可以修改IP，或同时修改CS和IP的指令统称为转移指令。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | AX  AH、AL | accumulator  累加寄存器 | | BX  BH、BL | base  基址寄存器 | | CX  CH、CL | count  计数寄存器 | | DX  DH、DL | data  数据寄存器 | | SP | Stack Pointer  堆栈指针寄存器  栈顶元素的偏移地址 | | BP | Base Pointer  基址指针寄存器 | | SI | Source Index  源变址寄存器 | | DI | Destination Index  目的变址寄存器 | | |  |  | | --- | --- | | IP | Instruction Pointer  指令指针寄存器  CPU当前读取指令的偏移地址 | | |  |  | | --- | --- | | CS | Code Segment  代码段寄存器  CPU当前读取指令的段地址 | | DS | Data Segment  数据段寄存器  要访问的数据的段地址 | | ES | Extra Segment  附加段寄存器 | | SS | Stack Segment  堆栈段寄存器  栈顶元素的段地址 | |

Flag

标志寄存器

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|  |  |  |  |  | OF  Overflow Flag  溢出标志 | DF  Direction Flag  方向标志 | IF  Interrupt Flag  中断标志 | TF  Trap Flag  陷阱标志 | SF  Sign Flag  符号标志 | ZF  Zero Flag  零标志 |  | AF  Auxiliary carry Flag  辅助进位标志 |  | PF  Parity Flag  奇偶标志 |  | CF  Carry Flag  进位标志 |
| 0 |  |  |  |  | NV  No Overflow | UP  UP | DI  Disable Interrupt |  | PL  PLus | NZ  Not Zero |  | NA  No Auxiliary |  | PO  Parity Odd |  | NC  No Carry |
| 1 |  |  |  |  | OV  OVerflow | DN  DowN | EI  Enable  Interrupt |  | NG  NeGative | ZR  ZeRo |  | AC  Auxiliary Carry |  | PE  Parity Even |  | CY  CarrY |

立即数

|  |  |  |  |
| --- | --- | --- | --- |
|  | debug | 源文件 | |
| 二进制 | / | 后缀b或B | 1、可以加一层中括号。 |
| 八进制 | / | 后缀o或O |
| 十进制 | / | 无后缀或后缀d或D |
| 十六进制 |  | 不能以字母开头，后缀h或H |
|  | / | '**字符**' | 1、扩展时高位补0。 |
|  | / | '**字符字符**' | 1、高位第1个字符，低位第2个字符。 |
| 段地址 | / | **段名** |  |
|  |  | offset **标号** | 1、相当于标号相对于当前段的偏移地址。例：   |  | | --- | | assume cs:codesg  codesg segment  start:  mov ax,offset codesg  mov ax,offset start  mov ax,offset s1  mov ax,offset s2  s1:  mov ax,offset codesg  mov ax,offset start  mov ax,offset s1  mov ax,offset s2  s2:  mov ax,offset codesg  mov ax,offset start  mov ax,offset s1  mov ax,offset s2  codesg ends  end start | |

|  |  |  |
| --- | --- | --- |
| 地址标号 | **标号名**: | 只有最后1个和CS关联的段可以包含地址标号。  1、后面跟数据声明或指令。  2、表示内存单元的地址。 |
| 数据标号 | **标号名** | 1、后面跟数据声明。  2、表示内存单元的地址和大小。 |

|  |
| --- |
| assume cs:codesg  codesg segment  start:  mov al,'1'  ;mov al,'12'  ;mov al,'123'  ;mov al,'1234'  mov ax,'1'  mov ax,'12'  ;mov ax,'123'  ;mov ax,'1234'  mov ax,datasg  mov ds,ax  mov a1[0],'1'  ;mov a1[0],'12'  ;mov a1[0],'123'  ;mov a1[0],'1234'  mov a2[0],'1'  mov a2[0],'12'  ;mov a2[0],'123'  ;mov a2[0],'1234'  mov a3[0],'1' ;后面增加一条指令0000h  mov a3[0],'12' ;后面增加一条指令0000h  mov a3[0],'123' ;后面增加一条指令3100h  mov a3[0],'1234' ;后面增加一条指令3100h  mov ax,4c00h  int 21h  ;start  codesg ends  end start |

|  |
| --- |
| define byte  define word  define double word  (db|dw|dd) **立即数**(,**立即数**)\*  (db|dw|dd) **正整数** dup(**立即数**(,**立即数**)\*)  db '**字符**(**字符**)\*'  (dw|dd)('**字符**'|'**字符字符**')(, ('**字符**'|'**字符字符**'))\* |

