

## 实验二 熟悉常用指令的使用

### 一、实验目的

熟悉指令系统，掌握常用指令的用法；通过实验加深对各种寻址方式的理解；能熟练使用DEBUG中的命令对指令进行反汇编，观察并了解机器代码。

### 二、实验题

**加法指令对条件标志位的影响**

SF=	1 结果为负	ZF=	1 结果为0
	0 否则		0 否则
CF=	1 和的最高有效位 有 向高位的进位		
	0 否则		
OF=	1 两个操作数符号相同，而结果符号与之相反		
	0 否则		

**减法指令对条件标志位（CF/OF/ZF/SF）的影响：**

CF=	1 被减数的最高有效位 有 向高位的借位
	0 否则
或	
CF=	1 减法转换为加法运算时 无进位
	0 否则（有进位）与加法运算刚好相反
OF=	1 两个操作数符号相反，而结果的符号与减数相同
	0 否则

CF 位表示 无符号数 减法的溢出。  
OF 位表示 带符号数 减法的溢出。

标志名	标志为1	标志为0
OF 溢出(是/否)	OV	NV
DF 方向(减量/增量)	DN	UP
IF 中断(允许/关闭)	EI	DI
SF 符号(负/正)	NG	PL
ZF 零(是/否)	ZR	NZ
AF 辅助进位(是/否)	AC	NA
PF 奇偶(偶/奇)	PE	PO
CF 进位(是/否)	CY	NC

1. 求出以下各十六进制数与十六进制数 62A0(0110 0010 1010 0000)正数之和，单步执行、观察标志位 SF、ZF、CF、OF 的值，并与自己的判断进行比较。

(1) 1234(0001)    (2) 4321    (3) CFA0    (4) 9D60

Mov 指令对标志位没有影响

正数 + 正数 = 负数（溢出）

负数 + 负数 = 正数（溢出）

提示：加法指令 ADD DST, SRC

2. 求出以下各十六进制数与十六进制数 4AE0 的差值，单步执行、观察标志位 SF、ZF、CF、OF 的值，并与自己的判断进行比较。

计算机中减法的本质采用补码的形式，本质是加法器，还是相当于做加法

正数 - 正数 = 正数 + (负数)

正数 - 负数 = 正数 + (正数) = 负数 (溢出)

负数 - 正数 = 负数 + 负数 = 正数 (溢出)

负数 - 负数 = 负数 + (正数)

