2023 36th SBC/SBMicro/IEEE/ACM Symposium on Integrated Circuits and Systems Design (SBCCI) | 979-8-3503-1834-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/SBCCI60457.2023.10261963

Design of an Advanced System-on-Chip Architecture for Chaotic Image Encryption

Arthur M. Lima*, Lucas G. Nardo†, Erivelton Nepomuceno‡, Janier Arias-Garcia†§, Jones Yudi*¶
*Graduate Program in Mechatronic Systems, University of Brasília, Brasília, Brazil.
†Graduate Program in Electrical Engineering, Federal University of Minas Gerais, Belo Horizonte, Brazil.
‡Center for Ocean Energy Research, Department of Electronic Engineering, Maynooth University, Maynooth, reland.
§Department of Electronic Engineering, Federal University of Minas Gerais, Belo Horizonte Brazil.
¶Department of Mechanical Engineering, University of Brasília, Brasília, Brazil.
arthur.mendes@aluno.unb.br, nardo@ufmg.br, erivelton.nepomuceno@mu.ie, janier-arias@ufmg.br, jonesyudi@unb.br

***Abstract*—With the rise of interconnected systems, security has become a crucial concern. As a result, there has been a growing interest in developing low-cost embedded cryptographic algorithms that are lightweight and can be integrated into System-on-Chip (SoC) devices. Digital chaotic systems have emerged as a promising approach for building secure communication systems, where various cryptosystems utilize chaotic dynamics to encrypt images into noise-like representations. Recent studies have demonstrated that finite-precision error can be used to derive chaos, which can be systematically applied to encrypt images. However, there is a lack of research on how SoC-based design constraints affect the overall performance of image encryption systems, especially for applications that are sensitive to latency. This study aims to address this gap by presenting an efficient architecture that explores a hardware/software co-design for image encryption based on finite-precision error, specifically designed for resource-limited devices and latency-sensitive applications. Our platform performs the capture and encryption of images with a size of 320×240. Using a benchmark image, results show that the developed cryptosystem architecture can encrypt images efficiently while offering low hardware occupation.**

***Index Terms*—System-on-Chip, chaos, chaotic systems, image encryption, finite-precision error, latency-sensitive applications.**

I. Introduction

Nowadays, there is a growing interest in developing lightweight and low-cost image encryption algorithms that can be embedded in resource-limited devices, such as System-on-Chip (SoC) products [1], [2]. These devices come with integrated sensors that can capture images and perform real-time surveillance, as well as sophisticated image-processing software that can recognize faces and read files. With the increasing adoption of SoC, ensuring image security has become a top priority since images contain a wealth of information [3].

Although there are many cryptographic methods available, traditional algorithms, such as 3DES (Triple Data Encryption Standard), Twofish, Blowfish, and AES (Advanced Encryption Standard), may not always be effective for encrypting images. Images have unique characteristics, such as strong pixel correlation and high redundancy, that make them difficult to encrypt using conventional methods [4], [5]. As a result, the use of chaotic systems for image encryption is increasingly being recommended since these systems possess properties that are necessary for ciphers to be considered secure [6].

Advancements in dedicated hardware solutions for embedded systems have led to the increased digital implementation of chaotic dynamics, which were previously difficult to achieve due to complex mathematical operations and latency-sensitive issues when implemented in a basic CPU (Central Processing Unit) using floating-point arithmetic [7]. In this paper, we present an image encryption architecture that uses chaos derived from the finite-precision error between two natural interval extensions of Chua's circuit [8] and implement it in SoC-FPGA (Field-Programmable Gate Array), taking advantage of the latest techniques for designing dedicated hardware solutions for embedded systems.

Based on the preceding discussion, this paper's key contributions can be summarized as follows:

- The presentation of a design for an efficient SoC-based image encryption scheme.
- An analysis of the proposed architecture's performance using various metrics, including hardware resource utilization and statistical tests.

The rest of the paper is described as follows: Section II introduces preliminary concepts to better understand the proposed solution of this work. Section III demonstrates the proposed architecture. Section IV and Section V present the performance of the image encryption algorithm under a series of tests and the final remarks of this paper, respectively.

