GISC2020
Global ICT Standards Conference

뉴노멀시대 선도를 위한
ICT 표준의 역할

Khronos Standard for AI and GPU computing

Hwanyong Lee, Ajou University, Khronos Group
Original Presentation of Neil Trevett, President of Khronos Group

GISC2020
Global ICT Standards Conference

뉴 노멀 시대
선도를 위한
ICT 표준의
역할

KHRONOS® GROUP

OpenCL™

SYCL™

OpenVX™

NNEF™

GISC2020
Global ICT Standards Conference

뉴 노멀 시대
선도를 위한
ICT 표준의
역할

Khronos Connects Software to Silicon

Open interoperability standards to enable software to effectively harness the power of 3D and multiprocessor acceleration

KHRONOS GROUP

Over 150 members worldwide
Any company is welcome to join

3D graphics, XR, parallel programming, vision acceleration and machine learning

Non-profit, member-driven standards-defining industry consortium

Open to any interested company

All Khronos standards are royalty-free

Well-defined IP Framework protects participant's intellectual property

Founded in 2000
>150 Members ~ 40% US, 30% Europe, 30% Asia

뉴 노멀 시대
선도를 위한
ICT 표준의
역할

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

Khronos Compute Acceleration Standards

Trained Neural Network
Exchange Format

NNEF

Import

Higher-level Languages and APIs
Streamlined development and performance portability

SYCL
Single source C++ programming with compute acceleration

OpenVX
Graph-based vision and inferencing acceleration

Lower-level APIs
Direct Hardware Control

Vulkan
GPU rendering + compute acceleration

SPIR
Intermediate Representation (IR) supporting parallel execution and graphics

OpenCL
Heterogeneous compute acceleration

Hardware

GPU

CPU
GPU
FPGA
DSP
AI/Tensor HW
Custom Hardware

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

Embedded Vision and Inferencing Acceleration

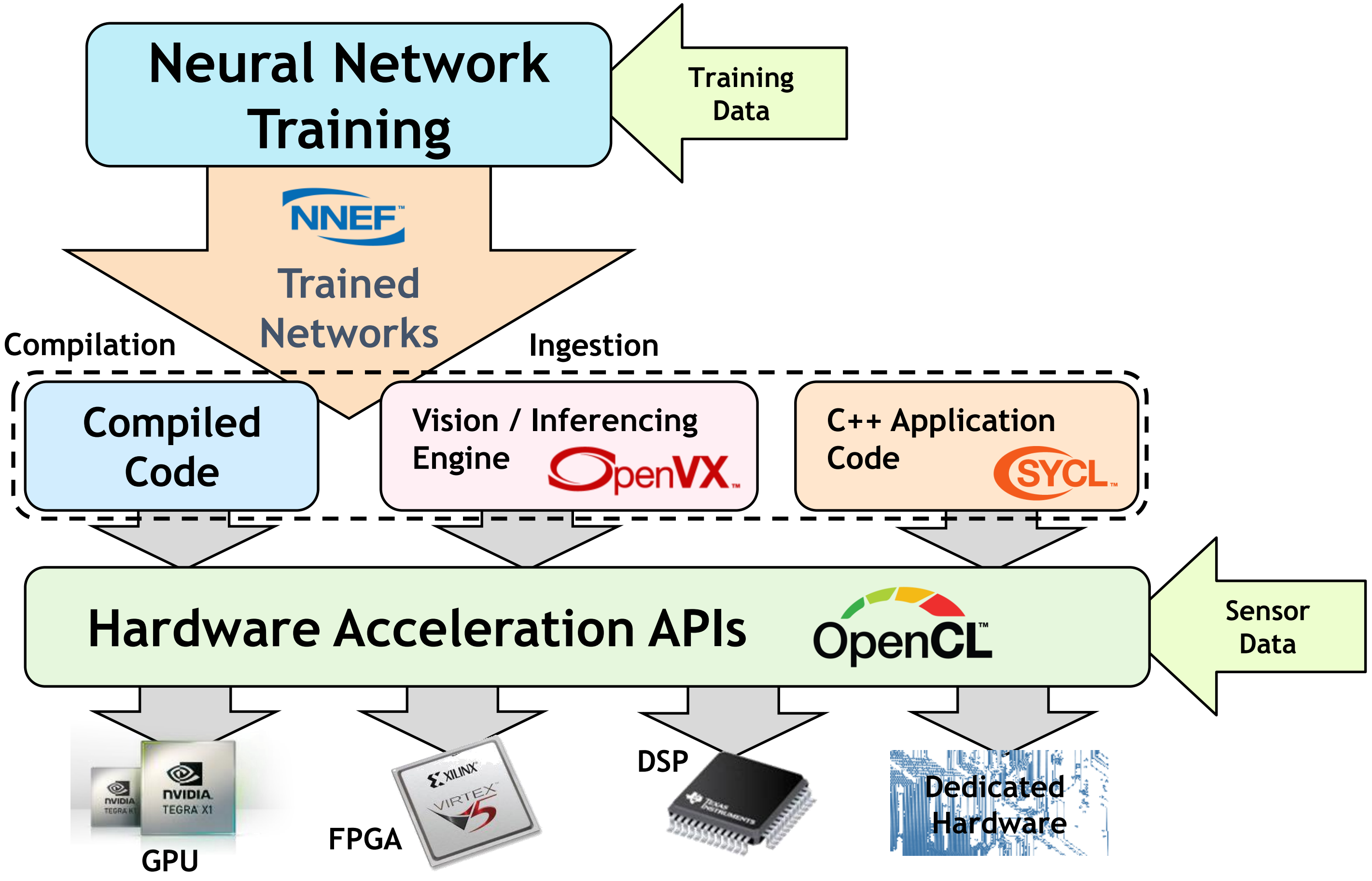
GISC2020
Global ICT Standards Conference

뉴 노멀 시대
선도를 위한
ICT 표준의
역할

Networks trained on high-end desktop and cloud systems

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

Diverse Embedded Hardware (GPUs, DSPs, FPGAs)



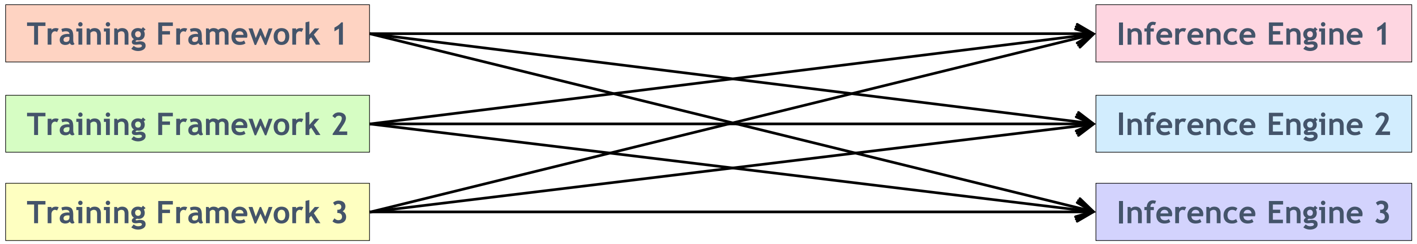
GISC2020
Global ICT Standards Conference

