操作系统
实验报告

院系：计算机学院

班级：计科 2 班

学号：19335174

姓名：施天予

指导老师：凌应标

2021.3.26

一、实验题目

加载执行 COM 格式用户程序的监控程序

二、实验目的

1、了解监控程序执行用户程序的主要工作

2、了解一种用户程序的格式与运行要求

3、加深对监控程序概念的理解

4、掌握加载用户程序方法

5、掌握几个 BIOS 调用和简单的磁盘空间管理

三、实验要求

1、知道引导扇区程序实现用户程序加载的意义

2、掌握 COM/BIN 等一种可执行的用户程序格式与运行要求

3、将自己实验一的引导扇区程序修改为 3-4 个不同版本的 COM 格式程序，每个程序缩小显示区域，在屏幕特定区域显示，用以测试监控程序，在 1.44MB 软驱映像中存储这些程序。

4、重写 1.44MB 软驱引导程序，利用 BIOS 调用，实现一个能执行 COM 格式用户程序的监控程序。

5、设计一种简单命令，实现用命令交互执行在 1.44MB 软驱映像中存储几个用户程序。

6、编写实验报告，描述实验工作的过程和必要的细节，如截屏或录屏，以证实实验工作的真实性。

四、实验方案

【实验环境】

1、实验运行环境：Windows10

2、虚拟机软件：VirtualBox

【实验工具】

1、汇编语言：NASM

2、文本编辑器：Notepad++

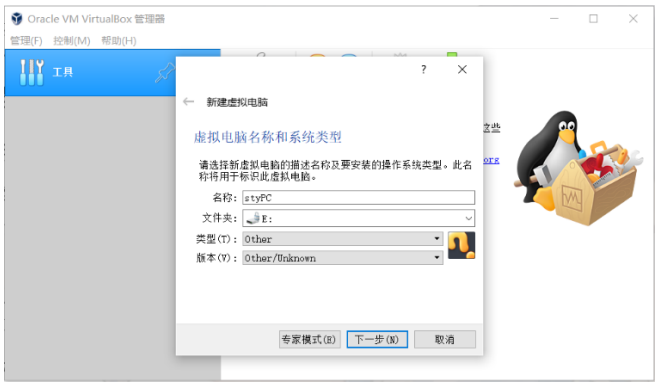
3、相关软件：WinHex

五、实验过程

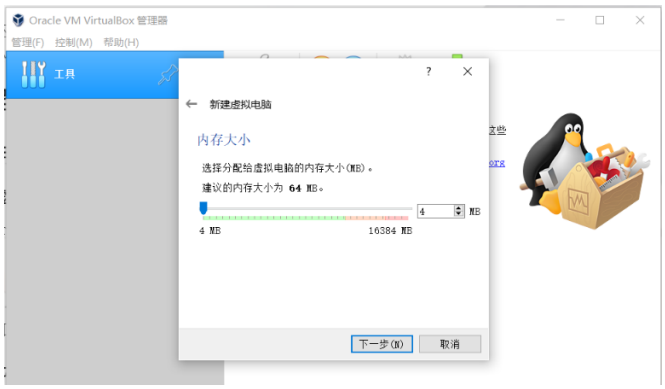
1、搭建和应用实验环境

【安装虚拟机】

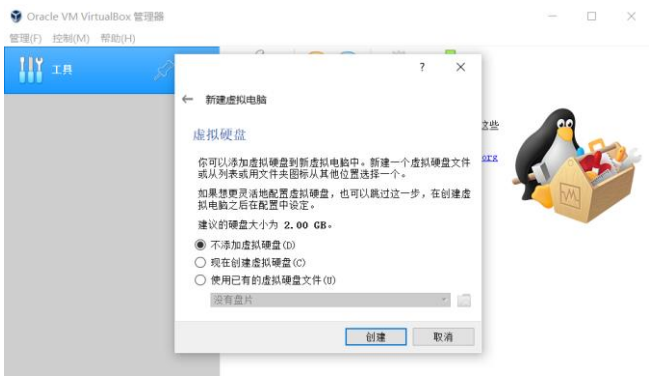
打开 VirtualBox，新建无操作系统的虚拟机



内存设置为 4MB

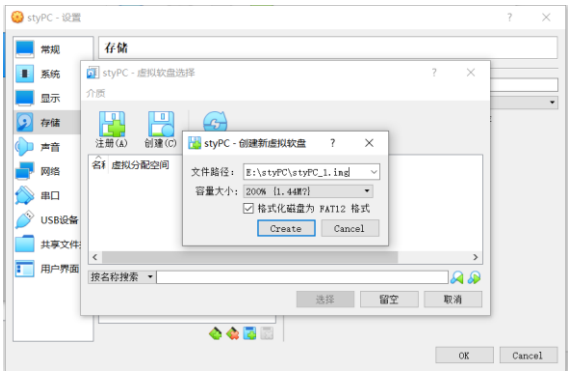


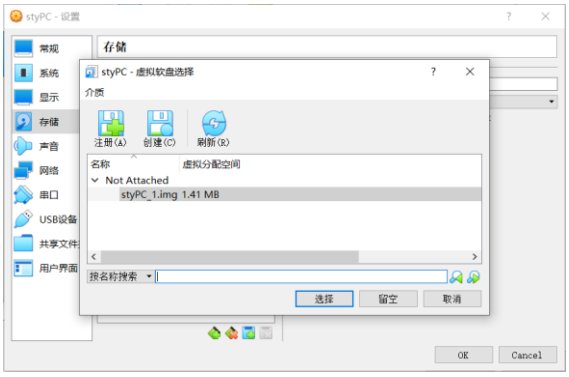
选择不添加虚拟硬盘



【虚拟软盘】

创建虚拟软盘后，选择注册虚拟软盘





2、设计监控程序

【设计代码】

```
org  7c00h  ;监控程序地址
OffsetOfUserPrg equ 8100h    ;用户程序地址

Start:
    mov ax, cs        ; 置其他段寄存器值与CS相同
    mov ds, ax        ; 数据段
    mov bp, Message   ; BP=当前串的偏移地址
    mov ax, ds        ; ES:BP = 串地址
    mov es, ax        ; 置ES=DS
cls:          ; 清屏
    mov ah,6
    mov al,0
    mov ch,0
    mov cl,0
    mov dh,24
    mov dl,79
    mov bh,7
    int 10h
show:
    mov cx, MessageLength  ; CX = 串长（=9）
    mov ax, 1301h     ; AH = 13h（功能号）、AL = 01h（光标置于串尾）
    mov bx, 0007h     ; 页号为0(BH = 0) 黑底白字(BL = 07h)
    mov dh, 0         ; 行号=0
    mov dl, 0         ; 列号=0
    int 10h       ; BIOS的10h功能：显示一行字符

input:
    mov ah,0
    int 16h
    cmp al,'1'
    jz switch
    cmp al,'2'
    jz switch
    cmp al,'3'
    jz switch
    cmp al,'4'
    jz switch
    jmp input
```

```
39
40  switch:
41      mov bl,'1'
42      cmp al,bl
43      jz program1
44      mov bl,'2'
45      cmp al,bl
46      jz program2
47      mov bl,'3'
48      cmp al,bl
49      jz program3
50      mov bl,'4'
51      cmp al,bl
52      jz program4
53
54  program1:
55      mov ax,cs                   ;段地址 ; 存放数据的内存基地址
56      mov es,ax                   ;设置段地址（不能直接mov es,段地址）
57      mov bx, OffsetOfUserPrg             ;偏移地址; 存放数据的内存偏移地址
58      mov ah,2                    ;功能号，读磁盘
59      mov al,1                    ;读入扇区数
