SSRG International Journal of Electronics and Communication Engineering
ISSN: 2348-8549/ https://doi.org/10.14445/23488549/IJECE-V13I11P122

Volume 13 Issue 11, 364-373, November 2026
© 2026 Seventh Sense Research Group®

Original Article

Area-Efficient and High-Throughput FPGA Implementation of the Enocoro128-v2 Stream Cipher for IEEE 802.15.4 Systems

Vivek Raj Kempanna[1], Dinesha Puttaraje Gowda[2]

[1,2]*Department of Electronics and Communication Engineering, Dayananda Sagar College of Engineering, Bengaluru, India*.

[1]*Corresponding Author: vivekgowda1990@gmail.com*

Received: 10 August 2026 Revised: 30 October 2026 Accepted: 06 November 2026 Published: 27 November 2026

Abstract *- A Stream Cipher (SC) is part of the symmetric cryptosystem, which plays a significant role in resource constraints and speed improvement of hardware implementation. Cryptographic approaches are very important for keeping data safe as communication and network technologies keep getting better. This manuscript uses an efficient Enocoro128-version2 (v2) stream cipher for IEEE 802.15.4 system applications. The Enocoro128 uses a 128-bit key and a 64-bit Initialization Vector (IV) to create an 8-bit Key Stream (KS). The Enocoro128-v2 uses three stages: initialization, internal state Updation, and output generation. The proposed Enocoro128-v2 offers low-cost implementation with minimal hardware resource usage and obtains higher Throughput with better hardware efficiency. The work is done using Verilog HDL in the Xilinx ISE tool and is implemented on an Artix-7 FPGA. Enocoro128-v2 uses 87 Slices, which is less than 1%, and runs at 414 MHz, using 100 milliwatts of total power. The Enocoro128-v2 achieves a high data transfer speed of 3.31 Gbps and has a hardware efficiency of 37.19 Mbps per slice. The new Enocoro128-v2 is being compared to other existing stream ciphers, and it shows better improvements in terms of resources and performance metrics.*

Keywords *- Enocoro128, Stream Cipher, FPGA, Lightweight, S-Box.*

1. Introduction

Cryptographic technologies are essential for safeguarding data security and privacy in contemporary digital applications. They are extensively utilized in areas including cloud computing, big data, digital governance, digital commerce, communications networks, and the Internet of Things (IoT) to safeguard sensitive data from security risks.

In resource-limited applications, a proficient cryptographic system must strike an appropriate equilibrium among robust security, rapid processing speed, minimal hardware complexity, and diminished resource use. Stream ciphers play an essential role in information security with confidentiality, among other types of cryptographic algorithms. Stream ciphers are implemented in either software or hardware approaches. These ciphers provide minimal error propagation and faster encryption/decryption operations. Most civil communication systems and international standards use stream ciphers to secure the data with confidentiality [1, 2]. The Lightweight Cryptographic (LWC) algorithms offer protective features with resource constraints for deployment in an embedded system. The LWC algorithms provide data integrity and confidentiality features. Suitable examples of LWC are block ciphers, which are implemented straightforwardly. Stream ciphers are typically low-power, compact, and faster than LWC and suitable for lower-constrained devices like Radio-Frequency Identification (RFID) tags for authentication. The stream ciphers are generally constructed using Feedback Shift Registers (FSRs) to generate Keystream bits. These Keystream bits are used further in the encryption/decryption process by performing the simple exclusive XOR operation.

In general, stream ciphers are classified based on FSRs, Add Rotation and Bitwise Xor (ARX), large table based, Cellular-Automata (CA)-based, and based approaches. The FSRs are either Linear Feedback Shift Registers (LFSR) or Feedback With Carry Shift Register (FCSR). The LFSR is further classified based on the Galois field and Fibonacci features.

These stream ciphers, except FSRs, are added with nonlinearity features to generate the Keystream bits. The nonlinearity features like filter, Substitution-Box (S-Box), clock-controlled, Finite State Machine (FSM), and Non-Linear FSR (NFSR) are used in SC to generate the Keystream bits [3, 4].

This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/)

The SCs are extended to their family with a new structural concept called Physical Clone-Resistant Function (PUFs). These ciphers are used to generate massive keystreams using random NFSRs. These are used in practical reconfigurable System-on-Chip (SoC) applications [5]. The hybrid mode stream cipher called PANAMA also acts as an SC or fast hashing function. These PANAMA ciphers are used effectively in Very Long-Instruction Word (VLIW) processor applications [6]. The TriviA stream cipher is introduced as a fast and secure authentication algorithm. The TriviA offers higher security by maintaining easy implementation without much modification in state size [7]. The extension of the Enocoro stream cipher called Enocoro128-verison2 based Pseudorandom Number Generator (PRNG) was introduced by Hitachi Ltd in 2010. This algorithm uses updated state generation than earlier Enocoro versions [8]. Statistical evaluations based on the National Institute of Standards and Technology (NIST) were performed on the enocoro128-v2 design in 2010 by Hell and Johansson [9], and the enocoro128-v2 successfully met all criteria. The enocoro128-v2 was subjected to many forms of analysis, including cube, algebraic, key-chosen IV attempts, time-memory-data-tradeoff assaults, guess-determine, and differentiating attempts to assess its performance [10].

1.1. Research Gaps

Despite the improvements in lightweight stream cipher implementations, there is still a need for an optimized Enocoro128-v2 architecture that improves continuous keystream generation without excessive FPGA resource overhead. Compact implementations currently available focus on hardware minimization, mainly by sequential execution of the datapath, which can limit processing speed and throughput. In particular, the sequential nature of the implementation of the ρ (rho), λ (lambda), S-box and state-update operations leads to clock-cycle dependencies that limit the rate of keystream generation. The research problem addressed in this work is to design an optimized hardware architecture that reduces sequential scheduling dependencies, improves the operating frequency and sustained throughput, and uses reasonable FPGA resource allocation. This is especially true for safe communication systems aimed at IEEE 802.15.4, where efficient cryptographic processing is desired under hardware resource and power constraints.

