

# 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： 实验四 中断与反跟踪

实验时间： 2018-4-28, 14: 00-17: 30, 2018-5-7, 14: 00-17: 30

实验地点： 南一楼 804 室

指导教师： 朱虹

专业班级 计算机 201601 班

学 号： U201614532 姓 名： 吕鹏泽

同组学生： 孙磊 报告日期： 2018 年 5 月 8 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 .....	3
2.	实验内容.....	4
3.	实验过程.....	5
3.1	任务 1.....	5
3.1.1	设计思想及存储单元分配.....	5
3.1.2	源程序 .....	5
3.1.3	实验步骤 .....	6
3.1.4	实验记录与分析.....	6
3.1.5	思考题 .....	8
3.2	任务 2.....	10
3.2.1	设计思想及存储单元分配.....	10
3.2.2	流程图 .....	11
3.2.3	源代码 .....	14
3.2.4	实验步骤 .....	16
3.2.5	实验记录与分析.....	16
3.2.6	思考题 .....	17
3.3	任务 3.....	19
3.3.1	设计思想及存储单元分配.....	19
3.3.2	流程图 .....	19
3.3.3	源代码 .....	19
3.3.4	实验步骤 .....	22
3.3.5	实验记录与分析.....	22
3.3.6	思考题 .....	23
3.4	任务 4.....	24
3.4.1	设计思想及存储单元分配.....	24
3.4.2	流程图 .....	25
3.4.3	源代码 .....	25
3.4.4	实验步骤 .....	33
3.4.5	实验记录与分析.....	33
3.5	任务 5.....	35
3.5.1	设计思想及存储单元分配.....	35
3.5.2	实验记录与分析.....	35
4.	总结与体会 .....	38

5. 参考文献.....	39
--------------	----

# 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、10H 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)，观察中断矢量表中的信息。
- (2) 编写程序，用 DOS 系统功能调用方式获取，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，要求在程序返回 DOS 操作系统后，输入键盘上的小写字母时都变成了大写字母。**

**要求：**

- (1) 在 DOS 虚拟机或 DOS 窗口下执行程序，中断服务程序驻留内存。
- (2) 在 DOS 命令行下键入小写字母，屏幕显示为大写，键入大写时不变。执行 TD，在代码区输入指令“`mov AX,0`”，看是否都变成了大写。
- (3) 选作：**另外**编写一个中断服务程序的卸载程序，将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。

可以采用计时、中断矢量表检查、堆栈检查、间接寻址等方式中的一种或多种方式反跟踪（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

用三种方式获取中断类型码 1H、10H 对应的中断处理程序的入口地址。

1. 直接运行 TD，在数据域使用 goto 语句输入 0:0H 跳转到中断矢量表，找到中断类型码为 1H (0:4H)、10H(0:40H)对应的中断处理程序的入口地址。

2. 系统功能调用 INT 21H，入口参数为 AH=35H, AL=中断类型号, 取中断信息，出口参数 ES:[BX]为中断的入口地址。

3. 先将 DS 赋值为 0，然后先后把主存中的 DS:[4H]→AX, DS:[6H]→BX 和 DS:[40H]→AX, DS:[42H]→BX 赋值给寄存器 BX 和 CX 来观察 1H、10H 的中断处理程序入口地址。

#### 3.1.2 源程序

##### 3.1.2.1 系统功能调用查看中断处理程序入口地址

```
INCLUDE MACRO.LIB
.MODEL SMALL
.386
.STACK 200
.DATA

.CODE
MOV AX,3501H
INT 21H
MOV AX,3510H
INT 21H
EXIT:
MOV AH,4CH
INT 21H
END
```

##### 3.1.2.2 直接读取相应内存单元

```
INCLUDE MACRO.LIB
.MODEL SMALL
.386
.STACK 200
.DATA

.CODE
START:
MOV AX,0
MOV DS,AX
MOV AX,DS:[04H];IP
MOV BX,DS:[06H];CS
```

```

MOV AX,DS:[40H];IP
MOV BX,DS:[42H];CS
EXIT:
MOV AH,4CH
INT 21H
END

```

### 3.1.3 实验步骤

1.直接运行 TD，在数据域使用 goto 语句输入 0:0H 跳转到中断矢量表，找到中断类型吗为 1H（0:4H）、10H(0:40H)对应的中断处理程序的入口地址。

2. 准备上机环境，编辑、汇编、连接文件 T1。运行 TD T1，在 TD 中单步调试，在数据与中观察 ES 和 BX 的数据，与 1 中得到的结果比对。

3. 编辑、汇编、连接文件 T2。运行 TD T2，在 TD 中单步调试，先后移送 1H 和 10H 中断处理程序入口地址后观察 AX 寄存器和 BX 寄存器的值，与 1 中的结果比对。

### 3.1.4 实验记录与分析

实验环境条件：WINDOWS10 下 DOSBox0.72； MASM.EXE 6.0； LINK.EXE 5.2； TD.EXE 5.0。

1.如图所示，在数据区使用 goto 语句跳转到 0:4H 观察 1H 中断处理程序入口地址为 0070H:[0008H]

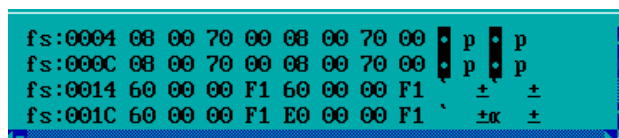


图 3.1.1 1H 调用入口地址

接着使用 goto 语句跳转到 0:40H 观察 10H 中断处理程序入口地址为 F100H:[0300H]

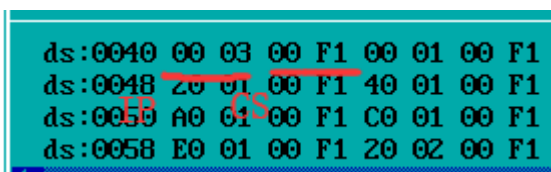


图 3.1.2 10H 调用入口地址

2.使用 35 号调用察看 01H 中断处理程序入口地址为 07FAH:[0B1AH]

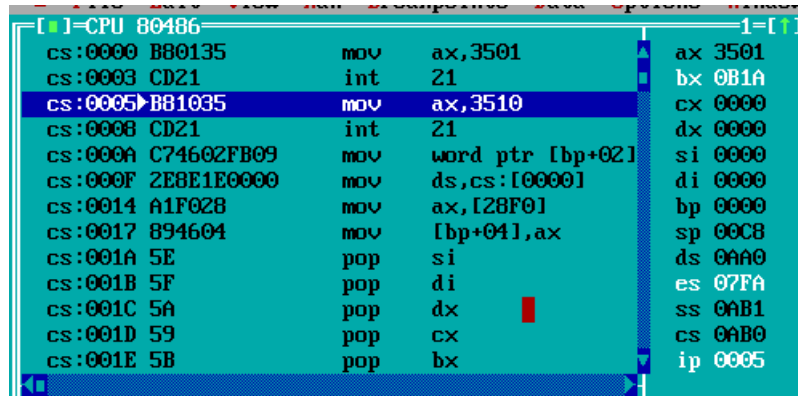


图 3.1.3 35 号调用获取 1H 入口地址

10H 中断处理程序入口地址为 F100H:[0300H]

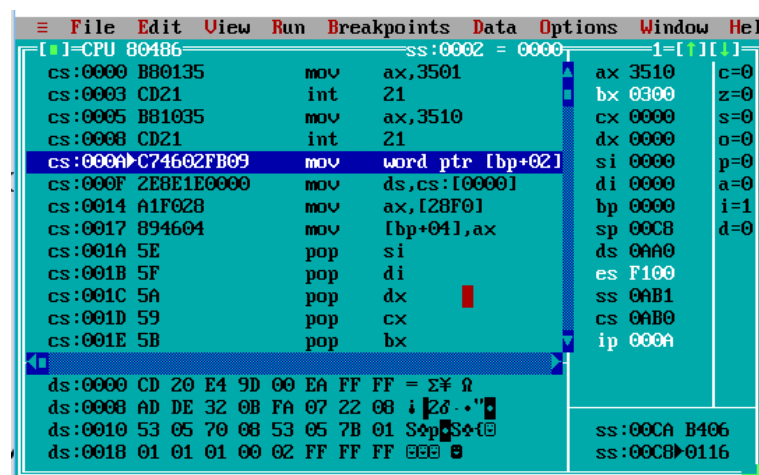


图 3.1.4 35 号调用获取 1H 入口地址

3.直接读取中断处理程序的入口地址，其中 AX 为中断处理程序的 IP 值，BX 为中断处理程序的 CS 值，得到 1H 中断处理程序的入口地址为 07FAH:[0B1AH],10H 中断处理程序的入口地址为 F100H:[0300H]



