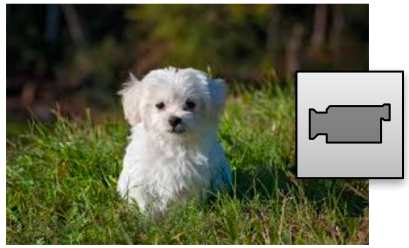
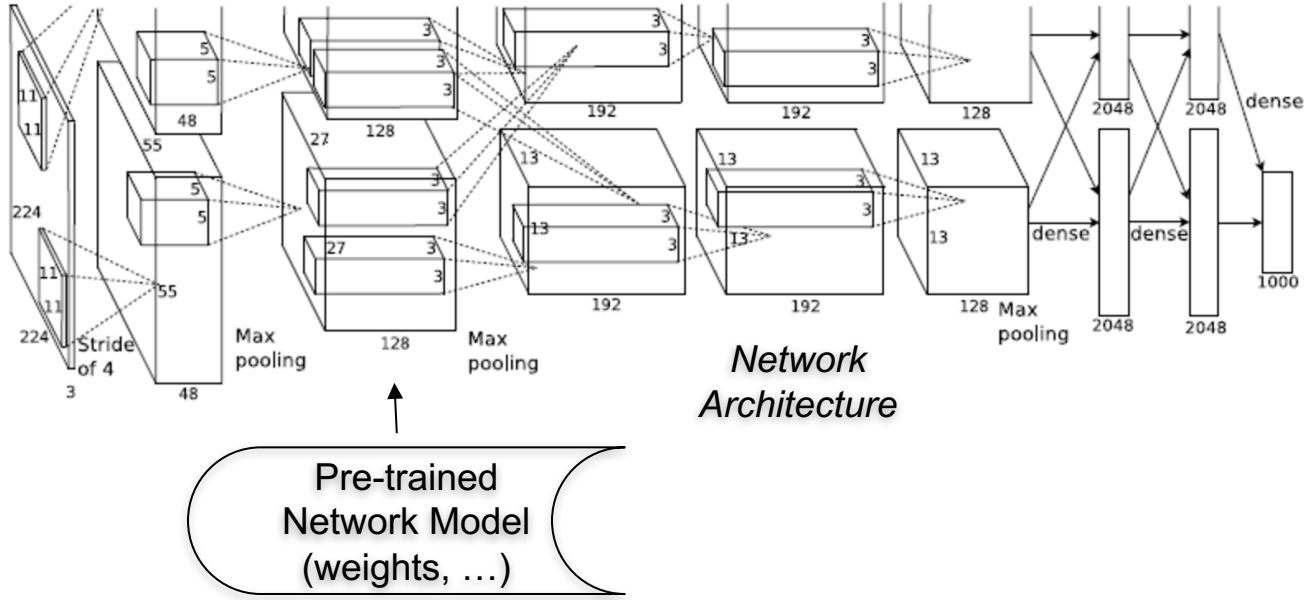
Standards for Vision Processing and Neural Networks

