



Vulkan 101

Tom Olson
Director, Graphics Research, ARM
Chair, Vulkan Working Group

What is Vulkan?

- **A 3D graphics API for the next 20 years**
 - Logical successor to OpenGL / OpenGL ES
 - Modern, efficient design
 - An open, industry-controlled standard
- **Here, now**
 - Released in February 2016
 - Available today for Windows / Linux
 - Shipping in Samsung Galaxy S7
 - Support announced in Android 'N'
- **Different!**
 - Fundamental change in philosophy
 - Requires corresponding changes in applications



Why did we do this?

- Traditional APIs had issues...
- Developers weren't happy

OpenGL Is Broken

May 30, 2014 by Joshua Barczak

The opinions expressed in this post are my own and are not shared, or sanctioned by anybody in particular (especially Khronos).

Rich Geldrich has a lot to say about this subject, and here's his list. The present state of OpenGL is incredibly

<http://www.joshbarczak.com/blog/?p=154>

Rich Geldreich's Tech Blog

Game and open source developer, graphics programmer

Sunday, May 11, 2014

Things that drive me nuts about OpenGL

Here's a brain dump of the things that sometimes drive me crazy about OpenGL and my coworkers. I'm also in a ranty-type mood today after grappling with the fact that the GL API needs a reboot because IMO Mantle/D3D12 are going to most likely replace it in the near future.

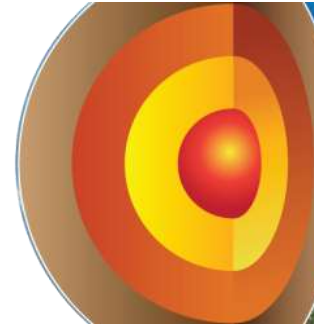
<http://richg42.blogspot.com/2014/05/things-that-drive-me-nuts-about-opengl.html>

Problems with OpenGL / OpenGL ES

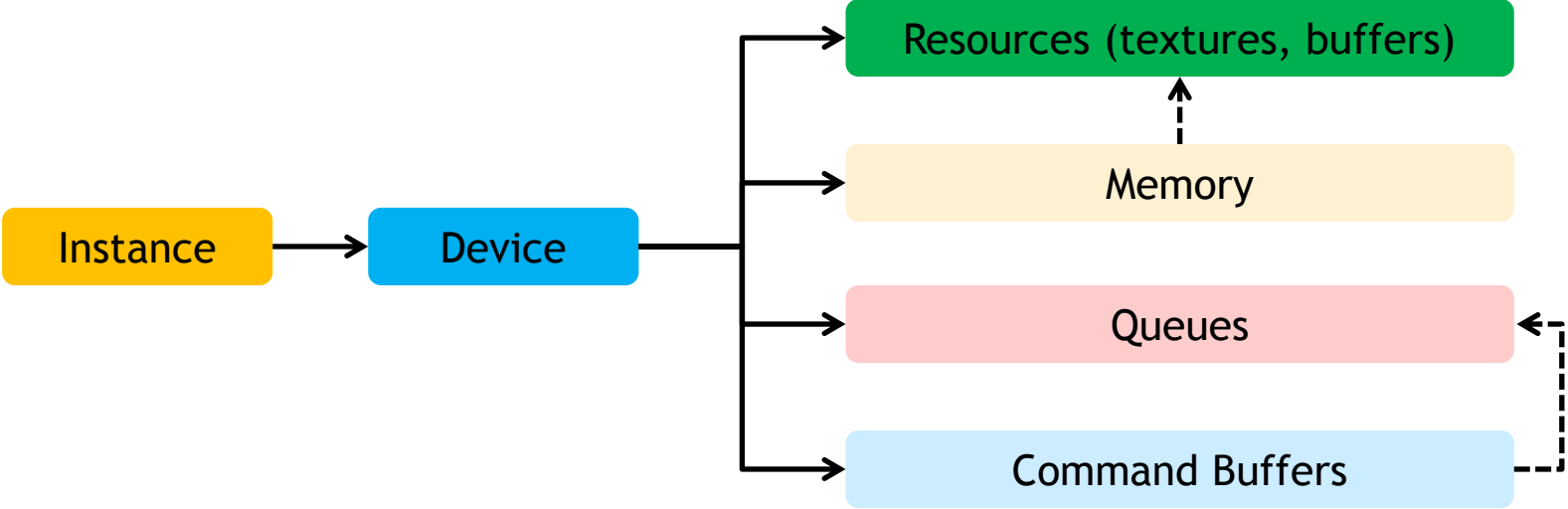
- **Programming model doesn't match GPU HW**
 - Especially in mobile
 - Driver magic hides the mismatch
- **CPU intensive**
 - Lots of state validation, dependency tracking
- **Complex, buggy, unpredictable drivers**
 - Different bugs and fast-paths on every GPU
- **Fundamentally single-threaded**
 - Can't use multi-core CPUs effectively
- **...not to mention twenty years of legacy cruft**

Enter Vulkan...

