# 第一章　微型计算机基础

## 第一节　概　　述

自 1946 年世界上第一台电子计算机问世以来。计算机技术得到了突飞猛进的发展。短短四十多年的时间，已经历了四代的更替：电子管计算机、晶体管计算机、集成电路计算机和大规模、超大规模集成电路计算机。80 年代初日本和美国又分别宣布了第五代“非冯·诺依曼”计算机和第六代“神经”计算机的研制计划。

计算机按其性能、价格和体积的不同，一般分为五大类：巨型机、大型机、中型机、小型机和微型机。

微型机是本世纪 70 年代初研制成功的。一方面是由于军事、空间及自动化技术的发展需要体积小、功耗低、可靠性高的计算机，另一方面，大规模集成电路技术的不断发展也为微型机的产生打下了坚实的物质基础。

微处理器是微型机的核心芯片，通常简称为 **μP** 或 **MP**(MicroProcessor)，它是将计算机中的运算器和控制器集成在一片硅片上制成的集成电路。这样的芯片也被称为中央处理单元，简称为 **CPU**(Central Processing Unit)。

微型计算机简称为 **μC** 或 **MC**(MicroComputer)，它是由微处理器、适量内存和 I/O 接口电路组成的计算机。

20 多年来，微处理器和微型计算机获得了极快的发展，几乎每两年微处理器的集成度翻一番，每 2～4 年更新换代一次，现已进入第五代。

1. 第一代(1971～1973 年)4 位或低档 8 位微处理器

1971 年美国 Intel 公司研制成功的 4004 是集成度为 2000 个晶体管/片的 4 位微处理器。1972 年 Intel 公司推出低档 8 位的 8008 也属于第一代微处理器产品。

第一代微处理器的指令系统比较简单，运算能力差、速度慢(基本指令的执行时间为 10～20μs)，但价格低廉。软件主要使用机器语言及简单的汇编语言。

2. 第二代(1974～1978 年)中高档 8 位微处理器

微处理器问世后，众多公司纷纷研制微处理器，逐步形成以 Intel 公司、Motorola 公司、Zilog 公司产品为代表的三大系列微处理器。1973～1975 年，中档微处理器以 Intel 8080、Motorola 的 MC6800 为代表。1976～1978 年，出现高档 8 位微处理器，典型产品为 Intel 8085、Z80 和 MC6809。

第二代微处理器比第一代有了较多改进，集成度提高 1～4 倍，运算速度提高 10～15 倍，指令系统相对比较完善，已具有典型的计算机体系结构以及中断、存储器直接存取(DMA)功能。软件除汇编语言外，还可使用 BASIC、FORTRAN 以及 PL/M 等高级语言。后期开始配上操作系统，如 CP/M(Control Program/Monitor)操作系统，它运用于以 8080A/8085A、Z80、MC6502 为 CPU，带有磁盘及各种外设的微型计算机系统。

3. 第三代(1978～1981 年)16 位微处理器

1977 年左右,超大规模集成电路工艺研制成功,一片硅片上可集成一万个以上的晶体管,16kb(bit)和 64kb 半导体存储器也已出现。微处理器及微型机从第二代发展为第三代。三大公司陆续推出 16 位微处理器芯片,如 Intel 8086 的集成度为 29000 晶体管/片,Z8000 为 17500 晶体管/片,MC68000 为 68000 晶体管/片。这些微处理器的基本指令执行时间约为 0.15μs。以各项性能指标看,比第二代微处理器提高了很多,已达到或超过原来中、低档小型机的水平。用这些芯片组成的微型机有丰富的指令系统、多级中断系统、多处理机系统、段式存储器管理以及硬件乘除运算等。除此以外,还配备了功能较强的系统软件。为方便原 8 位机用户,Intel 公司很快推出 8088,其指令系统完全与 8086 兼容,内部结构仍为 16 位,但外部数据总线是 8 位。并以 8088 为 CPU 组成了 IBM PC、PC/XT 等准 16 位机。由于其性能价格比高,很快占领了世界市场。与此同时,Intel 公司在 8086 基础上研制出性能更优越的 16 位微处理器芯片 80286,以 80286 为 CPU 组成 IBM PC/AT 高档 16 位机。

以上介绍的是 16 位微型机发展的一条途径,即在原 8 位机的基础上发展而来。另一条途径是将已流行的 16 位小型计算机微型化,例如美国 DEC 公司将 PDP-11/20 微型化为 LSI-11,将中档 PDP-11/34 微型化为 LSI-23,又如 NOVA 机微型化为 Micro NOVA 等等。

4. 第四代(1985 年后)32 位高档微处理器

1985 年,Intel 公司推出了 32 位微处理器芯片 80386。80386 有两种结构:80386SX 和 80386DX。这两者的关系类似于 8088 和 8086 的关系。80386SX 内部结构为 32 位,外部数据总线为 16 位,采用 80287 作协处理器,指令系统与 80286 兼容。80386DX 内部结构、外部数据总线皆为 32 位,采用 80387 作为协处理器。

1990 年,Intel 公司在 80386 基础上研制出新一代 32 位微处理器芯片 80486。它相当于把 80386、80387 及 8KB($2^3 \times 2^{10}$Byte)高速缓冲存储器集成在一块芯片上,性能比 80386 大大提高。

5. 第五代(1993 年后)64 位高档微处理器

1993 年 3 月,Intel 公司推出 64 位微处理器芯片 Pentium,它的外部数据总线为 64 位,工作频率为 66MHz,以它为 CPU 的 Pentium 机是一种 64 位高档微机。IBM、Apple 和 Motorola 三公司合作生产的 Power PC 芯片是又一种优异的 64 位微处理器芯片,以它为 CPU 的微型机型号为 Macintosh。

