Josh Millar *Imperial College London, London*
Yushan Huang *Imperial College London / Huawei*
Sarab Sethi, Hamed Haddadi *Imperial College London, London*
Anil Madhavapeddy *University of Cambridge, Cambridge*

Editor: Steven Y. Ko

Photo, istockphoto.com

Benchmarking Ultra-Low-Power µNPUs

Microcontrollers (MCUs) are widely used in resource-constrained environments due to their form factor and low cost and power consumption. Performing model inference on MCU-scale hardware improves privacy by running locally, lowers operating costs for model vendors, and eliminates dependence on network connectivity [1, 2]. However, deployments remain constrained by limited memory, throughput, and compute. The growing computational demands of modern models have catalyzed specialized accelerators across the computing spectrum, from data centers to embedded systems. At this resource-constrained end, MCU-scale neural processing units, or µNPUs, have emerged to provide real-time or near-real-time inference within milliwatt-scale power budgets.

This article details the first comparative benchmark of commercially available µNPU platforms, alongside fine-grained, independent results for several previously unevaluated platforms.

Our analysis uncovers expected trends and surprising gaps between hardware specifications and measured performance. Firstly, peak *giga operations per second* (GOPS), an accelerator's compute throughput, is a weak predictor of real-world performance [3]. Secondly, memory movement can dominate end-to-end latency, unlike in larger-scale hardware architectures.

INSIDE A µNPU

The µNPU design accelerates tensor operations using systolic arrays of processing elements (PEs). Each PE contains local multiply-accumulate units and often local weight memory, reducing contention and improving parallelism. The PE array connects to larger buffers and on-chip memory, which may be accelerator SRAM, tightly coupled memory, shared MCU SRAM, or external memory.

METHODOLOGY

Platforms

The benchmark covers a diverse set of commercially available platforms, alongside MCUs without dedicated neural hardware to quantify the benefit of acceleration. Our selection spans 4.8 to 512 GOPS, 128 KB to 2 MB SRAM, and arithmetic support from 1-bit quantized to floating-point execution.

- The **MAX78000** (or **MAX78K**) from **ADI** features both Cortex-M4F and RISC-V processors, along with a proprietary 30-GOPS CNN accelerator [4-6]. The latter has a dedicated 512 KB SRAM for input data, 442 KB for weights, and 2 KB for biases, and supports quantized operations at 1, 2, 4, and 8-bit precision. This fine-grained bit-width quantization is not yet widely supported on other platforms, or in libraries designed for embedded model compilation; LiteRT, for example, only supports 8-bit integer and 16-bit float weight quantization. The MAX78000 also has 512 KB flash and 128 KB of CPU-only SRAM.
- The **GAP8**, from **GreenWaves**, combines an 8-core RISC-V cluster with a 22.65-GOPS convolution engine for 8- or 16-bit acceleration. With 512 KB L2 RAM, up to 8 MB L3 SRAM, and 20 MB flash, it can store larger models or model ensembles.
- The **HX6538 WE2** (or **HX-WE2**), from **Himax**, features a Corstone-300 setup, with Cortex-M55 CPU and Ethos-U55 NPU, delivering up to 512 GOPS. The platform also features 512 KB TCM, 2 MB SRAM, and 16 MB flash.
- **NXP's MCXN947**, part of their MCX N94x line, features dual Cortex-M33 CPUs and NXP's eiQ Neutron NPU, and is built for low-power deployments, with 8-bit neural acceleration of only 4.8 GOPS, 512 KB RAM and 2 MB flash.

