

**课 程 实 验 报 告**

**课程名称： 汇编语言程序设计实验**

**实验名称： 实验五** 中断程序设计

**实验时间： 2017-11-21，18：30-21：50 实验地点： 南一楼804室**

**指导教师： 朱虹**

**专业班级：信息安全 201602班**

**学 号： U201614836 姓 名： 林晓斌**

**同组学生： 无 报告日期： 2017年 11 月 21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：2017.10.17

|  |  |  |
| --- | --- | --- |
| 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

[实验五 中断程序设计 3](#_Toc500882206)

[5.1 实验目的和要求 3](#_Toc500882207)

[5.2实验任务 3](#_Toc500882208)

[5.3实验内容 3](#_Toc500882209)

[5.3.1 任务1 3](#_Toc500882210)

[5.3.2 任务2 7](#_Toc500882211)

[5.4实验感想 12](#_Toc500882212)

[5.5 参考文献 12](#_Toc500882213)

# 实验五 中断程序设计

# 5.1 实验目的和要求

(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)提升对计算机系统的理解与分析能力。

# 5.2实验任务

任务1：用四种方式获取中断类型码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中断程序的入口地址；

（3）编写程序，直接读取相应内存单元，获取中断程序的入口地址；

（4）直接跟踪运行一个含有 INT 10H 语句的程序，按Alt+F7进入10H对应的中断处理程序,观察刚进入该程序时的地址。

使用TD观看有关信息即可，比较四种方式得到的中断程序的入口地址各是多少，它们是否相同。

任务2：编写一个接管时钟中断的中断服务程序并驻留内存，要求在屏幕的左上角显示时间。

要求：

（1）对程序中使用的DOS功能调用、BIOS调用都要予以说明。

（2）描述中断发生时，CPU的响应过程，以及中断处理程序运行到IRET时，CPU的处理过程。

# 5.3实验内容

5.3.1 任务1

（1）直接打开td.exe，在数据存储区跳转到0:0040，即在中断矢量表可见到10H中断的中断入口地址。可以看到其入口地址段地址为F000,偏移地址为1300，如图3.1.

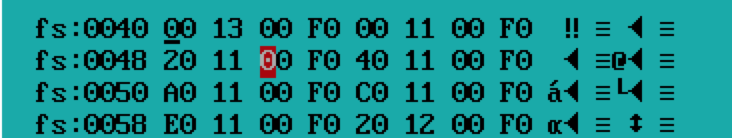


图5.1 10H中断入口地址

（2）编写程序，利用35h号调用（取中断信息，al=中断类型号，ES:[BX]=入口地址）

.386

STACK SEGMENT STACK USE16

DB 200 DUP(0)

