Welcome to

**DesignCon® 2022**
WHERE THE CHIP MEETS THE BOARD

**Conference**
April 5 – 7, 2022

**Expo**
April 6 – 7, 2022

Santa Clara Convention Center
Open Standard Acceleration APIs for Safety-Critical Graphics, Vision and Compute

Neil Trevett, Khronos President
Neil Trevett

Khronos President, NVIDIA VP Developer Ecosystems
ntrevett@nvidia.com | www.khronos.org | @neilt3d

Neil is the elected President of the Khronos Group, where he oversees the development of open standards for graphics and heterogeneous parallel computation, including for safety critical markets. By day, Neil is VP Developer Ecosystems at NVIDIA, where he works to enable applications to take advantage of GPUs.
Topics

Introduction to Khronos and open standard APIs for graphics and compute acceleration
Khronos safety-critical APIs including Vulkan SC and the new SYCL SC Exploratory Forum
The new Khronos AUTOSAR Liaison
The new Khronos EMVA Camera Working Group
Details on how to get involved!
Need for Embedded Computing Standards

Increasing Sensor Richness
Camera arrays and depth sensors such as Lidar fed into sophisticated processing including inferencing

Multiple Sensors Per System
Synchronization, coordination and processing of diverse sensor streams becomes essential

Cost and time to integrate and utilize sensors, GPUs and processors in embedded markets has become a major constraint on innovation and efficiency

Increasing Sensor Processing Demands
Sensor outputs need to be flexibly and efficiently generated and streamed into diverse acceleration processors

Proprietary APIs Hinder Innovation
Vendor-specific APIs to control cameras, sensors, close-to-sensor ISPs and processors prevent rapid integration of new technologies
Khronos Connects Software to Silicon

Open, royalty-free interoperability standards to harness the power of GPU, XR and multiprocessor hardware

3D graphics, augmented and virtual reality, parallel programming, inferencing and vision acceleration

Non-profit, member-driven standards organization, open to any company

Proven multi-company governance and Intellectual Property Framework

Founded in 2000

~ 200 Members | ~ 40% US, 30% Europe, 30% Asia
Khronos Active Standards

3D Graphics
Desktop, Mobile and Web

3D Assets
Authoring and Delivery

Portable XR
Augmented and Virtual Reality

Parallel Computation
Vision, Inferencing, Machine Learning

Safety Critical APIs

[Logos of various standards and APIs]
Khronos Compute Acceleration Standards

Increasing industry interest in parallel compute acceleration to combat the ‘End of Moore’s Law’

SYCL and SPIR were originally OpenCL sub projects

Higher-level Languages and APIs
- Streamlined development and performance portability
- Single source C++ programming with compute acceleration
- Graph-based vision and inferencing acceleration

Vulkan
- GPU rendering + compute acceleration

OpenVX
- Intermediated Representation (IR) supporting parallel execution and graphics

OpenCL
- Heterogeneous compute acceleration

Lower-level Languages and APIs
- Direct Hardware Control

GPU
- Intermediate Representation (IR) supporting parallel execution and graphics

CPU
- OpenVX
- OpenCL

FPGA
- Vulkan

DSP
- Intermediate Representation

AI/Tensor HW
- Intermediate Representation

Custom Hardware
Vulkan: Performance, Predictability, Portability

Vulkan is the only open standard modern GPU API
Not controlled by and tied to a specific platform

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

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

A Graphics API

Application
Single thread per context

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

Multiple Front-end Compilers
GLSL, HLSL etc.

Thin Driver
Explicit GPU Control

GPU

Application

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

Loadable debug and validation layers

Multiple graphics, command and DMA queues

A GPU API
SPIR-V enables a rich ecosystem of languages and compilers to target low-level APIs such as Vulkan and OpenCL, including deployment flexibility: e.g., running OpenCL kernels on Vulkan.
OpenCL – Low-level Parallel Programming

Programming and Runtime Framework for Application Acceleration
Offload compute-intensive kernels onto parallel heterogeneous processors
CPUs, GPUs, DSPs, FPGAs, Tensor Processors
OpenCL C or C++ kernel languages

Platform Layer API
Query, select and initialize compute devices

Runtime API
Build and execute kernels programs on multiple devices

Explicit Application Control
Which programs execute on what device
Where data is stored in memories in the system
When programs are run, and what operations are dependent on earlier operations

Complements GPU-only APIs
Simpler programming model
Relatively lightweight run-time
More language flexibility, e.g., pointers
Rigorously defined numeric precision
OpenCL 3.0 Processor Adoption

Product Conformance Status
https://www.khronos.org/conformance/adopters/conformant-products/opencl

OpenCL 3.0 Adopters

- arm
- Google
- Intel
- NVIDIA
- Qualcomm
- Codeplay
- Imagination
- Microsoft
- QNX
- VeriSilicon