新一代高性能 P6 已经问世,1995 年下半年 P6 可以进入商业领域,同时,装有 P6 芯片的 686 机将面世,预计 1996 年初便可主导市场。

## 第二节　计算机中的数制和编码

日常生活中,人们使用各种进制来表示数,如二进制、八进制、十进制、十六进制等等。由于用电子器件表示两种状态比较容易实现,所以,电子计算机中一般采用二进制。但人们又习惯于使用十进制数,因此在学习和掌握计算机的原理之前,需要了解二进制、十进

制、十六进制等表示法，及其相互关系和转换。

另外，人们经常使用的字母、符号、图形以及汉字，在计算机中也一律用二进制编码来表示，这些编码也是本节介绍的内容。

## 一、无符号数的表示及运算

### （一）无符号数的表示法

1. 十进制数的表示法

十进制计数法的特点是：

- 以 10 为底，逢 10 进位。
- 需要 10 个数字符号 0，1，2，…，9。

任何一个十进制数 $N_D$ 可以表示为：

$$N_D = \sum_{i=-m}^{n-1} D_i \times 10^i \qquad (1.2.1)$$

其中，m 表示小数位的位数，n 表示整数位的位数，$D_i$ 为十进制数字符号 0～9。例如，

$$135.7D = 1 \times 10^2 + 3 \times 10^1 + 5 \times 10^0 + 7 \times 10^{-1}$$

上式中的后缀 D 表示十进制数(Decimal)，但 D 可以省略。

2. 二进制数的表示法

二进制计数法的特点是：

- 以 2 为底，逢 2 进位。
- 需要两个数字符号 0，1。

一个二进制数可以表示为如下形式：

$$N_B = \sum_{i=-m}^{n-1} B_i \times 2^i \qquad (1.2.2)$$

例如，$1101.1B = 1\times2^3+1\times2^2+0\times2^1+1\times2^0+1\times2^{-1}$

上式中后缀 B 表示二进制数(Binary)。

3. 十六进制数的表示法

十六进制计数法的特点是

- 以 16 为底，逢 16 进位。
- 需要 16 个数字符号 0，1，2，…，9，A，B，C，D，E，F。其中 A～F 依次表示 10～15。

一个十六进制数可表示为如下形式：

$$N_H = \sum_{i=-m}^{n-1} H_i \times 16^i \qquad (1.2.3)$$

例如，$E5AD.BFH = 14\times16^3+5\times16^2+10\times16^1+13\times16^0+11\times16^{-1}+15\times16^{-2}$

上式中后缀 H 表示十六进制数(Hexadecimal)。

### （二）数制转换

1. 任意进制数转换为十进制数

二进制、十六进制、以至任意进制的数转换为十进制数的方法简单，可按式(1.2.2)、

(1.2.3)等展开求和即可。

2. 十进制数转换为二进制数

(1) 十进制整数转换为二进制整数

任何一个十进制数转换为二进制数后，都可以表示成为式(1.2.2)的形式。问题的核心在于求出 n 及 $B_i$。

下面通过一个简单的例子分析一下转换的方法。例如，

已知 $13D=1101B=1\times2^3+1\times2^2+0\times2^1+1\times2^0$

$B_3$ $B_2$ $B_1$ $B_0$

上式也可以表示为，

$$13D=1101B=(1\times2^2+1\times2)\times2+0\times2^1+1\times2^0$$
$$=((1\times2+1)\times2+0)\times2+1$$

$B_3$ $B_2$ $B_1$ $B_0$

可见，要确定 13D 对应的二进制数，只需从右到左分别确定 $B_0$、$B_1$、$B_2$ 和 $B_3$ 即可。显然，从上式可以归纳出以下转换方法，用 2 连续去除十进制数，直至商等于零为止。逆序排列余数便是与该十进制相应的二进制数各位的数值。过程如下：

2 | 13
2 | 6 …1(商6余1)→$B_0$
2 | 3 …0(商3余0)→$B_1$
2 | 1 …1(商1余1)→$B_2$
0 …1(商0余1)→$B_3$

∴ 13D=1101B

用与此类似的方法可以完成十进制数至十六进制数的转换，不同的是用 16 连续去除而已。

(2) 十进制小数转换为二进制小数。

根据(1.2.2)式，

$$0.8125D=B_{-1}\times2^{-1}+B_{-2}\times2^{-2}+B_{-3}\times2^{-3}+B_{-4}\times2^{-4}$$
$$=2^{-1}(B_{-1}+2^{-1}(B_{-2}+2^{-1}(B_{-3}+2^{-1}\times B^{-4})))$$

由上式可以看出，十进制小数转换为二进制小数的方法是，连续用 2 去乘十进制小数，直至乘积的小数部分等于“0”。顺序排列每次乘积的整数部分，便得到二进制小数各位的系数 $B_{-1}$，$B_{-2}$，$B_{-3}$，…。若乘积的小数部分永不为“0”，则根据精度的要求截取一定的位数即可。0.8125D 的转换过程如下，