Radhakrishna Giduthuri, AMD
radha.giduthuri@ieee.org

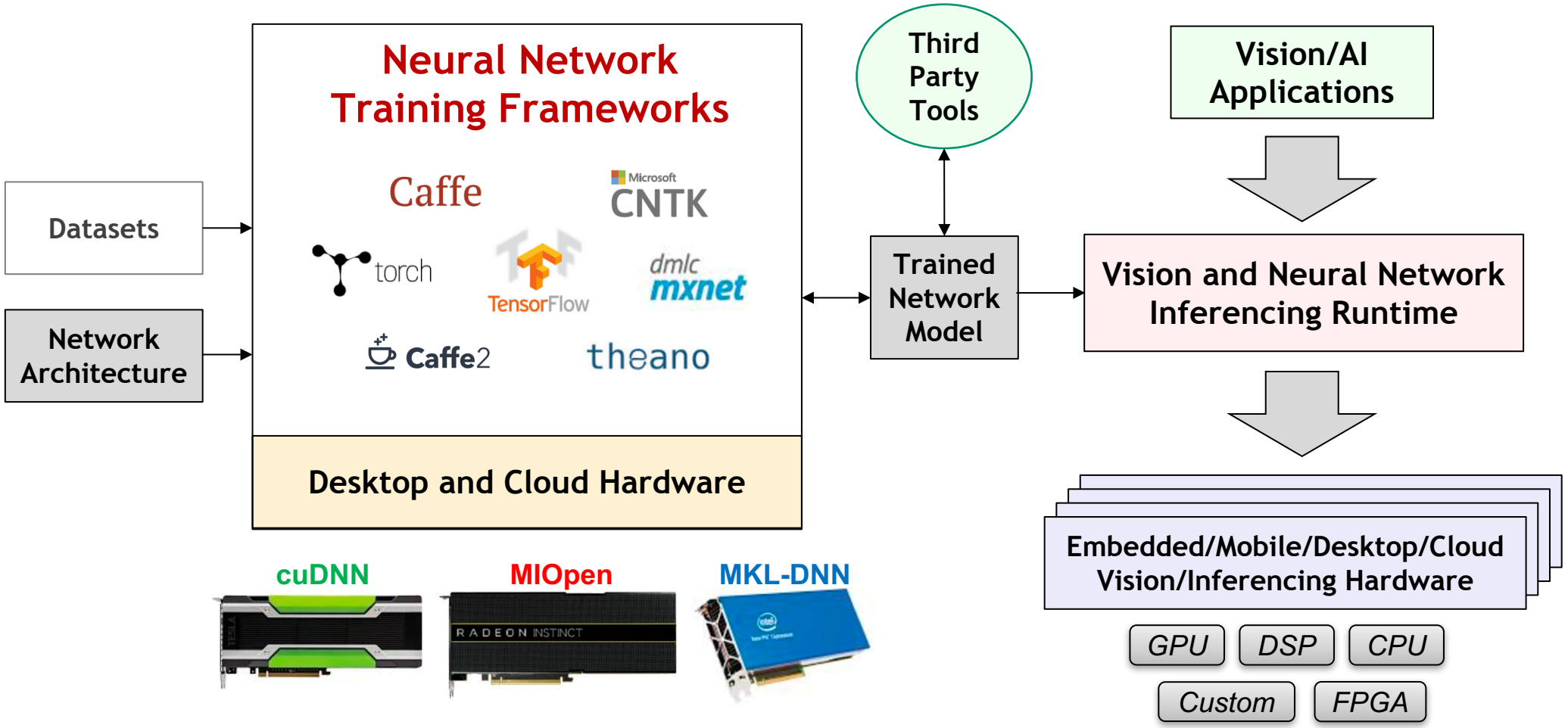
Agenda

- Why we need a standard?
- Khronos NNEF
- Khronos OpenVX

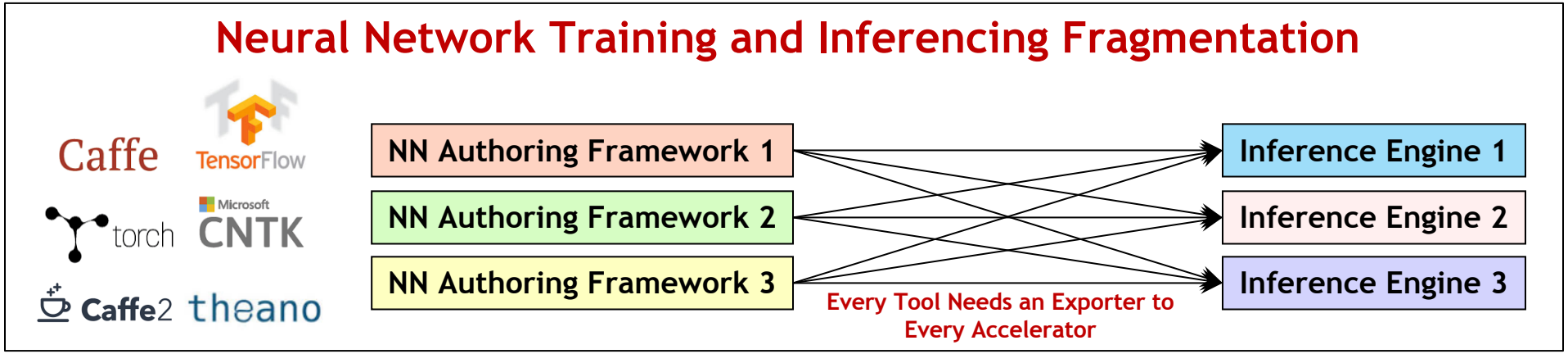


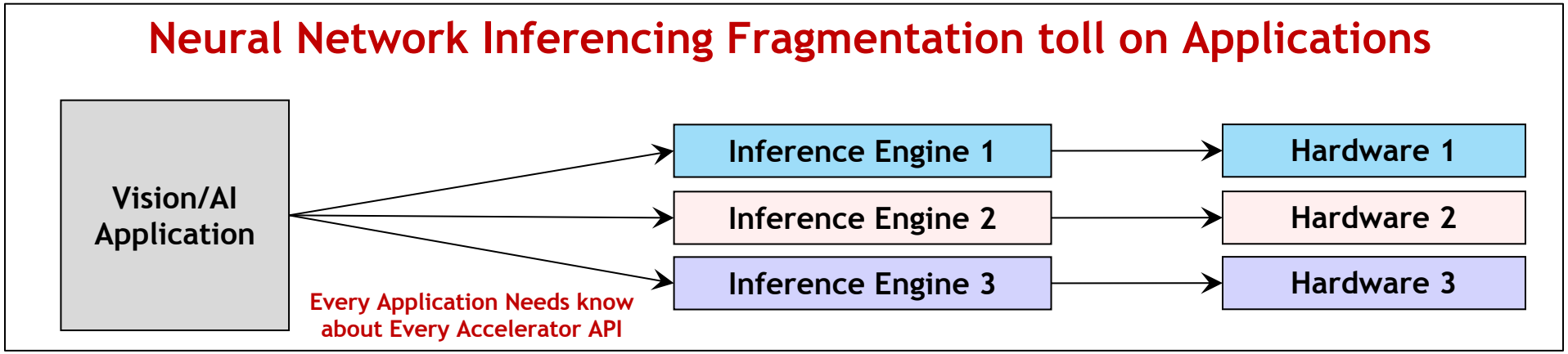


Neural Network End-to-End Workflow



Problem: Neural Network Fragmentation



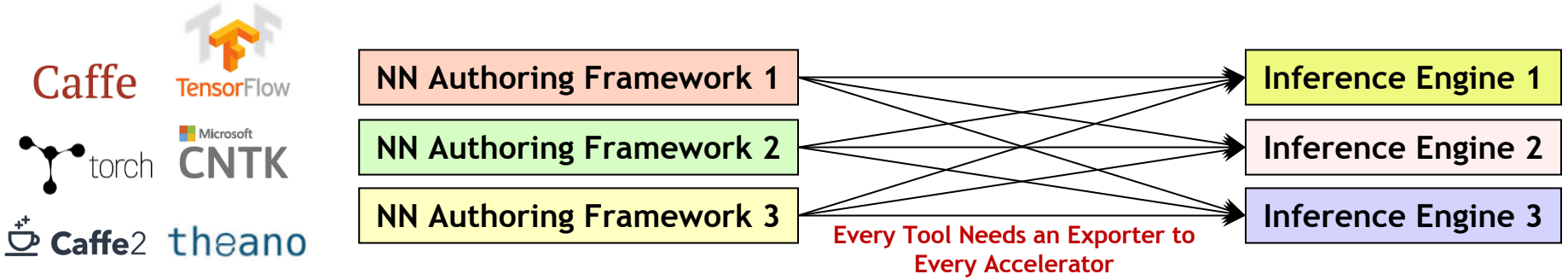


Khronos APIs Connect Software to Silicon

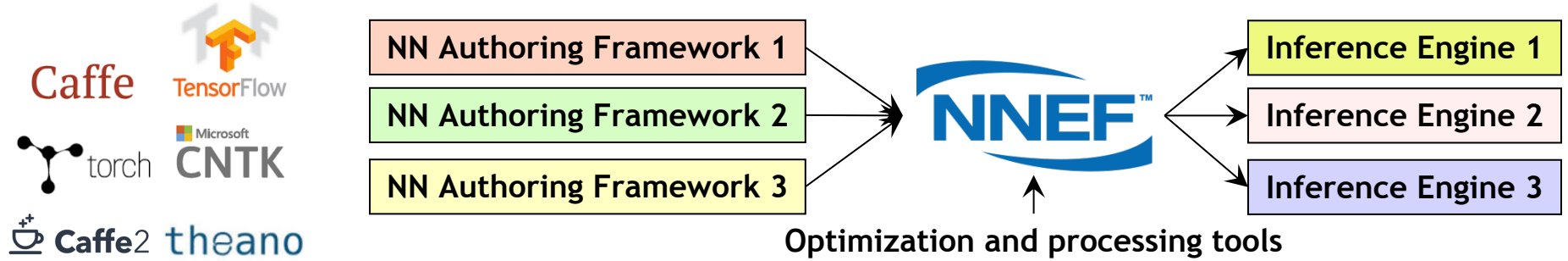
