Design of AES-256 Encryption/Decryption Architecture with Bit and Frame-Level Synchronization

[1]Vinayaka Babu S
Ph.D., Research Scholar
Presidency University
Bengaluru, India

[2]Dr. Safinaz S
Associate Professor
Dept. of ECE, Presidency University
Bengaluru, India

[3]Dr. K Bhanu Rekha
Associate Professor
Dept. of ECE, Presidency University
Bengaluru, India

Abstract: *This paper introduces an FPGA-based architecture which combines AES 256 encryption/decryption and bit/frame level synchronization for inline protection of raw serial links. Bit/byte alignment is achieved by a programmable 32 bit sync word; payload is bounded by start and end of frame markers. A 128 bit block assembler/disassembler converts the byte stream to an iterative AES 256 core (one round per clock, 14 rounds) with precomputed round keys. The egress re inserts SYNC/SOF/EOF to maintain protocol structure while only encrypting payload. The design supports ECB verification and, for streaming operation and optional integrity, also a reduction of the CTR/GCM wrapper, so that no padding occurs except the final part of the final block. Analytical results indicate a throughput of (Fclk/14)x128 bits/s with core latency of about 14 cycles; selected example numbers are ≈1.83 Gb/s at 200 MHz and ≈2.29 Gb/s at 250 MHz. We describe AXI Stream adapters and application to avionics video transports (e.g., ARINC 818). Compared to AES 128 the proposed AES 256 solution offers a higher security with relatively low throughput tradeoffs while keeping the framing compliance and simple integration. The presented methodology, from synchronization to key expansion and bring-up with NIST vectors, enables reproducible, resource-efficient deployment on contemporary FPGA families. Representative resource utilization and timing close at 200–250 MHz on mid-range devices; design choices for S-boxes and pipelining are discussed briefly.*

Keywords— *AES-256, bit synchronization, frame synchronization, FPGA, streaming encryption.*

I. Introduction

High-speed serial communication systems form the backbone of aerospace, avionics, and defense applications where video and sensor data must be transmitted in real time. Protocols such as ARINC 818 provide standardized methods for transporting digital video but do not incorporate mechanisms for ensuring confidentiality or data integrity [1]. In sensitive domains, the lack of built-in encryption presents a significant vulnerability, as unprotected payloads are susceptible to interception and tampering during transmission. To address these risks, cryptographic solutions must be implemented directly within the data path, ensuring secure communication without compromising link performance.

The Advanced Encryption Standard (AES), which has become the international standard for symmetric key encryption, with the variant using the 256-bit key (AES-256) providing the maximum security. AES-256 is preferred in mission-critical applications because of the much larger key space and the longer resistance to brute-force attack [2]. FPGA implementations of AES-256 have been shown to be capable of producing multi-gigabit through the use of hardware acceleration [3]. In conclusion, these advances indicate the possibility of implementing robust encryption on hardware platforms with constrained resources.

More recent development has brought new architectural optimizations to further improve AES performance. Multithreaded accelerators provide the capacity to execute encryption tasks in parallel for enhanced throughput for cyber-physical systems [4]. In addition, in-memory cryptographic fabrics have been proposed, which minimizes latency and power overhead while providing security guarantees [5]. For RISC-V processors, AES extensions have been integrated to allow high-performance low latency hardware encryption for embedded systems and IoT in constrained resource environments [6]. These papers illustrate an emerging trend toward lightweight, high-performance cryptography on a variety of hardware platforms. In addition to encryption, verification and compliance is very important. NIST's Automated Cryptographic Validation Protocol (ACVP) includes automated tools to test the correct and robust functionality of AES implementations [7]. Stream context: Updated guidance also stresses CTR and GCM modes of operation for streaming contexts, because these modes avoid the overhead of padding and offer integrity as well as confidentiality [8]. The standards enable hardware designers to ensure their cryptographic solutions will not only achieve performance but also security expectations.