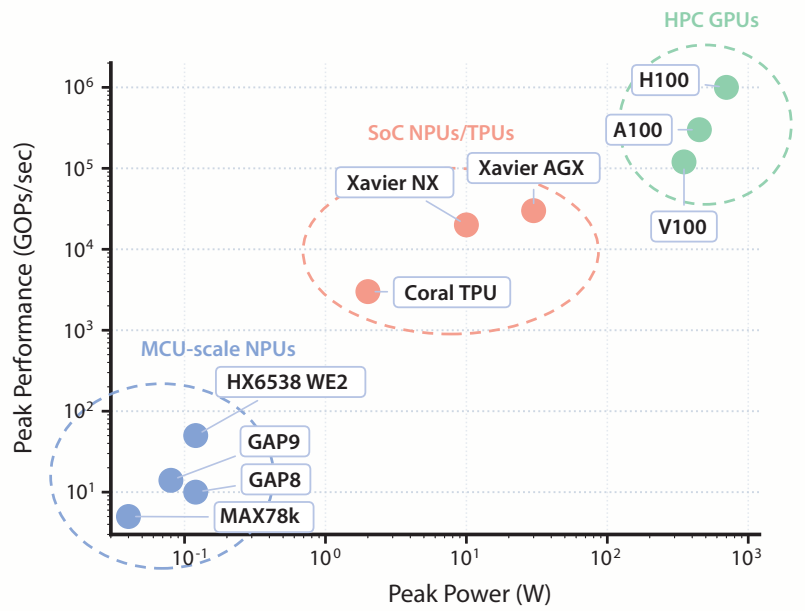

FIGURE 1. Performance of representative hardware accelerators [4]. µNPUs are uniquely suited for low-power, wearable systems.

TABLE 1. The platforms used in our benchmark, and their specifications [10-13].

MCU	CPU(s)	NPU	Flash	RAM	GOPs (max)	Bit Cap.
NPUs						
MAX78000	Cortex-M4; RISC-V	MAXIM-own	512 KB	512 KB NPU; 128 KB CPU	30	1, 2, 4, 8
HX-WE2 (Corstone-300)	Cortex-M55	Ethos-U55	16 MB	2 MB SRAM; 512 KB TCM	512	8, 16, 32
NXP-MCXN947	Cortex-M33 (2)	eIQ Neutron	2 MB	512 KB	4.8	8
GAP8	RISC-V	GAP-own	20 MB L3	512 KB L2; 8 MB L3	22.65	8, 16
Baseline MCUs						
STM32H74A3ZI	Cortex-M7	-	2 MB	1.4 MB	-	8, 16, 32
ESP32	Tensilica LX6	-	4 MB	520 KB	-	8, 16, 32

Our benchmark also includes the **STM32H7A3ZI** and **ESP32-S3** MCUs without neural hardware (although the latter has an extended instruction set for vector operations).

Models and Cross-Platform Uniformity

µNPUs are primarily optimized for convolutional models, with our benchmark suite covering a range of such models, varying convolutional depth, width, activation memory, parameter count, and post-processing requirements.

Ensuring uniformity is non-trivial: platforms expose different operator sets,

TABLE 2. The various models used in our benchmark.

Model	Input	#Params	#Layers	Conv Depth	Max Conv Ch.	Peak Act. Ram (MB, B=1)[1]	Lifetime Pressure (MB•steps)[2]	MMACs	MFLOPs
CIFAR10-NAS	3×32×32	298,762	11	10	128	0.07	18.39	74.25	36.38
ResidualNet	3×32×32	383,012	14	14	512	0.14	2.98	37.78	18.46
SimpleNet	3×32×32	383,012	14	14	512	0.14	2.45	38.00	18.46
AI8XAutoEnc	3×256	136,800	7	2	128	0.07	2.89	0.55	0.20
YoloV1	3×96×96	40,700	20	20	32	0.02	0.80	43.83	21.22

Note: MACs/FLOPs are forward-only. Peak activation RAM and Lifetime Pressure are measured with batch=1 and INT8 activations.

