Chapter 5

第5章 指令系统

本章学习目标

- 了解指令系统的格式。
- 了解指令系统的编码。
- 掌握 8086 指令系统的格式及功能。
- 掌握 8086 汇编语言程序设计的方法及应用。

本章首先介绍计算机指令系统的格式、寻址方式、编码等基本概念，然后重点介绍 8086 指令系统的格式及功能，最后介绍 8086 汇编语言程序设计的方法及应用。

5.1 指令系统概述

指令系统是指计算机所具有的各种指令的集合，它是软件编程的出发点和硬件设计的依据，反映了计算机硬件具有的基本功能。

5.1.1 指令的格式

机器指令是计算机硬件能够识别并直接执行的操作命令。指令由操作码和地址码两部分组成。操作码用来说明指令操作的性质及功能。地址码用来描述该指令的操作对象，由它给出操作数或操作数的地址以及操作结果的存放地址。

根据指令中所含地址码的个数，指令分为零地址指令、单地址指令、双地址指令、三地址指令和多地址指令。

一般来说，地址码较少的指令占用空间小，执行速度快，所以在结构简单的计算机指令系统中，零地址、单地址和双地址指令较多采用。而在功能较强的计算机中多采用双地址、三地址及多地址指令。

5.1.2 寻址方式

指令中的地址码部分指明了指令的操作数或操作数的地址以及操作结果的地址。这种指定操作数或者操作数的地址以及操作结果地址的方式称为寻址方式。

在指令中,需要设定源地址码和目的地址码。源地址码是操作数的地址,目的地址码是运算结果的地址。

计算机指令系统中常用的寻址方式有立即寻址、寄存器寻址、直接寻址、寄存器间接寻址、相对寻址、基址和变址寻址、隐含寻址等方式。为了方便讲解,本节以 8086 指令系统为例。

在 8086 指令系统中,指令中地址码的个数有零地址、单地址和双地址 3 种。在双地址指令中,目的地址码在前,源地址码在后。首先介绍常用的 MOV 指令。MOV 指令的格式为

```
MOV  目的地址码,源地址码
```

MOV 指令的功能是将源地址码指定的操作数传送到目的地址码。MOV 指令能实现将立即数送寄存器或存储器,实现寄存器和寄存器间、寄存器和存储器间数据的传送。目的地址码和源地址码可以使用不同的寻址方式。MOV 指令操作数的类型由寄存器长度确定,或者由类型操作符指定。

1. 立即寻址方式

在指令中直接给出操作数本身。操作数作为指令的一部分,在读取指令的时候就把数据读取了出来,这个操作数又称为立即数。这种方式不需要再根据地址寻找操作数,所以指令的执行速度较快。

在 8086 指令系统中,立即数是常量,可以是各种进制的数据、字符、字符串,还可以是数值表达式或符号常量。

【例 5-1】 在 8086 指令系统中,写出指令 MOV AH,20H 的执行结果。

解:这条指令中的源地址码是一个立即数,使用的是立即寻址方式。指令功能是把 8 位立即数 20H 传送到 AH 寄存器中。指令执行结果为 AH=20H。

2. 寄存器寻址方式

指令的地址码指定操作数所在的寄存器,这种方式称为寄存器寻址方式。寄存器数量少,只需少量的编码就可以表示寄存器,所以可以减少整个指令的长度。另外,寄存器中的操作数已经在 CPU 中,因此指令的执行速度较快。

在 8086 指令系统中,寄存器寻址方式中的地址码为寄存器的名称。

【例 5-2】 在 8086 指令系统中,已知 AX=1234H,BX=5678H,写出指令 MOV AX,BX 的执行结果。

解:指令 MOV AX,BX 的源地址码是寄存器 BX,目的地址码是 AX,都采用寄存器寻址方式。指令功能是把 16 位寄存器 BX 中的数值传送到 AX 寄存器。指令的执行结果为

AX＝5678H。

3. 直接寻址方式

指令中的地址码给出的是操作数所在的单元的实际地址，又称为有效地址。根据指令中的有效地址只需访问内存一次便获得操作数。这种寻址方式简单、直观，便于硬件实现。但是随着存储空间的不断增大，地址码会越来越长，会增加指令的长度。

