第二次实验报告

一、实验目的

1. 掌握串操作指令的使用；
2. 理解算数运算指令、BCD 码调整指令；
3. 熟练应用 DEBUG 调试汇编程序；

二、设计说明

[实验内容]

使用串操作指令 MOVSB 对一段内存单元中的内容（1，2，3，……，100）进行转移，再使用串操作指令 CMPS 对转移的内容进行比较来判断传输是否正确，若不正确则进行重新传输；接着对已经正确传输的 100 个数据进行无符号型的累加，最后使用 BCD 调整码，最终将答案放入内存，并将其显示在屏幕上。

[步骤分析]

(1) 将 1，2，3，……，100 存入数据段相应内存中；
(2) 将 1，2，3，……，100 转存入附加段中，并进行比较；
(3) 累加数据，并进行调整；

[前置知识]

(1) 数据的存入、转出与比较

串操作指令

① MOVS

格式：　[REP]MOVSB；字节传送
　　[REP]MOVSW；字传送
操作：　DS：SI→ES：DI
说明：　前缀 REP 指令用于重复，直到计数器 CX=0 为止
参数：
　　设置方向标志 DF（CLD，STD）
　　源串偏移量送给 SI，目的串偏移量送给 DI
　　传送次数送给 CX

② CMPS

格式：　[REPZ/REPNZ]CMPSB；
　　[REPZ/REPNZ]CMPSW；
操作：　（DS:SI）--（ES:DI）
说明：　REPZ:串未比较完成且已比较部分相等时继续比较
　　REPNZ:串未比较完成且已比较部分不等时继续比较

(2) 数据的累加和调整

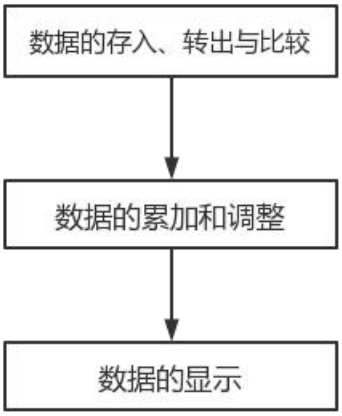
以字节为单位进行存储累加，通过循环完成。

以 AAM 对 AX 中的数据进行调整。

(3) 数据的显示

利用循环取出每一位数字，调整为 ASCII，并用 02H 号指令输出字符，通过循环输出每一个字符。

三、程序流程



四、实验结果

```
DOSBox 0.74-3, Cpu speed:   3000 cycles, Frameskip  0, Program:   DOSBOX

D:\>tasm /zi 4.ASM >X:\ASM.LOG

Turbo Assembler  Version 4.1  Copyright (c) 1988, 1996 Borland International

Assembling file:   4.ASM
Error messages:    None
Warning messages:  None
Passes:            1
Remaining memory:  465k


D:\>if exist 4.OBJ tlink /v/3 4.obj >X:\LINK.LOG

Turbo Link  Version 7.1.30.1. Copyright (c) 1987, 1996 Borland International
Warning: No stack


D:\>4.exe

MATCH
5050
(END)Here is the end of the program's output

Do you need to keep the DOSBox [Y,N]?
```

五、心得体会

学习了传操作的基本使用方法，输出字符的基本方法，学会了使用简单的汇编语言循环。

附：代码实现

```
DATA SEGMENT;定义数据段
      MKEY DB 0DH,0AH,'MATCH',0DH,0AH,'$';定义匹配时的输出字符
      NMKEY DB 0DH,0AH,'NOMATCH',0DH,0AH,'$';定义不匹配时的输出字
符
      SUM DW ?
DATA  ENDS;数据段结束
CODE SEGMENT;定义代码段
      ASSUME CS:CODE;说明代码段和数据段的位置
START:
;初始化
      MOV AX,1000H
      MOV DS,AX;定义数据段地址
      MOV AX,1
      MOV CX,100;初始化 AX 和 CX
      S:
      MOV [DI],AX
      INC AX
      INC DI
      LOOP S;将 1~100 循环存入 DS
;转存部分
      MOV DI,0;清空目的指针
      MOV AX,2000H
      MOV ES,AX;定义附加段地址
      MOV CX,100;设置循环次数
      CLD;清空方向标志，从低地址向高地址移动
      REP MOVSB;当 CX 不为 0 时继续配对
;比较部分
      MOV CX,100;设置循环次数
      REPZ CMPSB;当 CX 不为 0 且字符串匹配时继续配对
      JNZ MATCH;ZF=1 说明匹配成功
      JZ NOMATCH;ZF=0 说明匹配失败
;输出标志信息部分
MATCH:
      MOV AX,DATA
      MOV DS,AX;重新定义数据段
      LEA DX,MKEY;取出 MATCH 的偏移地址
      MOV AH,09H;调用 09H 指令输出 MATCH
      INT 21H
      JMP A;跳转到累加部分

NOMATCH:
      MOV AX,DATA
```

```
      MOV DS,AX;重新定义数据段
      LEA DX,NMKEY;取出 NOMATCH 的偏移地址
      MOV AH,09H;调用 09H 指令输出 NOMATCH
      INT 21H
      JMP A;跳转到累加部分
;累加部分
A:    MOV AX,1000H
      MOV DS,AX;重新定义数据段
      MOV CX,100
      XOR AX,AX;清空 AX
      XOR SI,SI;清空 SI
      XOR DI,DI;清空 DI
```

```
NEXT:
      XOR AX,AX;清空 AX
      ADD AL,[SI];利用 AL 存放一个字节中的数据
      MOV AH,0;修正高地址
      INC SI;指针位移
      ADD DX,AX;保证字长匹配,相加
      LOOP NEXT;循环相加
;输出部分
      MOV BX,DX;将结果转存入 BX
      MOV SI,0AH
      XOR CX,CX;清空计数器
      MOV AX,BX;将结果转存入 AX,准备除法
```

```
L:    XOR DX,DX
      DIV SI;取出 AX 的末位数
      PUSH DX;余数入栈保护数据
      INC CX;计数加一,为输出做准备
      CMP AX,0
      JNZ L;直到商为 0 时停止循环
```

```
PRINT:POP DX;数据出栈，FILO 顺序正确
      ADD DL,30h;转化成可以输出的 ASCII
      MOV AH,2
      INT 21h;调用输出字符的 02H 指令
      LOOP PRINT
```

```
      MOV AX,4C00H
      INT 21H;返回程序
```

```
CODE ENDS;代码段结束
```

```
END START;结束程序
```