60      mov dl,0                    ;驱动器号 ; 软盘为0，硬盘和U盘为80H
61      mov dh,0                    ;磁头号 ; 起始编号为0
62      mov ch,0                    ;柱面号 ; 起始编号为0
63      mov cl,2                    ;扇区号2
64      int 13H ;                   调用读磁盘BIOS的13h功能
65      jmp OffsetOfUserPrg
66  program2:
67      mov ax,cs                   ;段地址 ; 存放数据的内存基地址
68      mov es,ax                   ;设置段地址（不能直接mov es,段地址）
69      mov bx, OffsetOfUserPrg             ;偏移地址; 存放数据的内存偏移地址
70      mov ah,2                    ;功能号，读磁盘
71      mov al,1                    ;读入扇区数
72      mov dl,0                    ;驱动器号 ; 软盘为0，硬盘和U盘为80H
73      mov dh,0                    ;磁头号 ; 起始编号为0
74      mov ch,0                    ;柱面号 ; 起始编号为0
75      mov cl,3                    ;扇区号3
76      int 13H ;                   调用读磁盘BIOS的13h功能
77      jmp OffsetOfUserPrg
78  program3:
79      mov ax,cs                   ;段地址 ; 存放数据的内存基地址
80      mov es,ax                   ;设置段地址（不能直接mov es,段地址）
81      mov bx, OffsetOfUserPrg             ;偏移地址; 存放数据的内存偏移地址
82      mov ah,2                    ;功能号，读磁盘
83      mov al,1                    ;读入扇区数
84      mov dl,0                    ;驱动器号 ; 软盘为0，硬盘和U盘为80H
85      mov dh,0                    ;磁头号 ; 起始编号为0
86      mov ch,0                    ;柱面号 ; 起始编号为0
87      mov cl,4                    ;扇区号4
88      int 13H ;                   调用读磁盘BIOS的13h功能
89      jmp OffsetOfUserPrg
90  program4:
91      mov ax,cs                   ;段地址 ; 存放数据的内存基地址
92      mov es,ax                   ;设置段地址（不能直接mov es,段地址）
93      mov bx, OffsetOfUserPrg             ;偏移地址; 存放数据的内存偏移地址
94      mov ah,2                    ;功能号，读磁盘
95      mov al,1                    ;读入扇区数
96      mov dl,0                    ;驱动器号 ; 软盘为0，硬盘和U盘为80H
97      mov dh,0                    ;磁头号 ; 起始编号为0
98      mov ch,0                    ;柱面号 ; 起始编号为0
99      mov cl,5                    ;扇区号5
100     int 13H ;                   调用读磁盘BIOS的13h功能
101     jmp OffsetOfUserPrg
102
103 Message:
104     db "Welcome!",0AH,0DH
105     db "Please Enter the number to choose the different program: ",0AH,0D
106     db "1.Number",0AH,0DH
107     db "2.Name",0AH,0DH
108     db "3.Rectangle",0AH,0DH
109     db "4.Stone",0AH,0DH
110
111     MessageLength  equ ($-Message)
112     times 510-($-$$) db 0
113     db 0x55,0xaa
```