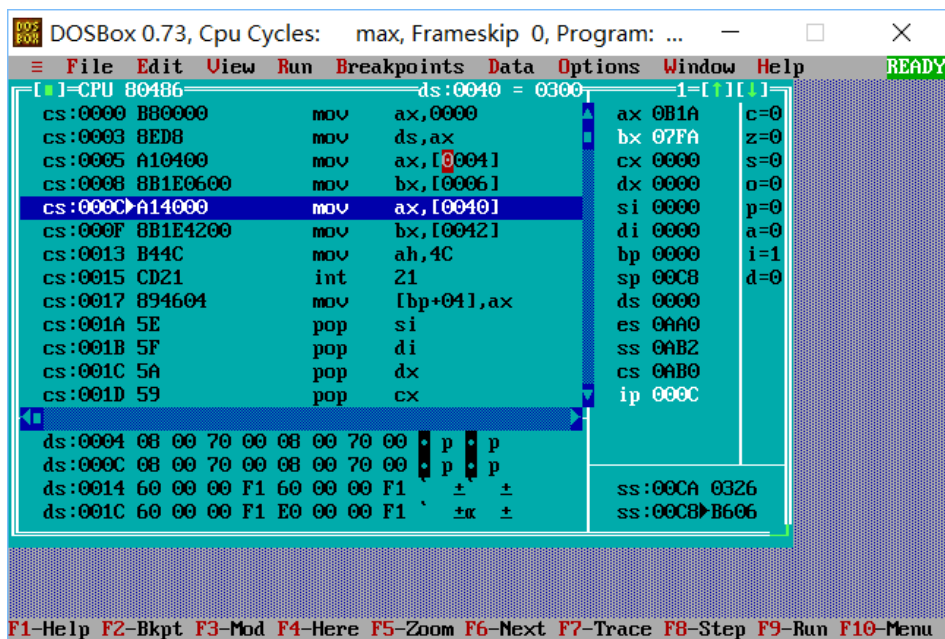


图 3.1.5 直接读取 1H 调用入口地址

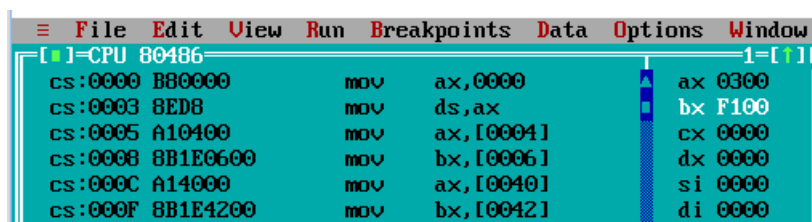


图 3.1.6 直接读取 10H 调用入口地址

通过比对发现，1H 中断处理程序入口地址在从数据区移动至寄存器过程中发生了改变，而 10H 中断处理程序则没有改变。

### 3.1.5 思考题

1. 打开 TD 之后，如何在数据区切换到中断向量表所在内存区域

在数据区使用 goto 语句跳转到 0:0，此即为中断向量表的起始地址，每 4 个字节为一个中断处理程序的入口地址，ip 存放在低 8 位，cs 存放在高 8 位。

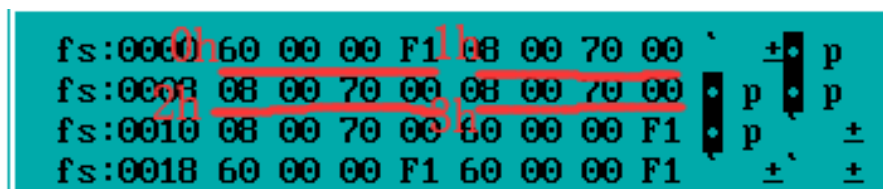


图 3.1.7 中断向量表内存区

2. 如何计算某个中断入口在中断向量表内的偏移地址？

实方式下，当中断号为  $nH$  时，其中断矢量表的偏移地址为  $n*4$ 。

3. 程序中如何使用系统功能调用获取中断入口地址？可以在 TD 中录入指令语句或编写完整程序来尝试。

见实验记录与分析 2

4. 程序中如何通过直接内存读取获取中断入口地址？可以在 TD 中录入指令语句或编写完整程序来尝试。

见实验记录与分析 3

5. 用 TD 把中断矢量表里的中断矢量的值随意改成其他值会有什么现象发生？（比如修改 21H, 1H, 3H 的中断矢量）

修改后运行中断时会死机。

## 3.2 任务 2

### 3.2.1 设计思想及存储单元分配

先判断 16H 调用的入口参数 AH 是否为 00H 和 10H 功能, 若不是则 JMP 到原中断处理程序, 结束。若是则 CALL 原中断处理程序获取输入字符, 然后返回新中断处理程序处理大小写, 结束。卸载程序: 先从中断矢量表中获取新中段处理程序的 CS:IP, 读取新中段处理程序 OLD\_INT 变量中的值, 将中断矢量表中 16H 中断处理程序的入口地址修改为原中断处理程序的入口地址即可。

