

# **Z80**

# **微型计算机原理**

长春邮电学院计算机教研室  
邮电部第六实验工厂

# 前 言

近几年来,计算机技术已经取得了惊人的迅猛展发。尤其是微型计算机出现之后,使得计算机渗透到国防尖端、工业、企业管理、日常生活等各个领域,其作用和成就正日益卓著,成了工业发展水平的标志之一。

目前计算机已成为发展新技术,改造老技术的强有力武器。为了迎接“四次工业革命”的挑战,计算机特别是微型机的生产,推广和应用已成为我国社会主义现代化建设的战略产业。

为此,广大干部,技术人员迫切需要学习掌握微型计算机的原理和应用技术。我们曾先后在学院内外为广大技术人员举办了几期Z80微型机学习班。在此基础上编写了《Z80微型计算机原理》、《TP801单板计算机使用手册》、《Z80指令手册》等三本书,供办Z80系列微型机培训班使用。亦可供工科院校相关专业的教师和学生使用Z80系列单板机时参考。

为了使初学者树立起整机概念,建议在讲完《Z80微型计算机原理》的前七章内容之后,补充介绍《TP801单板机使用手册》上有关TP801单板机的原理,结构与使用方法等相关知识。

利用微型计算机进行模拟量的检测与控制,是广大工程技术人员经常遇到的实际问题。为此,在第八章中选编了关于模数与数模变换器的一般原理与常用芯片等内容,供大家参考。

为了满足不同水平的读者需要,在《Z80微型计算机原理》一书后,附录有关算述运算程序和数字逻辑电路的初步知识,供初学者自学。

这套书由李振文同志主编。参加编审工作的同志还有朱景惠,曲雅珍、王庆宝、栾英等。

由于我们的水平有限,经验不足,书中难免有错误,恳切希望读者批评指正。

长春邮电学院计算机教研室  
长春邮电第六实验工厂

一九八四年四月

# 目 录

|                             |         |
|-----------------------------|---------|
| 第一章 微型计算机的基础知识.....         | ( 1 )   |
| 1.1 引言.....                 | ( 1 )   |
| 1.2 数制及不同数制之间的转换.....       | ( 1 )   |
| 1.3 数的定点和浮点表示.....          | ( 6 )   |
| 1.4 码制.....                 | ( 7 )   |
| 第二章 Z80—CPU的结构及定时.....      | ( 10 )  |
| 2.1 概述.....                 | ( 10 )  |
| 2.2 寄存器.....                | ( 11 )  |
| 2.3 总线及引脚 (Pin) .....       | ( 16 )  |
| 2.4 Z80-CPU的定时.....         | ( 17 )  |
| 第三章 存储器.....                | ( 23 )  |
| 3.1 存储器概念.....              | ( 23 )  |
| 3.2 只读存储器.....              | ( 33 )  |
| 3.3 读写存储器.....              | ( 40 )  |
| 第四章 Z80—CPU指令系统.....        | ( 48 )  |
| 4.1 概述.....                 | ( 48 )  |
| 4.2 寻址方式.....               | ( 50 )  |
| 4.3 数据传送指令.....             | ( 52 )  |
| 4.4 数据操作指令.....             | ( 57 )  |
| 4.5 程序控制指令.....             | ( 62 )  |
| 4.6 输入/输出 (I/O) 指令组 .....   | ( 65 )  |
| 4.7 CPU控制和位操作指令.....        | ( 67 )  |
| 4.8 关于Z80—CPU的中断.....       | ( 68 )  |
| 第五章 汇编语言程序设计.....           | ( 74 )  |
| 5.1 微型计算机使用的语言.....         | ( 74 )  |
| 5.2 程序设计步骤.....             | ( 83 )  |
| 5.3 测试、子程序与表.....           | ( 89 )  |
| 第六章 Z80—PIO并行输入/输出接口芯片..... | ( 100 ) |
| 6.1 概述.....                 | ( 100 ) |
| 6.2 Z80—PIO的引脚和框图 .....     | ( 100 ) |
| 6.3 PIO的初始化.....            | ( 107 ) |
| 6.4 PIO应用举例.....            | ( 116 ) |
| 第七章 Z80—CTC计算器/定时器芯片.....   | ( 125 ) |

