《微机原理与接口技术》实验报告

姓　　名：	
学　　号：	
专业班级：	
实验名称：	第一次实验
实验日期：	2023 年 10 月 17 日

备注：

（1）请将报告电子版发到邮箱 MrSuInterfaceWork@163.com，文件名：姓名-学号-班级-微机原理-第 X 次实验.docx。

（2）提交的内容：文档，实验源代码（有几个任务就提交几个源代码）

（3）邮件的主题和文件名同名。

（4）文档排版统一为小四仿宋，行间距离 1.5 倍行距。

（5）提交日期：下一次实验之前

一、实验目的

1)熟悉汇编程序开发(和调试)环境;

2)熟悉汇编程序基本结构和分段概念;

3)熟悉8086指令系统常见指令;

4)熟悉8086寻址方式和应用;

二、实验内容

1)实验1.1 寄存器读写(和调试)实验
2)实验1.2 一维数值数组处理实验
3)实验1.3 字符串处理实验
4)实验1.4 二维数组处理实验
5)实验1.5 位处理实验

三、实验过程和结果

3.1 寄存器读写(和调试)实验

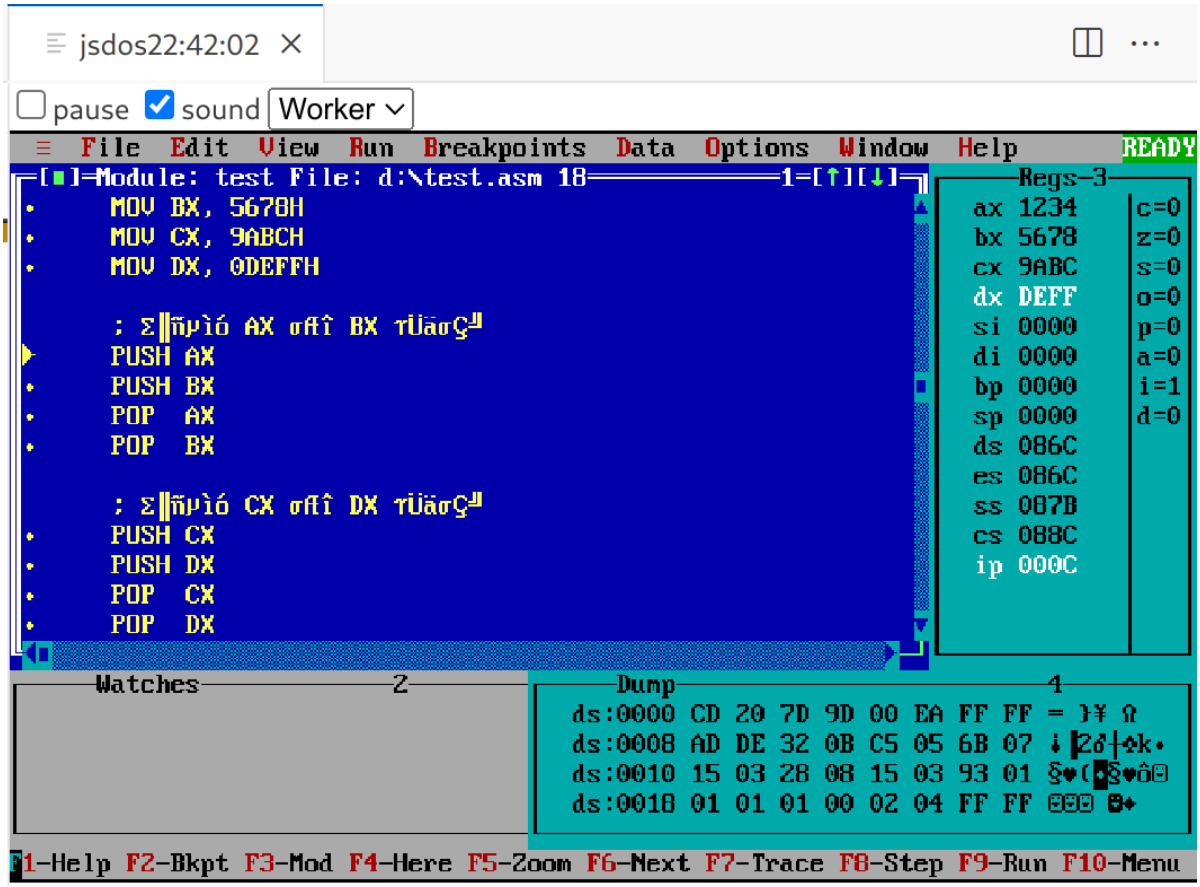
3.1.1 源代码

```
DATA SEGMENT

DATA ENDS

STACK SEGMENT
          DW 128 DUP(0)
STACK ENDS

CODE SEGMENT
          ASSUME CS:CODE, DS:DATA, SS:STACK
    START:
          MOV    AX, 1234H
          MOV    BX, 5678H
          MOV    CX, 9ABCH
          MOV    DX, 0DEFFH

    ; 交换 AX 和 BX 的值
          PUSH   AX
          PUSH   BX
          POP    AX
          POP    BX

    ; 交换 CX 和 DX 的值
          PUSH   CX
          PUSH   DX
          POP    CX
          POP    DX

    ; 交换 AX 的高 8 位 (AH) 和 DX 的低 8 位 (DL)
          PUSH   AX
```

