IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 31, 2022, accepted May 3, 2022, date of publication May 9, 2022, date of current version May 12, 2022.

*Digital Object Identifier 10.1109/ACCESS.2022.3173338*

# An Efficient and Reliable Chaos-Based IoT Security Core for UDP/IP Wireless Communication

**BENKHADDRA ILYAS[1], (Student Member, IEEE), SENOUCI MOHAMMED RAOUF[2], SENOUCI ABDELKADER[3], TANOUGAST CAMEL[4], SADOUDI SAID[5], AND HANG LEI[1]**

[1]School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu 610054, China
[2]School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China
[3]Faculty of Engineering and Environment, Northumbria University, Newcastle NE7 7YT, U.K.
[4]University of Lorraine, 57070 Metz, France
[5]Ecole Militaire Polytechnique, Algiers 16046, Algeria

Corresponding author: Benkhaddra Ilyas (benkhaddra.ilyas@hotmail.com)

**ABSTRACT** The ultimate focus of this paper is to provide a hyperchaos-based reconfigurable platform for the real-time securing of communicating embedded systems interconnected in networks according to the Internet of Things (IoT) standards. The proposed platform's Register Transfer Level (RTL) architecture is entirely developed and designed from scratch using the VHSIC Hardware Description Language (VHDL). The original idea consists of exploiting the nonlinearity of a discretized and optimized 4D Lorenz hyperchaotic system as an encryption keystream generator in a symmetric cryptosystem to secure wireless communicating embedded systems and adapted to the UDP/IP protocol. It was necessary to go through three essential steps to achieve this goal. First, a lightweight and energy-efficient hyperchaos-based encryption IP core is designed, implemented on an FPGA circuit and dedicated to IoT device security, denoted Hyperchaotic-based IoT Device Security Core (HC-IoT-DSC). The designed encryption IP core combines three subsystems: a multiple key size hyperchaotic key generator (HC-KG), a hyperchaotic synchronization by dynamic feedback modulation technique (HCS-DFM), and an online FIPS 140-2-based built-in self-security test (BISST) module. Second, a secure UDP/IP stack is totally implemented using the VHDL language. Third, the proposed architecture was integrated into real-world and real-time secure wireless communication at a distance of 2 km between two delocalized network nodes employing the Xilinx ML605 FPGA platform and the ZigBee E800-DTU module. A panoply of online/offline investigations and experiments were carried out intensely, deeply, and thoroughly to analyze, evaluate and validate the robustness and security aspects of the proposed scheme regarding all the aspects related to embedded system security. Notably, the evaluations were conducted in two phases for all the platform components before and after integrating the proposed security core in real-time wireless communication. The investigations and implementation findings validate that the proposed architecture can attain good performances, and confirm the feasibility of the adopted approach for IoT applications. Furthermore, the timing and power efficiency results present an excellent trade-off between design performance and high-security achievement.

**INDEX TERMS** IoT security core, hyperchaotic PRNG, RTL design, VHDL, UDP/IP stack, BISST, power efficiency, timing efficiency, ZigBee E800-DTU, secure wireless communication.

## I. INTRODUCTION

Information security is a research topic for which there is currently a strong revival of interest because of the tremendous development of telecommunications in recent years [1]. Nevertheless, security has also naturally appeared because of the popularization of the exchange of confidential information. Indeed, the notion of confidentiality has spread widely from an area that initially only concerned diplomacy, the army, or governments. Thus, confidentiality has become

The associate editor coordinating the review of this manuscript and approving it for publication was Mehdi Sookhak.

necessary for individuals through the trivialization of the exchange of information on large public communication networks such as the Internet of Things [2].

The IoT strings various objects with different models, capabilities, and qualities around the globe. The IoT has simplified daily life by merging the digital and physical eras. The IoT network's rapid expansion and the widespread use of IoT devices blur the line between the digital and physical worlds, exposing vast regions to potentially innovative attacks that traditional cybersecurity measures have not foreseen [3], [4]. In this setting, the major problem is to manage billions of things connected to diverse networks [5], [6]. This variety results in another significant problem called heterogeneity, which requires security to overcome the challenge of deploying efficient cryptographic algorithms and protocols on all IoT ecosystem components [5], [6].

The resource constraints of the IoT and the vast number of deployed and linked devices, which increase the heterogeneity impact and decrease the scalability ratio, complicate, if not make, the direct implementation of advanced security procedures in many circumstances. The absence of authentication and authorization standards for IoT devices fosters malicious attacks on quiet confidentiality attacks on network availability, such as denial-of-service (DoS) attacks [7]. Additionally, security problems significantly influence the safety of IoT devices, and several security concerns must be addressed. Therefore, the integration and interoperability of the IoT with various technologies provide an opportunity to rethink security principles around data collection, storage, and sharing to establish an inclusive, human-centered safe environment [8], [9]. Usually, not all security threats are apparent, and connection might have unexpected implications. Developing dependable and secure real-time systems that make the IoT worthwhile requires a robust security approach that ensures data privacy, confidentiality, integrity, authentication, and identifying and trusting both digital and physical data sources.

Creating a complete set of IoT standards may cover networking, communication, and data management and contribute to general interoperability. Developing designs and prototypes contributes to reducing fragmentation in early IoT systems. More precisely, caution must be used in selecting and developing appropriate solutions. Today's technological considerations will bind the IoT indefinitely, and leak standardization may cause restricting security and usage alternatives. However, this constraint is projected to be eased soon [10]. As actions are taken in the following years, it will be critical to understand the origins of IoT devices and their security. Security by design is a method of developing software and hardware that incorporates security from inception, even if it results in additional expenses, rather than being added after a cyber incidence. The need for security by design has grown critical as technology firms continue to produce a flood of IoT devices for consumers and businesses. Most of these objects were created with no security features, making them ideal candidates for security vulnerabilities. Securing linked devices and using a tiered security strategy is critical, which we refer to as security by the policy. This strategy aims to mitigate security risks using a diversified collection of independent security techniques applied at different levels of the IoT architecture.

The rapid IoT revolution imposes that security mechanisms be continuously inspected, and new security paradigms should be proposed [11], [12]. Among that, we can safely say that the IoT will be a revolutionary technology if we can overcome its weaknesses concerning architecture, standardization, and security. Unfortunately, the most current IoT solutions that rely on conventional cryptography architectures will soon reach their limits and cannot keep up with those complex technical challenges [13]. Therefore, we rationally need new security techniques to address the IoT challenges appropriately. Alternative solutions, utterly different from standard cryptographic techniques, are currently of apparent research interest in this context: chaos-based cryptography has always been the case.

Chaos-based cryptosystems and the IoT have become progressively prevalent since a few years ago [14]–[19]. Many researchers worldwide are now trying to develop new ways of integrating chaos and IoT to create highly secure but robust ecosystems and address technical and other issues. Chaos-based cryptosystems and IoT as standalone architectures have already proved highly disruptive. However, one can easily fall by modifying the architectures without effectively guaranteeing their operation or applying them to scenarios where the cost does not compensate for the improvement. More specifically, the IoT and chaos-based cryptosystem union seem convenient for both but spout various potential security and architectural challenges. Integrating chaos-based cryptosystems and the IoT should be analyzed carefully and taken with high caution to work together successfully. Merging these two systems should be addressed, considering the challenges identified above. Beyond the security aspect, which affects both systems, most research efforts should also be made to ensure high performance regarding all factors of the embedded system, such as energy efficiency and high speed. Therefore, we argue that there is a pressing need for more extensive research into IoT security by applying chaos cryptography.

A dynamical system is chaotic if a significant portion of its phase space simultaneously presents the following two characteristics: the phenomenon of sensitivity to initial conditions and a strong recurrence. These two properties lead to a highly disordered behavior rightly qualified as chaotic. Chaotic systems' unpredictability and sensitivity to initial conditions have attracted the attention of academics involved in information security. Indeed, the chaotic system's unpredictability arises from the fact that a minor change in the initial conditions results in drastically different behavior. This feature may conceal data inside a chaotic signal. The deterministic nature of chaotic systems enables the generation of similar chaotic signals from identical initial values and control parameters. Using two identical chaotic systems as

transmitter and receiver makes it conceivable to envision a communication that uses chaotic events to hide the communicated information. However, synchronization between two chaotic systems is challenging because of the extreme sensitivity to initial values.

In addition to the concepts of chaotic systems, hyperchaotic systems have more degrees of freedom than chaotic systems and are therefore closer to the natural systems they model. Hyperchaotic systems are modeled by no less than four differential equations that induce additional constraints (i.e., other mismatch parameters and other initial conditions) and give two or more positive Lyapunov exponents. Hyperchaotic systems present more complexity, large keyspace, and higher unpredictability than chaotic systems. Therefore, hyperchaotic systems are more suitable for chaos-based cryptosystem applications [20]–[22]. Several hyperchaotic systems have been introduced in the literature, such as the Rössler [23], Lorenz [24], Chen [25], and Liu [26]. The first hyperchaotic system, which the German Otto Rössler proposed, is related to the study of fluid flow; it follows from the Navier–Stokes equations. The mathematical model of this system was discovered as a result of work in chemical kinetics.

Recent studies categorize chaotic or hyperchaotic systems, either integer-order (IO) or fractional order (FO), into two main classes: systems with self-excited attractors (CSSA) and systems with hidden attractors (CSHA) [27]. An attractor is self-excited if its attraction basin includes at least one equilibrium point. Or else, the attractor is defined as hidden. According to [27], self-excited attractors can be identified by applying a simple calculation, making them incapable of resisting attractor reconstruction attacks. Therefore, the CSSA can be easily attacked in secure applications. However, the evaluation of equilibrium points of CSHA is arduous, which complicates the identification and localization of the hidden attractors. Despite this difficulty, suppleness in the system performance without changing parameters can be employed with the correct control techniques to transition between distinct coexisting states [27]. Recently, great attention has been given to modeling, studying, designing, identifying, and controlling CSHA [27].

Chaotic systems in continuous time are represented by a set of ordinary differential equations (ODEs) that have a unique solution. This solution is defined as a trajectory in phase space. It has the property of never going through the same point in this space, i.e., chaotic signals are bounded without periodicity, valid for a physical phenomenon of exorbitant chaos such as the atmosphere. A chaotic system cannot repeat its dynamic evolution since it can be in an infinity of states, and it is impossible to reproduce the same state with exactness. This phenomenon is true in continuous space, but when chaotic systems are discretized with finite precision and applying numerical resolution methods, the probability of revisiting a point in the phase space becomes nonzero. In addition, the results obtained numerically no longer represent the same system of ODEs since they are not defined in the same space even though the global behavior remains identical [27], [28]. Therefore, this property transforms the infinite trajectory of the chaotic system into a set of closed and finite trajectories with different lengths.

Solving chaotic systems is feasible by applying various numerical techniques such as Forward and Backward Euler, Trapezoidal, fourth-order Runge-Kutta (RK4), Adams-Bashforth, and Adams-Moulton. Hardware rounding and truncation, as well as other sources of mistakes (e.g., multistep, variable order, and variable step-size), are all factors that might affect the accuracy and convergence of numerical resolution algorithms [27], [28]. It remains challenging to choose the appropriate numerical approach for addressing a specific ODE issue. A similar difficulty is the estimate of the time-step, intending to achieve the lowest possible error in comparison to the exact solution and the numerical method stability. Indeed, suppose the numerical technique and the time-step are not chosen appropriately. In that case, the solution might diverge, converge, be incorrect, or have other undesirable computing consequences, leading to algorithm instability [27], [28].

In [28], a pertinent work was presented by Martín Alejandro Valencia-Ponce *et al.* It has been proven that the chaotic system behavior can be optimized by estimating the highest integration step in either one or multistep numerical methods. Mainly, the authors realize a stability analysis of several numerical methods by applying three different metaheuristics algorithms. The results confirm that the Kaplan-Yorke Dimension can be maximized while preserving the highest integration step. In the same context, Esteban Tlelo-Cuautle *et al.* in [27] realize a numerical simulation of IO and FO chaotic systems using one-step and multistep methods. The authors confirm that RK4 has the lowest error compared to other methods, and the Forward-Euler method generates the highest error. Therefore, many researchers apply this method, which has a low probability of developing undesired effects such as computational chaos or superstability. Additionally, the authors confirm that the numerical method and the integration step are directly related to the hardware resource consumption and design performance.

Chaos-based encryption suggests a new and efficient way of dealing with the problem of fast and highly secure data encryption. Many methods based on analog circuits are used to implement the chaotic behavior generators and the chaotic attractors associated with specific practical applications, such as switched capacitors or analog complementary metal-oxide-semiconductor (CMOS) technology. However, these methods exhibit some practical difficulties since the component values vary with age, temperature, etc. To overcome this problem, analog implementations can be enhanced using FPAAs to reduce mismatches using commercially available amplifiers. Additionally, one can infer that the design of integrated circuits is a challenge to develop lightweight cryptography applications suitable for hardware security for IoT [27]. Another approach performs digital implementation of chaotic

generators since the problem of parameter mismatch does not exist. It provides accuracy and a significant possibility of integration into the embedded system, allowing for many embedded applications. The originality of this approach is that it will enable low-cost data encryption for embedded systems while still providing a good trade-off between performance and hardware resources. However, digital implementations suffer from the problem of degradation due to the use of finite precision to perform computer arithmetic operations. Esteban Tlelo-Cuautle *et al.* in [27] presented excellent detailed guidelines starting from numerical simulation to FPAAs or FPGAs implementations of IO or FO chaotic systems.

Chaos-based encryption provides a novel and efficient approach to encrypting data securely. Numerous analog technologies, such as switched capacitors or analog complementary metal-oxide-semiconductor (CMOS) technology, are utilized to build chaotic behavior generators and attractors connected with specific practical applications. These approaches, however, provide significant practical issues since component values fluctuate with age, temperature, and so on. To address this issue, analog implementations may be upgraded and enhanced with FPAAs to eliminate mismatches when commercially available amplifiers are used [27]. Another technique is to create chaotic generators digitally, which eliminates the issue of parameter mismatch. It delivers precision and a high degree of integration with the embedded system, enabling a wide variety of embedded applications. This solution is novel because it allows for low-cost data encryption for embedded devices while maintaining an acceptable trade-off between performance and hardware resources. However, digital systems suffer from deterioration due to computer arithmetic operations being performed with limited accuracy. Esteban Tlelo-Cuautle *et al.* offered good extensive guidance in [27] for implementing IO and FO chaotic systems using FPAAs or FPGAs. The authors admit that implementing IO chaotic systems is much less straightforward than implementing FO chaotic systems. Effectively, the difficulty lies in the approximation of the FO derivatives. Notably, the analog implementation of FO systems was realized by means of Laplace transfer functions to approximate the FO derivatives. However, it is necessary to tame the ODEs to obtain system parameters in the same order of electronic components. Unlike when using FPGA technology, the FO systems are implemented in the time domain using numerical methods. In addition to the numerical resolution method and step-size selection, the authors emphasized a colossal and crucial factor to consider during the implementation: numbers arithmetic representation. In fact, contrary to the floating-point arithmetic, using the fixed-point arithmetic leads to consuming less memory bandwidth, providing faster speed, and attending higher power efficiency.

As explained above, a considerable accumulation of pertinent research works in chaos-based cryptosystems recently proposed solving security issues in different applications, such as image encryption [21], [22], [29]–[44], video encryption [45]–[47], watermarking [48], speech encryption [49], [50], PRNGs [51]–[60], secure communications [61]–[67], wireless communication [68], conventional cryptography algorithms [69], [70], and FO analog and digital implementation [20]. Adopting different approaches, the established cryptosystem prototypes were realized on various hardware and software platforms such as microcontrollers, ARM processors, FPGAs, GPUs, and analog circuits [27]. Table 1 presents the related-literature review and analysis. We can reaffirm the vital role and effectiveness of chaos-based cryptosystems in securing information systems. Despite the variety of the offered solutions, they do not comply with the required security level for several reasons. Most proposed cryptosystems presented poor analysis, except for some [20], [27]. In other words, the other works offer just some studies that weakly analyze software, hardware, and security levels since they could be used in cryptosystems for embedded system security. Moreover, some related works are limited to the simulation phase and present only randomness analysis with a statistical test suite applied to nonreal word data. Additionally, because of the high complexity at the architectural level, most of the proposed cryptosystems miss flexibility, reconfigurability, portability, and standardization in most cases. Therefore, updating or adapting these cryptosystems to other platforms or applications is highly complex. Moreover, most proposed cryptosystems have been developed for academic purposes and are unused in practice or real-world applications.

The present paper proposes a hyperchaos-based reconfigurable platform for real-time securing communicating embedded systems interconnected according to IoT standards. The designed platform is a modular RTL architecture fully developed and designed from scratch using the VHSIC Hardware Description Language (VHDL). The originality of the adopted encryption approach uses an optimized 4D hyperchaotic Lorenz system to construct a complex hyperchaotic pseudorandom number generator (HC-PRNG) for generating random data as encryption key matrices. In terms of design, the architectural study focuses on an adaptive layout using architectural optimization techniques to achieve a system embedded in an FPGA chip, which offers portability, easy adaptation, and reconfigurability with no technology constraints. Additionally, to consume less memory bandwidth, provide high throughput, and attend to power efficiency, the hardware implementation uses the 32 bits, fixed-point arithmetic model. To establish a robust trust in the design, our strategy is to adopt multiple layers of security where risks are managed using diverse security mechanisms. On the one hand, this strategy allows sharing security responsibilities between all platform components. On the other hand, to completely isolate key generation and secrets from any software exposure at any point in time (i.e., hardware-based security to obtain strong device protection). Our final solution is nothing more than achieving chaotic, secure wireless communication between two network nodes with the following contributions:

**TABLE 1. Related works review and analysis.**

| Ref. | Proposed design, Approach, Findings |
|---|---|
| [62] | An improved chaos-based text cryptosystem for real-time embedded system applications implemented on a 32 bits microcontroller. An implementation of the proposed cryptosystem with double precision float-point arithmetic is realized on a 32 bits Freescale ColdFire microcontroller using CodeWarrior software and C language. |
| [64] | A lightweight chaos-based image encryption method has been realized on the 32-bit Keil MCB2140 ARM development board. The authors use half of the key bits as initial values and the other half to randomly change the system's initial values. |
| [20] | Propose a detailed FPGA implementation of six fractional-order chaotic systems. The implementation approach is based on the combination of the Grünwald–Letnikov numerical method and the short-memory technique under the 32-bit fixed-point arithmetic. Mainly, the authors employ particular RAM and ROM blocks to design a reconfigurable architecture able to control the number of state variables and the length of memory. |
| [45] | Hardware implementation of a multiwing chaos-based real-time secure video communication system. Integrating two saw-tooth wave functions in the well-known chaotic Lorenz system to construct a new multiwing chaotic system. Euler's method is employed for the system discretization. |
| [50] | Proposed a novel encryption solution for secure audio transmission that uses the fast Fourier transform (FFT) and a new 3D Lorenz-logistic chaotic system. The novel chaotic system generates a new dynamic behavior by combining the conventional 3D chaotic Lorenz and the 1D logistic map systems. |
| [52] | A hardware implementation of six chaotic pseudorandom number generators. This research aims to identify efficient strategies for eliminating or consuming the minimum multipliers and dividers in the hardware design of different pseudorandom number generators. |
| [59] | Suggested a novel generalized pseudorandom number chaotic map based on the Newton complex map. The significant achievements of this research include the proposed model's capacity to create both integer and complex random numbers and its large and dynamic key size, which considerably improves security features. |
| [65] | Proposed an innovative, lightweight, and efficient chaos-based cryptosystem for securing low-resource nodes' network communication systems. The proposed cryptosystem is implemented on an Atmega1281 8 Bit microprocessor. |
| [68] | Proposed the randomness improvement of five chaotic map using mod 255 function. The suggested chaotic-maps PRNGs have been implemented on PIC-microcontroller and employed to realize a lightweight image cryptosystem. A secure M-2-M image communication through ZigBee channels have been conducted to validate the practicability of the proposed scheme. |
| [38] | FPGA implementation of three reliable image encryption algorithms on a VLSI architecture using two reconfigurable 4D hyperchaotic systems and an S-Box. Commonalities in mathematical models of hyperchaotic systems have been used to create a generic modular architecture, and parameters are selected where all multiplications/divisions are achieved using shift operations. |
| [39] | Real-time FPGA implementation of blur detection, compression, and chaotic encryption for image applications. The main aim is to perform blur detection, compression, and image encryption in parallel. The Advanced Encryption Standard (AES) algorithm and the modified Lorenz chaotic PRNG are merged to realize an efficient CBC mode. |
| [41] | Proposed a novel chaotic system to concept an image encryption system in a mobile Raspberry Pi3 model B microcomputer. A modified chaotic system is designed to build an RNG. The RK4 description is implemented using the Python language and the Spyder IDE software to solve the modified chaotic system. |
| [47] | A real-time video encryption algorithm based on a customized AES that integrates Henon's chaotic map. A mix row function, the chaotic Henon function, and Logical XOR are employed in place of the conventional shift row, the subbyte operations, and the multiple rounds of the original AES algorithm, which significantly speed up the encryption-decryption processes. |
| [48] | A completely undetectable nonblind real-time crypto-watermarking method was implemented on FPGA using Xilinx's high-level synthesis (HLS) hardware/software codesign approach. Original IP cores based on Haar Discrete Wavelet Transform (DWT) were designed, tested, and certified by HLS with a new chaos-based key generator. |
| [58] | A VHDL-based design model of a new method for constructing multiwing chaotic systems. The authors suggest that 3D Lorenz's continuous chaotic system can be amended by incorporating saw-tooth and sine functions. The proposed chaotic technique is used to build a novel chaos-based TRNG true random number generator. |
| [61] | Presented a method for secure communication based on a novel five-dimensional hyperchaotic system and its hardware implementation via a microcontroller unit (MCU). The proposed hyperchaotic system is discretized using the Euler resolution approach. The drive-response is an adopted synchronization mechanism. |

- FPGA Design of a lightweight and energy-efficient hyperchaos-based encryption IP core dedicated to IoT device security, termed the Hyperchaotic-based IoT Device Security Core (HC-IoT-DSC). The encryption IP core uses an HC-KG to generate pseudorandom key matrices of different sizes. Additionally, the designed IP core incorporates hyperchaotic synchronization by the dynamic feedback modulation technique (HCS-DFM). To guarantee online and continuous control of randomness quality, system availability, and security reliability, the proposed IP core integrates a FIPS 140-2-based built-in self-security test module (BISST). The BISST ensures four online statistical tests while realizing an environment failure protection and testing mechanism (EFPTM). The EFPTM is implemented in such a way as to guarantee system functioning under three different security levels;
- FPGA design of a secure UDP/IP stack. The proposed stack affords a low design latency and multiple high speeds of 10, 100, and 1000 Mb/s. In addition, the proposed UDP/IP interface is strengthened by many security measurements, such as port number control, MAC address random configuration, private internal static routing table hardware configuration, and IP packet fragmentation deactivation. The goals behind the static routing usage are from one side, to use

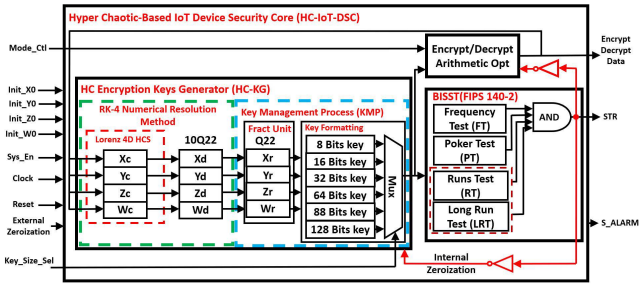


**FIGURE 1.** The implemented cryptosystem functional block diagram.

less bandwidth and to minimize memory and computation resource consumption. On the other hand, this technique enables good security because the routes are always known, and any changes in the network topology require the intervention of a trusted authority. The UDP/IP medium can resist MAC address spoofing and fragmentation attacks;

- Realize real-world and real-time secure wireless communication at a distance of 2 km between two delocalized network nodes employing the Xilinx ML605 FPGA platform and The ZigBee E800-DTU (Z2530-ETH-27) module;
- Online/offline investigations and experiments were carried out intensely, deeply, and thoroughly to analyze, evaluate and validate the robustness and security aspects of the proposed scheme regarding all the aspects related to embedded system security. Mainly, the evaluations were conducted for all the platform components in two phases before and after integrating the proposed security core in real-time wireless communication.

