Mühendislik Bilimleri ve Tasarım Dergisi
14(2), 182-194, 2026
e-ISSN: 1308-6693

Journal of Engineering Sciences and Design
DOI: 10.21923/jesd.1818679

Araştırma Makalesi Research Article

PIPELINED AES-CTR FOR UNDERWATER RGB IMAGE ENCRYPTION/DECRYPTION ON FPGA

Ahmet Ekrem AKEL[1], Ahmet Taha AYDIN[1], Berke AKGÜL[1*], Aslıhan KARTCI[1]

[1]Yıldız Technical University, Faculty of Electrical and Electronics, Department of Electronics and Communication Engineering, İstanbul, Türkiye

Keywords

AES, CTR Mode, FPGA, Image Encryption, Pipeline, Underwater Imaging, Verilog.

Abstract

This paper presents a pipelined FPGA implementation of the AES algorithm in the Counter (CTR) mode, specifically optimized for underwater RGB image encryption and decryption. Underwater imaging often suffers from limited bandwidth, so it requires secure transmission in real-time applications. These applications include, but are not limited to, autonomous underwater vehicles (AUVs) and remotely operated vehicles (ROVs). The proposed design achieves high throughput while maintaining low latency. By utilizing a multistage pipeline architecture implemented in Verilog, four pipeline configurations were developed and evaluated, ranging from a non-pipelined baseline to a fully pipelined design. The non-pipelined implementation achieves 1.16 Gb/s with 3106 LUTs and 2193 FFs. Moreover, the 2-stage and 6-stage versions reach 2.56 Gb/s and 6.40 Gb/s, respectively, while the fully pipelined implementation attains 12.80 Gb/s at a clock frequency of 100 MHz, utilizing 6403 LUTs and 2943 FFs. These results demonstrate the efficiency of deep pipelining and provide a favourable balance between performance and resource utilization compared to prior AES implementations.

FPGA ÜZERİNDE SU ALTI RGB GÖRÜNTÜ ŞİFRELEME/ÇÖZME İÇİN BORUHATLI (PIPELINED) AES-CTR

Anahtar Kelimeler

AES, CTR Modu, FPGA, Görüntü Şifreleme, Boru Hattı, Su Altı Görüntüleme, Verilog.

Özet

Bu çalışma, su altı RGB görüntü şifreleme ve çözme uygulamaları için optimize edilmiş, Sayıcı (CTR) modunda AES algoritmasının boru hatlı FPGA gerçekleştirimini sunmaktadır. Su altı görüntüleme sistemleri sınırlı bant genişliği altında çalıştığından, özellikle otonom su altı araçları (AUV) ve uzaktan kumandalı araçlar (ROV) gibi platformlarda güvenli ve gerçek zamanlı veri iletimi gerektirmektedir. Önerilen mimari, düşük gecikme ile verim arasında dengeli bir yapı hedefleyen çok aşamalı boru hatlı Verilog tasarımına dayanmaktadır. Boru hattı olmayan temel yapıdan tam boru hatlı mimariye kadar dört farklı boru hattı yapılandırması geliştirilmiş ve performans açısından değerlendirilmiştir. Boru hattı olmayan tasarım 3106 LUT ve 2193 FF kullanımı ile 1,16 Gb/s işlem hızına ulaşmaktadır. İki aşamalı ve altı aşamalı yapılandırmalar sırasıyla 2,56 Gb/s ve 6,40 Gb/s hız üretmektedir. Tam boru hatlı mimari ise 100 MHz çalışma frekansında 6403 LUT ve 2943 FF kullanımı ile 12,80 Gb/s veri işleme kapasitesi göstermektedir. Elde edilen bulgular, derin boru hattı mimarisinin verimi artırdığını ve kaynak kullanım etkinliğini koruduğunu ortaya koymaktadır. Önceki FPGA üzerindeki AES uygulamaları ile karşılaştırıldığında, önerilen tasarım bant genişliği ve enerji kısıtlarının belirleyici olduğu su altı sistemleri için uygun bir performans–kaynak dengesi sunmaktadır.

Alıntı / Cite

Akel, A. E., Aydın, A. T., Akgül, B., Kartcı, A., (2026). Pipelined AES-CTR for Underwater RGB Image Encryption/Decryption on FPGA, Journal of Engineering Sciences and Design, 14(2), 182-194.

Yazar Kimliği / Author ID (ORCID Number)
A. E. Akel, 0009-0009-6591-4018
A. T. Aydın, 0000-0000-2984-0386
B. Akgül, 0000-0001-6621-8537
A. Kartcı, 0000-0001-5690-7574

Makale Süreci / Article Process	
Başvuru Tarihi / Submission Date	06.11.2025
Revizyon Tarihi / Revision Date	26.02.2026
Kabul Tarihi / Accepted Date	24.04.2026
Yayım Tarihi / Published Date	30.06.2026

[*] İlgili yazar / Corresponding author: berke.akgul@yildiz.edu.tr, +90-538-042-4746

PIPELINED AES-CTR FOR UNDERWATER RGB IMAGE ENCRYPTION/DECRYPTION ON FPGA

Ahmet Ekrem AKEL[1], Ahmet Taha AYDIN[1], Berke AKGÜL[1†], Aslıhan KARTCI [1]
[1]Yıldız Technical University, Faculty of Electrical and Electronics, Department of Electronics and Communication Engineering, İstanbul, Türkiye

Highlights

- FPGA-based AES-128/CTR design optimized for underwater RGB image security.
- Multi-stage pipeline enables high throughput with low latency in encryption tasks.
- Resource-efficient design balances performance and hardware utilization.
- Applicable to real-time AUV and ROV secure communications.

Purpose and Scope
The purpose of this paper is to present an FPGA-based AES-128/CTR implementation. The design is specifically optimized for underwater RGB image encryption and decryption. The research aims to address the need for secure, real-time communication in underwater environments.

Design/methodology/approach
The proposed design employs a multistage pipelined architecture coded in Verilog. Four pipeline configurations were implemented and tested on an FPGA device. The design ranges from a non-pipelined baseline to a fully pipelined design. Performance metrics such as throughput and resource usage were analysed.

Findings
The results demonstrate that deeper pipelining dramatically improves throughput while maintaining minimal hardware utilization. The fully pipelined implementation achieved 12.8 Gb/s, providing a favourable trade-off between speed and resource consumption compared to traditional AES FPGA implementations.

Research limitations/implications
This study is limited to AES-128 in CTR mode on a specific FPGA platform. Future research could extend the work to AES-192/256, other operation modes, or alternative FPGA families. Exploring power optimization and integration with higher-level underwater communication systems is also suggested.

Practical implications
The findings enable the development of resource-efficient, real-time encryption systems. The design is suitable for autonomous underwater vehicles (AUVs) and remotely operated vehicles (ROVs). The design demonstrates how secure data transmission can be achieved under strict performance and power constraints.

Social Implications
This work supports advances in marine exploration, environmental monitoring, and defense applications. It contributes to safer and more efficient use of underwater robotics in both research and industry.

