We sincerely thank the Editor-in-Chief, Associate Editors, and all reviewers for their valuable time, insightful feedback, and constructive comments on our manuscript, “Design and Implementation of a Low-Power Memristor-Based Piccolo-80 Lightweight Encryption Algorithm Using VTM Logic Gates” (Manuscript ID: CJECE-OA-2025-Jul-160). In this revised submission, we have thoroughly addressed all reviewer feedback by providing detailed clarifications, adding additional experimental results, and expanding the discussions to improve further the quality, clarity, and reproducibility of the manuscript.

Reviewer: 1

1. **What is missing in the previous works, which is then fulfilled in this paper. Also, why are some of the previously reported works not enough for the current cryptography applications?**
   We sincerely thank the reviewer for their insightful question, which emphasizes the novelty and importance of our work.
   Previous implementations of the Piccolo cryptographic algorithm on FPGA, CMOS, and MeMOS platforms still had several limitations. They showed high dynamic power due to switching activity, significant hardware overhead and area, limited scalability for low-energy devices, and side-channel vulnerabilities such as DPA leakage from nonlinear layers like the S-Box. In the present study, we address these issues by implementing the entire Piccolo round function, including the S-Box, MixColumn over GF(2^4), and permutation layer, using designed VTM stateful logic gates (NAND/NOR, AND/OR, XOR, and XNOR). Additionally, earlier studies rarely analyzed power consumption across cryptographic components; our results show that the S-box consumes about 43% of dynamic power, diffusion roughly 29%, permutation around 18%, and key-mixing approximately 10%. By addressing these gaps, we present a complete round-based Piccolo-80 hardware implementation (94 XOR/XNOR, 68 AND/OR, 32 NAND/NOR per round), verified with Cadence, achieving 17.4 mW at 1.8 V and 133 MHz with 1214 GEs, demonstrating improved power efficiency and reduced area compared to CMOS and MeMOS designs. Furthermore, the throughput-to-area efficiency reaches 0.280 Mbps/GE, showing an improvement over the 0.189 Mbps/GE and 0.227 Mbps/GE reported for the protected and unprotected MeMOS designs.

2. **Demonstrate the relevance of the proposed hardware architecture in achieving better results, specifically showing that these results are due to the hardware design and methodology rather than the tools and technologies used.**
   We sincerely appreciate this valuable comment, as it clarifies that the observed improvements are due to architectural design rather than simulation tools. The performance enhancements are attributed to specific hardware choices:
   (i) In our work, we used *stateful memristor-based VTM logic gates* to implement the Piccolo round function. These gates eliminate the need for extra storage elements, thereby reducing circuit complexity.
   (ii) Efficient two-input VTM logic gates, which share the same structure but are programmed with different input voltages (e.g., NAND/NOR, AND/OR), decrease the total gate count.
   (iii) Most importantly, the single-step operation of VTM gates with inherent state retention minimizes switching activity, which directly lowers dynamic power consumption and improves energy efficiency.

   All designs were tested using the same Piccolo-80 workload (64-bit block, 80-bit key, 25 rounds), and the results reported included Gate Equivalents (GEs), throughput, and throughput-per-area to remove tool-specific effects. These findings indicate that the improvements are due to the architectural design and methodology, rather than the specific tools or technologies used.

3. **What are the advantages of the proposed method in terms of hardware and software implementations?**
   We thank the reviewer for raising this excellent point, which helps us highlight both the hardware and software advantages of our approach.
   On the hardware side, one of the main advantages of our proposed architecture comes from how we handle the diffusion layer. Multipliers are typically among the most expensive hardware blocks in terms of both area and power consumption. In our design, we replaced sequential or LUT-based multipliers with a parallel GF(2^4) multiplier that performs the multiplication in a single clock cycle. This parallel structure is implemented using just 16 XOR and 16 NAND gates (or 18 XOR and 12 NAND gates), where the critical path consists of 4 XOR gates and a single NAND/AND gate. As a result, the multiplication key in the MixColumn step shifts from a multi-cycle, energy-intensive process to an efficient, single-cycle operation. As a result, the parallel GF(2^4) multiplier performs the main diffusion computation in a single cycle with low gate count and short logic depth. This design reduces both power consumption and area while increasing throughput and efficiency, which clearly demonstrates the advantage of the chosen architecture. Furthermore, stateful VTM gates inherently combine logic and storage, eliminating unnecessary storage elements and resulting in a simpler, smaller, and more power-efficient design. Their single-step operation also lowers latency and simplifies control logic compared to IMPLY or MAGIC families, which require multiple steps.
   From a software and algorithm perspective, the proposed architecture directly supports all primary operations of Piccolo-80, including the S-box, MixColumn, key combination, and permutation, in hardware. This eliminates the need for additional software processing and reduces overhead. Furthermore, the modular and parallel design aligns with the natural structure of Piccolo, enabling more efficient hardware–software co-design, faster simulation, and simpler verification.

