Survey and Comparison of Milliwatts Micro controllers for Tiny Machine Learning at the Edge

Marco Giordano, Luigi Piccinelli, Michele Magno
Center for Project Based Learning - ETH Zürich
marco.giordano@pbl.ee.ethz.ch, lpiccinelli@ethz.ch, michele.magno@pbl.ee.ethz.ch

2022 IEEE 4th International Conference on Artificial Intelligence Circuits and Systems (AICAS) | 978-1-6654-0996-4/22/$31.00 ©2022 IEEE | DOI: 10.1109/AICAS54282.2022.9870017

***Abstract*—Low power Internet of Things devices are growing in number and computational capabilities, pushing to ubiquitous deployment of smart sensors that embed on board both the sensing and the processing. Thus, one of the most emerging requirements of such devices is to provide intelligence in resource-constrained processors that consume few milliwatts of power. This work focuses on surveying, comparing and evaluating seven different recent and popular microcontrollers with a power envelope from a few up to hundreds of milliwatts against a Convolutional Neural Networks workload for a non trivial task such as face recognition. The evaluation reports key points of tiny machine learning performance of the target microcontrollers in terms of inference time, power consumption, energy per inference and computational efficiency. Experimental results highlight best-in-class power consumption for Ambiq Apollo3 and Sony Spresense at $41.3\,\mu W/MHz$ and $128.2\,\mu W/MHz$ respectively. The computational efficiency primacy goes instead to the MAX78000 and then to xCORE.ai at $117\,MAC/cycle$ and $7.69\,MAC/cycle$ respectively, achieving the fastest inference at $1.4\,ms$ and $1.5\,ms$ respectively. The platforms that required the least energy per inference were the MAX78000 and GAP8, at $0.09\,mJ/inference$ and $0.52\,mJ/inference$ respectively. The benchmarked tinyML network will be released openly to allow other researchers to run future comparisons on novel low power microprocessors.**

***Index Terms*—Survey, Comparison, TinyML, CNN, Low-power, Benchmark**

I. Introduction

Machine learning has made great strides from the second half of the past century when it was theorized. It has been enabled by the large amount of data continuously generated and the growing computational capabilities, and can now be considered pervasive in everyday life, being at the core of applications such as speech recognition and computer vision [1].

Machine learning is usually associated with massive computational power requirements, as data elaboration is typically offloaded to servers, often running GPU clusters. Following the traditional scheme of information processing, data are acquired by sensor nodes, which are then sent over the network to such servers to be elaborated.

Recent developments are enabling a new class of machine learning, called Edge Artificial Intelligence (EdgeAI), which is shifting the paradigm from "cloud computing", where the computations are done in the cloud, to "edge computing" [2], where the data is elaborated directly on the sensor node, and only the useful part of the data is transmitted. Recently the new trend is toward the extreme edge with Tiny Machine Learning (TinyML), further constraining models' size to few kilobytes and enabling the models to run on microcontrollers with milliwatt-range power consumption [3].

Machine learning at the extreme edge is likely to become more and more widespread as the number of sensor nodes rapidly grows. Network resources are limited, especially when exploiting wireless connections, and the data generated by billions of devices make scaling difficult for cloud computing alone [2], [4].

TinyML brings benefits over different fronts [2], [3]. Lowering the burden on networks, which are often wireless and prone to congestion, Edge AI can save energy prolonging battery lifetime, since wireless transmission is usually the most energy-hungry activity of a sensor node. Moreover, since the data are not transmitted over the air in a shared medium, there are benefits in security and privacy. Latency can take great advantage as well, especially when feedback from the inference has to be provided to the node.

In this paper we provide an exhaustive overview of seven different promising microcontrollers architectures, highlighting their key characteristics and working principles. Moreover, a tiny neural network designed for face identification has been run on all the architectures, to benchmark their performance in terms of inference latency, Multiply Accumulate (MAC) per cycle, power and energy consumption per inference. The tinyML network will be released openly to allow further comparisons.

The paper is structured as follows:

- Section II: related work from the TinyML field is surveyed and the most prominent articles are discussed to assess a starting point for this work.
- Section III: the neural network benchmark is introduced, all the seven microcontrollers under analysis are reported and each architecture is analysed in detail pointing out differences, benefits as well as limitations.
- Section IV: the benchmark is explained and the running conditions of all the microcontrollers are stated. Results from the benchmark are reported and commented.
- Section V: conclusion are drawn and future work is planned.

