DRM/KMS for Android

Status update
Linux Plumbers, November 2018
Alistair Strachan <astrachan@google.com>
Agenda

- Problem statement
- Timeline
- Where we want to get to
- Current status
- Future
Problem statement
Pre-DRM world

Kernel

3D Driver Core
Video Codec

FBDEV
Display Driver
Scaler

OpenMAX
gralloc
hwcomposer

EGL / Open GL ES
Vulkan

ANDROID FRAMEWORK
Issues

- Too many kernel display driver interfaces for Android
  No requirement to standardize

- Most shipping implementations still use fbdev-based / custom driver interfaces

- Code duplication in kernel and userspace
  Every partner has their own display code, hardware composer HAL

- The display drivers are not upstream
  They are not necessarily compatible across kernel versions with different userspaces

- Testing and debugging display drivers requires vendor-specific tools and methods
  Limits our ability to validate the display driver through userspace testing
Timeline
2013 - present
Collabora worked to de-stage the Android 'sync' driver, merge with DRM/KMS explicit fencing support
Collabora/Linaro/Google worked to add support for explicit fencing to drm_hwcomposer, Mesa, and to enable this on open-source boards

2016
August 2016: Android N ships with HWC 2.0. Supports non-speculative fencing, to accommodate DRM/KMS atomic in/out fencing
October 2016: Pixel 1 ships with 3.18, CONFIG_FB_MSM
December 2016: Linux 4.9 LTS released

2017
May 2017: android-4.9 receives in/out fence backports from Linux 4.10
October 2017: Pixel 2 ships with 4.4, CONFIG_FB_MSM
November 2017: Linux 4.14 LTS released
Present / 2018

October 2018
Pixel 3 ships with 4.9, **CONFIG_DRM_MSM**

October 2018
Linux 4.19 LTS released

Future / 2019+

Android Q
Partners need to ship android-4.9, android-4.14 or android-4.19 kernels
All Q kernels have prerequisite DRM/KMS changes

**DRM everywhere**
Where we want to get to
DRM world

- 3D Driver Core
- Display Driver
- DRM
- Video Codec
- Scaler
- libdrm
- gralloc (e.g. minigbm)
- drm_hwcomposer
- OpenMAX
- EGL / Open GL ES
- Vulkan
- ANDROID FRAMEWORK

Legend:
- DRM Rendernode
- New DRM stack
- DRM interop
- Vendor HAL (unmodified)
- Vendor-defined kernel interface
- Implementation challenge for vendor
Objectives

- One kernel display driver interface for Android
  No more fbdev-based / custom display driver interfaces. Shipping implementations all use DRM/KMS atomic modesetting drivers

- More shared code in kernel and userspace
  Shipping implementations all share a hardware composer, or perhaps the HWC HAL is removed altogether

- Better debugging capabilities
  Build userspace tools which work with upstream DRM drivers to help diagnose e.g. synchronization problems

- Better testing of the DRM/KMS atomic modesetting userspace interface
  Enable testsuites like intel-gpu-tools in Android VTS
  Require shipping implementations to pass the tests
Current status
Pixel 3

- First Pixel phone to ship with DRM/KMS; not fbdev-based

**drivers/gpu/drm (core files)**

```
$ git diff --stat v4.9.96..android-msm-bluecross-4.9-pie-dr1-release drivers/gpu/drm/drm_*.{c,h}
drivers/gpu/drm/drm_bridge.c |   26 ++
drivers/gpu/drm/drm_dp_helper.c |   13 +-  
drivers/gpu/drm/drm_edid.c |  504 +++++++++++++++++++++++++++++++++++++++-
drivers/gpu/drm/drm_fops.c |   7 +
drivers/gpu/drm/drm_framebuffer.c |  3 +-  
drivers/gpu/drm/drm_mipi_dsi.c |  59 ++++
```

7 files changed, 608 insertions(+), 25 deletions(-)

**drivers/gpu/drm/{msm,bridge}**

```
$ git diff --shortstat v4.9.96..android-msm-bluecross-4.9-pie-dr1-release drivers/gpu/drm/{msm,bridge}
```

195 files changed, 127235 insertions(+), 406 deletions(-)

- Partners working to upstream these changes
Android Open Source Project

- Stopped forking, updated to latest versions of various Open Source projects
  - libdrm, mesa, drm_hwcomposer
- Linaro helped enable DRM/KMS on Hikey, Hikey960 in AOSP master
- TI/Linaro helped enable BeagleBoard X15 with DRM/KMS in AOSP master
- Pixel 3 released to AOSP (pie-dr1-release)
- The intel-gpu-tools project will be added to AOSP soon
Future
Future

- All boards in AOSP are DRM/KMS atomic modesetting based
- All shipping DRM/KMS implementations tested via Android VTS
- More automated external display validation using Chamelium/Chameleon board

https://www.chromium.org/chromium-os/testing/chamelium
THANK YOU