

微处理器与 微计算机

(MCS—8086用户手册)

第五机械工业部第二〇七研究所

目 录

第一章 引 论	(1)
第二章 功能说明	(4)
2.1 8086简介	(4)
2.2 8086的内部结构	(5)
2.2.1 执行部件(EU)	(5)
2.2.2 总线接口部件(BIU)	(6)
2.2.3 寄存器	(6)
2.2.4 指令系统的结构	(9)
2.2.5 算术逻辑部件(ALU)(图 2—9)	(19)
8086指令系统简表	(20)
2.2.6 存储器的结构	(29)
2.2.7 输入/输出结构	(29)
2.3 8086如何工作?	(29)
2.3.1 存储器的寻址方法	(29)
2.3.2 总线多路转换方法	(30)
2.3.3 总线周期定时(图 2—11)	(31)
2.3.4 锁定	(33)
2.3.5 总清(复位)	(34)
2.3.6 暂停(HALT)	(34)
2.3.7 中断	(35)

第三章 系统操作和接口	(37)
3.1 系统模式结构	(37)
3.2 系统核心	(38)
3.3 系统核心接口	(40)
3.3.1 时钟发生器和锁存器	(40)
3.3.2 收发器	(40)
3.4 I/O 接口	(42)
3.4.1 存贮器变换 I/O	(42)
3.4.2 存贮器变换 I/O 接口	(43)
3.4.3 I/O 变换 I/O 接口	(44)
第四章 指令系统	(47)
4.1 什么是指令系统	(47)
4.2 符号、缩写和记忆符	(47)
4.3 指令和数据格式	(50)
4.4 指令系统的细节说明	(50)
4.4.1 寻址方式	(51)
4.4.2 指令系统定时	(54)
4.4.3 数据传送	(5)
4.4.4 算术运算	()
4.4.5 逻辑操作	()
4.4.6 串操作	()
4.4.7 控制转移	()
4.4.8 处理器控制	()

指令表	(112)
第五章 器件说明	(120)
8086—16位HMOS微处理器	(120)
8282/8283 8 位锁存器	(144)
8284用于8086CPU的时钟发生器和驱动器	(149)
8286/8287 8 位的并行双向总线驱动器	(157)
8288用于8086CPU的总线控制器	(162)
8259A 可编程序中断控制器	(171)

第一章 引 论

Intel 8086是一种新型微计算机,它将中级的8080系列扩展到16位领域。8086芯片具有8位和16位两种处理器的功能特性。可以执行全部8080A/8085 8位的指令系统,另外还增加了一个新的强有力的16位指令系统,这就使熟悉现有8080器件的设计人员,在基本上使用同样的8080软件设备及开发工具时,能将性能提高10倍。

8086结构设计的目标是均衡地、全面地扩展现有8080的特点,并增加了8080中没有的处理能力。新增加的特性包括16位算术运算,带符号的8位和16位算术运算(包括乘法和除法),有效的可中断的字节串操作,以及改善了位的操作。特别是其中还包括了小型计算机的操作机构,比如重入代码、位置无关代码及动态浮动程序这样的操作。另外,处理器可以直接寻址到1兆字节的存储器,并设计成能够支持多处理器构成的系统。

这是如何实现的

8086是通过将先进工艺和结构设计结合起来来改善性能的。8086是采用最近发展起来的硅栅H—MOS工艺制造的第一台微型计算机,这种工艺在器件中做成4微米的按比例缩小的MOS晶体管并在片上偏置,从而使操作速度更快、更可靠。

采用这种高性能MOS工艺,典型的片上门传输延迟时间为2毫微秒,这样快的速度和高成本的肖特基TTL电路相当,这就得到了非常快的内部时钟速率:5 MHz (200 ns),比目前所用的任何一种单片中央处理器都快。因为用四个CPU时钟周期对应大约一个存储器周期,所以8086在存储器存取方面也是特别有效的。的确,为8086选择存储器芯片,要求其周期时间在500ns到800ns,而存取数据(地址到数据输入有效)则在295ns到460ns之间。

H-MOS工艺又产生了高密度电路。整个16位数据和微程序控制结构,使用了29000个晶体管,集成在大约225平方密尔的芯片上。许多采用大规模集成电路而复杂性不很高的外部芯片,其面积更大一些。芯片这样小就意味着随着生产经验的增长,成本就会降低,正如8080、8080A,和8085出现的情况一样。

