Enable hardware acceleration for GL applications without glamor on Xorg modesetting driver

Christopher Michael <cmichael@igalia.com>

José María Casanova Crespo <jmcasanova@igalia.com>

XDC 2022 – Minneapolis, Minnesota, USA

Outline

- Xserver Modesetting + Glamor + DRI3
- Issues on hardware with limited GPU memory.
- Technical Approach
- Current Status

Xserver and Glamor

- Xserver modesetting driver makes X work if your GPU has KMS (Kernel Mode Setting) available.
- If your GPU has a GL driver, GLAMOR can be enabled. Glamor will take advantage of GL to do the 2D composition so pixmaps will be GL textures.
- Modesetting implementation of Direct Rendering Infrastructure (DRI) requires having GLAMOR enabled.

The issue with limited GPU memory

- Low-end devices like Raspberry Pi 1-3 (256Mb-1Gb RAM) have limited GPU memory addressing up to 256Mb (CMA). This is not an issue on Raspberry Pi 4 because GPU can access memory using the MMU.
- Since bullseye Raspberry OS release, all Raspberry Pi 1-3 users are using the modesetting driver.
- With Glamor enabled using Mesa VC4 OpenGL driver for RPi3 works fine unless you run out of GPU memory....

Running out of GPU memory

- Open multiple applications full screen at Full HD resolution, being one of them Chromium with 6-8 tabs. Every window/tab will consume GPU memory.
- If you are lucky the application crashes, if you are unlucky the Xserver memory allocation fails and then Xserver crashes. We don’t want this user experience.

Glamor disabled but we want DRI

- To avoid this issue on pre-Rpi4 devices glamor is disabled by default on Raspberry OS.
- But, no DRI2/3 is available. So no HW accelleration is used for OpenGL applications.
- So we explored if we can have a DRI3 implementation on modesetting driver that does not requires glamor to be enabled.

Technical Approach

- Use X Server software rendering for desktop
- Allow DRI imported/exported pixmaps using OpenGL/ES driver
- Found xf86-video-nouveau patch: "enable dri3 support without glamor"

Enabling DRI3

Implement four functions:

- dri3_screen_init: Sets up DRI3 extension function calls
- dri3_open: Opens drm device fd and authorizes
- dri3_pixmap_from_fds: Creates a pixmap to wrap a dma-buf fd
- dri3_fds_from_pixmap: Backs an existing pixmap with a dma-buf fd

Enabling DRI3 (cont)

- DRI3 extension implements sharing direct rendered buffers between DRI clients and the X Server
  - Clients allocate render buffers themselves
  - Uses more secure sharing method based on Prime DMA-Bufs (file descriptors)
- Present extension shows rendered buffers on the screen
  - Handles synchronization of screen updates to Vblank
- GBM used to allocate buffers for pixmap backing
  - Linear modifier is negotiated so buffers can be composited by Xserver

Current Status

- Implementation is functional with VC4 (RPi 1-3) and V3D (RPi4).
- X Server renders desktop using software rendering.
- GL applications work using gbm bo’s for pixmap backing
- Compositors (mutter) work with this DRI3 implementation.
- Patches submitted upstream for review:

  https://gitlab.freedesktop.org/xorg/xserver/-/merge_requests/945

Q & A

Thank you

We are hiring: https://www.igalia.com/jobs/

Enable hardware acceleration for GL applications without glamor on Xorg modesetting driver

Christopher Michael <cmichael@igalia.com>

José María Casanova Crespo <jmcasanova@igalia.com>

XDC 2022 – Minneapolis, Minnesota, USA