操作系统-实验一

学号：18340020　姓名：陈贤彪 学院：数据科学与计算机学院

1.实验目的

1、了解原型操作系统设计实验教学方法与要求
2、了解计算机硬件系统开机引导方法与过程
3、掌握操作系统的引导程序设计方法与开发工具
4、复习加强汇编语言程序设计能力

2.实验要求

设计 IBM_PC 的一个引导扇区程序，程序功能是：用字符'A'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；

1.虚拟机安装，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'A'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第三张虚拟软盘的首扇区，并用此软盘引导你的 XXXPC，直到成功。

3.实验方案

1）实验环境

a)系统：windows 10-x64

2）实验工具

a)VM VirtualBox

虚拟机软件，用于模拟虚拟不同的操作系统，也可以创建多个虚拟软盘

b)NASM-2.07

汇编语言编译器，可以将写好的.asm文件编译成二进制文件.bin

c)FloppyWrite

硬软盘写入工具：能够将写好的.bin文件写进软盘.img文件

d)文本文档

编辑修改汇编代码.asm文件

3）实验原理

a)x86汇编语言寄存器

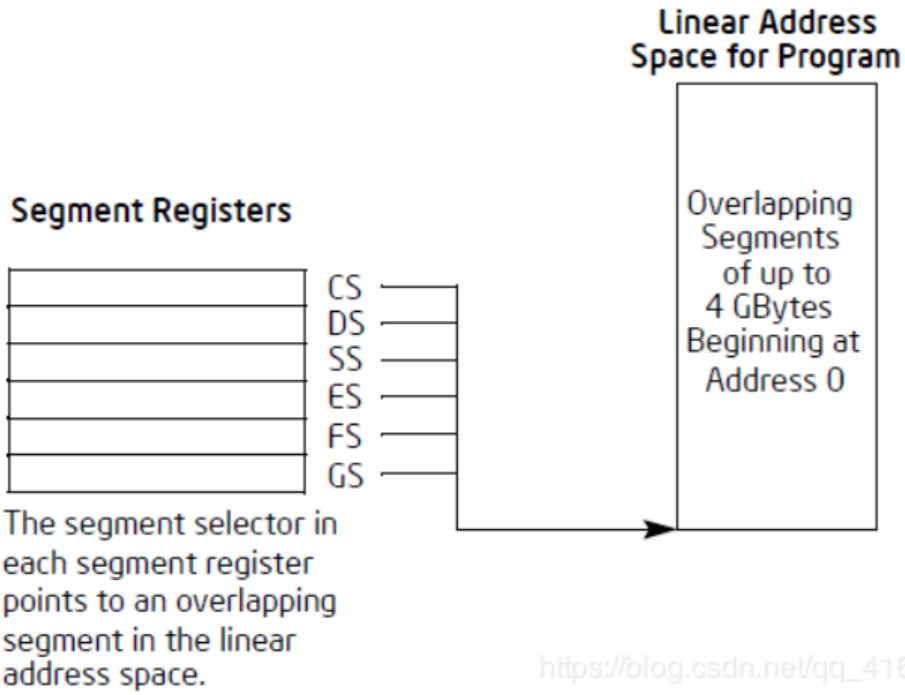
通用寄存器：

General-Purpose Registers

31 16	15 8	7 0	16-bit	32-bit
	AH	AL	AX	EAX
	BH	BL	BX	EBX
	CH	CL	CX	ECX
	DH	DL	DX	EDX
	BP			EBP
	SI			ESI
	DI			EDI
	SP			ESP

• AX/EAX — 操作数和结果的累加器
• BX/EBX — 指向数据段中的数据的指针
• CX/ECX — 用于字符串和循环操作的计数器
• DX/EDX — 输入输出指针
• SI/ESI —指向由 DS 寄存器指向的段中的数据的指针；字符串操作的源指针。也叫做源变址
• DI/EDI —指向由 ES 寄存器指向的段中的数据（或目标地址）的指针；字符串操作的目标指针。也叫做基变址
• SP/ESP — 栈顶指针
• BP/EBP — 指向栈上数据的指针，作用于函数调用的返回

