《微型计算机硬件软件及其应用》学习辅导

成景文 编著

人民邮电出版社

《微型计算机硬件软件及其应用》

学 习 辅 导

成景文 编著

人民邮电出版社

内 容 提 要

本书是以周明德编著、清华大学出版社出版的《微型计算机硬件软件及其应用》为主要教材的学习辅导材料。书中针对初学者普遍遇到的问题作了详细的讲解，还对各部分内容作了归纳和总结，十分便于理解和记忆。对 Z80CPU 的指令系统和中断方式中一些容易搞错的地方作了深入透彻的讲解，并详细介绍了汇编语言程序设计的方法和技巧，还给出了许多具有实用价值的例题和丰富的习题。为方便广大自学者，书末还给出了全部习题答案。

本书可帮助学习微型计算机这门课程的读者更快、更深入地学好这门课程，也可供大专院校有关专业的师生和科技人员参考。

《微型计算机硬件软件及其应用》
学 习 辅 导
成 景 文 编著
责任编辑 马月梅
人民邮电出版社出版
北京东长安街 27 号
中国科技情报研究所排版
北京振华胶印厂印刷
新华书店北京发行所发行
各地新华书店经售

开本：850×1168 1/32 1989 年 8 月 第一版
印张：16 页数：256 1989 年 8 月北京第 1 次印刷
字数：426 千字 印数：1—5 000 册

ISBN 7-115-03968-2/TP · 041
定价：7.10 元

前　言

推广和普及微型计算机是我国计算机技术发展的必然趋势。无论是从事工程技术工作，还是从事经济管理工作的人都会用到微型计算机，可以预料，在不远的将来，不懂计算机就会象今天不识字那样难于做好各种工作。因此，学习微型计算机的人越来越多。

作者在从事微型计算机教学的过程中发现，绝大多数人在初学时都会遇到许多问题，而有些问题在各种书籍中是找不到答案的。对于在校学生来说，他们可以通过教师答疑和批改作业来解决遇到的问题，而对于数量更多的电大学员和自学者来说，这就会给学习带来很多困难。如果把带有普遍性的问题整理出来，不仅可以帮助学习微机课的学生，而且可以帮助更多的人更快更好地学好微型计算机知识，这正是编写本书的目的。

本书是以周明德编著、清华大学出版社出版的《微型计算机硬件软件及其应用》为主要教材的学习辅导材料。全书共分七章，分别与教材的一至七章对应，但章内各节与教材无关。第一章讲了计算机的一些基础知识，主要讨论了各种数制之间的相互转换、补码的概念及其运算等；第二章到第七章以 Z80 微处理器为主要对象，重点讲了指令系统和汇编语言程序设计中的一些问题，对存储器、输入与输出、中断和并行接口电路各部分也作了一定的讲述，特别是对一些不易搞清楚的问题作了比较详细的讲解。从这里，可以找到许多其它手册和资料中找不到的内容，这些内容都是作者通过在机器上实验以后总结出来的，因此，通过本书的学习，不仅可以对 Z80 微处理器有一个透彻的了解，而且也可以掌握一般微型机的软、硬件技术及其应用基础。

本书是作者根据几年来从事微机教学辅导的讲稿，并整理了学生在答疑时和作业中比较普遍的问题编写而成的。书中针对这些问题作了深入详细的讲解，并对难于记忆和易于搞错的内容作了归纳和总结。由于微型计算机是一门实践性很强的技术，仅仅靠看书是很难真正掌握的，必须通过做大量的练习。为此，书中还列举了许多例题，并给出了大量习题。这些题目考虑了实用性、灵活性和广泛性，能够帮助读者更好地掌握和应用微型计算机，并能开拓知识面。实验也是学习微型计算机的一个重要环节，有条件的读者应尽量多做一些实验。为此，书中对实验方面的问题也作了一些讨论。学习的目的在于应用。因此，仅仅看懂了还远远没有达到学习微型计算机的目的，还必须能够熟练地应用。这就需要有一定的技巧和解决实际问题的能力。书中在这方面也作了必要的讲述，并给出了实用举例。

本书在编写过程中，得到了北京邮电学院计算机系徐定芳老师的帮助，她审阅了初稿并提出了许多宝贵意见，在此表示衷心的感谢。本书虽然出自作者之手，但也包含着与作者一起从事过微机课程教学工作的其他老师的智慧，书中有些问题也受到了学生提问的启发。在此，也向他们表示感谢。最后要特别感谢我爱人马爱英同志，她为本书两次抄写了书稿，并在文字上作了润色。