1.2. Novelty and Contribution

To fill this research gap, this work proposes an area-efficient and high-throughput FPGA implementation of the Enocoro128-v2 stream cipher based on a parallelized single-cycle datapath architecture. The proposed architecture reorders the operations of ρ and λ functions, S-box transformations and internal state-update mechanisms in an effort to reduce the sequential dependencies and boost the performance of the keystream generation. Unlike the conventional multi-cycle implementation as reported in [14], the proposed architecture produces one valid 8-bit keystream output every clock cycle after the initial 96-clock-cycle initialization phase. The implementation is evaluated using Verilog HDL on Xilinx Artix-7 FPGA platform (XC7A100T-3CSG324). The proposed design utilizes 87 slices and 301 LUTs. It has a maximum operating frequency of 413.817 MHz, throughput of 3.31 Gbps and total power consumption of 100 mW. The proposed implementation achieves about 2.19 times the operating frequency of the 8-bit architecture in [14] and about 10.95 times the reported throughput with only an additional nine FPGA slices. The results presented show that the proposed architectural reconfiguration provides a better throughput-area trade-off for the Enocoro128-v2 hardware implementation.

The paper is organized as follows: Section 2 explains the current stream cipher implementation and the performance study related to it. Section 3 explains in detail the proposed enocoro128-v2 hardware design and how it works. Section 4 shows the results and addresses about what they denote in the study. Finally, section 5 covers the whole work by showing improvements and looking ahead to the future.

2. Related Works

This part examines the creations of several lightweight stream ciphers and evaluates their performance. Li et al. [11] elucidate the implementation of the Plantlet, Lizard, Mickey, Grian-v1, and trivium stream ciphers on the FPGA architecture. The study examines all operational processes using hardware architectures. Every cipher is evaluated against its matching counterpart with enhancements in hardware limitations. Tsavos et al. [12] delineate the lightweight data streaming architecture implemented using programmable logic on the FPGA platform. This NLF module is incorporated into a data streaming system with reconfigurable logic to generate the corresponding cipher outputs. Alharbi et al. [13] present the eSTREAM ciphers with unrolling features for area efficiency analysis. These ciphers are implemented on ASIC and FPGA platforms to realize the performance metrics. The work also analyzes the effects of unrolling based on cell selection and buffer insertion. Pyrgas and Kitsos [14] explain the Enocoro128 Version 2.0 stream cipher for embedded devices.

The hardware designs of 8-bit and 4-bit datapath components are developed for the enocoro128 encryption algorithm. The study examines the hardware resources and contrasts current methodologies with enhancements in constraints. Dridi et al. [15] introduce the Secure Chaotic Generator-based Stream Cipher (SCSC) along with its implementation on FPGA. The SCSC uses a pseudorandom generator, discrete chaotic maps, and a mixing approach to random data generation. The SCSC achieves the 1.18 Gbps throughput with 1.09 Mbps/slice efficiency. A Hybrid Elementary Cellular Automaton (HECA) with adaptive mask (HECA-M) is described by Du et al. [16]. The driving and mask components make up the HECA-M. The suggested SC

has strong randomness and can withstand SC evaluations, according to statistical and secure assessments. A new lightweight sequential cipher based on S-box substitution and numerical addition is presented by Ye and Chen [17] with the goal of balancing security and resource savings. 280 is the intended security threshold. It has been confirmed that it can lower memory utilization while maintaining the security threshold, albeit at a minor performance cost. This structure produces a key stream with exceptional fictitious randomness. The improvement in cryptographic effectiveness is explained by Kuznetsov et al. [18] using a novel method for producing Substitution Boxes (S-boxes), considered essential for obtaining confusion and diffusion qualities in Substitution–Permutation Networks (SPNs). This development is especially important in the framework of smart networks, where striking an equilibrium between safety and efficiency is crucial. Khan et al. [19] present a thorough analysis of well-known LWSCs, offering insights into effectiveness, efficiency, and applicability in real-world use of resource-intensive streaming video cryptography on an ARM chip by keeping an emphasis on performance and resource usage. Two methods are implemented in a concurrent scenario by Madarro-Capó et al. [20]. They are used in SCs to quantify avalanche characteristics. The evaluation's findings contrast with the way the RC4 and HC256 SCs work in simultaneous and consecutive scenarios.

A linear technique for creating forgeries packets for the authenticated encryption algorithm ACORN is presented by Li and Cui [21], provided that the state is disclosed at a specific moment or that the key and IV have been identified. A comprehensive efficiency investigation of a few contemporary crypto graphic techniques on an IoT platform with limited resources is described by Sorescu et al. [22]. The outcomes show notable differences in the algorithms' efficiency. Various cryptographic techniques are presented in depth by Muhenga et al. [23], with a primary focus on symmetric key encryption and asymmetric key encryption, with hash functions that follow. Each iteration of stream ciphers or block ciphers displays extremely sophisticated symmetric algorithms, while the Caesar Cipher displays a fairly basic variable key.

The current MLOL, an innovative Authenticated Encryption with Associated Data (AEAD) technique based on the LOL scheme, is described by Zhang et al. [24]. MLOL improves safety and effectiveness by combining LFSR architecture optimizations with authenticated cryptography. A secure data transmission system for smart IoT sensor networks was suggested by Yordanov et al. [25], combining GSM/GPRS connectivity, ESP32 processing, and diverse sensors. With an average packet delivery time of 2.3 seconds, the system attained 98.7% classification precision and a 96% first-try transfer success rate. Wei et al. [26] suggested the probabilistic shaping-aided base Precoding Quantum Noise Stream Cipher (PSABP-QNSC) to lower encryption costs in QAM-QNSC systems. The usage of Gaussian-distributed bases improves security against eavesdropping and offers shaping advantage compared to traditional uniformly distributed bases. Pre-calculation, time-division multiplexing, timing synchronization, and partitioned storage optimization were used by Yu et al. [27] to create a hardware-oriented modular implementation of HXTS. The design effectively balanced area and efficiency with a peak throughput of 7.57 Gbps and full timing closure. Titanium 4-bit S-boxes designed for Feistel-based lightweight block ciphers were presented by Ventura et al. [28]. The suggested S-boxes are a competitive substitute for Feistel-based ciphers with bit-permutation layers as they offer robust security characteristics against differential, boomerang, algebraic, and linear attacks.

