中等职业教育国家规划教材（电子与信息技术专业）

单片机原理与应用

（第 2 版）

潘永雄　主编

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

内 容 简 介

本书以教育部下发的中等职业学校电子信息类专业“单片机原理与应用”课程教学大纲为依据，以增强型 MCS-51 单片机原理及其应用为主线，系统地介绍了增强型 MCS-51 系列单片机（包括 8XC5X、8XC5XX2、89C51RX、89C6XX2）的内部结构、指令系统、资源及扩展、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。

本书还配有教学指南、电子教案及习题解答（电子版），以方便教师教学使用。

本书是学习单片机原理与应用的入门教材，可作为中等职业学校电子信息类专业“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／潘永雄主编. —2 版 —北京：电子工业出版社，2005.1
中等职业教育国家规划教材·电子与信息技术专业
ISBN 7-121-00540-9

Ⅰ. 单… Ⅱ. 潘… 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（2004）第 113305 号

责任编辑：李 影 关雅莉
印 刷：
出版发行：电子工业出版社
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：17.75 字数：451.2 千字
印 次：2005 年 1 月第 1 次印刷
印 数：3 000 册 定价：22.40 元

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系。联系电话：（010）68279077。质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

前　言

单片机技术作为计算机技术的一个重要分支，广泛地应用于工业控制、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。目前国内中等职业学校电子技术、电力技术、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。

《单片机原理与应用》（中等职业教育国家规划教材）第 1 版于 2002 年 6 月出版，主要以标准 MCS-51 单片机芯片作为介绍对象，而现在标准 MCS-51 单片机芯片已停产，内容相对陈旧。几年来单片机技术发展非常迅速，原教材内容已不能很好地体现单片机技术现状及未来发展趋势。第 2 版在不违背“单片机原理与应用”教学大纲的前提下，根据单片机技术发展的趋势、中等职业学校电子与信息技术类专业培养目标、学生接受能力，以及使用本教材第 1 版任课教师的意见和建议，对第 1 版中的第 1～5 章和第 7 章的内容进行了全面修改，删除了第 1 版中“第 6 章 AD、DA 转换器”内容（原因是 AD、DA 转换器工作原理在“数字电路”课程中已介绍过，且目前各系列单片机芯片均有集成 AD、DA 转换器品种）。在编写过程中力求体现职业教育教材的特色，同时紧跟科技步伐，书中着重介绍了增强型 MCS-51 单片机应用系统中的硬件构成和连接、系统调试及维护技术，内容新颖、全面、实用。

本书共分 7 章，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 章全面、系统地介绍增强型 MCS-51 单片机的内部结构、系统资源、资源扩展方法，是本书的重点；第 3 章简要介绍 MCS-51 单片机指令系统、单片机应用程序结构、特点，以及典型子程序；第 4 章介绍 MCS-51 单片机中断控制器、定时／计数器、串行接口电路的功能和使用方法；第 5 章介绍增强 MCS-51 单片机内核衍生芯片（包括 8XC51RX 系列、P89C6XX2 系列）；第 6 章介绍输入/输出接口电路；第 7 章介绍单片机应用系统设计方法、技巧和注意事项。

考虑到职业教育教材的特点，在附录 A 中安排了较为详细的、可操作性强的实验内容。

本书由潘永雄雄老师主编，李晓苓、刘殊、李溪冰、刘向阳和潘景谦老师也参加了本书的编写工作。

本书可作为中等职业学校有关专业“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。

本书还配有教学指南，电子教案及习题答案（电子版），请有此需要的教师登录华信教育资源网（http://www.hxedu.com.cn）或与电子工业出版社联系，我们将免费提供。

E-mail:ve@phei.com.cn

由于我们水平有限，书中不当之处恳请读者批评、指正。

编　者

2004 年 3 月

目　录

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的，也是最习惯的计数方式，但十进制数需要用 0～9 十个数码表示，在计算机中使用显得很不方便。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：截止和饱和。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。这与二进制数非常类似，一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，即二进制数是计算机系统能认识、处理的惟一数制。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数后，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　二进制数

二进制数的特点是：只有两个数码，即 0 和 1，逢二进一。

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，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存储 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。一个 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 640KB，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}（1024，在计算机中，“1024”习惯上称为 1K）个状态，编码为 0000000000～1111111111。

