Khronos Dev Day - The Vulkan Sessions

1:20pm - What’s New in Vulkan
   - Tom Olson (Arm), David Neto (Google), Dan Ginsburg (Valve)

2:40pm - HLSL in Vulkan: There and back again
   - Greg Fischer (LunarG), Matthäus Chajdas (AMD), Hai Nguyen (Google)

4:00pm - Vulkan on Android: Gotchas and best practices
   - Arseny Kapoulkine (Roblox), Frederic Garnier (Samsung)

5:30pm - Getting explicit: How hard is Vulkan really?
   - Dustin Land (id Software) + distinguished panel
Outline

Vulkan: The State of the Union
   Tom Olson (Arm), Vulkan WG chair

The Shading Language Toolchain
   David Neto (Google), SPIR/SPIR-V WG chair

The Vulkan Portability Initiative
   Dan Ginsburg (Valve)
Vulkan Today - Availability

Drivers
- Production drivers from all major GPU vendors
- Desktop and mobile
- Some in open source

Platforms
- Windows
- Linux
- Android
- macOS / iOS
Vulkan Today - Game Engines

The Gang’s All Here...

- Vulkan rendering paths in many of the leading engines
Vulkan Content is Shipping...

Vulkan-only AAA Titles on PC

- DOOM VFR
- Wolfenstein II: The New Colossus
- Rise of the Tomb Raider
- F1 2017

AAA titles on Linux

- Mad Max
- Dawn of War

Dota 2 on PC and macOS
...on Mobile and Desktop

Plus....
Lineage 2 Revolution
Heroes of Incredible Tales
Dream League Soccer...
Vulkan Ecosystem Momentum

LunarG Vulkan SDK
Download rate increases every year since launch
http://vulkan.lunarg.com

Vulkan GitHub Open Source Projects in August 2016
- We’ve found 431 repository results

Today
- 1,798 repository results
Developer Activity

RenderDoc Users by API
Please Welcome Vulkan 1.1!

- **Multiview (KHR_Multiview)**
  - Render to multiple image views simultaneously
  - E.g. stereo pairs, cube map faces

- **Cross-process/API sharing (KHR_external_*)**
  - Share memory and sync primitives across API / process boundaries

- **Advanced Compute Functionality (KHR_16bit_storage, KHR_variable_pointers)**
  - Read and write 16-bit quantities stored in GPU memory
  - Refer to data structures using a restricted form of pointers

- **YCbCr support (KHR_sampler_ycbcr_conversion)**
  - Sample the YCbCr color formatted textures produced by many video codecs
  - Useful for compositing video streams and mixing them with other graphical content
Please Welcome *more* Vulkan 1.1!

- **Device Groups (KHR_device_group*)**
  - Enable homogeneous multi-GPU systems
  - Number of GPUs in system ~transparent to the application

- **HLSL support (KHR_relaxed_block_layout)**
  - Match HLSL memory data layout constraints
  - Same HLSL shaders across Vulkan and DX

- **Maintenance (KHR_maintenance*, KHR_get_memory_requirements2, ...)**
  - Bug fixes, cleanup, improved extensibility, better ways to do things
New Functionality in Vulkan 1.1

Protected Content
- Restrict access or copying from resources used for rendering and display
- Secure playback and display of protected multimedia content

Subgroup Operations
- Efficient mechanisms that enable parallel shader invocations to communicate
- Wide variety of parallel computation models supported

Example Subgroup Operations
A subgroup is a set of invocations (tasks) running on a GPU Compute Unit
(Note many GPUs typically support subgroup sizes of 32/64 invocations)
Vulkan 1.1 is Shipping Today

- **Specifications:** For community use and feedback
- **Conformance Tests:** For responsive bug fixing and enhancements
- **Tools:** compilers, SPIR-V tools, LunarG SDK / loader / layers
- **All in open source - welcoming issues and PRs**

- Conformant Vulkan 1.1 drivers and devices:

  AMD  arm  Imagination
  Intel  NVIDIA  Qualcomm
  RADV  ANV

Vulkan 1.1 is exposed in Android P developer preview.
SDK: New Vulkan API Developer Tools

Vulkan Layer Factory (VLF)
- Rapid layer development through hiding implementation details

Device Simulation Layer
- Simulate target device capabilities without requiring actual hardware

Assistant Layer
- Highlight potential performance issues, questionable usage patterns, common mistakes, and general risky behavior

Delivered with the LunarG Vulkan SDK
https://vulkan.lunarg.com/

Source available in the LunarG Vulkan Tools repository
https://github.com/LunarG/VulkanTools
CTS: Conformance Test Coverage

Khronos’s #1 engineering project

- Raise issues and PRs at https://github.com/KhronosGroup/VK-GL-CTS
The Evolving Ecosystem: It’s Complicated...

Stuff we do
- Specifications (API, SPIR-V, GLSL)
- Conformance tests
- Validation layers

Stuff we’re involved in
- SDKs
- Compilers and SPIR-V tools

Stuff other people do
- Trace and debug tools (RenderDoc etc)
- Documentation, best practices, example code
- https://vulkan.gpuinfo.org/
- ....lots of other cool stuff

Tricky for developers and contributors to navigate...

a gross oversimplification…
Public Vulkan Ecosystem Forum

Structure and goals
- Open Forum - Hosted by Khronos, open to everyone
  - Include ecosystem leads, Vulkan WG, and app developers
  - Enable cross-functional problem solving

Examples of current projects
- Address stability issues with HLSL on Vulkan
- Help people navigate the compiler / toolchain landscape
Vulkan Ecosystem Forum: Getting in Touch

Primary vehicle for engagement: GitHub issue tracker
- https://github.com/KhronosGroup/Vulkan-Ecosystem
- Used to gather issue reports and drive meeting agendas

Email list: public_vulkan_ecosystem@khronos.org

Regular phone calls (every Thursday at 9AM Pacific Time)
- Primarily project management issues, no deep technical discussions
- Intent is for users to participate on an as-needed basis:
  • You filed an issue and we feel we need to understand the impact a bit better
  • You are interested in helping to solve a problem and can motivate others within your company to contribute
  • …
State of Shader Tooling for Vulkan

David Neto, Google
A tale of two open source compilers

- **Glslang**
  - [KhronosGroup/glslang](https://KhronosGroup/glslang)
    - Reference parser for GLSL, ESSL
    - De facto GLSL → SPIR-V compiler
    - Used by Vulkan conformance test suite
    - Vulkan programming model features land here first! **Subgroups**
    - First mature HLSL → SPIR-V compiler

- **DXC**
  - [Microsoft/DirectXShaderCompiler](https://Microsoft/DirectXShaderCompiler) for HLSL
    - Announced GDC 2016, source release Spring 2017
    - “Spiregg” backend
    - Reuse Microsoft’s parser, add **SPIR-V code generation**
HLSL compilers: Guidance

- Language coverage
  - Glslang: SM 4, 5.1 *
    - First with Vulkan features
  - DXC/Spiregg: SM 5.1, 6ish *
    - Uses Microsoft’s parser

- So far. Compilers are shaped by their workloads!

- Platforms
  - Glslang: Windows, Linux, OSX, Android
  - DXC/Spiregg: Windows

- Code size
  - Glslang: Smallish / embeddable
Back out to high level shader languages
A toolbox for shader transformation

SPIR-V-Tools
spirv-opt

SPIR-V

Vulkan
Thank you

- These are exciting times for Vulkan shader tooling!
  - Thanks to the work of many organizations and individuals

- Hear more about HLSL for Vulkan, and SPIR-V transforms later today in:
  “HLSL in Vulkan: There and Back Again”
Vulkan Portability Initiative

Dan Ginsburg, Valve
Pervasive Vulkan 1.0

Major GPU Companies supporting Vulkan for Desktop and Mobile Platforms

Platforms

Desktop

Mobile (Android 7.0+)

Media Players

Consoles

Virtual Reality

Cloud Services

Embedded

Game Engines

Valve

id

CryEngine

Unity

Epic Games

Croteam

Serious Engine

Xenko
What about Vulkan on Apple Platforms?
What about Vulkan on Apple Platforms?

