Vistrutah on FPGA: High-Throughput Pipelined Architecture and Comparison with Wider AES Variant

Ahmet Malal and Oğuz Yayla

***Abstract*—In response to the National Institute of Standards and Technology (NIST)'s 2024 call for wider variants of the Advanced Encryption Standard (AES), this paper presents the first FPGA-based hardware evaluation of Vistrutah, a recently proposed wide-block cipher constructed from AES round primitives. Vistrutah is implemented on a Xilinx Kintex UltraScale+ KCU116 FPGA and evaluated under identical conditions against the published wider Rijndael variant WAES-256. The 256-bit full configuration achieves 211.57 Gbps at 826.44 MHz, corresponding to a 2.6% throughput difference and a 2.7% frequency difference relative to WAES-256. The design also reports lower power consumption (33.2% reduction) and reduced resource usage (17.7% reduction in flip-flops and LUTs combined). These results provide an initial hardware-based comparison of two AES-compatible 256-bit block constructions and offer practical data for assessing wide-block designs on FPGA platforms.**

***Index Terms*—Vistrutah, wide-block ciphers, WAES-256, FPGA implementation, high-throughput pipelined architecture.**

I. Introduction

The Advanced Encryption Standard (AES), standardized by NIST in 2001, defines the Rijndael block cipher with a fixed 128-bit block size and has remained the dominant choice for high-performance symmetric encryption for over two decades [1]. While the 128-bit block size remains secure for many applications, emerging high-volume workloads increasingly run into fundamental limits, such as the 2^{64} birthday bound on collision resistance [2], [3] and reduced throughput when wide permutation or AEAD modes are built from 128-bit components [4].

In August 2024, NIST announced its intention to evaluate and potentially standardize a wider Rijndael variant with a 256-bit block and a 256-bit key, referred to as Rijndael-256. The call specifically solicited research on (i) the cryptographic security implications of expanding Rijndael's block size and (ii) realistic performance evaluations on hardware platforms equipped with AES acceleration [5]. This has renewed interest in wide-block ciphers that aim to balance security with practical implementation efficiency.

Recent research has explored a variety of wide-block and large-state designs. Examples include authenticated-encryption modes such as HCTR [6] and HCTR2 [7], AES-based wide permutations such as Simpira [4], and large-state tweakable block ciphers, including Pholkos [8] and ASURA [9]. These designs aim to achieve improved throughput, enhanced diffusion over large message blocks, and increased security margins against birthday-bound limitations. At the same time, prior studies have noted structural challenges when directly widening Rijndael to larger internal states, including reduced diffusion efficiency and increased exposure to key-schedule weaknesses [10], which motivate the exploration of alternative wide-block designs.

Within this context, the Vistrutah cipher [11] was proposed as an AES-compatible wide-block construction supporting 256-bit and 512-bit states. Vistrutah preserves the AES substitution-permutation structure by employing standard AES round primitives while leveraging a permutation-based inline key schedule to avoid the limitations of Rijndael's iterative expansion. This design offers compatibility with existing hardware acceleration mechanisms while providing a structured approach to processing wider states efficiently. Although its specification and software-based performance evaluations have been published; to the best of our knowledge, no hardware implementation or comparative analysis against wider Rijndael variants has been reported to date.

This paper presents the first FPGA-based implementation of the Vistrutah cipher and a controlled comparison with the wider Rijndael variant, WAES-256 [12]. Both ciphers are implemented on the same Xilinx Kintex UltraScale+ KCU116 device using identical AES primitive modules, pipeline strategies, and synthesis constraints, ensuring a fair and reproducible evaluation. The analysis examines throughput, maximum operating frequency, critical-path delay, resource utilization, and on-chip power consumption to assess the architectural efficiency and scalability of Vistrutah in a modern reconfigurable hardware environment. The key contributions of this work are summarized below:

- **First hardware implementation of Vistrutah:** This work presents the first FPGA realization of the Vistrutah cipher, implemented using validated AES primitive modules to ensure correctness and comparability with prior AES-derived designs.
- **Fair, side-by-side evaluation against WAES-256:** A controlled comparison is conducted under identical device, architectural, and synthesis conditions, enabling an objective assessment of structural differences between the two wide-block designs.
- **Comprehensive post-implementation hardware analysis:** Detailed measurements of throughput, frequency, latency, critical-path delay, power consumption, and area usage provide new insights into the practicality of Vistrutah and other AES-based wide-block constructions.
- **Demonstration of efficiency gains:** Experimental results show that Vistrutah delivers higher throughput and lower

A. Malal is with the Institute of Applied Mathematics, Middle East Technical University, 06800 Ankara, Türkiye, and also with ASELSAN Inc., Ankara, Türkiye (e-mail: ahmet.malal@metu.edu.tr).

