

实验一 接管裸机的控制权 实验报告

数据科学与计算机学院 计算机科学与技术 2016 级

王凯祺 16337233

2018 年 3 月 11 日

1 实验目的

- 掌握 NASM 的语法
- 掌握用汇编器的用法
- 掌握创建空白软盘镜像的方法
- 掌握 WinHex 软件的使用方法
- 掌握虚拟机的设置及使用方法

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的直线运动，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。还要在屏幕某个区域以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虚拟软盘的首扇区，并用此软盘引导你的 PC，直到成功。

3 实验方案

本实验需要使用 NASM 汇编器、WinHex 软件和虚拟机。

3.1 NASM 汇编器的使用

在 Windows 下打开命令提示符，执行以下命令：

```
1 name -f bin stone.asm -o stone.com
```

如果编译成功，屏幕会显示 Warning 信息或不显示任何内容。

如果编译不成功，屏幕会显示 Error 信息以及 Warning 信息。

3.2 创建空白软盘镜像

关于创建空白镜像，我查了一下网上对 flp 格式的解释，只需要在二进制模式下写 1440×1024 字节的 0 即可创建空白镜像。使用 C++ 代码可创建该空白镜像。执行该代码会将新的空白镜像保存在当前目录的 new.flp 文件。

```
1 #include <bits/stdc++.h>
2
3 #define N (1440 * 1024)
4
5 using namespace std;
6
7 char a[N * 2];
8
9 int main() {
10     freopen("new.flp", "w", stdout);
11     memset(a, 0, sizeof(a));
12     fwrite(a, 1, N, stdout);
13 }
```

3.3 使用 WinHex 软件修改镜像

- 用 WinHex 打开编译好的 COM 文件、空白的 flp 镜像
- 将 COM 文件全文复制到 flp 镜像的 0x0000 字节处，须保证文件大小不变
- 将 0x01FE, 0x1FF 两个字节的值分别修改为 0x55, 0xAA

3.4 虚拟机的配置

实验环境：

- 操作系统环境 Ubuntu 16.10
- 虚拟机环境 VirtualBox 5.2.4 r1 19785 (Qt5.6.1)

虚拟机配置：

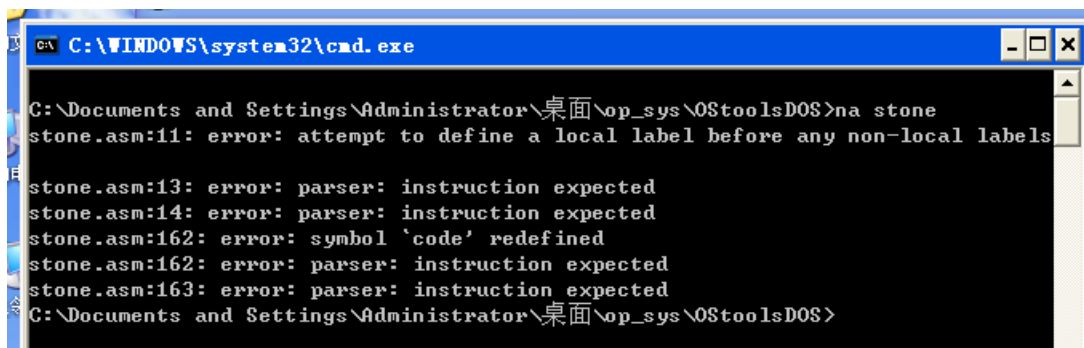
- 操作系统：其他 - DOS
- CPU：1 核心
- 内存：32 MB
- 显存：9 MB
- 软盘：使用镜像文件 a.flp