YES
What about Vulkan on Apple Platforms?

YES*

* With some small caveats
Vulkan on Apple Platforms

- **MoltenVK open source:** [https://github.com/KhronosGroup/MoltenVK](https://github.com/KhronosGroup/MoltenVK)
  - macOS/iOS Vulkan on Metal
  - Apache 2.0 License

- **LunarG macOS SDK:** [https://vulkan.lunarg.com/](https://vulkan.lunarg.com/)
  - Loader
  - Validation Layers
  - Tools
Valve - Vulkan Dota 2 on macOS

Vulkan delivering up to 50% performance increase over native OpenGL
Dota 2 macOS OpenGL vs Vulkan
MoltenVK Overview

• Vulkan on Metal
  - Low overhead
  - macOS 10.11+
  - iOS 9+

• Shaders converted with SPIRV-Cross
  - SPIR-V -> MSL
  - MSL VkPipelineCache
  - Extension for providing Metal binaries
LunarG macOS SDK

- Vulkan loader
  - ICD is MoltenVK
- Validation layers as used on Windows/Linux
- Two methods of deployment
  - Framework for static linking
  - Dynamic linking including loader and MoltenVK
- Tools
  - Shader compilers (SPIRV-Tools, glslang)
  - vulkaninfo
- Documentation
Future: Vulkan over D3D12 and more

- Motivation for building a Vulkan over D3D12 implementation:
  - Windows 10 S and Polaris are limited to Universal Windows Platform applications (UWP), which exclude Vulkan and OpenGL native drivers
  - Even where Vulkan can be supported, it’s often not bundled with the system ... making it harder for Firefox Quantum to run fast
- Gfx-rs is developing D3D12, Metal, and OpenGL backends for Vulkan Portability

Khronos welcomes collaboration with any open source library developers that wish to implement the Vulkan portable subset
Draft Vulkan Portable Subset Definitions

- Metal and DX12 can support almost all of Vulkan - very little needs to be removed
  - Triangle fans are not supported
  - Separate stencil reference masks are not supported
  - Vulkan Event functionality is currently not supported
  - Support for only a limited set of texture-specific swizzles
  - Allocation callbacks in object creation functions will be ignored

Not all subset features are removed on all platforms. Supported features will typically be increased over time as underlying platform capabilities increase.
Vulkan Portability Initiative Roadmap

**Widened Platform Support**
Diverse open source run-times over additional backends
E.g. Mozilla helping to drive gfx-rs for Vulkan over DX12, OpenGL and Metal
https://github.com/gfx-rs/gfx
https://github.com/gfx-rs/portability

**Enhanced SDK Layers and Tools**
E.g. Simulation layer to flag use of features not present in a selected target system

**Coordinated Subset Management**
Cross-platform standardized subsetted functionality queries
Conformance tests tailored per target - what is present must work!

TODAY
Free, open source tools, SDKs and libraries to bring Vulkan 1.0 applications to Apple using Metal
Call to Action

- The Vulkan Portability Landing Page is here:  
  - [https://khr.io/vulkanpilandingpage](https://khr.io/vulkanpilandingpage)
- Download the free open source MoltenVK library  
  - [https://github.com/KhronosGroup/MoltenVK](https://github.com/KhronosGroup/MoltenVK)
- The free macOS SDK can be downloaded from [LunarXchange](https://www.lunarxchange.com)  
  - You can submit issues and feedback at the same website
- Let us know if you find functionality or performance issues!  
  - We appreciate feedback as we refine and improve this portability solution  
  - [https://khr.io/vulkanpifeedback](https://khr.io/vulkanpifeedback)