

汇编语言程序设计实验报告

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：实验一 编程基础

实验时间：2019-3-20, 14:00-17:30 实验地点：南一楼 804 室 30 号实验台

指导教师：曹忠升

专业班级：计算机科学与技术 ACM1701 班

学号：U201714780 姓名：刘晨彦

同组学生：无 报告日期：2019 年 3 月 20 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：2019.03.20

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：

汇编语言程序设计实验报告

汇编语言程序设计实验报告

目录

1 实验目的与要求.....	1
2 实验内容.....	1
3 实验过程.....	3
3.1 任务 1	3
3.1.1 实验步骤.....	3
3.1.2 实验记录与分析	3
3.2 任务 2	5
3.2.1 实验步骤.....	5
3.2.2 实验记录与分析	5
3.3 任务 3	6
3.3.1 实验步骤.....	6
3.3.2 源代码	7
3.3.3 实验记录与分析	7
3.4 任务 4	9
3.4.1 设计思想及存储单元分配.....	9
3.4.2 流程图	11
3.4.3 源程序	15
3.4.4 实验步骤.....	19
3.4.5 实验记录与分析	19
4 总结与体会.....	25
参考文献.....	27

汇编语言程序设计实验报告

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. 《80X86 汇编语言程序设计》教材中 P31 的 1.14 题。

要求：

- (1) 直接在 TD 中输入指令，完成两个数的求和、求差的功能。求和/差后的结果放在(AH)中。
- (2) 请事先指出执行指令后(AH)、标志位 SF、OF、CF、ZF 的内容。
- (3) 记录上机执行后的结果，与（2）中对应的内容比较。

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，但对数据段中变量访问时所用到的变址寄存器采用 32 位寄存器。
- (2) 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。
- (3) 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编码形式，比如寄存器间接寻址、变址寻址、32 位寄存器与 16 位寄存器编码的不同、段前缀在代码里是如何表示的等）。
- (4) 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. 设计实现一个网店商品信息管理的程序。

1、实验背景

有一个老板在网上开了 1 个网店 SHOP，网店里 n 种商品销售。每种商品的信息包括：商品

汇编语言程序设计实验报告

名称（10 个字节，数据段中定义时，名称不足部分补 0），折扣（字节类型，取值 0~10；0 表示免费赠送，10 表示不打折，1~9 为折扣率；实际销售价格=销售价*折扣/10），进货价（字类型），销售价（字类型），进货总数（字类型），已售数量（字类型），推荐度【=（进货价/实际销售价格+已售数量/（2*进货数量））*128，字类型】。老板管理网店信息时需要输入自己的名字（10 个字节，数据段中定义时，不足部分补 0）和密码（6 个字节，数据段中定义时，不足部分补 0），登录后可查看商品的全部信息；顾客（无需登录）可以查看网店中每个商品除了进货价以外的信息。

例如：

BNAME DB 'ZHANG SAN', 0 ; 老板姓名

BPASS DB 'test', 0, 0 ; 密码

N EQU 30

SNAME DB 'SHOP', 0 ; 网店名称，用 0 结束

GA1 DB 'PEN', 7 DUP(0), 10 ; 商品名称及折扣

DW 35, 56, 70, 25, ? ; 推荐度还未计算

GA2 DB 'BOOK', 6 DUP(0), 9 ; 商品名称及折扣

DW 12, 30, 25, 5, ? ; 推荐度还未计算

GAN DB N-2 DUP('Temp-Value', 8, 15, 0, 20, 0, 30, 0, 2, 0, ?, ?) ; 除了 2 个已经具体定义了的商品信息以外，其他商品信息暂时假定为一样的。

2、功能一：提示并输入登录用户的姓名与密码

（1）使用 9 号 DOS 系统功能调用，先后分别提示用户输入姓名和密码（第一行显示将要访问的网店名称）。

（2）使用 10 号 DOS 系统功能调用，分别输入姓名和密码。输入的姓名字符串放在以 in_name 为首址的存储区中，密码放在以 in_pwd 为首址的存储区中，进入功能二的处理。

（3）若输入姓名时只是输入了回车，则将 0 送到 AUTH 字节变量中，跳过功能二，进入功能三；若在输入姓名时仅仅输入字符 q，则程序退出。

3、功能二：登录信息认证

（1）使用循环程序结构，比较姓名是否正确。若不正确，则跳到（3）。

（2）若正确，再比较密码是否相同，若不同，跳到（3）。

（3）若名字或密码不对，则提示登录失败，并回到“功能一（1）”的位置，提示并重新输入姓名与密码。

（4）若名字和密码均正确，则将 1 送到 AUTH 变量中，进到功能三。

4、功能三：计算指定商品的推荐度。

（1）提示用户输入要查询的商品名称。若未能找到该商品，重新提示输入商品名称。若只输入回车，则回到功能一（1）。

（2）判断登录状态，若是已经登录的状态，转到（3）。否则，转到（4）。

（3）在下一行显示该商品的名称，然后回到功能一（1）。

（4）计算该商品的推荐度，然后进入功能四。

汇编语言程序设计实验报告

3 实验过程

3.1 任务 1

3.1.1 实验步骤

1. 准备上机实验环境。

2. 在 TD 的代码窗口中的当前光标下输入第一个运算式对应的两个 8 位数值对应的指令语句 MOV AH,01001101B; MOV AL,-01110010B; ADD AH,AL; 观察代码区显示的内容与自己输入字符之间的关系; 然后确定 CS:IP 指向的是自己输入的第一条指令的位置, 单步执行三次, 观察寄存器内容的变化, 记录标志寄存器的结果。

预计 ADD 执行之后 (AH)=0DBH SF=1、OF=0、CF=0、ZF=0

重复上述过程, 将剩下几个表达式计算完毕, 比较结果。

3. 输入 MOV AH,+0110011B; MOV AL,+1011010B; ADD AH,AL; 观察标志位特点。

预计 ADD 执行后: (AH)=8DH;SF=1,OF=1,CF=0,ZF=0

输入 MOV AH,-0101001B; MOV AL,-1011101B; ADD AH,AL; 观察标志位特点。

预计 ADD 执行后: (AH)=7AH; SF=0,OF=1,CF=1,ZF=0

输入 MOV AH,+1100101B; MOV AL,-1011101B; ADD AH,AL; 观察标志位特点。

预计 ADD 执行后: (AH)=08H; SF=0,OF=0,CF=1,ZF=0

实验预计如表 1.1 所示:

表 1.1 任务一预测实验结果

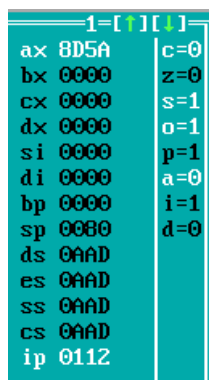
题号	X1	X2	AH	SF	OF	CF	ZF
(1)	+0110011B	+1011010B	8DH	1	1	0	0
(2)	-0101001B	-1011101B	7AH	0	1	1	0
(3)	+1100101B	-1011101B	08H	0	0	1	0

3.1.2 实验记录与分析

1. 实验环境条件: i7-7700HQ 2.80GHz, 8G 内存; WINDOWS 10 下 DOSBox0.72; TD.EXE 5.0。

2. 输入第一题指令 MOV AH,+0110011B; MOV AL,+1011010B; ADD AH,AL 后, 进行单步调试, 执行三条指令后的结果如图 1.1 所示。

汇编语言程序设计实验报告

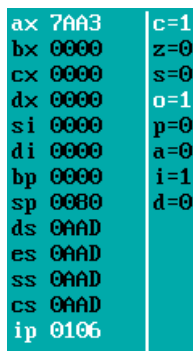


1=[↑][↓]	
ax	8D5A
bx	0000
cx	0000
dx	0000
si	0000
di	0000
bp	0000
sp	0080
ds	0AAD
es	0AAD
ss	0AAD
cs	0AAD
ip	0112
c	0
z	0
s	1
o	1
p	1
a	0
i	1
d	0

图 1.1 任务一第（1）小题测试截图

可以看出，计算结果 AH = 8DH,与标志位的状态(SF=1,OF=1,CF=0,ZF=0)与事前预期的是一致的。

3. 输入第二题指令 MOV AH,-0101001B; MOV AL,-1011101B; ADD AH,AL 后，进行单步调试，执行三条指令后的结果如图 1.2 所示。