Khronos is an International Industry Consortium of over 100 companies creating royalty-free, open standard APIs to enable software to access hardware acceleration for 3D graphics, Virtual and Augmented Reality, Parallel Computing, Vision Processing and Neural Networks

NNEF - Solving Neural Network Fragmentation

Before NNEF - NN Training and Inferencing Fragmentation



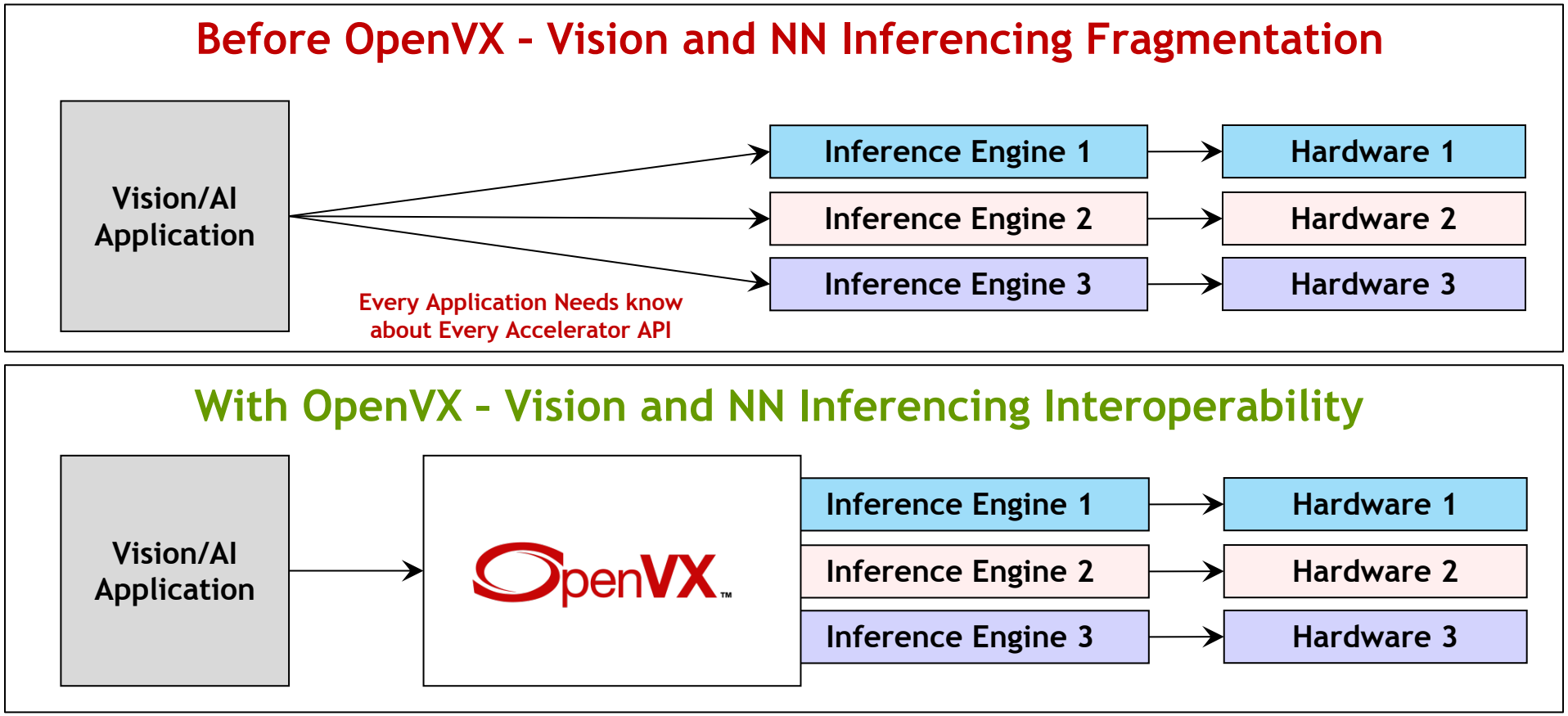
With NNEF- NN Training and Inferencing Interoperability



