厦門大學

《汇编语言》实验报告

（二）

姓　　名　宋浩元

学　　号　37220232203808

学　院　信息学院

专　业　软件工程

2024 年 9 月

1 实验目的

1) 了解汇编语言程序(源程序)的基本组成部分；

2) 掌握寻址方式以及传送类指令的工作原理；

3) 进一步掌握使用 DEBUG 相应命令进行程序修改以及指令的调试与运行。

2 实验环境

Masm for Windows 继承环境

DOSBOX 0.74-3

3 实验内容

(1) 将例程 a 汇编连接， 生成可执行文件 Hello.exe； 利用 Debug 调试 Hello.exe，
完成下述任务：

(2) a) 指出字符串“Hello， World！ ”所在的段地址，以及段内起始地址；
b) 如果去掉字符串定义语句（黄色加亮） 部分，程序运行结果如何？

```
MESS DB ‘Hello World!’ , 0DH, 0AH , 24H
```

c) 如果去掉下述语句，程序汇编， 连接以及运行过程中会有什么变化？

```
SSEG SEGMENT PARA STACK
        DW 256 DUP(?)
SSEG ENDS
```

d) 下述三条语句执行之后， AX,DS,DX 寄存器的取值各位多少？代表什么含义？ 针对你所操作的计算机， MESS 的地址是多少？

```
BEGIN： MOV AX, DSEG
MOV DS, AX
MOV DX, OFFSET MESS
```

e) 如果去掉下述语句，程序执行结果有何变化？为什么？

```
MOV AH， 9
INT 21H
```

(2) 将例程 b 汇编连接， 生成可执行文件 Cal.exe； 利用 Debug 调试 Cal.exe，完成下述任务：

a) 给出 X， Y， Z 的内存地址；
b) 单步执行该程序，观察寄存器 AL 以及标志寄存器的变化；
c) 执行完成下述代码后， Z 和 Z1 两个内存变量里面的值各是多少？ 各自代表什么含义？

```
IDIV BL
MOV Z, AL
MOV Z1, AH
```

d) 程序中， IMUL 以及 IDIV 只有一个操作数，请问另一个操作数在哪里？
e) 结合例程 a, 在例程 b 中， 补充代码， 显示下述提示信息： ”the result is :” 然后再输出结果， 请给出补充完整的程序代码以及运行结果。
(3) 利用 DEBUG 调试程序给出下述指令的运行结果：

```
I)                    MOV SP,50FFH
II)                   MOV DX, SP;
III) MOV SI,3040H     ；  建议先利用 E 内存修改命令，将[SI+2]处存入
IV) MOV DL, [SI+2]    自定义
非零数据， 之后， 观察指令运行结果。
V) MOV BX,2030H
VI) MOV WORD PTR[BX+SI],34
VII) MOV DI, SI
VIII) MOV BP, 2[BX+DI]   ；  参考题
IV
IX) LEA AX, [BX+SI+3]    ；  参考题
IV
X) PUSH BX               ；  SP=?
XI) LAHF                 ；  Flag=?
XII) XCHG DH, BL
```

DH=? DL=?

(4) 写出实现下述要求的一条（或几条） 汇编指令，并利用 Debug 程序进行验证。

- 将一个立即数送入寄存器 BX；
- 将一个立即数送入段寄存器 DS；
- 将变址寄存器 DI 送入一个存储单元中；
- 从存储单元中取出一个数到段寄存器 ES 中；