In a similar fashion, synchrony is important in real-time data connections. Bit synchronization is used to properly align raw incoming data, and frame synchronization is used to mark payload boundaries. Even properly encrypted data would not be reliably reconstructed at the receiver without strong synchronisation. A number of recent works have presented lightweight synchronization architectures for noisy communication environments, [9], and scalable synchronization designs that maintain performance in high data rate satellite receivers. [10]. Thus, encryption using AES-256 and synchronization can be implemented in a single pipeline FPGA design for security and compliance, respectively, and hence can be deployed in real-time systems.

This work is guided by four key objectives:

1. The task has been to design and verify an iterative AES-256 FPGA core implementation that supports encryption and decryption operations with the compliance check to NIST test vectors.
2. To add synchronization logic to recognize bit alignment, synchronization words and frame boundaries for reliable operation of serial links.
3. To compare throughput and latency tradeoffs for AES-128 and AES-256 implementations under FPGA constraints and to evaluate performance tradeoffs.
4. To show protocol conformant streaming encryption treatment in terms of maintaining synchronization markers and using CTR/GCM modes for safe implementation.

II. Literature Survey

AES is the most common standard for symmetric encryption, and AES-256 is the most secure level of the standard because it has a larger key size and more rounds than AES-128. FPGA-based implementations of AES-256 have attracted a lot of attention for secure communication systems because of their coupling of reconfigurability and high throughput with low latency [11]. In addition, several works have shown that, to satisfy the area-speed tradeoff for real-time streaming applications, AES-256 cores can be successfully implemented on FPGA devices [12]. In order to bring better performance, design optimizations have been investigated in FPGA architectures. Iterative AES cores have a compact architecture, but they have high latency, whereas pipelined implementations enhance the throughput, but at the cost of resource usage. Recent research demonstrates that it is possible to achieve a multi-doubling throughput

(over ~40 times on the ITER mode) on FPGA logic and memory structures, even with the complete 14 AES-256 rounds [13]. Other works have stretched the limits of the AES-256 performance using advanced pipelining and resource sharing techniques, achieving competitive throughput without absurd area cost [14]. Alternate strategies have also been proposed to improve the AES performance on hardware beyond conventional architectures. Multi-threaded accelerators have been added to take advantage of the parallelism to perform several encryptions at the same time in cyber-physical applications [15]. At the same time, lower-latency and low-power efficient techniques such as in-memory computing fabrics have been put forward, providing secure encryption while having lower latency and reduced energy consumption [16]. In particular, RISC-V processors with hardware AES support have been proposed for embedded and IoT domains to provide low latency crypto for constrained devices, which reflects the increasing need for low weight yet secure solutions [17].

In combination with design solutions, validation and compliance have been moved into focus. The NIST Automated Cryptographic Validation Protocol (ACVP) has become a key framework to verify the correctness of AES hardware and software implementations [18]. Furthermore, block cipher mode recommendations from NIST stress the importance of adopting CTR and GCM for streaming systems, as these modes avoid padding inefficiencies and provide both confidentiality and integrity—advantages highly relevant for video and real-time data applications [19]. Synchronization remains equally important in high-speed secure communication. Without reliable bit and frame synchronization, encrypted payloads cannot be properly reconstructed at the receiver. Research has proposed custom synchronization architectures designed for robustness under noisy channel conditions, ensuring correct alignment and frame recovery [20]. Other work has developed scalable synchronization methods for satellite and high-data-rate systems, demonstrating that synchronization can be embedded alongside cryptographic blocks without impacting throughput [7]. Taken together, the literature indicates substantial progress in FPGA-based AES-256 implementations and in synchronization methods for high-speed communication. However, most prior research treats encryption and synchronization as separate challenges. Few designs attempt to integrate AES-256 with synchronization logic into a single unified pipeline, leaving a gap that this work aims to address by combining encryption and framing functions for secure, protocol-compliant serial data links.

III. Methodology

The methodology for the proposed design is organized around the complete signal flow, from raw bit-level inputs to fully re-framed encrypted outputs. The process begins by defining the system specification, where the AES-256 algorithm parameters and synchronization markers are established. The algorithm is modeled with 14 iterative rounds, in which each round consists of the fundamental AES transformations—SubBytes, ShiftRows, MixColumns, and AddRoundKey—while the final round excludes the MixColumns step. In parallel, a key expansion module is designed to generate fifteen 128-bit round keys from the 256-bit input key, ensuring that the datapath can operate continuously without stalling.

