lab2实验报告

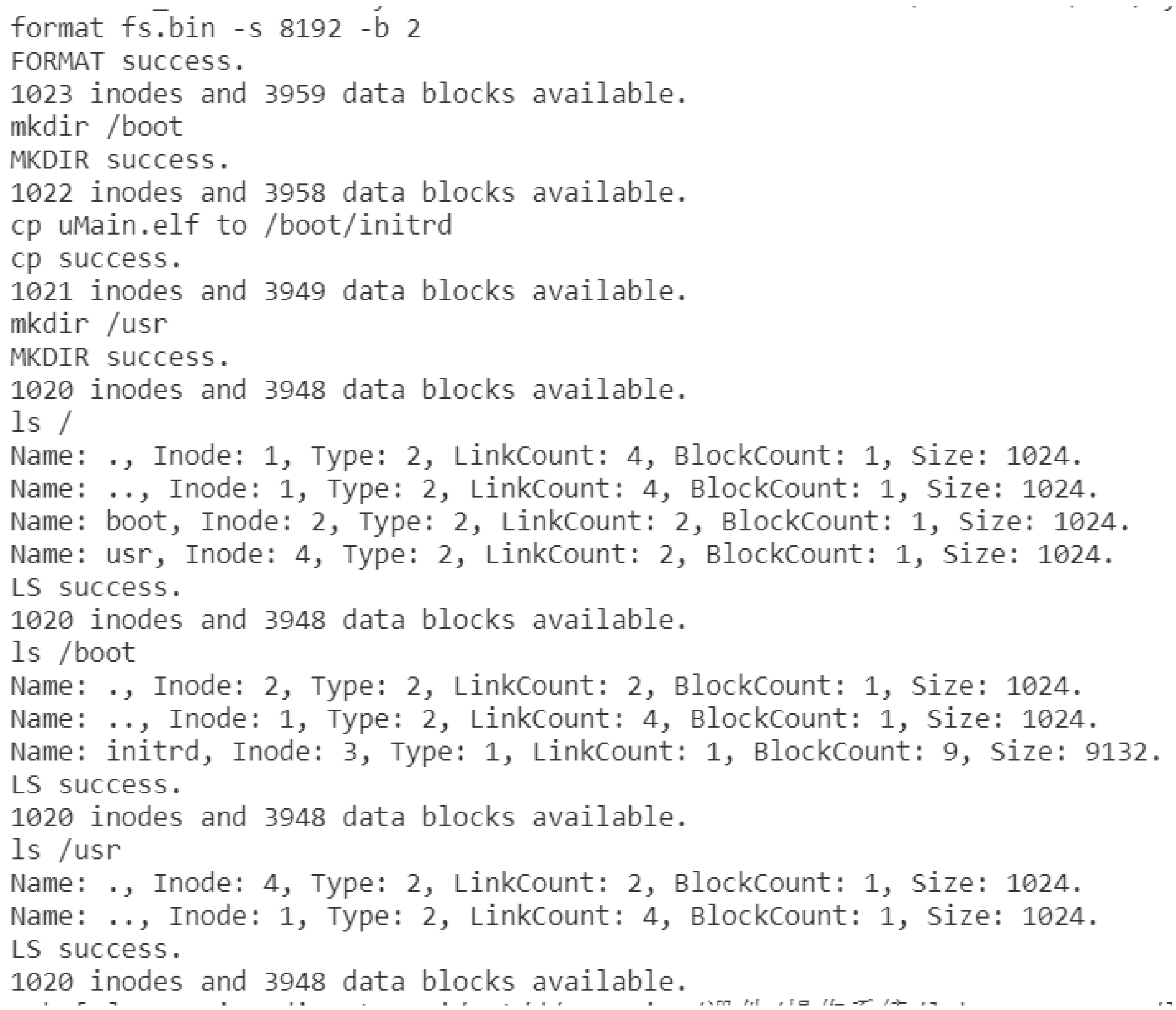
刘国涛 181860055 计算机科学与技术系

181860055@smail.nju.edu.cn

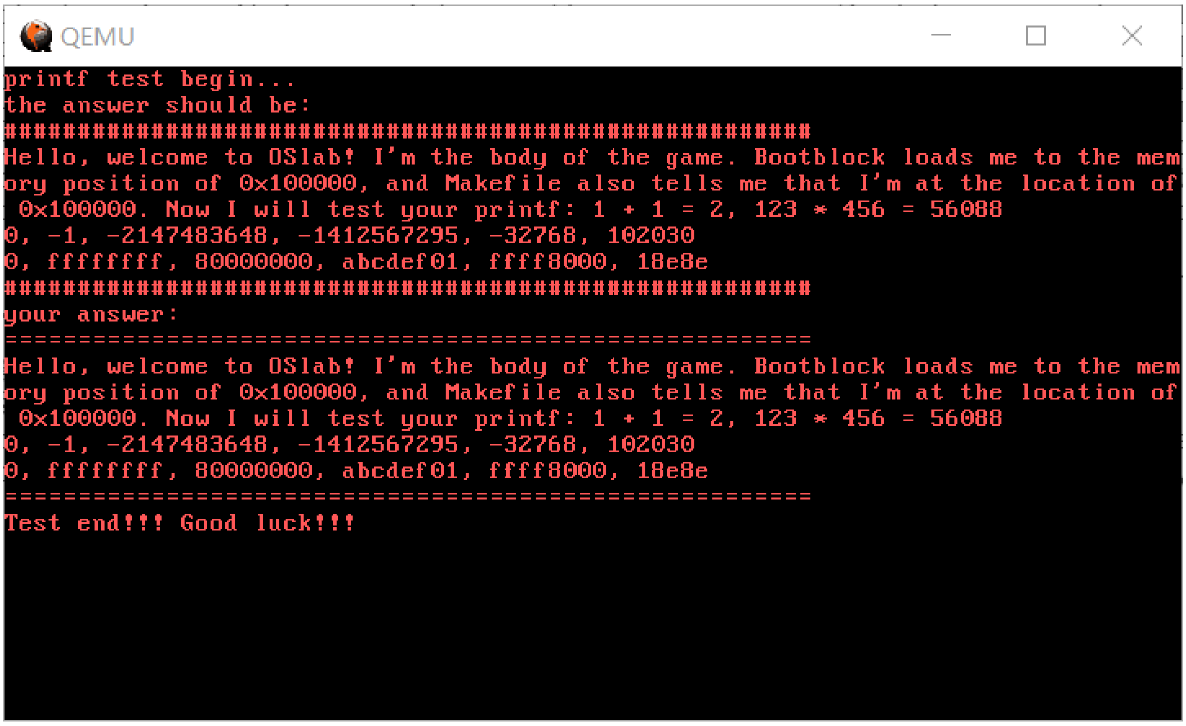
实验描述

实验进度:完成了所有实验内容

实验结果:



```
format fs.bin -s 8192 -b 2
FORMAT success.
1023 inodes and 3959 data blocks available.
mkdir /boot
MKDIR success.
1022 inodes and 3958 data blocks available.
cp uMain.elf to /boot/initrd
cp success.
1021 inodes and 3949 data blocks available.
mkdir /usr
MKDIR success.
1020 inodes and 3948 data blocks available.
ls /
Name: ., Inode: 1, Type: 2, LinkCount: 4, BlockCount: 1, Size: 1024.
Name: .., Inode: 1, Type: 2, LinkCount: 4, BlockCount: 1, Size: 1024.
Name: boot, Inode: 2, Type: 2, LinkCount: 2, BlockCount: 1, Size: 1024.
Name: usr, Inode: 4, Type: 2, LinkCount: 2, BlockCount: 1, Size: 1024.
LS success.
1020 inodes and 3948 data blocks available.
ls /boot
Name: ., Inode: 2, Type: 2, LinkCount: 2, BlockCount: 1, Size: 1024.
Name: .., Inode: 1, Type: 2, LinkCount: 4, BlockCount: 1, Size: 1024.
Name: initrd, Inode: 3, Type: 1, LinkCount: 1, BlockCount: 9, Size: 9132.
LS success.
1020 inodes and 3948 data blocks available.
ls /usr
Name: ., Inode: 4, Type: 2, LinkCount: 2, BlockCount: 1, Size: 1024.
Name: .., Inode: 1, Type: 2, LinkCount: 4, BlockCount: 1, Size: 1024.
LS success.
