华 中 科 技 大 学

课 程 实 验 报 告

课程名称：汇编语言程序设计实验

实验名称：汇编语言程序设计实验一

实验时间：**2019-9-16，19：00-22：20**

实验地点：南一楼

指导教师：鲁宏伟

专业班级：信息安全1802班

学 号：U201814864

姓 名：师天硕

报告日期：2019年9月16日

**成绩评定**

|  |  |  |
| --- | --- | --- |
| 实验完成质量（70分） | 报告撰写质量（30分） | 总成绩 |
| 实验步骤清晰、详细、深入，实验记录真实完整等 | 报告规范、完整、通顺、详实 |
|  |  |  |

**目 录**

[1实验目的 1](#_Toc19919291)

[2实验内容 1](#_Toc19919292)

[2.1任务1 1](#_Toc19919293)

[2.2任务2 2](#_Toc19919294)

[3实验过程 3](#_Toc19919295)

[3.1开发环境 3](#_Toc19919296)

[3.2任务一 3](#_Toc19919297)

[3.2.1 实验步骤 3](#_Toc19919298)

[3.2.4 修改后的源程序 3](#_Toc19919299)

[3.2.3 实验记录 4](#_Toc19919300)

[3.3任务二 6](#_Toc19919301)

[3.3.1实验步骤 6](#_Toc19919302)

[3.3.2修改后的源程序 6](#_Toc19919303)

[3.3.3实验记录 7](#_Toc19919304)

[4心得与体会 8](#_Toc19919305)

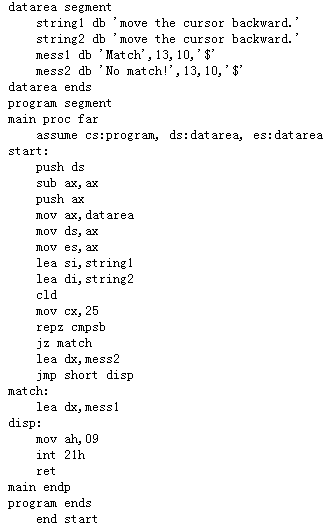
# 1实验目的

1. 掌握汇编源程序编辑工具、汇编程序、连接程序、调试工具 TD 的使用；
2. 理解数、符号、寻址方式等在计算机内的表现形式；
3. 理解指令执行与标志位改变之间的关系；
4. 熟悉常用的 DOS 功能调用；
5. 熟悉分支、循环程序的结构及控制方法，掌握分支、循环程序的调试方法；
6. 加深对转移指令及一些常用的汇编指令的理解。

# 2实验内容

## 2.1任务一

阅读以下程序，根据指令的执行流程，说明程序实现的目标，并采用汇编程序对改代码进行编辑、编译、连接和调试。



要求：

1. 查阅资料，说明指令“cld”和“repz cmpsb”的功能；
2. 查阅资料，说明“int 21h”的功能；
3. 借助TD工具，观察指令“repz cmpsb”完成后，哪些标志位发生了变化。
4. 程序中的两条“push”指令的作用是什么，有没有感觉代码中缺少与“push”指令相关的操作指令？如果有，尝试完善改短代码，并观察程序运行效果。
5. 运行程序，说明当前程序运行的结果，然后修改字符串string2，让程序产生不同的结果。

## 2.2任务二

阅读下列程序, 并指出程序执行之后, 以BUF2、BUF3、BUF4 为首址的3 个字节存储区中存放的数据。

|  |  |
| --- | --- |
|  |  |

要求：

1. 分别记录执行到“ MOV CX，10”和“INT 21H”之前的(BX)、(BP)、(SI)、(DI)各是多少。
2. 记录程序执行到退出之前数据段开始，40个字节的内容，指出程序运行结果是否与设想的一致。
3. 在标号LOPA前加上一段程序，实现新的功能：先显示提示信息“Press any key to begin!”, 然后，在按了一个键之后继续执行 LOPA 处的程序。

操作提示：使用 TD.EXE 调试程序时，应先单步执行各个语句，每执行一条语句，都应观察数据段中的内容以及相应寄存器的变化。首先注意观察对DS寄存器的赋值过程，并在TD的数据窗口定位待观察的数据区位置。其次，单步执行循环体两遍且正确理解了循环体语句的含义后，可在“MOV AH, 4CH”处设置断点，然后直接执行到断点处，回答(1)和(2)的问题。完成（3）的内容，涉及到“INT 21H”相关的输入和输出操作，参考任务1中显示字符串的方法。

