Fast Forward

Neil Trevett
Khronos President
NVIDIA VP Developer Ecosystems
ntrevett@nvidia.com | @neilt3d
November 2019
What are Open Standards?

Interoperability Standards define an agreed communication protocol between two ‘entities’

Common products use 100s of open standards

- Device to Wireless Networks
- Device to its Charger
- Internal Components to other Internal Components
- Downloaded Web content to the Web Browser
- Camera App to Video and Photo Playback Applications
- Games to 3D GPU Acceleration

Many Standard Defining Organizations (SDOs)
Each has a focus area of expertise that gathers an effective quorum
Each creates a safe space for cooperation
The Need for Interoperability Standards

Standards Grow Markets
By reducing consumer confusion and increasing capabilities and usability

Standards Reduce Costs
By sharing development between many companies and driving volume

Standards Accelerate Time to Market
With well-proven testing and interoperability

Standards Do Not Stifle Innovation
Companies can compete on implementation quality, performance, power etc. etc.

True OPEN Standards
Are not controlled by a single company - but by the industry - typically through an SDO
Well defined participation, governance and intellectual property frameworks
Khronos is an open, non-profit, member-driven industry consortium developing royalty-free standards to harness the power of silicon acceleration for demanding graphics rendering and computationally intensive applications such as 3D Graphics, Virtual Reality, Augmented Reality, and Machine Learning.
Khronos warmly welcomes Australian and Asian company participation!!
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

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

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

Rendering for scientific visualization and data analytics

Heterogeneous Communications between offload compute devices
Vulkan for Direct GPU Control

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

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

**Application**
- Single thread per context

**Thin Driver**
- Explicit GPU Control

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

**SPIR-V**
- Pre-compiled shaders

**Loadable debug and validation layers**

**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

**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
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
Pervasive Vulkan

Major GPU Companies supporting Vulkan for Desktop and Mobile Platforms

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

Game Engines

- Epic Games
- CryEngine
- Unity
- Valve
- Croteam
- Serious Engine
- NetEase Games

http://vulkan.gpuinfo.org/
Vulkan AAA Content Shipping on Desktop...

Vulkan-only AAA Titles on PC

AAA titles on Linux

Titles on PC AND macOS
...and Mobile

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

© The Khronos Group Inc. 2019 - Page 11
Vulkan 1.1 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
More rigorous conformance testing

Building Vulkan’s Future
Listen and prioritize developer needs
Drive GPU technology

Released Vulkan 1.1 Extensions
Reduced precision arithmetic types in shaders
Bindless resources
HLSL-compatible memory layouts
Formal memory model
Buffer references
Timeline semaphores
OpenGL-class lines and Interop

Roadmap Discussions
Machine Learning acceleration
Ray Tracing
Video encode / decode
Generalized subgroup operations

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

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

February 2016
Vulkan 1.0

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

March 2018
Vulkan 1.1

https://www.khronos.org/registry/vulkan/specs/1.1-khr-extensions/html/vkspec.html#extension-appendices-list
OpenGL Vulkan Interop

- Enables OpenGL applications to incrementally leverage Vulkan functionality
  - 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
  - See the Demo at the NVIDIA booth
Key Vulkan Online Open Source Resources

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

Vulkan Guide
Help for developers to get up and going with the world of Vulkan with kinks to many other useful resources
https://github.com/KhronosGroup/Vulkan-Guide

RenderDoc
Latest Release: v1.5 - 9 Oct, 2019
Download (Win x64)
Source Code • Documentation • Other builds • Contact

