A low latency GPU engine based reset mechanism for a more robust UI experience

Carlos Santa

Agenda:

- Problem Statement
- What’s the limitation in the GPU driver
- Proposed Solution: What is Timeout Detection and Recovery (TDR)
- How low can the latency be?
- A word about preemption
- Status of TDR in upstream
- Q/A

Problem statement: Stability and Robustness

- Looking at a specific stability problem affecting the UI experience under Intel Architecture when running GFX/Video playback use cases (video streaming type of app)

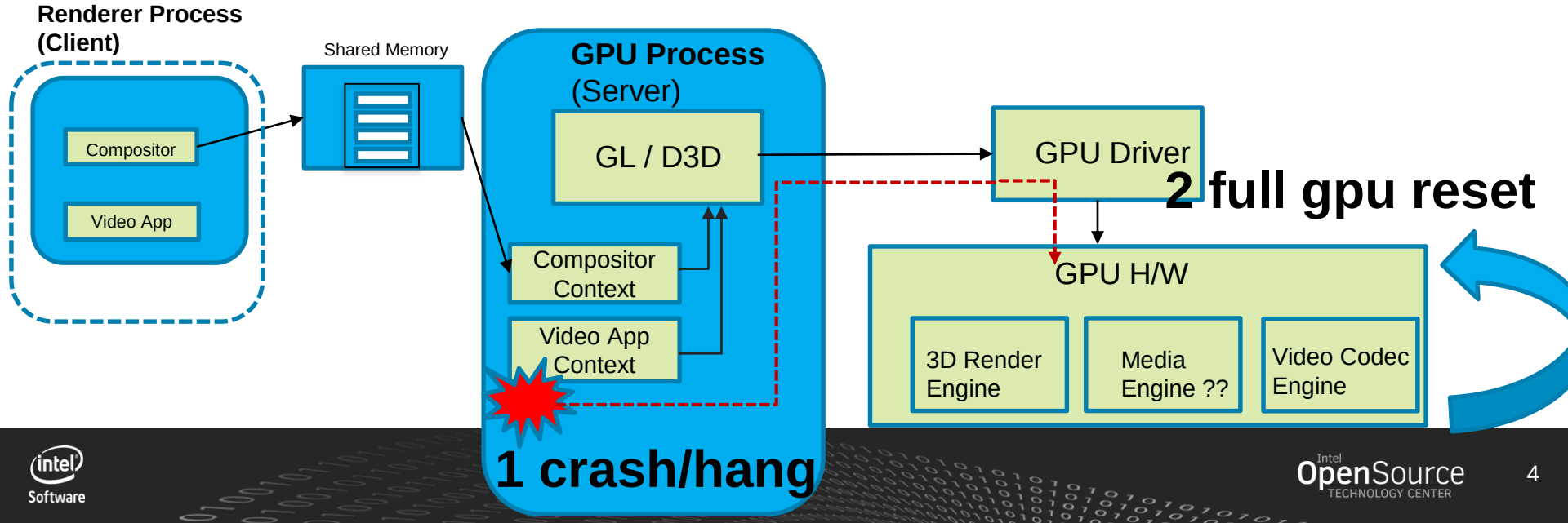
- The behavior was a **frozen UI**, followed by a **black screen** followed by **system reboot** (of course after some random time interval (hours to long long hours)).

- Spent some time understanding the GFX architecture in Chrome OS as well as a possible solution that could help here.

Current limitation

1. If a 3D client app “hangs” the GPU then the GPU process may get killed followed by a full GPU reset.

2. For a complex use case such as video decode many frames/objects are currently in flight so killing the GPU Process and resetting the GPU causes undesirables effects. We then realized…