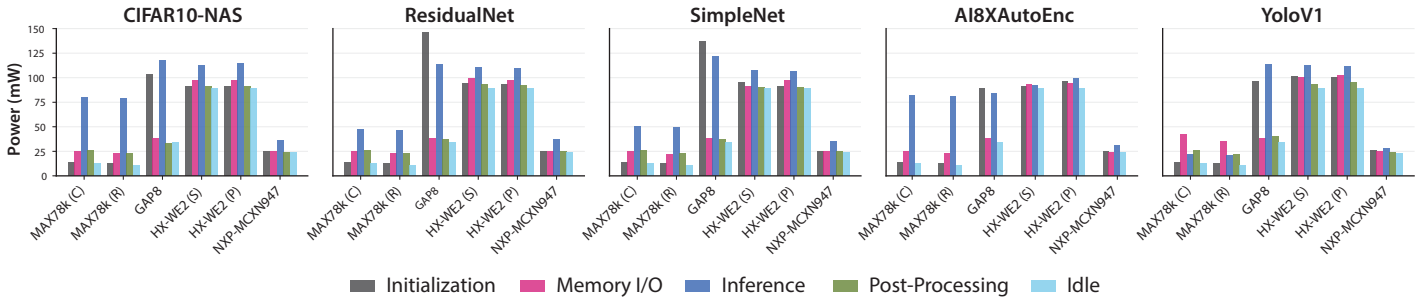

FIGURE 2. Power consumption breakdown for each stage, model, and µNPU.

memory layouts, and fallback behaviour. Certain operations are not universally supported, necessitating their implementation as CPU-side post-processing steps for relevant models. Each platform also has its own set of supported operator layouts; the MAX78000, for example, only supports 1D convolution with kernel sizes 1 to 9 and 2D with sizes of 1 by 1 or 3 by 3. Unsupported operations will fail to compile or be mapped to CPU execution and incur latency penalties.

We redesign our benchmark models around a shared, mutually compatible operator subset, and aim to avoid platform-specific structural rewrites during compilation. We use INT8 post-training quantization (PTQ), the only universally supported scheme across our platforms. The resulting models are, consequently, not necessarily optimal for each platform. Quantization-aware training and mixed-precision search are outside our scope.

Compilation and Deployment

The various benchmark platforms also support a range of model formats, from vendor-optimized variants of general model formats, e.g., LiteRT, to platform-specific formats. Our compiler workflow ingests Torch / LiteRT models along with various compiler flags (i.e., representative calibration data for PTQ, model input dimensions, platform-specific compiler options) and emits the binary / compiled model required by each platform.

Ethos-U55 models on the HX-WE2 are compiled using Vela [9]. We evaluate both its Size mode, minimizing SRAM usage, and *Performance* mode, minimizing latency using available arena cache[3].

Measurements and Metrics

We measure latency, power and energy-efficiency, in terms of number of inference operations per mJ. The impacts of various platform-specific performance optimizations are out of scope. Latency can be considered proportional to throughput, since batching and other amortizations are not practical on the given hardware.

Latency: Latency is measured using each platform's internal timer. We fix CPU/NPU frequency at 100 MHz. We run 10 consecutive inferences and report mean and standard deviation to account for run-to-run variability[4].

Power and energy-efficiency: We measure power using a high voltage power monitor at 50 kHz and 3.3 V, also averaging readings over 10 inferences. We report energy-efficiency as the number of inferences per mJ consumed.

Breaking Down Inference

We break execution into stages and measure per-stage overheads:

- **(NPU) Initialization:** setup overheads, including buffer allocation and kernel configuration.
- **I/O:** moving inputs and weights from flash or CPU SRAM into accelerator-accessible memory, and moving outputs back.
- **Inference:** executing the forward pass on the NPU.
- **Post-Processing:** CPU-side operations such as softmax and non-max suppression.

[1] Peak activation RAM: maximum live activation footprint during the forward pass, with batch=1 and INT8 activations, excluding parameter storage.

[2] Lifetime Pressure: sum over all tracked activations of size(t) in MB × lifetime_steps(t), where lifetime_steps(t) counts the number of subsequent leaf-module steps over which activation t must remain live until its last use. Larger values indicate reduced opportunities for buffer reuse.

[3] HX-WE2 (S) denotes model compilation with the Size optimization, and (P) with Performance.

[4] Runs show limited variance under bare-metal execution.

RESULTS AND DISCUSSION

Our benchmarks show that µNPU performance is governed by the complete execution path, and regularly dominated by initialization or data movement. GOPS alone does not accurately predict deployed performance.

