# 计算机组成原理实验报告

班级：计科 2 班

姓名：薄劲阳

学号：2020115025

# 实验一 汇编环境安装与基础实验

**实验内容：**

（1）下载安装 MASM5.0 集成开发软件，或者其他 x86 汇编语言开发环境；熟悉开发工具的使用方法；

（2）利用串操作指令在内存中存储字符串“Hello world”,通过观察窗口查看内存内容，检查每个字符的 ASCII 编码是否正确。

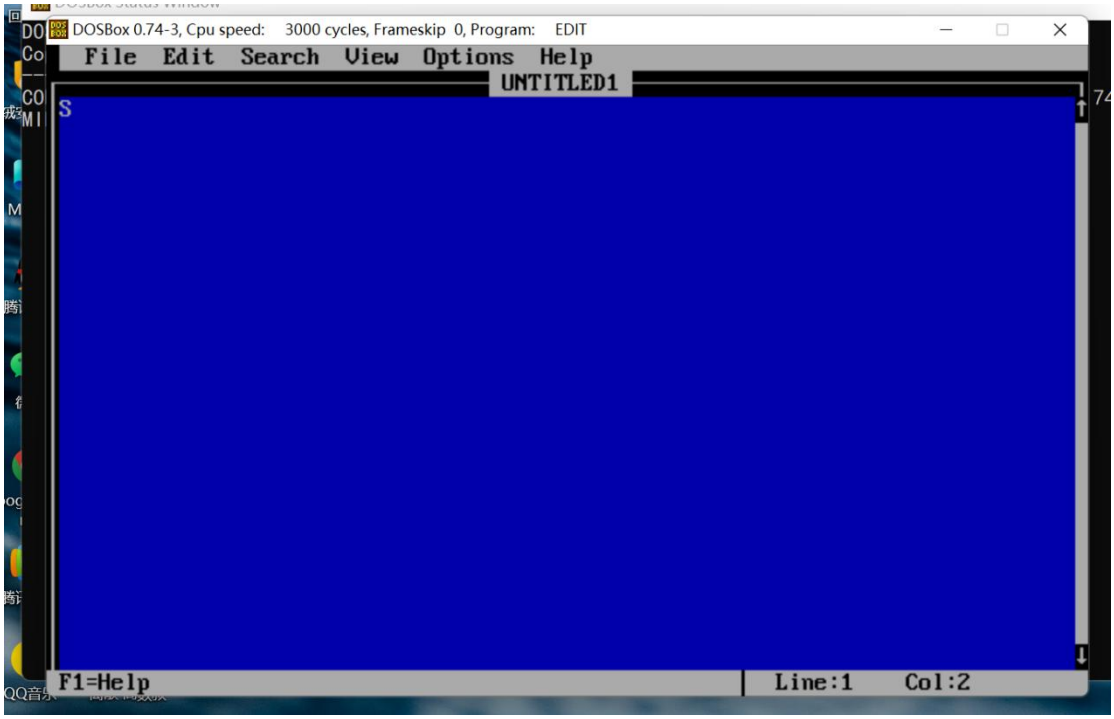
**要求：**

（1）编写汇编程序

（2）保存内存检测窗口截图

- 查看 cs：ip 中的内容

- 进入 edit 模式，编写程序



```
1.    data segment ;这里定义一个数据段
2.              tab  db  'Hello  world$'  ;这里用内存存放字节数据 'hellow world!',$用来判断字符串是否输出完毕
3.    data ends ;数据段的结束标志
4.
5.    code segment ;这里定义了一个代码段
6.        assume cs:code, ds:data ;这里把程序中定义的段与对应的段寄存器关联起来
7.
8.    start: ;这里是一个标号，根据 end 后面的标号判断这里是程序的开始位置
9.          mov ax,data
10.         mov ds,ax ;这里把数据段的地址放到数据段寄存器 ds 中
11.         lea dx,tab ;dx 中放将要显示数据的偏移地址
12.         mov ah,9h
13.         int 21h ;调用 21 号中断的 9 号功能来显示字符串
14.         mov ah,4ch
15.         int 21h
16.   code ends ;代码段的结束语
17.       end start ;定义程序从哪个标号处开始执行
```

