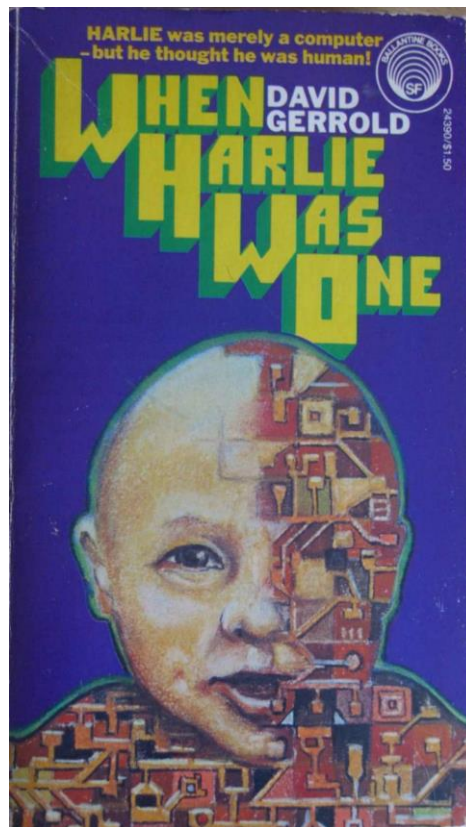




When Vulkan Was One Looking Back, Looking Forward

**Vulkan Working Group
and the Vulkan Community**

When Vulkan was One



<http://www.goodshowsir.co.uk/?p=7002>

HAPPY
BIRTHDAY.
TO YOU



Thanks to the community
for helping us reach this milestone!

Agenda

Vulkan's First Year

- Tom Olson (ARM)

Where Vulkan is Going

- Piers Daniell (NVIDIA)

Vulkan on the Move - Panel Discussion

- Hai Nguyen (Google)
- Karl Schultz (LunarG)
- Alen Ladavac (Croteam)
- Alon Or-bach (Samsung)
- Mike Schmit (AMD)
- Hans-Kristian Arntzen (ARM)



Vulkan's First Year

Tom Olson, ARM
Vulkan Working Group Chair
GDC 2017

A New Kind of API

Vulkan design goals

- Clean, modern architecture
- Portable across desktop and mobile
- Multi-thread / multi-core friendly
- *Efficient, predictable performance*

Key principles

- Explicit control - no driver magic
- Heavyweight SW functions moved to external tools