## 

# 3实验过程

## 3.1开发环境

本次实验中使用的环境配置如下：

1. 虚拟机版本：DOSBox v0.74-3

打开后自动执行以下命令，挂载并进入目录，设置环境变量

**mount x C:\80x86-asm-learning**

**set PATH=%PATH%;x:\usr;x:\usr\masm615\BIN;**

**x:**

（2）编译器及其版本：masm6.11

（3）调试工具：TD

（4）编程环境：Visual Studio Code

## 3.2任务一

### 3.2.1 实验步骤

①阅读代码，查看资料，确定程序功能

②准备上机实验环境，编写代码。

③经MASM汇编，LINK连接后并且确认源程序正确无误后，使用TD对连接后的EXE程序进行调试，打断点，在“repz cmpsb”指令前后记录标志位。

④查阅资料，修改程序，观察结果是否相同

⑤修改程序使得结果不同

### 3.2.4 修改后的源程序

datarea segment

    string1 db 'Move the cursor backwardx'

    string2 db 'Move the cursor backward.'

    mess1 db 'Match.',13,10,'$'

    mess2 db 'No Match!',13,10,'$'

datarea ends

prognam segment

main proc far

    assume cs:prognam,ds:datarea,es:datarea

start:

    mov ax,datarea

    mov ds,ax

    mov es,ax

    lea si,string1

    lea di,string2

    cld

    mov cx,25

    repz cmpsb

    jz match

    lea dx,mess2

    jmp short disp

match:

    lea dx,mess1

disp:

    mov ah,09

    int 21h

    mov ax 4c00h

    int 21h;

main endp

prognam ends

end start

### 3.2.3 实验记录

1. 查阅资料，说明指令“cld”和“repz cmpsb”的功能；

8086CPU提供了一组处理主存中连续存放的数据串的指令——串操作指令。每执行一次串操作指令，作为源地址指针的SI和作为目的地址指针的DI将自动修改。而修改的方向由标志位DF决定。

执行指令CLD指令后，DF=0，表示地址指针增，执行STD指令，DF=1，地址指针减。

CMPSB 指令是串比较指令，它可以比较原串和目的串是否相同，并根据减法结果设置标志位。修改SI、DI寄存器的值，指向下一个元素。B表示是字节串。

REPZ指令是重复前缀指令，把后面的CMPS指令循环执行。每执行一次指令CX减1并判断标志位。CX=1或ZF=1时结束。

在之前令CX为25，因此repz就是遍历判断两个串是否相等。讲结果存在ZF标志位中。

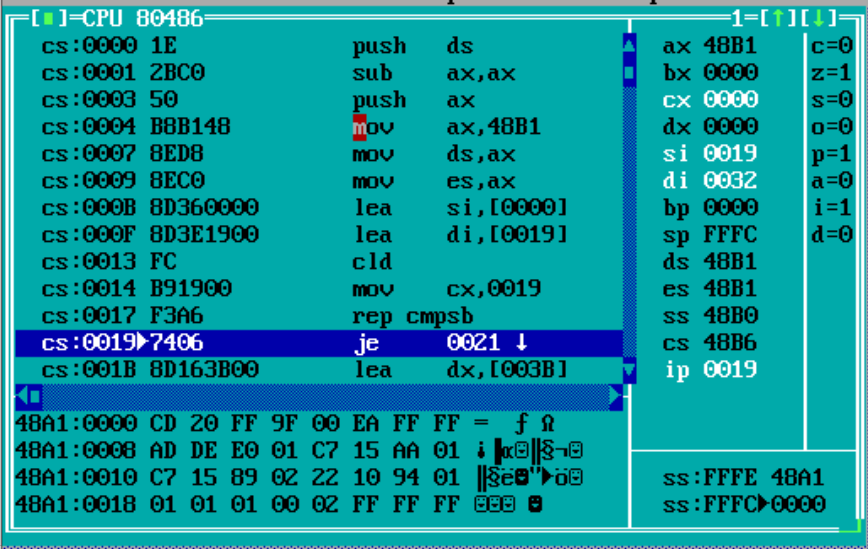
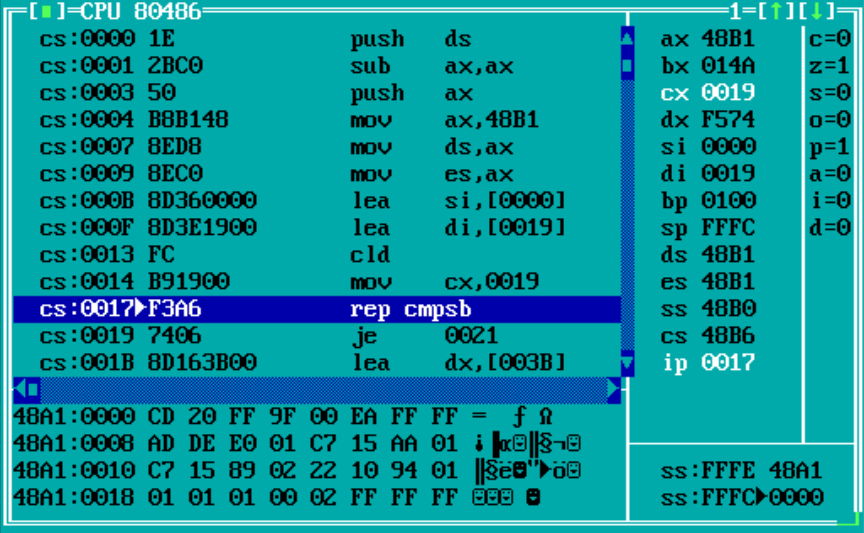
1. 查阅资料，说明“int 21h”的功能；

