State of DRM

(graphics driver subsystem, not the other thing)

Daniel Vetter, Intel OTC

Kernel Recipes 2016, Paris

tl;dr;

world domination progressing according to plan

kerneldoc

- much pretty, such awesome
- sphinx+rst+`/scripts/kernel-doc`
- https://dri.freedesktop.org/docs/drm/

kerneldoc … for DRM!

- everything atomic, see later
- drm_crtc.c split into ~10 files and docs added
- and more

dungeons&dragons

- DRI1 legacy drivers still exist
- 10+ years of bad UAPI and creative root holes
- now fully hidden behind `drm_legacy_*`
- … and core de-midlayered

IGT gpu tests

- port Intel tests to be generic
- starting to get used on many drivers
- plus CI systems!
- https://dri.freedesktop.org/docs/drm/gpu/drm-uapi.html#validating-changes-with-igt

userspace requirements

- require open source for any UAPI
- https://dri.freedesktop.org/docs/drm/gpu/drm-uapi.html#open-source-userspace-requirements

atomic display driver

- check/commit semantics
- flicker-free commit
- because hw
- because direct scanout saves power

atomic advances

- 20 drivers and counting, 2-3 more per release
- lots of small polish and boiler-plate removal
- docs, docs, docs
- modular helper library now fully proven

one atomic to rule them all

- drm_hwcomposer, for Android
- CrOS&Ozone, using wayland
- all the waylands

atomic future

- trouble with some legacy use-cases, cursors
- buffer allocation needs more UAPI
- benchmark mode

generic fbdev defio

- manual upload display to save memory bw
- FBDEV now remapped to KMS
- … dirty rectangle still missing for flips

simple display pipeline helper

- simple pipe + 1 connectors
- based on atomic, without the complexity
- DRM now (strictly) better than FBDEV

fences

- `struct completion`, but for DMA
- implicit: kernel takes care
- explicit: userspace passes fences around
- kernel-internally a `struct fence`

implicit fencing

- `reservation_object` on `dma_buf`
- TTM (amd/nouveau) always supported it
- support finally rolling out everywhere else
- for Linux desktop (both X&Wayland)

explicit fencing

- `struct sync_file`, exported as an FD
- more control for userspace
- less complexity in (vendor tree) drivers needed
- for Android

explicit fencing for rendering

- userspace fences fully destaged in 4.9
- EGL_ANDROID_native_sync for msm/freedreno in 4.9 and mesa-next
- interaction with implicit fencing is tricky

explicit fencing for atomic KMS

- Google made HWC2 to suit upstream
- blocked by embargo, hopefully 4.10
- drm_hwcomposer has support
- for details attend LPC

rsn Android on upstream*

* without a joke** of a graphics stack

** I'm biased ;-)

rendering?

- 2+1 vendor supported open drivers
- 1+1+1 reverse-engineered drivers
- plus virtual ones
- still dire :(

summary

- atomic rules them all
- docs, tests and lots of cleanup
- closed all major gaps for writing display drivers
- cross-driver fence for everyone
- even rendering shows some (slow) progress