中山大学数据科学与计算机学院
操作系统实验课程

实 验 报 告

教　　师　凌应标

学　　号　17341035

姓　　名　傅畅

实验名称　实验一（接管裸机控制权）

实 验 一
接 管 裸 机 控 制 权

姓名：傅畅　　**学号：**17341038

邮箱：fuch8@mail2.sysu.edu.cn

实验时间：周五（3-4 节）

目录

一、 实验目的

1) 理解操作系统的定义、功能与作用，了解嵌入式开发基本原理。

2) 掌握实验环境的构成与实验工具搭配，实现实验环境的搭建与简单应用。

3) 掌握裸机编程基本操作流程，着重理解内存与地址空间，显存原理，寄存器使用规则，实现接管裸机的控制权。

二、 实验要求

（一）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。

（二）接管裸机控制权

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；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。

三、 实验方案

（一）配置方法及相关工具说明

3.1.1 实验支撑环境

- 硬件：个人计算机
- 主机操作系统：Linux 4.18.0-16-generic 17-Ubuntu
- 虚拟机软件: Bochs 2.6.9

本次实验的程序开发工作在 Linux 环境下进行，便于调试与优化；存储与 CPU 等硬件资源来自 PC，由虚拟机软件的调度算法控制。

虚拟机为操作系统的运行提供一个虚拟化平台，它与主机的空间分离，理论上不会对主机的运行造成影响。即使实验操作系统崩溃，也可以通过重启虚拟机的方式继续实验。

Bochs 的使用依赖配置文件，通过配置文件指定不同的硬件，以及指定存储介质的映像文件 (BIOS 的 ROM 文件、磁盘文件等)。

3.1.2 实验开发工具

- 汇编语言工具：x86 汇编
- 程序编辑器：VScode
- x86 汇编器：NASM_v2.07
- 磁盘映像文件浏览编辑工具：wxHexEditor 0.23 Beta For Linux

本次实验主要在 VScode 编辑平台上使用汇编语言模式完成。汇编程序使用 NASM 的汇编命令生成 bin 文件，然后使用 wxHexEditor 检查格式完整，完成引导盘的设计。

（二）方案思想与程序流程

- 多小球独立并发移动
- 程序设计

程序的数据区主要包括小球个数 n，每个小球的方向 dir 以及四个方向移动，所对应的坐标改变连量 delx[4], dely[4]。同时另外开四个 db 变量：$counti, countj, countk, countl$，用于下面程序实现循环时的计数

程序在 Initial 段完成段寄存器的赋值后，开始进入 $show \rightarrow slide \rightarrow show$ 的循环，show 的部分，是将 n 个小球的位置逐个打印到其（posx, posy）所对应的位置上；然后利用冗余循环等待若干 ms 之后，将原来被打印的位置清零，不保留上次移动的痕迹。slide 的部分就是根据小球当前位置调整其下一步移动的方向，然后移动一步，并更新其位置。

（三）数据结构与算法

- 数据结构：数组

使用若干个长度为 n 的数组，其中小球的坐标（x,y），运动方向（$\delta x, \delta y$）使用了 4 个 db 数组，转化后小球在屏幕上位置 $(x * 80 + y) \times 2)$ 使用了一个 dw 数组。这样提升了代码的复用性与编程效率。

```
delx db 0x01,0x01,0xff,0xff
dely db 0x01,0xff,0x01,0xff  ; 00 01 10 11
posx db 0x00
posy db 0x00
poss dw 0x00
dir db 0x00

counti dw 0
countj dw 0
countk dw 0
countl dw 0
```

代码 1: stone_v0.asm 数组定义

- 算法：循环模拟

小球运动比较简单，直接模拟即可

（四）程序关键模块说明

- sleep

sleep 段主要完成延时与循环变量的清零。

```
    mov word[counti],delay
    mov word[countj],delay
sleeploop:
    dec word[counti]
    jnz sleeploop
    mov word[counti], delay
    dec word[countj]
    jnz sleeploop
```

代码 2: stone.asm sleeploop 段

- loop1

在前面已说明，此处仅贴代码。

```
l   mov word [counti], posx
    mov word [countj], posy
    mov word [countk], n
    mov word [countl], poss
    loop1:
        mov ax, 0x0
        mov bx, 0x0
        mov dx, 0x0
        mov bx, [counti]  ; 取出循环变量i所指posx位置的值

