

# 华中科技大学

## 课程实验报告

课程名称: 汇编语言

专业班级:

学 号:

姓 名:

指导教师:

报告日期: 2019 年 10 月 25 日

网络安全学院

# 实验一 熟悉汇编调试环境、查看寄存器和内存内容

## 1.1 实验目的与内容

### 1.1.1 实验目的

- 1、熟悉汇编语言调试环境。
- 2、熟悉 Debug 查看寄存器和内存内容。
- 3、跟踪程序运行，观察 CS 和 IP 寄存器的变化。
- 4、编程、编译、链接、跟踪以及源代码调错。

### 1.1.2 实验内容

请参考教师发放的实验手册和 ppt。

### 1.1.3 实验工具

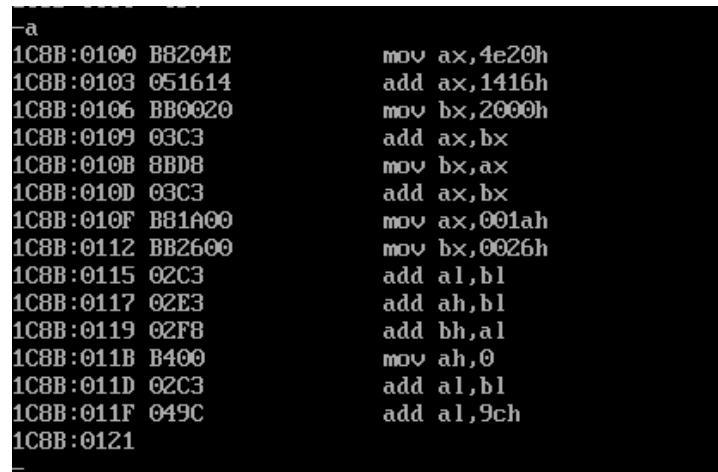
Dosbox0.74。

## 1.2 实验过程

### 1.2.1 任务一

(1) 实验任务 1

1) 过程截图：



```
-a
1C8B:0100 B8204E      mov ax,4e20h
1C8B:0103 051614      add ax,1416h
1C8B:0106 BB0020      mov bx,2000h
1C8B:0109 03C3      add ax,bx
1C8B:010B 8BD8      mov bx,ax
1C8B:010D 03C3      add ax,bx
1C8B:010F B81A00      mov ax,001ah
1C8B:0112 BB2600      mov bx,0026h
1C8B:0115 02C3      add al,bl
1C8B:0117 02E3      add ah,bl
1C8B:0119 02F8      add bh,al
1C8B:011B B400      mov ah,0
1C8B:011D 02C3      add al,bl
1C8B:011F 049C      add al,9ch
1C8B:0121
```

```

-r
AX=0000 BX=0000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0100 NU UP DI PL NZ NA PO NC
1C8B:0100 B8204E
MOV AX,4E20h
-t
AX=4E20 BX=0000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0103 NU UP DI PL NZ NA PO NC
1C8B:0103 051614
ADD AX,1416h
Trace Interrupt
-t
AX=5235 BX=0000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0105 NU UP DI PL NZ NA PE NC
1C8B:0105 BB0020
MOV BX,2000h
Trace Interrupt
-t
AX=6236 BX=2000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0109 NU UP DI PL NZ NA PE NC
1C8B:0109 03C3
ADD AX,BX
Trace Interrupt
-t
AX=8236 BX=2000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=010B NU UP DI PL NZ NA PE NC
1C8B:010B 8BD8
MOV BX,AX
Trace Interrupt
-
AX=8236 BX=8236 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=010D NU UP DI NG NZ NA PE NC
1C8B:010D 03C3
ADD AX,BX
Trace Interrupt
-t
AX=046C BX=8236 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=010F NU UP DI PL NZ NA PE CY
1C8B:010F B81A00
MOV AX,001Ah
Trace Interrupt
-t
AX=001A BX=8236 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0112 NU UP DI PL NZ NA PE CY
1C8B:0112 BB2600
MOV BX,0026h
Trace Interrupt
-t
AX=001A BX=0026 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0115 NU UP DI PL NZ NA PE CY
1C8B:0115 02C3
ADD AL,BL
Trace Interrupt
-t
AX=0040 BX=0026 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0117 NU UP DI PL NZ AC PO NC
1C8B:0117 02E3
ADD AH,BL
Trace Interrupt
-
AX=2640 BX=0026 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0119 NU UP DI PL NZ NA PO NC
1C8B:0119 02F8
ADD BH,AL
Trace Interrupt
-t
AX=2640 BX=0026 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=011B NU UP DI PL NZ NA PO NC
1C8B:011B B400
MOV AH,00h
Trace Interrupt
-t
AX=0040 BX=4026 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=011D NU UP DI PL NZ NA PO NC
1C8B:011D 02C3
ADD AL,BL
Trace Interrupt
-t
AX=0056 BX=4026 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=011F NU UP DI PL NZ NA PE NC
1C8B:011F 049C
ADD AL,9Ch
Trace Interrupt
-t
AX=0002 BX=4026 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0121 NU UP DI PL NZ AC PO CY
1C8B:0121 0000
ADD [BX+SI],AL
Trace Interrupt
-

