

**课 程 实 验 报 告**

**课程名称： 汇编语言程序设计实验**

**实验时段： 2017年9月-2017年11月**

**指导教师： 许 向 阳**

**专业班级： 信息安全1601**

**学 号： U201614804**

**姓 名： 黄 永 恒**

实验报告成绩

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 实验 序号 | 一 | 二 | 三 | 四 | 五 | 六 | 总评 |
| 成绩 |  |  |  |  |  |  |  |

指导教师签字：

                     日 期：

**计算机科学与技术学院**

**课 程 实 验 报 告**

**课程名称： 汇编语言程序设计实验**

**实验名称： 实验一 编程基础**

**实验时间： 2017-10-10，18：30-21：50 实验地点： 南一楼804室5号实验台**

**指导教师： 许向阳**

**专业班级：信息安全 201601班**

**学 号： U201614804 姓 名： 黄永恒**

**同组学生： 无 报告日期： 2017年 10 月 11日**

**原创性声明**

  本人郑重声明：本报告的内容由本人独立完成，有关观点、方法、数据和文献等的引用已经在文中指出。除文中已经注明引用的内容外，本报告不包含任何其他个人或集体已经公开发表的作品或成果，不存在剽窃、抄袭行为。

特此声明！

学生签名：

日期：

成绩评定

|  |  |  |
| --- | --- | --- |
| 实验完成质量得分（70分）（实验步骤清晰详细深入，实验记录真实完整等） | 报告撰写质量得分（30分）（报告规范、完整、通顺、详实等） | 总成绩（100分） |
|  |  |  |

指导教师签字：

                     日期：

目录

[1 实验目的与要求 1](#_Toc500536025)

[2 实验内容 1](#_Toc500536026)

[3 实验过程 2](#_Toc500536027)

[3.1 任务1 2](#_Toc500536028)

[3.1.1 源程序 2](#_Toc500536029)

[3.1.2 实验步骤 2](#_Toc500536030)

[3.1.3 实验记录与分析 3](#_Toc500536031)

[3.2 任务2 5](#_Toc500536032)

[3.2.1 源程序 5](#_Toc500536033)

[3.2.2 实验步骤 6](#_Toc500536034)

[3.2.3 实验记录与分析 6](#_Toc500536035)

[3.3 任务3 9](#_Toc500536036)

[3.3.1 源程序 9](#_Toc500536037)

[3.3.2 实验步骤 10](#_Toc500536038)

[3.3.3 实验记录与分析 10](#_Toc500536039)

[3.4 选做题 13](#_Toc500536040)

[3.4.1 设计思想及存储单元分配 13](#_Toc500536041)

[3.4.2 流程图 13](#_Toc500536042)

[3.4.3 源程序 14](#_Toc500536043)

[3.4.4 实验步骤 15](#_Toc500536044)

[3.4.5 实验记录与分析 15](#_Toc500536045)

[4 总结与体会 16](#_Toc500536046)

[参考文献 17](#_Toc500536047)

# 实验目的与要求

本次实验的主要目的与要求有下面6点，所有的任务都会围绕这6点进行，希望大家事后检查自己是否达到这些目的与要求。

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。

# 实验内容

**任务1：内存单元的访问。**

以四种不同的内存寻址方式，将自己学号的后四位依次存储到 以 XUEHAO开头的存储区中。

要求：在报告中给出完整的程序；给出运行效果截图；（不需要画流程图）；在程序注释中，明确指出访问存储单元时，用的是什么寻址方式。

**任务2：《80X86汇编语言程序设计》教材中 P45的 2.3题。**

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。

**任务3. 《80X86汇编语言程序设计》教材中 P45的 2.4题的改写。**

要求：(1) 实现的功能不变，对数据段中变量访问时所用到的寻址方式中的寄存器改成32位寄存器。

(2) 内存单元中数据的访问采用变址寻址方式。

(3) 记录程序执行到退出之前数据段开始40个字节的内容，检查程序运行结果是否与设想的一致。

(4)在TD代码窗口中观察并记录机器指令代码在内存中的存放形式，并与TD中提供的反汇编语句及自己编写的源程序语句进行对照，也与任务2做对比。（相似语句记录一条即可，重点理解机器码与汇编语句的对应关系，尤其注意操作数寻址方式的形式）。

