Embedded Vision and Inferencing Acceleration

- Networks trained on high-end desktop and cloud systems
- Apps link to compiled code or inferencing library
- Diverse Embedded Hardware (GPUs, DSPs, FPGAs)

Neural Network Training

Networks trained on high-end desktop and cloud systems

Compiled Code

Inferencing Engine

Application Code

Vision Library

Hardware Acceleration APIs

Sensor Data

Compilation Ingestion

© Copyright Khronos® Group 2019
Neural Network Training Acceleration

- Network training needs 32-bit floating-point precision
- ONNX: Authoring interchange
- GPU APIs and libraries for compute acceleration
- cuDNN, MiOpen, cIDNN, TPU
- Neural Net Training Frameworks
- PyTorch, Caffe, Keras, TensorFlow, Theano, PaddlePaddle

Desktop and Cloud Hardware
Embedded Vision and Inferencing Acceleration

Neural Network Training

Compiled Code
Inferencing Engine
Application Code
Vision Library

Hardware Acceleration APIs

Sensor Data

Diverse Embedded Hardware (GPUs, DSPs, FPGAs)

Networks trained on high-end desktop and cloud systems

Apps link to compiled code or inferencing library

Training Data

© Copyright Khronos® Group 2019
Neural Network Exchange Industry 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 Support

NNEF V1.0 released in August 2018
After positive industry feedback on Provisional Specification - maintenance update issued in March 2019
Extensions to V1.0 in Ratification for imminent release

ONNX 1.5 Released in April 2019
Introduced support for Quantization
ONNX Runtime being integrated with GPU inferencing engines such as NVIDIA TensorRT

NNEF Working Group Participants

ONNX Supporters

NNEF adopts a rigorous approach to design life cycles - especially needed for safety-critical or mission-critical applications in automotive, industrial and infrastructure markets
NNEF Tools Ecosystem

NNEF open source projects hosted on Khronos
NNEF GitHub repository under Apache 2.0
https://github.com/KhronosGroup/NNEF-Tools

Other NNEF Updates
NNEF Model Zoo is now available on GitHub
System-level conformance testing program is being created
Embedded Vision and Inferencing Acceleration

Networks trained on high-end desktop and cloud systems

Apps link to compiled code or inferencing library

Diverse Embedded Hardware (GPUs, DSPs, FPGAs)

Sensor Data

Neural Network Training

Training Data

Trained Networks

Compilation

Ingestion

Compiled Code

Inferencing Engine

Application Code

Vision Library

Hardware Acceleration APIs

© Copyright Khronos® Group 2019
Primary Machine Learning Compilers

<table>
<thead>
<tr>
<th>Import Formats</th>
<th>Caffe, Keras, MXNet, ONNX</th>
<th>TensorFlow Graph, MXNet, PaddlePaddle, 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>

- TensorFlow Graph
- MXNet
- PaddlePaddle
- Keras
- ONNX
- PyTorch
- ONNX
- TensorFlow Graph
- PyTorch

- OpenCL
- LLVM
- CUDA
- Metal
- OpenCL
- LLVM
- CUDA
- TPU IR
- XLA IR
- TensorFlow Lite / NNAPI (inc. HW accel)

- Third party components
- Under development
ML Compiler Steps

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 using node fusion 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 Vision and Inferencing Acceleration

Networks trained on high-end desktop and cloud systems

Apps link to compiled code or inferencing library

Diverse Embedded Hardware (GPUs, DSPs, FPGAs)

Sensor Data
PC/Mobile Platform Inferencing Stacks

Consistent Steps
1. Import and optimize trained NN model file
2. Ingest graph into ML runtime
3. Accelerate inferencing operations using underlying low-level API

Microsoft Windows
Windows Machine Learning (WinML)

Google Android
Neural Network API (NNAPI)
https://developer.android.com/ndk/guides/neuralnetworks/

Apple MacOS and iOS
CoreML
https://developer.apple.com/documentation/coreml
Inferencing Engines

Platforms
- Android NNAPI
- Microsoft WinML
- Apple CoreML

Desktop IHVs
- AMD MIVisionX over MIOpen
- Intel OpenVINO over cIDNN
- NVIDIA TensorRT over cuDNN

Mobile / Embedded
- Arm Compute Library
- Huawei MACE
- Qualcomm Neural Processing SDK
- Synopsis MetaWare EV Dev Toolkit
- TI Deep Learning Library (TIDL)
- VeriSilicon Acuity

