

考点一、理解指令系统的概念及指令格式

1、指令系统和指令格式

考点二、掌握常用的寻址方式并能熟练的使用他们

2、8086 寻址方式

2.1、数据寻址方式

考点三、掌握常用指令的格式、功能和使用方式，分析各种指令执行后的结果。

3.1、数据传送指令

3.2、算术运算指令

3.3、位操作类指令

3.4、控制转移指令

3.5、处理器控制指令

3.6、串操作指令

1、指令系统和指令格式

(1) 指令是要求计算机执行特定操作的命令，一条指令对应一种特定操作，比如加、减、转移、移位等。

(2) 指令系统是计算机所能执行的全部指令的集合，是计算机硬件和软件之间的桥梁，是汇编语言程序设计的基础。

(3) 计算机指令以二进制的形式存放在存储器中，用二进制编码形式表示的指令称为机器指令。用符号表示的指令称为汇编指令，具有直观易理解的特点。汇编指令与机器指令有一一对应的关系。而且每种机器的 CPU 指令系统的指令都有几十条，上百条之多。

(4) 指令包括操作码字段和操作数字段两部分。

- ① 操作码字段：规定指令的操作类型。说明计算机要执行的操作，如传输、运算、移位、跳转等操作，是指令中必不可少的组成部分。
- ② 操作数字段：说明在指令需要的操作数。可以是操作数本身，也可以是操作数地址或是地址的一部分，还可以是指向操作数的地址指针或其他有关操作数的信息。

2、8086 寻址方式

寻址是指寻找操作数或操作数地址的过程。

寻址方式：指令中给出的找到操作数或操作数地址采用的方式。

8086 指令系统的寻址方式主要有立即数寻址、寄存器寻址、存储器寻址、和 i/o 端口寻址。其中，存储器寻址可进一步分为直接寻址、寄存器间接寻址、寄存器相对寻址、基址变址寻址、相对基址变址寻址。I/o 端口指令 in 和 out 使用的端口寻址方式有直接寻址和间接寻址。

2.1、数据寻址方式

1、立即数寻址

操作数（为一常数）直接由指令给出（此操作数称为立即数）。立即数只能用于源操作数。

实例：mov ax, 1C8FH

mov byte ptr[2A00H], 8FH

2、寄存器寻址

- (1) 操作数放在某个寄存器中。
- (2) 源操作数与目的操作数数字长要相等。

(3) 寄存器寻址与段地址无关。

示例: MOV AX, BX

3、I/O 寻址方式 (端口直接寻址、端口间接寻址)

4、存储器寻址方式

(1) 存储器的单元地址由三个分量地址组合而成:

位移分量 (用 disp 表示)

基址分量 (用 BX、BP 基址寄存器表示)

变址分量 (用 SI、DI 变址寄存器表示)

(2) 五种寻址方式

- ① 直接寻址方式 $EA = \text{disp16}/\text{disp8}$
- ② 寄存器间接寻址 $EA = (DS/SS/CS/ES) * 16 + (BX/BP/SI/DI)$
- ③ 寄存器相对寻址 $EA = (BX/BP/SI/DI) + \text{disp16}/\text{disp8}$
- ④ 基址加变址寻址 $EA = (BX/BP) + (SI/DI)$
- ⑤ 基址加变址相对寻址 $EA = (BX/BP) + (SI/DI) + \text{disp16}/\text{disp8}$

寄存器常用搭配:

CS:IP	固定
DS:BX、SI、DI 或位移量	默认
SS:SP	固定
SS:BP	默认
ES:DI (用于字符串操作指令)	固定

3、8086 指令系统

8086 微处理器指令系统中根据指令的操作性质,可分为六大类:传送类指令、运算类指令、逻辑类指令、转移类指令、串操作指令、控制类指令。

注意:

- 1、指令的基本功能。
- 2、指令的执行对标志位的影响。
- 3、对寻址方式或寄存器使用的限制和隐含使用的情况。

3.1、数据传送指令

1、MOV 指令

MOV dst,src;

Dst:表示目的操作数。(字节或字操作数)