在8086指令系统中，直接寻址的地址码形式为[偏移地址]，或者为表示单元地址的符号变量。指令中直接给出的是段内偏移地址，默认段地址在DS中。

【例5-3】 在8086指令系统中，若已知AX＝1234H，内存数据段单元(2000H)＝11H，(2001H)＝22H。写出指令MOV AX,[2000H]的执行结果。

解：MOV AX,[2000H]的源地址码是[2000H]，是直接寻址方式；目的地址码是寄存器AX，是寄存器寻址方式。指令中AX是16位寄存器，所以操作数的类型是16位。指令的功能是从数据段2000H单元中取16位字传送到AX寄存器。源操作数的有效地址是2000H。指令的执行结果为AX＝2211H。

4. 寄存器间接寻址方式

指令中的地址码给出寄存器的名称，寄存器中存放操作数的有效地址。因为只需给出寄存器的名称，所以指令的长度较短；但是因为要根据寄存器中的有效地址访问内存才能得到操作数，指令的执行时间比寄存器寻址方式长。

在8086指令系统中，只有BX、BP、SI、DI这4个寄存器可以使用寄存器间接寻址方式。寄存器间接寻址方式的地址码形式为[BX]、[BP]、[SI]、[DI]。寄存器中为有效地址，一般根据默认原则确定段地址。如果段内偏移地址在BX、SI、DI中，则默认段地址在DS中；如果段内偏移地址在BP中，则默认段地址在SS中；在串处理指令中，如果段内偏移地址在DI中，则默认段地址在ES中。如果不采用默认的段地址，则可以在指令中指定段地址，这种指定段地址的指令称为段跨越指令。

【例5-4】 在8086指令系统中，若已知AX＝1234H，BX＝2000H，内存中数据段单元(2000H)＝11H，(2001H)＝22H。写出指令MOV AX,[BX]的执行结果。

解：指令MOV AX,[BX]的源地址码是[BX]，是寄存器间接寻址方式。BX中的值为有效地址，由BX确定段地址在数据段寄存器中，所以源操作数在内存数据段2000H单元。指令的功能是从内存数据段2000H单元中取16位字传送到AX寄存器。2000H单元的字数据是2211H。指令执行结果为AX＝2211H。

5. 存储器间接寻址方式

指令的地址码部分给出的是存放操作数地址的存储单元地址。存储器间接寻址方式至少要访问两次内存才能取出操作数，因此指令执行速度减慢。

在8086指令系统中没有存储器间接寻址方式。在欧姆龙的指令系统中，用@表示存储器间接寻址。指令MOV ＃10 D0表示将立即数10传送到D0存储区。MOV ＃FFFF @D0表示将立即数传送到D10存储区，指令中的D0不是最终目的地，而是用D0区内的10表示目的地址。

6. 相对寻址方式

指令的地址码中给定一个位移量，将程序计数器(PC)中的值加上这个位移量，得到操作数的有效地址。在这种寻址方式下，被访问的操作数的地址是不固定的，但是该地址相对于程序的位置却是固定的，所以不管程序装入到内存的什么地方，都可以正确读取操作数。这样可以实现与地址无关的程序设计。

在 8086 指令系统中，指令代码中含有一个相对位移量。这个位移量是目的指令和当前指令的地址差，是一个带符号的数。执行指令时，转移的目的指令地址是当前的 IP 值加上指令中的位移量。当位移量是 8 位带符号数时，位移量在±127B 范围内，是段内直接短转移；当位移量是 16 位带符号数时，位移量在±32KB 范围内，是段内直接近转移。

【例 5-5】 在 8086 指令系统中，若已知 IP＝1234H。转移指令 JMP ＋5 的执行结果是什么？

解：JMP 指令表示程序跳转到新指令执行。新指令的地址由 IP 的值加位移量 5 计算得到，所以程序会跳转到 1239H 单元，执行程序段 1239H 单元的指令。

7. 基址或变址寻址方式

在指令地址码中给出基址或变址寄存器的名称和一个形式地址，操作数的有效地址由寄存器中的值和形式地址相加得到。采用基址寄存器时称为基址寻址，采用变址寄存器时称为变址寻址。

