Proceedings of Engineering and Technology Innovation, vol. 31, 2025, pp. 113-145

Hardware Implementation of Chaotic Image Encryption on FPGA

Zainab Karim Ibrahim[1,*], Mohammed Abd-Ali Al-Ibadi[1], Abbas Abdul-Ameer Jasim[2]

[1]Department of Computer Engineering, College of Engineering, University of Basrah, Basrah, Iraq

[2]College of Oil and Gas Engineering, Basrah University for Oil and Gas, Basrah, Iraq

Received 06 June 2025; received in revised form 11 October 2025; accepted 13 October 2025

DOI: https://doi.org/10.46604/peti.2025.15243

Abstract

This paper proposes a new FPGA-based architecture for secure image encryption, utilizing chaotic maps for permutation and substitution mechanisms. Specifically, Duffing map is employed to generate permutation addresses, and the Henon map for value substitution; both techniques produce pseudo-random sequences with strong sensitivity to initial conditions. The hardware design implemented using Xilinx System Generator and deployed on an Artix-7 FPGA supports both grayscale and RGB image formats. A comprehensive performance analysis, utilizing Mean Squared Error, Histogram analysis, Correlation Coefficient Analysis, Number of Pixels Change Rate, and Unified Average Changing Intensity, demonstrates perfect image recovery and robust encryption resistance. The keys used in the encryption system are passed using the NIST STS randomness test. The single-round and multi-round designs deliver a processing acceleration ranging from 2.11 to 3.92 for image sizes 128×128 to 1024×1024 pixels, highlighting their effectiveness and practicality for real-time encryption in low-latency environments.

Keywords: duffing chaotic map, henon chaotic map, FPGA, image cryptography, XSG

1. Introduction

In today's digital world, much important information is shared over unsecured channels. Media like text documents, voice files, and images are all private data that need to be preserved. Images become a part of many sensitive fields and need to be protected while transmitting over any environment. In the digital era, secure transmission of sensitive data such as images, voice files, and documents is increasingly vital. Images, in particular, are integral to fields such as medical imaging, defense, and biometric systems, where data confidentiality must be preserved. Traditional encryption algorithms—like DES, AES, and Triple DES—are not always efficient for image data due to their high processing time and storage requirements when applied to large multimedia files [1].

A digital image is fundamentally represented as a two-dimensional array of pixels, where each pixel encodes either intensity or multi-channel color information. Grayscale images typically utilize 8-bit values (ranging from 0 to 255) to indicate brightness levels, while RGB images consist of three separate 2D matrices representing the red, green, and blue components.

Due to the substantial data volume associated with high-resolution images, there is a critical need for lightweight and high-speed encryption algorithms that can ensure data confidentiality and maintain computational efficiency [2].

Chaos-based encryption has emerged as an effective alternative, leveraging the unpredictability and sensitivity of chaotic systems. Chaotic maps such as the Logistic, Henon, and Duffing maps generate pseudo-random sequences ideal for both permutation (shuffling) and substitution (value modification) operations in image encryption. This dual-stage encryption ensures resistance to statistical and differential attacks, which is crucial for robust security.

* Corresponding author: pgs.zainab.kareem24@uobasrah.edu.iq

Numerous chaotic maps have been developed and adapted over the years to serve image encryption purposes, including the Logistic map, Lorenz map, Henon map, and Duffing map. For instance, Singh et al. [3] proposed a 3D Duffing chaotic map for image encryption and achieved Number of Pixels Change Rate (NPCR) values of 96.78% for grayscale and 96.32% for color images at a resolution of 512×512, outperforming traditional 2D maps in speed. Similarly, Gao et al. [4] employed a combination of high-order bitwise XOR, scrambling, and diffusion, using an enhanced sine-tent map to deliver high randomness and robustness. In another study, Hartley's chaotic map was integrated with an iterative mathematical transformation to achieve an NPCR of 99.6% [5]. Feixiang et al. [6] introduced a hybrid encryption scheme using permutation and diffusion phases based on the Zigzag and Henon maps. Through multiple iterations of a chaotic restricted Boltzmann machine (CRBM), three pseudo-random number sequences were generated for XOR operations with the red, green, and blue channels of the scrambled image.

In parallel, several other researchers explored FPGA implementations of chaotic systems. For example, in [7], the Chua chaotic system was implemented using an FPGA to enhance generator frequency, achieving a maximum of 25 MHz. Wisal et al. [8] proposed an FPGA-based accelerator unit for image cryptography, designed and simulated using MATLAB/Simulink and validated via hardware co-simulation to evaluate resource utilization. Ali and Ali [9] improved encryption security using a PWL chaotic map to generate an S-box for pixel substitution and non-linearity. In [10], an artificial neural network (ANN) was designed to predict chaotic system behavior and implemented on an FPGA. Grimaldo et al. [11] employed a PWL chaotic map for FPGA-based encryption, expanding the key space while optimizing for limited-resource environments. Al-Musawi et al. [12] used AI-driven chaotic systems for image encryption, reporting a synthesis frequency of approximately 24.66 MHz. Furthermore, FPGA-based acceleration of image steganography using first and second LSB techniques achieved a remarkable 250× speedup while consuming just 1% of hardware resources [13].

These developments underscore the potential of hardware-based implementations in significantly enhancing the speed and reducing the latency of cryptographic systems. Therefore, integrating dedicated hardware units for image encryption is a compelling approach to meet the demands of modern high-performance security applications.

2. Chaotic Maps

This work utilizes two well-known chaotic maps: The Duffing map and the Henon map. The Duffing map is employed to perform permutation operations on image pixels' locations, while the Henon map is responsible for substitution, contributing to the confusion and diffusion properties of the encryption process.

2.1. Duffing Map

This map is a continuous-time system. It was created by Georg Duffing in 1918 [14-15]. It can be defined mathematically as:

$$\ddot{x} + \delta\ddot{x} + \alpha x + \beta x^3 = \gamma \cos(\omega t) \tag{1}$$

where x is the displacement, δ is the damping coefficient, α, β are linear and nonlinear stiffness terms, and γ cos (ωt) is any external forcing. Duffing map, shown in Fig. 1, was modified as the 2D discrete-time dynamical system described as:

$$\begin{aligned} x_{n+1} &= y_n \\ y_{n+1} &= -bx_n + ay_n + y^3{}_n \end{aligned} \tag{2}$$

It accepts an initial point x_n, y_n, and maps it to another point x_{n+1}, y_{n+1}, where a and b are the map parameters that control the dynamics and chaotic behavior of the system. Parameter a controls chaos robustness and sensitivity. The chaotic nature of the map is directly proportional to the value of a. Parameter b controls the output sequence complexity of the system.

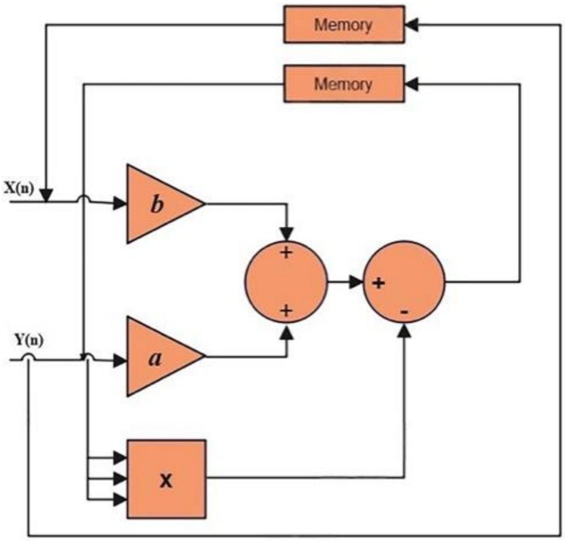


Fig. 1 Block diagram of the Duffing chaotic map

2.2. Henon Map

The Henon map is a discrete-time dynamical system that behaves chaotically. It was introduced by the French astronomer Michel Hénon in 1976 [16]. Mathematically, the Henon map, shown in Fig. 2, is a 2D iterative function defined by:

$$\begin{aligned} x_{n+1} &= y_n + 1 - a\,x^2{}_n \\ y_{n+1} &= b\,x_n \end{aligned} \tag{3}$$

Eq. (3) is a 2D map for obtaining a 1D sequence of values and can be simplified as:

$$x_{n+1} = 1 - ax^2 + bx_{n-1} \tag{4}$$

where x_n and y_n are the variables at the *n-th* iteration, and a and b control the chaotic behavior of the map.

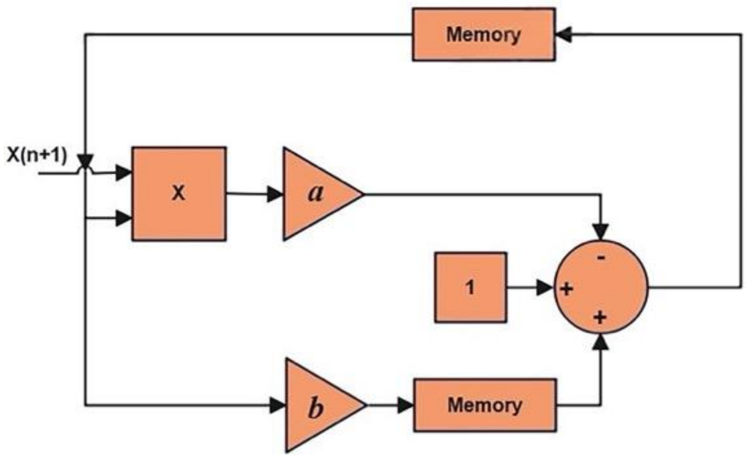


Fig. 2 Block diagram of the Henon chaotic map

3. FPGA Hardware Design for Grayscale Image Cryptography

The architecture proposed in this work for image cryptography, as illustrated in Fig. 3, is developed using the Xilinx System Generator (XSG) tool within the MATLAB/Simulink environment and is co-simulated on a Xilinx Artix-7 FPGA board. In this system, the pixels of the input image are first permuted using the Duffing chaotic map. Subsequently, each pixel is XORed with pseudo-random values generated by the Henon chaotic map as part of the substitution process, resulting in the final encrypted image output [17].

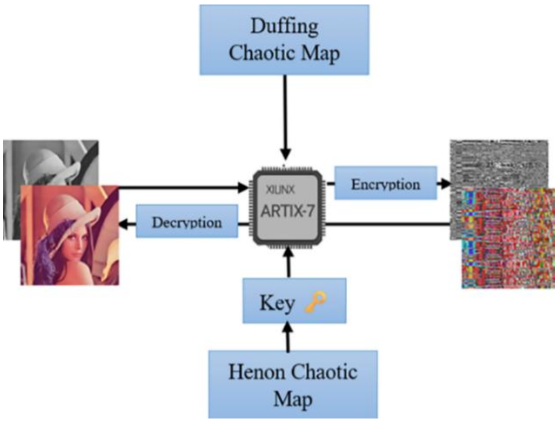


Fig. 3 Graphical Representation of Grayscale and RGB images encryption/ decryption system

The proposed hardware design for the digital image cryptographic system can be divided into three stages to produce the encrypted image. The first pre-processing stage is performed to prepare the image to enter the FPGA. The second stage includes the internal design of the FPGA using XSG. The last stage was performed to restore the final image setting to be ready to display.

3.1. *Pre-processing Stage*

After importing the plain image into the MATLAB/Simulink environment, multiple operations are presented to prepare the image before the encryption or decryption process. These steps, illustrated in Fig. 4, include:

(1) Transpose: A mathematical function to transpose pixels of the plain image.
(2) Reshape: Convert the image matrix to a 1D array output dimensionality.
(3) Frame Conversion: Set the sampling mode of the output signal to Frame-based.
(4) Unbuffer: Convert a frame to scalar samples, exported at a higher rate, to be familiar with FPGA speed.

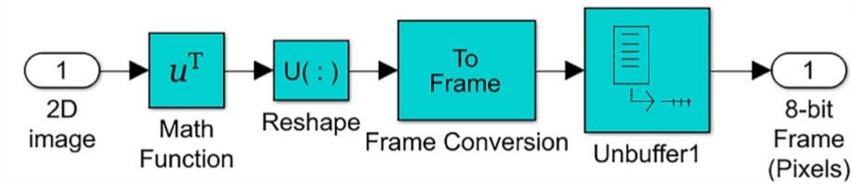


Fig. 4 MATLAB Simulink blocks of the image pre-processing stage

Fig. 5 illustrates the detailed settings of each block for the pre-processing block design. The math function block produces a real transposed image as an output. The reshape block converts the 2D transposed input image to a 1D array of pixels. The frame conversion block converts the sampling mode of the 1D input array to frame-based mode. Finally, unbuffed block upscales the sampling rate to be ready to enter the FPGA.

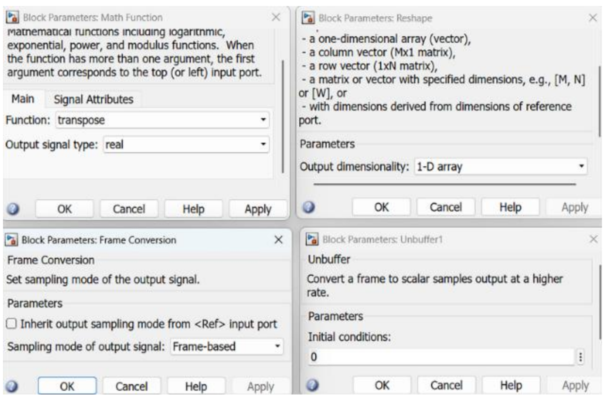


Fig. 5 Details of the MATLAB Simulink block for pre-processing

3.2. XSG Hardware Design Stage

To implement the hardware design, two iterated cascaded operations are performed in sequence. Each operation is implemented based on a chaotic map: the permutation stage is handled using the Duffing chaotic map, while the substitution stage is implemented using the Henon chaotic map.

3.2.1 Permutation

This operation has been driven by an RD/WR control signal, as shown in Fig. 6, to synchronize the Read or Write operation into the Single Port RAM with the correct address for each cycle. This control signal can be generated using a simple Relational block.

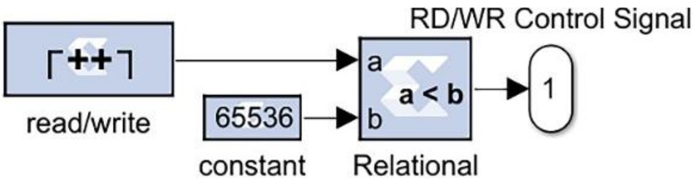


Fig. 6 XSG block design of the RD/WR control signal generator

When the RD/WR signal generates logic 1, the write cycle for the RAM starts, i.e., *we*=1, the pixels of the original input image are written into the RAM block using the address generated from the UP-Counter block. This counter counts from (0) to (image size -1), to address the memory locations for all image pixels. These image pixels are then read from RAM using the address generated by the Duffing Chaotic Map, as illustrated in Fig. 7 below.

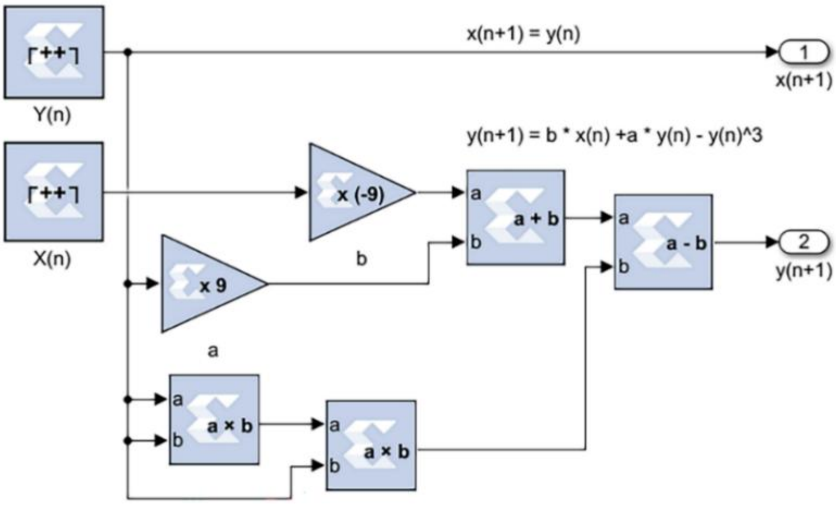


Fig. 7 XSG block design of the Duffing chaotic map

The Duffing map is a two-dimensional system that generates a pair of values, x and y, in each iteration. The generation of the pseudo-random sequence (key 2) is controlled by the control parameters a and b. For the proposed design of the permutation process, parameter a is set to 9, and parameter b is set to 9. The initial values of x and y are vectors that contain ascending values from 0 to N and 0 to M, respectively. The x, y initial values are set to fixed-point format with (n-bit) precision with no binary point, where n depends on the image dimensions. For a 128 × 128 image, the precision of the x and y is 8 bits wide, and so on. Since the RAM address space is one-dimensional, a simple mathematical conversion is applied to transform the 2D output of the Duffing map into a corresponding 1D address, as illustrated in Fig. 8, and can be defined as:

$$\text{1D address} = \text{Index}_{row} \times \text{Width}_{image} + \text{Index}_{column} \tag{5}$$

where Index_{row} and Index_{column} is the row and column indices, and the width_{image} is the width of the image.