Almost all Embedded Inferencing Engines use OpenCL to access accelerator silicon

Cross-platform Inferencing Engines
- OpenCV
- OpenVX

Both provide Inferencing AND Vision acceleration

Acceleration APIs
- Vulkan
- OpenGL ES
- DirectX 12
- ROCm
- CUDA
- SYCL
Khronos OpenVX and NNEF for 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

---

**Performance comparable to hand-optimized, non-portable code**
Real, complex applications on real, complex hardware
Much lower development effort than hand-optimized code
### OpenVX and OpenCV are Complementary

<table>
<thead>
<tr>
<th>Implementation</th>
<th>Community driven open source library</th>
<th>Callable library implemented and shipped by hardware vendors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conformance</td>
<td>Extensive OpenCV Test Suite but no formal Adopters program</td>
<td>Implementations pass defined conformance test suite to use trademark</td>
</tr>
<tr>
<td>Scope</td>
<td>Very wide. 1000s of imaging and vision functions Multiple camera APIs/interfaces</td>
<td>Tight focus on core hardware accelerated functions for mobile vision and inferencing. Uses external camera drivers</td>
</tr>
<tr>
<td>Acceleration</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. Uses OpenCL for Custom Nodes</td>
</tr>
<tr>
<td>Efficiency</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>Inferencing</td>
<td>Deep Neural Network module. API to construct neural 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</td>
</tr>
</tbody>
</table>
Embedded Vision and Inferencing Acceleration

- Networks trained on high-end desktop and cloud systems
- Apps link to compiled code or inferencing library
- Diverse Embedded Hardware (GPUs, DSPs, FPGAs)
- Neural Network Training
  - Training Data
- Trained Networks
  - Compilation
  - Ingestion
- Compiled Code
- Inferencing Engine
- Application Code
- Vision Library
- Hardware Acceleration APIs
- Sensor Data

© Copyright Khronos® Group 2019
GPU and Heterogeneous Acceleration APIs

OpenCL provides a programming and runtime framework for heterogeneous compute resources. Low-level control over memory allocation and parallel task execution - simpler and relatively lightweight compared to GPU APIs.


Application

Fragmented GPU API Landscape

CPU

GPU

FPGA

DSP

Custom Hardware

Heterogeneous Compute Resources

© Copyright Khronos® Group 2019
Safety Critical GPU API Evolution

OpenGL ES 1.0 - 2003
Fixed function graphics

OpenGL SC 1.0 - 2005
Fixed function graphics
safety critical subset

OpenGL ES 2.0 - 2007
Programmable Shaders

OpenGL SC 2.0 - April 2016
Programmable Shaders
safety critical subset

Vulkan 1.0 - 2016
Explicit Graphics and Compute

New Generation Safety
Critical API for Graphics,
Compute and Display
Vulkan SC Working Group Goals

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

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

Vulkan SC Working Group announced February 2019
Any company welcome to join Khronos and participate
OpenCL is Pervasive!

Hardware Implementations:
- Apple
- Atera
- AMD
- Arm
- IBM
- Imagination
- Intel
- Marvell
- MediaTek
- Qualcomm
- NVIDIA
- Samsung
- Texas Instruments
- Xilinx
- Vertisilicon
- ST

Desktop Creative Apps:
- Adobe
- blender
- ArcSoft
- SideFX
- Capture One
- otoy
- Dassault Systèmes
- CyberLink
- Chaos Group
- ptc
- Blackmagic Design
- GIMP
- Autodesk

Parallel Computation Languages:
- OpenACC
- SYCL
- OpenCL
- PyOpenCL

Linear Algebra Libraries:
- CLBlast
- ViennaCL
- SYCL-BLAS

Machine Learning Libraries:
- Intel cIDNN
- Mace Models for Computer Vision
- Huawei
- Intel MetaWare EV
- Qualcomm NNAPI Processing SDK for AI
- Caffe
- Arm Compute Library
- TI Deep Learning Library (TIDL)
- SyCL-DNN
- Verisilicon

Math and Physics Libraries:
- ArrayFire
- MATLAB
- Wolfram Mathematica
- ECL
- GNU Octave
- Bullet Physics Library
- C, C++, Fortran

Vision and Imaging Libraries:
- OpenVX
- OpenCV
- VisionCpp
- Halide
- OPenVL
- SniperCV

Machine Learning Inferencing Compilers:
- Glow
- TVM
- PlaidML

© Copyright Khronos® Group 2019
SYCL Single Source C++ Parallel Programming

- SYCL 1.2.1 Adopters Program released in July 2018 with open source conformance tests soon