(5)观察连续存放的二进制串在反汇编成汇编语言语句时，从不同字节位置开始反汇编，结果怎样？理解 IP/EIP指明指令起始位置的重要性。

**任务4. 选做题**

设有

DATA SEGMENT USE16

MSG1 DB ‘Hello 123’

LEN = $ - MSG1 ; MSG1中字符的个数

MSG2 DB LEN DUP ( 0 )

DATA ENDS

写一个程序，将MSG1中的串逆序后存储到变量 MSG2中。

（报告中，要求有寄存器分配说明，源程序，运行后数据区的截图）

# 实验过程

## 任务1

### 源程序

;-----------------------------------------------------------

DATA SEGMENT USE16

DAITI DB '4','8','0','4'

XUEHAO DB 4 DUP(0)

DATA ENDS

;-----------------------------------------------------------

STACK SEGMENT STACK

DB 200 DUP(0)

STACK ENDS

;-----------------------------------------------------------

CODE SEGMENT USE16

ASSUME CS:CODE,DS:DATA,SS:STACK

START: MOV AX,DATA

MOV DS,AX

MOV SI,OFFSET DAITI

MOV DI,OFFSET XUEHAO+1

MOV AL,DAITI

MOV XUEHAO,AL ;直接寻址

INC SI

MOV AL,[SI]

MOV [DI],AL ;间接寻址

INC SI

MOV AL,DAITI[SI]

MOV XUEHAO[SI],AL ;变址寻址

INC SI

LEA BX,DAITI

LEA BP,XUEHAO

MOV AL,[BX][SI]

MOV DS:[BP][SI],AL ;基址加变址

MOV AH,4CH

INT 21H

CODE ENDS

END START

### 实验步骤

1. 准备上机实验环境。

2. 使用记事本编辑录入源程序，存盘文件名为demo.asm。使用MASM 6.0汇编源文件。即masm demo观察提示信息，若出错，则用编辑程序修改错误，存盘后重新汇编，直至不再报错为止。

3. 使用连接程序LINK.EXE将汇编生成的CUBE.OBJ文件连接成可执行文件。即link demo；若连接时报错，则依照错误信息修改源程序。之后重新汇编和连接，直至不再报错并生成demo.exe文件。

4. 使用调试器TD.EXE对生成的文件进行调试。

### 实验记录与分析

1. 实验环境条件：i7-5500U 3.0GHz，8G内存；WINDOWS 10下DOSBox0.73；EDIT.EXE 2.0；MASM.EXE 6.0； LINK.EXE 5.2; TD.EXE 5.0。
2. 编译连接正常，如图3.1.1所示