- 编译，链接，运行程序
  - 编译

```
C:\>masm
Microsoft (R) Macro Assembler Version 5.00
Copyright (C) Microsoft Corp 1981-1985, 1987.  All rights reserved.

Source filename [.ASM]: hello
Object filename [hello.OBJ]:
Source listing  [NUL.LST]:
Cross-reference [NUL.CRF]:

  51748 + 464780 Bytes symbol space free

       0 Warning Errors
       0 Severe  Errors
```

  - 链接

```
C:\>link

Microsoft (R) Overlay Linker  Version 3.60
Copyright (C) Microsoft Corp 1983-1987.  All rights reserved.

Object Modules [.OBJ]: hello
Run File [HELLO.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
LINK : warning L4021: no stack segment
```

- 运行程序

```
C:\>hello
Hello world
```

- 查看 ds 段中内存中内容，判断是否进行了修改

```
-r
AX=FFFF  BX=0000  CX=0021  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=075C  ES=075C  SS=076B  CS=076D  IP=0000   NV UP EI PL NZ NA PO NC
076D:0000 B86C07        MOV     AX,076C
```

```
-d076c: 0 10
076C:0000  48 65 6C 6C 6F 20 77 6F-72 6C 64 24 00 00 00 00   Hello world$....
076C:0010  B8                                                .
```

## 实验二　内存读写

**实验内容：**

（1）在内存数据区连续存储 10 个随机整数，依次读出这 10 个数，并存储到内存的另一个区域。

（2）不要使用循环程序

**要求：**

（1）编写汇编程序

（2）保存内存检测窗口截图

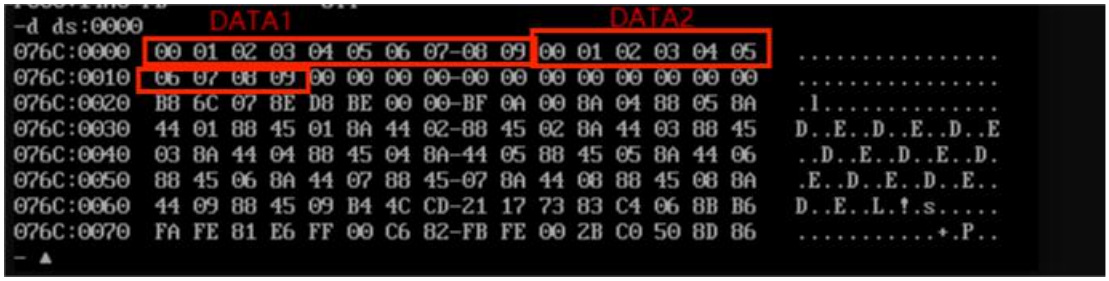
## 1. 汇编程序如下

```
1.    ；数据段
2.    DATA SEGMENT
3.        AREA1 DB 00H,01H,02H,03H,04H,05H,06H,07H,08H,09H ；AREA1 存放十个未知数
4.        AREA2 DB 10 DUP(0)  ；AREA2 为十个字节大小的的内存区，初始化为 0
5.    DATA ENDS
6.
7.    CODE SEGMENT
8.        ASSUME CS:CODE,DS:DATA  ;伪指令
9.    START:
10.       MOV AX,DATA
11.       MOV DS,AX  ；DS 指向 DATA 段
12.       ；获取偏移地址
13.       MOV SI,OFFSET AREA1
14.       MOV DI,OFFSET AREA2
15.       ;将 AREA1 中的内容搬运到 AREA2 中
16.       MOV AL,[SI]
17.       MOV [DI],AL
18.       ;搬运下一个内存单元（字节），偏移地址加一即可
19.       MOV AL,[SI+1]
20.       MOV [DI+1],AL
21.
22.       MOV AL,[SI+2]
23.       MOV [DI+2],AL
24.
25.       MOV AL,[SI+3]
26.       MOV [DI+3],AL
27.
28.       MOV AL,[SI+4]
29.       MOV [DI+4],AL
30.
31.       MOV AL,[SI+5]
32.       MOV [DI+5],AL
33.
34.       MOV AL,[SI+6]
35.       MOV [DI+6],AL
36.
37.       MOV AL,[SI+7]
```

```
38.         MOV [DI+7],AL
39.
40.         MOV AL,[SI+8]
41.         MOV [DI+8],AL
42.
43.         MOV AL,[SI+9]
44.         MOV [DI+9],AL
45.
46.         MOV AH,4CH
47.         INT 21H
48.     ;，这个中断调用指令就是告诉程序当程序里的指令（除了放在它最低行的“mov
49.     ah,4ch int 21h”）执行完毕后要做什么——返回 dos，此时程序就会结束，电脑
        界面上 dos 窗口（就是 windowsxp 运行 cmd 后出现的那个窗口）就会出现一行英文，
        其意思是“请按任意键继续”
50.     CODE ENDS
51.     END START
```

