调试工具 DEBUG 的使用

一、实验目的：

1.熟悉 DEBUG 的功能，运行并掌握其常用命令。

2.使用 DEBUG，通过数据传送指令观察 IBM-PC 机各种寻址方式的区别。

二、实验环境(硬件或软件)：

在 DOS 或 Windows 的命令行窗口执行命令 Debug.exe，进入 Debug 程序环境，熟悉 Debug 各个命令的用法,Windows 8 版本没有 Debug 环境，需要配置虚拟机环境。

三、实验原理：

1. DEBUG 的加载及其常用命令（A、U、R、D、E、T、P、G、Q）的使用情况。（注意：微机进入 DEBUG 状态下之后,一切立即数和地址数据均被默认为十六进制数,在输入时数的后面不加后缀“H”。）

2.按照程序要求编写简单程序段.

3.在 Debug 中输入简单的汇编程序片断，并调试运行，得出结果.

四、实验内容(实验步骤或者程序编写)：

（一）DEBUG 的加载及其常用命令

1. R：观看和修改寄存器的值

```
-R
AX=0000  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0ADF  ES=0ADF  SS=0ADF  CS=0ADF  IP=0100   NV UP EI PL NZ NA PO NC
0ADF:0100 F606159920    TEST    BYTE PTR [9915],20              DS:9915=00
-R CX
CX 0000
:100
-R
AX=0000  BX=0000  CX=0100  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0ADF  ES=0ADF  SS=0ADF  CS=0ADF  IP=0100   NV UP EI PL NZ NA PO NC
0ADF:0100 F606159920    TEST    BYTE PTR [9915],20              DS:9915=00
-
```

（修改 CX 的值）

2.E：改变内存单位的内容

```
-E 1AEF:100
1AEF:0100  00.12   00.34   00.     00.FF
-D 1AEF:100 110
1AEF:0100  12 34 00 FF 00 00 00 00-00 00 00 00 00 00 00 00   .4..............
1AEF:0110  00                                                .
-
```

3.D：显示内存区域的内容

```
-D 1AEF:100
1AEF:0100  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
1AEF:0110  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
1AEF:0120  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
1AEF:0130  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
1AEF:0140  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
1AEF:0150  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
1AEF:0160  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
1AEF:0170  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-D DS:100 15A
0ADF:0100  F6 06 15 99 20 75 0A 80-3E D2 99 00 75 BB 49 74   .... u..>...u.It
0ADF:0110  B8 BA 0D 8C EB 23 33 D2-87 D1 B8 01 34 00 CE 0A   .....#3.....4...
0ADF:0120  DF 99 89 16 E1 99 80 3E-C5 96 00 74 9C B4 40 CD   .......>...t..@.
0ADF:0130  21 72 5F C6 06 E3 99 1A-C3 E8 FF 0F FE 06 D2 96   !r_.............
0ADF:0140  80 3E D1 96 00 74 48 8B-1E 13 99 83 FB 00 7E 33   .>...tH.......~3
0ADF:0150  8B 0E E1 99 8B 16 DF 99-8B C1 0B                  ...........
-D DS:100 L1F
0ADF:0100  F6 06 15 99 20 75 0A 80-3E D2 99 00 75 BB 49 74   .... u..>...u.It
0ADF:0110  B8 BA 0D 8C EB 23 33 D2-87 D1 B8 01 34 00 CE      .....#3.....4..
-
```

（默认显示 128 个内存单元；可以限定始末单元；可以限定开始单元及显示的长度）

4. A：输入汇编指令

G：执行汇编指令

```
-A100
0ADF:0100 MOV AX,CS
0ADF:0102 MOV DS,AX
0ADF:0104 MOV DX,200
0ADF:0107 MOV AH,9
0ADF:0109 INT 21
0ADF:010B INT 20
0ADF:010D
-G=100
ABCD
Program terminated normally
-G=100 10B
ABCD
AX=0924  BX=0000  CX=0100  DX=0200  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0ADF  ES=0ADF  SS=0ADF  CS=0ADF  IP=010B   NV UP EI PL NZ NA PO NC
0ADF:010B CD20          INT     20
-
```