16 位二进制数，共有 2^{16}（65536，即 64K）个状态，编码为 0000000000000000～1111111111111111。

有些微处理器，如大多数 8 位的微处理器，就有 16 根地址线。由于每根地址线有两种可能的状态，所以可以用地址线状态的不同编码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 64K 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 64KB 的存储空间，或者说寻址能力为 64KB。

同样，对于 20 位二进制数，将有 2^{20}（1048576，即 1024K，在计算机中习惯上称为 1M）个状态，编码为 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 1MB 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：

$A_{n-1}\times10^{n-1}+A_{n-2}\times10^{n-2}+\cdots+A_2\times10^2+A_1\times10+A_0+B_1\times10^{-1}+B_2\times10^{-2}+\cdots$

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：

$A_{n-1}\times2^{n-1}+A_{n-2}\times2^{n-2}+\cdots+A_2\times2^2+A_1\times2+A_0+B_1\times2^{-1}+B_2\times2^{-2}+\cdots$

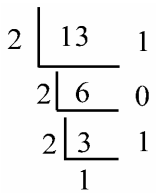
例如，1101.01 可以表示为 $1\times2^3+1\times1^2+0\times2+1+0\times2^{-1}+1\times2^{-2}$，即十进制数 13.25。

可见，将二进制数转换为十进制数不难，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，依次类推，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，乘积中的小数部分再乘以 2 得到的整数就是对应二进制数小数部分的百分位，依次类推，即可求出所有的小数位。

例 1.1 13.75 的整数部分是 13，小数部分是 0.75，转换为二进制数的过程如下所示。



13 相当于二进制数 1101；而 0.75×2=1.5，因此十分位为 1；0.5×2=1.0，因此百分位为 1，即 13.75=1101.11B。

1.1.3 十六进制数

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惟一存在的数制。

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

可见十六进制数位数短，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 位十六进制数，可以表示为

$$A_{n-1}\times16^{n-1}+A_{n-2}\times16^{n-2}+\cdots+A_2\times16^2+A_1\times16+A_0+B_1\times16^{-1}+B_2\times16^{-2}+\cdots$$

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 39103.3671875。

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：

$$A_{n-1}\times2^{n-1}+A_{n-2}\times2^{n-2}+\cdots+A_2\times2^2+A_1\times2+A_0+B_1\times2^{-1}+B_2\times2^{-2}+\cdots$$

如果在上式中，对于整数部分，从个位开始每四位分为一组；对于小数部分，从十分位开始，每四位分为一组，则上式可表示为

$$\begin{aligned}&A_{n-1}\times2^{n-1}+A_{n-2}\times2^{n-2}+\cdots+A_2\times2^2+A_1\times2+A_0+B_1\times2^{-1}+B_2\times2^{-2}+\cdots\\&=(A_{n-1}\times2^3+A_{n-2}\times2^2+A_{n-3}\times2+A_{n-4})\times2^{n-4}+(A_{n-5}\times2^3+A_{n-6}\times2^2+A_{n-7}\times2^1+A_{n-8})\times2^{n-8}+\cdots\\&+(A_7\times2^3+A_6\times2^2+A_5\times2^1+A_4)\times2^4+A_3\times2^3+A_2\times2^2+A_1\times2^1+A_0\\&+(B_0\times2^3+B_1\times2^2+B_2\times2^1+B_3)\times2^{-4}+\cdots+(B_{n-4}\times2^3+B_{n-3}\times2^2+B_{n-2}\times2^1+B_{n-1})\times2^{-(n-1)+4}\end{aligned}$$

上式与十六进制数表示非常接近，括号内就是对应的十六进制数的数码，而 2^{n-4} 就对应位的权，如：

$10101010B=(1\times2^3+0\times2^2+1\times2^1+0\times2^0)\times2^4+(1\times2^3+0\times2^2+1\times2^1+0\times2^0)$

$=A\times2^4+A$

$=A\times16+A$

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一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。**

二进制数	十进制数	十六进制数	二进制数	十进制数	十六进制数
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，**64K **种状态编码分别为：**0000H,0001H,0002H,…,0FFF0H,0FFF1H,…,0FFFEH,0FFFFH**。**