**2. 内存检测窗口如下：**

可以看到 DATA1 与 DATA2 内容一样，数据转移成功。




## 实验三 定点整数加减法

**实验内容：**

（1）在内存数据区定义 10 个小于 20 的定点整数，利用加法指令累加，结果存入内存指定单元；

（2）可以使用循环程序；

**要求：**

（1）编写汇编程序

（2）保存内存检测窗口截图

**1. 汇编程序如下：**

```
1.    DATA SEGMENT
2.        AREA1 DB 00H,01H,02H,03H,04H,05H,06H,07H,08H,09H  ; 连续的十个内存单元
3.        AREA2 DB ?  ; 在 AREA2 中随机存放数据
4.    DATA ENDS
5.
6.    CODE SEGMENT
7.        ASSUME CS:CODE,DS:DATA
8.    START:
9.        MOV AX,DATA
10.       MOV DS,AX  ; DS 指向 DATA（获取段地址）
11.
12.       MOV SI,OFFSET AREA1  ; 获取 AREA1 的偏移地址
13.       MOV CX,10  ; 循环变量
14.       MOV BL,0  ; 累加器
15.   AGAIN:
16.       MOV AL,[SI]
17.       ADD BL,[SI]
18.       INC SI
19.       INC DI
20.       DEC CX
21.       JNZ AGAIN  ; 如果不为零 跳转到 AGAIN
22.       HLT
23.       ; 将求和结果放入 AREA2 中
24.       MOV DI,OFFSET AREA2
25.       MOV [DI],BL
26.       MOV AH,4CH
27.       INT 21H
28.
29.   CODE ENDS
30.   END START
```

**2. 内存检测窗口如下：**

定义的 10 个数为 0~9，相加结果为 2D

```
DATA SEGMENT
        AREA1 DB 00H,01H,02H,03H,04H,05H,06H,07H,08H,09H
        AREA2 DB ?
```


