1. 实验目的与要求

（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

任务 1.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。

```
datarea 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
program segment
main proc far
    assume cs:program, ds:datarea, es:datarea
start:
    push ds
    sub ax,ax
    push ax
    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
    ret
main endp
program ends
    end start
```

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？分析一下会不会导致出现“栈不平衡”的现象。

(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。

任务 2. 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。

```
.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)
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
```

```
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
```

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 任务1：

3.1.1 实验步骤

1. 配置汇编环境，打开vs code，输入实验代码，保存为.asm文件。

2. 查阅资料，了解指令“cld”和“repz cmpsb”的功能。

3. 查阅资料，了解指令“int 21h”的资料。

4. 打开td工具，按 F7到指令“repz cmpsb”，再按F7，观察运行该指令后标志位的变化。

5. 运行程序，观察结果。然后修改string2的内容为‘Hello World!’，运行观察结果。

3.1.2 实验记录与分析

1. 输入代码

```
datarea 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
program segment
main proc far
        assume cs:program, ds:datarea, es:datarea
   start:
        push ds
        sub ax,ax
        push ax
        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
        ret
   main endp
   program ends
        end start
```

2. 观察标志位并记录

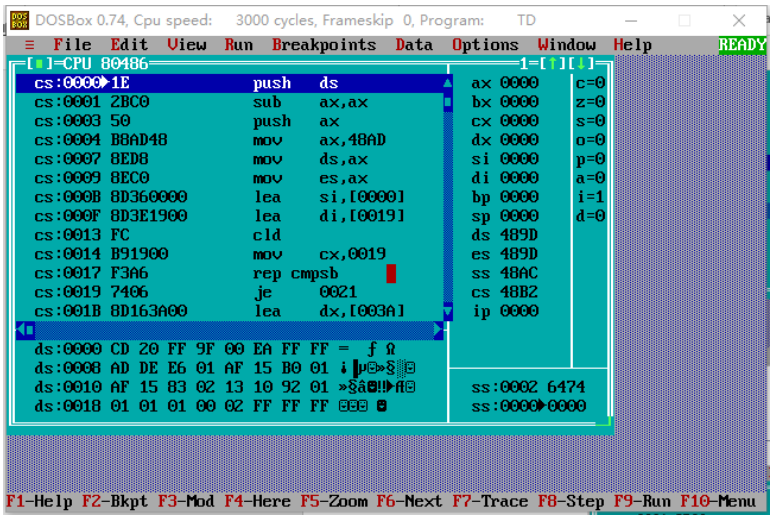

图 1-2-1

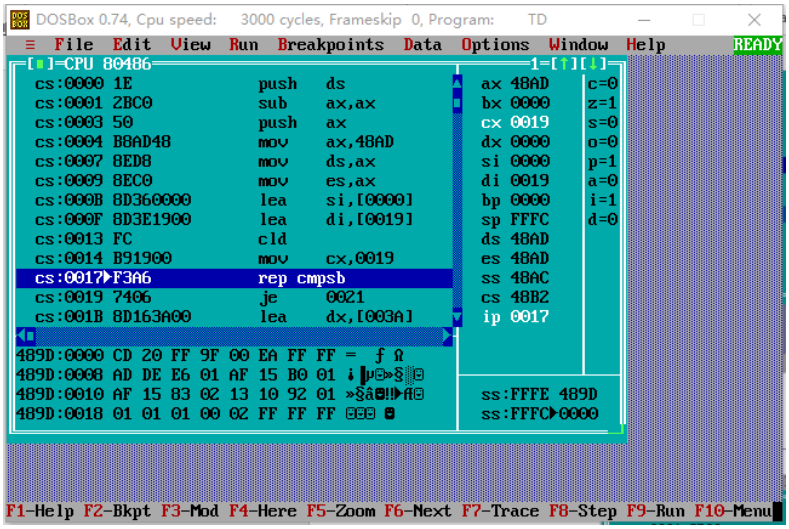

图 1-2-2

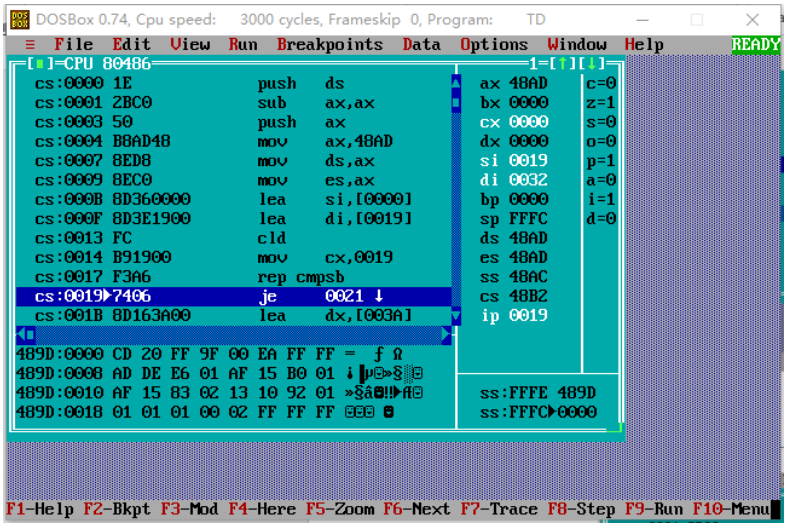

图 1-2-3

3. 程序运行结果

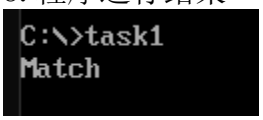

图 1-3-1

修改 string2 后的程序运行结果

```
string2 db 'Hello World!'
```

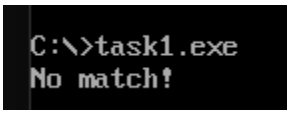

图 1-3-2

3.1.3 回答问题

（1） “cld”的功能是复位方向标志：DF<-0。

“repz cmpsb”的功能是如果ds:si和es:si所指向的两个字节相等，则继续比较。如果不相等，就停止循环。但是此时si和di已经自动加1，若找不到相等的那两个数，就要把si和di减1。

（2） “int 21h”是汇编的中断调用。此处输出一个以‘$’结尾的字符串。

（3） 指令“repz cmpsb”完成后，标志位相对上一步没有发生变化，相对一开始变化是z=0,p=0变为z=1,p=1。

（4） 先使堆栈指针sp减2，然后把一个字操作数存入堆栈顶部。进栈时，低字节存放于低地址，高字节存放于高地址，SP相应向低地址移动2字节单元。

缺少出栈指令POP。

会导致。由于堆栈的栈顶和内容随着程序的执行不断变化，因此编程时要注意入栈和出栈的数据要成对，要保持堆栈平衡。缺少POP指令，没有出栈，会造成堆栈不平衡。

（5）程序运行结果是：

Match

修改string2后的结果是：

No match!

3.2. 任务2：

3.2.1 实验步骤

1. 打开vs code，输入实验代码，保存为.asm文件。

2. 链接task2，打开td工具，一直按F7，按到“MOV CX,10”指令和“INT 21H”指令停止，观察BX,BP,SI,DI的值。

3. 记录程序执行到退出之前数据段开始，40 个字节的内容。

4. 在DATA SEGMENT USE16下面写

FLAG DB ”Press any key to begin!$”

在MOV CX,10后面 LOPA段前面写

LEA DX,FLAG

MOV AH,09H

INT 21H

MOV AH,01H

INT 21H

然后运行程序，敲入回车键，查看是否符合预期。

3.2.2 实验记录与分析

1.输入代码

```
.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)
```

```
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
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
```

