宫汇编语言参改手册

（续）

MCS-86描述符	
DI	目标变址寄存器（16位）
SI source	堆栈变址寄存器（16位）
CS	代码段寄存器（16位）
DS	数据段寄存器（16位）
ES	附加段寄存器（16位）
SS	堆栈段寄存器（16位）
REG8	名字或一个8位CPU寄存器地址的编码。
LSRC，RSRC	指的是指令操作数，当使用两个操作数时，通常叫左源操作数和右源操作数。最左边的操作数也叫作目标操作数，最右边的操作数也叫作源操作数。
reg	是一个指令说明中用以指定REG8或REG16的一个字段。
EA	有效地址（16位）
r/m	存取存贮器操作数时用的MODRM字节的位2，1，0。这个3位字段与mode和W字段一起定义EA。
mode	MODRM字节的位7，6。这个2位字段定义寻址方式。
W	一条指令中的一个位字段，它识别字节指令（W=0）和字指令（W=1）。
d	一条指令中的一个位字段，“d”识别方向，即：一个指定的寄存器是源操作数还是目标操作数。
(…)	括号指所包含的寄存器或存贮单元的内容。
(BX)	代表寄存器BX的内容，它意味着一个8位操作数被定位的地址。当在汇编指令中这样使用时，必须用方括号括起来。

（续）

MCS-86描述符	意　　义
((BX))	意指一个8位操作数，即由寄存器BX的内容所指的存贮单元的内容。这个记法只是叙述性的，在本章使用。它不能在源语句中出现。
(BX)+1,(BX)	意指（一个16位操作数的）地址，其低8位在由寄存器BX的内容所指的存贮单元中，高8位驻留在下一个连续存贮单元(BX)+1中。
((BX)+1,(BX))	指在地址(BX)+1,(BX)中的16位操作数。
并置，例如：	指一个由两个8位字节并置起来的16位字。低字节
((DX)+1:(DX))	在由DX所指的存贮单元中，高字节在下一个顺序的存贮单元中。
addr	存贮器中一个字节的地址（16位）。
addr-low	一个地址的低有效字节。
addr-high	一个地址的高有效字节。
addr+1:addr	存贮器中两个连续字节的地址，由addr地址开始。
data	立即操作数（当w=0时，8位；当w=1时，16位）。
data-low	16位数据字的低有效字节。
data-high	16位数据字的高有效字节。
disp	位　移
disp-low	16位位移的低有效字节。
disp-high	16位位移的高有效字节。
←	赋　值
+	加
−	减
*	乘

（续）

MCS-86描述符	意　　义
/	除
%	模
&	与
|	或
||	异　或

5.1　指令和伪指格式

这里简要说明的格式反映由Intel公司提供的和Intellec开发系统一起使用的汇编程序ASM-86所处理的汇编语言。

每行写一条汇编语言指令。如果一个分号不是出现在一个串中，则把该行的剩余部分看作为注释。如果一行用一个和号（“&”）开始，则把该行考虑为前一行（指令或命令，而不是注释）的继续。

