操作系统实验项目

接管裸机的控制权实验报告

郑戈涵　17338233　931252924@qq.com

摘要

本次实验总共完成两个任务：搭建和应用实验环境和接管裸机的控制权

目录

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

1. 知道原型操作系统设计实验的两条线路和前 6 个实验项目的差别
2. 掌握 PC 电脑的开机引导方法与过程的步骤
3. 在自己的电脑上安装配置引导程序设计的开发工具与环境
4. 参考样版汇编程序，完成在 PC 虚拟机上设计一个引导程序的完整工作。
5. 编写实验报告，描述实验工作的过程和必要的细节，以证实实验工作的真实性

3 实验内容

3.1 搭建和应用实验环境

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。

3.2 接管裸机的控制权

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

4 实验原理

4.1 主引导扇区

读取的主引导扇区数据有 512 字节，ROM-BIOS 程序将它加载到逻辑地址 0x0000:0x7c00 处，也就是物理地址 0x07c00 处，然后判断它是否有效。一个有效的主引导扇区，其最后两字节应当是 0x55 和 0xAA。ROM-BIOS 程序首先检测这两个标志，如果主引导扇区有效，则以一个段间转移指令 jmp 0x0000:0x7c00 跳到那里继续执行 [2]

4.2 字符显示原理

由于历史的原因，所有在个人计算机上使用的显卡，在加电自检之后都会把自己初始化到 80Œ25 的文本模式。在这种模式下，屏幕上可以显示 25 行，每行 80 个字符，每屏总共 2000 个字符的代码存放到显存里，第 1 个代码对应着屏幕左上角第 1 个字符，第 2 个代码对应着屏幕

左上角第 2 个字符，后面的依次类推。屏幕上的每个字符对应着显存中的两个连续字节，前一个是字符的 ASCII 代码，后面是字符的显示属性，包括字符颜色（前景色）和底色（背景色）。[2]

5 实验过程

5.1 编写汇编代码

代码参考了老师提供的 *stoneN.asm*, 程序有两个要求，一是字符动态显示，二是名字静态显示。

5.1.1 字符动态显示

为了让字符能够碰撞屏幕，字符的运动状态需要被记住，总共四种，左上，左下，右上，右下，每种状态遇到屏幕的四个边界都有对应的反应。为了使字符持续运动，运动过程需要在一个死循环中。代码开始先初始化段寄存器，然后进入循环，循环开始先延迟一段时间，用于控制画框的速度。然后设置初始运动状态 (右下) 和位置 (7,0), 用状态和预设定好的状态对应数字依次比较，并跳到不同状态对应的代码执行。下面以右下为例，右下状态下在遇到右边界和下边界时反弹，*decrement x,y* 坐标后需要分别与 25，80 比较是否到达边界。若到达，跳转到对应的代码执行。跳转到的这部分代码将直接修改坐标和状态。没到则跳转到显示部分 *show* 的代码执行。

```
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
```

显示部分的代码先将 *ax* 寄存器清零，并且计算 *x,y* 对应的显存中的偏移量

$$(80 * x + y) * 2$$

, 这么计算的理由在原理部分有描述。然后用 *ax* 寄存器设置属性，我将属性存在数据区，每次设置时取出并加 1，即可在每次显示的时候产生不同的效果。最后将属性放入显存 *[es:bx]* 对应位置中完成显示，一轮循环结束。

5.1.2 名字静态显示

静态显示的字符坐标是固定的，将字符属性放入显存中的对应位置就完成了显示。

5.2 在虚拟机中加载程序

写好汇编程序后将下述代码保存为 run.sh

代码 1: 生成镜像:run.sh

```
nasm main.asm -o main.bin
dd conv=sync if=main.bin of=boot.img bs=1440k count=1
```

在命令行中执行即可生成镜像 (.img) 文件，在 VMware 中创建新虚拟机并加载改镜像，查看结果。

6 程序使用说明

6.1 实验环境

6.1.1 WinHex 工具

WinHex 是一个德国软件公司 X-Ways 所开发的十六进制数据编辑处理程序

6.1.2 VMware Workstation 15

VMware Workstation 是 VMware 公司推出的一款桌面虚拟计算软件，具有 Windows、Linux 版本。此软件可以提供虚拟机功能，使计算机可以同时运行多个不同操作系统。此次实验中用于运行镜像文件。

