UNIVERSIDADE FEDERAL DE MINAS GERAIS
School of Engineering
Graduate Program in Electrical Engineering

Lucas Giovani Nardo

EXPLOITING THE CHAOTIC DYNAMICS: NOVEL APPROACHES FOR CRYPTOSYSTEMS, DIGITAL SYSTEMS, AND LYAPUNOV EXPONENT ESTIMATION

Belo Horizonte
2024

Lucas Giovani Nardo

EXPLOITING THE CHAOTIC DYNAMICS: NOVEL APPROACHES FOR CRYPTOSYSTEMS, DIGITAL SYSTEMS, AND LYAPUNOV EXPONENT ESTIMATION

Thesis presented to the Graduate Program in Electrical Engineering of the School of Engineering at the Universidade Federal de Minas Gerais in partial fulfillment of the requirements to obtain the degree of Doctor in Electrical Engineering.

Supervisor:	Janier Arias Garcia
Co-supervisor:	Erivelton Geraldo Nepomuceno

Belo Horizonte
2024

Nardo, Lucas Giovani.
N224e Exploiting the chaotic dynamics [recurso eletrônico] : novel approaches for cryptosystems, digital systems, and Lyapunov exponent estimation / Lucas Giovani Nardo. - 2024.
1 recurso online (106 f. : il., color.) : pdf.

Orientador: Janier Arias Garcia.
Coorientador: Erivelton Geraldo Nepomuceno.
.
Tese (doutorado) - Universidade Federal de Minas Gerais, Escola de Engenharia.

Apêndices: f. 97-106.

Bibliografia: f. 85-96.
Exigências do sistema: Adobe Acrobat Reader.

1. Engenharia elétrica - Teses. 2. Liapunov, Funções de - Teses. 3. Teoria do caos - Teses. 4. Sistemas caóticos - Teses. 5. Computadores - Medidas de segurança - Teses. 6. Galois, Teoria de - Teses. I. Arias García, Janier. II. Nepomuceno, Erivelton Geraldo. III. Universidade Federal de Minas Gerais. Escola de Engenharia. IV. Título.

CDU: 621.3(043)

Ficha catalográfica elaborada pelo bibliotecário Reginaldo César Vital dos Santos CRB/6 2165
Biblioteca Prof. Mário Werneck, Escola de Engenharia da UFMG

UNIVERSIDADE FEDERAL DE MINAS GERAIS

"EXPLOITING THE CHAOTIC DYNAMICS: NOVEL APPROACHES FOR CRYPTOSYSTEMS, DIGITAL SYSTEMS, AND LYAPUNOV EXPONENT ESTIMATION"

LUCAS GIOVANI NARDO

Tese de Doutorado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do grau de Doutor em Engenharia Elétrica. Aprovada em 22 de abril de 2024. Por:

Referência: Processo nº 23072.222941/2024-30 SEI nº 3195943

To those who inspired and without whom this thesis would not have been completed.

Acknowledgment

First and foremost, I would like to thank God, the Almighty, who has guided and provided me with the opportunity to reach another milestone in my academic journey.

I am thankful to my parents, Claudio and Ermelinda, for their invaluable advice, blessings, vast knowledge, and unwavering support. Their constant presence has shaped my character and played a pivotal role in making me the person I am today.

A special thanks go to my beloved wife, Vic. Her singular kindness and tireless support have been my greatest source of strength in pursuing my dreams. Without her, the completion of this thesis would not have been possible.

I am grateful to my sister, Rita, for her love and countless prayers that have helped me through challenging times.

My heartfelt appreciation goes out to my grandparents, Maria Cecília (*in memoriam*), Mario, Maria Aparecida (*in memoriam*), and Paulino (*in memoriam*), for their support and unconditional love.

My sheer gratitude goes to Prof. Dr. Janier Arias Garcia and Prof. Dr. Erivelton Geraldo Nepomuceno, my remarkable and patient supervisors. Their guidance and recognition have been instrumental in every step of this journey. They have shown me the marvels of science and engineering.

I would like to acknowledge the valuable contributions of the remaining members of my thesis committee. Their diligent reading, insightful comments, and thought-provoking questions improved the quality of this thesis.

I am grateful to fellow researchers I have had the privilege to work with at the Universidade Federal de Minas Gerais and Maynooth University. You have been important in the tough times and the adversities I have faced.

Last, but not least, I am indebted to the Universidade Federal de Minas Gerais for its academic aid, Maynooth University for hosting me during my internship, and Brazilian agencies FAPEMIG and CAPES for their financial assistance. These institutions have been fundamental to my academic ambitions.

"The whole of science is nothing more than a refinement of everyday thinking."
Albert Einstein.

Resumo

Após as inovadoras contribuições de Henri Poincaré e Edward Lorenz no campo de estudo da teoria do caos, sistemas caóticos começaram a receber significativa atenção na comunidade científica. Sistemas caóticos exibem características, como a sensibilidade às condições iniciais e o comportamento não periódico. Assim, pesquisadores têm usado essas características em diversas aplicações digitais, desde a geração de números pseudoaleatórios, até em algoritmos de otimização metaheurística. No entanto, implementar e simular sistemas caóticos em um ambiente digital é uma tarefa complexa, que exige demasiada consideração em muitas questões, como a representação numérica, precisão finita e a degradação dinâmica de suas propriedades caóticas. Diante desse cenário, esta tese demonstra que as órbitas geradas, via sistemas caóticos, podem variar em diferentes configurações de *hardware* e *software*, o que afeta negativamente aplicações de criptosistemas baseados em caos, por exemplo. Essa divergência se estende, também, para os modelos de sistemas caóticos matematicamente equivalentes quando a ordem de execução das suas respectivas operações aritméticas se difere. Além disso, embora esforços substanciais tenham sido dedicados à representação digital de sistemas caóticos e à mitigação da degradação dinâmica, pouca atenção foi dada ao desenvolvimento de circuitos digitais eficientes para tais sistemas. Assim, essa tese utiliza dessas percepções para explorar a dinâmica caótica por meio de três abordagens distintas. Primeiramente, um novo algoritmo de criptografia de imagens baseado em corpos finitos é apresentado. Ao adaptar o mapa logístico usando corpos de Galois para gerar números pseudoaleatórios, tal abordagem assegura resultados consistentes no processo de criptografia em diferentes dispositivos digitais, garantindo, também, a segurança contra ataques cibernéticos. Em segundo lugar, um novo método para estimar o maior e positivo expoente de Lyapunov de sistemas caóticos usando programas SPICE é proposto. Tal abordagem simplifica a respectiva estimativa ao analisar diretamente sistemas representados em diagramas de circuitos, inspirando-se em extensões intervalares e pseudo-órbitas de sistemas caóticos. Assim, usuários com habilidades básicas em programas semelhantes ao SPICE podem empregar facilmente esse método. Em terceiro lugar, utilizando o FPGA como uma plataforma digital e usando a representação numérica em ponto fixo e complemento de 1, uma solução eficiente em *hardware* para implementar o mapa tenda, como também um método de perturbação via *hardware* para reduzir a sua degradação dinâmica, são apresentados. O circuito digital resultante mantém as propriedades caóticas e atua como um gerador de números pseudoaleatórios para problemas de otimização por enxame de partículas. Por fim, esta tese não apenas explora sistemas caóticos em ambientes digitais, mas também oferece soluções práticas e *insights* com implicações para diversas aplicações.

Palavras-chave: caos; sistema caótico; computação aritmética; degradação dinâmica; corpo de Galois; arquitetura em *hardware*; encriptação de imagem; expoente de Lyapunov; gerador de número pseudoaleatório.

Abstract

After the groundbreaking contributions of Henri Poincaré and Edward Lorenz to the field of chaos, chaotic systems have received significant attention in the scientific community. These systems exhibit characteristics like sensitivity to initial conditions and non-periodic behavior. Researchers have used these features in diverse digital applications, from pseudo-random number generation to metaheuristic optimization algorithms. However, implementing and simulating chaotic systems in the digital realm is a complex task, necessitating careful consideration of many issues, such as number representation, finite precision, and dynamical degradation of their chaotic properties. This thesis shows that the orbits of chaotic systems can vary across distinct hardware and software setups, negatively impacting applications like chaos-based encryption schemes. This divergence extends to mathematically equivalent chaotic system models when the order to perform their arithmetic operations differs. Furthermore, while substantial efforts have been devoted to digitally representing chaotic systems and mitigating dynamical degradation, limited attention has been given to developing efficient digital circuits for these systems. Thus, this thesis uses these insights to exploit the chaotic dynamics through three distinct approaches. First, a novel chaotic image encryption algorithm based on finite fields is introduced. By adapting the logistic map using Galois field to generate pseudo-random numbers, this approach ensures that encryption results are consistent across different digital devices and guarantees security against cyberattacks. Second, a new method to estimate the positive largest Lyapunov exponent of chaotic systems using SPICE-like programs is proposed. This approach simplifies the estimation by directly analyzing systems represented in circuit diagrams, inspired by interval extensions and pseudo-orbits of chaotic systems. Thus, users with basic SPICE-like program skills can readily employ this method. Third, utilizing FPGA as a digital platform and using 1's complement fixed-point format, a hardware-efficient solution for implementing the tent map and also a hardware perturbation method to reduce its dynamical degradation are presented. The resulting digital circuit maintains chaotic properties and serves as a pseudo-random number generator for particle swarm optimization. In summary, this thesis not only explores chaotic systems in digital environments but also offers practical solutions and insights with implications for diverse applications.

Keywords: chaos; chaotic system; computer arithmetic; dynamical degradation; Galois field; hardware architecture; image encryption; Lyapunov exponent; pseudo-random number generator.

List of Figures

1.1 Number of documents related to chaos theory by year in *Scopus* database. 21

1.2 Taxonomy of chaotic dynamical systems. 22

1.3 The difference in machine configurations contributes to results discrepancies in four chaotic systems. 24

1.4 Divergent outcomes in four chaotic systems result from employing different natural interval extensions. 26

1.5 Orbit of a chaotic system in the digital domain. 27

1.6 Dynamical degradation on the tent map. 28

1.7 Methods to counteract the dynamical degradation. 29

2.1 A representation of the procedure for estimating the Lyapunov exponent of a dynamical system based on Wolf et al.'s method. 35

2.2 A 32-bit representation for fixed-point arithmetic. 36

2.3 Single-precision floating-point format adopted by IEEE 754-2019. 37

2.4 The autonomous Chua's circuit. 42

2.5 Divergence in the simulation of two equivalent Chua's circuits. 43

3.1 Flowchart diagram of the algorithm based on the logistic map and Galois field. 49

3.2 Plain and encrypted images, followed by their respective histograms. . . . 52

3.3 Reproducibility of chaos-based encryption algorithms on different devices. . 55

4.1 Schematic of how the largest Lyapunov exponent is determined. 61

4.2 The lower-bound error for the jerk circuit. 63

5.1 Flowchart representation of the digital tent map. 69

5.2 A digital representation of the tent map. 70

5.3 The results achieved by the proposed chaotic circuit and perturbation method. 73

5.4 Implemented architectures for PSO. 77

B.1 Schematic to determine the largest Lyapunov exponent for Chua's circuit. . 99

B.2 Calculation of the lower-bound error for Chua's circuit. 99

B.3 Schematic to determine the largest Lyapunov exponent for Jerk 1. 100
B.4 Calculation of the lower-bound error for Jerk 1. 100
B.5 Schematic to determine the largest Lyapunov exponent for Jerk 2. 101
B.6 Calculation of the lower-bound error for Jerk 2. 101
B.7 Schematic to determine the largest Lyapunov exponent for Jerk 3. 102
B.8 Calculation of the lower-bound error for Jerk 3. 102
B.9 Schematic to determine the largest Lyapunov exponent for Rössler circuit. 103
B.10 Calculation of the lower-bound error for Rössler circuit. 104
B.11 Schematic to determine the largest Lyapunov exponent for a hyperchaotic circuit. 105
B.12 Calculation of the lower-bound error for a hyperchaotic circuit. 106

List of Tables

1.1 The four continuous chaotic systems employed to simulate in two distinct devices. . . . 23

1.2 Description of each device used to simulate the four chaotic systems presented in Table 1.1. . . . 23

1.3 Natural interval extensions of four chaotic systems. . . . 25

1.4 Publication details and corresponding thesis chapters discussing results achieved by the doctoral student. . . . 30

2.1 Cayley tables for the group $\mathbb{Z}_3$. . . . 38

2.2 Polynomial addition and multiplication on $GF(2)[x]/(f)$. . . . 39

3.1 Parameters and the plain image factors f_{P_I}, which were used to perform the encryption algorithm. . . . 50

3.2 The results of the ENT test suite for the proposed cryptosystem. . . . 51

3.3 Correlation coefficients and information entropy for each test image. . . . 52

3.4 Results of key sensitivity analysis for Baboon image. . . . 53

3.5 NPCR and UACI results for the proposed encryption scheme. . . . 54

3.6 Description of each device used to simulate the proposed encryption scheme and the other two algorithms. . . . 55

3.7 Performance comparison using the Baboon image (512×512). . . . 56

4.1 List of chaotic circuits used to test the proposed method that estimates the LLE. . . . 62

4.2 The determined largest Lyapunov exponents using the proposed and Wolf et al.'s approaches. . . . 63

5.1 Approaches for counteracting the dynamical degradation in the tent map. . . 68

5.2 Components in digital tent map representation with varying resolutions, compared to a design using the logistic map. . . . 71

5.3 Comparing logical components of chaotic maps across different approaches. 72

5.4 Quantitative results demonstrate the chaotic nature and randomness of the proposed digital circuit. . . . 72

5.5 The results of the ENT test suite for the proposed digital tent map. 74
5.6 Implementation results in the PSO design. 78
5.7 Experiment parameters for validating the proposed digital tent map as a pseudo-random number generator in hardware for particle swarm optimization. 79
5.8 Convergence results in the PSO design. 79

6.1 Details of each additional publication achieved by the doctoral student during their postgraduate studies. 83

List of Abbreviations

AES	Advanced Encryption Standard.
ASCII	American Standard Code for Information Interchange.
BRAM	Block RAM.
CPSO	Chaotic Particle Swarm Optimization.
DES	Data Encryption Standard.
DSP	Digital Signal Processing.
FF	Flip-Flop.
FP	Floating-Point.
FPGA	Field-Programmable Gate Array.
FSM	Finite State Machine.
HUB	Half-Unit-Biased.
IEEE	Institute of Electrical and Electronics Engineers.
LE	Lyapunov Exponent.
LFSR	Linear Feedback Shift Register.
LLE	Largest Lyapunov Exponent.
LUT	Look-Up Table.
MATLAB	Matrix Laboratory.
MUX	Multiplexer.
NARMAX	Nonlinear Autoregressive Moving Average Model with Exogenous Inputs.
NIE	Natural Interval Extension.
NPCR	Number of Pixels Change Rate.
PRNG	Pseudo-Random Number Generator.
PSO	Particle Swarm Optimization.
RAM	Random-Access Memory.
SPICE	Simulation Program with Integrated Circuit Emphasis.
UACI	Unified Average Changing Intensity.
UTF-16	16-bit Unicode Transformation Format.
VHDL	VHSIC Hardware Description Language.
VHSIC	Very High-Speed Integrated Circuit.
XOR	Exclusive-OR.

List of Symbols and Notation

$(G, +, \cdot)$	A field.
$(G, \cdot)$	A group.
$\lvert x \rvert$	Absolute value of the real number x.
$\tilde{x}$	Bitwise complement of variable x.
C_i	Capacitance of the capacitor i.
$\dot{x}$	Derivative of a variable x with respect to time.
$\psi_{i,n}$	Error associated between orbit and i^{th} pseudo-orbit at the n^{th} instant.
$\lvert\lvert x \rvert\rvert$	Euclidean norm of vector x.
$\oplus$	Exclusive-OR operator.
$E[\cdot]$	Expected value.
$GF(q)$	Finite field of order q.
$f(x)$	Function of a variable x.
$I_{m \times m}$	Identity matrix of order m.
$\underline{X}$	Lower bound of the interval X.
eps	Machine epsilon.
$\max(f_1(x), f_2(x))$	Maximal value between functions $f_1(x)$ and $f_2(x)$.
$\min(f_1(x), f_2(x))$	Minimal value between functions $f_1(x)$ and $f_2(x)$.
$0_{m \times m}$	Null matrix of order m.
x^*	Optimum point of the fitness function $f(x)$.
$GF(q)[x]$	Polynomial ring over the ring $GF(q)$.
μ_x	Population mean of a random variable x.
$Pr(x)$	Probability of the event x occurring.
$\mathbb{R}$	Real number set.
$[\underline{X}, \overline{X}]$	Representation of an interval using lower and upper bounds.
$GF(q)[x]/f$	Residue class ring of the ring $GF(q)$ modulo f.
R_i	Resistance of the resistor i.
$\{x_n\}$	Sequence of the variable x with n values.
$\mathbb{Z}_p$	Set of integers modulo a prime number.
$\mathbb{R}^+$	Set of positive real values.
$sgn(x)$	Sign of the real number x.

σ_x	Standard deviation of a population for a random variable x.
b_i	The i^{th} bit in a bitstring.
$I_{i,n}$	The i^{th} interval I at the n^{th} instant.
$\hat{x}_{i,n}$	The i^{th} pseudo-orbit of the variable x at the n^{th} instant.
$\{\hat{x}_{i,n}\}$	The series of the i^{th} pseudo-orbit of the variable x with n values.
t_i	Time instant i.
ulp	Unit in the last place.
$\overline{X}$	Upper bound of the interval X.
x_n	Variable x at the n^{th} instant.

Contents

1 Introduction **21**

1.1 Motivation . . . 21

1.2 Aims . . . 30

1.3 Thesis-Related Publications . . . 31

1.4 Thesis Organization . . . 31

2 Preliminary Concepts **33**

2.1 Chaos and Characterization of Chaotic Dynamics . . . 33

2.1.1 A Mathematical Definition of Chaos . . . 33

2.1.2 Lyapunov Exponent . . . 34

2.2 Number Representation . . . 36

2.2.1 Fixed-Point Arithmetic . . . 36

2.2.2 Floating-Point Arithmetic and IEEE Standard 754-2019 . . . 36

2.3 Finite Field . . . 37

2.4 Interval Arithmetic . . . 40

2.4.1 Interval Extensions and the Lower-Bound Error . . . 41

3 Chaos-Based Cryptosystem **44**

3.1 State-of-the-Art and Problem Description . . . 44

3.2 Image Encryption Scheme . . . 46

3.3 Performance Analysis . . . 48

3.3.1 Key Space . . . 50

3.3.2 ENT Test Suite . . . 50

3.3.3 Correlation of Adjacent Pixels, Entropy, and Histogram Analysis . . . 51

3.3.4 Key Sensitivity and Differential Analysis . . . 52

3.3.5 Reliability on Different Devices . . . 54

3.3.6 Performance Comparison . . . 56

3.4 Final Remarks . . . 56

4 Lyapunov Exponent Estimation **58**

4.1 State-of-the-Art and Problem Description . . . 58

4.2 Lyapunov Exponent Estimation Method . . . 59
4.3 Results and Discussion . . . 61
4.4 Final Remarks . . . 63

5 Digital Chaotic Circuit **65**
5.1 State-of-the-Art and Problem Description . . . 65
5.2 Design of the Digital Tent Map . . . 66
5.2.1 Tent Map . . . 66
5.2.2 Method . . . 67
5.3 Results and Discussion . . . 69
5.3.1 Hardware Performance . . . 69
5.3.2 Hardware Architecture for Particle Swarm Optimization . . . 74
5.4 Final Remarks . . . 79

6 Conclusion and Final Remarks **81**
6.1 Overview . . . 81
6.2 Future Research . . . 82
6.3 Additional Doctoral Publications . . . 83

References **85**

A Algorithms **97**

B SPICE - Circuits and Graphics **98**

Chapter 1

Introduction

1.1 Motivation

A systematic investigation of chaos theory has been developed over the last decades. This development is evident in the increasing number of articles and the research community's interest, as shown in Figure 1.1.

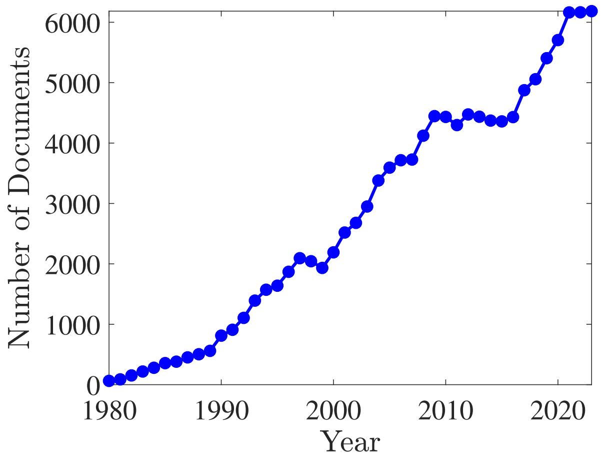


Figure 1.1: Number of documents related to chaos theory by year in *Scopus* database. It shows a noticeable growth trend in this area over the years. To sample these data, from the *Scopus* database on 19 February 2024, a search using `TITLE-ABS-KEY("chaos" OR "chaotic system")` was performed. The period evaluated is from 1980 to 2023. `Document evaluated: chaos_search.csv.`

Researchers have been discovering many chaotic systems in the most diverse fields, such as Lorenz system [65] in atmospheric sciences, Rössler system in chemistry [109], and the RLD circuit in electronic systems [24]. There are several chaotic systems in the literature, differing from each other in many ways, including the number of dimensions, whether there are hidden and/or self-excited attractors, and how the variables evolve over time. Figure 1.2 shows a taxonomy of chaotic systems, dividing them into discrete-time or continuous-time models.

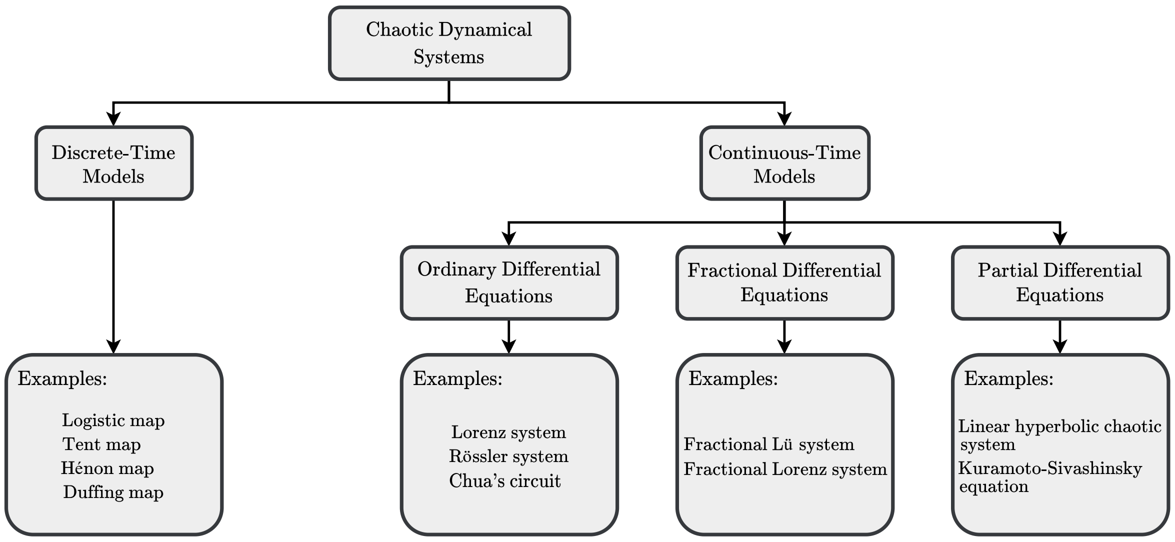


Figure 1.2: Taxonomy of chaotic dynamical systems. This curated classification divided the many chaotic systems into two main categories, separated by how the model variables evolve over time. This thesis investigates chaos in both discrete-time and continuous-time models.

The enthusiasm for chaotic systems arises from their interesting properties, such as sensitivity to initial conditions, transitivity, and dense periodic points [7]. These unique characteristics find application in diverse fields, ranging from crafting Pseudo-Random Number Generators (PRNGs), encryption algorithms, and secure chaotic communication systems [17, 82, 91, 108], to advancing artificial intelligence through techniques such as Chaotic Particle Swarm Optimization (CPSO) [3, 63], designing inductive sensors for precision [48], and developing models and controllers for system identification and control [1, 138].

Implementing and simulating chaotic systems, whether discrete-time or continuous-time, in the digital domain with finite precision presents a significant challenge due to their dynamic properties [86, 90]. As mentioned earlier, chaotic dynamical systems are sensitive, and a slight perturbation can lead to different orbits. This sensitivity also extends to hardware and software discrepancies when operating in the digital realm [113, 114]. Consequently, even if two systems share identical mathematical equations and initial conditions, they may produce distinct results when implemented on different hardware or software platforms.

For instance, Table 1.1 lists four distinct continuous chaotic systems that are widely employed in the literature. Using two devices, described in Table 1.2, but under the same MATLAB version, initial conditions, and algorithms, Figure 1.3 shows that, after some time, the chaotic orbits diverge from each other.

These results confirm that applications which require the exact reproduction of the chaotic system's dynamics, such as chaos-based encryption algorithms, face some challenges. This is because round-off errors can lead to dissimilar keystreams.

Table 1.1: The four continuous chaotic systems employed to simulate in two distinct devices.

Chaotic systems	
Chen's hyperchaotic system [34]	**Chua's circuit** [19]
$\begin{cases} \dot{x} = 36(y-x) \\ \dot{y} = -16x - xz + 28y - w \\ \dot{z} = xy - 3z \\ \dot{w} = x + 0.2 \end{cases}$	$\begin{cases} \dot{x} = 15.6(y - x - f(x)) \\ \dot{y} = x - y + z \\ \dot{z} = -28y \end{cases}$ $f(x) = -0.714x - 0.2145(\lvert x+1\rvert - \lvert x-1\rvert)$
Rössler system [109]	**Sprott's jerk system** [122]
$\begin{cases} \dot{x} = -y - z \\ \dot{y} = x + 0.2y \\ \dot{z} = 0.2 + z(x - 5.7) \end{cases}$	$\begin{cases} \dot{x} = y \\ \dot{y} = z \\ \dot{z} = -0.5z - x - 10^{-9}(e^{(y/0.026)} - 1) \end{cases}$

Table 1.2: Description of each device used to simulate the four chaotic systems presented in Table 1.1. The same MATLAB version, initial conditions, and algorithms were used in Devices 1 and 2.

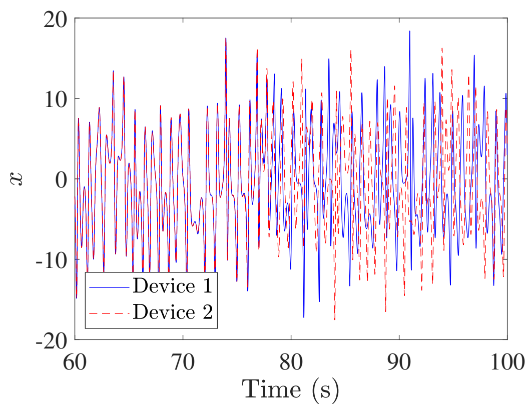
Description	
Device 1	Intel(R) Core(TM) i5-8265U CPU @ 1.60 GHz - Windows system
Device 2	Intel(R) Xeon(R) E5-2620 v4 CPU @ 2.10 GHz - Linux system

To address this problem, this thesis proposes a novel chaotic image encryption algorithm based on finite fields. Finite fields are mathematical structures that consist of a finite set of elements with well-defined operations, including addition, subtraction, multiplication, and division. These operations in finite fields have zero error, eliminating the presence of rounding errors. Thus, the logistic map was adapted using Galois field to generate pseudo-random numbers. In this way, the generated cipher is the same in different devices, not compromising the keystream. This approach has shown sufficient pseudo-random properties and is resistant against many cyberattacks.

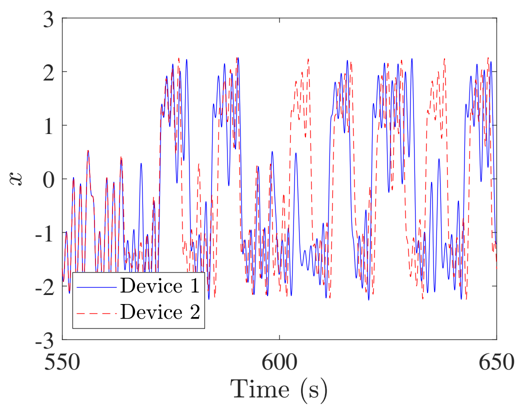
In addition to variations in hardware and software configurations, resultant chaotic orbits can also diverge in mathematically equivalent chaotic system models when the order of their arithmetic operations differs. Nepomuceno and Mendes [90] argued about the implementation sensitivity of chaotic systems apart from the sensitivity to initial conditions. They demonstrated the existence of multiple chaotic orbits in nonlinear continuous chaotic systems when discretization schemes are applied, even when the step size remains constant, and the initial conditions are unchanged. Similarly, Sayed et al. [114] found that the order of addition and multiplication can influence the dynamics of both continuous and discrete chaotic systems, displaying divergence in time series.

For illustration, consider the Natural Interval Extensions (NIEs)[1] of four chaotic

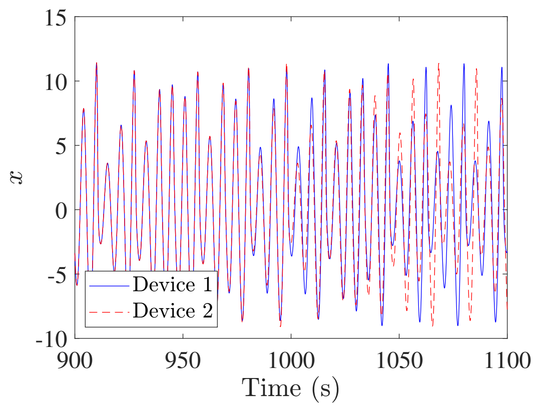
[1]To delve deeper into the concept and definitions of natural interval extension, find comprehensive information in Section 2.4.



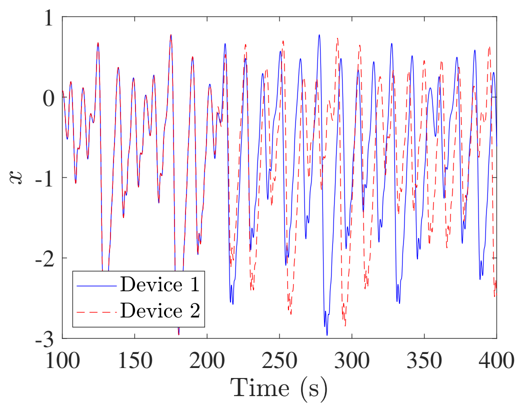
(a) Chen's hyperchaotic system.



(b) Chua's circuit.



(c) Rössler system.



(d) Sprott's jerk system.

Figure 1.3: The difference in machine configurations contributes to results discrepancies in four chaotic systems. Despite identical initial conditions on both machines, the device configuration can interfere with the system simulation, influencing the evolution of the system's orbit. All simulations used a step size of 0.01 s. The variable x represents the value of the x-orbit trajectory along time. `Algorithms: Hardware and Software Divergence / main_chen.m, main_chua.m, main_rossler.m, and main_sprott.m`.

systems, as presented in Table 1.3. Note these systems are mathematically equivalent to their counterparts, differing only in the sequence of basic arithmetic operations. When utilizing the same hardware and software configuration, discretization process (4^{th} order Runge-Kutta method), and identical initial conditions, simulations yield divergent results, as demonstrated in Figure 1.4, depending on which natural interval extension is used.

Mendes and Nepomuceno [70] demonstrated the exponential divergence in the distances between the orbits of simulated chaotic systems using natural interval extensions. They drew parallels to the calculation of the Lyapunov exponent, a quantitative index that measures the sensitivity of initial conditions in systems.

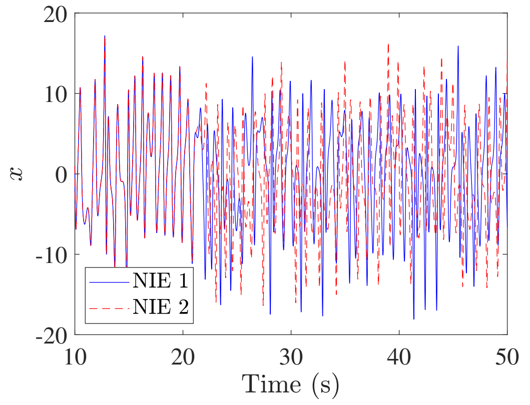
Building upon Mendes and Nepomuceno's research [70], a novel approach to directly determine the Largest Lyapunov Exponent (LLE) using SPICE-like programs, eliminating the need for auxiliary tools, is also proposed in this thesis. Analog electronic circuits represent one of the most popular categories of chaotic systems [121, 122]. Existing

Table 1.3: Natural interval extensions of four chaotic systems. Although mathematically equivalent to their counterparts, these extensions can produce different chaotic dynamics in computer simulations depending on the order of mathematical operations.

