操作系统实验报告

计科（大数据、人工智能方向）17341155 王永康

（实验四：具有中断处理的内核）

一、实验目的

学习中断机制知识，掌握中断处理程序设计的要求，掌握异步事件编程技术

二、实验内容

- 操作系统工作期间，利用时钟中断，在屏幕24行79列位置轮流显示'|'、'/'和'\'，适当控制显示速度，以方便观察效果。
- 编写键盘中断响应程序，原有的你设计的用户程序运行时，键盘事件会做出有事反应：当键盘有按键时，屏幕适当位置显示"OUCH! OUCH!"。
- 在内核中，对33号、34号、35号和36号中断编写中断服务程序，分别在屏幕1/4区域内显示一些个性化信息。再编写一个汇编语言的程序，作为用户程序，利用int 33、int 34、int 35和int 36产生中断调用这4个服务程序。

三、实验方案

操作系统实验工具与环境

- 实验支撑环境
  - 硬件：个人计算机
  - 主机操作系统：Windows10
  - 虚拟机软件：VMware,Dosbox
- 实验开发工具：
  - 汇编语言工具：x86汇编语言
  - 汇编编译工具：TASM+TCC+NASM
  - 磁盘写入工具：Winhex

四、实验过程

编写时钟中断程序：风火轮程序

利用时钟中断号8号，修改中断向量表，8号中断向量对应8*4和8*4+2两个地址，把9号中断需要执行的部分的偏移入口地址放置在8*4中，段地址放置在8*4+2中，具体代码如下：

```
org 1000h            ; 程序加载到100h，可用于生成COM
; 设置时钟中断向量（08h），初始化段寄存器
_start:

    mov ax,cs
```

```
    mov ds,ax           ; DS = CS
    mov es,ax

    xor ax,ax           ; AX = 0
    mov es,ax           ; ES = 0
    mov ax,0B800h       ; 文本窗口显存起始地址
    mov gs,ax       ; GS = B800h


    cli
    mov word [es:20h],Timer1    ; 设置时钟中断向量的偏移地址
    mov word [es:22h],cs
    sti

    mov ax,cs
    mov ds,ax           ; DS = CS
    mov es,ax


    ret



    jmp $           ; 死循环
; 时钟中断处理程序
Timer1:
    jmp short Timer
Timer:

    push ax
    push bx
    push cx
    push dx

    push bp
    push es

    dec byte [count]        ; 递减计数变量
    jnz _end            ; >0: 跳转

    mov al, byte[cnt]

    cmp al,0
    jne here1
    mov byte[gs:((80*15+70)*2)],'\'
    jmp here0

    here1:
    cmp al,1
    jne here2
    mov byte[gs:((80*15+70)*2)],'|'
    jmp here0

```

```
    here2:
    cmp al,2
    jne here3
    mov byte[gs:((80*15+70)*2)],'/'
    jmp here0

    here3:
    cmp al,3
    jne here0

    mov byte[gs:((80*15+70)*2)],'-'

    here0:
    inc byte[cnt]
    cmp byte[cnt],4
    jne here5
    mov byte[cnt],0
    here5:
    mov byte[count],2       ; 重置计数变量=初值delay
    ;sti

_end:
    mov al,20h          ; AL = EOI
    out 20h,al          ; 发送EOI到主8529A
    out 0A0h,al         ; 发送EOI到从8529A


    pop es
    pop bp
    pop dx
    pop cx

    pop bx
    pop ax

    iret            ; 从中断返回
_data:
    delay equ 4     ; 计时器延迟计数
    count db delay      ; 计时器计数变量，初值=delay
    cnt db 8
times 510-($-$$) db 0
dw 0xaa55
```

之后，把该程序放在内存偏移量为1000h的位置处，在进入后内核后**默认执行**该段程序，把它加载到1000h处，所以可以做到在屏幕右下角显示一个一直转的风火轮的程序。

