**80x86指令系统**

80x86指令系统，指令按功能可分为以下七个部分。  
　　(1) 数据传送指令。  
　　(2) 算术运算指令。  
　　(3) 逻辑运算指令。  
　　(4) 串操作指令。  
　　(5) 控制转移指令。  
　　(6) 处理器控制指令。  
　　(7) 保护方式指令。

3.3.1数据传送指令  
　　数据传送指令包括：通用数据传送指令、地址传送指令、标志寄存器传送指令、符号扩展指令、扩展传送指令等。  
　　一、通用数据传送指令  
　　1传送指令  
　　传送指令是使用最频繁的指令，格式：MOV DEST,SRC  
　　功能：把一个字节，字或双字从源操作数SRC传送至目的操作数DEST。

数据允许流动方向为：通用寄存器之间、通用寄存器和存储器之间、通用寄存器和段寄存器之间、段寄存器和存储器之间，另外还允许立即数传送至通用寄存器或存储器。但在上述传送过程中，段寄存器CS的值不能用传送指令改变。

例 3.12CPU内部寄存器之间的数据传送。  
　　MOV AL，DH　　　　；AL←DH　　　　(8位)  
　　MOV DS，AX　　　　；DS←AX　　　　(16位)  
　　MOV EAX，ESI　　　；EAX←ESI　　　(32位)  
　　例 3.13CPU内部寄存器和存储器之间的数据传送。  
　　MOV [BX],AX　　　　　　 ；间接寻址　　　　　(16位)  
　　MOV EAX，[EBX+ESI]　　　；基址变址寻址　　　(32位)  
　　MOV AL，BLOCK　　　　　 ；BLOCK为变量名，直接寻址(8位)  
例 3.14立即数送通用寄存器、存储器。  
　　MOV EAX，12345678H　　　；EAX←12345678H　　　(32位)  
　　MOV [BX]，12H　　　　　 ；间接寻址　　　　　　(8位)  
　　MOV AX，1234H；AX←1234H(16位)  
　　使用该指令应注意以下问题：  
　　·源和目的操作数不允许同时为存储器操作数；  
　　·源和目的操作数数据类型必须一致；  
　　·源和目的操作数不允许同时为段寄存器；  
　　·目的操作数不允许为CS和立即数；  
　　·当源操作数为立即数时，目的操作数不允许为段寄存器；  
　　·传送操作不影响标志位。  
　　2扩展传送指令  
　　格式：MOV SX　DEST，SRC  
　　　　　MOV ZX　DEST，SRC  
　　功能：将源操作数由8位扩展到16位送目的操作数，或由16位扩展到32位送目的操作数。其中MOVSX是按有符号数扩展，MOVZX是按无符号数扩展。无符号数或正数高位扩展为0，负数高位扩展为全“1”。

例 3.15带符号数扩展  
　　MOV BL，80H　　；　-128  
　　MOVSX AX，BL　　；　将80H扩展为FF80H后送AX中。  
　　例 3.16无符号数扩展  
　　MOV BL，80H　　；　128  
　　MOVZX AX，BL　　；　将80H扩展为0080H后送AX中。  
　　使用该指令应注意以下问题：  
　　·目的操作数应为16位或32位通用寄存器；  
　　·源操作数长度须小于目的操作数长度，为8位或16位通用寄存器或存储器操作数；  
　　·扩展传送操作不影响标志位。  
　　3交换指令  
　　(1) 格式：XCHG OPR1，OPR2  
　　功能：交换操作数OPR1和OPR2的值，操作数数据类型为字节、字或双字。允许通用寄存器之间，通用寄存器和存储器之间交换数据。  
　　例 3.17  
　　XCHG AX，BX；通用寄存器之间交换数据(16位)  
　　XCHG ESI，EDI；通用寄存器之间交换数据(32位)  
　　XCHG BX，/[SI/]；通用寄存器和存储器之间交换数据(16位)  
　　XCHG AL，/[BX/]；通用寄存器和存储器之间交换数据(8位)  
　　使用该指令应注意以下问题：  
　　·操作数OPR1和OPR2不允许同为存储器操作数；  
　　·操作数数据类型必须一致；  
　　·交换指令不影响标志位。  
　　如要实现存储器操作数交换，可用如下指令实现：  
　　　　　　　　　　　　　　　MOV AL，BLOCK1  
　　　　　　　　　　　　　　　XCHG AL，BLOCK2  
　　　　　　　　　　　　　　　MOV BLOCK1，AL

(2) 格式：BSWAP REG  
　　功能：将32位通用寄存器中，第1个字节和第4个字节交换，第2个字节和第3个字节交换。  
例 3.18  
　　MOV EAX，44332211H  
　　BSWAP EAX；EAX=11223344H  
　　使用该指令应注意以下问题：  
　　·操作数为32位通用寄存器；  
　　·交换指令不影响标志位。

二、堆栈操作指令  
　　1压栈指令  
　　(1) 格式：PUSH SRC  
　　功能：将源操作数压下堆栈，源操作数允许为16位或32位通用寄存器、存储器和立即数以及16位段寄存器。当操作数数据类型为字类型，压栈操作使SP值减2；当数据类型为双字类型，压栈操作使SP值减4。  
**例 3.19**  
　　PUSH AX　　　　　　　　　　；通用寄存器操作数入栈(16位)  
　　PUSH EBX　　　　　　　　　 ；通用寄存器操作数入栈(32位)  
　　PUSH [SI]　　　　　　　　　；存储器操作数入栈(16位)  
　　PUSH DWORD PTR [DI]　　　　；存储器操作数入栈(32位)  
　　PUSHW 0A123H　　　　　　　 ；立即数入栈(16位)  
　　PUSHD 20H　　　　　　　　　；立即数入栈(32位)  
　　(2) 格式：PUSHA  
　　　　　　　PUSHAD  
　　功能：PUSHA将16位通用寄存器压入堆栈，压栈顺序为AX，CX，DX，BX，SP，BP，SI，DI。  
PUSHAD将32位通用寄存器压入堆栈，压栈顺序为EAX，ECX，EDX，EBX，ESP，EBP，ESI，EDI。  
　　2出栈指令  
　　(1) 格式：POP DEST  
　　功能：从栈顶弹出操作数送入目的操作数。目的操作数允许为16或32位通用寄存器、存储器和16位段寄存器。当操作数数据类型为字类型，出栈操作使SP加2；当操作数数据类型为双字类型，出栈操作使SP加4。

**例 3.20**  
　　POP AX　　　　　　　　　　；操作数出栈送寄存器(16位)  
　　POP ECX　　　　　　　　　 ；操作数出栈送寄存器(32位)  
　　POP [BX]　　　　　　　　　；操作数出栈送存储器(16位)  
　　POP DWORD PTR [SI]　　　　；操作数出栈送存储器(32位)  
　　(2) 格式：POPA  
　　　　　　　POPAD  
　　功能：POPA从堆栈移出16字节数据，并且按顺序存入寄存器DI，SI，BP，SP，BX，DX，CX，AX中。  
　　POPAD从堆栈移出32字节数据，并且按顺序存入寄存器EDI，ESI，EBP，ESP，EBX，EDX，ECX，EAX中。  
　　使用堆栈操作指令应注意以下问题。  
　　(1) 目的操作数不允许为CS以及立即数。  
　　(2) 堆栈操作指令不影响标志位。  
　　三、地址传送指令  
　　(1) 格式：LEA REG，MEM  
　　功能：将源操作数的有效地址传送到通用寄存器，操作数REG为16位或32位通用寄存器，源操作数为16位或32位存储器操作数。  
**例 3.21**  
　　LEA BX，BLOCK；将BLOCK的有效地址传送到BX中(16位)  
　　LEA EAX，/[EBX/]；将EBX内容(有效地址)传送到EAX中(32位)  
　　(2) 格式LDS(ES，FS，GS，SS)REG，MEM  
　　功能：根据源操作数指定的偏移地址，在数据段中取出段地址和偏移地址分别送指定的段寄存器和指定的通用寄存器。

