《汇编语言》实验

实验一、实验二：查看 cpu 和内存，用机器指令和汇编指令编程；用机器指令和汇编指令编程

专业：　　计算机科学与技术

姓名：　　高星杰

学号：　　2021307220712

班级：　　计科 2102

报告上交时间：2023 年　4 月 9 日

一、实验目的

1、熟悉实验环境

2、掌握 Debug 的用法

3、比较 debug 和汇编源程序两种执行指令的方式

二、实验任务及要求

实验一任务：

（1）使用 debug 将下面程序写入内存逐条执行，观察每条指令执行后 cpu 中相关寄存器中内容的变化。

（2）将下面 3 条指令写入从 2000：0 开始的内存单元中，利用这三条指令计算 2 的 8 次方

（3）查看内存中的内容

Pc 机主板上的 rom 中写有一个生产日期，在内存 FFF00H~FFFFFFH 的某几个单元中，请找到这个生产日期并试图改变它。

（4）向内存从 B8100H 开始的单元中填写数据，如：

-e B810:0000 01 01 02 02 03 04 04、

实验二任务：

（1）使用 debug 将下面的程序写入内存中，逐条执行，根据指令执行后的实际运行情况填空。

（2）仔细观察图 3.19 中实验过程，然后分析：为什么 2000：0~2000:f 中的内会发生改变？

三、实验步骤

实验一任务：

（1）使用 debug 将下面程序写入内存逐条执行，观察每条指令执行后 cpu 中相关寄存器中内容的变化。

使用 a 命令将指令写入内存后，然后使用-u 查看内存中的指令：

```
-u
0770:0000 B8204E        MOV     AX,4E20
0770:0003 051614        ADD     AX,1416
0770:0006 BB0020        MOV     BX,2000
0770:0009 01D8          ADD     AX,BX
0770:000B 89C3          MOV     BX,AX
0770:000D 01D8          ADD     AX,BX
0770:000F B81A00        MOV     AX,001A
0770:0012 BB2600        MOV     BX,0026
0770:0015 00D8          ADD     AL,BL
0770:0017 00DC          ADD     AH,BL
0770:0019 B400          MOV     AH,00
0770:001B 00D8          ADD     AL,BL
0770:001D 049C          ADD     AL,9C
0770:001F 0000          ADD     [BX+SI],AL
```

可以看到每条指令已经写入的以 0770：0000 地址开头的内存中。（其次要修改 cs：ip 的值）

然后再使用-t 命令逐条执行并观察寄存器中的值（一共十四条指令）：

```
-t

AX=4E20  BX=0000  CX=0009  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0760  ES=0760  SS=076F  CS=0770  IP=0003   NV UP EI PL NZ NA PO NC
0770:0003 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=0770  IP=0006   NV UP EI PL NZ NA PE NC
0770:0006 BB0020        MOV     BX,2000
-
```

```
-t

AX=6236  BX=0000  CX=0009  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0760  ES=0760  SS=076F  CS=0770  IP=0006   NV UP EI PL NZ NA PE NC
0770:0006 BB0020        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=0770  IP=0009   NV UP EI PL NZ NA PE NC
0770:0009 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=0770  IP=000B   OV UP EI NG NZ NA PE NC
0770:000B 89C3          MOV     BX,AX
-
```

```
-t

AX=8236  BX=8236  CX=0009  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0760  ES=0760  SS=076F  CS=0770  IP=000D   OV UP EI NG NZ NA PE NC
0770:000D 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=0770  IP=000F   OV UP EI PL NZ NA PE CY
0770:000F 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=0770  IP=0012   OV UP EI PL NZ NA PE CY
0770:0012 BB2600        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=0770  IP=0015   OV UP EI PL NZ NA PE CY
0770:0015 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=0770  IP=0017   NV UP EI PL NZ AC PO NC
0770:0017 00DC          ADD     AH,BL
```

```
0770:0017 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=0770  IP=0019   NV UP EI PL NZ NA PO NC
0770:0019 B400          MOV     AH,00
- ▲
```

```
-t

AX=0040  BX=0026  CX=0009  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0760  ES=0760  SS=076F  CS=0770  IP=001B   NV UP EI PL NZ NA PO NC
0770:001B 00D8          ADD     AL,BL
```

```
-t

AX=0066  BX=0026  CX=0009  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0760  ES=0760  SS=076F  CS=0770  IP=001D   NV UP EI PL NZ NA PE NC
0770:001D 049C          ADD     AL,9C
```

```
-t

AX=0002  BX=0026  CX=0009  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0760  ES=0760  SS=076F  CS=0770  IP=001F   NV UP EI PL NZ AC PO CY
0770:001F 0000          ADD     [BX+SI],AL                      DS:0026=FF
```