## 实验四 寻址方式测试

**实验内容：**

使用不同的汇编指令，分别测试隐含寻址、立即寻址、直接寻址、间接寻址、寄存器寻址、寄存器间接寻址、偏移寻址和段寻址；

**要求：**

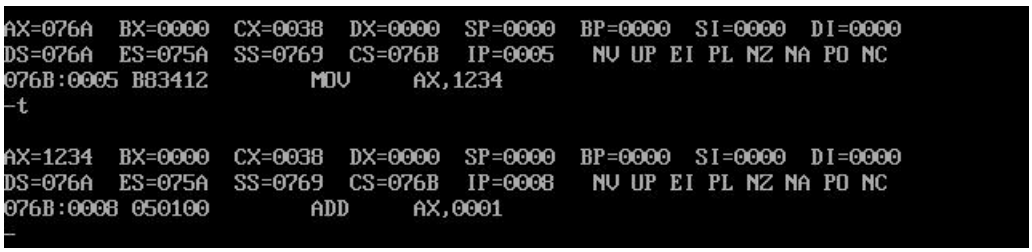
（1）编写汇编程序

```
1.  ;数据段
2.  DATA    SEGMENT
3.  DATA1   DW 2222H  ; 定义一个字单元 高八位和低八位均为 22H
4.  DATA2   DB 0 ;定义一个字节单元 存零
5.  DATA    ENDS
6.  ;代码段
7.  CODE    SEGMENT
8.  ASSUME CS:CODE,DS:DATA
9.  START:  MOV AX,DATA
10. MOV DS,AX  ; DS 指向 DATA 段（获取段地址）
11. MOV AX,1234H ;立即寻址 - 源操作数为立即数
12.
13. ADD AX,0001H;
```

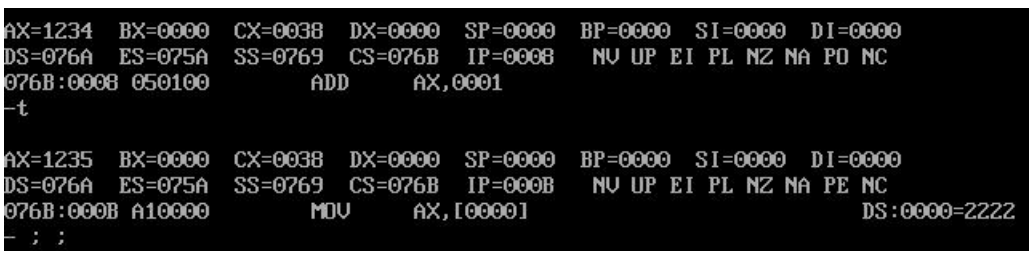
```
14.
15.    MOV AX,DS:[0000H] ;直接寻址-立即数存放在内存单元中
16.
17.    MOV AX,1234H
18.    MOV DX,AX ;寄存器寻址-源操作数为寄存器编号，立即数在寄存器中
19.
20.    MOV SI,1234H
21.    MOV AX,[SI]
22.    MOV DX,AX ;寄存器间接寻址-源操作数为寄存器编号，寄存器中存放立即数的内存
       地址
23.
24.    MOV AX,[SI+100H];寄存器相对寻址
25.
26.    MOV BX,0001H
27.    MOV SI,0500H
28.    MOV AX,[BX][SI] ;基址变址寻址
29.
30.    MOV AH,4CH
31.    INT 21H
32.    CODE    ENDS
33.    END START
```

（2）保存内存检测窗口截图

MOV AX,1234H;立即寻址




ADD AX,0001H;隐含寻址



MOV AX,DS:[0000H];直接寻址

```
AX=1235  BX=0000  CX=0038  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=000B   NV UP EI PL NZ NA PE NC
076B:000B A10000        MOV     AX,[0000]                       DS:0000=2222
-t

AX=2222  BX=0000  CX=0038  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=000E   NV UP EI PL NZ NA PE NC
076B:000E B83412        MOV     AX,1234
-_
```

MOV AX,1234H

MOV DX,AX;寄存器寻址

```
AX=2222  BX=0000  CX=0038  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=000E   NV UP EI PL NZ NA PE NC
076B:000E B83412        MOV     AX,1234
-t

AX=1234  BX=0000  CX=0038  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0011   NV UP EI PL NZ NA PE NC
076B:0011 8BD0          MOV     DX,AX
-t

AX=1234  BX=0000  CX=0038  DX=1234  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0013   NV UP EI PL NZ NA PE NC
076B:0013 BE3412        MOV     SI,1234
-_
```

