A JOURNEY THROUGH UPSTREAM ATOMIC KMS TO ACHIEVE DISPLAYPORT COMPLIANCE

ELC 2017, Portland OR
Manasi Navare
Open Source Technology Center, Intel
THE ULTIMATE GOAL

Make Intel graphics kernel driver DisplayPort* compliant and upstream it.
“A journey of a thousand miles begins with a single step”
WHAT IS DP COMPLIANCE?

Monitor 1

DP cable

Monitor 1

VESA Ultra HD 8K

Intel
WHAT HAPPENS WHEN YOU CONNECT A DP CABLE?

- **Hot Plug Detect Signal**
- **DPCD Read/Write**
- **Serialized/Encoded Data at Link Clock**

**PC DP Source**
- Main-Link
- AUX CH
- HPD

**Monitor DP Sink**
- HPD
- AUX CH
- Main-Link

**DP cable**
DISPLAY PORT LINK TRAINING

- **Main Link Parameters:**
  - Lanes – 1, 2 or 4
  - Link Rate – 1.62, 2.7, 5.4 or 8.1 Gbps/lane

- **Link Training:**
  - DP source configures the main link through link training sequence

---

**Diagram:**

1. HPD
2. Link Training Starts
3. Clock Recovery
4. TPS1
5. TPS2/3
6. Channel Equalization
7. Link Ready
8. Lane count/link rate set
HOW AM I GOING TO TEST COMPLIANCE?

Device Under Test

DP source

DP Monitor

DP sink

DP reference sink

DPR 120

Compliance Test Suite SW

USB

DisplayPort
How does this project map to the graphics stack?

Intel Integrated Graphics Device

Intel Graphics Driver I915

Linux Kernel

Libdrm

Intel GPU Tools

DDX

X Server

Userspace
WHAT IS KERNEL MODE SETTING?

Process of setting up clocks, scanout buffers, initializing the chip and lighting up displays.
WHY THE ATOMIC KMS ‘TWO STEP?’

Properties

Userspace

DRM_IOCTL_MODE_ATOMIC

state

plane

CRTC

connector

Kernel

device

plane

CRTC

connector

->atomic_check()

->atomic_commit()
FINALLY GOT THE BALL ROLLING!!!
WAIT, DID I ????
"ANYTHING THAT CAN GO WRONG, WILL GO WRONG"

DP compliance test FAILURE...
PROBLEM: DOES THE DRIVER HANDLE LINK FAILURES?

Link Failures, black screen...

- Userspace
- Kernel
- DP Sink

Request mode 2560x1600

Validate mode/Setup CRTC

Update HW/Link Training

Check()

Commit()

ERROR: Link training failed

Monitors

LINK FAILURE

Black screen
SOLUTION: STABLE LINK = SUCCESSFUL MODESET = PERFECT FRAME

Userspace

- Modeset 2560x1600
- Uevent indicating HW configuration changed

Success mode

Kernel

- Validate mode
- Link Training
- Check()
- Commit()

- Link Status = BAD
- Link Re-Training
- Link trained at 2.7 Gbps, 4 lanes

Success mode

Modeset at lower resolution

Link Status = GOOD

Success, mode set to 1920x1440

DP Sink

Monitor

Commit()
WHY ASYNCHRONOUS REPORTING THROUGH THIS PROPERTY?

1. Atomic check guarantees the requested mode
   - Atomic commit never supposed to fail
   - Exception: Link training can still fail

2. Link might fail after a successful modeset

3. Atomic allows non-blocking commits
   - Return to userspace before modeset has completed
DEVELOPERS ARE ALL AROUND THE GLOBE! COMES WITH UPSTREAMING CHALLENGES!
LINUS’ RULES = UPSTREAMING RULES

RULE 1: NO REGRESSIONS!!

- NO GPU HANGS!!
- NO BLACK SCREENS!!
RULE 2: Never Blame Userspace, it’s always Kernel’s fault!!

- Kernel patches require changes in userspace drivers
- How can you upstream the kernel changes without testing userspace changes?
MY JOURNEY...

- Steep Learning curve - Make use of the community
- Submit patches – First step outside your comfort zone
- Feedback is always constructive, don’t take it as criticism
- You will see a finish line – Don’t give up!
Q&A

Upstream i915: git://anongit.freedesktop.org/drm-intel
Documentation: https://01.org/linuxgraphics
Email: manasi.d.navare@intel.com