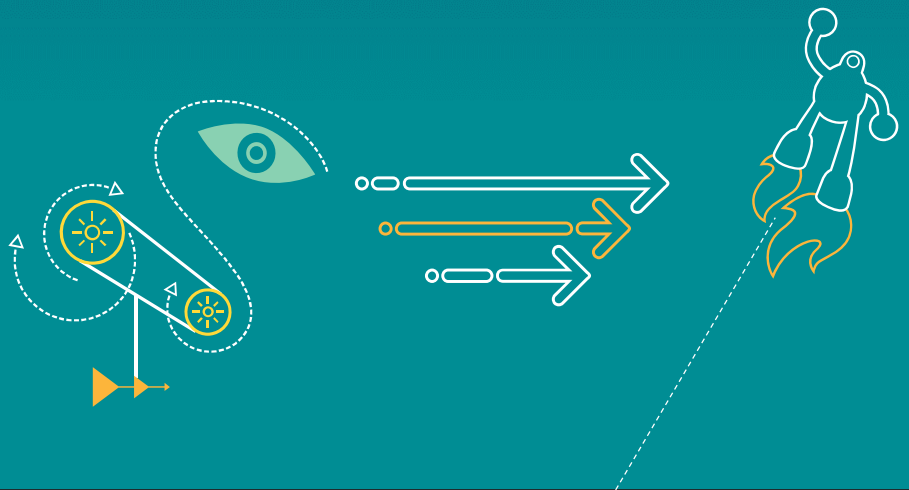
高通多媒体技术期刊　20150311

Qualcomm Technologies, Inc.



Confidential and Proprietary – Qualcomm Technologies, Inc.

Revision History

Revision	Date	Description
A	Mar 2015	Initial release

Note: There is no Rev. I, O, Q, S, X, or Z per Mil. standards.

Contents

- FD leaking issue in APK
- Camera
  - Android 5.0 Lollipop camera log丢失问题
  - MSM8994/MSM8992 I2C LED flash 驱动实现
  - Android 5.0 CTS camera 测试常见问题
  - Camera Power Debug

FD leaking issue in APK

FD leaking issue in APK

- 在最近一段时间的case处理中我们经常碰到一类问题是因为APK中存在FD泄露引起的，从log上主要表现为一下两方面:
  - 一， 跟dequeueBuffer issue相关的error log info

01-06 16:50:19.045 W/Adreno-EGLSUB(26396): <DequeueBuffer:700>: **dequeue native buffer fail**: Success, **buffer=0x5f67aa90, handle=0x0**
01-06 16:50:19.045 E/BufferQueue(227): [com.android.incallui/com.android.incallui.InCallActivity] dequeueBuffer: can't dequeue multiple buffers without setting the buffer count
01-06 16:50:19.045 W/Adreno-EGLSUB(26396): <DequeueBuffer:700>: **dequeue native buffer fail**: Invalid argument, **buffer=0x0, handle=0x0**
01-06 16:50:19.045 W/Adreno-EGL(26396): <qeglDrvAPI_eglSwapBuffers:3710>: EGL_BAD_SURFACE
01-06 16:50:19.045 W/HardwareRenderer(26396): EGL error: EGL_BAD_SURFACE
01-06 16:50:19.045 W/HardwareRenderer(26396): Mountain View, we've had a problem here. Switching back to software rendering.
01-06 16:50:19.115 E/Surface (26396): dequeueBuffer: IGraphicBufferProducer::**requestBuffer failed: 1717840517**
01-06 16:50:19.115 E/Surface (26396): dequeueBuffer failed (Unknown error -1717840517)
01-06 16:50:19.125 E/Surface (26396): Surface::unlockAndPost failed, no locked buffer
01-06 16:50:19.125 E/ViewRootImpl(26396): Could not unlock surface
01-06 16:50:19.125 E/ViewRootImpl(26396): java.lang.IllegalArgumentException