4 实验过程和结果

4.1 汇编程序的修改

4.1.1 删除程序段

我们直接编译老师提供的代码，发现编译错误如下：



```
C:\WINDOWS\system32\cmd.exe

C:\Documents and Settings\Administrator\桌面\op_sys\OSToolsDOS>na stone
stone.asm:11: error: attempt to define a local label before any non-local labels
stone.asm:13: error: parser: instruction expected
stone.asm:14: error: parser: instruction expected
stone.asm:162: error: symbol 'code' redefined
stone.asm:162: error: parser: instruction expected
stone.asm:163: error: parser: instruction expected
C:\Documents and Settings\Administrator\桌面\op_sys\OSToolsDOS>
```

```
1      .386
2      org 100h                ; 程序加载到100h，可用于生成COM
3      ASSUME cs:code,ds:code
4      code SEGMENT
```

以上程序的 1 - 4 行为原程序的 11 - 14 行。

我看到这几条编译错误是一脸懵的，并不知道出了什么问题。

于是我把编译的错误信息敲到 Google 里搜索，找到了一个以前的问题：

<https://stackoverflow.com/questions/11572307/nasm-error-parsing-instruction-expected?lq=1>

▲ That assembly language is MASM, not NASM.

3 For starters, NASM segments are defined differently.

▼ Instead of

✓ Code segment word public 'CODE'

we write

```
.section text
```

And that "ASSUME" declaration... You must have an ancient book. That is old, old MASM code. Brings back memories from the early 1980s for me!

There are many differences between NASM and MASM, and your code needs quite a bit of work to port. If you want to port that MASM code to NASM, see [MASM/NASM Differences](#) or the NASM documentation or google "NASM vs MASM"

TL;DR: you are writing MASM code, not NASM.

上面的解答非常精彩，大意是说：这是 MASM 代码，不可以用 NASM 编译。如果要使用 NASM 编译，须将代码段删除，或者换成 NASM 规定的代码段 (section .text)。答主还调侃了 MASM 是上

世纪 80 年代的东西，题主提的问题仿佛让我看到了一本古书。回答的最后还附有 MASM 和 NASM 的差别。

所以最后，我对这个问题的处理方式是：

- 删除 11 行、13 行、14 行、162 行、163 行
- 在原程序的 15 行前加上 `section .text`
- 在原程序的 155 行前加上 `section .data`

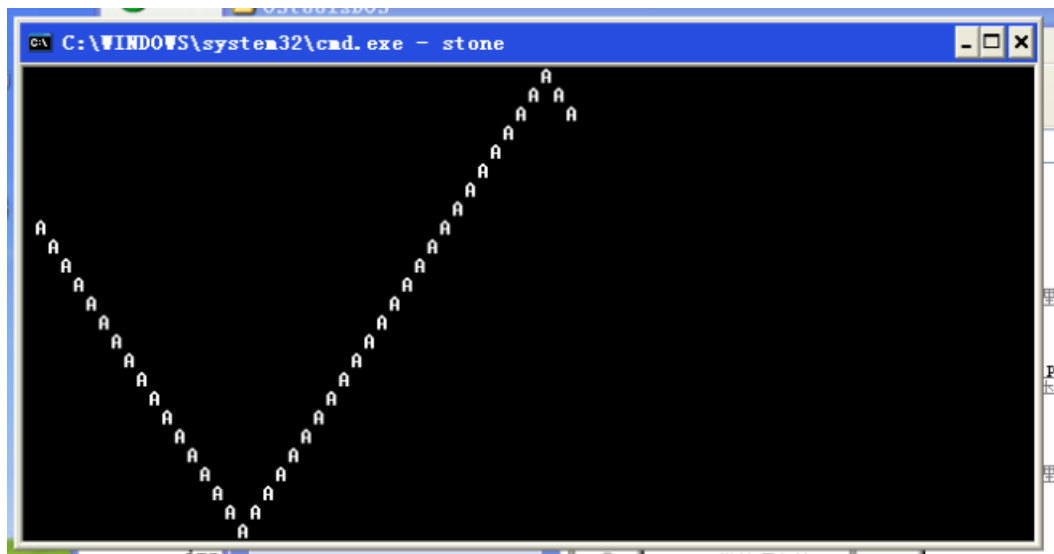
4.1.2 修改显示地址

修改了上述问题后，程序终于能编译通过了。我迫不及待点开来看，发现啥也没显示。

于是我立刻怀疑 `show` 模块有鬼。

果然不出所料，`es` 为 0，与显示的起始地址 `0xB8000` 不匹配。

由于之前已设置了初值 `gs = 0xB800`，我把 `es` 换成 `gs` 就解决了此问题。



修改以上代码，老师写的程序就成功地运行在我的 Windows XP 上了！

4.1.3 加入个性化信息

我想把单个字符 'A' 换成自己的名字、学号，就在数据段写了一个字符串。每次刷新屏幕时，用字符串指针 `si` 寄存器从头到尾遍历字符串，在屏幕上显示的就是我的学号和姓名信息了！

