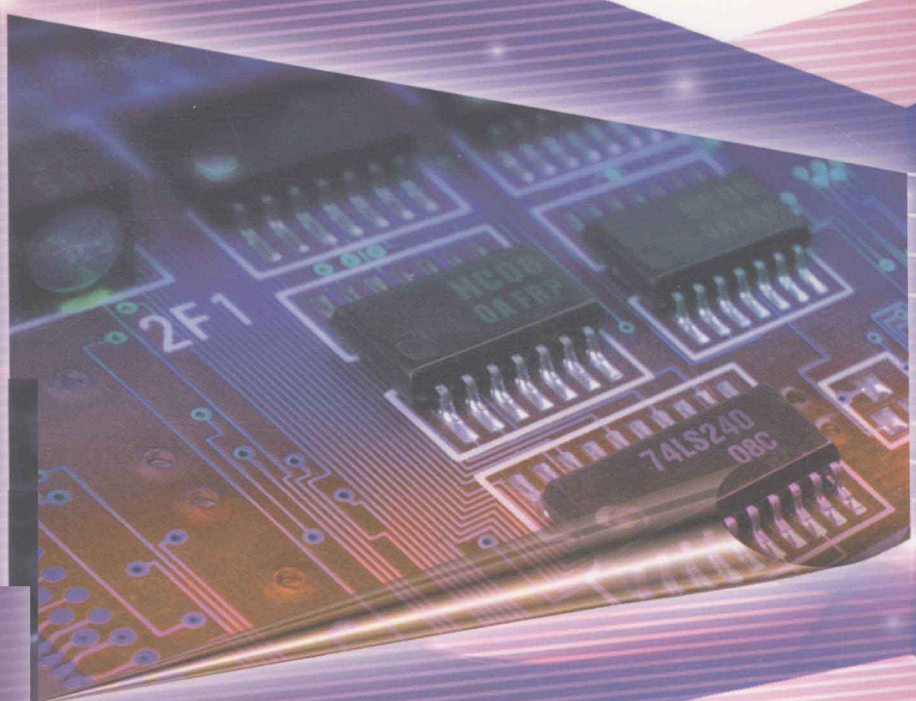
普通高等工科教育“十一五”规划教材

单片机原理及其应用

第2版

陈立周　陈宇　编



机械工业出版社
CHINA MACHINE PRESS

普通高等工科教育“十一五”

单片机原理及其应用

第 2 版

陈立周　陈　宇　编

机 械 工 业 出 版 社

本书是根据普通高等专科学校和高等职业技术学院机电类的教学培养计划，以及“单片机原理及其应用”课程的基本要求而编写的。内容包括单片机的基础知识、8051 系列单片机的结构、MCS-51 指令系统、编程技巧、存储器的扩展方法、中断、并口、串口、定时/计数器的结构与原理、C51 编程，以及单片机控制系统的硬件设计、软件调试等。由于本课程是实践性较强的课程，所以在内容上既注意讲述有关单片机的基础理论，也注意介绍在开发应用中会遇到的实际问题。

为适应近年来单片机技术的发展，本书在第 1 版的基础上作了修改补充，增加了串行扩展技术、对 PC 的串行通信、Windows 环境下集成开发软件等内容，以提高学生开发单片机应用系统的能力。还对第 1 版某些内容作了较详细的阐述，增加实例，使之更便于自学。

本书可作为普通高等专科学校和高等职业技术学院机电类专业有关单片机原理及应用、单片机控制系统、单片机接口之类课程的教材。也可以供电大和从事单片机控制系统开发工作的工程技术人员学习参考。

图书在版编目(CIP)数据

单片机原理及其应用/陈立周，陈宇编．—2 版．—北京：机械工业出版社，2008.5

普通高等工科教育“十一五”规划教材

ISBN 978-7-111-08197-5

Ⅰ．单… Ⅱ．①陈…②陈… Ⅲ．单片微型计算机—高等学校—教材 Ⅳ．TP368.1

中国版本图书馆 CIP 数据核字(2008)第 039150 号

机械工业出版社(北京市百万庄大街 22 号，邮政编码 100037)

责任编辑：贡克勤 版式设计：霍永明 责任校对：陈立辉

封面设计：王奕文 责任印制：洪汉军

北京铭成印刷有限公司印刷

2008 年 5 月第 2 版第 1 次印刷

184mm×260mm · 15 75 印张 · 385 千字

0001—4000 册

标准书号· ISBN 978-7-111-08197-5

定价：26 00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

销售服务热线电话：(010)68326294

购书热线电话：(010)88379639 88379641 88379643

编辑热线电话：(010)88379727

封面无防伪标均为盗版

前　言

本书是参照高等专科学校和高等职业技术学院有关电类专业的教学计划及其培养目标，以及就业岗位对从事单片机控制系统设计人员的要求而编写的。经过几年使用，我们感到有必要对本书初版内容作些修改，增加例题，使它更加切合当前的教学实际，更有利于当前高专、高职同学的自学。希望能为高专、高职的师生提供一本更加适合的单片机原理和应用的教材。