FD leaking issue in APK

- 二，跟fence相关的error log info

  02-05 08:29:40.589 W/Adreno-EGL(2709): <qeglDrvAPI_**eglDupNativeFenceFDANDROID**:6730>: EGL_BAD_PARAMETER
  02-05 08:29:40.589 E/GLConsumer(2709): [unnamed-2709-183] **syncForReleaseLocked: error dup'ing native fence fd: 0x3000**
  02-05 08:29:40.589 W/System.err(2709): java.lang.RuntimeException: Error during updateTexImage (see logcat for details)
  02-05 08:29:40.609 E/SurfaceTextureScreenNail(2709): updateTexImage failed
  02-05 08:29:40.609 E/BufferQueue(256): **[SurfaceView] queueBuffer: fence is NULL**
  02-05 08:29:40.609 E/Surface (2709): **queueBuffer: error queuing buffer to SurfaceTexture, -22**
  02-05 08:29:40.609 W/Adreno-EGLSUB(2709): <SwapBuffers:1324>: failed to queueBuffer
  02-05 08:29:40.609 W/Adreno-EGL(2709): <qeglDrvAPI_eglSwapBuffers:3799>: EGL_BAD_SURFACE

- 01-08 03:37:36.130 253 1790 E BufferQueue: [SurfaceView] queueBuffer: fence is NULL
  01-08 03:37:36.130 17902 7790 E Surface : queueBuffer: error queuing buffer to SurfaceTexture, -22
  01-08 03:37:36.130 17902 7790 W Adreno-EGLSUB: <SwapBuffers:1324>: failed to queueBuffer
  01-08 03:37:36.130 17902 7790 W Adreno-EGL: <qeglDrvAPI_eglSwapBuffers:3799>: EGL_BAD_SURFACE
  01-08 03:37:38.880 17902 7790 W Adreno-GSL: <gsl_ldd_control:412>: ioctl fd 81 code 0xc0140933 **(IOCTL_KGSL_TIMESTAMP_EVENT) failed: errno 22 Invalid argument**
  01-08 03:37:38.880 17902 7790 W Adreno-EGLSUB: **<SwapBuffers:1318>: gsl_device_3d_add_fence_event failed**
  01-08 03:37:38.880 17902 7790 W Adreno-EGL: <qeglDrvAPI_eglSwapBuffers:3799>: EGL_BAD_SURFACE

- W Adreno-GSL: <gsl_ldd_control:416>: ioctl fd 66 code 0xc0140933 **(IOCTL_KGSL_TIMESTAMP_EVENT) failed: errno 24 Too many open files**
  W Adreno-EGLSUB: <SwapBuffers:1338>: **gsl_device_3d_add_fence_event failed**

- <3>[3947.107624] kgsl kgsl-3d0: |**kgsl_add_fence_event| invalid fence fd**
  <3>[3973.297918] kgsl kgsl-3d0: |**kgsl_add_fence_event| invalid fence fd**

FD leaking issue in APK

- 背景知识
- 在Linux系统中，每个进程最多能打开的文件个数默认是1024，因此当进程打开的文件数接近或者到达这个限制，新的文件打开操作就会失败。
- 不光是文件打开操作，所有需要分配文件描述符(File Descriptor)的操作，比如用户空间的文件的create/open操作, 打开文件的dup()操作, fence的create和merge操作，socket的创建等，以及在内核空间调用get_unused_fd() (get_unused_fd_flags(0)) 都会失败。
- 在Linux系统中，可以通过通过ulimit –a 命令来查看系统的一些资源限制情况。ulimit 还有其他用法
- 查看打开文件数目限制　　ulimit -n
- 修改打开文件数目限制　　ulimit -n 32768