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

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

Third party kernel and shader languages

GLSL
HLSL

Third party kernel and shader languages

glslang
DXC

SPIR-V (Dis)Assembler

SPIR-V Cross

SPIR-V Validator

SPIRV-opt | SPIRV-remap

Optimization Tools

Environment spec for each target API used to drive compilation

GLSL
HLSL

Third party kernel and shader languages

GLSL
HLSL

SPIR-V Validator

SPIR-V Cross

SPIRV-opt | SPIRV-remap

Optimization Tools

Environment spec for each target API used to drive compilation

SPIR-V
Khronos-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

SYCL for
ISO C++
Front-end

C++ for
OpenCL in
clang
Front-end

Khronos cooperating
with clang/LLVM
Community

3rd Party-hosted
Open Source Projects

Khronos-hosted
Open Source Projects

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

SPIRV-Cross

Optimization Tools

GLSL
HLSL

Third party kernel and shader languages

glslang
DXC

SPIR-V
Khronos-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

SYCL for
ISO C++
Front-end

C++ for
OpenCL in
clang
Front-end

Khronos cooperating
with clang/LLVM
Community

3rd Party-hosted
Open Source Projects

Khronos-hosted
Open Source Projects

https://github.com/KhronosGroup/SPIRV-Tools
Open Source Layering Projects
Breaking through 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 clvk GLOVE Angle vkd3d DXVK D9VK</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OpenGL</td>
<td>gfx-rs Ashes</td>
<td>Angle</td>
<td></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>DXGI-11</td>
<td>gfx-rs Ashes</td>
<td>Angle</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Metal</td>
<td>MoltenVK gfx-rs</td>
<td>Angle</td>
<td>Growing interest Apple deprecation?</td>
<td>Need for consistent OpenGL everywhere, primarily for WebGL</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Vulkan is an effective porting layer for app portability and stack simplification.

Vulkan Portability enables multi-vendor layered subsets to be queryable and conformant.

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

Vulkan added OpenGL-style line extension.

Vulkan adding more compute for fuller support for OpenGL kernel deployment.

Vulkan added extensions to ease layering of DX.

Breaking through platform fragmentation.
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!
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

SPIRV-Cross
Convert SPIR-V shaders to Metal Shaders

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

macOS / iOS Run-time
Maps Vulkan to Metal

MoltenVK supports macOS 10.11 / iOS 9.0 and up
Open source for MacOS and iOS
Free to use - no fees or royalties including commercial apps

SPIRV-Cross
Convert SPIR-V shaders to Metal Shaders

Open source beta release for macOS
Vulkan Apps Shipping On Apple

Forsaken Remastered was just updated with Vulkan support! If you’re on Linux, you’re probably hitting 60fps with the existing OpenGL renderer, but it’s good to be future proof. If you’re on a Mac, though, you definitely want to switch. On my MacBook, the framerate goes from around 15 to a solid 60!

Initial Vulkan Performance On macOS With Dota 2 Is Looking Very Good

Written by Michael Lavelle in Valve on 1 June 2018 at 03:37 PM EDT. 34 Comments

Yesterday Valve released Vulkan support for Dota 2 on macOS. Indeed, this first major game relying upon MoltenVK for mapping Vulkan over the Apple Metal drivers is delivering performance gains.

Production Dota 2 on Mac Ships - up to 50% more perf than Apple’s OpenGL

First iOS Apps using MoltenVK ship through app store

Qt Running on Mac through MoltenVK

Multiple iOS and macOS apps shipping e.g. Forsaken Remastered

Google Filament PBR Renderer on Mac

Initial ports of DX games in progress using Vulkan on Mac

RPCS3 PlayStation 3 Emulator on Mac

Dolphin GameCube and Wii Emulator working on MacOS

Artifact from Steam ships on MoltenVK on macOS - first Vulkan-only Valve app on Mac

Artifact from Steam ships on MoltenVK on macOS - second Vulkan-only Valve app on Mac

Production Dota 2 on Mac Ships - up to 50% more perf than Apple’s OpenGL

June 2018

September 2018

November 2018

January 2019

June 2019

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

- DXVK - Direct3D 10/11 emulator running over Vulkan
  - Open source on GitHub - developed by Philip Rebohle 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 Proton tool
  - Enable thousands of PC games on Linux

Extensions created in response to DXVK issues
  - VK_EXT_transform_feedback
  - VK_EXT_depth_clip_enable
  - VK_EXT_host_query_reset
  - VK_EXT_texel_buffer_alignment
  - VK_EXT_shader_demote_to_helper_invocation

Other extensions used by DXVK
  - VK_EXT_conditional_rendering
  - VK_EXT_memory_budget
  - VK_EXT_memory_priority
  - VK_EXT_shader_viewport_index_layer
  - VK_EXT_vertex_attribute_divisor
  - VK_KHR_draw_indirect_count
  - VK_KHR_shader_draw_parameters

https://www.protondb.com
Khronos Active Initiatives

- **3D Graphics**
  - Desktop, Mobile, Web
  - Embedded and Safety Critical
  - Vulkan
  - OpenGL
  - WebGL
  - COLLADA

- **3D Assets**
  - Authoring and Delivery
  - glTF

- **Portable XR**
  - Augmented and Virtual Reality
  - OpenXR

- **Parallel Computation**
  - Vision, Inferencing, Machine Learning
  - SPIR
  - SYCL
  - NNEF
  - OpenCL
  - OpenVX

---

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
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.

