中等专业（职业）学校教材

单片机原理与应用

潘永雄　刘　殊　编

西安电子科技大学出版社
http:// www.xduph. com

中等专业(职业)学校教材

单片机原理与应用

潘永雄　刘　殊　编

西安电子科技大学出版社
2000

内容简介

本书以 MCS－51 单片机的应用为主线，结合典型应用实例，系统地介绍了 MCS－51 系列单片机的内部结构、指令系统、资源扩展、接口技术、单片机应用系统的硬件结构、开发过程及手段。在编写过程中，尽量避免过多地介绍程序设计方法和技巧，着重介绍系统的硬件连接、调试技巧，注重典型性和代表性，以期达到举一反三的效果。内容力求新颖、全面、实用。

本书是单片机原理与应用的入门教材，可以作为大中专学校有关专业“单片机原理与应用”课程的教材或教学参考书，亦可供从事单片机技术开发、应用的工程技术人员阅读。

图书在版编目(CIP)数据

单片机原理与应用/潘永雄，刘殊编. —西安：西安电子科技大学出版社，2000.6
中等专业(职业)学校教材
ISBN 7－5606－0843－4
Ⅰ. 单… Ⅱ. ①潘…②刘… Ⅲ. 单片微型计算机-专业学校-教材 Ⅳ. TP368.1
中国版本图书馆 CIP 数据核字(2000)第 21141 号

责任编辑 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)8227828 邮 编 710071
http：//www.xduph.com E-mail：xdupfxb@pub.xaonline.com
经 销 新华书店
印 刷 西安电子科技大学印刷厂
版 次 2000 年 6 月第 1 版 2000 年 6 月第 1 次印刷
开 本 787 毫米×1092 毫米 1/16 印张 18.5
字 数 437 千字
印 数 1～4 000 册
定 价 17.00 元
ISBN 7－5606－0843－4/TP・0438

前　言

单片机技术作为计算机技术的一个分支，广泛应用于工业控制、智能化仪器仪表、家用电器，甚至电子玩具等各个领域，它具有体积小、功能多、价格低廉、使用方便、系统设计灵活等优点。因此，越来越受到工程技术人员的重视，目前国内大中专学校电子技术、自动控制、计算机硬件等专业均先后开设了单片机原理与应用课程。

本书以单片机在电子技术中的应用为主线，以需要掌握和使用单片机技术的工程技术人员、大中专学校有关专业学生作为主要的服务对象。从实用角度出发，力争用通俗易懂的语言，由浅入深，系统、详细地介绍 MCS－51 系列单片机系统的硬件结构、指令系统、程序设计方法、接口技术等方面的基本知识，然后结合典型应用实例介绍单片机应用系统的开发过程和手段。在编写过程中，尽量避免过多地介绍程序设计方法和技巧，而着重介绍系统的硬件连接、调试技巧；注重典型性和代表性，以期达到举一反三的效果。内容力求新颖、全面、实用。

本书共分 9 章，其中第 1 章介绍计算机系统的基本结构、工作原理等基础知识；第 2 章简要介绍了在单片机控制系统中常用的集成电路芯片功能、引脚排列、注意事项等器件基础知识，为学习后续章节内容做准备；第 3 章全面、系统地介绍了 MCS－51 单片机的内部结构、系统组成以及资源扩展方法，是本书的重点；第 4 章简要地介绍了 MCS－51 指令系统和单片机应用程序的结构和特点；第 5 章介绍 MCS－51 的中断系统；第 6 章介绍 MCS－51 定时/计数器和串行接口的功能和使用方法；第 7 章介绍开关信号的输入/输出接口电路；第 8 章介绍模拟信号的输入/输出电路；第 9 章介绍单片机应用系统的设计方法、技巧和注意事项。

本书第 1～4、7 章由潘永雄编写，第 5、6、8、9 章由刘殊编写。

本书可作为大中专学校有关专业“单片机原理与应用”课程的教材或教学参考书，也可作为需要掌握和使用单片机技术的工程技术人员的参考资料。

由于我们水平有限，书中不当之处，恳请读者批评、指正。

编者

2000 年 2 月

第1章 基础知识

为了便于理解单片机系统的工作原理及存储器容量的大小，也为了便于理解数字、字母等字符在单片机系统中的表示方法及处理过程，有必要先介绍有关数制和码制等方面的基本知识。

1.1 数 制

在日常生活中，十进制是人们最熟悉也是最习惯的计数方式，但十进制数需要0～9等10个数码表示，在计算机中显得很不方便。原因是计算机的电路基础是数字电路，而数字电路中的晶体管只有两个稳定状态，要么截止，要么饱和。截止时，集电极输出高电平，定义为“1”态，饱和时，集电极输出低电平，定义为“0”态；又如脉冲宽度内为“1”，脉冲间歇期为“0”等等。这与二进制非常类似，1位二进制数也有两个状态(0和1)，这就是计算机中用二进制计数表示、运算、存储的原因，其实二进制数是计算机系统能认识、处理的唯一数制。但由于二进制数不够直观，位数较长，不便记忆。向计算机输入数据时，往往直接输入人们习惯的十进制数，计算机将其转化为二进制后，再处理；另一方面，计算机的处理结果也要转换为人们习惯的十进制数。因此，在计算机中，需要进行各种数制之间的转换，下面我们就简要介绍有关这方面的基本知识。

1.1.1 二进制

二进制的特点是：只有两个数码，即0和1；逢二进1。

1位二进制数只能表示0和1两个状态，为了表示更多的状态，可用两位或两位以上的二进制数表示，二进制数的位数与它能表示的状态数之间的关系如下：

1位二进制数，共有2^1(即2)个状态，分别编码为0、1。

2位二进制数，共有2^2(即4)个状态，分别编码为00、01、10、11。

4位二进制数，共有2^4(即16)个状态，分别编码为

0000	0001	0010	0011
0100	0101	0110	0111
1000	1001	1010	1011
1100	1101	1110	1111

8位二进制数，共有2^8(即256)个状态，分别编码为

00000000　00000001　00000010　00000011
00000100　00000101　00000110　00000111
00001000　00001001　00001010　00001011
⋮
11111100　11111101　11111110　11111111