- 在基于Linux的Android 系统，通过文件描述符(FD) 在不同进程，以及用户进程和内核之间共享Buffer 和Fence。
- 调用dequeueBuffer()时，如果当前Buffer还没有分配，就会先调用requestBuffer()去分配新的Buffer, 同时还要创建一个native handle 来关联这个Buffer。每一个native handle在创建时需要分配2 个FD, 一个FD跟Buffer本身关联,另一个FD跟 metadata相关联。
- 每一个Fence在创建(create)时也会分配一个FD， Fence 复制(duplicate)和合并(merge)也会产生一个新的FD。
- 此外其他一些常规的占用文件描述符(FD)的情况：
  - 每一个打开的文件(File) 会占用一个FD。
  - 每一个网络套接字(Socket)会占用一个FD。
  - 每打开一个管道(Pipe) 也会占用一个FD。
  - 每一个设备节点，Sysfs 节点打开都会占用一个FD。
  - 每一个进程在创建初始化时会默认占用三个FD，分别对应标准输入，标准输出和标准出错文件。

FD leaking issue in APK

运行如下命令来查看当前每个进程打开的文件:

adb shell lsof

还可以运行如下脚本 **go_lsof_loop.sh (next page)** 来监控进程打开的文件:

sh go_lsof_loop.sh>lsof_result.txt

我们可以检查是否存在某个进程打开的文件数目持续增加，如果确实持续增加，那么可以确定APK存在FD 泄露问题，需要从APK 层面检查。

查看lsof 输出结果:

COMMAND	PID	USER	FD	TYPE	DEVICE	SIZE/OFF	NODE	NAME
mediaserv	31306	media	exe	???	???	???	???	/system/bin/mediaserver
mediaserv	31306	media	0	???	???	???	???	/dev/null
mediaserv	31306	media	1	???	???	???	???	/dev/null
mediaserv	31306	media	2	???	???	???	???	/dev/null
mediaserv	31306	media	3	???	???	???	???	/dev/binder
mediaserv	31306	media	4	???	???	???	???	/dev/log/main
mediaserv	31306	media	5	???	???	???	???	/dev/log/radio
mediaserv	31306	media	6	???	???	???	???	/dev/log/events
<SNIP>								
mediaserv	31306	media	911	???	???	???	???	/priv-app/CP_Camera.apk
mediaserv	31306	media	912	???	???	???	???	/priv-app/CP_Camera.apk
mediaserv	31306	media	914	???	???	???	???	/priv-app/CP_Camera.apk
mediaserv	31306	media	915	???	???	???	???	/priv-app/CP_Camera.apk
mediaserv	31306	media	916	???	???	???	???	/priv-app/CP_Camera.apk
mediaserv	31306	media	mem	???	00:04	0	992352	/dev/ashmem/AudioFlinger::Client

FD leaking issue in APK

```
#!/bin/bash
# Android KGSL memory statistics gather thingy

if [ -t 0 ]; then stty -echo -icanon time 0 min 0; fi
count=0
keypress=''
while [ "x$keypress" = "x" ]; do
        let count+=1
        echo "*****************************"
        echo "** Test Count : $count"
        echo "** `adb shell uptime`"
        adb shell lsof
        echo "#################################"
done

if [ -t 0 ]; then stty sane; fi
echo "You pressed '$keypress' after $count loop iterations"
echo "Thanks for using this script."
```

FD leaking issue in APK

- 查看adb shell lsof 的输出结果，我们可以检查
- 1. 哪一个进程打开的文件描述符(FD)个数持续增加 (一般就是出问题的APK进程，从log里面也能确定)
- 2. 具体哪一类文件(None Name 节点名称)打开的数目过多，持续增加。

- 上面两个信息可以帮助APK开发人员查找和定位APK中存在FD 泄露的地方。

Camera

Android 5.0 Lollipop camera log丢失问题

- Android 5.0 Lollipop 中引入新的logger实现。新的User space logd代替原来的kernel based logger。
- 新的logger实现带来一系列好处，但是在打印log较多时会有log丢失的问题
- 该问题在camera debug时比较容易出现，特别是3A打印较多时。
- 如果需要打开较多camera log但是不希望有log丢失，可以暂时切换到原来的log实现，方法如下
  - 在device/qcom/<target>/AndroidBoard.mk 中添加
    - TARGET_USES_LOGD = false
  - 重新编译烧录Android image