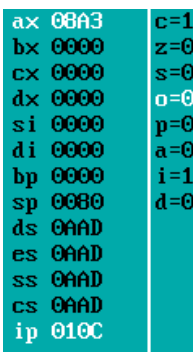


ax	7AA3
bx	0000
cx	0000
dx	0000
si	0000
di	0000
bp	0000
sp	0080
ds	0AAD
es	0AAD
ss	0AAD
cs	0AAD
ip	0106
c	1
z	0
s	0
o	1
p	0
a	0
i	1
d	0

图 1.2 任务一第（2）小题测试截图

可以看出，计算结果 AH = 7AH,与标志位的状态(SF=0,OF=1,CF=1,ZF=0)与事前预期的是一致的。

4. 输入第三题指令 MOV AH,+1100101B; MOV AL,-1011101B; ADD AH,AL 后，进行单步调试，执行三条指令后的结果如图 1.3 所示。



ax	08A3
bx	0000
cx	0000
dx	0000
si	0000
di	0000
bp	0000
sp	0080
ds	0AAD
es	0AAD
ss	0AAD
cs	0AAD
ip	010C
c	1
z	0
s	0
o	0
p	0
a	0
i	1
d	0

图 1.3 任务一第（3）小题测试截图

可以看出，计算结果 AH = 08H,与标志位的状态(SF=0,OF=0,CF=1,ZF=0)与事前预期的是一致的。

汇编语言程序设计实验报告

3.2 任务 2

3.2.1 实验步骤

1. 准备上机实验环境。

2. 建立汇编源程序，将题目代码写入文件；汇编源文件，生成目标文件；连接目标文件，建立可执行文件。

3.在 td 窗口点击 FILE 菜单中的 Open 选项,打开文件。找到“MOV CX, 10”和“INT 21H”，点击选中并按 F2 添加断点。

4.按 F9 运行程序至断点处暂停，记录 BX,BP,SI,DI 的值。

预计执行到“MOV CX, 10”之前，BX=0014H,BP=001EH,SI=0000H,DI=000AH。

预计执行到“INT 21H”之前，BX=001EH,BP=0028H,SI=000AH,DI=0014H。

相关预计如表 1.2 所示。

表 1.2 任务二预测实验结果 1

断点	BX	BP	SI	DI
MOV CX, 10	0014	001E	0000	000A
INT 21H	001E	0028	000A	0014

5.完成断点观察后重新进入，进行单步调试。在代码段右键选择 goto，输入 DS:0000,转调至 DS 区。注意查看数据段的变化，预计结束前 40 个字节的数据段数值如表 1.3 所示。

表 1.3 任务二预测实验结果 2

有效地址	前四十个字节的内容
DS0000	00 01 02 03 04 05 06 07
DS0008	08 09 00 01 02 03 04 05
DS0010	06 07 08 09 01 02 03 04
DS0018	05 06 07 08 09 0A 04 05
DS0020	06 07 08 09 0A 0B 0C 0D

3.2.2 实验记录与分析

1. 实验环境条件: i7-7700HQ 2.80GHz, 8G 内存; WINDOWS 10 下 DOSBox0.72; TD.EXE 5.0。

2. 在 DOSBOX 中打开 TD. EXE，通过 FILE-OPEN 打开 23. exe，在“MOV CX, 10”和“INT 21H”处添加断点，进行单步调试，观察在“MOV CX, 10”之前 BX,BP,SI,DI 的值，如图 1.4 所示。结果与预期相符合。

汇编语言程序设计实验报告

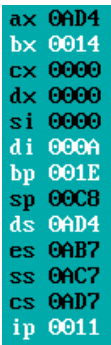


图 1.4 “MOV CX, 10”断点处的实验测试截图

3. 进行单步调试，观察在“INT 21H”之前 BX, BP, SI, DI 的值，如图 1.5 所示。结果与预期相符合。



图 1.5 “INT 21H”断点处的实验测试截图

4. 在代码段右键选择 goto，输入 DS:0000, 转调至 DS 区。获得结果如图 1.6 所示，与预期一致。

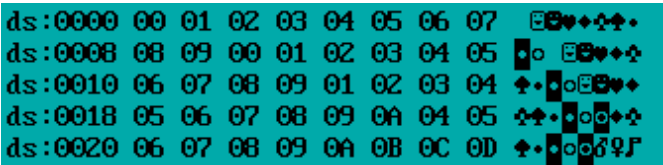


图 1.6 结束前 40 个字节的数据段数值

3.3 任务 3

3.3.1 实验步骤

1. 准备上机实验环境。
2. 修改任务 2 中的代码，使其满足任务 3 的要求
3. 对修改后的代码建立汇编源程序；汇编源文件，生成目标文件；连接目标文件，建立可执行文件。
4. 在 DOSBOX 中打开 TD.exe，点击 FILE-OPEN，选择 exe 文件，进行单步调试，注意观察数据段和标志符的变化，预计程序执行到退出之前数据段开始 40 个字节的内容，预计内容如表 1.4 所示。

表 1.4 任务三预测实验结果

有效地址	前四十个字节的内容
------	-----------

汇编语言程序设计实验报告

DS0000	00 01 02 03 04 05 06 07
DS0008	08 09 00 01 02 03 04 05
DS0010	06 07 08 09 01 02 03 04
DS0018	05 06 07 08 09 0A 04 05
DS0020	06 07 08 09 0A 0B 0C 0D

5. 在 TD 代码窗口中，观察并记录指令代码在内存中的存储形式，将并与 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。

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的结果。

3.3.2 源代码

```
. 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 ESI, 0
        MOV EDI, 0
        MOV EBX, 0
        MOV EBP, 0
        MOV CX, 10
LOPA:  MOV AL, [ESI]+BUF1
        MOV [EDI]+BUF2, AL
        INC AL
        MOV [EBX]+BUF3, AL
        ADD AL, 3
        MOV DS:[EBP]+BUF4, AL
        INC ESI
        INC EDI
        INC EBP
        INC EBX
        DEC CX
        JNZ LOPA
        MOV AH, 4CH
        INT 21H
CODE ENDS
END START
```

3.3.3 实验记录与分析

1. 实验环境条件: i7-7700HQ 2.80GHz, 8G 内存; WINDOWS 10 下 DOSBox0.72; TD.EXE 5.0。
2. 打开 DOSBOX, 使用 MASM 对源程序 23m.asm 进行编译, 再使用 LINK 对 23m.obj 进行连接, 生成 23m.exe 文件。

汇编语言程序设计实验报告

3. 打开 TD.EXE，点击 FILE-OPEN，选择 23m.exe。
4. 对文件结束位置设置断点，快速查看程序退出前数据段前 40 个字节的內容，如图 1-7 所示，结果与预测情况一致。

ds:0000	00 01 02 03 04 05 06 07	08 09 0A 0B 0C 0D 0E 0F
ds:0008	00 01 02 03 04 05 06 07	08 09 0A 0B 0C 0D 0E 0F
ds:0010	00 01 02 03 04 05 06 07	08 09 0A 0B 0C 0D 0E 0F
ds:0018	00 01 02 03 04 05 06 07	08 09 0A 0B 0C 0D 0E 0F
ds:0020	00 01 02 03 04 05 06 07	08 09 0A 0B 0C 0D 0E 0F

图 1-7 任务三结束前 40 个字节的数据段数值

5. 将反汇编代码与自己所写代码进行对比，也与任务二做对比。如图 1.8 与图 1.9，图 1.10 所示。

[CPU 80486]			
cs:0003	8ED8	mov	ds,ax
cs:0005	66BE00000000	mov	esi,00000000
cs:000B	66BF00000000	mov	edi,00000000
cs:0011	66BB00000000	mov	ebx,00000000
cs:0017	66BD00000000	mov	ebp,00000000
cs:001D	B90A00	mov	cx,000A
cs:0020	678A8600000000	mov	al,[esi]
cs:0027	6788870A000000	mov	[edi+0000000A],al
cs:002E	FEC0	inc	al
cs:0030	67888314000000	mov	[ebx+00000014],al
cs:0037	0403	add	al,03
cs:0039	673E88851E0000	mov	ds:[ebp+0000001E],al
cs:0041	6646	inc	esi
cs:0043	6647	inc	edi
cs:0045	6645	inc	ebp