在计算机系统中所用的寄存器、存储器的本质就是一组触发器。一个触发器，如 RS、D 型触发器等均有两个稳定的状态，即 0 态和 1 态，显然一个触发器可以表示 1 位二进制数。为了提高数据处理速度，在计算机中往往需要并行处理多位二进制数。习惯上，存储器中一个存储单元通常由 8 个触发器组成，即一个存储单元可以存放一个 8 位二进制数。8 位二进制数称为一个字节(Byte)，有 256 种状态，或者说可以表示 256 个符号。因此，存储器(包括内存储器和外存储器)容量单位常用字节(或千字节)表示，如某存储器的容量为 640 KB，即该存储器有 640×1024 个存储单元，每个存储单元的大小为一个字节。

10 位二进制数，共有 2^{10}(即 1024，在计算机中，“1024”习惯上称为 1 K)个状态，编码从 0000000000～1111111111。

16 位二进制数，共有 2^{16}(65 536，即 64 K)个状态，编码从 0000000000000000～1111111111111111。

有些微处理器，如大多数 8 位微处理器就有 16 根地址线，由于每根地址线只有两种可能状态，地址线状态的不同编码，将寻址不同的存储单元。因此，16 根地址线相当于 16 位二进制数，最多可以寻址 64 KB 个存储单元。而存储单元的大小一般是一个字节，所以对于具有 16 根地址线的微处理器来说，最多可以寻址 64 KB 的存储空间，或者说寻址能力为 64 KB。

同理，对于 20 位二进制数，将有 2^{20}(1 048 576，即 1024 K，在计算机中习惯上称为 1 M)个状态，编码从 00000000000000000000～11111111111111111111。

某些微处理器，如 Intel 8088 CPU，就有 20 根地址线，因此该微处理器最多可以寻址 1 MB 的存储空间。

为了不致引起混乱，二进制数用后缀字母 B 作标记，如二进制数 1110 记为 1110B。

1.1.2　二进制数与十进制数之间的转换

众所周知，对于 n 位十进制数，可以表示为

$$A_{n-1}\times 10^{n-1}+A_{n-2}\times 10^{n-2}+\cdots+A_2\times 10^2+A_1\times 10+A_0+B_1\times 10^{-1}+B_2\times 10^{-2}+\cdots$$

例如，9876.54 可以表示为 $9\times10^3+8\times10^2+7\times10+6+5\times10^{-1}+4\times10^{-2}$。

同理，n 位二进制数，也可以表示为

$$A_{n-1}\times 2^{n-1}+A_{n-2}\times 2^{n-2}+\cdots+A_2\times 2^2+A_1\times 2+A_0+B_1\times 2^{-1}+B_2\times 2^{-2}+\cdots$$

例如，1101.01 可以表示为 $1\times2^3+1\times1^2+0\times2+1+0\times2^{-1}+1\times2^{-2}$，即十进制数 13.25。

可见，二进制数转换成十进制数不难，只要按上式展开即可求出对应的十进制数。而十进制数转换为二进制数时，可以按如下规律进行：

整数部分除以 2 所得的余数，就是对应二进制数的个位；其商再除以 2 所得的余数就

是对应二进制数的十位，依此类推，即可获得对应二进制数的整数部分。

小数部分乘以 2 所得的整数就是对应二进制数小数部分的十分位，小数部分再乘以 2 所得的整数就是对应二进制数小数部分的百分位，依此类推，即可求出所有的小数位。

1.1.3 十六进制

由于二进制数由一长串的 0、1 组成，位数太长，不便书写和记忆；另一方面，二进制数和十六进制数之间的换算非常方便、直观。为此，书写时常用十六进制数表示二进制数，但必须注意引入十六进制数的目的仅仅是为了便于书写和记忆，被计算机认识、处理的数据依然是二进制数，或者说二进制数是计算机系统中唯一存在的数制。

十六进制的特点是“逢十六进一”，具有 16 个数码，分别用 0、1、2、…、9 和 A、B、C、D、E、F 表示。

1 位十六进制数可以表示 16 种状态，编码从 0～F；2 位十六进制数可以表示 16^2(即 256)种状态，编码从 00～FF；4 位十六进制数可以表示 16^4(65536，即 64 K)种状态，编码从 0000～FFFF；而 8 位十六进制数可以表示 16^8(即 4096 M)种状态，编码从 00000000～FFFFFFFF。

可见十六进制数位数少，便于书写和记忆。

为了不致引起误解，十六进制数要加后缀字母 H，如十六进制数“3F”记为“3FH”；而对于以字母开头的十六进制数，必须带有前缀 0(零)，以示区别于一般字符串，如十六进制数 FE 记为“0FEH”。

与十进制数类似，对于 n 位 16 进制数，可以表示为

$$A_{n-1}\times 16^{n-1}+A_{n-2}\times 16^{n-2}+\cdots+A_2\times 16^2+A_1\times 16+A_0+B_1\times 16^{-1}+B_2\times 16^{-2}+\cdots$$

例如，98BF.5EH 可以表示为 $9\times16^3+8\times16^2+B\times16+F+5\times16^{-1}+E\times16^{-2}$，即 39 103.367 187 5。

1.1.4 二进制数与十六进制数之间的转换

我们知道二进制数可以表示为

$$A_{n-1}\times 2^{n-1}+A_{n-2}\times 2^{n-2}+\cdots+A_2\times 2^2+A_1\times 2+A_0+B_1\times 2^{-1}+B_2\times 2^{-2}+\cdots$$

