单片微型计算机

原理　接口　应用

徐惠民　安德宁　编著

北京邮电学院出版社

内 容 简 介

本书是以单片微型机为中心介绍微机原理的新型教材。它从计算机基本知识入手，全面介绍微型计算机的组成、软件和接口，重点叙述了MCS-51单片微型机的结构，指令系统，程序设计以及对外部设备的接口，并介绍了各种常用接口芯片的特点及它们的应用。

本书可作为高等学校微机原理课或单片机原理课的教材，也可供工程技术人员参考或作为培训班的教材。

单片微型计算机原理·接口·应用

编　　著　徐惠民　安德宁

责任编辑　郑　捷

北京邮电学院出版社出版

100088　海淀区学院路42号

新华书店北京发行所发行　各地新华书店经售

通县向阳印刷厂印刷

787×1092毫米　1/16　印张17.0　字数423千字

1990年6月第一版　　1991年3月第二次印刷

印数：5501-15500册

ISBN 7-5635-0010-3/TN·12　定价：4.80元

前　言

在70年代以前，由于价格昂贵，对环境要求高等因素，只是少数单位才能装备计算机，掌握和使用计算机的人也不多。自70年代初期Intel公司首先推出4位微处理器以后，情况就起了根本性的变化。由于微处理器和微型计算机价格低廉，性能又在不断地完善，使得它们应用的领域越来越广。不仅进入了各行各业、工厂车间，也进入了中、小学，甚至千家万户。与此同时，需要掌握和使用计算机，特别是微型计算机的人也越来越多。因此，在各个高等学校，在各种短训班中，"微机原理"是一门经常开设，并且选修人数很多的热门课程。

由于历史的原因，"微机原理"课程多以8位微处理器为中心来安排和组织的，常见的微机原理书籍亦如此。特别是当Z80单板机推广使用之后，Z80几乎成了8位微处理器的代名词。然而，自70年代后期以来，各种集成度更高、性能价格比更加优越的单片微型计算机开始问世，并且正在得到推广使用。在我国，单片微型计算机受到重视，则是80年代中期的事，其标志之一是有一些介绍单片微型计算机的书籍问世。但阅读这些书籍，一般都需要一定的计算机知识，最好是学过微机原理，否则的话，阅读起来会有一定的困难。

本书的意图是将单片微型计算机和微机原理结合起来，使得没有学过计算机的读者也能比较顺利地阅读此书，从而既掌握一定的计算机知识，也学习了单片微型计算机。如果高等院校以此书作为教材，则通过一门课而不是象现在的两门课（微机原理，单片微机）就可进入单片机的领域。若此书对于单片微机的推广使用以及高校的课程改革有一点作用，将是我们的最大欣慰。

由于单片微机是集CPU、存贮器、I/O接口于一身的芯片，而介绍微机原理则是希望将这几部分逐一地加以展开。把这两者结合在一起加以叙述，对我们来说是个新的课题。我们期待着听到各位同行、专家、广大读者对本书的批评意见。

徐惠民　安德宁

1989年9月于北京邮电学院

目　录

绪　论

从1946年第一台电子计算机问世到现在不过40多年的历史，但是它的发展与变化却极为惊人。现在，经常在个人计算机上操作的人很难想象第一台电子计算机ENIAC是怎样的一个庞然大物：它有18000个电子管，占地面积约为300平方米，重量为30吨，消耗功率为50千瓦，价值48万美元，而其性能只相当于一个可编程的计算器(如HP-67)。与现在通用的一些微型计算机相比它的性能相差很远：例如它只能存贮750条指令，每秒钟只能进行360次乘法运算；而本书即将介绍的单片微型计算机可以存贮几万条指令，每秒钟进行的乘法运算可高达十万次，但价格只有ENIAC的十万分之一。

40多年来，电子计算机的发展已经历了四代：第一代为电子管数字计算机，其发展年代大约为1946年到1958年。此时，计算机的逻辑元件采用电子管。主存贮器采用磁芯、磁鼓，外存贮器已开始采用磁带。运算速度为每秒几千次到几万次。用途则主要用于科学计算。编写程序主要采用机器语言，后期逐渐发展了汇编语言。

