INTEGRATION, the VLSI journal 81 (2021) 1–16

Contents lists available at ScienceDirect

Integration

journal homepage: www.elsevier.com/locate/vlsi

Development of micro computer based mobile random number generator with an encryption application

Akif Akgul [a,*], Bilal Gurevin [a,b], Ihsan Pehlivan [a,b], Muhammed Yildiz [a,b], Mustafa C. Kutlu [a,b], Emre Guleryuz [a,b]

[a] Department of Computer Engineering, Faculty of Engineering, Hitit University, 19030, Corum, Turkey
[b] Department of Electrical and Electronics Engineering, Faculty of Technology, Sakarya University of Applied Sciences, 54187, Sakarya, Turkey

ARTICLE INFO

Keywords:
Chaos
Chaotic system
Random number generation
Encryption
Security analysis

ABSTRACT

In this study, an equation is derived by changing the parameter values of a chaotic system in the literature and analyzing its chaotic behavior. In the analysis of chaotic behaviors, chaotic system analyses such as phase portraits, sensitivity to initial conditions, Lyapunov exponential spectrum and dimension analysis, bifurcation diagram with Matlab® software were investigated. The derived equations were embedded into a mobile random number generator (RNG). RNG was developed on the Raspberry Pi 3 Model B microcomputer. It has been shown that the obtained random numbers can be securely used in encryption applications by successfully passing the statistical tests NIST 800-22, FIPS 140-1 and ENT. An image encryption application with the generated random numbers was implemented on the Raspberry Pi 3 Model B microcomputer. Finally, the encrypted image was subjected to security tests such as histogram analysis, correlation and entropy coefficients, correlation card, NPCR, UACI utilizing MATLAB software. The performance and reliability of the encryption algorithm were also measured. This paper improves the current state of art as it implements chaotic random number generation algorithm in a small footprint micro computer.

1. Introduction

Understanding the complexity of Industry 4.0, which is communication between physical objects and other systems, is vitally important in this era. In many areas of research, continuous data exchange takes place, from home automation systems to credit cards, industry and intelligent agricultural systems. Although an attempt is being made to prevent this data, which is constantly being transmitted, from being retrieved by unauthorised people, this cannot be fully achieved. The protection of the data that can be accessed by unauthorised people against this undesirable situation is one of the important areas of research. In this context, encryption techniques have been developed to ensure data security. Data is encrypted and decrypted using a key. Decrypting encrypted data is only possible with the correct key. Thus, the key generation is an important problem when encrypting and decrypting data.

Nowadays, the use of random numbers generated from chaotic systems as keys due to their complex dynamics is an interesting topic. Chaos which is underlying interdependence of seemingly random events, can be briefly expressed as an order of disorder [1,2]. The fact that chaotic signals behave like noise and respond to their parameters and initial conditions distinguishes chaos-based random number generators (RNG), especially in secure communication. RNG is a system that uses hardware or software methods to generate statistically independent numbers and has no correlation at the output. These generators provide an unpredictable random output that can be generated utilizing the previous data [3].

Recently, a considerable literature has grown up around the theme of chaos-based RNG. Su et al. examined the dynamic analysis of a chaotic system in his study [4]. Xian et al. performed the dynamic analysis of a chaotic system with a variety of attractors utilizing an FPGA application [5]. Zhu et al. developed a chaotic system using the unique randomness of the eye pupils and performed a random number generation [6]. In a study by Avaroglu and Turk, an RNG was designed utilizing chaotic spirals as a natural source of noise [7]. In a study conducted Sahin, modern block encryption algorithms were investigated [8]. In another

* Corresponding author. Department of Computer Engineering, Faculty of Engineering, Hitit University, 19030, Corum, Turkey.
E-mail addresses: akifakgul@hitit.edu.tr, akgulakif@gmail.com (A. Akgul), bilalsau@gmail.com (B. Gurevin), ipehlivan@sakarya.edu.tr (I. Pehlivan), yldzmuhammet92@gmail.com (M. Yildiz), mkutlu@subu.edu.tr (M.C. Kutlu), emre.guleryuz1@ogr.sakarya.edu.tr (E. Guleryuz).

https://doi.org/10.1016/j.vlsi.2021.04.010
Received 8 December 2020; Received in revised form 23 March 2021; Accepted 13 April 2021
Available online 1 June 2021

study conducted, RNG was designed using a chaotic system with a golden ratio [9]. Also, there are some embedded system applications in the literature. Akgul et al. realized an application to encrypt the vein images [10]. Vergera et al. proposed a new chaotic cryptosystem for the encryption of very high-resolution digital images by using arbitrary precision arithmetic [11]. Vergera et al. implemented a chaotic cryptosystem by performing parallel computing on embedded systems with multiprocessors [12].

There are other architectural structures of RNGs were examined to utilise in cryptological applications [13–16]. Fast image encryption algorithms are developed using chaotic Henon systems and logistic maps [17] and using the Lorenz chaotic system [18]. Akgul et al. developed an electronic board to investigate chaotic systems with a fraction order [19]. Varan et al. performed a circuit implementation to investigate the dynamic properties of the nonlinear chaotic Aizawa system [20] Çimen et al. Performing the modelling of a chaotic movement using artificial neural networks [21]. Akgul et al. encoded the images of the hand veins via generating random numbers from a chaotic system [22]. Coskun et al. developed a computer-controlled platform to find the most appropriate system parameters in ADC-based TRNG designs [23]. In another recent study, encryption was made with random numbers generated on LabVIEW [24]. There are two primary aims of this paper; the first aim is to critically analyze the implementation of image encryption via RNGs in a mobile micro computer, the latter is to ascertain the usability of a chaotic system to encrypt an image.

