汇 编 语 言 程 序 设 计 实 验 报 告

華中科技大學

课 程 实 验 报 告

课程名称： 汇编语言程序设计实验

实验名称： 实验四　中断与反跟踪

实验时间： 2019-4-24，14：00-17：30　**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： 张瀚元　**报告日期：** 2019 年 5 月 4 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：2019.05.04

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

汇 编 语 言 程 序 设 计 实 验 报 告

1 实验目的与要求

(1) 掌握中断矢量表的概念；
(2) 熟悉 I/O 访问，BIOS 功能调用方法；
(3) 掌握实方式下中断处理程序的编制与调试方法；
(4) 熟悉跟踪与反跟踪的技术；
(5) 提升对计算机系统的理解与分析能力。

2 实验内容

任务 1：用三种方式获取中断类型码 1H 、13H 对应的中断处理程序的入口地址。

要求：首先要进入虚拟机状态，然后

（1） 直接运行调试工具（TD.EXE），在其数据区观察中断矢量表中的信息。

（2） 编写程序，用 DOS 系统功能调用（具体调用方法见教材示例及附录中的描述）方式获取，观察功能调用相应的出口参数与“（1）”看到的结果是否相同 （使用 TD 观看出口参数即可）。

（3） 编写程序，直接读取相应内存单元，观察读到的数据与“（1）”看到的结果是否相同 （使用 TD 观看程序的执行结果即可）。

任务 2：编写一个接管键盘中断的中断服务程序并驻留内存，其主要功能是：在程序驻留并返回到 DOS 操作系统后，输入键盘上的大写字母时都变成了小写字母。

要求：

（1）在 DOS 虚拟机下执行程序，中断服务程序驻留内存。

（2）在 DOS 命令行下键入小写字母时，屏幕显示不变，键入大写时，屏幕显示为小写。执行 TD，在代码区输入指令“mov AX,0”，看是否都变成了小写。执行实验三任务 1 的程序，输入大小写是否正常？

（3）选作：另外单独编写一个中断服务程序的卸载程序，将自己驻留的键盘中断服务程序恢复到原来的状态（只需要还原中断矢量表的信息，先前驻留的程序可以不退出内存）。

任务 3：读取 CMOS 内指定单元的信息，按照 16 进制形式显示在屏幕上。

要求：

（1）在数据段定义一个待读取的 CMOS 内部单元的地址编号。再使用 IN/OUT 指令，读取 CMOS 内的指定单元的信息。

（2）将读取的信息用 16 进制的形式显示在屏幕上。若是时间信息，可以人工判断一下是否与操作系统显示的时间一致。

任务 4：数据加密与反跟踪

在实验三任务 1 的网店商品信息管理程序的基础上，增加输入用户名和密码时，最大错误次数的限制，即，当输入错误次数达到三次时，直接按照未登录状态进入后续功能。老板的密码采用密文的方式存放在数据段中，各种商品的进货价也以密文方式存放在数据段中。加密方法自选（但不应选择复杂的加密算法）。

可以采用计时、中断矢量表检查、堆栈检查、间接寻址等反跟踪方法中的几种方法组合起来进行反跟踪（建议采用两种反跟踪方法，重点是深入理解和运用好所选择的反跟踪方法）。

为简化录入和处理的工作量，只需要定义三种商品的信息即可。

任务 5：跟踪与数据解密

解密同组同学的加密程序，获取各个商品的进货价。

3 实验过程

3.1 任务 1

3.1.1 设计思想及存储单元分配

设计思想：将数据段定义在中断矢量表表头地址 0:0000H。首先通过 INT 1H 和 INT 13H 两个软中断来观察出口参数。其次通过将对应地址位置的数据送入 AX 和 DX 来观察中断程序的偏移地址和段首址。

存储单元分配：

AX：存放中断程序的段首址。

DX：存放中断程序的偏移地址。

3.1.2 源程序

```
INCLUDE MACRO.LIB
.386

STACK   SEGMENT USE16 STACK
        DB 200 DUP(0)
STACK   ENDS

DATA    SEGMENT USE16
DATA    ENDS

CODE   SEGMENT USE16
```

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

START:
        XOR AX, AX
        MOV DS, AX

        MOV AH, 35H
        MOV AL, 01H
        INT 21H
        MOV AH, 35H
        MOV AL, 13H
        INT 21H

        MOV AX, DS:[04H]
        MOV DX, DS:[06H]

        MOV BX, 4CH
        MOV AX, [BX]
        MOV DX, 2[BX]

