

实验2：加载执行COM格式用户程序的监控程序

院系	专业	年级	姓名
数据科学与计算机学院	人工智能与大数据	2018级	陈琮昊

一、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

二、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

三、实验方案：

1.运行环境：

该实验在Windows下运行。

所需软件：VMware Workstation 15.5 Pro 、 NASM 2.14.02 、 WinHex 19.8

虚拟机VMware 用来运行程序，NASM 用来生成机器码，WinHex 用来将得到的多个bin文件一并写入虚拟软盘。

2.实验流程：

- 1.准备好相关的软硬件
- 2.编写myos.asm程序，该程序为引导程序
- 3.将实验一的程序改写成不同的版本，用来测试监控程序

4.用NASM将.asm转为.bin

5.用winHex将几个.bin一并写入虚拟软盘

6.导入虚拟机运行并观察结果

3.程序分析:

根据要求,将实验一的程序改写成3个不同的版本,分别为stonea.asm、stoneb.asm、stonec.asm,其中stonea.asm的功能是在显示区域的左半侧反弹白色a,显示200个a后自动结束;stoneb.asm的功能是在显示区域的右半侧反弹蓝色b,显示200个b后自动结束;stonec.asm的功能是动态显示自己的个人学号和姓名。myos.asm为监控程序。下面分析myos.asm和stonec.asm两个程序:

(1) 监控程序:

```
;程序源代码 (myos.asm)
org 7c00h          ; BIOS将把引导扇区加载到0:7c00h处, 并开始执行
OffsetOfUserPrg1 equ 8100h    ;第一个用户程序加载位置
OffsetOfUserPrg2 equ 8600h    ;第二个用户程序加载位置
OffsetOfUserPrg3 equ 9100h    ;第三个用户程序加载位置
Start:
;清屏
mov ah,06h
mov al,0
mov ch,0
mov cl,0
mov dh,24
mov dl,79
mov bh,0fh
int 10h
    mov ax, cs          ; 置其他段寄存器值与CS相同
    mov ds, ax          ; CS=DS
    mov bp, Message      ; BP=当前串的偏移地址
    mov ax, ds           ; ES:BP = 串地址
    mov es, ax           ; ES=DS
    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
    mov ah,0
    int 16h
;利用软中断10H的0x0e号功能回显字符
mov ah,0eh
mov bl,0
int 10h              ; 调用10H号中断
cmp al,'a'           ; 键入a则执行第一个用户程序
je LoadnEx
cmp al,'b'           ; 键入b则执行第二个用户程序
je LoadnSecond
cmp al,'c'           ; 键入c则执行第三个用户程序
je LoadnThird
```

LoadnEx:

;读软盘或硬盘上的若干物理扇区到内存的ES:BX处:

```
mov ax,cs                ;段地址 ; 存放数据的内存基地址
mov es,ax                ;ES=CS
mov bx, OffsetOfUserPrg1 ;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,2                 ;起始扇区号 ; 起始编号为1
int 13H                  ;调用读磁盘BIOS的13h功能
; 第一个用户程序已加载到指定内存区域中
jmp OffsetOfUserPrg1
```

LoadnSecond:

```
mov ax,cs
mov es,ax
mov bx,OffsetOfUserPrg2
mov ah,2
mov al,1
mov dl,0
mov dh,0
mov ch,0
mov cl,3
int 13H
```

;第二个用户程序已加载到指定内存区域中

```
jmp OffsetOfUserPrg2
```

LoadnThird:

```
mov ax,cs
mov es,ax
mov bx,OffsetOfUserPrg3
mov ah,2
mov al,1
mov dl,0
mov dh,0
mov ch,0
mov cl,4
int 13H
```

;第三个用户程序已加载到指定内存区域中

```
jmp OffsetOfUserPrg3
```

AfterRun:

```
jmp $ ;无限循环
```

Message:

```
db 'Hello, Myos is loading .which program do you want to run?'
```

MessageLength equ (\$-Message)

```
times 510-($-$$) db 0
```

```
db 0x55,0xaa
```

;交互命令, 在初始界面会显示这样一段话, 通过输入a/b/c来执行相应程序

(2) `stonec.asm` 的关键代码:

;显示个人信息

ifm:

```
    dec     word[count]          ; 递减计数变量
    jnz     ifm                  ; >0: 跳转;
    mov     word[count],delay
    dec     word[dcount]         ; 递减计数变量
    jnz     ifm
    mov     word[count],delay
    mov     word[dcount],ddelay
;用简单的双循环控制时延, 这里时延50000x580个时间单位
```

```
    mov     ax, DisMessage      ;信息
    mov     bp, ax
    mov     cx, [strlen]        ;字符串长
    mov     ax, 01301h          ;写模式
    mov     bx, 000fh           ;页号0, 黑底白字
    mov     dh, [much]          ;行=b
    mov     dl, [much]          ;列=b
    int     10h                 ;10h号接口
    dec     byte[much]
    jnz     ifm
```

int 10h ; BIOS的10h功能: 显示一行字符

```
    mov ax, cs          ; 置其他段寄存器值与CS相同
    mov ds, ax          ; 数据段
    mov bp, Messageend  ; BP=当前串的偏移地址
    mov ax, ds          ; ES:BP = 串地址
    mov es, ax          ; 置ES=DS
    mov cx, MessageendLength ; 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
    mov ah,0
    int 16h
;利用软中断10H的0x0e号功能回显字符
    mov ah,0eh
    mov bl,0
    int 10h             ; 调用10H号中断
    cmp al,'e'          ; 输入e返回初始界面
    jz goback
```

goback:

```
    mov ax,cs           ;段地址 ; 存放数据的内存基地址
    mov es,ax           ;设置段地址
    mov bx, OffsetOfMyos ;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,1          ;起始扇区号 ; 起始编号为1
int 13H           ;调用读磁盘BIOS的13h功能
jmp OffsetOfMyos

