Vulkan 1.2 Launch
15th January 2020
Vulkan 1.2 is Launched!

- Specification is final and publicly available
- Multiple GPUs are passing conformance tests
  - First drivers are shipping today
- Proven roadmap process includes new functionality requested and proven by developers
- Improved performance, enhanced visual quality and easier development
- Work underway to rapidly upgrade open source ecosystem and tools to Vulkan 1.2

Continuing to build on significant Vulkan industry momentum
Pervasive Vulkan

Desktop and Mobile GPUs

Platforms

- Desktop
- Android (Android 7.0+)
  (Vulkan 1.1 required on Android Q)
- Apple (via porting layers)
- Media Players
- Consoles
- Virtual Reality
- Cloud Services
- Game Streaming
- Embedded

Engines

Note: The version of Vulkan available will depend on platform and vendor

This work is licensed under a Creative Commons Attribution 4.0 International License
Vulkan AAA Content

This work is licensed under a Creative Commons Attribution 4.0 International License
Vulkan Mobile Content
Vulkan Momentum

https://en.wikipedia.org/wiki/Vulkan_(API)

>50 Games Titles

>15 Games Engines

This work is licensed under a Creative Commons Attribution 4.0 International License.
Vulkan Ecosystem Evolution

**Strengthening Tools and Compilers**
- Improved developer tools (SDK, validation/debug layers)
- Shader toolchain improvements (size, speed, robustness)
- Shading language flexibility - HLSL and OpenCL C support
- Continuously strengthening conformance testing

**Vulkan 1.0 Extensions**
- Maintenance updates plus additional functionality
  - Multiview
  - Multi-GPU
  - Enhanced Windows Sys Integration
  - Increased Shader Flexibility: 16-bit storage, Variable Pointers
  - Enhanced Cross-Process and Cross-API Sharing

February 2016
Vulkan 1.0

**Vulkan 1.1 Extensions**
- Maintenance updates plus additional functionality
  - Reduced precision arithmetic types in shaders
  - Bindless resources
  - HLSL-compatible memory layouts
  - Formal memory model
  - Buffer references
  - Timeline semaphores

March 2018
Vulkan 1.1
Integration of 1.0 Extensions plus new functionality
e.g. Subgroup Operations

**Building Vulkan’s Future**
- Listening and prioritizing developer needs
- Driving GPU technology

**Roadmap Discussions**
- Machine Learning
- Ray Tracing
- Video encode / decode
- Variable Rate Shading
- Mesh Shaders

January 2020
Vulkan 1.2
Integration of 1.1 Extensions and SPIR-V 1.5

**Widening Platform Support**
- Pervasive GPU vendor native driver availability
- Open source drivers - ANV (Intel), AMDVLK/RADV (AMD)
- Vulkan Portability to macOS/iOS and DX12

This work is licensed under a Creative Commons Attribution 4.0 International License

© The Khronos® Group Inc. 2020 - Page 7
New Vulkan 1.2 Functionality in Core

- Vulkan 1.2 rolls up 23 previous released extensions into a new core Vulkan API
  - Improved performance, enhanced visual quality and easier development
- Creates a simplified specification and development target
  - Reduces uncertainty of extensions not being available on some platforms
  - Core features don’t need individual enabling

Requests from Vulkan Developers
- VK_KHR_timeline_semaphore - more manageable synchronization
- VK_KHR_descriptor_indexing - reusing descriptor layouts for multiple shaders
- VK_KHR_buffer_device_address - bindless resources
- VK_KHR_imageless_framebuffer - framebuffer definition without images
- VK_KHR_host_query_reset - easier resetting of queries

Improved Layering Support for Other 3D APIs
- VK_KHR_uniform_buffer_standard_layout - support HLSL constant buffer layouts
- VK_EXT_scalar_block_layout - more layout support for HLSL (optional)
- VK_KHR_draw_indirect_count - for OpenGL (optional)
- VK_KHR_separate_stencil_usage - to streamline DX ports
- VK_KHR_separate_depth_stencil_layouts - to streamline DX ports
- SPIR-V 1.4 - many HLSL features
- SPIR-V 1.5 - to support Vulkan 1.2

API Usability Improvements
- VK_KHR_driver_properties - reports latest passing CTS version
- VK_KHR_create_renderpass2 - more extensible renderpass objects
- Vulkan 1.1/1.2 unified feature and property structs