        MOV AH, 4CH
        INT 21H

CODE    ENDS
        END START
```

3.1.3 实验步骤

1.准备上机实验环境。

2.在 DOSBOX 中打开 TD.EXE 程序，在数据段使用 GOTO 跳转至 0：0000H，观察中断矢量表的信息。

3.使用 VISUAL STUDIO 编写程序，保存至 TASK1.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 TASK1.ASM，保存后重新汇编，直至不再报错为止。

4.使用连接程序 LINK.EXE 将生成的 TASK1.OBJ 文件连接成执行文件。

5.使用 TD.EXE 观察功能调用相应的出口参数。比较与步骤 2 中所观察到的是否相同。

6.使用 TD.EXE 观察读到的数据。比较与步骤 2 中所观察到的是否相同。

7.若观察得到的结果不同，分析原因。

8.简述如何在 TD 中在数据区切换到中断矢量表所在的内存区域。

9.简述如何计算某个中断入口在中断矢量表中断偏移地址。

3.1.4 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 TD，使用 GOTO 转调至 0：0000H，观察中断矢量表中的数据如图 4.1(a)、(b)所示

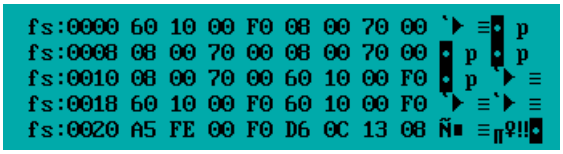

图 4.1(a)中断矢量表前 10 个中断处理程序入口地址

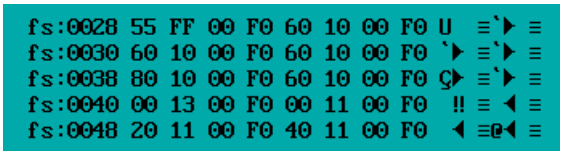

图 4.1(b)中断矢量表第 10 至第 19 个中断处理程序入口地址

根据截图可知，中断 1H 处理程序入口地址的 IP:0008H, CS:0070H, 中断 13H 处理程序入口地址的 IP:1140H, CS:F000H

3.汇编正常。

4.连接正常。

5.打开 TD 观察出口参数：

进入 INT 1H 后寄存器及堆栈状况如图 4.2 所示：

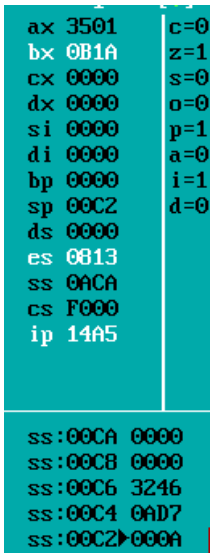

图 4.2 进入 INT 1H 后寄存器及堆栈状况截图

此时中断 1H 处理程序入口地址的 IP:0B1AH, CS:0813H。与中断矢量表观察的情况不同。同时可知 FLAGS、CS 和中断结束后转跳的地址依次被压入栈中。

进入 INT 13H 后寄存器及堆栈状况如图 4.3 所示：

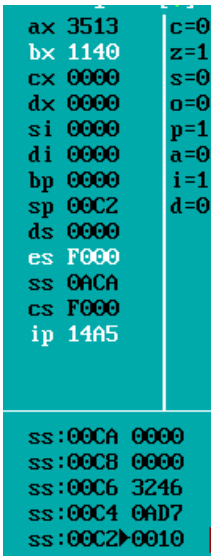

图 4.3 进入 INT 13H 后寄存器及堆栈状况截图

可见中断 13H 处理程序入口地址的 IP:1140H, CS:F000H。与在中断矢量表中显示的相同。同时可知 FLAGS、CS 和中断结束后转跳的地址依次被压入栈中。

6. 打开 TD 观察读取的数据，截图如图 4.4、4.5 所示

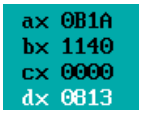

图 4.4 01H 中断时 AX 寄存器和 DX 寄存器数据截图

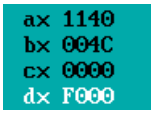

图 4.5 13H 中断时 AX 寄存器和 DX 寄存器数据截图

程序将对应中断程序入口地址的 IP 送入 AX 寄存器，将 CS 送入 DX 寄存器。由截图可知，01H 处理程序入口地址和中断矢量表中观察的不同，而 13H 处理程序入口地址和中断矢量表中观察的结果相同。

7.在 TD 中观察 1H 中断的偏移地址和段首址与中断矢量表中记录的不同，其原因在于 1 号与 3 号调用被 TD 使用，在 TD 中观察得到的结果受到 TD 的影响。

8.在数据区切换至中断矢量表所在的内存区域的方法是：右键选择 GOTO，转调至 0:0000，此时数据区显示的内容就是中断矢量表。

9.计算某个中断入口在中断矢量表中的偏移地址的方法为： (0:[N*4+2])→ CS

3.2 任务 2

3.2.1 设计思想及存储单元分配

设计思想：首先保存原先中断程序的首地址和偏移地址；关闭中断，将新的中断程序的段首址和偏移地址存入对应中断矢量表并开中断；利用 31H 系统功能调用主流程序。

新的中断程序：判断入口参数是否为 00H 或 10H，如果不是则调用原有中断程序，否则首先调用原先中断程序，然后判断 AL 中的出口参数是否为大写字母，若是则将其改为小写字母，否则不变，结束新的中断程序。

恢复原先中断程序：滞留程序中保留了 OLD_INT 的存储空间，利用现有中断矢量表获得新中断程序的段首址和偏移地址，前两个字的数据即为原先中断程序的段首址和偏移地址。将其更新至中断矢量表，恢复原先中断。

存储单元分配：

OLD_INT：保存原有中断程序的段首址和偏移地址。

DS：存放中断矢量表表头地址。

AH：存放入口和出口参数。

AL：存放出口参数。

3.2.2 流程图

程序流程图如图 4.6 所示。

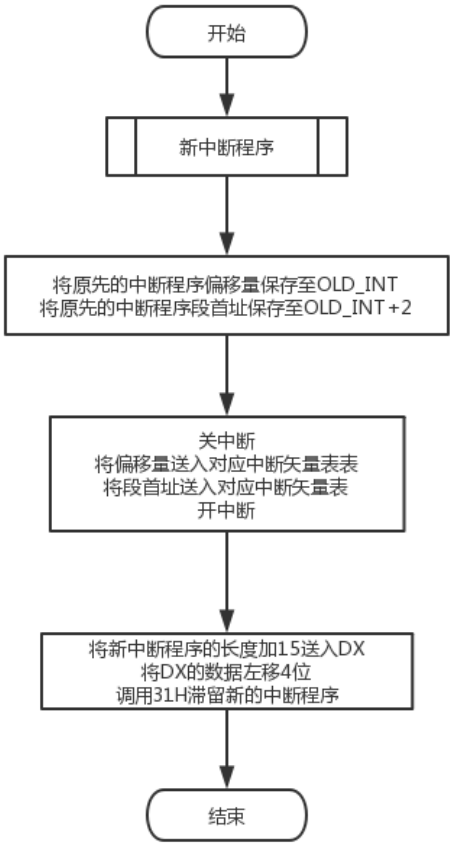

图 4.6 程序流程图

新中断程序的流程图如图 4.7 所示。

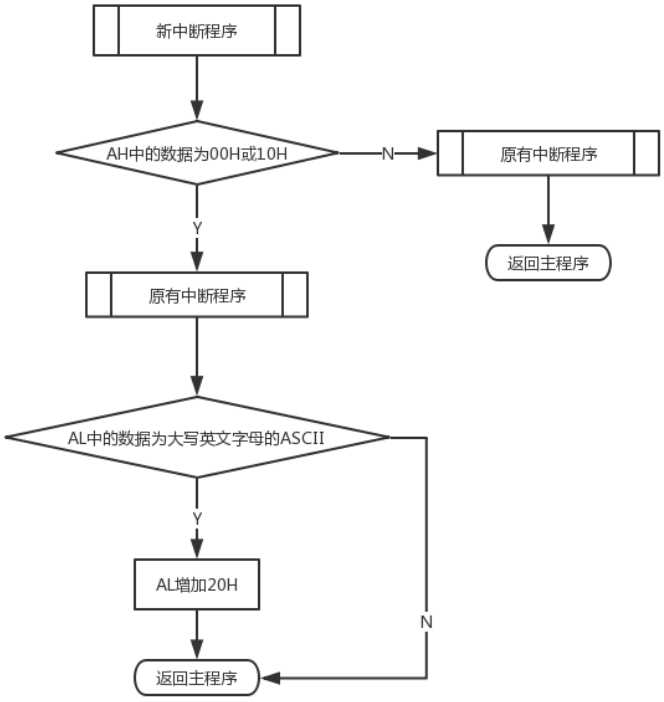

图 4.7 新中断程序的流程图

3.2.3 源程序

（1）接管键盘中断的中断服务程序：

