ECLIPSE FOUNDATION

X

list
CEA tech

AICE Working Group Online Meeting

June 10, 2022

Agenda

- Introduction
- AICE WG News
- CEA List N2D2
- Next steps & Q&A

ECLIPSE FOUNDATION

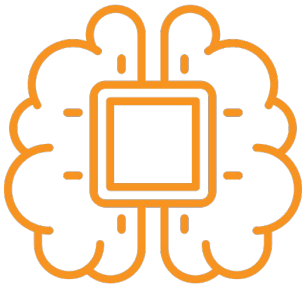
Agenda

- Introduction
- AICE WG News
- CEA List N2D2
- Next steps & Q&A

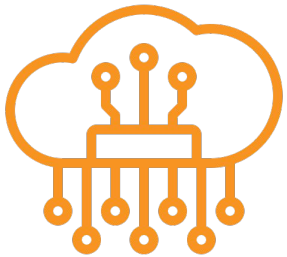
ECLIPSE FOUNDATION

Three new prospective strategic areas meet in the AICE OpenLab

Artificial Intelligence



Cloud



Edge

Requirements

- Testbeds and tools
- To Trust AI
- OSS is necessary
- Interoperability
- Frugality / Embedded
- AI OSS ecosystem
- Predictability
- Benchmarks

Mission AICE Working Group

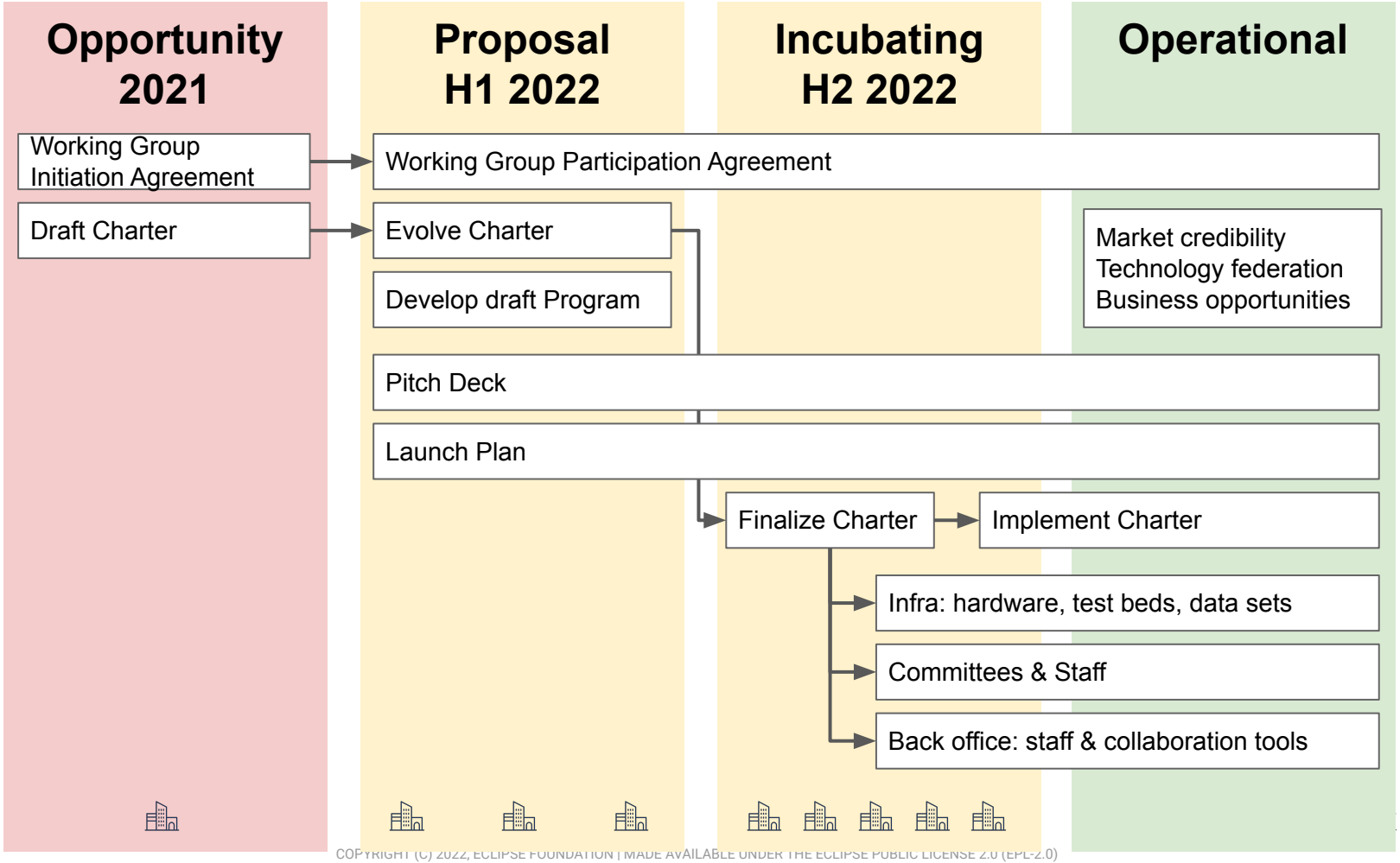
Promote the advancement, development and experimentation of open source software for *AI, Cloud & Edge* technologies

- Foster vendor neutral collaboration in AI, Cloud and Edge open source technologies
- Deliver verified reference architectures, blueprints and distributions
- Provide test suites, test tools, calibrated demo datasets
- Setup and operate the AICE OpenLab, a dedicated experimental infrastructure
- Ensure privacy, security, ethics and frugality requirements integrated in OpenLab activities

Agenda

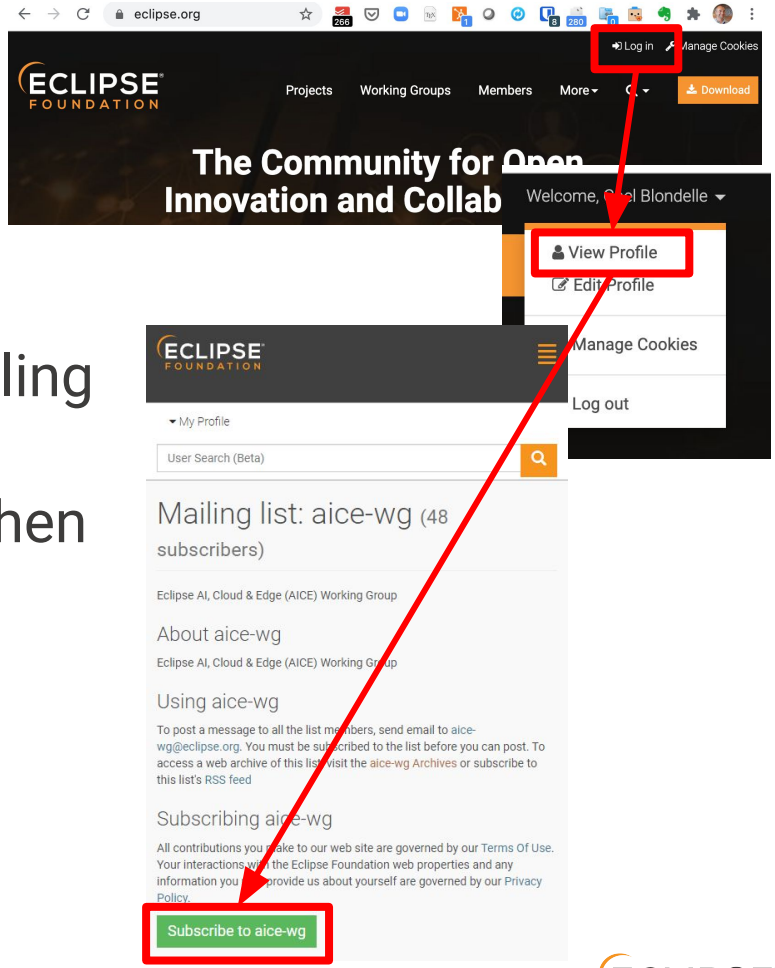
- Introduction
- AICE WG News
- CEA List N2D2
- Next steps & Q&A

ECLIPSE FOUNDATION



Join the ML

- Create an Eclipse.org Account
- Go to your Profile page then the Mailing List Tab
- Click on Manage your Mailing lists then search for AICE
- Click on Subscribe to aice-wg



Thank you to our early supporters

- AURA Healthcare
  - https://en.aura.healthcare
- University of Skövde
  - https://www.his.se/en
- Synesthesia
  - https://synesthesia.it
- Fraunhofer Fokus
  - https://www.fokus.fraunhofer.de
- Noosware
  - https://noosware.com
- Sustainable Digital Infrastructure Alliance
  - https://sdialliance.org
- ATB – Institut für angewandte Systemtechnik Bremen GmbH
  - https://www.atb-bremen.de

Help us grow the ecosystem

Code first Approach

Bring your projects & Use Cases

- Platform projects
- Vertical application frameworks
- Use cases / Testbeds / demonstrators
- Research project results for better dissemination and exploitation

Sponsor computing power for the OpenLab

- Provided by SDIA for the moment

Point us to potential partners / projects

- You are our best ambassadors!