内存单元

|  |  |
| --- | --- |
| ~~[~~**~~立即数~~**~~]~~  1、只能用于debug中输入指令。  **段寄存器**:**立即数**  **段寄存器**:[**立即数**] |  |
| (**段寄存器**:)?[bx(+**立即数**)?]  (**段寄存器**:)?[si(+**立即数**)?]  (**段寄存器**:)?[di(+**立即数**)?]  1、若不写立即数则默认0。  2、若不写段寄存器则默认ds。  (**段寄存器**:)?[bp(+**立即数**)?]  3、若不写段寄存器则默认ss。 | 1、寄存器提供结构体地址，立即数提供结构体的成员的“偏移”地址。立即数相当于c语言里的.**成员名**、->**成员名**运算符。 |
| (**段寄存器**:)?[bx+si]  (**段寄存器**:)?[bx+di]  1、若不写段寄存器则默认ds。  (**段寄存器**:)?[bp+si]  (**段寄存器**:)?[bp+di]  2、若不写段寄存器则默认ss。 | 1、第1个寄存器提供数组的地址，第2个寄存器提供数组的元素的“偏移”地址。 |
| (**段寄存器**:)?[bx+si+**立即数**]  (**段寄存器**:)?[bx+di+**立即数**]  1、若不写段寄存器则默认ds。  (**段寄存器**:)?[bp+si+**立即数**]  (**段寄存器**:)?[bp+di+**立即数**]  2、若不写段寄存器则默认ss。 |  |

