

# 操作系统实验报告

---

17341190 叶盛源

2019 年 3 月 22 日

## Contents

1 实验题目	2
2 实验目的	2
3 实验要求	2
4 实验方案	3
实验环境: . . . . .	3
实验工具: . . . . .	3
实验思想及实验过程: . . . . .	3
5 实验心得和总结:	8
参考文献	9

## 1 实验题目

- 1)搭建和应用实验环境
- 2)接管裸机的控制权

## 2 实验目的

- 1)安装虚拟机软件VMware，生成多个虚拟软盘，将其中一个用DOS格式化为DOS引导盘
- 2)学会使用虚拟机软件创建裸机模拟IMB-PC环境，同时生成一个大小为1.44MB软盘。学习使用汇编语言写一个程序，利用WinHex工具和NASM将程序转换成二进制数据存入生成的软盘第一个扇区中，并使用首扇区作为引导程序使裸机在开机的时候运行。

## 3 实验要求

- 1)生成一个基本配置的虚拟机XXXPC和多个1.44MB容量的虚拟软盘，将其中一个虚拟软盘用DOS格式化为DOS引导盘，用WinHex工具将其中一个虚拟软盘的首扇区填满你的个人信息。
- 2)设计IBM\_PC的一个引导扇区程序，程序功能是用字符从屏幕左边某行位置45度角下斜射出，保持一个可观察的适当速度直线运动，碰到屏幕的边后产生反射，改变方向运动，如此类推，不断运动；在此基础上，增加你的个性扩展，如同时控制两个运动的轨迹，或炫酷动态变色，个性画面，如此等等，自由不限。还要在屏幕某个区域特别的方式显示你的学号姓名等个人信息。将这个程序的机器码放进放进虚拟软盘的首扇区，并用此软盘引导你的XXXPC，直到成功。

## 4 实验方案

### 实验环境：

- 1) 实验运行环境：Windows10
- 2) 虚拟机软件：VMware Function

### 实验工具：

- 1) 汇编语言：NASM
- 2) 文本编辑器：VScode
- 3) 软盘操作工具：WinHex

### 实验思想及实验过程：

- 1) 安装主要工具：

在VMware官网下载虚拟机的体验版安装包，再在网上找到破解的密钥输入永久破解。接着按照要求安装并创建一个无操作系统的裸机，如下图所示：

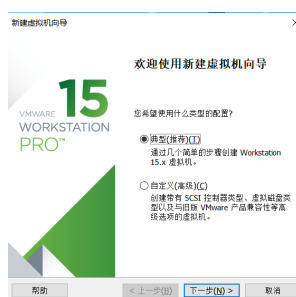


Figure 1: 安装VMware虚拟机

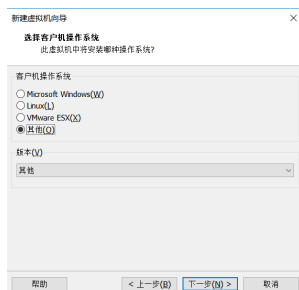


Figure 2: 创建无操作系统裸机



Figure 3: 创建无操作系统裸机

## 2) DOS格式化建立虚拟软盘：

在网上下载DOS操作系统镜像文件，进入DOS的BIOS界面进行设置并格式化建立的软盘，使得只有首扇区留下DOS的引导程序，再用这个软盘启动上一步建立的裸机，DOS操作系统界面和格式化后的软盘打开裸机后的画面如下图：

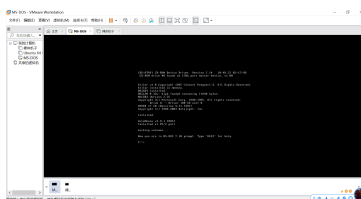


Figure 4: DOS操作系统启动界面

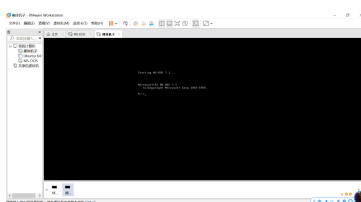


Figure 5: 格式化后的DOS引导盘启动裸机

之后利用Winhex工具将其中一个软盘personalInfor的首扇区512个字节填满个人信息。每个字节放我的学号的一位，刚好放满512字节，如下图：

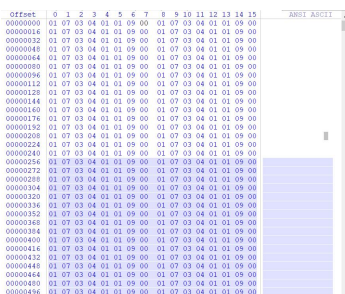


Figure 6: WinHex填个人资料

### 3) 导入程序指令：

我们需要将自己编写的汇编指令转换为二进制后放在软盘的首扇区，但裸机启动的时候，会自动调用首扇区中的内容，然后我们需要利用指令**org 7c00**让裸机将代码存入内存**7c00**中，这样可以确保不影响前面内存中存放的重要内容。