0.8125D×2=1.625　　得出 $B_{-1}=1$

0.625D×2=1.25　　得出 $B_{-2}=1$

0.25D×2=0.5　　得出 $B_{-3}=0$

0.50D×2=1.0　　得出 $B_{-4}=1$

所以，0.8125D=0.1101B

3. 二进制数与十六进制数之间的转换

因为 $2^4=16$，故二进制数转换为十六进制数只需以小数点为起点，向两端每 4 位二进制用 1 位十六进制数表示即可。例如，

101110.01011B=0110 1110.0101 1000 B=6E.58H

二进制数书写冗长易错，因此一般用十六进制表示数，比较简洁、方便。

### （三）二进制数的运算

1. 二进制数的算术运算

(1) 二进制加法

二进制加法运算规则为：

0+0=0，

0+1=1，

1+0=1，

1+1=0（进位 1）。

(2) 二进制减法

二进制减法运算规则为：

0−0=0，

1−1=0，

1−0=1，

0−1=1（有借位）。

(3) 二进制乘法

二进制乘法运算规则为：

$0\times0=1\times0=0\times1=0$，

$1\times1=1$。

(4) 二进制除法

二进制除法是乘法的逆运算。

2. 二进制数的逻辑运算

(1) “与”运算(AND)

“与”运算又称为逻辑乘，可用符号“·”或“∧”表示。A、B 两个逻辑变量进行“与”运算的规则如下：

| A | B | A∧B |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

由上可知，只有当 A、B 两变量皆为“1”时，“与”的结果才为“1”。

(2) “或”运算(OR)

“或”运算又称为逻辑加，可用符号“+”或“∨”表示。A、B 两个逻辑变量进行“或”运

算的规则如下：

| A | B | A∨B |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

由上可知，A、B 两变量中，只要有 1 个为“1”，“或”运算的结果就是“1”。

(3) “非”运算(NOT)

变量 A 的“非”运算的结果用 $\overline{A}$ 表示，“非”运算规则如下：

| A | $\overline{A}$ |
|---|---|
| 0 | 1 |
| 1 | 0 |

(4) “异或”运算(XOR)

“异或”运算用“∀”表示，逻辑变量 A、B 进行“异或”运算的规则如下：

| A | B | A∀B |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

由上可知，A、B 两变量取值相同时，“异或”结果为“0”。取值相异时，“异或”结果为“1”。换句话说，一个逻辑变量和“0”异或结果不变，和“1”异或则取反。

以上 4 种逻辑运算都是按位进行的，任何时候都不发生进位。下面举一个逻辑运算的例子，已知 A=11110101B，B=00110000B，则：

$\overline{A}$=00001010B

A∧B=00110000B

```
    1111 0101
∧   0011 0000
-------------
    0011 0000
```

A∨B=11110101B

```
    1111 0101
∨   0011 0000
-------------
    1111 0101
```

A∀B=11000101B

```
    1111 0101
∀   0011 0000
-------------
    1100 0101
```

## 二、带符号数的表示及运算

### （一）带符号数的表示法

日常生活中遇到的数，除上述的无符号数外，还有大量的带符号数。数的符号在计算机中也用二进制数表示，通常用二进制数的最高位表示数的符号。把一个数及其符号在机器中的表示加以数值化，这样的数称为机器数，而机器数所代表的数称为该机器数的真值。机器数可以用不同方法表示，常用的有原码、反码和补码表示法。

1. 原码

数 x 的原码记作$[x]_{原}$，如机器字长为 n，则原码的定义如下：

$$[x]_{原}=\begin{cases}x & ,\quad 0\leqslant x\leqslant 2^{n-1}-1\\ 2^{n-1}+|x| & ,\quad -(2^{n-1}-1)\leqslant x\leqslant 0\end{cases} \tag{1.2.4}$$

例如，当机器字长 n=8 时，

$[+1]_{原}=00000001$，　$[+127]_{原}=01111111$

$[-1]_{原}=10000001$，　$[-127]_{原}=11111111$

当机器字长 n=16 时，

$[+1]_{原}=0000000000000001$，　$[+32767]_{原}=0111111111111111$

$[-1]_{原}=1000000000000001$，　$[-32767]_{原}=1111111111111111$

由此看出，原码表示法中，最高位为符号位，正数为 0，负数为 1。其余 n−1 位表示数的绝对值。原码表示数的范围是$-(2^{n-1}-1)\sim+(2^{n-1}-1)$。8 位二进制原码表示数的范围是−127～+127，16 位二进制原码表示数的范围是−32767～+32767。原码表示法简单直观，但不便于进行加减运算。

2. 反码

数 x 的反码记作$[x]_{反}$，如机器字长为 n，反码定义如下：

$$[x]_{反}=\begin{cases}x & ,\quad 0\leqslant x\leqslant 2^{n-1}-1\\ (2^{n}-1)-|x| & ,\quad -(2^{n-1}-1)\leqslant x\leqslant 0\end{cases} \tag{1.2.5}$$

例如，当机器字长 n=8 时，

$[+1]_{反}=00000001$，　$[+127]_{反}=01111111$

$[-1]_{反}=11111110$，　$[-127]_{反}=10000000$

从反码表示法中可见，最高位仍为符号位，正数为 0，负数为 1。反码表示数的范围是$-(2^{n-1}-1)\sim+(2^{n-1}-1)$。8 位二进制数反码表示数的范围是−127～+127，16 位二进制数反码表示数的范围是−32767～+32767。

从上例中还可以看出，正数的反码与原码相同，负数的反码只需将其对应的正数按位求反即可得到。

3. 补码

(1) 补码表示法

数 x 的补码记做$[x]_{补}$，当机器字长为 n 时，补码定义如下：

$$[x]_{补}=\begin{cases}x & ,\quad 0\leqslant x\leqslant 2^{n-1}-1\\ 2^{n}-|x| & ,\quad -2^{n-1}\leqslant x<0\end{cases} \tag{1.2.6}$$