提示 减法指令 SUB DST, SRC

3. 将下面 3 条指令写入从 2000:0 开始的内存单元中，利用这 3 条指令计算 2 的 8 次方。

2^8

-a 2000:0

-d 2000:0

**MOV AX, 1**

**ADD AX, AX**

**JMP 2000:0003**

4. 使用 Debug，将下面的内容写入内存，逐条执行，观察每条指令执行后，CPU 中相关寄存器中内容的变化。

**b8 20 4e**

**05 16 14**

**bb 00 20**

**01 d8**

**89 c3**

**01 d8**

**b8 1a 00**

**bb 26 00**

**00 d8**

**00 dc**

**00 c7**

**b4 00**

**00 d8**

**04 9c**

5. 通过 debug 命令将下面的程序写入内存，逐条执行，根据指令执行后的实际运行情况填

空。

### 三 实验报告

要求： 1.每步操作过程及所用指令；2.用截图方式反应指令执行前后的相关寄存器或存储单元内容的变化，并对所得结果进行分析。

## 1. 任务一:各十六进制数与十六进制数 62A0 正数之和，单步执行、观察标志位 SF、ZF、CF、OF 的值，并与自己的判断进行比较。

(1) 1234(0001)    (2) 4321    (3) CFA0    (4) 9D60

### 1. 添加指令到代码段

```
-u
^ Error
-u
0770:0000 B8A062      MOV     AX,62A0
0770:0003 053412      ADD     AX,1234
0770:0006 4C          DEC     SP
0770:0007 CD21      INT     21
0770:0009 0000      ADD     [BX+SI],AL
0770:000B 0000      ADD     [BX+SI],AL
0770:000D 0000      ADD     [BX+SI],AL
0770:000F 0000      ADD     [BX+SI],AL
0770:0011 0000      ADD     [BX+SI],AL
0770:0013 0000      ADD     [BX+SI],AL
0770:0015 0000      ADD     [BX+SI],AL
0770:0017 0000      ADD     [BX+SI],AL
0770:0019 0000      ADD     [BX+SI],AL
0770:001B 0000      ADD     [BX+SI],AL
0770:001D 0000      ADD     [BX+SI],AL
0770:001F 0000      ADD     [BX+SI],AL
-t
AX=62A0 BX=0000 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=0770 IP=0003  NU UP EI PL NZ NA PO NC
0770:0003 053412      ADD     AX,1234
```

### 2. 使用-t 命令运行指令

(1) mov ax, 62A0

Add ax,1234

```
-t
AX=62A0 BX=0000 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=0770 IP=0003  NU UP EI PL NZ NA PO NC
0770:0003 053412      ADD     AX,1234

-t
AX=74D4 BX=0000 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=0770 IP=0006  NU UP EI PL NZ NA PE NC
0770:0006 B8A062      MOV     AX,62A0
```

1234 (0001)+

+ 62A0(0110)+ --> (0111)+

**Table 1:**

标志名	值	标志位
SF	0	PL
ZF	0	NZ
CF(最高位是否向更高位进位)	0	NC
OF	0	NV

(2) mov ax, 62A0

Add ax,4321

```
-t
AX=62A0 BX=0000 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=0770 IP=0009  NU UP EI PL NZ NA PE NC
0770:0009 052143      ADD     AX,4321
-t
AX=A5C1 BX=0000 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=0770 IP=000C  DU UP EI NG NZ NA PO NC
0770:000C B8A062      MOV     AX,62A0
```

4321 (0100)+

+ 62A0(0110)+ --> (1010)-

**Table 2:**

标志名	值	标志位
SF	1	NG
ZF	0	NZ
CF(最高位是否向更高位进位)	0	NC
OF	1	OV

(3) mov ax, 62A0

Add ax,cfa0

```
-t
AX=62A0 BX=0000 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=0770 IP=000F  DU UP EI NG NZ NA PO NC
0770:000F 05A0CF      ADD     AX,CFA0
-t
AX=3240 BX=0000 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=0770 IP=0012  NU UP EI PL NZ NA PO CY
0770:0012 B8A062      MOV     AX,62A0
```

CFA0 (1100)-

+ 62A0(0110)+ --> (0010)+

Table 3:

标志名	值	标志位
SF	0	PL
ZF	0	NZ
CF(最高位是否向更高位进位)	1	CY
OF	1	NV

(4) mov ax, 62A0

Add ax,9d60

```
X=62A0 BX=0000 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
S=0760 ES=0760 SS=076F CS=0770 IP=0015  NU UP EI PL NZ NA PO CY
770:0015 05609D      ADD     AX,9D60
t
X=0000 BX=0000 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
S=0760 ES=0760 SS=076F CS=0770 IP=0018  NU UP EI PL ZR NA PE CY
770:0018 0000      ADD     [BX+SI],AL      DS:0000=CD
```

9d60(1001)-

+ 62A0(0110)+ --> (0000) + (后位有进位)

Table 4:

