04/04/2019 mcharleb@qti.qualcomm.com Bangkok, Thailand

Qualcomm

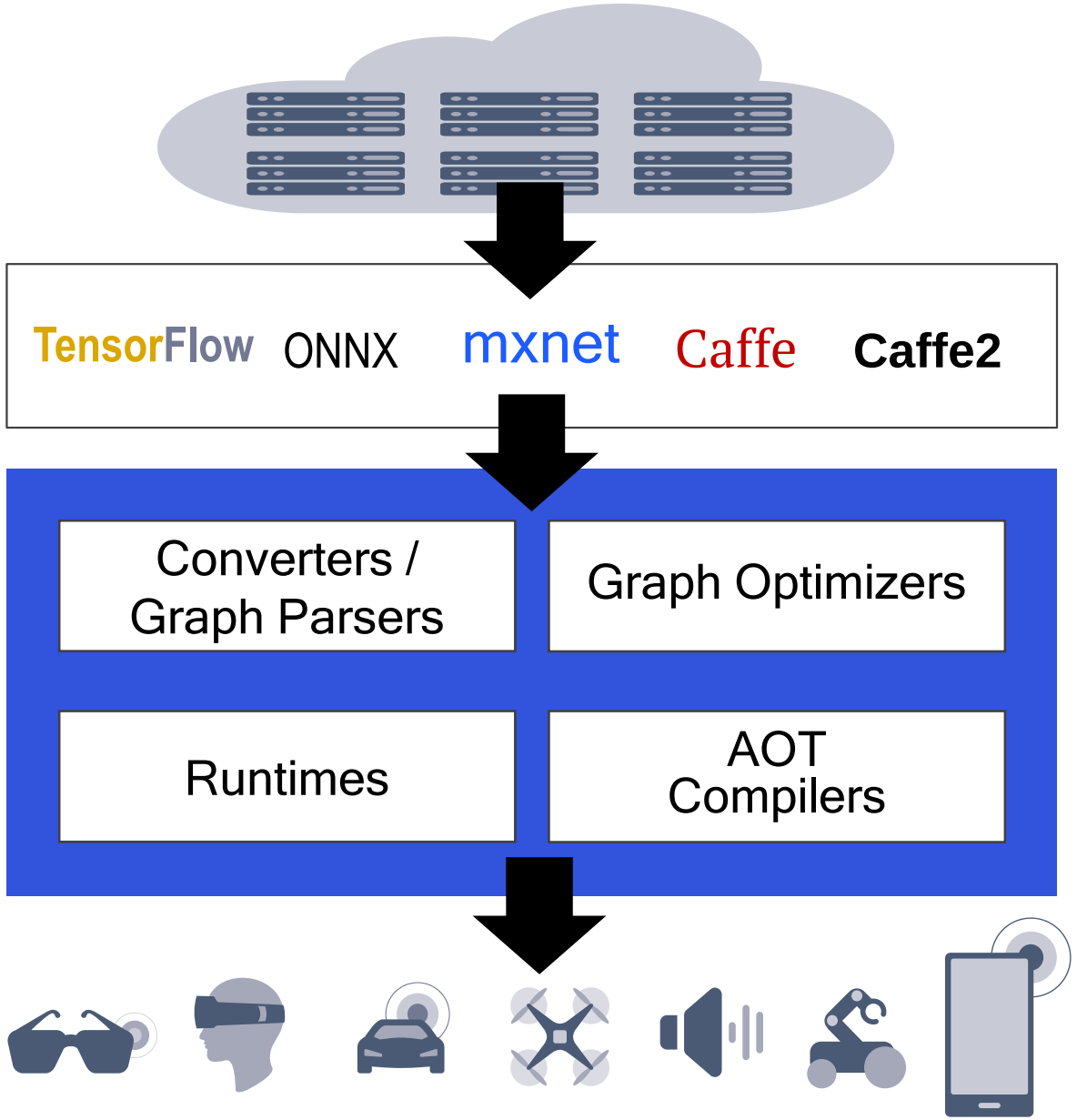
Inferencing at the Edge and Fragmentation Challenges

Mark Charlebois

Director Engineering
Qualcomm Technologies, Inc.

Agenda

- Overview of Inferencing at the Edge
  - Deployment Scenarios
  - Application Development Options
- Runtimes
- Graph Compilers
  - Four Levels of IR
- Leveraging Runtimes in Compiler Frameworks
- Conclusions

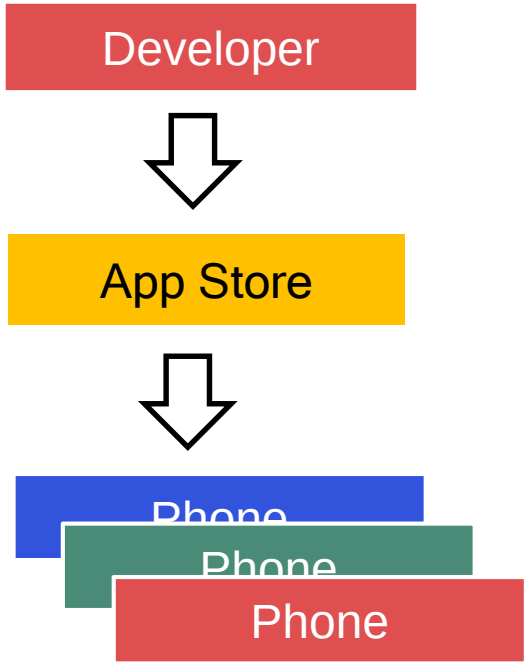


Inferencing at the Edge

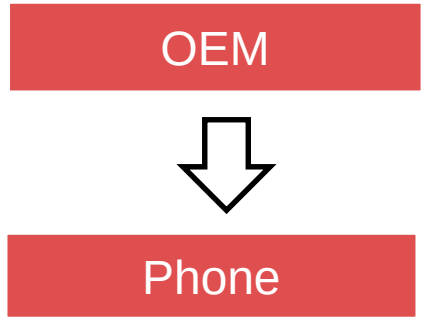
- Models developed on servers, deployed to edge devices
- Devices can range from MCUs to edge servers
- Simple networks can run on MCU
- Huge diversity of HW at the edge for inferencing acceleration
- Growing number of runtimes and graph compilers