```

```

DisMessage:      db      "ChenConghao 18340013"          ;个人信息内容
StrLen           dw      20
delay            equ     50000
ddelay          equ     580
count            dw      delay
dcount           dw      ddelay
much             db      24                              ;显示个数
Messageend:      db
MessageendLength equ ($-Messageend)
;第三个程序执行后显示交互命令，键入e则可返回初始界面

```

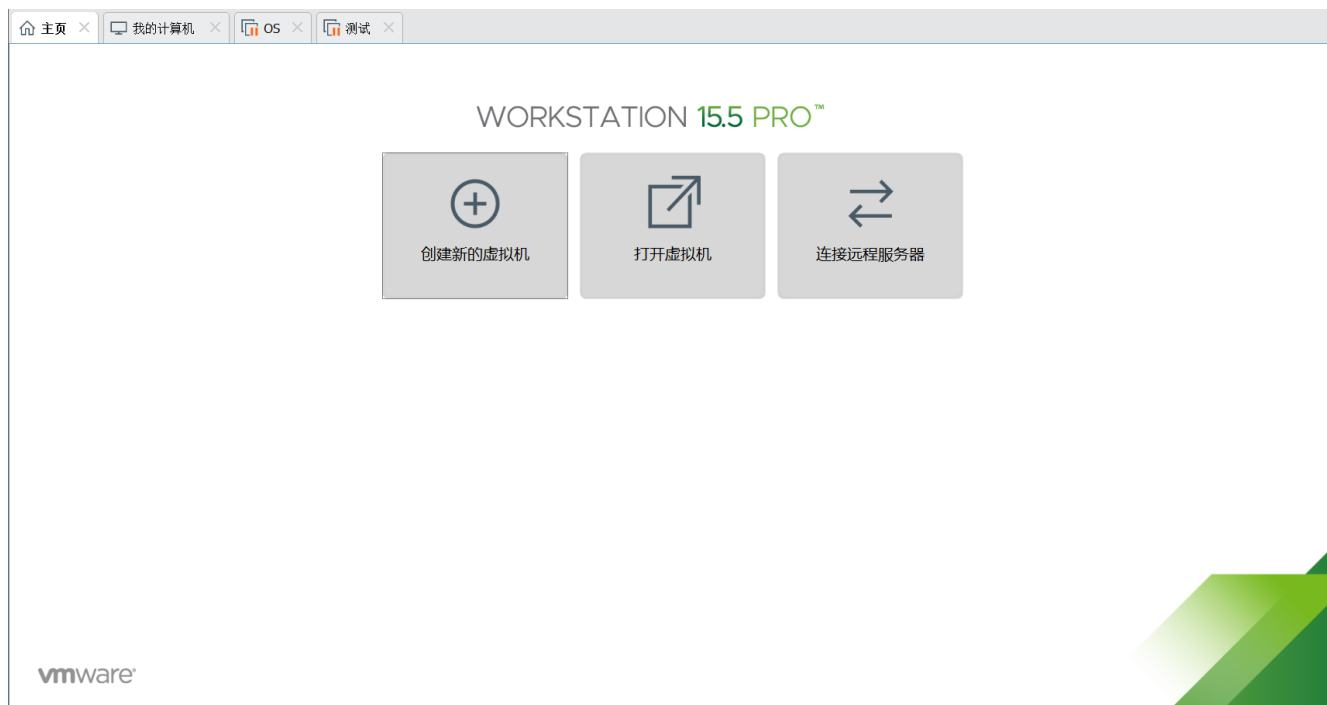
四、实验过程：

1.软件的下载与安装：

在官网下载 VMware Workstation 15.5 Pro：

<https://www.vmware.com/cn/products/workstation-pro/workstation-pro-evaluation.html>