3. Proposed Encoro128-v2 Stream Cipher

The Enocoro-80 was first created in 2007 by Watanabe et al. [29] as a hardware-oriented stream cipher, and it was selected in Estream Project. Later, the authors are designed Encoro128-v2 to improve the security features [30]. The Enocoro128 Version 2 (v2) stream cipher mainly has a 128-bit secret key (k), a 64-bit Initialization Vector (IV), and 8-bit random key generation output. The Hardware architecture of Enocoro128-version 2 stream cipher at the initialization stage is illustrated in Figure 1. The enocoro128-v2 mainly contains four stages to generate the keystream output. The four stages are (1) Initialization Stage, (2) Rho (ρ) function and Lambda (λ) Function generation (4) State Updation and Output generation stages.

3.1. Initialization Stage

The Encoro128-v2 contains mainly three parameters, namely: 32-byte buffer (b_0-b_{31}), two 8-bit Internal States (a_0, a_1), 8-Bit Counter (ctr), Rho (ρ) function, and Lambda (λ) Function operations. The 32-byte buffer is decomposed into three parts: the First 16 bytes (b_0-b_{15}) for key representation, the next 8 bytes (b_{16}-b_{23}) for IV representation, and the last 8 bytes (b_{24}-b_{31}) for constant declaration. The Initial values representation for Enocoro128-v2 is tabulated in Table 1.

The 16-byte key is decomposed into four separate 32-bit key units: k_1, k_2, k_3, and k_4. The 32-bit key (k_1) is assigned with input key of MSB 32-bits (b_0: b_3 = k [128:96]) followed by k_2 is assigned with next 32-bits (b_4: b_7 = k [95:64]), k_3 is assigned with next 32-bits (b_8: b_{11} = k [63:32]), and lastly, k_4 is assigned with input key of LSB 32-bits (b_{12}: b_{15} = k [31:0]). In contrast, the 64-bit IV is decomposed into eight separate IV units. The buffer b_{16} = IV [63:56], …, b_{23} = IV [7:0]. Each byte contains 8 bits of information.

3.2. Rho (ρ) and Lambda (λ) Function Operation

The enocoro128-v2 performs the ρ function operation to generate the updated internal state values (a_0^u, a_1^u) using a 128-bit key, linear transformation, and s-box operation. The

ρ function operation is performed using four steps. Firstly, generate two temporary registers (x_0 and x_1) by performing the XOR operation individually with internal state a_0 and a_1 with Sbox8 of 32-bit keys k_1 and k_2, respectively. The representation for the same is shown in Equation (1). Secondly, perform the linear transformation of temporary registers (x_0 and x_1) and store the results in other registers (y_0 and y_1). The linear transformation performs two operations Equation (3): (1) XOR operation of two registers (x_0 and x_1) and (2) XOR operation of x_0 with a product of GF_8 and x_1. Where GF_8 = d = 2 for enocoro128-v2.

$$x_0 = a_0 \oplus s_8(k_1) \text{ and } x_1 = a_1 \oplus s_8(k_2) \quad (1)$$

$$(y_0, y_1) = L_t(x_0, x_1) \quad (2)$$

$$L_t(x_0, x_1) = \begin{bmatrix} 1 & 1 \\ 1 & d \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \{(x_0 \oplus x_1), (x_0 \oplus GF_8(x_1)$$
$$d \in GF_8; \quad (3)$$

The Galois Field (GF) in enocoro128 is used to perform multiplication and addition operations over GF (2^8) and GF (2^4). The corresponding expressions for GF (2^8) and GF (2^4) are represented in Equation (4). Lastly, the updated internal states (a_0^u, a_1^u) is obtained using keys (k_3 and k_4) by performing the XOR with corresponding registers (y_0 and y_1), respectively, and are represented in Equation (5).

$$GF_8(x) = x^8 + x^4 + x^3 + x^2 + 1; \text{ and } GF_4(x) = x^4 + x + 1; \quad (4)$$

$$a_0^u = y_0 \oplus s_8(k_3) \text{ and } a_1^u = y_1 \oplus s_8(k_4) \quad (5)$$

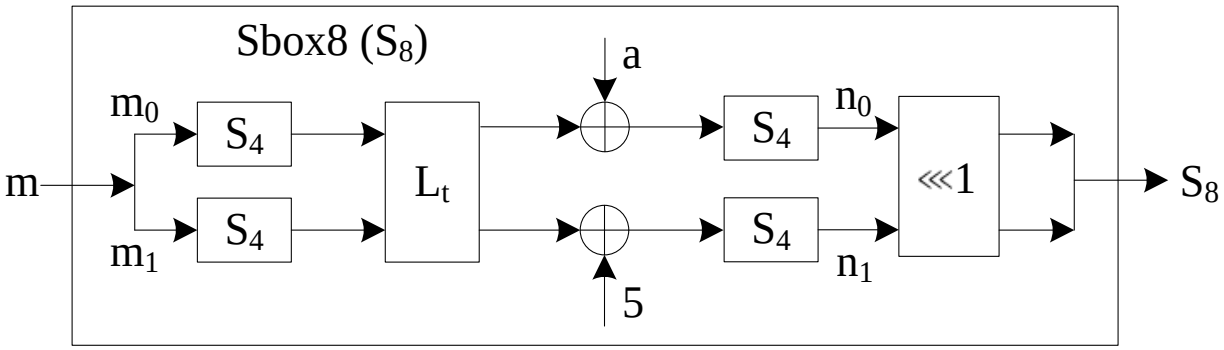

Fig. 1 Hardware architecture of Enocoro128-version 2 Stream Cipher at the initialization stage

Table 1. Initial values representation for Enocoro128-v2

Parameter	b_0: b_{15}	b_{16}: b_{23}	b_{24}	b_{25}	b_{26}	b_{27}	b_{28}	b_{29}	b_{30}	b_{31}	a_0	a_1	counter
Initial Values	key	IV	66	e9	4b	d4	ef	8a	2c	3b	88	4c	1
No. of bits	128	64	8	8	8	8	8	8	8	8	8	8	8