段寄存器：段寄存器用于保存 16 位的段选择器。段选择器是一种特殊的指针，用于确定内存中某个段的位置。



• CS寄存器包含代码段的段选择器，其中存储了正在执行的指令。

• DS、ES、FS和GS寄存器指向四个数据段。

• SS寄存器包含堆栈段的段选择器，其中存储了当前正在执行的程序、任务或处理程序的过程堆栈。所有堆栈操作都使用SS寄存器来查找堆栈段。

标志寄存器：用于标记当前计算

• CF：进位标志，用于表示无符号数运算是否产生进位或者借位，如果产生了进位或借位则值为 1，否则值为 0。

• ZF：零标志，用于表示运算结果是否为 0，结果为 0 时其值置1，否则置 0。

• SF：符号标志，用来标记有符号数运算结果是否小于 0，小于 0时置 1，否则置 0。

• OF：溢出标志，用于表示有符号运算结果是否溢出，发生溢出时置 1，否则置 0。

• DF：方向标志，决定字符串操作指令执行时指针寄存器的调整方向

b)内存和寻址模式

静态数据声明：可以在X86汇编语言中用汇编指令.DATA声明静态数据区（类似于全局变量），数据以单字节、双字节、或双字（4字节）的方式存放，分别用DB,DW, DD指令表示声明内存的长度。

简单指令：

mov：mov指令将第二个操作数（可以是寄存器的内容、内存中的内容或值）复制到第一个操作数（寄存器或内存）。

push：push指令将操作数压入内存的栈中，栈是程序设计中一种非常重要的数据结构，其主要用于函数调用过程中，其中ESP只是栈顶。

pop：pop指令与push指令相反，它执行的是出栈的工作。它首先将ESP指示的地址中的内容出栈，然后将ESP值加4.

add：add指令将两个操作数相加，且将相加后的结果保存到第一个操作数中。

inc，dec：inc,dec分别表示将操作数自加1，自减1

jmp：控制转移到label所指示的地址

cmp：cmp指令比较两个操作数的值，并根据比较结果设置机器状态字中的条件码。

d)主引导程序

"主引导记录"只有512字节,作用是告诉操作系统到硬盘的哪个位置查找操作系统,并启动 主引导记录分成三个部分:
1.第1-446字节:调用操作系统的程序操作码

2.第447-510字节:分区表

3.第511-512字节:主引导记录的签名(看是否有0x55和0xAA)

