
高通用户体验性能优化期刊

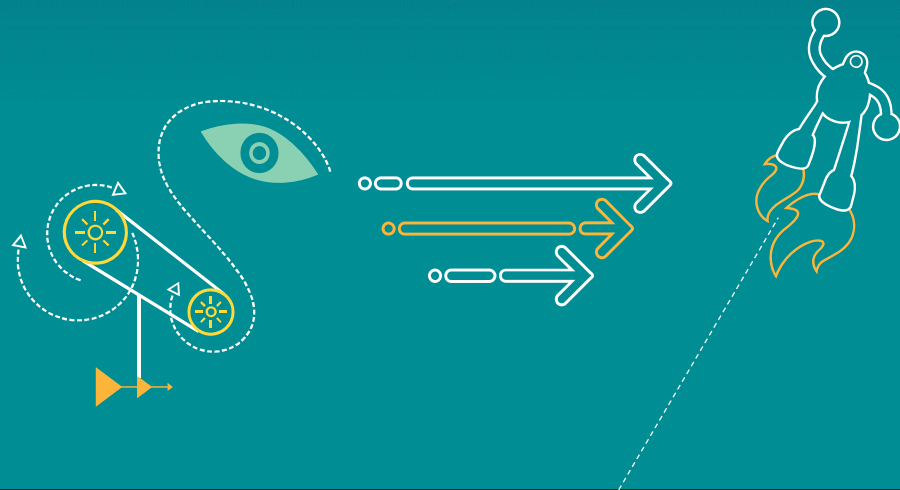
201512



Qualcomm Technologies, Inc.

Confidential and Proprietary – Qualcomm Technologies, Inc.

机密和专有信息——高通技术股份有限公司



Confidential and Proprietary – Qualcomm Technologies, Inc.

Confidential and Proprietary – Qualcomm Technologies, Inc.

NO PUBLIC DISCLOSURE PERMITTED: Please report postings of this document on public servers or web sites to: DocCtrlAgent@qualcomm.com. **禁止公开：**如在公共服务器或网站上发现本文档，请报告至：DocCtrlAgent@qualcomm.com.

Restricted Distribution: Not to be distributed to anyone who is not an employee of either Qualcomm or its affiliated without the express approval of Qualcomm's Configuration Management. **限制分发：**未经高通配置管理部门的明示批准，不得发布给任何非高通或高通附属及关联公司员工的人。 Not to be used, copied, reproduced, or modified in whole or in part, nor its contents revealed in any manner to others without the express written permission of Qualcomm Technologies, Inc. 未经高通技术股份有限公司明示的书面允许，不得使用、复印、复制、或修改全部或部分文档，不得以任何形式向他人透露其内容。

The user of this documentation acknowledges and agrees that any Chinese text and/or translation herein shall be for reference purposes only and that in the event of any conflict between the English text and/or version and the Chinese text and/or version, the English text and/or version shall be controlling. 本文档的用户知悉并同意中文文本和/或翻译仅供参考之目的，如英文文本和/或版本和中文文本和/或版本之间存在冲突，以英文文本和/或版本为准。

This document contains confidential and proprietary information and must be shredded when discarded. 未经高通明示的书面允许，不得使用、复印、复制全部或部分文档，不得以任何形式向他人透露其内容。本文档含有高通机密和专有信息，丢弃时必须粉碎销毁。

Qualcomm reserves the right to make changes to the product(s) or information contained herein without notice. No liability is assumed for any damages arising directly or indirectly by their use or application. The information provided in this document is provided on an "as is" basis. 高通保留未经通知即修改本档中提及的产品或信息的权利。本公司对使用或应用本文档所产生的直接或间接损失概不负责。本文档中的信息为基于现状所提供，使用风险由用户自行承担。

Qualcomm is a trademark of QUALCOMM Incorporated, registered in the United States and other countries. All QUALCOMM Incorporated trademarks are used with permission. Other product and brand names may be trademarks or registered trademarks of their respective owners. Qualcomm是高通公司在美国及其它国家注册的商标。所有高通公司的商标皆获得使用许可。其它产品和品牌名称可能为其各自所有者的商标或注册商标。

This technical data may be subject to U.S. and international export, re-export, or transfer ("export") laws. Diversion contrary to U.S. and international law is strictly prohibited. 本文档及所含技术资料可能受美国和国际出口、再出口或转移出口法律的 限制。严禁违反或偏离美国和国际的相关法律。

Qualcomm Technologies, Inc. 5775 Morehouse Drive San Diego, CA 92121 U.S.A.