任何一条指令都是由一系列标记组成。每一个标记可以是三种类型中之一：

名　字

常　数

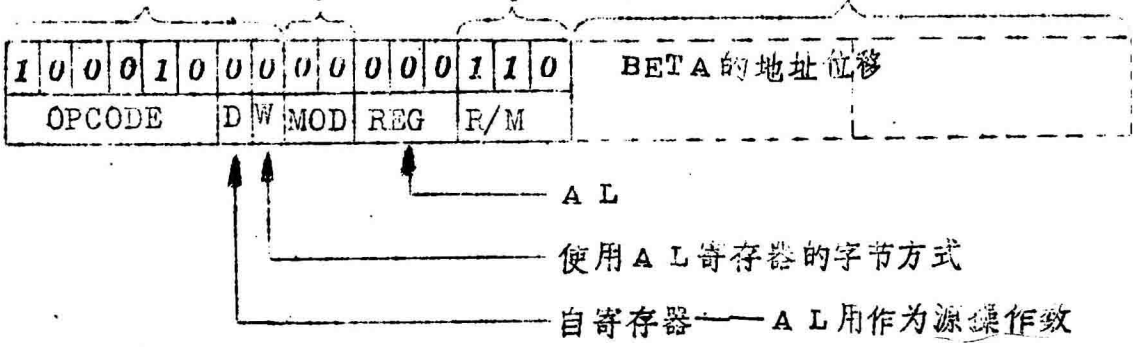
定义符

当两个连续在一起的标记可译码为某个其它标记时，必须用一个空格把它们分开；否则，空格没有意义，并且可以省略。但是，当需要时，可以插入附加的空格；计算机忽略它们。可以写任意行数长的注释，但每行必须用一个分号开始。汇编程序忽略注释和空白行。汇编程序不区分大写和小写字母。

上述规则的一个例外是字符串。汇编程序识别串内的所有字符、空格和空行。

第 272 页是本章按字母顺序排列的一个索引。在这些列表中，所有指令都涉及汇编语言记忆符。虽然每一条指令代码没有一个唯一的记忆符，但在该指令中有汇编程序识别正确代码用的足够信息，源和目标记忆符。这意味着，你不必考虑对于不同的源操作数和目标操作数需要不同MOV代码中的哪一个代码。当你写

EXAMPLE:MOV BETA,AL



时，汇编程序注意到：

D （方向位）

W （字位）

MOD （方式字段）

位移量

汇编程序选择正确方式来完成你所希望的操作。这一章描述那些代码是怎样操作的。

寻址方式

8086指令系统提供了几种不同的寻址操作数方法。大多数双操作数的指令允许存贮器或一个寄存器作为一个操作数，一个寄存器或指令内一个常数作为另一个操作数。不考虑存贮器到存贮器的操作。

存贮器中的操作数可用一个16位偏移地址直接寻址，或者用基址寄存器（BX或BP）和（或）变址寄存器（SI或DI）加上一个任

选的8位或16位位移常数来间接寻址。这个常数可以是一个纯数的名字。当使用一个名字时，该位移常数是该变量的偏移（见第一章）。

一个双操作数操作的结果可以指向存贮器或一个寄存器。单操作数操作可均匀应用到除立即常数外的任何操作数。实际上，所有8086操作可以规定8位或16位操作数。

1. 存贮器操作数。存贮器中的操作数可以用四种方法寻址：

- 16位偏移地址直接寻址
- 通过一个基址寄存器，BX或BP，任选用一个8位或16位位移量间接寻址。
- 通过一个变址寄存器，SI或DI，任选用一个8位或16位位移量间接寻址
- 通过一个基址寄存器和一个变址寄存器的和，任选用一个8位或16位位移量间接寻址

一个8086寄存器中或存贮器中的一个操作数的地址由每个指令中的三个字段来规定。这些字段是方式字段（mod）、寄存器字段（reg）和寄存器/存贮器字段（r/m）。如果使用这些字段，它们占指令序列的第二个字节。

方式字段占该字节的两个高有效位7，6，并规定了r/m字段（位2，1，0）是怎样用于操作数定位的。r/m字段可对装该操作数的寄存器命名或可规定一个指向存贮器中该操作数地址的寻址方式（在与mod字段结合时）。reg字段紧跟方式字段之后占有位5，4，3，可以规定一个操作数是一个8位寄存器或16位寄存器。在某些指令中，这个reg字段还给出了规定该指令的附加信息位，而不仅是给一个寄存器编码（见第六章和附录A）。

说明：存贮器操作数的有效地址（EA）根据mod和r/m字段来计算：

若mod=00 则DISP=0*，无disp-low 和disp-high.