The first section of this paper will examine chaotic behavior via the plotted Lyapunov exponential, phase portrait, Poincare section, spectrum and the bifurcation diagrams. In the second section; the selected equation will be analyzed and implemented for random number generation with the Runga Kutta-4 solution using the Python Software embedded in the Raspberry Pi 3 Model B microcomputer. There are a variety number of tests available for safety of a chaotic behaviour systems such as NIST 800-22, FIPS 140-1, ENT, AIS301, Diehard and Test U01. These tests will be detailed and randomly generated values will be analyzed [25–28]. In the third section; an image encoding application was performed using random number sequences that successfully passed the tests. Histogram analysis of the encrypted image, correlation maps, correlation and entropy coefficients, security analyses such as UACI and NPCR [29–32]. Finally, the results will be given and the consistency of the encryption algorithm against security threats will be demonstrated.

2. Used chaotic system and its dynamic analysis

2.1. Used chaotic system

This section details a dynamic analysis of the chaotic system to be used in random number generation. Analysis were performed with Matlab software. The chaotic system in Equation (2.2) [33] was implemented to derive the chaotic equations in Equation (2). These equations will be used in the study.

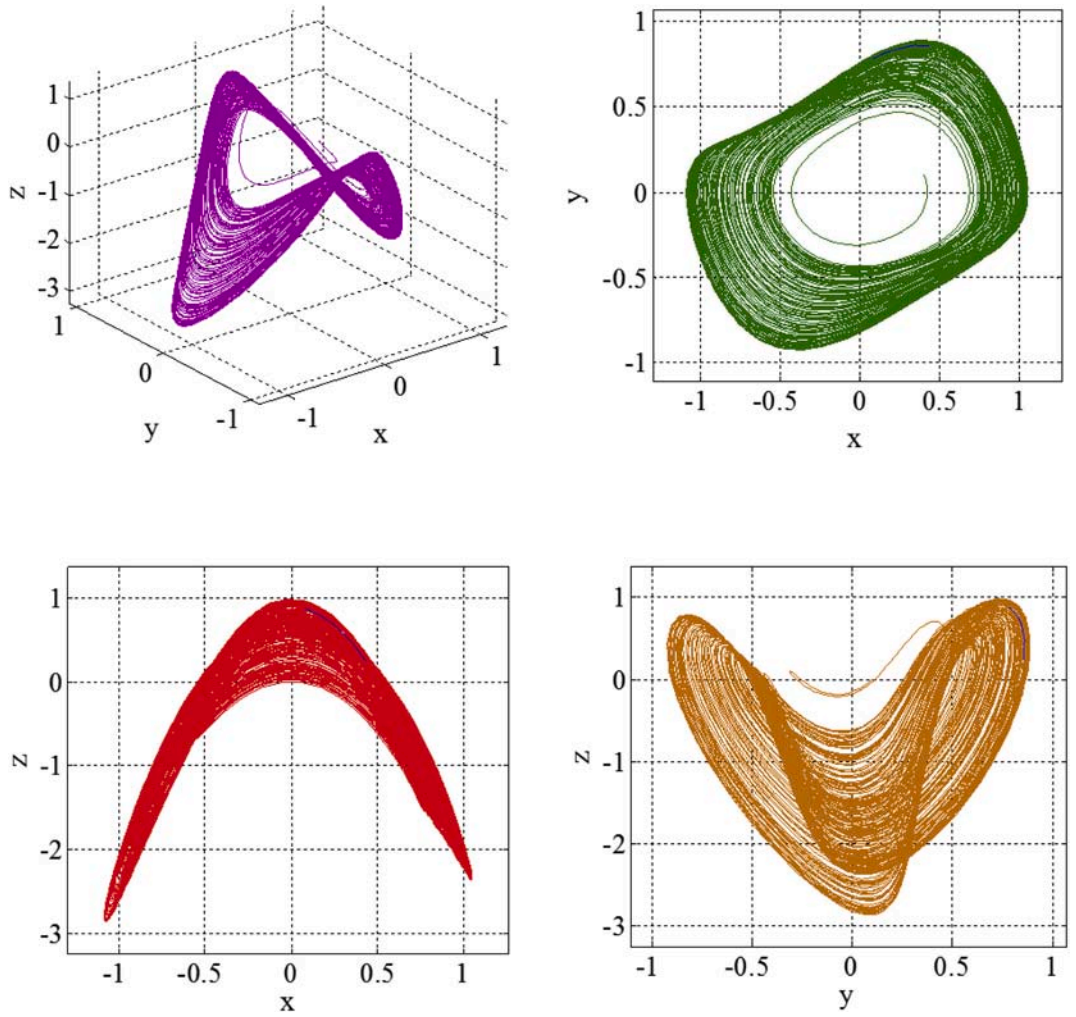

Fig. 1. Phase portraits of the system (x-y-z, x-y, x-z, y-z).

