Flexible Deployment of PyTorch Models on MCU-Class Devices Using ExecuTorch

Robert Kalmar
Principal ML SW Engineer

Martin Pavella
ML SW Engineer

PyTorch Conference Europe 2026

Microcontrollers: Low power, constrained heterogeneous compute platforms

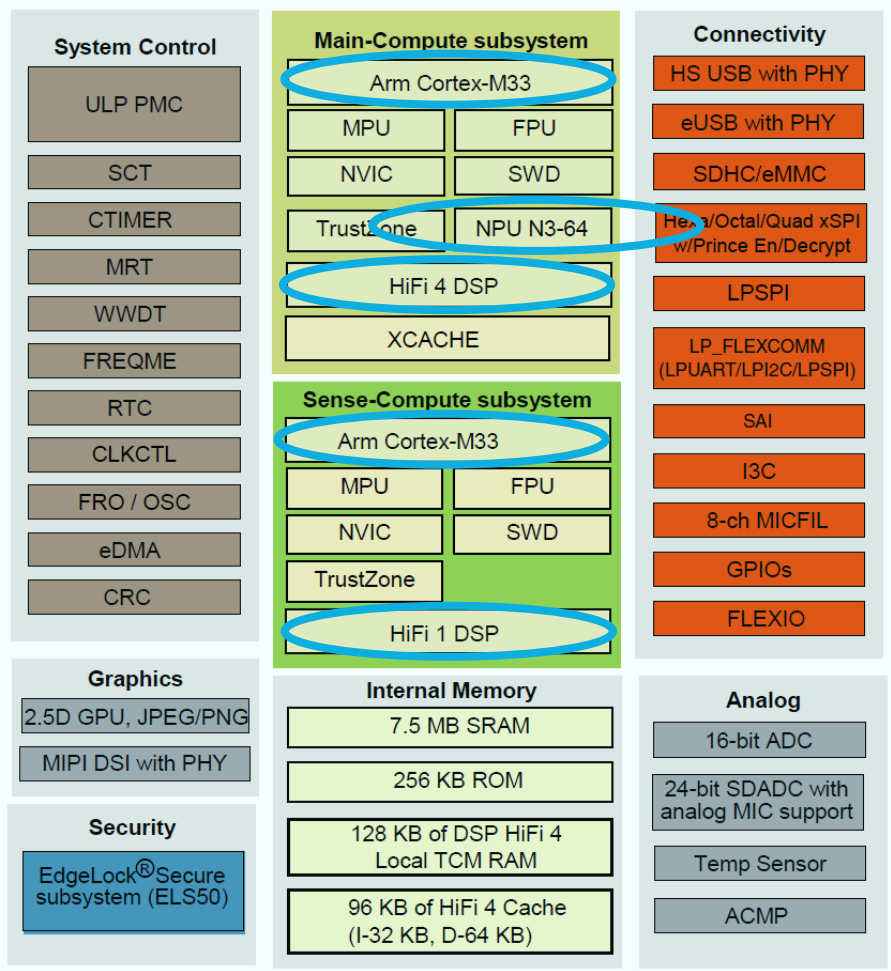
Modern MCUs combine:

- **CPU Cores** (Arm Cortex-M)
- **DSP Cores** (Cadence Tensilica HiFi, Arm Helium)
- **Neural Processing Units** (NXP eIQ Neutron, Arm Ethos-U)
- **Separate power and memory domains** (Main Compute, Sense/Always-on)

Examples

- NXP i.MX RT700
- Infineon PSOC Edge E83
- STMicroelectronics STM32H7
- Renesas RA8 Series

- Even MPUs: Real Time Domain in i.MX MPU, like i.MX 95



Source: NXP, i.MX RT700 Crossover Microcontroller Data Sheet

ExecuTorch Backends for Microcontrollers

Portable Kernels

Pure **C++** implementation of the Edge operators. Portable, decent performance, and limited quantized compute support.

Cortex-M Backend

Connector to Arm's **CMSIS-NN** library with optimized kernels for Cortex-M cores.

Cadence Backend

Connector to Cadence Neural Network Library (**nnlib**) for multiple DSP families (HiFi, Fusion G3, Vision P-Series).

