Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

A Review of Microcontroller and Microprocessor Architectures in IoT Devices

Shenelle Jiel T. Egloria
Department of Computer Engineering
University of Southern Mindanao
Kabacan, Philippines

Earl Lawrence S. Yanguas
Department of Computer Engineering
University of Southern Mindanao
Kabacan, Philippines

***Abstract* - The choice between a microcontroller (MCU) and a microprocessor (MPU) is one of the most important architectural decisions in Internet of Things (IoT) devices. This decision affects power efficiency, cost, scalability, and overall complexity. Even though both technologies are widely used, there are still not many organized comparisons that consider different system-level trade-offs. This article compares the computational capacity, energy consumption, deterministic behavior, memory architecture, peripheral availability, and usability of MCU and MPU systems. The results show that MCUs are better suited for ultra-low power, highly integrated embedded applications due to better performance-per-watt, while MPUs are suitable for high performance applications which requires complex operating systems, large memory footprints and intensive processing. New technologies such as TinyML, hardware accelerators and heterogeneous architectures are fast closing the previous differences. A systematic literature review methodology was used to ensure transparency and coverage. The study offers a coherent framework for comparison and highlights important directions for future research.**

Keywords: IoT, microcontroller, microprocessor, edge computing, TinyML, embedded systems

I. INTRODUCTION

One direct consequence of the rapid expansion of IoT has been the creation of an extensive set of specific requirements on performance, latency, energy usage, and pricing. Among the very first decisions for any design team is choosing the basic computing paradigm of their hardware; microcontroller (MCU) or microprocessor (MPU)-based [1], [7]. Microcontrollers have typically been used in embedded devices because of their simplicity, energy savings, small peripheral footprint, and predictable handling of interrupts. In contrast, microprocessors are becoming increasingly common in edge-computing gateways and intelligent hubs which require more raw computational ability, virtual memory support, and execution of entire operating systems like Linux.

In spite of the relevance of this choice, only a few studies compare microcontrollers and microprocessors systematically. Papers in literature tend to treat MCU/MPUs independently and do not propose multi-dimensional comparison criteria. Moreover, surveys are typically restricted to one dimension of the tradeoff, such as power [8] or security [9].

The current study bridges this gap in two ways. Firstly, it provides a comparative analysis of MCU and MPU designs based on six critical criteria for IoT systems: computational power, power consumption, real-time predictability, memory organization, peripheral support, and design difficulty. Secondly, the study highlights new developments like TinyML, hardware neural acceleration engines, and heterogeneous multi-core architectures that have been revolutionizing MCU- MPU systems. This review adopts a systematic approach to ensure that the results are transparent, reproducible, and exhaustive in addressing the pertinent literature, as explained in Section II.

This paper is organized as follows. Section II describes the methodology and systematic literature review process. Section III presents the key architectural distinctions between microcontrollers and microprocessors. Section IV provides a comparative analysis across the six selected dimensions. Section V discusses edge intelligence and the role of TinyML in IoT systems. Emerging factors influencing the MCU–MPU ecosystem are described in Section VI. Benchmark-based analysis and quantifiable results are presented in Section VII. The work is finally concluded in Section VIII, which also identifies future research directions

II. METHODOLOGY

This work efficiently adopted SLR, short for systematic literature review, in order to be transparent and for possible future editing if needed. This looks to be a very good method to review all the things known about MCU and MPU designs for IoT. I believe that this is a comprehensive coverage, but

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

some details could be missed. Nevertheless, the designs of IoT applications are discussed here in detail.