标志名	值	标志位
SF	0	PL
ZF	1	ZR
CF(最高位是否向更高位进位)	1	CY
OF	0	NV

2 任务二:各十六进制数与十六进制数 4AE0 的差值，单步执行、观察标志位 SF、ZF、CF、OF 的值，并与自己的判断进行比较。

1. 根据题目要求向代码段输入指令

```
2000:0034 0000      ADD     [BX+SI],AL
2000:0036 0000      ADD     [BX+SI],AL
2000:0038 0000      ADD     [BX+SI],AL
2000:003A 0000      ADD     [BX+SI],AL
2000:003C 0000      ADD     [BX+SI],AL
2000:003E 0000      ADD     [BX+SI],AL
2000:0040 0000      ADD     [BX+SI],AL
2000:0042 0000      ADD     [BX+SI],AL
2000:0044 0000      ADD     [BX+SI],AL
2000:0046 0000      ADD     [BX+SI],AL
-u 2000:0
2000:0000 B83412      MOV     AX,1234
2000:0003 2DE04A      SUB     AX,4AE0
2000:0006 B8905D      MOV     AX,5D90
2000:0009 2DE04A      SUB     AX,4AE0
2000:000C B89090      MOV     AX,9090
2000:000F 2DE04A      SUB     AX,4AE0
2000:0012 B804EA      MOV     AX,EA04
2000:0015 2DE04A      SUB     AX,4AE0
2000:0018 031E0600    ADD     BX,[0006]
2000:001C 50          PUSH    AX
2000:001D 53          PUSH    BX
2000:001E 58          POP     AX
2000:001F 5B          POP     BX
```

(1) 1234      (2) 5D90      (3) 9090      (4) EA04

2. 运行指令

(1)mov ax, 1234

sub ax, 4AE0

```
-t=2000:0
AX=1234 BX=C0FC CX=0009 DX=0000 SP=00FC BP=0000 SI=0000 DI=0000
DS=FFFF ES=0760 SS=2200 CS=2000 IP=0003 NU UP EI PL NZ NA PE NC
2000:0003 2DE04A      SUB     AX,4AE0
-t
AX=C754 BX=C0FC CX=0009 DX=0000 SP=00FC BP=0000 SI=0000 DI=0000
DS=FFFF ES=0760 SS=2200 CS=2000 IP=0006 NU UP EI NG NZ NA PO CY
2000:0006 B8905D      MOV     AX,5D90
```

1234(0001)-

+ ~4AE0 (1011)+ --> (0110)+

~4AE0 = 1011 0101 0011 0000

Table 5:

标志名	值	标志位
SF	1	NG
ZF	0	NZ
CF(最高位是否向更高位借位)	1	CY
OF	0	NV

(2) mov ax, 5d90

sub ax, 4AE0

```
-t
AX=5D90 BX=C0FC CX=0009 DX=0000 SP=00FC BP=0000 SI=0000 DI=0000
DS=FFFF ES=0760 SS=2200 CS=2000 IP=0009 NU UP EI NG NZ NA PO CY
0000:0009 2DE04A SUB AX,4AE0
-t
AX=12B0 BX=C0FC CX=0009 DX=0000 SP=00FC BP=0000 SI=0000 DI=0000
DS=FFFF ES=0760 SS=2200 CS=2000 IP=000C NU UP EI PL NZ NA PO NC
0000:000C B89090 MOV AX,9090
```

5D90(0101)-

+ ~4AE0 (1011)+ --> (0110)+

Table 6:

标志名	值	标志位
SF	0	PL
ZF	0	NZ
CF(最高位是否向更高位借位)	0	NC
OF	0	NV

(3) mov ax, 9090

sub ax, 4AE0

```
-t
AX=9090 BX=C0FC CX=0009 DX=0000 SP=00FC BP=0000 SI=0000 DI=0000
DS=FFFF ES=0760 SS=2200 CS=2000 IP=000F NU UP EI PL NZ NA PO NC
2000:000F 2DE04A SUB AX,4AE0
-t
AX=45B0 BX=C0FC CX=0009 DX=0000 SP=00FC BP=0000 SI=0000 DI=0000
DS=FFFF ES=0760 SS=2200 CS=2000 IP=0012 NU UP EI PL NZ NA PO NC
2000:0012 B804E0 MOV AX,E004
```

9090(1001)-

+ ~4AE0 (1011)+ --> (0100)+

Table 7:

标志名	值	标志位
SF	0	PL
ZF	0	NZ
CF(最高位是否向更高位借位)	0	NC
OF	1	OV

(4) mov ax, EA04

sub ax, 4AE0

```
-t
AX=EA04 BX=C0FC CX=0009 DX=0000 SP=00FC BP=0000 SI=0000 DI=0000
DS=FFFF ES=0760 SS=2200 CS=2000 IP=0015 00 UP EI PL NZ NA PO NC
2000:0015 2DE04A SUB AX,4AE0
-t
AX=9F24 BX=C0FC CX=0009 DX=0000 SP=00FC BP=0000 SI=0000 DI=0000
DS=FFFF ES=0760 SS=2200 CS=2000 IP=0018 00 UP EI NG NZ NA PE NC
2000:0018 031E0600 ADD BX,[0006] DS:0006=2F31
```

EA04(1110)-  
+ ~4AE0 (1011)+ --> (0011)+

Table 8:

标志名	值	标志位
SF	1	NG
ZF	0	NZ
CF(最高位是否向更高位借位)	0	NC
OF	0	NV

3 任务三：使用跳转指令计算 2 的 8 次方

```
AX=4AE0 BX=0000 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=0770 IP=002D 00 UP EI NG NZ NA PE CY
0770:002D 2D04EA SUB AX,EA04
-t
AX=60DC BX=0000 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=0770 IP=0030 00 UP EI PL NZ AC PO CY
0770:0030 0000 ADD [BX+SI],AL DS:0000=CD
-a 2000:0
2000:0000 mov ax,1
2000:0003 add ax,ax
2000:0005 jmp 2000:0003
2000:0007
--d 2000:0
^ Error
-d 2000:0
2000:0000 BB 01 00 01 C0 EB FC 00-00 00 00 00 00 00 00 00 .....
2000:0010 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
2000:0020 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
2000:0030 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
2000:0040 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
2000:0050 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
2000:0060 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
2000:0070 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
```



```

AX=0040 BX=0000 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=2000 IP=0005  NU UP EI PL NZ NA PO NC
2000:0005 EBFC JMP 0003
-t

AX=0040 BX=0000 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=2000 IP=0003  NU UP EI PL NZ NA PO NC
2000:0003 01C0 ADD AX,AX
-t

AX=0080 BX=0000 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=2000 IP=0005  NU UP EI PL NZ NA PO NC
2000:0005 EBFC JMP 0003
-t

AX=0080 BX=0000 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=2000 IP=0003  NU UP EI PL NZ NA PO NC
2000:0003 01C0 ADD AX,AX
-t

AX=0100 BX=0000 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=2000 IP=0005  NU UP EI PL NZ NA PE NC
2000:0005 EBFC JMP 0003
- ;

```

运算结果中  $AX=(0100)_{16}=(16^2)_{10}=(2^8)_{10}=(256)_{10}$

## 4 任务四:数据和指令在内存中的相互转换和使用

1. 使用-e 命令输入数据到指定地址后,使用-u 命令反汇编得结果展示

```

2000:0034 0000 ADD [BX+SI],AL
2000:0036 0000 ADD [BX+SI],AL
2000:0038 0000 ADD [BX+SI],AL
2000:003A 0000 ADD [BX+SI],AL
2000:003C 0000 ADD [BX+SI],AL
2000:003E 0000 ADD [BX+SI],AL
2000:0040 0000 ADD [BX+SI],AL
2000:0042 0000 ADD [BX+SI],AL
2000:0044 0000 ADD [BX+SI],AL
-u 2000:0
2000:0000 B8204E MOV AX,4E20
2000:0003 051614 ADD AX,1416
2000:0006 B80020 MOV BX,2000
2000:0009 01D8 ADD AX,BX
2000:000B 89C3 MOV BX,AX
2000:000D 01D8 ADD AX,BX
2000:000F B81A00 MOV AX,001A
2000:0012 B82600 MOV BX,0026
2000:0015 00D8 ADD AL,BL
2000:0017 00DC ADD AH,BL
2000:0019 00C7 ADD BH,AL
2000:001B B400 MOV AH,00
2000:001D 00D8 ADD AL,BL
2000:001F 049C ADD AL,9C
- ;