뉴 노멀 시대
선도를 위한
ICT 표준의
역할

NNEF Neural Network Exchange Format

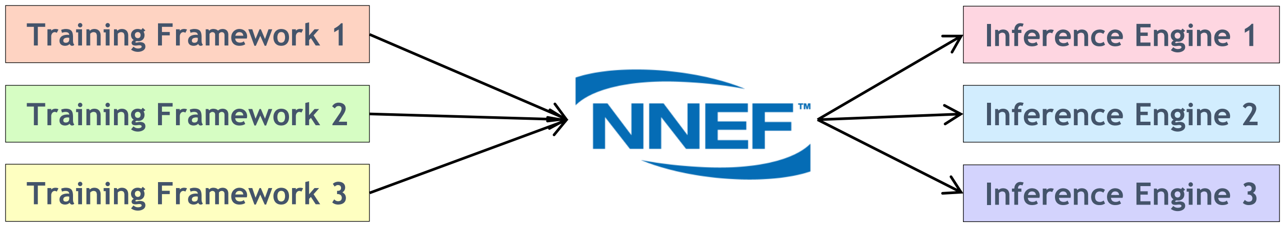
PyTorch
Caffe2
torch
mxnet
Keras
Microsoft CNTK
TensorFlow
PaddlePaddle
theano
Caffe
Chainer

Before - Training and Inferencing Fragmentation



Every Inferencing Engine needs a custom importer
from every Framework

After - NN Training and Inferencing Interoperability



Common optimization
and processing tools

NNEF and ONNX

NNEF	ONNX
Embedded Inferencing Import	Training Interchange
Defined Specification	Open Source Project
Multi-company Governance at Khronos	Initiated by Facebook & Microsoft
Stability for hardware deployment	Software stack flexibility

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 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

NNEF Working Group Participants

ONNX 1.6 Released in September 2019

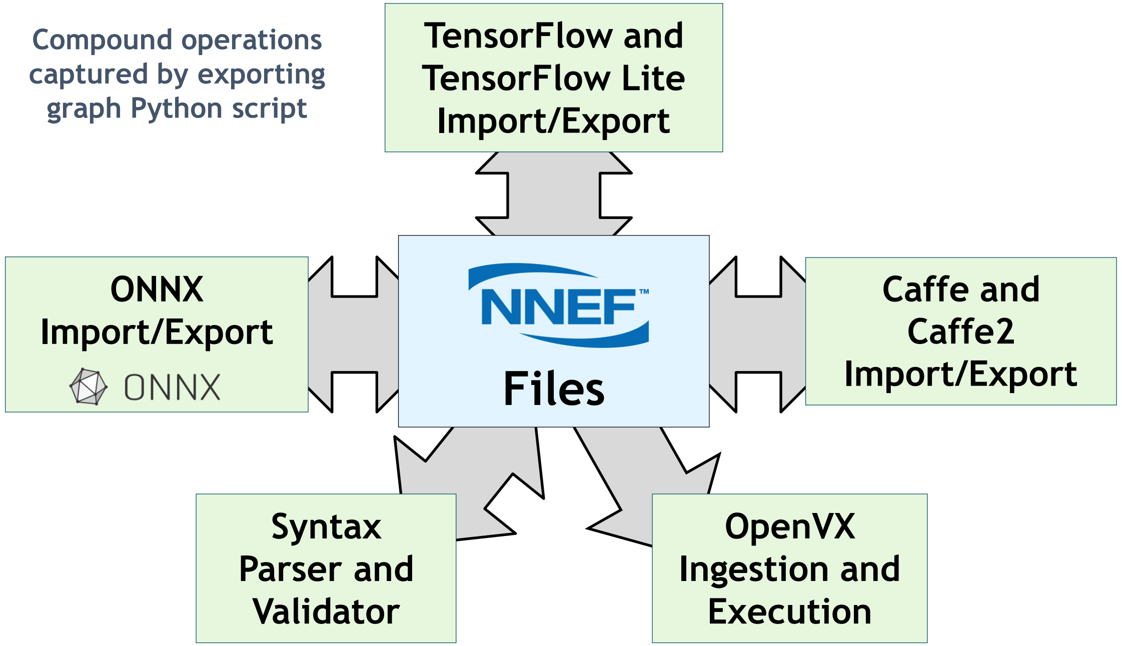
Introduced support for Quantization
ONNX Runtime being integrated with GPU inferencing engines such as NVIDIA TensorRT

