Lab1 report

PB20000328 杨博涵

实验目标

编写一个头部满足MultiBoot协议的OS内核，实现VGA、UART输出。

实验原理

qemu使用的是grub bootloader，支持MultiBoot启动，只要我们的内核头部满足MultiBoot协议就能被启动。

内核内容是在vga显存处写数据，同时在uart端口上写数据。

源代码说明

multibootHeader.s : 内核汇编代码

multibootHeader.ld : ld的script file，描述output file属性

Makefile : make文件

VGA中movl $0x2f652f68, 0xB8000 是在把”OK“两个字符移到VGA显存里供输出。

串口 movb $0x7a, %al 是在将一个ascii码字符0x7a放到串口缓冲区，然后输出到stdio。

代码布局说明

multiboot_header.bin 在内存占1320字节（文件属性）。VGA内存部分是从0xB8000 开始分配的，输出一个字符需要2Byte，而一个movl指令移动4Byte，所以两个movl指令中间目标地址相差为4

编译过程说明

使用make编译生成bin，使用qemu-system-i386 -kernel multibootHeader.bin -serial stdio使用模拟器。

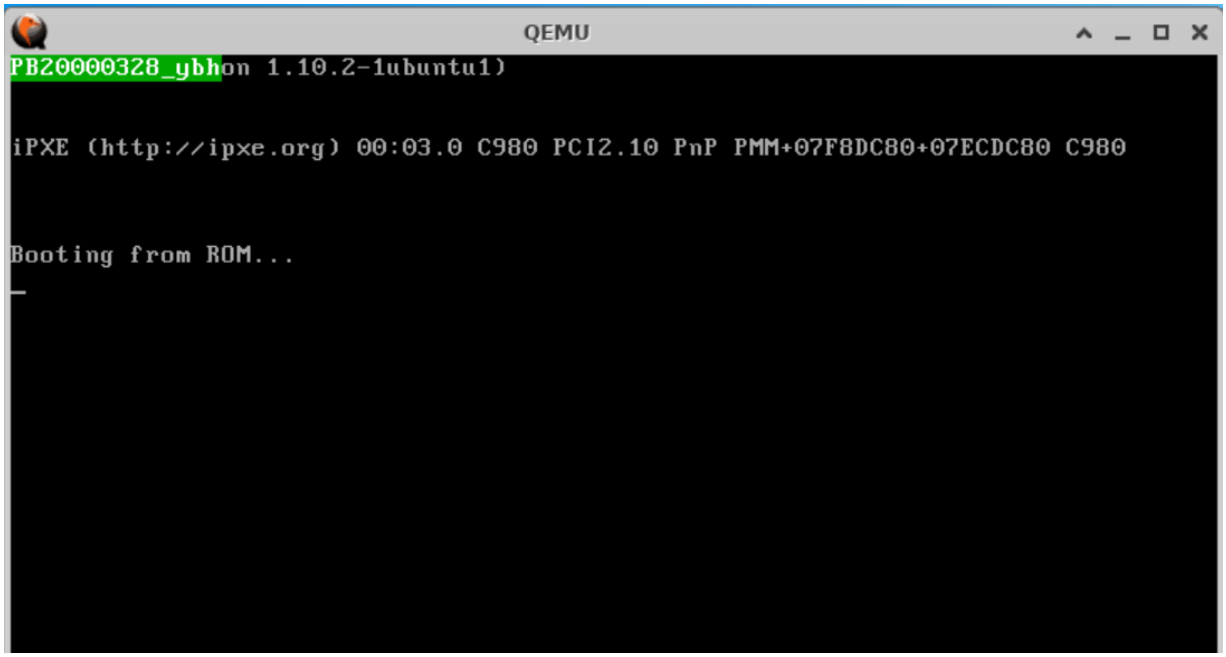
运行结果

内核没有指定存放地址，默认从0x0开始存代码。

0xB8000 ~ 0xB801B是本次实验使用到的VGA显存地址，内核中在这块地址上写了“PB20000328_ybh”字符串，同理uart是一个一个字节从串口输出到stdio。编译只需要在命令行使用make命令即可。

下图为结果截图

VGA：



UART:

```
yangbh@DESKTOP-8OP9CVU:/mnt/d/file_from_desktop/USTC_2023SP_OS/share$ make
gcc -c -m32 --pipe -Wall -fasm -g -O1 -fno-stack-protector multibootHeader.s -o
multibootHeader.o
ld -n -T multibootHeader.ld multibootHeader.o -o multibootHeader.bin
yangbh@DESKTOP-8OP9CVU:/mnt/d/file_from_desktop/USTC_2023SP_OS/share$ qemu-syste
m-i386 -kernel multibootHeader.bin -serial stdio
PB20000328_ybh
```

遇到的问题

第一次实验的耗时主要在搭建wsl2 ubuntu环境和了解gcc/ld/make等指令的用法。