图 1.8 任务三部分反汇编代码

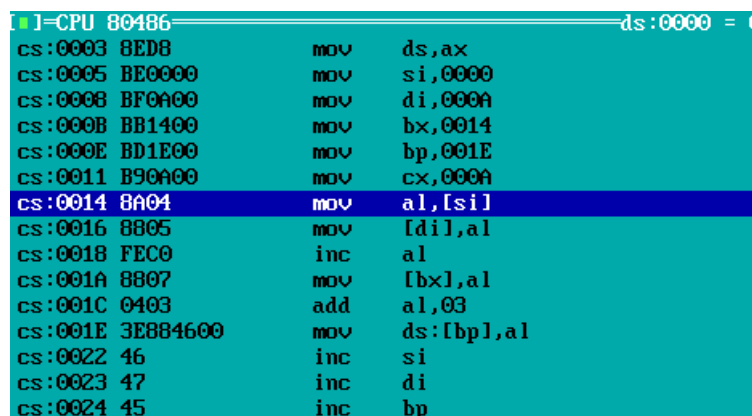
```

START: MOV AX, DATA
      MOV DS, AX
      MOV ESI, 0
      MOV EDI, 0
      MOV EBX, 0
      MOV EBP, 0
      MOV CX, 10
LOPA: MOV AL, [ESI]+BUF1
      MOV [EDI]+BUF2, AL
      INC AL
      MOV [EBX]+BUF3, AL
      ADD AL, 3
      MOV DS:[EBP]+BUF4, AL
      INC ESI
      INC EDI
      INC EBP

```

图 1.9 部分源代码

汇编语言程序设计实验报告



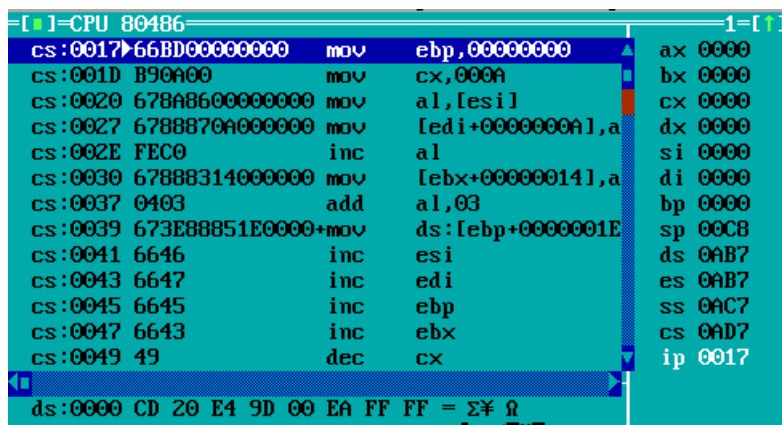
[]=CPU 80486		ds:0000 = 0	
cs:0003	8ED8	mov	ds,ax
cs:0005	BE0000	mov	si,0000
cs:0008	BF0A00	mov	di,000A
cs:000B	BB1400	mov	bx,0014
cs:000E	BD1E00	mov	bp,001E
cs:0011	B90A00	mov	cx,000A
cs:0014	8A04	mov	al,[si]
cs:0016	8805	mov	[di],al
cs:0018	FEC0	inc	al
cs:001A	8807	mov	[bx],al
cs:001C	0403	add	al,03
cs:001E	3E884600	mov	ds:[bp],al
cs:0022	46	inc	si
cs:0023	47	inc	di
cs:0024	45	inc	bp

图 1.10 任务二部分反汇编代码

对比发现如下：

- (1) 指令名称在反汇编代码和源代码中是相同的，如”mov”,”inc”等。
- (2) 在反汇编代码中，变址寻址部分直接将位移量以偏移地址形式显示。如源代码中的：”MOV [EDI]+BUF2, AL”在反汇编代码中为：”mov [edi+0000000A],al”。
- (3) 在反汇编代码中，START 和 LOPA 都表现为了偏移地址，如源代码中的：”JNZ LOPA”在反汇编代码中为：”jne 0020”。
- (4) 任务二和任务三的寻址方式不同，使得反汇编代码中的表示也不同，人物二中使用了寄存器间接寻址，在反汇编代码中保持不变，如：”MOV AL,[SI]”在反汇编代码中仍然为：”mov al,[si]”。
- (5) 在反汇编代码中，常数均转化成十六进制的形式，如源代码中的:”MOV CX,10”在反汇编代码中显示为:”mov cx 000A”

6. 在代码段执行 GOTO 语句，输入 cs:0017，并设置为当前操作语句，此时寄存器 ip 地址变为 0017，如图 1.11 所示。



[]=CPU 80486		-1=[]	
cs:0017	66BD00000000	mov	ebp,00000000
cs:001D	B90A00	mov	cx,000A
cs:0020	678A8600000000	mov	al,[esi]
cs:0027	6788870A000000	mov	[edi+0000000A],a
cs:002E	FEC0	inc	al
cs:0030	67888314000000	mov	[ebx+00000014],a
cs:0037	0403	add	al,03
cs:0039	673E88851E0000	mov	ds:[ebp+0000001E
cs:0041	6646	inc	esi
cs:0043	6647	inc	edi
cs:0045	6645	inc	ebp
cs:0047	6643	inc	ebx
cs:0049	49	dec	cx

ds:0000 CD 20 E4 9D 00 EA FF FF = Σ¥ Ω
ip 0017

图 1.11 执行 GOTO 之后的状态截图

3.4 任务 4

3.4.1 设计思想及存储单元分配

a)设计思想：

汇编语言程序设计实验报告

程序功能一主要是用了 9 号和 10 功能调用，完成输出和输入，其中包含了简单的判断语句：当只输入了回车时，程序进入功能三，当只输入了 q 时，程序退出。

程序功能二使用了两个单层循环结构进行字符串的匹配，若不匹配则输出提示返回功能一，若匹配则进入功能三。

功能三除了使用了 9 号和 10 号调用以外，最主要使用了双层循环结构，比较输入的商品字符串和存储在内存中的字符串是否相同。若不同，则重新要求输入商品名。若相同，根据用户登录与否创建分支结构：若登录，则重新输出商品名并回到功能一，未登录则计算商品推荐度并进入功能四。

功能四主要使用了多个分支结构，根据商品的推荐度给出推荐等级。

b)程序的主要算法：

串的比较：在比较用户名和密码时，首先判断在输入缓冲区的指定位置是否为 0DH，确保输入字符数和正确的字符串长度相等，否则重新输入。然后进入循环，从字符串首开始比较，不同则退出循环要求重新输入，直到相同且计数器为 0 退出。在比较商品名称时，则是从字符串首开始比较，比较至输入商品字符串结束且均相同时，判断商品字符串的下一位是否为 0DH，若不是，则说明输入的字符串是当前比较字符串的子串，依然不相同。