新中断处理程序存储单元分配: OLD\_INT, DW, 共 4 个字节, 存储旧中断处理程序的入口地址

### 3.2.2 流程图

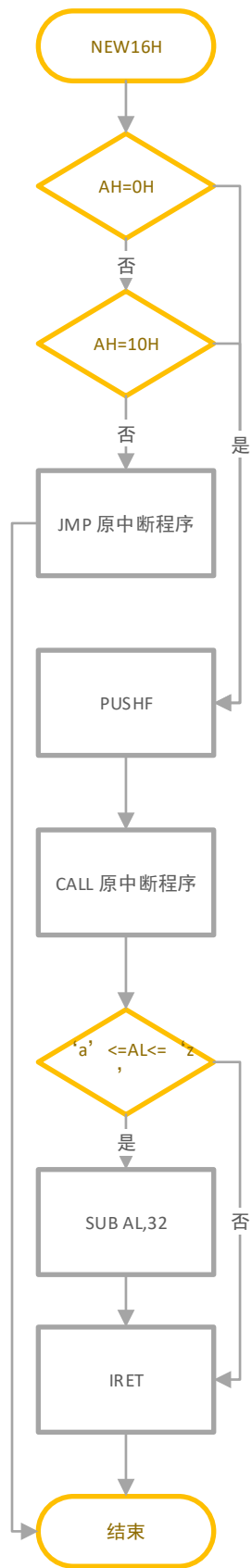


图 3.2.1 新中断处理程序

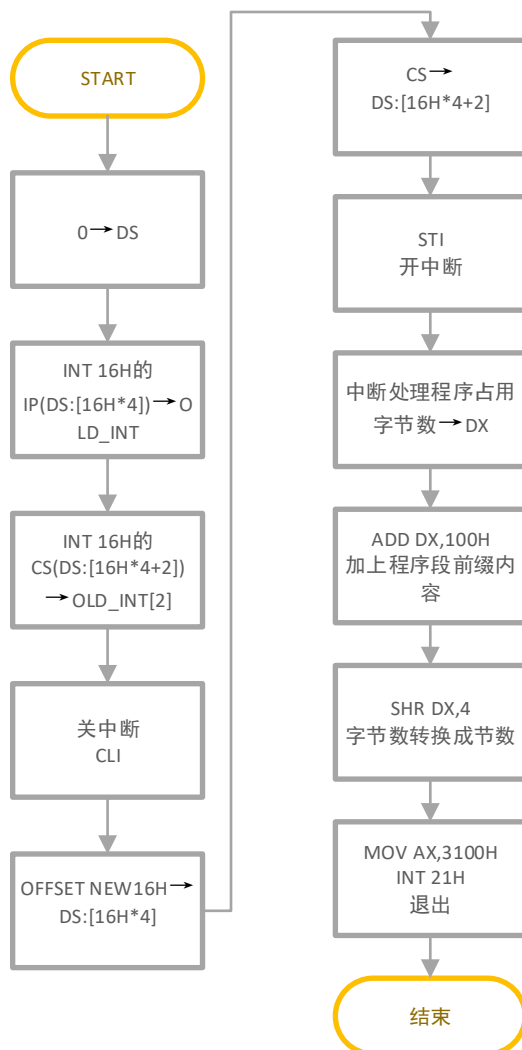


图 3.2.2 新中断处理程序安装程序

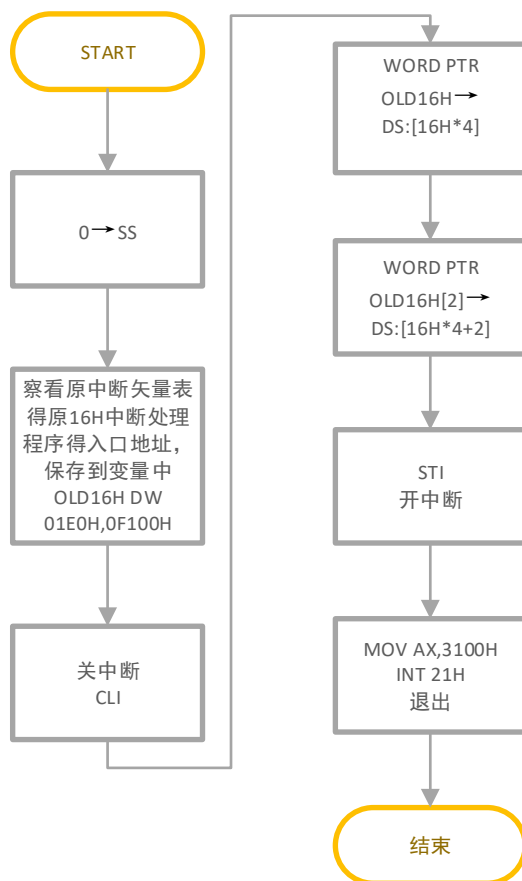


图 3.2.3 新中断处理程序卸载程序

### 3.2.3 源代码

#### 3.2.3.1 新中断处理程序

```

.386
CODE SEGMENT USE16
    ASSUME CS:CODE,SS:STACK
    OLD_INT DW ?,?           ;新程序中使用的变量，用于存放旧中断矢量
    KEY DW 0F1F1H
    NEW16H:
        CMP AH,0             ;判断是否为读键盘
        JE TO_UPPER
        CMP AH,10H           ;判断是否为读拓展键盘
        JE TO_UPPER
        JMP DWORD PTR OLD_INT ;继续原中断处理程序功能
    TO_UPPER:
        PUSHF
        CALL DWORD PTR OLD_INT
        CMP AL,'a'           ;ASCII 码大于等于'a'，转 TRANS，进一步判断 ASCII 码是否小于'z'
        JGE TRANS
        JMP QUIT
    TRANS:
        CMP AL,'z'           ;ASCII 码小于等于'z'
        JG QUIT
        SUB AL,32             ;小写转换为大写

```

```

JMP QUIT
QUIT:
IRET

START:
XOR AX,AX
MOV DS,AX                ;DS=0
MOV AX,DS:[16H*4]        ;取原 16H 的中断矢量的偏移部分并保存
MOV OLD_INT,AX
MOV AX,DS:[16H*4+2]      ;取原 16H 的中断矢量的段值并保存
MOV OLD_INT+2,AX

CLI                      ;关中断
MOV WORD PTR DS:[16H*4],OFFSET NEW16H
MOV DS:[16H*4+2],CS
STI                      ;开中断

MOV DX,OFFSET START+15
SHR DX,4                 ;字节数换成节数
ADD DX,10H               ;程序段前缀内容

MOV AL,0
MOV AH,31H
INT 21H
CODE ENDS
STACK SEGMENT STACK USE16
DB 200 DUP(0)
STACK ENDS
END START

```

### 3.2.3.2 卸载程序

```

.386

CODE SEGMENT USE16
ASSUME CS:CODE
START:
XOR AX,AX
MOV DS,AX                ;DS=0
MOV SS,AX                ;SS=0

MOV SS,WORD PTR DS:[16H*4+2];新中断处理程序的 cs
MOV AX,WORD PTR SS:[4]
CMP AX,0F1F1H
JNE EXIT
MOV AX,WORD PTR SS:[0]    ;获取旧中断处理程序的 ip
MOV BX,WORD PTR SS:[2]    ;获取旧中断处理程序的 cs

CLI
MOV DS:[16H*4],AX         ;恢复旧中断的 ip
MOV DS:[16H*4+2],BX       ;恢复旧中断的 cs
STI

EXIT:
MOV AL,0
MOV AH,31H
INT 21H
CODE ENDS

```



END START

### 3.2.4 实验步骤

1. 准备上机环境，编辑、汇编、连接文件 T3。
2. 运行 T3.EXE，在 DOS 窗口中输入 abc，观察是否转变为大写
3. 在 DOS 窗口中输入 ABCZ 及\*&+-等字符观察是否为正常输入
4. 继续在 DOS 窗口中输入 TD，在代码区输入指令“mov AX,0”，看是否都变成了大写
5. 编辑、汇编、连接卸载程序文件 T3\_U
6. 运行 T3\_U.EXE，重复步骤 2、3，观察小写字符、大写字符、特殊字符是否正常输入。
7. 多次驻留程序，观察现象
8. 重新打开一个 DOS 窗口，观察大小写是否改变

### 3.2.5 实验记录与分析

实验环境条件：WINDOWS10 下 DOSBox0.72； MASM.EXE 6.0； LINK.EXE 5.2； TD.EXE 5.0。

- 1.运行 T3.EXE 后，输入小写字符会变成大写字符，输入其他字符正常

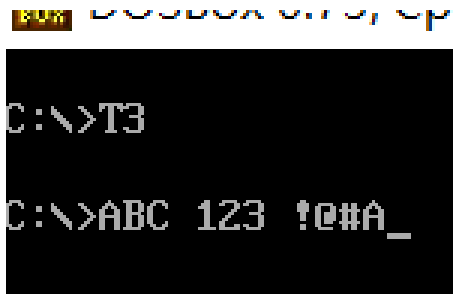


图 3.2.4 输入演示

- 2.运行 TD，在代码区键入“mov AX,0”，发现小写并没有转换成大写，可知新中断处理程序的影响范围有限。

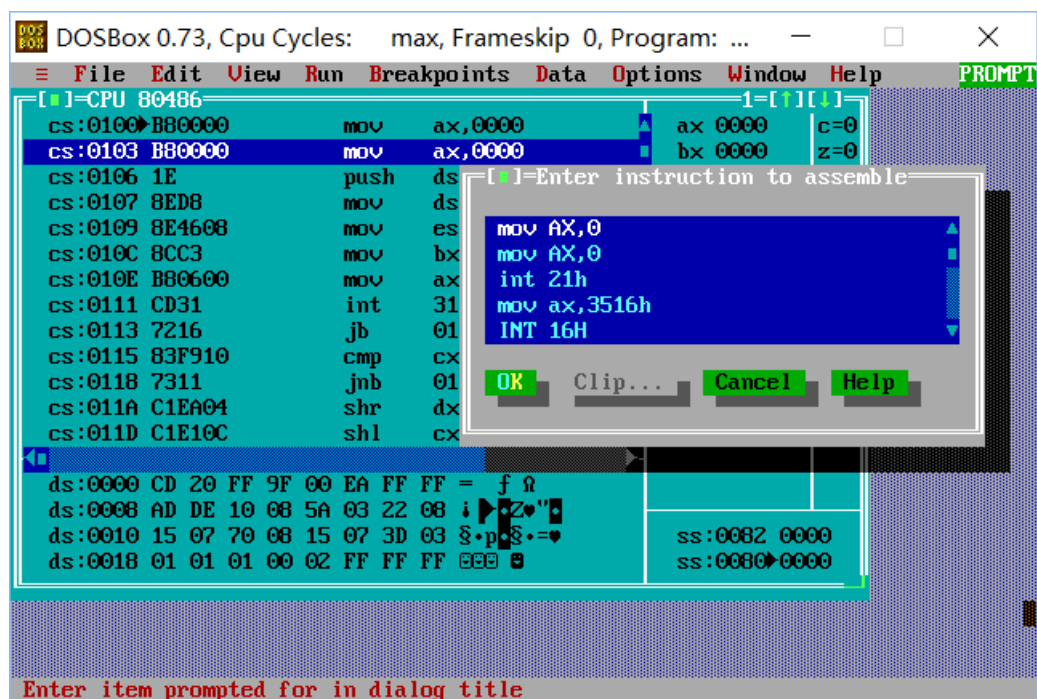


图 3.2.5 TD 中输入演示

3.运行 T3\_U.EXE，再次输入大小写字母，发现小写字母没有变成大写

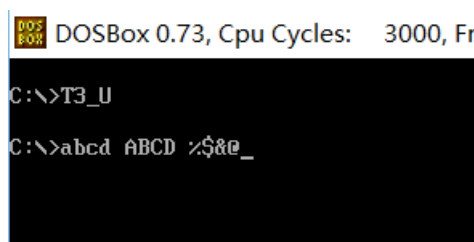


图 3.2.6 输入演示

### 3.2.6 思考题

1.有哪两种方式进入原中断服务程序？

CALL 和 JMP，使用 JMP 跳转执行完中断后会直接返回主程序，使用 CALL 跳转执行完中断后会跳转回新的中断处理程序，不要要记得在 CALL 前加上 PUSHF。