NNEF is a Cross-vendor Neural Net file format

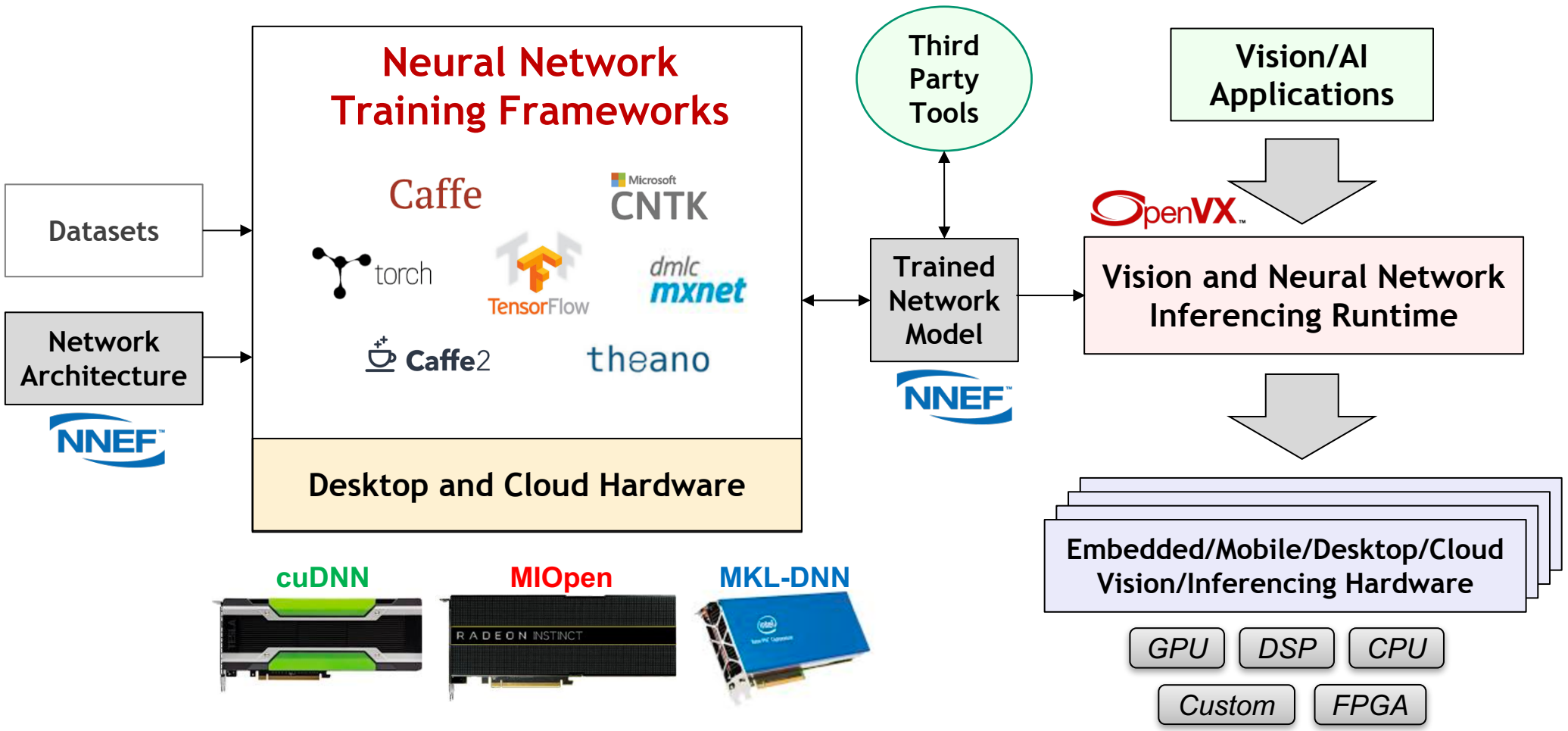
Encapsulates network formal semantics, structure, data formats, commonly-used operations (such as convolution, pooling, normalization, etc.)