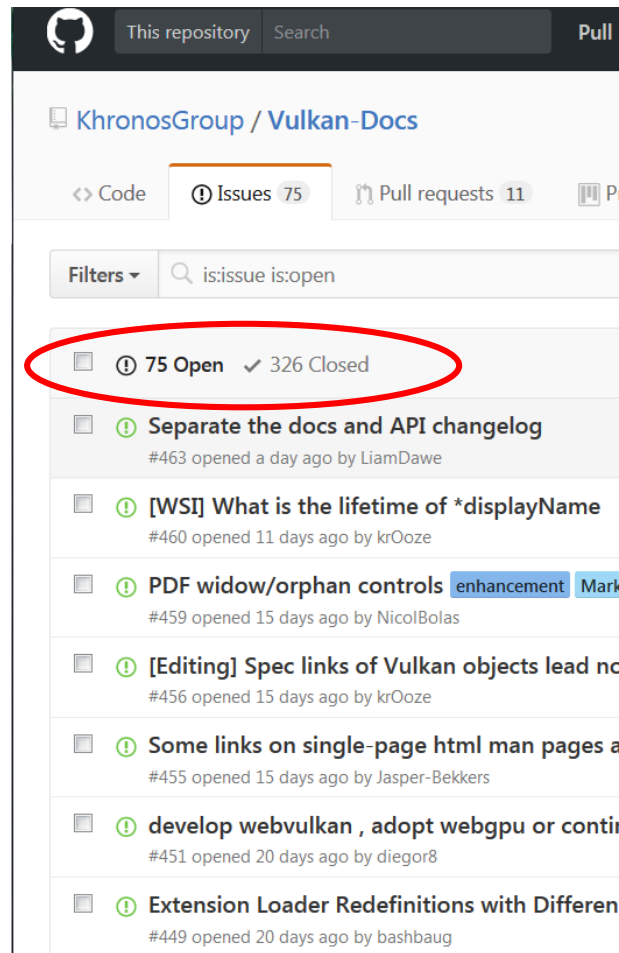
A New Philosophy

Community-facing and responsive

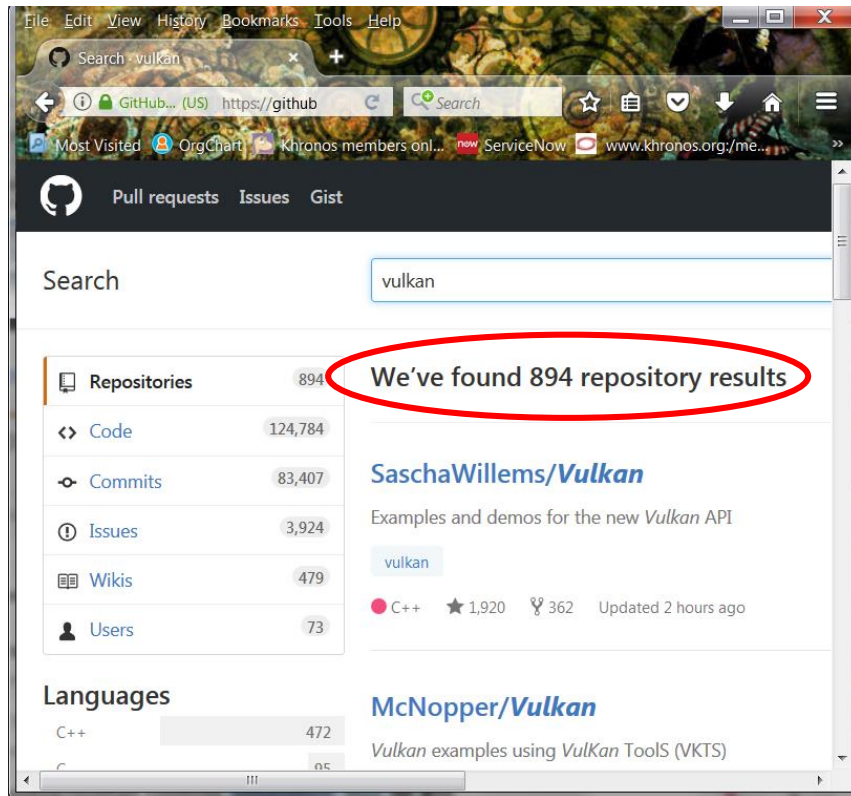
- Engage/partner with developer community
- Work in the open where we can
- Put as much as possible in open source
 - HLL shader compiler
 - SDK (ICD loader and validation layers)
 - Conformance test
 - Specification source

Commitment to complete releases

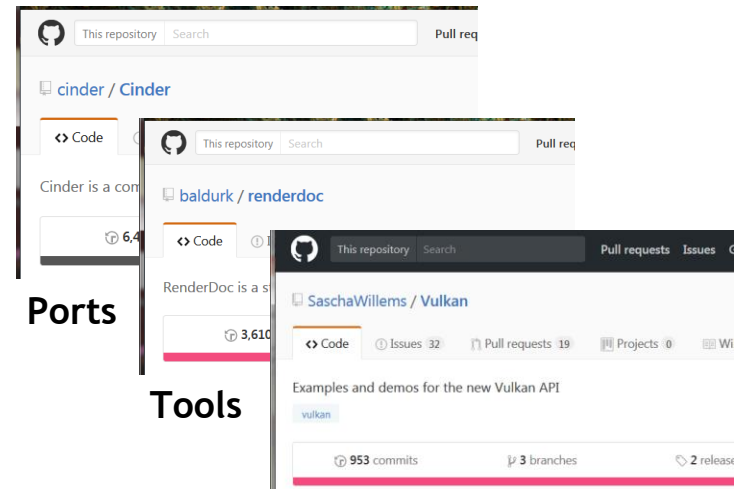
- Implementations available at release
- Compiler, loader, validation tools
- Conformance test



A Growing Community



A huge amount of activity
on GitHub



Tutorials and Examples

So how's it going?

Desktop

- Production drivers from all three leading GPU vendors
- Linux and Windows



Mobile

- Galaxy S7, NVIDIA Shield, Google Pixel/Nexus devices, Nintendo Switch
- Standard interface exposed in Android 7.0
- More coming every day!