ONNX Supporters

GISC2020
Global ICT Standards Conference

뉴 노멀 시대
선도를 위한
ICT 표준의
역할

NNEF Open Source Tools Ecosystem



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

NNEF

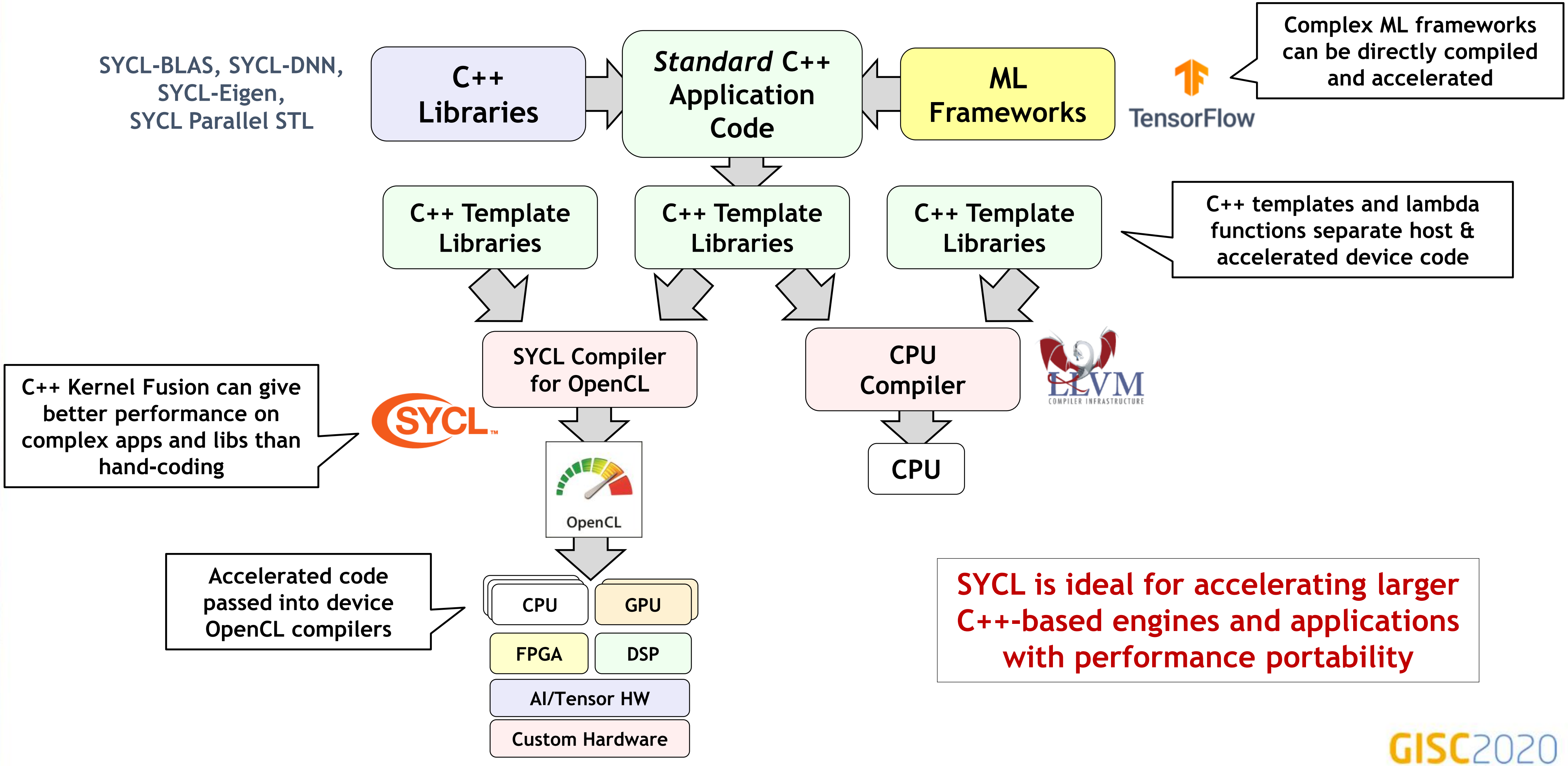
NNEF Model Zoo

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

SYCL Single Source C++ Parallel Programming

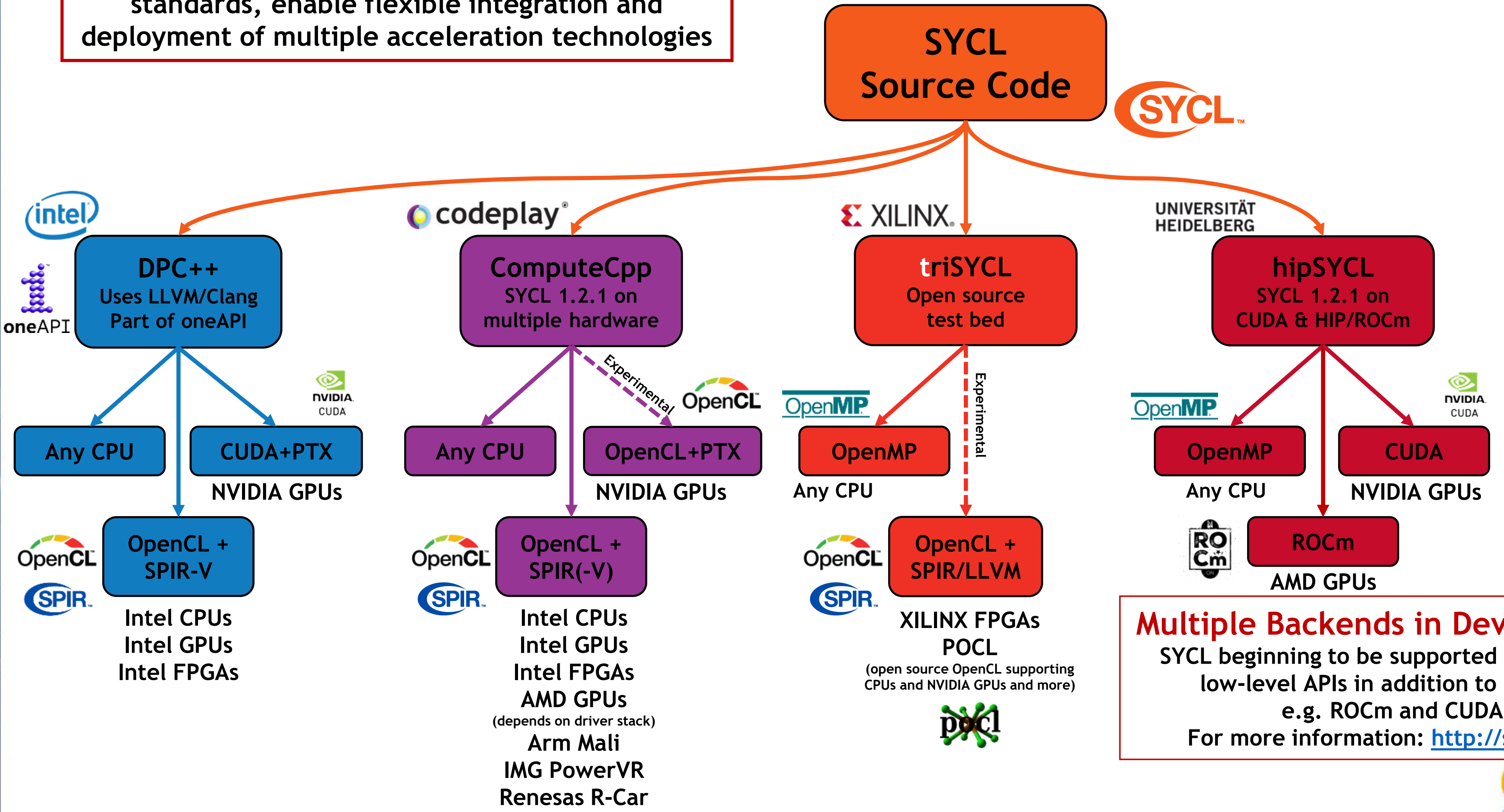


뉴 노멀 시대
선도를 위한
ICT 표준의
역할

SYCL Implementations

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

GISC2020
Global ICT Standards Conference

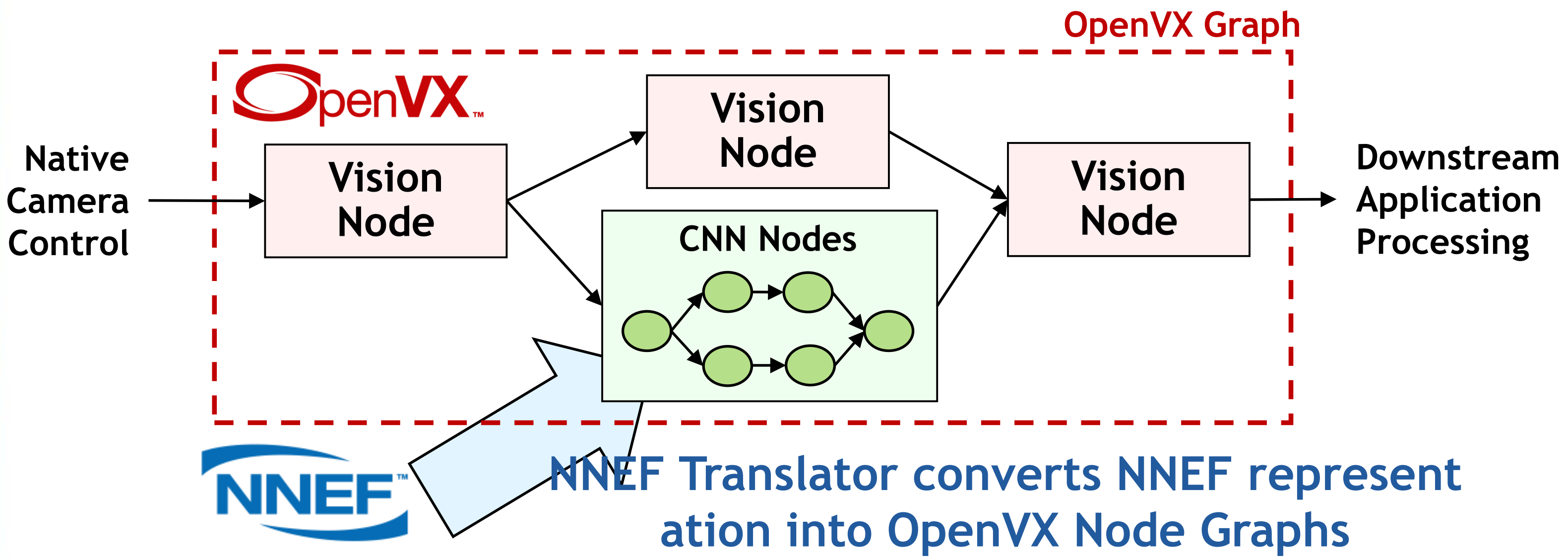
뉴 노멀 시대
선도를 위한
ICT 표준의
역할

OpenVX Cross-Vendor Vision and 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, optimized 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



AMD, cadence, ELVEES, ETRI, Imagination, intel, NVIDIA, QUALCOMM, SYNOPSYS, socionext, Texas Instruments, VeriSilicon

Hardware Implementations

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

Neural Net Extension, NNEF Kernel Import,
Safety Critical etc.

Open Source Conformance Test Suite

https://github.com/KhronosGroup/OpenVX-cts/tree/openvx_1.3

OpenCL Interop

Custom accelerated Nodes

Deployment Flexibility through Feature Sets

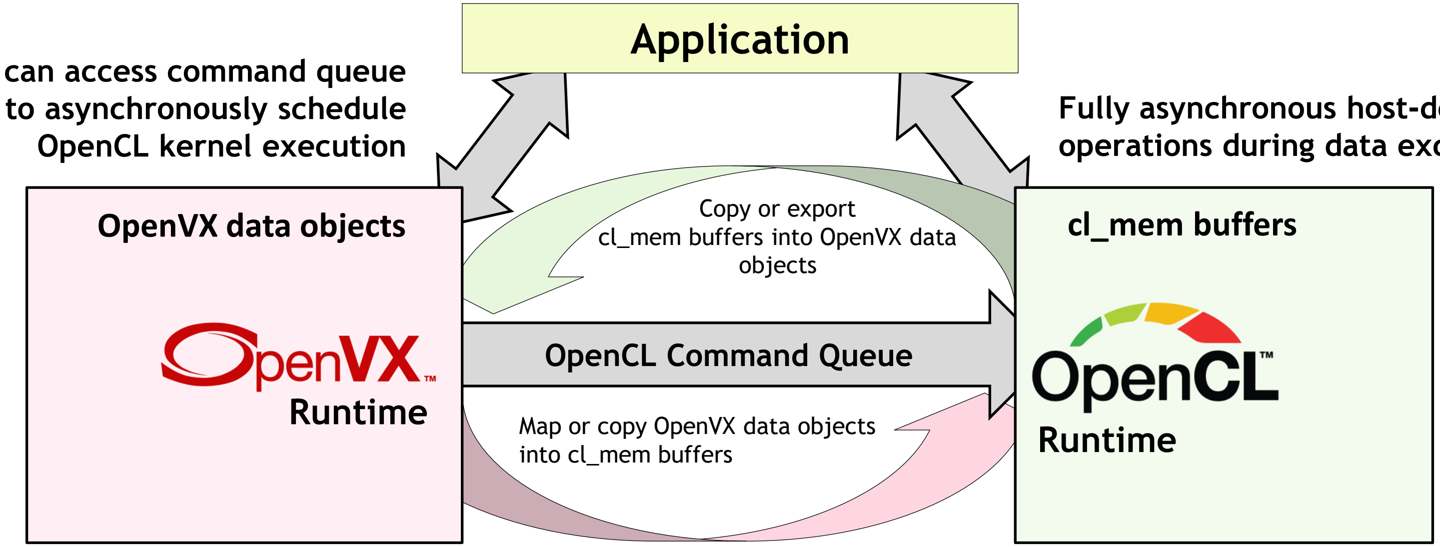
Conformant Implementations ship one or more complete feature sets
Enables market-focused Implementations

- 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)

https://www.khronos.org/registry/OpenVX/specs/1.3/html/OpenVX_Specification_1_3.html

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

Fully asynchronous host-device operations during data exchange



