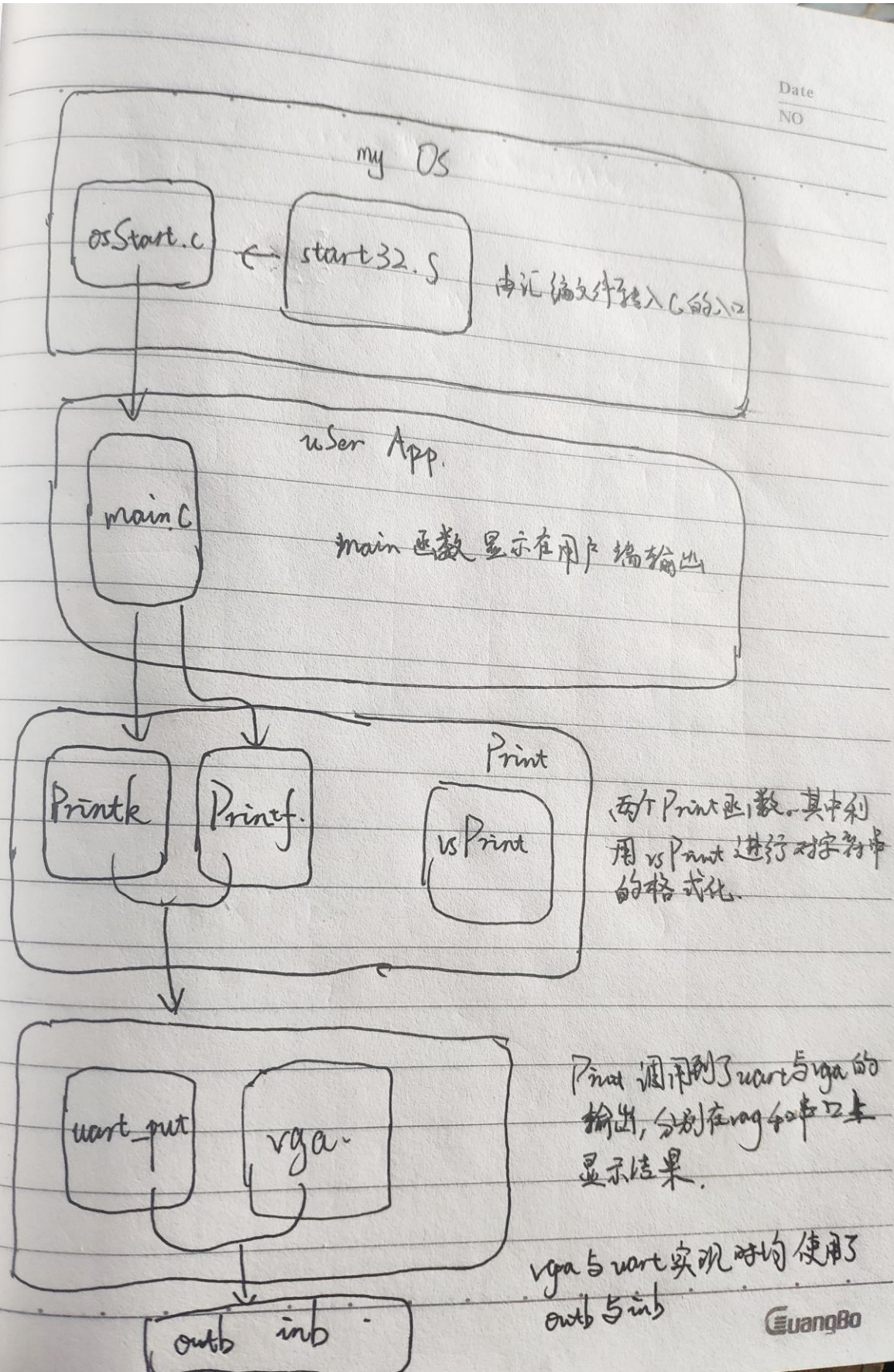
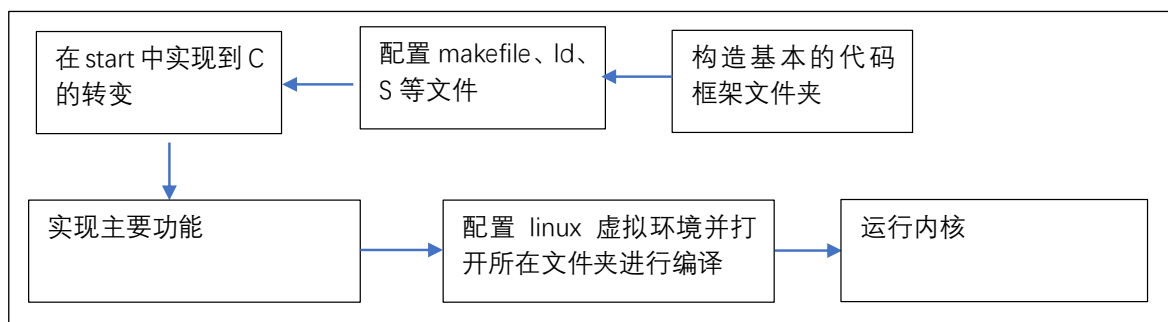