Help us frame the Working Group

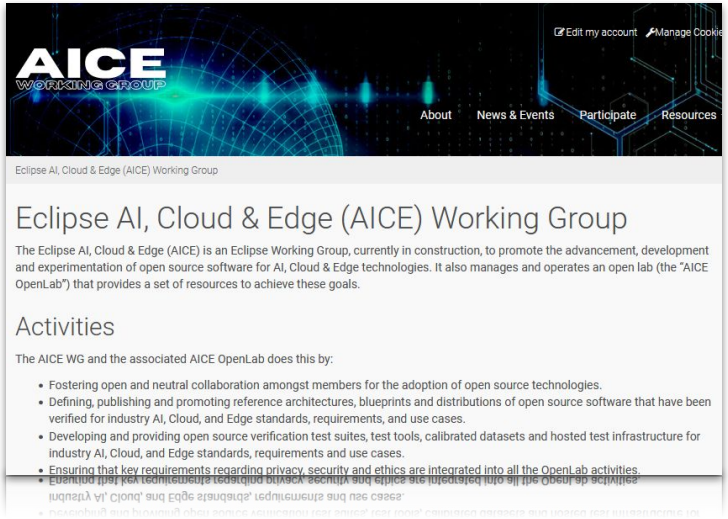
- Your requirements are the best!

AICE website

🔗 https://aice.eclipse.org

More content available

- The objective of the WG
- Testimonial
- How-to join / participate
- News and Event
- A resources section with
  - An updated version of the AURA demonstrator paper
  - Videos and slides from previous events
- Registration to the Mailing-list

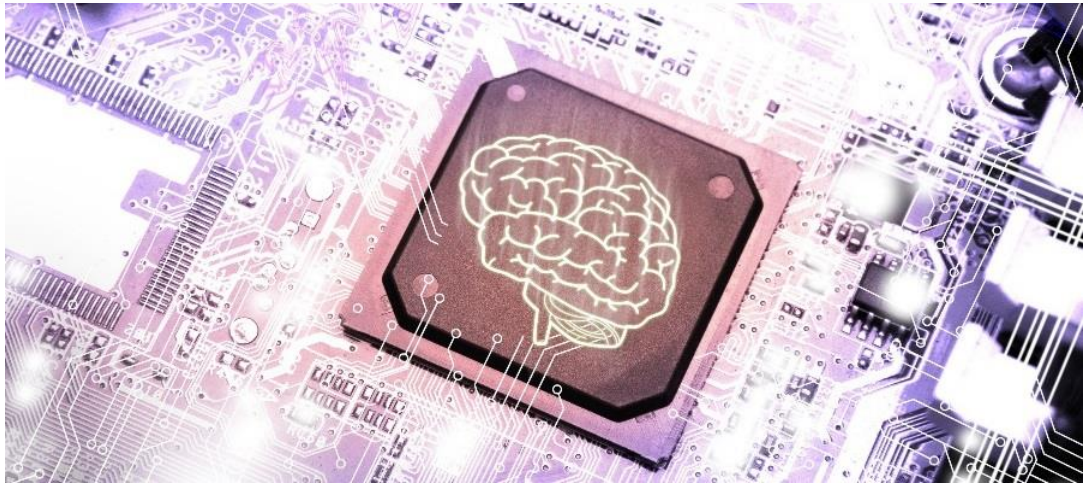


New, more modern, eye-candy and dedicated design forecasted in Q3

Agenda

- Introduction
- AICE WG News
- CEA List N2D2
- Next steps & Q&A

ECLIPSE FOUNDATION



Neural Networks Design and Deployment for Constrained Embedded Systems with N2D2 Framework

LIAE | 2022

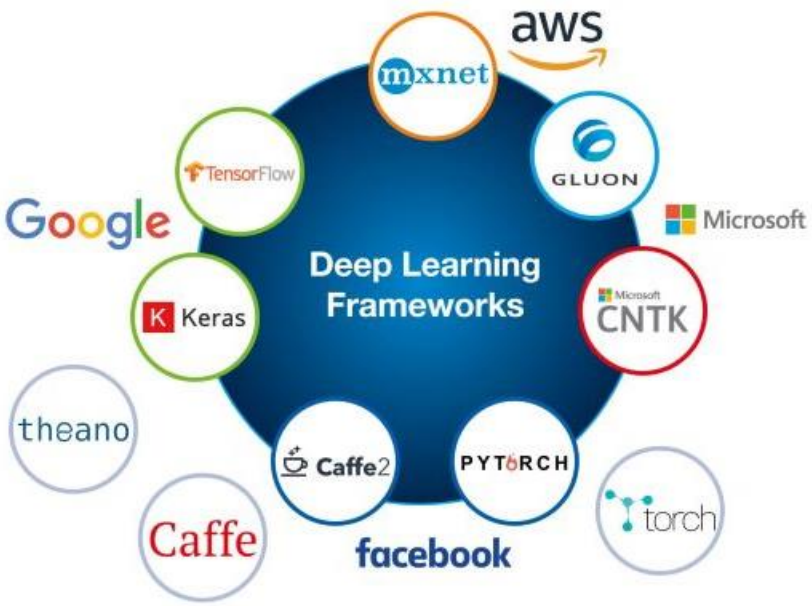
Olivier Bichler, David Briand, Vincent Lorrain, Thibaut Goetghebuer-Plachon, Johannes Thiele, Inna Kucher, Cyril Moineau, Vincent Templier

CEA LIST

olivier.bichler@cea.fr

CONTEXT / MOTIVATIONS

Behind every single major deep learning framework is a US company (GAFAM)!
Relying entirely on them today means buying their solutions/chips tomorrow because of unmatchable integration…