4. **It is better to compare the proposed work with other hardware implementations of different block ciphers.**
   We sincerely appreciate this insightful comment, as a comprehensive comparison clearly highlights the relevance and competitiveness of our contribution. In the revised manuscript, we have expanded the evaluation by comparing the proposed VTM-based Piccolo-80 implementation with reported hardware results of several well-known lightweight block ciphers. The comparative results are now presented in Table XI, which includes implementations such as ASCON, CLEFIA, PRINCE, PRESENT, LED, Camellia, SIMON, and SPECK. This thorough analysis shows that our design achieves one of the smallest hardware areas while maintaining competitive power dissipation and reliable performance, thereby reinforcing its suitability for lightweight and energy-constrained IoT applications.

5. **Please discuss your contributions versus the following publications in the introduction or comparison sections.**
   We sincerely thank the reviewer for this excellent suggestion, as it helps us clarify how our work fits within the broader field. The related publications have been discussed in both the Introduction and a newly added section, G. Comparison with Other Lightweight Block Ciphers, where a more detailed comparison is provided.

   I. **HIGHT and PRESENT block ciphers (Microelectronics Journal, 2019):**

In the field of lightweight cryptography, Rashidi (2019) concentrated on optimizing FPGA/CMOS-based implementations of the HIGHT and PRESENT ciphers. Their contributions include employing parallel-prefix adders (Ladner–Fischer, Han–Carlson, Kogge–Stone, Sklansky) for modular addition, algebraic simplification of PRESENT S-boxes, loop unrolling to increase throughput, and latch-based glitch filtering to minimize transient switching activity. They also performed side-channel resistance analysis, including power and timing attacks, and validated their designs on Virtex-5 and Spartan-3 FPGAs. This work shows how architectural and logic-level techniques can enhance execution time, throughput, and throughput-to-area ratio for traditional lightweight block ciphers.

II. CLEFIA 128-bit block ciphers (IET CDT, 2019/2020):

In 2020, Rashidi introduced efficient and flexible hardware structures for the CLEFIA block cipher, focusing on 128-bit data blocks with variable key sizes of 128, 192, and 256 bits. Their contributions focused on designing a unified processing element for the generalized Feistel network, simplifying the logic of the S0 S-box using Karnaugh mapping and algebraic factorization, and implementing the S1 S-box inversion in a composite field $F(2^4)$ instead of $F(2^8)$ to significantly reduce area cost. They also proposed flexible structures supporting different key sizes, emphasizing flexibility and scalability in cryptographic applications. These optimizations, implemented in 180 nm CMOS technology, showed improvements in execution time, throughput, and throughput/area efficiency over previous CLEFIA designs.

III. PRINCE Lightweight Block Cipher (IJCTA, 2020):

In Rashidi's work on the PRINCE lightweight cipher, the focus is on optimizing classical CMOS/FPGA hardware to achieve efficient trade-offs between area and speed. Two main architectures are introduced: a low-cost structure, in which S-boxes and inverse S-boxes are shared between forward rounds and the middle step to minimize resources, and a two-cycle design based on reconfigurable processing elements that significantly reduce latency. The paper also presents optimized logic designs for S-boxes and efficient matrix multiplications ($M/M^{-1}/M^0$), as well as a compact round-constant generator exploiting the α\alphaα-reflection property. These contributions collectively target short critical path delays, higher throughput, and competitive area in conventional CMOS and FPGA platforms, especially for RFID and latency-sensitive applications.

IV. PRESENT, SIMON, and LED (IET CDS, 2020):

Rashidi's work on flexible hardware structures of the PRESENT, SIMON, and LED block ciphers focused on improving configurability and throughput within conventional CMOS platforms. By designing area-optimized S-boxes and efficient MixColumns implementations, the study introduced reconfigurable architectures capable of supporting multiple key sizes and, in the case of SIMON, different block sizes. The key contribution lies in providing flexibility through control signals, allowing a single hardware structure to adapt its security level according to the needs of different IoT applications. Implemented in 180 nm CMOS, these designs achieved improved throughput-to-area efficiency and demonstrated their suitability for resource-constrained environments that nonetheless require adjustable levels of cryptographic strength.

