

sadamoo的专栏

目录视图

摘要视图

RSS 订阅

个人资料



sadamoo

访问：131451次

积分：1705

等级：

BLOG > 4

排名：第16466名

原创：7篇    转载：211篇

译文：0篇    评论：5条

文章搜索

文章分类

- [linux 设备驱动模型](#) (13)
- [android camera](#) (28)
- [linux input](#) (3)
- [linux i2c](#) (12)
- [linux lcd](#) (1)
- [linux ipc](#) (3)
- [android input](#) (1)
- [linux 内存管理](#) (19)
- [android multimedia](#) (36)
- [alsa](#) (15)
- [android audio](#) (4)
- [android class](#) (1)
- [android reboot](#) (2)
- [android miracast](#) (6)
- [linux socket](#) (5)
- [linux pipe](#) (2)
- [android wifi](#) (3)
- [mp4](#) (3)

【公告】博客系统优化升级

【收藏】Html5 精品资源汇集

博乐招募开始啦

android camera HAL v3.0详细介绍（二）

2015-12-21 11:11

925人阅读

评论(0)

收藏

举报

分类：[android camera](#) (27)

3.Startup and expected operation sequence

这段描述了使用camera API的详细步骤。其中涉及到的结构体和函数请参考文  
件：[platform/hardware/libhardware/include/hardware/camera3.h](#)

1. Framework层调用函数camera\_module\_t->common.open(), 将返回一个hardware\_device\_t类型的结构体。
2. Framework层检查字段hardware\_device\_t->version, 根据版本信息, 实例化一个适合这个版本的camera硬件设备的句柄。例如版本号是CAMERA\_DEVICE\_API\_VERSION\_3\_0, 则这个设备将被转化为camera3\_device\_t。
3. Framework层调用函数camera3\_device\_t->ops->initialize(), 并传递了framework层的回调函数指针。这个函数只能被调用一次, 且在调用函数open()之后, 在其他函数被调用之前。
4. Framework层调用函数camera3\_device\_t->ops->configure\_streams(), 向这个HAL层设备传递了输入输出的流信息。
5. Framework层分配grallocbuffer; 调用函数camera3\_device\_t->ops->register\_stream\_buffers(), 至少使用一个configure\_streams中列举的输出流。同一个流只能被注册一次。
6. Framework层通过调用camera3\_device\_t->ops->construct\_default\_request\_settings()获取用例的默认设置。这个在第三步之后任意地方进行调用。
7. Framework层使用默认设置集合中某一套设置, 且保证之前注册了至少一个输出流, 创建并向HAL层发第一个捕获请求。这个请求将通过调用函数camera3\_device\_t->ops->process\_capture\_request()发送到HAL层。HAL必须阻止函数返回, 直到HAL准备好接收下一个请求。
8. Framework层连续地提交请求。可能会调用函数register\_stream\_buffers()来注册没有注册过的流, 调用函数construct\_default\_request\_settings获取其他用例所需的默认设置。
9. 当一个请求的捕获开始时 (sensor开始曝光), HAL层将调用函数camera3\_callback\_ops\_t->notify()通知上层SHUTTER事件, 其中包括sensor开始曝光的帧号和时间戳。调用函数process\_capture\_result()处理这个帧号对应的数据之前, HAL层必须发出SHUTTER通知。
10. 流水线持续一些时间后, HAL层开始使用函数camera3\_callback\_ops\_t->process\_capture\_result()向framework层返回处理完的图像数据。返回结果的次序与提交请求的次序完全一致。多个请求可以被一次提交, 但这取决于camera HAL层设备的流水线深度。
11. 工作一段时间之后, framework层可能会停止提交新的请求, 等待其他请求被完成 (所有buffer被填充, 所有结果被返回), 然后再次调用函数configure\_streams()。这是为一组新的输入输出流重启camera硬件和流水线。前面配

## 文章存档

2016年06月 (8)  
2016年04月 (1)  
2016年03月 (4)  
2015年12月 (9)  
2015年08月 (1)

