



**KHRONOS™**  
GROUP

**Vulkan 1.1**

**March 2018**

**EMBARGOED UNTIL**  
**Wednesday 7th March, 6AM PT**

# Vulkan 1.1 Launch and Ongoing Momentum

## Strengthening the Ecosystem

Improved developer tools (SDK, validation/debug layers)  
More rigorous conformance testing  
Shader toolchain improvements (size, speed, robustness)  
Shading language flexibility - HLSL and OpenCL C support  
Vulkan Public Ecosystem Forum



February 2016  
Vulkan 1.0

Explicit Access to  
GPU Acceleration

## Vulkan 1.0 Extensions

Maintenance updates plus additional functionality

Explicit Building Blocks for VR  
Explicit Building Blocks for Homogeneous Multi-GPU  
Enhanced Windows System Integration  
Increased Shader Language Flexibility  
Enhanced Cross-Process and Cross-API Sharing



March 2018  
Vulkan 1.1

Integration of Proven and  
New Technology into Core

## Building Vulkan's Future

Deliver complete ecosystem - not just specs  
Listen and prioritize developer needs  
Drive GPU technology

## Widening Platform Support

Pervasive GPU vendor driver availability  
Port Vulkan apps to macOS/iOS and DX12  
Open source drivers

Vulkan 1.1 specification launching  
March 7th with open source  
conformance tests and tools, and  
multiple vendor implementations!

# New Generation GPU APIs

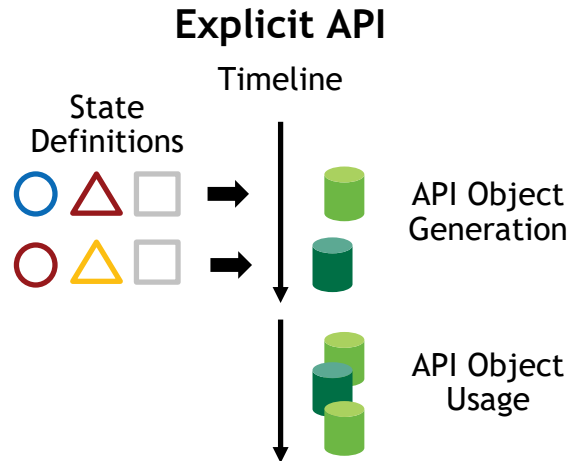
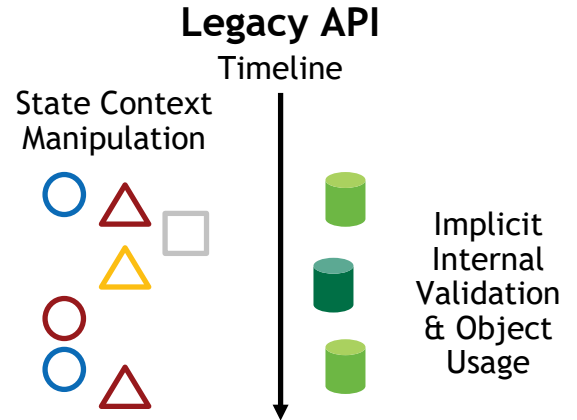
Non-proprietary, royalty-free open standard 'By the industry for the industry'  
Portable across multiple platforms - desktop and mobile  
Modern architecture | Low overhead | Multi-thread friendly  
EXPLICIT GPU access for EFFICIENT, LOW-LATENCY,  
PREDICTABLE performance



Vulkan is the primary platform 3D API on Android 7.0+

# Explicit GPU Access

- **Application tells the driver what it is going to do**
  - In enough detail that driver doesn't have to guess
  - When the driver needs to know it
- **In return, driver promises to do**
  - What the application asks for
  - When it asks for it
  - Very quickly
- **Predictable performance costs**
  - Creating pipelines, allocating memory, ...
- **No driver magic - no surprises - simpler drivers**
  - Remove guesswork and late decision-making
- **Putting control in the hands of developers**
  - Flexible scheduling of CPU and GPU workloads
  - Management of memory and synchronization



# Pervasive Vulkan 1.0



Major GPU Companies supporting Vulkan for Desktop and Mobile Platforms



## Platforms



Desktop



Mobile  
(Android 7.0+)



Media Players



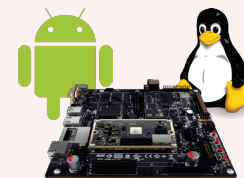
Consoles



Virtual Reality



Cloud Services

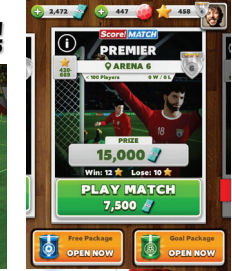


Embedded

## Game Engines



# Vulkan is Powering Mobile Gaming...



And more...  
 Lineage 2 Revolution  
 Heroes of Incredible Tales  
 Dream League Soccer...