2. 编写的中断驻留程序执行后能否正常返回到 DOS？DOS 是否还能正常工作？如果重复驻留多次，会有什么现象？

多次调试后可以正常运行，若驻留多次会暂用大量内存。

3.同时打开另外一个虚拟 DOS 窗口，键盘大小写是否被替代？

可以发现新的 dos 窗口中大小写未被替代。

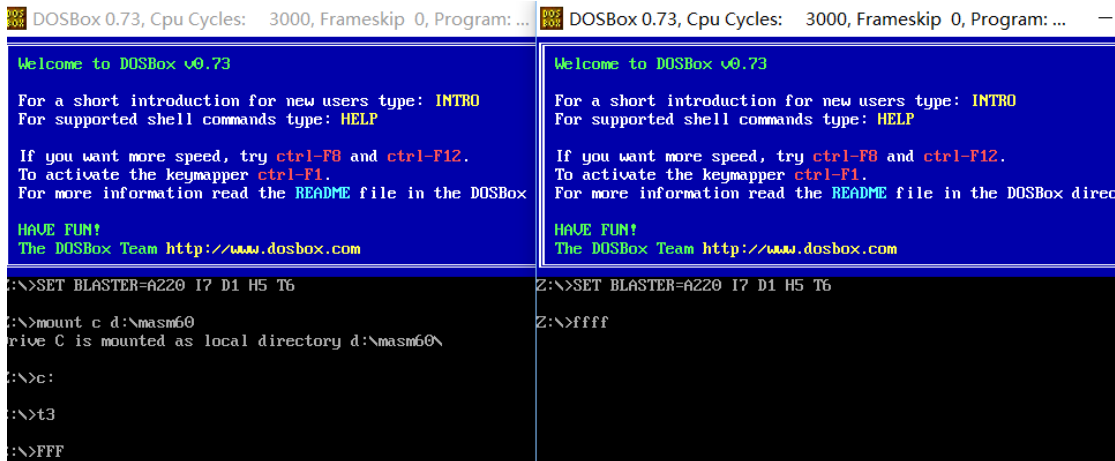


图 3.2.7 新 DOS 窗口输入演示

4. 如何确定自己编写的中断处理程序已被系统调用？（除了小写字母已经被替换成大写的途径之外）

在执行完新的中断处理程序后使用 `td` 运行一次 16H 的中断，在 INT 16H 处使用 `alt+f7` 进入中断，发现是自己写的新中断处理程序，可知已被系统调用。

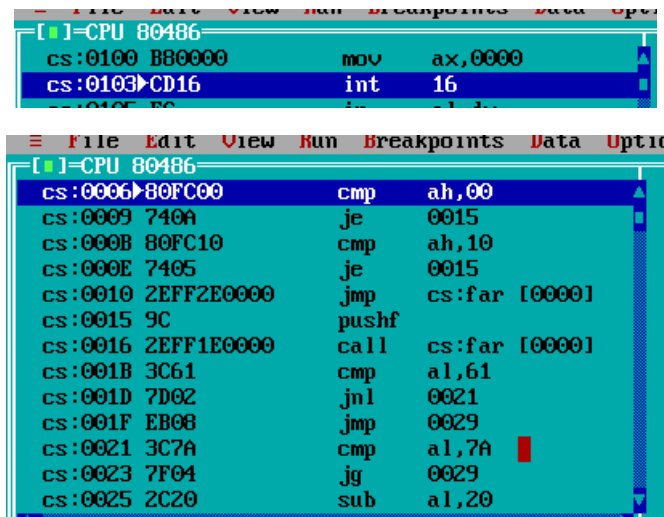


图 3.2.8 16H 功能调用

5. 选作的要求(3)应该如何实现，如何找到保存的原中断入口地址？ 如何保证不会错误恢复？（比如，你的程序还没有驻留，但运行了恢复程序）

通过中断矢量表进入新的中断处理程序，在新程序中 `INT_OLD` 变量中找到旧中断处理程序，然后将其覆盖到中断矢量表中的对应位置上。可以通过在新中断处理程序中声明一个变量 `KEY` 并对其赋特殊值，卸载程序通过访问该值来判断是否是新中断处理程序，若是则卸载，否则退出卸载程序。

### 3.3 任务 3

#### 3.3.1 设计思想及存储单元分配

使用变量存储访问地址，先用 OUT 指令将需要访问的地址送到 70H 中，然后使用 IN 指令从 71H 中读取访问地址的数据，读取的数据为 BCD 码，因此先将其转换为 10 进制数，然后使用 PRINTFAX 以 16 进制数输出。

#### 3.3.2 流程图

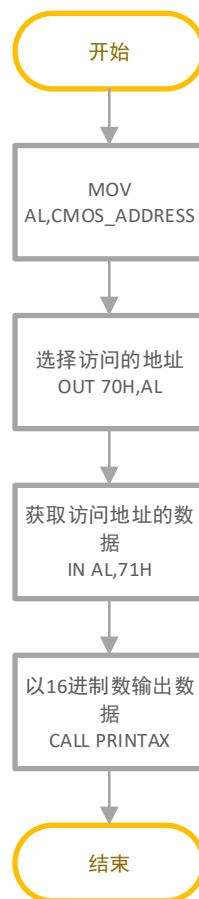


图 3.3.1 程序流程图

#### 3.3.3 源代码

```
INCLUDE MACRO.LIB
.386
DATA SEGMENT USE16
CMOS_ADDRESS DB ?
```

```

MSG1 DB 'input address',0ah,0dh,'$'
DATA ENDS
STACK SEGMENT USE16 STACK
    DB 200 DUP(0)
STACK ENDS
CODE SEGMENT USE16
    ASSUME CS:CODE,DS:DATA,SS:STACK
START:
    MOV AX,DATA
    MOV DS,AX

```

```

b:    WRITE MSG1
    CALL INPUT
    MOV CMOS_ADDRESS,BL
    CRLF
    MOV AL,CMOS_ADDRESS
    OUT 70H,AL
    IN AL,71H
    CBW
    PUSH AX
    CWDE
    CALL PRINTAX
    CRLF
    POP AX
    MOV BX,0
    MOV BL,AL
    SAR BL,4
    AND BL,0FH
    IMUL BX,10
    AND AL,0FH
    ADD BL,AL
    MOVSX EAX,BL
    CALL PRINTAX
    CRLF

```

```

    JMP B
EXIT:
    MOV AL,0
    MOV AH,31H
    INT 21H

```

;------以 X 进制输出有符号整数

;EAX——需要输出的数

PRINTAX PROC

PUSH EBX

PUSH CX

PUSH EDX

;保护现场

MOV EBX,16

XOR CX,CX

;计数器清 0

CMP EAX,0

JNL \_LOP1

NOT EAX

ADD EAX,1

PUSH EAX

OUT 1 ','

;输出负号

POP EAX

\_LOP1:

; (EAX)除以 P，所得商->EAX，余数入栈，CX++，记录余

数个数

XOR EDX,EDX

DIV EBX

PUSH DX

INC CX

```

    OR EAX,EAX
    JNZ _LOP1
    _LOP2:                                ;从栈中弹出一位 P 进制数，并将该数转换成 ASCII 码后输
出
    POP AX
    CMP AL,10
    JB _L1
    ADD AL,7
    _L1:                                ;输出 P 进制数
    ADD AL,30H
    MOV DL,AL
    MOV AH,2
    INT 21H
    LOOP _LOP2
    POP EDX
    POP CX
    POP EBX                            ;恢复现场
    RET
PRINTAX ENDP
INPUT PROC
    MOV AH,01H
    INT 21H
    CMP AL,'A'
    JGE J3
    CMP AL,'0'
    JGE J1

J1:
    CMP AL,'9'
    JLE J2
J2:
    SUB AL,30H
    JMP K5
J3:
    CMP AL,'E'
    JLE J4
J4:
    SUB AL,37H

K5:  MOV BL,AL
    SAL BL,4

    MOV AH,01H
    INT 21H

    CMP AL,'A'
    JGE K3
    CMP AL,'0'
    JGE K1

K1:
    CMP AL,'9'
    JLE K2
K2:
    SUB AL,30H
    JMP K6
K3:
    CMP AL,'E'
    JLE K4
K4:
    SUB AL,37H

K6:  OR BL,AL