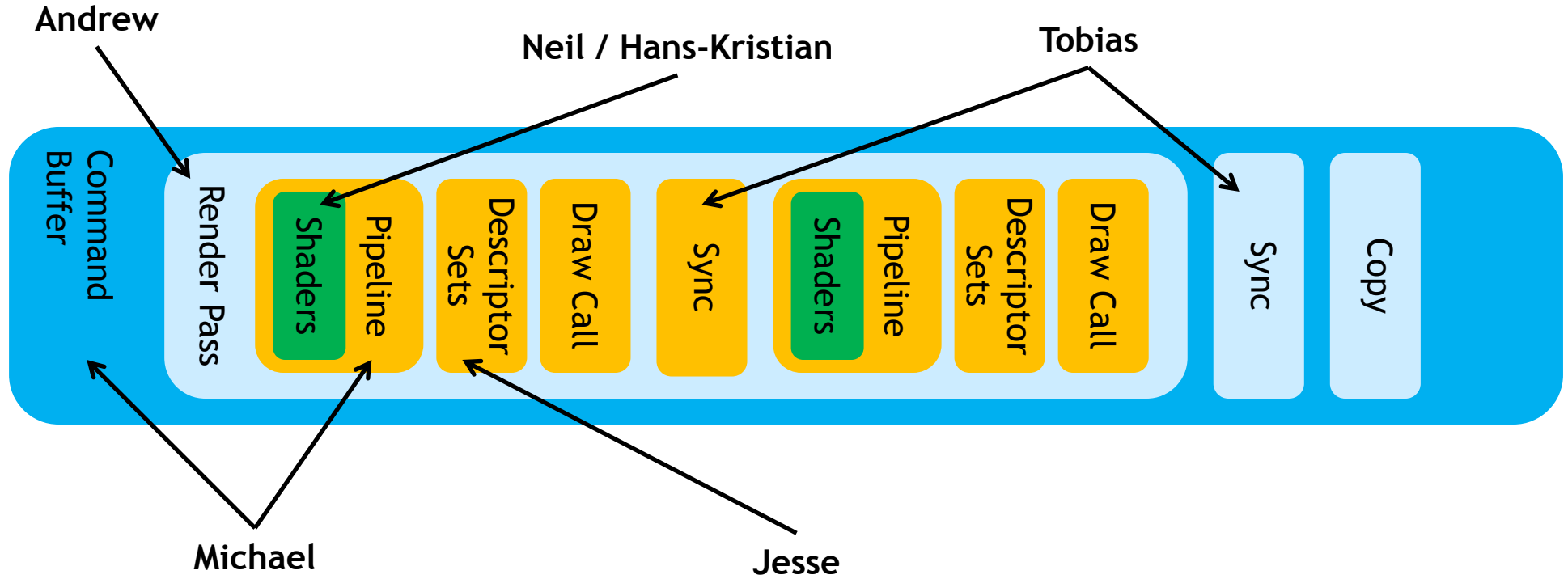
- Design discussions start in October 2012
- Moves into high gear in July/August 2014
 - Commitment from key ISVs
 - AMD donation of Mantle
- A lot of very hard work follows...
- Release to public in February 2016
 - Conformant drivers from four IHVs
 - GLSL to SPIR-V compiler
 - Debug and validation tools



Vulkan in one slide



Vulkan in ~~one slide~~ two slides



The principle of *Explicit Control*

- You promise to tell the driver
 - *What* you are going to do
 - *In sufficient detail* that it doesn't have to guess
 - *When* the driver needs to know it
- In return, driver promises to do
 - *What* you asked for
 - *When* you asked for it
 - *Very quickly*
- *No driver magic!*

OpenGL lets you specify important information very late, and change it at any time. It's convenient, but has huge performance costs.

OpenGL drivers often defer work until later, move it to another thread, or even ignore your commands, based on guesses about your intent. Vulkan drivers won't.