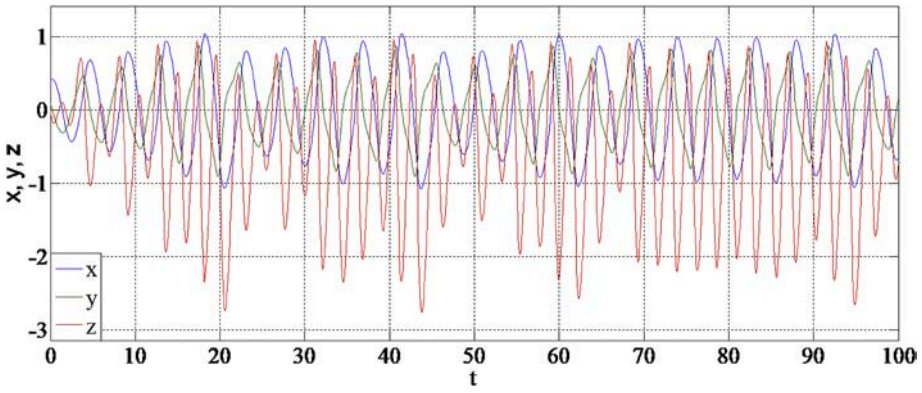

Fig. 2. Time series of the system (x, y, z).

$$\begin{aligned} \dot{x} &= y \\ \dot{y} &= -x + y.z \\ \dot{z} &= -x - 15.x.y - x.z \end{aligned} \tag{1}$$

The initial conditions of the chaotic system in equation are selected as $x_0 = 0.4, y_0 = 0.1, z_0 = 0$ and parameter values are selected as a = 1.9, b = 1.1, c = 11.5, d = 0.7. In next sections, dynamic analyses of the chaotic system will be performed.

$$\begin{aligned} \dot{x} &= a.y \\ \dot{y} &= -x + b.y.z \\ \dot{z} &= -x - c.x.y - d.x.z \end{aligned} \tag{2}$$

2.2. Dynamic analysis of chaotic system

Chaotic behavior has not been exhibited by every system. In order to a set differential equations to be accepted as a chaotic system, it must successfully pass some methods of analysis utilized in the literature. This section, therefore, examines some dynamic analyses such as phase portraits, time series, Lyapunov exponents and dimension, bifurcation diagram to analyze the chaotic behavior of the equations derived from the chaotic system in Equation [4,5,34,35]

2.2.1. Phase portrait and time series

The fact that the dynamic behavior of the phase portraits is complex, but within certain limits, the non-periodic behavior of the time series and their sensitive structure against the initial conditions show that the system exhibits a chaotic behavior [36]. The phase portraits x-y-z, x-y, x-z, y-z in Fig. 1 and the x-y-z time series in Fig. 2 were obtained for the chosen chaotic system. Fig. 3 analyzes the sensitivity of the z-phase to the initial value. As can be seen from the figures, the chaotic system exhibits a rich dynamic behavior and is sensitive to the initial conditions.

2.2.2. Lyapunov exponential and dimension analysis

The Lyapunov exponential (λ) is the mean of the separation angles of the curves in the phase space. The chaotic behaviour of the system is interpreted according to the state of. When the chaoticity of the system is investigated, Lyapunov exponents values must have a (+, 0, -) structure to be chaotic [37,38]. The Lyapunov exponential graphic in Fig. 4 shows the parameter range in which the system is in a chaotic state. Here, a Lyapunov exponential spectrum analysis was performed in the range of 0–1.5 according to the change of the parameter. When, Fig. 5 is investigated in detail, it appears that the system exhibits a chaotic behavior between approximately 0.95 and 1.15.

It can be interpreted whether a dynamic system is chaotic or not by calculating the Lyapunov dimension, known as the Kaplan-Yorke dimension. In order to talk about the chaoticity of the system, the Lyapunov dimension in a three-dimensional system must meet the 2¡D¡3 condition [39]. Equation 1.3 shows how to calculate the Lyapunov dimension. Equation (3) uses "j" for the number of system state variables and "λ_i" for Lyapunov exponents.

$$D = (j-1) + \frac{\sum_{i=1}^{j-1} \lambda_i}{|\lambda_j|} \tag{3}$$

Lyapunov exponentials of the chaotic system in the newly derived Equation (2) were chosen as $\lambda_1 = 0.12448$ $\lambda_2 = -0.00014555$, $\lambda_3 = -0.82275$. If Lyapunov values are chosen in Equation (3) instead, the

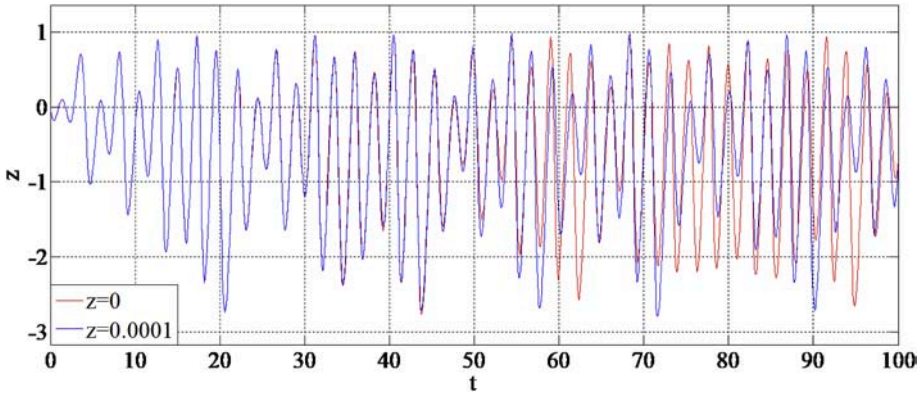

Fig. 3. Systems sensitivity to the initial conditions of z phase.