（2）将下面 3 条指令写入从 2000：0 开始的内存单元中，利用这三条指令计算 2 的 8 次方

使用 a 指令将程序写入到内存中并使用-u 来查看

```
-a 2000:0000
2000:0000 mov ax,1
2000:0003 add ax,ax
2000:0005 jmp 2000:0003
2000:0007
```

写在了以 2000：0000 为首地址的内存中。

然后修改 cs：ip 的值并用 t 指令执行程序：

```
-t

AX=0001  BX=0026  CX=0009  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0760  ES=0760  SS=076F  CS=2000  IP=0003   NV UP EI PL ZR NA PE NC
2000:0003 01C0          ADD     AX,AX
```

```
-t

AX=0002  BX=0026  CX=0009  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0760  ES=0760  SS=076F  CS=2000  IP=0005   NV UP EI PL NZ NA PO NC
2000:0005 EBFC          JMP     0003
```

```
-t

AX=0002  BX=0026  CX=0009  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0760  ES=0760  SS=076F  CS=2000  IP=0003   NV UP EI PL NZ NA PO NC
2000:0003 01C0          ADD     AX,AX
```

```
-t

AX=0002  BX=0026  CX=0009  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0760  ES=0760  SS=076F  CS=2000  IP=0003   NV UP EI PL NZ NA PO NC
2000:0003 01C0          ADD     AX,AX
-t

AX=0004  BX=0026  CX=0009  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0760  ES=0760  SS=076F  CS=2000  IP=0005   NV UP EI PL NZ NA PO NC
2000:0005 EBFC          JMP     0003
-t

AX=0004  BX=0026  CX=0009  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0760  ES=0760  SS=076F  CS=2000  IP=0003   NV UP EI PL NZ NA PO NC
2000:0003 01C0          ADD     AX,AX
-t

AX=0008  BX=0026  CX=0009  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0760  ES=0760  SS=076F  CS=2000  IP=0005   NV UP EI PL NZ NA PO NC
2000:0005 EBFC          JMP     0003
```

```
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=0005   NV UP EI PL NZ NA PO NC
2000:0005 EBFC          JMP     0003
-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=0003   NV UP EI PL NZ NA PO NC
2000:0003 01C0          ADD     AX,AX
-t

AX=0080  BX=0026  CX=0009  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0760  ES=0760  SS=076F  CS=2000  IP=0005   NV UP EI PL NZ NA PO NC
2000:0005 EBFC          JMP     0003
-t

AX=0080  BX=0026  CX=0009  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0760  ES=0760  SS=076F  CS=2000  IP=0003   NV UP EI PL NZ NA PO NC
2000:0003 01C0          ADD     AX,AX
-t

AX=0100  BX=0026  CX=0009  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0760  ES=0760  SS=076F  CS=2000  IP=0005   NV UP EI PL NZ NA PE NC
2000:0005 EBFC          JMP     0003
-
```