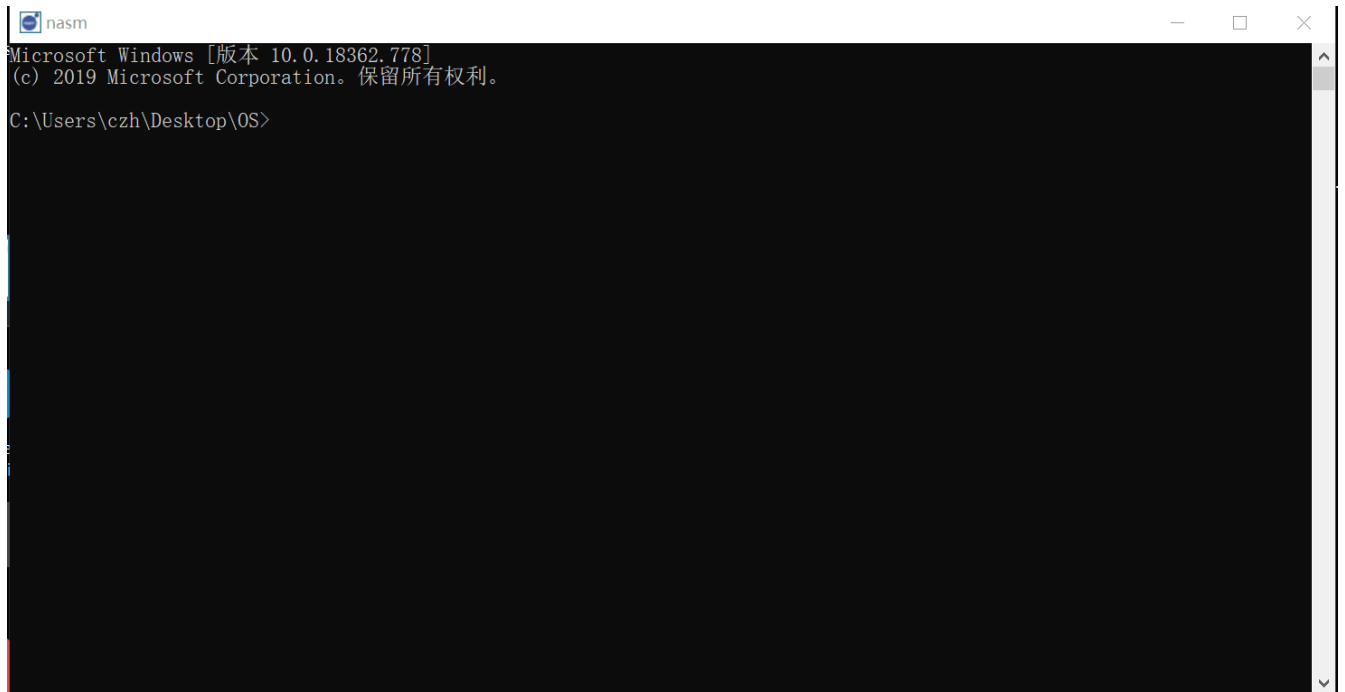
安装该软件后初始界面如下（已经输入密钥并初步设置后）：



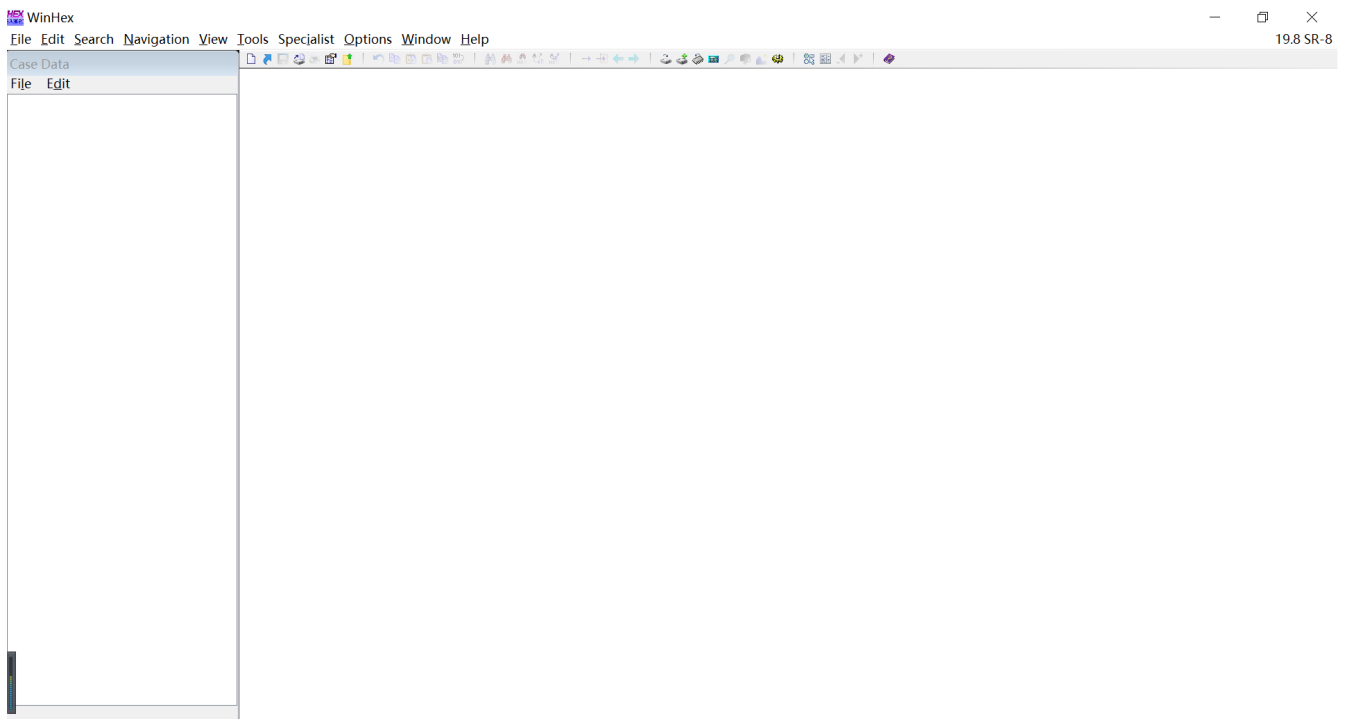
在官网下载 NASM 2.14.02：

<https://www.nasm.us/>

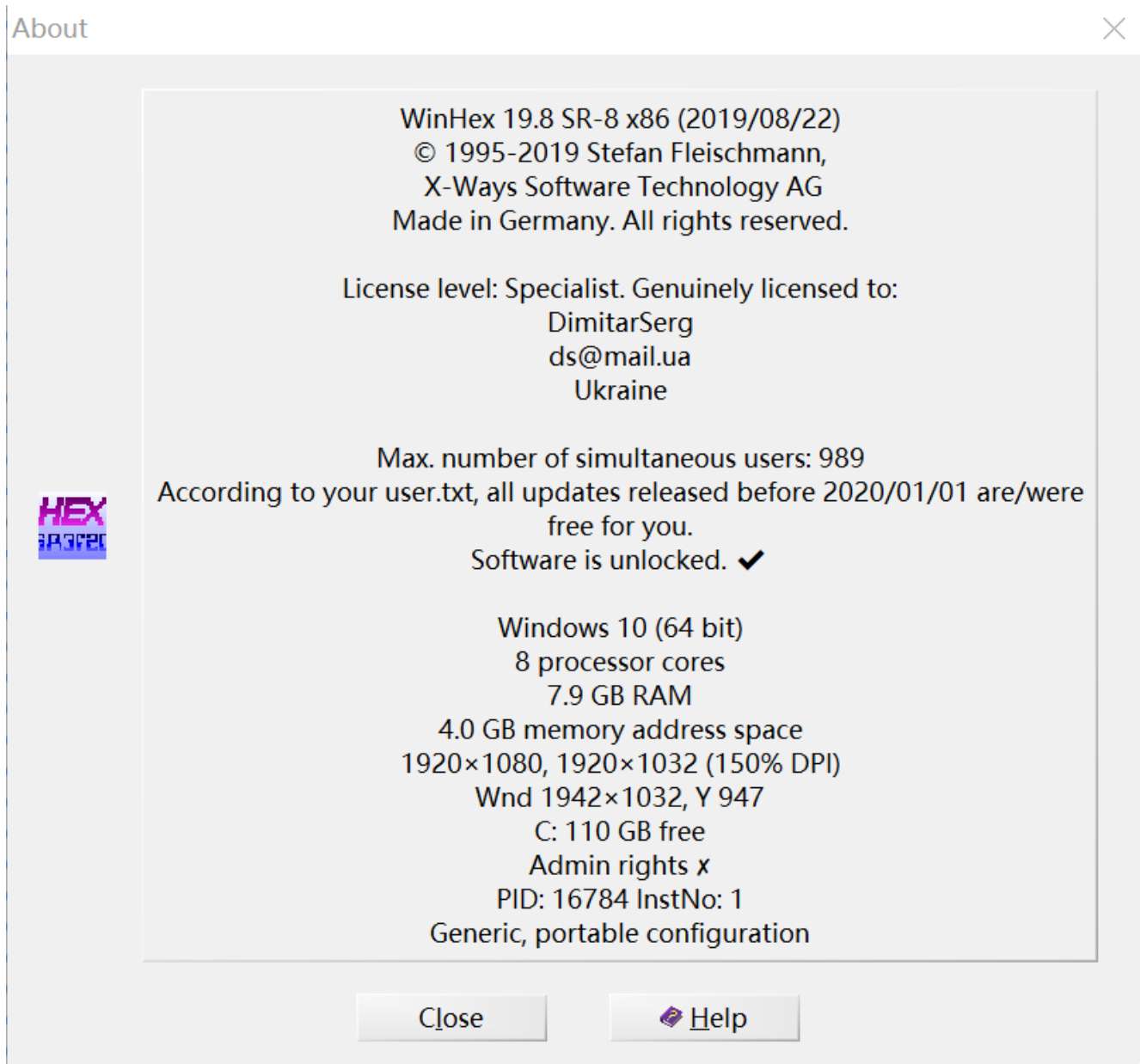
安装后打开 NASM 的界面如下：



下载 WinHex 19.8 压缩包，解压后找到 winhex.exe，打开的界面如下：



右上角可以看到版本为19.8，光标点击 19.8 SR-8 可以看到该软件还未解锁，使用起来会受限（比如不能保存超过 200KB 的文件）。因此在 Help 一栏里选择 Register，可以看到需要输入一个6行的注册码。在网上找到他人分享的注册码，输入并注册成功后该软件便被解锁，可以正常使用：



2.编写汇编代码：

需要编写4个汇编代码，1个是引导程序，另外3个是监控程序，其中引导程序为 `myos.asm`，3个监控程序分别为 `stonea.asm`、`stoneb.asm` 和 `stonec.asm`，3个监控程序是根据实验一的代码进行改动来实现相关的功能。具体代码内容见 `src` 文件夹。