6.1.3 nasm

Netwide Assembler 是一款基于英特尔 x86 架构的汇编与反汇编工具。在此次实验中用生成二进制文件 (.bin)

6.1.4 Ubuntu 18.04

本次实验需要制作镜像文件 (.img)，linux 系统提供了 dd 命令，可以用于将 bin 文件转换成镜像文件。

6.2 编译方法

6.2.1 系统要求

本程序要求 linux 操作系统, 需要安装 *nasm*。

6.2.2 编译过程与参数

在源代码目录下，执行下列代码即可得到镜像文件 boot.img。

代码 2: 生成镜像

```
./run.sh
```

6.3 运行与演示

在虚拟机软件 (此实验中为 VMware Workstation) 中创建虚拟机，将镜像作为软盘载入，设置随虚拟机启动后启动虚拟机即可。

运行结果如图 1 所示。

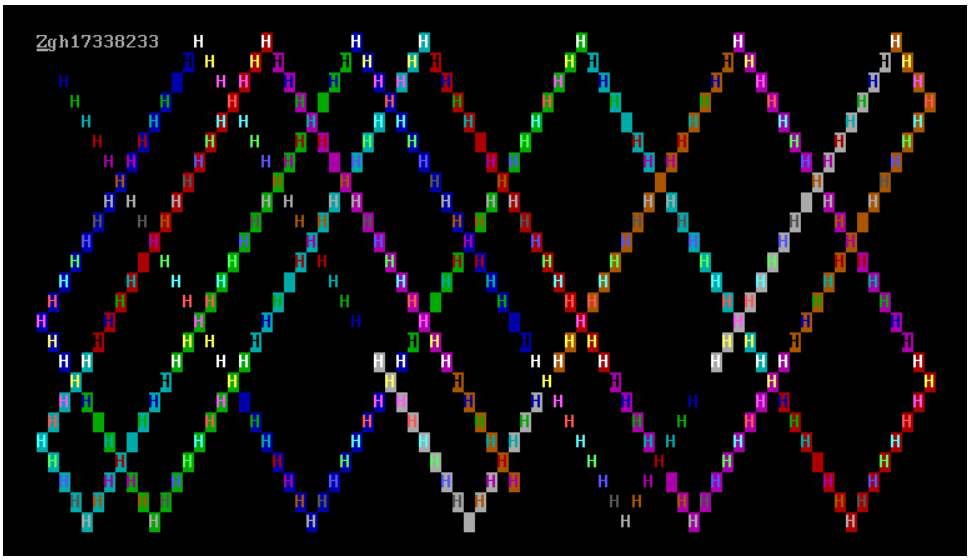

图 1: 运行过程截图

7 总结与讨论

7.1 特色, 不足与改进

本程序通过修改字体属性使得每次字符运动时都会显示出不同的样式。但是受镜像大小约束，我并没有将全名都显示出来。若要显示，不能将代码放在主引导区，需要利用其它盘块。

7.2 收获

通过本次实验，笔者复习了 x86 汇编，了解到了一些新的关键字的用法，比如 *equ,inc,dec* 等。也对操作系统的启动过程，字符的显示机制有了更深入的了解。同时，在本报告的编写过程中，笔者也练习了利用 LaTeX 编写文档的能力。[1, 3]

7.3 感想

作为操作系统的第一个实验，虽然任务的要求比较少，笔者但是第一次使用这些工具时有较强的生疏感，使用时也遇到了不少麻烦，比如 dd 命令在 *linux* 环境下自带，然而在 *windows* 环境下需要寻找替代品，这也是笔者后来改用 *linux* 系统完成编译工作的原因。并且笔者对 *x86* 汇编不如 *mips* 熟悉，而老师提供的代码也有一点小错误，导致运行时看不到结果。因此笔者花了一些时间复习了 x86 汇编。关于显存的问题，笔者查看 ppt 也没有完全理解，但是老师推荐的《x86 汇编语言：从实模式到保护模式》对这次实验的原理提供了详细的解释，对笔者有很大的帮助。所以这次实验总体上遇到的问题不算多，与同学讨论也能基本得到答案，还是比较顺利的。

References

1. Leslie Lamport, *LaTeX: a document preparation system*, Addison Wesley, Massachusetts, 2nd edition, 1994.
2. 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.
3. Contributors to Wikibooks, *LaTeX Bibliography Management*. Wikibooks, 2019., en.wikibooks.org/wiki/LaTeX/Bibliography_Management.