寄存器操作数。(不包括 cs 段寄存器)

存储器操作数。

Src: 表示源操作数。(字节或字操作数)

立即数操作数。

寄存器操作数。(包括段寄存器)

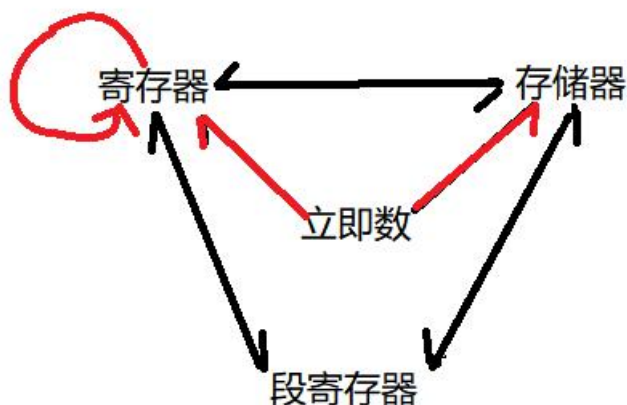
存储器操作数。

注意以下几点:

- (1) 目的操作数不能为立即数。
- (2) 目的操作数为段寄存器(Cs 不能作为目的操作数),源操作数不能为立即数。
- (3) 两操作数不能同时为存储器操作数。

(4) 操作数类型必须一致。

(5) 用 BX SI DI 来间接寻址时。默认段寄存器为 DS。用 BP 来间接寻址时，默认段寄存器为 ss。



2、堆栈指令

按照“先进后出”工作方式的存储区域，堆栈以字为单位进行压入弹出操作。

规定由 SS 指示堆栈段的段基址。堆栈指针 SP 始终指向堆栈的顶部，SP 的初值规定了所用堆栈区的大小。堆栈的最高地址叫栈底。

PUSH src;

示例: PUSH AX

Src: 表示源操作数。(字操作数)

寄存器操作数。(包括段寄存器)

存储器操作数。

(SP) ← SP-2

(SP+1:SP) ← src

AX = 1122H BX = 3344H CX = 5566H
SP = 2000H

开始时	PUSH AX SP=SP-2	PUSH BX SP=SP-2	PUSH CX SP=SP-2
1FFAH	1FFAH	1FFAH	SP -> 1FFAH
1FFBH	1FFBH	1FFBH	1FFBH 66H
1FFCH	1FFCH	SP -> 1FFCH	1FFCH 55H
1FFDH	1FFDH	1FFDH 44H	1FFDH 44H
1FFEH	SP -> 1FFEH	1FFEH 33H	1FFEH 33H
1FFFH	1FFFH 22H	1FFFH 22H	1FFFH 22H
SP -> 2000H	2000H 11H	2000H 11H	2000H 11H

POP dst;

示例: POP BX

Dst:表示源操作数。(字操作数)

寄存器操作数。(不包括CS)

存储器操作数。

src ← (SP+1:SP)
(SP) ← SP+2

注意以下几点:

- (1) 进栈操作(PUSH)先移后进。出栈操作(POP)先出后移。
- (2) 堆栈操作是字操作。而且不能是立即数。

3、交换指令 XCHG (交换两个操作数的内容)

XCHG reg, mem/reg

示例: XCHG AX, BX

XCHG [2000], CL

注意以下几点:

- (1) 两个操作数中必须有一个在寄存器中。
- (2) 操作数不能为段寄存器和立即数。
- (3) 源操作数和目的操作数类型要一致。

4、查表指令 XLAT

XLAT TABLE (TABLE 为一待查表格的首地址)

把待查表格的一个字节内容送到 AL 累加器中。在执行该指令前, 应将 TABLE 先送至 BX 寄存器中, 然后将待查字节与其在表格中距表首地址位移量送 AL, 即 $AL \leftarrow ((BX) + (AL))$ 。执行 XLAT 将使待查内容送到累加器。

本指令不影响状态标位, 表格长度不超过 256 字节。

如图, 数据段中存放有一张ASCII码转换表, 设首地址为2000H, 现欲查出表中第11个代码的ASCII码。

可用如下指令实现:

MOV BX, 2000H ; BX←表首地址

MOV AL, 0BH ; AL←序号

XLAT ; 查表转换

执行后: AL = 42H

还可用其他方法实现, 如:

MOV BX, 0BH

MOV AL, [BX+2000H]

2000H+0	30	'0'
	31	'1'
	32	'2'
	...	
	39	'9'
	41	'A'
2000H+11	42	'B'
	...	
	45	'E'
	46	'F'

5、LEA 指令 (传送偏移地址)

LEA reg16, mem;

源操作数必须是一个存储器操作数, 目的操作数必须是一个 16 位的通用寄存器。

注意以下两条指令的差别:

LEA AX, [1000H]; // (AX) = (1000H) 值

MOV AX, [1000H]; // (AX) = 1000H 地址

下列两条指令等效:

LEA BX, BUFFER

MOV BX, OFFSET BUFFER

OFFSET BUFFER 表示存储单元 buffer 的偏移地址。二者都用于取存储器单元的偏移地址, Lea 指令可以取动态地址。Offset 只能取静态地址。

6、IN 和 OUT 指令

只限于用累加器

(1) IN AL/AX, port/DX; port 端口号 0~255H

例如: IN AL, 80H; (AL) ← (80H 端口)

IN AL, DX; (AL) ← ((DX))

(2) OUT port/DX, AL/AX

例如: OUT 80H, AL; (80H 端口) ← (AL)

OUT DX, AL; ((DX)) ← (AL)

3.2、算术运算指令

可分为加法指令、减法指令、乘法指令、除法指令和十进制调整指令 5 种。

1、不带进位的加法指令 ADD

对六个状态为均产生影响。将目的操作数和源操作数相加, 再送入目的操作数。

2、带进位的加法指令 ADC

在形式上和功能上与 ADD 类似，只是相加时还要包括进位标志 CF 的内容。

3、加 1 指令 INC

类似于 c 语言中的++

4、减 1 指令 DEC

类似于 c 语言中的--

5、求补指令 NEG

对一个操作数取补码，相当于用 0 减去此操作数，故利用 neg 指令可得到负数的绝对值。


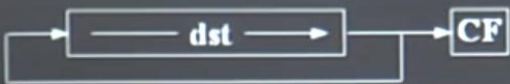
6、比较指令 CMP

执行两个操作数相减，但结果不送目的操作数。其结果只反映在标志位上。

指令格式	操作功能	对标志的影响					
		O	S	Z	A	P	C
ADD dst, src	$(dst) \leftarrow (dst) + (src)$	○	○	○	○	○	○
ADC dst, src	$(dst) \leftarrow (dst) + (src) + (CF)$	○	○	○	○	○	○
INC dst	$(dst) \leftarrow (dst) + 1$	○	○	○	○	○	×
DEC dst	$(dst) \leftarrow (dst) - 1$	○	○	○	○	○	×
NEG dst	$(dst) \leftarrow 0 - (src)$	○	○	○	○	○	1
CMP dst, src	$(dst) - (src)$	○	○	○	○	○	○

3.3、位操作类指令

可分为逻辑运算指令，移位指令和循环移位指令三种。

指令格式	操作功能	对标志的影响					
		O	S	Z	A	P	C
AND dst, src	$(dst) \leftarrow (dst) \wedge (src)$	0	○	○	×	○	0
OR dst, src	$(dst) \leftarrow (dst) \vee (src)$	0	○	○	×	○	0
XOR dst, src	$(dst) \leftarrow (dst) \oplus (src)$	0	○	○	×	○	0
ROL dst, 1/CL		○	○	○	×	○	○
ROR dst, 1/CL		○	○	○	×	○	○

位操作常见用法:

- (1) 操作数清零: XOR AX, AX 不仅把 AX 清零, 而且也影响了状态标志位。
- (2) 某几位取反: 用 XOR 指令, 把要取反的位和 1 进行异或, 不变的位和 0 异或。
- (3) 某几位清零: 用 AND 指令。需清零的位与 0 相与, 不变的位与 1 相与。
- (4) 某几位置一: 用 OR 指令。需要置 1 的位与 1 相或, 不变的位与 0 相或。
- (5) 字节中高低 4 位互换: 用 ROL 指令或 ROR 指令。

