课程报告—文件系统部分

董豪宇

小组分工

- 分工
  - 硬件 debug: 李宇轩
  - 文件系统 : 董豪宇
  - 网络支持 : 梁泽宇
- 整个组的目标比较杂，工作也相对独立

实验目标描述 (1)

- 原有目标
  - 将现有的 X86 ucore 移植到 MIPS 架构下
  - 已经完成

实验目标描述 (2)

- 现在的目标
  - 为文件系统加上创建文件的功能
  - 在此基础上将 FLASH 利用起来
- 之前对目标的理解有偏差，因此进度上落后了
  - ucore-plus 中的 yaffs2 不适用于 nor flash
  - linux 下的 jffs2 系统移植起来过于复杂
    - 适配 ucore vfs 和 dev 层的工作比较复杂
    - 锁，红黑树，进程控制块，信号 ...

实验目标描述 (3)

- 在 SFS 文件系统中扩展创建功能
  - 已经完成
  - 有改进余地
- 利用 FLASH 做永久储存
  - 正在进行

实现方案 – sfs 扩展

- SFS 扩展文件创建功能
  - 分配 dev_node, inode，并在父节点中写入相应信息
  - dev_node, inode 在文件系统中有接口
  - 写入相应信息可以通过追踪 `ls` 的过程，模仿读出相应信息自己实现

实现方案 – 利用 FLASH 做储存 (1)

- FLASH(主要是 nor FLASH) 的问题
  - 写之前要擦除
  - 擦除的时间很慢 (秒级)
  - 擦除以块计，每块 128K
  - 存在擦除寿命 (不关心)

实现方案 – 利用 FLASH 做储存 (2)

- 基本思路
  - 缓存策略
    - 避免直接写入
  - 异步擦除
    - 避免擦除等待

实现方案 – 利用 FLASH 做储存 (3)

- 缓存策略
  - 通常情况下的读写都在 RAM 上进行
  - 在合适的时机将内容写入到对应的已擦除块

实现方案 – 利用 FLASH 做储存 (4)

- 异步擦除
  - 异步擦除借鉴了 linux 下 jffs2 的做法
  - 创建内核线程
    - 检查当前是否可写入 / 擦除
    - 检查是否有脏块，可擦除则擦除 (不等待)
    - 检查是否有空块，有则写入
    - 检查是否有正在擦除的块，有则标记为已清除

实现方案 – 利用 FLASH 做储存 (5)

- 状态转换
  - CONSISTENT → DIRTY
    - 在 RAM 上有改动
  - DIRTY → CLEANING
    - 内核线程对其进行异步擦除
  - CLEANING → CLEAN
    - 擦除完成
  - CLEAN → CONSISTENT
    - 将 RAM 中的内容写入

实现方案 – 利用 FLASH 做储存 (6)

- 关于数据一致性
  - 方案的数据一致性很弱
  - 可能造成数据丢失
- 增强数据一致性
  - FLASH 不够大
  - 一份数据多份存储 对应关系会非常复杂

当前进展 – 利用 FLASH 做储存

- 在 qemu 中添加了对 FLASH 的支持
- 完成 FLASH SWAPPER
- 但是 DIRTY 标记还没有完成

未来计划 – 利用 FLASH 做储存

- 就 qemu 的修改 (包括如何使用学长的 qemu) 出一份简明的文档
- [重点] 完成 DIRTY 标记，在 qemu 上成功运行
- 上板调试
  - 可能要扩展现有的指令集

体会与收获

- 应对比较庞大的系统时经验不足

谢 谢