MSM8994/MSM8992 I2C LED flash 驱动实现

- 类似于Unified senor driver, MSM8994/MSM8992 引入Unified flash driver. 文档80-NL239-32_H_Camera_Sensor_Driver_Development 7.2节有相关介绍，但是没有太多细节和例子 (特别是I2C LED flash)。
- 下面以 adp1660 为例说明如果添加在MSM8994/MSM8992上添加I2C LED flash驱动
- 基本步骤:
  - 修改camera dtsi文件（ msm8994-camera-sensor-mtp.dtsi ），添加I2C LED flash需要使用的GPIO定义
  - 在mm-camera2/media-controller/modules/sensors/flash_libs/<flash_name>目录下添加 <flash_name>.c <flash_name>.h 和Android.mk文件。
  - 在flash_name.h文件中根据vendor/qcom/proprietary/mm-camerasdk/sensor/includes/flash_lib.h 对static flash_lib_t flash_lib_ptr数据结构进行赋值。

MSM8994/MSM8992 I2C LED flash 驱动实现 (2)

- 以adp1660为例，
  - msm8994-camera-sensor-mtp.dtsi 修改如下

```
-&soc {
-       led_flash0: qcom,camera-flash {
+&cci {
+       led_flash0: qcom,led-flash@0 {
                cell-index = <0>;
                compatible = "qcom,camera-flash";
-               qcom,flash-type = <1>;
-               qcom,flash-source = <&pmi8994_flash0 &pmi8994_flash1>;
-               qcom,torch-source = <&pmi8994_torch0 &pmi8994_torch1>;
+               qcom,cci-master = <0>;
+               pinctrl-names = "cam_default", "cam_suspend";
+               pinctrl-0 = <&cam_flash_en_active &cam_flash_now_active>;
+               pinctrl-1 = <&cam_flash_en_suspend &cam_flash_now_suspend>;
+               gpios = <&msm_gpio 23 0>,
+                       <&msm_gpio 24 0>;
+               qcom,gpio-flash-en = <0>;
+               qcom,gpio-flash-now = <1>;
+               qcom,gpio-req-tbl-num = <0 1>;
+               qcom,gpio-req-tbl-flags = <0 0>;
+               qcom,gpio-req-tbl-label = "FLASH_EN",
+                       "FLASH_NOW";
+               qcom,max-current = <750>;
+               qcom,max-duration = <1600>;
        };
-};

-&cci {
        actuator0: qcom,actuator@0 {
                cell-index = <0>;
                reg = <0x0>;
```

MSM8994/MSM8992 I2C LED flash 驱动实现 (3)

- Userspace 驱动，可以以mm-camera2/media-controller/modules/sensors/flash_libs/pmic/ 作为模板，但是flash_name.h需要添加更多内容，因为和PMIC类型的LED flash相比，需要添加上下电信息和i2c寄存器设置

Android 5.0 CTS camera 测试常见问题

- Android 5.0 CTS引入了对Camera 2 API的测试用例，因为对Camera 2 API的底层支持的差异，MSM8916/8939 (HAL1)和MSM8994 (测试时会用HAL3)遇到的问题也会不同.
- testDigitalZoomPreviewCombinations
  - MSM8916/8939
    - 确保修改 https://www.codeaurora.org/cgit/quic/la/platform/frameworks/base/commit/?id=a0496d34ffc5153bf933cacbd4be71a4a1bd4041 已经在版本中
    - 确保1280x960这个4:3的尺寸在default_<preview/picture/liveshot/video>_sizes 数组中(文件mct_pipeline.c)。
    - 确保最大照片尺寸的长宽比是4:3或者是16:9
    - 如果还有问题，请提case
  - MSM8994
    - MSM8994测试该项是会遇到有时过有时不过，特别是添加log时更难通过的问题。
    - 请使用msm8994-perf_defconfig 内核配置文件测试该项。Perf boot image建议用在量产版本中.