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

- Multiple Implementations shipping
  - TriSYCL, ComputeCpp, HipSYCL
  - [http://sycl.tech](http://sycl.tech)

- C++ Kernel Fusion can give better performance on complex apps and libs than hand-coding
  - [ideal for accelerating larger C++-based engines and applications](#)

- Accelerated code passed into device OpenCL compilers

- E.g. complex ML frameworks can be directly compiled and accelerated
SPIR-V Ecosystem

Third party kernel and shader languages

GLSL  HLSL

glslang  DXC

SPIR-V (Dis)Assembler

SPIR-V Validator

SPIRV-opt | SPIRV-remap

Optimization Tools

IHV Driver Runtimes

Environment spec used for compilation needs to match target runtime

**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 OSS**

Khronos-hosted Open Source Projects

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

SPIR-V Magic #: 0x07230203
SPIR-V Version 99
Builder's Magic #: 0x05F2000B
<id> bound is 50

OpMemoryModel Logical

OpEntryPreOp
Fragment shader
function <id>= 4

OpTypeVoid <id>= 2

OpTypeFunction <id>= 3
return type <id>= 2

OpFunction Result <id>= 2
Result <id>= 4
0

Function Type <id>= 3

SPIR-V Cross

GLSL  HLSL

OpenGL

Vulkan

IHV Driver Runtimes

SPIR-V Validator

SPIRV-opt | SPIRV-remap

Optimization Tools

Environment spec used for compilation needs to match target runtime

**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 Evolution

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

**Integration of Extensions plus New Core functionality**
- Vulkan-like loader and layers
- ‘Flexible Profile’ for Deployment Flexibility

**Spec Maintenance Updates**
- Regular updates for spec clarifications and bug fixes

**Repeat Cycle for next Core Specification**

May 2017 OpenCL 2.2

Target 2020 ‘OpenCL Next’
Flexible Profile and Feature Sets

- In OpenCL Next Flexible Profile features become optional for enhanced deployment flexibility
  - API and language features e.g. floating point precisions

- Feature Sets reduce danger of fragmentation
  - Defined to suit specific markets - e.g. desktop, embedded vision and inferencing

- Implementations are conformant if fully support feature set functionality
  - Supporting Feature Sets will help drive sales - encouraging consistent functionality per market
  - An implementation may support multiple Feature Sets

OpenCL 2.2 Functionality

- Khronos-defined
- OpenCL 2.2 Full Profile Feature Set
- Khronos-defined
- OpenCL 1.2 Full Profile Feature Set
- Industry-defined Feature Set E.g. Embedded Vision and Inferencing

= queryable, optional feature
OpenCL Next Feature Set Discussions

- What could be useful Feature Sets?
  - Feature sets for previous spec versions e.g. OpenCL 1.2?
  - ‘Desktop’ Feature Set to raise the universally available baseline above OpenCL 1.2?
  - Set of OpenCL functionality that runs efficiently over Vulkan?
  - Vertical market focused - e.g. inferencing, vision processing

- Some vertically integrated markets don’t care about cross-vendor app portability
  - But still value in industry standard programming framework - reduces costs for engineering, tooling, training etc.
  - Allow conformance for any combination of features - no Feature Sets
  - Enables minimal footprint OpenCL per system - ideal for Safety Certification

In some markets, e.g. desktop, software portability across multiple vendors’ accelerators is key

In some vertically integrated embedded markets, minimized driver footprint is key
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 a OpenVX Node in a graph

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

• Compilers for Machine Learning Conference Proceedings
  - www.c4ml.org

• Khronos Standards: OpenCL, OpenVX, NNEF, SYCL
  - Any company is welcome to join Khronos
  - www.khronos.org

• OpenCL application and library resources
  - https://www.khronos.org/opencl/resources/opencl-applications-using-opencl
  - https://www.iwocl.org/resources/opencl-libraries-and-toolkits/

• In-Depth Presentations on OpenCV and OpenVX
  - This room today at 1PM and 1:35PM

• Khronos EVS Workshop
  Hardware Acceleration for Machine Learning and Computer Vision through Khronos Open Standard APIs
  - Many more details on OpenVX, SYCL, OpenCL, NNEF
  - Hands-on exercises with OpenVX and NNEF
  - Thursday May 23, 9am-5pm
  - Rooms 203 and 204, Santa Clara Convention Center
  - https://www.khronos.org/events/2019-embedded-vision-summit