|                           |       |
|---------------------------|-------|
| 7.1 概述.....               | (125) |
| 7.2 CTC 的方框图及引脚.....      | (125) |
| 7.3 CTC 的工作方式和初始化.....    | (130) |
| 7.4 CCT 的硬件连接.....        | (141) |
| 第八章 数据变换器件.....           | (142) |
| 8.1 概述.....               | (142) |
| 8.2 数据变换器的术语.....         | (144) |
| 8.3 $D/A$ 变换器的一般原理.....   | (147) |
| 8.4 $D/A$ 变换器的几种实际产品..... | (150) |
| 8.5 $A/D$ 变换器的一般原理.....   | (157) |
| 8.6 $A/D$ 变换器的几种实际产品..... | (160) |
| 附录.....                   | (175) |
| 算术运算及程序举例.....            | (175) |
| A.1 二进制与 $BCD$ 的变换.....   | (175) |
| A.2 二进制乘法.....            | (179) |
| A.3 二进制除法(8位数).....       | (181) |
| A.4 浮点运算.....             | (183) |
| B. 几种常用的数字逻辑电路.....       | (187) |
| B.1 逻辑电路.....             | (187) |
| B.2 运算电路.....             | (188) |
| B.3 触发器与寄存器.....          | (191) |
| B.4 存储器.....              | (196) |

# 第一章 微型计算机的基础知识

## § 1.1 引言

随着计算机应用的推广和普及,随着大规模集成电路技术的飞速发展,七十年代初诞生了一代新型的电子计算机——微型计算机(MicroComputer)。它利用大规模集成电路技术把计算机的中央处理单元(CPU——Central processing Unit)——即计算机的运算器和控制器集成在一个芯片上,称为微处理器(Microprocessor)典型的型号有8080、6800、Z80等。

同样利用大规模集成电路技术做成了容量相当大的存储器(Memory)芯片,同时又把各种通用的或专用的可编程序的接口电路集成在一个芯片上。

这样,把CPU配上一定容量的RAM(随机存取存储器)、ROM(只读存储器)和接口电路(如:PIO并行接口电路、SIO串行接口电路)以及必要的外部设备,如CRT(带键盘)、打印机等等,就构成了一个微型计算机(或称为微型计算机系统)。

在有些专用的场合,还把CPU和一定容量的RAM和ROM,以及I/O接口电路,集成在一个芯片上,就构成了单片计算机。

同样,把CPU、RAM和ROM及I/O接口电路装在一块印刷电路板上,就成为单板计算机。

总之,微型计算机是以利用大规模和超大规模集成电路技术为特征,大大缩小了计算机的体积,同时也大大的降低了成本。而在功能上却赶上甚至超过了60年代的小型机。因此微型机发展极为迅速,应用极为广泛,已经深入到工农业生产、国防、文教、科研、以及日常生活等各个领域。

## § 1.2 数制及不同数制之间的转换

### 一、十进制数 (Decimal)

在日常生活中,人们最熟悉的是十进制数。它的特点是:

(I) 它有十个不同的数字: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9。(即基数为十)

(II) 逢十进一。(或借一顶十)

(III) 它是一个有权数。在表示数时,这些处于不同位置(或数位)的数字代表的意义也不同。例如1001,表示一千零一。可见同样一个数字1,由于它处于不同位

置, 放在最高位时, 它表示1000, 放在最低位时, 它表示1。

(IV)、任何一个十进制数, 都可表示成基数十的各次幂的和式。

$$\text{即: } D_5 D_4 D_3 D_2 D_1 D_0 = D_5 \times 10^5 + D_4 \times 10^4 + D_3 \times 10^3 + D_2 \times 10^2 + D_1 \times 10^1 + D_0 \times 10^0$$

其中,  $D_i = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9$ 。

上式又称为按权展开式。

## 二、二进制数 (Binary)

在数字电子计算机中, 通常并不采用十进制数, 而是采用二进制。因为计算机中使用高电平和低电平来表示两个不同的数码: 1, 0。

类似十进制数, 二进制数亦有四个特点:

<I>、它有两个数字: 0、1。(即基数为二)。

<II>、逢二进位(或借一顶二)。

<III>、它是一个有权数。

<IV>、任何一个二进制数, 都可写成基数二的各次幂的和式。

$$\text{即: } B_5 B_4 B_3 B_2 B_1 B_0 = B_5 \times 2^5 + B_4 \times 2^4 + B_3 \times 2^3 + B_2 \times 2^2 + B_1 \times 2^1 + B_0 \times 2^0$$

其中  $B_i = 0, 1$ 。

如二进制数1001可以写成:  $1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$ 。因此数1001在二进制中, 表示十进制数九, 而不再是一千零一。

## 三、十六进制数 (Hexadecimal):

这种数制亦类似十进制数, 有四个特点:

<I>、它有十六个不同的数字: 0、1、2、3、4、5、6、7、8、9, A(10), B(11), C(12), D(13), E(14), F(15)。(即基数为十六)

<II>、逢十六进一。(或借一顶十六)

<III>、它是有权数。

<IV>、任何一个十六进制数, 都可写成基数十六的各次幂的和式。

$$\text{即: } H_5 H_4 H_3 H_2 H_1 H_0 = H_5 \times 16^5 + H_4 \times 16^4 + H_3 \times 16^3 + H_2 \times 16^2 + H_1 \times 16^1 + H_0 \times 16^0$$

其中:  $H_i = 0, 1, 2, 3, \dots, E, F$ 。

如:  $1001H = 1 \times 16^3 + 0 \times 16^2 + 0 \times 16^1 + 1 \times 16^0$

所以数1001在十六进制数中, 它不再代表一千零一, 而是表示十进制数4096。

在微型计算机中, 经常使用这种16进制数, 其理由如下:

1. 它与二进制数之间的转换比较方便。

$$\text{例如: } 10011100B = (1 \times 2^3 + 1 \times 2^0) \times 16^1 + (1 \times 2^3 + 1 \times 2^2) \times 16^0 = 9CH$$

即每四个二进制数对应于一个十六进制数。微型机中的二进制数通常是8位或8位的。

整倍数（如16位、32位、64位），则相应的十六进制数为2位、4位、6位、8位等。

2、使用二进制，书写太长，不易记忆，且念起来不易懂。而使用十六进制数可以弥补上述缺点。

综上所述几种计数制，可以把它们的特点归纳如下：

<I>、每一种计数制都有一个固定的基数J，它的每一位可能取J个不同的数值。

<II>、它是逢“J”进位的（或借1顶“J”）。

<III>、它是一个有权数，它的每一个数位i，对应一个固定的值 $J^i$ ，而 $J^i$ 就称为该位的“权”。小数点左面各位的权依次是基数J的正次幂；而小数点右边各位的权依次是基数J的负次幂。

<IV>、它们都可写成基数J的各次幂的和式。

$$\text{即：} X = \sum_{i=-n}^{+m} X_i \cdot J^i$$

表1.1 二进制、十进制、十六进制数码对照表

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

#### 四、不同数制之间转换

<一>、十进制整数转换成二进制整数

[方法]：除二取余法，除到商是零为止，第一次余数（0或1）为二进制数的最低位。

如：(15)<sub>10</sub> = (1111)<sub>2</sub>

$$\begin{array}{r}
 2 \overline{) 15} \\
 \underline{2 \overline{) 7}} \quad \dots\dots \text{余 } 1 \text{ (最低位)} \\
 \underline{2 \overline{) 3}} \quad \dots\dots \text{余 } 1 \\
 \underline{2 \overline{) 1}} \quad \dots\dots \text{余 } 1 \\
 0 \quad \dots\dots \text{余 } 1
 \end{array}$$

<二>、十进制小数转换成二进制小数

[方法]：乘二取整法，乘到小数部分为零（或达到一定位数）为止。整数0（或1）为二进制小数的最高位。一般情况下，将十进制小数转换成二进制小数只能取其近似值。

如：  $(0.125)_{10} = (0.001)_2$

$$\begin{array}{r}
 0.125 \\
 \times 2 \\
 \hline
 0.250 \dots \dots \text{整数部分为 } 0 \text{ (最高位)} \\
 \times 2 \\
 \hline
 0.500 \dots \dots \text{整数部分为 } 0 \\
 \times 2 \\
 \hline
 1.000 \dots \dots \text{整数部分为 } 1
 \end{array}$$

又如：  $(0.68)_{10} = (0.1010111)_2$

$$\begin{array}{r}
 0.68 \\
 \times 2 \\
 \hline
 1.36 \dots \dots \text{整数部分为 } 1 \text{ (最高位)} \\
 \times 2 \\
 \hline
 0.72 \dots \dots \text{整数部分为 } 0 \\
 \times 2 \\
 \hline
 1.44 \dots \dots \text{整数部分为 } 1 \\
 \times 2 \\
 \hline
 1.44 \dots \dots \text{整数部分为 } 1 \\
 \times 2 \\
 \hline
 0.88 \dots \dots \text{整数部分为 } 0 \\
 \times 2 \\
 \hline
 1.76 \dots \dots \text{整数部分为 } 1 \\
 \times 2 \\
 \hline
 1.52 \dots \dots \text{整数部分为 } 1 \\
 \times 2 \\
 \hline
 1.04 \dots \dots \text{整数部分为 } 1
 \end{array}$$