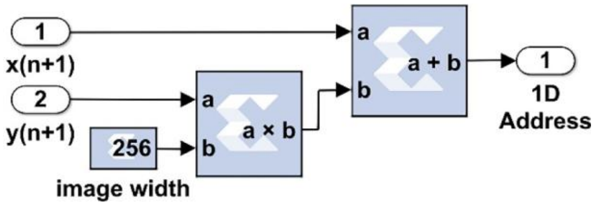


Fig. 8 XSG block design of the 2D to 1D address converter

The sequence of the Read and Write cycles, each using its corresponding address for the RAM, is controlled by a 2-to-1 Multiplexer as shown in Fig. 9. For writing the input image pixels in the single-port RAM, the sequential address is used to arrange the pixels. To read the pixels out of the single-port RAM chaotically, using a chaotic address produced by the duffing chaotic map.

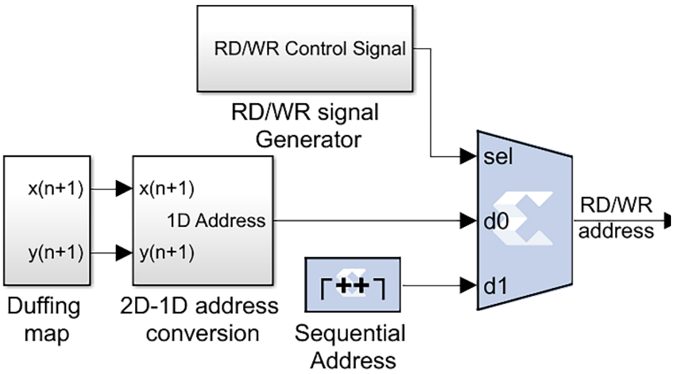


Fig. 9 XSG block design of the Read/Write address multiplexing.

Initially, the RAM block must be set before read and write cycles. The size of the RAM must be equal to the size of the image, so each location in the RAM is capable of storing an 8-bit pixel of the image. The initial value stored in the RAM can be any vector before being used to store pixels. The memory type used in the proposed system is *Block RAM*. To obtain effective read and write operations, the working mode of the RAM operation must be set to "*No read on write*,". Fig. 10 illustrates the settings of the Single-port RAM block used to process images of 256 × 256 size.

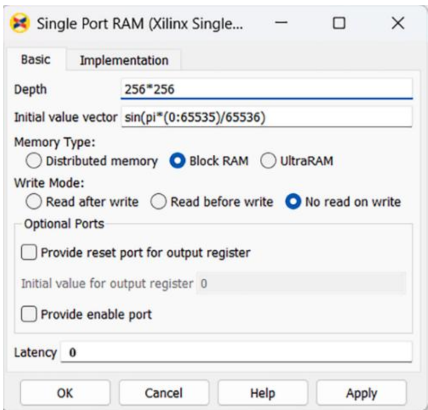


Fig. 10 XSG Single-port RAM block setting

The behavior of the RD/WR Control Signal Generator block is shown in Fig. 11. When it outputs a logic high (1), the multiplexer selects input d1, which corresponds to the sequential address generated by the UP-Counter. Simultaneously, the write enable signal (*we*) is set to 1, enabling the RAM block to enter the pixels during the write cycle. During this phase, all image pixels are written sequentially into the RAM locations. Conversely, when the control signal outputs a logic low (0), the multiplexer selects input d0, which represents the 1D permuted address derived from the Duffing map. At this point, (*we*) is set to 0, activating the read cycle and allowing the RAM to output pixels in a permuted (chaotic) order based on the Duffing-generated address sequence.

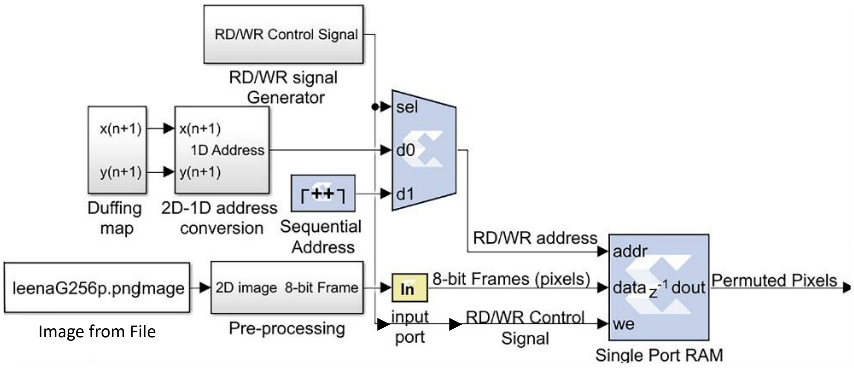


Fig. 11 XSG block design of the Read/Write cycles of the permutation process with a single-port RAM block

After reading the stored pixels out of the single-port RAM, the locations of the digital image pixels are permuted based on the address produced by the duffing chaotic map. As a result, the output image from the RAM block is produced as shown in Fig. 12. These permuted pixels are ready to enter the substitution process, which is the next stage of the proposed hardware cryptographic system.

(a) Original input image (b) Image after permutation

Fig. 12 Result of the permutation process on the 256 x 256 Lena image

3.2.2 Substitution

To strengthen the encryption process, a second layer of encryption is applied to the permuted image [18]. This is achieved through a substitution operation, in which pseudo-random values are generated using the Henon chaotic map, as defined in Eq. (4) and illustrated in Fig. 13. The generation of the pseudo-random sequence (*key 1*) is controlled by the control parameters *a* and *b*.

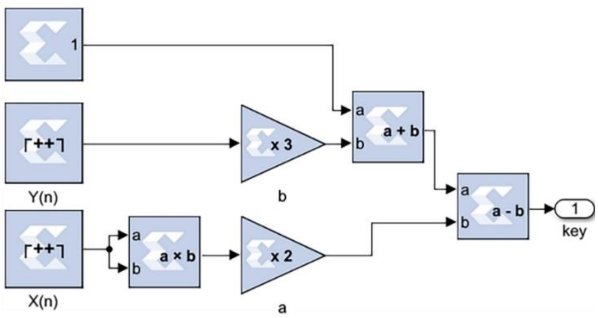


Fig. 13 XSG block design of the Henon chaotic map

For the proposed design of the substitution process, parameter *a* is set to 2, and parameter *b* is set to 3. The initial values of *x* and *y* are vectors that contain ascending values from 0 to N and 0 to M, respectively. Once the permuted image is retrieved from RAM, each pixel undergoes substitution by being XORed with the corresponding Henon-generated pseudo-random value. This operation enhances the overall encryption strength by introducing additional non-linearity and diffusion, resulting in a more secure encrypted image.

Fig. 14 illustrates the XOR operation (which represents the substitution process in this system) between the permuted pixels of the presented image (which are obtained from the permutation process) and a pseudo-random key sequence generated by the Henon chaotic map. The result of this process includes changing the intensities of the pixels to new values that cannot be discovered by any third party.

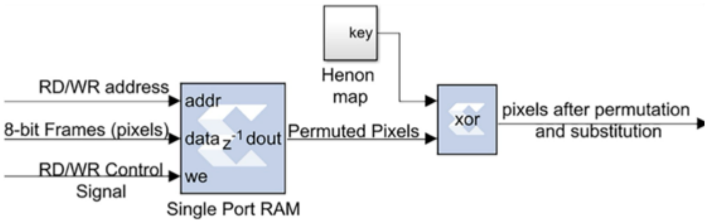


Fig. 14 XSG block design of the XOR between the henon map values and the permuted image pixels

3.3 Post-processing Stage

After the permutation stage, each pixel is immediately substituted before reaching the Gateway Out block. The post-processing stage is responsible for reassembling the final encrypted image. This stage comprises four MATLAB/Simulink blocks, as shown in Fig. 15, and is described as follows:

(1) Buffer: Converts scalar samples back into a frame-based output at a lower rate, ensuring compatibility with Simulink visualization and processing blocks.

(2) Reshape: Reconstructs the image by converting the 1D pixel array into a 2D image matrix.

(3) Transpose: Applies a matrix transposition to restore the correct spatial orientation of the image.

(4) Data Type Conversion: Converts the processed data into the uint8 format to match the expected image data type.

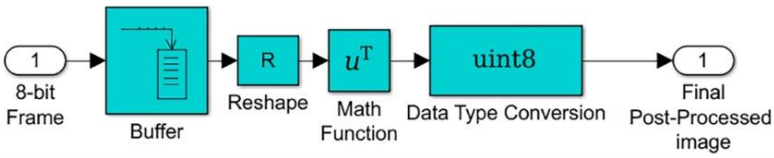


Fig. 15 MATLAB Simulink blocks of the image post-processing blocks

Fig. 16 illustrates the detailed settings of each block for this design. The buffer block receives the pixels after the permutation and substitution processes to maintain the sampling rate of the pixels that move from the FPGA to the Simulink environment. Another transpose function is needed on this side to reverse the effect of the transpose operation done in the encryption side. Reshape the 1D array to a 2D array of pixels again. Finally, convert the data type to an unsigned 8-bit integer type.

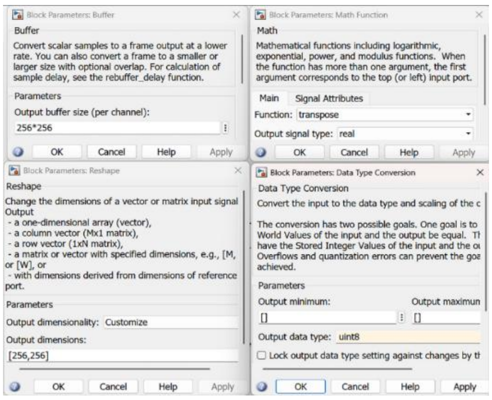


Fig. 16 Details of the MATLAB Simulink block for post-processing

The complete hardware design for the grayscale digital image encryption system is shown in Fig. 17. The complete hardware design of the encryption system, which includes the permutation and substitution processes, receives the plain image, performs pre-processing to prepare the pixels, then performs the permutation and substitution, and finally performs post-processing on the output to display the encrypted image.

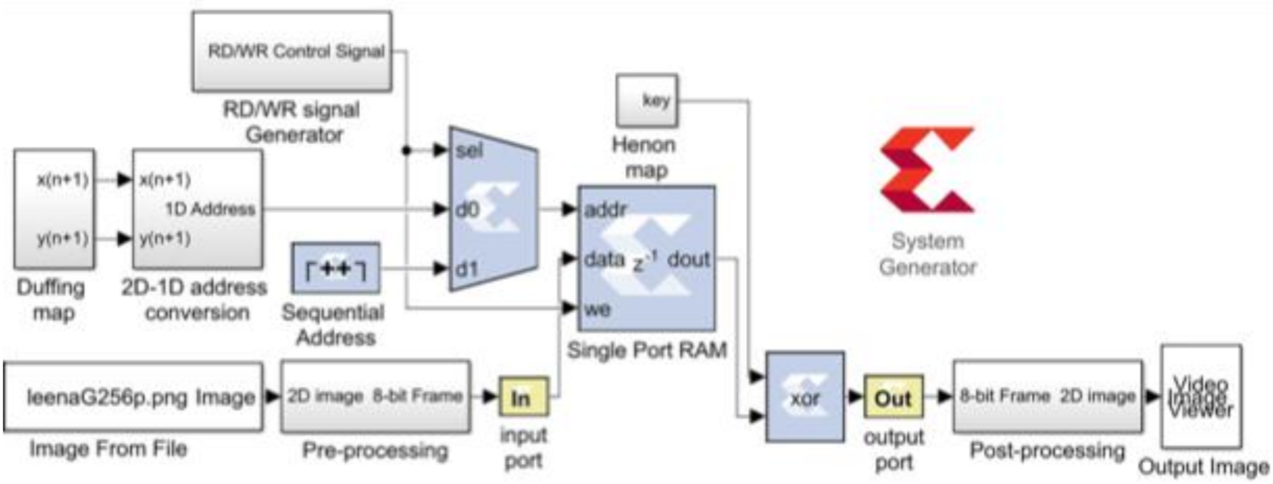
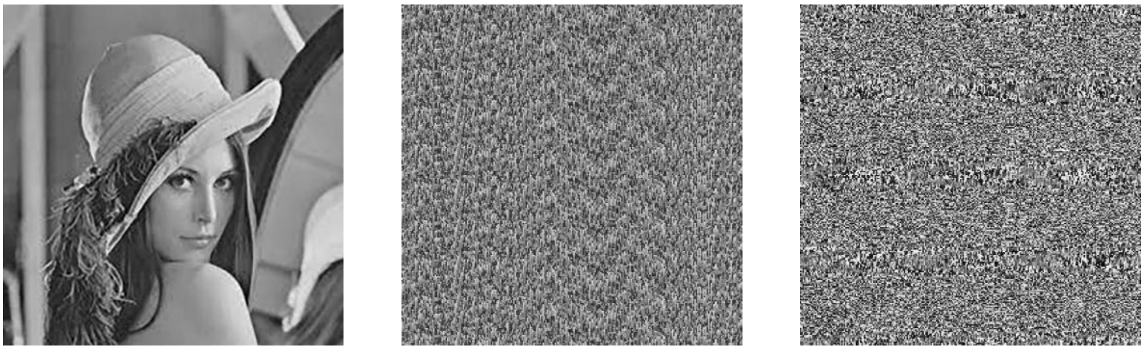


Fig. 17 XSG hardware design of a permutation substitution encryption system for grayscale images

Fig. 18 shows the original plain image, the permuted image, and the final encrypted image. From these results, it is clear how the two processes affect the encryption of the digital image, and how they try to reduce the patterns that help the intruder to discover the used algorithms in the encryption and recover the plain digital image.



(a) Original image (b) Permutation-only (c) Final encrypted image

Fig. 18 Image encryption results after a single round

The hardware design of the decryption process mirrors the encryption pipeline and begins with the substitution stage. In this stage, each pixel of the encrypted image is XORed with the same pseudo-random values generated by the Henon chaotic map that were used during encryption. This operation effectively reverses the initial substitution and restores the permuted image. The substitution stage is illustrated in Fig. 19.

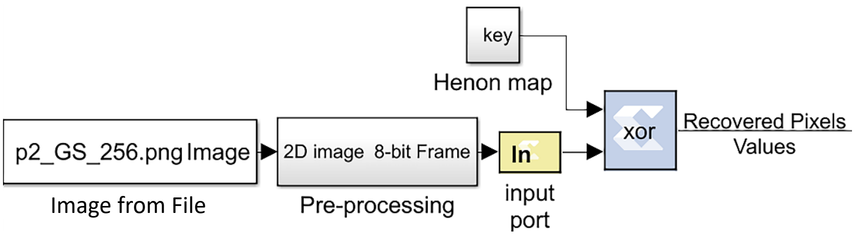


Fig. 19 Substitution in the decryption system

Next, the permutation stage is carried out by first selecting the 1D permuted addresses generated by the Duffing chaotic map to write the image pixels into the RAM. Subsequently, the sequential addresses from the UP-Counter are used to read the recovered image pixels from the RAM in their original order. This process effectively reverses the permutation applied during encryption. The permutation stage is illustrated in Fig. 20.

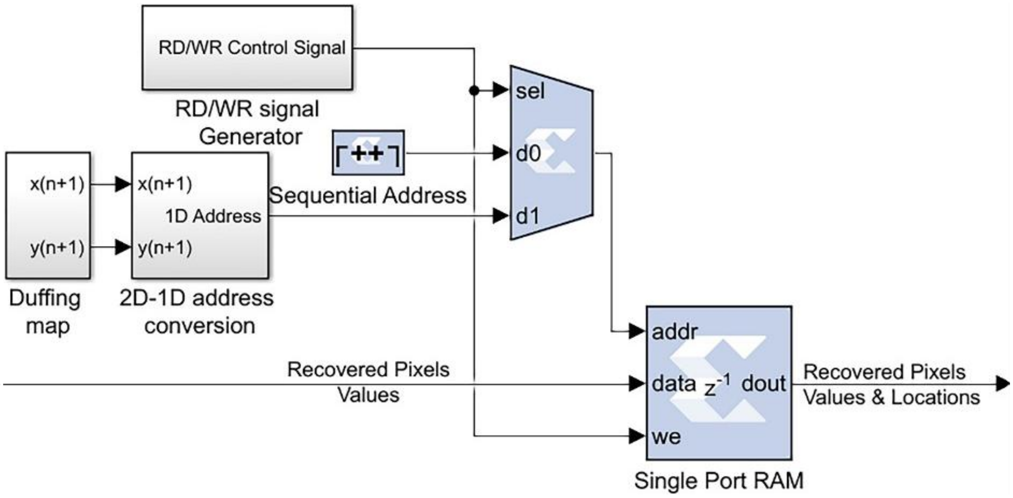


Fig. 20 Permutation in the decryption system

The complete hardware design of the decryption system is shown in Fig. 21. The complete hardware design of the decryption system, which includes the substitution and permutation processes, receives the encrypted image, performs pre-processing to prepare the pixels, then performs the substitution and permutation, and finally performs post-processing on the output to display the recovered image.