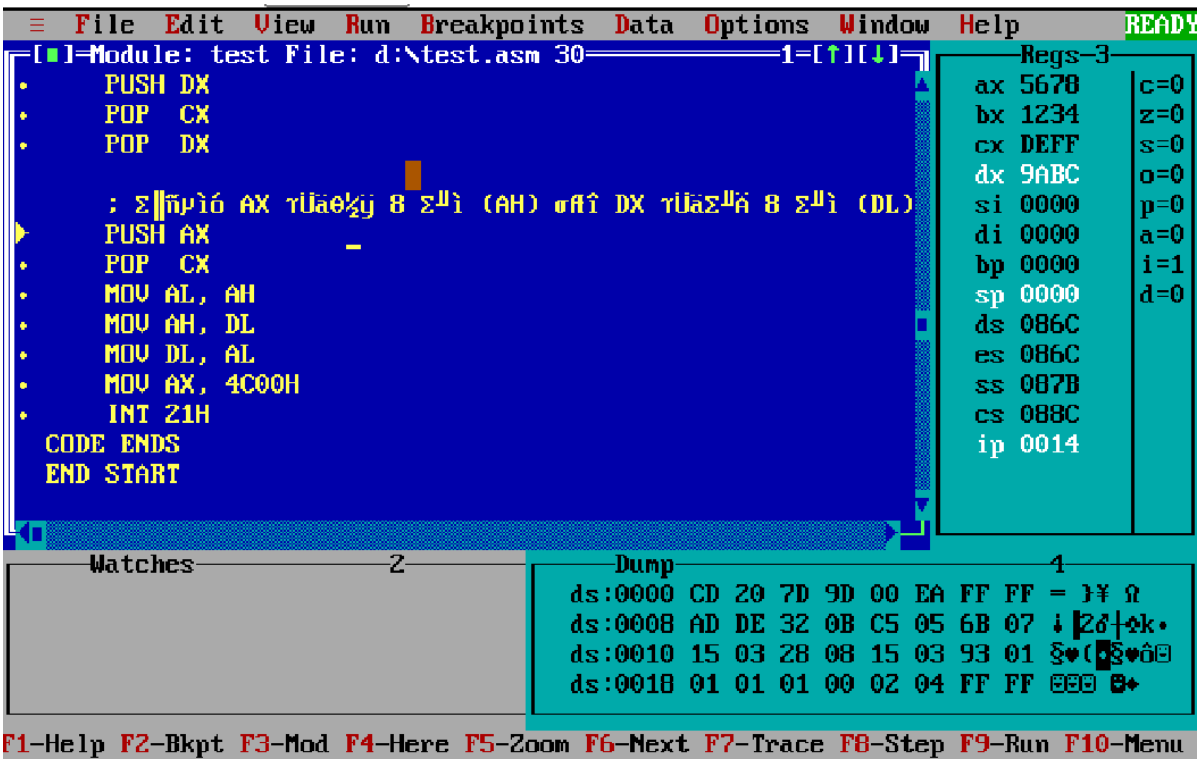
```
31              POP     CX
32              MOV     AL, AH
33              MOV     AH, DL
34              MOV     DL, AL
35              MOV     AX, 4C00H
36              INT     21H
37  CODE ENDS
38  END START
39
```

