

微机原理与接口技术（第二版）第三章 指令系统

[写在前面] 本参考答案供同学们在写作业的时候，遇到想不通的题时进行参考。请不要直接 copy 答案，否则对自己没有好处的。多思考，才有进步。

思考题与习题 参考答案

3.1 假定(DS)=2000H, (ES)=2100H, (SS)=1500H, (SI)=00A0H, (DI)=00B0H, (BX)=0100H, (BP)=0010H, 请指出下列指令源操作数的寻址方式。如果源操作数是存储器操作数，分别计算其有效地址和物理地址。

- (1)MOV AX, 00ABH
- (2)ADD AX, [100H]
- (3)XOR AX, [0050H]
- (4)MOV BX, [SI]
- (5)SUB AX, 0050H[BX] [DI]
- (6)CMP CL, [BX] [SI]
- (7)ADC AL, ES: [BP]
- (8)MOV DS, [BP] [SI]
- (9)AND BX, SS: [DI]
- (10)SBB 0050H[SI],BX

	EA	PA	寻址方式
(1)MOV AX, 00ABH	-	-	立即寻址
(2)ADD AX, [100H]	100H	20100H	直接寻址
(3)XOR AX, [0050H]	50H	20050H	直接寻址
(4)MOV BX, [SI]	00A0H	200A0H	间接寻址
(5)SUB AX, 0050H[BX] [DI]	200H	20200H	相对基址加变址寻址
(6)CMP CL, [BX] [SI]	01A0H	201A0H	基址加变址寻址
(7)ADC AL, ES: [BP]	0010H	21010H	间接寻址
(8)MOV DS, [BP] [SI]	00B0H	150B0H	基址加变址寻址
(9)AND BX, SS: [DI]	00B0H	150B0H	间接寻址
(10)SBB 0050H[SI],BX	-	-	寄存器寻址

3.2 指出下列指令的错误原因。

- | | |
|--------------------------------|--------------------------------------|
| (1)MOV DS, 1000H | 不能用立即寻址方式给段寄存器传数 |
| (2)MOV 10H, AL | 立即数不能作为目的操作数 |
| (3)INC [SI] | 未说明操作数类型 |
| (4)MOV 2000H[BX] [DI], [2000H] | 源目的操作数不能同时是存储器 |
| (5)MOV AL, 256 | 源目的操作数类型不匹配 |
| (6)SHR CL, 4 | 移位指令移位超过 1 时应存于 CL 中 |
| (7)MOV CS, AX | CS 不能作为目的操作数 |
| (8)ADD [AX], 1 | 寄存器间接寻址只能使用 BX/BP/SI/DI 中的一个，不能使用 AX |

- (9) MOV CX, BX+SI 运算表达式的结果需为确定的数值，而 **BX** 和 **SI** 不确定
- (10) PUSH CL **8086/8088** 的堆栈操作都是以字为单位
- (11) XOR DX, BL 源目的操作数类型不匹配
- (12) IN AL, 200H **IN** 指令源操作数不能为 **16** 位立即数 端口地址应为 **8** 位立即数表示，最多允许 **256** 个端
- (13) LEA BX, CX **LEA** 源操作数必须为存储器操作数
- (14) POP IP **IP** 不能为目的操作数
- (15) MOV BX, [CX+20H] 寄存器间接寻址有效地址由 **BX BP SI DI** 的一个指出

3.3 假设标志寄存器各标志初始值为 0，分别单独执行如下指令序列后，请指出 **AX** 寄存器和标志 **CF**、**ZF**、**SF**、**OF** 的值。

	AX	CF	ZF	SF	OF
(1) MOV AX, 1234H ROL AX, 1	2468H	0	0	0	0
(2) MOV AX, 5678H AND AX, 0F0FH	0608H	0	0	0	0
(3) MOV AX, 1995H ADC AX, 0FFFFH	1994H	1	0	0	0
(4) MOV AX, -1 INC AX <small>INC不影响CF</small>	0000H	0	1	0	0

	AX	CF	ZF	SF	OF
(5) XOR AX, AX SUB AX, 80H	0FF80H	1	0	1	0
(6) MOV AX, 65535 ADD AX, 1	0H	1	1	0	0
(7) MOV AL, 81H CBW <small>CBW不影响标志位</small>	FF81H	0	0	0	0
(8) MOV BX, 1938H PUSH BX POP AX	1938H	0	0	0	0

	AX	CF	ZF	SF	OF
(9) LEA BX,[7856H] MOV AX, BX	7856H	0	0	0	0
(10) MOV AX,1234H TEST AX, 1	1234H	0	1	0	0

3.4 已知指令序列如下:

ADD AX, BX

JO N1

JC N2

SUB AX, BX

JP N3

JNO N4

JNC N5

JMP N6

若 AX 和 BX 的初始值分别为以下情况, 则执行该指令序列后, 程序将分别转移至何处?为什么?