“单片机原理及其应用”是一门实践性较强的课程，既要加强基础知识和基本理论，又要提高其实用性。所以在修订中一方面着重于基本原理的讲述，另一方面根据近年来单片机技术的发展、片内功能的加强、开发软件和开发设备的更新换代，以及新的应用器件的不断出现，力求修订后能反映这些领域的发展与变化，使得修订后教材，在原理与应用、深度与广度方面，能够更好地适应高专与高职层次。

考虑到教学时数的限制，在内容上力求精简，使得教师能在规定学时内教完主要内容。所增加的部分内容，着重于实际应用、由浅入深，多用实例，以便于同学自学。本书目录中打 * 的部分，可以根据教学课时数和学生的具体情况选择使用，也可以不讲，待将来直接从事单片机开发时，作为自学内容。

本书由陈立周和陈宇编写，雷伍老师主审。在审阅中主审根据多年从事单片机开发的经验，对本书内容提了许多宝贵的意见，在此表示深切的感谢，并对福建工程学院计算机与信息科学系有关老师和工作人员在修订过程中给予的支持与帮助致以深切的谢意。

书中难免还会存在问题和错误，敬请使用本书的老师和同学给予批评指正。编者电子信箱为 chenlz@fjut.edu.cn。

编　者

目　录

第一章　单片机的基础知识

第一节　不同进位计数制及其互换

电子计算机包括单片机都是一种对数据信息进行处理与控制的机器，因此在学习计算机之前有必要先了解一下有关数的知识。

在人们日常生活中，都习惯于使用十进制，但在数字电路和计算机内部，由于只能通过电位高低表示1和0两个数码，所以计算机内部不用十进制而用二进制。这样，在使用计算机的时候，经常需要把常用的十进制转换为二进制。又由于用二进制表示一个数，所用的数码长，不但书写和阅读不方便，而且容易出错，所以书写时又常把二进制数据转换为十六进制。有些开发软件甚至要指定使用某种进制，为此学习单片机的第一步，先要熟悉这三种进位计数制间的互换。

二进制只使用0和1两个数码，超过1按逢2进1的规则进位。所以每左移1位，该位所代表的数值是前一位的2倍。每右移1位，数值为前一位的1/2。

十进制使用0～9共10个数码，超过9按逢10进1规则进位。所以每左移1位，该位所代表的数值是前一位的10倍。每右移1位，数值为前一位的1/10。

十六进制由16个数码组成，除0～9外，还另加A、B、C、D、E、F共6个字母，分别代表10、11、12、13、14、15，超过15按逢16进1规则进位。所以每左移1位，该位所代表的数值是前一位的16倍。每右移1位，数值为前一位的1/16。

任何一种进位计数制，其位数多少决定于所表示的数的大小，位数越多，所表示的数值也越大，考虑到本书所讲的单片机是一种字长为8位的计算机，所以下面讨论二进制时每个字节都取8位。超过8位的数，则用两个或两个以上字节表示。

为不使三种的进制数相混淆，一般可在二进制数的后面加上符号B，例如二进制数01100110B。十六进制数的后面加上符号H，例如6EH、4BH等；也可以不在后面加H，而在前面加$或0X，如$5E、$4B、0X5E、0X4B等等。不加B或H的数，一般就默认为十进制。

一、二进制与十六进制数的互换

一个4位二进制数，相当于1位的十六进制数。所以这两种进位制的数进行互换比较简单，可以把每4位的二进制数划为一组，然后对每一组进行相应的变换，例如二进制数01101110B转换为十六进制可写成：

0110	1110
6	E

同样，把十六进制转换成二进制，每一位的十六进制数对应4位二进制数，例如：

4	B
0100	1011

并按习惯十六进制数写成6EH、4BH，二进制数写成01101110B、和01001011B。如果被转换的是一个小数，则分组时应以小数点为准；整数从小数点开始，从右向左每4位划为一组，小数部分则从小数点开始，从左向右也以4位为一组，如果最后一组不足4位，可以用零补齐。以数1101100.11011为例，数的前后都要补0，使小数点前后都补足8位，即形成

0110 1100. 1101 1000

6 C. D 8

即1101100.11011B，等于6C.D8H

例1-1 将十六进制数00H和0FFH转换为二进制数。

解：00H=00000000B

0FFH=11111111B(在书写十六进制数时,若打头的数为A~F,则应在A~F之前再加一个0,以表示这是一个数而不是其他符号)。

二、二进制与十进制数的互换

对于二进制整数，各位数的权可以用底数为2的$n-1$次幂来确定，n表示该数的位数，即第一位的权为$2^0=1$，第2位的权为$2^1=2$，…若已知一个二进制数为10101010B，对应的十进制值应为170，它的按权展开式为

$$10101010B=1\times2^7+0\times2^6+1\times2^5+0\times2^4+1\times2^3+0\times2^2+1\times2^1+1\times2^0=170$$