由于作者水平有限，经验不足，书中缺点错误一定不少，欢迎各界读者批评指正。

作者

一九八六年十二月于清华大学

目　录

第一章　计算机基础知识

第一部分　学习辅导

1.1　数的表示与转换

在计算机中，所有的数据和指令都是用二进制码表示的，即只用0和1这两个符号表示。但在书写时，二进制数非常冗长。为了使书写简单，又能直接与二进制数对应，在书写时常常采用十六进制数，因为十六进制数与二进制数之间有非常明显的对应关系，这就是，4位二进制数对应1位十六进制数。它们之间的相互转换很容易，不必进行任何计算工作就能完成。表1-1是4位二进制数与1位十六进制数的对应关系，读者应当熟记。

表1-1

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

在日常生活中，我们都习惯于用十进制数表示数的大小。我们看到一个十进制数，立刻就能反映出它的数量大小概念，而二进制数和十六进制数就不那么直观。因此，二进制数以及十六进制数与十进制数之间的相互转换就是非常重要的。在学习微型计算机时，迅速而准确地完成上述转换是一个基本技能。

在书写时，可以用加英文字母后缀的办法来区别不同进位制的数，通常用 B（Binary）表示二进制，用 H（Hexdecimal）表示十六进制，用 D（Decimal）表示十进制，用 Q 表示八进制。对于十进制数也可以不用任何后缀。例如 1011B 代表二进制数 1011，2AF8H 代表十六进制数 2AF8，324Q 代表八进制数 324，而 125D 和 256D 分别代表十进制数 125 和 256。另外，也可将一个数用括号括起来，在括号外面用下标指出该数的进位制。例如 $(1010)_2$ 代表二进制数 1010，$(AB)_{16}$ 代表十六进制数 AB，$(32)_{10}$ 代表十进制数 32。本书均采用前一种表示法。

我们知道，在一个数中，不同位置的数字代表的数的大小是不同的，即位权不同。例如一个数

$$a_3a_2a_1a_0.a_{-1}a_{-2}a_{-3}$$

代表的数的大小为

$$a_3\times N^3+a_2\times N^2+a_1\times N^1+a_0\times N^0+a_{-1}\times N^{-1}+a_{-2}\times N^{-2}+a_{-3}\times N^{-3}$$

其中 a_i $(i=0,\pm1,\pm2,\cdots\cdots)$ $=0,1,2,\cdots\cdots$，$N-1$。N 代表进位制，即几进制数，N 就为几。

在二进制数中，从小数点往左，第 n 位的位权为 2^{n-1}，在十六进制数中，相同位置的位权是 $16^{(n-1)}$，而在十进制数中，则是 $10^{(n-1)}$。各种进制的数之间的相互转换，有固定的方法，这里不再叙述，读者可参阅有关资料。但这种方法并不实用，因为转换起来太费时间，还容易算错。这里介绍的是一种实用而有效的方法，我们把它称为“位权记忆法”。用这种方法完成一个数的转换，一般

不需要或只需很少的计算工作，而且一般只要心算就能求出，只需几秒钟就能完成。要想熟练地应用这种方法，首先必须记住一些关键的位权，这并不难做到，例如 $2^6=64$，$2^{10}=1024$，$2^{16}=65536$，都很好记忆。我们可以这样来记：左边肩膀上的数（指数）同右边的十进制数都有相同的数字。2 的 6 次方为 64，都有个 6;2 的 10 次方为 1024，都有 10；2 的 16 次方为 65536，也都有 6。这几个位权也很重要：$2^{10}=1024$ 为 1K，在计算机术语中，1K 不代表 1000，而是 1024。在学习计算机时经常会谈到多少多少 K，所以这个数应当记住，即 2 的 10 次方为 1K，也就是 1024。在 8 位微型机中，一般都有 16 根地址线，所以可寻址的范围就是 $2^{16}=65536$。我们常常会听到“内存容量为 64K”这句话，实际上 64K 就是 2^{16}，即 65536，这也是一个非常重要的数，应当记住。记住这几个数，2 的其他任意次方就很容易求出。这是因为，方次加 1，数值就为原来的二倍；反之，方次减 1，数值就为原来的一半。例如：2^{11} 为 2K，即 2048；2^{15} 为 32K，即 32768，等等。