展开

## 阅读排行

Android中基于NuPlayer (4525)  
我对linux理解之i2c 二 (3983)  
Android WifiDisplay分析 (3090)  
Android WifiDisplay分析 (2999)  
闲聊linux中的input设备 (2810)  
Android4.0 input touch解 (2121)  
device\_register和驱动dri (2091)  
android audio (2079)  
Android WifiDisplay分析 (1975)  
Bootloader之uBoot简介 (1959)

## 评论排行

Android4.0 input touch解 (1)  
Linux内存寻址和内存管理 (1)  
android多媒体框架之流媒体 (1)  
WifiP2pService的启动以及 (1)  
ALSA声卡驱动中的DAPM (1)  
Linux的i2c驱动详解 (0)  
基本的数据结构学习笔记 (0)  
Linux设备驱动模型学习之 (0)  
使用Camera2 替代过时 (0)  
Linux设备驱动模型之底层 (0)

## 推荐文章

\*Android RocooFix 热修复框架  
\* android6.0源码分析之Camera API2.0下的初始化流程分析  
\*Android\_GestureDetector手势滑动使用  
\*Android MaterialList源码解析  
\*Android官方开发文档Training系列课程中文版: 创建自定义View之View的创建

## 最新评论

WifiP2pService的启动以及P2P全村人的希望: 您好, 我想请问一下, 如何能够设置自己手机发出去的device name。我在做一个小程序, 希望手机检...  
ALSA声卡驱动中的DAPM详解之wsc\_168: 您好: 现在正在移植wm8962的驱动, 遇到了一些问题, 向您请教一些问题。wm8962芯片已经...  
Android4.0 input touch解析尹之梦: 我碰到的好像是这个触

置的一些流可能会被重复使用; 如果流的buffer已经注册到了HAL层, 它们将不再被注册。如果有一个被注册的输出流还存在, 则framework层将从第七步重新开始 (否则, 将从第五步开始)。

12. Framework层将调用函数camera3\_device\_t->common->close()结束camera会话。当framework层没有其他调用时, 可以在任何时间调用这个函数, 尽管这个调用会阻塞, 直到所有正在处理的捕获被完成 (所有结果被返回, 所有buffer被填充)。函数close()返回之后, 不允许HAL调用camera3\_callback\_ops\_t的任何函数。一旦函数close()被调用, framework层将不能调用其他任何HAL层设备函数。

13. 如果发生错误或者其他异步事件, HAL层必须调用函数camera3\_callback\_ops\_t->notify()告知上层对应的错误或者事件信息。一个与设备相关的致命错误被通知上层后, HAL应该像被调用了函数close()一样。但是, 在调用函数notify()之前, HAL必须取消或者完成所有未结束的数据捕获操作, 以便致命错误被上报之后, framework不会收到设备的任何回调函数。除了函数close(), 致命错误信息发出后, 其他函数只能返回-ENODEV或者NULL。

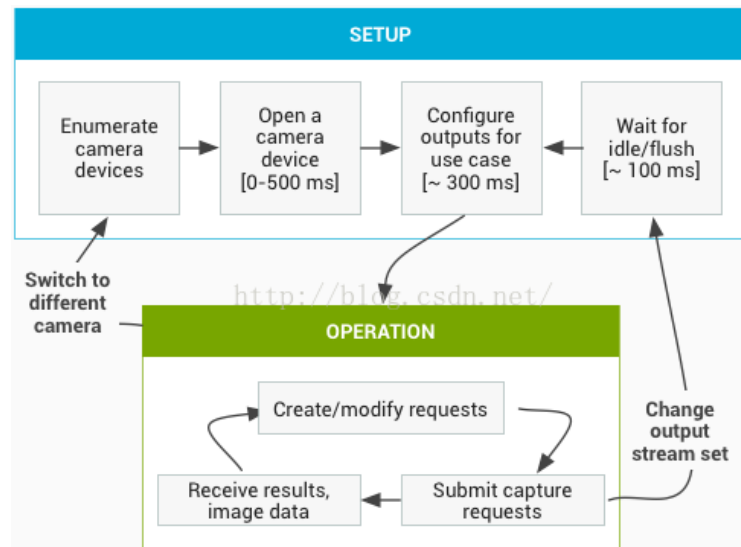


Figure4. Camera operational flow

## 4.Operational modes

Camera HAL层v3版本的设备实现两种可能的操作模式之一: limited模式和full模式。新的高端设备预计会支持full模式。Limited模式与camera HAL层v1版本一样, 对硬件需求很低, 用于旧的或者低价设备。Full模式是limited模式的超集, 如上面所述, 它们有着基本上一样的操作流程。