3.1.2 观察到的现象

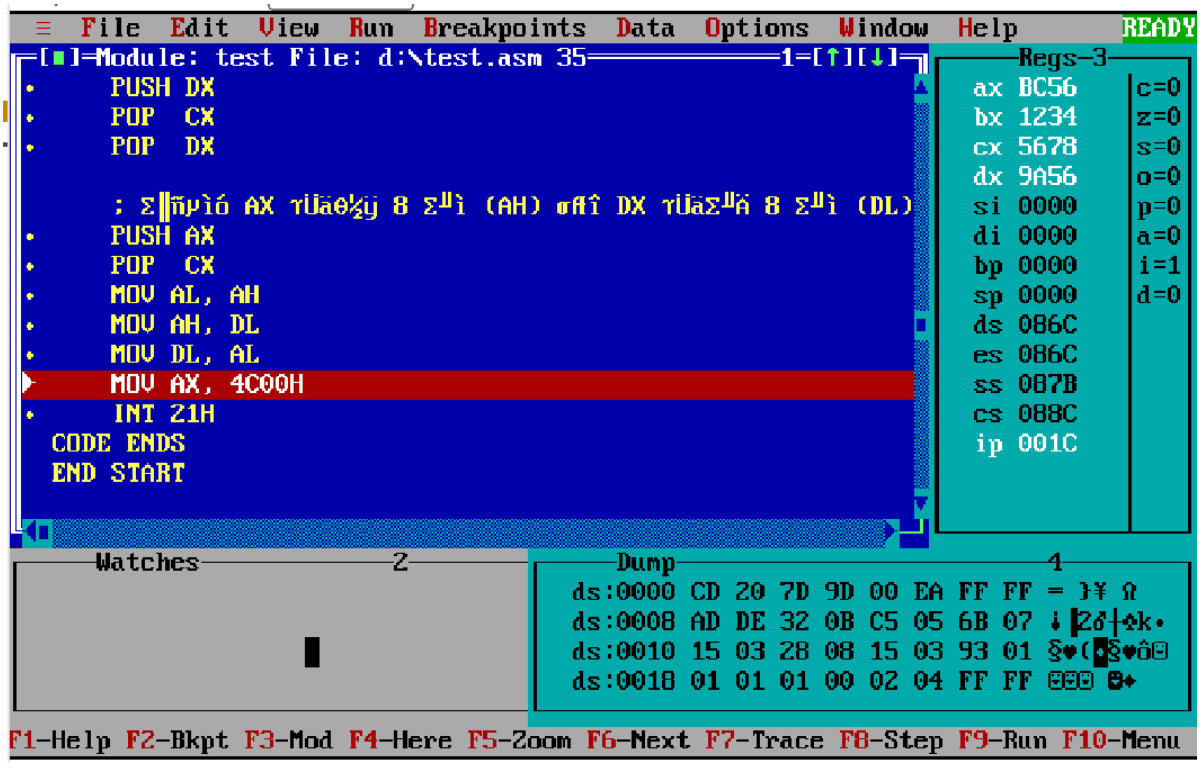
运行程序前：



运行程序后：首先交换AX和BX的值，CX和DX的值，在TD右侧的窗口可以看到具体的值：



随后，进行AX的高8位和DX的低8位交换。



3.2 一维数值数组处理实验

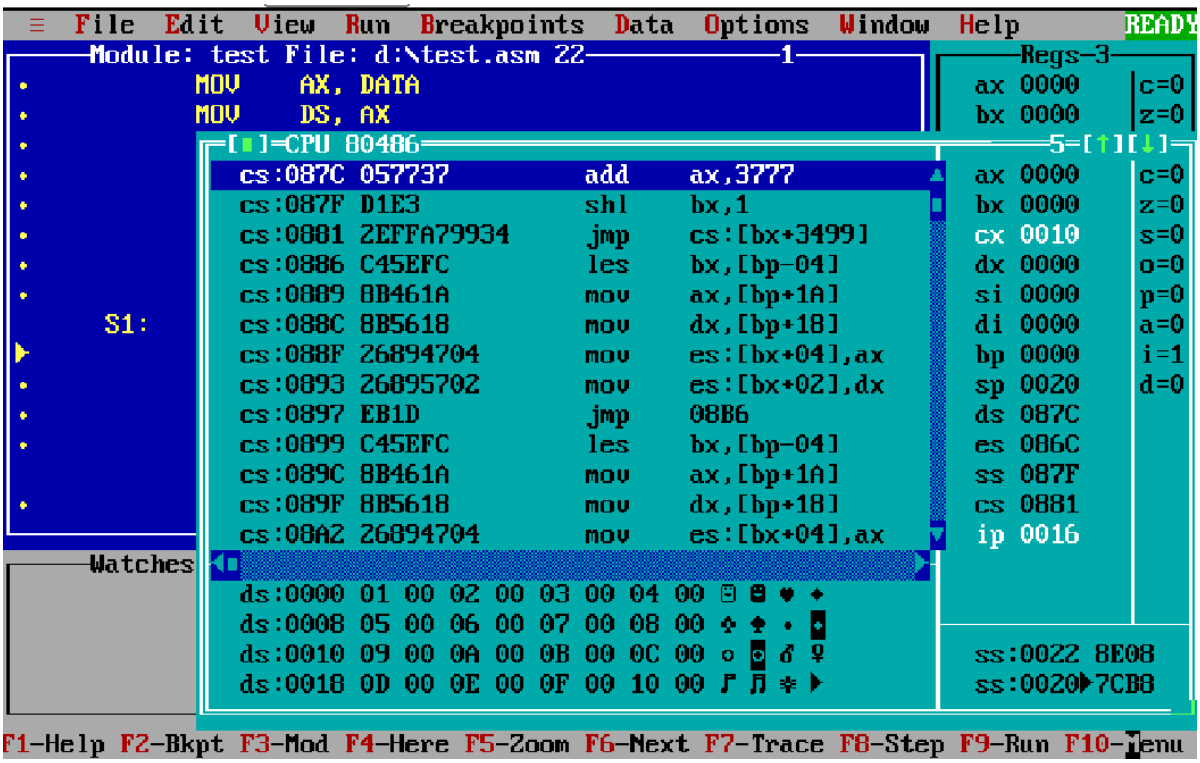
3.2.1 源代码

```
DATA SEGMENT
      NUMBERS DW 0001H, 0002H, 0003H, 0004H, 0005H, 0006H, 0007H, 0008H, 0009H, 000AH, 000BH, 000CH, 000DH, 000EH, 000FH, 0010H
      SUM     DW 0FFFFH
DATA ENDS

STACK SEGMENT
            DW 16 DUP(0)
```