Dependency building
Not fulfilling our strategies for embedded AI

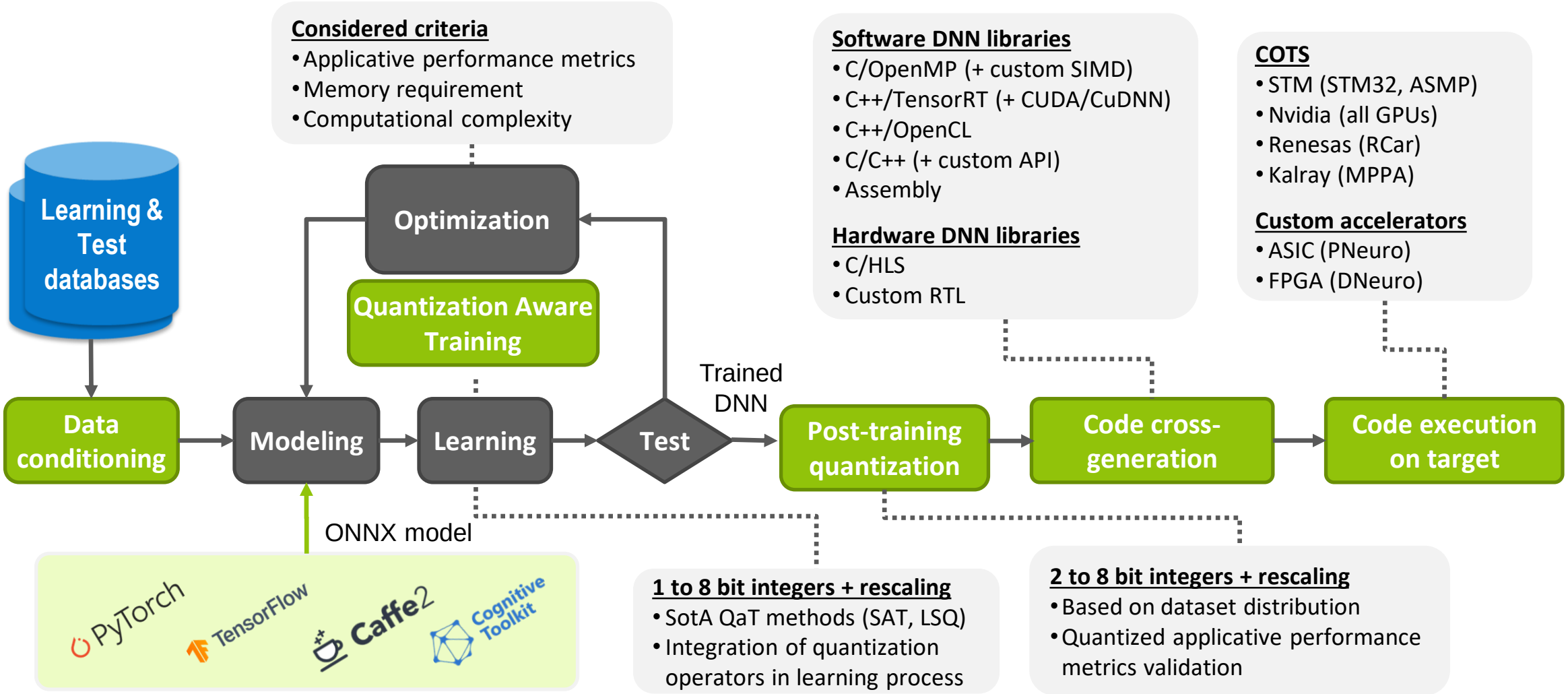
We own one of the only independent framework still able to compete: N2D2!

- ➔ **Build our own toolchain from algorithms to <u>sovereign</u> embed hardware**
- ➔ **Integrate innovative quantization/compression/pruning algorithms tailored for our hardware**
- ➔ **Targeted high level hardware generation to reduce cost and development time and remain competitive on sovereign technology node (28nm FDSOI)**
- ➔ Master high performance large-scale training implementation required to build the best performing systems
- ➔ Integrate innovative methods for reliability, robustness, dependability and explainability
- ➔ Integrate innovative methods for life-long continual learning

N2D2 FRAMEWORK NOW

GitHub:
https://github.com/CEA-LIST/N2D2/
Documentation :
https://cea-list.github.io/N2D2-docs/

- **A unique platform for the design and exploration of DNN applications**



N2D2 MODULES AND INTERFACES

Modeling & user interface:

- Native INI API
- Native Python API
- ONNX
- Keras integration
- PyTorch integration
- Partner interfaces

Core modules (C++)

- Database (public dataset drivers)
- Transformation (data pre/post processing)
- Cell (NN layers modeling)
  - Activation
  - Solver
  - Quantizer*
  - Filler
- Target (loss & score processing)
- Export (optimized code generation)
- Partner modules

Export modules

- C++/OpenMP
- TensorRT / CuDNN / CUDA
- PNeuro
- DNeuro
- MPPA
- R-Car
- HW Provider IPs

Export specialization (compute kernels level) :

- Générique MCU
- STM32
- STxP70/ASMP
- *Confidentiel*
- HW Provider LIBs

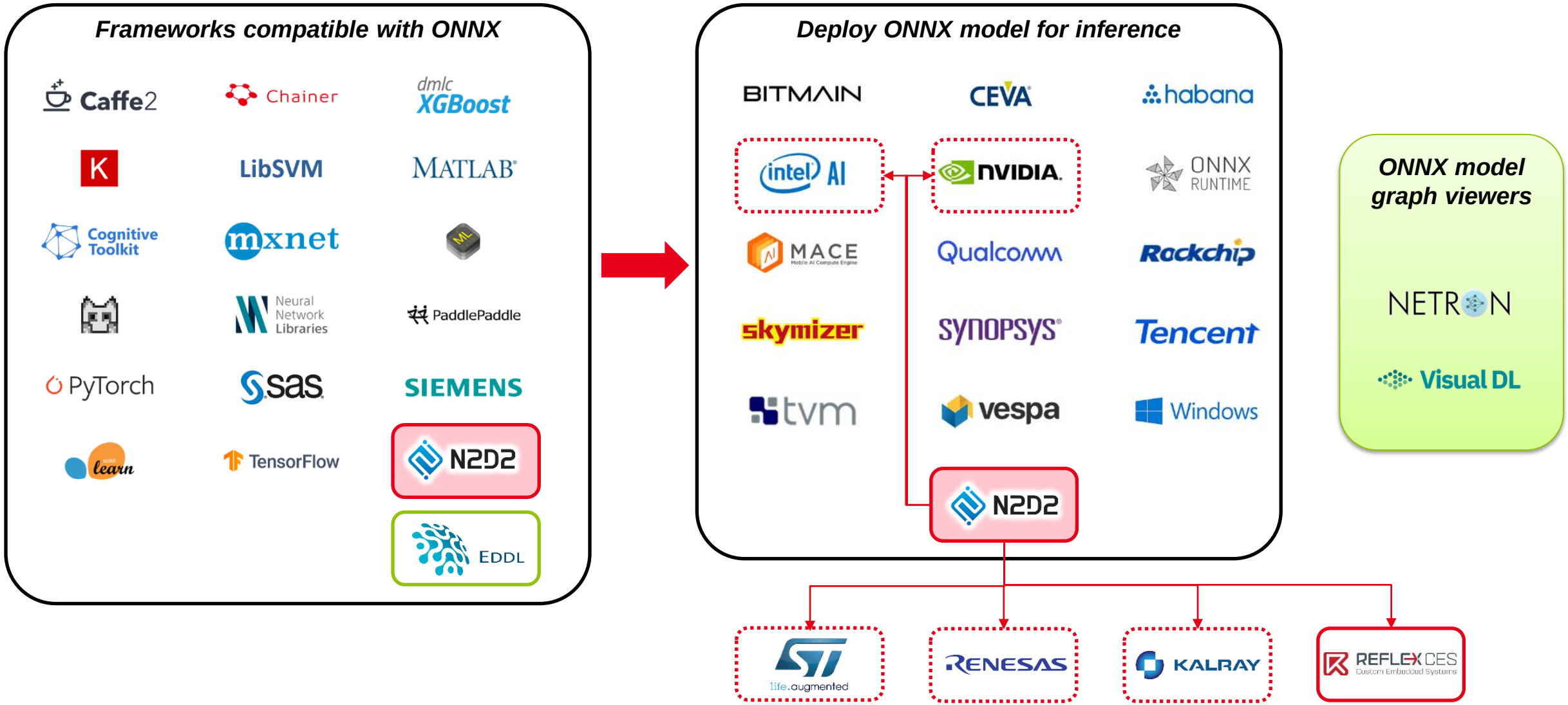
Open source

CEA IP under license

Partner IP under partner license

* *Partially open source*

N2D2 ONNX COMPATIBILITY



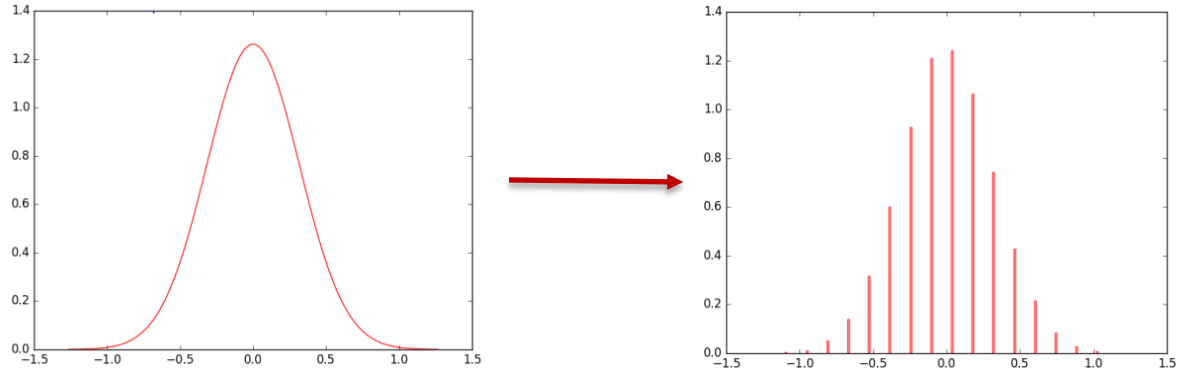
QUANTIZATION AWARE TRAINING

N2D2 implements two Quantization aware training (QAT) methods :