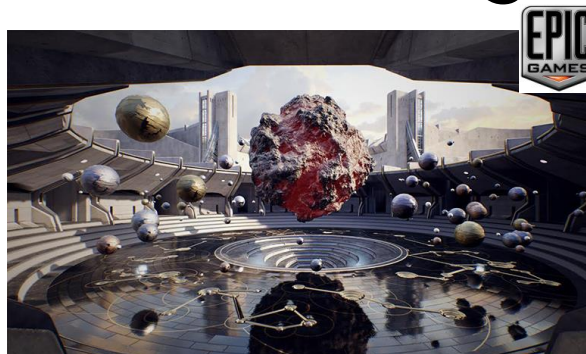


Nintendo Switch

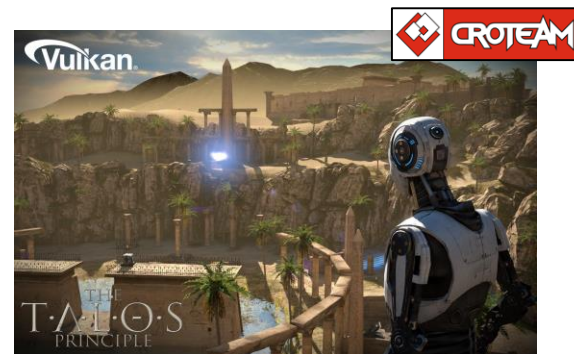
Vulkan Games and Game Engines



Dota 2 on Vulkan port of Source 2



'ProtoStar' demo on Vulkan port of Unreal Engine 4



Talos Principle on Vulkan port of Serious Engine



Developer Preview



Vulkan support in V1.8



Vulkan support coming

Lots more are coming



Game Studios publicly confirming that work is ongoing on Vulkan Titles

Does it work? Early indicators are promising...



DOOM Vulkan port – “30-40% increase in game performance”



ARM 'Lofoten' demo – “15% reduction in total system power”

The Bottom Line

- Vulkan momentum has exceeded our expectations
- Installed base is growing rapidly
 - Standard in desktop driver packages
 - Will be widely supported in Android 7.0
- There is an active and growing developer community
 - Khronos-supported tools at <https://github.com/KhronosGroup/>
 - ...but Khronos efforts are dwarfed by what the wider community is doing
- We hope you'll join them!



Where Vulkan is Going

Piers Daniell, NVIDIA
GDC, February 2017

Overview

- Features added since launch
- Features released for GDC
- Future directions



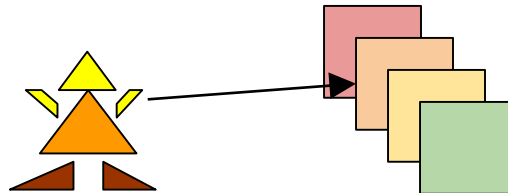
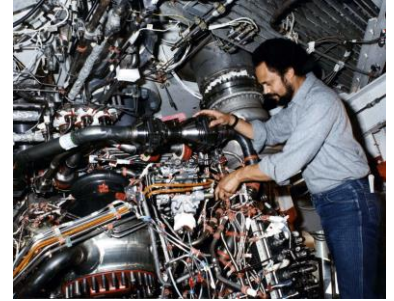
Feature Highlights since Launch

First-year Khronos Extensions

- **VK_KHR_get_physical_device_properties2**
 - Improves extensibility of Vulkan queries
 - Structs with pNext/sType added for core queries
 - Extensions can more easily add new feature, limit and format queries
 - Other extensions build on this
- **VK_KHR_shader_draw_parameters**
 - New shader variables
 - *DrawIndex* - iteration of draw from vkCmdDrawIndirect
 - *BaseVertex* - firstVertex/vertexOffset parameter of vkCmdDraw* commands
 - *BaseInstance* - firstInstance parameter of vkCmdDraw* commands

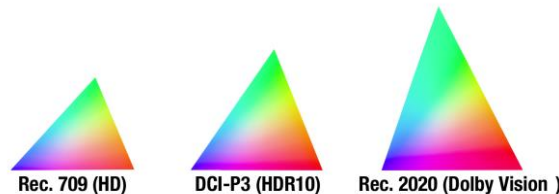
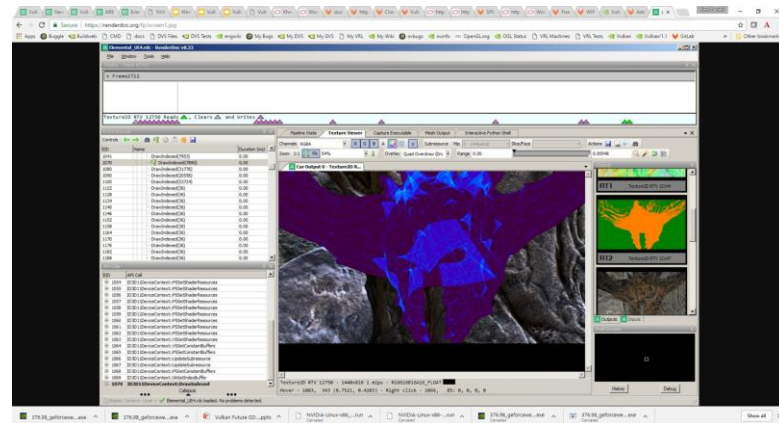
First Maintenance Extension