**例 3.22**  
　　LES BX，[SI]　　　　　　　　；将32位地址指针分别送ES和BX  
　　LSS EAX，[EDI]　　　　　　　；将48位地址指针分别送SS和EAX  
　　　  
**例 3.23**  
　　DATA1　　DD buff  
　　　  
　　LDS BX，DATA1；将buff的32位地址指针分别送DS和BX  
　　　  
地址传送指令对标志位无影响。  
　　四、标志寄存器传送指令  
　　(1) 格式：LAHF  
　　　　　　　SAHF  
　　功能：LAHF将标志寄存器中低8位送AH中。SAHF将AH中内容送标志寄存器中低8位。  
　　(2) 格式：PUSHF  
　　　　　　　POPF  
　　功能：PUSHF将标志寄存器低16位内容压入堆栈，SP←SP-2。POPF将当前栈顶一个字传送到标志寄存器低16位中，SP←SP+2。  
　　(3) 格式：PUSHFD  
　　　　　　　POPFD  
　　功能：PUSHFD将标志寄存器32位内容压入堆栈SP←SP-4。POPFD将当前栈顶一个双字传送到32位标志寄存器中，SP←SP+4。  
　　上述SAHF，POPF，POPFD均影响相应的标志寄存器内容。

五、查表指令  
　　格式：XLAT  
　　功能：将寄存器AL中的内容转换成存储器表格中的对应值。实现直接查表功能。  
　　XLAT指令规定：BX寄存器存放表的首地址，AL寄存器中存放表内偏移量，执行XLAT指令，以段寄存器DS的内容为段基址，有效地址为BX和AL内容之和，取出表中一个字节内容送AL中。  
　　**例 3.24**内存中有一起始地址为TABLE的编码表，试编程将表中顺序号为4的存储单元内容送寄存器AL。  
　　　　　　　　　·MODEL SMALL  
　　　　　　　　　·DATA  
　　TABLE 　　　　　DB 11H，22H，33H，44H，55H　某编码表  
　　　　　　　　　·CODE  
　　　　　　　　　·STARTUP  
　　　　　　　　　　MOV AL，4　　　　　　　　　　 ；AL←4  
　　　　　　　　　　MOV BX，OFFSET TABLE　　　　　；BX←TABLE表首地址  
　　　　　　　　　　XLAT　　　　　　　　　　　　　；结果在AL中，AL=55H  
　　　　　　　　　·EXIT  
　　　　　　　　　　END  
　　查表指令不影响标志位。

六、符号扩展指令  
　　(1) 格式：CBW  
　　功能：将AL中8位带符号数，进行带符号扩展为16位，送AX中。带符号扩展是指对正数高位扩展为全“0”，对负数高位扩展为全“1”。  
**例 3.25**AL=55H　经CBW扩展后　AX=0055H  
　　　　　　AL=A5H　经CBW扩展后　AX=FFA5H  
　　(2) 格式：CWD  
　　功能：将AX中16位带符号数，进行带符号扩展为32位，送DX和AX中。高16位送DX中，低16位送AX中。  
　　(3) 格式：CWDE  
　　功能：将AX中16位带符号数，进行带符号扩展为32位，送EAX中。  
　　(4) 格式：CDQ  
　　功能：将EAX中32位带符号数，进行带符号扩展为64位，送EDX和EAX中。低32位送EAX中，高32位送EDX中。  
　　符号扩展指令对标志位无影响。  
**3.3.2　算术运算指令**  
　　80x86指令包括加、减、乘、除四种基本算术运算操作及十进制算术运算调整指令。二进制加、减法指令，带符号操作数采用补码表示时，无符号数和带符号数据运算可以使用相同的指令。二进制乘、除法指令分带符号数和无符号数运算指令。  
　　一、加法指令  
　　格式：ADD DEST，SRC  
　　　　　ADC DEST，SRC  
　　功能：ADD是将源操作数与目的操作数相加，结果传送到目的操作数。ADC是将源操作数与目的操作数以及CF(低位进位)值相加，结果传送到目的操作数。  
　　源操作数可以是通用寄存器、存储器或立即数。目的操作数可以是通用寄存器或存储器操作数。  
ADD，ADC指令影响标志位为OF，SF，ZF，AF，PF，CF。

**例 3.26**  
　　MOV AX，9876H  
　　ADD AH，AL；AX=0E76H　CF=1　SF=0O　F=0　ZF=0　AF=0　PF=0  
　　ADC AH，AL；AX=8576H　CF=0　SF=1O　F=1　ZF=0　AF=1　PF=0  
　　二、减法指令  
　　格式：SUB DEST，SRC  
　　　　　SBB DEST，SRC  
　　功能：SUB将目的操作数减源操作数，结果送目的操作数。SBB将目的操作数减源操作数，还要减CF(低位借位)值，结果送目的操作数。  
　　源操作数可以是通用寄存器、存储器或立即数。目的操作数可以是通用寄存器或存储器操作数。  
SUB，SBB指令影响标志位为OF，SF，ZF，AF，PF，CF。  
**例 3．27**  
　　MOV AX， 9966H；AX=9966H  
　　SUB AL, 80H;AL=E6HCF=1SF=1OF=1ZF=0AF=0PF=0  
　　SBB AH, 80H;AH=18HCF=0SF=0OF=0ZF=0AF=0PF=1  
　　三、加1减1指令  
　　格式：INC DEST  
　　　　　DEC DEST  
　　功能：INC指令将目的操作数加1，结果送目的操作数。DEC指令将目的操作数减1，结果送目的操作数。目的操作数为通用寄存器或存储器操作数。  
　　INC，DEC指令影响标志位为OF，SF，ZF，AF，PF。  
**例 3．28**  
　　INC BL；BL←BL+1  
　　INC AX;AX←AX+1  
　　INC WORDPTR [BX];存储器操作数加1  
　　DEC BYTE PTR [SI];存储器操作数减1  
　　DEC EAX；EAX←EAX-1  
　　四、比较指令  
　　(1) 格式：CMP DEST，SRC  
　　功能：目的操作数减源操作数，结果不回送。源操作数为通用寄存器、存储器和立即数。目的操作数为通用寄存器、存储器操作数。  
　　CMP指令影响标志位为OF，SF，ZF，AF，PF，CF。

**例 3．29**  
　　CMP CX，3  
　　CMP WORD PTR [SI]，3  
　　CMP AX，BLOCK  
　　执行比较指令后，对状态标志位影响见表3.2。对于两个数的比较(AX-BX)有以下3种情况。

· 两个正数比较，使用SF标志位判断。  
　　　　SF=0，则AX≥BX，若ZF=1，则AX=BX  
　　　　SF=1，则AX<BX  
　　· 两个无符号数比较，使用CF标志位判断。  
　　　　CF=0，则AX≥BX，若ZF=1，则AX=BX  
　　　　CF=1，则AX<BX  
　　· 两个负数比较，使用SF标志位判断。  
　　　　SF=0，则AX≥BX，若ZF=1，则AX=BX  
　　　　SF=1，则AX<BX  
　　· 两个异符号数比较。  
　　　　如果OF=0，仍可用SF标志判断大小。  
　　　　如果OF=1，说明结果的符号位发生错误，所以  
　　　　　　　　　SF=0，则AX<BX  
　　　　　　　　　SF=1，则AX＞BX  
　　综上所述：两个异号数比较  
　　当OF=0，SF=0，则AX＞BX  
　　　　　　SF=1，则AX<BX  
　　当OF=1，SF=0，则AX<BX  
　　　　　　SF=1,则AX＞BX  
　　用逻辑表达式表示为：  
　　若OF∨-SF=0,则AX＞BX  
　　若OF∨-SF=1，则AX<BX  
　　(2) 格式：CMPXCHGDEST，REG  
　　功能：目的操作数减源操作数，  
　　　　如果DEST=SRC，则SRC→DEST。  
　　　　如果DEST≠SRC，则DEST→ACC(AL，AX，EAX)。  
　　源操作数允许为通用寄存器。目的操作数可以为通用寄存器，存储器操作数。  
　　CMPXCHG影响标志位为OF，SF，ZF，AF，PF，CF。  
　　(3) 格式：CMPXCHG8BMEM  
　　功能：EDX：EAX中值减存储器操作数。  
　　　　如果EDX：EAX=MEM64，则ECX：EBX→MEM64。  
　　　　如果EDX：EAX≠MEM64，则MEM64→EDX：EAX。  
　　该指令为64位比较交换指令，影响ZF标志位。  
　　**例 3．30**　　CMPXCHG8BQWORDPTR[EBX]  
　　五、交换相加指令  
　　格式：XADDDEST，REG  
　　功能：目的操作数加源操作数，结果送目的操作数。原目的操作数内容送源操作数。源操作数允许为通用寄存器。目的操作数允许为通用寄存器、存储器操作数。  
　　XADD指令影响标志位为OF，SF，ZF，AF，PF，CF。  
　　六、求补指令  
　　格式：NEGDEST  
　　功能：对目的操作数求补，用零减去目的操作数，结果送目的操作数。目的操作数为通用寄存器、存储器操作数。  
　　NEG指令影响标志位为OF，SF，ZF，AF，PF，CF。  
　　七、乘法指令  
　　(1) 格式：MULSRC  
　　　　　　　IMULSRC  
　　功能：MUL为无符号数乘法指令，IMUL为带符号数乘法指令。源操作数为通用寄存器或存储器操作数。目的操作数缺省存放在ACC(AL，AX，EAX)中，乘积存AX，DX：AX，EDX：EAX中。  
　　字节乘：ALSRC→AX  
　　字乘：AXSRC→DX∶AX  
　　双字乘：EAXSRC→EDX∶EAX  
　　MUL，IMUL指令执行后，CF=OF=0，表示乘积高位无有效数据；CF=OF=1表示乘积高位含有效数据，对其它标志位无定义。