4.1.4 减小程序体积

程序写好后，我发现我的可执行文件大小为 519 字节，大于规定的 510 字节，导致 511 - 512 字节处无法填写 `0x55AA`。经过对代码的细致检查，我发现有几条没用的语句，将其删除后把可执行文件的大小控制在 507 字节。

4.2 写入 flp 软盘镜像

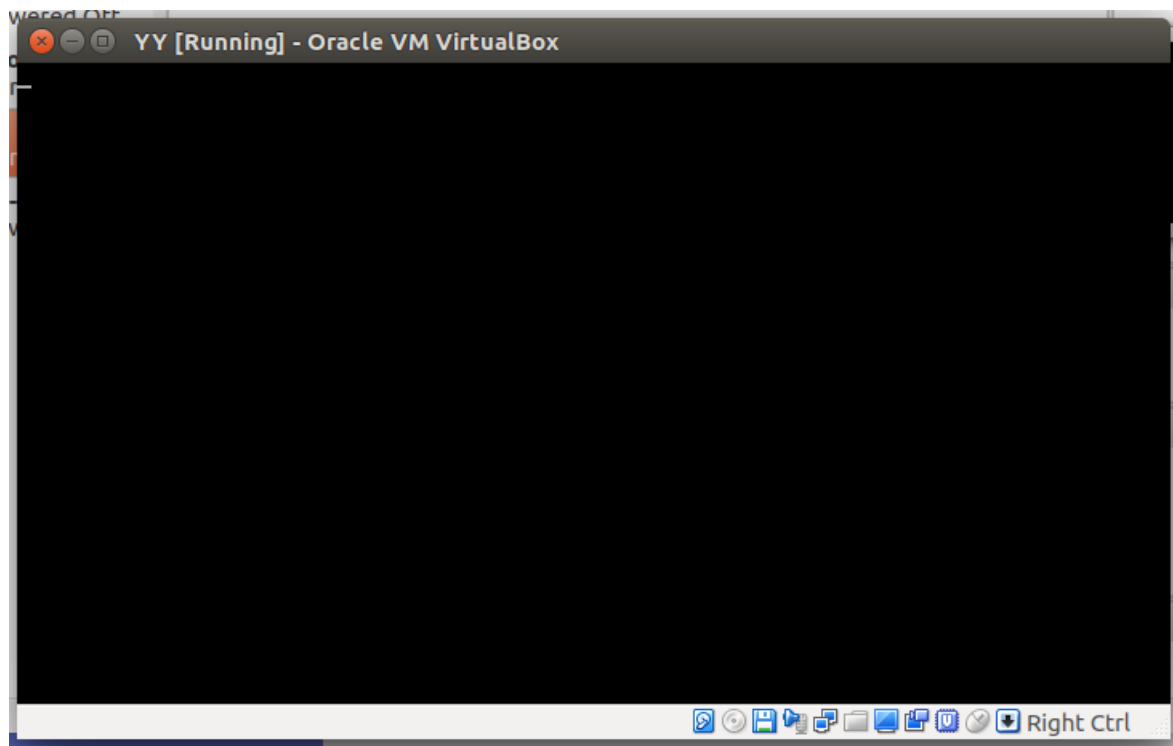
- 用 WinHex 打开编译好的 COM 文件、空白的 flp 镜像
- 将 COM 文件全文复制到 flp 镜像的 0x0000 字节处，须保证文件大小不变
- 将 0x01FE, 0x1FF 两个字节的值分别修改为 0x55, 0xAA