OpenVX/OpenCL Interop

GISC2020
Global ICT Standards Conference

뉴 노멀 시대
선도를 위한
ICT 표준의
역할

Open Source OpenVX & Samples

Fully Conformant
Open Source OpenVX 1.3
for Raspberry Pi

Raspberry Pi 3 and 4 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

"Raspberry Pi is excited to bring the Khronos OpenVX 1.3 API to our line of single-board computers. Many of the most exciting commercial and hobbyist applications of our products involve computer vision, and we hope that the availability of OpenVX will help lower barriers to entry for newcomers to the field."

Eben Upton
Chief Executive Raspberry Pi Trading

Open Source OpenVX Tutorial and Code Samples

https://github.com/rgiduthuri/openvx_tutorial
https://github.com/KhronosGroup/openvx-samples

OpenCL is Widely Deployed and Used

The industry's most pervasive, cross-vendor, open standard for low-level heterogeneous parallel programming

Desktop Creative Apps

Adobe, otoy, Dassault Systemes, Modo, SONY, darktable, Autodesk, Chaosgroup, CyberLink, ptc, Vegas Pro, Radeon ProRender, RealFlow, GIMP, Capture One, ArcSoft, Blackmagicdesign, Silhouette, blender, LuxCoreRender, SideFX, acdsee

Parallel Languages

SYCL, OpenACC (Directives for Accelerators), aparapi, PyOpenCL

Linear Algebra Libraries

SYCL-BLAS, ViennaCL, CLBlast

Machine Learning Libraries and Frameworks

Intel OpenVINO, Intel clDNN, Xiaomi MACE (Mobile AI Compute Engine), SYCL-DNN, Caffe, NNAPI, Acuity (ML/AI Software Foundation) VeriSilicon, Qualcomm Neural Processing SDK for AI, Synopsis MetaWare EV, TI DL Library (TIDL), Arm Compute Library