The Substitution box-8 (Sbox8 or S8) performs the mapping of the 8-bit inputs to corresponding 8-bit outputs. The Sbox8 mainly contains four sbox4, linear transformation, two XOR operations, and one left rotational operation. The Sbox8 architecture is represented in Figure 2. The Sbox4 representation is tabulated in Table 2. The Linear Transformation (L_t) and the Sbox8 are represented using Equations (6)-(8). Where constant p = 8'h4, and it is applicable only for enocor128-v2. Lastly, the Sbox8 output is generated by left rotation by 1-bit with (n_0, n_1) values. So $S_8(m) = (n_0 \parallel n_1) \lll 1$.

$$L_t(m_0\, m_1) = \begin{bmatrix} 1 & p \\ p & 1 \end{bmatrix} \begin{bmatrix} m_0 \\ m_1 \end{bmatrix} \quad m_0, m_1, p \in GF_4; \quad (6)$$

$$n_0 = S_4(S_4(m_0) \oplus \text{p}.S_4(m_1) \oplus 8'\text{ha}) \quad (7)$$

$$n_1 = S_4(\text{p}.S_4(m_0) \oplus S_4(m_1) \oplus 8'\text{h5}) \quad (8)$$

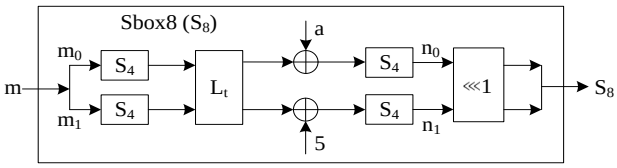

Fig. 2 Hardware architecture of the Sbox-8

Table 2. Sbox-4 (S4) representation

I	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
O	1	3	9	a	5	e	7	2	d	0	c	f	4	8	6	b

The Lambda (λ) function performs mainly three operations: Buffer Rotation (Byte-wise), performing XOR operation of internal state (a_0) with 31st Buffer (b_{31}) and Counter (ctr), and lastly, the three XOR operations in a feedback manner. These three operations are represented in Equations (9)-(11).

$$b_i^u = b_{i-1}; \text{ where } i \neq 0; \quad (9)$$

$$b_0^u = b_{31} \oplus a_0 \oplus ctr; \quad (10)$$

$$b_3^u = b_2 \oplus b_6; \quad b_8^u = b_7 \oplus b_{15}; \quad b_{17}^u = b_{16} \oplus b_{128}; \quad (11)$$

3.3. State Update and Output Generation

The state Updation operation is performed only after Initialization, rho, and lambda functions are generated. The Enocoro128-v2 performs four operations to update the state 96-times is as follows: (a) Updation of 31st Buffer by performing XOR with Counter (ctr): $b_{31}^u = b_{31} \oplus ctr$; (b) Update counter (ctr) by performing the GF (8) operation with value 2: $ctr = GF_8(ctr, 2)$. (c) Updation of Rho function: a_0^u, a_1^u. (d) Updation of Lambda function: $b_i^u, b_3^u, b_8^u, b_{17}^u$. Consider updated state output (a_1^u) as final 8-bit keystream output ($K_o = a_1^u$) for enocoro128-v2 stream cipher.

3.4. Encryption and Decryption Operation

The Encoro128 generated the 8-bit random key output and was used further in encryption and decryption operations. The Encoro128-based encryption/decryption process contains a 128-bit key, a 64-bit initialization vector (IV), 8-bit Plain Data Input (P_i), 8-bit Keystream Output (K_o), 8-bit Cipher Output (C_o), and 8-bit Decrypted Output (D_o). The encryption/decryption operation using keystream output is represented in Equation (12) as follows:

$$C_o = P_i \oplus K_o \text{ and } D_o = C_o \oplus K_o \quad (12)$$

4. Results and Discussion

This section discusses the enocoro128-v2 stream cipher simulation, synthesis, and performance comparison results. The Enocoro128 cipher and encoro128-based encryption/ decryption module are synthesized and deployed on the FPGA (Artix-7) device. The work is carried out in Verilog HDL using Xilinx ISE 14.7 environment for synthesis and Modelsim simulator for simulation processes. The Artix-7 FPGA (Device: XC7A100T, Package: CSG324) contains 15.85K logic cells, 126.8K slice registers, 63.4 K slice LUTs, 210 bonded Input-Output Buffers (IOBs), and 240 DSP slices. The resource utilization is estimated after a place-and-route process using Xilinx ISE tool. The synthesis and implementation analysis is performed using the default balanced design strategy of Xilinx. The simulation results of the Enocoro128-based Encryption/Decryption Module are represented in Figure 3. The system functions commence solely upon the activation of the global Clock (clk) with a low-active asynchronous Reset (rst). The control Signal (en) is activated once to initialize the values of the initialization stage. Define the 8-bit plain text in a sequence along with the 128-bit secret key (key) to generate the 8-bit random Keystream output (key_out). The Enocoro128-v2 needs 96 clock cycles for the startup stage. Upon finalizing the initialization, the suggested parallel datapath produces a valid 8-bit keystream output with each clock cycle. Consequently, the stated one-cycle latency signifies the steady-state output duration, whereas the 96 clock cycles denote the initial starting latency. The 8-bit Ciphertext (ct) and recovered decryption output (pt_o) were generated using the 8-bit Keystream output.

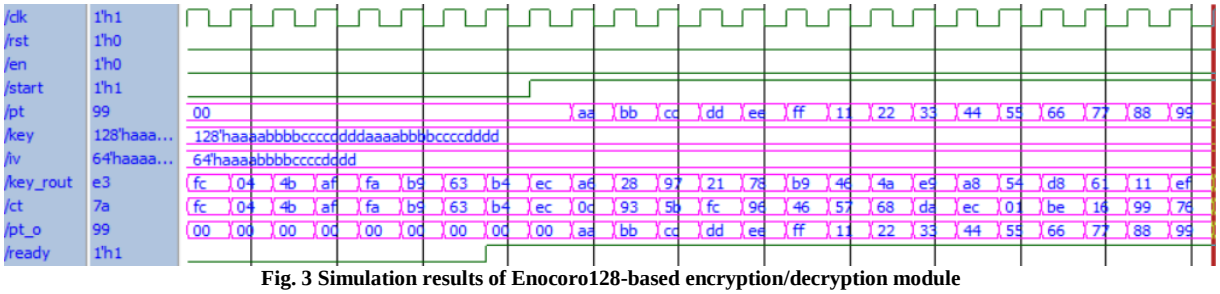

Fig. 3 Simulation results of Enocoro128-based encryption/decryption module

Table 3. Resource utilization of Enocoro128-based encryption/decryption module on Artix-7