Originality
The originality of this work lies in its pipeline-optimized AES-128/CTR FPGA implementation for underwater imaging applications. Unlike prior works, this design achieves high throughput with balanced hardware usage, offering practical value to both academic research and industrial practice.

† Corresponding author: berke.akgul@yildiz.edu.tr, +90-538-042-4746

1. Introduction

In underwater communication systems used by autonomous underwater vehicles (AUVs) and remotely operated vehicles (ROVs), secure and real-time data transmission is essential for operational safety, mission success, and data integrity. Applications such as naval operations, marine ecosystem monitoring, and underwater resource exploration demand robust protection against security breaches, which may result in mission failure, financial loss, or strategic risk (González-García et al., 2020).

However, securing underwater communications is inherently challenging due to the physical characteristics of the acoustic channel. Limited bandwidth, high latency, multipath propagation, and elevated bit error rates significantly constrain communication efficiency and reliability (Busacca et al., 2024). These constraints directly shape protocol design and restrict achievable performance. As a result, security mechanisms originally developed for terrestrial RF systems introduce substantial computational and communication overhead in underwater environments. This leads to a critical trade-off between system performance (throughput, latency, energy consumption) and security requirements (confidentiality, integrity). Moreover, the openness of acoustic and wireless links increases vulnerability to eavesdropping and man-in-the-middle attacks, while high latency complicates conventional authentication procedures (Jiang et al., 2018).

To address these constraints, prior research has pursued two main approaches. The first focuses on algorithmic optimization through lightweight cryptographic primitives tailored for resource-constrained environments. Examples include chaos-based encryption schemes that reduce computational complexity by avoiding conventional S-box operations (Peng et al., 2016; Goyal et al., 2022). While such methods improve efficiency, they may not provide the same level of security assurance as standardized and extensively cryptanalyzed algorithms.
The second approach employs established algorithms such as the Advanced Encryption Standard (AES). However, software-based AES implementations on embedded processors often deliver insufficient throughput for real-time applications and incur considerable energy consumption. To overcome these limitations, hardware acceleration using FPGAs offers a practical compromise, enabling high-throughput and low-latency implementation of standardized cryptographic primitives suitable for real-time underwater systems.

Prior studies, including the FPGA-based chaos-driven image encryption by Maazouz et al. (Maazouz et al., 2022) and the hybrid AES–chaos architectures explored by Yang et al. (Yang & Chien, 2020), have demonstrated the performance benefits achievable through hardware acceleration. However, most existing implementations either focus on hybrid or non-standard cryptographic schemes or target high-end FPGA platforms without systematically analyzing pipeline scalability and normalized efficiency on cost-effective mid-range devices.
In contrast, this work does not propose a new cryptographic primitive; rather, it delivers a hardware-centric optimization of standardized AES-128 in CTR mode specifically tailored for FPGA platforms.

The proposed CTR-specialized core integrates nonce/counter management directly into the datapath and employs a precomputed key schedule to minimize latency and maximize pipeline efficiency. To fully exploit the inherent parallelism of CTR mode, four pipeline configurations are systematically implemented and evaluated: non-pipelined, 2-stage, 6-stage, and fully pipelined. These configurations enable detailed analysis of latency–area–throughput trade-offs. The results demonstrate a clear progression in performance: 1.16 Gb/s (non-pipelined), 2.56 Gb/s (2-stage), 6.40 Gb/s (6-stage), and up to 12.80 Gb/s (fully pipelined) on a mid-range Artix-7 FPGA, with resource utilization remaining below 6500 LUTs and 3000 FFs.

When compared with earlier FPGA-based AES-CTR implementations such as Qu and Shou's 73.73 Gb/s design on a Virtex-5 LX85 (Qu et al., 2009), Fu and Hao's 27.10 Gb/s implementation on a Virtex-2 (Fu et al., 2005), and Guzmán et al.'s 34.89 Gb/s design on a Virtex-5 LX110T (Guzmán et al., 2016), the proposed architecture achieves competitive throughput per resource while targeting a significantly smaller and more cost-effective Artix-7 device. This balance of scalability, efficiency, and resource economy highlights the suitability of the design for real-time encrypted multimedia transmission in resource-constrained underwater AUV and ROV systems.

The main contributions of this work can be summarized as follows:

1. A datapath-integrated nonce/counter architecture eliminating external feedback latency in CTR mode.
2. A precomputed key schedule enabling sustained pipeline throughput.
3. A structured evaluation of four pipeline configurations to quantify latency–throughput–resource trade-offs.
4. An implementation achieving up to 12.80 Gb/s at 100 MHz on an Artix-7 device with fewer than 6500 LUTs and 3000 FFs.

The remainder of this paper is organized as follows. Section II provides the theoretical background of the AES algorithm, its modes of operation, and pipeline architectures. Section III describes the proposed AES-128 implementation under three pipeline configurations. Section IV presents the FPGA implementation and validation through MATLAB-based testing scenarios, with performance evaluated in terms of throughput and cryptographic robustness. Finally, Section V concludes the paper and outlines future research directions.

2. Theoretical Background

The Advanced Encryption Standard (AES) is a widely adopted symmetric block cipher standardized by NIST in 2001 as a replacement for DES and Triple DES (NIST, 2023). Due to its strong security guarantees and implementation efficiency, AES has become the dominant cryptographic primitive in hardware and software systems.

2.1. The AES Algorithm

AES operates on 128-bit data blocks and supports key lengths of 128, 192, and 256 bits. In this study, the AES-128 variant is employed, consisting of 10 encryption rounds. Each round applies four core transformations of SubBytes, ShiftRows, MixColumns, and AddRoundKey preceded by an initial key addition. These transformations provide nonlinear substitution, diffusion, and key-dependent mixing, forming the basis for secure and efficient hardware implementation.

SubBytes: Each byte of the state is substituted with a new value using a nonlinear substitution table (S-box). This operation introduces significant nonlinearity and strengthens resistance against both differential and linear cryptanalysis in Figure 1.

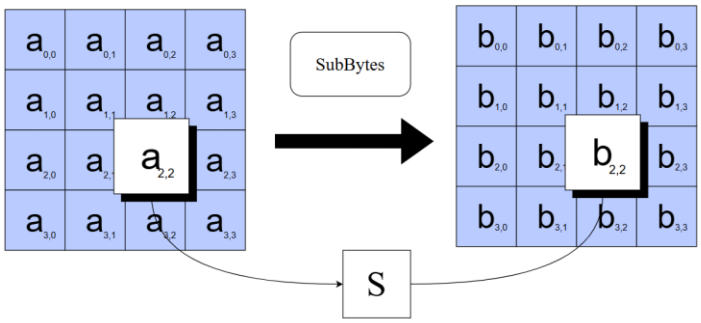


Figure 1. Illustration of the SubBytes transformation

ShiftRows: The AES state, represented as a 4×4 byte matrix, is subjected to cyclic row shifts. The first row remains unchanged, while the second, third, and fourth rows are shifted to the left by one-, two-, and three-byte positions, respectively. This transformation enhances inter-row diffusion by rearranging the byte positions in Figure 2.