```

```
RET
INPUT ENDP
CODE ENDS
END START
```

### 3.3.4 实验步骤

- 1.准备上机环境，编辑、汇编、连接文件 T4。
- 2.运行 T4.EXE，选择访问的地址，选择访问时间地址
- 3.将 DOS 窗口中输出的日期与实际日期比对

### 3.3.5 实验记录与分析

实验环境条件：WINDOWS10 下 DOSBox0.72； MASM.EXE 6.0； LINK.EXE 5.2； TD.EXE 5.0

运行 T4.EXE,观察数据结果，第一行为访问的数据地址（16 进制数），第二行是访问数据的 10 进制数，第三行是访问数据的 16 进制数输出，输出时间为 18 年 4 月 30 日 22 点 29 分 28 秒（10 进制表示），与 windows 系统时间对比，发现一致

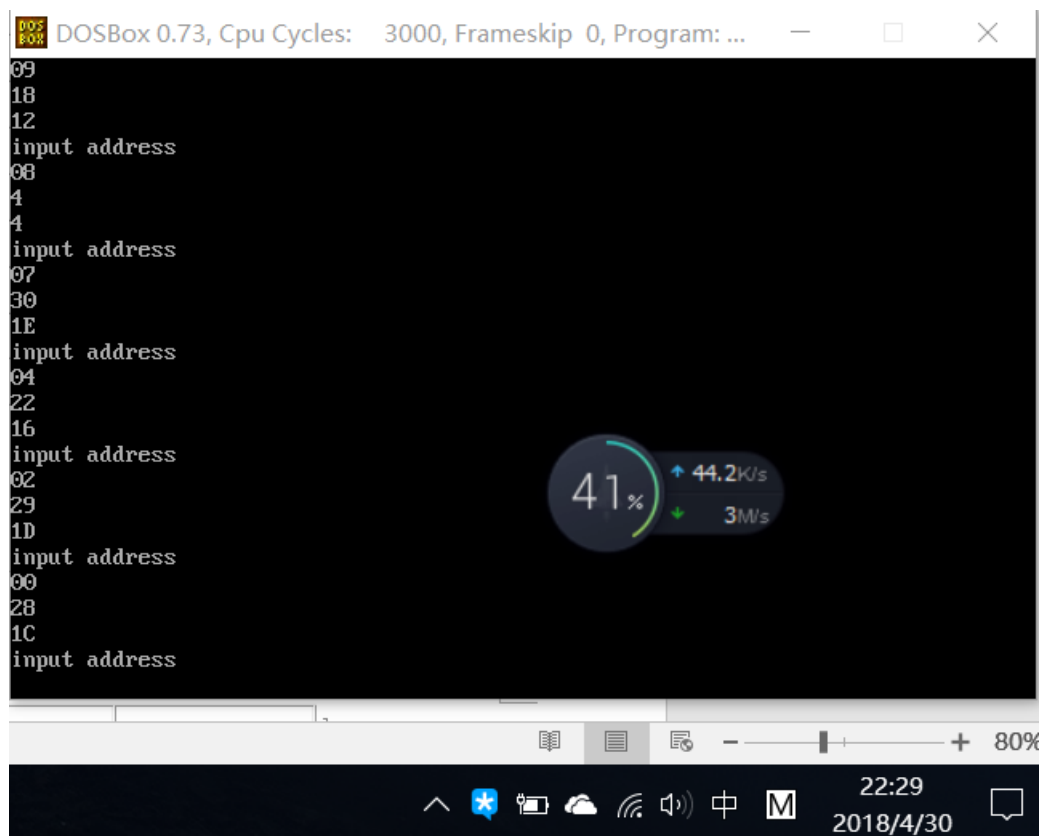


图 3.3.2 输出时间

### 3.3.6 思考题

CMOS 里的时间信息是按照压缩 BCD 码的形式存放的，举例说明压缩 BCD 码的格式是什么？

使用的是 8421 码



## 3.4 任务 4

### 3.4.1 设计思想及存储单元分配

设计思想：使用异或运算对用户名和商品进货价进行加密，再次异或 KEY 可以解密，异或运算的密钥 KEY 由密码计算得到。使用函数  $(X-20H)*3$  对登录密码进行加密。