# ... and Enabling Cross-Platform AAA Titles



Publicly announced games  
as of March 2018  
#Vulkan = 34  
#DX12 = 27  
[https://en.wikipedia.org/wiki/List\\_of\\_games\\_with\\_Vulkan\\_support](https://en.wikipedia.org/wiki/List_of_games_with_Vulkan_support)

Vulkan-only AAA  
Titles on PC



Dota 2 on PC  
and macOS



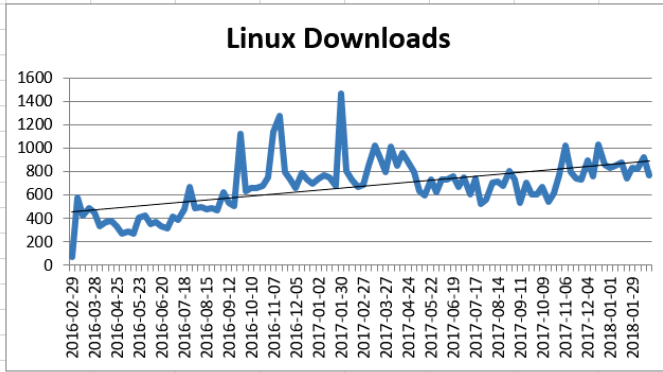
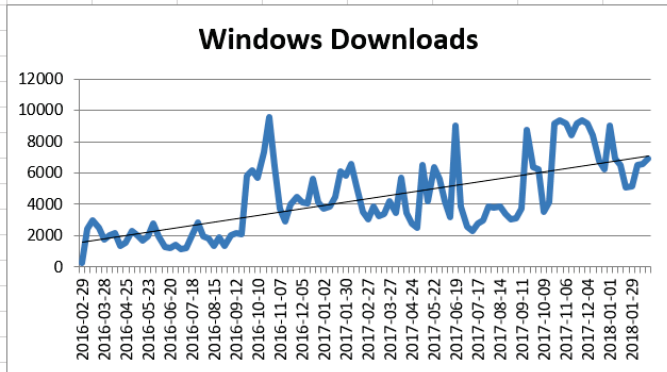
AAA on Linux



# Vulkan Ecosystem Momentum



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



## Vulkan GitHub Open Source Projects end of 2016

Repositories	431
Code	37,998
Issues	2,054
Users	43

SaschaWillems/**Vulkan**  
Examples and demos for the new **Vulkan** API  
Updated 2 days ago

## Today

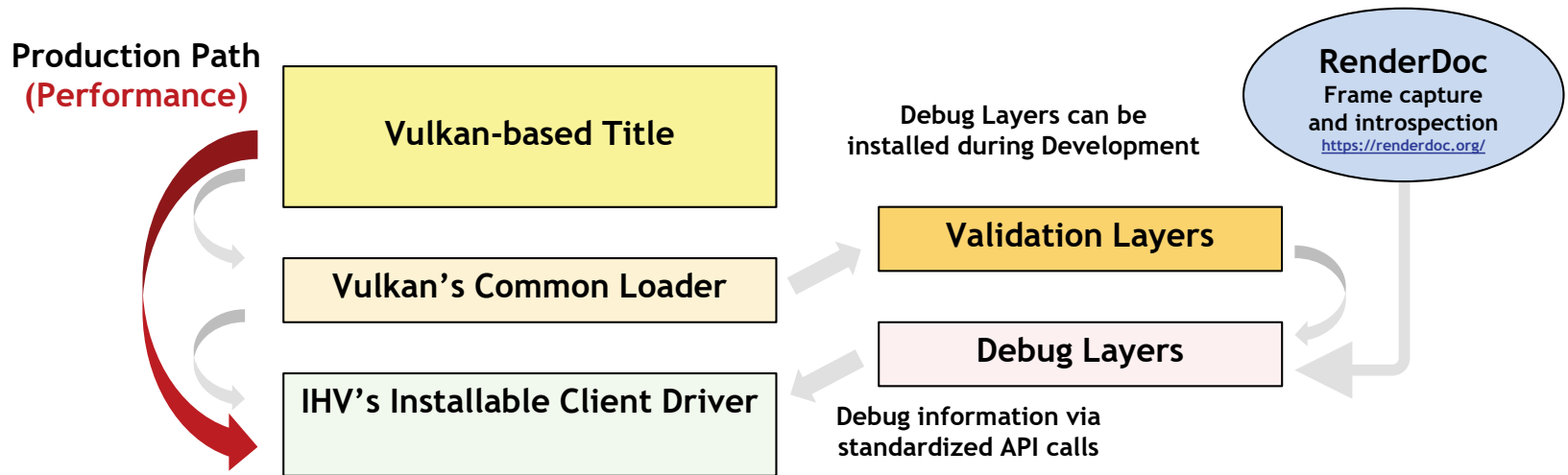
Repositories	1.8K
Code	
Commits	259K
Issues	8K
Topics	16
Wikis	577