【代码分析】

按照老师 PPT 中的代码改善，首先将用户程序地址设置为 8100h。因为在监控程序和用户程序切换时会残留之前显示的内容，所以我增加了一个清屏功能。用 int 16h 的 BIOS 功能调用，读取键盘输入，设置 1,2,3,4 跳到不同的用户程序。用户程序 1 在第 2 个扇区，用户程序 2 在第 3 个扇区，用户程序 3 在第 4 个扇区，用户程序 4 在第 5 个扇区。用 times 510-($-$$) db 0 和 db 0x55,0xaa 最后完成一个引导扇区程序。

3、 设计用户程序

【用户程序 1：彩色移动显示学号】

```
    Dn_Rt equ 1                 ;D-Down,U-Up,R-right,L-Left
    Up_Rt equ 2                 ;
    Up_Lt equ 3                 ;
    Dn_Lt equ 4                 ;
    delay equ 50000             ; 计时器延迟计数,用于控制画框的速度
    ddelay equ 580              ; 计时器延迟计数,用于控制画框的速度
    org 8100h

cls: ;清屏
    mov ah,6
    mov al,0
    mov ch,0
    mov cl,0
    mov dh,24
    mov dl,79
    mov bh,7
    int 10h

start:
    mov ax,cs                   ;获得程序运行时，代码段在内存的位置
    mov ds,ax                   ; DS = CS
    mov ss,ax                   ; SS = CS
    mov ax,0B800h               ; 文本窗口显存起始地址
    mov es,ax                   ; ES = B800h

loop1:
    dec word[count]             ; 递减计数变量
    jnz loop1                   ; >0: 跳转;
    mov word[count],delay
    dec word[dcount]                ; 递减计数变量
      jnz loop1
    mov word[count],delay
    mov word[dcount],ddelay

      mov al,1
      cmp al,byte[rdul]
    jz  DnRt
      mov al,2
```

```
      cmp al,byte[rdul]
    jz  UpRt
      mov al,3
      cmp al,byte[rdul]
    jz  UpLt
      mov al,4
      cmp al,byte[rdul]
    jz  DnLt
      jmp $

DnRt:
    inc word[x]
    inc word[y]
    mov bx,word[x]
    mov ax,14
    sub ax,bx
      jz  dr2ur
    mov bx,word[y]
    mov ax,40
    sub ax,bx
      jz  dr2dl
    jmp show
dr2ur:
      mov word[x],12
      mov byte[rdul],Up_Rt
      jmp show
dr2dl:
      mov word[y],38
      mov byte[rdul],Dn_Lt
      jmp show
UpRt:
    dec word[x]
    inc word[y]
    mov bx,word[y]
    mov ax,40
    sub ax,bx
      jz  ur2ul
    mov bx,word[x]
```

```
    mov ax,-1
    sub ax,bx
      jz  ur2dr
    jmp show
ur2ul:
      mov word[y],38
      mov byte[rdul],Up_Lt
      jmp show
ur2dr:
      mov word[x],1
      mov byte[rdul],Dn_Rt
      jmp show
UpLt:
    dec word[x]
    dec word[y]
    mov bx,word[x]
    mov ax,-1
    sub ax,bx
      jz  ul2dl
    mov bx,word[y]
    mov ax,-1
    sub ax,bx
      jz  ul2ur
    jmp show
ul2dl:
      mov word[x],1
      mov byte[rdul],Dn_Lt
      jmp show
ul2ur:
      mov word[y],1
      mov byte[rdul],Up_Rt
      jmp show
DnLt:
    inc word[x]
    dec word[y]
    mov bx,word[y]
    mov ax,-1
    sub ax,bx
```

```
      jz  dl2dr
    mov bx,word[x]
    mov ax,14
    sub ax,bx
      jz  dl2ul
    jmp show
dl2dr:
      mov word[y],1
      mov byte[rdul],Dn_Rt
      jmp show
dl2ul:
      mov word[x],12
      mov byte[rdul],Up_Lt
      jmp show

show:
    xor ax,ax
    mov word ax,[x]
    mov bx,80
    mul bx
    add word ax,[y]
    mov bx,2
    mul bx
    mov bx,ax
    mov si, NUMBER
    mov cx, 8
    color:
        mov byte al,[si]
```

```
        mov [es:bx],ax
        inc si
        inc bx
        inc bx
    loop color

    ; 输入空格后弹出程序
    mov ah,1
    int 16h
    mov bl,20h
    cmp al,bl
    jz Quit
    jmp loop1

Quit:
    jmp 7c00h
end:
    jmp $                   ; 停止画框，无限循环

datadef:
    count dw delay
    dcount dw ddelay
    rdul db Dn_Rt          ; 向右下运动
    x    dw -1
    y    dw 0
    NUMBER db "19335174"

    times 512-($-$$) db 0
```

【代码分析】

每个用户程序都要在开头 org 8100h，并且进行一次清屏操作。第一个用户程序与实验 1 的代码类似，只是变色移动的是学号，并且改了一下边界参数使其出现在屏幕左上方区域。在程序末尾使用 int 16h 的 1 号功能，实现捕捉键盘输入，在输入空格时，用 jmp 7c00h 返回监控程序。times 512-($-$$) db 0 在最后使生成的 COM 文件 512 字节。

