Linaro connect
Vancouver 2018

drm_hwcomposer

John Stultz
<john.stultz@linaro.org>

What is a Hardware Composer?

Layers of buffers

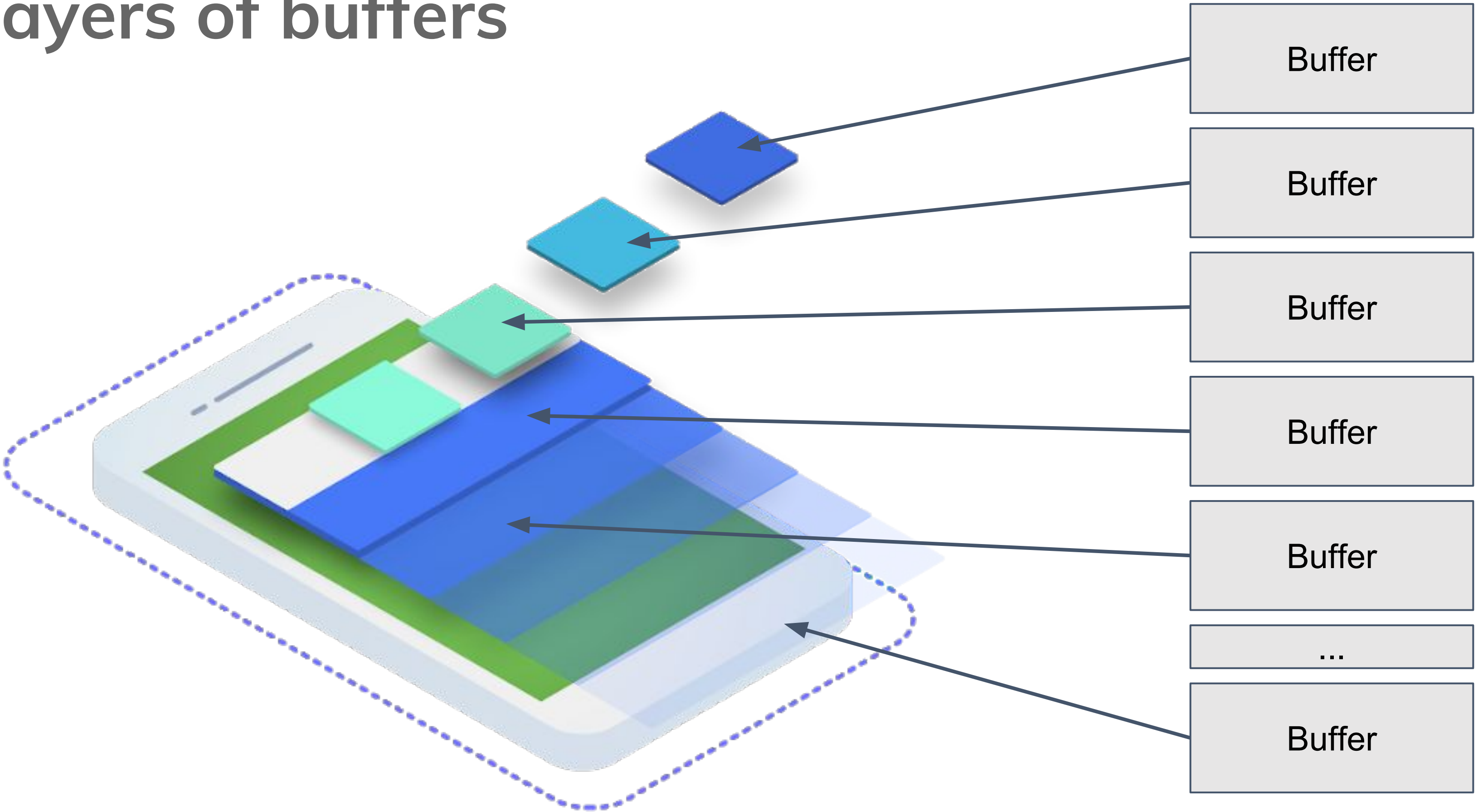
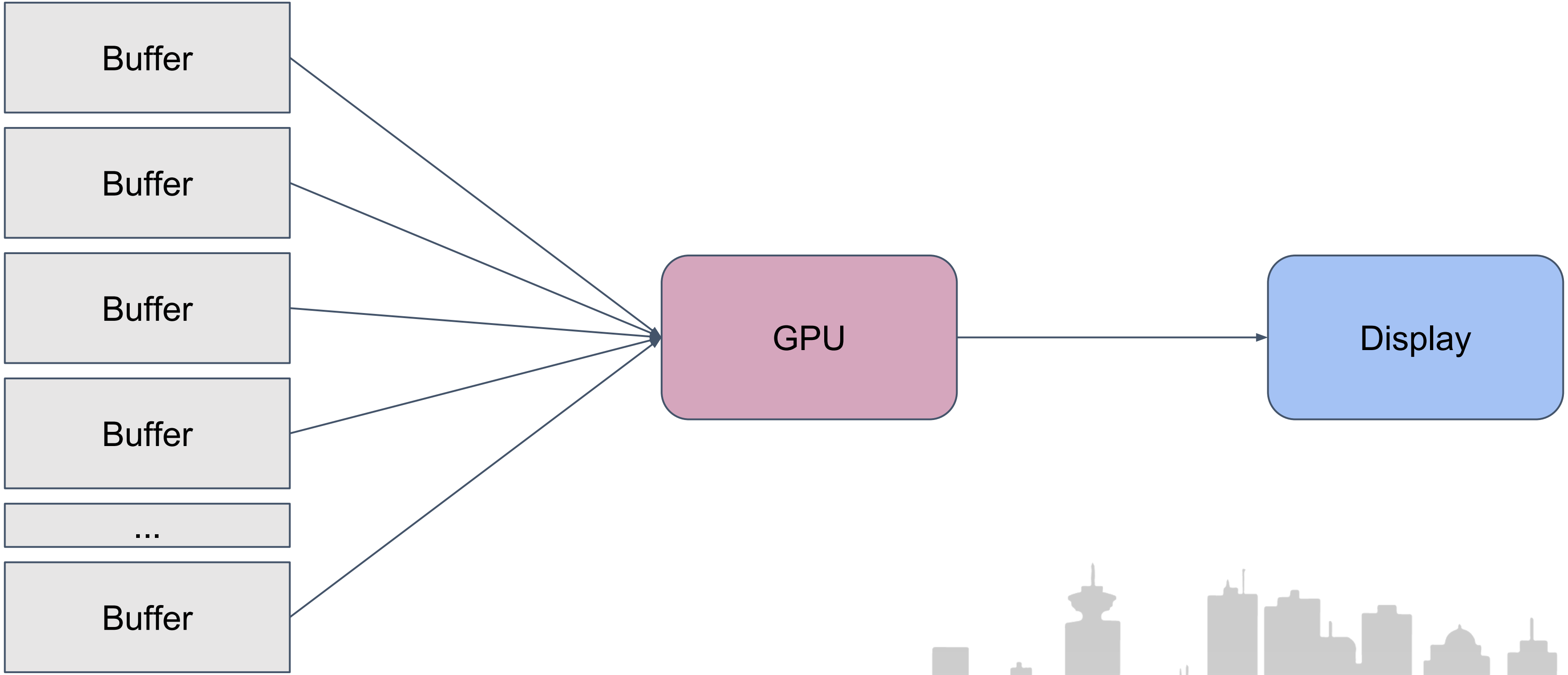
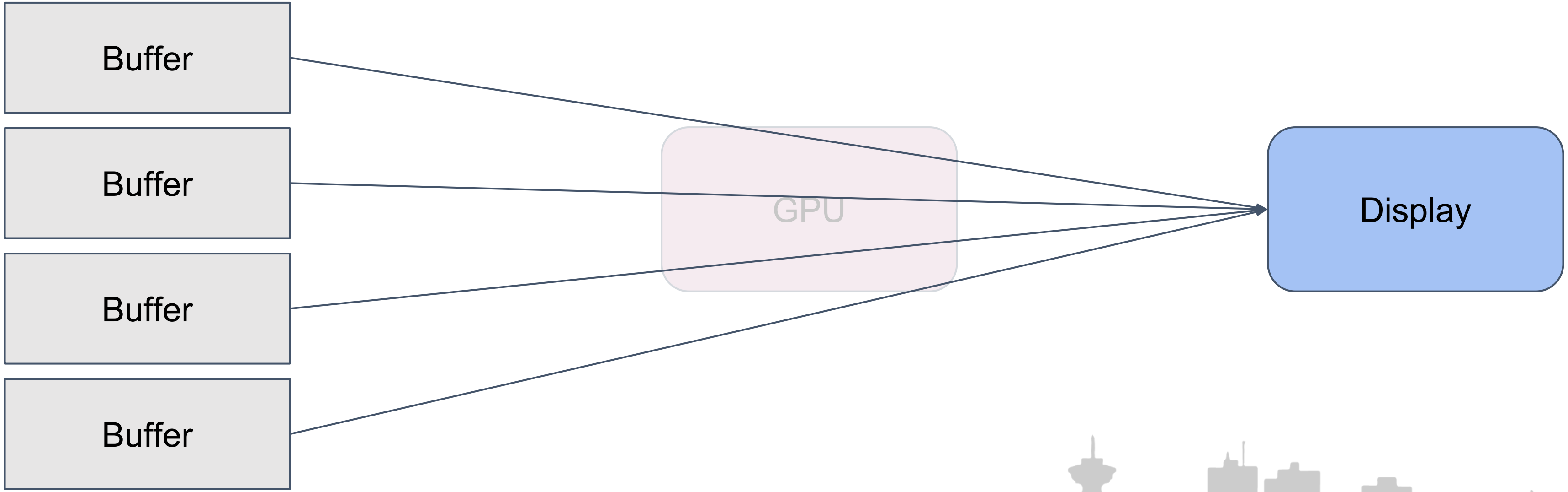


Image from https://source.android.com/ under the Creative Commons Attribution 3.0 license

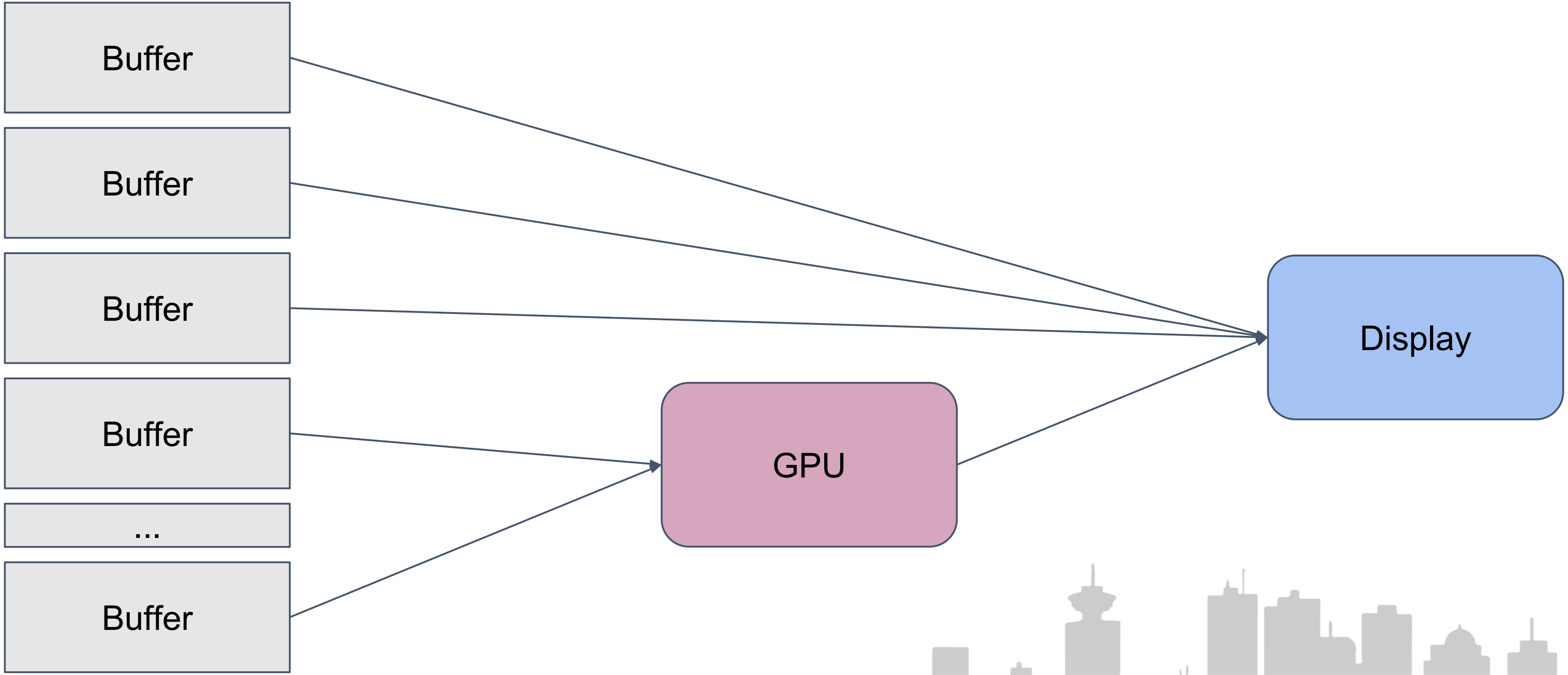
Displaying buffers



Displaying buffers



Displaying buffers



History/Background

`framebuffer_device_t`

Originally part of the gralloc code

Structure to access the fbdev

Provided a `post()` hook to post a frame-buffer to the display

HWC1 (ICS)

Started as HAL for accelerated composition, utilizing overlays and 2d-blitters

Simple interface: prepare(), set(), vsync()