Camera HAL层必须使用静态元数据Android.info.supportedHardwareLevel指明其所支持的模式。0表示支持limited模式, 1表示支持full模式。

简单来讲, limited模式设备不允许应用程序控制捕获数据信息的设置 (3A控制除外), 大分辨率图像的高帧率捕获, raw数据的获取, 或者上面所说的最大视频分辨率的YUV输出流 (对于大图像只支持JPEG)。

Limited模式行为的细节如下:

- Limited模式的设备不需要实现请求配置与实际捕获图像数据的完全匹配。相反, 将来有时改变配置将更高效, 可能也不需要配置必须对应一个同样的输出帧。快速改变配置可能会使得某些配置从来没有被使用过。但是, 有高分辨率 (大于1080P) 输出buffer的捕获必须使用指定的配置 (处理速度的描述如下)。

- 在limited模式下, 有高分辨率 (大于1080P) 输出buffer的捕获在函数process\_capture\_request()中会阻塞, 直到所有输出buffer都被填充。Full模式HAL层设备必须按照静态元数据中指定的速率, 处理相应像素格式的高分辨率请求的序列。HAL层调用函数process\_capture\_result()产生输出结果; framework层需要为函数process\_capture\_request()做好准备, 然后阻塞, 直到limited模式设备的高分辨率捕获请求被函数process\_capture\_result()执行完毕。

摸屏的问题，那到底该怎么改啊，求指教？

[Linux内存寻址和内存管理](#)  
zq606: 学习了

· Limited设备不需要支持大多数的配置、结果、静态信息。只有下面的配置期待被limited模式HAL层设备所支持：

- o android.control.aeAntibandingMode(controls)
- o android.control.aeExposureCompensation(controls)
- o android.control.aeLock(controls)
- o android.control.aeMode(controls)
- o [OFF means ON\_FLASH\_TORCH]
- o android.control.aeRegions(controls)
- o android.control.aeTargetFpsRange(controls)
- o android.control.afMode(controls)
- o [OFF means infinity focus]
- o android.control.afRegions(controls)
- o android.control.awbLock(controls)
- o android.control.awbMode(controls)
- o [OFF not supported]
- o android.control.awbRegions(controls)
- o android.control.captureIntent(controls)
- o android.control.effectMode(controls)
- o android.control.mode(controls)
- o [OFF not supported]
- o android.control.sceneMode(controls)
- o android.control.videoStabilizationMode(controls)
- o android.control.aeAvailableAntibandingModes(static)
- o android.control.aeAvailableModes(static)
- o android.control.aeAvailableTargetFpsRanges(static)
- o android.control.aeCompensationRange(static)
- o android.control.aeCompensationStep(static)
- o android.control.afAvailableModes(static)
- o android.control.availableEffects(static)
- o android.control.availableSceneModes(static)
- o android.control.availableVideoStabilizationModes(static)
- o android.control.awbAvailableModes(static)

- o android.control.maxRegions(static)
- o android.control.sceneModeOverrides(static)
- o android.control.aeRegions(dynamic)
- o android.control.aeState(dynamic)
- o android.control.afMode(dynamic)
- o android.control.afRegions(dynamic)
- o android.control.afState(dynamic)
- o android.control.awbMode(dynamic)
- o android.control.awbRegions(dynamic)
- o android.control.awbState(dynamic)
- o android.control.mode(dynamic)
- o android.flash.info.available(static)
- o android.info.supportedHardwareLevel(static)
- o android.jpeg.gpsCoordinates(controls)
- o android.jpeg.gpsProcessingMethod(controls)
- o android.jpeg.gpsTimestamp(controls)
- o android.jpeg.orientation(controls)
- o android.jpeg.quality(controls)
- o android.jpeg.thumbnailQuality(controls)
- o android.jpeg.thumbnailSize(controls)
- o android.jpeg.availableThumbnailSizes(static)
- o android.jpeg.maxSize(static)
- o android.jpeg.gpsCoordinates(dynamic)
- o android.jpeg.gpsProcessingMethod(dynamic)
- o android.jpeg.gpsTimestamp(dynamic)
- o android.jpeg.orientation(dynamic)
- o android.jpeg.quality(dynamic)
- o android.jpeg.size(dynamic)
- o android.jpeg.thumbnailQuality(dynamic)
- o android.jpeg.thumbnailSize(dynamic)
- o android.lens.info.minimumFocusDistance(static)
- o android.request.id(controls)