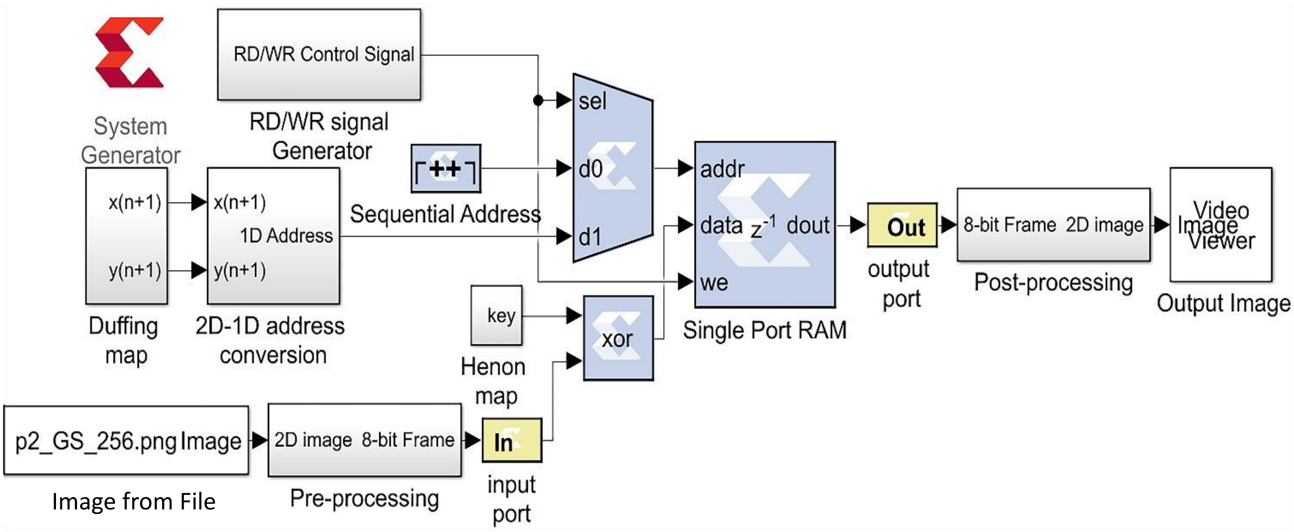


Fig. 21 XSG design of a substitution permutation decryption system for grayscale images

For key synchronization aspects, both sender and receiver entities share the same initial conditions and control parameters. Once synchronized, both sides evolve identically due to the deterministic nature of chaotic maps, enabling proper encryption and decryption processes.

4. XSG Hardware Design for RGB Image Cryptography

For RGB images, the proposed grayscale encryption hardware design can be extended to support the three-color channels: red, green, and blue [19-20]. To ensure time efficiency, the sequential and permuted addresses for the RAM blocks, the read/write control cycles, and the substitution processes are fully synchronized across all three channels. As illustrated in Fig. 22, the red, green, and blue channels operate in parallel, allowing their permuted data to be read and substituted simultaneously. This parallelism significantly enhances the overall processing speed of the hardware encryption system.

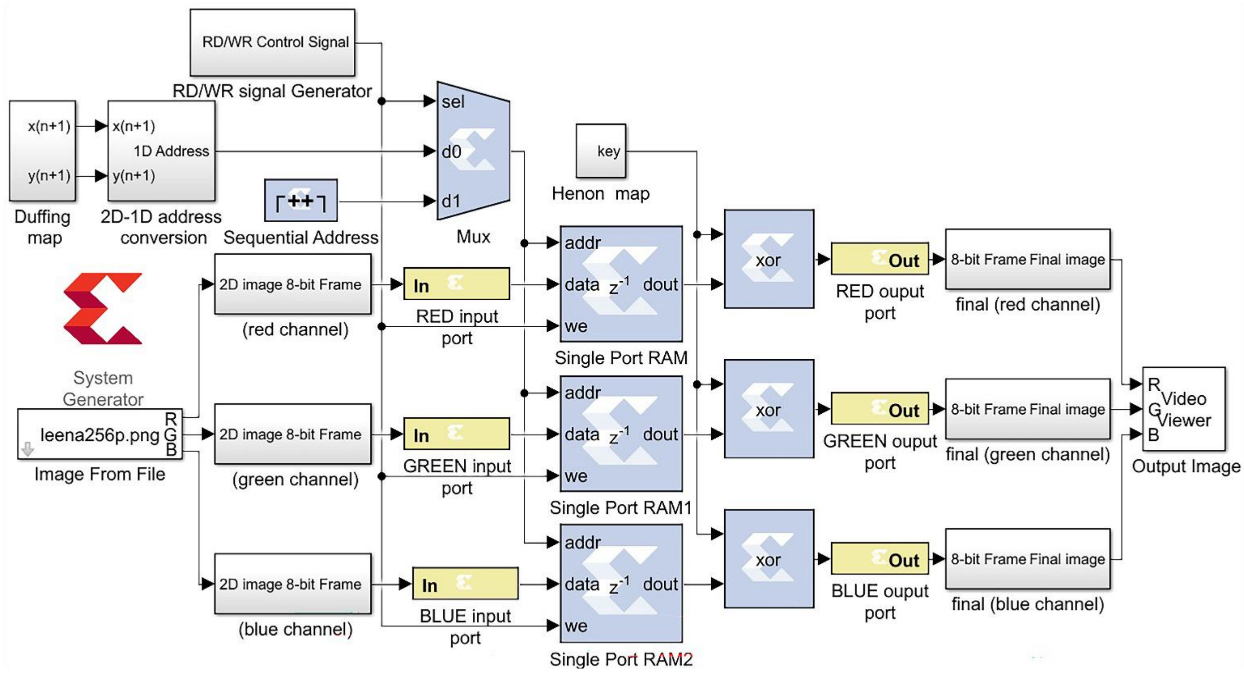


Fig. 22 XSG design for the RGB images encryption system

For the decryption process, the hardware design used for grayscale images is similarly extended to support the three channels of the RGB image. The address handling during the read/write cycles for the permutation stage, as well as the generation of pseudo-random numbers for the XOR-based substitution, is synchronized across all channels. This parallel execution accelerates the overall decryption process. Fig. 23 illustrates the synchronized three-channel operation of the RGB image decryption system.

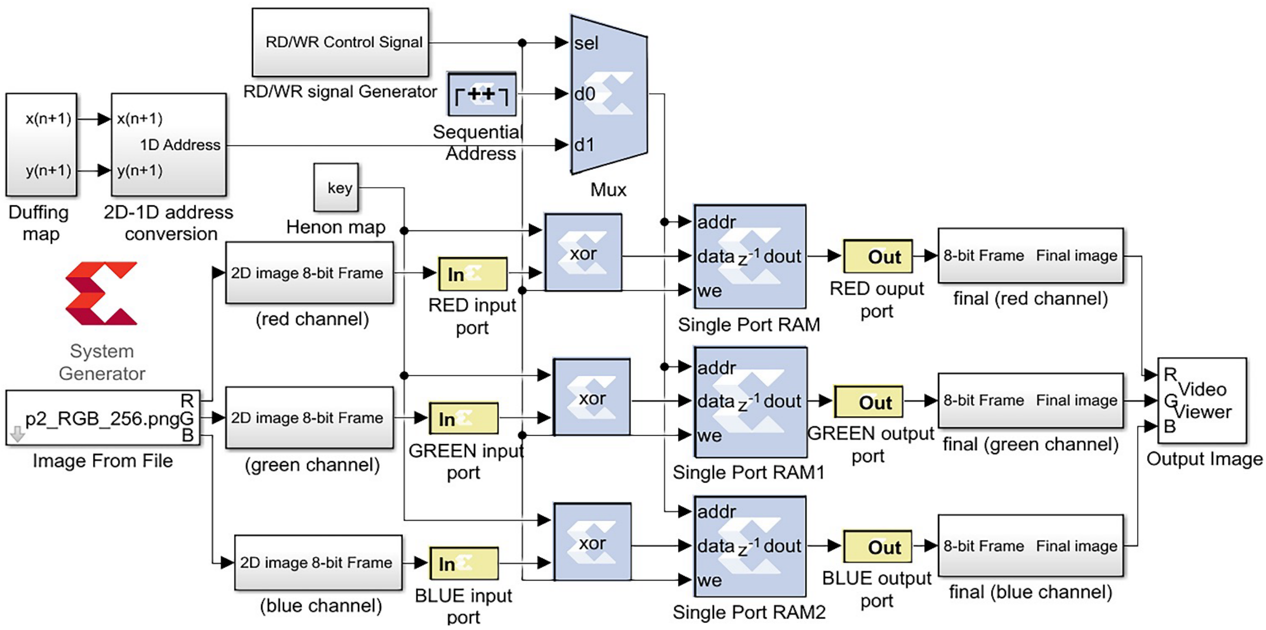
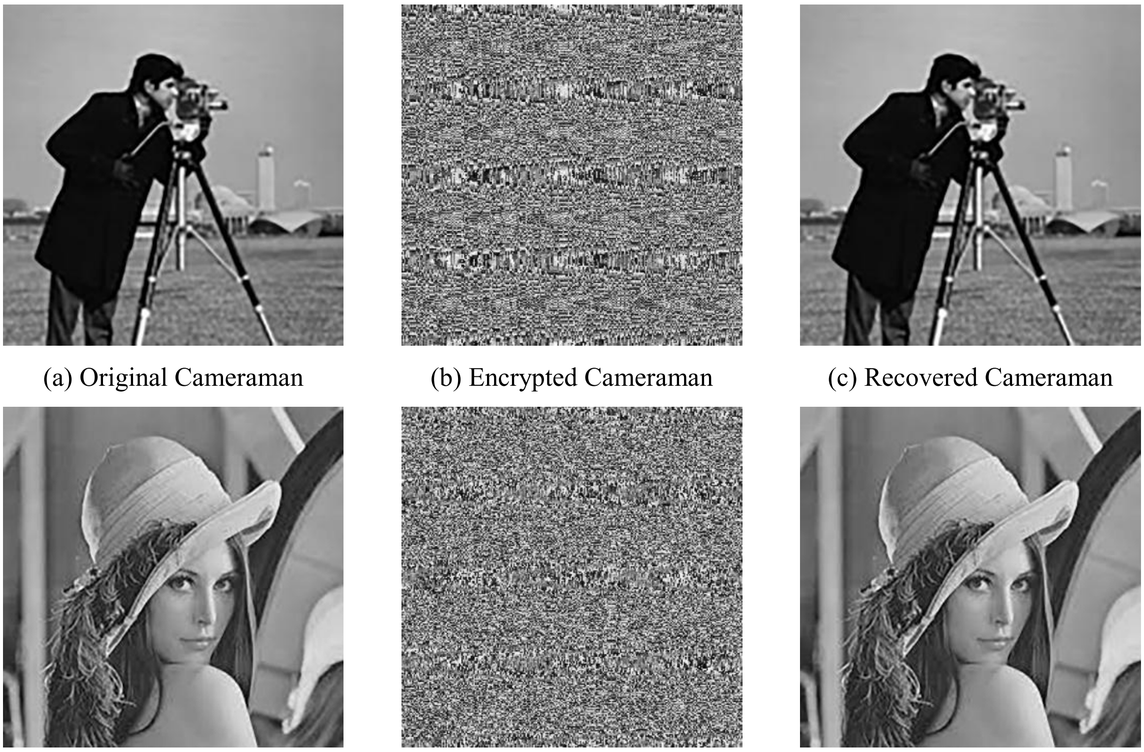


Fig. 23 XSG design for the RGB images decryption system

5. Results

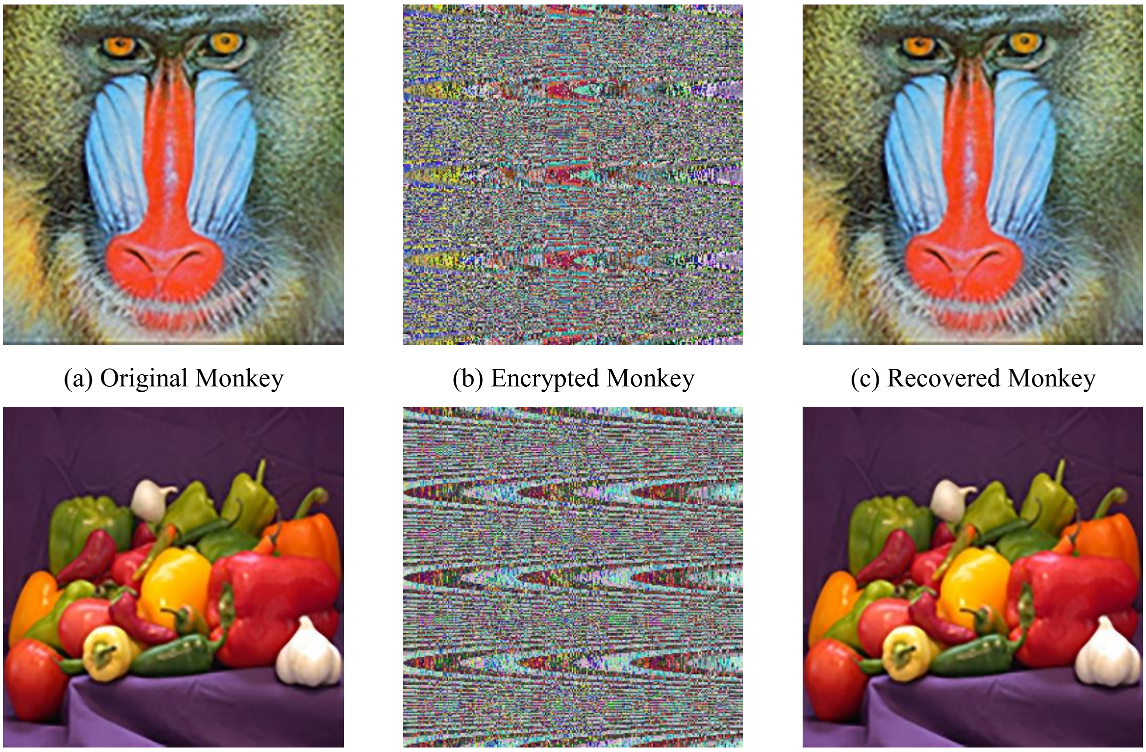
The proposed hardware design was tested using grayscale and RGB images of various sizes: 128×128, 256×256, 512×512, and 1024×1024. Grayscale images such as Cameraman and Leena, along with RGB images like Monkey and Peppers, were used to test the encryption and decryption performance. Figs. 24-25 presents a sample result for a 256×256 image processed by the encryption/decryption system.



(a) Original Cameraman (b) Encrypted Cameraman (c) Recovered Cameraman

(d) Original Leena (e) Encrypted Leena (f) Recovered Leena

Fig. 24 Grayscale images' testing results of the proposed hardware system



(a) Original Monkey (b) Encrypted Monkey (c) Recovered Monkey

(d) Original Peppers (e) Encrypted Peppers (f) Recovered Peppers

Fig. 25 RGB images testing results of the proposed hardware system

6. System Evaluation

To evaluate the performance of the proposed hardware cryptographic systems, two categories of metrics were employed: security metrics and hardware processing time. To ensure both the security robustness of the system, and the processing speed advancement between the software program and the hardware implementation of the same system.

6.1. Security Metrics

Three metrics were used to assess the system's ability to accurately recover the original plain image using MATLAB/Simulink scripts: Mean Squared Error (MSE), Correlation Coefficient Analysis (CCA), and NPCR.

6.1.1. Mean Squared Error (MSE)

The Mean Squared Error is a technique used to measure image recovery correctness. It checks that the original image and recovered image are identical. It can be calculated by using:

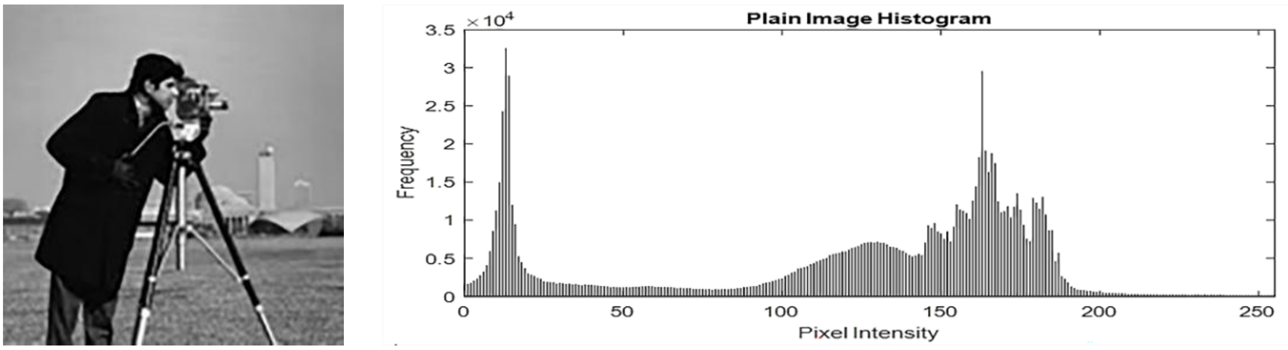
$$MSE = \frac{1}{M \times N}\left(\sum_{i=1}^{N}\sum_{j=1}^{M}\left(org\_img(i,j) - rec\_img(i,j)\right)^2\right) \tag{6}$$

where M and N are image dimensions, org_img(i,j) and rec_img(i,j) are the $(i,j)^{th}$ pixel position of the original and recovered images, respectively [21].