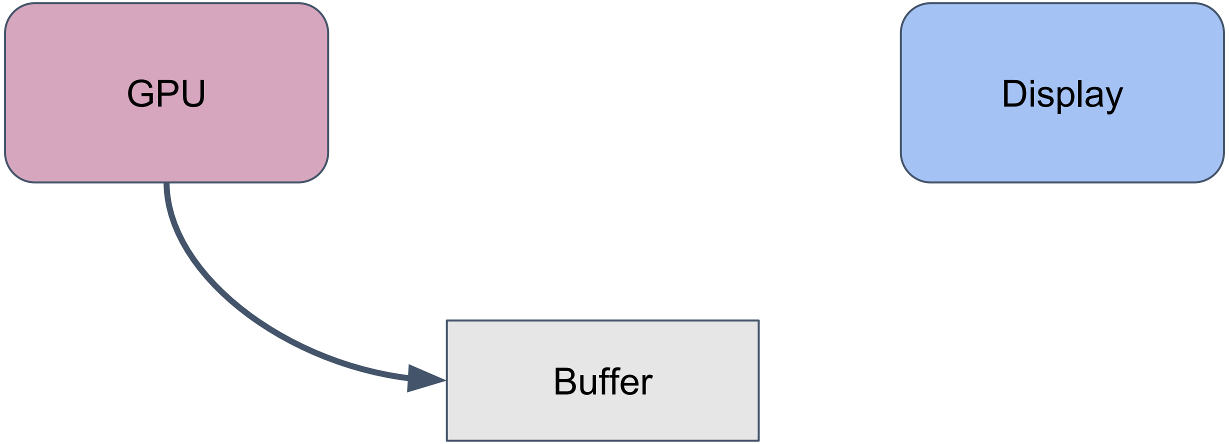
Android Sync (Jellybean)

Tool for explicit ownership and management of buffers as they move through the graphics pipeline

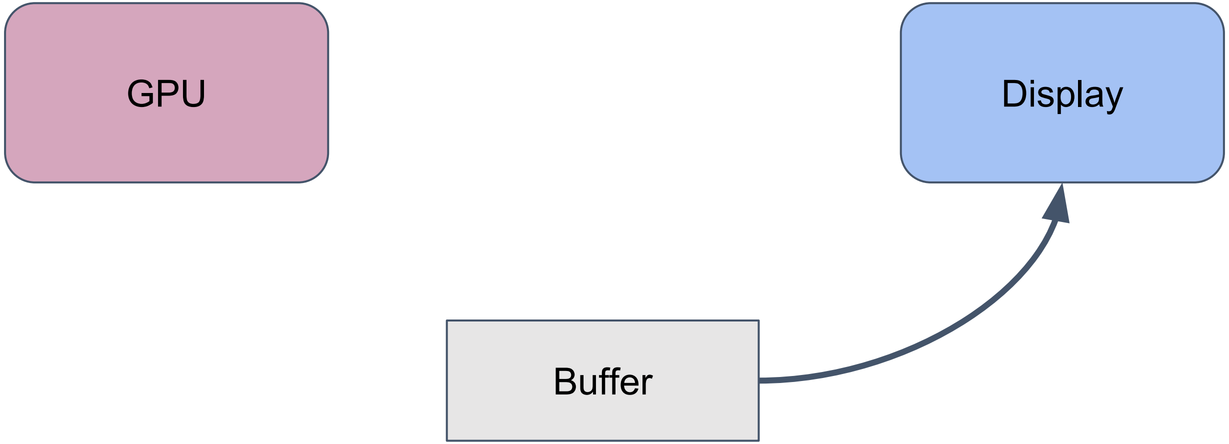
Conceptually similar to condition-variables

Allows different parts of the graphics pipeline (gpu, hwcomposer, display controller) to track when a buffer is ready to be displayed or has finished being displayed.

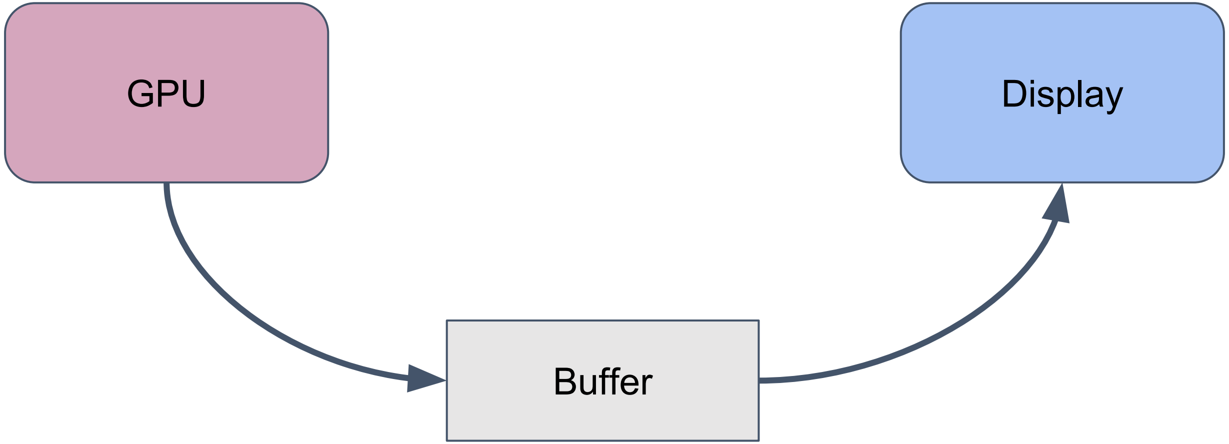
Single Buffering



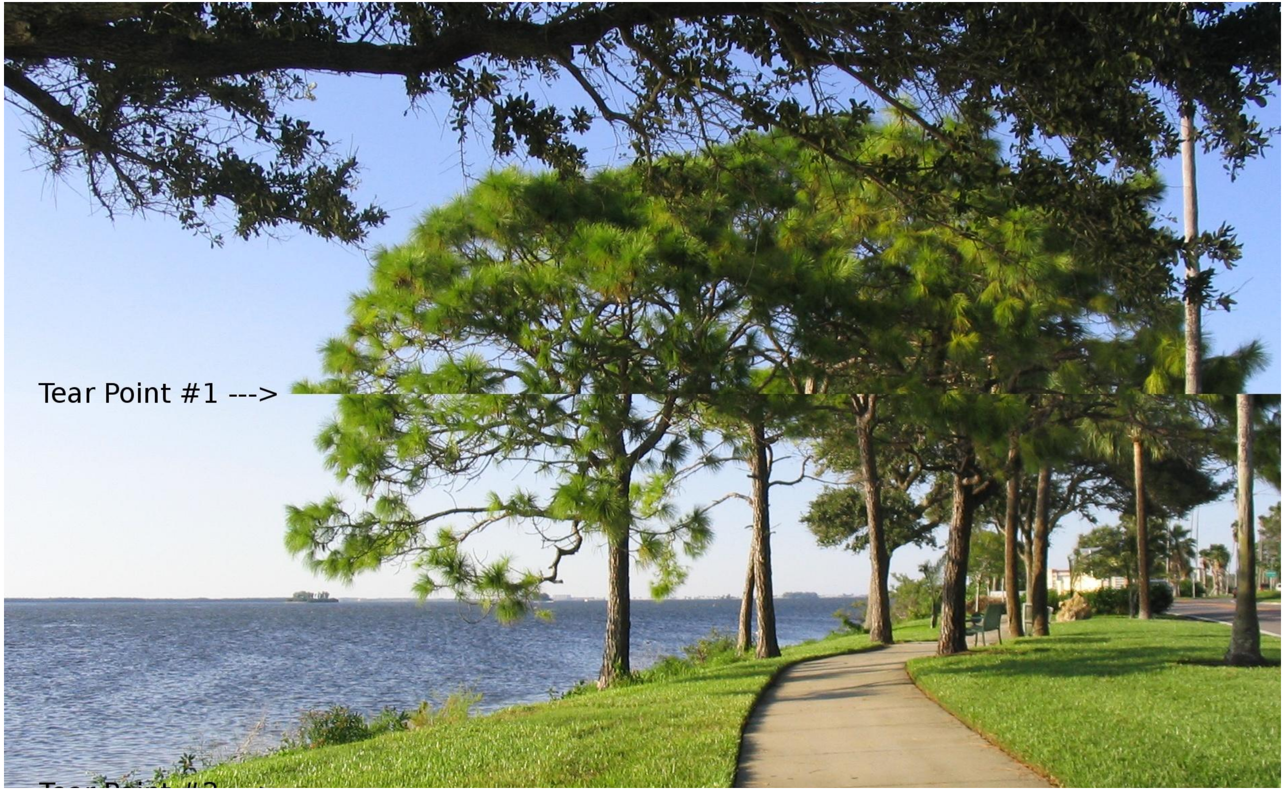
Single Buffering



Tearing

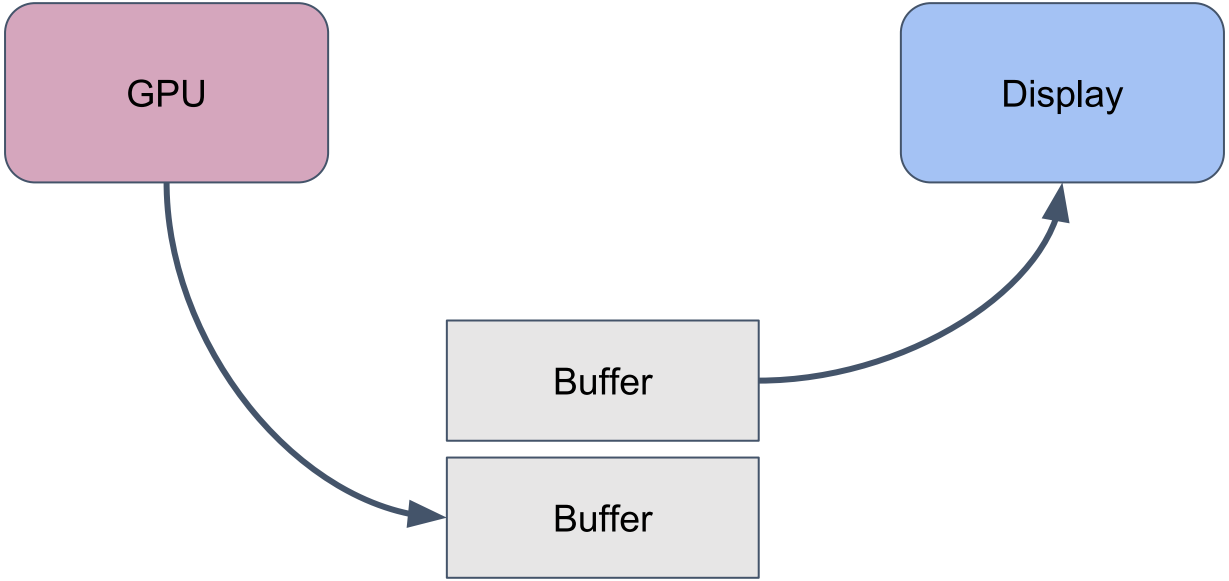


Tearing

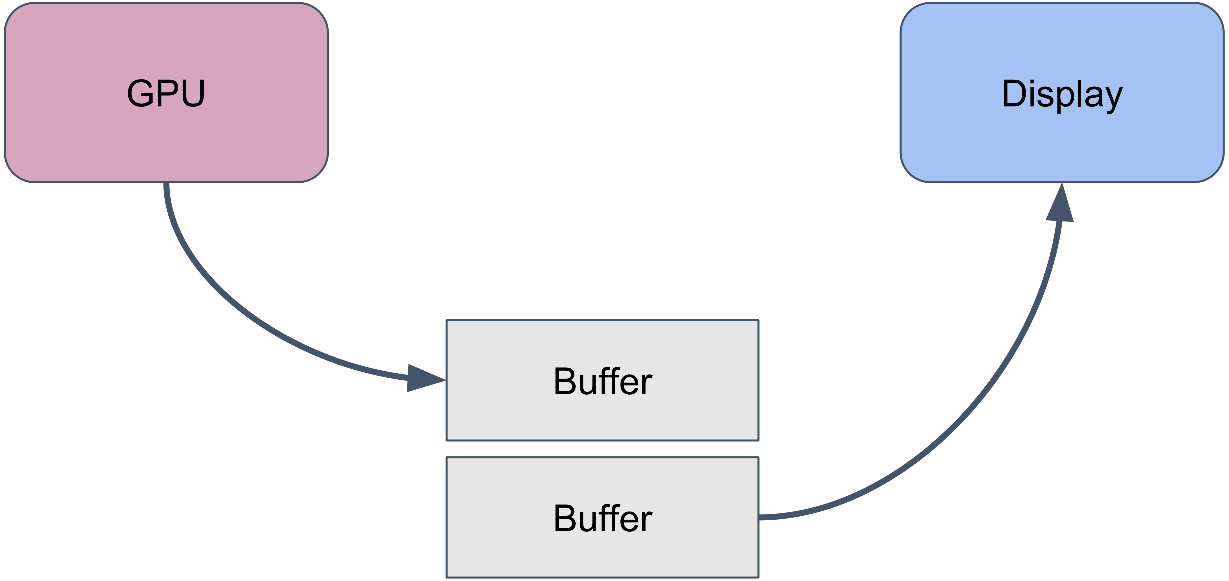


By Vanessaezekowitz - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=7437859