Resources	Enocoro128	Enc/Dec-Enocoro128
Slices	87	89
LUTs	301	304
LUT-FFs	165	164
Max. Frequency (MHz)	413.817	413.817
Dynamic Power (mW)	18	20
Total power (mW)	100	102
Latency (CC)	1	1
Throughput (Gbps)	3.31	3.31
Efficiency (Mbps/Slices)	38.06	37.192
Efficiency (Mbps/LUTs)	11	10.89

The work verifies through cycle-accurate simulation that the proposed Enocoro128-v2 architecture completes the first phase of initialization in 96 clock cycles. After initialization, the design produces a valid 8-bit keystream output in every subsequent clock cycle without any further multi-cycle scheduling. The timing analysis also confirms a maximum operating frequency of 413.817 MHz, which corresponds to a steady-state throughput of about 3.31 Gbps (8 × 413.817 MHz). Thus, the simulation and FPGA timing results confirm the claimed continuous keystream generation capability of the proposed architecture.

The resource usage and efficiency of the Enocoro128-v2 stream cipher together with its encryption/decryption module are presented in Table 3. The Encoro128-v2 employs 87 slices, 301 LUTs, and 165 LUT-FF pairings, functioning at a frequency of 413.9 MHz while requiring a total power of 100 mW on the Artix-7 FPGA. The implementation completed successfully with no warnings. The minimum clock period reported by the post-synthesis timing analysis is 2.417 ns. This means that the maximum operating frequency is 413.817 MHz. The maximum delay in the combinational paths is 0.677 ns. Conversely, the encryption/deception module based on Encoro128-v2 employs 89 slices, 304 LUTs, 164 LUT-FF pairs, and functions at a frequency of 413.9 MHz while requiring a total power of 102 mW on the Artix-7 FPGA. The estimated total power includes static and dynamic power. The power consumption of the proposed Enocoro128-v2 architecture is estimated by using Xilinx XPower Analyzer integrated with Xilinx ISE 14.7.

The analysis is performed on an Artix-7 XC7A100T FPGA, under normal process conditions, commercial temperature grade and ambient temperature 25 °C, using vector-less activity propagation with default assumptions regarding a switching activity. The standalone implementation of Enocoro128-v2 consumes 100 mW, including 18 mW dynamic power and 82 mW quiescent power. For the encryption/decryption implementation, the power consumption is 102 mW (dynamic power is 20 mW and quiescent power is 82 mW).

The results show that the additional encryption/decryption functionality results in only a marginal increase in the estimated total power consumption. The Encoro128-v2 based encryption /decryption module operates at 3.31 Gbps throughput with 1 Clock Cycle (cc) latency. The hardware efficiency of the encryption/decryption module achieves 37.192 Mbps/slices and 10.89 Mbps/LUTs concerning the Slices and LUTs, respectively.

4.1. Performance Comparison

The Encoro128-v2 stream cipher architecture is compared by resource and performance metrics with existing stream cipher architectures and tabulated in Table 4. The comparison mainly concerning on stream cipher type, FPGA device used, Area usage, operating frequency (Fmax), Throughput (Gbps), and efficiency (Mbps/Slices). The performance metrics comparison of the proposed enocoro128-v2 with previous stream ciphers is illustrated in Figure 4.

The Enocoro128 [3] utilizes more slices, FFs, lesser operating frequency, and Throughput on low-end FPGA than the proposed enocoro128-v2. The hardware architecture of the s-box construction of enocoro128 [3] uses the traditional approach. The Grain [11] operates at a lower frequency and is less efficient, using more slices and FFs compared to the suggested enocoro128-v2. In comparison to Mickey 2.0, the suggested enocoro128-v2 runs at higher frequency, throughput, and efficiency while using less space (Slices, LUTs, and FFs) [11].

A traditional S-box is used by Mickey 2.0 [11], and it is difficult to create key generation using two shift registers. During the key generation phase, the Plantlet [11] employs two shift registers (LFSR and NFSR) with a memory module and feedback mechanism for storing secret keys. Compared to the suggested Enocoro128-v2, the key generation phase's operation is straightforward and requires less chip area. However, compared to the suggested enocoro128-v2, the Plantlet [11], which operates at a lower frequency, achieves limited throughput and efficiency. Compared to the suggested enocoro128-v2, the Lizard [11] uses more area, operates at a lower frequency, and has higher throughput and efficiency. The Lizard [11] generates keys utilizing a NFSR in a 121-stage internal state process. The Lizard has extra second key addition stages as well as the internal state and update function [11]. These stages make use of extra chip surface by updating the state through a feedback loop.

Snow 2.0 [12] generates the key stream based on reconfigurable logic. The reconfigurable logic uses many registers, memory, adders, and finite state machine to construct Snow 2.0 [12]. The feedback function Updation and storing of the data in memory is a complex task in Snow 2.0. Snow 2.0 [12] utilizes more LUTs, operates at a lower frequency, and obtains the average Throughput and efficiency than the proposed enocoro128-v2. The Mickey 2.0, Grain128, and Trivium ciphers [13] are eSTREAM hardware-based ciphers. The proposed enocoro128-v2 offers a better chip area than Mickey 2.0, Grain128, and Trivium [13].Mickey 2.0 and Trivium operate at a lower frequency, by obtaining minimal Throughputs and efficiency than the proposed enocoro128-v2. However, Grain128 [13] gives better frequency than the proposed enocoro128-v2 due to simple state updation. The Secure Chaos-Based Stream Cipher (SCbSC) [15] is implemented using a hybrid approach to generate the random key. The SCbSC [15] utilizes more chip area (Slices), operates at moderate frequency, and obtains minimal Throughput and lower efficiency than the proposed enocoro128-v2.

