OS Lab5 文档

515030910211 姜子悦

Exercise1

在 env_create 中增加对 fs 环境的特殊处理，使它能使用 IO

jos 采用了以一个特殊的 env，fs env 作为文件系统的方式，所有对文件系统的调用都是通过 IPC 的方式，让这个特殊 env 来处理。

Exercise2

实现了 bc_pgfault，来完成通过缺页异常来把一个 page 读进 mem 的操作。

实现了 flush_block，完成保证 cache 和 disk 一致性的函数。Jos 对页的读写是修改 mem 中对应的 cache，而 flush_block 可以将这个 cache 刷回 disk 来真正完成写。

Exercise3

对照 free_block 实现 alloc_block，alloc_block 就是从 bitmap 中找一个空 block，把它标记为 0，并且清空数据后返回一个 block。

值得注意的是注释中提到的要记得 flush bitmap。

这个和 cse lab 基本上差不多，只是 jos 的整体实现简化了 inode。

Exercise4

实现这个文件系统需要我们来实现的最主要的部分，file_block_walk 负责找到一个给定文件某个 file block 所在的 slot，可能是一个 direct 的 data block，

有可能是一个 indirect 的。由于 jos 同样也只设计一级 indirect block，这个部分和 cse 的 lab 基本上也是一样的。

而 file_get_block，则用 file_block_walk 找到的 slot，来返回或 alloc 一个 data block。

Exercise5

Jos 文件系统的 ipc 调用方式如 guide 图所示，我们负责实现 serve 和的 devfile 即可，其他都帮我们实现好了。

这个部分先实现 read。

Serve 的 read 就是按注释做一些判断以后，直接调用帮我们实现好的 file read 即可，file read 会调用我们写好的 block 操作去写文件。

而 devfile 的 read，也很简单。把给的参数放进 ipc 调用结构中，然后调用 ipc 函数来 read 即可。

Exercise6

基本同上，把读改成写就行了

注意这次是在 devfile 步骤检查 buf 长度是否爆掉。而 read 的时候是在 serve 中检查的。

Exercise7

实现 fd_alloc，这是我们 ics cse 课闻名已久的 fd 操作。和 c 中打开一个文件，会返回一个文件描述符 fd 给我们一样。这个函数会打开文件并给我们一个

fd 句柄来操作文件。值得注意的是返回值需要 fd2num，而不是直接返回文件结构。

Exercise8

本来看上来讲 spawn，io 感觉这部分会很恐怖，感谢已经全部实现完了。

我们只需要实现一下系统调用就行了。

这个之前的 lab 实现过很多个了，获取 env 结构，然后设置相应属性即可。

要记得在 handler 中注册这个函数。

Question

1. 差不多 1 天？早上 7 点开始做，中间去上了一节 2 小时的课+半小时吃饭，下午 7 点做完。中间断断续续有些打断思路的事情。

Challenge

这次 challenge 感觉很难，没有一个会写的。cache 的 eviction policy 有点思路，记录一下每个 cache 的最近使用时间，然后 LRU 策略去替换即可。然而并没有实现。