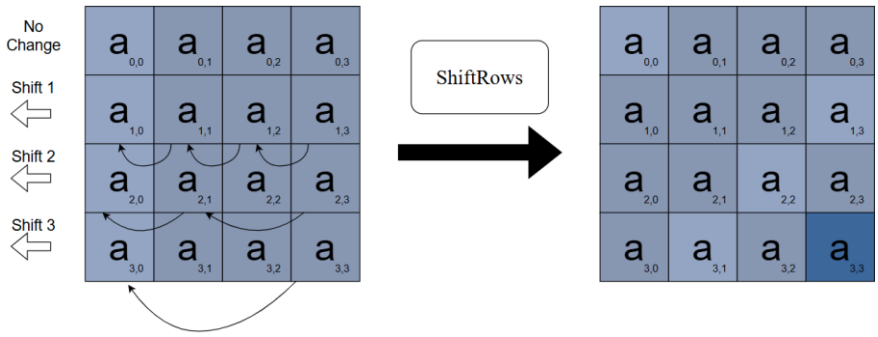


Figure 2. Illustration of the ShiftRows transformation

MixColumns: Each column of the state matrix is multiplied by a fixed polynomial matrix defined over the Galois Field GF(2^8). This operation spreads the influence of each byte throughout the entire column, thereby providing strong diffusion in Figure 3.

AddRoundKey: At the end of each round, the current state is XORed with the corresponding round key. This directly integrates the cipher key into the encryption process, tightly binding security to the strength of the key schedule in Figure 4. Since AES relies on hardware-friendly arithmetic operations such as XOR, table lookups, and

finite-field multiplications, it can achieve high throughput on FPGA platforms with moderate resource utilization. These characteristics make AES particularly well-suited for secure real-time data transmission in hardware-based implementations.

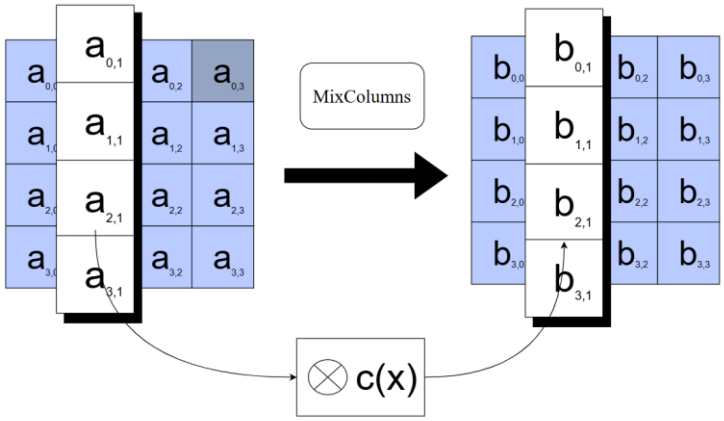


Figure 3. Illustration of the MixColumns transformation

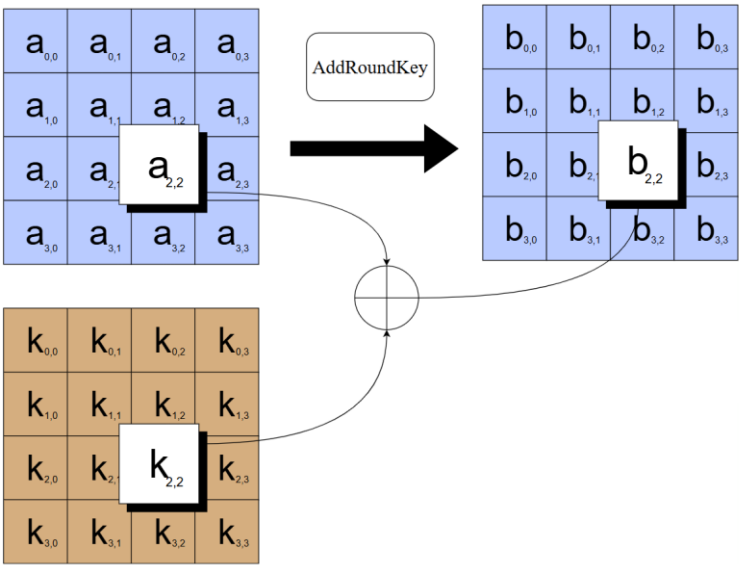


Figure 4. Illustration of the AddRoundKey transformation

2.2. Counter (CTR) Mode Operation

AES operates on fixed-size 128-bit data blocks. To encrypt data streams of arbitrary length, it is used in conjunction with a mode of operation, which directly affects confidentiality, throughput, and parallelization capability particularly for multimedia data such as video, audio, and images (Blazhevski et al., 2013).

Among the standardized modes, Electronic Codebook (ECB) and Cipher Block Chaining (CBC) are commonly referenced but suffer from structural or sequential dependencies that limit parallel performance. In contrast, Counter (CTR) mode enables full block-level parallelism and eliminates inter-block feedback, making it well suited for high-throughput FPGA implementations. For this reason, CTR mode is adopted in this work. To understand the concept of CTR mode, the ECB and CBC are also investigated in the below.

2.3. Modes of Operation: ECB, CBC, and CTR

AES can be deployed in various modes of operation depending on the security and performance requirements. In the simplest form, ECB encrypts each plaintext block P_i independently as

$$C_i = E_k(P_i) \qquad (1)$$

where $E_k(.)$ denotes AES encryption with the secret key k. Each ciphertext block C_i thus depends only on the

corresponding plaintext block P_i. While this is simple to implement, identical plaintext blocks produce identical ciphertext which leaks structural information.

In CBC, each plaintext block P_i is XORed with the previous ciphertext block C_{i-1} before encryption:

$$C_i = E_k(P_i \oplus C_{i-1}), \qquad C_0 = IV \qquad (2)$$

where IV is a random initialization vector. The corresponding decryption step is

$$P_i = D_k(C_i) \oplus C_{i-1} \qquad (3)$$

Here, each ciphertext depends on all preceding plaintext blocks, improving security but enforcing a strict sequential dependency that prevents parallel execution.

CTR mode eliminates this limitation. A nonce N is concatenated with an incrementing counter value CTR_i to form a unique input for each block:

$$K_i = E_k(N \parallel CTR_i) \qquad (4)$$

where K_i is the keystream block for index i. Encryption and decryption are then both performed using a simple XOR:

$$C_i = P_i \oplus K_i, \; P_i = C_i \oplus K_i \qquad (5)$$

Unlike ECB and CBC, the plaintext does not directly enter the AES core. Instead, AES generates a pseudorandom keystream from the nonce–counter sequence, which is then combined with the plaintext. Since K_i is independent across blocks, all blocks can be processed in parallel, making CTR highly efficient for pipelined hardware implementations. Moreover, transmission errors remain localized: a bit error in one ciphertext block only affects the corresponding plaintext block after decryption.