如果在上式中，对于整数部分，从个位开始每 4 位分为一组；对于小数部分，从十分位开始，每 4 位分为一组，则上式可表示为

$$\begin{aligned}&A_{n-1}\times 2^{n-1}+A_{n-2}\times 2^{n-2}+\cdots+A_2\times 2^2+A_1\times 2+A_0+B_1\times 2^{-1}+B_2\times 2^{-2}+\cdots\\&=(A_{n-1}\times 2^3+A_{n-2}\times 2^2+A_{n-3}\times 2+A_{n-4})\times 2^{n-4}+(A_{n-5}\times 2^3+A_{n-6}\times 2^2+A_{n-7}\times 2^1+A_{n-8})\times 2^{n-8}+\cdots+(A_7\times 2^3+A_6\times 2^2+A_5\times 2^1+A_4)\times 2^4\\&+A_3\times 2^3+A_2\times 2^2+A_1\times 2^1+A_0+(B_0\times 2^3+B_1\times 2^2+B_2\times 2^1+B_3)\times 2^{-4}+\cdots+(B_{n-4}\times 2^3+B_{n-3}\times 2^2+B_{n-2}\times 2^1+B_{n-1})\times 2^{-(n-1)+4}\end{aligned}$$

上式与十六进制表示非常接近，括号内就是对应十六进制的数码，而 2^{n-4}就是对应位的权，如：

$$10101010B = (1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0) \times 2^4 + (1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0)$$
$$= A \times 2^4 + A$$
$$= A \times 16 + A$$

因此，

$$10101010B = 0AAH$$

由此可见：

(1) 二进制数转换成十六进制数时，按如下规则进行：

对于二进制数的整数部分来说，从个位开始，每 4 位作为一组，划分整数部分(如果最后一组不足 4 位，可在前面补 1～3 个零)；对于二进制数的小数部分来说，从十分位开始，每 4 位作为一组，划分小数部分(同样，当最后一组不足 4 位时，可在后面补 1～3 个零)。然后把每组中的 4 位二进制数用对应的十六进制数表示，即可获得相应的十六进制数。如：

1110010101.10101B
=0011 1001 0101.1010 1000
3 9 5 A 8

即

1110010101.10101B=395.A8H

(2) 十六进制数转换为二进制数时，按如下规则进行：

将十六进制数的整数部分和小数部分的每 1 位十六进制数码用对应的 4 位二进制数表示，然后再删除整数部分前面和小数部分后面多余的零，即可获得对应的二进制数，如：

93FE.3A3H=1001 0011 1111 1110.0011 1010 0011B

又如：3E.CH =0011 1110.1100B
=111110.11B(删除整数部分前面的零和小数部分后面的零)

可见，二进制数与十六进制数之间的转换非常方便，只要记住 4 位二进制数 0000～1111 与十六进制数 0～F 之间的对应关系即可。下面是二进制数 0000～1111 对应的十六进制数和十进制数。

二进制	十进制	十六进制	二进制	十进制	十六进制
0000	0	0	1000	8	8
0001	1	1	1001	9	9
0010	2	2	1010	10	A
0011	3	3	1011	11	B
0100	4	4	1100	12	C
0101	5	5	1101	13	D
0110	6	6	1110	14	E
0111	7	7	1111	15	F

由此可见，用十六进制数表示二进制数是非常方便的，例如一个字节，当用十六进制数表示时，256 种状态的编码分别为

00H，01H，02H，...，0F0H，0F1H，0F3，...，0FEH，0FFH

又如 16 位二进制数(即两个字节)用十六进制数表示时，64 K 种状态编码分别为

0000H，0001H，0002H，...，0FFF0H，0FFF1H，...，0FFFEH，0FFFFH

不仅位数少了，也方便记忆。

1.1.5 二进制数和十六进制数的运算

1. 二进制数的运算

二进制数的四则运算包括加、减、乘、除，在学习单片机原理时，尤其需要掌握其中的加、减和乘法运算规则。

1）二进制数的加法

二进制数的加法运算规则为：0+0=0；0+1=1；1+0=1(交换律)；1+1=10(二进制中的“10”就是十进制的“2”)，例如：

$$\begin{array}{r} 1001\ 0110B \\ +\ 0111\ 0011B \\ \hline 1000\ 1001B \end{array}$$

2）二进制数的减法

二进制数向前借位时，为“10”，例如：

$$\begin{array}{r} 1001\ 0110B \\ -\ 0111\ 0011B \\ \hline 0010\ 0011B \end{array}$$

3）二进制数的乘法

二进制数的乘法运算规则为：0×0=0；0×1=1；1×0=1(交换律)；1×1=1，例如：

$$\begin{array}{r} 1001\ 0110B \\ \times\quad 101B \\ \hline 10010110 \\ 00000000 \\ +\ 10010110 \\ \hline 1011101110B \end{array}$$

2. 十六进制数的运算

十六进制数的四则运算包括加、减、乘、除，在学习单片机原理时，同样需要掌握其中的加、减和乘法运算规则。

十六进制数的加法运算规则与十进制数的加法运算规则相同，如 3H+4H=7H；7H+4H=0BH（结果是十进制数的 11，即十六进制数的 0BH)；8H+7H=0FH(结果是十进制数的 15，即十六进制数的 0FH)；8H+9H=11H(结果是十进制数的 17，即十六进制数的 11H)，例如：

$$\begin{array}{r} 346AH \\ +\ 589CH \\ \hline 8D06H \end{array}$$

在上式中，个位 A(即 10)加 C(即 12)，结果为 22，即十六进制数的“16”，向十位进 1，结果为 6；十位 6+9+1(个位进位得到)，结果为 16，即十六进制数中的“10”，向百位进 1，