（3）查看内存中的内容

Pc 机主板上的 rom 中写有一个生产日期，在内存 FFF00H~FFFFFFH 的某几个单元中，请找到这个生产日期并试图改变它。

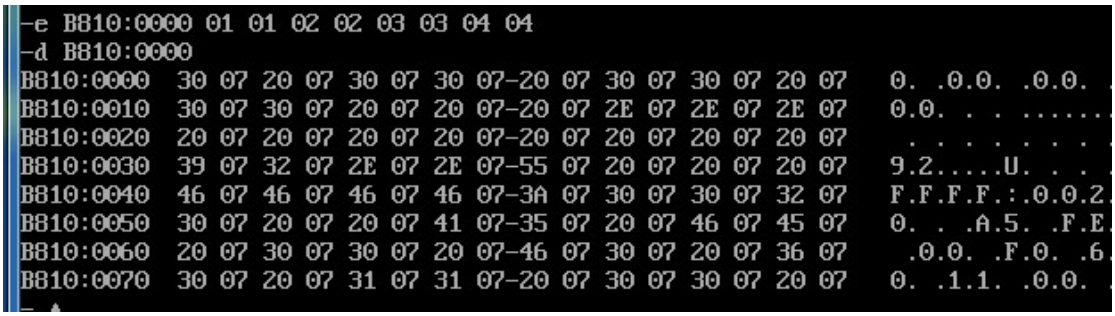
```
-d FFFF:0000
FFFF:0000  EA C0 12 00 F0 30 31 2F-30 31 2F 39 32 00 FC 55   .....01/01/92..U
FFFF:0010  60 10 00 F0 BB 13 A6 01-08 00 70 00 B1 13 A6 01   `.........p.....
FFFF:0020  08 00 70 00 60 10 00 F0-60 10 00 F0 60 10 00 F0   ..p.`...`...`...
FFFF:0030  A5 FE 00 F0 87 E9 00 F0-55 FF 00 F0 60 10 00 F0   ........U...`...
FFFF:0040  60 10 00 F0 60 10 00 F0-80 10 00 F0 60 10 00 F0   `...`.......`...
FFFF:0050  00 13 00 F0 00 11 00 F0-20 11 00 F0 40 11 00 F0   ........ ...@...
FFFF:0060  A0 11 00 F0 C0 11 00 F0-E0 11 00 F0 20 12 00 F0   ............ ...
FFFF:0070  C0 12 00 F0 C0 12 00 F0-40 12 00 F0 60 10 00 F0   ........@...`...
-
```

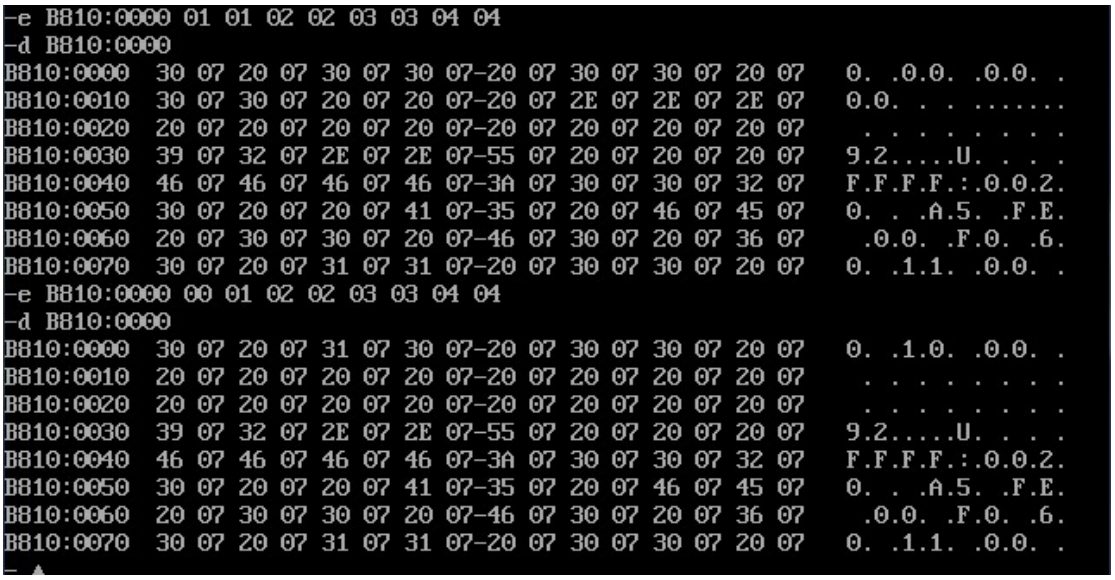
```
-d FFFF:0000
FFFF:0000  EA C0 12 00 F0 30 31 2F-30 31 2F 39 32 00 FC 55   .....01/01/92..U
FFFF:0010  60 10 00 F0 BB 13 A6 01-08 00 70 00 B1 13 A6 01   `.........p.....
FFFF:0020  08 00 70 00 60 10 00 F0-60 10 00 F0 60 10 00 F0   ..p.`...`...`...