```

```
11          mov al, [bx]
12
13          mov cx, 0x50   ;将x乘上行长度
14          mul cx
15
16          mov bx, [countj]   ;取出循环变量j所指posy位置的值
17          mov dl, [bx]
18          add ax,dx
19          mov cx, 0x2
20          mul cx
21
22          mov bx, ax
23          mov byte [es:bx],'*'
24          mov byte [es:bx+0x01],0x07
25          mov bx, [countl]   ;屏幕对应的实际位置，需要*2
26          mov [bx], ax
27
28          inc word [counti]
29          inc word [countj]
30          inc word [countl] ;循环变量l所指的poss是dw类型的变量，位长2Byte，所以每
                次循环要+2
31          inc word [countl]
32          dec word [countk]
33      jnz loop1
```

代码 3: stone_v0.asm loop1 段

- Slide

首先根据当前位置改变运动方向，然后移动位置

```
1   slide:
2       mov word [counti], posx
3       mov word [countj], posy
4       mov word [countl], dir
5       mov word [countk], n
6       loop3:                  ;循化n次，对每个位置独立考虑
7           mov bx,[counti]
8           mov dl, [bx]
9           mov bx,[countj]   ;取出x，y坐标，分别放在dl，dh中
10          mov dh, [bx]
11
12          mov bx, [countl]
13          mov al, [bx]
14          mov bx,0
15          mov bl, al        ;取出当前方向，放入bl
16          mov al, [delx+bx]   ;将$\Delta x \Delta y$分别放入al, ah
17          mov ah, [dely+bx]
```

```
18
19          add dl,al       ;计算沿当前方向运动到的目标位置
20          add dh,ah
21
22          mov cl, bl
23          cmp dl, 0xff
24          jne Endjudge1
25              xor cl,0x02   ;如果dl<0，改变$\Delta x$的方向
26              mov bx,[countl]
27              mov [bx],cl
28              jmp near loop3
29          Endjudge1:
30
31          cmp dl, LimitX   ;如果dl>LimitX，改变$\Delta x$的方向
32          jne Endjudge2
33              xor cl,0x2
34              mov bx,[countl]
35              mov [bx],cl
36              jmp near loop3
37          Endjudge2:
38
39          cmp dh, 0xff                ;如果dh<0，改变$\Delta y$的方向
40          jne Endjudge3
41              xor cl,0x01
42              mov bx,[countl]
43              mov [bx],cl
44              jmp near loop3
45          Endjudge3:
46
47          cmp dh,LimitY          ;如果dh>=LimitY，改变$\Delta y$的方向
48          jne Endjudge4
49              xor cl, 0x01
50              mov bx,[countl]
51              mov [bx],cl
52              jmp near loop3
53          Endjudge4:
54
55          mov bx, [countl]              ;将新的坐标和方向从寄存器中放回主存
56          mov [bx],cl
57          mov bx, [counti]
58          mov [bx], dl
59          mov bx, [countj]
60          mov [bx], dh
61
62          inc word[counti]
63          inc word [countj]
64          inc word [countl]
```

```
65          dec word [countk]
66      jnz loop3
67
68  jmp near show
```

代码 4: stone.asm slide 段

四、 实验过程与结果

（一）工具安装及使用

这一部分主要解决要求 1。

4.1.1 工具安装

- bochs 虚拟机安装时直接按照网络教程执行，没有什么大的烦恼
- NASM

Linux 下安装直接 sudo apt-get install nasm 即可

- wxHexEditor 0.32 Beta for Linux

在 ubuntu 下直接 sudo apt-get install wxhexeditor

4.1.2 工具使用

1) 配置 bochs 的指定映像文件

```
1  megs:128
2
3  romimage: file=/usr/local/share/bochs/gdb/share/bochs/BIOS-bochs-latest
4  vgaromimage: file=/usr/local/share/bochs/gdb/share/bochs/VGABIOS-lgpl-latest
5
6  floppya:1\_44=vfdd144, status=inserted
7  boot:floppy
8
9  log:bochsout.txt
10
11 mouse:enabled=0
12
13 display_library:x, options="gui_debug"
14
15 #gdbstub: enabled=1, port=1234, text_base=0, data_base=0, bss_base=0
```

代码 5: bochsrc

（二）实验操作说明与程序运行结果

这一部分主要用于解决要求 2。首先使用 NASM 的汇编命令：nasm -f bin stone_v0.asm -o vfdd144

from 2019-03-23 22-23-16.png from 2019-03-23 22-23-16.png from 2019-03-23 22-23-16.bb