Molecular Modelling Libraries

siremol.org, OpenMM, CHARMM (Chemistry at HARvard Macromolecular Mechanics), GROMACS (Fast. Flexible. Free.), Folding@Home, ForceBalance

Machine Learning Compilers

tvm.ai, GLOW, plaidML

Vision, Imaging and Video Libraries

FAST, OpenCV, ImageMagick, OpenVX, VisionCpp, Metashape, Halide, FFmpeg

Math and Physics Libraries

Bullet Physics Library, GNU Octave, Wolfram Mathematica, ArrayFire (C, C++, Fortran), Matlab

Accelerated Implementations

Apple, Altera, AMD, arm, OpenCL, Imagination, intel, Marvell, MediaTek, Qualcomm, NVIDIA, Samsung, ST, Texas Instruments, VeriSilicon, Xilinx

https://en.wikipedia.org/wiki/List_of_OpenCL_applications

GISC2020
Global ICT Standards Conference

뉴 노멀 시대
선도를 위한
ICT 표준의
역활

OpenCL – Low-level Parallel Programing

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

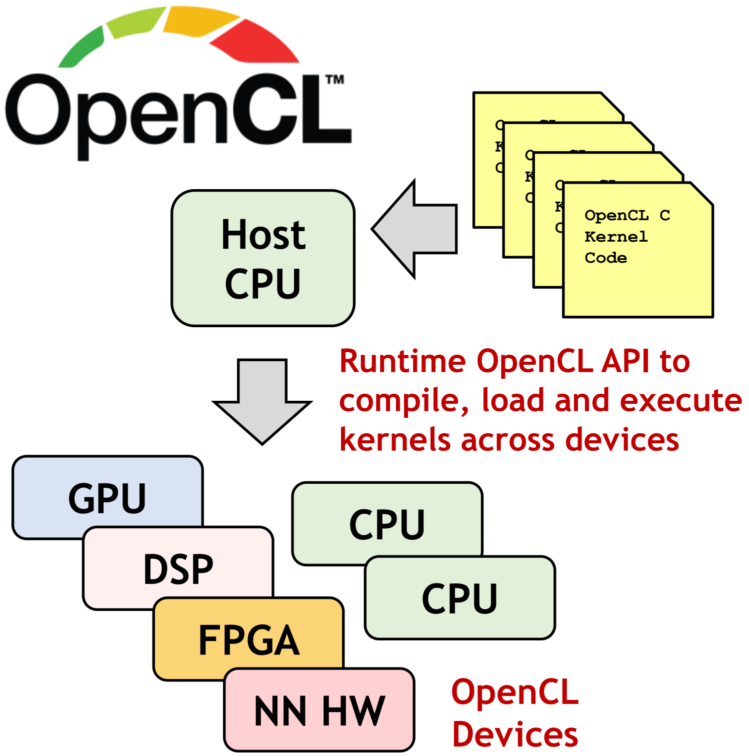
Query, select and initialize compute devices

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

OpenCL 3.0 Provisional Specification released in March 2020 for industry feedback

Increased Ecosystem Flexibility

All functionality beyond OpenCL 1.2 queryable plus macros for optional OpenCL C language features
New extensions that become widely adopted will be integrated into new OpenCL core specifications

OpenCL C++ for OpenCL

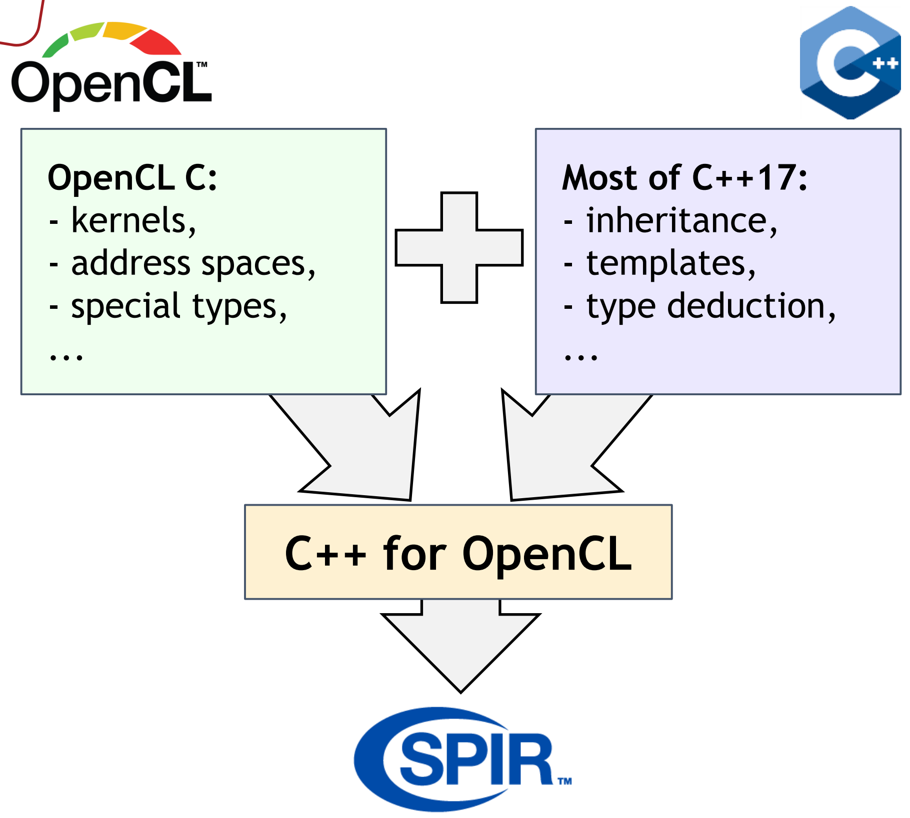
Open source C++ for OpenCL front end compiler combines OpenCL C and C++17 replacing OpenCL C++ language specification

Unified Specification

All versions of OpenCL in one specification for easier maintenance, evolution and accessibility
Source on Khronos GitHub for community feedback, functionality requests and bug fixes