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

https://www.khronos.org/exploratory/analytic-rendering/
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

3D Commerce

COLLADA™

OpenXR™

3D Assets
Authoring and Delivery

Portable XR
Augmented and Virtual Reality

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

Rendering for scientific visualization and data analytics

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
OpenXR provides cross-platform, high-performance access to AR and VR platforms and devices
OpenXR - Solving XR Fragmentation

* OpenXR 1.0 is focused on enabling cross-platform applications. Optional device plugin interface will be supported post V1.0
** Check OpenXR Landing Page for exact availability of OpenXR in shipping run-times and devices [www.khronos.org/openxr](http://www.khronos.org/openxr)

Before OpenXR
XR Market Fragmentation

After OpenXR
Wide interoperability of XR apps and devices
OpenXR is used with a 3D API

**Application or Engine**

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

- 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

© The Khronos® Group Inc. 2019 - Page 27

This work is licensed under a Creative Commons Attribution 4.0 International License
Companies Publicly Supporting OpenXR

OpenXR is a collaborative design
Integrating many lessons from proprietary ‘first-generation’ XR API designs
OpenXR Win-Win-Win

XR Vendors
Can bring more applications onto their platform by leveraging the OpenXR content ecosystem

XR ISVs
Can easily ship on more platforms for increased market reach

XR End-Users
Can run the apps they want on their system - reducing market confusion and increasing consumer confidence
OpenXR 1.0 Released July 2019!

Significant community feedback - thank you!
Improved OpenXR input subsystem, game engine editor support, loader ...

Provisional Specification
GDC, March 2019

Ratify and Release OpenXR 1.0 Specification
SIGGRAPH, July 2019

Finalize Conformance Test Suite
Enable Officially Conformant Implementations

Microsoft
OpenXR runtime for Windows Mixed Reality headsets and HoloLens 2 from Microsoft shipping TODAY
PLUS extensions to support HoloLens 2 hand tracking, eye tracking, spatial mapping and spatial anchors by end of year

Facebook
OpenXR support for Oculus Rift and Oculus Quest
Coming soon

‘Monado’ OpenXR open source implementation from Collabora shipping TODAY

OpenXR 1.0 plugin for Unreal Engine 4.23
Starting with preview 4

Many more coming
Bringing XR to the Web

Native XR Apps

Web XR Apps

The Web will Evolve into the Metaverse

Native 3D Engines

Web 3D Engines

Lifting OpenXR functionality into the Web stack

Close cooperation between WebXR and OpenXR

WebXR

WebGL

three.js

babylon.js

Native 3D Engines

Khronos provides the foundation for native and Web-based 3D/XR

This work is licensed under a Creative Commons Attribution 4.0 International License
XR and 5G
Leveraging High Bandwidth and Low Latency

Wireless mobile device with display and sensors

Location-aware Content Requests

MEC (Multi-access Edge Computing) Server
1. Processes sensor data, including machine learning for environmental lighting, occlusion, scene semantics, object reconstruction and UI
2. Generates imagery from 3D models, including stereo, foveal rendering, ray-tracing, optics pre-distortion, varifocal processing

OpenXR APIs can hide the 5G round trip from applications

NVIDIA EGX

Any needed assets loaded from the cloud to edge server

Generated Augmentations & Scenes

Display composition

Sensor handling

OpenXR

Rich Sensor Data

NVIDIA

Cuda

GLTF

Knubernetes

1. NOC Registry
2. CUDA-X
3. Kubernetes
4. NVIDIA Driver
glTF Real-time 3D Asset Transmission

- Compact to Transmit
- Simple and Fast to Load
- Describes Full Scenes
- Runtime Neutral
- Open and Extensible

glTF is an efficient, reliable run-time 3D transmission format with advanced photorealistic functionality

Functionality in Development
- Draco Mesh Compression
- Universal Compressed Textures
- Second generation PBR (absorption/attenuation, clear coat, subsurface scattering, anisotropy)
- Subdivision surfaces

glTF 2.0 - June 2017
Physically Based Rendering
**glTF 2.0 Scene Description Structure**

- **.gltf (JSON)**
  - Node hierarchy, PBR material textures, cameras

- **.bin**
  - Geometry: vertices and indices
  - Animation: key-frames
  - Skins: inverse-bind matrices

- **.png**
- **.jpg**
- **.ktx2**
  - Textures

---

**Mandatory Metallic-Roughness Materials**

**Optional Specular-Glossiness Materials**
**glTF Ecosystem Evolution**

**Tools!**

Striving for native glTF import and export from every tool. Catalyzed Blender IO as exemplar.