O. Yayla is with the Institute of Applied Mathematics, Middle East Technical University, 06800 Ankara, Türkiye (e-mail: oguz@metu.edu.tr).

power and area requirements than WAES-256.

II. Overview of AES, WAES, and the Vistrutah Cipher

Wide-block cryptography has gained renewed attention due to the increasing demand for high-throughput authenticated encryption, large-state hashing, and applications that must avoid the 2^{64}-block birthday limitation of 128-bit designs. Conventional block ciphers such as AES-128 and AES-256 operate on 128-bit states, which can restrict performance when constructing wide permutations or AEAD modes such as HCTR [6] and HCTR2 [7], and when scaling parallel hashing constructions such as Simpira [4]. To overcome these constraints, research has progressed in two directions: (i) extending AES to larger block sizes through wider Rijndael variants, and (ii) designing new wide-block ciphers that preserve AES compatibility while improving scalability.

A. AES

The Advanced Encryption Standard (AES) operates on a 16-byte state organized as a 4×4 matrix. Each encryption round applies four transformations: *SubBytes* (nonlinear byte substitution), *ShiftRows* (row-wise cyclic shifts), *MixColumns* (linear mixing over GF(2^8)), and *AddRoundKey* (XOR with a round key). AES performs 10, 12, or 14 rounds depending on the key length. The key schedule expands the cipher key into round keys using byte rotation, S-box substitution, and round constants. The uniform SPN structure and byte-level parallelism enable efficient hardware pipelining and FPGA implementation.

B. Wider AES Variant (WAES)

Before standardization, Rijndael supported multiple block sizes, including 160, 192, 224, and 256 bits [1]. Although AES was standardized with a fixed 128-bit block size, interest in larger Rijndael configurations has re-emerged following NIST's 2024 call to evaluate a 256-bit block variant. The wider Rijndael variant, commonly referred to as WAES-256, preserves AES's round structure while doubling the internal state from 16 to 32 bytes. The *SubBytes*, *MixColumns*, and *AddRoundKey* transformations operate in the same manner as AES but now process eight 32-bit columns. The *ShiftRows* offsets are adjusted to maintain diffusion across the expanded state, and the key schedule applies two S-box operations per word to support the increased block size.

Prior analyzes have noted structural drawbacks when Rijndael is widened directly, including weaker diffusion across rows and potential key-schedule vulnerabilities for larger N_b values [10]. These observations motivate the study of alternative AES-compatible wide-block constructions such as Vistrutah.

C. The Large-Block Cipher Family Vistrutah

The Vistrutah cipher [11] extends the Rijndael design philosophy to 256-bit and 512-bit blocks while maintaining full compatibility with AES round primitives. Vistrutah employs the standard AES transformations (*SubBytes*, *ShiftRows*, *MixColumns*, and *AddRoundKey*) within a structural "step" composed of two AES-like rounds followed by a full-width linear mixing layer. For Vistrutah-256, the state is divided into two parallel 128-bit branches, and the mixing layer provides rapid cross-branch diffusion. The structure of Vistrutah-256 is drawn in Fig. 1.

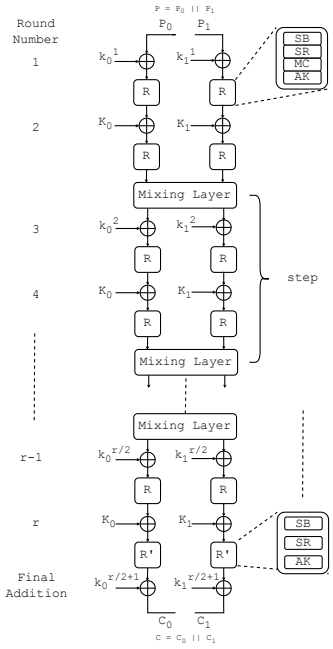

Fig. 1. The structure of Vistrutah-256 for an even number of rounds **r**, showing two parallel 128-bit branches, AES-like rounds **R**, the reduced final round **R'**, and the mixing layer applied after each pair of rounds. Round keys are supplied by the permutation-based inline key schedule.