- VK_KHR_maintenance1
 - Based on feedback from developers like you
- Fixes some holes, oversights and bugs from Vulkan 1.0:
 - Allows vkCmdFillBuffer on transfer-only queue
 - Enables vkCmdCopyImage to copy between 2D and 3D images
 - Multi-creation API consistency fix
 - Allow staging-only images with FORMAT_FEATURE_TRANSFER_BIT usage
 - Support left-handed NDC space for porting content from other APIs
 - No more “gl_Position.y = -gl_Position.y” required
 - Allow apps to trim command pool memory back to system
 - Adds rendering to 3D slices



Multi-Vendor Extensions

- VK_EXT_debug_marker
- VK_EXT_debug_report
 - Graphics debugging support
- VK_EXT_direct_mode_display
- VK_EXT_display_control
 - Explicit control over display
- VK_EXT_display_surface_counter
 - Vertical blanking period counter support
- VK_EXT_shader_subgroup_ballot
- VK_EXT_shader_subgroup_vote
 - Multi-vendor subgroup operations
- VK_EXT_swapchain_colorspace
 - HDR presentation



New Features Released at GDC

About KHX Extensions



What they are

- Prototypes for future core features or KHR extensions
- A way to signal what's coming
- A way to get community input before specs become final

Basis for study, experimentation and feedback

- Help us make Vulkan better, sooner!

NOT recommended for use in production code

- Unstable - we plan to rev specs rapidly based on feedback
- Implementations are not conformance tested
- Will be deprecated at some point

Happy 1st Birthday Extensions

- Set of extensions released for GDC
- Sharing extensions
 - VK_KHX_external_memory*
 - VK_KHX_external_semaphore*
 - VK_KHX_win32_keyed_mutex
- Multi-GPU extensions
 - VK_KHX_device_group
 - VK_KHX_device_group_creation
- VR extensions
 - VK_KHX_multiview
- Descriptor update extensions
 - VK_KHR_push_descriptor
 - VK_KHR_descriptor_update_template



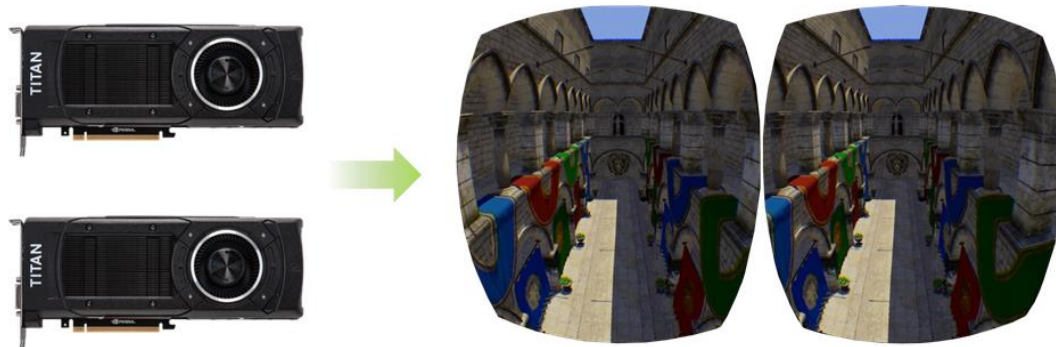
Vulkan Sharing and Interop

- **Memory sharing cross-process and cross-API:**
 - VK_KHR_external_memory
 - VK_KHR_external_memory_capabilities
 - VK_KHR_external_memory_fd
 - VK_KHR_external_memory_win32
- **Semaphore sharing cross-process and cross-API:**
 - VK_KHR_external_semaphore
 - VK_KHR_external_semaphore_capabilities
 - VK_KHR_external_semaphore_fd
 - VK_KHR_external_semaphore_win32
- **DX11 sharing:**
 - VK_KHR_win32_keyed_mutex



