A New PRNG Hardware Architecture Based on an Exponential Chaotic Map

Matheus B. R. Cardoso[†], Samuel S. da Silva[†], Lucas G. Nardo[‡], Roberto M. Passos Jr.[‡],
Erivelton G. Nepomuceno[§], IEEE Senior Member, and Janier Arias-Garcia[†‡*]
[†]Department of Electronic Engineering and [‡]Graduate Program in Electrical Engineering,
Federal University of Minas Gerais (UFMG), Belo Horizonte, MG, 31270-901 Brazil.
[§]Control and Modeling Group (GCOM), Department of Electrical Engineering,
Federal University of São João del-Rei (UFSJ), São João del-Rei, MG, 36307-352, Brazil.

2021 IEEE International Symposium on Circuits and Systems (ISCAS) | 978-1-7281-9201-7/20/$31.00 ©2021 IEEE | DOI: 10.1109/ISCAS51556.2021.9401653

***Abstract*—Recent works have been claiming efficient hardware architectures, showing a considerable endeavor to implement chaotic maps in the digital domain. However, there is a critical issue with the chaotic degradation in the digital environment due to its finite numeric precision, that it is still an unsettled topic in the research community. Additionally, less attention has been given to synthesize a methodological approach to how to calculate the exponential function in hardware. In this paper, two novel hardware designs to represent the exponential chaotic map have been suggested. We have employed a perturbation method to avoid the chaotic degradation. 64-bit fixed-point and 32-bit floating-point formats were investigated. Moreover, an approximation of Euler's number by a finite series and the Horner's method have been undertaken to further minimize the proposed hardware. Results show that both proposed hardware architectures consume a fewer number of components. The designed systems present a positive Lyapunov exponent, which suggests a chaotic behavior. Ultimately, the NIST SP 800-22 test, the histogram, and the autocorrelation function show that the new hardware architectures present pseudo-random properties.**

***Index Terms*—Exponential Chaotic Map, Chaotic Systems, Chaos, FPGA, PRNG, Computer Arithmetic.**

I. Introduction

Chaotic systems have been largely investigated since the revolutionary Poincaré's studies about the chaotic nature in the movement of celestial bodies [1]. Despite the diverse models of chaotic systems, a dynamic system must show particular characteristics to be considered chaotic, such as positive Lyapunov exponent, transitivity, sensitive dependence on initial conditions, and periodic points of the chaotic function must be dense in some metric space [2]. These characteristics make chaotic systems attractive in many applications, such as optimization problems [3], network security and communications systems [4], and cryptography [5].

Considering the special features present in reconfigurable hardware, as flexibility, fast processing, high parallelism and dedicated hardware [6], many FPGA hardware architectures have been designed to reproduce such chaotic systems [7], [8]. However several other issues have been reported in said systems, such as hardware efficiency [9], as well as the reliability of simulating chaotic systems in the digital domain. Reports show failed attempts to reproduce chaos due to degradation and spurious behavior caused by the way numbers are represented digitally [10], [11].

Recently, a new paper [12] has shown a new trend about hardware design for chaotic systems. Without disturbing the chaotic dynamics and to use as few components to compound the hardware as possible, the proposed work reduces the number of mathematical operations by substituting complex blocks of math operations for Shift Register and simple logic gates.

Here, we have explored an FPGA implementation of a novel hardware architecture of the exponential chaotic map. Based on two approaches that use different numeric representation, the 64-bit fixed-point and 32-bit floating-point, the proposed concept was designed by approximating Euler's number by a finite series. The obtained polynomial representation was simplified by means of Horner's method. To overcome the digital chaotic degradation, a perturbation process in the bits of the orbit is presented. Moreover, to the best of our knowledge, no paper adopts such an approach and minimizes the exponential function in hardware.

Based on the previous discussion, we can summarize the contributions of this paper to:

- A methodological design process for an implementation of a new hardware architecture of the exponential chaotic map, paying attention to its hardware resources and performance.
- A discussion about the different numerical representations, observing if both representations are able to reproduce chaos, and maintain a competitive resource usage when it comes to hardware implementations on FPGAs.