### 〈三〉、将二进制数转换成十进制数

这种方法比较简单，只须根据二进制数的第四个特点，把它按权展开，然后相加即可。

$$\begin{aligned}
 \text{如： } (1111.1010111)_2 &= 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 \\
 &\quad + 1 \times 2^{-1} + 1 \times 2^{-3} + 1 \times 2^{-5} + 1 \times 2^{-6} + 1 \times 2^{-7} \\
 &= 8 + 4 + 2 + 1 + 0.5 + 0.125 + 0.03125 + 0.15625 + 0.0078125 \\
 &= (15.6796875)_{10}
 \end{aligned}$$

$$(1111.1010111)_2 = (15.6796875)_{10}$$

### （四）、十进制整数转换为十六进制整数。



[方法]：除16取余法。（类似十进制整数转换为二进制整数方法）

如： $(25)_{10} = 19H$

$$16 \overline{) 25}$$

$16 \overline{) 1} \dots\dots \text{余 } 9 \text{ (最低有效位)}$

$0 \dots\dots \text{余 } 1$

(五)、十进制小数转换为十六进制小数

[方法]、乘16取整法。（类似十进制小数转换为二进制小数方法）

如： $(0.55)_{10} \approx 0.814H$

$$0.55$$

$$\times 16$$

$$\hline 330$$

$$+ 55$$

$8.80 \dots\dots$  整数部分为 8 (最高有效位)

$$\times 16$$

$1.28 \dots\dots$  整数部分为 1

$$\times 16$$

$4.48 \dots\dots$  整数部分为 4

则  $(0.55)_{10} \approx 0.814H$

(六)、十六进制数转换为十进制数方法类似二进制数转换为十进制数。

如： $15.4H = (21.25)_{10}$

即： $1 \times 16^1 + 5 \times 16^0 + 4 \times 16^{-1} = 16 + 5 + 0.25 = 21.25$

(七)、二进制数与十六进制数之间的转换

在微型计算机中，目前通用的字长为 8 位，因此可用两位 16 进制数表示，故 16 进制数在微型机中应用十分普遍。

因为  $2^4 = 16$ ，即四位二进制数码（0 或 1）的组合状态有 16 种，所以可以用四位二进制数码表示一位 16 进制数。只要熟悉了这种联系之后，二进制数与十六进制数之间的转换也是十分方便的。

1、二进制数转换为 16 进制数。

[方法]：把二进制数的整数部分由小数点向左、每四位一分，最后不足四位的在前面补 0；小数部分由小数点向右，每四位一分，最后不足四位的后面补 0。然后，把每四位二进制数用相应的 16 进制数代替，即可转换为 16 进制数。

如：(1 1011 1110 0011. 1001 1011 1) 可转换为：

$$0001 \ 1011 \ 1110 \ 0011. 1001 \ 1011 \ 1000$$

↓

↓

↓

↓

↓

↓

↓

1

B

E

3

·

9

B

8

即： $(1101111100011.100110111)_2 = 19BE3.9B8H$

2、十六进制数转换为二进制数

[方法]：不论是 16 进制整数或小数，只要把每一位 16 进制数用相应的四位二进制

数代替即可。

如：5AFH可转换为：

|      |      |      |
|------|------|------|
| 5    | A    | F    |
| ↓    | ↓    | ↓    |
| 1001 | 1010 | 1111 |

所以，5AFH = (100110101111)<sub>2</sub>

又如：D. 56AH可转换为：

|      |      |      |      |
|------|------|------|------|
| D    | 5    | 6    | A    |
| ↓    | ↓    | ↓    | ↓    |
| 1101 | 0101 | 0110 | 1010 |

所以：D. 56AH = (1101.010101101010)<sub>2</sub>

## § 1.3 数的定点和浮点表示

在计算机技术中，采用两种规定小数点位置的方法：定点法和浮点法。在定点法中，小数点的位置固定不变，这样表示的数叫做定点数。在浮点法中，小数点的位置是变动的，这样表示的数，叫做浮点数。与此相应，计算机可分为定点机和浮点机。

### 1. 定点数

定点数的小数点位置一般固定在最高位之前，或者固定在最低位之后。在第一种情况，计算机中的数总是小于1的小数；在第二种情况，计算机中的数总是整数。

小数点的位置一旦规定好，所有参加运算的操作数，都要用比例因子化成所要求的格式。既然所有操作数的小数点都在同一位置，那就可以直接在运算器中进行加减运算，不必事先对位，所以实现这种运算的电路比较简单。但是计算结果还要用比例因子折算成真实值。这又增加了编程的工作量。