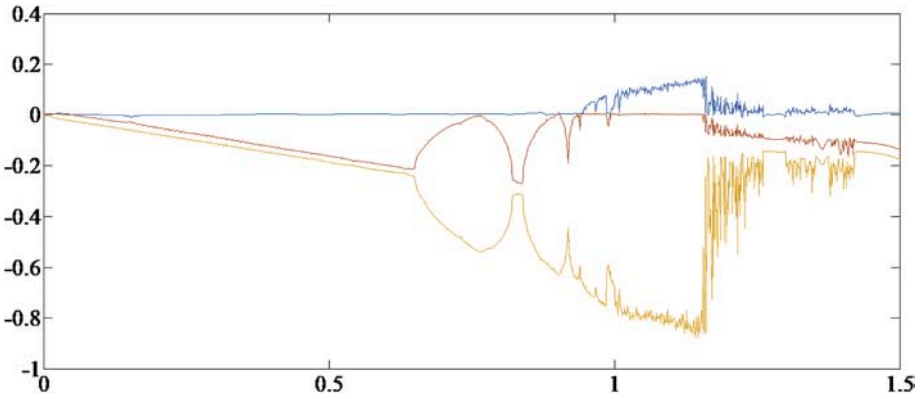

Fig. 4. Derived system's Lyapunov Spectrum Analysis for parameter 'b' (0–1.5).

Lyapunov dimension is given as D = 2.1511. Because 2¡D¡3 condition is met, the Lyapunov dimension can be called chaotic.

2.2.3. *Bifurcation diagram*

Bifurcation diagrams are most commonly used in dynamic systems analysis and is increasingly becoming one of the important methods of chaos analysis [40–42]. They can be used to analyze the system's sensitivity to initial values, and can be shown where the system is moving from periodic to chaotic [43,44]. The same results should be obtained at the same intervals in order to be able to talk about the chaos in the graphs obtained from the bifurcation analysis and the Lyapunov spectrum analysis. Fig. 6 shows the bifurcation diagram in the range of the system. The bifurcation results obtained in Fig. 6 with the Lyapunov analysis performed in Section 1.2.2 shall be consistent within the same intervals. This becomes clearer when a detailed analysis is performed in Fig. 7.

3. Mobile random number generator design and its statistical tests

3.1. *Design of mobile random number generator*

This section details the RNG design created in previous sections. Raspberry Pi 3 Model B was utilizing as a microcomputer for all random number generation operations. As shown in Fig. 8, the Raspberry Pi 3 Model B's desktop access was provided directly from the touchscreen. While the Python was used as a programming language, the Spyder IDE was used as a programming interface due to the capabilities it offers the user.

The pseudocode of steps followed in the RNG design is shown in Algorithm 1. When, the code was examine initial conditions and system parameters are entered after determining the equation to use for RNG. After entering the required step interval for the Runga Kutta-4 solution method, the system is then rendered discrete time in the x-y-z phases to obtain floating-point values. Float values are converted to a 32-bit binary form and the least significant bit (LSB) of these 32-bit numbers is processed. If we look at the 32-bit numbers in Fig. 9, the values begin to be the same as we approach the 0-bit value. Towards the end, it appears that different values appear independently of each other. Therefore, it is more advantageous to select the bit from the last bit. However, if the LSB bit selection is too small, the time increases.

Algorithm 1
Pseudo Code of Random Number Generation.

Input: Parameters and initial condition of chaotic system
Output: Tested random numbers)
1: START
2: Entering system parameters and initial condition of 3D chaotic system
3:Determination of the value of Δ_h
4: Sampling with determination Δ_h value for RK4
5: while *minimum 1MBit data* **do**
    Select "s = 8" bit LSB;
    Solving the 3D chaotic system using RK4 algorithm;
    Obtaining time series as float numbers (x, y and z);
    Convert float to 32 bit binary numbers;
    Select s bits one of x, y and z phases from RNG ;
    end while
6: Apply NIST-800-22-800-22 and FIPS 140-1 Tests for each minimum 1MBit data
7: if *test results* == *pass* **then**
    Ready tested random numbers for RNG applications;
else
    Test results == false;
    end if
7: EXIT

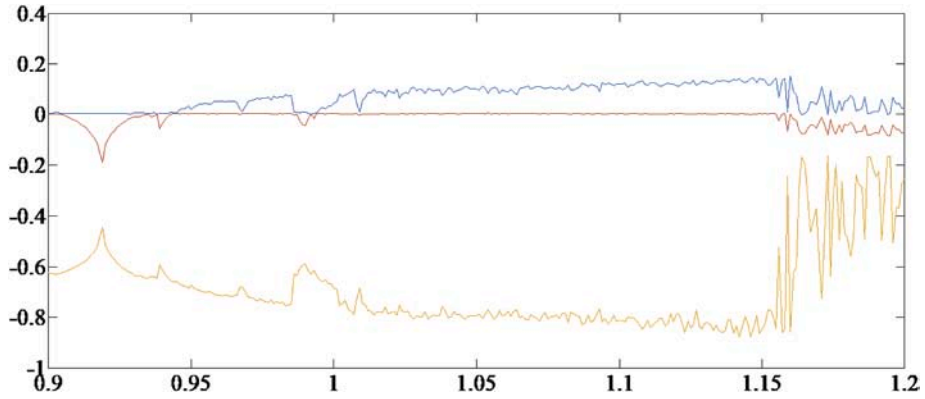

Fig. 5. Derived system's Lyapunov Spectrum Analysis for parameter 'b' (0.9–1.2).