(5) 使用 Debug 程序运行下述指令，给出运行结果(截屏)，如不能正常运行，请解释出错原因（可附上编译报错提示）， 如可以改正，请改正。

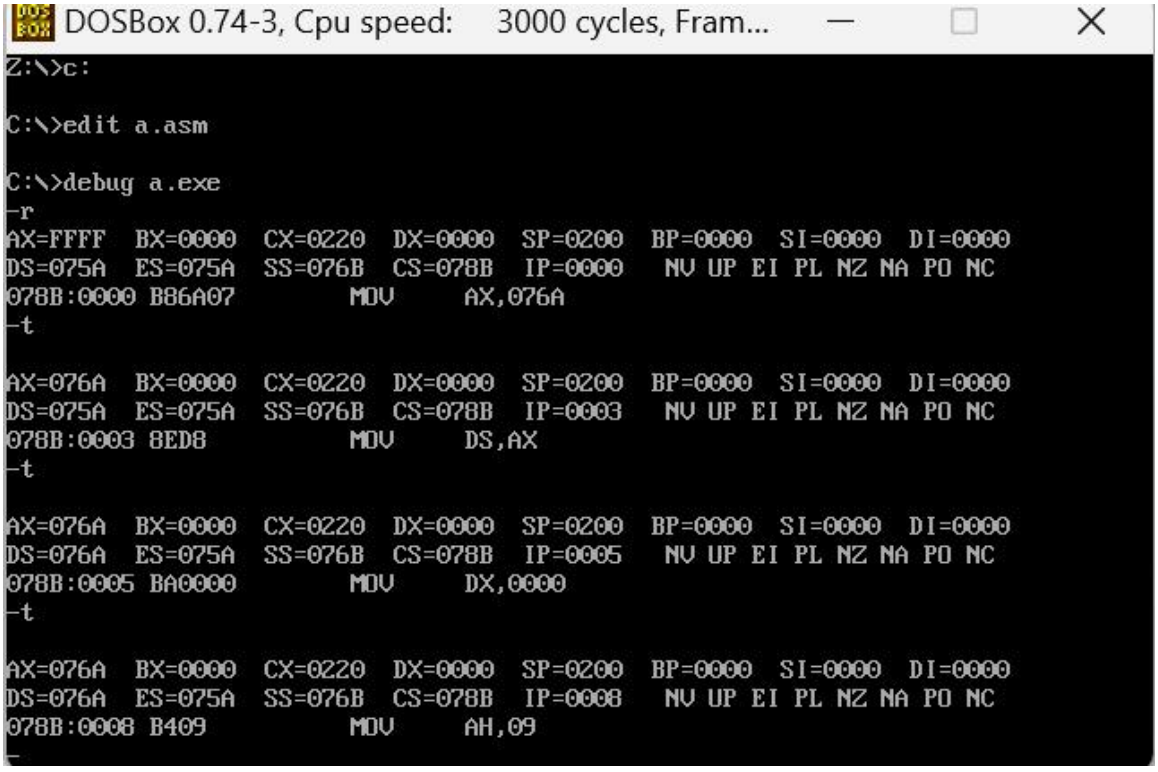
- MOV CX, EDH
- MOV [SI], 34H
- MOV [3000H], [2000H]
- MOV DS,100H

- MOV CS, [SI]
- MOV DS, ES

4 实验具体实现

实验一：

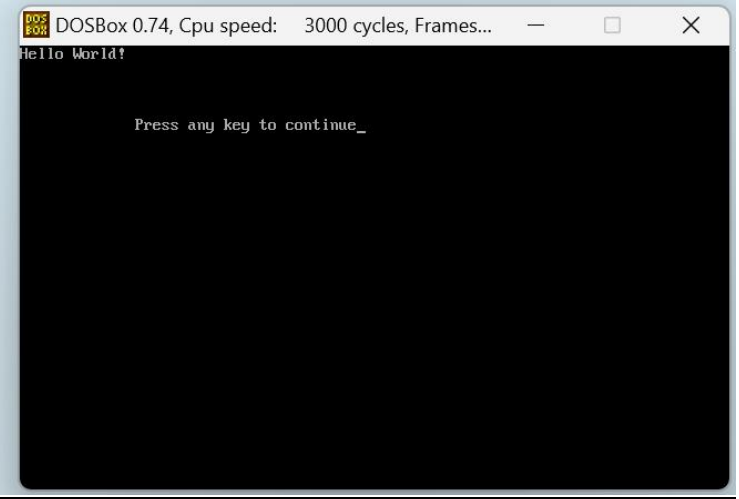
（a）段地址为 076A，段起始地址为 0000（即 offset mess）



（b）删除后运行出现了乱码情况

(3) 删除堆栈定义之后，在 link 的过程中出现 warning，但运行正常

```
01 DSEG    SEGMENT
02 MESS    DB      'Hello World!' , 0DH,0AH,24H
03 DSEG    ENDS
04
05
06
07 CSEG    SEGMENT
08     ASSUME  CS:CSEG,DS:DSEG
09 BEGIN:  MOV AX,DSEG
10     MOV DS,AX
11     MOV DX,OFFSET MESS
12     MOV AH,9
13
14     INT 21H
15     MOV AH,4CH
16     INT 21H
17 CSEG    ENDS
18     END BEGIN
19
20
```




```
Object filename [a.OBJ]:
Source listing  [NUL.LST]:
Cross-reference [NUL.CRF]:

  51738 + 464806 Bytes symbol space free

      0 Warning Errors
      0 Severe  Errors

C:\>link a.obj

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

Run File [A.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
LINK : warning L4021: no stack segment
```