The remainder of this article is described as follows: Section II presents preliminary concepts to understand the rest of this paper. The hardware design implementation, as well as the results are described in Section III and Section IV, respectively. Finally, Section V presents final concluding remarks.

II. Preliminary concepts

A. Exponential Map

The logistic map [13] is represented by $x_{i+1} = \lambda x_i(1 - x_i)$, where $x_i \in [0, 1)$. It is a simple equation with chaotic

Erivelton G. Nepomuceno, Ph.D., was supported by Brazilian Research Agencies: CNPq/INERGE (Grant No. 465704/2014-0), CNPq (Grant No. 425509/2018-4) and FAPEMIG (Grant No. APQ-00870-17). Lucas G. Nardo, M.Sc., was supported by FAPEMIG.

* Corresponding author: janier-arias@ufmg.br

978-1-7281-9201-7/21/$31.00 ©2021 IEEE

complexity. Consequently, this map is vastly used in the study of chaos. In this paper, a variation of the logistic map, known as the exponential map is used, which is defined as: $x_{i+1} = \lambda x_i e^{-x_i}$.

B. Horner's Method

Horner's method is an algorithm, proposed by William George Horner, to evaluate a polynomial in a specific value x_0 by writing the polynomial at the form $p(x) = a_0 + x(a_1 + x(a_2 + ... + x(a_{i-1} + xa_i)...))$ and recursively applying $x = x_0$ until the final sum produced by $a_0 + x_i(...)$ is computed. Horner's original approach solves a polynomial evaluation using a linear number of multiplications and additions [14].

C. Maclaurin Series and Horner's evaluation of the result

The Maclaurin series of a certain function can be described as an evaluation of the Taylor series represented by $f(x) = \sum_{k=0}^{\infty} f^{(k)}(a)\frac{(x-a)^k}{k!}$, when $a = 0$. Hence, the exponential function used in the exponential map could be approximated by the polynomial $e^{-x} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!} - \frac{x^5}{5!} + \cdots \forall x \in R$. Thus, it is possible to approximate the exponential function using only sums and multiplications.

For example, writing the polynomial approximation of e_n^{-x} for the first $n = 4$ terms, one would find: $e_{n=4}^{-x} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!}$. By means of Horner's rule, it was possible to rewrite this function as follows:

$$e_{n=4}^{-x} = x\left(x\left(x\left(\frac{x}{4!} - \frac{1}{3!}\right) + \frac{1}{2!}\right) - 1\right) + 1, \qquad (1)$$

We can evaluate (1) for $x = x_0$ using a recurrence approach, such as:

$$\begin{aligned} b_1 &= a_4 x_0 + a_3,\ b_2 = b_1 x_0 + a_2, \\ b_3 &= b_2 x_0 + a_1,\ b_4 = b_3 x_0 + a_0. \end{aligned} \qquad (2)$$

Here, $a_4 = \frac{1}{4!}$, $a_3 = \frac{-1}{3!}$, $a_2 = \frac{1}{2!}$, $a_1 = -1$ and $a_0 = 1$. Each new b term calculated is used to compute its next value. Thus, when b_4 is calculated, the evaluation of e_4^{-x} is completed.

When extending this result for $e_n^{-x_0}$, with $n \in \mathbb{Z}_+^*$. It is possible to write the following recurrence equation:

$$b_i = b_{i-1}x_0 + a_{n-i}, \qquad (3)$$

where $n \in \mathbb{Z}_+^*$ and $b_1 = a_n x_0 + a_{n-1}$. Thus, for $n = 4$, the value of b_1 would be $b_1 = a_4 x_0 + a_3$.

In (3), it is important to state that all values of a are given by the Maclaurin series of $e_n^{-x_0}$, truncated in the n^{th} position. Thus, they are previously known and only the b values must be calculated in order to evaluate this function.