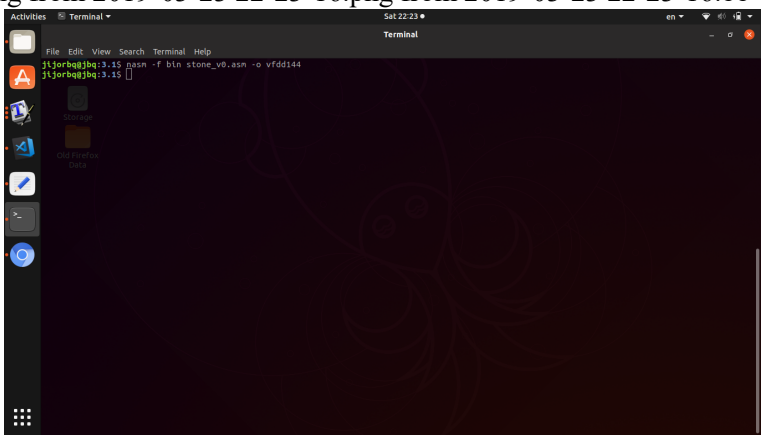
from 2019-03-23 22-23-16.bb 

图 1: 终端直接执行

终端直接执行:bochs -q -f bochsrc

from 2019-03-23 22-23-35.png from 2019-03-23 22-23-35.png from 2019-03-23 22-23-35.bb

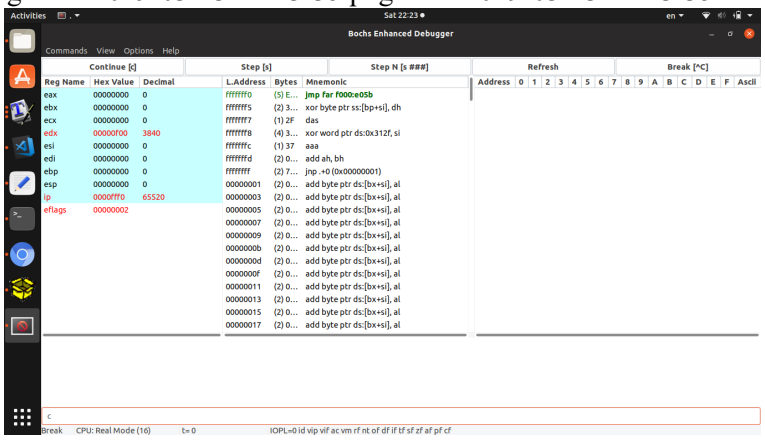
from 2019-03-23 22-23-35.bb 

图 2: 打开 bochs

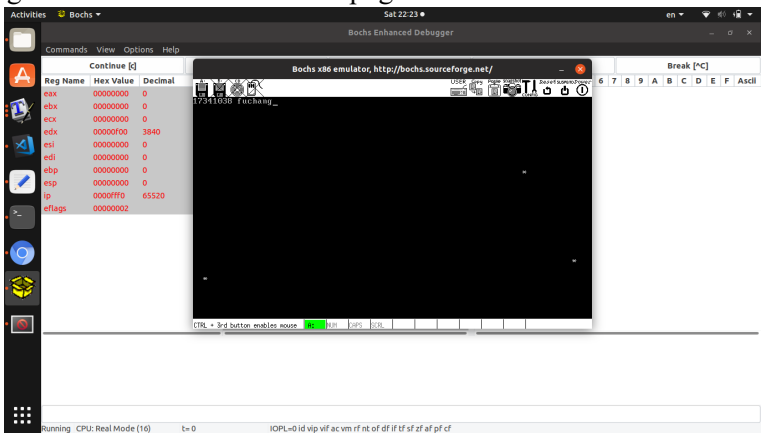
输入执行命令”c”

（三）疑难问题解决

• nasm 中乘除法的使用，很奇怪，它的默认被乘数/被除数是 ax，部分商放在 dl 中

• 有的时候对存储单位长度不太敏感，进行寄存器寻址/内存寻址的时候市场忽略 db, dw 等长度的区别，导致计算偏移的时候引入很奇怪的 bug

from 2019-03-23 22-23-44.png from 2019-03-23 22-23-44.png from 2019-03-23 22-23-44.bb



from 2019-03-23 22-23-44.bb

图 3: 显示

五、 实验总结

第一次写操作系统的实验，就向之前每学习一门语言总会以一个“Hello world”程序作为第一个程序；当我在课堂上第一次写出一个 mov byte,[es:0x00], “@” 的时候，心情和以前接触一门新的语言一样，打开了一扇新的大门。起步的程序并不难在算法，反而是学习编程的基本问题：怎样设计程序的逻辑，如何调试，特殊的语句用法如何尽快掌握。而且操作系统的编程环境相比高级语言所配置的各种炫酷赏心悦目的界面，OS 实验课要注定经历从无到有白手起家的感觉。这是 OS 相比学习其他项目更需要克服的一道心里障碍。总之第一次实验给我的信心非常好。