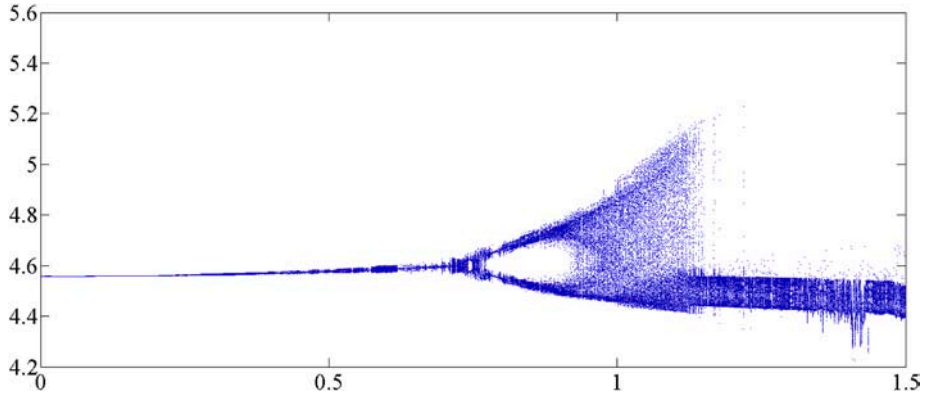

Fig. 6. Derived system's bifurcation diagram of parameter 'b' (0–1.5).

For this reason, s = 8 (LSB) is selected as the bit. A random sequence of numbers with a length of 1000000 bits is obtained by combining the 8-bit values obtained separately from each phase. As a final process, this record is subjected to NIST and FIPS tests. If no successful results are achieved, the process is repeated by changing the selected LSBs.

Fig. 9 shows the first 15 values of the float values obtained from the x phase with the RK4 solution. Then, all these values are converted to the 32-bit binary format. Because it is assumed that "s" is more sensitive than the LSB, it is selected as 8 bits, so that a separate sequence of 8-bit data is generated by taking the last 8 bits of each 32-bit binary value. Each of these 8 created data bits is added up one at a time to get a random number with a total length of 1000000 bits.

Fig. 10 shows oscilloscope images obtained with Raspberry Pi 3 Model B to show that random numbers obtained from x-y-z phases can be used in any application in real life.

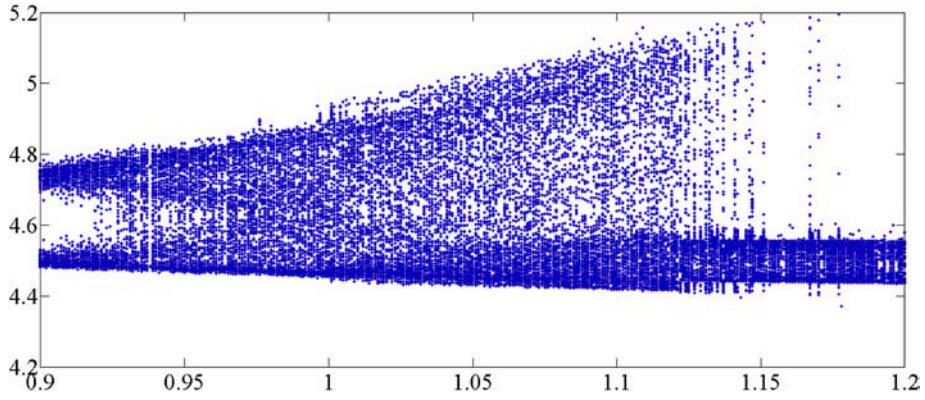

Fig. 7. Derived system's bifurcation diagram of parameter 'b' (0.9–1.2).

Fig. 8. Raspberry pi 3 model B desktop access.

3.2. RNG statistical tests

3.2.1. NIST 800-22 test

The NIST 800-22 test is an internationally recognized random number test. It consists of 16 different tests per se, and to speak of the randomness of the number, it must pass all 16 tests successfully. At each stage, the test results are evaluated according to their performance. As long as the $P \geq 0.001$ conditions are met in each test phase, the test result is considered successful [45].

The random numbers used in the test were obtained with 30 million bits obtained from each of the x, y and z systems. In the last case, we used 30 sequences from 1000000 bits. Tables 1–3 proves from the values that the random numbers obtained from the x-y-z phases (8 bits) of the chaotic system have successfully passed the NIST 800-22 test.

3.2.2. FIPS 140-1 test

FIPS 140-1 Test; It consists of four phases such as Monobit, Poker, Runs and Long Runs.

- Monobit test: It is measured in 20,000-bit sequences. It is desirable that every 20,000-bit sequence is a number between 9654 and 10,346, and the result is taken as an average.
- Poker test: measured in 5000 bit sequences. Each 5 thousand-bit sequence is converted into 4 bits and the test result is used for the 5000-bit sequence. This test result is asked to give a result of 1.03–57.4 for each 5000 bit sequences in a 1 million bit sequence, and the result at 1 million bits is the average of the test for all 5000 bit sequences.
- Run Test: It is calculated by taking into account the number of zeros that are continued successively in a 20,000-bit sequence. This Parameter has 6 different options, 1,2,3,4,5 and 6. 1 Million bit sequences are returned to 20 binary bit sequences, consecutive zeros are counted, and a test calculation is performed. There are various ideal results for 6 options of this parameter.
- Long-term test: Calculated from the number of more than 34 consecutive ones and zeros in a 20,000-Bit sequence.