高通技术股份有限公司，美国加利福尼亚州圣地亚哥市莫豪斯路 5775 号，邮编 92121

Revision History

Revision	Date	Description
A	Dec 2015	Initial release

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

内容

- UI optimization (SaveLayer)
- Perflock v3
- ALMK feature customization
- Important Performance document



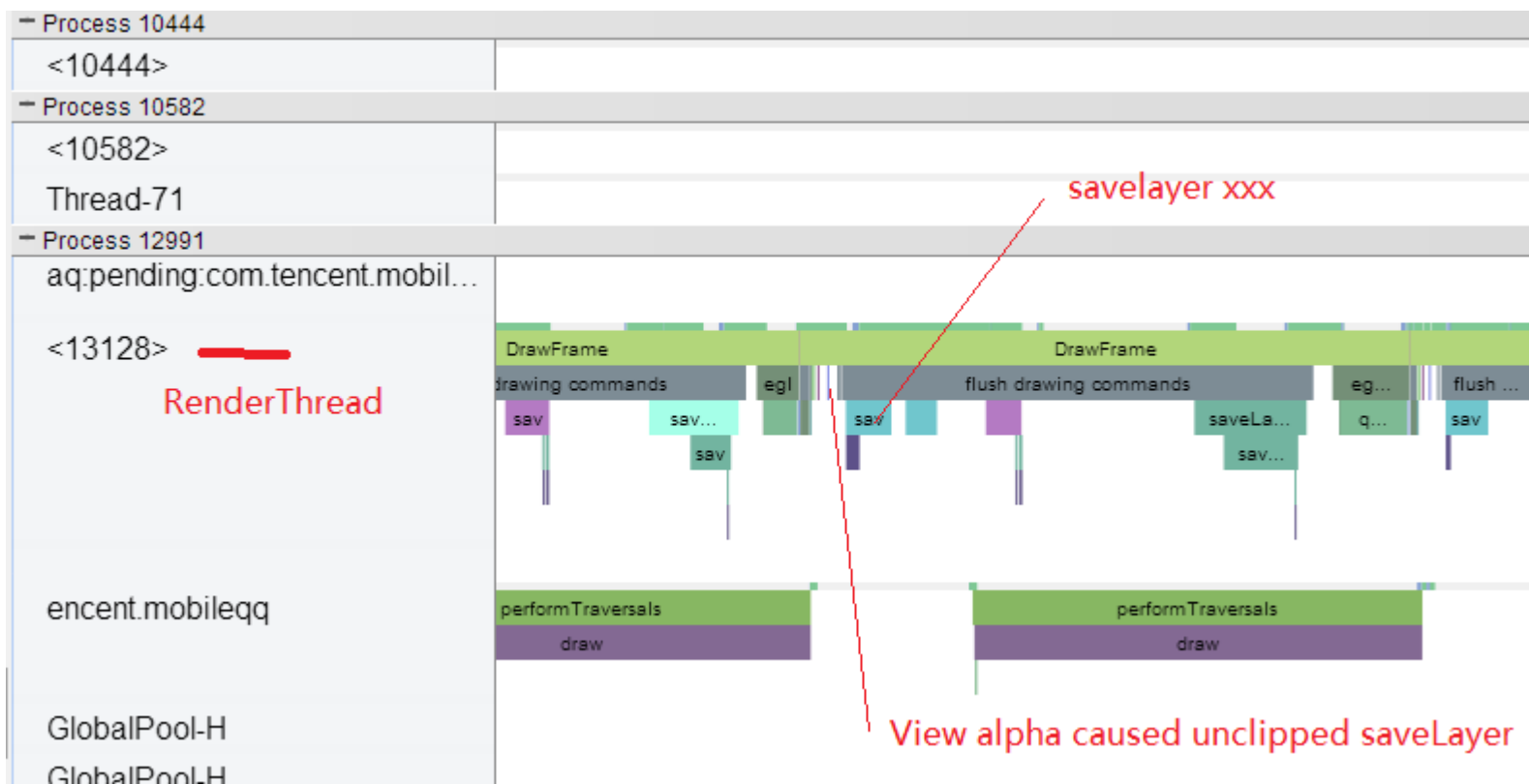
UI Optimization

检查saveLayer

请检查systrace 中RenderThread

- 1) 是否存在多次saveLayer,而且每次saveLayer时间都较长
- 2) 是否存在View alpha caused unclipped saveLayer trace信息