Moving Applications to OpenCL 3.0

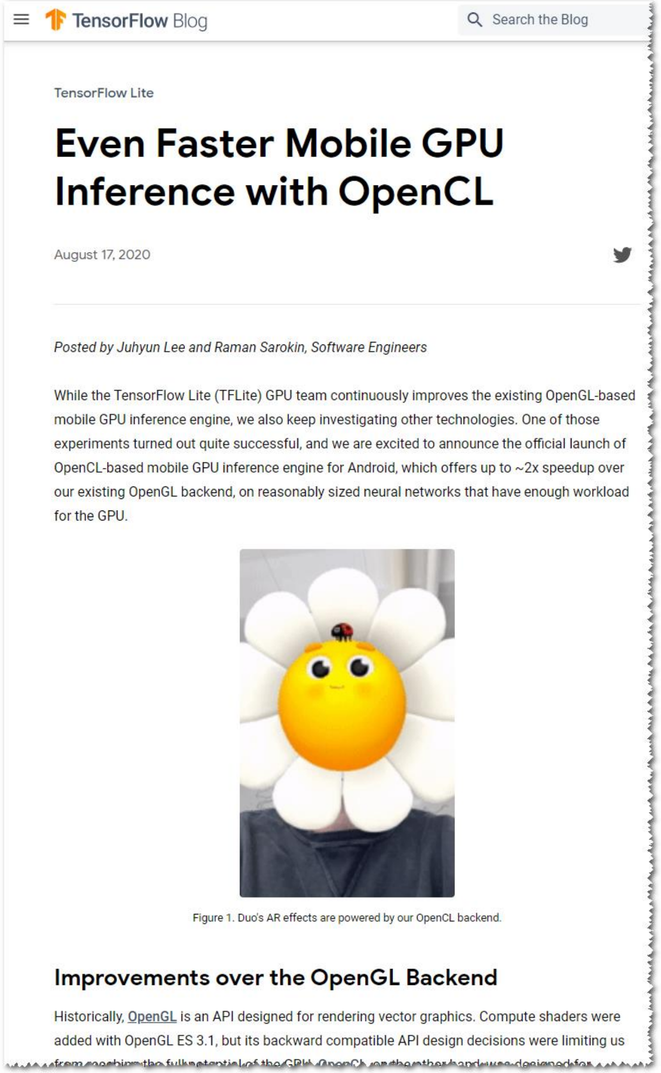
OpenCL 1.2 applications - no change
OpenCL 2.X applications - no code changes if all used functionality is present
Queries recommended for future portability



C++ for OpenCL
Supported by Clang and uses the LLVM compiler infrastructure
OpenCL C code is valid and fully compatible
Supports most C++17 features
Generates SPIR-V kernels

Google Ports TensorFlow Lite to OpenCL

뉴 노멀 시대
선도를 위한
ICT 표준의
역할



TensorFlow Blog
Search the Blog

TensorFlow Lite

Even Faster Mobile GPU Inference with OpenCL

August 17, 2020

Posted by Juhyun Lee and Raman Sarokin, Software Engineers

While the TensorFlow Lite (TFLite) GPU team continuously improves the existing OpenGL-based mobile GPU inference engine, we also keep investigating other technologies. One of those experiments turned out quite successful, and we are excited to announce the official launch of OpenCL-based mobile GPU inference engine for Android, which offers up to ~2x speedup over our existing OpenGL backend, on reasonably sized neural networks that have enough workload for the GPU.

Figure 1. Duo's AR effects are powered by our OpenCL backend.

Improvements over the OpenGL Backend

Historically, OpenGL is an API designed for rendering vector graphics. Compute shaders were added with OpenGL ES 3.1, but its backward compatible API design decisions were limiting us

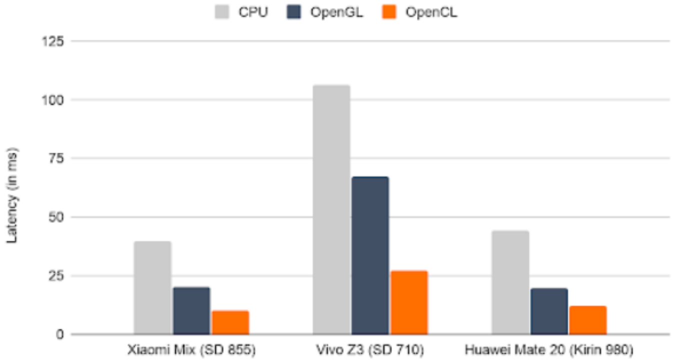


Figure 2. Inference latency of MNASNet 1.3 on select Android devices with OpenCL.

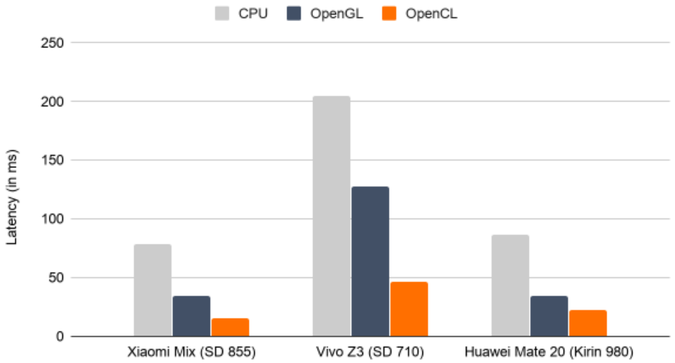


Figure 3. Inference latency of SSD MobileNet v3 (large) on select Android devices with OpenCL.

OpenCL providing ~2x inferencing speedup over OpenGL ES acceleration

TensorFlow Lite uses OpenGL ES as a backup if OpenCL not available …

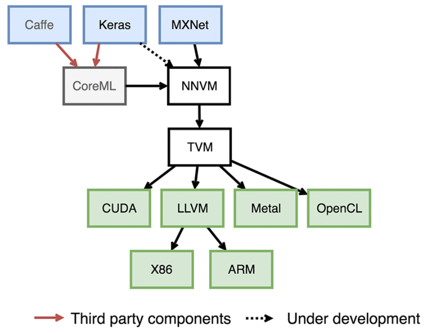
…but most mobile GPU vendors provide an OpenCL drivers - even if not exposed directly to Android developers

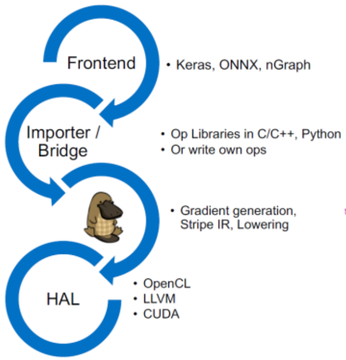
OpenCL is increasingly used as acceleration target for higher-level framework and compilers

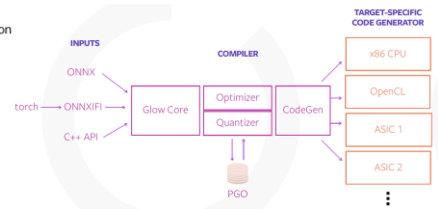
뉴 노멀 시대
선도를 위한
ICT 표준의
역할

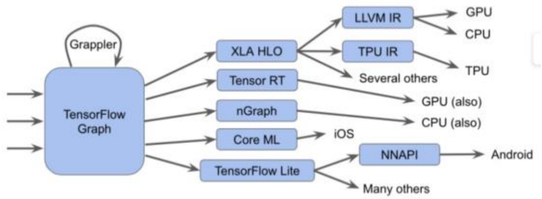
Primary Machine Learning Compilers

	tvm.ai (amazon)	plaidML (intel)	GLOW (Facebook)	XLA (Google)
Import Formats	Caffe, Keras, MXNet, ONNX	TensorFlow Graph, MXNet, PaddlePaddle, Keras, ONNX	PyTorch, ONNX	TensorFlow Graph, PyTorch, ONNX
Front-end / IR	NNVM / Relay IR	nGraph / Stripe IR	Glow Core / Glow IR	XLA HLO (MLIR)
Output	OpenCL, LLVM, CUDA, Metal	OpenCL, LLVM, CUDA	OpenCL LLVM	LLVM, TPU IR, XLA IR TensorFlow Lite / NNAPI (inc. HW accel)