```

## 2) 遇到的问题及解决方法:

### ①问题: DOSBox 显示窗口比例调整

解决方法: 修改 DOSBox 0.74 Options.bat 文件中参数 windowresolution 和

output（如将 windowresolution 设为 960x720，output 设为 opengl，则窗口会以 960 × 720 大小显示）。值得注意的是第一个参数的乘号为“x”。

## （2）实验任务 2

### 1) 过程截图：

```
-a 2000:0
2000:0000 B80100      mov ax,1
2000:0003 03C0      add ax,ax
2000:0005 EA03000020    jmp 2000:0003
2000:000A
-r
AX=2000 BX=0000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=2000 IP=0003  NU UP DI PL NZ NA PE NC
2000:0003 03C0      ADD     AX,AX
-r ip
IP 0003
:0000
-r
AX=2000 BX=0000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=2000 IP=0003  NU UP DI PL NZ NA PE NC
2000:0000 B80100      MOV     AX,0001h
-aa
```

```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Pro...
-r
AX=2000 BX=0000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=2000 IP=0003  NU UP DI PL NZ NA PE NC
2000:0000 B80100      MOV     AX,0001h
-t
AX=0001 BX=0000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=2000 IP=0003  NU UP DI PL NZ NA PE NC
2000:0003 03C0      ADD     AX,AX
Trace Interrupt
-t
AX=0002 BX=0000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=2000 IP=0003  NU UP DI PL NZ NA PO NC
2000:0005 EA03000020    JMP     2000:0003
Trace Interrupt
-t
AX=0002 BX=0000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=2000 IP=0003  NU UP DI PL NZ NA PO NC
2000:0003 03C0      ADD     AX,AX
Trace Interrupt
-t
AX=0004 BX=0000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=2000 IP=0003  NU UP DI PL NZ NA PO NC
2000:0005 EA03000020    JMP     2000:0003
Trace Interrupt
-
```

```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Pro...
AX=0001 BX=0000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=2000 IP=0003  NU UP DI PL NZ NA PO NC
2000:0005 EA03000020    JMP     2000:0003
Trace Interrupt
-t
AX=0001 BX=0000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=2000 IP=0003  NU UP DI PL NZ NA PO NC
2000:0003 03C0      ADD     AX,AX
Trace Interrupt
-t
AX=0002 BX=0000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=2000 IP=0003  NU UP DI PL NZ NA PO NC
2000:0005 EA03000020    JMP     2000:0003
Trace Interrupt
-t
AX=0002 BX=0000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=2000 IP=0003  NU UP DI PL NZ NA PO NC
2000:0003 03C0      ADD     AX,AX
Trace Interrupt
-t
AX=0004 BX=0000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=2000 IP=0003  NU UP DI PL NZ NA PO NC
2000:0005 EA03000020    JMP     2000:0003
Trace Interrupt
-a
```