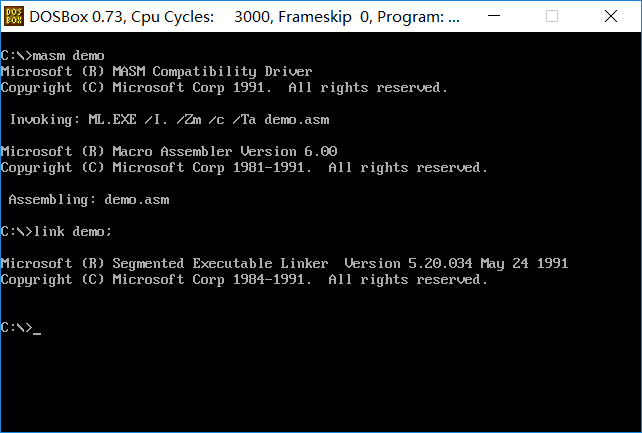


图3.1.1 正确编译连接

1. 输入td demo，进入调试界面，调试两步后，观察DS段,如图3.1.2所示

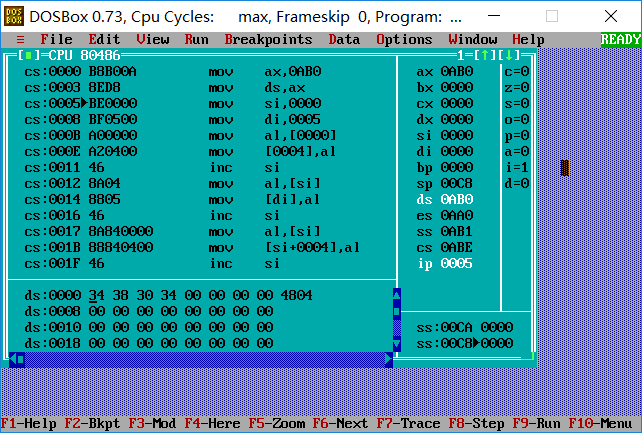


图3.1.2 调试两步后，DS段存储情况

1. 继续调试，将第一个数字以寄存器寻址的方式存储，如图3.1.3所示

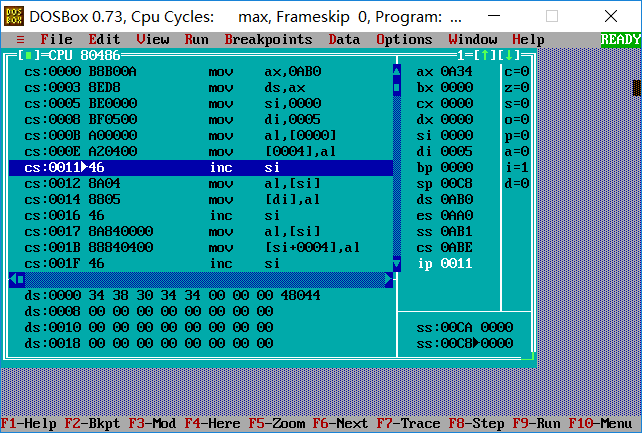


图3.1.3 以寄存器寻址方式存储

1. 继续调试，将第二个数字以寄存器间接寻址的方式存储，如图3.1.4所示

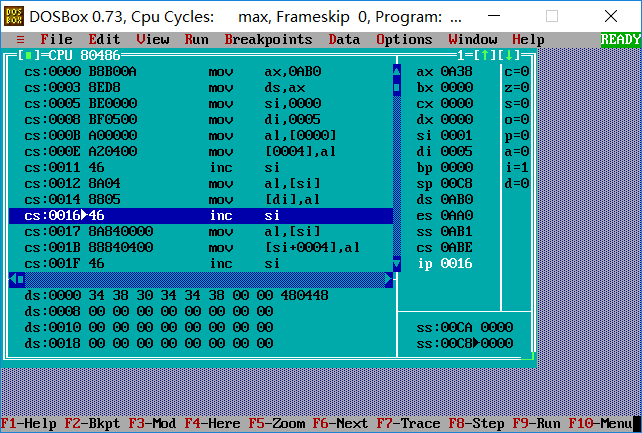


图3.1.4 以寄存器间接寻址方式存储

1. 继续调试，将第三个数字以变址寻址的方式存储，如图3.1.5所示

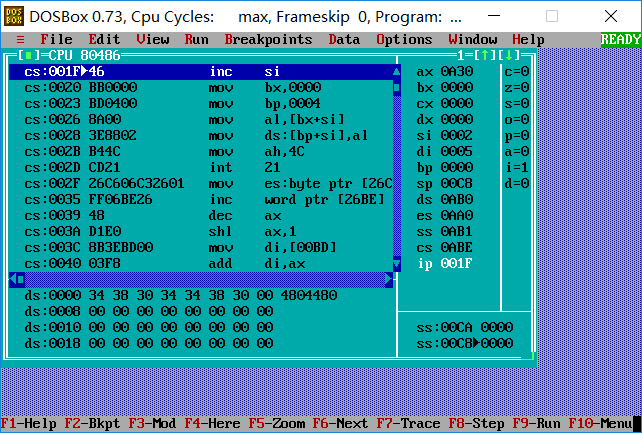


图3.1.5 以变址寻址方式存储

1. 继续调试，将第四个数字以基址加变址的方式进行存储，如图3.1.6所示

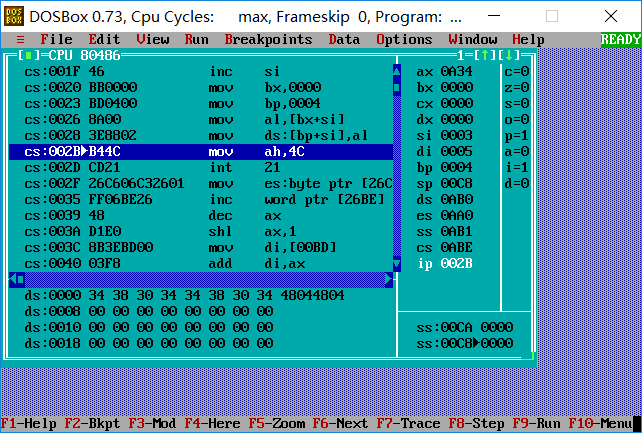


图3.1.6 以基址加变址方式存储

## 任务2

### 源程序

;---------------------------------------------------------

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

;----------------------------------------------------------

STACK SEGMENT STACK

DB 200 DUP(0)

