山东大学 计算机科学与技术 学院

汇编语言 课程实验报告

学号：202120130276	姓名：王云强	班级：21.2 班
实验题目：实验一		
实验学时：2	实验日期：2023.10.20	

实验目的：

1. 学习使用和熟悉 MASM、LINK、DEBUG、EDIT、TD 等汇编工具。

2. 掌握一般汇编语言程序的编程框架。

3. 学习汇编程序的基本编写习惯，包括但不限于寄存器使用规范、变量/标号命名、注释、对齐、分段、缩进等。

实验环境：Windows10、DOSBox-0.74、Masm64

源程序清单：

1. sample.asm（示例 1.1 源程序）

```
;PROGRAM TITLE GOES HERE -- Compare string
;××××××××××××××××××××××××××××××××××××××××××××××××××××××××
DATAREA SEGMENT    ;define data SEGMENT
string1     DB  'Move the cursor backward.'
string2     DB  'Move the cursor backward.'
;
mess1       DB  'Match.',13,10,'$'
mess2       DB  'No match!',13,10,'$'

DATAREA     ENDS
;××××××××××××××××××××××××××××××××××××××××××××××××××××××××

PROGNAM         SEGMENT         ;define code segment
;-------------------------------------------------------

main    proc    far
        ASSUME      CS:PROGNAM,DS:DATAREA,ES:DATAREA
START:              ;starting execution address

;set up stack for return
    PUSH    DS  ;save old data segment 压栈，保持当前状态
    SUB     AX,AX   ;put zero in AX
    PUSH    AX      ;save it on stack
;set DS register to current data segment
    MOV AX,DATAREA; datarea segment addr
    MOV DS,AX   ;into DS register
    MOV ES,AX   ;into ES register
;MAIN PART OF PROGRAM GOES HERE
    LEA     SI,string1  ;将string1的偏移地址放到SI中，使SI指向string1
    LEA     DI,string2  ;将string2的偏移地址放到DI中，使DI指向string2
```

```
30      LEA     DI,string2  ;将string2的偏移地址放到DI中，使DI指向string2
31      CLD                 ;将标志寄存器Flag的方向标志位DF清零。
32      MOV     CX,25       ;为后面循环做准备，CX=25，循环25次判断字符串是否一致
33      REPZ    CMPSB
34      JZ      MATCH
35      LEA     DX,mess2
36      JMP     SHORT DISP
37  MATCH:
38      LEA     DX,mess1
39  DISP:
40      MOV     AH,09
41      int     21H
42      RET                                 ;return to DS
43
44  main    ENDP                            ;end of main part of program
45  ;-------------------------------------------------------
46  PROGNAM         ENDS                    ;end of code segment
47  ;*******************************************************************
48                  END             START   ;end assembly
```

编译及运行结果：

1. 我们先通过 masm 和 link 来生成可执行文件。

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

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

  51746 + 464798 Bytes symbol space free

      0 Warning Errors
      0 Severe  Errors

C:\>link sample

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

Run File [SAMPLE.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
LINK : warning L4021: no stack segment
```

2. 此时 exe 的执行结果是：

```
C:\>link sample

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

Run File [SAMPLE.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
LINK : warning L4021: no stack segment

C:\>sample
Match.
```

3. 修改完数据后：

```
Copyright (C) Microsoft Corp 1981-1985, 1987.  All rights reserved.

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

  51746 + 464798 Bytes symbol space free

      0 Warning Errors
      0 Severe  Errors

C:\>link sample

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

Run File [SAMPLE.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
LINK : warning L4021: no stack segment

C:\>sample
No match!
```

4. 通过 Debug 修改字符串

①先使用 debug 程序

```
C:\>sample
Match.

C:\>debug sample.exe
-g
Match.
```

②查看程序

```
076F:000B 8D360000      LEA     SI,[0000]
076F:000F 8D3E1900      LEA     DI,[0019]
076F:0013 FC            CLD
076F:0014 B91900        MOV     CX,0019
076F:0017 F3            REPZ
076F:0018 A6            CMPSB
076F:0019 7406          JZ      0021
076F:001B 8D163B00      LEA     DX,[003B]
076F:001F EB04          JMP     0025
-u
076F:0021 8D163200      LEA     DX,[0032]
076F:0025 B409          MOV     AH,09
076F:0027 CD21          INT     21
076F:0029 CB            RETF
076F:002A 7383          JNB     FFAF
076F:002C C4068BB6      LES     AX,[B68B]
076F:0030 FA            CLI
076F:0031 FE81E6FF      INC     BYTE PTR [BX+DI+FFE6]
076F:0035 00C6          ADD     DH,AL
076F:0037 82FBFE        CMP     BL,FE
076F:003A 002B          ADD     [BP+DI],CH
076F:003C C0            DB      C0
076F:003D 50            PUSH    AX
076F:003E 8D86FBFE      LEA     AX,[BP+FEFB]
```

③断点设置在程序的主要部分之前