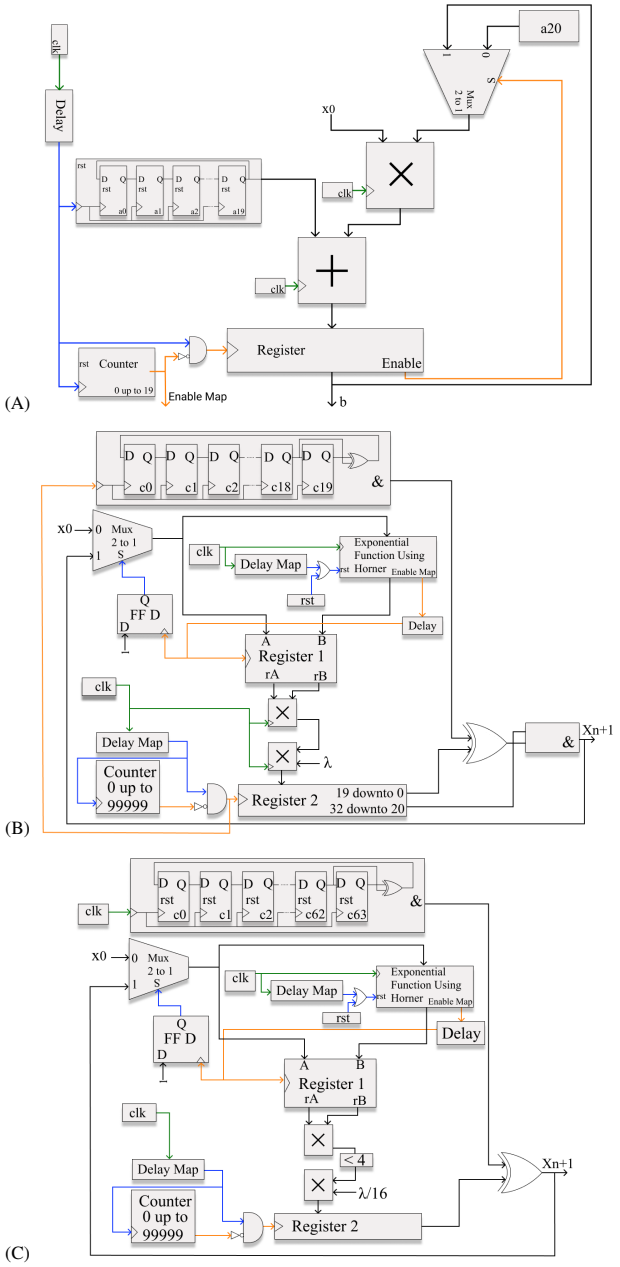

Fig. 1. Proposed hardware architectures for the following cases: (A) Exponential calculation, x_0 is the block input, while b is its output. (B) The complete map design in floating-point arithmetic. (C) The complete map design in fixed-point arithmetic. For all three figures, black lines represent the flow of floating-point numbers or fixed-point numbers, green lines represent one-bit clock signals, blue lines represent delayed one-bit signals and orange lines represent one-bit control signals. Furthermore, the symbol "&" represents concatenation operation, the same notation used in the VHDL hardware description language.

III. Hardware Implementation

The calculation of the exponential function is based on (3), derived in section II. We have reached a desired outcome for NIST test suite and Lyapunov exponent using $n = 20$. After 20 iterations, $b = b_{20}$ and the exponential approximation by Horner's method is completed (see Fig. 1-A). To better

understand the process to get to b_{20}, the control signals are analyzed first, then the delay signals, and finally, the clock signals. In the first clock cycle, the Select signal (S) will be '0', meaning that a_{20} will be selected in the Multiplexer. In this way, it will calculate b_1, then the control signal called "Enable" coming from the Register to the Multiplexer will shift the S value from '0' to '1', meaning that our recurrence equation has started.