Exposing New Hardware Capabilities
- VK_KHR_image_format_list - improve image view performance
- framebufferIntegerColorSampleCounts - more multi-sample formats
- VK_KHR_sampler_mirror_clamp_to_edge - widely supported mode (optional)
- VK_KHR_sampler_filter_minmax - for newer GPUs (optional)
- VK_KHR_shader_viewport_index_layer - for newer GPUs (optional)
- VK_KHR_shader_float16_int8 - proper fp16 support (optional)
- VK_KHR_shader_float_controls - control over rounding, etc. (optional)
- VK_KHR_vulkan_memory_model - precise memory model spec (optional)
- VK_KHR_shader_subgroup_extended_types - more subgroup types (optional)
- VK_KHR_8bit_storage - 8-bit types in SSBOs/UBOs (optional)
- VK_KHR_shader_atomic_int64 - (optional)
- VK_KHR_depth_stencil_resolve - resolve modes for depth/stencil (partly optional)

Vulkan 1.2 deliberately does not mandate new hardware functionality so that all Vulkan GPU drivers are able to be upgraded
Timeline Semaphore Primitive

- **Before:** separate VkFence and VkSemaphore for synch with host & across device queues
  - Binary state - so many fences and semaphores often needed to synch parallel operations

- **Timeline Semaphore is much simpler to manage and much more powerful**
  - Unified - covers all synchronization across device queues and host
  - 64-bit monotonically increasing value that multiple threads can update and wait on
  - Apps use the 64-bit state to define their own thread communication protocols

- **See [Khronos Timeline Semaphore Blog](https://github.com/KhronosGroup/Vulkan-ExtensionLayer)**
  - For more details

An open source implementation of the timeline semaphore API is also available as a layer over Vulkan 1.1

https://github.com/KhronosGroup/Vulkan-ExtensionLayer
Vulkan 1.2 Hardware Support

Five GPU vendors have Vulkan 1.2 implementations passing Vulkan 1.2 conformance tests at the time of specification launch
PLUS open source Mesa RADV driver for AMD
For release status updates see the Vulkan Public Release Tracker

NVIDIA’s Vulkan 1.2 drivers are available today with full functionality for both Windows and Linux
Open Source Ecosystem Upgrading to Vulkan 1.2

**RenderDoc**
Latest Release: v1.5 - 9 Oct, 2019

*RenderDoc Debugger*
Single-frame capture and detailed introspection of any application
https://renderdoc.org/

**Vulkan Samples**
Collection of samples and resources to aid developing optimized Vulkan applications
https://github.com/KhronosGroup/Vulkan-Samples

**Vulkan SDK with Development/Debug Layers**
Windows, Linux - Ubuntu packages, Linux- Tarball, macOS
www.vulkan.lunarg.com

**Vulkan Guide**
Help for developers to get up and going with the world of Vulkan with links to many other useful resources
https://github.com/KhronosGroup/Vulkan-Guide
HLSL as First Class Vulkan Shading Language

- **DXC** - Microsoft’s next-gen HLSL compiler
  - Open sourced in January 2017
  - Based on LLVM/Clang

- **Spiregg**: HLSL to SPIR-V within DXC
  - Google and others contributing
  - Same front-end and validation as D3D

- **Covers all native HLSL features**
  - Math types, Control flows, Functions, enums
  - Resource types and methods, Namespaces, structs
  - 16-bit and 32-bit types
  - Shader Model 6.2 and below

- **Adding Vulkan 1.2 and extensions support**
  - Vendors are enabled to contribute
  - E.g. VKRay extension added by NVIDIA

---

Many Vulkan Games Already Shipping Using HLSL

[Blog on using HLSL with Vulkan](#)
### Vulkan and Open Source Layering Projects

**Fighting Platform Fragmentation**

<table>
<thead>
<tr>
<th>Layers Over</th>
<th>Vulkan</th>
<th>OpenGL</th>
<th>OpenCL</th>
<th>OpenGL ES</th>
<th>DX12</th>
<th>DX9-11</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vulkan</td>
<td>Zink</td>
<td>clspv</td>
<td>GLOVE</td>
<td>vkd3d</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OpenGL</td>
<td>gfx-rs</td>
<td>Ashes</td>
<td>Angle</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DX12</td>
<td>gfx-rs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DX9-11</td>
<td>gfx-rs</td>
<td>Ashes</td>
<td>Angle</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Metal</td>
<td>MoltenVK</td>
<td>gfx-rs</td>
<td>Angle</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Vulkan added OpenGL-style line extension**
- **Vulkan adding more compute for fuller support for OpenCL kernel deployment**
- **Vulkan adding extensions to ease layering of DX**