Vistrutah supports multiple configurations that combine different block sizes, key sizes, and numbers of rounds. The 256-bit block version uses a 256-bit key and operates with 14 rounds in its full configuration, targeting a 256-bit security level. The 512-bit block variant supports either a 256-bit or a 512-bit key, with the full 512/512 configuration employing 18 rounds to meet a 512-bit security target. Reduced-round "short" variants are also defined for lightweight or high-speed contexts, such as 10 rounds for the 256-bit block version and 12 rounds for the 512-bit block version (with 256-bit keys), offering 256-bit security. By including these configurations, the authors aim to cover different block–key size options while maintaining the same two-round step structure with a linear mixing layer.

Unlike AES and WAES, which rely on iterative key expansion, Vistrutah uses a permutation-based inline key schedule. This avoids the area and latency overhead associated with traditional key scheduling and eliminates dependencies across rounds. The combination of AES-compatible round functions, a lightweight key schedule, and efficient cross-branch mixing makes Vistrutah well suited for high-throughput hardware implementations.

Although Vistrutah specifications and software performance results are available, to the best of our knowledge, no hardware implementation or comparative study against wider Rijndael

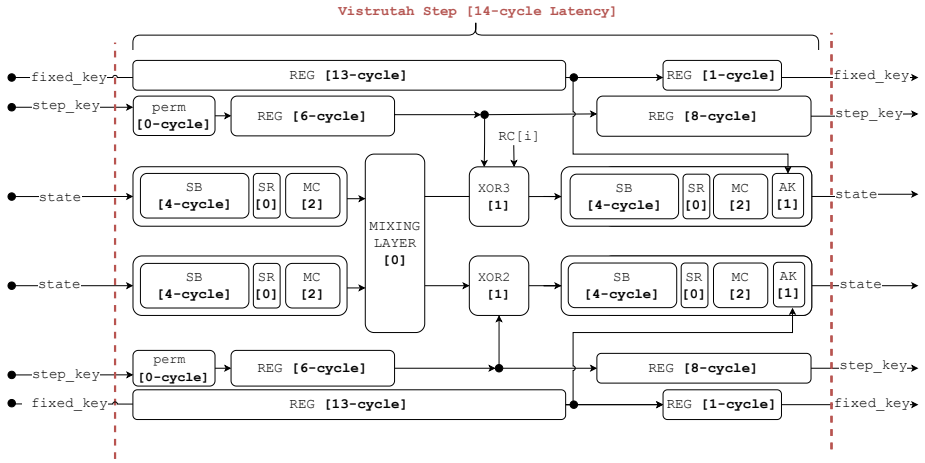

Fig. 2. Pipeline structure of a single Vistrutah step, consisting of two AES-like rounds and a zero-latency mixing layer, combined with a synchronized permutation-based inline key update. The design achieves a fixed 14-cycle latency per step and sustains one-block-per-cycle throughput. The same organization applies to both the 256-bit and 512-bit variants, where the 512-bit configuration processes four parallel 128-bit branches.

variants has been published. This motivates the hardware evaluation presented in this work.

III. Proposed High-Throughput Pipelined Architecture

The proposed architecture implements Vistrutah as a deeply pipelined design aimed at achieving one-block-per-cycle throughput on the Xilinx Kintex UltraScale+ KCU116. Because Vistrutah is constructed entirely from AES round primitives, the architecture follows a round-based organization while employing sub-pipelining to maximize operating frequency. To ensure a fair comparison with WAES-256 [12], all AES transformation modules used in our Vistrutah implementation were taken directly from [12] without any modification. Accordingly, performance differences between the two ciphers arise completely from their algorithmic structure rather than from underlying primitive implementations.