Deployment Scenarios

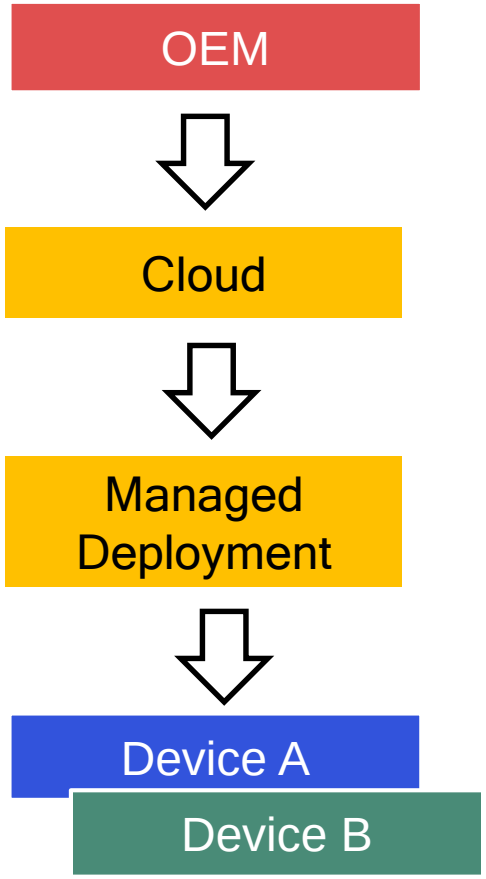
Known Models, Unknown Devices



Known Models, Known Device

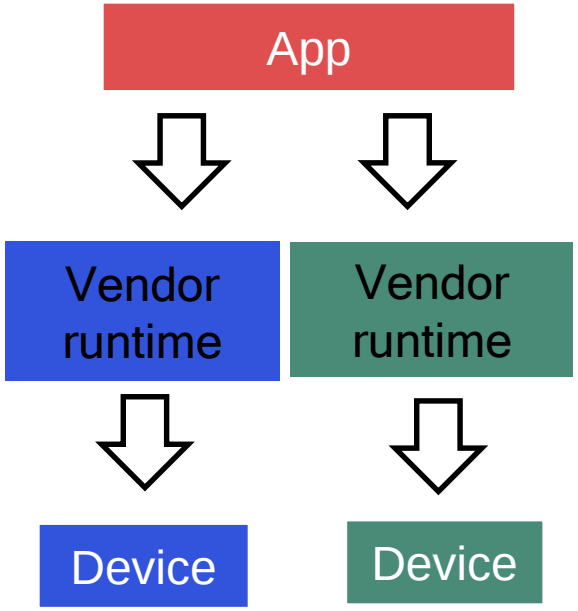


Known Models, Known Devices

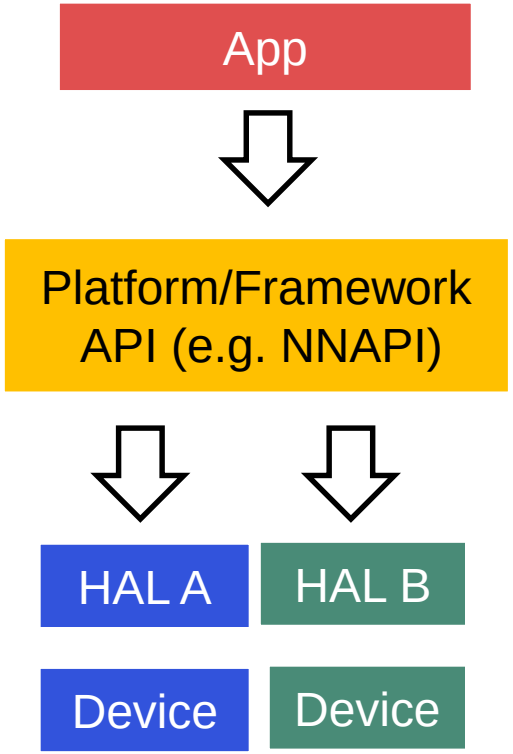


Application Development Options