The remainder of the article is structured as follows: Section II details the main steps to implement the proposed security core and the corresponding results and analysis. Then, the design of a secure UDP/IP stack is presented in Section III. Moreover, performance analysis and connectivity tests of the UDP/IP interface are given. Section IV is devoted to realizing secure wireless communication. Furthermore, the realization results, performance details, security analysis, and comparison are discussed. Finally, this research is concluded in Section V.

## II. SECURITY CORE DESIGN AND IMPLEMENTATION

This section presents the developed cryptosystem, a lightweight and energy-efficient hyperchaos-based encryption IP core implemented in an FPGA circuit and dedicated to IoT device security. The proposed security scheme combines three subsystems, an HC-KG (hyperchaotic key generator), a synchronization mechanism, and an online statistical test battery, as shown in Figure 1. Initially, we present the hardware architectures of the random number generator, the statistical tests battery FIPS 140-2 [71], and the adopted synchronization technique. Second, to validate the security aspects of the proposed scheme, several experiments and tests were established to realize an online and offline evaluation of the proposed architecture. Finally, we conclude this section by presenting the obtained results and related interpretations.

### A. HYPERCHAOTIC ENCRYPTION KEY GENERATOR (HC-KG)

A cost-effective and optimum technique for constructing a hyperchaotic embedded system is to develop a customized hardware architecture that is compatible with a digital numerical resolution method. We may mention both one-step and multistep numerical resolution methods (i.e., Euler, Runge-Kutta, Adams-Bashforth, Adams-Moulton, etc.). In contrast to the Euler method, which is a numerical process for solving first-order differential equations for a given starting conditions, the RK4 method yields the most accurate solutions [20], [27], [28]. Indeed, in numerical analysis, the RK4 technique is an iterative numerical approach in that the initial estimate of the solution is used to produce a somewhat more exact second estimate, and so on [27], [72]. Additionally, as explained earlier in the introduction section, the hardware performance strongly depends on the numerical method, step-size, and selected arithmetic representation. In this regard, and considering our embedded ciphering application, the hardware implementation is developed and written in VHDL using structural description logic. This low-level design approach aims to resolve an optimized 4D Lorenz hyperchaotic system (1) using the RK4 numerical resolution method to provide a more accurate approximation of the solution and fulfill the needs of onboard applications in terms of physical resource usage, power efficiency, and speed.

$$\begin{cases} \dot{x} = a(y - x) \\ \dot{y} = bx - y - xz + w \\ \dot{z} = xy - cz \\ \dot{w} = -dx \end{cases} \tag{1}$$

with mismatch parameters $a = 10$, $b = 28$, $c = \frac{8}{3}$ *and* $d = 5$ and the initial condition values $x_0 = y_0 = z_0 = w_0 = -10$.

#### 1) MATHEMATICAL MODELING OF THE RK4 METHOD

The hyperchaotic Lorenz system is described by the nonlinear equations system as follows:

$$\begin{cases} \dot{x} = F(x, y, z, w) \\ \dot{y} = G(x, y, z, w) \\ \dot{z} = Q(x, y, z, w) \\ \dot{w} = U(x, y, z, w) \end{cases} \tag{2}$$

where $x(t_0) = x_0$, $y(t_0) = y_0$, $z(t_0) = z_0$ *and* $w(t_0) = w_0$ Moreover, $F$, $G$, $Q$ *and* $U$ are nonlinear functions. To solve the nonlinear equation system (2), the RK4 technique uses the following equation system:

$$\begin{cases} x_{n+1} = x_n + h/6(k_0 + 2k_1 + 2k_2 + k_3) \\ y_{n+1} = y_n + h/6(m_0 + 2m_1 + 2m_2 + m_3) \\ z_{n+1} = z_n + h/6(l_0 + 2l_1 + 2l_2 + l_3) \\ w_{n+1} = w_n + h/6(p_0 + 2p_1 + 2p_2 + p_3) \end{cases} \tag{3}$$

where $h$ is the discretization step and is $h = 0.001$, and different derivatives are $k_i$, $m_i$, $l_i$ *and* $p_i$ with $i = (0, 1, 2, 3)$.

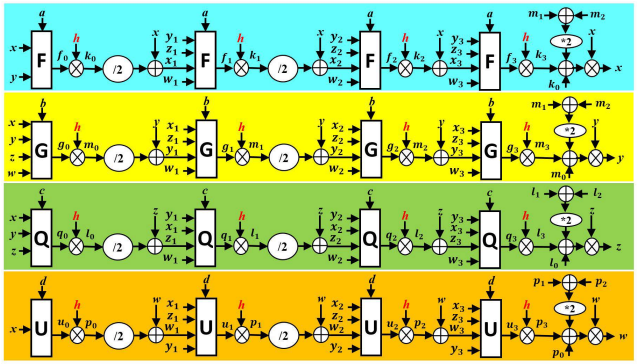

**FIGURE 2.** RTL-Design of the RK4 method to implement the 4D Hyperchaotic Lorenz system.

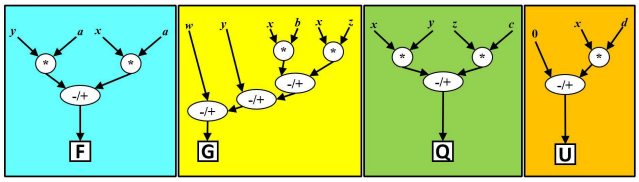

**FIGURE 3.** RTL-Design of the ODEs of the 4D Hyperchaotic Lorenz system.

The $k_i$ derivatives are defined as follows:

$$\begin{cases} k_0 = F(t_n, x_n) \\ k_1 = F(t_n + h/2, x_n + h/2k_0) \\ k_2 = F(t_n + h/2, x_n + h/2k_1) \\ k_3 = F(t_n + h, x_n + hk_2) \end{cases} \tag{4}$$

where $x_n$ is an arbitrary starting point chosen at an arbitrary $t_n$, $k_0$ is the derivative function at the start of the integration interval, $k_1$ and $k_2$ are the derivatives at the middle of the integration interval, and finally, $k_3$ is the derivative at the endpoint of the integration interval. Applying the same analogy, the derivatives $m_i$, $l_i$ and $p_i$ are computed. Figure 2 depicts the RTL design that implements the RK4 numerical method. The proposed architecture depends on control parameters a, b, c, and d, the integration step $h = 0.001$, and functions F,G,Q and U as shown in Figure 3.

It should be noted that continuous Lorenz hyperchaotic variables are real numbers. To better compromise performance and cost, fixed-point arithmetic is used in a 32 bits format (10Q22). More precisely, 10 and 22 bits encode the integer and fractional parts, respectively. We have replaced the multiplication and/or division operations (i.e., the most used operations by the RK4 method) with right and left shifting operations to reduce FPGA resource utilization. Consequently, the proposed design uses only six additions and seven multiplications per clock cycle. Therefore, we minimize the number of DSP modules, and only basic arithmetic operations are used in our implementation.This architectural optimization minimizes the consumed logic slices, leading to a high throughput rate and a low design latency.

### 2) HARDWARE IMPLEMENTATION OF THE HC-KG

Knowing that implementing chaotic generators on FPGA suffers from the finite precision problem, which causes losses in the natural chaotic dynamics of the implanted generator, we propose a very original solution to thwart this problem (see Figure 1, HC Encryption Keys Generator). The basic idea of our solution is based on increasing the length of fractional parts of data while reducing that of integer parts, and we only consider fractional parts to construct the encryption keys, hence the designation HC-RNG (Hyperchaotic Random Number Generator) of the proposed hyper chaos-based cryptographic key generator. Equations system (5) describes our approach for generating random keys:

$$\begin{cases} x_r = fract(x) \\ y_r = fract(y) \\ z_r = fract(z) \\ w_r = fract(w) \end{cases} \tag{5}$$

where $fract\ (u) = (u| - En((u|)$ and $En\ ((u|)$ represents the integer part of $u$.

The real-time implementation is realized utilizing an FSM comprising four states (see Figure 4). A random key of 88 bits (11 bytes) is generated. Then, judging from the latter, the KMP (Key Management Process) is launched in parallel to construct the other keys in different sizes (8, 16, 32, 64, 88, and 128 bits). The operations performed in each state are as follows:

- **Initialization**: this operation is realized initially and not included in the FSM. This means that the hyperchaotic system outputs are set to zero, and the initial conditions are assigned. We use an asynchronous reset signal during this phase, active in the low logic state (reset = '0'), to put the whole generation process in an idle status by forcing all system parameters to their initial values. We prepare for the first integration step in the RK4 method. The triggering of the key generation is conditioned by the external signal Sys_En.
- **State 1**: Calculation of the initial derivatives $k_0$, $m_0$, $l_0$ *and* $p_0$ (Equation (1), system (4)) as well as the intermediate points. The machine unconditionally switches to the State2 state at the next clock edge.
- **State 2**: Calculation of the derivatives $k_1$, $m_1$, $l_1$ *and* $p_1$ at the mid integration interval (Equation 2, system (4)) and the intermediate points. The machine unconditionally switches to the State3 state at the next clock edge.
- **State 3**: Calculation of the derivatives $k_2$, $m_2$, $l_2$ *and* $p_2$ at the mid integration interval (Equation 3, system (4)) and the intermediate points. The machine unconditionally switches to the State4 state at the next clock edge.
- **State 4**: Calculation of the final derivatives $k_3$, $m_3$, $l_3$ *and* $p_3$ (Equation 4, system (4)) and the final intermediate points. In this step, if the requested key is achieved, we stop the generation and return to the initial status; otherwise, we switch to State 1. In this state, all computed derivatives and the final intermediate points are delivered to the KMP from one hand to calculate the final solutions (Equation (3)) and from the other hand to extract the fractional parts (Equation (5)) and construct cryptographic keys.

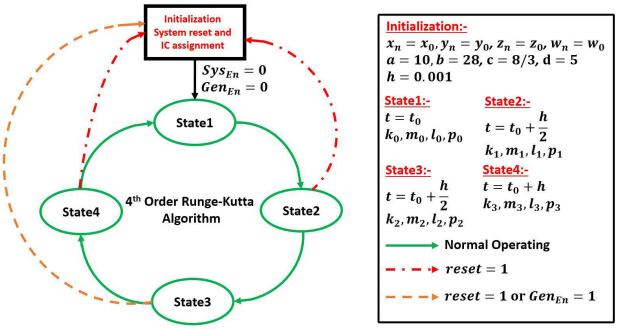


**FIGURE 4.** The implemented FSM of the RK4 method.

It is worth mentioning that it was possible to use another FSM state to realize all tasks performed by the KMP. However, for optimization reasons and to give more architectural flexibility to the proposed keys generator, it was preferable to add this process that runs parallel with the RK4 FSM. The advantage of this choice is the possibility of generating a multisize key (8, 16, 32, 64, 88, and 128 bits) in only four (04) clock cycles. Conversely, [73]–[75] implement the same system and use the same resolution method, but the keys are delivered in mono-size format after 10, 8, and 6 clock cycles. In addition to ensuring continuous randomness quality verification and control, the random number generator is attached to an onboard FIPS 140-2 test battery.

Based on the results of the test, three operating modes are envisaged. The first mode is the standard system operating with security level 1 (the generated key passes all the tests). The second mode is operating under security level 2. The generated keys fail FIPS 140-2 tests less than five times in this situation. Then, an internal zeroization process puts the generation and encryption processes in the initial state (back forward to initialization, reassigning the initial conditions' values, and waiting for Sys_En to restart the generation and encryption/decryption operations), and the generated keys are systematically rejected. The third mode is the system dysfunction mode under security level 3, where the generated keys fail the tests over five times. In this mode, to protect the secret parameters of the key generator (initial conditions, values, and mismatch parameters) and the plaintext, a security alarm is generated, and an external interruption is necessary. The latter activates a finite loop to randomly change the secret parameters (physical formatting), eliminating all attempts to recover these parameters. Furthermore, the only way to quit this mode and return to normal operating mode is the FPGA board reconfiguration (reloading the FPGA configuration file). The next subsection details the development and implementation of the proposed FIPS 140-2.

### B. HARDWARE IMPLEMENTATION OF THE BISST MODULE

Several sets of statistical tests exist in the literature; the best known are those of NIST SP800-22 [76], [77], DIEHARD [78], AIS20/31 [79]–[81], TestU01 [82], NIST SP800-90B [83], ENT [84], and FIPS 140-2. TestU01 is the most complete and difficult RNG test suite, including eight subbatteries with over 282 statistical tests. Excluding FIPS 140-2, the other tests' hardware implementation is very complicated and consumes high physical resources. This difficulty is because of the use of several very complex mathematical functions. Thus, in our work, four standard statistical tests are selected: the frequency test (FT), Poker test (PT), Runs test (RT), and Long-Run test (LRT). These tests are used in the FIPS 140-2 cluster, and they are included in the AIS20/31, NIST SP800-22, and TestU01 tests. Although FIPS 140-2 is less stringent than other statistical tests, its hardware implementation is very efficient, making it the best candidate for onboard and constrained resource applications. Mainly, these tests are applied to a binary sequence of 20,000 bits and can be summarized in two steps: a calculation of a statistical quantity and a comparison of the latter to a predefined decision interval. We cannot find a clear and detailed development about the processes followed to obtain the decision intervals in the literature. The following subsection details each of the four tests.

#### 1) THE PROPOSED ARCHITECTURE OF THE BISST MODULE

The FIPS 140-2 is used as a statistical test battery and, at the same time, as an environment failure protection and testing mechanism (EFPTM). In other words, the FIPS 140-2 is used as a BISST (built-in self-security test) that guarantees the reliability and availability of our cryptosystem. The hardware implementation is provided in three processes, the frequency test, Poker test, and run test combined with the longest run in the third process. The FIPS 140-2 module takes the generated keys, constructs a binary sequence of 20, 000 bits, and outputs STR and S_ALRM flags. The STR is the statistical test result indicator (i.e., equals '1' if the tested sequences pass all tests called security level-1). Additionally, the STR is used to launch an internal zeroization process to reinitialize the keys generator and end the data encryption if the tested sequences fail the statistical tests less than five times (called security level-2). S_ALARM is a security alarm generated when the sequences fail the tests over five times, which causes system dysfunction, and the key generator stops operating. Here, a zeroization process is activated using an external interruption signal (security level-3).

##### *a: FREQUENCY TEST*

The form of the FSM (see Figure 5) comprises four states. During IDLE, we wait for data, and we switch to the next state (Read) only if the reset signal equals '0' and data are available at the input. In this state, all outputs and internal signals are set to zero. The read state uses two counters, the $D_{in}\_CT$ counter to count the number of bits in the binary sequence and the *ones_CT* counter to count the number of ones in the same sequence. Each time a data bit is read, the counter $D_{in}\_CT$ is incremented and compared to the value of 20, 000. If $D_{in}\_CT$ equals 20, 000, and the reset is '0', then we pass to the Result state; otherwise, we keep incrementing until we reach the value 20, 000. In parallel, the

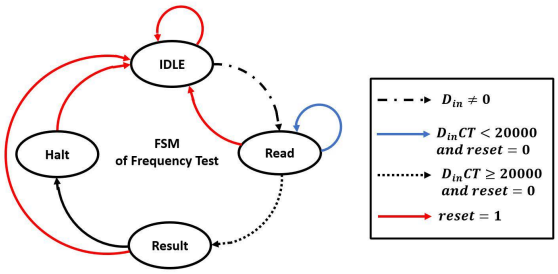


**FIGURE 5.** The implemented FSM of the frequency test.

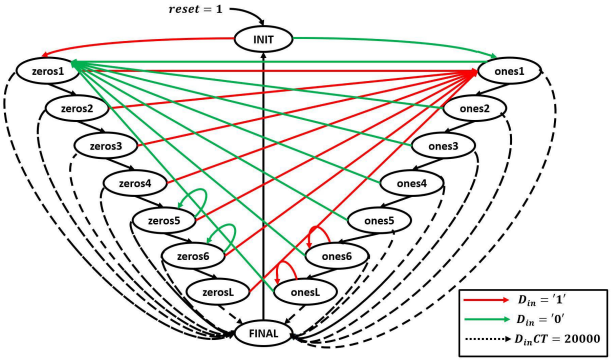


**FIGURE 6.** The implemented FSM of the Run and Long Run tests.

counter *ones_CT* is only incremented if the data bit is '1'. In the results, *ones_CT* is compared to the interval decision. Therefore, if the relation $9,725 < Ones\_Ct < 10,275$ is verified, the test is passed, and the resulting output is set to '1'. Otherwise, the sequence fails the test, and the result is set to '0'. If reset is '0', we pass to the next state Halt. Finally, in the Halt state, if the reset is '1', we set both $D_{in}\_CT$ and *ones_CT* to zero, and we pass to state IDLE.

### b: RUNS AND LONG RUN TESTS

The same algorithm implements both runs and long-run tests. Figure 6 shows the FSM diagram of these two tests. In total, there are 16 states: an initialization state INIT, 14 states for counting each type of Run (onesones1, ones2…ones6, onesL and zeros1, zeros2…zeros6, zerosL), and a final state Final. All system outputs and internal signals are zero during the INIT state. The data's bit value conditions the transition to the next state. If the latter is '1', the next state is ones1; otherwise, we transition to zeros1. The INIT state is reached in two cases: either an asynchronous reset is set to '1' at any step of the test or a return from the final state FINAL. In-state ones1, two counters are incremented; $D_{in}\_CT$ and $ones1\_CT$, then we compare $D_{in}\_CT$ to 20, 000. If this comparison is verified, we transition to the FINAL state. Otherwise, we switch to either ones2 (data's bit is '1') or zeros1 (data's bit is '0') and so forth. If the data value keeps the value '1', unconditional switching from ones2 to onesL until the appearance of a value '0' where we pass to zeros1. Similarly, unconditional switching is realized from zeros1 to zerosL if data keep the value '0' until a '1' appearance, so the next state is ones1, and the same process is repeated.

We realize the runs test during states ones1 to ones6 and zeros1 to zeros6. Parallel to the incrementation of $D_{in}\_CT$, another counter $onesi\_CT$, and $zerosi\_CT$ are incremented with $i \in \{1, 2, \ldots 5\}$ to define the number of ones and zeros in the tested binary sequences, respectively. The transition from one of these states to the FINAL state is realized only if $D_{in}\_CT = 20,000$. In states ones6 and zeros6, both the long-run and run tests are implemented. If the data's bit holds the same value ('1' or '0'), a counter $L\_CT$ is incremented until we obtain 26, then we switch either to the onesL state (data's bit has held '1') or zerosL state (data's bit has held '0'). Otherwise, two counters are incremented $ones6\_CT$ and $zeros6\_CT$ to continue the execution of the Run test. For both situations, if $D_{inCT} = 20,000$, we pass to state FINAL. A decision is made over the state Final by comparing all obtained counters' values with predefined intervals.