(1) (AX)= 1234H, (BX)= 5678H;

(2) (AX)= 8765H, (BX)= 7654H;

(3) (AX)= 4325H, (BX)= 8761H;

(4) (AX)= 4321H, (BX)= 7762H;

(5) (AX)= 5678H, (BX)= 1234H。

(1)(AX) = 1234H, (BX) = 5678H;	N4
1234H+5678H=68ACH AX-BX OF=0 (正数减正数不会溢出)	
(2)(AX) = 8765H, (BX) = 7654H;	N3
(3)(AX) = 4325H, (BX) = 8761H;	N6
AX-BX, CF=1(有借位) 所以: N6	
(4)(AX) = 4321H, (BX) = 7762H;	N1
AX+BX, OF=1 (正数加正数, 结果为负数, 溢出)	
(5)(AX) = 5678H, (BX) = 1234H;	N3

3.6 编写程序实现下述功能:

(1) 将立即数 27H 送 DL，立即数 8EH 送 AL，然后将 AL 和 DL 相加，结果存放在 DL 中；

```
(1) MOV DL, 27H
    MOV AL, 8EH
    ADD DL, AL
```

(2) 将立即数 1234H 送 AX，5678H 送 DX，然后将 DX 减去 AX，结果存放在 AX 中；

```
(2) MOV AX, 1234H
    MOV DX, 5678H
    SUB DX, AX
    MOV AX, DX 或 XCHG AX, DX
```

(3) 将立即数 1A2BH 送偏移地址为 1234H 的字存储单元中，再将该字存储单元内容加 1；

```
(3) MOV [1234H], 1A2BH
    INC [1234H]
    或: INC WORD PTR [1234H]
```

(4) 将 BX 寄存器的高 4 位清零，其余位不变；

```
(4) AND BX, 00FFH
```

(5) 将 DX 寄存器的最高位置 1，其余位不变；

```
(5) OR DX, 8000H
```

(6) 先将 CX 寄存器的奇数位取反，然后将 CX 寄存器的偶数位置 1；

```
(6) XOR CX, 0AAAAH  10101010101010
    OR CX, 5555H      01010101010101
```

(7) 测试 AX 是否为偶数，如果 AX 为偶数则将 CL 置 1，否则 CL 为 0；

```
(7) TEST AX, 0001H
    JNZ NEXT
    MOV CL, 1
    NEXT: MOV CL, 0
```

(8) 测试 AL 低 7 位含二进制 1 的个数是否为偶数，如果是则将 AL 最高位置 1(其余位不变)，否则将 AL 的最高位清 0(其余位不变)；

```
(8) AND AL, 7FH      0111 1111
    JNP NEXT
    OR AL, 80H
    NEXT:
```

(9) 不使用其他寄存器，将 AL 寄存器的高 4 位和低 4 位内容互换；

```
(9) ROL AL, 1      或: ROR AL, 1
    ROL AL, 1      ROR AL, 1
    ROL AL, 1      ROR AL, 1
    ROL AL, 1      ROR AL, 1
```

(10) 编程将物理地址从 10000H 至 100FFH 的内存单元内容全部清零；

```

(10) MOV AX, 1000H
    MOV DS, AX    #段基地址
    MOV BX, 0     #偏移地址
AGAIN: MOV BYTE PTR [BX], 0
    INC BX
    CMP BX, FFH
    JNE AGAIN
    (BYTE PTR 可加可不加)

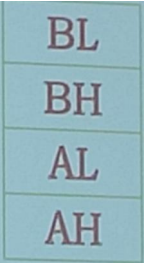
```

- (11)不使用其他指令和寄存器，利用堆栈和入栈出栈指令将 AX 和 BX 寄存器的内容交换；
 (12)不使用乘法指令，将存放在 AX 中的无符号数内容乘以 21，结果仍然存放在 AX 中；
 (13)计算双字数 DX:AX(DX 为高 16 位，AX 为低 16 位)的相反数，结果仍然存放在 DX:AX 中。

```

(11) PUSH AX
    PUSH BX
    POP AX
    POP BX

```



```

(12) MOV BX, AX
    MOV CL, 4
    SHL AX, CL
    ADD AX, BX
    MOV CL, 2
    SHL BX, CL
    ADD AX, BX

```

SHL左移一位乘2

AX=17AX

BX=4AX

```

(13) XOR BX, BX
    MOV CX, BX
    SUB BX, AX
    SBB CX, DX
    MOV DX, CX
    MOV AX, BX

```

BX,CX清零

先用SUB做低位减法，再用SBB做高位减法。

3.8 在 10100H 和 10101H 单元中分别存放 74H、85H (表示条件转移指令 JZ),若 CS=1000H, ZF=1,则执行完这条指令后，IP=()。

思路：

- JZ 指令占 2 个字节，第 1 字节为操作码 74H,第 2 字节应该为偏移量，即 85H。
- JZ 指令存放在 10100H 和 10101H 单元，由于 CS=1000H，因此指令偏移地址=0100H。
- 执行 JZ 指令时的 IP=0100H+2=0102H。

- 那么，执行完 JZ 指令后， $IP=0102H+偏移量$
- 而 85H 为 8 位带符号数，和 0102H 相加时需要扩展
- 所以，最后的 $IP=0102H+FF85H=0087H$ 。