对于二进制小数，其小数点以后各位的权，可以用底数为2的负n次幂来确定，n同样表示位数，即从小数点向右算起，第1位的权为$2^{-1}=0.5$，第2位的权为$2^{-2}=0.25$，…例如求11001100.00110011B的十进制值，其按权展开式为

$$\begin{aligned}11001100.00110011B=&1\times2^7+1\times2^6+0\times2^5+0\times2^4+1\times2^3+1\times2^2+0\times2^1+0\times2^0\\&+0\times2^{-1}+0\times2^{-2}+1\times2^{-3}+1\times2^{-4}+0\times2^{-5}+0\times2^{-6}+1\times2^{-7}\\&+1\times2^{-8}=204.19921875\end{aligned}$$

反过来：要将十进制整数转换为二进制数，可以采用逐次除以2余数反序排列的方法，所谓反序排列；指第1次除以2的余数排在最低位。以十进制数25为例，逐次除以2列式如下：

$$\begin{aligned}25\div2&=12\cdots\cdots\text{余}1\\12\div2&=6\cdots\cdots\text{余}0\\6\div2&=3\cdots\cdots\text{余}0\\3\div2&=1\cdots\cdots\text{余}1\\1\div2&=0\cdots\cdots\text{余}1\end{aligned}$$

由于8位微型计算机习惯将二进制数写成8位，可得

$$25=00011001B$$

如果二进制数不超过8位，即十进制数不超过255，也可以不必列式，直接用口算转换。

要把十进制小数转换为二进制数，则小数部分不是用除2，而是逐次乘2，每次乘积若产生整数则将整数个位(即所为溢出位)按正序排列，小数部分继续乘2。以33.6875为例，其整数部分

$$33=00100001B$$

其小数部分逐次乘 2

$$0.6875 \times 2 = 1.375 \quad \cdots\cdots \text{小数点左边整数为 } 1$$
$$0.375 \times 2 = 0.75 \quad \cdots\cdots \text{小数点左边整数为 } 0$$
$$0.75 \times 2 = 1.5 \quad \cdots\cdots \text{小数点左边整数为 } 1$$
$$0.5 \times 2 = 1 \quad \cdots\cdots \text{小数点左边整数为 } 1$$

可得

$$33.6875 = 00100001.10110000B$$

三、十进制与十六进制数的互换

由于已经掌握了十进制与二进制的互换以及十六进制与二进制的互换，因此要把十进制数转换为十六进制数，可以先转换成二进制数，再改写成十六进制数。反之，十六进制数也可以先改成二进制数，再转换成十进制数。

十六进制数也可以按各位的权，利用按权展开式，求出该数对应的十进制数值。整数各位的权等于底数为 16 的 $n-1$ 次幂，（n 为位数），即第 1 位的权为 $16^0 = 1$，第 2 位的权为 $16^1 = 16$，依次类推。例如十六进制的数为 8A71H，可用按权展开式求出十进制值，即

$$8A71H = 8 \times 16^3 + 10 \times 16^2 + 7 \times 16^1 + 1 \times 16^0 = 35441$$

对于十六进制的小数，其小数点以后各位的权，同样可以用底数为 16 的负 n 次幂来确定，n 表示位数，即从小数点向右算起，第一位的权为 $16^{-1} = 0.0625$，第二位的权为 $16^{-2} = 0.00390625$……。例如某个十六进制小数为 0.4AC9H，转换为十进制值的按权展开式为

$$0.4AC9H = 4 \times 16^{-1} + 10 \times 16^{-2} + 12 \times 16^{-3} + 9 \times 16^{-4} = 0.2921295$$

反过来，要把十进制转换为十六进制数，其方法与十进制数转换为二进制相似，即整数部分采用逐次除以 16，余数反序排列的方法。

$$13562 \div 16 = 847 \quad \cdots\cdots \text{余 } 10\text{（记作 0AH）}$$
$$847 \div 16 = 52 \quad \cdots\cdots \text{余 } 15\text{（记作 0FH）}$$
$$52 \div 16 = 3 \quad \cdots\cdots \text{余 } 4$$
$$3 \div 16 = 0 \quad \cdots\cdots \text{余 } 3$$

可得

$$13562 = 34FAH$$

十进制小数转换为十六进制小数，同样采用小数部分逐次乘 16，每次乘积若产生整数，则将所得整数按正序排列，例如十进制小数 0.359375 转换为十六进制数：

$$0.359375 \times 16 = 5.75 \quad \cdots\cdots \text{小数点左边整数为 } 5$$
$$0.75 \times 16 = 12.0 \quad \cdots\cdots \text{小数点左边整数为 } 0CH$$