若 mod=01 则 DISP=disp-low 符号扩展到16位，无disp-high

若 mod=10 则DISP=disp-high:disp-low

若 r/m=000 则EA=(BX)+(SI)+DISP

若 r/m=001 则EA=(BX)+(DI)+DISP

若 r/m=010 则EA=(BP)+(BP)+DISP

若 r/m=011 则EA=(BP)+(DI)+DISP

若 r/m=100 则EA=(SI)+DISP

若 r/m=101 则EA=(DI)+DISP

若 r/m=110 则EA=(BP)+DISP*

若 r/m=111 则EA=(BX)+DISP

*"当mod=00和r/m=110 则EA=disp-high:disp-low "除外

访问16位目的码的指令把EA译码为寻址低位字节；该字由EA+1，EA寻址。

编码：

mod reg r/m	disp-low	disp-high

2. 段超控前缀。通用寄存器BX和指示器BP可作为基址寄存器。当BX是基址寄存器时，由缺省规定的操作数驻留在当前数据段，且DS寄存器用于计算该操作数的物理地址。当BP是基址寄存器时，由缺省规定的操作数驻留在当前堆栈段，且SS段寄存器用于计算该操作数的物理地址。当基址和变址寄存器都使用时，由缺省规定的操作数在由该基址寄存器所确定的段中，即BX意味着使用DS，BP意味着使用SS。当单独使用一个变址寄存器时，由缺省规定的操作数驻留在当前数据段。其它存贮器操作数中大多数的物理地址是利用DS段寄存器按缺省来计算的(注意下面有例外情况)。这些汇编程序缺省(assembler-default)段寄存器的选择可通过在访问指令(referencing instruction)

之前加段超控前缀来超控。

说明：由一个段前缀的reg字段选择的段寄存器用于计算这个前缀后面的指令的物理地址。这个前缀可以与LOCK和(或)REP前缀结合使用，尽管后者有某些要求和后果——见REP。

编码：

001 reg 110

根据下列表给reg赋值

	段
0 0	ES
0 1	CS
1 0	SS
1 1	DS

例外：

由SP寄存器寻址的所有操作数的物理地址用SS段寄存器计算，可以不超控SS。串原语操作的目标操作数(由DI寄存器寻址)的物理地址用ES段寄存器计算，可以不超控ES。

3. 寄存器操作数：四个16位通用寄存器和四个16位指示器和变址寄存器可以交替地作为几乎所有16位操作中操作数。应注意的三个例外是乘法、除法和某些串操作，它们隐式地使用AX寄存器。HL组的八个8位寄存器可以交替地作为8位操作中的操作数。乘、除和某些串操作隐式地使用AL寄存器。

说明：寄存器操作数可以由一个识别字段(distinguished field)来指出，在这种情况下，REG将表示所选择的寄存器；或由一个编码字段(encoded field)来指出，在这种情况下，EA将表示由r/m字段所选择的寄存器。没有"W"位的指令总是访问16位寄存器(哪怕它们真要访问任何寄存器)；有"W"位的指令根据"W"访问8位或

16位寄存器。

编码：

通用寄存器：

识别字段：

reg	或	reg

对于mode = 11　EA=r/m（一个寄存器）：

11	reg

根据下面的表给reg赋值：

16位〔w=1〕		8位〔w=0〕	
000	AX	000	AL
001	CX	001	CL
010	DX	010	DL
011	BX	011	BL
100	SP	100	AH
101	BP	101	CH
110	SI	110	DH
111	DI	111	BH

访问作为16位目的码的标志寄存器文件的指令用符号FLAGS表示该文件：

FLAGS　X:X:X:X:(OF):(DF):(IF):(TF):(SF):(ZF):X:(AF):X:(PF):X:(CF)