Following this, the delay block has two functions. First, it guarantees that the clock signal will be delayed when it gets to the Shift Register, which was used to store the pre-computed values of a. In this way, the delay block synchronizes the sum operation that happens right after the multiplication. Its second function is to delay, in the same proportion, the clock signal that reaches the Counter. After 20 clock cycles, the calculation will be stopped, meaning that the evaluation is over and the Register contains the value of $b = b_{20}$. The clock signals, represented by the green lines, were set in the simulation to meet the specifications of the low-cost FPGA *Artix 7* family.

Besides the procedure for calculating the exponential function, a perturbation method is provided to keep chaotic behavior longer. It is necessary due to the finite numerical representation of the digital domain, introduced by both numerical representations used. The chosen method is similar to other perturbation approaches found in the literature, especially in the article [15]. The perturbation scheme used included the addition of a bit-to-bit XOR operation between the 20 least significant bits of the map's mantissa and the LSFR register for the floating-point approach. For the fixed-point approach, all bits were disturbed.

The exponential chaotic map using floating-point is shown in Fig. 1-B, while Fig. 1-C shows the approach for fixed-point implementation. It is important to mention that for the fixed-point approach, a constant value of $\lambda = 17.4$ was chosen and, in order to multiply a number for 17.4 on the proposed hardware architecture, the authors combined a shift left logical operation of 4 positions (multiplying by 16). To compensate, the second multiplier block had one of its input values fixed at $17.4/16 = 1.0875$, as Fig. 1-C shows.

Once the exponential function is done, calculating the result for the current iteration of the map, the "Enable Map" signal, taken from the Counter shown in Fig. 1-A, will enable Register 1 to record the results of the calculation and, also, will set the output of a Flip Flop D (represented as "FF D") to '1'. Therefore, after the first calculation of the exponential approximation, the "FF D" will change the S value in the Multiplexer to always choose the previous iteration of the map, respecting the map's recurrence equation: $x_{n+1} = \lambda x_n e^{-x_n}$.

Moreover, in Fig. 1-B and Fig. 1-C, two delay blocks are presented, which will work to maintain the operations in synchronization. The one called "Delay" is calculated based on how many clock cycles the exponential will take to give the correct value, and the other called "Delay Map" will keep up with how many clock cycles the whole approach takes. After calculating 100000 values of x_{n+1}, the counter will stop Register 2, and no more values of x_{n+1} will be computed. For this reason, our approach will produce the 100000 first values of the sequence.

Finally, for the floating-point approach, all the bits from the internal LSFR flip flops are concatenated. Then, the XOR operation is performed between these 20 concatenated bits and the last 20 bits of the chaotic map's answer. After this, another concatenation must occur to give 32-bit single-precision answers. Meanwhile, in the fixed-point, it is possible to disturb all bits.

The main difference between fixed and floating-point approaches is that the fixed-point is going to use 63 bits for the fractional part, while using only one bit for the integer part. Therefore, the disturbance method will be applied in all bits without divergent behavior on this approach. While the floating-point generates values from an interval of [0,7), the fixed-point generates values from [0,2), since only one bit was used for the integer part on the latter.

IV. Results

Adopting the low-cost FPGA *Artix 7 XC7A100T-1CSG324C* and the hardware description language VHDL with the Xilinx Vivado 2017.4 tool, the two schemes, shown in Fig. 1-B and in Fig. 1-C, were implemented. Table I exhibits the resource consumption for both implementations. It is possible to observe that both architectures consume fewer components than the ones available on the FPGA, as well as present low power consumption.

To prove that both hardware architectures present chaotic properties, the Lyapunov exponent, based on Wolf's method [16], was calculated. Applying $\lambda = 17.4$ and $x_0 = 0.31$, two sequences of the exponent were calculated using all bits of the orbits for both numerical representations. Note that both sequences present a positive exponent, indicating chaotic series (see Table I).