II. Preliminary Concepts

A. Chua's circuit

Four linear components are used to create this chaotic circuit: a resistor, an inductor, and two capacitors; all

Lucas G. Nardo, MEng, was supported by FAPEMIG. Erivelton Nepomuceno, PhD, was supported by Brazilian research agencies: CNPq (Grant Nos. 465704/2014-0, 425509/2018-4 and 311321/2020-8) and FAPEMIG (Grant No. APQ-00870-17). Jones Yudi, PhD, was supported by Fundação de Apoio à Pesquisa do Distrito Federal (FAPDF) and by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq).

979-8-3503-1834-0/23/$31.00 ©2023 IEEE

connected by a nonlinear diode (NLD) called Chua's diode [9]. The dynamics of this system are described in (1). The electric current of Chua's diode is defined by $I_{NLD}(V_{C_1})$, as presented in (2), where the breaking points of the nonlinear diode are represented by B_p, and the slopes by G_a and G_b.

$$\begin{cases} C_1\dfrac{dV_{C_1}}{dt} &= \dfrac{V_{C_2}-V_{C_1}}{R} - I_{NLD}(V_{C_1}) \\ C_2\dfrac{dV_{C_2}}{dt} &= \dfrac{V_{C_1}-V_{C_2}}{R} + I_L \\ L\dfrac{dI_L}{dt} &= -V_{C_2}, \end{cases} \tag{1}$$

$$I_{NLD}(V_{C_1}) = \begin{cases} G_bV_{C_1} + B_p(G_b - G_a), & \text{if} \quad V_{C_1} < -B_p \\ G_aV_{C_1}, & \text{if} \quad |V_{C_1}| \leq B_p \\ G_bV_{C_1} + B_p(G_a - G_b), & \text{if} \quad V_{C_1} > B_p. \end{cases} \tag{2}$$

To create the image encryption architecture, we will prototype this chaotic system using programmable logic.

B. Finite-precision error

To obtain a clearer understanding of the method, the definitions are following presented based on the paper [10]:

Definition 2.1: Modeled by differential equations in the form $\dot{x}_n = f(x_n)$, a true orbit of a given system is a sequence of values represented by $x_n = [x_0,\ x_1,\ x_2,\ x_3, \ldots,\ x_n]$.

When performing a numerical simulation, it is not possible to compute the true orbit due to the finite precision and constraints of computers. Instead, a pseudo-orbit is calculated.

Definition 2.2: A pseudo-orbit m approximates a true orbit x_n and it is represented by $\hat{x}_{m,n} = [\hat{x}_{m,0},\ \hat{x}_{m,1}, \ldots,\ \hat{x}_{m,n}]$, such that $|x_n - \hat{x}_{m,n}| \leq \delta_{m,n}$, where $\delta_{m,n} \in \mathbb{R}_+$ is the error bound.

By using pseudo-orbits, it is possible to obtain an interval that can be used to determine the position of the true orbit. Nevertheless, identifying this interval is a challenging task since the values for the true orbit and error bound are not available. As a solution, an error threshold is established through the application of natural interval extensions.

Definition 2.3: An interval-valued function $F(X)$ of an interval variable X is a natural interval extension of a function $f(x)$, where $F(X) = f(x)$ holds. Here, an interval denotes a closed set of real numbers such that $x \in \mathbb{R}$ and $X = [\underline{X}, \overline{X}]$ [10]–[12].

The following is an example of a natural interval extension given by (3) and (4):

$$C_1\frac{dV_{C_1}}{dt} = \frac{V_{C_2}-V_{C_1}}{R} - I_{NLD}(V_{C_1}), \tag{3}$$

$$C_1\frac{dV_{C_1}}{dt} = \frac{V_{C_2}}{R} - \frac{V_{C_1}}{R} - I_{NLD}(V_{C_1}). \tag{4}$$

It is important to note that the equations mentioned above are mathematically identical. However, in the digital realm, the propagation of errors during a numerical simulation leads to the generation of different outcomes by these equations. Therefore, the method for determining the Lower Bound Error (LBE) of two natural interval extensions is as follows:

Definition 2.4: Consider two pseudo-orbits $\hat{x}_{a,n}$ and $\hat{x}_{b,n}$ derived from two natural interval extensions of the function $f(x)$. The lower bound error $\delta_{\alpha,n}$ is determined by (5):

$$\delta_{\alpha,n} = \frac{|\hat{x}_{a,n} - \hat{x}_{b,n}|}{2}. \tag{5}$$

By implementing the 4^{th} order Runge-Kutta (RK4) method, Chua's circuit was modeled twice using different pseudo-orbits as depicted in (3) and (4). The outcome is presented in Fig. 1. Employing a 32-bit precision, Fig. 1 shows the divergence of the pseudo-orbits and the gradual increase of errors as the iteration count N advances. It is noteworthy that the accuracy loss of decimal places in the simulation, closely linked to error propagation, can be observed through a logarithmic scale.