一种功能增强了的结构

8086结构上的增强是由于有一个强有力的寄存器结构,增强了的存储器寻址能力,几乎无限制的中断级,以及强有力的输入/输出接口电路。

与8080/8085的CPU不同,8086的寄存器可以处理16位数据,也可以处理8位数据。有一个通用寄存器组为16位算术/逻辑指令提供操作数。这个通用寄存器组含有四个16

位通用数据寄存器（也可以作为八个8位寄存器进行存取）、两个16位存贮器基址指示寄存器，以及两个16位变址寄存器。所有数据操作指令适用于所有的寄存器，而某些寻址方式却指定使用某些特定的寄存器。总起来讲，通用数据寄存器比8位CPU芯片上的寄存器要多一倍，从而获得了复杂的运算能力、非常灵活的存贮器寻址方式、以及很高的计算吞吐量。

第二组寄存器指定为段寄存器组，它扩充了芯片的寻址能力。8086可以寻址1兆字节的存贮器，对比之下8080/8085寻址能力则是65,536字节。在本组寄存器中，可以有四个64k字节的程序或数据段的地址位置值存放在四个16位段寄存器中。芯片还可以控制一个64k字节的地址空间作为输入/输出口。

指令系统

8086指令系统能够以几种不同的方式寻址操作数。一般来说，存贮器中的操作数，可以用一个16位的偏移地址直接寻址，或者用基址和/或变址寄存器加上一个任意的8位或16位偏移常数，来进行间接寻址。

采用一种双操作数指令格式的方法，可以把存贮器或某一寄存器用作一个操作数，而把一个寄存器或指令中的一个常数用作另一个操作数。在这些情况中，两个操作数的操作结果，可以存入源操作数中的任何一个，但是，串接常数（立即数）指令除外。另一方面，单操作数操作适用于任何一个寄存器或存贮器操作数，同样立即数指令除外。

在这种指令格式中，8086提供了四种基本算术操作（加、减、乘和除）的各种变化。提供了8位和16位算术操作和带符号的及不带符号的算术操作，带符号的数值采用标准的2的补码表示。因此，加法和减法可以是带符号的和不带符号的两种，用标志置位来区分带符号的和不带符号的操作。

借助于校正操作，8086还可以直接进行十进制数字表示的非压缩的ASCII代码，或者压缩的十进制代码的运算。标准逻辑操作，移位及传送数据等等，都适用于8位和16位的操作数。另外一些指令可以支持叫做指示器的32位地址目标的传送，这种指示器由一个16位偏移地址和一个16位段地址组成。

还提供了一些单字节指令，除去执行控制字节和字串的各种基本操作之外，如果加上一个特殊的前缀，其中每条指令还能重复执行。单操作形式可以组合起来形成复杂的串操作，由专门的迭代操作来控制其重复。结果是为执行复杂的串功能产生了严密的、有效的循环。

为了处理程序流程，提供了两种基本的调用、转移和返回操作。一种是在现行代码段中转移控制，另一种是将控制转移到任意代码段，然后将该代码段变为现行代码段。8086支持直接和间接转移，两种转移都使用标准的寻址方式。段内调用和转移指定一个自相关的偏移量，因此允许位置无关代码。