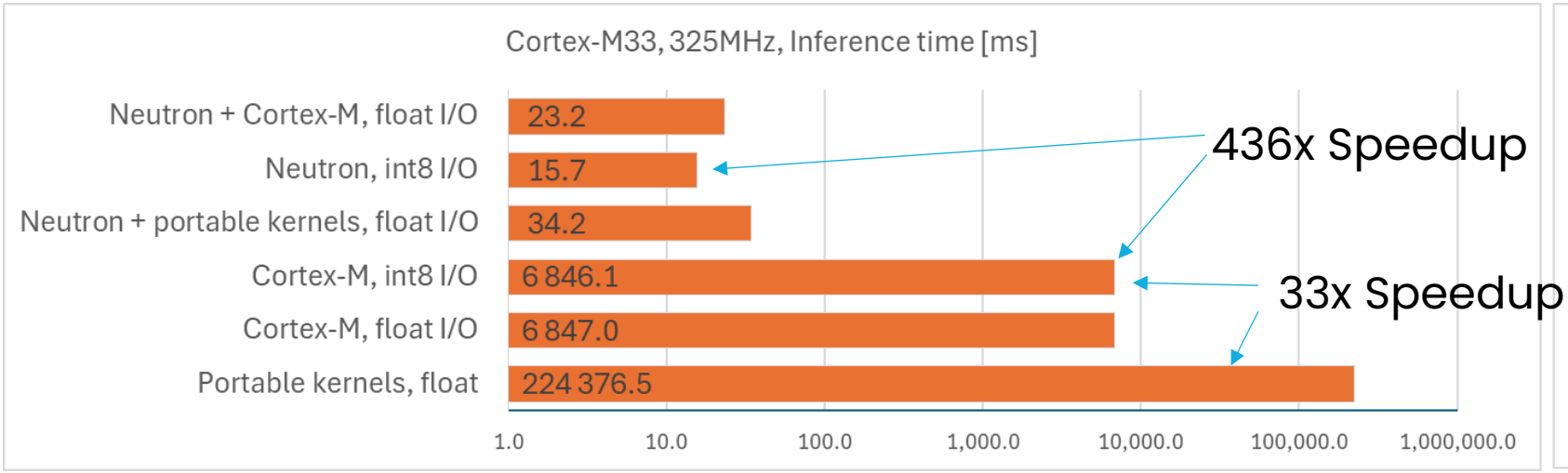
Ethos-U Backend

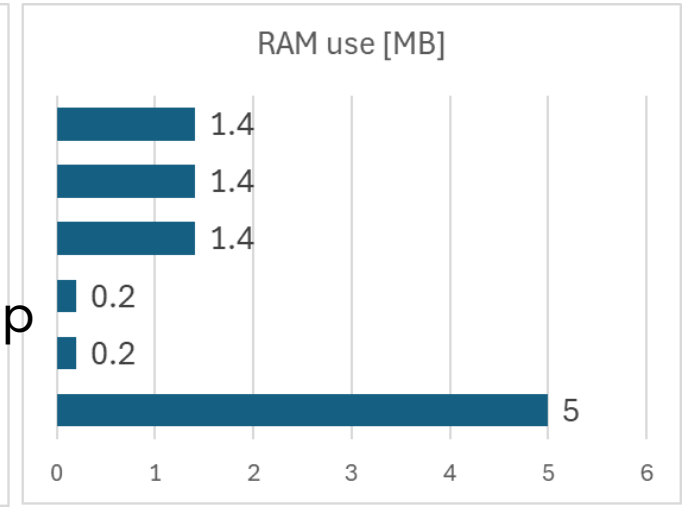
Targets **Arm Ethos-U NPU** (Ethos-U55, U65, U85).