2.4. Pipeline Architectures in AES

Pipelining is a widely used technique in digital design to increase performance by partitioning computations into sequential stages that execute concurrently. In AES implementations, pipelining enables multiple plaintext blocks to be processed in parallel, thereby significantly improving throughput while balancing latency and hardware utilization. Two primary pipelining approaches are employed in AES hardware architectures. Inter-round pipelining inserts registers between consecutive AES rounds. For AES-128, each of the 10 rounds may be implemented as a separate pipeline stage, or grouped into fewer stages (e.g., 2-stage or 6-stage designs). This approach is particularly effective in feedback free modes such as ECB and CTR, achieving high throughput on FPGA platforms in Figure 5a.

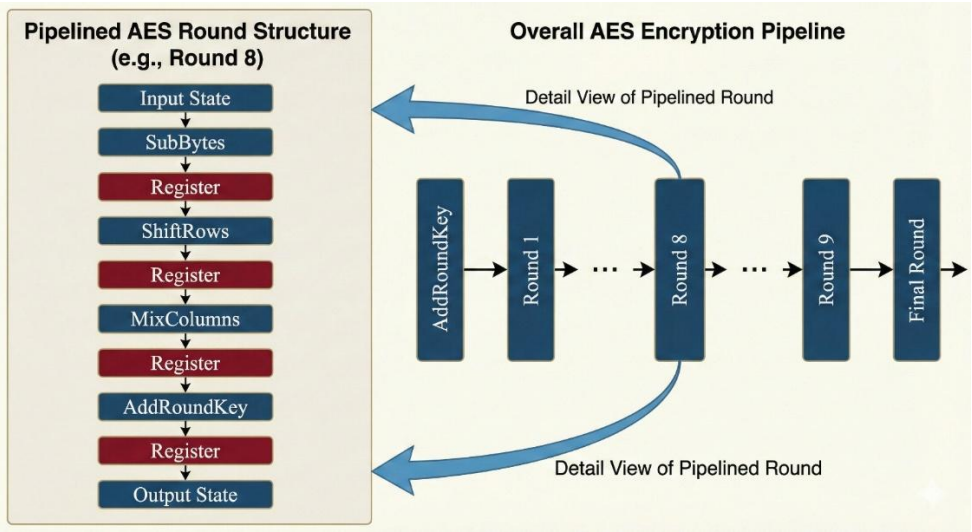


Figure 5a. Conceptual views of pipelining approaches in AES: inter-round

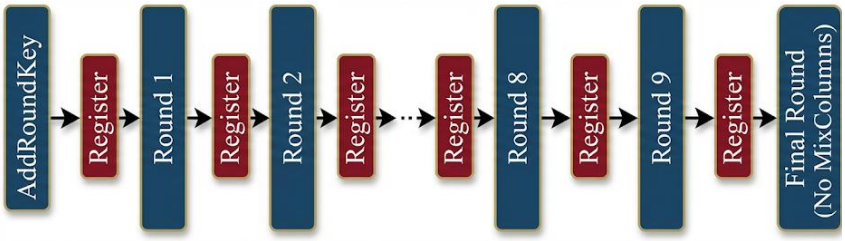


Figure 5b. Conceptual views of pipelining approaches in AES: inner-round

Inner-round pipelining (also referred to as the c-slow technique) introduces registers within a single round, between its internal transformations—SubBytes, ShiftRows, Mix-Columns, and AddRoundKey. This technique is advantageous in feedback-dependent modes like CBC, CFB, or OFB, where inter-round pipelining cannot be fully exploited. By interleaving multiple independent data streams, inner-round pipelining
ensures pipeline utilization and further enhances throughput in Figure 5b.

In this work, inter-round pipelining has been adopted in the AES core to increase throughput while maintaining reasonable FPGA resource usage. Although the selected mode of operation was not separately pipelined in this design, its intrinsic block-level independence naturally allows for such extensions in future implementations.

3. Material and Method

Conventional AES cores are typically designed in a mode-agnostic fashion, where the mode of operation logic such as counter management in CTR is handled externally. However, this separation may introduce unnecessary overhead and limit efficiency in hardware pipelines.

3.1. CTR Mode Specialized AES Core

To optimize the AES core for CTR operation, we designed a specialized architecture with two key FPGA-oriented optimizations: First, the counter-update mechanism is integrated directly into the data path. Rather than externally updating and supplying the nonce–counter pair at each cycle, which would require waiting for the updated counter value to be computed off-core and then transmitted back, incurring at least a one-cycle delay, the core autonomously generates this value internally, eliminating this external latency and simplifying the control logic. Second, the key schedule is fully precomputed during initialization and stored in a dedicated register array. This eliminates the combinatorial delay of on-the-fly key generation and provides simultaneous access to all round keys, which is crucial for maintaining pipeline throughput. The overall architecture of the proposed AES-CTR core is illustrated in Figure 6.

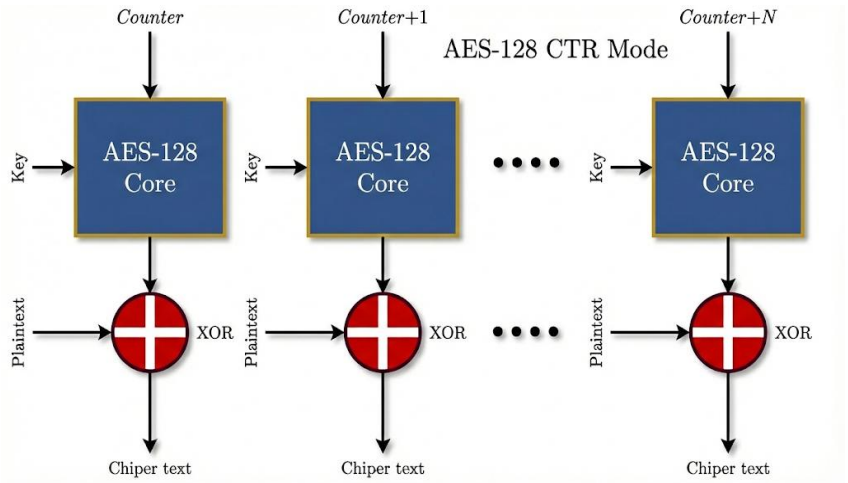


Figure 6. Illustration of the AES CTR mode

3.2. Pipelined AES-128 CTR Core Implementation

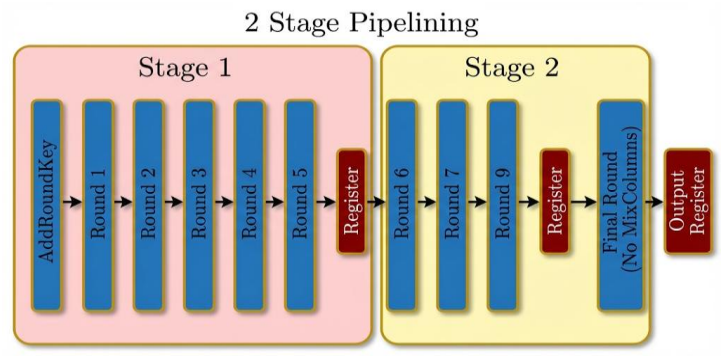
In this study, the AES-128 core was modified to support pipelined execution, with the objective of evaluating throughput–resource trade-offs under different pipeline depths. Three configurations were implemented: 2-stage, 6-stage, and fully pipelined.