### c: POKER TEST

The FSM diagram of the Poker test is similar to that of the frequency test, with four states. In addition, we use 17 counters, $D_{in}\_CT$, to count the number of bits in the binary test sequence and 16 counters to count the number of occurrences of each 4-bit chunk combination ($CT0$, $CT2$, …, $CT15$). During the IDLE state, we wait for data, and we switch to the next state (Read) only if the reset signal equals '0' and data are available at the input. All outputs, internal signals, and counters are set to zero in this state. In Read, we read the input by a chunk of 4 bits. To this end, a 4-bit register is used to save the input bits. Once the register is full, according to the possible combinations, a test is performed to increment the value of one of the counters $CT0$, $CT2$, …, $CT15$. At the same time, a test is carried out on the value of $D_{in}\_CT$ compared to 20, 000. If this is the case, then we transition to state Result. The result compares *ones_CT* to the interval decision. Therefore, if the relation $9,725 < Ones\_Ct < 10,275$ is verified, the test is passed, and the resulting output is set to '1'. Otherwise, the sequence fails the test, and the result is set to '0'. If reset is '0', we pass to the next state Halt. We compute the quantity given by formula (6) in this state. Therefore, the sum of the squares of the counters' values ($i \in \{0, 2, \ldots, 15\}$) is calculated. A comparison of the obtained values is performed with formula (7). Based on the comparison result, a decision is made. If '0', we switch unconditionally to state Halt. In Halt, if the reset is '1', we set all counters and pass them to state IDLE.

$$\begin{cases} \chi^2 = \dfrac{16}{k}\left(\sum_{i=0}^{i=15}(n_i)^2\right) - k \end{cases} \tag{6}$$

$$\begin{cases} 2.16 < chi^2 < 46.17 \\ 1563175 < \sum_{i=0}^{i=15} n_i^2 < 1576929 \end{cases} \tag{7}$$

## C. THE HCS-DFM SYNCHRONIZATION METHOD

In our implementation, only techniques that consider the addition of user data can be used. CS-DFM (Chaotic

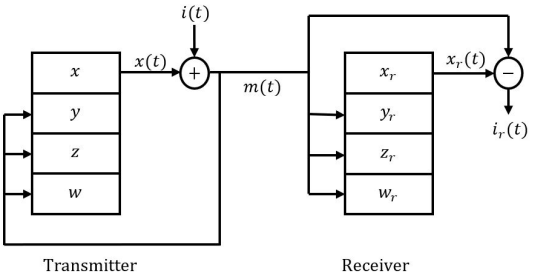


**FIGURE 7.** The implemented HCS-DFM synchronization scheme.

Synchronization by Dynamic Feedback Modulation), developed by V. Milanovic et M.E. Zaghloul in [85] is part of this class. In this subsection, we present our approach based on CS-DFM. We apply the same principle of CS-DFM but with an extension to hyperchaotic systems; therefore, it is called HCS-DFM (the H for Hyper). The difference is that HCS-DFM synchronization is more dedicated explicitly to hyperchaos secure communication. Therefore, the synchronization uses a coupling between two identical hyperchaotic systems in Master-Slave configuration. Indeed, the signal transmitted to the slave system is a mixture of two signals, the information (plaintext) signal and the chaotic keys generated by the master system. This informational mixture is transmitted to the slave system and reinjected (feedback signal) to the master system. The objective is to create the same chaotic dynamic in both Master and Slave systems. Accordingly, the same key used for the encryption is generated by the Slave system. In the same way, the master and slave systems are decomposed into two subsystems (1 and 2) to achieve this configuration, as shown in Figure 7. Thus, to recover the information, it suffices to perform the inverse ''mixing'' operation between the coupling signal and the decryption key generated by the slave system after synchronization. Note that this configuration can use any hyperchaotic signals or combinations to create the encryption/decryption keys. In all cases, synchronization is ensured. To verify this method, in what follows, we detail the mathematical modeling and dynamic error stability proof of the proposed synchronization technique. Note that we assume the absence of noise in the transmission channel, and the mixing operation is a logic XOR.

We use the Lorenz hyperchaotic system defined by systems (1), (8) and (9) to describe the transmitter and receiver dynamics, respectively.

$$\begin{cases} \dot{x} = a(y - x) \\ \dot{y} = (b - z)m(t) - y + w \\ \dot{z} = m(t)y - cz \\ \dot{w} = -dm(t) \end{cases} \tag{8}$$

$$\begin{cases} \dot{x}_r = a(y_r - x_r) \\ \dot{y}_r = (b - z_r)m(t) - y_r + w_r \\ \dot{z}_r = m(t)y_r - cz_r \\ \dot{w}_r = -dm(t) \end{cases} \tag{9}$$

with $m(t) = i(t) + x(t)$, where $x(t)$ is the chaotic signal and $i(t)$ is the plaintext.

*Theorem:* The dynamic error is stable if and only if we introduce the information $i(t)$, the systems (8) and (9) must be globally asymptotically stable in the vicinity of the origin.

*Proof:* The dynamic error between the two systems is defined by $e = T - R$, where $T = (x,\ y,\ z,\ w)$ and $R = (x_r,\ y_r,\ z_r,\ w_r)$.

As announced, it is assumed that there is no noise, and that the signal $m(t)$ arrives at the receiver without distortion. Accordingly, the parameters of the two systems are equivalent. The dynamics of the error between the two systems are then given by:

$$\begin{cases} \dot{e}_1 = a(e_2 - e_1) \\ \dot{e}_2 = m(t)e_3 - e_2 + e_4 \\ \dot{e}_3 = m(t)e_2 - ce_3 \\ \dot{e}_4 = 0 \end{cases} \tag{10}$$

Let $E(e, t)$ be the Lyapunov function, such that the term $m(t)$ disappears.

$$E(e, t) = \frac{1}{2}\left(\frac{1}{a}{e_1}^2 + {e_2}^2 + {e_3}^2 + {e_4}^2\right) \tag{11}$$

The derivative of the Lyapunov function gives

$$\begin{aligned} \dot{E}(e, t) &= \left(\frac{1}{a}e_1\dot{e}_1 + e_2\dot{e}_2 + e_3\dot{e}_3 + e_4\dot{e}_4\right) \\ &= -ce_3{}^2 - \frac{1}{2}e_2{}^2 - \left(e_1 - \frac{1}{2}e_2\right)^2 - e_2\left(\frac{1}{4}e_2 - e_4\right) \end{aligned} \tag{12}$$

Parameters $a$ and $c$ are defined to be positive. When $t \to \infty$, the committed errors are close to zero or zero; then, the following product $e_2(\frac{1}{4}e_2 - e_4)$ is close to zero. Therefore, the derivative $\dot{E}(e, t)$ is negative definite. This allows us to say that: $e_3 \to 0$, $e_2 \to 0$, $e_1 - \frac{1}{2}e_2 \to 0$ then $e_1 \to 0$. According to BARBALAT's lemma (Lyapunov extension) [86], when $t \to \infty$ in system (8), $\dot{E}(e, t)$ is uniformly continuous once the variables of the system converge. We find that $e_1 = \dot{e}_1 = 0$, $e_2 = \dot{e}_2 = 0$, and $e_3 = \dot{e}_3 = 0$, so $e_4 = \dot{e}_4 = 0$. Finally, because the Lyapunov function is positive and its derivative is negative definite, we see that the error $e(t)$ converges to zero (when $t \to \infty, e(t) \to 0$), and therefore, the transmitter and receiver system is globally asymptotically stable and synchronized for any data type.

## III. DESIGN OF A SECURE UDP/IP STACK

There is a real need to design our communication medium within the conformity and adherence to a certain number of international standards in terms of good Quality of Service (QoS) and high performances (i.e., throughput, low area, flexibility, reliability, and simplicity) and security design. From this point, an RTL design of a secure UDP/IP stack is fully implemented using VHDL language. The choice of the VHDL language was taken to achieve, on the one hand, a flexible and reconfigurable architecture (i.e., the concept of

modular design), which gives the possibility of easy adaptation in the future without dependability to the hardware target technology, and on the other hand, a minimum of risk in terms of security issues.

Mainly, our approach was inspired by the design supplied by Xilinx given in Virtex-6 FPGA Embedded Tri-Mode Ethernet MAC UG800 [87]. The Xilinx design only implemented functional verification without experimental performance evaluations or analysis. Therefore, our contributions are as follows:

- Give a detailed architecture with all points that could be modified or subject to possible optimization.
- Provide experimental tests, performance evaluation, and metrics analysis concerning occupied physical area or hardware resources consumption, throughput, and protocol efficiency.
- Adapt and integrate the UDP/IP stack into the proposed security core for IoT devices.
- One of our important objectives is to achieve the simplicity and clarity of the design within compliance with international standards. In our work, the implantations are carried out based on the criteria defined in the Open Systems Interconnection (OSI) [88]. The User Datagram Protocol (UDP) is appropriately represented using the IETF RFC 768 [89]–[92]. Additionally, the Internet Protocol version 4 (IPv4) is implemented as explained in the IETF RFC 791 [93]. Furthermore, the Address Resolution Protocol (ARP) IETF RFC 826 [94] is used for the ARP Core, while the Ethernet protocol IEEE 802.3 [95] is exploited to implement the Ethernet MAC. Figure 8 illustrates the relationship between the OSI model and our UDP/IP architecture for further details.

Figure 9 represents the block diagram of the implemented UDP/IP stack from the physical layer to the user interface. It mainly consists of a user interface (UI) and a UDP module composed of UDP_TX, UDP_RX, and IPv4 modules. This latter comprises four subblocks IPv4_TX, IPv4_RX, ARP, and Tx_arbitrator. The IPv4 Core allows us to encapsulate (multiplex and demultiplex) the UDP datagram in the IP packets and vice versa to pass through the MAC layer of the FPGA platform. The Tx_arbitrator module allowed us to control and manage access to the MAC_TX channel when the ARP and IPv4 modules request a transmission simultaneously. The ARP module reads the MAC_RX data parallel to the IPV4_RX path. Subsequently, it manages the communication of ARP requests and the timeout if no response is received. The IPv4 module is developed to ignore every packet except the following: IPv4 packets, broadcast packets, and packets intended for our IP address. Once all these verifications are satisfied, the received header data are valid, and the module asserts the start of the reception.

The MAC interface is relatively straightforward, with separate clocks for the receiver and the transmitter. Each interface (RX and TX) has an 8-bit data bus. On the one hand, this interface is used to communicate with the ARP module using the AXI bus, and on the other hand, it is used to interface with the modules transmit engine and receive engine. In addition,

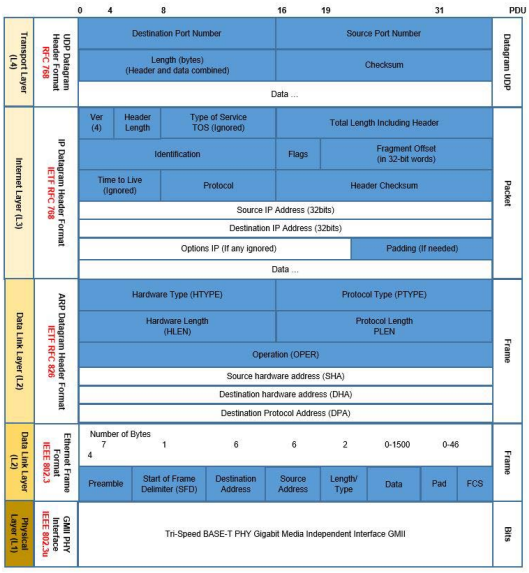


**FIGURE 8. The relationship between the OSI model and our UDP/IP architecture.**

the Xilinx MAC connects to the external ethernet PHY (copper) via a Gigabit Media Independent Interface (GMII). Note that the MAC wrapper is fundamentally provided with more subblocks, but to adapt it to our requirements and for optimization reasons, we have removed all modules that we do not need in our application.

The UI shown in Figure 9 is used as a driver module for the UDP/IP core to control transmission and reception processes, the IP and MAC address configuration, and ensure the traffic routing path employing an internal private routing table. In other words, this interface plays the role of an API in the application layer that allows communication with and controls the UDP stack. The first process is used for the static configuration of the FPGA's IP address. As a security measure, the MAC address of the platform changes randomly. The objective of this action is to avoid MAC spoofing attacks. In parallel with this process, an FSM is used to manage the transmission and reception of the data. First, we start UDP header capture and cannot send data. Second, if the reception is finished with validation, we pass it to user data transmission; otherwise, we must wait until the reception is completed. Finally, the transmitted data are routed according to a specific predefined routing table. The goals behind the static routing usage are from one side, to use less bandwidth and to minimize resource consumption (memory and computation). On the other hand, this technique gives the possibility of having good security because the routes are always known, and any changes in the network topology require the intervention of a trusted authority.

## IV. APPLICATION TO WIRELESS COMMUNICATION ENCRYPTION

This section presents our final solution, which is nothing more than achieving chaotic secure wireless communication

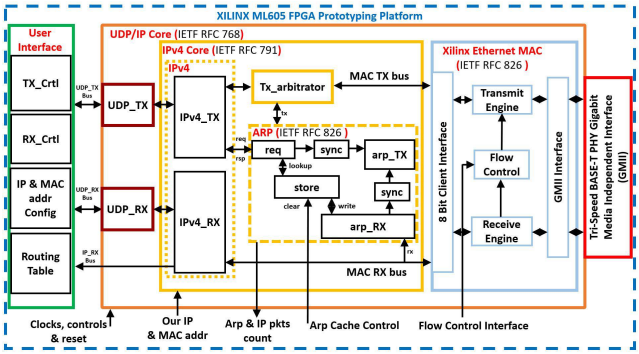


**FIGURE 9.** The implemented UDP/IP stack functional diagram.

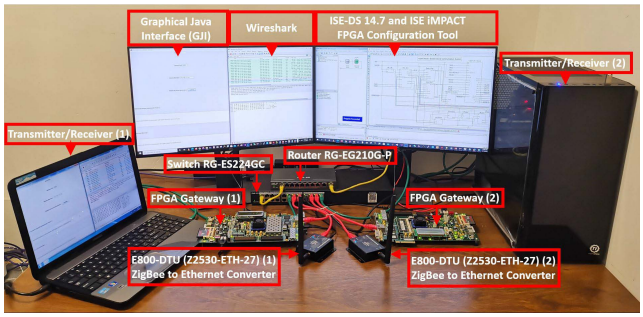


**FIGURE 10.** Experimental setup of the proposed hyperchaos-based secure wireless communication.

between two network nodes with the following main requirements:

- The adopted encryption solution is based on the use of the HC-KG for the generation of pseudorandom data as key encryption matrices;
- HCS-DFM is the chosen synchronization method;
- Use the FIPS140-2-based BISST to guarantee online and continuous control of the generated stream key randomness quality and the security core reliability;
- To establish robust trust in the design (i.e., security by design), the main philosophy is to completely isolate key generation and secrets from any software exposure at any point in time (hardware-based security to obtain strong device protection);
- The UDP/IP stack is the selected communication interface;
- Offer an encryption IP core that can be interfaced with any IP (Internet Protocol) network.

These requirements could only be achieved with several tools and methods. In particular, the hardware-software codesign approach makes the overall architecture flexible and easily reconfigurable. Figure 10 represents the experimental setup of the wireless secure communication system, which is composed of:

1. A development ASUS workstation and TOSHIBA Laptop, containing the following tools:
   - ISE-DS 14.7 software environment for VHDL programming and generation of configuration bitstreams;
   - iMPACT tool for FPGA configuration;
   - Eclipse IDE 2020-09 for Java development of the graphical interfaces to communicate between the PC and the FPGA board;
   - Wireshark software allows network packet analysis. This tool uses GTK+ software for its user interface implementation and pcap for packet capture [96];
2. Two XILINX ML605 development and prototyping platforms based on Virtex6-XC6VLX240T;
3. Two ZigBee to ethernet modules, Ebyte E800-DTU (Z2530-ETH-27);
4. Network router RG-EG210G-P;
5. Network Switch RG-ES224GC.

### A. CRYPTOSYSTEM FUNCTIONAL ARCHITECTURE

The developed cryptosystem architecture is shown in Figure 11, comprising two parts: hardware and software. The hardware part includes the chaos-based cryptographic key generator (HC-KG), key management module (KMM), data management module (DMM), encrypt/decrypt module (EDM), TX/RX Ctrl module (TRCM), UDP/IP stack, FIPS 140-2 (BISSTM) and Clocks generator module (CGM). The software part is a graphical Java interface (GJI) that consists of seven subsections, local port number (LPN), remote port number (RPN), remote machine IP address (RMA), TX/RX UDP socket, validate configuration (VC), sending the message (SM) and received message (RM).

1. **In a hyperchaotic key generator (HC-KG) and HCS-DFM method**, the chaotic key generator has two main functions: the generation of chaotic encryption keys matrices and the hyperchaotic HS-DFM synchronization method. The architecture of the chaotic key generator is developed based on the architecture represented by the previous section, except that its integration into the overall cryptosystem requires some modifications. These modifications are described in the addition of two states to its implemented FSM, as shown in Figure 4. The first state is added to avoid starting the generation of the encryption keys before achieving synchronization and fully receiving the message. The second state is added to manage different security issues when internal or external zeroization is needed;
2. **Key management module (KMM)**: The key management module handles the key matrix formatting and key size selection;
3. **Data management module (DMM)**. This module ensures the interfacing of the cryptosystem with the TRCM module and transferring received text to encryption or decryption operations;
4. **Encrypt/Decrypt module (EDM)**; The EDM module implements two cryptographic modes, encryption and decryption modes. Mainly, this module uses an exclusive OR (XOR) to couple or decouple encryption and text;
5. The **TX/RX Ctrl Module (TRCM) and UDP/IP** stack allow scheduling all data transmission and reception operations. The modules in question are realized using the architecture detailed in section II. Note that we have kept the same architecture, except some modifications have been

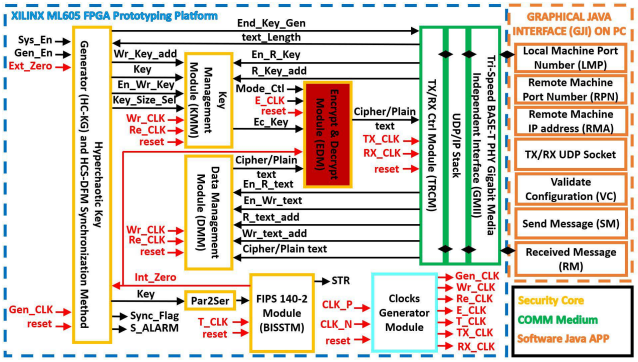


**FIGURE 11.** Block diagram of the proposed cryptosystem.

**TABLE 2.** UDP packet routing table.

| Device | IP address |
|---|---|
| ASUS workstation | 192.168.110.249 |
| TOSHIBA Laptop | 192.168.110.12 |
| FPGA Gateway 1 | 192.168.110.100 |
| FPGA Gateway 2 | 192.168.110.101 |
| E800-DTU ZigBee 1 | 192.168.110.83 |
| E800-DTU ZigBee 2 | 192.168.110.43 |

introduced to interface with the proposed hyperchaotic cryptosystem. The TRCM response module is developed using two processes. The first is a combinatorial process that implements a Moore-type finite state machine that manages the transmission/reception of data and the generation of control signals for the KMM and DMM modules. The second is a sequential process that serves as a control unit for the first process and the routing of UDP packets presented in Table 2 and Table 3.

### B. ENCRYPTION AND DECRYPTION PROCESS

Figure 12 shows the encryption/decryption flowchart of the developed cryptosystem. The whole process consists of five (05) steps as follows:

- **Step 1**: Establishing the cryptosystem initialization by fixing the HC-KG initial values and control parameters;
- **Step 2**: Before any message exchange, a synchronization test is realized. If the two FPGA gateways are synchronized, the process goes to the encryption key generation step, and a synchronization flag is activated to start message exchange; otherwise, we wait until synchronization is achieved.
- **Step 3**: In parallel with message reception, encryption/decryption key generation is launched;
- **Step 4**: During the encryption/decryption phase and according to the selected operating mode, the generated key in step 3 is used to either encrypt or decrypt the received message;
- **Step 5**: In this step, another test is performed to compare the lengths of the generated key and the received message. If they are equal, we stop key generation, and the treated message is transmitted.

Note that if FIPS 140-2 (BISST) detects a system anomaly at any step of the encryption/decryption process, we should

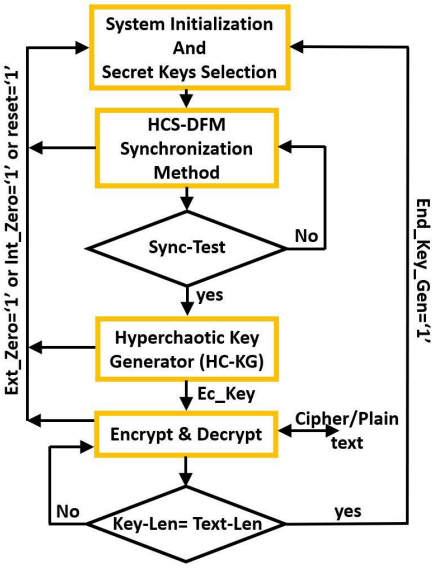


**FIGURE 12.** The encryption/decryption flowchart of the proposed cryptosystem.

return to the initialization step, and all operations are interrupted.

### C. HC-KG SIMULATION RESULTS

Before the real-time measurement of our solution, we carried out two types of simulations. The first one is a MATLAB simulation, which aims to investigate and validate the chaotic aspect of the proposed HC-KG. To achieve this, the HC-KG is digitized by using the RK4 resolution method. Then, a bifurcation analysis is conducted to verify the behavior of the HC-KC and compare it to the original 4D Lorenz chaotic system. Moreover, the dynamical degradation effect is discussed through this simulation. The second simulates the hardware architecture with ModelSim-SE 10.4 software (i.e., functional simulation). In all cases, the simulations and real-time measurements are applied to the 4D Lorenz chaotic system described by the nonlinear dynamic equations (1) with an integration step $h = 0.001$ and initial values $x_0 = y_0 = z_0 = w_0 = -10$. In addition, hyperchaotic signals were obtained such that the hyperchaotic signals $(x,\ y,\ z, w)$ were represented with (10Q22) bits fixed-point arithmetic, 10 bits integer, and 22 bits decimal. The random signals $(x_r, y_r, z_r, w_r)$ are the fraction part (22 bits) of the hyperchaotic signals. The RK4 simulation results are given in Figure 13 and Figure 14, while those of the functional simulations are depicted in Figure 15. The MATLAB simulation results are used to reference both the functional and real-time measurements. The results obtained by simulating the hardware are very similar to those obtained with MATLAB software.

#### 1) BIFURCATION DIAGRAM

The bifurcation diagram is employed to study the different transitions to the chaos of a nonlinear dynamic system. This type of diagram highlights the technique leading to chaos dynamics, namely, the period-doubling cascade. Figure 16

**TABLE 3.** Routing table cases.

| | Cas 1 | | Cas 2 | |
|---|---|---|---|---|
| FPGA Gateway | Source address | Destination address | Source address | Destination address |
| 1 | 192.168.110.249 | 192.168.110.83 | 192.168.110.83 | 192.168.110.249 |
| 2 | 192.168.110.12 | 192.168.110.43 | 192.168.110.43 | 192.168.110.12 |

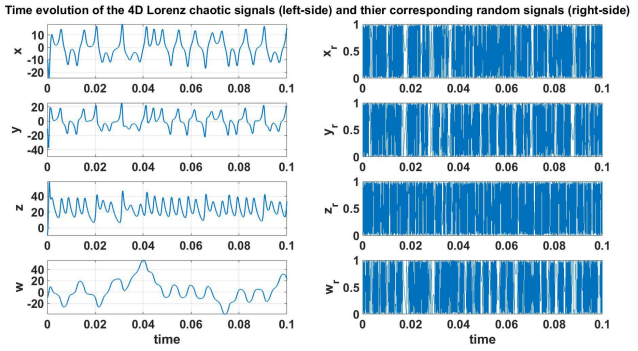


**FIGURE 13.** Time evolution of the 4D Lorenz chaotic signals (left side) and corresponding random signals (right side).