（4）ax = 076A， DS = 076A， DX = 0000， mess 地址为 074A：0000

```
C:\>debug a.exe
-r
AX=FFFF  BX=0000  CX=0220  DX=0000  SP=0200  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=076B  CS=078B  IP=0000   NV UP EI PL NZ NA PO NC
078B:0000 B86A07        MOV     AX,076A
-t

AX=076A  BX=0000  CX=0220  DX=0000  SP=0200  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=076B  CS=078B  IP=0003   NV UP EI PL NZ NA PO NC
078B:0003 8ED8          MOV     DS,AX
-t

AX=076A  BX=0000  CX=0220  DX=0000  SP=0200  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=076B  CS=078B  IP=0005   NV UP EI PL NZ NA PO NC
078B:0005 BA0000        MOV     DX,0000
-t

AX=076A  BX=0000  CX=0220  DX=0000  SP=0200  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=076B  CS=078B  IP=0008   NV UP EI PL NZ NA PO NC
078B:0008 B409          MOV     AH,09
```

（5）去掉后程序将不会显示“Hello, World!”消息，并结束运行

实验二：

（1）x：076A:0000 y:076A:0001 z:076A:0002

```
C:\>debug b.exe
-r
AX=FFFF  BX=0000  CX=0050  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=0769  CS=076B  IP=0000   NV UP EI PL NZ NA PO NC
076B:0000 B86A07        MOV     AX,076A
-       t

AX=076A  BX=0000  CX=0050  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=0769  CS=076B  IP=0003   NV UP EI PL NZ NA PO NC
076B:0003 8ED8          MOV     DS,AX
-d 076A:0000
076A:0000  05 04 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  B8 6A 07 8E D8 A0 00 00-02 06 01 00 B3 08 F6 EB   .j..............
076A:0020  8A 1E 00 00 B7 00 2B C3-B3 02 F6 FB A2 02 00 88   ......+.........
076A:0030  26 03 00 A0 02 00 B4 00-B3 0A F6 F3 8B D0 81 C2   &...............
076A:0040  36 30 B4 02 CD 21 8A D6-B4 02 CD 21 B4 4C CD 21   60...!.....!.L.!
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-
```

(2)

MOV AL, X 后，AL = 5

ADD AL, Y 后，AL = 9（无进位，CF=0）

IMUL BL 后（BL=8），AX = 72（AL=72, AH=0），OF=0（无溢出）

SUB AX, BX 后（BX=0005），AX = 67（AL=67, AH=0），CF=0（无借位）

IDIV BL 后（BL=2），AL = 33，AH = 1，ZF=0，SF=0

DIV BL（BL=10）后，AL = 3，AH = 3，ZF=0，SF=0

(3)

