DRM-backend tests in Weston’s GitLab CI

Leandro Ribeiro
leandro.ribeiro@collabora.com
About me

• Interested in graphics and FOSS in general
• Working at Collabora since March 2020
  – Internship project in Weston
    • DRM-backend testing
Some background

- Weston has multiple backends
- Different ways to output content
- One of them is the DRM-backend
  - It uses libdrm to:
    - Select modes (refresh rate, color depth, screen resolution)
    - Handle DRM objects (connectors, CRTCks, etc)
    - Control the display pipeline
Display pipeline

- renderer (GL or Pixman) → screenshot.png
- framebuffer
- primary plane
- CRTC → encoder → connector
Display pipeline
Problem statement

• Weston has a test suite that runs in GitLab CI
  - We want to test as much as possible of Weston’s functionalities in an automated fashion
  - However we can’t assure that the machine that will run the tests has a graphics card available
  - So how can we run DRM-backend tests in Weston’s GitLab CI?
Problem statement

- Thoroughly testing the DRM-backend
  - Screenshots are very handy for testing purposes
  - We can use screenshots to compare composed images with reference images in the test suite
  - However Weston disable planes composition to take screenshots
Proposed solution

- Automated DRM-backend tests in Weston
  - For the rendering part, we don’t need to modify anything
    - Mesa has support for software rendering drivers (e.g. LLVMpipe)
    - They are automatically selected in the absence of real hardware
  - For the KMS part, we can use VKMS (Virtual KMS)
    - We can run Weston on top of it
    - Some modifications in the test suite are necessary
Proposed solution

- Changes in Weston’s GitLab CI
  - Compile a Linux image with VKMS enabled
  - Run a virtual machine using virtme (QEMU based) and the Linux image that we’ve compiled
  - Run tests inside the virtual machine
  - Very simple DRM-backend test introduced
Weston’s test suite

Weston

Need to perform animation

GL renderer

Need to perform pageflip

GL calls to something the CPU can handle (LLVMpipe)

Mesa

Linux with VKMS enabled

libdrm

memory

CPU
Writeback connectors
Writeback screenshooter

1. Wayland
2. Weston
3. Prepare atomic commit with writeback setup
4. DRM perform atomic commit and schedule writeback job
5. pageflip event
6. Weston waits for writeback completion
7. Weston overwrites client buffer with writeback content
8. Wayland
9. screenshot.png
Writeback screenshooter
References

- [PATCH v6 0/3] drm/vkms: Introduces writeback support
- Weston MR: VKMS support
- Weston MR: Writeback connector support
- Weston MR: Writeback connector screenshooter
- The DRM/KMS subsystem from a newbie’s point of view
- Kernel DOCS: KMS (Kernel Mode Setting)
- Kernel DOCS: Writeback connectors
Summary

- Need to automatically test Weston’s planes functionality in the absence of real hardware
- Run Weston on top of VKMS in GitLab CI
- Add support for writeback connectors in VKMS and Weston
- Add writeback screenshooter in Weston
- Tests are basic at the moment, but the plan is to extend the test coverage as VKMS extends its functionalities