Design and Embedded FPGA Implementations of Chaos-based cryptosystems, for securing IoT data

Fethi Dridi

RÉPUBLIQUE TUNISIENNE

MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ DE MONASTIR

FACULTÉ DES SCIENCES DE MONASTIR

Ecole Doctorale : Matériaux, Dispositifs et Microsystèmes (EDMD)

THÈSE

Présentée pour l'obtention du diplôme de

DOCTORAT DE L'UNIVERSITÉ DE MONASTIR

Faculté des Sciences de Monastir

Spécialité : ÉLECTRONIQUE ET MICROÉLECTRONIQUE

Présentée par :

Fethi DRIDI

Sujet :

Conception et implémentation sur FPGA des systèmes cryptographiques chaotiques : Application à l'Internet des Objets (IdO).

Design and Embedded FPGA Implementations of Chaos-based cryptosystems, for securing IoT data.

Directeur de thèse : **Mohsen MACHHOUT**

Directeur de thèse : **Safwan EL ASSAD**

Soutenue le 25/06/2022 devant la Commission d'Examen composée de :

M. Kamel BESBES	Professeur, FSM	Président
M. René LOZI	Professeur, Université Côte d'Azur	Rapporteur
M. Moez FEKI	Professeur, ESSTHS	Rapporteur
M. Anis SAKLY	Professeur, ENIM	Examinateur
M. Mohsen MACHHOUT	Professeur, FSM	Directeur de thèse
M. Safwan EL ASSAD	Maître de conférences (HDR), NU	Directeur de thèse

---

Laboratoire d'Électronique et Microélectronique (Code : LR99ES30)
Laboratoire IETR UMR 6164, équipe VAADER

Acknowledgments

First, I would like to witness my gratitude to Almighty God, who gave me the courage and strength to pursue this thesis and opened the gates of knowledge for me.

I have started my final internship project for my master's degree in Electronics and Microelectronics Lab Research EμE. It was March 2017. I never thought this training would be the reason for choosing me to pursue a long and fruitful Ph.D. journey full of fluctuations (The 1st registration was on January 23, 2019). What's for sure is that I will never forget this opportunity which was wonderful, overwhelming, and full of lessons and responsibilities. Thankfully, my Ph.D. was completed successfully in which we delivered many journals and international conferences. This wouldn't have happened without the aid and support of countless people over the past four years.

First of all, I am deeply grateful to my advisors ***Mr. Mohsen MACHHOUT****,* ***Mr. Safwan EL ASSAD****, and my co-supervisor* ***Mr. Wajih EL HADJ YOUSSEF*** *who gave me this opportunity and believed in me from the very beginning. Knowing that this Ph.D. experience wasn't always easy, it was always a pleasure, with a lot of excitement, to be up to the challenge and to beat paper deadlines. Working with you improved me a lot as a student, as a researcher, and as a person. It wasn't straightforward for me to understand how to think like a researcher. You have taught me, both consciously and unconsciously, how good work is done. I appreciate all your contributions of time and ideas to make my Ph.D. experience productive. You have always listened to my ideas and discussions with you frequently led to progress. Your ability to approach research problems and your high scientific standards set an example. I admire your ability to balance research interests and personal pursuits. I am thankful for the excellent example you have provided me as a successful and ambitious researcher.*

I would first of all like to thank the members of the jury for their presence, for their careful reading of my thesis as well as for the remarks they will address to me during this defense in order to improve my work. I thank Professors ***René LOZI*** *and* ***Moez FEKI*** *for finding the time to read my thesis and for their valuable feedback. I would like to thank Professor* ***Anis SAKLY*** *for being examinator of this jury. And finally, I would like to thank Professor* ***Kamel BESBES*** *for the honor he gave me when he agreed to chair this jury.*

I would like to thank all EμE laboratory members, especially the Lab head Pr. ***Mohsen MACHHOUT****, and I have been very privileged to get support from many great people who became friends over the last several years. In addition, I would like to thank all IETR laboratory members (the lab where I did my internship and collaboration), but one special thanks go to my director* ***Mr. Safwan EL ASSAD****, for always being by my side during tough times and for his continuous encouragement and especially when I was in France.*

I would like to thank my dear lovely mother in the world, ***Khera****, and my dear soulful father* ***Bechir*** *and the big advisor for me. Also, I thank my dear lovely sister* ***Hanen*** *and my dear feisty brothers* ***Mohamed*** *and* ***Hamdi*** *for their illimitable love. Your encouragements and your faiths in me gave me the strength to achieve my goal.*

I would especially like to thank the love of my life ***Takwa RHIMI*** *who was always at my side and encouraged me all the time.*

Fethi DRIDI

“ “ “ “ “ “ ‘

Abstract

I.o.T devices we use every day are becoming connected entities across the planet. They combine autonomous embedded sensory objects that are resource constrained in terms of computing capabilities, energy and memory capacities. Moreover, these devices can be affected by various categories of security. This imposes to design new cryptographic methods which are efficient in terms of security, time overhead and energy consumption. To meet the requirements set out, especially in terms of security, we have developed, in this thesis, chaos-based cryptographic primitives. First, we implemented (on FPGA board) and evaluated the statistical security and hardware metrics of some chaotic maps, specifically the Skew Tent, PWLCM, Logistic, 3D-Chebyshev map and LFSRs, that are the basic components of the proposed chaotic generators. Based on the previous results, we then designed, implemented (on FPGA board) and analyzed four secure PRNGs-CS and their corresponding stream ciphers. All these chaotic systems uses a predefined coupling matrices M that achieve a weak mixing of the chaotic maps, which avoid, on the one hand, the divide-and-conquer attacks on chaotic maps, and on the other hand, to increase the randomness of the sequences produced as well as their lengths. Besides, the proposed PRNGs-CS contain at least one polynomial map of degree two or three to make very difficult the algebraic attack. The experimental results obtained demonstrate the high degree of security and the good hardware metrics achieved by the proposed chaotic systems. Finally, we designed, implemented and evaluated the performance of a new chaos-based encryption/decryption architecture, operating in CBC mode and uses our developed LSPT-PRNG system. The confusion operation of the system is performed by a strong proposed dynamic circular S-box. The diffusion operation is achieved by a 2-D modified cat map and a Horizontal Addition Diffusion followed by a Vertical Addition Diffusion. The experimental results obtained demonstrate that the proposed cryptosystem can successfully resist various known attacks and therefore can be used to secure sensitive data.

Résumé

Les dispositifs I.o.T que nous utilisons chaque jour deviennent des entités connectées à travers la planète. Ils combinent des objets sensoriels embarqués autonomes qui sont contraints en ressources en termes de capacités de calcul, d'énergie et de capacités de mémoire. De plus, la sécurité de ces dispositifs peut être compromise par différentes attaques cryptographiques. Cela impose de concevoir de nouvelles méthodes cryptographiques performantes en termes de sécurité, de temps de calcul et de consommation d'énergie. Pour répondre aux exigences énoncées, surtout en termes de sécurité, nous avons développé, dans cette thèse, des primitives cryptographiques basées sur le chaos. Tout d'abord, nous avons implémenté (sur carte FPGA) et évalué la sécurité statistique et les métriques matérielles de certaines cartes chaotiques, spécifiquement, la carte Skew Tent, PWLCM, Logistic, 3D-Chebyshev et LFSRs, qui sont les composants de base des générateurs chaotiques proposés. Sur la base des résultats précédents, nous avons ensuite conçu, implémenté (sur carte FPGA) et analysé quatre PRNGs-CS sécurisés et les systèmes de chiffrement par flux correspondants. Tous ces systèmes chaotiques utilisent des matrices de couplage prédéfinies M qui réalisent un faible mélange des cartes chaotiques, ce qui permet d'éviter, d'une part, l'attaque diviser et conquérir sur les cartes chaotiques, et d'autre part, d'augmenter le caractère aléatoire des séquences produites ainsi que leurs longueurs. De plus, les PRNGs-CS proposés contiennent au moins un polynôme de degré deux ou trois pour rendre difficile l'attaque algébrique. Les résultats expérimentaux obtenus démontrent le haut degré de sécurité et les bonnes métriques matérielles obtenues par les systèmes chaotiques proposés. Enfin, nous avons conçu, mis en œuvre et évalué les performances d'une nouvelle architecture de chiffrement/déchiffrement basée sur le chaos, fonctionnant en mode CBC et utilisant notre système LSPT-PRNG développé. L'opération de confusion du système est réalisée par une S-box circulaire dynamique proposée. L'opération de diffusion est réalisée par la carte chaotique 2-D CAT modifiée et une diffusion d'addition horizontale suivie d'une diffusion d'addition verticale. Les résultats expérimentaux obtenus démontrent que le cryptosystème proposé peut résister avec succès à diverses attaques connues et peut donc être utilisé pour sécuriser des données sensibles.

Introduction Générale

Les travaux de recherche de cette thèse ont été effectués pendant les années 2018-2022. Ils ont été réalisés dans le cadre d'une Convention de codirection internationale de thèse signée entre la Faculté des Sciences de Monastir, Tunisie (Laboratoire d'Electronique et micro-électronique) et Nantes Université/Polytech Nantes, France (laboratoire IETR UMR 6164, équipe VAADER). Tous les travaux présentés pendant ces quatre années de thèse s'inscrivent dans le domaine de la cryptographie et de la sécurité de l'information.

Avec le développement des réseaux sans fil de cinquième génération (5G), on s'attend dans les prochaines années à voir s'établir une connexion efficace entre les humains et les machines afin de garantir la flexibilité nécessaire pour gérer des réseaux avec des besoins en qualité de service hétérogènes. Actuellement, les opérateurs font face à divers défis et challenges lors du déploiement des communications IoT à travers les réseaux existants. Inévitablement, l'augmentation du nombre de dispositifs connectés pose des problèmes de charge et de congestion qui auront un impact important sur les systèmes de communication sans fil. Les dispositifs IoT nécessitent principalement une batterie de longue durée de vie, une portée étendue, une capacité plus grande pour prendre en charge des millions de dispositifs avec un coût de déploiement faible. Pour répondre à ces caractéristiques, plusieurs technologies ont été proposées et développées pour assurer la meilleure efficacité des réseaux étendus à faible consommation énergétique (LPWAN).

D'une part, l'IoT laisse entrevoir des promesses considérables, tant pour les entreprises que pour les administrations et les citoyens. D'autre part, son déploiement représente un défi majeur en termes de sécurité et de respect de la vie privée. En effet, des attaques à l'encontre d'objets connectés, toutes plus impressionnantes les unes que les autres, agitent régulièrement la toile du fait de leur impact désastreux. À titre d'exemple, en 2019, une liste de violations de données et de cybers attaques a eu lieu, atteignant une fuite de 114,6 millions d'enregistrements en août 2019 [1]. Whats app [2], Facebook [3], PDF (Portable Document Format) [4] ont tous été exposés en raison des

vulnérabilités de leurs plateformes.

Ces évènements donnent lieu à une double lecture de l'IoT. La première optimiste, qui consiste à voir un progrès social considérable qui aboutira à la mise en œuvre de services intelligents pour une efficience accrue dans divers domaines. La seconde pessimiste, qui consiste à voir un vecteur de fragilisation de la société tout entière en mettant l'accent sur les vulnérabilités déjà exploitées et les prochaines à venir. En plus des besoins en matière de sécurité afin de prévenir les cybers attaques et d'assurer le bon fonctionnement de l'IoT, de nombreux experts ont tiré la sonnette d'alarme concernant les risques d'atteinte à la vie privée. En effet, les données traitées, collectées et transmises par ces objets qui nous entourent révèlent des informations sur nos habitudes, comportements, activités privées voire notre profil psychologique. Considérons le cas des montres connectées qui mesurent le rythme cardiaque, la tension artérielle et les kilomètres parcourus: qu'advient-il des masses de données collectées ? Sont-elles exclusivement utilisées par l'application associée à la montre en question ou sont-elles revendues à des tiers ?

Outre les problématiques liées aux données personnelles, la sécurité est essentielle au déploiement de l'IoT. En effet, les objets étant accessibles à distance de par leur connectivité, des citoyens malintentionnés pourraient être tentés de les faire dysfonctionner. Ainsi, pour répondre aux exigences sécuritaires de l'IoT, il est nécessaire d'avoir recours à différents outils, dont des algorithmes cryptographiques.

Dans ce contexte, cette thèse tente de prendre tous les éléments nécessaires pour mettre en place de nouvelles solutions de sécurité qui permettent de sécuriser n'importe quel contenu de données transmises ou stockées.

Dans ce mémoire, nous nous intéressons à la conception, la réalisation et l'évaluation des performances des générateurs de séquences chaotiques et de cryptosystèmes basés chaos.

Les systèmes chaotiques sont déterministes, ils produisent des signaux de caractéristiques très proches des signaux aléatoires et possèdent une forte sensibilité aux conditions initiales et aux paramètres de contrôle qui forment la clé secrète. Un infime changement dans les conditions initiales ou les paramètres de contrôle provoque un important changement dans la séquence pseudo-chaotique générée par le système. Basé sur ses propriétés, les cryptosystèmes basés chaos, sont une alternative intéressante à la cryptographie standard, pour achever les services de la sécurité telle que la confidentialité, l'intégrité, l'authenticité, la non-répudiation et le contrôle d'accès.

L'objectif principal de la thèse est d'apporter des solutions robustes basées chaos de la sécurité, comme la confidentialité, l'intégrité des données. Les solutions proposées ont un meilleur niveau de sécurité par rapport aux méthodes standards (grâce à la non-linéarité élevée des cartes chaotiques et des clés de chiffrement dynamiques) et leurs débits sont pour la plupart proches de ceux des méthodes standards. D'autre part, elles sont appropriées pour des implémentation logiciel et matériel. Ce travail de recherche, dans

un premier temps, portera sur la conception, l'implémentation et l'analyse de quatre générateurs de séquences chaotiques (PRNGs-CS). Ces générateurs utilisent des cartes chaotiques de base, une matrice de couplage faible et une technique de multiplexage chaotique ou un opérateur XOR pour la fonction de sortie. Ensuite, quatre chiffrements de flux chaotiques sécurisés basés sur les PRNGs-CS proposés sont réalisés. En vue d'approuver expérimentalement et les performances des systèmes chaotiques proposés, nous avons réalisé leur implémentation matérielle sur le composant FPGA PYNQ-Z2. Puis, nous avons développé des couches de substitution (confusion) et de permutation (diffusion), qui sont d'une part, dynamiques, et d'autre part, variables selon la sortie des séquences pseudo-chaotiques produites par l'un des quatre générateurs proposés. Ces couches chaotiques sont utilisées pour la conception et la mise en œuvre d'un chiffrement par blocs efficace en mode CBC.

Cette thèse est structurée en quatre chapitres, une introduction et une conclusion générale. Dans le chapitre I, nous introduirons les concepts de base qui seront nécessaires à la compréhension de la suite de la thèse. Nous commencerons par introduire des notions de base sur la sécurité et nous citons les services essentiels de la sécurité que le système doit assurer. Ensuite, nous introduirons les principes de base cryptographique utilisée dans le domaine de la sécurité de l'information, et nous nous focalisons sur la cryptographie symétrique. En outre, nous expliquons les chiffrements de flux et les chiffrements par blocs et leurs modes de fonctionnement. Enfin, nous relatons l'état de l'art sur les cryptosystèmes basés chaos, après avoir mentionné les propriétés des signaux chaotiques.

Le chapitre II quant à lui aura pour objectif de présenter quelques cartes chaotiques discrètes, notamment les cartes Skew Tent, PWLCM, Logistic, 3D Chebychev et un registre à décalage à rétroaction linéaire (LFSR) comme base des chiffrements par flux et par bloc basés sur le chaos proposés au cours de cette thèse. Une étude détaillée y sera donc effectuée, visant la comparaison de ces cartes chaotiques, en termes de sécurité et surtout en termes de coût matériel, ce qui constitue notre premier apport. Ceci passera par une analyse utilisant aussi bien des outils de simulation numérique tels que la représentation temporelle, l'attracteur dans l'espace des phases du système, auto et intercorrélation, histogramme, et tests de l'Institut National des Normes et de la Technologie (NIST); que des outils pour mesurer les performances de l'implémentation matérielle de ces cartes chaotiques sur une cible FPGA, tels que coût du matériel (en LUT, registres, slices, etc.), fréquence de fonctionnement (en MHz), débit (en Mbit/s) et efficacité (débit /slices). Cette étude permettra ainsi de justifier le choix des cartes chaotiques qui seront utilisées pour la construction des cryptosystèmes robustes proposés dans cette thèse.

En se basant sur les résultats statistiques obtenus dans le chapitre II, nous pouvons confirmer que les cartes chaotiques ne peuvent pas être utilisées seules comme générateur

de séquences pseudo-aléatoires. La question est de savoir comment combiner certaines d'entre elles pour construire des générateurs de séquences pseudo-aléatoires robustes, contre les attaques statistiques et cryptographiques, et efficaces, en matière de coût matériel, de débit et d'efficacité. L'objectif du chapitre III est de concevoir et de mettre en œuvre, de manière efficace et sécurisée, quatre chiffrements de flux basés sur quatre générateurs de nombres pseudo-chaotiques (PRNGs-CS) robustes, notre deuxième contribution, à savoir: LSP-PRNG (combinant la carte Logistic, Skew Tent et PWLCM), LST-PRNG (combinant la carte Logistic, Skew Tent et 3D Chebyshev), LSPT-PRNG (combinant la carte logistic, Skew Tent, PWLCM et 3D Chebyshev), LST_RC-PRNG (combinant la carte logistic, Skew Tent et 3D Chebyshev, chacune dans une cellule récursive). Tous les PRNGs-CS proposés contiennent au moins polynôme de degré 2 (carte logistic) ou de degré 3 (carte 3D Chebyshev) pour rendre très difficile l'attaque algébrique. De même, chacun de ces systèmes chaotiques utilise une matrice de couplage prédéfinie M qui réalise un faible mélange des cartes chaotiques impliquées dans un chiffrement de flux (PRNG-CS) donné, ce qui évite, d'une part, l'attaque "diviser et conquérir" sur les cartes chaotiques, et d'autre part, d'augmenter le caractère aléatoire des séquences produites ainsi que leurs longueurs. En vue d'approuver expérimentalement la validité de ces systèmes chaotique, nous avons effectué leur implémentation matérielle sur la plate-forme FPGA Xilinx XC7Z020 PYNQ-Z2, utilisant le langage de description matériel de circuit intégré à très haut débit (VHDL). Les résultats obtenus confirment l'intérêt de ces systèmes pour leur utilisation dans la sécurité des flux de données.

Dans le chapitre IV, nous avons conçu et implémenté un nouveau système de chiffrement par blocs basé sur le chaos en mode CBC, et nous avons analysé ses performances en termes de sécurité, composant par composant, et de vitesse de calcul, c'est notre troisième contribution. Ce cryptosystème est basé sur une architecture de substitution permutation et diffusion (SPD). Il utilise le système LSPT-PRNG du chapitre III et une nouvelle S-box forte pour effectuer une opération de substitution circulaire (processus de confusion non linéaire). Une carte chaotique 2-D CAT modifiée et une diffusion d'addition horizontale (HAD) suivie d'une diffusion d'addition verticale (VAD) sont introduites pour réaliser le processus de diffusion. Les clés dynamiques du processus de confusion et de diffusion sont alimentées par le système LSPT-PRNG. Nous quantifions les performances, couche par couche et globalement, en nous appuyant sur la panoplie des tests existant dans la littérature (bijective, non-linéarité, corrélation, sensibilité à la clé, temps de calcul, etc.), que nous avons implémenté. Une seule boucle de chiffrement sera suffisante pour atteindre les niveaux de sécurité requis. L'analyse de sécurité indique que le système cryptographique basé sur le chaos proposé est résistant à divers types d'attaques cryptographiques.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Abbreviations

3GPP	3rd Generation Partnership Project
5G	Fifth Generation
AES	Advanced Encryption Standard
B.C.	Before Christ
BIC	Bits Independence Criterion
BR	Bit Reorganization
CBC	Cipher Block Chaining
CFB	Cipher FeedBack
CLB	Configurable Logic Block
CTR	CounTeR
DES	Data Encryption Standard
DP	Differential approximation Probability
DSP	Digital Signal Processing
DUT	Device Under Test
ECB	Electronic CodeBook
ECRYPT	European Network of Excellence for Cryptology
ET	Encryption Throughput
FFs	Flip-Flops
FPGA	Field-programmable gate array
FSTM	Finite Skew Tent Map

GSM	Global System for Mobile Communications
HAD	Horizontal Addition Diffusion
HD	Hamming Distance
IOB	Input Output Block
IoT	Internet of Things
IST	Information Societies Technology
LFSR	Linear-Feedback Shift Register
LP	Linear approximation Probability
LPWAN	Low Power Wide Area Network
LSB	Least Significant Bit
LTE	Long Term Evolution
LUT	LookUp Table
NCpB	Number of needed Cycles per Bytes
NIST	National Institute of Standards and Technology
NLFSRs	Nonlinear Feedback Shift Registers
NPCR	Number of Pixel Change Rate
OFB	Output FeedBack
OTP	One-Time Pad
PCNG	Pseudo Chaotic Number Generator
PDF	Portable Document Format
PN	Pseudo Noise
PRNG	Pseudo Random Number Generator
PRNGs-CS	Pseudo Random Number Generator of Chaotic Sequences
PWLCM	Piecewise Linear Chaotic Map
SAC	Strict Avalanche Criterion

SCA	Side-Channel Attacks
SoCs	Systems on Chips
SPD	Substitution–Permutation Diffusion
SPN	Substitution Permutation Network
UACI	Unified Average Changing Intensity
VAD	Vertical Addition Diffusion
VHDL	Very High Speed Integrated Circuit Hardware Description Language
WNS	Worst Negative Slack

General Introduction

> *If you reveal your secrets to the wind, you should not blame the wind for revealing them to the trees.*

Khalil Gibran, *The Wanderer*, 1932

This document covers the work that I have carried out during the period 2018-2022 as a Ph.D. student. It was carried out within the framework of an international agreement for co-direction of thesis signed between the Faculty of Sciences of Monastir, Tunisia (Laboratory of Electronics and Microelectronics) and Nantes University, France (VAADER team, IETR UMR 6164 laboratory). All results described here are located in the area of cryptography and information security.

With the development of fifth-generation wireless networks (5G), it is expected in the next few years to see an efficient connection between humans and machines to ensure the flexibility needed to manage networks. with heterogeneous quality of service needs. Currently, operators face various challenges when deploying IoT communications across existing networks. Inevitably, the increase in the number of connected devices poses load and congestion issues that will have a big impact on wireless communication systems. IoT devices mainly require long battery life, extended range, the larger capacity to support millions of devices with low deployment cost. To meet these characteristics, several technologies have been proposed and developed to ensure the best efficiency of

wide-area networks with low energy consumption (LPWAN).

On the one hand, the IoT promises significant benefits for businesses, administrations, and individuals in general. On the other hand, its deployment represents a major challenge in terms of security and privacy. Indeed, attacks against connected objects, each more impressive than the last, regularly agitate the web because of their impact or their originality. As an example, in 2019, a list of data breaches and cyber-attacks took place, reaching a leak of 114.6 million records in August 2019 [1]. Whats app [2], Facebook [3], PDF (Portable Document Format) [4] have all been exposed due to the vulnerabilities in their platforms.

These events provide a double reading of the IoT. The first one is optimistic it consists in seeing considerable social progress, which will lead to the implementation of intelligent services for increased efficiency in various sectors. The second is pessimistic, it consists in considering it as a vector of the vulnerability of the whole society, emphasizing the vulnerabilities already exploited and the next to come. In addition to the need for security to prevent cyber-attacks and ensure the good operation of the IoT, many experts have sounded the alert about the risks of invasion of privacy. Indeed, the data processed, collected, and transmitted by these devices that are around us reveal information about our habits, behaviors, private activities, and even our psychological profile. Let's consider the case of connected watches that measure heart rate, blood pressure and kilometers traveled: what happens to the masses of data collected? Are they used exclusively by the application associated with the particular watch or are they sold to third parties?

In addition to the issues related to personal data, security is essential to the deployment of IoT. Indeed, since the devices are connected and can be accessed remotely, malicious persons could be attracted to make them dysfunctional. Thus, to meet the security requirements of IoT, it is necessary to use various tools, including cryptographic algorithms.

In this context, this thesis tries to take all the necessary elements to set up new security solutions that allow to secure any transmitted or stored data content.

In this dissertation, we are interested in the design, realization, and performance evaluation of generators of chaotic sequences and chaos-based cryptosystems.

Chaotic systems are deterministic, they produce signals with characteristics very close to random signals and have a high sensitivity to the initial conditions and control parameters that form the secret key. A small change in the initial conditions or control parameters produces a large change in the pseudo-chaotic sequence generated by the system. Based on these properties, chaos-based cryptosystems are an interesting alternative to standard cryptography for providing security services such as confidentiality, integrity, authenticity, non-repudiation and access control.

The main objective of the thesis is to provide robust chaos-based solutions for data confidentiality. The proposed solutions have a better level of security compared to standard methods (thanks to the high non-linearity of chaotic maps and dynamic encryption keys) and their throughputs are mostly close to those of standard methods. They are suitable for software and hardware implementations. This research work will first focus on the design, implementation, and analysis of four generators of chaotic sequences (PRNGs-CS). These generators use basic chaotic maps, weak coupling matrices, and a chaotic multiplexing technique or XOR operator for the output function. Then, four secure chaos-based stream ciphers are realized, based on the proposed PRNGs-CS. In order to approve experimentally the validity and performance of the proposed chaotic systems, we realized their hardware implementation on the FPGA component PYNQ-Z2 from Xilinx. Finally, we developed substitution (confusion) and permutation (diffusion) layers, which are on the one hand, dynamic, and on the other hand, variable according to the output of the pseudo-chaotic sequences produced by one of the four generators proposed. These chaotic layers are used for the design and implementation of an efficient block cipher in CBC mode.

This thesis is organized into four chapters, an introduction, and a general conclusion. In Chapter I, we introduce the basic concepts that will be necessary to understand the remainder of the thesis. We start by introducing some basic notions about security and show the essential security services that the system must provide. Next, we introduce

the basic cryptographic principles used in the area of information and system security and focus on symmetric cryptography. Also, we explain the principle of stream ciphers and block ciphers and their modes of operation. Finally, we report on the state of the art of chaos-based cryptosystems, after discussing the properties of chaotic systems.

Chapter II presents some discrete chaotic maps, including Skew Tent, PWLCM, Logistique, 3D Chebyshev, and a Linear Feedback Shift Register (LFSR) as a basis for the design of chaos-based stream and block ciphers proposed in this thesis. A detailed study therefore carried out, aiming at the comparison of these chaotic maps, in terms of security and especially in terms of material cost, which constitutes our first contribution. This involves analysis using both numerical simulation tools such as temporal representation, attractor in system phase space, auto and cross-correlation, histogram, and National Institute of Standards and Technology (NIST) tests; and tools to measure the performance of the hardware implementation of these chaotic maps on the FPGA PYNQ-Z2 target, using the very high speed integrated circuit hardware description language (VHDL): LUTs, registers, slices, operating frequency (MHz), throughput (Mbit/s) and efficiency (throughput/slices). This study will thus make it possible to justify the choice of the chaotic maps that will be used for the construction of robust cryptosystems proposed in this thesis.

Based on the statistical results obtained in Chapter II, we can confirm that chaotic maps cannot be used alone as secure key generators. The question is how to combine some of them to build secure key generators, against statistical and cryptographic attacks, and efficient, in terms of hardware cost, throughput, and efficiency. Based on the performance results of chaotic maps, obtained in Chapter II, we designed, implemented and analyzed in Chapter III, four stream ciphers based on four robust and secure pseudo-chaotic number generators (PRNGs-CS), namely: LSP-PRNG (combining the discrete Logistic, Skew Tent and PWLCM map), LST-PRNG (combining the discrete Logistic, Skew Tent and 3D Chebyshev map), LSPTPRNG (combining the discrete Logistic, Skew Tent, PWLCM and 3D Chebyshev map), and LST_RC-PRNG (combining the discrete

Logistic, Skew Tent, and 3D Chebyshev map, each in a recursive cell). All proposed PRNGs-CS contain at least one polynomial map of degree 2 (Logistic map) or degree three (3D Chebyshev map) to make very difficult the algebraic attack. Likewise, each of these chaotic systems uses a predefined coupling matrix M which achieves a weak mixing of the chaotic maps involved in a given PRNG-SC, which avoid, on the one hand, the divide-and-conquer attacks on chaotic maps, and on the other hand, to increase the randomness of the sequences produced as well as their lengths. The hardware implementation is performed on the same Xilinx XC7Z020 PYNQ-Z2 FPGA platform. The results obtained confirm the interest of these realized stream ciphers for their use to secure stream data.

In Chapter IV, we designed and implemented a new chaos-based block cipher in CBC mode, and we analyzed its performance in terms of security component by component and computing speed, this is our third contribution. The proposed cryptosystem is based on a substitution-permutation an diffusion network (SPDN). It uses the LSPT-PRNG system of Chapter III and a new strong S-box to perform a circular substitution operation (non linear confusion process). A 2-D modified cat map and a Horizontal Addition Diffusion (HAD) followed by a Vertical Addition Diffusion (VAD) are introduced to perform the diffusion process. The dynamic keys of the confusion and diffusion process are fed by the LSPT-PRNG system. The proposed architecture makes it possible to achieve high levels of security with a single encryption loop. The security analysis indicates that the proposed chaos-based cryptosystem is resistant to various types of cryptographic attacks.

Chapter I

State of the Art

> *Chaos often breeds life, when order breeds habit.*

Henry Adams, *The Education of Henry Adams*, 1907

Summary

I.1 Introduction

The now-famous expression, "knowledge is power" (in Latin: *"Scientia potestas est"*), was coined by Francis Bacon in 1597 and assures how much knowledge is essential to us, humans. However, on the other side, it means that people are also clung to the idea of protecting their information, and any leakage in this data will grant this knowledge-power to other parties. Here comes the idea of Cryptology originating from the ancient Greek *kryptós* ("hidden, secret") and *lógos* ("word"). It is the practice and study of techniques for secure communication in the presence of adversarial behavior. Cryptology is often—and mistakenly—considered a synonym for cryptography and occasionally for cryptanalysis, but specialists in the field have for years adopted the convention that cryptology is the more inclusive term, encompassing both cryptography and cryptanalysis. Cryptography (from the Greek *gráphein*, "to write") was first mentioned in 1658 and was originally the study of the principles and techniques by which information could be hidden using ciphers and revealed only by the legitimate users employing the shared secret key. It now encompasses the whole area of key-controlled transformations of information into forms that are either impossible or computationally infeasible for unauthorized persons to duplicate or undo. Cryptanalysis (from the Greek *analýein*, "to loosen" or "to untie") was firstly used in 1923. It is the science (and art) of recovering or forging cryptographically secured information without knowledge of the key.

This chapter discusses the fundamental concepts of cryptography in general. In Section I.2, a brief overview of cryptography including the security requirements, foundations, and basic services is presented. Then, in Section I.3, we focus on symmetric cryptography. In this section, we explain the stream ciphers and block ciphers and their modes of operation, confusion, and diffusion primitives. Furthermore, in Section I.4, a brief overview of chaos-based cryptography is made. Finally, Section I.5 concludes the Chapter.

I.2 Cryptography: foundation and basic concepts

In this section, a brief introduction to the history of cryptography including the security requirements, foundations, and basic services is presented.

I.2.1 A Brief Introduction to the History of Cryptography

Historically, cryptography was seen as an art rather than a science. Originally, the hidden forms were attributed to ancient Egypt around 4000 years B.C. [5, 6]. Scribes were communicating at the time by writing messages in hieroglyphs. This skill of writing in hieroglyphs was passed down from father to son and was subsequently broken by Champollion [7]. Later, the Roman emperor Julius Caesar employed another approach, the Caesar cipher, to encrypt his private documents. It substituted every letter in a message with the letter three places later in the alphabet. The original message may be retrieved by executing the inverse transformation, which is now known as decryption. The Caesar cipher was the first known usage of a substitution cipher [8]. In the first half of the twentieth century [9], modern cryptography was established. The Enigma machine, created by German inventor Arthur Scherbius after the end of World War I and patented in 1928, is one of the most famous examples from that period [10].

When compact computers and low-cost PCs were available, the genuine need for security arose. People could buy them for home use starting in the 1970s once large-scale integration enabled the construction of a sufficiently powerful microprocessor on a single semiconductor chip. Then, in the early 1990s, the development of the Internet laid the stage for a whole new era. Although the initial decades of modern cryptography were primarily for military uses, cryptography is today employed in every sector (commerce, finance, industry, health care, etc.) and is considered an inevitable element in our century. The history of cryptology is a fascinating story in itself, but not the scope of this thesis. Kahn's monograph [5] is an excellent resource for learning about the history of cryptology. Nowadays, cryptography is essential in all the devices we own and use; it is at the heart of computer security and communications. Without a doubt, the rise of the Internet of Things has increased the need and importance of cryptography.

Today's cryptography is titled "Modern Cryptography". Kerckhoffs's principle is one of the basic principles of modern cryptography. It was formulated at the end of the nineteenth century by Dutch cryptographer Auguste Kerckhoffs [11]. The principle goes as follows: **"A cryptographic system should be secure even if everything about the system, except the key, is public knowledge"**. In [11], Kerckhoffs covered the solutions of military cryptography that were most up-to-date at that time. They gave a practical, experience-based approach, including six design principles for military ciphers:

1. The system must be practically, if not mathematically, indecipherable.
2. It must not be required to be secret, and it must be able to fall into the hands of the enemy without inconvenience.
3. Its key must be communicable and retainable without the help of written notes, and changeable or modifiable at the will of the correspondents.
4. It must be applicable to telegraphic correspondence.
5. Apparatus and documents must be portable, and its usage and function must not require the concourse of several people.
6. Finally, it is necessary, given the circumstances that command its application, that the system be easy to use, requiring neither mental strain nor the knowledge of a long series of rules to observe.

Kerckhoffs's principle was reformulated (perhaps independently) by Claude Shannon as **"The enemy knows the system"** In that form it is called Shannon's maxim [12]. The concept is that the cipher's security is based only on the keys and not on the algorithm. This illustrates one of the fundamental ideas of modern cryptography. In fact, in cryptography, the key may be compromised and changed with another without the need to modify the cipher. This principle pertains to the security of ciphers in general. We apply this idea when constructing our proposed crypto-systems.

I.2.2 Security Services

Cryptography provides information security by using a set of techniques. A security service is a specific security goal that can be reached by using cryptography. These goals can be all achieved at the same time in one application, or only some of them can be achieved. The main goals are [13]:

- *Confidentiality* is a service used to keep the content of information from all but those authorized to have it. Secrecy is a term synonymous with confidentiality and privacy. There are numerous approaches to providing confidentiality, ranging from physical protection to mathematical algorithms which render data unintelligible

- *Data integrity* is a service which addresses the unauthorized alteration of data. To assure data integrity, one must have the ability to detect data manipulation by unauthorized parties. Data manipulation includes such things as insertion, deletion, and substitution.

- *Authentication* is a service related to identification. This function applies to both entities and information itself. Two parties entering into a communication should identify each other. Information delivered over a channel should be authenticated as to origin, date of origin, data content, time sent, etc. For these reasons this aspect of cryptography is usually subdivided into two major classes: entity authentication and data origin authentication. Data origin authentication implicitly provides data integrity (for if a message is modified, the source has changed).

- *Non-repudiation* is a service which prevents an entity from denying previous commitments or actions. When disputes arise due to an entity denying that certain actions were taken, a means to resolve the situation is necessary. For example, one entity may authorize the purchase of property by another entity and later deny such authorization was granted. A procedure involving a trusted third party is needed to resolve the dispute.

These four basic concepts are the framework to derive the rest of the other security services as access control, digital signatures, authorization, etc.

I.3 Symmetric Ciphers

Modern cryptography can be divided into two major categories, namely symmetric cryptography and asymmetric cryptography [14–16]. Mainly, these two branches differ by the usage of the key. At the time, symmetric encryption algorithms uses the same key for both encryption and decryption. In fact, asymmetric encryption algorithm allows a pair of keys to be used on each side: a public key used for encryption and a private key used for decryption [17]. It is almost 1000 times slower than symmetric cryptography ones. This makes asymmetric cryptography not well-suited to most wireless sensors of the IoT which are low-cost computing devices [18–20].

In this thesis, we focus on symmetric cryptography especially for stream ciphers and block ciphers, and we describe in the next section the above-listed types of symmetric encryption algorithms.

I.3.1 Stream Ciphers

Stream ciphers were inspired by One-Time Pad (OTP) ciphers, which employ a new key for each plaintext. OTP encryption is the only proven safe encryption since it utilizes the key just once and is the same size as the plaintext to be encrypted. This encryption only xors the plaintext bits with the key. When the keys used are completely random, OTP ensures perfect secrecy [12]. However, there are two major issues here. The first is the difficulty in obtaining random numbers, and the second is the impractical solution of exchanging extremely big keys if the plaintext was very huge. As a result, stream ciphers emerged as the solution to these challenges. They produce a pseudo-random stream of bits using a small secret key that the parties may easily communicate. Modern stream ciphers employ this key and an algorithm to produce a very lengthy pseudo-random keystream sequence. This sequence will then be XORed with the provided plaintext to generate the ciphertext. Stream ciphers are generally characterized by their hight speed compared to block ciphers in hardware, and less complex hardware implementation.

A stream cipher generates successive elements of the keystream based on an internal state. This state is updated in essentially two ways: if the state changes independently

of the plaintext or ciphertext messages, the cipher is classified as a synchronous stream cipher. By contrast, self-synchronizing stream ciphers update their state based on previous ciphertext digits.

Several research projects have focused on safe stream cipher designs in recent years. Many of these have been proposed for software and hardware implementation.

In 2004, a project under the Information Societies Technology (IST) Program of the European Network of Excellence for Cryptology (ECRYPT), called "eStream" was tasked with seeking a strong stream cipher [21,22]. The eSTREAM project was a multi-year effort, running from 2004 to 2008, to promote the design of efficient and compact stream ciphers suitable for widespread adoption. A total of 34 algorithms have been submitted to eSTREAM. All of them were designed primarily for software, hardware, or both profiles. As a result of the project, a portfolio of new stream ciphers was announced in April 2008. The eSTREAM portfolio was revised in September 2008 and currently contains seven stream ciphers that fall into two profiles. Profile 1 contains stream ciphers more suitable for software applications with high throughput requirements and contains the following ciphers: HC-128, Rabbit, Salsa20/12, and SOSEMANUK. Profile 2 stream ciphers are particularly suitable for hardware applications with restricted resources such as limited storage, gate count, or power consumption and contain the following ciphers: Grain, MICKEY 2.0, and Trivium.

However, the selected algorithms are still considered unproven and the objective is rather to provide a set of algorithms and design principles for further study. Although the practical cryptanalysis of one of these algorithms shortly after the end of the contest [23] confirms this impression, it remains undeniable that the eSTREAM contest has advanced the state of the art of cryptography for stream ciphers and restored some confidence in this type of algorithm, as evidenced by the inclusion of ZUC in the 3rd Generation Partnership Project (3GPP) Long Term Evolution standards (LTE) [24] for mobile devices. ZUC algorithms have been used for the security part of well-known standards, especially wireless communication protocols [25]. It is part of the 128-EEA3 and the 128-EIA3 protocols used for confidentiality and integrity, respectively, in the wireless transmissions in LTE. This set of protocols has been developed by 3GPP and

GSM association. ZUC cipher [26] has two 128-bit inputs, Key and Initial Vector (IV). It is a word-oriented stream cipher that outputs a 32-bit word. It consists of three main logical parts, namely the Linear Feedback Shift Register (LFSR), the layer for Bit Reorganization (BR), and the final part which is a nonlinear function F.

Nevertheless, until now most of the eSTREAM ciphers are still not definitely secure [27]. Chaos-based stream ciphers are used to enhance the security issue [28].

I.3.2 Block Ciphers

In cryptography, a block cipher is a deterministic algorithm operating on fixed-length groups of bits, called blocks. They are specified elementary components in the design of many cryptographic protocols and are widely used to implement the encryption of large amounts of data, including data exchange protocols. It uses blocks as an unvarying transformation.

A block cipher consists of two paired algorithms, one for encryption, E, and the other for decryption, D [29]. Both algorithms accept two inputs: an input block of size n bits and a key of size k bits; and both yield an $n-bit$ output block. The decryption algorithm D is defined to be the inverse function of encryption, i.e., $D = E^{-1}$. More formally [30,31], a block cipher is specified by an encryption function:

$$E_k(P) := E(K,P) : \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n \tag{I.1}$$

which takes as input a key K, of bit length k (called the key size), and a bit string P, of length n (called the block size), and returns a string C of n bits. P is called the plaintext, and C is termed the ciphertext. For each K, the function $E_K(P)$ is required to be an invertible mapping on $\{0,1\}^n$. The inverse for E is defined as a function:

$$E_k^{-1}(C) := D_K(C) = D(K,C) : \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n \tag{I.2}$$

taking a key K and a ciphertext C to return a plaintext value P, such that: $\forall K : D_K(E_K(P)) = P$.

Block ciphers first appeared in 1977, when NIST chose the “Lucifer” symmetric key

cipher as the data encryption standard (DES) [32]. Then, after several reviews and attacks in many scientific works [33–35], and when double DES was also attacked [36], Triple DES was recommended as the primary encryption approach to be used in the industry. Later, the need for longer plaintext and longer keys led to the famous Advanced Encryption Standard-AES, which was initially known as Rijndael [37,38]. There are too many block ciphers to list them all, but DES and AES are the two most famous examples.

One important type of iterated block cipher known as a substitution–permutation network (SPN) takes a block of the plaintext and the key as inputs and applies several alternating rounds consisting of a substitution stage followed by a permutation stage to produce each block of ciphertext output [39]. The non-linear substitution boxes substitutes the plaintext by the ciphertext, creating Shannon's confusion. The linear permutation boxes then dissipates redundancies, creating diffusion [29, 40]. In cryptography, confusion and diffusion are two properties of the operation of a secure cipher identified by Claude Shannon in 1945 [41]. Confusion refers to making the relationship between the ciphertext and the symmetric key as complex and involved as possible; diffusion refers to dissipating the statistical structure of plaintext over the bulk of ciphertext. This complexity is generally implemented through a well-defined and repeatable series of substitutions and permutations. These properties, when present, work to thwart the application of statistics and other methods of cryptanalysis.

Another type of block cipher is the Feistel cipher (also known as the Feistel network). A Feistel network uses a round function, a function that takes two inputs: a data block and a subkey and returns one output of the same size as the data block [42]. In each round, the round function is run on half of the data to be encrypted, and its output is XORed with the other half of the data. This is repeated a fixed number of times, and the final output is the encrypted data.

Other operations often used in block ciphers include Add-Rotate-XOR (ARX), which uses addition, rotation, and exclusive-OR operations while avoiding the use of S-boxes; block ciphers based on nonlinear feedback shift registers (NLFSRs); and hybrids that combine different types of the above block ciphers.

A block cipher by itself allows encryption only of a single data block of the cipher's

block length. For a variable-length message, the data must first be partitioned into separate cipher blocks. In the simplest case, known as electronic codebook (ECB) mode, a message is first split into separate blocks of the cipher's block size, and then each block is encrypted and decrypted independently. However, such a naive method is generally insecure because equal plaintext blocks will always generate equal ciphertext blocks (for the same key), so patterns in the plaintext message become evident in the ciphertext output [43]. To overcome this limitation, several so-called block cipher modes of operation have been designed [42] and standardized by NIST [44]. The general concept is to use randomization of the plaintext data based on an additional input value, frequently called an initialization vector, to create what is termed probabilistic encryption [31]. These modes are the Cipher Block Chaining (CBC), Cipher FeedBack (CFB), Output FeedBack (OFB), and CounTeR mode (CTR). The CBC mode is used in block ciphers, the other modes are used in stream ciphers.

I.4 Chaos Applications in Cryptography

Nowadays, the necessity to protect data has become an important challenge. Secure communication of sensitive data is not only limited to ensuring the security of military information and diplomatic correspondence or to protect national security but also to secure private human life and commercial sectors. To solve these issues cryptography schemes are used. In this context, the growing popularity of cryptographic techniques leads to well ensure messages and data confidentiality. Chaos theory was first discovered in the computer system by Edward Lorenz in 1963 [45] and was introduced in cryptography by Matthews in the 1990s [46]. Researchers have been attracted by chaos theory in the cryptography field due to its interesting such as deterministic nature, sensitivity to initial conditions, unpredictability, and complex structure. Over the past two decades, several cryptographic systems based on chaotic systems / maps have been proposed such as pseudo random number generators and cipher encryption algorithms. Unfortunately, there are those which suffer from security problems or slow performance. Also, some of them are not suitable to smart devices that have very limited resources in terms of

memory, computing power, and battery supply. For such cipher algorithms that are particularly suited for this purpose, the main challenge is to design lightweight cryptographic ciphers that cope with the trade-offs between security, cost, and performance.

I.4.1 Chaos-Based Stream Ciphers

The protection of information against unauthorized eavesdropping and exchanges is essential, in particular for military, medical, and industrial applications. Nowadays, cryptographic attacks are more and more numerous and sophisticated, consequently, new effective and fast techniques of information protection appeared or are under development. In this context, recent works have focused on designing new chaos-based algorithms, which provide reliable security while minimizing the cost of hardware and computing time. A chaotic system, although deterministic and not true random, has an unpredictable behavior, due to its high sensitivity to initial condition and control parameter which constitute the secret key. It can generate an aperiodic analog signal whenever its phase space is continuous (i.e. with an infinity of values). However, when its phase state is discrete (with a finite set of values), its orbits must be periodic, even with very long period.