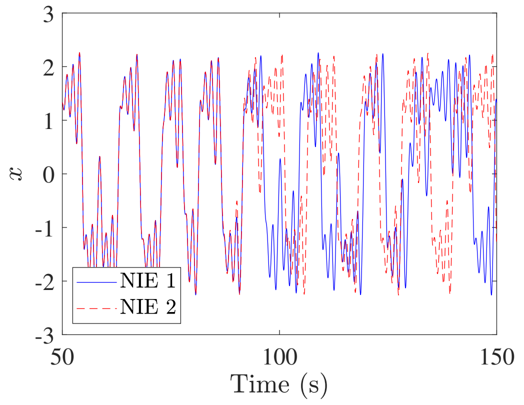
Chaotic systems	
Chen's hyperchaotic system [34]	
Natural interval extension 1	Natural interval extension 2
$\begin{cases} \dot{x} = 36(y-x) \\ \dot{y} = -16x - xz + 28y - w \\ \dot{z} = xy - 3z \\ \dot{w} = x + 0.2 \end{cases}$	$\begin{cases} \dot{x} = 36y - 36x \\ \dot{y} = -16x - xz + 28y - w \\ \dot{z} = xy - 3z \\ \dot{w} = x + 0.2 \end{cases}$
Chua's circuit [19]	
Natural interval extension 1	Natural interval extension 2
$\begin{cases} \dot{x} = 15.6(y - x - f(x)) \\ \dot{y} = x - y + z \\ \dot{z} = -28y \end{cases}$ $f(x) = -0.714x - 0.2145(\lvert x+1\rvert - \lvert x-1\rvert)$	$\begin{cases} \dot{x} = 15.6y - 15.6x - 15.6f(x) \\ \dot{y} = x - y + z \\ \dot{z} = -28y \end{cases}$ $f(x) = -0.714x - 0.2145(\lvert x+1\rvert - \lvert x-1\rvert)$
Rössler system [109]	
Natural interval extension 1	Natural interval extension 2
$\begin{cases} \dot{x} = -y - z \\ \dot{y} = x + 0.2y \\ \dot{z} = 0.2 + z(x - 5.7) \end{cases}$	$\begin{cases} \dot{x} = -y - z \\ \dot{y} = x + 0.2y \\ \dot{z} = 0.2 + xz - 5.7z \end{cases}$
Sprott's jerk system [122]	
Natural interval extension 1	Natural interval extension 2
$\begin{cases} \dot{x} = y \\ \dot{y} = z \\ \dot{z} = -0.5z - x - 10^{-9}(e^{(y/0.026)} - 1) \end{cases}$	$\begin{cases} \dot{x} = y \\ \dot{y} = z \\ \dot{z} = -0.5z + [-x - 10^{-9}(e^{(y/0.026)} - 1)] \end{cases}$

estimation methods typically require data extraction in software or a laboratory before applying a Lyapunov exponent estimation tool. The proposed method aims to assist the semiconductor industry by allowing the calculation of the LLE directly from systems represented in circuit diagrams. This approach is inspired by natural interval extensions that utilize divergent trajectories to calculate the lower-bound error. The exponent is determined by analyzing the slope of the line, which is derived from the lower-bound error. To calculate the exponent, a straightforward linear fit to the logarithm of the divergence observed in two circuit trajectories is utilized.

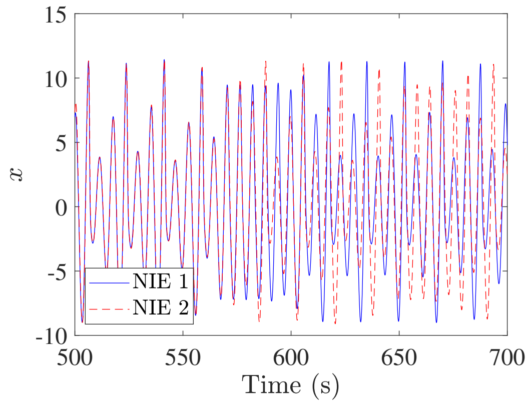
In addition to the challenges mentioned earlier for chaotic systems in finite-precision environments, it is crucial to highlight the dynamical degradation of these systems. In a digital domain, a chaotic system may undergo a substantial reduction in the largest possible orbit. As a result, several of its properties such as sensitivity to initial conditions and ergodicity, which are valid in continuous settings, may deteriorate when the system is discretized for finite computing precision. This deterioration is attributed to errors introduced during the discretization process, significantly impacting the system's dynamics and overall behavior.



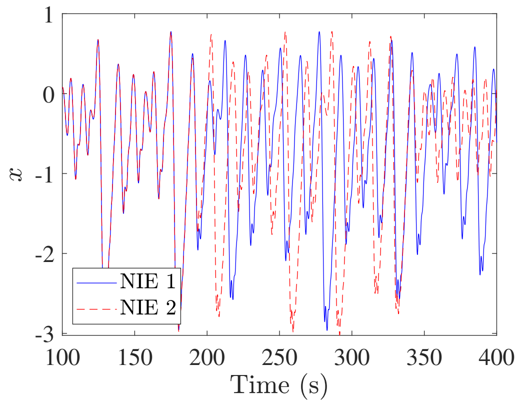
(a) Chen's hyperchaotic system.



(b) Chua's circuit.



(c) Rössler system.



(d) Sprott's jerk system.

Figure 1.4: Divergent outcomes in four chaotic systems result from employing different natural interval extensions. The simulation was conducted with an integration step of 0.01 s for all four systems. The x variable represents the trajectory of the x-orbit along time. `Algorithms: Natural Interval Extension / main_chen.m, main_chua.m, main_rossler.m, and main_sprott.m`.

Specifically, in a digital environment that uses a binary (base-2) number system with a precision of pr bits, the discrete space has a size of 2^{pr}. Consequently, as mentioned, the chaotic orbit eventually becomes periodic. Figure 1.5 vividly illustrates a typical orbit of a chaotic system in this environment. After a transient period, the orbit will settle into a cyclic pattern with a maximum length not exceeding 2^{pr}, which is the largest possible orbit [59].

To illustrate the dynamical degradation process, consider the commonly employed tent map [10], defined by $x_{n+1} = \mu \min(x_n, 1 - x_n)$, where $x_n \in [0, 1)$. Figure 1.6 shows six examples of such dynamical degradation. When employing an unsigned fixed-point number representation with a word length of 16 bits and a fractional length of 15 bits, the map converges into a loop of repeating values after a few iterations, displaying predictable periodic behavior rather than chaos. Note that the largest possible orbit of the tent map in this setup is 2^{15}, or 32,768 different values in the orbit. However, this does not actually occur.

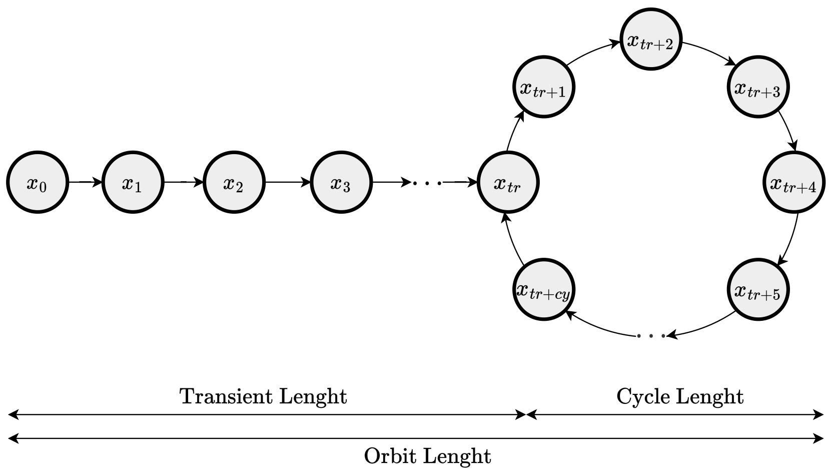
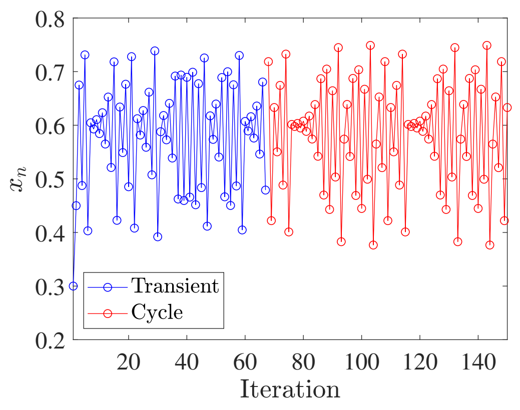


Figure 1.5: Orbit of a chaotic system in the digital domain, exhibiting indefinite recurrence after an initial transient period. Adapted from [59].

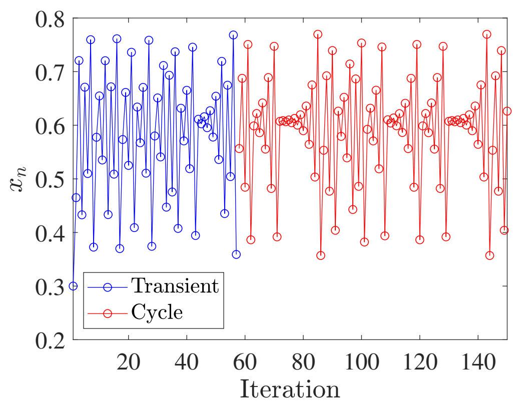
This finding can significantly affect chaos-based applications. For example, a chaos-based PRNG susceptible to such dynamical degradation does not present the desired randomness and cannot be used in any application at all. Consequently, the research community has shown a growing interest in analyzing and quantifying this degradation in digitized chaos, with a primary focus on mitigating its adverse effects on chaos-based digital systems [12, 59, 64]. As a result, numerous articles have tackled the problem of dynamical degradation in chaotic systems within the digital environment [12, 64], offering various solutions, including using high precision and employing chaotic systems with multiple dimensions. Figure 1.7 provides an overview of these approaches.

Although many ways are effective in diminishing such a problem, many methods are not suitable for hardware implementation. Indeed, Antonelli et al. [4] studied the dynamical degradation of chaotic maps in digital environments, using five chaotic maps and statistical quantifiers. Their findings revealed that while increasing the number of bits improves precision, the cost/benefit ratio between hardware implementation and chaotic map performance is often not favorable. Moreover, Wei et al. [134] implemented a five-dimensional hyperchaotic Burke-Shaw system in an FPGA (Field-Programmable Gate Array). Nonetheless, this implementation consumes substantial FPGA resources and demands a significant amount of energy to run.

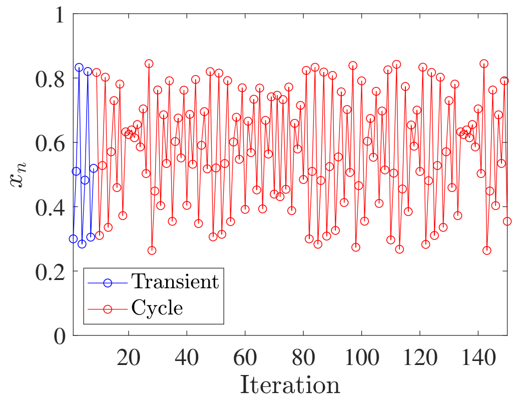
Concurrently, the literature review shows there is an interest in improving the performance of chaotic systems. In fact, the discovery of straightforward chaotic systems with complex dynamics has consistently been an interesting research scope [57]. Features such as the number of elements/components and the execution time are extremely important in such projects [103]. Analog circuits proposed by Sprott [121] are designed from simple mathematical models and common electronic components. Furthermore, Kennedy showed a rigorous proof of what an autonomous circuit must contain to exhibit chaos [51, 52]. This allows further development in building simple chaotic circuits.



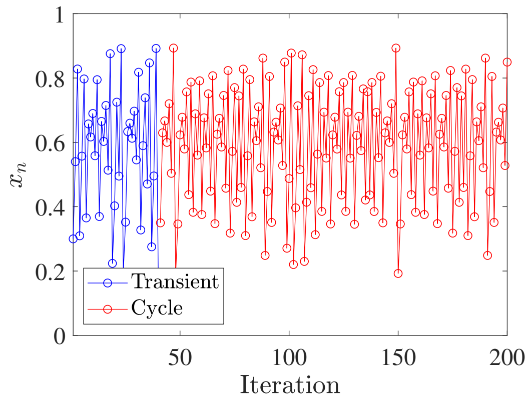
(a) $\mu = 1.5000$.



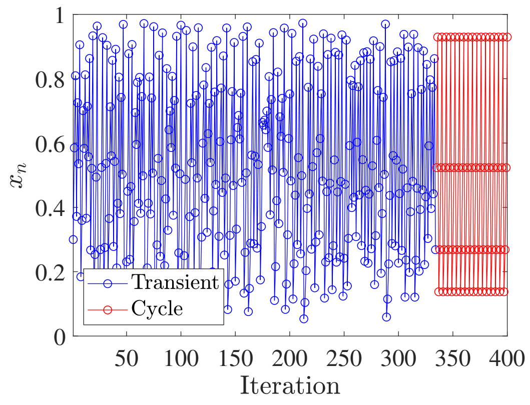
(b) $\mu = 1.5499$.



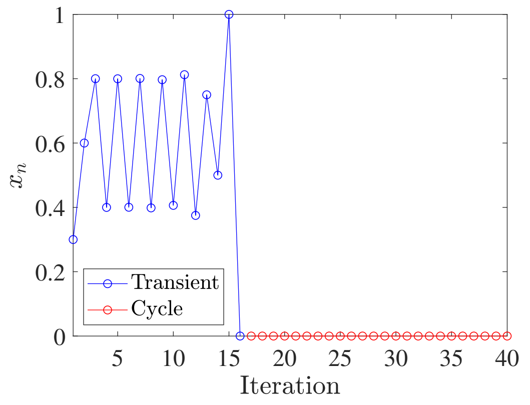
(c) $\mu = 1.7000$.



(d) $\mu = 1.7999$.



(e) $\mu = 1.9500$.



(f) $\mu = 1.9999$.

Figure 1.6: Dynamical degradation on the tent map for distinct parameters μ. The x_n variable indicates the value of the system trajectory throughout the iterations. As the parameter μ is varied, the system exhibits different degrees of dynamical degradation. `Algorithm: Dynamical Degradation / dynamical_degradation_tent.m`.

Despite considerable advances in the synthesis of chaotic analog circuits [51, 52, 57, 121, 122], the same cannot be said about the digitization of these systems. Great efforts have been made to represent chaotic maps on digital circuits, avoiding the dynamical degradation as well [77, 123, 131], but little attention has been paid to their hardware optimization

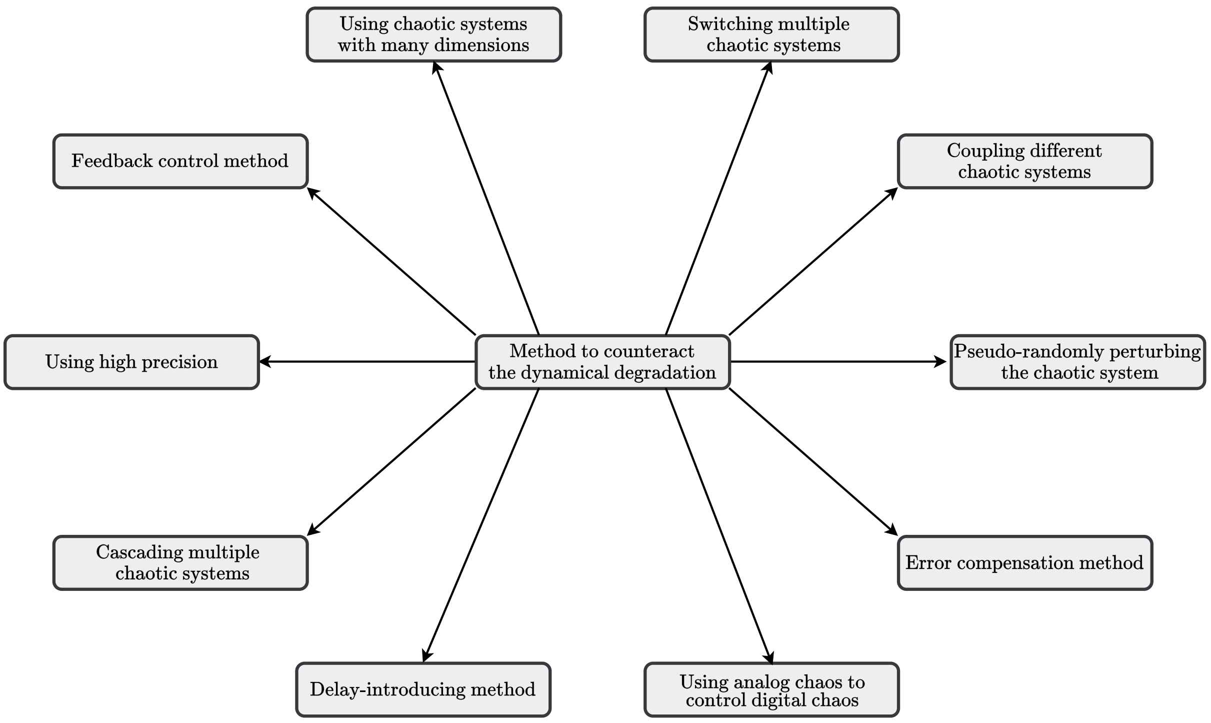

Using chaotic systems with many dimensions
Switching multiple chaotic systems
Feedback control method
Coupling different chaotic systems
Using high precision
Method to counteract the dynamical degradation
Pseudo-randomly perturbing the chaotic system
Cascading multiple chaotic systems
Error compensation method
Delay-introducing method
Using analog chaos to control digital chaos

Figure 1.7: Methods to counteract the dynamical degradation. Numerous approaches exist in the literature to mitigate the effects of the dynamical degradation of chaotic systems.

[87]. In other words, few papers deal with constructing digital chaotic systems with as few components as possible while also counteracting dynamical degradation. However, the design of efficient and simple digital chaotic systems is a fundamental issue triggered by increasing applications of chaos in the most diverse areas.

The simplicity of digital chaotic systems is closely linked to the number of arithmetic operations [87]. In their work, Nepomuceno et al. [87] utilized the logistic map to create a minimal digital chaotic system. Another alternative to the logistic map is the tent map, a binary shift chaotic map. However, it is well-known that these digital systems often suffer from dynamical degradation. Thus, this thesis also introduces a straightforward digital chaotic tent map based on the 1's complement fixed-point format. This number representation reduces both hardware and power consumption, thus improving execution time. To counteract dynamical degradation, a clear-cut perturbation method that involves altering only the least significant bit of the digital tent map using an XOR (eXclusive-OR) gate is employed. The main contribution here lies in significantly reducing logical resource consumption per bit while preserving the chaotic properties of the map and mitigating dynamical degradation.

Compared to other state-of-the-art articles in the literature, the proposed tent map showcases a remarkable reduction of at least 58% in resource consumption per bit. Moreover, it substantiates its pseudo-random attributes and practicality by seamlessly integrating it into a hardware-based pseudo-random number generator used in particle swarm optimization.

1.2 Aims

After presenting an overview of the field and identifying existing gaps, the thesis aims to explore the intricate dynamics of chaotic systems by providing a methodological approach to estimate the Lyapunov exponent of chaotic circuits, as well as techniques to design a cryptosystem and implement a digital chaotic map. To this end, attention will be paid to dynamical degradation and numerical issues, such as round-off errors and finite precision.

In order to satisfy the main aim, the following specific objectives are defined:

1. To prove that finite field and chaotic systems can create efficient and secure encryption algorithms, also showing that constructive limitations of computers interfere with the performance of different cryptosystems.

2. To establish that the concept of interval extensions and lower-bound error can be extended to circuits that exhibit chaotic behavior, also connecting these concepts in the estimation of the largest Lyapunov exponent of chaotic systems.

3. To analyze how number representations can affect the design of a digital chaotic map, through the evaluation of dynamical degradation, statistical tests, logical resource consumption, and the conservation of its chaotic properties.

4. To demonstrate that the proposed mechanism to mitigate dynamical degradation of the respective implemented chaotic map is effective and hardware-realizable, while the other approaches are not.

5. To indicate that chaotic systems can be applicable in many demanding situations in the industry.

Table 1.4: Publication details and corresponding thesis chapters discussing results achieved by the doctoral student.

Status	**Publication**	**Chapter**
Published	[83] - Nardo, L. G., Nepomuceno, E. G., Bastos, G. T., Santos, T. A., Butusov, D. N., and Arias-Garcia, J. (2021). A reliable chaos-based cryptography using Galois field. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(9):091101	Chapter 3
Published	[79] - Nardo, L., Nepomuceno, E., Muñoz, D., Butusov, D., and Arias-Garcia, J. (2023). A hardware-efficient perturbation method to the digital tent map. *Electronics*, 12(8):1953	Chapter 5
Accepted	[80] - Nardo, L. G., Nazaré, T. E., Nepomuceno, E., Arias-Garcia, J., and Butusov, D. N. (2024a). Computation of the largest Lyapunov exponent using SPICE-like programs. In *2024 7th IFAC Conference on Analysis and Control of Nonlinear Dynamics and Chaos (IFAC ACNDC)*, pages 1–6	Chapter 4

1.3 Thesis-Related Publications

This thesis consists of three articles: two have already been published, and one has been accepted to be presented at an international conference. It is possible to find a chronological list of these articles in Table 1.4.

1.4 Thesis Organization

The remainder of this thesis is structured into five chapters, which will be gradually described as follows:

- **Chapter 2**: this chapter introduces some fundamental concepts that are necessary for understanding the rest of this thesis. First, a theoretical explanation of chaotic systems is given. Next, Lyapunov exponents and one of the most traditional methods for estimating them are described. Then, two number representations, fixed-point and floating-point arithmetic, are presented. Finally, aspects of finite fields, interval arithmetic, interval extensions, and lower-bound error, which are essential for fully interpreting the subsequent chapters, are introduced.

- **Chapter 3**: the proposed chaos-based cryptosystem, highlighting its key differentiators from cutting-edge projects, is outlined. Next, a detailed report of the encryption and decryption process is presented. Then, the performance evaluation of the cipher against statistical tests and cyberattacks is described. Finally, the relationship between the proposed image encryption scheme and other algorithms is evidenced. This shows that the proposed cipher can perform image encryption and decryption on different devices, while others may not.

- **Chapter 4**: an overview of existing methods for estimating the Lyapunov exponent of chaotic systems is presented. It emphasizes that no approach directly determines the largest Lyapunov exponent using SPICE-like programs. The proposed method is then detailed, and the results of applying the method to six benchmark chaotic circuits are presented.

- **Chapter 5**: this chapter begins by describing the problem of designing digital chaotic circuits. It then provides a brief overview of the binary shift chaotic map, also known as the tent map. Next, the proposed method for designing this digital chaotic circuit is presented. The results of this method are then discussed, which show that the design consumes few logical resources while still exhibiting chaotic properties. Finally, an application of particle swarm optimization using the digital chaotic circuit is presented.

- **Chapter 6**: the last chapter brings the conclusion and final remarks about this document. It gives an overview, recapping the previous chapters and the main results concerning the new cryptosystem proposed, the Lyapunov exponent estimation method, and the digital chaotic map. Moreover, it delineates potential avenues for future research, laying the groundwork for the exploration of this field. Finally, a catalog of other articles closely related to this thesis, which were published or accepted for publication during the postgraduate study, is presented.

Chapter 2

Preliminary Concepts

2.1 Chaos and Characterization of Chaotic Dynamics

Numerous elements in the world demonstrate nonlinear characteristics to varying degrees, which is why there is significant interest in the study of nonlinear dynamical systems. Among the prevalent nonlinear phenomena observed in nature, chaos stands out. Initially observed in the context of the three-body problem [76], which involves the study of the interactions between three bodies under gravitational forces, chaotic behavior has subsequently been identified in various domains [9, 38, 142].

Chaos theory asserts that chaos is deterministic, implying that chaotic behavior adheres to established laws, with initial conditions and the principles of physics and mathematics governing these systems. Furthermore, chaotic systems exhibit sensitivity to initial conditions and parameters, such that even minor perturbations or uncertainties in these factors can significantly impact their long-term behavior. This is why chaotic systems are renowned for their seemingly random behavior and the challenge of accurately predicting their future dynamics [99].

Indeed, chaos is intriguing due to its deterministic yet unpredictable nature, which goes against the principle of causality. Chaos theory underscores that the reliability of the causality principle is not only limited by the uncertainty principle but also by the inherent instability of the underlying laws of nature [99].

Given the information provided above, it becomes essential to establish a mathematical definition of chaos and elucidate the fundamental principles governing such systems.

2.1.1 A Mathematical Definition of Chaos

Devaney's definition of chaos is popular and widely accepted. The definitions and results in this subsection were obtained from Devaney's work [25] and other relevant documents [7, 72], which outline three essential properties that characterize a chaotic system. According to Devaney, a dynamical system $f : X \to X$ is chaotic if:

- f is transitive,
- the periodic orbits of f are dense in the set X,
- f has sensitive dependence on initial conditions.

The notion of transitivity implies that for any nonempty subsets A and B of X, there exists a positive integer n such that $f^n(A) \cap B$ is nonempty. In simpler terms, it means that for any two points chosen from the domain of f, there exists an orbit that passes through these two points.

Density implies that one can find points within a set that are arbitrarily close to any point in the space. In mathematical words, for a set A and a subset B of A, B is considered dense in A if, for any point $a \in A$, there is a point $b \in B$ such that $||a - b|| < \epsilon$ for $\epsilon > 0$. Additionally, periodic points of a function f are points in the set X where applying the function f repeatedly leads back to the initial point. In simpler terms, a point $x \in X$ is considered a periodic point of f if there exists a positive integer n such that $f^n(x) = x$.

Devaney invokes the concept of sensitive dependence on initial conditions, which occurs when there are two initial conditions, x_0 and x_0', in close proximity, defined as $||x_0 - x_0'|| < \delta$ for $\delta > 0$, and in the n^{th} iteration, their respective orbits diverge by a distance greater than ϵ for $\epsilon > 0$, expressed as $||f^n(x_0) - f^n(x_0')|| \geq \epsilon$.

While Devaney's definition of chaos is widely recognized, it is crucial to consider alternative perspectives. Robinson [105] and Wiggins [135] offer differing definitions, notably omitting the condition related to the density of periodic orbits of f. In addition, Hunt and Ott [42] introduce a perspective by defining chaos based on an entropy-like quantity, despite other definitions that emphasize the sensitive dependence of initial conditions.

Demonstrating that a deterministic system adheres to Devaney's chaos definition or any other widely accepted definition can be quite challenging without the requisite mathematical tools. Consequently, scientists and engineers resort to alternative quantitative and qualitative measures, such as the cobweb plot, bifurcation diagram, and Lyapunov exponent, to describe the chaotic dynamics.

2.1.2 Lyapunov Exponent

The Lyapunov Exponent (LE) quantifies the rate at which nearby orbits of dynamical systems separate. Chaotic systems have at least one positive exponent in their strange attractor. The physical meaning of a positive Lyapunov exponent is that even nearby orbits, corresponding to nearly identical states, will diverge exponentially over time. Indeed, the formal definition of LE asserts that for any two trajectories at time t, denoted as $x_t = f^t(x_0)$ and $x_t + \delta x_t = f^t(x_0 + \delta x_0)$, which start in close proximity, they will exponentially separate. This separation follows the relationship $||\delta x_t|| \approx e^{LEt}||\delta x_0||$ [23].

Wolf et al.'s method [136] estimates the LE using a time series from dynamical systems, making it an advantageous approach that can be applied directly to experimental data. This iterative method involves determining a fiducial trajectory (a reference time-series data) and analyzing the points in the neighborhood of this trajectory to observe the rate of divergence. To achieve this, nearby trajectories are calculated, starting at a small initial distance from the fiducial trajectory. These nearby trajectories are iteratively adjusted so that after a time interval, their separation is renormalized back to a short distance from the fiducial trajectory while minimizing the orientation (angle) change relative to it. This process is repeated iteratively over the entire time series. The LE is then obtained from the average rate of exponential divergence between the fiducial trajectory and the nearby trajectories:

$$\text{LE} = \frac{1}{t_K - t_0} \sum_{k=1}^{K} \log_2 \frac{L'(t_k)}{L(t_{k-1})}, \tag{2.1}$$

where K denotes the total number of replacement steps, while L and L' represent the distances between trajectories in the initial and final steps, respectively.

Figure 2.1 illustrates the process of estimating the Lyapunov exponent using Wolf et al.'s method. Given the evolution time (t), the minimal and maximal initial distance of the orbits (L), and the angular separation (θ) between the evolved and replacement elements, the divergence of the trajectories is calculated between the fiducial trajectory (red line) and the other ones (blue lines). To ensure an accurate estimation of the Lyapunov exponent, it is essential to determine the appropriate embedding dimension and embedding delay, as the method relies on Takens's theorem [126]. Approaches such as the false nearest neighborhood method and mutual information [55] can be used to identify suitable values for these parameters. However, this process is complex and can lead to errors, such as underestimation of the LE.

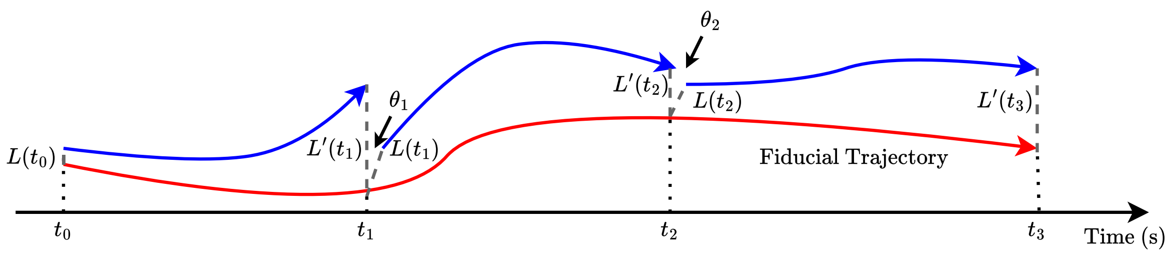


Figure 2.1: A representation of the procedure for estimating the Lyapunov exponent of a dynamical system based on Wolf et al.'s method [136].

2.2 Number Representation

Digital technology is indispensable in both simulating and implementing chaotic systems. Computers are routinely employed for simulating chaotic systems, while microcontrollers and other embedded devices find frequent use in the practical application of such systems. Hence, it is imperative to provide an overview of the number representations employed in digital devices. Presently, there are three key formats in use: fixed-point arithmetic, floating-point arithmetic, and posit arithmetic. This work focuses on the first two, with detailed explanations in the subsequent subsections.

2.2.1 Fixed-Point Arithmetic

To computationally represent systems that rely on real numbers, fixed-point representation can be employed. In fixed-point arithmetic, a number is divided into three components: the sign, integer, and fractional parts [95]. Figure 2.2 illustrates this using a 32-bit format — 1 bit for the sign, 15 bits for the integer, and 16 bits for the fractional part.

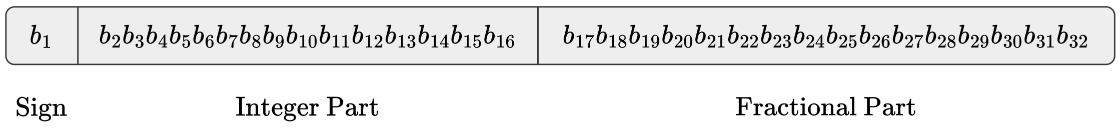

b_1
$b_2b_3b_4b_5b_6b_7b_8b_9b_{10}b_{11}b_{12}b_{13}b_{14}b_{15}b_{16}$
$b_{17}b_{18}b_{19}b_{20}b_{21}b_{22}b_{23}b_{24}b_{25}b_{26}b_{27}b_{28}b_{29}b_{30}b_{31}b_{32}$
Sign
Integer Part
Fractional Part

Figure 2.2: A 32-bit representation for fixed-point arithmetic.

Unlike its counterparts, this number representation boasts superior hardware performance at the cost of limited flexibility, enabling storage of a restricted range of values [31, 95]. For instance, as shown in Figure 2.2, it accommodates numbers within the range of 2^{-16} to 2^{15}.

2.2.2 Floating-Point Arithmetic and IEEE Standard 754-2019

Floating-point arithmetic, which began to see widespread use in the mid-1950s, is founded on exponential notation. In a digital system, a Floating-Point (FP) number is represented as:

$$\mathrm{FP} = (-1)^S \cdot Mn \cdot 2^E, \tag{2.2}$$

where S represents the sign bit, Mn stands for the mantissa, and E corresponds to the exponent part [95].

Unlike the fixed-point format, floating-point arithmetic has the remarkable ability to store an extensive range of numbers, rendering it indispensable in a wide array of real-world scenarios [31]. This unparalleled versatility prompted a historic collaboration between scientists and engineers, giving rise to the IEEE 754 floating-point standard [95].

The IEEE 754 floating-point standard establishes the formats, arithmetic operations, rounding modes, and handling of exceptions for floating-point numbers. Interested readers are recommended to consult the standard itself [43] and refer to Overton's book [95] for a more comprehensive explanation of how IEEE 754 floating-point works.

In terms of format, the IEEE 754 standard primarily utilizes two precision options: single and double precision. Figure 2.3 visually depicts the bit arrangement for single precision format, which consists of a 32-bit word length. In this arrangement, the most significant bit is designated as the sign bit, where "1" signifies negative numbers and "0" denotes positive numbers, following a similar convention to fixed-point arithmetic. Additionally, the 8-bit exponent in single precision has a bias of 127, while the remaining 23 bits are dedicated to the mantissa, and these mantissa bits are normalized. For double precision, characterized by a 64-bit word length, 1 bit is allocated for the sign, 11 bits are dedicated to the exponent, and the remaining 52 bits are reserved for the mantissa [43].