SaschaWillems/**Vulkan**  
Examples and demos for the new *Vulkan* API  
vulkan vulkan-api  
MIT license Updated 2 hours ago



# Vulkan Layer Architecture

- Layered design for cross-vendor tools innovation and flexibility
  - IHVs plug into a common, extensible layer architecture for code validation, debugging and profiling during development without impacting production performance
- LunarG open-source Vulkan SDK ships on Windows, Linux and Mac
  - Validation, debug, and device simulation layers



# New Vulkan Developer Tools

- **Vulkan Layer Factory (VLF)**
  - Rapid layer development through hiding implementation details
- **Device Simulation Layer**
  - Simulate target device capabilities, without requiring actual physical hardware
- **Assistant Layer**
  - Best practices layer that highlights potential performance issues, questionable usage patterns, common mistakes, and items not specifically prohibited by the Vulkan specification but that may lead to application problems

**Delivered with the LunarG Vulkan SDK**

<https://vulkan.lunarg.com/>

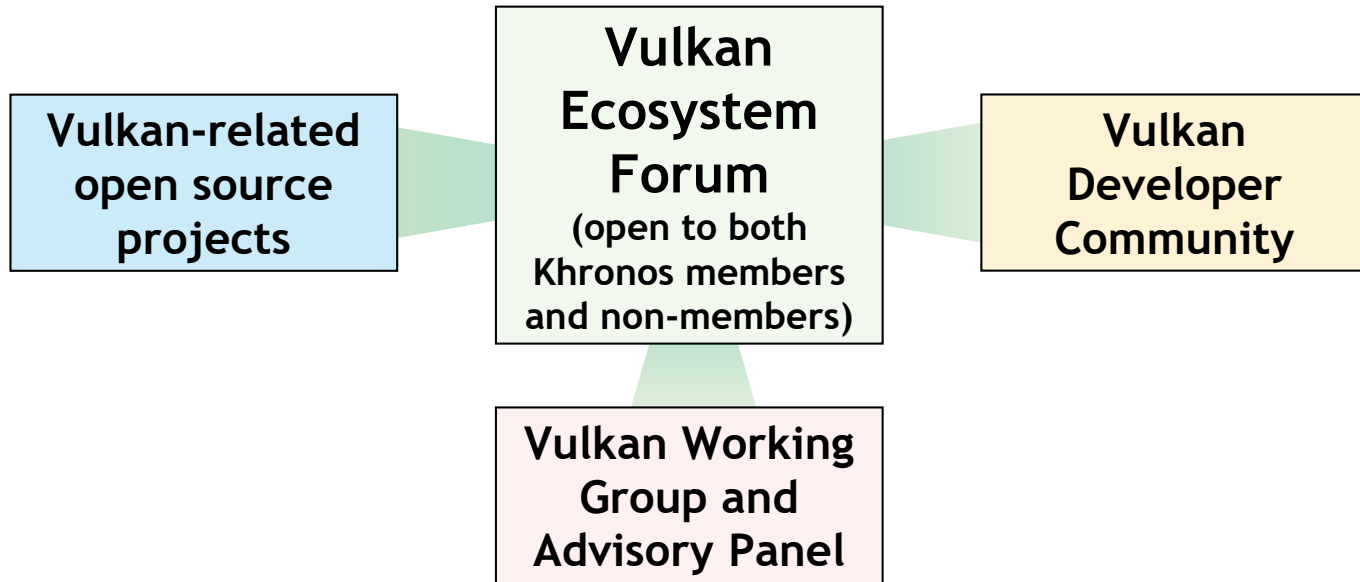
**Source available in the LunarG Vulkan Tools repository**

<https://github.com/LunarG/VulkanTools>



# Public Vulkan Ecosystem Forum

A new public forum to share ecosystem issues and opportunities -  
and coordinate solutions!