# 操作系统实验 2 报告

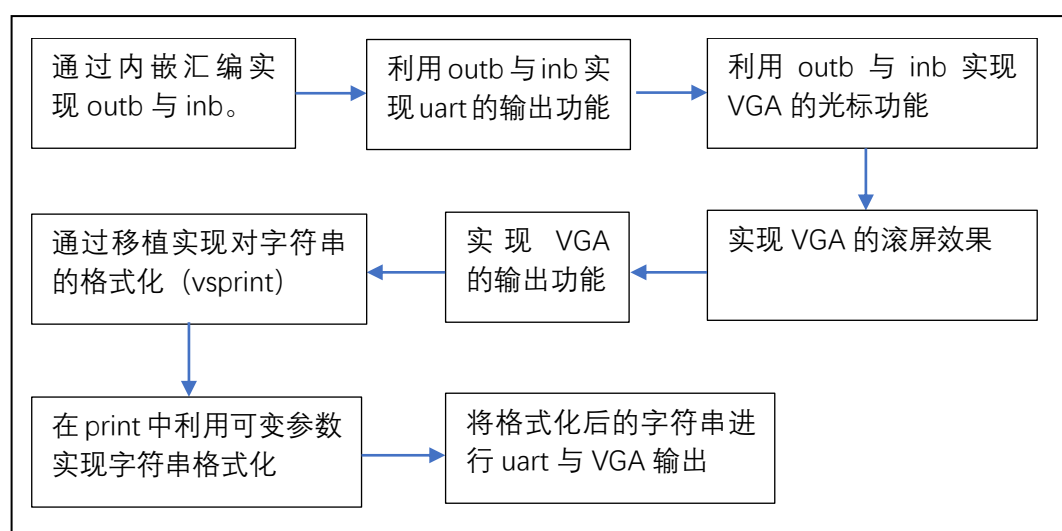
## 软件框图



## 主流程及其实现



## 主要功能模块及其实现



## 源代码说明

在本次实验的根目录下，除了配置了 makefile 与 DS 文件，还有 4 个文件夹：multibooheader、myOS、output、userApp。multibooheader 已经不陌生，其中放置了有关 multibooheader 协议的 S 文件。在 output 中，原本为空文件夹，用来输出我们编译后生成的文件。在该文件夹中的目录结构与根目录相似，主要是为了对每个文件输出时不产生混淆。在 userApp 中存放的是 main.c 文件与该文件夹下的 makefile 文件，Main 文件是用来测试本次实验的功能的入口。在 myOS 文件夹中存放了本次实验的主要源文件。直接存放在此该目录下的文件有

DS 文件、该文件夹下的 makefile 文件、链接器文件、一个配置地址的 S 文件和一个包含了 main 函数的与 multibootheader 中的 S 文件相关联的 C 文件。这个文件也是从汇编到 C 的转变。在该目录下则有 3 个实现相关功能的文件夹: dev、i386、printk。每个文件夹下都有一个相应的 makefile 文件以及实现功能的 C 语言源文件。I386 文件夹下是实现 IO (inb、outb) 的源文件, dev 下存放的是在 uart 与 vga 上实现给定字符串并输出的功能的源文件, 而在 printk 中则是实现 printf 与 printk 的源文件。每个子目录下的 makefile 文件的输出目录都是在 output 中的同名目录。