FFFF:0030  A5 FE 00 F0 87 E9 00 F0-55 FF 00 F0 60 10 00 F0   ........U...`...
FFFF:0040  60 10 00 F0 60 10 00 F0-80 10 00 F0 60 10 00 F0   `...`.......`...
FFFF:0050  00 13 00 F0 00 11 00 F0-20 11 00 F0 40 11 00 F0   ........ ...@...
FFFF:0060  A0 11 00 F0 C0 11 00 F0-E0 11 00 F0 20 12 00 F0   ............ ...
FFFF:0070  C0 12 00 F0 C0 12 00 F0-40 12 00 F0 60 10 00 F0   ........@...`...
-e FFFF:0000 00 00 00
-d FFFF:0000
FFFF:0000  EA C0 12 00 F0 30 31 2F-30 31 2F 39 32 00 FC 55   .....01/01/92..U
FFFF:0010  60 10 00 F0 BB 13 A6 01-08 00 70 00 B1 13 A6 01   `.........p.....
FFFF:0020  08 00 70 00 60 10 00 F0-60 10 00 F0 60 10 00 F0   ..p.`...`...`...
FFFF:0030  A5 FE 00 F0 87 E9 00 F0-55 FF 00 F0 60 10 00 F0   ........U...`...
FFFF:0040  60 10 00 F0 60 10 00 F0-80 10 00 F0 60 10 00 F0   `...`.......`...
FFFF:0050  00 13 00 F0 00 11 00 F0-20 11 00 F0 40 11 00 F0   ........ ...@...
FFFF:0060  A0 11 00 F0 C0 11 00 F0-E0 11 00 F0 20 12 00 F0   ............ ...
FFFF:0070  C0 12 00 F0 C0 12 00 F0-40 12 00 F0 60 10 00 F0   ........@...`...
-
```