由前面的介绍可看出，十六进制数不仅位数少了，而且也方便记忆。

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（二进制数中的“10”就是十进制数的“2”）。例如：

```
  1001 0110B
+ 0111 0011B
 10000 1001B
```

（2）二进制数的减法

二进制数向前借位时，为“10”，例如：

```
  1001 0110B
- 0111 0011B
  0010 0011B
```

（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=0，1×0=0（交换律），1×1=1。例如：

```
   1001 0110B
×       101B
  10010110
 00000000
+10010110
1011101110 B
```

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：

```
  3 4 6 A H
+ 5 8 9 C H
  8 D 0 6 H
```

在上式中，个位的 A（10）加 C（12），结果为 22，即十六进制数的“ 16 ”，向十位进 1，结果为 6；十位的 6+9+1（个位进位），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），结果为 13，即十六进制数中的“ D ”；千位 3+5，结果为 8。

```
  746AH
- 589CH
  1BCEH
```

在上式中，个位向十位借 1 后，变成十六进制数的“ 1A ”，即十进制数的 26，再减 C（即十进制数的 12），结果为 14，即十六进制数的“ E ”；十位原本是“ 6 ”，个位借 1 后变为“ 5 ”，十位再向百位借 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 1 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 ”。

```
      7  4  6  AH
×        9  CH
（进位）3  4  7
```

```
      5  7  4  F  8
（进位）2  3  5
  +  4  1  7  B  A
     4  6  F  0  9  8
```

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数乘法运算的中间结果转为十六进制数，例如 6×9 的结果为十进制数的 54，转化为十六进制数是 36H。

1.2 码制

在这一节中，主要介绍本课程常用到的 ASC II 码，原码、反码、补码，以及十进制数的二进制表示法——BCD 码等方面的基本知识。

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了编码。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惟一，例如用 0110000B 表示字母 A，0110001B 表示字母 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，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 ASC II 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可参考有关资料。

在计算机系统中，存储单元的长度通常为 8 位二进制数（一个字节），为了存取方便，规定一个存储单元存放一个 ASC II 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 ASC II 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 ASC II，即扩展 ASC II 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 ASC II 码，基本 ASC II 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 ASC II 码，扩展 ASC II 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）

十进制数毕竟是人们最习惯的计数方式，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。一位十进制数包含 0～9 十个数码，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 码规定了十进制数 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（每一位十进制数用相应的 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，需要先转成十进制

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，最高位规定为“0”；对于负数，最高位为“1”。例如：

56H 可以表示为 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 可以表示为 1 1010110。

0256H 可以表示为 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 可以表示为 1 000 0010 0101 0110。

1. 原码

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

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，绝对值部分按位取反。

3. 补码

在计算机内，带符号数并不是用原码或反码表示，而是用补码表示，引入原码、反码的目的只是为了方便理解补码概念而已。不用原码表示的原因是：用原码表示时，0 的原码并不惟一，0 的原码可以表示为 0 0000000（+0），也可以表示为 1 0000000（-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，减法就可以转化为加法运算，例如：

56H-23H=56H-23H+100H （100H 是 8 位二进制能表示的最大数，加上 100H 后，对计算结果没有影响，原因是 8 位二进制无法存放 100H 中的‘1’，即 b8 位）

=56H+100H-23H

=56H+0DDH

=1 33H

=33H（由于 8 位二进制不能存放 b8 位，结果 133H 中最高位“1”自然丢失）

可见在 8 位二进制中，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。显然在 8 位二进制中，23H 与 0DDH 互为补码。

补码的定义为：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，表示该数为正数，数值部分就是真值。而最高位为 1

时是负数，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-32768～+32767。

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（16 位二进制数-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 位二进制数的补码是不同的，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代数式 12×34+56÷7-8 的过程。