Focus Area	*Search Terms (combined with AND/OR)*
Processor types	`"microcontroller" OR "MCU" OR "microprocessor" OR "MPU" OR "ultra-low power processor"`
IoT contex	`"Internet of Things" OR "IoT" OR "edge computing" OR "wireless sensor network" OR "WSN"`
Comparison axes	`"architecture" OR "power consumption" OR "real-time" OR "memory" OR "TinyML" OR "edge AI" OR "security"`

A. Research Questions

Three primary research questions (RQs) guided the search and synthesis:

1. (Architectural Trade-offs): When implemented in IoT edge nodes, how do MCU and MPU architectures vary in terms of power efficiency, memory hierarchy, real-time determinism, and peripheral integration?
2. (Convergence through Innovation): What breakthroughs in hardware and software—such as heterogeneous multicore architectures, integrated neural processing units (NPUs), and TinyML inference—are reducing the conventional divide between MCU and MPU capabilities?
3. (Selection Frameworks): What methodical frameworks, measurements, or methods for making decisions are available in the literature to help designers decide between an MPU-based and MCU-based architecture for a particular Internet of Things application?

Criterion	*Inclusion*	*Exclusion*
Publication period	2015 – 2025 (to capture recent trends: TinyML, NPUs, hybrid architectures)	Before 2015, unless the work is foundational and cited by multiple recent papers
Language	English	Non-English
Document type	Peer-reviewed journal articles, conference papers, and authoritative industry white papers (e.g., STMicroelectronics, Infineon, DigiKey, MistyWest)	Blogs, non-technical articles, product datasheets without architectural analysis
Relevance	Direct comparison of MCU vs MPU, OR detailed architecture analysis of either processor type in an IoT context	Server/desktop CPUs, FPGA-only designs without processor discussion, purely theoretical proposals without empirical or architectural data
Data availability	Provides quantitative (e.g., mA, MHz, memory size) or qualitative (e.g., "deterministic," "unpredictable delays") data for at least two of the six comparison axes (processing, memory, power, real-time, development, security)	No architectural or performance data; opinion pieces without technical substance

B. Search Databases

The following digital libraries and databases were searched systematically. The objective was to identify peer-reviewed articles, conference proceedings and authoritative industry technical reports.

Database	*Purpose*
IEEE Xplore	Primary source for hardware, embedded systems, and real-time computing
ACM Digital Library	Software architectures, operating systems, and development frameworks
Scopus	Broad multidisciplinary coverage across engineering and computer science
arXiv	Preprints and emerging research (especially TinyML and edge AI)
Google Scholar	Supplementary searches and citation tracking for grey literature from recognized industry sources (e.g., DigiKey, Infineon, STMicroelectronics)

C. Search Terms and Query Strategy

The following search terms and Boolean combinations were applied. Queries were adapted to the syntax of each database but the main logic was the same.
Core search strings:
Example full query (IEEE Xplore):
`("microcontroller" OR "MCU") AND ("microprocessor" OR "MPU") AND ("IoT" OR "Internet of Things") AND ("power consumption" OR "real-time" OR "memory architecture").`
Example full query (arXiv):
`ti: (microcontroller OR microprocessor) AND abs: (IoT OR edge computing OR TinyML).`

D. Inclusion and Exclusion Criteria

The following standards were used during title/abstract screening to determine which papers were included or excluded.

E. Selection Process and Paper Count

There were three steps in the selecting process:

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

Stage	*Description*	*Papers Remaining*
Initial identification	Total papers retrieved from all databases after duplicate removal	247
Title/abstract screening	Papers removed if clearly off-topic (e.g., desktop CPUs, non-IoT applications)	98
Full-text review	Papers assessed against inclusion/exclusion criteria; papers lacking sufficient data or relevance removed	32
Final selection	Papers chosen for synthesis because they collectively covered all six comparison axes	16 (as cited in the reference list)

Note on final selection: The 16 references were chosen to strike a mix between current developments (e.g., TSARKA Labs 2025, DigiKey 2025) and foundational studies (e.g., Rossi et al. 2017, Campi 2017). Only when industry sources offered particular architectural information (such as memory sizes, clock rates, and NPU specs) not found in the peer-reviewed literature at the time of writing where they cited.