V. Camellia Block Cipher for Security of the IoT (IET CDT, 2020):

Rashidi's work on the *Camellia* block cipher introduced flexible and high-throughput CMOS hardware structures targeted at IoT applications. By employing composite-field arithmetic for S-Box inversion, extensive use of 2-input NAND/NOR gates, and resource-sharing techniques, the proposed architecture reduced both critical path delay and area overhead. A key contribution was the ability to reconfigure the cipher to support multiple key sizes (128, 192, and 256 bits), thereby

adapting the security level based on application needs. This flexibility allowed a single unified architecture to handle different cryptographic requirements while maintaining competitive throughput and throughput/area ratios in 180 nm CMOS. Overall, the emphasis of Rashidi's study was on reconfigurability and speed optimization within conventional silicon logic platforms for robust IoT security.

VI. SIMON and SPECK Lightweight Block Ciphers (IJCTA, 2019):

Rashidi's study on SIMON and SPECK presented high-throughput and flexible ASIC architectures implemented in 180 nm CMOS. The designs employed a Sklansky parallel-prefix adder for modular addition in SPECK to reduce critical path delay and improve performance. For SIMON, a tree-structured XOR was introduced to shorten logic depth, enabling higher operating frequency. Additionally, the proposed work features flexible hardware structures that support multiple block sizes (64, 96, 128 bits) and key lengths (128, 144, 192, 256 bits), allowing the same architecture to scale across various security levels and application domains. These implementations emphasized critical path reduction, throughput improvement, and configurability in ASIC platforms, providing versatile solutions for lightweight cryptography within conventional CMOS logic.

VII. Glitch-less Hardware (Integration/VLSI Journal, 2020/2022):

Rashidi's work on glitch-less block cipher implementations addresses the problem of unwanted short pulses (glitches) in CMOS hardware designs of lightweight ciphers such as PRESENT, HIGHT, and SPECK. These glitches, caused by unequal propagation delays in logic paths, increase dynamic power consumption and vulnerability to side-channel attacks. The proposed solution introduced a glitch filter circuit that not only removes chatter and extraneous switching activity but also doubles as a register to store intermediate data. Implemented in 180 nm CMOS, the approach achieved glitch-free signals with negligible area overhead while maintaining competitive throughput and area metrics. Thus, the main contribution of this work lies in logic-level reliability and power stabilization within conventional CMOS/FPGA implementations, reducing switching noise without altering the underlying cipher algorithms.

Following the notable works of Rashidi that enhance lightweight block ciphers through algorithmic and CMOS/FPGA-level optimizations—such as high-throughput and area-efficient designs for HIGHT and PRESENT, multi-key-size flexibility for CLEFIA, performance tuning for PRINCE on standard platforms, reconfigurable CMOS architectures for Camellia, and glitch-filtering to stabilize traditional CMOS logic—this study explores a different aspect of innovation: device-level design using emerging memristive technologies.

We focus on a different cipher (Piccolo-80) and a different technological substrate, presenting a memristor-based implementation built on the Voltage-to-Memristance (VTM) framework.

The main innovation is mapping the entire Piccolo-80 datapath, including S-Box substitution, GF(2^4) diffusion, permutations, and key mixing, onto stateful VTM logic gates that combine storage and computation, thereby executing Boolean functions in a single step. In our design, we replaced sequential or lookup table-based multipliers with a parallel GF(2^4) multiplier that completes multiplication in a single clock cycle. This parallel structure utilizes only 16 XOR and 16 NAND gates (or 18 XOR and 12 NAND gates), with the critical path comprising 4 XOR gates and one NAND/AND gate. As a result, the MixColumn multiplication shifts from a multi-cycle, energy-consuming process to an efficient, single-cycle operation. The parallel GF(2^4) multiplier performs the main diffusion computation in one cycle, achieving a low gate count and shallow logic depth, which reduces power consumption and area while increasing throughput and