```
-g0b

AX=076A  BX=0000  CX=007A  DX=0000  SP=FFFC  BP=0000  SI=0000  DI=0000
DS=076A  ES=076A  SS=0769  CS=076F  IP=000B   NV UP EI PL ZR NA PE NC
076F:000B 8D360000      LEA     SI,[0000]                       DS:0000=6F4D
```

④使用 d 查看数据段

```
-d0
076A:0000  4D 6F 76 65 20 74 68 65-20 63 75 72 73 6F 72 20   Move the cursor
076A:0010  62 61 63 6B 77 61 72 64-2E 4D 6F 76 65 20 74 68   backward.Move th
076A:0020  65 20 63 75 72 73 6F 72-20 62 61 63 6B 77 61 72   e cursor backwar
076A:0030  64 2E 4D 61 74 63 68 2E-0D 0A 24 4E 6F 20 6D 61   d.Match...$No ma
076A:0040  74 63 68 21 0D 0A 24 00-00 00 00 00 00 00 00 00   tch!..$.........
076A:0050  1E 2B C0 50 B8 6A 07 8E-D8 8E C0 8D 36 00 00 8D   .+.P.j......6...
076A:0060  3E 19 00 FC B9 19 00 F3-A6 74 06 8D 16 3B 00 EB   >........t...;..
076A:0070  04 8D 16 32 00 B4 09 CD-21 CB 73 83 C4 06 8B B6   ...2....!.s.....
```

⑤看通过 e 命令修改数据区的字符串

```
-e29
076A:0029  62.66   61.6f   63.72   6B.77   77.61   61.72   72.64
076A:0030  64.2e   2E.20

-d0
076A:0000  4D 6F 76 65 20 74 68 65-20 63 75 72 73 6F 72 20   Move the cursor
076A:0010  62 61 63 6B 77 61 72 64-2E 4D 6F 76 65 20 74 68   backward.Move th
076A:0020  65 20 63 75 72 73 6F 72-20 66 6F 72 77 61 72 64   e cursor forward
076A:0030  2E 20 4D 61 74 63 68 2E-0D 0A 24 4E 6F 20 6D 61   . Match...$No ma
076A:0040  74 63 68 21 0D 0A 24 00-00 00 00 00 00 00 00 00   tch!..$.........
076A:0050  1E 2B C0 50 B8 6A 07 8E-D8 8E C0 8D 36 00 00 8D   .+.P.j......6...
076A:0060  3E 19 00 FC B9 19 00 F3-A6 74 06 8D 16 3B 00 EB   >........t...;..
076A:0070  04 8D 16 32 00 B4 09 CD-21 CB 73 83 C4 06 8B B6   ...2....!.s.....
```

⑥ 运行，显示不匹配。

```
-g
No match!

Program terminated normally
```

问题及收获：

1.熟悉在DOSBox 环境下使用 masm 进行编译、使用link进行链接、使用debug或td进行断点调试。

2.熟悉汇编程序的编写框架，尝试改变汇编代码重新编译链接运行观察寄存器的变化。

3.尝试使用汇编开发的IDE例如MASMPlus 进行编程的高效编程开发、调试。

4.运行了课本实例1.1的代码，熟悉了一些转移指令、文本比较指令、条件跳转指令、无条件跳转指令等。

5.对代码有了更深刻的认识。从29行开始解读，首先通过 LEA 指令，将string1和string2的地址分别放入了SI和DI寄存器中（string1和string2就是MATCH和NO MATCH）。之后CLD命令是方向标志位DF清零，即默认从前向后遍历，接下来检查字符串。MOV CX, 25是将立即数25放入CX循环计数器中，REPZ CMPSB结合CX寄存器初始值25，保证一共比较25个字符结束循环。JZ MATCH意思是如果经过字符全部比较之后，全部字符一样，则转向 MATCH 部分，即将mess1的地址（存放Match.）放入DX寄存器。而如果不一样，则不会跳向 MATCH，而是会继续执行，将 mess2地址放入DX寄存器，之后进行段内短转移，跳到 DISP

部分。即在这里相当于 C 语言中的 IF 判断，如果 25 次循环都一样，则将 mess1 放入 DX 寄存器，反之有一次不一样，则将 mess2 放入 DX 寄存器，最终都进入了 DISP 部分代码。DISP 部分代码分三行，第一行是 MOV AH 09，是将 09 放入 AH 寄存器，第二行是 int 21H，即调用 21H 号中断，调用 DOS 系统工作，去查找 AH 中的内容，结合起来就是实现 DOS 调用的 09 号功能，作用是将 DX 寄存器储存的数据作为地址，访问该地址的数据，并打印出来（即打印 Match.或 No match!）。这两句进行了系统调用。第三行就是 RET，即告诉系统代码段执行结束，接下来返回数据段。

6.在需要直接调用程序查看程序运行结果时，需要先 masm sample (sample 为.ASM 文件)，之后再回车三下，因为这时编译器会询问三个问题，大部分是问是否需要详细信息，回车默认是不需要。之后再 link sample，再回车三下(同上)，之后再 sample 回车进行执行。