Press Briefing
SIGGRAPH 2017
Neil Trevett | Khronos President
NVIDIA | VP Developer Ecosystem
ntrevett@nvidia.com | @neilt3d
www.khronos.org
Latest Khronos Updates

- **July 2017 - SIGGRAPH**
  - OpenGL 4.6 Released
  - Vulkan Interop, Momentum and Portability Initiative
  - Flash EOL and WebGL Momentum
  - glTF 2.0 Momentum

- **June 2017 - Web3D Conference**
  - glTF 2.0 Released

- **May 2017 - IWOCL and Embedded Vision Summit**
  - Khronos Vision and Inferencing Strategy for AR
  - OpenCL 2.2 Released and forward-looking roadmap discussions
  - OpenVX 1.2 Released - with Neural Net Extension and Safety Critical version

- **February 2017 - GDC**
  - WebGL 2.0 released
  - Vulkan VR and multi-GPU extensions
  - OpenXR announced for cross-platform VR and AR
OpenGL 4.6 Release

- OpenGL is 25 years old this year - still going strong!
  - Widely used and being evolved to meet customer needs
  - [https://khronos.org/registry/OpenGL/index_gl.php](https://khronos.org/registry/OpenGL/index_gl.php)

- SPIR-V Ingestion into core
  - Significant increase in shading language and
  - compiler tooling flexibility
  - Glslang open source compiler updated to support
  - OpenGL 4.6 functionality

- Brings multiple ARB/EXT extensions into core
  - Anisotropic filtering for improved texture quality - previously IP encumbered
  - Ability to turn off error checking for improved performance
  - Improved parameter handling to reduce CPU overhead when batching geometry rendering
  - Expanded pipeline queries
  - Offset clamp to suppress “light leak” artifacts when rendering shadows
  - Improved shader intrinsics for improved functionality and performance

- New extensions
  - Launch multiple shader compile threads to improve shader compile throughput
SPIR-V Ecosystem

SPIR-V
- Khronos defined and controlled cross-API intermediate language
- Native support for graphics and parallel constructs
  - 32-bit Word Stream
  - Extensible and easily parsed
- Retains data object and control flow information for effective code generation and translation

Khronos has open sourced these tools and translators

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

Third party kernel and shader languages

- GLSL
- HLSL
- glslang

Third party kernel and shader languages

- OpenCL C Front-end
- OpenCL C++ Front-end
- LLVM
- SPIR-V Cross
- SPIR-V (Dis)Assembler
- SPIR-V Validator
- Other Intermediate Forms

Other Intermediate Forms

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

IHV Driver Runtimes

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

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

Khronos has open sourced these tools and translators

© Copyright Khronos Group 2017 - Page 4
OpenGL Ecosystem

- OpenGL 4.6 Drivers immediately available
  - NVIDIA releasing beta OpenGL 4.6 drivers
  - https://developer.nvidia.com/opengl-driver

- OpenGL Conformance Tests in open source
  - Enabling for open source projects such as MESA

“The open sourcing of the OpenGL conformance test suite and ongoing work between Khronos and X.org will also allow for non-vendor led open source implementations to achieve conformance in the near future”

David Airlie, senior principal engineer at Red Hat, and developer on Mesa/X.org projects

- Vulkan Interop Extensions released with OpenGL 4.6

Vulkan Interop with OpenGL provides developers significant flexibility on how they use/transition between both APIs
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

DOOM on Vulkan port of id Tech 6

Doom’s Vulkan patch is a PC performance game-changer

Vulkan support in Unity 5.6

Vulkan support since V1.8

5.4 Preview Released on July 25th
Vulkan Momentum

Games Studios publicly confirming that work is ongoing on Vulkan Titles

#Vulkan Games on PC = 25
#DX12 Games on PC = 20
As of July 2017


All Major GPU Companies shipping Vulkan Drivers - for Desktop and Mobile Platforms

Mobile, Embedded and Console Platforms Supporting Vulkan

Android 7.0
Nintendo Switch
Android TV
Embedded Linux

http://www.game-debate.com/news/23312/up-for-debate-vulkan-vs-directx-12-which-is-the-better-graphics-api

© Copyright Khronos Group 2017 - Page 7
Market Demand for Universal 3D Portability

Community Outreach at GDC 2017
Create a hybrid Portability API?