Table 4. Performance comparison of proposed Enocor128 cipher with existing stream ciphers

Ciphers	FPGA	Slices (Area)	LUTs	FFs	Fmax (MHz)	Throughput (Mbps)	Efficiency (Mbps/Slice)
Enocoro128 [3]	Spartan-3	442	NA	362	149	1200	0.33
Enocoro128 [3]	Spartan-3	292	NA	239	118	900	0.38
Proposed Work	Spartan-3	453	836	330	150.86	1207	2.664
Mickey 128 2.0 [13]	Spartan-6	317	410	360	370	370	1.17
Trivium [13]	Spartan-6	510	370	575	100	100	0.19
Proposed Work	Spartan-6	89	289	177	291.92	2336	26.25
Grain -128 [13]	Virtex-7	198	206	212	768.76	768.76	3.88
Proposed Work	Virtex-7	89	306	166	579.198	4634	52.06
Grain v1 [11]	Artix-7	111	361	166	250	4000	36.03
Mickey 2.0 [11]	Artix-7	107	370	374	384	384	3.59
Lizard [11]	Artix-7	150	466	241	200	1200	4.98
Plantlet [11]	Artix-7	62	153	194	303	303	4.89
Snow 2.0 [12]	Artix-7	NA	3297	NA	64	2040	5.1
Proposed Work	Artix-7	87	301	165	413.817	3310	38.06

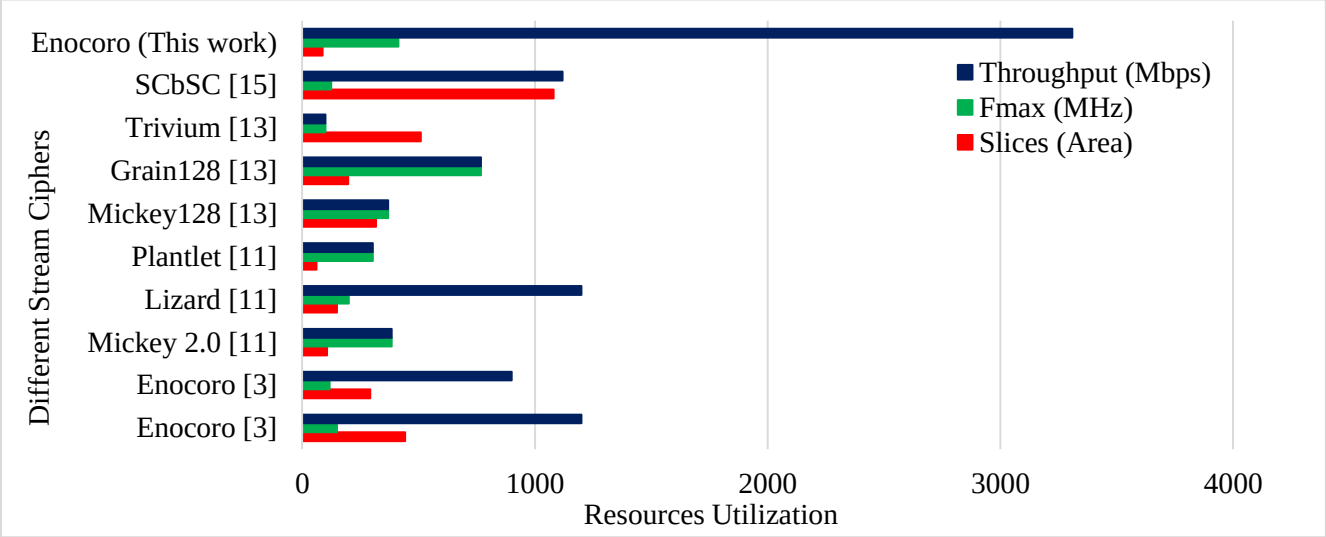

Fig. 4 Comparative analysis of different stream ciphers

Comparative results in Table 4 are collected from previously published FPGA implementation studies and proposed Enocoro128-v2 implementation. The performance numbers reported here are extracted from the respective publications and not from an independent synthesis of all reference architectures on the same FPGA devices and tool settings. The comparison is meant to be a literature-based evaluation of architectural efficiency, since factors like FPGA families, device capacities, synthesis tools and implementation constraints can influence the hardware performance.

4.2. Performance Comparison with Enocoro [14]

The current work's contribution is based on an architecture-based restructuring of the Enocoro128-v2 datapath for parallel to produce one valid 8-bit keystream output per clock cycle after the 96-cycle initialization phase, rather than just using a different FPGA device, synthesis tool, or simulation environment. Table 5 shows the performance comparison between the proposed Enocoro-128v2 and Ref [14]. Compact 8-bit and 4-bit Enocoro128-v2 architectures have been developed by Pyrgas and Kitsos [14].

Their 8-bit design achieves 302 Mbps with 78 slices at 189 MHz, producing 8-bit output with a 5-clock-cycle round Latency. Sequential clock-cycle scheduling is expressly used in their architecture: The Liner operation itself needs several clock cycles, but the λ-function XOR operations are carried out sequentially. As a result, the control subsystem employs counters to regulate the clock cycles inside each round as well as the rounds that are completed. The proposed design reorganizes the ρ and λ functions, S-box, linear

transformation, state-update logic, and output generation into simultaneous byte-level combinational paths, thereby therefore minimizing the sequential dependencies among sub-operations. The design effectively accomplishes the essential state update and provides a valid 8-bit keystream output with each clock cycle subsequent to initialization. This architectural parallelization facilitates the transition from 5 cycles per round in [14] to only 1 clock cycle for every valid keystream output. The proposed architecture consists of 87 slices, compared to the 78 slices in [14], demonstrating that the additional hardware resources are intentionally utilized for significantly higher parallelism and performance.