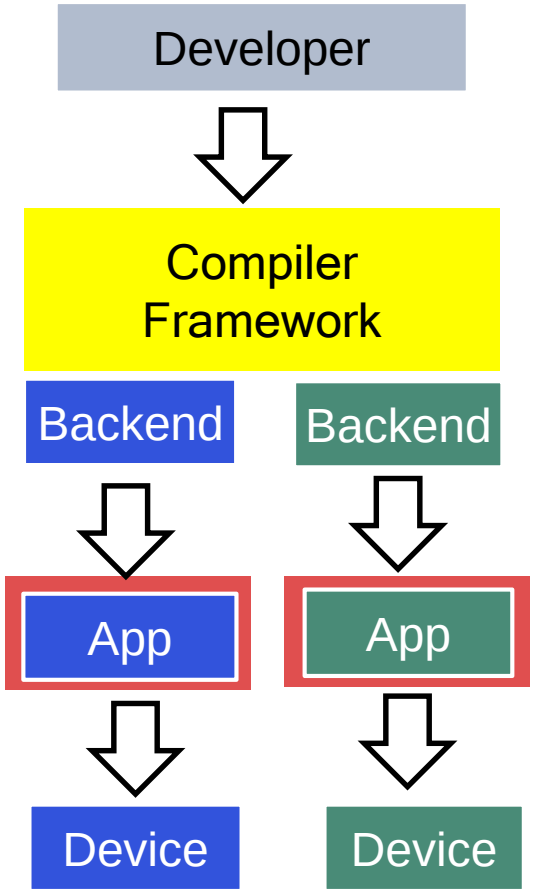
App Using Vendor SDKs



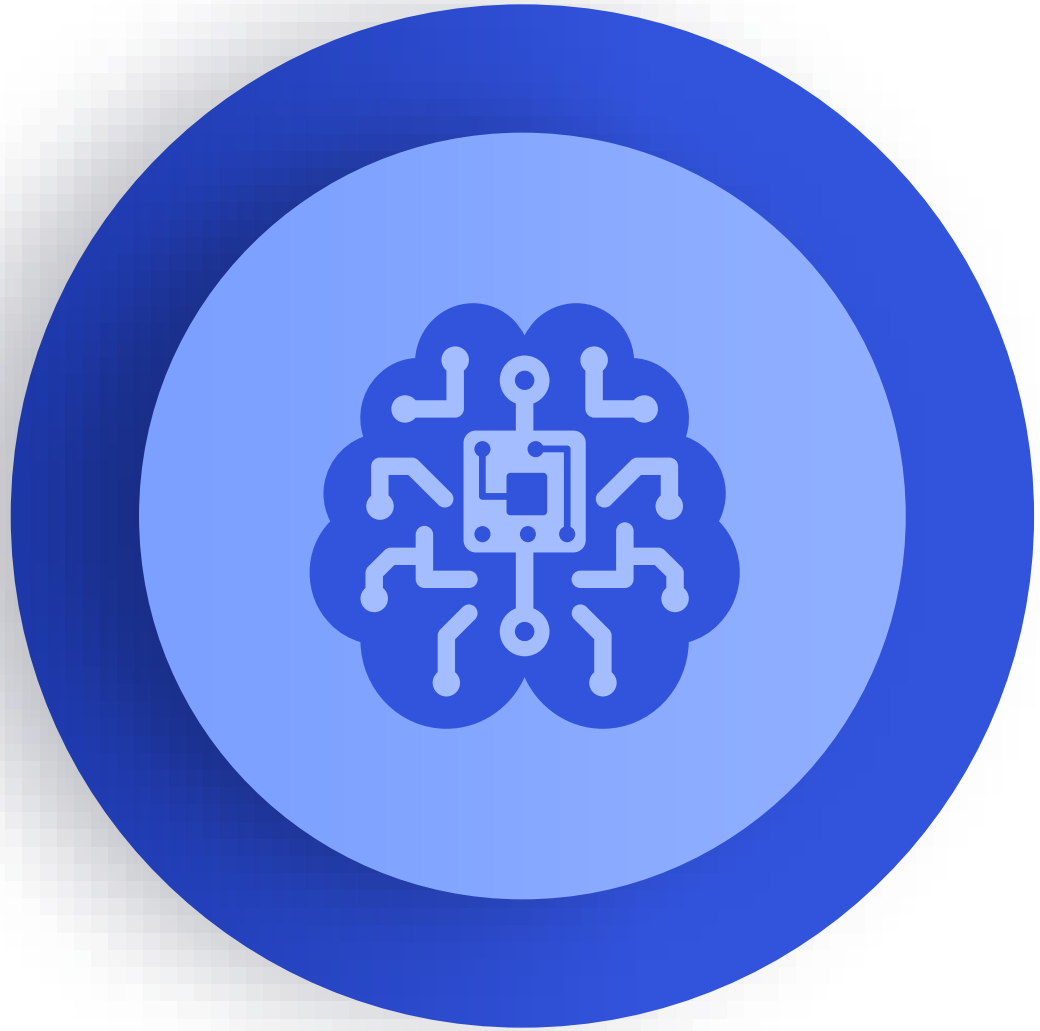
App using OS / Framework API



App Using Compiled Model

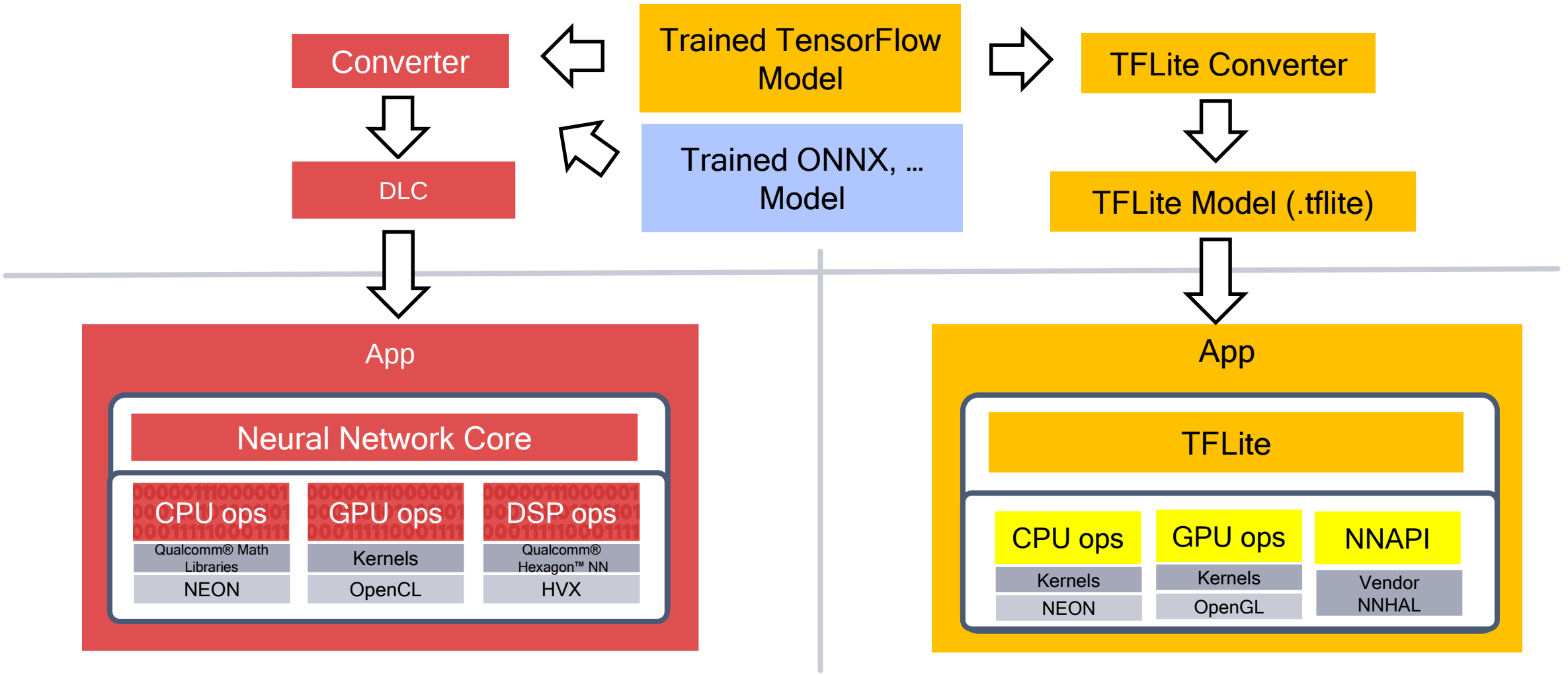


Runtimes



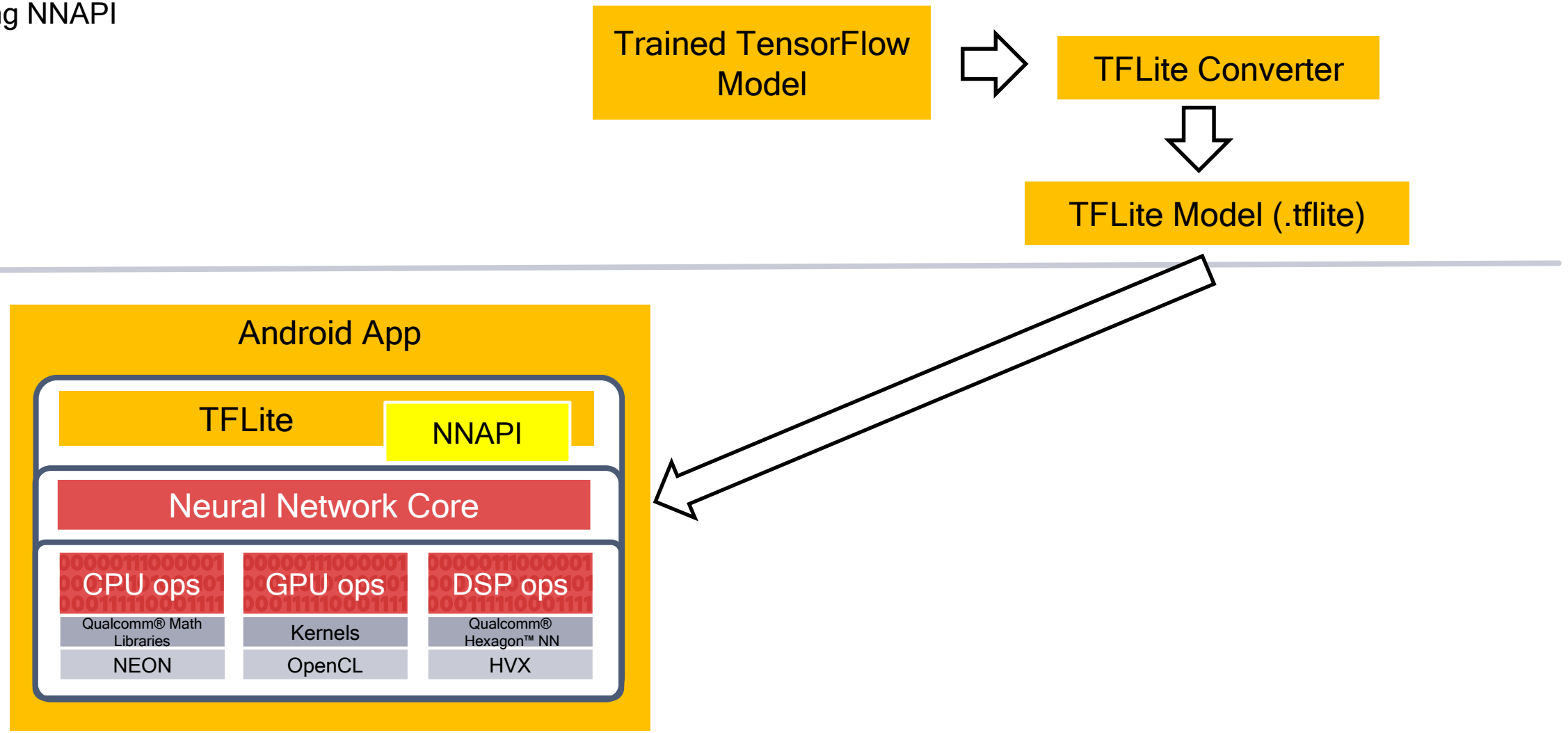