efficiency. This clearly highlights the benefits of the proposed architecture. By storing information as resistance instead of charge, the VTM approach minimizes unnecessary charge movement, thereby reducing switching activity, dynamic power consumption, and hardware overhead, all while maintaining the algorithm's correctness. This device-level perspective addresses the root cause of excessive transitions, complementing previous CMOS-focused strategies that reconfigure algorithms or architectures or suppress glitches afterward.
Under this architecture, the design achieves 17.4 mW at 133 MHz with a hardware footprint of only 1214 gate equivalents (GEs), resulting in up to 32% lower power consumption and nearly 20% smaller area compared to hybrid MeMOS implementations. Simulation results from Cadence Spectre software further confirm the power and hardware efficiency of the proposed design. The implementation consumes just 17.4 mW of power at 1.8 V and 133 MHz, using less space than comparable designs. Additionally, the throughput-to-area efficiency reaches 0.280 Mbps/GE, improving upon the 0.189 Mbps/GE and 0.227 Mbps/GE reported for protected and unprotected MeMOS designs. Gate-level power analysis also shows that the substitution layer accounts for approximately 43% of the dynamic power, followed by diffusion (29%), permutation (18%), and key mixing (10%), highlighting the primary power hotspots and validating the effectiveness of VTM-based optimization. Furthermore, the inherent non-volatility and reduced switching activity of VTM gates significantly lower power leakage, thereby increasing resistance against side-channel attacks such as Differential Power Analysis (DPA). Overall, the two lines of research complement each other: Rashidi's contributions improve throughput, flexibility, and robustness within established CMOS/FPGA flows, while our work demonstrates that emerging device technologies can provide ultra-low-power, scalable, and more secure implementations of lightweight cryptography. This positions VTM-based memristive architectures as not only compact and energy-efficient but also more resistant to power Analysis attacks, making them highly suitable for resource- and energy-constrained IoT environments where both performance and security are essential.

Reviewer: 2

1. **Figure 3 should be resized to fit the paper layout without leaving large blank spaces on the right. In addition, the captions above and below the figure should be revised for clarity.**
   We appreciate the reviewer's suggestion. Figure 3 has been resized to eliminate blank space, and its captions have been revised for clarity.
2. **In Equation (6), each term is written with an XOR operation with "1," which is redundant and potentially misleading. Moreover, in the definition of t_2, the repeated appearance of p_0q_0 results in an expression with limited meaningful interpretation.**
   We appreciate the reviewer's valuable comment. In the revised manuscript, the unnecessary "$\oplus 1$" terms have been removed, and the repeated "p_0q_0" in the definition of t_2 has been corrected.

   $$t_3 = p_0q_3 \oplus p_1q_2 \oplus p_2q_1 \oplus p_3q_0 \oplus p_3q_3$$

   $$t_2 = p_0q_2 \oplus p_1q_1 \oplus p_2q_1 \oplus p_2q_0 \oplus p_2q_3 \oplus p_3q_2 \oplus p_3q_3$$

   $$t_1 = p_0q_1 \oplus p_1q_0 \oplus p_1q_3 \oplus p_2q_2 \oplus p_2q_3 \oplus p_3q_1 \oplus p_3q_2$$

   $$t_0 = p_0q_0 \oplus p_1q_3 \oplus p_2q_2 \oplus p_3q_1$$

3. **The title of Table III refers to "NAND/NOR," but the contents correspond to AND/OR logic outputs, which is inconsistent. For Tables VIII and IX, the reported hardware implementation results lack details on design assumptions (e.g., simulation parameters,**

process node), making them less convincing. The manuscript should provide a more rigorous description of the implementation methodology.

We thank the reviewer for bringing these issues to our attention. We have fixed the inconsistency in Table III: the title now correctly reflects the logic functions shown (changed from “NAND/NOR” to “AND/OR” to match the table contents).

We sincerely thank the reviewer for the valuable comment regarding the hardware implementation results (Tables VIII and IX). In the revised manuscript, the FPGA platform implementation has been clarified in detail: the Piccolo-80 design was synthesized on a Xilinx Virtex-5 XC5VFX200T FPGA fabricated in 65 nm CMOS technology, with functional verification performed in ModelSim at a 10 ns clock period (100 MHz). Post-synthesis timing analysis using Xilinx ISE reported a maximum operating frequency of 427.716 MHz.

For the memristor-based implementation, simulations were conducted in Cadence Virtuoso using the JART VCM v1b model, with a supply voltage of 1.8 V and an operating frequency of 133 MHz. Power dissipation was estimated in Cadence Spectre by separating static and dynamic components. As memristors, unlike CMOS technology, are not defined by a specific process node, in this work, they are described based on the adopted physical device model (JART VCM v1b) and nanometer-scale thin-film parameters, since in memristive architectures, the characterization depends more on the composition of layers, thin-film thickness, and fabrication process rather than on conventional lithographic channel length scaling.