**Consistency!**

Avoid dialects at all costs! Sample viewer and Asset Validator in open source. Sample models and asset generator for unit tests.

**Functionality!**

Balancing functionality versus complexity. glTF is extensible - only bring widely adopted extensions into core.

---

Sample Viewer for accurate Ground Truth glTF renderings

glTF Mesh compression extension provides up to 25x geometry compaction
Draco glTF Mesh Compression Extension

- Library for compressing and decompressing 3D geometric meshes and point clouds
  - Draco designed and built for compression efficiency and speed - great fit with glTF!
  - [https://github.com/google/draco](https://github.com/google/draco)

- Draco glTF extension launched in February 2018
  - [https://github.com/KhronosGroup/glTF/blob/master/extensions/2.0/Khronos/KHR_draco_mesh_compression/README.md](https://github.com/KhronosGroup/glTF/blob/master/extensions/2.0/Khronos/KHR_draco_mesh_compression/README.md)

- Google has released Draco encoders and decoders in open source
  - C++ source code encoder to compress 3D data
  - C++ and JavaScript decoders for the encoded data
  - [https://github.com/google/draco/tree/gltf_2.0_draco_extension](https://github.com/google/draco/tree/gltf_2.0_draco_extension)

- glTF/Draco compression already in use
  - Blender, three.js, BABYLON.JS, Adobe Dimension, glTF pipeline, FBX2glTF, AMD Compressorator and glTF sample models

![Mesh Compression Ratios](image-url)
Universal Textures for glTF

- Fragmentation of GPU texture formats is significant issue for developers
  - Binomial’s ‘Basis Universal’ technology enables JPEG-sized texture assets
  - Transcodable on-the-fly to natively supported compressed GPU formats

- glTF Universal Texture extension uses KTX2 subset as a flexible container
  - Precisely defined for consistent, cross-vendor generation and validation
  - Wide range of (un)(super)compressed texture formats used in Vulkan/DirectX/Metal
  - Supports streaming and full random access to MIP levels
  - Open source tools to create, transcode and upload to WebGL, OpenGL and Vulkan
  - [https://github.com/KhronosGroup/KTX-Software/tree/ktx2](https://github.com/KhronosGroup/KTX-Software/tree/ktx2)

*ASTC support in development
Universal Textures: Compression Ratios

<table>
<thead>
<tr>
<th>Method</th>
<th>File Size</th>
<th>GPU Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncompressed</td>
<td>12,582,912</td>
<td>2,778,518</td>
</tr>
<tr>
<td>PNG</td>
<td>315,619</td>
<td>232,104</td>
</tr>
<tr>
<td>JPEG</td>
<td>2,097,152</td>
<td>2,097,152</td>
</tr>
<tr>
<td>ETC1S</td>
<td>7,000,000</td>
<td></td>
</tr>
<tr>
<td>Universal</td>
<td>10,500,000</td>
<td></td>
</tr>
</tbody>
</table>

FlightHelmet_baseColor
2048 x 2048, RGB

JPEG must be fully decompressed into GPU memory.

Universal textures can be directly transcoded to compressed GPU textures.
KTX2 and .basis files

Two complementary container formats for Basis Universal assets

**BINOMIAL**
‘Basis Universal’ texture compression technology
Enables JPG-sized textures that can be transcoded on-the-fly to natively supported compressed GPU formats

Binomial and Google open sourced ‘Basis Universal’ compressor and transcoder
C++ or WebAssembly code for handling
‘.basis’ format textures in native apps and web sites
https://github.com/binomialLLC/basis_universal
Fine if you are in full control of your texture assets and rendering

Binomial’s ‘Basis Universal’ technology contributed to glTF
Rigorously-defined KTX2 container format supports wide range of texture formats used in Vulkan/DirectX/Metal with streaming and full random access to MIP levels
glTF extension uses KTX2 subset with supercompressed payload using Basis Universal Technology
Great for cross-vendor distribution of textures to multiple applications and engines
Next Generation glTF PBR Materials

- Demand for advanced PBR for photorealistic assets
  - Beyond current ‘Metallic-Roughness’ and ‘Specular-Glossiness’
  - E.g. Absorption/attenuation, clear coat, subsurface scattering, anisotropy

- Extending Metallic-Roughness parameters
  - Consistency and fallbacks for performance for any device

- Inspiration from Dassault Systèmes Enterprise PBR Shading Model (DSPBR)