App Examples

- Qualcomm® Neural Processing SDK
- TensorFlow Lite

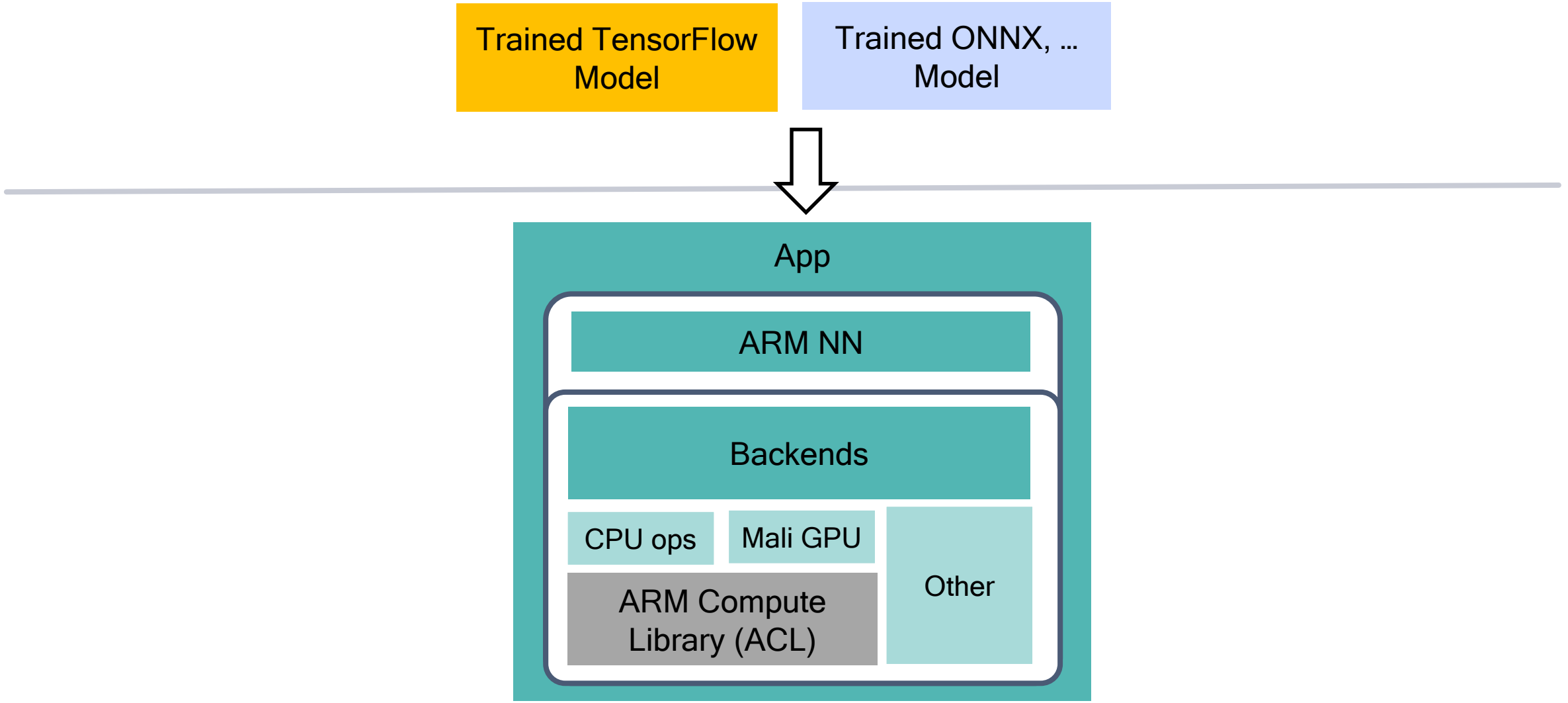


Android App Example (NNAPI)

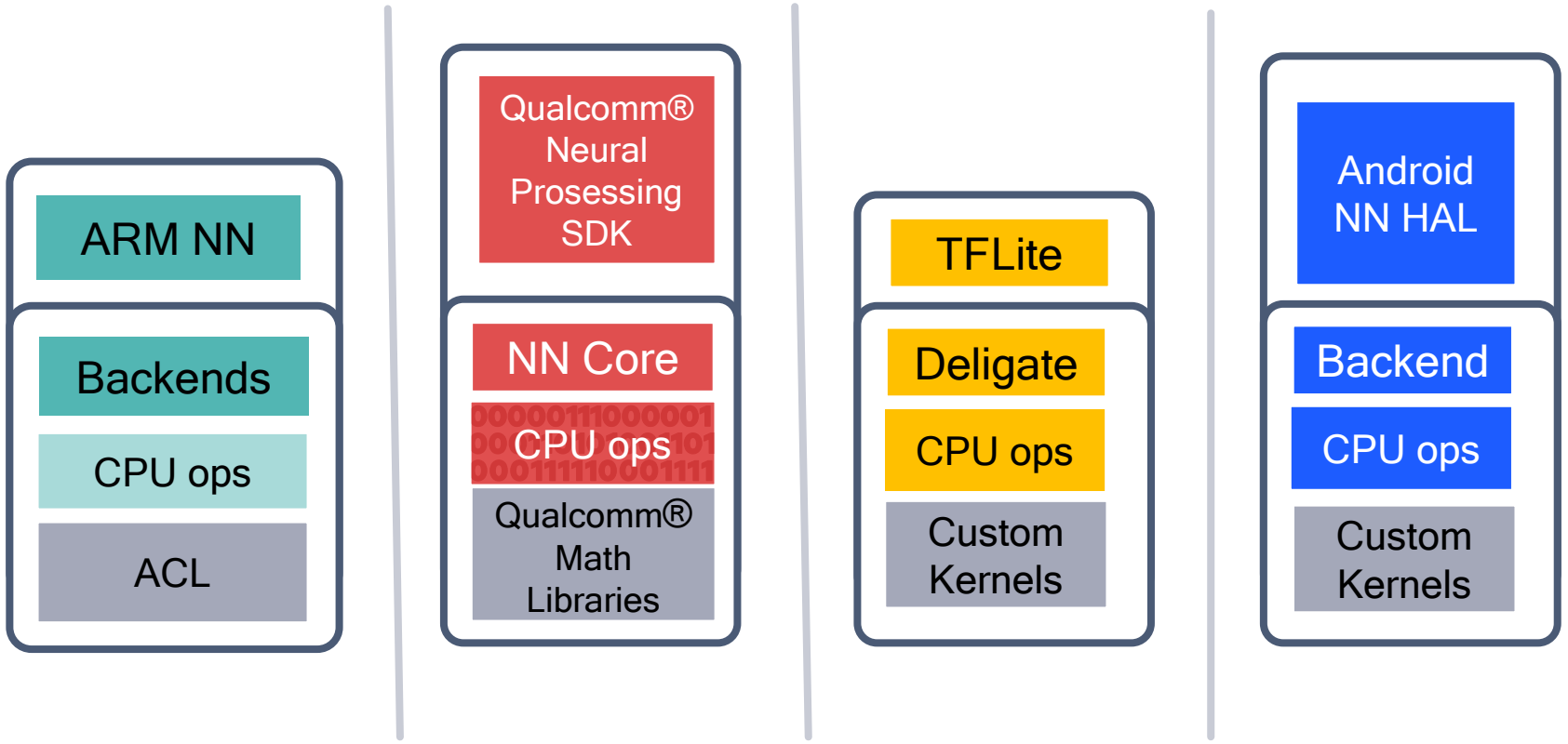
- TFLite App running on Qualcomm® device using NNAPI