【用户程序 2：彩色移动显示姓名】

```
    Dn_Rt equ 1                 ;D-Down,U-Up,R-right,L-Left
    Up_Rt equ 2                 ;
    Up_Lt equ 3                 ;
    Dn_Lt equ 4                 ;
    delay equ 50000             ; 计时器延迟计数,用于控制画框的速度
    ddelay equ 580              ; 计时器延迟计数,用于控制画框的速度
    org 8100h

cls: ;清屏
    mov ah,6
    mov al,0
    mov ch,0
    mov cl,0
    mov dh,24
    mov dl,79
    mov bh,7
    int 10h

start:
    mov ax,cs                   ;获得程序运行时，代码段在内存的位置
    mov ds,ax                   ; DS = CS
    mov ss,ax                   ; SS = CS
    mov ax,0B800h               ; 文本窗口显存起始地址
    mov es,ax                   ; ES = B800h

loop1:
    dec word[count]             ; 递减计数变量
    jnz loop1                   ; >0：跳转;
    mov word[count],delay
    dec word[dcount]                ; 递减计数变量
      jnz loop1
    mov word[count],delay
    mov word[dcount],ddelay

      mov al,1
      cmp al,byte[rdul]
    jz  DnRt
      mov al,2
      cmp al,byte[rdul]
    jz  UpRt
      mov al,3
      cmp al,byte[rdul]
    jz  UpLt
      mov al,4
      cmp al,byte[rdul]
    jz  DnLt
      jmp $

DnRt:
    inc word[x]
    inc word[y]
    mov bx,word[x]
    mov ax,25
    sub ax,bx
      jz  dr2ur
    mov bx,word[y]
    mov ax,40
    sub ax,bx
      jz  dr2dl
    jmp show
dr2ur:
      mov word[x],23
      mov byte[rdul],Up_Rt
      jmp show
dr2dl:
      mov word[y],38
      mov byte[rdul],Dn_Lt
      jmp show
UpRt:
    dec word[x]
    inc word[y]
    mov bx,word[y]
    mov ax,40
    sub ax,bx
      jz  ur2ul
    mov bx,word[x]
    mov ax,12
    sub ax,bx
      jz  ur2dr
    jmp show
ur2ul:
      mov word[y],38
      mov byte[rdul],Up_Lt
      jmp show
ur2dr:
      mov word[x],14
      mov byte[rdul],Dn_Rt
      jmp show
UpLt:
    dec word[x]
    dec word[y]
    mov bx,word[x]
    mov ax,12
    sub ax,bx
      jz  ul2dl
    mov bx,word[y]
    mov ax,-1
    sub ax,bx
      jz  ul2ur
    jmp show
ul2dl:
      mov word[x],14
      mov byte[rdul],Dn_Lt
      jmp show
ul2ur:
      mov word[y],1
      mov byte[rdul],Up_Rt
      jmp show
DnLt:
    inc word[x]
    dec word[y]
    mov bx,word[y]
    mov ax,-1
    sub ax,bx
      jz  dl2dr
    mov bx,word[x]
    mov ax,25
    sub ax,bx
      jz  dl2ul
    jmp show
dl2dr:
      mov word[y],1
      mov byte[rdul],Dn_Rt
      jmp show
dl2ul:
      mov word[x],23
      mov byte[rdul],Up_Lt
      jmp show

show:
    xor ax,ax
    mov word ax,[x]
    mov bx,80
    mul bx
    add word ax,[y]
    mov bx,2
    mul bx
    mov bx,ax
    mov si, NAME
    mov cx, 3 ; 字符串长度
    color:
        mov byte al,[si]
        mov [es:bx],ax
        inc si
        inc bx
        inc bx
    loop color

    ; 输入空格后弹出程序
    mov ah,1
    int 16h
    mov bl,20h
    cmp al,bl
    jz Quit
    jmp loop1

Quit:
    jmp 7c00h
end:
    jmp $                       ; 停止画框，无限循环

datadef:
    count dw delay
    dcount dw ddelay
    rdul db Dn_Rt           ; 向右下运动
    x    dw 12
    y    dw 0
    NAME db "STY"

    times 512-($-$$) db 0
```