Table 5. The performance comparison of proposed Enocoro-128v2 with Reference [14]

Parameters	Enocoro [14]	Present work
Datapath	8-bit	8-bit
FPGA	Artix-7 (XC7A35T)	Artix-7 (XC7A100T)
Output width	8 bits	8 bits
Main architectural Method	Sequential multi-cycle datapath	Parallelized single-cycle datapath
Control operation	Round counter and intra-round cycle counter	Reduced cycle scheduling for continuous output
Frequency (MHz)	189	413.817
Slices	78	87
Round/output cycle requirement for Latency	5 cycles/round	1 cycle/output
Throughput	302 Mbps	3.31 Gbps
Efficiency (Throughput/Slice)	3.87 Mbps/slice	38.06 Mbps/slice

The proposed Enocoro128-v2 architecture uses 87 FPGA slices instead of 78 slices in the existing architecture presented in Ref. [14], which is about 11.54 % increase in slice utilization. This extra hardware overhead is intentionally added to allow the parallel execution of the internal cryptographic operations and to avoid sequential multi-cycle scheduling. The architecture proposed here is evaluated on the Artix-7 XC7A100T with Xilinx ISE 14.7, although implementation results using the Artix-7 XC7A35T FPGA are reported in Ref. [14]. Both devices belong to the same family of FPGAs. However, the device capacity might play a role in the synthesis results. Table 5 mainly compares the architectural improvements like the parallelized datapath, reduced cycle scheduling, and higher keystream generation rate. The throughput of proposed architecture is determined by the steady-state output rate instead of initialization latency. Given that Enocoro128-v2 generates an 8-bit keystream result and the suggested design yields a valid 8-bit output with each clock cycle post-initialization, the Throughput (T) is expressed using Equation (13):

$$T = (N_{\text{out}} \times f_{\text{clk}})/C_{\text{out}}. \quad (13)$$

N_{out} is equals 8 bits per output, f_{clk} is 413.817 MHz, and C_{out} is 1 clock cycle/output. Consequently, the throughput is determined using the actual steady-state output rate and, for Reference [14] is expressed using Equations (14) and (15), respectively.

$$T = (8 \times 413.817)/1 = 3.31\ Gbps. \quad (14)$$

$$T = (8 \times 189)/5 = 302.4\ Mbps \approx 302\ Mbps \quad (15)$$

Consequently, the stated 3.31 Gbps derives from a single 8-bit valid output for each clock cycle, rather than merely multiplying the operational frequency by the data width. The proposed architecture therefore attains around 10.95× more throughput compared to the 8-bit architecture referenced in [14]. It is important to note that 96 clock cycles are essential for the initial phase of Enocoro128-v2 setup. The specified 1-cycle delay relates to the consistent interval between each valid 8-bit keystream output post-initialization, except for the startup latency to the initial output.

The suggested study distinctly specifies the parallel single-cycle datapath and minimized sequential scheduling as the key methodological contribution. The straightforward comparison with [14], in addition with the precise throughput analysis, has been integrated to clearly highlight the architectural enhancement of proposed design.

4.3. Discussion

The enhanced performance of the suggested Enocoro128-v2 design is mainly due to the improvement of the internal datapath and keystream-generation tasks. The ρ (rho) and λ (lambda) functions, S-box transformations, state-update procedures, and control actions are arranged to diminish redundant hardware complexity and augment data processing efficiency. Specifically, the S-box and simplified state-update mechanisms decrease combinational overhead, allowing the design to achieve a maximum frequency of 413.817 MHz while utilizing just 87 slices, 301 LUTs, and 165 LUT-FF pairs. The structure accomplishes 1-cycle latency, hence significantly boosting its high sustained throughput.

The suggested design offers a beneficial area–performance balance in comparison to other lightweight stream-cipher solutions discussed in existing literature. The findings indicate that Enocoro128-v2 attains superior operating frequency and throughput compared to several other implementations, all while preserving minimal FPGA resource use. The suggested design achieves a throughput of 3.31 Gbps and a slice efficiency of 38.06 Mbps in the stream-cipher implementation, indicating that the expansion in area is successfully leveraged to attain markedly enhanced processing performance.

These enhancements illustrate that the architectural refinements successfully equilibrate area, frequency, latency, throughput, and hardware efficiency, rendering the suggested design appropriate for high-speed, resource-limited IEEE 802.15.4 systems.The proposed work mainly focuses on FPGA-based architectural optimization and performance evaluation of the Enocoro128-v2 stream cipher, rather than a complete IEEE 802.15.4 protocol-level implementation. The obtained hardware performance in terms of 3.31 Gbps throughput, 413.817 MHz operating frequency and low resource consumption demonstrates the potential of the proposed architecture for resource-constrained wireless communication applications. However, packet-level encryption, protocol integration and IEEE 802.15.4 compliance validation are beyond the scope of the present study. Therefore, the suggested cipher architecture could be a potential cryptographic building block for future development of secure transceivers based on IEEE 802.15.4.

5. Conclusion and Future Scope

This work describes the development and construction of the effective and high-throughput Enocoro128-v2 stream cipher on Artix-7 FPGA. The Enocoro128-v2 cipher produces an 8-bit random keystream using a 64-bit IV and a 128-bit secret key. The primary steps of the Enocoro128-v2 encryption include output creation, internal state update, and startup.

The Enocoro128-v2 cipher's internal state update is largely dependent on the creation of the rho and lambda functions. The Enocoro128-v2 stream cipher uses less than 1% of the chip area and only 100 mW of total power on an Artix-7 FPGA to operate at 414 MHz. On the Artix-7 processor, the Enocoro128-v2 stream cipher delivers 3.31 Gbps throughput with an efficiency of 37.19 Mbps/slice. In comparison to current stream ciphers, the suggested Enocoro128-v2 stream cipher offers a little gain in chip area (Slices and LUTs), frequency, throughput, and hardware efficiency. The digital IEEE 802.15.4 transceiver will eventually employ the Enocoro128-v2 stream cipher for secure phase encryption and decryption.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Funding Statement

There has been no significant financial support for this work that could have influenced its outcome.