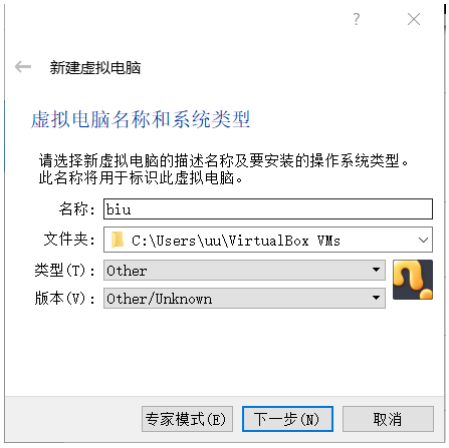
4）实验思路

4.实验过程

1）虚拟机的配置

虚拟机我使用的是VM VirtualBox，首先新建一个虚拟pc

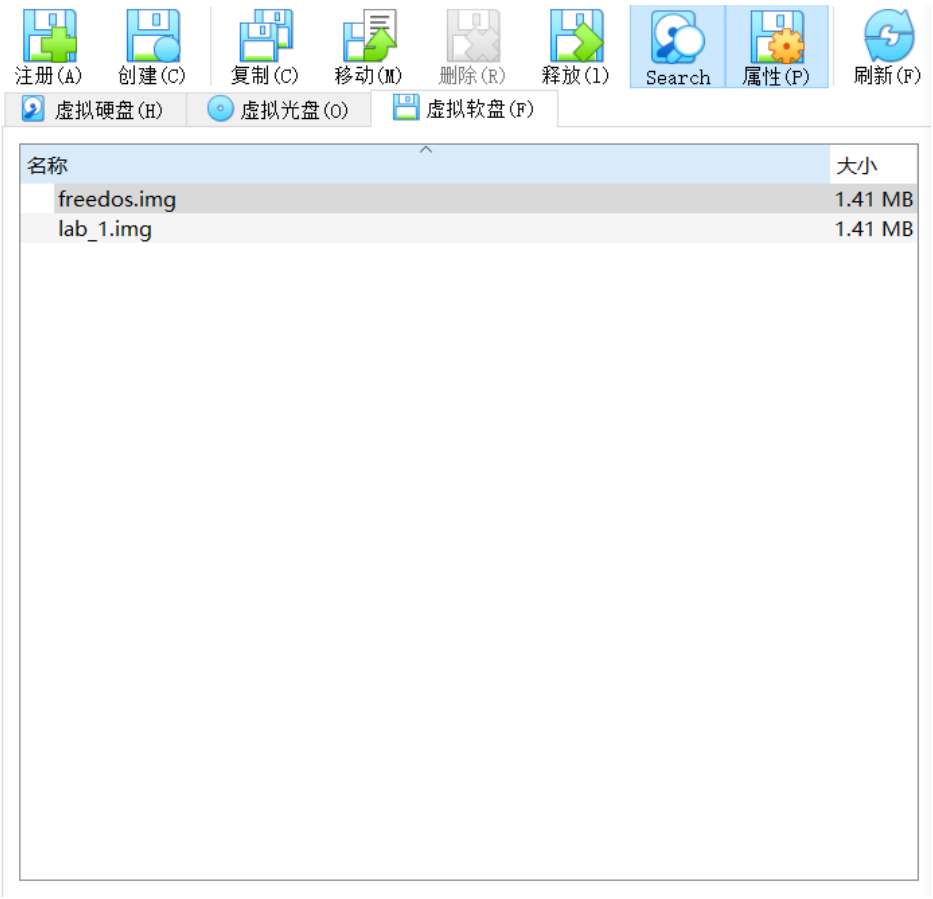
类型选择other，版本选择Other/Unknown



然后不需要添加硬盘便可以

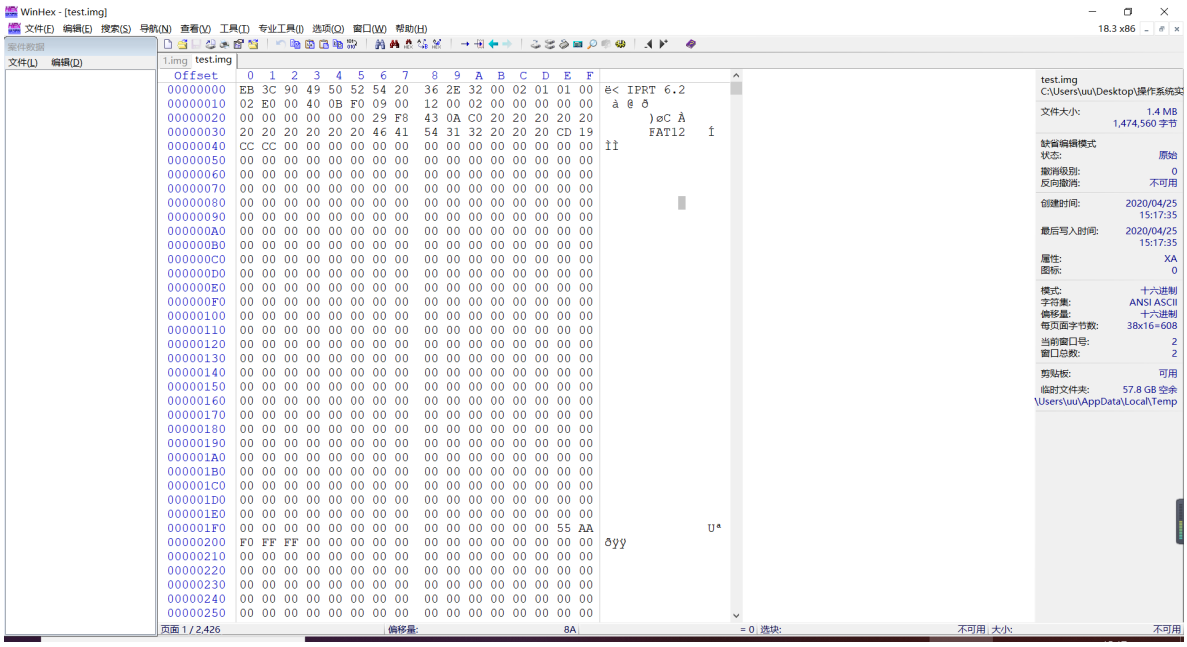
2）虚拟软盘的生成，并使用WinHex工具