OpenVX - Solving Inferencing Fragmentation

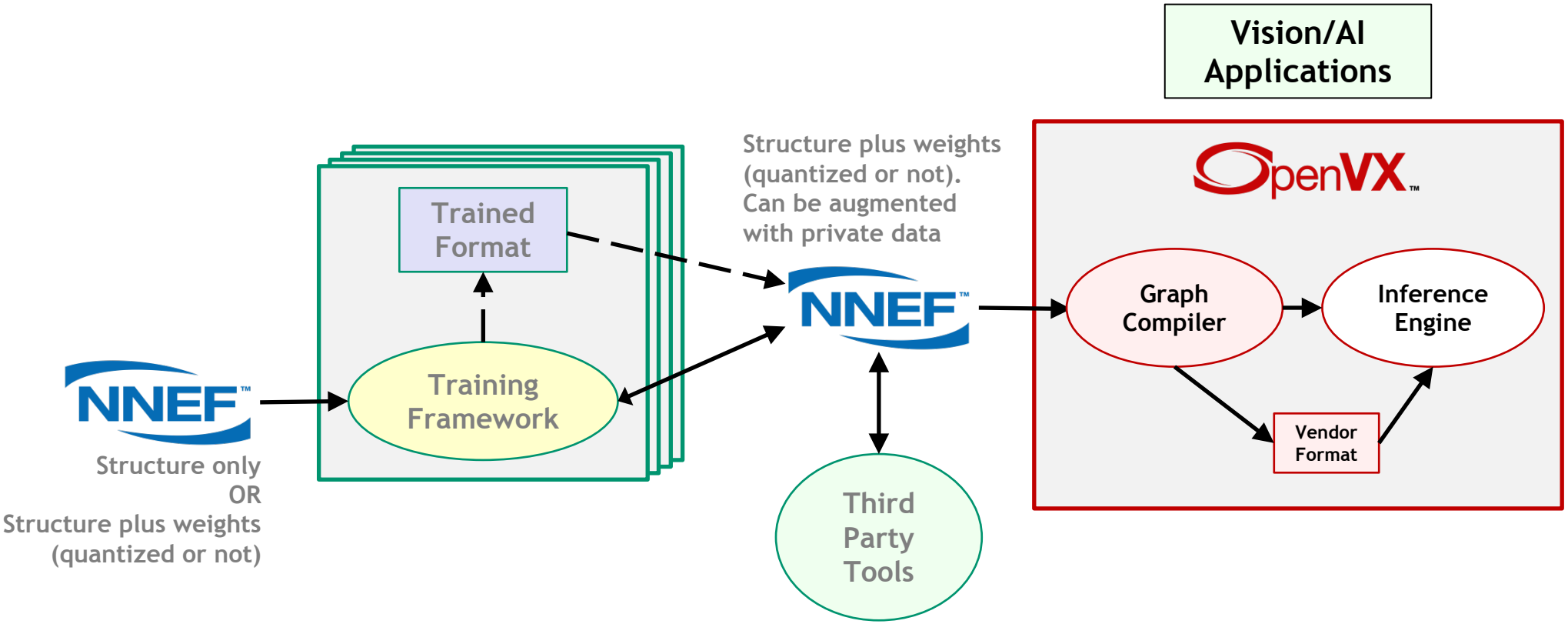


OpenVX is an open, royalty-free standard for cross platform acceleration of computer vision and neural network applications.

Neural Net Workflow with Khronos Standards



Application Development Workflow



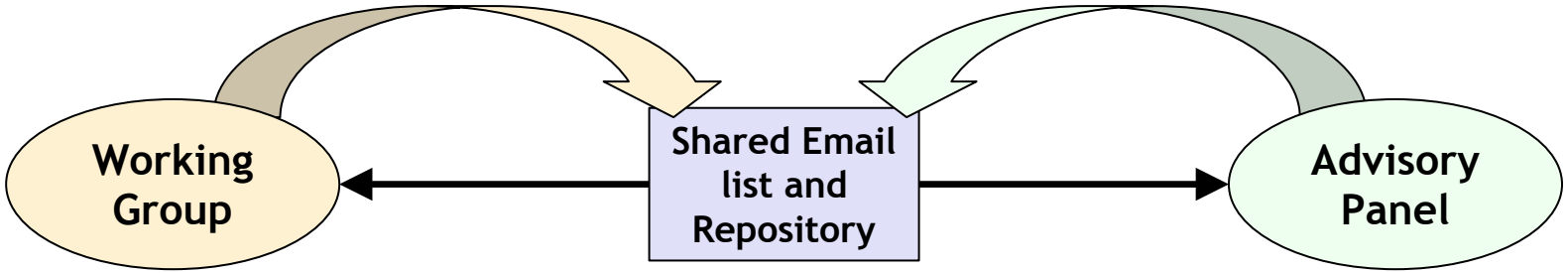
NNEF Status and Roadmap

- **V1.0 is under development, industry comments are being sought now**
  - NNEF has formed an advisory panel, you are invited today to participate
- **First version will focus on interface between framework and embedded inference engines**
  - 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
  - Allow authoring and retraining (3rd party tools)
  - Address a wider range of applications (outside vision apps)
  - Increase the expressive power of the format

Khronos Advisory Panels

Specification drafts and invitations for requirements and feedback

Requirements and feedback on specification drafts



Khronos Members
Any company can join
Membership Fee
Sign NDA and IP Framework

Hosted by Khronos.
Under Khronos NDA

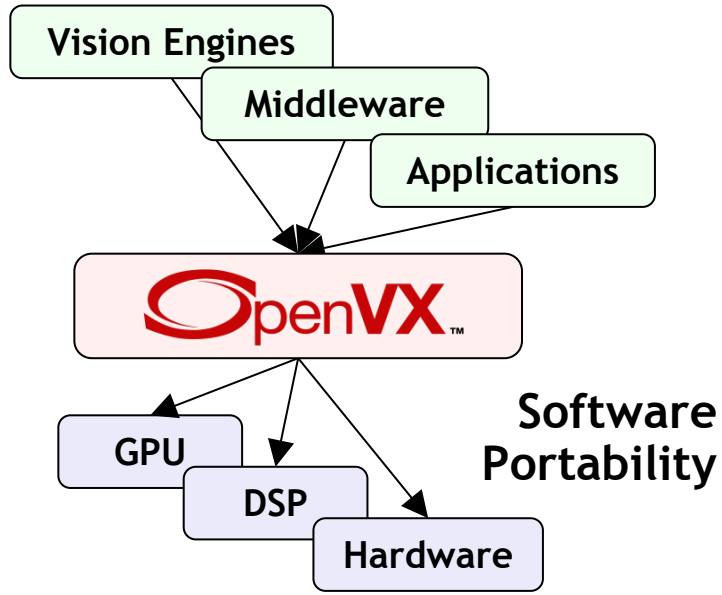
Khronos Advisors
Invited industry experts
$0 Cost
Sign NDA and IP Framework

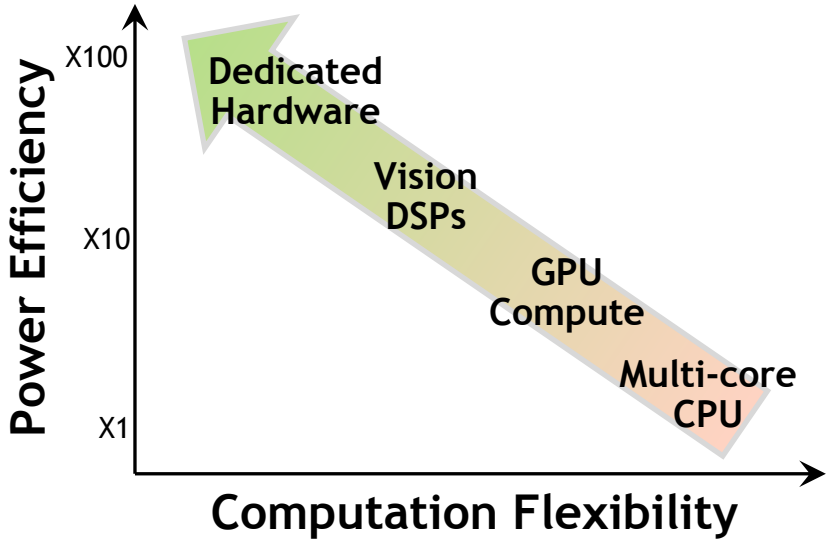
Advisory Panels Active for NNEF, Vulkan, and OpenCL/SYCL