```
.386

CODE    SEGMENT USE16
        ASSUME CS:CODE, SS:STACK
OLD_INT DW ?, ?

NEW16H:
        CMP     AH, 00H
        JE      LOWERCASE
        CMP     AH, 10H
        JE      LOWERCASE
        JMP     DWORD  PTR  OLD_INT

LOWERCASE:
        PUSHF
        CALL    DWORD  PTR  OLD_INT
        CMP     AL, 41H
        JAE     NEXT
        JMP     QUIT
```

```
NEXT:
        CMP     AL, 5AH
        JBE     NEXT1
        JMP     QUIT
NEXT1:
        ADD     AL, 20H
QUIT:
        IRET


START:  XOR AX, AX
        MOV DS, AX
        MOV AX, DS:[16H * 4]
        MOV OLD_INT, AX                 ;将原先中断程序的偏移量保存
        MOV AX, DS:[16H * 4 + 2]
        MOV OLD_INT + 2, AX             ;将原先中断程序的段值保存
        CLI                             ;关中断
        MOV WORD PTR DS:[16H * 4], OFFSET NEW16H   ;更新偏移量
        MOV DS:[16H * 4 + 2], CS        ;更新段值
        STI                             ;开中断
        ;滞留内存
        MOV DX, OFFSET START + 15
        MOV CX, 4
        SHR DX, CL
        ADD DX, 10H
        MOV AL, 0
        MOV AH, 31H
        ;程序结束
        INT 21H
CODE ENDS

STACK   SEGMENT USE16 STACK
        DB 200 DUP(0)
STACK   ENDS
        END START
```

（2）恢复原先键盘中断程序的代码：

```
INCLUDE MACRO.LIB
.386

CODE    SEGMENT USE16
        ASSUME CS:CODE, SS:STACK

START:  XOR AX, AX
```

```
        MOV DS, AX
        CLI                                 ;关中断
        MOV WORD PTR DS:[16H * 4], 11E0H    ;更新偏移量
        MOV WORD PTR DS:[16H * 4 + 2], 0F000H         ;更新段值
        STI                                 ;开中断
        MOV AH, 4CH
        INT 21H
CODE ENDS

STACK   SEGMENT USE16 STACK
        DB 200 DUP(0)
STACK   ENDS
        END START
```

3.2.4 实验步骤

1.准备上机实验环境。

2.使用 VISUAL STUDIO 修改实验一中的程序，要求满足本次实验要求，保存至 TASK2.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 TASK2.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 TASK2.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.同时打开另外一个虚拟 DOS 窗口，观察键盘大小写是否被替代。

3.2.5 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. 连接过程中未发生异常

4. 检查是否满足设计要求：

（1）执行中断服务程序，执行后在 DOS 窗口输入‘TASK2’,测试截图如图 4.8 所示。测试显示功能正常。

图 4.8 运行 TASK2.EXE 后输入大写字母显示截图

（2）执行卸载中断服务程序，执行后在 DOS 窗口输入‘TASK2’，测试截图如图 3.9 所示，该程序功能正常。

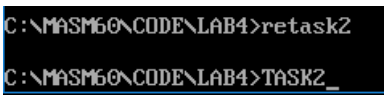

图 4.9 运行 RETASK2.EXE 后输入大写字母显示截图

5. 同时打开另外一个虚拟 DOS 窗口，观察键盘大小写是否被替代。测试截图如图 4.10 所示。测试结果显示功能正常。

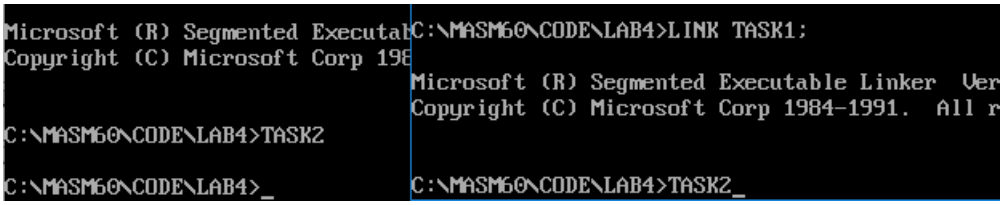

图 4.10 新建另一虚拟 DOS 窗口测试截图

3.3 任务 3

3.3.1 设计思想及存储单元分配

设计思想：将待读取的地址编号存入变量 NUM 中，将 NUM 的值送入 AL 后利用 IN/OUT 指令读取指定单元的内容，并存入 AL 中。然后调用 RADIX 函数以十六进制输出。

单元分配：

NUM：存放待读取的地址编号。

BUFA：输出缓冲区。

AX：存放 RADIX 输入变量。

3.3.2 流程图

程序流程图如图 4.11 所示，子程序 RADIX 流程图见图 3.6。

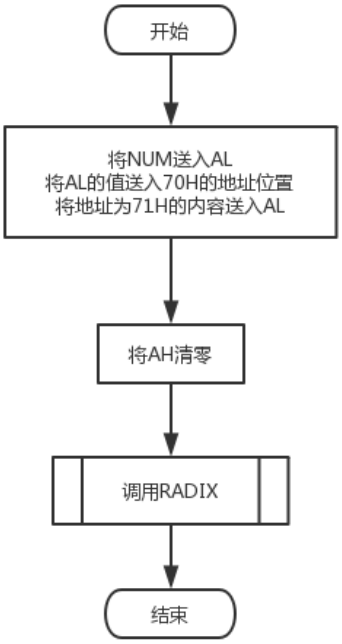

图 4.11 程序流程图

3.3.3 源程序

