

# 计算机原理第一次实验报告

张蔚桐 2015011493 自55

2017 年 3 月 24 日

## 1 实验内容

### 1.1 DEBUG的使用练习

启动DEBUG, 用A 命令输入实验指导书中程序段, 用U 命令对此程序作反汇编, 在显示屏上逐行阅读程序, 并将机器语言和助记符对照, 体会机器码和指令助记符(尤其是指令中的立即数) 之间的对应关系。分别用G、P 及T命令执行此程序, 并随时用D 及R 命令检查有关内存单元及寄存器内容。本程序在一块数据区写入了0 F 十六个数, 用D 命令可以看到相应数据区内容在程序运行前后的变化, 用W 命令将程序存于磁盘, 用L 命令将程序取回并反汇编检查程序是否复原。

### 1.2 汇编语言上机

用文本编辑器完成以下汇编程序, 保存后采用TASM和LINK先后进行编译和连接。在debug模式下检查程序执行和内存变化情况, 发现程序完成了内存段的拷贝和语句输出

```
1 ; Experiment 1 (Tutorial Guide , p.12)
2
3 name prob12
4
5 data segment
6
7 buffer1 db 0,1,2,3,4,5,6,7,8,9
8          db 0ah,0bh,0ch,0dh,0eh,0fh
9 ;just a count
10 buffer2 db 10h dup(0)
```

```
11 mess db 'HAVE DONE',13,10,'$'  
12 ;"HAVE DONE\r\n"  
13 data ends  
14  
15 stack segment para stack;  
16 db 100 dup(?)  
17 stack ends;  
18  
19 code segment  
20 assume cs:code, ds:data, es:data, ss:stack  
21 ;Reg for visit data block  
22  
23 start: mov ax, data  
24 mov ds, ax  
25 mov es, ax  
26 ; set up the user data segment  
27  
28 lea si, buffer1  
29 ; maybe like a pointer in C  
30 lea di, buffer2  
31 mov cx, 10h  
32  
33 next: mov al, [si]  
34 ; load the *si (data in buffer1)  
35 mov [di], al  
36 ; *di=al  
37 ; may just like the exchange?  
38 inc si  
39 inc di  
40 ;++si,++di  
41 dec cx  
42 ;cx—  
43  
44 jnz next  
45 ;cpy buffer1 to buffer2  
46
```

```
47 lea dx, mess
48 ;prepare for output
49 mov ah, 9
50 int 21h
51 ;when ah == 9 means output the string in ds:dx
52
53 mov ah, 4ch
54 int 21h
55 ;when ah == 4c means return (al);
56
57 code ends
58
59 end start
```

### 1.3 选做内容

在上面汇编代码的基础上进行修改，利用循环和累加的操作完成对30H 39H 及41H 46H的写入操作

```
1 name prob13
2
3 data segment
4 buffer1 db 10h dup(0)
5 ;calloc(10);
6 mess db 'HAVE DONE',13,10,'$'
7 ;"HAVE DONE\r\n"
8 data ends
9
10 stack segment para stack
11 db 10 dup(0)
12 ;when using DEBUG, the size of stack
13 ;is no less than 6
14 stack ends
15
16 code segment
17 assume cs:code, ds:data, es:data, ss:stack
18 ;Reg for visit data block
```

```
19
20 start: mov ax, data
21 mov ds, ax
22 mov es, ax
23 ; set up the user data segment
24 lea di, buffer1
25 ; a pointer in C
26
27 mov cx, 0ah
28 mov al, 30h
29 loop1: mov [di], al
30 ; *di=al
31 inc al
32 inc di
33 ;++al,++di
34 loop loop1
35 ;cpy al to buffer2
36
37 mov cx, 06h
38 mov al, 41h
39 loop2: mov [di], al
40 inc al
41 inc di
42 loop loop2
43
44 lea dx, mess
45 ;prepare for output
46 mov ah, 9
47 int 21h
48 ;when ah == 9 means output the string in ds:d
49 mov ah, 4ch
50 int 21h
51 ;when ah == 4c means return (al);
52
53 code ends
54
```

```
55 end start
```

## 2 程序运行情况

1.1中的程序0H FH 依次存放在DS:0200 DS:020F 中。首先将SI、CX、AL 初始化，每次将AL 中的值传到DS:[SI] 指向的内存中，地址SI、内容AL 加1，CX 计数减1。循环10 次后，通过INT 结束程序。

1.2中的程序通过累加和循环完成内存块之间的拷贝工作

1.3中的选做程序通过累加和两段循环的方式完成对内存块的写入工作，和1.2不同的是，采用了LOOP命令进行流程控制

## 3 总结

通过这次实验，我熟悉了DOS环境和DEBUG的相关使用方法，对汇编语言也有了更多的体验和了解。为后续实验做好了准备

```

C:\>debug
-a
073F:0100 MOV SI,200
073F:0103 MOV CX,10
073F:0106 MOV AL,0
073F:0108 MOV [SI],AL
073F:010A INC SI
073F:010B INC AL
073F:010D DEC CX
073F:010E JNZ 108
073F:0110 INT 3
073F:0111
-U
073F:0100 BE0002      MOV     SI,0200
073F:0103 B91000      MOV     CX,0010
073F:0106 B000        MOV     AL,00
073F:0108 8804        MOV     [SI],AL
073F:010A 46          INC     SI
073F:010B FEC0        INC     AL
073F:010D 49          DEC     CX
073F:010E 75F8        JNZ     0108
073F:0110 CC          INT     3
073F:0111 F0          LOCK
073F:0112 46          INC     SI
073F:0113 7400        JZ      0115
073F:0115 00B200B2     ADD     [BP+SI+B200],DH
073F:0119 16          PUSH    SS
073F:011A 99          CWD
073F:011B 002E072E     ADD     [2E07],CH
073F:011F 07          POP     ES
-G

AX=0010 BX=0000 CX=0000 DX=0000 SP=00FD BP=0000 SI=0210 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0110  NU UP EI PL ZR NA PE NC
073F:0110 CC          INT     3
-D 0200
073F:0200 00 01 02 03 04 05 06 07-08 09 0A 0B 0C 0D 0E 0F .....
073F:0210 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
073F:0220 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
073F:0230 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
073F:0240 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
073F:0250 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
073F:0260 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
073F:0270 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
-

```

图 1: 1.1运行截图

```

C:\>debug test2.exe
-u
0774:0000 B86A07      MOV     AX,076A
0774:0003 8ED8        MOV     DS,AX
0774:0005 8EC0        MOV     ES,AX
0774:0007 BE0000      MOV     SI,0000
0774:000A BF1000      MOV     DI,0010
0774:000D B91000      MOV     CX,0010
0774:0010 8A04        MOV     AL,[SI]
0774:0012 8805        MOV     [DI],AL
0774:0014 46          INC     SI
0774:0015 47          INC     DI
0774:0016 49          DEC     CX
0774:0017 75F7        JNZ     0010
0774:0019 BA2000      MOV     DX,0020
0774:001C B409        MOV     AH,09
0774:001E CD21        INT     21
-g
HAVE DONE

Program terminated normally
-d 076A:0000
076A:0000 00 01 02 03 04 05 06 07-08 09 0A 0B 0C 0D 0E 0F .....
076A:0010 00 01 02 03 04 05 06 07-08 09 0A 0B 0C 0D 0E 0F .....
076A:0020 48 41 56 45 20 44 4F 4E-45 0D 0A 24 00 00 00 00 HAVE DONE.$....
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: 1.2运行截图