

**本科生实验报告**

实验课程: 操作系统原理实验

任课教师: 陈鹏飞

实验题目:lab2

专业名称: 计算机科学与技术

学生姓名:张玉瑶

学生学号: 23336316

实验地点: 实验中心B203

实验时间:2025.3.19

**Section 1 实验概述**

本次实验围绕实模式下的汇编编程展开，通过四个任务循序渐进地掌握关键系统编程技术：

1. **显存直接操作**  
   在MBR引导程序中实现字符串输出，学习显存地址计算和字符属性控制，完成从固定位置到动态定位的进阶。
2. **BIOS中断应用**  
   通过int 0x10实现光标控制和键盘输入回显，深入理解实模式中断机制。
3. **结构化编程**  
   将伪代码转换为汇编，实现分支判断、循环控制和函数调用（，强化流程控制能力。
4. **动态图形效果**  
   综合运用方向控制、显存计算和中断延时，开发字符回旋程序，实现彩色字符沿屏幕边缘顺时针旋转的动画效果。

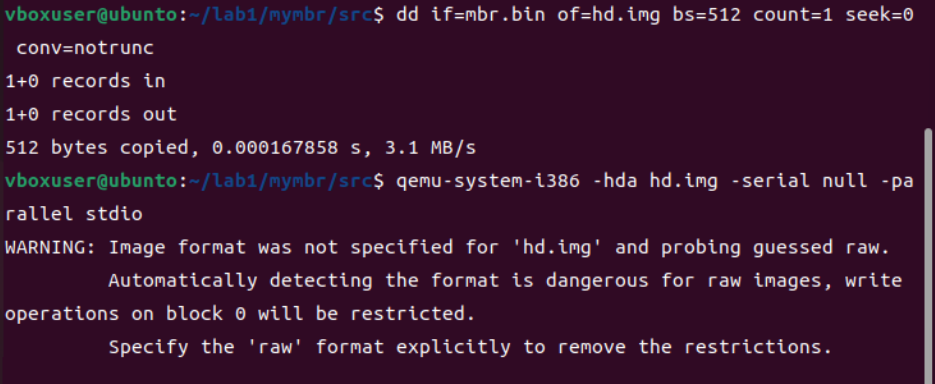
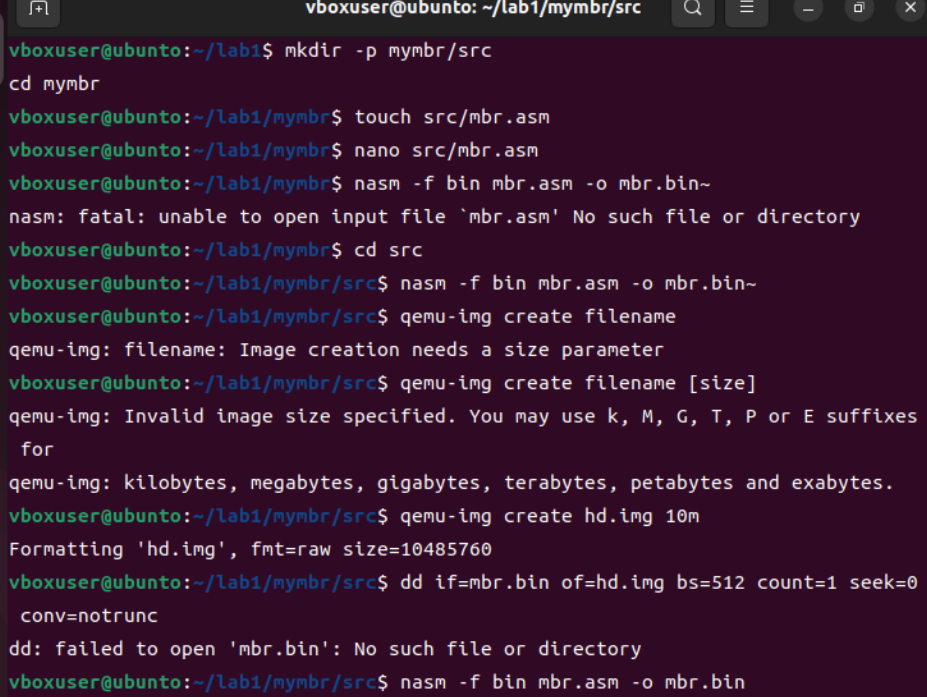
**Section 2 实验步骤与实验结果**

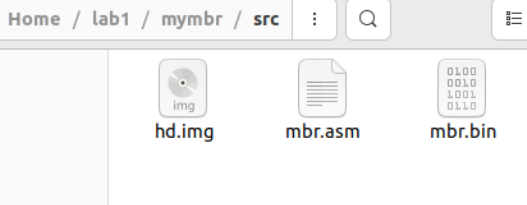
------------------------- **实验任务1** -------------------------