STACK 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

LOOPA: 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 LOOPA

MOV AH, 4CH

INT 21H

CODE ENDS

END START

### 实验步骤

* 1. 准备上机环境，编辑、汇编、连接文件DEMO。
  2. 使用TD.EXE观察CUBE的执行情况。即 TD DEMO.EXE回车

（1）记录初始(BX), (BP),(SI),(DI)各是多少。

（2）记录执行到“MOV CX，10”之前时，(BX), (BP),(SI),(DI)各是多少。

（3）记录执行到“INT 21H”之前时，(BX), (BP),(SI),(DI)各是多少。

（4）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。

### 实验记录与分析

1. 实验环境条件：i7-5500U 3.0GHz，8G内存；WINDOWS 10下DOSBox0.73；EDIT.EXE 2.0；MASM.EXE 6.0； LINK.EXE 5.2; TD.EXE 5.0。

2. 编译连接正常，如图3.2.1所示

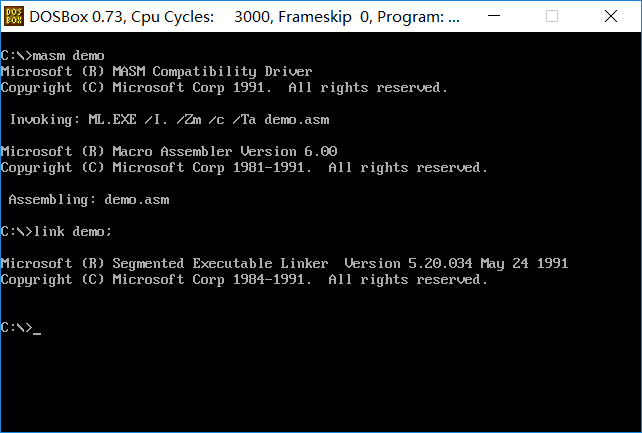


图3.2.1 正常编译连接

* 1. 输入td demo，进入调试界面，(BX), (BP),(SI),(DI)初始值如图3.2.2所示

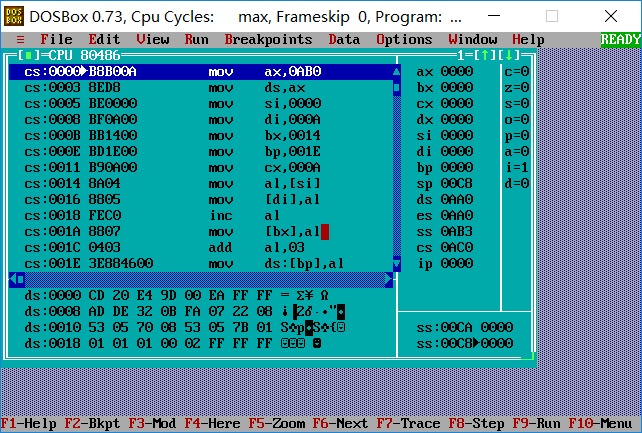


图3.2.2 (BX), (BP),(SI),(DI)初始值

* 1. 执行到“MOV CX，10”之前时，(BX), (BP),(SI),(DI)值如图3.2.3所示

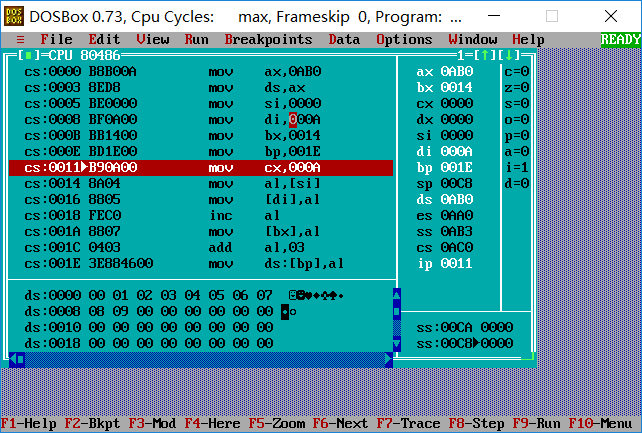


图3.2.3 执行到“MOV CX，10”之前时，(BX), (BP),(SI),(DI)值

* 1. 执行到“INT 21H”之前时，(BX), (BP),(SI),(DI)值如图3.2.4所示

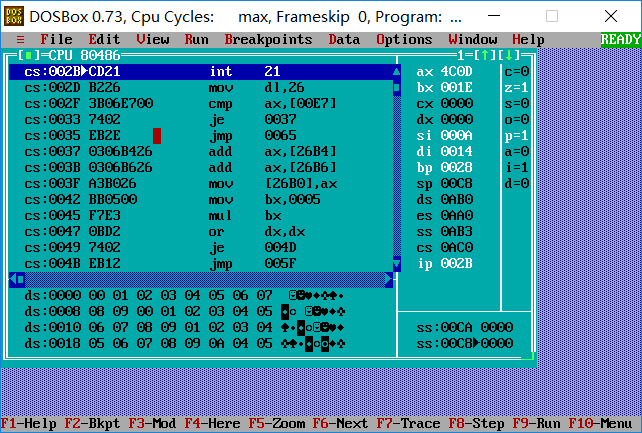


图3.2.4 执行到“INT 21H”之前时，(BX), (BP),(SI),(DI)值及

执行到退出之前数据段开始40个字节的内容

* 1. 程序执行到退出之前数据段开始40个字节的内容，预测为