### 4) 设计程序：

#### i) 展示学生信息：

利用bios提供的中断int10的13h号功能,实现在显示器上输出字符串。bx可以设置显示姓名序号的前景色背景色等属性,代码如下图:

```
showData:
    mov ax,Name
    mov bp,ax ; es:bp串地址
    mov cx,13
    mov ax,1301h ; ah=13 显示字符串
    mov bx,101ah ;b1 6-4背景色 3位1前景色高亮 2-0 前景色RGB
    mov dx,0000h
    int 10h

    mov ax,Number
    mov bp,ax
    mov cx,8
    mov ax,1301h
    mov bx,101bh
    mov dx,0100h
    int 10h
```

Figure 7: 显示学生信息

#### ii) 编写弹球程序：

已知裸机的屏幕大小为25\*80，当我们把值传到**B800**开始

内存(显存), 值所对应的的ASCII码就会显示在屏幕上, 每个字符占两个字节, 一个是ASCII码一个是属性, 利用这个我们可以实现弹球的显示。我们将四个方向定义为常数1到4。同时为小球设置一个初始点, 并默认开始时候向右上方移动。小球每移动一格, 利用类似switch case, 判断是否需要改变移动方向, 并用jmp指令跳转到show函数在屏幕上显示, 每次显示一个字符之后, 就让字符显示的属性值加一个数, 使得弹珠呈现动态变化。举例子如下:

```
; 类似switch case语句
mov al,1
cmp al,byte[rdu1]
jz DnRt
mov al,2
cmp al,byte[rdu1]
jz UpRt
mov al,3
cmp al,byte[rdu1]
jz UpLt
mov al,4
cmp al,byte[rdu1]
jz DnLt
```

Figure 8: 类switch case语句

```
DnRt:
inc word[x]
inc word[y]
mov bx,word[x]
mov ax,25 ;[125]
sub ax,Rt
jz dr2ur
mov bx,word[y]
mov ax,88 ;每行88个字符
sub ax,Rt
jz dr2dl
jmp show

dr2ur:
mov word[x],25
mov byte[rdul],Up_Rt
jmp show

dr2dl:
mov word[y],78
mov byte[rdul],Dn_Lt
jmp show
```

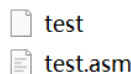
Figure 9: 右上移动示例

#### 5) 使用NASM编译程序:

打开nasm, 并输入指令nasm+文件名.asm, 编译完后在同一个目录下会生成一个同名的二进制文件, 如图:

```
D:\NASM>nasm test.asm
D:\NASM>
```

Figure 10: nasm编译



test  
test.asm

Figure 11: 生成bin文件

#### 6) 使用winHex修改软盘:

使用WinHex打开刚刚生成的test二进制文件和软盘flp文件, 将test的内容替换到WinHex的第一个扇区(512Byte)中。注意: 程序的大小不能超多512B, 否则会出错, 替换后软盘的大小也不能改变。如图:

hex[0p]	hex[1p]	test	myos1	hex[2p]	hex[3p]	hex[4p]
Offset	0	1	2	3	4	5
00000000	00	00	00	00	00	00
00000001	00	00	00	00	00	00
00000002	00	00	00	00	00	00
00000003	00	00	00	00	00	00
00000004	00	00	00	00	00	00
00000005	00	00	00	00	00	00
00000006	00	00	00	00	00	00
00000007	00	00	00	00	00	00
00000008	00	00	00	00	00	00
00000009	00	00	00	00	00	00
0000000A	00	00	00	00	00	00
0000000B	00	00	00	00	00	00
0000000C	00	00	00	00	00	00
0000000D	00	00	00	00	00	00
0000000E	00	00	00	00	00	00
0000000F	00	00	00	00	00	00
00000010	00	00	00	00	00	00
00000011	00	00	00	00	00	00
00000012	00	00	00	00	00	00
00000013	00	00	00	00	00	00
00000014	00	00	00	00	00	00
00000015	00	00	00	00	00	00
00000016	00	00	00	00	00	00
00000017	00	00	00	00	00	00
00000018	00	00	00	00	00	00
00000019	00	00	00	00	00	00
0000001A	00	00	00	00	00	00
0000001B	00	00	00	00	00	00
0000001C	00	00	00	00	00	00
0000001D	00	00	00	00	00	00
0000001E	00	00	00	00	00	00
0000001F	00	00	00	00	00	00
00000020	00	00	00	00	00	00
00000021	00	00	00	00	00	00
00000022	00	00	00	00	00	00
00000023	00	00	00	00	00	00
00000024	00	00	00	00	00	00
00000025	00	00	00	00	00	00
00000026	00	00	00	00	00	00
00000027	00	00	00	00	00	00
00000028	00	00	00	00	00	00
00000029	00	00	00	00	00	00
0000002A	00	00	00	00	00	00
0000002B	00	00	00	00	00	00
0000002C	00	00	00	00	00	00
0000002D	00	00	00	00	00	00
0000002E	00	00	00	00	00	00
0000002F	00	00	00	00	00	00
00000030	00	00	00	00	00	00
00000031	00	00	00	00	00	00
00000032	00	00	00	00	00	00
00000033	00	00	00	00	00	00
00000034	00	00	00	00	00	00
00000035	00	00	00	00	00	00
00000036	00	00	00	00	00	00
00000037	00	00	00	00	00	00
00000038	00	00	00	00	00	00
00000039	00	00	00	00	00	00
0000003A	00	00	00	00	00	00
0000003B	00	00	00	00	00	00
0000003C	00	00	00	00	00	00
0000003D	00	00	00	00	00	00
0000003E	00	00	00	00	00	00
0000003F	00	00	00	00	00	00
00000040	00	00	00	00	00	00
00000041	00	00	00	00	00	00

Figure 12: 软盘替换后内容

hex[0p]	hex[1p]	test	myos1	hex[2p]	hex[3p]	hex[4p]
Offset	0	1	2	3	4	5
00000000	00	00	00	00	00	00
00000001	00	00	00	00	00	00
00000002	00	00	00	00	00	00
00000003	00	00	00	00	00	00
00000004	00	00	00	00	00	00
00000005	00	00	00	00	00	00
00000006	00	00	00	00	00	00
00000007	00	00	00	00	00	00
00000008	00	00	00	00	00	00
00000009	00	00	00	00	00	00
0000000A	00	00	00	00	00	00
0000000B	00	00	00	00	00	00
0000000C	00	00	00	00	00	00
0000000D	00	00	00	00	00	00
0000000E	00	00	00	00	00	00
0000000F	00	00	00	00	00	00
00000010	00	00	00	00	00	00
00000011	00	00	00	00	00	00
00000012	00	00	00	00	00	00
00000013	00	00	00	00	00	00
00000014	00	00	00	00	00	00
00000015	00	00	00	00	00	00
00000016	00	00	00	00	00	00
00000017	00	00	00	00	00	00
00000018	00	00	00	00	00	00
00000019	00	00	00	00	00	00
0000001A	00	00	00	00	00	00
0000001B	00	00	00	00	00	00
0000001C	00	00	00	00	00	00
0000001D	00	00	00	00	00	00
0000001E	00	00	00	00	00	00
0000001F	00	00	00	00	00	00
00000020	00	00	00	00	00	00
00000021	00	00	00	00	00	00
00000022	00	00	00	00	00	00
00000023	00	00	00	00	00	00
00000024	00	00	00	00	00	00
00000025	00	00	00	00	00	00
00000026	00	00	00	00	00	00
00000027	00	00	00	00	00	00
00000028	00	00	00	00	00	00
00000029	00	00	00	00	00	00
0000002A	00	00	00	00	00	00
0000002B	00	00	00	00	00	00
0000002C	00	00	00	00	00	00
0000002D	00	00	00	00	00	00
0000002E	00	00	00	00	00	00
0000002F	00	00	00	00	00	00
00000030	00	00	00	00	00	00
00000031	00	00	00	00	00	00
00000032	00	00	00	00	00	00
00000033	00	00	00	00	00	00
00000034	00	00	00	00	00	00
00000035	00	00	00	00	00	00
00000036	00	00	00	00	00	00
00000037	00	00	00	00	00	00
00000038	00	00	00	00	00	00
00000039	00	00	00	00	00	00
0000003A	00	00	00	00	00	00
0000003B	00	00	00	00	00	00
0000003C	00	00	00	00	00	00
0000003D	00	00	00	00	00	00
0000003E	00	00	00	00	00	00
0000003F	00	00	00	00	00	00
00000040	00	00	00	00	00	00
00000041	00	00	00	00	00	00