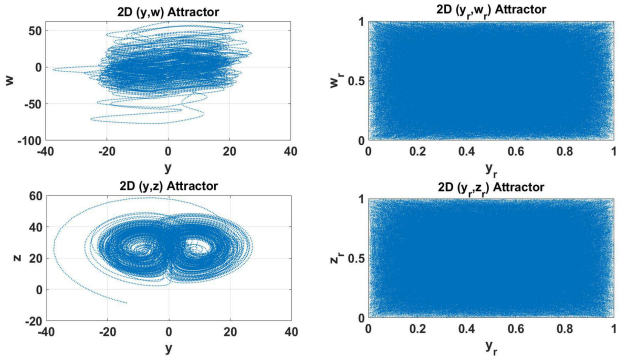


**FIGURE 14.** MATLAB simulation results: Strange Attractors of the 4D Lorenz chaotic system: (a) y-w phase plane, (b) y-z phase plane. Strangeattractors of the random 4D Lorenz chaotic system: (c) yr-wr phase plane, (d) yr-zr phase plane.

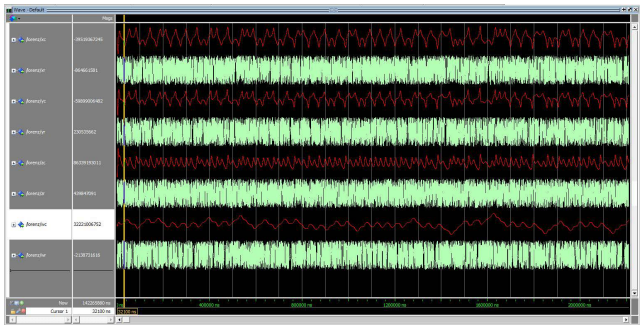

**FIGURE 15.** ModelSim-SE functional simulation results.

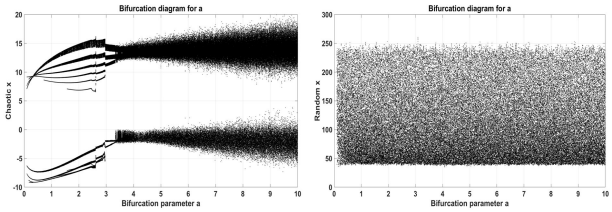


**FIGURE 16.** Bifurcation diagram: Original Lorenz chaotic system (left) andrandom Lorenz system (right).

illustrates the behavior of the original 4D Lorenz chaotic system and its random variant, defined by equations (1) and (5) as a function of the parameter $a$. In particular, for $a = 1/2$, we observe a doubling of the period called here bifurcation. Before falling into chaos, there is a cascade of period doublings. After doubling the period, the previous periodic orbit is still present but unstable. A chaotic system, therefore, has an infinity of periodic orbits. From the bifurcation diagram of the original system, we can deduce the parameters that lead to the chaotic regime of a nonlinear system. In contrast, the bifurcation diagram of the random system does not present any doubling of the period, which confirms, on the one hand, the nonperiodicity of the system and, on the other hand, its random behavior.

### 2) DYNAMICAL DEGRADATION EFFECT

In this subsection, the dynamical degradation effect is discussed. In our study, two types of error are considered: discretization and computing precision errors. The discretization error is related to approximating the classical derivative of the continuous chaotic system, the numerical resolution method, and the choice of the sampling step. Furthermore, nonlinear systems are particularly sensitive to the type and the selected the computing precision. Regardless of the finite precision, the system's dynamics are always too limited compared to its real behavior, and it is impossible to avoid the usage of truncation round operations in the intermediate calculations. In this simulation, we choose three configurations to solve the system. The first is to use the same resolution method and precision and change the step. The second is to use the same resolution method and step but change the accuracy. The last one is to use different resolution methods with the same precision and sampling step. The values in Table 4 are obtained by simulation, and these values clearly show that the error is inversely proportional to the sampling steps and the fraction part in the selected precision. Additionally, the RK4 error is smaller than the Euler error, confirming that the RK4 method is more accurate than the Euler method.

## D. REAL-TIME MEASUREMENTS

Comparing the real-time implementation findings, illustrated in Figure 17 and Figure 18, with those obtained by simulations, we find that they are too similar. Finally, the obtained results are satisfactory and determine the feasibility of embedding the proposed security architecture in a real-time and optimized way by targeting a specific hardware

**TABLE 4. Simulation results of different error types according to numerical method and size-step.**

| Resolution Method | Computing Precision | Step | Samples | Min error ($\times 10^{-7}$) | Max error ($\times 10^{-7}$) |
|---|---|---|---|---|---|
| RK4 | (12Q52) bits | 0.01 | 800000 | -1.48 | 8.565 |
| | | 0.001 | | -2.657 | 7.265 |
| | (10Q22) bits | 0.01 | | -12.53 | 20.06 |
| Euler | (12Q52) bits | 0.01 | 800000 | -2.745 | 7.52 |
| | | 0.001 | | -162.1 | 104.9 |
| | (10Q22) bits | 0.01 | | -162.1 | 152.6 |

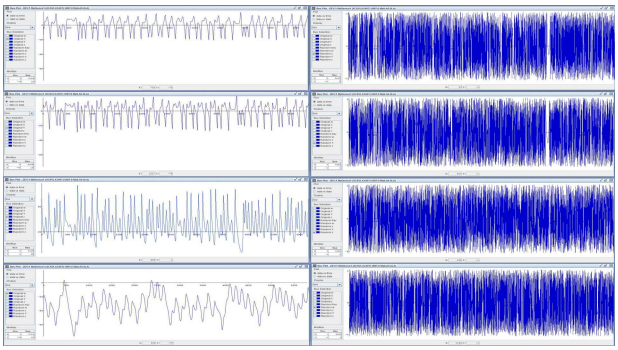

**FIGURE 17. Real-time implementation: 4D Lorenz hyperchaotic signals (left side) and corresponding random signals (right side).**

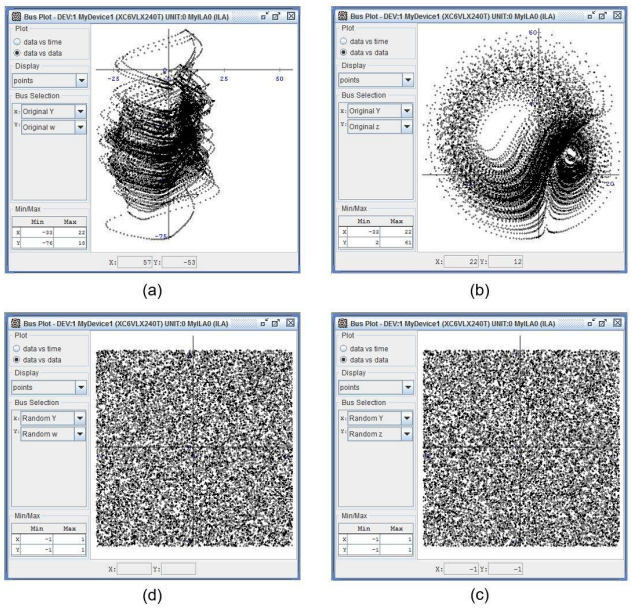


**FIGURE 18. Real-time implementation: Strange Attractors of the 4D Lorenz chaotic system: (a) y-w phase plane, (b) y-z phase plane. Strange attractors of the random 4D Lorenz chaotic system: (c) yr-wr phase plane, (d) yr-zr phase plane.**

system, such as FPGA technology. Thus, we undoubtedly validate our implementation method and the adopted approach to developing a novel 4D hyperchaotic-based security core.

### *E. EXPERIMENTAL SETUP*

After developing our cryptosystem and presenting the simulation results, this section presents its FPGA implementation integrated into a real-world application for establishing a real-time and secure wireless messaging exchange between

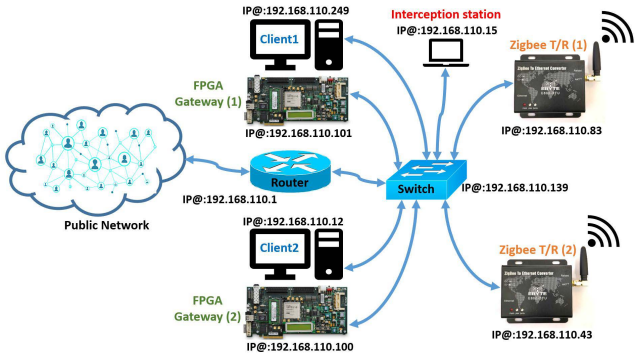


**FIGURE 19. Network topology and IPaddress configuration.**

two network nodes. We begin with the presentation of the experimentations, and we conclude with the presentation of the different obtained hardware synthesis results and related performance and security analysis.

The basic idea is to carry out a messaging exchange between two nodes according to the network topology and IP address configuration of Figure 19. In this configuration, the FPGA platforms embedding the proposed cryptosystem architecture of Figure 20 are operated as a secure gateway. In other words, any information exchange can only be done through the FPGA gateway. In this experiment, all communications are realized in full-duplex mode as follows:

- Using GJI, client1 sends a message to FPGA gateway (1);
- The FPGA gateway (1) encrypts client1's message and sends it to ZigBee E800-DTU (1). This latter acts as a UDP client of FPGA gateway (1) and at the same time as a network coordinator for the ZigBee E800-DTU (2). Additionally, ZigBee E800-DTU (2) is configured to be a UDP client of FPGA gateway (2) and a terminal node of ZigBee E800-DTU (1). Therefore, all messages from FPGA gateway (1) are automatically transferred to ZigBee E800-DTU (2) through ZigBee E800-DTU (1) and then routed to FPGA gateway (2) and vice versa.
- The message received by FPGA gateway (2) is decrypted and sent to client 2 to be displayed on the GJI.

The goal is to hide the plaintext in the hyperchaotic keys. Each plaintext character is encoded in 8 bits and encrypted by an 8-bit portion of the key. To avoid losing information, we obtain more diffusion in the ciphertext. The second type of result relates to the functioning of our chaotic cryptosystem in a real-world application. The goal is to hide the plaintext in the hyperchaotic keys. Each plaintext character is

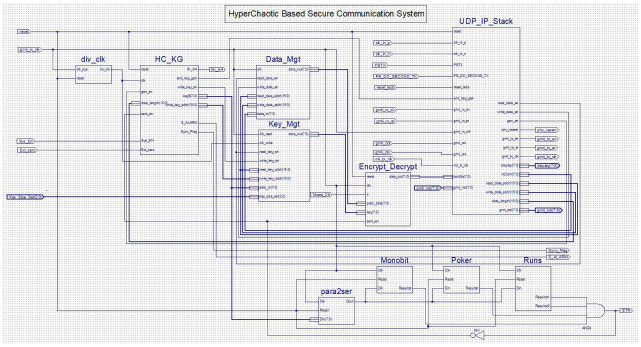


**FIGURE 20.** The implemented schematic of the proposed cryptosystem.

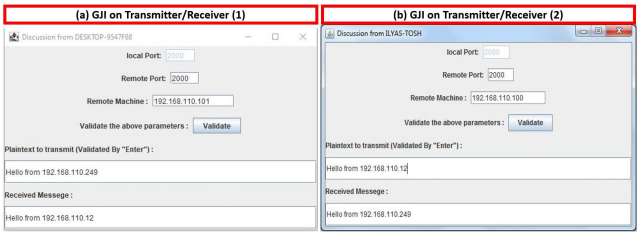


**FIGURE 21.** Messaging exchange using the developed graphical java interface.

encoded in 8 bits and encrypted by an 8-bit portion of the key. To avoid losing information, we obtain more diffusion in the ciphertext.

By using the developed GJI, as shown in Figure 21, the communication functions properly and without losing information. Therefore, on the one hand, we can realize a messaging exchange between two network nodes and, on the other hand, validate the correct functioning of secure communication.

To better analyze the obtained results, in what follows, we present the different captures of UDP packets provided by Wireshark 3.4.6 software [96]. For reasons of organization of the paper and to avoid repetition, we only give Wireshark captures for a single transmission from Client 2 to Client 1. In this experiment, another station is used to intercept the UDP traffic between Client 1 and Client 2. To do this, one of the ports of the switch RG-ES224GC is configured as a mirroring port and connected to the interception station. On the one hand, this configuration allows a duplicate of all the network traffic between Client 1 and Client 2 to be obtained, and on the other hand, the same traffic can be captured and analyzed under the interception station by using Wireshark software.

Figure 22 illustrates the packets captured during the communication between Client 2 and FPGA gateway 2 or between FPGA gateway 1 and Client 1. Note that those packets represent the message before encryption (plaintext) or after decryption (ciphertext). Consequently, the exchange between clients and the FPGA platform is always in clear mode. Unexpectedly, the communication between the two FPGA platforms through the ZigBee modules is always in cipher mode, as shown in Figure 23. In other words, only the FPGA gateways handle text encryption and decryption.

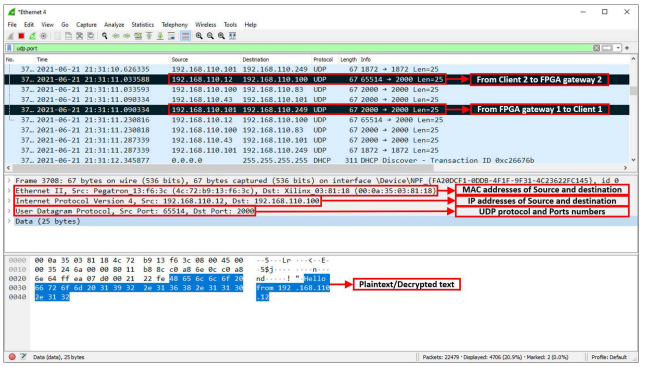


**FIGURE 22.** Captured plaintext on wireshark software.

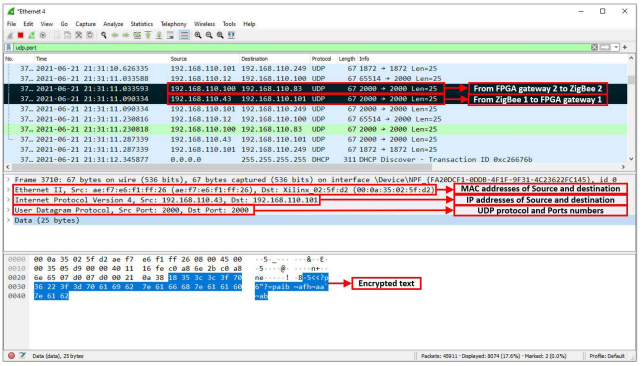


**FIGURE 23.** Captured ciphertext on wireshark software.

Consequently, the hardware-based security criterion is perfectly ensured by the proposed cryptosystem. This criterion guarantees strong system protection by isolating all secrets (i.e., keys generation, encryption, and decryption) from any software exposure at any point in time.

### *F. CRYPTOSYSTEM SYNTHESIS RESULTS AND PERFORMANCE ANALYSIS*

This part mainly concerns the implementation results and performance analysis to demonstrate the robustness of the proposed security core. Several tools were used to carry out these results: MATLAB 2020b software, ISE-DS 14.7 (Integrated Synthesis Environment-Design Suite), Eclipse IDE 2020-09 for Java developers, and ModelSim-SE 10.4 of Mentor Graphics. A panoply of investigations has been performed to investigate the proposed design, according to the following:

- The occupied FPGA area is estimated based on the used Flip Flops (FFs), Lookup Tables (LUTs), Block Random Access Memory (BRAM), and Digital Signal Processing unit (DSP). More precisely, every slice in Virtex-6 XC6VLX240T contains four (04) LUTs and eight (08) FFs, accordingly the AS (Area Size) in terms of the LS (Logic Slice) is equivalent to a quarter of (LUTs+FFs/2);
- Maximum Post Place and Route operating Frequency (MPRF) and throughput, which is defined as the number of bits by a unit of time and can be formulated by $TP = \left(\frac{O\_Size}{O\_latency}\right) = (O\_Size \times GF)\,(\frac{Gb}{s})$, where $O_size$is the output size, $O\_latency$ is the delay to obtain a new

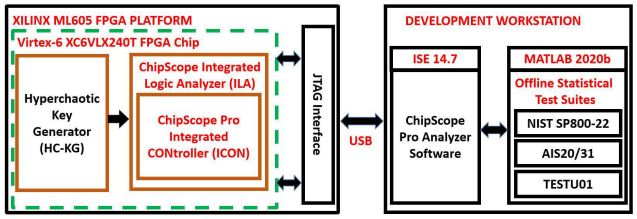


**FIGURE 24.** The implemented architecture for real-time data collection.

output, called output latency modeled by $O\_latency = \left(\frac{N\_cycles}{MPRF}\right)(ns)$, $N\_cycles$ is the number of clock cycles to obtain one output (design latency) (in our case, $O\_Size = 8\ bits$ and $N\_cycles = 1$) and generation $GF$ is the reverse of $O\_latency$;

- The power is evaluated for the register transfer abstraction level (RTAL) at the post place and route stage, considering all the hardware implementation details (physical constraints, placement and routing delays, device settings, ambient temperature, MPRF). Therefore, the power value is more accurate and closer to that measured when the FPGA circuit is configured. Using the Xilinx Xpower Analyzer tool [97], both static and dynamic power can be estimated. Accordingly, in our work, only the dynamic power consumption is presented with the following environmental conditions: industrial temperature grade ($-65\ to + 125°C$), junction temperature $53.2°C$, small size board ($4^{"} \times 4^{"}$), number of board layers ($12\ to\ 15$), power supply ($Vcc_{int} = 1\ V,\ Vcc_{aux} = 2.5\ V$);
- Design efficiency is based on two crucial criteria, timing efficiency and power efficiency, which can be expressed by $T_{eff} = \left(\frac{MPRF}{AS}\right)\left(\frac{MHz}{LS}\right)$ and $P_{eff} = \left(\frac{Power}{AS}\right)\left(\frac{mW}{LS}\right)$, respectively.
- Security analysis and randomness characteristics tests include keyspace and key sensitivity analysis, histogram, information entropy analysis, and statistical tests. Statistical tests comprise three test suites: NIST SP 800-22, TESTU01, and AIS20/31. Moreover, for our sake and for the correct application of these tests, each of the seven test suites has been developed and implemented under the MATLAB 2020b environment.

Figure 24 presents the implemented architecture and experimental setup for real-time data collection for statistical tests. Note that the data used for randomness quality evaluation and security analyses are not issued from simulation but represent the raw random sequences (without postprocessing) generated physically (real word data) after physically embedding the key generator on the XILINX ML605 FPGA target. This data acquisition is carried out by using the hardware architecture debugging tool ChipScope analyzer [98]. The data are captured using a trigger and stored in an internal buffer. A PC interface can collect the data stored in this buffer through a JTAG link. This method offers a good acquisition speed and allows an accurate evaluation of the random statistical characteristics of the generated keys and a clear understanding of the internal functioning of the designed security core.

### 1) HC-IoT-DSC SYNTHESIS RESULTS

epicts, on the one hand, the synthesis results obtained after place and route and, on the other hand, the implementation performance analysis summary of the proposed HC-IoT-DSC. The implementation targeted the Xilinx ML605 FPGA platform (Virtex-6 XC6VLX240T). Note that in our implementation, the selected step-size is $h = 0.001$. First, we discuss the results obtained for each submodule constituting the HC-IoT-DSC. Next, we present the total resource consumption and the maximum frequency of the entire system.

Regarding physical resources, the RK4 and PT submodules occupy almost 90% of the total physical area size of the proposed security core (737 out of 877 $LS$). In the same context, comparing the implementations of all the submodules, we find that only these two submodules require multiplication operations. Thus, the consumed DSP blocks in the security core are only those used by these two submodules. Consequently, they consume more than 65% of the total power (15.25 out of 23.35 $mW$). Another critical measure is the maximum operating frequency. We can clearly say that submodules occupying the smallest physical area operate in high frequency and consume less than 35% of the total power (8.1 out of 23.35 $mW$). It is evident that the proposed FIPS 140-2 can achieve high speed, allowing real-time analysis of the hyperchaotic key stream generator to predict any deviation from normal functioning. These different synthesis results demonstrate that the proposed hyperchaotic-based security core can be easily and efficiently implemented on an FPGA target by using only 877 $LS$ (2%) of the logic slices with 108 DSP blocks (15%) and no block RAM under a maximum frequency of $MPRF = 29.465\ MHz$ and total power of 23.35 $mW$.

To effectively evaluate the hardware implementation of the proposed security core, we use some evaluation metrics directly related to the maximum operating frequency. The metrics are the throughput rate and the time latency, where different throughput values are presented. Moreover, these values correspond to 8-, 16-, 16-, 22-, 32-, 64-, and 128-bit key lengths. However, the longest critical path (design latency) to generate one key, independent from the size, is four cycles of the maximum operating frequency (MPRF). From Table 5, we have reached the highest throughput of 0.2490 $Gb/s$ for the 4D Lorenz hyperchaotic system, where the throughput for the keystream generator varies between a minimum of 0.0589 $Gb/s$ (for 8 bits keys) and a maximum of 0.9424 $Gb/s$ (128 bits keys). Additionally, the latency of the entire security system is 135.754 $ns$. As a result, it can be stated that the proposed security solution has a good trade-off between high speed and low logic resources, which is very attractive for securing IoT communication systems.