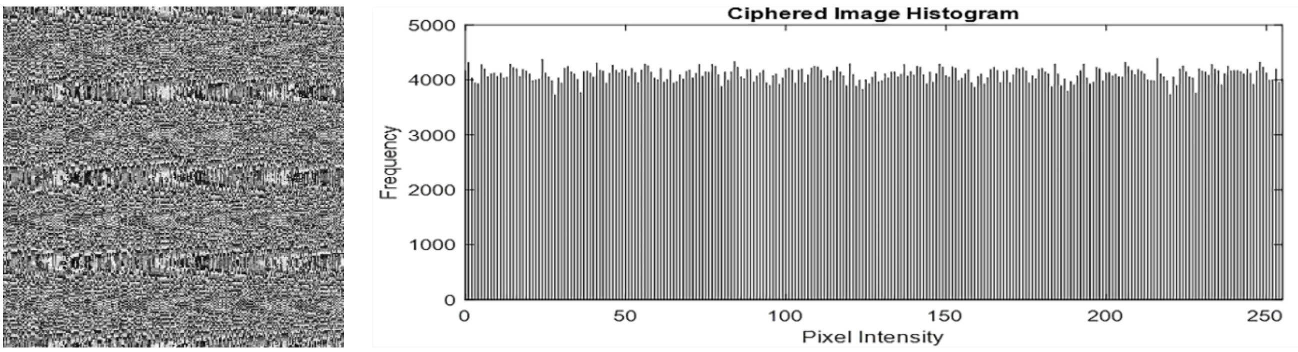
For the proposed hardware-based encryption/decryption system, the MSE between the original and recovered images was consistently zero across all tested image sizes and types (both grayscale and RGB), indicating perfect image recovery by the decryption process.

6.1.2. Histogram Analysis

A histogram is a statistical function that describes the distribution of pixel intensities over the digital image for each color level. It is commonly used to test image encryption accuracy. Strong image encryption systems should present a flat distribution of pixels' intensities [18]. Figs. 26-27 illustrates the histogram analysis of a sample of grayscale and RGB images sized 256 × 256.

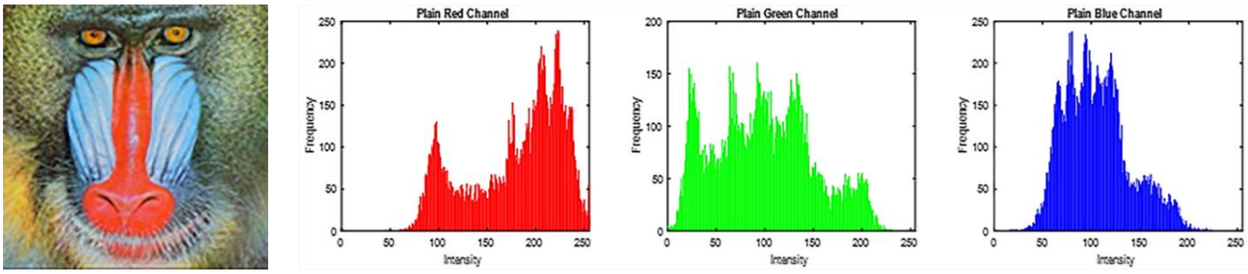


(a) Plain image

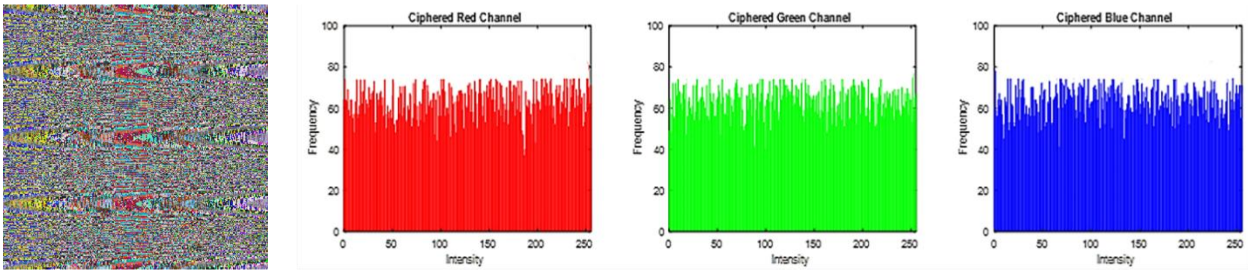


(b) Encrypted image

Fig. 26 Histogram of a 256×256 plain and encrypted grayscale image



(a) Plain image



(b) Encrypted image

Fig. 27 Histogram of a 256×256 plain and encrypted RGB image

6.1.3. Coefficient Correlation Analysis (CCA)

Another statistical metric used to evaluate the encryption performance is the Correlation Coefficient Analysis (CCA), which measures the relationship between adjacent pixels in an image. An effective encryption algorithm should significantly reduce the correlation among neighboring pixels, whether in the horizontal, vertical, or diagonal directions, resulting in correlation coefficients close to zero. The decorrelation indicates high randomness in the encrypted image and improved resistance to attacks; the CCA values are calculated by:

$$CCA(x,y)=\frac{E\left[\left(x-\mu_x\right)\left(y-\mu_y\right)\right]}{\left(\sigma_x\sigma_y\right)} \tag{7}$$

where x, y are adjacent pairs of pixels, μx, μy are the mean values of x and y, σx, σy are the standard deviations of x and y, and $E[\cdot]$ is the expectation function [12]. Tables 1-2 present the CCA values of grayscale and RGB images in plain and encrypted form.

Table 1 CCA values for grayscale plain and encrypted images

Image		Cameraman			Leena		
		H	V	D	H	V	D
128x128	Plain	0.9502	0.9510	0.9510	0.9586	0.9583	0.9583
	Encrypted	-0.0035	-0.0042	-0.0018	0.0025	0.0037	0.0041
256x256	Plain	0.9486	0.9506	0.9506	0.9941	0.9941	0.9941
	Encrypted	0.0021	-0.0050	-0.0045	0.0041	-0.0029	-0.0029
512x512	Plain	0.9969	0.9970	0.9970	0.9993	0.9993	0.9993
	Encrypted	0.0043	0.0031	0.0033	-0.0044	-0.0026	-0.0022
1024x1024	Plain	0.9998	0.9998	0.9998	0.9999	0.9999	0.9999
	Encrypted	0.0024	-0.0054	0.0049	-0.0018	-0.0021	0.0027

Table 2 CCA values for RGB plain and encrypted images

Image		128 × 128			256 × 256			512 × 512			1024 × 1024		
		R	G	B	R	G	B	R	G	B	R	G	B
Peppers Plain	H	0.7909	0.9387	0.9240	0.8420	0.9653	0.9437	0.9567	0.9882	0.9869	0.9968	0.9993	0.9991
	V	0.7890	0.9346	0.9186	0.8401	0.9643	0.9420	0.9562	0.9880	0.9867	0.9967	0.9992	0.9991
	D	0.7890	0.9346	0.9186	0.8401	0.9643	0.9420	0.9562	0.9880	0.9867	0.9967	0.9992	0.9991
Peppers Encrypted	H	-0.0192	-0.0243	0.0743	-0.0329	0.0532	0.0765	-0.0213	0.0413	0.0412	0.0112	0.0326	-0.0217
	V	0.0184	-0.0254	-0.0698	0.0233	-0.0544	-0.0645	-0.0210	0.0392	0.0403	0.0107	0.0202	0.0112
	D	-0.0184	0.0250	0.0695	0.0237	0.0539	0.0631	0.0385	0.0371	0.0450	0.0100	0.0218	-0.0111
Monkey Plain	H	0.9390	0.9044	0.9249	0.8989	0.8552	0.8565	0.8902	0.8600	0.8441	0.9911	0.9883	0.9882
	V	0.9383	0.9041	0.9249	0.8977	0.8542	0.8565	0.8885	0.8580	0.8433	0.9911	0.9882	0.9881
	D	0.9383	0.9041	0.9249	0.8977	0.8542	0.8565	0.8885	0.8580	0.8433	0.9911	0.9882	0.9881
Monkey Encrypted	H	-0.0209	0.0172	0.0720	0.0381	-0.0257	0.0337	0.0562	-0.0382	-0.0684	0.0207	-0.0356	-0.0411
	V	0.0215	-0.0141	0.0708	0.0403	-0.0243	0.0320	0.0550	0.0780	0.0670	0.0210	-0.0342	0.0420
	D	0.0222	-0.0143	0.0710	0.0033	0.0245	-0.0409	0.0558	0.0780	-0.0669	-0.0210	0.0342	0.0417

Correlation values range between −1 and +1. As shown in the tables above, values close to zero indicate the absence of any significant linear relationship between adjacent pixels, which is a highly desirable property in secure image encryption applications.

Table 3 presents a comparison study of the CCA values for the proposed hardware cryptographic design with those of other existing works for an image sized 256 × 256. It can be noted that the proposed design provides a good permutation for image pixels, which are interpreted from the low correlation values between horizontal, vertical, and diagonal adjacent pixels.

Table 3 Comparison study for CCA with previous works

Image name		Proposed design			[8]	[22]		
		H	V	D	Avg.	H	V	D
Peppers	R	-0.0192	0.0184	-0.0184	-0.076	-	-	-
	G	-0.0243	-0.0254	0.0250	-0.0290	-	-	-
	B	0.0743	-0.0698	0.0695	0.0050	-	-	-
Monkey	R	0.0381	0.0403	0.0033	-	-0.0728	0.0383	-0.0051
	G	-0.0257	-0.0243	0.0245	-	-0.0727	0.0265	-0.0017
	B	0.0337	0.0320	-0.0409	-	-0.0691	0.0359	-0.0015
Leena		0.0041	-0.0029	-0.0029	-	0.0056	-0.0024	-0.0035

6.1.4. Information Entropy

Information entropy is also used to test randomness. It quantifies the unpredictability or randomness of the pixel intensities in the encrypted image. Entropy measures the mean uncertainty or randomness in the image's pixel intensities using the plain and encrypted image [15], and can be defined by:

$$H(X) = -\sum_{j=0}^{m}\sum_{i=0}^{n} p(x(i,j)) \cdot \log_2(p(x(i,j))) \tag{8}$$

where *p(x(i,j))* is the probability of occurrence of pixel *x* at (i, j) index. Table 4 shows the information entropy results for 256 × 256 plain and encrypted images, comparing them with previous works. As can be seen, all of the entropy values are close to 8. As a result, the methods that are used are more efficient and secure.

Table 4 Comparison Study on Information Entropy

Name	Channels	Proposed design	[8]
Peppers	R	7.9993	7.9906
	G	7.9972	7.9903
	B	7.9920	7.9908

6.1.5. Number of Pixels Change Rate (NPCR) and Unified Averaged Changed Intensity (UACI)

NPCR is a fundamental quantitative metric used to evaluate the diffusion properties of image encryption systems. It can be represented as:

$$NPCR = \frac{\sum_{i,j} D(i,j)}{(M \times N)} \tag{9}$$

where:

$$\sum_{i,j} D(i,j) = \{ {}^{1}_{0} \;\; {}^{if}\;\; {}^{C \neq P}_{otherwise} \tag{10}$$

M × N: image dimensions, and C, P = Encrypted-image and Plain-image, respectively.

UACI measures the mean difference in pixel intensity between the plain and encrypted images, indicating the encryption scheme's ability to diffuse changes. It can be calculated by:

$$UACI = \frac{1}{MN}\left[\sum_{i,j} \frac{|\,\mathrm{C1(i, j)} - \mathrm{C2(i, j)}\,|}{255}\right] \times 100\% \tag{11}$$

where C_1 and C_2 are two encrypted images to be tested, divided by the maximum pixel intensity (for 8-bit pixels), and MN are the image dimensions. For various test images, the results are presented in Table 5.

Table 5 NPCR and UACI values for different image sizes

Type	Name		128×128		256×256		512×512		1024×1024	
			NPCR	UACI	NPCR	UACI	NPCR	UACI	NPCR	UACI
Grayscale Images	Leena		99.7667%	33.3215	99.6777%	33.3352	99.6095%	33.4099	99.6186%	33.3798
	Cameraman		99.6995%	33.4289	99.6362%	33.4295	99.6574%	33.4498	99.6521%	33.4051
RGB Images	Peppers	R	99.6826%	33.3155	99.6078%	33.3945	99.6887%	33.3287	99.7517%	33.3364
		G	99.6521%	33.3599	99.6658%	33.4412	99.7402%	33.4178	99.7975%	33.4511
		B	99.6812%	33.4372	99.6201%	33.4526	99.7780%	33.4509	99.8468%	33.4537
	Monkey	R	99.5934%	33.4066	99.6231%	33.4218	99.5907%	33.4431	99.5925%	33.4261
		G	99.6277%	33.3956	99.6641%	33.4337	99.6205%	33.4299	99.9424%	33.4195
		B	99.6155%	33.4510	99.6228%	33.4502	99.6143%	33.3381	99.5965%	33.4377

NPCR ≥ 99.58% means the system exhibits a strong avalanche effect. A single pixel change in the plaintext affects almost every pixel in the encrypted image. This is a hallmark of a secure cipher resistant to differential cryptanalysis. While 33.4635% is the ideal value for UACI, the results in the range of 33.3% to 33.6% are generally considered to be acceptable for a secure encryption scheme [23-24].

Table 6 presents a comparison study that examines the NPCR and UACI for the proposed hardware cryptographic design with previous studies. The comparison was made using grayscale images of Leena and the Cameraman of sizes 256 × 256. The proposed hardware cryptographic system achieved high values for NPCR and UACI as compared with existing work.

Table 6 Comparison study of NPCR and UACI values with the previous studies

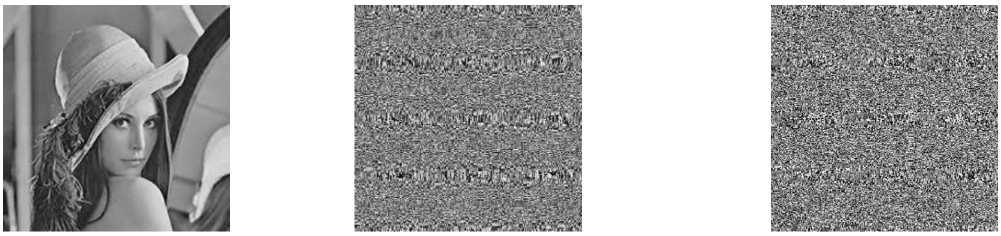
Name	Proposed design		[4]	
	NPCR	UACI	NPCR	UACI
Cameraman	99.6362%	33.4295	-	-
Leena	99.6777%	33.3352	99.5979%	33.3695

6.1.6. Key sensitivity analysis

For a robust cryptographic system, the key used in the encryption process must be sensitive to any change. While utilizing the chaotic maps for generating the keys that are used in both the permutation and the substitution processes, these maps are characterized by their sensitivity to initial conditions and control parameters. So, a small change in the key led to completely different encryption results [5]. Fig. 28 illustrates the different encrypted results of Peppers's image with two different keys shown in Table 7. The original image in Fig. 28 (a) was encrypted once using *Key_1* and in another trial using *Key_2* to produce the results in Fig. 28(b) and Fig. 28(c). Note that M, N are image dimensions, and the values of the control parameters and initial values are based on [25].

Table 7 Two different keys to test sensitivity

#	Permutation using (Duffing map)				Substitution using (Henon map)			
	X_i	Y_i	*a*	*b*	X_i	Y_i	*a*	*b*
Key_1	0: M-1	0: N-1	2	3	0: M-1	0: N-1	2	3
Key_2	0: M-1	0: N-1	9	9	0: M-1	0: N-1	2	3



(a) Original image (b) Encrypted image using *Key_1* (c) Encrypted image using *Key_2*

Fig. 28 Encrypted Peppers' image using two different keys

6.1.7 Key space analysis

One of the most important requirements for the security of any encryption system is that the key space should be large enough to resist brute-force attacks. For the proposed system, the encryption system depends on the key generated by the Duffing map and Henon map. For each map, there are two initial value vectors with two control parameters. Table 8 presents a calculation study of the key space used in the proposed single-round work.