Figure 13: 生成二进制文件内容

## 7) 用软盘启动裸机:

将软盘的第一个扇区替换为自己编写的汇编指令的二进制代码后，就可以开启虚拟机，选择以虚拟软盘启动，启动后虚拟机就会立刻将我们的程序加载到虚拟机的内存空间中并运行，如下图：



Figure 14: 选择软盘启动



Figure 15: 启动虚拟机

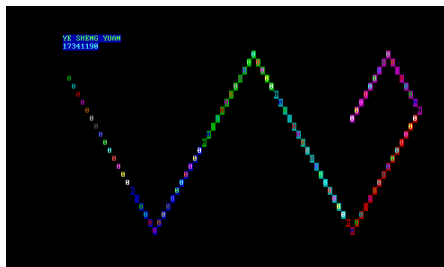


Figure 16: 程序运行画面-1

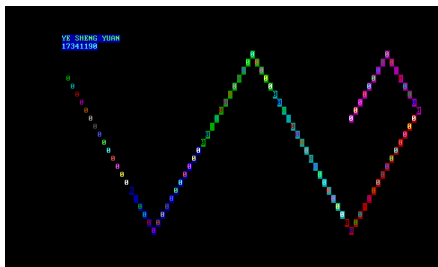


Figure 17: 程序运行画面-2

## 5 实验心得和总结：

通过这个实验，我体验了如何安装并创建虚拟机，还有创建软盘并将我写的汇编程序转化成二进制代码更改软盘的第一个扇区，再用软盘启动虚拟机，经过这个过程，我对理论课上学到的知识有了更深的体会和感悟。理论课上，老师给我们讲解了很多的知识和实验要注意的事情，但在没有亲手体验之前，十分的难理解也难记忆。但当我自己实践了一次之后我才真正的明白了很多课堂上一知半解的内容，深刻理解了“纸上得来终觉浅”的道理。

在这个实验之前，我们虽然上过计算机组成原理课，但其实当时进行的汇编练习比较少，因为实验课写汇编也是写mips的指令，所以对x86汇编十分生疏。但通过这次练习，我对汇编语言有了更深的理解和体验。刚开始我遇到了很多的困难，各种代码段看不懂，指令不明白，但最后在不断的查阅资料，和同学相互交流帮助下，渐渐的看懂了老师给的示例代码，也学会了去改参数和增删代码实现个性化的效果，最终也顺利完成了任务，终于汇编语言渐渐熟悉了起来，对这种底层的程序设计语言的理解也必定会加深我在高级语言上的编写能力，也会让我对计算机程序、系统等方面有更深理解。

这次实验过程解决了我不少以前不懂的问题。比如说为什么要加上org 7c00h，其实是因为裸机启动后内存7c00h以前都可能被一些重要的功能所占用，从7c00开始是经过测试比较稳定的，从7c00内存开始加载程序并运行。而用DOS操作系统运行.com文件是使用了指令org 100h，经过查阅知道DOS操作系统前100h的内存放了程序前缀段之类的重要代码，也是不能随便往里面加载用户的程序的，所以要从org 100h开始。还有就是裸机开始运行后首先会进入BIOS（基本输入输出系统）进行各种设置和自检查后才会启动从软盘引导扇区读取内容加载到内存并运行，之后才会进入操作系统。而我们因为是裸机没有操作系统，所以就谢了一个



简单的小程序放在引导扇区让它运行。听老师说以后会一步步慢慢实验一个自己的简单的操作系统，对此我还是充满期待的！

实验1结束了，实验2是建立在实验1只上更难的内容，制作监控程序，这对我来说是一个更大的挑战，希望有了实验1的基础，我们更快更轻松的完成实验2所要求的内容。每当完成一项实验都会充满了自豪感，让我很享受，希望能保持这个状态，继续努力！

## References

- [1] <https://blog.csdn.net/cielozhang/article/details/6171783/>
- [2] [https://blog.csdn.net/lulipeng\\_cpp/article/details/8161982](https://blog.csdn.net/lulipeng_cpp/article/details/8161982)
- [3] <https://www.cnblogs.com/alwaysking/p/7789282.html>