这里，x没有定义。

4　立即操作数。除了乘、除和串操作外，所有双操作数的操作允许一个源操作数作为立即数据出现在该指令中。有8位字节（它是

低位字节的符号扩展）的16位立即操作数可缩短到8位。

立即操作数有三点应当说明：

·立即操作数总跟在指令中寻址方式位移常数之后（当出现常数时）。

·16位立即操作数的低位字节总在高位字节之前。

·有S：W＝11的指令的8位立即操作数符号扩展到16位值。

下面每种类型的指令，给出以下信息：

① 描述的英文名称或短语

② 指令的二进制编码

③ 指令所花的时间，用时钟周期表示（使用5-MHz 时钟时，一个周期是200ns；使用8-MHz 时钟时，一个周期是125ns）。

④ 分步操作说明

⑤ 在指令操作期间， 标志寄存器置位（为1）或复位（为0）的列表（见附录C）。

⑥ 一般地说明何时使用该指令，如何工作，可能使用或行使的缺省，指出要记住它与其它指令或命令之间相互影响。

⑦ 例 子

给予指令的时间取决于操作数的种类。这些时间对于寄存器到寄存器操作和立即数据到寄存器操作是固定不变的，例如：

```
MOV DX, AX  ;takes 2 cycles.
MOV DX, 444 ;takes 4 cycles, regardless of which
            ;register or what data.
```

为了计算有效地址，存贮器中的操作数花了一些额外时间。这些附加的周期在所列的时间中由项“+EA”表示。所需要的时间总数依据三个因素而变化：

a. 存贮器操作数的地址表达式中使用哪一种寻址

b. 是否需要一个段超控前缀字节。

c. 对于字操作数，该字的第一字节是驻留在偶地址上还是驻留在

奇地址上。

下面的列表表示，每一种寻址方式在取首字节在偶地址上的8位存贮器操作数或16位存贮器操作数（字）所需要的附加周期，对于驻留在奇数存贮器地址的字，加4个周期。如果用了一个段超控，则加2个周期。

	寻址方式	加
①	直接16位偏移地址	6
	例如：MOV BX, SIMPLE_NAME	
	需8 + 6即14个周期	
②	间接通过基址或变址寄存器	5
	例如：MOV CX,〔BX〕	
	MOV CX,〔SI〕	
	每个需8 + 5即13个周期	
③	间接通过基址或变址寄存器，带位移常数	9
	例如：MOV DX, SIMPLE-NAME〔BX〕	
	MOV SIMPLE-NAME〔DI〕, CX	
	每个需8 + 9即17个周期	
④	间接通过一个基址寄存器与一个变址寄存器的和	7或8
	例如：MOV DX,〔BX〕〔SI〕	
	MOV〔BX〕〔DI〕, CX	
	每个需8 + 7即15个周期	
⑤	间接通过基址与变址寄存器的和，加位移常数	11或12
	例如：MOV DX, SIMPLE-NAME〔BX〕〔SI〕	
	MOV SIMPLE-NAME〔BX〕〔DI〕, CX	
	每个需8 + 11即19个周期	

如果SIMPLE_NAME 驻留在一个奇地址上，包含该变量的上述每

一个地址表达式应需要4个附加的时钟周期。如果需要一个段超越(见第四章中的ASSUME)，则需额外加2个周期。于是，指令MOV ES:SIMPLE_NAME,CX 将需要16个周期而不是14个周期，当SIMPLE_NAME 的第一个字节在一个奇地址上时，将需要20个周期。

3 指令系统的组织

这一节按以下六个功能组来描述指令：

数据传送

算计操作

逻辑操作

串操作

控制转移

处理器控制

上面表中提到的前三组中的每一组进一步细分为一批代码，这些代码规定该指令是作用于立即数据，寄存器还是存贮器地址，是处理16位字还是8位字节，以及使用什么寻址方式。列出所有这些代码的清单。并详细地作了解释，但是你不必分别为每一个代码编码。你的程序上下文使汇编程序自动地产生正确代码。在提到的三个功能组中的每一组内，有三类通用的指令：

寄存器或存贮器空间到寄存器或者从寄存器到寄存器或存储器空间。

立即数据到寄存器或存贮器

累加器到寄存器、存贮器或口地址，或者由寄存器、存贮器或口地址到累加器。

1. 数据传送

数据传送操作分为四类：

- 通用传送
- 累加器特殊(accumulator-specific)传送

。目的地址(address-object)传送

。标志寄存器传送

除了SAHF和POPF外，没有一个影响标志置位。

(1) 通用传送。提供了四种通用数据传送操作。虽然有特殊例外，这些数据传送操作可适用于大多数操作数。通用传送(除了XCHG外)是允许一个段寄存器作为一个操作数的唯一操作。