第二代是晶体管计算机，其发展年代大致为1958年至1964年。晶体管代替了电子管作为计算机的逻辑元件。主存贮器仍为磁芯，外存贮器开始使用磁盘。计算机软件也有了很大发展，高级语言和编译程序已很普遍。计算机运算速度提高到每秒几万次到几十万次。其应用也已扩展到各种事务的数据处理，并开始用于工业控制。

第三代计算机开始采用中、小规模集成电路，其发展年代为1964年至1971年。到了1971年出现了集成在一块大规模集成电路上的微处理器——微型计算机的核心。一般认为第四代计算机的起点是在70年代的中期以后，那时，大规模集成电路芯片开始用于一些中、大型计算机。80年代则仍然是第四代计算机的时代。第四代计算机有四个发展方向，有人称为四个"Super"。一是速度高，处理能力极强的巨型计算机，即Supercomputer。巨型机的运算速度可达每秒几亿次，甚至达到十亿次/秒。第二个Super是小巨型机，即Minisupercomputer。它是并行计算技术和多微处理器系统相结合的产物。90年代以后的小巨型机可能包含成千上万个微处理器，其性能接近巨型机，但价格要低得多。超级小型机是另一个Super，即Superminicomputer。当前的超级小型机是在70年代后期的16位小型计算机基础上发展起来的32位小型计算机，其性能大大超过了传统的小型机，同时又保持了小型机的价格便宜等特点。最后一种Super是采用VLSI(超大规模集成)技术的超级微型机，即Supermicrocomputer。它们都采用超大规模集成的32位微处理器芯片，性能价格比都很高。

第四代计算机的四个发展动向中有两个是和微处理器及微型计算机有关的。微型计算机(Microcomputer)与其它计算机的区别在于它的中央处理器(CPU)是采用大规模集成技术集成在一块硅片上；而其它的大、中、小型计算机的CPU是由相当多的集成电路组成。为了和其他计算机的CPU相区别，称微型计算机的CPU芯片为微处理器，即Microprocessor，简称为MPU(Microprocessing Unit)。微型计算机按字长来分，可以有四种：

最早出现的微处理器字长为4位，是1971年Intel公司推出的4004微处理器，它采用PMOS工艺，平均指令周期约20微秒。但使用最广泛的则是TI公司的TMS-1000系列。4位微处理

器虽然性能比较简单，但由于价格便宜(低于1美元)，因此，在玩具、计算器、游戏机以及其他较简单的控制场合仍在广泛使用。

最早的8位微处理器8008也是1971年Intel公司推出的。但8位微处理器的典型代表则是1974年Intel公司推出的8080，Motorola公司的6800，以及1976年Zilog公司的Z80，它们都采用NMOS工艺，平均指令周期约2微秒。比较高档的8位微处理器象Intel 8088，其对外仍为8条数据线，但内部处理时，则采用16条数据线。在这一期间，还出现了8位单片微型计算机，它是将一台微型计算机的最基本部件，包括微处理器，一定容量的存贮器和一部分接口电路都集成在一块硅片上而成的。1976年出现的Intel 8048是8位单片微型计算机的早期产品。

16位微处理器最早出现在1974年。现在比较典型的芯片则还要晚一些时候出现：Intel 8086出现于1978年；1979年为Zilog公司的Z8000；1980年为Motorola的M68000，它的内部数据线有32条，故也有人称之为"准32位"机。这些微处理器的指令周期可小于0.6微秒，性能上已达到或接近中档小型机的水平。1982年出现的Intel 80286是8086的高档产品，它不仅比8086增加了20多条基本指令，直接寻址范围也由1M增加到16M，其他性能也有不少改进。

16位微处理器芯片已进入超大规模集成电路的行列，例如Intel 80286已包含130000个器件。32位微处理器则无例外地都属于VLSI芯片。1981年，Intel的32位微处理器iAPX 432开始问世。现在使用比较普遍的则是1984年Motorola公司的68020以及1985年Intel公司的80386。80386的集成度已达275000个器件。用这些微处理器组成的微型计算机系统，在性能上已可以和小型机甚至某些中型机相比。