### 2) 遇到的问题及解决方法：

- ①问题：将内存修改后-t 命令单步执行，会出现程序一直循环的情况  
原因：IP 没有指向需要运行代码的目标地址

解决方法：需要先通过-r 命令修改 CS 和 IP 的值，使其指向待运行代码的地址，方能成功单步执行

### (3) 实验任务 3

#### 1) 过程截图：

```
-d fff0:0 ff
FFF0:0000 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
***Duplicate Line(s)***
FFF0:0050 00 00 00 CF 00 50 B0 20-E6 20 58 CF 00 00 00 00 ...0.P0 f XD....
FFF0:0060 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
***Duplicate Line(s)***
FFF0:00F0 EA C0 12 00 F0 30 31 2F-30 31 2F 39 32 00 FC 55 j@..p01/01/92.iU
-

-d fff0:0 ff
FFF0:0000 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
***Duplicate Line(s)***
FFF0:0050 00 00 00 CF 00 50 B0 20-E6 20 58 CF 00 00 00 00 ...0.P0 f XD....
FFF0:0060 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
***Duplicate Line(s)***
FFF0:00F0 EA C0 12 00 F0 30 31 2F-30 31 2F 39 32 00 FC 55 j@..p01/01/92.iU
-e fff0:00f5
FFF0:00F5 30.32 31.35 2F.2f 30.31 31.30 2F.2f 39.31 32.39
-d fff0:0 ff
FFF0:0000 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
***Duplicate Line(s)***
FFF0:0050 00 00 00 CF 00 50 B0 20-E6 20 58 CF 00 00 00 00 ...0.P0 f XD....
FFF0:0060 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
***Duplicate Line(s)***
FFF0:00F0 EA C0 12 00 F0 30 31 2F-30 31 2F 39 32 00 FC 55 j@..p01/01/92.iU
```

#### 2) 实验结果：

由-d 命令查找到生产日期应该为 fff0: 00f5 及其后 8 个内存单元的 01/01/92，通过-e 命令对这几个单元修改后再次查看发现内存单元内的数据并没有改变。

经查询得知此处几个内存空间位于 ROM 地址空间，此处的数据是只读的，无法修改，因此在修改后也并不发生变化。

### (4) 实验任务 4:

#### 1) 过程截图：

```
-e b810:0000 01 01 02 02 03 03 04 04
-d b810:0000
B810:0000 20 07 20 07 20 07 20 07-20 07 20 07 20 07 20 07 . . . . .
***Duplicate Line(s)***
B810:0040 2D 07 20 07 20 07 20 07-20 07 20 07 20 07 20 07 -. . . . .
B810:0050 20 07 20 07 20 07 20 07-20 07 20 07 20 07 20 07 . . . . .
B810:0060 20 07 20 07 43 07 58 07-3D 07 30 07 30 07 30 07 . .C.X.=.0.0.0.
B810:0070 42 07 20 07 20 07 43 07-53 07 3D 07 32 07 30 07 B. . .C.S.=.2.0.
-
```

#### 2) 实验结果：

向内存从 B8100 开始的单元写入数据后查看发现原始数据并未修改。

原因与上题类似，B8100 也为 ROM 地址空间（范围为 C0000 到 FFFFF），此处数据为只读的，无法修改。

而通过尝试，发现在主存储器地址空间（00000 到 9FFFF）与显存地址空间（A0000 到 BFFFF）数据是可以修改的。（见下图）

```
-e 0100:0 01 01 02 02 03 03 04 04
-d 0100:0
0100:0000 01 01 02 02 03 03 04 04-00 00 00 00 00 00 00 .....
0100:0010 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
***Duplicate Line(s)***
0100:0070 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
-

-e A000:0
A000:0000 00. 00.01 00.01 00.02 00.02 00.03 00.03 00.04
A000:0008 00.04
-d A000:0
A000:0000 00 01 01 02 02 03 03 04-04 00 00 00 00 00 00 .....
A000:0010 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
***Duplicate Line(s)***
A000:0070 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
-
```