3.配置相关文件：

在这一步，将前面几个存有汇编代码的 `.asm` 文件放入 `NASM` 的工作目录下，然后打开 `NASM`，敲入如下指令：

```
nasm
Microsoft Windows [版本 10.0.18362.778]
(c) 2019 Microsoft Corporation。保留所有权利。

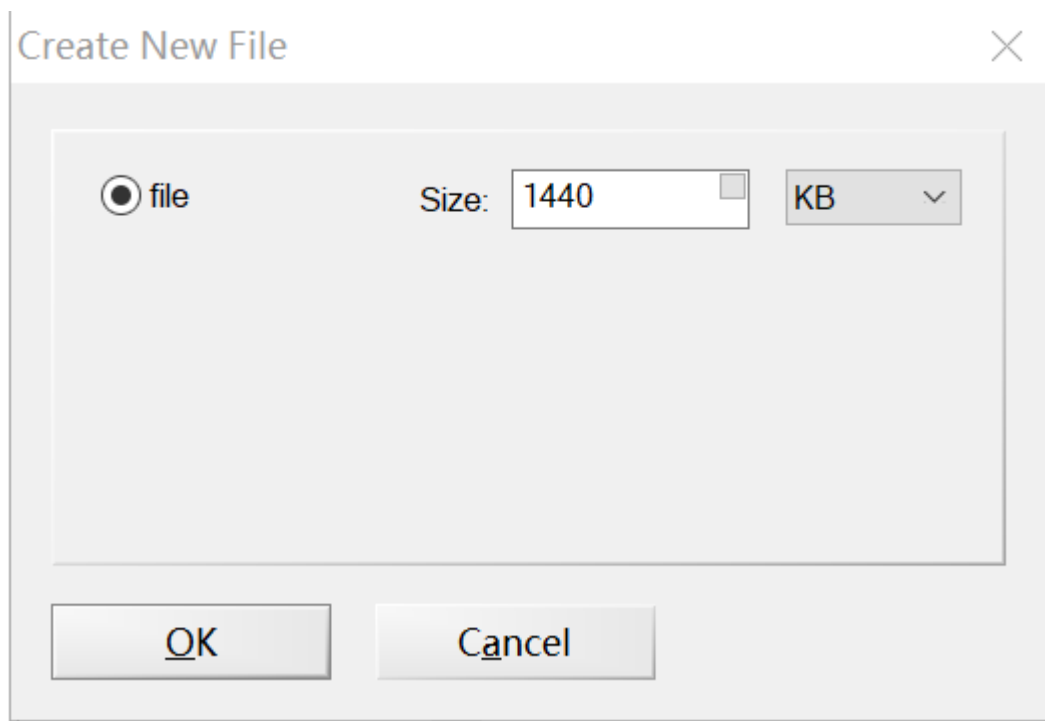
C:\Users\czh\Desktop\OS>nasm -f bin stonea.asm -o stonea.bin
C:\Users\czh\Desktop\OS>nasm -f bin stoneb.asm -o stoneb.bin
C:\Users\czh\Desktop\OS>nasm -f bin stonec.asm -o stonec.bin
C:\Users\czh\Desktop\OS>nasm -f bin myos.asm -o myos.bin
C:\Users\czh\Desktop\OS>
```