2-stage pipeline: As it is shown in Figure 7a, Stage 1 includes the initial AddRoundKey and Rounds 1–5, while Stage 2 covers Rounds 6–9. The Final Round is implemented as a dedicated stage since it omits MixColumns. This grouping reduces the critical path compared to a non-pipelined core.

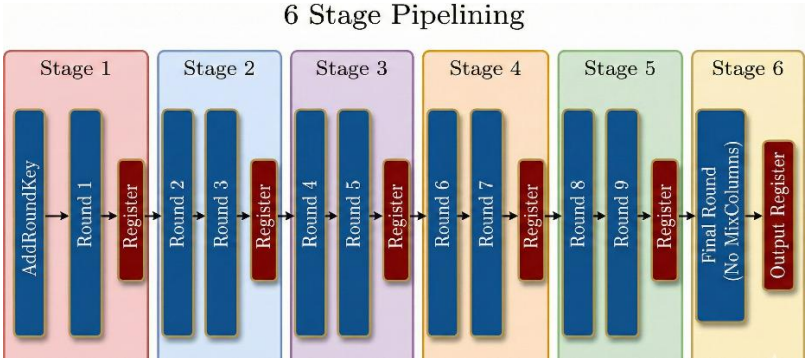
6-stage pipeline: The rounds are divided more evenly into six stages: Init+R1, R2–R3, R4–R5, R6–R7, R8–R9, and Final. This finer granularity improves throughput over the 2-stage design while maintaining moderate resource requirements. This is shown in Figure 7b.

Full pipeline: Each round, together with the initial AddRoundKey and Final Round, is mapped to its own pipeline stage. Once the pipeline is filled, one encrypted block is produced per cycle, maximizing throughput at the cost of increased LUT/FF usage. As pipeline depth increases, the critical path shortens, maximum clock frequency improves, and throughput rises.

However, this comes at the expense of additional FPGA resources due to register overhead. These implementations thus provide a tuneable balance between performance and area, allowing designers to select a configuration best suited to the target application.



(a)



(b)

Figure 7. Different pipeline implementations in AES

3.3 RGB Image Processing

RGB images are typically represented in 24-bit format, with 8 bits per channel (R, G, B). To align the image data with the 128-bit block size of AES, a preprocessing step is applied. First, the image is decomposed into three matrices representing the individual color channels. Each channel is then serialized into 128-bit blocks, which form the input to the AES encryption. Rather than merging the channels into a single stream, each is encrypted independently, preserving image structure and simplifying processing. MATLAB is employed for preprocessing, splitting the image into color channels and arranging pixel values into 128-bit segments. These blocks are then provided to the AESCTR core for encryption. This procedure ensures both data confidentiality and compatibility with real-time transmission requirements.

4. Implementation on FPGA

4.1. FPGA Board Testing

The design is tested using synthetic underwater RGB images and datasets such as EUVP (Zhang et al., 2022) and UIEB (Li et al., 2019), which contain realistic underwater distortions. The encryption and decryption correctness

is verified using pixel-by-pixel comparison.

4.2. Hardware Platform and Development Tools

The implementation was carried out on the Digilent Nexys 4 DDR development board, which features a Xilinx Artix-7 XC7A100T FPGA. The AES-128/CTR core was synthesized on this platform and tested under different pipeline configurations. The system clock frequency was set to 100 MHz. The design and synthesis flow was performed using the Xilinx Vivado Design Suite, while MATLAB's serial port libraries were used for the software side of UART communication.

This setup provided a complete end-to-end testing environment between the FPGA and the computer. To further analyse the statistical properties of the encrypted and original images, a Python-based testing tool was developed. This tool performed entropy measurements, adjacent-pixel correlation analysis, and histogram evaluations, thereby providing additional validation of encryption quality.

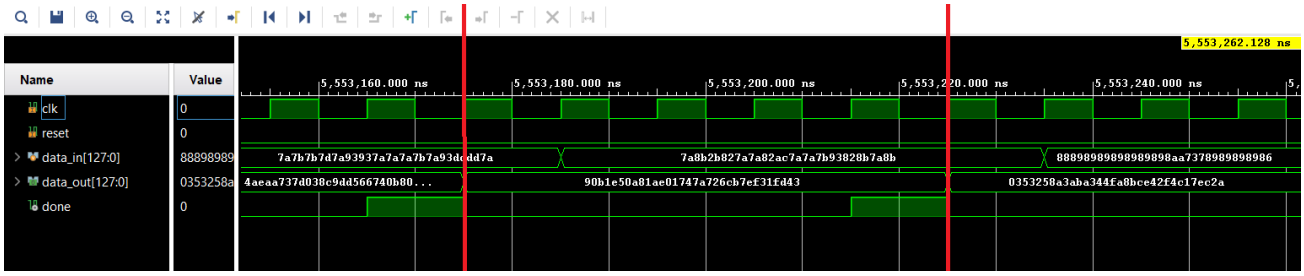
4.3. System Architecture

Test images were read in MATLAB, divided into 128-bit blocks, and transmitted to the FPGA board via the Universal Asynchronous Receiver-Transmitter (UART) interface at 57,600 baud for debugging purposes. On the FPGA, the AES-128 core was implemented in Counter mode and synthesized with multiple pipeline configurations (non-pipeline, 2-stage, 6-stage, and full pipeline). Both encryption and decryption operations were performed on the FPGA, ensuring that the entire cryptographic process was executed in hardware.

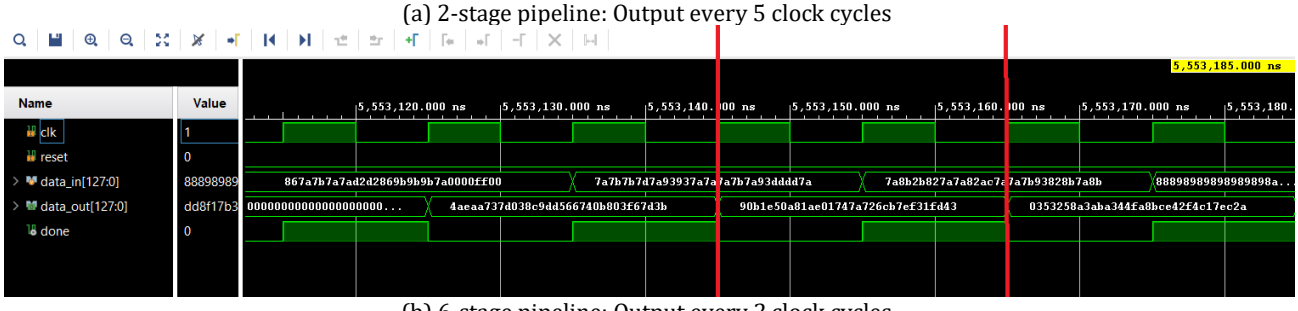
4.4. Hardware Resource Utilization and Performance Comparison

The output generation rate for each configuration was measured in the simulation, illustrating the trade-off between throughput and resource utilization.