在 8086 指令系统中，基址或变址寻址方式的地址码形式为“[基址寄存器名][变址寄存器名]＋位移量”或者“位移量[基址寄存器名][变址寄存器名]”或者“[基址寄存器名 ＋变址寄存器名＋位移量]”，这些都是等价的书写形式。由基址寄存器或变址寄存器确定默认的段地址。

【例 5-6】 在 8086 指令系统中，若已知 AX＝1234H，BP＝2000H，SI＝0001H，BX＝2000H，内存数据段单元(2003H)＝11H，(2004H)＝22H，内存堆栈段单元(2003H)＝33H，(2004H)＝44H，写出指令 MOV AX，[BP][SI]＋2 和 MOV AX，[BX][SI]＋2 的执行结果。

解：指令 MOV AX，[BP][SI]＋2 的源地址码有一个基址指针 BP、一个变址寄存器 SI 以及一个位移量 2，是基址变址寻址方式。BP 中的 2000H 加 SI 中的 0001H 加位移量 2，得到有效地址 2003H。由 BP 确定段地址在堆栈段寄存器中。到堆栈段访问 2003H 单元，取得字数据 4433H，传送到 AX 中。指令执行结果为 AX＝4433H。

指令 MOV AX，[BX][SI]＋2 的源地址码有一个基址寄存器 BX、一个变址寄存器 SI 以及一个位移量 2，是基址变址寻址方式。BX 中的 2000H 加 SI 中的 0001H 加位移量 2，得到有效地址 2003H。由 BX 确定段地址在数据段寄存器中。到数据段访问 2003H 单元，取得字数据 2211H，传送到 AX 中。指令执行结果为 AX＝2211H。

8. 隐含寻址方式

指令中不指出操作数地址，而是在操作码中隐含着操作数的地址。

【例 5-7】 8086 指令系统中的 CBW 指令不需要指定操作数，系统默认操作数为 AL。

该指令功能是将 AX 的低 8 位寄存器 AL 的值扩展到 AX 寄存器中。若已知 AX＝1234H，写出指令 CBW 的执行结果。

解：AX＝0034H，将低 8 位的最高位符号位扩展到整个 AX 寄存器中。

一台计算机的指令系统寻址方式多种多样，给程序员编程带来了方便，但也使得计算机控制器的实现具有一定的复杂性。对于一台计算机而言，可能采用上述的一些寻址方式、这些基本寻址方式的组合或稍加变化。

5.1.3　指令类型

指令系统的功能决定了一台计算机的基本功能。不同类型的计算机硬件功能不同，具有不同的指令集合，但是有些类型的指令是共同的。常见的指令类型有以下几种：

(1) 数据传送类指令。完成数据在主存和 CPU 寄存器之间的传输。

(2) 算术运算类指令。对数据进行算术运算，包括加法、减法、乘法、除法等算术运算。有些性能较强的计算机还具有浮点运算指令等。

(3) 逻辑运算类指令。对数据进行逻辑运算，包括逻辑与、或、非、异或等运算。这些逻辑运算对数据进行位运算，位和位之间没有进位传递关系。有些计算机还有位操作指令，如位测试、位清除等。