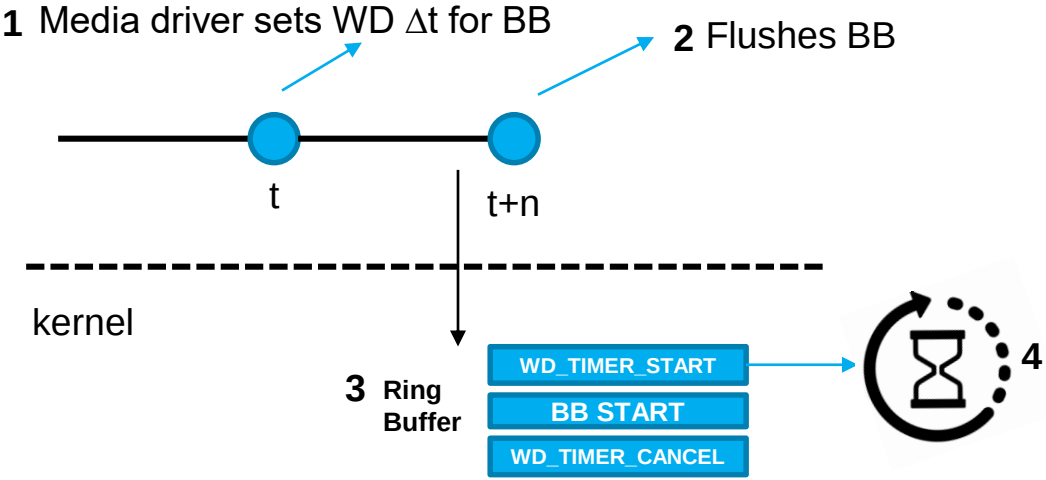


Proposed solution: Timeout Detection & Recovery

- New feature for Intel GPUs (upstreaming is wip) that can increase both stability and robustness by allowing ***applications*** to enable hang detections on individual batch buffers.
- **Timeout Detection** and **Recovery** (TDR) allows for the different engines in the GPU to be reset independently (as opposed to a full GPU reset).

- Generally speaking, the implementations introduces a new IRQ handler in the i915 driver as well as two new gpu watchdog command instructions ***before*** and ***after*** the emitted batch buffer's start instruction in the GPU's ring buffer.

TDR: Step by step



4 WD runs until a given time threshold Δt or the WD_TIMER_CANCEL is reached.

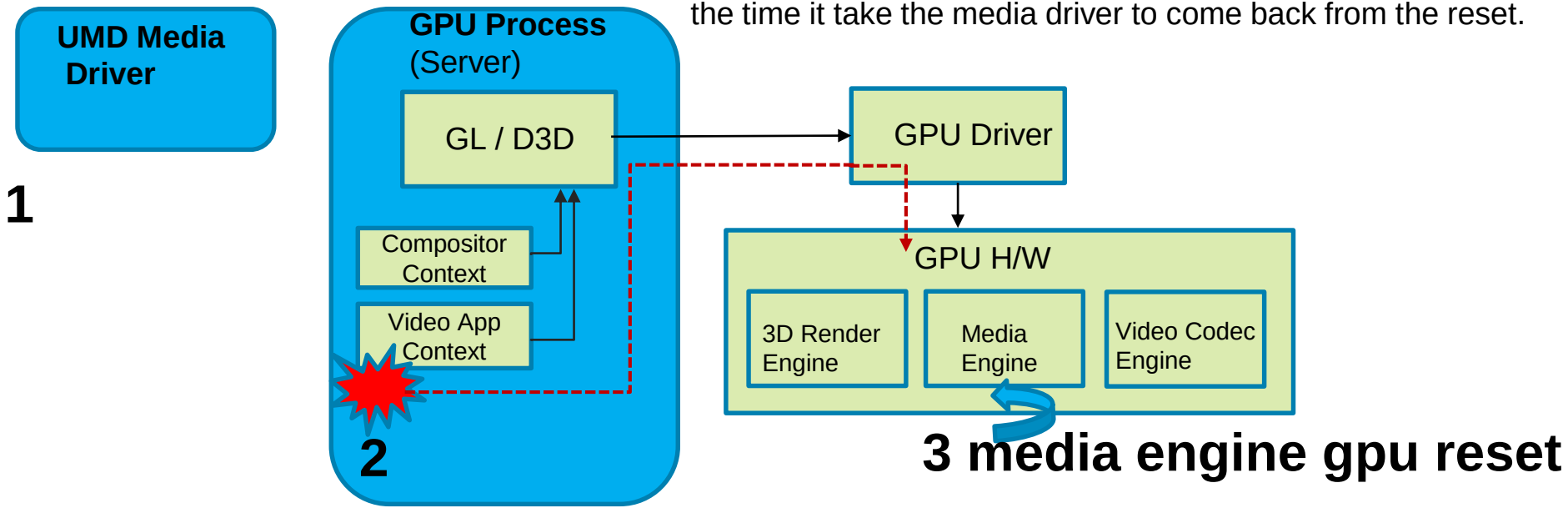
5 If the timer reaches the Δt then an interrupt is fired and is handled by the IRQ. A GPU hang is detected!