Dedicated neural hardware provides large latency gains over MCU-only execution on convolution-heavy workloads. On CIFAR10-NAS, HX-WE2(P) runs in roughly 11.1 ms, compared with 550 ms on the STM32H7A3ZI and 555 ms on the ESP32, approximately a 50× reduction. On YoloV1, HX-WE2(P) runs in 11.4 ms, compared with 340 ms and 1166 ms on the STM32H7A3ZI and ESP32, approximately 30× and 102× reductions. These gains are not universal: for lightweight models relying more on fully connected layers than convolutions, the STM32H7A3ZI remains competitive.

The MAX78000, with its Cortex-M4 active[5], provides the best overall efficiency when NPU initialization is included, with sub-30 ms end-to-end latency across our benchmarks. HX-WE2(P) is nevertheless faster end-to-end: 1.93× faster than the MAX78000 Cortex-M4 configuration and 3.07× faster than its RISC-V configuration, but at 3.13× and 3.33× greater average power draw.

This is counterintuitive because the MAX78000 can achieve lower accelerator-only latency. At 100 MHz, the MAX78000 Cortex-M4 configuration gives an average 2.48× inference-latency improvement over HX-WE2 (P). HX-WE2 wins end-to-end because of lower I/O overheads: on the MAX78000, I/O takes 6.10× longer than inference with the Cortex-M4 active and 9.80× longer with the RISC-V. In the worst case, ResidualNet on the RISC-V spends 44.89 ms on I/O but only 2.96 ms on inference, meaning over 90% of end-to-end time is I/O, largely from moving weights into accelerator memory [7, 8].

Vela's *Performance* optimizations mostly reduce HX-WE2 latency compared with Size, but this is non-uniform across models, and energy-efficiency can degrade as model complexity grows.

The NXP-MCXN947 is the clearest example of GOPS being a weak proxy for deployed performance. Although its eIQ Neutron accelerator is rated at only 4.8 GOPS, it achieves consistent sub-30 ms end-to-end latency, comparable to the MAX78000, due to low initialization and memory-transfer overheads.

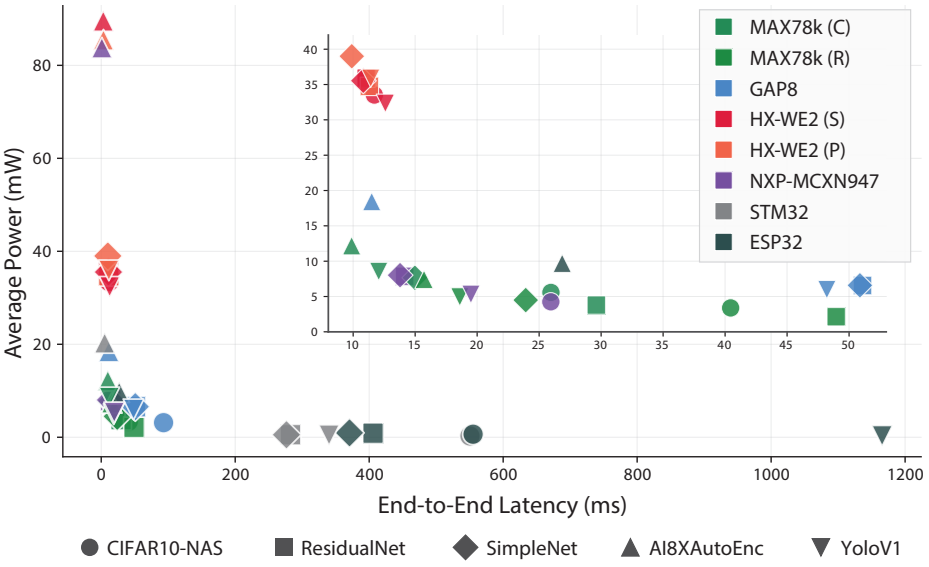

FIGURE 3. A visualization of average end-to-end latency vs. power draw for evaluated models and platforms. The inset graph provides a magnified view of platforms with lower end-to-end latency.

WHAT NEXT FOR µNPUs?

Hardware-aware model design. Neural architecture search should target platform-specifics constraints, alongside activation lifetime, buffer reuse, operator support, and data movement.