TABLE I
RESOURCE CONSUMPTION FOR BOTH HARDWARE ARCHITECTURES. NOTE THAT ALTHOUGH FLOATING-POINT ARITHMETIC IS MORE COMPLEX THAN FIXED-POINT ARITHMETIC, ITS HARDWARE PRESENTED A LOWER CONSUMPTION OF COMPONENTS AND A MUCH HIGHER FREQUENCY OF OPERATION.

Resources	**32-bit floating-point**	**64-bit fixed-point**
LUT	915 (1.44%)	1466 (2.31%)
LUTRAM	23 (0.12%)	66 (0.35%)
FF	1101 (0.87%)	1256 (0.99%)
DSP	6 (2.50%)	48 (20%)
Power	96 mW	90 mW
Frequency	121.95MHz	43.47MHz
Lyapunov Exponent	0.7576	2.2733

Fig. 2 shows three sequences of the Poincaré map. The sequences in (A) and (C) use all the bits for the floating-point and for the fixed-point approaches, respectively. Furthermore, using floating-point arithmetic, the sequence in (B) considered only the 20 least significant bits of the map. The orbit using all the 32 bits of the floating-point map preserves the shape of the Poincaré map, which is commonly found in the literature. However, due to the perturbation method in all of the bits

in the other two sequences, the shape of the Poincaré map is not rebuilt. Nevertheless, the system still presents chaotic behavior.

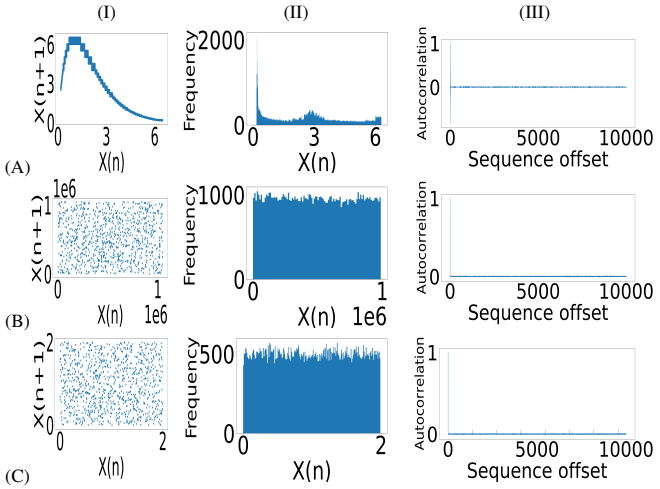

Fig. 2. Results of the proposed hardware architectures. Line (A) shows the achieved results for the 32-bit floating-point. Line (B) exhibits the results considering only the 20 least significant bits for the 32-bit floating-point approach. Line (C) shows the results considering the fixed-point approach. The Poincaré map, the histogram, and the autocorrelation function, for each sequence, are shown in columns (I), (II), and (III), respectively. In all tests, we have considered $\lambda = 17.4$ and $x_0 = 0.31$.

All the three previous sequences were tested in the NIST SP 800-22 test suite. The SP 800-22 consists of a collection of 15 tests to evaluate the randomness of both true-random numbers and pseudo-random numbers [17], [18]. In each test, a $P - value$ is provided and if $P - value \geq \alpha$, where α means significance level, the sequence has passed the test. For $\alpha = 0.01$ and sequences with bit stream length of 1000000 bits, the results obtained from the proposed hardware architectures are shown in Table II. It is noticeable from the results in Table II, that the proposed chaotic map passed in all tests in (B) and in (C), indicating that these sequences have a good pseudo-randomness. It is clear that in (A) it failed most of the tests where it got $P - values$ equal to zero. This is an interesting result, as it suggests some advantage of fixed-point representation over floating-point. In fact, the sequence generated in (A) does not have a considerable variation of values on the exponent bitstring.

Uniform histograms and autocorrelation functions similar to a delta function are expected for suitable pseudo-random sequences. Fig. 2 also shows the histogram and the autocorrelation function. For the exponential map using floating-point and considering all the 32 bits, the histogram is non-uniform. However, the histograms of the hardware-based proposed chaotic map using floating-point, considering only the least significant 20 bits, and the hardware scheme using fixed-point present histogram-shapes close to the uniform distribution. Also, note that all the sequences show no apparent correlation between samples.