- **LSQ (Esser 2019) : Learned Step Size Quantization**
  - Weights and activations quantization support
  - Start from a trained full precision model
  - Quantized a DNN (8-bits) required just one epoch training time
  - Going to replace the Post Training Quantization module of N2D2

- **SAT (Jin 2019) : Scale-Adjusted Training**
  - Weights and activations quantization support
  - Outperform LSQ method in all the quantization mode
  - Long fine-tuning process (at least 150-epochs…)
  - Our quantizer reference

Weight distribution

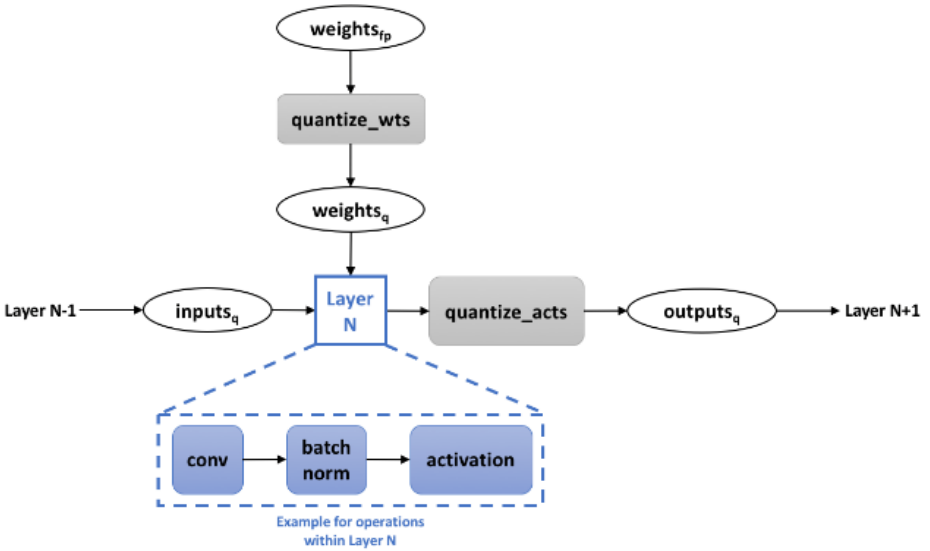


full precision

4 bits

QUANTIZATION AWARE TRAINING FLOW

- **Principle:**
  - Take into account the required precision during the training
- **How does it work ?**
  - Full-precision weights of convolution layer are quantized prior to the convolution operation
  - The output of convolution operation is passed to BN layer
  - The output of BN layer is quantized
  - The network adjusts both – quantized and full precision data (weights and activations) through the backpropagation process
  - At the end of the training the network adjusts its parameters for the particular precision



QUANTIZATION AWARE TRAINING RESULTS

MobileNet-v1 - SAT ImageNet Performances - Integer ONLY

	Quantization Range (bits)				
Top-1 Precision	Weights	Activations	Parameters	Memory	Alpha
72.60 %	8	8	4 209 088	4.2 MB	1.0
71.50 %	4	8	4 209 088	2.6 MB	1.0
65.00 %	2	8	4 209 088	1.8 MB	1.0
60.15 %	1	8	4 209 088	1.4 MB	1.0
70.90 %	4	4	4 209 088	2.6 MB	1.0
64.60 %	3	3	4 209 088	2.2 MB	1.0
57.00 %	2	2	4 209 088	1.8 MB	1.0

https://n2d2.readthedocs.io/en/latest/quant/qat.html

- **Paper results reproduced**
- **Advanced features :**
  - Modification of weights quantization to go to full integer representation – patent deposited
  - Progressive quantization of activations to go lower than 4 bits – patent deposit is ongoing !

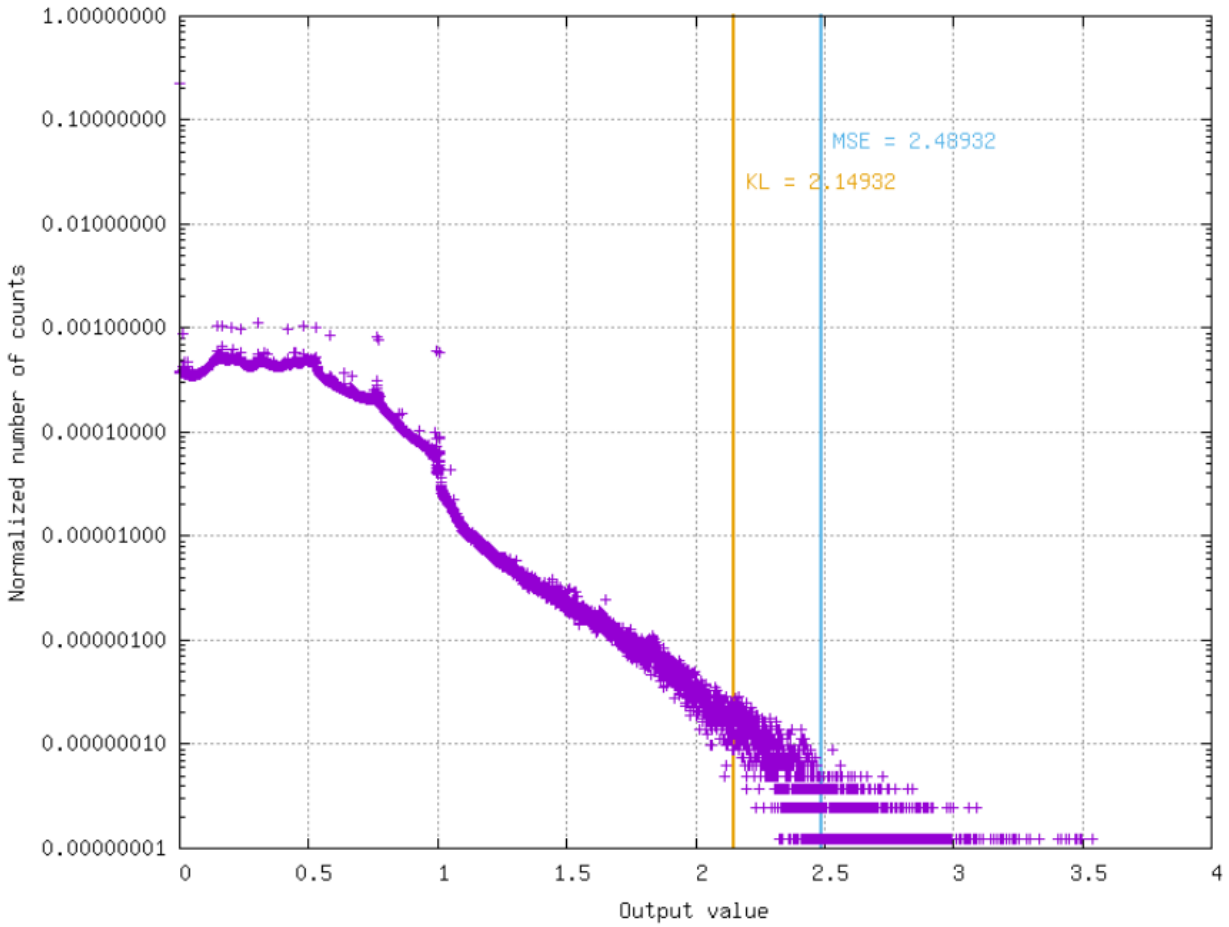
POST-TRAINING QUANTIZATION WITH N2D2

- **Post-training quantization algorithm in 3 steps**
  - Weights normalization : in the range [-1.0, 1.0]
    - Per layer normalization
    - Per layer and per output channel normalization :
      finer grain, better usage of the quantized range for some output channels
  - Activations normalization : [-1.0, 1.0] for signed outputs / [0.0, 1.0] for unsigned outputs
    - Find **optimal quantization threshold value** of the activation output of each layer using the validation dataset
    - Iterative process: need to take into account previous layers normalizing factors
  - Quantization
    - Inputs, weights, biases and activations are quantized to the desired *nbbits* precision
    - Convert ranges from [-1.0, 1.0] to $[-2^{nbbits-1} - 1, 2^{nbbits-1} - 1]$
      and [0.0, 1.0] to $[0, 2^{nbbits} - 1]$ taking into account all dependencies