1020 inodes and 3948 data blocks available.
```



```
printf test begin...
the answer should be:
#######################################################
Hello, welcome to OSlab! I'm the body of the game. Bootblock loads me to the mem
ory position of 0x100000, and Makefile also tells me that I'm at the location of
 0x100000. Now I will test your printf: 1 + 1 = 2, 123 * 456 = 56088
0, -1, -2147483648, -1412567295, -32768, 102030
0, ffffffff, 80000000, abcdef01, ffff8000, 18e8e
#######################################################
your answer:
=======================================================
Hello, welcome to OSlab! I'm the body of the game. Bootblock loads me to the mem
ory position of 0x100000, and Makefile also tells me that I'm at the location of
 0x100000. Now I will test your printf: 1 + 1 = 2, 123 * 456 = 56088
0, -1, -2147483648, -1412567295, -32768, 102030
0, ffffffff, 80000000, abcdef01, ffff8000, 18e8e
=======================================================
Test end!!! Good luck!!!
```

代码修改:

1. 编写 `cp` 函数，实现复制
2. 编写 `keyboardHandle` 函数，实现键盘响应
3. 完善 `syscallPrint` 函数
4. 完善 `printf` 函数，调用转换函数使之能够格式化输出

实验感受

通过对输出函数的代码编写，进一步理解了printf所引起的系统的一系列工作流。通过中断机制，系统可以实现在输出设备上进行输出，而通过一步步的抽象封装，输出函数变得易于调用。

其中最让我感到有趣的就是printf一步步的封装过程，体现了代码低耦合的设计思想