总共提供了16种条件转移。可以对带符号和不带符号的特征以及奇偶、溢出、零和符号条件进行测试。

一些简单系统

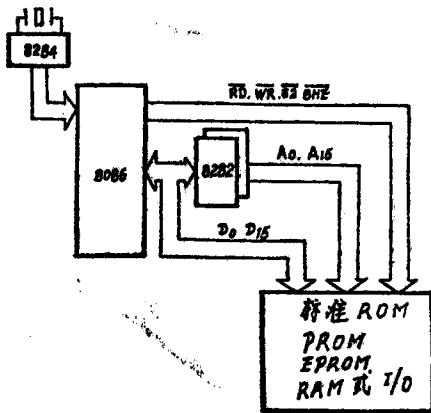


图1 系统构成

(a) 工作于最小模式时，8086只需要11块元件就可以构成一个完整的系统，其中包括时钟，2 K字节的RAM和4 K字节的ROM

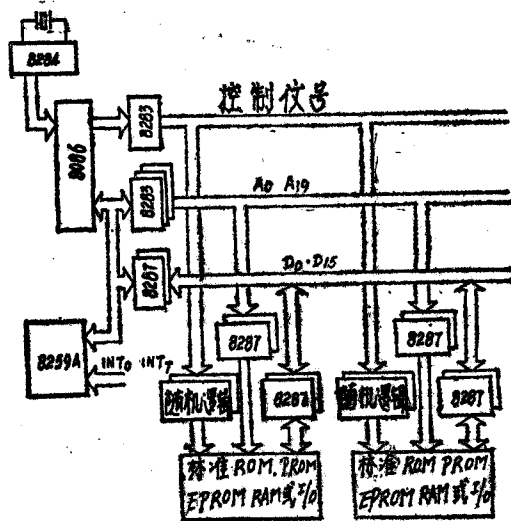
额外的直接存贮器存取控制和用于多处理器操作的总线锁定能力。

在缓冲系统中，对于5 MHz的CPU操作，要求存贮器和I/O存取时间从接收地址开始为395ns，从接收读命令开始为290ns。

所有系统设计的基本准则是它的性能。初步研究表明，以8086为基础的系统的性能，平均来说要比以8080为基础的系统提高一个数量级。根据不同的程序类型，可以期望，8086的执行速度要比8080A的速度快7到12倍，同时，程序一般要短10%到25%。然而，有时对于每条8080指令存在着变换为8086指令的问题，虽然8080程序能够很容易的传送到8086，但为了使8086发挥最大的效能还是要求对某些程序进行重写。

在图1(a)所示的一种最小模式系统中，8086产生一些控制信号，供存贮器和外围器件同地址和数据总线相互作用而使用，同时产生一个定时信号来锁存地址。如前所述，在这种结构中，对于5 MHz的8086，要求存贮器和I/O器件的存取时间，从接收地址开始大约为430ns，从接收读或写允许信号开始大约为205ns。

在大型缓冲结构中（图1(b)），处于最大模式的8086，只在产生最小模式控制信号所必需的几根引线端产生编码的状态信息。这里所必需的全部编码信息要足以推动8288总线控制器产生多总线相容的控制信号，以及地址锁存器和数据收发器所需的定时信号。这时，最小模式的控制信号引线端可以承担另外的功能，比如



(b) 加上一些支持元件，就可以构成非常大的系统

第二章 功 能 说 明

2.1 8086 简介

8086是一个完整的微处理器，它适用于具有各种不同复杂程度的通用计算机系统。最简单的以8086为基础的系统，在复杂性方面可同以8080为基础的系统相比较，但在速度上要快好几倍。最复杂的情况是8086构成的多处理器系统，其中每一个处理器都可以存取一兆字节的存贮器。

8086的存贮器系统以8位字节构成，但存贮器以16位的字进行传送，处理起来如同以字节传送一样容易。指令系统中提供了位、字节、字和块（串）操作。8086可以执行带符号的算术运算及可中断的串操作，并且可以采用浮动子程序，重入编程及多重处理。8086要求单一的5伏电源，采用标准的40条引线的双列直插式封装（DIP）。

8086采用标准封装而能实现这样多的功能是通过两种方法来实现的。首先，同存贮器和外部设备进行外部通讯通过一个20位定时多路转换的地址和数据总线来实现（节2.3.2中详细介绍）。其次，采用内部结构上的转换以使处理器适应用户要求的不同复杂程度的系统。在最简单的系统中，8086自己产生其总线控制信号；而在比较复杂的系统中，总线控制则由一个8288总线控制器来承担，8086的8根外引线需要进行转换以执行所要求的相应的功能。8086的第33条引线（MN/MX），根据使用8288与否，可以将其固定地连至电路板的地线或+5V电源线来实现转换。

8086 系列器件概括如下：

8086 CPU

8284 时钟驱动器

8282 8线锁存器

8283 反相8线锁存器

8286 8线3态收发器

8287 反相8线3态收发器

8288 总线控制器

8259A 中断控制器

MCS—86系统的总线结构同MCS—80和MCS—85外围器件兼容。这就使得用户可以利用现有器件和硬件设计。现存的8080系统软件也适用于MCS—86系统。如果一个8080程序与指令执行时间、指令长短、存贮器地址或特定代码（比如用作屏蔽的代码）无关，则这个程序就可以在汇编语言一级进行机器翻译，并且可以无需修改地汇编为8086代码，这是因为8080的寄存器组为8086寄存器组的一个子集（见图2—5）。为MCS—80编制的与时间和空间有关的子程序，必须用不同的方法加以重新设计。在某些情