- Wide industry collaboration for compatibility
  - Dassault Systèmes
  - Google Filament
  - Microsoft BabylonJS
  - NVIDIA MDL
  - OTOY Octane

Join the GitHub Discussion!
https://github.com/KhronosGroup/glTF/issues/1442
glTF Evolution

Extensions shipping and in development
- Draco Mesh Compression
- Universal Compressed Textures
- Second generation PBR (absorption/attenuation, clear coat, subsurface scattering, anisotropy)
- Subdivision surfaces

Baseline of significant industry adoption

glTF 2.0 - June 2017
Native AND Web Apps
Metallic-Roughness and Specular-Glossiness PBR

Seeking Requirements
- Advanced Animation
- LOD and Streaming
- Compressed Point Clouds
- Cross-asset linking
- Enhanced Metadata
- Composability
- Instancing
- CAD/BIM model support
- Encryption and security
- 3D Printing

Nexgen glTF
Version and timing not yet decided
Incorporate popular extensions (with fallbacks)

The glTF Roadmap is Driven by Developer Feedback
Join the GitHub Discussion!
https://github.com/KhronosGroup/glTF/issues/1442

glTF 1.0 - December 2015
Primarily for WebGL
Uses GLSL for materials

This work is licensed under a Creative Commons Attribution 4.0 International License
© The Khronos® Group Inc. 2019 - Page 42
VRM Using glTF 2.0

Hiroki Omae - Unity
3D Commerce - The Opportunity

3D Commerce = E Commerce enhanced with the use of 3D Models on any platform - including VR and AR

IKEA catalog uses augmented reality to give a virtual preview of furniture in a room - August 2013

IKEA Communications AB

Early Experience Shows
Increased customer engagement!
Strengthened brand loyalty!
Deeper product understanding!
More online sales!
Fewer returns!

So why is 3D Commerce taking so long to become widespread?

= $$$!
3D Commerce - Today’s Reality

I wish I had high quality, realistic 3D models for virtual promotional photoshoots!

Everyone defines their product data for sizes and colors differently - nothing is consistent!

I need the materials in my 3D models to look completely realistic!

Products don’t come with 3D data - and I can’t physically scan them all fast enough!

CAD tools don’t let me easily generate the data I need for E Commerce!

The green couch looks blue on some devices - lots of product returns are expensive!

Many 3D products on my ecommerce website first appear upside down! I have to hand-tune 1000s of models!

Complex retail pipeline with hundreds of companies and millions of products

Many friction points: tooling, technical and commercial

3D Commerce can’t reach industrial scale so...

Interoperability standards to the rescue!
Khronos 3D Commerce Initiative

Working Group Announced SIGGRAPH 2019

Creating specifications and guidelines to align the 3D asset workflow from product design through manufacturing and each stage of retail to end-user delivery platforms

Broad Industry Participation from tooling, retail, technology and platform companies
3D Commerce - Four Areas of Focus

**Asset Creation Guidelines**
For tools and product designers to create assets with consistent data to be used through the 3D Commerce pipeline

**Product Configuration**
Universal product configurability data and guidelines on how to drive consistent product display

**Metadata**
Structured metadata definitions and examples to consistently carry product information through the retail pipeline

**Viewer Validation and Certification**
Test models, reference viewer, display analysis tools and capability specifications to guarantee a consistent and accurate end user experience

**First Goals**
Industry cooperation to urgently develop solutions to address priority problem areas
3D Commerce Khronos Synergy

- 3D Asset Format
- WebGL
- WebGL
- OpenXR
- Vulkan
- OpenVX

Portable AR and VR Apps
High-performance cross-platform 3D graphics
Vision processing and inferencing for AR and scanning

Khronos 3D Commerce
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

3D Commerce

COLLADA

Guidelines for creating APIs to streamline system safety certification

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

Heterogeneous Communications between offload compute devices

Rendering for scientific visualization and data analytics
Khronos Open Standard Compute APIs

Increasing interest in parallel heterogeneous compute acceleration to combat the ‘End of Moore’s Law’
SYCL Single Source C++ Parallel Programming

SYCL is ideal for accelerating larger C++-based engines and applications

C++ Kernel Fusion can give better performance on complex apps and libs than hand-coding

Multiple SYCL libraries for vision and inferencing
SYCL-BLAS, SYCL-DNN, SYCL-Eigen, SYCL Parallel STL