如果上述两个条件都满足，而且draw时间比较长，需要优化UI



如何找到调用SaveLayer的代码

两种方法

- 1) 在android.view.view中setAlpha函数设置断点，观察绘制过程中调用栈
- 2) 直接在setAlpha中打印调用栈:

```
public void setAlpha(float alpha) {  
  
    ...  
  
    new Exception().printStackTrace();  
    System.out.println("className " + getClass().getName());  
  
}
```

Output:

```
android.view.View.setAlpha(View.java:10563)  
com.tencent.mobileqq.activity.recent.DrawerFrame.a(ProGuard:866)  
com.tencent.mobileqq.activity.recent.DrawerFrame.handleMessage(ProGuard:959)  
com.tencent.util.WeakReferenceHandler.handleMessage(ProGuard:26)  
android.os.Handler.dispatchMessage(Handler.java:102)  
android.os.Looper.loop(Looper.java:135)  
android.app.ActivityThread.main(ActivityThread.java:5254)
```

找到导致savelayer调用的view

通过命令dump UI 层级: pkg 是应用程序的包名

adb shell dumpsys activity -c package *pkg*

UI 层级:

com.android.internal.policy.impl.PhoneWindow\$DecorView{3b2cbc1f V.ED.... ... 0,0-1080,1920}

android.widget.LinearLayout{3257bb6c V.E..... ... 0,0-1080,1920}

android.view.ViewStub{bfde535 G.E..... ... 0,0-0,0 #1020377}

android.widget.FrameLayout{394b2b40 V.E..... ... 0,0-1080,1920 #1020002 android:id/content}

降低/消除saveLayer的调用

三种方法

1.删除 alpha动画: 删减调用设置alpha属性的代码

2.如果view内容相互不重叠, 覆写View类的hasOverlappingRendering方法

```
public boolean hasOverlappingRendering (){  
    return false;  
}
```

3. 如果上述两种方法都不可行, 那么可以在动画开始的时候, 把view的layer类型设置为 LAYER_TYPE_HARDWARE ,或LAYER_TYPE_SOFTWARE

```
interestView.setLayerType(View. LAYER_TYPE_HARDWARE, null);
```

saveLayer分析

Analysis

当view的属性改变时，DrawFrame函数会把最新的alpha值传递到hwui中

CreateLayer函数执行费时，尽量避免频繁调用

glFramebufferTexture2D	OpenGLRender.cpp	830
OpenGLRenderer::createLayer	OpenGLRenderer.cpp	779
OpenGLRenderer::saveLayer(kClipToLayer_SaveFlag)	OpenGLRenderer.cpp	609
SaveLayerOp::applyState	DisplayListOp.h	366
RenderNode::setViewProperties		905
RenderNode::issueOperations	RenderNode.cpp	636
DrawRenderNodeOp::Replay	DisplayListOp.h	1489

Reference

[**http://developer.android.com/reference/android/view/View.html#setAlpha\(float\)**](http://developer.android.com/reference/android/view/View.html#setAlpha(float))



Perflock V3 on Android M

Perflock V3

- **介绍**

- Perflock V3在Android M系统上开始使用，在原来一套Perflock（V2）的基础上做了一系列改进，演变成Perflock V3。

- **V3 vs. V2**

- V3的包名改成android.util.boostFramework
- V3对于部分常见的Perflock参数做了向下兼容

Perflock V3

▪ **使用方法**

- Java和Native的使用方法和Perflock v2的差不多，除了import包名和参数个数发生了变化。

例子：

设置CPU0的最低频率到1.3GHz，并维持2s

- Perflock v2的用法

```
Import org.codeaurora.performance ;
```

```
mPerf = new Performance();
```

```
Int duration = 2000;
```

```
mPerf.perfLockAcquire(duration, 0x020D);
```

Perflock V3

- Perflock v3的用法

Import `android.util.boostFramework;`

Performance mPerf = new Performance();

int duration = 2000;

int list[];

list[0] = MPCTLV3_MIN_FREQ_CLUSTER_BIG_CORE_0; //0x40800000 , 这里我们假定CPU0是在Big cluster上

list[1] = 0x5DC; // 0x5DC = 1300

mPerf.perfLockAcquire(duration, list);