推荐度的计算：为了避免进价除以售价商为 0 的情况，首先将被除数乘以 128 以获得正确结果。

显示：为了正确显示输出和输入，在每一次输入和输出之后都确保程序会输出回车换行，否则容易出现输入内容被下一次输出所覆盖的情况。

c)存储单元分配：

BNAME：用户名的首地址。

BPASS：密码的首地址。

IN_NAME:存放输入用户名的首地址。

IN_PWD：存放输入密码的首地址。

IN_ITEM:：存放输入商品名称的首地址。

AUTH：字节变量，存放登录状态。

SNAME：字节变量，存放网点名称。

GA1、GA2、GAN：字节变量，存放商品信息。

BUF1、BUF2、BUF3、BUF4:字节变量，存放输出提示语句。

CRLF：字节变量，存放回车换行。

d)寄存器分配：

AX：主要用于算术运算

CX, BX：主要用于计数器

SI, DI：临时寄存器，存储计算结果。

汇编语言程序设计实验报告

3.4.2 流程图

代码总体流程图如 1.12 所示：

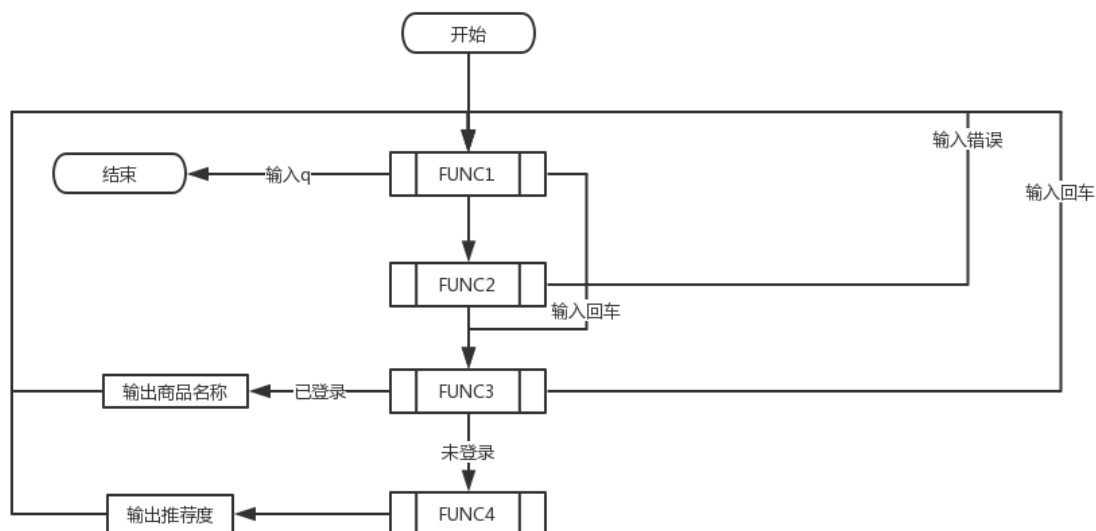


图 1.12 程序总体流程图

程序各功能流程图如图 1.13，1.14，1.15(a), (b), (c)，1.16 所示：

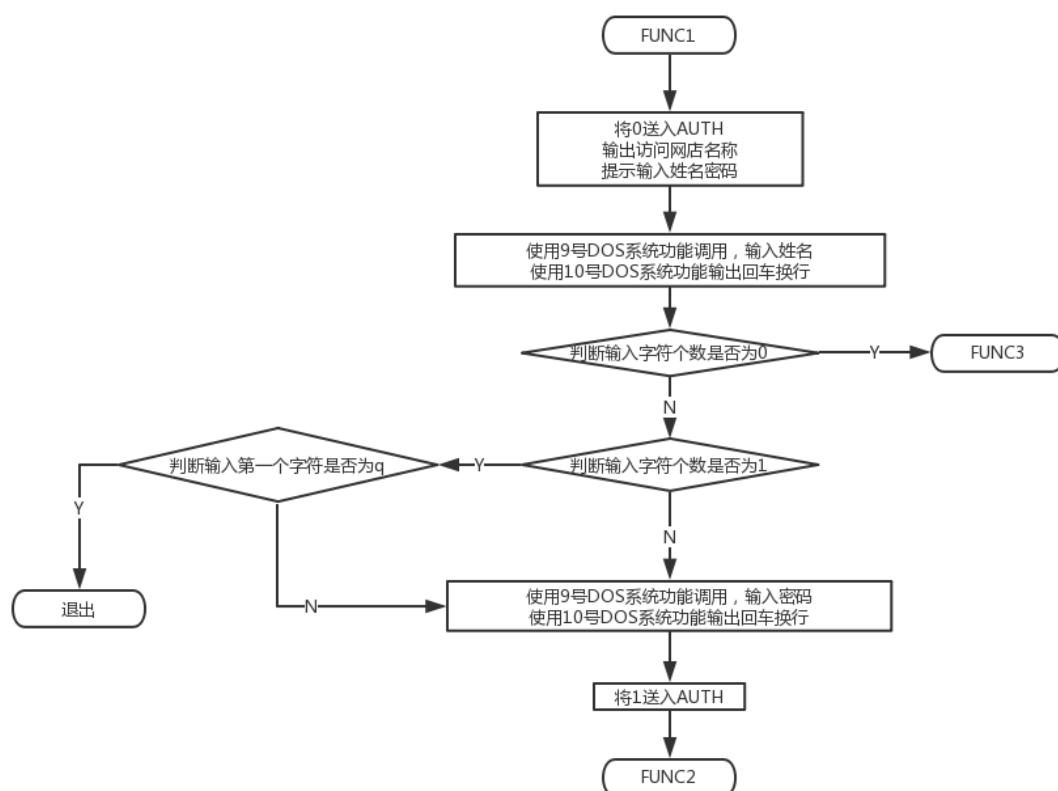
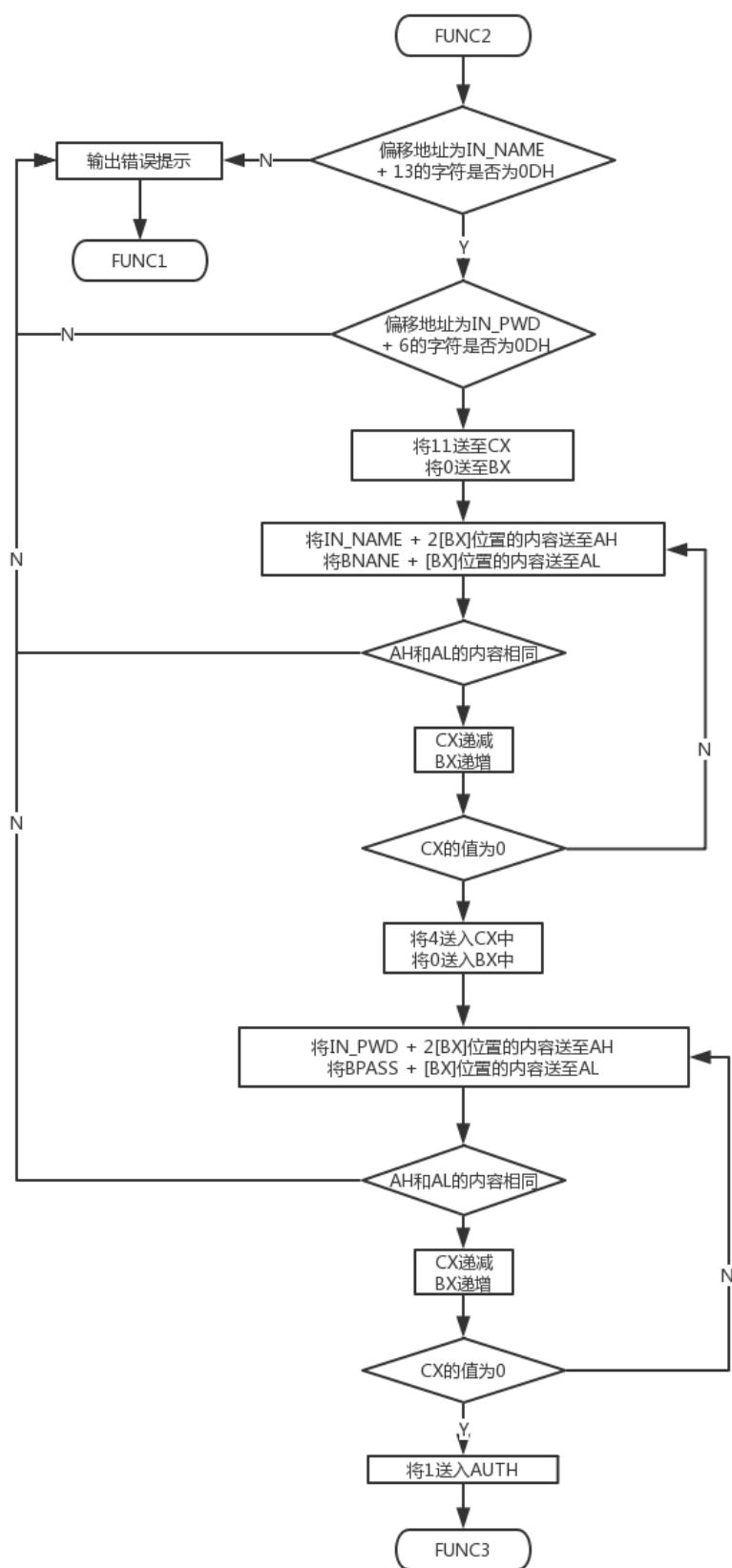