### 2) UDP/IP STACK SYNTHESIS RESULTS

This section reports the hardware realization results and performance analysis investigating the performance of

**TABLE 5. Security core synthesis results and implementation performance analysis summary.**

| Target | | Module | | | | | |
|---|---|---|---|---|---|---|---|
| Device Virtex-6 | XC6VLX240T | HC Encryption Key Generator ($h = 0.001$) | | FIPS 140-2 | | | HC-IoT-DSC |
| Package | ff1156 | | | | | | |
| Speed | -1 | RK4 | KMP | FT | PT | RT & LRT | |
| Physical Resources | LUT | 1265 | 31 | 101 | 965 | 268 | 2630/150720 (1%) |
| | FF | 345 | 20 | 66 | 1092 | 182 | 1705/301440 (1%) |
| | BRAM | 0 | 0 | 0 | 0 | 0 | 0/416(0%) |
| | DSP | 60 | 0 | 0 | 48 | 0 | 108/768 (15%) |
| | AS (LS) | 360 | 11 | 34 | 377 | 90 | 877/37680 (2%) |
| Timing and Power Analysis | MPRF (MHz) | 31.129 | 394.321 | 532.340 | 174.785 | 549.843 | 29.465 |
| | Output Latency (ns) | 128.497 | 10.144 | / | / | / | 135.754 |
| | Design Latency | 4 | / | / | / | / | 4 |
| | TP(Gb/s) | $TP_{32}$ 0.249 | / | / | / | / | $TP_8$ 0.0589, $TP_{16}$ 0.1178, $TP_{22}$ 0.1619, $TP_{32}$ 0.2356, $TP_{64}$ 0.4712, $TP_{88}$ 0.6479, $TP_{128}$ 0.9424 |
| | Power (mW) | 11.45 | 2.38 | 1.49 | 3.80 | 1.85 | 23.35 |

the proposed UDP/IP stack. The design has been evaluated according to the following factors: the occupied FPGA area, MPRF, throughput TP, power consumption, timing efficiency, and power efficiency. Additionally, the different metrics related to the QoS aspect (i.e., transfer rate (TR) and transfer efficiency ratio (TER)) are specified.

In preparation for throughput and TER measurements, the Xilinx ML605 FPGA board is configured to transmit and receive a 1472 bytes UDP datagram payload (DPL) with an overhead (DOH) of 28 bytes (8 bytes UDP header and 20 bytes IP header), which conduct to a maximum packet length of 1500 bytes. Furthermore, a JAVA app was built to manage communication between the FPGA board and a personal computer (PC) equipped with a Realtek PCIe GbE family controller and an Intel Core i7-10700 CPU at 2.90 GHz running Windows 10 Pro. In addition to the UDP socket configuration, the JAVA app executes the following operations:

1. Send and receive a fixed data block size (1472 bytes) for a test duration (TD) of 20 seconds and a throttle UDP bandwidth (B) of 106 bits/s;

2. Compute $TR_{FS} = 2\left(\frac{N_{packet} \times D_{PL}}{TD \times B}\right)(MB/s)$, where $TR_{FS}$ is the transfer rate for a fixed data packet size and $N_{packet}$ is the number of successfully transmitted or received packets by the FPGA board;

3. Compute the $TER = \left(\frac{N_{rx-packet}}{N_{tx-packet}}\right) \times 100$;

4. Send and receive a variable data block size (64 to 1472 bytes) for a test duration (TD) of 20 seconds and a throttle UDP bandwidth of 106 bits/s.

5. Compute the transfer rate $TR_{VS} = \frac{1}{8}\left(\frac{D_{PL}}{D_{PL}+D_{OH}} \times S\right)(MB/s)$, where $TR_{VS}$ is the transfer rate for variable data packet size and $S$ is the connection speed (in our case, $S = 1000\ MB/s$);

6. Compare the computed $TR_{VS}$ to the predicted transfer rate.

Implementing the UDP/IP architecture on the ML605 FPGA platforms has produced several results. The first result relates to the consumption of resources. As indicated in Table 6, the proposed architecture consumes low physical resources, with only 653 logic slices (1%), two BRAM (1%), and no DSP blocks. This result also verifies the constraint of the open, flexible, and simple hardware architecture, allowing the addition of possible future hardware modules (ICMP, TCP, DHCP). The low number of BRAMs and DSP nonuse gives us high independence from the FPGA target. In other words, it offers good portability of the developed solution with no technology constraint.

The second result corresponds to the critical operating frequencies and power consumption of the design. The UDP/IP stack can operate at a maximum frequency of 123.229 MHz (0.986 Gb/s), almost very close to the theoretical frequency of 125 MHz (1 Gb/s). This time performance is more than sufficient for Ethernet-based communication at a maximum rate 1 Gbits/s. Additionally, it fulfills the constraint set at the start of the design. The low power consumption (40.62 mW) can be explained by the low physical cost and confirmed by the obtained timing efficiency and the power efficiency values.

The third result is the QoS metrics benchmarking. This test aims to study the behavior of UDP/IP for data packets with variable sizes. The obtained TR at the output is 116.475 MB/s (114.65 MB/s) with a TER of 99.12% (98.67%) for transmission and reception, respectively. Figure 25 illustrates a comparison between the predicted and measured values of the transfer ratio TRVS. The obtained measurements indicate good convergence between those two values. These results show that our UDP/IP can be implemented on FPGA technology to provide high-speed communication.

**TABLE 6. Summary of UDP/IP Stack implementation performance analysis.**

| Target | | Module | | | |
|---|---|---|---|---|---|
| Device | Virtex-6 XC6VLX240T | UI | UDP+IPv4+ARPEMAC | | UDP/IP Stack |
| Package | ff1156 | | | | |
| Speed | -1 | | | | |
| Physical resources | LUT | 113 | 1683 | 103 | 1899/150720 (1%) |
| | FF | 91 | 1181 | 148 | 1420/301440 (1%) |
| | BRAM | 0 | 1(RAMB36E1) 1(RAMB18E1) | 0 | 2 1/416 (0%) 1/832 (0%) |
| | DSP | 0 | 0 | 0 | 0/768 (0%) |
| | AS (LS) | 40 | 570 | 45 | 653/37680 (1%) |
| Timing and Power Analysis | MPRF (MHz) | / | 130.242 | 161.009 | 123.229 |
| | Latency (ns) | / | / | / | 8.1149 |
| | TP(Gb/s) | / | 1.041 | 1.288 | 0.986 |
| | Power (mW) | 0.7 | 23.53 | 16.39 | 40.62 |
| Design Efficiency | $T_{eff}$ ($\frac{MHz}{LS}$) | / | 0.228 | 3.577 | 0.188 |
| | $P_{eff}$ ($\frac{mW}{LS}$) | 0.0175 | 0.041 | 0.364 | 0.062 |
| QoS Metrics | $TR_{FS}$ (MB/s) | TX | | RX | |
| | | 116.475 | | 114.65 | |
| | TER (%) | FPGA to PC | | PC to FPGA | |
| | | 99.12 | | 98.67 | |

The QoS metrics are given for the UDP/IP stack, including all submodules (UI+UDP+IPv4+ARP+EMAC)

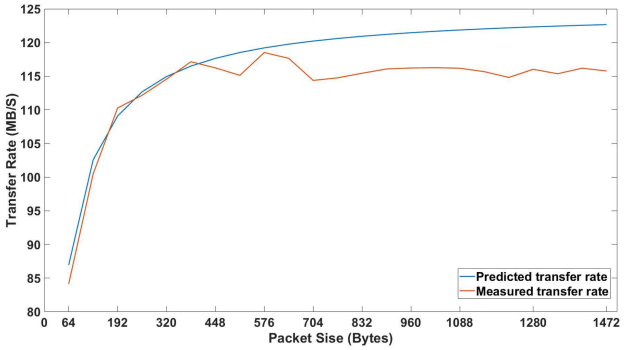


**FIGURE 25. Measured and estimated transfer rate comparison.**

We present the following comparative study to demonstrate that our architecture meets the requirements. This comparative study focuses on the consumed physical resources, timing and power analysis, communication features, and design properties. Table 7 reports a performance comparison between our implantation and five other similar works ( [99]–[101], and [102]), from which we can deduce the following:

Our architecture occupies the fourth range compared to other architectures regarding the achieved throughput with a 0.986 $Gb/s$. The proposed UDP/IP stack consumes fewer FPGA resources in slices, BRAM, and DSP with high flexibility and simplicity. Moreover, our UDP/IP stack is the only architecture that ensures multientries ARP functionality with a minimum BRAM. Therefore, the designed stack offers good portability with minor modifications and high technology independency. These various remarks show that our architecture and that proposed in [100] are the best candidates for onboard applications. Indeed, those two architectures present an almost complete system compared to the other designs concerning different comparison criteria and needed requirements. Furthermore, our stack is superior in terms of security. This property can be explained by the fact that while our conception, many security measurements have been taken into consideration, such as MAC address and port number control, static routing configuration, and IP packet fragmentation deactivation.

### 3) THE GLOBAL CRYPTOSYSTEM SYNTHESIS RESULTS

The consumption of material resources in terms of slices and the critical operating frequency of the developed architecture are summarized in Table 8. These results show that our architecture can be efficiently implemented on FPGA technology. The cryptosystem can operate at a maximum post place and route frequency of 25.26 MHz. This temporal performance is more than sufficient for the operation of an Ethernet communication at a maximum rate of 10 $\frac{Mbits}{s}$, where the obtained throughput TP is approximately 0.033 $\frac{Gb}{s}$. Moreover, our architecture permits low physical resource consumption at high speed.

Indeed, the execution on the ML605 platform uses only 1523 logic slices (4%) with 2 RAM blocks under a power consumption of 63.97 $mW$. This result also verifies the constraint of the open hardware architecture, which is presented in the timing $0.0165\frac{MHz}{LS}$ and the power 0.024 $\frac{mw}{LS}$ efficiencies. Hence, this allows future architectural optimizations and gives the possibility to add more features.

**TABLE 7.** Performance comparison of the implemented UDP/IP stack.

| | | [99] | [100] | [101] | [102] | Ours |
|---|---|---|---|---|---|---|
| FPGA Device | | V5/SP6 | SF2 | XCVU190 | XCKU040 | V6 |
| Physical resource | AS (LS) | 4679/2288 | 853 | 5179 | 2513 | 653 |
| | BRAM | 19/23 | N/A | 40 | 11 | 2 |
| | DSP | 0/0 | N/A | N/A | N/A | 0 |
| Timing and Power Analysis | MPRF ($MHz$) | 66.3/62.3 | 25 | 250 | 312.5 | 123.229 |
| | Latency ($ns$) | 633/979 | 690 | N/A | N/A | 8.1149 |
| | TP ($Gb/s$) | 0.865 | 1 | 40 | 10 | 0.986 |
| | Speed ($Mb/s$) | Tri-Mode | Tri-Mode | N/A | N/A | Tri-Mode |
| | $TR_{FS}$ ($MB/s$) | 570 | N/A | N/A | N/A | $\leq$116.475 |
| | TER (%) | N/A | $\leq$99 | N/A | N/A | $\leq$99.12 |
| | Power ($mW$) | N/A | 41.2 | N/A | N/A | 40.62 |
| Design Efficiency | $T_{eff}$ ($\frac{MHz}{LS}$) | 0.0141/0.0272 | 0.0293 | 0.0482 | N/A | 0.188 |
| | $P_{eff}$ ($\frac{mW}{LS}$) | N/A | 0.0483 | N/A | N/A | 0.062 |
| Communication Features | Duplex mode | Full | Full | Full | Full | Full |
| | Length (byte) | 1512 | 512 | 1500 | 64 | 1472 |
| | ARP | Yes | Yes | Yes | Yes | (255 entries) |
| | RARP | N/A | N/A | N/A | N/A | N/A |
| | ICMP | Yes | N/A | Yes | N/A | N/A |
| | TCP channel | N/A | N/A | N/A | N/A | N/A |
| | DHCP | Yes | N/A | N/A | N/A | N/A |
| | Routing | N/A | Yes | N/A | N/A | Yes |
| | Checksum | Yes | Yes | Yes | Yes | Yes |
| | PHY Medium | GMII | N/A | N/A | XGMII | GMII |
| Design Properties | Standardization | Yes | Yes | Yes | Yes | Yes |
| | Flexibility | High | High | High | High | High |
| | Simplicity | High | High | High | High | High |
| | Security | N/A | N/A | N/A | N/A | Yes |

SP= Spartan, V= Virtex, SF= Smart Fusion, XCVU= Xilinx UltraScale, XCKU= Xilinx Kintex UltraScale, Tri-Mode= 10/100/1000, N/A= not presented.

**TABLE 8.** Synthesis results and performance analysis summary of the proposed cryptosystem.

| Target | | Module | | |
|---|---|---|---|---|
| Device<br>Package<br>Speed | Virtex-6 XC6VLX240T<br>ff1156<br>-1 | UDP/IP stack | HC-IoT-DSC | Proposed Cryptosystem |
| Physical Resources | LUT | 1899 | 2630 | 4529/150720 (3%) |
| | FF | 1420 | 1705 | 3125/301440 (1%) |
| | BRAM | 2 | 0 | 2/416(**0%**) |
| | DSP | 0 | 108 | 108/768 (14%) |
| | AS (LS) | 653 | 877 | 1523/37680 (4%) |
| Timing and Power Analysis | MPRF (MHz) | 123.229 | 29.465 | 25.26 |
| | Output Latency (ns) | 8.1149 | 135.754 | 237.529 |
| | Design Latency (Cycle) | / | 6 | 6 |
| | TP (Gb/s) | 0.986 | 0.9424 | 0.033 |
| | Power (mW) | 40.62 | 23.35 | 63.97 |
| Design Efficiency | $T_{eff}$ ($\frac{MHz}{LS}$) | 0.188 | 0.0818 | 0.0165 |
| | $P_{eff}$ ($\frac{mW}{LS}$) | 0.062 | 0.064 | 0.024 |

### G. SECURITY ANALYSIS

Mainly, the effectiveness of a cryptosystem is reflected directly by its resistance level to different security attacks. Many MATLAB scripts have been implemented to process the experimental data using security analysis and performance metrics to evaluate this level. Such tests include key sensitivity, histogram, chi-square, differential attacks, correlation, floating frequency, and information entropy analyses. Moreover, all the investigations are discussed and compared to related works.

#### 1) HC-KG KEY SENSITIVITY ANALYSIS

The main characteristics of hyperchaotic systems are unpredictability and high sensitivity to slight variations in initial conditions (IC) ($x_0$, $y_0$, $z_0$ and $w_0$) and mismatch or control parameters (MP) ($a$, $b$, $c$, and $d$). Indeed, the

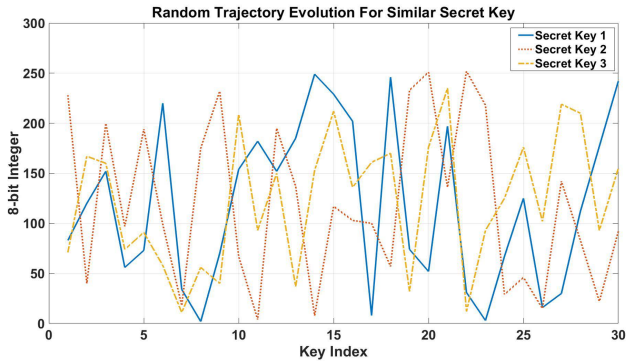


**FIGURE 26.** The HC-KG key sensitivity impact on the random trajectory.

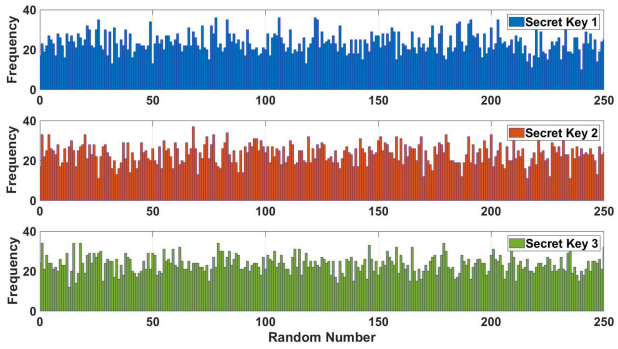


**FIGURE 27.** The HC-KG histogram analysis for three similar secret keys.

unpredictability property comes from a minimal variation in IC and/or in the MP inducing a radically different evolution in the dynamics of the hyperchaotic system, in that the latter is sensitive to IC. Accordingly, the proposed random generator should provide other keys even if it uses a very close IC or MP as secret keys or seed values. To this end, an experiment was conducted to measure the influence of a slight level change of at least significant bit positions on the resemblances of the generated keys.

In Figure 26, we illustrate the first thirty (30) 8-bit keys in the random trajectory of the proposed HC-KG by using three similar secret keys (see Table 9) with a one-bit change (equivalent to $10^{-16}$), whereas the three trajectories behave differently. Thus, the proposed HC-KG is very sensitive even at the bit change level in the secret keys. This is not a disadvantage but an advantage that makes hyperchaotic systems good candidates for multiuser communications. Indeed, we can generate an infinity of encryption keys from a given hyperchaotic generator; it suffices for this to slightly modify the values of its parameters.

### 2) HC-KG KEYSPACE ANALYSIS

In this subsection, the key size is analyzed from two different angles. The first comprises determining the key size from the initial conditions and control parameter codification. The second is to deduct the key size from the key sensitivity analysis of the previous subsection. Indeed, the dynamic of the proposed HC-KG is related to eight different 32 bits values, represented on four initial conditions ($x_0$, $y_0$, $z_0$ and $w_0$) and four control parameters ($a$, $b$, $c$, and $d$). Moreover, the secret key can be any of the $2^{32}$ values. Likewise, it can be any of the eight values. Hence, this gives a key-size of $(2^{32})^8 = 2^{256}$. The key sensitivity analysis determined that the proposed HC-KG is very sensitive to any change equal to $10^{-16}$. Thus, the keyspace is larger than $10^{16}$. Similarly, the secret key can be any of the eight values. Therefore, the key size is $(10^{16})^8 = 10^{128} \approx 2^{425}$. According to the Advanced Encryption Standard (AES), a random number generator is resilient against brute force attacks if it has a keyspace of secret keys larger than $2^{128}$. Comparing the obtained key spaces to the required criteria, the proposed HC-KG is large enough to resist exhaustive attacks. A comparison of the key space is accessible in Table 10.

### 3) HC-KG HISTOGRAM ANALYSIS

The uniform distribution property in the generated random sequences is an essential key factor regarding security. In other words, the repetition frequency of each element in a random sequence. One of the most commonly used methods for verifying this property is the histogram. Figure 27 illustrates the histogram of three random sequences of 1 *Mbits* each (131, 072 integers of 8 bits), generated by using the secret keys of Table 9. It is documented that the histograms are uniform and notably different. Therefore, the proposed HC-KG can generate uniform random sequences and is resilient against statistical analysis attacks.

### 4) HC-KG INFORMATION ENTROPY ANALYSIS

In this test, the entropy is computed using Maurer's universal statistical algorithm, a compression type test detailed in [86], [107]. Furthermore, the same algorithm is adopted by the statistical test batteries NIST SP 800-22 and AIS20/31. This algorithm requires a sequence of $n$ bits, which is divided into two chunks, $Q(\geq 10 \bullet 2^L)$ initialization blocks and $K(\approx 10 \bullet 2^L)$ test blocks with $6 \leq L \leq 16$ and $K = \left(\frac{n}{L}\right] - Q$. Each chunk is a set of $L$ *bits* blocks or templates (in our case, $L = 8$). Next, we sweep the whole sequence by a block of $L$ *bits* looking for the closest preceding exact bit block template match and recording the distance in multiple applying blocks. Then, we compute the $log_2$ of all distances for all the $L$ *bits* templates within the test blocks. More precisely, it effectively gives the number of digits in the binary expansion of each distance [108]–[110]. Finally, we average all the expansion lengths by the number of test blocks. The following expression gives the information entropy:

$$H = \frac{1}{K} \sum_{i=Q+1}^{Q+K} log_2 k \tag{13}$$

where k is the number of indices since the previous occurrence of the *ith* template. In this experiment, to realize an in-depth analysis of the information entropy $H$, the latter is measured at different operating frequencies varying between 1 *to* 29 *MHz*. Therefore, the secret key 1 of Table 9 is used to

**TABLE 9.** Secret keys.

| Keys | Secret Key |
|---|---|
| **Key 1** | $x_0 = -10 + 10^{-16};\ y_0 = -10;\ z_0 = -10;\ w_0 = -10; a = 10; b = 28; c = \frac{8}{3}; d = 5$ |
| **Key 2** | $x_0 = -10;\ y_0 = -10 + 10^{-16};\ z_0 = -10;\ w_0 = -10; a = 10; b = 28; c = \frac{8}{3}; d = 5$ |
| **Key 3** | $x_0 = -10;\ y_0 = -10;\ z_0 = -10;\ w_0 = -10; a = 10 + 10^{-16}; b = 28; c = \frac{8}{3}; d = 5$ |

**TABLE 10.** Keyspace comparison.

| Criteria | Ours | [103] | [54] | [42] | [104] | [105] | [36] | [106] | [65] |
|---|---|---|---|---|---|---|---|---|---|
| **Keyspace** | $2^{256}$ *and* $2^{425}$ | $2^{158}$ | $2^{240}$ | $2^{160}$ | $2^{111}$ | $2^{192}$ | $2^{324}$ | $2^{400}$ | $2^{224}$ |
| **Key sensitivity** | $10^{-16}$ | × | $10^{-8}$ | × | $10^{-16}$ | $10^{-10}$ | × | × | × |

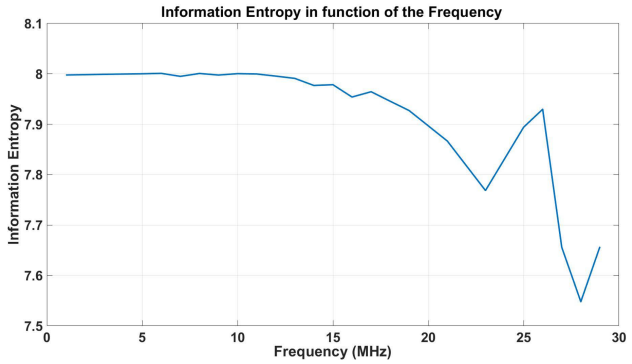


**FIGURE 28.** The HC-KG information entropy analysis results.

generate 23 random sequences of $4 \bullet 10^6 bits$ each (524, 288 integers of 8-bit) at 23 different frequency values. Figure 28 shows the obtained entropy measurements. We remark that the entropy level is nearly uniform and stable for all the studied frequencies with an average of $H = 7.9873$. Therefore, the randomness aspect is validated, and the proposed HC-KG is secure against information entropy attacks.