使用了 3 中在程序开头修改 1H 和 3H 的中断矢量处理程序来抵制动态调试跟踪，在进入功能时用检查中断矢量表是否被正确修改来判断是否被跟踪。

间接转移抵制静态反汇编：各个菜单功能调用使用间接转移指令访问,并在程序中掺入干扰语句。

新增存储单元：

E, DW, 退出程序地址；

GN3, DW, 功能 3 的地址

MFUN1, DW, 菜单功能 1 地址

MFUN2, DW, 菜单功能 2 地址

MFUN3, DW, 菜单功能 3 地址

MFUN4, DW, 菜单功能 4 地址

MFUN5, DW, 菜单功能 5 地址

OLDINT1, DW, 1 号中断的原中断矢量（用于中断矢量表反跟踪）

OLDINT3, DW, 3 号中断的原中断矢量

3.4.2 流程图

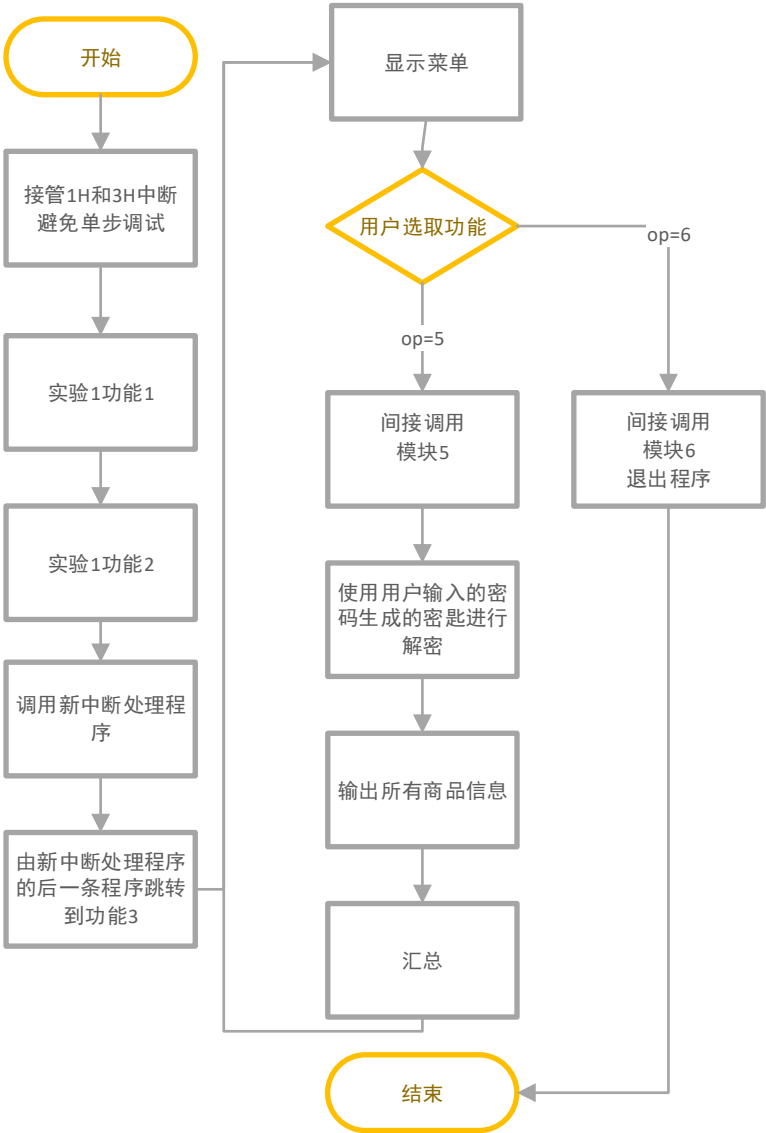


图 3.4.1 程序主体流程图

3.4.3 源代码

本次实验游客登录只有功能 6（退出程序），老板界面由功能 5（输出所有商品信息）、功能 6

3.4.3.1 TASM

```
NAME WAN1
EXTRN PRINT_IFO:NEAR
PUBLIC PRINTAX,PrintASCII,S1,S2,GA1,GB1,N,NEWKEY
.386
```

```

INCLUDE MACRO.LIB
STACK SEGMENT USE16 PARA STACK 'STACK'
DB 200 DUP(0)
STACK ENDS
DATA SEGMENT USE16 PARA PUBLIC 'D1'
BNAME DB 'L' XOR KEY          ;真实姓名为 LVPENGZE。采用函数 X XOR 57H 对用户名进行
加密
    DB 'V' XOR KEY,'P' XOR KEY,'E' XOR KEY,'N' XOR KEY,'G' XOR KEY,'Z' XOR KEY,'E' XOR KEY
    DB ('#'-20H)*3             ;用户名结束符
    DB 34H                     ;随机字符

BPASS DB ('B' -20H)*3         ;真实密码为 BOS。采用函数(X-20H)*3 对保存的密码进行编码。
    DB ('O' -20H)*3
    DB ('S' -20H)*3
    DB ('#'-20H)*3             ;密码结束符
    DB 34H,57H                 ;随机字符
AUTH DB ?                     ;标记登陆状态
N EQU 3
KEY EQU ('B'-'O'+'S') ;加密密钥与密码相关
NEWKEY DB ?                   ;输入密码对应的密钥
S1 DB 'SHOP1','$'             ;网店 1
GA1 DB 'PEN',7 DUP(0)
    DW 5 XOR KEY,56,70,50,? ;进货价、销售价、进货总数、已售数量、利润率
GA2 DB 'BOOK',6 DUP(0)
    DW 23 XOR KEY,30,25,10,?
GA3 DB 'BAG',7 DUP(0)
    DW 15 XOR KEY,30,30,0,?

S2 DB 'SHOP2','$'             ;网店 2
GB1 DB 'PEN',7 DUP(0)
    DW 4 XOR KEY,56,70,50,? ;进货价、销售价、进货总数、已售数量、利润率
GB2 DB 'BOOK',6 DUP(0)
    DW 32 XOR KEY,30,25,10,?
GB3 DB 'BAG',7 DUP(0)
    DW 13 XOR KEY,30,30,0,?

MSG1 DB 0AH,0DH,'Input your account:',0DH,0AH,'$'
MSG2 DB 'Input your password:',0DH,0AH,'$'
MSG3 DB 'WRONG ACCOUNT',0DH,0AH,'$'
MSG4 DB 'Enter the name of the item:',0DH,0AH,'$'
MSG5 DB 0AH,0DH,'Input Your Option',0DH,0AH,'$'

MENU5 DB '5.Export all product information','$'
MENU6 DB '6.Program exit','$'

COUNT DB 3

Product_name DB 'Product name', '$'
Purchase_price DB 'Purchase price', '$'
Selling_price DB 'Selling price', '$'
Total_purchases DB 'Total purchases', '$'
Sold_quantity DB 'Sold Quantity', '$'
Profit_rate DB 'Profit rate', '$'
Ranking DB 'Ranking', '$'

SelectShop DB '1.SHOP1 2.SHOP2','$'
in_name DB 11                 ;姓名缓冲区
    DB ?
    DB 11 DUP(0)
in_pwd DB 7                   ;密码缓冲区
    DB ?
    DB 7 DUP(0)

```

```

in_goods DB 11                                ;商品名缓冲区
          DB ?
          DB 11 DUP(0)

E         DW EXIT                               ;地址表（用于间接转移反跟踪）,E 为退出程序
GN3       DW GONG_NENG3                         ;功能 3 的地址
MFUN5     DW PRINT_IFO                         ;菜单功能 5 地址


OLDINT1 DW 0,0                                ;1 号中断的原中断矢量（用于中断矢量表反跟踪）
OLDINT3 DW 0,0                                ;3 号中断的原中断矢量


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


GONG_NENG1:                                ;-----功能 1，提示
登陆
CALL ClearS
WRITE MSG1
READ in_name
CRLF
R1:  cli                                    ;设置新的中断矢量,混在程序中，使用 jmp 语句执行
    jmp R2
    db 'step1'

R9:
WRITE MSG2
READ in_pwd
CRLF

MOV AL,in_pwd[2]
ADD AL,in_pwd[4]
SUB AL,in_pwd[3]                            ;输入密码转换成的 key，若输入正确则解密正确，否则解密错误
MOV NEWKEY,AL

CMP AUTH,2
JNE EXIT
MOV BL,in_name[2]
CMP BL,'q'
JE Y1
JMP Y2
Y1:
JMP WORD PTR E                            ;输入'q'，退出程序
    db 'login success'                    ;干扰

Y2:  MOV BL,in_name[1]

```

```

CMP BL,0
JE UNLOGIN                ;输入回车,转未登录
JMP GONG_NENG2            ;否则转功能 2
    db 'maybe next time' ;干扰

UNLOGIN:
    lea bx,AUTH
    mov byte ptr DS:[BX],0 ;标记登陆状态
    jmp word ptr GN3        ;未登录状态, 转功能 3

GONG_NENG2:                ;-----功能 2, 判断登陆
    XOR ECX,ECX            ;循环比较姓名
    MOV CL,in_name[1]
    MOV ESI,0
LOOP1:
    MOV BL,in_name[ESI+2]
    XOR BL,NEWKEY
    CMP BL,BNAME[ESI]
    JNE ACCOUNTWRONG        ;姓名不匹配转 ACCOUNTWRONG
    INC ESI
    DEC ECX
    JNZ LOOP1

    MOV BX,'#'
    SUB BX,20H
    IMUL BX,3
    CMP BL,BNAME[ESI]
    JNZ ACCOUNTWRONG        ;输入的字符串是定义字符串的子集

    XOR ECX,ECX            ;循环比较密码
    MOV CL,in_pwd[1]
    MOV ESI,0
LOOP2:
    MOVSX BX,in_pwd[ESI+2]
    SUB BX,20H
    IMUL BX,3
    CMP BL,BPASS[ESI]
    JNE ACCOUNTWRONG        ;密码不匹配转 ACCOUNTWRONG
    INC ESI
    DEC ECX
    JNZ LOOP2

    ;XOR BX,BX
    MOV BX,'#'
    SUB BX,20H
    IMUL BX,3
    CMP BL,BPASS[ESI]

JNZ ACCOUNTWRONG
JMP SUCCESS                ;账号密码都相同转登陆成功

ACCOUNTWRONG:                ;登陆失败
    DEC COUNT
    JZ UNLOGIN
    WRITE MSG3
    MOV AH,01H
    INT 21H
    JMP GONG_NENG1

```

```

SUCCESS:                                ;登陆成功
    LEA BX,AUTH
    MOV BYTE PTR DS:[BX],1              ;间接赋值，干扰修改登录状态
    jmp word ptr GN3[0]                 ;已登录状态，转功能 3
    DB 'GOOD JOB'                       ;干扰
GONG_NENG3:                             ;-----功能 3
    mov bx,es:[1*4]                     ;检查中断矢量表是否被调试工具阻止修改或恢复
    inc bx
    jmp bx                               ;正常修改了的化，这里将转移到 TESTINT，否则就不知道转到哪了
Y3:  CALL ClearS
    CALL DisplayMenu
    CALL SelectFun

    CMP AL,'5'
    JE _FUN5
    CMP AL,'6'
    JE _FUN6
_FUN5:
    CALL WORD PTR MFUN5
    JMP _FUN_NEXT
_FUN6:
    JMP EXIT

_FUN_NEXT:
    MOV AH,01H
    INT 21H
    JMP WORD PTR GN3

EXIT:
    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

    MOV AH,4CH
    INT 21H                            ;退出程序

;-----
;以 10 进制输出有符号整数
;EAX——需要输出的数
PRINTAX PROC
    PUSH EBX
    PUSH CX
    PUSH EDX                            ;保护现场
    MOV EBX,10
    XOR CX,CX                            ;计数器清 0
    CMP EAX,0
    JNL _LOP1
    NOT EAX
    ADD EAX,1
    PUSH EAX
    OUT1 ','                             ;输出负号
    POP EAX
_LOP1:                                ;(EAX)除以 P，所得商->EAX，余数入栈，CX++，记录余
数个数

```

```

XOR EDX,EDX
DIV EBX
PUSH DX
INC CX
OR EAX,EAX
JNZ _LOP1
_LOP2:                                     ;从栈中弹出一位 P 进制数，并将该数转换成 ASCII 码后输
出
POP AX
CMP AL,10
JB _L1
ADD AL,7
_L1:                                     ;输出 P 进制数
ADD AL,30H
MOV DL,AL
MOV AH,2
INT 21H
LOOP _LOP2
POP EDX
POP CX
POP EBX                                   ;恢复现场
RET
PRINTAX ENDP

;-----
;重复输出字符
;AL——需要重复输出的字符
;AH——需要重复输出的字符数
PrintASCII PROC
PUSH ECX
MOVSX ECX,AH
_PUTCHAR_:
OUT1 AL
LOOP _PUTCHAR_
POP ECX
RET
PrintASCII ENDP
R2:    mov  ax,OFFSET NEWINT
        jmp R3
        db 'step2'
R3:    mov  es:[1*4],ax
        jmp R4
        db 'step3'
R4:    mov  es:[1*4+2],cs
        jmp R5
        db 'step4'
R5:    mov  AUTH,2                               ;标记这一段程序被执行
        jmp R6
        db 'step5'
R6:    mov  es:[3*4],ax
        jmp R7
        db 'step6'
R7:    mov  es:[3*4+2],cs
        jmp R8
        db 'step7'
R8:    sti
        jmp R9
        db 'step8'
;模块功能:显示菜单
;模块名称:DisplayMenu
;传入参数:无
;参数传入方式: 无