Improved neural operator support. Most platforms support limited operators, optimized for linear and convolutional layers plus pooling, element-wise activation, and addition, but excludes operators central to other model families. Unsupported layers can, in principle, be offloaded to the CPU, but this requires a tightly integrated CPU/NPU software stack that minimizes data transfer and synchronization overheads. Fine-grained CPU fallback is still limited on most evaluated platforms, and post-inference reads of intermediate activations can be latency-expensive.

Dynamic execution. Static deployment restricts early-exit networks [14], conditional computation, runtime model selection, and other input-adaptive methods for reducing latency and energy.

Standardized model formats and profiling. Unified deployment flows, reliable latency/energy profiling, and memory-aware profilers would reduce cross-platform engineering overhead and expose bottlenecks earlier [15].

PRACTICAL GUIDANCE FOR DEVELOPERS

- **MAX78000: continuous or battery-powered deployments.** The MAX78000 achieves the strongest GOPS/mW efficiency and lowest idle power. Its main limitation lies with I/O: data movement can vastly exceed NPU execution time. Use it when models can remain resident and power-gating can be exploited.
- **HX-WE2: latency-sensitive deployments.** The HX-WE2 gives the lowest end-to-end latency, at increased active and idle power than comparable platforms, making it best suited to latency-sensitive, event-driven deployments.
- **MCXN947: ultra-low-power deployments.** Despite only supporting 4.8 GOPS, the MCXN947 remains competitive because of low initialization and I/O overheads.
- **GAP8: large or multi-model deployments.** While it does not achieve competitive latency, its 8 MB L3 SRAM and 20 MB flash are valuable for hosting larger models or model ensembles.

[5] MAX78k (C) denotes use of its Cortex-M CPU, and (R) its RISC-V CPU.

Baseline MCUs: viable for convolution-light models, particularly those dominated by fully connected layers or simple 1D operations.

Limitations

We normalize CPU and NPU operating frequencies, improving comparability but not capturing each platform's peak performance. INT8 quantization, again, improves comparability but hides platform-specific advantages such as MAX78000 sub-byte quantization or HX-WE2 floating-point acceleration. Our shared NPU-compatible operator set also excludes platform-specific redesigns, QAT, operator fusion, and broader layer support.

CONCLUSIONS

Our evaluation reveals both expected trends and surprising disparities. Dedicated accelerators can deliver up to two orders of magnitude lower latency than MCUs on convolution-heavy models, yet GOPS alone does not explain real-world performance. As embedded deployments become more diverse and memory-intensive, the next gains will come not only from higher arithmetic throughput, but also from reducing data movement, broadening operator support, and improving deployment-level profiling. ■

Glossary

GOPS - Giga Operations Per Second. Peak arithmetic throughput of an accelerator.

Systolic array. Grid of processing elements, each with a multiply-accumulate unit and local memory, that passes data between neighbouring elements to parallelize tensor operations.

Activation lifetime. How long an intermediate tensor remains in memory before its final use, used to estimate memory pressure.

Quantization-aware training (QAT). Training a model while simulating quantization effects, often improving accuracy.

Operator fusion. Combining multiple model operations into one optimized execution step to reduce memory movement and overhead.

Josh Millar is a PhD candidate in the Department of Computing at Imperial College London, U.K. His research focuses on machine learning for embedded systems, with an emphasis on microcontroller-scale accelerators, low-power inference, and energy-aware system design for resource-constrained sensing platforms.

Yushan Huang is a senior researcher at 2012 Labs, Huawei, Beijing, China. He received a PhD from Imperial College London, U.K. His research focuses on low-energy on-device machine learning and the resource characterization of accelerator-equipped microcontrollers.

Sarab Sethi is an Assistant Professor in Ecosystem Sensing at Imperial College London, U.K., where he leads the Ecosystem Sensing research group. His research focuses on developing scalable, real-time sensing systems and machine-learning approaches for understanding natural environments, with a particular focus on acoustic sensing for autonomous biodiversity monitoring. His work bridges embedded systems, ecological data analysis, and practical conservation decision-making.

Hamed Haddadi is the Professor of Human-Centred Systems in the Department of Computing at Imperial College London, U.K. In his industrial role, he is the Chief Scientist at Brave Software, where he works on developing privacy-preserving analytics protocols. He is also the Co-Founder and CSO of Mulini SRL, focusing on securing consumer and industrial cyber-physical systems. His research spans IoT, applied machine learning, security, and privacy-preserving systems, including the design of AIoT platforms that can process data efficiently on-device while respecting users' privacy.