Feedback - AVOID CREATING A FOURTH API!!!
Would need new specification, CTS, Documentation.
Additional developer learning curve.
A whole new specification to name, brand, promote.
Would INCREASE industry fragmentation
Vulkan Portability TSG Process

Open source project with identical goals already underway - come and help!
https://github.com/gfx-rs/ghx

Vulkan Portability Deliverables
1. Vulkan Subset Diff Spec
2. Vulkan Subset Development Layer
3. Vulkan Subset API Library over DX12/Metal
4. SPIRV-Cross Translator
5. Vulkan Subset Conformance Tests

Layers, APIs, Translators and Tests all to be developed and released in open source

API Overlap Analysis

Identify Vulkan features not directly mappable to DX12 and Metal

Possible proposals for Vulkan extensions for enhanced portability (and possibly Web robustness) sent to Vulkan WG

New Vulkan functionality may affect the overlap analysis

Expand/test existing open source SPIRV-Cross Tool

Metal Shading Language
HLSL
OpenCL and Vulkan

Single source C++ programming. Great for supporting C++ apps, libraries and frameworks

Industry working to bring Heterogeneous compute to standard ISO C++
C++17 Parallel STL hosted by Khronos
Executors - for scheduling work
“Managed pointers” or “channels” - for sharing data

OpenCL for DSPs
- Embedded imaging, vision and inferencing
- Flexible reduced precision
- Conformance without IEEE 32 Floating Point
- Explicit DMA

Help bring OpenCL-class compute to Vulkan

2011
OpenCL 1.2
OpenCL C Kernel Language

2015
OpenCL 2.1
SPIR-V in Core

2017
OpenCL 2.2
C++ Kernel Language
Clspv OpenCL C to Vulkan Compiler

- 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

- Uses new Vulkan extensions to support OpenCL C compute operations
  - VK_KHR_16bit_storage/SPV_KHR_16bit_storage
  - VK_KHR_variable_pointers/SPV_KHR_variable_pointers

- 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
Flash EOL and WebGL Momentum

Adobe has long played a leadership role in advancing interactivity and creative content – from video, to games and more – on the web. Where we’ve seen a need to push content and interactivity forward, we’ve innovated to meet those needs. Where a format didn’t exist, we invented one – such as with Flash and Shockwave. And over time, as the web evolved, these new formats were adopted by the community. In some cases, formed the basis for open standards, and became an essential part of the web.

But as open standards like HTML5, WebGL, and WebAssembly have matured over the past several years, most now provide many of the capabilities and functionalities that plugins pioneered and have become a viable alternative for content on the web. Over time, we’ve seen helper apps evolve to become plugins, and more recently, have seen many of these plugin capabilities get incorporated into open web standards.

Today, most browser vendors are integrating capabilities once provided by plugins directly into browsers and deprecating plugins.

WebGL - the Worlds MOST Portable 3D Platform

WebGL 1.0 brought mobile-class 3D to the Web (OpenGL ES 2.0)
WebGL 2.0 brings desktop-class graphics to your browser!! (OpenGL ES 3.0)
Expect a leap in interactivity and sophistication as WebGL 2.0 and WebAssembly become increasingly pervasive

http://caniuse.com/#feat=webgl

http://webglstats.com/webgl2
glTF - Cross-Platform 3D Asset Transmission

<table>
<thead>
<tr>
<th>Audio</th>
<th>Video</th>
<th>Images</th>
<th>3D</th>
</tr>
</thead>
<tbody>
<tr>
<td>MP3</td>
<td>H.264</td>
<td>JPEG</td>
<td>glTF</td>
</tr>
</tbody>
</table>

- napster
- YouTube™
- Facebook

New market opportunities for 3D content creation and deployment!