Table 8 Key space calculation

Size / Variable	128×128		256×256		512×512		1024×1024	
Phase	Permutation	Substitution	Permutation	Substitution	Permutation	Substitution	Permutation	Substitution
x	14-bit	14-bit	16-bit	16-bit	18-bit	18-bit	20-bit	20-bit
y	14-bit	14-bit	16-bit	16-bit	18-bit	18-bit	20-bit	20-bit
a	4-bit	2-bit	4-bit	2-bit	4-bit	2-bit	4-bit	2-bit
b	4-bit	2-bit	4-bit	2-bit	4-bit	2-bit	4-bit	2-bit
One-round	$2^{36\text{-bit}}$	$2^{32\text{-bit}}$	$2^{40\text{-bit}}$	$2^{36\text{-bit}}$	$2^{44\text{-bit}}$	$2^{40\text{-bit}}$	$2^{48\text{-bit}}$	$2^{44\text{-bit}}$

6.1.8 NIST STS Randomness test

To test the randomness of the keys used in the proposed system for permutation and substitution, fifteen randomness tests from the NIST STS are employed. This test can be performed directly by downloading it from their website, or it can be performed separately. The result of each test is represented as a p-value. For each test to pass, its corresponding p-value must be greater than or equal to a specific amount called alpha (a). The common value of alpha is 0.01 [25]. The resulting p-value of the NIST STS test is shown in Table 9.

Table 9 P-value of NIST STS test for Key_1 and Key_2

#	Test Name	Key _1	Pass/Fail	Key_2	Pass/Fail
1	Frequency Test	0.021413	Pass	0.032578	Pass
2	Block Frequency Test	0.019918	Pass	0.019568	Pass
3	Runs Test	0.054146	Pass	0.042568	Pass
4	Longest Run Test	0.070485	Pass	0.076584	Pass
5	Binary Matrix Rank Test	0.091413	Pass	0.020415	Pass
6	DFT Test	0.035174	Pass	0.060074	Pass
7	Non-overlapping Template	0.065241	Pass	0.019000	Pass
8	Overlapping Template	0.500000	Pass	0.510200	Pass
9	Universal Test	0.009245	Fail	0.008546	Fail
10	Linear Complexity	0.500000	Pass	0.600102	Pass
11	Serial Test	0.500000	Pass	0.029358	Pass
12	Approximate Entropy	0.057694	Pass	0.064791	Pass
13	Cumulative Sums	0.032786	Pass	0.046821	Pass
14	Random Excursions	0.007146	Fail	0.003001	Fail
15	Random Excursions Variant	0.072325	Pass	0.083102	Pass

6.2. Hardware Processing Time Calculation

The second category of evaluation involves measuring the processing time of the proposed hardware design and comparing it with the execution time of a software-based cryptographic system. Fig. 29-30 illustrates the total processing times for both implementations across various test images, providing a comprehensive view of performance differences. The software execution times for the permutation and substitution operations were obtained using MATLAB 2017a/Simulink scripts, which ran on a personal laptop equipped with an Intel Core i7-1255U processor (1.70 GHz) and 16 GB of RAM. In contrast, the hardware processing time, which accounts for FPGA-specific timing parameters, can be estimated as:

$$\mathrm{T}_{\mathrm{total}} = \left(\mathrm{WR}_{\mathrm{cycles}} + \mathrm{RD}_{\mathrm{cycles}}\right) \times T_{clk} \tag{12}$$

where $\mathrm{T}_{\mathrm{total}}$ represents the total processing time, $\mathrm{WR}_{\mathrm{cycles}}$ denotes the number of clock cycles required for the write cycle, $\mathrm{RD}_{\mathrm{cycles}}$ is the number of clock cycles required for the read cycle, and $\mathrm{T}_{\mathrm{clk}}$ is the clock period measured in (nanoseconds).

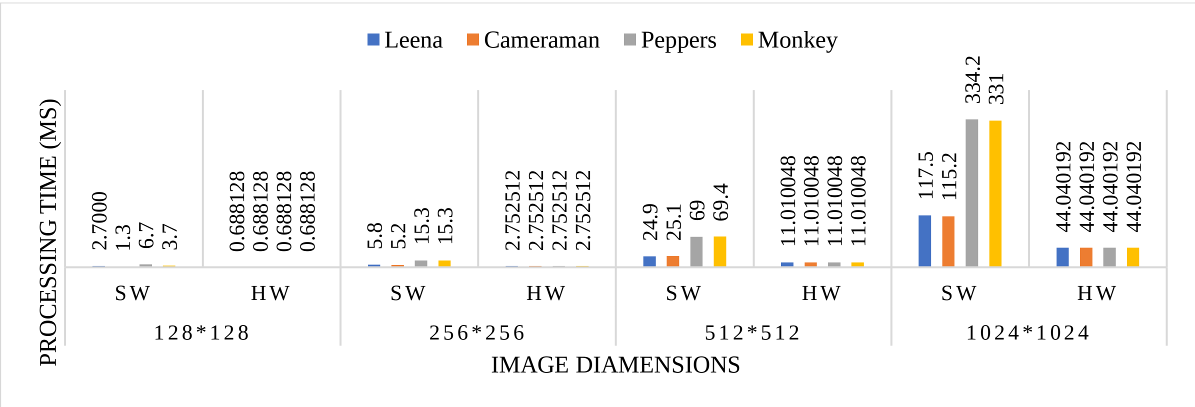


Fig. 29 Software vs. hardware processing time of the permutation/ substitution encryption system

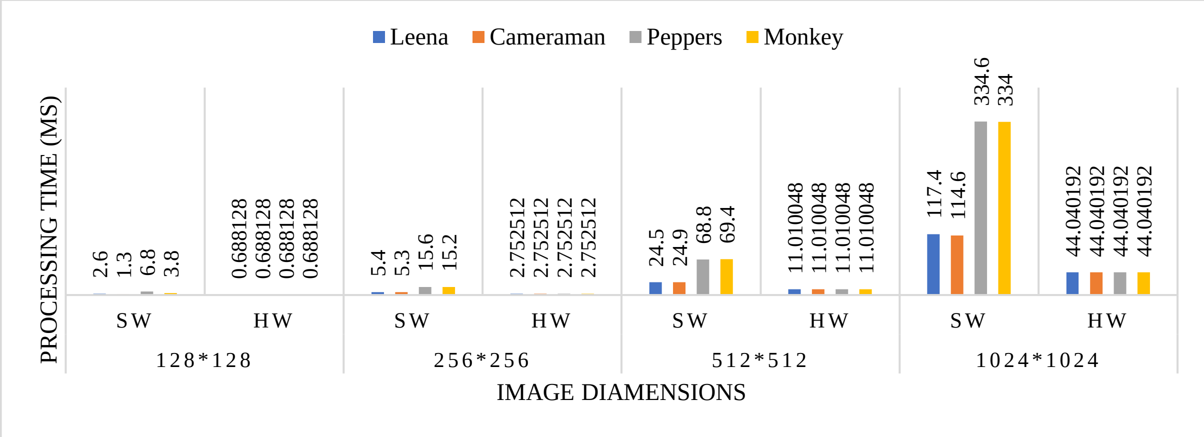


Fig. 30 Software vs. hardware processing time of the permutation/ substitution decryption system

The system's clock period was iteratively adjusted to achieve an acceptable Worst Negative Slack (WNS) value, ensuring that no timing violations or data loss occur. A positive WNS indicates that the clock period is longer than necessary, potentially resulting in underutilized timing margins and reduced performance. Conversely, a negative WNS implies that the clock period is insufficient for processing data within a single clock cycle, leading to potential timing failures [8, 20]. Table 10 presents the optimized clock periods for the proposed cryptographic systems across different image types and sizes.

Table 10 Value of clock period and WNS

Image / Size	128 x 128		256 x 256		512 x 512		1024 x 1024	
	Grayscale	RGB	Grayscale	RGB	Grayscale	RGB	Grayscale	RGB
Clock period (ns)	20	20	21	21	17	17	20	21
WNS (ns)	0.614	0.692	0.960	0.843	0.581	0.429	0.086	0.215

One of the most important aspects of hardware processing is the throughput. Throughput describes the number of bits processed per second [26]. Throughput can be calculated by:

$$\text{Throughput} = f_{\max} \times 8 \tag{13}$$

where

$$f_{\max} = \frac{1000}{\textit{Actual clock period}}\text{, and actual clock period = target clock period} - \text{WNS} \tag{14}$$

The throughput describes the amount of data that the system can process in a specific amount of time. A higher throughput means a higher ability to encrypt the digital image in a specific time period. Table 11 presents the detailed arguments required to calculate the throughput in bits per second (bps) and Megapixels per second (Mpix/s), depending on the actual clock period used.

Table 11 Throughput for the proposed Design

Image Size	Image Color type	Target clock Period (ns)	WNS (ns)	Actual clock Period (ns)	Fmax (MHz)	Pixels/Image	Throughput bit/s	Throughput (Mpix/s)
128 x 128	Grayscale	20	0.614	19.386	51.58	16,384	412,640,000	51.58
	RGB	20	0.692	19.308	51.79	16,384	414,320,000	51.79
256 x 256	Grayscale	21	0.960	20.040	49.90	65,536	399,200,000	49.90
	RGB	21	0.843	20.157	49.61	65,536	396,880,000	49.61
512 x 512	Grayscale	17	0.581	16.419	60.90	262,144	487,200,000	60.90
	RGB	17	0.429	16.571	60.35	262,144	482,800,000	60.35
1024 x 1024	Grayscale	20	0.086	19.914	50.22	1,048,576	401,760,000	50.22
	RGB	21	0.215	20.785	48.11	1,048,576	384,880,000	48.11

Table 12 shows a comparison study between the throughput of the proposed hardware design and the throughput of other existing designs. For an existing design that depends on multiple chaotic maps, the proposed hardware cryptographic system advances the throughput by approximately 1.7 times when testing the two systems with 256 × 256 grayscale and RGB images.

Table 12 Comparison analysis for throughput with other designs

Image size	Color format	Proposed design (Mpix/s)	[26]
256 × 256	Grayscale	49.90	-
	RGB	49.61	29.171

7. Discussion

As discussed in the previous sections, the proposed hardware design demonstrates a shorter processing time compared to its software-based counterpart. This performance improvement is effectively captured using the Speedup Factor, which is computed as:

$$\text{Speedup Factor} = \frac{\text{processing time of software program}}{\text{processing time of hardware design}} \tag{15}$$

Fig. 31 shows the speed-up factor between the proposed hardware cryptographic system and the software program for the same system. It shows that the hardware implementation processing time is advanced by 3.92 for 128×128 image size, 2.11 for 256×256 image size, 2.26 for 512×512 image size, and 2.67 for 1024×1024 image size.

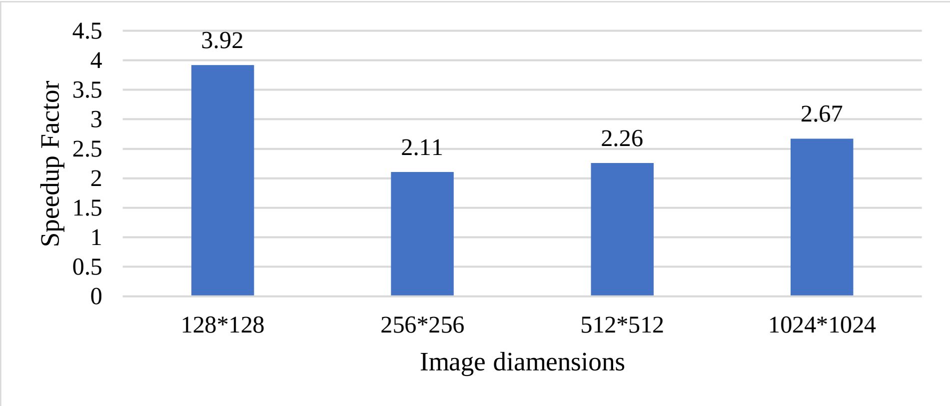


Fig. 31 Speedup factor of the proposed system

It is observed that the hardware acceleration achieved for grayscale images is identical to that of RGB images of the same dimensions. This is due to the parallel processing of the red, green, and blue channels in the RGB implementation, which allows the system to maintain consistent performance across image types. Once the system was designed, synthesized, and implemented, the overall FPGA resource utilization was recorded and is summarized in Table 13.

Table 13 FPGA resources utilization

Image Size	Color Format	LUT	FF	BRAM	DSP	IO	BUFG
128 x 128	Grayscale	2025 (1.5%)	2724 (1%)	4 (1%)	13 (1.7%)	17 (4%)	1 (3%)
	RGB	2067 (1.5%)	2728 (1%)	12 (3.2%)	13 (1.7%)	49 (12%)	1 (3%)
256 x 256	Grayscale	2012 (1.5%)	2723 (1%)	16 (4.3%)	13 (1.7%)	17 (4%)	1 (3%)
	RGB	2043 (1.5%)	2723 (1%)	48 (13%)	13 (1.7%)	49 (12%)	1 (3%)
512 x 512	Grayscale	2509 (1.8%)	3589 (1%)	64 (17.5%)	22 (2.9%)	17 (4%)	1 (3%)
	RGB	2620 (1.9%)	3657 (1%)	192 (52%)	22 (2.9%)	49 (12%)	1 (3%)
1024 x 1024	Grayscale	2086 (1.5%)	2858 (1%)	64 (17.5%)	13 (1.7%)	17 (4%)	1 (3%)
	RGB	2114 (1.5%)	2858 (1%)	192 (52%)	13 (1.7%)	49 (12%)	1 (3%)
Available		133800	267600	365	740	400	32

To compare our proposed hardware design with existing image encryption systems, the proposed hardware cryptographic system focused on the acceleration considerations by measuring the processing time in seconds. The comparison results are presented in Table 14 with references [27].

Table 14 Comparison study for hardware acceleration in (ns) with existing studies

Image Size	Proposed design	[27]
256×256	0.002752512	0.00656

8. Multi-Round Cryptography for Grayscale Images

To improve the security of the cryptographic system for digital images, the permutation-substitution encryption processes will be iteratively performed for 5 rounds. Each round contains different keys for both permutation and substitution [28]. The output of the first round will be fed loop again for the other four rounds. In the fifth round, the final encrypted pixels will be out from the FPGA through the output port (Gateway out block).

As the design will be iterated five rounds, five keys are needed for permutation and another five keys are needed for substitution. Each of them is completely different. For the permutation, five keys will be generated using the Duffing chaotic map, but with different settings. By controlling the initial values and the control parameters of each map. The details of each key are illustrated in Table 15 and shown in Fig. 32.

Table 15 Key Settings for Permutation Iterations

Keys	a	b	x_n	y_n
Iteration_1	9	9	0: N-1	0:M-1
Iteration_2	8	9	0: N-1	0:M-1
Iteration_3	7	9	0: N-1	0:M-1
Iteration_4	6	9	0: N-1	0:M-1
Iteration_5	5	9	0: N-1	0:M-1

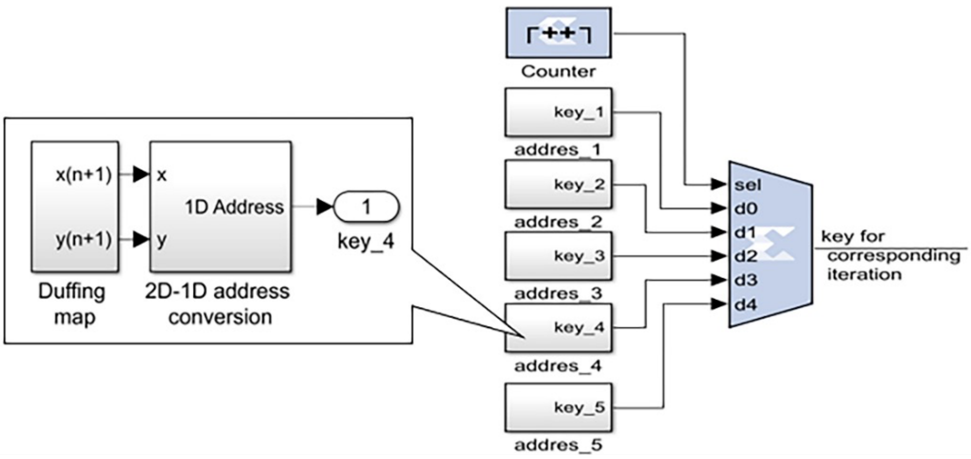


Fig. 32 Iteration keys for permutation

In the first iteration, the selection line selects d0 of the multiplexer to pass the first key to be used to permute the pixels' locations in the permutation process. When the first iteration is complete, the second key will be used in the second iteration, and so on until reaching the fifth iteration.

Also, for the substitution process, five different keys are used for each round. Five Henon chaotic maps are used with different settings; each produces a specific key. Then, for each round, the multiplexer will choose one to perform the substitution based on this key until completing the five rounds of the proposed cryptographic system, illustrated in Table 16 and shown in Fig. 33.