In the field of chaos-based digital communication systems, the chaotic signal has been one of the main concerns in recent decades and widely used to secure communication. Then, in chaos-based cryptography, discrete chaotic maps are used in most chaotic systems (encryption, steganography, watermark, hash functions), to generate pseudo-random chaotic sequences with robust cryptographic properties [47–55]. In a stream cipher, the pseudo-random generator (PRNG) is the most important component since all the security of the system depends on it. For this, a new category of pseudo-chaotic number generators (PCNG) has been recently built to secure stream-data [56–59]. These PCNGs use combined chaotic maps because single chaotic maps are not secure for use in stream ciphers.

In 2017, *M. Abu Taha et al.* [60] designed a novel stream cipher based on an efficient chaotic generator; the results obtained from the cryptographic analysis and of common statistical tests indicate the robustness of the proposed stream cipher. In 2018, *Ons et*

al. [61] developed two new stream ciphers based on pseudo chaotic number generators (PCNGs) that integrate discrete chaotic maps and use the weak coupling as well as switching technique introduced by Lozi [62]. Indeed, the obtained results show that the proposed stream ciphers can be used in practical applications including secure network communication.

In 2019, *Ding et al.* [63] proposed a new lightweight stream cipher system based on chaos, a chaotic system, and two Nonlinear Feedback Shift Registers (NLFSRs) are used. The results show that the stream cipher here has good cryptographic characteristics. In 2020, *Abdelfatah et al.* [64] proposed several efficient multimedia encryption techniques based on four combined chaotic maps (Arnold Map, Lorenz Map, Chebyshev Map and Logistic Map) using serial or parallel connections. With the rapid growth of Internet of Things (IoT) technology that connects devices with low power consumption and low computing resources, the hardware implementation of chaotic and non-chaotic ciphers is more adequate than the software implementation. Note that few chaotic encryption systems are realized in the hardware implementation [65–67].

I.4.2 Chaos-Based Block Ciphers

Among the existing state-of-the-art approaches, chaos-based cryptosystem has proved to be widely efficient in data security [68–72] due, on the one hand, to the powerful characteristics of chaotic sequences such as deterministic, unpredictable, random-like behavior, and high sensitivity to initial conditions and control parameters (Secret Key) [73–75], and on the other hand, to their efficient properties of confusion and diffusion performed by chaotic maps [76–80].

Symmetric chaos-based encryption/decryption systems are classified into two categories: block ciphers and stream ciphers. The former is the main symmetric key cryptosystems, as their design is related to Shannon's theory of information security based on confusion and diffusion operations [76, 81]. Their security properties are well studied [82, 83] and these ciphers encrypt the plaintext on a block of bits: 128, 256, 512, 1024, etc.

Compared to traditional symmetric cryptography, chaos-based encryption / decryp-

tion schemes are more flexible (generic design, variable block size, and secret key can be easily enlarged), easier to implement, and have intrinsic security related to properties of chaotic sequences [82]. In addition, some of them use a substitution layer based on key-dependent S-boxes and not on fixes S-boxes as used in the Advanced Encryption Standard (AES) algorithm [84].

A key element of all chaos-based cryptosystems is the chaotic generator, which provides the dynamic keys (encryption keys) for the confusion and diffusion layer. The security, as well as the efficiency of the system, largely depend on the chaotic generator used.

However, most chaos-based cryptosystems in the literature use simple chaotic maps to supply the confusion and diffusion layer. Such cryptosystems can be broken by side-channel attacks [85, 86].

Chen et al. [87] have proposed a fast chaos-based image encryption scheme with a dynamic state variables selection mechanism, which achieves high confusion and diffusion process. *Farajallah et al.* [88] introduced three versions of a chaos-based cryptosystem based on a structure similar to the cryptosystems of *Zhang et al.* [89] and *Fridrich* [77]. The confusion layer is realized by the modified 2-D cat map, and 32-bit and 8-bit logistic maps are used as the diffusion layer for the first two versions, and a modified Finite Skew Tent Map (FSTM) in the third version.

Qiao et al. [90] have proposed a secure cryptosystem based on chaotic components and the AES S-Box is used as a confusion layer [84]. A global diffusion operating on the entire image is carried out using a Horizontal Addition Diffusion (HAD) followed by a Vertical Addition Diffusion (VAD), introduced by *Omrani et al.* [91], then a permutation operation based on the modified 2-D cat map to reinforce the diffusion effect.

Wang et al. [92] published a block cipher cryptosystem using dynamic S-boxes based on a tent map. Their system operates on blocks using different generated S-boxes and uses 32 iterations of substitution and left cyclic shift. On the same idea, *Zhang et al.* [93] proposed a cryptosystem based on a circular S-box as a confusion process and a keystream buffer as a diffusion layer. The system operates on the entire plain image.

I.5 Conclusion

In this chapter, a cryptographic general idea has been presented. First, we presented the foundations and the general concepts of cryptography. The basic security services are listed along with an explanation for each service. The main principle of cryptography Kerckhoffs' principle is explained. After that, we presented the symmetric algorithms. We explained the stream ciphers. Then we explained the block ciphers and their modes of operation. Also, we introduce chaos theory briefly. Finally, we presented a review of block ciphers, pseudo-random number generators, and stream ciphers based on chaotic maps. To sum up, this chapter comes to lay the foundations for the next chapters that manage other deeper aspects of cryptography.

Chapter II

Study of some Chaotic Maps: Statistical Test and Hardware metrics

> *Random numbers should not be generated with a method chosen at random.*
>
> Donald Ervin Knuth, *The Art of Computer Programming*, 1968–2011

Summary

II.1 Introduction

It is known that chaotic signals exhibit very interesting cryptographic properties for data security such as: deterministic, extreme sensitivity to system conditions and parameters (secret key), auto and correlation similar to pseudo-random signals, ergodicity, etc. These chaotic signals can be generated by specific nonlinear dynamic systems. Indeed, simple recurrence equations can generate rich chaotic dynamics if the control parameters are well-positioned. In many simple recurrence equations the right choice of these parameters is done by referring to the bifurcation diagram and the Lyapunov exponent.

Since 1990, many mono and multidimensional chaotic maps and generators have been developed in the literature which is used in various applications such as pseudo-random number sources, communication, and information security, control and automation, etc. We cite, for example, the works that inspired us to implement the proposed chaotic generators [61, 94–100].

In information security, chaotic maps are used in stream cipher [99], block cipher [101], hashing [102], steganography, and digital watermarking [103, 104]. They have the potential to replace traditional pseudo-random number generators such as maximum length PN sequences, Gold and Kasami generators, and so on.

However the use of these chaotic maps alone in practical data security applications is not secure as we will see in this Chapter.

The performance of these chaotic maps is quantified using signal level tests (phase space, auto, and cross-correlation, histogram, Chi-square test, and NIST tests).

The remainder of this chapter is organized as follows. In Section II.2 we define a collection of common and standard security tools useful for measuring the performance of chaotic sequences. Section II.3 presents the security performance and hardware metrics of some discrete chaotic maps including: Skew Tent, PWLCM Logistic, and 3D Chebyshev maps, as base of proposed chaos-based stream ciphers designed during this thesis. Finally, Section II.4 concludes this chapter.

II.2 Common and standard security performance evaluation tools

To quantify the cryptographic properties of the produced pseudo-chaotic sequences, a series of tests must be carried out such as phase space or mapping, discrete variation, histogram, Chi-square test, auto and cross-correlation. Each of the tests measures a particular characteristic, such as the uniformity of the distribution. The overall results of these tests give an idea of the degree of randomness of the sequences generated. The pseudo-random behavior of the sequences generated is closely related to the statistical characteristics of these sequences. NIST tests [105], among others [106], serve as a benchmark for quantifying and comparing the cryptographic properties of pseudo-random binary sequences. besides that, we present the hardware metrics to evaluate the speed performance and Efficiency of PRNGs and stream ciphers. Moreover, in the case of generators of simple structures, the tests of the Lyapunov exponent and the bifurcation diagram make it possible to fix the optimal values of the parameters to reach the chaotic regions.

All these tests are necessary. However, they do not allow to confirm at 100% the robustness of the sequences generated, due to the frequently powerful cryptanalysis.

In this chapter and chapter III, for all experiments, we generate 100 different sequences of 3,125,100 32-bit samples using 100 random secret keys. But we only used 3,125,000 samples per sequence (i.e. 10^8 bits). Indeed, for each key, the first 100 sam-

ples generated per sequence are produced by the system internally but are not used to deviate from the transitional regime.

II.2.1 Phase space

The phase space trajectory is one of the characteristics of the generated sequence that reflects the dynamic behaviour of the system and does not provide any additional information about the system. For all phase space analysis, we plot the mapping of a decimal random sequence formed by 3,125,000 samples.

II.2.2 Correlation analysis

The properties of a random sequence are that the values in the sequences are not repeated or correlated, and the cross-correlation of two sequences x and y (generated with slightly different keys) is close to zero. The correlation coefficient ρ_{xy} of the two sequences x and y is calculated by the following mathematical equations:

$$\rho_{xy} = \frac{Cov\left(x, y\right)}{\sqrt{D\left(x\right)}\sqrt{D\left(y\right)}} \tag{II.1}$$

where:

$$Cov\left(x, y\right) = \frac{1}{N}\sum_{i=1}^{N}\left[x_i - E\left(x\right)\right]\left[y_i - E\left(y\right)\right] \tag{II.2}$$

$$E\left(x\right) = \frac{1}{N}\sum_{i=1}^{N}x_i \tag{II.3}$$

$$D\left(x\right) = \frac{1}{N}\sum_{i=1}^{N}\left[x_i - E\left(x\right)\right]^2 \tag{II.4}$$

with xi and yi are the values of x and y respectively. We produce two sequences of a random numbers computed with nearby initial conditions, each composed of 3,125,000 samples, to analyse its correlation properties.

II.2.3 NIST test analysis

One of the most popular standards for investigating the randomness of binary data is the NIST statistical test [107]. This test is a statistical package that consists of 188 tests

and sub-tests (globally 15 different tests) that were proposed to assess the randomness of arbitrarily long binary sequences. Those tests focus on a variety of different types of non-randomness that could exist in a sequence. For each test, a P-value is calculated to indicate the result of the test. A *P-value* ≥ 0.01 would mean that the sequence would be considered to be random with a confidence of 99 %. A *P-value* ≤ 0.01 would mean that the conclusion was that the sequence is non-random with a confidence of 99 %.

To apply the NIST test, we generate 100 different sequences of 3,125,000 32-bit (i.e. 10^8 bits) samples each, using 100 random secret keys.

II.2.4 Histogram analysis

Another key property of any robust pseudo chaotic number generator is to provide a uniform distribution in the whole phase space.

II.2.5 Chi-square test analysis

We apply the Chi-square test in order to assert the uniformity of generated sequences. The statistical Chi-square test χ^2 is calculated by the following formula:

$$\chi^2_{exp} = \sum_{i=0}^{N_c-1} \frac{(O_i - E_i)^2}{E_i} \tag{II.5}$$

where, $N_c = 1000$ is the number of classes, O_i is the number of calculated samples in the ith class E_i and $E_i = Ns/N_c$ is the expected number of samples of a uniform distribution.

To prove the uniformity of a generated sequence, the experimental value of Chi-square must be lower than the theoretical one $\chi^2_{exp} < \chi^2_{th}$. Also, more the experimental value of Chi-square is smaller than the theoretical one, better is the uniformity of the generated sequence.

II.2.6 Hardware metrics

Field-programmable gate arrays (FPGAs) are reprogrammable integrated circuits that contain an array of programmable logic blocks. FPGA chip adoption is driven by their

flexibility, hardware-timed speed and reliability, and parallelism.

FPGAs consist of three types of modules:

- Configurable Logic Block (CLB): It is a fundamental building block of an FPGA. Each CLB contains multiple slices. Each slice consists of function generators (LUTs), registers, multiplexers, carry logic etc.
- Switch matrix or interconnection wires: Signals between CLBs and from CLBs to IOBs are routed through Switch matrices.
- IO blocks (IOB): An IOB is the basic input/output function block

Fig. II.1 shows the architecture of an FPGA. It includes logic blocks, input/output (I/O) cells, and interconnecting matrix.

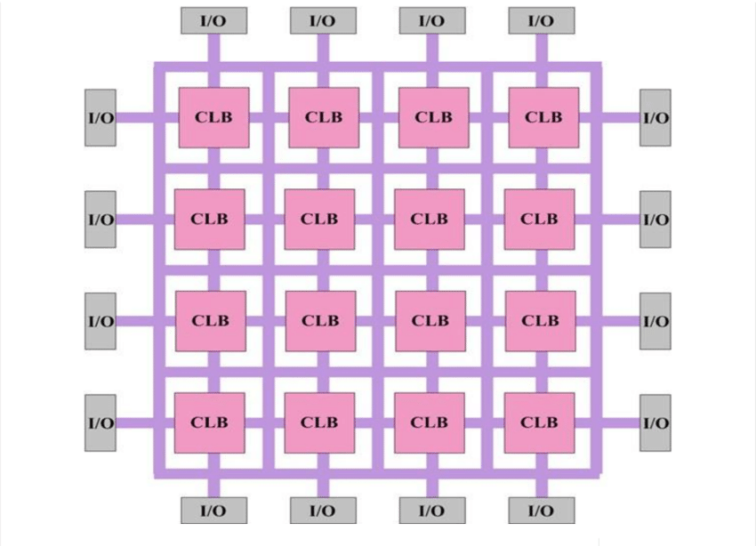


Figure. II.1: An FPGA Architecture Overview

Various FPGA families differ in the way flip-flops and LUTs are packaged together, so it is important to understand flip-flops and LUTs.

Flip-Flops (FFs): are binary shift registers used to synchronize logic and save logical states between clock cycles within an FPGA circuit. On every clock edge, a flip-flop latches the 1 or 0 (TRUE or FALSE) value on its input and holds that value constant until the next clock edge. The Flip-Flop Symbol is shown in Fig. II.2:

Look-Up Tables (LUTs): much of the logic in a CLB is implemented using very small amounts of RAM in the form of LUTs. It is easy to assume that the number

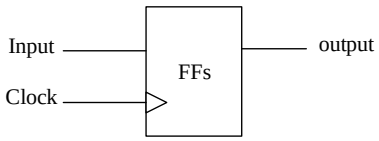


Figure. II.2: Flip-Flop Symbol

of system gates in an FPGA refers to the number of NAND gates and NOR gates in a particular chip. But, in reality, all combinatorial logic (ANDs, ORs, NANDs, XORs, and so on) is implemented as truth tables within LUT memory. A truth table is a predefined list of outputs for every combination of inputs. We give in Fig. II.3 an example of a 6-input LUT.

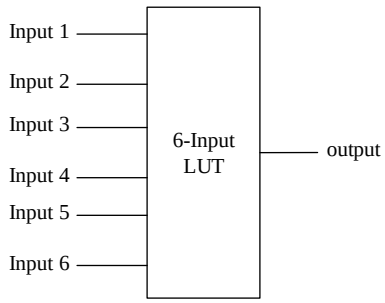


Figure. II.3: Six-Input LUT

The FPGA design flow is shown in the figure II.4 below:

This step involves analysis of the design requirements and functional simulation. The output of this step is a document which describes the design architecture, structural blocks, their functions and interfaces. Design engineer design the architecture according to system specification.

Design Entry: the design is described in a formal hardware description language (HDL). The most common HDLs are VHDL and Verilog. This step involves writing of test environments and behavioral models. Indeed, simulation and design verification takes place throughout the FPGA design flow. We present in Fig. II.5 an example of a testbench.

A testbench is used to verify the specified functionality of a design. It provides the stimuli for the Device Under Test (DUT) and analyses the DUT's responses or stores

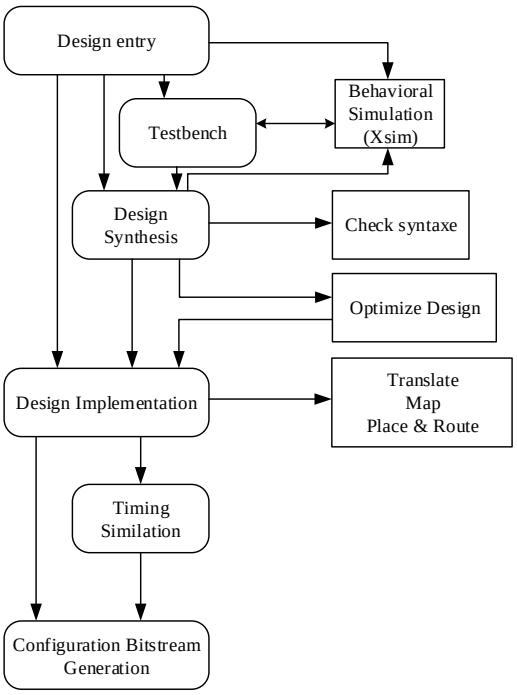


Figure. II.4: FPGA design flow

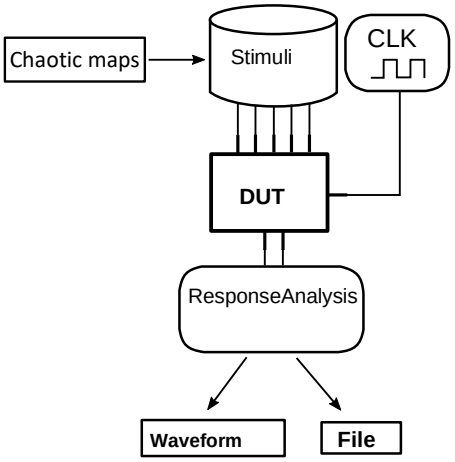


Figure. II.5: Example of a testbench

them in a file. Simulation tools visualize signals by means of a waveform which the designer compares with the expected response. In case the waveform does not match the expected response, the designer has to correct the source code.

Design Synthesis: design synthesis will read the design entered, and will generate the Gate Level Netlist supported for implementation.

Design Implementation: a synthesis tool generate netlist which is mapped onto particular device's internal structure. The main phase of the implementation step is place and route which allocates FPGA resources such as logic cells and connection wires. Then these configuration data are written to a special file by a program called bitstream generator.

We analyze the performance of the implementation of the chaotic maps in terms of resources used (area, DSP), speed (maximum frequency—Max. Freq., throughput), and efficiency (in terms of throughput/slices). The efficiency parameter gives us an overall idea of the hardware metrics performance of the implementation. Furthermore, we give the power consumption.

$$Max.Freq. = \frac{1}{Ti - WNSi}\,[MHz]\,. \tag{II.6}$$

where: Ti is the target clock period (ns) used during the implementation run "i" and $WNSi$ is the Worst Negative Slack (ns) of the target clock used during the implementation run "i" and must be positive, or very close to zero.

$$Throughput = N \times Max.Freq.\,[Mbps]\,. \tag{II.7}$$

$$Efficiency = \frac{Throughput}{Slices}\,[Mbps/Slices]\,. \tag{II.8}$$

In this thesis, all studied chaotic maps and pseudo chaotic generators were implemented on a Xilinx XC7Z020 PYNQ-Z2 FPGA hardware platform and have been coded in VHDL using the Xilinx Vivado design suite (V.2017.2).

PYNQ is an open-source project from Xilinx that makes it easy to design embedded systems with Xilinx Zynq Systems on Chips (SoCs). The programmable logic of the ZYNQ XC7Z020 provides 13,300 logic slices, each with four 6-input LUTs and 8 flip-flops, 630 KB block RAM, and 220 DSP slices.

II.3 Performance Analysis of some Chaotic Maps

Chaotic maps are dynamic systems defined in real by recurrence relations, given by the following equation:

$$x_i(n) = f(x_1(n-1), x_2(n-1), ..., x_m(n-1)), \ i = 1, 2, ..., m \tag{II.9}$$

Where $x \in S$, $f : S \to S$ is a function with m variables, $S \subset [0, 1]$ *or* $[-1, 1]$.

Some one-dimensional chaotic maps (1D) such as the Logistic map [108], the Piecewise Linear Chaotic Maps (PWLCM), and the Skew Tent [99], and two-dimensional chaotic maps (2D) such as the Backer map [109], the Cat map, the standard map, and the Lozi map [110,111], and Three-dimensional chaotic maps (3D) such as 3D Chebyshev map [112] are studied in the literature and widely used for the design of random number generators and in chaos-based crypto-systems.

The chaotic generators proposed in this thesis are based on the following discrete chaotic maps: Logistic map, Skew Tent map, PWLCM map, and 3D Chebyshev map using finite precision N = 32 bits and also an Linear Feed Back Sift Register (LFSR) of 32-bit.

As shown in Fig. II.6, the chaotic map is a dynamical system expressed by mathematical equations. In general, a chaotic map, has the initial value(s) X_0, the control parameters as the secret key K, then determine iteratively, the output $X(n)$ for a given n.

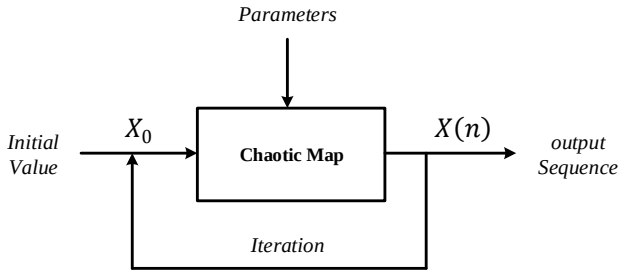


Figure. II.6: The general chaotic map architecture

In this section, we consider the real and discrete equations of the four 1D non-linear chaotic maps used: Skew Tent, PWLCM, Logistic, and the 3D Chebyshev chaotic map

and also an LFSR of 32-bit. These chaotic maps form the basic component of the proposed pseudo-chaotic number generators.

Note that only, in the case of the Logistics map, the bifurcation diagram and the Lyapunov exponent are given in real. Indeed, in the discrete case, the control parameter is fixed at its optimal value = 4.

II.3.1 Performance Evaluation of the Skew Tent map

The Skew Tent map is a piecewise linear map, given by the following equation [113,114]:

$$x(n) = f(x(n-1), p) = \begin{cases} \frac{x(n-1)}{p} & if\ 0 \leq x(n-1) \leq p \\ \\ \frac{1-x(n-1)}{1-p} & if\ p < x(n-1) \leq 1 \end{cases} \tag{II.10}$$

where: $f : S \to S, S = [0, 1]$, $x(n) \in S$ and p is the control parameter of the chaotic map which varies in the following interval: $0 < p < 1$.

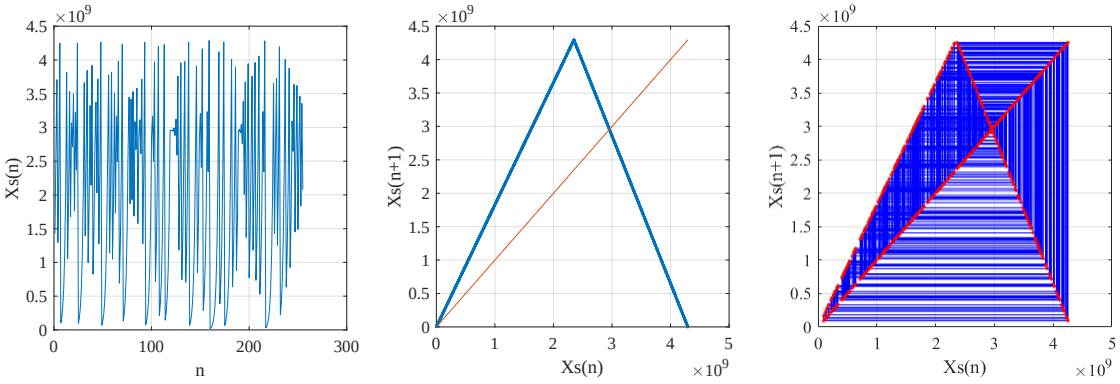
The equation of the discretized Skew Tent map is defined by:

$$XS(n) = \begin{cases} \left\lfloor \frac{2^N \times XS(n-1)}{P_s} \right\rfloor & if\ 0 < XS(n-1) < P_s \\ \\ \left\lfloor 2^N \times \frac{[2^N - XS(n-1)]}{2^N - P_s} \right\rfloor & if\ P_s < XS(n-1) < 2^N \\ \\ 2^N - 1 & otherwise \end{cases} \tag{II.11}$$

Where $XS(n)$ takes an integer value that belongs to the interval $[1, 2^N - 1]$ and P_s is the control parameter of the Skew Tent map, in the range $[1, 2^N - 1]$. And, $\lfloor Z \rfloor$ (*Floor function*) is the greatest integer less than or equal to Z and $X(n)$ takes an integer values $\in [1, 2^N - 1]$ and $N = 32$ is the precision used.

❖ Phase space

In fig II.7 we present respectively the discrete variation, mapping, and attractor (the signature) of the discrete Skew Tent map, for the initial condition $XS(0) = 2,816,384,857$ and the control parameter P_s equal to 2,348,838,240.

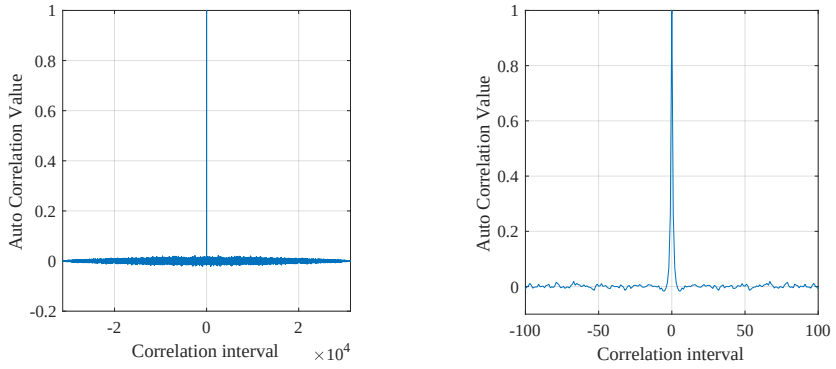


(a) Discrete variation (b) Phase space (c) Attractor

Figure. II.7: Discrete variation, phase space trajectory, and attractor of sequence $Xs(n)$ generated by the discrete Skew Tent map.

❖ Correlation analysis

In Fig. II.8 and Fig. II.9, we give the obtained results of the auto and cross-correlation respectively; also, zooming of these results is presented. As we can see, the correlation between the generated sequences Xs and $Xs' = 2,816,384,858$ (a sligt modification compared to Xs) is near zero ($\rho_{Xs,Xs'} = -0.000016$).



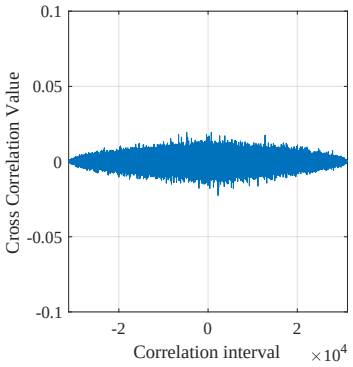
(a) Auto-correlation of sequence Xs

(b) A zoom of the auto-correlation on 200 samples of sequence Xs

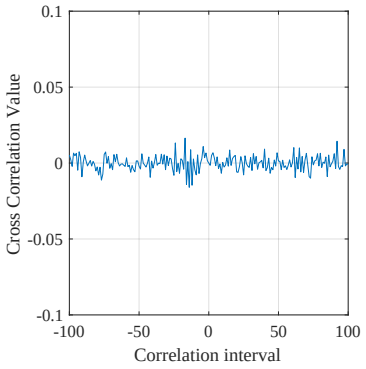
Figure. II.8: Auto-correlation of sequence Xs generated by Skew Tent map.

❖ NIST test analysis

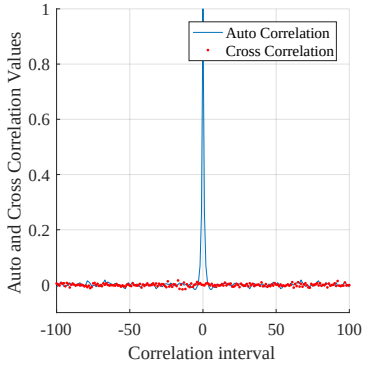
The results of the NIST test on an example of generated sequence $Xs(n)$ are shown in Fig. II.10 and Table II.1. The results obtained show that the $Xs(n)$ sequence do not



(a) Cross-correlation of sequences Xs and Xs'



(b) A zoom of the cross-correlation on 200 samples of sequences Xs and Xs'



(c) A zoom of the cross-correlation of sequences Xs and Xs' and of the auto-correlation of sequence Xs'

Figure. II.9: Cross-correlation functions of sequences Xs and Xs' generated by the Skew Tent map.

pass all the NIST tests, the number of successful tests is higher compared to the results given by the Logistic map for an $Xl(n)$ sequence (see section II.3.3, paragraph II.3.3).

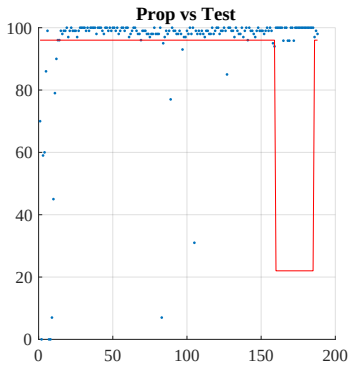


Figure. II.10: NIST test results of the Skew Tent map.

Table II.1: P-values and Proportion results of NIST test for the Skew Tent map.

Test	P-value	Proportion (%)
Frequency test	0.000	70.000
Block-frequency test	0.000	0.000
Cumulative-sums test	0.000	59.000
Runs test	0.000	86.000
Longest-run test	0.163	99.000
Rank test	0.000	0.000
FFT test	0.000	0.000
Nonperiodic-templates	0.430	96.284
Overlapping-templates	0.983	99.000
Universal	0.000	95.000
Approximate Entropie	0.014	94.000
Random-excursions	0.264	99.479
Random-excursion-variant	0.252	99.306
Serial test	0.083	98.000
Linear-complexity	0.384	98.000

❖ Histogram analysis

Fig. II.11 shows the histogram of a generated sequence $Xs(n)$. Visually, the histogram obtained seems not to be uniform, but it is more uniform than that of the Logistic map (see section II.3.3, Fig. II.25).

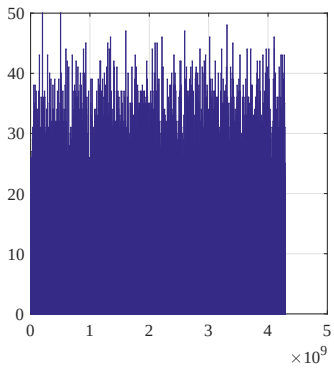


Figure. II.11: Histogram of a sequence Xs generated by the discrete Skew Tent map.

❖ Chi-square test analysis

The experimental value of the Chi-square test is greater than the theoretical value ($\chi^2_{th} = 1073.64$, $\chi^2_{exp} = 1113.45$) asserting that the Skew Tent map is not uniform.

❖ **Hardware metrics**

The implementation of the Skew Tent map was realized on the PYNQ Z-2 FPGA prototyping board from Xilinx. For implementation, the Skew Tent code was written in VHDL with 32-bit fixed-point data formats, then synthesized, and implemented using the Xilinx Vivado design suite (V.2017.2). Vivado design tools essentially made it possible to carry out the various steps from design to implementation on the target FPGA board. It allows, among other things, description, synthesis, simulation, and implementation of a design, then programming it on a chip from one of the different families of Xilinx FPGAs. In Fig. II.12, we summarize the different steps of the design flow that were performed under Vivado for the performance evaluation of the discrete Skew Tent map. Note that the same steps for all studied chaotic maps such as PWLCM, Logistic, 3D-Chebyshev, and the pseudo chaotic generator were repeated since Design entry up to configuration bitstream generation.

The top level block diagram of the FPGA-based chaotic Skew Tent map has been illustrated in Fig. II.13.

One bit signals, namely, Clk, Clk_enable, and reset have been presented at the inputs of the component Skew Tent map. These signals have an obligation for the timing of the whole component and synchronizing these components with their connected system. The initial values, which are essential for the system startup, are embedded into the design for the distension of FPGA chip resource utilization. In the implemented FPGA-based chaotic map, there is one 32-bit output signal X(n) that has convenience with fixed-point number standard and one-bit ce_out signal to indicate if the output signal is ready. The results obtained from Xilinx Xsim Simulator is illustrated in Fig. II.14.

In the following, we provide an overview of our proposed hardware implementation results. The area, speed, efficiency, and power consumption performances of the proposed designs are obtained by the implementation of our VHDL code using Vivado design suite (V.2017.2). The areas of the Skew Tent map implementations on FPGA are compared using slices, Flip-Flops, and lookup tables (LUTs), which are the basic logic block of Xilinx FPGAs. Latency, maximum frequency, and throughput together to determine the speed of execution. Efficiency parameter represents throughput to slices ratio and

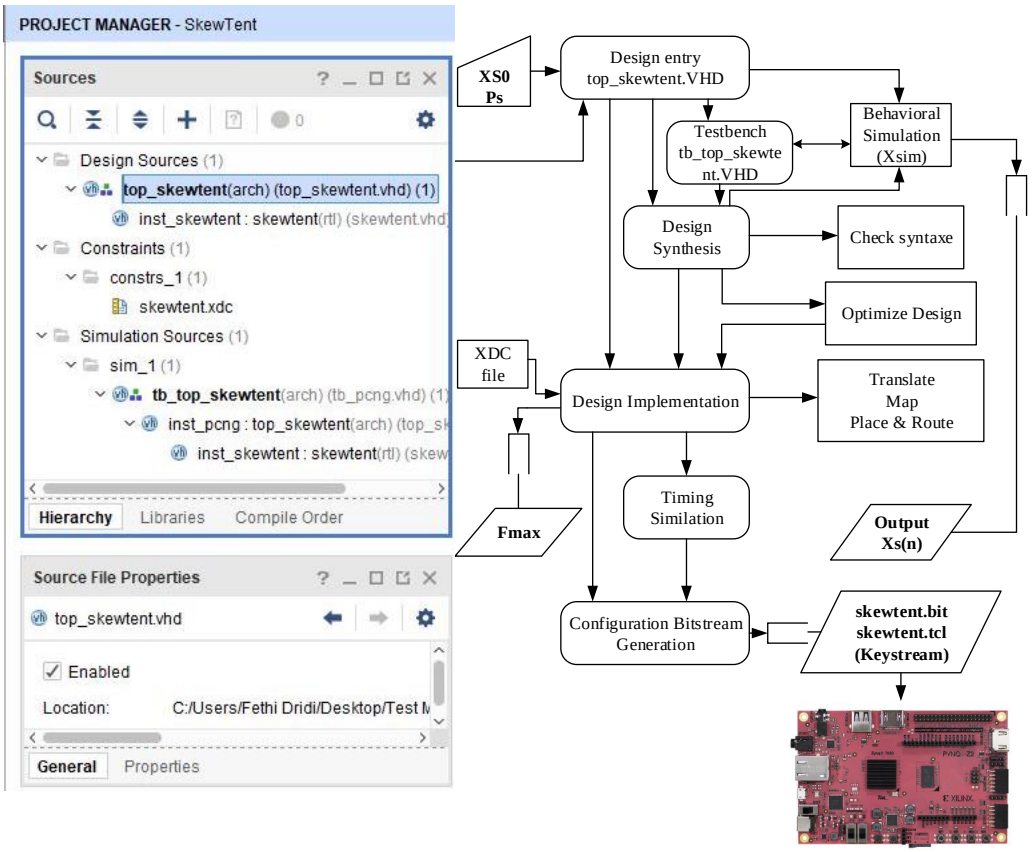


Figure. II.12: FPGA conception flow (under Vivado) of the Skew Tent map.

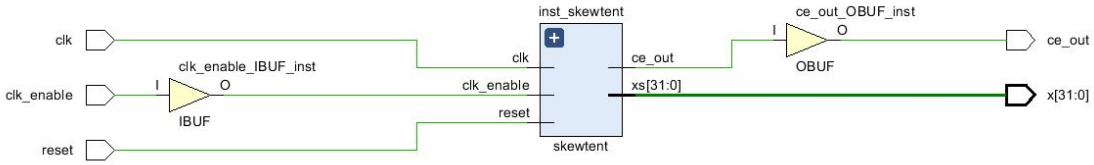


Figure. II.13: RTL architecture of the Skew Tent map.

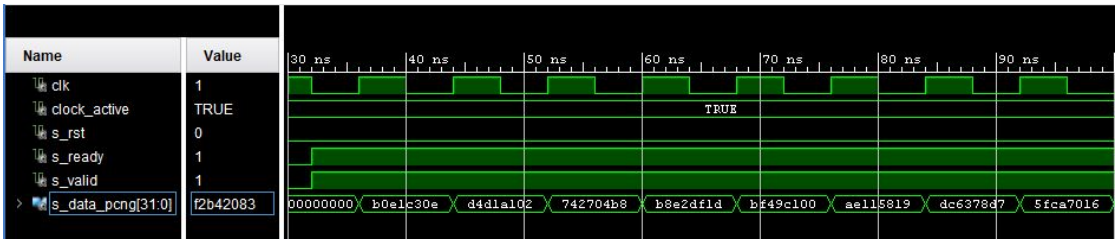


Figure. II.14: Xilinx Xsim Simulator results of the Skew Tent map.

gives us an overall idea of the hardware metrics performance of the implementation.

To get a good insight upon the efficiency, and performance, The Skew Tent map is implemented on the PYNQ Z-2 (7z020clg400-1) Xilinx FPGA as target platforms. The design have been tested after place and route using simulation to ensure the correct functionality.

At the end of the Place & Route process which is performed after synthesizing of chaotic system, PYNQ Z-2 FPGA chip statistics have been obtained and given in Table II.2.

Table II.2: Hardware metrics of the Skew Tent map.

Resources used	Area	LUTs	2,830 /5.32%
		FFs	57 /0.05%
		Slices*	853 /6.41%
	DSPs		0 /0.00%
Speed	WNSi [ns]		0.059
	Ti [ns]		28
	Max. Freq. [MHz]		35.78
	Throughput [Mbps]		1,145.27
Efficiency [Mbps/Slices]			1.342
Power [W]			0.070

Note: Each slice contains four LUTs with 6-input and eight FFs.

The Skew Tent map uses 2,830 LUTs and 57 FFs. Our implementation produces one sample of the Skew Tent map at each clock cycle and can operate at 35.78 MHz with a throughput of 1,145.27 Mbps using equation II.6 and II.7. We note that the Skew Tent map is slower than the Logistic and the 3D-Chebyshev map presented in the following sections.

II.3.2 Performance Evaluation of the PWLCM map

The Piecewise Linear Chaotic Maps (PWLCM) is another piecewise linear chaotic map, described by the following equation:

$$x(n) = f(x(n-1), p) = \begin{cases} \frac{x(n-1)}{p} & if\ 0 \leq x(n-1) \leq p \\ \frac{[x(n-1)-p]}{0.5-p} & if\ p \leq x(n-1) \leq 0.5 \\ f(1-x(n-1), p) & otherwise \end{cases} \quad \text{(II.12)}$$

where: $x(n) \in [0,1]$ and p the control parameter is such that: $0 < p < 0.5$.

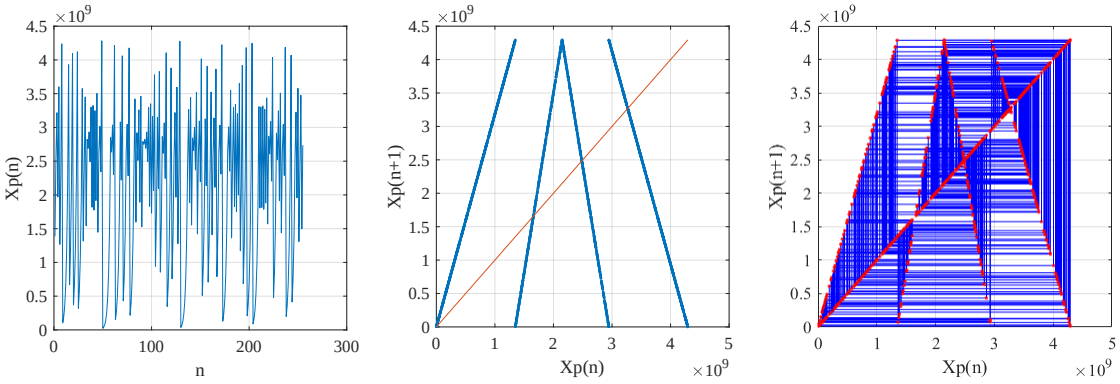
Due to its good cryptographic characteristics, compared to the other chaotic maps studied in this chapter, PWLCM map is widely used in data encryption. The equations of the Discrete PWLCM map is given by [115]:

$$XP(n) = \begin{cases} \left\lfloor \frac{2^N \times XP(n-1)}{P_p} \right\rfloor & if\ 0 < XP(n-1) < P_p \\ \left\lfloor 2^N \times \frac{[XP(n-1)-P_p]}{2^{N-1}-P_p} \right\rfloor & if\ P_p < XP(n-1) < 2^{N-1} \\ \left\lfloor 2^N \times \frac{[2^N - XP(n-1)-P_p]}{2^{N-1}-P_p} \right\rfloor & if\ 2^{N-1} < XP(n-1) < 2^N - P_p \\ \left\lfloor 2^N \times \frac{[2^N - XP(n-1)]}{P_p} \right\rfloor & if\ 2^N - P_p < XP(n-1) < 2^N \\ 2^N - 1 & otherwise \end{cases} \quad \text{(II.13)}$$

Where P_p is the control parameter of the PWLCM map, in the range $[1, 2^{N-1} - 1]$.

❖ Phase space

We give in the Fig. II.15 the discrete variation, phase space, and attractor of a sequence Xp produced by the discrete PWLCM map, with $P = 1,348,838,240$, and an initial state $Xp(0) = 830,235,384$. Resulted attractor shows the signature of the PWLCM map.

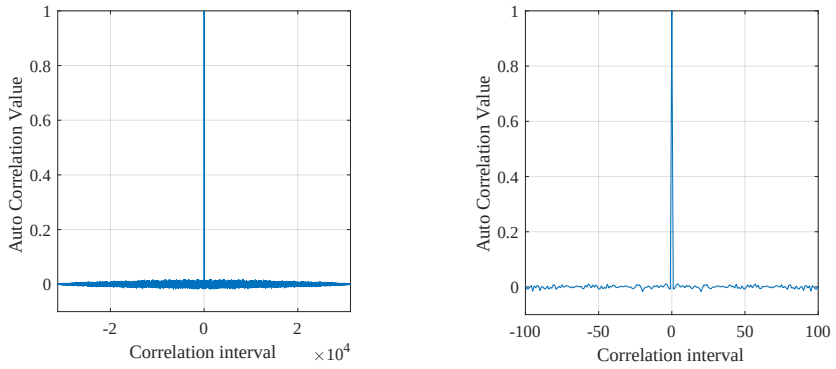


(a) Discrete variation (b) Phase space (c) Attractor

Figure. II.15: Discrete variation, phase space trajectory, and attractor of the sequence $Xp(n)$ generated by the discrete PWLCM map.

❖ Correlation analysis

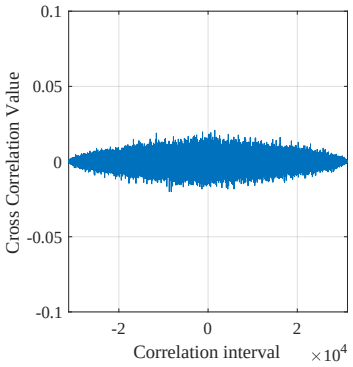
We give in Fig. II.16 and Fig. II.17, the auto-correlation function of the Xp sequence and the cross-correlation functions of the Xp and Xp' sequences generated (with the same condition as the Logistic map) by the PWLCM map respectively; also, zooming of these results is presented. The value of the correlation coefficient $\rho_{Xp,Xp'}$ obtained is very low ($\rho_{Xp,Xp'} = -0.0113$). Therefore, there is no correlation between the generated sequences.



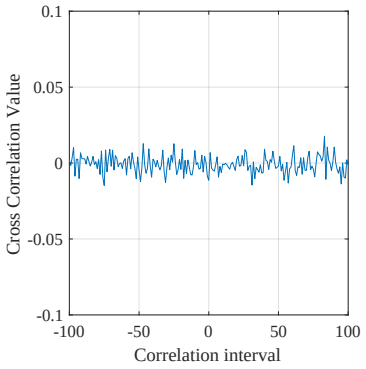
(a) Auto-correlation of sequence Xp

(b) A zoom of the auto-correlation on 200 samples of sequence Xp

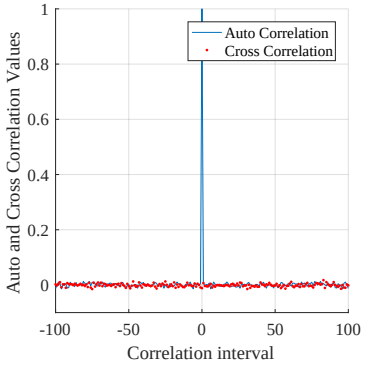
Figure. II.16: Auto-correlation of sequence Xp generated by PWLCM map.



(a) Cross-correlation of sequences Xp and Xp'



(b) A zoom of the cross-correlation on 200 samples of sequences Xp and Xp'



(c) A zoom of the cross-correlation of sequences Xp and Xp' and of the auto-correlation of sequence Xp'

Figure. II.17: Cross-correlation functions of sequences Xp and Xp' generated by the PWLCM map.