**例 3．31**  
　　MUL BL；字节乘  
　　MUL WORD PTR [SI]；字乘  
　　IMUL BYTE PTR [DI]；字节乘  
　　IMUL DWORD PTR [ECX]；双字乘  
　　如果使用IMUL指令，积采用补码形式表示。  
　　(2) 格式：IMULDEST，SRC  
　　功能：将目的操作数乘以源操作数，结果送目的操作数。目的操作数为16位或32位通用寄存器或存储器操作数。源操作数为16位或32位通用寄存器、存储器或立即数。  
　　源操作数和目的操作数数据类型要求一致。乘积仅取和目的操作数相同的位数，高位部分将被舍去，并且CF=OF=1。其它标志位无定义。  
　　(3) 格式：IMUL DEST，SRC1，SRC2  
　　功能：将源操作数SRC1与源操作数SRC2相乘，结果送目的操作数。目的操作数DEST为16位或32位，允许为通用寄存器。源操作数SRC1为16位或32位通用寄存器或存储器操作数。源操作数SRC2允许为立即数。  
　　**例 3．32**IMULEAX，[EBX]，12H  
　　要求目的操作数和源操作数SRC1类型相同，当乘积超出目的操作数部分，将被舍去，并且使CF=OF=1，在使用这类指令时，需在IMUL指令后加一条判断溢出的指令，溢出时转错误处理执行程序。  
　　八、除法指令  
　　格式：DIV SRC  
　　　　　IDIV SRC  
　　功能：DIV为无符号数除法，IDIV为带符号数除法。源操作数作为除数，为通用寄存器或存储器操作数。被除数缺省在目的操作数AX，DX：AX，EDX：EAX中。  
　　字节除法：AX/SRC商→AL，余数→AH  
　　字除法：DX·AX/SRC商→AX，余数→DX  
　　双字除法：EDX·EAX/SRC商→EAX，余数→EDX  
　　由于被除数必须是除数的双倍字长，一般应使用扩展指令进行高位扩展。当进行无符号数除法时，被除数高位按0扩展为双倍除数字长。当进行有符号数除法时，被除数以补码表示。可使用扩展指令CBW，CWD，CWDE，CDQ进行高位扩展。例如：  
　　MOV AX，BLOCK  
　　CWD；被除数高位扩展  
　　MOV BX，1000H  
　　IDIV BX  
　　对于带符号除法，其商和余数均采用补码形式表示，余数与被除数同符号。当除数为零或商超过了规定数据类型所能表示的范围时，将会出现溢出现象，产生一个中断类型码为“0”的中断。执行除法指令后标志位无定义。  
　　九、BCD算术运算  
　　十进制数在机器中采用BCD码表示，以压缩格式存放，即一个字节存储2位BCD码，BCD加减法是在二进制加减运算的基础上，对其二进制结果进行调整，将结果调整成BCD码表示形式。

　　(1) 格式：DAA  
　　功能：将存放在AL中的二进制和数，调整为压缩格式的BCD码表示形式。  
　　调整方法：若AL中低4位大于9或标志AF=1(表示低4位向高4位有进位)，则  
　　　　　　　　　AL+6→AL,1→AF，  
若AL中高4位大于9，或标志CF=1，(表示高4位有进位)，则  
　　　　　　　　　AL+60H→AL,1→CF，  
　　DAA指令一般紧跟在ADD或ADC指令之后使用，影响标志位为SF，ZF，AF，PF，CF。OF无定义。  
**例 3．33**  
　　ADD AL，BL  
　　DAA  
　　(2) 格式：DAS  
　　功能：将存放在AL中的二进制差数，调整为压缩的BCD码表示形式。  
　　调整方法：若AL中低4位大于9或标志AF=1(表示低4位向高位借位)，则  
　　　　　　　　　AL-6→AL,1→AF  
若AL中高4位大于9或标志CF=1(表示高4位向高位借位)，则  
　　　　　　　　　AL-60H→AL,1→CF  
　　DAS指令一般紧跟在SUB或SBB指令之后使用，影响标志位为SF，ZF，AF，PF，CF。OF无定义。  
**例 3．34**  
　　SUB AL，BL  
　　DAS  
　　十、ASCII算术运算  
　　数字0～9的ASCII码为30H～39H，机器采用一个字节存放一位ASCII码，对于ASCII码的算术运算是在二进制运算基础上进行调整。调整指令有加、减、乘、除四种调整指令。  
　　(1) 格式：AAA  
　　功能：将存放在AL中的二进制和数，调整为ASCII码表示的结果。  
　　调整方法：若AL中低4位小于或等于9，仅AL中高4位清0，AF→CF。若AL中低4位大于9或标志AF=1(进位)，则AL+6→AL,AH+1→AH,1→AF,AF→CF,AL中高4位清0。  
　　AAA指令一般紧跟在ADD或ADC指令之后使用，影响标志位为AF，CF。其它标志位无定义。  
　　**例 3．35**  
　　MOV AX，0036H  
　　ADD，AL，35H  
　　AAA；AX=0101H  
　　(2) 格式：AAS  
　　功能：将存放在AL中的二进制差数，调整为ASCII码表示形式  
　　调整方法：若AL中低4位小于等于9，仅AL中高4位清0，AF→CF。若AL中低4位大于9或标志AF=1，则AL-6→AL,AH-1→AH，1→AF,AF→CF，AL中高4位清0。  
　　AAS指令一般紧跟在SUB，SBB指令之后使用，影响标志位为AF，CF。其它标志位无定义。  
　　**例 3．36**  
　　MOV AX，0132H  
　　SUB AL，35H  
　　AAS；AX=0007H  
　　(3) 格式：AAM  
　　功能：将存放在AL中的二进制积数，调整为ASCII码表示形式。  
　　调整方法：AL/10商→AH，余数→AL  
　　AAM指令一般紧跟在MUL指令之后使用，影响标志位为SF，ZF，PF。其它标志位无定义。  
**例 3．37**  
　　MOV AL，07H  
　　MOV BL，09H  
　　MUL BL；AX=003FH  
　　AAM;AX=0603H  
　　(4) 格式：AAD  
　　功能：将AX中两位非压缩BCD码(一个字节存放一位BCD码)，转换为二进制数的表示形式。  
　　调整方法：AH10+AL→AL0→AH  
　　AAD指令用于二进制除法DIV操作之前，影响的标志位为SF，ZF，PF。其它标志位无定义。  
　　**例 3．38**  
　　MOV AX，0605H  
　　MOV BL，09H  
　　AAD；AX=0041H  
　　DIV BL；AX=0207H  
　　使用该类指令应注意，加法、减法和乘法调整指令都是紧跟在算术运算指令之后，将二进制的运算结果调整为非压缩BCD码表示形式，而除法调整指令必须放在除法指令之前进行，以避免除法出现错误的结果。  
　　使用算术运算类指令应注意：  
　　·如果没有特别规定，参与运算的两个操作数数据类型必须一致，且只允许一个为存储器操作数；  
　　·如果参与运算的操作数只有一个，且为存储器操作数，必须使用PTR伪指令说明数据类型；  
　　·操作数不允许为段寄存器。  
　　·目的操作数不允许为立即数；  
　　·如果是存储器寻址，则存储器各种寻址方式均可使用。  
**3．3．3逻辑运算指令**  
　　一、逻辑指令  
　　1逻辑与指令  
　　格式：AND DEST，SRC  
　　功能：目的操作数和源操作数按位进行逻辑与运算，结果存目的操作数中。源操作数可以是通用寄存器、存储器或立即数。目的操作数可以是通用寄存器或存储器操作数。  
　　**例 3．39**  
　　AND AL，BL  
　　AND EBX，ECX  
　　AND [DI]，1101H  
　　AND指令常用于将操作数中某位清0(称屏蔽)，只须将要清0的位与0，其它不变的位与1即可。  
　　**例 3．40**AND AL，0FH；将AL中高4位清0，低4位保持不变。  
　　AND指令影响标志位为SF，ZF，PF，并且使OF=CF=0。  
　　2逻辑或指令  
　　格式：**OR** DEST，SRC  
　　功能：目的操作数和源操作数按位进行逻辑或运算，结果存目的操作数中。源操作数可以是通用寄存器、存储器或立即数。目的操作数可以是通用寄存器或存储器操作数。  
　**例 3．41**  
　　**OR** AX，BX  
　　**OR** ECX，[EAX]  
　　**OR**指令常用于将操作数中某位置1，只须将要置1的位或1，其它不改变的位或0即可。  
　　**例 3．42　　OR** AL，80H；将AL中最高位置1。  
　　**OR**指令影响标志位为SF，ZF，PF。并且使OF=CF=0。  
　　3逻辑异或指令  
　　格式：XOR DEST，SRC  
　　功能：目的操作数和源操作数按位进行逻辑异或运算，结果送目的操作数。源操作数可以是通用寄存器、存储器或立即数。目的操作数可以是通用寄存器或存储器操作数。