况下，由于争用内部总线，可能使得8086执行一条指令的时间因指令的位置不同而不同，但某一给定的序列是可重复的。另外，8086中向量中断是以不同方法处理的，这就要求修改8080的中断软件。

2.2 8086的内部结构

8086微处理器的内部结构分为如下两个主要功能区：

- 执行/控制部件 (EU)
- 总线接口部件 (BIU)

图 2—1 示出了这两个部件之间的相互关系，并在每个部件之内又进一步划分了各种处理功能。它们可以直接相互作用，但大部分情况是执行各自独立的功能。

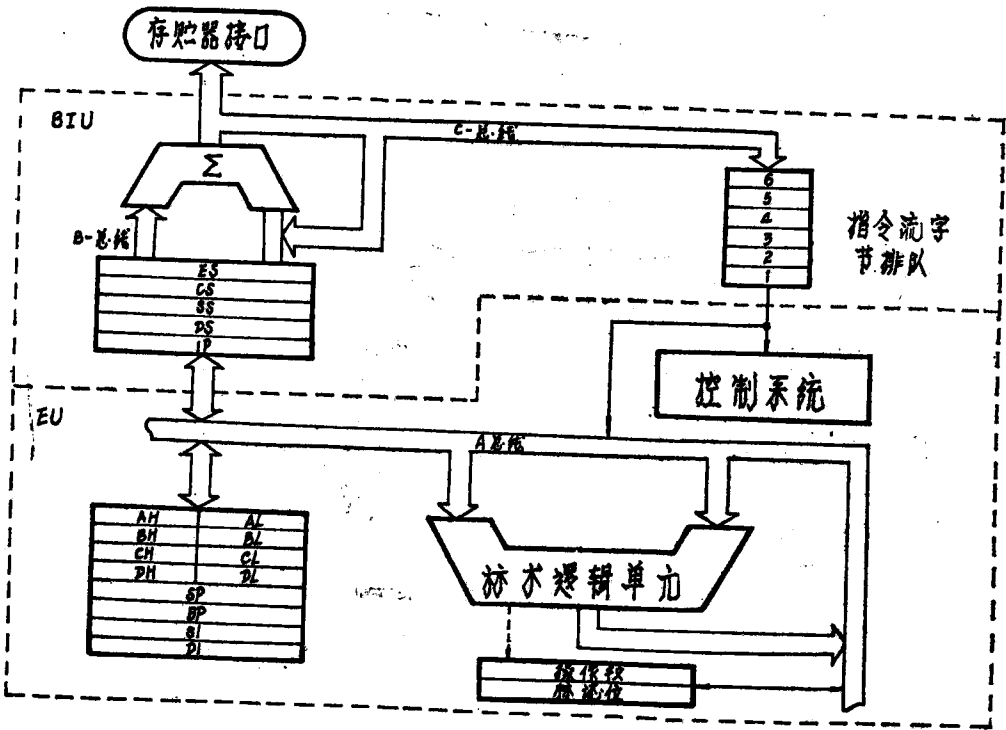


图 2—1 8086功能框图

2.2.1 执行部件(EU)：

EU含有数据寄存器和算术—逻辑部件（ALU），因此可以执行基本的处理功能。EU从BIU中接收预取的指令，并将没有进行浮动处理的操作数地址转回BIU（参见节

2.3.1 关于存储器寻址的讨论)。然后,EU经BIU接收存储器操作数,加以处理,并将结果送到BIU以便存储起来。

2.2.2 总线接口部件 (BIU):

BIU的目的是尽量加大总线带宽的利用,因为这是限制处理器速度的主要因素。为此采用了两种方法。首先,在EU需要指令之前进行指令预取,将其排队缓冲。排队缓冲器最多含有六个字节的指令流等待译码和执行。因此EU不需要等待为取一条新指令所要求的一个总线周期。其次,BIU提供的涉及操作数的取和存、地址浮动处理及总线控制等功能,全都是同EU处理并行进行的。

2.2.3 寄存器:

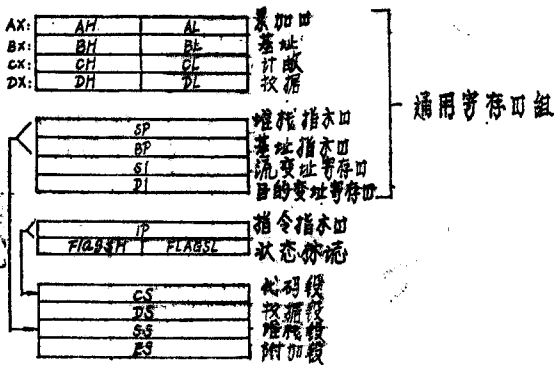


图 2-2 8086寄存器格式

8086处理器含有三组四个16位的寄存器和一组九个一位的标志寄存器。三组16位寄存器是通用寄存器组、指示寄存器和变址寄存器组,以及段寄存器组。有一个16位的指令指示器,它不能直接进行存取,但可用控制转移指令来加以控制(见图2.2)。

(AX,BX,CX,DX)寄存器组叫做通用寄存器组或HL组(见图2-3)。通用寄存器可以不加限制地参与8086的算术和

逻辑操作。某些其它的8086操作(如串操作)指定某一通用寄存器作特殊用途。其用途可用下面的助记符来表示:

- AX: 累加器
- BX: 基址
- CX: 计数
- DX: 数据

通用寄存器有一个和其它寄存器不同的性质,即是其高、低两半可以分别寻址,这样就可把通用寄存器看作是两组各四个8位的寄存器,分别称之为H和L。

累加器可从另外的意义上加以区分,

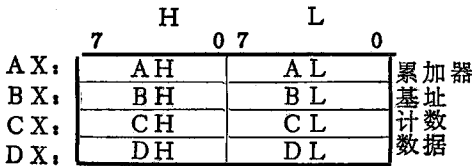


图 2-3 通用寄存器

即采用累加器作为数据传送、算术和逻辑指令的目的，比使用通用寄存器使得编程更为紧凑（详见节 2.2.4）。8086 处理器中其余的寄存器是不可分的，必须将其看作含有一个 16 位字加以存取，而不管是否使用它的高位字节和低位字节。

(SP、BP、SI、DI) 寄存器组称为指示器和变址寄存器组（或称 P 和 I 组，见图 2-4）。本组里的寄存器，其相同点是一般含有在一个段内寻址所用的偏移地址。像通用寄存器一样，指示器和变址寄存器可以参与 8086 的 16 位算术和逻辑操作。

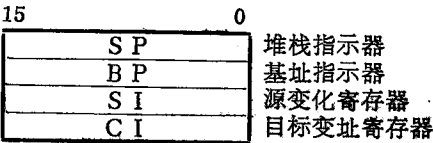


图 2-4 指示器和变址寄存器

本组的寄存器还有一个相同点，就是可以用于地址计算。然而，也有一些不同的地方，因此可以将其分为两组，即 P 组或指示器组 (SP、BP)，以及 I 组或称变址寄存器组 (SI、DI)。其差别在于指示器都假定含有现行堆栈段内的偏移地址，而变址器则都假定含有现行数据段内的偏移地址（串操作除外）。涉及这些寄存器的助记符为：

- SP: 堆栈指示器
- BP: 基址指示器
- SI: 源变址寄存器
- DI: 目的变址寄存器

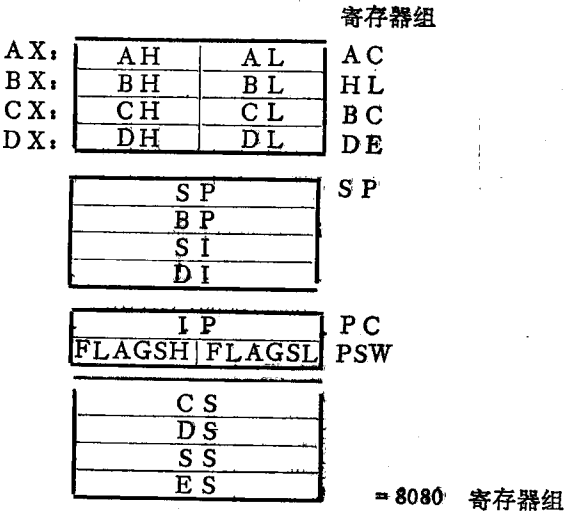


图 2-5 8080 寄存器为 8086 寄存器的一个子集

某些 8086 寄存器的功能类似于 8080 系列处理器中寄存器的功能（见图 2-5）。8086 中的 BH 和 BL 寄存器对应于 8080 中的 H 和 L 寄存器。CH 和 CL 寄存器对应于 B 和 C，而 DH 和 DL 则对应于 D 和 E。8080 中的 SP 和 PC 可以直接转换为 8086 中的 SP 和 IP，但是，它们的标志寄存器稍有不同。