4. **Figure 10, which presents the “Architecture of the Piccolo algorithm using VTM stateful logic gates,” represents the core innovation of this work. This section should be expanded and described in greater detail.**

   We sincerely thank the reviewer for this valuable comment, which allows us to provide a more precise and detailed explanation of Figure 10.

   Figure 10 illustrates the entire architecture of Piccolo-80, constructed using VTM stateful logic gates. In this design, a 64-bit input is initially divided into four 16-bit words, each of which is divided into four 4-bit nibbles. These data blocks are combined with encryption keys, which consist of two parts: the whitening keys (WKs), used at the start to establish key dependence, and the round keys (RKs), XORed with the data at specific stages—particularly before the S-boxes or after intermediate steps. The nibbles are processed in parallel through S-box layers, implemented using VTM-based XOR/XNOR and AND/OR gates. The basic design of these gates is illustrated in Figure 2 and serves as the foundation for all logical operations in the architecture. The results from the S-boxes are directed to the MixColumn block, where multiplication over GF(2^4) is performed modulo x^4+x+1. This multiplier’s hardware design, shown in Figure 4, is directly based on Equation (6) and fully implemented with the proposed VTM gates. After the MixColumn operation, the data passes through another S-box layer and is then XORed with round keys before entering the Round Permutation (RP) block (Figure 3), which applies a fixed permutation to the nibbles. The key advantage of this architecture is that all stages—from key addition to S-boxes, the GF(2^4) multiplier, and the round permutation—are built entirely with VTM stateful gates. Each gate performs both computation and storage in a single step, eliminating the need for intermediate registers, reducing switching activity, and significantly lowering power consumption and hardware size compared to traditional CMOS and hybrid MeMOS designs.

5. **The FPGA implementation is presented primarily as functional validation, but no direct same-platform comparison with the proposed VTM architecture is provided. This weakens the strength of the performance claims.**

We sincerely thank the reviewer for bringing this important point to our attention. As clarified in the revised manuscript, the FPGA implementation of Piccolo-80 was primarily introduced to validate the algorithm's functionality on reconfigurable hardware. These results are now reported separately in Table IX, using platform-specific metrics such as Slices, LUTs, frequency, throughput, and Power Consumption. They are not intended for direct comparison with circuit-level implementations. Our performance claims regarding area and power are supported by the ASIC-level results summarized in Table X, where the proposed VTM design is compared with previous MeMOS implementations in terms of gate equivalents (GEs), power dissipation, throughput, and throughput-to-area efficiency.

6. **In Table IX, the first part reports FPGA results using Slices, which reflect logic resource usage on specific FPGA devices. However, because Slice definitions differ across FPGA families, these results cannot be directly compared in terms of area. Later in the same table, the metric changes to Gate Areas (GAs), i.e., 2-input NAND equivalents, which introduces inconsistency in the comparison methodology. A unified and consistent area metric should be adopted.**
   We sincerely thank the reviewer for highlighting this important issue for our attention. We agree that displaying FPGA results in Slices alongside circuit-level results in Gate Equivalents (GEs) in the same table creates inconsistency, as Slice definitions vary across FPGA families and are not directly comparable to GE-based area metrics. To address this, we have revised the manuscript by separating the results in Table IX into two tables. The revised Table IX now includes only FPGA implementation platforms, reported in terms of Slices, LUTs, frequency, efficiency, and power consumption, which are platform-specific metrics used mainly for functional validation. The revised Table X presents circuit-level implementations, reported consistently in terms of GEs, throughput, and power dissipation, providing a uniform basis for comparing MeMOS and the proposed VTM design.

7. **The statement that digital circuits consume more power when processing key bits with value “1” than with value “0” is not entirely accurate. Power consumption in CMOS circuits is primarily determined by switching activity (signal transitions), not by the static logic value. The authors are encouraged to revise this explanation to reflect a more rigorous understanding of dynamic power and its relevance to side-channel leakage.**
   We appreciate the reviewer’s insightful comment. We agree that the original statement was not sufficiently precise. Power consumption in CMOS circuits is primarily determined by dynamic switching activity rather than the static logic value of “0” or “1.” Consequently, side-channel leakage arises from correlations between this switching activity and the processed key bits. An attacker can measure the power at specific times and detect even slight variations in switching activity, which may help infer the key bits. S-Box operations are a significant source of side-channel leakage, allowing attackers to exploit power analysis to recover whitening and round keys.