OpenCL 3.0 Adopters
Shipping Conformant Implementations
## ML Compiler Steps

<table>
<thead>
<tr>
<th>Import Formats</th>
<th>Front-end / IR</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Caffe, Keras, MXNet, ONNX</td>
<td>NNVM / Relay IR</td>
<td>OpenCL, LLVM, CUDA, Metal</td>
</tr>
<tr>
<td>TensorFlow Graph, MXNet, PaddlePaddle, Keras, ONNX</td>
<td>nGraph / Stripe IR</td>
<td>OpenCL, LLVM, CUDA</td>
</tr>
<tr>
<td>PyTorch, ONNX</td>
<td>Glow Core / Glow IR</td>
<td>OpenCL LLVM</td>
</tr>
<tr>
<td>TensorFlow Graph, PyTorch, ONNX</td>
<td></td>
<td>LLVM, TPU IR, XLA IR</td>
</tr>
</tbody>
</table>

### Consistent Steps
1. **Import Trained Network Description**
2. **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 (e.g., combined primitive commands like DirectML)

### Embedded NN Compilers
- CEVA Deep Neural Network (CDNN)
- Cadence Xtensa Neural Network Compiler (XNNC)
SYCL Single Source C++ Parallel Programming

- **C++ Libraries**
  - Standard C++ Application Code
  - ML Frameworks
  - SYCL Compiler
  - CPU Compiler
  - Other Backends
  - SYCL Accelerates C++-based engines and applications with performance portability
  - SYCL 2020 Launch February 2021
  - Closer alignment with C++17
  - Smaller code size, faster performance
  - New Features:
    - Unified Shared Memory
    - Parallel Reductions
    - Subgroup Operations
    - Class template Argument Deduction

- **C++ Template Libraries**
  - CPU
  - GPU
  - FPGA
  - DSP
  - AI/Tensor HW
  - Custom Hardware

- **GPU**
  - One-MKL
  - One-DNN
  - OneDPC
  - SYCL-BLAS
  - SYCL-Eigen
  - SYCL-DNN
  - SYCL Parallel STL

- **FPGA**

- **DSP**

- **Custom Hardware**

- **CPU**

- **OpenCL**
  - Complex ML frameworks can be directly compiled and accelerated
  - C++ templates and lambda functions separate host & accelerated device code

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

- **Accelerated code passed into device OpenCL compilers**
SYCL Implementations in Development

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

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
The Origin of OpenVX

Engines and Applications

3D Graphics API
Driver

Vulkan

GPU

Driver Model
An open API standard enables multiple silicon vendors to ship drivers with their silicon. Silicon vendors can aggressively optimize drivers for their own silicon architecture.
OpenVX is the industry’s only API standard enabling portable access to vendor-optimized vision drivers.

OpenVX
Vision API
Driver

High-level Abstraction
3D graphics is always accelerated by a GPU, so a low-level GPU API can provide cross-vendor portability.

BUT Vision processing is accelerated by a wide variety of hardware architectures.

SO OpenVX needs a higher-level graph abstraction to enable optimized cross-vendor drivers.

Vision Processing Graph

Vision Node

Vision Node

Vision Node

Vision Node

Vision Node

Vision Node
OpenVX Cross-Vendor Vision and Inferencing

High-level graph-based abstraction for portable, efficient vision processing

- Optimized OpenVX drivers created, optimized and shipped by processor vendors
- Implementable on almost any hardware or processor with performance portability
- Graph can contain vision processing and NN nodes for global optimization
- Run-time graph execution need very little host CPU interaction

OpenVX Graph

Vision Node

CNN Nodes

Vision Node

Downstream Application Processing

Native Camera Control

Open-Source Convertors

ONNX
TensorFlow
TensorFlow Lite
Caffe
Caffe2

NNEF Import converts a trained Neural Network into OpenVX Graph
Layers are represented as OpenVX nodes

Vendors optimize and ship drivers for their platform
Full list of conformant OpenVX implementations here:
https://www.khronos.org/conformance/adopters/conformant-products/openvx

https://github.com/KhronosGroup/NNEF-Tools
Embedded Vision and Inferencing

Networks trained on high-end desktop and cloud systems

Applications link to compiled inferencing code or call vision/inferencing API

Diverse Embedded Hardware
- Multi-core CPUs, GPUs
- DSPs, FPGAs, Tensor Cores
  * Vulkan only runs on GPUs

Open industry standards, enable flexible integration and deployment of multiple acceleration technologies

Sensor Data

Neural Network Training
- Training Data

Trained Networks
- Compilation
- Ingestion

Compiled Code
- Vision / Inferencing Engine
- C++ Application Code

Hardware Acceleration APIs
- OpenCL
- Vulkan

* Vulkan only runs on GPUs

Applications link to compiled inferencing code or call vision/inferencing API

Networks trained on high-end desktop and cloud systems

Diverse Embedded Hardware
- Multi-core CPUs, GPUs
- DSPs, FPGAs, Tensor Cores
  * Vulkan only runs on GPUs

Open industry standards, enable flexible integration and deployment of multiple acceleration technologies
Growing Need for APIs for Functional Safety

Demand for advanced GPU-accelerated graphics and compute is growing in an increasing number of industries where safety is paramount, such as automotive, autonomy, avionics, medical, industrial, and energy.