6 If the BB completes before the Δt and execution reaches WD_TIMER_CANCEL then WD is cancel and nothing happens.

Δt = threshold
WD = GPU watchdog
t = time interval

Proposed solution:

1. UMD Media Driver starts the watchdog timer after sending batch buffers
2. At some time later the media engine is detected to be in hung state after the watchdog timer has expired
3. The GPU driver resets only the affected media engine
4. Because the UMD Media driver knows when the faulty batch got submitted it could take actions during the the time it take the media driver to come back from the reset.



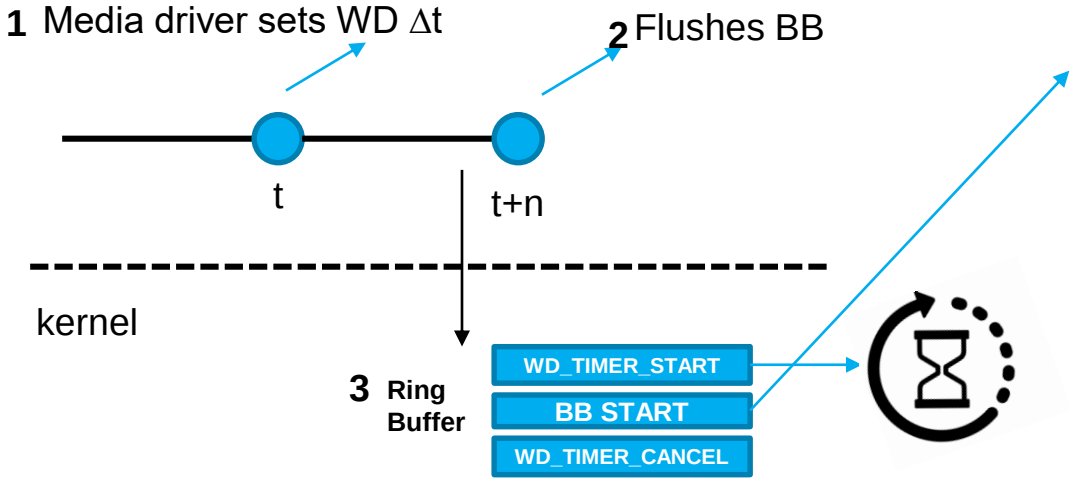
How low can the latency be?

- The whole mechanism works by an arbitrary threshold value that can be set from the application through an ioctl.

- However, the ***threshold*** can't be too low or else it can generate too many false positives.

- Right now, we are setting the threshold value with respect to the screen resolution (1080p=50ms, 4K=100ms, 8K=500ms and 16K=2000ms), however, we are still evaluating all these values.