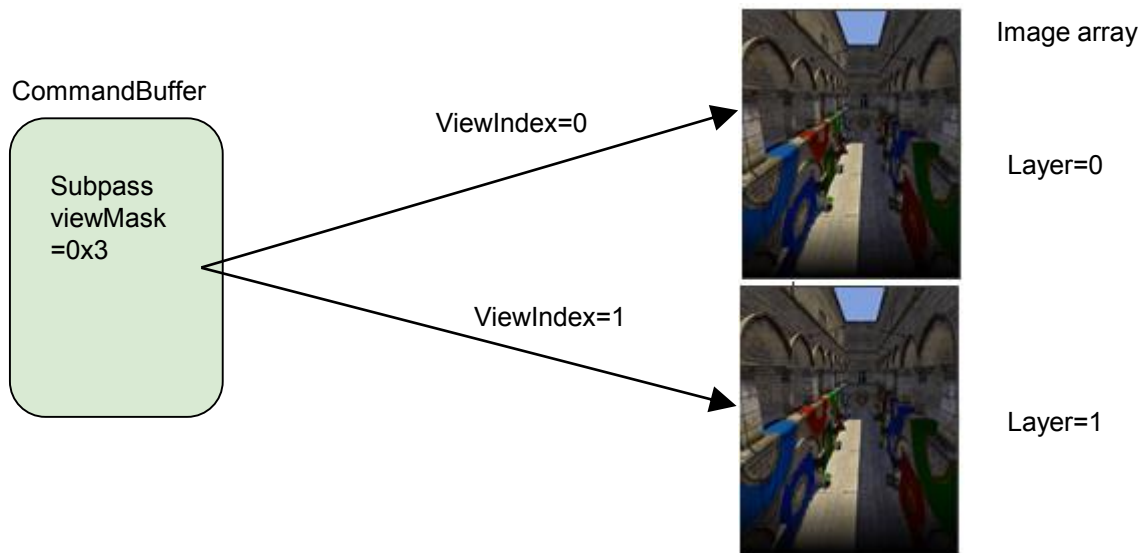
Homogenous multi-device rendering

- Native multi-GPU support for NVIDIA SLI and AMD Crossfire platforms
- Supports explicit AFR, SFR and VR rendering algorithms
- See Jeff Bolz's Vulkan Multi-GPU talk earlier today



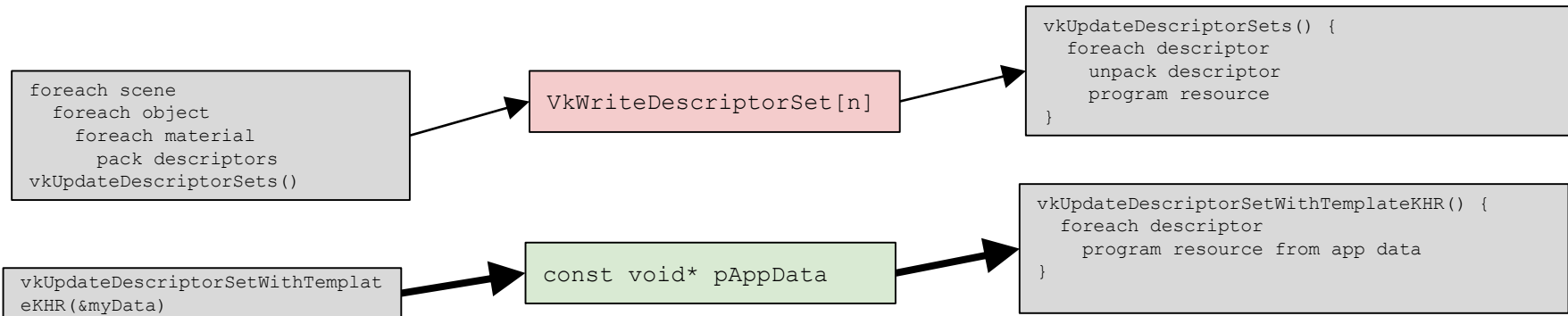
VK_KHR_multiview

- Render to multiple views from a single command buffer
- Each subpass can render to a subset of total views
- Each view is rendered to a separate layer
- Shader variable *ViewIndex* identifies the view being rendered
- Acceleration for stereo rendering and cube face rendering