接下来可以在和 NASM 相同的目录下找到上述四个 .bin 文件:

	stoneb.bin	2020/5/7 10:46	BIN 文件	1 KB
	stonea.bin	2020/5/7 10:46	BIN 文件	1 KB
	stonec.bin	2020/5/7 10:46	BIN 文件	1 KB
	stonec.asm	2020/5/7 10:45	ASM 文件	4 KB
	stonea.asm	2020/5/7 10:40	ASM 文件	4 KB
	stoneb.asm	2020/5/7 10:39	ASM 文件	4 KB
	myos.bin	2020/5/7 10:27	BIN 文件	1 KB
	myos.asm	2020/5/7 14:21	ASM 文件	3 KB

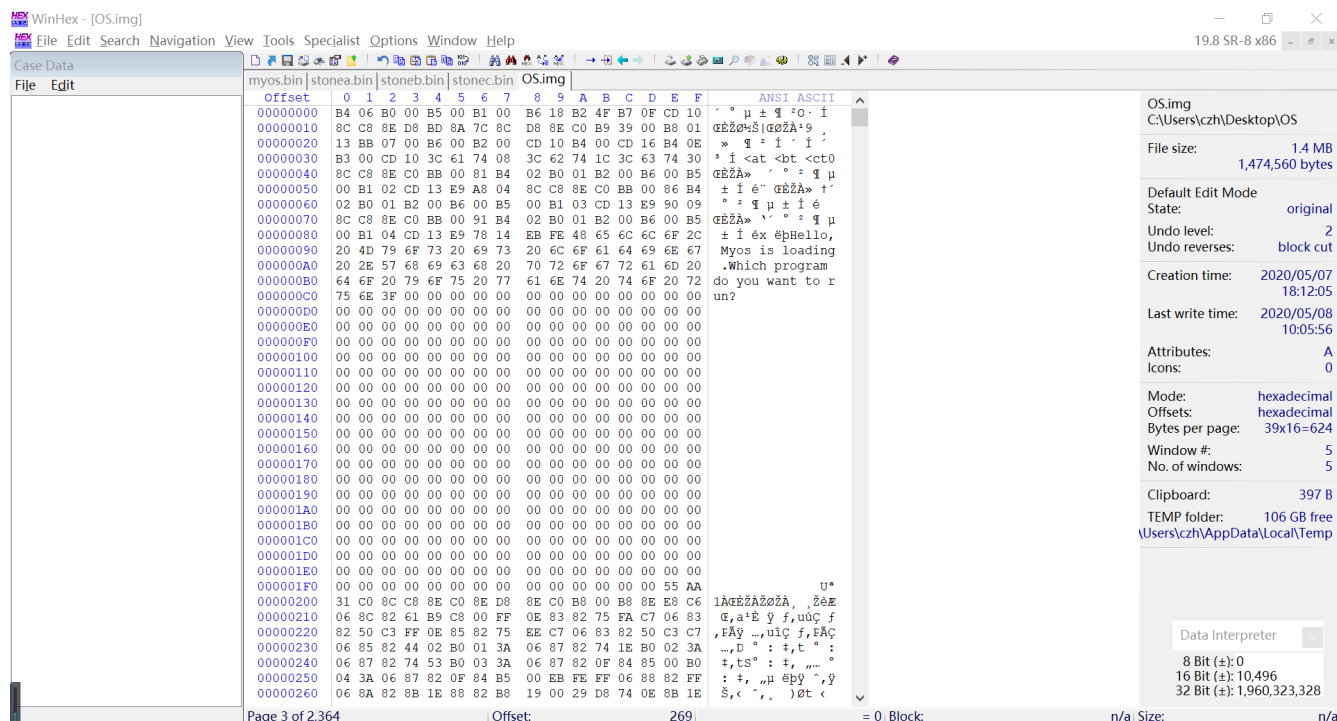
至此，就将几个汇编代码编译成了机器代码，接下来就是要将这些文件导入一个镜像文件中:

打开 WinHex，首先新建一个文件: File->New，然后选择 1440KB，该文件就是最终要的镜像文件。



然后在 winHex 内打开上述四个 .bin 文件，然后将这四个文件按照如下顺序复制粘贴至新建的文件中：

myos.bin -> stonea.bin -> stoneb.bin -> stonec.bin。在粘贴过程中需要注意，第一个粘贴的起始位置为 0x000，第二个为 0x200，第三个为 0x400，第四个为 0x600。粘贴完成后，保存并命名为 os.img，至此便得到本实验所需的镜像文件：



