Improving Frame Timing Accuracy
X, DRM and Mesa

Keith Packard
keithp.com
Valve

Introduction

- What do we want?
  - Every frame displayed precisely when the application expects it.
  - “Fast enough” frame rate.
- Why is this hard?
  - Lots of moving parts:
    - application scene changes
    - compositing environment changes
    - power/thermal management
  - Asynchronous processing
    - Applications queue rendering to GPU
    - Display must wait for GPU completion

Rate Limiting

- Keep apps from getting too far ahead
  - Avoid long delays when apps crash
  - Reduce resource consumption
  - Reduce lag
- “Buffer Back Pressure”
  - Allocate limited # of buffers
  - Block waiting for free buffer before drawing
- # of Buffers Varies
  - Depends on window system status
- Need a More Consistent Technique

Vblank Events

- Allow apps to know when VBlank happens
- RegisterDisplayEvent(VK_DISPLAY_EVENT_TYPE_FIRST_PIXEL_OUT_EXT)
- Fence signaled at next Vblank
- Only works on direct Display targets
- Only works until you drop a frame
- Mixes Display action with Fences.

Wait for Present

- Allow apps to know when Present happens
  - Directly throttle presentations
- Block thread waiting for specific present
  - No callbacks, no events
- Doesn't use fences in API
  - Much easier to implement
- Uses application-provided presentation ID
  - Wait – display timing needs one of those too!

Accurate Display Timing

- Tell apps when vblank will be before rendering starts
- Allow apps to specify when frames should be displayed
- Get frames displayed on time
- Tell apps when frames were displayed
  - And when rendering was complete, in the same time domain

OpenGL

- GLX_OML_sync_control
  - Specify target present frame count
  - Avoids early frame presentation
- But, no feedback about when frames were actually presented
  - Many kludges required to guess
- GLX_EXT_swap_control
  - Sets (min) number of frames per presentation
  - No feedback on actual presentation time.

Current Vulkan APIs

- GOOGLE_display_timing
  - Specify absolute (CLOCK_MONOTONIC) time for frame
  - Feedback about when frames were presented
    - May be delayed by a long time (but not with Mesa).
- EXT_calibrated_timestamps
  - Get GPU/OS clocks values for the “same time”
  - Allows conversion between GPU and OS time domains

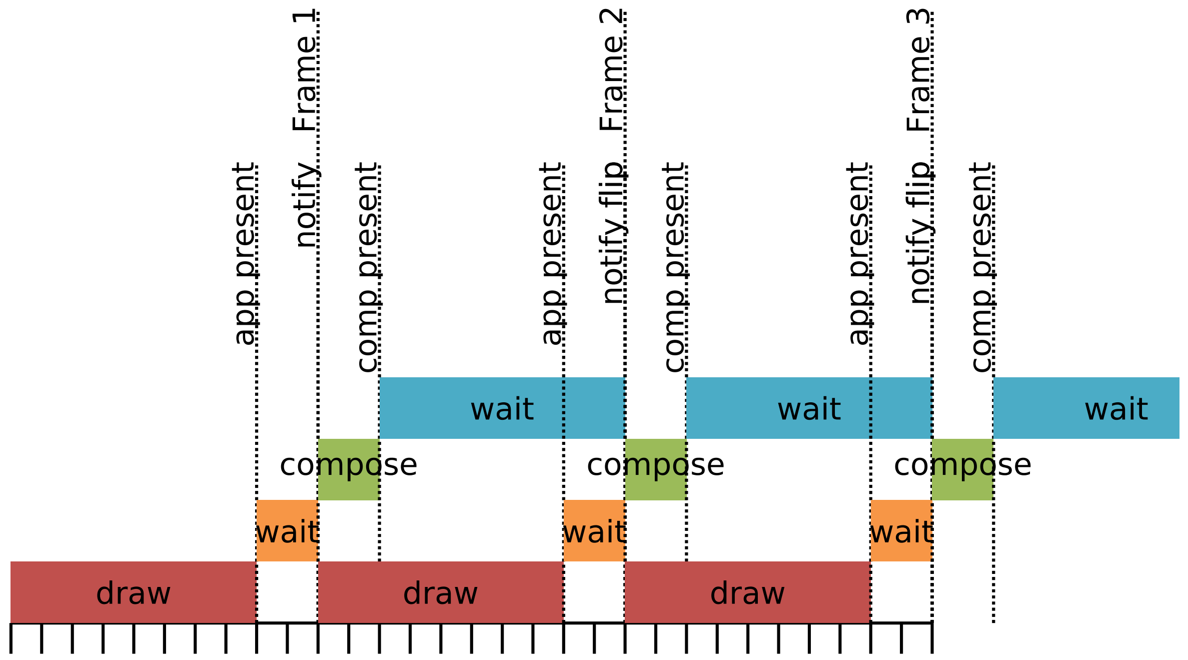
Upcoming Vulkan Ideas

- Improve display timing
  - Deal with variable rate displays
  - Provide "display for at least this long" semantics
  - Find something better than "not before"
    - Clock skew and/or precision issues
    - But hardware can't do "nearest"
  - Split out presentation ID to new extension
    - ID shared with wait-for-present extension

X

- Present extension spec is ready
  - Specify target frame for PresentPixmap
  - Provides feedback on when PresentPixmap was processed
- But the implementation lags
  - When the desktop is composited

Current X Composited



Current X Compositing Process

- Each app rendering request generates damage events to compositor
- Compositor collects damage
- At 'suitable time', compositor draws and calls PresentPixmap

Let X Composite Sometimes

- Compositor tells X which windows it can handle
- X server composites them when possible
- Eventual goal:
  - Share DRM compositor layer between window systems

Linux Flip API

- Current API is awkward
  - Finite event limit in kernel mixes flips and vblank notifies
  - Applications must work-around in user space
    - Test for failure, attempt to empty pending events, retry
  - Times in µS instead of nS
    - Doesn't match Vulkan time precision
- Single queue spot
  - Queue other buffers in user space
- No 'unqueue'
  - Commit to planned frame up front
- Blocks waiting for rendering(?)
  - The non-atomic path does
  - And I think the atomic does as well.
- Cannot actually support “Mailbox” mode.

Queue without blocking

- Kernel can move to HW when rendering completes.
- Allow user space to continue.
- Alternative is to have user space take an event and delay queuing until then.

Multiple flips queued

- For same frame
  - Kernel picks last one ready at vblank
  - Idles (and notifies) when possible
- For future frames
  - Allow user space to go idle for longer

Cancel queued entries

- Useful when queued for many future frames
  - avoid displaying from terminated apps
- Necessary if we don't get multi-queue
  - Handle all of that from user space

Summary

- Extend Vulkan to provide more usable API
- Fix timing under composited X
- Enhance Linux flip API
  - Make flips more reliable
  - Support Mailbox mode
  - Provide ns resolution

Thanks!

Keith Packard
keithp@keithp.com
Valve

VALVE®