**例 3．43**  
　　XOR AX，BX  
　　XOR [BX]，1010H  
　　XOR指令常用于将操作数中某些位取反，只须将要取反的位异或1，其它不改变的位异或0即可。  
　　**例 3．44**　　XOR AL，OFH；将AL中低4位取反，高4位保持不变。  
　　XOR指令影响标志位为SF，ZF，PF，并且使OF=CF=0。

　　4逻辑非指令  
　　格式：NOT DEST  
　　功能：对目的操作数按位取反，结果回送目的操作数。目的操作数可以为通用寄存器或存储器。  
　　**例 3．45**  
　　NOT EAX  
　　NOT BYTE PTR [BX]  
　　NOT指令对标志位无影响。  
　　5[测试](http://lib.csdn.net/base/softwaretest)指令  
　　格式：TEST DEST，SRC  
　　功能：目的操作数和源操作数按位进行逻辑与操作，结果不回送目的操作数。源操作数可以为通用寄存器、存储器或立即数。目的操作数可以为通用寄存器或存储器操作数。  
　　**例 3．46**  
　　TEST DWORD PTR [BX]，80000000H  
　　TEST AL，CL  
　　TEST指令常用于测试操作数中某位是否为1，而且不会影响目的操作数。如果测试某位的状态，对某位进行逻辑与1的运算，其它位逻辑与0，然后判断标志位。运算结果为0，ZF=1，表示被测试位为0；否则ZF=0，表示被测试位为1。  
**例 3.47**　　TEST AL，80H；测试AL中最高位  
　　　　　　　　JNZ NEXT；如果最高位为1，转到标志NEXT处。  
　　TEST指令影响标志位为SF，ZF，PF，并且使OF=CF=0。  
　　二、移位指令  
　　移位指令对操作数按某种方式左移或右移，移位位数可以由立即数直接给出，或由CL间接给出。移位指令分一般移位指令和循环移位指令。

1一般移位指令  
　　(1) 算术/逻辑左移指令。  
　　格式：SAL DEST，OPRD  
　　　　　SHL DEST，OPRD  
　　功能：按照操作数OPRD规定的移位位数，对目的操作数进行左移操作，最高位移入CF中。每移动一位，右边补一位0。如图312(a)所示。目的操作数可以为通用寄存器或存储器操作数。  
　　SAL，SHL指令影响标志位OF，SF，ZF，PF，CF。

**例 3.48**  
　　SHL BYTE PTR [DI]，2  
　　SAL BX，CL  
　　(2) 算术右移指令。  
　　格式：SAR DEST，OPRD  
　　功能：按照操作数OPRD规定的移位次数，对目的操作数进行右移操作，最低位移至CF中，最高位(即符号位)保持不变。如图312(b)所示。目的操作数可以为通用寄存器或存储器操作数。  
　　SAR指令影响标志位OF，SF，ZF，PF，CF。  
　**例 3．49**  
　　SAR AL，5  
　　SAR WORD PTR /[ECX/],CL  
　　(3) 逻辑右移指令。  
　　格式：SHR DEST，SRC  
　　功能：按照操作数OPRD规定的移位位数，对目的操作数进行右移操作，最低位移至CF中。每移动一位，左边补一位0。如图312(c)所示，目的操作数可以为通用寄存器或存储器操作数。  
　　SHR指令影响标志位OF，SF，ZF，PF，CF。  
**例 3.50**  
　　SHR BYTE PTR [SI],3  
　　SHR EDX,CL  
　　算术/逻辑左移，只要结果未超出目的操作数所能表达的范围，每左移一次相当于原数乘2。算术右移只要无溢出，每右移一次相当于原数除以2。  
　　2循环移位指令  
　　格式：ROL DEST，OPRD  
　　　　　ROR DEST，OPRD  
　　　　　RCL DEST，OPRD  
　　　　　RCR DEST，OPRD  
　　功能：循环左移指令ROL，见图313(a)所示，目的操作数左移，每移位一次，其最高位移入最低位，同时最高位也移入进位标志CF。循环右移指令 ROR见图313(b)所示，目的操作数右移，每移位一次，其最低位移入最高位，同时最低位也移入进位标志CF。  
　　带进位循环左移指令RCL，见图313(c)所示，目的操作数左移，每移动一次，其最高位移入进位标志CF，CF移入最低位。带进位循环右移指令RCR，见图313(d)所示，目的操作数右移，每移动一次，其最低位移入进位标

目的操作数可以为通用寄存器或存储器操作数。循环移位指令影响标志位CF，OF。其它标志位无定义。  
　　**例 3.51**  
　　ROL AL，CL  
　　ROR BX，5  
　　RCL ECX，3  
　　RCR BYTE PTR [SI]，CL  
**例 3.52**　　将一个2位数压缩的BCD码转换成二进制数。  
　　　·MODEL SMALL  
　　　·DATA  
　　BCD DB 01011001B  
　　BIN DB?  
　　　　CODE  
　　　·START UP  
　　　　MOV AL，BCD  
　　　　MOV BL，AL  
　　　　AND BL，0FH  
　　　　AND AL，0F0H  
　　　　MOV CL，4  
　　　　ROR AL，CL  
　　　　MOV BH，0AH  
　　　　MUL BH  
　　　　ADD AL，BL  
　　　　MOV BIN，AL  
　　　·EXIT  
　　　　END  
　　3双精度移位指令  
　　格式：SHLD DEST，SRC，OPRD  
　　　　　SHRD DEST，SRC，OPRD  
　　功能：对于由目的操作数DEST和源操作数SRC构成的双精度数，按照操作数OPRD给出的移位位数，进行移位。SHLD是对目的操作数进行左移，如 图314(a)所示，SHRD是对目的操作数进行右移，如图314(b)所示。先移出位送标志位CF，另一端空出位由SRC移入DEST中，而SRC 内容保持不变。目的操作数可以是16位或32位通用寄存器或存储器操作数。源操作数SRC允许为16位或32位通用寄存器。操作数OPRD可以为立即数或 CL。目的操作数和源操作数SRC数据类型必须一致。