MOV SI,1234H

MOV AX,[SI];寄存器间接寻址

```
AX=1234  BX=0000  CX=0038  DX=1234  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0013   NV UP EI PL NZ NA PE NC
076B:0013 BE3412        MOV     SI,1234
-t

AX=1234  BX=0000  CX=0038  DX=1234  SP=0000  BP=0000  SI=1234  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0016   NV UP EI PL NZ NA PE NC
076B:0016 8B04          MOV     AX,[SI]                         DS:1234=0508
-t

AX=0508  BX=0000  CX=0038  DX=1234  SP=0000  BP=0000  SI=1234  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0018   NV UP EI PL NZ NA PE NC
076B:0018 8B840001      MOV     AX,[SI+0100]                    DS:1334=08EC
-_;
```

MOV AX,[SI+100H];寄存器相对寻址

```
AX=0508  BX=0000  CX=0038  DX=1234  SP=0000  BP=0000  SI=1234  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0018   NV UP EI PL NZ NA PE NC
076B:0018 8B840001      MOV     AX,[SI+0100]                    DS:1334=08EC
-t

AX=08EC  BX=0000  CX=0038  DX=1234  SP=0000  BP=0000  SI=1234  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=001C   NV UP EI PL NZ NA PE NC
076B:001C BB0100        MOV     BX,0001
- ;
```

MOV BX,0001H

MOV SI,0500H

MOV AX,[BX][SI];偏移寻址（基址变址寻址）

```
AX=08EC  BX=0000  CX=0038  DX=1234  SP=0000  BP=0000  SI=1234  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=001C   NV UP EI PL NZ NA PE NC
076B:001C BB0100        MOV     BX,0001
-t

AX=08EC  BX=0001  CX=0038  DX=1234  SP=0000  BP=0000  SI=1234  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=001F   NV UP EI PL NZ NA PE NC
076B:001F BE0005        MOV     SI,0500
-t

AX=08EC  BX=0001  CX=0038  DX=1234  SP=0000  BP=0000  SI=0500  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0022   NV UP EI PL NZ NA PE NC
076B:0022 8B00          MOV     AX,[BX+SI]                      DS:0501=000A
-t

AX=000A  BX=0001  CX=0038  DX=1234  SP=0000  BP=0000  SI=0500  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0024   NV UP EI PL NZ NA PE NC
076B:0024 B44C          MOV     AH,4C
- ;
```

# 实验五 堆栈实验

**实验内容：**

在内存开辟一个堆栈区，利用堆栈操作指令实现 4 个已赋值寄存器内容入栈，然后出栈，重新保存到对应寄存器。

**要求：**

（1）编写汇编程序

```
1.    DATA    SEGMENT
2.    DATA1   DW 2222H
3.    DATA2   DB ?
```

```
DATA    ENDS
；堆栈段
STACK   SEGMENT
DW 500 DUP(0)  ；500 个内存单元（字节） 存放 500 个 0
STACK   ENDS

CODE    SEGMENT
ASSUME CS:CODE,DS:DATA,SS:STACK
START:  MOV AX,DATA
MOV DS,AX
；堆栈段特点——先进后出
MOV AX,0000H
MOV BX,1000H
MOV CX,2000H
MOV DX,3000H
；压栈
PUSH AX
PUSH BX
PUSH CX
PUSH DX

MOV AX,0H
MOV BX,0H
MOV CX,0H
MOV DX,0H
；出栈
POP DX
POP CX
POP BX
POP AX

MOV AH,4CH
INT 21H
CODE    ENDS
END START
```