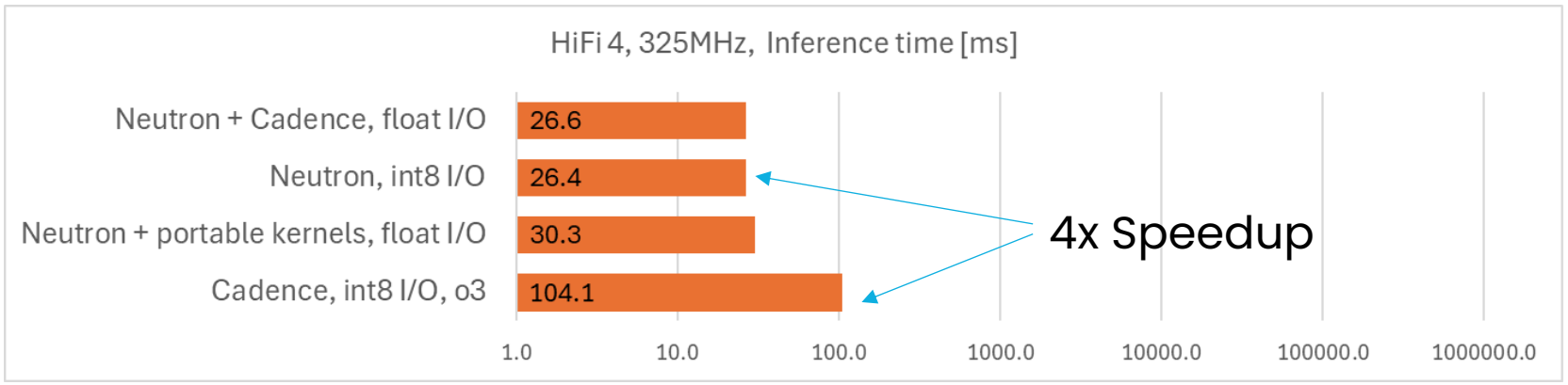
eIQ Neutron Backend

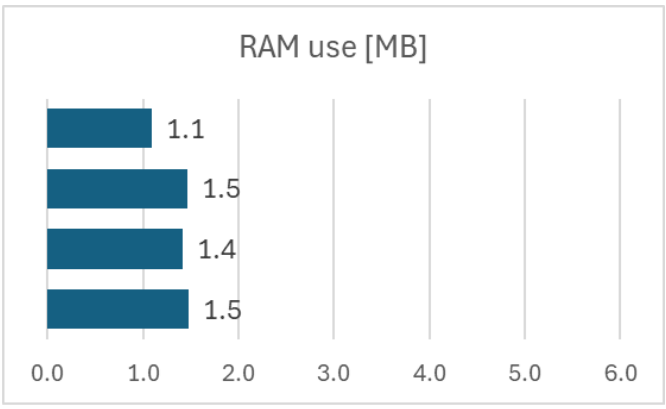
Targets NXP **eIQ Neutron NPU** family present on NXP SoC's.

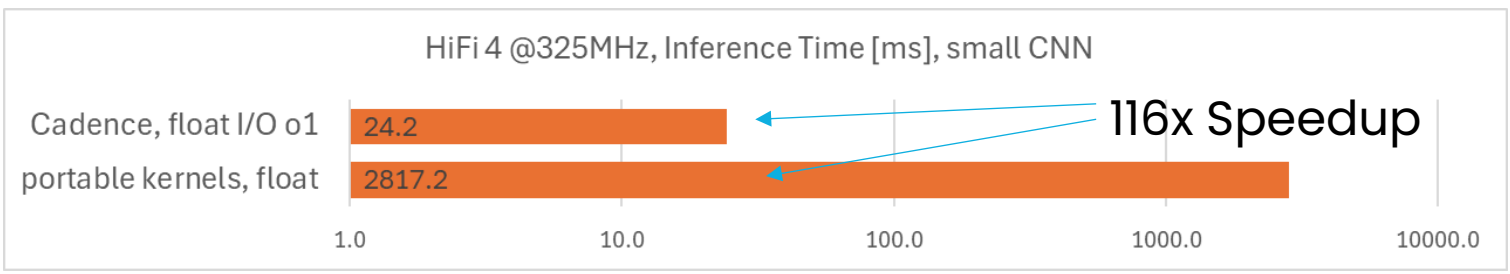
Performance Comparison: MobileNet V2 64x64 on i.MX RT700 @ 325 MHz