改动无效，因为没有这个权限无法修改只读信息。

（5）向内存从 B8100H 开始的单元中填写数据，如：

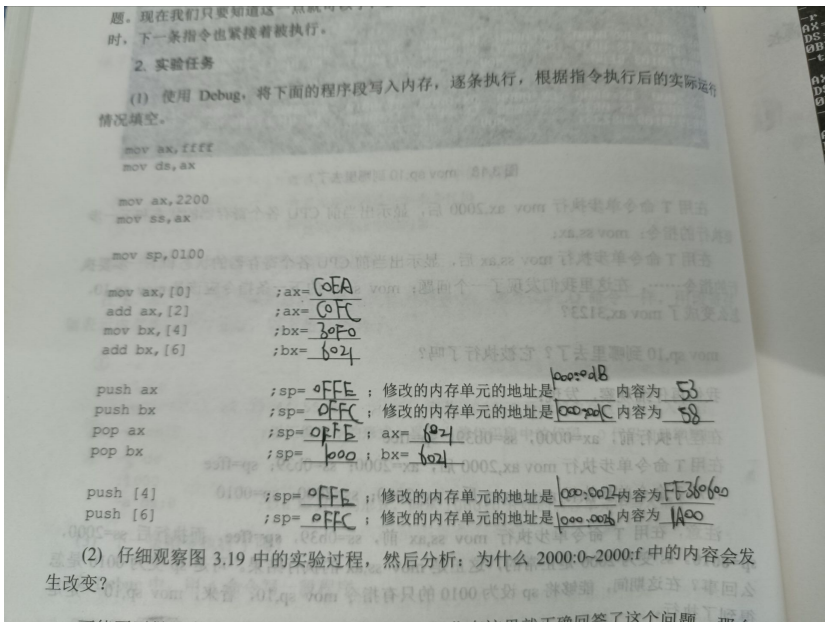
-e B810:0000 01 01 02 02 03 04 04





实验二任务：

（1）使用 debug 将下面的程序写入内存中，逐条执行，根据指令执行后的实际运行情况填空。



题。现在我们只要知道这一点就可以了。

时，下一条指令也紧接着被执行。

2. 实验任务

(1) 使用 Debug，将下面的程序段写入内存，逐条执行，根据指令执行后的实际运行情况填空。

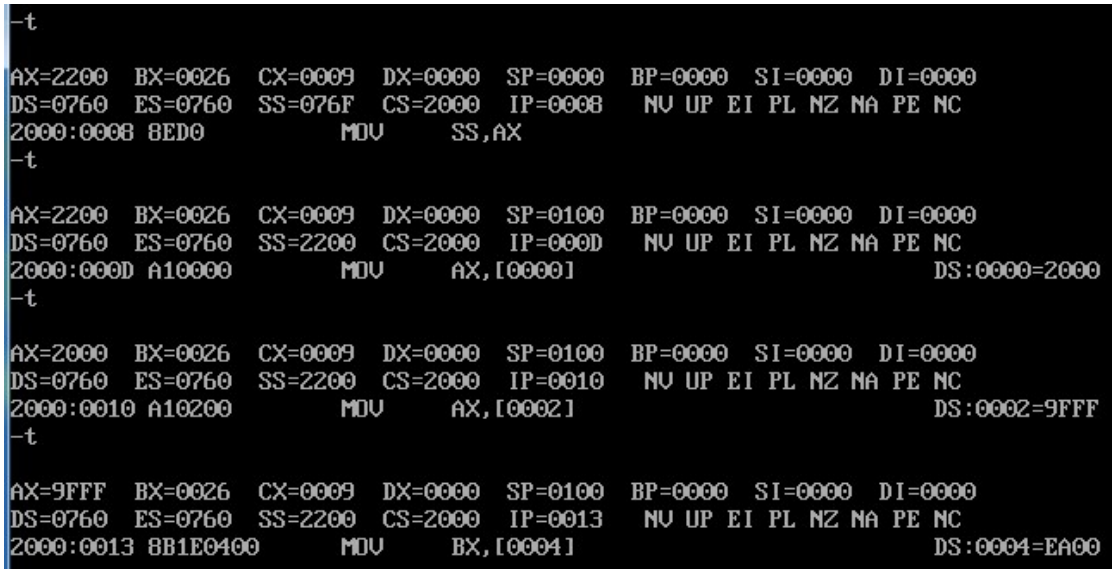
```
mov ax,ffff
mov ds,ax

mov ax,2200
mov ss,ax

mov sp,0100

mov ax,[0]      ;ax=00EA
add ax,[2]      ;ax=00FC
mov bx,[4]      ;bx=30F0
add bx,[6]      ;bx=1024

push ax         ;sp=00FE ; 修改的内存单元的地址是0000:00FB 内容为 53
push bx         ;sp=00FC ; 修改的内存单元的地址是0000:00FC 内容为 58
pop ax          ;sp=00FE ; ax=1024
pop bx          ;sp=0100 ; bx=1024

push [4]        ;sp=00FE ; 修改的内存单元的地址是0000:0027 内容为 FF30F0
push [6]        ;sp=00FC ; 修改的内存单元的地址是0000:0026 内容为 1A00
```