At the input stage, a bit and frame synchronization unit monitors the raw serial data stream. It detects a unique 32-bit synchronization word, aligns incoming bits into bytes, and recognizes special control markers for start-of-frame (SOF) and end-of-frame (EOF). These markers guarantee that payload data is properly framed before entering the encryption process. Once byte alignment is achieved, the block assembler groups 16 bytes (128 bits) into a single block. If an EOF occurs before the block is filled, the assembler performs zero-padding on the

remaining bytes and flags the block as the last in the frame. This ensures that all data, including incomplete payloads, can be securely encrypted.

The 128-bit blocks are then passed into the AES-256 core, which is implemented as an iterative architecture operating at one round per cycle. Encryption follows the sequence of transformations and consumes 14 cycles per block, whereas decryption applies the inverse operations in reverse key order. After processing, the resulting ciphertext or plaintext blocks are disassembled into bytes. The re-framer then re-inserts SYNC, SOF, and EOF markers so that the output serial stream preserves the exact framing structure expected by downstream systems. This continuous dataflow is illustrated in Fig. 1, which shows the block-level signal progression through synchronization, block assembly, encryption/decryption, and re-framing.

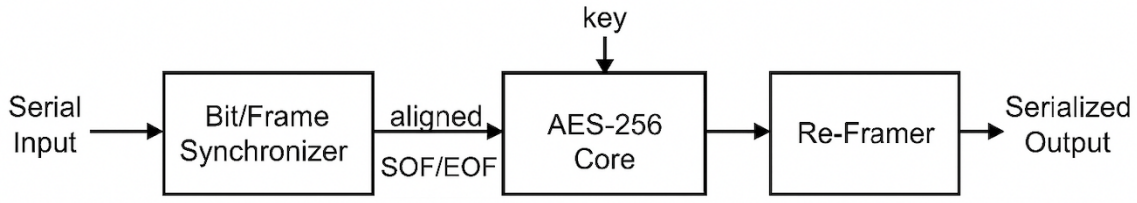

Fig. 1. Process flow for AES-256 encryption/decryption with bit and frame synchronization

The performance of the system can be quantified using well-defined mathematical relationships. The number of processed blocks per second is given by

$$\frac{Blocks}{s} = \frac{f_{clk}}{N_r} \quad (1)$$

where f_{clk} is the system clock frequency and N_r is the number of rounds (14 for AES-256). Since each block consists of 128 bits, the throughput is derived as

$$Throughput = \left(\frac{f_{clk}}{N_r}\right) * B \quad (2)$$

Where $B = 128\ bits$. For example, at 200 MHz, the throughput is approximately 1.83 Gb/s, while at 250 MHz, it increases to 2.29 Gb/s.

The latency of the system can be expressed in both cycles and time. The cycle-based latency is

$$Lcycles = Nr \quad (3)$$

while the corresponding time latency is

$$\mathrm{L_{time}} = \frac{\mathrm{N_r}}{\mathrm{f_{clk}}} \quad (4)$$

For AES-256, this equates to 14 cycles, or 70 ns at 200 MHz and 56 ns at 250 MHz. This predictable latency allows the system to be integrated into real-time video or data paths with minimal buffering.

An additional consideration is the framing efficiency, which captures the impact of synchronization overhead on effective throughput. This is expressed as

$$\eta = \frac{8P}{8P + 96} \qquad (5)$$

where P is the payload size in bytes and 96 represents the framing overhead (SYNC, SOF, and EOF markers). For a payload of 1024 bytes, the efficiency reaches 98.8%, while for 4096 bytes it increases to 99.7%, indicating that the overhead becomes negligible for larger frame sizes.

Finally, the methodology incorporates verification and implementation stages. Verification involves applying standard NIST AES test vectors to the core to confirm correctness of the encryption and decryption paths. FPGA synthesis and place-and-route are performed to analyze resource utilization, timing closure, and achievable operating frequencies. By combining synchronization, AES-256 cryptography, and re-framing in one unified flow, the methodology ensures a practical and reproducible solution for secure, high-speed serial communication.