结果为 0；百位 4＋8＋1(十位进位得到)，结果为 13，即十六进制数中的“D”；千位 3＋5，结果为 8。

$$\begin{array}{r} 7'4'6'A\ H \\ -\ 5\ \ 8\ \ 9\ C\ H \\ \hline 1\ \ B\ \ C\ E\ H \end{array}$$

在上式中，个位向十位借 1 后，变成十六进制数的“1A”，再减 C，即 26 减 12(即 0CH)，结果为 14，即十六进制数的“E”；十位原本是“6”，个位借后变为“5”，十位向百位借 1，变成十六进制数的“15”，即 21，减 9，结果为 12，即十六进制数的“C”；百位原本是“4”，十位借后变为“3”，百位向千位借 1，变成十六进制数的“13”，即 19，减 8，结果为 11，即十六进制数的“B”。

$$\begin{array}{r} 7\ 4\ 6\ A\ H \\ \times\ \ \ \ \ 9\ C\ H \\ \text{(进位)}\ \ 3\ 4\ 7\ \ \ \ \\ \hline 5\ 7\ 4\ F\ 8 \\ \text{(进位)}\ \ 2\ 3\ 5\ \ \ \ \ \\ +\ \ \ 4\ 1\ 7\ B\ A\ \ \ \\ \hline 4\ 6\ F\ 0\ 9\ 8 \end{array}$$

可见，十六进制数的乘法与十进制数的乘法运算方法类似，但必须注意将十六进制乘法运算的中间结果转为十六进制数，例如 6×9，结果为十进制数的 54，应转化为十六进制数 36H。

1.2 码　　制

在这一节中，主要介绍本课程常用到的 ASCII 码，以及原码、反码、补码，十进制数的二进制表示法——BCD 码等方面的基本知识。

由于计算机内部所有的数据均采用二进制代码表示，但通过输入设备(如键盘)输入的信息和通过输出设备(如显示器、打印机)输出的信息却是多种多样的，既有字母、数字，又有各种控制字符，甚至汉字。为了方便，人们对计算机中常用的符号进行编码，当通过输入设备向计算机输入某个字符时，计算机会自动将该字符转化为对应的二进制数，再进行处理，同时计算机也将处理结果还原为对应的字符。于是，字符所对应的二进制数就称为该字符的编码。

1.2.1 英文字符的表示方法——ASCII 码

由于计算机只能处理二进制数，因此除了数值本身需要用二进制数形式表示外，字符，包括数码(如 0，1，2，3，4，5，6，7，8，9)、字母(如 A，B，C，D，...，X，Y，Z 及 a，b，c，d，...，x，y，z)、特殊符号(如%，!，＋，－，＝等)也必须用二进制数表示，即在计算机中需将数码、字母、特殊符号等代码化，以便于计算机识别、存储和处理。

英文属于典型的拼音文字，由字母、数字、特殊符号等组合而成，但这些字母、数字、特殊符号的数目毕竟有限，不过百余个。我们知道 7 位二进制数可以表示 128 种状态，如

果每一种状态代表某个特定的字母或数字，则 7 位二进制数可表示 128 个字符。

例如，可用 0110000B 表示数字 0，0110001B 表示数字 1，1000001B 表示大写字母 A 等。但这种编码方式并不唯一，如用 0110000B 表示数字 A，0110001B 表示数字 B，1000001B 表示数字 0 也未尝不可。为了便于不同计算机系统和不同操作者之间的信息交换，看来有必要规范字母与 7 位二进制数之间的对应关系。目前在计算机系统中普遍采用美国标准信息交换代码(American Standard Code for Information Interchange II，简称 ASCII 码)。

该标准用 7 位二进制数表示一个字符，最多可以表示 128 个字符，编码与字符之间的对应关系如附录 A 所示。

在计算机系统中，存储单元的长度通常为 8 位二进制数(即一个字节)，为了存取方便，规定一个存储单元存放一个 ASCII 码，其中低 7 位表示字母本身的编码，第 8 位(bit7)用作奇偶校验位或规定为零(通常如此)。因此，也可以认为 ASCII 码的长度为 8 位(但 bit7 为 0)。128 个字符对于某些特殊应用来说，可能不够，因此就采用 8 位的 ASCII，即扩展 ASCII 码(共有 256 个代码)。其中前 128 个(高位为 0)编码用于表示基本的 ASCII 码，基本 ASCII 码主要用于表示数字、英文字母(大、小写)、标点符号、控制字符等；后 128(高位为 1)个编码用于表示扩展的 ASCII 码，扩展 ASCII 用于表示一些特殊的符号，如希腊字母等。

1.2.2 BCD 码(二进制编码的十进制数)

十进制毕竟是人们最习惯的记数方式，在向计算机输入数据时，常用十进制数输入，但计算机毕竟只认识二进制数，因此每一位十进制数必须用二进制数表示。1 位十进制数包含 0～9 这 10 个数码，必须用 4 位二进制数表示。这样就需要确定 0～9 与 4 位二进制数 0000B～1111B 之间的对应关系，其中较常用的 8421 BCD 码与十进制数 0～9 与 4 位二进制数编码之间的对应关系如下：

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

注：在 BCD 码中，不使用 1010(0AH)～1111(0FH)。

例如，4567.89 的 BCD 码为：0100 0101 0110 0111.1000 1001(每一位十进制用相应的 4 位二进制数表示即可)。

尽管 BCD 码比较直观，但 BCD 码与二进制数之间的转换并不方便，需要转成十进制数后，才能转换为二进制数，反之亦然。

1.2.3 计算机中带符号数的表示方法

在计算机中，对于带符号数来说，一般用最高位表示数的正、负。对于正数，最高位定义为“0”，对于负数，最高位为“1”，例如：56H 可以表示 0 1010110(对于 8 位二进制数来