POST-TRAINING QUANTIZATION WITH N2D2

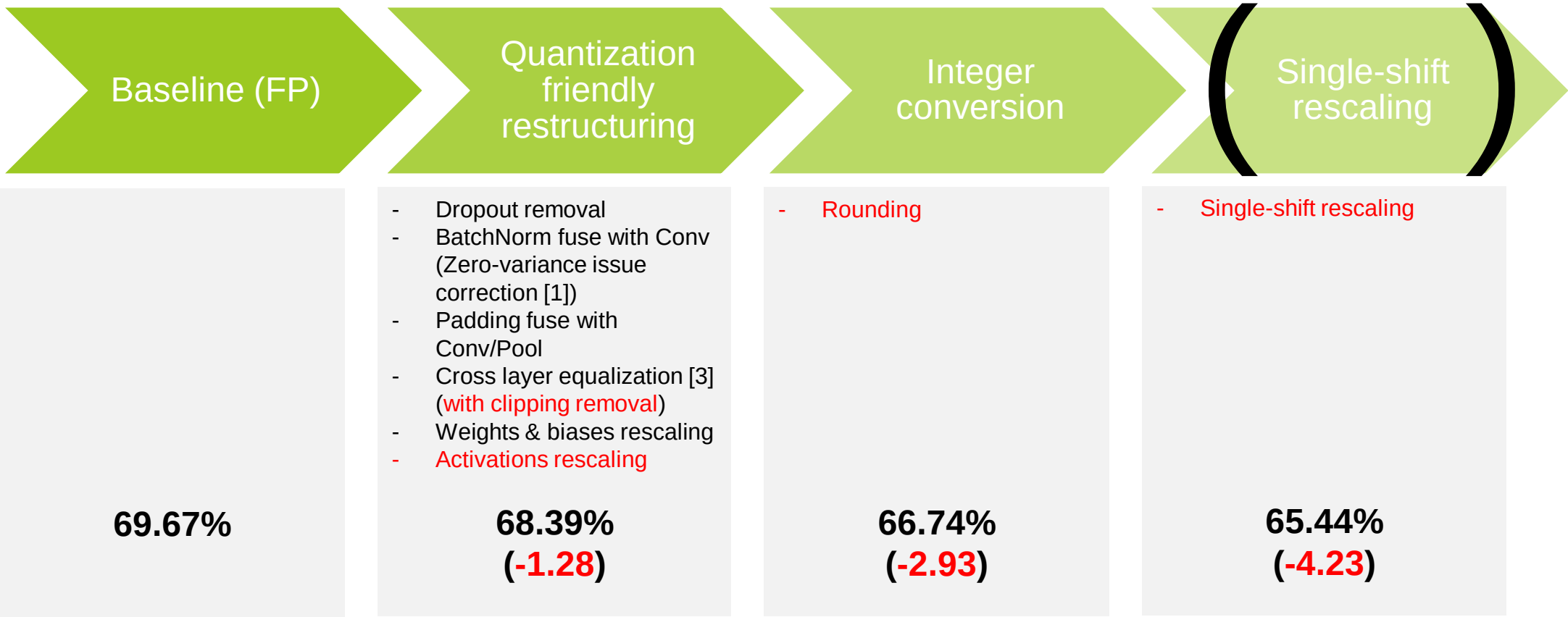
- **Find optimal quantization threshold value of the activation output of each layer**
  - Compute histogram of activation values
  - Find threshold that minimizes distance between original distribution and clipped quantized distribution using on of the two distance algorithms :
    - Mean Squared Error (MSE)
    - Kullback–Leibler divergence metric (KL-divergence)

Threshold value = activation scaling factor to be taken into account during quantization



POST-TRAINING QUANTIZATION WITH N2D2

- **Performances (post-training quantization)**
  - Accuracy loss analysis: example with MobileNet_V2 ONNX model from PyTorch



N2D2 HARDWARE EXPORTS

N2D2 ➔ TensorRT on Drive PX2

GPU (NVidia)
C++/TensorRT
(+ CUDA/CuDNN)