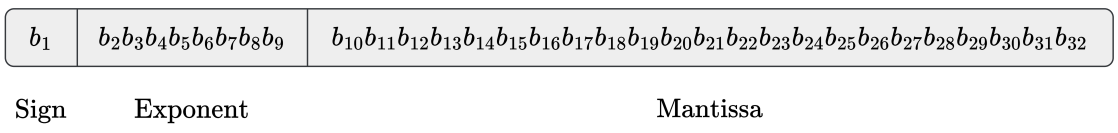

b_1
$b_2b_3b_4b_5b_6b_7b_8b_9$
$b_{10}b_{11}b_{12}b_{13}b_{14}b_{15}b_{16}b_{17}b_{18}b_{19}b_{20}b_{21}b_{22}b_{23}b_{24}b_{25}b_{26}b_{27}b_{28}b_{29}b_{30}b_{31}b_{32}$
Sign
Exponent
Mantissa

Figure 2.3: Single-precision floating-point format adopted by IEEE 754-2019.

Much like fixed-point arithmetic, the precisions employed in the IEEE 754-2019 floating-point standard are incapable of representing all real numbers. Consequently, simulations may exhibit round-off errors and yield varying results, as discussed in the previous chapter. To address these issues, potential solutions include the use of finite fields, an algebraic structure that avoids rounding errors, or the application of interval arithmetic, a mathematical approach that captures solutions within an interval while accounting for uncertainties.

2.3 Finite Field

Finite fields are a well-developed algebraic structure much used in several mathematical applications on communication problems. Numerous textbooks cover this branch of algebra. The definitions and results of this section were obtained from Lidl and Niederreiter's [60] and McEliece's [69] books. They were slightly altered in order to fit this context.

Henceforth, within the domain of finite field theory, the convention dictates the representation of two binary operations within a set through the symbols "+" and "·".

Definition 2.1. *Let G be a set and "·" a binary operation defined on G. $(G, \cdot)$ is a group if the following three properties hold:*

- *"·" is associative; that is, for each element a, b and $c \in G$, $(a \cdot b) \cdot c = a \cdot (b \cdot c)$;*

- *there is an identity element e in G such that for all $a \in G$, $e \cdot a = a \cdot e = a$;*
- *for each $a \in G$, there is an inverse element $a^{-1} \in G$ such that $a \cdot a^{-1} = a^{-1} \cdot a = e$.*

Moreover, if $a \cdot b = b \cdot a$, for all a, $b \in G$, then $(G, \cdot)$ is called an Abelian group.

Example 2.1. *$(\mathbb{R}, +)$ is an example of an Abelian group.*

Definition 2.2. *A field $(G, +, \cdot)$ is a set G with two binary operations, denoted by "$+$" and "$\cdot$", such that:*

- *G is an Abelian group under "$+$" with identity element 0;*
- *nonzero elements of G form an Abelian group under "$\cdot$";*
- *distributive laws $a \cdot (b + c) = a \cdot b + a \cdot c$ and $(b + c) \cdot a = b \cdot a + c \cdot a$ hold to all a, b and $c \in G$.*

Example 2.2. *Let $\mathbb{Z}_3$ be the set of integers modulo 3. Without loss of generality, it is represented its residue classes/elements as $\mathbb{Z}_3 = \{0, 1, 2\}$. Hence, the following Cayley tables are obtained, as Table 2.1 shows.*

Table 2.1: Cayley tables for the group $\mathbb{Z}_3$ with "+" on the left and operation "·" on the right, taken from mod 3.

$+$	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

$\cdot$	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

In general, $\mathbb{Z}_p$ is a finite field for a p prime. From now on, it is denoted each finite field with q elements as $GF(q)$ (Galois Field) and q is always a power prime, where $q = p^r$, for p a prime integer and $r \geq 1$.

Thereafter, given a positive integer $r > 1$, two different ways are presented to construct the finite field $GF(q^r)$ from $GF(q)$. For the next definition, $GF(q)[x]$ means the set (actually, ring) of all polynomials whose coefficients are on $GF(q)$. In this set, it is possible to define the addition and multiplication of polynomials, noticing the coefficients belong to $GF(q)$.

Definition 2.3. *A polynomial $f \in GF(q)[x]$ is said to be irreducible over $GF(q)$ if f has positive degree and $f = b \cdot c$, where b and $c \in GF(q)[x]$ implies that either b or c is a constant polynomial.*

In order to construct finite fields with more elements from $GF(q)$, the following result is considered:

Theorem 2.1. *For a polynomial $f \in GF(q)[x]$, the residue class ring $GF(q)[x]/(f)$ is a field if and only if f is irreducible over $GF(q)$.*

Proof. For a comprehensive proof, please refer to [60]. □

On $GF(q)[x]/(f)$, it considers the classical polynomial operations of addition "+" and multiplication "·" modulo f. Given a, $b \in GF(q)[x]/(f)$, if the product $a \cdot b$ has a degree greater than or equal to the degree of f, then it is replaced by its remainder when divided by f. Furthermore, the polynomial addition operates as usual.

Example 2.3. *Let $GF(2)[x]/(x^2+x+1)$ be the residue class ring. Hence, the only elements (residue classes) in $GF(2)[x]/(x^2+x+1)$ are 0, 1, x and $x+1$. Since $f = x^2+x+1$ is irreducible over $GF(2)$, then $GF(2)[x]/(f)$ is actually a finite field, as shown in Table 2.2.*

Table 2.2: Polynomial addition and multiplication on $GF(2)[x]/(f)$. The coefficients are taken mod 2. In particular, the polynomial multiplication is equivalent to its remainder when divided by f.

$+$	0	1	x	$x+1$
0	0	1	x	$x+1$
1	1	0	$x+1$	x
x	x	$x+1$	0	1
$x+1$	$x+1$	x	1	0

$\cdot$	0	1	x	$x+1$
0	0	0	0	0
1	0	1	x	$x+1$
x	0	x	$x+1$	1
$x+1$	0	$x+1$	1	x

Therefore, $GF(2)[x]/(f) = \{0,\ 1,\ x,\ x+1\}$ is a $4-$element finite field.

Additionally, it is possible to employ matrices in order to represent elements of a finite field $GF(q)$.

Definition 2.4. *The companion matrix of a polynomial $f = a_0+a_1x+\ldots+a_{m-1}x^{m-1}+x^m \in GF(q)[x]$ is defined to be:*

$$A_{m\times m} = \begin{pmatrix} 0 & 0 & 0 & \ldots & 0 & -a_0 \\ 1 & 0 & 0 & \ldots & 0 & -a_1 \\ 0 & 1 & 0 & \ldots & 0 & -a_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & 1 & -a_{m-1} \end{pmatrix}_{m\times m} . \tag{2.3}$$

It is well-known that:

$$f(A) = a_0 I_{m\times m} + a_1 A_{m\times m} + a_2 A^2_{m\times m} + \ldots + a_{m-1} A^{m-1}_{m\times m} + A^m_{m\times m} = 0_{m\times m}, \tag{2.4}$$

where $I_{m\times m}$ denotes the identity matrix and $0_{m\times m}$ the null matrix. Thus, given $f \in GF(q)[x]$ irreducible of $m-$degree over $GF(q)$, and $A_{m\times m}$ its corresponding $m\times m$ companion matrix, the finite field $GF(q^m)$ may also be represented as:

$$GF(q)[A] = \{a_0 I_{m\times m} + \ldots + a_{s-1}A^{s-1}I_{m\times m} : a_i \in GF(q), 0 \leq i \leq s-1\}, \tag{2.5}$$

whose usual addition and product matrices are developed regarding the condition $f(A) = 0_{m\times m}$.

From the two finite field representations above, it is possible to connect them as:

$$g \in GF(q)[x]/(f) \leftrightarrow g(A) \in GF(q)[A]. \tag{2.6}$$

Thus, it is possible to identify binary representations of real numbers as polynomials in $GF(2)[x]$ and, from exposed in Equation (2.6), as matrices over $GF(2)$.

2.4 Interval Arithmetic

The contemporary method of interval arithmetic has gained prominence within the scientific community, largely due to Moore's book [73]. Finite precision in digital computers introduces round-off errors, and interval arithmetic offers a rigorous means to encapsulate solutions for modeling equations. This approach effectively constrains the exact solutions of both linear and nonlinear systems that involve uncertainties [74].

At the core of interval arithmetic lies the concept of an interval, defined as follows [73, 74]:

Definition 2.5. *An interval is a closed set of a real number $x \in \mathbb{R}$ represented by $[\underline{X}, \overline{X}] = \{x : \underline{X} \leq x \leq \overline{X}\}$, where $\underline{X}$ is the lower bound and $\overline{X}$ is the upper bound of the interval.*

In the context of intervals, the four fundamental arithmetic operations hold as follows:

- Addition: $[\underline{X} + \underline{Y}, \overline{X} + \overline{Y}]$.
- Subtraction: $[\underline{X} - \overline{Y}, \overline{X} - \underline{Y}]$.
- Multiplication: $[\min(\underline{X}\cdot\underline{Y}, \underline{X}\cdot\overline{Y}, \overline{X}\cdot\underline{Y}, \overline{X}\cdot\overline{Y}), \max(\underline{X}\cdot\underline{Y}, \underline{X}\cdot\overline{Y}, \overline{X}\cdot\underline{Y}, \overline{X}\cdot\overline{Y})]$.
- Division: $\left[\min\left(\frac{\underline{X}}{\underline{Y}}, \frac{\underline{X}}{\overline{Y}}, \frac{\overline{X}}{\underline{Y}}, \frac{\overline{X}}{\overline{Y}}\right), \max\left(\frac{\underline{X}}{\underline{Y}}, \frac{\underline{X}}{\overline{Y}}, \frac{\overline{X}}{\underline{Y}}, \frac{\overline{X}}{\overline{Y}}\right)\right], \nexists\, 0 \in [\underline{Y}, \overline{Y}]$.

Moreover, to understand the functioning of a simulation for a determined system using interval arithmetic, consider the following definitions [88]:

Definition 2.6. *An orbit is a sequence of n values, denoted as $\{x_n\} = [x_0, x_1, x_2, x_3, \ldots, x_n]$, generated by the system.*

Definition 2.7. *A pseudo-orbit, denoted as* $\{\hat{x}_{i,n}\} = [\hat{x}_{i,0}, \hat{x}_{i,1}, \ldots, \hat{x}_{i,n}]$*, serves as an approximation of the true orbit and must satisfy the condition:*

$$|x_n - \hat{x}_{i,n}| \leq \psi_{i,n}, \tag{2.7}$$

where $\psi_{i,n} \in \mathbb{R}^+$ *is the error. A pseudo-orbit is created as a result of computer finite-precision.*

In this way, a pseudo-orbit is just an interval in which the true orbit exists. Equation (2.8) will now give an interval associated with each estimation of a pseudo-orbit:

$$I_{i,n} = [\hat{x}_{i,n} - \psi_{i,n}, \hat{x}_{i,n} + \psi_{i,n}], \tag{2.8}$$

Determining the error of the pseudo-orbit and, consequently, the interval comprising the true orbit is a complex task. Therefore, Nepomuceno and Martins [88] introduced the lower-bound error as a metric in numerical simulations to estimate error-bound propagation.

2.4.1 Interval Extensions and the Lower-Bound Error

Before defining the lower-bound error, it is crucial to establish interval extensions [74]:

Definition 2.8. *Let* f *be a function of real variable* x*, a natural interval extension of* f *is an interval-valued function* F *of an interval variable, with the property* $F(x) = f(x)$*.*

Considering again Chua's circuit equations [19, 20]:

$$\begin{cases} \dot{x} &= 15.6(y - x - f(x)) \\ \dot{y} &= x - y + z \\ \dot{z} &= -28y, \end{cases} \tag{2.9}$$

$$f(x) = -0.714x - 0.2145(|x+1| - |x-1|). \tag{2.10}$$

Example 2.4. *Examine an example of such interval extensions from Chua's circuit:*

$$\dot{x} = 15.6(y - x - f(x)), \tag{2.11}$$

$$\dot{x} = 15.6y - 15.6x - 15.6f(x). \tag{2.12}$$

Although Equations (2.11) and (2.12) are mathematically equivalent, the sequence of their basic arithmetic operations differs. The floating-point standard [43, 95] does not adhere to the commutative, associative, or distributive properties. As a result, when Equations (2.11) and (2.12) are estimated using a computer-based approach with the same initial conditions, the results may differ, generating two different pseudo-orbits.

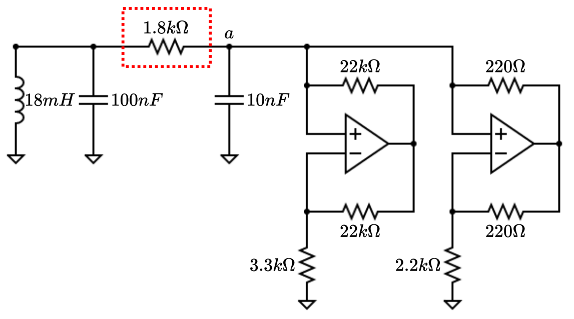
Now, proceeding to formally define the lower-bound error in the following theorem:

Theorem 2.2. *Let be $\{\hat{x}_{a,n}\}$ and $\{\hat{x}_{b,n}\}$ two pseudo-orbits derived from two interval extensions. The lower-bound error of such a system is $\psi_{\alpha,n} = \dfrac{|\hat{x}_{a,n} - \hat{x}_{b,n}|}{2}$, where either $\psi_{a,n} \geq \psi_{\alpha,n}$ or $\psi_{b,n} \geq \psi_{\alpha,n}$.*

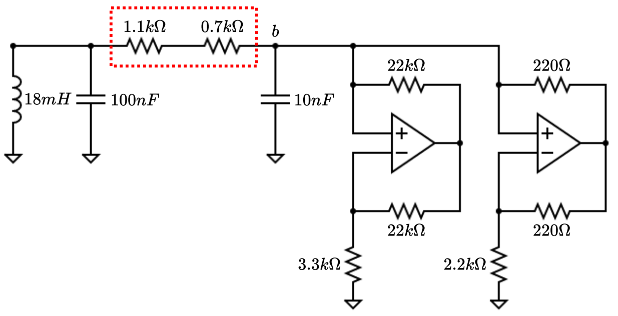
Proof. For a comprehensive proof, please refer to [88]. □

This theorem establishes that, within a pair of pseudo-orbits, at least one must exhibit an error equal to or greater than the lower-bound error. Furthermore, the lower-bound error serves as a metric for gauging the difference between the simulated dynamical system, the pseudo-orbit, and the true orbit.

The behavior exhibited through interval extensions from Chua's circuit equations is also observable in circuits. Considering two equivalent Chua's circuits depicted in Figure 2.4, it is essential to note that Chua's circuits 1 and 2 share the relationship $1.8k\Omega = 1.1k\Omega + 0.7k\Omega$. Although these circuits are equivalent and share the same initial conditions, their simulations in a digital environment may produce different results. According to LTspice simulations, the chaotic orbits of the circuits depicted in Figure 2.4 diverge over time, as shown in Figure 2.5.



(a) Chua's circuit 1.



(b) Chua's circuit 2.

Figure 2.4: The autonomous Chua's circuit. Due to the drawbacks of the floating-point standard, the simulation of these two equivalent circuits may diverge.

This leads to the following observation:

Remark 2.1. *The finite precision inherent in digital devices introduces significant limitations. With regards to the IEEE 754-2019 floating-point standard, even minor variations in the circuit's schematic representation can impact the entire numerical simulation of its behavior. This is due to the inherent inability of computers to accurately represent all real numbers. Such challenges are akin to those encountered in simulating interval extensions.*

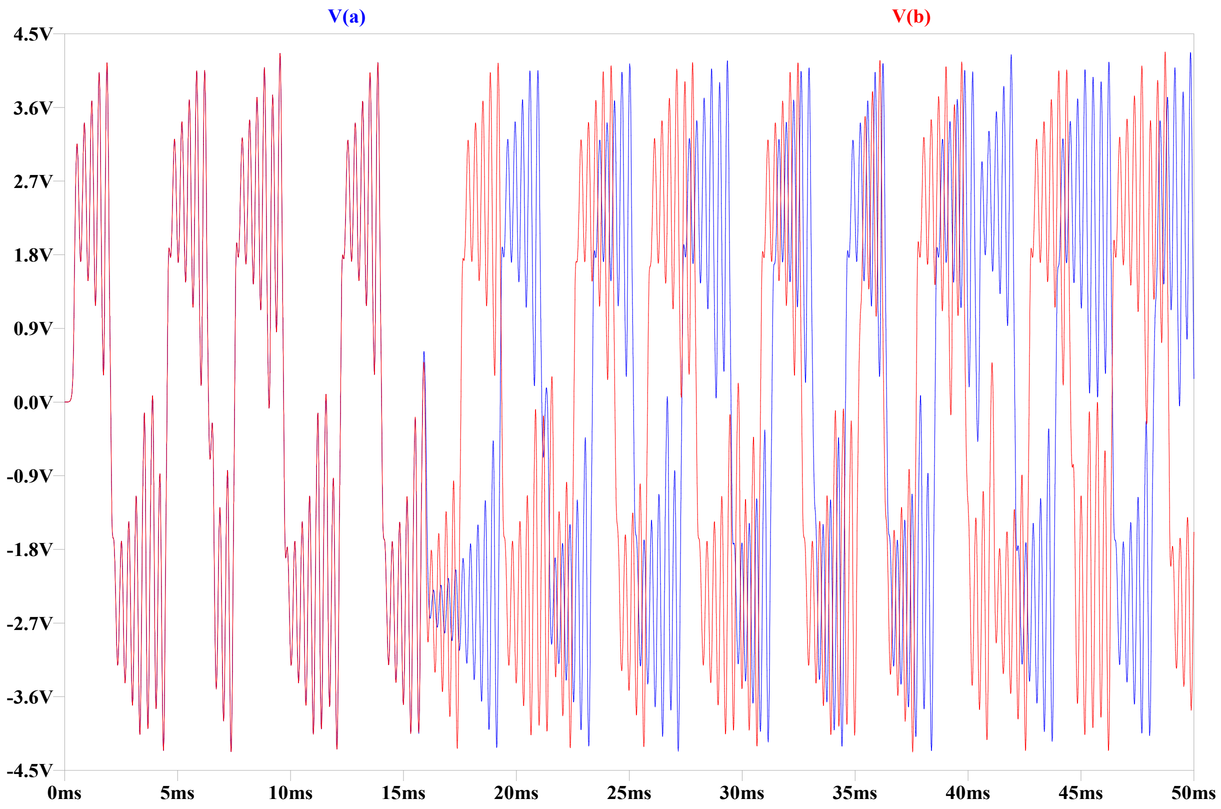


Figure 2.5: Divergence in the simulation of two equivalent Chua's circuits. When simulating the voltages at nodes a and b of the circuits shown in Figure 2.4, the orbits eventually diverge. `Schematic: Divergence in Circuits / chua_divergence.asc`.

Chapter 3

Chaos-Based Cryptosystem

Part of the work in this chapter has been published in the following: Nardo, L. G., Nepomuceno, E. G., Bastos, G. T., Santos, T. A., Butusov, D. N., and Arias-Garcia, J. (2021). A reliable chaos-based cryptography using Galois field. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(9):091101.

3.1 State-of-the-Art and Problem Description

The vast information flow on the internet necessitates efficient and reliable encryption algorithms. This has led to attention from the scientific community, resulting in the investigation and development of various approaches, such as AES (Advanced Encryption Standard), Blowfish, DES (Data Encryption Standard), RSA, Twofish, and chaos-based ones [66].

Chaos-based encryption algorithms have attracted considerable attention, particularly in the realm of image encryption schemes. Their superior performance, when contrasted with traditional encryption algorithms, makes them an intriguing and promising choice for securing digital images [56, 133]. Indeed, since Fridrich's pioneering work [30], numerous chaotic image encryption schemes have been suggested. For instance, Gao and Chen [33] proposed a cryptosystem that employs the logistic map to shuffle the image and Chen's hyperchaotic system to encrypt the respective image. Differently, Haroun and Gulliver [39] exploited the three-dimensional discrete Lorenz system as the chaotic element in the encryption algorithm. Taking advantage of FPGA fast processing, Zheng et al. [145] applied an n-dimensional discrete chaotic system to encrypt multi-images. In addition to the chaotic system, Rodríguez-Orozco et al. [106] used voice recognition as an access key, providing more security to the cryptosystem. Furthermore, some papers use metaheuristic optimization methods, such as firefly algorithm [29], to ensure the cipher security [2]. Sinha and Sahu [120] investigated an image encryption algorithm with shuffling, rotation, and swamping processes, where Chebyshev polynomial with adaptive firefly technique is used in order to optimize the generation of the secret key.

Özkaynak [148] reviewed chaos-based image encryption techniques and criticized many approaches, arguing they are weak against certain cyberattacks. Moreover, Özkaynak showed a checklist indicating what chaos-based encryption algorithms must comprise. At the same time, other approaches have been developed to improve chaos-based encryption algorithms, such as diffusion and confusion methods to increase cipher security [102], and perturbation processes to mitigate dynamical degradation [59]. Nevertheless, there is little attention on the effect of finite precision in different digital devices and its influence on chaos-based encryption algorithms. In fact, as previously shown in Chapter 1, chaotic systems may present contrasting behaviors in different software and hardware setups.

The application of finite fields in cryptography has been investigated as a possible way to tackle issues related to finite precision. Wang et al. [130] proposed an encryption algorithm based on Galois field that ensures the security of tumor ultrasound images during the transmission. Shah and Shah [115] recognized a remarkable impact of the arithmetic properties of a finite field on the security features of symmetric and asymmetric cryptosystems. Shah et al. [116] applied finite fields to a color image encryption application. The results, according to the authors, outperformed previous chaos-based encryption schemes. Moreover, Broumandnia [11] described an algorithm adopting finite fields in chaotic maps. That paper focused on improving the security and speed of encryption using Galois fields simultaneously in the diffusion and confusion operations.

As previously mentioned, a literature review shows there are few papers focused on the effects of finite precision in digital devices and its influence on chaos-based encryption algorithms. In other words, chaos-based encryption schemes require reproducibility in different digital devices, but floating-point standards, such as the IEEE standard for floating-point arithmetic [43] does not fulfill this requirement, which may result in distinct ciphers. To address this challenge, this chapter sheds light on this problem and introduces a new cryptosystem based on a chaotic system and the Galois field to compose a pseudo-random number generator. The chaotic system chosen is the logistic map [68]. The great advantage of the proposed technique is that all arithmetic operations occur in one field. This means there is no need for rounding. As a consequence, the reproducibility of the operations is not influenced by different digital devices and the secret key to encrypt and decrypt are the same. Moreover, utilizing matrices over $GF(2)$ to depict binary representations of real numbers, the proposed cryptosystem requires fewer iterations for encrypting images than the alternative current methods.

The remainder of this chapter is presented as follows: the encryption algorithm based on finite fields and chaos theory is shown in Section 3.2. The performance analysis of the cryptosystem is shown in Section 3.3. Ultimately, Section 3.4 reports on the implications of the results shown, as well as final remarks about the encryption scheme.

3.2 Image Encryption Scheme

The proposed scheme for generating the keystream can be summarized in the following steps. This scheme involves creating a keystream using a chaotic system and finite fields theory, matching the dimensions of the image to be encrypted. Each part of the keystream represents 8 bits of data, corresponding to each pixel in the image.

It is important to note that this cryptosystem adheres to Kerckhoffs' principle, which states that the security of a cryptosystem should depend only on the secrecy of the key, not the algorithm itself [75]. The steps were described using standard MATLAB algorithms applicable to all digital devices. However, this chaos-based image encryption algorithm can be implemented in any programming language.

The steps are as follows:

- **Step 1**: Read the plain image (P_I) with size $H \times W$, where H and W are the height and width of the image, respectively. Each pixel in the image is represented by an 8-bit unsigned integer.

- **Step 2**: Split the plain image in two blocks (bl_1 and bl_2) with size $H \times \frac{W}{2}$.

- **Step 3**: For each block, compute a factor ($f_{bl_{1,2}}$) defined as:

$$f_{bl_{1,2}} = \frac{1}{256\left(H \cdot \frac{W}{2}\right)} \sum_{i=1}^{H} \sum_{j=1}^{W/2} bl_{1,2}(i,j), \tag{3.1}$$

  where i and j are the coordinates of each pixel in the blocks.

- **Step 4**: Convert the two factors, previously obtained, to binary numbers as:

$$c_{1,2} = \text{dec2bin}\left(\frac{f_{bl_{1,2}}}{eps}\right), \tag{3.2}$$

  where eps is the machine epsilon and dec2bin is an algorithm to convert decimal integers to binary numbers.

- **Step 5**: For each binary number c_1 and c_2, use only the first 32 bits. Thereafter, concatenates to each other as:

$$f_{P_I} = \text{strcat}(c_1, c_2), \tag{3.3}$$

  where strcat is an algorithm that concatenates strings horizontally. Thus, the factor f_{P_I} is dependent on the plain image to be encrypted.

- **Step 6**: Choose 64-bit binary numbers for the initial condition x_0 and the bifurcation parameter λ of the logistic map. Both numbers are represented using 2Q62 fixed-point format.

  For friendly-user applications, a sequence of 8 ASCII encoded characters may be required and easily changed into a 64-bit string. This method allows an initial condition of up to 4096 bits. Moreover, it is important to mention each bit of these numbers is allocated separately in an array; thus, f_{P_I}, x_0, and λ are 1×64 vectors.

- **Step 7**: Take a 64^{th}-degree irreducible polynomial p. In this cryptosystem, the irreducible polynomial $p \in GF(q)[x]$ was chosen as[1]:

$$p(x) = x^{64} + x^{63} + x^8 + x^2 + 1, \tag{3.4}$$

  and an $m \times m$ companion matrix P is then made. Given that $p(x)$ is a 64^{th}-degree polynomial, it follows that P is represented by a 64×64 matrix.

- **Step 8**: In order to obtain a finite field representation of f_{P_I}, x_0, and λ, do a 64-time iteration of these following equations:

$$F_{P_I} = (F_{P_I} + f_{P_I}(65 - k) \cdot P^{k-1}) \bmod 2, \tag{3.5}$$
$$X_0 = (X_0 + x_0(65 - k) \cdot P^{k-1}) \bmod 2, \tag{3.6}$$
$$\Lambda = (\Lambda + \lambda(65 - k) \cdot P^{k-1}) \bmod 2, \tag{3.7}$$

  where mod is the modulo operation, k is the respective iteration and F_{P_I}, X_0, and Λ are 64×64 matrices, which are $GF(2)[P]$ representations of f_{P_I}, x_0, and λ, respectively.

- **Step 9**: Merge the user initial condition X_0 and the factor F_{P_I} as:

$$X_0' = (F_{P_I} + X_0) \bmod 2. \tag{3.8}$$

  In such a manner, the initial condition X_0 of the logistic map and the plain image factor F_{P_I} are linked, making the system resistant to differential attacks.

- **Step 10**: From the original logistic map [68], a recursive function was redefined using a representation as matrices over $GF(2)$:

[1]In this proposed scheme, it is imperative to use a 64^{th}-degree irreducible polynomial. This is because the plain image factor, the initial condition, and the bifurcation parameter of the logistic map are all 64-bit strings. In this way, a 64×64 matrix representation of these parameters will be obtained from the companion matrix derived from the chosen irreducible polynomial.

$$X_{n+1} = (\Lambda \cdot X_n(1 - X_n)) \bmod 2. \tag{3.9}$$

Equation (3.9) should be iterated $\frac{H \cdot W}{64 \cdot 8}$ times. As each image pixel is represented by an 8-bit unsigned integer, the polynomial $p(x)$ has 64-degrees and, since it is worked here with 512×512 images, the number of iterations is $\frac{512 \cdot 512}{64 \cdot 8} = 512$. In such a situation, X is a three-dimensional matrix of size $64 \times 64 \times 512$.

Usually, chaos-based image encryption schemes iterate the algorithm $H \cdot W$ times. Thus, an important feature of this cryptosystem is the fact that its computational complexity is reduced.

- **Step 11**: Format the three-dimensional binary matrix X to a two-dimensional matrix with size $(H \cdot W) \times 8$, as:

$$Key = \text{reshape}(X, [(H \cdot W), 8]), \tag{3.10}$$

  where reshape is an algorithm that reshapes an array.

- **Step 12**: Thenceforth, convert the Key matrix to decimal values by multiplying Key with an array $2^{n_{array}}$, where $n_{array} = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{bmatrix}^T$. In such a way, the keystream is $Key \in [0, 255]$.

- **Step 13**: Reshape matrix Key into one which has the same height and width of image to be encrypted as:

$$Key = \text{reshape}(Key, [H, W]). \tag{3.11}$$

- **Step 14**: Encrypt the plain image P_I using the keystream Key and the bitwise XOR operation with all the pixels, as shown in Equation (3.12). The result is the cipher image C_I.

$$C_I(i, j) = P_I(i, j) \oplus Key(i, j). \tag{3.12}$$

The encryption algorithm is described in a flowchart diagram as seen in Figure 3.1. The decryption process, as is well-known, is the reverse of encryption. It can be done by applying **Step 14** again in the cipher image.

3.3 Performance Analysis

A series of eight tests were conducted to analyze the security and efficiency of the proposed encryption scheme, namely: key space, ENT test suite, correlation of adjacent pixels,

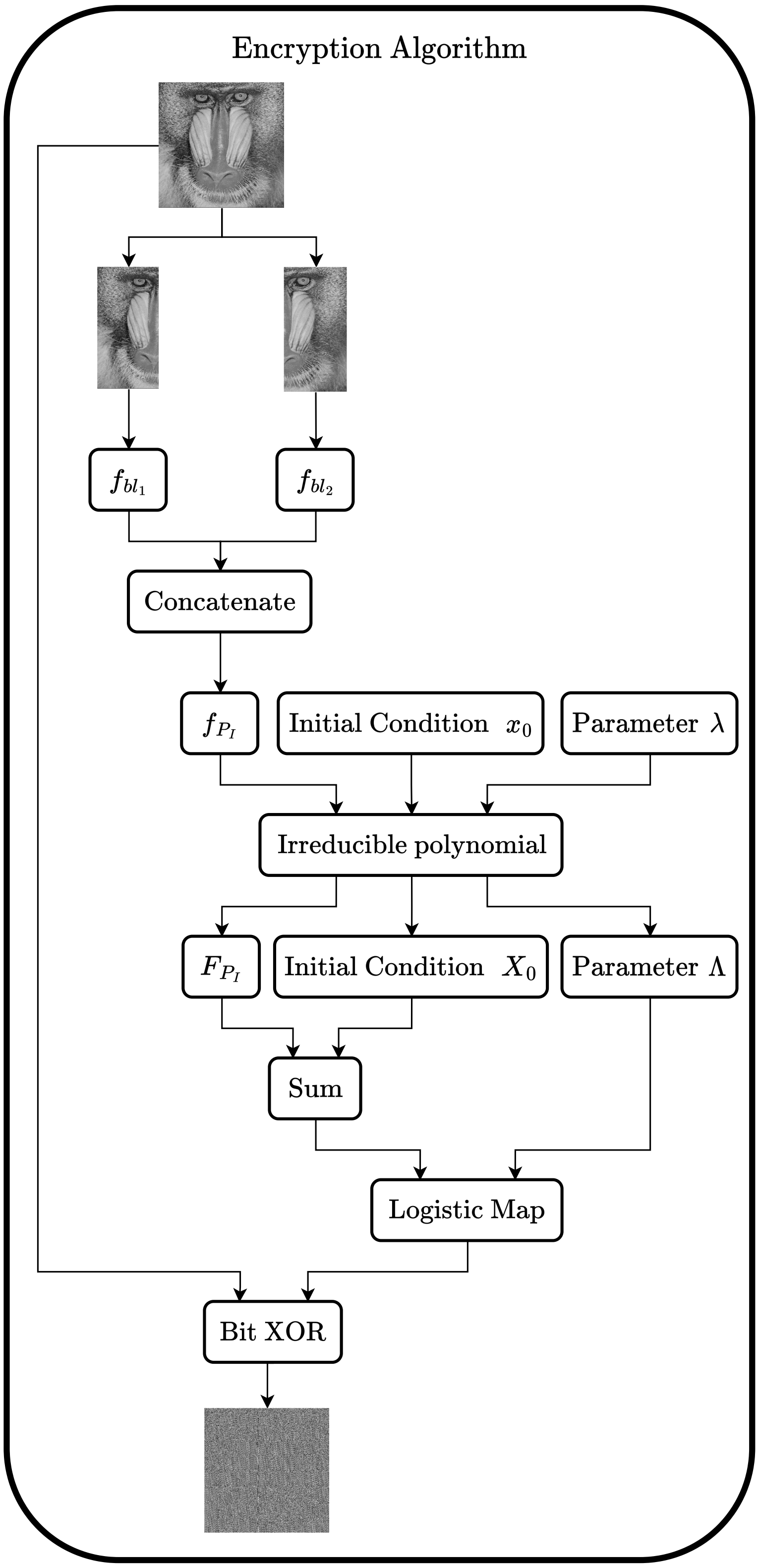


Figure 3.1: Flowchart diagram of the algorithm based on the logistic map and Galois field. The novelty lies in its orderly encryption process, which can be consistently reproduced on different machines, unlike many chaos-based cryptosystems that neglect the impact of finite precision in computers. This scheme considers Kerckhoffs' principle [75], which increases its robustness.

information entropy, histogram analysis, key sensitivity analysis, differential attack analysis, and reproducibility. Additionally, the 512×512 Baboon, Elaine, and Pepper were used to perform such tests. The parameters, as well as the factors, dependent on the plain images, which were used to generate the keystream, are shown in Table 3.1.