References

[1] Lin Jiao, Yonglin Hao, and Dengguo Feng, “Stream Cipher Designs: A Review,” *Science China Information Sciences*, vol. 63, no. 3, pp. 1-25, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[2] Raghavendra Ananth, and Narayana Swamy Ramaiah, “An Exhaustive Review of the Stream Ciphers and Their Performance Analysis,” *International Journal of Reconfigurable and Embedded Systems*, vol. 13, no. 2, pp. 360-371, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[3] Charalampos Manifavas et al., “A Survey of Lightweight Stream Ciphers for Embedded Systems,” *Security and Communication Networks*, vol. 9, no. 10, pp. 939-1265, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[4] Sergiy O. Gnatyuk et al., “Improved Secure Stream Cipher for Cloud Computing,” *International Conference on ICT in Education, Research and Industrial Applications*, pp. 183-197, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[5] Ayoub Mars, and Wael Adi, “New Family of Stream Ciphers as Physically Clone-Resistant VLSI-Structures,” *Cryptography*, vol. 3, no. 2, pp. 1-22, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[6] Joan Daemen, and Craig Clapp, “Fast Hashing and Stream Encryption with PANAMA,” *International Workshop on Fast Software Encryption*, pp. 60-74, 1998. [CrossRef] [Google Scholar] [Publisher Link]

[7] Avik Chakraborti et al., “Trivia: A Fast and Secure Authenticated Encryption Scheme,” *International Workshop on Cryptographic Hardware and Embedded Systems*, pp. 330-353, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[8] Pseudorandom Number Generator Enocoro–Specification Ver. 2.0, Hitachi Ltd, 2010. [Online]. Available: https://www.cryptrec.go.jp/en/cryptrec_13_spec_cypherlist_files/PDF/23_00espec.pdf

[9] Martin Hell, and Thomas Johansson, “*Security Evaluation of Stream Cipher Enocoro-128v2*,” CRYPTREC Technical Report, pp. 1-31, 2010. [Google Scholar] [Publisher Link]

[10] Lin Ding, Chenhui Jin, and Jie Guan, “Slide Attack on Standard Stream Cipher Enocoro-80 in the Related-Key Chosen IV Setting,” *Pervasive and Mobile Computing*, vol. 24, pp. 224-230, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[11] Bohan Li, Meicheng Liu, and Dongdai Lin, “FPGA Implementations of Grain v1, Mickey 2.0, Trivium, Lizard and Plantlet,” *Microprocessors and Microsystems*, vol. 78, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[12] Marios Tsavos, Nicolas Sklavos, and George Ph. Alexiou, "Lightweight Security Data Streaming, based on Reconfigurable Logic, for FPGA Platform," *2020 23rd Euromicro Conference on Digital System Design (DSD)*, Kranj, Slovenia, pp. 277-280, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[13] Fares Alharbi et al., "Analysis of Area-Efficiency vs. Unrolling for eSTREAM Hardware Portfolio Stream Ciphers," *Electronics*, vol. 9, no. 11, pp. 1-17, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[14] Lampros Pyrgas, and Paris Kitsos, "Compact Hardware Architectures of Enocoro-128v2 Stream Cipher for Constrained Embedded Devices," *Electronics*, vol. 9, no. 9, pp. 1-14, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[15] Fethi Dridi et al., "The Design and FPGA-based Implementation of a Stream Cipher based on a Secure Chaotic Generator," *Applied Sciences*, vol. 11, no. 2, pp. 1-19, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[16] Peng Du et al., "A Novel Hybrid Elementary Cellular Automata and its Application in a Stream Cipher," *Applied Sciences*, vol. 14, no. 21, pp. 1-21, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[17] Jun Ye, and Yabing Chen, "SC-SA: Byte-Oriented Lightweight Stream Ciphers based on S-box Substitution," *Symmetry*, vol. 16, no. 8, pp. 1-20, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[18] Oleksandr Kuznetsov et al., "Enhancing Smart Communication Security: A Novel Cost Function for Efficient S-Box Generation in Symmetric Key Cryptography," *Cryptography*, vol. 8, no. 2, pp. 1-15, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[19] Mohsin Khan, Håvard Dagenborg, and Dag Johansen, "Performance Evaluation of Lightweight Stream Ciphers for Real-Time Video Feed Encryption on ARM Processor," *Future Internet*, vol. 16, no. 8, pp. 1-15, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[20] Evaristo José Madarro-Capó et al., "Practical Improvement in the Implementation of Two Avalanche Tests to Measure Statistical Independence in Stream Ciphers," *Computation*, vol. 12, no. 3, pp. 1-13, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[21] Yunqiang Li, and Ting Cui, "Linear Forgery Attacks on the Authenticated Encryption Cipher ACORN-Like," *Chinese Journal of Electronics*, vol. 34, no. 1, pp. 257-265, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[22] Tiberius-George Sorescu et al., "Comparative Performance Analysis of Lightweight Cryptographic Algorithms on Resource-Constrained IoT Platforms," *Sensors*, vol. 25, no. 18, pp. 1-17, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[23] Rashid Muhenga et al., "Comprehensive Analysis of Cryptographic Algorithms: Implementation and Security Insights," *Engineering Proceedings*, vol. 104, no. 1, pp. 1-12, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[24] Xinyue Zhang et al., "Encryption Algorithm MLOL: Security and Efficiency Enhancement based on the LOL Framework," *Cryptography*, vol. 9, no. 1, pp. 1-22, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[25] Stanimir Yordanov et al., "Chaos-based Secure Data Transmission in Intelligent IoT Sensor Networks," *Engineering Proceedings*, pp. 1-18, 2026. [CrossRef] [Google Scholar] [Publisher Link]

[26] Shuang Wei et al., "Probabilistic Shaping-Assisted Bases Precoding in QAM Quantum Noise Stream Cipher," *Photonics*, vol. 13, no. 3, pp. 1-19, 2026. [CrossRef] [Google Scholar] [Publisher Link]

[27] Zhe Yu et al., "Hardware Implementation of HXTS Dedicated Algorithm based on Sector Encryption," *IEEE Access*, vol. 14, pp. 127972-127992, 2026. [CrossRef] [Google Scholar] [Publisher Link]

[28] David Carcaño Ventura, Lil María Xibai Rodríguez-Henríquez, and Saúl Eduardo Pomares Hernández, "Titanium S-boxes: A New Family of 4-Bit Optimal S-Boxes for Feistel-based Lightweight Block Ciphers," *IEEE Access*, vol. 14, pp. 128189-128201, 2026. [CrossRef] [Google Scholar] [Publisher Link]

[29] Dai Watanabe et al., "Enocoro-80: A Hardware-Oriented Stream Cipher," *2008 Third International Conference on Availability, Reliability and Security*, Barcelona, Spain, pp. 1294-1300, 2008. [CrossRef] [Google Scholar] [Publisher Link]

[30] Dai Watanabe et al., "Update on Enocoro Stream Cipher," *2010 International Symposium on Information Theory & Its Applications*, Taichung, Taiwan, pp. 778-783, 2010. [CrossRef] [Google Scholar] [Publisher Link]