定点数所能表示的数的范围，决定于字长，在字长较短的情况下，数的范围较小，因而计算精度不高。

当字长为  $n$  位时，小数点规定在最高位之前，则此时定点数  $N$  的取值范围是：

$$2^{-n} \leq N \leq 1 - 2^{-n}.$$

如果小数点规定在最低位之后，则此时定点数的取值范围是：  $1 \leq N \leq 2^n - 1$ 。

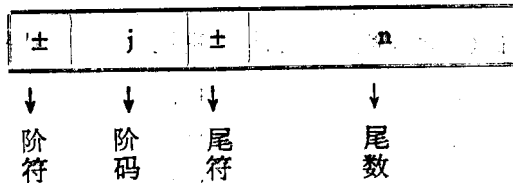
若运算的数小于  $N$  的下限值，机器把它当成 0 处理。而参加运算的数大于  $N$  的上限值时，称为“溢出”。对溢出的处理，各种机器可能是不同的。在以上的定点数讨论中，没有涉及数的正负号，称这种数为无符号数。在多数情况下，处理的数是带有符号的数。定点数的符号位放在数的最高位之前。且规定符号位为 0 表示正，符号位为 1 表示负。

## 2. 浮点数

浮点数是把一个二进制数  $N$  表示成如下形式:  $N = \pm n \times 2^{\pm j}$

其中  $n$  称为尾数, 其值为  $0 < n < 1$ , 它前面的符号称为尾符, 而  $j$  称为阶码, 是二进制整数, 它前面的符号称为阶符。

浮点数在机器中的格式如下图所示。



如二进制数 1011.1101 可写成如下图点形式:

$$\begin{aligned} &+ 0.10111101 \times 2^{+100} \\ &+ 0.010111101 \times 2^{+101} \\ &+ 0.0010111101 \times 2^{+110} \end{aligned}$$

从上面所举的浮点数的例子可以看到, 阶 (包括它的正负号) 指出小数点的位置。改变阶的值, 就是改变了数的小数点位置。所以, 浮点数体现出小数点可以浮动的概念。尾符表示整个数是正数还是负数。

在机器字长相同的情况下, 浮点数表示的数值范围比定点数要大得多。所以用浮点数进行科学计算精度大大提高。

需要指出的是, 在定点机上进行浮点运算, 需要用程序来实现。

## § 1.4 码 制

### 一、带符号数的表示方法

#### 1. 机器数与真值

上面提到的二进制数, 没有提到符号问题, 所以它是一种无符号数。但是, 在机器中, 数显然是会有正有负的, 那么符号是怎样表示的呢? 通常一个数的最高位规定为符号位。即若是字长为 8 位的, 则  $D_7$  为符号位,  $D_6 \sim D_0$  为数字位。符号位用 0 表示正, 用 1 表负。如:  $(01011011)_2 = +(91)_{10}$ , 而  $(11011011)_2 = -(91)_{10}$ 。

这样连同一个符号位在一起的作为一个数看待, 就称为机器数; 而它的数值部分为机器数的真值。

为了机器结构简单, 运算方便, 把减法变为加法, 所以, 在机器中的负数有三种表示方法——原码、反码、补码。

#### 2. 原码

按上所述, 负数的符号位用 1 表示, 这样的表示方法就称为负数的原码。

如:  $X = -105$ , 则  $[X]_{\text{原}} = 11101001$

#### 3. 反码

负数的反码表示法，就是将负数的原码符号不动，其数字部分每位对应变反（即0变1、1变0）。

如：[11101001]<sub>反</sub> = 10010110

八位二进制负数的反码表示如表1.2所示：

表1.2 数的表示法

| 二进制数码表示   | 原码(十进制数) | 反码(十进制表示) | 补码(十进制表示) |
|-----------|----------|-----------|-----------|
| 1000 0000 | - 0      | - 127     | - 128     |
| 1000 0001 | - 1      | - 126     | - 127     |
| 1000 0010 | - 2      | - 125     | - 126     |
| 1000 0011 | - 3      | - 124     | - 125     |
| ⋮         | ⋮        | ⋮         | ⋮         |
| 1111 1101 | - 125    | - 2       | - 3       |
| 1111 1110 | - 126    | - 1       | - 2       |
| 1111 1111 | - 127    | - 0       | - 1       |