【代码分析】

用户程序 2 与用户程序 1 基本类似，只是彩色移动的是姓名大写缩写。再改了一下边界的参数，实现在屏幕左下角显示。

【用户程序 3：“A”字矩形和学号姓名显示】

```
    Dn_Rt equ 1                 ;D-Down,U-Up,R-right,L-Left
    Up_Rt equ 2                 ;
    Up_Lt equ 3                 ;
    Dn_Lt equ 4                 ;
    delay equ 50000             ; 计时器延迟计数,用于控制画框的速度
    ddelay equ 580              ; 计时器延迟计数,用于控制画框的速度
    org 8100h

cls: ;清屏
    mov ah,6
    mov al,0
    mov ch,0
    mov cl,0
    mov dh,24
    mov dl,79
    mov bh,7
    int 10h

start:
    mov ax,cs                   ;获得程序运行时，代码段在内存的位置
    mov ds,ax                   ; DS = CS
    mov ss,ax                   ; SS = CS
    mov ax,0B800h               ; 文本窗口显存起始地址
    mov es,ax                   ; ES = B800h

loop1:
    dec word[count]             ; 递减计数变量
    jnz loop1                   ; >0：跳转;
    mov word[count],delay
    dec word[dcount]                ; 递减计数变量
      jnz loop1
    mov word[count],delay
    mov word[dcount],ddelay

      mov al,1
      cmp al,byte[rdul]
    jz  DnRt
      mov al,2
      cmp al,byte[rdul]
    jz  UpRt
      mov al,3
      cmp al,byte[rdul]
    jz  UpLt
      mov al,4
      cmp al,byte[rdul]
    jz  DnLt
      jmp $

DnRt:
    inc word[x]
    inc word[y]
    mov bx,word[x]
    mov ax,14
    sub ax,bx
      jz  dr2ur
    mov bx,word[y]
    mov ax,40
    sub ax,bx
      jz  dr2dl
    jmp show
dr2ur:
      mov word[x],12
      mov byte[rdul],Up_Rt
      jmp show
dr2dl:
      mov word[y],38
      mov byte[rdul],Dn_Lt
      jmp show
UpRt:
    dec word[x]
    inc word[y]
    mov bx,word[y]
    mov ax,40
    sub ax,bx
      jz  ur2ul
    mov bx,word[x]
    mov ax,-1
    sub ax,bx
      jz  ur2dr
    jmp show
ur2ul:
      mov word[y],38
      mov byte[rdul],Up_Lt
      jmp show
ur2dr:
      mov word[x],1
      mov byte[rdul],Dn_Rt
      jmp show
UpLt:
    dec word[x]
    dec word[y]
    mov bx,word[x]
    mov ax,-1
    sub ax,bx
      jz  ul2dl
    mov bx,word[y]
    mov ax,-1
    sub ax,bx
      jz  ul2ur
    jmp show
ul2dl:
      mov word[x],1
      mov byte[rdul],Dn_Lt
      jmp show
ul2ur:
      mov word[y],1
      mov byte[rdul],Up_Rt
      jmp show
DnLt:
    inc word[x]
    dec word[y]
    mov bx,word[y]
    mov ax,-1
    sub ax,bx
```

```
      jz  dl2dr
    mov bx,word[x]
    mov ax,14
    sub ax,bx
      jz  dl2ul
    jmp show
dl2dr:
      mov word[y],1
      mov byte[rdul],Dn_Rt
      jmp show
dl2ul:
      mov word[x],12
      mov byte[rdul],Up_Lt
      jmp show

show:
    xor ax,ax
    mov word ax,[x]
    mov bx,80
    mul bx
    add word ax,[y]
    mov bx,2
    mul bx
    mov bx,ax
    mov si, NUMBER
    mov cx, 8
    color:
        mov byte al,[si]
        mov [es:bx],ax
        inc si
        inc bx
        inc bx
    loop color

    ; 输入空格后弹出程序
    mov ah,1
    int 16h
    mov bl,20h
    cmp al,bl
    jz Quit
    jmp loop1

Quit:
    jmp 7c00h
end:
    jmp $                    ; 停止画框，无限循环

datadef:
    count dw delay
    dcount dw ddelay
    rdul db Dn_Rt          ; 向右下运动
    x    dw -1
    y    dw 0
    NUMBER db "19335174"

    times 512-($-$$) db 0
```

【代码分析】

用户程序 3 沿用了之前实验 0 的代码，在屏幕右上角显示一个“A”字矩形，并在中间显示学号姓名，黑底白字。

【用户程序 4：Stone】

```
    Dn_Rt equ 1                 ;D-Down,U-Up,R-right,L-Left
    Up_Rt equ 2                 ;
    Up_Lt equ 3                 ;
    Dn_Lt equ 4                 ;
    delay equ 50000                ; 计时器延迟计数,用于控制画框的速度
    ddelay equ 580                 ; 计时器延迟计数,用于控制画框的速度
    org 8100h

cls:    ;清屏
    mov ah,6
    mov al,0
    mov ch,0
    mov cl,0
    mov dh,24
    mov dl,79
    mov bh,7
    int 10h

start:
    mov ax,cs                   ;获得程序运行时，代码段在内存的位置
    mov ds,ax                   ; DS = CS
    mov ss,ax                   ; SS = CS
    mov ax,0B800h               ; 文本窗口显存起始地址
    mov es,ax                   ; ES = B800h
    mov byte[char],'A'

loop1:
    dec word[count]             ; 递减计数变量
    jnz loop1                   ; >0: 跳转;
    mov word[count],delay
    dec word[dcount]               ; 递减计数变量
      jnz loop1
    mov word[count],delay
    mov word[dcount],ddelay

      mov al,1
      cmp al,byte[rdul]
    jz  DnRt
```

```
      mov al,2
      cmp al,byte[rdul]
    jz  UpRt
      mov al,3
      cmp al,byte[rdul]
    jz  UpLt
      mov al,4
      cmp al,byte[rdul]
    jz  DnLt
      jmp $

DnRt:
    inc word[x]
    inc word[y]
    mov bx,word[x]
    mov ax,25
    sub ax,bx
      jz  dr2ur
    mov bx,word[y]
    mov ax,80
    sub ax,bx
      jz  dr2dl
    jmp show
dr2ur:
      mov word[x],23
      mov byte[rdul],Up_Rt
      jmp show
dr2dl:
      mov word[y],78
      mov byte[rdul],Dn_Lt
      jmp show
UpRt:
    dec word[x]
    inc word[y]
    mov bx,word[y]
    mov ax,80
    sub ax,bx
      jz  ur2ul
```

```
    mov bx,word[x]
    mov ax,12
    sub ax,bx
      jz  ur2dr
    jmp show
ur2ul:
      mov word[y],78
      mov byte[rdul],Up_LL
      jmp show
ur2dr:
      mov word[x],14
      mov byte[rdul],Dn_Rt
      jmp show

UpLt:
    dec word[x]
    dec word[y]
    mov bx,word[x]
    mov ax,12
    sub ax,bx
      jz  ul2dl
    mov bx,word[y]
    mov ax,39
    sub ax,bx
      jz  ul2ur
    jmp show

ul2dl:
      mov word[x],14
      mov byte[rdul],Dn_Lt
      jmp show
ul2ur:
      mov word[y],41
      mov byte[rdul],Up_Rt
      jmp show

DnLt:
    inc word[x]
```

```
    dec word[y]
    mov bx,word[y]
    mov ax,39
    sub ax,bx
      jz  dl2dr
    mov bx,word[x]
    mov ax,25
    sub ax,bx
      jz  dl2ul
    jmp show

dl2dr:
      mov word[y],41
      mov byte[rdul],Dn_Rt
      jmp show

dl2ul:
      mov word[x],23
      mov byte[rdul],Up_Lt
      jmp show

show:
    xor ax,ax                   ; 计算显存地址
    mov ax,word[x]
    mov bx,80
    mul bx
    add ax,word[y]
    mov bx,2
```

```
    mul bx
    mov bp,ax
    mov ah,0Fh
    mov al,byte[char]
    mov word[es:bp],ax
    ; 输入空格后弹出程序
    mov ah,1
    int 16h
    mov bl,20h
    cmp al,bl
    jz Quit
    jmp loop1

Quit:
    jmp 7c00h
end:
    jmp $                       ; 停止画框，无限循环

datadef:
    count dw delay
    dcount dw ddelay
    rdul db Dn_Rt           ; 向右下运动

    x dw 12;     小球起始位置
    y dw 52
    char db 'A'

    times 512-($-$$) db 0
```