ML Compiler Steps

	tvm.ai (amazon)	plaidML (intel)	GLOW (Facebook)	XLA (Google)
Import Formats	Caffe, Keras, MXNet, ONNX	TensorFlow Graph, MXNet, PaddlePaddle, Keras, ONNX	PyTorch, ONNX	TensorFlow Graph, PyTorch, ONNX
Front-end / IR	NNVM / Relay IR	nGraph / Stripe IR	Glow Core / Glow IR	XLA HLO / MLIR
Output	OpenCL, LLVM, CUDA, Metal	OpenCL, LLVM, CUDA	OpenCL LLVM	LLVM, TPU IR, XLA IR TensorFlow Lite / NNAPI (inc. HW accel)

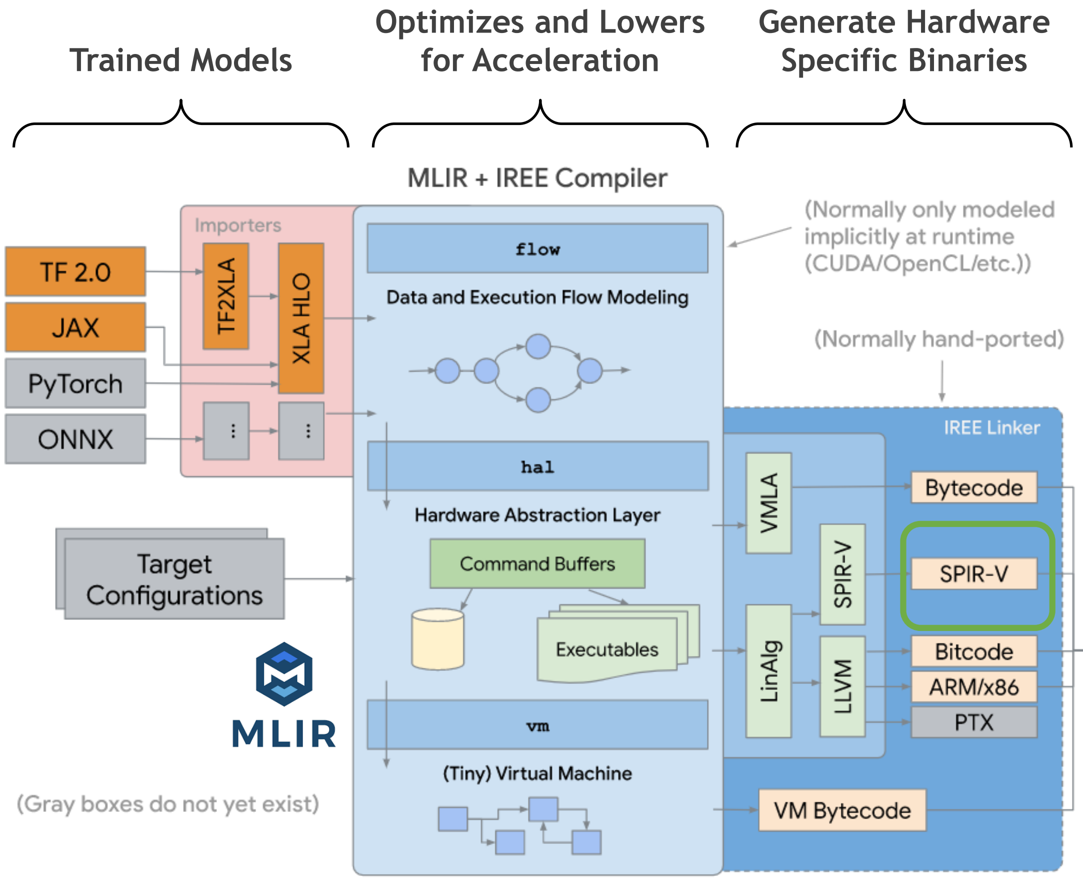
Embedded NN Compilers
CEVA Deep Neural Network (CDNN)
Cadence Xtensa Neural Network Compiler (XNNC)

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

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)

Google MLIR and IREE Compilers



IREE is a research project today. Google is working with Khronos working groups to explore how SPIR-V code can provide effective inferencing acceleration on APIs such as Vulkan

MLIR

Multi-level Intermediate Representation

Format and library of compiler utilities that sits between the trained model representation and low-level compilers/executors that generate hardware-specific code

IREE

Intermediate Representation Execution Environment

Lowers and optimizes ML models for real-time accelerated inferencing on mobile/edge heterogeneous hardware

Contains ***scheduling*** logic to communicate data dependencies to low-level parallel pipelined hardware/APIs like Vulkan, and ***execution*** logic to encode dense computation in the form of hardware/API-specific binaries like SPIR-V

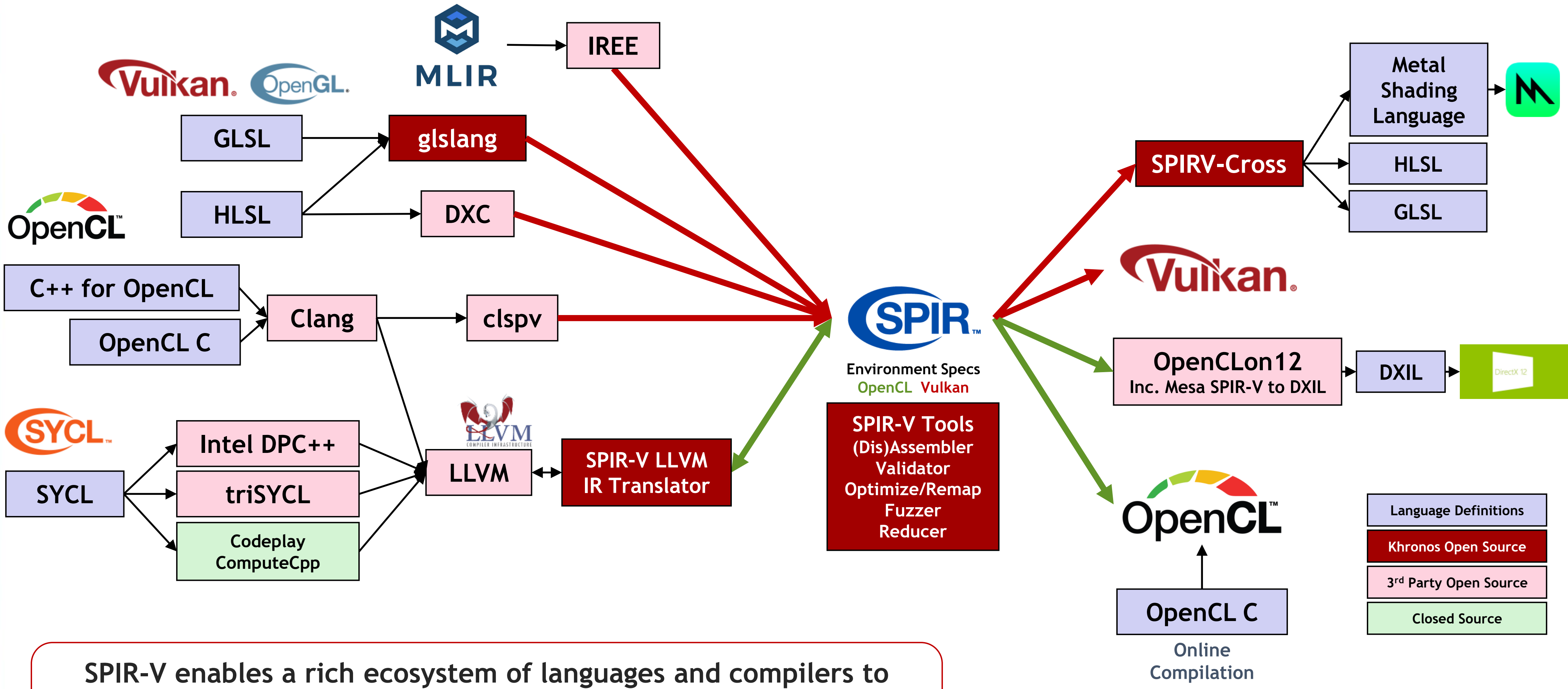
GISC2020
Global ICT Standards Conference