II. Related Work

Machine learning on the edge has seen a growing interest both by the academic and industry research community. Efforts have been directed both in direction of novel hardware

implementation, as well as optimized software frameworks. A prime example of this is the RISC-V based Parallel Ultra Low Power (PULP) processor from ETH Zürich and the MAX78000 chip from Maxim Integrated. Similarly, several machine learning frameworks have been proposed. TensorFlow Lite for Microcontrollers is a very well-known ML framework to bring neural networks to MCUs developed by Google. It allows to convert models trained on TensorFlow and provides an interpreter written in C++ to bring inference at the edge, supporting a vast class of architectures, including Cortex-A and Cortex-M cores. FANN-on-MCU [5] is another popular machine learning framework developed by ETH Zürich and it allows an optimized porting of multilayer perceptrons trained with FANN [6], targeting Cortex-M and PULP architectures. Silicon vendors such as ST Microelectronics are also developing in-house tools such as Cube.AI to ease the porting of neural networks on their products. In particular, Cube.AI has been analysed in [7], albeit no comparison with other comparable frameworks is provided.

Several works in literature report comparisons between different cores under the same machine learning task. However, the majority of papers fail to provide a comprehensive overview of the most promising architectures, focusing most of the time on few of them. Authors in [5] provide a comparison in terms of latency and power consumption of the two architectures, Cortex-M4 and PULP, over three different applications. Authors in [8] on the other hand compare two implementations of the same Cortex-M4 cores, highlighting the energy efficiency of the microcontrollers. Authors in [9] compare two different frameworks, TensorFlow Lite for Microcontrollers and Cube.AI, albeit offering results only on a Cortex-M4 microcontroller. In [10] and [3] several possible use of TinyML systems are analysed, and a exhaustive survey of edge machine learning frameworks is reported. However, quantitative results are reported for only two of the analysed frameworks for the former, and none are reported in the latter.

Due to the rapidly developing field of TinyML, authors in [11] highlight the need for a hardware benchmark. Since no benchmark has become a de-facto standard in the TinyML community, in this paper we propose a simple, yet meaningful, computer vision task derived from [8]. The same task of face identification is performed on every architecture, with details about the implementation on each microcontroller reported in Section V.

III. Devices under analysis

In this section, an overview of the surveyed microcontrollers is reported. Key points of each architecture, as well as software toolchains used to train and deploy the model, are highlighted. Test conditions are clearly stated, as the chosen configuration if more than one were available. Moreover, the motivation for which the architecture has been chosen in this study is reported.

A. SAMD51

The SAMD51 is a general purpose Cortex-M4 microcontroller from Microchip. The tested configuration had $1\,\mathrm{MB}$ of flash memory, $256\,\mathrm{kB}$ of RAM memory and a maximum frequency of $120\,\mathrm{MHz}$. It has been chosen because it represents a class of average widespread commercial microcontrollers. TensorFlow Lite for Microcontrollershas been chosen as the framework to port the neural network model to the target hardware. The reasons are double: it is among the most popular frameworks in the field and is compatible with the optimization brought by CMSIS-NN [12] on Cortex-M4.

B. Apollo3

The Apollo 3 is an ultra low-power microcontroller from Ambiq. It exploits a proprietary Subthreshold Power Optimized Technology (SPOT) making it the most energy efficient Cortex-M4 microcontroller on the market. It is commercially available in a $1\,\mathrm{MB}$ of flash and $384\,\mathrm{kB}$ of RAM configuration, with a top frequency of $96\,\mathrm{MHz}$. TensorFlow for microcontrollers has been again chosen for the neural network porting.

C. Spresense

This microcontroller is based on the CXD5602 chip released by Sony and has the peculiarity of being a 6-core Cortex-M4 and a single-core Cortex-M0, with a maximum frequency of $156\,\mathrm{MHz}$. The Cortex-M0 duty is to control and manage the system, with the possibility of power gating the Cortex-M4 cores, thus saving power. It features $1.5\,\mathrm{MB}$ of SRAM where are stored both weights and activations of the neural network and $256\,\mathrm{kB}$ of system SRAM. The chip is compatible with the real-time operating system (RTOS) NuttX that runs on one core and it communicates in a star fashion with the other 5 Cortex-M4 cores, scheduling tasks taking advantage of Asynchronous Multiprocessing (ASMP). To port the network on the microcontroller the model has been trained with NNabla, a proprietary deep learning library belonging to Spresense SDK. This choice was made with fairness as main target, namely exploiting the maximal capabilities of the hardware.