——MOV履行从源(最右边的)操作数到目标(最左边的)操作数的一个字节或字的传送。

——PUSH使SP寄存器减2，然后将一个字从源操作数传送到由当前SP所寻址的堆栈单元(stack element)。

——POP将一个字操作数从SP寄存器所寻址的堆栈单元传送到目标操作数，然后SP加2。

——XCHG将字节或字源操作数与目标操作数交换。段寄存器不能是XCHG的操作数。

(2) 累加器特殊传送。提供了三种累加器特殊传送操作。

——IN将一个字节(或字)从一个输入口传送到AL寄存器(或AX寄存器)该口用编码中的数据字节指定，允许固定存取口0～255，或由DX寄存器中的一个口数指定，允许变量存取64K个输入口。

——OUT，除了从累加器传送到输出口外，与IN类似。

——XLAT履行一个查表字节(table lookup byte)传送。AL寄存器用作一个变址到一个由BX寄存器寻址的256个字节的表的变址寄存器。把如此选择的字节操作数传送到AL。

(3) 目的地址传送。提供了三种目的地址传送操作：

——LEA(加载有效地址)把源操作数的偏移地址传送到目标操作数。源操作数必须是一个存贮器操作数，目标操作数必须是一个16位的通用寄存器、指示器或变址寄存器。

——LDS（把指示器加载到DS）从源操作数（必须是一个双字存贮器操作数）传送一个“目的指示器（pointer-object）”（即，一个含有一个偏移地址和一个段地址的32位目的码）到一对目标寄存器。段地址传送到DS段寄存器。偏移地址传送到你编码的16位通用寄存器、指示器或变址寄存器。

——LES（把指示器加载到ES），除了段地址传送到ES段寄存器外，与LDS类似。

(4) 标志寄存器传送。提供了四种标志寄存器传送操作：

——LAHF（用标志寄存器加载AH）把标志寄存器SF，ZF，AF，PF和CF（8080标志）传送到AH寄存器的特定位。

——SAHF（把AH存入标志寄存器）AH寄存器的特定位传送到标志寄存器SF，ZF，AF，PF 和CF。

——PUSHF（把标志压入堆栈）SP寄存器减2，所有标志寄存器传送到由SP寻址的堆栈单元的特定位。

——POPF（把标志弹出堆栈）由SP寻址的堆栈单元的特定位传送到标志寄存器，然后SP加2。

2. 算术操作

8086提供了若干不同类型的四种基本的数字操作。提供了8位和16位操作及有符号和无符号算术操作。使用有符号值的标准2的补码表示法。加法和减法操作作为有符号和无符号两种操作。在这些情况下，标志置位允许所作的有符号和无符号操作之间的差别（见条件转移）。提供修正操作以允许按未组合的十进制数字或组合的十进制表示法直接履行算术操作。