F. Quality Assessment of Included Sources

Each selected paper was assessed using the following four criteria:

Criterion	*Description*	*Scoring*
Technical depth	Does the paper provide detailed architectural data (transistor-level techniques, benchmark results, power measurements) rather than only high-level claims?	High / Medium / Low
Recency	For rapidly evolving fields like TinyML, NPUs, and hardware security, has the work been published within the previous five years? For fundamental architectural notions, it may have been published up to ten years ago.	High (≤5 years) / Medium (6–10 years) / Low (>10 years)
Citation impact	Has the work been cited by subsequent research? Used as a secondary indicator, not a strict filter.	Not used for exclusion; noted for context
Conflict handling	Empirical research were preferred over opinion pieces where sources disagreed (for example, on development complexity or security claims), and both perspectives were covered in the story.	N/A

Since this is a qualitative, comparative evaluation rather than a statistical or clinical study, no formal risk-of-bias methodology (such as PRISMA for clinical trials) was employed. However, by clearly noting contradicting results and limits, transparency was attained.

G. *Data Extraction and Synthesis*

- The following information was pulled into a common format from each of the 16 final papers:
- Basic information: Author(s), year, publication venue
- Processor type discussed: MCU only, MPU only, or both
- Quantitative measures: external memory capacity, on-chip memory size (flash/SRAM), clock speed, active current, and sleep current
- Qualitative findings: Real-time determinism, development complexity, security features
- Emerging trend indicators: TinyML support, NPU inclusion, hybrid design elements

Extracted data were then grouped by the six comparison axes (see Table 2 in the main text). Divergent findings were noted in the narrative synthesis. No meta-analysis was performed due to heterogeneity in reported metrics across studies.

H. *Limitations of This Review*

The following limitations are acknowledged:

1. Language bias: Only English-language papers were included, potentially excluding relevant non-English research.
2. Publication bias: Peer-reviewed venues may underreport negative results (e.g., security failures in commercial MCUs).
3. Industry sources: While valuable for current specifications, white papers may contain marketing bias; such claims were cross-verified against independent research where possible.
4. Rapid evolution:The field of TinyML and NPU integration is evolving quickly; some data may become outdated within 12–18 months.
5. No formal meta-analysis: Quantitative synthesis across studies was not possible due to inconsistent reporting of power and performance metrics.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

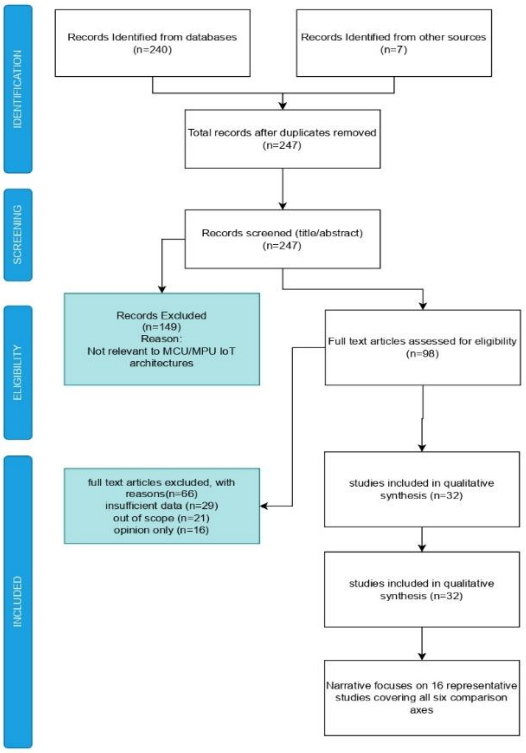

Fig. 1. *PRISMA-style flow diagram of the literature selection process.*

III. ARCHITECTURAL DISTINCTIONS

A. *Microcontroller Architecture*

An MCU integrates a processor core, on-chip flash memory, SRAM and a rich set of peripherals (GPIO, timers, ADC, communication interfaces) on a single die. Such a high level of integration results in short interconnects, low capacitance and deterministic access to internal buses, leading to low active and static power. Typical IoT-class MCUs (e.g., ARM Cortex-M4 or Cortex-M33) operate at clock speeds of tens to a few hundred MHz, and draw milliwatts in active mode [2], [8].