#### 4、补码

负数的补码表示法，就是将负数的反码在最低位加1所形成。

如表1.2所示。

## 二、BCD码（二——十进制数）

所谓BCD码，就是用四位二进制数对十进制数的每一位进行编码。

如：(97)<sub>10</sub> = (1001 0111)<sub>BCD</sub>

但须注意，这种数制与二进制数不同，它的数的权不同。

如：(97)<sub>10</sub> = (0110 0001)<sub>2</sub>

这种数制（或码制）的优点是既照顾了人们使用十进制数的习惯，又考虑到计算机的特点。微型机的输入、输出经常采用这种码制。

若是简单的运算，可用BCD码直接进行。如果运算较为复杂，则应将输入的BCD码转换成二进制数，进行运算后，再转换成BCD码，以便输出。

## 三、ASCII码（字符信息的编码）

字符一词在这里指的是字母、数字以及其它一些符号的总称。如：A、B、1、2、+、-、>等等，都看做是字符。

大家都知道，计算机是在二进制数的基础上组织起来的，它只能识别0和1两种符号。要想把各种字符输入到计算机中，必须用二进制数对它们进行编码。对于输出也是如此。

现代数字电子计算机广泛采用美国信息交换标准代码——ASCII 码。它用七位二进制数编码。

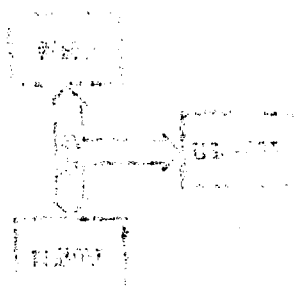
如：A 的 ASCII 码为：41H

B 的 ASCII 码为：42H

1 的 ASCII 码为：31H

2 的 ASCII 码为：32H

详见 ASCII 码表。



## 第二章 Z80-CPU 的结构及定时

### § 2.1 概述

Z80-CPU是美国 Zilog 公司研制生产的超大规模集成电路的微处理器芯片。它是在充分吸取了美国 Intel 公司生产的8080A 微处理器芯片的经验后研制的。所谓CPU, 是中央处理单元 (Central Processing Unit) 的简称, 它是微型计算机的核心。微型计算机的算术运算、逻辑判别, 数据传送等功能均需由 CPU 来担任, 因此一般在讨论微型计算机方面的基础知识时, 总是首先讨论CPU。

如果将微型计算机加以简化, 可以认为只由CPU、存储器、输入/输出(简称I/O)接口这三个主要部分组成。三者通过总线加以联系, 如图2.1所示。数据或信息通过 I/O 接口送入微型机, 根据指令经总线向CPU传送, 或送存储器存储。送入 CPU 的数据可以按指定要求进行算术运算或逻辑判别。CPU 将运算或判别的结果, 可以送入存储器存起来, 或再经I/O接口向外输出。所以概括起来说, CPU的功能是对数据(或信息)进行运算或传送, 以及对CPU本身或对存储器和I/O接口进行控制。可以说, CPU 是整个微型计算机中的中枢。

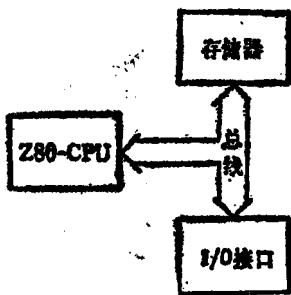


图2.1微计算机简化示意图型

Z80-CPU与8080A CPU相比较, 作了三个方面的改进。首先是减少了芯片数目, 由原来的三种芯片减为单一CPU芯片。其次是采用一种+5V电源。再就是在指令系统功能方面增加了中断、变址寻址、数据块传送及位操作等功能, 在结构上增加了辅助寄存器、中断矢量寄存器及刷新计数器等。但是为了便利用户, Z80-CPU在软件上与8080ACPU是兼容的。在Z80-CPU投入市场不久, Intel也研制成了与Z80-CPU功能相当的8085CPU。目前还研制成了功能更强的8088。而美国 Motorola 公司

也在原来的M6800CPU的基础上研制成了M6809, 这大概是目前8位CPU中功能最强的了。8位的CPU芯片国外还有些别的公司生产, 但都不如前三者应用广泛。

CPU的操作是按一定节奏(即同步脉冲)进行的。而这节奏的快慢就是时钟频率。Z80-CPU的时钟频率有两种: 2 MHz (Z80-CPU) 和4 MHz (Z80A-CPU)。TP-801用的CPU是Z80-CPU, 即钟频是2 MHz的那一种。