Table 3.1: Parameters and the plain image factors f_{P_I}, which were used to perform the encryption algorithm. Binary representations are changed into a matrix over $GF(2)$ before being used in the adapted logistic map. The binary representation uses the 2Q62 fixed-point format.

Parameter	**Decimal Representation**	**Binary Representation**
λ	3.99	11.11111101011100001010001111010111000010100011110101110000101000
x_0	0.10	00.00011001100110011001100110011001100110011001100110011001100110
f_{Baboon}	2.041025400639708	10.0000101010000000010100100000000010000011011000011100001100000000
f_{Elaine}	2.283803106275167	10.0100100010100111010100100000000111111010001111110100000000000000
f_{Pepper}	3.733667851371541	11.1011101111101000110101000000000011110001000111000110100000000000

3.3.1 Key Space

A brute-force attack on image encryption involves using trial and error to attempt all possible keystreams until the correct one is found and the image is decrypted. It is beneficial for an encryption scheme to have as large a key space as possible, but an encryption scheme with a key space larger than 2^{100} is already considered secure against such attacks [41, 93].

In the proposed approach, the seed is represented in a 64×64 matrix, which implies an impressive key space of up to 2^{4096}. In this case, this was achieved by using the initial condition (**Step 6**) and an irreducible polynomial (**Step 7**). Alternatively, the irreducible polynomial can be fixed and public, and an initial condition of 4096-bit may be required. This can be done by a string of 256 characters using the UTF-16 encoding (MATLAB standard).

3.3.2 ENT Test Suite

The generated keystream Key (Equation (3.11)) was analyzed with the help of the ENT test suite [128], a tool that has been extensively used [17, 125]. By running 6 different statistical tests, this series of tests present a satisfactory indicator of quality for encryption methods, as it can identify pseudo-random features in a sequence. Starting with an input sequence with a bitstream length of 600000 bits, Table 3.2 shows the result for each test. Since the input sequence passed all tests, the generated keystream in this scheme delivers acceptable pseudo-random properties.

Table 3.2: The results of the ENT test suite for the proposed cryptosystem. `Document tested: bit_cryptosystem.bin`.

Statistical Test	Test Output	Decision
Entropy	0.99999 bits per bit.	✓
Optimum compression	Optimum compression would reduce the size of this 600000 bit file by 0%.	✓
Chi-square	Chi square distribution for 600000 samples is 0.52, and randomly would exceed this value 47.13% of the times.	✓
Arithmetic Mean	0.5005 (0.5 = random).	✓
Monte Carlo Value for π	3.16992 (error 0.90%).	✓
Serial Correlation Coefficient	0.00052 (totally uncorrelated = 0.0).	✓

3.3.3 Correlation of Adjacent Pixels, Entropy, and Histogram Analysis

The correlation of adjacent pixels, information entropy, and histogram analysis are important measures to ensure the quality of a cryptosystem. In a plain image, the correlation among adjacent pixels is high, close to one. Its histogram is non-uniform and, as the plain image shows information, its entropy is low, indicating low disorder. Otherwise, noise-like images have a low correlation coefficient, close to zero. Its histogram is uniform and the entropy is approximately 8, implying high disorder [41, 66, 93].

The correlation coefficients are given by [26]:

$$\rho(x,y) = \frac{E[(\{x\}-\mu_x)(\{y\}-\mu_y)]}{\sigma_x\sigma_y}, \tag{3.13}$$

where $\{x\}$ is the series of pixels in a determined position, $\{y\}$ is the series of adjacent pixels of $\{x\}$, $E[\cdot]$ is the expected value, $\mu_{x,y}$ and $\sigma_{x,y}$ is the mean and standard deviation for $\{x\}$ and $\{y\}$, respectively.

Moreover, information entropy of images is calculated by [66]:

$$J(x) = \sum_{i=0}^{2^8-1} Pr(x_i) log_2 \frac{1}{Pr(x_i)}, \tag{3.14}$$

where $J(x)$ is the entropy in bits, x is an input and $Pr(x)$ is the probability of x.

Figure 3.2 exhibits the plain images and the cipher images, followed by their respective histograms. The proposed cryptosystem is capable of encrypting images efficiently. Furthermore, the histograms display a decrease of pixel variance between the plain and the noise-like images. Table 3.3 shows the entropy and correlation coefficients for each tested plain and cipher image. The obtained results are as expected, with a correlation coefficient roughly zero and information entropy close to 8 bits for cipher images.

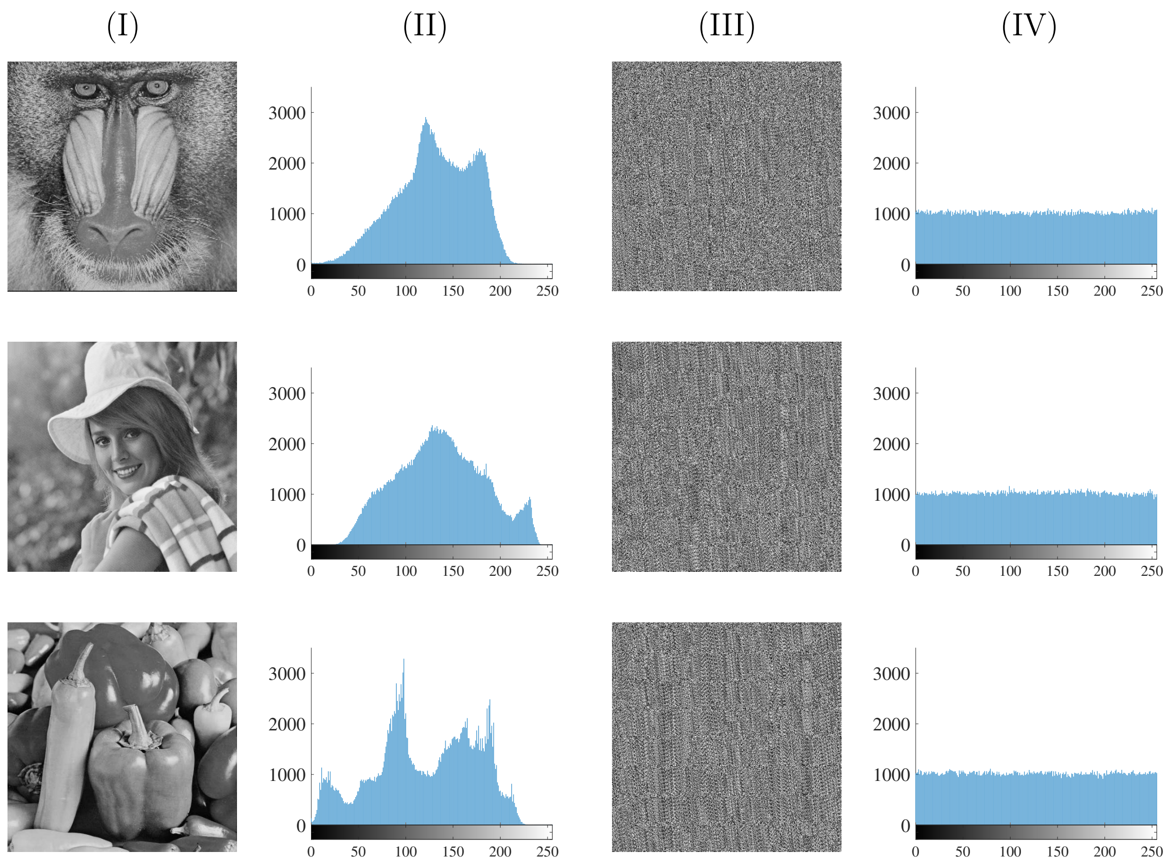


Figure 3.2: Plain and encrypted images, followed by their respective histograms. The image encryption scheme is able to encrypt meaningful images to noise-like ones. The histograms are shown in second and fourth columns for the plain and encrypted images, respectively. `Algorithm: Image Encryption / GFcipher.m`.

Table 3.3: Correlation coefficients and information entropy for each test image. As expected, the correlation coefficient in all directions are near zero for encrypted images. In addition, encrypted images present an entropy $J(x) \approx 8$, implying high disorder. `Algorithm: Image Encryption / GFcipher.m`.

Image	**Correlation Coefficient**			**Entropy** (*bits*)
	Horizontal	**Vertical**	**Diagonal**	
Baboon (plain image)	0.8700	0.8292	0.8002	7.3050
Baboon (cipher image)	0.0075	-0.0005	-0.0038	7.9992
Elaine (plain image)	0.9726	0.9696	0.9667	7.5130
Elaine (cipher image)	-0.0037	0.0150	-0.0056	7.9991
Pepper (plain image)	0.9812	0.9837	0.9663	7.5952
Pepper (cipher image)	0.0114	0.0272	-0.0038	7.9992

3.3.4 Key Sensitivity and Differential Analysis

Key sensitivity and differential analysis measure the effect on the keystream based on a minor change in the initial conditions, and the image that will be encrypted, respectively. Meanwhile, key sensitive analysis indicates whether the cryptosystem can have numerous possible keystreams, indicating robustness to this scheme, the differential attack analysis indicates whether the encryption algorithm is safe against this type of attack [32].

First of all, in efficient encryption algorithms, even a small change in the initial conditions impacts the generation of the keystream, resulting in a cipher image that is very different from the one obtained with an unperturbed keystream. In this way, key sensitivity analysis is accomplished by disturbing either the initial condition x_0 by 10^{-14} or f_{P_I} by 10^{-14}. The encryption process is then performed after this disturbance to produce the cipher image C_{I2}. The difference between the cipher images is quantified by [143]:

$$\text{Diff}_1 = \frac{1}{H \cdot W} \sum_{i=1}^{H} \sum_{j=1}^{W} |sgn(C_{I1}(i,j) - C_{I2}(i,j))| \cdot 100\%, \tag{3.15}$$

where H and W are the height and width of the cipher-images C_{I1} (with no perturbed values) and C_{I2} (with perturbed values).

From the keystream obtained via disturbance, the decryption process is performed using the cipher-image C_{I1}, obtaining the plain image P_{I2}. Equation (3.16) computes the difference between the plain images P_{I1} and P_{I2} [143]:

$$\text{Diff}_2 = \frac{1}{H \cdot W} \sum_{i=1}^{H} \sum_{j=1}^{W} |sgn(P_{I1}(i,j) - P_{I2}(i,j))| \cdot 100\%. \tag{3.16}$$

Table 3.4 presents differences obtained from distinct keystreams encrypting the Baboon image. Results show the difference is nearly 100%, indicating the outcome is dissimilar.

Table 3.4: Results of key sensitivity analysis for Baboon image. With a slight disturbance in the initial conditions, the outcome is absolutely dissimilar compared to the result obtained with an initial condition without disturbance. `Algorithms: Image Encryption / GFcipher_sensitivity.m and GFcipher_sensitivity2.m`.

Secret Key	**Diff₁**(%)	**Diff₂**(%)
$0.1 + 10^{-14}$	99.54	99.54
$f_{P_I} + 10^{-14}$	99.60	99.60

The Number of Pixels Change Rate (NPCR) and the Unified Average Changing Intensity (UACI) are the two equations to quantify the system's vulnerability to differential attack. To execute such analysis, two identical plain images are encrypted, where a randomly chosen pixel of one of the two images is slightly modified by one. These equations are described by Equations (3.17)-(3.18), respectively [32]:

$$\text{NPCR} = \frac{1}{H \cdot W} \sum_{i=1}^{H} \sum_{j=1}^{W} |sgn(C_{I1}(i,j) - C_{I2}(i,j))| \cdot 100\%, \tag{3.17}$$

$$\text{UACI} = \frac{1}{H \cdot W} \sum_{i=1}^{H} \sum_{j=1}^{W} \frac{|C_{I1}(i,j) - C_{I2}(i,j)|}{255} \cdot 100\%. \tag{3.18}$$

By examining these equations, we can observe that NPCR quantifies the number of pixels whose values have changed between the two encrypted images, whereas UACI calculates the average difference between the encrypted images. For secure image encryption, it is generally recommended that NPCR values surpass 99% while aiming for UACI values around 33% to ensure optimal performance [137].

Table 3.5 presents the NPCR and UACI indices from Baboon, Elaine, and Pepper images. As expected, in accordance with the criteria established in [137] for such indices, the proposed encryption algorithm is secure against differential attack.

Table 3.5: NPCR and UACI results for the proposed encryption scheme. `Algorithm: Image Encryption / GFcipher_npcr_uaci.m`.

Image	**NPCR (%)**	**UACI (%)**	**Decision**
Baboon	99.68%	33.36%	✓
Elaine	99.59%	33.51%	✓
Pepper	99.57%	33.47%	✓

3.3.5 Reliability on Different Devices

The main feature of the proposed cryptosystem is solving a recurrent problem when dealing with chaos-based encryption algorithms, where the user may be unable to decrypt the data if the device used has a different hardware and/or software setup than the one used for the encryption process [112]. Common chaos-based methods apply the normalized raw output of a chaotic system to perform the encryption. Such methods fail in real cases, where users need to exchange encrypted images and be able to decode them in contrasting devices. This is due to finite-precision errors, present in computers [43, 95]. Thus, distinct processors and software may have slightly diverse ways to calculate the same expression, which is a problem when modeling such systems.

This method is robust against hardware discrepancy. Representing binary numbers as elements of a finite field when iterated with the logistic map, users are able to obtain the plain image without data loss, when encrypting the image on Device 1 and decrypting it on Device 2. As previously mentioned, this occurs because this technique performs all the arithmetic operations in a field. This means there is no need for rounding; consequently, such approach does not have round-off error.

Additionally to the proposed approach, two other methods (References [82] and [36]) were tested to demonstrate the problem regarding data exchange when dealing with chaos-based encryption algorithms. Table 3.6 shows the device configuration used to encrypt and decrypt the plain image. Figure 3.3 (a) and (b) demonstrate the methods cannot decipher the information, while the proposed encryption algorithm (c) is able to fully recover the image.

Table 3.6: Description of each device used to simulate the proposed encryption scheme and the other two algorithms (References [82] and [36]). The same MATLAB version was used in Devices 1 and 2.

Device	**Description**
1 - Encryption	Intel(R) Core(TM) i5-8265U CPU @ 1.60 GHz - Windows system
2 - Decryption	Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10 GHz - Linux system

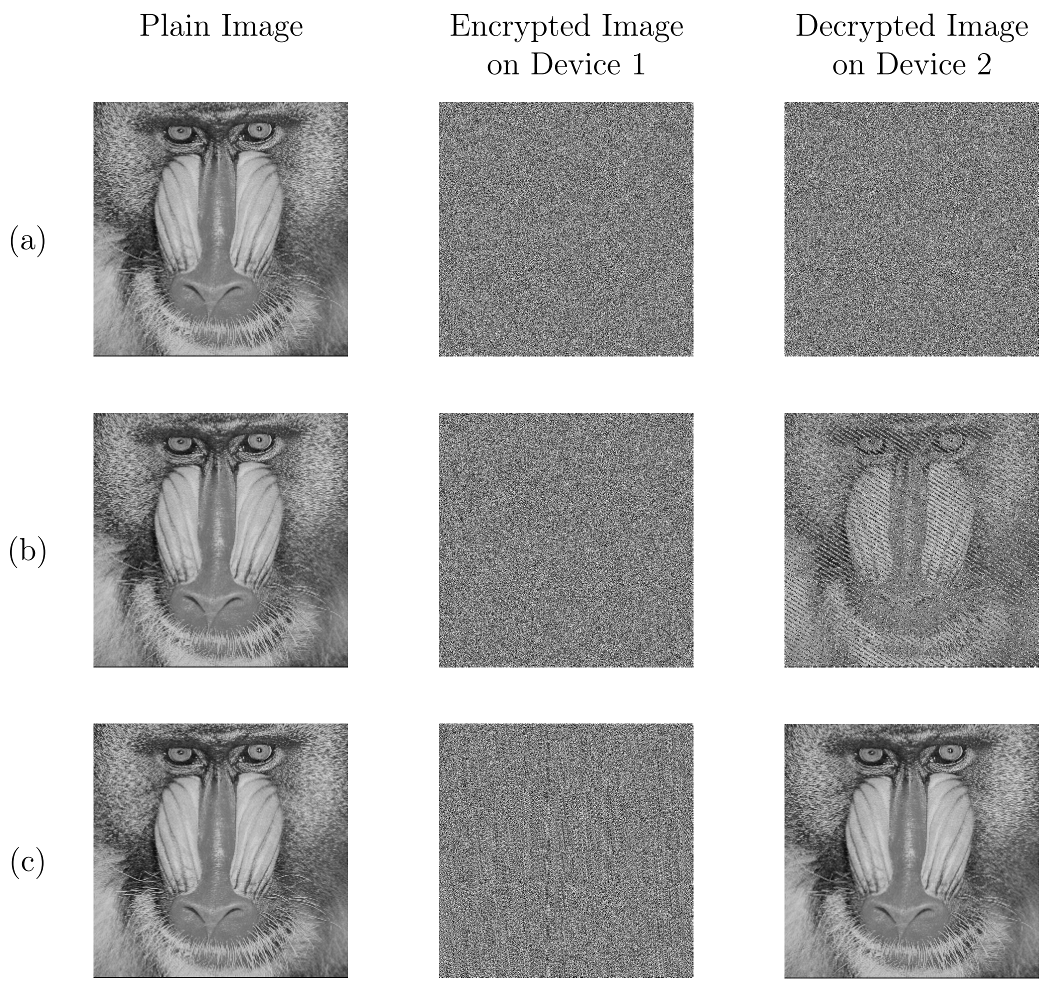


Figure 3.3: Reproducibility of chaos-based encryption algorithms on different devices. Line (a) and (b) presents the encryption and decryption results using the algorithms reported in [82] and [36], respectively. Using such algorithms, it was not possible to recover the plain image on Device 2 after encrypting it on Device 1. Line (c) exhibits the results applying the proposed technique. Unlike the previous results, after encrypting the image on Device 1 and decrypting it on Device 2, the original image is fully recovered, without data loss. `Algorithms: Image Encryption / Comparison / NNGcipher.m, GHGcipher.m and GFcipher.m.`

3.3.6 Performance Comparison

In order to validate the performance of the proposed encryption scheme, the results for the Baboon image were compared with other results found in the literature. Table 3.7 illustrates such performance comparison, showcasing the effectiveness and robustness of the proposed encryption algorithm. Notably, the algorithm demonstrates either comparable or superior performance across all evaluated criteria, affirming its potential for secure image encryption applications.

Table 3.7: Performance comparison using the Baboon image (512×512). The comparison reveals that the suggested scheme exhibits comparable or superior performance when juxtaposed with other algorithms.

Criteria		**Proposed Cryptosystem**	**References**			
			[91]	[71]	[15]	[37]
Key Space		2^{4096}	2^{283}	2^{1024}	$5.46 \cdot 10^{80}$	2^{256}
Entropy		7.9992	7.9993	7.9993	7.9993	NA
Correlation Coefficient	Horizontal	0.0075	−0.0018	−0.0036	−0.0108	0.0220
	Vertical	−0.0005	0.0007	0.0001	0.0087	−0.0083
	Diagonal	−0.0038	−0.0013	0.0023	0.0058	−0.0013
NPCR (%)		99.68	99.61	99.70	99.61	NA
UACI (%)		33.36	33.48	32.28	33.43	NA

3.4 Final Remarks

This chapter proposed a new image encryption algorithm based on chaos and Galois field theories. Adopting a straightforward and easy-to-implement one-dimensional logistic map, a 64-degree irreducible polynomial on $GF(2)[x]$, and a condition based on the plain image, the designed chaos-based scheme demonstrates remarkable efficiency in encrypting images.

Results show the cryptosystem is secure against many cyberattacks, mainly brute force attacks since the algorithm has an astonishing key space of up to 2^{4096}. Moreover, this approach passed the ENT test suite, which guarantees acceptable pseudo-random properties.

Most notably, the image encryption algorithm showed reproducibility in distinct digital devices, i.e. the algorithm is reliable across different hardware and software setups. Additionally, a comparative study with other encryption schemes shows this approach is just as efficient as others.

In summary, the pros and cons of the chaos-based cryptosystem presented in this chapter can be outlined as follows:

- Advantages:

1. The speed of the proposed encryption algorithm is notable. By representing binary forms of real numbers as matrices over GF(2), the logistic map requires fewer iterations for encrypting images compared to alternative approaches.
2. The operations within finite fields exhibit zero errors, eliminating the occurrence of rounding errors. Consequently, the cryptosystem can consistently yield identical results across various machine configurations, in contrast to other approaches that involve floating-point numbers.
3. The method demonstrates robust security against a range of well-known cyber-attacks, including threats like brute-force attacks and differential attacks.

- Disadvantages:

  1. Some applications utilizing simple microcontrollers with limited resources and low memory capacity may experience restricted performance when implementing the proposed encryption algorithm. In such cases, alternative approaches over finite fields that do not involve matrix operations are recommended.

Chapter 4

Lyapunov Exponent Estimation

Part of the work in this chapter has been accepted to be published in the following: Nardo, L. G., Nazaré, T. E., Nepomuceno, E., Arias-Garcia, J., and Butusov, D. N. (2024a). Computation of the largest Lyapunov exponent using SPICE-like programs. In *2024 7th IFAC Conference on Analysis and Control of Nonlinear Dynamics and Chaos (IFAC ACNDC)*, pages 1–6.

4.1 State-of-the-Art and Problem Description

As previously described in Chapter 2, one way to provide evidence for the chaotic nature of a system is to compute its positive largest Lyapunov exponent, which estimates the exponential divergence of the system's nearby trajectories [6, 47, 94, 107, 136]. The computation of LLE is based on a theorem first provided by Oseledets [94], which establishes the theoretical background for estimating the LLE. Since then, many numerical algorithms have been proposed for computing Lyapunov exponents.

The first practical method for estimating the Lyapunov exponent was developed by Wolf et al. [136]. This classical algorithm[1] involves the Gram-Schmidt reorthonormalization and logarithmic calculations of perturbation lengths [6]. Later, Rosenstein et al. [107] and Kantz [47] developed fast and robust algorithms for estimating Lyapunov exponents based on statistical properties of the local divergence rates of nearby trajectories. These three methods can be categorized as time-series approaches, which use Takens's theorem [92, 126] to reconstruct a chaotic dynamical system from a series of state observations. Other examples of time-series approaches include the ones developed by Sano and Sawada [111], Eckmann and Ruelle [28], Parlitz [98], and Yao et al. [141].

Other techniques to estimate LEs are model-based ones. Despite time-series methods which usually have to reconstruct the attractor to estimate such exponent, the concept behind the model-based ones is that it requires just the original equations of motion of the

[1]For more details about this method, see Subsection 2.1.2.

system. Mendes and Nepomuceno [70] used natural interval extensions of chaotic dynamical equations to estimate the largest Lyapunov exponent. Thenceforth, Nepomuceno et al. [89] extended the concept to other types of models, such as the NARMAX (Nonlinear AutoRegressive Moving Average model with eXogenous inputs). Moreover, Peixoto et al. [100] employed concepts from interval arithmetic and rounding modes to compute the LLE.

Although the first algorithms to estimate Lyapunov exponents date back to the 1980s, there is still interest from the academic and industry community in creating new and more efficient methods. For instance, Kim and Choe [54] presented a novel algorithm based on high-precision computation and the works of Rosenstein et al. [107] and Kantz [47]. Matsuoka and Hiraide [67] introduced a numerical method to estimate the LE of nonlinear maps using a shift transform. Moreover, Zhou et al. [147] developed a simple and easy-to-perform computational procedure which requires few parameters to compute the LLE. Afterwards, Zhou and Wang [146] extended the previous method to calculate the second-largest Lyapunov exponent, an important factor in characterizing hyperchaos from dynamical systems.

While various methods exist in the literature for estimating Lyapunov exponents, none directly determine the largest Lyapunov exponent using SPICE-like programs without auxiliary tools. Electronic circuits, among the many chaotic systems in the literature, stand out as one of the most popular. However, estimating the Lyapunov exponent in these circuits typically involves data extractions before applying a Lyapunov exponent estimation tool. This is particularly true for methods based on time series, demanding substantial time, energy, and effort. Consequently, this chapter proposes a novel method for estimating the LLE using circuit diagrams in SPICE-like programs. This method may revolutionize semiconductor industry practices by facilitating the identification of potentially chaotic circuits, helping prevent failures in verification processes, and enabling a quick estimation of the Lyapunov exponent.

The rest of this chapter is laid out as follows: the proposed method is explained in Section 4.2. Section 4.3 presents the results obtained by the proposed approach when applying it to six distinct circuits. Lastly, final remarks are shown in Section 4.4.

4.2 Lyapunov Exponent Estimation Method

The method to estimate the LLE from chaotic circuits can be summarized by the following steps:

- **Step 1**: Create two equivalent circuits following the definitions given in Section 2.4 and the idea behind them in Figure 2.4. Use a SPICE-like program for the simulation.

  Selecting these circuits is straightforward through the application of a single equivalence relation among circuits. Utilize linear components such as resistors, inductors, and capacitors for this equivalence.

- **Step 2**: Insert the inputs of a differential amplifier in each circuit's chaotic signal that you want to measure.

  All the resistors in such a subtractor have the same values, making this circuit a unity gain differential amplifier. The output is the simple difference between the chaotic pseudo-orbits of the analog circuits being measured.

- **Step 3**: Set the same initial conditions for both circuits.

  The limitations imposed by the IEEE 754-2019 floating-point standard [43] highlight the sensitivity of circuits to even minor alterations. Consequently, such slight modifications in the circuits lead to divergent trajectories during numerical simulations. Hence, it is not necessary to set the initial conditions differently.

- **Step 4**: Perform the circuit simulation in Transient Analysis mode with the stop time and time-step settled within the SPICE-like program.

  Up to this point, note that the user must establish a few necessary parameters to determine the LLE, for example, initial conditions and time-step. Unlike time-series methods described in the literature, which require finding the embedding dimension and embedding delay, the proposed method does not.

- **Step 5**: Utilize the graphical tools within the circuit simulation program to plot the output of the differential amplifier, which measures the lower-bound error. Next, define the initial and final points of the slope on the logarithmic curve representing the absolute value of the lower-bound error. The line fitted to this slope corresponds to the largest Lyapunov exponent.

  As presented in Section 2.4, the lower-bound error measures the discrepancy between simulated dynamical systems (or pseudo-orbits) and the true orbit. In this case, if a circuit is chaotic, the distance between these two entities must be locally exponentially divergent. Thus, a slope of the logarithm of the lower-bound error captures this divergence and quantifies it as a number that is precisely the definition of the largest Lyapunov exponent.

The complete set, illustrated in Figure 4.1, serves as a model for designing simulations to estimate the LLE. By employing two equivalent Sprott's jerk circuits [122], the differential amplifier calculates the divergence between the pseudo-orbits. Acting as interval extensions, these two equivalent circuits facilitate the calculation of lower-bound error and enable the estimation of the LLE. For further insights into the Lyapunov exponent estimation using interval extensions, refer to [70].

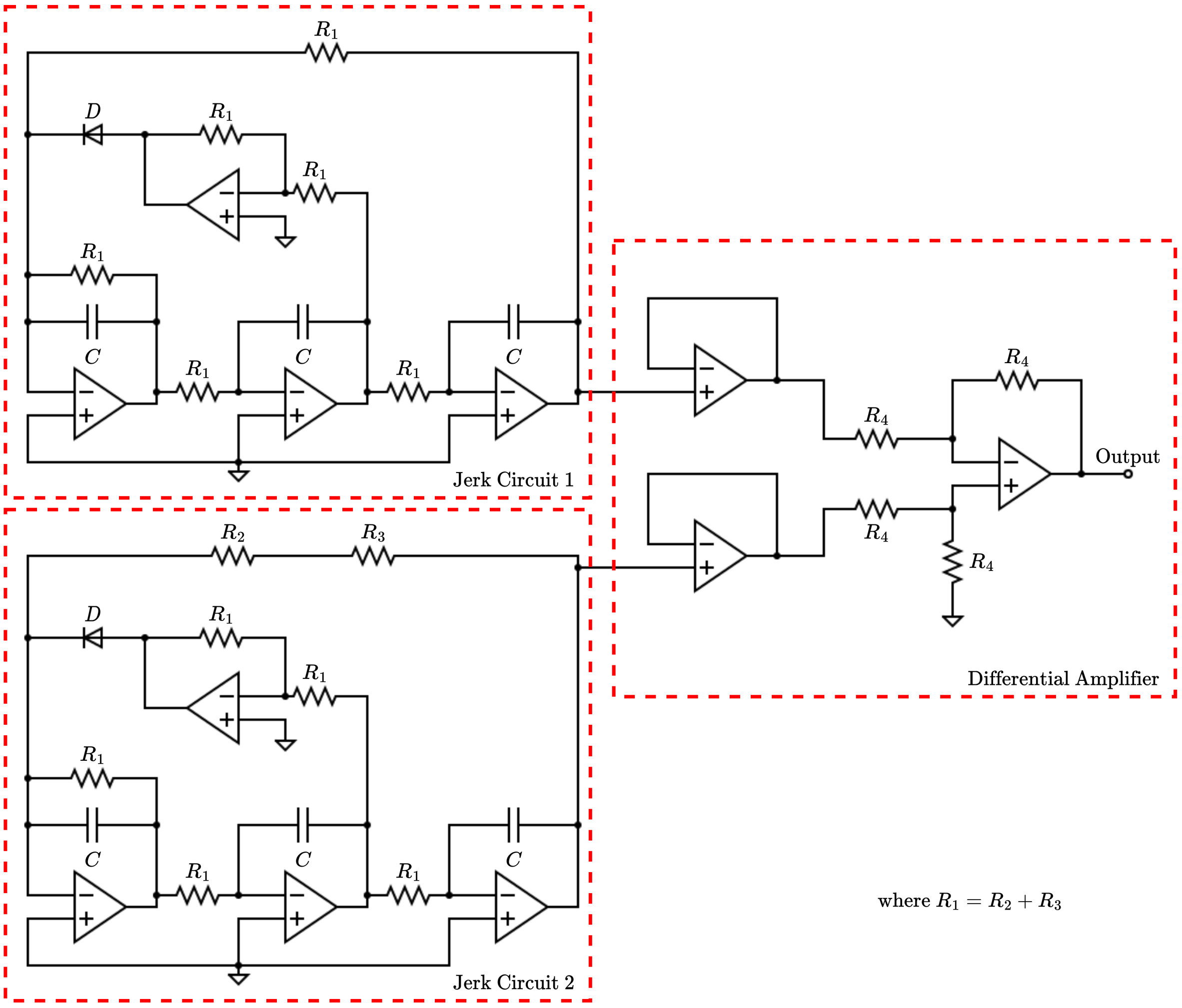


Figure 4.1: Schematic of how the largest Lyapunov exponent is determined. This is the first work that describes a methodological approach to estimating such an exponent using SPICE-like programs. Note that the jerk circuits are equivalent, where $R_1 = R_2 + R_3$, but slightly different from the point of view of the IEEE 754-2019 floating-point standard. This difference is enough to make the trajectories of these circuits diverge.

4.3 Results and Discussion

In this section, six known analog circuits are discussed [19, 46, 62, 101, 118, 122], and the proposed method is applied to estimate their largest Lyapunov exponent. These circuits were chosen for their diversity and to demonstrate the applicability of the proposed approach in different scenarios. Some of the circuits are nonautonomous [46, 62], while others are autonomous [19, 101, 118, 122]. One circuit emulates the known Rössler attractor [46], and one circuit exhibits hyperchaotic behavior [118]. The chaotic equations that model the circuits studied are summarized in Table 4.1. All circuit simulations were performed using the LTspice XVII Simulator - Analog Devices on a Windows 10 Enterprise with an Intel(R) Core(TM) i7-8700 CPU @ 3.20 GHz processor. Additionally, the double precision (64 bits) floating-point format was used to obtain all results.