BUF1:0,1,2,3,4,5,6,7,8,9 BUF2:0,1,2,3,4,5,6,7,8,9

BUF3:1,2,3,4,5,6,7,8,9,A BUF4:4,5,6,7,8,9,A,B,C,D

实际结果如图3.2.4及图3.2.5所示

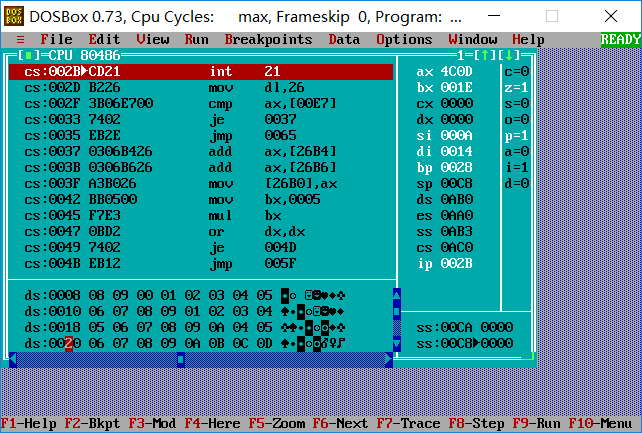


图3.2.5 执行到退出之前数据段开始40个字节的内容

由此可知，程序运行结果与设想的一致。

## 任务3

### 源程序

;------------------------------------------------------------

.386

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

;------------------------------------------------------------

STACK SEGMENT USE32 STACK

DB 200 DUP(0)

STACK ENDS

;------------------------------------------------------------

CODE SEGMENT USE16

ASSUME CS:CODE,DS:DATA,SS:STACK

START: MOV AX,DATA

MOV DS,AX

MOV ESI,0

MOV ECX,10

LOOPA: MOV AL,BUF1[ESI]

MOV BUF2[ESI],AL

INC AL

MOV BUF3[ESI],AL

ADD AL,3

MOV BUF4[ESI],AL

INC ESI

DEC ECX

JNZ LOOPA

MOV AH,4CH

INT 21H

CODE ENDS

END START

### 实验步骤

1. 准备上机环境，编辑、汇编、连接文件DEMO。
2. 使用TD.EXE观察DEMO的执行情况。
3. 设置断点到程序退出前
4. 观察数据段开始40个字节的内容
5. 在程序运行的过程中，注意指令在内存中的存放方式以及源程序代码与反汇编代码的区别。
6. 观察连续存放的二进制串在反汇编成汇编语言语句时，从不同字节位置开始反汇编，结果怎样？

### 实验记录与分析

1. 实验环境条件：i7-5500U 3.0GHz，8G内存；WINDOWS 10下DOSBox0.73；EDIT.EXE 2.0；MASM.EXE 6.0； LINK.EXE 5.2; TD.EXE 5.0。
2. 编译连接正常，如图3.3.1所示