可得

$$0.359375 = 0.5CH$$

例1-2　将常用的十六进制数 0FFH、0100H、0200H、0400H、0800H、0FFFFH、010000H 化为二进制数和十进制数。

解：先按每一位十六进制数对应 4 位二进制数的原则，将十六进制数转换为二进制数，再用按权展开式或直接用口算转换为十进制：

$$0FFH = 11111111B = 255$$
$$0100H = 00000001\ 00000000B = 256$$

0200H = 00000010 00000000B = 512
0400H = 00000100 00000000B = 1024(简称 1K)
0800H = 00001000 00000000B = 2048(简称 2K)
0FFFFH = 11111111 11111111 = 65535(简称 64K)
010000H = 00000001 00000000 00000000B = 65536

第二节 带符号的二进制数

一、带符号二进制数与不带符号二进制数的区别

在数学运算中，表示一个数的正负，可以在数的前面冠以正号或负号。但计算机只能辨认 0 和 1 两个数码，不能辨认其他符号，如果要表示正负，可以在字长为 8 位的二进制数中，将最高位规定为符号位，最高位为 0 表示该数为正，最高位为 1 表示该数为负。例如数 01101111B 表示该数为 +1101111B，而数 11101111B 则表示该数为 -1101111B。这种将最高位定为符号位的二进制数，称为带符号的二进制数。对于 8 位字长的带符号二进制数来说，扣除符号位外，表示数值的仅有 7 位，所能表示的值范围为 +127 ~ -127。

应该注意，同样一个二进制数，它既可以是不带符号数，也可以是带符号数。例如二进制数 11001100B，既可以看作是不带符号数 204，也可以是带符号数 -76，到底它是 204 还是 -76，取决于事先约定，仅从数的本身是无法判别的。

例如下面要介绍的相对转移指令中的偏移量，约定必须使用带符号数，因此出现在偏移量中的二进制数，总是认为它是一个带符号数，在填写偏移量时，也应使用带符号数，而程序中的地址值，约定为不带符号数，因此出现在地址中的数都是不带符号数。

总之，一个二进制数是带符号的二进制数，还是不带符号的二进制数，数的本身是无法区别的，只能根据它出现的场合，以及该场合约定使用什么数才能区分它们。

二、带符号数的表示方法

上面讲过，一个带符号数，它的最高位是符号位，其余表示数值。这种表示方式称为原码，实际上，带符号的二进制数，除了原码表示法之外，还可以用反码与补码表示，下面分别加以介绍。

(一) 原码(True form)

用原码表示一个带符号二进制数，其最高位为符号位，其余表示数值，例如：

$$x = +1010101B \quad (x)_{t.f} = 01010101B$$
$$x = -1010101B \quad (x)_{t.f} = 11010101B$$

式中，x 为真值；$(x)_{t.f}$表示真值 x 的原码，对于数 0

$$(+0)_{t.f} = 00000000B$$
$$(-0)_{t.f} = 10000000B$$

(二) 反码(One's complement)

反码也是带符号数的一种表示法，它同样规定最高位为符号位，其余则要看是正数还是负数。对于正数，其余各位表示数值，也就是正数的反码与原码相同。对于负数，其余各位将真值取反，即将 1 换成 0，0 换成 1，例如：

$$x = +1010101B \quad (x)_{o.c} = 01010101B$$

$$x=-1010101\text{B}\quad (x)_{o.c}=10101010\text{B}$$

式中，x 为真值；$(x)_{o.c}$表示真值 x 的反码。对于数 0 的反码，也有两种形式：

$$(+0)_{o.c}=00000000\text{B}$$

$$(-0)_{o.c}=11111111\text{B}$$

（三）补码(Two's complement)

补码仍然把最高位定为符号位，对于正数，其余各位表示数值，可见对一个确定的正数，它的补码、反码与原码完全相同。对于负数，除符号位不变外，其余各位逐位取反后再加 1，简称为取反加 1，例如：

$$x=+1010101\text{B}\quad (x)_{t.c}=01010101\text{B}$$

$$x=-1010101\text{B}\quad (x)_{t.c}=10101011\text{B}$$

式中，x 为真值；$(x)_{t.c}$表示真值 x 的补码。

补码这个概念与某个具体计数器的最大容量有关，以常用的 8 位二进制数为例，扣除最高位作为符号位外，计数的最大容量为 7 位数。当计数器计至 128 时，最高位产生溢出，又由于计数器只有 7 位，溢出的数就被丢弃。这个被丢弃的值就是最大容量值，又称为模，用符号 Mod 表示。对于模等于 128 的 7 位计数器，0 与 128 的数码相同，或者说 0 与 128 等价。即

$$0=0000000\text{B}$$

$$128=(1)0000000\text{B}$$

括号中的 1 就是被丢弃的数。可见，若模为 m，则 a 与 $a+m$ 等价，例如模为 128 时，1 与 129 等价，2 与 130 等价。因为不计及溢出数，所以计数器内的示值是相同的，即

$$a=a+m(\text{mod 为 } m)$$

再把这个概念推广到负数领域，例如 $a=(-2)$，代入上式有

$$(-2)=(-2)+128=126\quad (\text{mod 为 } 128)$$

即模为 128 时，(-2)与 126 等价，或者说这两个数互为补数，同样，可推出(-3)的补码为 125；(-4)的补码为 124。

为了进一步理解这个概念，可以用时钟做为例子，时钟的钟面最大示数为 12。时钟走到 12 点就等于 0 点，数 12 就被自动丢弃。可见时钟的钟面是一个模为 12 的计数器。时钟的时针向前拨 3 个字($a=3$)，跟向前拨 15 个字($a+m=3+12=15$)都停在同一位置上，也就是说 +3 与 +15 等价。