(4) 移位操作类指令。移位操作是对数据进行相邻数据位之间的传递操作，分为算术移位、逻辑移位和循环移位 3 种，每种移位操作又分为左移和右移。

(5) 程序控制类指令。用于控制程序执行的顺序和方向，主要包括条件转移指令、无条件转移指令、循环指令、子程序调用和返回指令、中断指令等。

(6) 输入输出操作指令。完成主机和外围设备之间的信息传送。有的计算机有专门的输入输出指令，有的则把外设接口看作特殊的存储器单元，用传送类指令实现访问。

(7) 串操作指令。针对主存中连续存放的一系列字或字节，完成串传送、串比较、串查找等功能。串可以由非数值数据(如字符串)构成，可以方便地完成字符串处理。

(8) 处理器控制指令。直接控制 CPU 以实现某种功能，如空操作指令、停机指令等。

以上的指令种类，在一台计算机指令系统中并不是全部具备。例如，有的计算机没有串处理指令。

5.2　8086 指令系统

8086 指令系统按功能可分为 7 类：处理器控制类指令、数据传送类指令、算术运算类指令、位操作类指令、串操作类指令、控制转移类指令、中断指令。

8086 指令系统中的指令格式不同，有的指令还有特殊的约定。但是这些指令也有一些统一的规定：

- 指令中的两个操作数不能同时为存储单元。
- 指令中的两个操作数不能同时为段寄存器。

- 目的操作数不能是立即数。
- 指令中的操作数必须有类型，即长度是字节还是字必须是明确的。立即数类型不明确，如数据 5，可以是 8 位的 05H，也可以是 16 位的 0005H。未定义过类型的存储单元类型也不明确，如 2000H 单元，可能表示 2000H 字单元，也可能表示 2000H 字节单元。
- 指令中的操作数类型必须一致。必须都为字节类型，或者都为字类型。
- 目的操作数要避免使用 CS 和 IP。CS、IP 用于指示程序中要执行的指令地址，如果直接修改 CS、IP 的值，可能会造成微处理器不能正常工作。

在汇编语言程序中，可以给一条指令所在的单元命名，称为标号。一个标号代表了一条指令的地址。标号写在指令的前面，用“:”分界。在同一个程序中，标号名必须是唯一的。

在汇编语言程序中，“;”后的部分作为注释，用于方便程序的阅读，计算机不会执行注释。

为了方便指令的讲解，下面的例题中，在指令后面用注释方式给出答案或者分析，并在必要的时候给指令加序号。注意，汇编语言程序中一行只能写一条指令且没有序号。

5.2.1 处理器控制类指令

处理器控制类指令完成对标志寄存器中标志位的处理以及对处理器的控制。这类指令都只有操作码。

1. 标志位处理指令

(1) CF 清 0 指令：CLC。

(2) CF 置 1 指令：STC。

(3) CF 取反指令：CMC。

(4) DF 清 0 指令：CLD。

(5) DF 置 1 指令：STD。

(6) IF 清 0 指令：CLI。

(7) IF 置 1 指令：STI。

2. 处理器控制指令

(1) 空操作指令：NOP。

指令功能：不执行任何操作。调试程序时主要用于占据一定存储单元，以便在正式运行时用其他必要指令代替。另外，CPU 读取 NOP 指令、分析指令都需要时间，可以使用 NOP 指令达到让程序延时的功能。

(2) 停机指令：HLT。

指令功能：使 CPU 暂停工作，等待一次外部硬件中断的到来，让 CPU 退出暂停状态，可继续执行后面的程序指令。

(3) 等待指令：WAIT。

指令功能：使 CPU 处于等待状态，直到 CPU 芯片的$\overline{\text{TEST}}$引脚信号有效。

5.2.2　数据传送类指令

数据传送类指令是汇编语言程序设计中最常用的指令。可以在寄存器和寄存器之间、寄存器和存储器之间、AL 或 AX 寄存器和端口之间进行数据传送。数据传送类指令只改变目的操作数的值，除了标志寄存器传送指令，其他传送类指令均不影响标志位。

1. 传送指令 MOV

指令格式：MOV　目的操作数，源操作数

指令功能：实现立即数到寄存器或主存、寄存器与主存之间、寄存器与段寄存器之间、主存与段寄存器之间的传送。使用 MOV 指令时要注意以下两点。

- 不能向 CS、IP 传送数据。
- 立即数不能送段寄存器。

【例 5-8】 判断表 5.1 中指令的正误。如果指令有错，给出错误原因。

解：如表 5.1 所示。

表 5.1　指令正误判断

指　　令	正误判断	错 误 原 因
MOV AX,BH	错误	操作数类型不一致
MOV AL,[BX]	正确	
MOV AX,[DX]	错误	寄存器间接寻址不能使用 DX 寄存器
MOV IP,AX	错误	IP 不能作为目的操作数
MOV DS,0200H	错误	立即数不能送段寄存器
MOV DX,0200H	正确	
MOV [2000H],5	错误	类型不明确
MOV [2000H],[2003H]	错误	两个操作数都是存储单元
MOV DS,AX	正确	
MOV [BX],5	错误	类型不明确

2. 数据交换指令 XCHG

指令格式：XCHG　操作数 1，操作数 2

指令功能：实现寄存器和寄存器、寄存器和存储单元之间的数据交换。不能对段寄存器进行操作。

【例 5-9】 已知 AX＝1234H，BX＝0000H，数据段单元(0000H)＝11H，(0001H)＝22H。指令 XCHG AX,[BX]的执行结果是什么？