All glTF spec development on open GitHub:
https://github.com/KhronosGroup/glTF

---

OpenGL Transmission Format
Efficient transmission of 3D scenes and assets

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

glTF 1.0
Aimed at loading assets into WebGL apps
Uses GLSL for materials
Released December 2015

WebGL™

WebGL 2.0

Cool, portable materials
Rendering API independence
 Released June 2017
glTF Ecosystem Momentum

Tools
- Blender exporter for glTF 2.0 now in Beta!
- glTF Validator
- glTF Test
- glTF Test Models
- Tutorial
  - https://github.com/KhronosGroup/glTF-Tutorials

Convert | Optimize
- Visual Studio
  - glTF in Visual Studio Code
  - glTF-VCeCode extension

Blender exporter
- Unity
  - Unity Exporter
- PIXYZ SOFTWARE
- A-Frame
- Xeogl
- NVIDIA
- Autodesk
- FORGE
- PEX
- nvpro-pipeline

Apps & Engines
- Blender exporter for glTF 2.0 now in Beta!
- glTF 2.0 support complete
- glTF 2.0 support in progress

100,000+ glTF 2.0 models for download!

100,000+ glTF 2.0 models for download!

Microsoft using glTF 2.0 to bring 3D to Office!

- OBJ2GLTF
- COLLADA
- OBJ2GLTF
- COLLADA2GLTF
- Assimp
  - Open Asset Import Library

- Babylon.js
- Three.js
- Cesium
- Sketchfab
- Paint 3D
- View 3D

- NVPRO
- PEX

- Blender exporter
- COLLABA2GLTF
- glTF-Pipeline - Convert glTF 1.0->2.0
  - Drag and drop COLLADA -> glTF
    - http://cesiumjs.org/convertmodel.html
Blender glTF 2.0 Exporter

Open-source Beta Release!
https://github.com/KhronosGroup/gltf-Blender-Exporter
Google Draco glTF Extension!

- Library for compressing and decompressing 3D geometric meshes and point clouds
  - [https://github.com/google/draco](https://github.com/google/draco)
- 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
- Draco glTF extension is in progress and ready for feedback!
  - Draco designed and built for compression efficiency and speed - great fit with glTF!
  - [https://github.com/KhronosGroup/glTF/pull/874](https://github.com/KhronosGroup/glTF/pull/874)

Typical Draco compression ratios
Khronos APIs Connect Software to Silicon

Khronos is an International Industry Consortium of over 100 companies creating royalty-free, open standard APIs to enable software to access hardware acceleration for 3D graphics, Virtual and Augmented Reality, Parallel Computing, Neural Networks and Vision Processing.
Vulkan GDC Extensions

Vulkan 1.0.42 with new extension sets

Explicit Building Blocks for VR

Multiview
Efficiently render geometry to multiple surfaces
Each with its own viewing parameters
E.g. render stereo pairs or environment maps

Resource Sharing
Share memory and synchronization primitives
Works across process and instance boundaries

Descriptor Updates
Update resource references between
draw or compute dispatch calls
Efficiently repeatedly update fixed set of resources

Explicit Building Blocks for Multi-GPU

Works with NVIDIA SLI and AMD Crossfire
Does NOT support dGPU/iGPU

“Device Group” is set of physical devices
Acts as single logical device

Access separate GPUs only for explicit control
Memory allocation and binding resources
Command Buffer Recording/Submission
Synchronization

Enables a variety of operating modes
AFR (alternate frame), SFR (Sequential frame)
VR SLI Stereo view rendering

Shipped 1.0.42 beta drivers on the day the specifications were released PLUS building block Vulkan extensions for VRWorks on Maxwell and Pascal
Native Acceleration APIs for VR/AR