在微处理器由低档向高档发展的同时，单片微型计算机也在不断地发展：一方面80年代出现字长16位的高档机，使其性能和应用领域都大大地扩展了；另一方面是各种高性能的专用单片机的出现，如可专用于信号处理的TMS 320系列信号处理单片机。目前，在我国8位单片微型计算机因软件齐全，价格低廉，在应用中处于主导地位，16位单片微机和专用单片机也已进入普及应用的阶段。

包括单片微机在内的微型计算机应用范围是非常广泛的。

1. 用于科学计算。在发展科学技术和生产中所遇到的各种数字问题的计算统称为科学技术计算。计算机由于运算速度之高是其它运算工具无法比拟的，所以在科学技术计算中发挥了强大的威力。当然，微型计算机由于速度和容量的限制，在科学技术计算方面，还不能和大型机、巨型机相竞争。

2. 用于管理工作中的数据处理。所谓数据处理是指企业管理、会计、统计、资料管理和试验资料整理等大量的数据加工、合并、分类等项工作。例如：用计算机进行企业生产计划、产品质量、劳动组织、仓库管理、经营销售等各方面的管理，可以帮助企业管理人员作出正确的决策和判断，从而大大地提高企业管理水平。

3. 用于过程控制。利用计算机进行生产过程的实时控制，不仅可以大大提高自动化水平，提高控制的准确度，提高产品的质量，而且可以降低成本，减轻劳动强度。近年来，微机控制在电力、机械、石油、化工、铁路运输及轻工业等许多部门都得到了广泛的应用，且收到较好的效果。

4. 用于仪器仪表。微型计算机用于仪器仪表中，使它们具有数据存贮、数据处理、自

动测试、自动校准及自动诊断故障的能力，扩大了仪器仪表功能，提高了测量精度和测量的可靠性。

5．用于计算机辅助设计。计算机辅助设计简称 CAD，它是利用计算机的功能部分地代替人工来进行各种产品的分析和设计，从而缩短设计周期，提高设计的成功率和可靠性。例如：大规模集成电路的设计就离不开CAD，此外，CAD 还广泛应用于机械、建筑甚至服装等行业。

6．用于通信。微型计算机不仅用于原有通信设备的改造更新，如纵横制交换机的自动计费设备等，而且广泛地应用于许多近代通信设备中，如各种型号的程控交换机中普遍地采用了微处理器。而计算机技术和通信技术相结合的产物——计算机通信网，不仅成为现代化通信的重要手段，而且它本身也表明了近代通信与计算机技术密不可分的关系。

第一章　计算机基础知识

§1.1　计算机中数的表示方法及运算

在数字计算机的设计与使用上常使用的数制是二进制、十进制和十六进制。十进制是人们习惯的数制，但在计算机内部几乎没有例外地都采用二进制，十六进制只是为了将二进制数表示得更简略时才用到。在这一节中讨论数的表示则着重于负数在机器中的表示和运算。

一、机器数和真值

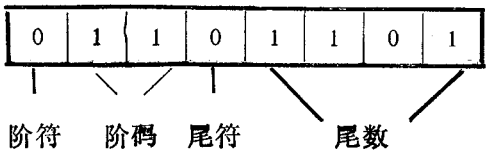
一个数在机器(即计算机)中的表示形式称为机器数。机器数有如下几个特点：

1．机器数所能表示的数的范围受到机器字长的限制。在计算机中，作为数据传送、存贮和运算基本单位的一组二进制字符称为一个字(Word)，一个字中的二进制字符的数目称为字长(Word Length)。计算机的字长确定以后，机器数所表示的数值范围大小也就一定了。如对于8位字长的计算机来说，机器数的范围为$(00000000)_2$—$(11111111)_2$，即对应于十进制数0—255。为了扩大机器数表示的范围，有时可以用两个字甚至多个字表示一个数，例如对于8位机来说，若用两个字来表示一个正数，其数值范围为0—65535。但应注意，这种多字表示方法是在计算机应用中的一种处理方式，从计算机本身来说，则是以单字所表示的数来确定它本身能表示的数的范围的。