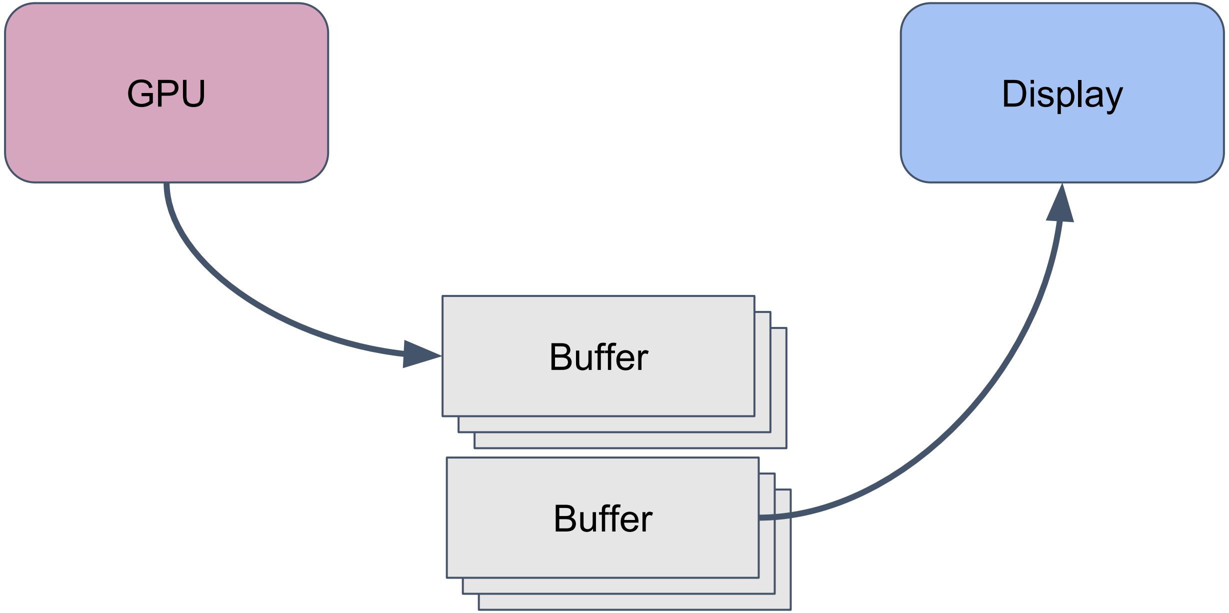
Double Buffering



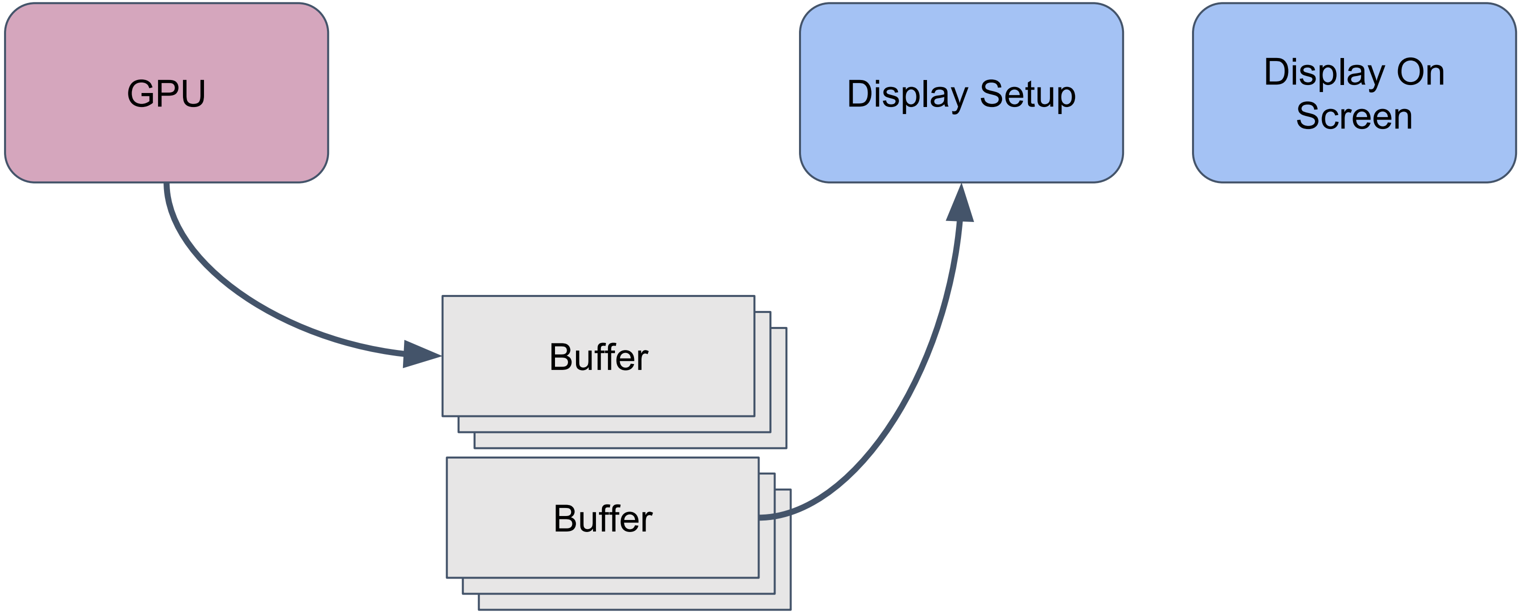
Double Buffering



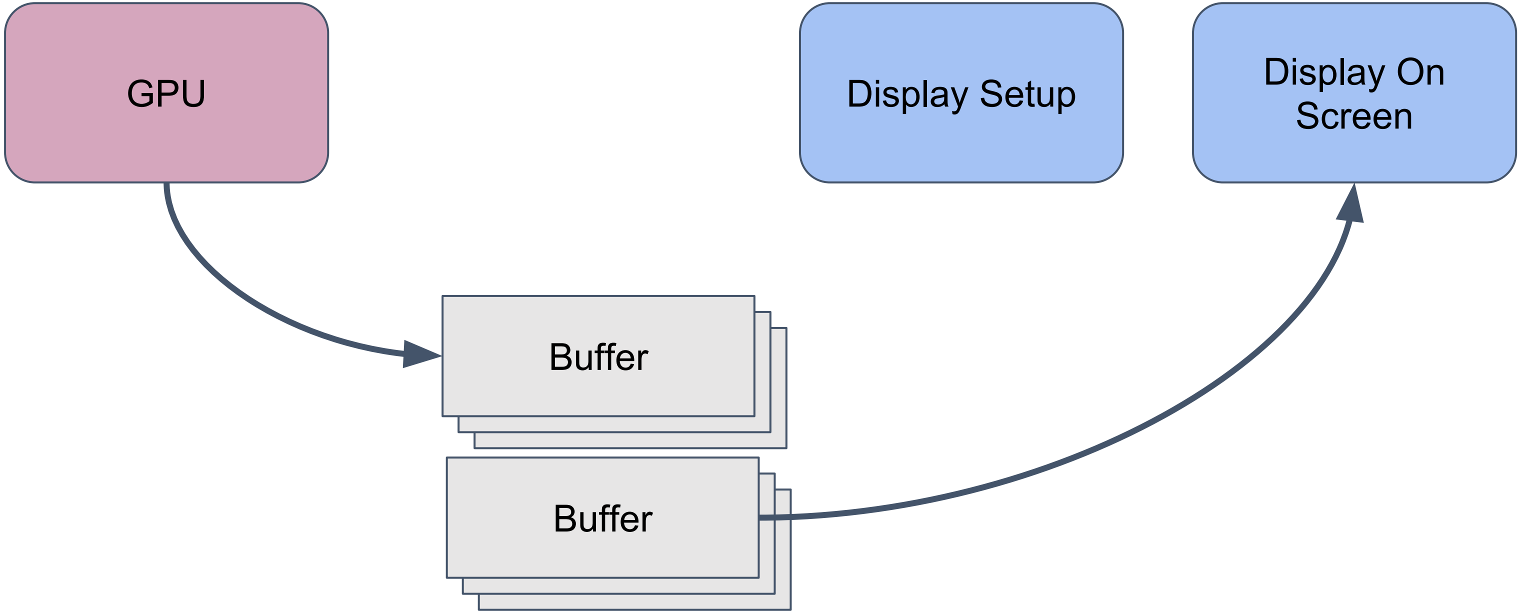
It's complicated



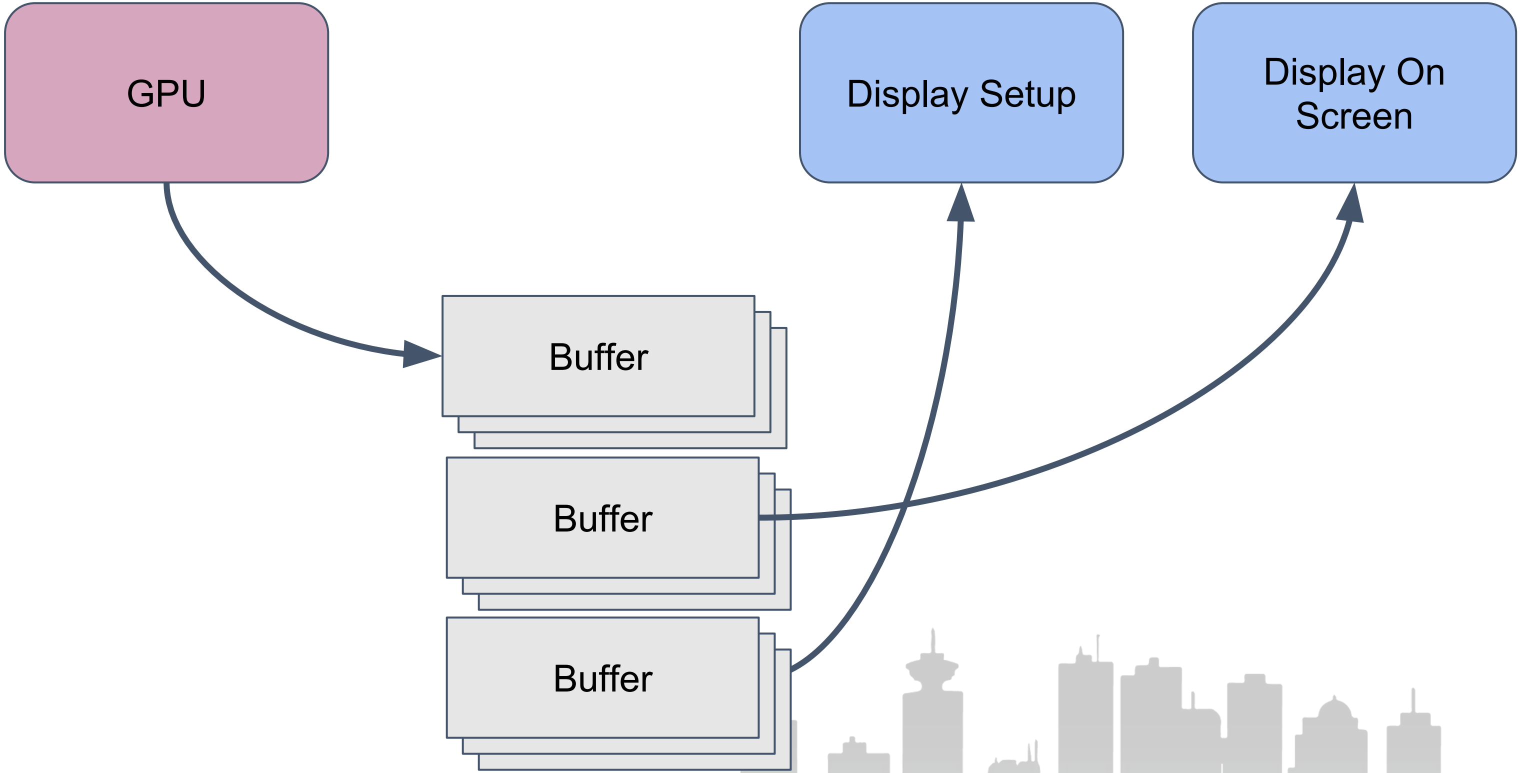
It's even more complicated



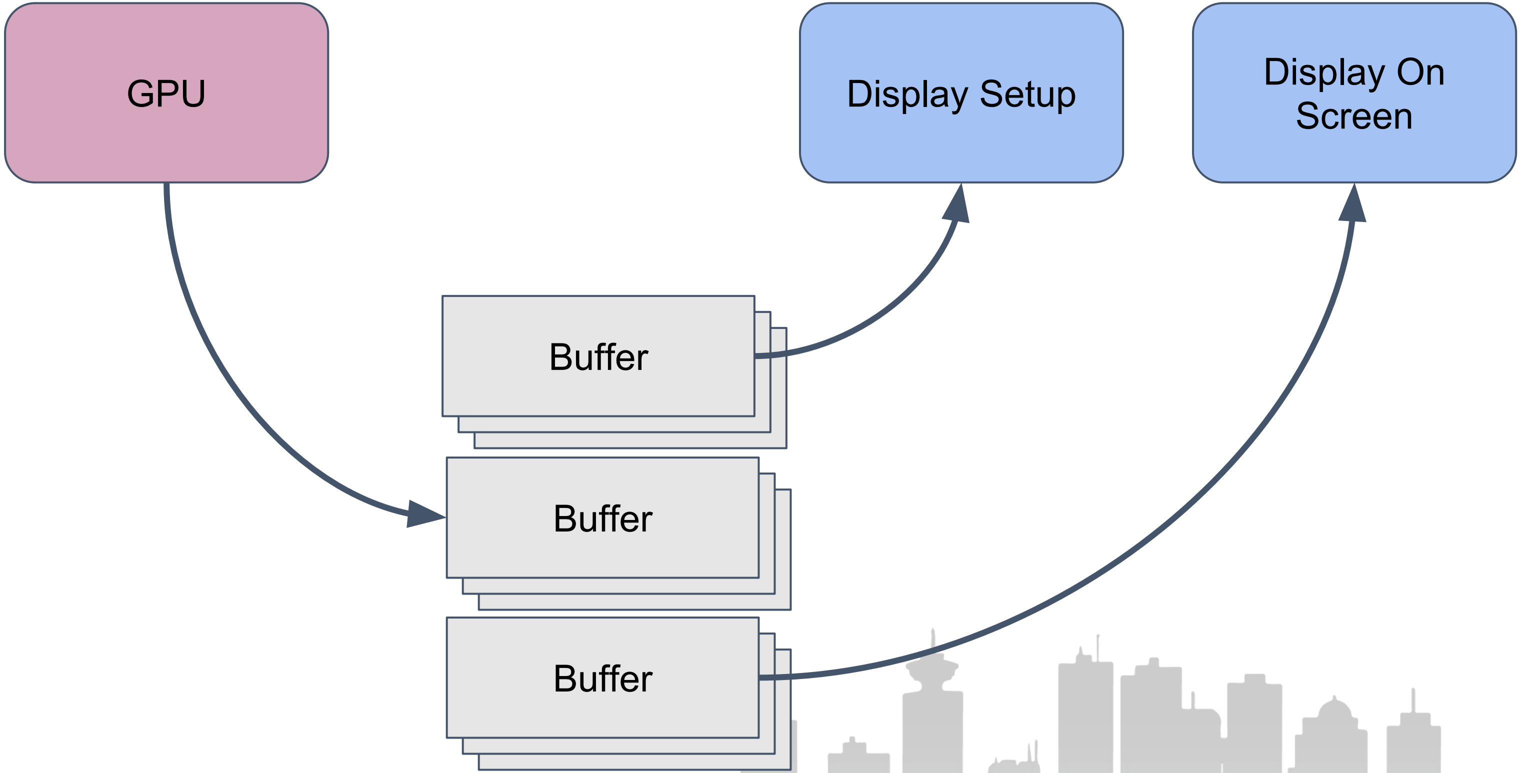
It's even more complicated



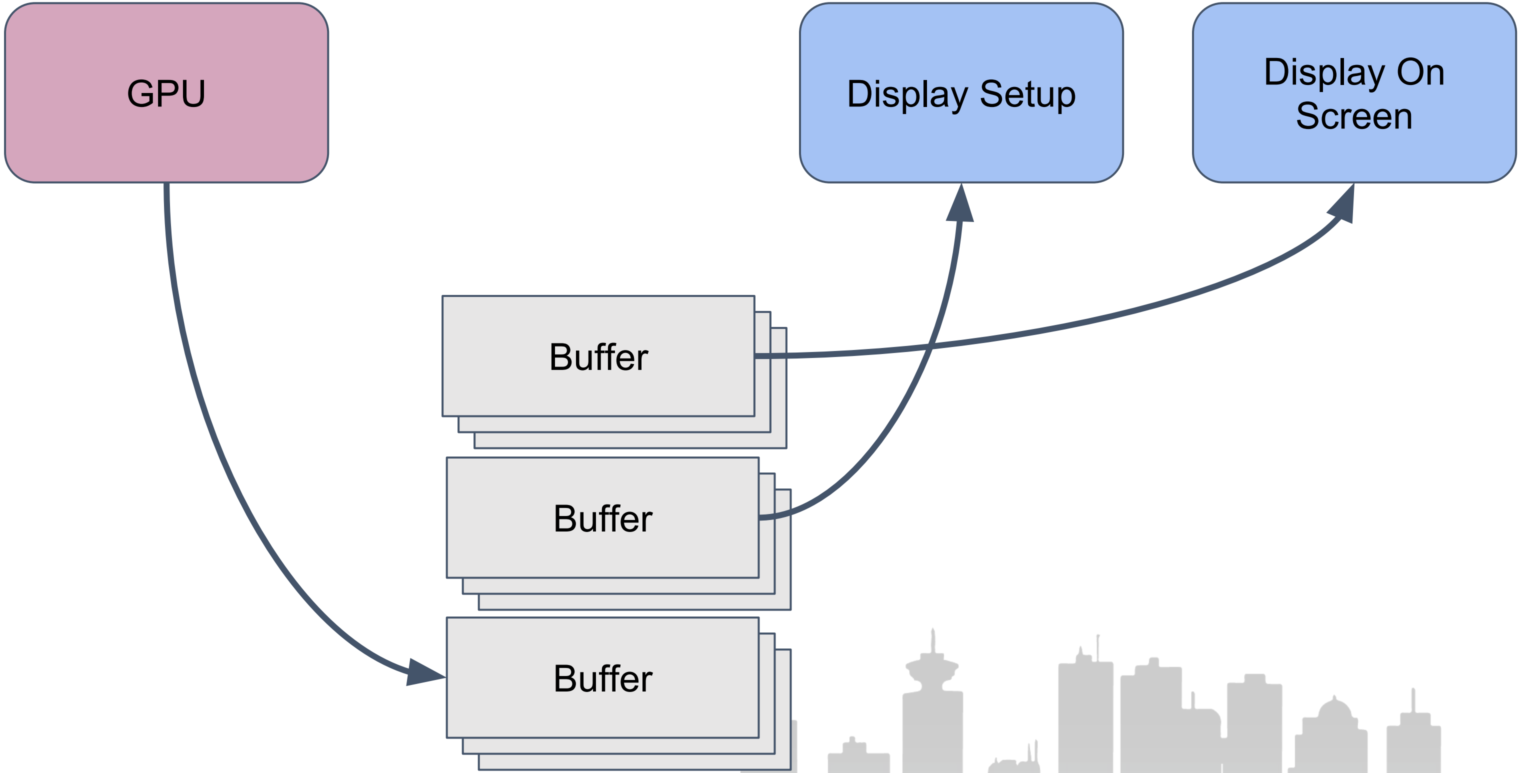
Triple Buffering



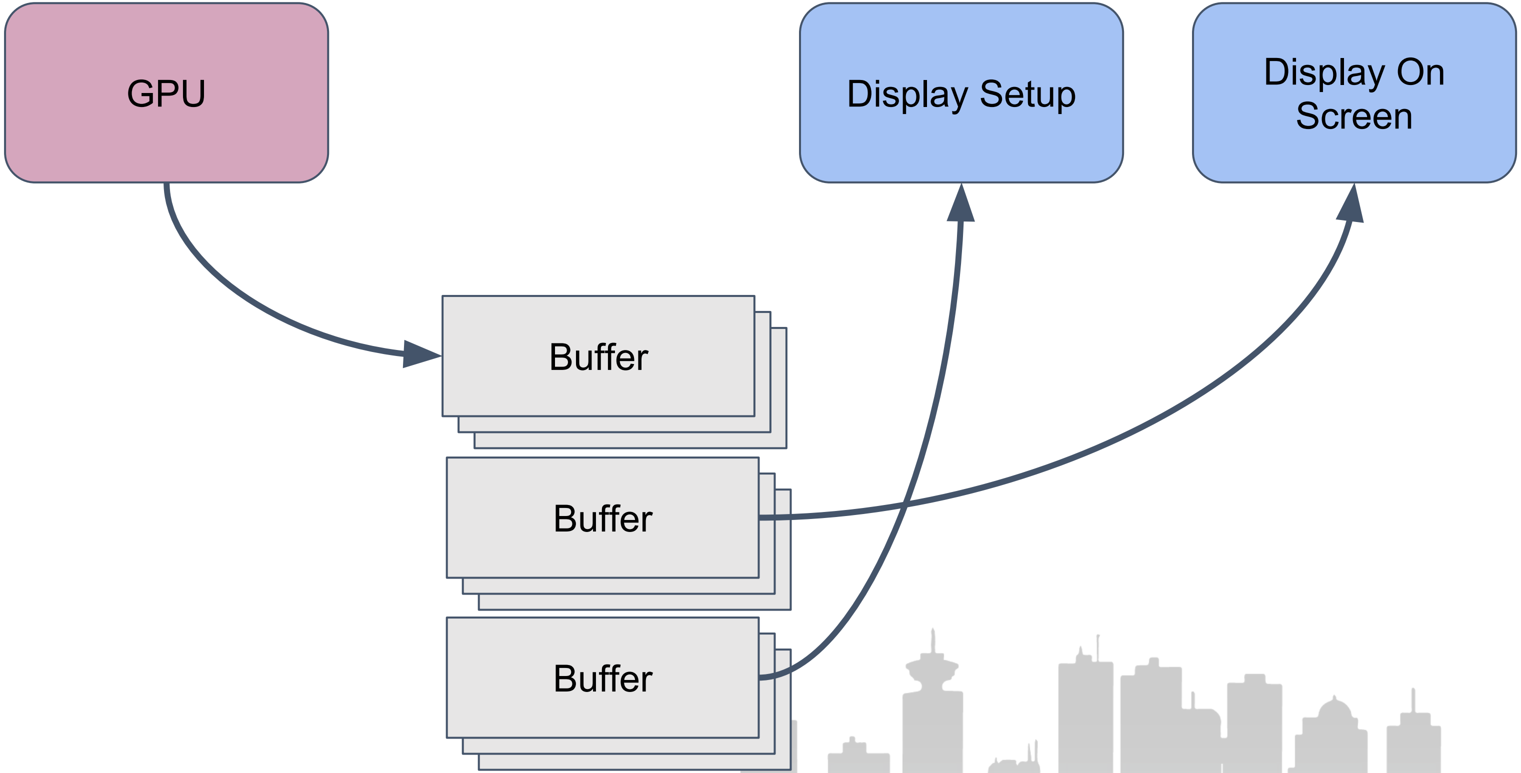
Triple Buffering



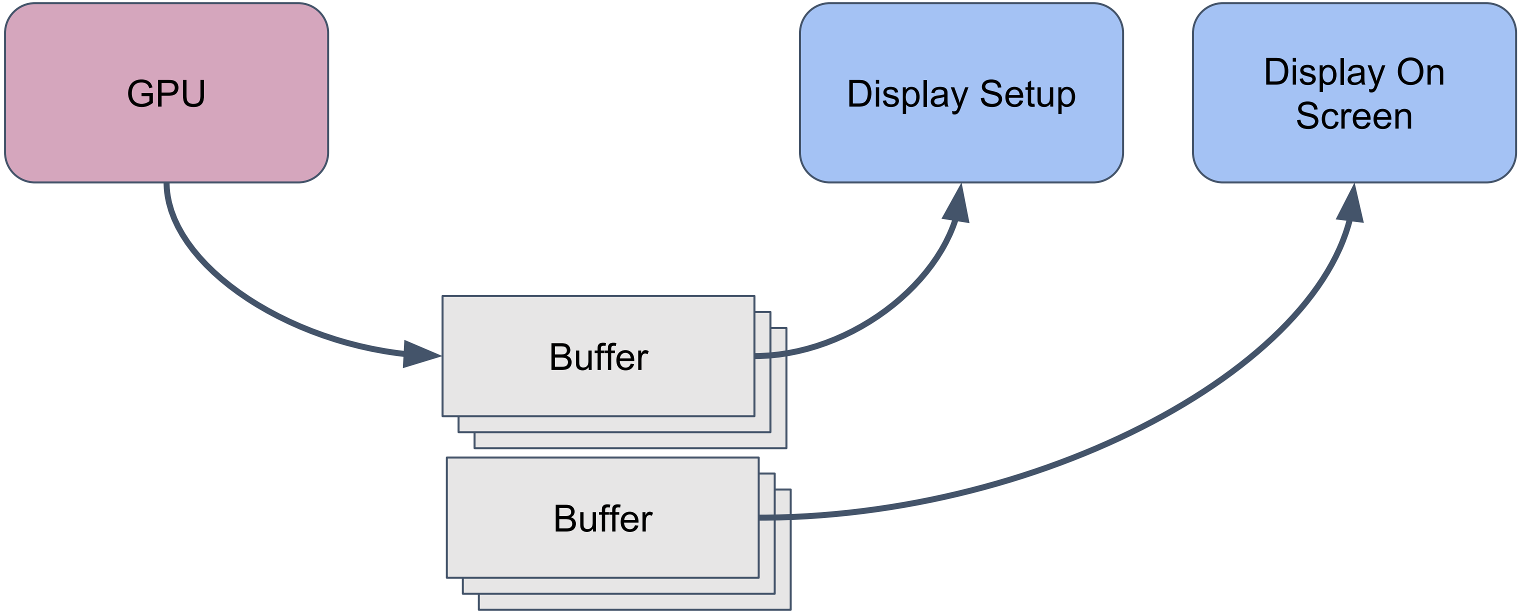