**Vulkan Portability enables multi-vendor layered subsets to be queryable and all present functionality to be tested for conformance**

**Demand for Vulkan everywhere, even if no native drivers on platform**

- Vulkan Portability is an effective porting layer for other APIs for portability and stack simplification

© The Khronos® Group Inc. 2020 - Page 13
SPIR-V Ecosystem

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

OpenCL C
Front-end

OpenCL C++
Front-end

Intel DPC++

SYCL for ISO C++
Front-end

CLSpV

LLVM to SPIR-V Bi-directional Translators

Khronos cooperating closely with clang/LLVM Community

3rd Party-hosted Open Source Projects

Khronos-hosted Open Source Projects

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

Environment spec for each target API used to drive compilation

Optimization Tools

SPRIV-opt | SPRIV-remap

SPRIV (Dis)Assembler

SPRIV-Validator

SPRIV-Cross

3rd party kernel & shader languages

GLSL

HLSL

MSL

HLSL

GLSL

glslang

DXC

C++ for OpenCL in clang Front-end

Intel DPC++

SYCL for ISO C++
Front-end

3rd Party-hosted Open Source Projects

Khronos-hosted Open Source Projects

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

This work is licensed under a Creative Commons Attribution 4.0 International License
Running DX Games on Linux Over Vulkan

- DXVK - Vulkan-based implementation of Direct3D 9/10/11
  - GitHub open source by Philip Rebohle and Joshua Ashton with support from Valve
- Vulkan has added multiple extensions to support efficient layering of D3D
  - Removing impedance mismatches between the two APIs
- DXVK, Wine Windows Compatibility Layer and Valve’s Proton tool
  - Enables thousands of Steam PC games on Linux

Vulkan 1.2 Functionality for efficient D3D layering/HLSL

- VK_KHR_host_query_reset
  - Easier resetting of queries
- VK_KHR_uniform_buffer_standard_layout
  - To support HLSL constant buffer layouts
- VK_EXT_scalar_block_layout
  - More layout support for HLSL (optional)
- VK_KHR_separate_stencil_usage
  - Used in many DX ports
- VK_KHR_separate_depth_stencil_layouts
  - Used in many DX ports
- SPIR-V 1.4/1.5 include many HLSL features
Deploying OpenCL C Over Vulkan

- **Clspv** - Google’s experimental compiler for OpenCL C to Vulkan SPIR-V
  - Open source - tracks top-of-tree LLVM and clang, not a fork
- Adobe Premiere Rush has 200K lines of OpenCL C kernel code
  - Professional-quality, cross-platform video capture and editing system
  - Now shipping on Android on Vulkan
Vulkan Portability Initiative on Apple

Almost all mandatory Vulkan 1.0 functionality is supported:
- No Triangle Fans
- No separate stencil reference masks

Selected Optional Features and Extensions are added as required - driven by industry input and feedback
- Robust buffer access
- BC texture compressed formats
- Fragment shader atomics
- Tessellation

https://github.com/KhronosGroup/MoltenVK

Open source SDK to build, run, and debug applications on macOS - including validation layer support
https://vulkan.lunarg.com/

MoltenVK supports macOS 10.11 / iOS 9.0 and up

Open source beta release for macOS

Open source for MacOS and iOS
Free to use - no fees or royalties including commercial apps

This work is licensed under a Creative Commons Attribution 4.0 International License
<table>
<thead>
<tr>
<th>Month</th>
<th>Event Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>June 2018</td>
<td>Production Dota 2 on Mac Ships - up to 50% more perf than Apple's OpenGL</td>
</tr>
<tr>
<td>September 2018</td>
<td>Multiple iOS and macOS apps shipping e.g. Forsaken Remastered</td>
</tr>
<tr>
<td>June 2019</td>
<td>Initial Vulkan Performance On macOS With Dota 2 Is Looking Very Good</td>
</tr>
<tr>
<td>June 2019</td>
<td>Qt Running on Mac through MoltenVK</td>
</tr>
<tr>
<td>September 2018</td>
<td>First iOS Apps using MoltenVK ship through app store</td>
</tr>
<tr>
<td>November 2018</td>
<td>Google Filament PBR Renderer on Mac</td>
</tr>
<tr>
<td>November 2018</td>
<td>Initial ports of DX games in progress using Vulkan on Mac</td>
</tr>
<tr>
<td>January 2019</td>
<td>Artifact from Steam ships on MoltenVK on macOS - first Vulkan-only Valve app on Mac</td>
</tr>
<tr>
<td>June 2019</td>
<td>Underlords from Steam ships on MoltenVK on macOS - second Vulkan-only Valve app on Mac</td>
</tr>
</tbody>
</table>