加载程序如下：

```
_Load_Timer proc
    push ds
    push es
        mov ax,cs               ;段地址 ； 存放数据的内存基地址
        mov es,ax               ;设置段地址（不能直接mov es,段地址）
```

```
        mov bx, Time_OffSet  ;偏移地址; 存放数据的内存偏移地址
        mov ah,2                    ; 功能号
        mov al,1                    ;扇区数
        mov dl,0                    ;驱动器号 ; 软盘为0，硬盘和U盘为80H
        mov dh,0                    ;磁头号 ; 起始编号为0
        mov ch,0                    ;柱面号 ; 起始编号为0
        mov cl, byte ptr _Timer_sector                ;起始扇区号 ; 起始编号为1
        int 13H ;                   调用读磁盘BIOS的13h功能
        ; 用户程序a.com已加载到指定内存区域中
        mov bx, Time_OffSet
        call bx

    pop es
    pop ds
    ret
_Load_Timer endp
```

修改9号中断编程

同理，对9号中断的向量表的位置9*4和9*4+2处存入新的9号中断**执行程序的开始偏移地址**和**段地址**；新9号中断执行的功能是，只能按下键盘，就会在屏幕上显示“OUCH!OUCH!”的字样，并隔一段时间会消失。具体代码如下：(具体说明见注释)

```
NewINT9 proc
        ;保护寄存器
      push ax
        push bx
        push cx
        push bp
        push es

    in al,60h
    mov ax,cs
    mov ds,ax

        ;;;;;;;;;;;以下是显示OUCH
    ;mov es,St


        mov  si, offset Stt
    mov ax,0B800h
        mov es,ax


    mov es,ax
    mov di,(12*80 + 5)*2

    mov cx,10

        s:mov al,[si]
    mov es:[di],al
```

```
    inc si
    add di,2
    loop s
    ;;;;;;;;;;

        ;;;延迟一段时间
        call LOOPSs

        ;;;;;清除OUCH
        mov si,offset Smm
    mov ax,0B800h
    mov es,ax
    mov di,(12*80 + 5)*2
      mov cx,10
  s3:mov al,[si]
    mov es:[di],al
    inc si
    add di,2
    loop s3

        ;;;;;;清除结束


        ;;;;貌似下面是说发生到芯片端口，告诉中断结束？？？

   mov al,20h           ; AL = EOI
     out 20h,al         ; 发送EOI到主8529A
   out 0A0h,al          ; 发送EOI到从8529A


        pop es
        pop bp
        pop cx
        pop bx
        pop ax
    iret

    ret
NewINT9 endp

```

上述代码来自**内核**，修改9号中断的位置是在**内核的用户加载程序**处，并且在调用完用户程序后**恢复**9号中断的原始功能——赋回原值。与此相关的内核代码：

```
_Load proc
    push ds
    push es
        ;;;;;;
        ;;……
        ;;;;;;相关部分代码：
        xor ax,ax
        mov es,ax
        mov ax,es:[24h]
```

```
        ;;保存9号中断的原始功能
        mov word ptr cs:[stack1],ax
        mov ax, es:[26h]
        mov word ptr cs:[stack1+2],ax

        ;;修改9号中断
        cli
        mov word ptr es:[24h],offset NewINT9
        mov ax,cs
        mov word ptr es:[26h],ax
        sti

        mov ax,cs
        mov ds,ax
        mov es,ax


        ;;;执行用户程序
        call bx


        ;;;;执行结束，恢复中断
        xor ax,ax
        mov es,ax

        cli
        mov ax,cs:[stack1]
        mov word ptr es:[24h],ax

        mov ax, cs:[stack1+2]
        mov word ptr es:[26h],ax
        sti

    mov ax,cs
    mov es,ax
    mov ds,ax

    here4:
    pop es
    pop ds
    ret
_load endp
```

这样做，可以使调用用户程序的时候，按键盘会显示“OUCH！OUCH!”的信息，执行结束自动恢复9号中断。

修改33、34、35、36号程序编程

原理同上，新的33、34、35、36中断指向的程序作用是：去加载第4、5、6、7扇区的程序到内存中，并且运行；

相关代码：

```
NewINT33 proc
    push ds
    push es

        mov byte ptr cs:[Sectors],4;;;指定加载第4扇区的用户程序
        call LOAD_for_INT ;;;调用加载函数

    pop es
    pop ds

    iret
    ret
NewINT33 endp
```