8. **The manuscript infers resistance against DPA attacks based on circuit-level reasoning (reduced switching activity and power fluctuations) but does not present actual DPA**

experiments or statistical analysis. For stronger validation, at least basic leakage assessment (e.g., TVLA t-test or correlation analysis) should be provided.
We sincerely thank the reviewer for this valuable comment. We agree that the current manuscript suggests resistance to Differential Power Analysis (DPA) attacks based solely on architectural reasoning—specifically, reduced switching activity and stabilized power profiles—without including actual DPA experiments. Conducting a comprehensive leakage assessment, such as the TVLA t-test or correlation-based analysis, would undoubtedly strengthen the validation. However, such experiments were beyond the scope of this work, which primarily focuses on architectural design and circuit-level simulation, with emphasis on power consumption and area utilization. In response to the reviewer's concern, we have revised the manuscript to explicitly acknowledge this limitation and clarify that our claims are based on circuit-level observations rather than empirical attack validation. We also emphasize that incorporating statistical side-channel evaluation is an essential direction for future research, and we plan to conduct such experiments in follow-up studies.

9. **Several references cited in the manuscript are relatively old. The authors are encouraged to update the literature review with more recent works (2023–2025), particularly in lightweight cryptographic hardware implementations, to strengthen the relevance and currency of the study.**
We thank the reviewer for highlighting the importance of incorporating recent literature. In the revised manuscript, the Introduction and Discussion sections have been updated to include studies from 2023 to 2025 that strengthen the currency and relevance of lightweight cryptographic hardware research. In particular, we have added:

- ASCON hardware (NIST LWC winner): recent FPGA and ASIC implementations that examine folded versus unrolled pipelines, reporting area, throughput, power trade-offs, and side-channel considerations.
- Device-level and in-memory approaches: new results on RRAM-based lightweight encryption (e.g., GIFT) and in-memory AES, highlighting logic-in-memory computation, non-volatility, and improved efficiency compared with CMOS-only designs.
- Emerging building blocks: studies on SRAM-based compute-in-memory (CiM) S-boxes and chaos-based S-boxes, which target low area and improved side-channel resistance.
- Piccolo security context: a 2023 study presenting a CPA-resilient Piccolo-80 implementation validated on FPGA platform, demonstrating the need for targeted side-channel protections and supporting our positioning on device-level security.

These additions ensure that the manuscript reflects the latest advancements (2023–2025) in lightweight cryptographic hardware and emerging device-level directions.

Reviewer: 3

1. **The abstract of the paper should be improved. The first sentence can state the importance of the content, then the gaps in the corresponding literature. Key contributions of the paper should be expressed clearly, and then the paper's major findings should be provided. In particular, suggest adding a couple of sentences highlighting the contributions of this work to the state of the art and the major potential implications from the conducted research.**
   We sincerely thank the reviewer for this valuable comment and constructive suggestion regarding the abstract. We agree that the abstract should emphasize the importance of the work, identify gaps in the literature, clearly state the key contributions, and summarize the main findings. Therefore, we have revised the abstract as follows: (i) we added context on the growing importance of lightweight cryptography in energy-constrained IoT environments; (ii) we highlighted the limitations of previous CMOS, FPGA, and MeMOS implementations of Piccolo-80; (iii) we explicitly stated the main contributions of this study, including the complete round-accurate mapping of Piccolo-80 operations to VTM memristor gates, detailed power profiling, and comparative efficiency; and (iv) we clarified the significant results, showing reduced power and area along with potential implications for secure IoT applications. The revised abstract now provides a clearer overview of the motivation, contributions, and outcomes of this work.

2. **The motivation of the proposed method should be stated in the introduction.**
   We sincerely thank the reviewer for this valuable comment. We agree that the motivation for our proposed method, as outlined in the introduction, is essential. Therefore, we have added the following paragraph at the end of the prior works discussion: While previous studies mainly focused on architectural and logic-level optimizations of Piccolo-80 in conventional CMOS, FPGA platforms, and hybrid MeMOS technologies, these implementations still face significant challenges, including high dynamic power consumption, increased circuit area, additional hardware overhead, limited scalability, and increased vulnerability to side-channel attacks. These limitations make them less suitable for energy-constrained environments, such as IoT devices, and highlight the need for alternative hardware solutions. This work introduces a new memristor-based design for Piccolo-80 using the VTM method. By utilizing the non-volatile nature of memristors, the proposed architecture enables stateful single-step operations, reduces switching activity, features compact logic components, and consumes less power. The entire round function, including the S-box and MixColumn operations, is implemented entirely with VTM stateful logic gates. This architecture addresses the power, area, and scalability challenges of earlier designs, emphasizing the potential of emerging memristor technologies for secure, energy-efficient IoT applications.