STACK ENDS

CODE SEGMENT USE16

ASSUME CS:CODE

START: XOR AX,AX

MOV DS,AX

MOV AX,3510H

INT 21H

MOV AH,4CH

INT 21H

CODE ENDS

END START

程序运行后，可以看到bx显示1300，es显示f000，即10h的入口地址段地址为F000,偏移地址为1300，如图3.2。

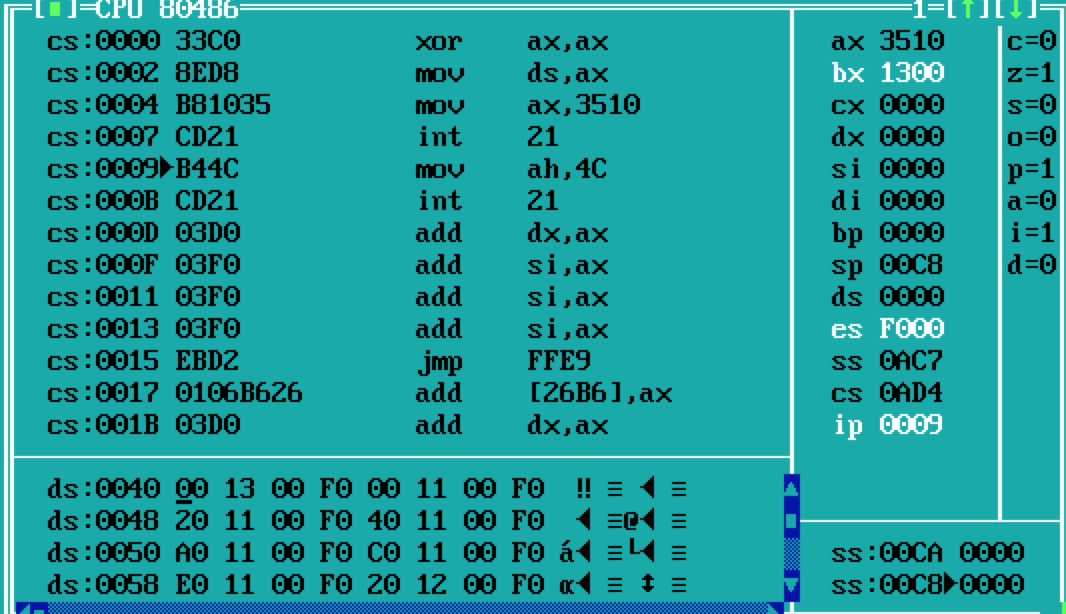


图5.2dos调用观察10h入口地址

（3）编写程序，直接将10h的入口地址放入变量old\_int中，源代码如下：

.386

STACK SEGMENT STACK USE16

DB 200 DUP(0)

STACK ENDS

CODE SEGMENT USE16

ASSUME CS:CODE

START: OLD\_INT DW ?,?

XOR AX,AX

MOV DS,AX

MOV AX,DS:[10H\*4]

MOV OLD\_INT,AX

XOR AX,AX

MOV AX,DS:[10H\*4+2]

MOV OLD\_INT+2,AX

INT 21H

MOV AH,4CH

INT 21H

CODE ENDS

END START

可以观察到在cs：0000段中开辟的变量old\_int为1300 F000，如图5.3，即为10h的入口地址值。

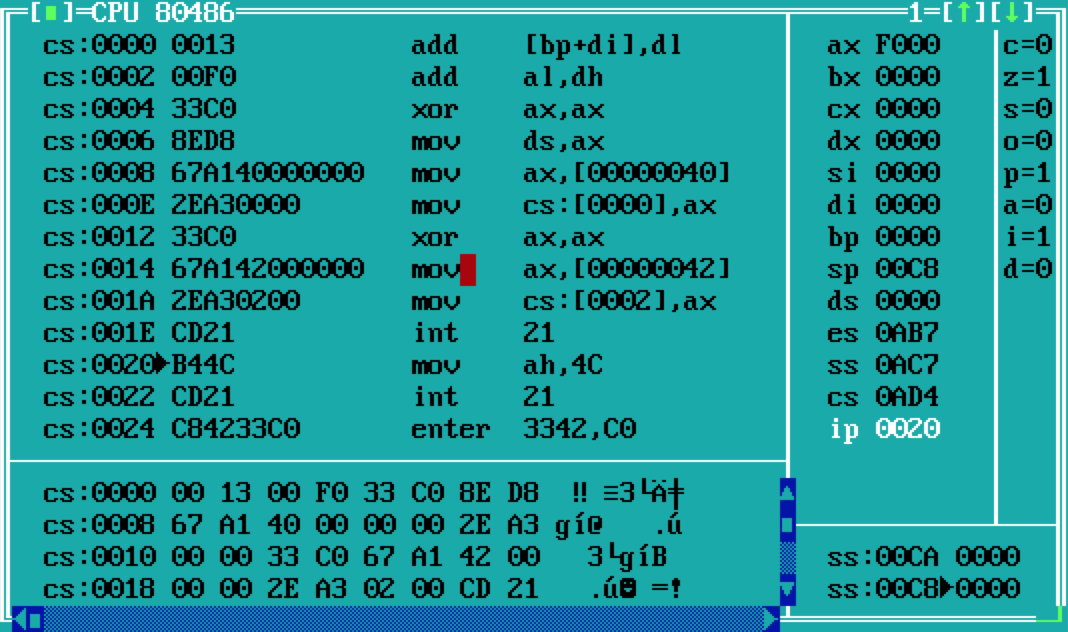


图5.3直接存储10h入口地址

（4）源程序如下：

.386

STACK SEGMENT STACK USE16

DB 200 DUP(0)