**1.1**

* 任务要求：复现example1，使得MBR被加载到0x7C00后在（0，0）处开始输出“hello world”。
* 思路分析：学习样例给出的代码，了解怎么利用汇编语言在显存上的某个位置进行输出。
* 实验步骤：

1. 在mymbr/src创建文件 mbr.asm
2. 把代码复制到文件中
3. Nasm编译器来将代码编译成二进制文件
4. 生成了MBR后，我们将其写入到硬盘的首扇区。我们首先创建一个“硬盘”，这个“硬盘”并不是一个真实的硬盘，实际上是一个预先指定大小的文件而已，又被称为“虚拟磁盘”。
5. 然后将MBR写入hd.img的首扇区
6. 写入MBR后我们就可以启动qemu来模拟计算机启动了
7. 启动后的效果如下。可以看到第一行已经输出“Hello World”。









**1.2**

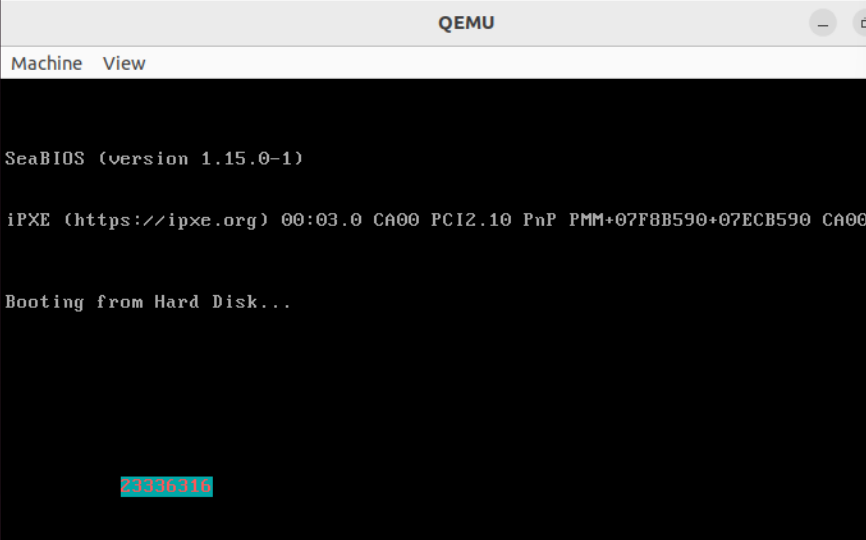
* 任务要求：请修改example 1的代码，使得MBR被加载到0x7C00后在(16,10)处开始输出我的学号23336316。
* 思路分析：修改颜色的代码，增加调整位置的代码。
* 实验步骤：

1. 在上一个代码中，加入下图两行代码：

mov ah , 0x3c ;背景色调成蓝色，前景色调成亮红色

mov di , (16\*80+10);第十六行第十个位置开始输出

1. Nasm编译器来将代码编译成二进制文件
2. 生成了MBR后，我们将其写入到硬盘的首扇区。我们首先创建一个“硬盘”，这个“硬盘”并不是一个真实的硬盘，实际上是一个预先指定大小的文件而已，又被称为“虚拟磁盘”。
3. 然后将MBR写入hd.img的首扇区
4. 写入MBR后我们就可以启动qemu来模拟计算机启动了
5. 启动后的效果如下。可以看到第16行第10列已经输出“23336316”。

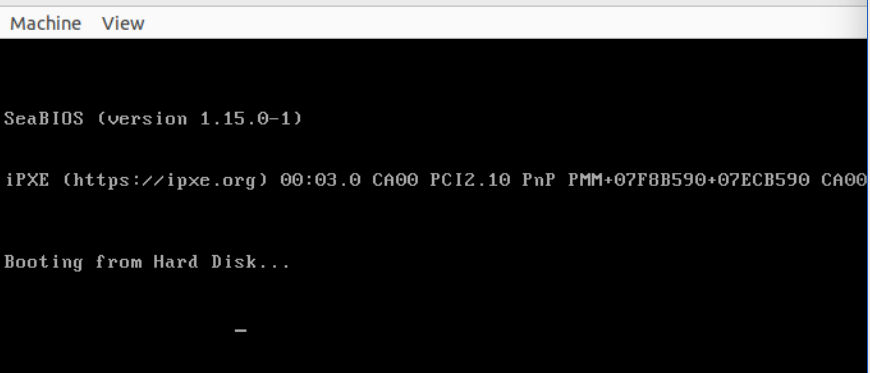


------------------------- **实验任务2**-------------------------

**2.1**

* 任务要求：请探索实模式下的光标中断，利用中断实现光标的位置获取和光标的移动。
* 思路分析：通过BIOS中断int 0x10获取和设置光标位置，实现光标的读取和移动操作
* 实验步骤：