In practice, memory management is typically done by a simple Memory Protection Unit (MPU, not to be confused with a microprocessor), which provides basic region-based access control but does not implement the address translation necessary for virtual memory. So, most MCUs run either bare-metal firmware or a lightweight real-time operating system (RTOS) such as FreeRTOS or Zephyr. In ARM-based MCUs with no address translation and a Nested Vectored Interrupt Controller (NVIC), extremely low and highly predictable interrupt latencies are achieved, typically in the range of 12-15 CPU cycles.

B. *Microprocessor Architecture*

MPUs are meant for general-purpose, high-performance computing. They feature a central processing unit with deeper pipelines, branch prediction, multi-level caches and a Memory Management Unit (MMU) supporting virtual memory and multitasking operating systems. MPUs typically used in IoT edge gateways are single or quad-core ARM Cortex-A53/A72 devices clocked at 1–2 GHz. MPUs have much larger memory capacity than MCUs, since they use external DRAM and flash memory, often hundreds of megabytes to gigabytes, but this leads to higher power consumption and increased board complexity [1], [4].

The MMU fully supports Linux, Android or Windows IoT, providing rich software ecosystem and better process isolation. However, the combined effect of cache hierarchies, DMA and complex OS scheduling leads to non-deterministic timing behavior making it very difficult to provide hard real-time guarantees without dedicated co-processors.

C. *Convergence and Hybrid Architectures*

In recent years the features of MCUs and MPUs have been converging. High-end MCUs such as the STM32H7 series with dual Cortex-M7 and –M4 cores, external memory controllers and hardware graphics accelerators are starting to nibble away at territory previously held by low-end MPUs. The other hand, power-efficient MPUs such as NXP i.MX 8ULP or STMicroelectronics STM32MP1 series, run a Cortex-M core with the application processor, enabling real-time handling in a Linux system. Heterogeneous architectures combining a low-power MCU for sensor monitoring and a high-performance MPU for data analytics are becoming common in edge gateways [6], [15].

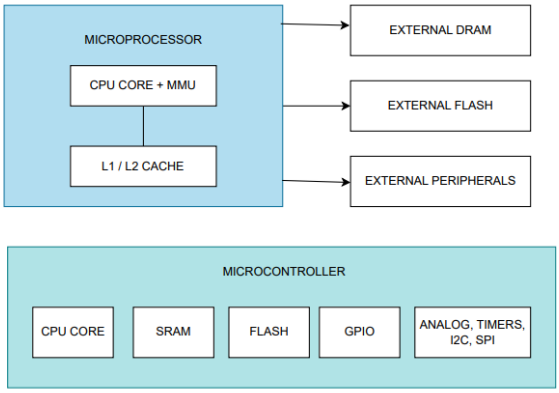

Fig. 2. *Simplified block diagram contrasting MCU and MPU architectures.*

IV. COMPARATIVE ANALYSIS

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

TABLE I. Summary of MCU vs. MPU Characteristics for IoT Applications

Dimension	*MCU*	*MPU*
Processing Capability	10–480 MHz, single-issue, often in-order. Floating-point optional.	500 MHz–2+ GHz, superscalar, out-of-order, advanced SIMD/FPU.
Energy Consumption	Active: 50–150 µA/MHz; deep sleep: <1 µA; tens of mW total active	Active: 150–500 mA total; sleep: mA range; Watts to tens of Watts.
Real-Time Determinism	Interrupt latency 12–20 CPU cycles, extremely deterministic	Latency varies significantly (caching, OS scheduling).
Memory Architecture	On-chip flash (64 kB–2 MB), SRAM (16 kB–1 MB); some support external SPI RAM	External LPDDR/DDR DRAM (256 MB–4 GB), eMMC/SD card, large caches.
Peripherals Availability	Rich integrated: GPIO, ADC, DAC, PWM, CAN, I²C, SPI, USB OTG, capacitive touch. Wireless MCUs integrate BLE/Zigbee.	Limited integrated: basic GPIO, some I²C/SPI; most peripherals require external chips
Memory Architecture	Bare-metal or RTOS; low-level HW access; frameworks (Arduino, Mbed) lower barrier.	Linux/Yocto; POSIX compatibility; complex BSP; easier multitasking.