图 1.13 程序功能 1 实现流程图

汇编语言程序设计实验报告



汇编语言程序设计实验报告

图 1.14 程序功能 2 实现流程图

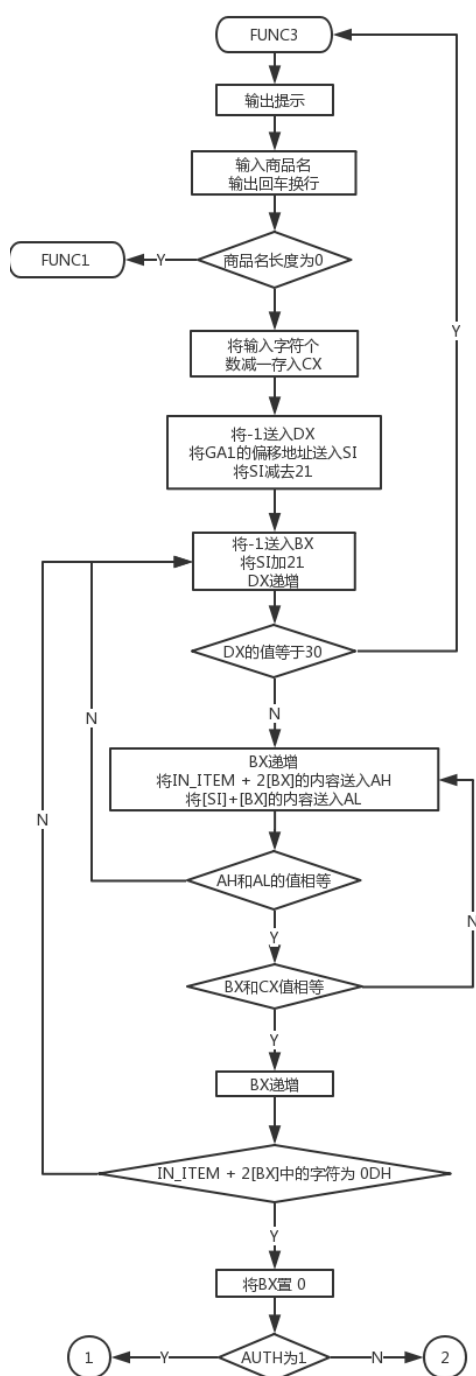


图 1.15(a) 程序功能 3 实现流程图 1

汇编语言程序设计实验报告

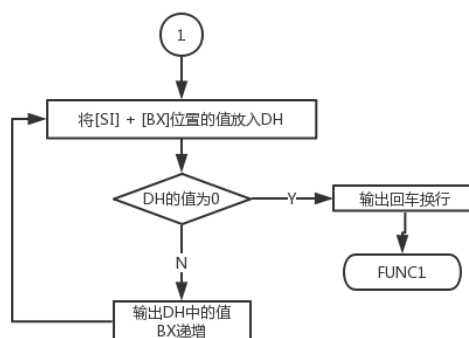


图 1.15(b) 程序功能 3 实现流程图 2

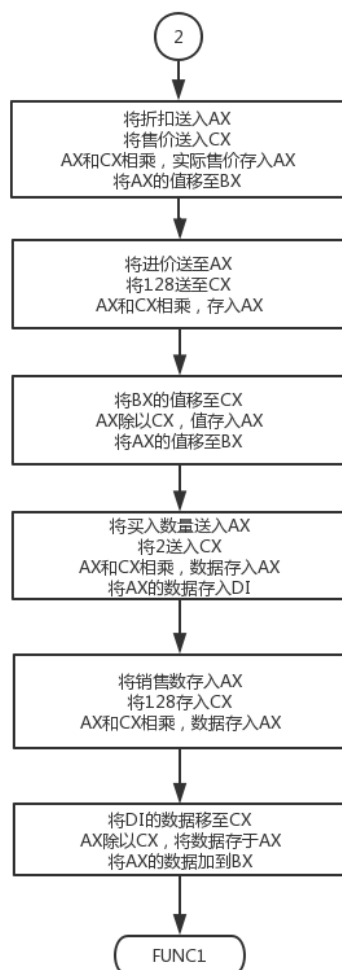


图 1.15(c) 程序功能 3 实现流程图 3

汇编语言程序设计实验报告

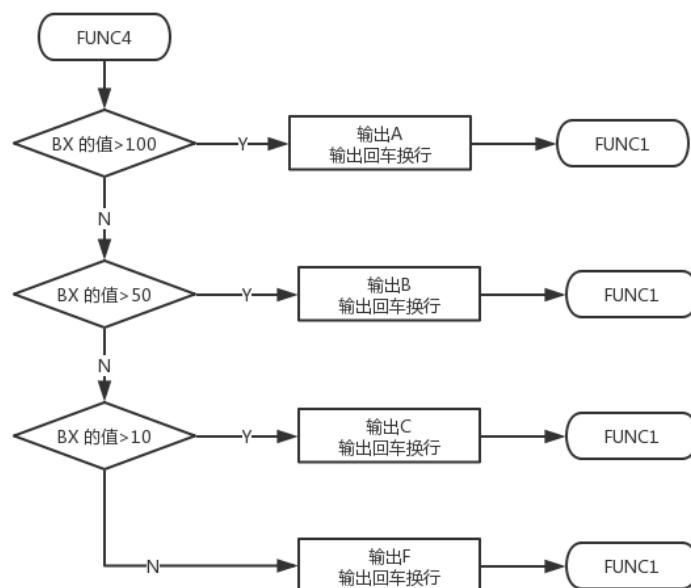


图 1.16 程序功能 4 实现流程图

3.4.3 源程序

```
. 386
STACK SEGMENT USE16 STACK
    DB 200 DUP(0)
STACK ENDS
DATA SEGMENT USE16
BNAME DB 'liu chenyang', 0 ;owner's name
BPASS DB 'test', 0, 0 ;correct password
IN_NAME DB 15 ;place storing name
    DB 0
    DB 15 DUP(0)
IN_PWD DB 10 ;place storing password
    DB 0
    DB 10 DUP(0)
IN_ITEM DB 20
    DB 0
    DB 20 DUP(0)
N EQU 30
AUTH DB 0
SNAME DB 'ONLINE SHOP'
GA1 DB 'PEN', 7 DUP(0), 10
    DW 35, 56, 70, 25, ?
GA2 DB 'BOOK', 6 DUP(0), 9
    DW 12, 30, 25, 5, ?
GAN DB N-2 DUP('Temp-Value', 8, 15, 0, 20, 0, 30, 0, 2, 0, ?, ?)
BUF1 DB 'WELCOME! YOU ARE VISITING ONLINE SHOP', 0AH, 0DH, '$'
BUF2 DB 'PLEASE ENTER YOUR NAME AND PASSWORD:', 0AH, 0DH, '$'
BUF3 DB 'FAIL TO LOG IN!', 0AH, 0DH, '$'
BUF4 DB 'PLEASE ENTER THEN ITEM YOU WOULD LIKE TO INQUIRE:', 0AH, 0DH, '$'
CRLF DB 0DH, 0AH, '$'
```

汇编语言程序设计实验报告

```
DATA    ENDS

CODE    SEGMENT USE16
        ASSUME CS:CODE, DS:DATA, SS: STACK
START:  MOV AX, DATA
        MOV DS, AX
FUNC1:  MOV AUTH, 0
        LEA DX, BUF1           ;print: YOU ARE VISITING ONLINE SHOP
        MOV AH, 9
        INT 21H
        LEA DX, BUF2           ;print: PLEASE ENTER YOUR NAME AND PASSWORD
        MOV AH, 9
        INT 21H
        LEA DX, IN_NAME        ;scanf: name
        MOV AH, 10
        INT 21H
        LEA DX, CRLF           ;回车换行
        MOV AH, 9
        INT 21H

        CMP IN_NAME + 1, 0      ;if entered nothing, goto FUNC3
        JE  FUNC3

        CMP IN_NAME + 1, 1      ;if entered only one char, goto CODION1 to see more
        JNE CODION1

        CMP IN_NAME+2, 'q'      ;if the name is 'q', exit
        JE  EXT

CODION1: LEA DX, IN_PWD         ;scanf: password
        MOV AH, 10
        INT 21H
        LEA DX, CRLF           ;回车换行
        MOV AH, 9
        INT 21H

        MOV AUTH, 1            ;else let AUTH be 1 and goto FUNC2
        JMP  FUNC2

FUNC2:  CMP IN_NAME + 13, 0DH
        JNE WARN
        CMP IN_PWD + 6, 0DH
        JNE WARN
        MOV CX, 11
        MOV BX, 0
LOP1:   MOV AH, IN_NAME + 2[BX]
        MOV AL, BNAME + [BX]
        CMP AH, AL
        JNE WARN
        DEC CX
        INC BX
        CMP CX, 0
        JNE LOP1
        MOV CX, 4
        MOV BX, 0
LOP2:   MOV AH, IN_PWD + 2[BX]
        MOV AL, BPASS + [BX]
```