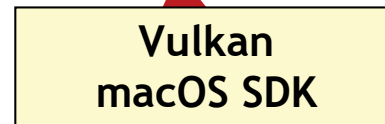


Go to <https://github.com/KhronosGroup/Vulkan-Ecosystem> to join the conversation!

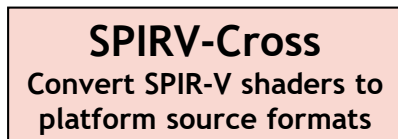
# Bringing Vulkan 1.0 Apps to Apple Platforms



Dota 2 running on Mac up to 50% faster than native OpenGL



Open source SDK to build, run, and debug applications on macOS including validation layer support



MoltenVK for macOS and iOS  
For macOS 10.11, iOS 9.0 and up



Previously a paid product  
Now released into OPEN SOURCE  
Completely free to use - no fees or royalties  
- including for commercial applications

# SPIR-V Ecosystem

**Open source tools and translators**

<https://github.com/KhronosGroup/SPIRV-Tools>

MSL  
HLSL  
GLSL

**SPIRV-Cross**

**SPIR-V (Dis)Assembler**

**SPIR-V Validator**

**SPIRV-opt | SPIRV-remap**

**Additional Intermediate Forms**

**SPIR-V Optimizations**

- Inlining (exhaustive)
- Store/Load Elimination
- Dead Code Elimination
- Dead Branch Elimination
- Common Uniform Elimination
- Loop Unrolling and Constant Folding
- Common Subexpression Elimination

**Third party kernel and shader languages**

GLSL HLSL

**gslang** **DXC**

**SPIR-V**

- Khronos defined cross-API IR
- Native graphics and parallel compute
- Easily parsed/extended 32-bit stream
- Data object/control flow retained for effective code generation/translation

**OpenCL C Front-end** **OpenCL C++ Front-end** **SYCL Front-end**

```

SPIR-V Magic #: 0x07230203
SPIR-V Version 99
Builder's Magic #: 0x051a00BB
<id> bound is 50
0
OpMemoryModel
Logical
GLSL450
OpEntryPoint
Fragment shader
function <id> 4
OpTypeVoid
<id> is 2
OpTypeFunction
<id> is 3
return type <id> is 2
OpFunction
Result Type <id> is 2
Result <id> is 4
0
Function Type <id> is 3
    
```

**LLVM to SPIR-V Bi-directional Translator**

**LLVM**



Khronos liaising with Clang/LLVM Community  
E.g. discussing SPIR-V as supported Clang target