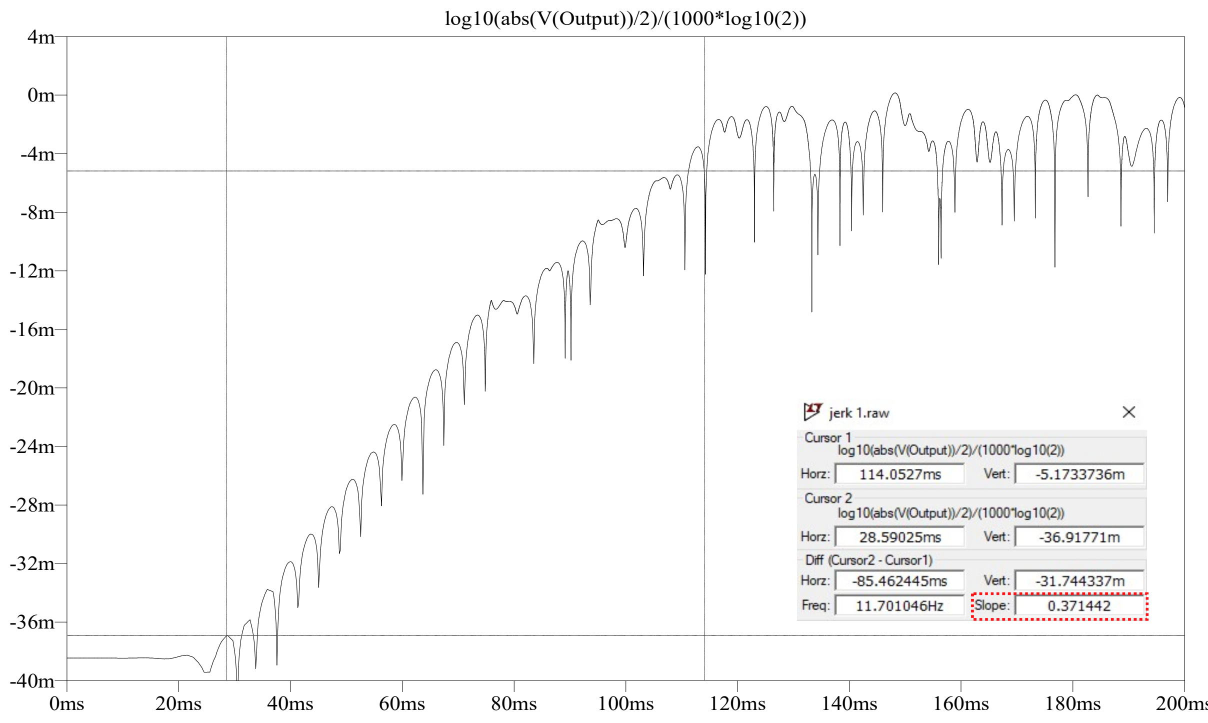
Table 4.1: List of chaotic circuits used to test the proposed method that estimates the LLE. The initial conditions are arbitrarily chosen but fixed for both equivalent circuits. Here, a, b, c, d, and e are parameters, and x, y, and z are variables for each equation. Additionally, $f(\dot{x})$ in the Jerk 1 equation refers to the diode model.

Reference	Circuit	Equations	Time Step	Initial Conditions
[19]	Chua	$\begin{cases} \dot{x} = a(y - x - f(x)) \\ \dot{y} = x - y + z \\ \dot{z} = -by \end{cases}$ $f(x) = cx + 0.5(d-e)(\lvert x+1\rvert - \lvert x-1\rvert)$	$0.1\mu s$	$x_0 = 52.502946\mu V$ $y_0 = 29.168286pV$ $z_0 = 29.168287nA$
[122]	Jerk 1	$\begin{cases} \dot{x} = y \\ \dot{y} = z \\ \dot{z} = -az - x - f(\dot{x}) \end{cases}$	$1\mu s$	$x_0 = 5.2998666\mu V$ $y_0 = 10.599933\mu V$ $z_0 = 10.599864\mu V$
[62]	Jerk 2	$\begin{cases} \dot{x} = y \\ \dot{y} = z \\ \dot{z} = -az - y \pm (\lvert x\rvert - 1) \end{cases}$	$1\mu s$	$x_0 = -1.5979735V$ $y_0 = 30.618183\mu V$ $z_0 = 10.599607\mu V$
[101]	Jerk 3	$\begin{cases} \dot{x} = y \\ \dot{y} = z \\ \dot{z} = -az - by + c(sgn(x) - x) \end{cases}$	$0.1\mu s$	$x_0 = -126.2131nV$ $y_0 = 996.1779\mu V$ $z_0 = 1.992482mV$
[46]	Rössler	$\begin{cases} \dot{x} = -y - z \\ \dot{y} = x + ay \\ \dot{z} = b + z(x - c) \end{cases}$	$10\mu s$	$x_0 = -1.5765196mV$ $y_0 = 3.4585271mV$ $z_0 = -590.70514\mu V$
[118]	Hyperchaotic	$\begin{cases} \dot{x} = -y - z - w \\ \dot{y} = x \\ \dot{z} = a(y - sgn(y)) - bz \\ \dot{w} = -cz \end{cases}$	$1\mu s$	$x_0 = -25.340626nV$ $y_0 = -670.19903nV$ $z_0 = 53.000852mV$ $w_0 = -51.94018mV$

Considering the importance of reproducibility in scientific research, as emphasized by the conclusions of Nazaré et al. [85], which assert that outcomes achieved using a SPICE-like program can differ across different computers and software versions, it is crucial to grant access to the schematic files used in this study. This ensures that other researchers can replicate the tests under comparable circumstances, thereby validating the results. To facilitate this process, the schematic files used in this chapter are provided in Appendix B.

For instance, the estimated largest Lyapunov exponent for the Jerk 1 circuit [122] is shown in Figure 4.2. The figure depicts the logarithm of the lower-bound error for the two equivalent circuits shown in Figure 4.1. The initial and final points for calculating the slope are at the start of the lower-bound error increase and when the error stops rising. The red box in the figure highlights this slope, representing the estimated largest Lyapunov exponent ($bits/ms$). Further details on all simulated circuits, including the equivalence relation and a graph illustrating divergence and LLE, can be found in Appendix B.

Table 4.2 compares the estimated largest Lyapunov exponents using both the proposed approach and Wolf et al.'s method. The exponents for each circuit demonstrate good agreement between the two methods, with absolute differences ranging from 0.24% to 9.64%. More importantly, all circuits exhibit positive Lyapunov exponents. Given that the simulations are conducted on a digital computer, the binary logarithm illustrates the rate of information loss over time.



```
jerk 1.raw
Cursor 1
        log10(abs(V(Output))/2)/(1000*log10(2))
Horz: 114.0527ms        Vert: -5.1733736m
Cursor 2
        log10(abs(V(Output))/2)/(1000*log10(2))
Horz: 28.59025ms        Vert: -36.91771m
Diff (Cursor2 - Cursor1)
Horz: -85.462445ms      Vert: -31.744337m
Freq: 11.701046Hz       Slope: 0.371442
```

Figure 4.2: The lower-bound error for the jerk circuit [122]. The red box highlighting the slope is the linear fit of the lower-bound error. The value in such a box is the estimation of the largest Lyapunov exponent (LLE = 0.371442 $bits/ms$). The x-axis is time, and the y-axis is the binary logarithm of the absolute value of the lower-bound error. `Schematic: Largest Lyapunov Exponent / jerk 1.asc`.

Table 4.2: The determined largest Lyapunov exponents using the proposed and Wolf et al.'s approaches. The exponents for each circuit exhibit consistent agreement between the two methods. `Schematics: Largest Lyapunov Exponent / chua.asc, jerk 1.asc, jerk 2.asc, jerk 3.asc, rossler.asc, hyperchaotic.asc`.

Circuit	**Largest Lyapunov Exponent** ($bits/ms$)		**Difference (%)**
	The proposed approach	**Wolf et al.'s method**	
Chua	3.0289	2.7626	9.6395
Jerk 1	0.3714	0.3705	0.2429
Jerk 2	0.8058	0.7759	3.8536
Jerk 3	0.4193	0.4147	1.1092
Rössler	0.6882	0.6981	−1.4181
Hyperchaotic	2.7555	2.8116	−1.9953

4.4 Final Remarks

This chapter introduces a novel method for estimating the largest Lyapunov exponent of chaotic circuits. Unlike traditional approaches that rely on time-series or model-based methods, the proposed technique utilizes circuits in SPICE-like programs. This marks a distinctive trend in Lyapunov exponent estimation, opening up new possibilities for research.

The concept of exponential divergence in chaotic systems enables the measurement of the LLE through a differential amplifier. This involves a straightforward application of a linear fit to the logarithm of the divergence between the pseudo-trajectories of two equivalent circuits. This method offers a simple yet powerful approach to quantifying the dynamic behavior of chaotic systems.

Using the LTspice software from Analog Devices, the proposed method estimates the largest Lyapunov exponent for six analog circuits, demonstrating a high correlation with Lyapunov exponent values obtained through the well-known Wolf et al.'s method [136]. Taking Sprott's jerk circuit [122] as an example, the estimated LLE differs by only 0.2429% from Wolf et al.'s method.

In summary, the following advantages and disadvantages of the proposed method in this chapter can be outlined:

- Advantages:

  1. The proposed technique is straightforward. In contrast to time-series approaches, it does not necessitate certain parameters, such as embedding dimension, embedding delay, or initial distances of the neighborhood.
  2. The method is user-friendly, catering to users with limited knowledge of dynamical systems theory, who can readily apply the proposed technique.
  3. The estimation of the largest Lyapunov exponent is fast compared to other methods.

- Disadvantages:

  1. If the circuit diagram is not available, the method may incur demanding, and alternative methods from existing literature may prove more effective.
  2. The proposed method exclusively estimates the largest Lyapunov exponent. If there is a need to estimate other exponents to comprehensively characterize the dynamics of the chaotic system, this method is not recommended.

Chapter 5

Digital Chaotic Circuit

Part of the work in this chapter has been published in the following: Nardo, L., Nepomuceno, E., Muñoz, D., Butusov, D., and Arias-Garcia, J. (2023). A hardware-efficient perturbation method to the digital tent map. *Electronics*, 12(8):1953.

5.1 State-of-the-Art and Problem Description

In the digital era, extensive research has focused on designing digital systems, with reconfigurable hardware like FPGAs emerging as a recommended solution due to their flexibility, reliability, and rapid development [127, 129]. Many papers [16, 18, 104] have used reconfigurable hardware for implementing chaotic systems and chaos-based applications.

Despite advances, applying chaotic systems in finite-precision environments encounters challenges leading to dynamical degradation and shortened cycle lengths [59]. Various strategies to mitigate these issues include high finite precision, cascading/coupling multiple chaotic systems, switching, and pseudo-random perturbations [64], as previously described in Chapter 1.

While progress has been made in countering dynamical degradation, existing approaches often lack simplicity and hardware efficiency. For instance, a Lyapunov exponent-based parameter perturbation method proposed by Cao et al. [12] preserved chaotic features in the tent map but sacrificed hardware efficiency. Despite extensive work on representing chaotic maps in digital circuits [77, 123, 131], hardware optimization is frequently overlooked, resulting in resource-intensive systems with high energy consumption [134].

Efficient hardware configuration is critical given the increasing use of chaos across diverse fields. However, few papers focus on implementing chaotic maps and systems with an emphasis on efficiency. For example, Palacios-Luengas et al. [96] developed a digital noise generator using the inverted one-dimensional tent map, claiming fewer logical resources consumption. Additionally, Nepomuceno et al. [87] minimized the logistic map using digit-complement and an XOR gate-based multiplication algorithm, emphasizing the intrinsic link between the simplicity of a digital-chaotic model and the number of mathematical operations.

This chapter addresses the importance of minimizing hardware resource consumption, avoiding dynamical degradation, and creating smaller circuits. It introduces a methodological approach to design a simple binary tent map, competing with state-of-the-art chaotic circuits by adopting the 1's complement fixed-point format. Successfully implemented in hardware, this map maintains chaotic properties, demonstrating excellent power efficiency, logical resource consumption, and execution time. The chapter also presents a simple and efficient method to counteract dynamical degradation in the digital tent map, using an XOR gate to disturb each iteration. Additionally, the practical applicability of the digital tent map in optimization problems is showcased.

The rest of this chapter is organized as follows: Section 5.2 describes the tent map, its digital design, and the perturbation process. Section 5.3 summarizes the main results and corresponding applications. Finally, Section 5.4 provides concluding remarks.

5.2 Design of the Digital Tent Map

5.2.1 Tent Map

The tent map [10] is described by:

$$x_{n+1} = \begin{cases} \mu x_n, & \text{if } \; 0 \leq x_n < 0.5 \\ \mu(1 - x_n), & \text{if } \; 0.5 \leq x_n < 1, \end{cases} \tag{5.1}$$

where the bifurcation parameter $\mu \in [0, 2]$, x_n is the n^{th} state of the system, and $x_n \in [0, 1)$. The model of this system is simple yet exhibits chaotic behavior, which is sensitive to initial conditions. As one of the objectives is to design a hardware architecture for a chaotic map that consumes fewer logical resources while maintaining its chaotic properties, it is crucial to select a map that requires minimal mathematical operations, since the simplicity of digital chaotic systems is directly related to the number of arithmetic operations involved. Hence, opting for the tent map over other chaotic maps is a judicious decision. Additionally, the tent map has been successfully applied in various fields, including radar design [35], image encryption [12, 58], and metaheuristic optimization algorithms [49]. These applications demonstrate the importance of creating a simple and efficient hardware architecture for this chaotic system. Finally, as presented in [22], the analysis of the tent map can be simplified using binary representations, which further highlights its advantages for digital circuit applications.

5.2.2 Method

In this section, a method to decrease the number of arithmetic operations and a perturbation approach to iterate Equation (5.1) is presented. In this context, a fixed-point representation is chosen for its superior hardware performance, despite its reduced flexibility and capability to represent a narrower range of values compared to floating-point arithmetic. Furthermore, given that the tent map's domain is $x_n \in [0,1)$ and the objective is to implement a digital chaotic system with minimal logical resources, the use of floating-point arithmetic is unnecessary in this context.

As $x_n \in [0,1)$, using a fixed-point representation, the fractional binary number is represented by:

$$x = 0.b_1 b_2 b_3 \ldots b_{pr-1} b_{pr}, \tag{5.2}$$

where 0 is the hidden bit, and b_{pr} represents the pr^{th} bit. In this paper, representations with $pr \geq 16$ are investigated, where b_{pr} also indicates the resolution.

In such a way, considering 1's complement [97]:

$$M = 2^{pr} - ulp, \tag{5.3}$$

where M is the complementation constant and ulp is the unit in the last place.

Let x be a binary number and $\tilde{x}$ its respective bitwise complement. Hence:

$$x + \tilde{x} = M = 2^{pr} - ulp. \tag{5.4}$$

Note that $1_{10} = 1_2 \approx 0.111\ldots11_2 = x + \tilde{x} = 1_2 - ulp$. Thus, as long as $pr \to \infty$ and $x_n \in [0,1)$, the ulp can be neglected:

$$\tilde{x} \approx 1_2 - x. \tag{5.5}$$

Since the subtraction only occurs when $x_n \geq 0.5_{10}$ (seen in Equation (5.1)), the first step must confirm such a condition. Because $0.5_{10} = 0.100\ldots0_2$, the most significant bit b_1 is the only bit needed to verify the aforesaid condition. Therefore, when $b_1 = 0$, $x_n < 0.5_{10}$, no subtraction is required. Otherwise, when $x_n \geq 0.5_{10}$, the complement operation is necessary, according to Equation (5.5). These conditions can be computed by:

$$\tilde{b}_1 x_n + b_1 \tilde{x}_n, \tag{5.6}$$

where x_n is the binary number of the n^{th} state. In this context, it is clear that XOR gates can be used.

To simplify Equation (5.1), it is necessary to use $\mu = 2$, because it allows performing arithmetic multiplication using shift operations. However, the tent map displays dynamical degradation when $\mu = 2$, resulting in a periodic function. Indeed, orbits of binary shift chaotic maps eventually converge to zero [149]. In this way, an XOR gate between the least two significant bits is applied. Thus, in the digital domain, the tent map with the perturbation method can be represented as:

- Digital tent map:
$$x_{n+1} = \begin{cases} 2x_n, & \text{if} \quad b_1 = 0 \\ 2\tilde{x}_n, & \text{if} \quad b_1 = 1. \end{cases} \tag{5.7}$$
- Perturbation method:
$$b_{pr} = b_{pr-1} \oplus b_{pr}. \tag{5.8}$$

It is important to explain that an XOR gate is used since it has a 50% chance of outputting logical high or low level. This operation is sufficient to cause a disturbance and reduce the degradation phenomenon caused by the reduction of precision in digital circuits [12, 59], as will be seen in Section 5.3.

Table 5.1 compares the many existing methods to counteract such an issue in the tent map. It can be observed that many approaches are more complex to implement in hardware than perturbations at a logic-gate level. If such techniques were conceived as a hardware architecture, they would certainly consume many logical resources.

Table 5.1: Approaches for counteracting the dynamical degradation in the tent map. The 'Complexity' category was assessed based on the number of logical components that would be utilized if the approach were implemented using a programmable hardware device.

Approaches for counteracting dynamical degradation		
Reference	**Complexity**	**Description**
[12]	High	Trigonometric functions of Chebyshev map were used to pseudo-randomly perturb the tent map.
[44]	Medium	The authors perturbed a system based on tent maps with a switching method based on a coupling matrix.
[45]	Medium	The authors created a modified tent map using modulo and scaling operators to enlarge its chaotic region.
[110]	High	A hybrid chaotic system was created by coupling the logistic, Hénon and tent maps.
[132]	High	The tent map was modeled dividing it into 2^N parts and selecting a specific part to disturb the entire system.
Proposed approach	Low	An XOR gate is used to perturb the least two significant bits of every iteration.

Figure 5.1 shows the flowchart representation of the digital tent map. Note the small number of blocks and operations that can still generate chaotic orbits. This is due that 1's complement fixed-point numerical representation allows the replacement of arithmetic operations using a few logic gates.

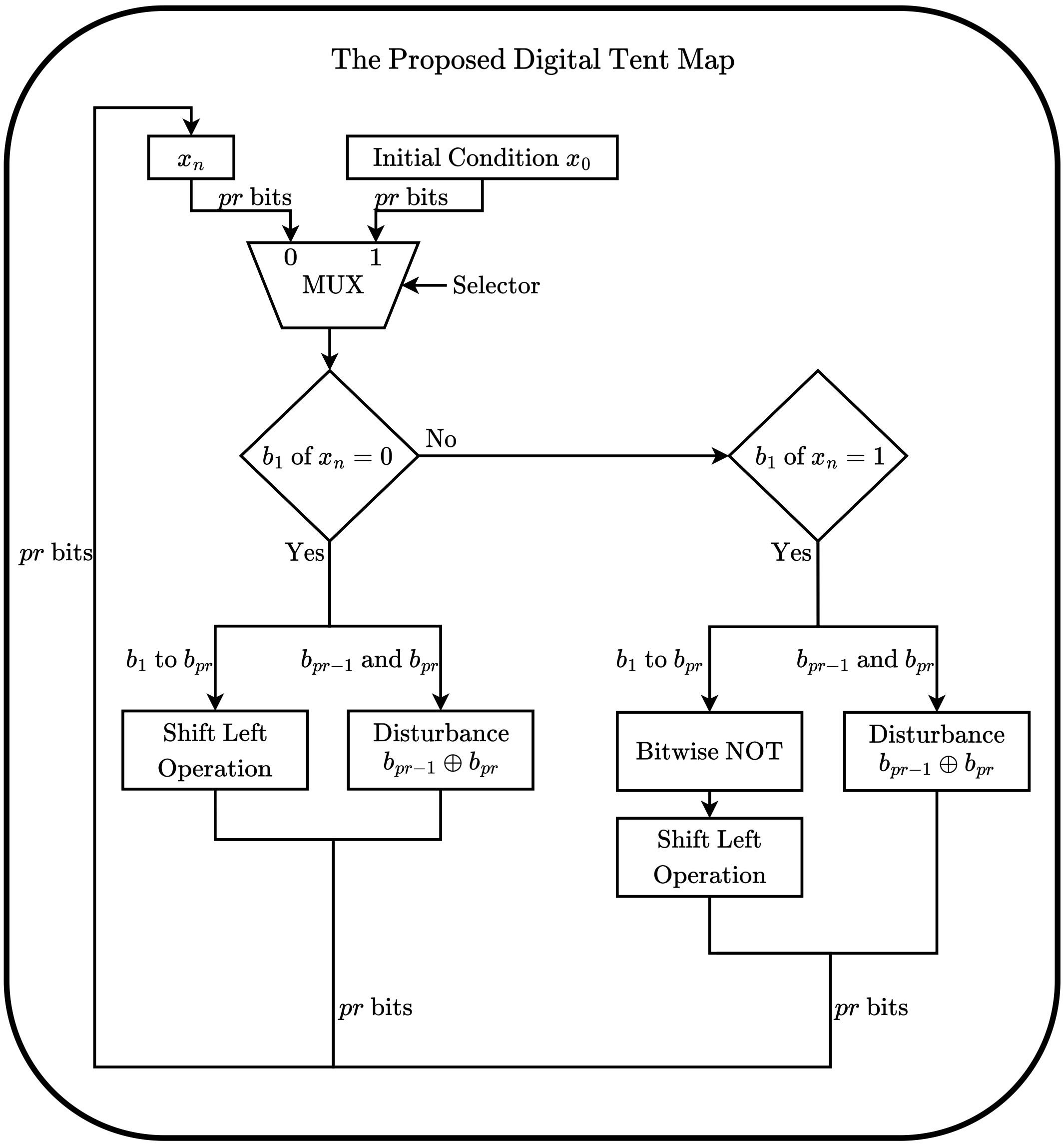


Figure 5.1: Flowchart representation of the digital tent map.

5.3 Results and Discussion

5.3.1 Hardware Performance

The designed circuit was implemented using the VHDL (VHSIC Hardware Description Language) in the Xilinx Vivado 2019.1 design tool, and its operation was simulated in ModelSim - Mentor Graphics. The low-cost FPGA device Artix 7 XC7A100TFGG676 - 3 was adopted to embed such a chaotic system. This choice is based on the fact that this semiconductor device is commonly used for prototyping digital circuits. Moreover, FPGA offers several advantages, including flexibility, reliability, robustness, and fast processing [129].

The tent map was implemented using three different fixed-point numerical precision formats: 16-bit, 32-bit, and 64-bit. These formats were chosen due to their common usage in many designs. One advantage of this circuit is its ability to adapt to different precision requirements based on the specific application it will be used for.

To demonstrate this result, the 16-bit implementation of the tent map is presented in Figure 5.2. This digital circuit presents chaotic behavior, yet contains only a small number of logical components and has a latency of one clock cycle. In the figure, the block labeled '&' represents the concatenation operation. The first fifteen XOR gates determine which condition for the tent map is satisfied in the n^{th} iteration and, perform the complement operation if necessary. To save FPGA resources, the use of an XOR gate to execute $b_1 \oplus b_1$ is avoided, since the resultant bit is always zero and will be discarded after a shift left operation. Additionally, note that the perturbation process is performed by the rightmost XOR gate, since $(b_1 \oplus b_{pr-1}) \oplus (b_1 \oplus b_{pr}) = b_{pr-1} \oplus b_{pr}$.

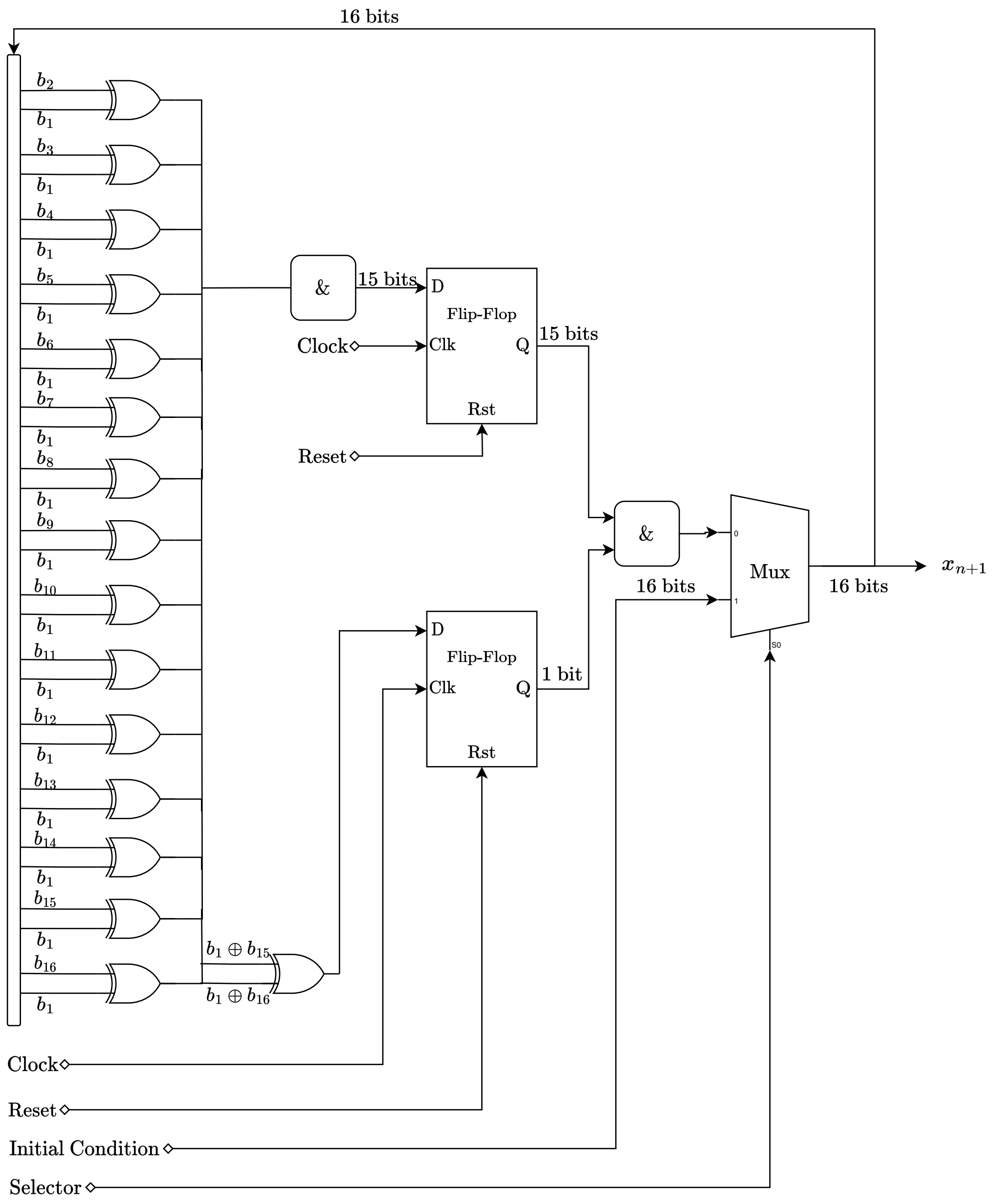


Figure 5.2: A digital representation of the tent map.

Table 5.2 shows which components are used to represent the tent map. Given the many synthesis and implementation strategies in the Xilinx Vivado design tool, the choice was made to optimize the area of the implemented digital circuit as much as possible. The proposed scheme is compared to [87], which also focuses on minimizing hardware. Note that the proposed circuit contains fewer logical resources, consumes less than 1% of the available capacity in the FPGA, does not use DSP (Digital Signal Processing) resources, and has a superior operating frequency.

Table 5.2: Components in digital tent map representation with varying resolutions, compared to a design using the logistic map. The value in parentheses indicates the percentage of LUTs, FFs, Logic Utilization, and Registers used relative to the available FPGA resources. `Projects: Digital Tent Map / tent_map.xpr`.

The proposed digital tent map					
Bits	**FPGA Device**	**Look-Up Table (LUT)**	**Flip-Flop (FF)**	**Frequency (MHz)**	**Power (mW)**
16	Xilinx	17 (0.03%)	16 (0.01%)	344.83	125
32	Artix	33 (0.05%)	32 (0.03%)	344.83	157
64	7	65 (0.10%)	64 (0.05%)	333.33	216
Nepomuceno et al. [87] - logistic map					
Bits	**FPGA Device**	**Logic Utilization**	**Registers**	**Frequency (MHz)**	**DSP**
16	Intel	9 (0.06%)	32 (0.05%)	125	1 (1.19%)
32	Cyclone	39 (0.25%)	68 (0.11%)	108	3 (3.57%)
64	V	169 (1.06%)	86 (0.14%)	68	9 (10.71%)

Although Xilinx and Intel produce FPGAs with different architectures, it is relevant to mention that the proposed tent map presents the same results concerning Logic Utilization and Registers as the LUTs and FFs indicated in Table 5.2 when using the same Cyclone V device used in [87]. Therefore, the proposed hardware architecture remains superior to the one shown by Nepomuceno et al. [87].

Additionally, Table 5.3 compares the logical resource consumption among various tent-map-based works and other discrete chaotic maps. These other maps were selected because they have a similar number of operations to the tent map and are well-known, widely applicable chaotic maps. The sixth column shows the ratio between the total number of logical resources used and the number of bits, providing an objective measure of the simplicity of a chaotic system. This is important because many papers do not specify the components used. Notably, the number of components used in the proposed digital circuit is lower than in the other studies, with a reduction of at least 58%.

To underscore the chaotic nature of the proposed digital tent map and validate the efficacy of the perturbation method, the largest Lyapunov exponent was estimated using Wolf's et al. method [55, 136]. This calculation was performed on a decimal representation converted from the generated binary data produced by ModelSim. The LLE reported in

Table 5.3: Comparing logical components of chaotic maps across different approaches. The 'Ratio' category signifies the relationship between logical resource consumption and the number of bits employed in each approach.

Chaotic Map	References	FPGA Device	Bits	Logical Resources	Ratio
Ship Map	[53]	Not Specified	10	54	5.4
Twisted Tent Map			10	54	5.4
Logistic Map	[129]	Intel Cyclone II	8	66	8.3
Tent Map			23	756	32.9
Baker's Map	[149]	Intel Cyclone III	32	199	6.2
Bernoulli Map			32	199	6.2
Logistic Map			38	2634	69.3
Tent Map			32	219	6.8
Ship Map	[123]	Not Specified	32	163	5.1
			64	323	5.0
Tent Map			32	161	5.0
			64	321	5.0
Twisted Tent Map			32	168	5.3
			64	325	5.1
Tent Map	Proposed	Xilinx Artix 7	16	33	2.1
			32	65	2.0
			64	129	2.0

the literature is 1 $bit/iteration$ and, as can be seen in Table 5.4, the estimated values are in good agreement with the expected value. Moreover, as the largest Lyapunov exponent is positive, this can be considered evidence of the chaotic properties present in the proposed digital circuit. Furthermore, the return map and two time series with slightly different initial conditions were plotted, as shown in Figure 5.3. From Figure 5.3, it is possible to note that the digital circuit preserves the tent shape of the map, and the chaotic orbits diverge over time, indicating its sensitivity to initial conditions[1].

Table 5.4: Quantitative results demonstrate the chaotic nature and randomness of the proposed digital circuit. Notably, the tent map maintains its characteristics regardless of the precision used. `Algorithms: Digital Tent Map / tent_map.m`.

The proposed digital tent map			
Bits	**LLE** ($bits/iteration$)	**One-sample Kolmogorov-Smirnov test P-value**	**Runs test P-value**
16	0.992	1.000	0.859
32	0.995	0.999	0.807
64	0.996	0.344	0.941

[1]Please note that the time series depicted in Figure 5.3 does not exhibit periodic behavior, in contrast to the pattern observed in Figure 1.6 from Chapter 1.

The tent map, a type of one-dimensional piecewise linear map, is required to have a specific correlation function and probability distribution [8]. In Figure 5.3, the histogram and the autocorrelation function for all bit precision formats of the proposed digital tent map are presented. The histograms show a uniform distribution, indicating that all values of $x_n \in [0, 1)$ are equally likely to be the circuit output. Additionally, the lack of correlation in the generated orbits indicates that the samples are independent of each other. These results align with previous findings in the literature.

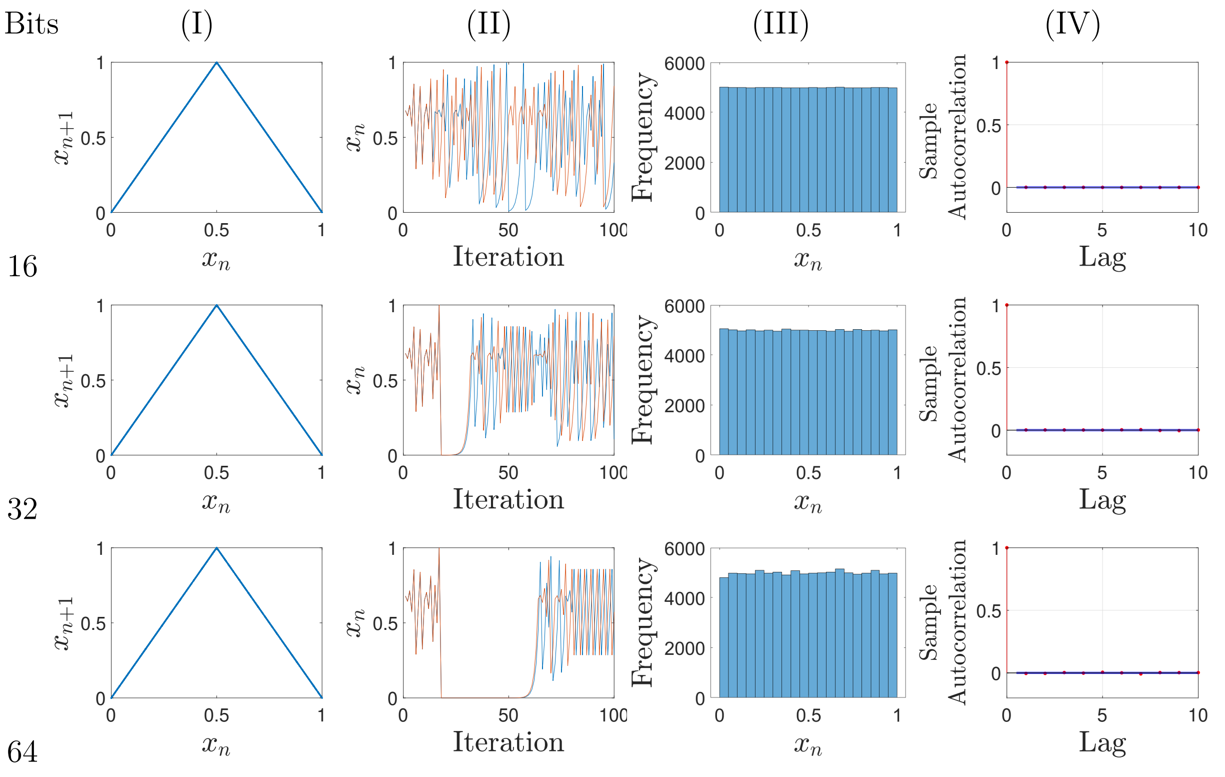


Figure 5.3: The results achieved by the proposed chaotic circuit and perturbation method. Column (I) shows the return map. Column (II) shows two chaotic time series with initial conditions x_0 (blue line) and $x_0 + ulp$ (brown line), respectively. Ultimately, columns (III) and (IV) show the histogram and the autocorrelation function, respectively. `Algorithms: Digital Tent Map / tent_map.m`.