A. A. Processing Capability

Higher clock frequencies, wider issue widths, and more sophisticated out-of-order execution provide microprocessors with a much higher peak compute throughput. For example, a quad-core Cortex-A53 at 1.4 GHz is a few thousand CoreMark while a typical Cortex-M4 at 180 MHz is about 600 CoreMark [14], [17]. The MPU's floating-point and SIMD (NEON) units enable media processing, cryptographic work, and neural-network inference well beyond what an MCU can do without a dedicated accelerator.

An MCU has plenty of cycles to spare for many IoT tasks—temperature logging, motor control, simple state machines. In fact, over-provisioning an MPU for such tasks consumes energy and makes design more complex. So workload-dependent selection.

B. B. Energy Consumption

Microcontrollers' best feature is still their power efficiency. A modern ARM Cortex-M33 MCU (like the nRF5340) uses about 50 µA/MHz when it's active and as little as 1.2 µA when it's in deep sleep with the RTC running. A normal wireless sensor node doesn't use more than 50 mW of power. This lets a coin-cell battery last for years.

An MPU-based system with external DRAM, on the other hand, uses 200–500 mA when it is running, which is 1–3 W. Even with aggressive dynamic voltage and frequency scaling (DVFS) and deep sleep modes, the quiescent current of the external memory and power management ICs keeps sleep power in the milliampere range. Most MPU devices that run on batteries need to be charged often or come with a big battery pack [2], [8].

C. Real-Time Determinism

The NVIC on ARM Cortex-M processors lets you handle interrupts in a predictable way with low latency (usually 12 CPU cycles for the Cortex-M4). MCUs can guarantee response times in the microsecond range when they work with single-cycle I/O toggling and predictable bus arbitration. This is very important for applications that need to be safe, like medical infusion pumps, industrial motor drives, and automotive braking [1].

Cache state, TLB misses, bus contention, and the scheduling latency of a multi-tasking OS all have a big effect on interrupt latency on an MPU. Real-time patches, like PREEMPT_RT for Linux, can lower the worst-case latency, but they can't match the cycle-accurate determinism of a bare-metal MCU. For systems with mixed levels of criticality, designers often use a real-time MCU co-processor with an MPU Linux core.

D. Memory Architecture

MCUs have built-in flash memory (usually 64 kB to 2 MB) and SRAM (16 kB to 1 MB). Access latencies are very low (tens of nanoseconds), and the power per bit is very low. But because of its limited capacity, firmware images and runtime data can only be so big. External memory controllers (like the STM32H7's FMC) are now built into some high-end MCUs, but this is not the norm.

MPUs use both volatile memory (LPDDR2/3/4, DDR3/4) and non-volatile storage (eMMC, SD card). The sizes range from 256 MB to several GB, which makes it possible to run complex operating systems, large machine-learning models, and keep extensive logs. The trade-off is that

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

the board is more complicated, the signals are less reliable, and the memory refresh and bus driving use more power.

E. Development Complexity

From a software point of view, MPU platforms are usually easier to develop for as they run a full operating system with standard POSIX APIs, file systems, networking stacks and package managers. Developers can write applications in Python, Node.js or C++ and take advantage of a broad ecosystem of libraries. A/B root filesystem strategies allow for simpler over-the-air (OTA) updates.

Traditionally MCU development has been a low level C and direct register manipulation affair. But modern frameworks like Arduino, Mbed OS and MicroPython have dropped the barrier to entry considerably. The RTOS ecosystem (Zephyr, FreeRTOS) provides a rich middleware, and vendors provide graphical configuration tools (STM32CubeMX, MCUXpresso). Still, developers have to manually handle memory allocation and write device drivers for custom sensors, which increases the time-to-market.