1. 使用 int 0x10 的功能号 0x03 获取最开始光标位置。
2. 使用功能号 0x02 设置新的光标位置，实现光标移动。
3. 再使用 int 0x10 的功能号 0x03 获取修改后光标位置。

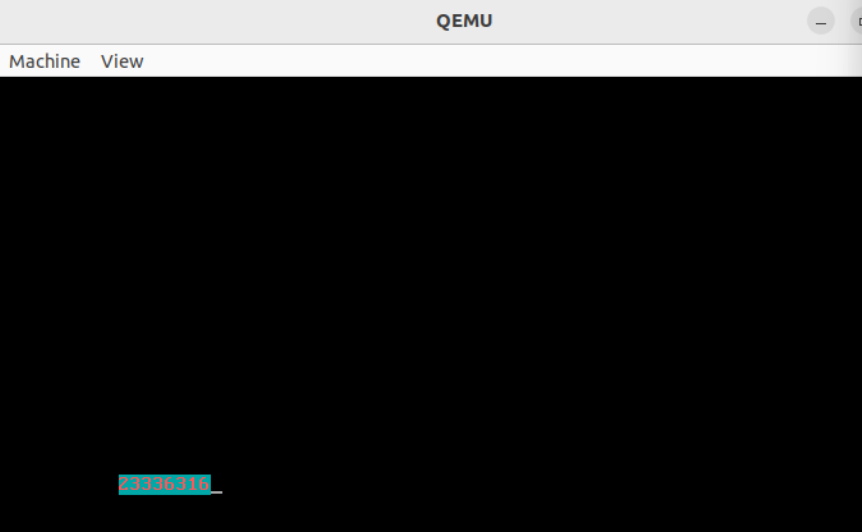


**2.2**

* 任务要求：请修改1.2的代码，使用实模式下的中断来输出你的学号。说说你是怎么做的，并将结果截图。
* 思路分析：使用BIOS中断 int 0x10 的功能号 0x02 设置光标位置，并通过功能号 0x09 显示学号字符。无循环地依次输出学号每个字符。
* 实验步骤：

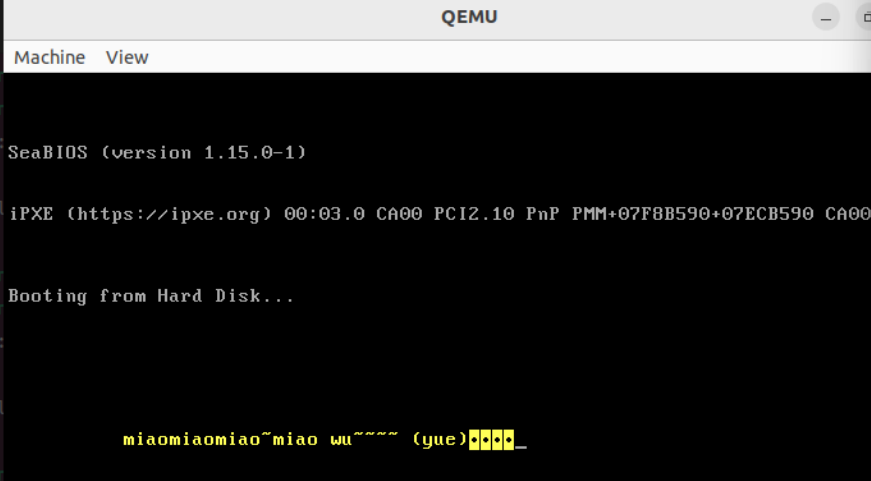
1. 使用 int 0x10 的功能号 0x02 将光标定位到指定位置。

2. 使用功能号 0x09 逐个显示学号字符，设置颜色。



* 任务要求： 在2.1和2.2的知识的基础上，探索实模式的键盘中断，利用键盘中断实现键盘输入并回显。
* 思路分析： 通过 BIOS 键盘中断 int 0x16 获取键盘输入，利用功能号 0x00 获取按键的 ASCII 码与扫描码，并将输入的字符回显到屏幕上。
* 实验步骤：

1. 使用 int 0x10 的功能号 0x02 设定光标的初始位置，设定光标颜色。
2. 使用 int 0x16 的功能号 0x00 等待键盘输入，读取按键的 ASCII 码。
3. 通过 int 0x10 的功能号 0x09 实现字符回显。



------------------------ **实验任务3**-------------------------

* 任务要求： 将伪代码转换为汇编代码，并放置在标号之后，可以正常运行test文件。
* 思路分析：

**1.分支逻辑**：用cmp和条件跳转实现三级判断。

**2.循环逻辑**：通过my\_random获取随机数，按偏移量存入数组。

**3.函数逻辑**：遍历字符串，字符+9后调用打印函数，需平衡栈空间。

* 实验代码：

; If you meet compile error, try 'sudo apt install gcc-multilib g++-multilib' first