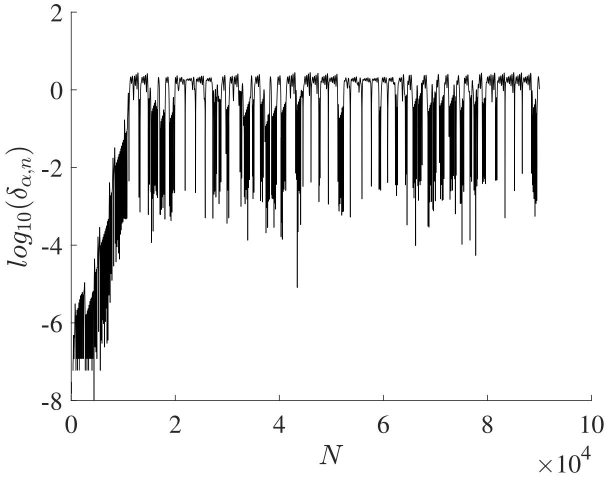


Fig. 1. The lower bound error from two pseudo-orbits of Chua's circuit. We use this random characteristic to create the keystream of our image encryption method.

C. Encryption algorithm

The keystream of the presented algorithm is derived from the pseudorandom sequence of the lower bound error. Based on the work in [8], the proposed encryption scheme can be summarized in the following steps:

Step 1: Read the grayscale plain image P_I with dimensions $H \times W$, where H and W represent the height and width of the image, respectively.

Step 2: Compute a factor F_P defined as:

$$F_P = \Big[\sum_{i=1}^{H}\sum_{j=1}^{W} P_I(i,j) \oplus P_{avg}\Big] \times 10^{-8}, \tag{6}$$

where i and j are the respective coordinate values of each pixel in the image P_I, P_{avg} is the image pixel average, and $\oplus$ refers to the bit-wise XOR operation. It is important to recognize that the factor F_P is dependent on the plain image that is being encrypted.

Step 3: Select a value for the initial condition V_{C_1} of Chua's circuit and incorporate the factor F_P into the chosen initial condition V_{C_1}:

$$V'_{C_1} = V_{C_1} + F_P. \tag{7}$$

In this way, we obtain a distinct keystream for each image, ensuring robustness against differential attacks.

Step 4: Choose the other initial conditions V_{C_2} and I_L, and simulate Chua's circuit twice, with different natural interval extensions. The discretization method is the RK4 method with an integration step of $h = 10^{-6}$. During the simulation, the number of iterations is determined by $tr + H \times W - 1$, wherein tr represents the number of transient iterations that are ignored.

Step 5: To derive a single sequence S, calculate the logarithm of the lower bound error from the two sequences S_1 and S_2 that were generated by each natural interval extension during the prior simulation:

$$S = log_{10} \frac{|S_1 - S_2|}{2}. \tag{8}$$

Step 6: Normalize the sequence S:

$$S_{Norm} = \text{uint8}(\text{mod}(S \times 10^5,\ 256)), \tag{9}$$

where uint8 is an algorithm utilized to transform the sequence into an 8-bit positive integer, while the operator mod denotes the modulo operation.

Step 7: Obtain the keystream K, transforming the sequence S_n in an array with equivalent format of the plain image P_I:

$$K = \text{vec2mat}(S_{Norm},\ W), \tag{10}$$

where vec2mat denotes the process of converting a vector into a matrix.

Step 8: To encrypt the plain image P_I, apply the bit-wise XOR operation between all of its pixels and the keystream K. The resulting output will be a cipher image C_I:

$$C_I(i,j) = P_I(i,j) \oplus K(i,j). \tag{11}$$

Figure 2 illustrates the image cryptosystem. After encrypting the image, the decryption process essentially involves reversing the encryption procedure, bringing the noise-like image back to its original form. It is worth noting that the image encryption algorithm presented here can be readily adjusted for colored images. For instance, the generation of the keystream can be accomplished using the methodology outlined in [13].