STACK ENDS

CODE SEGMENT USE16

ASSUME CS:CODE

START: XOR AX,AX

MOV DS,AX

INT 10H

MOV AH,4CH

INT 21H

CODE ENDS

END START

按下alt+f7后可以观察到其直接跳转到10h的入口地址（段地址f000，偏移地址1300），如图5.4

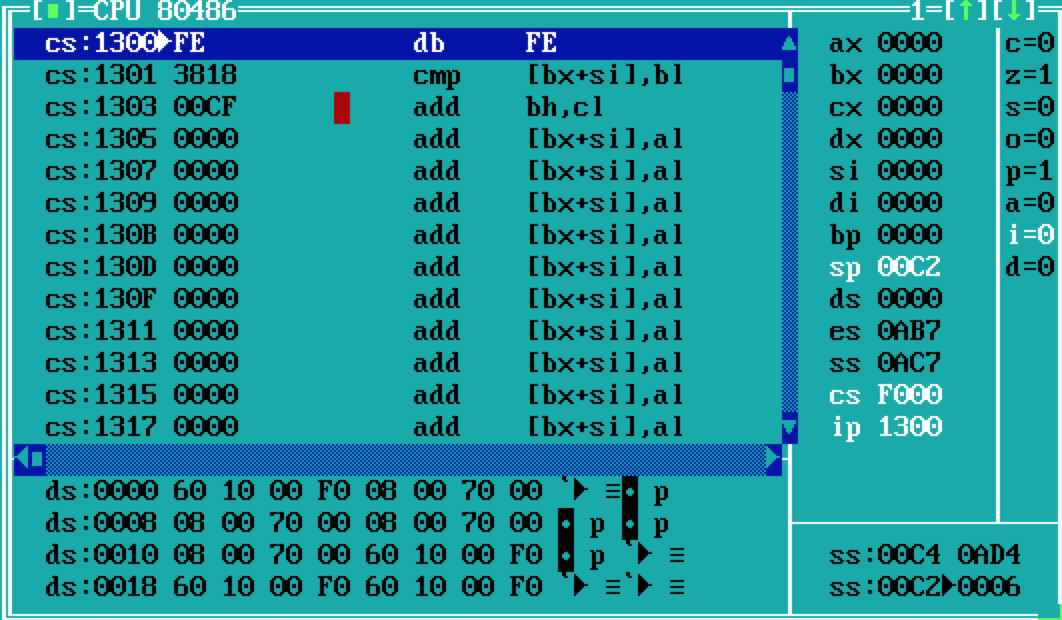


图5.4直接跳转10h入口地址

由此可见上述四种方式取得的10h的入口地址都一致。

5.3.2 任务2

int 08H调用为系统定时器外中断，故将在用新的08h替换原来的08h时加入显示时间的指令即可显示出时间，显示需要大量用到int 10h中的调用，需要仔细研究规则。将新调用替换旧调用后，按下q会将08h调用恢复，按下s则会将其驻留与屏幕左上角。注意计算其需驻留的内存单元数。

5.3.2.1 源代码如下:

.386

STACK SEGMENT USE16 STACK

DB 200 DUP(0)

STACK ENDS

;

CODE SEGMENT USE16

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

;

COUNT DB 18

HOUR DB ?,?,':'

MIN DB ?,?,':'

SEC DB ?,?

BUF\_LEN=$-HOUR

CURSOR DW ?

OLD\_INT DW ?,?

;

NEW08H PROC FAR

PUSHF

CALL DWORD PTR CS:OLD\_INT

DEC CS:COUNT

JZ DISP

IRET

DISP: MOV CS:COUNT,18

STI

PUSHA

PUSH DS

PUSH ES

MOV AX,CS

MOV DS,AX

MOV ES,AX

CALL GET\_TIME

MOV BH,0 ;获取0号显示页面当前光标的位置

MOV AH,3

INT 10H;

MOV CURSOR,DX

MOV BP,OFFSET HOUR

MOV BH,0

MOV DH,0

MOV DL,0

MOV BL,07H

MOV CX,BUF\_LEN

MOV AL,0 ;13H用于将字符串写到指定的光标位置

MOV AH,13H

INT 10H

MOV BH,0

MOV DX,CURSOR

MOV AH,2

INT 10H

POP ES

POP DS

POPA

IRET

NEW08H ENDP

;