The pseudo-random properties of the tent map can be observed through the P-values obtained from one-sample Kolmogorov-Smirnov and Runs statistical tests, as well as the ENT test suite analysis. The Kolmogorov-Smirnov test compares the sample distribution with a theoretical uniform distribution, while the Runs test examines the sequence for randomness. Table 5.4 shows the P-value obtained for each bit precision. Since all P-values are greater than or equal to the significance level of 0.01, these tests do not reject the null hypothesis that the sequence generated by the digital circuit is random and uniformly distributed at the 1% significance level.

Table 5.5 displays the results of the ENT test suite, demonstrating that the bitstream of the map passes all tests across various precision formats. This leads to the conclusion that the map can effectively generate pseudo-random numbers. As demonstrated by Herring and Palmore [40], PRNGs and chaotic systems are interconnected. Given that the digital circuit functions as a PRNG, a practical application could be its use in statistical sampling within a particle swarm optimization hardware architecture.

Table 5.5: The results of the ENT test suite for the proposed digital tent map. The value in parentheses is the expected value in each test. `Documented tested: tent_16_bits.bin, tent_32_bits.bin, and tent_64_bits.bin`

The proposed digital tent map				
Test		**Bits**		
		16	**32**	**64**
Entropy (1)	Value	1	1	1
	Decision	✓	✓	✓
Optimum Compression (0%)	Value	0%	0%	0%
	Decision	✓	✓	✓
χ^2 Distribution (10% $\sim$ 90%)	Value	88.44	87.21	58.07
	Decision	✓	✓	✓
Arithmetic Mean (0.5)	Value	0.4999	0.5001	0.4997
	Decision	✓	✓	✓
Monte Carlo Value for π (π)	Value	3.1432	3.1556	3.1561
	Decision	✓	✓	✓
Serial Correlation Coefficient (0)	Value	0.000005	0.001145	-0.006190
	Decision	✓	✓	✓

5.3.2 Hardware Architecture for Particle Swarm Optimization

The mono-objective optimization problems rely on finding a point, in a space of innumerable optimization variables, in which the objective function achieves its minimum solution [14].

A mono-objective optimization problem can be stated as:

$$x^* = \min(f(x)) \qquad (5.9)$$
$$\text{subject to: } \begin{cases} g_i(x) \leq 0 & \forall\ i = 1, 2, \ldots, r \\ h_j(x) = 0 & \forall\ j = 1, 2, \ldots, s \end{cases},$$

where x^* is the optimum point of the fitness function $f(x)$, and $g_i(x)$ and $h_j(x)$ are the constraints.

A classical metaheuristic algorithm[2] to solve mono-objective problems is Particle Swarm Optimization (PSO) [50]. PSO is a method that uses a population of candidate solutions,

[2]It is important to mention that metaheuristic algorithms do not necessarily ensure a globally optimal solution. However, they often offer solutions that are sufficiently satisfactory for various optimization problems.

namely particles, where each particle i moves in the search space in order to find the most promising area. Such an algorithm uses a velocity vector to update the position of each particle. The position of each particle is updated based on population social behavior, i.e., the swarm of particles [21]. From iteration to iteration, each particle i advances in the direction of its previous local best (p_{ilbest}) and the best global position (p_{gbest}) of the swarm.

The velocity v_i and position p_i of each particle i are updated by Equation (5.10) and Equation (5.11), respectively [144]:

$$v_i(j+1) = \omega v_i(j) + a_{c1}\text{rand}_1(p_{ilbest} - p_i(j)) + a_{c2}\text{rand}_2(p_{gbest} - p_i(j)), \tag{5.10}$$

$$p_i(j+1) = p_i(j) + v_i(j+1), \tag{5.11}$$

where j is the current iteration, ω is the inertia weight, a_{c1} and a_{c2} are acceleration coefficients, and rand_1 and rand_2 are two independently uniform random numbers within range of $[0, 1]$. Algorithm 1 summarizes the steps performed in a particle swarm optimization.

Algorithm 1: Particle Swarm Optimization.

```
Result: Best fitness function.
for $i$ = 1:Maximum number of Particles do
    Initialize the particle's position and velocity;
    Evaluate the particle by calculating the fitness function $f$;
    Set $p_{ilbest}$ as the particle initial position;
    Set $p_{gbest}$ according to the previous evaluation;
    $i \leftarrow i + 1$;
end
while $j \leq$ Maximum number of Iterations do
    for $i$ = 1:Maximum number of Particles do
        Update particle's velocity $v_i(j+1)$ according to Equation (5.10);
        Update particle's position $p_i(j+1)$ according to Equation (5.11);
        Calculate the fitness function $f$ of the particle;
        if $f(p_i(j)) < f(p_{ilbest})$ then
            $f(p_{ilbest}) \leftarrow f(p_i(j))$;
            $p_{ilbest} \leftarrow p_i(j)$;
            if $f(p_{ilbest}) < f(p_{gbest})$ then
                $f(p_{gbest}) \leftarrow f(p_{ilbest})$;
                $p_{gbest} \leftarrow p_{ilbest}$;
            end
        end
        $i \leftarrow i + 1$;
    end
    $j \leftarrow j + 1$;
end
```

Here, the chaotic properties of the proposed digital tent map are used to generate pseudo-random numbers. For this purpose, a well-known hardware architecture of parallel PSO based on [5] is employed to address real-time optimization problems. The integration of the digital tent map as a PRNG within this hardware architecture is important. This is particularly significant due to the demand from various embedded applications requiring real-time solutions, such as online parameter estimation, neural network training, and mobile robot path planning [78].

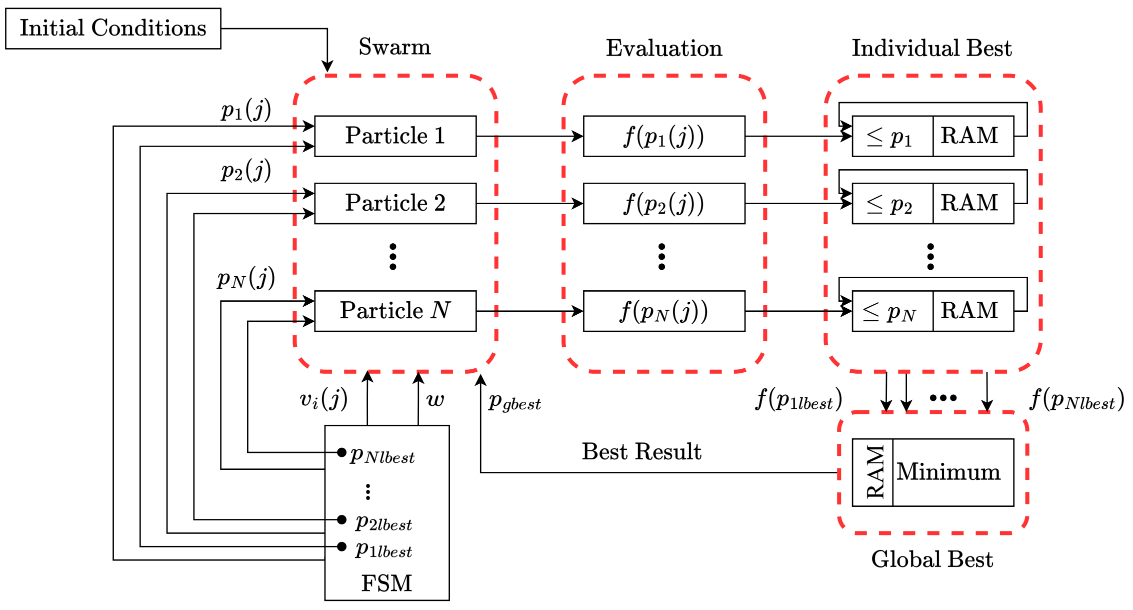
Figure 5.4-a shows the general hardware architecture used. This architecture was assembled in a parallel approach to improve running speed. It uses floating-point arithmetic to obtain a dynamic range for representing large and small numbers, and it is divided into five components: the Finite State Machine (FSM), the swarm unit, the evaluation unit, and the individual and global best detection units. While the FSM synchronizes the hardware operation, the swarm unit is responsible for updating the velocity and the position of each particle, according to Equations (5.10) and (5.11). Figure 5.4-b illustrates how the velocity and position of the particle are updated. The architecture is executed in five states, and in each state, all the operations are computed simultaneously. The architecture is composed of just one floating-point multiplier, one floating-point adder/subtractor, and the proposed digital tent map. To avoid using two more multipliers, instead of the tent map generating uniform random numbers within a range of $[0, 1]$, the digital circuit generates uniform random numbers within a range of $[0, a_{c1}]$ and $[0, a_{c2}]$. Additionally, as the proposed digital tent map performs the operations using fixed-point arithmetic, a fixed-to-float converter is applied to match the specified conditions of the hardware architecture for particle swarm optimization.

The evaluation unit carries out the fitness function evaluation. In this task, the hardware architecture was tested using benchmark functions applied in the literature. The chosen ones were the Sphere and Rosenbrock functions, where each one imposes different scenarios also found in real problems. The Sphere function is a unimodal and convex one, its domain is $x_k \in [-5.12, 5.12]$, for all $k = 1, \ldots, d$, and the global minimum is $x^* = 0$ at the position $x_k = 0$. The Sphere function is defined as [124]:

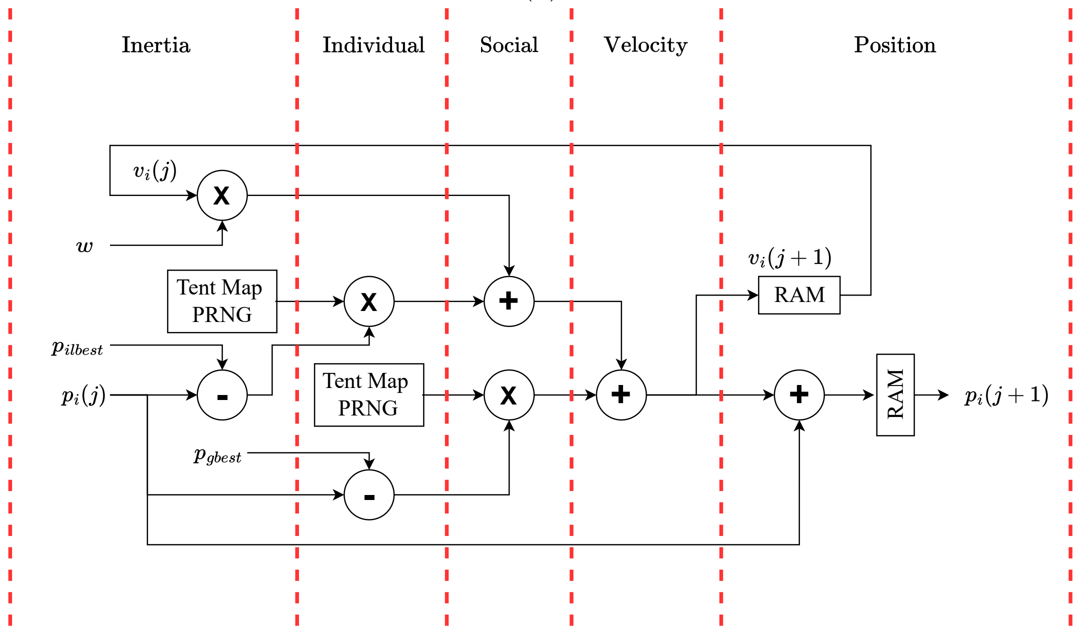
$$f(x) = \sum_{k=1}^{d} x_k^2. \tag{5.12}$$

The Rosenbrock function is typical non-convex and a multimodal function, its domain is $x_k \in [-5.12, 5.12]$, for all $k = 1, \ldots, d$, and the global minimum is $x^* = 0$ at the position $x_k = 1$. The Rosenbrock function is defined as [27]:

$$f(x) = \sum_{k=1}^{d/2} 100(x_{2k} - x_{2k-1}^2)^2 + (1 - x_{2k-1})^2. \tag{5.13}$$



(a)



(b)

Figure 5.4: Implemented architectures for PSO. (a) Hardware architecture for particle swarm optimization. (b) Particle architecture which uses the digital tent map.

The best individual unit is responsible to determine if the current fitness value of each particle is lower than its respective best fitness value. Thus, if $f(p_i(j)) < f(p_{ilbest})$, the best particle position p_{ilbest} of the particle i is updated and stored in a RAM (Random-Access Memory).

Lastly, the global best unit determines the minimum value among all the N best fitness values. If $f(p_{ilbest}) < f(p_{gbest})$, the best global position p_{gbest} is updated and stored in a RAM. It is important to note that these values are not effectively stored in a Block RAM (BRAM). The RAMs shown in Figures 5.4-a and 5.4-b are based on Flip-Flops.

The hardware architecture was implemented using the IEEE standard for floating-point arithmetic (IEEE 754) [43] with a bit-width of 27 bits. This bit-width consists of 1 bit for the signal, 8 bits for the biased exponent, and 18 bits for the mantissa. To generate pseudo-random numbers, the digital tent map was configured with a bit-width of 20 bits, following the guidelines provided by the authors in [5]. This configuration strikes the optimal balance between FPGA resources usage and the response of the PSO algorithm.

It is crucial to note that the digital tent map produces a pseudo-random number at every clock cycle, ensuring no impact on the latency of the proposed particle architecture [5]. The adoption of the proposed digital chaotic map as PRNG significantly improves the PSO algorithm's performance, replacing the previous use of a Linear Feedback Shift Register (LFSR) as a random number source. The LFSR technique is associated with significant drawbacks, including self-correlation and the potential for reconstructing parameters from a sequence of outputs [119].

Table 5.6 shows the synthesis results of the implemented hardware for particle swarm optimization. Utilizing the low-cost Artix 7 - XC7A100TFGG676-3 enables the efficient embedding of this optimization technique.

Table 5.6: Implementation results in the PSO design. The value in parentheses indicates the percentage of LUTs and FFs used relative to the available FPGA resources. `Project: Digital Tent Map / HPPSO_v6.xpr`.

Hardware for Particle Swarm Optimization					
Hardware	**Look-Up Table (LUT)**	**Flip-Flop (FF)**	**DSP**	**Frequency (MHz)**	**Power (mW)**
PSO-Sphere	15187 (23.95%)	9569 (7.55%)	20 (8.33%)	113.64	331
PSO-Rosenbrock	15036 (23.72%)	9019 (7.11%)	10 (4.17%)	113.64	295

The hardware architecture for particle swarm optimization, employing the digital tent map as a source of pseudo-random numbers, was validated using $N = 10$ particles to optimize problems with $d = 6$ dimensions. The parameter set used in the experiment is shown in Table 5.7. The linear decrease of the inertia weight is noteworthy, aiming to refine the final solution and improve the convergence results. The initial and maximum velocities were chosen to prevent the particle from leaving the search space.

All the benchmark functions were optimized 16 times, with each run utilizing distinct initial particle positions. This design ensured the preservation of swarm diversity, convergence, and the quality of results, allowing the estimation of the global minimum of these benchmark functions in each run. Table 5.8 summarizes the results obtained.

Ultimately, it is certainly important to mention that a crucial point of this application is the numerical conversion of the random number generated in fixed-point to floating-point format. Because there is a nonlinearity present in such conversion, the numbers generated by the tent map could lose their random characteristics. However, this is not the case.

Table 5.7: Experiment parameters for validating the proposed digital tent map as a pseudo-random number generator in hardware for particle swarm optimization.

Parameter	**Value**
Swarm size N	10
Dimension d	6
Maximum number of iterations	15000
Inertia weight w	Linear decrease from 0.8 to 0.1
Search space domain	$[-5.12, 5.12]$
Initial velocity	1
Maximum velocity	$[-3, 3]$

Table 5.8: Convergence results in the PSO design. `Documents evaluated: results_sphere.txt` and `results_rosenbrock.txt`.

Hardware for Particle Swarm Optimization				
	Solution of each fitness function after 15000 iterations			
Function	**Minimum**	**Mean**	**Median**	**Standard Deviation**
Sphere	$2.39 \cdot 10^{-38}$	$3.27 \cdot 10^{-38}$	$2.96 \cdot 10^{-38}$	$1.16 \cdot 10^{-38}$
Rosenbrock	$6.49 \cdot 10^{-8}$	0.0059	0.0026	0.0080

After the fixed-to-float-point conversion, the Kolmogorov-Smirnov and Runs tests were applied to ensure the generated sequence is pseudo-random and uniform. The P-values obtained for each test are 0.875 and 0.829, respectively, for a significance level of 0.01. Thus, as all P-values are greater than or equal to the significance level, the proposed digital tent map retained its uniform distribution and pseudo-random feature after conversion, as well as contributed to the PSO hardware performance.

5.4 Final Remarks

This chapter introduces a digital tent map, outlined through the VHDL hardware description language, and implemented on a low-cost FPGA utilizing 1's complement fixed-point numerical representation. The remarkable aspect of this implementation lies in its ability to significantly minimize the utilization of logical resources, thereby achieving a streamlined representation of the system while preserving its chaotic properties.

The findings highlight that employing basic logic elements, such as shift registers and XOR gates, alongside elementary operations like digit-complement, is ample for inducing chaotic behavior in the system. Furthermore, the method employed to prevent dynamical degradation has demonstrated satisfactory outcomes, proving to be notably less resource-intensive from a hardware perspective compared to alternative approaches present in the literature.

In exploring the applicability of the digital chaotic map proposed in this chapter, the results demonstrate that the map's chaotic behavior can effectively emulate a PRNG within a hardware architecture tailored for particle swarm optimization. This application proves particularly effective in optimizing two benchmark functions.

In brief, the advantages and disadvantages of the digital chaotic circuit suggested in this chapter can be summarized as follows:

- Advantages:
    1. The digital tent map presented in this chapter is characterized by its simplicity, demanding few logical resources. Notably, it exhibits low latency, delivering an output within just one clock cycle.
    2. Given its low consumption of logical resources, the digital chaotic circuit can be seamlessly integrated into other resource-constrained digital circuits requiring seemingly random signals for specific tasks and applications.
    3. The approach recommended to mitigate dynamical degradation in finite precision environments is readily applicable in hardware, offering a straightforward alternative to complex techniques employed by other methods.
- Disadvantages:
    1. As the precision of the proposed digital circuit decreases, quantization errors increase, leading the tent map's orbit to eventually adopt a cyclic pattern. Consequently, applications requiring a substantial amount of pseudo-random data must carefully assess the trade-off, considering the expansion of circuit precision. However, it is important to note that this choice comes at the cost of consuming additional logical resources.

Chapter 6

Conclusion and Final Remarks

6.1 Overview

Motivated by the scientific contributions of Henri Poincaré and Edward Lorenz to the field of chaos, many researchers have been studying chaos theory. In today's digital world, it is necessary to digitize chaotic systems or conceive completely digital chaotic systems in order to simulate them or create useful applications. However, implementing these systems in a digital domain is a complex task. As previously seen in Chapter 1, different hardware and software setups can affect the performance of chaotic systems. In computer simulations, even with the same initial conditions, but with different machine configurations, the chaotic pseudo-orbits of the system may diverge, as noticed by Chen's hyperchaotic system, Chua's circuit, Rössler system, and Sprott's jerk circuit. This divergence also extends to mathematically equivalent chaotic systems of such models when the order to perform their arithmetic operations differs. Furthermore, when employing the tent map, the adopted number of bits and number representation can induce dynamical degradation of the chaotic properties of the map, showing periodic orbits instead of chaotic behavior.

Drawing from these insights, this thesis delves into three distinct approaches explored in detail throughout the document. First, it was demonstrated that binary numbers represented as elements of a finite field and iterated with the logistic map do not propagate errors. This makes the method robust against hardware and software discrepancies, not affecting, for example, chaos-based encryption schemes. The method was then applied to a new image encryption algorithm. By adopting an easy-to-implement logistic map using a 64-degree irreducible polynomial on $GF(2)[x]$ as a pseudo-random number generator, the designed chaos-based scheme proved to be reliable and secure against cyberattacks.

In the second approach, this thesis shows a correlation between the divergence of chaotic pseudo-orbits and the Lyapunov exponent. The concept of interval extensions of mathematical models is extended to chaotic circuits, enabling the measurement of the largest Lyapunov exponent through the separation of pseudo-orbits in equivalent circuits.

This technique has been proven to correctly estimate the Lyapunov exponent of six chaotic circuits. Additionally, the method is simple and user-friendly, as it does not require the user to have a deep knowledge of chaos theory or dynamical systems. The user only needs to know how to use the SPICE program interface tools.

Ultimately, a hardware-efficient perturbation method in a simple digital tent map was presented. By employing 1's complement fixed-point number representation, this approach resulted in the development of a straightforward and flexible digital chaotic map architecture implemented on an FPGA. The design, requiring only 33 components (16 XOR gates, 1 multiplexer, and 16 D-type flip-flops) for the 16-bit precision version, achieves an efficient ratio of 2.1 logical components per bit. This represents a significant reduction of at least 58% compared to state-of-the-art references. The logic-gate level perturbation method, primarily using an XOR gate, effectively mitigated dynamical degradation. The system's chaotic and statistical properties were preserved and validated through various analyses, including the largest Lyapunov exponent and time series. Furthermore, the digital tent map was confirmed as a pseudo-random number generator through statistical tests, showcasing its practical application in hardware architecture for particle swarm optimization.

6.2 Future Research

The results presented in this thesis open up new frontiers and research opportunities. Below are some possibilities.

Galois field is a promising area, but it is important to be aware of the dynamical degradation of chaotic systems over finite fields. Yang and Liao [139, 140] have already analyzed the period of the logistic map over a finite field. However, it would be interesting to apply some well-known methods to mitigate dynamical degradation, briefly described in Chapter 1, in some chaotic systems over finite fields and evaluate their effectiveness.

One research trend in the conception of chaos-based encryption schemes is to consider high-order irreducible polynomials in Galois fields. This is because using high-order polynomials to build a matrix representation of finite fields might increase the complexity, security, and speed of the encryption method. Additionally, such a method can be implemented on FPGAs, which can lead to the best allocation of resources and the creation of lightweight encryption methods.

Regarding the estimation of Lyapunov exponents using SPICE-like programs, a future research topic could involve investigating extensions of the proposed approach in this thesis to a wide range of hyperchaotic circuits to estimate the second-largest Lyapunov exponent, as this is important for characterizing hyperchaos.

While this paper minimized the tent map to save FPGA resources and mitigate dynamical degradation, there are many other chaotic maps that could be studied and evaluated. For example, some other binary shift chaotic maps, such as Arnold's cat map, Baker's map, and ship map, should be studied. Additionally, the use of such maps in other demanding applications should be assessed.

Another topic to examine is the creation of genuinely digital chaotic systems. This thesis implemented and simulated digitized chaotic systems that have been widely used in many areas. However, the exploration of creating genuinely digital systems with chaotic properties, as scrutinized by Wang et al. [131], remains a relatively uncharted territory. Creating a chaotic circuit using Boolean algebra and binary arithmetic, and proving its chaoticity, is certainly worth the effort.

6.3 Additional Doctoral Publications

In addition to the publications that encapsulate the contributions of this thesis, the doctoral student has made other valuable contributions to the scientific community during their postgraduate studies. This section provides an overview of publications and submissions closely related to this thesis, arranged chronologically in Table 6.1, taking into account the publication status.

Table 6.1: Details of each additional publication achieved by the doctoral student during their postgraduate studies.

Status	Publication
Published	[13] - Cardoso, M. B. R., Silva, S. S., Nardo, L. G., Passos, R. M., Nepomuceno, E. G., and Arias-Garcia, J. (2021). A new PRNG hardware architecture based on an exponential chaotic map. In *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, volume 2021-May, pages 1–5. IEEE
Published	[117] - Silva, S. S., Cardoso, M., Nardo, L., Nepomuceno, E., Hübner, M., and Arias-Garcia, J. (2023). A new chaos-based PRNG hardware architecture using the HUB fixed-point format. *IEEE Transactions on Instrumentation and Measurement*, 72:1–8
Published	[61] - Lima, A. M., Nardo, L. G., Nepomuceno, E., Arias-Garcia, J., and Yudi, J. (2023). Design of an advanced System-on-Chip architecture for chaotic image encryption. In *2023 36th SBC/SBMicro/IEEE/ACM Symposium on Integrated Circuits and Systems Design (SBCCI)*, pages 1–6. IEEE
Published	[84] - Nazaré, T., Nardo, L., Arias-Garcia, J., and Nepomuceno, E. (2023). Ensuring resilience in ocean energy power plants: A review on cybersecurity. In *2023 European Wave and Tidal Energy Conference (EWTEC)*, volume 15, pages 1–9
Accepted	[81] - Nardo, L. G., Nepomuceno, E., Arias-Garcia, J., Chen, T. M., and Butusov, D. N. (2024b). Dual authentication on a secure communication channel to image transmission. *Journal of Vibration Testing and System Dynamics*, pages 1–12

The paper [13] explores a novel hardware architecture for modeling a chaotic system and creating a pseudo-random number generator. Cardoso et al. [13] introduced a new methodological approach for calculating the exponential function. Adopting the exponential chaotic map, an approximation of Euler's number by a finite series and Horner's method have been undertaken to minimize the proposed hardware.

In a separate study, the article [117] investigates the application of fixed-point number representation using the Half-Unit-Biased (HUB) format within a bi-coupled chaotic map. The research reveals the significant advantages of the HUB format in designing efficient hardware architectures, particularly in achieving rounding to the nearest number through truncation. The paper also demonstrates the utility of the bi-coupled chaotic map in generating pseudo-random numbers.

Another contribution, presented in [61], exploits the finite-precision error in the divergence of chaotic pseudo-orbits of Chua's circuit. The study employs the latest techniques in hardware/software co-design for embedded systems to construct an efficient architecture for encrypting images, leveraging finite-precision error as a source of unpredictability.

While interning at Maynooth University and the Center of Ocean Energy Research, the doctoral student acknowledged the potential of ocean energy for power generation. Given the critical role of power generation in society and the economy, safeguarding ocean energy power plants against cyberattacks is essential. In [84], the paper conducts a thorough survey on the implementation of cybersecurity measures in ocean energy, underscoring the significance of minimizing vulnerabilities in power plant cybersecurity.

Lastly, the forthcoming paper [81] accepted for publication in the "Journal of Vibration Testing and System Dynamics," uses the proposed chaos-based encryption algorithm from [83] to suggest a method to transmit images where transmission channels may be vulnerable to attacks. The technique employs a dual authentication factor, combining an encrypted image from an encryption scheme with a chaotic signal generated by two synchronized Rössler systems. Breaking the image transmission method requires knowledge of both the encryption algorithm's seed and the initial conditions used to generate the Rössler system orbits.

References

[1] Aguirre, L. A., Maquet, J., and Letellier, C. (2002). Induced one-parameter bifurcations in identified nonlinear dynamical models. *International Journal of Bifurcation and Chaos*, 12(1):135–145.

[2] Ahmed, H. A., Zolkipli, M. F., and Ahmad, M. (2019). A novel efficient substitution-box design based on firefly algorithm and discrete chaotic map. *Neural Computing and Applications*, 31(11):7201–7210.

[3] Alatas, B., Akin, E., and Ozer, A. B. (2009). Chaos embedded particle swarm optimization algorithms. *Chaos, Solitons & Fractals*, 40(4):1715–1734.

[4] Antonelli, M., Micco, L. D., Larrondo, H., and Rosso, O. (2018). Complexity of simple, switched and skipped chaotic maps in finite precision. *Entropy*, 20(2):135.

[5] Arboleda, D. M. M., Llanos, C. H., Coelho, L. S., and Ayala-Rincón, M. (2009). Hardware architecture for particle swarm optimization using floating-point arithmetic. In *2009 9th International Conference on Intelligent Systems Design and Applications (ISDA)*, pages 243–248. IEEE.

[6] Balcerzak, M., Pikunov, D., and Dabrowski, A. (2018). The fastest, simplified method of Lyapunov exponents spectrum estimation for continuous-time dynamical systems. *Nonlinear Dynamics*, 94(4):3053–3065.

[7] Banks, J., Brooks, J., Cairns, G., Davis, G., and Stacey, P. (1992). On Devaney's definition of chaos. *The American Mathematical Monthly*, 99(4):332.

[8] Baranovsky, A. and Daems, D. (1995). Design of one-dimensional chaotic maps with prescribed statistical properties. *International Journal of Bifurcation and Chaos*, 5(6):1585–1598.

[9] Borah, M., Gayan, A., Sharma, J. S., Chen, Y., Wei, Z., and Pham, V.-T. (2022). Is fractional-order chaos theory the new tool to model chaotic pandemics as Covid-19? *Nonlinear Dynamics*, 109(2):1187–1215.

[10] Brock, W. A. (1986). Distinguishing random and deterministic systems: Abridged version. *Journal of Economic Theory*, 40(1):168–195.

[11] Broumandnia, A. (2020). Image encryption algorithm based on the finite fields in chaotic maps. *Journal of Information Security and Applications*, 54:102553.

[12] Cao, L.-C., Luo, Y.-L., Qiu, S.-H., and Liu, J.-X. (2015). A perturbation method to the tent map based on Lyapunov exponent and its application. *Chinese Physics B*, 24(10):100501.

[13] Cardoso, M. B. R., Silva, S. S., Nardo, L. G., Passos, R. M., Nepomuceno, E. G., and Arias-Garcia, J. (2021). A new PRNG hardware architecture based on an exponential chaotic map. In *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, volume 2021-May, pages 1–5. IEEE.

[14] Carrano, E. G., Soares, L. A. E., Takahashi, R. H. C., Saldanha, R. R., and Neto, O. M. (2006). Electric distribution network multiobjective design using a problem-specific genetic algorithm. *IEEE Transactions on Power Delivery*, 21(2):995–1005.

[15] Chai, X., Chen, Y., and Broyde, L. (2017). A novel chaos-based image encryption algorithm using DNA sequence operations. *Optics and Lasers in Engineering*, 88:197–213.

[16] Chang, D., Li, Z., Wang, M., and Zeng, Y. (2018). A novel digital programmable multi-scroll chaotic system and its application in FPGA-based audio secure communication. *AEU - International Journal of Electronics and Communications*, 88:20–29.

[17] Chen, X., Qian, S., Yu, F., Zhang, Z., Shen, H., Huang, Y., Cai, S., Deng, Z., Li, Y., and Du, S. (2020). Pseudorandom number generator based on three kinds of four-wing memristive hyperchaotic system and its application in image encryption. *Complexity*, 2020:1–17.

[18] Chou, H.-G., Chuang, C.-F., Wang, W.-J., and Lin, J.-C. (2013). A fuzzy-model-based chaotic synchronization and its implementation on a secure communication system. *IEEE Transactions on Information Forensics and Security*, 8(12):2177–2185.

[19] Chua, L. O., Wu, C. W., Huang, A., and Zhong, G.-Q. (1993a). A universal circuit for studying and generating chaos. I. Routes to chaos. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 40(10):732–744.

[20] Chua, L. O., Wu, C. W., Huang, A., and Zhong, G.-Q. (1993b). A universal circuit for studying and generating chaos. II. Strange attractors. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 40(10):745–761.

[21] Costa, A. L. X., Silva, C. A. D., Torquato, M. F., and Fernandes, M. A. C. (2019). Parallel implementation of particle swarm optimization on FPGA. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 66(11):1875–1879.

[22] Crampin, M. and Heal, B. (1994). On the chaotic behaviour of the tent map. *Teaching Mathematics and its Applications*, 13(2):83–89.

[23] Cvitanović, P., Artuso, R., Mainieri, R., Tanner, G., and Vattay, G. (2020). *Chaos: Classical and Quantum*. ChaosBook.org.

[24] Deane, J. H. B. and Hamill, D. C. (1990). Instability, subharmonics, and chaos in power electronic systems. *IEEE Transactions on Power Electronics*, 5(3):260–268.

[25] Devaney, R. L. (2022). *An introduction to chaotic dynamical systems*. CRC Press.

[26] Diaconu, A.-V. and Dascalescu, A. C. (2017). Correlation distribution of adjacent pixels randomness test for image encryption. *Proceedings of the Romanian Academy Series A - Mathematics, Physics, Technical Sciences, Information Science*, 18:351–359.