**SPIR-V 1.3 released with Vulkan 1.1**

IHV Driver Runtimes



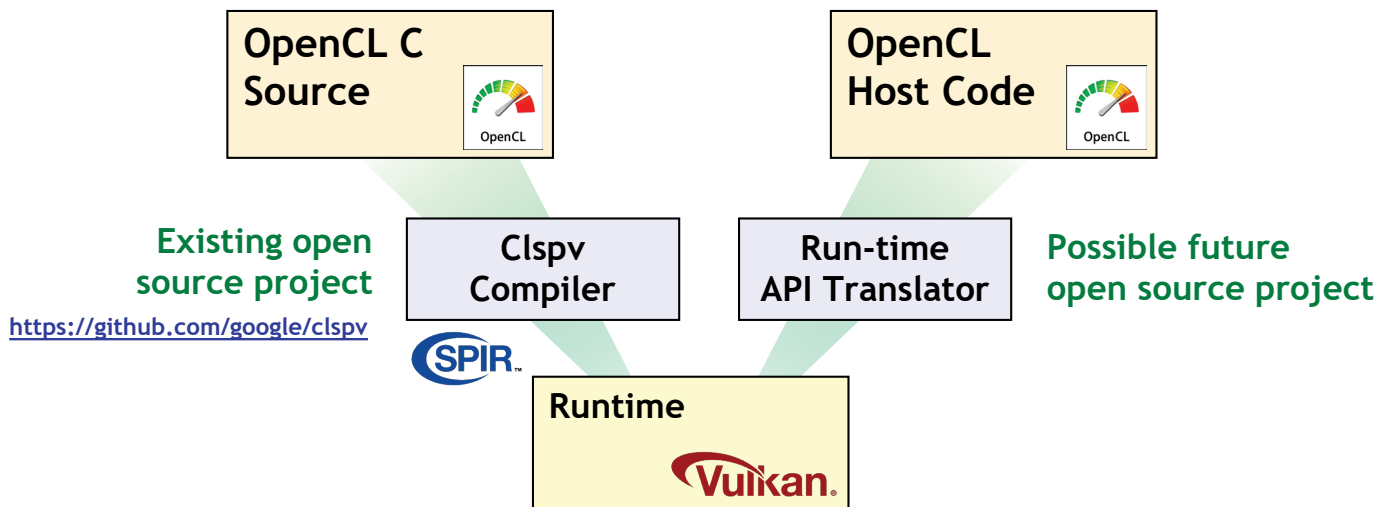
# Clspv OpenCL C to Vulkan Compiler



Adobe



- Experimental collaboration between Google, Codeplay, and Adobe
  - Successfully tested on over 200K lines of Adobe OpenCL C production code
  - Released in open source <https://github.com/google/clspv>
  - Tracks top-of-tree LLVM and clang, not a fork
- Compiles OpenCL C's programming model to Vulkan's SPIR-V execution environment
  - Proof-of-concept that OpenCL compute can be brought seamlessly to Vulkan

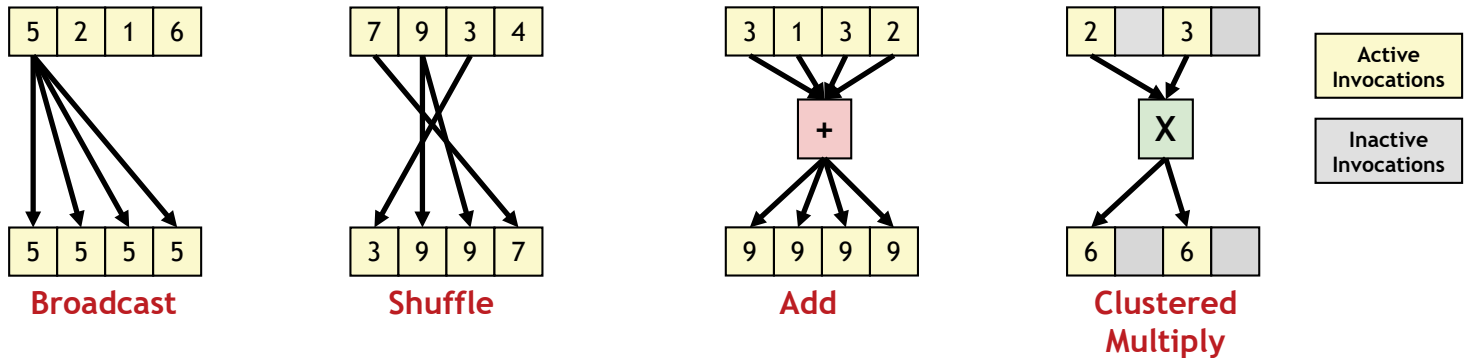


# New Functionality in Vulkan 1.1

- 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)



# Proven Extensions Now in Vulkan 1.1 Core

- **Multiview**
  - A single render pass can render to multiple image views simultaneously
  - Use cases include rendering left and right eye views to a stereo VR headset, or six face views of a cube map, with a single draw call
- **Device Groups**
  - Enables homogeneous multi-GPU systems such as AMD CrossFireX and NVIDIA SLI for high-performance gaming and VR
  - Device groups make the number of GPUs in the system relatively transparent to the application
  - Applications can be written to use one or many GPUs with a minimum of changes
- **Cross-process and Cross-API sharing**
  - Share memory and sync primitives (semaphores and fences) between APIs in a single application, or between multiple applications
  - Many applications, e.g. allowing a compositor to present images from Vulkan and OpenGL ES applications to the same display device
  - This features is used in the Valve Steam VR SDK and other advanced mobile platforms
- **Advanced Compute Functionality**
  - Read and write 16-bit quantities stored in GPU memory, and to refer to data structures using a restricted form of pointers
  - Greatly expands Vulkan's ability to support GPU compute kernels
- **HLSL support**
  - Relaxed block layout enables support for the same memory data layout constraints as Microsoft's HLSL
  - Enables identical HLSL shaders in both Vulkan and DX applications
  - Easier translation of HLSL into SPIR-V, the portable compiled shader format accepted by Vulkan
- **YCbCr support**
  - Sample the YCbCr color formatted textures produced by many video codecs
  - Useful for compositing video streams and mixing them with other graphical content





# Vulkan 1.1 Shipping Today

- **Specification:** in open source for community use and feedback
- **Conformance Tests:** in open source for responsive bug fixing and enhancements
- **Tools:** LunarG SDK and validation/debug/simulation/assistant layers - all in open source

GPU Vendors with conformant Vulkan 1.1 drivers



**Khronos' Ongoing Vulkan Mission**  
Continue to build the complete Vulkan Ecosystem  
Specifications, tests, tools and community  
Listen and prioritize developer needs  
Drive GPU technology in the industry