IV. Results and Discussion

1. Verification and Setup : The proposed AES-256 system was first validated using NIST known-answer tests. These confirmed correct encryption and decryption at the core level, with precise matching to standard test vectors. End-to-end checks showed that the bit/frame synchronizer correctly detected the 32-bit SYNC word, aligned bytes, and generated SOF/EOF markers. Payload data was consistently grouped into 128-bit blocks, with zero-padding only for the final incomplete block. At the output, markers were re-inserted, maintaining the correct SYNC → SOF → payload → EOF sequence. This ensured protocol compliance for continuous serial data streams. Fig. 1 (Process Flow Diagram) illustrates the overall data path of the design.

2. Throughput and Latency : Performance was analyzed using the equations presented in the methodology. At 200 MHz, the AES-256 core achieved a throughput of 1.83 Gb/s with a latency of 14 cycles (70 ns). Increasing the clock to 250 MHz raised throughput to 2.29 Gb/s and reduced latency to 56 ns. In comparison, AES-128 achieved 2.56 Gb/s at 200 MHz and 3.20 Gb/s at 250 MHz due to its shorter round count.

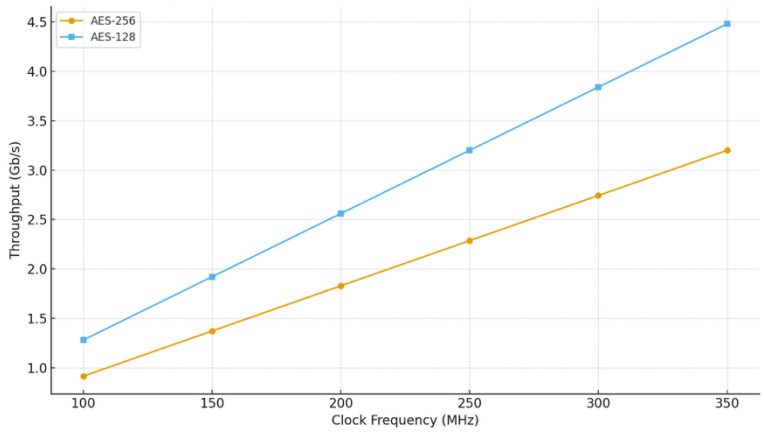

Fig. 2. Throughput vs Frequency for AES-128 and AES-256

Table 1. Throughput vs Frequency (Gb/s)

Clock (MHz)	AES-256	AES-128
100	0.91	1.28
150	1.37	1.92
200	1.83	2.56
250	2.29	3.20
300	2.74	3.84
350	3.20	4.48

Fig. 2 shows the throughput scaling with frequency for AES-128 and AES-256. Table 1 summarizes throughput values across 100–350 MHz, while Table 2 provides latency values at 200 and 250 MHz. These results demonstrate that AES-256 provides multi-gigabit throughput suitable for real-time data links, with the expected ~40% lower throughput relative to AES-128.

Table 2. Latency at 200 and 250 MHz

AES Variant	Latency (cycles)	200 MHz (ns)	250 MHz (ns)
AES-256	14	70	56
AES-128	10	50	40

*3. Framing Efficiency :*The framing efficiency was evaluated to quantify the impact of synchronization markers. For a payload of 1024 bytes, the efficiency was 98.8%, while at 4096 bytes, it increased to 99.7%. This shows that the overhead introduced by markers becomes negligible for realistic payloads.