例如：当机器字长 n＝8 时，

$[+1]_{补}=00000001$，　　$[-1]_{补}=2^8-|-1|=11111111$

$[+127]_{补}=01111111$，　　$[-127]_{补}=2^8-|-127|=10000001$

补码表示法中，最高位仍为符号位，正数为 0，负数为 1。

补码表示数的范围为$-2^{n-1}\sim+(2^{n-1}-1)$。8 位二进制补码表示数的范围为－128～＋127，16 位二进制补码表示数的范围是－32768～＋32767。

从上例中还可以看出，正数的补码与它的原码、反码均相同，负数的补码等于它的反码加 1，也就是说，负数的补码等于其对应正数的补码按位求反（包括符号位）再加 1。如上例中，求－127 的补码过程可以简化如下：

$[-127]_{补}=\overline{[+127]_{补}}+1=\overline{01111111}+1=10000001$

8 位二进制数的原码、反码和补码如表 1.2.1 所示。

**表 1.2.1　原码、反码和补码表**

| 二进制数 | 无符号数 | 带符号数 | | |
|---|---|---|---|---|
| | | 原　码 | 补　码 | 反　码 |
| 00000000 | 0 | ＋0 | ＋0 | ＋0 |
| 00000001 | 1 | ＋1 | ＋1 | ＋1 |
| 00000010 | 2 | ＋2 | ＋2 | ＋2 |
| ⋮ | | | | |
| 01111110 | 126 | ＋126 | ＋126 | ＋126 |
| 01111111 | 127 | ＋127 | ＋127 | ＋127 |
| 10000000 | 128 | －0 | －128 | －127 |
| 10000001 | 129 | －1 | －127 | －126 |
| ⋮ | | | | |
| 11111101 | 253 | －125 | －3 | －2 |
| 11111110 | 254 | －126 | －2 | －1 |
| 11111111 | 255 | －127 | －1 | －0 |

## （二）真值与补码之间的转换

1. 真值转换为补码

根据补码的定义便可以完成真值到补码的转换。

2. 补码转换为真值

（1）正数补码转换为真值

由于正数的补码是其本身，因此，正数补码的真值 $x=[x]_{补}(0\leqslant x\leqslant 2^{n-1}-1)$。

（2）负数补码转换为真值

负数补码和与其对应的正数补码之间存在如下关系：

$$[x]_{补}\xrightarrow{求补运算}[-x]_{补}\xrightarrow{求补运算}[x]_{补}$$

其中，x 是带符号数，正负皆可。求补运算是将一个二进制数按位求反加 1 的运算。对此关系此处不加证明，只用实例说明。

设：x＝＋1，则－x＝－1。

已知：$[x]_{补}=[+1]_{补}=00000001$

对$[x]_{补}$做求补运算的过程如下：

$\overline{[x]_{补}}+1=\overline{[+1]_{补}}+1=\overline{0000\ 0001}+1=11111111=[-1]_{补}=[-x]_{补}$

由这个例子可以看出，当 x 为正数时，对其补码进行求补运算，结果是－x 的补码。

若设：x＝－1，则－x＝＋1。

已知：$[x]_{补}=[-1]_{补}=11111111$。

对$[x]_{补}$做求补运算的过程如下：

$\overline{[x]_{补}}+1=\overline{[-1]_{补}}+1=\overline{11111111}+1=00000001=[+1]_{补}=[-x]_{补}$

由此例可以看出，当 x 为负数时，对其补码进行求补运算，结果是－x 的补码。

上述两例验证了$[x]_{补}$与$[-x]_{补}$之间的关系。显然，对负数补码进行求补运算的结果是该负数对应的正数的补码，也就是该负数的绝对值。因此，负数补码转换为真值的办法如下：将负数补码按位求反加 1(即求补运算)，即可得到该负数补码对应的真值的绝对值。也就是说，对负数而言，$|x|=\overline{[x]_{补}}+1$。

[**例 1.2.1**]　求下列数的补码

① 设 x＝＋127D，求$[x]_{补}$。

应用十进制数转换为二进制数的原则，可以得出 x＝01111111B。故$[x]_{补}=[+127]_{补}$＝01111111。

② 设 x＝－127D，求$[x]_{补}$。

因为对$[x]_{补}$进行求补运算便可得到$[-x]_{补}$。因此，$[x]_{补}=[-127]_{补}=\overline{[+127]_{补}}+1$＝$\overline{01111111}+1=10000001$。

[**例 1.2.2**]　求以下补码的真值。

① 设$[x]_{补}=01111110$，求 x。

因为该补码的最高位为“0”，即符号位为“0”，该补码对应的真值是正数。则 $x=[x]_{补}$＝01111110＝＋126D。

② 设$[x]_{补}=10000010$，求 x。

因为该补码的最高位为“1”，即符号位为“1”，该补码对应的真值是负数，其绝对值为：

$|x|=\overline{[x]_{补}}+1=\overline{10000010}+1=01111101+1=01111110=+126D$

则 x＝－126D。

### （三）补码的运算

1. 补码加法

补码加法的规则是

$$[x+y]_{补}=[x]_{补}+[y]_{补}$$

其中，x、y 为正负数皆可，下面用 4 个例子来验证这个公式的正确性。

已知：$[+51]_{补}=00110011$　　$[+66]_{补}=01000010$

$[-51]_{补}=11001101$　　$[-66]_{补}=10111110$

| 则： | 十进制加法 | 二进制(补码)加法 |
|---|---|---|
| ① | +66 | 01000010=[+66]补 |
| | +) +51 | +) 00110011=[+51]补 |
| | +117 | 01110101=[+117]补 |
| ② | +66 | 01000010=[+66]补 |
| | +) −51 | +) 11001101=[−51]补 |
| | +15 | [1] 00001111=[+15]补 |
| ③ | −66 | 10111110=[−66]补 |
| | +) +51 | +) 00110011=[+51]补 |
| | −15 | 11110001=[−15]补 |
| ④ | −66 | 10111110=[−66]补 |
| | +) −51 | +) 11001101=[−51]补 |
| | −117 | [1] 10001011=[−117]补 |