[27] Dixon, L. C. W. and Mills, D. J. (1994). Effect of rounding errors on the variable metric method. *Journal of Optimization Theory and Applications*, 80(1):175–179.

[28] Eckmann, J.-P. and Ruelle, D. (1985). Ergodic theory of chaos and strange attractors. *Reviews of Modern Physics*, 57(3):617–656.

[29] Fister, I., Perc, M., Kamal, S. M., and Fister, I. (2015). A review of chaos-based firefly algorithms: Perspectives and research challenges. *Applied Mathematics and Computation*, 252:155–165.

[30] Fridrich, J. (1997). Image encryption based on chaotic maps. In *1997 IEEE International Conference on Systems, Man, and Cybernetics*, volume 2, pages 1105–1110. IEEE.

[31] Gaffar, A. A., Mencer, O., Luk, W., and Cheung, P. Y. K. (2004). Unifying bit-width optimisation for fixed-point and floating-point designs. In *2004 12th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM)*, pages 79–88. IEEE.

[32] Gan, Z.-H., Chai, X.-L., Han, D.-J., and Chen, Y.-R. (2019). A chaotic image encryption algorithm based on 3-D bit-plane permutation. *Neural Computing and Applications*, 31(11):7111–7130.

[33] Gao, T. and Chen, Z. (2008). A new image encryption algorithm based on hyper-chaos. *Physics Letters A*, 372(4):394–400.

[34] Gao, T., Chen, Z., Yuan, Z., and Chen, G. (2006). A hyperchaos generated from Chen's system. *International Journal of Modern Physics C*, 17(4):471–478.

[35] Garcés, H. and Flores, B. C. (2006). Statistical analysis of Bernoulli, logistic, and tent maps with applications to radar signal design. In *Radar Sensor Technology X*, volume 6210, pages 1–10. SPIE.

[36] Guan, Z.-H., Huang, F., and Guan, W. (2005). Chaos-based image encryption algorithm. *Physics Letters A*, 346(1-3):153–157.

[37] Gupta, R., Pachauri, R., and Singh, A. K. (2019). Image encryption method using dependable multiple chaotic logistic functions. *International Journal of Information Security and Privacy*, 13(4):53–67.

[38] Gupta, V., Mittal, M., and Mittal, V. (2020). Chaos theory: An emerging tool for arrhythmia detection. *Sensing and Imaging*, 21(1):10.

[39] Haroun, M. F. and Gulliver, T. A. (2015). Real-time image encryption using a low-complexity discrete 3D dual chaotic cipher. *Nonlinear Dynamics*, 82:1523–1535.

[40] Herring, C. and Palmore, J. I. (1989). Random number generators are chaotic. *ACM SIGPLAN Notices*, 24(11):76–79.

[41] Hu, T., Liu, Y., Gong, L.-H., and Ouyang, C.-J. (2017). An image encryption scheme combining chaos with cycle operation for DNA sequences. *Nonlinear Dynamics*, 87(1):51–66.

[42] Hunt, B. R. and Ott, E. (2015). Defining chaos. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 25(9):097618.

[43] Institute of Electrical and Electronics Engineers (IEEE) (2019). IEEE standard for floating-point arithmetic. *IEEE Std 754-2019*, pages 1–82.

[44] Jallouli, O., Assad, S. E., Chetto, M., Lozi, R., and Caragata, D. (2015). A novel chaotic generator based on weakly-coupled discrete skewtent maps. In *2015 10th International Conference for Internet Technology and Secured Transactions (ICITST)*, pages 38–43. IEEE.

[45] Jeoti, V. and Khan, M. A. (2015). On the enlargement of robust region of chaotic tent map for the use in key based substitution-box (S-Box). *Journal of Computer Science*, 11(3):517–525.

[46] Kafi, D. A., Jamal, R. K., Mosa, K. I., and Alnaemme, K. A. (2019). The numerical and experimental work of chaos system in three dimensions phase space using Rössler circuit. *Indian Journal of Natural Sciences*, 9(52):16699–16710.

[47] Kantz, H. (1994). A robust method to estimate the maximal Lyapunov exponent of a time series. *Physics Letters A*, 185(1):77–87.

[48] Karimov, T., Nepomuceno, E. G., Druzhina, O., Karimov, A., and Butusov, D. (2019). Chaotic oscillators as inductive sensors: Theory and practice. *Sensors*, 19(19):4314.

[49] Kaur, G. and Arora, S. (2018). Chaotic whale optimization algorithm. *Journal of Computational Design and Engineering*, 5(3):275–284.

[50] Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. In *1995 International Conference on Neural Networks (ICNN)*, volume 4, pages 1942–1948. IEEE.

[51] Kennedy, M. P. (1993a). Three steps to chaos. I. Evolution. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 40(10):640–656.

[52] Kennedy, M. P. (1993b). Three steps to chaos. II. A Chua's circuit primer. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 40(10):657–674.

[53] Khani, F. and Ahmadi, A. (2013). Digital realization of twisted tent map and ship map with LFSR as a pseudo-chaos generator. In *2013 13rd International Conference on Computer and Knowledge Engineering (ICCKE)*, pages 207–212. IEEE.

[54] Kim, B. J. and Choe, G. H. (2010). High precision numerical estimation of the largest Lyapunov exponent. *Communications in Nonlinear Science and Numerical Simulation*, 15(5):1378–1384.

[55] Kodba, S., Perc, M., and Marhl, M. (2005). Detecting chaos from a time series. *European Journal of Physics*, 26(1):205–215.

[56] Kwok, H. S. and Tang, W. K. S. (2007). A fast image encryption system based on chaotic maps with finite precision representation. *Chaos, Solitons & Fractals*, 32(4):1518–1529.

[57] Lai, Q., Kuate, P. D. K., Liu, F., and Iu, H. H.-C. (2020). An extremely simple chaotic system with infinitely many coexisting attractors. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 67(6):1129–1133.

[58] Li, C., Luo, G., Qin, K., and Li, C. (2017). An image encryption scheme based on chaotic tent map. *Nonlinear Dynamics*, 87(1):127–133.

[59] Li, S., Chen, G., and Mou, X. (2005). On the dynamical degradation of digital piecewise linear chaotic maps. *International Journal of Bifurcation and Chaos*, 15(10):3119–3151.

[60] Lidl, R. and Niederreiter, H. (1994). *Introduction to finite fields and their applications*. Cambridge University Press.

[61] Lima, A. M., Nardo, L. G., Nepomuceno, E., Arias-Garcia, J., and Yudi, J. (2023). Design of an advanced System-on-Chip architecture for chaotic image encryption. In *2023 36th SBC/SBMicro/IEEE/ACM Symposium on Integrated Circuits and Systems Design (SBCCI)*, pages 1–6. IEEE.

[62] Linz, S. J. and Sprott, J. C. (1999). Elementary chaotic flow. *Physics Letters A*, 259(3-4):240–245.

[63] Liu, B., Wang, L., Jin, Y.-H., Tang, F., and Huang, D.-X. (2005). Improved particle swarm optimization combined with chaos. *Chaos, Solitons & Fractals*, 25(5):1261–1271.

[64] Liu, Y., Luo, Y., Song, S., Cao, L., Liu, J., and Harkin, J. (2017). Counteracting dynamical degradation of digital chaotic Chebyshev map via perturbation. *International Journal of Bifurcation and Chaos*, 27(3):1750033.

[65] Lorenz, E. N. (1963). Deterministic nonperiodic flow. *Journal of the Atmospheric Sciences*, 20(2):130–141.

[66] Luo, Y., Du, M., and Liu, J. (2015). A symmetrical image encryption scheme in wavelet and time domain. *Communications in Nonlinear Science and Numerical Simulation*, 20(2):447–460.

[67] Matsuoka, C. and Hiraide, K. (2015). Computation of entropy and Lyapunov exponent by a shift transform. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 25(10):103110.

[68] May, R. M. (1976). Simple mathematical models with very complicated dynamics. *Nature*, 261(5560):459–467.

[69] McEliece, R. J. (1987). *Finite fields for computer scientists and engineers*. Kluwer Academic Publishers.

[70] Mendes, E. M. A. M. and Nepomuceno, E. G. (2016). A very simple method to calculate the (positive) largest Lyapunov exponent using interval extensions. *International Journal of Bifurcation and Chaos*, 26(13):1650226.

[71] Mondal, B., Singh, S., and Kumar, P. (2019). A secure image encryption scheme based on cellular automata and chaotic skew tent map. *Journal of Information Security and Applications*, 45:117–130.

[72] Monteiro, L. H. A. (2019). *Sistemas dinâmicos*. Livraria da Física Editorial.

[73] Moore, R. E. (1979). *Methods and applications of interval analysis*. SIAM.

[74] Moore, R. E., Kearfott, R. B., and Cloud, M. J. (2009). *Introduction to interval analysis*. SIAM.

[75] Mrdovic, S. and Perunicic, B. (2008). Kerckhoffs' principle for intrusion detection. In *2008 13th International Telecommunications Network Strategy and Planning Symposium (NETWORKS)*, pages 1–14. IEEE.

[76] Musielak, Z. E. and Quarles, B. (2014). The three-body problem. *Reports on Progress in Physics*, 77(6):065901.

[77] Muthuswamy, B. and Banerjee, S. (2015). *A route to chaos using FPGAs*. Springer International Publishing.

[78] Muñoz, D. M., Llanos, C. H., Coelho, L. S., and Ayala-Rincón, M. (2010). Hardware particle swarm optimization based on the attractive-repulsive scheme for embedded applications. In *2010 International Conference on Reconfigurable Computing and FPGAs (ReConFig)*, pages 55–60. IEEE.

[79] Nardo, L., Nepomuceno, E., Muñoz, D., Butusov, D., and Arias-Garcia, J. (2023). A hardware-efficient perturbation method to the digital tent map. *Electronics*, 12(8):1953.

[80] Nardo, L. G., Nazaré, T. E., Nepomuceno, E., Arias-Garcia, J., and Butusov, D. N. (2024a). Computation of the largest Lyapunov exponent using SPICE-like programs. In *2024 7th IFAC Conference on Analysis and Control of Nonlinear Dynamics and Chaos (IFAC ACNDC)*, pages 1–6.

[81] Nardo, L. G., Nepomuceno, E., Arias-Garcia, J., Chen, T. M., and Butusov, D. N. (2024b). Dual authentication on a secure communication channel to image transmission. *Journal of Vibration Testing and System Dynamics*, pages 1–12.

[82] Nardo, L. G., Nepomuceno, E. G., Arias-Garcia, J., and Butusov, D. N. (2019). Image encryption using finite-precision error. *Chaos, Solitons & Fractals*, 123:69–78.

[83] Nardo, L. G., Nepomuceno, E. G., Bastos, G. T., Santos, T. A., Butusov, D. N., and Arias-Garcia, J. (2021). A reliable chaos-based cryptography using Galois field. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(9):091101.

[84] Nazaré, T., Nardo, L., Arias-Garcia, J., and Nepomuceno, E. (2023). Ensuring resilience in ocean energy power plants: A review on cybersecurity. In *2023 European Wave and Tidal Energy Conference (EWTEC)*, volume 15, pages 1–9.

[85] Nazaré, T. E., Nepomuceno, E. G., Martins, S. A. M., and Butusov, D. N. (2020). A note on the reproducibility of chaos simulation. *Entropy*, 22(9):953.

[86] Nepomuceno, E. G. (2014). Convergence of recursive functions on computers. *The Journal of Engineering*, 2014(10):560–562.

[87] Nepomuceno, E. G., Lima, A. M., Arias-García, J., Perc, M., and Repnik, R. (2019a). Minimal digital chaotic system. *Chaos, Solitons & Fractals*, 120:62–66.

[88] Nepomuceno, E. G. and Martins, S. A. M. (2016). A lower bound error for free-run simulation of the polynomial NARMAX. *Systems Science & Control Engineering*, 4(1):50–58.

[89] Nepomuceno, E. G., Martins, S. A. M., Lacerda, M. J., and Mendes, E. M. A. M. (2018). On the use of interval extensions to estimate the largest Lyapunov exponent from chaotic data. *Mathematical Problems in Engineering*, 2018:1–8.

[90] Nepomuceno, E. G. and Mendes, E. M. A. M. (2017). On the analysis of pseudo-orbits of continuous chaotic nonlinear systems simulated using discretization schemes in a digital computer. *Chaos, Solitons & Fractals*, 95:21–32.

[91] Nepomuceno, E. G., Nardo, L. G., Arias-Garcia, J., Butusov, D. N., and Tutueva, A. (2019b). Image encryption based on the pseudo-orbits from 1D chaotic map. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(6):061101.

[92] Noakes, L. (1991). The Takens embedding theorem. *International Journal of Bifurcation and Chaos*, 1(4):867–872.

[93] Norouzi, B. and Mirzakuchaki, S. (2014). A fast color image encryption algorithm based on hyper-chaotic systems. *Nonlinear Dynamics*, 78(2):995–1015.

[94] Oseledets, V. I. (1968). The multiplicative ergodic theorem: The Lyapunov characteristic numbers of dynamical systems. *Transactions of the Moscow Mathematical Society*, 19:179–210.

[95] Overton, M. L. (2001). *Numerical computing with IEEE floating point arithmetic*. SIAM.

[96] Palacios-Luengas, L., Duchen-Sánchez, G. I., Aragón-Vera, J. L., and Vázquez-Medina, R. (2012). Digital noise generator design using inverted 1D tent chaotic map. *VLSI Design*, 2012:1–10.

[97] Parhami, B. (2010). *Computer arithmetic: Algorithms and hardware designs*. Oxford University Press.

[98] Parlitz, U. (1992). Identification of true and spurious Lyapunov exponents from time series. *International Journal of Bifurcation and Chaos*, 2(1):155–165.

[99] Peitgen, H.-O., Jürgens, H., and Saupe, D. (2004). *Chaos and fractals: New frontiers of science*. Springer.

[100] Peixoto, M. L. C., Nepomuceno, E. G., Martins, S. A. M., and Lacerda, M. J. (2018). Computation of the largest positive Lyapunov exponent using rounding mode and recursive least square algorithm. *Chaos, Solitons & Fractals*, 112:36–43.

[101] Piper, J. R. and Sprott, J. C. (2010). Simple autonomous chaotic circuits. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 57(9):730–734.

[102] Qayyum, A., Ahmad, J., Boulila, W., Rubaiee, S., A., Masood, F., Khan, F., and Buchanan, W. J. (2020). Chaos-based confusion and diffusion of image pixels using dynamic substitution. *IEEE Access*, 8:140876–140895.

[103] Qiu, M., Yu, S., Wen, Y., Lü, J., He, J., and Lin, Z. (2017). Design and FPGA implementation of a universal chaotic signal generator based on the verilog HDL fixed-point algorithm and state machine control. *International Journal of Bifurcation and Chaos*, 27(3):1750040.

[104] Ramalingam, B., Ravichandran, D., Annadurai, A. A., Rengarajan, A., and Rayappan, J. B. B. (2018). Chaos triggered image encryption - A reconfigurable security solution. *Multimedia Tools and Applications*, 77(10):11669–11692.

[105] Robinson, C. (2008). What is a chaotic attractor? *Qualitative Theory of Dynamical Systems*, 7(1):227–236.

[106] Rodríguez-Orozco, E., García-Guerrero, E., Inzunza-Gonzalez, E., López-Bonilla, O., Flores-Vergara, A., Cárdenas-Valdez, J., and Tlelo-Cuautle, E. (2018). FPGA-based chaotic cryptosystem by using voice recognition as access key. *Electronics*, 7(12):414.

[107] Rosenstein, M. T., Collins, J. J., and Luca, C. J. D. (1993). A practical method for calculating largest Lyapunov exponents from small data sets. *Physica D: Nonlinear Phenomena*, 65:117–134.

[108] Rybin, V., Babkin, I., Kvitko, D., Karimov, T., Nardo, L., Nepomuceno, E., and Butusov, D. (2023). Estimating optimal synchronization parameters for coherent chaotic communication systems in noisy conditions. *Chaos Theory and Applications*, 5(3):141–152.

[109] Rössler, O. E. (1976). An equation for continuous chaos. *Physics Letters A*, 57(5):397–398.

[110] Sankpal, P. R. and Vijaya, P. A. (2022). Fusion implementation of security mechanisms for secured transmission of images. In *2022 3rd International Conference for Emerging Technology (INCET)*, pages 1–8. IEEE.

[111] Sano, M. and Sawada, Y. (1985). Measurement of the Lyapunov spectrum from a chaotic time series. *Physical Review Letters*, 55:1082–1085.

[112] Santos, T. A., Magalhães, E. P., Fiorio, D. R., and Nepomuceno, E. G. (2019). On the reliability of computational chaos-based cryptography for information exchange. In *2019 Simpósio Brasileiro de Automação Inteligente (SBAI)*, pages 2776–2781. Galoá.

[113] Sayed, W. S., Mohamed, S. M., Elwakil, A. S., Said, L. A., and Radwan, A. G. (2022). Numerical sensitivity analysis and hardware verification of a transiently-chaotic attractor. *International Journal of Bifurcation and Chaos*, 32(07):2250103.

[114] Sayed, W. S., Radwan, A. G., Fahmy, H. A. H., and El-Sedeek, A. (2020). Software and hardware implementation sensitivity of chaotic systems and impact on encryption applications. *Circuits, Systems, and Signal Processing*, 39(11):5638–5655.

[115] Shah, D. and Shah, T. (2020). A novel discrete image encryption algorithm based on finite algebraic structures. *Multimedia Tools and Applications*, 79(37-38):28023–28042.

[116] Shah, T., Ali, A., Khan, M., Farooq, G., and Andrade, A. A. (2020). Galois ring $GR(2^3, 8)$ dependent 24×24 S-Box design: An RGB image encryption application. *Wireless Personal Communications*, 113(2):1201–1224.

[117] Silva, S. S., Cardoso, M., Nardo, L., Nepomuceno, E., Hübner, M., and Arias-Garcia, J. (2023). A new chaos-based PRNG hardware architecture using the HUB fixed-point format. *IEEE Transactions on Instrumentation and Measurement*, 72:1–8.

[118] Singh, J. P. and Roy, B. K. (2019). Simplest hyperchaotic system with only one piecewise linear term. *Electronics Letters*, 55(7):378–380.

[119] Singh, S. K., Gupta, M. D., Mani, S., and Chauhan, R. K. (2020). Design of LFSR circuit based on high performance XOR gate. In *2020 International Conference on Electrical and Electronics Engineering (ICE3)*, pages 656–660. IEEE.

[120] Sinha, R. K. and Sahu, S. S. (2019). Adaptive firefly algorithm based optimized key generation for image security. *Journal of Intelligent & Fuzzy Systems*, 36(5):4437–4447.

[121] Sprott, J. C. (2000). Simple chaotic systems and circuits. *American Journal of Physics*, 68(8):758–763.

[122] Sprott, J. C. (2011). A new chaotic jerk circuit. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 58(4):240–243.

[123] Sreenath, H. and Narayanan, G. (2018). FPGA implementation of pseudo chaos-signal generator for secure communication systems. In *2018 International Conference*

on Advances in Computing, Communications and Informatics (ICACCI), pages 804–807. IEEE.

[124] Stacey, A., Jancic, M., and Grundy, I. (2003). Particle swarm optimization with mutation. In *2003 Congress on Evolutionary Computation (CEC)*, volume 2, pages 1425–1430. IEEE.

[125] Stoyanov, B. and Kordov, K. (2014). Novel image encryption scheme based on Chebyshev polynomial and Duffing map. *The Scientific World Journal*, 2014:1–11.

[126] Takens, F. (1981). Detecting strange attractors in turbulence. In *Dynamical Systems and Turbulence, Warwick 1980*, pages 366–381.

[127] Vipin, K. and Fahmy, S. A. (2019). FPGA dynamic and partial reconfiguration. *ACM Computing Surveys*, 51(4):1–39.

[128] Walker, J. (2008). ENT: A pseudorandom number sequence test program.

[129] Wang, H., Song, B., Liu, Q., Pan, J., and Ding, Q. (2014). FPGA design and applicable analysis of discrete chaotic maps. *International Journal of Bifurcation and Chaos*, 24(4):1450054.

[130] Wang, N., Di, G., Lv, X., Hou, M., Liu, D., Zhang, J., and Duan, X. (2019). Galois field-based image encryption for remote transmission of tumor ultrasound images. *IEEE Access*, 7:49945–49950.

[131] Wang, Q., Yu, S., Li, C., Lu, J., Fang, X., Guyeux, C., and Bahi, J. M. (2016). Theoretical design and FPGA-based implementation of higher-dimensional digital chaotic systems. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 63(3):401–412.

[132] Wang, X.-Y. and Wang, L.-L. (2011). A new perturbation method to the tent map and its application. *Chinese Physics B*, 20(5):050509.

[133] Wang, X.-Y. and Wang, Q. (2014). A fast image encryption algorithm based on only blocks in cipher text. *Chinese Physics B*, 23(3):030503.

[134] Wei, Z., Rajagopal, K., Zhang, W., Kingni, S. T., and Akgül, A. (2018). Synchronisation, electronic circuit implementation, and fractional-order analysis of 5D ordinary differential equations with hidden hyperchaotic attractors. *Pramana*, 90(4):50.

[135] Wiggins, S. (1992). *Chaotic transport in dynamical systems*. Springer.

[136] Wolf, A., Swift, J. B., Swinney, H. L., and Vastano, J. A. (1985). Determining Lyapunov exponents from a time series. *Physica D: Nonlinear Phenomena*, 16(3):285–317.

[137] Wu, Y., Noonan, J. P., and Agaian, S. (2011). NPCR and UACI randomness tests for image encryption. *Journal of Selected Areas in Telecommunications*, 4:31–38.

[138] Xin, G. and Jue-Bang, Y. (2005). Chaos and chaotic control in a fractional-order electronic oscillator. *Chinese Physics*, 14(5):908–913.

[139] Yang, B. and Liao, X. (2017). Period analysis of the logistic map for the finite field. *Science China Information Sciences*, 60(2):022302.

[140] Yang, B. and Liao, X. (2018). Some properties of the logistic map over the finite field and its application. *Signal Processing*, 153:231–242.

[141] Yao, T.-L., Liu, H.-F., Xu, J.-L., and Li, W.-F. (2012). Estimating the largest Lyapunov exponent and noise level from chaotic time series. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 22(3):033102.

[142] Yin, T. and Wang, Y. (2022). Predicting the price of WTI crude oil futures using artificial intelligence model with chaos. *Fuel*, 316:122523.

[143] Zhang, Y. (2016). The image encryption algorithm with plaintext-related shuffling. *IETE Technical Review*, 33(3):310–322.

[144] Zhang, Y., Wang, S., and Ji, G. (2015). A comprehensive survey on particle swarm optimization algorithm and its applications. *Mathematical Problems in Engineering*, 2015:1–38.

[145] Zheng, H., Yu, S., and Xu, X. (2014). A systematic methodology for multi-images encryption and decryption based on single chaotic system and FPGA embedded implementation. *Mathematical Problems in Engineering*, 2014:1–15.

[146] Zhou, S. and Wang, X. (2020). Simple estimation method for the second-largest Lyapunov exponent of chaotic differential equations. *Chaos, Solitons & Fractals*, 139:109981.

[147] Zhou, S., Wang, X., Wang, Z., and Zhang, C. (2019). A novel method based on the pseudo-orbits to calculate the largest Lyapunov exponent from chaotic equations. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(3):033125.

[148] Özkaynak, F. (2018). Brief review on application of nonlinear dynamics in image encryption. *Nonlinear Dynamics*, 92(2):305–313.

[149] Öztürk, I. and Kılıç, R. (2018). Digitally generating true orbits of binary shift chaotic maps and their conjugates. *Communications in Nonlinear Science and Numerical Simulation*, 62:395–408.

Appendix A

Algorithms

The algorithms and simulations presented in this thesis were developed using several tools. These include MATLAB R2018a by MathWorks, Vivado 2019.1 by Xilinx - AMD, ModelSim by Mentor Graphics - Siemens, and LTspice XVII by Analog Devices. If you are interested, you can access the algorithms used in this thesis at `https://github.com/gnlucas/Thesis.git`.

Appendix B

SPICE - Circuits and Graphics

In this appendix, all the benchmark circuits used in Chapter 4 are shown. Note that Figures B.1, B.3, B.5, B.7, B.9, and B.11 indicate the chosen equivalence relation, which was applied to linear components (resistors, inductors, and capacitors) as it is more straightforward. Each equivalence relation is highlighted with a red box.

Furthermore, Figures B.2, B.4, B.6, B.8, B.10, and B.12 illustrate the divergence of pseudo-orbits in equivalent chaotic circuits and the slope of the logarithm of such a divergence, which characterizes the LLE. LTspice XVII was used here since it provides all the necessary tools to estimate the Lyapunov exponent.

The benchmark circuits presented demonstrate the effectiveness of the proposed method in estimating the LLE of chaotic circuits. It is believed that this appendix will provide researchers with valuable insights into the practical implementation of the approach, as also it will encourage further exploration of the topic.

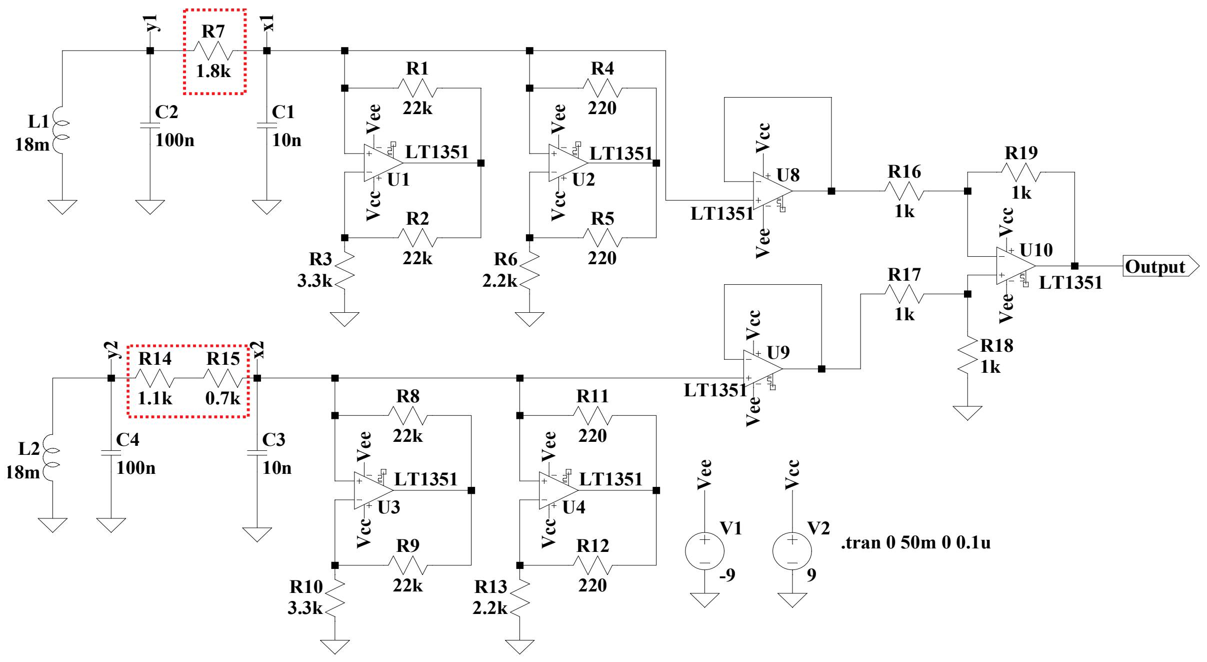
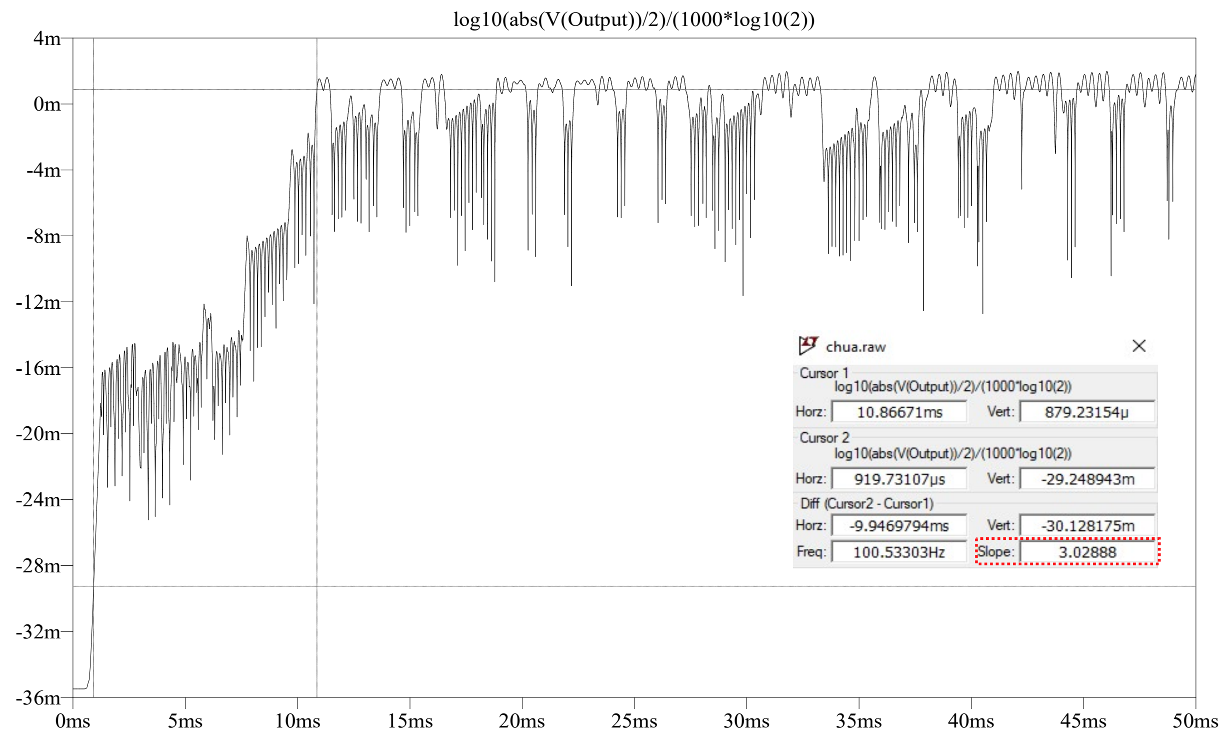


Figure B.1: Schematic to determine the largest Lyapunov exponent for Chua's circuit [19]. Both circuits are equivalent since $R_7 = R_{14} + R_{15} = 1.8\ k\Omega$. `Schematic: Largest Lyapunov Exponent / chua.asc`.



Cursor 1
log10(abs(V(Output))/2)/(1000*log10(2))
Horz: 10.86671ms Vert: 879.23154µ
Cursor 2
log10(abs(V(Output))/2)/(1000*log10(2))
Horz: 919.73107µs Vert: -29.248943m
Diff (Cursor2 - Cursor1)
Horz: -9.9469794ms Vert: -30.128175m
Freq: 100.53303Hz Slope: 3.02888

Figure B.2: Calculation of the lower-bound error for Chua's circuit, where the slope indicates the largest Lyapunov exponent (LLE $= 3.02888\ bits/ms$). `Schematic: Largest Lyapunov Exponent / chua.asc`.