说，b7 位表示数的正负，b6～b0 表示数的绝对值)；－56H 可以表示 1 1010110。

0256H 可以表示 0 000 0010 0101 0110(对于 16 位二进制数来说，b15 位表示数的正负，b14～b0 表示数的绝对值)；－0256H 可以表示 1 000 0010 0101 0110。

1. 原码

对于带符号数来说，用最高位表示带符号数的正负，其余各位表示该数的绝对值，这种表示方法称为原码表示法，如上所示。

2. 反码

带符号数也可以用反码表示，反码与原码的关系是：

正数的反码与原码相同，如$[56H]_{反}=[56H]_{原}=0\ 1010110B$。

负数的反码等于对应正数的原码按位求反。因此，求－56H 反码的过程如下：

对应正数 56H 的原码为 0 1010110；按位求反后为 1 0101001，即－56H 的反码为：10101001。或者说，正数的反码与原码相同，而负数的反码是对应负数原码除符号位外，绝对值部分逐位取反。

3. 补码

在计算机内，带符号数并不是用原码或反码表示的，而是用补码表示。引入原码、反码的目的是为了方便理解补码概念。不能用原码表示的原因是：用原码表示时，“0”的原码并不唯一，0 的原码可以表示为 0 0000000(即＋0)，也可以表示为 1 0000000(即－0)，这会造成混乱；再就是用原码表示时，减法并不能转化为加法运算，反码也存在类似问题。在计算机中，带符号数用补码表示后，减法可以转化为加法运算，例如：

56H－23H ＝56H－23H＋100H（100H 是 8 位二进制数能表示的最大数，加上 100H
＝56H＋100H－23H　后，对计算结果没有影响，原因是 8 位二进制数无法
＝56H＋0DDH　　存放 100H 中的“1”，即 b8 位）
＝1 33H
＝33H（由于 8 位二进制数不能存放 b8 位，结果 133H 中最高位“1”自然丢失）

显然，56H－23H 的结果与 56H＋0DDH 相同，即引入补码后，减法可以用加法来完成。可见，在 8 位二进制数中，23H 与 0DDH 互为补码。

补码的定义如下：

正数的补码与反码、原码相同；

负数的补码等于它的反码加 1。因此，求－23H 补码的过程如下：

对应正数 23H 的原码为 0 0100011；按位求反后为 1 1011100，即－23H 的反码；反码加 1 后为 1 1011101，即－23H 的补码为 1 1011101(相当于无符号数的 0DDH)。

可见，用补码表示时，最高位为 0 时，表示该数为正数，数值部分就是真值；而最高位为 1 时，为负数，数值部分并不是它的真值，必须再求补后，才得到该数的绝对值，如上例中的－23H 的补码为 1 1011101，按位取反后为 00100010，加 1 后为 00100011，即 23H。

对于 8 位二进制数来说，补码表示的范围是－128～＋127；对于 16 位二进制数来说，补码表示的范围是－32 768～＋32 767。

例如，求 16 位二进制数中－23H 的补码过程如下：

对应正数 23H 的原码为 0 000 0000 0010 0011，按位取反后为 1 111 1111 1101 1100；加 1 后为 1 111 1111 1101 1101(即 16 位二进制－23H 的补码，相当于无符号数的 0FFDDH)。可见，对于同一个数，作为 8 位二进制数的补码和作为 16 位二进制数的补码不同。这一点要特别留意。

1.3 计算机的基本认识

为了理解计算机的硬件组成、工作原理及过程，我们先来看用算盘计算如下代数式的过程：

$$12\times34+56\div7-8=$$

首先要有算盘作为计算工具，在计算机里用“运算器”(简称为 ALU，即算术逻辑运算单元)作为计算工具，由它承担算术运算和逻辑运算。因为在计算机里，除了需要进行加、减、乘、除四则运算外，还需要进行逻辑与、或、非以及异或等逻辑运算。其次需要纸和笔记录算式、中间结果及最终结果。在计算机中，起到纸和笔作用的器件是存储器和寄存器(寄存器在 CPU 内，存取速度快，但数量少，仅适用于存放中间结果，而存储器由成千上万个存储单元组成，与寄存器相比，存取速度慢一些，常用于存放数据、运算指令等)。在计算上述算式时，先计算 12×34，并把中间结果记录下来；然后再计算 56÷7，再记录中间结果；把上述两步中间结果相加，并记录下来；再减 8。以上计算步骤由人脑控制，如果用计算机计算，则由计算机内的控制器完成，控制器在时钟信号的控制下，从存储器中取出计算步骤(即指令)和数据，并根据指令操作码内容发出相应的控制信号。此外，为了向计算机输入数据、指令，还需要输入设备，如键盘；同时，为了输出处理结果或显示机器的状态，还需要输出设备，如显示器。因此，一个计算机系统的基本结构如图 1－1 所示。