可以看出，不论被加数、加数是正数还是负数，只要直接用它们的补码(包括符号位)相加，当结果不超出补码表示范围时，运算结果是正确的补码。但当运算结果超出补码表示范围时，结果就不正确了，这种情况称为溢出。上例②、④中由最高位向更高位的进位由于机器字长的限制而自动丢失，不会影响运算结果的正确性。

2. 补码减法

补码的减法规则是：

$$[X - Y]_{补} = [X]_{补} + [-Y]_{补}$$

下面仍用四个例子对此规则的正确性加以验证。

| | 十进制 | 二进制(补码) |
|---|---|---|
| ① | +66 | 01000010=[+66]补 |
| | −) +51 | +) 11001101=[−51]补 |
| | +15 | [1] 00001111=[+15]补 |
| ② | +66 | 01000010=[+66]补 |
| | −) −51 | +) 00110011=[+51]补 |
| | +117 | 01110101=[+117]补 |
| ③ | +51 | 00110011=[+51]补 |
| | −) +66 | +) 10111110=[−66]补 |
| | −15 | 11110001=[−15]补 |
| ④ | −51 | 11001101=[−51]补 |
| | −) −66 | +) 01000010=[+66]补 |
| | +15 | [1] 00001111=[+15]补 |

可见，无论被减数、减数是正数还是负数，上述补码减法的规则是正确的。在计算机中，利用这个规则，通过对减数进行求补运算而将减法变成加法。例中由最高位向更高位

的进位同样会自动消失而不影响运算结果的正确性。

由上述分析，可以看出：计算机中的带符号数用补码表示时，有许多优点。第一，负数的补码与对应正数的补码之间的转换可以用同一方法——求补运算实现，因而可简化硬件。第二，可以将减法变为加法运算，从而省去了减法器。第三，无符号数及带符号数的加法运算可用同一电路完成，结果都是正确的。例如，两个内存单元的内容分别为 11110001 及 00001100，无论它们代表无符号数还是带符号数，运算结果都是正确的。

| | 无符号数 | 有符号数 |
|---|---|---|
| 11110001 | 241 | $[-15]_{补}$ |
| +) 00001100 | +) 12 | +) $[+12]_{补}$ |
| 11111101 | 253 | $[-3]_{补}$ |

## 三、二进制编码

### （一）二进制编码的十进制数（BCD 码）

尽管计算机采用的二进制数的表示法及运算规则简单，但书写冗长、不直观且易出错，因此计算机的输入输出仍采用人们习惯的十进制数。当然，十进制数在计算机中也需用二进制编码表示。这种编码有多种形式，BCD(Binary-Coded Decimal)码比较常用。4 位二进制有 16 种组合态，当用来表示十进制数时，要舍去 6 种组合态。BCD 码有两种形式，即压缩 BCD 码和非压缩 BCD 码。

1. 压缩 BCD 码

压缩 BCD 码的每一位用 4 位二进制表示，一个字节表示两位十进制数。例如，10010110B 表示十进制数 96D。

2. 非压缩 BCD 码

非压缩 BCD 码用 1 个字节表示一位十进制数，高 4 位总是 0000，低 4 位的 0000～1001 表示 0～9。例如，00001000B 表示十进制数 8。

两种 BCD 码的部分编码见表 1.2.2。

**表 1.2.2　BCD 码表**

| 十进制数 | 压缩 BCD 码 | 非压缩 BCD 码 |
|---|---|---|
| 0 | 00000000 | 00000000 |
| 1 | 00000001 | 00000001 |
| 2 | 00000010 | 00000010 |
| ⋮ | ⋮ | ⋮ |
| 9 | 00001001 | 00001001 |
| 10 | 00010000 | 00000001　00000000 |
| 11 | 00010001 | 00000001　00000001 |
| ⋮ | ⋮ | ⋮ |

### （二）字母和符号的编码（ASCII 码）

计算机处理的信息除了数字之外还需要处理字母、符号等，例如键盘输入及打印机、

CRT 输出的信息大部分是字符。因此，计算机中的字符也必需采用二进制编码的形式。编码有多种，微型机中普遍采用的是 ASCII(American Standard Code for Information Interchange)码，即美国标准信息交换代码。ASCII 码用 8 位二进制对字符进行编码，见附录 1.1。