（A：以上的程序要在屏幕上显示“ABCD”四个字符。首先用 E 命令将“ABCD$”四个字符预先放在内存 CS:200 处，然后执行 A100 命令输入汇编程序代码。前两行汇编指令用于将段寄存器 CS 的值赋给段寄存器 DS。第三到第五行汇编代码的作用是显示以“$”为结尾的字符串。最后一行用于结束程序）

（G：汇编程序运行后在屏幕上显示出“ABCD”四个字符。接下来在 DEBUG 中执行 G=100 10B，意思是从地址 CS：100 开始，一直运行到 CS：10B 停止。观看运行结果。命令执行后，不但显示出字符串“ABCD”，而且列出当前寄存器和标志位的值。）

5.U：对机器代码反汇编显示

```
-G=100
ABCD
Program terminated normally
-G=100 10B
ABCD
AX=0924  BX=0000  CX=0100  DX=0200  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0ADF  ES=0ADF  SS=0ADF  CS=0ADF  IP=010B   NV UP EI PL NZ NA PO NC
0ADF:010B CD20          INT     20
-U100
0ADF:0100 8CC8          MOV     AX,CS
0ADF:0102 8ED8          MOV     DS,AX
0ADF:0104 BA0002        MOV     DX,0200
0ADF:0107 B409          MOV     AH,09
0ADF:0109 CD21          INT     21
0ADF:010B CD20          INT     20
0ADF:010D BB4974        MOV     BX,7449
0ADF:0110 B8BA0D        MOV     AX,0DBA
0ADF:0113 8CEB          MOV     BX,CS
0ADF:0115 2333          AND     SI,[BP+DI]
0ADF:0117 D287D1B8      ROL     BYTE PTR [BX+B8D1],CL
0ADF:011B 0134          ADD     [SI],SI
0ADF:011D 00CE          ADD     DH,CL
0ADF:011F 0ADF          OR      BL,BH
-
```

6.T：执行汇编程序，单步跟踪

```
-R
AX=0000  BX=0000  CX=0110  DX=0000  SP=FFFE  BP=0000  SI=0000  DI=0000
DS=0B25  ES=0B25  SS=0B25  CS=0B25  IP=0100   NV UP EI PL NZ NA PO NC
0B25:0100 8CC8          MOV     AX,CS
-T=100

AX=0B25  BX=0000  CX=0110  DX=0000  SP=FFFE  BP=0000  SI=0000  DI=0000
DS=0B25  ES=0B25  SS=0B25  CS=0B25  IP=0102   NV UP EI PL NZ NA PO NC
0B25:0102 8ED8          MOV     DS,AX
-T

AX=0B25  BX=0000  CX=0110  DX=0000  SP=FFFE  BP=0000  SI=0000  DI=0000
DS=0B25  ES=0B25  SS=0B25  CS=0B25  IP=0104   NV UP EI PL NZ NA PO NC
0B25:0104 BA0002        MOV     DX,0200
-T

AX=0B25  BX=0000  CX=0110  DX=0200  SP=FFFE  BP=0000  SI=0000  DI=0000
DS=0B25  ES=0B25  SS=0B25  CS=0B25  IP=0107   NV UP EI PL NZ NA PO NC
0B25:0107 B409          MOV     AH,09
-T

AX=0925  BX=0000  CX=0110  DX=0200  SP=FFFE  BP=0000  SI=0000  DI=0000
DS=0B25  ES=0B25  SS=0B25  CS=0B25  IP=0109   NV UP EI PL NZ NA PO NC
0B25:0109 CD21          INT     21
-
```