F. Peripherals Availability

A major advantage of MCUs is the great variety and number of on-chip peripherals. The common IoT-focused MCU includes multi-channel 12-bit ADCs, DACs, analog comparators, op-amps, capacitive touch sensing, motor-control timers, and numerous digital interfaces (UART, SPI, I2C, I2S, CAN-FD, USB). Certain wireless MCU families (for example, Silicon Labs EFR32, Nordic nRF52/53) embed Bluetooth Low Energy, Zigbee, or Thread radios on the same die, reducing BOM and power.

MPUs typically have a small set of peripherals, a few UARTs, I²C, SPI, while analog blocks are almost non-existent. External chips (ADC, DAC, GPIO expander, motor driver) are needed to read a temperature sensor or drive an actuator, adding cost, board area, and design effort. Often that is enough to tip the scales in favor of MCU for sensor-heavy, space-constrained IoT devices.

V. EDGE INTELLIGENCE AND TINYML

The emergence of ultra-low-power machine learning inference (TinyML) is fundamentally altering the role of MCUs in IoT systems. Now, frameworks like TensorFlow Lite for Microcontrollers, Edge Impulse and microTVM enable keyword spotting, anomaly detection and simple image classification on devices with as little as 64 kB of RAM [12]. TinyML addresses latency, bandwidth and privacy issues with cloud-based inference by processing data locally.

TinyML models based on MCU use model compression techniques such as quantization (int8), pruning, and neuro-architecture search to fit into tight memory budgets. For example, a keyword spotting model with 90% accuracy can run in < 20 kB of SRAM on a Cortex-M4. Integrated neural network accelerators on MCU dies (e.g., ARM Ethos-U55, Syntiant NDP, GreenWaves GAP9) push the performance envelope to hundreds of GOPS at sub-milliwatt power levels [16].

On the MPU side, edge gateways with GPUs or external NPUs (Intel Movidius, Google Coral) can run full-size image classification (MobileNet, ResNet) and object detection (YOLO) at frame rates suitable for video analytics. The cost is higher power consumption (several watts) so these solutions are more suitable for mains powered infrastructure nodes than for battery powered sensors. The trend is a hierarchical intelligence stack, with MCU nodes doing initial filtering and wake-word detection, and MPU gateways doing complex, multi-stream processing and decision fusion.

VI. EMERGING TRENDS

Several technology trends are reshaping the MCU-MPU landscape:

Heterogeneous System on Chip (SoC) Integration: SoCs that combine one or more Cortex-M real-time cores with Cortex-A application processors (such the STM32MP1 and i.MX 8M Plus) enable a single chip to execute both high-level application functions and real-time control. Workloads can be dynamically divided to achieve the optimal power-performance ratio [6], [15].

RISC V Open ISA: The open RISC-V instruction set is democratizing processor design by enabling custom extensions for the Internet of Things. RISC-V cores are beginning to show up in both MCU-class processors (like the SiFive E2 series) and Linux-capable MPU-class chips (like the StarFive JH7110), offering a shared ISA basis that may further blur architectural barriers.

Energy Harvesting and Intermittent Computing: Battery-free devices that harvest energy from light, vibration, or RF require MCUs that can checkpoint processor state into non-volatile memory and restart after power loss. Because of their simple architecture and minimal beginning energy, MCUs are especially well suited to this field; MPUs, on the other hand, are currently unsuited for intermittent operation due to their complex boot procedures and DRAM refresh requirements.

Hardware Enforced Security: Both architectures are implementing hardware-based trust. ARM TrustZone provides verified firmware upgrades, secure boot, and distinct execution environments for Cortex-M (TrustZone-M) and Cortex-A (TrustZone-A) cores. Physical unclonable functions (PUFs) and cryptographic accelerators are used to

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

protect device identity and keys. As IoT security risks grow, hardware-level security measures are becoming essential rather than optional [9], [18].