如果将时钟向后拨定义为负数，那么时钟向后拨 3 个字($a=(-3)$)，跟时针向前拨 9 个字($a=a+m=(-3)+12=9$)都停在相同位置上，也就是 -3 与 9 等价。因为对于模为 12 的钟面来讲，-3 可以用其补码 9 来表示。

要求得一个数的补码，可以用公式 $a=a+m(\text{mod 为 } m)$，也可以采用正数补码等于原码，负数补码等于取反加 1 的方法求得(注意:取反加 1 不包括符号位)。反过来，要从补码求原码，同样用取反加 1，表 1-1 是 8 位带符号二进制数的原码、反码、补码对照表。注意表中的

$$(+0)_{t.c}=00000000\text{B}$$

$$(-0)_{t.c}=00000000\text{B}$$

而 10000000B 不是(-0)的补码，而是(-128)的补码。

表 1-1 二进制数原码、反码、补码对照表

十进制数	二进制数	原码	反码	补码	十进制数	二进制数	原码	反码	补码
+0	00000000	00000000	00000000	00000000	-1	-0000001	10000001	11111110	11111111
+1	00000001	00000001	00000001	00000001	-2	10000010	10000010	11111101	11111110
+2	00000010	00000010	00000010	00000010	⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮	-126	11111110	11111110	10000001	10000010
+126	01111110	01111110	01111110	01111110	-127	11111111	11111111	10000000	10000001
+127	+1111111	01111111	01111111	01111111	-128	-1000000	无法表示	无法表示	10000000
-0	-0000000	10000000	11111111	00000000					

三、带符号二进制数的运算

二进制数和十进制数的运算规则基本相同，所不同的仅仅前者逢 2 进 1，后者逢 10 进 1，借位时，从高位借 1 到低位，前者当 2，后者当 10。

对于不带符号的二进制数可以直接进行加、减，但应注意当两个不带符号二进制数相减时，不允许用小的数去减大的数，因为数值小的数减去数值大的数，差一定是负数，不带符号数的前提是没有符号，显然也不允许有负数，如果这样做，减的结果也必然是错误的。

对于带符号的二进制数，情况比较复杂，因为带符号二进制数有三种表示方法，其中反码用得较少，下面主要介绍原码与补码运算中应注意的问题。

原码运算时，首先要把符号与数值分开。例如两数相加，先要判断两数的符号，如果同号，可以做加法，如果异号，实际要做减法，减后的差作为两数之和，和数的符号与绝对值较大的数的符号相同；两数相减也是一样，也要先判断符号，然后决定是相加还是相减，还要根据两数的大小与符号决定两数之差的符号。

补码运算不存在符号与数值分开的问题，而且加法运算就一定是相加。减法运算就一定是相减，而不论两个数的正负。

设有 x、y 两个数，用补码表示如下：

$$x = 10011111\text{B}(-97\text{ 的补码})$$

$$y = 00001000\text{B}(+8\text{ 的补码})$$

若求 x、y 之和，可不用考虑两数的符号，直接相加，得出的和为 $x + y = 10100111\text{B}$（-89的补码），可见直接相加时，不论同号还是异号，都是直接相加，其结果必定是正确的。

若求 $x - y$，也可以直接相减，即

$$\begin{array}{rl} x = 10011111\text{B} & (-97\text{ 的补码}) \\ -y = 00001000\text{B} & (+8\text{ 的补码}) \\ \hline x - y = 10010111\text{B} & (-105\text{ 的补码}) \end{array}$$

若求 $y - x$，同样可以直接相减，即

$$\begin{array}{rl} y = 00001000\text{B} & (+8\text{ 的补码}) \\ -x = 10011111\text{B} & (-97\text{ 的补码}) \\ \hline y - x = 01101001\text{B} & (+105\text{ 的补码}) \end{array}$$

也就是说做减法时，不问两数符号如何，都是直接相减，其相减结果不论数值还是符号都将是正确的。

在上述 $y-x$ 算式中，最高位发生进位，在字长为 8 位的计算机中，若运算结果没有超出补码的记数容量（－128 ～ ＋127），只是因为符号位引起的进位，这时的进位被视为自然丢弃，在计算机运算中，这种自然丢弃是允许的，因为它不影响结果的正确。自然丢弃的特征是第 7 位和第 8 位同时产生进位，即所谓双进位。