As illustrated in Fig. 2, each Vistrutah step is mapped to a 14-stage fully parallel pipeline. The step consists of two sub-pipelined AES-like rounds, followed by a zero-latency mixing layer and a synchronized permutation-based inline key update path, all implemented with pure wiring or independent functional units to sustain one-block-per-cycle throughput. Since Vistrutah supports multiple round configurations, the total latency scales linearly with the number of steps. Specifically, each Vistrutah step consists of two AES-like rounds and occupies 14 pipeline stages (7 cycles per round). Consequently, the total latency is 70, 84, 98, and 126 cycles for the 10, 12, 14, and 18-round configurations, respectively. Sub-pipelining is applied inside each AES round to reduce the critical path: the *SubBytes* transformation uses a 4-cycle composite-field S-box pipeline, *MixColumns* is implemented as a 2-cycle pipeline, and XOR operations (and *AddRoundKey*) require a single cycle. Both the step-mixing layer and the *ShiftRows* transformation are implemented as pure wiring on the FPGA fabric and thus introduce no additional latency.

The inline key schedule of Vistrutah is also fully pipelined and operates in sync with the data path. Because Vistrutah employs a permutation-based key derivation mechanism rather than an iterative AES key expansion, key updates impose effectively zero-cycle computational costs. Only pipeline registers that match the data path depth are used to maintain timing and synchronization. No BRAMs or DSP blocks are used in our design. All functional units are implemented using LUTs and flip-flops. To sustain one-block-per-cycle throughput, all parallel 128-bit branches of Vistrutah are instantiated independently, and no hardware resources are reused across pipeline stages.

This architecture enables a direct and fair comparison between Vistrutah and WAES-256, since both implementations share identical AES primitive modules, synthesis settings, pipeline scheduling methodologies, and high-throughput design philosophies. The results reflect the core structural properties of Vistrutah. This ensures that any measured performance differences come from the cipher itself rather than from implementation choices.

IV. Implementation and Results

This section presents the FPGA implementation details and performance evaluation of the proposed Vistrutah architectures on the Xilinx Kintex UltraScale+ KCU116 platform. All synthesis and implementation experiments were conducted using Vivado 2023.1 with default tool settings. A timing constraint was applied by iteratively tightening the target clock period until the `run_implementation` step completed without timing violations. Post-implementation results are reported for all metrics, including resource utilization, maximum operating frequency, and on-chip power.

Throughput is computed using the one-block-per-cycle model as Throughput = BlockSize $\times$ $f_{\max}$, consistent with the high-throughput design philosophy of both WAES-256 and Vistrutah. On-chip power consumption was estimated using Vivado's post-implementation power analyzer with default signal toggle assumptions, and all resource counts reflect final device-level mapping. To ensure a fair comparison, the HDL code published for WAES-256 [12] was re-synthesized under identical constraints. The reported results match those in the original work

TABLE I
TOTAL ON-CHIP POWER, THROUGHPUT-TO-AREA RATIO, CRITICAL DELAY, MAXIMUM FREQUENCY, THROUGHPUT, AND HARDWARE UTILIZATION RESULTS FOR KINTEX ULTRASCALE+ KCU116[†,‡].

Design	Block Size (Bits)	Key Size (Bits)	Number of Rounds	Latency (Cycle)	Resource (Slices/LUT/FF/BRAM)	Throughput (Gbps)	Max-freq. (MHz)	Critical Delay (ns)	TPS (Mbps/Slice)	Total On-Chip Power (W)
AES-128[12]	128	128	10	70	18313 FF 17241 LUT	122.95	960.6	1.04	7.13	4.084
AES-256[12]	128	256	14	98	23909 FF 24260 LUT	117.52	918.2	1.09	4.84	5.778
WAES-256[*][12]	256	256	14	126	52371 FF 45591 LUT	206.11	805.1	1.24	4.52	10.853
WAES-256 x2 core[+][12]	2×256	256	14	126	91204 FF 104529 LUT	426.66	833.33	1.20	4.68	20.815
WAES-256 x4 core[12]	4×256	256	14	126	209275 FF 181309 LUT	742.63	724.63	1.38	4.09	41.373
Vistrutah-Short	256	256	10	70	28037 FF 28943 LUT	222.608	869.56	1.15	7.69	5.214
Vistrutah-Short	512	256	10	70	45181 FF 47292 LUT	437.606	854.70	1.17	9.25	8.448
Vistrutah-Short	512	512	12	84	53128 FF 54334 LUT	437.606	854.70	1.17	8.05	9.278
Vistrutah-Full[*]	256	256	14	98	39272 FF 40828 LUT	211.57	826.44	1.21	5.18	7.252
Vistrutah-Full[+]	512	256	14	98	66146 FF 70162 LUT	430.25	840.33	1.19	6.50	12.462
Vistrutah-Full	512	512	18	126	84602 FF 87236 LUT	430.25	840.33	1.19	4.93	15.173