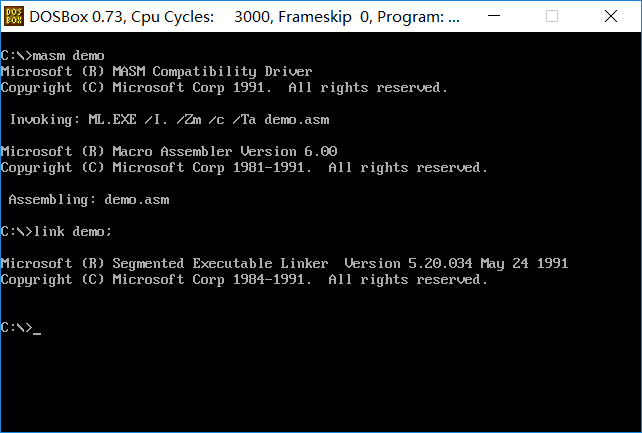


图3.3.1 正常编译连接

1. 输入td demo，进入调试器，设置断点，运行到程序退出前，观察到数据段开始40个字节的内容与预想一样，如图3.3.2所示

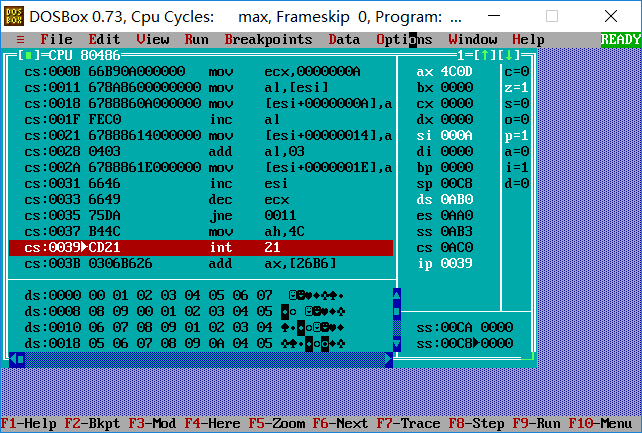


图3.3.2 运行到程序退出之前

1. 在程序运行过程中，由于任务3采用了32位寄存器，且采用变址寻址方式，因此变址寻址中的立即数变成8位。并且反汇编语言与源代码相比，将实际地址进行运算以16进制显示出来。并且发现反汇编语言、汇编语言与机器代码是一一对应关系。如图3.3.3，图3.3.4，图3.3.5所示