【代码分析】

用户程序 4 使用了老师的 stone 程序，改进代码使之显示在屏幕右下角。

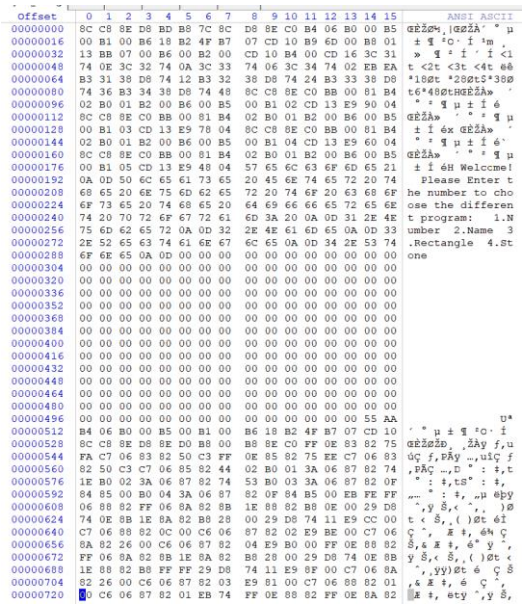
4、NASM 编译

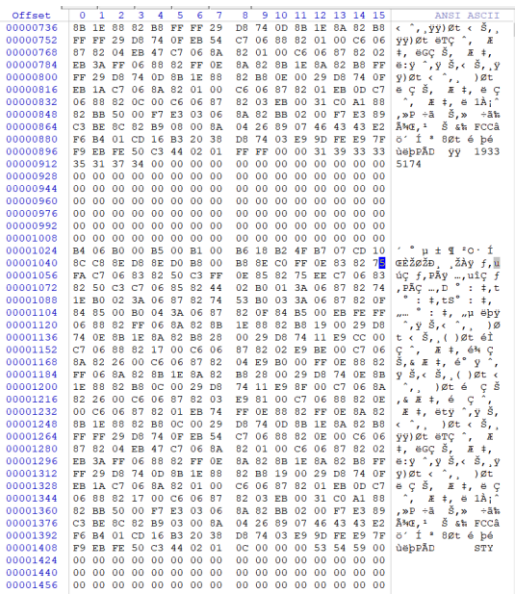
nasm

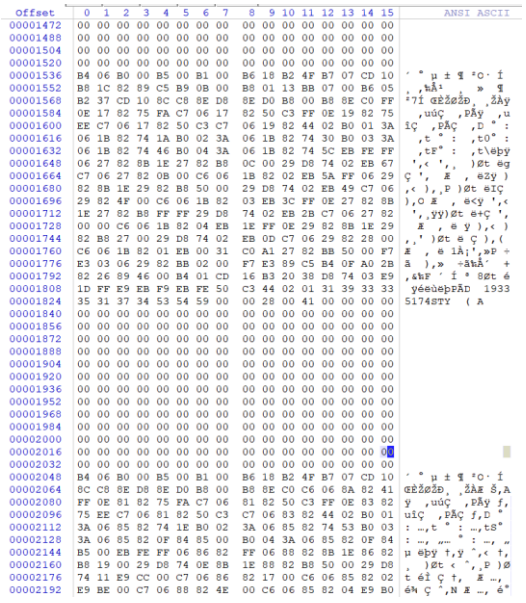
```
Microsoft Windows [版本 10.0.18363.1441]
(c) 2019 Microsoft Corporation。保留所有权利。

E:\nasm>nasm -f bin os.asm -o os.com

E:\nasm>nasm -f bin 1.asm -o 1.com

E:\nasm>nasm -f bin 2.asm -o 2.com

E:\nasm>nasm -f bin 3.asm -o 3.com

E:\nasm>nasm -f bin 4.asm -o 4.com
```