VII. QUANTITATIVE RESULTS AND DISCUSSION

Table II gives representative performance and power numbers extracted from the benchmarks and datasheets considered in this work [14], [16], [17] to quantify the qualitative comparison.

TABLE II. Quantitative Benchmark Data for Representative IoT Processors

Metric	*ARM Cortex-M4 (STM32F407, 168 MHz)*	*ARM Cortex-A53 (RPi Zero 2W, 1 GHz)*	*ARM Cortex-M55 + Ethos-U55 (Alif E3, 160 MHz)*
CoreMark	608	2200 (est.)	850 (CPU) + NPU TOPS
Active power (core + mem)	50 mW	900 mW	65 mW (CPU) / 20 mW (NPU inf.)
Performance/Watt (CoreMark/mW)	12.2	2.4	—
Keyword-spotting energy	2.8 µJ per inference [16]	280 µJ per inference (est.)	0.9 µJ per inference [16]
Deep-sleep current	2.8 µJ	2.5 mA (power-management IC)	4.5 µA

The data are consistent with the architectural analysis. The MCU-class devices deliver an order of magnitude better performance per watt for typical IoT workloads especially when the workload is event-driven with long idle periods. The advantage holds for edge AI workloads too: a keyword-spotting inference on a Cortex-M4 uses two orders of magnitude less energy than the MPU, mostly because the MPU must keep external RAM in self-refresh and repeatedly re-initialise OS states. The MCU energy advantage is further magnified with the addition of an on-die NPU (Ethos-U55) , where inference energy is pushed into the nanojoule range.

Conversely, the MPU's raw performance and memory headroom is invaluable for application workloads that need the full Linux stack, managing a complex file system, or memory intensive data analytics. SPEC CPU and MiBench [17] benchmark data show that Cortex-A cores are 5-10× faster than Cortex-M devices for general-purpose integer and floating-point tasks, making them the only viable choice for data-intensive edge gateways.

VIII. CONCLUSION

MCU and MPU architectures for Internet of Things systems were systematically compared in this article. It offers an organized framework for architects and designers choosing between MCUs and MPUs by looking at six crucial factors: processing capacity, energy consumption, real-time determinism, memory architecture, peripheral availability, and development complexity.

The data demonstrates that MCUs continue to be the preferred platform for low-power, real-time, sensor-focused applications. Their single-chip integration, sub-microampere sleep currents, and deterministic interrupt behavior enable battery-powered devices with multi-year lifetimes. On the other hand, despite the drawbacks of higher power consumption and more complicated boards, MPUs are essential for applications needing sophisticated multitasking operating systems, big memory footprints, and high-bandwidth data processing.

The old gap, importantly, is closing. Today's MPUs have real-time auxiliary cores and dynamic power scaling, while advanced MCUs now have external memory controllers, neural accelerators and TrustZone-M security. Heterogeneous SoCs, combining both architectures on a single silicon chip, are on the rise. TinyML is driving a tiered edge-intelligence architecture, and it has further enabled MCUs to perform intelligent activities that were the exclusive province of MPUs.

Three topics should be the focus of future research. First, the existing CoreMark-centric metrics must be replaced with a uniform benchmarking technique in order to objectively evaluate event-driven, intermittent workloads across MCU and MPU platforms. Second, IoT design would be significantly accelerated by formal design-space exploration tools that automatically suggest an ideal architecture or hybrid configuration given a workload definition, energy budget, and real-time limitations. Finally, more research should be done on holistic security frameworks that measure the attack surface and energy cost of hardware-enforced security across both architectures given the expanding threat landscape.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

REFERENCES

[1] T. Adegbija, A. Rogacs, C. Patel, and A. Gordon-Ross, "Microprocessor Optimizations for the Internet of Things: A Survey," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 1, pp. 7–20, Jan. 2018, doi: 10.1109/TCAD.2017.2717782.

[2] I. Sultan and M. T. Banday, "Ultra-Low Power Microcontroller Architectures for IoT Devices," in *Proc. IEEE ICSSIT*, 2023, pp. 482–488, doi: 10.1109/ICSSIT55814.2023.10060949.