## 代码布局说明





所有的引导模块将按页 (4KB) 边界对齐, 物理内存地址从 1M 处开始

## 编译过程说明:

在 Ubuntu 中先搜索到 lab2 的目录, 然后通过指令 make 完成编译, 可以看到

在 output 目录中的对应目录中分别输出了与根目录下对应文件相同文件。

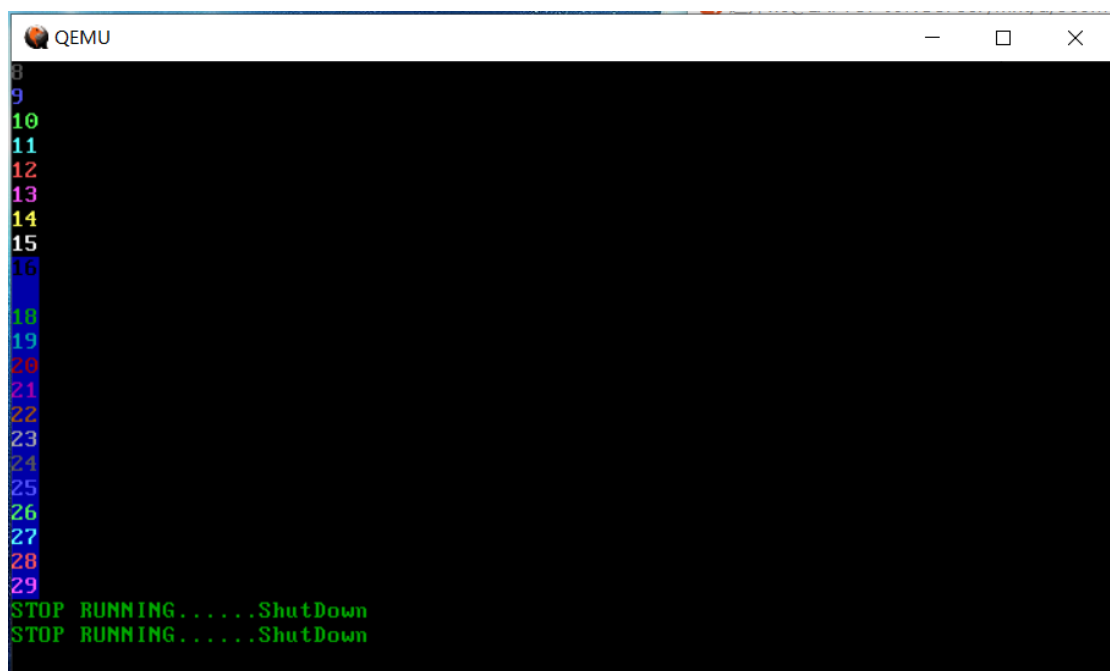
```
ws@LAPTOP-J9NGC786:/mnt/d/360MoveData/Users/asus/Desktop/os2020-labs/lab2/src/2_multiboot2myMain$ make
ld -n -T myOS/myOS.ld output/multibootheader/multibootHeader.o output/myOS/start32.o output/myOS/osStart.o output/myOS/dev/uart.o output/myOS/dev/vga.o output/myOS/i386/io.o output/myOS/printk/myPrintk.o output/myOS/printk/vsprintf.o output/userApp/main.o -o output/myOS.elf
ws@LAPTOP-J9NGC786:/mnt/d/360MoveData/Users/asus/Desktop/os2020-labs/lab2/src/2_multiboot2myMain$ export DISPLAY=:0
ws@LAPTOP-J9NGC786:/mnt/d/360MoveData/Users/asus/Desktop/os2020-labs/lab2/src/2_multiboot2myMain$ qemu-system-i386 -kern
```

 multibootheader	2020/3/18 12:5
 myOS	2020/3/19 18:4
 userApp	2020/3/20 12:4
 myOS.elf	2020/3/20 12:4

可以看到，make 命令确保源代码目录下没有不正确的.o 文件以及文件的互相依赖。它们分别链向源代码目录下的真正的 i386 所需要的真正的子目录。编译后产生的文件在图上可见，有 multiboot、start32、osStart、uart、vga、io、myPrintk、vsprintf、main、myOS 的输出文件以及标注了他们的输出目录。

### 运行和运行结果说明：

在 Ubuntu 中通过 QEMU 启动已经编译生成的 bin 文件，得到 Linux 的图形化界面运行结果，显示需要的输出。其中 VGA 输出结果是 1-29 个数字并以与该数字对应的编码的颜色渲染，同时在前后有执行的开始与结束说明。



在 linux 命令窗口的显示结果则不带有颜色，但是输出内容与前者相同。

```
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
STOP RUNNING. .... ShutDown
```

### 遇到的问题和解决方案：

1. 不会自己编写字符串的格式化函数。

通过网络工具移植。

2. 不了解如何输出到 VGA 上。

咨询助教并选择了直接用 C 语言的指针进行修改值

3. 不知道光标的实现方法。

参考老师给出的文件并且查阅网络资料完成。