2. 运行到相应指令，观察结果

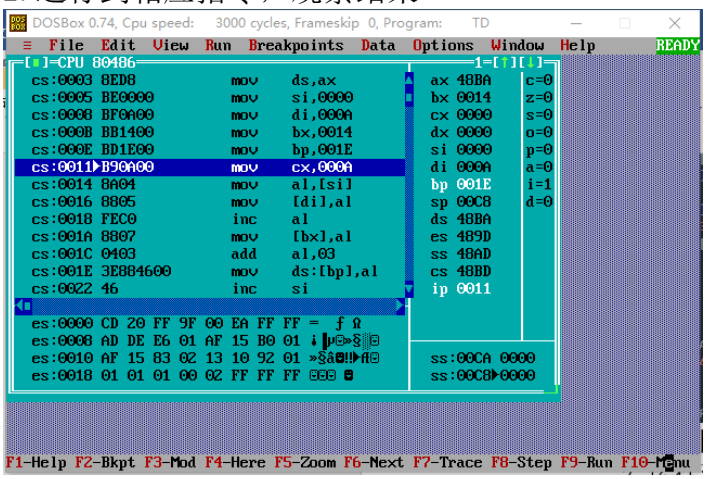

图2-2-1

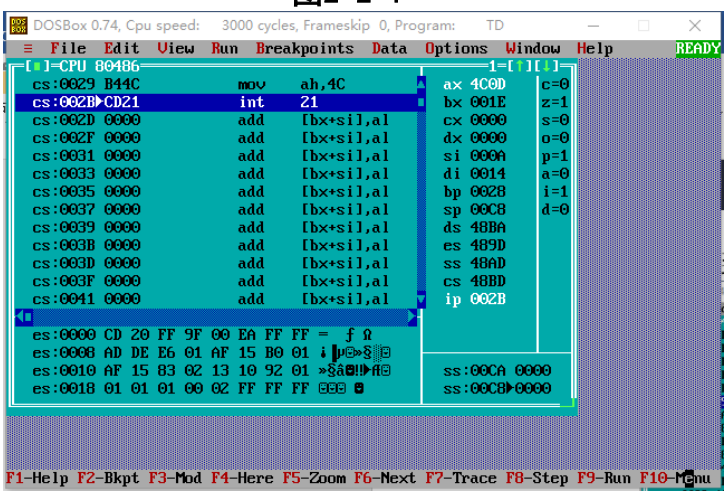

图2-2-2

3. 记录程序执行到退出之前数据段开始，40 个字节的内容

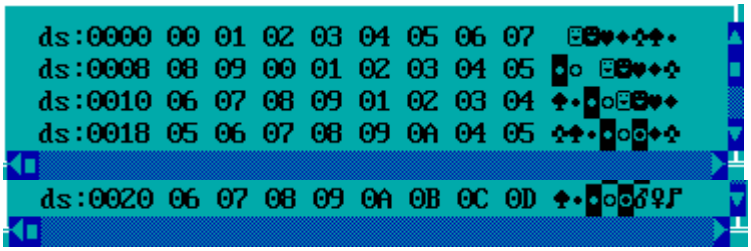

图2-3

4. 更改代码，运行程序

```
        .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)
FLAG    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
        LEA DX,FLAG
        MOV AH,09H
        INT 21H
        MOV AH,01H
        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
```

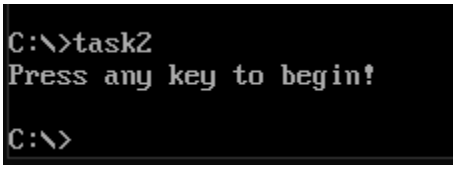

图2-4

3.2.3 回答问题

（1） 执行到“MOV CX,10”之前：

BX=0014 BP=001E SI=0000 DI=000A

执行到“INT 21H”之前

BX=001E BP=0028 SI=000A DI=0014

（2）记录如下

00	01	02	03	04	05	06	07
08	09	00	01	02	03	04	05
06	07	08	09	01	02	03	04
05	06	07	08	09	0A	04	05
06	07	08	09	0A	0B	0C	0D

设想为00-09后面都是00，与设想不一致。

（3） 在DATA SEGMENT USE16下面写

```
FLAG DB ”Press any key to begin!$”
```

在“MOV CX,10”下面和LOPA段前面写

```
LEA DX,FLAG
MOV AH,09H
INT 21H
MOV AH,01H
INT 21H
```

全部代码如上文3.2.2.4所示。

4.总结与体会

在刚接触汇编时练习这两个实验，困难颇多，如何寻址，指令的含义都不是太懂。通过这次实验，我了解了汇编程序编写的大致过程，基本掌握了汇编程序，连接程序，TD工具的使用。也大致认识了一些指令的含义，认识了指令执行与标志位改变的一些关系，熟悉了DOS一部分功能的调用。

通过这次实验，我从一开始对汇编一些畏惧，转变成对汇编的兴趣，通过了解汇编的一些指令，逐步深入对汇编的认识，让我对汇编的心态发生改变，希望能通过后续的实验，加强对汇编的学习，不断深化汇编的认知，增强汇编水平，多掌握一些技术，为后续工作打好基础。