PRIME Synchronization

XDC 2016

Alex Goins, Andy Ritger

NVIDIA

PRIME Output Slaving: Synchronization

Introduction: PRIME Output Slaving

Enables the sequence:

- One GPU renders and transfer pixels through GEM shared buffers.
- Another GPU displays the results.

Useful for a variety of cases:

- Optimus laptops:
    - Integrated GPU (iGPU) connected to display, discrete GPU (dGPU) is not.
- USB DisplayLink adapters.
- Desktop motherboards with 'iGPU Multi-Monitor'.

PRIME Output Slaving: Synchronization

PRIME Output Slaving Terminology

"output slave":

- The GPU which drives the display.
- Receives pixels from the "master" GPU.
- "sink"
- For this discussion, typically an Intel integrated GPU.

"output master":

- The GPU which does the rendering.
- Delivers pixels to the "slave" GPU.
- "source"
- For this discussion, typically an NVIDIA discrete GPU.

PRIME Output Slaving: Synchronization

Problem Statement

Until recently, PRIME output slaving was unsynchronized and single-buffered:

- RandR handshake between output master and output slave:
  - Share a single screen-sized GEM buffer.
- Output master writes to buffer whenever.
  - Typically every damage event, or maybe batched.
- Output slave reads from the buffer whenever.
  - Typically at the refresh rate of the monitor.
- This results in tearing.

PRIME Output Slaving: Synchronization

Step #1: Explicit Buffer Updates

- Add ScreenRec::PresentSharedPixmap().
- Instead of master updating shared buffer when it wants:
- Output slave calls master's PresentSharedPixmap() after the previous frame has been displayed.
    - Typically after vblank notification.
- Output master's PresentSharedPixmap() updates the shared buffer.

Better, but still inherently racy.

PRIME Output Slaving: Synchronization

Step #2: Double Buffering

- X server RandR code creates two shared GEM buffers.

- Add rrScrPrivRec::rrEnableSharedPixmapFlipping().
  - X server RandR code calls output slave's rrEnableSharedPixmapFlipping().
  - Instead of rrScrPrivRec::rrCrtcSetScanoutPixmap().

- Output slave alternates between these buffers
  - Call output master's PresentSharedPixmap() for each buffer on alternating frames.

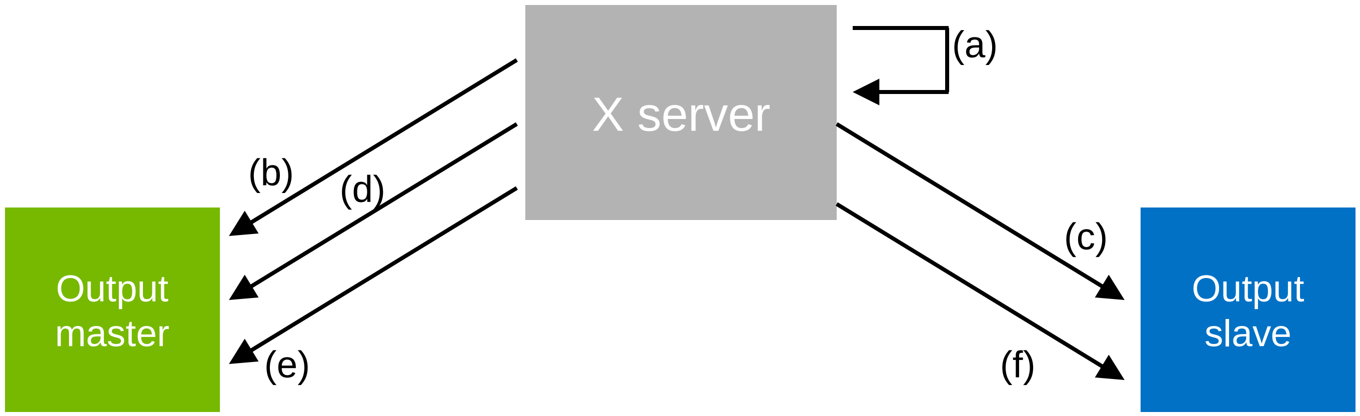
PRIME Output Slaving: Synchronization

Step #3: When Idle, Wait for Damage, not VBlank

- The sequence described so far requires an event every vblank.
- Wasteful if nothing is changing.
- Add ScreenRec::RequestSharedPixmapNotifyDamage().
- Add ScreenRec::SharedPixmapNotifyDamage().
    - Slave calls master's RequestSharedPixmapNotifyDamage().
    - Master calls slave's SharedPixmapNotifyDamage() when there is damage.
- Optional: if master does not provide RequestSharedPixmapNotifyDamage(), slave can fall back to vblank events.