(1) 标志寄存器置位。六个标志寄存器由反映操作结果的某些性质的算术操作来置位或清除。它们通常遵循这些规律（见附录C）

——CF，当操作产生一个从结果的高位的进位（由加法）或一个

到结果的高位的借位（由减法）时，CF置位；否则，清零。

——AF，当操作产生一个从结果的低四位的进位（由加法）或一个到结果的低四位的借位（由减法）时，AF置位；否则，清零。

——ZF，当操作的结果是零时，ZF置位；否则，清零。

——SF，当操作结果的高位置位时，SF置位；否则，清零。

——PF，当操作结果的低8位的模2和(modulo 2 sum)是零(偶数奇偶校验)时，PF置位；否则，清零(奇数奇偶校验)。

——OF，当操作产生向结果的高位的进位但高位不产生进位时，OF置位，反之亦然；否则，清零。

(2) 加法。提供了五种加法操作：

——ADD履行源操作数和目标操作数的加法，并把结果回送到目标操作数。

——ADC（带进位加）履行源操作数和目标操作数的加法，若发现CF标志顶先置位，则加1，并把结果回送到目标操作数。

——INC（增1）履行源操作数和1的加法，并把结果回送到该操作数。

——AAA（加法的未组合BCD（ASCII）调整）履行AL中二未组合十进制操作数相加结果的修正，产生一个未组合十进制的和。

——DAA（加法的十进制调整）履行AL中二组合十进制操作数相加结果的修正，产生一个组合十进制的和。

(3) 减法。提供了七种减法操作：

——SUB履行从目标操作数中减去源操作数的减法，并把结果回送到目标操作数。

——SBB（带借位减）履行从目标操作数减去源操作数的减法，若发现CF标志顶先置位，则减1，并把结果回送到目标操作数。

——DEC（减1）履行从源操作数减去1的减法，并把结果回送

到该操作数。

——NEG（取补）履行从零减去源操作数的减法，并把结果回送到该操作数。

——CMP（比较）履行从目标操作数减去源操作数的减法，使标志受影响，但不回送结果。

——AAS（减法的未组合BCD（ASCII）调整）履行AL中二未组合十进制操作数相减结果的修正，产生一个未组合十进制的差。

——DAS（减法的十进制调整）履行AL中二组合十进制操作数的相减结果的修正，产生一个组合十进制的差。

(4) 乘法。提供了三种乘法操作：

——MUL履行累加器（AL或AX）与源操作数的无符号乘法，把一个双倍长结果回送到该累加器及其扩展寄存器（8位操作的AL和AH；16位操作的AX和DX）。当结果的高半部是非零时，CF和OF置位。

——IMUL（整数乘法）除了它履行一个带符号乘法外，与MUL类似。当结果的高半部不是结果的低半部的符号扩展时，CF和OF置位。

——AAM（乘法的未组合BCD（ASCII）调整）履行AX中二未组合十进制操作数相乘结果的修正，产生一个未组合十进制的积。

(5) 除法。提供了三种除法操作和两种符号扩展操作，以支持带符号的除法：

——DIV履行累加器及其扩展（8位操作的AL和AH；16位操作的AX和DX）被源操作数的无符号除法，并把单倍长的商回送到累加器（AL或AX），把单倍长的余数回送到累加器扩展寄存器（AH或AX）。标志没有定义。用零做除数产生一个方式0的中断。

——IDIV（整数除法），除了它履行一个带符号的除法外，与

DIV类似。

——AAD（除法的未组合BCD(ASCII)调整）履行AL中二未组合十进制操作数相除之前的被除数的修正，其结果将产生一个未组合十进制的商。

——CBW（字节转换为字）履行一个AL到AH的符号扩展。

——CWD（字转换为双字）履行一个AX到DX的符号扩展。

3. 逻辑操作

8086为8位和16位操作数提供了基本的逻辑操作。

(1) 单操作数操作。提供了三种单操作数逻辑操作：

——NOT形成源操作数的二进制反码，并把结果回送到该操作数。标志不受影响。

——为存贮器和寄存器操作数提供了四种移位操作，SHL（逻辑左移），RSH（逻辑右移），SAL（算术左移）和SAR（算术右移）可利用单位(single bit)移位和用从CL寄存器减去的移位计数的变位(variable bit)移位。CF标志变为移出的最后一位；只为计数为1的移位定义OF，且当最后的符号位值与先前该符号位值不同时，OF置位；PF，SF和ZF被置到反映出结果的值。

——为存贮器和寄存器操作数提供了四种环移操作，ROL（环左移），ROR（环右移），RCL（通过CF环左移）和RCR（通过CF环右移）。单位(single bit)环移和具有从CL寄存器得到的环移计数的变位(variable bit)环移是有用的。CF标志变为环移出的最后一位；OF只是为计算为1的移位定义的。当最后的符号位值与先前该符号位值不同时，OF置位。

(2) 双操作数操作。提供了四种双操作数逻辑操作。CF和OF标志在所有操作上都清零；SF，PF和ZF反映该结果。

——AND履行源操作数与目标操作数的位逻辑与（bitwise logical conjunction），并把结果回送到目标操作数。

——TEST履行与AND同样的操作，影响标志，但不回送结果。

——OR履行源操作数与目标操作数的位逻辑或（bitwise logical inclusive disjunction），并把结果回送到目标操作数。