❖ NIST test analysis

The NIST test results of a sequence Xp generated by the discrete PWLCM map are shown in Fig. II.18 and Table II.3. Some sub-tests failed, but they are close to the acceptance threshold. In addition, we note that the PWLCM map outperforms the other chaotic maps studied, Skew Tent and Logistic maps, in terms of security.

❖ Histogram analysis

The histogram of a sequence Xp generated by the discrete PWLCM is shown in Fig. II.19.

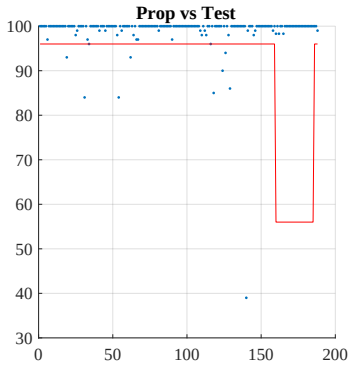


Figure. II.18: NIST test results of the PWLCM map.

Table II.3: P-values and Proportion results of NIST test for the PWLCM map.

Test	P-value	Proportion (%)
Frequency test	0.000	70.000
Block-frequency test	0.000	0.000
Cumulative-sums test	0.000	59.000
Runs test	0.035	86.000
Longest-run test	0.000	99.000
Rank test	0.534	0.000
FFT test	0.475	0.000
Nonperiodic-templates	0.017	96.284
Overlapping-templates	0.000	99.000
Universal	0.290	95.000
Approximate Entropie	0.000	94.000
Random-excursions	0.217	99.479
Random-excursion-variant	0.330	99.306
Serial test	0.000	98.000
Linear-complexity	0.071	98.000

❖ Chi-square test analysis

The experimental value obtained from the previous histogram of the Chi-square test is $\chi^2_{exp} = 1146.92$, which is greater than the theoretical value of $\chi^2_{th} = 1073.64$. As a result, the discrete PWLCM map's distribution is non-uniform.

❖ Hardware metrics

Table II.4 shows the PWLCM map's hardware metrics. Compared to the other chaotic maps, the PWLCM is slower than the Skew Tent, the Logistic, and the 3D-Chebyshev map and uses approximately three times more hardware resources than the Skew Tent map.

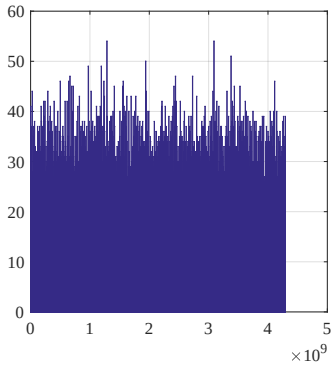


Figure. II.19: Histogram of the sequence Xp generated by the discrete PWLCM map.

Table II.4: Hardware metrics of the PWLCM map.

Resources used	Area	LUTs	7,369 /13.85%
		FFs	63 /0.06%
		Slices	2,159 /16.23%
	DSPs		0 /0.00%
Speed	WNSi [ns]		0.108
	Ti [ns]		31.2
	Max. Freq. [MHz]		32.16
	Throughput [Mbps]		1,029.20
Efficiency [Mbps/Slices]			0.476
Power [W]			0.105

II.3.3 Performance Evaluation of the Logistic map

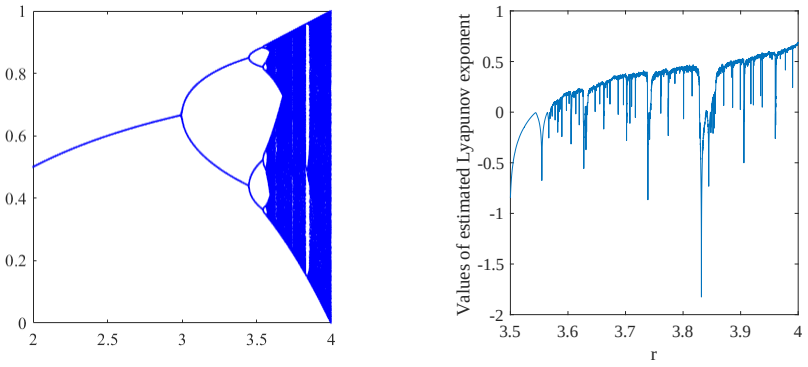
The logistic map was originally a model of population development, published in 1845 by Pierre Verhulst [116]. In 1947, Ulam and Von Neumann used it as a pseudo-random number generator [117] because of the simplicity of its recurrence equation. Since then, it has been one of the most used maps in cryptographic applications. Its recurrence equation is given by:

$$x(n) = f(x(n-1), r) = r \times x(n-1) \times (1 - x(n-1)) \tag{II.14}$$

With $f : S \to S$, $S = [0, 1]$, $x(n) \in S$, $r \in [1, 4]$

In Fig. II.20, we present the known curves of the bifurcation diagram and the

Lyapunov exponent respectively of the logistic map. As expected, the chaos region is obtained for $r \geq 3.57$. In addition, the value $r = 4$ is optimal, because in this case, the amplitude $x(n)$, $n = 1, 2, ...$ covers all the dynamics $\in [0, 1]$ of the map. For this reason, in the discrete version of the map, we set the value of the control parameter to the value corresponding to $r = 4$.



(a) Bifurcation Diagram (b) Lyapunov Exponent

Figure. II.20: Bifurcation Diagram and Lyapunov Exponent of the Logistic map.

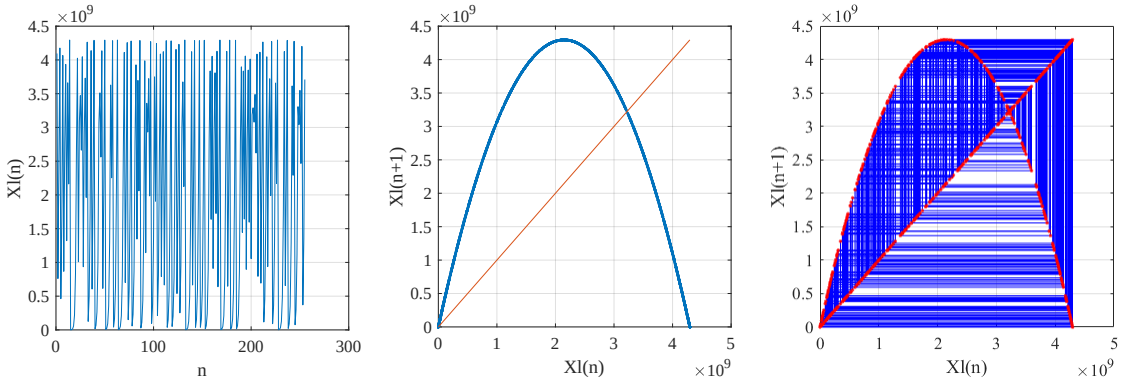
The discrete Logistic map equation for the control parameter set to 4 is given by the following equation [118, 119]:

$$XL(n) = \begin{cases} \left\lfloor \frac{XL(n-1)\left[2^N - XL(n-1)\right]}{2^{N-2}} \right\rfloor & if\ XL(n-1) \neq \left[3 \times 2^{N-2} - 1,\ 2^{N-1}\right] \\ 3 \times 2^{N-2} - 1 & if\ XL(n-1) = 3 \times 2^{N-2} \\ 2^N - 1 & if\ XL(n-1) = 2^{N-1} \end{cases} \quad \text{(II.15)}$$

❖ **Phase space**

Fig. II.21, illustrates the discrete variation of a given sequence $XL(n)$ generated by the discrete Logistic map, the phase space (mapping), and the attractor formed of 3,125,000 samples. The chosen initial condition XL(0) is equal to 198,304,613.

The produced attractor trajectory clearly shows the signature relating to the Logistic map.

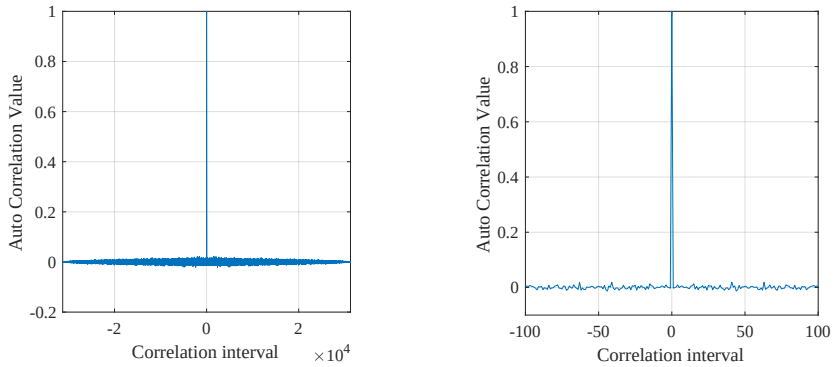


(a) Discrete variation (b) Phase space (c) Attractor

Figure. II.21: Discrete variation, phase space trajectory, and attractor of a given sequence $Xl(n)$ generated by the discrete Logistic map.

❖ Correlation analysis

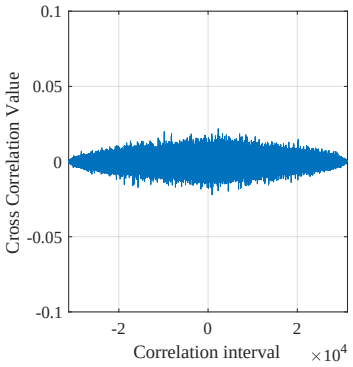
To evaluate the correlation between two sequences Xl and Xl' produced using two Keys (K1, K2) which just have the LSB (Least Significant Bit) bit different, we draw in Fig. II.22 the auto-correlation of sequence Xl and zoom of this auto-correlation on 200 samples. Also, we plot the cross-correlation between the two sequences Xl and Xl', a zoom on it, and a zoom of the auto and cross-correlation in Fig. II.23. Then, we calculate the correlation coefficient $\rho_{Xl,Xl'}$. The obtained value is equal to $-0{,}0027$. Obtained results show good auto and cross-correlation characteristics.



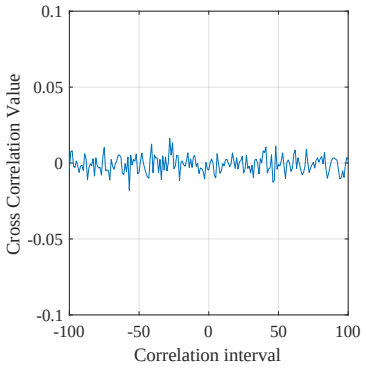
(a) Auto-correlation of sequence Xl

(b) A zoom of the auto-correlation on 200 samples of sequence Xl

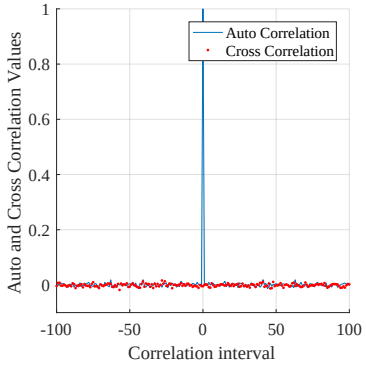
Figure. II.22: Auto-correlation of sequence Xl generated by Logistic map.



(a) Cross-correlation of sequences Xl and Xl'



(b) A zoom of the cross-correlation on 200 samples of sequences Xl and Xl'



(c) A zoom of the cross-correlation of sequences Xl and Xl' and of the auto-correlation of sequence Xl'

Figure. II.23: Cross-correlation functions of sequences Xl and Xl' generated by the Logistic map.

❖ NIST test analysis

In Fig. II.24 we present the obtained proportion versus test for the discrete Logistic map. As we can see, that the sequences do not pass all the NIST tests and that some tests are far from the acceptance threshold of a test materialized by the red line. Notice that the minimum pass rate for each statistical test with the exception of the random excursion (variant) test is approximately = 96 for a sample size = 100 binary sequences. The minimum pass rate for the random excursion (variant) test is approximately = 22 for a sample size = 24 binary sequences. In Table II.5, we give the *P-value* and the proportion for the 15 NIST tests. The obtained results show that the Logistic map does not have good cryptographic statistical properties for all values $[1, 2^N - 1]$ or $]0, 1]$.

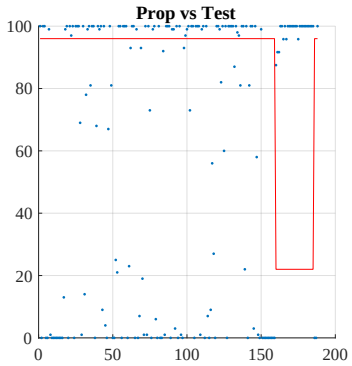


Figure. II.24: NIST test results of the Logistic map.

Table II.5: P-values and Proportion results of NIST test for the Logistic map.

Test	P-value	Proportion (%)
Frequency test	0.000	100.000
Block-frequency test	0.000	0.000
Cumulative-sums test	0.000	100.000
Runs test	0.000	0.000
Longest-run test	0.000	0.000
Rank test	0.401	99.000
FFT test	0.000	1.000
Nonperiodic-templates	0.000	57.182
Overlapping-templates	0.000	0.000
Universal	0.000	0.000
Approximate Entropie	0.000	0.000
Random-excursions	0.052	95.313
Random-excursion-variant	0.493	99.769
Serial test	0.000	0.000
Linear-complexity	0.000	100.000

❖ Histogram analysis

We present in Fig. II.25 the histogram of the generated sequence $Xl(n)$. Visually, the produced sequence $Xl(n)$ is not uniform over all values $[1, 2^N - 1]$.

❖ Chi-square test analysis

We apply the Chi-Square test in order to confirm the non-uniformity of generated sequences. The calculated experimental value χ^2_{exp} for a given produced sequence is equal to 38698.41 which is significantly higher than the theoretical one χ^2_{th}, equal to 1073.64. This asserts the non-uniformity of the sequence.

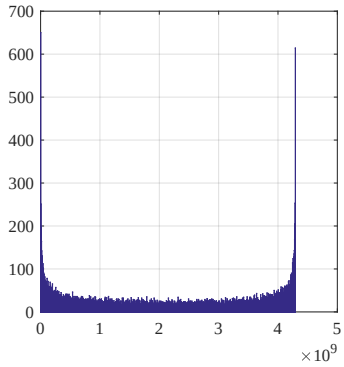


Figure. II.25: Histogram of sequence Xl generated by the discrete Logistic map.

❖ **Hardware metrics**

Table II.6 gives the hardware resources used for the discrete Logistic map. The obtained values show that the Logistic map is characterized by a high throughput compared to the throughput of the other studied chaotic maps and uses fewer hardware resources.

Table II.6: Hardware metrics of the Logistic map.

Resources used	Area	LUTs	77 /0.14%
		FFs	49 /0.05%
		Slices	33 /0.25%
	DSPs		4 /1.82%
Speed	WNSi [ns]		0.102
	Ti [ns]		12
	Max. Freq. [MHz]		84.04
	Throughput [Mbps]		2,689.52
Efficiency [Mbps/Slices]			81.500
Power [W]			0.083

II.3.4 Performance Evaluation of the 3D-Chebyshev map

The discrete 3D-Chebyshev (third order) map:

$$XT(n) = \left\lfloor 2^{(-2N+2)} \times \left(4 \times \left(XT - 2^{(N-1)}\right)^3 - 3 \times 2^{(2N-2)} \times \left(XT - 2^{(N-1)}\right)\right) + 2^{(N-1)} \right\rfloor \quad \text{(II.16)}$$

where $1 \leq XT\,(n) \leq 2^N - 1$.

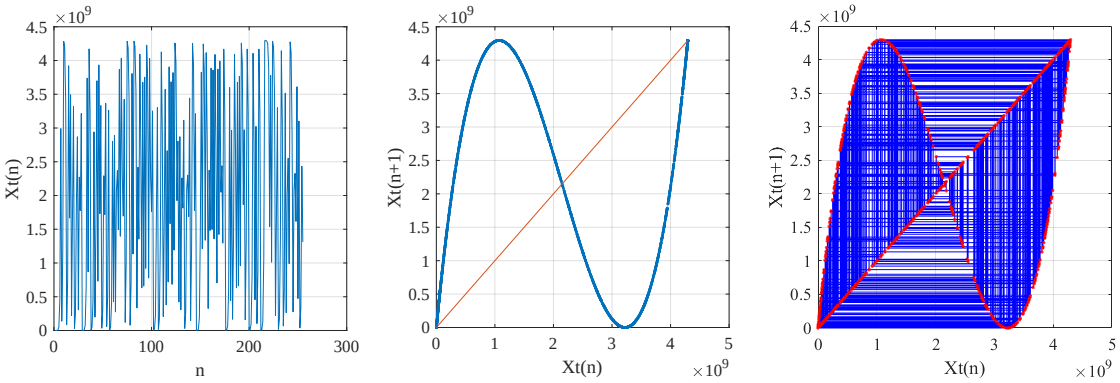
This is the discretized equation of the standrd 3D Chebyshev map [118]:

$$x\,(n) = 4[x\,(n-1)]^3 - 3x\,(n-1) \tag{II.17}$$

with $x\,(n) \in [-1,\ 1]$

❖ Phase space

We plot the discrete variation, mapping, and attractor of the discrete 3D-Chebyshev map in Fig. II.26, with an initial condition $Xt(0) = 875$. Resulted attractor shows the signature of the Chebyshev third order map.

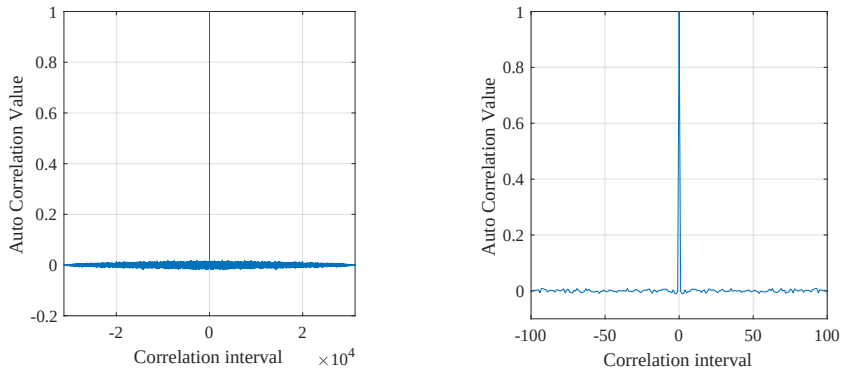


(a) Discrete variation (b) Phase space (c) Attractor

Figure. II.26: Discrete variation, phase space trajectory, and attractor of sequence $Xt(n)$ generated by the discrete 3D-Chebyshev map.

❖ Correlation analysis

The auto-correlation function of sequence Xt and a zoom of the auto-correlation on 200 samples of this sequence are shown in Fig. II.27. The Cross-correlation functions of sequences Xt and Xt' created by the 3D-Chebyshev map, as well as a zooming of the auto and cross-correlation, are shown in Fig. II.28. As a result, the resultant sequences, which are constructed using slightly different seeds, have no correlation. This is quantified by the correlation coefficient $\rho_{Xt,Xt'}$, which is equal to -0.00086.



(a) Auto-correlation of sequence Xt

(b) A zoom of the auto-correlation on 200 samples of sequence Xt

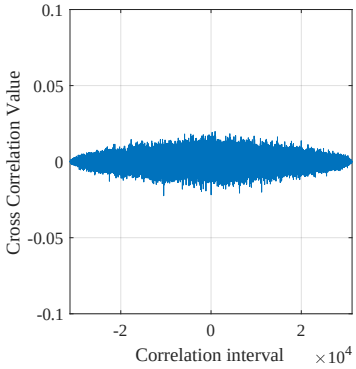
Figure. II.27: Auto-correlation of sequence Xt generated by 3D-Chebyshev map.

❖ NIST test analysis

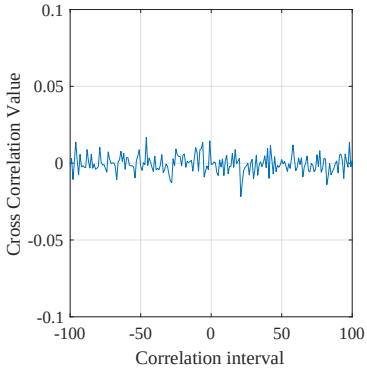
Fig. II.29 and Table II.7 give the results of NIST test applied on a sequence $Xt(n)$. Obtained results show that sequences $Xt(n)$ do not pass all the NIST tests. Also, we remark that the 3D-Chebyshev map has worse security performance than the Skew Tent and the PWLCM map, and approximately it has similar security as the logistic map.

Table II.7: P-values and Proportion results of NIST test for the 3D-Chebyshev map.

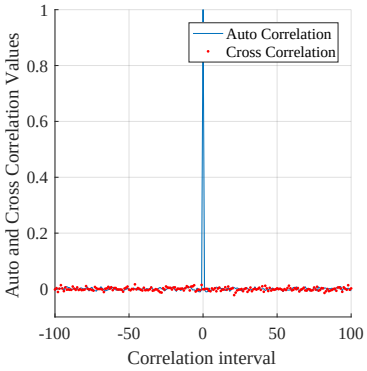
Test	P-value	Proportion (%)
Frequency test	0.000	100.000
Block-frequency test	0.000	0.000
Cumulative-sums test	0.000	100.000
Runs test	0.035	0.000
Longest-run test	0.000	0.000
Rank test	0.115	100.000
FFT test	0.000	0.000
Nonperiodic-templates	0.000	66.507
Overlapping-templates	0.000	0.000
Universal	0.000	0.000
Approximate Entropie	0.000	0.000
Random-excursions	0.004	96.115
Random-excursion-variant	0.248	99.775
Serial test	0.000	0.000
Linear-complexity	0.760	99.000



(a) Cross-correlation of sequences Xt and Xt'



(b) A zoom of the cross-correlation on 200 samples of sequences Xt and Xt'



(c) A zoom of the cross-correlation of sequences Xt and Xt' and of the auto-correlation of sequence Xt'

Figure. II.28: Cross-correlation functions of sequences Xt and Xt' generated by the 3D-Chebyshev map.

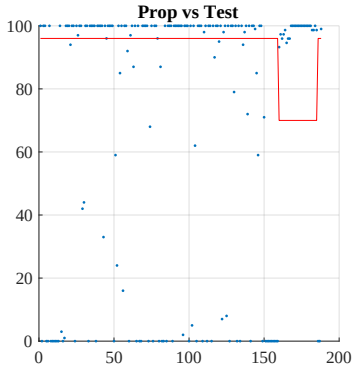


Figure. II.29: NIST test results of the 3D-Chebyshev map.

❖ **Histogram analysis**

In Fig. II.30, we give the histogram of a sequence produced by the 3D-Chebyshev map. As we can see that, the histogram of $Xt(n)$ is not uniform.

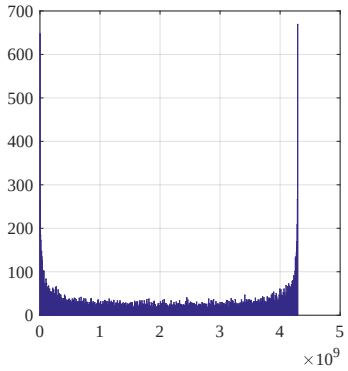


Figure. II.30: Histogram of sequence Xt generated by the discrete 3D-Chebyshev map.

❖ **Chi-square test analysis**

The Chi-square test is used to confirm the sequence's $Xt(n)$ non-uniformity. The experimental value χ^2_{exp} is 41865.00, which is much greater than the theoretical value χ^2_{th}, which is 1073.64. This confirms the sequence's non-uniformity.

❖ **Hardware metrics**

In Table II.8, we present the hardware metrics of the Chebyshev third order map. We note that the 3D-Chebyshev map is slower than the Logistic map and uses fewer hardware resources compared to the Skew Tent and PWLCM map.

II.3.5 Performance Evaluation of a LFSR

Linear-feedback shift register (LFSR) is a shift register whose input bit is a linear function of its previous state. The most commonly used linear function of single bits is exclusive-or (XOR). Thus, an LFSR is most often a shift register whose input bit is driven by the XOR of some bits of the overall shift register value.

The bits in the LFSR state which influence the input are called taps. A maximum-length LFSR produces an m-sequence (i.e. it cycles through all possible $2^n - 1$ state

Table II.8: Hardware metrics of the 3D-Chebyshev map.

Resources used	Area	LUTs	286 /0.54%
		FFs	47 /0.04%
		Slices	87 /0.65%
	DSPs		12 /5.45%
Speed	WNSi [ns]		0.031
	Ti [ns]		22
	Max. Freq. [MHz]		45.51
	Throughput [Mbps]		1,456.59
Efficiency [Mbps/Slices]			16.742
Power [W]			0.053

within the shift register except the state where all bits are zero), unless it contains all zeros, in which case it will never change. The sequence of numbers generated by this method is random. The period of the sequence is $2^n - 1$, where n is the number of shift registers used in the design.

There are two conventional forms of LFSR designs: Fibonacci or external feedback LSFRs and Galois or internal feedback LSFRs.

1. Fibonacci LFSRs: Fig. II.31 shows an n-stage Fibonacci LFSR. It consists of n flip-flops and a number of XOR gates placed on the external feedback path.

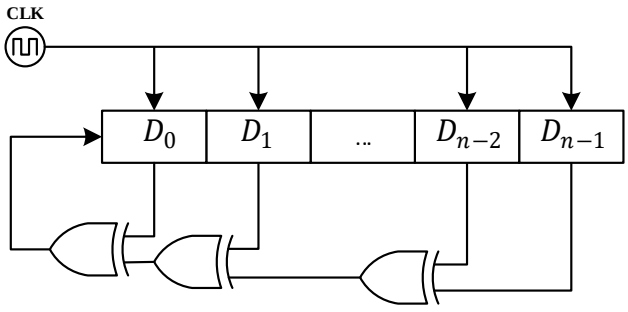


Figure. II.31: An n-stage (external feedback) Fibonacci LFSR.

2. Galois LFSRs: Similarly, an n-stage Galois LFSR with each XOR gate placed between two adjacent flip-flops, as shown in Fig. II.32, is referred to internal

feedback LSFR. Normally, this circuit runs faster than its corresponding Fibonacci LFSR because each stage introduces at most one XOR-gate delay.

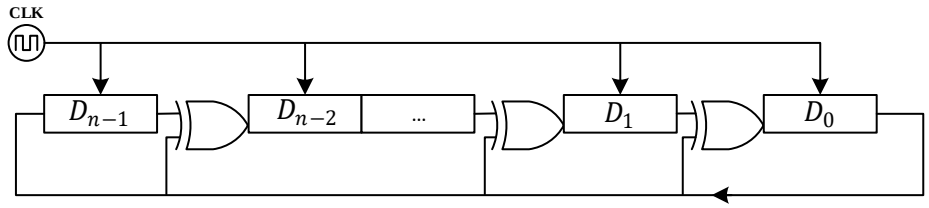


Figure. II.32: An n-stage (internal feedback) Galois LFSR.

The 32-bit LFSR with maximum length feedback used in our work is defined by the following primitive polynomial.

$$Q(n) = x^{32} + x^{22} + x^2 + x + 1 \tag{II.18}$$

The circuit diagram for 32-bit LFSR with maximum length polynomial is shown in Fig. II.33. The sequences generated (random outputs) are periodic of period equal to $2^{32} - 1 = 4,294,967,295$.

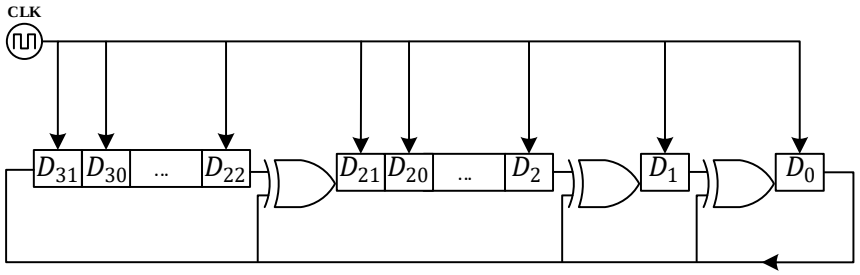
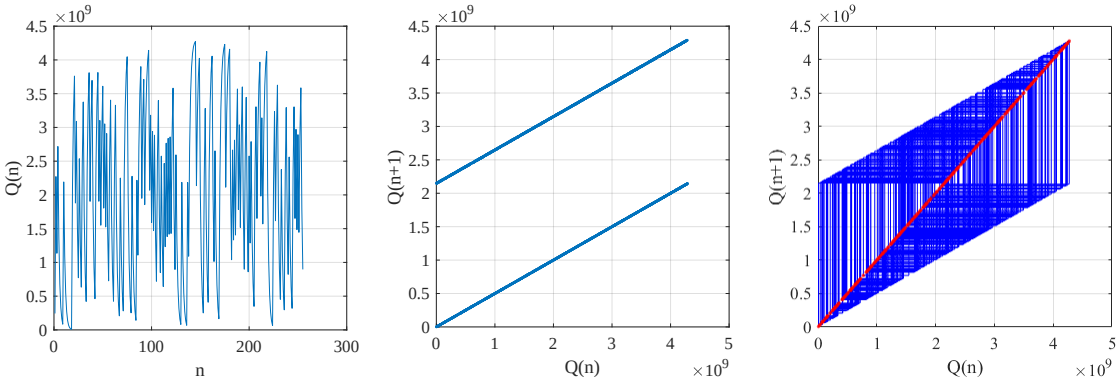


Figure. II.33: Circuit Diagram of 32-bit (internal feedback) LFSR.

❖ Phase space

We give in the Fig. II.34 the the discrete variation, phase space, and attractor of a sequence Q, formed by 3,125,000 samples and generated by the LFSR, with $seed = 00001111001001000110010001110100$ generated randomly.

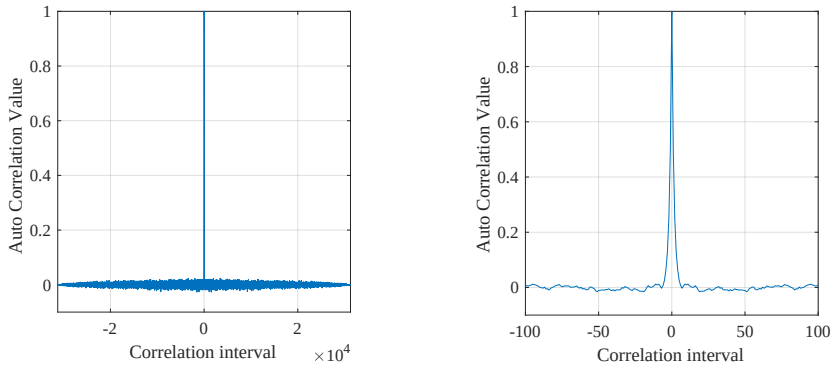


(a) Discrete variation (b) Phase space (c) Attractor

Figure. II.34: Discrete variation, phase space trajectory, and attractor of sequence $Q(n)$ generated by the LFSR.

❖ Correlation analysis

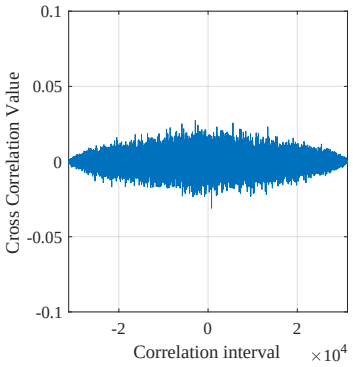
We give in Fig. II.35, the auto-correlation function of sequence Q and a zoom of the auto-correlation on 200 samples of this sequence. Fig. II.36 presents the Cross-correlation functions of sequences Q and Q' generated by the LFSR, and a zoom of the auto and cross-correlation. We note that the cross-correlation function is very low (maximum value = 0.025). Consequently, there is no correlation between the generated sequences, that are produced using slightly different seeds. The correlation coefficient $\rho_{Q,Q'}$ which is equal -0.0110 confirms this result.



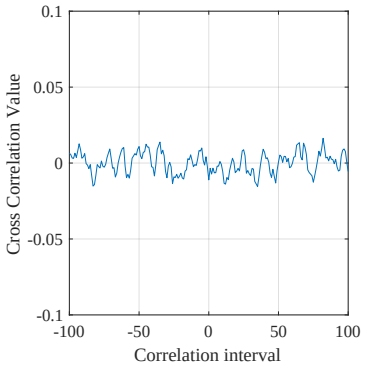
(a) Auto-correlation of sequence Q

(b) A zoom of the auto-correlation on 200 samples of sequence $Q(n)$

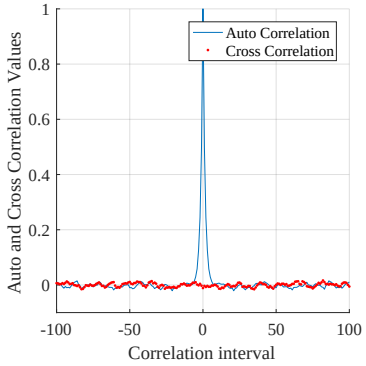
Figure. II.35: Auto-correlation of sequence $Q(n)$ generated by LFSR.



(a) Cross-correlation of sequences Q and Q'



(b) A zoom of the cross-correlation on 200 samples of sequences Q and Q'



(c) A zoom of the cross-correlation of sequences Q and Q' and of the auto-correlation of sequence Q'

Figure. II.36: Cross-correlation functions of sequences Q and Q' generated by the LFSR.

❖ NIST test analysis

In Fig. II.37 and Table II.9, we present the obtained results of the NIST test. Generated sequences are of a size 27.5 Mbits (32-bit sequence with length of 27500101). The pseudo random sequence of the LFSR generator fulfill the following statistical tests:

- Random excursions test
- Random excursion variant test

For both mentioned tests, *P-values* are located above the statistical significance level line ($\alpha = 0,01$) and these values allow to ascertain that the generator passes the two mentioned statistical tests. The *P-values* calculated for the other tests are equal to zero.

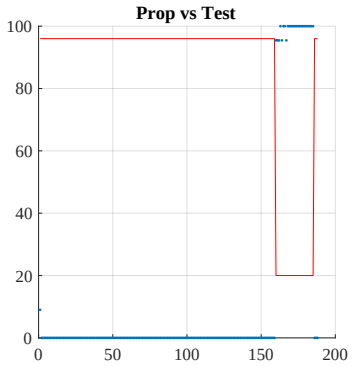


Figure. II.37: NIST test results of the LFSR.

Table II.9: P-values and Proportion results of NIST test for the LFSR.

Test	P-value	Proportion (%)
Frequency test	0.000	9.000
Block-frequency test	0.000	0.000
Cumulative-sums test	0.000	0.000
Runs test	0.000	0.000
Longest-run test	0.000	0.000
Rank test	0.000	0.000
FFT test	0.000	0.000
Nonperiodic-templates	0.000	0.000
Overlapping-templates	0.000	0.000
Universal	0.000	0.000
Approximate Entropie	0.000	0.000
Random-excursions	0.539	97.159
Random-excursion-variant	0.329	100.000
Serial test	0.000	0.000
Linear-complexity	0.000	0.000

❖ Histogram analysis

We plot in Fig. II.38 the histograms of sequence Q generated by the LFSR. The obtained histogram is obviously uniform.

❖ Chi-square test analysis

We calculate the theoretical and experimental values of the Chi-square test to assert the uniformity of the generated sequence. The experimental value obtained is equal to 989.48 which is less than the theoretical one.

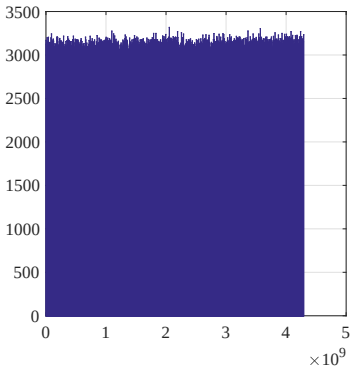


Figure. II.38: Histogram of sequence Q generated by the LFSR.

❖ **Hardware metrics**

Table II.10 summarizes the hardware resources used for the 32-bit LFSR. It uses 62 flip-flops after post implementation.

Table II.10: Hardware metrics of the LFSR.

Resources used	Area	LUTs	2 /< 0.01%
		FFs	62 /0.06 %
		Slices	19 /0.14%
	DSPs		0 /0.00%
Speed	WNSi [ns]		5.965
	Ti [ns]		8
	Max. Freq. [MHz]		491.40
	Throughput [Mbps]		15,724.81
Efficiency [Mbps/Slices]			827.621
Power [W]			0.118

II.3.6 Performance Evaluation of the 3D-Chebyshev map coupled with a LFSR

It is well known that all Chebyshev polynomials are characterized by their short periodicity and the non-uniformity of the histograms of the sequences generated.

To enhance the periodicity of the 3D-Chebyshev and also its uniformity, we couple it with a LFSR as show in Fig. II.39. The LFSR used is defined by Eq. (II.18).

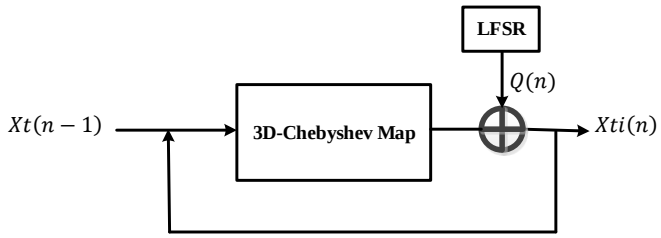
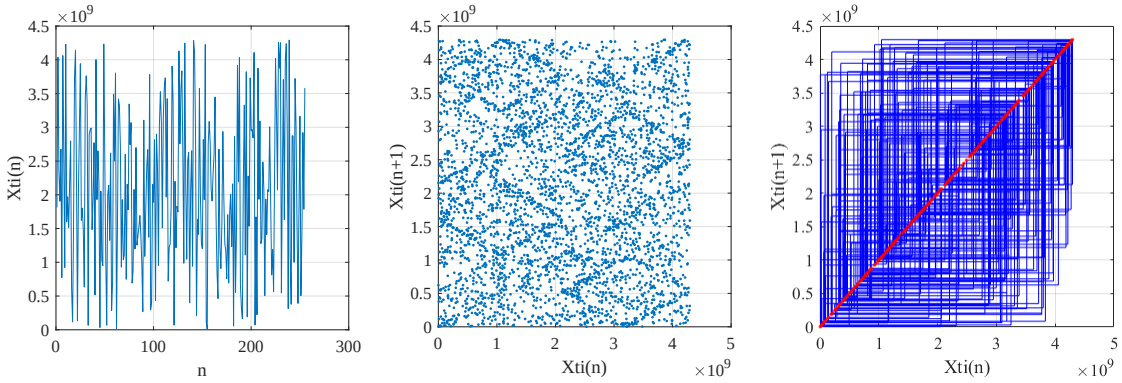


Figure. II.39: 3D-Chebyshev map coupled with the used LFSR using a XOR operator.

❖ Phase space

Fig. II.40 gives the discrete variation, mapping, and attractor of sequence Xti(n) generated by the system of Fig. II.39. The resulting mapping in Fig. II.40b seems to be random compared to the known mapping of a Skew Tent, PWLCM, Logistic, and a Chebyshev third order map. Also, the attractor obtained does not correspond to the known attractors. This is due to the coupling technique.

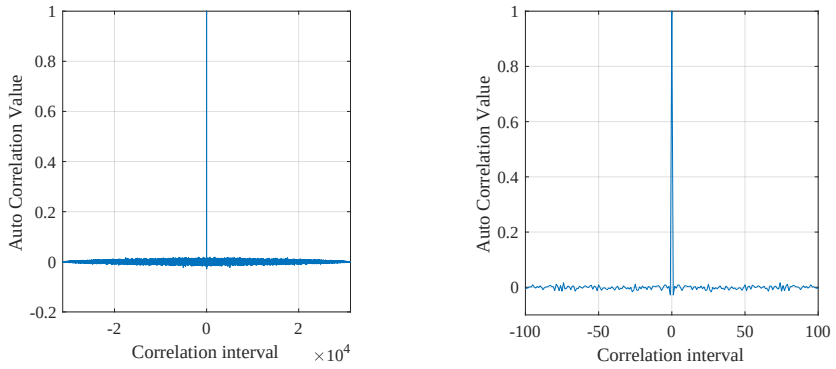


(a) Discrete variation (b) Phase space (c) Attractor

Figure. II.40: Discrete variation, phase space trajectory, and attractor of sequence $Xti(n)$ generated by the discrete 3D-Chebyshev map with LFSR.

❖ Correlation analysis

The 3D-Chebyshev map coupled with the given LFSR has good auto and cross-correlation properties as shown in Fig. II.41 and Fig. II.42 respectively. This result is confirmed by the value of the correlation coefficient of two sequences Xti and Xti', generated with nearby initial values, which is very low ($\rho_{Xti,Xti'} = -0.0042$).



(a) Auto-correlation of sequence Xti

(b) A zoom of the auto-correlation on 200 samples of sequence Xti

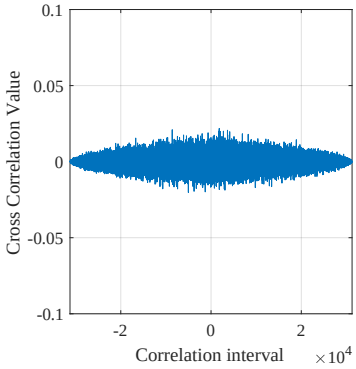
Figure. II.41: Auto-correlation of sequence Xti generated by 3D-Chebyshev map with LFSR.

❖ NIST test analysis

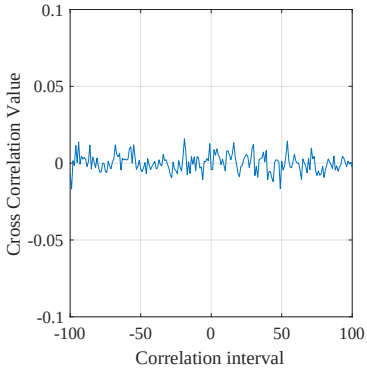
Fig. II.43 and Table II.11 presents the results of the NIST for sequences generated by the distrubed 3D-Chebyshev map with LFSR. Obtained results show that all NIST tests, except the Block-frequency test, have passed demonstrating that the coupled technique with an LFSR improves the cryptographic properties of the 3D-Chebyshev map.

Table II.11: P-values and Proportion results of NIST test for the 3D-Chebyshev map with LFSR.

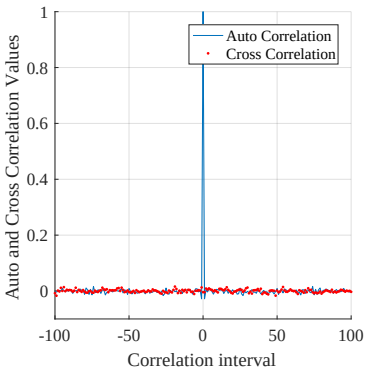
Test	P-value	Proportion (%)
Frequency test	0.249	100.000
Block-frequency test	0.000	98.000
Cumulative-sums test	0.608	100.000
Runs test	0.335	99.000
Longest-run test	0.494	100.000
Rank test	0.983	99.000
FFT test	0.115	100.000
Nonperiodic-templates	0.524	99.020
Overlapping-templates	0.304	100.000
Universal	0.798	98.000
Approximate Entropie	0.213	100.000
Random-excursions	0.284	99.265
Random-excursion-variant	0.290	99.592
Serial test	0.259	99.000
Linear-complexity	0.514	100.000



(a) Cross-correlation of sequences Xti and Xti'



(b) A zoom of the cross-correlation on 200 samples of sequences Xti and Xti'



(c) A zoom of the cross-correlation of sequences Xti and Xti' and of the auto-correlation of sequence Xti'

Figure. II.42: Cross-correlation functions of sequences Xti and Xti' generated by the 3D-Chebyshev map with LFSR.

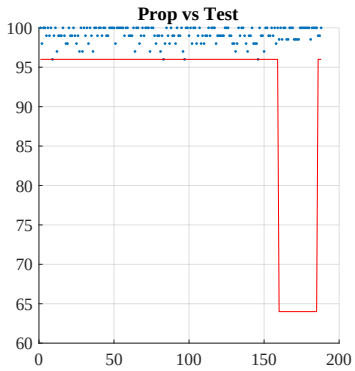


Figure. II.43: NIST test results of the 3D-Chebyshev map with LFSR.

❖ Histogram analysis

We present in Fig. II.44 the histogram for generated sequence by the system of Fig. II.39. We can visually observed that the histogram has uniform distribution.

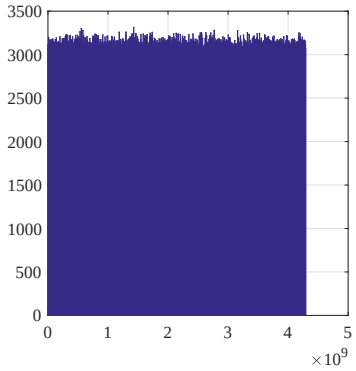


Figure. II.44: Histogram of sequence Xti generated by the discrete 3D-Chebyshev map with LFSR.

❖ Chi-square test analysis

We calculate the theoretical and experimental values of the Chi-square test to assert the uniformity of the generated sequences. The experimental value obtained is equal to 999.48 and it less than the theoretical one. The uniformity is reached when using a coupling technique. Consequently, the 3D-Chebyshev map coupled with LFSR can be used as a pseudo-random number generator of chaotic sequences.