但要注意，如果得数超过 8 位补码所允许表示范围（即超出 127 ～ －128），则其进位称之为溢出。溢出的特征是第 7 位和第 8 位只有一个位产生进位。溢出与自然丢弃显然不同。判别属于哪一种，则要看是双进位还是单进位，双进位属于自然丢弃，单进位则属于溢出，自然丢弃是允许的。溢出则不允许，因为产生溢出表示其数值超出计算机字长所能允许的范围，运算结果必然错误。

例 1-3　求下列两组数之和，其中一组为 ＋1100100B、＋1000011B，另一组为 －1111000、－0011000。

解：

$x=+1100100B \quad y=+1000011B$

$$\begin{array}{rr} (x)_{补}=01100100B & (+100)_{补} \\ +(y)_{补}=01000011B & +(+57)_{补} \\ \hline (x+y)_{补}=10100111B & (-89)_{补} \end{array}$$

$x=-1111000B \quad y=-0011000B$

$$\begin{array}{rr} (x)_{补}=10001000B & (-120)_{补} \\ +(x)_{补}=11101000B & +(-24)_{补} \\ \hline (x+y)_{补}=101110000B & (+112)_{补} \end{array}$$

例中第一组，只有第 7 位产生进位，第 8 位没有进位。在第二组中只有第 8 位产生进位，第 7 位没有进位。两者都是单进位，都属于溢出，表示它们的和超过 7 位补码所能表示的范围，不能用 7 位补码表示。运算答案 －89 和 ＋112 显然都是错误的。

“溢出”是带符号二进制数进行加减运算时因数值进位影响到符号位而产生的一种结果。对于不带符号数的第 8 位不是符号，所以不使用溢出这个概念。

综上所述，在原码运算中，如果将减法运算化为加法运算，例如将 $x-y$ 化为 $x+(-y)$，但因为原码运算要考虑两个数的符号，正数与负数相加实际还要做减法，所以这种转换没有什么实际意义。但如果这两个数是用补码表示，即 $(x)_{补}-(y)_{补}=(x)_{补}+(-y)_{补}$。由于补码运算时无须考虑两个数的符号，加法运算就是相加，真正把减法运算转化为加法运算，这就是为什么计算机转移指令中的偏移量计算采用补码的原因。

例 1-4　求 2070H 与 12H 的差。

解：一般运算时，两个数的位数最好要相同，因此运算前先将 12H 写成 16 位，也就是求 2070H 与 0012H 的差，写成算式为

$$\begin{array}{r} 2070H \\ -\ 0012H \\ \hline 205EH \end{array}$$

也可以转换为补码运算，转换为补码后可将减法运算转换为加法运算

$$(2070H)_{补} - (0012H)_{补} = (2070H)_{补} + (-0012H)_{补}$$

2070H 为正数，它的补码等于 2070H，0012 为负数，它的补码可将 10000000 00010010 取反加 1 等于 11111111 11101110(FFEEH)写成算式为

$$\begin{array}{r} 2070H \\ +\ FFEEH \\ \hline 205EH \end{array}$$

即把减法运算转化为加法运算后结果相同。

第三节 BCD 码及文字符号代码

一个二进制数，可以表示一个不带符号数，也可以表示一个带符号数，而且可以按照事先约定代表一个文字、一个符号，或者代表某个约定的内容。当用它代表文字或符号时称它为代码，例如 00100100B，如作为数，它是代表不带符号数 24H，或者是带符号数 +24H。如果事先约定，又可以表示符号$。所谓约定，可以按某个标准规定约定，也可以自行约定。下面介绍两种按标准约定的常用文字符号代码。

一、BCD 码(Binary-coded decimal)

BCD 码是一种约定以二进制形式表示十进制数的编码，又称二-十进制码，它貌似二进制，实际是十进制数。

BCD 码以 4 位为一组，选用 0000B ~ 1001B 的 10 种状态代表 0 ~ 9 共 10 个数，舍弃其余的 6 种状态。当 BCD 码与十进制数进行互换时，可以按 4 位为一组，逐组进行互换。

例 1-5 将十进制数 84.7 转换为 BCD 码。

解：8 4. 7 0

1000 0100. 0111 0000

例 1-6 将 BCD 码 10010100.01110010 转换为十进制数。

解：1001 0100. 0111 0010

9 4. 7 2

要将一个二进制数转换为 BCD 码，通常先将它转换为十进制数，然后再转换为 BCD 码。同样将 BCD 转换为二进制数，也是先转换成十进制数，再转换为二进制数。

例 1-7 将二进制数 11110011B 转换为 BCD 码。

解：

$11110011B = 1\times2^7 + 1\times2^6 + 1\times2^5 + 1\times2^4 + 0\times2^3 + 0\times2^2 + 1\times2^1 + 1\times2^0 = 243$

十进制数为 2 4 3

BCD 码为 0010 0100 0011

或按习惯写成两个字节即二进制数 11110011B 的 BCD 码等于$(0000001001000011)_{BCD}$

例 1-8 将 BCD 码$(10001001)_{BCD}$转换为二进制数。

1000 1001

十进制数为 8 9

二进制数 01011001B

BCD 码的低 4 位向高 4 位进位要遵循逢 10 进 1 的原则，这与二进制显然不同，二进制的低 4 位要向高 4 位进位，必须遵循逢 16 进 1。因此两个 BCD 码相加。先按二进制数的相加方法，但要以 4 位为一组，逐组相加，凡相加后的和大于 9 者，还应进行加 6 修正。