Triple Buffering



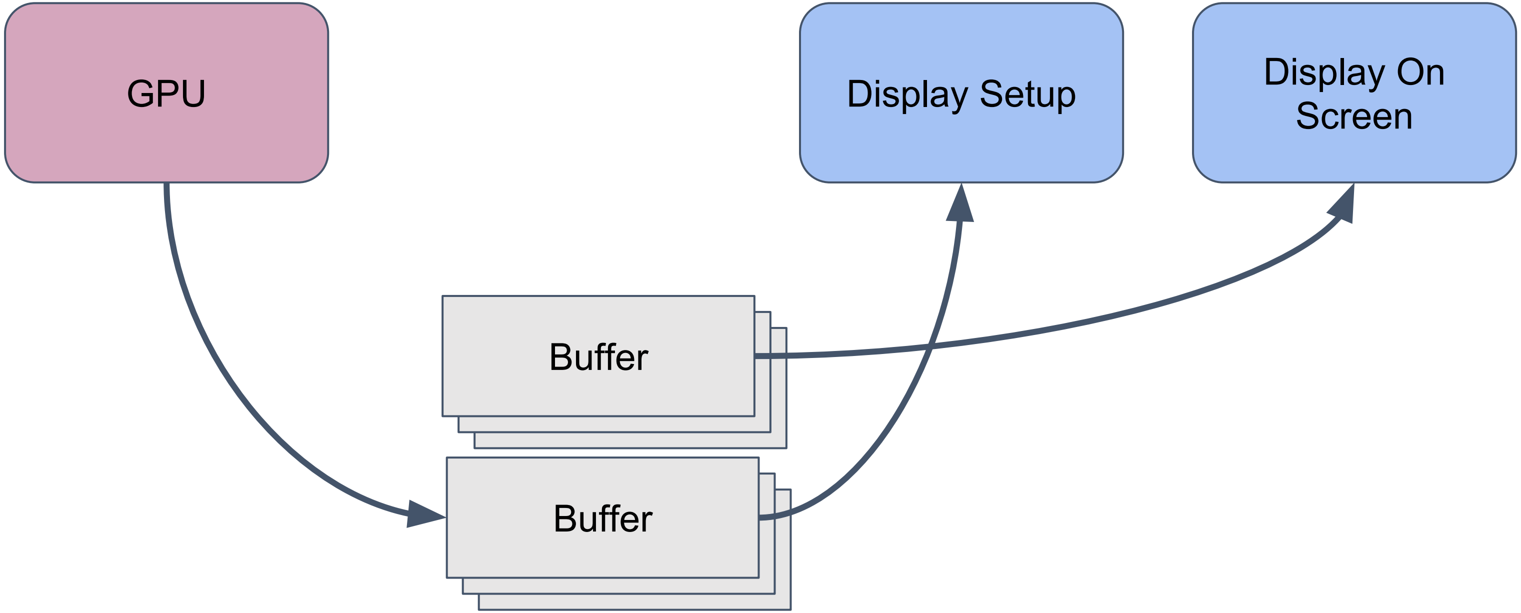
Triple Buffering



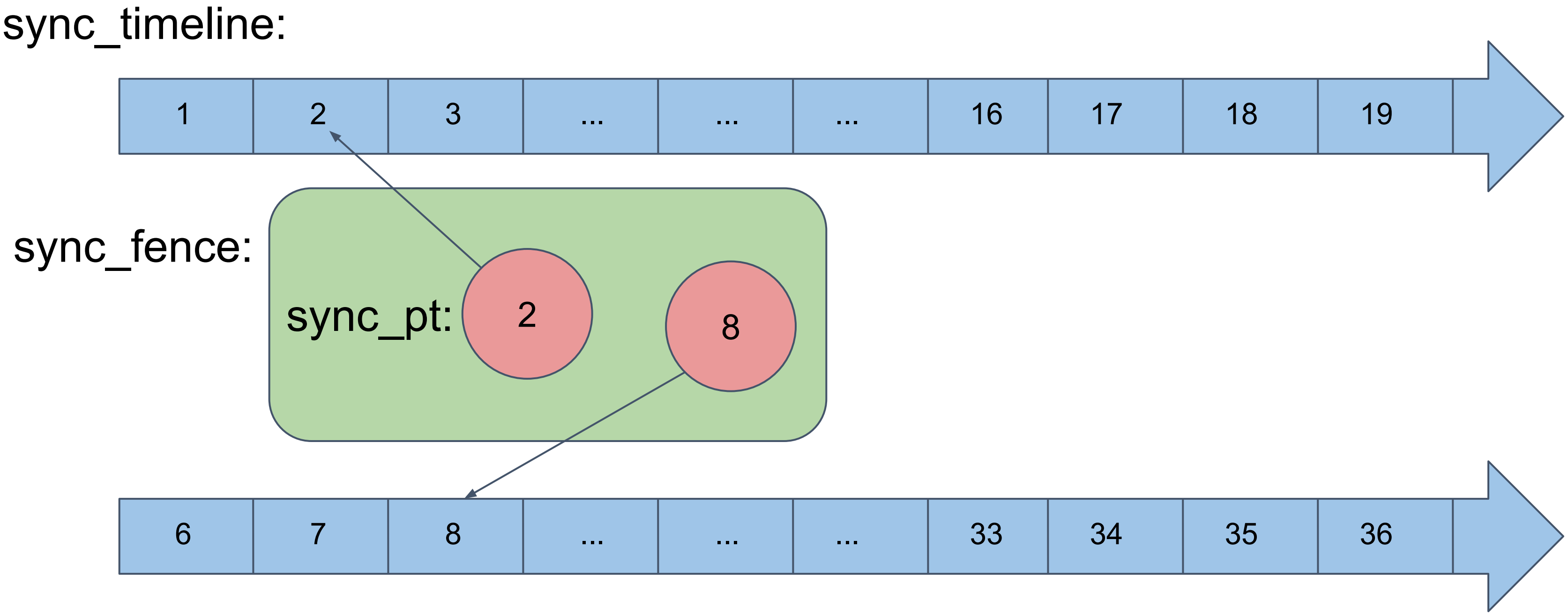
Ideally



Ideally



Android Sync Concepts

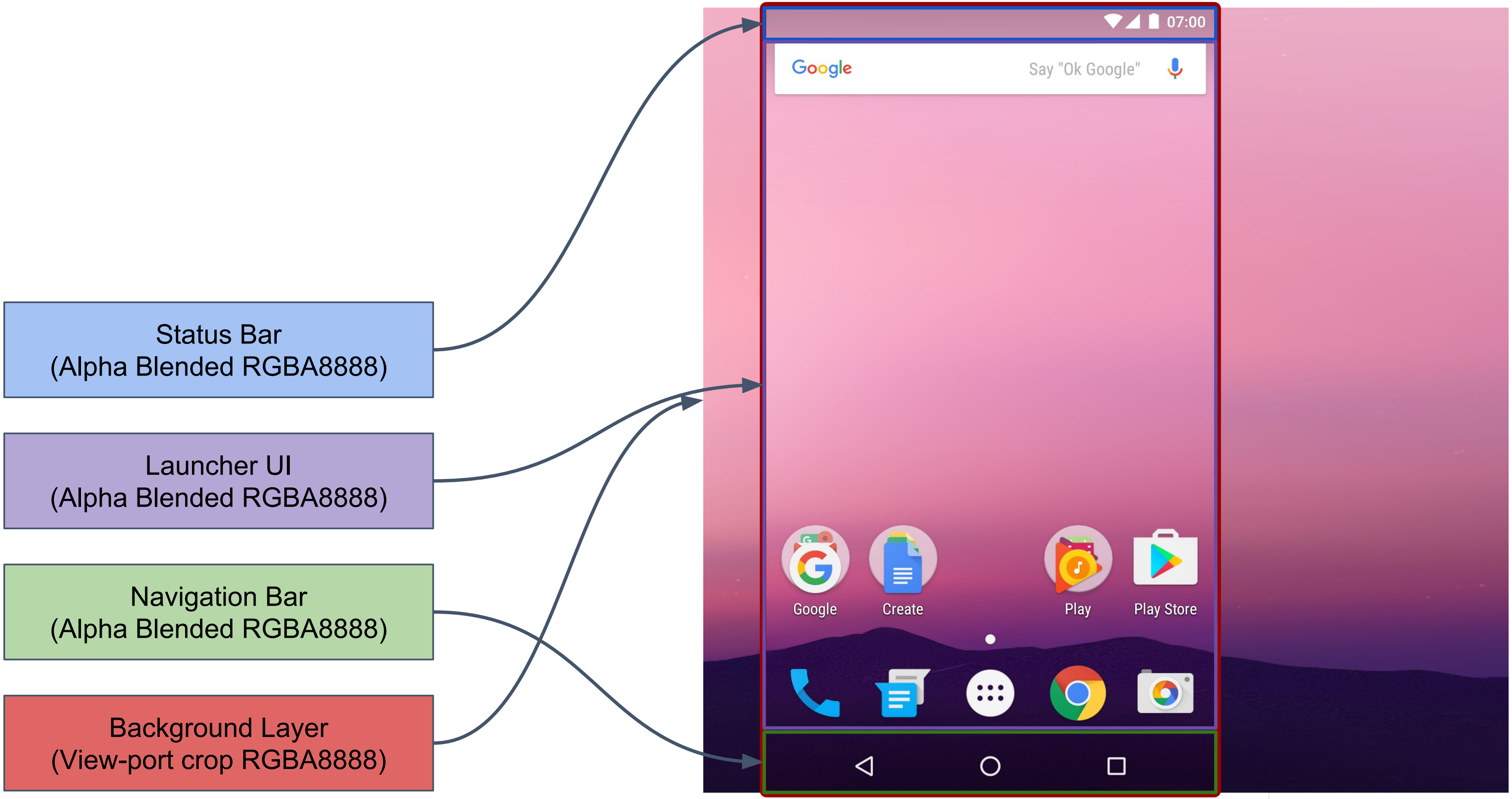


ADF - Atomic Display Framework (Lollipop)

Allowed for atomic changes to the display settings - all or nothing

- Avoids tearing, glitches, & getting into bad hardware states

API tied together with Sync points, allows for buffers to be marked ready and then after display marked as released.