```
;读取 CMOS 制定给单元的信息，按 16 进制显示在屏幕上
INCLUDE MACRO.LIB
.386
;堆栈段定义
STACK   SEGMENT USE16 STACK
        DB 200 DUP(0)
STACK   ENDS

;数据段定义
DATA    SEGMENT USE16
BUFA    DB 50 DUP(0)
DATA    ENDS

;代码段定义
CODE    SEGMENT USE16
        ASSUME CS:CODE,DS:DATA, SS:STACK

RADIX PROC
        PUSHA
        MOV EBX, 16
        LEA SI, BUFA
```

```
        XOR CX, CX
LOP1:   XOR EDX, EDX
        DIV EBX
        PUSH DX
        INC CX
        OR EAX,EAX
        JNZ LOP1
LOP2:   POP AX
        CMP AL, 10
        JB L1
        ADD AL,7
L1:     ADD AL, 30H
        MOV [SI], AL
        INC SI
        LOOP LOP2
        MOV BYTE PTR [SI], 0DH
        MOV BYTE PTR [SI + 1], 0AH
        MOV BYTE PTR [SI + 2], '$'
        WRITE BUFA
        POPA
        RET
RADIX ENDP
NUM     DB 4
START:  MOV AL, NUM
        OUT 70H, AL
        IN AL, 71H
        XOR AH, AH
        CALL RADIX
        MOV AH, 4CH
        INT 21H
CODE    ENDS
        END START
```

3.3.4 实验步骤

1.准备上机实验环境。

2.使用 VISUAL STUDIO 修改实验一中的程序，要求满足本次实验要求，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 TASK3.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 TASK3.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。

3.3.5 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. 连接过程中未发生异常

4. 执行程序。分别读取 CMOS 中数据地址中的 09H，08H，07H 和 06H 中的值，分别对应年，月，日，结果如图 4.12 所示。

图 4.12 年，月，日截图

5. 执行程序，分别读取 CMOS 中数据地址中的 04H，02H 和 00H 中的值，分别当前时间的时，分，秒，结果如图 4.13 所示。

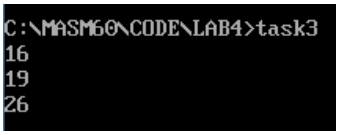

图 4.13 时，分，秒获取结果

6. 对比当前系统时间，如图 4.14 所示，可知程序功能正常。

图 4.14 系统时间

3.4 任务 4

3.4.1 设计思想及存储单元分配

设计思想：使用计时和检查中断矢量表的方式来抵制动态跟踪调试。对密码加密，采用每一个密码字符与后一个字符异或的方式来加密，商品进货量通过数字与字符“L”异或进行加密。同时，定义变量 TURNS,每次登录错误进行加一计数，当 TURNS 大于 3 时自动进入到访客模式。

单元分配：见实验三。

3.4.2 源程序

主程序修改后源代码如下：

```
NAME SHOP
EXTRN COUNT_RECOM:NEAR ,RANK_RECOM:NEAR    ,LDISPLAY:NEAR, INQURE: NEAR,
```

```
ALTER: NEAR
            PUBLIC GA1,RANK
              PUBLIC    GA1, RANK
              INCLUDE MACRO.LIB
        .386
        STACK     SEGMENT USE16 STACK 'STACK'
              DB 200 DUP(0)
        STACK     ENDS
        DATA      SEGMENT USE16 PUBLIC 'DATA'
        BNAME      DB 'liu chenyan', 0, 0          ;owner's name
        BPASS      DB 't' XOR 'e', 'e' XOR 's', 's' XOR 't', 't' XOR 0DH, 0, 0                ;correct password
        IN_NAME DB 15                              ;palce 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'
        CHOICE    DB 0
        RANK        DW 50 DUP(0)
        GA1         DB 'PEN', 7 DUP(0), 10
                   DW 35 XOR 'L', 56, 70, 25, ?
        GA2         DB 'BOOK', 6 DUP(0), 9
                   DW 12 XOR 'L', 30, 25, 5, ?
        GAN         DB 1 DUP('Temp-Value', 8, 15 XOR 'L', 0, 20, 0, 30, 0, 2, 0, ?, ?)
        BUF1        DB 0AH, 0DH,'----------------------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, '$'
        BUF4 DB 'PLEASE ENTER THEN ITEM YOU WOULD LIKE TO PURCHASE:', 0AH, 0DH, '$'
        BUF5 DB 'SORRY THE ITEM YOU WANT ISNT AVAILABLE', 0AH, 0DH, '$'
        BUF6        DB 'PLEASE ENTER THE ITEM THAT YOU WOULD LIKE TO CHANGE ITS
INFORMATION:', 0AH, 0DH, '$'
        BUF7 DB '---------------------------------FUNCTION MENU---------------------------------', 0AH, 0DH, '$'
        BUF8 DB '1. INQURE ITEM INFORMATION                                    2. ALTER ITEM
INFORMATION', 0AH, 0DH, '$'
        BUF9 DB '3. CALCULATE RECOMMENDATION                                    4. RANK
RECOMMENDATION      ', 0AH, 0DH, '$'
```

```
    BUF10       DB '5. DISPLAY ALL ITEM INFORMATION                                    6. QUIT
', 0AH, 0DH,'$'
    BUF11       DB '--------------------------------------------------------------------------------', 0AH, 0DH, '$'
    BUF12      DB 'PLEASSE ENTER THE NUMBER OF THE OPERATION YOU WANT CONDUCT:', 0AH,
0DH, '$'
    BUF13       DB 'RECOMMENDATION OF ALL ITEMS CALCULATED', 0AH, 0DH, '$'
    BUF14       DB 'RECOMMENDATION OF ALL ITEMS RANKED', 0AH, 0DH, '$'
    BUF15       DB '1. INQURE ITEM INFORMATION                                         2. QUIT
', 0AH, 0DH, '$'
    BUF16       DB 'TRIED TOO MANY TIMES, INTO THE CUSTOMER MODE', 0AH, 0DH, '$'
    TURNS       DB 0
    OLDINT1   DW 0, 0
    OLDINT3   DW 0, 0
    DATA        ENDS