❖ Hardware metrics

Table II.12 summarizes the hardware resources used for the 3D-Chebyshev coupled with 32-bit LFSR in terms of resources used (LUTs, FFs, and DSP blocks), speed, and power consumption. It uses in total 367 Look-Up Tables, 71 Flip Flops numbers, and 12 DSP blocks. Most of the resources are used by the recursive cell with 365 LUTs, 39 FFs, and 12 DSP blocks for the 3D-Chebyshev map and 2 LUT and 32 FFs for the LFSR one. For the computing performance, in the customized architecture, the number of clock cycles, which is required to generate a new sample is one cycle that operates at 46.85 MHz which achieves a throughput of 1.5 Gbps.

Table II.12: Hardware metrics of the 3D-Chebyshev coupled with LFSR map.

Resources used	Area	LUTs	367 /0.69%
		FFs	71 /0.07%
		Slices	115 /0.86%
	DSPs		12 /5.45%
Speed	WNSi [ns]		0.006
	Ti [ns]		21.35
	Max. Freq. [Mhz]		46.85
	Throughput [Mbps]		1,499.25
Efficiency [Mbps/Slices]			13.036
Power [W]			0.056

II.4 Conclusion

In this chapter, we evaluate the statistical security and hardware metrics of some chaotic maps, specifically the Skew Tent map, PWLCM, Logistic, 3D-Chebyshev, an LSFR, and the 3D-Chebyshev coupled with LFSR, which are the basic components of the proposed chaotic generators presented in the following chapters. We first presented the common and standard security tools for measuring the statistical security performance of the studied chaotic maps. Then, we analyze the hardware metrics performance of the implementation of these chaotic maps in terms of resources used, speed, and power consumption.

We note that the Logistic map is the fastest and uses fewer hardware resources than other maps but it has the weakest cryptographic properties compared to the Skew Tent, PWLCM, and Chebyshev third-order maps. The PWLCM map is characterized by its good cryptographic properties which are better than those of the other studied chaotic maps but it is the slowest one and uses more hardware resources.

Also, we have shown that, xoring the output of a chaotic map with an LFSR improves the cryptographic properties of the coupled system.

Chapter III

Design, FPGA-Based Implementation and Analysis of Stream Ciphers Based on Secure PRNGs of Chaotic Sequences

“ Fully secure systems don’t exist today and they won’t exist in the future. ”

Adi Shamir

Summary

III.1 Introduction

A common and important element in chaos-based cryptography, especially in a stream cipher, is the pseudo random number generator of chaotic sequences (PRNG-CS). A variety of PRNGs-CS with software implementation has been published in the literature [60,61,80,81,88]. However, due to the rapid growth of Internet of Things (IoT) and some hardware based applications, it is necessary to have such primitive in hardware implementation such as an FPGA. Based on the statistical results obtained in the previous Chapter II, we can confirm that chaotic maps cannot be used alone as secure PRNG-CS. The question is how to combine some of them to construct secure, against statistical and cryptanalytic attacks, and effective, in terms of hardware cost, throughput and efficiency PRNGs-CS. A trade-off between security and effectivity is normally made and this is first of all in relation to the degree of security required for a given application.

In this chapter, we designed, implemented and analyzed four Secure PRNGs-CS, namely: LSP-PRNG (combining the discrete Logistic, Skew Tent and PWLCM map), LST-PRNG (combining the discrete Logistic, Skew Tent and 3D Chebyshev map), LSPT-PRNG (combining the discrete Logistic, Skew Tent, PWLCM and 3D Chebyshev map), LST_RC-PRNG (combining the discrete Logistic, Skew Tent, and 3D Chebyshev map, each in a recursive cell) and their corresponding stream cipher systems.

All proposed PRNGs-CS contain at least one polynomial map of degree 2 (Logistic map) or degree three (3D Chebyshev map) to make very difficult the algebraic attack.

Likewise, each of these chaotic systems uses a predefined coupling matrix M which achieves a weak mixing of the chaotic maps involved in a given PRNG-SC, which avoid, on the one hand, the divide-and-conquer attacks on chaotic maps, and on the other hand, to increase the randomness of the sequences produced as well as their lengths.

The hardware implementation is achieved on a Xilinx XC7Z020 PYNQ-Z2 FPGA platform, using Very High-Speed Integrated Circuit Hardware Description Language (VHDL). The simulation, synthesis and implementation are performed using the Xilink Vivado design Suite environment (V.2017.2).

III.2 Block diagram of stream ciphers based on Secure PRNGs-CS

The block diagram of a stream encryption/decryption system is presented in Fig. III.1. As we can see, the stream encryption/decryption algorithm comes down to an XOR operation between the plaintext and the keystream for encryption; the ciphertext, and the keystream for decryption. The security of such a system depends entirely on the keystream delivered by the keystream generator. If the keystream is perfectly random and the period tends to infinity, then the encryption/decryption system becomes unconditionally secure (called a one-time pad). The keystream generator takes as input a secret key and an initial value (IV) used to overcome known plain text attack. The IV is changed with each new session and must be used only once. Thus, the sequences generated in the different sessions with the same secret key are different. Recall that, stream ciphers are used to encrypt data (bits or samples) continuously such as network communications or selective video encryption. In section III.5, we analyze in detail the performances in terms of hardware metrics and in terms of security obtained from the stream ciphers based on the proposed PRNGs-CS.

In the following, we will describe in detail the proposed secure PRNGs-CS as a secure keystream generator. Recall that the discrete equations of all the chaotic maps involved in this chapter have been defined in chapter II.

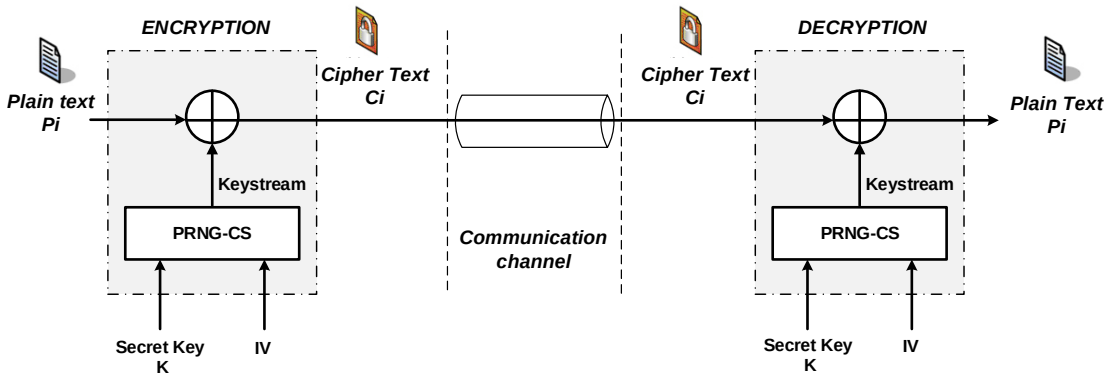


Figure. III.1: Block diagram of a stream encryption/decryption system.

III.3 Architecture description of the proposed PRNGs-CS

All the proposed secure PRNGs-CS have the same general structure but differ in their internal state and slightly change in their output function. In this section we describe in details the architecture of proposed secure PRNGs-CS and then in section IV.4, we evaluate their hardware and security analysis, before concluding.

III.3.1 Architecture and detailed description of the proposed LSP-PRNG

The architecture of the proposed LSP-PRNG is given in Fig. III.2. Its internal state is formed by three discrete chaotic maps weakly coupled, namely, the Logistic map, the Skew Tent, and the Piecewise Linear Chaotic Map (PWLCM), and the output function is based on a chaotic multiplexing technique.

The proposed LSP-PRNG takes the secret key K and the initial vector IV as input and calculates the initial values $XL(0)$, $XS(0)$ and $XP(0)$ of the three chaotic maps: Logistic, Skew Tent and PWLCM respectively. The IV of the system supplies the initial vectors of the three chaotic maps: IVL, IVS, and IVP each is of size N=32 bits, and the secret key K provides all the initial conditions and parameters of the chaotic maps and the parameters of the coupling matrix M, as summarized in Table III.1.

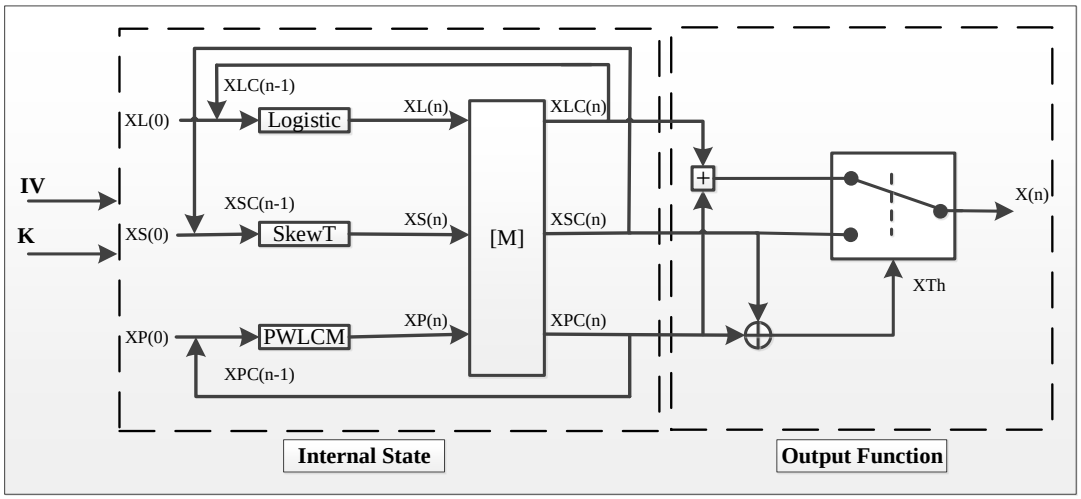


Figure. III.2: Architecture of the proposed LSP-PRNG.

Table III.1: The initial conditions and parameters that form the secret key.

Symbol	Definition
$XL0$, $XS0$, and $XP0$	Initial conditions of the chaotic maps: Logistic, Skew Tent, and PWLCM respectively, ranging from 1 to $2^N - 1$.
P_s	Control parameter of the Skew Tent map, in the range $[1,2^N - 1]$.
P_p	Control parameter of the PWLCM map, in the range $[1,2^{N-1} - 1]$.
ϵ_{ij}	Parameters of the coupling matrix M, in the interval $[1,2^k]$ with $k \leq 5$.

The size of the secret key, noted $|K1|$, of the proposed LSP-PRNG is given by

$$|K1| = |XL| + |XS| + |XP| + |Ps| + |Pp| + (6 \times |\epsilon_{ij}|) = 189\ bits \tag{III.1}$$

(see below the form of the Matrix M containing ϵ_{ij})

Where $|XL| = |XS| = |XP| = |P_s| = 32\ bits$, $|P_p| = 31\ bits$, and $|\epsilon_{ij}| = 5\ bits$.

The key space of the secret key is 2^{189} different combinations which are large enough to make the brute-force attack infeasible [120]. Indeed, it is generally accepted that a key space of size equal or greater to 2^{128} makes the brute-force attack infeasible.

The initial conditions and parameters are supplied the secret key K as follows:

$$\begin{cases} XL = K(0\ to\ 31) \\ XS = K(32\ to\ 63) \\ XP = K(64\ to\ 95) \\ Ps = K(96\ to\ 127) \\ Pp = K(128\ to\ 158) \\ \varepsilon_{12} = K(159\ to\ 163) \\ \varepsilon_{13} = K(164\ to\ 168) \\ \varepsilon_{21} = K(169\ to\ 173) \\ \varepsilon_{23} = K(174\ to\ 178) \\ \varepsilon_{31} = K(179\ to\ 183) \\ \varepsilon_{32} = K(184\ to\ 188) \end{cases} \tag{III.2}$$

The initial values $XL(0)$, $XS(0)$, and $XP(0)$of the three chaotic maps are calculated as follows:

$$\begin{cases} XL\,(0) = \mathrm{mod}\ (\,(XL0 + IVin)\,,\ 2^N) \\ XS\,(0) = \mathrm{mod}\ (\,(XS0 + IVin)\,,\ 2^N) \\ XP\,(0) = \mathrm{mod}\ (\,(XP0 + IVin)\,,\ 2^N) \end{cases} \tag{III.3}$$

Where:

$$IVin = IVL \oplus IVS \oplus IVP \tag{III.4}$$

And

$$\begin{cases} IVL = IV(0\ to\ 31) \\ IVS = IV(32\ to\ 63) \\ IVP = IV(64\ to\ 95) \end{cases} \tag{III.5}$$

The internal state function achieves the weak coupling of the chaotic maps and produces the future samples $XLC(n)$, $XSC(n)$ and $XPC(n)$ from which the output function, by using a chaotic switching technique, produces the output sequence $X(n)$ (see Fig. III.2).

The coupling system is governed by the following equation:

$$\begin{bmatrix} XLC(n) \\ XSC(n) \\ XPC(n) \end{bmatrix} = M \times \begin{bmatrix} XL(n) \\ XS(n) \\ XP(n) \end{bmatrix} \tag{III.6}$$

Where:

$$\begin{aligned} XL(n) &= F_l[XL(n-1)] \\ XS(n) &= F_s[XS(n-1)] \\ XP(n) &= F_p[XP(n-1)] \end{aligned} \tag{III.7}$$

And M is the weak coupling matrix, given by:

$$M = \begin{bmatrix} \epsilon_{11} & \epsilon_{12} & \epsilon_{13} \\ \epsilon_{21} & \epsilon_{22} & \epsilon_{23} \\ \epsilon_{31} & \epsilon_{32} & \epsilon_{33} \end{bmatrix} \tag{III.8}$$

with ϵ_{ij} are the weakly coupling parameters, and $F_l[XL(n-1)]$, $F_s[XS(n-1)]$, and $F_p[XP(n-1)]$ are the discrete functions of the chaotic maps: Logistic, Skew Tent and PWLCM defined in Chapter II.

$XL(n)$, $XS(n)$, and $XP(n)$ are denoted as the maps output values at instant n of the: Logistic, Skew Tent, and PWLCM map respectively, defined as follows:

The first three outputs of the matrix M, $XLC(1)$, $XSC(1)$, $XPC(1)$ are given by the following equation:

$$\begin{bmatrix} XLC(1) \\ XSC(1) \\ XPC(1) \end{bmatrix} = M \times \begin{bmatrix} XL(1) \\ XS(1) \\ XP(1) \end{bmatrix} \tag{III.9}$$

With:

$$XL(1) = Logistic\left\{\mathrm{mod}\left(XL(0), 2^N\right)\right\} \tag{III.10}$$

$$XS(1) = SkewT\left\{\mathrm{mod}\left(XS(0), 2^N\right), P_s\right\} \tag{III.11}$$

$$XP(1) = PWLCM\left\{\mathrm{mod}\left(XP(0), 2^N\right), P_p\right\} \tag{III.12}$$

Then, for $n \geq 2$ and $n \leq N_s$, (N_s is the number of the desired samples), the three outputs of the matrix M are governed by equation (III.6), with:

$$XL(n) = Logistic\left\{\mathrm{mod}\left(XLC(n-1), 2^N\right)\right\} \tag{III.13}$$

$$XS(n) = SkewT\left\{\mathrm{mod}\left(XSC(n-1), 2^N\right), \quad P_s\right\} \tag{III.14}$$

$$XP(n) = PWLCM\left\{\mathrm{mod}\left(XPC(n-1), 2^N\right), \quad P_p\right\} \tag{III.15}$$

The obtained multiplexed samples of the sequence $X(n)$ are controlled by the chaotic sample $X_{th}(n)$ and a threshold T:

$$X(n) = \begin{cases} (XPC(n) + XLC(n)) \ mod\ 2^N & if\ 0 < X_{th}(n) < T \\ XSC(n) & otherwise \end{cases} \tag{III.16}$$

Where $X_{th}(n) = XPC(n) \oplus XSC(n)$ and $T = 0.8 \times 2^N$.

The following Algorithm III.1, shows the generation of the pseudo-random sequence $X(n)$.

III.3.2 Architecture of the proposed LST-PRNG

The architecture of the proposed LST-PRNG is given in Fig. III.3. The internal state is formed by three discrete chaotic maps weakly coupled, namely, the Logistic map, the Skew Tent, and the 3D Chebyshev map (3D Ch) combined with a linear feedback shift register (LFSR). The later, as demonstrated in chapter II, allows to enhance the periodicity of the 3D-Chebyshev and also its uniformity. The output function is based on an addition modulo and XOR operation.

Compared to the architecture of Fig. III.2, the proposed LST-PRNG architecture has

Algorithm III.1 $LSP-PRNG$: Generation of the pseudo-random sequence $X(n)$.

Input : IV = Initial Vector of $LSP-PRNG$;
K = Secret key of the $LSP-PRNG$;
Output: Keystream $X(n)$

begin
  initialization
  $XL(0) = mod((XL0 + IVin), 2^N)$;
  $XS(0) = mod((XS0 + IVin), 2^N)$;
  $XP(0) = mod((XP0 + IVin), 2^N)$;
  $T = 0.8 \times 2^N$
  Samples generation
  $\epsilon_{11} = (2^N - \epsilon_{12} - \epsilon_{13})$;
  $\epsilon_{22} = (2^N - \epsilon_{21} - \epsilon_{23})$;
  $\epsilon_{33} = (2^N - \epsilon_{31} - \epsilon_{32})$;
  Calculation of the first output
  $XL(1) = Logistic\left\{\text{mod}\ \left(XL(0), 2^N\right)\right\}$
  $XS(1) = SkewT\left\{\text{mod}\ \left(XS(0), 2^N\right),\ P_s\right\}$
  $XP(1) = PWLCM\left\{\text{mod}\ \left(XP(0), 2^N\right),\ P_p\right\}$
  $XLC(1) = (\epsilon_{11} \times XL(1)) + (\epsilon_{12} \times XS(1))) + (\epsilon_{13} \times XP(1))$;
  $XSC(1) = (\epsilon_{21} \times XL(1)) + (\epsilon_{22} \times XS(1)) + (\epsilon_{23} \times XP(1))$;
  $XPC(1) = (\epsilon_{31} \times XL(1)) + (\epsilon_{32} \times XS(1)) + (\epsilon_{33} \times XP(1))$;
  LSP-PRNG's first output
  $X_{th} = XPC(1) \oplus XSC(1)$;
  if $X_{th} < T$ **then**
    $X(1) = (XPC(1) + XLC(1)) mod 2^N$
  else
    $X(1) = XSC(1)$
  end
  while $n \geq 2$ *and* $n \leq Ns$ **do**
    Internal State
    $XL(n) = Logistic\left\{\text{mod}\left(XLC(n-1),\ 2^N\right)\right\}$
    $XS(n) = SkewT\left\{\text{mod}\left(XSC(n-1),\ 2^N\right),\ P_s\right\}$
    $XP(n) = PWLCM\left\{\text{mod}\left(XPC(n-1),\ 2^N\right),\ P_p\right\}$
    $XLC(n) = (\epsilon_{11} \times XL(n)) + (\epsilon_{12} \times XS(n))) + (\epsilon_{13} \times XP(n))$;
    $XSC(n) = (\epsilon_{21} \times XL(n)) + (\epsilon_{22} \times XS(n)) + (\epsilon_{23} \times XP(n))$;
    $XPC(n) = (\epsilon_{31} \times XL(n)) + (\epsilon_{32} \times XS(n)) + (\epsilon_{33} \times XP(n))$;
    LSP-PRNG's output
    $X_{th} = XPC(n) \oplus XSC(n)$;
    if $X_{th} < T$ **then**
      $X(n) = (XPC(n) + XLC(n)) mod 2^N$
    else
      $X(n) = XSC(n)$
    end
  end
end

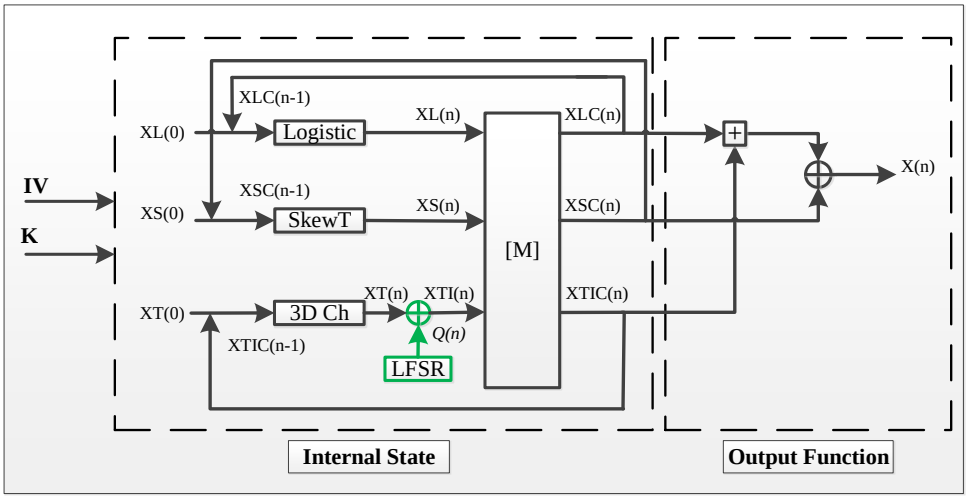


Figure. III.3: Architecture of the proposed LST-PRNG.

better hardware metrics, because the hardware metrics of the 3D Ch map are better than those of the PWLCM map. In addition, the use of the 3D Ch enhance the robustness of the system against algebraic attacks.

The size of the secret key of the proposed LST-PRNG is:

$$|K2| = |XL0| + |XS0| + |XT0| + |P_s| + (6 \times |\epsilon_{ij}|) + |Q0| = 190 \;\; bits \tag{III.17}$$

where $|XT0| = |Q0| = 32 \;\; bits$; and $XT0$, $Q0$ are the initial conditions of the 3D Ch and of the Linear Feedback Shift Register (LFSR) respectively.

The key space contains 2^{190} different values, which is large enough to make brute-force attacks infeasible.

The initial values $XL(0)$, $XS(0)$ and $XT(0)$ of the used chaotic maps are given by:

$$\begin{cases} XL\,(0) = \text{mod}\;((XL0 + IVin)\,,\; 2^N) \\ XS\,(0) = \text{mod}\;((XS0 + IVin)\,,\; 2^N) \\ XT\,(0) = \text{mod}\;((XT0 + IVin)\,,\; 2^N) \end{cases} \tag{III.18}$$

Where:

$$IVin \;=\; IVL \;\oplus\; IVS \;\oplus\; IVT \tag{III.19}$$

and IVL, IVS and IVT are provided by the initial value IV as follows:

$$\begin{cases} IVL{=}IV\,(0 \;\; to \;\; 31) \\ IVS{=}IV\,(32 \;\; to \;\; 63) \\ IVT{=}IV\,(64 \;\; to \;\; 95) \end{cases} \tag{III.20}$$

The output samples of the coupling matrix M (defined in (III.8)) $XLC(n)$, $XSC(n)$ and $XTIC(n)$ are used to produce the output sequence $X(n)$ as showed in Fig. III.3. The internal system is governed by the following equation:

$$\begin{bmatrix} XLC(n) \\ XSC(n) \\ XTIC(n) \end{bmatrix} = M \times \begin{bmatrix} XL(n) \\ XS(n) \\ XTI(n) \end{bmatrix} \tag{III.21}$$

With:

$$XTI\,(n) = XT\,(n) \oplus Q\,(n) \tag{III.22}$$

and $XL(n)$, $XS(n)$, and $XT(n)$ are denoted as the maps output values at instant n of the: Logistic, Skew Tent, and 3D Ch map respectively.

The first three outputs of the matrix M, $XLC\,(1)$, $XSC\,(1)$, $XTIC\,(1)$ are given by:

$$\begin{bmatrix} XLC(1) \\ XSC(1) \\ XTIC(1) \end{bmatrix} = M \times \begin{bmatrix} XL(1) \\ XS(1) \\ XTI(1) \end{bmatrix} \tag{III.23}$$

With:

$$XTI\,(1) = XT\,(1) \oplus Q\,(1) \tag{III.24}$$

And:

$$XL(1) = Logistic\left\{\mathrm{mod} \quad \left(XL(0), 2^N\right)\right\} \tag{III.25}$$

$$XS(1) = SkewT\left\{\mathrm{mod} \quad \left(XS(0), 2^N\right), P_s\right\} \tag{III.26}$$

$$XT(1) = 3D \;\; Ch\left\{\mathrm{mod} \quad \left(XT(0), 2^N\right)\right\} \tag{III.27}$$

Then, for $n \geq 2$ and $n \leq N_s$, the outputs of the matrix M, $XLC(n)$, $XSC(n)$ and $XTIC(n)$ are given by equation (III.21), with:

$$XL(n) = Logistic\left\{\mathrm{mod}\left(XLC(n-1),\ 2^N\right)\right\} \tag{III.28}$$

$$XS(n) = SkewT\left\{\mathrm{mod}\left(XSC(n-1),\ 2^N\right),\ P_s\right\} \tag{III.29}$$

$$\begin{cases} XT\,(n) = 3D\ Ch\left\{\mathrm{mod}\left(XTIC\,(n-1)\,,\ 2^N\right)\right\} \\ XTI\,(n) = XT\,(n) \oplus Q\,(n) \end{cases} \tag{III.30}$$

Finally the output $X(n)$ is calculated by (see Fig. III.3):

$$X(n) = \mathrm{mod}\ \left((XLC(n) + XTIC(n))\,,\ 2^N\right)\ \oplus\ XSC(n) \tag{III.31}$$

The generation of the pseudo-random sequence $X\,(n)$ is detailed in Algorithm III.2.

III.3.3 Architecture of the proposed LSPT-PRNG

In some applications (military, industrial, etc.) it is necessary to have a maximum security to the detriment of hardware metrics. To this end, we propose the LSPT-PRNG architecture shown in Fig. III.4. The internal state of the proposed LSPT-PRNG additionally includes the PWLCM map compared to the internal state of the LST-PRNG, and its output function is a little different compared to the previous ones shown in architectures of Figs. III.2 and III.3.

The size of the secret key, noted $|K3|$, is:

$$|K3| = |XP0| + |XS0| + |XL0| + |XT0| + |Q0| + |P_p| + |P_s| + (9 \times |\varepsilon_{ij}|) = 268\ \ bits \tag{III.32}$$

As a result, the key space has 2^{268} different values, which is big enough to keep brute-force attacks ineffective.

The initial values $XP(0)$, $XS(0)$, $XL(0)$, and $XT(0)$ of the four chaotic maps are

Algorithm III.2 $LST-PRNG$: Generation of the pseudo-random sequence $X(n)$.

Input : IV = Initial Vector of $LST-PRNG$;
K = Secret key of the $LST-PRNG$;
Output: Keystream X(n)

begin
  initialization
  $XL(0) = mod((XL0 + IVin), 2^N)$;
  $XS(0) = mod((XS0 + IVin), 2^N)$;
  $XT(0) = mod((XT0 + IVin), 2^N)$;
  Samples generation
  $\epsilon_{11} = (2^N - \epsilon_{12} - \epsilon_{13})$;
  $\epsilon_{22} = (2^N - \epsilon_{21} - \epsilon_{23})$;
  $\epsilon_{33} = (2^N - \epsilon_{31} - \epsilon_{32})$;
  Calculation of the first output
  $XL(1) = Logistic\left\{\text{mod}\ \left(XL(0), 2^N\right)\right\}$
  $XS(1) = SkewT\left\{\text{mod}\ \left(XS(0), 2^N\right),\ P_s\right\}$
  $XT(1) = 3DCh\left\{\text{mod}\ \left(XT(0), 2^N\right)\right\}$
  $XTI(1) = XT(1) \oplus Q(1)$
  $XLC(1) = (\epsilon_{11} \times XL(1)) + (\epsilon_{12} \times XS(1))) + (\epsilon_{13} \times XTI(1))$;
  $XSC(1) = (\epsilon_{21} \times XL(1)) + (\epsilon_{22} \times XS(1)) + (\epsilon_{23} \times XTI(1))$;
  $XTIC(1) = (\epsilon_{31} \times XL(1)) + (\epsilon_{32} \times XS(1)) + (\epsilon_{33} \times XTI(1))$;
  LST-PRNG's first output
  $X(1) = mod((XLC(1) + XTIC(1)), 2^N) \oplus XSC(1)$
  while $n \geq 2$ *and* $n \leq Ns$ **do**
    Internal State
    $XL(n) = Logistic\left\{\text{mod}\left(XLC(n-1),\ 2^N\right)\right\}$
    $XS(n) = SkewT\left\{\text{mod}\left(XSC(n-1),\ 2^N\right),\ P_s\right\}$
    $XT(n) = 3DCh\left\{\text{mod}\left(XTIC(n-1),\ 2^N\right)\right\}$
    $XTI(n) = XT(n) \oplus Q(n)$
    $XLC(n) = (\epsilon_{11} \times XL(n)) + (\epsilon_{12} \times XS(n))) + (\epsilon_{13} \times XTI(n))$;
    $XSC(n) = (\epsilon_{21} \times XL(n)) + (\epsilon_{22} \times XS(n)) + (\epsilon_{23} \times XTI(n))$;
    $XTIC(n) = (\epsilon_{31} \times XL(n)) + (\epsilon_{32} \times XS(n)) + (\epsilon_{33} \times XTI(n))$;
    LST-PRNG's output
    $X(n) = mod((XLC(n) + XTIC(n)), 2^N) \oplus XSC(n)$
  end
end

given by:

$$\begin{cases} XP(0) = \text{mod}\ ((XP0 + IVin),\ 2^N) \\ XS(0) = \text{mod}\ ((XS0 + IVin),\ 2^N) \\ XL(0) = \text{mod}\ ((XL0 + IVin),\ 2^N) \\ XT(0) = \text{mod}\ ((XT0 + IVin),\ 2^N) \end{cases} \tag{III.33}$$

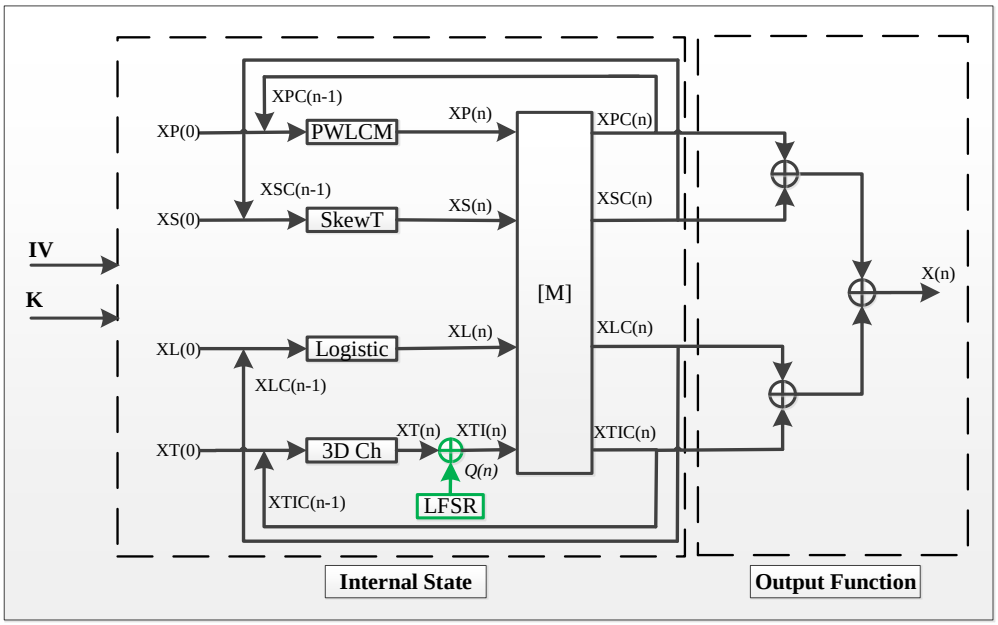


Figure. III.4: Architecture of the proposed LSPT-PRNG.

Where:

$$IVin = IVP \oplus IVS \oplus IVL \oplus IVT \tag{III.34}$$

And IVP, IVS, IVL and IVT are provided by the initial value IVas folows:

$$\begin{cases} IVP{=}IV\,(0 \;\; to \;\; 31) \\ IVS{=}IV\,(32 \;\; to \;\; 63) \\ IVL{=}IV\,(64 \;\; to \;\; 95) \\ IVT{=}IV\,(96 \;\; to \;\; 127) \end{cases} \tag{III.35}$$

The output X(n) is calculated by:

$$X(n) = XPC(n) \oplus XSC(n) \oplus XLC(n) \oplus XTIC(n) \tag{III.36}$$

The coupling system is defined by the following relation:

$$\begin{bmatrix} XPC\,(n) \\ XSC\,(n) \\ XLC\,(n) \\ XTIC\,(n) \end{bmatrix} = M \times \begin{bmatrix} XP\,(n) \\ XS\,(n) \\ XL\,(n) \\ XTI\,(n) \end{bmatrix} \tag{III.37}$$

where:

$$M = \begin{bmatrix} M_{11} & \epsilon_{12} & \epsilon_{13} & \epsilon_{14} \\ \epsilon_{21} & M_{22} & \epsilon_{23} & \epsilon_{24} \\ \epsilon_{32} & \epsilon_{32} & M_{33} & \epsilon_{34} \\ \epsilon_{41} & \epsilon_{42} & \epsilon_{43} & M_{44} \end{bmatrix} \tag{III.38}$$

with M_{11}= $(2^N - \epsilon_{12} - \epsilon_{13} - \epsilon_{14})$, M_{22}= $(2^N - \epsilon_{21} - \epsilon_{23} - \epsilon_{24})$, M_{33}= $(2^N - \epsilon_{31} - \epsilon_{32} - \epsilon_{34})$, and M_{44}= $(2^N - \epsilon_{41} - \epsilon_{42} - \epsilon_{43})$.

$XP(n)$, $XS(n)$, $XL(n)$, and $XT(n)$ are denoted as the maps output values at instant n of the: PWLCM, Skew Tent, Logistic, and 3D Chebyshev map respectively, defined as follows:

The first sample is given by:

$$XP(1) = PWLCM\left\{\text{mod}\quad \left(XP(0), 2^N\right),\quad P_p\right\} \tag{III.39}$$

$$XS(1) = SkewT\left\{\text{mod}\quad \left(XS(0), 2^N\right),\quad P_s\right\} \tag{III.40}$$

$$XL(1) = Logistic\left\{\text{mod}\quad \left(XL(0), 2^N\right)\right\} \tag{III.41}$$

$$XT(1) = 3D\ \ Ch\left\{\text{mod}\quad \left(XT(0), 2^N\right)\right\} \tag{III.42}$$

Then, for $n \geq 2$ and $n \leq N_s$, the samples are calculated by (N_s is the number of the desired samples):

$$XP(n) = PWLCM\left\{\text{mod}\left(XPC(n-1),\ 2^N\right),\quad P_p\right\} \tag{III.43}$$

$$XS(n) = SkewT\left\{\text{mod}\left(XSC(n-1),\ 2^N\right),\quad P_s\right\} \tag{III.44}$$

$$XL(n) = Logistic\left\{\mathrm{mod}\left(XLC(n-1),\ 2^N\right)\right\} \tag{III.45}$$

$$\begin{cases} XT(n) = 3D\ Ch\left\{\mathrm{mod}\left(XTIC(n-1),\ 2^N\right)\right\} \\ XTI(n) = XT(n) \oplus Q(n) \end{cases} \tag{III.46}$$

The proposed LSPT-PRNG's function is explained in Algorithm III.3.

III.3.4 Architecture of the proposed LST_RC-PRNG

The architecture of the proposed LST_RC-PRNG is on the one hand, partly based on one of our previous PRNGs [61, 62], and on the other hand, it takes into account the vulnerabilities detected by SCA [121, 122] in one of our other PRNGs [60]. This new architecture makes it possible to resist SCAs. The proposed system comprises three one-delay recursive cells, shown in blue, containing weakly coupled chaotic maps. The three recursive one-delay cells are protected against SCAs by using a mixing technique based on three internal pseudo-random numbers: PRNL, PRNS, and PRNT respectively, depicted in red, as shown in Fig. III.5.

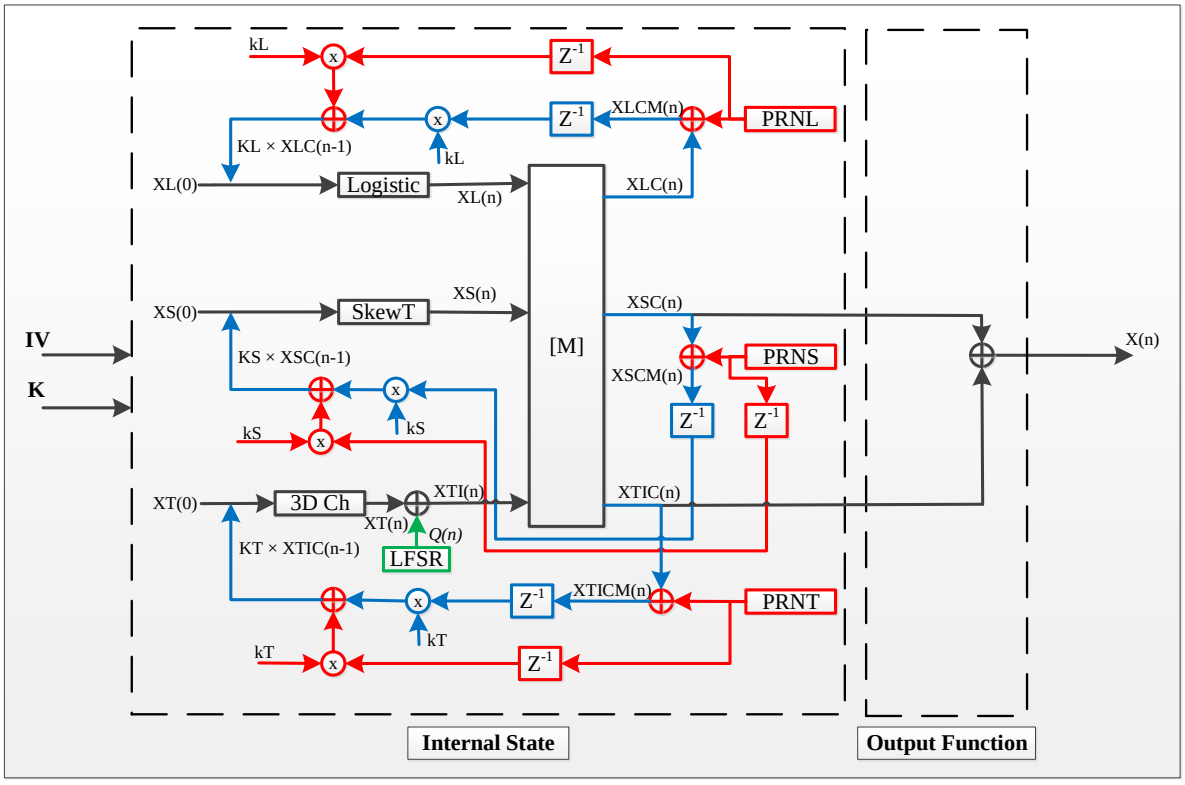


Figure. III.5: Architecture of the proposed LST_RC-PRNG

Algorithm III.3 $LSPT-PRNG$: Generation of the pseudo-random sequence X(n)

Input : IV = Initial Vector of $LSPT-PRNG$;
K = Secret key of the $LSPT-PRNG$;

Output: Keystream $X(n)$

begin

initialization

$XP(0) = mod((XP0 + IVin), 2^N)$
$XS(0) = mod((XS0 + IVin), 2^N)$
$XL(0) = mod((XL0 + IVin), 2^N)$
$XT(0) = mod((XT0 + IVin), 2^N)$

Samples generation

$M_{11} = (2^N - \epsilon_{12} - \epsilon_{13} - \epsilon_{14})$
$M_{22} = (2^N - \epsilon_{21} - \epsilon_{23} - \epsilon_{24})$
$M_{33} = (2^N - \epsilon_{31} - \epsilon_{32} - \epsilon_{34})$
$M_{44} = (2^N - \epsilon_{41} - \epsilon_{42} - \epsilon_{43})$

Calculation of the first sample

$XP(1) = PWLCM\left\{[mod(XP(0), 2^N), P_p]\right\}$
$XS(1) = SkewT\left\{[mod(XS(0), 2^N), P_s]\right\}$
$XL(1) = Logistic\left\{mod(XL(0), 2^N)\right\}$
$XT(1) = 3D\,Ch\left\{mod(XT(0), 2^N)\right\}$
$XTI(1) = XT(1) \oplus Q(1)$
$XPC(1) = (M_{11} \times XP(1)) + (\epsilon_{12} \times XS(1))) + (\epsilon_{13} \times XL(1)) + (\epsilon_{14} \times XTI(1))$
$XSC(1) = (\epsilon_{21} \times XP(1)) + (M_{22} \times XS(1)) + (\epsilon_{23} \times XL(1)) + (\epsilon_{24} \times XTI(1))$
$XLC(1) = (\epsilon_{31} \times XP(1)) + (\epsilon_{32} \times XS(1))) + (M_{33} \times XL(1)) + (\epsilon_{34} \times XTI(1))$
$XTIC(1) = (\epsilon_{41} \times XP(1)) + (\epsilon_{42} \times XS(1)) + (\epsilon_{43} \times XL(1)) + (M_{44} \times XTI(1))$

LSPT-PRNG's first sample

$X(1) = XPC(1) \oplus XSC(1) \oplus XLC(1) \oplus XTIC(1)$

while $n \geq 2\, and\, n \leq Ns$ **do**

Internal State

$XP(n) = PWLCM\left\{[mod(XPC(n-1), 2^N), P_p]\right\}$
$XS(n) = SkewT\left\{[mod(XSC(n-1), 2^N), P_s]\right\}$
$XL(n) = Logistic\left\{mod(XLC(n-1), 2^N)\right\}$
$XT(n) = 3D\,Ch\left\{mod(XTIC(n-1), 2^N)\right\}$
$XTI(n) = XT(n) \oplus Q(n)$
$XPC(n) = (M_{11} \times XP(n)) + (\epsilon_{12} \times XS(n))) + (\epsilon_{13} \times XL(n)) + (\epsilon_{14} \times XTI(n))$
$XSC(n) = (\epsilon_{21} \times XP(n)) + (M_{22} \times XS(n)) + (\epsilon_{23} \times XL(n)) + (\epsilon_{24} \times XTI(n))$
$XLC(n) = (\epsilon_{31} \times XP(n)) + (\epsilon_{32} \times XS(n))) + (M_{33} \times XL(n)) + (\epsilon_{34} \times XTI(n))$
$XTIC(n) = (\epsilon_{41} \times XP(n)) + (\epsilon_{42} \times XS(n)) + (\epsilon_{43} \times XL(n)) + (M_{44} \times XTI(n))$

LSPT-PRNG's output

$X(n) = XPC(n) \oplus XSC(n) \oplus XLC(n) \oplus XTIC(n)$

end

end

The proposed LST_RC-PRNG takes as input an initial vector (IV) and a secret key (K). The IV of the system provides the initial vectors of the three chaotic maps, IVL, IVS, and IVT; the initial condition XS0 of the Skew Tent map; and the initial seeds X0_L, X0_S, and X0_T (of 128 bits each) of the three pseudo-random numbers PRNL, PRNS, and PRNT. The output of each PRN is of size N = 32 bits. The secret key K provides the initial conditions and parameters of the LST_RC-PRNG listed in Table III.2.

Table III.2: Composition of the secret key K

Symbol	Definition
XL0 and XT0	The initial conditions of the chaotic maps: Logistic and 3D Chebyshev respectively, ranging from 1 to $2^N - 1$.
XLC1, XSC1, and XTIC1	The initial conditions of the delayed values in recursive cells: Logistic, Skew Tent, and 3D Chebyshev respectively, in the range $[1, 2^N - 1]$.
Q0	The initial value Q0 of the Linear Feedback Shift Register (LFSR).
KL, KS, and KT	The coefficients of the recursive cells: Logistic, Skew Tent, and 3D Chebyshev respectively, ranging from 1 to $2^N - 1$.
P_s	The control parameter of the Skew Tent map, in the range $[1, 2^N - 1]$.
ϵ_{ij}	The parameters of the coupling matrix M, in the interval $[1, 2^k]$ with $k \leq 5$.

Note that $XLC1$, $XSC1$, $XTIC1$ means $XLC(-1)$, $XSC(-1)$ and $XTIC(-1)$.

The secret key is produced here by Xorshif generator [123] and its size is given by:

$$\begin{aligned} |K4| = |XL0| + |XT0| + |XLC1| + |XSC1| + |XTIC1| + |Q0| \\ + |KL| + |KS| + |KT| + |P_s| + (6 \times |\varepsilon_{ij}|) = 350\, bits \end{aligned} \tag{III.47}$$

Where $|XLC1| = |XSC1| = |XTIC1| = |KL| = |KS| = |KT| = 32\, bits$.

Thus, the key space contains 2^{350} different combinations of the secret key, which is large enough to make brute force attack impracticable.

The initial values (inputs) of the three chaotic maps $XL(0)$, $XS(0)$ and $XT(0)$ are:

$$\begin{cases} XL\,(0) = \text{mod}\ (\,(XL0 + KL \times XLC1 + IVin)\,,\ 2^N) \\ XS\,(0) = \text{mod}\ (\,(XS0 + KS \times XSC1 + IVin)\,,\ 2^N) \\ XT\,(0) = \text{mod}\ (\,(XT0 + KT \times XTIC1 + IVin)\,,\ 2^N) \end{cases} \tag{III.48}$$

where:

$$IVin \ = \ IVL \ \oplus \ IVS \ \oplus \ IVT \tag{III.49}$$

The coupling system is defined by the following relation:

$$\begin{bmatrix} XLC(n) \\ XSC(n) \\ XTIC(n) \end{bmatrix} = M \times \begin{bmatrix} XL(n) \\ XS(n) \\ XTI(n) \end{bmatrix} \tag{III.50}$$

With:

$$XTI\,(n) = XT\,(n) \oplus Q\,(n) \tag{III.51}$$

And $XL(n)$, $XS(n)$, and $XT(n)$ are denoted as the maps output values at instant n of the: Logistic, Skew Tent, and 3D Ch map respectively.