뉴 노멀 시대
선도를 위한
ICT 표준의
역할

GISC2020
Global ICT Standards Conference

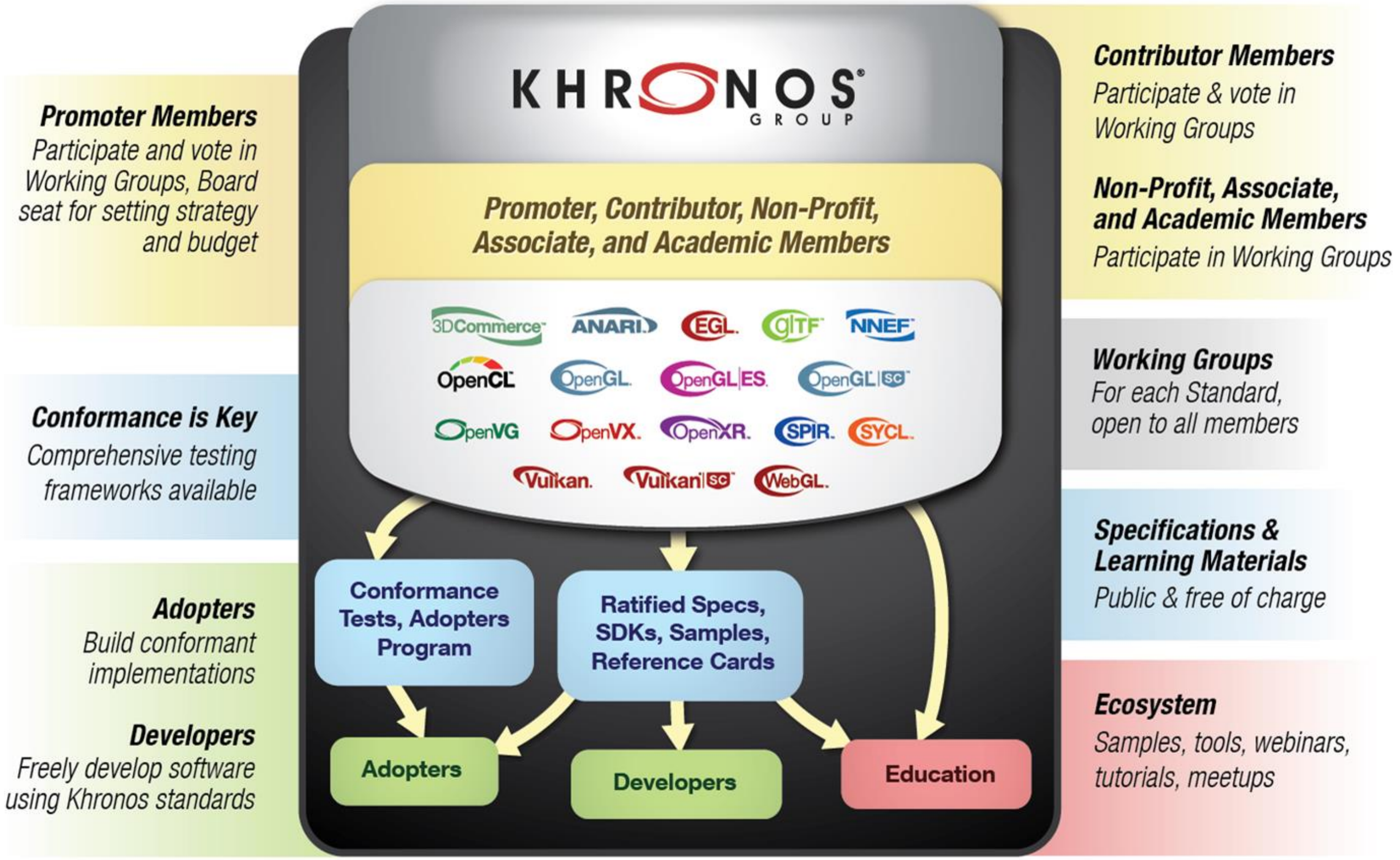
뉴 노멀 시대
선도를 위한
ICT 표준의
역할

SPIR-V Language Ecosystem



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 C kernels on Vulkan

Khronos for Global Industry Collaboration



Khronos membership is open to any company

Influence the design and direction of key open standards that will drive your business

Accelerate time-to-market with early access to specification drafts

Provide industry thought leadership and gain insights into industry trends and directions

Benefit from Adopter discounts

www.khronos.org/members/

ntrevett@nvidia.com | @neilt3d

한국담당 : 이환용

Hwanyong.lee@gmail.com

Resources

- Khronos Website and home page for all Khronos Standards
  - https://www.khronos.org/
- OpenCL Resources and C++ for OpenCL documentation
  - https://www.khronos.org/opencl/resources
  - https://github.com/KhronosGroup/Khronosdotorg/blob/master/api/opencl/assets/CXX_for_OpenCL.pdf
- OpenVX Tutorial, Samples and Sample Implementation
  - https://github.com/rgiduthuri/openvx_tutorial
  - https://github.com/KhronosGroup/openvx-samples
  - https://github.com/KhronosGroup/OpenVX-sample-impl/tree/openvx_1.3
- NNEF Tools
  - https://github.com/KhronosGroup/NNEF-Tools
- SYCL Resources
  - http://sycl.tech
- SPIR-V User Guide
  - https://github.com/KhronosGroup/SPIRV-Guide
- MLIR Blog
  - https://blog.tensorflow.org/2019/04/mlir-new-intermediate-representation.html
- IREE GitHub Repository
  - https://google.github.io/iree/