APIs for Accelerating Embedded Vision and Inferencing

Industry Overview of Options and Trade-offs

Neil Trevett
Khronos President
NVIDIA VP Developer Ecosystems

ntrevett@nvidia.com | @neilt3d
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

Khronos is 20 years old this year!
Embedded Vision and Inferencing Acceleration

Networks trained on high-end desktop and cloud systems

Neural Network Training

Training Data

Trained Networks

Compilation

Ingestion

Compiled Code

Inferencing Engine

Application Code

Vision Library

Hardware Acceleration APIs

Sensor Data

Diverse Embedded Hardware (GPUs, DSPs, FPGAs)

Apps link to compiled code or inferencing library

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
Khronos Compute Acceleration Standards

Higher-level APIs
Streamlined development and performance portability

Lower-level APIs
Direct Hardware Control

- GPU rendering + compute acceleration
- Intermediate Representation (IR) supporting parallel execution and graphics
- Heterogeneous compute acceleration

- Single source C++ programming with compute acceleration
- Graph-based vision and inferencing acceleration

Increasing industry interest in parallel compute acceleration to combat the ‘End of Moore’s Law’
Embedded Vision and Inferencing Acceleration

Networks trained on high-end desktop and cloud systems

Neural Network Training

Trained Networks

Compilable Code

Inferencing Engine

Application Code

Vision Library

Hardware Acceleration APIs

Sensor Data

Diverse Embedded Hardware (GPUs, DSPs, FPGAs)

Apps link to compiled code or inferencing library

cuDNN, CUDA, Vulkan, OpenCL, SPIR, SYCL, OpenVX, OpenCV

This work is licensed under a Creative Commons Attribution 4.0 International License
Neural Network Exchange Formats

Before - Training and Inferencing Fragmentation

Training Framework 1 ➔ Inference Engine 1
Training Framework 2 ➔ Inference Engine 2
Training Framework 3 ➔ Inference Engine 3

Every Inferencing Engine needs a custom importer from every Framework

After - NN Training and Inferencing Interoperability

Training Framework 1 ➔ Neural Network Exchange Format ➔ Inference Engine 1
Training Framework 2 ➔ Neural Network Exchange Format ➔ Inference Engine 2
Training Framework 3 ➔ Neural Network Exchange Format ➔ 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

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

© The Khronos® Group Inc. 2020 - Page 7
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

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
SYCL Single Source C++ Parallel Programming

SYCL-Blas, SYCL-DNN, SYCL-Eigen, SYCL Parallel STL

C++ Libraries → Standard C++ Application Code → ML Frameworks

C++ Template Libraries

SYCL Compiler for OpenCL

CPU Compiler

OpenCL

CPU

CPU

GPU

FPGA

DSP

AI/Tensor HW

Custom Hardware

Accelerated code passed into device OpenCL compilers

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

Complex ML frameworks can be directly compiled and accelerated

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

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

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

© The Khronos® Group Inc. 2020 - Page 10
SYCL Implementations

SYCL, OpenCL and SPIR-V, as open industry standards, enable flexible integration and deployment of multiple acceleration technologies.

SYCL Source Code

SYCL enables Khronos to influence ISO C++ to (eventually) support heterogeneous compute.

Multiple Backends in Development
SYCL beginning to be supported on multiple low-level APIs in addition to OpenCL e.g. ROCm and CUDA
For more information: http://sycl.tech
OpenVX Cross-Vendor Inferencing

OpenVX
High-level graph-based abstraction for portable, efficient vision processing
Graph can contain vision processing and NN nodes - enables global optimizations
Optimized OpenVX drivers created and shipped by processor vendors
Implementable on almost any hardware or processor with performance portability
Run-time graph execution need very little host CPU interaction

Performance comparable to hand-optimized, non-portable code
Real, complex applications on real, complex hardware
Much lower development effort than hand-optimized code
OpenVX 1.3 Released October 2019

Functionality Consolidation into Core 1.3
Neural Net Extension, NNEF Kernel Import, Safety Critical etc.

Deployment Flexibility through Feature Sets
Conformant Implementations ship one or more complete feature sets
Enables market-focused Implementations
- 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 Conformance Test Suite
https://github.com/KhronosGroup/OpenVX-cts/tree/openvx_1.3
OpenVX Open Source Implementation & Apps

Open Source OpenVX Tutorial and Code Samples
https://github.com/rgiduthuri/openvx_tutorial
https://github.com/KhronosGroup/openvx-samples

Open Source OpenVX 1.3 for Raspberry Pi
Raspberry Pi 3 Model B with Raspbian OS
Memory access optimization via tiling/chaining
Highly optimized kernels on 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
### OpenVX and OpenCV are Complementary