Finally, it is possible to compare this work with other state-of-the-art implementations available in the literature.

TABLE II

$P - value$ RESULTS AFTER RUNNING THE SP 800-22 TEST SUITE: (A) FLOATING-POINT CONSIDERING ALL 32 BITS; (B) FLOATING-POINT CONSIDERING THE 20 LEAST SIGNIFICANT BITS; (C) 64 BITS FIXED-POINT. IN ALL TESTS, WE HAVE USED $\lambda = 17.4$, $x_0 = 0.31$ AND A SEQUENCE WITH BIT STREAM LENGTH OF 1000000 BITS.

Test	(A)	(B)	(C)
Frequency	0.000000	0.228590	0.888660
Block Frequency ($m = 128$)	0.000000	0.518118	0.256523
Cusum-Forward	0.000000	0.372855	0.972191
Cusum-Reverse	0.000000	0.334489	0.891349
Runs	0.000000	0.298084	0.752017
Long Runs of Ones	0.048229	0.520249	0.707144
Rank	0.000000	0.341610	0.419146
Spectral DFT	0.000000	0.992678	0.388356
Non-overlapping Templates ($m = 9$, $B = 000000001$)	0.000000	0.116838	0.497266
Overlapping Templates ($m = 9$)	0.000000	0.166615	0.419389
Universal	0.000000	0.111677	0.604572
Approximate Entropy ($m = 10$)	0.000000	0.937093	0.394210
Random Excursions ($x = +1$)	0.000000	0.475778	0.260944
Random Excursions Variant ($x = -1$)	0.000000	0.437545	0.835268
Linear Complexity ($M = 500$)	0.639876	0.308306	0.955737
Serial ($m = 16$, $\nabla\Psi_m^2$)	0.473209	0.234743	0.734313

Unfortunately, no fair comparisons can be made, because to the best of our knowledge, this paper is the first to present an implementation of the exponential chaotic map using Horner's rule and LSFR registers. However, it is possible to compare it (in terms of frequency and logic usage) to other chaotic maps in the literature. Table III compares this paper with the literature. It is possible to realize that our approach is competitive than the logistic and Chebyshev maps. Nonetheless, the maps presented in Table III may have very different dynamics, and because some maps have simpler functions to compute than others, it is expected that their resource consumption is lower.

TABLE III

COMPARISON BETWEEN DISTINCT HARDWARE APPROACHES.

Map	Reference	Logic Elements / LUT	Max. frequency (MHz)
Exponential 32-bits	This Paper	915	121.95
Exponential 64-bits	This Paper	1466	43.47
Bernoulli (LSFR)	[15]	199	404.37
Tent (LSFR)	[15]	219	206.10
Baker's (LSFR)	[15]	199	500.00
Chebyshev (LSFRs)	[15]	2525	43.47
TRNG	[15]	311	254.97
Logistic	[15]	2634	42.01
CSS-2	[19]	549	–

V. CONCLUSION

In this paper, we have described two hardware architectures based on an exponential chaotic map. Applying the approximation of Euler's number by a Maclaurin series and using Horner's method, we have efficiently implemented the exponential chaotic map on a low-cost FPGA. Our approach took 20 clock cycles to evaluate the exponential function. Results also show that the proposed designs consume few logical resources, demand low power consumption, show a high frequency of operation, and display chaotic dynamics with positive Lyapunov exponents. Finally, both the chaotic map with floating-point or fixed-point present significant pseudorandom properties, with uniform histogram, autocorrelation function similar to white Gaussian noise and pass all proposed SP 800-22 NIST tests.

REFERENCES

[1] H. Poincaré, "Sur le problème des trois corps et les équations de la dynamique," *Acta mathematica*, vol. 13, no. 1, pp. A3–A270, 1890.