[*] Configurations directly comparable in block size, key size, and number of rounds between WAES-256 and Vistrutah (256-Bit, 14 rounds).
[+] Configurations used to evaluate multi-core or extended block-size scalability and parallelism between both ciphers.

TABLE II
SUMMARY COMPARISON OF VISTRUTAH-FULL (256/256/14) VS. WAES-256

Metric	WAES-256 [12]	Vistrutah-Full	Change (%)
Throughput (Gbps)	206.11	211.57	↑ 2.6
Max Frequency (MHz)	805.1	826.44	↑ 2.7
Critical Delay (ns)	1.24	1.21	↓ 2.4
TPS (Mbps/Slice)	4.52	5.18	↑ 14.6
Total Power (W)	10.853	7.252	↓ 33.2
Flip-Flops (FF)	52,371	39,272	↓ 25.0
Look-Up Tables (LUT)	45,591	40,828	↓ 10.4
Average Resource (LUT+FF) Reduction	–	–	↓ 17.7

within normal tool variation, confirming that the implementation environment reproduces prior measurements accurately.

Table I summarizes the FPGA implementation results for AES, WAES, and all Vistrutah configurations. Entries marked with an asterisk [*] indicate configurations that match in block size, key size, and round count, allowing for a direct comparison between WAES-256 and Vistrutah-256. Similarly, rows marked with a plus [+] correspond to designs used to examine scalability, where Vistrutah-512 is compared with the WAES-256 ×2 core architecture rather than the single-core version.

The short Vistrutah variants (10- and 12-round designs) use fewer pipeline stages and fewer functional units, which reduces overall resource usage and leads to slightly higher operating frequencies. The 512-bit configurations provide approximately double the throughput of the 256-bit designs because they process twice the block size at similar operating frequencies. These wider designs also require more LUTs and flip-flops since additional parallel 128-bit branches are instantiated. Latency scales linearly with the number of rounds in all configurations. Power consumption follows the amount of active logic and the achieved operating frequency.

Table I shows how block size and the number of rounds affect latency, area, throughput, frequency, and power consumption in both WAES and Vistrutah designs, enabling a direct comparison across matching configurations.

To examine configurations that directly match the NIST wider-AES proposal, a detailed comparison is provided in Table II. Table II contrasts WAES-256 with the Vistrutah-Full (256/256/14) configuration. This pairing matches block size, key size, and number of rounds and is relevant because the NIST call focuses on evaluating 256-bit Rijndael-like constructions. Both designs achieve similar maximum frequencies: 805.1 MHz for WAES-256 and 826.44 MHz for Vistrutah-Full. They share identical AES round primitives, which dominate the critical path. The small frequency difference produces a corresponding change in throughput (206.11 Gbps vs. 211.57 Gbps, a 2.6% increase).

The critical-path delays are close for both designs because they use the same AES primitive modules. Vistrutah-Full reports 1.21 ns, while WAES-256 reports 1.24 ns. In contrast, the resource usage differs more noticeably. WAES-256 uses 52,371 flip-flops and 45,591 LUTs, whereas Vistrutah-Full uses 39,272 flip-flops and 40,828 LUTs. These differences mainly come from the key-schedule structures: WAES-256 has iterative key expansion, while Vistrutah-Full uses a permutation-based inline key update that requires less logic. When averaged across LUT and flip-flop usage, the overall resource reduction is 17.7%.

All power values rely on Vivado's default toggle-rate assumptions, and actual dynamic power may vary under realistic workloads. Maximum frequency results may also show small variations across implementation seeds. Multiple runs were performed to check stability, and the relative trends between WAES-256 and Vistrutah remained consistent.

Since Vistrutah uses fewer LUTs and flip-flops, it operates with lower overall energy. This is reflected in the measured power values: WAES-256 reports 10.853 W, and Vistrutah-Full reports 7.252 W, which corresponds to a 33.2% difference.