D. Cortex-M55

This novel micro-architecture has been released by ARM but not yet implemented into any microcontroller. It is based on a ARMv8 architecture, as opposed to the ARMv7 of the previous Cortex-M4 based microcontrollers, offering one more pipeline stage - 4 instead of 3 - and a wider internal bus: 64-bit bus instead of a 32-bit one. The performance advantage over the Cortex-M4 counterpart is reportedly around 20%, with the biggest gains in machine learning tasks. The ALU of this new architecture in fact supports up to 8 8-bit MACs/Cycle using Single Instruction Multiple Data (SIMD) instructions. The network has been directly implemented with CMSIS-NN operations which support vectorized computation, namely M-Profile Vector Extension (MVE). Its computational efficiency has been tested based on the official hardware emulator released by ARM. Hence, only the inference efficiency of the architecture is reported, since no hardware implementations were available at the moment of writing.

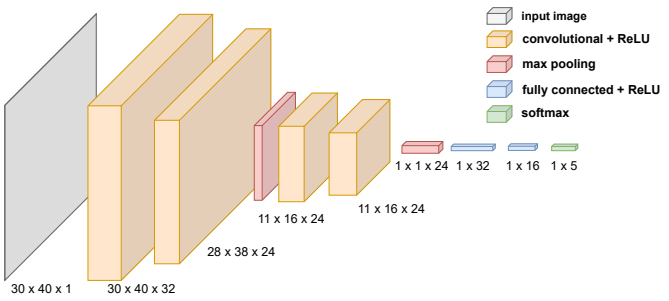

Fig. 1. Overview of the neural network.

E. PULP

The Parallel Ultra Low Power (PULP) processor is a joint effort between the University of Bologna and ETH Zürich with the goal of developing open-hardware neural network accelerators. PULP is a distributed processor based on a RISC-V instruction set architecture (ISA) and released under an open source licence. A commercial implementation by Greenwave Technologies of such architecture is the GAP8, which has been included in this study. GAP8 is a 9-core processor, of which one core is the fabric controller (FC) and controls the other eight, enables a number of peripheral, and takes care of DMA transfers. The maximum clock frequency of the FC is $250\,\mathrm{MHz}$, while the cluster cores are limited to $170\,\mathrm{MHz}$. An L1 cache is shared among the eight cores of the cluster, as well as an instruction cache. Moreover, a CNN accelerator is present in the same interconnect of the eight cluster cores. The neural network has been trained with TensorFlow, then the code for the processor has been automatically generated by the AutoTiler tool, provided by Greenwave technologies.

F. xCORE.ai

xCORE.ai is a chip made by the company XMOS, and its application field focuses on high-throughput edge computing. The architecture is peculiar: it is divided into two disentangled tiles, and each tile has a 1MB single-cycle SRAM, a novel Vector Processing Unit (VPU) and a 5-stage pipeline that feeds 8 logical cores. The operation mode is distinctive as well. Instructions from the 5-stage pipeline are issued in a round-robin fashion, and the cores have allocated $1/5^{th}$ of processing cycles if up to five cores are enabled, or $1/n^{th}$ of processing cycles if more than five cores are enabled. The maximum frequency of the processor is $700\,\mathrm{MHz}$. xCore.ai VPU has been designed to be optimized for deep tensors, with depth multiple of 32. Shallow networks do not exploit as much as possible the accelerator, in contrast to ARM's one that is optimized for depth multiple of 4. The model has been once again trained with TensorFlow, and then ported on the architecture with a set of proprietary tools provided by XMOS to best optimize inference on their hardware.

G. MAX78000

The MAX78000 is a dual-core Cortex-M4 and RISC-V based microcontroller with a 64-core convolution neural network (CNN) accelerator engine. The CNN accelerator has a maximum frequency of $50\,\mathrm{MHz}$ and it is seen as a peripheral from the two main cores. It has its own memories for neural network weights and biases and inference can be offloaded to the accelerator while the main core performs other tasks. It has been included in the study since it showed very promising performance, both in inference efficiency and energy. The neural network has been trained on PyTorch, exploiting the custom operations provided by Maxim Integrated.

IV. Neural Network

The neural network on which this study is based is referenced from [8]. The model is a feed-forward convolutional neural network (CNN) with square 3x3 filters and ReLU activation functions apart from the last layer, which has a softmax activation function. The network performs a classification task against 5 classes starting from an image of 40x30 pixels of resolution. This neural network has been chosen as a meaningful workload to test the different architectures in order to be fair to all the implementations: it had to be challenging and implementable at the same time on each platform.