加载函数代码：

```
LOAD_for_INT proc
    push ds
    push es
        mov ax,cs                ;段地址 ; 存放数据的内存基地址
        mov es,ax                ;设置段地址（不能直接mov es,段地址）
        mov bx, OffSetOfUserPrg4  ;偏移地址; 存放数据的内存偏移地址
        mov ah,2                 ; 功能号
        mov al,1                 ;扇区数
        mov dl,0                 ;驱动器号 ; 软盘为0，硬盘和U盘为80H
        mov dh,0                 ;磁头号 ; 起始编号为0
        mov ch,0                 ;柱面号 ; 起始编号为0
        mov cl,byte ptr es:[Sectors]              ;起始扇区号 ; 起始编号为1
        int 13H ;                调用读磁盘BIOS的13h功能
        ; 用户程序a.com已加载到指定内存区域中
        mov bx, OffSetOfUserPrg4
        call bx
    pop es
    pop ds

    ret
LOAD_for_INT endp
```

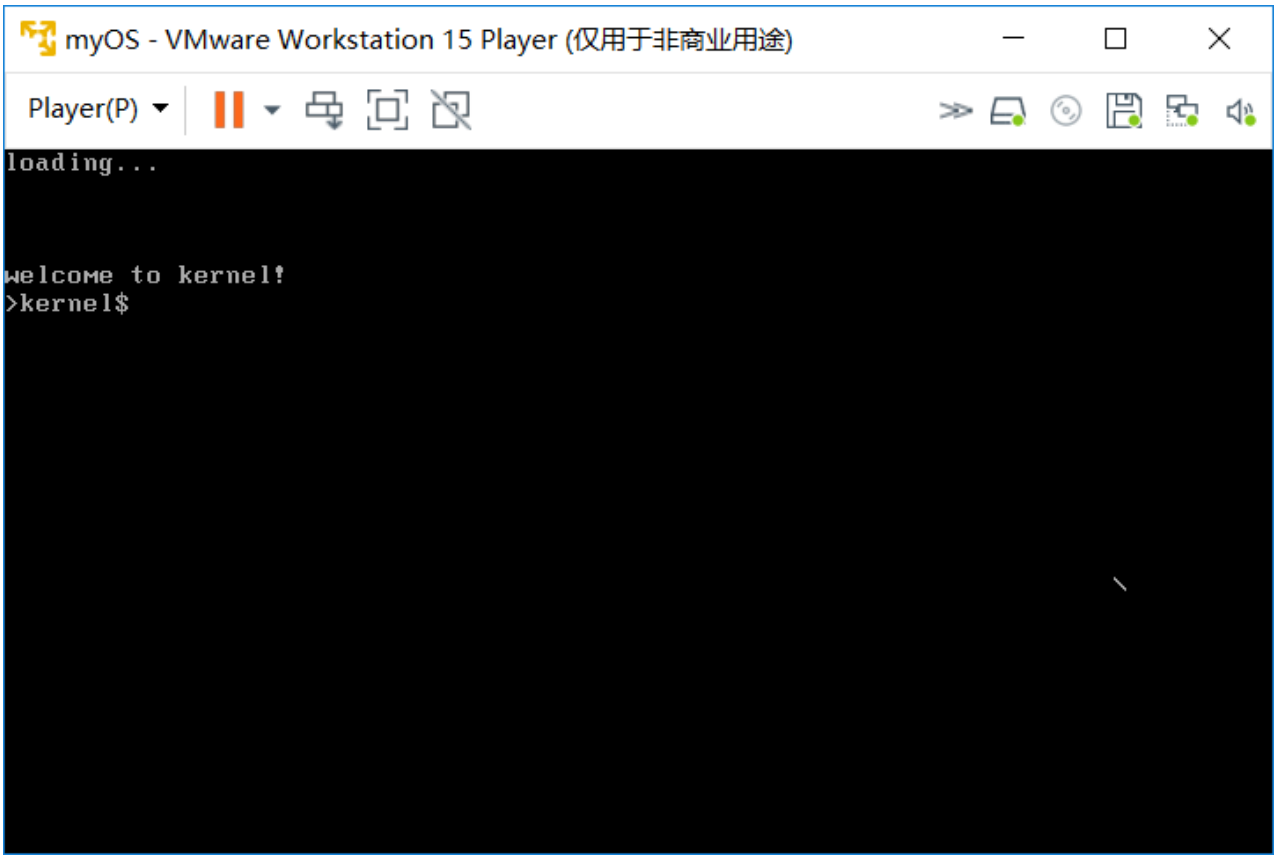
再次，编写用户程序：

```
;;wang yongkang,haha
org 09000h
start:
    mov ax,cs
    mov ds,ax
    mov es,ax

    int 33

    int 34

    int 35
```