解：

```
XCHG AX,[BX]      ;AX=2211H,数据段单元(0000H)=34H,(0001H)=12H
```

指令 XCHG AX,[BX]将 AX 中的数据和用 BX 间接寻址所表示的存储单元中的数据互换。源地址码[BX]的有效地址是 0000H,到数据段中的(0000H)单元访问字数据 2211H,与 AX 互换。指令执行结果：AX＝2211H,数据段单元(0000H)＝34H,(0001H)＝12H。

3. 换码指令 XLAT

指令格式：XLAT

指令功能：用 BX＋AL 的和作为数据段中存储单元的偏移地址,从该单元取一字节数据传送到 AL 中。XLAT 指令常用于查表,BX 为表的首地址,AL 为要查的数据在表中的序号位置,查得的数据存放在 AL 中。该指令不能单独执行,执行前要准备好 BX 和 AL 的值。

【例 5-10】 在内存数据段的 2000H 单元开始,存放着 a～z 的字母表。查找表中第 5 个字母,放到 AL 中。

解：

```
MOV   BX,2000H      ;BX 为字母表首地址。BX=2000H
MOV   AL,5          ;AL 为要查找的数据在表中的序号。AL=05H
XLAT                ;将表中 2005H 单元的字节数取出送 AL
                    ;AL=45H,为字符 f 的 ASCII 码
```

4. 入栈指令 PUSH

指令格式：PUSH　源操作数

指令功能：将源操作数传送到堆栈的栈顶。栈顶的位置由 SS、SP 指定。指令执行时,堆栈指针 SP 减 2,再将数据入栈。源操作数必须是字类型。SP 的值会自动减 2。

【例 5-11】 已知 AX＝1234H,SS＝0AF9H,SP＝0FFEEH,PUSH AX 指令的执行结果是什么?

解：

```
PUSH AX      ;SP=0FFECH,(0AF9:FFEDH)=12H,(0AF9:FFECH)=34H
```

当前堆栈段栈顶地址为 0AF9:FFEEH 单元。执行 PUSH AX 指令,先将 SP 的值减 2,为 0FFECH,再将 AX 的字数据存入栈顶的两字节中。指令执行结果：SP＝0FFECH,(0AF9:FFEDH)＝12H,(0AF9:FFECH)＝34H。PUSH AX 指令执行前后堆栈的变化如图 5.1 所示。

