

# 《汇编语言》

## 实验报告

实验名称：实验二 汇编命令（伪指令）实验

姓 名：李\*\*

学 号：512016\*\*\*\*

专业班级：计科卓越 1601

实验时间：2018 年 4 月 15 日

## 一、实验目的

- (1) 进一步熟悉汇编语言的汇编、连接、运行的全过程。
- (2) 了解汇编语言的程序结构、掌握动态调试工具 DEBUG 的使用。
- (3) 掌握部分伪指令的功能，编程时会使用伪指令。
- (4) 熟悉汇编语言中数据定义伪指令的书写形式。
- (5) 掌握 DEBUG 的使用。

## 二、实验要求

- (1) 仔细阅读教材中有关伪指令部分；
- (2) 仔细阅读实验教程中 DEBUG 的使用部分；
- (3) 用 DEBUG 中的 D 或 E 命令检查带符号数据和不带符号数据在内存中的表示方法；
- (4) 用 DEBUG 中的 D 命令观察 DB、DW、DD 存储整数数据的格式；
- (5) 用 DEBUG 中的 D 命令观察 DD 存储实数的格式。

## 三、实验步骤

- (1) 编写一个汇编程序，使用 DW、DB、DD 命令向内存写入数据（如图 2-1）。

```
1 CODESEG SEGMENT
2 ASSUME CS:CODESEG
3     DW 0123H,0456H,0789H,1011H
4     DB 01H,23H,04H,56H,07H,89H,10H,11H
5     DD 01230456H,07891011H
6
7     MOV AH,4CH
8     INT 21H
9
10 CODESEG ENDS
11 END
```

图 2-1 使用 DW、DB、DD 命令向内存写入数据

(2) 使用 debug 命令观察存储数据的格式。

①使用-u 命令观察指令所在地址和指令的具体实现。可以观察到指令的初始地址为 076A:0000（如图 2-2）。

```

DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Pro...
Z:\>mount c c:/masm
Drive C is mounted as local directory c:/masm\
Z:\>c:
C:\>debug 201.exe
-u
076A:0000 2301      AND     AX,[BX+DI]
076A:0002 56        PUSH    SI
076A:0003 0489      ADD     AL,89
076A:0005 07        POP     ES
076A:0006 1110      ADC     [BX+SI],DX
076A:0008 0123      ADD     [BP+DI],SP
076A:000A 0456      ADD     AL,56
076A:000C 07        POP     ES
076A:000D 8910      MOV     [BX+SI],DX
076A:000F 115604    ADC     [BP+04],DX
076A:0012 2301      AND     AX,[BX+DI]
076A:0014 1110      ADC     [BX+SI],DX
076A:0016 8907      MOV     [BX],AX
076A:0018 B44C      MOV     AH,4C
076A:001A CD21      INT     21
076A:001C 0000      ADD     [BX+SI],AL
076A:001E 0000      ADD     [BX+SI],AL
  
```

图 2-2 用 u 命令查看指令地址

②使用-d 命令查看 076A:0000 到 076A:0070 的数据内容

```

DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Pro...
Z:\>mount c c:/masm
Drive C is mounted as local directory c:/masm\
Z:\>c:
C:\>debug 201.exe
-d 076A:0
076A:0000 23 01 56 04 89 07 11 10 01 23 04 56 07 89 10 11 #.U.....#.U....
076A:0010 56 04 23 01 11 10 89 07 B4 4C CD 21 00 00 00 00 U.#.....L.!....
076A:0020 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
076A:0030 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
076A:0040 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
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 .....
  
```

图 2-3 用 d 命令查看数据

## 四、实验结果

我们使用 DW、DB、DD 写入相同的 4 个字（64 位）数据，但是通过观察，我们发现它们在内存中的位置并不是我们想的那样按顺序安放。

076A:0000 23 01 56 04 89 07 11 10      (DW)

076A:0008 01 23 04 56 07 89 10 11      (DB)

076A:0010 56 04 23 01 11 10 89 07      (DD)

我们可以发现 DB 以字节写入的数据是按顺序安放的，而 DB 和 DW 却不是这样。我们可以理解为：高地址存高位，低地址存低位。例如 0123H，用 DW 写入，01 是高位放在高地址，23 是低位放在低地址，这就体现了 DW 是按字存储的，将字拆分成两个字节存储。

## 附源代码

```
CODESEG SEGMENT
ASSUME CS:CODESEG

    DW 0123H,0456H,0789H,1011H
    DB 01H,23H,04H,56H,07H,89H,10H,11H
    DD 01230456H,07891011H

    MOV AH,4CH

    INT 21H

CODESEG ENDS

END
```