- Vision sensor(s)
- OpenVX
- OpenCL
- NNEF
- WebGL
- Vulkan
- OpenGL
- OpenGL ES
- glTF

**Tracking and Positioning**
- Geometric scene reconstruction
- Semantic scene understanding (Neural Networks)

**Download 3D augmentation object and scene data**

**Generate Low Latency 3D Content and Augmentations**

**Interact with sensor, haptic and display devices**

**VR/AR Application**
OpenXR - Solving AR/VR Fragmentation

Before OpenXR
VR Market Fragmentation

After OpenXR
Wide Interoperability of VR Apps and Devices

Device Discovery Device Events Sensor tracking Haptics HMD Parameters

Installable Drivers

Application Interface

Device Layer

VR App 1
VR App 2
VR App 3
VR App 4

VR Device 1
VR Device 2
VR Device 3
VR Device 4
VR Device 5

Proprietary Engine

Steam VR OSVR oculus Samsung Gear VR Daydream

Steam VR OSVR oculus Samsung Gear VR Daydream

Unity Unreal WebVR Proprietary Engine
OpenXR Working Group Members

Design work started in December 2016
Typically 12-18 months to develop a V1.0 specification
OpenXR and VR Run-times - a Win-Win

Access to any OpenXR Application

Any successful standard encourages and enables healthy industry competition

OpenXR will not replace VR run-times - or ‘outlaw’ existing interfaces

OpenXR will simply provide cross-vendor APIs that can be exposed by a runtime to access more apps and devices

Access to any OpenXR Device

OpenXR for portable AR AND VR apps and devices with initial focus on VR
Layered Ecosystem and VR in the Web

Applications

Native VR Apps

3D Web Apps

VR Web Apps

Web Middleware

Native Middleware

Browser APIs

Native APIs

Khronos APIs

VR and AR in the Web = ‘The Metaverse’
WebGL Timeline

2003 1.0
2004 1.1
2007 2.0
2012 3.0
2014 3.1
2015 3.2

- 32-bit integers and floats
- NPOT, 3D/depth textures
- Texture arrays
- Multiple Render Targets

Compute Shaders

- Tessellation and geometry shaders
- ASTC Texture Compression
- Floating point render targets
- Debug and robustness for security

Fixed function Pipeline
Programmable Vertex and fragment shaders

Driver Update
Silicon Update
Silicon Update
Driver Update
Silicon Update

Pervasive OpenGL ES 2.0
OpenGL and OpenGL ES ship on every desktop and mobile OS

Conformance Testing is vital for Cross-Platform Reliability
WebGL 2.0 conformance tests are very thorough 10x more tests than WebGL 1.0 tests

Epic’s Rivalry demo using full Unreal Engine 4 on mobile
https://www.youtube.com/watch?v=jRr-G95GdaM

WebGL Next using Vulkan Portability
possible Compute Shader Extension

GDC 2017

© Copyright Khronos Group 2017 - Page 25
**WebGL 2.0 Fast Building Momentum!**

- **Enhanced visual quality, performance, features**
  - Instancing Multiple render targets | Uniform buffers | Transform feedback
  - Multisampled Renderbuffers | 3D textures | NPOT textures
  - More texture formats | Occlusion queries | Vertex array objects
  - Sampler objects | Sync objects | Fragment depth | Primitive restart | ...

- **WebGL 2.0 now available in Chrome/Firefox!**
  - Chrome: Releasing to desktop and Android
  - Firefox: Released on all platforms
  - Edge/Safari: plan to ship WebGL 2.0

- **Desktop support at 60% - rising fast**

http://webglstats.com/webgl2

WebVR Running Over WebGL 2.0
WebGL and WASM (Web Assembly)

C/C++ Code with Native API calls
  ↓
Clang/LLVM
  ↓
Emscripten
  ↓
WASM Modules + JS Glue Code

For calling Web APIs in WASM Modules there needs to be:
1) a native API for WASM source code
2) a JS API for run-time invocation
3) a simple mapping between them to minimize JS Glue Code