ARM NN Runtime Example



CPU Runtime Fragmentation



Graph Compilers

Graph Compilers

ONNX Runtime

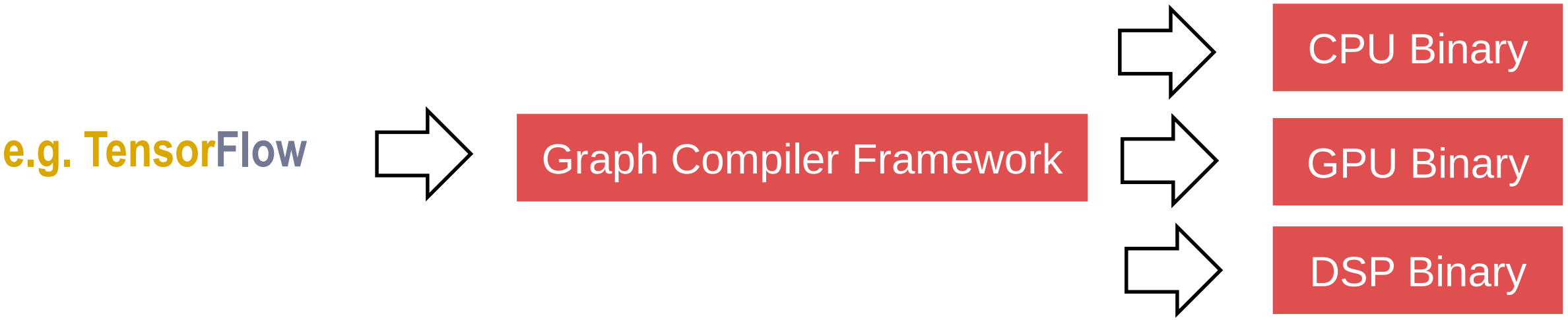
TVM

nGraph

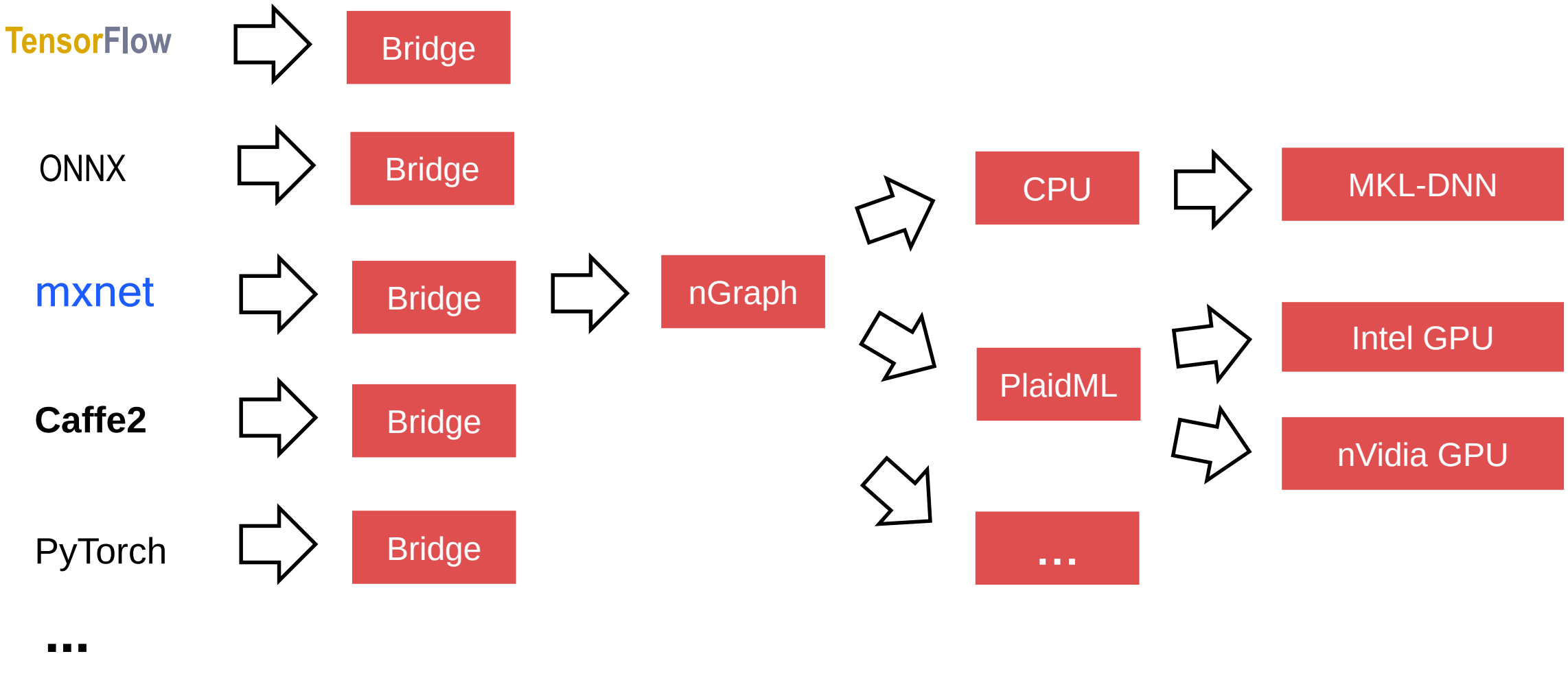
ONNC

XLA

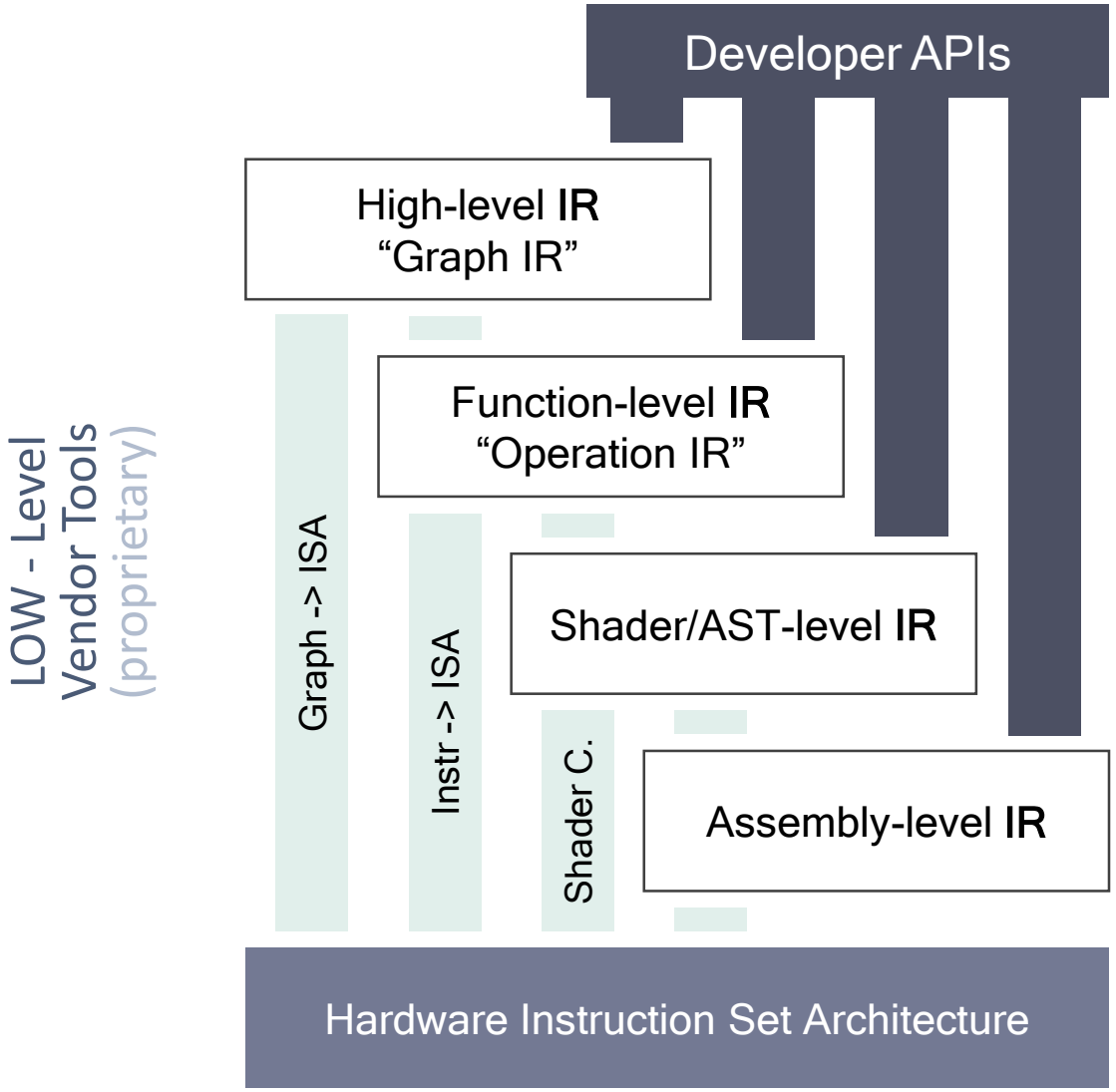
GLOW



Framework Layer Cake - Example nGraph



Four Levels of IR



- Parse Graph, first pass optimization and partitioning
- Graph segment optimization
- Memory reuse (non-backend specific)
- Layout transformation

- HW specific subgraph optimizations
- HW optimizations:
  - Explicit memory layout, parallelization pattern

- Generate shaders, DSL code, bitcode
- Integrate hand optimized kernels

- Generate assembly and schedule for target ISA

Credit: Enrico Ros, Qualcomm Technologies Inc

Projects Mapped to Four Levels of IR

TensorFlow ONNX mxnet **Caffe2** PyTorch ...