### 5) RANDOMNESS CHARACTERISTICS ANALYSIS OF THE HC-KG

In this section, to evaluate the randomness aspect of the proposed HC-KG, a variant of random sequences is generated and tested using NIST SP800-22, TESTU01, and AIS.20/31.

#### *a: NIST SP800-22 TESTS*

A statistical test consists of stating a hypothesis concerning a set of data and then checking whether the obtained observations are plausible within the framework of this hypothesis. The hypothesis to be tested is called the null hypothesis $H_0$. It is imperatively accompanied by its alternative hypothesis called $H_a$. Hypothesis $H_0$ is the one we are trying to refute, the one that is ''true'' until we prove the contrary. The hypothesis $H_a$, contrary to $H_0$, is the one we seek to demonstrate. For each test, the result leads to a decision: accept or reject $H_0$. To reach an objective decision, first, the null hypothesis $H_0$ is established considering its alternative hypothesis $H_a$. Then, to test the established hypothesis, an appropriate statistical test with a level of significance $\alpha$ is specified. Moreover, the sampling distribution of the statistical test under $H_0$ should be found. Based on the previous steps, the rejection region is defined. Finally, the value of the statistical test using the sample data is computed. In our case, the hypothesis $H_0$ is that the studied or ''tested'' sequence is random, which induces $H_a$: the tested sequence is not random. The appropriate choice of statistical tests for testing $H_0$ is the NIST SP800-22 test series. The significance level $\alpha$ of the test represents the probability that hypothesis $H_0$ is rejected when it should have been accepted. In practice, an upper limit of the first type for the level $\alpha$, most often 5% (significant), 1% (very substantial) or 0.1% (highly significant). More precisely, our approach evaluates the distribution of the P_Values for each test in NIST SP800-22 while calculating the P_ValueT. If this value is less than 0.1% (highly significant decision threshold), then the conclusion is that the sequence does not satisfy the corresponding random criterion. Otherwise, we can consider the sequence to be random and uniformly distributed.

As the proposed HC-KG can generate encryption keys of different sizes, for each key size, our analysis was carried out on 1073 sequences of one million bits (1, 000, 000 bits) for all tests. Table 11 summarizes the P_ValueT to examine the distribution of the P_Values of each test. As a reminder, this must be less than 0.1% to consider that the sequence does not meet the uniform distribution criterion. As can be observed from Table 11, all the generated keys pass the statistical tests. Therefore, we confirm the randomness aspect of the generated keys, and the proposed HC-KG has an excellent statistical performance concerning the NIST SP800-22 standard.

#### *b: TESTU01 TESTS*

Currently, TestU01 is the most complete and challenging test for RNGs, including eight (08) subbatteries with over 282 statistical tests. These subbatteries are SmallCrush, Crush, BigCrush, Alphabit, Rabbit, PseudoDIEHARD, FIPS 140-2, and NIST SP800-22. The SmallCrush is applied first. If the tested sequence passes, the crush tests are involved, and the more complex the BigCrush will be applied. Generally, if the tested sequence passes the previous battery, there is a high probability of succeeding in the remaining tests. Moreover, in these subbatteries, the tested sequences and the

**TABLE 11.** NIST SP800-22 test results: The$P_Value_T$ of the various tests.

| Key | | 8-bit | 16-bit | 22-bit | 32-bit | 64-bit | 88-bit | 128-bit | Obs. |
|---|---|---|---|---|---|---|---|---|---|
| Tests | | | | | $P\_Value_T$ | | | | |
| Approximate Entropy | | 0.7396 | 0.9115 | 0.9544 | 0.8491 | 0.8116 | 0.6642 | 0.7026 | Pass |
| Block Frequency | | 0.9017 | 0.9555 | 0.9900 | 0.5522 | 0.8094 | 0.7369 | 0.5801 | Pass |
| Cumulative Sums | Forward | 0.8702 | 0.8271 | 0.6182 | 0.8156 | 0.6698 | 0.8165 | 0.6513 | Pass |
| | Reverse | 0.8048 | 0.6411 | 0.7994 | 0.7939 | 0.4894 | 0.5622 | 0.9032 | |
| FFT | | 0.8916 | 0.8762 | 0.9070 | 0.4516 | 0.4438 | 0.6543 | 0.9689 | Pass |
| Frequency | | 0.9394 | 0.7632 | 0.7504 | 0.6891 | 0.8367 | 0.6227 | 0.5169 | Pass |
| Lempel-Ziv Compression | | 0.9896 | 0.6281 | 0.9326 | 0.8964 | 0.9812 | 0.4441 | 0.9326 | Pass |
| Linear Complexity | | 0.8787 | 0.7664 | 0.9765 | 0.6344 | 0.9612 | 0.9841 | 0.6889 | Pass |
| Longest Runs of Ones | | 0.7172 | 0.9164 | 0.6713 | 0.7210 | 0.7716 | 0.9224 | 0.9792 | Pass |
| Nonperiodic Templates | | 0.9149 | 0.8801 | 0.9873 | 0.9614 | 0.9695 | 0.9931 | 0.9946 | Pass |
| Overlapping Template | | 0.8502 | 0.9784 | 0.7610 | 0.8528 | 0.9593 | 0.9887 | 0.9075 | Pass |
| Random Excursions | | 0.8066 | 0.8719 | 0.9337 | 0.9661 | 0.9080 | 0.7416 | 0.9416 | Pass |
| Random Excursions Variant | | 0.9511 | 0.9531 | 0.8057 | 0.9680 | 0.9520 | 0.9131 | 0.8799 | Pass |
| Rank | | 0.9164 | 0.7224 | 0.3944 | 0.7248 | 0.6317 | 0.7377 | 0.8736 | Pass |
| Runs | | 0.8901 | 0.9534 | 0.6425 | 0.7503 | 0.8637 | 0.7122 | 0.6321 | Pass |
| Serial | | 0.9062 | 0.6244 | 0.6092 | 0.6493 | 0.9288 | 0.5877 | 0.9847 | Pass |
| Maurer's Universal Statistical | | 0.8602 | 0.5244 | 0.9778 | 0.4155 | 0.8704 | 0.7228 | 0.9470 | Pass |
| 1073 analyses were carried out, and the success rate was $\geq 99\%$ | | | | | | | | | |

parameters of the tests are not fixed, making TestU01 more flexible. In these tests, TestU01 is used to verify whether the generated sequences behave randomly and flow a uniform probability distribution over the interval [0, 1]. P_Values within $[0.001, 0.9995]$ are considered accepted. Table 12 presents the obtained results of the TestU01 batteries. The proposed HC-KG passes all tests; therefore, it has good randomness and statistical quality.

### *c: AIS20/31 TESTS*

In this subsection, the randomness quality of the proposed HC-GK is proved by using the AIS20/31 test suite that we developed under MATLAB software according to the requirements and the specifications of BSI. This suite includes nine statistical tests (T0 to T8) (see Annex B). AIS20/31 is organized into two procedures, A and B, conducted in seven steps (A-1 to A-7) and five (B-1 to B-5) for procedures A and B. In step A-1, test T0 is applied to a sequence of at least $2^{16} \bullet 48$ bits. However, in steps A-2 to A-7, tests T1-T5 are used for a sequence of 20000 bits and repeated 257 times. Procedure A is passed if and only if all 1285 basic tests $(1 \times T0 + (T1\ to\ T5) \times 257)$ have been passed. If more than one basic test failed, procedure A failed. If precisely one basic test has failed, the second run of procedure A is tolerable. If one basic test failed within a second repetition, procedure A failed. Procedure B applies the uniform distribution (T6a and T6b) test and the homogeneity test (T7a and T7b) for widths 1, 2, 4, 8 on a 100000-bit sequence, followed by Coron's test (T8) on a 25600 + 2560 bit sequence. Note that each of the tests above represents one step in procedure B, which results in five stages with five basic tests in total. Procedure B is passed if all the basic tests have been passed. Similar to procedure A, procedure B falls if more than one basic test fails, but procedure B's second repetition is acceptable if just one basic test has been unable. If procedure B's second execution and one basic test fail, procedure B falls, and a third repetition is not allowed. Table 13 illustrates the obtained results of AIS20/31 tests, and the proposed HC-KG successfully passes all the basic tests for procedures A and B. Therefore, according to the AIS20/31 standard, the proposed solution can produce an excellent random sequence with high randomness and sufficient entropy density.

### 6) CRYPTOSYSTEM KEY SENSITIVITY ANALYSIS

In this subsection, we determine the sensitivity of the proposed cryptosystem to all encryption and decryption keys. This sensitivity can be evaluated following two methods. The first is to quantify the influence of a insignificant level change in the encryption key on the similarities of the generated ciphertexts. Indeed, changing one bit in the encryption key must produce a dissimilar ciphertext. The other is to validate the impossibility of recovering the plaintext when a slight change in the decryption key value is made. Accordingly, five similar keys were used to encrypt a plaintext of 1472 bytes. The obtained ciphertexts and decrypted text for the 30 first bytes of plaintext are represented in Figure 29. The obtained ciphers are different, demonstrating that the proposed encryption scheme is susceptible to tiny changes in the encryption keys. Alternatively, decrypting the cipher encrypted by key 0 using the other keys (key 1, key 2, key 3, key 4) produced incorrect plaintexts, as shown in Figure 29. However, the decryption process generates the correct plaintext using the valid encryption key. Therefore, it proves that the proposed decryption scheme is also sensitive to bit-level changes in the decryption keys.

### 7) CRYPTOSYSTEM AVALANCHE EFFECT ANALYSIS

To measure the sensitivity of the proposed cryptosystem to a slight change in the plaintext, the AE (Avalanche Effect) is used. Generally, a small change in the plaintext should cause no less than 50% change in the generated ciphertext. Similarly, the same sensitivity can be estimated using the MSE

**TABLE 12.** TestU01test results: The P_values of the various tests.

| Battery | Parameters | Tests | P_Values | Obs. |
|---|---|---|---|---|
| SmallCrush | Standard | 10 15 Pass | | |
| Crush | Standard | 96 | 144 | Pass |
| BigCrush | Standard | 106 | 160 | Pass |
| Alphabit | Standard | 9 | 17 | Pass |
| Rabbit | Standard | 26 | 40 | Pass |
| PseudoDIEHARD | Standard | 15 | 21 | Pass |
| NIST SP800-22 | Standard | 16 | 17 | Pass |
| FIPS 140-2 | Standard | 4 | 4 | Pass |

**TABLE 13.** AIS.31 test suite results compared to required thresholds.

| Test | | Limits | Result | Obs. |
|---|---|---|---|---|
| **Procedure A** | | | | |
| Disjointness (T0) | | 0 | 0 | Pass |
| Monobit (T1) | | $9654 < X_M < 10346$ | 9996 | Pass |
| | | $H_{min} \geq 0.9509269$ | 0.9962538 | Pass |
| | | $H_1 \geq 0.9991363$ | 0.9998504 | Pass |
| Poker (T2) | | $1.03 < \chi^2 < 57.4$ | 15.08 | Pass |
| Runs (T3) | Run=1 | $2267 - 2733$ | 2497 | Pass |
| | Run=2 | $1079 - 1421$ | 1249 | Pass |
| | Run=3 | $502 - 748$ | 624 | Pass |
| | Run=4 | $233 - 402$ | 313 | Pass |
| | Run=5 | $90 - 233$ | 157 | Pass |
| | Run=6+ | $90 - 233$ | 155 | Pass |
| Longest Run (T4) | | 0 | 0 | Pass |
| Autocorrelation (T5) | | $2326 < Z_\tau < 2674$ | 2498 | Pass |
| **Procedure B** | | | | |
| Uniform Distribution (T6) | T6a | $\lvert Prb_{(1)} - 0.5\rvert < 0.025$ | 0.00015 | Pass |
| | T6b | $\lvert Prb_{0(1)} + Prb_{1(0)} - 1\rvert < 0.020$ | 0.00096 | |
| Homogeneity (T7) | T7a | Width=3 $Limit \geq 15.13$ | 22.5438 | Pass |
| | T7b | Width=4 | 27.2872 | |
| Entropy estimation (T8) | | $H > 7.976$ | 7.9978 | Pass |

For the tests T1 to T5, each test has been performed 257 times. Thus, illustrated values in the table represent the 257 average.

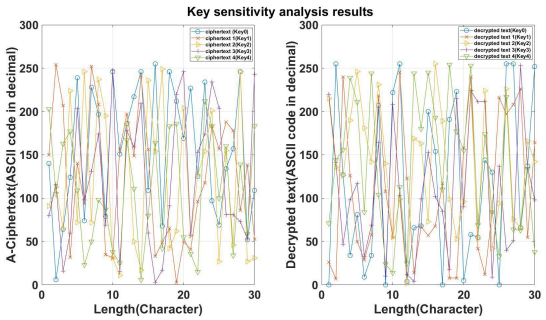


**FIGURE 29.** The proposed cryptosystem key sensitivity analysis.

(mean square error). The MSE shows that the accumulated square error between two ciphertexts results from similar plaintexts. The MSE is formulated by:

$$MSE(j) = \frac{1}{N * B} \sum_{i=0}^{N-1} \left(ct_j(i) - ct_{j+1}(i)\right)^2, \; j \in (0, 1, 2, 3) \tag{14}$$

where $c_1$ and $c_2$ are two ciphers, and $N = 1472$ *bytes*, and $B$ denote the most significant supported byte value of the generated ciphertext. In our case, $B = 255$. Usually, if MSE $\geq$ 30 dB, the difference in the plaintext is very clear. In this test, five similar plaintexts of 1472 bytes are encrypted with the same key to generate five ciphers. Then, the AE and MSE are estimated, as illustrated in Table 14. The obtained results are within the required thresholds. Therefore, the proposed cryptosystem verifies and accomplishes the plaintext sensitivity test.

**TABLE 14.** The proposed cryptosystem avalanche effect analysis.

| test | AE (%) | MSE (dB) |
|---|---|---|
| **Test 0** | 50.5944 | 42.3239 |
| **Test 1** | 50.4840 | 42.8844 |
| **Test 2** | 50.5095 | 42.3214 |
| **Test 3** | 50.1763 | 42.3572 |

### 8) CRYPTOSYSTEM DIFFERENTIAL ATTACK ANALYSIS

Differential attack analysis is another approach to measure the diffusion performance of an encryption scheme. More precisely, we quantify the capacity of an encryption scheme to resist differential attacks when a minor change in the

**TABLE 15.** The proposed cryptosystem differential attack analysis.

| | Ours | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | test 1 | test 2 | test 3 | test 4 | [111] | [66] | [112] | [107] | [113] | [42] |
| **NPCR** | 99.6603 | 99.5244 | 99.5923 | 99.3885 | 99.5913 | 99.5800 | 99.6103 | 99.6100 | 99.6054 | 99.6070 |
| **UACI** | 33.8019 | 33.9236 | 33.2257 | 33.2427 | 33.3861 | 33.5600 | 33.4615 | 33.5200 | 33.9547 | 33.4395 |
| **Correlation Coef.** | 0.0012 | 0.0097 | 0.0010 | 0.0018 | −0.0032 | 0.0022 | 0.0020 | 0.0006 | 0.00049 | 0.0047 |

plaintext is made (i.e., a one-bit change in the plaintext should produce a different ciphertext). NPCR, UACI, and correlation coefficients are employed to evaluate the dependency level between the ciphertext and the plaintext. NPCR, UACI, and the correlation can be formulated as follows:

$$NPCR(\%) = \frac{100}{N}\sum_{i=0}^{N-1} D\,(i) \tag{15}$$

$$UACI(\%) = \frac{100}{N * B}\sum_{i=0}^{N-1} |ct_1 - ct_2| \tag{16}$$

$$R_{ct_1 ct_2} = \frac{cov(ct_1, ct_2)}{\sqrt{D\,(ct_1)}\sqrt{D\,(ct_2)}} \tag{17}$$

where $ct_1$ and $ct_2$ are two ciphers of the same size $N = 1472$ *bytes* and $B$ denotes the largest supported ciphertext byte value of the generated ciphertext. In our case, $B = 255$. If $ct_1(i) \neq ct_2(i)$ then $D(i) = 1$; otherwise, $D(i) = 0$. Additionally, $cov\,(ct_1, ct_2) = \frac{1}{N}\sum_{i=1}^{N}(ct_{1i} - E(ct_1))(ct_{2i} - E(ct_2))$, $D\,(ct_1) = \frac{1}{N}\sum_{i=1}^{N}(ct_{1i} - E(ct_1))^2$ and $E\,(ct_1) = \frac{1}{N}\sum_{i=1}^{N} ct_{1i}$. In this experiment, a plaintext is modified to generate four similar plaintexts. Then, the cipher of the original plaintext and the other ciphers are used to calculate the NPCR, the UACI, and the correlation coefficients. Table 15 illustrates the obtained results. We observe that the obtained values for all the tests are within the required standard. Therefore, the proposed cryptosystem can efficiently resist differential attacks.

### 9) CRYPTOSYSTEM CHOSEN-PLAINTEXT ANALYSIS

Always in the same context of differential attacks, this analysis shows the vulnerability of a CPA (chosen-plaintext attack) and its effect on an encryption scheme. According to [113], if a cryptosystem can resist CPAs, it can resist other conventional attacks. To demonstrate this, two plaintexts are chosen: the first is a text of 1472 "A" characters, and the other is a text of 1472 "B" characters. Table 16 lists the obtained entropy values and correlation coefficients of plaintext and ciphers. We can observe that the entropies of ciphers are close to 8, and the correlation values are very close to 0. Therefore, the proposed scheme is secure against chosen plaintext attacks compared to other conventional attacks.

### 10) CRYPTOSYSTEM HISTOGRAM ANALYSIS

Histogram analysis is commonly used to validate that the generated ciphers are uniformly distributed and to verify the effectiveness of a cryptosystem against statistical attacks.

**TABLE 16.** The proposed cryptosystem chosen-plaintext analysis.

| Text | | Entropy | Correlation coefficient |
|---|---|---|---|
| "A" | **Plaintext** | 0 | – |
| | **Ciphertext** | 7.9314 | 0.0036 |
| "B" | **Plaintext** | 0 | – |
| | **Ciphertext** | 7.9314 | 0.0038 |

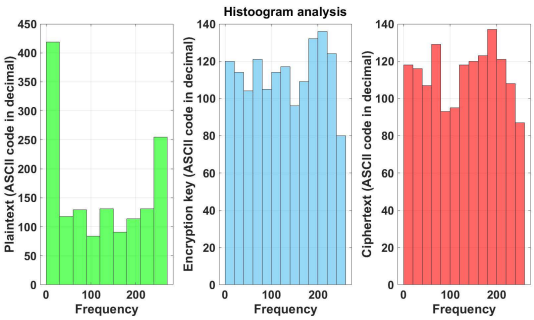


**FIGURE 30.** The proposed cryptosystem histogram analysis.

Figure 30 shows the histogram of the plaintext, the encryption key, and the ciphertext. The plaintext histogram is nonuniform and irregularly distributed. Nevertheless, the histograms of the encryption key and the produced cipher are uniformly distributed. Thus, the proposed encryption scheme can resist statistical attacks.

### 11) CRYPTOSYSTEM CHI-SQUARE ANALYSIS

This analysis comes to complete that test based on the histogram to confirm the randomness of the produced ciphers. To this end, the independence and goodness-of-fit tests used in NIST SP800-90B recommendations are exploited to verify this uniformity through the chi-square metric. Five ciphers of 1472 bytes each are generated and tested in this test. Table 17 shows the obtained chi-square values compared to the required theoretical thresholds. All the tested text passed the test; therefore, the histogram uniformity is validated, and the proposed encryption scheme has good randomness.

### 12) CRYPTOSYSTEM CORRELATION COEFFICIENT ANALYSIS

Using correlation coefficients, we verify the independence between ciphertext adjacent characters in this analysis. Indeed, a good encryption scheme had better eradicate the high character correlation in plaintext and produce a random cipher that can resist statistical attacks. Formula (17) is used to compute the correlation, where $ct_1$ and $ct_2$ represent adjacent characters of the same ciphertext. Furthermore, five

**TABLE 17.** The proposed cryptosystem chi-square analysis.

| Chi-square Tests | Threshold | Cipher 1 | Cipher 2 | Cipher 3 | Cipher 4 | Cipher 5 | Observation |
|---|---|---|---|---|---|---|---|
| **Independence** | $T \leq 329.3828$ | 274.5128 | 250.0874 | 305.5231 | 237.0600 | 224.4686 | Pass |
| **Goodness-of-fit** | $T \leq 27.8772$ | 8.8627 | 5.2469 | 10.5157 | 9.3932 | 6.2338 | Pass |

**TABLE 18.** The proposed cryptosystem correlationcoefficient analysis.

| | Ours | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | test 1 | test 2 | test 3 | test 4 | [111] | [66] | [112] | [114] | [107] | [42] |
| **Correlation coefficients** | 0.0064 | −0.0029 | 0.0044 | 0.0051 | −0.0032 | 0.0022 | 0.0020 | 0.0006 | 0.00049 | 0.0047 |

**TABLE 19.** The proposed cryptosystem information entropy analysis.

| | Ours | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | test 1 | test 2 | test 3 | test 4 | test 5 | [111] | [66] | [112] | [114] | [107] | [42] |
| **Plaintext** | 4.7892 | 4.6155 | 4.6154 | 4.7013 | 4.6871 | – | – | – | – | – | 6.8981 |
| **Cyphertext** | 8.0122 | 7.9797 | 7.9367 | 8.0344 | 8.0505 | 7.9981 | 7.9894 | 7.9992 | 7.9969 | 7.9979 | 7.9974 |

plaintexts are encrypted, and their ciphers are tested. Table 18 lists the obtained results. All the values are close to zero, demonstrating that the produced cipher has good randomness and its characters are highly uncorrelated. Thus, the proposed encryption scheme can resist statistical attacks.

### 13) CRYPTOSYSTEM FLOATING FREQUENCY ANALYSIS

Similar to the histogram and chi-square analysis, the floating frequency opts to evaluate the occurrence of different elements in all test windows of a ciphertext. Likewise, this occurrence should be uniformly distributed. To this end, a plaintext of 1472 *bytes* is encrypted using a random key. To measure the floating frequency, we consider a gap size of 256 elements and count the occurrence of different elements within that gap. Then, the chosen gap is shifted by one element to the right, and the floating frequency is computed. Figure 31 shows that the calculated floating frequency is nearly uniform. Thus, the proposed method verifies the randomness property. This analysis is fundamental, as it specifies whether an attacker can construct the whole plaintext or understand the encryption process from a subsequence of the cipher. Therefore, the obtained results demonstrate that this attack was unsuccessful.