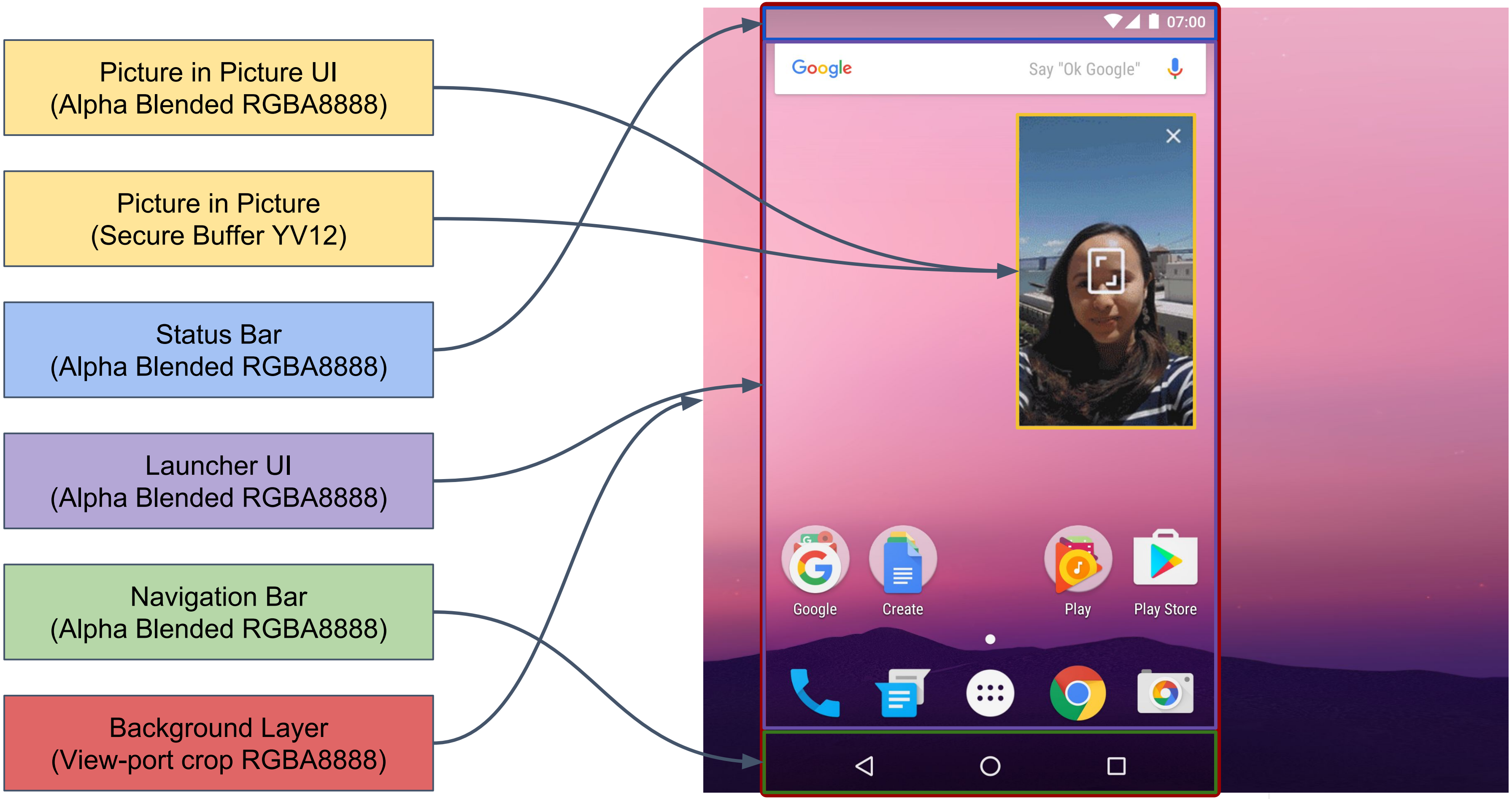
Status Bar
(Alpha Blended RGBA8888)

Launcher UI
(Alpha Blended RGBA8888)

Navigation Bar
(Alpha Blended RGBA8888)

Background Layer
(View-port crop RGBA8888)

Credit: By AOSP - Screenshot, Apache License 2.0, https://commons.wikimedia.org/w/index.php?curid=50949458



Picture in Picture UI
(Alpha Blended RGBA8888)

Picture in Picture
(Secure Buffer YV12)

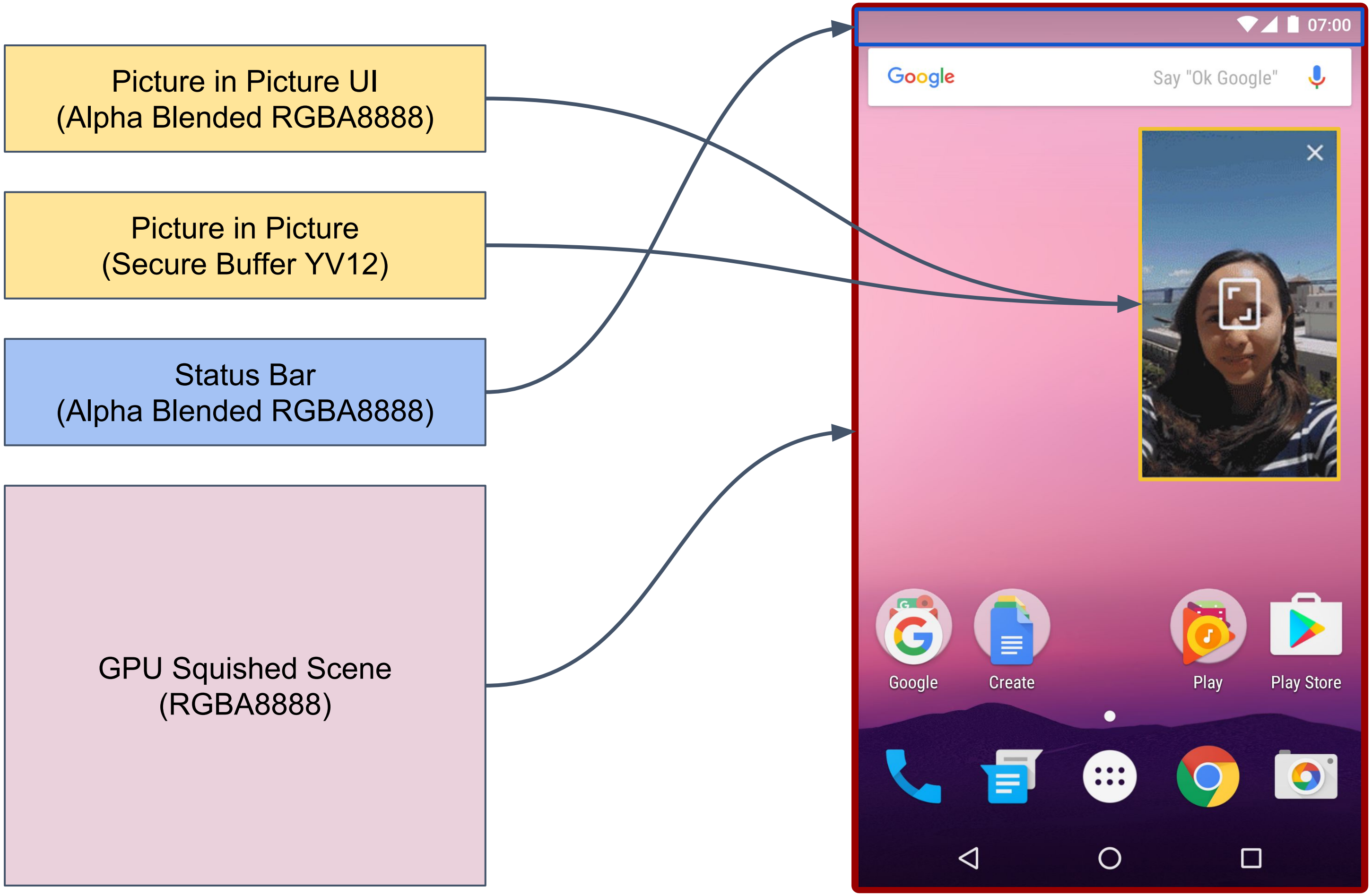
Status Bar
(Alpha Blended RGBA8888)