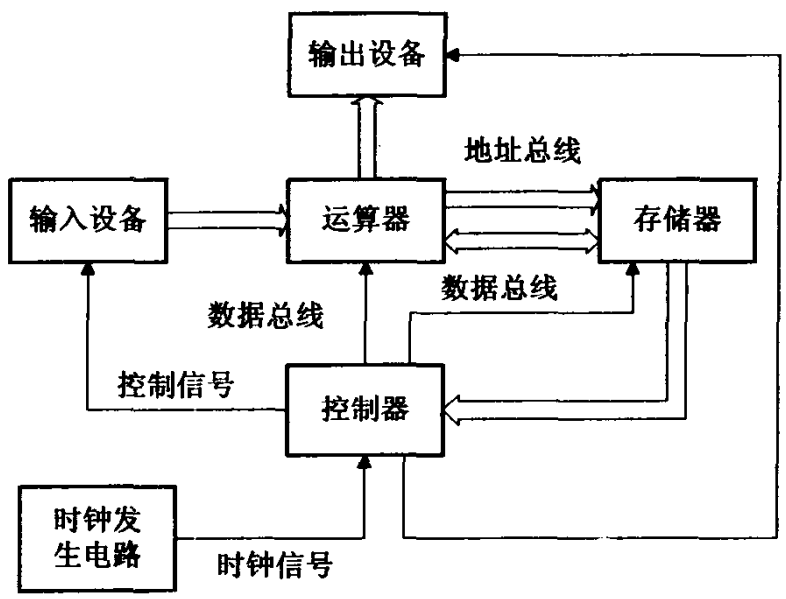


图 1－1　计算机基本结构

在计算机中，往往把运算器、控制器做在一起，称为中央处理器(Central Processor Unit，简称 CPU)，有时也称为微处理器(Micro Processor Unit，即 MPU)。为了进一步减小电路板面积，提高系统可靠性，将输入、输出接口电路、时钟电路以及存储器、运算器、控

制器等部件集成在一个CPU芯片内，就成为单片机。其含义是一个芯片就具备了一个完整计算机系统所必须具备的基本部件。为了适应不同的需要，将不同功能的外围电路集成在CPU内，形成系列化产品，就构成了所谓“嵌入式”单片机控制芯片。

1. 总线概念

在计算机系统中，常包含以下几种总线：

(1) 地址总线(Address Bus，简称AB)，一般为单向，用于传送地址信息，如图1-1中运算器与存储器之间的地址线。地址线的数目决定了可以寻址的存储空间。一根地址线有两个状态，即可以区分两个不同的存储单元，或者说可以寻址两个存储单元；两根地址线有4个状态，可以寻址4个存储单元；……；8位微处理器通常有16根地址线，可以寻址2^{16}，即64 K个存储单元，一般存储单元的大小为一个字节，因此8位微机的寻址范围为64 KB。

(2) 数据总线(Data Bus，简称DB)，一般为双向，用于CPU与外设，或外设与外设之间传送数据(包括实际意义的数据和指令码)信息。在计算机中，为了提高处理速度，总是一次处理由多位二进制数组成的信息。在运算器中，数据线的数目应与待处理的数据位数相同。因此，运算器数据线的数目往往不止一条，一般为4条、8条或16条。运算器数据线的多少称为微机的“字长”。字长是衡量微处理器功能、运算速度以及精度的重要指标之一，也是划分微处理器档次的重要依据。根据字长，可以将微处理器分为1位机、4位机、8位机、16位机、32位机、64位机等。1位机的运算器只有1根数据线，每次只能处理1位二进制数，工业上常用于取代继电器，控制线路的通和断、设备的开和关。4位机有4根数据线，常用于家用电器，如电视机、空调机、洗衣机等的控制电路中。8位机功能强大，不仅可用于工业控制、家用电器，也可以作为通用微机系统的中央处理器。

(3) 控制总线(Control Bus，简称CB)，是计算机系统中所有控制信号线的总称，在控制总线中传送的信息是控制信息。

2. 时钟周期、机器周期及指令周期

(1) 时钟周期：计算机在时钟信号的作用下，以节拍方式工作。因此，必须有一个时钟发生器电路，输入微处理器的时钟信号的周期称为时钟周期。

(2) 机器周期：机器完成一个动作所需要的时间称为机器周期，一般由一个或一个以上的时钟周期组成，例如在MCS-51系列单片机中，一个机器周期由12个时钟周期组成。

(3) 指令周期：执行一条指令(如“MOV A，#34H”，该指令的含义是将立即数34H传送到微处理器内的累加器A中)所需时间称为指令周期，它由一个到数个机器周期组成。指令周期的长短取决于指令的类型，即指令将要进行的操作步骤及复杂程度：简单指令，如INC A(累加器A中内容加1)可能只需要一个机器周期，而复杂指令，如MUL AB(累加器A乘以寄存器B，并将结果放在寄存器B和累加器A中)将需要多个机器周期。

1.3.1 计算机的工作过程及其内部结构

1. CPU的内部结构

图1-1中的运算器和控制器等部件往往做在同一芯片内，称为中央处理器(CPU)，8

位通用微处理器内部基本结构可用图 1－2 描述，它由算术逻辑运算单元(Arithmetic Logic Unit，简称 ALU)、累加器 A(8 位)、寄存器 B(8 位)、程序状态字寄存器 PSW(8 位)、程序计数器 PC(有时也称为指令指针，16 位)、地址寄存器 AR(16 位)、数据寄存器 DR(8 位)、指令寄存器 IR(8 位)、指令译码器 ID、控制器等部分组成，其中：