Int 21h是由DOS提供的系统功能调用，一组实现特殊功能的子程序供程序员在编写自己的程序时调用，以减轻编程的工作量。涉及屏幕显示、文件管理、I/O管理等等，每个子程序都有一个功能号，在调用int 21h之前传入寄存器AH中。

这里之前讲AH置为9，表示输出DS:DX的以“$”结尾的字符串，

1. 借助TD工具，观察指令“repz cmpsb”完成后，哪些标志位发生了变化。

如下图1所示，指令“repz cmpsb”完成前后标志位没有发生变化，说明匹配过程中，字节串都相同。



**图1 指令“repz cmpsb”完成前（左）、完成后（右）的标志位**

1. 程序中的两条“push”指令的作用是什么，有没有感觉代码中缺少与“push”指令相关的操作指令？如果有，尝试完善改短代码，并观察程序运行效果。

程序加载后，DS存放着程序所在内存的段地址，这个内存区域的前256个字节存放在的是PSP（程序段前缀）。PSP是DOS操作系统在执行程序时为程序所建立的一个信息块，里面包括了传递给待运行程序的命令行参数，程序运行结束时返回DOS所需的地址等有用的信息。

这里讲DS和0压入栈内，栈里存储的是程序结束后应该返回DOS的代码逻辑地址，在程序最后执行指令ret，即pop ip, pop cs时，通过修改CS、IP寄存器返回DOS系统，这是一种存储上下文的方式。

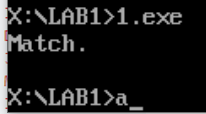
现在通常使用的方式是，在最后直接调用int 21h;DOS系统调用返回DOS。用如下语句

mov ax 4c00h

int 21h;

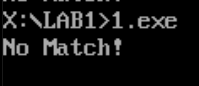
1. 运行程序，说明当前程序运行的结果，然后修改字符串string2，让程序产生不同的结果。

运行程序结果如图3。两个字符串是相同的，因此输出mess1



**图2程序1的运行结果**

将string1 db 'Move the cursor backward.'改为string1 db 'Move the cursor backwardx'在未改变字符串长度的情况下，使两个字符串不同。运行结果如图4。



**图3程序1修改后的运行结果**

## 3.3任务二

### 3.3.1实验步骤

①准备上机环境，编写实验程序

②经MASM汇编，LINK连接后并且确认源程序正确无误后，使用TD对连接后的EXE程序进行调试，打断点，执行至mov cx,10后记录(BX)，(BP)，(SI)，(DI)的值。继续打断点至int 21h处，记录(BX)，(BP)，(SI)，(DI)的值。然后在内存数据窗口跳转至DS段，记录DS段的前40个字节的数据。