SHLD，SHRD指令常用于位串的快速移位、嵌入和删除等操作，影响标志位为SF，ZF，PF，CF，其它标志位无定义。  
　　三、位操作指令  
位操作指令包括位测试和位扫描指令，可以直接对一个二进制位进行测试，设置和扫描。  
　　1位测试和设置指令  
　　格式：BT DEST，SRC  
　　　　　BTC DEST，SRC  
　　　　　BTR DEST，SRC  
　　　　　BTS DEST，SRC  
　　功能：按照源操作指定的位号，测试目的操作数，当指令执行时，被测试位的状态被复制到进位标志CF。  
　　BT将SRC指定的DEST中一位的数值复制到CF。BTC将SRC指定的DEST中一位的数值复制到CF，且将DEST中该位取反。BTR将SRC 指定的DEST中一位的数值复制到CF，且将DEST中该位复位。BTS将SRC指定的DEST中一位的数值复制到CF，且将DEST中该位置位。  
　　目的操作数为16位或32位通用寄存器或存储器，源操作数为16位或32位通用寄存器，以及8位立即数，当源操作数为通用寄存器时，必须同目的操作数类型一致。源操作数SRC以两种方式给出目的操作数的位号，即  
　　· SRC为8位立即数，以二进制形式直接给出要操作的位号；  
　　· SRC为通用寄存器，如果DEST为通用寄存器，则SRC中二进制值直接给出要操作的位号。如果DEST为存储器操作数，通用寄存器SRC为带符号整数， SRC的值除以DEST的长度所得到的商作为DEST的相对偏移量，余数直接作为要操作的位号。DEST的有效地址为DEST给出的偏移地址和DEST相 对偏移量之和。  
　　BT，BTC，BTR，BTS指令影响CF标志位，其它标志位无定义。  
　**例 3.53**  
　　MOV AX，1234H  
　　MOV ECX，5  
　　BT AX，CX　　　　　　　；CF=1AX=1234H  
　　BTC AX，5　　　　　　　；CF=1；AX=1214H  
　　BTS AX，CX；　　　　　 ；CF=0AX=1234H  
　　BTR EAX，ECX　　　　　 ；CF=1EAX=00001214H  
  
**例 3.54**

　　　　　　·MODEL SMALL  
　　　　　　·586  
　　　　　　·DATA  
　　　DATA1　DW 1234H，5678H  
　　　　　　·CODE  
　　　　　　·START UP  
　　　　　　　BTC DATA1，3；CF=0(DATA1)=123CH  
　　　　　　　MOV CX，20  
　　　　　　　BTR DATA1，CX；CF=1[DATA+2]=5668H  
　　　　　　·EXIT  
　　　　　　　END  
　　2位扫描指令  
　　格式：BSFDEST，SRC  
　　　　　BSRDEST，SRC  
　　功能：BSF从低位开始扫描源操作数，若所有位都是0，则ZF=0，否则ZF=1。并且将第一个出现1的位号存入目的操作数。BSR从高位开始扫描源操作数，若所有位都是0，则ZF=0，否则ZF=1。并且将第一个出现1的位号存入目的操作数。  
　　源操作数可以为16位32位通用寄存器或存储器。目的操作数为16位或32位通用寄存器。源操作数和目的操作数类型必须一致。  
　　BSF，BSR指令影响ZF标志位，其它标志位无定义。  
**例 3.55**  
　　MOV EBX，0F333EE00H  
　　BSR EAX，EBX；ZF=1EAX=0000001FH=31  
　　BSF EDX，EBX；ZF=1EDX=00000009H  
　　3进位标志指令  
　　(1) 格式：CLC。功能：清除进位标志。  
　　(2) 格式：STC。功能：设置进位标志。  
　　(3) 格式：CMC。功能：进位标志取反。  
　　4条件设置字节指令  
　　条件设置指令用于根据条件设置某一状态字节或标志字节，见表33。  
　　格式：SETcondDEST  
　　功能：测试条件(cond)若为真，则将目的操作数置01H，否则置00H。目的操作数允许为8位通用寄存器或8位存储器操作数。  
　　条件cond与条件转移指令中的条件相同，共分三类。  
　　(1) 以标志位状态为条件可以测试的标志位为ZF，SF，OF，CF，PF。  
　　(2) 以两个无符号数比较为条件条件为高于、高于等于、低于、低于等于。  
　　(3) 以两个带符号数比较为条件条件为大于、大于等于、小于、小于等于。  
　　SET指令不影响标志位。  
　　使用逻辑运算类指令应注意：  
　　· 如果没有特别规定，参与运算的两个操作数类型必须一致，且只允许一个为存储器操作数；  
　　· 如果参与运算的操作数只有一个，且为存储器操作数，必须使用PTR伪指令说明其数据类型；　  
　　· 操作数不允许为段寄存器；  
　　· 目的操作数不允许为立即数；  
　　· 如果是存储器寻址，则前面介绍的各种存储器寻址方式均可使用。

**3.3.4控制转移类指令**  
　　计算机执行程序一般是顺序地逐条执行指令。但经常须要根据不同条件做不同的处理，有时需要跳过几条指令，有时需要重复执行某段程序，或者转移到另一个程序段去执行。用于控制程序流程的指令包括转移、循环、过程调用和中断调用。  
　　一、转移指令  
　　1无条件转移指令  
　　格式：JMP TARGET  
　　功能：使程序无条件地转移到指令规定的目的地址TARGET去执行指令。转移分为短转移、段内转移(近程转移)和段间转移(远程转移)。  
　　(1) 段内直接转移：  
　　格式：JMP SHORT TARGET；短转移  
　　JMP NEAR PTR TARGET；近程转移  
　　功能：采用相对寻址将当前IP值(即JMP指令下一条指令的地址)与JMP指令中给出的偏移量之和送IP中。段内短转移(SHORT)指令偏移量为8 位，允许转移偏移值的范围为-128~+127。段内近程转移(NEAR)指令在16位指令模式下，偏移量为16位，允许转移偏移值范围为-215~+ 215-1。在32位指令模式下，偏移值范围为-231~+231-1。

**例 3.56**  
　　　　　JMP NEXT

　　　　　　  
　　NEXT：MOV AL，BL  
　　本例为无条件转移到本段内，标号为NEXT的地址去执行指令，**汇编**程序可以确定目的地址与JMP指令的距离。  
　　(2) 段内间接转移：  
　　格式：JMP REG  
　　JMP NEAR PTR [REG]  
　　功能：段内间接转移，其中JMP REG指令地址在通用寄存器中，将其内容直接送IP实现程序转移。JMP NEAR PTR [REG]指令地址在存储器中，默认段寄存器根据参与寻址的通用寄存器来确定，将指定存储单元的字取出直接送IP实现程序转移。在16位指令模式，转移偏 移值范围为。在32位指令模式，转移偏移值范围为。

**例 3.57**　设DS=1000HEBX=00002000H。  
　　JMP BX　　　　　　　　　　；将2000H送IP  
　　JMP NEAR PTR [BX]　　　　 ；将地址1000∶2000单元存放的一个字送IP  
　　JMP NEAR PTR [EBX]　　　　；将段选择符为1000H，偏移地址为00002000H单元存放的双字送EIP。  
　　(3) 段间直接转移：  
　　格式：JMP FAR PTR TARGET  
　　功能：段间直接转移，FAR PTR说明标号TARGET具有远程属性。将指令中由TARGET指定的段值送CS，偏移地址送IP。  
**例 3.58**　JMP FAR PTR NEXT。  
　　在16位指令模式下，段基地送CS，偏移地址为16位，转移偏移值范围；在32位指令模式下，代码段选择符送CS，偏移地址为32位，转移偏移值范围为。  
　　(4) 段间间接转移:  
　　格式：JMP FAR PTR [Reg]  
　　功能：段间间接转移，由FAR PTR [Reg]指定的存储器操作数作为转移地址。  
在16位指令模式下，存储器操作数为32位，包括16位段基址和16位偏移地址。  
**例 3.59**  
　　JMP FAR PTR [BX]　　　　　　；数据段双字存储单元低字内容送IP  
　　　　　　　　　　　　　　　　；数据段双字存储单元高字内容送CS  
　　在32位指令模式下，存储器操作数包括16位选择符。  
**例 3.60**JMP FAR PTR [EAX]  
　　指令中包含指向目标地址指针的门描述符或TSS描述符的指针，其所指的存储器操作数中仅选择符部分有效，指示调用门、任务门或TSS描述符起作用，而偏移部分不起作用。