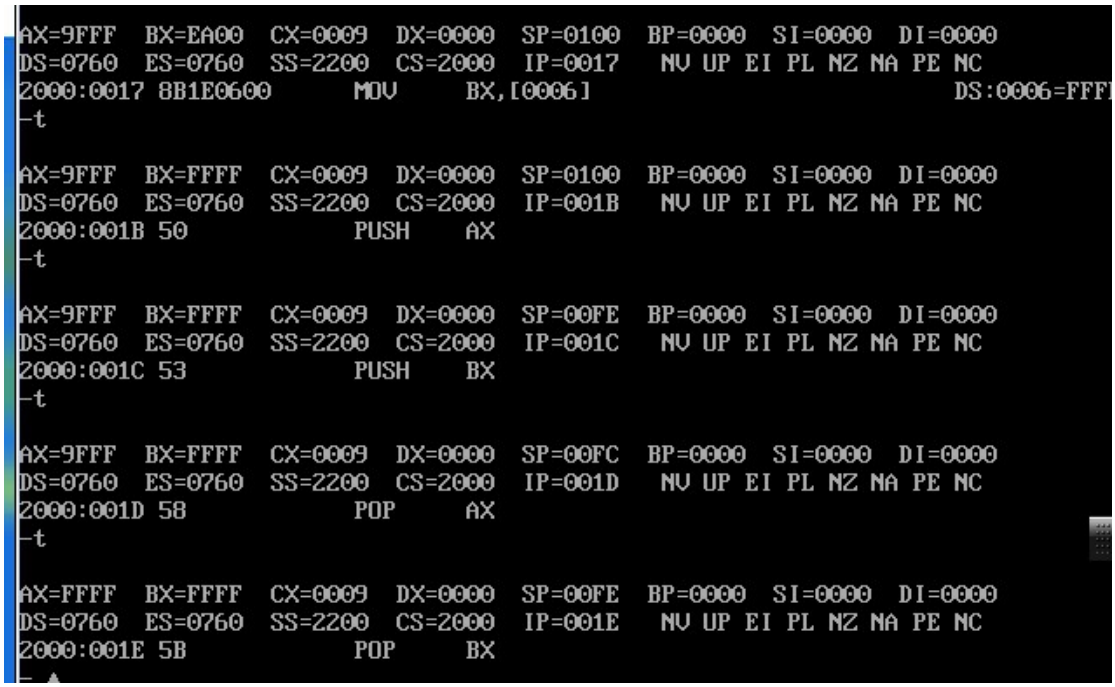
(2) 仔细观察图 3.19 中的实验过程，然后分析：为什么 2000:0~2000:f 中的内容会发生改变？

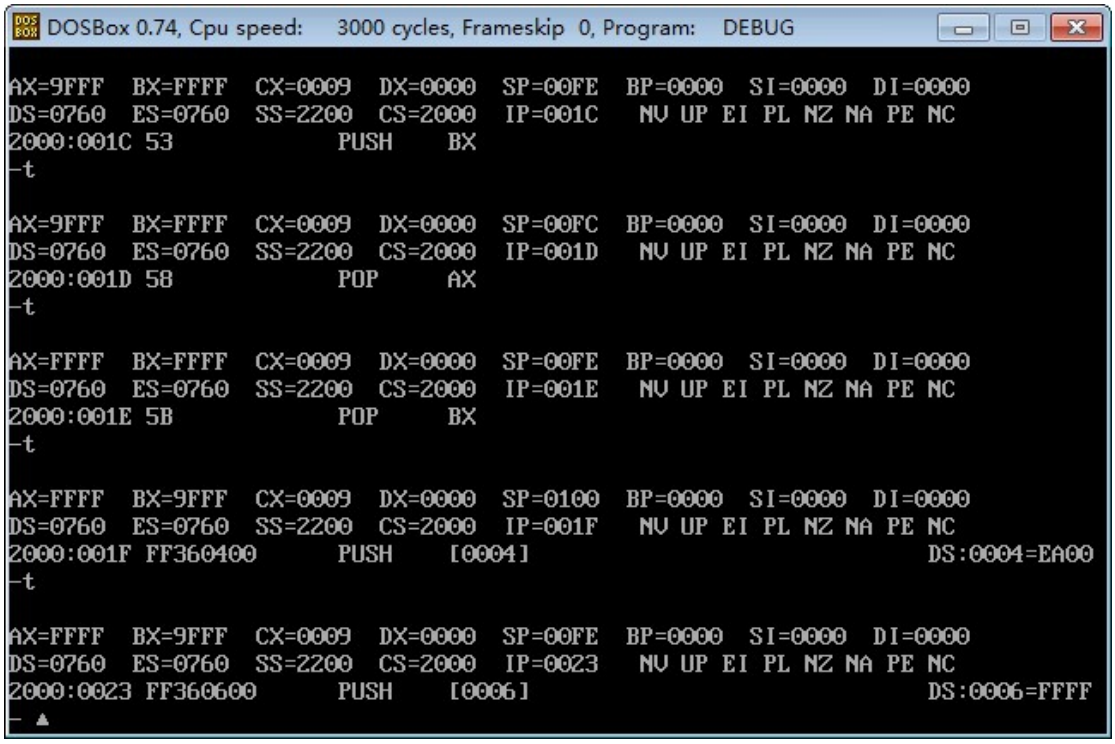
使用 a 命令写入内存

```
2000:0000
-a 2000:0000
2000:0000 mov ax,ffff
2000:0003 ds,ax
          ^ Error
2000:0003 mov ds,ax
2000:0005 mov ax,2200
2000:0008 mov ss,ax
2000:000A mov sp,0100
2000:000D mov ax,[0]
2000:0010 mov ax,[2]
2000:0013 mov bx,[4]
2000:0017 mov bx,[6]
2000:001B push as
               ^ Error
2000:001B push ax
2000:001C push bx
2000:001D pop ax
2000:001E pop bx
2000:001F push [4]
2000:0023 push [6]
2000:0027
```