## 第三节　微型计算机系统的组成、分类和配置

### 一、微型计算机系统的组成

微型计算机系统与一般的计算机系统一样，由硬件和软件两部分组成，如图 1.3.1 所示。

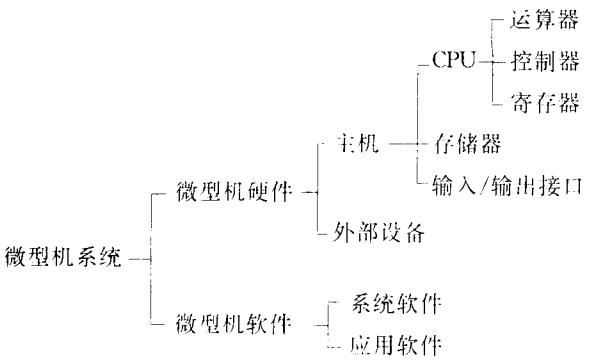


图 1.3.1　微型机系统的组成

#### (一) 微型计算机硬件

1. CPU

CPU 是微型机的核心芯片，它包括运算器、控制器和寄存器三个主要部分。运算器也称为算术逻辑单元 ALU(Arithmatic and Logic Unit)。顾名思义，运算器的功能是完成数据的算术和逻辑运算。控制器一般由指令寄存器、指令译码器和控制电路组成。控制器根据指令的要求，对微型机各部件发出相应的控制信息，使它们协调工作，从而完成对整个计算机系统的控制。CPU 内部的寄存器用来存放经常使用的数据。

2. 存储器(Memory)

存储器又称为主存(Main Storage)或内存，是微型机的存储和记忆装置，用以存放数据和程序。微型机的内存通常采用半导体存储器。

(1) 内存单元的地址和内容

内存中存放的是数据和程序，从形式上看，均为二进制数。一般将 8 位二进制数记做一个字节(Byte)，每一个内存单元中存放一个字节的二进制信息，内存容量就是它所能包含的内存单元的数量。通常以字节为单位，1024($2^{10}$)字节记做 1KB，$2^{20}$字节记做 1MB。

微型机通过给各个内存单元规定不同地址来管理内存。这样，CPU 便能识别不同的内存单元，正确地对其进行操作，见图 1.3.2。

显然，内存单元的地址和内存单元的内容是两个完全不同的概念。例如在图 1.3.2

中，第 6 号内存单元的地址是 00006H，而其内容是 11001111B，即 CFH。

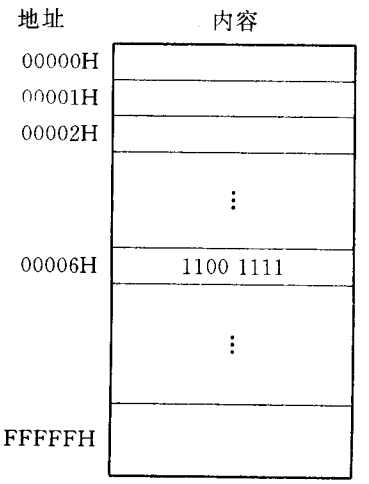


图 1.3.2　内存单元的地址和内容

（2）内存的操作

CPU 对内存的操作有两种：读或写。读操作是 CPU 将内存单元的内容读入 CPU 内部，而写操作是 CPU 将其内部信息传送到内存单元保存起来。显然，写操作的结果改变了被写内存单元的内容，是破坏性的，而读操作是非破坏性的，即，该内存单元的内容在信息被读“走”之后仍保持原信息。

（3）内存的分类

按工作方式，内存可分为两大类：随机存储器 RAM（Random Access Memory）和只读存储器 ROM（Read Only Memory）。

RAM 可以被 CPU 随机地读写，故又称为读写存储器。这种存储器用于存放用户装入的程序、数据及部分系统信息。当机器断电后，所存信息消失。

ROM 中的信息只能被 CPU 读取，而不能由 CPU 任意写入，故称为只读存储器，机器断电，信息仍保留。这种存储器用于存放固定的程序，如：基本的 I/O 程序，BASIC 解释程序以及用户编写的专用程序。ROM 中的内容只能用专用设备写入。

3. 输入/输出（I/O）设备和输入/输出接口（I/O Interface）

I/O 设备是微型机系统的重要组成部分。程序、数据及现场信息要通过输入设备输入给微型机。CPU 计算的结果通过输出设备输出到外部。常用的输入设备有键盘、鼠标器、数字化仪、扫描仪、A/D 变换器等。常用的输出设备有显示器、打印机、绘图仪等。磁盘、磁带既是输入设备，又是输出设备。

外设的种类繁多，有机械式、电动式、电子式等，且一般说来，与 CPU 相比，工作速度较低，外设处理的信息有数字量、模拟量、开关量等，而微型机只能处理数字量。另外，外设与微型机工作的逻辑时序也可能不一致。鉴于上述原因，微型机与外设间的连接及信息的交换不能直接进行，而需设计一个**“接口电路”**作为微型机与外设之间的桥梁。这种接口电路又叫做“I/O 适配器”（I/O Adapter）。

综上所述，微型机硬件主要由 CPU、内存、I/O 接口和 I/O 设备组成。微型机各部件之间是用系统总线连接的。系统总线就是传送信息的公共导线，一般有三组总线。地址总线 AB(Address Bus)传送 CPU 发出的地址信息，是单向总线。数据总线 DB(Data Bus)传送数据信息，是双向总线，CPU 既可通过 DB 从内存或输入设备读入数据，又可通过 DB 将 CPU 内部数据送至内存或输出设备。控制总线 CB(Control Bus)传送控制信息。其中，有的是 CPU 向内存及外设发出的信息，有的是外设等发送给 CPU 的信息，因此，CB 中每一根线的传送方向是一定的。微型机的系统结构可用图 1.3.3 表示，图中 CB 作为一个整体，用双向表示。