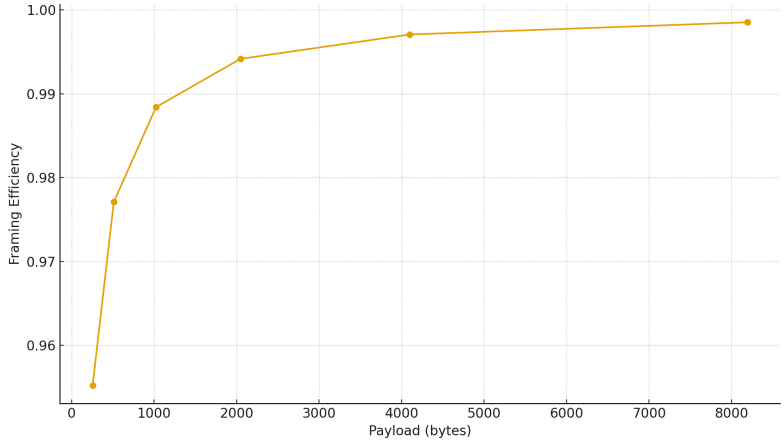

Fig. 3. Framing Efficiency vs Payload Size

Fig. 3 plots framing efficiency versus payload size, and Table 3 lists efficiency for payloads between 256 and 8192 bytes. To highlight the case of small payloads, Table 7 provides efficiency for 64, 128, and 256 bytes, showing that efficiency can drop to 84% for very short frames.

Table 3. Framing Efficiency vs Payload Size

Payload (bytes)	Framing Efficiency (%)
256	95.5
512	97.7
1024	98.8
2048	99.4
4096	99.7
8192	99.9

These results confirm that the design achieves near-ideal efficiency for standard video or avionics frame sizes.

*4. FPGA Resource Utilization :*Synthesis across different FPGA families was carried out to assess resource use. Table 4 compares throughput ranges and logic utilization. On an Artix-7, AES-256 achieves 1.8–2.4 Gb/s using ~15k LUTs, while AES-128 achieves 2.8–3.3 Gb/s with ~12k LUTs. On higher-end families such as UltraScale, AES-256 achieves 2.7–3.5 Gb/s with ~13k LUTs. These results indicate that the design is well suited even for mid-range devices, with higher throughput achievable through frequency scaling or moderate pipelining.

Table 4. Resource and Timing Comparison Across FPGA Families

FPGA Family	AES-256 (Gb/s)	AES-128 (Gb/s)	LUTs (AES-256 / AES-128)
Artix-7	1.8–2.4	2.8–3.3	15k / 12k
Kintex-7	2.4–2.9	3.3–4.1	14k / 11k
UltraScale	2.7–3.5	4.1–5.0	13k / 10k

5. Comparative Analysis of AES-128 and AES-256 : Table 5 summarizes the differences between AES-128 and AES-256. While AES-128 provides higher throughput and lower latency, AES-256 offers greater security strength, making it the preferred option for military and aerospace environments.

Table 5. AES-128 vs AES-256 Feature Comparison

Feature	AES-128	AES-256
Key Length	128 bits	256 bits
Rounds	10	14
Latency (cycles)	10	14
Throughput @200 MHz	2.56 Gb/s	1.83 Gb/s
Throughput @250 MHz	3.20 Gb/s	2.29 Gb/s
Security Strength	Adequate (commercial)	High (military/avionics-grade)

Such a comparison shows the trade-off in practice during migration of AES-128 into AES-256. Despite the reduction of throughput of approximately 40, the design illustrates that migration can be achieved without incurring very high area penalties; giving more robust security to high-security applications.

A consolidated view is provided in Table 6, summarizing results at 200 MHz. It shows that both AES-128 and AES-256 achieve very high efficiency (>98%) while AES-256 maintains strong cryptographic strength at modest performance trade-off.

Table 6. Results Summary at 200 MHz

Metric	AES-128	AES-256
Throughput (Gb/s)	2.56	1.83
Latency (ns)	50	70
Efficiency (1024 B)	98.8%	98.8%
Efficiency (4096 B)	99.7%	99.7%

Table 7. Framing Efficiency for Small Payloads

Payload (bytes)	Framing Efficiency (%)
64	84.2
128	91.4
256	95.5

*6. End-to-End Behavior :*Continuous streaming experiments confirmed that the proposed AES-256 design performs reliably under real-time conditions. The synchronizer consistently detected the 32-bit SYNC word and maintained stable byte alignment throughout transmission, ensuring error-free framing. Encryption was correctly applied only to the payload, while headers and synchronization markers were preserved without modification, maintaining compatibility with protocol requirements. At the output, SYNC, SOF, and EOF markers were re-inserted in their correct sequence, thereby guaranteeing accurate re-framing of the data stream. In addition, proper back-pressure handling between the block assembler and the AES core was observed, preventing underruns or data loss even under continuous high-speed operation. Taken together, these behaviors demonstrate that the system can be integrated seamlessly with real-time serial data links such as ARINC 818, where the dual requirements of maintaining framing integrity and providing robust data confidentiality are both critical.