In order to the generated random numbers to be considered successful, they must successfully pass these four phases [46]. Table 4 shows the FIPS 140-1 test result values of random number sequences obtained from 8-bit data from each of the x-y-z dimensions. Compared to the previous section with the success criteria, it can be seen that each of the tests is positive.

3.2.3. ENT test

ENT test is used to test a variety of byte arrays generated by pseudo-random number generator applications. It is a test application developed by John Walker who conducts tests [47]. 5 different statistical tests in the ENT test that define the randomness of bit strings are available. ENT

	0
0	0.40854
1	0.415123
2	0.419713
3	0.422289
4	0.422849
5	0.421405
6	0.417984
7	0.412621
8	0.405362
9	0.396261
10	0.385377
11	0.372773
12	0.358515
13	0.342673
14	0.325319
15	0.306524

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	0	1	1	1	1	1	0	1	1	0	1	0	0	0	1	0	0	1	0	1	1	0	0	0	0	0	1	1	0	0	1
1	0	0	1	1	1	1	1	0	1	1	0	1	0	1	0	0	1	0	0	0	1	0	1	1	0	0	0	0	1	0	0	0
2	0	0	1	1	1	1	1	0	1	1	0	1	0	1	1	0	1	1	1	0	0	1	0	0	1	0	0	1	1	1	0	0
3	0	0	1	1	1	1	1	0	1	1	0	1	1	0	0	0	0	0	1	1	0	1	1	0	0	1	0	0	0	1	1	0
4	0	0	1	1	1	1	1	0	1	1	0	1	1	0	0	0	0	1	1	1	1	1	1	1	1	0	1	0	1	1	0	1
5	0	0	1	1	1	1	1	0	1	1	0	1	0	1	1	1	1	1	0	0	0	0	1	0	0	1	1	1	0	0	1	1
6	0	0	1	1	1	1	1	0	1	1	0	1	0	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	1	0
7	0	0	1	1	1	1	1	0	1	1	0	1	0	0	1	1	0	1	0	0	0	0	1	1	0	0	0	0	0	1	0	1
8	0	0	1	1	1	1	1	0	1	1	0	0	1	1	1	1	1	0	0	0	1	0	1	1	1	0	1	0	0	0	1	1
9	0	0	1	1	1	1	1	0	1	1	0	0	1	0	1	0	1	1	1	0	0	0	1	0	1	1	0	0	0	1	1	1
10	0	0	1	1	1	1	1	0	1	1	0	0	0	1	0	1	0	1	0	1	0	0	0	0	0	0	1	0	1	0	1	1
11	0	0	1	1	1	1	1	0	1	0	1	1	1	1	1	0	1	1	0	1	1	1	0	0	0	0	0	1	1	0	1	1
12	0	0	1	1	1	1	1	0	1	0	1	1	0	1	1	1	1	0	0	0	1	1	1	1	0	1	0	1	0	0	1	1
13	0	0	1	1	1	1	1	0	1	0	1	0	1	1	1	1	0	1	1	1	0	0	1	0	1	1	1	0	0	1	0	1
14	0	0	1	1	1	1	1	0	1	0	1	0	0	1	1	0	1	0	0	1	0	0	0	0	0	0	1	0	0	1	1	0
15	0	0	1	1	1	1	1	0	1	0	0	1	1	1	0	0	1	1	1	1	0	0	0	0	1	0	1	0	1	0	0	0

Fig. 9. Conversion of float numbers resolved from dimension x to binary number format.