A word about preemption



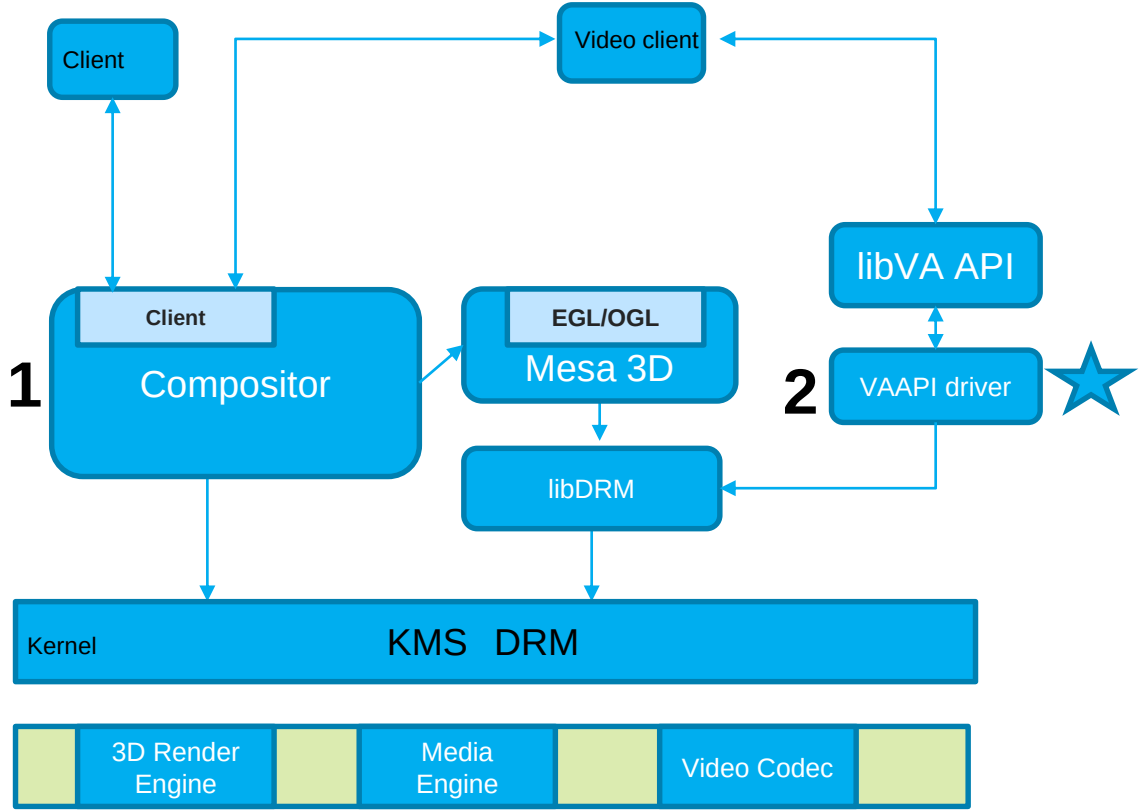
What happens if the BB sequence gets preempted before the WD timer gets canceled?

During preemption, the driver must cancel the WD_TIMER_CANCEL command as part of the preemption sequence.

What happens to the timer that was already ticking?

Δt = threshold
WD = GPU watchdog
t = time interval

How a compositor could benefit?



1. A compositor is fundamentally tasked to produce frames

2. In the past, by the time we detected that the GPU was hung it was too late for the compositor to recover (screen freeze, green or black screen or a system reboot).

3. A video client app can now determine early on whether a “task” has caused the Media Engine to crash and if so flag to the compositor to show the current frame while the Media Engine comes back from the reset.

Status of TDR in upstream:

	Accepted in upstream	Comments
TDR – Reset Engine	✓ Yes	
TDR – with GuC	WIP	
TDR - Watchdog	WIP	
IGT – TDR Watchdog	WIP	
	Prototype	Comments
TDR - Watchdog	Ubuntu OS w/ drm-tip	iHD and i965 Media Stacks
ffmpeg media decode	Ubuntu OS w/ drm-tip	validated
Video APK ARC++	Chromium OS – cros-4.14	validated

How to get involved?

- All of this work is happening in upstream
- **TDR kernel patches**
  - Code review: https://lists.freedesktop.org/archives/intel-gfx/2019-January/185543.html
- **i965 Media Driver in user space**
  - Code review at: https://github.com/intel/intel-vaapi-driver/pull/429
  - I can be reached on **IRC** as **csanta**
    work email: carlos.santa AT intel.com

Questions or feedback?