    CODE        SEGMENT USE16 PARA PUBLIC 'CODE'
                ASSUME CS:CODE, DS:DATA, SS: STACK
    START:    MOV AX, DATA
          MOV DS, AX
          XOR    AX,AX                        ;接管调试用中断，中断矢量表反跟踪
               MOV    ES,AX
               MOV    AX,ES:[1*4]                ;保存原 1 号和 3 号中断矢量
               MOV    OLDINT1,AX
               MOV    AX,ES:[1*4+2]
               MOV    OLDINT1+2,AX
               MOV    AX,ES:[3*4]
               MOV    OLDINT3,AX
               MOV    AX,ES:[3*4+2]
               MOV    OLDINT3+2,AX
               CLI                                ;设置新的中断矢量
               MOV    AX,OFFSET NEWINT
               MOV    ES:[1*4],AX
               MOV    ES:[1*4+2],CS
               MOV    ES:[3*4],AX
               MOV    ES:[3*4+2],CS
               STI

    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
        CRLF                ;回车换行


        CMP IN_NAME + 1, 0              ;if entered nothing, goto FUNC3
        JE   FUNC3_1

        CMP IN_NAME + 1, 1              ;if entered only one char, goto CODION1 to see more
        JNE CODON1

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

CODON1:  LEA DX, IN_PWD             ;scanf: password
        MOV AH,10
        INT 21H
        CRLF                ;回车换行

        JMP   FUNC2

FUNC2:     CMP IN_NAME + 13, 0DH ;START OF THE FUNCTION 2
        JNE WARN
        MOV CX, 11
        MOV BX, 0
LOP1:       MOV AH, IN_NAME + 2[BX]     ;COMPARE YOUR NAME
        MOV AL, BNAME + [BX]
        CMP AH, AL
        JNE WARN
        DEC CX
        INC BX
        CMP CX, 0
        JNE LOP1
        ;计时反跟踪开始
        CLI
        MOV AH, 2CH
        INT 21H
        PUSH DX
        ;开始比对密码是否正确
        CMP IN_PWD + 6, 0DH
        JNE WARN
        ;终止计时程序
        MOV AH, 2CH
```

```
        INT 21H
        STI
        CMP DX, [ESP]
        POP DX
        JNE EXT
        MOV CX, 4
        MOV BX, 0
LOP2:   MOV AH, IN_PWD + 2[BX]      ;COMPARE YOUR PASSWORD
        MOV AL, IN_PWD + 3[BX]
        XOR AH, AL
        MOV AL, BPASS + [BX]
        CMP AH, AL
        JNZ WARN
        DEC CX
        INC BX
        CMP CX, 0
        JNE LOP2
        MOV AUTH, 1
        JMP CHECK
        ;修改中断矢量表反跟踪
CHECK:  MOV BX, ES:[1*4]
        INC BX
        JMP BX
        ;登录用户名或密码错误
WARN:   WRITE BUF3                  ;print: FAIL TO LOG IN
        INC TURNS
        CMP TURNS, 3
        JB FUNC1
        WRITE BUF16
        CRLF
        JMP FUNC3_1
FUNC3_1:
        WRITE BUF7
        WRITE BUF15
        WRITE BUF11
        WRITE BUF12
        MOV AH, 1
        INT 21H
        MOV CHOICE, AL
        CRLF                        ;回车换行
        CMP AL, 0AH
        JE FUNC3_1

        CMP CHOICE, '1'
```

```
        JE SUBFUNC1

        CMP CHOICE, '2'
        JE SUBFUNC6

        JMP FUNC3_1

FUNC3:  WRITE BUF7              ;BELOW PRINT THE FUNCTION MENU
        LEA DX, BUF8
        MOV AH, 9
        INT 21H
        LEA DX, BUF9
        MOV AH, 9
        INT 21H
        LEA DX, BUF10
        MOV AH, 9
        INT 21H
        LEA DX, BUF11
        MOV AH, 9
        INT 21H
        LEA DX, BUF12
        MOV AH, 9
        INT 21H                 ;ABOVE PRINT THE FUNCTION MENU
        MOV AH, 1
        INT 21H
        MOV CHOICE, AL
        CRLF                    ;回车换行
        CMP AL, 0AH
        JE FUNC3