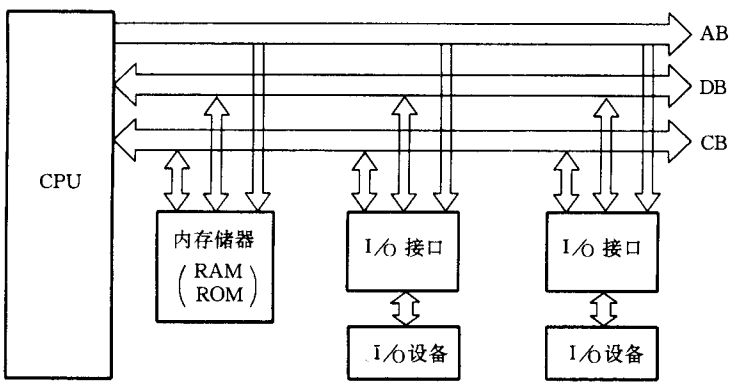


图 1.3.3　微型机的外部结构框图

### (二) 微型计算机软件

微型机软件是为了运行、管理和维护微型机而编制的各种程序的总和。软件和硬件是微型机系统不可分离的两个重要组成部分。没有软件，微型机就无法工作。

微型机软件包括系统软件和应用软件。应用软件就是用户为解决各种实际问题而编制的各种程序。系统软件主要包括操作系统 OS(Operating System)和系统应用程序。操作系统是控制微型机的资源，如 CPU、存储器及 I/O 设备等，使应用程序得以自动执行的程序。系统应用程序很多，如各种语言的汇编、解释、编译程序，诊断和调试程序，文字处理程序，服务性工具程序，数据库管理程序等。

## 二、微型计算机的分类

微型机的分类方法很多。按微处理器的位数，可分为 1 位、4 位、8 位、32 位和 64 位机等。按功能和结构可分为单片机和多片机。按组装方式可分为单板机和多板机，等等。

利用大规模集成电路工艺将微型机的三大组成部分——CPU、内存和 I/O 接口集成在一片硅片上，这就是单片机(Single-Chip Computer)。使用简单的开发装置可以对它进行在线开发。单片机在工业过程控制、智能化仪器仪表和家用电器中得到广泛的应用。

若将微型机的 CPU、内存、I/O 接口电路安装在一块印刷电路板上就组成了单板机，单板机结构简单，价格低廉，性能较好，经过开发后，可用于过程控制、各种仪器仪表、机器

的单机控制、数据处理等，且普遍用作学习“微机原理”的实验机型。

若将1位或4位等数位的算术逻辑部件等电路集成在一块芯片上即成为位片式微处理器。多片位片及控制电路连接而成的计算机就叫做位片机。位片一般采用双极型工艺制成，因此速度比较高，比一般MOS芯片高1～2个数量级。用户可根据需要组成各种不同字长的位片机，因此目前受到人们的关注。

本书将主要介绍多片多板微型机IBM PC及PC/XT。

## 三、IBM PC及PC/XT的配置

以8088为CPU的微型机IBM PC及PC/XT的硬件部分由主机和外设组成，主机采用大板结构，此板又称为系统板，水平放置在机箱内。

### (一) 系统板

系统板上的元件按功能分为五大部分：微处理器及其支持芯片、RAM、ROM、I/O接口电路及I/O扩展槽，如图1.3.4所示。

<table>
<tr><td>时钟<br>控制器<br>8284</td><td>微处理器<br>8088<br>8087</td><td colspan="2">20位4通道<br>DMA控制器<br>8237</td><td>16位3通道<br>定时/计数器<br>8253</td><td>8级中断<br>优先级控制器<br>8259</td></tr>
<tr><td colspan="3">盒式磁带接口</td><td rowspan="2">ROM</td><td rowspan="3" colspan="2">RAM</td></tr>
<tr><td colspan="3">扬声器接口</td></tr>
<tr><td colspan="3">键盘接口</td><td>I/O扩展槽</td></tr>
</table>

图1.3.4 IBM PC及PC/XT系统板功能结构图

1. 微处理器及其支持芯片

(1) 微处理器8088及8087

8088是准16位微处理器，时钟频率4.77MHz，有20条地址线，故可寻址$2^{20}=1$MB，即可配接具有1M存储单元的内存。8088有两种工作方式，最小模式是单处理机方式，只允许8088接于系统中。而最大模式是多处理机方式，除8088外，系统可配接浮点协处理器8087，这样，可使PC及PC/XT的浮点运算速度提高约100倍。8087的有关内容请参见本章第4节。

(2) 总线控制器8288

8288将工作在最大模式的8088的状态信号$S_2$～$S_0$进行译码，产生相应的控制信号并加以驱动，以实现8088对内存及外设的控制。

(3) 时钟信号发生与驱动器8284

8284芯片外接频率为14.31818MHz的石英晶振，它可输出供系统使用的14.31818MHz的OSC信号，4.77MHz的CLK信号和2.387MHz的PCLK信号。

(4) 可编程定时/计数器8253

8253具有3个16位定时/计数通道，全部被系统使用。通道0是定时器，它为系统用

时钟提供恒定的时间基准，此通道每隔 55ms 向 CPU 发一信号，8088 对此信号进行计数，平均 18.2 次计时 1 秒，用此来计算时钟的时间。通道 1 用于动态存储器刷新的定时。通道 2 输出方波至扬声器，此方波的频率和持续时间可以由程序设置，以此控制扬声器发声的音调和时间。

(5) DMA(Direct Memory Access)控制器 8237

用于直接存储器存取控制的 8237 有 4 个 DMA 通道。通道 0 用于动态存储器的刷新，通道 2 用于软盘与内存的 DMA 传送，通道 3 用于硬盘与内存间的 DMA 传送，只有通道 1 为用户保留。