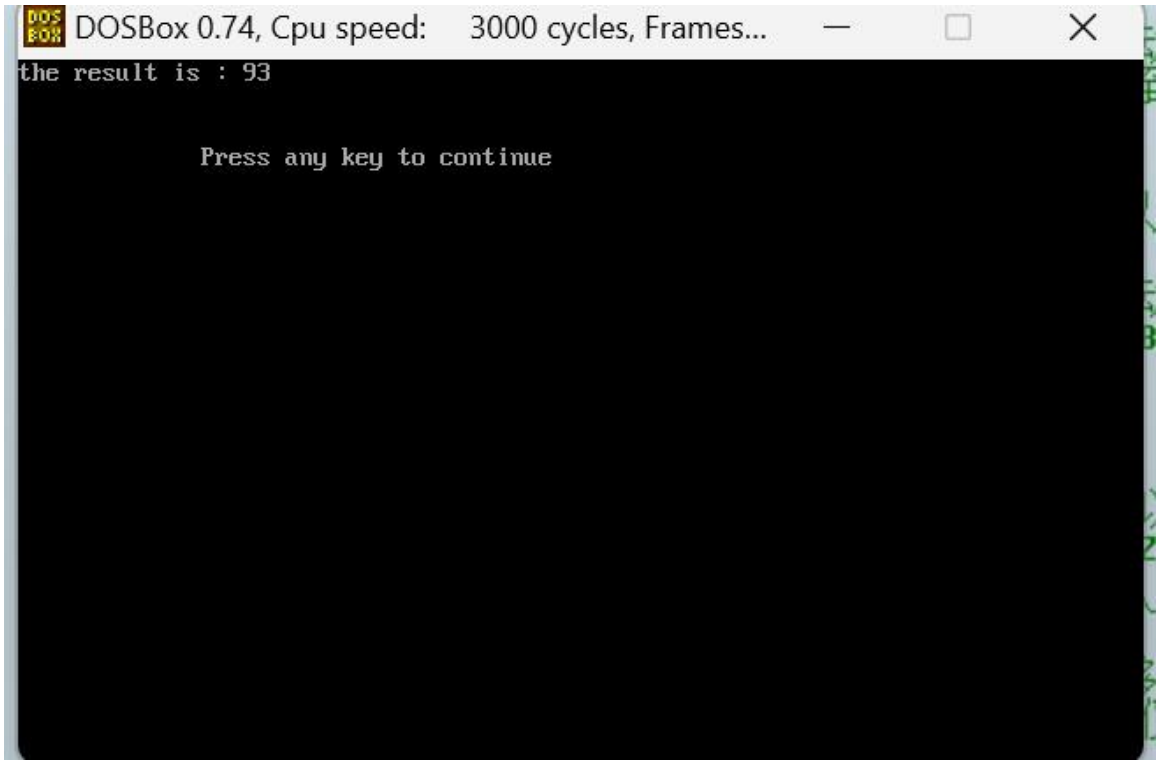
Z 的值是 33，代表 67 除以 2 的商。

Z1 的值是 1，代表 67 除以 2 的余数。

(4)

另一个操作数默认在 ax 寄存器中。

(5)



实验三：

I) MOV SP,50FFH

结果：SP = 50FFH

II) MOV DX, SP

结果：DX = 50FFH

III) MOV SI,3040H

结果：SI = 3040H

IV) MOV DL, [SI+2]

使用 E 命令在内存地址 3042H 处设置一个 45H。

结果：DL = 45H

V) MOV BX,2030H

- 结果：BX = 2030H

VI) MOV WORD PTR[BX+SI],34

- 计算地址：BX + SI = 2030H + 3040H = 5070H
- 结果：在内存地址 5070H 和 5071H 处存储值 0034H

VII) MOV DI, SI

- 结果：DI = SI = 3040H

VIII) MOV BP, 2[BX+DI]

- 在内存地址 5072H（即 BX+DI+2 = 2030H + 3040H + 2 = 5072H）处设置一个 45H。
- 结果：BP = 0045

IX) LEA AX, [BX+SI+3]

- 计算地址：BX + SI + 3 = 2030H + 3040H + 3 = 5073H
- 结果：AX = 5073H（LEA 指令加载有效地址到目标寄存器）

X) PUSH BX

- SP 递减 2，并将 BX 的值压入堆栈。
- 新的 SP 值 = 50FFH - 2 = 50FDH

XI) LAHF

- 将标志寄存器的低 8 位（FLAGS 的低 8 位）加载到 AH 寄存器。
- 结果取决于之前的操作，但假设没有改变任何标志位，AH 可能包含之前的标志状态。