汇编语言程序设计实验报告

```
CMP AH, AL
JNZ WARN
DEC CX
INC BX
CMP CX, 0
JNE LOP2
MOV AUTH, 1
JMP FUNC3

WARN:  LEA DX, BUF3          ;print: FAIL TO LOG IN
      MOV AH, 9
      INT 21H
      JMP FUNC1

FUNC3:  LEA DX, BUF4          ;print: PLEASE ENTER THEN ITEM YOU WOULD LIKE TO INQUIRE
      MOV AH, 9
      INT 21H
      LEA DX, IN_ITEM        ;scan the item user want to inquire
      MOV AH, 10             ;scan the item user want to inquire
      INT 21H                ;scan the item user want to inquire
      LEA DX, CRLF           ;回车换行
      MOV AH, 9              ;回车换行
      INT 21H                ;回车换行
      CMP IN_ITEM + 1, 0     ;if user entered nothing, goto FUNC1
      JE  FUNC1              ;if user entered nothing, goto FUNC1
      MOV CL, IN_ITEM + 1
      SUB CL, 1
      MOV CH, 0
      MOV DX, -1             ;DX counts the term of item
      MOV SI, OFFSET GA1
      SUB SI, 21
LOP3:  MOV BX, -1             ;BX counts the term of literal
      ADD SI, 21
      INC DX
      CMP DX, 30
      JE  FUNC3
LOP4:  INC BX
      MOV AH, IN_ITEM + 2[BX]
      MOV AL, [SI]+[BX]
      CMP AH, AL
      JNE LOP3
      CMP BX, CX
      JNE LOP4
      INC BX
      CMP IN_ITEM + 2[BX], 0DH
      JNE LOP3
      MOV BX, 0

      CMP AUTH, 1
      JE  OPITEM

      CMP AUTH, 0
      JE  RECOM

OPITEM: MOV DH, [SI]+[BX]
      CMP DH, 0
      JE  OPITEM1
```

汇编语言程序设计实验报告

```
MOV DL, DH
MOV AH, 2
INT 21H
INC BX
JMP OPITEM
OPITEM1: LEA DX, CRLF          ;回车换行
MOV AH, 9                    ;回车换行
INT 21H
JMP FUNC1

RECOM: MOV AL, [SI] + 10      ;DISCOUNT IN AX
MOV AH, 0
MOV CX, [SI] + 13           ;SALE PRICE IN CX
MUL CX                      ;SALE * DISCOUNT IN AX
MOV CX, 10
MOV DX, 0
DIV CX                      ;ACTUAL SALE PRICE IN AX
MOV BX, AX                  ;ACTUAL SALE PRICE IN BX
MOV AX, [SI] + 11          ;PURCHASE PRICE
MOV CX, 128
MUL CX                      ;PURCHASE PRICE * 128 IN AX
MOV CX, BX                  ;ACTUAL SALE PRICE IN CX
MOV DX, 0
DIV CX                      ;PURCHASE PRICE * 128 / ACTUAL SALE PRICE IN AX
MOV BX, AX                  ;PURCHASE PRICE * 128 / ACTUAL SALE PRICE IN BX
MOV AX, [SI] + 15          ;NUM OF PURCHASE IN AX
MOV CX, 2                   ;2 IN CX
MUL CX                      ;2 * NUM OF PURCHASE IN AX
MOV DI, AX                  ;2 * NUM OF PURCHASE IN DI
MOV AX, [SI] + 17          ;NUM OF SALE
MOV CX, 128
MUL CX                      ;NUM OF SALE * 128 IN AX
MOV CX, DI                  ;2 * NUM OF PURCHASE IN CX
MOV DX, 0
DIV CX                      ;NUM OF SALE * 128 / 2 * NUM OF PURCHASE IN AX
ADD BX, AX
JMP FUNC4
FUNC4: CMP BX, 100
JA PUTA
CMP BX, 50
JA PUTB
CMP BX, 10
JA PUTC
CMP BX, 10
JBE PUTF
PUTA: MOV DL, 'A'
MOV AH, 2
INT 21H
LEA DX, CRLF              ;回车换行
MOV AH, 9                  ;回车换行
INT 21H                    ;回车换行
JMP FUNC1
PUTB: MOV DL, 'B'
MOV AH, 2
INT 21H
LEA DX, CRLF              ;回车换行
MOV AH, 9                  ;回车换行
```

汇编语言程序设计实验报告

```
INT 21H                ;回车换行
JMP FUNC1
PUTC: MOV DL, 'C'
      MOV AH, 2
      INT 21H
      LEA DX, CRLF      ;回车换行
      MOV AH, 9         ;回车换行
      INT 21H          ;回车换行
      JMP FUNC1
PUTF: MOV DL, 'F'
      MOV AH, 2
      INT 21H
      LEA DX, CRLF      ;回车换行
      MOV AH, 9         ;回车换行
      INT 21H          ;回车换行
      JMP FUNC1
EXT:  MOV AH, 4CH
      INT 21H

CODE  ENDS
      END START
```

3.4.4 实验步骤

1. 准备上机实验环境。
2. 使用 VISUAL STUDIO 编辑程序，实现要求功能，保存至 SHOP.ASM。使用 MASM6.0 汇编源文件，观察提示信息，若出错则返回重新编辑 SHOP.ASM，保存后重新汇编，直至不再报错为止。
3. 使用连接程序 LINK.EXE 将生成的 SHOP.OBJ 文件连接成执行文件。
4. 执行程序。按照程序设计要求进行交互，检查是否达到程序设计要求。
5. 使用 TD.EXE 观察 SHOP.EXE 的执行情况。即 TD SHOP.EXE 回车
 - (1) 观察 CS、IP、SP、DS、ES、SS 的值。
 - (2) 单步执行开始 2 条指令，观察 DATA 的实际值，以及 DS 的改变情况。
 - (3) 观察 DS: 0 开始数据区，找到各变量在数据段中的位置和值。
 - (4) 观察执行 9 号语句时，源操作数 DX 的值是否与待输出变量的偏移地址相同。
 - (5) 执行 10 号语句指令，输入用户名。观察 IN_NAME 缓冲区的值的变换。
 - (6) 执行字符串比较段时，观察比较过程中程序如何转跳。
 - (7) 执行乘法与除法操作时，注意观察 AX，DX 值的变化以及计算后结果传向何处。
6. 将程序重新装入 TD 中（或将 CS: IP 重置到 MOV AH, 9 的位置），在执行 9 号功能调用之前，用 TD 将数据段中 INPUT 缓冲区的 '\$' (24H) 改成其他数值（如 00H），再执行 9 号功能调用，观察现象。
7. 10 号功能调用时，输入的字符数超过定义的数量时，观察现象。

3.4.5 实验记录与分析

1. 实验环境条件: i7-7700HQ 2.80GHz, 8G 内存; WINDOWS 10 下 DOSBox0.72; TD.EXE 5.0。
2. 汇编源程序时未发生异常
3. 连接过程未发生异常
4. 模块功能测试:

汇编语言程序设计实验报告

(1) 功能一测试:

网店名称及输入提示语输出测试, 测试结果如图 1.17 所示, 测试结果显示该功能正常。

```
WELCOME! YOU ARE VISITING ONLINE SHOP
PLEASE ENTER YOUR NAME AND PASSWORD:
```

图 1.17 输出网店名称与提示语句测试截图

未登录模式访问测试, 测试结果如图 1.18 所示, 测试结果显示该功能正常。

```
WELCOME! YOU ARE VISITING ONLINE SHOP
PLEASE ENTER YOUR NAME AND PASSWORD:

PLEASE ENTER THEN ITEM YOU WOULD LIKE TO INQUIRE:
_
```

图 1.18 未登录模式访问测试截图

退出程序测试, 测试结果如图 1.19 所示, 测试结果显示该功能正常。

```
WELCOME! YOU ARE VISITING ONLINE SHOP
PLEASE ENTER YOUR NAME AND PASSWORD:
q
C:\MASM60>_
```

图 1.19 退出程序测试截图

登录模式访问测试, 测试结果如图 1.20 所示, 测试结果显示该功能正常。

```
WELCOME! YOU ARE VISITING ONLINE SHOP
PLEASE ENTER YOUR NAME AND PASSWORD:
liu chenyang
test
PLEASE ENTER THEN ITEM YOU WOULD LIKE TO INQUIRE:
```

图 1.20 登录模式访问测试截图

(2) 功能二测试:

比较用户输入姓名正确性测试, 测试结果如图 1.20 和 1.21 所示, 测试结果显示该功能正常。

```
WELCOME! YOU ARE VISITING ONLINE SHOP
PLEASE ENTER YOUR NAME AND PASSWORD:
LIU CHENYAN
test
FAIL TO LOG IN!
WELCOME! YOU ARE VISITING ONLINE SHOP
PLEASE ENTER YOUR NAME AND PASSWORD:
```