OS.img	2020/5/7 14:37	光盘映像文件	1,440 KB
--------	----------------	--------	----------

接下来便是将该文件导入虚拟机并运行。

4.导入虚拟机：

打开虚拟机，选择 创建新的虚拟机 一项，然后进行如下配置：

安装客户机操作系统

虚拟机如同物理机，需要操作系统。您将如何安装客户机操作系统？

☒ 稍后安装操作系统(S)。

创建的虚拟机将包含一个空白硬盘。

帮助

< 上一步(B)

下一步(N) >

取消

选择客户机操作系统

此虚拟机中将安装哪种操作系统？

客户机操作系统

☐ Microsoft Windows(W)

☐ Linux(L)

☐ VMware ESX(X)

☒ 其他(O)

版本(V)

其他

▼

帮助

< 上一步(B)

下一步(N) >

取消

于是所使用的虚拟机也已配置完成。（后续步骤中选择磁盘容量可视情况而定，其他步骤则不赘述了。）

在上一步中，已经通过 `winHex` 得到了最终的 `os.img` 镜像文件，接下来就是将 `os.img` 文件导入虚拟机，步骤如下：

打开已创建好的虚拟机，然后选择 `编辑虚拟机设置`，然后添加 `软盘`，在 `连接` 一栏选择 `使用软盘映像文件`，将 `os.img` 导入，`确定` 即可。

设备状态

☐ 已连接(C)

☒ 启动时连接(O)

连接

☐ 使用物理驱动器(P):

自动检测

☒ 使用软盘映像文件(M):

C:\Users\czh\Desktop\OS\OS.img

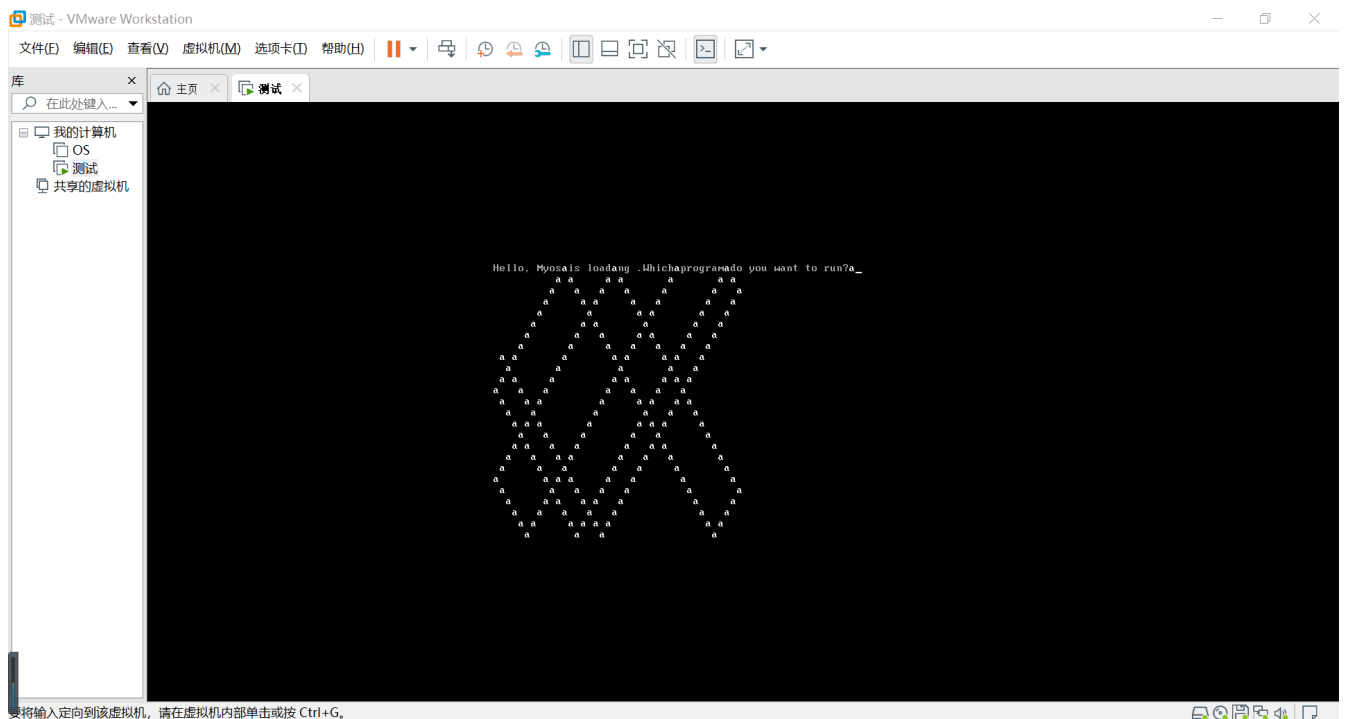
创建(I)... 浏览(B)...

