Userspace ABI and GFX KMD upstreaming requirements

Daniel Vetter

Upstream Requirements

- Linus Torvalds: Never break existing userspace, ever
- Dave Airlie: Everything in the graphics subsystem needs open-source userspace http://airlied.livejournal.com/73115.html
- These are pretty much the only two rules we can’t bend/break/change and get away with it. They are also really different from a vendor/proprietary driver perspective and often result in unplanned delays when upstreaming features.

What is all considered Userspace ABI

- Everything that userspace might notice
- Obvious cases IOCTLs, structs but also a lot more:
- Register bits preserved for userspace, e.g. content protection
- Special programming sequence to cooperate with userspace, e.g. hw locks
- Unintended/documented side effects of everything
- w/a, tuning bits settings that make existing userspace hang the gpu/crash more often
- https://xkcd.com/1172/
- ...

Why Open Source Userspace for GFX KMD?

- KMD/UMD are very tightly coupled for graphics drivers.
- Without source code access it’s often impossible to debug regressions and more important to fix them without breaking other versions of the UMD out there.
- Never changing drivers to avoid regressions once merged isn’t an option - Linux kernel internals change constantly and fast.
- Once merged ABI is cast in stone and can’t be fixed, need both pieces for (public) review to make sure the new interface accomplishes its stated goals.
- Hence requirement from Dave Airlie that a) “no regressions” policy from Linus Torvalds only applies to open source userspace for gfx drivers and hence b) open source userspace is an upstream merge requirement.
- Dave Airlie explicitly rejects “toy” userspace like simple demos or testcases to fulfill this requirement - it must be real UMD code.

Best practices

- Take open source UMD requirements into account in the planning phase already to avoid unexpected blockers.
- Pull in OTC UMD people from the start to avoid surprises when reviewing the interface.
- For smaller features: Just build the UMD support yourself since that avoids external dependencies for delivering features to upstream. It’s open source, so this is totally fine!
- When reusing existing interfaces make sure the open source UMDs have matching use-cases.

Corner-cases to Avoid

- Sneaking something in without telling, it’ll get reverted eventually. E.g. preserve bits for content protection in userspace (by claiming that the BIOS needs that).
- Relying on ioctls in different use-cases than open source userspace and planning to ship with these, sooner or later it’ll be the use-case that gets broken in a regression fix for open source UMDs. E.g. pin IOCTL and GPA.
- Develop or even submit for merging the KMD side without the open source UMD side having even started yet. It will result in endless delays.
- Generally don’t push the boundaries if you want to ship on vanilla upstream.

Q & A