Anil Madhavapeddy is the Professor of Planetary Computing at the University of Cambridge Computer Laboratory, U.K. He co-directs the Centre for Earth Observation and the Cambridge Centre for Carbon Credits, with the aim of accelerating natural climate solutions that contribute towards ending deforestation and improving biodiversity globally. He has decades of experience constructing large-scale systems and has contributed to many open-source projects such as OCaml, Docker, Xen, and OpenBSD.

REFERENCES

[1] S.S. Saha, S.S. Sandha, and M. Srivastava. 2022. Machine learning for microcontroller-class hardware: A review. *IEEE Sensors Journal*, 22(22):21362–21390.

[2] J. Lin, W.-M. Chen, Y. Lin, C. Gan, S. Han, et al. 2020. MCUNet: Tiny deep learning on IoT devices. *NeurIPS*, 33:11711–11722.

[3] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer. 2020. How to evaluate deep neural network processors: TOPS/W (Alone) considered harmful. *IEEE Solid-State Circuits Magazine*, 12(3):28–41.

[4] A. Moss, H. Lee, L. Xun, C. Min, F. Kawsar, and A. Montanari. 2022. Ultra-low-power DNN accelerators for IoT: Resource characterization of the MAX78000. 2022. *SenSys*, 934–940.

[5] M. Clay, C. Grecos, M. Shirvaikar, and B. Richey. 2022. Benchmarking the MAX78000 artificial intelligence microcontroller for deep learning applications. *Real-Time Image Processing and Deep Learning 2022*, SPIE.

[6] Y. Huang, T. Gong, S. Jang, F. Kawsar, and C. Min. 2024. Energy characterization of tiny AI accelerator-equipped microcontrollers. 2024. *HumanSys*, 1–6.

[7] C. Jeon, T. Gong, J. Yi, F. Kawsar, and C. Min. 2025. TinyMem: Boosting multi-DNN inference on tiny AI accelerators with weight memory virtualization. 2025. *HotMobile*, 1-6.

[8] T. Gong, F. Kawsar, and C. Min. 2025. DEX: Data channel extension for efficient CNN inference on tiny AI accelerators. 2025. *NeurIPS*, 37:43925–43951.

[9] Arm Ltd. "Ethos-U Vela Compiler." Arm Developer Documentation, Version 1.3, 2024. https://developer.arm.com/documentation/109267/0103/Tool-support-for-the-Arm-Ethos-U-NPU/Ethos-U-Vela-compiler

[10] Analog Devices. "MAX78000 Artificial Intelligence Microcontroller with Ultra-Low-Power Convolutional Neural Network Accelerator." Datasheet, Rev. 5, 2024. https://www.analog.com/media/en/technical-documentation/data-sheets/max78000.pdf

[11] GreenWaves Technologies. "GAP8 IoT Application Processor." Product Brief, Version 1.9, 2020.

[12] Himax Technologies. "HX6538-A WE2 AI Processor." Preliminary Datasheet, Version 01, Jul. 2023. https://files.seeedstudio.com/wiki/grove-vision-ai-v2/HX6538_datasheet.pdf

[13] NXP Semiconductors. "FRDM-MCXN947 Board User Manual." User Manual UM12018, Rev. 2.0, Aug. 2024. https://docs.nxp.com/bundle/UM12018/page/topics/Evk_overview.html

[14] S. Laskaridis, A. Kouris, and N.D. Lane. 2021. Adaptive inference through early-exit networks: Design, challenges and directions." Éditions *Maison des Langues: EMDL*, 1-6.

[15] V.J. Reddi, D. Kanter, P. Mattson, J. Duke, T. Nguyen, R. Chukka, K. Shiring, K.-S. Tan, M. Charlebois, W. Chou, et al. 2022. MLPerf mobile inference benchmark: An industry-standard open-source machine learning benchmark for on-device AI. *Proceedings of Machine Learning and Systems*, 4:352–369.