## 1.2.2 任务二

(1) 过程截图：

```
-r
AX=0000 BX=0000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0100 NU UP DI PL NZ NA PO NC
1C8B:0100 0000 ADD [BX+SI],AL
-a
1C8B:0100 B82211 mov ax,1122h
1C8B:0103 BB4433 mov bx,3344h
1C8B:0106 B96655 mov cx,5566h
1C8B:0109 BC0020 mov sp,2000h
1C8B:010C 50 push ax
1C8B:010D 53 push bx
1C8B:010E 51 push cx
1C8B:010F 58 pop ax
1C8B:0110 59 pop cx
1C8B:0111
-
```

```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG32
-r
AX=0000 BX=0000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0100 NU UP DI PL NZ NA PO NC
1C8B:0100 B82211 MOV AX,1122h
-t
AX=1122 BX=0000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0103 NU UP DI PL NZ NA PO NC
1C8B:0103 BB4433 MOV BX,3344h
Trace Interrupt
-t
AX=1122 BX=3344 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0106 NU UP DI PL NZ NA PO NC
1C8B:0106 B96655 MOV CX,5566h
Trace Interrupt
-t
AX=1122 BX=3344 CX=5566 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0109 NU UP DI PL NZ NA PO NC
1C8B:0109 BC0020 MOV SP,2000h
Trace Interrupt
-t
AX=1122 BX=3344 CX=5566 DX=0000 SP=2000 BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=010C NU UP DI PL NZ NA PO NC
1C8B:010C 50 PUSH AX
Trace Interrupt
-a_
```

```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG32
AX=1122 BX=3344 CX=5566 DX=0000 SP=1FFC BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=010E NU UP DI PL NZ NA PO NC
1C8B:010E 51 PUSH CX
Trace Interrupt
-t
AX=1122 BX=3344 CX=5566 DX=0000 SP=1FFA BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=010F NU UP DI PL NZ NA PO NC
1C8B:010F 58 POP AX
Trace Interrupt
-t
AX=5566 BX=3344 CX=5566 DX=0000 SP=1FFC BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0110 NU UP DI PL NZ NA PO NC
1C8B:0110 59 POP CX
Trace Interrupt
-t
AX=5566 BX=3344 CX=3344 DX=0000 SP=1FFE BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0111 NU UP DI PL NZ NA PO NC
1C8B:0111 0000 ADD [BX+SI],AL
Trace Interrupt
-t
AX=5566 BX=3344 CX=3344 DX=0000 SP=1FFE BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0113 NU UP DI PL NZ NA PE NC
1C8B:0113 0000 ADD [BX+SI],AL
Trace Interrupt
-a
```

(2) 题目问题回答：执行后，AX 内容为 5566H，BX 内容为 3344H，CX 内容为 3344H，SP 内容为 1FFE。其中，SP 的值在压栈的时候在减小，出栈的时候在增加。

(3) 老师留下的问题回答：

①代码的输入内存地址：逻辑地址为 1CBB:0100 到 1CBB:0110 的内存中

②如何确定：通过-a 命令输入汇编代码时通过每行前面的逻辑地址确定的，即代码段寄存器和指令指示器确定。

③内存目标地址原来的数据：AX，BX，CX，DX 初始值均为 0000，

④如何执行输入代码：-t 命令（若 CS 和 IP 的初始值并不指向输入的代码，则还需先通过-r 命令修改其值，使其能指向代码的内存地址）。

⑤单步执行后目标存储器内容的变化：mov 命令分别修改了 AX，BX，CX 的值，push 命令入栈仅改变了堆栈指示器 SP 的值（减小），pop 命令出栈同时改变了 SP 的值，以及出栈数据对应的寄存器（AX，CX）的值。

### 1.2.3 任务三

(1) 过程截图：

```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG32
-a CS:0000
1CBB:0000 8BC3      mov ax,bx
1CBB:0002 2BC0      sub ax,ax
1CBB:0004 FFE0      jmp ax
1CBB:0006
-r
AX=0000 BX=3344 CX=5566 DX=0000 SP=1FFE BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0000 NU UP DI PL ZR NA PE NC
1CBB:0000 8BC3      MOV     AX,BX
-t
AX=3344 BX=3344 CX=5566 DX=0000 SP=1FFE BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0002 NU UP DI PL ZR NA PE NC
1CBB:0002 2BC0      SUB     AX,AX
Trace Interrupt
-t
AX=0000 BX=3344 CX=5566 DX=0000 SP=1FFE BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0004 NU UP DI PL ZR NA PE NC
1CBB:0004 FFE0      JMP     AX
Trace Interrupt
-t
AX=0000 BX=3344 CX=5566 DX=0000 SP=1FFE BP=0000 SI=0000 DI=0000
DS=1C8B ES=1C8B SS=1C8B CS=1C8B IP=0006 NU UP DI PL ZR NA PE NC
1CBB:0006 8BC3      MOV     AX,BX
Trace Interrupt
-
```