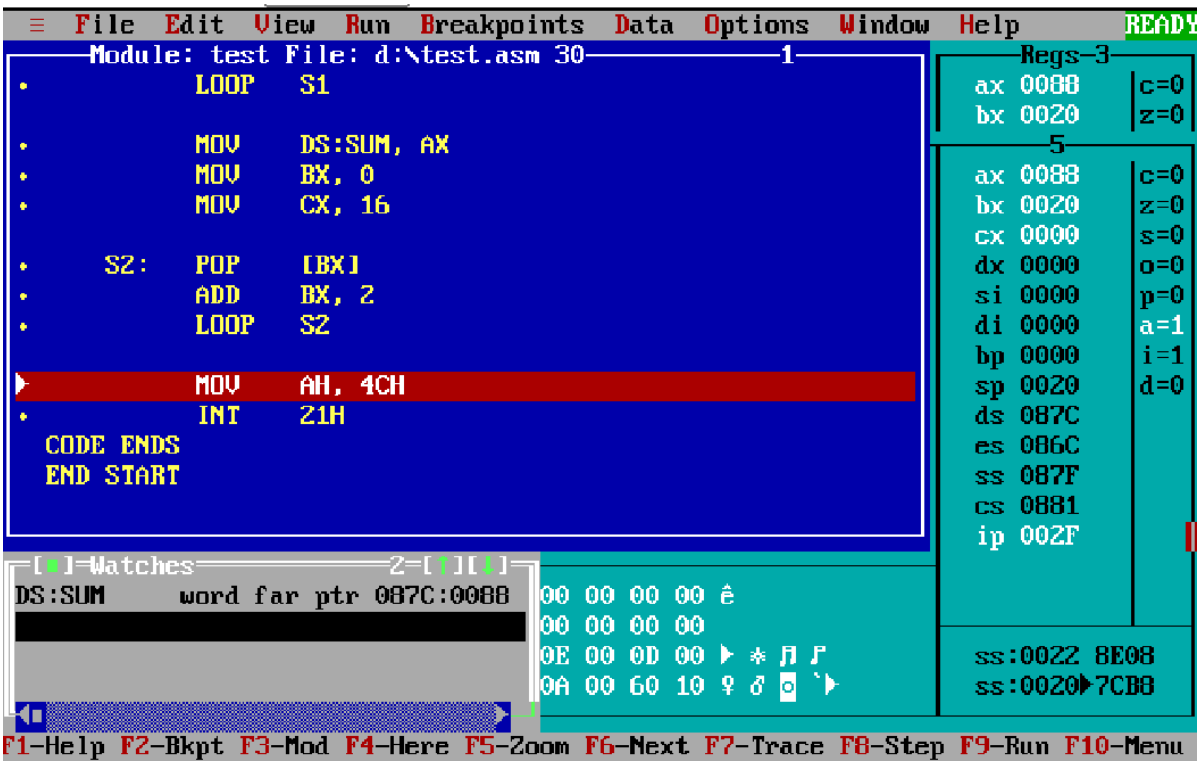
```
STACK ENDS

CODE SEGMENT
            ASSUME CS:CODE, DS:DATA, SS:STACK
      START:
            MOV    AX, DATA
            MOV    DS, AX
            MOV    AX, STACK
            MOV    SS, AX
            MOV    SP, 20H
            MOV    AX, 0
            MOV    BX, 0
            MOV    CX, 16
      S1:
            ADD    AX, [BX]
            PUSH   [BX]
            ADD    BX, 2
            LOOP   S1

            MOV    DS:SUM, AX
            MOV    BX, 0
            MOV    CX, 16

      S2:   POP    [BX]
            ADD    BX, 2
            LOOP   S2

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

```

3.2.2 观察到的现象

运行程序，初始化结束后可以在TD.exe窗口输入DS并回车，查看代码段的数据：



按F2设置断点之后，按F9执行整个程序，会自动暂停，此时再次通过监视窗口查看，可以发现DS段的数据均逆序存储；在Watches窗口添加 SUM值，可以查看到SUM的值为0088H,符合预期结果。