The network has been quantized to 8-bit integers in all the implementations reported in this work. This choice was made to keep the comparison fair within all architectures, given that some only support quantized inference (MAX78000), and for others acceleration is available only for quantized operands (Cortex-M4).

The network accounts for a total of $39.7\,\mathrm{kB}$ in parameters and a total of $8.4 \cdot 10^6$ multiply accumulate (MAC) operations. An overview of the neural network can be found in Figure 1

V. Results

Our evaluation focused on four different metrics:

- Power efficiency
- Energy per inference
- Inference efficiency
- Inference time

The power requirements of the different devices are evaluated with the *power efficiency* metric, which is usually expressed in $\mu\mathrm{W/MHz}$. Power efficiency gives important insights about the architecture, while still being dependent on silicon technology. Figure 2 (a) exposes the rather large accelerator featured by the MAX78000, scoring the worst value of $1260\,\mu\mathrm{W/MHz}$, almost 30 times higher than the Apollo3, which on the other hand is a microcontroller designed with power efficiency in mind.

The *energy per inference*, plotted in Figure 2 (b) and expressed in $\mathrm{mJ/inf}$, is a metric strictly related to the power efficiency and the overall speed of the system. Energy is in fact defined as the temporal integral of power, and thus poses a trade-off between power consumption and inference latency. The MAX78000 is a clear example of this trade-off since, albeit being the least power efficient, is the least energy hungry.

A more transparent metric towards architecture efficiency is the *inference efficiency*, plotted in Figure 2 (c). This metric

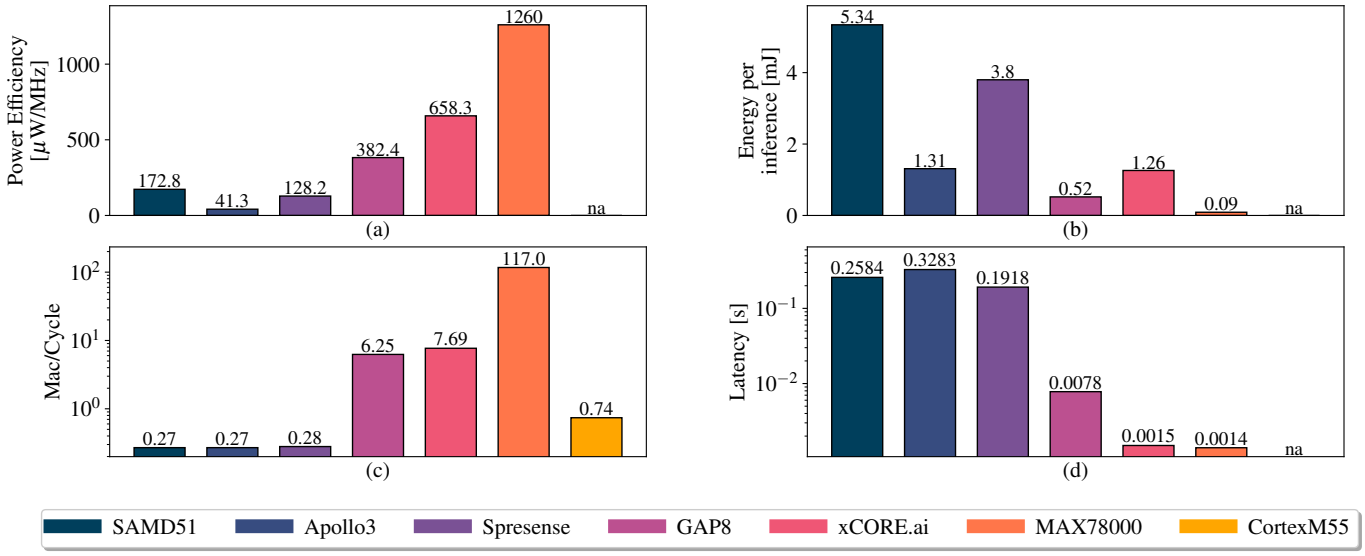

Fig. 2. Data from the selected architectures. In (a) the power efficiency is plotted. In (b) the energy required per inference is plotted. In (c) the inference efficiency is plotted. In (d) the latency of the inference is plotted.

assesses how many cycles are required to perform a MAC operation. Lower numbers mean that more data can be processed with the same amount of machine cycles, and numbers above $1\,\mathrm{MAC/cycle}$ expose the parallelism of the architecture: more data can be outputted in a single clock cycle. The clear winner in this comparison is again the MAX78000, with $117\,\mathrm{MACs/cycle}$, with the other parallel architectures as PULP and xCORE.ai runner ups. Single-core Cortex-M4 machines on the other hand take around 4 cycles to perform a MAC operation, with the exception of the Cortex-M55, which, albeit single-core, takes full advantage of SIMD instructions.