XII) XCHG DH, BL

- 结果：DH = 30H, BL = 50H。

实验四：

（1）从图一可以看出 bx 由 0000 变成了 1234

（2）从图一和图二可以看出 ax 由 0000 变成了 1234

（3）从图二可以看出 ds 由 073F 变成了 1234

（4）从图三可以看出 es 由 073F 变成了 1234

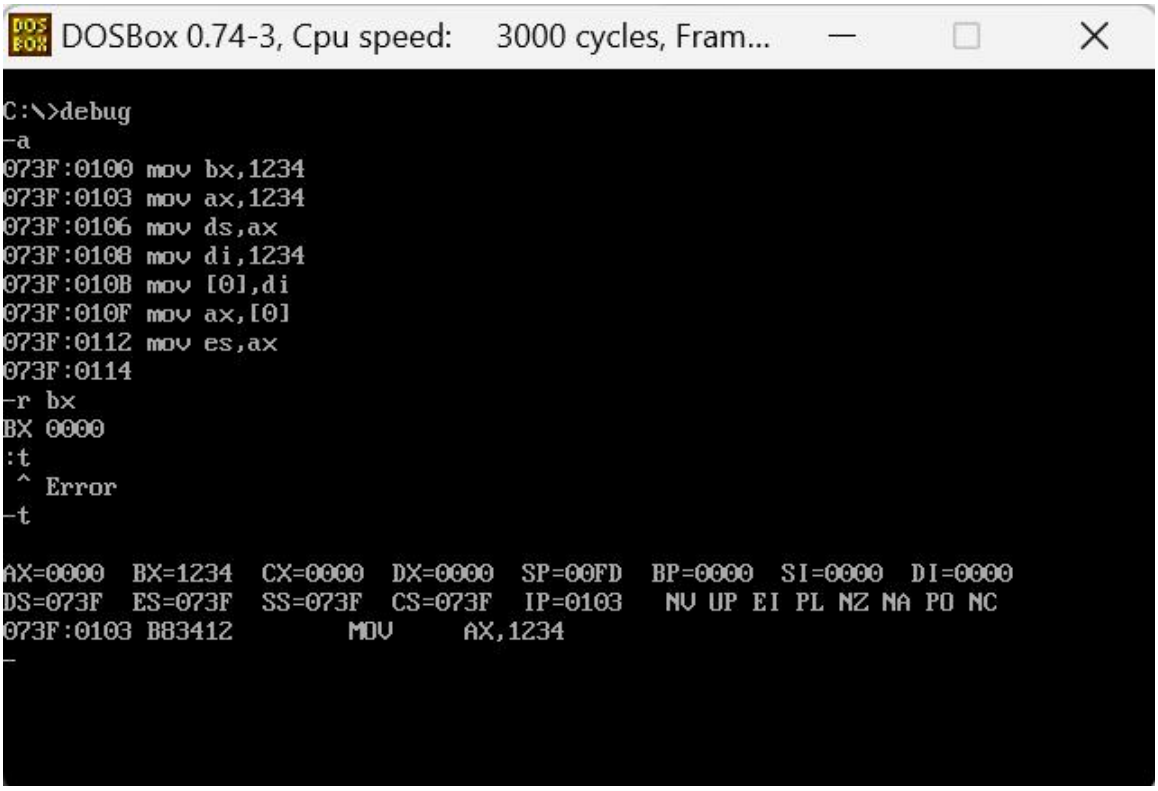

DOSBox 0.74-3, Cpu speed: 3000 cycles, Fram...
C:\>debug
-a
073F:0100 mov bx,1234
073F:0103 mov ax,1234
073F:0106 mov ds,ax
073F:0108 mov di,1234
073F:010B mov [0],di
073F:010F mov ax,[0]
073F:0112 mov es,ax
073F:0114
-r bx
BX 0000
:t
^ Error
-t
AX=0000 BX=1234 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0103 NV UP EI PL NZ NA PO NC
073F:0103 B83412 MOV AX,1234
-

表 1 图一

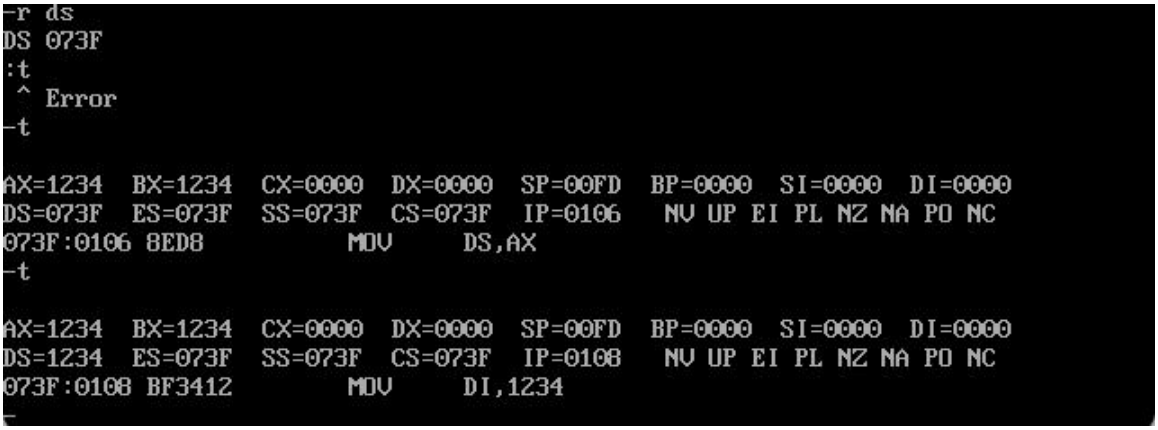

-r ds
DS 073F
:t
^ Error
-t
AX=1234 BX=1234 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0106 NV UP EI PL NZ NA PO NC
073F:0106 8ED8 MOV DS,AX
-t
AX=1234 BX=1234 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=1234 ES=073F SS=073F CS=073F IP=0108 NV UP EI PL NZ NA PO NC
073F:0108 BF3412 MOV DI,1234
-