3.3 字符串处理实验

3.3.1 源代码

```
DATA SEGMENT
        MaxLength    DB 100
        ActualLength DB ?
        STRING       DB 100 DUP(?)
        ChangeRow    DB 0DH,0AH,'$'
        UpperCase    DB 100 DUP(?)
DATA ENDS
```

```
8
9
10  STACK SEGMENT
11          DW 128 DUP(0)
12  STACK ENDS
13
14  CODE SEGMENT
15                      ASSUME CS: CODE, DS: DATA, SS: STACK
16          MAIN:
17                      MOV     AX, DATA
18                      MOV     DS, AX
19          INPUT:
20                      MOV     DX,OFFSET MaxLength
21                      MOV     AH,10
22                      INT     21H
23
24          ;转换为大写
25                      MOV     BX,OFFSET STRING
26                      MOV     DI,OFFSET UpperCase
27                      MOV     CH,0
28                      MOV     CL,ActualLength
29          ;CX=ACTUALLENGTH
30          TO_UPPER:
31                      MOV     AL,[BX]
32                      CALL    TurnCapital
33                      MOV     [DI],AL
34                      INC     BX
35                      INC     DI
36                      LOOP    TO_UPPER
37
38          ; 添加终止字符到UpperCase
39                      MOV     BYTE PTR [DI], 24H
40
41          ; 输出大写字符串
42          OUTPUT_UPPER:
43                      MOV     DX,OFFSET ChangeRow
44                      MOV     AH,9
45                      INT     21H
46
47                      MOV     DX,OFFSET UpperCase
48                      MOV     AH,9
49                      INT     21H
50
51          ;转换为小写, CX=ACTUALLENGTH
52                      MOV     BX,OFFSET STRING
53                      MOV     CH,0
54                      MOV     CL,ActualLength
55          TO_LOWER:
56                      MOV     AL,[BX]
57                      CALL    TurnSmall
58                      MOV     [BX],AL
59                      INC     BX
60                      LOOP    TO_LOWER
61
62          ; 添加终止字符到STRING
63                      MOV     BYTE PTR [BX], 24H
```

```
64
65          ; 输出小写字符串
66          OUTPUT_LOWER:
67                          MOV     DX,OFFSET ChangeRow
68                          MOV     AH,9
69                          INT     21H
70
71                          MOV     DX,OFFSET STRING
72                          MOV     AH,9
73                          INT     21H
74
75          END_MAIN:       MOV     AH,4CH
76                          INT     21H
77
78
79  TurnCapital PROC
80          ;AL中的字符转为大写
81                          CMP     AL,'a'
82                          JL      DONE_CAP
83                          CMP     AL,'z'
84                          JG      DONE_CAP
85                          SUB     AL,20H
86          ; 大写=小写-20H
87          DONE_CAP:
88                          RET
89  TurnCapital ENDP
90
91  TurnSmall PROC
92          ;AL中的字符作大小写转换
93                          CMP     AL,'A'
94                          JL      DONE_SMALL
95                          CMP     AL,'Z'
96                          JG      DONE_SMALL
97                          ADD     AL,20H
98          DONE_SMALL:
99                          RET
100 TurnSmall ENDP
101
102 CSEG ENDS
103         END MAIN
104
```

3.3.2 观察到的现象

输入长度不大于100的字符串，可以在屏幕上输出全部大写和全部小写的版本，完成了实验任务的加强版。

```
D:\>set PATH=C:\TASM

D:\>TASM D:\test.asm
Turbo Assembler  Version 4.1  Copyright (c) 1988, 1996 Borland International

Assembling file:   D:\test.asm  to  test.OBJ
*Warning* D:\test.asm(10) Reserved word used as symbol: STACK
*Warning* D:\test.asm(102) Unmatched ENDS: CSEG
Error messages:    None
Warning messages:  2
Passes:            1
Remaining memory:  466k


D:\>TLINK D:\test
Turbo Link  Version 7.1.30.1. Copyright (c) 1987, 1996 Borland International
Warning: No stack

D:\>D:\test
HeLlo
HELLO
hello
D:\>
```