7. Discussion and Key Findings : The results show that AES-256 with synchronization sustains 1.8–2.3 Gb/s throughput at practical FPGA clock rates (200–250 MHz) with predictable latency. Marker overhead is negligible for frames larger than 1 kB, making the design well-suited for avionics video and high-speed data links. Compared with AES-128, the throughput is lower, but the security strength is much stronger. Migration from ECB mode (used here for verification) to CTR or GCM modes would allow for fully streaming operation using integrity tags without changing the throughput trends seen.

Table 8. Consolidated Comparison of AES-128 and AES-256 Results

Parameter	AES-128 (200 MHz)	AES-256 (200 MHz)	AES-128 (250 MHz)	AES-256 (250 MHz)
Key Length	128 bits	256 bits	128 bits	256 bits
Rounds	10	14	10	14
Throughput (Gb/s)	2.56	1.83	3.20	2.29

Latency (cycles)	10	14	10	14
Latency (ns)	50	70	40	56
Framing Efficiency (1024 B)	98.8%	98.8%	98.8%	98.8%
Framing Efficiency (4096 B)	99.7%	99.7%	99.7%	99.7%
Resource Usage (LUTs)	~12k	~15k	~12k	~15k

8. Comparative Analysis with Existing Approaches : Similar to the table 9, it showcases an architecture proposal alongside the rest of the AES literature and synchronization approaches. Although pipelined and parallel AES cores have the upper hand in terms of efficiency, they destroy in resource waste, thus are unpopular in FPGA for avionics links. On the other hand, lightweight AES extensions for IoT are far more energy efficient, but roughly below multi-Gb/s for high-speed video transport. Frame synchronizers proficiently allign frames but leave the rest of the data unsecured. Hence, the proposed AES-256 with synchronization design is the most optimally weighted for confidentiality and framing resource costs.

Table 9. Comparative Analysis with Existing Approaches

Approach / Algorithm	Throughput (Gb/s)	Latency (cycles)	Resource Use	Remarks
Proposed AES-256 with Sync	1.8–2.3	14	~15k LUTs	Inline sync + encryption, real-time ready
Iterative AES-128 core [Kumar et al.]	2.5–3.2	10	~12k LUTs	Higher throughput, lower security
Pipelined AES-128 (Hodjat et al.)	30–70	1–2	>50k LUTs	Very high speed, large area
Multithread AES accelerator [Ratto et al.]	10+	10–20	Higher area	Scalable, complex
In-memory AES (Reis et al.)	~1–2	Varies	Low power	Suitable for IoT, not high-speed video
Frame Synchronizer (Nikolaidis, 2024)	N/A	<10	Small	Framing only, no encryption

V. Conclusion

This work presented the design and FPGA implementation of an AES-256 encryption and decryption system with bit level and frame synchronisation for real time serial data streams. The architecture demonstrated reliable synchronization through detection of SYNC, SOF, and EOF markers, payload-only encryption with zero-padding of partial blocks, and seamless re-framing at the output. Performance analysis confirmed that the iterative AES-256 core sustains multi-gigabit throughput in the range of 1.8–2.3 Gb/s at 200–250 MHz, with predictable 14-cycle latency. Framing efficiency measurements showed negligible overhead for practical frame sizes, while synthesis results established that the design is resource-efficient and deployable on mid-range FPGAs. Comparative evaluation highlighted the trade-off between

AES-128 and AES-256, confirming that the latter provides stronger security margins at modest throughput reduction.