☐ 只读(E)

5.运行显示:

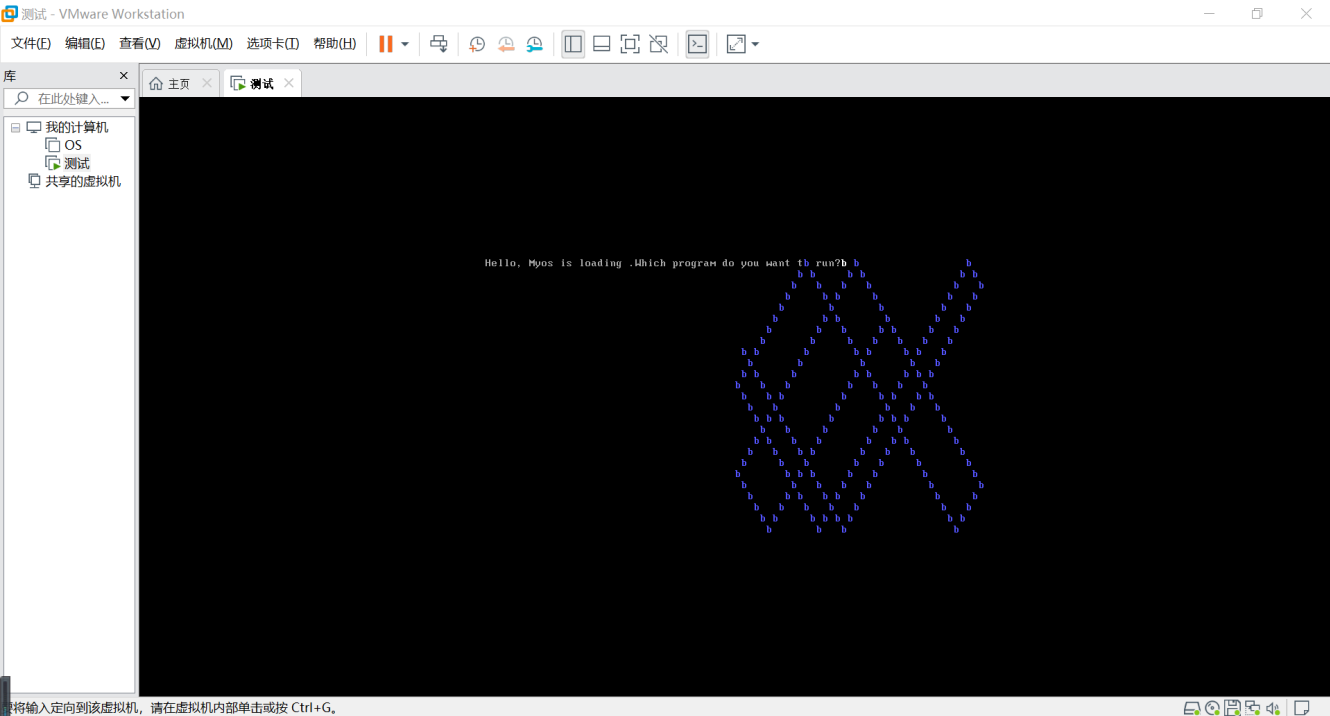
添加软盘完成后，打开虚拟机，从虚拟软盘启动后即可看到如下结果：

(1) 键入a：



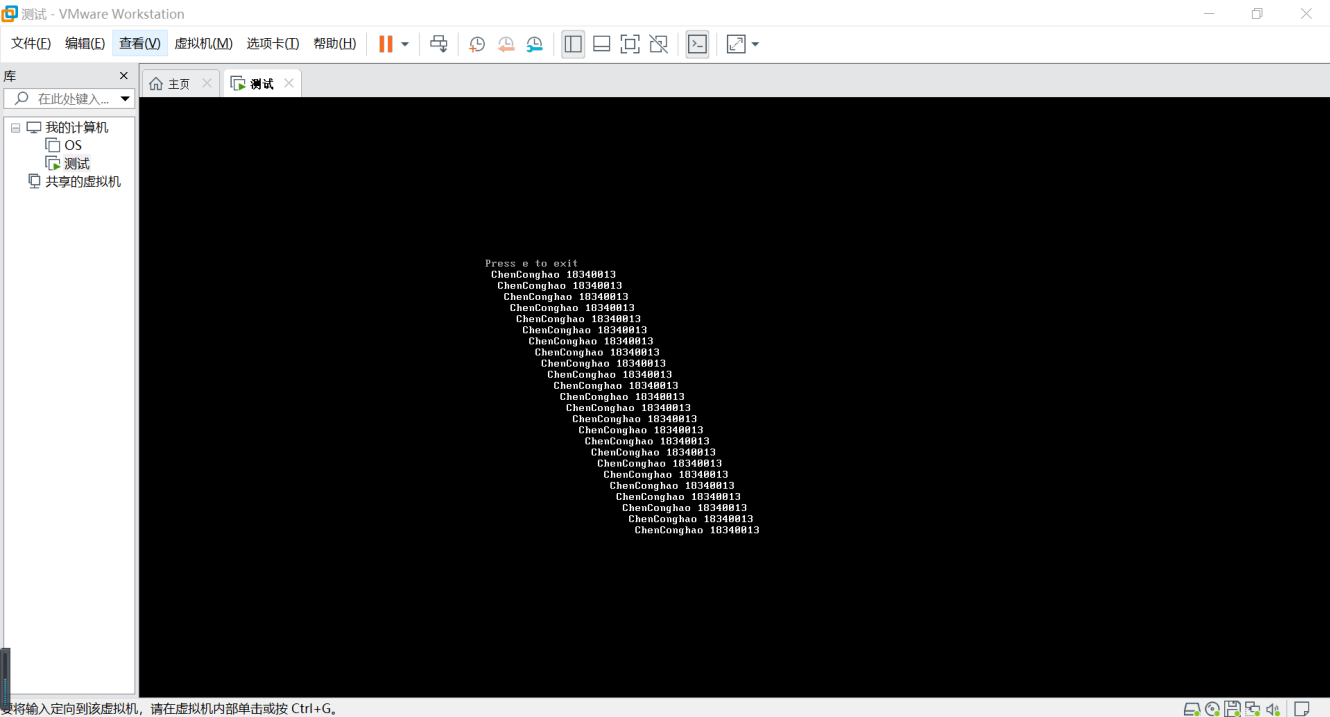
可以看到，成功在显示区域的左半部分反弹白色a。

(2) 键入**b**：



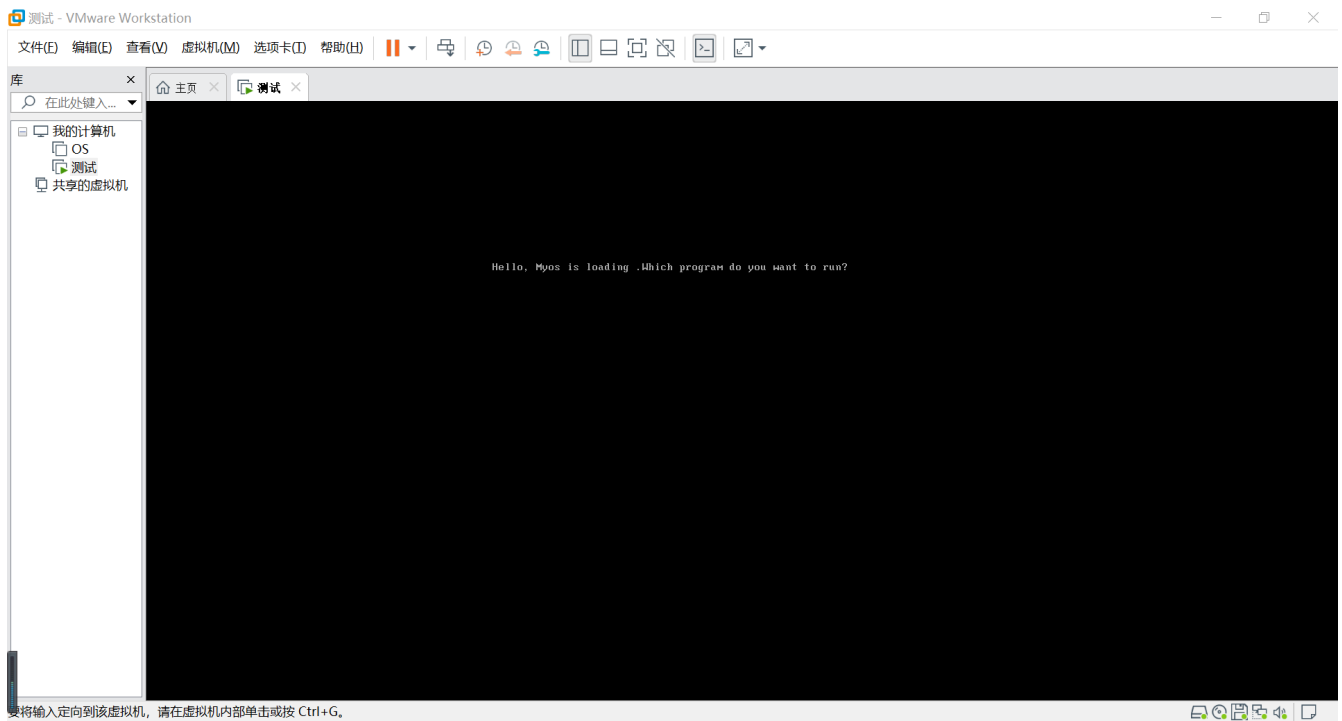
可以看到，成功在显示区域的右半部分反弹蓝色b。

(3) 键入**c**：



可以看到，成功显示了24行的个人信息。

(4) 键入**e**：



可以看到，输入e之后返回到了初始界面。

五、实验总结：

本次实验从操作的流程上来说，和上次实验基本相同，都是编写好代码、得到机器码、生成镜像文件，最后导入虚拟机运行。只不过代码和上次有所不同，这次要写1个引导程序和3个监控程序，当然3个监控程序可以根据上次实验的代码来改，在改动的过程中一定要注意几个程序之间的协调，否则结果就是跑崩了。生成镜像文件一步也有些不同，这次要把几个文件联合得到一个镜像文件。在这个实验过程中，让我耗费时间最多的地方就是生成镜像文件这个环节。上一次实验是直接将一个 .bin 文件变成 .img 文件，可以说很轻松；这一次是要将多个 .bin 文件导入一个 .img 文件，通过查找网上的相关资料，需要用 WinHex 来完成这一步。