5. 出栈指令 POP

指令格式：POP　目的操作数

指令功能：将堆栈的栈顶数据传送给目的操作数,再将堆栈指针 SP 加 2。源操作数必须是字类型。SP 的值会自动加 2。

【例 5-12】 已知 AX＝1234H,CX＝5678H,SS＝0AF9H,SP＝0FFEEH,执行下面指令

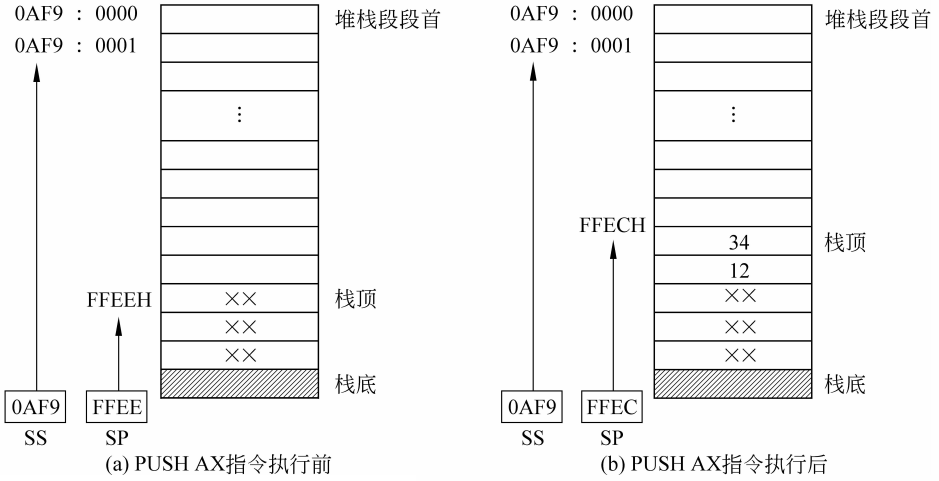

图 5.1　例 5-11 堆栈变化

的结果是什么？

```
PUSH   AX
POP    CX
```

解：

```
PUSH   AX      ;SP=0FFECH,(0AF9:FFEDH)=12H,(0AF9:FFECH)=34H
POP    CX      ;CX=1234H,SP=0FFEEH
```

当前堆栈段栈顶地址为 0AF9:FFEEH 单元。执行 PUSH AX 指令，先将 SP 的值减 2，为 0FFECH，再将 AX 的字数据存入栈顶的两字节中。指令执行结果：SP＝0FFECH，(0AF9:FFEDH)＝12H，(0AF9:FFECH)＝34H。执行 POP CX 指令，将栈顶数据 1234H 传送到 CX 中，CX＝1234H。然后 SP 加 2，为 0FFEEH。

6. 有效地址传送指令 LEA

指令格式：LEA　目的寄存器，存储单元

指令功能：取得存储单元的偏移地址传送到目的寄存器。源操作数必须是存储单元。

【例 5-13】 已知数据段中单元(1234H)＝11H，(1235H)＝22H。下面指令的执行结果是什么？

```
MOV  AX,1234H
MOV  BX,[1234H]
LEA  CX,[1234H]
```

解：

```
MOV  AX,1234H          ;AX=1234H
MOV  BX,[1234H]        ;BX=2211H
```

```
LEA  CX,[1234H]          ;CX=1234H
```

指令 MOV AX,1234H 是将立即数 1234H 传送到 AX 寄存器。指令执行结果：AX＝1234H。指令 MOV BX,[1234H]是将数据段 1234H 单元的字数据传送到 BX 寄存器。指令执行结果：BX＝2211H。指令 LEA CX,[1234H]是将数据段 1234H 单元的偏移地址传送到 CX 寄存器。指令执行结果：CX＝1234H。

7. 取逻辑地址指令 LDS/LES

指令格式：LDS　目的寄存器,存储单元

　　LES　目的寄存器,存储单元

指令功能：对于存储单元内的 4 字节,将低 2 字节的内容传送到目的寄存器,将高 2 字节的内容传送到 DS(LDS 指令)或 ES(LES 指令)。

【例 5-14】 已知 DS＝0100H,AX＝1234H,BX＝2000H,数据段单元(2000H)＝11H,(2001H)＝22H,(2002H)＝33H,(2003H)＝44H。指令 LDS AX,[BX]的执行结果是什么?

解：

```
LDS  AX,[BX]             ;AX=2211H,DS=4433H
```

指令 LDS AX,[BX]由源地址码访问数据段(2000H)单元,将低 2 字节的数据 2211H 传送到 AX 寄存器,将高 2 字节的数据 4433H 传送到 DS 段寄存器。指令执行结果：AX＝2211H,DS＝4433H。

8. 标志寄存器传送指令

标志寄存器传送指令有 LAHF、SAHF、PUSHF、POPF。

LAHF 指令的功能：将标志寄存器的低 8 位传送到 AH 寄存器中。

SAHF 指令的功能：将 AH 寄存器的值传送到标志寄存器的低 8 位。

PUSHF 指令的功能：将标志寄存器入栈。

POPF 指令的功能：将栈顶数据字传送到标志寄存器。

9. 输入输出指令 IN/OUT

8086 微处理器与 I/O 端口进行数据交换时,需要用专用的输入输出(IN/OUT)指令。IN 指令的功能是将 I/O 端口中的数据输入到微处理器的累加器 AX 或 AL 中,OUT 指令的功能是将微处理器的累加器 AX 或 AL 中的数据输出到 I/O 端口中。

IN/OUT 指令的寻址方式有直接寻址和间接寻址两种。当端口地址≤0FFH 时,采用直接寻址方式,即在指令中直接写端口地址。当端口地址＞0FFH 时,采用间接寻址方式。在 DX 中存放 I/O 端口地址,指令中用 DX 或(DX)寻址端口。端口地址≤0FFH 时也可以采用间接寻址方式。在 IN/OUT 指令中,只能使用 AL 或 AX 与端口交换数据。选择 AL 还是 AX,取决于端口内数据的位数和数据总线的宽度。表 5.2 给出了 IN/OUT 指令格式的几种情况。