首先打开VM VirtualBox，点击工具，便可以出现一个新的界面

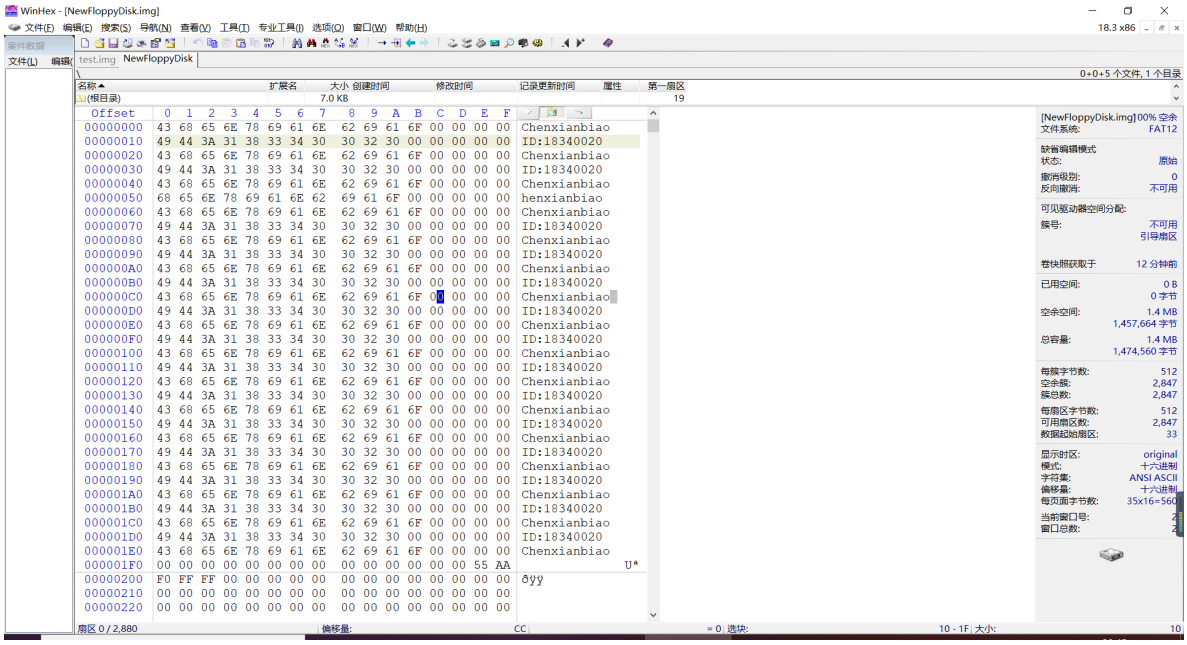


然后点击创建就可以生成自己想要大小的虚拟软盘

之后使用WinHex软件打开虚拟软盘，界面如下：



对照ascii码表填写入自己的个人信息Chenxianbiao id：18340020，之后结果为



3）引导程序的编写

```
org 7C00h
```

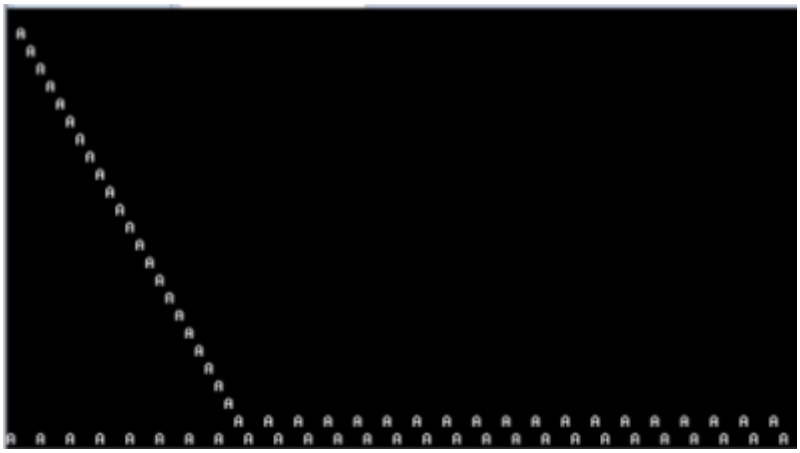