```

2. 使用-t 命令运行指令

```

-t=2000:0
AX=4E20 BX=0000 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=2000 IP=0003  NU UP EI PL NZ NA PE NC
2000:0003 051614 ADD AX,1416
-t

AX=6236 BX=0000 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=2000 IP=0006  NU UP EI PL NZ NA PE NC
2000:0006 B80020 MOV BX,2000
-t

AX=6236 BX=2000 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=2000 IP=0009  NU UP EI PL NZ NA PE NC
2000:0009 01D8 ADD AX,BX
-t

AX=8236 BX=2000 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=2000 IP=000B  OU UP EI NG NZ NA PE NC
2000:000B 89C3 MOV BX,AX

```

```
AX=8236 BX=8236 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=2000 IP=000D 00 UP EI NG NZ NA PE NC
2000:000D 01D8 ADD AX,BX
-t

AX=046C BX=8236 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=2000 IP=000F 00 UP EI PL NZ NA PE CY
2000:000F B81A00 MOV AX,001A
-t

AX=001A BX=8236 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=2000 IP=0012 00 UP EI PL NZ NA PE CY
2000:0012 B82600 MOV BX,0026
-t

AX=001A BX=0026 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=2000 IP=0015 00 UP EI PL NZ NA PE CY
2000:0015 00D8 ADD AL,BL
-t

AX=0040 BX=0026 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=2000 IP=0017 00 UP EI PL NZ AC PO NC
2000:0017 00DC ADD AH,BL
-t
```

```
AX=2640 BX=0026 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=2000 IP=0019 00 UP EI PL NZ NA PO NC
2000:0019 00C7 ADD BH,AL
-t

AX=2640 BX=4026 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=2000 IP=001B 00 UP EI PL NZ NA PO NC
2000:001B B400 MOV AH,00
-t

AX=0040 BX=4026 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=2000 IP=001D 00 UP EI PL NZ NA PO NC
2000:001D 00D8 ADD AL,BL
-t

AX=0066 BX=4026 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=2000 IP=001F 00 UP EI PL NZ NA PE NC
2000:001F 049C ADD AL,9C
-t

AX=0002 BX=4026 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=2000 IP=0021 00 UP EI PL NZ AC PO CY
2000:0021 0000 ADD [BX+SI],AL DS:4026=00
-t
```

结论:内存中的数据可以看作数据也可以看作指令

## 5 任务五:栈的相关操作

使用-a 和-u 命令写入和反汇编指令

```
2000:002B 0000 ADD [BX+SI],AL
2000:002D 0000 ADD [BX+SI],AL
2000:002F 0000 ADD [BX+SI],AL
2000:0031 0000 ADD [BX+SI],AL
2000:0033 0000 ADD [BX+SI],AL
2000:0035 0000 ADD [BX+SI],AL
2000:0037 0000 ADD [BX+SI],AL
2000:0039 0000 ADD [BX+SI],AL
2000:003B 0000 ADD [BX+SI],AL
2000:003D 0000 ADD [BX+SI],AL
2000:003F 0000 ADD [BX+SI],AL
-u 2000:0
2000:0000 B8FFFF MOV AX,FFFF
2000:0003 BED8 MOV DS,AX
2000:0005 B80022 MOV AX,2200
2000:0008 BED0 MOV SS,AX
2000:000A BC0001 MOV SP,0100
2000:000D A10000 MOV AX,[0000]
2000:0010 03060200 ADD AX,[0002]
2000:0014 B81E0400 MOV BX,[0004]
2000:0018 031E0600 ADD BX,[0006]
2000:001C 50 PUSH AX
2000:001D 53 PUSH BX
2000:001E FF360400 PUSH [0004]
-t
```