首先要有算盘作为计算工具，在计算机里用“运算器”（简称 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，除了需要对数据进行加、减、乘、除四则运算外，还需要进行逻辑与、或、非、异或等逻辑运算。其次需要纸和笔记录算式、计算步骤、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（寄存器在中央处理器内，特点是存取速率快，但数量少，用于存放中间结果，而存储器由成千上万个存储单元组成，容量大，与寄存器相比，存取速率慢一些，常用于存放数据、计算步骤，即指令）。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，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，如果改用计算机进行，可用计算机汇编语言指令写出如下的计算步骤。

```
MOV B,#34       ;将乘数 34 送 CPU 内寄存器 B。
MOV A,#12       ;将被乘数 12 送 CPU 内寄存器 A。
MUL AB          ;计算 12×34，乘积的高 8 位存放在寄存器 B 中，低 8 位存放在寄存器 A 中。
MOV R2,A
MOV R3,B        ;将中间结果暂时保存到寄存器 R2、R3 中。
MOV B,#7        ;将除数 7 送 CPU 内寄存器 B。
MOV A,#56       ;将被除数 56 送 CPU 内寄存器 A。
DIV AB          ;计算 56÷8，商存放在寄存器 A 中，余数存放在寄存器 B 中。
ADD A,R2        ;低 8 位相加。
MOV R2,A        ;把结果暂存到 R2 中。
MOV A,R3        ;取高 8 位。
ADDC A,#00      ;低 8 位相加时可能产生进位，用 ADDC 指令将 A 与 00 相加，即可将低 8
                ;位相加产生的进位加到高 8 位中。
MOV R3,A        ;把结果暂存到 R3 中。
MOV A,R2        ;取低 8 位。
CLR C           ;清进位标志。
```

```
SUBB A,#08      ;低 8 位减 8。
MOV R2,A        ;把结果送 R2 中。
MOV A,R3        ;取高 8 位。
SUBB A,#00      ;12×34＋56÷7 获得的中间结果减 8 时，低 8 位可能产生借位，因此需
                ;要用 SUBB 指令将高 8 位与 00 相减。
MOV R3,A        ;把结果送到 R3 中，这样 12×34+56÷7-8 的最后结果就保存在 R3、R2 中。
```

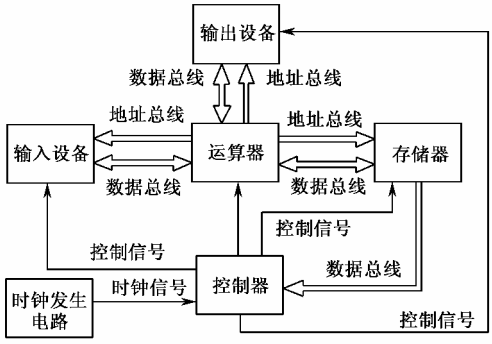


图 1-1　计算机基本结构

上述计算步骤存放在存储器中，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（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，计算机系统的基本结构如图 1-1 所示。

在计算机中，往往把运算器、控制器做在一个芯片上，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，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、控制器等部件集成到一个芯片内，就成为单片机（也称为微控制器，即 Micro Controller Unit，简称 MCU）。这就表示一个芯片就具备了一个完整的计算机系统所必需的基本部件。为了适应不同的需求，将不同功能的外围电路，如定时器、中断控制器、A/D 及 D/A 转换器、串行通信接口电路等集成在一个芯片内，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

一个电路总是由元器件通过导线连接而成的。在模拟电路中，器件、部件间一般是串行关系，彼此之间的连线并不多，关系也不复杂。但在以微处理器为核心的计算机电路中，器件、部件都要与微处理器相连，需要的连线多，如果仍采用模拟电路的连线方式，在微处理器与各器件间单独连线，则连线数量将多得惊人，为此在计算机电路中引入了总线的概念：即每一器件的数据线连接在一起，构成数据总线；地址线连接在一起，构成地址总线；然后与 CPU 的数据、地址总线相连。为避免混乱，任何时候只允许一个设备与 CPU 通信，因此需要控制线进行控制。所有器件的 8 根数据线全部接到 8 根公用的数据线上，即相当于各个器件并联起来。但仅这样还不行，如果有两个器件同时送出数据，一个为 0，一个为 1，那么接收方接收到的究竟是什么呢？这种情况是不允许的，所以要通过控制线进行控制，使器件分时工作，任何时候只能有一个器件发送数据（可以有多个器件同时接收）。器件的数据线也就被称为数据总线，器件所有的控制线被称为控制总线。在单片机内部或者外部存储器及其他器件中有存储单元，这些存储单元要被分配地址才能使用。分配地址当然也是以电信号的形式给出的，由于存储单元比较多，所以用于地址分配的线也较多，这些线