表 2 图二

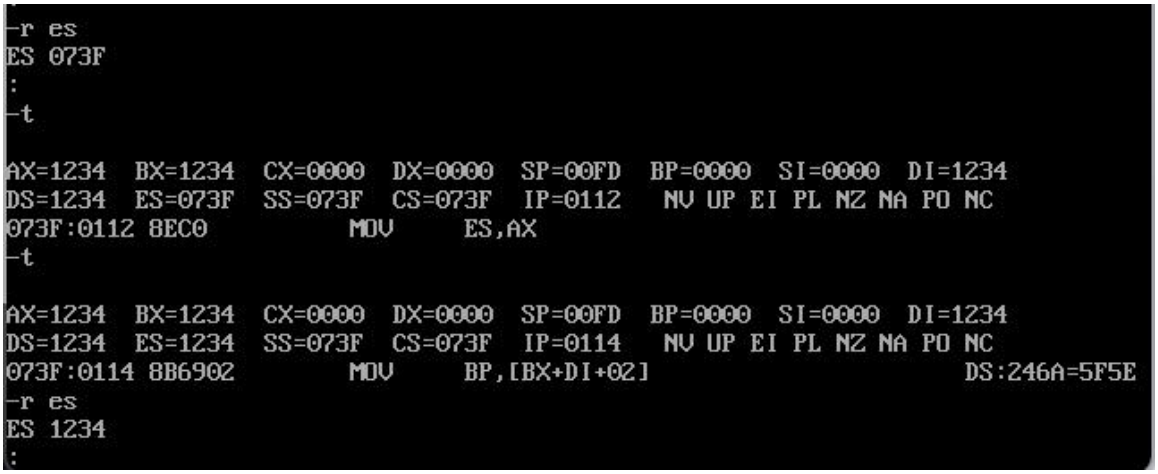


表 3 图三

实验五：

（先进入 debug 写好代码）

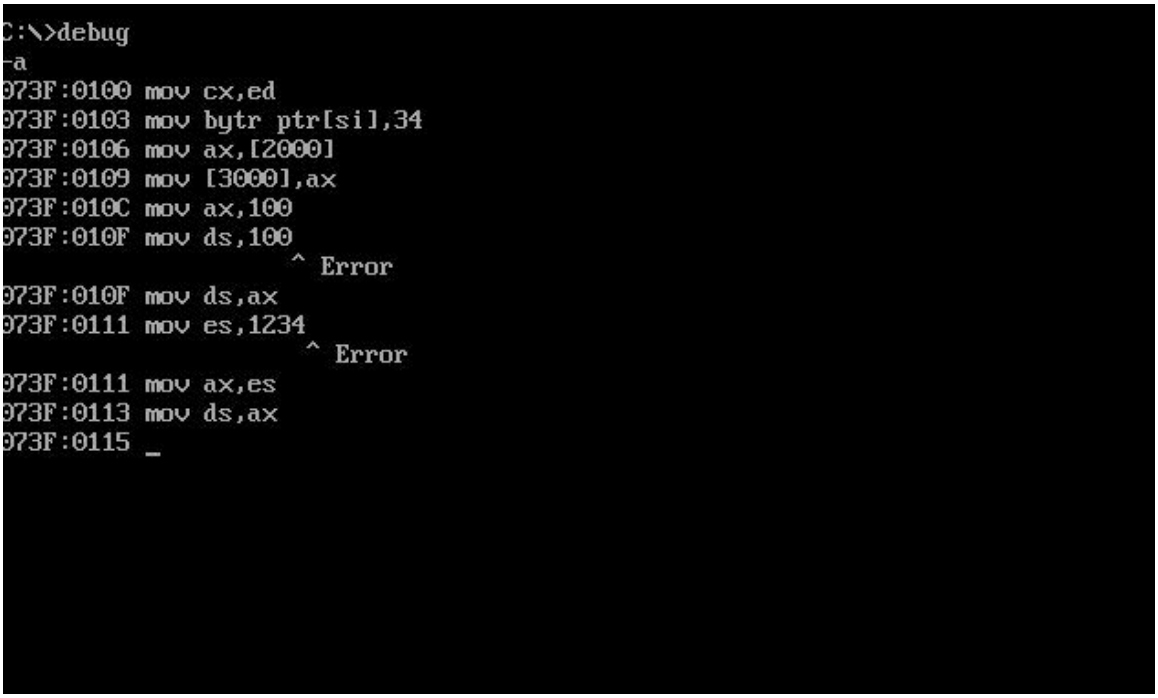


表 4 图四

（1）cx 从 0000 变成 00ed

表 5 图五

（2）需要输入 byte ptr 才不会报错，可以发现[si]中的值已经变成 34h

表 6 图六

（3）需要中转，不能直接对段寄存器赋值