2．机器数可以用来只表示正数，有时也称为无符号数，也可以用来表示带符号数，即可正可负。此时，数的符号在机器中也数码化了，即将一个字的最高位定为符号位，其余各位为数值位。最高位为0表示正数，最高位为1表示负数。例如$N_1=+1011011$的机器数可表为$N_1=01011011$，$N_2=-1001001$的机器数可表为$N_2=11001001$。带符号数也可以用两个字来表示一个数，此时，符号位仍定为两个字的最高一位。

3．机器数还可以用来表示带小数点的数。通常有两种表示方法：定点表示法和浮点表示法。在定点表示法中，小数点在数中的位置是固定不变的。对于任意一个二进制数N总可以表示为纯整数(或纯小数)和一个2的整数次幂的乘积：$N=2^P\times S$。其中S称为N的尾数，P称为N的阶码，2称为阶码的底。通常定点表示法中$P=0$，定点数只有符号位和尾数。而在浮点表示法中，小数点在数中的位置是浮动的，因而阶码P是一个可变的数值，它可为正数，也可为负数；同样，尾数S可为正数，也可为负数。故在浮点表示法中阶码和尾数要分别表示，并且各自都有自己的符号位。

设有一个数为$N=2^3\times 13$，则其浮点数为$N=+(10)^{+11}\times 1101$(此处P、S均用二进制数表示)。在机器中相应的表示形式为：



浮点数表示的数的范围远远超过定点数，但是浮点数的算术运算也要复杂得多。微型计算机中这两种表示法都可以用，但用得较多的是定点表示法。在大中型计算机中，数有整型数和实型数之分，实型数即为浮点数。

一个机器数所表示的数值称为机器数的真值。真值可以用二进制数表示也可以用十进制数表示，但根据一般的习惯，常用十进制数表示，例如机器数10110110的真值为－54。

二、原码、反码和补码

带符号数的机器数有若干种表示方法，即为原码、反码和补码。

1. 原码

上面提到的在数的前面加一位符号位，并用 0 表示正数，1 表示负数的方法，即为原码表示法。

如：$X_1=67 \quad [X_1]_{原}=01000011$

$X_2=-67 \quad [X_2]_{原}=11000011$

在原码表示法中 0 有两种表示法，即：

$[+0]_{原}=00000000$，$[-0]_{原}=10000000$

原码表示简单易懂，而且与真值的转换方便。但原码表示的数不便于计算机运算，因为在两原码数运算时，首先要判断它们的符号，然后再决定用加法还是用减法。若是采用反码或补码表示法，则可以在两个数的运算中避免减法运算，即将减法运算转换为加法运算。

2. 反码

一个数的反码很容易求得。如果是正数，则它的反码与原码相同；如果是负数，则是它所对应的正数(即绝对值)连符号位在内按位取反而得到的，即所有的'1'都换成'0'，所有的'0'都换成'1'。也就是说：

若$X=+x_1x_2\cdots x_n$　　则$[X]_{反}=0x_1x_2\cdots x_n$

若$X=-x_1x_2\cdots x_n$　　则$[X]_{反}=1\overline{x}_1\overline{x}_2\cdots\overline{x}_n$

如果已知一个数的反码，要求它所表示的真值，若是正数则可直接求解，若是负数则可将符号位除外的数值部分各位取反得到负数的原码，然后再求真值。例如：

$[X]_{反}=01010011$，$[X]_{原}=01010011$，$X=+83$；

$[Y]_{反}=10110011$，$[Y]_{原}=11001100$，$Y=-76$。

在反码表示法中，零也有两种形式，即：

$[+0]_{反}=00000000$，$[-0]_{反}=11111111$。

在微型计算机中，反码表示法用得较少，因此，对于反码的运算也就不作介绍了。

3. 补码

补码是由补数的概念引出来的。我们称一个计量系统所能表示的最大量程为模。若模用K表示，则当满足

$$Z=nK+Y \qquad (1\text{-}1)$$

时，称Z 和Y 互为补数。通常n取1，也可以取其它正整数。两个互为补数的数，实际上是代表同一个事物。例如：一个圆周角是360°，在这个系统中，270°和－30°互为补数，因为：