图 1.21(a) 错误输入用户名测试截图

```
WELCOME! YOU ARE VISITING ONLINE SHOP
PLEASE ENTER YOUR NAME AND PASSWORD:
';[!LK.3879~
test
FAIL TO LOG IN!
```

图 1.21(b) 错误输入用户名测试截图

比较用户入密码正确性测试, 测试结果如图 1.20 和 1.22 所示, 测试结果显示该功能正常。

汇编语言程序设计实验报告

```
WELCOME! YOU ARE VISITING ONLINE SHOP
PLEASE ENTER YOUR NAME AND PASSWORD:
liu chenyang
testtest
FAIL TO LOG IN!
WELCOME! YOU ARE VISITING ONLINE SHOP
PLEASE ENTER YOUR NAME AND PASSWORD:
_
```

图 1.22(a) 错误输入密码测试截图

```
WELCOME! YOU ARE VISITING ONLINE SHOP
PLEASE ENTER YOUR NAME AND PASSWORD:
liu chenyang
6*.>?
FAIL TO LOG IN!
```

图 1.22(b) 错误输入密码测试截图

登陆失败并回到功能一测试，测试结果如图 1.21 和 1.22 所示，测试结果显示该功能正常。
登陆成功并进入功能三测试，测试结果如图 1.20 所示，测试结果显示该功能正常。

(3) 功能三测试

输出查询提示测试，测试结果如图 1.23 所示，测试结果显示该功能正常。

```
PLEASE ENTER THEN ITEM YOU WOULD LIKE TO INQUIRE:
```

图 1.23 输出查询提示测试截图

未能找到输入商品测试，测试结果如图 1.24 所示，测试结果显示该功能正常。

```
PLEASE ENTER THEN ITEM YOU WOULD LIKE TO INQUIRE:
car
PLEASE ENTER THEN ITEM YOU WOULD LIKE TO INQUIRE:
_
```

图 1.24 未能找到输入商品测试截图

输入回车返回功能一测试，测试结果如图 1.25 所示，测试结果显示功能正常。

```
PLEASE ENTER THEN ITEM YOU WOULD LIKE TO INQUIRE:

WELCOME! YOU ARE VISITING ONLINE SHOP
PLEASE ENTER YOUR NAME AND PASSWORD:
_
```

图 1.25 输入回车返回功能一测试截图

登陆模式下正确查询测试，测试结果如图 1.26 所示，测试结果显示功能正常。

```
PLEASE ENTER THEN ITEM YOU WOULD LIKE TO INQUIRE:
PEN
PEN
WELCOME! YOU ARE VISITING ONLINE SHOP
PLEASE ENTER YOUR NAME AND PASSWORD:
_
```

图 1.26 登陆模式下正确查询测试截图

未登录模式下正确查询测试，测试结果如图 1.27 所示，测试结果显示功能正常。

汇编语言程序设计实验报告

```
PLEASE ENTER THEN ITEM YOU WOULD LIKE TO INQUIRE:
BOOK
B
WELCOME! YOU ARE VISITING ONLINE SHOP
PLEASE ENTER YOUR NAME AND PASSWORD:
```

图 1.27 未登录模式下正确查询测试截图

(4) 功能四测试

推荐度显示测试，测试结果如图 1.27 所示，测试结果显示功能正常。

5. 用 TD 调入 shop.exe 后

(1) (CS) = 0B0AH, (IP) = 0000H, (SP) = 00C8H, (DS) = 0AB7H, (ES) = 0AB7H, (SS) = 0AC7H。可从段首址取值的不同得知，代码段、数据段、堆栈段处在不同的位置，且先后顺序为：数据段、堆栈段、代码段。

(2) 单步执行第二条指令，此时(DS) = 0AD4H，TD 数据区跳至 ES:0000H 处，若需回到 DS:0000H，则需要使用 goto 指令。发现此时各段的实际次序为：堆栈段、数据段、代码段，与源程序定义的各段的次序相一致。

(3) DS: 0000H 开始的数据区存放了 BNAME 变量为首址的用户名，DS: 000CH 开始的数据区存放了以 BPASS 变量为首址的密码。DS: 0012 至 DS: 0045 分别为：IN_NAME、IN_PWD、IN_ITEM 的缓冲区，DS: 0046 开始的数据区存放了网店名称，DS: 0051H 开始的数据区存储了商品的各种信息。具体存放形式如图

```
ds:0000 6C 69 75 20 63 68 65 6E liu chen
ds:0008 79 61 6E 00 74 65 73 74 yan test
ds:0010 00 00 0F 00 00 00 00 00 *
ds:0018 00 00 00 00 00 00 00 00
ds:0020 00 00 00 0A 00 00 00 00
ds:0030 00 00 00 00 00 00 00 00
ds:0038 00 00 00 00 00 00 00 00
ds:0040 00 00 00 00 00 00 4F 4E ON
ds:0048 4C 49 4E 45 20 53 48 4F LINE SHO
ds:0050 50 50 45 4E 00 00 00 00 PPEN
ds:0058 00 00 00 0A 23 00 38 00 8
ds:0060 46 00 19 00 00 00 4F F 1 BO
ds:0068 4F 4B 00 00 00 00 00 00 OK
ds:0070 09 0C 00 1E 00 19 00 05 09 1 0
ds:0078 00 00 00 54 65 6D 70 2D Temp-
ds:0080 56 61 6C 75 65 00 0F 00 Value
ds:0088 14 00 1E 00 02 00 00 00 11 1 0
ds:0090 54 65 6D 70 2D 56 61 6C Temp-Ual
ds:0098 75 65 00 0F 00 14 00 1E ue 11 1
ds:00A0 00 02 00 00 00 54 65 6D 8 Tem
```

图 1.28 具体部分数据存放情况截图

其中，DS: 02C7H 开始存放了程序运行中所需要的所有提示信息字符串，具体存放形式如图 1.29 所示。

汇编语言程序设计实验报告

```

ds:02C0 00 1E 00 02 00 00 00 57  ▲  W
ds:02C8 45 4C 43 4F 4D 45 21 20  ELCOME!
ds:02D0 59 4F 55 20 41 52 45 20  YOU ARE
ds:02D8 56 49 53 49 54 49 4E 47  VISITING
ds:02E0 20 4F 4E 4C 49 4E 45 20  ONLINE
ds:02E8 53 48 4F 50 0A 0D 24 50  SHOP
ds:02F0 4C 45 41 53 45 20 45 4E  LEASE EN
ds:02F8 54 45 52 20 59 4F 55 52  TER YOUR
ds:0300 20 4E 41 4D 45 20 41 4E  NAME AN
ds:0308 44 20 50 41 53 53 57 4F  D PASSWO
ds:0310 52 44 3A 0A 0D 24 46 41  RD:
ds:0318 49 4C 20 54 4F 20 4C 4F  IL TO LO
ds:0320 47 20 49 4E 21 0A 0D 24  G IN!
ds:0328 50 4C 45 41 53 45 20 45  PLEASE E
ds:0330 4E 54 45 52 20 54 48 45  NTER THE
ds:0338 4E 20 49 54 45 4D 20 59  N ITEM Y
ds:0340 4F 55 20 57 4F 55 4C 44  OU WOULD
ds:0348 20 4C 49 4B 45 20 54 4F  LIKE TO
ds:0350 20 49 4E 51 55 52 45 3A  INQUIRE:
ds:0358 0A 0D 24 0D 0A 24 00 00  J$J$

```

图 1.29 字符串具体存放截图

(4) 程序执行 9 号语句时，源操作数 DX 的值与待输出变量的偏移地址相同，如图 1.30 所示。

图 1.30 DX 的值测试截图

(5) 执行 10 号语句指令时，正确输入用户名，执行后找到对应位置观察 IN_NAME 中的值的变换。其中 IN_NAME + 1 处的值为 0BH，为字符串长度，IN_NAME + 2 开始为具体字符串值。如图 1.31 所示。

```

ds:0010 00 00 0F 0B 6C 69 75 20  *liu
ds:0018 63 68 65 6E 79 61 6E 0D  chenyan

```

图 1.31 字符串存储截图

(6) 执行字符串比较时，采用循环结构，程序在 CS:0066H 至 CS:0077H 中循环跳转，每次循环 CX 递减，BX 递增，具体如图 1.32:

汇编语言程序设计实验报告

[]-CPU 80486				1
cs:0066	8AA71400	mov	ah,[bx+0014]	ax 6969
cs:006A	8A870000	mov	al,[bx]	bx 0002
cs:006E	3AE0	cmp	ah,al	cx 0009
cs:0070	7527	jne	0099	dx 035B
cs:0072	49	dec	cx	si 0000
cs:0073	43	inc	bx	di 0000
cs:0074	83F900	cmp	cx,0000	bp 0000
cs:0077	75ED	jne	0066 ↑	sp 00C8

图 1.32 循环结构运行截图

(7) 执行乘法操作时, AX 的值发生了改变而由于高位为零, DX 无变化, 乘法的结果存入了 AX 中, 如图 1.33 所示:

[]-CPU 80486				1
cs:0121	8A440A	mov	al,[si+0A]	ax 0230
cs:0124	B400	mov	ah,00	bx 0000
cs:0126	8B4C0D	mov	cx,[si+0D]	cx 0038
cs:0129	F7E1	mul	cx	dx 0000
cs:012B	B90A00	mov	cx,000A	si 0051
cs:012E	BA0000	mov	dx,0000	di 008C

图 1.33 乘法运算后寄存器情况截图

执行除法操作时, AX 和 DX 的值都有可能发生变化, 除法的商存入 AX 中, 余数存入 DX 中, 如图 1.34 所示。

[]-CPU 80486				1
cs:0158	8BCF	mov	cx,di	ax 0016
cs:015A	BA0000	mov	dx,0000	bx 0050
cs:015D	F7F1	div	cx	cx 008C
cs:015F	03D8	add	bx,ax	dx 0078
cs:0161	EB00	jmp	0163	si 0051

图 1.34 除法运算后寄存器情况截图

6.将程序重新装入 TD 中, 在执行 9 号功能调用前, 用 TD 将数据段中缓冲区 DS:02E7 的 24H 改为其他数值, 结果如图 1.35 和 1.36 所示。可见程序在没有遇见 24H 的情况下继续输出, 直到遇到在下一段字符串末尾的 24H 后停止输出。

File Edit View Run Breakpoints Data Options Window									
[]-CPU 80486									
cs:0000	B8D40A	mov	ax,0AD4						
cs:0003	8ED8	mov	ds,ax						
cs:0005	C606450000	mov	byte ptr [0045],00						
cs:000A	BAC702	mov	dx,02C7						
cs:000D	B409	mov	ah,09						
cs:000F	CD21	int	21						
cs:0011	BAEF02	mov	dx,02EF						
cs:0014	B409	mov	ah,09						
cs:0016	CD21	int	21						
cs:0018	BA1200	mov	dx,0012						
cs:001B	B40A	mov	ah,0A						
cs:001D	CD21	int	21						
cs:001F	BA5B03	mov	dx,035B						
cs:0022	B409	mov	ah,09						
cs:0024	CD21	int	21						
ds:02D0	59 4F 55 20 41 52 45 20	YOU ARE							
ds:02D8	56 49 53 49 54 49 4E 47	VISITING							
ds:02E0	20 4F 4E 4C 49 4E 45 20	ONLINE							
ds:02E8	53 48 4F 50 0A 0D 00 50	SHOP P							
ds:02F0	4C 45 41 53 45 20 45 4E	LEASE EN							

图 1.35 修改缓冲区数据截图

汇编语言程序设计实验报告

```
C:\MASM60>TD
Turbo Debugger Version 5.0 Copyright (c) 1988,96 Borland International
WELCOME! YOU ARE VISITING ONLINE SHOP
PLEASE ENTER YOUR NAME AND PASSWORD:
```

图 1.36 程序输出截图

7.10 号功能调用时,输入的字符数超过或等于定义的数量(15)时,由于无法读入回车,程序无法完成输入动作。当输入的字符小于等于定义的数量(15)时,程序才能完成输入动作,继续执行程序。如图 1.37 所示。

```
WELCOME! YOU ARE VISITING ONLINE SHOP
PLEASE ENTER YOUR NAME AND PASSWORD:
123456789ACBDEF_
```

1.37(a) 输入的字符数超过或等于定义的数量时程序截图

```
WELCOME! YOU ARE VISITING ONLINE SHOP
PLEASE ENTER YOUR NAME AND PASSWORD:
123456789ACBDE
test
FAIL TO LOG IN!
```

1.37(b) 当输入的字符小于等于定义的数量时程序截图

4 总结与体会

本次实验中,实验者熟悉了 DOSBOX 的使用与操作环境,学会了建立目标源程序、汇编源文件,生成目标文件、连接目标文件与运行调试程序。其中对运行调试工具 Turbo Debugger 的使用有了基本的了解和熟悉。同时,实验者了解了反汇编的定义,对文件在内存中如何存储有了直观的认识。

实验者首先学会了如何对源代码文件进行汇编(MASM),连接(LINK)与运行,同时学会了将虚拟机挂载在需要的位置。

其次实验者对于 Turbo Debugger 有了基本的认识:了解 TD 界面有五个区域,其中代码区能够查看代码的偏移地址、机器指令和目标代码;数据区能够查看偏移地址,数据内容和各个字节所对应的 ASCII 符号;寄存器区能够查看各寄存器的值,且能够切换 16 位与 32 位寄存器;标志寄存器区列出了主要标志位的值;堆栈区则能够查看有效地址与该地址对应的字单元中的内容。在实验中学会熟练使用一些常见的快捷键如: F2: 添加断点, F5: 放大, F8: 单步调试, F9: 运行, F10: 菜单栏。同时在代码区使用右键 GOTO 可以转跳到制定的代码段,而在数据区使用 GOTO 则可以转跳至对应的偏移地址,查看所存储的数据。

在任务一的第二小题中遇到的问题是对于负数的输入存在问题,以-0101001B为例,尝试输入:“10101001B(加符号位)”、“-0101001B(直接输入)”、“11010111B(补码)”,其中直接保留符号输入与补码输入效果相同,存在溢出的情况,结果均为 7AH,因为溢出而产生错误。而采用直接加符号位的方式输入,可以在 AH 获得正确答案的绝对值 86H,符号标志 SF 为 1。由于直接加符号位的做法不符合实际应用,所以最终选择直接输入的方式计算结果,即使存在着由溢出而引发的错误。通过这个问题,弄清了标志寄存器中各个主要标志位所代表的意义,能够帮助实验者更好的了解计算的经过。

汇 编 语 言 程 序 设 计 实 验 报 告

在任务二中，实验者学会了如何添加断点并使用一些快捷键，在需要快速跳至断点时，可选择 F9 直接运行至断点，而在观察数据段开始的 40 个字节内容时，可选择 F8 单步调试，帮助理解每一行机器指令的作用。同时，学会在数据区使用 GOTO，快速跳转至所关心的偏移地址位置。

在实验三中，实验者通过将任务二中的代码改写成满足任务三要求的代码，加深了对汇编语言寻址方式的理解，例如在一开始，没有意识到题目中：“变址寄存器用 32 位寄存器”的要求，仍旧保留了 SI, DI, BX, BP 等寄存器，导致汇编失败。将其改为 ESI, EDI, EBX, EBP 之后汇编成功。在比较反汇编语言与源代码时，加深了对机器码与汇编语句对应关系的理解。在代码区使用 GOTO 转跳时，理解了 IP 指明起始位置的重要性，同时学会了在代码区快速转跳。

在实验四中，实验者第一次完整实现了一个汇编程序，尤其是对寄存器的使用，不同寻址方式的理解有了更深的认识，并通过 debug 与实验报告撰写的过程提高了使用 TD 的熟练程度。

本次实验中，实验者更加体会到了实验前预习的重要性，上机前准备越充分，上机时的目标就更加明确，解决问题也更有针对性。

同时，通过解决程序中出现的 bug，认识了一些报错提示，例如 A2008 说明指令语法错误。

本次实验，对汇编语言有了一个具体的认识，但是对 DOSBOX 和 TD 的使用仍有加强之处，同时自己汇编的能力还有待提高。

汇 编 语 言 程 序 设 计 实 验 报 告

参考文献

- [1] 许向阳. 80X86 汇编语言程序设计上机指南. 武汉: 华中科技大学出版社, 2007: 1-61
- [2] 王元珍. 曹忠升. 韩宗芬. 80X86 汇编语言程序. 武汉: 华中科技大学出版社, 2007