[3] A. M. Afachao and A. M. Abu-Mahfouz, "A Review of Intelligent IoT Devices at the Edge," in *Proc. ICAIoT*, 2022, doi: 10.1109/ICAIoT53028.2022.9765432.

[4] D. B. Thomas and W. Luk, "FPGA Accelerated Simulation of Microprocessor-Based Systems," *IEEE Trans. VLSI Syst.*, vol. 17, no. 11, pp. 1527–1539, Nov. 2009, doi: 10.1109/TVLSI.2008.2004564.

[5] F. Campi, "Power-Shaping Microprocessor Systems for IoT," in *Proc. DAC*, 2017, doi: 10.1145/3061639.3062285.

[6] L. Benini, A. Pullini, and D. Rossi, "Ultra-Low Power Parallel Computing for the Internet of Things," *IEEE Micro*, vol. 36, no. 5, pp. 38–47, Sep./Oct. 2016, doi: 10.1109/MM.2016.78.

[7] P. P. Ray, "A Survey on Internet of Things Architectures," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 30, no. 3, pp. 291–319, 2018, doi: 10.1016/j.jksuci.2016.10.003.

[8] S. Mittal, "A Survey of Techniques for Improving Energy Efficiency in Embedded Computing Systems," *Int. J. Comput. Aided Eng. Technol.*, vol. 6, no. 4, pp. 440–459, 2014, doi: 10.1504/IJCAET.2014.064759.

[9] M. Shafique, T. Theocharides, S. Garg, and J. Henkel, "Robust Machine Learning Systems: Challenges, Current Trends, Perspectives, and the Road Ahead," *IEEE Des. Test*, vol. 37, no. 2, pp. 30–57, 2020, doi: 10.1109/MDAT.2020.2973592.

[10] W. Shi and S. Dustdar, "The Promise of Edge Computing," *Computer*, vol. 49, no. 5, pp. 78–81, May 2016, doi: 10.1109/MC.2016.145.

[11] M. Ali, J. Zhou, and S. Chen, "Edge Computing for Internet of Things: A Survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 450–465, Feb. 2018, doi: 10.1109/JIOT.2017.2778502.

[12] J. Lin, W.-M. Chen, Y. Lin, C. Gan, and S. Han, "MCUNet: Tiny Deep Learning on IoT Devices," in *Adv. Neural Inf. Process. Syst.*, 2020, doi: 10.48550/arXiv.2007.10319.

[13] J. Ko, C. Lu, M. Srivastava, J. Stankovic, A. Terzis, and M. Welsh, "Wireless Sensor Networks for Healthcare," *Proc. IEEE*, vol. 98, no. 11, pp. 1947–1960, Nov. 2010, doi: 10.1109/JPROC.2010.2065210.

[14] A. Haseeb, M. Faizan, M. F. Khan, and M. T. Iqrar, "Microprocessors and Microcontrollers: Past, Present, and Future," in *Functional Reverse Engineering of Machine Tools*, CRC Press, 2019, doi: 10.1201/9780429022876-2.

[15] R. Cheour et al., "Microcontrollers for IoT: Optimizations, Computing Paradigms, and Future Directions," in *IEEE WF-IoT*, 2020, doi: 10.1109/WF-IoT48130.2020.9221471.

[16] C. Banbury et al., "MLPerf Tiny Benchmark: Benchmarking TinyML Systems," *arXiv*, 2021, doi: 10.48550/arXiv.2106.07597.

[17] M. R. Guthaus et al., "MiBench: A Free, Commercially Representative Embedded Benchmark Suite," *IEEE Micro*, vol. 21, no. 4, pp. 26–31, 2001, doi: 10.1109/40.943921.

[18] S. N. Matheu et al., "A Survey of Cybersecurity Certification for the Internet of Things," *ACM Comput. Surv.*, vol. 53, no. 4, pp. 1–36, 2020, doi: 10.1145/3372725.