(6) 可编程中断控制器 8259

8259 用于 8 级中断优先权的控制。关于中断的概念及 8259 的结构和用法详见第五章。

2. ROM

IBM PC 及 PC/XT ROM 容量为 64KB，但只用了其中的 40KB。地址 F6000H～FDFFFH 的 32KB 中固化了 Basic 解释程序，地址 FE000H～FFFFFH 中固化了基本输入/输出系统 **BIOS**(Basic Input and Output System)。BIOS 是一组管理程序，包括上电自检程序、DOS 引导程序、日时钟管理程序、基本 I/O 设备如显示器、键盘、打印机等的驱动程序，等等。

3. RAM

系统板上的存储器芯片共 4 列，9 片/列组成带奇偶校验的 64KB 内存，后期产品采用高度集成化的存储器芯片后，大板上可安装 256KB 内存，某些兼容机甚至在大板上插接 640KB 内存。

4. I/O 接口电路

在系统板上还有 IBM PC 及 PC/XT 的音频盒式磁带机、键盘及扬声器等的接口电路。盒式磁带常作为低档无磁盘驱动器微型机的外存储器，PC 机的这部分接口电路主要为与低档机兼容，已不常用。

5. I/O 扩展槽

IBM PC 有 5 个，PC/XT 有 8 个 62 芯 I/O 扩展插槽，亦称 I/O 通道，这些插槽相同序号的插脚串接在一起。62 芯插槽分两排，A 排有 31 条引线，对应接口电路板的元件面，B 排也有 31 条引线，对应接口电路板的焊接面。IBM 公司对 62 芯 I/O 通道信号的名称、功能、方向、时序、引脚排列都作了明确规定。通道信号包括数据线 $DB_7$～$DB_0$，地址线 $AB_{19}$～$AB_0$，以及控制线、电源线、地线等。有关 I/O 通道的详细介绍见第五章。

PC/XT 的第 8 个插槽与其余 7 个槽的信号及用法不同，它用于连接扩展机箱，以便再为用户提供 7 个 I/O 通道。不过，随着微机的发展，新机型不断出现，扩展机箱已不再使用。

### (二) I/O 接口选件

IBM PC 及 PC/XT 的主要外部设备除了上述的扬声器和键盘外，还有显示器、打印机、软盘驱动器、硬盘驱动器以及绘图仪、数字化仪等等，这些外设都有相应的接口电路，

插入 I/O 扩展槽后，便可使外设投入使用。

当系统板上的 256KB 内存容量不够时，可在 I/O 槽中插入 IBM 公司为用户提供的存储器扩充选件以扩大内存容量，直至 640KB。

用户设计的接口电路也可做为 I/O 选件而插入 I/O 扩展槽中。

## 第四节　微处理器

### 一、Intel 8086/8088

#### (一) 8086/8088 的功能结构

微处理器 8086、8088 结构类似，都由算术逻辑单元 ALU、累加器、专用和通用寄存器、指令寄存器、指令译码器、定时和控制电路等组成。后四部分相当于控制器。不过，按其功能可以分为两大部分——总线接口单元 BIU(Bus Interface Unit)和执行单元 EU(Execution Unit)。如图 1.4.1 所示。

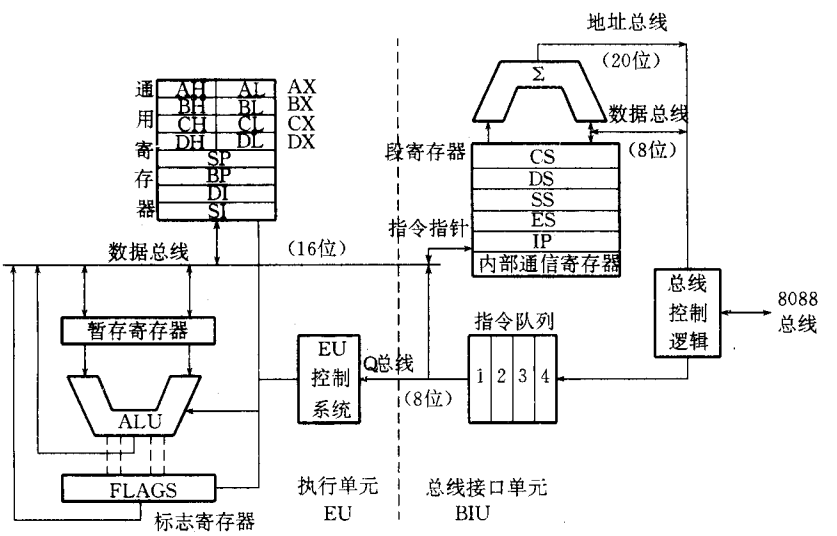


图 1.4.1　8088 功能结构图

执行单元 EU 由 8 个通用寄存器、1 个标志寄存器、算术逻辑单元 ALU 及 EU 控制电路组成。总线接口单元 BIU 包括 4 个段寄存器、1 个指令指针寄存器、1 个与 EU 通讯的内部寄存器、先入先出的指令队列、总线控制逻辑及计算 20 位实际物理地址的加法器 Σ。8088 的指令队列包括 4 个字节，而 8086 为 6 个字节。

EU 的功能是执行指令。EU 从指令队列取出指令代码，将其译码，发出相应的控制信息。数据在 ALU 中进行运算。运算结果的特征保留在标志寄存器 FLAGS 中。BIU 的功能是负责与存储器、I/O 接口传送信息。当 EU 从指令队列中取走指令，指令队列出现空