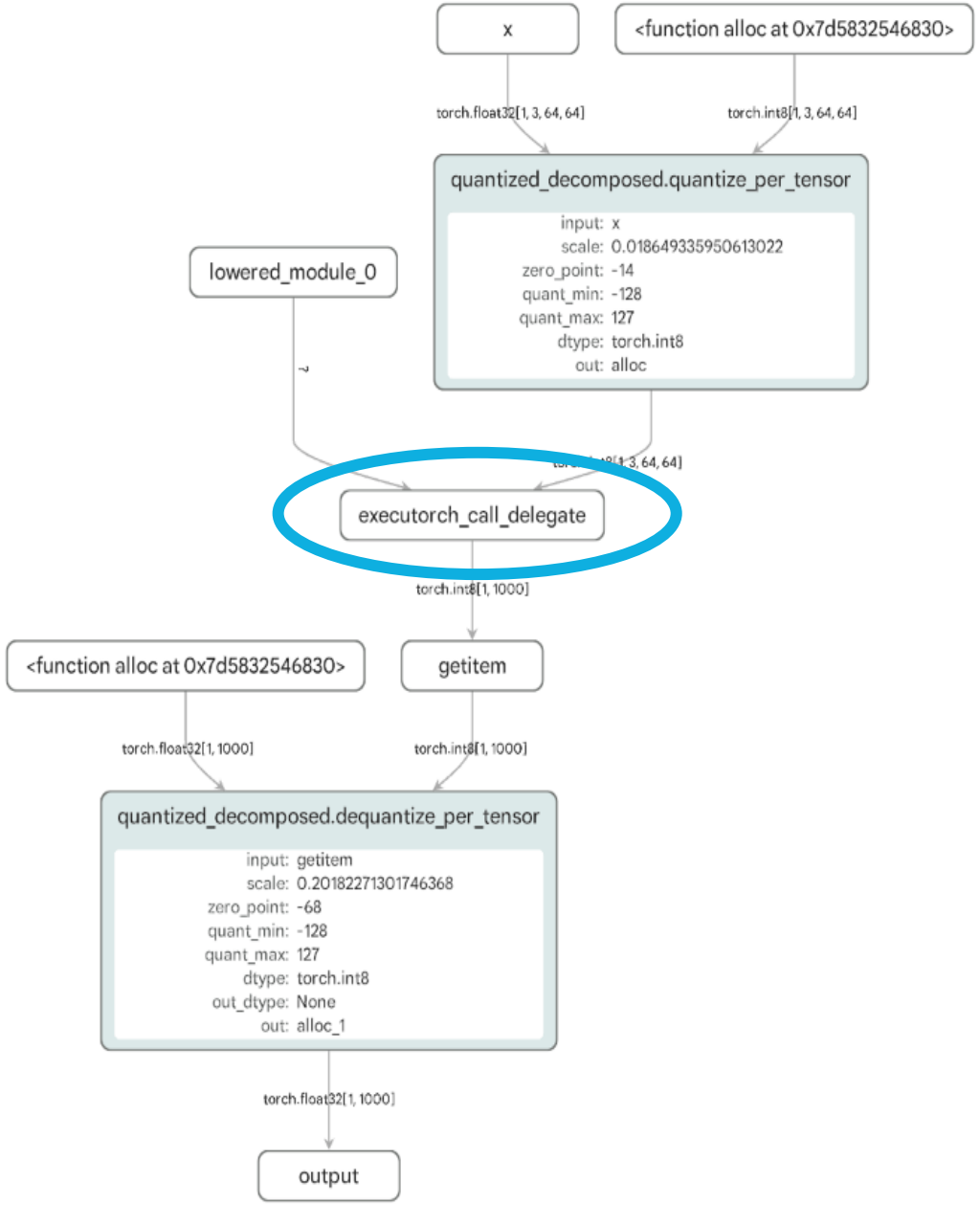




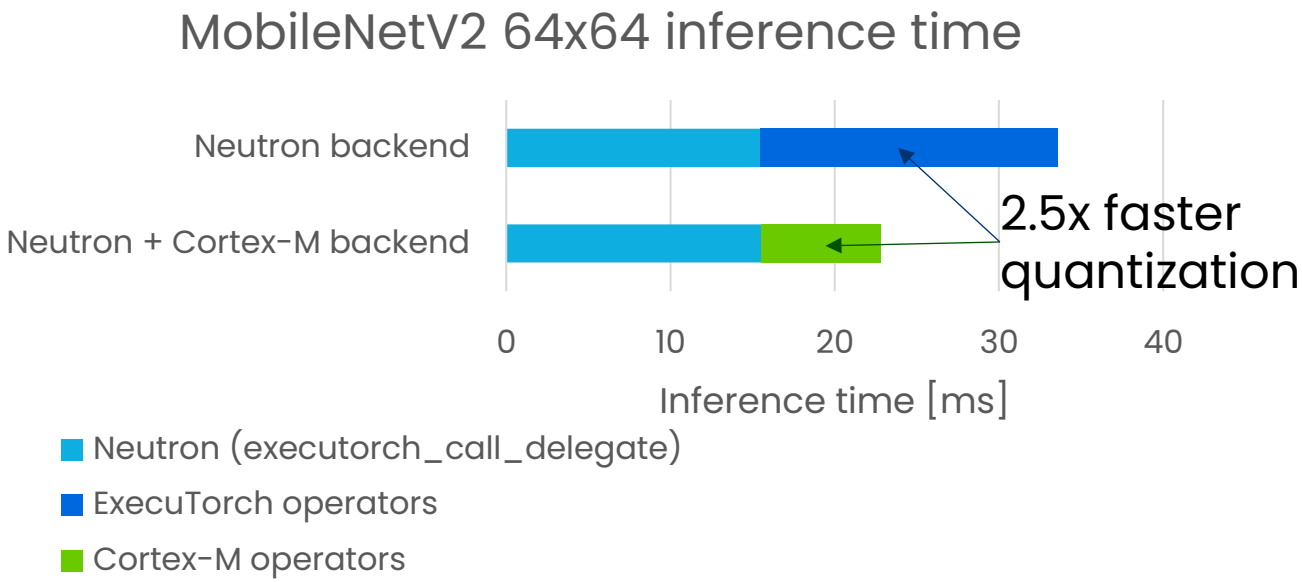




Typical ML Model Deployment



- **Most of the model** is handled by the **fastest backend** (Neutron)
- Some operators aro not supported
  - Large inference speed impact
- Why not **use another backend** for the unsupported operators?
  - Replace the [de]quantize with Cortex-M operators for **32% speedup!**
  - Replace the [de]quantize with Cadence operators for **12% speedup**!



Combining multiple backends

- Provided by ExecuTorch **recipes**
- **Platform-specific recipes** can combine different backends
  - Define pre-processing passes, quantizers, partitioners…
  - The export functions will handle the rest

- Example: **Neutron backend + Cortex-M backend**
  - **Quantize** with Neutron quantizer and with Cortex-M quantizer
  - Apply Neutron and Cortex-M **pre-processing passes**
  - Apply Neutron partitioner and **replace supported operators with Neutron delegate call**
  - Remaining operators are **replaced by Cortex-M operators**

Sounds easy right?

Combining multiple backends

Problem

- Backends sometimes require **specific quantization parameters**, and some **aten operators must be preserved** into the edge dialect
