存储设备 IO 优化向导

发布版本：1.0

作者邮箱：cmc@rock-chips.com

日期：2018.03

文件密级：公开资料

前言

概述

产品版本

芯片名称	内核版本
全系列	4.4

读者对象

本文档（本指南）主要适用于以下工程师：

技术支持工程师

软件开发工程师

修订记录

日期	版本	作者	修改说明
2018-03-01	V1.0	陈谋春	

---

1 概述

虽然存储设备的性能在近年来也一直在稳步提升，从 RAW NAND 到 EMMC，再到现在 UFS 都在进步，但是和 CPU 相比依然是一个低俗外设，并且 IO 是不可抢占的，所以应用程序在交互中去等待 IO 会带来非常糟糕的用户体验，大部分程序员都意识到了这一点，所以绝大多数的程序尽量都会想办法改善这一点，例如预读和异步 IO 等。但是有一些场景是很难规避 IO 的影响，比如启动应用和本地音视频播放。如果在这两个场景中，还有其他程序来竞争 IO，那对用户体验来说可能是无法接受的。

2 现有的优化

我们已前面说的两个场景中的本地音视频播放[1]细化，设想有这样一个场景：某一天用户心血来潮要整理设备上的照片或其他数据，通过文件浏览器把大量的文件拷贝到一张新的 TF 卡上，这个过程是漫长的，这时候他打开一个本地视频观看，但是不幸的事情来了，视频卡顿非常严重。

碰到这种情况，工程师们肯定能理解，文件拷贝和视频播放在竞争 IO，导致解码器没有及时得到新数据，进而导致丢帧或音视频不同步。但是用户不一定能接受这个解释，他觉得为什么不能让文件拷贝的速度放慢一些，优先保证视频播放的用户体验。

这时候又轮到工程师来想办法了，在这个场景中，IO 实际上可以分两类，视频播放和文件拷贝，把其中视频播放的 IO 认为是关键 IO，问题就变成了如何避免关键 IO 拥塞。

让我们先看看万能的谷歌是怎么处理这个问题的，最新的 Android 会设置所有视频相关进程的 IO 优先级到 RT 级，这种优化的效果肯定是有的，但是还有几个问题：

- **效果有限**：RT 只能改善，而不能避免关键 IO 的拥塞，因为文件拷贝的进程会更频繁的发起 IO 请求，所以经常会碰到这种情况，视频播放的 IO 请求到来的时候，设备驱动正在执行文件拷贝的 IO，由于 IO 不可抢占，此时只能等这个请求完成，视频播放的请求才能得到服务
- **对第三方应用支持有限**：如果第三方应用是通过 MediaCodec 或 OMX 来实现音视频播放的，此时流媒体数据的请求是应用自己发起的，并不会被自动配置成 RT 级
- **没有考虑启动应用的场景**: 实测在文件拷贝的同时去启动应用，速度慢了近两倍

3 如何改进

3.1 结合 Cgroup 控制

Kernel 除了 IO 优先级以外，还有一个 BLK Cgroup 控制，目前支持两种控制策略，权重和节流。二者都是通过标准的 Cgroup 方式提供配置接口的，这里就不详细描述了。把进程分组以后，可以给不同的分组配置不同的权重，例如给视频播放进程配置 1000 的权重，给文件拷贝分配 10 的权重，则内核会尽量按 99：1 的比例来分配存储设备的带宽，当然实际上的带宽比例不会是这样的，因为两个进程发起 IO 的频度是不一样，内核只能保证短时间内的带宽按配置的权重来分配。而节流的话，则是可以给每个分组每个设备设置一个性能阈值，内核来保证性能不超过这个阈值，例如给文件拷贝分配 2MB/s 的读带宽，则内核会严格按这个指标分配带宽，即使此时设备带宽还没有用完。

3.2 提升第三方应用兼容性

大部分媒体应用都是直接调用 MediaPlayer 来实现播放，这样会通过 MediaServer 来读取流媒体数据，而这个进程 Android 已经配置了 RT 优先级。但是还有一部分应用是直接调用 MediaCodec 来实现播放的，比如特殊的加密格式视频，又或则为了实现帧合成等特殊处理，在这种情况下是由视频应用本身来读取流媒体数据，然后再送给 MediaCodec 的。为了改善这个问题，可以在 dequeueInputBuffer 函数中把当前线程的 IO 优先级和权重都配置到最高。

3.3 识别启动应用的场景

所有的应用启动都要经过 Activity Manager Service(后面简称 AMS)，所以在 AMS 的 startActivity 中识别启动应用开始，在 Activity.onStart 中认为启动应用结束。

3.4 限制文件拷贝性能

为了进一步提升前台应用和媒体应用的用户体验，还需要降低文件拷贝的 IO 性能，即在识别到启动应用和音视频播放的场景后，会同时根据一个白名单主要是各种文件浏览器）降低特定活动进程的 IO 优先级和权重，同时发出一个 BROADCAST_IO_JANK 消息，各种系统服务如 mtp 服务、下载器等可以在收到这个消息的时候降低 IO 优先级和权重

3.5 提供框架层接口

框架层提供 IO 优先级和权重的配置接口，但是普通应用只支持降低优先级和权重，只有 system 权限的应用可以提高优先级和权重。

4 整体流程

- Step1: 开机启动过程中，init 会为每个系统服务设定 IO 优先级和权重，其中 Media 相关的进程全部都是 RT 和最高权重，debuggerd、logcatd 等都是 IDLE 和最低权重
- Step2: AMS 会在 Activity 的不同生命周期，设置不同的 IO 优先级和权重，其中 TOP_APP > FORGROUND > BACKGROUD
- Step3: 媒体播放过程中会设置一个属性 media.codec.running=true，来表示当前正在进行音视频编解码，并且会发出 BROADCAST_IO_JANK 消息，通知系统服务调低 mtp、下载器和其他白名单中应用程序的 IO 优先级和权重到最低，并在播放结束后延迟发出 BROADCAST_IO_RESUME（如果在延迟时间内触发新的 BROADCAST_IO_JANK，会先移除这个延迟消息），通知系统服务恢复到正常优先级和权重
- Step4: 在应用启动过程中，如果 media.codec.running==false，则会发出 BROADCAST_IO_JANK 消息，通知系统服务调低 mtp、下载器和其他白名单中应用程序的 IO 优先级和权重到最低，并在启动结束后延迟发出 BROADCAST_IO_RESUME（如果在延迟时间内触发新的 BROADCAST_IO_JANK，会先移除这个延迟消息），通知系统服务恢复到正常优先级和权重

  根据本文介绍的这些方法，整理了一个参考补丁:*blkio_patch.tar.bz2*，工程师可以根据需求定制和修改。

5 更多改进

1. 在线音视频缓存流媒体数据一般都是直接存在内存中的，一是可以保证性能，二是Flash设备的寿命有限，所以不需要考虑IO优化↩