The first three outputs of the matrix M, $XLC\,(1)\,,\ XSC\,(1)\,,\ XTIC\,(1)$ are given by:

$$\begin{bmatrix} XLC(1) \\ XSC(1) \\ XTIC(1) \end{bmatrix} = M \times \begin{bmatrix} XL(1) \\ XS(1) \\ XTI(1) \end{bmatrix} \tag{III.52}$$

With:

$$XTI\,(1) = XT\,(1) \oplus Q\,(1)$$

And:

$$XL(1) = Logistic\left\{\text{mod}\ \left(XL(0), 2^N\right)\right\} \tag{III.53}$$

$$XS(1) = SkewT\left\{\text{mod}\ \left(XS(0), 2^N\right),\ P_s\right\} \tag{III.54}$$

$$XT(1) = 3D\ Ch\left\{\text{mod}\ \left(XT(0),2^N\right)\right\} \tag{III.55}$$

Afterward, for $n \geq 2$ and $n \leq N_s$ the outputs of the matrix M, $XLC(n)$, $XSC(n)$ and $XTIC(n)$ are given by equation (III.50), with:

$$XL(n) = Logistic\left\{\text{mod}\left(KL \times XLC(n-1),\, 2^N\right)\right\} \tag{III.56}$$

$$XS(n) = SkewT\left\{\text{mod}\left(KS \times XSC(n-1),\, 2^N\right),\ P_s\right\} \tag{III.57}$$

$$XT(n) = 3D\ Ch\left\{\text{mod}\left(KT \times XTIC(n-1),\, 2^N\right)\right\} \tag{III.58}$$

$$XTI\,(n) = XT\,(n) \oplus Q\,(n) \tag{III.59}$$

Where N_s is the number of the desired samples, and $XLC(n-1)$, $XSC(n-1)$, $XTIC(n-1)$ are the unmasked inputs of the three chaotic maps, $Q(n)$ is the output of the LFSR, and the matrix M is :

$$M = \begin{bmatrix} M_{11} & \epsilon_{12} & \epsilon_{13} \\ \epsilon_{21} & M_{22} & \epsilon_{23} \\ \epsilon_{31} & \epsilon_{32} & M_{33} \end{bmatrix} \tag{III.60}$$

With $M_{11} = (2^N - \varepsilon_{12} - \varepsilon_{13})$, $M_{22} = (2^N - \varepsilon_{21} - \varepsilon_{23})$, and $M_{33} = (2^N - \varepsilon_{31} - \varepsilon_{32})$.

The masking operations aim to randomize the intermediate results and it is carried out by adding a random value to the outputs of the weak coupling samples $XLC(n)$, $XSC(n)$, and $XTIC(n)$.

$$\begin{aligned} XLCM(n) &= XLC(n) \oplus PRNL(n) \\ XSCM(n) &= XSC(n) \oplus PRNS(n) \\ XTICM(n) &= XTIC(n) \oplus PRNT(n) \end{aligned} \tag{III.61}$$

Where $XLCM(n)$, $XSCM(n)$ and $XTICM(n)$ represent the masked outputs of the recursive cells: Logistic, Skew Tent, and 3D Chebyshev, respectively, and $PRNL(n)$, $PRNS(n)$ and $PRNT(n)$ are random integer values generated by the Xorshift pseudo-random number generator of random integer values, in the range $[1,2^N - 1]$. To get the

same output X(n) for the same secret key and the same IV, the masking operations are reversed at the inputs of the chaotic maps.

$$\begin{aligned} XLC(n-1) \times KL{=}XLCM(n-1) \times KL \oplus PRNL(n-1) \times KL \\ XSC(n-1) \times KS{=}XSCM(n-1) \times KS \oplus PRNS(n-1) \times KS \\ XTIC(n-1) \times KT{=}XTICM(n-1) \times KT \oplus PRNT(n-1) \times KT \end{aligned} \tag{III.62}$$

Note that PRNs are based on Xoshiro's RNG, which was developed by David Blackman and Sebastiano Vigna [124] in 2019, which serves as a parameter module for PRNs. The Xoshiro construction itself is based on the Xorshift concept invented by George Marsaglia [123]. Therefore the masking operation is an effective countermeasure to protect the implementation against power analysis-based Side-Channel Attacks (SCAs) [125, 126]. Note that the VHDL implementation of these PRNs produce 32 bits at each clock cycle.

Algorithm III.4, summarizes the full operation of the proposed LST_RC-PRNG.

III.4 Hardware implementation and security analysis of the proposed PRNGs-CS

The implementation of the proposed PRNGs-CS and their corresponding stream ciphers is realized on the PYNQ Z-2 FPGA prototyping board from Xilinx. The description of these chaotic systems are done in VHDL with 32-bit fixed-point data formats, then synthesized, and implemented using the Xilinx Vivado design suite (V.2017.2). Vivado design tools essentially make it possible to carry out the various steps from design to implementation on the target FPGA board. It allows, among other things, description, synthesis, simulation, and implementation of a design, then programming it on a chip from one of the different families of Xilinx FPGAs. At the right of Fig. III.6, we summarize the different steps of the design flow that were performed under Vivado for the performance evaluation of the four proposed chaotic systems. At the left of Fig. III.6, we show the design sources, constraintes source and simultion sources for the LST_RC-PRNG system.

Algorithm III.4 $LST\_RC-PRNG$: Generation of the pseudo-random sequence $X(n)$

Input : IV = Initial Vector of $LST\_RC-PRNG$;
K = Secret key of the $LST\_RC-PRNG$;

Output: Keystream $X(n)$

begin

initialization

$XL(0) = mod((XL0 + KL \times XLC1 + IVin), 2^N)$

$XS(0) = mod((XS0 + KS \times XSC1 + IVin), 2^N)$

$XT(0) = mod((XT0 + KT \times XTIC1 + IVin), 2^N)$

Samples generation

$M_{11} = (2^N - \epsilon_{12} - \epsilon_{13})$

$M_{22} = (2^N - \epsilon_{21} - \epsilon_{23})$

$M_{33} = (2^N - \epsilon_{31} - \epsilon_{32})$

Calculation of the first sample

$XL(1) = Logistic\left\{mod(XL(0), 2^N)\right\}$

$XS(1) = SkewT\left\{[mod(XS(0), 2^N), P_s]\right\}$

$XT(1) = 3D\,Ch\left\{mod(XT(0), 2^N)\right\}$

$XTI(1) = XT(1) \oplus Q(1)$

$XLC(1) = (M_{11} \times XL(1)) + (\epsilon_{12} \times XS(1))) + (\epsilon_{13} \times XTI(1))$

$XSC(1) = (\epsilon_{21} \times XL(1)) + (M_{22} \times XS(1)) + (\epsilon_{23} \times XTI(1))$

$XTIC(1) = (\epsilon_{31} \times XL(1)) + (\epsilon_{32} \times XS(1)) + (M_{33} \times XTI(1))$

LST_RC-PRNG's first sample

$X(1) = XSC(1) \oplus XTIC(1)$

while $n \geq 2\, and\, n \leq Ns$ **do**

Internal State

Unmasking operations

$XLC(n-1) \times KL = XLCM(n-1) \times KL \oplus PRNL(n-1) \times KL$

$XSC(n-1) \times KS = XSCM(n-1) \times KS \oplus PRNS(n-1) \times KS$

$XTIC(n-1) \times KT = XTICM(n-1) \times KT \oplus PRNT(n-1) \times KT$

$XL(n) = Logistic\left\{mod(KL \times XLC(n-1), 2^N)\right\}$

$XS(n) = SkewT\left\{[mod(KS \times XSC(n-1), 2^N), P_s]\right\}$

$XT(n) = 3D\,Ch\left\{mod(KT \times XTIC(n-1), 2^N)\right\}$

$XTI(n) = XT(n) \oplus Q(n)$

$XLC(n) = (M_{11} \times XL(n)) + (\epsilon_{12} \times XS(n))) + (\epsilon_{13} \times XTI(n))$

$XSC(n) = (\epsilon_{21} \times XL(n)) + (M_{22} \times XS(n)) + (\epsilon_{23} \times XTI(n))$

$XTIC(n) = (\epsilon_{31} \times XL(n)) + (\epsilon_{32} \times XS(n)) + (M_{33} \times XTI(n))$

LST_RC-PRNG's output

$X(n) = XSC(n) \oplus XTIC(n)$

end

end

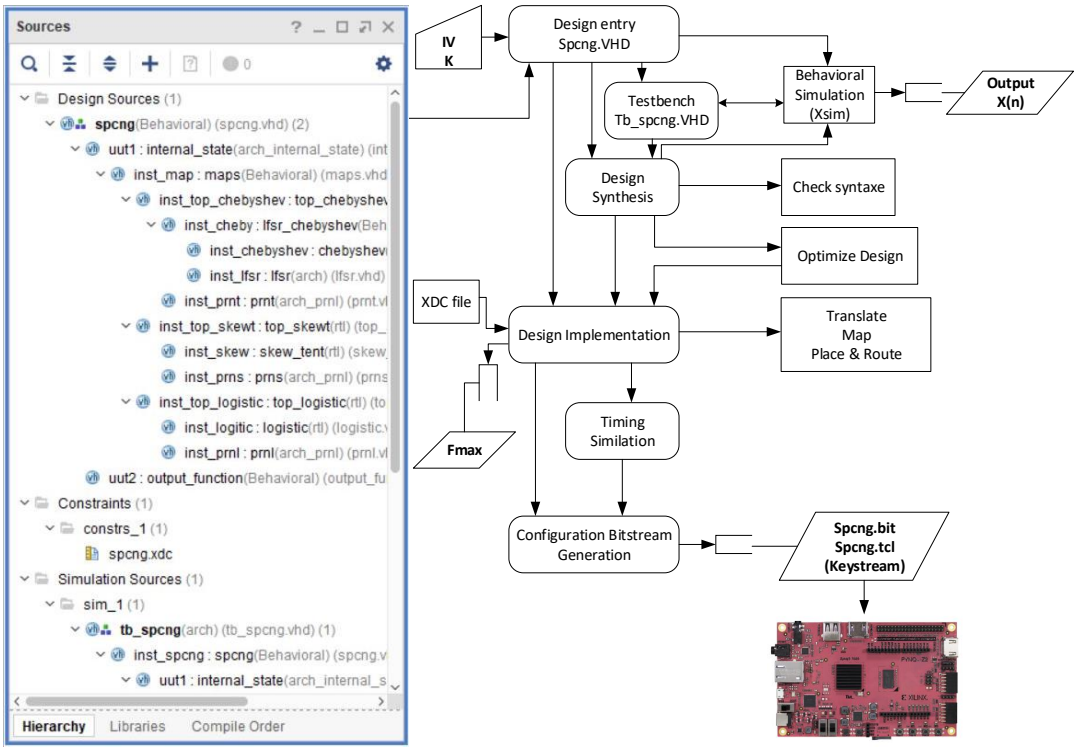


Figure. III.6: FPGA conception flow (under Vivado) of the proposed chaotic systems.

First, we describe the proposed chaotic systems using a hierarchical description containing several modules described in VHDL. Second, the synthesis step, checks the VHDL description of the PRNG-CS, converts it into a gate-level representation, and creates a netlist. Third, we perform a behavioral simulation of the chaotic systems to check their validity and make sure that the results obtained are consistent. The simulation is invoked directly by the Xsim simulator integrated into the Vivado tools and the results obtained are displayed in a chronogram (see Fig. III.7.**a**) and saved in an output test file for testing (Nist, histogram, HD, etc). At this step we can assess the statistical performance of the chaotic systems. Fourth, the design implementation performs: first, Translate, that merges the Netlists resulting from the design synthesis and the specified constraints file (Xilinx Design Constraint XDC file), then Map, which fits the design with the available resources of the target FPGA. After that, Place & Route process, places the components and routes them, respecting the constraints specified during the translation, to obtain a configuration file. At this step, we get the maximum frequency

and hardware resources summarized in the implementation reports. After the design implementation, we perform the post-implementation timing simulation to get the true timing delay information of the chaotic systems as shown in the chronogram of Fig. III.7.**b**, for the LST_RC-PRNG system as an example of illustration.

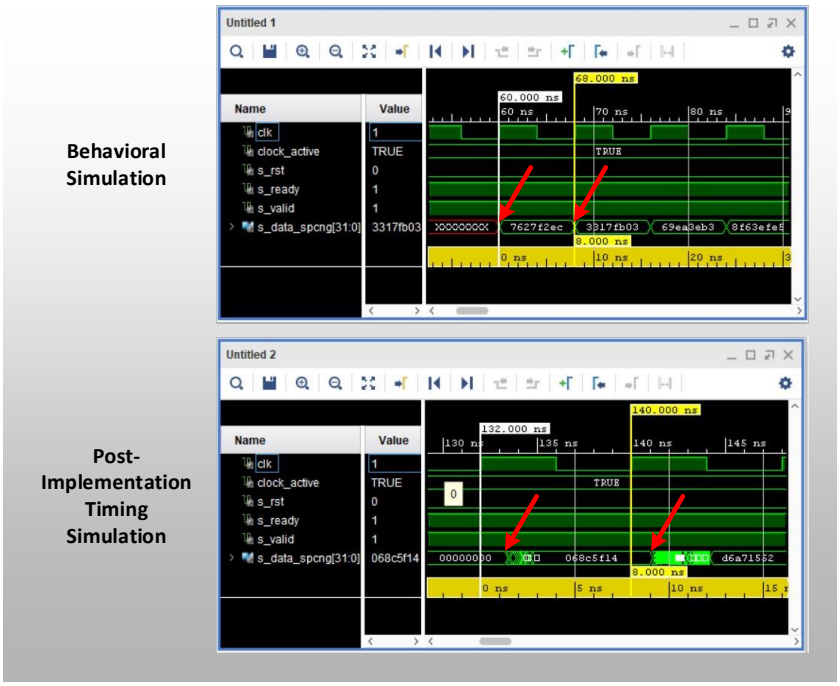


Figure. III.7: **a**. Behavioral Simulation. **b**. Post-Implementation Timing simulation.

Finally, we generate a programming file (BIT) to program the Xilinx device PYNQ-Z2 FPGA.

III.4.1 Hardware cost of the proposed PRNGs-CS

In this section, we quantify the performance of the proposed PRNGs-CS implementation in terms of resources used (Area, DSP), speed (Max_Freq, Throughput), efficiency, and power consumption. The efficiency parameter (in terms of throughput/slices) gives us an overall idea of the hardware metrics performance of the implementation. We recall below the definition of the three hardware parameters:

$$Max\_Freq = \frac{1}{Ti - WNSi}\,[MHz]\,. \tag{III.63}$$

Where Ti is the target clock period (ns) used during the implementation run "i" and $WNSi \geq 0$ is the Worst Negative Slack (ns) of the target clock used during the implementation run "i".

$$Throughput = N \times Max\_Freq\,[Mbps]\,. \tag{III.64}$$

$$Efficiency = \frac{Throughput}{Slices}\,[Mbps/Slices]\,. \tag{III.65}$$

The hardware performances of the proposed designs are obtained from the implementation of our VHDL code using Xilinx Vivado Design Suite V.2017.2. The proposed designs have been tested after place and route using timing simulation to ensure the correct functionality.

Table III.3 lists the results of our four proposed PRNGs-CS designs.

Table III.3: Harware perfromance comparison of the proposed PRNGs-CS using ZYNQ PYNQ Z2 FPGA.

			PRNGs-CS			
			LSP-PRNG	LST-PRNG	LSPT-PRNG	LST_RC-PRNG
Resources used	Area	LUTs	10,400 /19.55%	3,525 /6.63%	11,391 /21.41%	3,916 /7.36%
		FFs	349 /0.33%	434 /0.41%	480 /0.45%	981 /0.92%
		Slices	3,096 /23.28%	1,024 /7.70%	3,337 /25.09%	1,151 /8.65%
	DSPs		13 /5.91%	25 /11.36%	28 /12.73%	22 /10.00%
Speed	WNSi [ns]		0.022	0.056	0.072	0.097
	Ti [ns]		31.60	26.20	32.30	26.80
	Max. Freq. [MHz]		31.66	38.24	31.09	37.44
	Throughput [Mbps]		1,013.36	1,224	995.14	1,198.36
Efficiency [Mbps/Slices]			0.32	1.20	0.29	1.04
Power [W]			0.152	0.099	0.182	0.101

To get the overheads, we compare the implementation results obtained from our proposed PRNGs-CS architectures. Depending on the design metrics, we can choose the adequate PRNG-CS architecture suitable for the need of the application such as

FPGAS-based RFID tags [127] or FPGAS-based wireless sensor nodes [128].

From Table III.3 and Fig. III.8, we can observe that the LST-PRNG consumes the minimum of slices with 1,024 slices, and the LSPT-PRNG consumes the maximum slices with 3,337 slices.

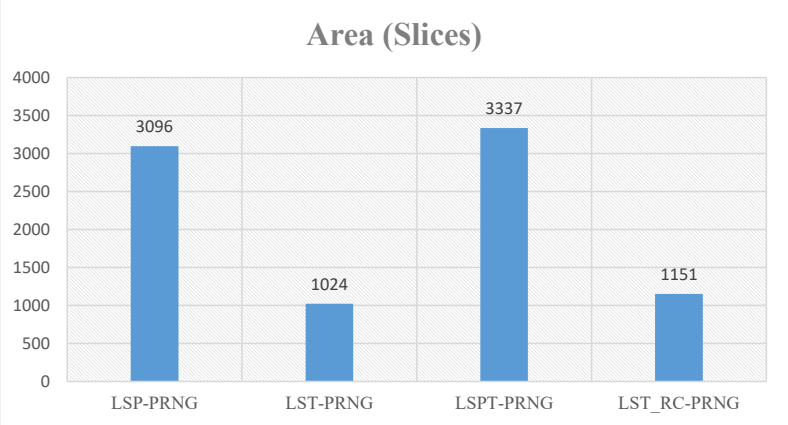


Figure. III.8: Area comparison of LSP-PRNG, LST-PRNG, LSPT-PRNG, and LST_RC-PRNG.

As shown in Fig. III.9, we notice that all the proposed PRNGs-CS provide a high speed: 1013.36 Mbps, 1224 Mbps, 995.14 Mbps and 1198.36 Mbps for the LSP-PRNG, LST-PRNG, LSPT-PRNG and LST_RC-PRNG, respectively. The LST-PRNG offers the best throughput. In addition, performance in terms of throughput and efficiency of LST-PRNG and LST_RC-PRNG are similar.

From Fig. III.10, we can notice that LST-PRNG architecture is the best suited to meet IoT application needs when efficiency is considered with 1.20 Mbps/Slices.

LST-PRNG and LST_RC-PRNG architectures provide from afar lower power consumption compared to LSP-PRNG and LSPT-PRNG with 99 mW and 101 mW respectively as depicted in Fig. III.11.

The comparison of the hardware metrics with some chaotic and non-chaotic systems will be done in the following on the various stream ciphers based on the above proposed PRNGs-CS.

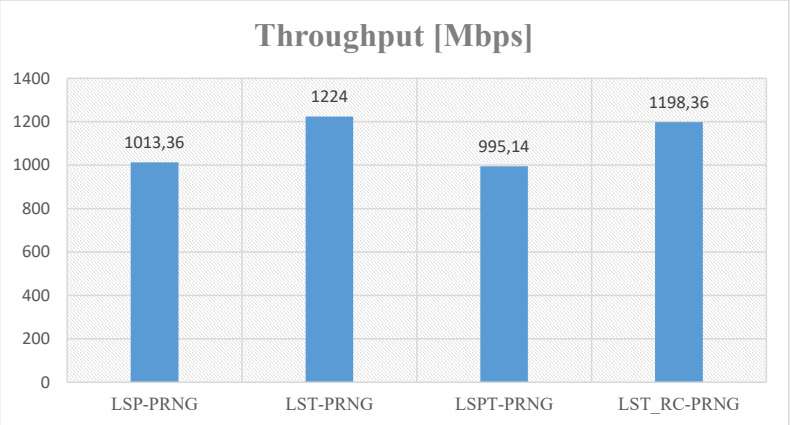


Figure. III.9: Throughput comparison of LSP-PRNG, LST-PRNG, LSPT-PRNG, and LST_RC-PRNG.

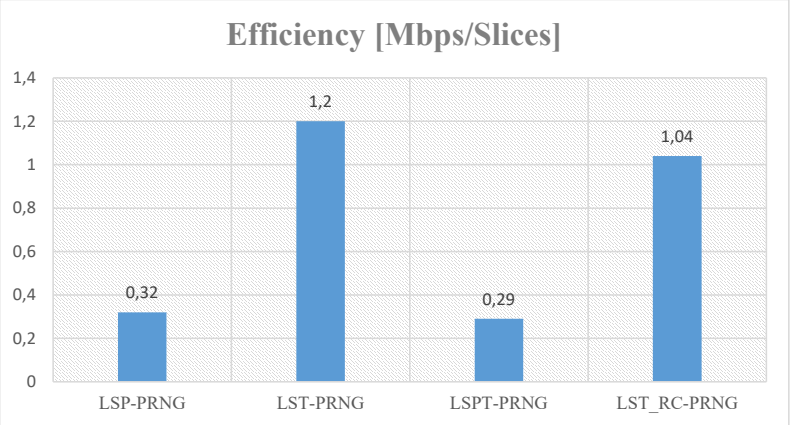


Figure. III.10: Efficiency comparison of LSP-PRNG, LST-PRNG, LSPT-PRNG, and LST_RC-PRNG.

III.4.2 PRNGs-CS resilience against statistical attacks

To quantify the statistical cryptographic properties of the sequences generated by the proposed PRNGs-CS, a series of tests must be applied. Each test measures a particular characteristic such as the correlation between generated sequences or their uniformity, and the overall results of these tests give an idea of the degree of randomness of the sequences produced. The pseudo chaotic behavior of the generated sequences is closely linked to the statistical characteristics of these sequences. The National Institute of

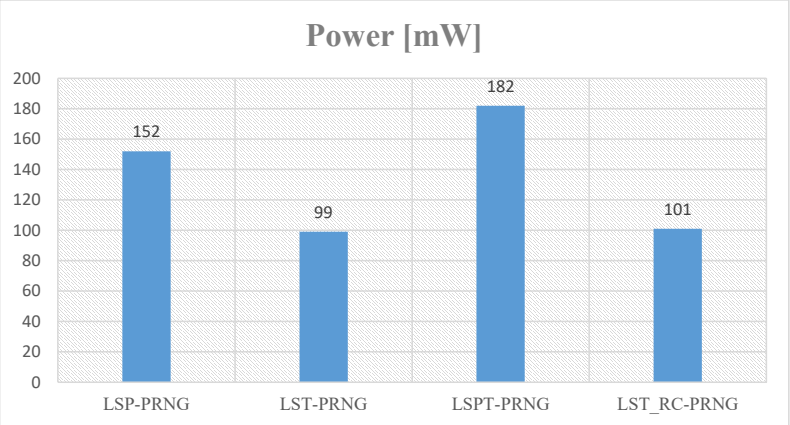


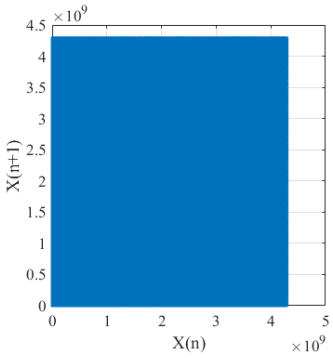
Figure. III.11: Dynamic power comparison of LSP-PRNG, LST-PRNG, LSPT-PRNG, and LST_RC-PRNG.

Standards and Technology (NIST) test serve among others tests (TestU01, DieHARD) as a reference to quantify and compare the statistical properties of any pseudo random number sequences.

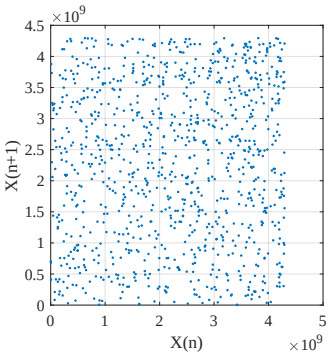
Note that the Lyapunov exponents of the chaotic maps used are positive, however it is not obvious to compute the Lyapunov exponents of the new stream cipher we propose here. Nevertheless, its chaotic nature is due mainly to the weak coupling of the chaotic maps. The weak coupling mechanism of chaotic maps have been thoroughly studied [62] leading generally to high quality of PRNGs-CS. The chaotic nature of them, is highlighted by the uncorrelated distribution of their iterates and uniformity of their histograms (Figs. III.12 and III.13).

III.4.2.1 Phase space test

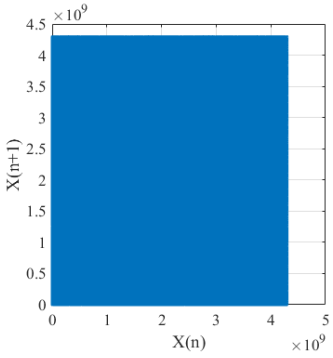
We draw in Fig. III.12 (a, c, e, g) the phase space or mapping of sequences X1, X2, X3, and X4 generated by the proposed LSP-PRNG, LST-PRNG, LSPT-PRNG, and LST_RC-PRNG respectively and formed by 3,125,000 samples out of the 3,125,100 samples generated to deviate from the transitional regime T_r= 100, and in Fig. III.12 (b, d, f, h), we show the mapping of 1000 samples (a zoom) taken randomly from X1, X2, X3, and X4 respectively.



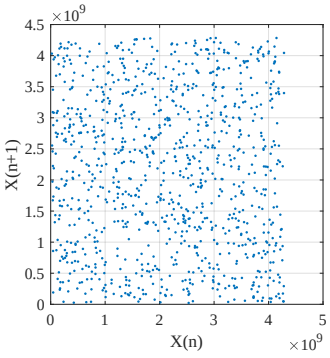
(a) Mapping of sequence X1



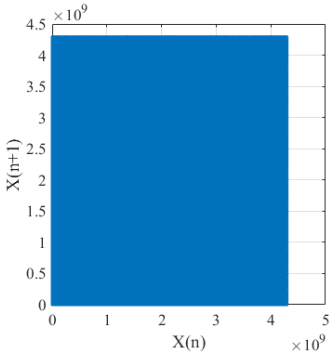
(b) Zoom on the mapping



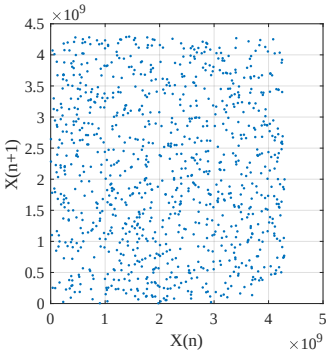
(c) Mapping of sequence X2



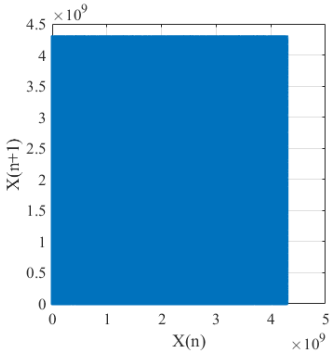
(d) Zoom on the mapping



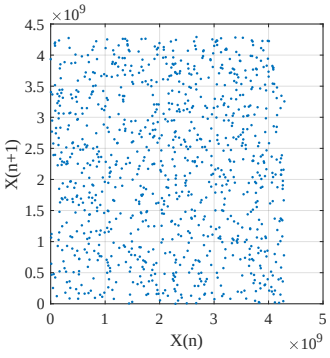
(e) Mapping of sequence X3



(f) Zoom on the mapping



(g) Mapping of sequence X4



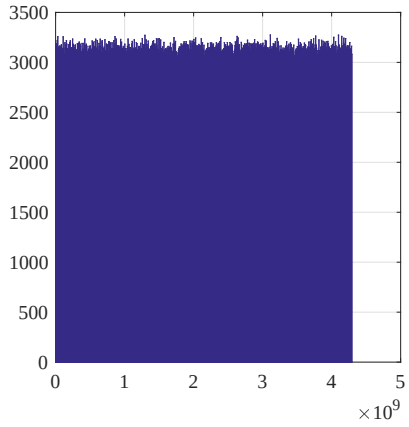
(h) Zoom on the mapping

Figure. III.12: Mapping of sequences X1, X2, X3, and X4, generated by LSP-PRNG, LST-PRNG, LSPT-PRNG, and LST_RC-PRNG respectively and a zoom of these mapping.

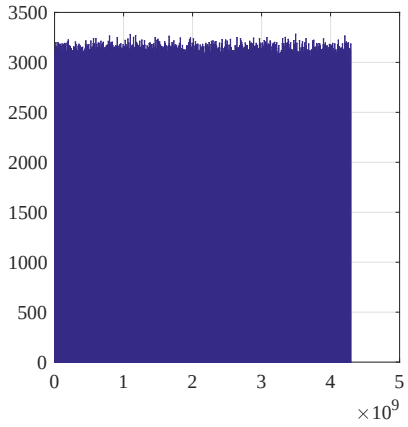
Already, from Fig. III.12.(b, d, f, h), the region looks like a totally disordered region, indicating the lack of correlation between adjacent sample values.

III.4.2.2 Histogram and Chi-square tests

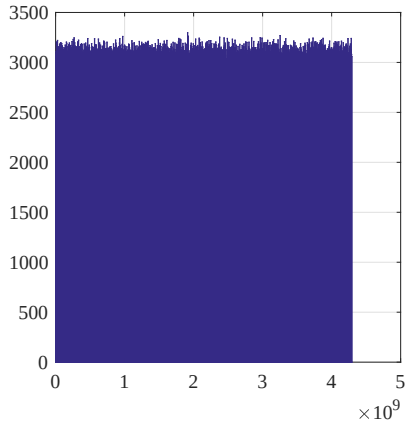
An important key property of a secure PRNG-CS is that the sequences generated should have a uniform distribution. The histogram of sequences X1, X2, X3, and X4 produced is given in Fig. III.13, the uniformity of which is observed visually.



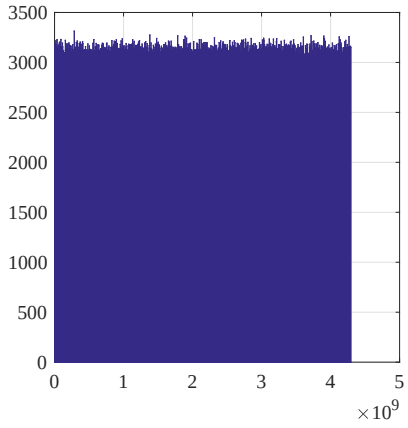
(a) The histogram of generated sequence X1



(b) The histogram of generated sequence X2



(c) The histogram of generated sequence X3



(d) The histogram of generated sequence X4

Figure. III.13: The histograms of sequences X1, X2, X3, and X4 generated by LSP-PRNG, LST-PRNG, LSPT-PRNG, and LST_RC-PRNG respectively.

The visual uniformity result should be confirmed by the chi-square test formulated as follows:

$$\chi^2_{ex} = \sum_{i=0}^{N_c-1} \frac{(O_i - E_i)^2}{E_i} \tag{III.66}$$

Where:

- $N_c = 1000$: Number of classes
- O_i: Number of calculated samples in the ith class E_i.
- $E_i = N_s/N_c$: expected number of samples of a uniform distribution.
- Ns: represent the number of samples produced, here Ns = 3,125,000

Experimental and theoretical values of the Chi-Square test for sequences X1, X2, X3, and X4 generated by the LSP-PRNG, LST-PRNG, LSPT-PRNG, and LST_RC-PRNG respectively are presented in Table III.4.

Table III.4: Theoretical and experimental values of the Chi-Square test for the proposed PRNGs-CS.

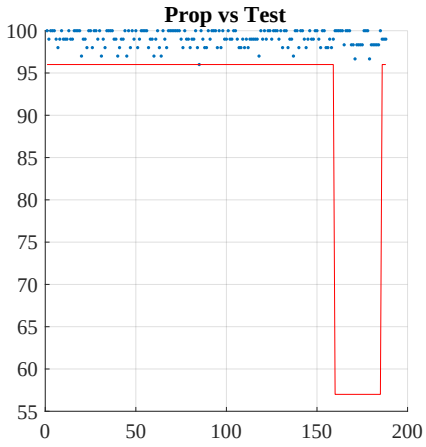
Chi-square test value	LSP-PRNG	LST-PRNG	LSPT-PRNG	LST_RC-PRNG
χ^2_{th}	1073.6426	1073.6426	1073.6426	1073.6426
χ^2_{exp}	941.5878	896.3603	915.5385	908.4748

The experimental value of the Chi-square test, for all proposed PRNG-CS, is less than the theoretical one, asserting thus the histogram uniformity. This test was performed on 100 different sequences (each containing 10^6 bits) using 5 different secret keys and all sequences were uniform.

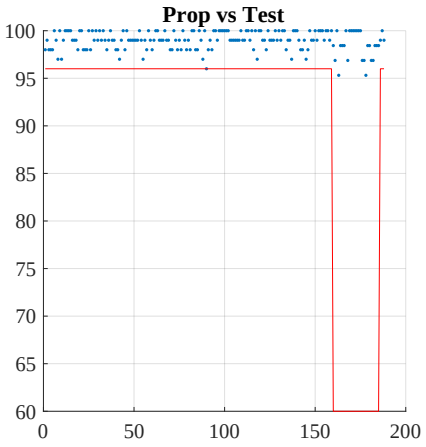
III.4.2.3 NIST test

Another important key property of a secure PRNG-CS is that the sequences generated should pass the statistical NIST test (already defined in chapter II). The results obtained, given in Fig. III.14 and Table III.5, indicate that the sequences generated by the proposed PRNGs-CS pass all the statistical 15 tests, with success.

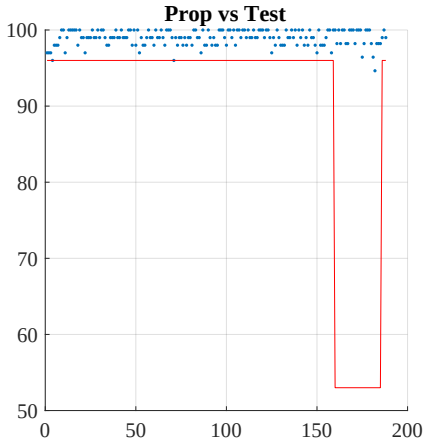
This means that the proposed PRNGs-CS produce indistinguishable sequences of integer random sequences.



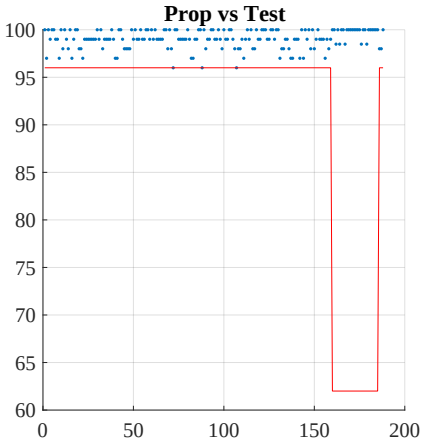
(a) NIST tests results of the proposed LSP-PRNG



(b) NIST tests results of the proposed LST-PRNG



(c) NIST tests results of the proposed LSPT-PRNG



(d) NIST tests results of the proposed LST_RC-PRNG

Figure. III.14: NIST test results of the proposed PRNGs-CS.

III.4.2.4 Key sensitivity analysis (using Hamming Distance)

The sensitivity on the key is an essential property for any PRNG. Naturally, a small change in the secret key should cause a large change in the output sequences. In order to verify this characteristic, we calculate the Hamming Distance of two sequences generated with only one bit change (least significant bit of the parameter P_s) for all proposed PRNGs-CS. We calculate the average Hamming Distance HD between two generated sequences $S1$ and $S2$, over 5 random secret keys. The $H_D(S_1, S_2)$ is defined by the

Table III.5: P-values and Proportion results of NIST test for the proposed PRNGs-CS.

	LSP-PRNG		LST-PRNG		LSPT-PRNG		LST_RC-PRNG	
Test	P-value	Prop. %	P-value	Prop. %	P-value	Prop. %	P-value	Prop. %
Frequency test	0.494	100	0.475	98	0.115	97	0.616	100
Block-frequency test	0.760	99	0.319	99	0.851	97	0.182	97
Cumulative-sums test (2)	0.757	100	0.691	98	0.394	96.500	0.825	99.5
Runs test	0.367	100	0.883	98	0.988	98	0.956	100
Longest-run test	0.983	99	0.936	100	0.437	98	0.868	100
Rank test	0.720	98	0.972	99	0.658	98	0.182	99
FFT test	0.575	98	0.475	97	0.475	99	0.868	99
Nonperiodic-templates (148)	0.527	99.115	0.489	99.074	0.484	99.054	0.507	98.912
Overlapping-templates	0.596	99	0.911	100	0.384	100	0.956	99
Universal	0.335	98	0.335	99	0.035	97	0.575	98
Approximty entropie	0.475	99	0.367	100	0.262	100	0.658	99
Random-excursions (8)	0.352	99.792	0.471	98.242	0.371	99.107	0.511	99.432
Random-excursions-variant (18)	0.468	98.611	0.367	98.438	0.374	98.710	0.376	99.832
Serial test (2)	0.460	99	0.900	99.500	0.647	99.500	0.290	98
Linear-complexity	0.401	99	0.554	99	0.071	99	0.834	100

following equation:

$$H_D\left(S_1, S_2\right) = \frac{1}{Nb} \sum_{i=1}^{Nb} \left(S_1\left(i\right) \oplus S_2\left(i\right)\right) \tag{III.67}$$

with Nb is the number of bits in an produced sequence.

The obtained average value of Hamming Distance for the proposed LSP-PRNG, LST-PRNG, LSPT-PRNG, and LST_RC-PRNG are presented in Table III.6. These values are close to the optimal value of 50%. This result illustrates the high sensitivity on the secret key of the proposed PRNGs-CS.

Table III.6: Values of HD for the proposed PRNGs-CS.

PRNG-CS	HD %
LSP-PRNG	50.0022
LST- PRNG	50.0004
LSPT- PRNG	50.0012
LST_RC- PRNG	50.0024

III.5 Performance analysis of stream ciphers based on the proposed PRNGs-CS

In this section, we first give the hardware metrics obtained by the stream ciphers (SC) based on the proposed PRNGs-CS, and compare them with those of some published systems. Then, and we assess their security against a known cryptanalytic analysis.

III.5.1 Hardware metrics of proposed chaos-based stream ciphers

To quantify the performance of the proposed implementation of the chaos-based stream ciphers in terms of resources used (Area, DSP), speed (Max_Freq, Throughput), efficiency, and power consumption, we cipher just four pixels. The proposed PRNGs-CS produce at each clock cycle a 32-bit sample, and we used the least significant 8 bits for XORing with the each pixel (8 bits) to generate the corresponding encrypted pixel. The hardware metrics results of the proposed chaos-based stream ciphers are given in Table III.7 and, as expected, they relate to the results of the various chaotic maps in Chapter II.

The comparison of the hardware metrics of the proposed chaos-based stream ciphers with several chaotic and non-chaotic systems (from eSTREAM project phase-2 focus hardware profile) is summarized in Table III.8. This comparison is difficult to interpret due to the differences in characteristics of the FPGAs tested. However, considering the clock rate of the FPGA board and the efficiency achieved, we can make this comparison. Thus, the proposed chaos-based stream ciphers present competitive hardware metrics compared to those obtained from most other chaotic and non-chaotic systems, except the Trivium cipher. However, since 2007, different types of attacks have been applied

Table III.7: Hardware metrics results of the proposed chaos-based stream ciphers.

			Chaos-based stream ciphers			
			LSP-SC	LST-SC	LSPT-SC	LST_RC-SC
Resources used	Area	LUTs	10,420 /19.59%	3,549 /6.67%	11,416 /21.46%	3,965 /7.45%
		FFS	448 /0.42%	531 /0.50%	577 /0.54%	1,071 /1.01%
		Slices	3,160 /23.71%	1,049 /7.89%	3,385 /25.45%	1,186 /8.92%
	DSPs		13 /5.91%	25 /11.36%	28 /12.73%	22 /10.00%
Speed	WNSi [ns]		0.050	0.060	0.149	0.217
	Ti [ns]		30.90	27.20	32.40	27.40
	Max. Freq. [MHz]		32.41	36.84	31.00	36.78
	Throughput [Mbps]		1,037.27	1,179.07	992.21	1,177.20
Efficiency [Mbps/Slices]			0.32	1.12	0.29	0.99
Power [W]			0.146	0.083	0.162	0.083

to eSTREAM ciphers, there by revealing some weaknesses, in particular on Trivium cipher [129, 130]. Indeed, in Trivium AND gates are the only nonlinear elements to prevent attacks that exploit, among other things, the linearity of linear feedback shift registers.

III.5.2 Cryptanalytic analysis

In order to assess the security of the proposed chaos-based stream ciphers against the most common attacks, we perform in the following the key sensitivity and the statistical analysis on various ciphered images.

III.5.2.1 Sensitivity analysis

A robust cryptosystem should also be sensitive to the secret key; that is, changing a one bit in the secret key must produce a completely different encrypted image. This sensitivity is conventionally measured by two parameters which are the NPCR (Number of

Table III.8: Hardware metrics usage comparison of several chaotic and non-chaotic systems.

Cipher	Device	Frequency [MHz]		Slices	Throughput	Efficiency
		Clock frequency	Max. Freq.		[Mbps]	[Mbps/slices]
LSP-SC	Pynq Z2	125	32.41	3,160	1,037.27	0.32
LST-SC	Pynq Z2	125	36.84	1,049	1,179.07	1.12
LSPT-SC	Pynq Z2	125	31.00	3,385	992.21	0.29
LST_RC-SC	Pynq Z2	125	36.78	1,186	1,177.20	0.99
LWCB SC [65]	Zynq7000	-	18.5	2,363 LUTs	565	-
Lorenz's chaotic System [66]	Virtex-II	50	15.598	1,926	124	0.06
Chaos-ring [67]	Virtex-6	125	464.688	1,050	464.688	0.44
Trivium [131]	Spartan 3	50	190	388	12,160	31.34
Grain-128 [132]	Virtex- II	50	181	48	181	3.77
Mickey-128 [132]	Virtex- II	50	200	190	200	1.05

Pixel Change Rate) and the UACI (Unified Average Changing Intensity) [133]. Besides, of these two parameters which operate on the bytes, we use the Hamming distance HD which operates on the bits (in our opinion HD parameter is more precise than NPCR and UACI parameters). The expressions of these parameters are given below, with C_1, C_2 are the two ciphered images of the same plain image P.

$$NPCR = \frac{1}{M \times N \times P} \sum_{i=1}^{M} \sum_{j=1}^{N} \sum_{p=1}^{P} D\left(i,j,p\right) \times 100\% \tag{III.68}$$

$$D\left(i,j,p\right) = \begin{cases} 1 & if\ C_1\left(i,j,p\right) \neq C_2\left(i,j,p\right) \\ 0 & if\ C_1\left(i,j,p\right) = C_2\left(i,j,p\right) \end{cases} \tag{III.69}$$

Where M, N, and P are the width, height, and plane sizes of C_1 and C_2. The NPCR measures the percentage of different pixel numbers between two ciphered images.

$$UACI = \frac{1}{M \times N \times P \times 255} \sum_{i=1}^{M} \sum_{j=1}^{N} \sum_{p=1}^{P} |C_1(i,j,p) - C_2(i,j,p)| \times 100\% \qquad \text{(III.70)}$$

Which measures the average intensity of differences between the two images.

For a random image, the expected values of NPCR, UACI, and HD are 99.6094%, 33.4635%, and 50% respectively. Table III.9 shows the results obtained of NPCR, UACI, and HD for the plain images Lena, Peppers, Baboon, Barbara, and Boats of the same size $512 \times 512 \times 3$ grayscale images. As we can see from these results, the NPCR, UACI, and HD values obtained are very close to the optimal values. These values indicate that the proposed chaos-based stream ciphers are very sensitive to slight modifications of the secret key.