Android 5.0 CTS camera 测试常见问题 (2)

- testDualCameraPreview在msm8916/msm8939上测试不过
  - 因为msm8916/msm8939只有一个ISP，同时开启两个Bayer sensor做预览会失败。
  - 可以使用下面的workaround

```
--- a/services/camera/libcameraservice/CameraService.cpp
+++ b/services/camera/libcameraservice/CameraService.cpp
@@ -342,6 +342,13 @@ status_t CameraService::validateConnect(int
cameraId,
return -EACCES;
}

+ for(int i = 0; i < mNumberOfCameras; i++){
+ if(mClient[i] != 0 && i != cameraId){
+ ALOGE("can not connect two cameras simultaneously ");
+ return - EUSERS;
+ }
+ }
+
ICameraServiceListener::Status currentStatus =
getStatus(cameraId);
if (currentStatus == ICameraServiceListener::STATUS_NOT_PRESENT) {
ALOGI("Camera is not plugged in,“
```

注: 该修改也可以规避因为前后摄公用电而导致testMultiCameraRelease测试不过的问题

Camera Power

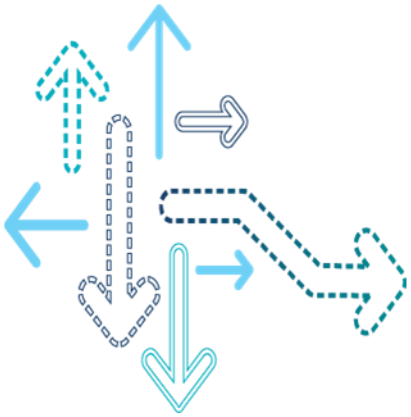
- 请参考 80-NP961-1_C_Camera_Power_Debug_Guide (如果在D&D上没找到，可以通过case申请)
- 几个关键优化点
  - 使用MDP composition，代替GPU composition
    - 在代码上表现为使用SurfaceView代替TextureView做Camera预览显示
    - 文档80-NP961-1 有相关介绍，具体代码修改可以参考下面这个代码提交：

    https://www.codeaurora.org/cgit/quic/la/platform/packages/apps/SnapdragonCamera/commit/?id=6ee7e415bf6f73d423df8da68f72fa6fdabde714
    - 我们的测试结果显示这个修改在msm8994可以降低大概100mA的电流 （预览界面下）
  - 使用perf_defconfig kernel配置
    - 在文件device/qcom/<target>/AndroidBoard.mk中修改 KERNEL_DEFCONFIG := <target>_defconfig为 KERNEL_DEFCONFIG := <target>-perf_defconfig
    - 如果在<target>_defconfig有自己的配置，需要合并到<target>-perf_defconfig
    - 请在Power测试时，务必使用perf_defconfig Kernel配置，测试显示该修改在msm8994上可以降低大概200mA的电流

    注：<target>根据平台不同而不同，比如8974, target为msm8994。8916/8939为msm8916
- 使用较小预览尺寸，比如在MSM8994上使用1080P代替2K的预览尺寸。
- 建议关闭全尺寸liveshot，而是使用预览尺寸liveshot

Camera Power Debug注意事项

- 如果使用高通MTP或者QRD样机做对比机。一定要确保两者的Camera参数完全一致：
  - ZSL或者non-ZSL
  - Preview size是否一致
  - Preview fps是否一致
  - Face detection 是否打开
  - Camera 应用是否一致 - 建议都用SnapdragonCamera测试。
- 使用相对干净的系统环境做测试，关闭不必要的后台服务和应用
- 如果不是使用SnapdragonCamera作为默认Camera应用，需要对比默认Camera应用和SnapdragonCamera的电流差异。对比后对应用进行优化
- 80-NP961-1文档中提到的一些方法是以牺牲用户体验作为代价的，客户需要自己权衡决定.
- 需要检查VFE/CPP clock
- 需要支持请提交case 到MM power
  - Problem Area 1: Multimedia
  - Problem Area 2: Power
  - Problem Area 3: Camera



Questions?

https://support.cdmatech.com