7.Q：退出 DEBUG，回到 DOS 状态

```
-Q

C:\DOCUME~1\Owner>_
```

（二）将数据段中的两个数求和，结果保存在数据段中

（1）用 E 命令键入机器语言程序和数据段的初值。

—E DS：000 23 01 25 00

—E DS：006 2A 2A 2A （2A 为*，是为便于查看数据段内容而设置）

—E CS：100 A1 00 00 03 06 02 00

—E CS：107 A3 04 00 CB

（2）用 D 命令检查数据段、代码段内容。

—D DS：0

—D CS：100

```
-E DS:000 23 01 25 00
-E DS:006 2A 2A 2A
-E CS:100 A1 00 00 03 06 02 00
-E CS:107 A3 04 00 CB
-D DS:0
0ADF:0000  23 01 25 00 00 9A 2A 2A-2A F0 4F 03 43 05 8A 03   #.%...***.O.C...
0ADF:0010  43 05 17 03 43 05 0F 04-01 01 01 00 02 FF FF FF   C...C...........
0ADF:0020  FF FF FF FF FF FF FF FF-FF FF FF FF FC 04 4E 01   ..............N.
0ADF:0030  03 0A 14 00 18 00 DF 0A-FF FF FF FF 00 00 00 00   ................
0ADF:0040  05 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0ADF:0050  CD 21 CB 00 00 00 00 00-00 00 00 00 00 20 20 20   .!...........
0ADF:0060  20 20 20 20 20 20 20 20-00 00 00 00 00 20 20 20            .....
0ADF:0070  20 20 20 20 20 20 20 20-00 00 00 00 00 00 00 00            ........
-D CS:100
0ADF:0100  A1 00 00 03 06 02 00 A3-04 00 CB 00 75 BB 49 74   ............u.It
0ADF:0110  B8 BA 0D 8C EB 23 33 D2-87 D1 B8 01 34 00 CE 0A   .....#3.....4...
0ADF:0120  DF 99 89 16 E1 99 80 3E-C5 96 00 74 9C B4 40 CD   .......>...t..@.
0ADF:0130  21 72 5F C6 06 E3 99 1A-C3 E8 FF 0F FE 06 D2 96   !r_.............
0ADF:0140  80 3E D1 96 00 74 48 8B-1E 13 99 83 FB 00 7E 33   .>...tH.......~3
0ADF:0150  8B 0E E1 99 8B 16 DF 99-8B C1 0B C2 74 21 B8 00   ............t!..
0ADF:0160  42 CD 21 33 C9 B4 40 CD-21 80 3E E3 99 00 74 08   B.!3..@.!.>...t.
0ADF:0170  41 BA E3 99 B4 40 CD 21-B4 3E CD 21 E9 6A FA B4   A....@.!.>.!.j..
-_
```

（3）用 T 命令逐条执行上述程序，注意观察每条指令执行后相应寄存器的变化。

```
-R IP
IP 00FF
:00FE
-T

AX=0123  BX=0000  CX=FFFF  DX=8C0D  SP=FFEA  BP=0000  SI=0000  DI=0000
DS=0ADF  ES=0ADF  SS=0ADF  CS=0ADF  IP=0100   NV UP EI PL NZ NA PE NC
0ADF:0100 A10000        MOV     AX,[0000]                       DS:0000=0169
-T

AX=0169  BX=0000  CX=FFFF  DX=8C0D  SP=FFEA  BP=0000  SI=0000  DI=0000
DS=0ADF  ES=0ADF  SS=0ADF  CS=0ADF  IP=0103   NV UP EI PL NZ NA PE NC
0ADF:0103 03060200      ADD     AX,[0002]                       DS:0002=0025
-T

AX=018E  BX=0000  CX=FFFF  DX=8C0D  SP=FFEA  BP=0000  SI=0000  DI=0000
DS=0ADF  ES=0ADF  SS=0ADF  CS=0ADF  IP=0107   NV UP EI PL NZ NA PE NC
0ADF:0107 A30400        MOV     [0004],AX                       DS:0004=9A00
-T

AX=018E  BX=0000  CX=FFFF  DX=8C0D  SP=FFEA  BP=0000  SI=0000  DI=0000
DS=0ADF  ES=0ADF  SS=0ADF  CS=0ADF  IP=010A   NV UP EI PL NZ NA PE NC
0ADF:010A CB            RETF
-
```

（4）用 U 命令反汇编本程序。

—U 100 10A

```
-U 100 10A
0ADF:0100 A10000        MOV     AX,[0000]
0ADF:0103 03060200      ADD     AX,[0002]
0ADF:0107 A30400        MOV     [0004],AX
0ADF:010A CB            RETF
-
```