Launcher UI
(Alpha Blended RGBA8888)

Navigation Bar
(Alpha Blended RGBA8888)

Background Layer
(View-port crop RGBA8888)

Credit: By AOSP - Screenshot, Apache License 2.0, https://commons.wikimedia.org/w/index.php?curid=50949458



Picture in Picture UI
(Alpha Blended RGBA8888)

Picture in Picture
(Secure Buffer YV12)

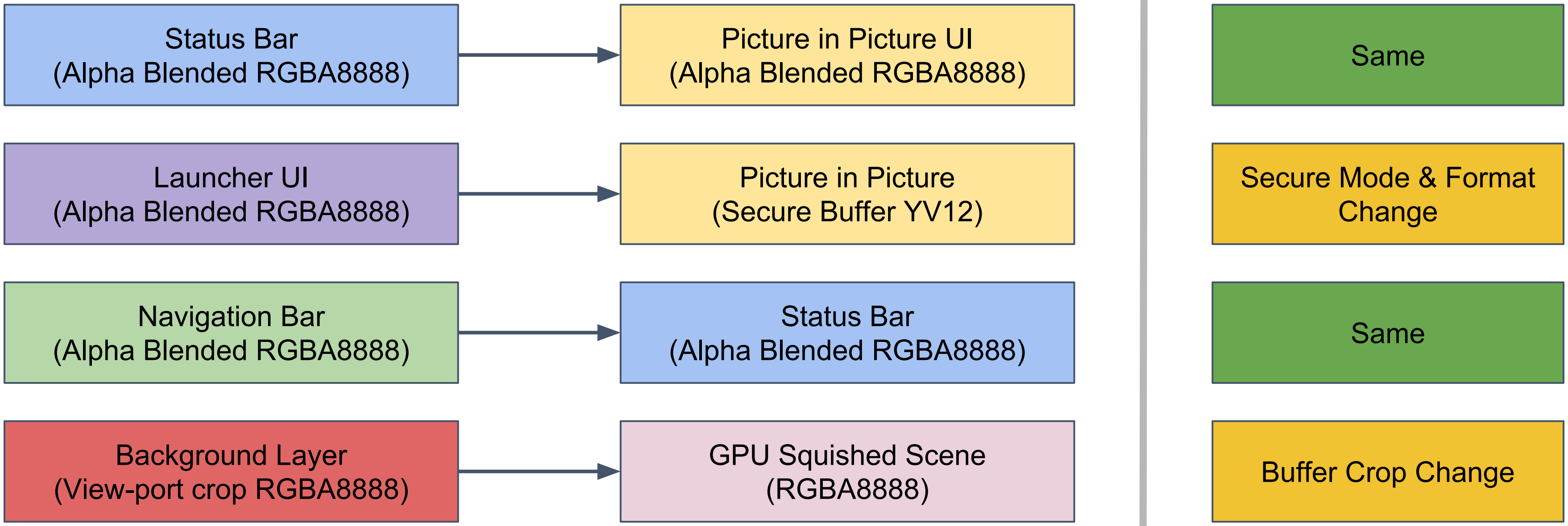
Status Bar
(Alpha Blended RGBA8888)

GPU Squished Scene
(RGBA8888)

Credit: By AOSP - Screenshot, Apache License 2.0, https://commons.wikimedia.org/w/index.php?curid=50949458

Display Layer Properties

(ignoring position & size changes)

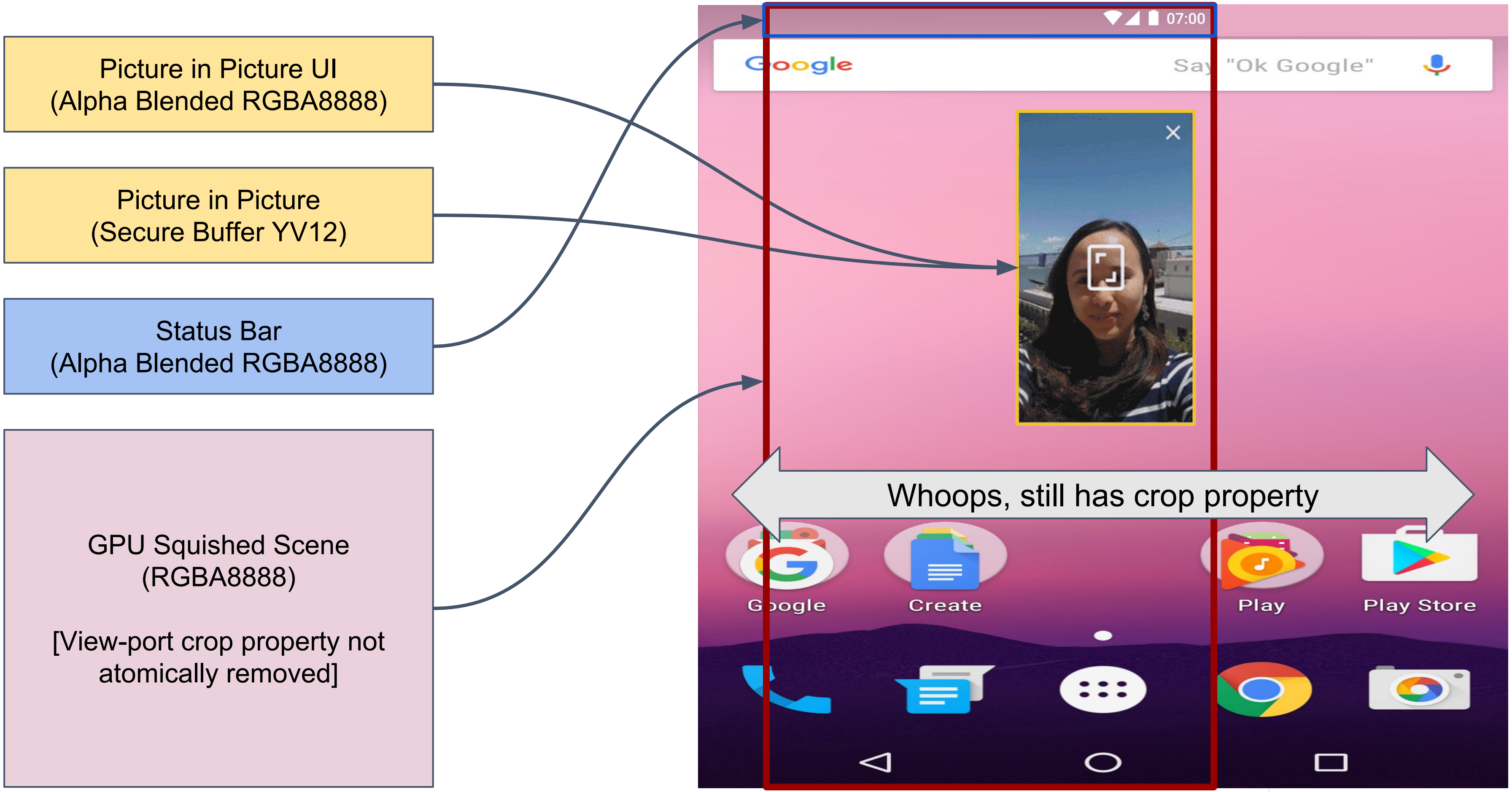


Status Bar (Alpha Blended RGBA8888) → Picture in Picture UI (Alpha Blended RGBA8888): Same

Launcher UI (Alpha Blended RGBA8888) → Picture in Picture (Secure Buffer YV12): Secure Mode & Format Change

Navigation Bar (Alpha Blended RGBA8888) → Status Bar (Alpha Blended RGBA8888): Same

Background Layer (View-port crop RGBA8888) → GPU Squished Scene (RGBA8888): Buffer Crop Change



Picture in Picture UI
(Alpha Blended RGBA8888)

Picture in Picture
(Secure Buffer YV12)

Status Bar
(Alpha Blended RGBA8888)

GPU Squished Scene
(RGBA8888)

[View-port crop property not atomically removed]

Whoops, still has crop property

Credit: By AOSP - Screenshot, Apache License 2.0, https://commons.wikimedia.org/w/index.php?curid=50949458

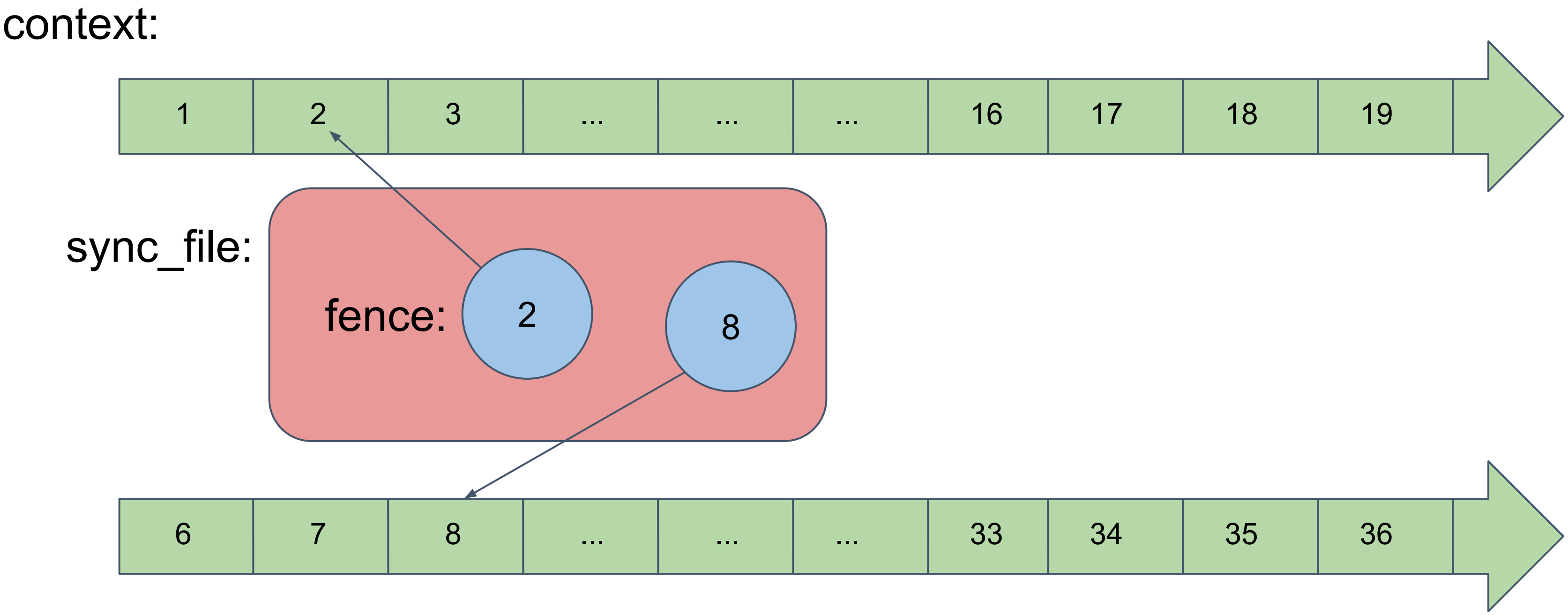
HWC2 (Nougat)

Increased the interface complexity (now 43 methods!)

Moved to non-speculative fencing (“buffer content no longer needed” to “buffer content is displayed”)

Additional support for multiple displays

DRM Fences (landed ~v4.6)



context:

1	2	3	...	...	...	16	17	18	19

sync_file:

fence: 2 8

6	7	8	...	...	...	33	34	35	36

Atomic KMS/DRM (~v4.10)

High-level equivalent to Android's ADF, but properly upstream

Allows for all-or-nothing changes to display settings

Signaled via DRM fences

Landed in 4.10 in the kernel (backported to android-4.9)

drm_hwcomposer

drm_hwcomposer

- HWC implementations, started by ChromeOS team
- Uses the kernel KMS/DRM interfaces
- HWC1 implementation shipped on the PixelC
- HWC2 implementation created, utilizing Atomic KMS interfaces, but never shipped
- Picked up by RobertFoss, RobHerring, and other community members
- Working on QEMU (virgil), db410c, db820c, raspi3, hikey, hikey960, + others.
- https://gitlab.freedesktop.org/drm-hwcomposer/drm-hwcomposer

drm_hwcomposer goals

Validate Atomic DRM/KMS

Provide a working default HWC implementation (replace fbdev)

- Minimize system specific portions of HWC (Importer and Planner).
- Allow systems to share the rest - fairly complicated logic

Motivate vendors to converge to DRM/KMS display solutions

drm_hwcomposer limitations

- Requires v4.10 (or android-4.9) or newer kernels
- Fancy pre-compositor work done by ChromeOS team dropped as it wasn’t used by anyone.
- Current supported drivers are very simple compared to shipping vendor implementations.
- Importer and Planner are also tied closely to gralloc implementations (currently gbm_gralloc, hisi_grallocs, and minigbm_gralloc).
- Needs a lot of cleanup and generalization

drm_hwcomposer TODO

Major cleanup needed in Validate/Plan logic

Look at reimplementing Planner optimizations around squishing planes that don’t change.