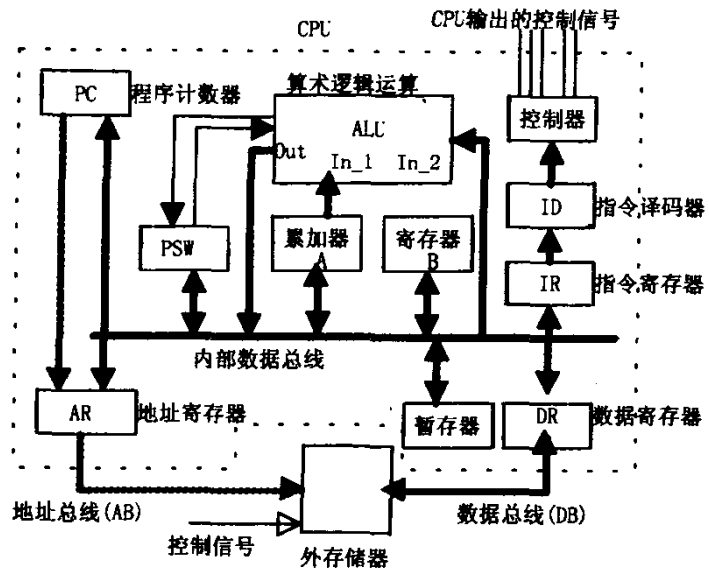


图 1－2　CPU 的内部结构简图

(1) 程序计数器 PC 是 CPU 内部的寄存器，用于记录将要执行的指令代码所在存储单元的地址编码。一般说来，PC 长度与 CPU 地址线引脚数目一致，例如 8 位微机的 CPU 一般具有 16 根地址线(A15～A0)，PC 的长度也是 16 位。复位后，PC 具有确定值，例如在 MCS－51 系列单片机中，复位后，PC＝0000H，即复位后将从程序存储器的 0000H 单元读取第一条指令码。由于复位后，PC 的值就是第一条指令代码存放的单元，因此程序设计时，必须了解复位后 PC 的值是什么，以便确定第一条指令从存储器哪一存储单元开始存放。PC 具有自动加 1 的功能，即从存储器中读出或写入一个字节后，PC 会自动加 1 指向下一存储单元。

(2) 地址寄存器 AR(Address Register，16 位)用于存放将要寻址的外部存储器单元的地址信息，对于指令码所在地址，由程序计数器 PC 产生；而指令中操作数所在存储单元的地址码，由指令的操作数给定。地址寄存器 AR 通过地址总线 AB 与外部存储器相连。

(3) 指令寄存器 IR(Instruction Register)用于存放取指阶段读出的指令代码的第一字节，即操作码，使指令译码器 ID 的输入保持不变，存放在 IR 中的指令码经指令译码器 ID 译码后，输入控制器，产生相应的控制信号，以完成指令规定的动作。

(4) 数据寄存器 DR 用于存放写入外部存储器或 I/O 端口的数据信息，可见，数据寄存器 DR 对输出数据具有锁存功能，数据寄存器与外部数据总线 DB 直接相连。

(5) 算术逻辑运算单元 ALU 主要用于算术(加、减、乘、除)、逻辑(与、或、非以及异或)运算，由于 ALU 内部没有寄存器，参加运算的操作数，必须放在累加器 A 中，同时也用于存放运算结果，例如执行

```
ADD A，B ；A←A＋B
```

指令时，累加器 A 中的内容通过输入口 In_1 输入 ALU，寄存器 B 通过内部数据总线经输入口 In_2 输入 ALU，A＋B 的结果通过 ALU 的输出口 Out、内部数据总线，再返回累加器 A。

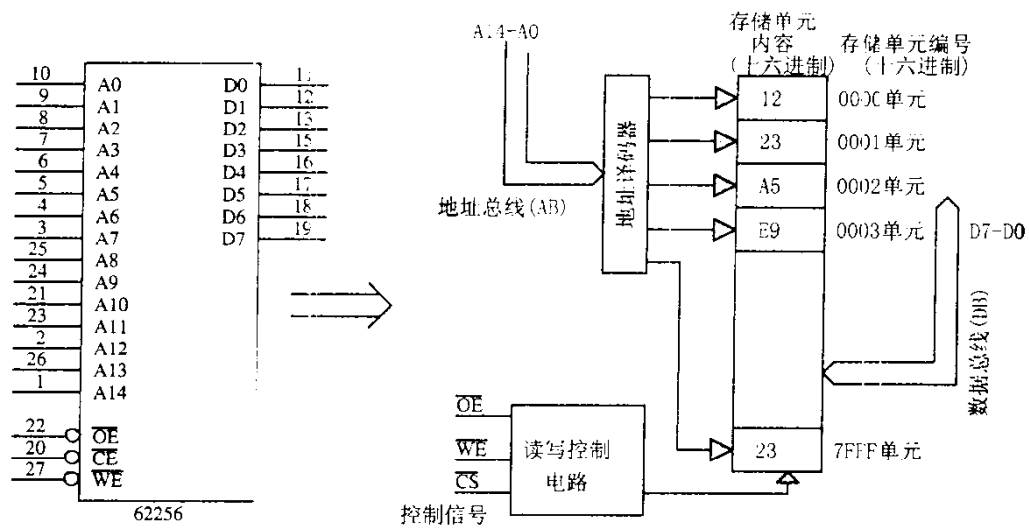
程序状态字寄存器 PSW 用于记录运算过程中的状态，如是否溢出、进位等。

2. 存储器

存储器是计算机系统中必不可少的存储设备，主要用于存放程序(指令)和数据。尽管寄存器和存储器均用于存储信息，但 CPU 内的寄存器数量少，存取速度快，主要用于临时存放参加运算的操作数和中间结果；而存储器一般在 CPU 外(但单片机 CPU 例外，其内部一般含有一定容量的存储器)，单独封装。在存储器芯片内，存储单元数目多，从几千字节到数百兆字节，能存放大量的信息，但存取速度比 CPU 内部的寄存器要慢得多。

1) 内部结构

存储器内部结构可以用图 1 - 3 描述，由地址译码器、存储单元、读写控制电路等部分组成。



存储器芯片　　内部结构示意图

图 1 - 3　存储器芯片及内部结构

寄存器或存储器中的一个存储单元，等效于一组触发器，每个触发器有两个稳定状态，可以记录一位二进制数。存储单元包含的触发器个数称为存储单元的字长，对于并行存取的存储器芯片，存储单元内包含的触发器的个数与存储器芯片数据线相同。

例如由 8 个触发器并排在一起构成的存储单元的字长为 8 位，它可以存放一个 8 位二进制数(一个字节)。在计算机中，为了提高处理速度，一次操作(如数据传送或运算)要同时处理多位二进制数。因此，在并行存取的存储器芯片中，一个存储单元的容量通常为 8 位。