III. Finite-precision error-based cryptosystem for image encryption

A. The proposed pseudorandom number generator

An evaluation of whether the randomness induced is sufficient for image encryption algorithms is presented. The keystream was generated using the parameters listed below, based on Chua's circuit described in [14]: $L = 19$mH, $R = 1.8$KΩ, $C_1 = 10$nF, $C_2 = 100$nF, $G_a = -0.68$mS, $G_b = -0.37$mS, $B_p = 1.1$V. The initial conditions were: $V_{C_1} = 0.5$V, $V_{C_2} = -0.2$V, $I_L = 0$A. Chua's circuit was implemented using the identical equations shown in (1) and (2) with the natural interval extensions presented in (3) and (4). Moreover, the factor F_P, added to the initial condition V_{C_1}, was obtained from an 8-bit grayscale Lena image.

The resulting sequence was decoded into an 8-bit word considering the steps of the image encryption algorithm. Then, using the NIST test suite [15], from a keystream length of 1000000 bits and a significance level $\alpha = 0.01$, Table I shows that the sequence originated from the finite-precision error of Chua's circuit has random behavior. This yields enough arguments to consider feasibility in security. Such results discarded the 6000 initial iterations. This is due to the proximity between the pseudo-orbits at initial iterations.

TABLE I
The results of the NIST statistical test suite of the proposed Pseudorandom Number Generator (PRNG).

Test	P-value	Result
Frequency Test (Monobit)	0.231900487	Random
Frequency Test within a Block	0.721126262	Random
Run Test	0.012163257	Random
Longest Run of Ones in a Block	0.82235657	Random
Binary Matrix Rank Test	0.429011519	Random
Discrete Fourier Transform (Spectral) Test	0.23099701	Random
Non-Overlapping Template Matching Test	0.166886445	Random
Overlapping Template Matching Test	0.051919453	Random
Maurer's Universal Statistical test	0.053126029	Random
Linear Complexity Test	0.633084152	Random
Serial test	0.941514724 0.965833956	Random
Approximate Entropy Test	0.689632821	Random
Cummulative Sums (Forward) Test	0.125683449	Random
Cummulative Sums (Reverse) Test	0.416684402	Random
Random Excursions Test	0.252844376 0.498194057	Random

Using the profiling tool *gprof*, it is determined how much ARM execution time is spent in each part of Chua's circuit simulation by establishing a frequency for sampling rate and interrupting the system to check where it has stopped. The processor runs at $100MHz$ and $1MHz$ was settled for the frequency sampling. As this process of profiling is not entirely accurate, depending on the sampling clock frequency, it was taken the average of 100 profiling samples of 90000 solution elements. The profiling results show that most execution time is spent in the execution of the Natural Interval Extensions process, around 16.57%, followed by the Discretization process, around 23.13%, which are suitable processes for hardware implementation. Ultimately, 38.48% is taken by the standard initialization routines.

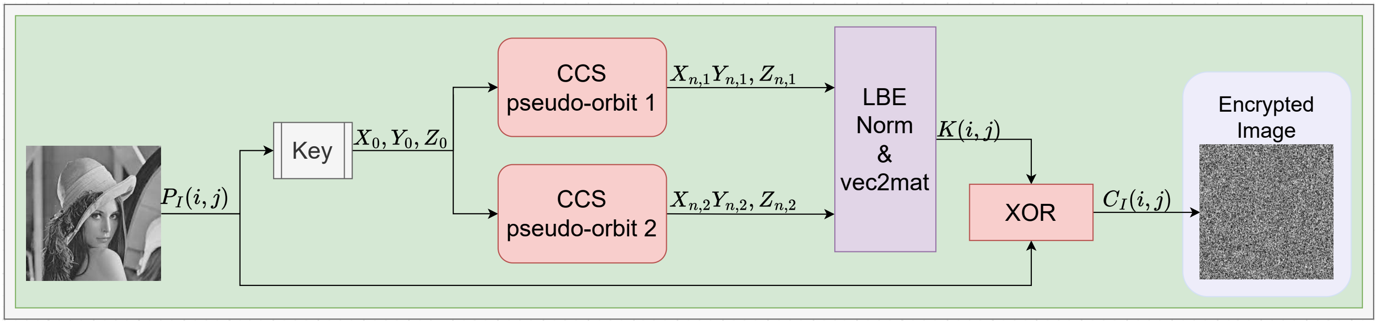


Fig. 2. The proposed encryption algorithm. The acronym CCS stands for Chua's circuit simulation.