(2) 题目问题回答：执行 3 条指令后（各执行 1 次），IP 被修改 3 次。每次执行一条语句便会修改 1 次 IP 的值。最后 IP 的值为运行 3 条指令前 IP 的值（此处为 0000）。

(3) 老师留下的问题回答：

①代码的输入内存地址：逻辑地址为 1CBB:0000 到 1CBB:0006 的内存中

②如何确定：通过 -a 命令输入汇编代码时通过每行前面的逻辑地址确定的，即代码段寄存器和指令指示器确定。

③内存目标地址原来的数据：(AX)=0000, (BX)=3344, (CX)=5566, (DX)=0000

④如何执行输入代码：-t 命令（若 CS 和 IP 的初始值并不指向输入的代码，则还需先通过 -r 命令修改其值，使其能指向代码的内存地址）。

⑤单步执行后目标存储器内容的变化：mov 命令将 BX 中的值赋给了 AX，sub 命令将 AX 中的值置零，jmp 命令使 IP 又重新直到 mov 命令处。此时目标存储器的值和初始值相同。

(4) 遇到的问题及解决：

问题：不知道为何 jmp ax 命令便可又回到 mov 命令处继续执行。

原因：jmp ax 是修改 IP 的值为 AX 中的值，该代码恰好起始偏移地址为 0000，使 jmp 跳转后恰好使 IP 指向命令的起始处。

## 1.2.4 任务四

(1) 过程截图：



```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX
C:\>masm
Microsoft (R) Macro Assembler Version 5.00
Copyright (C) Microsoft Corp 1981-1985, 1987. All rights reserved.

Source filename [.ASM]: ye28
Object filename [ye28.OBJ]:
Source listing [NUL.LST]:
Cross-reference [NUL.CRF]:

51756 + 464788 Bytes symbol space free

0 Warning Errors
0 Severe Errors

C:\>link

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

Object Modules [.OBJ]: ye28
Run File [YE28.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:

C:\>_
```

```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG32

C:\>debug32 ye28.exe
Debug32 - Version 1.0 - Copyright (C) Larson Computing 1994

CPU = 486, Real Mode, Id/Step = 0402, A20 disabled
-r
AX=0000 BX=0000 CX=00FC DX=0000 SP=00C8 BP=0000 SI=0000 DI=0000
DS=1C95 ES=1C95 SS=1CA6 CS=1CB3 IP=0000  NU UP DI PL NZ NA PO NC
1CB3:0000 B8A51C      MOV     AX,1CA5h
-t
AX=1CA5 BX=0000 CX=00FC DX=0000 SP=00C8 BP=0000 SI=0000 DI=0000
DS=1C95 ES=1C95 SS=1CA6 CS=1CB3 IP=0003  NU UP DI PL NZ NA PO NC
1CB3:0003 8ED8      MOV     DS,AX
Trace Interrupt
-t
AX=1CA5 BX=0000 CX=00FC DX=0000 SP=00C8 BP=0000 SI=0000 DI=0000
DS=1CA5 ES=1C95 SS=1CA6 CS=1CB3 IP=0005  NU UP DI PL NZ NA PO NC
1CB3:0005 B93200    MOV     CX,0032h
Trace Interrupt
-t
AX=1CA5 BX=0000 CX=0032 DX=0000 SP=00C8 BP=0000 SI=0000 DI=0000
DS=1CA5 ES=1C95 SS=1CA6 CS=1CB3 IP=0007  NU UP DI PL NZ NA PO NC
1CB3:0007 B80000    MOV     AX,0000h
Trace Interrupt
-
```

```
.....
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG32
1CB3:0012 49      DEC     CX
Trace Interrupt
-t
AX=09C4 BX=0065 CX=0000 DX=0000 SP=00C8 BP=0000 SI=0000 DI=0000
DS=1CA5 ES=1C95 SS=1CA6 CS=1CB3 IP=0013  NU UP DI PL ZR NA PE NC
1CB3:0013 75F9      JNZ     Short 000E ;no
Trace Interrupt
-t
AX=09C4 BX=0065 CX=0000 DX=0000 SP=00C8 BP=0000 SI=0000 DI=0000
DS=1CA5 ES=1C95 SS=1CA6 CS=1CB3 IP=0015  NU UP DI PL ZR NA PE NC
1CB3:0015 A30000    MOV     [0000],AX
Trace Interrupt
-t
AX=09C4 BX=0065 CX=0000 DX=0000 SP=00C8 BP=0000 SI=0000 DI=0000
DS=1CA5 ES=1C95 SS=1CA6 CS=1CB3 IP=0017  NU UP DI PL ZR NA PE NC
1CB3:0017 B44C      MOV     AH,4Ch
Trace Interrupt
-t
AX=1C44 BX=0065 CX=0000 DX=0000 SP=00C8 BP=0000 SI=0000 DI=0000
DS=1CA5 ES=1C95 SS=1CA6 CS=1CB3 IP=0019  NU UP DI PL ZR NA PE NC
1CB3:0019 CD21      INT     21h ;End Program
Trace Interrupt
-t
Program terminated RC =196
```