- o android.request.id(dynamic)
- o android.scaler.cropRegion(controls)
- o [ignores (x,y),assumes center-zoom]
- o android.scaler.availableFormats(static)
- o [RAW not supported]
- o android.scaler.availableJpegMinDurations(static)
- o android.scaler.availableJpegSizes(static)
- o android.scaler.availableMaxDigitalZoom(static)
- o android.scaler.availableProcessedMinDurations(static)
- o android.scaler.availableProcessedSizes(static)
- o [full resolution notsupported]
- o android.scaler.maxDigitalZoom(static)
- o android.scaler.cropRegion(dynamic)
- o android.sensor.orientation(static)
- o android.sensor.timestamp(dynamic)
- o android.statistics.faceDetectMode(controls)
- o android.statistics.info.availableFaceDetectModes(static)
- o android.statistics.faceDetectMode(dynamic)
- o android.statistics.faceIds(dynamic)
- o android.statistics.faceLandmarks(dynamic)
- o android.statistics.faceRectangles(dynamic)
- o android.statistics.faceScores(dynamic)

#### 5. Interaction between the application capturerequest, 3A control, and the processing pipeline

根据3A控制模块的配置，camera流水线会忽略应用程序请求中的一些参数，而使用3A控制模块提供的值代替。例如，当自动曝光开启时，曝光时间，帧率，sensor的敏感参数都由3A算法控制，应用程序提供的值全被忽略。3A事务为帧所设置的参数值必须包含在输出的元数据中。下面的表格描述了3A模块的不同模式和被这些模式所控制的属性。属性的定义见文件[platform/system/media/camera/docs/docs.html](http://platform/system/media/camera/docs/docs.html)。

Parameter	State	Properties controlled
android.control.aeMode	OFF	None
	ON	android.sensor.exposureTime android.sensor.frameDuration android.sensor.sensitivity    android.lens.aperture (if supported) android.lens.filterDensity (if supported)

	ON_AUTO_FLASH	Everything is ON, plus android.flash.firingPower, android.flash.firingTime, and android.flash.mode
	ON_ALWAYS_FLASH	Same as ON_AUTO_FLASH
	ON_AUTO_FLASH_RED_EYE	Same as ON_AUTO_FLASH
android.control.awbMode	OFF	None
	WHITE_BALANCE_*	android.colorCorrection.transform. Platform-specific adjustments if android.colorCorrection.mode is FAST or HIGH_QUALITY.
android.control.afMode	OFF	None
	FOCUS_MODE_*	android.lens.focusDistance
android.control.videoStabilization	OFF	None
	ON	Can adjust android.scaler.cropRegion to implement video stabilization
android.control.mode	OFF	AE, AWB, and AF are disabled
	AUTO	Individual AE, AWB, and AF settings are used
	SCENE_MODE_*	Can override all parameters listed above. Individual 3A controls are disabled.

所列的3A算法的控制大部分都是与旧的API参数一一匹配（例如曝光补偿，场景模式，或者白平衡模式）。

图2中的图像处理模块的控制都是基于一个相似的原则，通常每个模块有三中模式：

- OFF：这个处理模块不使能。去马赛克，颜色校正和tone曲线调整模块必须使能。
- FAST：在这种模式，与off模式相比，处理模块不会降低输出帧率，但是对于产生高质量输出时就不受这个限制了。典型地，它被用于预览或者视频录制模式，或者静态图片的快速捕获。在一些设备中，这种模式与OFF模式一样（不工作就不会降低帧率）；在一些设备中，它与HIGH\_QUALITY模式一样（高质量处理也不会降低帧率）。
- HIGHQUALITY：在这种模式中，处理模块产生最高质量的结果，如果需要会降低输出帧率。典型地，它被用于高质量静态图片的捕获。一些包括手动控制的模块，可以替代FAST或者HIGHQUALITY。例如，图像校正模块支持一个颜色转换矩阵，而tone曲线调整支持一个任意全局tone映射曲线。