B. Hardware/Software cryptosystem architecture

The proposed cryptosystem was implemented into the ZedBoard Evaluation Kit which consists of a Zynq-7000 device. This device is composed of three parts: (1) the Processing System (PS), which consists of a dual-core ARM Cortex A9 processor; (2) the Programmable Logic (PL), which is an FPGA fabric with features for optimization; and (3) the Advanced eXtensible Interface (AXI), which is a configurable interface for integrating the PS and PL. Additionally, a CMOS sensor is responsible for capturing the images used by the system. The digital part of the system receives the output from the CMOS sensor and performs all of the image processing involved in the encryption process. The prototype was developed using the Xilinx Design Tools for Hardware/Software co-design, which involved programming the PL using C/C++ with HLS (High-Level Synthesis) and synthesizing the hardware using HDL (Hardware Description Languages). Xilinx Tools also include a library of Intellectual Property (IP) blocks for synthesizing AXI interfaces and for general purposes.

The CMOS sensor outputs the image in RGB format that is later translated to 8-bit grayscale using (12), which is a weighted sum of red (R), green (G), and blue (B) values:

$$Gray = 5.1R + 10.2G + 1.74B. \tag{12}$$

The PS may issue the camera initialization by setting control registers, which configures image acquisition resolution. Images are captured at a maximum rate of 30 fps in QVGA (Quarter-Video-Graphic-Array) with a resolution of 320×240 pixels that is comparable to 256×256 benchmark images [16]. The AXI disposes of a VDMA (Video Direct Memory Access) interface that provides easy access to DDR (Double Data Rate). Images are automatically captured and stored directly in a frame buffer within the DDR and accessible to PS.

The proposed architecture, as presented in Fig. 3, assigns all calculations to PL. It is expected to achieve better performance results by using DSPs (Digital Signal Processors) and custom hardware.

Regarding the communication settings, the designed HLS hardware interface required three AXI channels: two streaming channels for data burst (input/output) working with an AXI DMA, and another memory-mapped channel to set the initial conditions and to initialize the process. The input and output stream channels are synchronized by the same source. This ensures that data is flown continuously during execution; the input variables $X_n,\ Y_n,\ Z_n$ should yield as soon as it is processed the output variables $X_{n+1},\ Y_{n+1},\ Z_{n+1}$.

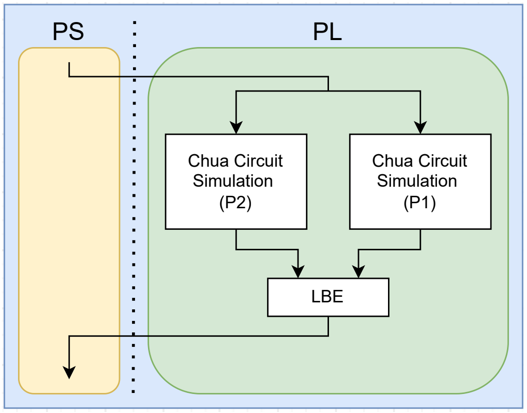


Fig. 3. Block level description of the proposed architecture.