JavaScript

| WASM modules access Web APIs (like WebGL) through JS call-outs |
| JavaScript calls WASM Modules |
| JavaScript WebAPI |

WebGL Next will need:
1) a portable native APIs that can run on any system
2) a JS binding of that API for use via JavaScript
3) a simple mapping between the two

Mozilla’s Obsidian WebGL Next Proposal:
1) defines as Vulkan Subset as a portable native API
2) binds the Vulkan subset to JavaScript
3) for a simple mapping between the two
Strong glTF Industry Momentum

Please let us know if your company logo is not here and you would like it added!

Publicly Stated Support for glTF
What’s New in glTF 2.0

• Physically Based Rendering (PBR) material definitions
  - Material information stored in textures

• Graphics API neutral
  - Proven by engine implementations using WebGL, Vulkan and Direct3D
  - GLSL materials moved to extension for existing content and specialized use cases

• Morph Targets
  - Enhanced animation system

• Improvements
  - Binary glTF in core
  - Dozens of refinements for enhanced performance and a tighter, clearer specification

• Download the glTF 2.0 Reference Guide!
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**
  - Textures

- Geometry
- Texture based PBR materials
glTF 2.0 PBR - Consistency Across Engines

- glTF on WebGL Reference Implementation
  
  [Link](http://github.khronos.org/glTF-WebGL-PBR/)

- glTF Model on Laugh Engine over Vulkan
  
  [Link](https://github.com/jian-ru/laugh_engine)

- glTF Model running on Metal PBR Renderer
  
  [Link](https://twitter.com/warrenm/status/891558755657175040)

- glTF on View 3D over DX12
  
Vision and Neural Net Acceleration Challenge

Neural Net Training Frameworks

Trained Networks

Vision/Artificial Intelligence (AI) Applications

Vision and Neural Net Inferencing Runtime

Embedded/Mobile Vision/Inferencing Hardware

Desktop and Cloud Hardware
NNEF - Solving Neural Net Fragmentation

Before NNEF - NN Training and Inferencing Fragmentation

Caffe  
TensorFlow  
Microsoft CNTK  
theano

NN Authoring Framework 1
NN Authoring Framework 2
NN Authoring Framework 3

Inference Engine 1
Inference Engine 2
Inference Engine 3

Every Tool Needs an Exporter to Every Accelerator

With NNEF - NN Training and Inferencing Interoperability

Caffe  
TensorFlow  
Microsoft CNTK  
theano

NN Authoring Framework 1
NN Authoring Framework 2
NN Authoring Framework 3

Inference Engine 1
Inference Engine 2
Inference Engine 3

NNEF is a Cross-vendor Neural Net file format
Encapsulates network formal semantics, structure, data formats, commonly-used operations (such as convolution, pooling, normalization, etc.)
NNEF Status and Roadmap

- V1.0 is under development, will soon start to solicit industry comments
  - NNEF will form an advisory panel, you are invited today to participate

- First version will focus on embedded inference
  - But will allow training as secondary goal

- Support ‘First cut’ range of network types
  - Field is moving very fast but we aim to keep up with developments

- NNEF Roadmap
  - Track development of new network types
  - Address a wider range of applications (outside vision apps)
  - Increase the expressive power of the format
OpenVX

Wide range of vision hardware architectures
OpenVX provides a high-level Graph-based abstraction
->
Enables Graph-level optimizations!
Can be implemented on almost any hardware or processor!
->
Portable, Efficient Vision Processing!