3. **How does the VTM approach improve upon traditional memristor-CMOS hybrid (MeMOS) designs?**
   We sincerely thank the reviewer for this insightful question. Traditional hybrid MeMOS designs combine memristors with CMOS transistors, which reduce area compared to pure CMOS; however, they still face significant drawbacks, such as high dynamic power consumption, hardware overhead, and limited scalability due to their reliance on CMOS circuitry. The proposed VTM approach improves this by enabling stateful, single-step logic operations directly within memristors. In this approach, basic logic structures, such as NAND/NOR or AND/OR, are implemented using memristive elements. By applying two different voltages, the same structure can be reconfigured to perform the function of another gate. This capability reduces design complexity, minimizes switching activity, and increases flexibility in hardware implementation. As a result, the VTM-based architecture achieves lower dynamic power, fewer gates, and better

scalability compared to hybrid MeMOS. Simulation results further confirm that the proposed method reduces power consumption by up to 32% and area by nearly 20% relative to MeMOS implementations, while maintaining cryptographic strength and performance. Furthermore, the throughput-to-area efficiency achieves 0.280 Mbps/GE, exceeding the 0.189 Mbps/GE and 0.227 Mbps/GE reported for the protected and unprotected MeMOS designs.

4. **What is the role of single-step stateful logic in reducing circuit complexity?**
   We thank the reviewer for this insightful question. In traditional CMOS or hybrid MeMOS circuits, logic functions usually require multiple transistors and cascaded stages, which increase the number of devices, interconnections, and switching activity. In contrast, the proposed VTM-based architecture implements all essential logic gates (AND/OR, NAND/NOR, XOR, and XNOR) as single-step, stateful operations directly within memristive elements. This approach eliminates the need for cascaded gate combinations, reduces intermediate switching, and streamlines the routing process. As a result, circuit complexity is significantly decreased, leading to a smaller size, lower dynamic power consumption, and enhanced scalability for cryptographic hardware in resource-constrained environments.

5. **How are the S-Boxes in Piccolo-80 implemented using VTM-based memristors?**

   We sincerely thank the reviewer for this important question. In this work, the 4-bit S-Box of Piccolo-80 is implemented entirely using the proposed VTM-based memristive logic. The nonlinear substitution is achieved with a lightweight combination of four NOR gates, three XOR gates, and one XNOR gate, interconnected through their outputs and inputs to realize the nonlinear mapping. This hardware-oriented approach eliminates the need for a lookup table (LUT) and ensures that each output bit depends on multiple input bits, providing the necessary nonlinearity and diffusion required for cryptographic strength. By executing logic operations in a single step and storing states in memristive resistance instead of charge, the VTM gates simplify hardware design, reduce area, and cut circuit complexity and power consumption by minimizing switching activity. These features not only improve energy efficiency but also enhance robustness against side-channel attacks, such as Differential Power Analysis (DPA).

6. **What trade-offs exist between parallelism and power consumption in this architecture?**

   We appreciate the reviewer's insightful question. In the proposed VTM-based memristor design, the main trade-off between parallelism and power consumption occurs during GF(2^4) multiplication, one of the most resource-intensive operations in the Piccolo-80 cipher. In our design, this multiplication is performed using a compact parallel GF(2^4) multiplier that completes the operation in a single cycle, requiring as few as 16 XOR and 16 NAND gates (or 18 XOR and 12 NAND gates), thereby significantly reducing both latency and hardware overhead. Implementing this operation in parallel decreases critical path delays and increases throughput, but also raises the number of gates, which can lead to higher switching activity. To address this, the architecture utilizes single-step VTM logic gates and an area-optimized circuit design, thereby reducing switching activity and hardware complexity compared to CMOS/MeMOS solutions. This approach balances higher throughput, strong energy efficiency, and a compact area, making it ideal for resource-constrained IoT devices.