NNVM/Relay, nGraph, XLA Frontend, GLOW Graph IR	High Level IR
TVM, PlaidML, cuDNN, XLA Backend, GLOW Op IR	Operator Level IR
LLVM, OpenCL, CUDA	Shader/AST Level IR
ARMv8 Assembly, Hexagon Assembly, PTX	Assembly Level IR

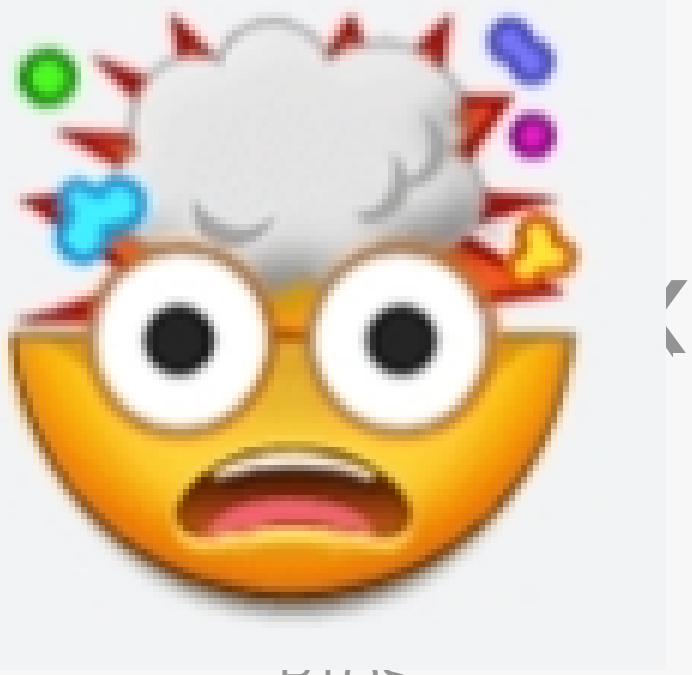
Leveraging Runtimes in Compiler Frameworks

Frameworks, Platforms, HW, All Growing

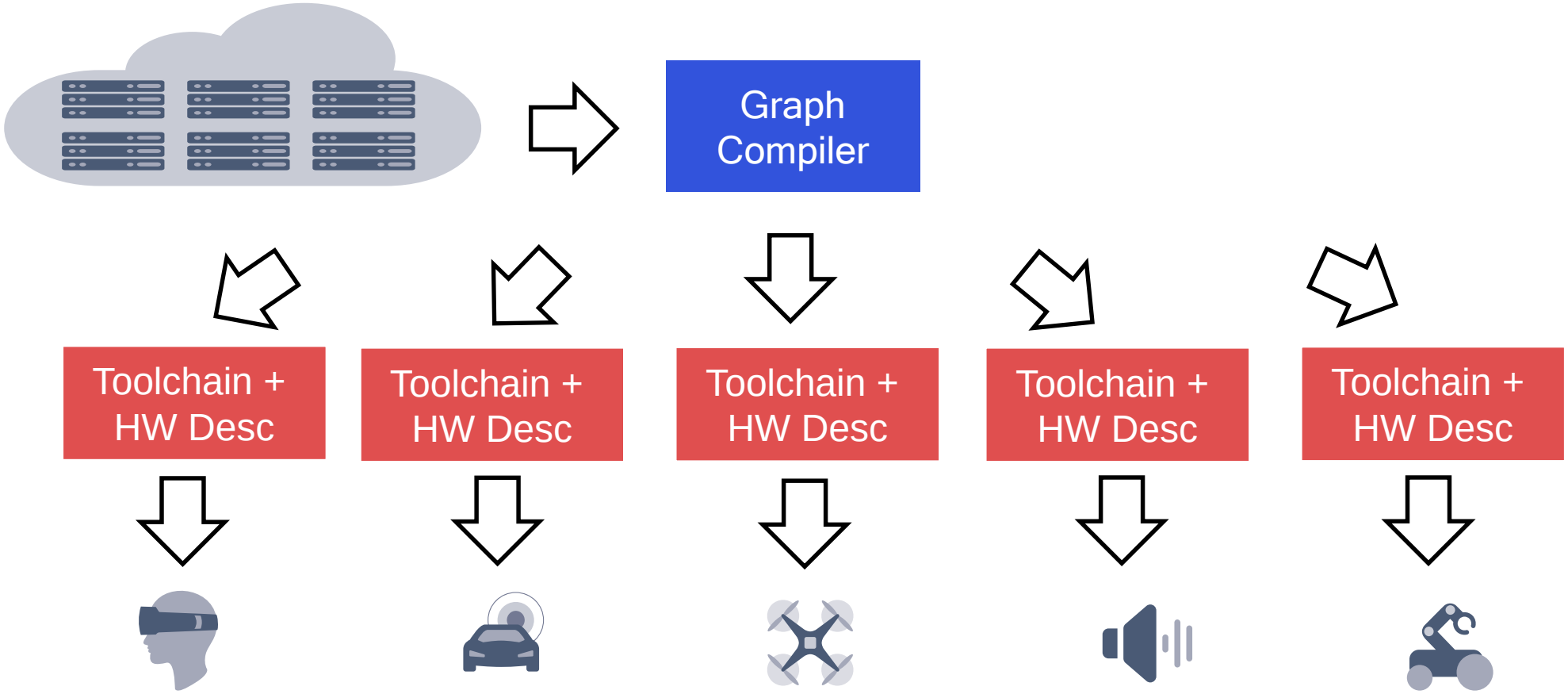
Framework		Runtime / Graph Compiler		OS		HW / Compilers	
TensorFlow	X	Qualcomm Neural Processing SDK	X	Open Embedded	X	Hexagon SDK	=
Caffe		ARM NN		Android		OpenCL	
Caffe2		TFLite		Ubuntu / Fedora / etc		CUDA	
MxNet		Relay / TVM / LLVM		Windows		ARM M4 / A53 / A76 / etc	
Pytorch		Glow / LLVM		RTOS			
...		...		...		...	

Frameworks, Platforms, HW, All Growing

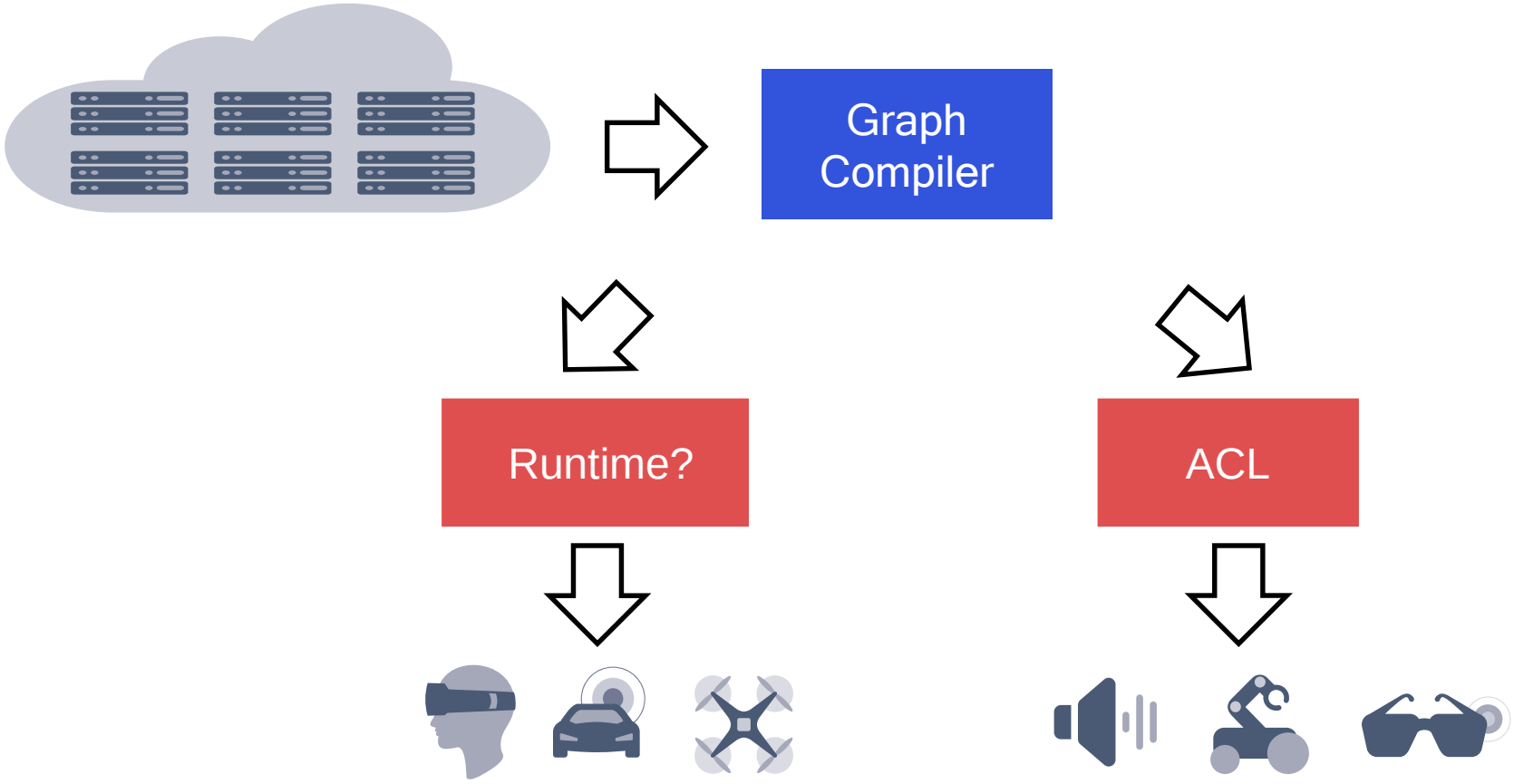
Framework		Runtime / Graph Comp		HW / Compilers	
TensorFlow	X	Qualcomm Neu Processing SD	X	Hexagon SDK	=
Caffe		ARM NN		OpenCL	
Caffe2		TFLite		CUDA	
MxNet		Relay / TVM / L		ARM M4 / A53 / A76 / etc	
Pytorch		Glow / LLVM		RTOS	
...		...		...	