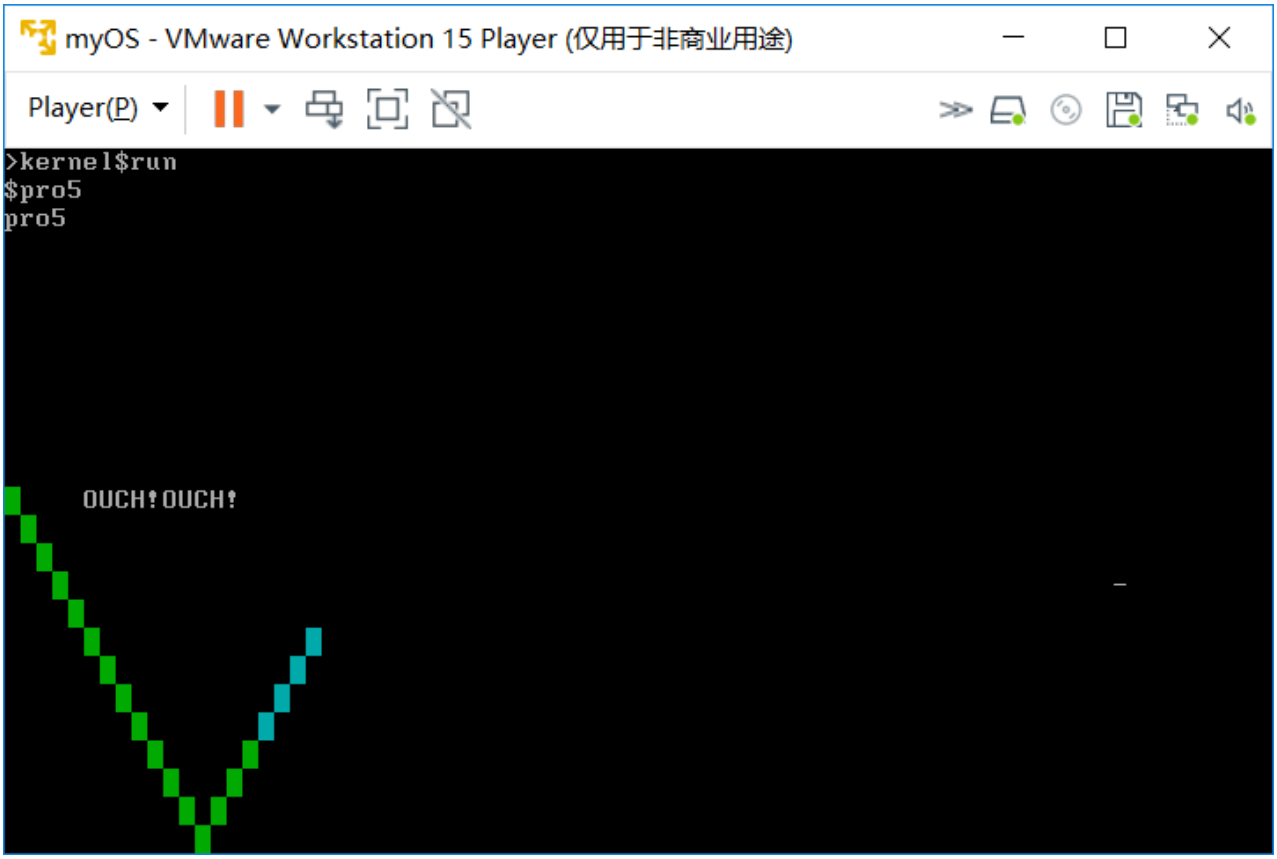
```
13
14      int 36
15
16      ret
17      jmp $
18
19  times 510-($-$$) db 0
20  dw 0xaa55
```