$$270^\circ=360^\circ+(-30^\circ)$$

从物理上讲，270°和－30°是代表同一个角度。

一个n位二进制数X的模值为2^n，因此，X的补码应为：

$$[X]_{补}=2^n+X \qquad (1-2)$$

由于字长n位的机器只能表示n位数，因此，2^n(它是一个n+1位的数100…0)在机器中仅能以n个0来表示。或者说，2^n和0在机器中的表示形式是一样的。

如果将n位字长的存数单元的最高位留作符号位，对于正数和负数的补码可以这样来求：

当 $X=+x_{n-2}x_{n-3}\cdots x_1x_0$时

$$[X]_{补}=2^n+X=0x_{n-2}x_{n-3}\cdots x_1x_0$$

可见正数的补码与原码相同。

当 $X=-x_{n-2}x_{n-3}\cdots x_1x_0$时

$$\begin{aligned}[X]_{补}&=2^n+X\\&=2^n-x_{n-2}x_{n-3}\cdots x_1x_0\\&=2^{n-1}+2^{n-1}-x_{n-2}x_{n-3}\cdots x_1x_0\\&=2^{n-1}+(2^{n-1}-1)+1-x_{n-2}x_{n-3}\cdots x_1x_0\\&=2^{n-1}+(11\cdots1-x_{n-2}x_{n-3}\cdots x_1x_0)+1\\&=2^{n-1}+\overline{x}_{n-2}\overline{x}_{n-3}\cdots\overline{x}_1\overline{x}_0+1\\&=\overline{0x}_{n-2}\overline{x}_{n-3}\cdots\overline{x}_1\overline{x}_0+1\end{aligned}$$

所以，负二进制数的补码等于它的绝对值按位取反后加1。即

$$[X]_{补}=\begin{cases}X & 若2^{n-1}>X\geqslant 0\\ \overline{|X|}+1 & 0>X\geqslant -2^{n-1}\end{cases}$$

例如，要求X=－87的补码，即$[X]_2=-01010111$

$$\begin{aligned}[X]_{补}&=\overline{01010111}+1\\&=10101000+1\\&=10101001\end{aligned}$$

“求反加1”需要作两步运算，这个过程也可以简化为一步，即只对负二进制数的各位中最低一位1以左的各位求反，而最低一位1和右边各位都不变，即可得到负数的补码。例如要求$[X]_2=-00001000$的补码(对应十进制数为－8)，则只需对前面四个0求反，低四位不变，就可得到补码，即$[X]_{补}=11111000$。读者可以自己思考，为什么这个规则是正确的。

若是已知一个负数的原码而要求出它的补码时，其规则为除符号位之外的各位求反后加1。这种规则只是负数补码规则的一个具体应用，补码规则的本身和符号位并无关系。例如，－1001的补码为0111，－0010的补码则为1110。

补码再次求补以后就可得原数，即真值。对于正数当然没有这样做的必要。对负数的补码经过再次求补并加上负号后，就为原值。例如，对$[X]_{补}=10110$，则$X=-[[X]_{补}]_{补}=-01010$。

在补码表示法中，0只有一种表示形式，即$[0]_{补}=00\cdots0$。对于8位二进制数来说，用补码所表示的数的范围为－128—＋127。

三、补码运算

在微型计算机中，带符号数一般都以补码的形式在机器中存放和进行运算。这是因为补码的加减运算比原码的简单：它是符号位与数值部分一起参加运算，并且能自动获得结果（包括符号和数值都在内）。

设X和Y是两个正数，可以证明两个数和的补码等于两个数补码的和：

$$\begin{aligned}[X+Y]_{补}&=2^n+(X+Y)\\&=(2^n+X)+(2^n+Y)\\&=[X]_{补}+[Y]_{补}\end{aligned}$$