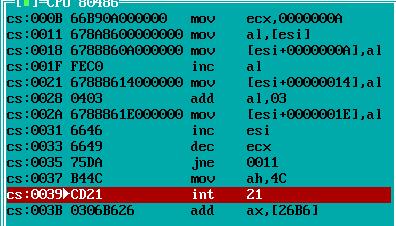


图3.3.3 32位寄存器

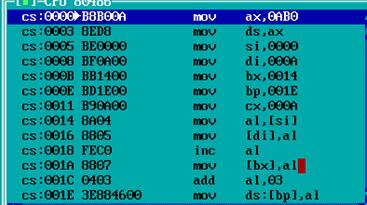


图3.3.4 16位寄存器

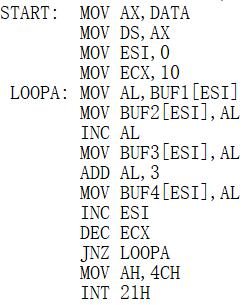


图3.3.5 汇编源代码

1. 重新运行程序，观察CS段存储内容，发现机器语言与CS段存储字节相对应，如图3.3.6所示

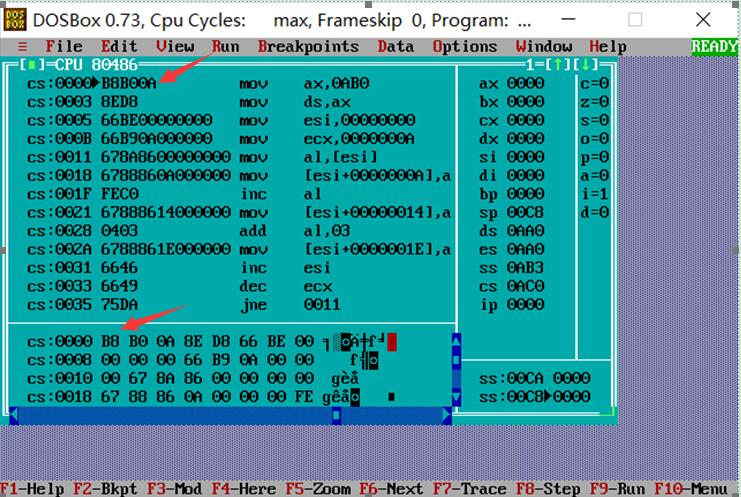


图3.3.6 CS段机器语言与存储字节

1. 修改IP值为0005，运行单步运行程序，如图3.3.7所示，发现程序跳过0004之前的语句，直接执行IP所指向的语句，由此可以看出IP/EIP指明的指令起始位置十分重要，稍有不对或者被误更改，程序的执行顺序都会出乱子。

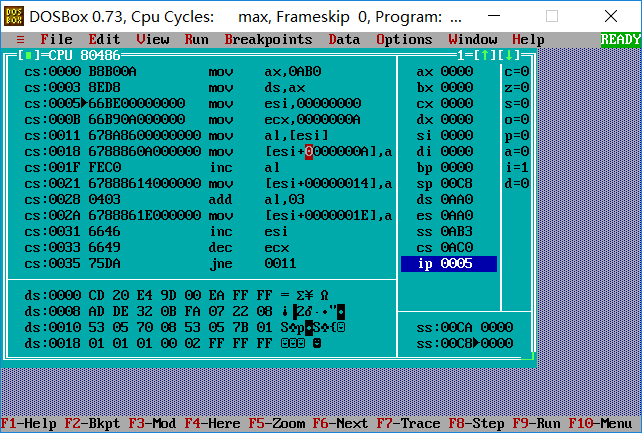


图3.3.7 修改IP值后程序状态

## 选做题

### 设计思想及存储单元分配

将MSG1中的串逆序后存储到变量 MSG2中，可以考虑用变址寻址的方法，用一个指针指向原串末尾，然后存储到目的串中，同时指针向前移动，直到串中元素存储完毕。

1.存储单元分配

MSG1：存储字符串‘Hello 123’

MSG2：初始为空，存储目的串

LEN：字节变量，其值为MSG1中字符串的长度

2.寄存器分配

CX：存放LEN

SI: 作为MSG1的偏移地址

DI：作为MSG2的偏移地址

AX：将DATA段数据传送给DS

AL：作为临时寄存器

### 流程图

图3.4.1是选做题的程序流程图。

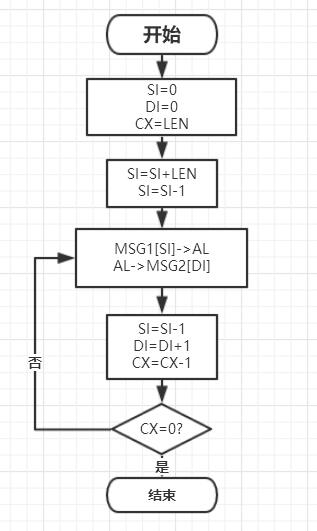


图3.4.1 逆序存储程序流程图

### 源程序

;--------------------------------------------------------------

DATA SEGMENT USE16

MSG1 DB 'Hello 123'

LEN = $ - MSG1 ; MSG1中字符的个数

MSG2 DB LEN DUP ( 0 )

DATA ENDS

;-------------------------------------------------------------

STACK SEGMENT STACK

DB 200 DUP(0)

STACK ENDS

;------------------------------------------------------------

CODE SEGMENT USE16

ASSUME CS:CODE,DS:DATA,SS:STACK

START: MOV AX,DATA

MOV DS,AX

MOV SI,0

MOV DI,0

MOV CX,LEN

ADD SI,LEN

DEC SI ;此时SI指向MSG1最后一个字符

LOOPA: MOV AL,MSG1[SI]

MOV MSG2[DI],AL

DEC SI

INC DI

DEC CX ;若CX等于0，则表明已传递完毕

JNZ LOOPA

MOV AH,4CH

INT 21H

CODE ENDS

END START

### 实验步骤

1. 准备上机环境，编辑、汇编、连接文件DEMO。
2. 使用TD.EXE观察DEMO的执行情况。

（1）打断点使程序运行至退出之前。

（2）观察DS：0开始的数据段，观察此时的值是否符合预期要求。

### 实验记录与分析

1. 实验环境条件：i7-5500U 3.0GHz，8G内存；WINDOWS 10下DOSBox0.73；EDIT.EXE 2.0；MASM.EXE 6.0； LINK.EXE 5.2; TD.EXE 5.0。
2. 编译连接正常，如图3.4.2所示

