12.3.5 程序控制指令

1 无条件转移指令JMP

JMP指令的操作是无条件地使程序转移到指定的目标地址，并从该地址开始执行新的程序段。

段内转移：CS不变，仅改变IP的值；

段间转移：CS、IP寄存器的内容均发生改变。

转移指令对状态标志位没有影响。

1）段内直接转移

格式：JMP　Label

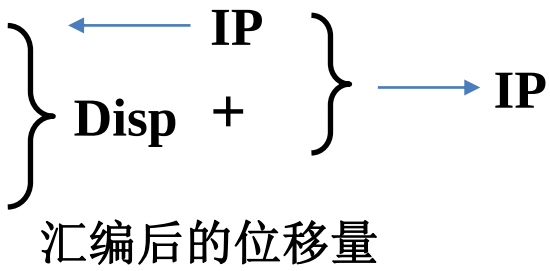
操作：LAB→IP

Label：段内标号，也称为符号地址，汇编程序确定转移目标代码的偏移地址，必须在同一代码段内。

即：JMP　XXXXH　；偏移地址

例，

```
        ⋮
        MOV  AX,    BX
        JMP  NEXT
        INC  CX
        ⋮
NEXT:   MOV  BX,    0
```



2）段内间接转移

格式：JMP　OPRD

OPRD：为通用寄存器或存储器根据相应寻址方式而获得的一个16位操作数来作为转移目标偏移地址。

操作：OPRD→IP

例，若DS=3000H，BX=1200H，
[31200H]=50H，[31201H]=23H

```
JMP    BX              ;  1200H→IP
JMP    WORD PTR  [BX]      ;  2350H→IP
```

3）段间直接转移

格式：JMP FAR PTR Label

Label：段间标号，也称远标号，在另一个代码段。汇编程序确定要转移的16位段地址和16位偏移地址，即标号所在的代码段及其位置的偏移地址。

操作：　段地址→ CS，偏移地址→IP

例，JMP FAR PTR NEXT

即，JMP XXXXH: XXXXH

4）段间间接转移

格式：JMP　OPRD

OPRD：是一个根据寻址方式而获得的32位存储器操作数。

操作：OPRD(高16位) → CS

OPRD(低16位) →IP

例，若DS=3000H，BX=3000H,[33000]=0BH,

[33001H]=20H，[33002H]=10H,[33003H]=80H

JMP　DWORD PTR [BX]

200BH→IP，8010H →CS

2 条件转移指令

- 根据前一条指令执行后标志位的状态来决定是否转移。
- 若满足转移条件，则转移到指定的地址，否则顺序执行下一条指令。
- 转移地址采用直接寻址方式的短转移，即只能以当前IP(转移指令的下一条)为中心的-128∽127字节。
- 条件转移不影响标志位。

简单条件转移指令

标志位	指令	转移条件	含义
CF	JC	CF=1	有进位/借位转移
	JNC	CF=0	无进位/借位转移
ZF	JE/JZ	ZF=1	相等/等于0转移
	JNE/JNZ	ZF=0	不相等/不等于0转移
SF	JS	SF=1	负数转移
	JNS	SF=0	正数转移
OF	JO	OF=1	有溢出转移
	JNO	OF=0	无溢出转移
PF	JP/JPE	PF=1	偶数个1转移
	JNP/JPO	PF=0	奇数个1转移

无符号数条件转移指令

指令	转移条件	含义
JA/JNBE	CF=0且ZF=0	A>B
JAE/JNB	CF=0	A≥B
JB/JNAE	CF=1	A<B
JBE/JNA	CF=1或ZF=1	A≤B

有符号数条件转移指令

指令	转移条件	含义
JG/JNLE	SF=OF且ZF=0	A>B
JGE/JNL	SF=OF或ZF=1	A≥B
JL/JNGE	SF≠OF且ZF=0	A<B
JLE/JNG	SF≠OF或ZF=1	A≤B

3 循环控制指令

- 循环的次数必须先送入CX寄存器中。
- 控制转向的目标偏移地址是以当前IP内容(循环控制指令的下一条)为中心的-128～127字节范围内标号所代表的偏移地址。
- 循环控制指令不影响状态标志位，标志位主要由之前指令改变。