Table 16 Key settings for substitution iterations

Keys	*a*	*b*	x_n	y_n
Iteration_1	2	3	0: N-1	0:M-1
Iteration_2	2	4	0: N-1	1:M
Iteration_3	2	5	0: N-1	2:M+1
Iteration_4	2	6	0: N-1	3:M+2
Iteration_5	2	7	0: N-1	4:M+3

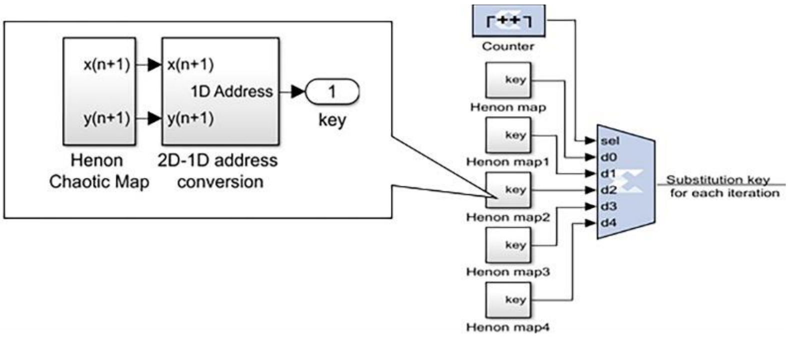


Fig. 33 Iteration keys for substitution

The same counter that is used for the permutation process iterations is used as a selection line to choose the corresponding address for substitution to achieve better synchronization at each round. Fig. 34 illustrates the substitution process during each round.

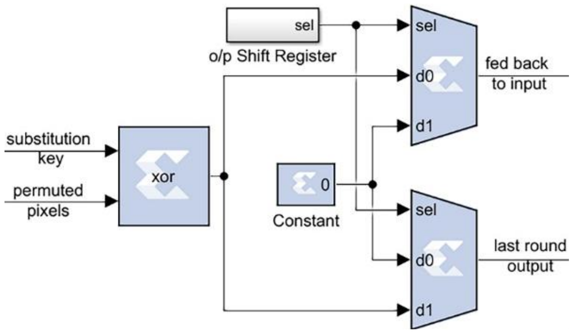


Fig. 34 Multi-round substitution process

The output of the multi-round substitution process is controlled by a 4-bit shift register that contains the sequence [0 0 0 0 1]. The selection signal of the multiplexer changed from one state to another after completing the entire round (total time for permutation-substitution). Fig. 35 shows the complete design of a multi-round encryption system for digital grayscale images.

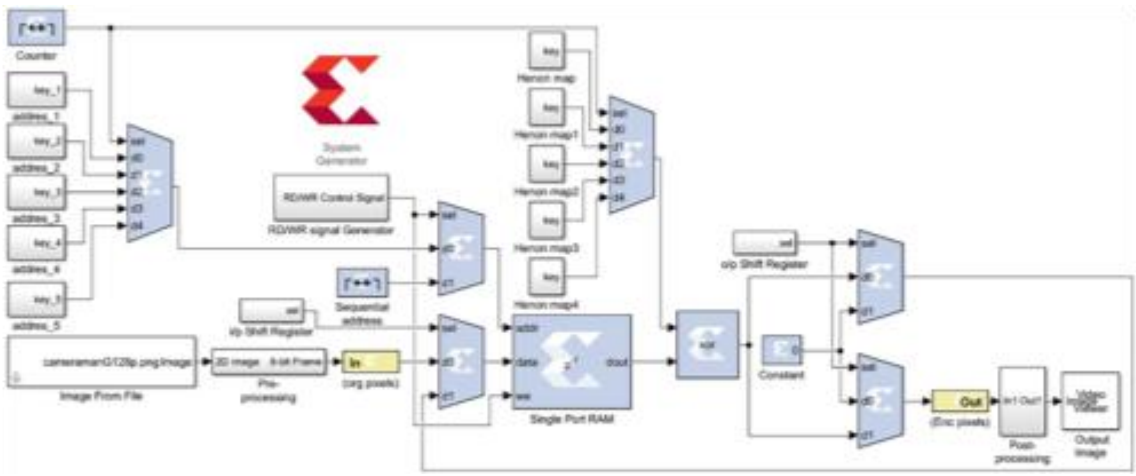


Fig. 35 XSG hardware design of a multi-round encryption system for grayscale images

The encryption system starts with the permutation process in the first round. The input shift register (which contains the sequence [0 1 1 1 1]) will produce a logic zero as a selection signal for the input multiplexer, allowing the original image pixels to be written into the single-port RAM sequentially during the write cycle of the first round. At the same time, the counter produces 0, which causes the selection of the multiplexer to select the first key for use in the first round. So, the key_1 will be used to read pixels out of the RAM chaotically. On its way to the output, the substitution process is performed on the permuted pixels using the first key of the substitution, which is selected using the same counter value that selects the first key for the permutation. The first-round encrypted pixels of the digital image will be fed back to the input, which is caused by the output shift register that produces a logic zero.

The second round starts by passing the feedback pixels (outs from the first round) into the input to perform another permutation-substitution round. In the permutation process, the feedback pixels pass through the multiplexer (in the second round, the input shift register produces logic 1) to be written into the RAM sequentially, then read chaotically using key_2 of the permutation key, which is selected by *sel* of the multiplexer that is driven by the counter value. The output of the single-port RAM will be XORed with key_2 of the substitution keys to produce the encrypted form of round 2. So on until reaching round 5 (the last round in the proposed system), the pixels that were permuted using key_5, then XORed with key_5 of the substitution process, will pass through the multiplexer to the output port to perform the final encrypted digital image. Table 17 shows the detailed settings of the encryption.

Table 17 Detailed settings for a multi-round encryption system

Iteration no.	Counter	i/p register	Permutation key (Duffing map)	Substitution key (Henon map)	o/p register
1	1	0	Key_1	Key_1	0
2	2	1	Key_2	Key_2	0
3	3	1	Key_3	Key_3	0
4	4	1	Key_4	Key_4	0
5	5	1	Key_5	Key_5	1

On the other hand, the proposed hardware decryption system will be a mirror of the encryption system. The decryption process starts with substitution, then permutation, as shown in Fig. 36. The decryption process will be iterative for five rounds to restore both pixels' intensities and the original locations of the original input digital image.

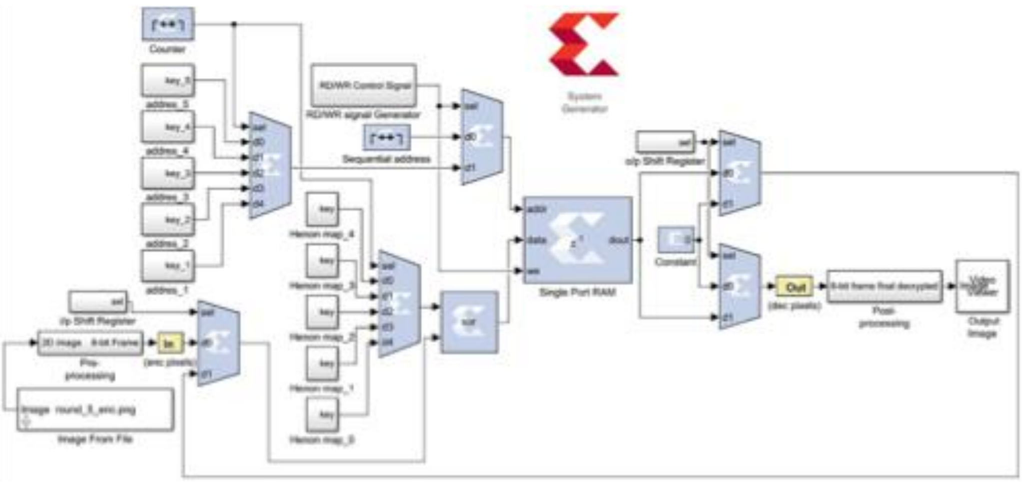


Fig. 36 XSG hardware design of a multi-round decryption system for grayscale images

In the proposed hardware design for image decryption, the keys that are used in the encryption rounds will be reversed. For round 1 in the decryption system, key_4 (generated from the Duffing chaotic map) is used for permutation, and the fourth key (produced by the Henon chaotic map) is used for substitution. In the second round, key_3 is used, and so on until reaching the last round, which uses key_0.

9. Multi-Round Cryptography for RGB Images

For an RGB image, the multi-round cryptographic system can be created by expanding the design of the single-channel grayscale image encryption/decryption system. Figs. 37-38 shows the complete design for encryption and decryption of RGB images.

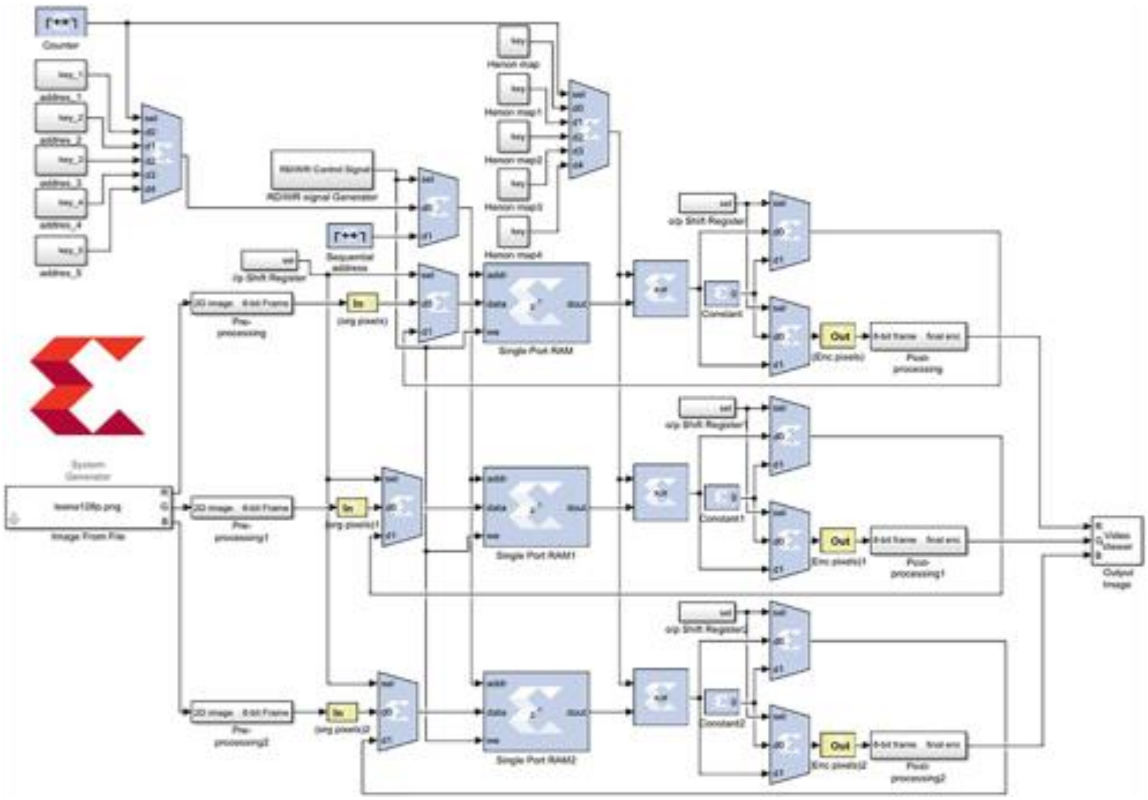


Fig. 37 XSG hardware design of a multi-round encryption system for RGB images

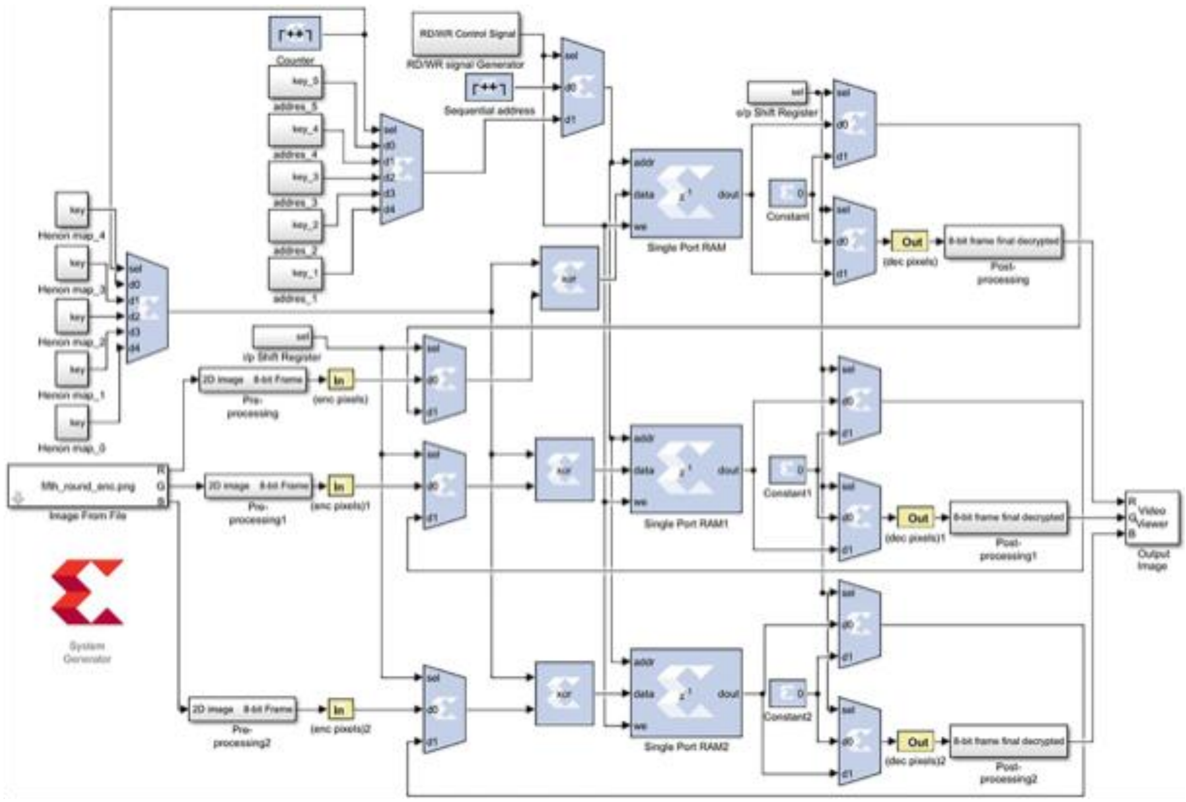
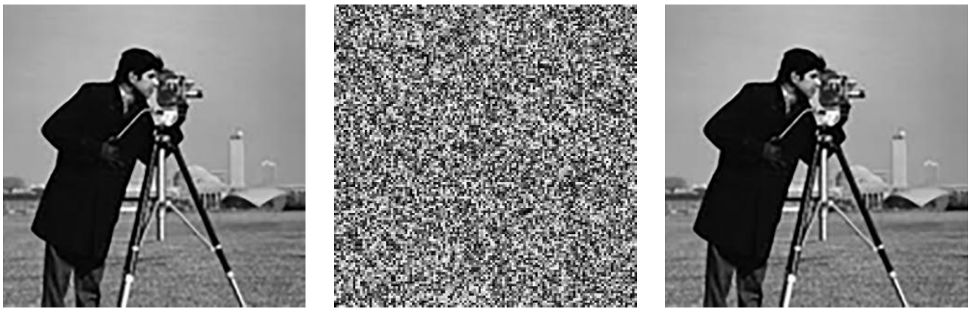


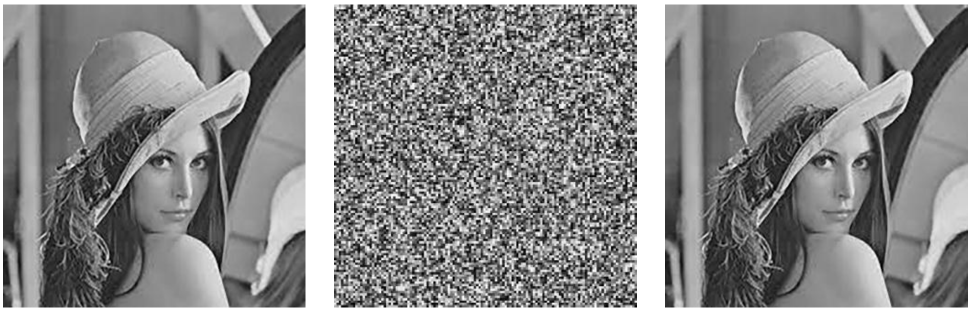
Fig. 38 XSG hardware design of a multi-round decryption system for RGB images

10. Results

Figures 39-40 show the visual representation of the results of the proposed five-round permutation-substitution cryptographic system. Fig. 39 demonstrates the cryptographic results of grayscale images, using the standard 'Cameraman' and 'Leena' test images. The encrypted digital image result (Figs. 39b, 39e) is visually indistinguishable from random noise, asserting the effective dissipation of pixel value correlations and the strength of the permutation-substitution processes. Crucially, the decryption process ideally recovers the original image (Figs. 39c, 39f) without any loss of information, validating the correctness of the algorithm. Similarly, Fig. 40 shows that the system successfully extends these properties to RGB color images, producing a secure ciphertext and a flawless reconstruction.