%include "head.include"

; you code here

your\_if:

mov eax,[a1]

cmp eax,40

jge if\_case

cmp eax,18

jge else\_if

shl eax,5

mov [if\_flag],eax

jmp end\_program

if\_case:

add eax,3

mov ecx,5

idiv ecx

mov [if\_flag],eax

jmp end\_program

else\_if:

mov ebx,eax

shl ebx,1

mov eax,80

sub eax,ebx

mov [if\_flag],eax

jmp end\_program

end\_program:

mov eax,0

xor ebx,ebx

; put your implementation here

your\_while:

mov eax,[a2]

while\_loop:

cmp ebx,25

jge end\_loop

call my\_random

shl ebx,1

mov ecx,[while\_flag]

add ebx,ecx

mov [ebx],eax

mov ecx,[a2]

add ecx,1

mov [a2],ecx

mov ebx,[a2]

jmp while\_loop

end\_loop:

%include"end.include"

your\_function:

mov ecx,0

mov esi,[your\_string]

for\_loop:

mov al,[esi+ecx]

test al,al

je end\_function

add al,9

push ecx

push eax

call print\_a\_char

add esp,4

pop ecx

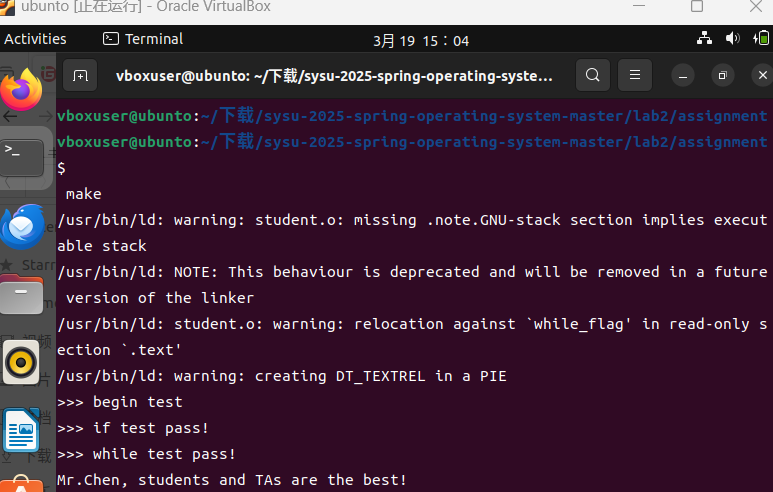
inc ecx

jmp for\_loop

end\_function:

ret

; put your implementation here



------------------------ **实验任务4**-------------------------

* 任务要求：做一个字符回旋程序。用汇编代码实现一个字符回旋程序，使其能够在qemu显示屏上面，以不同的颜色、字符内容进行顺时针绕圈。
* 思路分析： 通过方向变量d控制字符移动路径（右0下1左2上3循环），动态切换字符和颜色，利用显存地址计算实现屏幕回旋显示。
* 实验步骤：

1. **初始化**：设置80x25文本模式，初始化位置(x,y)、方向(d)、颜色(c)和字符(s)索引。

mov ax,0x0002

int 0x10

mov ax,0xB800

mov es,ax

mov byte [x],0

mov byte [y],0

mov byte [d],0 ;direction

mov word [c],0 ;color

mov word [s],0 ;char

1. **方向控制：**根据d值执行移动逻辑，到达边界时转向

1.loop:

cmp byte [d],0

je go\_right

cmp byte [d],1

je go\_down

cmp byte [d],2

je go\_left

cmp byte [d],3

je go\_up

2.go\_right:

inc byte [x]

cmp byte [x],79

jb show

mov byte [d],1

dec byte [x]

jmp show

1. **显示处理：**计算显存地址，循环使用预设字符和颜色

show:

mov ax,0

mov al,[y]

mov bx,80

mul bx

add al,[x]

adc ah,0

shl ax,1

mov di,ax

mov si, chars

add si,[s]

mov al,[si]

inc word [s]

cmp word [s],10

jb get\_color

mov word [s],0

get\_color:

mov si,cols

add si,[c]

mov ah,[si]

inc word [c]

cmp word [c],11

jb print

mov word [c],0

1. **延时循环**：通过BIOS中断实现动画效果，持续绕圈显示

; delay

mov cx, 0x0001

mov dx, 0x86A0

mov ah, 0x86

int 0x15

jmp loop

效果展示：



**Section 5 实验总结与心得体会**

太难了，不要那么难啊，我们学的如此简陋，理论课也没跟上，为啥一上来就满汉全席了。班上虽然有很多大佬，但是不意味着我不是小垃圾呀，呜呜呜，写代码写的要崩溃了呜呜呜，无能狂怒，拼尽全力无法战胜！