GET\_TIME PROC

MOV AL,4

OUT 70H,AL

JMP $+2

IN AL,71H

MOV AH,AL

AND AL,0FH

SHR AH,4

ADD AX,3030H

XCHG AH,AL

MOV WORD PTR HOUR,AX

MOV AL,2

OUT 70H,AL

JMP $+2

IN AL,71H

MOV AH,AL

AND AL,0FH

SHR AH,4

ADD AX,3030H

XCHG AH,AL

MOV WORD PTR MIN,AX

MOV AL,0

OUT 70H,AL

JMP $+2

IN AL,71H

MOV AH,AL

AND AL,0FH

SHR AH,4

ADD AX,3030H

XCHG AH,AL

MOV WORD PTR SEC,AX

RET

GET\_TIME ENDP

;

BEGIN: PUSH CS

POP DS

MOV AX,3508H

INT 21H

MOV OLD\_INT,BX

MOV OLD\_INT+2,ES

MOV DX,OFFSET NEW08H

MOV AX,2508H

INT 21H

;...

STORE: MOV AH,0

INT 16H

CMP AL,'s'

JNE NEXT

MOV DX,OFFSET BEGIN+15;使代码数向上取整

MOV CL,4

SHR DX,CL;除以16

ADD DX,50H;驻留begin下的代码区域

MOV AL,0

MOV AH,31H;31号调用驻留代码段

INT 21H

NEXT: MOV AH,0

INT 16H

CMP AL,'q'

JNE NEXT

LDS DX,DWORD PTR OLD\_INT

MOV AX,2508H

INT 21H

MOV AH,4CH

INT 21H

CODE ENDS

END BEGIN

5.3.2.2 DOS与BIOS调用说明

new08h模块:

MOV BH,0 ;

MOV AH,3

INT 10H;

获取0号显示页面当前光标的位置，bh输入页号（图形模式为0），dh、dl中存放行与列，cx中存放当前光标模式

MOV AL,0

MOV AH,13H

INT 10H

用于将字符串写到指定的光标位置，al=0时表示写后光标位置不变，dh，dl分别存放了起始光标的位置，es：[bp]中存放待显示字符串的首地址。

MOV DX,CURSOR

MOV AH,2

INT 10H

Dh、dl中存放需要置光标位置的行与列，将光标重新清零。

Begin模块:

MOV AX,3508H

INT 21H

35号调用取08h调用的中断矢量，es：[bx]中放入口地址，将旧的08h入口地址存于old\_int中，然后将new\_int首地址放入dx

MOV DX,OFFSET NEW08H

MOV AX,2508H

INT 21H

25号调用将新的08h调用替换原来，al存放中断类型号，dx存放新调用首地址。

Store模块：

MOV AL,0

MOV AH,31H;31号调用驻留代码段

INT 21H

Al中放置退出码（我也不知道写啥，试了试0可以）dx为驻留节数（保险起见我多加了50h）

Next模块:

MOV AH,0

INT 16H

Al中存放输入字符的acsii码。

5.3.2.3 CPU响应过程

CPU中断响应时需要在堆栈中保护标志寄存器的内容并自动清除某些标志位（为了使CPU在执行中断程序时不再允许新的中断发生）。当CPU处理完中断返回原来程序时，中断返回指令iret会将中断响应时保存在堆栈中的标志寄存器内容还原。而这种对标志寄存器的处理是子程序无法做到的。

# 5.4实验感想

本节内容新介绍了一种程序处理方式，与以往的子程序调用不同，它事实上是一种对未知情况的反应。我觉得它与子程序最大的不同就在于子程序可以预见调用的时机，但是中断程序则是由系统工作环境决定（随机性）。中断程序可以提高cpu资源的利用率。但是第一次写中断程序，特别是一开始概念还没很清晰的时候，完全找不到思路，这是我在本次实验第一个困难的地方。第二个就是巨大的dos调用库和bios调用库，很多时候找不到自己想要的操作。经过本次试验，虽然对中断程序有了一定的了解，但是还是很难自己构建一个中断程序，我觉得这需要深厚的汇编基础，我还需加倍努力多加练习。

# 5.5 参考文献

[1] 王元珍、曹忠升、韩宗芬.80X86汇编语言程序设计(第一版).华中科技大学出版社，2005年:1-348

[2] 许向阳.80X86汇编语言程序设计上机指南(第一版).华中科技大学出版社,2007年:1-224