### 14) CRYPTOSYSTEM INFORMATION ENTROPY ANALYSIS

Information entropy is used to measure the amount of information, a concept of information theory. The more orderly a system is, the lower information entropy is; conversely, the more confusing the entropy is. The formula can calculate information entropy:

$$H(s) = \sum_{i=0}^{2^L-1} p(s_i) log_2 \frac{1}{p(S_i)} \tag{18}$$

where $p(S_i)$ denotes the probability of symbol $s_i$. The closer the information entropy is to 8, the more random the image is. In the test, five plaintexts and their ciphers are tested. Table 19 lists the obtained entropy values. We can see that the

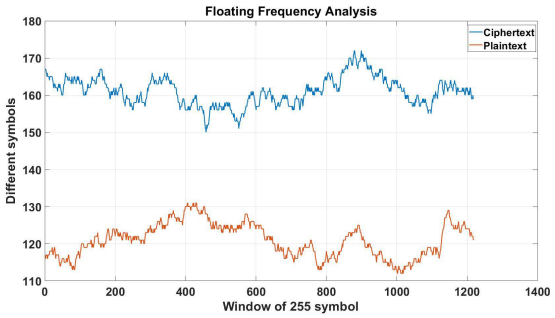


**FIGURE 31.** The proposed cryptosystem floating frequency analysis.

entropies of plaintexts are lower and different; however, when encrypted, the entropies of ciphers are close to 8. Therefore, the proposed encryption scheme produces ciphers with high randomness. Therefore, our cryptosystem can provide defense against entropy attacks.

### 15) CRYPTOSYSTEM COMPUTATIONAL COMPLEXITY AND TIMING ANALYSIS

The execution time and computational complexity are essential criteria for evaluating an encryption scheme's performance. This analysis focuses on estimating three metrics: the synchronization time, the key generation time, and the encryption/decryption time. These metrics are expressed as follows:

$$T_{Gen} = D_L \bullet CLK \tag{19}$$

$$T_{Sync} = NC \bullet T_{Gen} \tag{20}$$

$$\begin{aligned} T_{ED} &= N \bullet T_{Gen} + T_{Sync} \\ &= (N + NC) \bullet T_{Gen} \\ &= \hat{N} * T_{Gen} \end{aligned} \tag{21}$$

where:

- $T_{Gen}$, $T_{Sync}$ and $T_{ED}$ are the key generation time, the synchronization time, and the encryption/decryption time, respectively;

**TABLE 20. Computational complexity and timing analysis.**

| Encryption/Decryption time (s) | | | | | | |
|---|---|---|---|---|---|---|
| | Schemes | | | | | |
| Plaintext length (Byte) | [66] | [114] | [115] | [116] | [117] | Ours |
| $2^{14}$ | – | – | – | – | 0.06 | 0.0039 |
| $2^{16}$ | 13.95 | 0.736 | 1.01 | – | 0.23 | 0.0157 |
| $2^{18}$ | – | – | – | 1.53 | 0.68 | 0.0629 |
| Computation Complexity | | | | | | |
| Computation cost | [42] $3N^2+13N+3$ | | [118] $17\,(M+N)$ | | [36] $20MN$ | Ours $14N$ |

**TABLE 21. Comparison to related works.**

| Parameters | [39] | [48] | [40] | [120] | [21] | [22] | Ours |
|---|---|---|---|---|---|---|---|
| Chaos system | AES/Modified Lorenz System | Switched Chaotic System | Logistic Map and Henon system | Modified Chaotic Transition Map | 5D MSTS Hyperchaotic Dynamo system | 4D MS Hyperchaotic system | Discretized Lorenz System |
| Numerical Method | – | Euler | Recursive integration | Recursive integration | FE[4]/ trapezoidal/ RK4 | FE/ BE[5]/ RK4 | RK4 |
| Application | Image Encryption | Image Encryption | Image Encryption | Speech encryption | Image encryption | Image Encryption | Real-time wireless secure Communication |
| Implementation | VHDL | VHDL/C++ | VHDL/C | VHDL | VHDL | VHDL | VHDL/JAVA |
| Simulation | √ | √ | √ | √ | √ | √ | √ |
| Hardware Demo | – | √ | √ | √ | √ | √ | √ |
| Device | XC5VLX50T | Genesys-II | Cyclone-III | Virtex-5 | Cyclone-IV | Cyclone-IV | Virtex-6/ZigBee |
| Area size (LS) | 1866 | 1539 | 3216 | 541 | 434 | 423 | 1523 |
| MPRF (MHz) | 144.11 | 307 | 167.83 | 95.384 | 97.3 | 89.88 | 25.26 |
| Output Latency (ns) | – | – | – | – | 860 | 720 | 237.529 |
| Design Latency (Cycle) | – | – | – | 5 | 43 | 36 | 6 |
| TP (Gb/s) | 3.458 | – | 1.34 | 1.526 | – | – | 0.033 |
| Power (mW) | – | 529 | 137.06 | – | – | – | 63.97 |
| $T_{eff}$ $(MHz/LS)$ | 0.0772 | 1.999 | 0.0521 | 0.1763 | 0.2241 | 0.2124 | 0.0165 |
| $P_{eff}$ $(mW/LS)$ | – | 0.345 | 0.0426 | – | – | – | 0.024 |
| Keyspace | – | $2^{769}$ | $2^{128}$ | $2^{128}/2^{210}$ | – | – | $2^{256}$ / $2^{425}$ |
| Key size (bits) | 128 | 32 | 128 | 128 | – | – | Multiple[1] |
| Randomness | – | NIST SP800-22, DIEHARD | NIST SP800-22 | – | – | – | 3 tests suites[2] |
| Security Analysis | √ | √ | √ | √ | √ | √ | √ |
| Synchronization | – | – | – | – | – | – | √ |
| Communication Medium | – | – | – | – | – | – | UDP/IP stack and wireless ZigBee E800-DTU |

**1**. Support different key sizes (8, 16, 22, 32, 64, 88, and 128 bits). **2**. NIST SP800-22, TestU01, AIS20/31, and FIPS 140-2. **4**. FE= Forward-Euler. **5**. BE= Backward-Euler

- $D_L$ is the design latency, which in our case is 6;
- CLK is the system clock period; in our experiment, 25 MHz;
- $NC$ is the clock cycle required to achieve synchronization, and $N$ is the size of plaintext.

The experiment showed that for a multitude of tests, synchronization is always achieved after $NC = 120$ iterations. Each iteration is equivalent to $D_L = 6$ clock cycles. Moreover, the proposed encryption and decryption methods are based on diffusion and inverse diffusion principles.

These principles are realized by applying an XOR (Exclusive Or) between plaintext and an OTP (One-Time Pad key). Accordingly, the encryption/decryption time is proportional to the plaintext size, which is proportional to the key generation time, as shown by equation (21). In addition, regardless of the key size, the generation process consumes six clock cycles and requires only six additions and seven multiplications. Hence, the proposed scheme is less complex and time-efficient with a linear computation complexity of $14N$. Table 20 lists the obtained encryption/decryption time and the computation complexity and compares them to related works. The proposed scheme presents the best performance in terms of encryption/decryption time and computation complexity.

#### 16) CRYPTOSYSTEM PERFORMANCE COMPARISON

This subsection is devoted to comparing the proposed encryption scheme with various recent works that provide chaos-based key generation and their related information security applications (i.e., image encryption, speech encryption, and wireless secure communication). In this comparison, all aspects are considered, exclusively those related to hardware implementation and security analysis. As shown in Table 21, our scheme is the most evaluated and analyzed among all the presented works. More precisely, the randomness of the proposed system has been verified using three different test suites, especially the more complex and challenging (i.e., TestU01). However, only some or null test suites are used in the other systems. In addition, the proposed scheme presents the best keyspace after the approach suggested in [48] because this latter used four chaotic systems for the key generation, in contrast to only one in our scheme. Moreover, in terms of power consumption and occupied physical space, our design with 63.97 $mW$ is a good candidate for constrained objects such as IoT devices. It is worth mentioning that the occupied FPGA area is presented in logic slices (LS), except those implemented on Altera Cyclone IV ( [21], [22]) are presented in logic array blocks (LABs). Precisely, the authors in [21], [22] provide the consumed physical area in term of logic elements (LEs). Additionally, according to the Altera Cyclone IV handbook [119], each LAB contains 16 LEs. Therefore, to realize a reasonable comparison, we divide the number of LEs by 16 to obtain the consumed FPGA area in terms of LABs, which is comparable to LS. Additionally, the ability of the proposed scheme to provide encryption keys with different sizes simplifies its implementation on a variety of platforms. In contrast, it can be used with varying data formats and support any data size. Furthermore, all the related works are limited to the simulation phase or laboratory demonstration except those proposed in [21], [22], [39]. However, our work has simulated, implemented, and integrated the proposed scheme into a real-time wireless secure text transmission.

## V. CONCLUSION

The present work comprised an FPGA design and realized embedded hyperchaotic communication for real-time and secure wireless data transmissions. The developed architectures are modular and comprise a UDP/IP stack and a chaotic security core. It was necessary to go through several stages to arrive at the final solution. The first step comprises the study, modeling, and implementation of FPGA technology of a UDP/IP interface, intending to control the network communications physically. To do this, we constructed an RTL architecture that relies on a VHDL description of UDP and IP network protocols. This stack has been developed to achieve, on the one hand, a flexible and reconfigurable architecture that gives the possibility of easy adaptation in the future without dependability to the hardware target technology. On the other hand, there is minimum risk regarding security issues. Indeed, the developed UDP/IP interface has low latency and high speed close to the 1 Gb/s theoretical speed. Moreover, this stack is designed to support static routing that uses less bandwidth and to resist many attacks, particularly fragmentation and MAC address spoofing attacks.

The second step is fundamental. It consists of the study and implementation of the chaotic-based security core. This core was implemented by the VHDL description of the RK4 method to generate random data as encryption key matrices. The original idea employs the fundamental properties in terms of nonlinearity, unpredictability, and extreme sensitivity to initial conditions to develop key encryption. In addition, our security core ensures the synchronization mechanism, called HCS-DFM. It is based on the regeneration of chaotic data by a feedback dynamic using an observer. Its advantage is the synchronization of a hyperchaotic signal mixed with useful information. Therefore, the reliability of HCS-DFM synchronization highly depends on how the data and the chaotic signal are incorporated.

The proposed security core has been profoundly and intensely evaluated. Additionally, the proposed solution presents a complete and profound analysis in terms of chaos validation, security aspects, statistical characterization, and architectural conception. Moreover, to the best of our knowledge, the proposed security core is the only one in the literature that its randomness quality and its security aspects have been tested with more than 18 security analyses, including the hardest and the most complex, and applied to real-world data.

Additionally, power consumption was considered very soon in the conception process. Furthermore, to demonstrate the design efficiency of the proposed security, for the first time in the literature, we introduce a crucial criterion, power efficiency ($P_{eff}$), which is the ratio between the power consumption and occupied area size. The obtained results ($P_{eff} = 0.024\ \frac{mW}{LS}$) confirm that our architecture has been effectively designed and implemented with compliance to embedded system requirements.

The last step is integrating the UDP/IP stack and the chaotic security core to develop a secure platform for real-time communicating systems and interconnected devices according to the IoT standards. The obtained real-time results are very satisfactory and validate the adopted approach. Our conclusions can be summarized as follows:

• The advent of digital programmable circuits, such as reconfigurable FPGA-type circuits, makes it possible to design hyperchaotic systems while avoiding the drawbacks of an analog design. Indeed, this type of circuit allows the prototyping and hardware implementation of digital electronic architectures permitting the generation of hyperchaotic signals;

• The adopted hardware/software codesign methodology offers excellent flexibility to the developed systems. It facilitates future updates by using the notion of IP design reuse;

• Indeed, our architecture was tested for secure wireless communication, but because of the high modularity of the proposed hardware design, our cryptosystem is easily configurable for other applications, such as image encryption and speech encryption;

• The proposed security core was intensely, deeply, and thoroughly analyzed and evaluated from all the aspects related to embedded system security applications;

• The obtained results present a good trade between design efficiency and high security;

• Using the fixed-point arithmetic model in the hardware implementation of the chaos-based security core leads to consuming less memory bandwidth, providing faster speed, and attending higher power efficiency. The fixed-point arithmetic permits a beneficial and interesting trade-off between fast speed and minimal resource cost.

• The randomness of our solution has been tested by three (03) statistical test suites. Thus, our security core is more secure with superior randomness and no post-processing; • The proposed security core is equipped with BISST to guarantee online and continuous control of the reliability and availability of the proposed security core;

Finally, according to the above, we can confidently confirm and enforce the statement about the superiority of the proposed security core compared to related works for embedded system security. Additionally, regarding the quantity and quality of the provided efforts and the obtained results, we confidently say that this research could be used as a strong roadmap for similar and future works. Finally, the rich results of this work open up several horizons for further development and future research. They can be classified into three crucial topics:

• First, work on the digital hyperchaotic synchronization component for error minimization and exploring other techniques;

• Second, we work on the data encoding component to be able to use chaos to secure bit-level data. That is, to generate chaotic cryptographically safe sequences;

• Finally, work on the cryptanalysis component to properly assess the degree of confidentiality offered by the designed cryptosystem. In other words, add other security analyses, specifically those related to the hardware, such as power analysis attacks, side-channel attacks, and environmental tests (i.e., temperature, voltage, electromagnetic tests).

## REFERENCES


[1] R. Khan, P. Kumar, D. N. K. Jayakody, and M. Liyanage, "A survey on security and privacy of 5G technologies: Potential solutions, recent advancements, and future directions," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 1, pp. 196–248, 1st Quart., 2020.

[2] B. K. Chae, "The evolution of the Internet of Things (IoT): A computational text analysis," *Telecommun. Policy*, vol. 43, no. 10, Nov. 2019, Art. no. 101848.

[3] R. Leszczyna, "Review of cybersecurity assessment methods: Applicability perspective," *Comput. Secur.*, vol. 108, Sep. 2021, Art. no. 102376.

[4] M. Pham and K. Xiong, "A survey on security attacks and defense techniques for connected and autonomous vehicles," *Comput. Secur.*, vol. 109, Oct. 2021, Art. no. 102269.

[5] S. Kiran and G. Gupta, "Development models and patterns for elevated network connectivity in Internet of Things," *Mater. Today, Proc.*, Jul. 2021.

[6] N. N. Srinidhi, S. M. Dilip Kumar, and K. R. Venugopal, "Network optimizations in the Internet of Things: A review," *Eng. Sci. Technol., Int. J.*, vol. 22, no. 1, pp. 1–21, Feb. 2019.

[7] J. J. Kponyo, J. O. Agyemang, G. S. Klogo, and J. O. Boateng, "Lightweight and host-based denial of service (DoS) detection and defense mechanism for resource-constrained IoT devices," *Internet Things*, vol. 12, Dec. 2020, Art. no. 100319.

[8] E. Lee, Y.-D. Seo, S.-R. Oh, and Y.-G. Kim, "A survey on standards for interoperability and security in the Internet of Things," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 2, pp. 1020–1047, 2nd Quart., 2021.

[9] M. Gomba and B. Nleya, "Architecture and security considerations for Internet of Things," in *Proc. Global Wireless Summit (GWS)*, 2017, pp. 252–256.

[10] K. Främling and M. Maharjan, "Standardized communication between intelligent products for the IoT," *IFAC Proc. Volumes*, vol. 46, no. 7, pp. 157–162, May 2013.

[11] J. Neeli and S. Patil, "Insight to security paradigm, research trend & statistics in Internet of Things (IoT)," *Global Transitions Proc.*, vol. 2, no. 1, pp. 84–90, Jun. 2021.

[12] L. Babun, K. Denney, Z. B. Celik, P. McDaniel, and A. S. Uluagac, "A survey on IoT platforms: Communication, security, and privacy perspectives," *Comput. Netw.*, vol. 192, Jun. 2021, Art. no. 108040.

[13] A. Lohachab, A. Lohachab, and A. Jangra, "A comprehensive survey of prominent cryptographic aspects for securing communication in post-quantum IoT networks," *Internet Things*, vol. 9, Mar. 2020, Art. no. 100174.

[14] A. Alarifi, S. Sankar, T. Altameem, K. Jithin, M. Amoon, and W. El-Shafai, "A novel hybrid cryptosystem for secure streaming of high efficiency H.265 compressed videos in IoT multimedia applications," *IEEE Access*, vol. 8, pp. 128548–128573, 2020.

[15] Z. Rahman, X. Yi, I. Khalil, and M. Sumi, "Chaos and logistic map based key generation technique for AES-driven IoT security," in *Proc. Int. Conf. Heterogeneous Netw. Quality, Rel., Secur. Robustness*. Springer, 2021, pp. 177–193.

[16] A. Hedayatipour and N. McFarlane, "An encryption architecture suitable for on chip integration with sensors," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 11, no. 2, pp. 395–404, Jun. 2021.

[17] G. R. W. Thoms, R. Muresan, and A. Al-Dweik, "Chaotic encryption algorithm with key controlled neural networks for intelligent transportation systems," *IEEE Access*, vol. 7, pp. 158697–158709, 2019.

[18] B. Bordel, R. Alcarria, T. Robles, and M. S. Iglesias, "Data authentication and anonymization in IoT scenarios and future 5G networks using chaotic digital watermarking," *IEEE Access*, vol. 9, pp. 22378–22398, 2021.

[19] L. Li, G. Wen, Z. Wang, and Y. Yang, "Efficient and secure image communication system based on compressed sensing for IoT monitoring applications," *IEEE Trans. Multimedia*, vol. 22, no. 1, pp. 82–95, Jan. 2020.

[20] A. D. Pano-Azucena, B. Ovilla-Martinez, E. Tlelo-Cuautle, J. M. Muñoz-Pacheco, and L. G. de la Fraga, "FPGA-based implementation of different families of fractional-order chaotic oscillators applying Grünwald–Letnikov method," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 72, pp. 516–527, Jun. 2019.

[21] S. Vaidyanathan, A. Sambas, B. Abd-El-Atty, A. A. A. El-Latif, E. Tlelo-Cuautle, O. Guillen-Fernandez, M. Mamat, M. A. Mohamed, M. Alcin, M. Tuna, I. Pehlivan, I. Koyuncu, and M. A. H. Ibrahim, "A 5-D multi-stable hyperchaotic two-disk dynamo system with no equilibrium point: Circuit design, FPGA realization and applications to TRNGs and image encryption," *IEEE Access*, vol. 9, pp. 81352–81369, 2021.

[22] S. Vaidyanathan, A. Sambas, E. Tlelo-Cuautle, A. A. A. El-Latif, B. Abd-El-Atty, O. Guillén-Fernández, K. Benkouider, M. A. Mohamed, M. Mamat, and M. A. H. Ibrahim, "A new 4-D multi-stable hyperchaotic system with, no., balance point: Bifurcation analysis, circuit simulation, FPGA realization and image cryptosystem," *IEEE Access*, vol. 9, pp. 144555–144573, 2021.

[23] L. Runzi, "Adaptive function project synchronization of Rössler hyperchaotic system with uncertain parameters," *Phys. Lett. A*, vol. 372, no. 20, pp. 3667–3671, 2008.

[24] K. S. Sudheer and M. Sabir, "Adaptive modified function projective synchronization between hyperchaotic Lorenz system and hyperchaotic Lu system with uncertain parameters," *Phys. Lett. A*, vol. 373, no. 41, pp. 3743–3748, Oct. 2009.

[25] A. E. Matouk and A. A. Elsadany, "Achieving synchronization between the fractional-order hyperchaotic novel and Chen systems via a new nonlinear control technique," *Appl. Math. Lett.*, vol. 29, pp. 30–35, Mar. 2014.

[26] J. Huang, "Adaptive synchronization between different hyperchaotic systems with fully uncertain parameters," *Phys. Lett. A*, vol. 372, nos. 27–28, pp. 4799–4804, Jun. 2008.

[27] E. Tlelo-Cuautle, A. D. Pano-Azucena, O. Guillén-Fernández, and A. Silva-Juárez, *Analog/Digital Implementation of Fractional Order Chaotic Circuits and Applications*. Springer, 2020.

[28] M. A. Valencia-Ponce, E. Tlelo-Cuautle, and L. G. de la Fraga, "Estimating the highest time-step in numerical methods to enhance the optimization of chaotic oscillators," *Mathematics*, vol. 9, no. 16, p. 1938, Aug. 2021.

[29] T. U. Haq and T. Shah, "4D mixed chaotic system and its application to RGB image encryption using substitution-diffusion," *J. Inf. Secur. Appl.*, vol. 61, Sep. 2021, Art. no. 102931.

[30] S. Hraoui, F. Gmira, M. F. Abbou, A. J. Oulidi, and A. Jarjar, "A new cryptosystem of color image using a dynamic-chaos Hill cipher algorithm," *Proc. Comput. Sci.*, vol. 148, pp. 399–408, Jan. 2019.

[31] J. S. Khan and S. K. Kayhan, "Chaos and compressive sensing based novel image encryption scheme," *J. Inf. Secur. Appl.*, vol. 58, May 2021, Art. no. 102711.

[32] G. Kaur, R. Agarwal, and V. Patidar, "Chaos based multiple order optical transform for 2D image encryption," *Eng. Sci. Technol., Int. J.*, vol. 23, no. 5, pp. 998–1014, Oct. 2020.

[33] Z. Feixiang, L. Mingzhe, W. Kun, and Z. Hong, "Color image encryption via Hénon-zigzag map and chaotic restricted Boltzmann machine over blockchain," *Opt. Laser Technol.*, vol. 135, Mar. 2021, Art. no. 106610.

[34] Z. Man, J. Li, X. Di, Y. Sheng, and Z. Liu, "Double image encryption algorithm based on neural network and chaos," *Chaos, Solitons Fractals*, vol. 152, Nov. 2021, Art. no. 111318.

[35] V. R. Falmari and M. Brindha, "Privacy preserving biometric authentication using chaos on remote untrusted server," *Measurement*, vol. 177, Jun. 2021, Art. no. 109257.