Accelerated code passed into device OpenCL compilers

E.g. complex ML frameworks can be directly compiled and accelerated

Single application source file using STANDARD C++

TensorFlow

Multiple SYCL libraries for vision and inferencing
SYCL-BLAS, SYCL-DNN, SYCL-Eigen, SYCL Parallel STL

C++ templates and lambda functions separate host & device code

SyCL for OpenCL

OpenCL Devices

CPU

GPU

DSP

FPGA

Other technologies

CPU

Custom Processor

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

- **ComputeCpp:** Codeplay Software's v1.2.1 conformant implementation available to download today
- **triSYCL:** Open-source test-bed to experiment with the specification of the OpenCL SYCL C++ layer and to give feedback to Khronos
- **HipSYCL:** SYCL 1.2.1 implementation that builds upon NVIDIA CUDA/AMD HIP/ROCm
- **LLVM/clang SYCL Compiler:** Compiles C++-based SYCL source files into code for both CPU and a wide range of compute accelerators

**SYCL Enables Khronos to Influence ISO**

To enable standard C++ to (eventually) support heterogeneous compute

**Multiple Backend Support Coming**

SYCL beginning to be supported on low-level APIs in addition to OpenCL e.g. Vulkan and CUDA

http://sycl.tech

**Intel Adoption**

Intel's 'One API' Initiative uses SYCL


This work is licensed under a Creative Commons Attribution 4.0 International License
OpenVX Cross-Vendor Inferencing

OpenVX
A high-level graph-based abstraction for portable, efficient vision processing
Optimized OpenVX drivers created and shipped by processor vendors
Can be implemented on almost any hardware or processor
Graph can contain vision processing and NN nodes - enables global optimizations
Run-time graph execution can be almost completely autonomously from the host CPU

Performance comparable to hand-optimized, non-portable code
Real, complex applications on real, complex hardware
Much lower development effort than hand-optimized code

NNEF Translator converts NNEF representation into OpenVX Node Graphs

Hardware Implementations
OpenVX 1.3 Released October 2019

OpenVX 1.3 Feature Sets
Enables deployment flexibility while avoiding fragmentation
Implementations with one or more complete feature sets are conformant
- Baseline Graph Infrastructure (enables other Feature Sets)
- Default Vision Functions
- Enhanced Vision Functions (introduced in OpenVX 1.2)
- Neural Network Inferencing (including tensor objects)
  - NNEF Kernel import (including tensor objects)
  - Binary Images
- Safety Critical (reduced features for easier safety certification)


Open Source Prototype OpenVX 1.3 Conformance Test Suite
Finalization expected before the end of 2019
https://github.com/KhronosGroup/OpenVX-cts/tree/openvx_1.3

Open Source OpenVX Tutorial and Code Samples
https://github.com/rgiduthuri/openvx_tutorial

Open source OpenVX 1.3 for Raspberry Pi
Raspberry Pi 3 Model B with Raspbian OS
Automatic optimization of memory access patterns via tiling and chaining
Highly optimized kernels leveraging multimedia instruction set
Automatic parallelization for multicore CPUs and GPUs
Automatic merging of common kernel sequences

https://github.com/KhronosGroup/OpenVX-sample-impl/tree/openvx_1.3
Extending OpenVX with Custom Nodes

OpenVX/OpenCL Interop
- Provisional Extension
- Enables custom OpenCL acceleration to be invoked from OpenVX User Kernels
- Memory objects can be mapped or copied

Kernel/Graph Import
- Provisional Extension
- Defines container for executable or IR code
- Enables arbitrary code to be inserted as an OpenVX Node in a graph

OpenVX user-kernels can access command queue and cl_mem objects to asynchronously schedule OpenCL kernel execution

Application

OpenVX data objects

cl_mem buffers

OpenCL Command Queue

Map or copy OpenVX data objects into cl_mem buffers

Copy or export cl_mem buffers into OpenVX data objects

Fully asynchronous host-device operations during data exchange
Neural Network Exchange Formats

Before - Training and Inferencing Fragmentation

Every Inferencing Engine needs a custom importer from every Framework

Training Framework 1 -> Inference Engine 1
Training Framework 2 -> Inference Engine 2
Training Framework 3 -> Inference Engine 3

After - NN Training and Inferencing Interoperability

Neural Network Exchange Format

Training Framework 1 -> Inference Engine 1
Training Framework 2 -> Inference Engine 2
Training Framework 3 -> Inference Engine 3