（2）保存内存检测窗口截图

首先给 AX,BX,CX,DX 分别赋值为 0000，1000,2000,3000

```
AX=076A  BX=0000  CX=0429  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=07AA  IP=0005   NV UP EI PL NZ NA PO NC
07AA:0005 B80000        MOV     AX,0000
-t

AX=0000  BX=0000  CX=0429  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=07AA  IP=0008   NV UP EI PL NZ NA PO NC
07AA:0008 BB0010        MOV     BX,1000
-t

AX=0000  BX=1000  CX=0429  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=07AA  IP=000B   NV UP EI PL NZ NA PO NC
07AA:000B B90020        MOV     CX,2000
-t

AX=0000  BX=1000  CX=2000  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=07AA  IP=000E   NV UP EI PL NZ NA PO NC
07AA:000E BA0030        MOV     DX,3000
-t

AX=0000  BX=1000  CX=2000  DX=3000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=07AA  IP=0011   NV UP EI PL NZ NA PO NC
07AA:0011 50            PUSH    AX
- ;
```

然后将其按照 AX,BX,CX,DX 的顺序入栈，之后改变四个寄存器的值全为 0

```
AX=0000  BX=1000  CX=2000  DX=3000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=07AA  IP=0011   NV UP EI PL NZ NA PO NC
07AA:0011 50            PUSH    AX
-t

AX=0000  BX=1000  CX=2000  DX=3000  SP=FFFE  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=07AA  IP=0012   NV UP EI PL NZ NA PO NC
07AA:0012 53            PUSH    BX
-t

AX=0000  BX=1000  CX=2000  DX=3000  SP=FFFC  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=07AA  IP=0013   NV UP EI PL NZ NA PO NC
07AA:0013 51            PUSH    CX
-t

AX=0000  BX=1000  CX=2000  DX=3000  SP=FFFA  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=07AA  IP=0014   NV UP EI PL NZ NA PO NC
07AA:0014 52            PUSH    DX
-t

AX=0000  BX=1000  CX=2000  DX=3000  SP=FFF8  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=07AA  IP=0015   NV UP EI PL NZ NA PO NC
07AA:0015 B80000        MOV     AX,0000
- ;
```

```
AX=0000  BX=1000  CX=2000  DX=3000  SP=FFF8  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=07AA  IP=0015   NV UP EI PL NZ NA PO NC
07AA:0015 B80000        MOV     AX,0000
-t

AX=0000  BX=1000  CX=2000  DX=3000  SP=FFF8  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=07AA  IP=0018   NV UP EI PL NZ NA PO NC
07AA:0018 BB0000        MOV     BX,0000
-t

AX=0000  BX=0000  CX=2000  DX=3000  SP=FFF8  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=07AA  IP=001B   NV UP EI PL NZ NA PO NC
07AA:001B B90000        MOV     CX,0000
-t

AX=0000  BX=0000  CX=0000  DX=3000  SP=FFF8  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=07AA  IP=001E   NV UP EI PL NZ NA PO NC
07AA:001E BA0000        MOV     DX,0000
-t

AX=0000  BX=0000  CX=0000  DX=0000  SP=FFF8  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=07AA  IP=0021   NV UP EI PL NZ NA PO NC
07AA:0021 5A            POP     DX
-_;
```

最后按照 DX,CX,BX,AX 的顺序出栈，可以看到结果为 0000,1000,2000,3000，与初始结果相同。

```
AX=0000  BX=0000  CX=0000  DX=0000  SP=FFF8  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=07AA  IP=0021   NV UP EI PL NZ NA PO NC
07AA:0021 5A            POP     DX
-t

AX=0000  BX=0000  CX=0000  DX=3000  SP=FFFA  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=07AA  IP=0022   NV UP EI PL NZ NA PO NC
07AA:0022 59            POP     CX
-t

AX=0000  BX=0000  CX=2000  DX=3000  SP=FFFC  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=07AA  IP=0023   NV UP EI PL NZ NA PO NC
07AA:0023 5B            POP     BX
-t

AX=0000  BX=1000  CX=2000  DX=3000  SP=FFFE  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=07AA  IP=0024   NV UP EI PL NZ NA PO NC
07AA:0024 58            POP     AX
-t

AX=0000  BX=1000  CX=2000  DX=3000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=07AA  IP=0025   NV UP EI PL NZ NA PO NC
07AA:0025 B44C          MOV     AH,4C
-_;
```