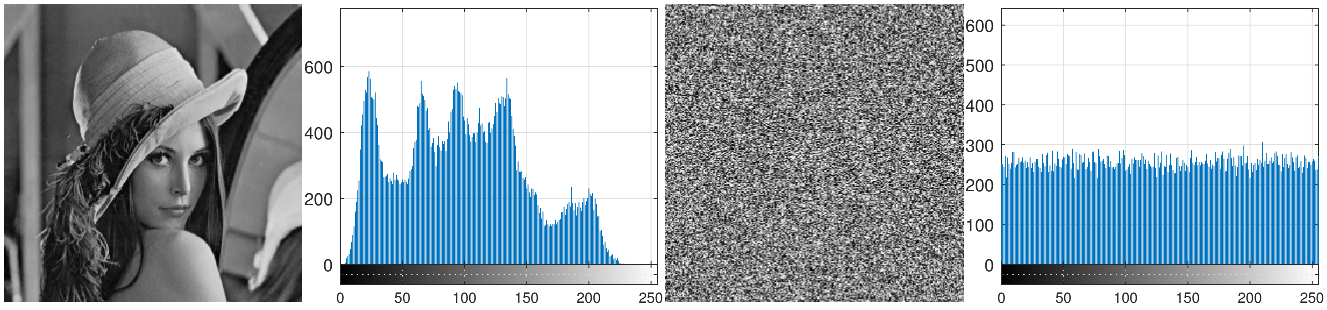
The main resources consumed by the proposed system using single floating-point precision are 19904 (37.41%) for LUTs (LookUp Tables), 904 (5.20%) for LUTRAMs (LookUp Table Random Access Memories), 20017 (18.81%) for FFs (Flip-Flops), 40.50 (28.93%) for BRAMs (Block Random Access Memories), and 147 (66.82%) for DSPs (Digital Signal Processors) of the total device available.

IV. Image Encryption Results

Although the architecture is capable of capturing QVGA images, we conducted statistical tests to evaluate the performance of our image encryption scheme employing the well-known Lena image, which has dimensions of 256×256.

A. Histogram analysis

By examining histograms of plain images, it is possible to identify patterns, as shown in Fig. 4-b. The encryption reshapes the histogram by rearranging pixel values to create a uniform distribution, as seen in Fig. 4-d. Effective encryption



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

Fig. 4. Results of image encryption using the Lena image along with the respective histograms of the plain and cipher images.

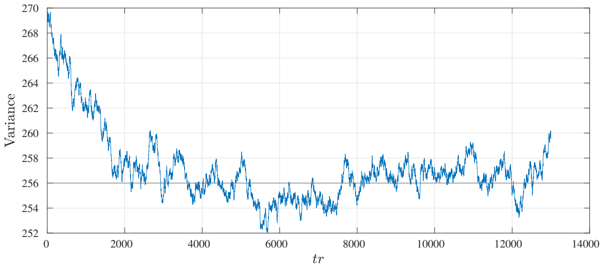


Fig. 5. Variance of the cipher image for the Lena image.

methods generate histograms that are closer to an ideal variance.

Concerning the cryptosystem, Fig. 5 shows the variance moving average. Initial results for $tr < 2000$ reveal a transient dumped stage before reaching a stable interval near the ideal value (256). Furthermore, variance is not sufficient to evaluate the uniform distribution of pixels [17], [18]. Some other metrics are important:

Maximum Deviation (M_D) is employed to measure the distance between the pixel-level distribution of plain image P_I and cipher image C_I:

$$M_D = \frac{D_0 + D_{255}}{2} + \sum_{i=1}^{254} |D_{C_{Ii}} - D_{P_{Ii}}|. \tag{13}$$

Larger deviations yield further disruption between cipher image and plain image which improves the security of the encryption system. In (13), D_i corresponds to the frequency of an specific pixel level i in a histogram.

Irregular Deviation (I_D) measure the distance between the cipher image to an statistical uniform distribution which considers the cipher image histogram average:

$$I_D = \sum_{i=0}^{255} |D_i - \mu_D|, \tag{14}$$

where μ_D is the average of pixel-level values. Small values of I_D indicate higher encryption security.

To measure the distance between the cipher image and a uniform distribution considering the ideal variance from $\sigma^2 = \frac{H \times W}{256}$, Deviation from Uniform Histogram (D_{UH}) can be calculated by:

$$D_{UH} = \frac{\sum_{i=0}^{255} |D_{C_{Ii}} - \sigma^2|}{H \times W}. \tag{15}$$

D_{UH} close to 0 indicates that the histogram of the cipher image is ideally and uniformly distributed.

The results in Table II were experimentally obtained from the average values for $tr > 8000$. When compared to other works, the proposed system scored the best result for the quantifier M_D, and the remaining results are very similar to the other ones.

TABLE II
HISTOGRAM RESULTS FOR ANALYSIS AND COMPARISON.

Image	Metric	Cipher image			
		Ours	Refs.		
		32 bits	[18]	[19]	[20]
Lena	M_D	38218	37982	37565	37435
	I_D	20258	19893	20032	20166
	D_{UH}	0.0497	0.0499	0.0496	0.0479

B. Adjacent pixels correlation analysis

The correlation coefficient measures the proximity of values between a pixel and its neighbors in an image [21]. For plain images, this measure should hold a significant value close to 1. For cipher images, the correlation coefficient must be close to 0 [22], [23]. Table III reaffirms the aforementioned, with the proposed encryption architecture presenting good results of correlation coefficients.