例 1-9　将 BCD 码$(01011000)_{BCD}$与$(01101001)_{BCD}$相加

解：

$$\begin{array}{rcc} & 0101 & 1000 \\ + & 0110 & 1001 \\ \hline & 1100 & 0001 \end{array}$$

进位↵

由于低 4 位和为 17，大于 9，除进位 1 外，余数还应加 6 修正。高 4 位和为 12，也大于 9，也应加 6 修正，即上式的和应予以修正如下：

	1100	0001	原得数	检验	58
+	0110	0110	高 4 位和低 4 位各加 6 修正		+69
0001	0010	0111	正确值		127

两个 BCD 码相减，凡低 4 位有向高 4 位借位者，都要进行减 6 修正，无借位的当然无需修正。

例 1-10　将 BCD 码$(10000101)_{BCD}$与$(00101000)_{BCD}$相减。

解：

$$\begin{array}{rcc} & 1000 & 0101 \\ - & 0010 & 1000 \\ \hline & 0101 & 1101 \end{array}$$

借位→

	0101	1101	原得数	检验	85
−		0110	低 4 位应减 6 修正		−28
	0101	0111	正确值		57

二、ASCII 码

ASCII 码是美国信息交换标准代码的简称，由 American Standard Code for Information Interchang的第一个字母组成。

计算机只能辨认或存储 0 和 1 两个数码，也就是计算机内部只能使用二进制数，但在编制计算机程序或信息时，还会碰到许多文字符号，这就需要把文字符号改成一串二进制代码，即把文字符号数码化，现在国际通用的文字符号代码是 ASCII。

ASCII 码共 128 个，用 00000000～01111111 代表，其中英文大小写字母共 52 个，0～9 数字 10 个，常用书写符号(!%等等)和常用运算符号(如 +、−、<、>等)32 个，另外有控制符号 34 个，共计 128 个。例如英文大写字母 A 的编码为 01000001，或写成十六进制 41H。数码 5 的编码为 00110101，或写成十六进制的 35H。这里 35H 代表一个 ASCII 码，可见 35H 可以根据约定赋予不同的物理意义。ASCII 码表见书后附录。

ASCII 码实际只占用一个字节的 7 位，余下的一个最高位可作为奇偶校验位。

例 1-11　数 00110011B 的含义。

解：作为不带符号数时表示 51，作为带符号数时表示 +51，作为 ASCII 码则表示数码 3。

第四节 单片机系统的组成

电子计算机由中央处理单元(简称 CPU)、存储器、输入输出接口及外围设备所组成。如果把中央处理单元集成在一小块芯片上，这种芯片就称为微处理器(Microprocesser)，由微处理器和相应存储器、输入输出接口所组成的计算机则称为微型计算机。如果再进一步把微处理器、存储器及某些输入输出接口也一起集成在一个芯片上，这种芯片则称之为单片机。可见微型计算机和单片机其组成都是一样的，只是结构不同而已。

单片机也有的称之为单片微型计算机(Single Chip Microcomputer)、微控制器(Micro Controller Unit,简称 MCU)。但“单片机”这种名称已经为国内大部分人所接受，所以本书统一称为“单片机”。

计算机除硬件外，还必须配上相应软件，加上软件后的计算机称为计算机系统。单片机也一样，单片机芯片虽然已经包含了微处理器、存储器及某些输入输出接口等硬件，但要运行也还需要软件，有时还需配置系统所需要而片内没有的外围设备。可见一个完整的单片机系统也是由硬件和软件所组成。

一、单片机系统的硬件

硬件是构成单片机系统的所有电子、机械和磁性的部件和设备，包括单片机本身以及在外围扩充的存储器、输入输出接口以及相关的外围设备。图 1-1 是单片机系统的硬件组成，其中点画线框内表示集成在单片机片内的部件。

图中 CPU 即中央处理单元的简称，它由运算器、控制器和片内时钟振荡器等单元电路组成。运算器用于完成算术运算、逻辑运算及移位操作等。控制器负责从程序中逐条取出指令，经对指令进行译码分析，然后发出执行命令。执行相应的操作并做好取出下一条指令的准备。