[2] J. Banks, J. Brooks, G. Cairns, G. Davis, and P. Stacey, "On Devaney's Definition of Chaos," *The American Mathematical Monthly*, vol. 99, no. 4, p. 332, apr 1992.

[3] J.-H. Chen, H.-T. Yau, and J.-H. Lu, "Implementation of FPGA-Based Charge Control for a Self-Sufficient Solar Tracking Power Supply System," *Applied Sciences*, vol. 6, no. 2, p. 41, feb 2016.

[4] S. Sadoudi and C. Tanougast, "Robust hyperchaotic synchronization via analog transmission line," *The European Physical Journal Special Topics*, vol. 225, no. 1, pp. 119–126, feb 2016.

[5] L. G. Nardo, E. G. Nepomuceno, J. Arias-Garcia, and D. N. Butusov, "Image encryption using finite-precision error," *Chaos, Solitons & Fractals*, vol. 123, pp. 69–78, jun 2019.

[6] K. Vipin and S. A. Fahmy, "FPGA Dynamic and Partial Reconfiguration," *ACM Computing Surveys*, vol. 51, no. 4, pp. 1–39, sep 2018.

[7] Y. Wu, Y. Zhou, and L. Bao, "Discrete Wheel-Switching Chaotic System and Applications," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 12, pp. 3469–3477, dec 2014.

[8] D. K. Shah, R. B. Chaurasiya, V. A. Vyawahare, K. Pichhode, and M. D. Patil, "FPGA implementation of fractional-order chaotic systems," *AEU - International Journal of Electronics and Communications*, vol. 78, pp. 245–257, aug 2017.

[9] L. Palacios-Luengas, G. I. Duchen-Sánchez, J. L. Aragón-Vera, and R. Vázquez-Medina, "Digital Noise Generator Design Using Inverted 1D Tent Chaotic Map," *VLSI Design*, vol. 2012, no. 1, pp. 1–10, oct 2012.

[10] R. Corless, "What good are numerical simulations of chaotic dynamical systems?" *Computers & Mathematics with Applications*, vol. 28, no. 10-12, pp. 107–121, nov 1994.

[11] E. G. Nepomuceno, S. A. Martins, B. C. Silva, G. F. Amaral, and M. Perc, "Detecting unreliable computer simulations of recursive functions with interval extensions," *Applied Mathematics and Computation*, vol. 329, pp. 408–419, jul 2018.

[12] E. G. Nepomuceno, A. M. Lima, J. Arias-García, M. Perc, and R. Repnik, "Minimal digital chaotic system," *Chaos, Solitons & Fractals*, vol. 120, pp. 62–66, mar 2019.

[13] R. M. May, "Simple mathematical models with very complicated dynamics," *Nature*, vol. 261, no. 5560, pp. 459–467, jun 1976.

[14] W. S. Dorn, "Generalizations of Horner's Rule for Polynomial Evaluation," *IBM Journal of Research and Development*, vol. 6, no. 2, pp. 239–245, apr 1962.

[15] I. Öztürk and R. Kılıç, "Digitally generating true orbits of binary shift chaotic maps and their conjugates," *Communications in Nonlinear Science and Numerical Simulation*, vol. 62, pp. 395–408, sep 2018.

[16] A. Wolf, J. B. Swift, H. L. Swinney, and J. A. Vastano, "Determining Lyapunov exponents from a time series," *Physica D: Nonlinear Phenomena*, vol. 16, no. 3, pp. 285–317, jul 1985.

[17] F. Pareschi, R. Rovatti, and G. Setti, "On Statistical Tests for Randomness Included in the NIST SP800-22 Test Suite and Based on the Binomial Distribution," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 491–505, apr 2012.

[18] "A statistical test suite for random and pseudorandom number generators for cryptographic applications," apr 2010.

[19] G. Gugapriya, K. Rajagopal, A. Karthikeyan, and B. Lakshmi, "A family of conservative chaotic systems with cyclic symmetry," *Pramana*, vol. 92, no. 4, p. 48, apr 2019.