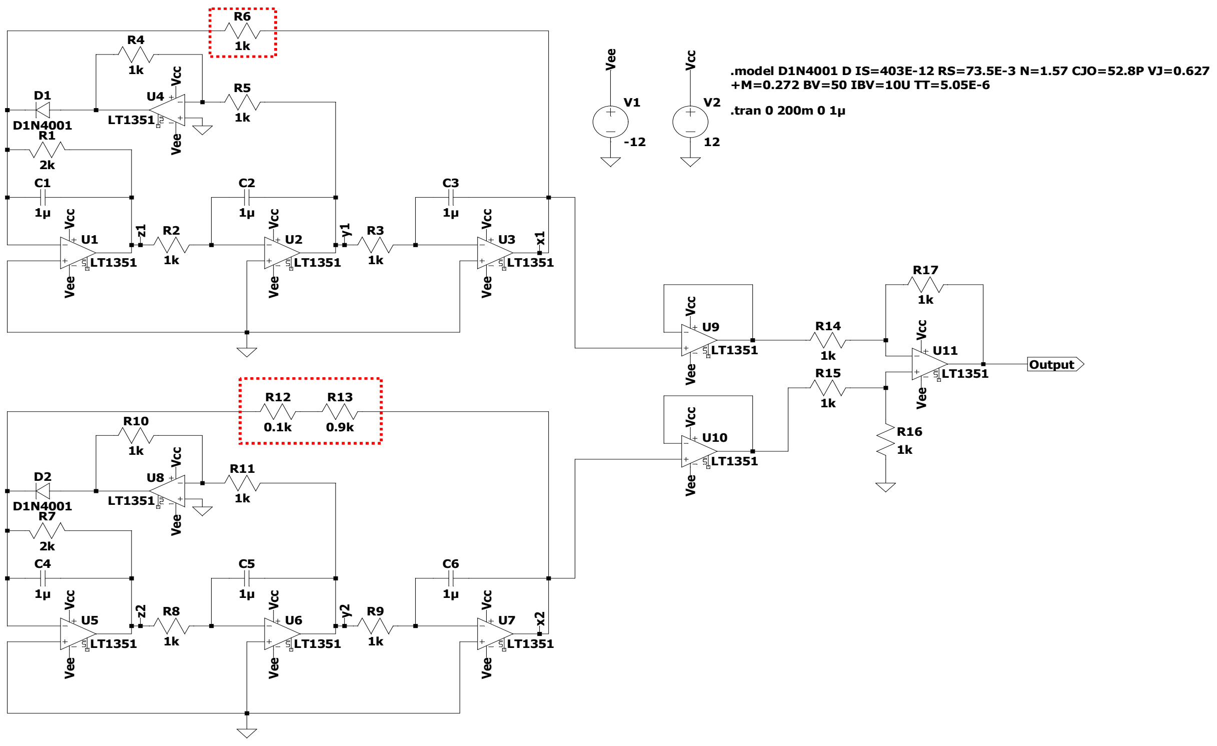
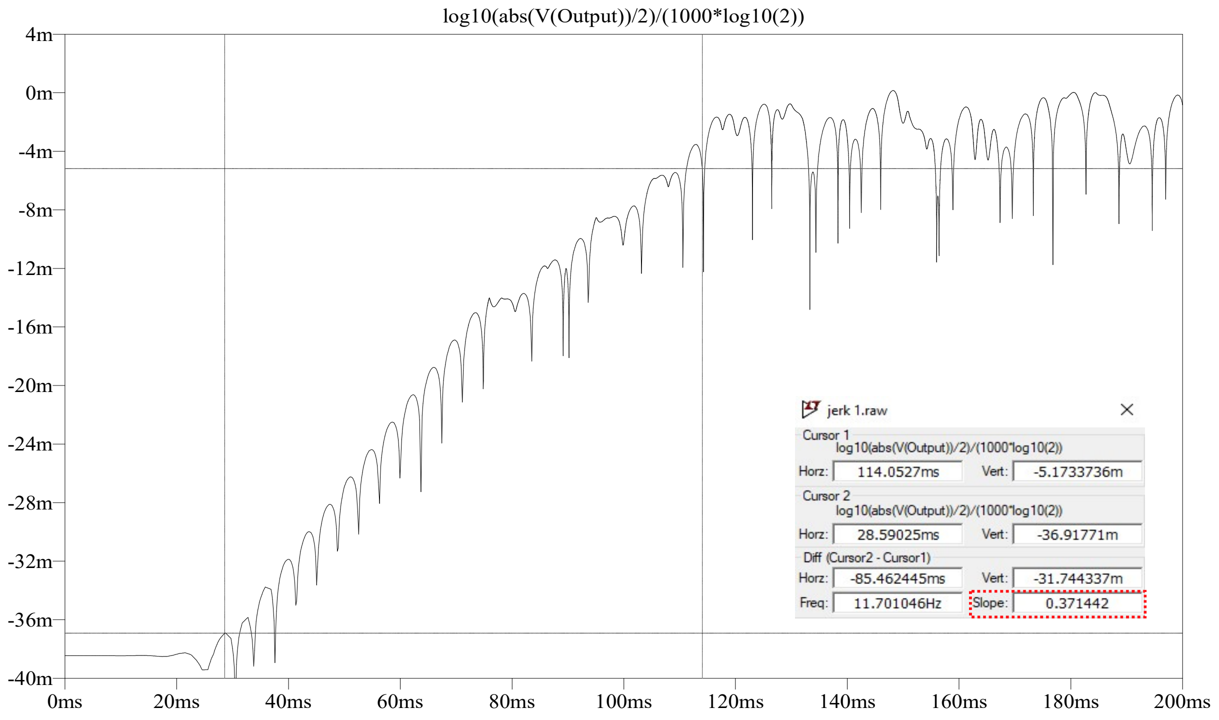

```
.model D1N4001 D IS=403E-12 RS=73.5E-3 N=1.57 CJO=52.8P VJ=0.627
+M=0.272 BV=50 IBV=10U TT=5.05E-6
.tran 0 200m 0 1µ
```

Figure B.3: Schematic to determine the largest Lyapunov exponent for Jerk 1 [122]. Both circuits are equivalent since $R_6 = R_{12} + R_{13} = 1\ k\Omega$. Schematic: Largest Lyapunov Exponent / jerk 1.asc.


jerk 1.raw

Cursor 1 log10(abs(V(Output))/2)/(1000*log10(2))
Horz: 114.0527ms Vert: -5.1733736m

Cursor 2 log10(abs(V(Output))/2)/(1000*log10(2))
Horz: 28.59025ms Vert: -36.91771m

Diff (Cursor2 - Cursor1)
Horz: -85.462445ms Vert: -31.744337m
Freq: 11.701046Hz Slope: 0.371442

Figure B.4: Calculation of the lower-bound error for Jerk 1, where the slope indicates the largest Lyapunov exponent (LLE $= 0.371442\ bits/ms$). Schematic: Largest Lyapunov Exponent / jerk 1.asc.

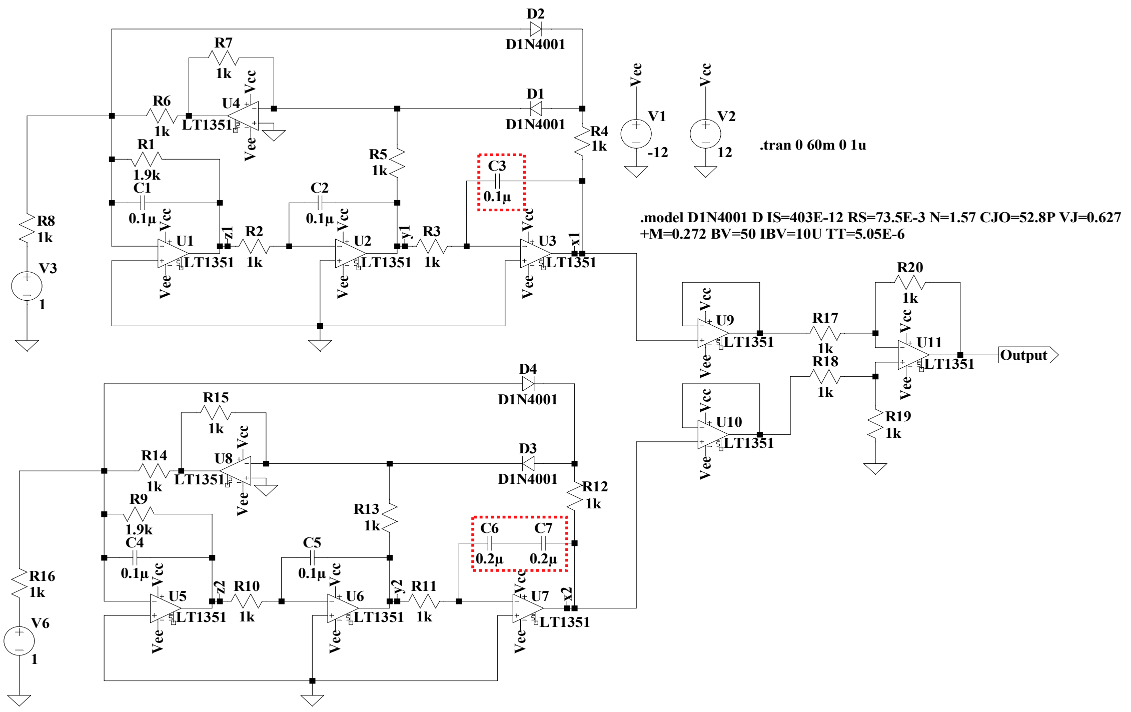
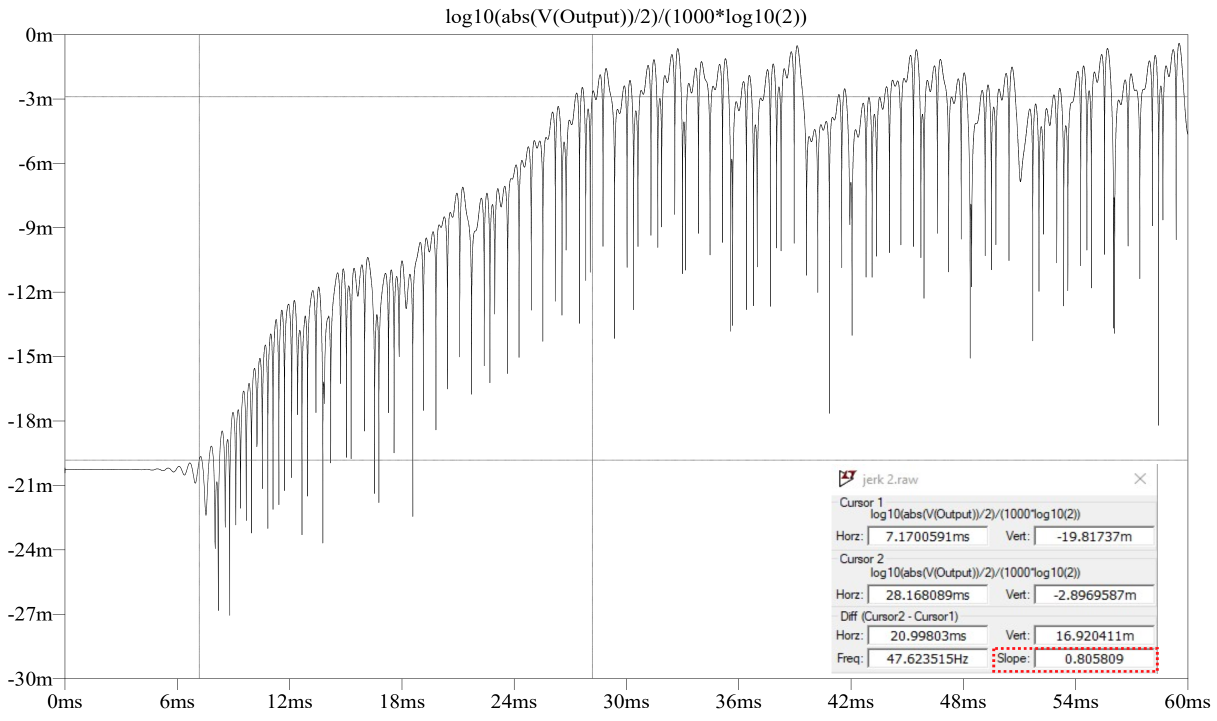


Figure B.5: Schematic to determine the largest Lyapunov exponent for Jerk 2 [62]. Both circuits are equivalent since $\frac{1}{C_3} = \frac{1}{C_6} + \frac{1}{C_7} = 0.1\ \mu F$. `Schematic: Largest Lyapunov Exponent / jerk 2.asc.`



```
Cursor 1
    log10(abs(V(Output))/2)/(1000*log10(2))
Horz: 7.1700591ms      Vert: -19.81737m
Cursor 2
    log10(abs(V(Output))/2)/(1000*log10(2))
Horz: 28.168089ms      Vert: -2.8969587m
Diff (Cursor2 - Cursor1)
Horz: 20.99803ms       Vert: 16.920411m
Freq: 47.623515Hz      Slope: 0.805809
```

Figure B.6: Calculation of the lower-bound error for Jerk 2, where the slope indicates the largest Lyapunov exponent (LLE = 0.805809 $bits/ms$). `Schematic: Largest Lyapunov Exponent / jerk 2.asc.`

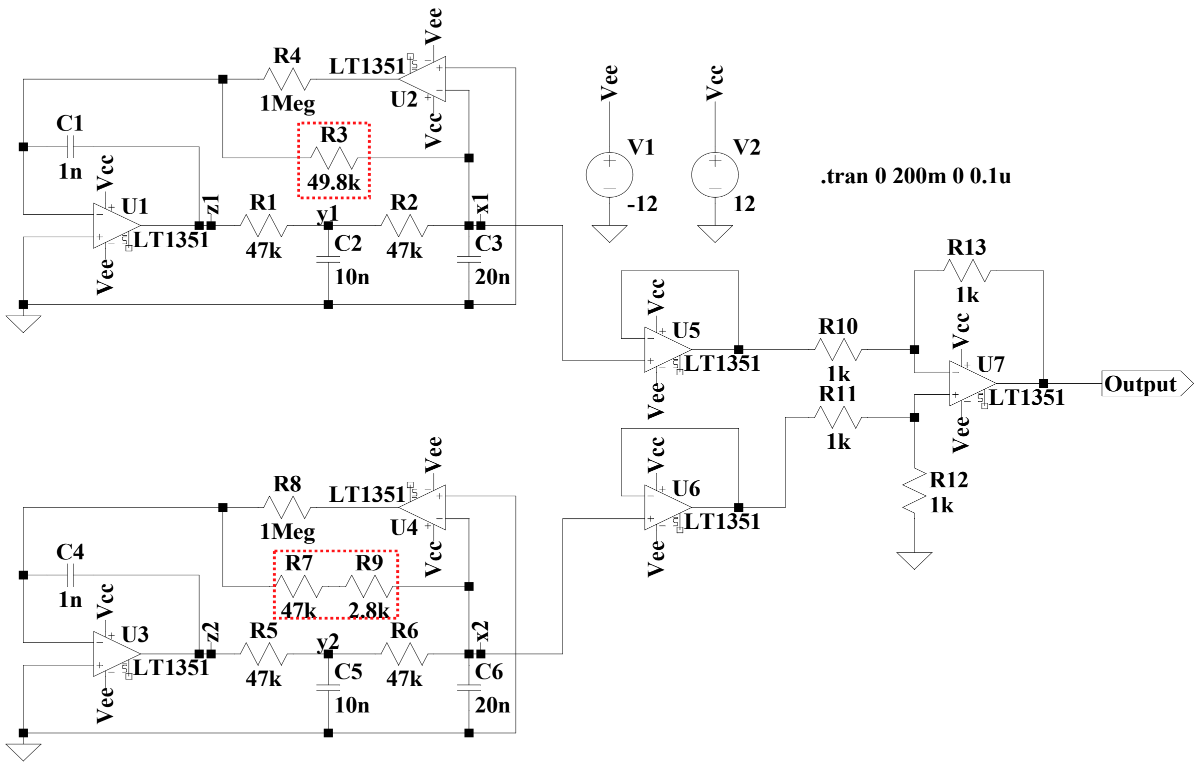
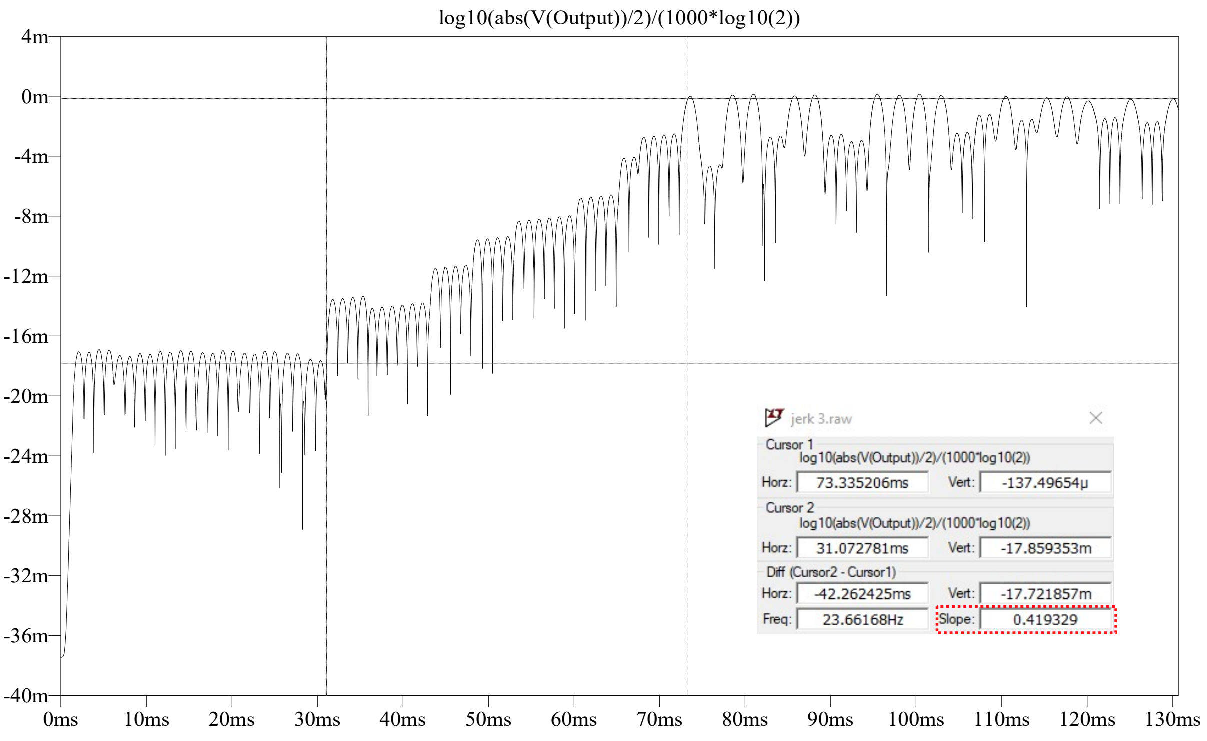
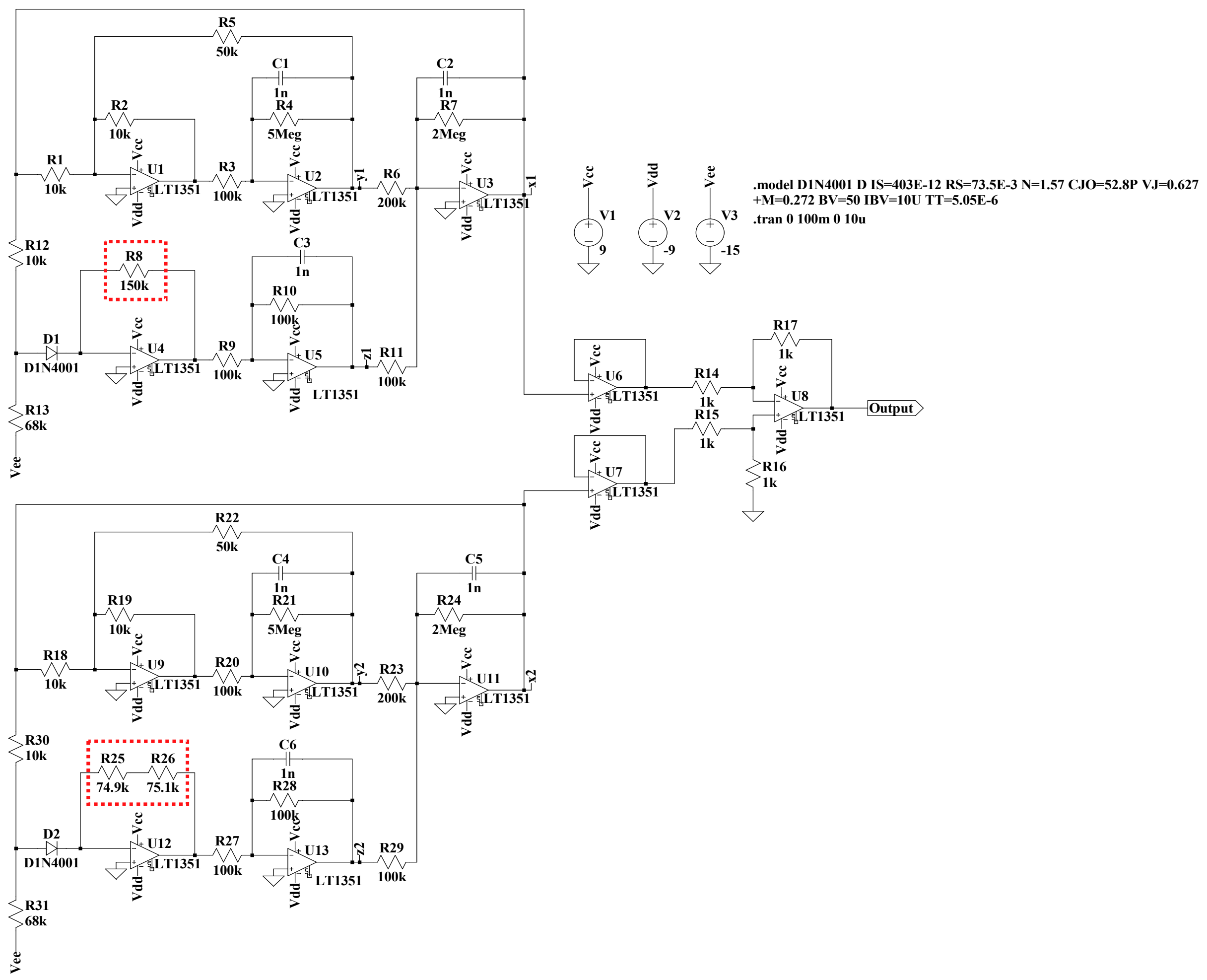


Figure B.7: Schematic to determine the largest Lyapunov exponent for Jerk 3 [101]. Both circuits are equivalent since $R_3 = R_7 + R_9 = 49.8\ k\Omega$. `Schematic: Largest Lyapunov Exponent / jerk 3.asc`.



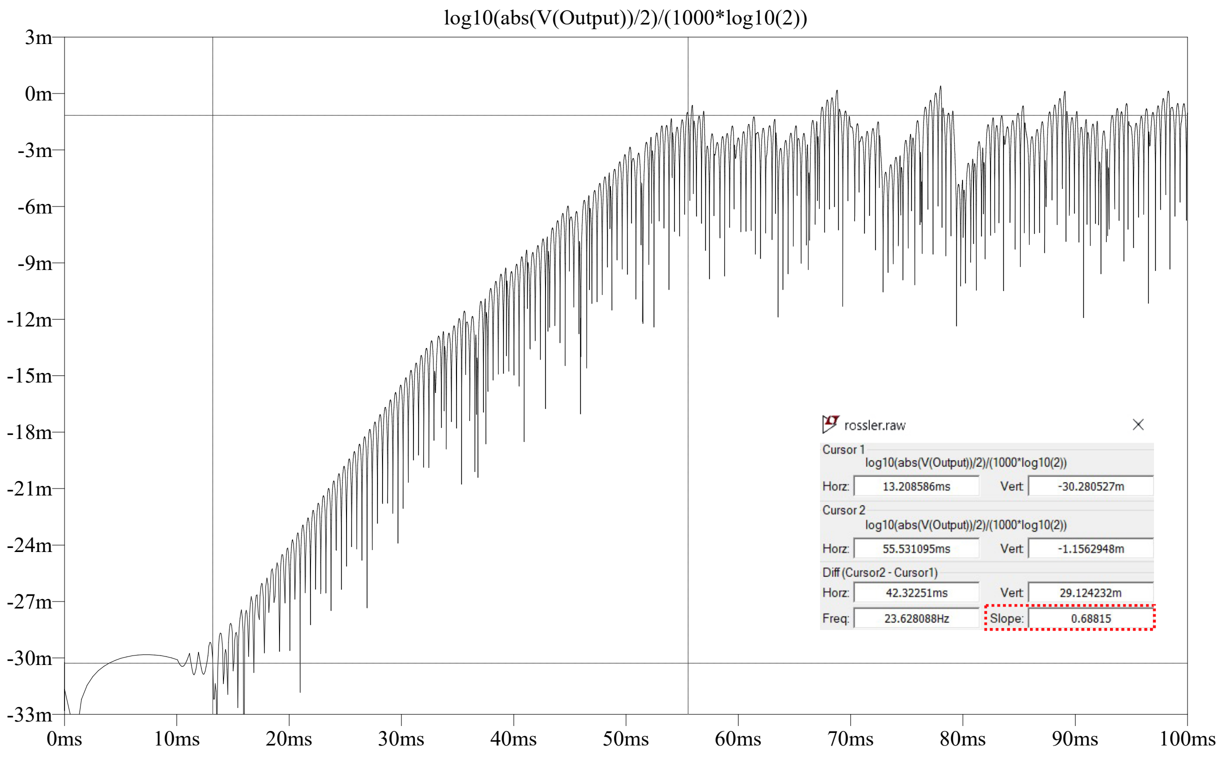
```
Cursor 1
    log10(abs(V(Output))/2)/(1000*log10(2))
Horz: 73.335206ms        Vert: -137.49654µ
Cursor 2
    log10(abs(V(Output))/2)/(1000*log10(2))
Horz: 31.072781ms        Vert: -17.859353m
Diff (Cursor2 - Cursor1)
Horz: -42.262425ms       Vert: -17.721857m
Freq: 23.66168Hz         Slope: 0.419329
```

Figure B.8: Calculation of the lower-bound error for Jerk 3, where the slope indicates the largest Lyapunov exponent (LLE $= 0.419329\ bits/ms$). `Schematic: Largest Lyapunov Exponent / jerk 3.asc`.



```
.model D1N4001 D IS=403E-12 RS=73.5E-3 N=1.57 CJO=52.8P VJ=0.627
+M=0.272 BV=50 IBV=10U TT=5.05E-6
.tran 0 100m 0 10u
```

Figure B.9: Schematic to determine the largest Lyapunov exponent for Rössler circuit [46]. Both circuits are equivalent since $R_8 = R_{25} + R_{26} = 150\ k\Omega$. `Schematic: Largest Lyapunov Exponent / rossler.asc`.



rossler.raw

Cursor 1
log10(abs(V(Output))/2)/(1000*log10(2))
Horz: 13.208586ms Vert: -30.280527m

Cursor 2
log10(abs(V(Output))/2)/(1000*log10(2))
Horz: 55.531095ms Vert: -1.1562948m

Diff (Cursor2 - Cursor1)
Horz: 42.32251ms Vert: 29.124232m
Freq: 23.628088Hz Slope: 0.68815

Figure B.10: Calculation of the lower-bound error for Rössler circuit, where the slope indicates the largest Lyapunov exponent (LLE = 0.68815 $bits/ms$). `Schematic: Largest Lyapunov Exponent / rossler.asc.`

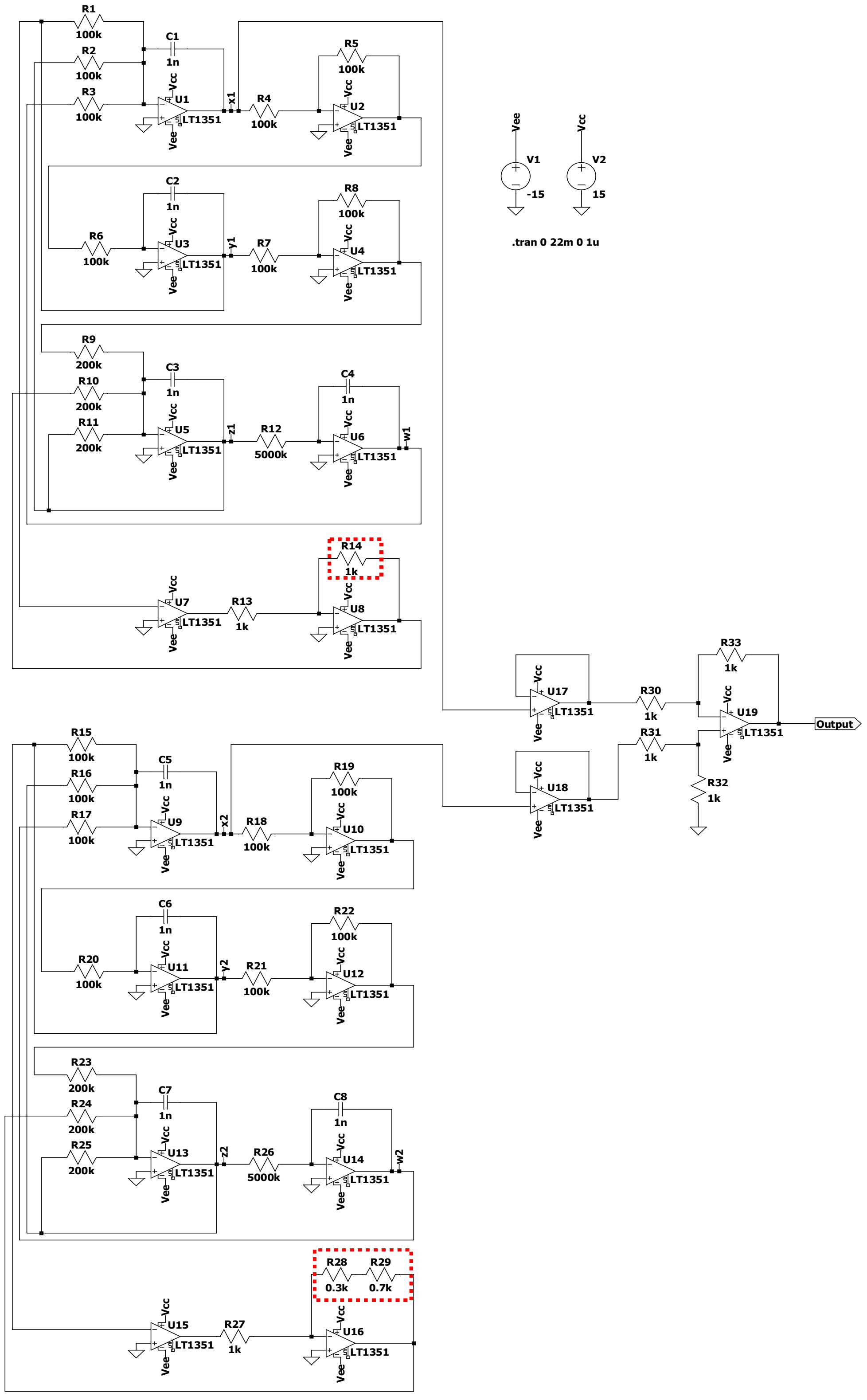
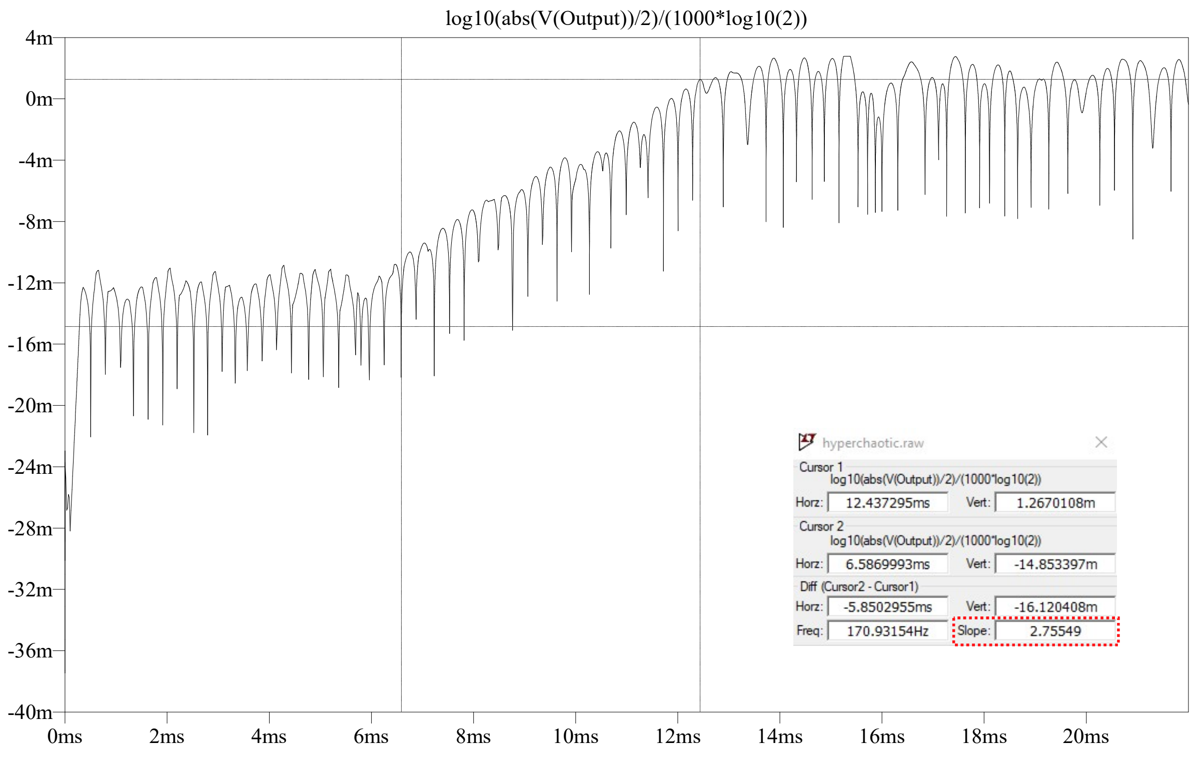


Figure B.11: Schematic to determine the largest Lyapunov exponent for a hyperchaotic circuit [118]. Both circuits are equivalent since $R_{14} = R_{28} + R_{29} = 1\ k\Omega$. `Schematic: Largest Lyapunov Exponent / hyperchaotic.asc`.



Cursor 1
log10(abs(V(Output))/2)/(1000*log10(2))
Horz: 12.437295ms Vert: 1.2670108m

Cursor 2
log10(abs(V(Output))/2)/(1000*log10(2))
Horz: 6.5869993ms Vert: -14.853397m

Diff (Cursor2 - Cursor1)
Horz: -5.8502955ms Vert: -16.120408m
Freq: 170.93154Hz Slope: 2.75549

Figure B.12: Calculation of the lower-bound error for a hyperchaotic circuit, where the slope indicates the largest Lyapunov exponent (LLE = 2.75549 $bits/ms$). `Schematic: Largest Lyapunov Exponent / hyperchaotic.asc.`