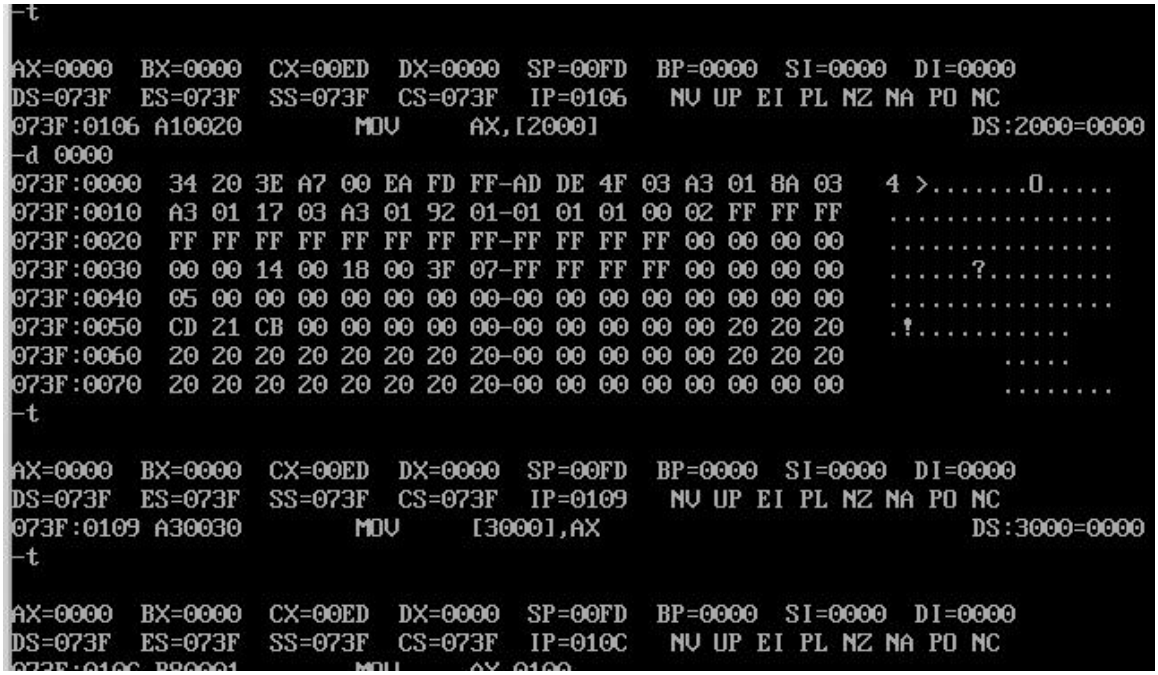


表 7 图七

（4）与（3）一样需要中转

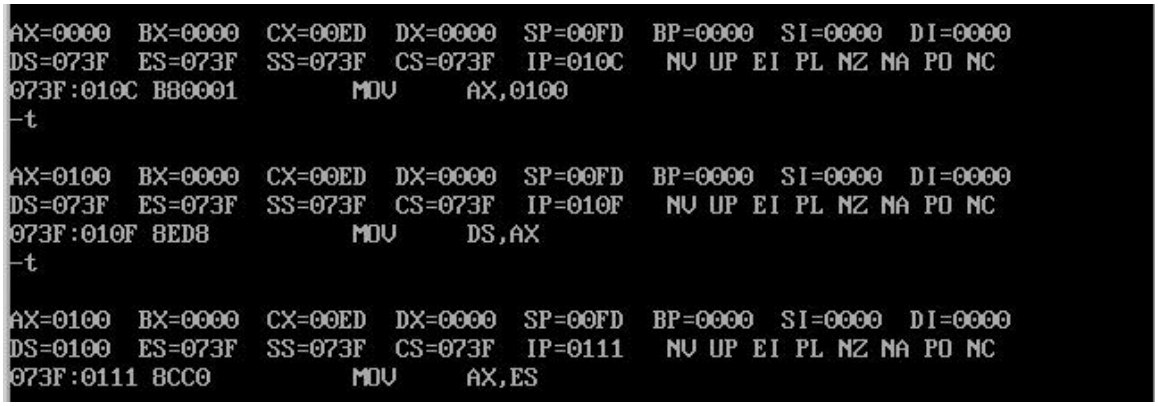


表 8 图八

（5）如果输入第四条命令，发现 cs 的值被改变，但不是期望的 0000，而是 F000，并且程序会陷入崩溃/死循环？

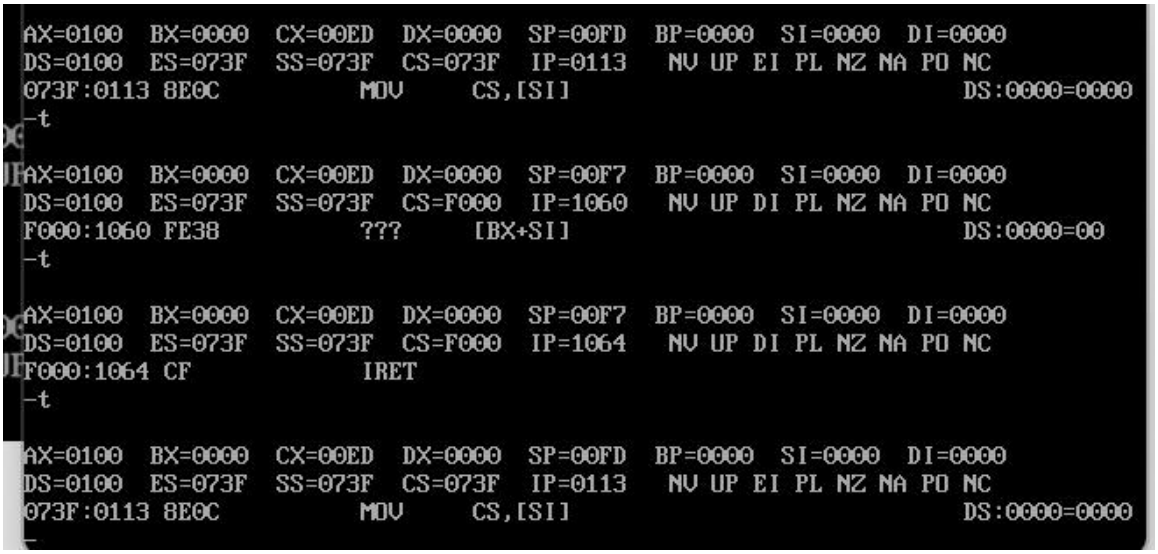
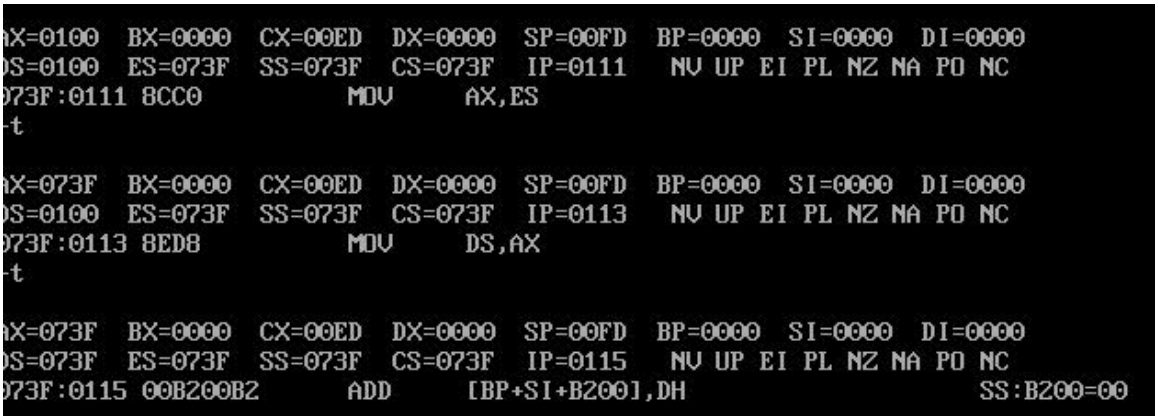


表 9 图九

（6）同样需要寄存器中转



5 实验分析与总结.....

（1）学会了掌握寻址方式以及传送类指令的工作原理；如立即数寻址，寄存器寻址，直接寻址，寄存器间接寻址，寄存器相对寻址，基址变址寻址，相对基址变址寻址。

（2）进一步掌握了使用 DEBUG 相应命令进行程序修改以及指令的调试与运行。

（3）了解了汇编语言程序(源程序)的基本组成部分，汇编程序主要由数据段，代码段，堆栈段，段定义和初始化以及指令等组成。

（4）熟悉了集成环境和模拟 dos 环境的使用，了解部分特性。