2条件转移指令  
　　该类指令是根据上一条指令对标志寄存器中标志位的影响来决定程序执行的流程，若满足指令规定的条件，则程序转移；否则程序顺序执行。  
条件转移指令的转移范围为段内短转移或段内近程转移，不允许段间转移。段内短转移(short)的转移偏移值范围为-128~+127。段内近程转移，在16位指令模式下转移偏移值范围为，在32位指令模式下转移偏移值范围为。  
　　条件转移指令包括四类：单标志位条件转移；无符号数比较条件转移；带符号数比较条件转移；测试CX条件转移。  
　　格式：Jcc TARGET  
　　功能：若测试条件‘CC’为真，则转移到目标地址TARGET处执行程序。否则顺序执行。  
　　(1) 单标志位条件转移指令，见表34。  
　　**例 3.61**　JZ NEXT；若标志ZF=1则转移到标号NEXT处执行。  
　　(2) 无符号数比较条件转移，见表35。  
**例 3.62**JA NEXT；无符号数A与B比较，若A>B则转移到标号NEXT处执行程序

假设x为某值且存放在寄存器AL中，试编程将求出的函数值f(x)存放在AH中。  
　　　　·MODEL TINY  
　　　　·CODE  
　　　　·STARTUP  
　　　　　CMPAL，0  
　　　　　JGE BIG  
　　　　　MOV AL，0FFH  
　　　　　JMP DONE  
　　BIG： JE DONE  
　　　　　MOV AL，1  
　　DONE：MOV AH，AL  
　　　·EXIT  
　　　　END  
**例 3.66**编程实现把BX寄存器内的二进制数用十六进制数的形式在屏幕上显示出来。  
　　　　　·MODEL TINY  
　　　　　·CODE  
　　　　　·STARTUP  
　　　　　　MOV CH，4  
　　AGAIN： MOV CL，4  
　　　　　　ROL BX，CL  
　　　　　　MOV AL，BL  
　　　　　　ANDAL，0FH  
　　　　　　**OR** AL，30H  
　　　　　　CMP AL，3AH  
　　　　　　JB NEXT  
　　　　　　ADD AL，07H  
　　 NEXT： MOV DL，AL；DL←要显示的ASCII码  
　　　　　　MOV AH，2；显示  
　　　　　　INT 21H  
　　　　　　DECCH  
　　　　　　JNZ AGAIN  
　　　　　·EXIT  
　　　　　　END  
　　二、循环控制指令  
　　这类指令用(E)CX计数器中的内容控制循环次数，先将循环计数值存放在(E)CX中，每循环一次(E)CX内容减1，直到(E)CX为0时循环结束。  
　　格式：LOOPcc TARGET  
　　功能：将(E)CX内容减1，不影响标志位，若(E)CX不等于0，且测试条件‘CC’成立，则转移到目标地址TARGET处执行程序。转移范围为-128~+127。如表38所示。

**例 3.67**　计算  
　　　　　　·MODEL TINY  
　　　　　　·CODE  
　　　　　　·STARTUP  
　　　　　　　XOR EAX，EAX  
　　　　　　　MOV EDX，1  
　　　　　　　MOV ECX，1000  
　　　SUM：　 ADD EAX，EDX  
　　　　　　　INC EDX  
　　　　　　　LOOPD SUM  
　　　　　　·EXIT  
　　　　　　　END  
**例 3.68**　找出以ARRAY为首地址的100个字数组中的第一个非0项，送AX寄存器中。  
　　　　　·MODELSMALL  
　　　　　·DATA  
　　　　　　ARRAYDW 0，0，0，0，1010H，…；(100个字)  
　　　　　·CODE  
　　　　　·STARTUP  
　　　　　　MOV CX，64H  
　　　　　　LEA BX，ARRAY  
　　　　　　MOV SI，0FFFEH  
　　ZERO：　INC SI  
　　　　　　INC SI  
　　　　　　CMP WORD PTR [BX+SI],0  
　　　　　　LOOPZ ZERO  
　　　　　　MOV AX，[BX+SI]  
　　　　　·EXIT  
　　　　　　END  
　　关于过程调用和返回指令将在子程序一节中介绍。  
**3.3.5串操作指令**  
　　80x86提供处理字符串的操作。串指连续存放在存储器中的一些数据字节、字或双字。串操作允许程序对连续存放大的数据块进行操作。  
　　串操作通常以DS：(E)SI来寻址源串，以ES：(E)DI来寻址目的串，对于源串允许段超越。(E)SI或(E)DI这两个地址指针在每次串操作 后，都自动进行修改，以指向串中下一个串元素。地址指针修改是增量还是减量由方向标志来规定。当DF=0，(E)SI及(E)DI的修改为增量；当DF= 1，(E)SI及(E)DI的修改为减量。根据串元素类型不同，地址指针增减量也不同，在串操作时，字节类型SI，DI加、减1；字类型SI，DI加、减 2；双字类型ESI，EDI加、减4。如果需要连续进行串操作，通常加重复前缀。重复前缀可以和任何串操作指令组合，形成复合指令，见表39。  
　　一、重复前缀指令

二、方向标志指令  
　　格式：CLD/STD  
　　功能：CLD为清除方向标志，即将DF置‘0’。STD为设置方向标志，即将DF置‘1’。  
　　三、串传送指令  
　　基本格式：[REP]MOVS DESTS, SRCS  
　　[REP] MOVSB/MOVSW/MOVSD  
　　功能：将DS：(E)SI规定的源串元素复制到ES：(E)DI规定的目的串单元中，见表310。