|  |  |  |
| --- | --- | --- |
| 有寄存器 | | / |
| 没有寄存器 | 指令只进行字节操作或只进行字操作 |
| 字节操作 | 命令名后插入 byte ptr |
| 字操作 | 命令名后插入 word ptr |
| 双字操作 | 命令名后插入 dword ptr |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| inc  1、内容加1。  dec  2、内容减1。   |  |  | | --- | --- | | 立即数 | × | | 内存单元 | 需要单元长度 | | 寄存器 |  | | 段寄存器 | × | |  |
| add  sub  逻辑与指令and  逻辑或指令or  adc  1、左+=右+cf  sbb  2、左-=右+cf   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  |  | 右 | | | | |  |  | 立即数 | 内存单元 | 寄存器 | 段寄存器 | | 左 | 立即数 | × | | | | | 内存单元 | 需要单元长度 | × |  | × | | 寄存器 |  |  |  | × | | 段寄存器 | × | | | | |  |
| mul   |  |  | | --- | --- | | 乘数 | 结果 | | 都是8位 | AX | | 都是16位 | DX高16位，AX低16位 |   div   |  |  |  |  | | --- | --- | --- | --- | | 除数 | 被除数 | 余数 | 商 | | 8位 | AX 16位 | AH | AL | | 16位 | DX高16位，AX低16位 | DX | AX |  |  |  | | --- | --- | | 立即数 | × | | 内存单元 | 需要单元长度 | | 寄存器 |  | | 段寄存器 | × | |  |
| mov   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  |  | 右 | | | | |  |  | 立即数 | 内存单元 | 寄存器 | 段寄存器 | | 左 | 立即数 | × | | | | | 内存单元 | 需要单元长度 | × |  |  | | 寄存器 |  |  |  |  | | CS | × | | | | | 除CS段寄存器 | × |  |  | × | |  |
| push  1、SP-=2，数据写入SS:SP。  pop  2、将SS:SP数据读出，SP+=2。   |  |  |  | | --- | --- | --- | |  | push | pop | | 立即数 | × | | | 内存单元 |  |  | | 寄存器 |  |  | | CS |  | × | | 除CS段寄存器 |  |  |   pushf  3、SP-=2，标志寄存器写入SS:SP。  popf  4、将SS:SP数据读出到标志寄存器，SP+=2。 |  |
| 无条件转移指令jmp   |  |  |  | | --- | --- | --- | |  |  | (标号的地址-jmp指令后字节的地址) | | 段内短转移 | jmp short 标号  1、这里的标号不包括段名。 | -128到127 | | 段内近转移 | jmp near ptr 标号  1、这里的标号不包括段名。 | -32768到32767 | | jmp word ptr内存单元  1、内存单元存放目的偏移地址。 |  | | 段间转移 | ~~jmp 段地址:偏移地址~~  1、只能用于debug中输入指令 |  | | jmp far ptr 标号  1、这里的标号不包括段名。 | | jmp dword ptr 内存单元  1、高地址处的字存放目的段地址，低地址处的字存放目的偏移地址。 |   jmp 寄存器  1、修改IP的内容。 |  |
| 有条件转移指令jcxz  jcxz 标号  1、若CX==0，跳转。  2、这里的标号不包括段名。  3、(标号的地址-jcxz指令后字节的地址)在-128到127之间。 |  |
| 循环指令loop  loop 标号  1、先CX-=1，后决定是否跳转。若CX!=0，跳转。  2、这里的标号不包括段名。  3、(标号的地址-loop指令后字节的地址)在-128到127之间。 | 1、例：   |  |  | | --- | --- | | assume cs:codesg  codesg segment  start:  db 126 dup(0)  loop start  codesg ends  end start | assume cs:codesg  codesg segment  start:  db 127 dup(0)  loop start  codesg ends  end start | | assume cs:codesg  codesg segment  loop start  db 127 dup(0)  start:  codesg ends  end start | assume cs:codesg  codesg segment  loop start  db 128 dup(0)  start:  codesg ends  end start | | 正确 | 错误 | |
| |  |  | | --- | --- | | 转移指令ret | 转移指令call | | ret  1、相当于pop IP。 | call 标号  1、相当于push IP jmp near ptr 标号。  2、这里的标号不包括段名。 | | call 16位寄存器  1、相当于push IP jmp 16位寄存器。 | | call word ptr 内存单元  1、相当于push IP jmp word ptr内存单元。 | | retf  2、相当于pop IP pop CS。 | call far ptr 标号  1、相当于push CS push IP jmp far ptr 标号。  2、这里的标号不包括段名。 | | call dword ptr 内存单元  1、相当于push CS push IP jmp dword ptr 内存单元。 | |  |
| cmp  1、对标志寄存器的影响与sub指令相同，但不保存减的结果。   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  |  | 右 | | | | |  |  | 立即数 | 内存单元 | 寄存器 | 段寄存器 | | 左 | 立即数 | × | | | | | 内存单元 | 需要单元长度 | × |  | × | | 寄存器 |  |  |  | × | | 段寄存器 | × | | | | | 1、   |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 无符号数 | |  |  |  | | --- | --- | --- | |  | ZF | CF | | 0 | 左≠右 | 左≥右 | | 1 | 左＝右 | 左＜右 | | | 有符号数 | |  |  | | --- | --- | |  | ZF | | 0 | 左≠右 | | 1 | 左＝右 |  |  |  |  |  | | --- | --- | --- | --- | |  |  | SF | | |  |  | 0 | 1 | | OF | 0 | 左≥右 | 左＜右 | | 1 | 左＜右  左－右＜下界 | 左＞右  左－右＞上界 | | |
| 根据无符号数比较结果转移   |  |  | | --- | --- | | je(jump equal) | 相等（ZF＝1） | | jne | 不相等（ZF＝0） | | jb(jump below) | 小于（CF＝1） | | jnb | 大于等于（CF＝0） | | ja(jump above) | 大于（CF＝0∧ZF＝0） | | jna | 小于等于（CF＝1∨ZF＝1） | |  |
| |  |  |  | | --- | --- | --- | |  | df | if | | 置0 | cld | cli | | 置1 | std | sti | |  |
| movsb   |  |  | | --- | --- | | df==0 | 相当于  mov es[di]:ds:[si]  inc si  inc di | | df==1 | 相当于  mov es[di]:ds:[si]  dec si  dec di |   rep movsb  1、相当于   |  | | --- | | s: movsb  loop s |   movsw   |  |  | | --- | --- | | df==0 | 相当于  mov es[di]:ds:[si]  inc si  inc si  inc di  inc di | | df==1 | 相当于  mov es[di]:ds:[si]  dec si  dec si  dec di  dec di |   rep movsw  1、相当于   |  | | --- | | s: movsw  loop s | |  |
| 逻辑移位指令  shl（shift left）  shr（shift right）  1、移出的位写入CF，空位补0。   |  |  |  | | --- | --- | --- | | 左 | 立即数 | × | | 内存单元 | 需要单元长度 | | 寄存器 |  | | 段寄存器 | × |   第二个操作数是1或cl。 |  |
| in (al|ah) (字节型立即数|dx)  out (字节型立即数|dx) (al|ah)  读写端口号指定的端口的数据。 |  |