Descriptor Update Extensions

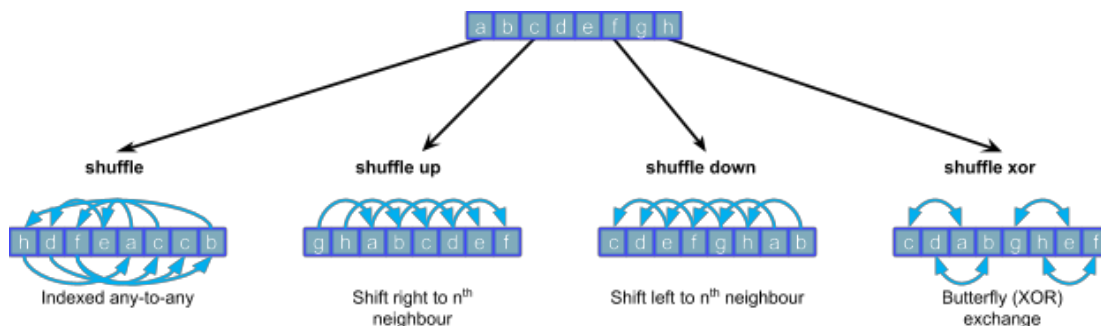
- **VK_KHR_push_descriptor**
 - Descriptor updates recorded inline in command buffer
 - Allows easier code porting from other APIs
 - Better performance in some cases
- **VK_KHR_descriptor_update_template**
 - Allows update of a fixed set of descriptor from pointer to app data
 - Bypasses intermediate descriptor update structure packing/unpacking
 - Can be used with push descriptors



Future Directions

Future Directions: Shader Functionality

- Advanced compute
 - Enable richer compute algorithms
 - Improved performance for capable hardware
 - More tightly defined behavior
- Improved HLSL support
 - cbuffer layout and offset support
- Subgroup operations
 - Vote, broadcast, **shuffle**, quad and arithmetic operations



Future Directions: WSI

- **Shared presentable images**
 - Allows application to update image while in use by presentation engine
- **Allow partial updates of presentable image**
 - Avoids wasting time updating areas which haven't changed
- **Add dedicated memory option for individual buffers and images**
 - Enables easier sharing and interop
- **Windows Full-screen ownership**
 - Allow Vulkan to take control and lose control of full screen display
- **Windows Memory residency control**
 - Allow apps to be a better citizen when memory resources are under pressure
- **VK_KHR_get_surface_capabilities2 / VK_KHR_get_display_properties2**
 - Extensible surface and display caps queries

Future Directions: Other

- **Maintenance2:**
 - View compressed image formats as integers
 - CreateRenderPass2 to allow better extendability and fix missing aspect information
 - Point/line clipping behavior query
 - Fix tessellation winding order inconsistency
- **Add ARB_texture_filter_minmax functionality**
 - To be exposed on supported devices

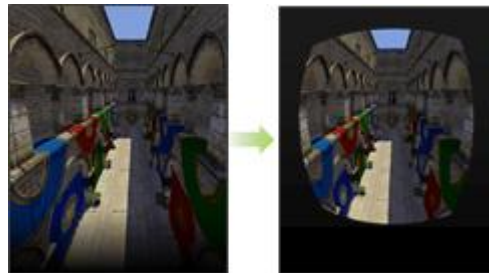
Vendor Extensions

- Vulkan is fully extensible
- GPU vendors have been innovating unique features
- AMD, Imagination and NVIDIA have added extensions so far
- Update on NVIDIA vendor extensions:
 - Friday 10am, Room 2000

Stereo rendering



Lens distortion



The Future of Vulkan

- Depends on you!
- Vulkan is open
 - File issues in GitHub
 - Send us your suggestions and wishlists
 - Contribute

Thanks!



Vulkan Survey

Informal Survey of Discussions

Survey Information

- **Sources**

- r/vulkan
- GameDev OpenGL and Vulkan Forum
- Khronos Vulkan Forum

- **Time Span**

- September 2016 to Present

- **Categorized by High Level Topics**

- Compute, Descriptor Sets, Driver, Synchronization, etc.

r/vulkan

- **134 Posts Total**

- Drivers (8%)
- Shaders (4%)
- Memory / Swapchain (3%)

- **Notes**

- Announcements are shared and discussed
- Decent split between developers and users

GameDev OpenGL and Vulkan Forum

- **33 Posts Total**

- Shaders (12%)
- Memory / Formats (9%)
- Descriptor Sets / Drivers / Queues / Swapchain (6%)