- In case of conflicting quantization requirements, extra Quantize/Dequantize nodes are inserted
  - Slower inference

Solution

- **Identify operators supported** by the preferred backend **beforehand**
- Quantize **only these operators** with the **preferred quantizer**, and the rest with the fallback quantizer
- Preserve aten operators according to the backends' requirements

Result

- **Optimized model** utilizing multiple backends for **maximum acceleration!**
- NXP has a **working implementation** combining **Neutron backend** + **Cortex-M backend**

Case Study: Power Optimization Journey for Audio Model

- Optimize an audio processing model inference <3ms with power consumption <25mW

Model Part	inference time [ms]	Neutron NPU [ms]	memory format change @ CPU
Encoder	0.379 (1.406*)	0.376 (0.326*)	0.003 (1.08*)
Recurrent part	0.903 (0.901*)	0.903 (0.900*)	0.000 (0.001*)
Decoder	0.612 (1.406*)	0.477 (0.449*)	0.135 (0.957*)
CPU Ops	2.849 (2.852*)	N/A	N/A
Total	**4.743 (6.565*)**	**1.756 (1.675*)**	**0.138 (2.038*)**

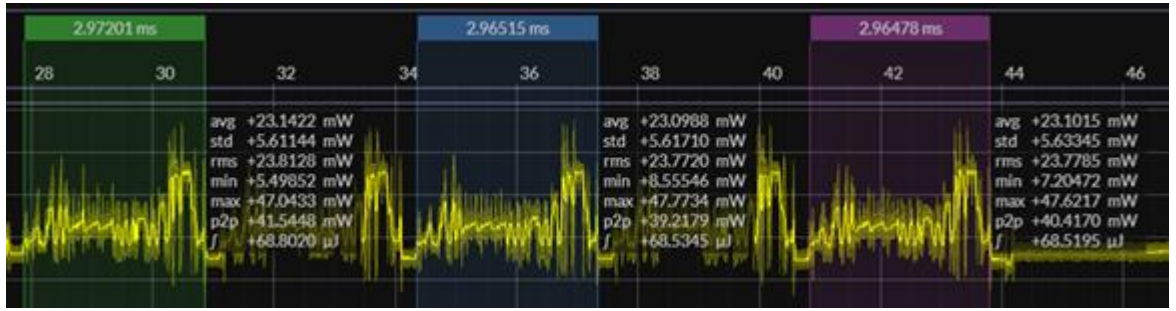
(*) Inference time when the memory format conversion is handled on CPU with ExecuTorch's transpose operator