(3) 实验结果：该源代码的功能为求 1 开始连续 50 奇数的和。最后所求的结果为 09C4H（即 2500）。

(2) 老师留下的问题回答：

①汇编出错提示及原因：

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

Source filename [.ASM]: ye28
Object filename [ye28.OBJ]:
Source listing [NUL.LST]:
Cross-reference [NUL.CRF]:
ye28.ASM(13): Out of memory
```

原因：13 行代码中半角逗号“,”打成了全角逗号“，”。

②删除.386 的错误提示：显示有三行对齐/组合方式不当

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

Source filename [.ASM]: ye28
Object filename [ye28.OBJ]:
Source listing [NUL.LST]:
Cross-reference [NUL.CRF]:
ye28.ASM(2): error A2025: Improper align/combine type
ye28.ASM(5): error A2025: Improper align/combine type
ye28.ASM(8): error A2025: Improper align/combine type

51756 + 464788 Bytes symbol space free

0 Warning Errors
3 Severe Errors
```

③删除 USE16 的错误提示：竟然没有错误提示！

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

Source filename [.ASM]: ye28
Object filename [ye28.OBJ]:
Source listing [NUL.LST]:
Cross-reference [NUL.CRF]:

51756 + 464788 Bytes symbol space free

0 Warning Errors
0 Severe Errors
```

(3) 遇到的问题：

1) 删除 USE16 在 masm 过程中并未报错

2) 源程序中经 debug32 反汇编后，标识符 NEXT 被替换成了 Short 000E，变量 SUM 被替换为[0000]，并且不知道如何查看最后 SUM 变量的值。

### 1.3 实验总结

本次实验收获颇多，熟悉了汇编语言的调试环境，学会使用了 DOSBox 及其分辨率的修改配置。熟悉了 debug 查看寄存器和内存的内容，学会使用-r 查看修改寄存器内容，-d 查看内存内容，-e 修改内存内容，-a 以汇编指令格式写入内存，-t 单步执行查看命令

执行结果，等等。

同时，在本次实验中，通过打汇编程序代码，并运行 `masm`、`link` 调试源代码错误，再用 `debug` 运行可执行文件，基本熟悉了编写汇编程序再到编译、执行的完整过程。

本次实验过程中遇到了不少问题，但都在最后一一解决，也深刻感受到了作为汇编语言，其操作十分接近计算机底层，但同时操作会比较繁琐复杂，容易出错。

## 1.4 尚未解决问题

- (1) 任务四中，删除 `USE16` 在 `masm` 过程中并未报错
- (2) 任务四中，源程序中经 `debug32` 反汇编后，标识符 `NEXT` 被替换成了 `Short 000E`，变量 `SUM` 被替换为 `[0000]`，并且不知道如何查看最后 `SUM` 变量的值。
- (3) 任务四中，可执行文件 `ye28.exe` 不能直接打开。

## 1.5 附件

任务四：ye28.asm 文件源代码：

```
.386
DATA    SEGMENT USE16
SUM      DW 0
DATA    ENDS
STACK   SEGMENT USE16 STACK
        DB 200 DUP(0)
STACK   ENDS
CODE    SEGMENT USE16
        ASSUME DS:DATA,SS:STACK,CS:CODE
START:  MOV AX,DATA
        MOV DS,AX
        MOV CX,50
        MOV AX,0
        MOV BX,1
NEXT:   ADD AX,BX
        INC BX
        INC BX
        DEC CX
        JNE NEXT
        MOV SUM,AX
        MOV AH,4CH
        INT 21H
CODE    ENDS
        END START
```