(AF,CF,DF,IF,OF,PF,SF,TF,ZF) 寄存器组叫做标志寄存器或F组。本组中的标志都是一位，用以记录处理器的状态信息并控制处理器的操作。标志寄存器的助记符为：

AF: 辅助进位	PF: 奇偶
CF: 进位	SF: 符号
DF: 方向	TF: 陷井
IF: 中断允许	ZF: 零
OF: 溢出	

AF,CF,PF,SF和ZF标志相当于8080的标志，一般反映的是最后一次算术或逻辑操作的状态。如果指令操作的结果为零，则ZF标志置位。如果结果的最高有效位出现一个1，则SF标志置位。PF标志置位为1表示奇偶性为偶数。最高位上有一个进位或借位输出，则置位CF标志，而位3到位4有一进位输出，或者从位4到位3有一个借位，则置位AF标志。OF标志加入这一组，反映的是带符号运算的溢出条件。DF,IF和TF标志用来对处理器进行某些控制。DF标志控制串操作指令的方向（自动增量或自动减量）。IF标志用来允许或封锁外部中断。TF标志使处理器进入单步方式以便于调试程序。

图2—6示出了标志寄存器的格式，标志可以通过推入标志操作来存入。图2—7示出了8080标志同8086标志的对应关系。注意，在MCS—80系列中，推入堆栈或从堆栈弹出的PSW不仅包含有标志，而且还包含累加器的内容。8086标志的进一步讨论及其功能是第四章中的内容。

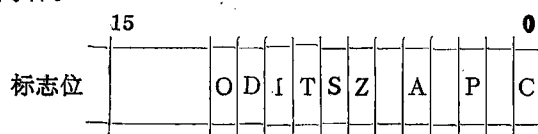


图 2—6 标志寄存器

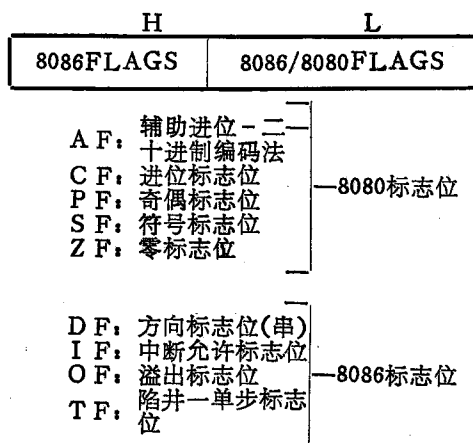


图 2—7 8080/8086标志寄存器对应关系

(CS,DS,SS,ES) 寄存器组称为段寄存器组或S组。由于段寄存器用于所有的存储器地址计算,所以在这方面起着重要作用(见节2.3.1)。段寄存器的助记符为:

CS: 代码段

DS: 数据段

SS: 堆栈段

ES: 附加段

CS寄存器的内容确定了现行代码段。所有的取指令都是相对于CS,并使用指令指示器(IP)作为一个偏移量。

DS寄存器的内容确定了现行数据段。除去涉及到BP或SP,或者一条串指令中的DI的数据外,所有数据参量都是相对于DS。可以强使数据参量相对于另外三个段寄存器中的一个,这要通过具有一个字节的段超越前缀的先行指令来实现。

SS寄存器的内容确定了现行堆栈段。明显或不明显涉及到SP或BP的所有数据参量都是相对于SS的,这包括所有推入和弹出操作,其中也有那些由调用、中断和返回操作所引起的推入和弹出操作。涉及BP(而不是SP)的数据存取可以强使其相对于另外三个段寄存器之一,这要通过采用专门的一字节基址前缀来实现(见第四章)。

ES寄存器的内容确定了现行附加段。附加段通常作为一个附加段数据来对待。使用DI的串指令中的数据存取是相对于ES的。

未装入或打入段寄存器的程序称为动态浮动程序。可以将这样一个程序断开,移入存储器中的一个新的位置,并根据新的段寄存器的值重新起动。段寄存器如图2—8所示。

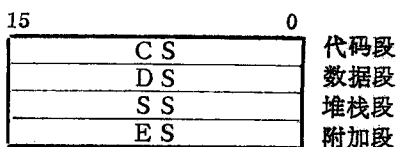


图 2—8 段寄存器组

2.2.4 指令系统的结构:

这里介绍的指令按功能分为六组:

- 数据传送
- 算术运算
- 逻辑操作
- 串操作
- 控制转移
- 处理器控制

上面所说的前三组中的每一组又可进一步细分为一个代码组合，这个组合指明指令是依据立即数、寄存器，还是依据存储器单元而作用；处理的是16位的字，还是8位的字节；以及使用的是那种寻址方式。所有这些代码，在本书中都详细列出并加以说明，但在编写汇编语言程序的时候，用户不必对每一条进行详细编码，其程序文本会自动地使汇编程序产生正确的代码。所说的三个功能组中的每一组里，都有三类通用指令：

- 寄存器或存储器到/由寄存器
- 立即数到寄存器或存储器
- 累加器到/由寄存器、存储器或I/O口。

汇编语言编程手册介绍了8086指令系统的句法。

数据传送：

数据传送操作分为四类：

- 通用
- 累加器专用
- 地址目标
- 标志

除SAHF和POPF外，不影响标志置位。

通用传送：

提供了四种通用数据传送操作，它们可以用于大多数操作数，尽管也有一些特殊的例外情况。通用传送（除XCHG外）是允许一个段寄存器作为一个操作数的唯一操作。

——MOV执行从源操作数到目的操作数的一个字节或一个字的传送。

——PUSH SP寄存器减2，然后，将一个字从源操作数传送到由现行SP寻址的堆栈单元。

——POP 将一个字操作数从由SP寄存器寻址的堆栈单元传送到目的操作数，然后SP寄存器加2。

——XCHG 将一个字节或一个字的源操作数同目的操作数进行交换。段寄存器不能作为XCHG的操作数。

累加器专用传送：

提供了三种累加器专用传送操作：

——IN（或INW）将一个字节（或一个字）从一个输入口传送到AL寄存器（对于INW为AX寄存器）。输入口这样来指定，即用串接在指令中的一个数据字节，作为口的地址，可以存取固定口0~255，或者用DX寄存器中的内容作为口的地址，可以存

取可变口到64K。

——OUT (或OUTW) 与IN (或INW) 类似, 不同的是从累加器传送到输出口。

——XLAT 执行一个查表字节变换。AL寄存器用作被BX寄存器寻址的一个256字节表的变址器。将这样选择的字节操作数传送到AL。

地址目标传送:

提供了三种地址目标传送:

——LEA (装入有效地址) 将源操作数的偏移地址传送到目的操作数。源操作数必须是一个存储器操作数, 而目的操作数则应该是一个16位通用寄存器、指示器或变址寄存器。

——LDS (将指示器装入DS) 将一个“指示器目标” (即包含一个偏移地址和一个段地址的32位目标) 从源操作数 (必须是一个存储器操作数) 传送到一对目的寄存器。段地址被传送到DS段寄存器, 而偏移地址则必须传送到16位的通用寄存器、指示器或变址寄存器。

——LES (将指示器装入ES) 与LDS类似, 不同的只是将段地址传送到ES段寄存器。

标志寄存器传送:

提供了四种标志寄存器传送操作:

——LAHF (将标志装入AH) 将标志寄存器SF, ZF, AF, PF和CF (8080标志) 传送到AH寄存器指定的位。

——SAHF (将AH存入标志寄存器) 将AH寄存器的指定位传送到标志寄存器SF、ZF、AF、PF和CF。

——PUSHF (推入标志) SP寄存器减2, 并将所有标志寄存器传送到由SP寻址的堆栈单元中指定的位。

——POPF (弹出标志) 将SP寄存器寻址的堆栈单元中指定的位传送到标志寄存器, 然后SP加2。

算术运算:

8086提供了具有许多不同变化的四种基本数学操作。提供了8位和16位操作, 也提供了带符号的和不带符号的算术运算。带符号的数值采用标准2的补码表示。加法和减法操作可以是带符号的和不带符号的操作。在这种情况下, 可以从标志寄存器的置位来区分所做的是带符号的或是不带符号的操作 (见条件转移)。提供了校正操作, 这就允许按照非压缩的十进制数字或压缩的十进制表示直接进行算术运算。

标志寄存器置位:

六个标志寄存器可根据算术运算的结果来置位或者清除,以反映操作结果的某些性质。一般遵从下列规则:

- CF: 如果操作结果在最高位有一个进位输出(加法),或者有一个借位输入(减法),则CF置位;否则,清除CF。
- AF: 如果操作结果在低四位有一个进位输出(加法),或有一个借位输入(减法),则AF置位;否则,清除AF。
- ZF: 如果操作结果为零,则ZF置位;否则,清除ZF。
- SF: 如果操作结果的最高位为1,则SF置位;否则,清除SF。
- PF: 如果操作结果的低八位以2为模的和是0(偶数奇偶),则PF置位;否则,清除PF(奇数奇偶)。
- OF: 如果操作结果在最高位有一个进位输入,而最高位没有进位输出,或者反之,则OF置位;否则,清除OF。

加法:

提供了五种加法操作:

- ADD 执行两个源操作数的加法,并将结果转回到其中的一个操作数。
- ADC (带进位的加法) 执行两个源操作数的加法,若预先发现CF标志置位再加1,并将结果转回到其中的一个操作数。
- INC (加1) 执行源操作数加1,并将结果转回到操作数。
- AAA (非压缩的BCD(ASCII)加法调整) 两个非压缩的十进制操作数相加时结果放在AL,AAA执行一种校正,得到一个非压缩的十进制和。
- DAA (十进制加法调整) 两个压缩的十进制操作数相加的结果放在AL,DAA执行一种校正,得到一个压缩的十进制和。

减法:

提供了七种减法操作:

- SUB 执行两个源操作数的减法,并将结果转回到其中的一个操作数。
- SBB (带借位的减法) 执行两个源操作数的减法,若预先发现CF标志置位再减1,并将结果转回到其中的一个操作数。
- DEC (减1) 执行从源操作数中减1操作,并将结果转回到源操作数。
- NEG (取补) 执行零减源操作数,并将结果转回到操作数。
- CMP (比较) 执行两个源操作数的减法,影响标志,但不转回结果。
- AAS (非压缩的BCD(ASCII)减法调整) 两个非压缩的十进制操作数相减时结果放在AL,AAS执行一种校正,得到一个非压缩的十进制差。
- DAS (十进制减法调整) 两个压缩的十进制操作数相减时结果放在AL,DAS

执行一种校正，得到一个压缩的十进制差。

乘法：

提供了三种乘法操作：

——MUL 执行累加器（AL或AX）和源操作数的不带符号的乘法，将双倍字长的结果转回到累加器及其延伸部分（对于8位操作为AL和AH，对于16位操作为AX和DX）。若结果的高半部分不为零，则CF和OF置位。

——IMUL（整乘）类似于MUL，不同的只是执行带符号的乘法。如果结果的高半部分不是结果的低半部分符号的扩展，则CF和OF置位。

——AAM（非压缩BCD(ASCII)乘法调整）两个非压缩的十进制操作数相乘时结果放在AX中，AAM执行一种校正，得到一个非压缩的十进制乘积。

除法：

提供了三种除法操作，并且为了支援带符号的除法而提供了两种符号扩展操作：

——DIV 执行一条不带符号的除法，累加器及其延伸部分（对8位操作为AL和AH，对16位操作为AX和DX）被源操作数来除，将单倍长度的商转回到累加器（AL或AX），将单倍长度的余数转回到累加器延伸部分（AH或DX）。标志不定。被零除则产生0型中断。

——IDIV（整除）类似于DIV，不同的只是执行一条带符号的除法。

——AAD（非压缩BCD(ASCII)除法调整）执行两个非压缩的十进制操作数相除之前AL中被除数的一种校正，因此结果将得到一个非压缩的十进制商。

——CBW（将字节转换为字）执行AL寄存器到AH寄存器的符号扩展。

——CWD（字转换为双倍字）执行AX寄存器到DX寄存器的符号扩展。

逻辑运算：

8086为8位和16位操作数提供了基本逻辑操作。

单操作数操作：

提供了三种单操作数逻辑操作：

——NOT形成源操作数的1的补码，（即按位取反）并将结果转回到操作数。不影响标志。

——移位操作为存贮器和寄存器操作数提供四种移位操作，SHL（逻辑左移），SHR（逻辑右移），SAL（算术左移），SAR（算术右移）。可进行一位移位，也可由CL寄存器给出移位位数进行可变位数移位。CF标志是最后一次移出的一位的值，OF只对移位位数为1时是确定的，并且当最终符号位的值不同于以前的符号位的值时，OF