C. Information entropy analysis

Entropy is a measure of uncertainty within a communication system. It quantifies the degree of unpredictability or information content. For cipher images, entropy should be approximately 8. According to the average entropy measured during the initial transient in Fig. 6, the system is rapidly increasing until entropy reaches a stable interval near $H \approx 7.9972$ for $tr > 5000$.

TABLE III
CORRELATION COEFFICIENTS RESULTS.

Images		Correlation Coefficients			Mean
		Diagonal	Vertical	Horizontal	
Lena	Plain image	0.91793	0.96934	0.93998	0.94242
	Ref. [8]	0.00405	0.00302	0.00113	0.00273
	Ref. [24]	-0.00130	0.01410	0.00540	0.00693
	Ref. [25]	0.00470	0.02450	-0.01330	0.01417
	Ours (32 bits)	0.00359	0.00339	0.00102	0.00267

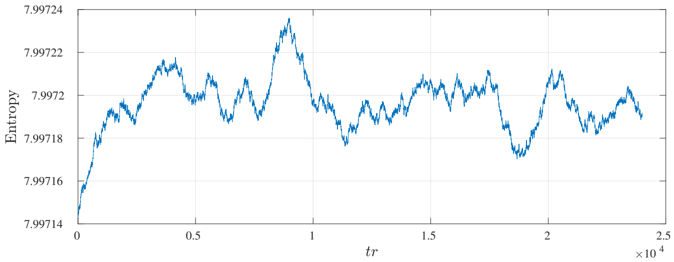


Fig. 6. Entropy of the cipher image for the Lena image.

D. Differential analysis

The resistance of an encryption system against differential attacks is commonly assessed using two metrics: the Unified Average Changing Intensity (UACI) and the Number of Pixel Change Rate (NPCR). For the Lena image, the NPCR and UACI of our study are 0.99621 and 0.33533, respectively, which demonstrates that the results adhere to the limits established in [26], indicating the algorithm's ability to withstand differential attacks.

V. CONCLUSION

In this paper, we introduce a real-time image encryption system implemented in an SoC device using High-Level Synthesis tools. The design was prototyped in a Zynq-7000 development kit ZedBoard. Our proposed finite-precision error-based encryption design utilizes finite-precision error of chaotic systems to generate the keystream, allowing for an efficient implementation of a hardware accelerator using FPGA reconfigurable logic. Results demonstrate that the proposed architecture is capable of encrypting images efficiently. Further research could explore different chaos-based encryption algorithms and investigate hardware/software trade-offs to enhance system performance.

REFERENCES

[1] B. Ramalingam, A. Rengarajan, and J. B. B. Rayappan, "Hybrid image crypto system for secure image communication – A VLSI approach," *Microprocess. Microsyst.*, vol. 50, pp. 1–13, March 2017.

[2] C.-H. Yang and S.-J. Huang, "Secure color image encryption algorithm based on chaotic signals and its FPGA realization," *Int. J. Circuit Theory Appl.*, vol. 46, pp. 2444–2461, December 2018.

[3] Z. H. Gan, X. L. Chai, D. J. Han, and Y. R. Chen, "A chaotic image encryption algorithm based on 3-D bit-plane permutation," *Neural Comput. Appl.*, vol. 31, pp. 7111–7130, November 2019.

[4] X. Chai, Y. Chen, and L. Broyde, "A novel chaos-based image encryption algorithm using DNA sequence operations," *Opt. Lasers Eng.*, vol. 88, pp. 197–213, January 2017.

[5] X. Wu, K. Wang, X. Wang, and H. Kan, "Lossless chaotic color image cryptosystem based on DNA encryption and entropy," *Nonlinear Dyn.*, vol. 90, pp. 855–875, October 2017.

[6] G. Alvarez and S. Li, "Some basic cryptographic requirements for chaos-based cryptosystems," *Int. J. Bifurcat. Chaos*, vol. 16, pp. 2129–2151, August 2006.