```

```

;传出参数:无
;参数传出方式: 无
;备注:负责人: 吕鹏泽
DisplayMenu PROC
    PUSH AX

    MOV AL,''
    MOV AH,10
    CALL PrintASCII
    MOV AL,'-'
    MOV AH,60
    CALL PrintASCII
    CRLF

    MOV AH,AUTH
    CMP AH,0
    JE _DM_UNLOGIN_
    JMP _DM_LOGIN_
_DM_LOGIN_:

    MOV AL,''
    MOV AH,20
    CALL PrintASCII
    WRITE MENU5
    CRLF
    MOV AL,''
    MOV AH,20
    CALL PrintASCII
    WRITE MENU6
    CRLF
    JMP _DM_R_
_DM_UNLOGIN_:

    MOV AL,''
    MOV AH,20
    CALL PrintASCII
    WRITE MENU6
    CRLF
    JMP _DM_R_
_DM_R_:
    MOV AL,''
    MOV AH,10
    CALL PrintASCII
    MOV AL,'-'
    MOV AH,60
    CALL PrintASCII
    CRLF
    POP AX
    RET
DisplayMenu ENDP
;-----
;根据登录状态选择功能
;AL——记录并返回选择的功能
SelectFun PROC
    _S_RESTART:
    WRITE MSG5
    MOV AH,01H
    INT 21H
    MOV AH,AUTH
    CMP AH,0
    JE _S_UNLOGIN_
    JMP _S_LOGIN_
_S_LOGIN_:

```



```

CMP AL,'5'
JE _S_R_
CMP AL,'6'
JE _S_R_
JMP _S_RESTART
_S_UNLOGIN_:
CMP AL,'6'
JE _S_R_
JMP _S_RESTART
_S_R_:
PUSH AX
CRLF
POP AX
RET
SelectFun ENDP

```

;-----清屏

```

ClearS PROC
PUSH AX
MOV AH,00H
    MOV AL,03H
    INT 10H
POP AX
RET
ClearS ENDP

```

;-----

;从键盘输入 10 进制数，结果存在 CX 中

;传出参数：CX,输入的 10 进制数,如果输入错误则 CX=-1,如果输入回车 CX=-2

```

INPUT_NEW_INFO PROC
PUSH AX
PUSH BX
MOV CX,0
MOV AH,01H
INT 21H
CMP AL,0DH
JE _INI_0DH ;只输入了回车
JMP _INI_JUDGE
_INI_START:
MOV AH,01H
INT 21H
CMP AL,0DH
JE _INI_R
_INI_JUDGE:
CMP AL,'0'
JL _INI_WRONG ;输入字符不合法
CMP AL,'9'
JG _INI_WRONG ;输入字符不合法
SUB AL,'0' ;转换成数字
MOVSX BX,AL
IMUL CX,10
ADD CX,BX
JMP _INI_START
_INI_WRONG:
MOV CX,-1
JMP _INI_R
_INI_0DH:
MOV CX,-2
JMP _INI_R
_INI_R:
POP BX
POP AX

```

```

RET
INPUT_NEW_INFO ENDP

NEWINT: iret
TESTINT: jmp Y3                ;进入正常功能区
        DB 'WRONG'
CODE ENDS
END START

```

### 3.4.3.2 T1.ASM

见实验 3 任务 1

### 3.4.3.3 MICRO.LIB

见实验 3 任务 1

## 3.4.4 实验步骤

1. 准备上机环境，编辑、汇编、连接文件 T3。
2. 运行 T3.EXE，在 DOS 窗口中输入错误账号密码，观察 3 次后是否进入游客模式
3. 计算利润率等信息，观察是否正确
4. 使用文本编辑器观察并比较明文存放账号密码的 exe 文件

## 3.4.5 实验记录与分析

思考题：

1. 若密码是用明文存放在数据段中的，如何更快地获取密码？

使用文本编辑器观察明文存放的账号密码，结果如图所示，可以很快得到账号密码。



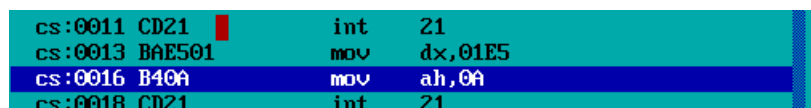
## 3.5 任务 5

### 3.5.1 设计思想及存储单元分配

通过阅读 TD 生成的反汇编指令，利用更改 ip 的值跳过反汇编程序，找到输入的用户名、密码存放的地址以及实际的密文形式的用户名、密码的存放地址并记录。找到密码比较的程序段，观察输入的密码是如何转换成密文的，然后用这一步的逆过程将密文形式的正确密码转换成明文。

### 3.5.2 实验记录与分析

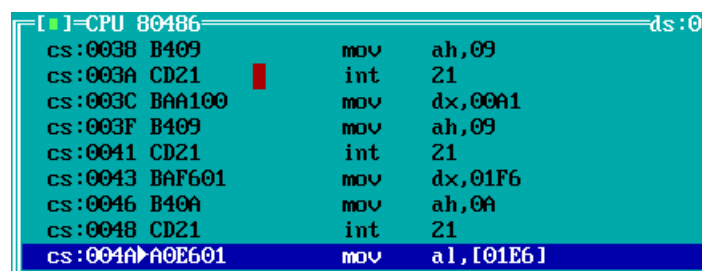
1.使用 TD 调试工具，知道到如图所示步骤时要求输入用户名，可以得到输入用户名存放地址为 ds:01e5h



cs:0011	CD21	int	21
cs:0013	BAE501	mov	dx,01E5
cs:0016	B40A	mov	ah,0A
cs:0018	CD21	int	21

图 3.4.4 获取输入用户名缓冲区

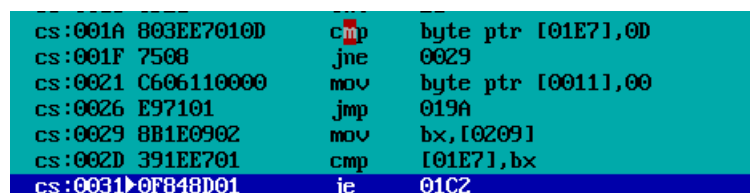
同理，输入密码存放地址为 ds:01f6



cs:0038	B409	mov	ah,09
cs:003A	CD21	int	21
cs:003C	BAA100	mov	dx,00A1
cs:003F	B409	mov	ah,09
cs:0041	CD21	int	21
cs:0043	BAF601	mov	dx,01F6
cs:0046	B40A	mov	ah,0A
cs:0048	CD21	int	21
cs:004A	A0E601	mov	al,[01E6]

图 3.4.5 获取输入密码缓冲区

2.继续观察反汇编程序，可以看到 cs:001a 是判断空格字符，cs:0029 是判断'q'字符，进而更改 ip 的值，观察出 cs:019a 是未登录程序段，cs:01c2 是退出程序段



cs:001A	803EE7010D	cmp	byte ptr [01E7],0D
cs:001F	7508	jne	0029
cs:0021	C606110000	mov	byte ptr [0011],00
cs:0026	E97101	jmp	019A
cs:0029	8B1E0902	mov	bx,[0209]
cs:002D	391EE701	cmp	[01E7],bx
cs:0031	0F848D01	je	01C2