然后逐条执行并返回寄存器结果：


-t
AX=2200 BX=0026 CX=0009 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=2000 IP=0008 NV UP EI PL NZ NA PE NC
2000:0008 8ED0 MOV SS,AX
-t
AX=2200 BX=0026 CX=0009 DX=0000 SP=0100 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=2200 CS=2000 IP=000D NV UP EI PL NZ NA PE NC
2000:000D A10000 MOV AX,[0000] DS:0000=2000
-t
AX=2000 BX=0026 CX=0009 DX=0000 SP=0100 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=2200 CS=2000 IP=0010 NV UP EI PL NZ NA PE NC
2000:0010 A10200 MOV AX,[0002] DS:0002=9FFF
-t
AX=9FFF BX=0026 CX=0009 DX=0000 SP=0100 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=2200 CS=2000 IP=0013 NV UP EI PL NZ NA PE NC
2000:0013 8B1E0400 MOV BX,[0004] DS:0004=EA00


AX=9FFF BX=EA00 CX=0009 DX=0000 SP=0100 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=2200 CS=2000 IP=0017 NV UP EI PL NZ NA PE NC
2000:0017 8B1E0600 MOV BX,[0006] DS:0006=FFF
-t
AX=9FFF BX=FFFF CX=0009 DX=0000 SP=0100 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=2200 CS=2000 IP=001B NV UP EI PL NZ NA PE NC
2000:001B 50 PUSH AX
-t
AX=9FFF BX=FFFF CX=0009 DX=0000 SP=00FE BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=2200 CS=2000 IP=001C NV UP EI PL NZ NA PE NC
2000:001C 53 PUSH BX
-t
AX=9FFF BX=FFFF CX=0009 DX=0000 SP=00FC BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=2200 CS=2000 IP=001D NV UP EI PL NZ NA PE NC
2000:001D 58 POP AX
-t
AX=FFFF BX=FFFF CX=0009 DX=0000 SP=00FE BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=2200 CS=2000 IP=001E NV UP EI PL NZ NA PE NC
2000:001E 5B POP BX



（2）仔细观察图 3.19 中实验过程，然后分析：为什么 2000：0~2000:f 中的内会发生改变？

答：因为用 t 指令进行调试的时候，会产生中断。而为了保存结果，cpu 则先将标志寄存器进栈、再吧当前 cs 的值进栈，最后将 ip 的值进栈。

四、实验体会与讨论

debug 和汇编源程序执行的区别：

1. Debug 是一种程序调试工具，用于在程序运行过程中进行调试和排错。它可以让程序员在程序运行时暂停程序的执行，查看程序状态、变量值、堆栈信息等，并单步执行程序，以便找出程序中的错误和问题。

2. 汇编源程序是一种低级别的程序代码，它直接使用 CPU 指令来操作计算机硬件。汇编源程序需要经过汇编器的处理，将其转换为机器码，然后才能在计算机上运行。汇编源程序通常用于开发系统级别的软件，如操作系统、驱动程序等。

因此，Debug 和汇编源程序执行的区别在于，Debug 是一种调试工具，用于调试程序，而汇编源程序是一种程序代码，需要经过汇编器处理才能在计算机上运行。

实验总结：

本次实验第一次使用汇编语言，熟悉了基本的使用汇编语言编写程序，输入程序的方式，了解到了 cs:ip 的修改，怎么执行一条指令，具体有对 mov、a、t、e 命令的熟悉和对寄存器的熟悉，对以后再深入了解计算机的内部有很大帮助，有助于培养计算思维。