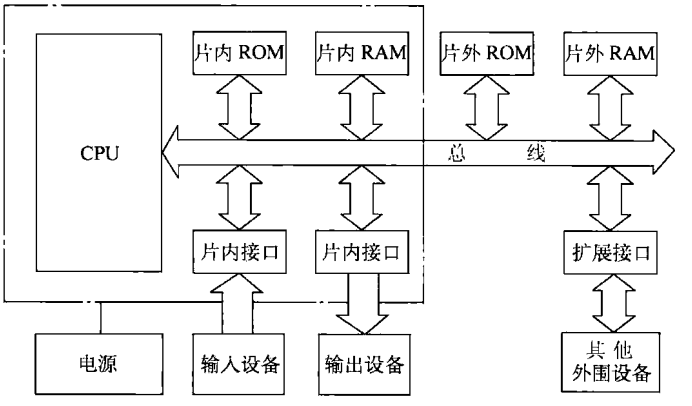


图 1-1 单片机系统的硬件组成

存储器是计算机的重要组成部分，它用来存储程序和数据。存储程序要用只读存储器 ROM(Read Only Memory,简称 ROM)，存储数据的要用随机存取存储器(Random Access Memory，简称 RAM)，如果片内存储器不够使用，还可以在片外扩展。

图中总线是作为系统各部件间的公共连接通道，所有部件都通过总线与 CPU 或其他部件相连接。单片机系统的总线包括数据总线(DB)、地址总线(AB)和控制总线(CB)，或称三总线，数据总线中数据流向由地址总线和控制总线的信号进行控制。为满足系统的需求，单片机要在外部扩展存储器和外设接口时，所需的三总线由单片机的引脚提供。

单片机系统的外围设备简称外设，常用的外设包括输入设备即用于人机联系的设备，例

如输入数据用的键盘、开关及各种传感器。也包括输出设备即显示运算结果用的数码管、显示器、微型打印机以及用于控制的设备，例如步进电动机、继电接触器、触点开关等等。

输入输出设备与主机的连接电路称为接口，简称 I/O 接口。接口是主机与外设之间的连接部件，设置它的目的一是为了实现外设与总线的隔离。因为众多的存储器和外围设备都接在总线上，某一时刻 CPU 又只能与一个存储单元或一个外围设备间传送信息，因此外设与总线就不能直接相通，而要通过一个 I/O 接口，这样，CPU 就可以在某一时刻通过地址总线选通一个接口，把该接口的外设与总线相连，其余外设接口处于阻断状态，以达到与总线隔离的目的，避免工作时的相互干扰。

隔离一般用三态门组成，三态门是一种可控的门电路，只有控制端使能时，输出端才受输入端控制。图 1-2a 是高电平控制的三态门，只有控制端为高电平时输出端才受输入端控制。与图 1-2a 相反，图 1-2b在控制端为低电平时输出端才受输入端控制，当输出端不受输入端控制时，其输出端呈高阻状态，相当于三态门将总线与外设隔离。由低电平控制的三态门真值表如表 1-2 所示。图 1-3 是由三态门构成的 I/O 接口。

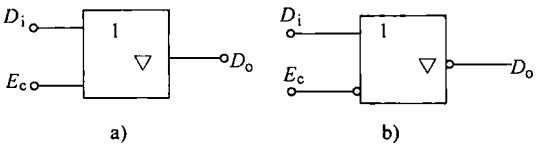


图 1-2　由三态门构成的 I/O 接口

a）高电平控制的三态门　b）低电平控制的三态门

表 1-2　由低电平控制的三态门真值表

输入端电平	控制端电平	输出端电平	输入端电平	控制端电平	输出端电平
0	1	高阻态	0	0	0
1	1	高阻态	1	0	1

接口除了隔离功能外，还有锁存或变换功能。锁存可以采用 D 触发器，因为 D 触发器一经触发，它的状态可以保持不变直到下一次触发为止。图 1-4 是用 D 触发器作为 I/O 接口的连接图。和三态门一样，它也是通过地址总线选通，被选通的锁存器，可以把 CPU 送来的数据暂时放在接口中，等待需要时再从接口送给外设。

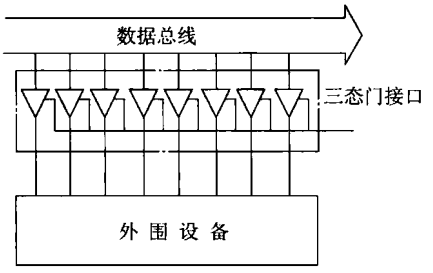


图 1-3　由三态门构成的 I/O 接口

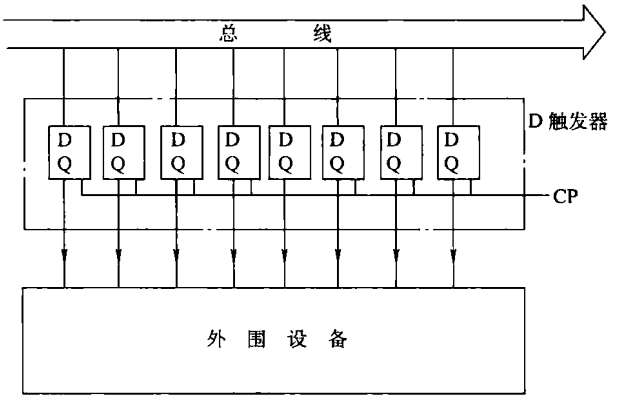


图 1-4　用 D 触发器作为 I/O 接口