Common Optimization and processing tools

Two Neural Network Exchange Format Initiatives

<table>
<thead>
<tr>
<th>NNEF</th>
<th>ONNX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Embedded Inferencing Import</td>
<td>Training Interchange</td>
</tr>
<tr>
<td>Defined Specification</td>
<td>Open Source Project</td>
</tr>
<tr>
<td>Multi-company Governance at Khronos</td>
<td>Initiated by Facebook &amp; Microsoft</td>
</tr>
<tr>
<td>Stability for hardware deployment</td>
<td>Software stack flexibility</td>
</tr>
</tbody>
</table>

ONNX and NNEF are Complementary
ONNX moves quickly to track authoring framework updates
NNEF provides a stable bridge from training into edge inferencing engines
NNEF and ONNX Industry Support

NNEF V1.0 released in August 2018
After positive industry feedback on Provisional Specification.
Maintenance update issued in September 2019
Extensions to V1.0 released for expanded functionality

ONNX 1.6 Released in September 2019
Introduced support for Quantization
ONNX Runtime being integrated with GPU inferencing engines such as NVIDIA TensorRT

NNEF Working Group Participants

ONNX Supporters
NNEF Tools Ecosystem

TensorFlow and TensorFlow Lite Import/Export

Caffe and Caffe2 Import/Export

ONNX Import/Export

Syntax Parser and Validator

OpenVX Ingestion and Execution

Live

Imminent

NNEF Model Zoo

Now available on GitHub. Useful for checking that ingested NNEF produces acceptable results on target system

NNEF adopts a rigorous approach to design lifecycle

Especially important for safety-critical or mission-critical applications in automotive, industrial and infrastructure markets

NNEF open source projects hosted on Khronos

NNEF GitHub repository under Apache 2.0

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

NNEF Model Zoo

Now available on GitHub. Useful for checking that ingested NNEF produces acceptable results on target system

NNEF adopts a rigorous approach to design lifecycle

Especially important for safety-critical or mission-critical applications in automotive, industrial and infrastructure markets

NNEF open source projects hosted on Khronos

NNEF GitHub repository under Apache 2.0

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

This work is licensed under a Creative Commons Attribution 4.0 International License
# Primary Machine Learning Compilers

<table>
<thead>
<tr>
<th>Import Formats</th>
<th>Caffe, Keras, MXNet, ONNX</th>
<th>TensorFlow Graph, MXNet, Paddle, Keras, ONNX</th>
<th>PyTorch, ONNX</th>
<th>TensorFlow Graph, PyTorch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Front-end / IR</td>
<td>NNVM / Relay IR</td>
<td>nGraph / Stripe IR</td>
<td>Glow Core / Glow IR</td>
<td>XLA HLO</td>
</tr>
<tr>
<td>Output</td>
<td>OpenCL, LLVM, CUDA, Metal</td>
<td>OpenCL, LLVM, CUDA</td>
<td>OpenCL LLVM</td>
<td>LLVM, TPU IR, XLA IR TensorFlow Lite / NNAPI (inc. HW accel)</td>
</tr>
</tbody>
</table>

- **Caffe, Keras, MXNet, ONNX**: Import formats include TensorFlow Graph, MXNet, Paddle, Keras, ONNX.
- **PyTorch, ONNX**: Front-end IR includes Glow Core / Glow IR.
- **XLA HLO**: Output includes LLVM, TPU IR, XLA IR TensorFlow Lite / NNAPI (inc. HW accel).

![Diagram of Primary Machine Learning Compilers]
ML Compiler Steps

**Embedded NN Compilers**
- CEVA Deep Neural Network (CDNN)
- Cadence Xtensa Neural Network Compiler (XNNC)

**ML Compiler Steps**

1. Import Trained Network Description
2. Apply graph-level optimizations e.g. node fusion, node lowering and memory tiling
3. Decompose to primitive instructions and emit programs for accelerated run-times

**Consistent Steps**

- Fast progress but still area of intense research
- If compiler optimizations are effective - hardware accelerator APIs can stay ‘simple’ and won’t need complex metacommands (combined primitive commands) like DirectML
OpenCL - Low-level Parallel Programming

- Low-level programming of heterogeneous parallel compute resources
  - One code tree can be executed on CPUs, GPUs, DSPs and FPGA ...