该语句使程序装载到正确的位置

在程序最后添加

```
times 510-($-$$) db 0    ; 填充0，一直到第510字节
db 55h, 0AAh             ; 扇区末尾两个字节为0x55和0xAA
```

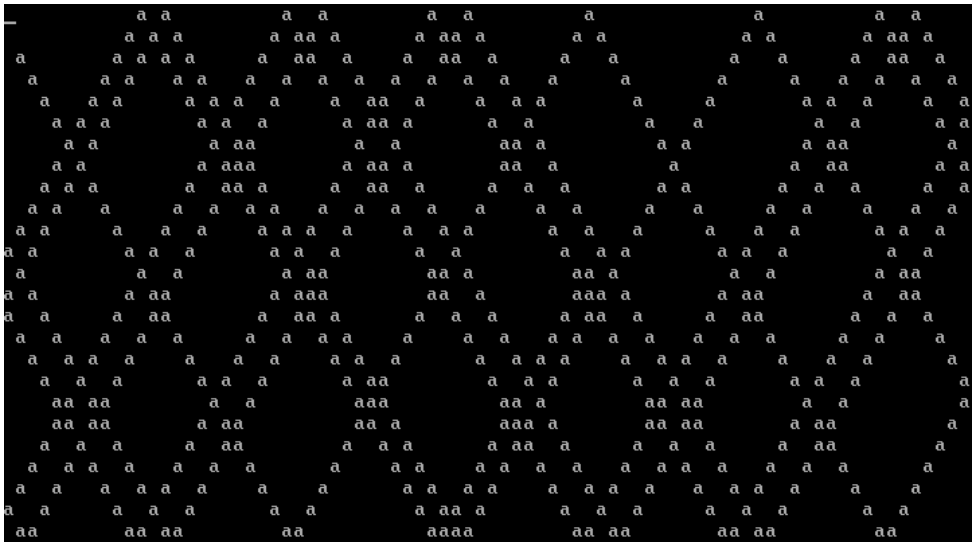
这两行代码的作用是将程序的剩余部分用0填充，直到第510个字节。之后，填充第511和第512个字节为0x55和0xAA。这样子编译出的程序机器码将正好是512字节。