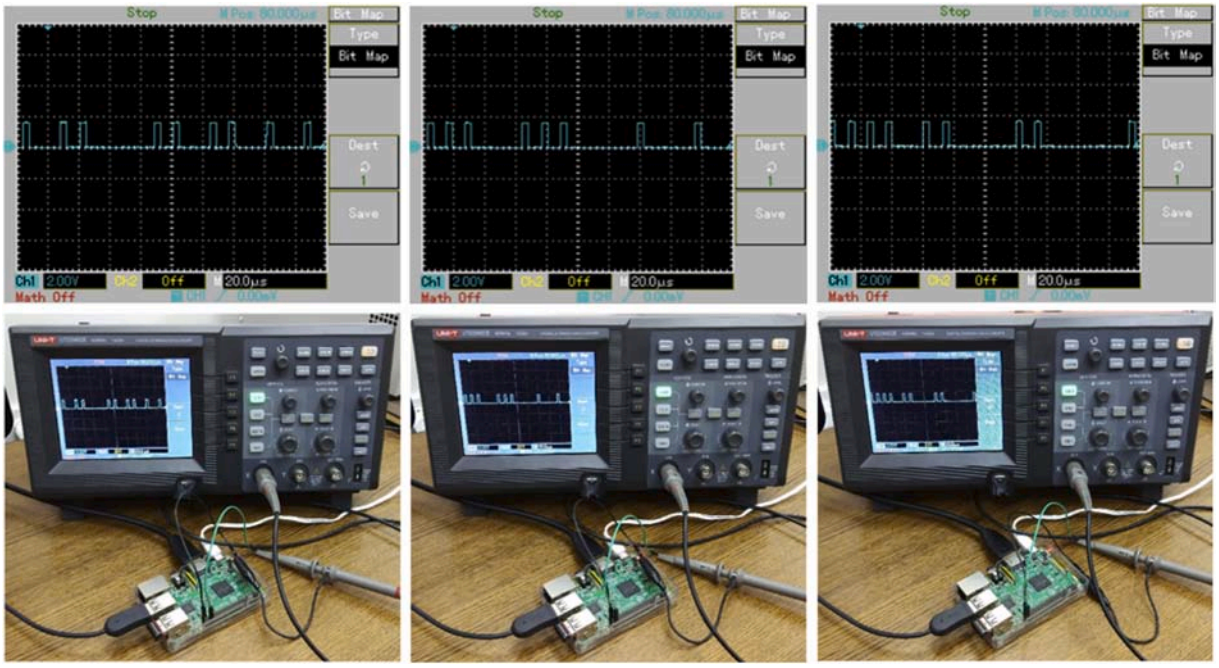

Fig. 10. Oscilloscope images obtained with Raspberry Pi 3 Model B to show that random numbers obtained from x-y-z phases can be used in any application in real life. Random numbers generated from x-y-z phases are displayed on the oscillacope by using Raspberry Pi's digital pins.

Table 1
NIST-800-22 test results of x phase random numbers from 10 sequences.

Statistical Tests	P-values of X Phase										Result
	Seq 1	Seq 2	Seq 3	Seq 4	Seq 5	Seq 6	Seq 7	Seq 8	Seq 9	Seq 10	
Frequency (Monobit) Test	0,1835	0,1192	0,2919	0,9777	0,5837	0,1789	0,1842	0,4099	0,3049	0,3183	Successful
Block-Frequency Test	0,9886	0,1649	0,0702	0,4610	0,3056	0,1400	0,9067	0,2273	0,5022	0,3641	Successful
Cumulative-Sums Test	0,1415	0,1167	0,2201	0,8686	0,2452	0,2636	0,0825	0,4951	0,5877	0,2621	Successful
Runs Test	0,737	0,0240	0,5708	0,4226	0,8024	0,0791	0,9174	0,0602	0,4324	0,9164	Successful
Longest-Run Test	0,6039	0,5758	0,4253	0,3382	0,8563	0,1592	0,7289	0,6873	0,4104	0,5961	Successful
Binary Matrix Rank Test	0,3413	0,8762	0,0524	0,8100	0,6383	0,0956	0,9482	0,2054	0,0501	0,2586	Successful
Discrete Fourier Transform Test	0,2513	0,0563	0,2465	0,5756	0,8688	0,7831	0,7273	0,0475	0,8114	0,2871	Successful
Non-Overlapping Templates Test	0,0103	0,0038	0,1032	0,0648	0,0734	0,1741	0,0046	0,0257	0,1121	0,0007	Successful
Overlapping Templates Test	0,4714	0,0958	0,3870	0,4721	0,1636	0,5811	0,8367	0,2090	0,0206	0,1799	Successful
Maurer's Universal Statistical Test	0,9012	0,8093	0,2836	0,2873	0,8252	0,1502	0,6055	0,1665	0,4424	0,5970	Successful
Approximate Entropy Test	0,6074	0,1259	0,8708	0,6373	0,1393	0,9607	0,5509	0,3680	0,2935	0,0706	Successful
Random-Excursions Test (x = −4)	0,5684	0,3920	0,2703	0,9492	0,2843	0,8167	0,4367	0,5582	0,4387	0,2678	Successful
Random-Excursions Variant Test (x = −9)	0,649	0,2980	0,8898	0,2646	0,3546	0,9667	0,5326	0,7952	0,8765	0,3414	Successful
Serial Test-1	0,7416	0,0393	0,1277	0,8443	0,9136	0,5158	0,6658	0,5121	0,7385	0,6218	Successful
Serial Test-2	0,9123	0,1554	0,6097	0,8193	0,8504	0,4285	0,7238	0,3073	0,8974	0,7541	Successful
Linear-Complexity Test	0,11	0,8527	0,7154	0,6373	0,1393	0,5540	0,5509	0,2225	0,2429	0,9616	Successful

Table 2
NIST-800-22 test results of y phase random numbers from 10 sequences.