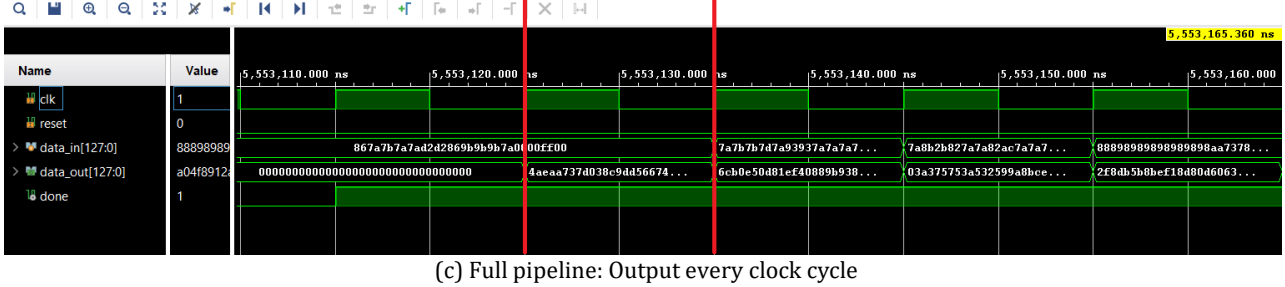
2-stage pipeline: The system produced one output every five clock cycles. Although the shallow depth keeps the processing latency relatively high, throughput improves compared to the non-pipelined design as shown in Figure 8a.



(a) 2-stage pipeline: Output every 5 clock cycles



(b) 6-stage pipeline: Output every 2 clock cycles



(c) Full pipeline: Output every clock cycle

Figure 8. Simulation results of AES-128/CTR pipelines with different depths: (a) 2-stage, (b) 6-stage, and (c) full pipeline

6-stage pipeline: With deeper partitioning, the system produces an output every two clock cycles. This finer granularity significantly increased throughput, albeit with higher resource usage as shown in Figure 8b.

Full pipeline: Mapping each round to its own stage resulted in one output block per clock cycle once the pipeline was filled. This configuration maximized throughput, but at the cost of greater FPGA resource consumption due to extensive register insertion as shown in Figure 8c. Overall, the results confirm that increasing pipeline depth reduces the critical path, improves maximum clock frequency, and significantly increases throughput. However, this benefit comes with higher resource utilization.

The 2-stage pipeline represents a balanced option for moderate performance with lower area cost, the 6-stage pipeline offers a trade-off favouring higher throughput, and the full pipeline maximizes performance at the expense of FPGA resources. The choice of configuration thus depends on the performance and resource constraints of the target application.

4.5. Throughput Calculation

The throughput of the AES-128/CTR design is defined as the number of encrypted bits produced per second, i.e., the effective data processing rate of the encryption engine. It depends on the block size, the system clock frequency, and the number of cycles required to generate an output in each pipeline configuration. All implementations were operated with a 100 MHz clock, corresponding to a 10 ns period. Since AES processes 128-bit blocks, the throughput can be expressed as:

$$Throughput = 128 \times \frac{f_{clk}}{N_{cycles}} \qquad (6)$$

where N_{cycles} denotes the number of clock cycles required to produce one output block, and f_{clk} is the system clock frequency. As the pipeline depth increases, N_{cycles} decreases, thereby improving the throughput. The measured performance and resource utilization for each pipeline configuration are summarized in Table I.

Table 1. Pipeline performance metrics for AES-128 CTR implementations

Pipeline	LUT	FF	DSP	BRAM	WNS [ns]	Power [W]	Throughput [Gbps]
Non-pipelined	3106	2193	0	0	0.481	0.13	1.16
2-stage	3222	2353	0	0	1.196	0.128	2.56
6-stage	5103	2628	0	0	1.289	0.182	6.40
Full pipeline	6403	2943	0	0	0.600	0.244	12.8

As seen in Table 1, throughput increases from 1.16 Gbps in the non-pipelined implementation to 12.8 Gbps in the fully pipelined core, with a corresponding increase in resource utilization. The 2-stage configuration represents a balanced trade-off between performance and area, the 6-stage configuration further improves throughput while moderately increasing resources, and the full pipeline achieves maximum performance at the cost of higher LUT and FF usage.

4.6. Cryptographic Validation

The cryptographic robustness of the proposed FPGA-based AES-128/CTR implementation was evaluated using standard statistical tests on image data. The encrypted and decrypted versions of the test image are presented in Figure 9.

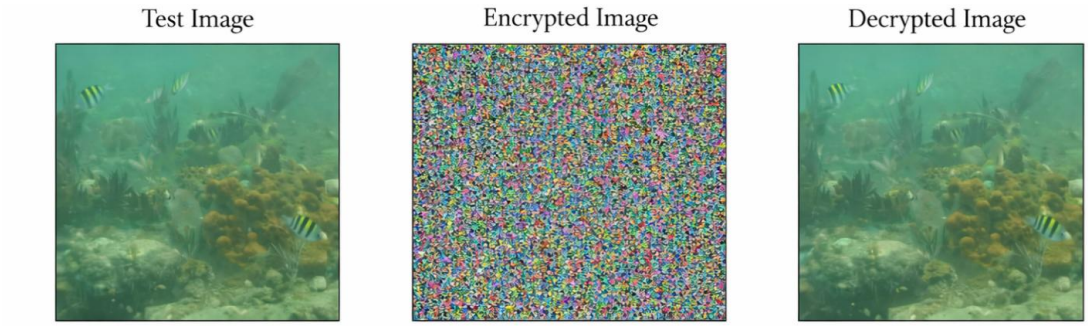


Figure 9. MATLAB output of underwater image encryption

Histogram analysis: As shown in Figure 10, the histogram of the original image is highly non-uniform, while the histogram of the encrypted image exhibits a nearly uniform distribution across all three-color channels. This confirms that the ciphertext achieves high statistical randomness.

Entropy analysis: The randomness of the encrypted images was quantified using Shannon entropy, defined as

$$H = -\sum_{i=0}^{255} p(i)\log_2(p(i)) \tag{7}$$

For an 8-bit grayscale image, the ideal entropy value is 8. In our tests, the encrypted images achieved entropy values of ~7.99, very close to the theoretical maximum. This indicates excellent information diffusion and is consistent with prior FPGA-based cryptographic image processing studies.