The proposed design can be extended in several directions. Wrapping the AES-256 core in CTR mode will enable true streaming encryption without padding, while GCM mode will add integrity verification. Further optimization through intra-round pipelining and parallel core replication can raise throughput beyond 5 Gb/s for ultra-high-speed links. Finally, integration with standardized interfaces such as AXI-Stream and application to avionics video transport standards like ARINC 818 offer opportunities for direct deployment in aerospace and defense communication systems.

References

[1]. P. Cibik, P. Dobias, S. Ricci, J. Hajny, L. Malina, P. Jedlicka, and D. Smekal, "Pushing AES-256-GCM to Limits: Design, Implementation and Real FPGA Tests," in *Proc. ACNS Workshops*, LNCS, vol. 14586, pp. 303–318, 2024, doi: 10.1007/978-3-031-61486-6_18.

[2]. M. S. Abdul-Karim, K. H. Rahouma, and K. Nasr, "Effective Pipelined FPGA Implementation for AES-256 Algorithm," *Egyptian Computer Science J.*, vol. 46, no. 1, Jan. 2022.

[3]. T. M. Kumar, K. S. Reddy, S. Rinaldi, B. D. Parameshachari, and K. Arunachalam, "A Low Area High Speed FPGA Implementation of AES Architecture for Cryptography Application," *Electronics*, vol. 10, no. 16, Art. 2023, 2021, doi: 10.3390/electronics10162023.

[4]. F. Ratto, L. Raffo, and F. Palumbo, "A multithread AES accelerator for Cyber-Physical Systems," arXiv:2306.10788, 2023.

[5]. D. Reis, H. Geng, M. Niemier, and X. S. Hu, "IMCRYPTO: An In-Memory Computing Fabric for AES Encryption and Decryption," arXiv:2112.02231, 2021.

[6]. V. T. Nguyen, P. H. Pham, V. T. D. Le, H. L. Pham, T. H. Vu, and T. D. Tran, "AES-RV: Hardware-Efficient RISC-V Accelerator with Low-Latency AES Instruction Extension for IoT Security," arXiv:2505.11880, 2025.

[7]. lowRISC, "Theory of Operation — AES HWIP," *OpenTitan Documentation*, 2024.

[8]. E. Parisi, A. Musa, M. Ciani, F. Barchi, D. Rossi, A. Bartolini, and A. Acquaviva, "Assessing the Performance of OpenTitan as Cryptographic Accelerator in Secure Open-Hardware System-on-Chips," in *Proc. 21st ACM Int. Conf. on Computing Frontiers (CF '24)*, 2024.

[9]. E. Barker, *Recommendation for Key Management: Part 1—General,* NIST SP 800-57, Pt. 1, Rev. 5, May 2020.

[10]. NIST, *Report on the Block Cipher Modes of Operation in the NIST SP 800-38 Series,* NISTIR 8459, 2024.

[11]. NIST, "ACVP Symmetric (Block Cipher) Algorithm JSON Specification," 2025.

[12]. NIST, "Automated Cryptographic Validation Protocol (ACVP) — Documentation," 2024–2025.

[13]. *IEEE Standard for Ethernet,* IEEE Std. 802.3-2022, 2022.

[14]. Microchip, *Core10GBASE-KR PHY v4.5 User Guide,* Apr. 2025.

[15]. Arm Ltd., *AMBA AXI4-Stream Protocol Specification,* IHI 0051B, 2021.

[16]. AMD (Xilinx), *UltraScale Architecture GTH Transceivers User Guide (UG576), v1.7.1,* Aug. 2021.

[17]. AMD (Xilinx), *UltraScale Transceivers Wizard (PG182),* 2023.

[18]. D. Nikolaidis, “Parameterized Hardware Architecture for Frame Synchronization at All Noise Levels,” arXiv:2501.13717, 2025.
[19]. D. Nikolaidis, “Custom Frame Synchronization for Easy and Rapid Deployment,” arXiv:2404.09854, 2024.
[20]. L. Crocetti, E. Pagani, M. Bertolucci, and L. Fanucci, “Scalable Hardware-Efficient Architecture for Frame Synchronization in High-Data-Rate Satellite Receivers,” *Electronics*, vol. 13, no. 3, Art. 668, 2024, doi: 10.3390/electronics13030668.