中断

1、执行完mov SS,寄存器后CPU不响应中断。（将设置SP的指令紧靠在设置SS的指令后。）

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 内中断   |  |  | | --- | --- | |  | 中断类型码 | | 除法错误 | 0 | | 单步执行 | 1 | | into指令 | 4 | | int指令 | 字节型立即数 |   单步中断  CPU执行完1条指令后，若TF==1，则产生单步中断（相当于int 1）。  int指令   |  |  | | --- | --- | | int 字节型立即数  相当于  pushf  TF=0 IF=0  push CS  push IP  mov IP,4\*字节型立即数  mov CS,4\*字节型立即数+2 | iret  相当于  pop IP  pop CS  popf |   BIOS提供的中断例程   |  |  |  | | --- | --- | --- | | int 中断类型码 | ah子程序编号 |  | | int 19h |  | 进行操作系统的引导 | | int 10h | 2 | 置光标  bh页号（0～7）  dh行号（0～24）  dl列号（0～79） | | 9 | 在光标位置显示字符  al字符  bl样式（闪烁、背景RGB、高亮、前景RGB）  bh页号（0～7）  cx重复个数 |   DOS提供的中断例程   |  |  |  | | --- | --- | --- | | int 中断类型码 | ah子程序编号 |  | | int 21h | 4ch | 程序返回  al返回值 | | 9 | 在光标位置显示字符串（遇到行尾自动换行，遇到最后一行自动上卷一行）  ds:dx字符串（'$'表示结束） | |
| 外中断   |  |  | | --- | --- | | IF=0 | 不响应可屏蔽中断 | | IF=1 | 响应可屏蔽中断 |   不可屏蔽中断的中断类型码是2。  可屏蔽中断  若IF=1，相当于int 中断类型码。   |  |  |  | | --- | --- | --- | |  | 中断类型码 |  | | 键盘输入（按下、松开） | 9 | 扫描码（按下：通码。松开：断码。断码=通码+80h）说明按下或松开的键在键盘上的位置，被送入端口地址为60h的寄存器。 |   BIOS提供的中断例程   |  |  |  | | --- | --- | --- | | int 中断类型码 | ah子程序编号 |  | | int 9 |  | 若扫描码是字符键扫描码，将扫描码和字符码（ASCII码）送入内存中的BIOS键盘缓冲区（可存储15个键盘输入，高字节扫描码，低字节字符码）；若扫描码是控制键和切换键的扫描码，将其转变为状态字节后写入内存中存储状态字节的单元（0040:17，Insert、CapsLock、NumLock、ScrollLock、Alt、Ctrl、左shift、右shift）。对键盘进行相应的控制。 | |

# 数据传送指令

# 算术运算指令

# 逻辑指令

# 转移指令

# 处理机控制指令

# 串处理指令