图 3.4.6 登录判断

3. 执行完账号密码输入后进入到用户名的比对 (cs:006c)，可以发现用户名存放在 ds:0 处，内容为 SUNLEI。

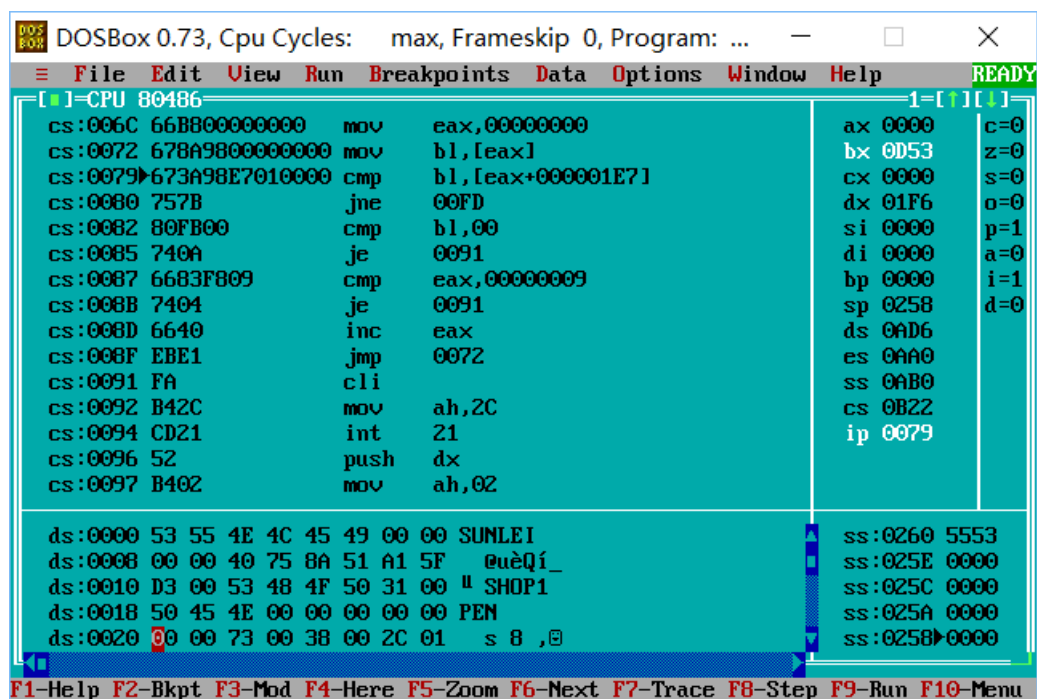


图 3.4.7 获取用户名

4.继续执行,cs:0091 处为用户名比对完毕,然后是一个计时反汇编,通过 Ctrl+N 修改 ip 的值跳过该反汇编程序段

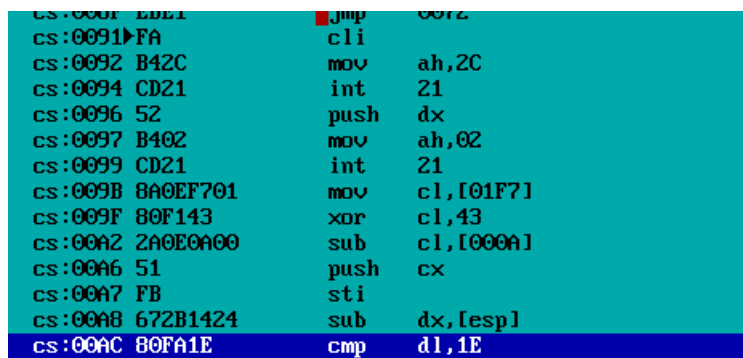


图 3.4.8 跳过计时反汇编

在计时反汇编程序段中红色圈出的部分是输入字符串长度转换成密文的指令,并存放在 cl 中,转换的方式是与 43h 异或。

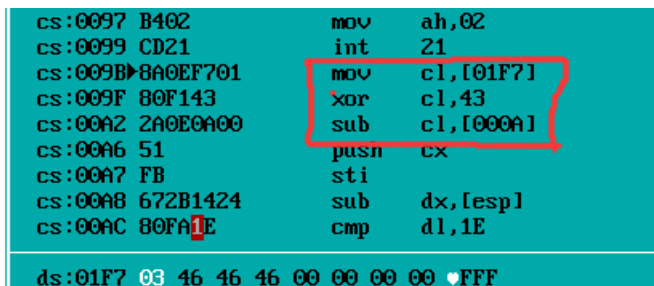


图 3.4.8 获取加密方式

5.密码比较部分为 00e1 至 00f2,其中 jne 0016 为比较错误。密文转换方式为将输入的字符减去 29H 后乘 3 转换成密文,且密文的地址为 000BH

```

cs:00D9 FFE3      jmp     bx
cs:00DB 53        push    bx
cs:00DC 41        inc     cx
cs:00DD 46        inc     si
cs:00DE 204641    and     [bp+41],al
cs:00E1 0FB684F801 movzx   ax,byte ptr [si+01F8]
cs:00E6 83E829    sub     ax,0029
cs:00E9 F6E2      mul     dl
cs:00EB 3A840B00   cmp     al,[si+000B]
cs:00EF 7525      jne     0116
cs:00F1 46        inc     si
cs:00F2 E2ED      loop    00E1

```

图 3.4.9 获取密文密码存放地址

在数据区跳转到密码存放的位置，40H 为密码长度的密文，由 4 知明文为 40H XOR 43H=3,将 75H,8AH,51H 先除 3，再加上 29H 后转换为明文，得密码为 50H,57H,44H 对应的 ASCII 码为 PWD

```

ds:000A 40 75 8A 51 A1 5F D3 00 00 00 00 00 00 00 00
ds:0012 53 48 4F 50 31 00 50 45 SHOP1 PE
ds:001A 4E 00 00 00 00 00 00 00 00 00 00 00 00 00 00
ds:0022 73 00 3B 00 2C 01 64 00 00 00 00 00 00 00 00

```

图 3.4.10 破解密码

#### 6.以 SUNLEI,PWD 登录系统

```

user name:SUNLEI
password:PWD
MENU
1 Seek Goods Message
2 Change Goods Message
3 Calculated average utilization
4 Calculation of profit margin ranking
5 Output all commodity information
6 EXIT
Input 1-6 of the number into the corresponding function:
▲

```

图 3.4.11 成功登录

使用功能 5 输出商品信息获取进货价：

```

SHOP1
Name      InPrice  outPrice  In_num    ut_num    1&2Avr_use_rate
PEN       35 56 300 100 0
BOOK      12 30 25 5 0
BAG       30 50 20 10 0
SHOP2Name InPrice  outPrice  In_num    ut_num    1&2Avr_ranking
BOOK      12 28 20 15 0
PEN       35 50 30 30 0
BAG       30 50 20 10 0

```

图 3.4.12 输出商品信息获取进货价

## 4. 总结与体会

通过第一个四课时的实验，让我对中断处理程序的调用、存储、驻留等有了更清晰的认识。任务 1 用三种不同方式访问中断处理程序的入口地址让我熟练掌握了快速从主存中观察中断矢量表并获取其入口地址，让我看到了汇编编程的灵活性，也让我熟练掌握了计算中断处理程序的入口地址。任务 2 通过编写键盘驱动的中断处理程序，将小写变为大写，我体会到了中断处理程序的巧妙之处。任务 3 让我理解并掌握了 `in` 和 `out` 指令的使用方法以及对 BCD 码有了更深刻的理解。

任务 4 和任务 5 让我懂得了汇编中的跟踪与反跟踪，使我初步体会了软件的破解过程以及反破解的方法，也让我看到了软件加密的重要性。本次破解程序的重要思路就是找到输入的用户名和密码的存放地址，并且要注意利用修改 `ip` 的值来跳过反跟踪程序，然后以其存放地址来确定用户名判断和密码判断的代码区，然后在代码区中找到密文形式的账号和密码的存放地址，接着通过观察输入的密码转换成密文的过程来退出转换过程的反函数，然后将密文密码转换成明文，用户名同理。当然，这种方法比较适合像本次实验这样加密方式简单的加密，对于加密算法复杂的加密，如 RSA 算法，求出其反函数本身是不可能的事，也就谈不上破解密码了。

在破解过程中难免会遇到反跟踪陷阱，通过观察 `cli` 标识和 `sti` 标识可以确定修改中断的反汇编陷阱，使用 `Ctrl+N` 修改 `ip` 的值即可跳过这些陷阱，堆栈保护的陷阱可以观察 `push` 和 `pop` 语句获得其位置，而间接转移可以访问数据区来获得地址。在 C 语言中，可以添加无效语句、加密密码、试错次数限制来达到反跟踪的目的，与汇编语言比较，C 语言反跟踪的方式较单一，反跟踪效果差，要想获得强反跟踪效果，还是要用汇编语言写。

在编写程序时，对于 `cli` 和 `sti` 这样明显的语句难免会暴露出修改中断的反跟踪陷阱，因此可以通过添加无效的 `cli`, `sti` 语句、把必要的执行语句放到 `cli`, `sti` 语句之间，将 `cli`、`sti` 之间的语句使用 `jmp` 连接混入到程序中、使用子程序的方式关中断等方法来混乱破解者。

## 5. 参考文献

[1] 王元珍,曹忠升,韩宗芬.80X6 汇编语言程序设计.版本(第 1 版).武汉.华中科技大学出版社,2005 年 4 月.