一个camera子系统能够支持的最大帧率是多个元素的函数：

- 输出图像流所需要的分辨率
- Imager对binning / skipping模式的支持
- imager接口的带宽
- 各个ISP处理模块的带宽

因为对于不同的ISP和sensor，这些因子有很大的变化，camera HAL层接口想要抽象一个尽可能简单的带宽限制模型。这个模型有如下特性：

- Sensor总是输出能满足应用程序请求的输出流大小的最小分辨率图像数据。这个最小分辨率至少与被请求的最大输出流大小一样大。
- 因为任何情况可能会用到被配置的输出流的几个或者所有，所以sensor和ISP必须被配置为能够将一个图像同时缩放到所有输出流中。
- JPEG流就像请求的被处理的YUV流；在请求中，它们直接被当作JPEG流来引用。
- JPEG处理器能够同时处理camera流水线的剩余部分，但不能在同一个时刻处理多于一张图片。

(全文完)

顶

0

踩

0

上一篇

android camera HAL v3.0详细介绍（一）

下一篇

android camera接口介绍

我的同类文章

android camera (27)

• 使用Camera2 替代过时的C...

2016-06-09

阅读 113

• Android Camera从Camera ...

2016-06-02

阅读 67

• Android Camera API2中采...

2016-06-02

阅读 77

• Android5.1中surface和Cpu...

2016-06-01

阅读 56

• Android Camera HAL V3 Ve...

2016-03-03

阅读 137

• Android Camera HAL3中预...

2016-03-01

阅读 356

• Android4.2.2 Camer系统架...

2016-06-03

阅读 46

• Android Camera HAL3中预...

2016-06-02

阅读 84

• Android Camera HAL3中拍...

2016-06-02

阅读 51

• Android Camera API2.0下全...

2016-03-03

阅读 289


• Android Camera API2中采...

2016-03-03

阅读 258


更多文章

参考知识库



Android知识库

12836 关注 | 1500 收录



算法与数据结构知识库

1732 关注 | 2466 收录

猜你在找

- 数据结构和算法

数据结构基础系列(1): 数据结构和算法

Android之数据库详解

以性别预测为例，谈谈数据挖掘中常见的分类算法

Android底层技术：HAL驱动开发
- Android Camera HAL V3 Vendor Tag及V1V3参数转换

Android Camera HAL V3 Vendor Tag及V1V3参数转换

Android camera子系统HAL层介绍集锦

Android camera子系统HAL层介绍集锦

Android overlay 学习 二 Android camera preview and take

控制算法

mac

awb

校正方法

android下载

win7纯净版

html5教

查看评论

暂无评论

您还没有登录,请[\[登录\]](#)或[\[注册\]](#)

\* 以上用户言论只代表个人观点，不代表CSDN网站的观点或立场

核心技术类目												
全部主题	Hadoop	AWS	移动游戏	Java	Android	iOS	Swift	智能硬件	Docker	OpenStack		
VPN	Spark	ERP	IE10	Eclipse	CRM	JavaScript	数据库	Ubuntu	NFC	WAP	jQuery	
BI	HTML5	Spring	Apache	.NET	API	HTML	SDK	IIS	Fedora	XML	LBS	Unity
Splashtop	UML	components	Windows Mobile	Rails	QEMU	KDE	Cassandra	CloudStack	FTC			
coremail	OPhone	CouchBase	云计算	iOS6	Rackspace	Web App	SpringSide	Maemo				
Compuware	大数据	aptech	Perl	Tornado	Ruby	Hibernate	ThinkPHP	HBase	Pure	Solr		
Angular	Cloud Foundry	Redis	Scala	Django	Bootstrap							

[公司简介](#) | [招贤纳士](#) | [广告服务](#) | [银行汇款帐号](#) | [联系方式](#) | [版权声明](#) | [法律顾问](#) | [问题报告](#) | [合作伙伴](#) | [论坛反馈](#)