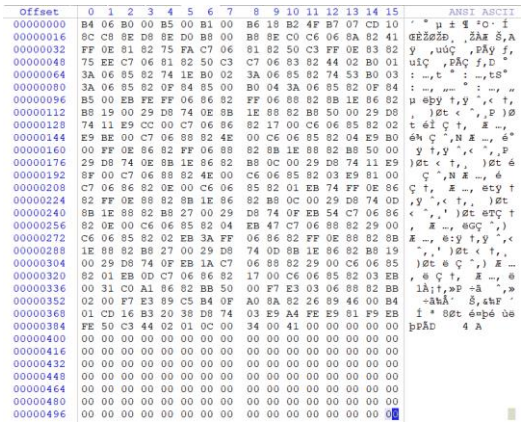
5、使用 WinHex 修改软盘

使用 WinHex 打开所有的 COM 文件和软盘文件。将监控程序 os.com 替换到第一个扇区（0-511 字节），将 1.com 替换到第二个扇区（512-1023 字节），将 2.com 替换到第三个扇区（1024-1535 字节），将 3.com 替换到第四个扇区（1536-2047 字节），将 4.com 替换到第五个扇区（2048-2559 字节）。

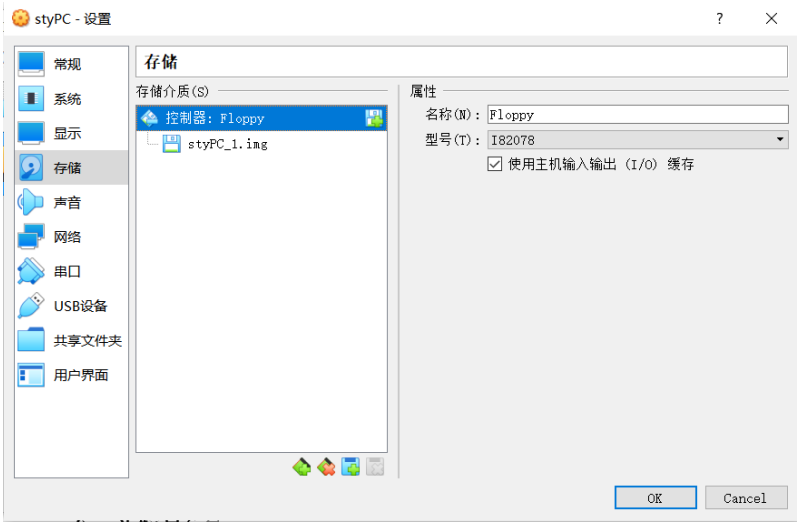




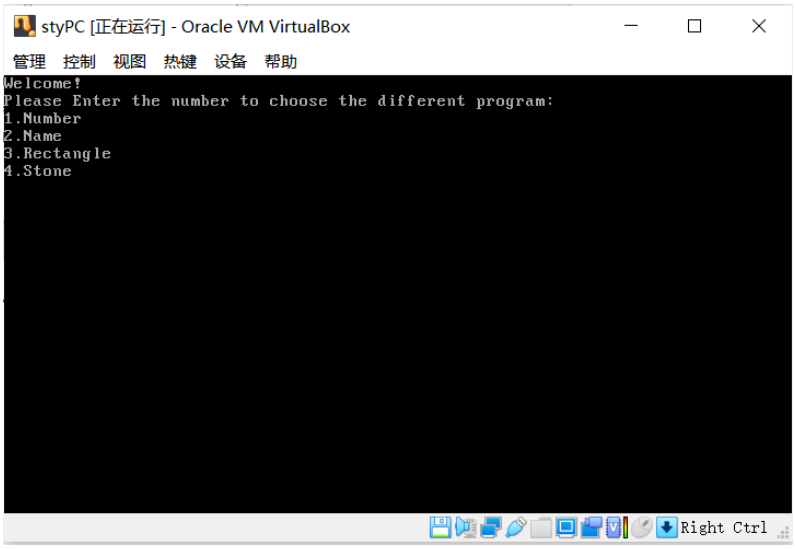




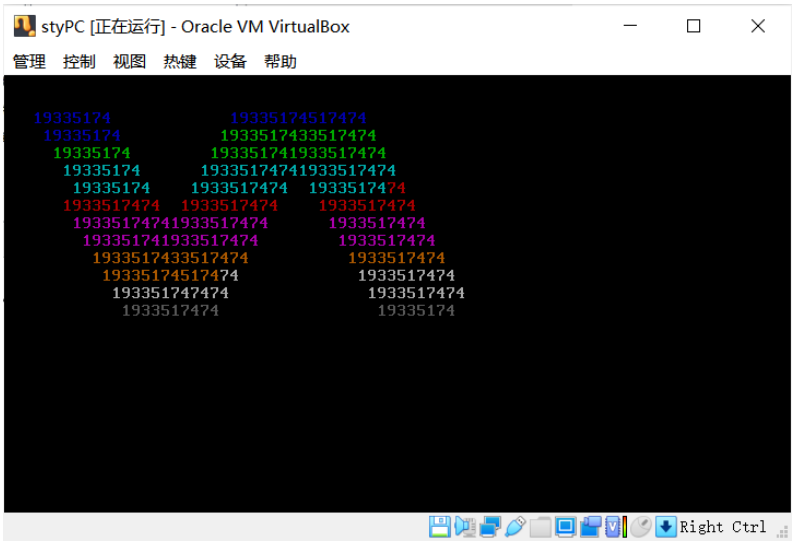
6、用软盘启动裸机



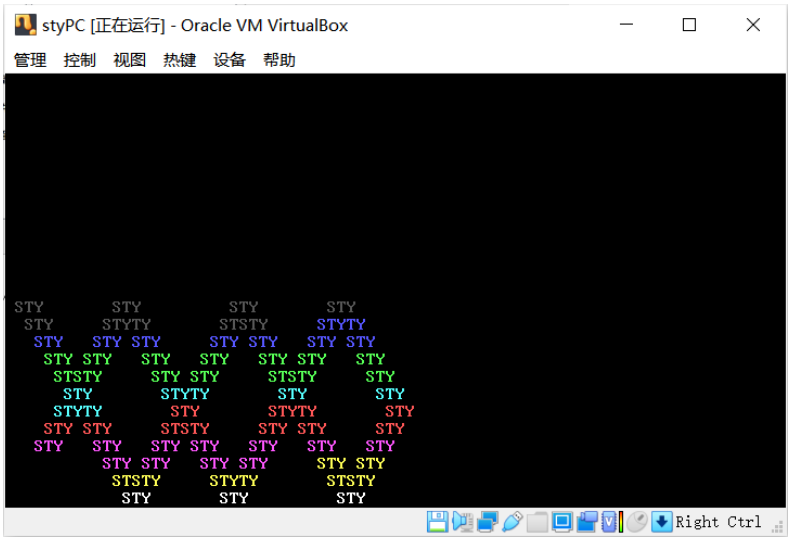
监控程序



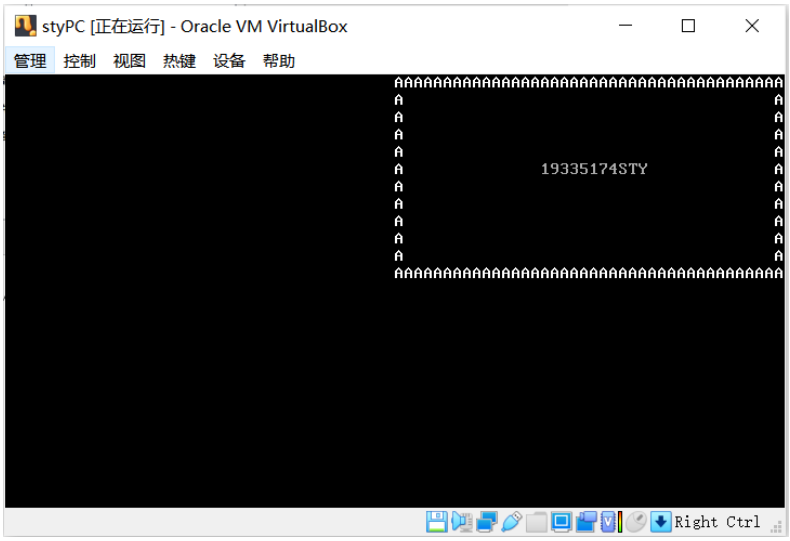
用户程序 1



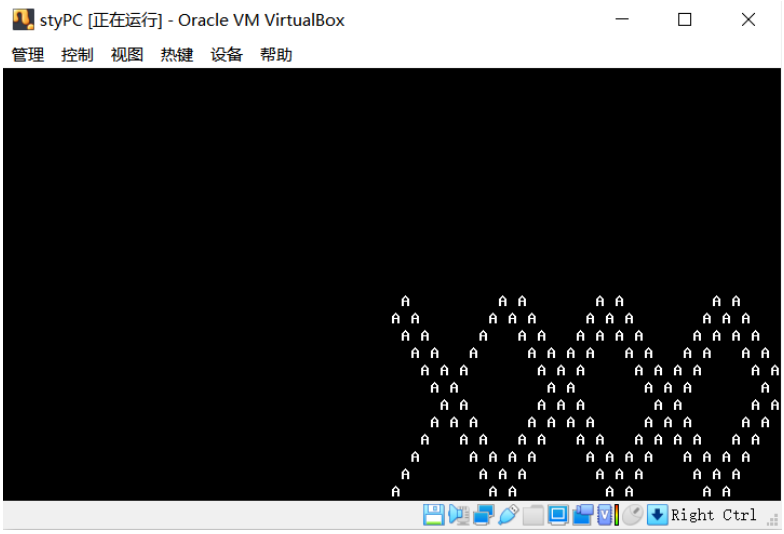
用户程序 2



用户程序 3



用户程序 4



六、实验总结