[36] S. Ma, Y. Zhang, Z. Yang, J. Hu, and X. Lei, "A new plaintext-related image encryption scheme based on chaotic sequence," *IEEE Access*, vol. 7, pp. 30344–30360, 2019.

[37] F. S. Hasan and M. A. Saffo, "FPGA hardware co-simulation of image encryption using stream cipher based on chaotic maps," *Sens. Imag.*, vol. 21, no. 1, pp. 1–22, Dec. 2020.

[38] M. D. Gupta and R. K. Chauhan, "Secure image encryption scheme using 4D-hyperchaotic systems based reconfigurable pseudo-random number generator and S-box," *Integration*, vol. 81, pp. 137–159, Nov. 2021.

[39] A. A. Rezk, A. H. Madian, A. G. Radwan, and A. M. Soliman, "On-the-fly parallel processing IP-core for image blur detection, compression, and chaotic encryption based on FPGA," *IEEE Access*, vol. 9, pp. 82726–82746, 2021.

[40] A. Hafsa, M. Gafsi, J. Malek, and M. Machhout, "FPGA implementation of improved security approach for medical image encryption and decryption," *Sci. Program.*, vol. 2021, pp. 1–20, Feb. 2021.

[41] A. Akgul, B. Gurevin, I. Pehlivan, M. Yildiz, M. C. Kutlu, and E. Guleryuz, "Development of micro computer based mobile random number generator with an encryption application," *Integration*, vol. 81, pp. 1–16, Nov. 2021.

[42] H. Wen, C. Zhang, P. Chen, R. Chen, J. Xu, Y. Liao, Z. Liang, D. Shen, L. Zhou, and J. Ke, "A quantum chaotic image cryptosystem and its application in IoT secure communication," *IEEE Access*, vol. 9, pp. 20481–20492, 2021.

[43] P. Ayubi, S. Setayeshi, and A. M. Rahmani, "Deterministic chaos game: A new fractal based pseudo-random number generator and its cryptographic application," *J. Inf. Secur. Appl.*, vol. 52, Jun. 2020, Art. no. 102472.

[44] D. A. Trujillo-Toledo, O. R. López-Bonilla, E. E. García-Guerrero, E. Tlelo-Cuautle, D. López-Mancilla, O. Guillén-Fernández, and E. Inzunza-González, "Real-time RGB image encryption for IoT applications using enhanced sequences from chaotic maps," *Chaos, Solitons Fractals*, vol. 153, Dec. 2021, Art. no. 111506.

[45] Y. Li, Z. Li, M. Ma, and M. Wang, "Generation of grid multi-wing chaotic attractors and its application in video secure communication system," *Multimedia Tools Appl.*, vol. 79, nos. 39–40, pp. 29161–29177, Oct. 2020.

[46] J. S. Teh, M. Alawida, and Y. C. Sii, "Implementation and practical problems of chaos-based cryptography revisited," *J. Inf. Secur. Appl.*, vol. 50, Feb. 2020, Art. no. 102421.

[47] A. Hafsa, M. Fradi, A. Sghaier, J. Malek, and M. Machhout, "Real-time video security system using chaos-improved advanced encryption standard (IAES)," *Multimedia Tools Appl.*, vol. 81, pp. 2275–2298, Oct. 2021.

[48] R. Kaibou, M. S. Azzaz, M. Benssalah, D. Teguig, H. Hamil, A. Merah, and M. T. Akrour, "Real-time FPGA implementation of a secure chaos-based digital crypto-watermarking system in the DWT domain using co-design approach," *J. Real-Time Image Process.*, vol. 18, pp. 2009–2025, Feb. 2021.

[49] M. Boumaraf and F. Merazka, "Secure speech coding communication using hyperchaotic key generators for AMR-WB codec," *Multimedia Syst.*, vol. 27, no. 2, pp. 247–269, Apr. 2021.

[50] P. Sathiyamurthi and S. Ramakrishnan, "Speech encryption algorithm using FFT and 3D-Lorenz–logistic chaotic map," *Multimedia Tools Appl.*, vol. 79, nos. 25–26, pp. 17817–17835, Jul. 2020.

[51] X. Wang, W. Zhang, W. Guo, and J. Zhang, "Secure chaotic system with application to chaotic ciphers," *Inf. Sci.*, vol. 221, pp. 555–570, Feb. 2013.

[52] A. A. Rezk, A. H. Madian, A. G. Radwan, and A. M. Soliman, "Multiplierless chaotic pseudo random number generators," *AEU, Int. J. Electron. Commun.*, vol. 113, Jan. 2020, Art. no. 152947.

[53] B. Karakaya, A. Gülten, and M. Frasca, "A true random bit generator based on a memristive chaotic circuit: Analysis, design and FPGA implementation," *Chaos, Solitons Fractals*, vol. 119, pp. 143–149, Feb. 2019.

[54] F. Yu, Q. Wan, J. Jin, L. Li, B. He, L. Liu, S. Qian, Y. Huang, S. Cai, Y. Song, and Q. Tang, "Design and FPGA implementation of a pseudo-random number generator based on a four-wing memristive hyperchaotic system and Bernoulli map," *IEEE Access*, vol. 7, pp. 181884–181898, 2019.

[55] M. Garcia-Bosque, A. Pérez-Resa, C. Sánchez-Azqueta, C. Aldea, and S. Celma, "Chaos-based bitwise dynamical pseudorandom number generator on FPGA," *IEEE Trans. Instrum. Meas.*, vol. 68, no. 1, pp. 291–293, Jan. 2019.

[56] M. S. Azzaz, C. Tanougast, S. Sadoudi, R. Fellah, and A. Dandache, "A new auto-switched chaotic system and its FPGA implementation," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 18, no. 7, pp. 1792–1804, Jul. 2013.

[57] E. A. A. Hagras and M. Saber, "Low power and high-speed FPGA implementation for 4D memristor chaotic system for image encryption," *Multimedia Tools Appl.*, vol. 79, nos. 31–32, pp. 23203–23222, Aug. 2020.

[58] M. S. Azzaz, R. Fellah, C. Tanougast, and R. Kaibou, "Design and FPGA implementation of TRNG based on a new multi-wing attractor in Lorenz chaotic system," *Eur. Phys. J. Special Topics*, vol. 230, nos. 18–20, pp. 3469–3480, Nov. 2021.

[59] M. J. Barani, P. Ayubi, M. Y. Valandar, and B. Y. Irani, "A new pseudo random number generator based on generalized Newton complex map with dynamic key," *J. Inf. Secur. Appl.*, vol. 53, Aug. 2020, Art. no. 102509.

[60] J. R. Pulido-Luna, J. A. López-Rentería, N. R. Cazarez-Castro, and E. Campos, "A two-directional grid multiscroll hidden attractor based on piecewise linear system and its application in pseudo-random bit generator," *Integration*, vol. 81, pp. 34–42, Nov. 2021.

[61] Z. Peng, W. Yu, J. Wang, Z. Zhou, J. Chen, and G. Zhong, "Secure communication based on microcontroller unit with a novel five-dimensional hyperchaotic system," *Arabian J. Sci. Eng.*, vol. 47, pp. 813–828, Mar. 2021.

[62] M. A. Murillo-Escobar, C. Cruz-Hernández, F. Abundiz-Pérez, and R. M. López-Gutiérrez, "Implementation of an improved chaotic encryption algorithm for real-time embedded systems by using a 32-bit microcontroller," *Microprocessors Microsyst.*, vol. 45, pp. 297–309, Sep. 2016.

[63] M. Murillo-Escobar, F. Abundiz-Pérez, C. Cruz-Hernández, and R. López-Gutiérrez, "A novel symmetric text encryption algorithm based on logistic map," in *Proc. Int. Conf. Commun., Signal Process. Comput.*, vol. 4953, 2014, pp. 1–5.

[64] S. Janakiraman, K. Thenmozhi, J. B. B. Rayappan, and R. Amirtharajan, "Lightweight chaotic image encryption algorithm for real-time embedded system: Implementation and analysis on 32-bit microcontroller," *Microprocess. Microsyst.*, vol. 56, pp. 1–12, Feb. 2018.
[65] M. R. Senouci, S. Sadoudi, B. Djamaa, and M. A. Senouci, "A lightweight efficient chaos-based cryptosystem for constrained-node networks," *Int. J. Commun. Syst.*, vol. 33, no. 10, p. e4215, Jul. 2020.
[66] R. Sujarani, D. Manivannan, R. Manikandan, and B. Vidhyacharan, "Lightweight bio-chaos crypt to enhance the security of biometric images in Internet of Things applications," *Wireless Pers. Commun.*, vol. 119, pp. 2517–2537, Mar. 2021.
[67] Z. Qiao, S. El Assad, and I. Taralova, "Design of secure cryptosystem based on chaotic components and AES S-box," *AEU, Int. J. Electron. Commun.*, vol. 121, Jul. 2020, Art. no. 153205.
[68] E. E. García-Guerrero, E. Inzunza-González, O. R. López-Bonilla, J. R. Cárdenas-Valdez, and E. Tlelo-Cuautle, "Randomness improvement of chaotic maps for image encryption in a wireless communication scheme using PIC-microcontroller via zigbee channels," *Chaos, Solitons Fractals*, vol. 133, Apr. 2020, Art. no. 109646.
[69] Z. Qiao, S. El Assad, and I. Taralova, "Design of secure cryptosystem based on chaotic components and AES S-box," *AEU, Int. J. Electron. Commun.*, vol. 121, Jul. 2020, Art. no. 153205.
[70] S. Aruna and G. Usha, "HPAC-sbox—A novel implementation of predictive learning classifier and adaptive chaotic S-box for counterfeiting sidechannel attacks in an IoT networks," *Microprocessors Microsyst.*, vol. 81, Mar. 2021, Art. no. 103737.
[71] *Security Requirements for Cryptographic Modules*, 2002.
[72] S. Sadoudi, C. Tanougast, and M. S. Azzaz, "A new robust additive hyperchaos masking algorithm for secure digital communications," in *Proc. Int. Conf. Control, Decis. Inf. Technol. (CoDIT)*, May 2013, pp. 501–504.
[73] S. Sadoudi, C. Tanougast, M. S. Azzaz, and A. Dandache, "Design and FPGA implementation of a wireless hyperchaotic communication system for secure real-time image transmission," *EURASIP J. Image Video Process.*, vol. 2013, no. 1, pp. 1–18, Dec. 2013.
[74] S. Sadoudi, C. Tanougast, M. S. Azzaz, and A. Dandache, "Embedded hyperchaotic Lorenz generator for secure communications," in *Proc. IEEE 11th Int. New Circuits Syst. Conf. (NEWCAS)*, Jun. 2013, pp. 1–4.
[75] M. S. Azzaz, C. Tanougast, S. Sadoudi, and A. Dandache, "Real-time FPGA implementation of Lorenz's chaotic generator for ciphering telecommunications," in *Proc. Joint IEEE North-East Workshop Circuits Syst. TAISA Conf.*, Jun. 2009, pp. 1–4.
[76] L. Bassham, III, *et al.*, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," Tech. Rep. NIST SP 800-22 Revision 1a, 2010.
[77] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, and E. Barker, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," Booz Allen Hamilton, McLean, VA, USA, Tech. Rep., 2001.
[78] A. Vaskova, C. López-Ongil, E. S. Millán, A. Jiménez-Horas, and L. Entrena, "Accelerating secure circuit design with hardware implementation of diehard battery of tests of randomness," in *Proc. IEEE 17th Int. On-Line Test. Symp.*, Jul. 2011, pp. 179–181.
[79] *Functionality Classes and Evaluation Methodology for Deterministic Random Number Generators*.
[80] *A Proposal for Functionality Classes for Random Number*.
[81] W. Schindler and W. Killmann, "Evaluation criteria for true (physical) random number generators used in cryptographic applications," in *Proc. Int. Workshop Cryptograph. Hardw. Embedded Syst.* Springer, 2002, pp. 431–449.
[82] P. L'Ecuyer and R. Simard, "TestU01: AC library for empirical testing of random number generators," *ACM Trans. Math. Softw.*, vol. 33, no. 4, pp. 1–40, 2007.
[83] M. S. Turan, E. Barker, J. Kelsey, K. A. McKay, M. L. Baish, and M. Boyle, *Recommendation for the Entropy Sources Used for Random Bit Generation*, Standard NIST SP 800-90B, 2018, p. 102.
[84] J. Hernandez-Castro and D. F. Barrero, "Evolutionary generation and degeneration of randomness to assess the indepedence of the ent test battery," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jun. 2017, pp. 1420–1427.
[85] V. Milanović and M. E. Zaghloul, "Synchronization of chaotic neural networks and applications to communications," *Int. J. Bifurcation Chaos*, vol. 6, no. 12b, pp. 2571–2585, Dec. 1996.
[86] Z. Wu, Y. Xia, and X. Xie, "Stochastic Barbalat's lemma and its applications," *IEEE Trans. Autom. Control*, vol. 57, no. 6, pp. 1537–1543, Jun. 2012.
[87] "LogiCORE IP virtex-6 FPGA embedded tri-mode Ethernet MAC wrapper v2.2 guide," Xilinx, San Jose, CA, USA, Tech. Rep. UG800, 2022.
[88] *Information Processing Systems—Open Systems Interconnection*, 1989.
[89] *User Datagram Protocol*, 1980.
[90] *Management Information Base for the User Datagram Protocol (UDP) draft-ietf-ipv6-rfc2013-update-04*, 2004.
[91] *UDP Usage Guidelines*, 2017.
[92] *Transport Features of the User Datagram Protocol (UDP) and Lightweight UDP (UDP-Lite)*, 2020.
[93] *Internet Protocol DARPA Internet Program Protocol Specification*, 1981.
[94] *Internet Protocol DARPA Internet Program Protocol Specification to 48. Bit Ethernet Address for Transmission on Ethernet Hardware*, 1982.
[95] *IEEE Standard for Ethernet*, 2018.
[96] *Wireshark 3.5.0 Development Release*.
[97] "Xilinx power tools tutorial," Xilinx, San Jose, CA, USA, Tech. Rep. UG733, 2022.
[98] "Chipscope pro 10.1 software and cores user guide," Xilinx, San Jose, CA, USA, Tech. Rep. UG029, 2022.
[99] B. Batmaz and A. Doğan, "1 Gbit/s UDP/IP offload engine IP core with PCIe interface," *J. Circuits, Syst. Comput.*, vol. 27, no. 4, Apr. 2018, Art. no. 1850053.
[100] B. Batmaz and A. Doğan, "CoAP acceleration on FPSoC for resource constrained Internet of Things devices," *IEEE Internet Things J.*, vol. 8, no. 24, pp. 17790–17801, Dec. 2021.
[101] M. Parelkar and D. Jetly, "High performance UDP/IP 40Gb Ethernet stack for FPGAs," in *Proc. Int. Symp. Appl. Reconfigurable Comput.* Springer, 2018, pp. 255–268.
[102] K. De-Wei, Y. Guo-Shun, and L. Xiao-Qiang, "The design and implementation of 10 gigabit Ethernet link based on FPGA," *Microelectron. Comput.*, vol. 36, no. 12, pp. 21–25, 2019.
[103] F. Dridi, S. El Assad, C. Atamech, W. E. Youssef, and M. Machhout, "Design and implementation on FPGA board of a chaos-based stream cipher," in *Proc. 15th Int. Conf. Internet Technol. Secured Trans. (ICITST)*, Dec. 2020, pp. 1–5.
[104] M. Irfan, A. Ali, M. A. Khan, M. Ehatisham-ul-Haq, S. N. M. Shah, A. Saboor, and W. Ahmad, "Pseudorandom number generator (PRNG) design using hyper-chaotic modified robust logistic map (HC-MRLM)," *Electronics*, vol. 9, no. 1, p. 104, Jan. 2020.
[105] B. Bouteghrine, C. Tanougast, and S. Sadoudi, "Novel image encryption algorithm based on new 3-D chaos map," *Multimedia Tools Appl.*, pp. 1–23, Apr. 2021.
[106] U. S. Choi, S. J. Cho, and S. W. Kang, "Color medical image encryption using 3D chaotic cat map and NCA," in *Proc. 10th IFIP Int. Conf. New Technol., Mobility Secur. (NTMS)*, Jun. 2019, pp. 1–5.
[107] Attaullah, T. Shah, and S. S. Jamal, "An improved chaotic cryptosystem for image encryption and digital watermarking," *Wireless Pers. Commun.*, vol. 110, no. 3, pp. 1429–1442, Feb. 2020.
[108] Y. Kim, C. Guyot, and Y.-S. Kim, "On the efficient estimation of min-entropy," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 3013–3025, 2021.
[109] H. Yamamoto and Q. Liu, "Highly sensitive universal statistical test," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2016, pp. 700–704.
[110] S. Park, B. G. Choi, T. W. Kang, K. W. Park, J. J. Lee, S. W. Kang, and J. B. Kim, "Analysis of entropy estimator of true random number generation using beta source," in *Proc. 34th Int. Tech. Conf. Circuits/Syst., Comput. Commun. (ITC-CSCC)*, Jun. 2019, pp. 1–3.
[111] F. Djimasra, J. D. D. Nkapkop, N. Tsafack, J. Kengne, J. Y. Effa, A. Boukabou, and L. Bitjoka, "Robust cryptosystem using a new hyper-chaotic oscillator with stricking dynamic properties," *Multimedia Tools Appl.*, pp. 1–17, Apr. 2021.
[112] A. Li, A. Belazi, S. Kharbech, M. Talha, and W. Xiang, "Fourth order MCA and chaos-based image encryption scheme," *IEEE Access*, vol. 7, pp. 66395–66409, 2019.
[113] T. Bonny, "Chaotic or hyper-chaotic oscillator? Numerical solution, circuit design, MATLAB HDL-coder implementation, VHDL code, security analysis, and FPGA realization," *Circuits, Syst., Signal Process.*, vol. 40, no. 3, pp. 1061–1088, Mar. 2021.
[114] A. Qayyum, J. Ahmad, W. Boulila, S. Rubaiee, Arshad, F. Masood, F. Khan, and W. J. Buchanan, "Chaos-based confusion and diffusion of image pixels using dynamic substitution," *IEEE Access*, vol. 8, pp. 140876–140895, 2020.
[115] S. Zhou, X. Wang, Y. Zhang, B. Ge, M. Wang, and S. Gao, "A novel image encryption cryptosystem based on true random numbers and chaotic systems," *Multimedia Syst.*, vol. 28, pp. 95–112, May 2021.

[116] F. Masood, M. Driss, W. Boulila, J. Ahmad, S. U. Rehman, S. U. Jan, A. Qayyum, and W. J. Buchanan, "A lightweight chaos-based medical image encryption scheme using random shuffling and XOR operations," *Wireless Pers. Commun.*, pp. 1–28, May 2021.
[117] R. I. Abdelfatah, "Secure image transmission using chaotic-enhanced elliptic curve cryptography," *IEEE Access*, vol. 8, pp. 3875–3890, 2020.
[118] C. Cao, K. Sun, and W. Liu, "A novel bit-level image encryption algorithm based on 2D-LICM hyperchaotic map," *Signal Process.*, vol. 143, pp. 122–133, Feb. 2018.
[119] *Cyclone IV Device Handbook*, vol. 1, 2016.
[120] W. S. Sayed, M. F. Tolba, A. G. Radwan, and S. K. Abd-El-Hafiz, "FPGA realization of a speech encryption system based on a generalized modified chaotic transition map and bit permutation," *Multimedia Tools Appl.*, vol. 78, pp. 16097–16127, Jun. 2019.

**BENKHADDRA ILYAS** (Student Member, IEEE) received the State Engineering Diploma degree in telecommunication from the Polytechnic School, Algiers, Algeria, in 2010, and the first master's degree in electrical, electronic engineering, and industrial computing sciences, and the second master's degree in radiocommunication and reliable electronic systems from the University of Lorraine, Metz, France, in 2016 and 2017, respectively. He is currently pursuing the Ph.D. degree with the School of Information and Software Engineering, University of Electronic Science and Technology of China (UESTC). His current research interests include chaos-based cryptography, reliable embedded systems based on MPSOC and NOC, the IoT, and blockchain technology.

**SENOUCI MOHAMMED RAOUF** is currently pursuing the Ph.D. degree with the School of Computer Science and Engineering, University of Electronic Science and Technology of China (UESTC), Chengdu, China. His research interests include applied cryptography, chaotic-based cryptography, network security, the IoT, and blockchain technology.

**SENOUCI ABDELKADER** received the State Engineering Diploma degree in electronics from Odessa High School, former USSR, in 1989, the master's degree in signal processing from the University of Sidi Bel-Abbès, Algeria, in 1996, the magister degree in electronics from the University of Science and Technology, Oran, Algeria, in 2006, and the Ph.D. degree in electronics from the University of Jijel, Algeria, in 2014. His current research interests include cryptography, nonlinear control, and control and synchronization of chaos.

**TANOUGAST CAMEL** received the Ph.D. degree in microelectronic and electronic instrumentation from Henri Poincaré University, Nancy, France, in 2001, and the HDR degree from the University of Metz, in 2009, an authorization/accreditation to supervise research. He joined the Electronic Instrumentation Laboratory of Nancy, Electronic Architectures Group, in 2003, and the Microelectronic and Sensor Interface Laboratory of Metz (LICM), in 2008. He has been the Head Research of the networked adaptive and self-organized systems at LICM. In 2013, he joined LCOMS Laboratory, where he is currently the Deputy Director and the ASEC Team Leader, for research in microelectronics, embedded systems, and smart sensors. He is also a Full Professor with the University of Lorraine, France. He has authored or coauthored more than 120 publications and several books. He has participated to many editorial boards of books and international journals. He has been the supervisor of 21 Ph.D. thesis. His research interests include radio communication, design and implementation of real-time processing architectures, architectural optimization, SoCs and NoCs development, computing vision, image processing, and cryptography.

**SADOUDI SAID**, photograph and biography not available at the time of publication.

**HANG LEI** received the Ph.D. degree in computer science from the University of Electronic Science and Technology of China, China, in 1997. After graduation, he conducted research in the fields of real-time embedded operating systems, operating system security, and program verification, as a Professor with the Department of Computer Science, University of Electronic Science and Technology of China. He is currently a Professor (Doctoral Supervisor) with the School of Information and Software Engineering, University of Electronic Science and Technology of China. His research interests include big data analytics, machine learning, and program verification.