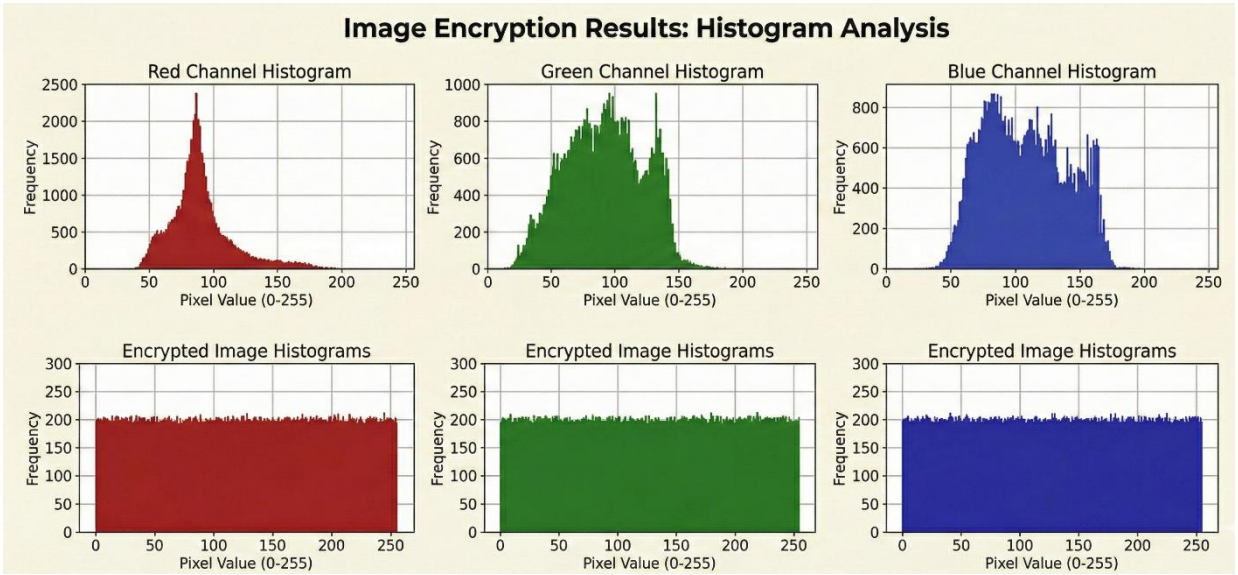


Figure 10. Comparison of histogram distributions before (top) and after (bottom) AES-128/CTR encryption

Correlation analysis: The correlation coefficient between adjacent pixels was used to evaluate spatial redundancy. For two neighbouring pixels x and y, the correlation is given by

$$r = \frac{E[(x - E[x])(y - E[y])]}{\sqrt{D[x] \cdot D[y]}} \tag{8}$$

where $E[\cdot]$ and $D[\cdot]$ denote expectation and variance, respectively. In the original images, $r \approx 0.97$, indicating high dependency, whereas in the encrypted images the value dropped close to zero, confirming that spatial dependencies were effectively removed.

Differential analysis: Resistance to differential attacks was quantified using NPCR and UACI metrics. NPCR is defined as

$$NPCR = \frac{\sum_{i=0}^{M}\sum_{j=1}^{N} D(i,j)}{M \times N} \times 100\% \rightarrow D(i,j) = \begin{cases} 1, & if\ C_1(i,j) \neq C_2(i,j), \\ 0, & if\ C_1(i,j) = C_2(i,j), \end{cases} \tag{9}$$

and C_1, C_2 are two ciphertext images differing in one plaintext pixel. UACI is defined as

$$UACI = \frac{1}{M\ X\ N}\sum_{i=1}^{M}\sum_{j=1}^{N}\frac{|C_1(i,j) - C_2(i,j)|}{255} \times 100\% \tag{10}$$

For the encrypted images, NPCR reached ~99.61% and UACI ~33.41%, both of which are very close to the ideal values (100% and 33%, respectively), confirming strong resistance to differential attacks. (Yang & Chien, 2020)

4.7. Comparison to State-of-the-Art Implementations

When comparing different FPGA implementations, raw throughput values alone can be misleading, as they are directly influenced by the device's operating frequency. To enable a fairer evaluation, throughput is normalized by dividing by the clock frequency, yielding throughput-per-MHz. This metric allows consistent comparison across platforms by reflecting architectural efficiency rather than absolute device speed.

Table 2. Comparison of AES-128 CTR implementations on different FPGA devices

Work	Mode	Key	FPGA	Fmax [MHz]	CpB	Thpt [Gb/s]	Gb/s/M	LUT	FF / BRAM
This work (non-pipe)	**CTR**	**128**	**A7-100T**	**100.0**	**11**	**1.16**	**0.012**	**3106**	**2193 / 0**
This work (2-stage)	**CTR**	**128**	**A7-100T**	**100.0**	**5**	**2.56**	**0.026**	**3222**	**2353 / 0**
This work (6-stage)	**CTR**	**128**	**A7-100T**	**100.0**	**2**	**6.40**	**0.064**	**5103**	**2628 / 0**
This work (full)	**CTR**	**128**	**A7-100T**	**100.0**	**1**	**12.80**	**0.128**	**6403**	**2943 / 0**
Qu & Shou (2009)	CTR	128	V5-LX85	576.1	1	73.73	0.128	22994	--- / ---
Fu & Hao (2005)	CTR	128	V2	212.5	1	27.10	0.128	17887	--- / ---
Guzmán et al. (2016)	CTR	128	V5-LX110T	272.6	1	34.89	0.128	11677	1484 / ---
Badillo et al. (2021)		128	Virtex-7	213	56	0.479	0.002	1425	1900/0
Lee et al. (2022)		128	KCU116	500	31	2.064	0.004	2195	907/0
Penagos et. al. (2024)		128	Zynq-7000	164.9	2,75	7.67	0.046	8772	1095/-
Penagos et al. (2024)		128	Kintex-7	238.7	2,75	11.11	0.046	7498	1076/-
Murugan et al. (2020)		128	Spartan-3	67.75	1	8.672	0.128	2156	680/-
Murugan et al. (2020)		128	Virtex-4	112.37	1	14.383	0.128	2127	684/-
Daoud et al. (2019) (HLS)		128	Zynq-7000	192	19	1.29	0.006	1417	830/0
Daoud et al. (2019)		128	Zynq-7000	219	1	28	0.128	9481	6787/0
Visconti et al. (2021)		128	ZCU102	220	1	28.16	0.128	13043	3877/-
Shahbazi et al.(2020)		128	Virtex-5	622.4	1	79.7	0.128	14966	19123/0

CpB: cycles per block. **Thpt**: throughput. **Gb/s/M**: throughput per MHz

The full pipeline implementation on the A7-100T achieved a normalized throughput of 0.128 Gbps/MHz. Notably, this value is identical to those obtained on higher-end platforms, which is shown in the Table II. As an example, in the V5-LX85 and V5-LX110T, which report normalized throughputs of 0.128 Gbps/MHz despite their substantially higher operating frequencies. For instance, the V5-LX85 delivers 73.73 Gbps at 576.1 MHz, and the V5-LX110T reaches 34.89 Gbps at 272.6 MHz, both yielding the same normalized value.