③然后重新更改程序，通过系统21H中断的9号调用显示相应的字符串，然后再通过系统21H中断的1号调用读取输入字符，中断返回，继续执行LOPA处的程序。

### 3.3.2修改后的源程序

.386

stack segment use16 stack

    db 200 dup(0)

stack ends

data segment use16

    buf1    db 0,1,2,3,4,5,6,7,8,9

    buf2    db 10 dup (0)

    buf3    db 10 dup (0)

    buf4    db 10 dup (0)

    string  db 'Press any key to begin!', '$'

data ends

code segment use16

    assume cs:code, ds:data, ss:stack

start:  mov ax, data

        mov ds, ax

        mov si, offset buf1

        mov di, offset buf2

        mov bx, offset buf3

        mov bp, offset buf4

        mov cx, 10

        mov dx, offset string

        mov ah, 9

        int 21h

        mov ah, 1

        int 21h

lopa:   mov al, [si]

        mov [di], al

        inc al

        mov [bx], al

        add al, 3

        mov ds:[bp], al

        inc si

        inc di

        inc bp

        inc bx

        dec cx

        jnz lopa

        mov ah, 4ch

        int 21h

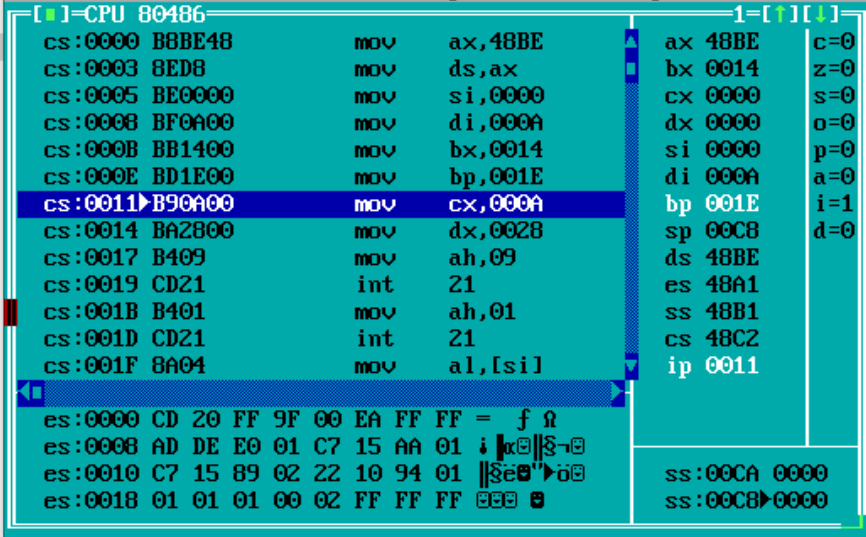
code ends

end start

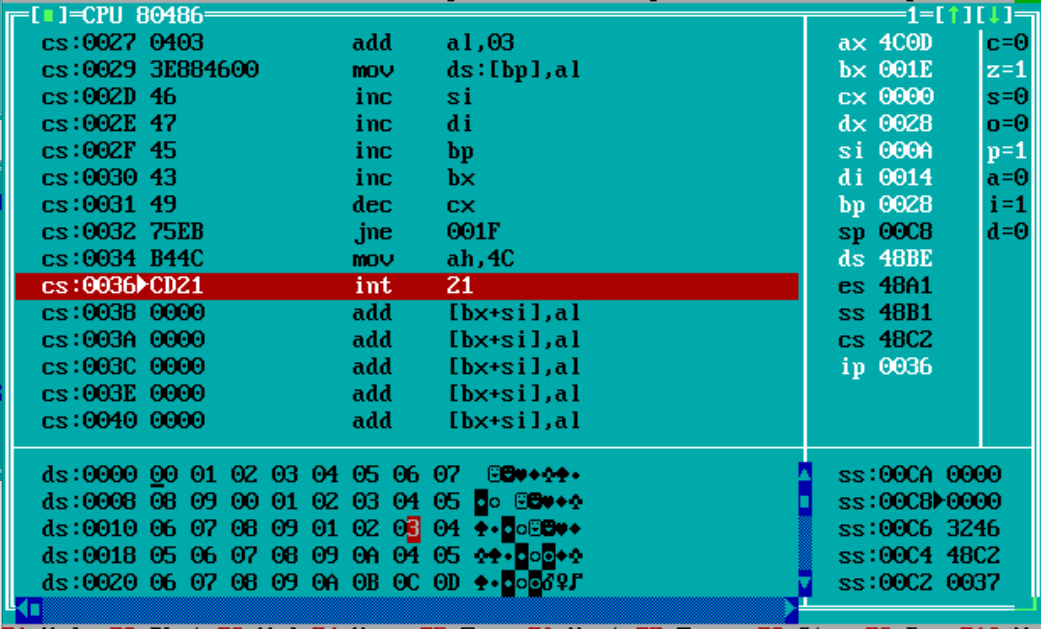
### 3.3.3实验记录

1. 分别记录执行到“ MOV CX，10”和“INT 21H”之前的(BX)、(BP)、(SI)、(DI)各是多少。

如图所示

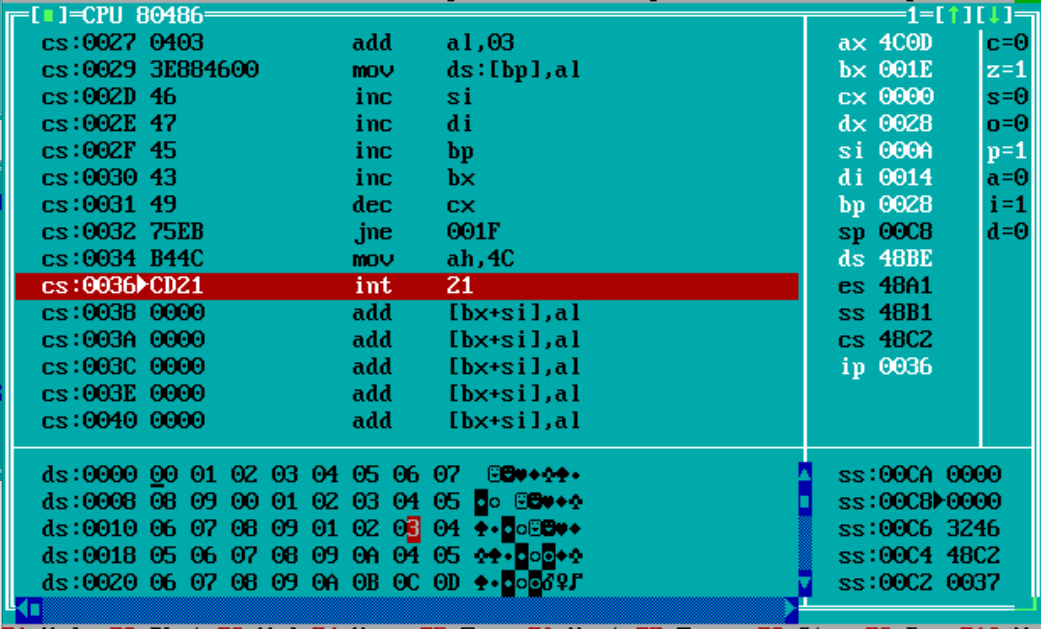


**图4 执行到“ MOV CX，10”之前寄存器的值**



**图5 执行到“INT 21H”之前寄存器的值**

1. 记录程序执行到退出之前数据段开始，40个字节的内容，指出程序运行结果是否与设想的一致。



**图6 执行到“INT 21H”之前数据段开始40个字节的值**

1. 在标号LOPA前加上一段程序，实现新的功能：先显示提示信息“Press any key to begin!”, 然后，在按了一个键之后继续执行 LOPA 处的程序。

在lopa之前增加如下程序。利用

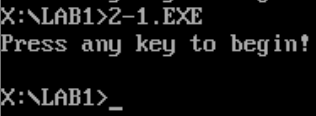
mov ah, 9

int 21h

mov ah, 1

int 21h

实现功能，如下图所示



**图6 修改后程序的运行结果**

# 4心得与体会

本次实验是学习汇编语言的第一次实验。在这次实验中，我第一次编写了完整的汇编语言程序，对汇编程序的基本结构有了了解，学会了一些伪指令的使用方法。学会利用masm对程序进行编译链接，在dosbox虚拟机中运行。同时学习了td调试工具的使用方法，之前学习的抽象的程序运行过程具现的变成了寄存器和内存中的数据的变化，这大大加深了我对汇编语言的理解，在以后的学习中，对于不明白的汇编语句可以使用调试，实验每一个指令每一段代码的实际表现。具体的新学到的知识方面，我了解到了x86汇编对字符串的串指令、与DOS系统交互的系统调用、DOS系统中程序运行中PSP(程序段前缀)的相关知识等。

你