*This work is licensed under a Creative Commons Attribution 4.0 International License*
Vulkan and OpenGL for Professional Apps

- Professional applications beginning to transition to Vulkan
  - Reduced CPU bottlenecks and multi-threading
  - Parallel compute, graphics and data movement
  - Vulkan now has OpenGL-style lines extension for CAD
  - Advanced functionality coming - such as ray tracing

- Vulkan OpenGL Interop enables incremental transition to Vulkan
  - Modern-style, shared explicit memory objects

- Dassault Systèmes achieves interactive object space AO in CATIA, an OpenGL application
  - Using the NVIDIA Vulkan VKRay vendor extension for Ray Tracing
Other Vulkan Updates

January 2020
Vulkan Portability Initiative

Enabling Vulkan applications on platforms without native drivers by layering cleanly queryable subsets of Vulkan over DX12, Metal and other APIs

Multiple Layered Vulkan Implementations

- Additional open source run-times over additional backends
  E.g. gfx-rs for Vulkan over Metal and DX12 - useful for Vulkan on UWP platforms such as Windows 10 S, Polaris, Xbox One.
  Secondary backends include OpenGL/D3D11
  - https://github.com/gfx-rs/gfx
  - https://github.com/gfx-rs/portability

Portability Extension

Layered implementations can portably expose what Vulkan functionality is not supported

TODAY

Open source tools, SDKs and libraries to bring Vulkan 1.0 applications to Apple using Metal

Enhanced Vulkan Layers

Extend DevSim/Validation Layers to flag or simulate queries for features not present

Extend Vulkan Conformance Test Suite

To handle layered implementations - what is present must work!

This work is licensed under a Creative Commons Attribution 4.0 International License
Khronos Standards for Immersive Computing

- Vision and sensor processing - including neural network inferencing for machine learning
- Portable interaction with VR/AR sensor, haptic and display devices
- High-performance, low-latency 3D Graphics

Download 3D object and scene data
OpenXR is used with a 3D API

High-performance, low-latency 3D rendering and composition*
  Multiview
  Context priority
  Front buffer rendering
  Tiled rendering (beam racing)
  Variable rate rendering

Display, composition and optical correction parameters

Cross-platform access to XR HMDs and sensors
  XR application lifecycle
  Input device discovery and events
  Sensor tracking and pose calculation
  Frame timing and display composition
  Haptics Control

* OpenXR can be used with other 3D APIs such as Direct3D, OpenGL and OpenGL ES

This work is licensed under a Creative Commons Attribution 4.0 International License
Analytic Rendering Exploratory Group

Analytic Rendering is image generation performed primarily to gain and communicate insights into complex data sets primarily for scientific visualization and data analytics.

Is there a need for a cross-platform open standard API?

Visualization Apps and Engines have to be ported to multiple APIs

Cross-vendor API to provide access to state-of-the-art rendering across multiple platforms
Potential Analytic Rendering API Design

Rather than specifying the details of the rendering process, an Analytic Rendering API would enable a visualization application to simply describe the relationship between objects in a scene to be rendered and leave the details of the rendering process to a backend renderer.

Some Initial Exploratory Group Members:
- Delta H
- Intel
- Kitware
- Oak Ridge National Laboratory
- NVIDIA
- SURFACE Engineering Company
- TACC

Khronos Exploratory Groups discuss the need for a new standard with no cost or IP implications. Open to all - even non-members - more details:

https://www.khronos.org/exploratory/analytic-rendering/
Khronos and Vulkan Background

January 2020
Over 150 members worldwide
Any company is welcome to join

>150 Members ~ 40% US, 30% Europe, 30% Asia

Open, non-profit, member-driven industry consortium creating advanced, royalty-free interoperability standards for 3D graphics, augmented and virtual reality, parallel programming, vision acceleration and machine learning

This work is licensed under a Creative Commons Attribution 4.0 International License
# Khronos Active Initiatives