3.4、控制转移指令

无条件转移指令 (JMP)、条件转移指令 (Jcc)、循环控制指令 (LOOP) 和过程调用与返回指令 (CALL 与 RET) 四种。

1、无条件转移指令 JMP

可以修改 IP 或同时修改 CS 和 IP 的指令统称为转移指令。即转移指令就是可以控制 CPU 执行内存中某处的代码的指令。

- (1) 转移的距离: 段内和段间

只修改 IP 时, 称为段内转移, eg: `jmp ax`。

同时修改 CS 和 IP 时, 称为段间转移, eg: `jmp 1000:0`。

(2) 目标地址的给出方式: 直接和间接

(3) 无条件转移的寻址方式有四种

段内直接寻址方式、段内间接寻址方式、段间直接寻址方式、段间间接寻址方式

由于转移指令对 IP 的修改范围不同, 段内转移又分为: 短转移和近转移。

短转移 IP 的修改范围为 $-128 \sim 127$

近转移 IP 的修改范围为 $-32768 \sim 32767$

段内直接近转移: `JMP NEAR PTR 标号`;

段内直接短转移: `JMP SHORT 标号`;

段内间接转移: `JMP WORD PTR 标号`;

段间直接转移: `JMP FAR PTR 标号`;

段间间接转移: `JMP DWORD PTR 标号`;

3.5、处理器控制指令

1、CF 设置指令

`CLC`; $0 \rightarrow CF$

`STC`; $1 \rightarrow CF$

2、DF 设置指令

`CLD`; $0 \rightarrow DF$

STD; 1→DF

3、IF 设置指令

CLI; 0→IF

STI; 1→IF

4、HLT(CPU 进入暂停状态)

3.6、串操作指令

串传送指令 (MOVS)、串读取指令 (LODS)、串存储指令 (STOS)、串比较指令 (CMPS)、串扫描指令 (SCAS)。另外还有三个指令重复前缀 (REP、REPE/REPZ、REPNE/REPNZ)。

在使用串操作指令之前，需要先做一个准备工作，这些准备工作包括指明源串、目的串的起始位置，所要处理的数据的数量，以及数据查找的方向，即对 DS、SI、ES、DI、CX、DF 设定初值。

指令格式	操作功能
MOVSB/W	$((ES):(DI)) \leftarrow ((DS):(SI))$ $(SI) \leftarrow (SI) \pm 1$ $(DI) \leftarrow (DI) \pm 1$ $(SI) \leftarrow (SI) \pm 2$ $(DI) \leftarrow (DI) \pm 2$
LODSB/W	$(AL)/(AX) \leftarrow ((DS):(SI))$ $(SI) \leftarrow (SI) \pm 1$ 或 $(SI) \leftarrow (SI) \pm 2$
STOSB/W	$((ES):(DI)) \leftarrow (AL)/(AX)$ $(DI) \leftarrow (DI) \pm 1$ 或 $(DI) \leftarrow (DI) \pm 2$
REP	$(CX) \neq 0$ 时重复

(1) 目的串在附加段中段，地址为 ES, 偏移地址由 DI 表示。

(2) 源串在数据段中，段地址为 DS, 偏移地址由 SI 表示。

(3) 每执行一次串操作指令，指针都会自动进行修改。以便指向下一个待操作数所在的单元。

(4) 方向标志 DF 用于控制串指针的移动方向。DF=0 为递增方向, 执行一次字节操作, SI 和 DI 就增 1, 执行一次字操作, SI 和 DI 就增 2; DF=1 为递减方向, 执行一次字节操作, SI 和 DI 就减 1, 执行一次字操作, SI 和 DI 就减 2。

(5) 如果不加重复前缀, 串操作指令只执行一次。

(6) 如果加重复前缀, 串操作指令会自动重复执行, 重复的次数由 CX 寄存器决定, CX 寄存器中存放串数据的长度。