4.3 用虚拟机执行程序

将 flp 软盘镜像挂载到虚拟机上，即可运行程序。

4.3.1 修复初始化问题

老师写的程序能运行在 Windows XP 上，但是在虚拟机却不能正确运行，如下图，只有一个光标在显示。



我感到非常奇怪，决定去探个究竟。

我决定写一个简单的程序在虚拟机上跑：

```
1      org 100h                ; 程序加载到100h，可用于生成COM
2 start:
3      mov ax,cs
4      mov es,ax                ; ES = 0
5      mov ds,ax                ; DS = CS
6      mov es,ax                ; ES = CS
7      mov ax,0B800h            ; 文本窗口显存起始地址
8      mov gs,ax                ; GS = B800h
9      mov byte[char], 'A'
```

```

10 show:
11     xor ax,ax                ; 计算显存地址
12     mov ax,word[x]
13     mov bx,80
14     mul bx
15     add ax,word[y]
16     mov bx,2
17     mul bx
18     mov bx,ax
19     mov ah,0Fh              ; 0000: 黑底、1111: 亮白字 (默认值为07h)
20     mov al,byte[char]       ; AL = 显示字符值 (默认值为20h=空格符)
21     mov [gs:bx],ax          ; 显示字符的ASCII码值
22 end:
23     jmp $                  ; 停止画框, 无限循环
24
25 datadef:
26     x     dw 7
27     y     dw 0
28     char db 'A'

```

这一跑,就跑出问题来了!屏幕上显示了'A'字符,只是它是在左上角(0,0)位置,与我们预期的(7,0)位置不符。我怀疑是汇编程序没有帮我初始化,就在第10行前加上两行:

```

1     mov word [x], 7
2     mov word [y], 0