3.4 二维数组处理实验

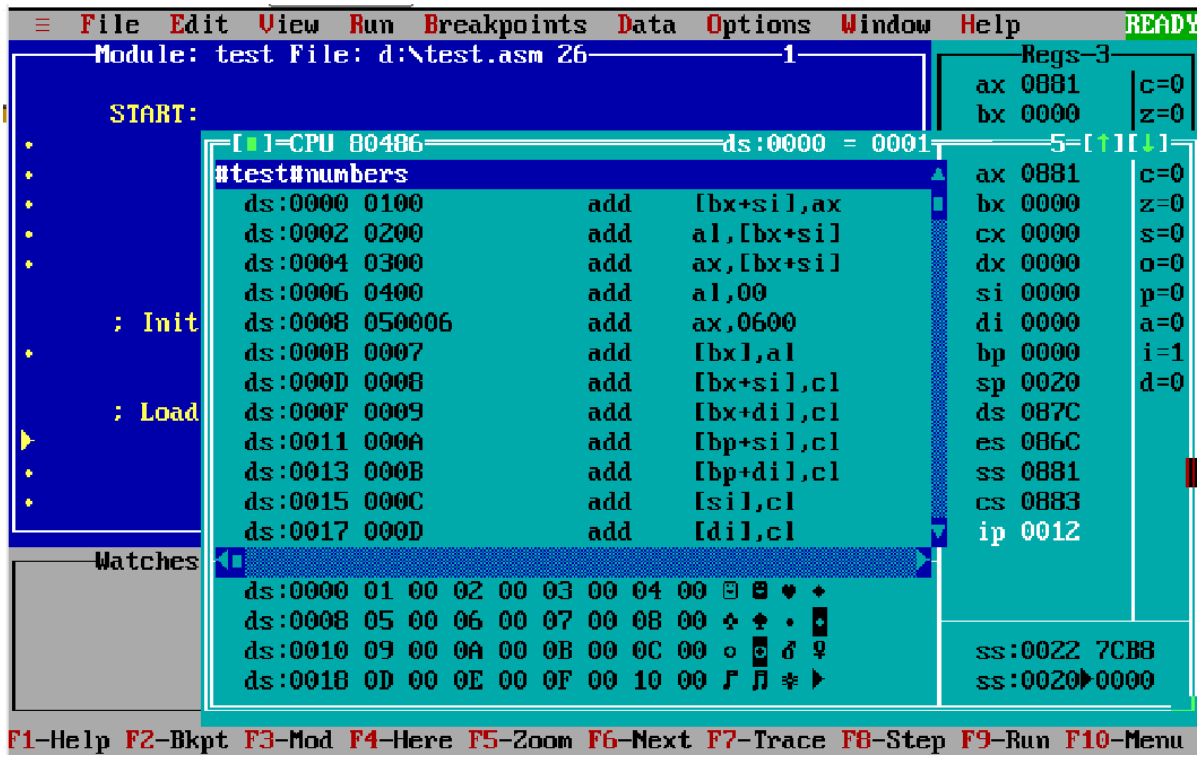
3.4.1 源代码

```
DATA SEGMENT
    NUMBERS   DW 0001H, 0002H, 0003H, 0004H, 0005H, 0006H, 0007H, 0008H, 0009H, 000AH, 000BH, 000CH, 000DH, 000EH, 000FH, 0010H,0011H, 0012H, 0013H, 0014H, 0015H, 0016H, 0017H, 0018H,0019H, 001AH, 001BH, 001CH, 001DH, 001EH, 001FH, 0020H
    ChangeRow DB 0DH,0AH,'$'
    MAXVAL    DW ?
DATA ENDS

STACK SEGMENT
          DW 16 DUP(0)
STACK ENDS

CODE SEGMENT
                ASSUME CS:CODE, DS:DATA, SS:STACK
    MINVAL      DW     ?                            ; Store MINVAL at the beginning of the code segment

    START:
                MOV    AX, DATA
                MOV    DS, AX
                MOV    AX, STACK
                MOV    SS, AX
                MOV    SP, 20H

    ; Initialize BX to point to the start of the array
                MOV    BX, OFFSET NUMBERS

    ; Load the first number into AX to initialize MAXVAL and MINVAL
                MOV    AX, [BX]
```

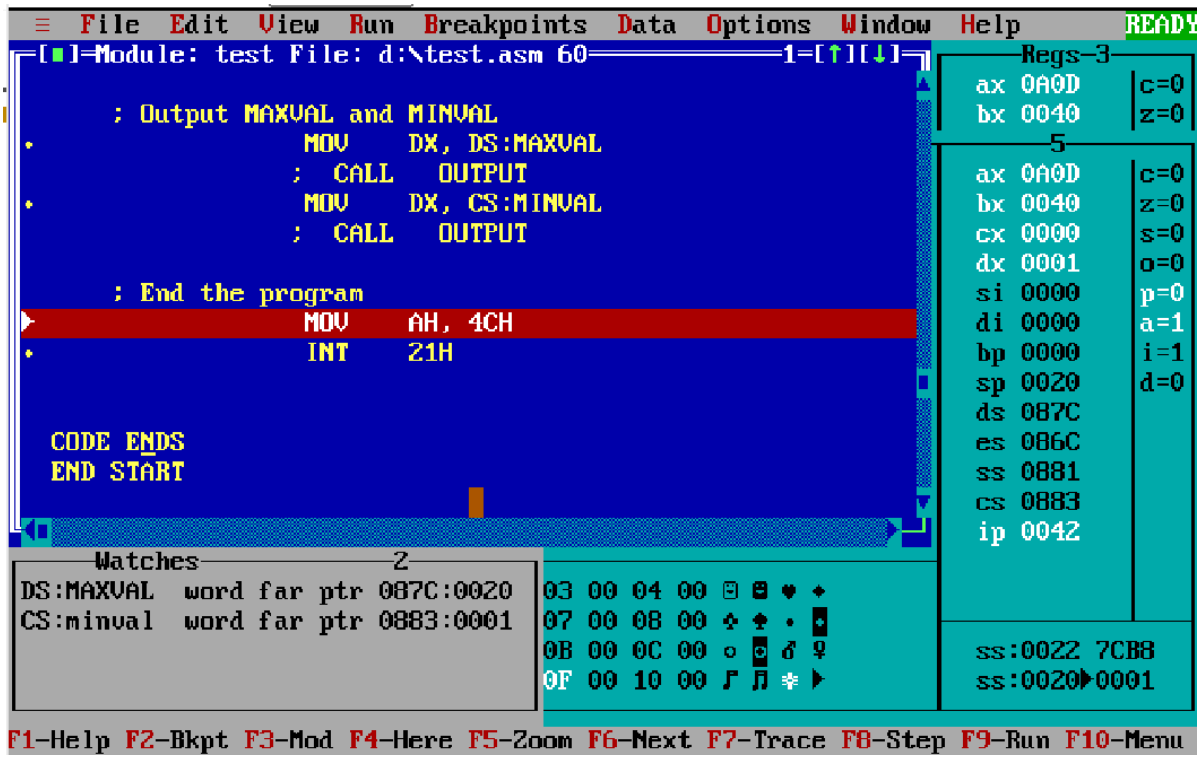
```
                MOV     DS:MAXVAL, AX
                MOV     CS:MINVAL, AX

    ; Set CX to 32 (4*8) for loop count
                MOV     CX, 32

    FIND_MIN_MAX:
    ; Compare current number with MAXVAL
                CMP     AX, DS:MAXVAL
                JLE     NOT_MAX
                MOV     DS:MAXVAL, AX
    NOT_MAX:
    ; Compare current number with MINVAL
                CMP     AX, CS:MINVAL
                JGE     NOT_MIN
                MOV     CS:MINVAL, AX
    NOT_MIN:
    ; Move to the next number in the array
                ADD     BX, 2
                MOV     AX, [BX]
                LOOP    FIND_MIN_MAX

    ; Output MAXVAL and MINVAL
                MOV     DX, DS:MAXVAL
               ;  CALL   OUTPUT
                MOV     DX, CS:MINVAL
               ;  CALL   OUTPUT

    ; End the program
                MOV     AH, 4CH
                INT     21H


CODE ENDS
END START