        CMP CHOICE, '1'
        JE SUBFUNC1

        CMP CHOICE, '2'
        JE SUBFUNC2

        CMP CHOICE, '3'
        JE SUBFUNC3

        CMP CHOICE, '4'
        JE SUBFUNC4

        CMP CHOICE, '5'
        JE SUBFUNC5
```

```
        CMP CHOICE, '6'
        JMP SUBFUNC6
SUBFUNC1:
        CALL INQURE
        CMP AUTH, 0
        JE FUNC3_1
        JMP FUNC3
SUBFUNC2:
        CALL ALTER
        JMP FUNC3
SUBFUNC3:
        CALL COUNT_RECOM
        WRITE BUF13
        JMP FUNC3
SUBFUNC4:
        CALL RANK_RECOM
        WRITE BUF14
        JMP FUNC3
SUBFUNC5:
        CALL LDISPLAY
        JMP FUNC3
SUBFUNC6:
        MOV AUTH, 0
        MOV TURNS, 0
        JMP FUNC1
        ;新的中断矢量表 1H 和 3H 的终端地址
NEWINT: IRET
TESTINT: JMP FUNC3
EXT:    CLI                             ;还原中断矢量
        MOV   AX,OLDINT1
        MOV   ES:[1*4],AX
        MOV   AX,OLDINT1+2
        MOV   ES:[1*4+2],AX
        MOV   AX,OLDINT3
        MOV   ES:[3*4],AX
        MOV   AX,OLDINT3+2
        MOV   ES:[3*4+2],AX
        STI
;程序退出
        MOV AH, 4CH
        INT 21H
CODE    ENDS
        END START
```

3.4.3 实验步骤

1.准备上机实验环境。

2.使用 VISUAL STUDIO 修改实验一中的程序，要求满足本次实验要求，保存至 FSHOP.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 FSHOP.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 FSHOP. OBJ,SHOPE1.OBJ, SHOPE2.OBJ, SHOPE3.OBJ 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。

3.4.4 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. 连接过程中未发生异常

4.检查程序是否满足设计要求：

（1）登录错误三次进入访客模式测试，测试截图如图 4.15 所示，测试结果显示功能正常。

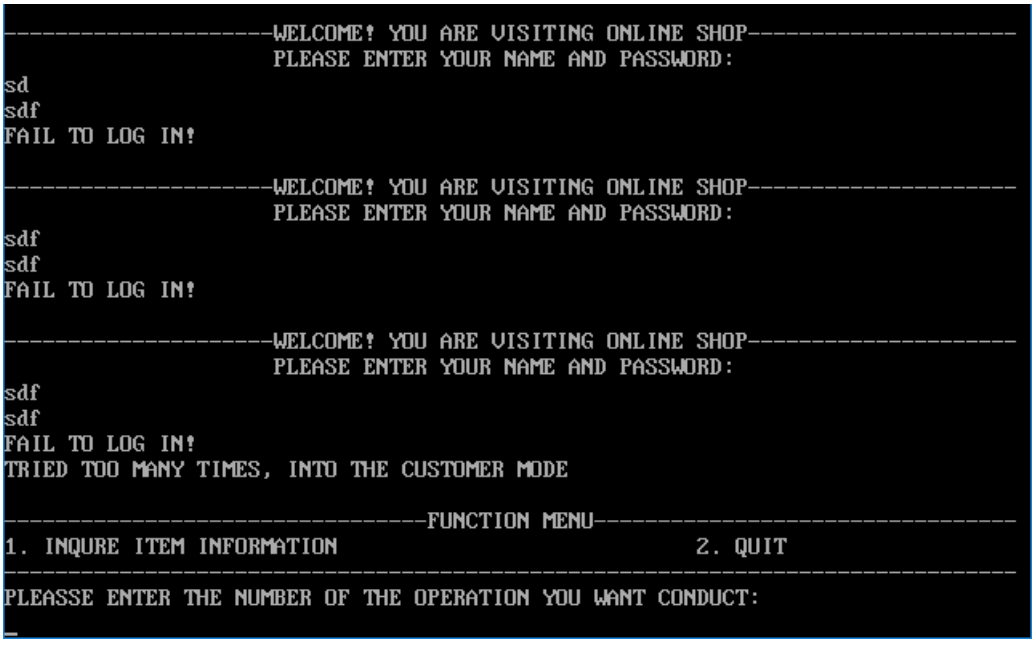

图 4.15 登陆错误三次进入访客模式测试截图

（2）密码以密文方式保存在内存测试，测试结果如图 4.16 所示，密码为用户名之后的四位字符，分别为 11H, 16H, 07H, 79H，而密码明文为”test”，故测试结果显示功能正常。

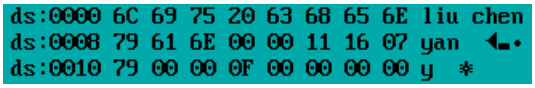

图 4.16 密码以密文方式保存在内存中测试截图

（3）商品进货价以密文方式保存在数据段中测试，测试结果如图 4.17 所示，内存位置 DS:00C8 所在位置的数据并非 0023H 而是以密文方式保存的 006FH。测试结果显示功能正常。