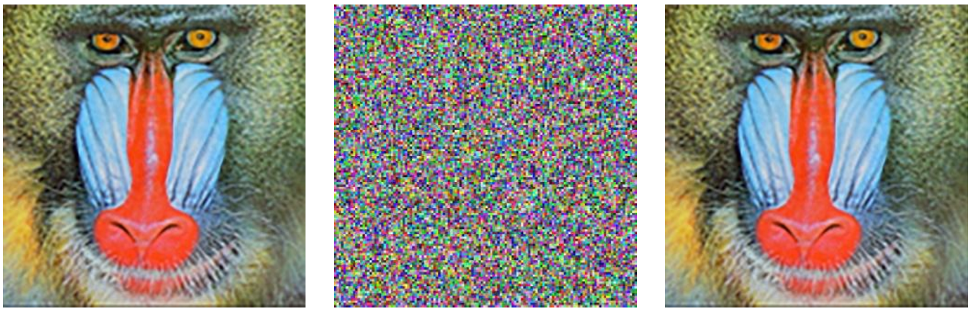


(a) Original Cameraman (b) Encrypted Cameraman (c) Recovered Cameraman

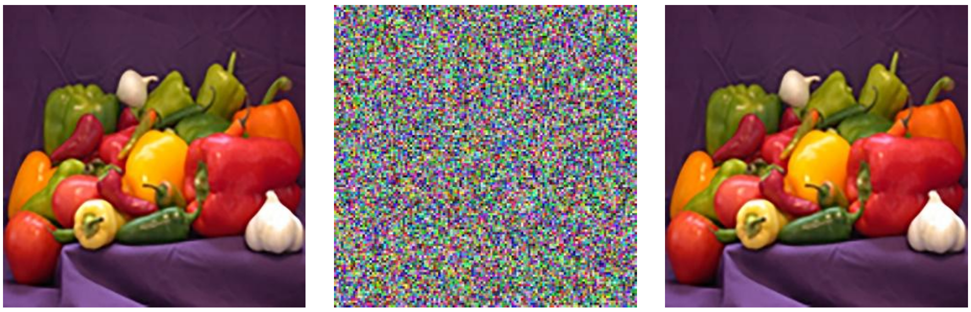


(d) Original Leena (e) Encrypted Leena (f) Recovered Leena

Fig. 39 Grayscale image testing



(a) Original Monkey (b) Encrypted Monkey (c) Recovered Monkey



(d) Original Peppers (e) Encrypted Peppers (f) Recovered Peppers

Fig. 40 RGB image testing

11. System Evaluation

To evaluate the proposed hardware multi-round cryptographic system, two types of metrics are used to test the security robustness and processing speed. To confirm the security of the proposed hardware cryptographic system, seven security tests are used, while the main goal is the speed advancement of the proposed hardware implementation over a software program, which is measured by comparing the processing time of both systems.

11.1. Security Metrics

To evaluate the security robustness of the multi-round cryptographic system, the following metrics have been used:

(1) MSE: The mean squared error between the original and recovered image from the multi-round cryptographic system is zero for all tested standards.

(2) Histogram Analysis: The histogram analysis tool has been used to illustrate the distribution of pixel intensities before and after encryption. Fig. 41 illustrates the details of the histograms for the tested grayscale and RGB images.

The histogram of the encrypted image shows a flat distribution of pixels, which means a uniform distribution of pixel values.

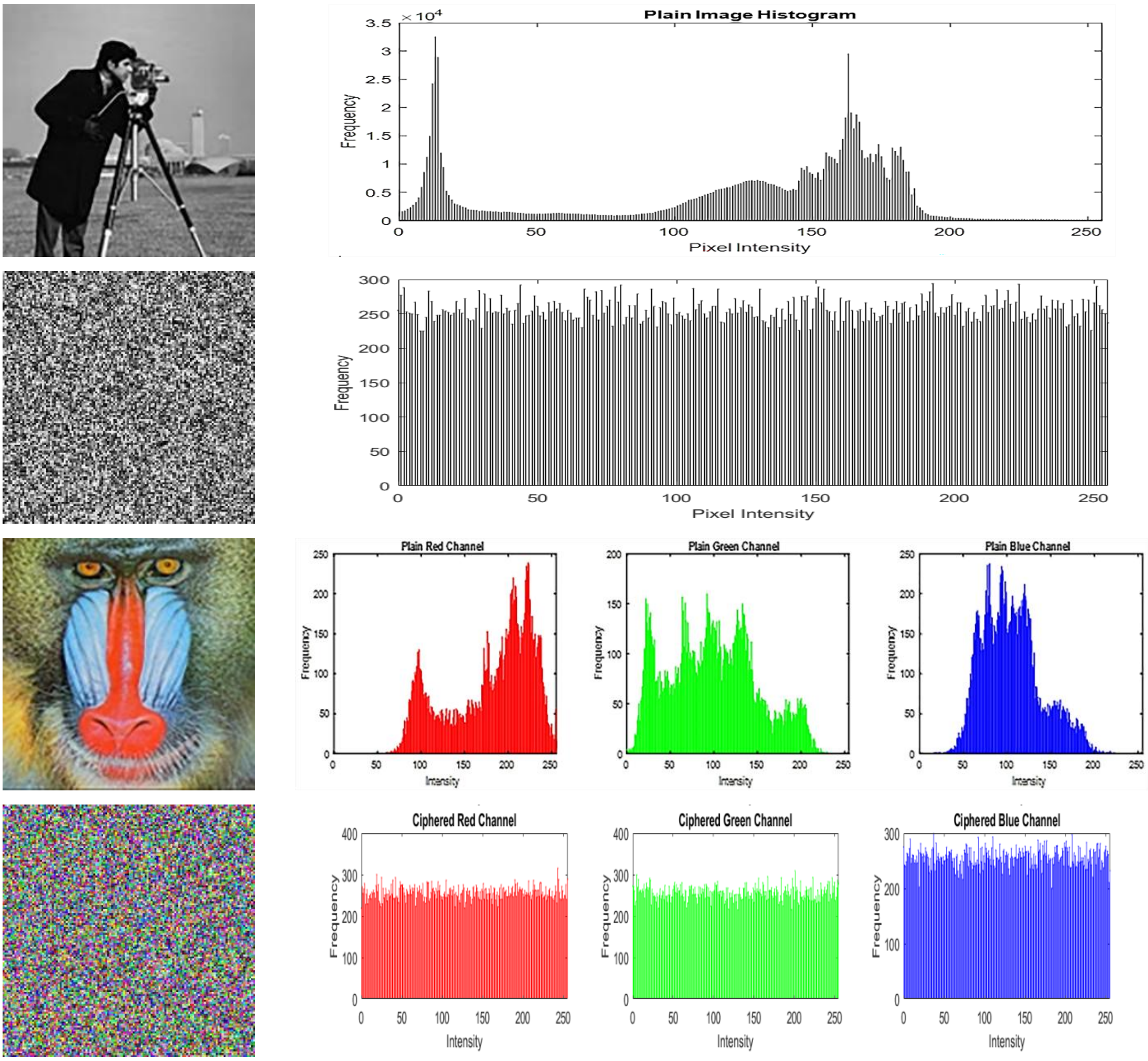


Fig. 41 Histogram of multi-round cryptographic system results

(3) CCA: To test the correlation between adjacent pixels in the encrypted image CCA test has been used. Tables 18-19 show the results of the CCA analysis of grayscale and RGB images.

Table 18 CCA analysis for grayscale images

Image		Cameraman			Leena		
		H	V	D	H	V	D
128x128	Plain	0.9502	0.9510	0.9510	0.9586	0.9583	0.9583
	Encrypted	0.00023	0.00062	0.00034	0.00044	0.00039	0.00031
256x256	Plain	0.9486	0.9506	0.9506	0.9941	0.9941	0.9941
	Encrypted	0.00013	0.00019	0.00018	0.00020	0.00023	0.00022
512x512	Plain	0.9969	0.9970	0.9970	0.9993	0.9993	0.9993
	Encrypted	0.00015	0.00027	0.00029	0.00031	0.00038	0.00037
1024x1024	Plain	0.9998	0.9998	0.9998	0.9999	0.9999	0.9999
	Encrypted	0.00025	0.00028	0.00028	0.00031	0.00033	0.00030

Table 19 CCA analysis for RGB images

Image		128 × 128			256 × 256			512 × 512			1024 × 1024		
		R	G	B	R	G	B	R	G	B	R	G	B
Peppers Plain	H	0.7909	0.9387	0.9240	0.8420	0.9653	0.9437	0.9567	0.9882	0.9869	0.9968	0.9993	0.9991
	V	0.7890	0.9346	0.9186	0.8401	0.9643	0.9420	0.9562	0.9880	0.9867	0.9967	0.9992	0.9991
	D	0.7890	0.9346	0.9186	0.8401	0.9643	0.9420	0.9562	0.9880	0.9867	0.9967	0.9992	0.9991
Peppers Encrypted	H	0.00070	0.00072	0.00072	-0.00069	0.00065	0.00061	0.00035	0.00032	0.00021	0.00012	0.00027	0.00049
	V	0.00068	0.00070	0.00072	-0.00096	0.00062	0.00060	0.00051	0.00030	0.00028	-0.00017	0.00025	-0.00045
	D	0.00068	0.00070	0.00071	0.00094	0.00060	0.00060	0.00051	0.00029	0.00032	-0.00017	0.00024	0.00046
Monkey Plain	H	0.9390	0.9044	0.9249	0.8989	0.8552	0.8565	0.8902	0.8600	0.8441	0.9911	0.9883	0.9882
	V	0.9383	0.9041	0.9249	0.8977	0.8542	0.8565	0.8885	0.8580	0.8433	0.9911	0.9882	0.9881
	D	0.9383	0.9041	0.9249	0.8977	0.8542	0.8565	0.8885	0.8580	0.8433	0.9911	0.9882	0.9881
Monkey Encrypted	H	0.00025	-0.00031	0.00021	0.00030	0.00039	0.00034	0.00045	0.00032	0.00017	0.00043	0.00026	-0.00018
	V	0.00028	0.00037	-0.00020	-0.00032	-0.00038	-0.00034	0.00042	0.00033	0.00019	0.00040	0.00030	-0.00012
	D	0.00029	-0.00038	-0.00022	0.00030	0.00038	0.00034	0.00042	0.00035	0.00020	0.00047	0.00029	-0.00012

Table 20 shows the comparison between the proposed multi-round cryptographic hardware design and the existing works. For RGB 256 × 256 peppers, monkey, and grayscale Leena digital images, the proposed hardware cryptographic system achieves lower vertical, horizontal, and diagonal correlation between adjacent pixels.

Table 20 CCA analysis comparison

Image name		Proposed multi-round design			Proposed single-round design			[28]		
		H	V	D	H	V	D	H	V	D
Peppers	R	-0.00069	0.00065	0.00061	-0.0192	0.0184	-0.0184	0.0045	0.0101	0.0104
	G	-0.00096	0.00062	0.00060	-0.0243	-0.0254	0.0250	-0.0027	0.0017	0.0178
	B	0.00094	0.00060	0.00060	0.0743	-0.0698	0.0695	0.0024	0.0042	0.0116
Monkey	R	0.00030	0.00039	0.00034	0.0381	0.0403	0.0033	-0.0110	0.0041	0.0085
	G	-0.00032	-0.00038	-0.00034	-0.0257	-0.0243	0.0245	-0.0070	0.0091	0.0150
	B	0.00030	0.00038	0.00034	0.0337	0.0320	-0.0409	0.0011	0.0022	0.0048
Leena		0.00044	0.00039	0.00031	0.0041	-0.0029	-0.0029	-0.0042	0.0022	0.0090

(4) Information Entropy: Table 21 shows the entropy of the proposed hardware multi-round cryptographic system, with the previous single-round cryptographic system and existing works.

Table 21 Information entropy analysis of multi-round with single-round, and existing works

Name	Channels	Proposed multi-round design	Proposed single-round design	[28]
Peppers	R	7.9999	7.9993	7.9921
	G	7.9997	7.9972	7.9916
	B	7.9999	7.9920	7.9929

(5) NPCR and UACI: To test how random the proposed multi-round cryptographic system has achieved on the image pixels, NPCR and UACI are used. The results are shown in Table 22.

Table 22 NPCR and UACI analysis

Type	Name		128×128		256×256		512×512		1024×1024	
			NPCR	UACI	NPCR	UACI	NPCR	UACI	NPCR	UACI
Grayscale Images	Leena		99.9986%	33.3654	99.9988%	33.4261	99.9982%	33.4521	99.9986%	33.5632
	Cameraman		99.9979%	33.2348	99.9996%	33.4251	99.9983%	33.4472	99.9992%	33.6025
RGB images	Peppers	R	99.9997%	33.2248	99.9998%	33.3255	99.9987%	33.4029	99.9995%	33.4502
		G	99.9991%	33.3145	99.9983%	33.4452	99.9989%	33.4366	99.9988%	33.4417
		B	99.9989%	33.2487	99.9989%	33.4632	99.9984%	33.4624	99.9985%	33.4789
	Monkey	R	99.9991%	33.3624	99.9997%	33.2516	99.9991%	33.3265	99.9983%	33.4217
		G	99.9985%	33.2145	99.9993%	33.3985	99.9999%	33.4786	99.9991%	33.4666
		B	99.9972%	33.3211	99.9991%	33.4125	99.9996%	33.4656	99.9990%	33.4343

Table 23 shows the NPCR and UACI for the proposed multi-round and single-round cryptographic design with existing works. From the table below, it is clear that there is an improvement in the NPCR and UACI values, provided that iterating the permutation-substitution processes provides robustness for the system security.

Table 23 NPCR and UACI of single and multi-round with existing works

Name	Proposed multi-round design		Proposed single-round design		[28]	
	NPCR	UACI	NPCR	UACI	NPCR	UACI
Cameraman	99.9988%	33.4261	99.6362%	33.4295	99.6017%	33.4609
Leena	99.9996%	33.4251	99.6777%	33.3352	99.6109%	33.4233

(6) Key Space Analysis: For the proposed multi-round hardware cryptographic system, the key space is expanded five times that of a single round. Table 24 shows the details for calculating the key space for the proposed hardware multi-round cryptographic system.

Table 24 Key space calculation

Size / Variable	128×128		256×256		512×512		1024×1024	
	Permutation	Substitution	Permutation	Substitution	Permutation	Substitution	Permutation	Substitution
x	14-bit	14-bit	16-bit	16-bit	18-bit	18-bit	20-bit	20-bit
y	14-bit	14-bit	16-bit	16-bit	18-bit	18-bit	20-bit	20-bit
a	4-bit	2-bit	4-bit	2-bit	4-bit	2-bit	4-bit	2-bit
b	4-bit	2-bit	4-bit	2-bit	4-bit	2-bit	4-bit	2-bit
One-round	$2^{36\text{-bit}}$	$2^{32\text{-bit}}$	$2^{40\text{-bit}}$	$2^{36\text{-bit}}$	$2^{44\text{-bit}}$	$2^{40\text{-bit}}$	$2^{48\text{-bit}}$	$2^{44\text{-bit}}$
5-round	$2^{180\text{-bit}}$	$2^{160\text{-bit}}$	$2^{200\text{-bit}}$	$2^{180\text{-bit}}$	$2^{220\text{-bit}}$	$2^{200\text{-bit}}$	$2^{240\text{-bit}}$	$2^{220\text{-bit}}$

A comparison study was made between existing works and the proposed hardware cryptographic system, as shown in Table 25. As the key space of the proposed hardware cryptographic system increases, the system's robustness increases to face brute force attacks, which indirectly contributes to robustness against statistical attacks, but it does not automatically make a system robust against statistical attacks.

Table 25 key space comparison

Key space	Proposed	[24]
	2^{180}	2^{100}

(7) NIST Randomness Test: To apply the NIST Test on the collection of keys that are used in the multi-round permutation and substitution, the overall used keys are either concatenated or XORed to test the randomness of the complete cryptographic system. NIST results are shown in Table 26.

Table 26 NIST test

#	Test Name	Key _concatenated	Pass/Fail	Key_XORed	Pass/Fail
1	Frequency Test	0.025641	Pass	0.500000	Pass
2	Block Frequency Test	0.034859	Pass	0.060144	Pass
3	Runs Test	0.071364	Pass	0.035462	Pass
4	Longest Run Test	0.036541	Pass	0.023555	Pass
5	Binary Matrix Rank Test	0.012935	Pass	0.086214	Pass
6	DFT Test	0.062541	Pass	0.033658	Pass
7	Non-overlapping Template	0.043251	Pass	0.029511	Pass
8	Overlapping Template	0.535241	Pass	0.500062	Pass
9	Universal Test	0.092145	Pass	0.052201	Pass
10	Linear Complexity	0.452183	Pass	0.422071	Pass
11	Serial Test	0.535867	Pass	0.033200	Pass
12	Approximate Entropy	0.072541	Pass	0.095646	Pass
13	Cumulative Sums	0.063489	Pass	0.782114	Pass
14	Random Excursions	0.021485	Pass	0.022997	Pass
15	Random Excursions Variant	0.036524	Pass	0.033021	Pass

The randomness quality of the keys used in the proposed hardware cryptographic system was validated using the NIST statistical test suite. The composite keys, formed through either concatenation or an XOR operation, successfully passed all 15 tests. As detailed in Table 26, all computed p-values exceeded the threshold of alpha value 0.01, asserting the statistical indistinguishability of the generated keys.