7. **How does the proposed design mitigate leakage against DPA attacks compared to CMOS?**

We sincerely thank the reviewer for this valuable question. In traditional CMOS-based implementations, dynamic power consumption is strongly correlated with data-dependent switching activity, and the relationship between processed data and power consumption is a significant source of exploitable leakage under Differential Power Analysis (DPA). Since power consumption in CMOS circuits primarily depends on switching activity, even minor changes can be detected and exploited by an attacker to extract sensitive information. In contrast, the proposed VTM-based memristor design reduces this relationship because logic states are stored as stable resistance values rather than transient charges. This mechanism minimizes switching activity and stabilizes power traces, thereby decreasing the variations that attackers could exploit. As a result, the VTM-based implementation provides greater resistance against DPA leakage.

8. **What simulation parameters (e.g., voltage, frequency) were chosen, and why?**

   The authors appreciate the reviewer's important question. The design was simulated in the Cadence Spectre environment using the JART VCM v1b memristor model. In prior work on VTM-based logic (2021), the proposed logic gates were identified as suitable operating points at a supply voltage of 1.8/3.3 V with a memristor threshold of 1 V. This operating point provided stable and reliable gate performance while maintaining low dynamic power consumption. The threshold of 1 V aligns with the switching characteristics of the memristor model, allowing consistent resistance-state transitions without excessive leakage or delay. The simulation was run at 133 MHz, which offers an effective trade-off between throughput and energy efficiency.

9. **How scalable is the proposed architecture for larger key sizes (e.g., Piccolo-128)?**

   We appreciate the reviewer for raising this important question. The internal structure of Piccolo remains consistent across both the 80-bit and 128-bit versions, with differences mainly in the number of rounds and the key schedule. Therefore, the datapath of the proposed architecture—comprising logic components such as S-Boxes, GF(2^4) multipliers, and permutation networks—can be used entirely without modification. The only modifications involve the round control logic and a slight increase in memory to support the longer key. In Piccolo-128, increasing the number of rounds from 25 to 31 results in proportional increases in total encryption time and energy consumption, while the critical path delay per round and hardware area remain mostly unchanged. Additionally, using the proposed VTM-based memristor logic reduces hardware complexity and switching activity, supporting low power consumption and a compact hardware design. As a result, even in Piccolo-128, key features like low power, small size, and scalability are preserved.

10. **How does the proposed design compare in throughput with other lightweight ciphers such as PRESENT or KATAN?**

    We sincerely thank the reviewer for this question. In the proposed VTM-based implementation, Piccolo-80 achieves a throughput of 340 Mbps at 133 MHz, corresponding to 0.28 Mbps/GE. Under comparable conditions, PRESENT reaches approximately 1492.5 Mbps at 100 MHz; thus, the reported throughput of the proposed design is lower than that of PRESENT. By contrast, KATAN implementations typically employ an iterative 254-round architecture, focusing on area minimization, which results in lower throughput. Additionally, the proposed design uses only 1214 GE—much less than PRESENT (4214 GE) and KATAN (6768 GE)—while offering competitive power dissipation of 17.4 mW at 133 MHz. This combination of compact hardware and low power, together with the achieved throughput, makes the design well-suited for resource-constrained IoT devices.

11. **The paper should include a deeper comparative analysis of latency and throughput across different platforms (FPGA, CMOS, MeMOS, VTM) to complement power/area comparisons.**

We thank the reviewer for this valuable suggestion. Based on your suggestion, in the revised manuscript, we conducted a deeper comparative analysis of latency and throughput across platforms to complement the power/area comparisons. Specifically:

- Table IX (FPGA): Summarizes Piccolo-80 implementations across various FPGA families and lists Frequency (MHz), Efficiency (Mbps/Slice), Power (mW), and Latency (ns).
- Table X (CMOS/MeMOS/VTM for Piccolo-80): Offers a direct comparison between Hybrid MeMOS (protected/unprotected) and the proposed VTM design, showing Frequency, Throughput (Mbps), Power Dissipation (mW), Throughput/Area (Mbps/GE), and Latency (ns). We also examine the latency reduction and throughput improvement of VTM compared to MeMOS.
- Table XI (expands the comparison to well-known lightweight block ciphers): including ASCON (in CMOS and CMOS/MRAM), CLEFIA, PRINCE, PRESENT, LED, Camellia, SIMON, and SPECK—and reports CPD (ns) as a latency proxy alongside Throughput (Mbps), Thr./Area, technology, and operating conditions, to enable comparison of the proposed implementation/architecture with other lightweight cryptographic algorithms.