[7] C. H. Ho, C. W. Yu, P. Leong, W. Luk, and S. J. E. Wilton, "Floating-point FPGA: Architecture and modeling," *IEEE Trans. Very Large Scale Integr. VLSI Syst.*, vol. 17, pp. 1709–1718, December 2009.

[8] 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, June 2019.

[9] L. O. Chua, C. W. Wu, A. Huang, and G.-Q. Zhong, "A universal circuit for studying and generating chaos. I. Routes to chaos," *IEEE Trans. Circuits Syst. I Fundam. Theory Appl.*, vol. 40, pp. 732–744, October 1993.

[10] E. G. Nepomuceno and S. A. M. Martins, "A lower bound error for free-run simulation of the polynomial NARMAX," *Syst. Sci. Control Eng.*, vol. 4, pp. 50–58, January 2016.

[11] R. E. Moore, *Methods and applications of interval analysis*. Society for Industrial and Applied Mathematics, 1979.

[12] R. E. Moore, R. B. Kearfott, and M. J. Cloud, *Introduction to interval analysis*. Society for Industrial and Applied Mathematics, 2009.

[13] H. Liu and X. Wang, "Color image encryption based on one-time keys and robust chaotic maps," *Comput. Math. with Appl.*, vol. 59, pp. 3320–3327, March 2010.

[14] L. A. Aguirre and L. A. B. Tôrres, "Fixed point stability analysis of Chua's circuit: A case study with a real circuit," *J. Circuits Syst. Comput.*, vol. 07, pp. 111–115, April 1997.

[15] L. E. Bassham, A. L. Rukhin, J. Soto, J. R. Nechvatal, M. E. Smid, E. B. Barker, S. D. Leigh, M. Levenson, M. Vangel, D. L. Banks, N. A. Heckert, J. F. Dray, and S. Vo, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," National Institute of Standards and Technology, Tech. Rep. April, 2010.

[16] University of Southern California - Signal and Image Processing Institute, "USC-SIPI image database," http://sipi.usc.edu/database/, accessed 2023.

[17] Y. Y. Ghadi, S. A. Alsuhibany, J. Ahmad, H. Kumar, W. Boulila, M. Alsaedi, K. Khan, and S. A. Bhatti, "Multi-chaos-based lightweight image encryption-compression for secure occupancy monitoring," *J. Healthc. Eng.*, vol. 2022, p. 7745132, 2022.

[18] M. Alawida, J. S. Teh, A. Samsudin, and W. H. Alshoura, "An image encryption scheme based on hybridizing digital chaos and finite state machine," *Signal Process.*, vol. 164, pp. 249–266, November 2019.

[19] H. Diab, "An efficient chaotic image cryptosystem based on simultaneous permutation and diffusion operations," *IEEE Access*, vol. 6, pp. 42 227–42 244, July 2018.

[20] L. Huang, S. Cai, X. Xiong, and M. Xiao, "On symmetric color image encryption system with permutation-diffusion simultaneous operation," *Opt. Lasers Eng.*, vol. 115, pp. 7–20, April 2019.

[21] Y. Zhou, L. Bao, and C. L. P. Chen, "A new 1D chaotic system for image encryption," *Signal Process.*, vol. 97, pp. 172–182, April 2014.

[22] T. Hu, Y. Liu, L.-H. Gong, and C.-J. Ouyang, "An image encryption scheme combining chaos with cycle operation for DNA sequences," *Nonlinear Dyn.*, vol. 87, pp. 51–66, January 2017.

[23] R. Enayatifar, A. H. Abdullah, I. F. Isnin, A. Altameem, and M. Lee, "Image encryption using a synchronous permutation-diffusion technique," *Opt. Lasers Eng.*, vol. 90, pp. 146–154, March 2017.

[24] J. Gayathri and S. Subashini, "An efficient spatiotemporal chaotic image cipher with an improved scrambling algorithm driven by dynamic diffusion phase," *Inf. Sci.*, vol. 489, pp. 227–254, July 2019.

[25] L. Xu, Z. Li, J. Li, and W. Hua, "A novel bit-level image encryption algorithm based on chaotic maps," *Opt. Lasers Eng.*, vol. 78, pp. 17–25, March 2016.

[26] Y. Wu, J. P. Noonan, and S. Agaian, "NPCR and UACI randomness tests for image encryption," *Cyber J.*, vol. 2011, pp. 31–38, 2011.