```

3.4.2 观察到的现象

初始化结束后，可以在DS段观察到代码段的数据。



运行完程序后，在watches窗口可以查看最大值和最小值。



3.5 位处理实验

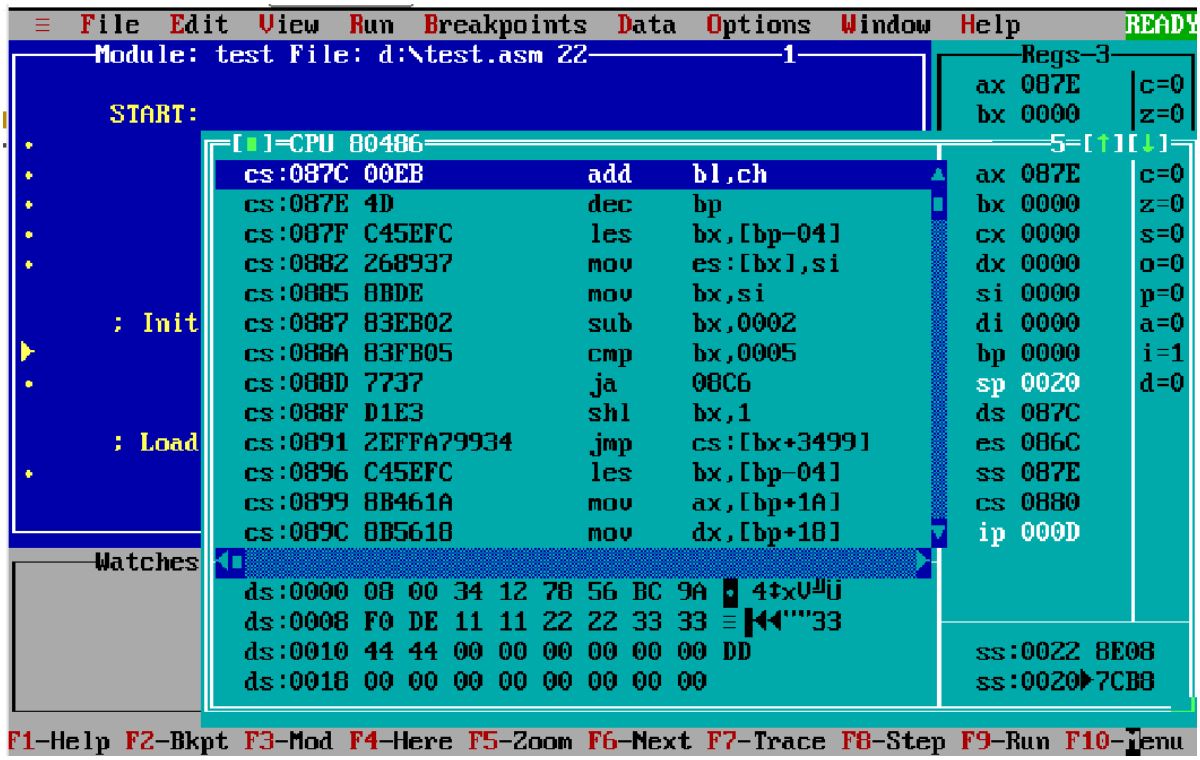
3.5.1 源代码

```
DATA SEGMENT
    COUNT   DW 8                                                    ;
8 words follow
    NUMBERS DW 1234H, 5678H, 9ABCH, 0DEF0H, 1111H, 2222H, 3333H, 4444H
    ENDNUM  DW ?                                                    ;
Placeholder for the end of the numbers
DATA ENDS