Support SSD and Faster-RCNN

GPU generic
C++/OpenCL

GPU

FPGA

HLS **FPGA** (Intel)
C++/OpenCL

HLS **FPGA** (Xilinx)
C/HLS

DNeuro (ceatech)
RTL

Dataflow configurable RTL library

N2D2

A unified tool for multiple hardware targets

MPPA (KALRAY)
C++/OpenCL
KaNN API

CPU x86 / ARM / DSP
C++/OpenMP
C++/OpenCL

CPU/DSP

Spike

ASIC/SoC

PNeuro (ceatech)
RTL/ASM

DSP-like programmable SIMD processor

R-Car (RENESAS)
CNN-IP C API

ASMP ( )
C++/OpenMP/CVA8

STM32 ( )
C++/DSP intrinsics

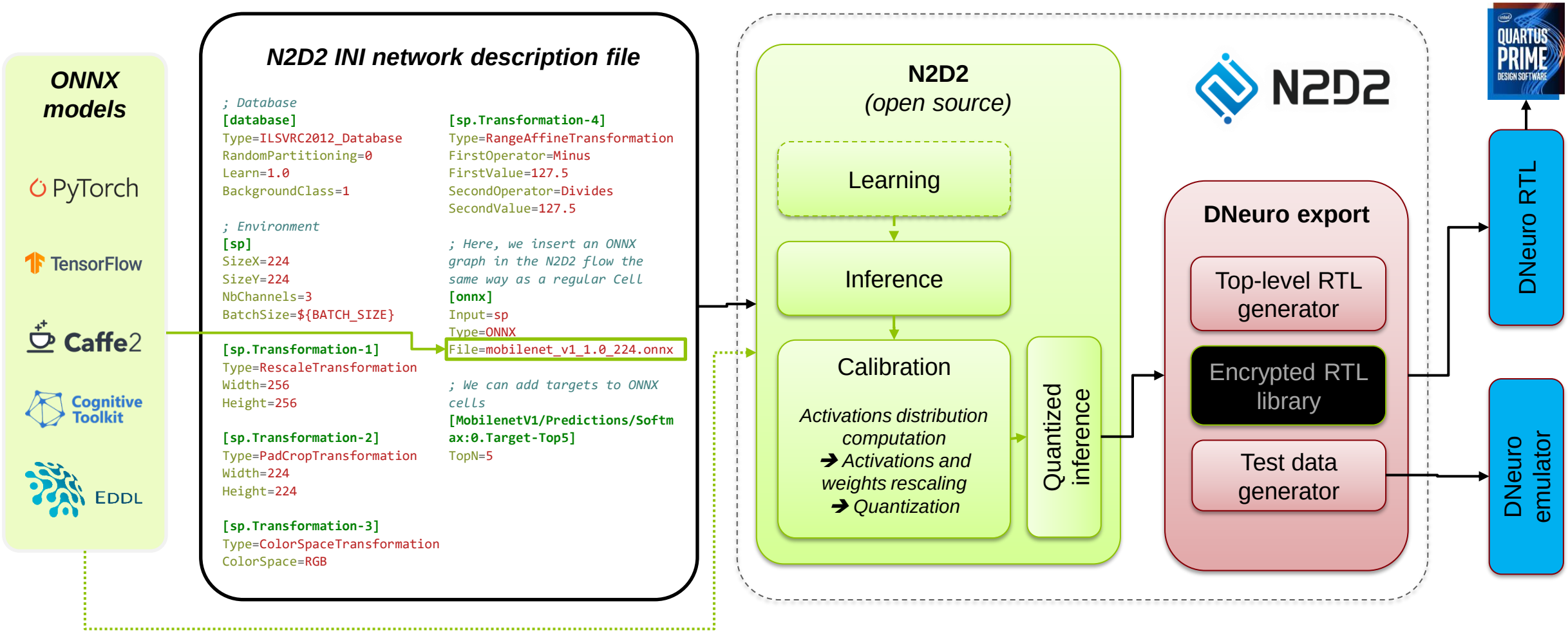
NeuroSpike (ceatech)
RTL

Generic spike
SystemC

Generic / not optimized for a specific product

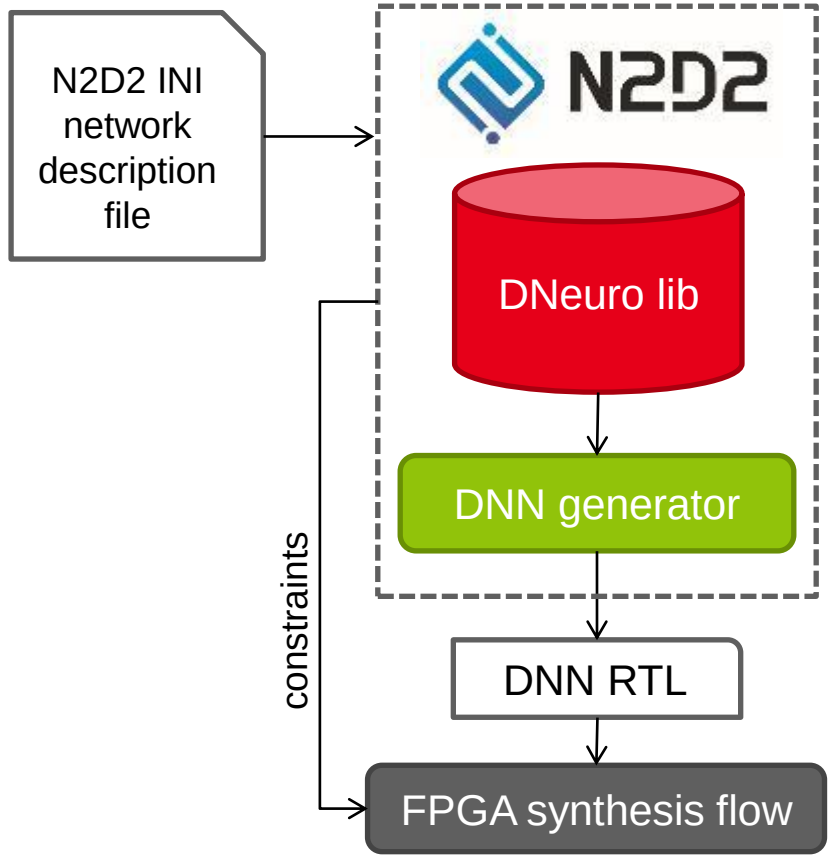
N2D2 EXPORT FLOW

- Example with the DNeuro IP



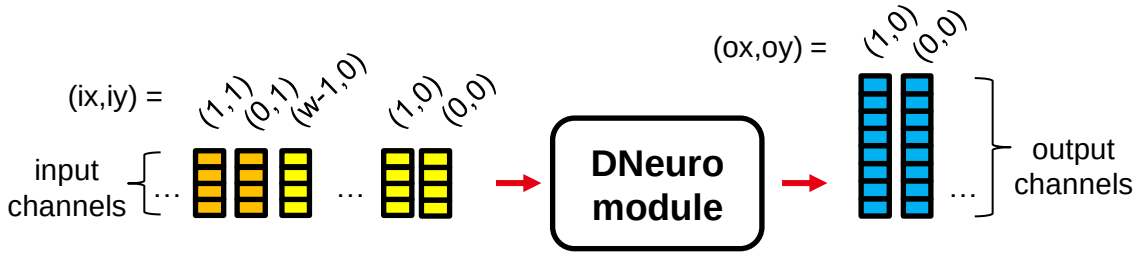
THE DNEURO IP

- **DNeuro, RTL HW library for FPGA**
  - Complete and independent RTL IP for DNN integration on FPGA
  - Dataflow computation, designed to use the DSP available on FPGA
  - Generated in a few steps from the DNN description and weights
- **Main features**
  - Data flow architecture requiring few memory (potentially **no** DDR)
  - Very high use rate of the DSP per cycle (**> 90%**)
  - Configurable precision (integers from 4 to 16 bits, typically **8 bits**)
  - Up to 4 MAC/DSP operations per cycle
- **Low complexity IP, optimized for Intel and Xilinx FPGA**
- **Support convolutional layers (Fully-CNN)**
  - Convolution and max pooling layers
  - Unit map connectivity and stride support
  - **Ongoing work: QAT support, ASIC + FPGA support for classification, segmentation and object detection (SSD) tasks**



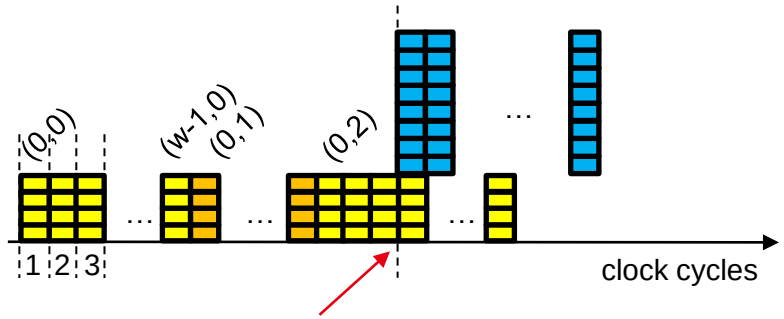
DNEURO WORKING PRINCIPLE

Dataflow modules



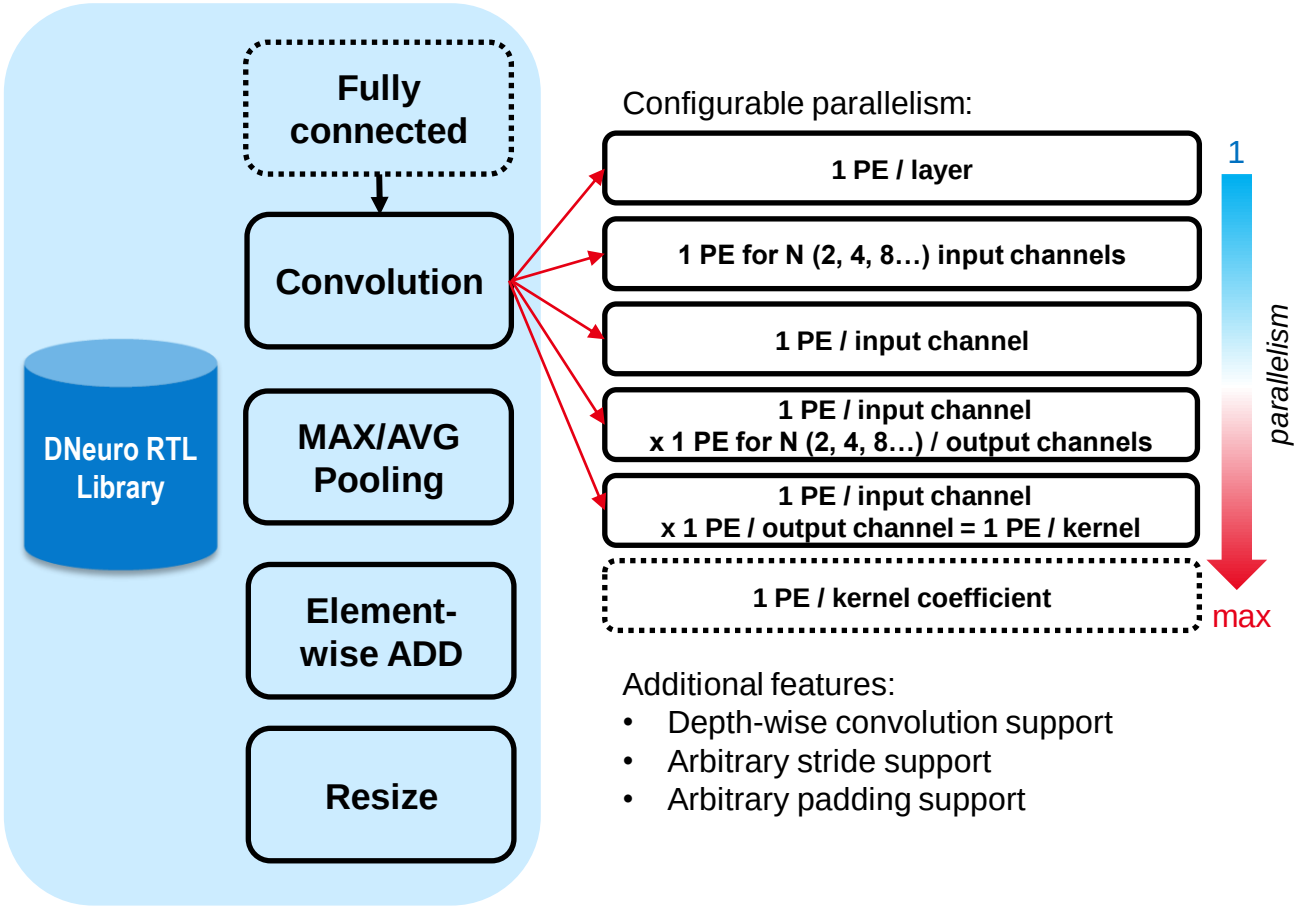
Tunable parallelized computation

- Down to K_{size} cycle / output (ox,oy)
- For 32 input x 32 output channels
  ➔ up to 1024 parallel compute units