- OpenCL C or C++ language to write kernel programs to execute on any compute device
  - Platform Layer API - to query, select and initialize compute devices
  - Runtime API - to build and execute kernels programs on multiple devices

- The programmer gets to control:
  - What programs execute on what device
  - Where data is stored in various speed and size memories in the system
  - When programs are run, and what operations are dependent on earlier operations
OpenCL is Widely Deployed and Used

Hardware Implementations

Desktop Creative Apps

Linear Algebra Libraries

Parallel Computation Languages

Math and Physics Libraries

Vision and Imaging Libraries

Machine Learning Inferencing Compilers

Machine Learning Libraries

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

**OpenCL Extension Specs**
- Scratch-Pad Memory Management
- Vulkan / OpenCL Interop
- Extended Subgroups
- SPIR-V 1.4 ingestion for compiler efficiency
- SPIR-V Extended debug info

**Regular Maintenance Updates**
Regular updates for spec clarifications, formatting and bug fixes
[https://www.khronos.org/registry/OpenCL/](https://www.khronos.org/registry/OpenCL/)

**Integration of Extensions plus New Core functionality**

**Focus for OpenCL Next is ‘Deployment Flexibility’**
Flexible Profile enables embedded vendors to ship targeted functionality for their customers and be officially conformant

Repeat Cycle for next Core Specification

Target 2020 ‘OpenCL Next’
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

![Diagram showing the Clspv Compiler and OpenCL C Source connected to the OpenCL Host Code via the RUNTIME API Translator.]
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

- **3D Commerce**
- **COLLADA™**
- **OpenXR™**
- **Vulkan®**
- **OpenGL®**
- **OpenGL ES®**
- **WebGL®**
- **EGL®**
- **SPIR™**
- **SYCL™**
- **NNEF™**

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
Safety Critical GPU API Evolution

OpenGL SC 1.0 - 2005
Fixed function graphics safety critical subset

OpenGL SC 2.0 - April 2016
Programmable Shaders safety critical subset

OpenGL ES 1.0 - 2003
Fixed function graphics

OpenGL ES 2.0 - 2007
Programmable Shaders

Vulkan 1.0 - 2016
Explicit Graphics and Compute

New Generation Safety Critical APIs for Graphics, Compute and Display

Industry Need for GPU Acceleration APIs designed to ease system safety certification is increasing ISO 26262 / ASIL-D

Vulkan is Compelling Starting Point for SC GPU API Design
- Widely adopted, royalty-free open standard
- Low-level explicit API - smaller surface area than OpenGL
- Not burdened by debug functionality
- Very little internal state
- Well-defined thread behavior

Potential OpenCL SC work will leverage the deployment flexibility of ‘OpenCL Next’ to minimize API surface area

Khrnos Vulkan SC Working Group started work in February 2019

Clearly Definable Design Goals to Adapt Vulkan for SC
- Reduce driver size and complexity
- Offline pipeline creation, no dynamic display resolutions
- Deterministic Behavior
- No ignored parameters, static memory management, eliminate undefined behaviors
- Robust Error Handling
- Error callbacks so app can respond, Fatal error callbacks for fast recovery initiation

C API - MISRA C Compliance
Khronos Standards Immersive Computing

Vision and sensor processing - including neural network inferencing for machine learning

Download 3D object and scene data

Portable interaction with VR/AR sensor, haptic and display devices

High-performance, low-latency 3D Graphics
Khronos Proven Process and Organization

Open membership. Any company is welcome to join. One company one vote

Open specifications. ROYALTY-FREE through a strong, modern IP Framework

Any member, or non-member, can propose new standards initiatives

Software

Open Source Conformance Tests and Adopters Programs

Non-profit organization - Membership and Adopters fees cover expenses

Invest where strong industry momentum and relevance - let Darwinism rule!

Silicon
Khronos Ecosystem Engagement

Khronos Forums and Slack Channels

Contribute to open source specs, CTS, tools and ecosystem

Advisory Panels

Working Groups

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/

Khronos members under Khronos NDA and IP Framework participate and vote in working group meetings
https://www.khronos.org/members/

Open to all!
https://community.khronos.org/
www.khr.io/slack
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
- Enhance your company reputation as an industry leader through Khronos participation
Thank You and Resources

- Khronos is creating cutting-edge royalty-free open standards
  - For 3D, compute, inferencing gaming
- These slides and information on Khronos Standards
  - www.khronos.org
- Any company is welcome to join Khronos
  - https://www.khronos.org/members/
  - We warmly welcome members from Australia and Asia
- Dedicated developer resources
  - Khronos Developer Forum: https://community.khronos.org/
  - Khronos Developer Slack Channel: www.khr.io/slack
- We are happy to help answer any questions!
  - Neil Trevett, Khronos President: ntrevett@nvidia.com, @neilt3d
  - Khronos Developer Relations, Kris Rose: kris@khronos.org, @kristoferrose