```

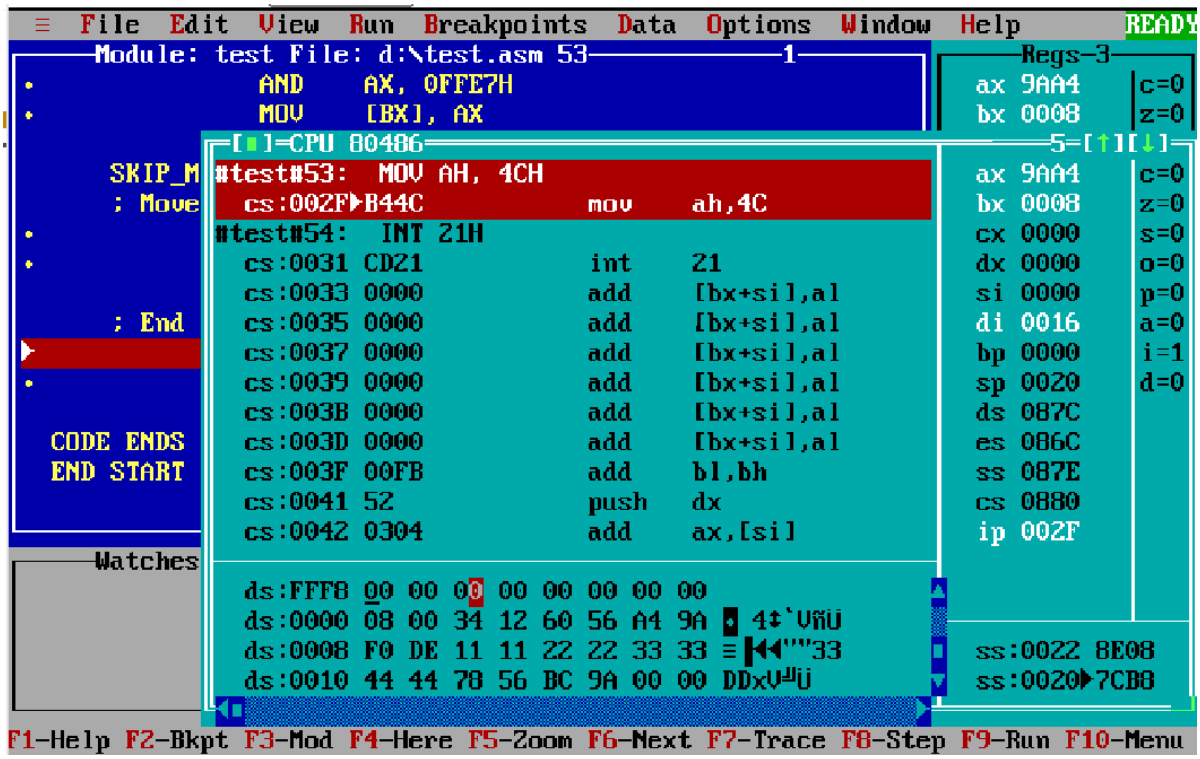
```
STACK SEGMENT
          DW 16 DUP(0)
STACK ENDS

CODE SEGMENT
              ASSUME CS:CODE, DS:DATA, SS:STACK

    START:
              MOV    AX, DATA
              MOV    DS, AX
              MOV    AX, STACK
              MOV    SS, AX
              MOV    SP, 20H

    ; Initialize BX to point to the start of the array
              MOV    BX, OFFSET NUMBERS
              ADD    BX, 2                          ; Skip the count

    ; Load the count into CX
              MOV    CX, [OFFSET NUMBERS]

    ; Initialize DI to point to the end of the array
              LEA    DI, ENDNUM

    PROCESS:
    ; Load the current number into AX
              MOV    AX, [BX]

    ; Check if D3 and D4 bits are set
              TEST   AX, 0018H                      ; D3 and D4 bits
              JZ     SKIP_MOVE                      ; If not set, skip

    ; Move the number to the end
              MOV    [DI], AX
              ADD    DI, 2

    ; Clear D3 and D4 bits
              AND    AX, 0FFE7H
              MOV    [BX], AX

    SKIP_MOVE:
    ; Move to the next number in the array
              ADD    BX, 2
              LOOP   PROCESS

    ; End the program
              MOV    AH, 4CH
              INT    21H

CODE ENDS
END START
```

3.3.2 观察到的现象

初始化结束后，可以在DS段看到初始信息：



运行结束后，可以发现D_3、D_4位为0的数据已经完成了修改，并且复制到了代码段的最后（5678H、9ABCH）：



四、收获

通过微机原理实验一，我深入理解了汇编程序的基本结构和分段概念，特别是如何在实际应用中使用8086指令系统的常见指令。此外，我还掌握了8086的寻址方式，尤其是在处理一维数值数组、字符串、二维数组和位操作时的应用。这次实验加深了我对汇编语言编程的认识，为我后续的学习打下了坚实的基础。