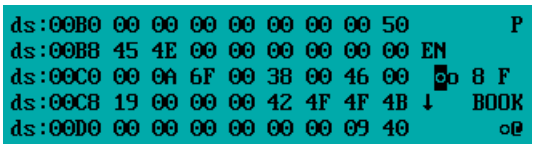

图 4.17 商品进货价以密文方式保存在数据段中测试截图

（4）使用 TD 进行跟踪调试无效测试时，测试结果如图 4.18 所示，在修改中断矢量表时遭到调试工具组织，程序出错。测试结果显示功能正常。

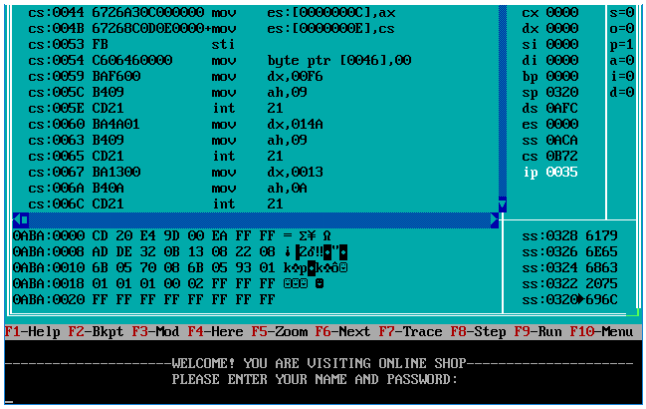

图 4.18 修改中断矢量使 TD 跟踪调试失败截图

（5）计时反跟踪使 TD 跟踪调试失败测试，测试结果如图 4.19 所示，可知当计时不同时，程序直接退出。测试结果显示功能正常

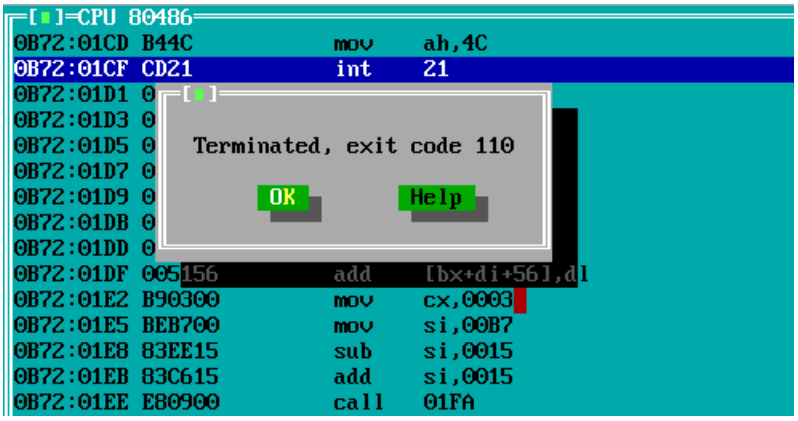

图 4.19 计时反跟踪使 TD 跟踪调试失败测试截图

3.5 任务 5

3.5.1 解决思路

破解思路：首先在 TD 中打开带破解的程序，阅读程序，观察编写者是否使用了一些地址跟踪调试的方法，如果存在，则直接在 TD 中修改该部分代码，防止 TD 被程序阻止。

其次观察代码，获得代码中的密码长度和用户名两个信息，同时根据代码中对输入的密码进行加密的代码，得到编写者对密码的加密方式。

观察程序输出信息的代码，获得代码中对输出商品进货价解密的代码，反向推出编写使用的数据加密方法。

3.5.2 实验步骤

1.准备上机实验环境。

2.直接打开程序运行，观察程序入口情况。

3.打开 TD，观察代码中存储的用户名、密码长度、密码加密方式，商品进货价的加密方式和程序编写者使用的反跟踪方法。

4.利用以上获得的信息尝试用不同方式解密程序。

5.尝试解决以下问题：

（1）将密码明文存放在数据段中，尝试更快的获取密码。

（2）将商品进货价以明文存放在数据段中，尝试更快的获取进货价。

（3）说明如何在程序中观察反跟踪的代码。说明如何应对反跟踪程序。

（4）尝试通过修改 AUTH 的值来达到获取进货价的目的。尝试通过观察程序计算推荐度的过程来获取进货价。

3.5.3 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. 观察程序入口状况：尝试登陆三次错误后将直接进入访客模式，如图 4.20 所示。

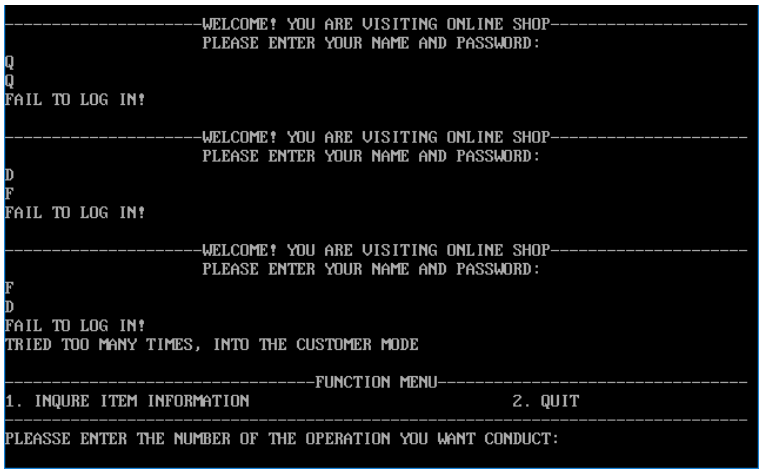

图 4.20 程序无法直接访问登陆模式

3. 打开 TD 观察程序：

（1）观察代码中存储的用户名和加密的密码。观察情况如图 4.21 所示，可知用户名为：zhanghy，密文密码为 0AH 09H 12H 1DH 10H 0CH。

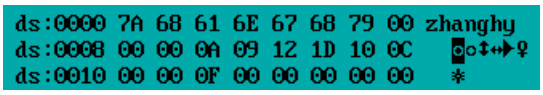

图 4.21 登录用户名

（2）观察代码中存储的密码长度。观察情况如图 4.22 所示，可知登录密码长度为 6 位。

图 4.22 代码中显示的登录密码长度

（3）观察密码加密方式：由图 4.22 所示，可知存储输入密码的缓冲区首址为 DS:0025H，输入缓冲区的密码字符和其前 11H 地址上的数据进行异或后即为加密后的密文密码。再次观察代码，得到图 4.23。可知首址为 DS:0014H 的存储区域存储输入的用户名。故加密方法为：密码的第 i 个字符与用户名的第 i 个字符进行异或操作。

图 4.23 输入用户名的存储位置截图

（4）观察商品进货价的加密方式。观察程序主题跳转情况，进入输出全部商品信息的函数，观察输出商品进货价时程序的解密情况，如图 4.24 所示。可知商品进货价的加密方式为：进货价与 5AH 进行异或操作。

图 4.24 商品进货价解密方式截图

（5）观察程序编写者使用的反跟踪方法。观察反汇编程序，使用的反跟踪方法如图 4.25(a)、(b)、(c)所示。

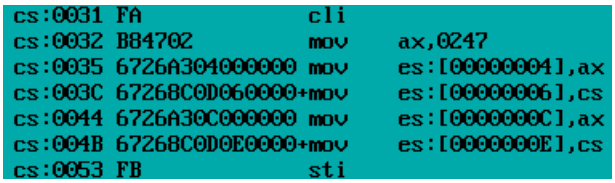

图 4.25(a) 通过检查中断矢量表反跟踪截图

图 4.25(b) 通过计时反跟踪截图

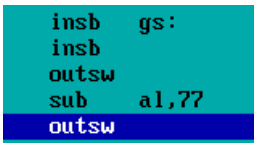

图 4.25(c) 通过增加冗余代码反跟踪截图