This comparison highlights that, while absolute throughput values are higher on these larger devices, the A7-100T achieves equivalent efficiency per MHz. Moreover, it achieves this with significantly fewer LUT and FF resources, highlighting its advantage in terms of resource efficiency. These results demonstrate that even smaller FPGA platforms can deliver performance comparable to high-end devices, making the A7-100T a robust alternative for efficient AES implementations in resource-constrained environments.

5. Result and Discussions

This work has demonstrated that significant performance gains for secure underwater communication. Hence, this can be achieved through FPGA-specific micro-architectural optimization of a standardized cryptographic primitive. By designing a CTR-specialized AES-128 core that integrates noncecounter management directly into the data path and employs a precomputed key schedule, we eliminated external control delays and created a pipeline-friendly architecture. The systematic implementation of three distinct pipeline configurations (2-stage, 6-stage, and fully pipelined) provided a clear characterization of the throughput-resource trade-off, offering designers a spectrum of choices from area-efficient to high-throughput implementations. Cryptographic validation confirmed the robustness of the approach, with statistical tests indicating strong resistance to differential attacks.

In summary, the primary contribution of this work lies not in a new cryptographic algorithm, but in a highly efficient FPGA implementation that makes the standardized AES-128/CTR algorithm practically viable for real-time, secure data transmission in challenging underwater environments. The results provide clear guidelines for selecting the appropriate pipeline depth based on the specific performance requirements and resource constraints of the target application. Future work will focus on extending the system for integration into remotely operated vehicles (ROVs), enabling secure, real-time data transmission in practical underwater missions. While the present design employed pipeline optimization at the round level, further gains could be achieved through fine-grained pipelining at the operational level (e.g., SubBytes, ShiftRows, MixColumns), thereby improving efficiency.

Acknowledgement

This work was supported by The Scientific and Technological Research Council of Turkiye (TUBITAK), Department

of Science Fellowships and Grant Programmes (BIDEB) through the 2232-B International Fellowship for Early Stage Researchers Program under Grant Number 121C126.

Conflict of Interest

No conflict of interest was declared by the authors.

References

Algredo-Badillo, I., Ramírez-Gutiérrez, K.A., Morales-Rosales, L.A., Pacheco Bautista, D. and Feregrino-Uribe, C., 2021. Hybrid pipeline hardware architecture based on error detection and correction for aes. Sensors, 21(16), p.5655.

Arul Murugan, C., Karthigaikumar, P. and Sathya Priya, S., 2020. FPGA implementation of hardware architecture with AES encryptor using sub-pipelined S-box techniques for compact applications. Automatika: časopis za automatiku, mjerenje, elektroniku, računarstvo i komunikacije, 61(4), pp.682-693.

Blazhevski, D., Bozhinovski, A., Stojchevska, B. and Pachovski, V., 2013. Modes of operation of the AES algorithm, pp.212-216.

Busacca, F., Galluccio, L., Palazzo, S., Panebianco, A., Qi, Z. and Pompili, D., 2024. Adaptive versus predictive techniques in underwater acoustic communication networks. Computer Networks, 252, p.110679.

Daoud, L., Hussein, F. and Rafla, N., 2019. High-level synthesis optimization of aes-128/192/256 encryption algorithms. International Journal of Computers and Their Applications, 29, pp.129-136.

Daoud, L., Hussein, F. and Rafla, N., 2019. Optimization of advanced encryption standard (aes) using vivado high level synthesis (hls).

González-García, J., Gómez-Espinosa, A., Cuan-Urquizo, E., García-Valdovinos, L.G., Salgado-Jiménez, T. and Escobedo Cabello, J.A., 2020. Autonomous underwater vehicles: Localization, navigation, and communication for collaborative missions. Applied sciences, 10(4), p.1256.

Goyal, S.B., Ravi, R.V., Verma, C., Raboaca, M.S. and Enescu, F.M., 2022. A lightweight cryptographic algorithm for underwater acoustic networks. Procedia Computer Science, 215, pp.266-273.

Guzmán, I.C., Nieto, R.D. and Bernal, Á., 2016. FPGA implementation of the AES-128 algorithm in non-feedback modes of operation. Dyna, 83(198), pp.37-43.

Jiang, S., 2018. On securing underwater acoustic networks: A survey. IEEE Communications Surveys & Tutorials, 21(1), pp.729-752.

Lee, U., Kim, H.K., Lim, Y.J. and Sunwoo, M.H., 2022, May. Resource-efficient FPGA implementation of advanced encryption standard. In 2022 IEEE International Symposium on Circuits and Systems (ISCAS) (pp. 1165-1169). IEEE.

Li, C., Guo, C., Ren, W., Cong, R., Hou, J., Kwong, S. and Tao, D., 2019. An underwater image enhancement benchmark dataset and beyond. IEEE transactions on image processing, 29, pp.4376-4389.

Maazouz, M., Toubal, A., Bengherbia, B., Houhou, O. and Batel, N., 2022. FPGA implementation of a chaos-based image encryption algorithm. Journal of King Saud University-Computer and Information Sciences, 34(10), pp.9926-9941.

National Institute of Standards and Technology (NIST)., 2023. Federal information processing standards publication 197-upd1: Advanced encryption standard (AES). https://doi.org/10.6028/NIST.FIPS.197-upd1

Penagos, H.P., Arenas, A.S.P. and Ortiz, M.A., 2024. Low area implementation of the advanced encryption standard with counter mode for system-on-chip-fpga. Elektron: ciencia y tecnología en la electrónica de hoy, 8(2), pp.71-76.

Peng, C., Du, X., Li, K. and Li, M., 2016. An Ultra-Lightweight Encryption Scheme in Underwater Acoustic Networks. Journal of Sensors, 2016(1), p.8763528.

Qu, S., Shou, G., Hu, Y., Guo, Z. and Qian, Z., 2009, May. High throughput, pipelined implementation of AES on FPGA. In 2009 International Symposium on Information Engineering and Electronic Commerce (pp. 542-545). IEEE.

Shahbazi, K. and Ko, S.B., 2020. High throughput and area-efficient FPGA implementation of AES for high-traffic applications. IET Computers & Digital Techniques, 14(6), pp.344-352.

Visconti, P., Velazquez, R., Capoccia, S. and De Fazio, R., 2021. High-performance AES-128 algorithm implementation by FPGA-based SoC for 5G communications. International Journal of Electrical and Computer Engineering (IJECE), 11(5), pp.4221-4232.

Yang, C.H. and Chien, Y.S., 2020. FPGA implementation and design of a hybrid chaos-AES color image encryption algorithm. Symmetry-Basel, 12(2), 189.

Yongzhi, F., Hao, L., Zhang, X. and Yang, R., 2005, December. Design of an Extremely High-Performance Counter Mode AES Reconfigurable Processor. In Proceedings of the Second International Conference on Embedded Software and Systems (ICESS 05), Xian, China, pp. 16-18.

Zhang, W., Zhuang, P., Sun, H.H., Li, G., Kwong, S. and Li, C., 2022. Underwater image enhancement via minimal color loss and locally adaptive contrast enhancement. IEEE transactions on image processing, 31, pp.3997-4010.