## 3D Graphics
- Desktop, Mobile, Web
- Embedded and Safety Critical

## 3D Assets
- Authoring and Delivery

## Portable XR
- Augmented and Virtual Reality

## Parallel Computation
- Vision, Inferencing, Machine Learning

### Guidelines for creating APIs to streamline system safety certification

- **Heterogeneous Communications**
  - between offload compute devices

- **Exploratory Groups**
  - Making High-Level Languages more effective at acceleration offload

- **Rendering for scientific visualization and data analytics**

---

This work is licensed under a Creative Commons Attribution 4.0 International License

© The Khronos® Group Inc. 2020 - Page 28
Vulkan and New Generation GPU APIs

Modern architecture | Low overhead | Multi-thread friendly
EXPLICIT GPU access for EFFICIENT, LOW-LATENCY, PREDICTABLE performance

Vulkan is a non-proprietary, royalty-free open standard
Portable across multiple platforms - desktop, mobile and embedded

Note: The version of Vulkan available will depend on platform and vendor
Vulkan for Direct GPU Control

**High-level Driver Abstraction**
- Layered GPU Control
- Context management
- Full GLSL compiler
- Error detection

**Thin Driver**
- Explicit GPU Control

**Multiple Front-end Compilers**
- GLSL, HLSL etc.

**Application**
- Memory allocation
- Thread management
- Explicit Synchronization
- Multi-threaded generation of command buffers

**Loadable debug and validation layers**

**Complex drivers**
- Cause overhead and inconsistent behavior across vendors
- Always active error handling
- Full GLSL preprocessor and compiler in driver
- OpenGL vs. OpenGL ES

**Simpler drivers**
- Application has the best knowledge for holistic optimization - no 'driver magic'
- Explicit creation of API objects before usage - efficient, predictable execution
- Easier portability - no fighting with different vendor heuristics
- Validation and debug layers loaded only when needed
- SPIR-V intermediate language: shading language flexibility
- Unified API across mobile and desktop platforms
- Multiple graphics, command and DMA queues

**GPU**
- A Graphics API

**GPU**
- A GPU API

---

© The Khronos® Group Inc. 2020 - Page 30

This work is licensed under a Creative Commons Attribution 4.0 International License
Khronos Ecosystem Engagement

Contribute to open source specs, CTS, tools and ecosystem

- Khronos Forums and Slack Channels
  - Open to all! https://community.khronos.org/
  - www.khr.io/slack

- Advisory Panels
  - Spec fixes and suggestions made under the Khronos IP Framework. Open source contributions under repo’s CLA - typically Apache 2.0 https://github.com/KhronosGroup
  - Advisors under the Khronos NDA and IP Framework can comment and contribute to requirements and draft specifications https://www.khronos.org/advisors/

- Working Groups
  - $0
  - Khronos members under Khronos NDA and IP Framework participate and vote in working group meetings https://www.khronos.org/members/
Khronos New Initiative Process

Khronos members and/or non-members → Idea for a new Khronos Initiative

Khronos organizes a discussion forum → Initiative Proposal

Khronos establishes an Exploratory Group to CREATE a Statement of Work

Open to all

No IP Commitments

High-level requirements and use case discussions

SOW

Under NDA

Khronos Working Group

Khronos Members

Work and participants protected by Khronos IP Framework

Khronos extends an open invitation to any Embedded Vision Alliance members that have business needs for new industry open standards to make a New Initiative Proposal
Get Involved!

- Khronos is creating cutting-edge royalty-free open standards
  - For 3D, compute, vision, inferencing acceleration
- These slides and information on Khronos Standards: [www.khronos.org](http://www.khronos.org)
- Any company is welcome to join Khronos: [https://www.khronos.org/members/](https://www.khronos.org/members/)
- Khronos Developer Forum: [https://community.khronos.org/](https://community.khronos.org/)
- Khronos Developer Slack Channel: [www.khr.io/slack](http://www.khr.io/slack)
- Neil Trevett: ntrevett@nvidia.com | @neilt3d

**Benefits of Khronos membership**

- Gain early insights into industry trends and directions
- Influence the design and direction of key open standards that will drive your business
- Accelerate your time-to-market with early access to specification drafts
- Network with domain experts from diverse companies in your industry
- State-of-the-art IP Framework protects your Intellectual Property
- Publicly Release Specifications and Conformance Tests
- Enhance your company reputation as an industry leader through Khronos participation