4. 利用以上获得的信息尝试用不同方式解密程序。

（1）破解密码登录程序：

根据观察已知密码加密方式为：密码的第 i 个字符与用户名的第 i 个字符进行异或操作。且用户名为：zhanghy（7AH 68H 61H 6EH 67H 68H 79H），密文密码为 0AH 09H 12H 1DH 10H 0CH，故密码为：70H 61H 73H 73H 77H 64H，翻译为字符为：passwd。登陆结果如图 4.26 所示。可知破解成功。获得商品进货价信息如图 4.27(a)、(b)、(c)所示。

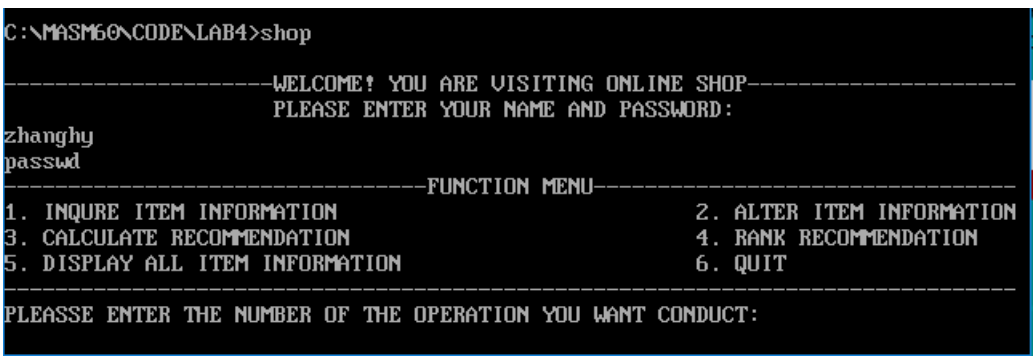

图 4.26 破解密码后登录程序截图

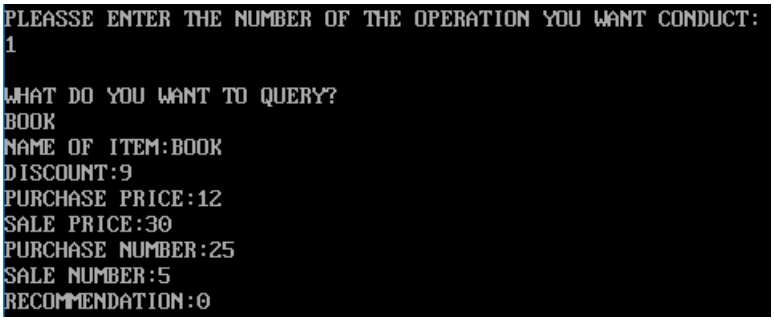

图 4.27(a) BOOK 的信息截图

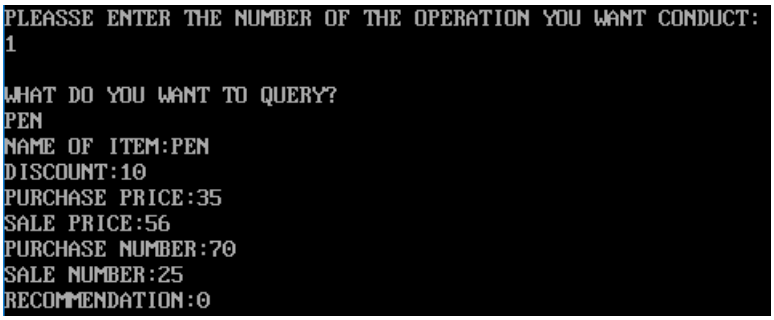

图 4.27(b) PEN 的信息截图

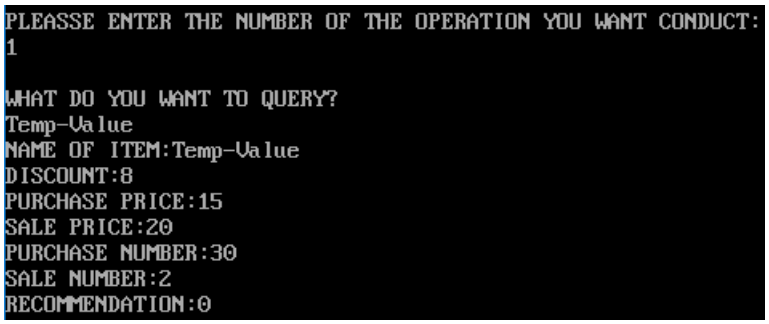

图 4.27(c) Temp-Value 的信息截图

（2）根据商品进货价加密方式破解信息：

根据观察知商品进货价的加密方式为：进货价与 5AH 进行异或操作。在 TD 中可知商品进货价的加密后的数据为：PEN: 79H, BOOK: 56H, Temp-Value: 55H，如图 4.28 所示。故解密后的数据为：PEN: $(35)_{10}$, BOOK: $(12)_{10}$, Temp-Value: $(15)_{10}$ 。和前述破解方式比较可知破解成功。

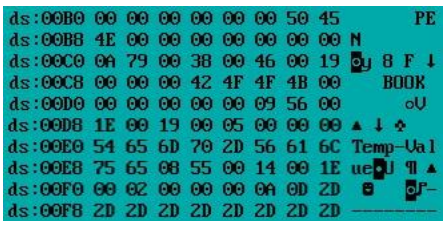

图 4.28 密文保存的商品进货价截图

（3）直接获得子程序入口地址破击信息：

观察程序，已知程序使用了修改中断矢量表的方式阻止动态调试，可获得登录模式的

程序入口，如图 4.29 所示。故将程序在 TD 中修改为：JMP 019E，结果如图 4.30 所示。商品信息获取结果与 4.27(a)、(b)、(c)一致。

图 4.29 登陆模式下的程序入口偏移地址

图 4.30 直接进入登录模式测试截图

5. 解决以下问题：

（1）将密码明文存放在数据段中，尝试更快的获取密码。

若密码以明文方式保存在数据段中，可以首先找到比对输入密码和密码的比较代码，观察代码使用了哪个位置上的数据进行比较，该位置上存储的即为明文密码。

（2）将商品进货价以明文存放在数据段中，尝试更快的获取进货价。

商品进货价使用明文存放在数据段中，可直接观察代码，找到调用商品进货价的子程序，如：推荐度计算，输出商品信息，观察其调用数据位置，即可找到商品进货价格。

（3）说明如何在程序中观察反跟踪的代码。说明如何应对反跟踪程序。

若程序中存在反跟踪代码，则往往需要使用关中断和开中断。故可以通过观察程序中的开关中断部分，检查是否存在反跟踪的设计。对于存在的反跟踪程序，如一般的计时反跟踪，可以通过在 TD 中直接修改反汇编代码使反跟踪程序失效。

（4）尝试通过修改 AUTH 的值来达到获取进货价的目的。尝试通过观察程序计算推荐度的过程来获取进货价。

不能通过修改 AUTH 的值来达到获取进货价的目的。登陆模式的入口条件似乎不是 AUTH 而是用户名和密码输入正确与否。通过观察程序计算推荐度的过程能够获取进价，因为此时能够通过代码了解到数据的加密方式和存储位置。

4 总结与体会

本次实验，我首次直观的观察到中断矢量表，并加深了对中断的理解，了解到中断发生之后程序是如何执行的。在任务二中，通过接管原有的键盘中断程序，能够直接将大写字母替换为小写字母，这是之前在 C 语言实验中无法做法到的，加深了我对计算机汇编层面的认识。

在设计反跟踪与破解他人程序中，我认识到了各种加密方法和反跟踪程序，这些方法能够利用不同的工具，巧妙地抓住跟踪调试的特点并加以阻止。同时，在破解他人程序

时，我也认识到这些反跟踪的技巧都还存在缺陷，跟踪与反跟踪由于矛与盾的关系，此消彼长，相辅相成。

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