于是下载好 WinHex 以后将机器码按照顺序复制到 img 文件里，在这个过程中需要注意的一点就是：写入第一个 .bin 文件从 0x000 开始写，第二个要从 0x200 开始写，第三个要从 0x400 开始写.....这个过程完成以后，就可以得到 1.44MB 的虚拟软盘。就在这时麻烦来了：“文件太大不让保存”。结果百度一下才发现原来是没有注册码的原因。于是就上网找到了一位他人分享的注册码，输入并注册成功后就可以保存了。后续步骤就简单了，最终成功实现了实验二。刚刚拿到这个实验任务的时候可以说是“云里雾里”，完全搞不明白它究竟是在干什么。后来通过查阅老师给的参考资料以及上网搜索相关资料才弄清楚实验二整个的这样一个原理和过程。整个过程耗费了不少的时间，因为有些东西第一次接触非常难搞，而且参考资料对于这部分内容几乎都是直接跳过，所以还要花时间去找资料，有些内容从直观上无法理解，还要进行更多的尝试才能逐渐摸透，搞得我一度不想进行实验了。不过后来还是重振旗鼓把这个实验做完了，做完那一刻感觉整个人都好了，希望自己下次能够提高效率吧。

六、参考文献：

李忠，王晓波，余洁.《x86汇编语言：从实模式到保护模式》.电子工业出版社，2012.

凌应标. 01实验课 (1) .pptx

凌应标. BINCOM.docx

<https://blog.csdn.net/blizmax6/article/details/9474601>

<https://blog.csdn.net/xssywsh/article/details/88616120>