该指令对标志位无影响。  
　　如果加重复前缀REP，则可以实现连续存放的数据块的传送，直到(E)CX=0为止。  
　　在16位指令模式下，使用SI，DI，CX寄存器；在32位指令模式下，使用ESI，EDI，ECX寄存器。  
　　例 3.69  
　　　·MODEL SMALL  
　　　·DATA  
　SRC 　DB 1,2,3,…(100个字节)  
　DEST　DB 100DUP(?)  
　　　·CODE  
　　　·STARTUP  
　　　　MOV AX，＠DATA  
　　　　MOV ES，AX  
　　　　MOV CX，100  
　　　　LEA SI，SRC  
　　　　LEA DI，DEST  
　　　　CLD  
　　　　REP MOVSB  
　　　·EXIT  
　　　　END  
　　该程序将起始地址为SRC的100个字节内容传送到起始地址为DEST的存储单元。  
　　四、串比较指令  
　　基本格式：[REPE/Z] [REPNZ/NE] CMPS DESTS, SRCS  
　　　　　　　[REPE/Z] [REPNZ/NE] CMPSB/CMPSW/CMPSD  
　　功能：由DS：(E)SI规定的源串元素减去ES：(E)DI指出的目的串元素，结果不回送，仅影响标志位CF，AF，PF，OF，ZF，SF。当源 串元素与目的串元素值相同时，ZF=1；否则ZF=0。每执行一次串比较指令，根据DF的值和串元素数据类型自动修改(E)SI和(E)DI。  
　　在串比较指令前加重复前缀REPE/Z，则表示重复比较两个字符串，若两个字符串的元素相同则比较到(E)CX=0为止，否则结束比较。在串比较指令 前加重复前缀REPNE/NZ，则表示若两个字符串元素不相同时，重复比较直到(E)CX=0为止，否则结束比较。  
　　例 3.70　编程实现两个串元素比较，如相同则将全“1”送SUT单元，否则全“0”送SUT单元。  
　　　　　　·MODEL SMALL  
　　　　　　·DATA  
　　　　DEST　DB ‘A B C D E F G H’  
　　　　SRC　 DB ‘A B C E F F F E’  
　　　　SUT　 DB?  
　　　　　　·CODE  
　　　　　　·STARTUP  
　　　　　　　MOV AX，＠DATA  
　　　　　　　MOV ES，AX  
　　　　　　　MOV CX，8  
　　　　　　　LEA SI，DEST  
　　　　　　　LEA DI，SRC  
　　　　　　　CLD  
　　　　　　　REPE CMPSB  
　　　　　　　JZ EQUL；ZF=1；CX=0  
　　　　　　　MOV BH，0；CX≠0，ZF=0  
　　　　　　　JMP DONE  
　　　EQUL：　MOV BH，0FFH  
　　　DONE：　MOV SUT，BH  
　　　　　　·EXIT  
　　　　　　　END  
　　五、串扫描指令  
　　格式①： [REPE/Z] [REPNE/NZ] SCAS DESTS  
　　格式②： [REPE/Z] [REPNE/NZ] SCASB/SCASW/SCASD  
　　功能：由AL，AX或EAX的内容减去ES：(E)DI规定的目的串元素，结果不回送，仅影响标志位CF，AF，PF，SF，OF，ZF。当AL， AX或EAX的值与目的串元素值相同时，ZF=1；否则ZF=0。每执行一次串扫描指令，根据DF的值和串元素数据类型自动修改(E)DI。  
　　在串扫描指令前加重复前缀REPE/Z，则表示目的串元素值和累加器值相同时重复扫描，直到CX/ECX=0为止，否则结束扫描。若加重复前缀 REPNE/NZ，则表示当目的串元素值与累加器值不相等时，重复扫描直到CX/ECX=0时为止，否则结束扫描。  
该指令影响标志位为CF，AF，PF，SF，OF，ZF。  
　　例 3.71　在内存DEST开始的6个单元寻找字符‘C’，如找到将字符‘C’的地址送ADDR单元，否则0送ADDR单元。  
　　　　·MODEL SMALL  
　　　　·DATA  
　　DEST　DB ‘A B C D E F’  
　　ADDR　DW?；存“C”的地址，所以设置为字类型  
　　　　·CODE  
　　　　·STARTUP  
　　　　　MOV AX,＠DATA  
　　　　　MOV ES，AX  
　　　　　MOV CX，6  
　　　　　LEA DI，DEST  
　　　　　MOV AL，‘C’  
　　　　　CLD  
　　　　　REPNE SCASB  
　　　　　JZ EQUL  
　　　　　MOV DI，0  
　　　　　JMP DONE  
　EQUL：　DEC DI  
　DONE：　MOV ADDR，DI  
　　　　·EXIT  
　　　　　END  
　　六、 串装入指令  
　　格式：LODS SRCS  
　　　　　LODSB/LODSW/LODSD  
　　功能：将DS：SI/ESI所指的源串元素装入累加器(AL，AX，EAX)中，每装入一次都按照DF值以及串元素类型自动修改地址指针SI/ESI，该指令一般不须加重复前缀，并且不影响标志位。  
　　七、 串存储指令  
　　格式：[REP] STOS DESTS  
　　　　　[REP] STOSB/STOSW/STOSD  
　　功能：将累加器/[AL，AX，EAX/]中值存入ES：DI/EDI所指的目的串存储单元中，每传递一次，都按DF值以及串元素类型自动修改地址指 针DI/EDI。若加重复前缀REP，则表示将累加器的值连续送目的串存储单元，直到CX/ECX=0时为止。  
该指令不影响标志位。  
　　3.3.6输入/输出指令  
　　一、 输入指令  
　　格式：IN DEST, SRC  
　　功能：根据源操作数SRC给出的端口地址，将操作数从指定端口传送到目的操作数DEST处，其中DEST为AL，AX或EAX，端口地址SRC可以直接形式给出8位端口地址，或由DX寄存器以间接形式给出。  
　　例 3.72  
　　IN AL，10H  
　　IN AX，20H  
　　IN EAX，30H  
　　IN AL，DX  
　　IN AX，DX  
　　IN EAX，DX  
　　二、 输出指令  
　　格式OUT DEST, SRC  
　　功能：将源操作数SRC送到目的操作数DEST所指定的端口。其中源操作数SRC为AL，AX或EAX，目的操作数可以8位端口地址方式直接给出或以DX寄存器间接方式给出。  
使用输入、输出指令应注意：  
　　· 直接寻址方式端口地址为8位，共有0～255个端口地址；  
　　· 间接寻址方式，只能用DX作为地址寄存器，寻址范围为64K字节；  
　　· 每个I/O地址对应的端口的数据长度为8位，传送8位数据占用一个端口地址，传送16位数据占用2个端口地址，传送32位数据占用4个端口地址。  
　　三、 串输入指令  
　　格式：[REP] INS DESTS, DX  
　　[REP] INSB/INSW/INSD  
　　功能：根据DX给出的端口地址，从外设读入数据送入以ES：DI/EDI为地址的目的串存储单元中，每输入一次，均根据DF的值和串元素类型自动修改 DI/EDI的值。若加重复前缀REP，则表示连续从外设输入串元素存入目的串存储单元中，直到CX/ECX=0为止。  
　　例 3.73　从端口地址为1000H处取数存入内存BLOCK单元。  
　　　　　·MODEL SMALL  
　　　　　·DATA  
　　　　　　BLOCKDB?  
　　　　　·CODE  
　　　　　·STARTUP  
　　　　　　MOV AX，＠DATA  
　　　　　　MOV ES，AX  
　　　　　　CLD  
　　　　　　LEA DI，BLOCK  
　　　　　　MOV DX，1000H  
　　　　　　INS BLOCK，DX  
　　　　　·EXIT  
　　　　　　END  
　　四、串输出指令  
　　格式：[REP] OUTS DX,SRCS  
　　　　　[REP] OUTSB/OUTSW/OUTSD  
　　功能：将DS：SI/ESI所指的源串元素，按照DX寄存器指定的端口地址送往外设，每输出一次，均根据DF的值和串元素类型自动修改SI/ESI的值，若加重复前缀REP，则表示连续向外设输出串元素，直到CX/ECX=0时为止。  
　　例 3.74　将内存BLOCK为首地址的100个字符送往端口地址为2000H的外设。  
　　　　　·MODEL SMALL  
　　　　　·DATA  
　　　　　　BLOCKDB ‘A，B，…’(100个字符)  
　　　　　·CODE  
　　　　　·STARTUP  
　　　　　　CLD  
　　　　　　LEA SI，BLOCK  
　　　　　　MOV CX，100  
　　　　　　MOV DX，2000H  
　　　　　　REP OUTSB  
　　　　　·EXIT  
　　　　　　END  
　　在使用带重复前缀的串输入输出指令时，必须考虑端口的数据准备或接收状态。  
　　所有输入输出指令均不影响标志位。  
　　3.3.7处理器控制  
　　一、 总线封锁前缀  
　　格式：LOCK指令  
　　功能：LOCK为指令前缀，可以使LOCK引脚变成逻辑0，在LOCK引脚有效期间，禁止外部总线上的其它处理器存取带有LOCK前缀指令的存储器操作数。  
　　可加LOCK前缀的指令：  
　　(1) ADD/SUB/ADC/SBB/OR/XOR/AND Mem, Reg/imm；  
　　(2) NOT/NEG/INC/NEC Mem；  
　　(3) XCHG Reg, Mem或XCHG Mem, Reg；  
　　(4) BT/BTS/BRT/BTC Mem, Reg/imm。  
　　Mem为存储器操作数，Reg为通用寄存器，imm为立即数。  
　　二、空操作  
　　格式：NOP  
　　功能：空操作，除使IP/EIP增1外，不做任何工作。该指令不影响标志位。  
　　三、处理器等待指令  
　　格式：WAIT  
　　功能：检查BUSY引脚状态，等待协处理器完成当前工作。  
　　四、处理器暂停指令  
　　格式：HLT  
　　功能：暂停程序的执行。当产生一个外部中断或非屏蔽中断时，才继续执行下一条指令。  
　　3.3.8中断指令与DOS功能调用  
　　一、中断指令  
　　在实模式下，中断矢量以4个字节存放在中断矢量表中，中断矢量表为1k字节(00000H~003FFH)，中断矢量表允许存放256个中断矢量，每 个中断矢量包含一个中断服务程序地址(段值和16位偏移地址)，中断矢量地址指针由中断类型码乘以4得到。  
　　在保护模式下，用中断描述符表代替中断矢量表，每个中断由8个字节的中断描述符来说明，中断描述符表允许256个中断描述符，每个中断描述符包含一个中断服务地址(段选择符、32位偏移地址、访问权限等)。中断描述符地址指针由中断类型码乘以8得到。