图2.2示出Z80-CPU芯片的结构方框图。主要由运算逻辑单元、各种寄存器、计数器、译码器及控制器组成。本章下列各节将从使用Z80-CPU的角度作介绍, 只叙述其功能, 不涉及芯片内部逻辑电路的构造。也就是说, 从编写程序的角度来观察CPU的各组成部分。在总线结构一节对CPU的40条引线(Pin)的功能作简要叙述, 以便单独

使用Z80-CPU时可以进行硬件方面的设计。最后一节介绍Z80-CPU的定时。

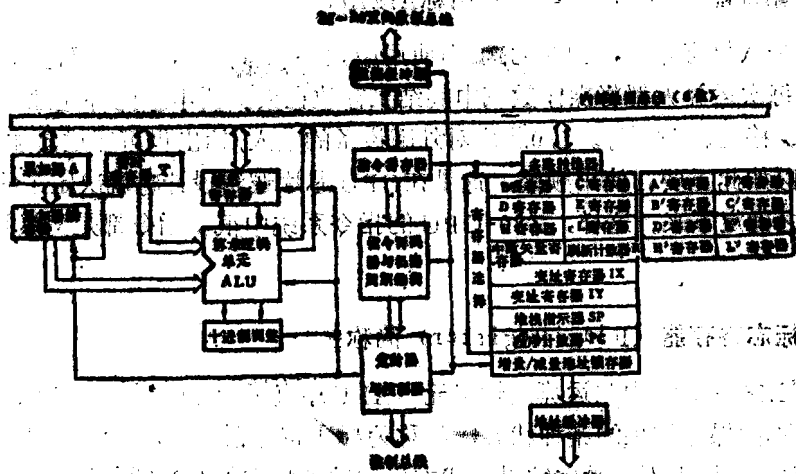


图2.2 Z80-CPU方框图 A15~A0地址总线

## § 2.2 寄存器

Z80-CPU中共有208位寄存器可供程序员使用。它们实质上是由二态存储单元所组成的静态读/写存储器（RAM）。两组六个工作寄存器可以单独用作8位的寄存器或成对地用作16位寄存器。另有两组累加器及标志寄存器。其余尚有四个16位专用寄存器及两个专供中断及刷新用的寄存器。需要指出，这里介绍的寄存器是在今后编制程序所必需了解的，而不包括CPU中所有作数据暂存用的寄存器（参阅图2.2）

### 一、通用寄存器

Z80-CPU中通用寄存器分主寄存器（Main Register Set）及辅助寄存器（Alternate Register Set）。前者包括累加器A、标志寄存器F及六个8位寄存器B、C、D、E、H、L。后者包括八个与主寄存器相对应的8位寄存器A'、B'、C'、D'、E'及H'、L'。

通用寄存器能在程序直接控制下工作，利用通用寄存器可以暂存参加运算的操作数和中间结果，而不需要访问存储器。在需要存储16位数据或地址时可以成“对”使用，即BC、DE、HL合起来用。在习惯上常将HL作存储器的地址寄存器，将BC作计数器来使用。这与开始使用8080的指令系统的习惯有关。在Z80-CPU的指令系统中数据的传送或运算只是在主寄存器间（或与存储器）进行，辅助寄存中内容必须通过交换指令，调入主寄存器后才能参加运算或传送。

累加器A及标志寄存器虽然都属于通用寄存器，但其功能各有特点。下面分别作些介绍。

## 二、累加器 (Accumulator) 简称A或Acc

它是个八位寄存器，Z80-CPU中使用最频繁的寄存器，与运算逻辑单元(ALU)直接相连，每次运算大都将操作数或中间结果存放在累加器中。此外还通过ALU来实现逻辑操作、移位、循环等。I/O芯片对外围设备的输入/输出也可通过累加器来实行。累加器之所以如此重要、使用频繁，这是在CPU芯片逻辑设计时就决定了的，CPU的指令也就随之确定下来。这样，Z80-CPU的指令系统中很多指令都是与累加器有关系的了。

## 三、标志寄存器 (Flag Register) 简称F

它是在Z80-CPU中与运算逻辑单元(ALU)直接相连的另一个8位寄存器，作为反映ALU不同操作结果的某种状态，故称为标志寄存器(或称为状态寄存器)。其中有6个状态位，标志不同操作结果。状态位的排列状态如图2.3所示。

D<sub>7</sub> D<sub>6</sub> D<sub>5</sub> D<sub>4</sub> D<sub>3</sub> D<sub>2</sub> D<sub>1</sub> D<sub>0</sub>

|   |   |   |   |   |     |   |    |
|---|---|---|---|---|-----|---|----|
| S | Z | × | H | × | P/V | N | CY |
|---|---|---|---|---|-----|---|----|