```

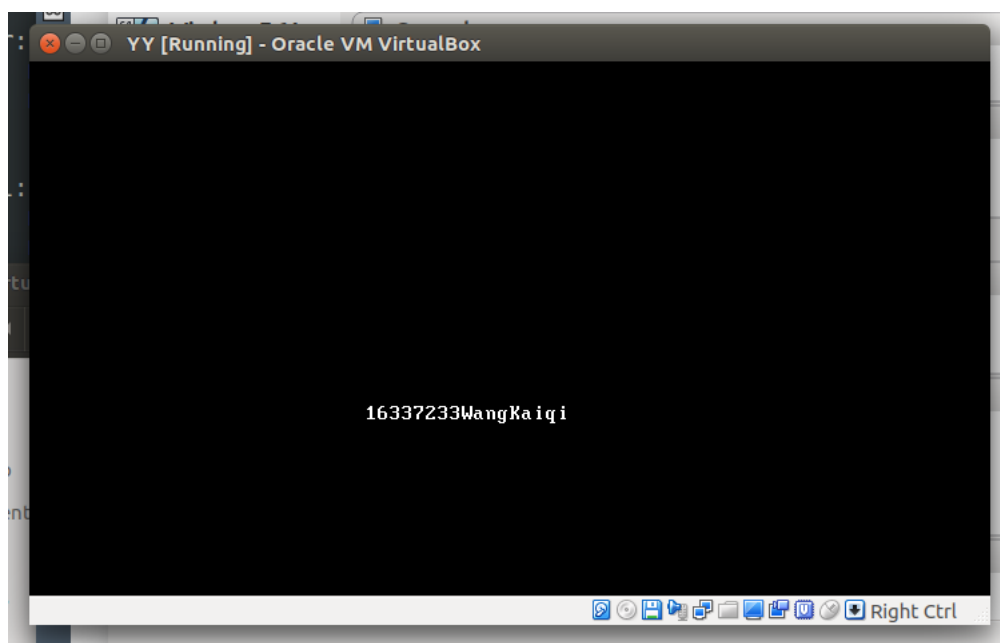
修改后,成功在虚拟机上运行!NASM程序在虚拟机运行时确实没有初始化变量。

4.3.2 后续

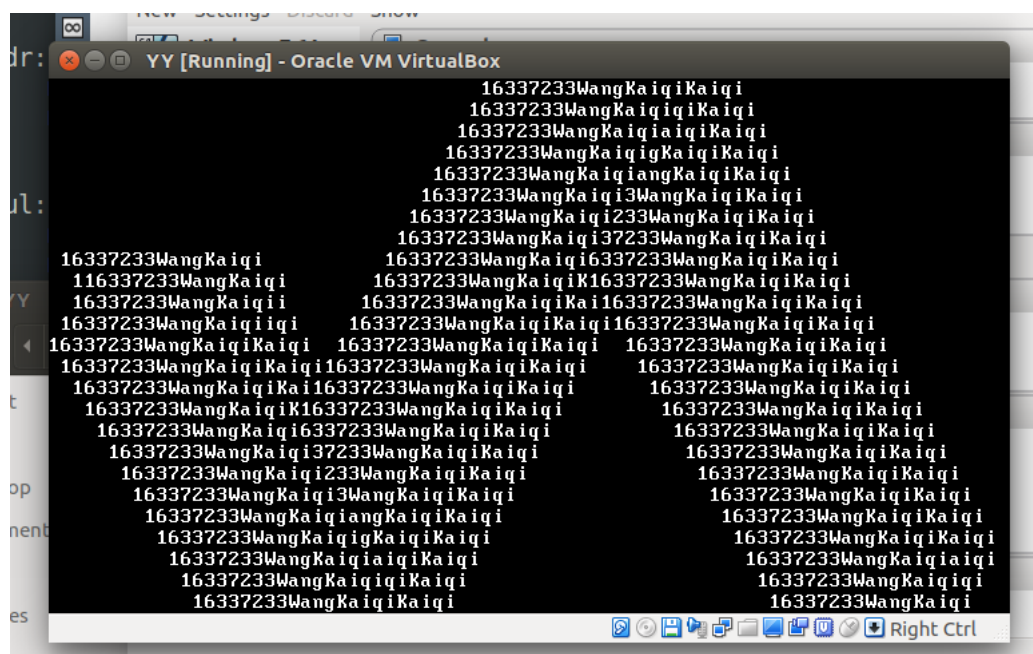
舍友告诉我,裸机上程序入口应为0x7c00,而不是0x0100,所以变量没有初始化。于是我把org 100h改为org 7c00h,并把变量初始化的过程删掉,顺利在虚拟机上运行。

4.4 最终运行效果图

4.4.1 某一帧的画面



4.4.2 运行轨迹图



5 实验总结

这次实验的意义很大，因为它锻炼了我的独立动手能力、资料查找能力，也让我了解到了操作系统是如何运行的，了解到 NASM 与 MASM 的区别。

在实验过程中，我遇到了很多的错误，包括无法编译、屏幕上无输出、在虚拟机上无法正确运行。我通过查找资料、自己动手尝试，测试究竟是哪里出问题，找到问题的根源。找到问题的根源还不够，怎么改还是一个挑战。由于汇编语言是上个学期计算机组成原理的内容，里面的指令我都记得差不多了，同样是需要上网搜索指令的用法。

在我自己遇到问题时，会跟其他同学讨论。我在自己的机器上运行成功，却在舍友的机器上不能正确运行，只能显示光标。舍友告诉我：“你的 0x55AA 与你的可执行程序重叠了。”我在编写时想着最后那几个字符应该无关紧要，重叠了就让它重叠嘛；后来修改后在舍友的机器上可以运行，更正了我的观点。因为每个字符都是指令的一部分，如果被修改了，指令也会被修改。

在其他同学遇到问题时，我也会指出对方的问题。比如我舍友的软盘镜像只能在他自己的机器上跑，却不能在我的机器上跑，显示 No bootable device 错误。我怀疑是镜像有问题，查看镜像大小，少了 3 个字节。他改好后，再在我的机器上运行，就运行成功了。

总之，这次实验的意义很大，让我不再停留在课本，而是亲身实践，写出一个非常简单的操作系统出来。