In safety-critical systems a compute or display system failure would pose a significant safety risk.
Functional Safety Certification

Safety Certification
Performed at the system level

Development Process defined in safety-critical standards
1) Document system design, safety requirements, software architecture and software design
2) Test and verify at each level against design documentation
3) Provide certification evidence packages to demonstrate documentation and testing

Reducing certification effort and costs
System runtime components should:
1) Be streamlined as far as possible to reduce documentation and testing surface area
2) Have deterministic behavior to simplify design and testing
3) Implement robust and unambiguous fault handling

In the ISO 26262 V-Model system development process testing and verification occur in reverse order from design and implementation

Industry safety-critical standards include
- RTCA DO-178C Level A / EASA ED-12C Level A (avionics)
- ISO 26262 ASIL D (automotive)
- IEC 61508 (industrial)
- IEC 62304 (medical)
Safety Certification and Open Standard APIs

Need for APIs to streamline system-level safety-critical certifications
- Streamlined
- Deterministic
- Robust

Growing need for embedded hardware acceleration
- Advanced processing of multiple advanced sensors
- Smart systems through machine learning and inferencing
- Advanced displays and user interfaces

Growing need for well-defined hardware software interoperability in safety critical industry
- Decoupling software and hardware for easier development and integration of new components
- Cross-generation reusability
- Cross-platform reusability
- Field upgradability

Growing demand for state-of-the-art open, cross-vendor, acceleration API standards that are designed by and for the safety-critical industry

K H R O N O S
G R O U P
Khronos Safety Critical GPU API Evolution

Khronos has close to 20 years experience in adapting mainstream APIs for safety-critical markets. Leveraging proven mainstream APIs with shipping silicon implementations and developer tooling and familiarity.

Vulkan SC targets any systems requiring safety critical graphics and/or compute. E.g., automotive, autonomy, avionics, medical, industrial, and energy.

Vulkan SC has significantly higher performance and flexibility than OpenGL SC. Enabling new safety-critical markets requiring graphics and compute AND cross-platform standalone compute.

OpenGL SC will continue to be supported by Khronos, but new developments will focus on Vulkan SC.
Vulkan SC 1.0 Design Philosophy

Vulkan 1.2 is a compelling starting point
- Widely adopted, royalty-free open standard
- Explicit control of device scheduling, synchronization and resource management
- Smaller surface area than OpenGL
- Not burdened by runtime debug functionality
- Very little internal state
- Well-defined thread behavior
- Ingests SPIR-V IR - no runtime front-end compiler

Vulkan SC enables system implementers deploying GPU-accelerated graphics and compute to meet safety-critical obligations and provide certification evidence packages with reduced cost and effort
Vulkan SC can also be invaluable for real-time embedded applications, even if not formally safety-certified

Streamlined
- Remove non-essential runtime functionality
  - Sparse memory
  - Descriptor update templates
  - Certain types of object deleter

Deterministic
- Predictable execution times and results
  - Offline compilation of pipelines
  - Static memory allocation

Robust
- Removing Ambiguity
  - No ignored parameters or undefined behaviors
  - Enhanced fault handling and reporting functionality
  - Rigorous conformance test suite
  - MISRA C alignment
Vulkan SC Robustness

Fault Handling and Reporting
Application registers functions at device creation which the driver can call if a fault is detected
Application can interrogate type and level of a fault together with implementation-specific data

Vulkan SC Conformance Test Suite
Freely available to all under Apache 2.0 open-source license
Leverages extensive Vulkan test suite with added SC-specific tests
System integrators can use to confirm and document Vulkan SC implementation compatibility

MISRA C
Vulkan SC 1.0 is aligned with MISRA C software development guidelines
Developed by the MISRA Consortium for embedded system code safety, security, portability and reliability and alignment with safety-critical standards
Vulkan SC Offline Compiled Pipelines

A Vulkan Pipeline defines how the GPU processes data

SPIR-V

JSON Pipeline Description

Lists all SPIR-V modules used with related state

Implementation-Specific Pipeline Cache Compiler (PCC)

Pipeline Cache Utility

Extracts information from pipeline cache files to analyze dataflow and the amount of memory used by the processing in the pipeline