<table>
<thead>
<tr>
<th><strong>Implementation</strong></th>
<th>Community driven open source library</th>
<th>Callable API implemented, optimized and shipped by hardware vendors</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Conformance</strong></td>
<td>Extensive OpenCV Test Suite but no formal Adopters program</td>
<td>Implementations must pass Khronos Conformance Test Suite to use trademark</td>
</tr>
<tr>
<td><strong>Scope</strong></td>
<td>100s of imaging and vision functions Multiple camera APIs/interfaces</td>
<td>Tight focus on dozens of core hardware accelerated functions plus extensions and accelerated custom nodes. Uses external camera drivers</td>
</tr>
<tr>
<td><strong>Inferencing</strong></td>
<td>Deep Neural Network module to construct networks from layers for forward pass computations only. Import from ONNX, TensorFlow, Torch, Caffe</td>
<td>Neural Network layers and operations represented directly in the OpenVX Graph. NNEF direct import, ONNX through NNEF converter</td>
</tr>
<tr>
<td><strong>Acceleration</strong></td>
<td>OpenCV 3.0 Transparent API (or T-API) enables function offload to OpenCL devices</td>
<td>Implementation free to use any underlying API such as OpenCL. Can use OpenCL for Custom Nodes</td>
</tr>
<tr>
<td><strong>Efficiency</strong></td>
<td>OpenCV 4.0 G-API graph model for some filters, arithmetic/binary operations, and well-defined geometrical transformations</td>
<td>Graph-based execution of all Nodes. Optimizable computation and data transfer</td>
</tr>
<tr>
<td><strong>IP Protection</strong></td>
<td>None. Source code licensed under BSD. Some modules require royalties/licensing</td>
<td>Protected under Khronos IP Framework - Khronos members agree not to assert patents against API when used in Conformant implementations</td>
</tr>
</tbody>
</table>
# 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</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>TensorFlow Lite / NNAPI</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(inc. HW accel)</td>
</tr>
</tbody>
</table>

![Diagram showing the relationships between different machine learning compilers and their import formats, front-end/IR, and output formats.](image)
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

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

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

Consistent 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
OpenCL - Low-level Parallel Programming

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

- 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

Desktop Creative Apps
- Adobe
- otoy
- Modo
- CHAOGROUP
- CyberLink
- Autodesk
- GIMP
- ArcSoft
- Blackmagic Design
- Capture One
- Blender
- Sony
- SideFX

Parallel Languages
- OpenACC
- SYCL
- aparapi
- PyOpenCL

Linear Algebra Libraries
- SYCL-BLAS
- ViennaCL
- CLBlast

Machine Learning Libraries and Frameworks
- OpenVINO™
- MACE
- SYCL-NN
- Caffe
- Halide
- Acuity
- VisionCpp
- OpenVX

Vision and Imaging Libraries
- OpenCV
- NNAPI
- CIDNN
- OpenVX

Machine Learning Compilers
- plaidML
- tvm
- Wolfram Mathematica

Math and Physics Libraries
- GNU Octave
- Matlab
- ArrayFire
- C++, Fortran

Hardware Implementations

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

**OpenCL Extensions**
- Asynchronous Copies for DSPs
- Extended Subgroups
- Extended Versioning

**Expanding Language Ecosystem**
- Tighter LLVM integration and cooperation
- Open source C++ for OpenCL Kernel Language
- SPIR-V 1.4 ingestion for compiler efficiency
- SPIR-V Extended debug info

**Improving Software Ecosystem**
- Tool, libraries, ICD Loader
- Regular Maintenance Updates
  - Spec clarifications, formatting and bug fixes
  - [https://www.khronos.org/registry/OpenCL/](https://www.khronos.org/registry/OpenCL/)

**Extension Pipeline**
- Vulkan/OpenCL Interop
- Recordable Command buffers
- Unique Device Ids
- ML Primitives
- Device Topology
- Unified Shared Memory

**Repeat The Cycle**

**Target 2020**
- OpenCL Next

**Focus on ‘Deployment Flexibility’ to reach more processors and platforms**
Engaging with the Khronos Ecosystem

- **Working Groups**: Contribute to open source specs, CTS, tools and ecosystem.
  - $0
    - Open to all!
      - [https://community.khronos.org/](https://community.khronos.org/)
      - [www.khr.io/slack](http://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
  - $0
    - [https://github.com/KhronosGroup](https://github.com/KhronosGroup)
- **Khronos Forums and Slack Channels**: Invited Advisors under the Khronos NDA and IP Framework can comment and contribute to requirements and draft specifications
  - $0
    - [https://www.khronos.org/advisors/](https://www.khronos.org/advisors/)
- **Working Groups**: Khronos members under Khronos NDA and IP Framework participate and vote in working group meetings. Starts at $3.5K/yr.
  - $0
    - [https://www.khronos.org/members/](https://www.khronos.org/members/)
- **Any member or non-member can propose a new Khronos initiative**
Safety Critical GPU API Evolution

OpenGL ES 1.0 - 2003
Fixed function graphics

OpenGL ES 2.0 - 2007
Programmable Shaders

OpenCL

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

Vulkan 1.0 - 2016
Explicit Graphics and Compute

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

 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

Khirons Vulkan SC Working Group started work in February 2019

This work is licensed under a Creative Commons Attribution 4.0 International License
Need for New Camera Control API Standard?

- Khronos suspended work on OpenKCam standard several years ago
  - Mobile market went proprietary - but embedded market has different needs

- OpenKCAM was aiming at advanced control of ISP and camera with cross-platform portability
  - Generate sophisticated image stream for advanced imaging & vision apps
  - Portable access to growing sensor diversity: e.g. depth sensors and sensor arrays
  - Cross sensor synch: e.g. synch of multiple camera and MEMS sensors
  - Advanced, high-frequency per-frame burst control of camera/sensor: e.g. ROI
  - Multiple input, output re-circulating streams with RAW, Bayer or YUV Processing
Thank You!

- Khronos is creating cutting-edge royalty-free open standards
  - For 3D, compute, vision, inferencing acceleration
- Information on Khronos Standards: [www.khronos.org](https://www.khronos.org)
- Any company is welcome to join Khronos: [https://www.khronos.org/members/](https://www.khronos.org/members/)
- 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
- Gather industry requirements for future open standards
- Draft Specifications Confidential to Khronos members
- Publicly Release Specifications and Conformance Tests
- 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