Output data starts when all inputs data in the first neuron's receptive field is arrived
(e.g. when the 3rd pixel of the 3rd image line is arrived for a 3x3 convolution)

DNeuro RTL library modules



Additional features:

- Depth-wise convolution support
- Arbitrary stride support
- Arbitrary padding support

N2D2 HARDWARE EXPORTS

DNEURO FPGA DEMONSTRATOR

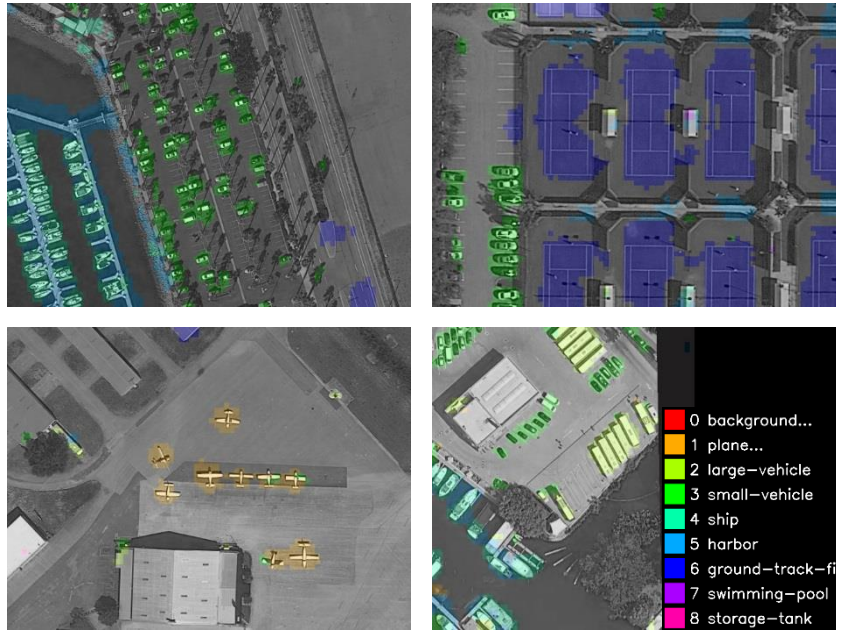
- **DOTA dataset segmentation with MobileNet-based DNN**
  - Automated DNeuro IP RTL generation from the DNN description and weights
  - Achieves ~160 FPS on Arria 10 SX270 for 640x480 images @ 200 MHz (w/o external DDR) ➔ **300 GOPS**

Arria® FPGA • SoC

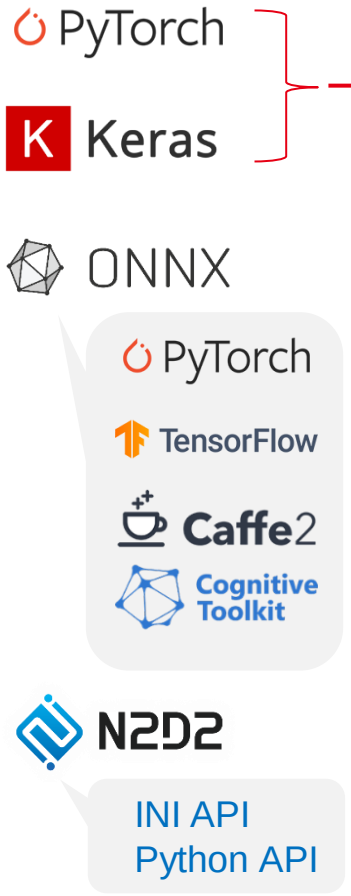
Stratix® FPGA • SoC

KINTEX® UltraSCALE

VIRTEX® UltraSCALE



N2D2 INTEROPERABILITY



Keras interoperability example

```
import numpy as np
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers

import keras_interoperability

# training parameters
batch_size = 128
epochs = 10
# Model / data parameters
num_classes = 10
input_shape = (28, 28, 1)

# the data, split between train and test sets
(x_train, y_train), (x_test, y_test) =
keras.datasets.mnist.load_data()

# Scale images to the [0, 1] range
x_train = x_train.astype("float32") / 255
x_test = x_test.astype("float32") / 255
# Make sure images have shape (28, 28, 1)
x_train = np.expand_dims(x_train, -1)
x_test = np.expand_dims(x_test, -1)

# convert class vectors to binary class matrices
y_train = keras.utils.to_categorical(y_train, num_classes)
y_test = keras.utils.to_categorical(y_test, num_classes)
```

```
tf_model = tf.keras.Sequential(
    [
        keras.Input(shape=input_shape),
        layers.Conv2D(32, kernel_size=(3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Conv2D(64, kernel_size=(3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Flatten(),
        layers.Dense(num_classes, activation="softmax"),
    ]
)

model = keras_interoperability.wrap(tf_model,
batch_size=batch_size)

model.compile(loss="categorical_crossentropy",
optimizer="SGD", metrics=["accuracy"])

model.fit(x_train, y_train, batch_size=batch_size,
epochs=epochs, validation_split=0.1)

score = model.evaluate(x_test, y_test, verbose=0)
print("Test loss:", score[0])
print("Test accuracy:", score[1])
```

N2D2 HIGHLIGHTS AND FUTURE WORK

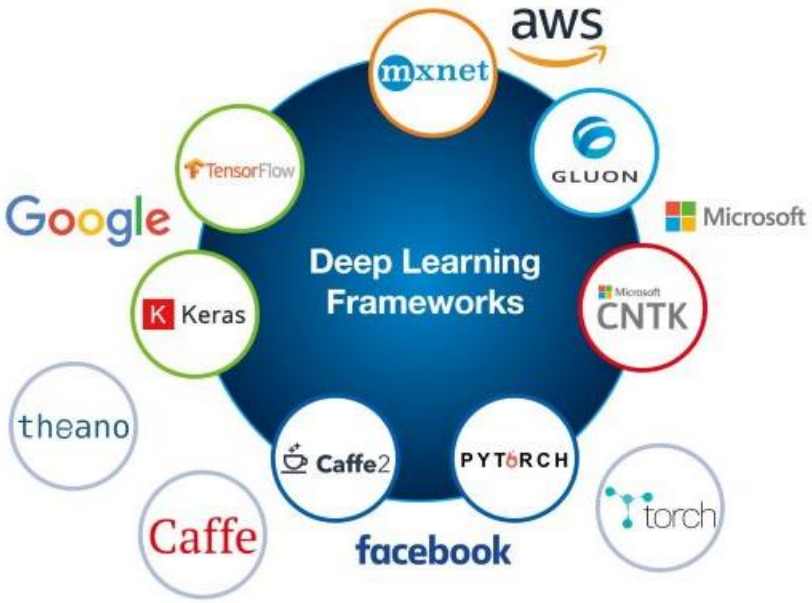
- **The platform of choice for Quantization-Aware Training (QAT)**
  - The only framework that **implement both LSQ and SAT**, the two top-performing SotA QAT methods
  - Training speed at least **x2** compared to the reference PyTorch implementation
  - Efficient and automated multi-GPU support (towards better load management than PyTorch)

- **Towards optimized lower than 8 bits C++ inference export**
  - Generic export for HW with C++/OpenMP programming model
  - Easy integration of SIMD / Intrinsic instructions
  - Compatible with HLS for Catapult (Mentor)

- **Towards greater user-friendliness**
  - **Full API is already available in Python with documentation**
  - Seamless integration with PyTorch and TensorFlow

