和处理各种字符，如大小写英文字母、标点符号、运算符号等。这些又如何表示呢？在计算机里，字母、各种符号以及指挥计算机执行操作的指令，都是用二进制数的组合来表示的，称为二进制编码。

1. 二进制编码的十进制数

十进制数由 0～9 这 10 个数码组成。要表示这 10 个数码需要用 4 位二进制数，这称为二进制编码的十进制数，简称为 BCD 数(Binary Coded Decimal)。用 4 位二进制数编码表示 1 位十进制数的方法很多，较常用的是 8421 BCD 码，如表 1-3 所示。

表 1-3　8421 BCD 编码表

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

8 位二进制数可以放 2 个十进制数位，这种表示的 BCD 数称为压缩的 BCD 数。而把用 8

位二进制数表示1个十进制数位的数称为非压缩的BCD数。例如，将十进制数1994用压缩的BCD数表示为

0001100110010100B

而用非压缩的BCD数表示为

00000001 00001001 00001001 00000100B

十进制数与BCD数的转换是比较直观的，但BCD数与二进制数之间的转换是不直接的，要先将其转换为十进制数，反之亦然。

2. ASCII(American Standard Code for Information Interchange)码

计算机是用8位二进制数表示一个字符，普遍采用ASCII码，常用字符的ASCII码如表1-4所示。

表1-4 常用字符的ASCII码

字　符	ASCII码(H)
0～9	30～39
A～Z	41～5A
a～z	61～7A
换行LF	0A
回车CR	0D

十进制数的10个数码0～9的ASCII码是30H～39H，它们的低4位与其BCD码相同，且又是用8位二进制数表示1个十进制数，因此也称非压缩BCD数为ASCII BCD数。

1.1.6 BCD数的加减运算

1. BCD数加法

BCD数低位与高位之间是逢“十”进一的，而4位二进制数是逢“十六”进一的。因此，当两个BCD数相加时，相加各位的结果都在0～9之间，则其加法运算规则与二进制数的加法规则完全相同；若大于9，则应对其进行加6调整。如：48+59，因为低4位相加，和为17大于9，高4位与低4位的进位相加，和为10也大于9，故都应加6调整，其运算和调整过程如下：

```
  0100 1000
+ 0101 1001
-----------
  1010 0001
+ 0110 0110
-----------
 10000 0111
```

和为107。

2. BCD数减法

两个BCD数相减，若本位的被减数大于或等于减数，则减法规则与二进制数完全相同；反之，就会向高位借位，十进制数向高位借1作10，而按二进制运算规则，借1作16，因此应进行减6调整。如28－19，低位8减9，向高位借位故应减6调整。其运算与调整过程如下：

```
  0010 1000
- 0001 1001
-----------
  0000 1111
- 0000 0110
-----------
  0000 1001
```

差为9。

通常，计算机中都设置有二、十进制数的调整电路，BCD 数的运算也把数当做二进制数做二进制数的运算，运算后再调整。

1.2 逻辑单元与逻辑部件

逻辑部件是用来对二进制数进行寄存、传送和变换的数字部件，其种类繁多，本书仅简单地介绍微型计算机中常用的几种逻辑部件。构成逻辑部件的基本单元电路是触发器，所以首先介绍触发器。

1.2.1 触发器

触发器是具有记忆功能的基本逻辑单元电路。它能接收、保存和输出逻辑信号“0”和“1”。各类触发器都可以由逻辑门电路组成。

1. 基本 RS 触发器

基本 RS 触发器是最简单的触发器，它是将两个与非门的输入与输出交叉连接构成，如图 1-6 所示。触发器的两个输入端分别是 $\overline{R}$ 和 $\overline{S}$，其中 $\overline{S}$ 端称为置 1 或置位(set)端，$\overline{R}$ 端称为置 0 或复位(reset)端。触发器有两个输出端 Q 和 $\overline{Q}$，在正常工作时，它们总是处于互补的状态。我们用 Q 端的状态来表示触发器的状态。由与非门的逻辑功能决定，要使触发器为 1 状态，可使 $\overline{S}=0$，$\overline{R}=1$。同样要使触发器为 0 状态，需令 $\overline{R}=0$，$\overline{S}=1$。触发器一旦为 1 状态(或 0 状态)，$\overline{S}$(或 $\overline{R}$)端从 0 变成 1，触发器将保持 1 状态(或 0 状态)不变。即 $\overline{R}=1$，$\overline{S}=1$ 时触发器的状态不变。$\overline{R}$ 和 $\overline{S}$ 不能同时为 0，因为同时为 0 时，Q 和 $\overline{Q}$ 都为 1。当这种输入状态消失时，触发器的 Q 可能为 0，也可能为 1，至于是 0 还是 1，是不确定的。

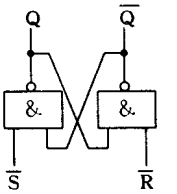

图 1-6 基本 RS 触发器的结构

2. 同步 RS 触发器

基本 RS 触发器中，输入端的触发信号直接控制触发器的状态。但在实际应用中，还希望触发器受一个时钟信号控制，做到按时钟信号的节拍翻转。这个控制信号称为时钟脉冲 CP(Clock Pulse)。引入 CP 后，触发器的状态不是在输入信号(R、S 端)变化时立刻转换，而是等待时钟信号到达时才转换。在多个这种触发器组成的电路中，各触发器受同一个时钟控制，触发器都在同一个时刻翻转，故得名同步 RS 触发器，而基本 RS 触发器称为异步 RS 触发器。

同步 RS 触发器的电路结构如图 1-7 所示。该电路由基本 RS 触发器和控制电路两部分组成。在时钟脉冲未到来时(即 CP=0 时)，由于控制电路的两个与非门均被封锁，它们的输出都为 1，使基本 RS 触发器维持原状态不变。在时钟脉冲作用期间(即 CP=1 时)，控制电路的两个非与门均被开启，R 和 S 端的输入被反相后送到基本 RS 触发器的输入端。由基本 RS 触发器的逻辑功能可知，若 RS=01 则触发器被置位，若 RS=10，则触发器被复位；RS=00 时触发器的状态不变；RS=11 的输入状态，对同步 RS 触发器是不允许的。

3. D 触发器

同步 RS 触发器工作时，不允许 RS 端的输入信号同时为 1。如果将 R 端改接到控制电路另一个与非门的输出端，只在 S 端加入输入信号，S 端改称为 D 端，同步 RS 触发器就转换成了 D 触发器。D 触发器的电路结构、逻辑符号和真值表如图 1-8 所示。由于总是将 D 端的输入反相后作为另一个与非门的输入信号，故无论 D 端的状态如何，都满足 RS 触发器的约束条