存储器芯片内存储单元数目与存储器芯片地址线条数有关。例如，图 1 - 3 中的 HM62256 随机读写静态存储器芯片含有 15 根地址线(A14～A0)，可以寻址 2^{15} 共 32 K 个存储单元。为了便于存取，给每个存储单元编号(存储器地址线状态编码就是存储单元的编码)，图 1 - 3 中的 32 K 个存储单元的地址编码从 0000H～7FFFH。由于该芯片每个存储单元可以容纳一个 8 位二进制数，即该芯片存储容量为 32 KB。

但存储单元长度也可以大于或小于 8 位，例如 PIC16C56 单片机内的程序存储器为 1 K×12 位，即共有 1024 个存储单元，每个存储单元可以存放 12 位二进制数，即存储单元的字长为 12 位。

存储单元地址编码与存储单元中的内容是两个不同的概念，存储单元地址编码的长度由存储器芯片所包含的存储单元的个数决定。例如，6264 存储器芯片含有 8 K 个(13 根地

址线)存储单元，地址编码从0000000000000B～1111111111111B(用十六进制表示时，地址编码为0000H～1FFFH)，每个存储单元长度为8位。因此，每个存储单元的内容可以是00H～FFH之间的二进制数。又如，图1－3中的62256存储器含有32 K个(15根地址线)存储单元，地址编码从000000000000000B～111111111111111B(用十六进制表示时，地址编码为0000H～7FFFH)，每个存储单元的长度也是8位，图中0000H单元内容为12H，0001H单元内容为23H，而0002H单元内容为0A5H。而PIC16C56单片机程序存储器容量为1 K×12 bit，因此存储单元地址编码从0000000000B～1111111111B(用十六进制表示时，地址编码为000H～3FFH)，每个存储单元长度为12位。因此，每个存储单元的内容可以是000H～FFFH之间的二进制数。

根据存储器能否随机读写，将存储器分为两大类：只读存储器(Read Only Memory，简称ROM)和随机读写存储器(Random Access Memory，简称RAM)。根据存储器存储单元结构和信息保存方式的不同，又可以将随机读写存储器分为静态RAM(采用双极型晶体管结构，存取速度快，无需刷新，但构成一个存储单元所需的晶体管数目较多，集成度低，价格略高)和动态RAM(采用CMOS工艺，依靠MOS管栅极与衬底之间寄生电容保存信息，一般均为单管结构，集成度高，但寄生电容漏电大，容量小，信息保存时间仅为毫秒级，需要刷新电路，造成动态RAM存储器系统电路复杂，不适用于仅需要少量存储容量的单片机系统)。

在单片机系统中，所需的存储器容量不大，外围电路要尽可能简单，因此很少使用动态RAM，除非需要几百KB～几MB的存储容量。

只读存储器中“只读”的含义是信息写入后，只能读出，不能随机修改，适合存放系统监控程序。目前在单片机系统中，广泛采用紫外光可擦写的只读存储器(简称UN－EPROM)、一次性编程的OTP EPROM(内部结构与EPROM存储器相同，只有没有擦除窗，因此信息写入后，不能再删除，因此称为一次性编程的只读存储器)、电可擦写的EEPROM(结构与EPROM类似，但绝缘栅很薄，高速电子可穿越绝缘层，中和浮栅上的正电荷，起到擦除目的，即可通过高电压擦除)、Flash ROM(类似于EEPROM，但擦除速度快)等。

2) 存储器读操作

下面以CPU读取存储器中地址编号为0000H的存储单元内容为例，说明CPU读存储器中某一存储单元信息的操作过程(如图1－4所示)：

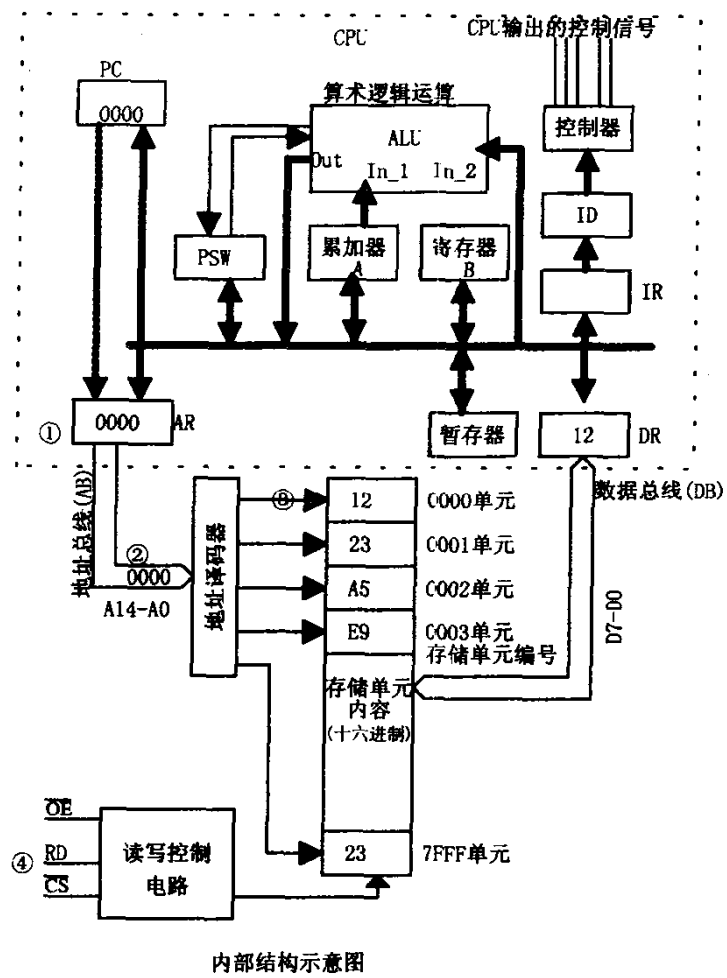


图1－4　CPU读取存储器操作过程示意图

(1) CPU地址寄存器AR给出将要读取的存储单元地址信息，即0000H；