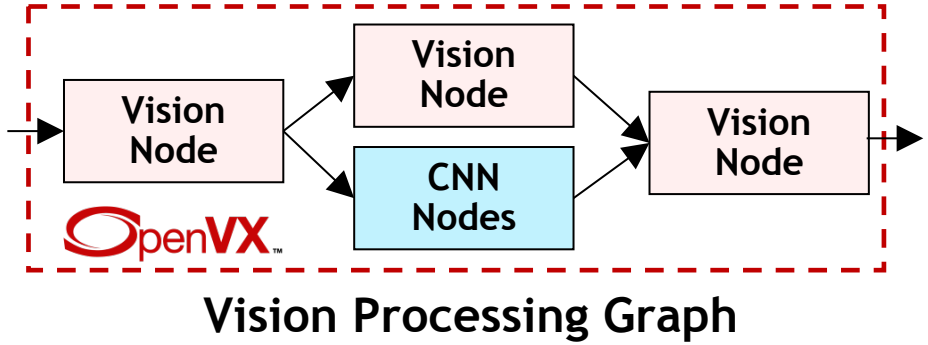
An open, royalty-free standard for cross platform acceleration of computer vision and neural network applications.

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!



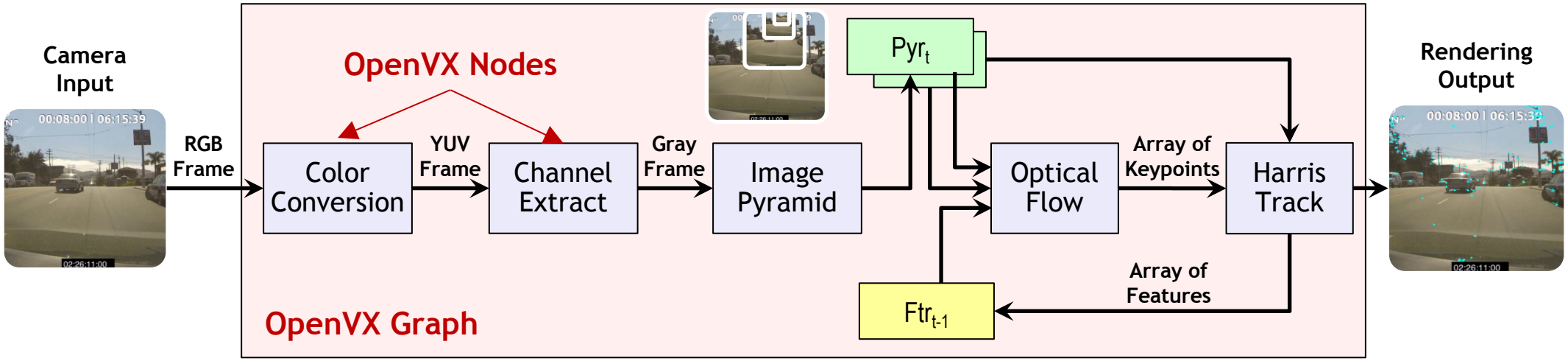




Vision Processing Graph

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.



Feature Extraction Example Graph

OpenVX Efficiency through Graphs..

Graph Scheduling

Split the graph execution across the whole system: CPU / GPU / dedicated HW

Faster execution or lower power consumption

Memory Management

Reuse pre-allocated memory for multiple intermediate data

Less allocation overhead, more memory for other applications

Kernel Fusion

Replace a sub-graph with a single faster node

Better memory locality, less kernel launch overhead

Data Tiling

Execute a sub-graph at tile granularity instead of image granularity

Better use of data cache and local memory

Simple Edge Detector in OpenVX

```
vx_graph g = vxCreateGraph();

vx_image input = vxCreateImage(1920, 1080);
vx_image output = vxCreateImage(1920, 1080);
vx_image horiz = vxCreateVirtualImage(g);
vx_image vert = vxCreateVirtualImage(g);
vx_image mag = vxCreateVirtualImage(g);

vxSobel3x3Node(g, input, horiz, vert);
vxMagnitudeNode(g, horiz, vert, mag);
vxThresholdNode(g, mag, THRESH, output);

status = vxVerifyGraph(g);
status = vxProcessGraph(g);
```