HW Diversity Is a Scalability Issue

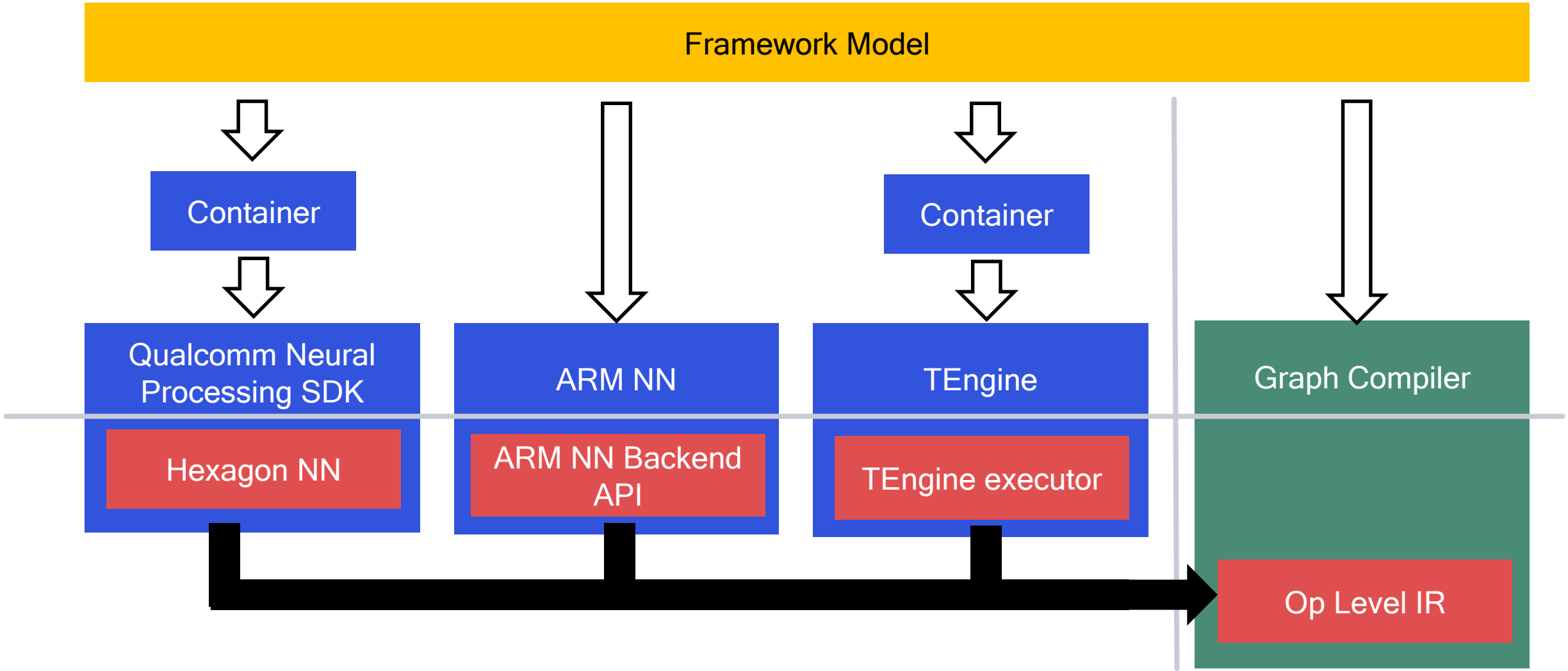


Tradeoff Some Performance for Scale

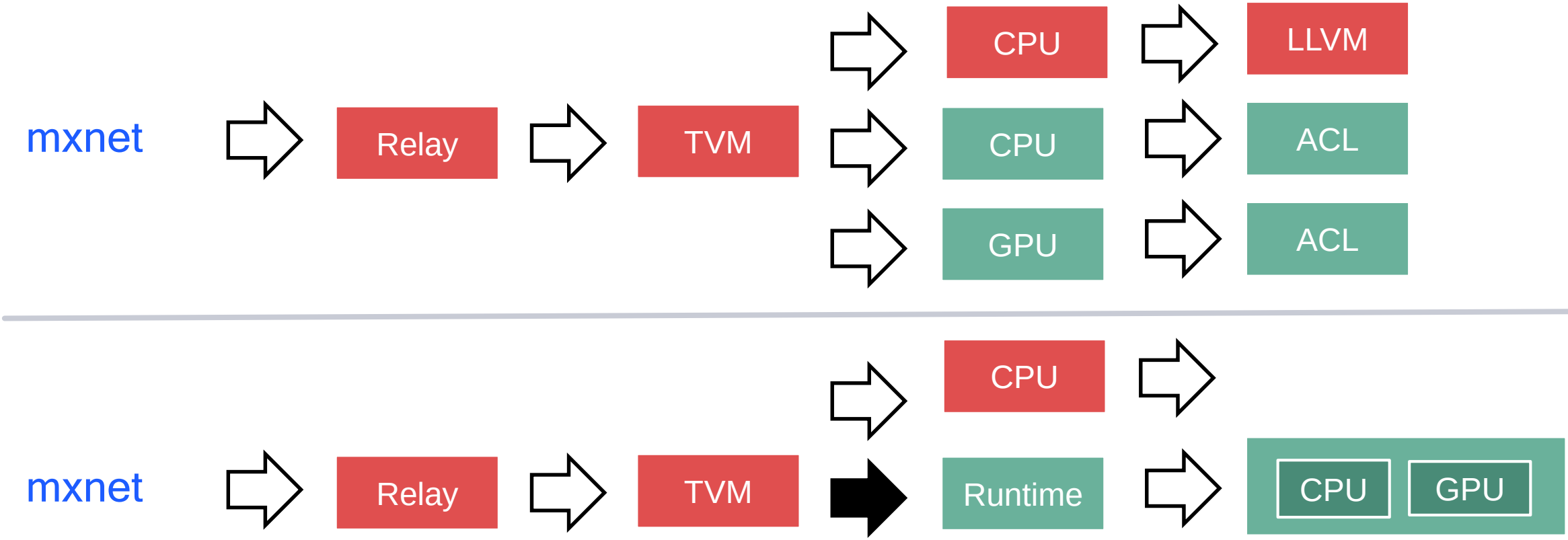


Conclusions

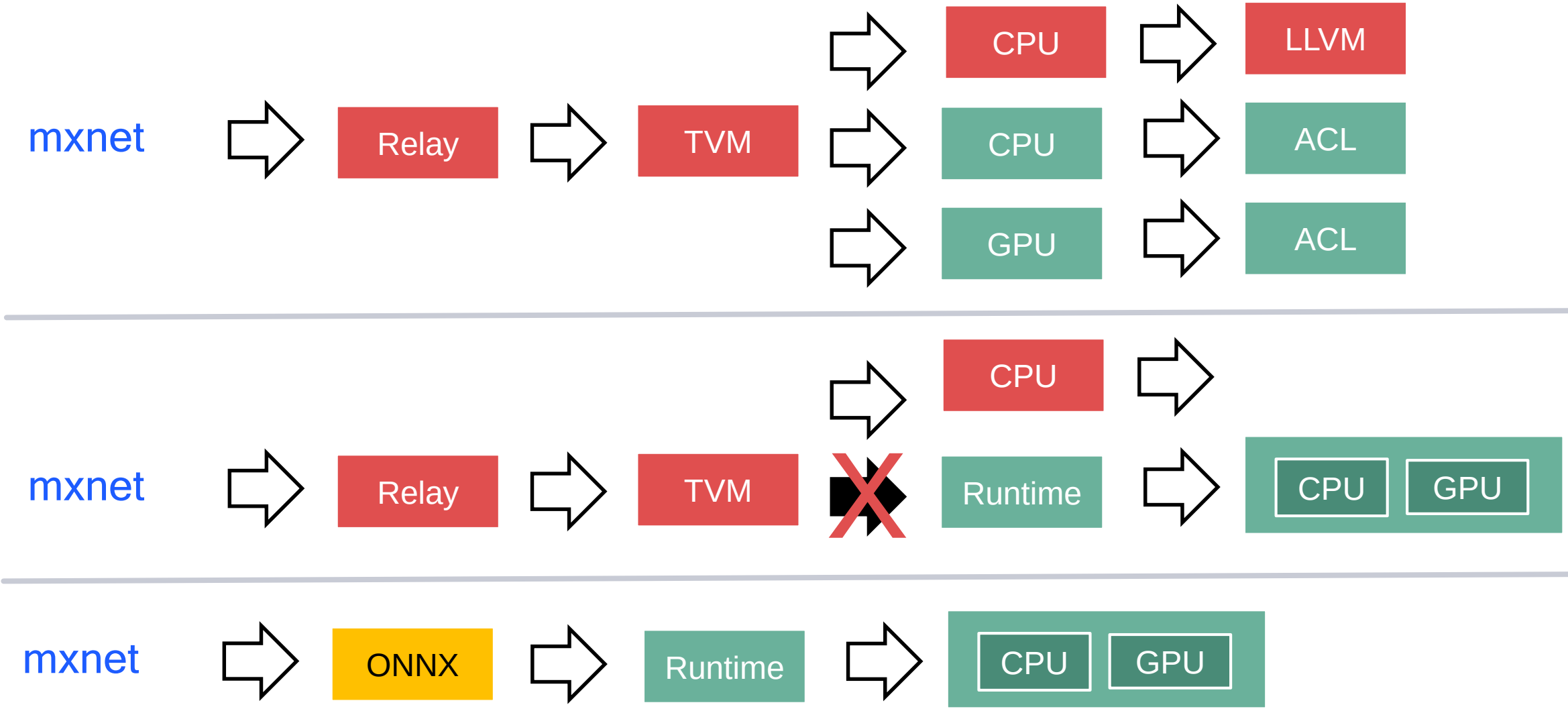
Runtime Components Map to Op Level IR



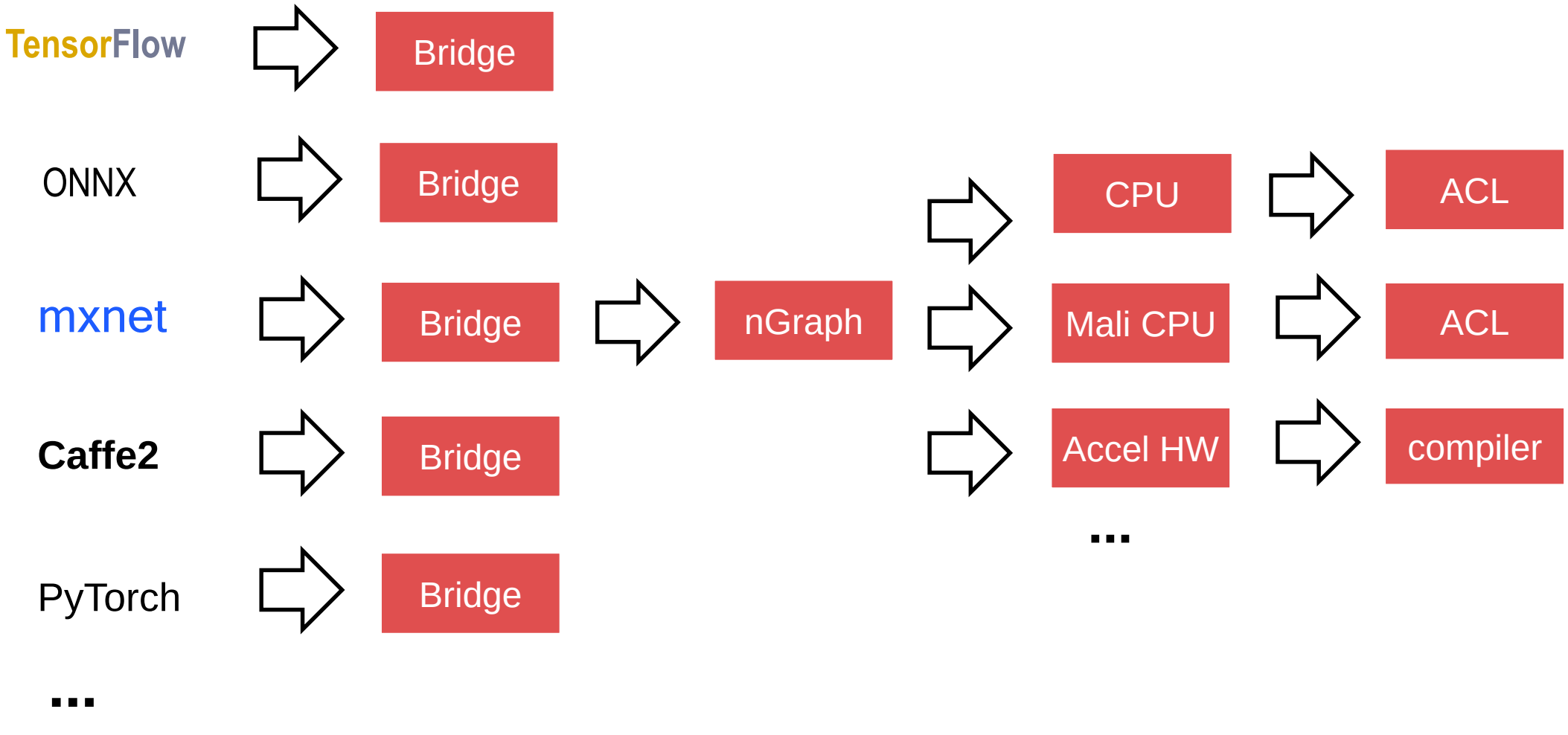
Using Full Runtime as a HW Backend



Using Full Runtime as a HW Backend



ARM Inferencing Edge Server



Addressing Graph Compiler IR Fragmentation

- Each compiler framework makes different tradeoffs, but eventually may be able to share components: e.g. code generation for OpenCL, LLVM, etc
- Too many frameworks, compilers and formats to address them all.
- Too much disparate HW for all frameworks to support.
- TensorFlow/TFLite and ONNX formats can provide the most scale for edge device inferencing runtimes.
- If Compiler Frameworks supported a common runtime backend API (like ARM NN Backend API) to bind to operator IR would enable graph compilers to support more edge devices with optimized backends, and would provide a common API for new backends (e.g. Hexagon NN) vs individual ports to each project.

Addressing CPU Runtime Fragmentation

- Make ACL the “best of breed” CPU Runtime
- Consolidate TFLite CPU runtime, Android NN CPU runtime, and ARM NN CPU backend
- Paves the way for others to follow: TEngine, MACE, Qualcomm® Neural Processing SDK, …

Qualcomm

Thank you!

Follow us on:

For more information, visit us at:

www.qualcomm.com & www.qualcomm.com/blog