Loader, layers, and extensions

- Vulkan has no dependencies on external APIs
 - ICD loader is built-in
 - Window system binding is (semi) built-in
- A side benefit: Layers
 - Loader can install intercept libraries (“layers”)
 - E.g. trace, debug
- Extensions
 - Must be enabled at initialization time

Multithreading

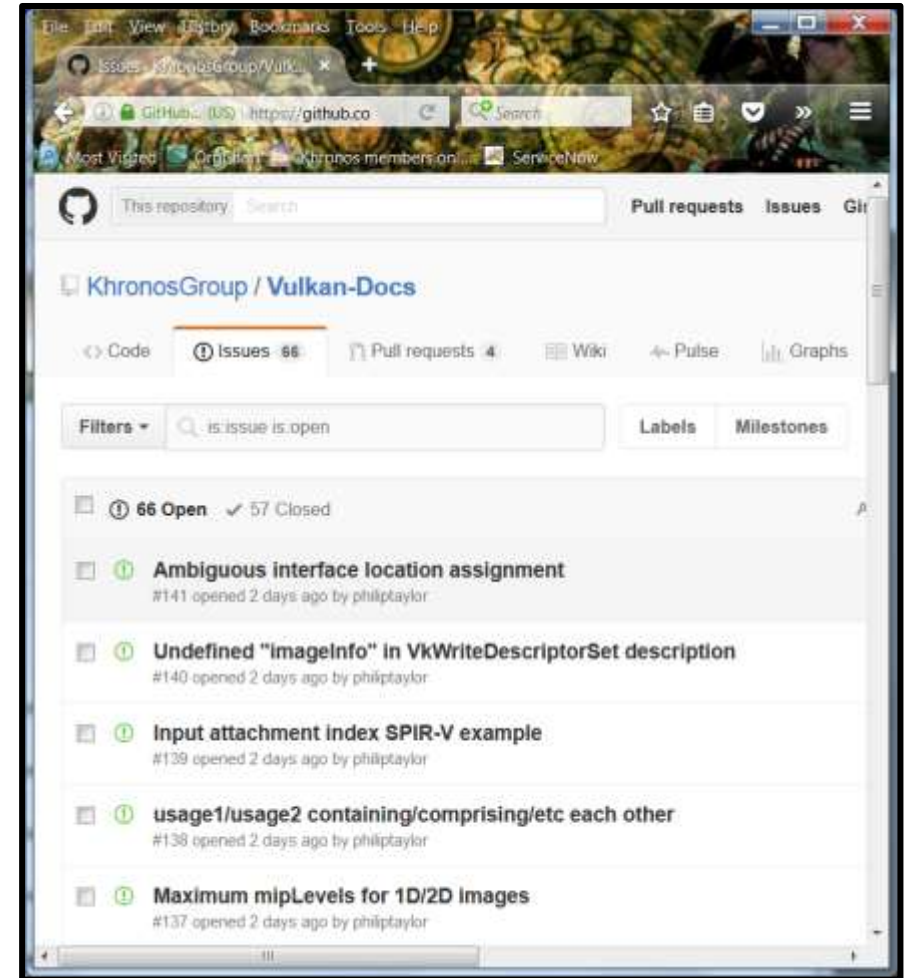
- All objects visible / accessible to all threads
- Most operations are *externally synchronized*
 - Application must prevent unsafe concurrent access
 - E.g., recording to the same command buffer
 - E.g., submitting to the same queue
 - Application must manage object lifetimes
 - Note, many objects are immutable
 - Concurrent read access is OK
- Allocation / creation *are* internally synchronized and may block
 - Per-thread pool allocators keep this reasonably cheap

Error handling

- **Vulkan is optimized for correct applications**
 - Does not (generally) check for invalid usage
 - Does not track dependencies
 - Does not (generally) provide thread safety
 - Breaking the rules results in undefined behavior
- **Vulkan *does* check for errors you can't predict**
 - Out of memory
 - Device lost
 - Other system errors...
- **Layers to the rescue!**
 - Can enable *validation layers* during development

Community

- **A new attitude**
 - ISV member input drove key decisions
 - Consulted with hundreds of developers
- **Strong commitment to open source**
 - Loader
 - Validation and other layers
 - SPIR-V tools: compiler, validator, ...
 - Conformance tests
 - Specification
- **All at <https://github.com/KhronosGroup>**



Should you be using Vulkan?

- **Challenges**

- Verbose and complex
- Lots of exposed sharp edges
- Lots to learn

- **Opportunities**

- Much lower driver overhead
- ...which you can spread across multiple threads
- More predictable performance
- Mobile friendly

- **Realities**

- Ecosystem is still immature
- Will need to ship GL/DX versions for years to come