关于栈的操作

**MOV AX,0FFFFH**

**MOV DS,AX**

MOV AX,2200

MOV SS,AX

MOV SP,0100

堆栈指针初始化

MOV AX,[0] ;AX=C0EA

```
-t
AX=C0EA BX=4026 CX=0009 DX=0000 SP=0100 BP=0000 SI=0000 DI=0000
DS=FFFF ES=0760 SS=2200 CS=2000 IP=0010 NU UP EI PL NZ AC PO CY
2000:0010 03060200 ADD AX,[0002] DS:0002=0012
```

送的字单元,直接寻址方式

-d ffff:0

ADD AX,[2] ;AX=C0FC

```
-t
AX=C0FC BX=4026 CX=0009 DX=0000 SP=0100 BP=0000 SI=0000 DI=0000
DS=FFFF ES=0760 SS=2200 CS=2000 IP=0014 NU UP EI NG NZ NA PE NC
0000:0014 8B1E0400 MOV BX,[0004] DS:0004=30F0
```

MOV BX,[4] ;BX=30F0

```
-t
AX=C0FC BX=30F0 CX=0009 DX=0000 SP=0100 BP=0000 SI=0000 DI=0000
DS=FFFF ES=0760 SS=2200 CS=2000 IP=0018 NU UP EI NG NZ NA PE NC
2000:0018 031E0600 ADD BX,[0006] DS:0006=2F31
```

ADD BX,[6] ;BX=6021

```
-t
AX=C0FC BX=6021 CX=0009 DX=0000 SP=0100 BP=0000 SI=0000 DI=0000
DS=FFFF ES=0760 SS=2200 CS=2000 IP=001C NU UP EI PL NZ NA PE NC
2000:001C 50 PUSH AX
```

PUSH AX ;SP=00FE; 修改的内存单元的地址是 2200:00FC, 内容为 C0FC

Sp 先减 2,再压入 AX 的数据入栈

```
-t
AX=C0FC BX=6021 CX=0009 DX=0000 SP=00FE BP=0000 SI=0000 DI=0000
DS=FFFF ES=0760 SS=2200 CS=2000 IP=001D NU UP EI PL NZ NA PE NC
2000:001D 53 PUSH BX
```

PUSH BX ;SP=00FC;修改的内存单元的地址是 2200:00FA, 内容为 6021

```
-t
AX=C0FC BX=6021 CX=0009 DX=0000 SP=00FC BP=0000 SI=0000 DI=0000
DS=FFFF ES=0760 SS=2200 CS=2000 IP=001E NU UP EI PL NZ NA PE NC
0000:001E 5B POP AX
```

POP AX ;SP=00FE; AX=6021

```
-t
AX=6021 BX=6021 CX=0009 DX=0000 SP=00FE BP=0000 SI=0000 DI=0000
DS=FFFF ES=0760 SS=2200 CS=2000 IP=001F NU UP EI PL NZ NA PE NC
2000:001F 5B POP BX
```

POP BX ;SP=0100; BX=C0FC

```
-t
AX=6021 BX=C0FC CX=0009 DX=0000 SP=0100 BP=0000 SI=0000 DI=0000
DS=FFFF ES=0760 SS=2200 CS=2000 IP=0020 NU UP EI PL NZ NA PE NC
2000:0020 FF360400 PUSH [0004] DS:0004=30F0
```

PUSH [4] ;SP=00FE ; 修改的内存单元的地址是 2200:00FC, 内容为(20004)=30F0

```
-t
AX=6021 BX=C0FC CX=0009 DX=0000 SP=00FE BP=0000 SI=0000 DI=0000
DS=FFFF ES=0760 SS=2200 CS=2000 IP=0024 NU UP EI PL NZ NA PE NC
2000:0024 FF360600 PUSH [0006] DS:0006=2F31
```

PUSH [6] ;SP=00FC ; 修改的内存单元的地址是 2200:00FA, 内容为(20006)=

6021

```
t
X=6021 BX=C0FC CX=0009 DX=0000 SP=00FC BP=0000 SI=0000 DI=0000
S=FFFF ES=0760 SS=2200 CS=2000 IP=002B NU UP EI PL NZ NA PE NC
000:002B 0000 ADD [BX+SI],AL DS:C0FC=00
```