（5）用 R 命令修改 IP 的内容为 0100 重新执行上述程序。

```
-R IP
IP 010A
:0100
-T

AX=0123  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0ADF  ES=0ADF  SS=0ADF  CS=0ADF  IP=0103   NV UP EI PL NZ NA PE NC
0ADF:0103 03060200      ADD     AX,[0002]                       DS:0002=0025
-T

AX=0148  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0ADF  ES=0ADF  SS=0ADF  CS=0ADF  IP=0107   NV UP EI PL NZ NA PE NC
0ADF:0107 A30400        MOV     [0004],AX                       DS:0004=0148
-T

AX=0148  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0ADF  ES=0ADF  SS=0ADF  CS=0ADF  IP=010A   NV UP EI PL NZ NA PE NC
0ADF:010A CB            RETF
-
```

（6）用 G 命令运行程序。

—G =CS：0100 107

观察：IP＝0107 AX＝0148

—G =CS：0100 10A

观察：IP＝ 010A AX＝ 0148

```
-G =CS:0100 107

AX=0148  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0ADF  ES=0ADF  SS=0ADF  CS=0ADF  IP=0107   NV UP EI PL NZ NA PE NC
0ADF:0107 A30400        MOV     [0004],AX                       DS:0004=0148
-G =CS:0100 10A

AX=0148  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0ADF  ES=0ADF  SS=0ADF  CS=0ADF  IP=010A   NV UP EI PL NZ NA PE NC
0ADF:010A CB            RETF
-
```

（三）自己试着写段程序：

```
-A 100
0ADF:0100 DB '1234567890'
0ADF:010A CLD
0ADF:010B MOV SI,100
0ADF:010E MOV DI,200
0ADF:0111 MOV CX,A
0ADF:0114 REP MOVSB
0ADF:0116
-G=10A 116

AX=0148  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=010A  DI=020A
DS=0ADF  ES=0ADF  SS=0ADF  CS=0ADF  IP=0116   NV UP EI PL NZ NA PE NC
0ADF:0116 33D2          XOR     DX,DX
-D 100 L A
0ADF:0100  31 32 33 34 35 36 37 38-39 30                      1234567890
-D ES:200 L A
0ADF:0200  31 32 33 34 35 36 37 38-39 30                      1234567890
-
```

五、实验结果及分析：

在命令窗口中启动 DEBUG，随着启动成功后，将显示连接符“-”，这时，可输入各种 DEBUG 命令。另外在提示符“-”下才能输入命令，在按“回车”键后，该命令才开始执行命令是单个字母，命令和参数的大小写可混合输入可用 F1、F2、F3、Ins、Del、左移键、右移键等编辑键来编辑本行命令当命令出现语法错误时，将在出错位置显示“^ Error”可用 Ctrl+C 或 Ctrl+Break 来终止当前命令的执行，还可用 Ctrl+S 或 Ctrl+Num Lock 来暂停屏幕显示(当连续不断地显示信息时)，用 Alt+Enter 可以全屏。

使用各种 DEBUG 命令都有其各自的格式。有些语法错误系统会在出错位置显示 Error 可是有些却不是，像在使用 G 命令时，不加等号直接输入一个 G 会直接退出 DEBUG 模式。退出后，之前修改的内存单元内容有可能会改变，导致前功竟弃。又例如使用 G 执行程序前要先用 A 写入汇编指令而不是直接写机器指令。

同时在写机器指令时也深深感悟到了为什么汇编语言会出现，由于一个数字的录入错误，结果将会和自己想要的不一样。

实验后不足的地方就是还未能记住 DEBUG 常用命令，包括他们的格式和用途，由于不熟练在实验过程中经常翻阅，降低了效率。这需要日常多去使用甚至是下功夫进行记忆。

另外，通过实验我们应该要加深 2 进制和 16 进制的意识。计算机只能识别 0 和 1，为了方便，我们用 16 进制去表示。DEBUG 工具默认的数字都是 16 进制，尤其是在查看内存（D 命令）时，可以限定始末位置，也可以限定开始位置和显示长度，这些用的都是 16 进制。其实 DEBUG 也很好地让我们去类比 10 进制。例如 D 命令默认显示的内存单元一行刚好是 16 个，这也就方便我们的查找。

通过 DEBUG 工具的使用，不仅能加深理解和巩固课堂上老师所学的知识，而且还能了解计算机的底层调试软件的工作过程，为今后学习后续课程打好良好的基础。