The *inference time* metric reports the time spent by the device to run one inference. As reported in Figure 2 (d), it varies greatly in our survey, ranging from $1.4\,\mathrm{ms}$ of the best architecture, the MAX78000, to almost $330\,\mathrm{ms}$ of the Apollo3, which was the slowest. All the tests have been run at the maximum frequency provided by the microcontroller, in order to be fair towards the faster implementations. It is an important metric to observe because it tells whether a device can or cannot match pure performance requirements, for instance desired minimum frame rates in vision application or hard real-time deadlines in audio processing.

VI. Conclusion

In this work, seven microcontroller architectures have been analysed and compared against a TinyML deep learning workload. Four key metrics, which highlight efficiency in its different and various facets, have been proposed and commented. The devices with higher parallelism, such as the MAX78000, PULP RISC-V based multi-cores, and xCORE.ai, showed the best performances in terms of inference latency and efficiency, with the MAX78000 achieving up to $117\,\mathrm{MAC/cycles}$, as well as in energy per inference, while falling back the single cores in power efficiency, where the Apollo3 outscored the competitors with a remarkable $41.3\,\mu\mathrm{W/MHz}$. Future work will include comparisons against different machine learning workloads, exposing possible trade-offs between performance and flexibility, which can occur if an architecture is optimized towards certain types of networks.

Acknowledgment

The authors would like to thank Xiaying Wang and Luca Benini for the support on the evaluation on the PULP platform.

References

[1] I. H. Sarker, "Machine learning: Algorithms, real-world applications and research directions," *SN Computer Science*, vol. 2, no. 3, pp. 1–21, 2021.
[2] K. Cao, Y. Liu, G. Meng, and Q. Sun, "An overview on edge computing research," *IEEE access*, vol. 8, pp. 85714–85728, 2020.
[3] D. L. Dutta and S. Bharali, "TinyML Meets IoT: A Comprehensive Survey," *Internet of Things*, vol. 16, p. 100461, Dec. 2021.
[4] V. Jeličić *et al.*, "An energy efficient multimodal wireless video sensor network with ez430–rf2500 modules," in *5th International Conference on Pervasive Computing and Applications*, pp. 161–166, 2010.
[5] X. Wang, M. Magno, L. Cavigelli, and L. Benini, "FANN-on-MCU: An Open-Source Toolkit for Energy-Efficient Neural Network Inference at the Edge of the Internet of Things," *IEEE IoT Journal*, vol. 7, no. 5, pp. 4403–4417, 2020.
[6] S. Nissen, "Implementation of a Fast Artificial Neural Network Library (fann)," p. 92.
[7] V. Falbo, , *et al.*, "Analyzing Machine Learning on Mainstream Microcontrollers," in *Applications in Electronics Pervading Industry, Environment and Society*, Lecture Notes in Electrical Engineering, (Cham), pp. 103–108, Springer International Publishing, 2020.
[8] M. Giordano, P. Mayer, and M. Magno, "A Battery-Free Long-Range Wireless Smart Camera for Face Detection," ENSsys '20, pp. 29–35, Association for Computing Machinery, Nov. 2020.
[9] A. Osman, U. Abid, L. Gemma, M. Perotto, and D. Brunelli, "TinyML Platforms Benchmarking," *arXiv:2112.01319 [cs]*, Nov. 2021.
[10] R. Sanchez-Iborra and A. F. Skarmeta, "TinyML-Enabled Frugal Smart Objects: Challenges and Opportunities," *IEEE Circuits and Systems Magazine*, vol. 20, no. 3, pp. 4–18, 2020.
[11] C. R. Banbury *et al.*, "Benchmarking TinyML Systems: Challenges and Direction," *arXiv:2003.04821 [cs]*, Jan. 2021. arXiv: 2003.04821.
[12] L. Lai, N. Suda, and V. Chandra, "CMSIS-NN: Efficient Neural Network Kernels for Arm Cortex-M CPUs," *arXiv:1801.06601*, Jan. 2018.