PRIME Output Slaving: Synchronization

Control Flow Diagram (page 1 of 2)



(a) X server allocates 2 PRIME GEM buffers.

(b) X server calls master's SharePixmapBacking() => exports PRIME GEM buffers for slave.

(c) X server calls slave's rrEnableSharedPixmapFlipping() => gives slave chance for bookkeeping.

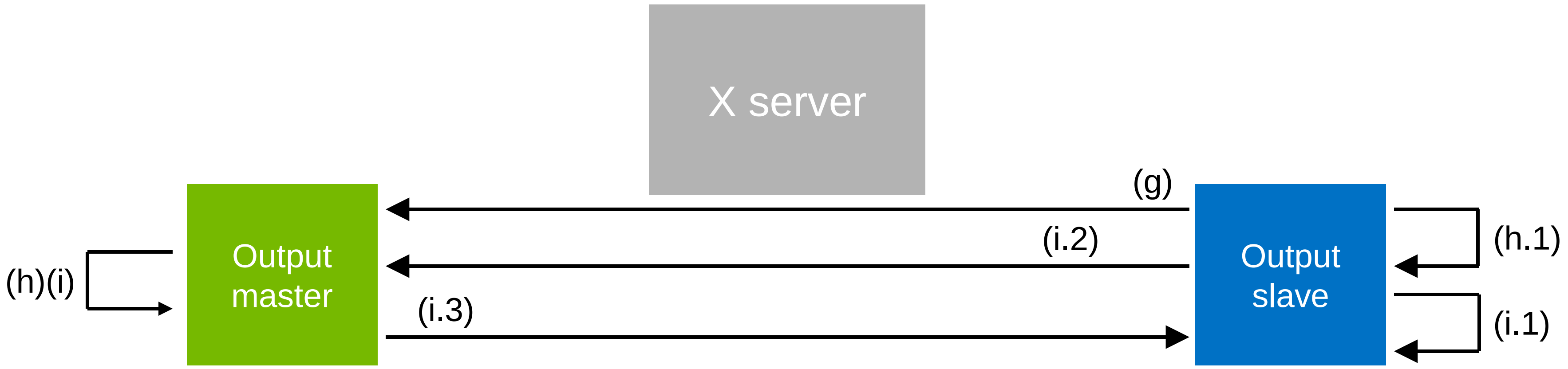
(d) X server calls master's rrStartFlippingPixmapTracking() => gives master chance for bookkeeping.

(e) X server calls master's PresentSharedPixmap() => populates front buffer.

(f) X server calls slave to perform modeset, which will display the front pixmap registered in (c); request vblank event.

PRIME Output Slaving: Synchronization

Control Flow Diagram (page 2 of 2)



(g) Slave's vblank handler triggers; slave calls master's PresentSharedPixmap().

(h) If master has new content, update new buffer and return TRUE:

(h.1) This causes slave to flip to new buffer, requesting vblank event; goto (g).

(i) Else, master does not have new content, return FALSE:

(i.1) If master does not provide RequestSharedPixmapNotifyDamage(), schedule vblank event; goto (g).

(i.2) Else, master provides RequestSharedPixmapNotifyDamage(), call it.

(i.3) When master receives damage, master calls slave's SharedPixmapNotifyDamage(); goto (g).