Power Efficiency

Computation Flexibility

- Dedicated Hardware
- Vision DSPs
- GPU Compute
- Multi-core CPU

Vision Engines
Middleware
Applications
Software Portability

Vision Processing Graph

Vision Node

Vision Node

Vision Node

Vision Node
OpenVX Evolution

OpenVX 1.0
Spec released October 2014

Conformant Implementations
AMD
Cadence
CEVA
Intel
Imagination
NVIDIA
Socionext
Synopsys
VeriSilicon

OpenVX 1.1
Spec released May 2016

Conformant Implementations
Cadence
Imagination
Texas Instruments

AMD OpenVX Tools
- Open source, highly optimized for x86 CPU and OpenCL for GPU
- “Graph Optimizer” looks at entire processing pipeline and removes, replaces, merges functions to improve performance and bandwidth
- Scripting for rapid prototyping, without re-compiling, at production performance levels
http://gpuopen.com/compute-product/amd-openvx/

New Functionality
Expanded Nodes Functionality
Enhanced Graph Framework

OpenVX Tools
- Conditional node execution
- Feature detection
- Classification operators
- Expanded imaging operations

Extensions
Neural Network Acceleration
Graph Save and Restore
16-bit image operation

Safety Critical
OpenVX 1.1 SC for safety-certifiable systems

New Functionality
Under Discussion
NNEF Import
Programmable user kernels with accelerator offload

OpenVX Roadmap

OpenVX 1.2
Spec released May 2017
OpenVX - Graph-Level Abstraction

- OpenVX developers express a graph of image operations (‘Nodes’) - Using a C API
- Nodes can be executed on any hardware or processor coded in any language - Implementers can optimize under the high-level graph abstraction
- Graphs are the key to run-time power and performance optimizations - E.g. Node fusion, tiled graph processing for cache efficiency etc.

OpenVX Nodes

OpenVX Graph

Feature Extraction Example Graph
OpenVX 1.2 and Neural Net Extension

- Convolution Neural Network topologies can be represented as OpenVX graphs
  - Layers are represented as OpenVX nodes
  - Layers connected by multi-dimensional tensors objects
  - Layer types include convolution, activation, pooling, fully-connected, soft-max
  - CNN nodes can be mixed with traditional vision nodes

- Import/Export Extension
  - Efficient handling of network Weights/Biases or complete networks

- OpenVX will be able to import NNEF files into OpenVX Neural Nets
OpenCL - Low-level Parallel Programming

- Low-level, explicit 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 2.2 Released in May 2017

- **OpenCL 1.2** (2011)
  - Becomes industry baseline for heterogeneous parallel computing

- **OpenCL 2.0** (2013)
  - Enables new class of hardware
  - SVM
  - Generic Addresses
  - On-device dispatch

- **OpenCL 2.1 SPIR-V 1.0** (2015)
  - SPIR-V in Core
  - Kernel Language Flexibility

- **OpenCL 2.2 SPIR-V 1.2** (2017)
  - OpenCL C++ Kernel Language
  - Static subset of C++14
  - Templates and Lambdas
  - **SPIR-V 1.2**
  - OpenCL C++ support
  - **Pipes**
    - Efficient device-scope communication between kernels
  - **Code Generation Optimizations**
    - Specialization constants at SPIR-V compilation time
    - Constructors and destructors of program scope global objects
    - User callbacks can be set at program release time

- [https://www.khronos.org/opencl/](https://www.khronos.org/opencl/)
OpenCL Ecosystem

Hardware Implementers
Desktop/Mobile/Embedded/FPGA

OpenCL 2.2 - Top to Bottom C++

100s of applications using OpenCL acceleration
Rendering, visualization, video editing, simulation, image processing, vision and neural network inferencing

SYCL
Single Source C++ Programming

OpenCL
Core API and Language Specs

SPIR
Portable Kernel Intermediate Language

© Copyright Khronos Group 2017 - Page 41
OpenCL as Language/Library Backend

- Caffe: C++ based neural network framework
- Halide: Language for image processing and computational photography
- C++ AMP: Accelerated Massive Parallelism with Microsoft Visual C++
- MulticoreWare: Open source project on Bitbucket
- SYCL: Single Source C++ Programming for OpenCL
- aparapi: Java language extensions for parallelism
- OpenCV: Vision processing open source project
- OpenACC: Compiler directives for Fortran, C and C++
- TensorFlow: Open source software library for machine learning