在微型机中，8 位二进制数称为一个字节（Byte），16 位二进制数，即两个字节，称为一个字（Word），32 位二进制数，即四个字节，称为一个长字（Long Word），而每 1 位二进制数称为一个位（Bit），若用十六进制数表示，则 2 位十六进制数就是一个字节，4 位十六进制数即为一个字，8 位十六进制数为一个长字。由于十六进制数比二进制数书写简便，看起来也直观，所以与十进制数的转换也更容易。在十六进制数中，我们应该记住下面几个位权：

$$16^2=256$$
$$16^3=4096$$
$$16^4=65536$$

记住这些数，我们就很容易将一个十六进制数化为十进制数。例如：

$FFFFH = 16^4 - 1 = 65535$

$1000H = 16^3 = 4096$，即 4K

$3FFH = 4 \times 16^2 - 1 = 1023$

其实，下面几个数也很容易记住，FFH＝255，400H＝1024（1K），1000H＝4096（4K）。后面这两个数可以这样来记，第 3 位的“4”代表 1K，第 4 位的“1”代表 4K，记住了这些数，在应用中是很方便的。

我们认为，十六进制数是如此的方便，甚至在将二进制数与十进制数相互转换时，即使先经过十六进制数作媒介，也比直接转换方便得多。例如，将二进制数 100011111101 化为十进制数，直接转换显然不能马上得出结果。如果将其先写成十六进制数，得 8FDH，就很容易得出答案，“8”代表 2K，即 2048，FDH＝FFH−2＝255−2＝253,所以最后结果为 2301。同样，在将十进制数化为二进制数时，先经过十六进制数也较方便。例如将 1000 化为二进制数，我们知道，400H＝1024，24＝18H，所以 1000＝400H−18H＝3E8H＝1111101000B。

在进行小数部分的转换时，则采用二进制数比较方便，只要记住小数点后面第一位为 0.5，以后各位依次为 0.25，0.125，0.0625，……，即可容易地完成转换。

另外，为了在计算机内直接表示十进制数，可采用 BCD 码（Binary Coded Decimal），即用二进制编码的十进制数来表示。也就是将每 1 位十进制数用 4 位二进制数表示。如:0000 表示 0，1001 表示 9，等等。所以将一个十进制数表示为 BCD 数也是很容易的。例如:128＝000100101000BCD，35＝00110101BCD。需要注意，BCD 数与十进制数有直接对应关系。而与二进制数或十六进制数没有直接对应关系。所以，在将二进制数或十六进制数与 BCD 数相互转换时，必须先经过十进制数。例如:

$$01000010B = 42H = 66 = 01100110BCD$$

而不能直接将 01000010B 写为 01000010BCD。

还须注意，由于 10011000BCD 和 10011000B 在计算机内的形式是完全一样的，而它们所代表的实际值不同，因此在计算或进行其他处理时方法也不同（例如 BCD 数的加法和二进制数的加法就不同）。所以在处理一个数时必须明确它表示的是什么数。

在大多数显示器和打印机中，所显示和打印的字符都是以 ASCII 码表示的。所以无论是数字还是其他字符，都可以用 ASCII 码表示。这时每一个符号都要用 8 位二进制数来表示，不同的符号（字母、数字等）对应不同的 ASCII 码。这可查 ASCII 码表（见附录八），这里不再细述。

一个符号的 ASCII 码还可以将这个符号用单引号引起来表示，如'1'表示数字 1 的 ASCII 码，即 31H；'AB'代表字母 A 和 B 的 ASCII 码，即 4142H。

总之，在计算机中，最常用的表示数的方法是二进制。十六进制与二进制直接对应，是为了书写或数码显示方便而采用的。另外，数据还可以用 BCD 数表示。而所有的数据或字符串都可以用 ASCII 码表示。

1.2 二进制数与 BCD 数的运算

一、算术运算

二进制数的加减运算非常简单。对于加法运算，有下述三条规则：

① 0+0=0

② 1+0=1，0+1=1

③ 1+1=0，向高位进 1

例如：

$$\begin{array}{r} 00101101 \\ +\ 01011010 \\ \hline 10000111 \end{array} \quad \leftarrow \text{进位}$$

对于减法，有下述四条规则：

① 0−0＝0

② 1−0＝1

③ 1−1＝0

④ 0−1＝1，向高位借 1

例如：

$$\begin{array}{r} 10110101 \\ -\ 00101010 \\ \hline 10001011 \end{array} \quad \leftarrow \text{借位}$$

二进制数的乘除运算与十进制数类似，这里不再细述。需要指出的是，将一个二进制数左移一位，低位补 0，相当于原数乘 2；将其右移一位，高位补 0，相当于原数除 2。二进制数的另一个优点是很容易判断它是偶数还是奇数。如果它的最低位为 0，则为偶数；如果最低位是 1，则为奇数。