由上面可以看到，Perflock v3传递的参数是Resource ID和Value分开传入的

- Native层的用法的变化也类似。
- PerfLock v3支持的常用参数请查看本文最后的参考文档中的Table2-1 《Commonly used opcodes》

Perflock V3

▪ 向下兼容

- 尽管PerfLock v3兼容v2所有的ResourceID，但是对于Value部分只支持比较常用的几大类别，分别是：

CPUFREQ_MAJOR_OPCODE // CPU频率相关设置，例如min_freq, max_freq

ONDEMAND_MAJOR_OPCODE // Ondemand governor 设置，目前已经比较少用

CORE_HOTPLUG_MAJOR_OPCODE // Hotplug相关设置，例如min_online, max_online

INTERACTIVE_MAJOR_OPCODE // Interactive governor设置

SCHED_MAJOR_OPCODE // scheduler设置，目前只支持scheduler_mostly_idle

- ResourceID的兼容可以查看Request.cpp当中的数组定义old_opcde_to_new_opcde
- Value的兼容可以查看函数Request::TranslateValueToNew

Perflock V3

参考文档

80-NR256-2 B MPCTL Feature.pdf



ALMK feature customization

ALMK feature customization

▪ 介绍

ALMK feature 是我们对 LMK feature 的优化,可以提升用户的体验和系统性能.我们已经enable 了该feature.目前我们需要OEM 作小小的改动.因为OEM的产品名称可能不是我们默认的产品名称.当然我们正在对这个feature 做优化,优化后是不需要做这样的修改.

这里以8952 chipset 举例.如果是其它平台请做类是的修改.

修改文件 source/device/qcom/common/rootdir/etc/init.qcom.post_boot.sh

```
#Enable adaptive LMK and set vmpressure_file_min
```

```
ProductName=`getprop ro.product.name`
```

```
if [ "$ProductName" == "msm8952_32" ] || [ "$ProductName" == "msm8952_32_LMT" ];
```

```
then echo 1 > /sys/module/lowmemorykiller/parameters/enable_adaptive_lmk
```

```
echo 53059 > /sys/module/lowmemorykiller/parameters/vmpressure_file_min
```

```
elif [ "$ProductName" == "msm8952_64" ] || [ "$ProductName" == "msm8952_64_LMT" ];
```

```
then echo 1 > /sys/module/lowmemorykiller/parameters/enable_adaptive_lmk
```

```
echo 81250 > /sys/module/lowmemorykiller/parameters/vmpressure_file_min fi
```

由于你的产品的名称不是"msmXXX" 所以需要根据你的产品名称做相应的修改.例如,如果你的产品名称是"**cool_phone**" 则修改如下:

ALMK feature customization

- 需要改动patches

```
#Enable adaptive LMK and set vmpressure_file_min
```

```
ProductName=`getprop ro.product.name`
```

```
if [ "$ProductName" == "msm8952_32" ] || [ "$ProductName" == "msm8952_32_LMT" ] || [ "$ProductName" == "cool_phone" ]; //32bit build
```

```
then echo 1 > /sys/module/lowmemorykiller/parameters/enable_adaptive_lmk
```

```
echo 53059 > /sys/module/lowmemorykiller/parameters/vmpressure_file_min
```

```
elif [ "$ProductName" == "msm8952_64" ] || [ "$ProductName" == "msm8952_64_LMT" ] || [ "$ProductName" == "cool_phone" ]; //64bit build
```

```
then echo 1 > /sys/module/lowmemorykiller/parameters/enable_adaptive_lmk
```

```
echo 81250 > /sys/module/lowmemorykiller/parameters/vmpressure_file_min fi
```



Important performance document

重要文档

Document No	Description
80-NT978-1	Performance Improvement Patches Android Lollipop Builds
80-P3936-1	Perf Improvement Patches Android Marshmallow Builds
80-NR256-2	MPCTL Feature (适用于PerfLock V3)
80-P0584-1	Common Performance Issues Debugging Guide
80-NV303-1	MSM8916/MSM8909 Memory Optimization Guidelines
80-NR256-6	Debugging Common Memory Performance Issues
80-NT384-1	PerfLock API Overview

Questions?

<https://support.cdmatech.com>