V. CONCLUSION

This work presents an FPGA implementation of the Vistrutah cipher and a side-by-side comparison with WAES-256 on the same UltraScale+ device. The results show that both designs achieve similar operating frequencies due to their shared AES primitives, while differences in area and power primarily arise from their key-schedule structures. These findings contribute to the hardware evaluation of wide-block AES-compatible designs and offer practical data for assessing 256-bit block candidates in reconfigurable platforms.

ACKNOWLEDGMENT

The authors would like to thank the designer of the Vistrutah cipher for providing the reference source code, which helped verify the correctness of the implementation.

REFERENCES

[1] J. Daemen and V. Rijmen, *The Design of Rijndael: AES - The Advanced Encryption Standard (Information Security and Cryptography)*, 1st ed. Springer, 2002.

[2] M. Bellare and P. Rogaway, "The security of block ciphers," in *Advances in Cryptology – CRYPTO 1996*. Springer, 1996, pp. i–xv.

[3] P. Rogaway, "Notes on the birthday problem, block ciphers, and prp security," https://web.cs.ucdavis.edu/~rogaway/papers/bday.pdf, 2004.

[4] S. Gueron and N. Mouha, "Simpira v2: A Family of Efficient Permutations Using the AES Round Function," in *The 22nd Annual International Conference on the Theory and Application of Cryptology and Information Security, ASIACRYPT 2016*, ser. Lecture Notes in Computer Science, vol. 10031, Hanoi, Vietnam, Dec. 2016, pp. 95–125. [Online]. Available: https://inria.hal.science/hal-01403414

[5] NIST, "Nist proposes to standardize wider variant of aes," https://csrc.nist.gov/news/2024/nist-proposes-to-standardize-wider-variant-of-aes, 2024, accessed: 2025-02-09.

[6] P. Wang, D. Feng, and W. Wu, "Hctr: a variable-input-length enciphering mode," in *Proceedings of the First SKLOIS Conference on Information Security and Cryptology*, ser. CISC'05. Berlin, Heidelberg: Springer-Verlag, 2005, p. 175–188. [Online]. Available: https://doi.org/10.1007/11599548_15

[7] P. Crowley, N. Huckleberry, and E. Biggers, "Length-preserving encryption with HCTR2," Cryptology ePrint Archive, Paper 2021/1441, 2021. [Online]. Available: https://eprint.iacr.org/2021/1441

[8] J. Bossert, E. List, S. Lucks, and S. Schmitz, "Pholkos – efficient large-state tweakable block ciphers from the aes round function," in *Topics in Cryptology – CT-RSA 2022: Cryptographers' Track at the RSA Conference 2022, Virtual Event, March 1–2, 2022, Proceedings*. Berlin, Heidelberg: Springer-Verlag, 2022, p. 511–536. [Online]. Available: https://doi.org/10.1007/978-3-030-95312-6_21

[9] A. Tanaka, R. Shiba, K. Sakamoto, M. Rahman, T. Shiraya, and T. Isobe, "Asura: An efficient large-state tweakable block cipher for arm environment," in *Progress in Cryptology – INDOCRYPT 2024: 25th International Conference on Cryptology in India, Chennai, India, December 18–21, 2024, Proceedings, Part I*. Berlin, Heidelberg: Springer-Verlag, 2024, p. 143–164. [Online]. Available: https://doi.org/10.1007/978-3-031-80308-6_7

[10] L. May, M. Henricksen, W. Millan, G. Carter, and E. Dawson, "Strengthening the key schedule of the aes," in *Information Security and Privacy*, L. Batten and J. Seberry, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 226–240.

[11] R. Avanzi, A. Chakraborti, B. Chakraborty, and E. List, "The large block cipher vistrutah," *IACR Transactions on Symmetric Cryptology*, vol. 2025, no. 3, p. 1–150, Sep. 2025. [Online]. Available: https://tosc.iacr.org/index.php/ToSC/article/view/12467

[12] A. Malal and C. Tezcan, "First fully pipelined high throughput fpga implementation and gpu optimization of wider variant of aes," *Journal of Cryptographic Engineering*, vol. 16, no. 1, p. 1, 2026.