Writeback support

Wider support of more advanced hardware

drm_hwcomposer / libdrm / mesa3d

All are freedesktop.org hosted projects.

Recently updated AOSP projects to be closer to upstream

Work being done to upstream AOSP changes, and keep AOSP more closely aligned with upstream.

But this becomes more difficult to do as more shipping devices utilize these libraries

Possible Futures

Google deprecated ADF

From upstream perspective, Atomic KMS/DRM is the direction to go forward with.

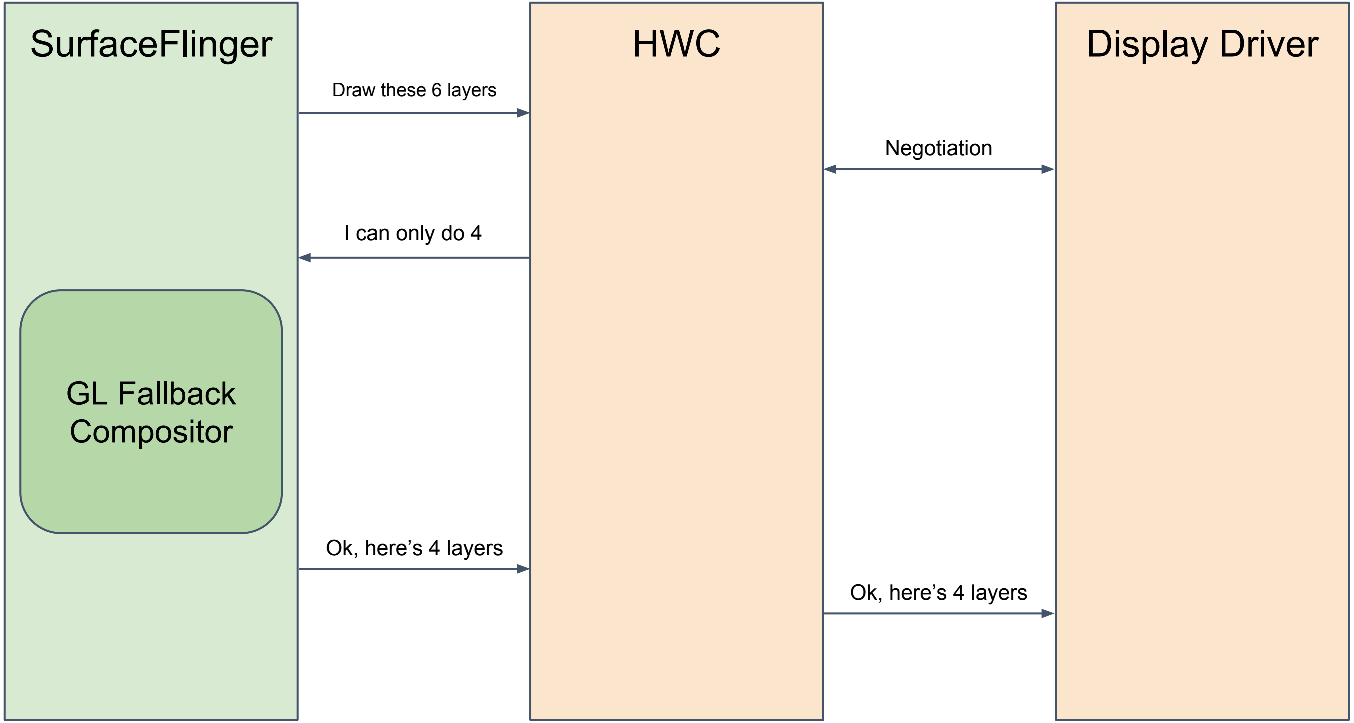
Many vendors unfortunately still do their own thing, utilizing legacy fbdev drivers

Atomic KMS/DRM may have some functionality limits & we need to address them

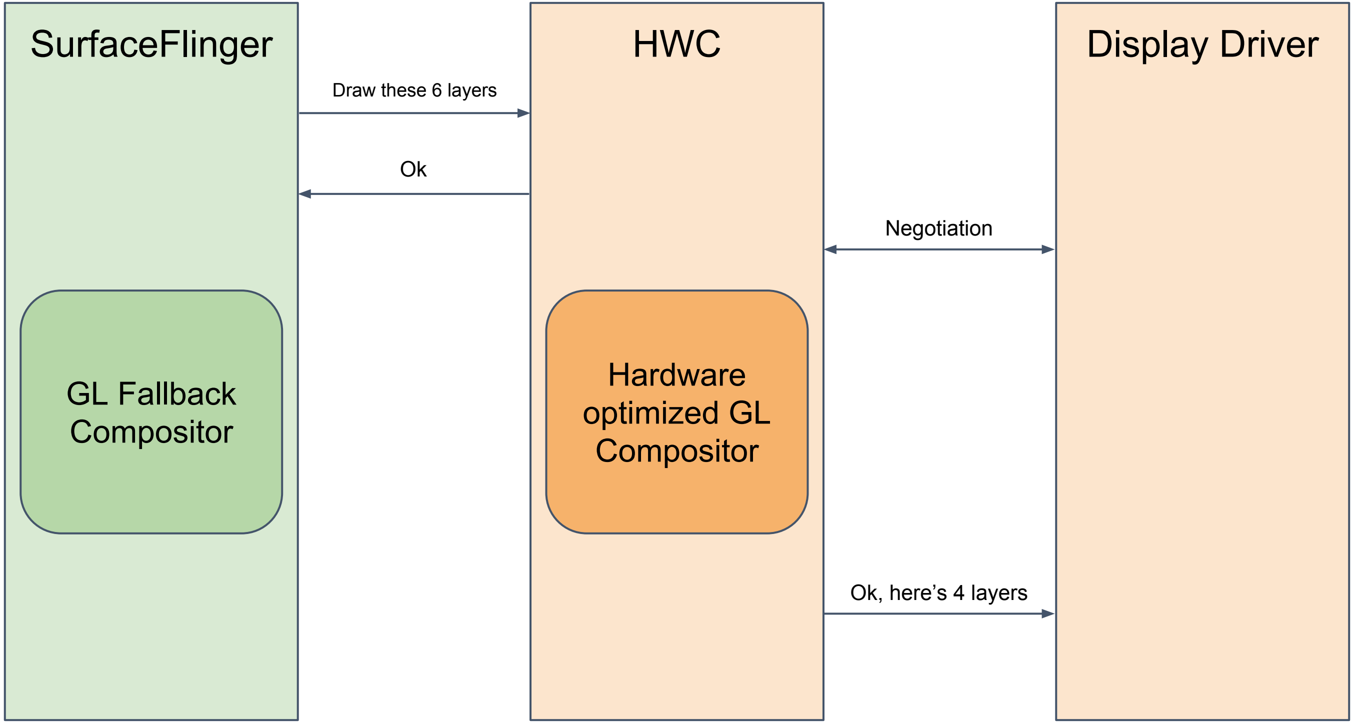
Need vendors to start migrating to help find and address those issues

Plans to use drm_hwcomposer to prototype Nextgen HWC/Gralloc

Design


SurfaceFlinger
HWC
Display Driver
Draw these 6 layers
Negotiation
I can only do 4
GL Fallback Compositor
Ok, here’s 4 layers
Ok, here’s 4 layers