一开始的时候，我之间将老师给的参考程序直接运行，改了一下后发现程序是能够跑动的，但是老师的程序是有逻辑错误的



这大概是老师的一些用意，想让我们熟悉一些汇编语言的使用

于是我修改好老师的代码，于是就完成基本的实验要求

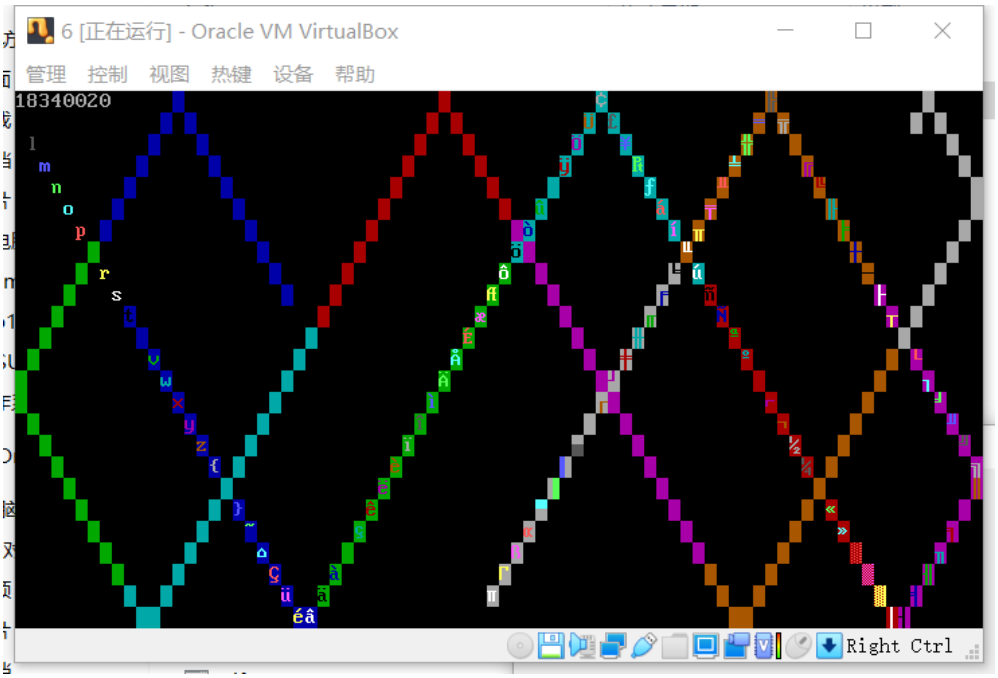


之后我加上打印自己的学号，以及改变滑动字符的颜色以及字母，代码如下

```
;添加学号
mov byte [es:0x00],'1'
mov byte [es:0x01],0x07
mov byte [es:0x02],'8'
mov byte [es:0x03],0x07
mov byte [es:0x04],'3'
mov byte [es:0x05],0x07
mov byte [es:0x06],'4'
mov byte [es:0x07],0x07
mov byte [es:0x08],'0'
mov byte [es:0x09],0x07
mov byte [es:0x0a],'0'
mov byte [es:0x0b],0x07
mov byte [es:0x0c],'2'
mov byte [es:0x0d],0x07
mov byte [es:0x0e],'0'


;每次打印+1
inc byte[color]
inc byte[char]
```

示意图如下：



因为老师说加上的东西越多越好，但是再加上自己名字后，编写的代码超出了512字节

```
D:\new\nasm>nasm -f bin 1.asm -o 1.bin
1.asm:186: error: TIMES value -24 is negative
```