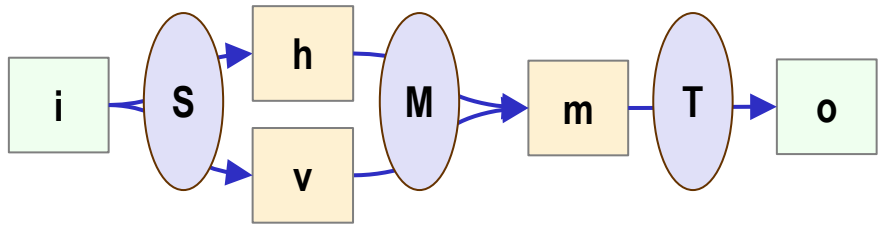
Declare Input and Output Images

Declare Intermediate Images

Construct the Graph topology

Compile the Graph

Execute the Graph



OpenVX Evolution

Conformant Implementations

AMD
cadence
CEVA
Neolek
intel
Imagination
NVIDIA
socionext
SYNOPSYS
VeriSilicon

OpenVX 1.0
Spec released October 2014

Conformant Implementations

cadence
Imagination
intel
Texas Instruments

New Functionality
Expanded Nodes Functionality
Enhanced Graph Framework

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/

OpenVX 1.1
Spec released May 2016

New Functionality
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

OpenVX 1.2
Spec released May 2017

New Functionality Under Discussion

NNEF Import

Programmable user kernels with accelerator offload

Streaming/pipelining

OpenVX Roadmap

KHRONOS GROUP

AMD's open-source implementation

- **Highly-optimized for x86 CPU and OpenCL GPU**

- **Available on github.com**

  http://github.com/GPUOpen-ProfessionalCompute-Libraries/amdovx-modules

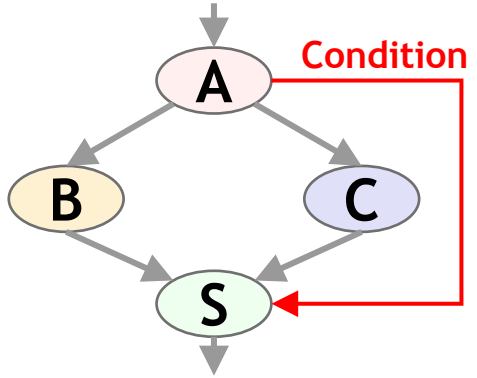
- **Additional modules**
  - LOOM: highly optimized library for real-time 360 degree video stitching
  - NN: neural network module built on top MIOpen (OpenCL-based machine learning primitives)

    Includes a tool to import pre-trained Caffe models into NN (develop branch)

    …

New OpenVX 1.2 Functions

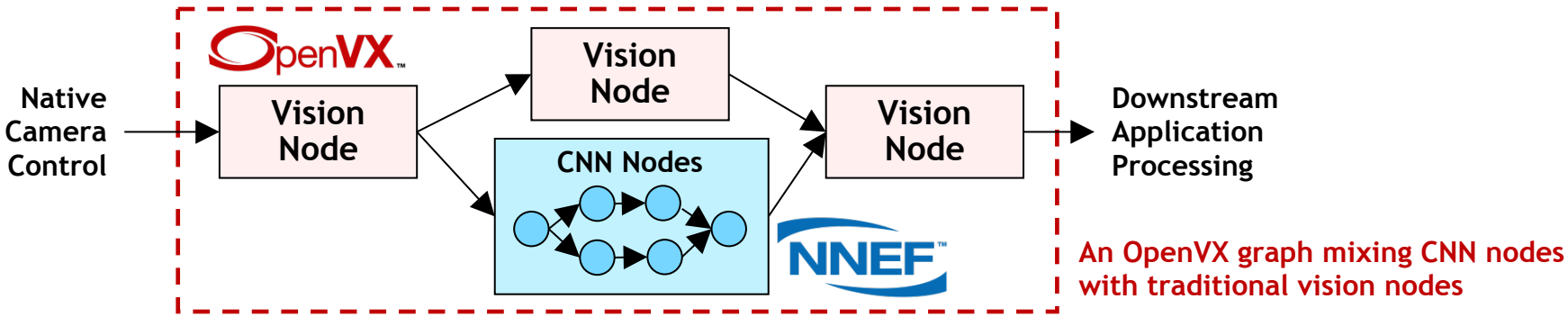
- **Feature detection: find features useful for object detection and recognition**
  - Histogram of gradients – HOG
  - Local binary patterns – LBP
  - Template matching
  - Line finding
- **Classification: detect and recognize objects in an image based on a set of features**
  - Import a classifier model trained offline
  - Classify objects based on a set of input features
- **Image Processing: transform an image**
  - Generalized nonlinear filter: Dilate, erode, median with arbitrary kernel shapes
  - Non maximum suppression: Find local maximum values in an image
  - Edge-preserving noise reduction
- **Conditional execution & node predication**
  - Selectively execute portions of a graph based on a true/false predicate
- **Many, many minor improvements**
- **New Extensions**
  - **Import/export:** compile a graph; save and run later
  - **16-bit support:** signed 16-bit image data
  - **Neural networks:** Layers are represented as OpenVX nodes



If A then S ← B else S ← C

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**

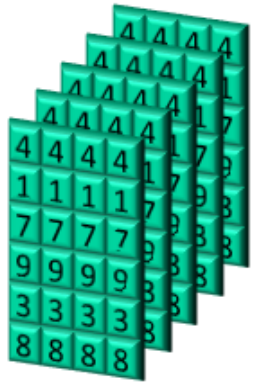


OpenVX Neural Network Extension

- Two main parts: (1) a tensor object and (2) a set of CNN layer nodes
- A vx_tensor is a multi-dimensional array that supports at least 4 dimensions

1-D tensor: [6]
i.e., 6-element vector

2-D tensor: [6, 4]
i.e., 6 by 4 matrix



3-D tensor: [6, 4, 5]

4-D tensor: [6, 4, 5, 3]

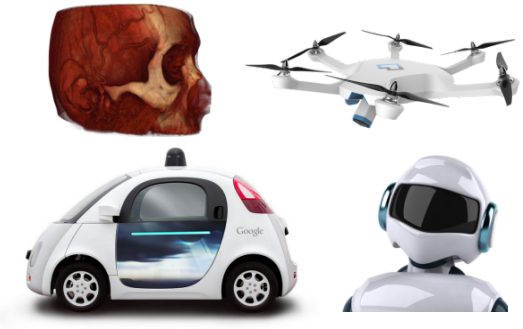
- Tensor creation and deletion functions
- Simple math for tensors
  - Element-wise Add, Subtract, Multiply, TableLookup, and Bit-depth conversion
  - Transposition of dimensions and generalized matrix multiplication
  - vxCopyTensorPatch, vxQueryTensor (#dims, dims, element type, Q)

OpenVX Neural Network Extension

- **Tensor types of INT16, INT7.8, INT8, and U8 are supported**
  - Other types may be supported by a vendor
- **Conformance tests will be up to some "tolerance" in precision**
  - To allow for optimizations, e.g., weight compression
- **Eight neural network "layer" nodes:**

vxActivationLayer	vxConvolutionLayer	vxDeconvolutionLayer
vxFullyConnectedLayer	vxNormalizationLayer	vxPoolingLayer
vxSoftmaxLayer	vxROIPoolingLayer	…