Table III.9: NPCR, UACI and HD values

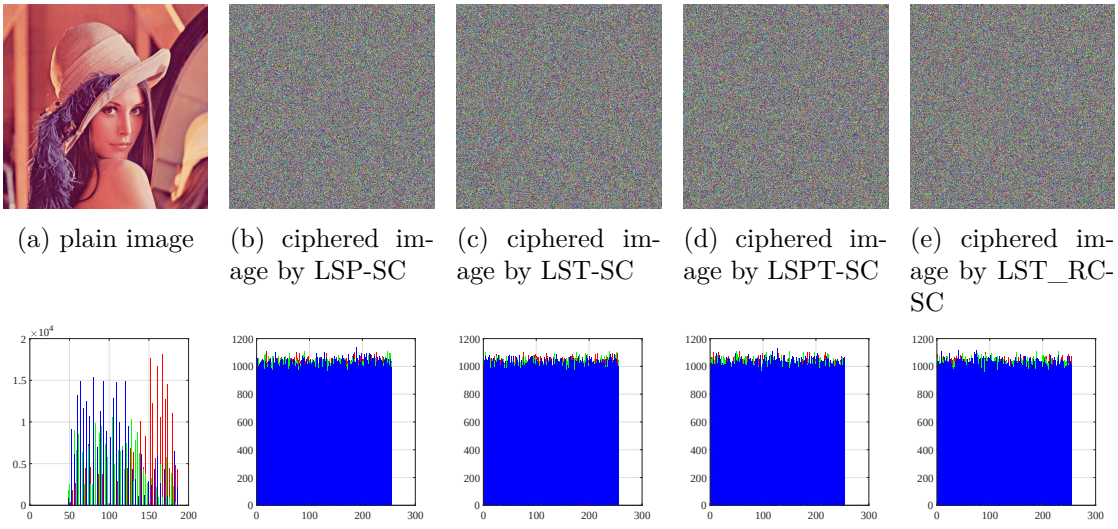
Chaos-based Stream ciphers	Test	Lena	Peppers	Baboon	Barbara	Boats
LSP- SC	NPCR %	99.5966	99.6156	99.6206	99.6206	99.5966
	UACI %	33.4374	33.4900	33.4769	33.4863	33.4377
	HD %	49.9755	50.0010	49.9868	49.9868	49.9755
LST-SC	NPCR %	99.6059	99.6111	99.6181	99.5933	99.6199
	UACI %	33.4565	33.4634	33.4486	33.4971	33.4809
	HD %	50.0162	50.0079	50.0050	49.9926	50.0480
LSPT-SC	NPCR %	99.6159	99.6135	99.6056	99.6135	99.6159
	UACI %	33.4637	33.4453	33.4751	33.4373	33.4606
	HD %	50.0391	49.9834	49.9912	49.9834	49.9912
LST_RC-SC	NPCR %	99.6129	99.6157	99.6091	99.6030	99.6092
	UACI %	33.4442	33.5027	33.4216	33.4094	33.4938
	HD %	49.9984	49.9859	49.9922	49.9921	49.9944

III.5.2.2 Statistical analysis

In order to analyze the resilience of the proposed chaos-based stream ciphers against most statistical attacks, we perform the following statistical analysis: histogram, chi-square, entropy and correlation.

❖ Histogram and chi-square analysis

The histogram of an encrypted image is an important feature in evaluating the performance of the encryption process. It illustrates how the gray levels of the pixels in an image are distributed and should be very close to a uniform distribution. In Figs. III.15, III.16, III.17, III.18, and III.19, we give the results obtained for Lena, Peppers, Baboon, Barbara, and Boats of size $512 \times 512 \times 3$, in (a) the plain images, (b) the cipher images by LSP-SC, (c) the cipher images by LST-SC, (d) the cipher images by LSPT-SC, (e) the cipher images by LST_RC-SC, and in (f), (g), (h), (i), (j) their histograms respectively.

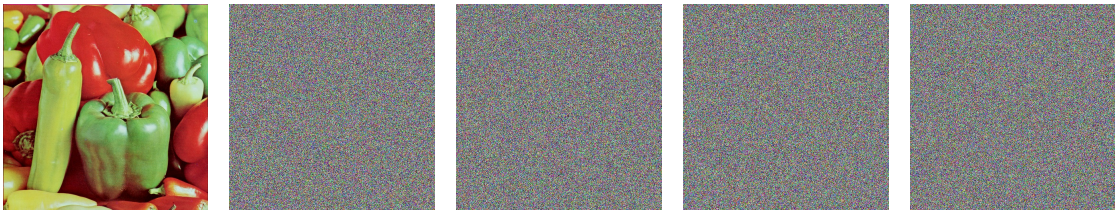


(a) plain image (b) ciphered image by LSP-SC (c) ciphered image by LST-SC (d) ciphered image by LSPT-SC (e) ciphered image by LST_RC-SC

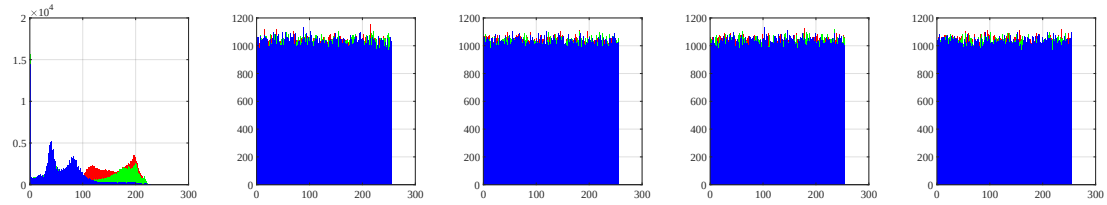
(f) histogram of plain image (g) histogram of ciphered image by LSP-SC (h) histogram of ciphered image by LST-SC (i) histogram of ciphered image by LSPT-SC (j) histogram of ciphered image by LST_RC-SC

Figure. III.15: Results of Lena image.

It is observed that the histograms of the ciphered images are very close to the uniform distribution and are completely different from the plain images. We apply the chi-square test, using equation ((III.66)), on ciphered images to statistically confirm their uniformity, where here: $N_c = 2^8 = 256$ is the number of levels, O_i is the calculated occurrence frequency of each gray level $i \in [0, 255]$ in the histogram of the ciphered image, and E_i is the expected occurrence frequency of the uniform distribution, calculated by $E_i = image\ size\ in\ bytes/N_c$. The distribution of the histograms tested are uniform if they satisfy the following condition: $\chi^2_{ex} < \chi^2_{th}(N_c - 1, \alpha) = 293.24$ *(for* $N_c = 256$ *and*

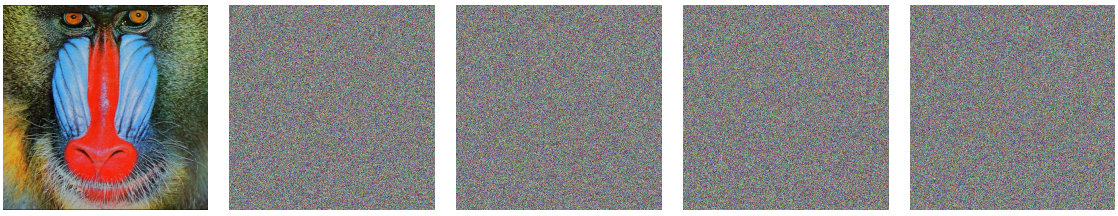


(a) plain image (b) ciphered image by LSP-SC (c) ciphered image by LST-SC (d) ciphered image by LSPT-SC (e) ciphered image by LST_RC-SC

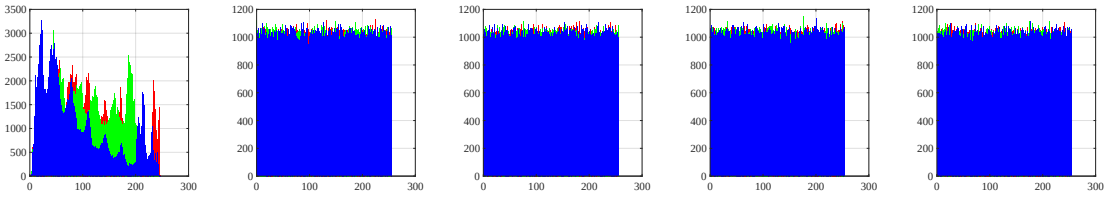


(f) histogram of plain image (g) histogram of ciphered image by LSP-SC (h) histogram of ciphered image by LST-SC (i) histogram of ciphered image by LSPT-SC (j) histogram of ciphered image by LST_RC-SC

Figure. III.16: Results of Peppers image.

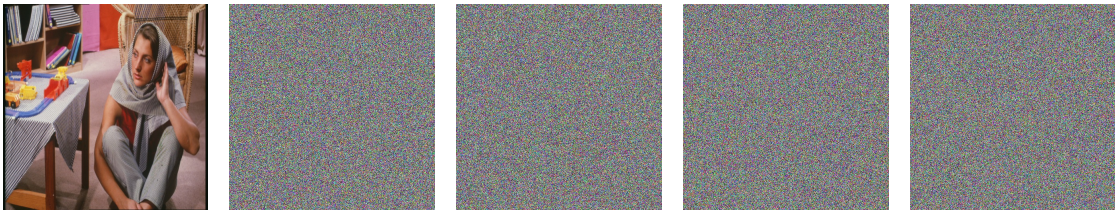


(a) plain image (b) ciphered image by LSP-SC (c) ciphered image by LST-SC (d) ciphered image by LSPT-SC (e) ciphered image by LST_RC-SC

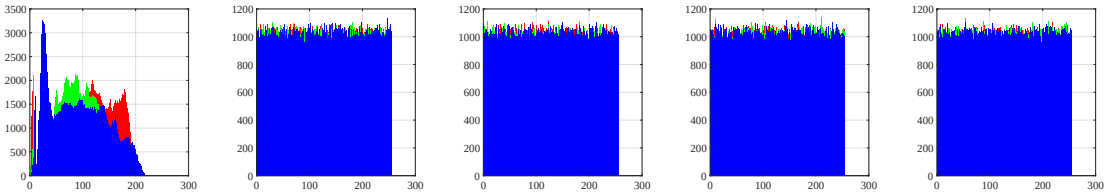


(f) histogram of plain image (g) histogram of ciphered image by LSP-SC (h) histogram of ciphered image by LST-SC (i) histogram of ciphered image by LSPT-SC (j) histogram of ciphered image by LST_RC-SC

Figure. III.17: Results of Baboon image.

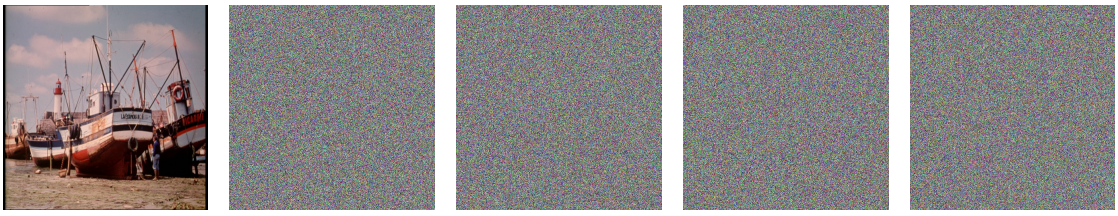


(a) plain image (b) ciphered image by LSP-SC (c) ciphered image by LST-SC (d) ciphered image by LSPT-SC (e) ciphered image by LST_RC-SC

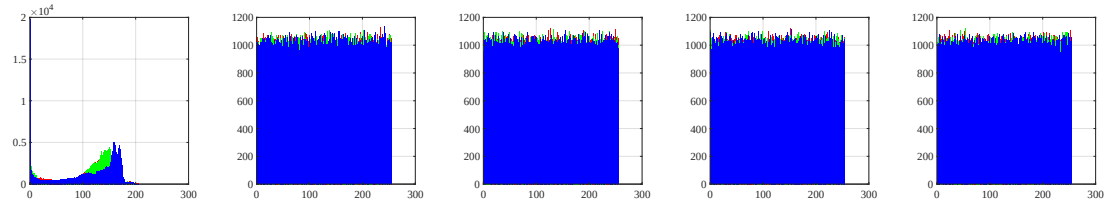


(f) histogram of plain image (g) histogram of ciphered image by LSP-SC (h) histogram of ciphered image by LST-SC (i) histogram of ciphered image by LSPT-SC (j) histogram of ciphered image by LST_RC-SC

Figure. III.18: Results of Barbara image.



(a) plain image (b) ciphered image by LSP-SC (c) ciphered image by LST-SC (d) ciphered image by LSPT-SC (e) ciphered image by LST_RC-SC



(f) histogram of plain image (g) histogram of ciphered image by LSP-SC (h) histogram of ciphered image by LST-SC (i) histogram of ciphered image by LSPT-SC (j) histogram of ciphered image by LST_RC-SC

Figure. III.19: Results of Boats image.

$\alpha = 0.05$*)*. The results obtained of the chi-square test, given in Table III.10, indicate that the histograms of the ciphered images tested are uniform because their experimental values are smaller than the theoretical values.

Table III.10: Chi-square results on the histograms tested.

Chi-square test value		Lena	Peppers	Baboon	Barbara	Boats
χ^2_{th}		293.2478	293.2478	293.2478	293.2478	293.2478
χ^2_{exp}	LSP-SC	240.8893	251.3581	257.1510	243.0104	240.6367
	LST-SC	221.4082	265.3704	241.5293	266.8132	228.1328
	LSPT-SC	264.9030	244.7487	260.0397	280.6595	254.4102
	LST_RC-SC	263.9935	245.5104	275.7038	239.2747	276.2025

❖ Entropy analysis

The random behavior of the ciphered image can be quantitatively measured by entropy information given by Shannon [134]:

$$H(C) = -\sum_{i=0}^{N_c-1} P(c_i) \times log_2(P(c_i)) \tag{III.71}$$

where $H(C)$ is the entropy of the encrypted image, and $P(c_i)$ is the probability of each gray level appearance ($c_i = 0, 1, \ldots, 255$). In the case of equal probability levels, the entropy is maximum (=8). The closer the experimental entropy value is to the maximum value, the more robust the encryption algorithm. We give in Table III.11, the results obtained from the entropy test on the plain and encrypted images. It is clear that the obtained entropies of ciphered images are close to the optimal value. Then, from these results, the proposed stream cipher has a high degree level of resilience.

Table III.11: Entropy results obtained.

Entropy		Lena	Peppers	Baboon	Barbara	Boats
Plain image		5.68222	7.66982	7.72644	7.69869	7.30541
Ciphered image	LSP-SC	7.99977	7.99976	7.99976	7.99977	7.99977
	LST- SC	7.99979	7.99975	7.99977	7.99975	7.99979
	LSPT- SC	7.99975	7.99977	7.99976	7.99973	7.99973
	LST_RC- SC	7.99975	7.99977	7.99974	7.99978	7.99974

❖ **Correlation analysis**

In an original image, each pixel is highly-correlated with adjacent pixels in a horizontal, vertical, and diagonal direction. A good encryption algorithm should produce encrypted images with correlation and redundancy as low as possible (close to zero) between adjacent pixels. To assess the correlation, we perform the following: First, we randomly select 8000 pairs of two adjacent pixels from the image under test, then we calculate the correlation coefficients by using the following equation:

$$\rho_{xy} = \frac{Cov\left(x,y\right)}{\sqrt{D\left(x\right)}\sqrt{D\left(y\right)}} \tag{III.72}$$

where:

$$Cov\left(x,y\right) = \frac{1}{N}\sum_{i=1}^{N}\left[x_i - E\left(x\right)\right]\left[y_i - E\left(y\right)\right] \tag{III.73}$$

$$E\left(x\right) = \frac{1}{N}\sum_{i=1}^{N}x_i \tag{III.74}$$

$$D\left(x\right) = \frac{1}{N}\sum_{i=1}^{N}\left[x_i - E\left(x\right)\right]^2 \tag{III.75}$$

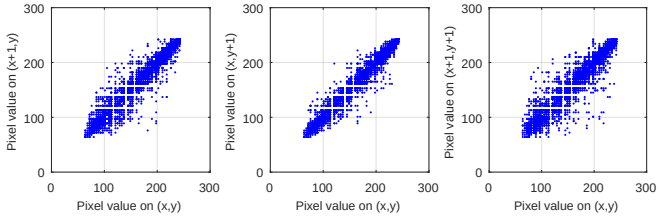
With x and y are the grayscale values of two adjacent pixels in the image under test. The obtained results are shown in Table III.12 and Figs. III.20-III.24.

It appears from Table III.12, that the correlation coefficients for the plain images are close to 1, which shows that the pixels are highly correlated, whereas for the encrypted images the correlation coefficients are close to 0 which proves that there is no correlation between the plain and ciphered images. Therefore, there is no similarity between plain and encrypted images proving the very good achieved confusion by the proposed chaos-based stream ciphers.

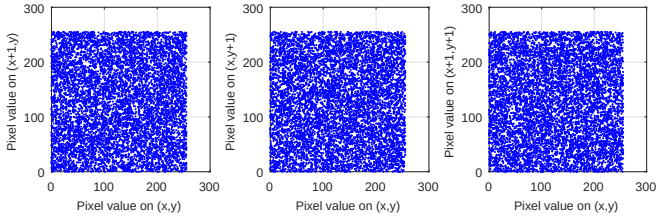
According to all these results of the histogram, entropy, and correlation, the proposed chaos-based stream ciphers present a good ability to resist statistical attacks.

Table III.12: Correlation coefficients of two adjacent pixels in the plain and ciphered images.

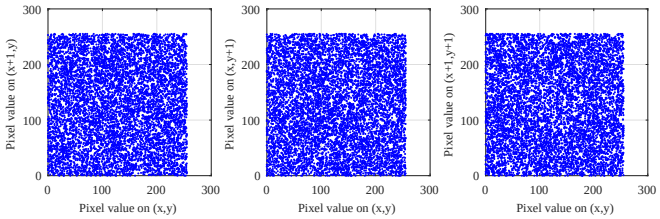
Image			Horizontal	Vertical	Diagonal
Lena	Plain image		0.97444	0.98468	0.96601
	Ciphered image	LSP-SC	0.01204	0.00549	0.00739
		LST-SC	-0.00414	0.00020	-0.00595
		LSPT-SC	-0.00770	0.00437	-0.03169
		LST_RC-SC	-0.00543	0.02178	0.00335
Peppers	Plain image		0.96216	0.96712	0.95239
	Ciphered image	LSP-SC	0.01803	-0.01284	0.00973
		LST-SC	-0.00927	-0.00896	-0.00004
		LSPT-SC	-0.00797	0.01538	-0.00102
		LST_RC-SC	-0.00805	0.02088	-0.01539
Baboon	Plain image		0.95444	0.93200	0.91957
	Ciphered image	LSP-SC	-0.00380	-0.00371	-0.00347
		LST-SC	0.00374	-0.00295	-0.01533
		LSPT-SC	-0.01895	-0.00289	-0.00559
		LST_RC-SC	-0.00582	-0.00737	0.01131
Barbara	Plain image		0.92453	0.97042	0.90707
	Ciphered image	LSP-SC	0.00939	-0.02223	0.00264
		LST-SC	-0.01218	-0.00765	0.01441
		LSPT-SC	-0.00748	0.00685	-0.00556
		LST_RC-SC	-0.01402	-0.00611	-0.00975
Boats	Plain image		0.96971	0.97280	0.94049
	Ciphered image	LSP-SC	-0.01362	0.00053	0.01487
		LST-SC	-0.01202	-0.00099	0.02195
		LSPT-SC	-0.01370	-0.00055	-0.01461
		LST_RC-SC	-0.00196	-0.00863	-0.01858



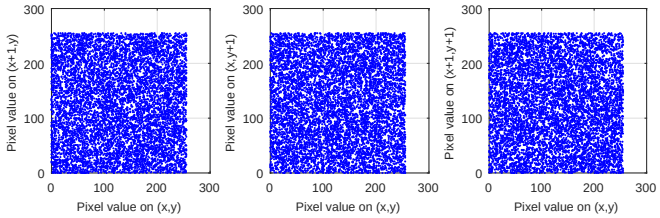
(a) Distributions of adjacent pixels of Lena image in the three directions: horizontal, vertical, and diagonal respectively.



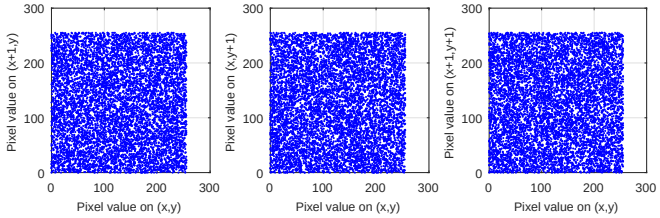
(b) Distributions of adjacent pixels of ciphered Lena by LSP-SC in the three directions: horizontal, vertical, and diagonal respectively.



(c) Distributions of adjacent pixels of ciphered Lena by LST-SC in the three directions: horizontal, vertical, and diagonal respectively.

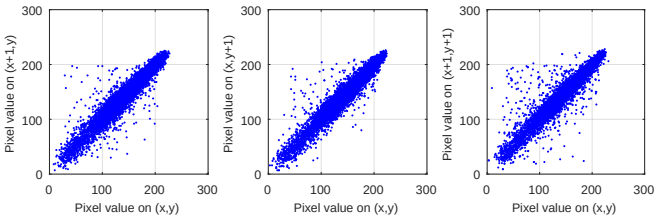


(d) Distributions of adjacent pixels of ciphered Lena by LSPT-SC in the three directions: horizontal, vertical, and diagonal respectively.

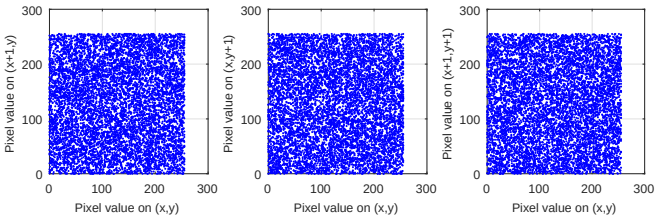


(e) Distributions of adjacent pixels of ciphered Lena by LST_RC-SC in the three directions: horizontal, vertical, and diagonal respectively.

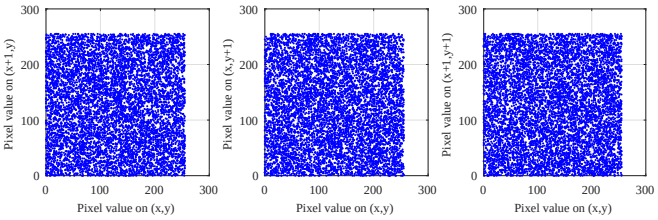
Figure. III.20: Distributions of adjacent pixels of Lena image and its ciphered images by LSP-SC, LST-SC, LSPT-SC, and LST_RC-SC respectively.



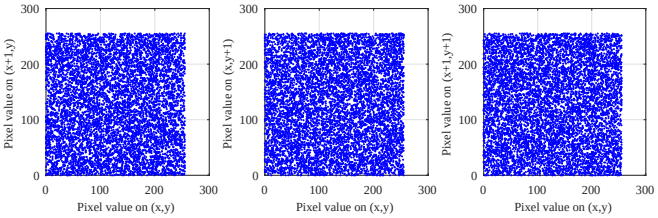
(a) Distributions of adjacent pixels of Peppers image in the three directions: horizontal, vertical, and diagonal respectively.



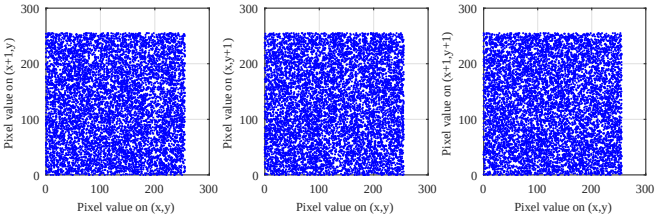
(b) Distributions of adjacent pixels of ciphered Peppers by LSP-SC in the three directions: horizontal, vertical, and diagonal respectively.



(c) Distributions of adjacent pixels of ciphered Peppers by LST-SC in the three directions: horizontal, vertical, and diagonal respectively.

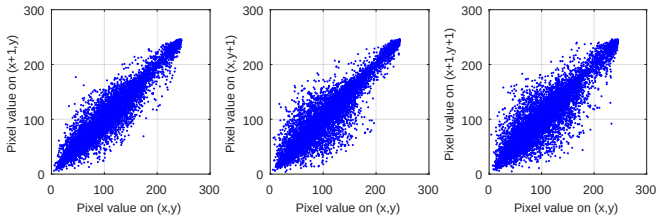


(d) Distributions of adjacent pixels of ciphered Peppers by LSPT-SC in the three directions: horizontal, vertical, and diagonal respectively.

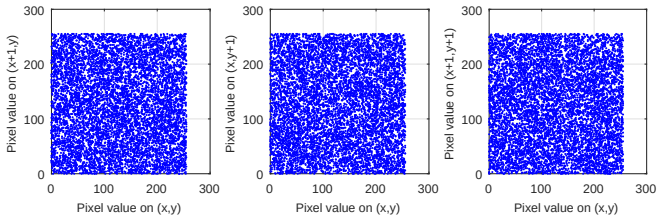


(e) Distributions of adjacent pixels of ciphered Peppers by LST_RC-SC in the three directions: horizontal, vertical, and diagonal respectively.

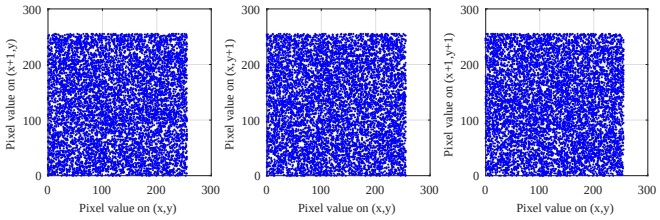
Figure. III.21: Distributions of adjacent pixels of Peppers image and its ciphered images by LSP-SC, LST-SC, LSPT-SC, and LST_RC-SC respectively.



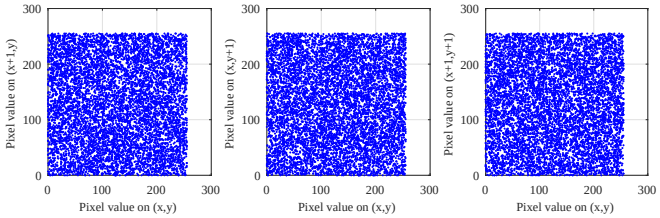
(a) Distributions of adjacent pixels of Baboon image in the three directions: horizontal, vertical, and diagonal respectively.



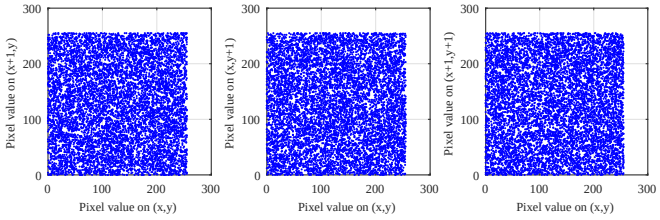
(b) Distributions of adjacent pixels of ciphered Baboon by LSP-SC in the three directions: horizontal, vertical, and diagonal respectively.



(c) Distributions of adjacent pixels of ciphered Baboon by LST-SC in the three directions: horizontal, vertical, and diagonal respectively.

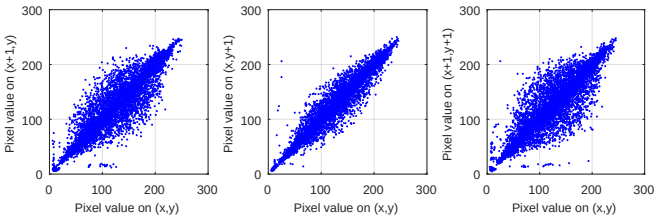


(d) Distributions of adjacent pixels of ciphered Baboon by LSPT-SC in the three directions: horizontal, vertical, and diagonal respectively.

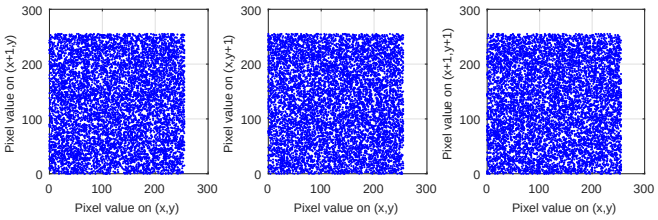


(e) Distributions of adjacent pixels of ciphered Baboon by LST_RC-SC in the three directions: horizontal, vertical, and diagonal respectively.

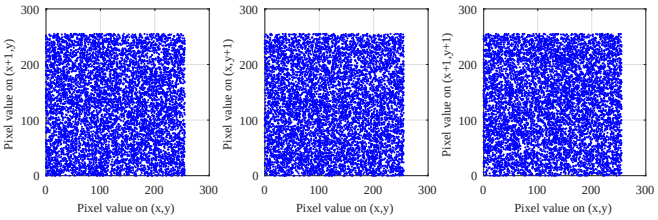
Figure. III.22: Distributions of adjacent pixels of Baboon image and its ciphered images by LSP-SC, LST-SC, LSPT-SC, and LST_RC-SC respectively.



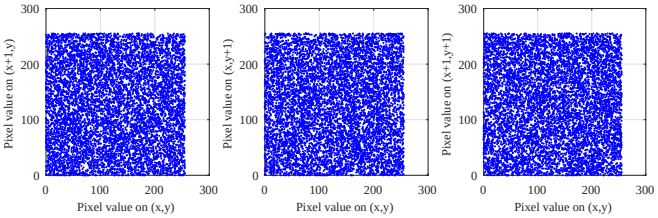
(a) Distributions of adjacent pixels of Barbara image in the three directions: horizontal, vertical, and diagonal respectively.



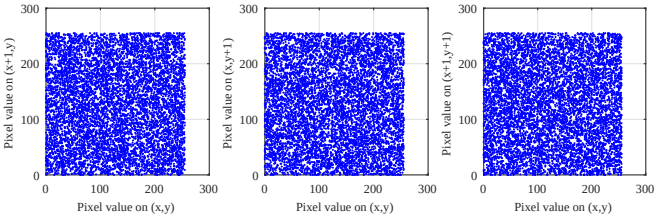
(b) Distributions of adjacent pixels of ciphered Barbara by LSP-SC in the three directions: horizontal, vertical, and diagonal respectively.



(c) Distributions of adjacent pixels of ciphered Barbara by LST-SC in the three directions: horizontal, vertical, and diagonal respectively.

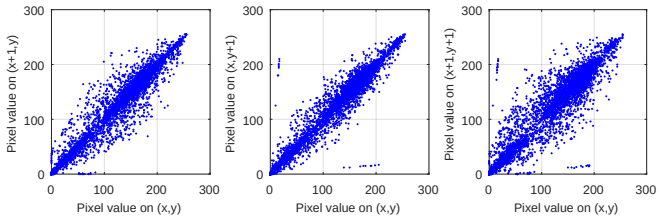


(d) Distributions of adjacent pixels of ciphered Barbara by LSPT-SC in the three directions: horizontal, vertical, and diagonal respectively.

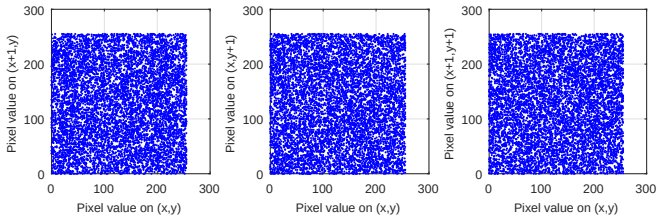


(e) Distributions of adjacent pixels of ciphered Barbara by LST_RC-SC in the three directions: horizontal, vertical, and diagonal respectively.

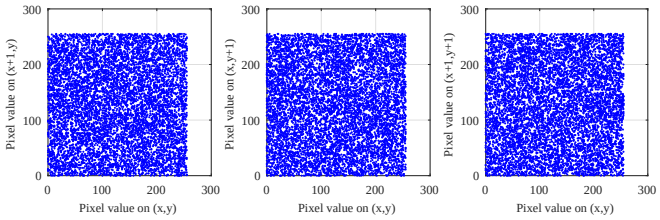
Figure. III.23: Distributions of adjacent pixels of Barbara image and its ciphered images by LSP-SC, LST-SC, LSPT-SC, and LST_RC-SC respectively.



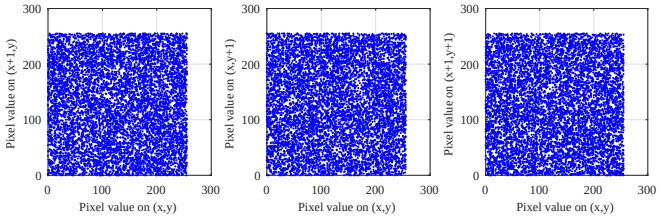
(a) Distributions of adjacent pixels of Boats image in the three directions: horizontal, vertical, and diagonal respectively.



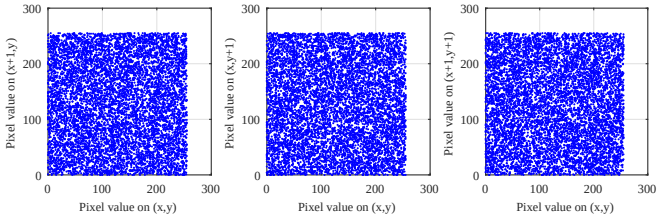
(b) Distributions of adjacent pixels of ciphered Boats by LSP-SC in the three directions: horizontal, vertical, and diagonal respectively.



(c) Distributions of adjacent pixels of ciphered Boats by LST-SC in the three directions: horizontal, vertical, and diagonal respectively.



(d) Distributions of adjacent pixels of ciphered Boats by LSPT-SC in the three directions: horizontal, vertical, and diagonal respectively.



(e) Distributions of adjacent pixels of ciphered Boats by LST_RC-SC in the three directions: horizontal, vertical, and diagonal respectively.

Figure. III.24: Distributions of adjacent pixels of Boats image and its ciphered images by LSP-SC, LST-SC, LSPT-SC, and LST_RC-SC respectively.

III.6 conclusion

In this Chapter, we studied and implemented on a Xilinx PYNQ-Z2 FPGA hardware platform using VHDL a novel chaos-based stream ciphers (SC) using a proposed secure pseudo-random number generators of chaotic sequences. The proposed chaotic systems use a weekly coupling matrix which prevents divide-and-conquer attacks on the initial vector (IV). One of the proposed systems, the LST_RC-PRNG system, includes countermeasures against side channel attacks (SCA). Next, we analyzed the cryptographic properties of the proposed PRNGs-CS and evaluated the performance of their hardware metrics. The results obtained demonstrate on the one hand, the high degree of security and on the other hand, the good hardware metrics achieved by the proposed PRNG-CS. After that, we realized the corresponding chaos-based stream ciphers and asserted their resilience against cryptanalytic attacks. Further, we evaluated their hardware metrics and compared them to those of some chaotic and non-chaotic systems. All the results obtained indicate that the proposed chaos-based stream ciphers are a good candidates for encrypting private data.

Chapter IV

Design, Implementation, and Analysis of a Block Cipher Based on a Secure Chaotic Generator

“ Crypto will not be broken, it will be bypassed. ”

Adi Shamir

Summary

IV.1 Introduction

Among the existing state-of-the-art approaches, chaos-based cryptosystem has proved to be widely efficient in data security [68–70] due, on the one hand, to the powerful characteristics of chaotic sequences such as deterministic, unpredictable, random-like behavior, and high sensitivity to initial conditions and control parameters (Secret Key) [73–75], and on the other hand, to their efficient properties of confusion and diffusion performed by chaotic maps [76–80].

Symmetric chaos-based encryption/decryption systems are classified into two categories: block ciphers and stream ciphers. The former is the main symmetric key cryptosystems, as their design is related to Shannon's theory of information security based on confusion and diffusion operations [76, 81]. Their security properties are well studied [82, 83] and these ciphers encrypt the plaintext on a block of bits: 128, 256, 512, 1024, etc.

Compared to traditional symmetric cryptography, chaos-based encryption/decryption schemes are more flexible (generic design, variable block size, and secret key can be easily enlarged), easier to implement, and have intrinsic security related to properties of chaotic sequences [82]. In addition, some of them use a substitution layer based on key-dependent S-boxes and not on fixes S-boxes as used in the Advanced Encryption Standard (AES) algorithm [84].

A key element of all chaos-based cryptosystems is the chaotic generator, which provides the dynamic keys (encryption keys) for the confusion and diffusion layer. The security, as well as the efficiency of the system, largely depend on the chaotic generator used.

However, most chaos-based cryptosystems in the literature use simple chaotic maps to supply the confusion and diffusion layer. Such cryptosystems can be broken by side-channel attacks [121, 122].

In this chapter, we propose an efficient encryption/decryption system based on the LSPT-PRNG studied in the previous chapter III and a strong key-dependent S-box. The proposed chaos-based cryptosystem design takes into account the weaknesses and strengths of the systems mentioned above. The proposed LSPT-PRNG can make the divide-and-conquer and the SCA attacks infeasible and it is used to build the key-dependent S-box. It also provides the encryption/decryption keys necessary for the substitution and permutation operation [62, 135–137].

The remainder of this chapter is organized as follows: In section IV.2, we describe the proposed chaos-based cryptosystem architecture with a deep insight of the LSPT-PRNG, the S-box with the circular substitution process, and the diffusion process, then we give the pseudo-code of the encryption/decryption algorithms. In section IV.3, we present the experiments results and security analysis of the proposed cryptosystem and its computing performance. Finally, section IV.4 summarizes the whole chapter.

IV.2 Proposed chaos-based cryptosystem

The architecture of the proposed chaos-based cryptosystem is shown in Fig. IV.1, for the encryption process and in Fig. IV.2, for the decryption process. This structure achieves high confusion-diffusion effects. On the encryption side, the confusion is achieved by a circular substitution layer based on a proposed strong S-box, which is the non-linear element of the cryptosystem as in the Advanced Encryption Standard (AES) algorithm. The diffusion is achieved by a permutation process based on the modified 2D-Cat map, followed by Horizontal Addition Diffusion (HAD) and then Vertical Addition Diffusion (VAD). The confusion and diffusion layers use the proposed LSPT-PRNG. On the decryption side, apart from the LSPT-PRNG, reverse diffusion and reverse substitution are achieved by reverse processes of those used in encryption, as shown in Fig. IV.2.

The block cipher operation can be repeated r times, if necessary, to obtain the final secure encrypted image.

In the following, we describe in detail each element of the proposed cryptosystem.

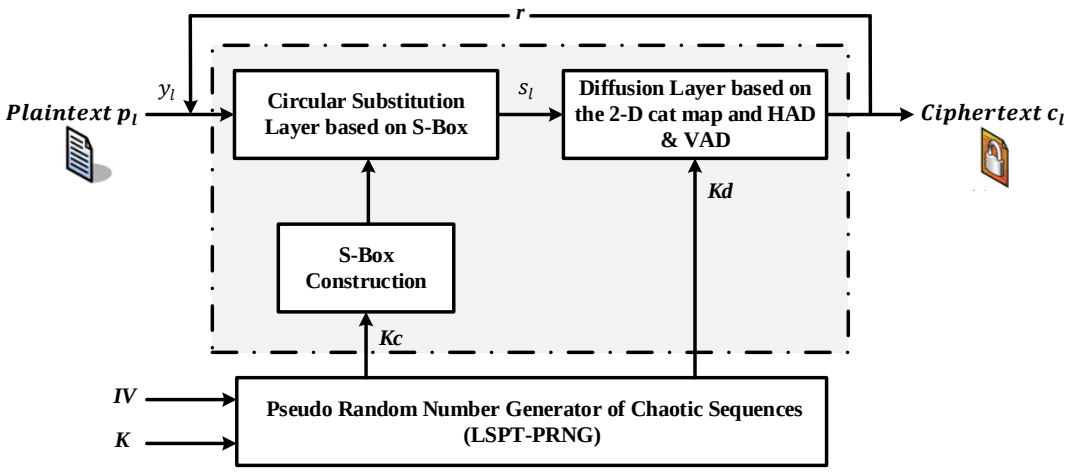


Figure. IV.1: Diagram of the encryption process.

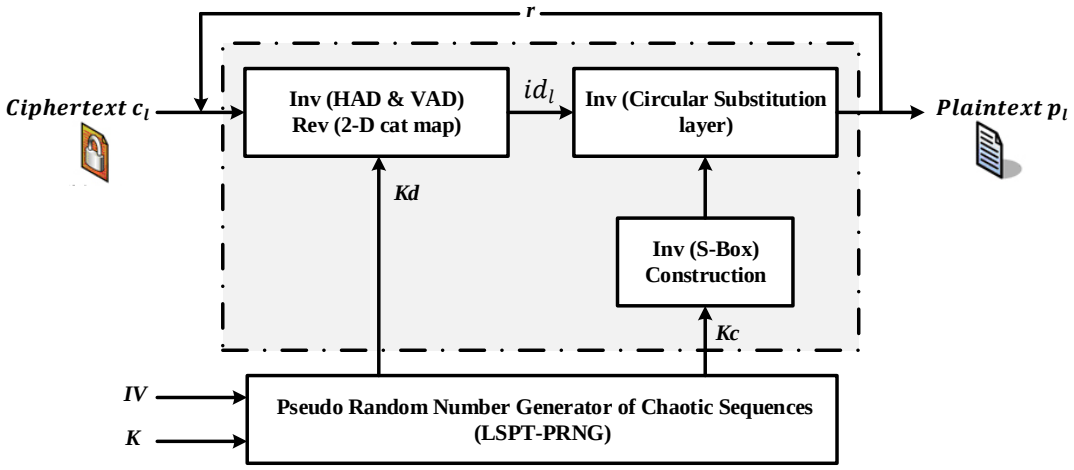


Figure. IV.2: Diagram of the decryption process.

IV.2.1 Description of the proposed pseudo-random number generator of chaotic sequences (LSPT-PRNG)

The proposed LSPT-PRNG, which is a fundamental element of any chaos-based cryptosystem, takes as input an Initial Vector (IV) and a secret Key (K) and outputs the dynamical keys (encryption/decryption keys) Kc for the confusing process and Kd for the diffusion process. it is described and detailed in chapter III (see section III.3.3). we recall its architecture in Fig. IV.3.

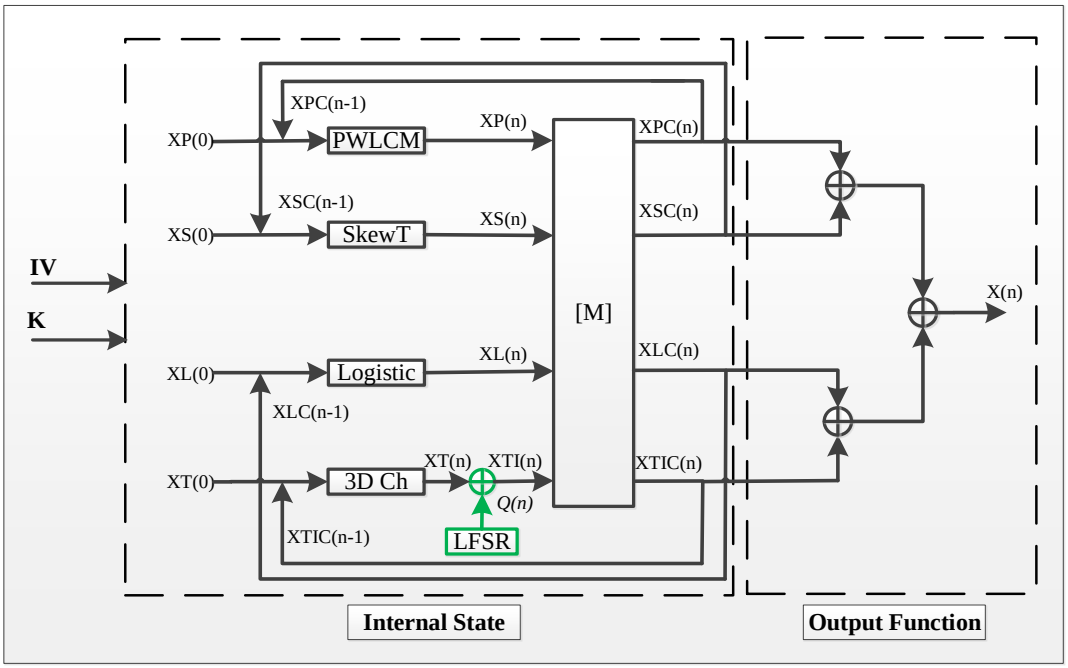


Figure. IV.3: The architecture of the proposed pseudo-random number generator of chaotic sequences.

The output X(n) is calculated by:

$$X(n) = XPC(n) \oplus XSC(n) \oplus XLC(n) \oplus XTIC(n) \tag{IV.1}$$

IV.2.2 Circular substitution process based on the S-box and their inverse processes

In the following, we first describe the construction process of the S-box, then we give the equation of the circular substitution process based on this S-box. Recall that the S-box is generally the only component of a cryptosystem that gives rise to a non-linear mapping between inputs and outputs. Its main role is to make the cryptosystem immune against differential and linear cryptanalysis, so the security of the cryptosystem is strongly increased.

IV.2.2.1 S-box construction

Mathematically, an $n \times n$ S-box is a non-linear mapping $S : \{0,\ 1\}^n \rightarrow \{0,\ 1\}^n$, where $\{0,\ 1\}^n$ represents the vector spaces of elements from GF (2). Since the key-dependent S-boxes do not offer any specific properties to the attackers, then, they are more secure than fixed S-boxes. Therefore, many methods have been proposed to design and create key-dependent S-boxes in conventional cryptography, as SEAL [120] that use Secure Hash Algorithm (SHA) and especially in chaos-based cryptography [48, 138–143]. All of these later methods are based on iterating discrete chaotic maps to generate a 1-D table of size 256, containing unique chaotic values belonging to 0 and 255. The proposed S-box is based on our LSPT-PRNG and uses the approach of Çavuşoğlu et al. [140].

The block diagram of the 8×8 S-box design algorithm is given in Fig. IV.4.