DEEPGREEN INTRODUCTION AND GOALS

Behind every single major deep learning framework is a US company (GAFAM)!
Relying entirely on them today means buying their solutions/chips tomorrow because of unmatchable integration…



Dependency building
Not fulfilling our strategies for embedded AI

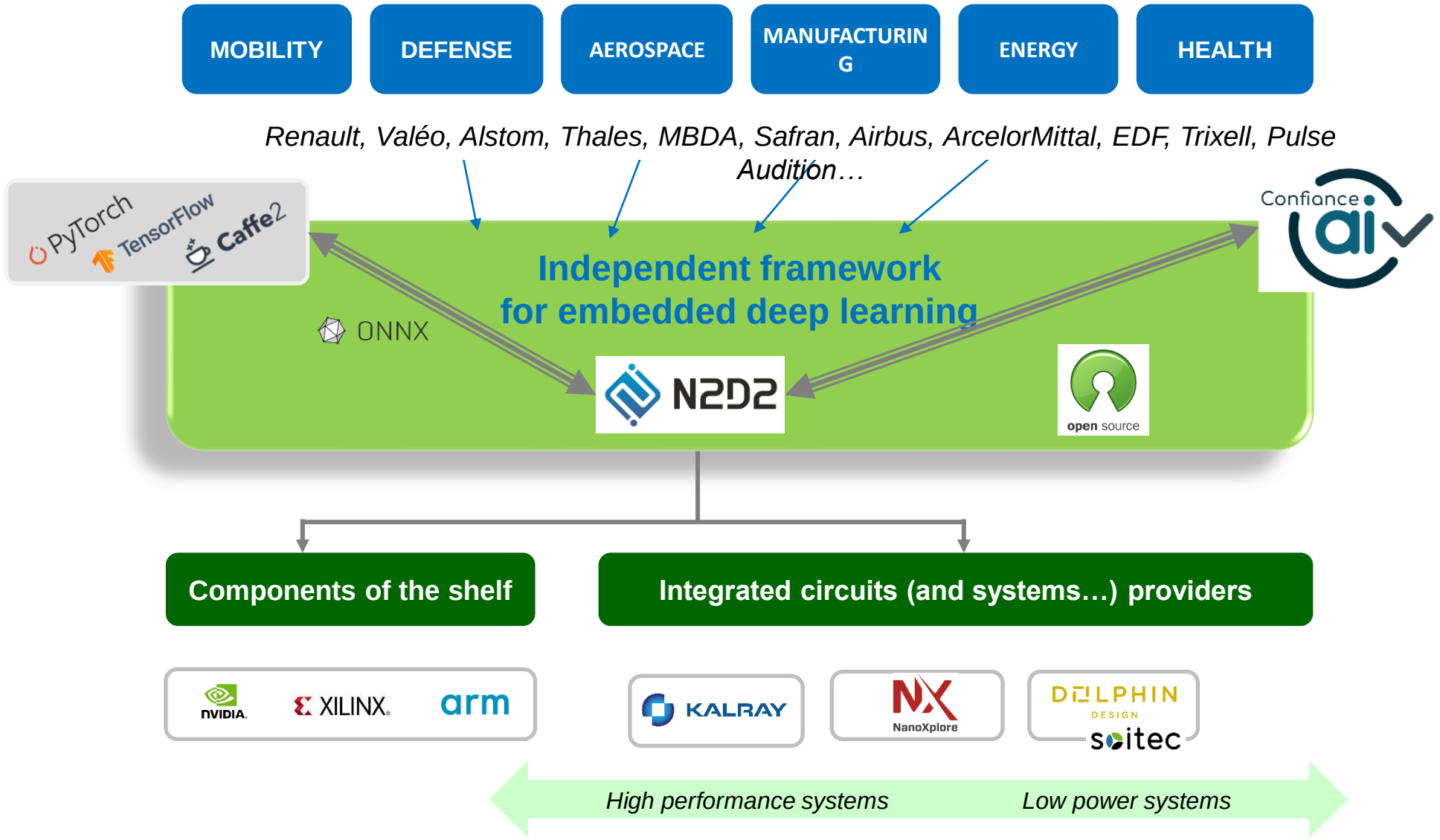
N2D2

DeepGreen

DeepGreen goals:

- To provide a software platform that specifically meets the needs of **AI embeddability**
- Meet the requirements of code **openness**, durability and sovereignty demanded by all actors
- **Facilitate the choice of hardware targets and accelerate the deployment of AI on embedded targets**
- **Put European suppliers of components & hardware IP on an equal footing with American players in terms of AI deployment tools**
- Give start-ups and industrial users (from a selection of fields where embedded AI is key such as automotive, aeronautics, aerospace, smart manufacturing ...) the tools to deploy their artificial intelligence algorithm on a large scale on various embedded targets in demanding and constrained environments;
- Allow the consortium members to contribute to the development and to orient the choice of functionalities according to their own needs.

DEEPGREEN ACTORS



DEEPGREEN GUIDELINE

Integration of innovative algorithms for embedded systems

- Frugal learning: continuous / with few examples
- Learning with compression / quantization
- Semi-supervised / unsupervised learning

Performance and interoperability

- Multi-GPU, distributed computing, multi-platform
- Interoperability with other major frameworks
- Support of hardware solutions proposed by the French and European industry

High level material design / synthesis

- Complete design flow from algorithm to hardware
- Benchmarking and performance projection
- High-level synthesis

Integration of reliability, safety and trust constraints

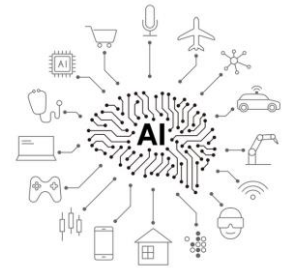
- Integration of formal guarantees
- Robustness to adversarial attacks
- Interpretability, explainability

PLATFORM FOR INNOVATION

OPEN AND INTEROPERABLE WITH PARTNERS' TOOLS OR USED BY THEM

FOR AN EMBEDDED IA WITH A LOWER ENVIRONMENTAL IMPACT

AND A HIGH DEGREE OF TRUST



DEEPGREEN PLAN

DEEPGREEN CORE DEVELOPMENTS

Developments in the core of the plateform

- LOT 1 : Robust optimization of deep graphs
- LOT 2 : High level hardware design and benchmarking
- LOT 3 : Confidence for embedded devices
- LOT 4 : Innovative algorithms dedicated to use cases with embedded devices
- LOT 5 : Performances, continuos integration and interoperability

SOVEREIGN COMPONENTS INTEGRATION

- LOT 6 : COTS MCU/GPU components integration
- LOT 7 : Generic kernels of high performances (Open CL pour multi-cibles, TVM) integration
- LOT 8 : Generic SDK base for dedicated components

DEEPGREEN WITH INDUSTRIAL PARTNERS

INTEGRATION DE COMPOSANTS SOUVERAINS

- LOT A1 : High performance components integration, KALRAY
- LOT A2 : FPGA components integration, NANOXPLORE
- LOT A3 : Integration of ultra low power components, DOLPHIN
- LOT A4 : High performance FPGA components integration, THALES

USE-CASES

- LOT A5 : Evolving on-board recognition and geolocation functions on UAVs (THALES)
- LOT A6 : Prediction and detection of anomalies on autonomous systems (TRIXELL)
- LOT A7 : Image recognition for production control and georeferencing (ARCELORMITTAL)
- LOT A8 : Image processing under reliability constraints: medical, satellite and nuclear (ARCYS)
- LOT A9 : Ultrasonic rail monitoring for predictive maintenance (ALSTOM)

Thank you

Neural Networks Design and Deployment for Constrained Embedded Systems with N2D2 Framework

Olivier Bichler (olivier.bichler@cea.fr)
David Briand
Vincent Lorrain
Thibaut Goetghebuer-Plachon
Johannes Thiele
Inna Kucher
Cyril Moineau

Centre de Saclay
Nano-Innov PC 172 - 91191 Gif sur Yvette Cedex

INSTITUT CARNOT CEA LIST

Agenda

- Introduction
- AICE WG News
- CEA List N2D2
- Next steps & Q&A

ECLIPSE FOUNDATION

Coming next…

- Planning a face to face meeting at EclipseCon community day in Ludwigsburg, Monday, October 24th
  - Save the date!

- Monthly meetings
  - Next one in September
  - Presentation TBC
  - Date to be announced on the ML