➔ 66.8mW Power consumption

Measures:

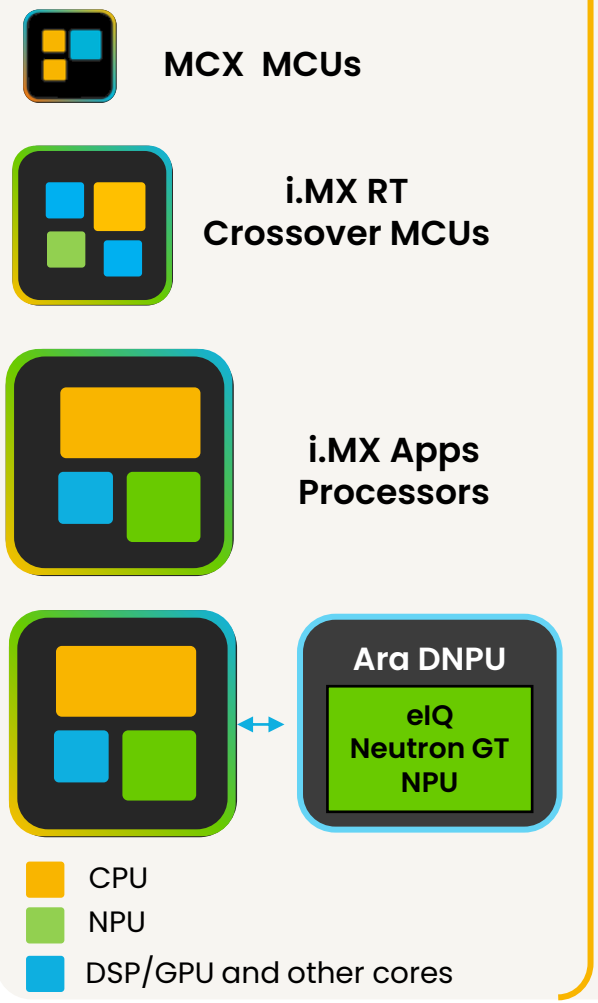
- Channel format conversion delegated to eIQ Neutron NPU
- Delegate remaining CPU ops to NPU ➔ NPU only inference
- Neutron Kernels optimizations
- Power domain optimization

Core freq. [MHz]	Inference time [ms]	Power consumption during inference [mW]	Power consumption over 3 ms with deep sleep [mW]
325	1.41	~72.34	~34.06
192	**2.39**	~28.3	**~22.6**
155	**2.97**	**~23.12**	~22.9



Intelligent Edge systems enabled today by NXP

Expansive AI-ready HW portfolio



eIQ® Neutron NPU
Highly scalable, flexible and power optimized integrated AI acceleration cores

eIQ® Neutron GT NPU
New class of integrated AI acceleration cores that scales to 100s of TOPs

Ara Discrete NPUs
High-performance AI co-processors enabling platform scalability and deployment of Agentic-AI and Physical-AI on the edge

Edge-native AI SW enablement

eIQ® Toolkit
AI/ML software toolkit for model creation, optimization and porting for TensorFlow, ONNX and Pytorch models.

eIQ® AI Hub
Cloud based portal hosting all other eIQ services. for rapid prototyping and production deployment.

eIQ® Time Series Studio
Automated ML model creation from sensor signals. No data science required.

eIQ® Model Creator
AI building highly optimized AI models (with ModelCat) No data science required.

eIQ® GenAI Flow
Context aware generative AI application development with domain-specific knowledge and data guardrails.

eIQ® Agentic AI Framework
Build autonomous edge AI systems orchestrated in the edge seamlessly with edge and cloud agents.

Get in touch

Robert Kalmar, Martin Pavella,
Jiri Ocenasek, Irina Korcakova,
Roman Janik, Simon Strycek,
Vaclav Novak

robert.kalmar@nxp.com
martin.pavella@nxp.com

nxp.com