11.2 Hardware Processing Time

The hardware vs. software processing time of the encryption and decryption multi-round system is illustrated in Figs. 42-43. The clock and WNS of the proposed multi-round cryptographic system used to calculate the actual hardware processing time are illustrated in Table 27.

Table 27 Clock period with WNS

Image / Size	128 x 128		256 x 256		512 x 512		1024 x 1024	
	Grayscale	RGB	Grayscale	RGB	Grayscale	RGB	Grayscale	RGB
Clock period (ns)	20	20	21	21	17	17	20	21
WNS (ns)	0.614	0.692	0.960	0.843	0.581	0.429	0.086	0.215

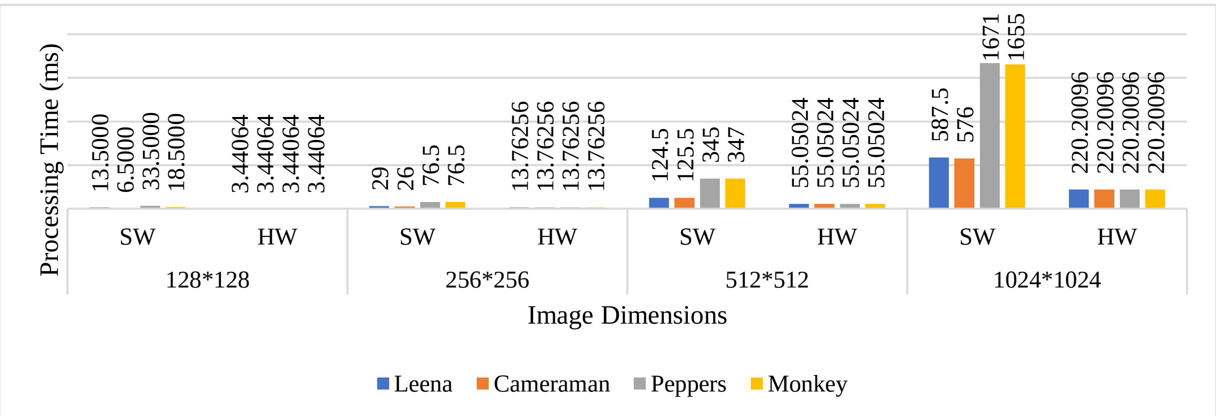


Fig. 42 Software vs. hardware processing time of a multi-round permutation-substitution encryption system

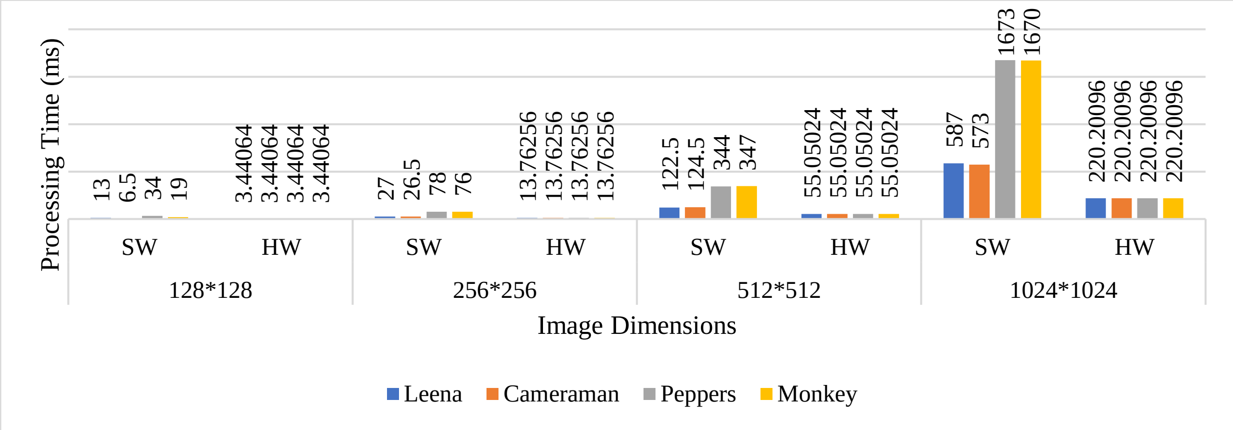


Fig. 43 Software vs. hardware processing time of the multi-round permutation-substitution decryption system

By analyzing the processing time of the multi-round cryptographic system compared to the single-round cryptographic system, it can be observed that the processing time becomes longer for both software and hardware. The processing times of both the software program and the proposed hardware cryptographic implementation are nearly extended to five times to produce the final encrypted image.

To show the advancement in the proposed hardware multi-round cryptographic system over the software version, the speedup factor has been calculated. The results are shown in Fig. 44. The multi-round cryptographic system achieves the throughput shown in Table 28.

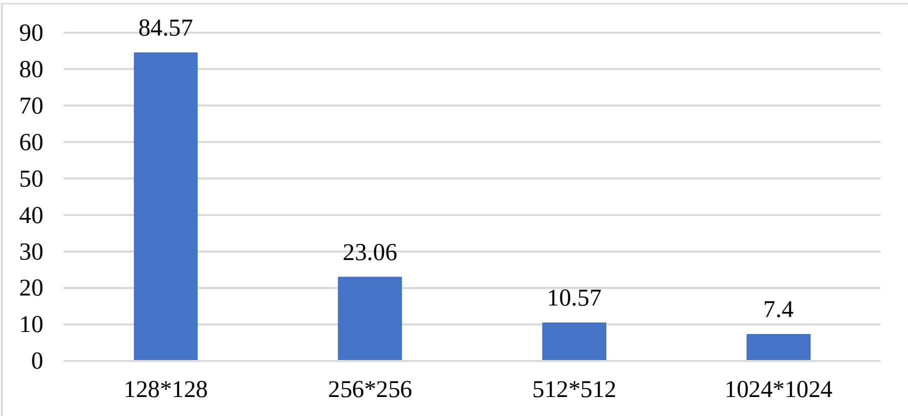


Fig. 44 Speedup factor of a multi-round cryptographic system

Table 28 Multi-round cryptographic system throughput

Image Size	Image Colour type	Target clock Period (ns)	WNS (ns)	Actual clock Period (ns)	Fmax (MHz)	Pixels/Image	Throughput bit/s
128 x 128	Grayscale	20	0.614	19.386	51.58	16,384	412,640,000
	RGB	20	0.692	19.308	51.79	16,384	414,320,000
256 x 256	Grayscale	21	0.960	20.040	49.90	65,536	399,200,000
	RGB	21	0.843	20.157	49.61	65,536	396,880,000
512 x 512	Grayscale	17	0.581	16.419	60.90	262,144	487,200,000
	RGB	17	0.429	16.571	60.35	262,144	482,800,000
1024 x 1024	Grayscale	20	0.086	19.914	50.22	1,048,576	401,760,000
	RGB	21	0.215	20.785	48.11	1,048,576	384,880,000

The FPGA resource utilization summary in Table 29 validates the hardware efficiency of the proposed cryptographic system implementation. The design is notably lightweight in terms of logic, which is represented by LUTs (<2.3%) and FFs (<1.5%), indicating an optimized data path. The primary resource consumption is attributed to Block RAM (BRAM), which scales linearly with image resolution and number of color channels, as expected for an image buffer. This predictable scaling and low overall usage of logic cells assert the architectural efficiency and suitability for embedded vision applications.

Table 29 Multi-round cryptographic system resource utilization

Image Size	Color Format	LUT	FF	BRAM	DSP	IO	BUFG
128 x 128	Grayscale	2271 (1.6%)	2727 (1%)	4 (1%)	13 (1.7%)	17 (4%)	1 (3%)
	RGB	2369 (1.7%)	2728 (1%)	12 (3.2%)	13 (1.7%)	49 (12%)	1 (3%)
256 x 256	Grayscale	2215 (1.6%)	2723 (1%)	16 (4.3%)	13 (1.7%)	17 (4%)	1 (3%)
	RGB	2444 (1.8%)	2723 (1%)	48 (13%)	13 (1.7%)	49 (12%)	1 (3%)
512 x 512	Grayscale	2708 (2.1%)	3588 (1%)	64 (17.5%)	22 (2.9%)	17 (4%)	1 (3%)
	RGB	3021 (2.2%)	3659 (1%)	192 (52%)	22 (2.9%)	49 (12%)	1 (3%)
1024 x 1024	Grayscale	2289 (1.7%)	2859 (1%)	64 (17.5%)	13 (1.7%)	17 (4%)	1 (3%)
	RGB	2319 (1.7%)	2859 (1%)	192 (52%)	13 (1.7%)	49 (12%)	1 (3%)
Available		133800	267600	365	740	400	32

12. Conclusion

This research successfully designed and implemented a new robust hardware design for a single- and 5-round permutation-substitution image cryptographic system, capable of securing both grayscale and RGB images of varying sizes. The proposed system leveraged the Duffing chaotic map for permutation and the Henon chaotic map for the substitution after setting their initial values and control parameters. The cryptographic system was realized using the Xilinx System Generator (XSG), enabling seamless integration of the Simulink design environment with the Xilinx Artix-7 FPGA platform. The key conclusion points can be summarized in four primary regions.

(1) High security robustness, which is achieved through a 5-round architecture that was validated through statistical metrics (MSE, histogram, CCA, entropy, NPCR, UACI) and successfully passed the NIST STS, asserting the system's resistance against statistical and differential attacks.

(2) Hardware acceleration asserted through Hardware co-simulation of the proposed design demonstrated significant performance gains, with acceleration factors (speed-up factors) ranging from 2.11 to 3.92 across different image sizes, underscoring the efficiency of the FPGA implementation over software programs.

(3) As a practical efficiency, the proposed hardware multi-round cryptographic system achieved high throughput with moderate FPGA resource utilization, asserting its feasibility for real-time cryptographic applications.

(4) The architectural effectiveness achieved by the integration of a single-port RAM with a custom control signal generator and chaotic map generators has been proven effective for creating a scalable and secure cryptographic pipelined architecture.

Future research will focus on extending the proposed architecture to support higher-resolution images, optimizing the design for low-power FPGA platforms, and exploring hybrid chaotic systems to further enhance encryption complexity and security performance.

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] W. Stallings, Cryptography and Network Security Principles and Practice, 5th ed., New York: Pearson Education, 2011.

[2] L. Tan and J. Jiang, Digital Signal Processing Fundamentals & Applications, 2nd ed., New York: Academic Press - Elsevier, 2013.

[3] S. Singh, R. Parida, and C. Pradhan, “Comparative Analysis of Image Encryption Using 2D and 3D Variations of Duffing Map,” Proceedings of International Conference on Communications and Signal Processing (ICCSP), pp. 751-754, 2018.

[4] Z. Gao, Z. Liu, and L.Wang, “An Image Encryption Algorithm Based on the Improved Sine-Tent Map,” Discrete Dynamics in Nature and Society, vol. 2021, no.1, article no. 9187619, pp. 1-16, 2021.

[5] Y. Abanda, A. Tiedeu, and G. Kom, “Image Encryption with Fusion of Two Maps,” Security of Communication Networks, vol. 2021, no. 1, article no. 6624890, 2021.

[6] Z. Feixiang, L. Mingzhe, W. Kun, and Z. Hong, “Color Image Encryption via Hénon-Zigzag Map and Chaotic Restricted Boltzmann Machine over Blockchain,” Optics & Laser Technologies, vol. 135, article no. 106610, 2021.

[7] W. Al-Mosawi, W. Wali, and M. Al-Ibadi, “Implementation of Chaotic System Using FPGA,” Proceedings of 6th Asia-Pacific Conference on Intelligent Robot Systems (ACIRS), pp. 83-88, 2021.

[8] W. Al-Mosawi, W. Wali, and M. Al-Ibadi, “Field-Programmable Gate Array Design of Image Encryption and Decryption Using Chua's Chaotic Masking,” International Journal of Electrical and Computer Engineering, vol. 12, no. 3, pp. 2414 -2424, 2022.

[9] T. Ali, and R. Ali, “A Novel Color Image Encryption Scheme Based on a New Dynamic Compound Chaotic Map and S-Box,” Multimedia Tools and Applications, vol. 81, article no. 15, pp. 20585-20609, 2022.

[10] W. Al-Mosawi, W. Wali, and M. Al-Ibadi, “New Artificial Neural Network Design for Chua Chaotic System Prediction Using FPGA Hardware Co-Simulation,” International Journal of Electrical and Computer Engineering, vol. 12, no. 2, pp. 1955-1964, 2021.

[11] C. Grimaldo, C. Marquez, E. Cuautle, and E. Cantón, “FPGA Implementation of a Chaotic Map with no Fixed Point,” Electronics, vol. 12, no. 2, article no. 444, pp. 1-13, 2023.

[12] W. Al-Mosawi, M. Al-Ibadi, and W. Wali, “Artificial Intelligence Techniques for Encrypting Images Based on the Chaotic System Implemented on a Field-Programmable Gate Array,” IAES International Journal of Artificial Intelligence, vol. 12, no. 1, pp. 347-356, 2023.

[13] L. Sabeeh and M. Al-Ibadi, “FPGA-Based Accelerator for Image Steganography,” Methods and Applications for Modeling and Simulation of Complex Systems (AsiaSim 2023), vol. 1911, pp. 1-12, 2023.

[14] A. Khudhair, A. Maolood, and E. Gbashi, “Symmetry Analysis in Construction Two Dynamic Lightweight S-Boxes Based on the 2D Tinkerbell Map and the 2D Duffing Map,” Symmetry, vol. 16, no. 7, article no. 872, 2024.

[15] J. Zhou, Y. Li, M. Wang, “Research on the Threshold Determination Method of the Duffing Chaotic System Based on Improved Permutation Entropy and Poincaré Mapping,” Entropy, vol. 25, article no. 1654, 2023.

[16] Agarwal, “A Review of Image Scrambling Technique Using Chaotic Maps,” International Journal of Engineering and Technology Innovation, vol. 8, no. 2, pp. 492-497, 2018.

[17] E. Tlelo-Cuautle, J. Rangel-Magdaleno, and L. de la Fraga, Engineering applications of FPGAs, 1st ed., Switzerland: Springer International Publishing, 2016.

[18] C. Zhang, S. Zhang, K. Liang, and Z. Chen, “Double Image Encryption Algorithm Based on Parallel Compressed Sensing and Chaotic System,” IEEE Access, vol. 12, pp. 54745-54757, 2024.

[19] S. Kanwal, S. Inam, M. Othman, A. Waqar, M. Ibrahim, F. Nawaz, & et. al, “An Effective Color Image Encryption Based on Henon Map, Tent Chaotic Map, and Orthogonal Matrices,” Sensors, vol. 22, article no. 4359, 2022.

[20] B. Issa et al., “Design of High Precision Radix-8 MAF Unit with Reduced Latency,” Proceedings of 2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), pp. 1-6, 2020.

[21] T. Hodson, T. Over, and S. Foks, “Mean Squared Error, Deconstructed,” Journal of Advances in Modeling Earth Systems, vol. 13, no. 12, 2021.

[22] R. Aboughalia and O. Alkishriwo, "Color Image Encryption Based on Chaotic Block Permutation and XOR Operation," Libyan International Conference on Electrical Engineering and Technologies (LICEET2018), Tripoli, Libya, 2018.

[23] Y. Wu, J. Noonan, and S. Agaian, "NPCR and UACI Randomness Tests for Image Encryption," Cyber Journals: Multidisciplinary Journals in Science and Technology, Journal of Selected Areas in Telecommunications (JSAT), pp. 31-38, 2011.

[24] K. Hosny, Y. Elnabawy, R. Salama and A. Elshewey, "Multiple Image Encryption Algorithm using Channel Randomization and Multiple Maps," Scientific Reports, vol. 14, no. 30597, pp. 238-246, 2024.

[25] H. Wen, Y. Lin, S. Kang, X. Zhang, and K. Zou, "Secure Image Encryption Algorithm Using Chaos-Based Block Permutation and Weighted Bit Planes Chain Diffusion," iScience, vol. 27, no. 1, no. 108610, 2024.

[26] Y. Khazaal, K. Zidan, and F. Hasan, "FPGA Hardware Co-Simulation of Image Encryption Using Hybrid LFSR and Chaos Maps," Intelligent Systems and Networks (ICISN2021), vol. 243, pp. 487-498, 2021.

[27] M. Yamni, A. Daoui, P. Plawik, O. Al-Farraj, and A. Abd Al-Latif, "A Hardware-Accelerated Approach to Chaotic Image Encryption: LTB Map and FPGA Implementation," IEEE Access, vol. 12, pp. 103921-103940, 2024.

[28] T. Anujaa, A. Ahamed, V. Baranwal, V. Thanikaiselvan, S. Subashanthini, C. Devi, et al., "A Lightweight Multi-Round Confusion-Diffusion Cryptosystem for Securing Images Using a Modified 5D Chaotic System," Scientific Reports, vol. 15, article no. 31986, 2025.