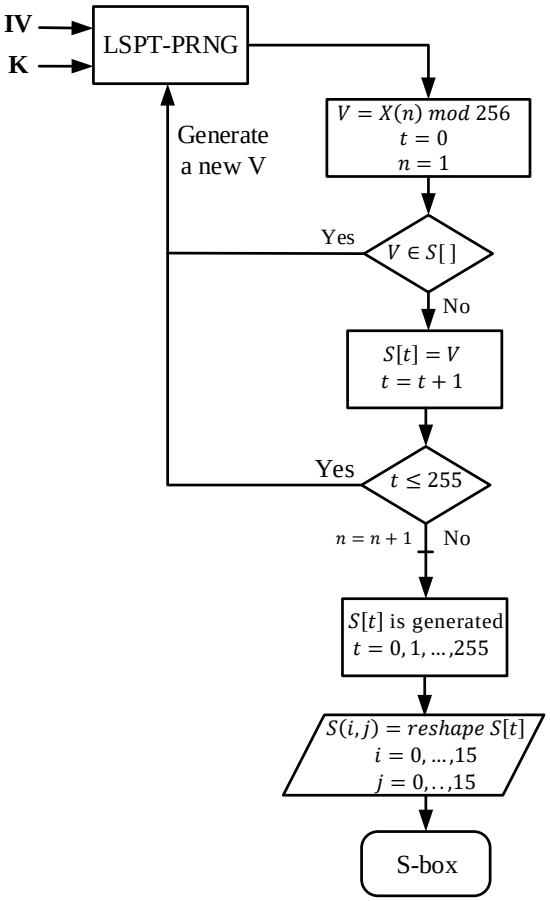


Figure. IV.4: Block diagram of the proposed chaos-based 8×8 S-box construction.

The design steps of the S-box generation algorithm are as follows:

1. define an empty 1-D S-box array $S[\,]$;
2. 8-bit LSB values of the output sequence $X(n)$ are used to obtain the value V;
3. this last value obtained V is checked whether it is in the 1-D S-box array $S[\,]$.
4. If this value is already present, it is discarded, else it is added to the 1-D S-box array $S[\,]$. Afterward, new 8-bit values are generated and repeated until all 256 values are completed in the form of a sequence $S[t] = \{S[0], \ldots, S[255]\}$, where each value $\in [0,\ 255]$ is unique to others in it, so the constructed 8×8 S-box is obviously bijective (by construction).

The number n of needed samples to construct the S-box is the size of the dynamic key K_c: $|K_c| = n \times 32 - bit$.

In Table IV.1, we give an example of an S-box obtained by our proposed algorithm, presented in its usual form, a 16×16 matrix of byte values in hexadecimal presentation.

Table IV.1: An example of an S-box is obtained by the proposed algorithm.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	2E	3E	5D	03	7C	B7	AD	9B	E0	62	67	50	95	BC	24	83
1	A6	CB	DD	F5	BE	39	9A	C3	2C	64	EA	EC	05	3B	26	2A
2	F8	B4	D0	C5	30	FD	E6	69	70	74	7A	77	F7	1A	E5	DF
3	15	7F	8B	32	51	F1	C9	93	40	5C	48	8E	2F	65	E2	CC
4	FB	D8	BD	04	C7	0B	E1	06	90	B6	A1	EF	14	2B	D5	82
5	4B	D1	B8	59	92	E3	CD	B9	38	1F	71	73	EE	F9	46	45
6	C0	A8	00	6E	9C	6A	9F	A9	78	6F	87	5E	D9	BA	47	42
7	68	DA	F0	53	58	31	D3	36	34	20	33	8D	D4	8C	4C	3D
8	1D	4E	61	E9	5A	5F	4D	6B	AA	C4	91	3F	4A	1E	02	0F
9	09	72	88	CA	F2	85	99	9D	FF	CF	B2	2D	52	23	E4	3C
A	C8	EB	AF	81	1C	FE	9E	B5	7B	55	ED	07	41	89	94	7E
B	12	11	D7	49	AB	4F	21	DE	0C	D6	17	BF	56	6D	F3	01
C	1B	A5	3A	8A	27	E7	8F	0E	A3	75	BB	84	76	CE	25	B1
D	FA	DB	18	B0	A0	C1	44	54	A7	28	43	13	22	A2	35	FC
E	C6	C2	66	63	AC	A4	7D	F6	96	29	10	86	16	D2	37	6C
F	79	08	97	E8	DC	0A	19	0D	B3	98	80	AE	F4	57	5B	60

IV.2.2.2 Circular substitution equation based on the constructed S-box.

Once the S-box is constructed, then we use it to perform the substitution operation. Unlike the circular substitution used by Zhang et al. [93], which is performed in ECB mode on the entire plain image, we achieve here the circular substitution in CBC mode on blocks of size 1024-pixel each. The advantages of proceeding in CBC mode, on the one hand, because it is a secure mode (ECB mode is not secure), on the other hand, the encryption/decryption processes are carried out on blocks that can be stored and computed on IoT devices which are constrained resource in terms of computing capabilities, and also energy and memory capacities. In addition, in ECB mode, the decryption process requires receiving the entire ciphered image before decrypting it.

To start the substitution process, the input plain image is split into b_l blocks p_l, each size $|p_l| = 1024$ pixels (32×32 bytes). Notice that the elements of the S-box are considered to be circular (the last element follows the first element) with a head pointer h initialized to a constant (or a random) value in the range of 0 to 255. Each pixel $p_l(k)$, of a given block l, is substituted by an element of the S-box, $s_l(k)$ according to the pixel value $p_l(k)$, the previous substituted pixel $q_{l-1}(k)$ and the pointer h (see Eqs (IV.2), (IV.3), and (IV.4)). After a pixel is substituted, the value of the pointer h is adapted to a new value using the second row of Eq (IV.2), where $m_h \in [0,\ 255]$ is a random value.

$$\begin{cases} s_l(k) = S[y_l(k) + h] \ mod\ 256 \\ h = s_l(k) \oplus m_h \end{cases} \tag{IV.2}$$

Where:

$$y_l(k) = p_l(k) + q_{l-1}(k) \tag{IV.3}$$

$$q_{l-1}(k) = \begin{cases} ivb & if\ l = 0 \\ c_{l-1}(k) & if\ l > 0 \end{cases} \tag{IV.4}$$

With $c_{l-1}(k)$ is the previous ciphered pixel obtained after the horizontal and vertical addition diffusion of the block sp_{l-1}, which is the permuted of the substituted block s_{l-1}, $k = 1, 2, \ldots, |p_l|$ $l = 1, 2, \ldots, b_l$

$b_l = (image\ size/block\ size) = (R \times C \times P/|p_l|)$ (R: number of rows, C: number of

columns, and P: number of planes ($P = 1$ for a grayscale image and $P = 3$ for an RGB image).

From Eqs (IV.2), (IV.3), and (IV.4), we can observe that all the same pixel values of the plain image are substituted by different values, and then the security is strengthened against cryptographic attacks. In addition, there is an intrinsic diffusion effect in each substituted block.

IV.2.2.3 Inverse substitution process

The reverse substitution operation is carried out by first, the construction of the inverse S-box, IS which is derived from the S-box $S(t)$ by a simple operation satisfying the following equation:

$$IS\left[S\left(t\right)\right] = t \tag{IV.5}$$

used during the inverse substitution process;

then, performing the inverse substitution itself using the following formula to recover each plain pixel $p_l\left(k\right)$:

$$\begin{cases} p_l\left(k\right) = \left(IS[s_l\left(k\right)] + 256 - h\right) \; mod \; 256 \\ h = s_l\left(k\right) \oplus m_h \end{cases} \tag{IV.6}$$

IV.2.3 Diffusion process and its inverse

The diffusion process of the proposed cryptosystem is realized by using a permutation layer based on the modified 2-D cat map followed by a Horizontal Addition Diffusion (HAD) & a Vertical Addition Diffusion (VAD).

IV.2.3.1 Permutation layer based on the modified 2-D cat map and its reverse

To permute a substituted block $s_l\left(k\right)$, using the modified 2-D cat map, we transform it to a matrix form $s_l\left(i,\ j\right)$ such that [88]:

$$s_l\left(i,\ j\right) = s_l\left(k\right), \quad k = \left(i-1\right) \times M + j; \quad i = 1, \cdots M; \quad j = 1, \cdots M \tag{IV.7}$$

The equation of the modified 2-D cat map [144] and its optimized implementation [88] are given below:

$$\begin{bmatrix} i_n \\ j_n \end{bmatrix} = Mod\left(\begin{bmatrix} 1 & u_d \\ v_d & (1+u_d v_d) \end{bmatrix} \begin{bmatrix} i \\ j \end{bmatrix} + \begin{bmatrix} r_{id}+r_{jd} \\ r_{jd} \end{bmatrix}, \begin{bmatrix} M \\ M \end{bmatrix} \right) \tag{IV.8}$$

$$\begin{cases} i_n = Mod\left((i + u_d \times j + Z_1), M \right) \\ j_n = Mod\left((Z_3 + Z_2 \times j), M \right) \end{cases} \tag{IV.9}$$

With: $Z_1 = r_{id} + r_{jd}$, $Z_2 = u_d \times v_d + 1$, and $Z_3 = v_d \times i + r_{jd}$.

The values of Z_1 and Z_2 are calculated once per round and the value of Z_3 is calculated M times per round ($with\ M = \sqrt{|p_l|} = 32 = 2^q - 1$).

The position of each permuted element $(i_n,\ j_n)$ is unique, so the 2-D cat map is bijective. However, the cat map is not an invertible function because of the modulo operation but it is reversible. So, the same following relation can be used to perform the permutation and the reverse permutation:

$$sp_l\left(i_n,\ j_n\right) = s_l\left(i,\ j\right), \quad i = 1, \cdots M, \quad j = 1, \cdots M \tag{IV.10}$$

The dynamic key Kd is formed by four elements $Kd = \{u_d, v_d,\ r_{id}, r_{jd}\}$, $each \in [1,\ M-1]$ and it is fed by the LSPT-PRNG. The size Kd is $|Kd| = 4 \times q = 20 - bit$.

IV.2.3.2 Horizontal and vertical addition diffusion HAD & VAD and their inverses

The operations of horizontal and vertical addition diffusion [91] on a given permuted block $sp_l\left(i, j\right)$ are described by the following mathematical model, and shown in Fig. IV.5:

$$sph_l(i,j) = HAD\left[sp_l(i,j)\right]$$
$$= \begin{cases} sp_l(i,j) + sp_l(i,j-1) \ \ mod\ 256 & for\ 1 \le i \le M\ and\ 1 < j \le M \\ sp_l(i,j) + sp_l(i-1,\ M) \ \ mod\ 256 & for\ 1 < i \le M\ and\ j = 1 \\ sp_l(i,j) + sp_l(M,M)\ mod\ 256 & for\ i = j = 1 \end{cases} \tag{IV.11}$$

$$c_l(i,j) = VAD\left[sph_l(i,j)\right]$$
$$= \begin{cases} sph_l(i,j) + sph_l(i-1,j) \ \ mod\ 256 & for\ 1 < i \le M\ and\ 1 \le j \le M \\ sph_l(i,j) + sph_l(M,\ j-1) \ \ mod\ 256 & for\ i = 1\ and\ 1 < j \le M \\ sph_l(i,j) + sph_l(M,M)\ mod\ 2^L & for\ i = j = 1 \end{cases} \tag{IV.12}$$

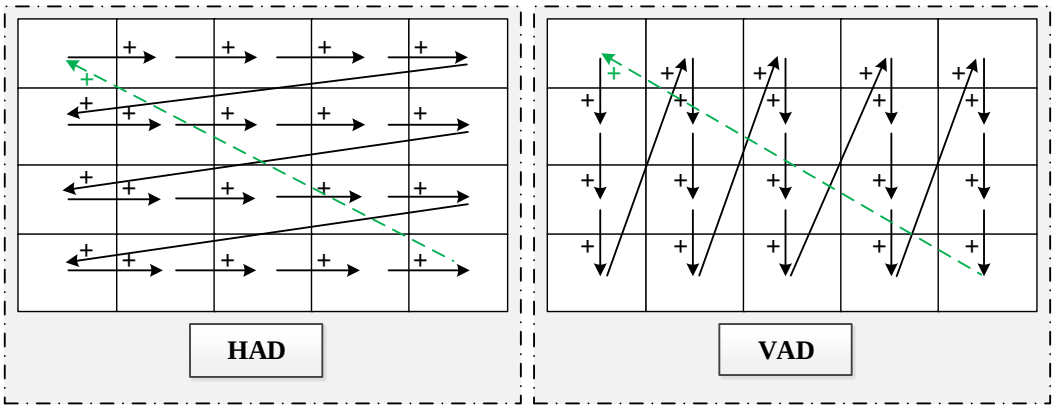


Figure. IV.5: Horizontal and Vertical Addition Diffusion.

The inverse relations of the vertical and horizontal addition diffusion are given by:

$$sph_l(i,j) = InvVAD\left[c_l(i,j)\right]$$
$$= \begin{cases} c_l(i,j) - c_l(i-1,j) \ \ mod\ 256 & for\ 1 < i \leq M\ and\ 1 \leq j \leq M \\ c_l(i,j) - c_l(M,j-1) \ \ mod\ 256 & for\ i = 1\ and\ 1 < j \leq M \\ c_l(i,j) - c_l(M,M) \ \ mod\ 256 & for\ i = j = 1 \end{cases} \tag{IV.13}$$

$$sp_l(i,j) = InvHAD\left[sph_l(i,j)\right]$$
$$= \begin{cases} sph_l(i,j) - sph_l(i,j-1) \ \ mod\ 256 & for\ 1 \leq i \leq M\ and\ 1 < j \leq M \\ sph_l(i,j) - sph_l(i-1,M) \ \ mod\ 256 & for\ 1 < i \leq M\ and\ j = 1 \ \ and\ j = 1 \\ sph_l(i,j) - sph_l(M,M)\, mod\ 256 & for\ i = j = 1 \end{cases} \tag{IV.14}$$

Finally, we summarize in algorithms IV.1 and IV.2, the full operation of encryption and decryption process respectively.

IV.3 Experiments results and security analysis

Hereafter, we quantify the performance of the proposed chaos-based cryptosystem. First, we estimate the number r of rounds required to have a secure system, then we assess the performance of the proposed chaotic generator, histogram chi-square, test de Nist on the generated sequences, and finally, we analyze the security of the cryptosystem against the usual cryptographic attacks.

All the simulations were implemented using MATLAB (R2016b) on a PC with a 2.5 GHz processor Intel Core i5-7300HQ CPU, 16GB RAM, under Windows 10 Professional, and 64-bit operating system.

Algorithm IV.1 Encryption process of the proposed chaos-based cryptosystem.

Input : IV = Initial Vector of $LSPT - PRNG$;
K = Secret key of the $LSPT - PRNG$;
ivb = Initialization vector used in the first block;
b_l = Number of blocks;
p_l = Plaintext block;
$LSPT - PRNG(IV, K) = (Kc, Kd)$;

Output: The block under test is encrypted;

begin
  Generate the needed dynamic keys $|Kc|$ to construct the S-box and $|Kd|$ to perform the permutation;
  Produce the initial vector ivb with a prng to encrypt the first block and produce h, m_h values (with a prng);
  Construct the S-box using the four steps of Fig. IV.4;
  for $m = 1$ *to* r **do**
    for $l = 1$ *to* b_l **do**
      Substitution process
      for $k = 1$ *to* $|p_l|$ **do**
        Calculate $y_l(k)$ by Eqs (IV.3), (IV.4);
        Perform the substitution process using Eq (IV.2) to obtain $s_l(k)$;
      end
      Permutation process for the diffusion
      $M = \sqrt{|p_l|}$;
      for $i = 1$ *to* M **do**
        for $j = 1$ *to* M **do**
          Calculate $k = (i-1) \times M + j$;
          Perform the permutation of pixels: $sp_l(i_n, j_n) = s_l(i, j)$;
        end
      end
      HAD diffusion
      for $i = 1$ *to* M **do**
        for $j = 1$ *to* M **do**
          Calculate horizontal diffusion using Eq. (IV.11);
        end
      end
      VAD diffusion
      for $i = 1$ *to* M **do**
        for $j = 1$ *to* M **do**
          Calculate vertical diffusion using Eq. (IV.12);
        end
      end
    end
  end
end

Algorithm IV.2 Decryption process of the proposed chaos-based cryptosystem.

Input : IV = Initial Vector of $LSPT - PRNG$;
K = Secret key of the $LSPT - PRNG$;
ivb = Initialization vector used in the first block;
b_l = Number of blocks;
p_l = Plaintext block;
$LSPT - PRNG(IV, K) = (Kc, Kd)$;

Output: The block under test is decrypted;

begin
  Generate the needed dynamic keys $|Kc|$ to construct the S-box and $|Kd|$ to perform the permutation;
  Produce the initial vector ivb with a prng to encrypt the first block and produce h, m_h values (with a prng);
  Construct the S-box using the four steps of Fig. IV.4;
  for $m = r$ *to* 1 **do**
    for $l = 1$ *to* b_l **do**
      Inverse VAD diffusion
      for $i = M$ *to* 1 **do**
        for $j = M$ *to* 1 **do**
          Calculate inverse vertical diffusion using Eq. (IV.13);
        end
      end
      Inverse HAD diffusion
      for $i = M$ *to* 1 **do**
        for $j = M$ *to* 1 **do**
          Calculate inverse horizontal diffusion using Eq. (IV.14);
        end
      end
      Reverse Permutation process for the diffusion
      $M = \sqrt{|p_l|}$;
      for $i = 1$ *to* M **do**
        for $j = 1$ *to* M **do**
          Calculate $k = (i-1) \times M + j$;
          Perform $c_l(i, j) = c_l(k)$;
          Calculate $[i_n, j_n]$ using Eq. (IV.9);
          Perform the permutation of pixels: $c_l(i_n, j_n) = id_l(i, j)$;
        end
      end
      Inverse Substitution process
      for $k = 1$ *to* $|p_l|$ **do**
        Calculate $y_l(k)$ by Eq. (IV.5);
        Perform the substitution process using relation (IV.6) to obtain $p_l(k)$;
      end
    end
  end
end

IV.3.1 Estimation of the number of rounds required

In this section, we provide the number r of rounds that will be used in the chaos-based cryptosystem implementation. For this, we calculate the average Hamming Distance (HD) between two ciphered images C_1, using 100 secret keys, and C_2 using the same secret keys, each with a different LSB-bit. This test is applied to 10 different plain images. The $HD(C_1,\ C_2)$ is defined by the following equation:

$$HD\left(C_1,\ C_2\right) = \frac{1}{Nb}\sum_{i=1}^{Nb}\left(C_1\left(i\right) \oplus\ C_2(i)\right) \tag{IV.15}$$

with Nb is the number of bits in an encrypted image. The results obtained of the Hamming distance for $r = 1,\ 2,\ and\ 3$ are listed in Table IV.2.

Table IV.2: HD versus the round times in the encryption process.

Image	Size	HD (%)		
		$r = 1$	$r = 2$	$r = 3$
Airplane	512×512×3	50.0010	49.9974	49.9998
Black	256×256×1	50.0027	50.0050	49.9977
Bridge	512×512×1	50.0039	49.9962	50.0021
Cameraman	256×256×1	49.9936	50.0018	50.0000
Flowers	256×256×3	49.9993	50.0034	49.9955
Goldhill	512×512×3	49.9999	49.9988	49.9996
Kiel	512×512×1	50.0010	49.9989	50.0036
Lena	512×512×3	50.0001	49.9997	49.9972
Sailboat	512×512×3	49.9961	50.0050	50.0000
White	256×256×1	50.0107	49.9948	50.0009

It is noteworthy, from Table IV.2, that for $r \geq 1$ the HDs for the various plain images encrypted by the proposed encryption algorithm are close to the optimal value of 50%. So, for all subsequent tests, we use $r\ =\ 1$.

This result is also confirmed by the plain text sensitivity test carried out in paragraph IV.3.3.3, see Fig. IV.17.

IV.3.2 Performance analysis of the proposed PRNG-CS

The performances obtained of the proposed chaotic generator, such as histogram and chi-square test, NIST test, and key sensitivity analysis using Hamming distance are

analyzed in Chapter III section III.4.2.

IV.3.3 Security analysis of the proposed chaos-based cryptosystem

In this section we first give the performance of the proposed S-box, then we evaluate the security of the proposed chaotic encryption system in terms of statistical attacks and cryptanalytic analysis and finally, we estimate the computing performance.

IV.3.3.1 Performance analysis of the proposed key-dependent S-Box

A strong S-box should have some important properties, based on information theory analysis. The main properties, besides the bijectivity, are non-linearity, strict avalanche criterion (SAC), output bits independence criterion (BIC), equiprobable input/output XOR distribution, differential approximation probability (DP), and linear approximation probability (LP). To demonstrate the performance of the proposed S-box, we quantify these properties and compare the results obtained with those obtained from several S-boxes in the literature.

From Table IV.3, we can observe that the constructed S-box meets all the requirements of a robust S-box and the performance obtained are close to those obtained by [140–142, 145, 146].

Table IV.3: Performance comparison of different S-boxes.

S-box	Non-linearity			SAC			BIC-SAC	BIC Non-linearity	DP	LP
	Min	Avg	Max	Min	Avg	Max				
Proposed S-box	102	104	108	0.4375	0.4948	0.5625	0.4991	103.42	10	0.1094
Çavuşoğlu et al. [140]	104	106	110	0.4218	0.5039	0.5937	0.5058	103.40	10	0.1406
Lambić et al. [141]	106	106.75	108	-	0.5034	-	0.5014	103.78	10	0.1328
Ahmed et al. [142]	106	107.5	108	-	0.4943	-	0.4982	104.35	10	0.1250
Lai et al. [145]	104	105	110	0.3906	0.5014	0.5937	0.5028	102.75	10	0.1250
Al Solami et al. [146]	106	108.5	110	-	0.5017	-	0.5026	104	10	0.1094

You can see from Table IV.1 that the constructed S-box has all the different output values of the interval $[0,\ 255]$, so it satisfies the requirement of bijectivity.

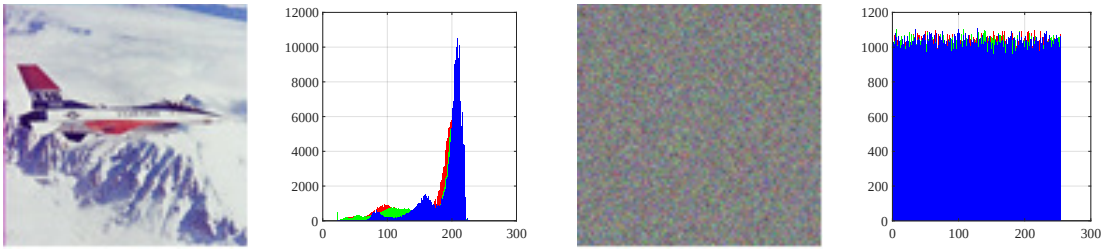
By comparing the properties of our S-box with other S-boxes, we can claim that the proposed chaos-based S-box is better than others in some measures. Therefore, the proposed S-box is suitable for use in the proposed encryption scheme.

IV.3.3.2 Statistical analysis

To prove the robustness of the proposed chaos-based cryptosystem against statistical attacks, we perform the following experiments on various plain images: histogram, chi-square test, correlation, and entropy analysis.

❖ **Histograms of encrypted images**

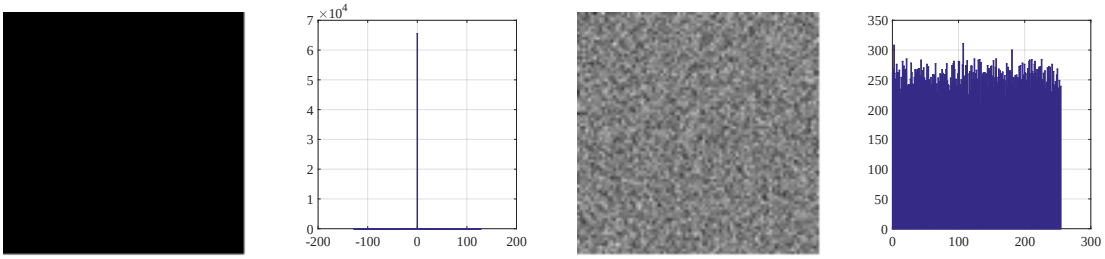
The encrypted image should have a uniform distribution and this is a basic condition for any cryptosystem to be strong against statistical attacks. We encrypted 10 gray and color plain images with different sizes and features and then we deducted the histograms of the plain images and corresponding ciphered images. The obtained results are given in Figs. IV.6-IV.15, and on each figure, we show: (a) the plain image, (b) the histogram of the plain image, (c) the corresponding ciphered image, and (d) the histogram of the ciphered image.



(a) (b) (c) (d)

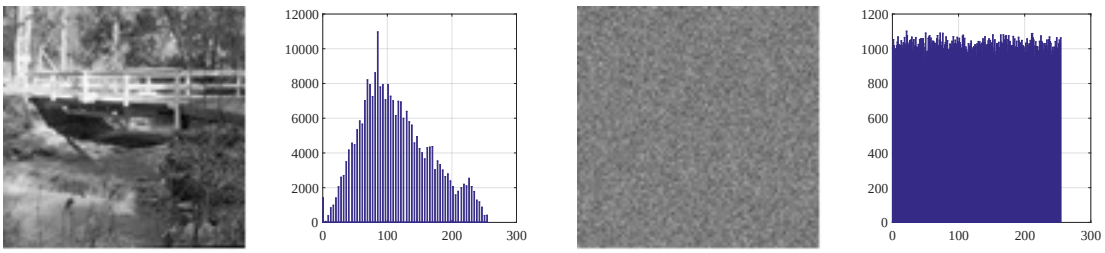
Figure. IV.6: Histograms: (a) image Airplane, (b) histogram of plain Airplane, (c) encrypted Airplane, and (d) histogram of encrypted Airplane.

Note that visually the histograms of the encrypted images appear to have a uniform distribution and are very different from those of the single images. This means that there is no visual information that can be observed from the ciphered image of the proposed chaos-based cryptosystem.



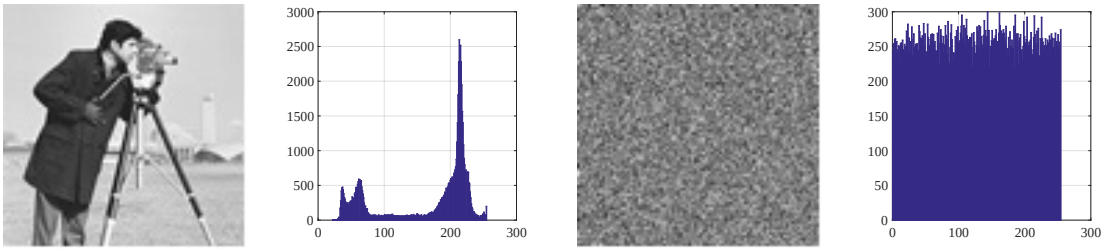
(a) (b) (c) (d)

Figure. IV.7: Histograms: (a) image Black, (b) histogram of plain Black, (c) encrypted Black, and (d) histogram of encrypted Black.



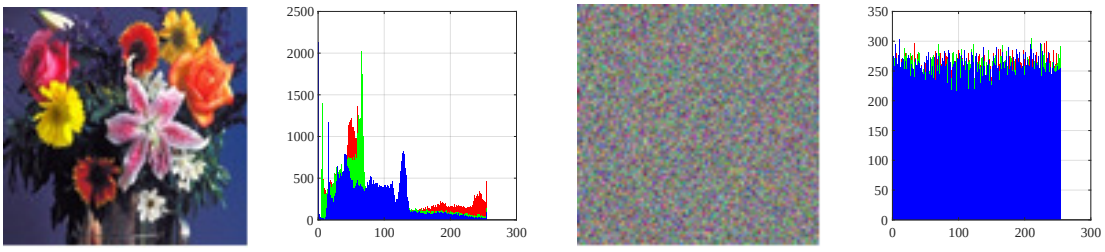
(a) (b) (c) (d)

Figure. IV.8: Histograms: (a) image Bridge, (b) histogram of plain Bridge, (c) encrypted Bridge, and (d) histogram of encrypted Bridge.



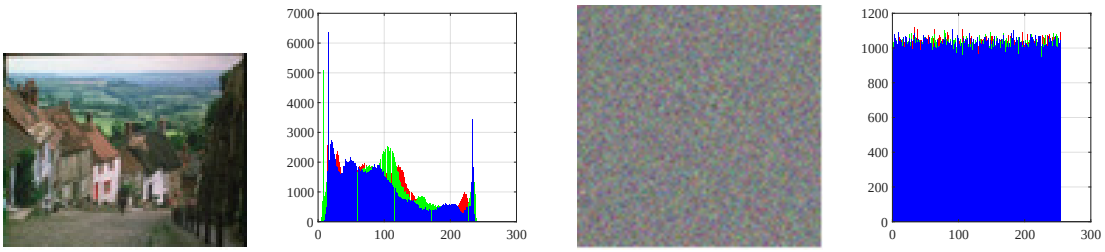
(a) (b) (c) (d)

Figure. IV.9: Histograms: (a) image Cameraman, (b) histogram of plain Cameraman, (c) encrypted Cameraman, and (d) histogram of encrypted Cameraman.



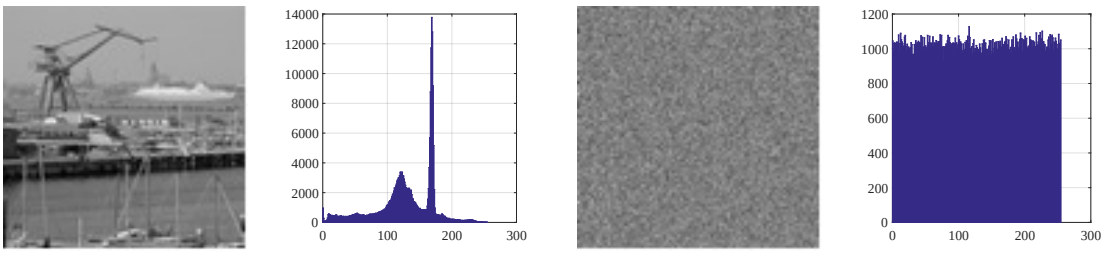
(a) (b) (c) (d)

Figure. IV.10: Histograms: (a) image Flowers, (b) histogram of plain Flowers, (c) encrypted Flowers, and (d) histogram of encrypted Flowers.



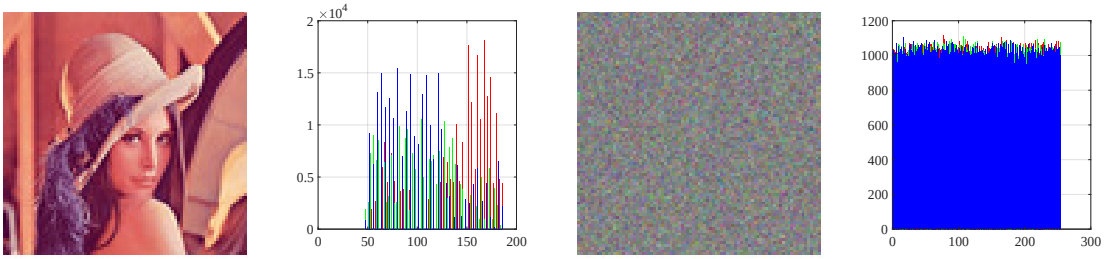
(a) (b) (c) (d)

Figure. IV.11: Histograms: (a) image Goldhill, (b) histogram of plain Goldhill, (c) encrypted Goldhill, and (d) histogram of encrypted Goldhill.



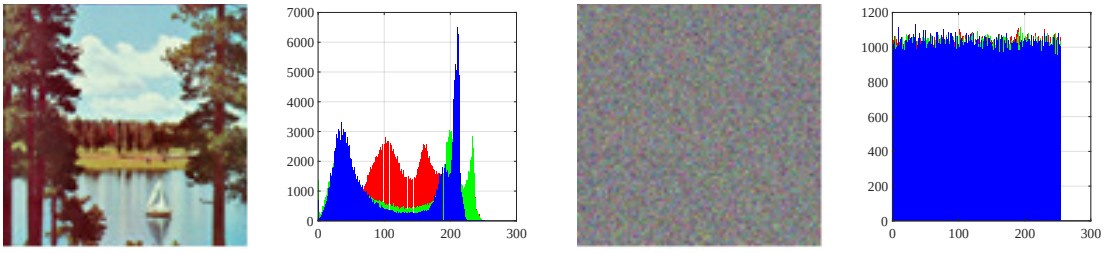
(a) (b) (c) (d)

Figure. IV.12: Histograms: (a) image Kiel, (b) histogram of plain Kiel, (c) encrypted Kiel, and (d) histogram of encrypted Kiel.



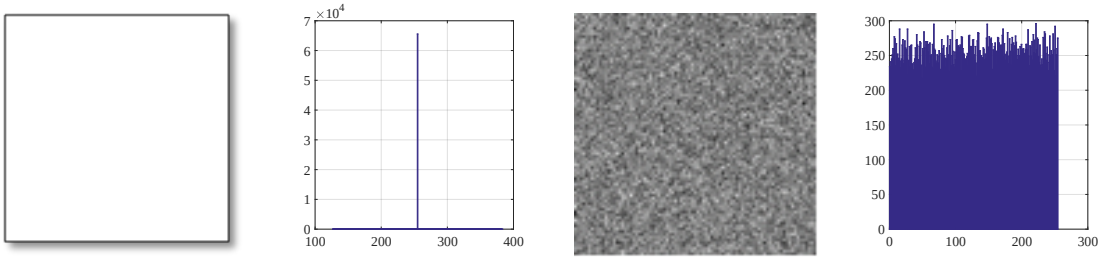
(a) (b) (c) (d)

Figure. IV.13: Histograms: (a) image Lena, (b) histogram of plain Lena, (c) encrypted Lena, and (d) histogram of encrypted Lena.



(a) (b) (c) (d)

Figure. IV.14: Histograms: (a) image Sailboat, (b) histogram of plain Sailboat, (c) encrypted Sailboat, and (d) histogram of encrypted Sailboat.



(a) (b) (c) (d)

Figure. IV.15: Histograms: (a) image White, (b) histogram of plain White, (c) encrypted White, and (d) histogram of encrypted White.

To ensure uniformity, the chi-square test is applied (using Eq. (III.66)) to statistically confirm the uniformity of a histogram. But here, N_c is the number of values (256), O_i are the observed occurrence frequencies of each color value $\in$ [0, 255] in the histogram of the encrypted image, and E_i is the expected occurrence frequency of the uniform distribution, given by $E_i = (L \times C \times P)/N_c$. The distribution of the histogram tested is uniform if it satisfies the following condition: $\chi^2_{exp} < \chi^2_{th}(N_c - 1, \alpha) = 293.2478$ $(for\ N_c = 256\ and\ \alpha = 0.05)$ [80].

In Table IV.4, we reported the experimental values of the Chi-Square test for the 10 ciphered images. The values of the experimental chi-square obtained indicate that the histograms of the ciphered images tested are uniform.

Table IV.4: Chi-square test.

Image	Size	Chi-square test
Airplane	512×512×3	256.7715
Black	256×256×1	234.0391
Bridge	512×512×1	256.6113
Cameraman	256×256×1	275.5625
Flowers	256×256×3	236.6458
Goldhill	512×512×3	239.6576
Kiel	512×512×1	243.4629
Lena	512×512×3	249.1270
Sailboat	512×512×3	242.1380
White	256×256×1	273.6641

❖ Correlation Analysis

Another key property of a secure cryptosystem is that the correlation of the encrypted

image should be close to 0 and very different from the correlation of plain-image which is generally close to 1. To measure this property, we randomly selected $N = 8000$ pairs of two adjacent pixels in the Horizontal (HC), Vertical (VC), and Diagonal (DC) directions from the original and the ciphered images. Then we calculated the correlation coefficients by using the following equation:

$$\rho_{xy} = \frac{cov(x,y)}{\sqrt{D(x)}\sqrt{D(y)}} \tag{IV.16}$$

Where:

$$cov\,(x,y) = \frac{1}{N}\sum_{i=1}^{N}\left([x_i - E(x)]\,[y_i - E(y)]\right) \tag{IV.17}$$

$$E\,(x) = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{IV.18}$$

$$D\,(x) = \frac{1}{N}\sum_{i=1}^{N}\left(x_i - E(x)\right)^2 \tag{IV.19}$$

And x and y are the gray-level values of two adjacent pixels in the image.

In Table IV.5, we give the mean correlation coefficient value based on 10 images ciphered by 100 different secret keys. From this table, the correlation coefficients in all directions of the plain images are close to one, which shows that the pixels are highly correlated, while those of encrypted images are near to zero, which proves that there is no correlation between the plain and ciphered images.

These results can be observed visually in Figure IV.16, in which we show the correlation of adjacent pixels of Goldhill ($512 \times 512 \times 3$) and encrypted Goldhill) in the three directions: horizontal, vertical, and diagonal respectively.

❖ Entropy Analysis

The random behavior of the encrypted image can also be evaluated using the entropy information defined by:

$$H\,(C) = \sum_{i=0}^{N_c-1} P\,(c_i) \times log_2 \frac{1}{P\,(c_i)} \tag{IV.20}$$

Where $H(C)$ is the entropy of the ciphered image C, $P(c_i)$ is the occurrence number

Table IV.5: The correlation coefficient of 8000 pairs of two adjacent pixels of the plain and ciphered images.

Image	Size		Plain image			Encrypted image		
			HC	VC	DC	HC	VC	DC
Airplane	512×512×3	R	0.97191	0.96051	0.93848	0.00006	-0.00153	0.00024
		G	0.95150	0.96915	0.92863	-0.00154	-0.00041	-0.00098
		B	0.96226	0.94538	0.92419	-0.00033	-0.00056	-0.00004
Black	256×256×1		-	-	-	-0.00167	-0.00044	-0.00074
Bridge	512×512×1		0.93981	0.92720	0.89716	-0.00130	0.00172	0.00084
Cameraman	256×256×1		0.92013	0.95496	0.89590	0.00110	0.00193	0.00028
Flowers	256×256×3	R	0.95049	0.96800	0.93340	0.00197	-0.00128	0.00153
		G	0.91532	0.94200	0.88863	-0.00082	-0.00019	0.00071
		B	0.92085	0.94834	0.89510	-0.00092	0.00009	0.00182
Goldhill	512×512×3	R	0.97767	0.97649	0.95975	0.00039	0.00052	-0.00058
		G	0.98180	0.98496	0.96998	-0.00047	0.00028	0.00148
		B	0.98453	0.98641	0.97339	-0.00140	0.00084	-0.00024
Kiel	512×512×1		0.90591	0.82571	0.78020	0.00168	-0.00127	-0.00044
Lena	512×512×3	R	0.97529	0.98540	0.96464	0.00055	-0.00041	0.00001
		G	0.96666	0.98011	0.95321	-0.00162	0.00049	-0.00023
		B	0.93357	0.95552	0.91819	0.00077	-0.00006	0.00042
Sailboat	512×512×3	R	0.95595	0.95302	0.94030	-0.00041	0.00070	-0.00001
		G	0.97088	0.96404	0.95027	-0.00096	0.00038	-0.00015
		B	0.97054	0.96872	0.95208	-0.00164	0.00048	-0.00018
White	256×256×1		-	-	-	-0.00165	0.00025	0.00036

of each level ($i = 0, 1, 2, ..., 255$). In case of equal probability levels ($P(c_i) = 2^{-8}$), the entropy is maximal, $H(C) = 8$.

We give in Table IV.6, the results obtained from the entropy test on the plain and encrypted images. It is clear that the obtained results of ciphered images are close to the optimal value. Then, from these results, the proposed chaos-based cryptosystem has a high degree level of resilience.

According to all these results of the histogram, chi-square, correlation, and information entropy analysis the proposed chaos-based cryptosystem is secure against statistical attacks.

IV.3.3.3 Cryptanalytic analysis

In the following part, we perform some habitual cryptanalytic analyses such as the key sensitivity and the plaintext sensitivity test.

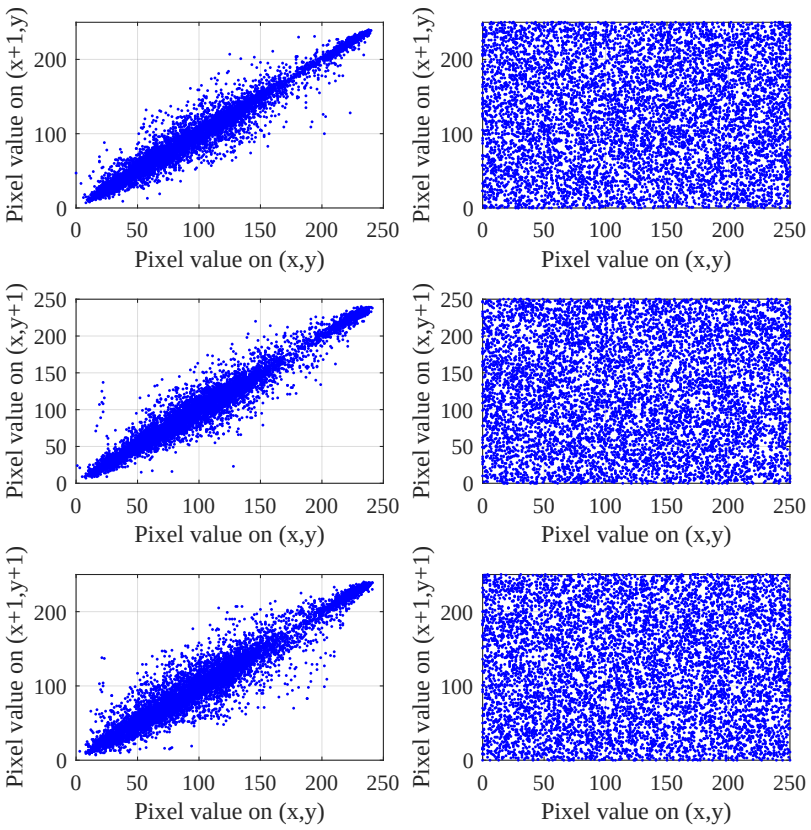


Figure. IV.16: Distribution of adjacent pixels in the plain and encrypted images of Goldhill in the three directions: horizontal, vertical, and diagonal respectively.

Table IV.6: Entropy results obtained.

Image	Size	Plain	Encrypted
Airplane	512×512×3	6.6639	7.9997
Black	256×256×1	0.0000	7.9974
Bridge	512×512×1	5.7056	7.9992
Cameraman	256×256×1	6.9046	7.9977
Flowers	256×256×3	7.5434	7.9991
Goldhill	512×512×3	7.6220	7.9997
Kiel	512×512×1	6.9589	7.9994
Lena	512×512×3	5.6822	7.9998
Sailboat	512×512×3	7.7632	7.9998
White	256×256×1	0.0000	7.9968

❖ **Key sensitivity**

A secure encryption/decryption system should be very sensitive to the secret key, i.e. even a one-bit change in the secret key should produce a completely different encrypted image. The testing scenario of key sensitivity is as follows; first, a plain-image I is encrypted using a secret key K_1 to obtain C_1, then the same plain-image I is encrypted using a secret key K_2 different from K_1 by just a one bit LSB to obtain C_2. Three parameters, namely the Number of Pixels Change Rate (NPCR), the Unified Average Changing Intensity (UACI) [133], and the Hamming Distance (see Eq. (IV.15)) are used to evaluate the key sensitivity and are defined as follows:

$$NPCR = \frac{1}{R \times C \times P} \sum_{i=1}^{R} \sum_{j=1}^{C} \sum_{p=1}^{P} D\left(i,j,p\right) \times 100\% \tag{IV.21}$$

$$D\left(i,j,p\right) = \begin{cases} 0 & if\ C_1\left(i,j,p\right) = C_2\left(i,j,p\right) \\ 1 & if\ C_1\left(i,j,p\right) \neq C_2\left(i,j,p\right) \end{cases} \tag{IV.22}$$

$$UACI = \frac{1}{R \times C \times P \times 255} \sum_{i=1}^{R} \sum_{j=1}^{C} \sum_{p=1}^{P} |C_1 - C_2| \times 100\% \tag{IV.23}$$

In the previous equations, i, j, and p are the row, column, and plane indexes of the image, respectively. R, C, and P are, respectively, the length, width, and plane sizes of the image.

This test is repeated over 100 different secret keys.

Table IV.7 presents the average results obtained of NPCR, UACI, and HD.

The resulting values obtained are near to the optimal values of NPCR, UACI, and HD which are 99.6094%, 33.4635%, and 50% respectively. Therefore, the proposed scheme is very sensitive to small changes in the secret key.

❖ **Plaintext sensitivity test**

To resist known and chosen-plaintext attacks, which are related to the differential attack, the cryptosystem must be sensitive to a one-bit change in the plaintext. The plain text sensitivity attack test case is almost identical to the key sensitivity attack.

Table IV.7: Average NPCR, UACI, and HD key sensitivity tests.

Image	Size	key sensitivity test		
		NPCR (%)	UACI (%)	HD (%)
Airplane	512×512×3	99.6065	33.4666	50.0052
Black	256×256×1	99.6099	33.4323	50.0094
Bridge	512×512×1	99.6098	33.5052	50.0019
Cameraman	256×256×1	99.5992	33.4234	49.9844
Flowers	256×256×3	99.6040	33.4772	49.9941
Goldhill	512×512×3	99.6069	33.4464	49.9942
Kiel	512×512×1	99.6042	33.4650	49.9976
Lena	512×512×3	99.6088	33.4664	49.9907
Sailboat	512×512×3	99.6077	33.4680	50.0090
White	256×256×1	99.6231	33.5524	50.0000

Indeed, after having encrypted a plain image, we choose 21-pixel positions [91] and we change, in turn, their LSB bit, then we encrypt them and calculate the 21 HDs.

In Fig. IV.17, we show for each position the HDs of the 10 test images (red dots) as well as their average values (green line). As we can see, for each position, the average HD is close to the optimal value of 50 %. Furthermore, the maximum value of the HD is equal to 50.1602 % and the minimum value is equal to 49.8039 % which is close to the optimal value.

Moreover, in Table IV.8, we give the average NPCR, UACI, and HD on 21 positions for each image. From these results, the average values of NPCR, UACI, and HD obtained are almost equal to the optimal value. These values indicate that the proposed chaos-based cryptosystem is very sensitive to slight modifications of the plaintext.