也可以证明这两数差的补码等于被减数的补码及减数负值补码之和：

$$\begin{aligned}[X-Y]_{补}&=2^n+(X-Y)\\&=2^n+X+2^n+(-Y)\\&=[X]_{补}+[-Y]_{补}\end{aligned}$$

即证明了在补码运算中，两数差的运算可以用加法来代替。一般而言，两个数的补码运算可按以下步骤进行：

1. 把参与运算的两数连同其前面的正负号，变成补码；
2. 进行补码加法，得到两数运算结果的补码，若最高位上有进位则舍弃不要；
3. 若要求运算结果的真值，则按补码数求真值的办法进行。

例1.1 用补码运算求64－10。

解：1) $[64]_{补}=01000000$

$[-10]_{补}=11110110$

2)做加法：

$$01000000+11110110=00110110$$

3)求真值。由于是正数，可直接求：

$$(00110110)_2=54$$

例1.2 用补码运算求64－65。

解：1) $[64]_{补}=01000000$

$[-65]_{补}=10111111$

2)做加法：

$$01000000+10111111=11111111$$

3)求真值。由于是负数，要对结果求反加1后再加上负号：

$$(11111111)_{补}=-(00000001)=-1$$

若参加运算的两个补码数相加的结果，超过了机器所允许表示的范围，将使符号位出现错误，如两正数相加符号位为1，两负数相加符号位为0，即得出了不正确的结果，这种情况称为溢出。

例1.3 用补码运算求64＋65。

解：1) $[64]_{补}=01000000$

$[65]_{补}=01000001$

2)做加法：

$$01000000+01000001=10000001$$

此时两个正数相加，其结果的符号位为1，表明出现了溢出。也就是说，在字长8位的情况下，64+65不可能通过单字表示的补码运算来得出正确的结果。为了得到正确结果，可采用双字表示(16位)的补码来运算：

$$[64]_{补}=0000000001000000$$
$$[65]_{补}=0000000001000001$$
$$[64+65]_{补}=0000000010000001$$

此时，和的16位补码数的最高位仍为0，因此没有出现溢出，结果是正确的。

在微型计算机中，带符号数都以补码数在机器中存放。根据指令，这些数可以进行加法运算，也可以进行减法运算。但在实际机器中只有加法器，减法运算也是通过加法运算来完成的。这是因为：

$$[X]_{补}-[Y]_{补}=[X]_{补}+2^n-[Y]_{补}$$
$$=[X]_{补}+[-[Y]_{补}]_{补}$$

所以减法运算只要对减数取补以后，再通过加法运算就可以达到目的。

例如：
$$[X]_{补}=10011100$$
$$[Y]_{补}=01000110$$
$$[-[Y]_{补}]_{补}=10111010$$

因此，
$$[X]_{补}-[Y]_{补}=10011100+10111010$$
$$=01010110$$

其结果和(10011100−01000110)完全一致。

四、原码的乘、除运算

实现原码的乘、除运算时，要分别确定积或商的符号及数值部分。

两个用原码表示的数相乘(或相除)时，乘积(或商)的符号按同号乘、除结果为正，异号乘、除结果为负。这正好可以用两数原码的符号位通过异或运算来取得。

积或商的数值部分可按二进制数的乘、除法则来获得。由于符号位是分别处理决定的，因此，在求积或商的数值部分时，可把两数都当作正数来处理。也可以用类似方法求两个无符号数的积或商。设有两个无符号数1001和0101相乘，按一般习惯，两者相乘可按以下方式进行：

$$\begin{array}{r} 1001 \\ \times\ 0101 \\ \hline 1001 \\ 0000 \\ 1001 \\ +\ 0000 \\ \hline 0101101 \end{array}$$

但是，在计算机中，一般不按这种方法来求积，因为它要一次求出n组二进制数的和，这对于硬件的要求太高，即相当于要求有一个n位并行加法器，一般微型计算机中不配备这样的硬件。在计算机中实现乘、除运算有许多种算法，这里先介绍一种部分积右移的乘法算法。