1) LOOP指令

格式：LOOP Label

操作：CX−1→CX

- CX≠0，Label→IP
- CX=0，退出循环，执行下一条指令

例，
```
    LEA   BX,   Mem1
    MOV   CX,   100
T1: MOV   BYTE PTR[BX], 0
    INC   BX
    LOOP  T1
```

2) LOOPZ/LOOPE指令

格式：LOOPZ　Label

　　LOOPE　Label

操作：CX−1→CX

- CX≠0、ZF=1，Label→IP
- CX=0或ZF=0，退出循环，执行下一条指令

ZF=1、CX=0 ；ZF=0、 CX ≠0； ZF=0、 CX =0

3) LOOPNZ/LOOPNE指令

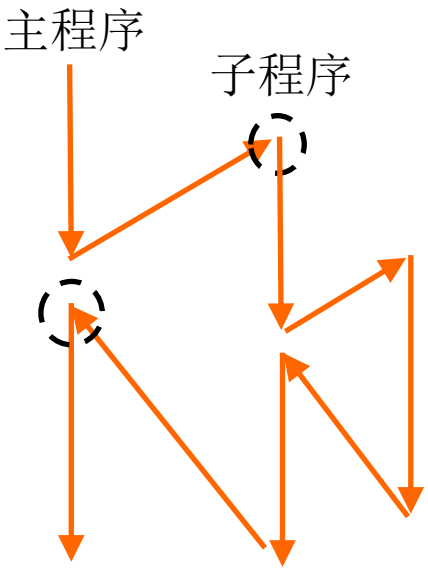
格式：LOOPNZ　Label

　　LOOPNE　Label

操作：CX−1→CX

- CX≠0、ZF=0，　Label →IP
- CX=0或ZF=1，退出循环，执行下一条指令

4 过程调用和返回



程序执行过程中，主程序可根据需要随时调用子程序（过程），子程序执行完后，返回到主程序（断点处）继续执行。

调用指令CALL：

保存主程序断点地址（返回地址）

子程序入口地址→ IP（CS和IP）

返回指令RET：

主程序返回地址→IP（CS和IP）

1）段内直接调用

格式：CALL　PROC

操作：SP−2→SP

IP高8位→SP+1

IP低8位→SP

PROC→IP

PROC：一个近过程的符号地址，汇编程序确定调用子程序的入口偏移地址，必须在同一代码段内。

即：CALL　XXXXH

2）段内间接调用

格式：CALL　OPRD

OPRD：为通用寄存器或存储器根据相应寻址方式而获得的一个16位操作数来作为入口偏移地址。

操作：SP−2→SP

IP高8位→SP+1

IP低8位→SP

OPRD→IP

3）段间直接调用

格式：CALL FAR PTR PROC

PROC：远过程的符号地址，在另一个代码段内。

汇编程序确定要调用子程序的16位段地址和16位偏移地址。

操作：　SP−2→ SP，CS→([SP+1],[SP])

SP−2→ SP，IP→ ([SP+1],[SP])

段地址→ CS，偏移地址→IP

4）段间间接调用

格式：CALL　OPRD

OPRD：是一个根据寻址方式而获得的32位存储器操作数。

操作：　SP−2→ SP，CS→ ([SP+1],[SP])

SP−2→ SP，IP→ ([SP+1],[SP])

高16位→ CS，低16位→IP

5）返回指令RET

格式：RET

操作：

段内：（[SP+1],[SP]) →IP,　SP+2→ SP

段间：（[SP+1],[SP]) →IP,　SP+2→ SP

（[SP+1],[SP]) →CS,　SP+2→ SP

返回指令一般位于子程序的最后一条，不影响标志位。

5 中断指令

中断指令用于产生软中断，以执行一段特殊用途的中断处理子程序。

格式：INT　n

n中断向量码(中断类型码)，取值范围0～255。

中断向量地址=n×4。

CPU根据中断向量地址读取内存，取出中断服务子程序的入口地址。

操作：

①SP−2→SP，FLAGS →([SP+1],[SP])

②TF → 0，IF → 0

③SP−2→SP，CS →([SP+1],[SP])

④SP−2→SP，IP →([SP+1],[SP])

⑤ ([n×4+1],[n×4]) → IP

⑥ ([n×4+3],[n×4+2]) → CS

中断返回指令

格式：IRET

中断返回指令位于中断服务子程序的最后一条，用于返回被中断的程序。

操作：

① ([SP+1],[SP]) → IP ， SP+2→SP

②([SP+1],[SP]) → CS ， SP+2→SP

③([SP+1],[SP]) → FLAGS ， SP+2→SP

12.3.6 处理器控制指令

类别	汇编格式	操　　作
标志位操作指令	CLC	CF→0，清进位标志位
	STC	CF→1，置进位标志位
	CMC	进位标志位取反
	CLD	DF→0，清方向标志位
	STD	DF→1，置方向标志位
	CLI	IF→0，清中断标志位，关中断
	STI	IF→1，置中断标志位，开中断
外部同步指令	HLT	使CPU处于暂停状态，常用于等待中断
	WAIT	CPU进入等待状态，用于协处理器或外部设备的同步
	ESC	处理器交权指令，用于与协处理器的配合
	LOCK	总线锁定指令
	NOP	空操作指令，消耗3个时钟周期