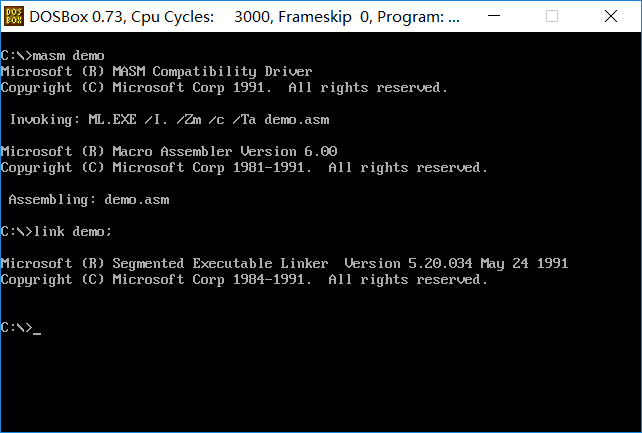


图3.4.2 正常编译连接

1. 输入td demo，进入调试器，设置断点，使得程序运行到退出前，观察DS：0开始的数据段，如图3.4.3所示

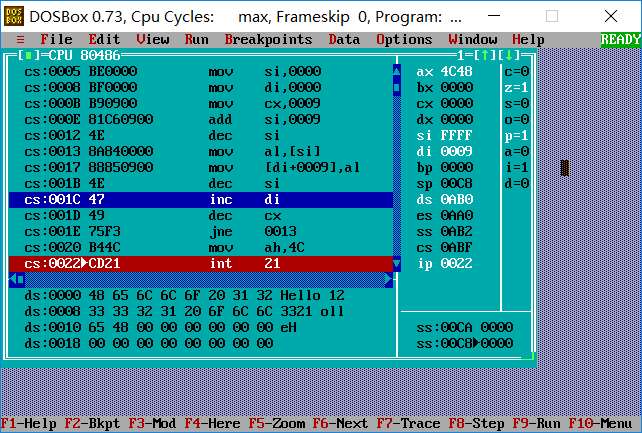


图3.4.3 程序运行后数据段内容

由上图可知，数据段已经存储了倒序的字符串，与预期一致。

# 总结与体会

通过任务1的实验，让我知道了TD不仅可以调入现成的执行程序进行调试，而且能随时输入和测试单一一条指令是否正确，执行效果如何，这将方便未来的学习过程。同时，应用四种寻址方式进行存储也让我对六种寻址方式更加熟悉。

通过任务2的实验，我熟悉了如何在TD中设置断点，观察程序运行到指定位置的状态，这对于我们检查程序运行是否正常有很大帮助。

通过任务3的实验，我了解了32位与16位的寄存器的区别，并且初步了解汇编语言、机器语言以及反汇编语言之间的关系，更重要的是通过改变IP/EIP的值，发现程序异常，更加了解IP/EIP的作用以及重要性，平时不敢动它们的值。

通过选做题的实验，我开始初步设计一个汇编程序，这让我对最近所学过的汇编知识有了更加深刻的理解，通过应用这些知识，尝到了汇编的乐趣。

本次上机不仅提高了编程水平，熟悉了工具的使用，而且加深了对一些知识的理解。主要的经验教训如下：

首先，更加感受到实验前准备的意义。例如：上机前准备越充分（如先编好源程序，制定好准备做的一些步骤），上机的时候目的越明确，可以解决较多的问题。

其次，录入程序时要注意一些细节，比如中文分号、字母O等问题，虽然汇编程序指出其所在行有错，但很难发现具体是哪个符号错了，耽误了不少时间。TD在程序细节的观察、动态修改方面有很大的作用，要主动用TD的调试功能帮助自己发现、理解与解决问题。

最后，由于操作不够熟练，时间比较紧张等原因，还有些问题需要以后进一步解决。

总之，这次实验只是开头，我相信自己会越来越熟练，越来越顺利！

# 参考文献

[1] 王元珍 曹忠升 韩宗芬.《80x86汇编语言程序设计》“第二章 寻址方式”.华中科技大学出版社