图2.3 Z80-CPU的标志寄存器的状态位

这些标志分两类，一类标志叫做测试标志，主要是向CPU提供某种判别的条件，以用来作为执行程序的转移，调用或返回的条件。如根据某个标志位的状态测试结果是“0”还是“1”，来决定是否执行某种预定的操作，另一类标志叫做非测试标志，主要用于二~十进制码(BCD码)运算。

测试标志有：

测试标志有：

### 1. 符号标志 (Sign) 简称S

在带符号的数(如二进制补码)运算时，若操作结果为负，则将此符号位置位(置1)；反之，则将它复位(置0)。由于在8位微型机中一般用最高位(D<sub>7</sub>)代表数的符号，所以这个标志反映了累加器中最高位(D<sub>7</sub>)的状态。

### 2. 零标志 (Zero) 简称Z

如果操作结果是零，则将此标志位置位(置1)，这对算术运算及逻辑运算都适用。例如，检查输入/输出设备在数据传送中是否出错，这时只要将再次传送的数据与原来的数据进行异或逻辑运算(XOR)或比较运算(CP)。若此时操作结果为零，则表示两次传送结果完全相同。否则其中至少有一位出错。

### 3. 进位标志 (Carry) 简称CY或C

当进行两个数据相加时，最高位产生进位，则将它置位(置1)；反之，则将它复位(置0)。做减法运算需要借位时，CY也被置1。此外在执行移位及循环指令时，也改变了CY标志状态。而且根据需要可以直接用程序使其置1或置0。

### 4. 奇偶/溢出标志 (Parity/Overflow) 简称P/V

这个标志位有两个用途：一是用于逻辑运算(即“与”、“或”、“异或”运算)结果的奇偶性，当结果中有偶数个“1”，则将它置位；反之，则将它复位(置0)。一是用于算术运算时是否发生二的补码溢出，如果发生溢出，则此标志位置1，否则



置0。

所谓溢出是算术运算结果超过了用二进制数表示数的范围(即+127~-128)。这是因为一般8位微型机均用最高位表示数的符号,最高位为1是负数。

如两个符号相同的数相加,而和的符号却与此符号不同,则说明发生溢出。

例如

| 符号位 |                        | 符号位 |                        |
|-----|------------------------|-----|------------------------|
| +)  | 0 111, 1000 (+120)     | +)  | 1 000, 1001 (-119)     |
|     | 0 110, 1001 (+150)     |     | 1 001, 1011 (-101)     |
|     | 和 1 110, 0001 (-95) 溢出 |     | ① 0 010, 0100 (+36) 溢出 |
|     |                        | CY  |                        |

而两个符号不同的数相加,一般不会产生溢出。

若两个符号相同的数相减,一般不会产生溢出。而当符号不同的数相减时,差的符号与被减数符号相反,则产生溢出。

例如

|     |                         |    |                         |
|-----|-------------------------|----|-------------------------|
| -)  | 1 001, 0111 (-73)       | -) | 0 011, 1011 (+59)       |
|     | 0 011, 1011 (+59)       |    | 1 011, 0111 (-73)       |
|     | ↑ 0 111, 1100 (+124) 溢出 |    | ① 1 000, 0100 (-124) 溢出 |
| 无借位 |                         | 借位 |                         |

溢出条件还可以归纳为下列表达式,即

$$P/V = C_{10} \oplus CY$$

其中 $C_{10}$ 表示和(或差)的 $D_0$ 向 $D_1$ 进位(或借位),CY表示进位标志置位非测试标志。

5. 半进位标志 (Halfcarry) 简称H

在进行二~十进制(BCD码)运算中,低半字节(低四位)向高半字节(高四位)进位(或借位),则此标志置1。这个标志用于校正BCD码加法或减法的结果。

6. 减标志 (Nagate) 简称N

在进行BCD码运算时,十进制调整指令DAA利用此标志来区别加法或减法,从而作出应有的修正。如果DAA前一次操作是减法,则此N标志置1,反之则置0。

当累加器A和标志寄存器当作一个寄存器对来使用时,称程序状态字(Program State Word),或处理器状态字(Processor State Word)简称PSW。Z80-CPU中有两组独立的累加器A, A'和标志寄存器F, F'。可以有两个状态字PSW及PSW'。

#### 四、算术逻辑单元 (Arithmetic and Logic Unit) 简称ALU

算术逻辑单元又称运算器,其主要功能除完成各种算术如加、减、增1、减1和逻辑