Hundreds of languages, frameworks and projects using OpenCL to access vendor-optimized, heterogeneous compute runtimes

Over 4,000 GitHub repositories using OpenCL: tools, applications, libraries, languages - up from 2,000 two years ago
SYCL Ecosystem

- Single-source heterogeneous programming using STANDARD C++
  - Use C++ templates and lambda functions for host & device code
  - Layered over OpenCL
- Fast and powerful path for bring C++ apps and libraries to OpenCL
  - C++ Kernel Fusion - better performance on complex software than hand-coding
  - Halide, Eigen, Boost.Compute, SYCLBLAS, SYCL Eigen, SYCL TensorFlow, SYCL GTX
  - triSYCL, ComputeCpp, VisionCpp, ComputeCpp SDK ...
- More information at http://sycl.tech

Developer Choice

The development of the two specifications are aligned so code can be easily shared between the two approaches

C++ Kernel Language
- Low Level Control
  - ‘GPGPU’-style separation of device-side kernel source code and host code

Single-source C++
- Programmer Familiarity
  - Approach also taken by C++ AMP and OpenMP
OpenCL Evolution Discussions

Single source C++ programming. Great for supporting C++ apps, libraries and frameworks

SYCL
SYCL 1.2
C++11 Single source programming
2011
OpenCL

SYCL
SYCL 2.2
C++14 Single source programming
2015
OpenCL

SYCL
SYCL 2.2
C++ Kernel Language
2017
OpenCL

Industry working to bring Heterogeneous compute to standard ISO C++
C++17 Parallel STL hosted by Khronos
Executors - for scheduling work
“Managed pointers” or “channels” - for sharing data
Hoping to target C++ 20 but timescales are tight

OpenCL 1.2++?
Focus on embedded imaging, vision and inferencing
Make FP32 optional for DSPs and general power efficiency

‘OpenCL-V’
Converge Vulkan and OpenCL
Layered Vision/ Neural Net Ecosystem

Implementers may use OpenCL or Vulkan to implement OpenVX nodes on programmable processors.

OpenVX enables the graph to be extended to include hardware architectures that don’t support programmable APIs.

And then developers can use OpenVX to enable a developer to easily connect those nodes into a graph.

The OpenVX graph abstraction enables implementers to optimize execution across diverse hardware architectures for optimal power and performance.
Safety Critical APIs

New Generation APIs for safety certifiable vision, graphics and compute e.g. ISO 26262 and DO-178B/C

OpenGL SC 1.0 - 2005
Fixed function graphics subset

OpenGL ES 1.0 - 2003
Fixed function graphics

OpenGL SC 2.0 - April 2016
Shader programmable pipeline subset

OpenGL ES 2.0 - 2007
Shader programmable pipeline

OpenVX SC 1.1 Released 1st May 2017
Restricted "deployment" implementation executes on the target hardware by reading the binary format and executing the pre-compiled graphs

Khronos SCAP
'Safety Critical Advisory Panel'
Guidelines for designing APIs that ease system certification.
Open to Khronos member AND industry experts

OpenCL SC TSG Formed
Working on OpenCL SC 1.2
Eliminate Undefined Behavior
Eliminate Callback Functions
Static Pool of Event Objects
OpenVX SC - Safety Critical Vision Processing

- OpenVX 1.1 - based on OpenVX 1.1 main specification
  - Enhanced determinism
  - Specification identifies and numbers requirements
- MISRA C clean per KlocWorks v10
- Divides functionality into “development” and “deployment” feature sets
  - Adds requirement to support import/export extension

Diagram:
- OpenVX SC Development Feature Set (Create Graph)
- Verify
- Export
- Binary format
- Import
- OpenVX SC Deployment Feature Set (Execute Graph)
- Implementation-dependent format
- No graph creation API
- Entire graph creation API