Pipeline Cache Containers

Application

Memory for pipelines is reserved at device at device creation time as fixed size pools. Similarly sized pipelines can be assigned to the same pool to minimize memory size and fragmentation. Avoids need for runtime memory allocation

Offline

Runtime

DESIGNCON 2022
WHERE THE CHIP MEETS THE BOARD

APRIL 5 – 7, 2022

#DesignCon
OpenVX SC Profile

Minimizes Run-time Surface Area and Implementation Size
- Eases system-level safety certification
- Separated Development and Deployment environments

Robust Specification
- Annotated specification with Functional Requirement tag numbers
- MISRA-C compliant headers

The OpenVX SC profile combined with ingestion of trained Neural Networks enables OpenVX as a cross-platform inferencing engine for safety critical markets.
SYCL Safety-Critical Exploratory Forum

Exploring real-world industry requirements for open and royalty-free high-level compute APIs suitable for safety-critical markets

Khronos SYCL Safety-Critical Exploratory Forum

Online discussion forum and weekly Zoom calls

No detailed design activity to protect participants IP

Any company is welcome to join
No cost or IP Licensing obligations
Project NDA to cover Exploratory Forum Discussions

Proven Khronos Exploratory Process to ensure industry requirements are fully understood before starting standardization initiatives

No detailed design activity to protect participants IP

Explore if consensus can be built around an agreed Scope of Work document
Discuss what standardization activities can best execute actions in the Scope of Work

Agreed SOW document released from NDA and made public

Initiation of Khronos Working Group to execute the SOW

More information and signup instructions
https://www.khronos.org/syclsc
AUTOSAR Khronos Liaison

1. Provide Khronos members with information on AUTOSAR activities
2. Enable Khronos members to inform and influence AUTOSAR initiatives
3. Bring AUTOSAR requirements and use cases to Khronos working groups
4. Encourage normative use of Khronos standards by AUTOSAR
AUTOSAR Khronos Liaison Structure

WG-SAF: Khronos raises agenda topics relevant to Safety, e.g., Rust

Example activities

Any existing Working group or Concept

New AUTOSAR concept for safe GPU Architecture, driven by Khronos

AUTOSAR Council with access to AUTOSAR Information

Khronos Technical Liaisons

Example working groups
Khronos EMVA Camera Working Group

Open, cross-vendor standard for camera, sensor and ISP control has multiple benefits

Cross-vendor portability of camera/sensor code for easier system integration of new sensors

Preservation of application code across multiple generations of cameras and sensors

Sophisticated control over sensor stream generation for effective downstream accelerated processing

Application controlling sensor stream generation and processing in real time

Khronos Camera Working Group announced in February 2022

EMVA and Khronos cooperated since 2020 to understand need for a new API for standard camera and image capture control

Over 70 companies met at the Khronos EMVA Camera Exploratory Group through 2021 to create consensus on a Scope of Work document

Working Group is open to all Khronos members and meeting weekly to execute the Scope of Work
Typical Software Stack using Camera API

Frameworks & Middleware:
- GStreamer
- OpenVX

Application

Camera System API

Transport:
- CSI-2
- USB
- Ethernet

Physical Devices:
- Sensors
- Lenses
- Lights
- Processors

API in scope

Some libraries may be in scope

Named transport layers, frameworks and operating systems are illustrative examples.
Camera API Terminology

Physical Devices = queryable and controllable via a Device ID:
Logical Device = set of Devices queried and controlled via a single Device ID
Frame = Image + Metadata accessed via Frame ID
Stream = sequence of Frames
Camera = a Logical Device that exports one or more Streams from the Camera System
Camera API Working Group Organization

Khronos
Any company can join with Membership Fee
Membership Agreement grants access to ALL Working Groups AND Advisory Panels
Under Khronos NDA and IP Framework

EMVA
All EMVA members have a standing invitation to join Camera Advisory Panel at no charge

Drafts
Camera System API Working Group
Design decisions and execution responsibility

Feedback
Camera System API Advisory Panel
Design suggestions and draft feedback

Other Khronos Working Groups and Advisory Panels

Advisory Panel
Membership by invitation with no Membership Fee
Advisory Panel Agreement grants access to ONLY Camera Advisory Panel
Under Khronos NDA and IP Framework

More information
https://www.khronos.org/camera
Get Involved!

Any company is welcome to join Khronos to influence standards development
https://www.khronos.org/members/ or email memberservices@khronosgroup.org

More information on Vulkan SC, OpenVX and OpenCL

All Khronos members can participate in the new Camera Working Group
https://www.khronos.org/camera

Get involved in the new SYCL SC Exploratory Forum at zero cost
https://www.khronos.org/syclsc

Khronos is developing a growing family of open, royalty-free API
standards relevant to embedded and safety-critical markets
Thank you!

QUESTIONS?