Safety Critical APIs



OpenGL SC 1.0 - 2005
Fixed function graphics subset

OpenGL SC 2.0 - April 2016
Shader programmable pipeline subset

Experience and Guidelines

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

OpenGL ES 1.0 - 2003
Fixed function graphics

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

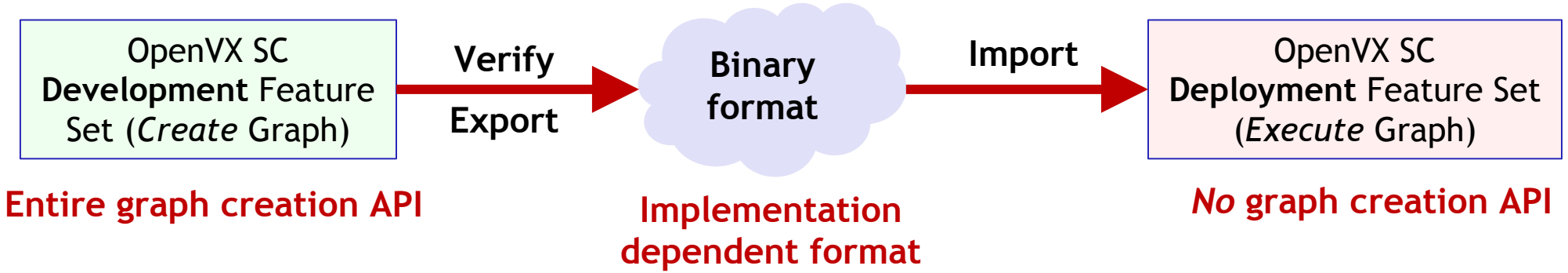
'Safety Critical Advisory Panel'
Guidelines for designing APIs that ease system certification.
Open to Khronos member AND industry experts

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



How OpenVX Compares to Alternatives

	OpenVX	OpenCV	OpenCL
Governance	Open standard API designed to be implemented and shipped by IHVs	Community-driven, open source library	Open standard API designed to be implemented and shipped by IHVs
Programming Model	Graph defined with C API and then compiled for run-time execution	Immediate runtime function calls – reading to and from memory	Explicit kernels are compiled and executed via run-time API
Built-in Vision Functionality	Small but growing set of popular functions	Vast. Mainly on PC/CPU	None. User programs their own or call vision library over OpenCL
Target Hardware	Any combination of processors or non-programmable hardware	Mainly PCs and GPUs	Any heterogeneous combination of IEEE FP-capable processors
Optimization Opportunities	Pre-declared graph enables significant optimizations	Each function reads/writes memory. Power performance inefficient	Any execution topology can be explicitly programmed
Conformance	Implementations must pass conformance to use trademark	Extensive Test Suite but no formal Adopters program	Implementations must pass conformance to use trademark
Consistency	All core functions must be available in conformant implementations	Available functions vary depending on implementation / platform	All core functions must be available in all conformant implementations

OpenVX Benefits and Resources

- **Faster development of efficient and portable vision applications**
  - Developers are protected from hardware complexities
  - No platform-specific performance optimizations needed
- **Graph description enables significant automatic optimizations**
  - Scheduling, memory management, kernel fusion, and tiling
- **Performance portability to diverse hardware**
  - Hardware agility for different use case requirements
  - Application software investment is protected as hardware evolves
- **OpenVX Resources**
  - OpenVX Overview
    - https://www.khronos.org/openvx
  - OpenVX Specifications: current, previous, and extensions
    - https://www.khronos.org/registry/OpenVX
  - OpenVX Resources: implementations, tutorials, reference guides, etc.
    - https://www.khronos.org/openvx/resources

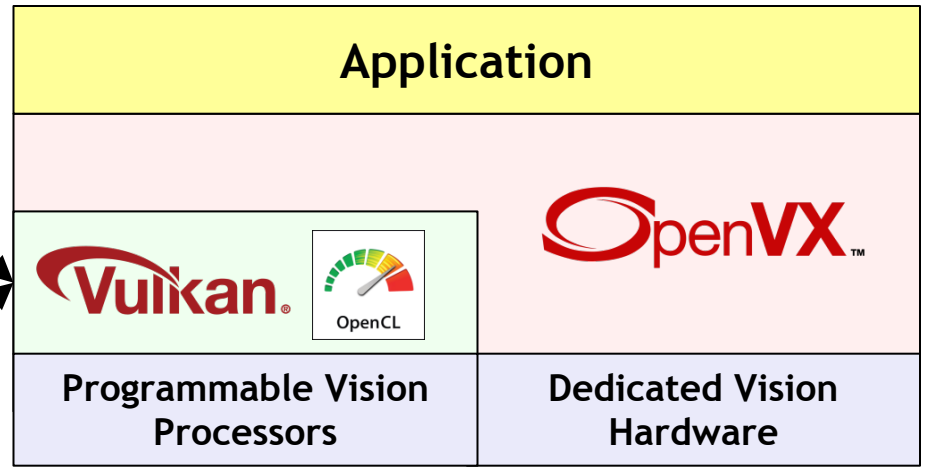
Layered Vision/ Neural Net Ecosystem

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

Vulkan Roadmap
Enhanced compute - especially useful for vision and inferencing on mobile and Android platforms

OpenCL Roadmap
Flexible precision for widespread deployment on low-cost embedded processors



And then implementors 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

Any Questions?

- **Khronos working on a comprehensive set of solutions for vision and inferencing**
  - Layered ecosystem that include multiple developer and deployment options
- **These slides and further information on all these standards at Khronos**
  - www.khronos.org
- **Please let us know if we can help you participate in any Khronos activities!**
  - You are very welcome – and we appreciate your input!
- **Please contact us with any comments or questions!**
  - Radhakrishna Giduthuri | radha.giduthuri@ieee.org