　　中断指令格式：INT n  
　　功能：产生中断类型码为n的软中断，该指令包含中断操作码和中断类型码两部分，中断类型码n为8位，取值范围为0～255(00H～FFH)。  
　　软中断执行过程：  
　　· 将标志寄存器FLAGS(或EFLAGS)压入堆栈；  
　　· 清除TF和IF标志位；  
　　· CS，IP/EIP压入堆栈；  
　　· 实模式下，n×4获取中断矢量表地址指针；保护模式下，n×8获取中断描述符表地址指针；  
　　· 根据地址指针，从中断矢量表或中断描述符表中取出中断服务程序地址送IP/EIP和CS中，控制程序转移去执行中断服务程序。  
　　中断返回指令格式：IRET/IRETD  
　　功能：该指令实现在中断服务程序结束后，返回到主程序中断断点处，继续执行主程序。  
　　中断返回执行过程：  
　　· IRET指令弹出堆栈中数据送IP，CS，FLAGS；  
　　· IRETD指令弹出堆栈中数据送EIP，CS，EFLAGS。  
　　其它中断类指令如表311所示。

二、DOS功能调用  
　　系统功能调用是MS—DOS为程序员编写**汇编**语言源程序提供的一组子程序，包括设备管理、文件管理和目录管理等。  
　　DOS规定使用软中断指令INT 21H作为进入各功能子程序的总入口，再为每个功能调用规定一个功能号，引用功能号即可进入相应的子程序入口。DOS系统功能调用的使用方法归纳如下：  
　　(1) 传送入口参数到指定的寄存器中；  
　　(2) 把要调用功能的功能号送入AH寄存器中；  
　　(3) 用INT 21H指令转入子程序入口；  
　　(4) 相应的子程序运行结束后，可以按照规定取得出口参数。  
　　常用系统功能调用简介。  
　　1键盘输入单字符  
　　这是1号系统功能调用，其调用格式为  
　　　MOV AH，1  
　　　INT 21H  
　　该功能调用无入口参数。其功能为系统等待键盘输入，如是Ctrol-Break键则退出；否则将键入字符的ASCII码送入AL寄存器中，并且通过显示器显示该字符。  
　　2键盘输入字符串  
　　这是0AH号系统功能调用，其功能为将键盘输入的字符串写入内存单元中。因此，首先在内存中定义一个缓冲区，缓冲区第一个字节存放规定字符串的最大字 节数，第二个字节由系统送入实际键入的字符数，从第三个字节开始用于存放键入的字符串，最后通过键入回车键来表示字符串的结束。如果实际键入的字符数未达 到最大规定数，其缓冲区的空余区间填0；如果实际键入数超过缓冲区的容量，则超出的字符自动丢失，而且响铃警告。注意，回车键值也存于缓冲区中。  
　　**例 3.75**　使用格式举例。  
　　　　　·MODEL SMALL  
　　　　　·DATA  
　　　BUF　 DB 20  
　　　　　　DB?  
　　　　　　DB 20 DUP(?)  
　　　　　·CODE  
　　　　　·STARTUP  
　　　　　　MOV DX，OFFSET BUF  
　　　　　　MOV AH，0AH  
　　　　　　INT 21H  
　　　　　·EXIT  
　　　　　　END  
　　该程序在BUF为首地址的缓冲区定义了20个字符串字节的缓冲区，并且将缓冲区首地址送入DX中，调用0AH号子程序，系统等待用户键入字符串，每键 入一个字符，其相应的ASCII码将被写入缓冲区中，直到键入回车键，由系统输入实际键入字符数，送入缓冲区第二个字节中。  
　　3输出单字符  
　　这是2号系统功能调用，其使用格式为：  
　　MOV DL，‘A’  
　　MOV AH，2  
　　INT 21H  
　　执行2号系统功能调用，将置入DL寄存器中的字符(以ASCII码形式表示)通过显示器显示出来(或从打印机输出)。  
　　4输出字符串  
　　这是9号系统功能调用，其功能是将指定的内存缓冲区中的字符串从显示器显示输出(或从打印机输出)，缓冲区中的字符串以字符‘＄’作为结束标志。  
　　**例 3.76**使用格式举例。  
　　　　·MODEL SMALL  
　　　　·DATA  
　　BUF　DB ‘Thank you $’  
　　　　·CODE  
　　　　·STARTUP  
　　　　　MOV DX，OFFSET BUF  
　　　　　MOV AH，9  
　　　　　INT 21H  
　　　　·EXIT  
　　　　　END  
　　5返回[操作系统](http://lib.csdn.net/base/operatingsystem)  
　　这是4CH号系统功能调用，使用格式为  
　　MOV AH，4CH  
　　INT 21H  
　　在用户程序结束处插入此调用，则返回到DOS操作系统，显示器显示系统提示符。  
习题与思考题  
1数据寻址方式有哪几种?  
216位指令模式下和32位指令模式下的存储器寻址方式各有哪几种寻址方式?并比较它们相似与不同之处。  
3程序地址寻址方式有哪几种?  
4什么是堆栈地址寻址方式?  
5指令编码格式是由哪几部分组成的?各部分的含义是什么?  
6 80x86的指令格式由哪几部分组成?  
7 80x86指令系统按其功能可分为几部分?  
8数据传送指令包括哪些类型?  
9堆栈的含义是什么?80x86所用的堆栈有什么特点?  
10堆栈操作指令有哪几种?  
11 XLAT指令在使用时有哪些规定?  
12符号扩展指令在什么情况下使用?  
13十进制算术运算调整指令在什么情况下使用?它们都是跟在哪些指令的后面?  
14哪些指令采用隐含寻址?  
15使用算术运算类指令应注意哪些问题?  
16逻辑运算指令有几种?  
17测试指令和比较指令在使用时有什么不同?  
18算术移位指令和逻辑移位指令有什么不同?  
19控制转移类指令的作用是什么?有哪几种?  
20什么叫串?串操作指令有哪些?串前缀在什么情况下使用?  
21输入/输出指令起什么作用?寻址方式有哪些?  
22设DS=2000H，SS=3000H,BP=0200H,SI=4000H,BUF=1000H,EAX=00001000H,EBX= 00002000H,假设按16位实模式操作，确定下列每条指令访问内存的物理地址，并且指出源操作数及目的操作数的寻址方式。  
(1) MOV AL，[1234H](2) MOV EDX，[BX]  
(3) MOV CL，[BX+100H](4) MOV [SI],EBX  
(5) MOV AH，BUF[BX+SI](6) MOV EAX,[BP+1234H]  
(7) MOV [EAX+EBX],DH  
23试指出下列指令中的错误。  
(1) MOV [BX],[SI](2) MOV AH,DX  
(3) INC [BX](4) MOV DS,SS  
(5) XCHG AX，2000H(6) MOV AX，[BX+DX]  
(7) XCHG [SP],ES(8) ADD [AX],BX  
(9) MOV AX，DI+SI(10) INAL，BX  
24指出下列算术逻辑指令执行后标志CF，ZF，SF，PF，OF和AF的状态。  
MOV AL，80H  
DEC AL  
ADD AL，10H  
SUB AL，10H  
MOV AL，3AH  
AND AL，0F0H  
**OR** AL，0F0H  
XOR AL，0F0H  
25使AX寄存器清0有多种方式，试写出这多条指令。  
26写出把首地址为BUF的字节缓冲区中第5个字节数送AL寄存器的指令，要求使用以下几种寻址方式：  
(1) 寄存器间接寻址；  
(2) 寄存器相对寻址；  
(3) 基址变址寻址。  
27试分别使用数据传送指令、交换指令和堆栈操作指令，实现将首地址为BLOCK的内存单元中两个数据字交换。BLOCK变量定义如下：  
BLOCK DW 10H，20H  
28设一个字节数据X存放在AL寄存器中，试说明下列程序的功能。  
XOR AH，AH  
SAL AX，1  
MOV BX，AX  
MOV CL，2  
SAL AX，CL  
ADD AX，BX  
29试编程实现：  
(1) AL寄存器的低4位清0；  
(2) BL寄存器的低4位置1；  
(3) CL寄存器的低4位取反；  
(4) 测试DL寄存器的最低2位是否为0，若是将0送入AL寄存器；否则将1送AL寄存器。  
30试编程统计在AX寄存器中有多少个1，并将结果送DL寄存器中。  
31试编程统计在内存BLOCK单元开始按字节存放的100个带符号数中有多少负数，并将结果存放在DL寄存器中。