- **Notes**

- Announcements have separate forum
- Developer heavy

Khronos Vulkan Forum

- **97 Posts Total**

- Memory Handling (9%)
- Compute / Shaders (7%)
- Descriptor Sets / Pipeline / Render Targets / Swapchain (6%)

- **Notes**

- Announcements have separate forum
- Developer heavy

Tidbits

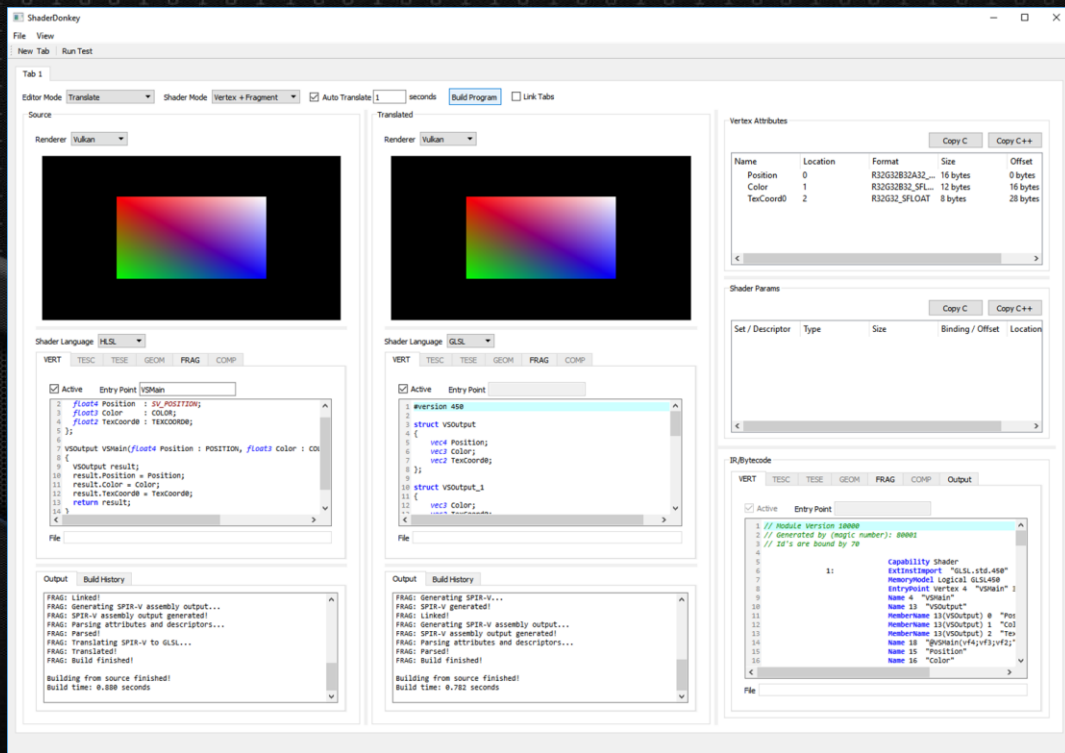
- **Things that still trip people up**
 - Memory
 - Descriptor Sets
 - Queues
- **Most referenced samples**
 - Sascha Willem's samples
- **Center of discussions**
 - How to do Vulkan correctly
 - Not many discussions on performance or multithreaded rendering

tinyrenderers

- **Single header implementation of Vulkan and D3D12 renderers**
- **Features**
 - MSAA
 - Basic buffer handling (index, uniform, vertex)
 - Mipmapping
- **Samples in Cinder and GLFW**
 - Very experimental TinyCI samples
 - Single header implementation of basic Cinder on Windows
- **Windows only for now**
 - Linux coming soon
- **github.com/chaoticbob/tinyrenderers**

ShaderDonkey

- Shader exploration tool
- Features
 - Compare HLSL to GLSL
 - Vulkan preview renderer
 - D3D12 preview renderer
 - Powered by
 - glslang
 - SPIRV-Cross
 - Descriptor set extraction
 - Vertex attributes extraction
- See it in demo area
- Will be open sourced soon



The background is a dark gray with a fine, grid-like texture. Overlaid on this are several abstract, flowing shapes in a slightly lighter shade of gray. A prominent diagonal band runs from the top left towards the bottom right. In the lower-left quadrant, there is a large, dark, oval shape with a thin blue outline. The word "Thanks" is centered in the upper half of the image in a large, white, sans-serif font.