IV.3.3.4 Robustness against noise and occlusion

Encrypted images transmitted may be affected by channel noise and data loss. In this paragraph we study the resistance capacity of the proposed cryptosystem against noise and occlusion. We utilize salt and pepper noise and occlusion attack [90].

We added salt and pepper noise with different intensities (0.01, 0.02, 0.05, and 0.1) to the encrypted Cameraman image (see Fig. IV.9.(c)), as shown in Figs. IV.18 (a), (b), (c), (d) respectively, and then we decrypted them. The recovered images are shown in Figs. IV.18 (e), (f), (g), (h) respectively. It can be seen that under limited noise, the

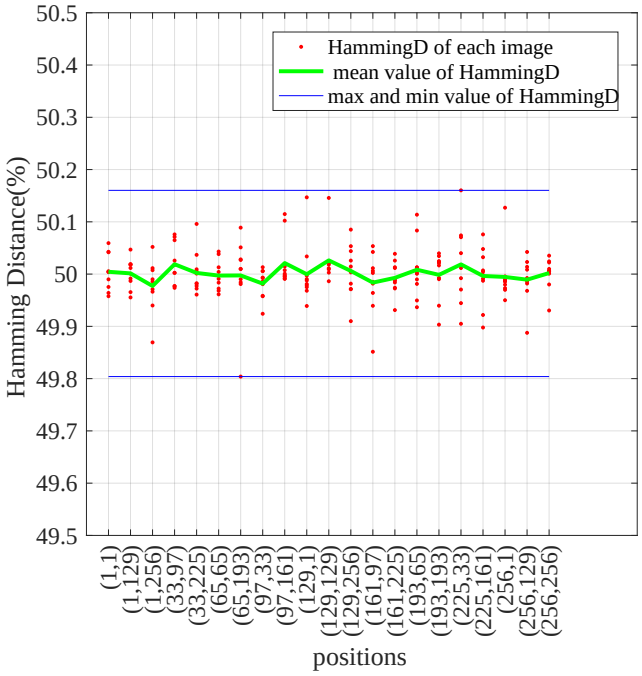


Figure. IV.17: Plain-text sensitivity test evaluated by the Hamming Distance.

Table IV.8: NPCR, UACI, and HD of the plaintext sensitivity test.

Image	Size	Plaintext sensitivity		
		NPCR (%)	UACI (%)	HD (%)
Airplane	512×512×3	99.6102	33.4659	49.9972
Black	256×256×1	99.6067	33.4840	50.0143
Bridge	512×512×1	99.6114	33.4960	50.0060
Cameraman	256×256×1	99.6068	33.4616	49.9907
Flowers	256×256×3	99.6016	33.4563	49.9954
Goldhill	512×512×3	99.6081	33.4460	50.0003
Kiel	512×512×1	99.6075	33.4658	49.9974
Lena	512×512×3	99.6092	33.4669	50.0028
Sailboat	512×512×3	99.6058	33.4837	50.0062
White	256×256×1	99.6008	33.5139	49.9981

decrypted images are always identified. Thus, our proposed decryption system has good robustness against noise attacks.

Another kind of attack is occlusion. To assess the robustness of the cryptosystem against such an attack, first, we occulted the Cameraman encrypted image (see Fig. IV.9.(c)) with different data loss sizes and positions: 1/16, 1/8, 1/4, and 1/2, as shown

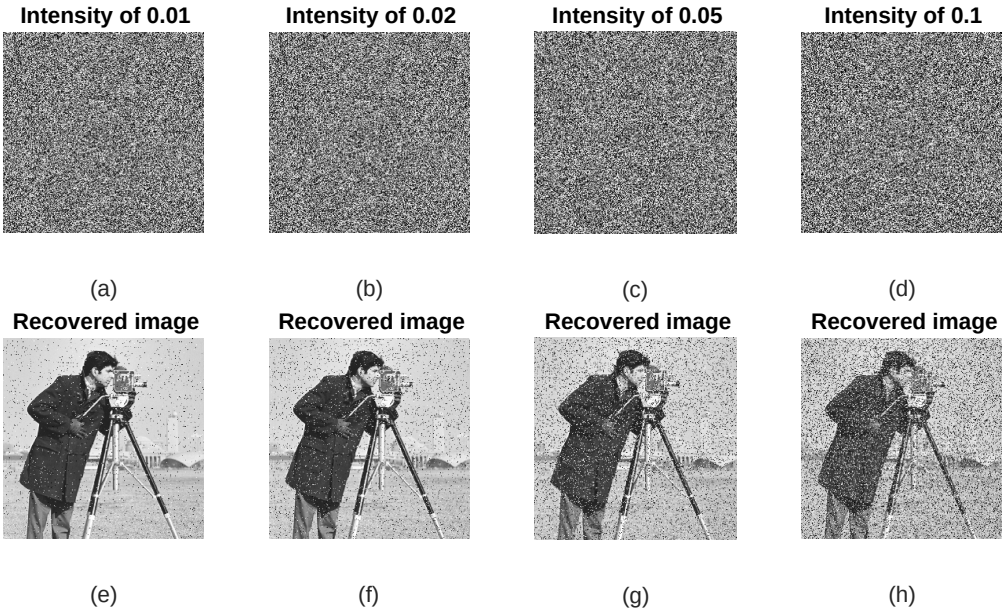


Figure. IV.18: Robustness against salt and pepper noise.

in Figs. IV.19 (a), (b), (c) and (d) respectively, then, we decrypted and shown them in Figs. IV.19 (e), (f), (g) and (h). We can observe that all the recovered images are recognized but their qualities decrease with the increase of the occlusion size. Therefore, the proposed cryptosystem has a high level of robustness against occlusion attacks.

IV.3.3.5 The speed performance of the proposed chaos-based cryptosystem

Computing performance is an important issue for practical applications and largely depends on the programming languages used. Matlab language is not a good candidate to achieve high computing performance compared to C language, VHDL description language, etc. However, with Matlab, we can design and realize cryptosystems faster than the other languages mentioned.

We evaluated the computational performance: average encryption time, average Encryption Throughput (ET), and average Number of needed Cycles per Bytes (NCpB) of the proposed cryptosystem and we compared its performance (NCpB) with other cryptosystems designed in Matlab language.

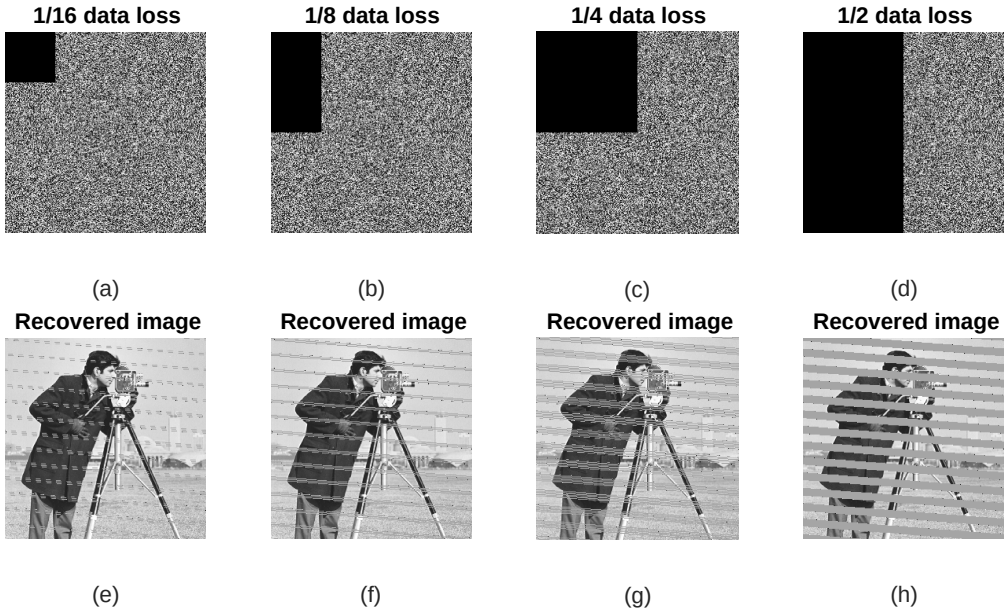


Figure. IV.19: Robustness against occlusion attack.

The ET and NCpB are given by the following Equation:

$$ET = \frac{Image\ Size\ (Bytes)}{Average\ Encryption\ Time\ (second)} \tag{IV.24}$$

$$NCpB = \frac{CPU\ Speed\ (Hertz)}{ET\ (Bytes/s)} \tag{IV.25}$$

In Table IV.9, we give the computing performance obtained by the proposed encryption system using 100 different secret keys, for four plain images.

Table IV.9: Computing performance of the proposed encryption system.

Image	Size	Encryption Time (milli-second)	ET (Mbps)	NCpB (Cycles/Byte)
Cameraman	256×256×1	39.3	1.589	1501
Flowers	256×256×3	119.2	1.5726	1516
Kiel	512×512×1	163.9	1.5249	1564
Lena	512×512×3	602.3	1.2451	1915

In Table IV.10, we compare the NCpB of our algorithm for the Lena image of size $512 \times 512 \times 1$ with the NCpBs of other chaos-based encryption algorithms.

Table IV.10: Comparison of NCpBs from different encryption schemes.

Cryptosystem	NCpB
Proposed	1568
Qiao et al. [90]	77386
Luo et al. [147]	95368
Huang et al. [148]	15642

As Table IV.10 shows, the proposed encryption system has higher computational performance efficiency than the others.

IV.4 Conclusion

We proposed a new secure chaos-based cryptosystem system for block cipher in CBC mode and evaluated its performance. The proposed cryptosystem achieves high confusion diffusion effects thanks to a robust pseudo-random number generator of chaotic sequences, strong circular substitution based on the proposed S-box, and a solid diffusion layer. The proposed LSPT-PRNG is formed by four discrete 1-D chaotic maps weakly coupled by a predefined coupling matrix M, to avoid the divide-and-conquer attacks, and to increase the randomness of the sequences produced as well as their lengths. The diffusion layer is performed by a 2-D modified cat map and a Horizontal Addition Diffusion (HAD) followed by a Vertical Addition Diffusion (VAD). The experimental results and the security analysis demonstrate that the proposed cryptosystem can successfully resist various attacks known in literature, such as statistical attacks, brute force attacks, noise attacks, and occlusion attacks, and therefore can be used to secure sensitive data. In a future study, we plan to address the hardware implementation of the system.

General Conclusion and Perspectives

Being at the heart of the digital world, data security is one of the most studied topics, especially with the massive use of the Internet of Things. Indeed, it involves the use of embedded devices, small sensors, and tiny devices that communicate with each other. These devices suffer from many constraints such as limited memory, energy, lifetime and processing power. In addition, a fraction of the total hardware of these devices is dedicated to security. These requirements lead to the need for specific security primitives for pervasive devices. Therefore, the use of the chaos-based cryptography as a promising solution to meet these requirements. This is the subject of this thesis.

In this manuscript, three contributions have been made related to our area of interest. They offer a better level of security and good efficiency. So, their interest in being used to secure the data transmitted and stored. They offer an optimized security/cost/performance compromise.

In Chapter II, we evaluate the statistical security and hardware metrics of some chaotic maps, which are the basic components of the proposed chaotic generators. We first presented the common and standard security tools for measuring the statistical security performance of the studied chaotic maps. Then, we analyze the hardware metrics performance of the implementation of these chaotic maps in terms of resources used, speed, and power consumption.

In Chapter III, we studied and implemented on a Xilinx PYNQ-Z2 FPGA hardware platform using VHDL a novel chaos-based stream ciphers using a proposed secure pseudo-random number generators of chaotic sequences. The proposed chaotic systems use a weekly coupling matrix which prevents divide-and-conquer attacks on the initial vector (IV). One of the proposed systems, the LST_RC-PRNG system, includes countermeasures against side channel attacks (SCA). Next, we analyzed the cryptographic properties of the proposed PRNGs-CS and evaluated the performance of their hardware metrics. The results obtained demonstrate on the one hand, the high degree of security and on the other hand, the good hardware metrics achieved by the proposed PRNG-CS. After that, we realized the corresponding chaos-based stream ciphers and asserted their resilience against cryptanalytic attacks. Further, we evaluated their hardware metrics. All the results obtained indicate that the proposed chaos-based stream ciphers are a good candidates for encrypting private data.

In Chapter IV, We proposed a new secure chaos-based cryptosystem system for block cipher in CBC mode and evaluated its performance. The proposed cryptosystem achieves high confusion diffusion effects thanks to a robust pseudo-random number generator of chaotic sequences, strong circular substitution based on the proposed S-box, and a solid diffusion layer. The diffusion layer is performed by a 2-D modified cat map and a Horizontal Addition Diffusion (HAD) followed by a Vertical Addition Diffusion (VAD). The experimental results and the security analysis demonstrate that the proposed cryptosystem can successfully resist various attacks known in literature, such as statistical attacks, brute force attacks, noise attacks, and occlusion attacks, and therefore can be used to secure sensitive data.

For future works, we plan to design authenticated encryption and address the problem of secure hardware implementation against physical attacks.

141

Personal Publications

Published Journal papers

Fethi Dridi, S. El Assad, W. El Hadj Youssef, M. Machhout, A. E. Samhat, Design, fpga-based implementation and performance of a pseudo random number generator of chaotic sequences, Advances in Electrical and Computer Engineering 21 (2) (2021) 41–48. doi:10.4316/AECE.2021.02005.

Fethi Dridi, S. El Assad, W. El Hadj Youssef, M. Machhout, R. Lozi, The design and fpga-based implementation of a stream cipher based on a secure chaotic generator, Applied Sciences 11 (2) (2021) 625. doi:10.3390/app11020625.

Fethi Dridi, C. Atamech, S. El Assad, W. E. Youssef, M. Machhout, Fpga implementation of a chaos-based stream cipher and evaluation of its performances, International Journal of Chaotic Computing 7 (1) (2020) 179–186. doi:10.20533/ijcc.2046.3359.2020.0023.

M. Madani, S. El Assad, **Fethi Dridi**, R. Lozi, Enhanced design and hardware implementation of a chaos-based block cipher for image protection, Journal of Difference Equations and Applications (2022) 1–21doi:10.1080/10236198.2022.2069496.

W. El Hadj Youssef, A. Abdelli, **Fethi Dridi**, M. Machhout, Hardware implementation of secure lightweight cryptographic designs for iot applications, Security and Communication Networks 2020. doi:10.1155/2020/8860598.

W. El Hadj Youssef, A. Abdelli, **Fethi Dridi**, R. Brahim, M. Machhout, An efficient lightweight cryptographic instructions set extensions for iot device security, Security and Communication Networks (2021). doi:10.1155/2022/9709601.

Published Conference papers

Fethi Dridi, S. El Assad, C. Atamech, W. E. Youssef, M. Machhout, Design and implementation on fpga board of a chaos-based stream cipher, in: 2020 15th International Conference for Internet Technology and Secured Transactions (ICITST), IEEE, 2020, pp. 1–5. doi:10.23919/ICITST51030.2020.9351328.

Fethi Dridi, S. El Assad, W. E. Youssef, M. Machhout, Fpga implementation of a pseudo-chaotic number generator and evaluation of its performance, in: 2019 International Conference on Internet of Things, Embedded Systems and Communications (IINTEC), IEEE, 2019, pp. 231–234. doi:10.1109/IINTEC48298.2019.9112124.

Submitted Journal and Conference papers

Fethi Dridi, S. El Assad, W. El Hadj Youssef, M. Machhout, Design, implementation, and analysis of a bloc cipher based on a secure chaotic generator, to be soumitted soon, 19 Pages.

Bibliography

[1] L. Irwin, "List of data breaches and cyber attacks in august 2019 – 114.6 million records leaked," *available at https://www.itgovernance.co.uk/blog/list-of-data-breaches-and-cyber-attacks-in-august-2019-114-6-million-records-leaked*, 2019, [Online; accessed 2019]. vii, 2

[2] C. Osborne, "Whatsapp vulnerability exploited through malicious gifs to hijack chat sessions," *available at https://www.zdnet.com/article/whatsapp-vulnerability-exploited-through-malicious-gifs-to-hijack-chat-sessions/*, 2019, [Online; accessed 2019]. vii, 2

[3] L. MATSAKIS and I. LAPOWSKY, "Everything we know about facebook's massive security breach," *available at https://www.wired.com/story/facebook-security-breach-50-million-accounts/*, 2018, [Online; accessed 2018]. vii, 2

[4] B. Yirka, "Two major security vulnerabilities found in pdf files," *available at https://techxplore.com/news/2019-10-major-vulnerabilities-pdf.html*, 2019, [Online; accessed 2019]. vii, 2

[5] D. Kahn, *The Codebreakers: The comprehensive history of secret communication from ancient times to the internet.* Simon and Schuster, 1996. 8

[6] L. Kruh, "Codes, ciphers & other cryptic & clandestine communication: Making and breaking secret messages from hieroglyphs to the internet," *Cryptologia*, vol. 24, no. 1, p. 71, 2000. 8

[7] S. Vaudenay, *A classical introduction to cryptography: Applications for communications security*. Springer Science & Business Media, 2005. 8

[8] V. Kleist, "The code book: the science of secrecy from ancient egypt to quantum cryptography [book review]," *IEEE Annals of the History of Computing*, vol. 24, no. 2, pp. 97–98, 2002. 8

[9] J. Katz and Y. Lindell, *Introduction to modern cryptography*. CRC press, 2020. 8

[10] A. Scherbius, "Ciphering machine," *United States Patent 1,657,411*, Jan. 24 1928. 8

[11] A. Kerckhoffs, "La cryptographic militaire," *Journal des sciences militaires*, pp. 5–38, 1883. 9

[12] C. E. Shannon, "Communication theory of secrecy systems," *The Bell system technical journal*, vol. 28, no. 4, pp. 656–715, 1949. 9, 11

[13] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of applied cryptography*. CRC press, 1996. 10

[14] E. Fujisaki and T. Okamoto, "Secure integration of asymmetric and symmetric encryption schemes," in *Annual international cryptology conference*. Springer, 1999, pp. 537–554. 11

[15] Y. Kumar, R. Munjal, and H. Sharma, "Comparison of symmetric and asymmetric cryptography with existing vulnerabilities and countermeasures," *International Journal of Computer Science and Management Studies*, vol. 11, no. 03, pp. 60–63, 2011. 11

[16] G. J. Simmons, "Symmetric and asymmetric encryption," *ACM Computing Surveys (CSUR)*, vol. 11, no. 4, pp. 305–330, 1979. 11

[17] K. M. Martin, "Everyday cryptography," *The Australian Mathematical Society*, vol. 231, no. 6, 2012. 11

[18] S. Charlwood and P. James-Roxby, "Evaluation of the xc6200-series architecture for cryptographic applications," in *International Workshop on Field Programmable Logic and Applications*. Springer, 1998, pp. 218–227. 11

[19] T. Hardjono and L. R. Dondeti, *Security in Wireless LANS and MANS (Artech House Computer Security)*. Artech House, Inc., 2005. 11

[20] W. Stallings, *Cryptography and network security, 4/E*. Pearson Education India, 2006. 11

[21] eSTREAM, "eSTREAM: the ECRYPT Stream Cipher Project," https://www.ecrypt.eu.org/stream/, 2012, [Online; accessed January-2019]. 12

[22] M. Robshaw, "The estream project," in *New Stream Cipher Designs*. Springer, 2008, pp. 1–6. 12

[23] M. Hell and T. Johansson, "Breaking the f-fcsr-h stream cipher in real time," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2008, pp. 557–569. 12

[24] E. U. T. R. A. Network, "3rd generation partnership project; technical specification group services and system aspects; general packet radio service (gprs) enhancements for evolved universal terrestrial radio access network (e-utran) access," *EUTRA Network*, 2011. 12

[25] P. Kitsos, N. Sklavos, G. Provelengios, and A. N. Skodras, "Fpga-based performance analysis of stream ciphers zuc, snow3g, grain v1, mickey v2, trivium and e0," *Microprocessors and Microsystems*, vol. 37, no. 2, pp. 235–245, 2013. 12

[26] "Etsi/sage specification. specification of the 3gpp confidentiality and integrity algorithms 128-eea3 128-eia3. document 2: Zuc specification, version: 1.5," Date: 4th January 2011. 13

[27] C. Manifavas, G. Hatzivasilis, K. Fysarakis, and Y. Papaefstathiou, "A survey of lightweight stream ciphers for embedded systems," *Security and Communication Networks*, vol. 9, no. 10, pp. 1226–1246, 2016. 13

[28] J. Machicao, A. G. Marco, and O. M. Bruno, "Chaotic encryption method based on life-like cellular automata," *Expert Systems with Applications*, vol. 39, no. 16, pp. 12 626–12 635, 2012. 13

[29] T. W. Cusick and P. Stanica, *Cryptographic Boolean functions and applications*. Academic Press, 2017. 13, 14

[30] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, "Chapter 7: Block ciphers," *Handbook of Applied Cryptography*, 1996. 13

[31] M. Bellare and P. Rogaway, "Introduction to modern cryptography," *Ucsd Cse*, vol. 207, p. 207, 2005. 13, 15

[32] F. Pub, "Data encryption standard (des)," *FIPS PUB*, pp. 46–3, 1999. 14

[33] E. Biham and A. Shamir, "Differential cryptanalysis of des-like cryptosystems," *Journal of CRYPTOLOGY*, vol. 4, no. 1, pp. 3–72, 1991. 14

[34] E. F. Foundation, "Cracking des: Secrets of encryption research, wiretap politics and chip design," 1998. 14

[35] S. Kumar, C. Paar, J. Pelzl, G. Pfeiffer, and M. Schimmler, "Breaking ciphers with copacobana–a cost-optimized parallel code breaker," in *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer, 2006, pp. 101–118. 14

[36] R. C. Merkle and M. E. Hellman, "On the security of multiple encryption," *Communications of the ACM*, vol. 24, no. 7, pp. 465–467, 1981. 14

[37] J. Daemen and V. Rijmen, "The design of rijndael: Aes-the advanced encryption standard springer science & business media," 2013. 14

[38] P. FIPS, "197, advanced encryption standard (aes), 2001. us department of commerce/national institute of standards and technology. true random number generator," *Constructive Side-Channel Analysis and Secure Design*, vol. 7275, pp. 151–166. 14

[39] L. Keliher, H. Meijer, and S. Tavares, "Modeling linear characteristics of substitution-permutation networks," in *International Workshop on Selected Areas in Cryptography*. Springer, 1999, pp. 78–91. 14

[40] A. Biryukov, G. Gong, and D. R. Stinson, *Selected Areas in Cryptography: 17th International Workshop, SAC 2010, Waterloo, Ontario, Canada, August 12-13, 2010, Revised Selected Papers*. Springer, 2011, vol. 6544. 14

[41] C. E. Shannon, "A mathematical theory of cryptography," *Mathematical Theory of Cryptography*, 1945. 14

[42] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of applied cryptography*. CRC press, 2018. 14, 15

[43] ——, "Applied cryptography," *CRC, Boca Raton*, 1996. 15

[44] M. Dworkin *et al.*, "Recommendation for block cipher modes of operation: methods for format-preserving encryption," *NIST Special Publication*, vol. 800, p. 38G, 2016. 15

[45] E. N. Lorenz and K. Haman, "The essence of chaos," *Pure and Applied Geophysics*, vol. 147, no. 3, pp. 598–599, 1996. 15

[46] R. Matthews, "On the derivation of a "chaotic" encryption algorithm," *Cryptologia*, vol. 13, no. 1, pp. 29–41, 1989. 15

[47] X.-Y. Wang, J.-J. Zhang, F.-C. Zhang, and G.-H. Cao, "New chaotical image encryption algorithm based on fisher–yatess scrambling and dna coding," *Chinese Physics B*, vol. 28, no. 4, p. 040504, 2019. 16

[48] A. Belazi, A. A. Abd El-Latif, and S. Belghith, "A novel image encryption scheme based on substitution-permutation network and chaos," *Signal Processing*, vol. 128, pp. 155–170, 2016. 16, 116

[49] J. Amigo, L. Kocarev, and J. Szczepanski, "Theory and practice of chaotic cryptography," *Physics Letters A*, vol. 366, no. 3, pp. 211–216, 2007. 16

[50] L. Kocarev, "Chaos-based cryptography: a brief overview," *IEEE Circuits and Systems Magazine*, vol. 1, no. 3, pp. 6–21, 2001. 16

[51] O. Datcu, C. Macovei, and R. Hobincu, "Chaos based cryptographic pseudo-random number generator template with dynamic state change," *Applied Sciences*, vol. 10, no. 2, p. 451, 2020. 16

[52] L. Acho, "A chaotic secure communication system design based on iterative learning control theory," *Applied Sciences*, vol. 6, no. 10, p. 311, 2016. 16

[53] N. Abdoun, S. El Assad, T. Manh Hoang, O. Deforges, R. Assaf, and M. Khalil, "Designing two secure keyed hash functions based on sponge construction and the chaotic neural network," *Entropy*, vol. 22, no. 9, p. 1012, 2020. 16

[54] D. Battikh, S. El Assad, T. M. Hoang, B. Bakhache, O. Deforges, and M. Khalil, "Comparative study of three steganographic methods using a chaotic system and their universal steganalysis based on three feature vectors," *Entropy*, vol. 21, no. 8, p. 748, 2019. 16

[55] T.-L. Liao, P.-Y. Wan, and J.-J. Yan, "Design of synchronized large-scale chaos random number generators and its application to secure communication," *Applied Sciences*, vol. 9, no. 1, p. 185, 2019. 16

[56] N. K. Pareek, V. Patidar, and K. K. Sud, "Image encryption using chaotic logistic map," *Image and vision computing*, vol. 24, no. 9, pp. 926–934, 2006. 16

[57] L. Kocarev and G. Jakimoski, "Logistic map as a block encryption algorithm," *Physics Letters A*, vol. 289, no. 4-5, pp. 199–206, 2001. 16

[58] M. François, T. Grosges, D. Barchiesi, and R. Erra, "Pseudo-random number generator based on mixing of three chaotic maps," *Communications in Nonlinear Science and Numerical Simulation*, vol. 19, no. 4, pp. 887–895, 2014. 16

[59] X.-y. Wang and X. Qin, "A new pseudo-random number generator based on cml and chaotic iteration," *Nonlinear Dynamics*, vol. 70, no. 2, pp. 1589–1592, 2012. 16

[60] M. A. Taha, S. E. Assad, A. Queudet, and O. Deforges, "Design and efficient implementation of a chaos-based stream cipher," *International Journal of Internet Technology and Secured Transactions*, vol. 7, no. 2, pp. 89–114, 2017. 16, 63, 77

[61] O. Jallouli, S. El Assad, M. Chetto, and R. Lozi, "Design and analysis of two stream ciphers based on chaotic coupling and multiplexing techniques," *Multimedia tools and applications*, vol. 77, no. 11, pp. 13 391–13 417, 2018. 17, 21, 63, 77

[62] R. Lozi, "Emergence of randomness from chaos," *International Journal of Bifurcation and Chaos*, vol. 22, no. 02, p. 1250021, 2012. 17, 77, 89, 113

[63] L. Ding, C. Liu, Y. Zhang, and Q. Ding, "A new lightweight stream cipher based on chaos," *Symmetry*, vol. 11, no. 7, p. 853, 2019. 17

[64] R. I. Abdelfatah, M. E. Nasr, and M. A. Alsharqawy, "Encryption for multimedia based on chaotic map: Several scenarios," *Multimedia Tools and Applications*, vol. 79, no. 27, pp. 19 717–19 738, 2020. 17

[65] G. Gautier, M. Le Glatin, S. El Assad, W. Hamidouche, O. Déforges, S. Guilley, and A. Facon, "Hardware implementation of lightweight chaos-based stream cipher," in *International Conference on Cyber-Technologies and Cyber-Systems*, 2019, pp. 5–pages. 17, 97

[66] C. Tanougast, "Hardware implementation of chaos based cipher: Design of embedded systems for security applications," in *Chaos-Based Cryptography*. Springer, 2011, pp. 297–330. 17, 97

[67] I. Koyuncu, M. Tuna, I. Pehlivan, C. B. Fidan, and M. Alçın, "Design, fpga implementation and statistical analysis of chaos-ring based dual entropy core true random number generator," *Analog Integrated Circuits and Signal Processing*, vol. 102, no. 2, pp. 445–456, 2020. 17, 97

[68] Y. Liu, X. Tong, and S. Hu, "A family of new complex number chaotic maps based image encryption algorithm," *Signal Processing: Image Communication*, vol. 28, no. 10, pp. 1548–1559, 2013. 17, 112

[69] X. Zhang, L. Shao, Z. Zhao, and Z. Liang, "An image encryption scheme based on constructing large permutation with chaotic sequence," *Computers & Electrical Engineering*, vol. 40, no. 3, pp. 931–941, 2014. 17, 112

[70] J. A. E. Fouda, J. Y. Effa, S. L. Sabat, and M. Ali, "A fast chaotic block cipher for image encryption," *Communications in Nonlinear Science and Numerical Simulation*, vol. 19, no. 3, pp. 578–588, 2014. 17, 112

[71] J. S. Khan and S. K. Kayhan, "Chaos and compressive sensing based novel image encryption scheme," *Journal of Information Security and Applications*, vol. 58, p. 102711, 2021. 17

[72] L. Liu, D. Jiang, X. Wang, X. Rong, and R. Zhang, "2d logistic-adjusted-chebyshev map for visual color image encryption," *Journal of Information Security and Applications*, vol. 60, p. 102854, 2021. 17

[73] D. Caragata, S. El Assad, and M. Luduena, "An improved fragile watermarking algorithm for jpeg images," *AEU-International Journal of Electronics and Communications*, vol. 69, no. 12, pp. 1783–1794, 2015. 17, 112

[74] F. Pichler and J. Scharinger, "Finite dimensional generalized baker dynamical systems for cryptographic applications," in *International Conference on Computer Aided Systems Theory*. Springer, 1995, pp. 465–476. 17, 112

[75] N. Masuda and K. Aihara, "Cryptosystems with discretized chaotic maps," *Ieee transactions on circuits and systems i: fundamental theory and applications*, vol. 49, no. 1, pp. 28–40, 2002. 17, 112

[76] C. E. Shannon, "Communication theory of secrecy systems," *The Bell system technical journal*, vol. 28, no. 4, pp. 656–715, 1949. 17, 112

[77] J. Fridrich, "Symmetric ciphers based on two-dimensional chaotic maps," *International Journal of Bifurcation and chaos*, vol. 8, no. 06, pp. 1259–1284, 1998. 17, 18, 112

[78] N. K. Pareek, V. Patidar, and K. K. Sud, "Image encryption using chaotic logistic map," *Image and vision computing*, vol. 24, no. 9, pp. 926–934, 2006. 17, 112

[79] Z.-l. Zhu, W. Zhang, K.-w. Wong, and H. Yu, "A chaos-based symmetric image encryption scheme using a bit-level permutation," *Information Sciences*, vol. 181, no. 6, pp. 1171–1186, 2011. 17, 112

[80] S. El Assad and M. Farajallah, "A new chaos-based image encryption system," *Signal Processing: Image Communication*, vol. 41, pp. 144–157, 2016. 17, 63, 112, 130

[81] S. El Assad, M. Farajallah, and C. Vladeanu, "Chaos-based block ciphers: An overview," in *2014 10th International Conference on Communications (COMM)*. IEEE, 2014, pp. 1–4. 17, 63, 112

[82] G. Alvarez and S. Li, "Some basic cryptographic requirements for chaos-based cryptosystems," *International journal of bifurcation and chaos*, vol. 16, no. 08, pp. 2129–2151, 2006. 17, 18, 112

[83] S. Lian, J. Sun, and Z. Wang, "Security analysis of a chaos-based image encryption algorithm," *Physica A: Statistical Mechanics and its Applications*, vol. 351, no. 2-4, pp. 645–661, 2005. 17, 112

[84] J. Daemen and V. Rijmen, *The design of Rijndael*. Springer, 2002, vol. 2. 18, 112

[85] R. Nguyen, "Penetration testing on a c-software implementation a-1709rns006-c," *Internal Report*, 2018. 18

[86] R. Nguyen, A. Facon, S. Guilley, G. Gautier, and S. El Assad, "Speed-up of sca attacks on 32-bit multiplications," in *International Conference on Codes, Cryptology, and Information Security*. Springer, 2019, pp. 31–39. 18

[87] J.-x. Chen, Z.-l. Zhu, C. Fu, H. Yu, and L.-b. Zhang, "A fast chaos-based image encryption scheme with a dynamic state variables selection mechanism," *Communications in Nonlinear Science and Numerical Simulation*, vol. 20, no. 3, pp. 846–860, 2015. 18

[88] M. Farajallah, S. El Assad, and O. Deforges, "Fast and secure chaos-based cryptosystem for images," *International Journal of Bifurcation and Chaos*, vol. 26, no. 02, p. 1650021, 2016. 18, 63, 119, 120

[89] W. Zhang, K.-w. Wong, H. Yu, and Z.-l. Zhu, "An image encryption scheme using reverse 2-dimensional chaotic map and dependent diffusion," *Communications in Nonlinear Science and Numerical Simulation*, vol. 18, no. 8, pp. 2066–2080, 2013. 18

[90] Z. Qiao, S. El Assad, and I. Taralova, "Design of secure cryptosystem based on chaotic components and aes s-box," *AEU-International Journal of Electronics and Communications*, vol. 121, p. 153205, 2020. 18, 135, 139

[91] T. Omrani, R. Rhouma, and R. Becheikh, "Licid: a lightweight image cryptosystem for iot devices," *Cryptologia*, vol. 43, no. 4, pp. 313–343, 2019. 18, 120, 135

[92] Y. Wang, K.-W. Wong, X. Liao, and T. Xiang, "A block cipher with dynamic s-boxes based on tent map," *Communications in Nonlinear Science and Numerical Simulation*, vol. 14, no. 7, pp. 3089–3099, 2009. 18

[93] X. Zhang, Z. Zhao, and J. Wang, "Chaotic image encryption based on circular substitution box and key stream buffer," *Signal Processing: Image Communication*, vol. 29, no. 8, pp. 902–913, 2014. 18, 118

[94] R. Lozi, "Giga-periodic orbits for weakly coupled tent and logistic discretized maps," *arXiv preprint arXiv:0706.0254*, 2007. 21

[95] ——, "New enhanced chaotic number generators," *Indian Journal of Industrial and Applied Mathematics*, vol. 1, no. 1, pp. 1–23, 2008. 21

[96] G. Zheng, D. Boutat, T. Floquet, and J.-P. Barbot, "Secure communication based on multi-input multi-output chaotic system with large message amplitude," *Chaos, Solitons & Fractals*, vol. 41, no. 3, pp. 1510–1517, 2009. 21

[97] S. Phatak and S. S. Rao, "Logistic map: A possible random-number generator," *Physical review E*, vol. 51, no. 4, p. 3670, 1995. 21

[98] L. Kocarev and G. Jakimoski, "Pseudorandom bits generated by chaotic maps," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 50, no. 1, pp. 123–126, 2003. 21

[99] L. Shujun, M. Xuanqin, and C. Yuanlong, "Pseudo-random bit generator based on couple chaotic systems and its applications in stream-cipher cryptography," in *International conference on cryptology in India*. Springer, 2001, pp. 316–329. 21, 29

[100] S. El Assad, H. Noura, and I. Taralova, "Design and analyses of efficient chaotic generators for crypto-systems," in *Advances in Electrical and Electronics Engineering-IAENG Special Edition of the World Congress on Engineering and Computer Science 2008*. IEEE, 2008, pp. 3–12. 21

[101] G. Jakimoski and L. Kocarev, "Chaos and cryptography: block encryption ciphers based on chaotic maps," *Ieee transactions on circuits and systems i: fundamental theory and applications*, vol. 48, no. 2, pp. 163–169, 2001. 21

[102] D. Xiao, X. Liao, and S. Deng, "Parallel keyed hash function construction based on chaotic maps," *Physics Letters A*, vol. 372, no. 26, pp. 4682–4688, 2008. 21

[103] X. Wu and Z.-H. Guan, "A novel digital watermark algorithm based on chaotic maps," *Physics Letters A*, vol. 365, no. 5-6, pp. 403–406, 2007. 21

[104] A. Mooney, "Chaos based digital watermarking," in *Intelligent Computing Based on Chaos*. Springer, 2009, pp. 315–332. 21

[105] 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 and hamilton inc mclean va, Tech. Rep., 2001. 22

[106] J. Walker, "Ent: a pseudorandom number sequence test program," *Software and documentation available at http://www.fourmilab.ch/random/*, 2008, [Online; accessed 27-November-2018]. 22

[107] 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 and hamilton inc mclean va, Tech. Rep., 2001. 23

[108] N. K. Pareek, V. Patidar, and K. K. Sud, "Image encryption using chaotic logistic map," *Image and vision computing*, vol. 24, no. 9, pp. 926–934, 2006. 29

[109] J. Fridrich, "Symmetric ciphers based on two-dimensional chaotic maps," *International Journal of Bifurcation and chaos*, vol. 8, no. 06, pp. 1259–1284, 1998. 29

[110] R. Lozi, "Un attracteur étrange (?) du type attracteur de hénon," *Le Journal de Physique Colloques*, vol. 39, no. C5, pp. C5–9, 1978. 29

[111] R. Lozi and E. Cherrier, "Noise-resisting ciphering based on a chaotic multi-stream pseudo-random number generator," in *2011 International Conference for Internet Technology and Secured Transactions*. IEEE, 2011, pp. 91–96. 29

[112] X. Huang, "Image encryption algorithm using chaotic chebyshev generator," *Nonlinear Dynamics*, vol. 67, no. 4, pp. 2411–2417, 2012. 29

[113] M. CRAMPIN and B. Heal, "On the chaotic behaviour of the tent map," *Teaching Mathematics and its Applications: An International Journal of the IMA*, vol. 13, no. 2, pp. 83–89, 1994. 30

[114] N. Masuda, G. Jakimoski, K. Aihara, and L. Kocarev, "Chaotic block ciphers: from theory to practical algorithms," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 53, no. 6, pp. 1341–1352, 2006. 30

[115] S. Lian, J. Sun, J. Wang, and Z. Wang, "A chaotic stream cipher and the usage in video protection," *Chaos, Solitons & Fractals*, vol. 34, no. 3, pp. 851–859, 2007. 37

[116] P.-F. Verhulst, "Recherches mathématiques sur la loi d'accroissement de la population," *Journal des économistes*, vol. 12, p. 276, 1845. 41

[117] S. M. Ulam, "On combination of stochastic and deterministic processes," *Bull. Amer. Math. Soc.*, vol. 53, p. 1120, 1947. 41

[118] S. El Assad and R. Lozi, "Chaos-based cryptography," *internal report*, 2019. 42, 47

[119] J. Peng, M. You, Z. Yang, and S. Jin, "Research on a block encryption cipher based on chaotic dynamical system," in *Third International Conference on Natural Computation (ICNC 2007)*, vol. 5. IEEE, 2007, pp. 744–748. 42

[120] B. Schneier, *Applied cryptography: protocols, algorithms, and source code in C*. john wiley & sons, 2007. 66, 116

[121] R. Nguyen, "Penetration testing on a c-software implementation aff1709rns006-c," *internal report*, lot C (September 2018). 77, 112

[122] R. Nguyen, A. Facon, S. Guilley, G. Gautier, and S. El Assad, "Speed-up of sca attacks on 32-bit multiplications," in *International Conference on Codes, Cryptology, and Information Security*. Springer, 2019, pp. 31–39. 77, 112

[123] S. Vigna, "Further scramblings of marsaglia's xorshift generators," *Journal of Computational and Applied Mathematics*, vol. 315, pp. 175–181, 2017. 79, 82

[124] D. Blackman and S. Vigna, "Scrambled linear pseudorandom number generators," *arXiv preprint arXiv:1805.01407*, 2018. 82

[125] J.-S. Coron, F. Rondepierre, and R. Zeitoun, "High order masking of look-up tables with common shares," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 40–72, 2018. 82

[126] J.-S. Coron, A. Roy, and S. Vivek, "Fast evaluation of polynomials over binary finite fields and application to side-channel countermeasures," in *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer, 2014, pp. 170–187. 82

[127] M. Feldhofer, M. Aigner, T. Baier, M. Hutter, T. Plos, and E. Wenger, "Semi-passive rfid development platform for implementing and attacking security tags," in

2010 International Conference for Internet Technology and Secured Transactions. IEEE, 2010, pp. 1–6. 87

[128] A. Engel, B. Liebig, and A. Koch, "Feasibility analysis of reconfigurable computing in low-power wireless sensor applications," in *International Symposium on Applied Reconfigurable Computing*. Springer, 2011, pp. 261–268. 87

[129] C. Manifavas, G. Hatzivasilis, K. Fysarakis, and Y. Papaefstathiou, "A survey of lightweight stream ciphers for embedded systems," *Security and Communication Networks*, vol. 9, no. 10, pp. 1226–1246, 2016. 96

[130] A. Maximov and A. Biryukov, "Two trivial attacks on trivium," in *International Workshop on Selected Areas in Cryptography*. Springer, 2007, pp. 36–55. 96

[131] K. Gaj, G. Southern, and R. Bachimanchi, "Comparison of hardware performance of selected phase ii estream candidates," in *State of the Art of Stream Ciphers Workshop (SASC 2007), eSTREAM, ECRYPT Stream Cipher Project, Report*, vol. 26, 2007, p. 2007. 97

[132] P. Bulens, K. Kalach, F.-X. Standaert, and J.-J. Quisquater, "Fpga implementations of estream phase-2 focus candidates with hardware profile," in *State of the Art of Stream Ciphers Workshop (SASC 2007), eSTREAM, ECRYPT Stream Cipher Project, Report*, vol. 24, 2007, p. 2007. 97

[133] Y. Wu, J. P. Noonan, S. Agaian *et al.*, "Npcr and uaci randomness tests for image encryption," *Cyber journals: multidisciplinary journals in science and technology, Journal of Selected Areas in Telecommunications (JSAT)*, vol. 1, no. 2, pp. 31–38, 2011. 97, 134

[134] Y. Wu, Y. Zhou, G. Saveriades, S. Agaian, J. P. Noonan, and P. Natarajan, "Local shannon entropy measure with statistical tests for image randomness," *Information Sciences*, vol. 222, pp. 323–342, 2013. 102

[135] S. El Assad and H. Noura, "Generator of chaotic sequences and corresponding generating system," 2011. 113

[136] R. Hamza, "A novel pseudo random sequence generator for image-cryptographic applications," *Journal of Information Security and Applications*, vol. 35, pp. 119–127, 2017. 113

[137] F. Dridi, S. El Assad, W. El Hadj Youssef, M. Machhout, and R. Lozi, "The design and fpga-based implementation of a stream cipher based on a secure chaotic generator," *Applied Sciences*, vol. 11, no. 2, p. 625, 2021. 113

[138] F. Özkaynak and A. B. Özer, "A method for designing strong s-boxes based on chaotic lorenz system," *Physics Letters A*, vol. 374, no. 36, pp. 3733–3738, 2010. 116

[139] X. Wang and Q. Wang, "A novel image encryption algorithm based on dynamic s-boxes constructed by chaos," *Nonlinear Dynamics*, vol. 75, no. 3, pp. 567–576, 2014. 116

[140] Ü. Çavuşoğlu, A. Zengin, I. Pehlivan, and S. Kaçar, "A novel approach for strong s-box generation algorithm design based on chaotic scaled zhongtang system," *Nonlinear dynamics*, vol. 87, no. 2, pp. 1081–1094, 2017. 116, 126

[141] D. Lambić, "A novel method of s-box design based on discrete chaotic map," *Nonlinear dynamics*, vol. 87, no. 4, pp. 2407–2413, 2017. 116, 126

[142] H. A. Ahmed, M. F. Zolkipli, and M. Ahmad, "A novel efficient substitution-box design based on firefly algorithm and discrete chaotic map," *Neural Computing and Applications*, vol. 31, no. 11, pp. 7201–7210, 2019. 116, 126

[143] T. Shah, A. Qureshi, and M. Usman, "A novel color image encryption scheme based on arnold's cat map and 16-byte s-box," *Applications and Applied Mathematics: An International Journal (AAM)*, vol. 16, no. 1, p. 33, 2021. 116

[144] K.-W. Wong, B. S.-H. Kwok, and W.-S. Law, "A fast image encryption scheme based on chaotic standard map," *Physics Letters A*, vol. 372, no. 15, pp. 2645–2652, 2008. 120

[145] Q. Lai, A. Akgul, C. Li, G. Xu, and Ü. Çavuşoğlu, "A new chaotic system with multiple attractors: Dynamic analysis, circuit realization and s-box design," *Entropy*, vol. 20, no. 1, p. 12, 2018. 126

[146] E. Al Solami, M. Ahmad, C. Volos, M. N. Doja, and M. M. S. Beg, "A new hyperchaotic system-based design for efficient bijective substitution-boxes," *entropy*, vol. 20, no. 7, p. 525, 2018. 126

[147] Y. Luo, R. Zhou, J. Liu, S. Qiu, and Y. Cao, "An efficient and self-adapting colour-image encryption algorithm based on chaos and interactions among multiple layers," *Multimedia Tools and Applications*, vol. 77, no. 20, pp. 26 191–26 217, 2018. 139

[148] L. Huang, S. Cai, X. Xiong, and M. Xiao, "On symmetric color image encryption system with permutation-diffusion simultaneous operation," *Optics and Lasers in Engineering*, vol. 115, pp. 7–20, 2019. 139

160