In Practice



SurfaceFlinger

GL Fallback Compositor

Draw these 6 layers

Ok

HWC

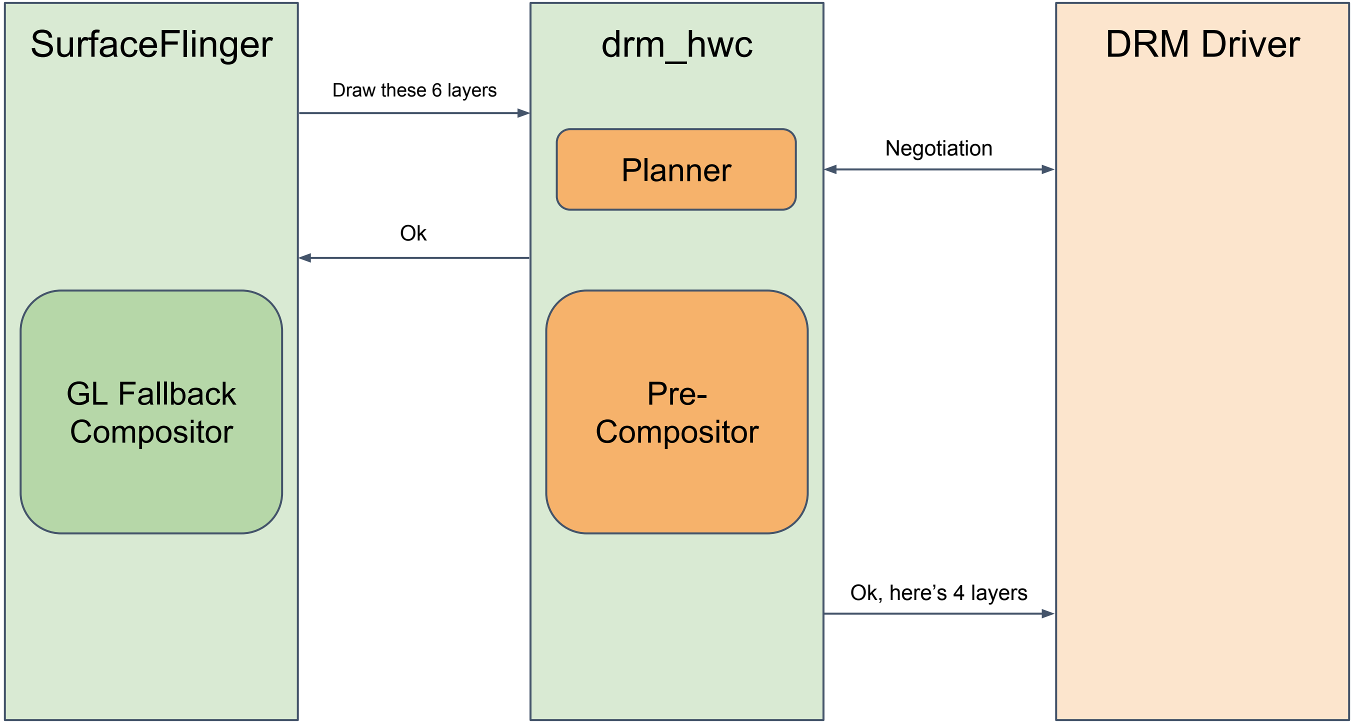
Hardware optimized GL Compositor

Negotiation

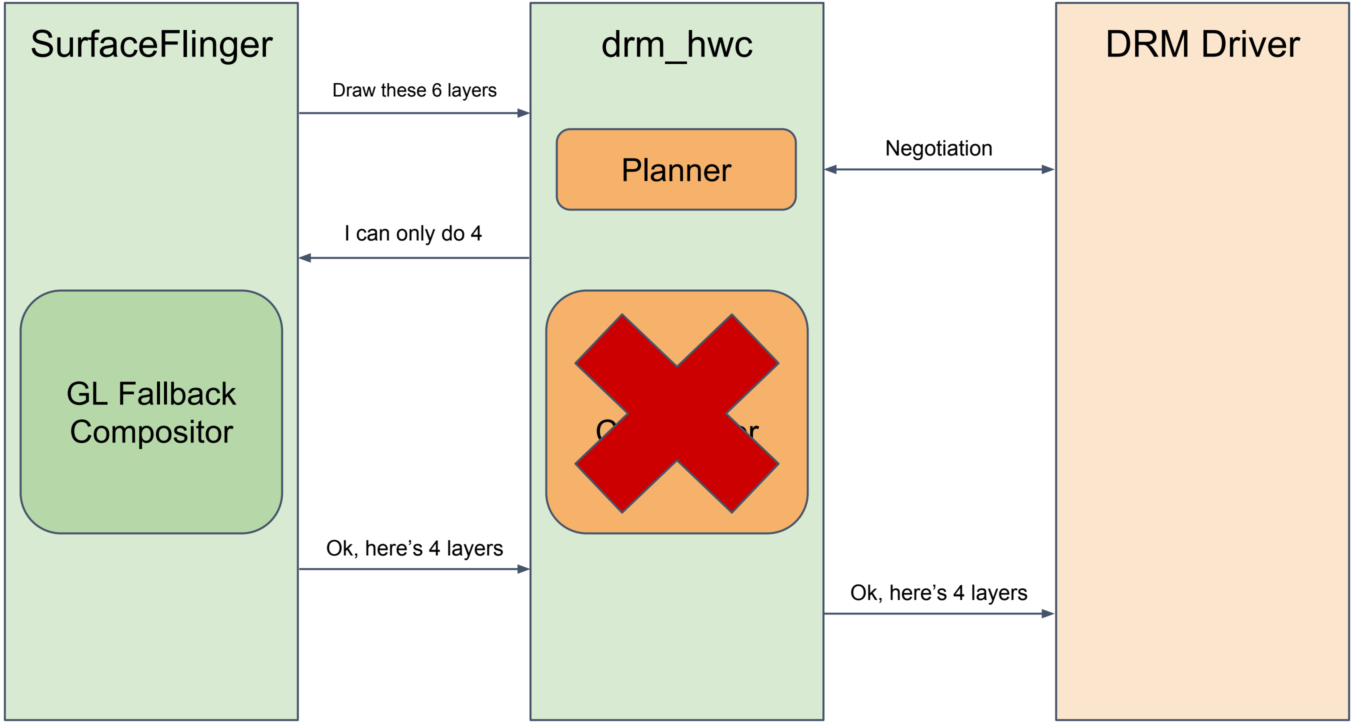
Ok, here’s 4 layers

Display Driver

drm_hwcomposer previously



Current drm_hwcomposer



SurfaceFlinger

GL Fallback Compositor

Draw these 6 layers

I can only do 4

Ok, here’s 4 layers

drm_hwc

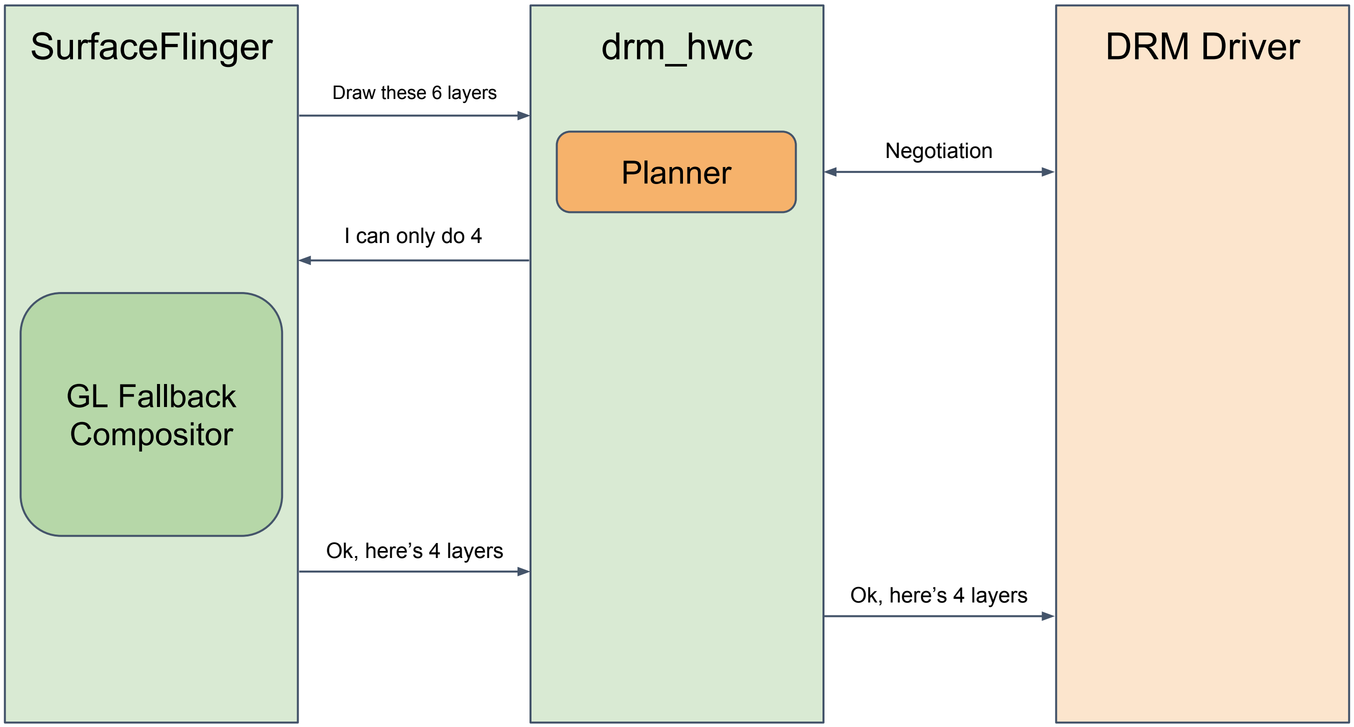
Planner

Negotiation

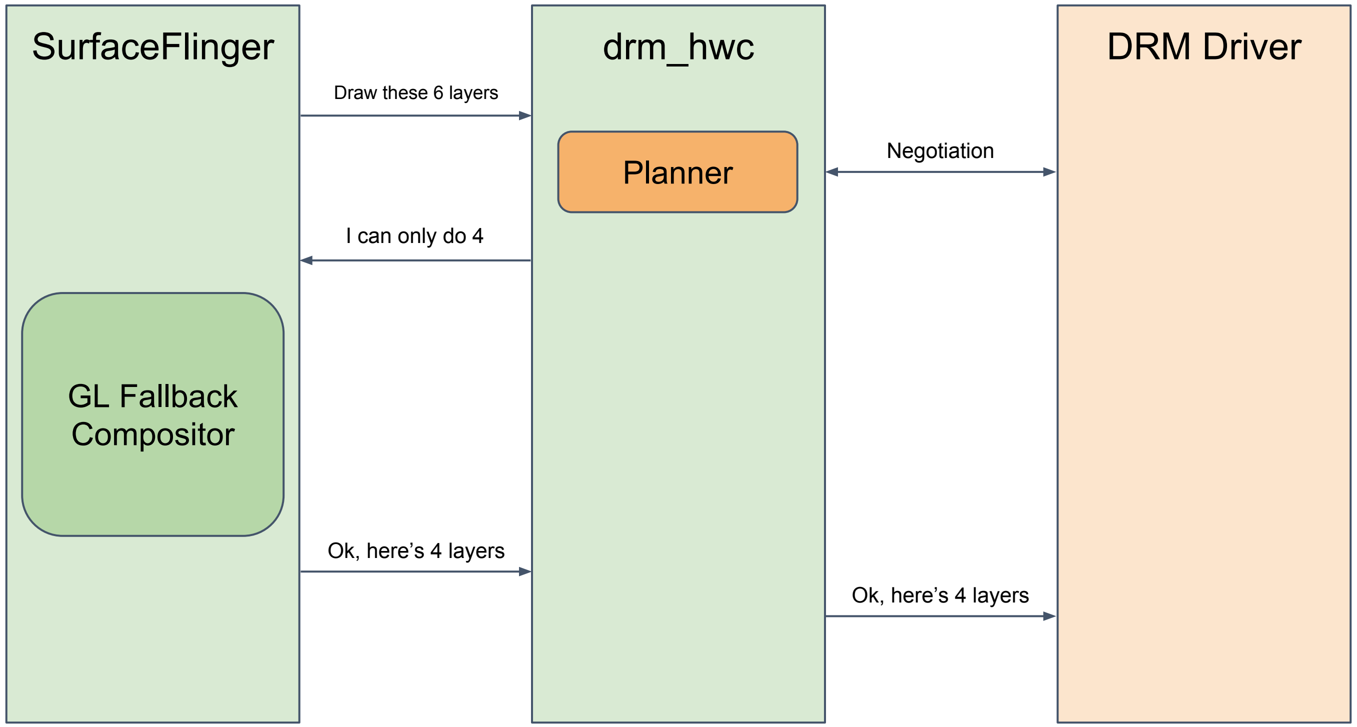
Ok, here’s 4 layers

DRM Driver

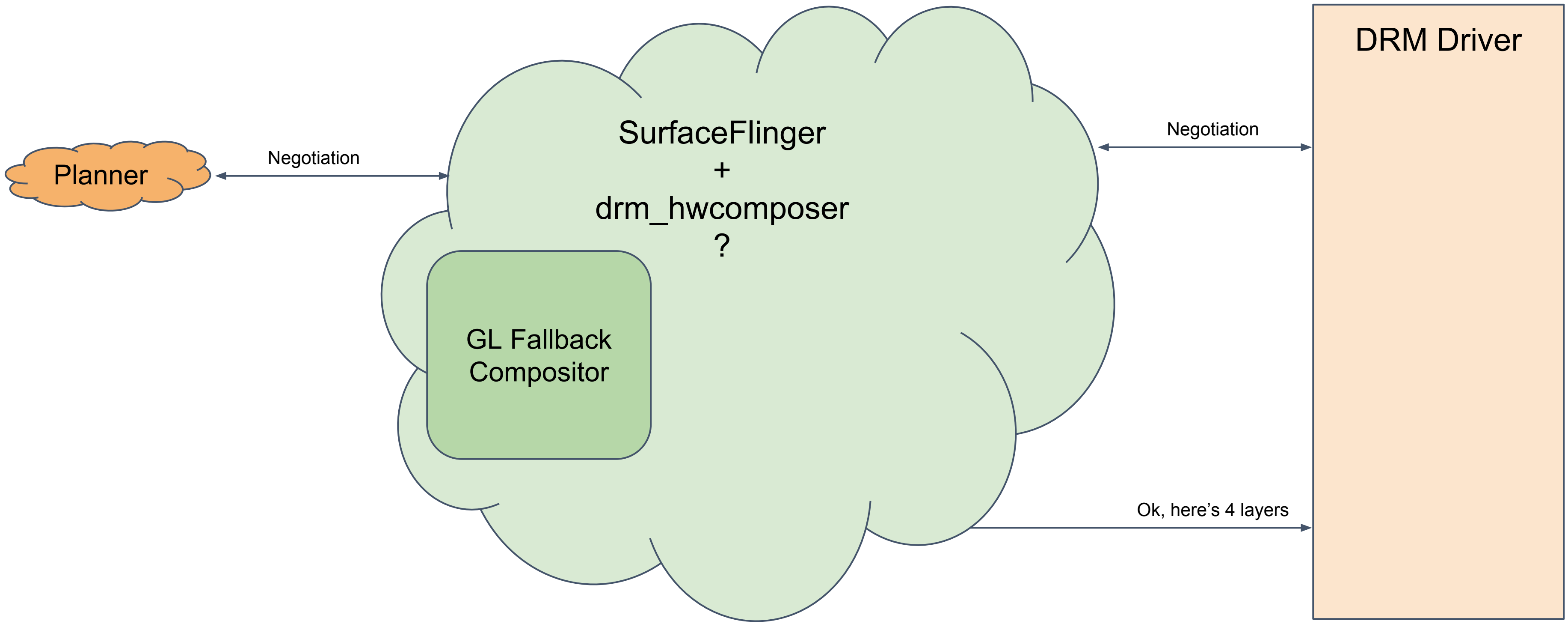
Current drm_hwcomposer



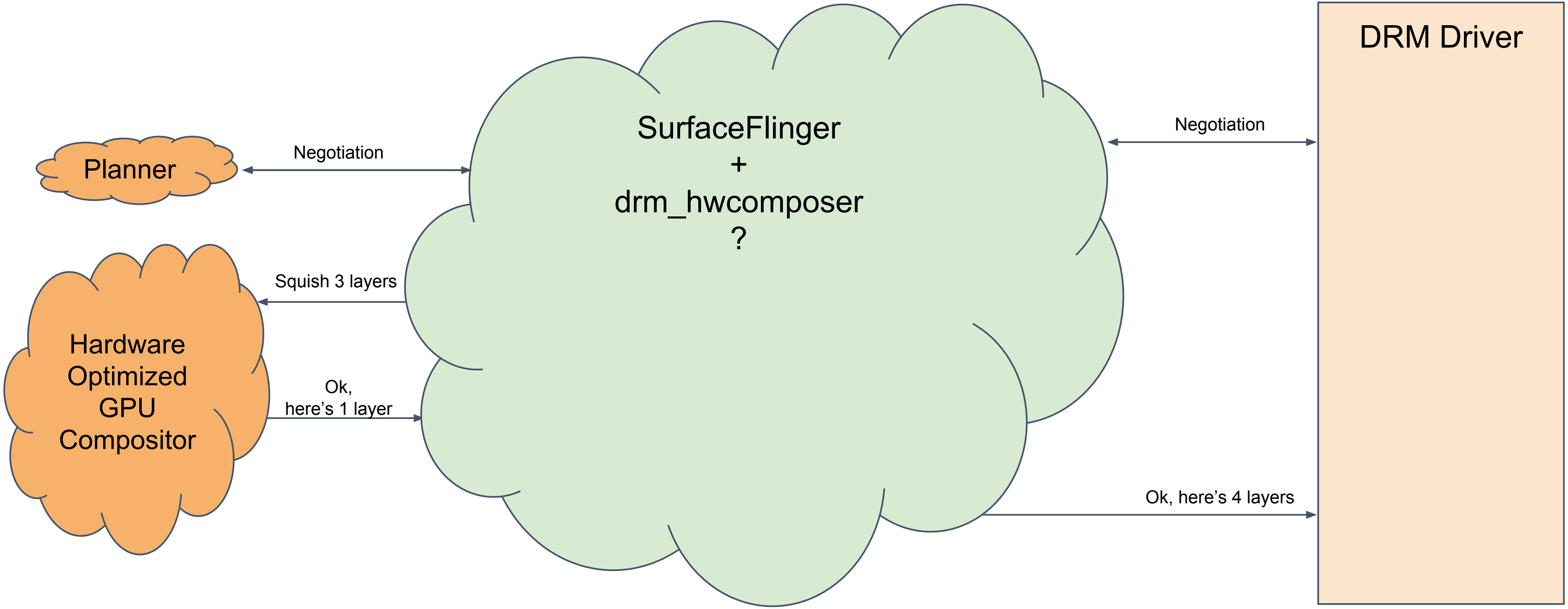
Current drm_hwcomposer



Potential future???



Potential future???



Suggestions

- **Stop using fbdev** drivers & migrate to DRM/KMS drivers
- **Work with community** to get DRM infrastructure expanded to fully support your device needs!
- Send your display developers to XDC! - https://xdc2018.x.org/
- **Upstream those DRM drivers!**

Credits

Eric Gilling's LPC13 talk: https://www.youtube.com/watch?v=rhNRItGn4-M

Greg Hackmann's LPC13 talk: https://www.youtube.com/watch?v=3wFqXysISQg

Marissa Wall, Sean Paul & Zach Reizner's LPC16 talk:
https://blog.linuxplumbersconf.org/2016/ocw//system/presentations/4185/original/LPC%20HWC%202.0%20&%20drm_hwcomposer%20.pdf

Marissa Wall, Sean Paul & Nat Duca's Prototyping Nextgen HWC/Gralloc via drm_hwcomposer doc:
https://docs.google.com/document/d/1wtkB2w2GL_oRJn4bHGvtF27wgv-EiYj5kF8KQtaBHI0/edit#

Portions of this slide deck are modifications based on work created and shared by the Android Open Source Project and used according to terms described in the Creative Commons 3.0 Attribution License.

Linaro connect
Vancouver 2018

Thank you!

john.stultz@linaro.org