Thanks

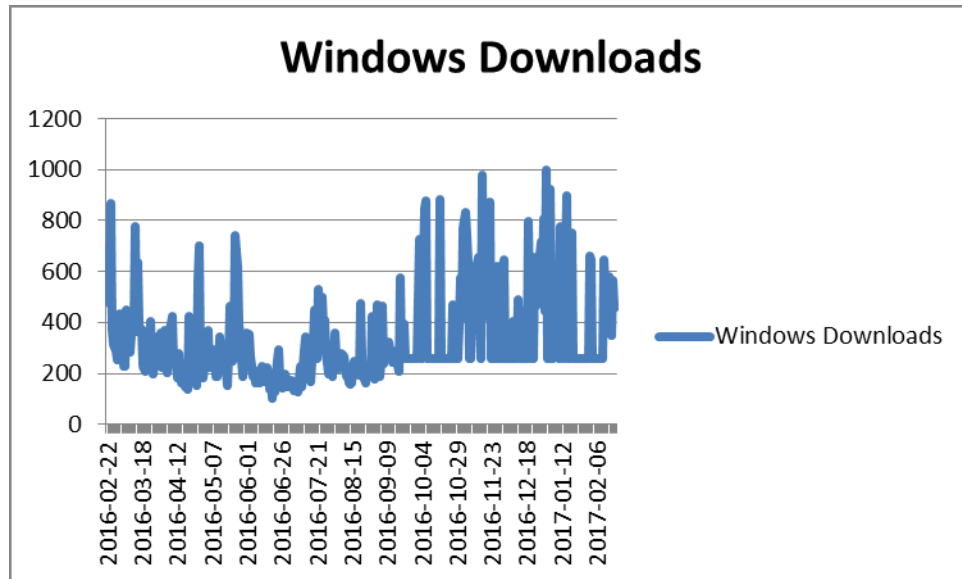
Hai Nguyen
haing@google.com

Vulkan Ecosystem Update

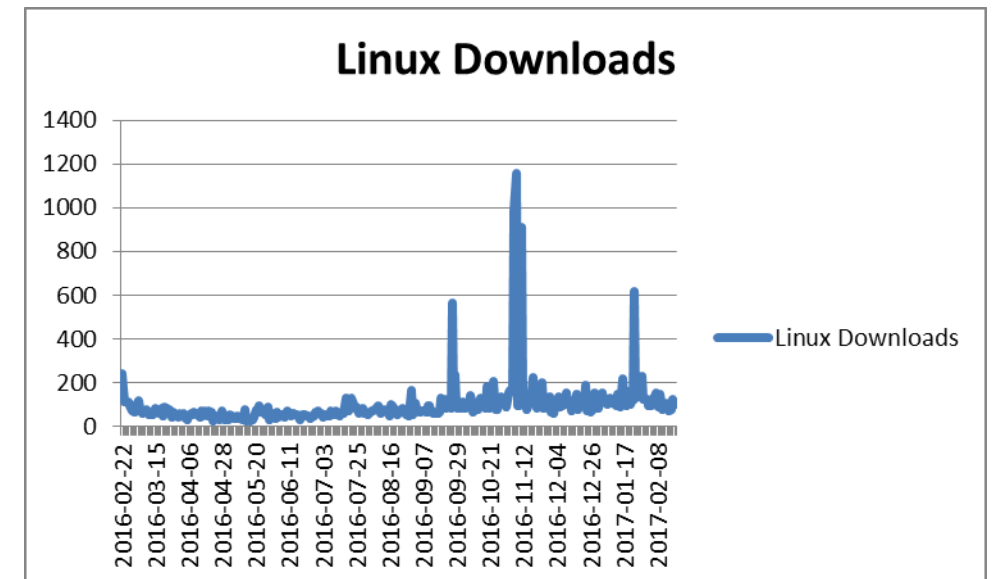
GDC February 2017

Karl Schultz, LunarG Inc

LunarG SDK Downloads



Windows
Total: 122,606



Linux
Total: 34,650

Validation Layer Status

- 2600 Valid Usage statements in the specification
- 1849 are implemented
- 71% complete
- More engineers working to finish up
- Please let us know if you find any that you'd like implemented sooner than later.
 - File an issue on GitHub: KhronosGroup / Vulkan-LoaderAndValidationLayers

Vulkan Tools

- GitHub: LunarG / VulkanTools
- vktrace
 - Improved stability and completeness
 - Records host writes to persistently mapped buffers

Vulkan Samples

- GitHub: LunarG / VulkanSamples
- More Samples
 - e.g., memory barriers
- Tutorial