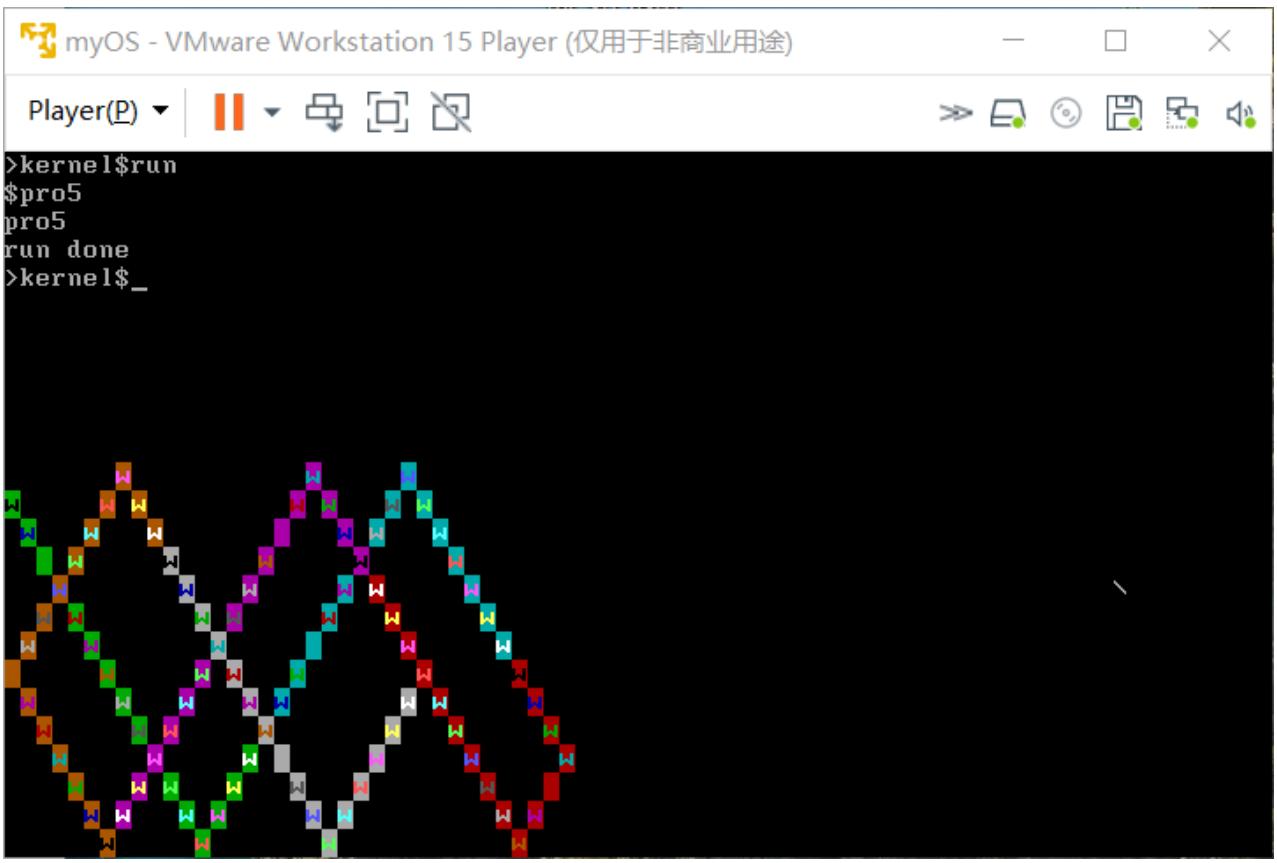
五、实验结果

- 时钟中断演示:



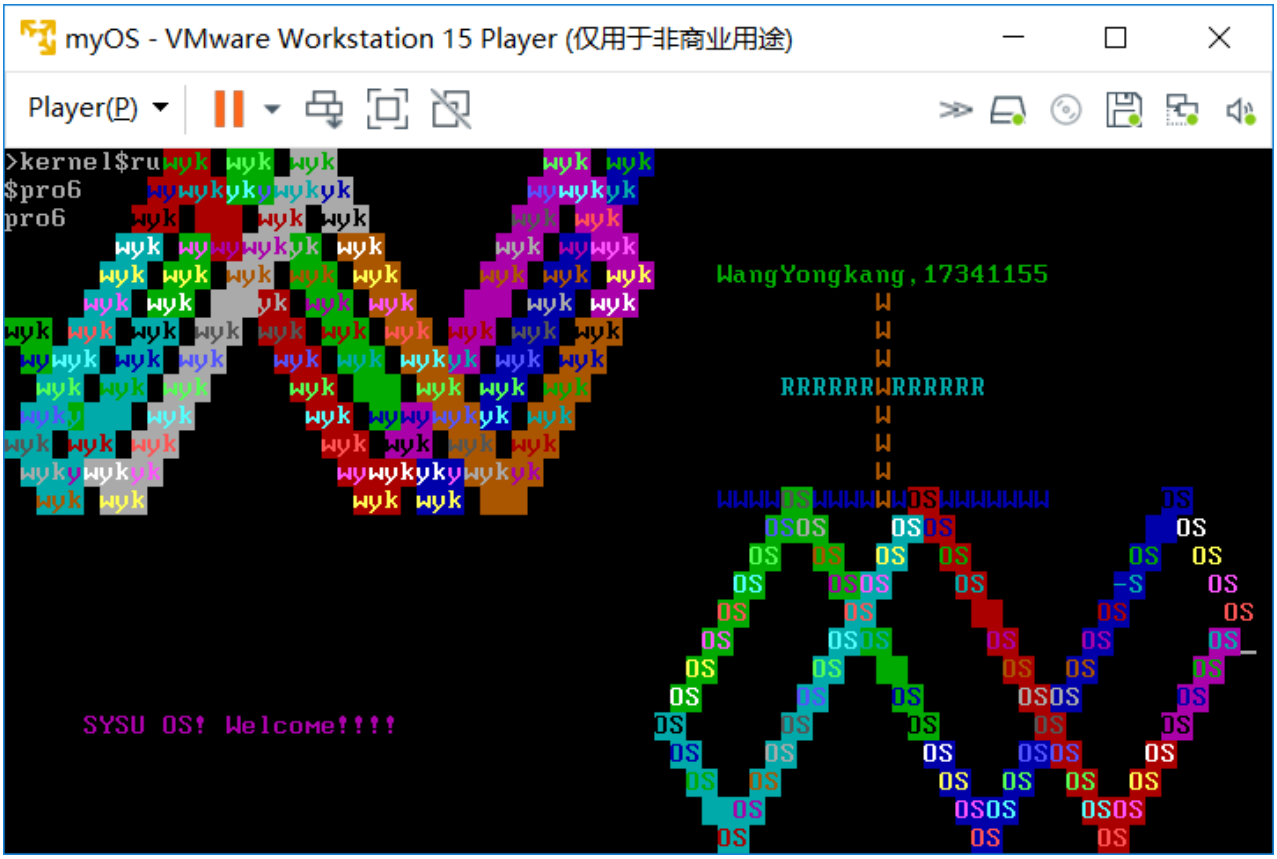
- 9号中断演示





- 执行33、34、35、36中断

  其中通过运行程序6，pro5，将会接连调用pro1,pro2,pro3,pro4,演示效果如下：



- 最终软盘a.img的各项说明：

a.img的扇区	储存内容
1	遇到程序myOS.bin
2	储存软盘信息的文件
3	时钟中断程序timer.bin
4	左上角1/4程序（pro1）
5	左下角1/4程序（pro2）
6	右上角1/4程序（pro3）
7	右下角1/4程序（pro4）
8	弹跳字符程序（pro5）
9	int33,34,35,36测试程序（pro6）
12-16	KER.bin,内核程序

六、实验总结

这一次中断编程的实验，主要有以下体会：

- 遇到很多问题与**寄存器的保存**有关；

在实验实现过程中，以9号中断的改写为例，开始时没有注意在9号中断里面保护相关寄存器，导致原用户程序出错；

- **段地址保存，偏移地址的保存**

  更加深入地认识到x86实模式下的寻址模式，段值加偏移，所以中断实现的关键也是需要保存段值和偏移。

- 还遇到一个未解决的问题：
  - 时钟中断的风火轮程序在进入**内核后14秒**才出现，感觉比较奇怪，按照正常的逻辑，应该一来就应该正常运行，这里出现的时间延迟比较奇怪，个人猜测是虚拟机本身虚拟的配置问题引起的。

参考文献

[1] 凌应标. “04实验课.ppt”，中山大学计算机科学系，2015-3

[2] 于渊.《一个操作系统的实现》[M].电子工业出版社，2009-6-1

[3] 王爽.《汇编语言(第3版)》[M].清华大学出版社，2013-9