在这次实验中，我对软盘驱动裸机有了更深入的了解，自己写一个监控程序和几个用户程序，并且如何调用用户程序和在调用完成时返回是我一开始困惑的地方。我也明白了如何用 int 16 的功能，int 10 输出字符串，也明白了 jmp 更多的用处。

在实验过程中，我也遇到了几个问题。比如我一开始以为 0x55 和 0xaa 是每个扇区末尾都要加的，导致我的程序出错，后来我才知道这只是引导扇区所要求的。在设置字符串的位置时，我一开始把 mov dh，05h 和 mov dl，37h 合并成了 mov dx 0537h，却不能正常显示字符串，也不知道是什么原因。在设计用户程序如何返回监控程序时，一开始不知道如何操作，后来发现老师 ppt 有 int 16 的 1 号功能可以读键盘才设计出来，但在返回时，我一开始想用 ret 指令却不能正常返回，后来想起可以用 jmp 7c00h 才完成了这个功能。希望这些我现在还有疑问的问题在日后我都能顺利解决！

虽然整个实验让我精疲力竭，在一次次编译和调试失败时我都痛苦万分，但俗话说“功夫不负有心人”，经过长时间的努力，收获成功时的我也是快乐无比的。希望在后面的实验中，我也能再接再厉，继续加油，勇往直前！

七、参考文献

《汇编语言 （第 3 版）》

《X86 汇编语言：从实模式到保护模式》

https://blog.csdn.net/mirage1993/article/details/29908929

https://blog.csdn.net/iostream992/article/details/83077838