**Log**

1. 实验目的

探索xv6的日志系统,要人为地创建一个crash来说明为什么需要日志,并删除xv6日志系统中一个效率低的部分

二、实验内容

[Homework: xv6 log (mit.edu)](https://pdos.csail.mit.edu/6.828/2018/homework/xv6-new-log.html)

三、实验步骤与实验结论

第一部分：创建问题

（1）代码替换log.c中的commit（）

BBB行导致日志中的第一个块的数据被写入第0块，而不是它应该被写到的那个块的块号。在文件创建期间，日志中的第一个块(其中的数据为新文件对应i节点所在的块数据)会将新文件的inode更新为非零类型(即标志为已被分配)。行BBB导致更新inode后的块数据被写入到第0块(永远不会读取第0块)，使磁盘上的inode仍然被标记为未分配。CCC行迫使强行崩溃。而AAA行禁止init发生这种报错行为，因为在shell启动之前会创建文件。

（2）使用下面这段代码替换log.c中的recover\_from\_log():

这个修改会使日志恢复的功能不生效。

（3）删除Makefile中QEMUEXTRA项下的-snapshot选项,这样硬盘镜像就能得到修改,然后删除掉fs.img

第二部分：解决问题

1. 修改recover\_from\_log()

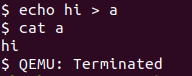
此时不会出现 panic, 但尽管执行了echo hi>a，a文件依旧是一个空文件。因为再次启动OS时， 缓冲区的内容已经丢失了，install\_trans操作并没有实际写入内容。这里也反映出必须保证文件操作的原子性。

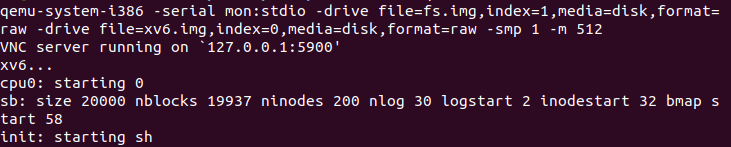
1. Q 修改磁盘块内容的流程？

A struct buf是缓冲区，想要修改磁盘块的内容，就得先读到缓冲区，对缓冲区修改完成后，再写入on-disk log，然后系统再从on-disk log中读到正确的磁盘块中。

Q struct log与log在磁盘中的起始块多个数据块的关系?

Astruct log是in-memory log，而磁盘中的log是on-disk log。







四、反思总结

如果修改了install\_trans（即直接从缓冲区将数据写入物理内存）之后，按照之前的操作，会出现no type的 panic