PRIME Output Slaving: Synchronization

Step #4: Output Slave Fenced Flipping

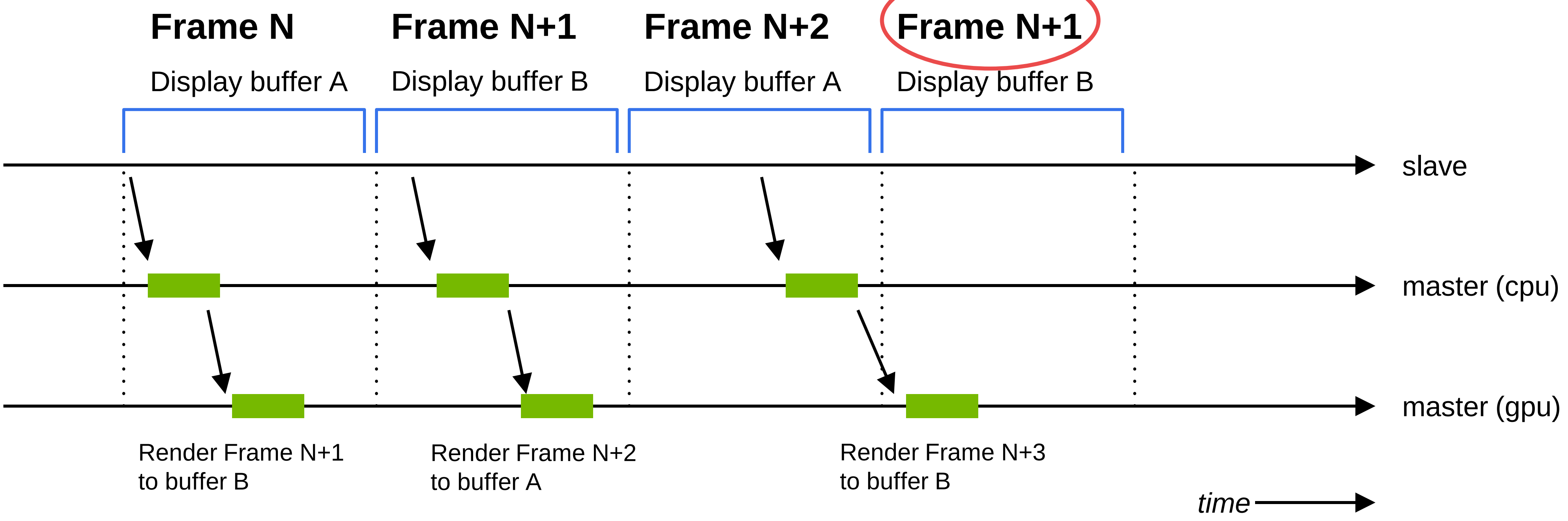
- Output master's PresentSharedPixmap():
  - Kick off work on the GPU, to write to shared buffer.
  - Doesn't necessarily wait for GPU work to complete.

- Output slave's subsequent flip could race ahead of master's GPU work.
  - Results in output slave displaying two-frame-old content.

PRIME Output Slaving: Synchronization

Step #4: Output Slave Fenced Flipping (continued)

Buffer B will not contain Frame N+3 content until the master (gpu) work completes.



 NVIDIA

PRIME Output Slaving: Synchronization

Step #4: Output Slave Fenced Flipping (continued)

- Use fence to block flip until until master's work completes.
- Master attaches fence before updating buffer, then signals fence after GPU completes.
- i915 kernel driver updated to honor fences when flipping (Linux 4.5).

PRIME Output Slaving: Synchronization

OpenGL Syncing To VBlank

- Master-rendered synced-to-VBlank OpenGL, needs to throttle to slave's vblank.
- In NVIDIA's PresentSharedPixmap() implementation:
  - Copying from X screen to GEM object is treated as virtual vblank by OpenGL.
  - Very implementation-dependent, but good for other output masters to be aware of.
- X server passes RRCrtcPtr to master's StartFlippingPixmapTracking():
  - This lets OpenGL correlate an RandR output name with the PRIME output slaving.
  - E.g., `__GL_SYNC_DISPLAY_DEVICE=HDMI-0 glxgears` behaves as expected (where "HDMI-0" is a slave output).

PRIME Output Slaving: Synchronization

Status

xf86-video-modesetting:

- Supports being a PRIME sync output slave.
- Supports being a PRIME sync output master.

nvidia X driver:

- Supports being a PRIME sync output master.

PRIME Synchronization enabled by default when X server and both drivers support it.

- Can be disabled by setting the "PRIME Synchronization" output property to 0.

PRIME Output Slaving: Synchronization

Conclusion and Future Work

- xf86-video-modesetting *should* work on any DRM KMS driver.
  - But only really tested on i915, so far.
- Fenced flipping should be implemented in other DRM drivers.
  - No way to detect in user-space if the kernel driver supports fenced flips.
- USB devices don't provide reliable vblank events:
  - xf86-video-modesetting blacklists usb devices from PRIME sync.
- Reverse PRIME would require more work:
  - GPUs that cannot scan out from sysmem would require copying from GEM shared buffer to shadow vidmem.
  - Honor fence during sysmem to vidmem copy.

PRIME Output Slaving: Synchronization

Thank You

- Thank you to Alex Goins (agoins 'at' nvidia.com) for all the work on this.
  - Contact Alex for questions.
- End-user focused documentation here:
  - https://devtalk.nvidia.com/default/topic/957814/linux/prime-and-prime-synchronization/