4）程序写入软盘

首先把写好的.asm文件放到nasm文件夹下，然后输入以下指令

```
nasm -f bin  文件名.asm -o 文件名.bin
```

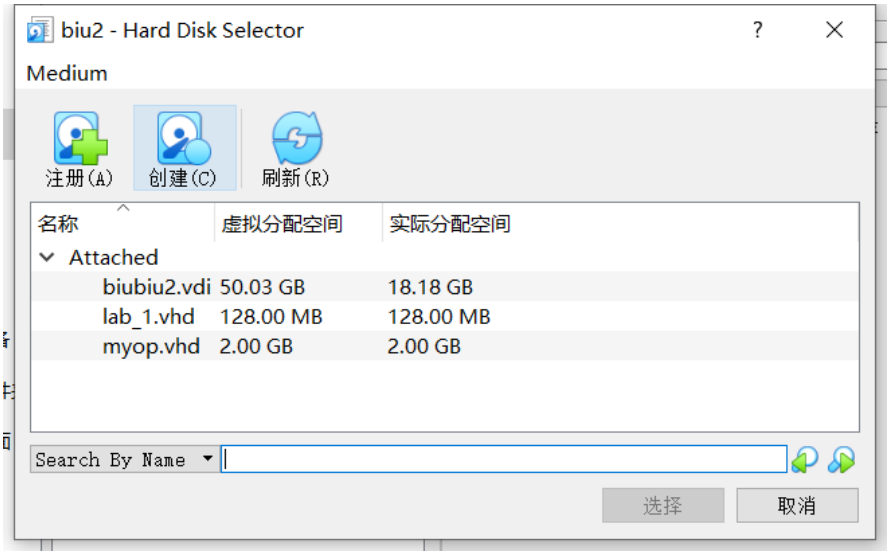
然后就可以使用FloppWriter写入工具

打开工具后界面如下：

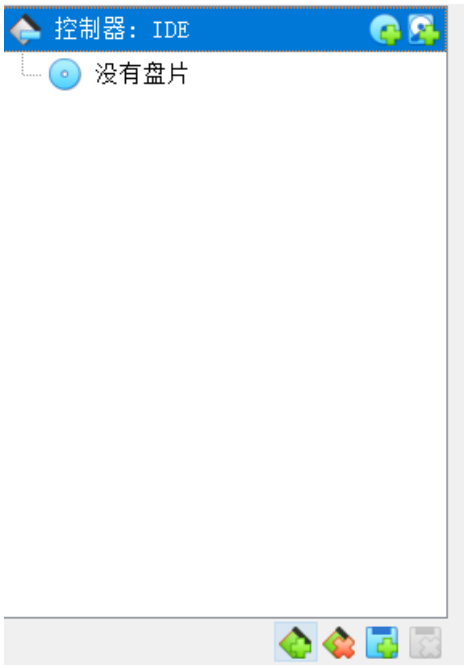
点击Write File to Image,首先选取.bin文件，然后再选取.img文件，然后就成功了

5）将该软盘设置为pc引导盘

这个一开始我遇到了一些问题，因为我发现VM VirtualBox默认虚拟出来的pc是使用硬盘引导的，点击添加盘片的时候不能选择img文件，只要vhd文件



之后我就发现下方有一个添加其他的选项



点击添加floppy后，就可以添加img文件了，操作后结果如下：

操作完之后，就可以双击虚拟pc就可以看到效果了

6.实验总结

这次实验验做的是很累的，因为大二上学期我们上的计组课程是以 Mips体系结构为中心，对 X86 指令涉及的很少，这次实验应该是把 X86 指令复习了一遍。这次实验的收获：

知道了 X86 指令和 Mips 指令的巨大不同,X86 指令更加的复杂，寄存器多且灵活，跟 Mips 整齐的模式大相径庭，还有就是 X86 的汇编语言的运算指令都有规定好了的寄存器， 只能将数字移到那些寄存器中进行计算，这与 MIPS 也是完全不同的，指令方面也是，X86 的指令多且庞杂，比 MIPS 多了好多好多其他的指令，这些都是需要自己去学的。

更加深入的学习使用了虚拟机，在大一的时候我开始使用虚拟机来模拟LINUX环境，那时候就下载了VM VirtualBox，这次装一个裸机还是比较快的，不过这次使用软盘来进行引导，因为之前都是使用硬盘的，所以这次加深了我对计算机一些硬件的理解与使用

明白了一个扇区的概念，一个扇区是512字节，超过了一次就知道不能超过512字节

懂得了如何将一个汇编程序转成机器码，并且在自己创建的虚拟机上面运行。

这个实验让我最印象深刻的就是汇编语言，相比高级语言，汇编语言真的要求更高，更苛刻，所以需要我们特别的小心，特别的仔细，想想当初那些操作机器码的前辈们就更加困难了，他们的条件那么艰苦，却能做的很好，我就更不应该抱怨了，应该更努力。

总体来说，由于这次是操作系统的第一次实验，遇到的坑还是不少的，一些工具的使用还是不太熟悉，不过经过不断地反复尝试，最后出现我需要的界面的时候还是很有成就感的。