计算机中可以用 BCD 数来表示十进制数，自然也可以对 BCD 数进行运算。对 BCD 数进行运算时，可先按二进制数进行运算，然后再予以调整。下面举例说明计算方法。

将 00110100BCD(34)和 01001000BCD(48)相加，得

$$\begin{array}{r} 00110100 \\ +\ 01001000 \\ \hline 01111100 \end{array}$$

显然结果不是 BCD 数，因为低四位为十六进制数 C，而 BCD 数只能是 0～9。为了得到正确的结果，可再进行调整。调整的原理和办法如下：

因为二进制数相加时，低 4 位大于或等于 16 时才向高 4 位进

位，而 BCD 数(即十进制数)相加时，低 4 位大于或等于 10 就应向高 4 位进位，二者之差为 6。所以，将 BCD 数按二进制数相加后，若本来应该进位而未进位，即结果大于或等于 A，则只要在该 4 位上加 6 即可向高 4 位产生进位。

若按二进制相加时已产生了进位，则说明该 4 位向前进了一位，而本 4 位减少了 16。按 BCD 数加法规则，向前进一位，本 4 位只应减少 10，所以在这种情况下，该 4 位也应加上 6。

按二进制相加后，如果 4 位中既不超过 9，也未向前进位，则加法结果是正确的，不需要进行调整。

综上所述，BCD 数加法的运算规则如下:

(1) 将 BCD 数按二进制数相加。

(2) 先看低 4 位，若大于 9(即 A 到 F)，或者向高 4 位产生了进位，则将低 4 位加 6；否则不加。

(3) 再看高 4 位，同低 4 位一样处理。

减法的规则同加法相似，不同之处在于，将加法规则中的加改为减，进位改为借位。

在下面的例子中，为了书写方便，都写成十六进制数的形式。例如:

①
```
      34  BCD
+     48  BCD
-------------
      7C  H      (低 4 位大于 9)
```
调整(低位加 6):
```
+      6
-------
      82  BCD
```
即 34+48=82。

②
```
      78  BCD
+     49  BCD
-------------
      C1  H   (低 4 位有进位，高 4 位大于 9)
```

调整(加 66H): +66

1 27 BCD

即 78+49＝127。

③ 45 BCD

+ 23 BCD

二进制加法得: 68 H (高、低 4 位既不大于 9，也

调整(不加) 无进位)

68 BCD

即 45+23＝68。

④ 72 BCD

− 35 BCD

二进制减法得: 3D H (低 4 位有借位)

调整(低位减 6): −6

37 BCD

即 72−35＝37。

二、逻辑运算

逻辑运算是将二进制数按位进行的，每一位的运算结果不影响其他任何位。

基本的逻辑运算有“与”(AND)、“或”(OR)、“异或”(XOR)和“非”(NOT)，也叫“反”。前三种运算是在两个数之间进行的，而后一种运算只对一个数进行。下面简述各种运算规则。

(1) “与”，用符号“∧”或 AND 表示。

当两个数都为 1 时，结果为 1，否则结果为 0。

(2) “或”，用符号“∨”或 OR 表示。

两个数中只要有一个为 1，结果就为 1，否则为 0。

(3) “异或”，用符号“⊕”或 XOR 表示。

两个数不同时，结果为 1，相同时结果为 0。

(4) “反”，用上画横线来表示。

原数为0，结果为1；原数为1，结果为0。

例：已知　A＝10010101，B＝11000110

求　　$A \wedge B, A \vee B, A \oplus B, \overline{A}, \overline{B}$

解：

$$\begin{array}{rr} & 10010101 \\ \wedge & 11000110 \\ \hline & 10000100 \end{array}$$

即 $A \wedge B = 10000100$。

$$\begin{array}{rr} & 10010101 \\ \vee & 11000110 \\ \hline & 11010111 \end{array}$$

即 $A \vee B = 11010111$

$$\begin{array}{rr} & 10010101 \\ \oplus & 11000110 \\ \hline & 01010011 \end{array}$$

即 $A \oplus B = 01010011$

$A = 10010101$

$\overline{A} = 01101010$

$B = 11000110$

$\overline{B} = 00111001$

1.3　补码的概念和表示

对初学者来说，补码的概念往往是一个难点，也最容易出错。

在计算机中，为了表示带符号数，引进了原码、反码和补码的概念，而补码用得最多。在介绍补码之前，我们先看看计算机的运算特点。

在计算机中，总是以特定字长的二进制数进行运算的。例如在8位机中，大部分数据都是8位的。在对8位二进制数进行加法运