Statistical Tests	P-values of Y Phase										Result
	Seq 1	Seq 2	Seq 3	Seq 4	Seq 5	Seq 6	Seq 7	Seq 8	Seq 9	Seq 10	
Frequency (Monobit) Test	0,56191	0,5700	0,6101	0,1957	0,8603	0,1789	0,9888	0,3030	0,9267	0,4703	Successful
Block-Frequency Test	0,47336	0,2719	0,7270	0,3677	0,9393	0,1400	0,4007	0,4684	0,6423	0,8546	Successful
Cumulative-Sums Test	0,75706	0,4355	0,5957	0,2657	0,4768	0,2636	0,5553	0,1832	0,9645	0,4429	Successful
Runs Test	0,05961	0,8147	0,1022	0,5487	0,3544	0,0791	0,5041	0,3958	0,3321	0,2089	Successful
Longest-Run Test	0,74611	0,6602	0,1636	0,8475	0,2908	0,1592	0,7091	0,0551	0,2146	0,5402	Successful
Binary Matrix Rank Test	0,35217	0,4007	0,0890	0,7054	0,4044	0,0956	0,4577	0,6886	0,7845	0,3441	Successful
Discrete Fourier Transform Test	0,04545	0,2955	0,6073	0,2188	0,3884	0,7831	0,4247	0,7481	0,0829	0,6864	Successful
Non-Overlapping Templates Test	0,00514	0,0013	0,0285	0,0096	0,0439	0,1741	0,0396	0,0102	0,1715	0,0210	Successful
Overlapping Templates Test	0,6057	0,3564	0,2987	0,1444	0,6619	0,5811	0,6704	0,4500	0,9522	0,2021	Successful
Maurer's Universal Statistical Test	0,44388	0,4323	0,4128	0,2542	0,5700	0,1502	0,4441	0,2798	0,1667	0,8922	Successful
Approximate Entropy Test	0,25962	0,9370	0,0900	0,0910	0,1418	0,9607	0,9604	0,5243	0,8699	0,8779	Successful
Random-Excursions Test (x = −4)	0,16593	0,4123	0,6360	0,4355	0,8335	0,8167	0,7705	0,3618	0,9539	0,7754	Successful
Random-Excursions Variant Test (x = −9)	0,4958	0,1372	0,7896	0,4812	0,1525	0,9667	0,5639	0,5328	0,7228	0,5064	Successful
Serial Test-1	0,51622	0,0864	0,5378	0,5098	0,0478	0,5158	0,6170	0,3746	0,2256	0,8113	Successful
Serial Test-2	0,24476	0,3857	0,1120	0,5775	0,0316	0,4285	0,5053	0,8630	0,0893	0,4302	Successful
Linear-Complexity Test	0,32704	0,7726	0,0535	0,0910	0,4325	0,5540	0,6725	0,4185	0,3342	0,1718	Successful

Table 3
NIST-800-22 test results of z phase random numbers from 10 sequences.

Statistical Tests	P-values of Z Phase										Result
	Seq 1	Seq 2	Seq 3	Seq 4	Seq 5	Seq 6	Seq 7	Seq 8	Seq 9	Seq 10	
Frequency (Monobit) Test	0,9029	0,3942	0,1014	0,6355	0,2150	0,5157	0,8619	0,3154	0,9124	0,1868	Successful
Block-Frequency Test	0,5050	0,4814	0,7986	0,8257	0,7683	0,8320	0,2169	0,9288	0,1116	0,3140	Successful
Cumulative-Sums Test	0,7477	0,2772	0,1975	0,9567	0,2730	0,7903	0,9862	0,1745	0,9046	0,2012	Successful
Runs Test	0,1096	0,6332	0,5818	0,0275	0,2733	0,6466	0,7188	0,8313	0,0914	0,6296	Successful
Longest-Run Test	0,3937	0,6466	0,5204	0,4243	0,4969	0,9122	0,3486	0,8619	0,2845	0,1586	Successful
Binary Matrix Rank Test	0,7038	0,5751	0,3578	0,8677	0,4929	0,5949	0,8258	0,1993	0,7796	0,1106	Successful
Discrete Fourier Transform Test	0,4247	0,9488	0,9780	0,8765	0,0878	0,0540	0,5029	0,3783	0,7411	0,7342	Successful
Non-Overlapping Templates Test	0,0507	0,6616	0,0129	0,4826	0,0817	0,0890	0,6543	0,2893	0,0005	0,0546	Successful
Overlapping Templates Test	0,8565	0,5384	0,0263	0,8744	0,4233	0,6423	0,1686	0,9930	0,4758	0,6546	Successful
Maurer's Universal Statistical Test	0,5288	0,5362	0,3515	0,2354	0,4497	0,1302	0,3213	0,5843	0,8353	0,6841	Successful
Approximate Entropy Test	0,5061	0,5362	0,1480	0,2995	0,1959	0,6136	0,3284	0,9990	0,3579	0,6683	Successful
Random-Excursions Test (x = −4)	0,2135	3,8530	0,2365	0,6545	0,2578	0,6589	0,0549	0,6774	0,3498	0,7654	Successful
Random-Excursions Variant Test (x = −9)	0,1828	0,3196	0,0432	0,0524	0,4389	0,8240	0,3656	0,0976	0,0052	0,1095	Successful
Serial Test-1	0,5992	0,4029	0,2460	0,6805	0,8996	0,6833	0,2761	0,9458	0,0788	0,4357	Successful
Serial Test-2	0,9364	0,1920	0,4731	0,7677	0,9160	0,8146	0,3607	0,9458	0,1387	0,7189	Successful
Linear-Complexity Test	0,3663	0,5801	0,8857	0,8857	0,8675	0,0491	0,8147	0,2492	0,0466	0,7013	Successful

test results of random numbers obtained from x, y and z outputs are values are shown in Table 5. According to Table 5, the last 8 of x, y and z passing all tests Random numbers generated from bit values can safely be used in areas where they are needed. can be used.

4. Encryption application and its security analysis

This section details a mobile encryption application of an image with random numbers generated from values obtained by discretizing the x, y, z phases of the derived system. These derived values produced from the Runga Kutta-4 solution. This method was implemented by the Spyder IDE (Python 3.7). Subsequently, the Matlab® program performed security analysis for the encrypted image to measure the quality of the encryption algorithm and the security of the encrypted image.

4.1. Encryption application

In this section, a sample image with Raspberry Pi 3 Model B was encrypted as a mobile phone with random numbers generated from the x-y-z phases of the derived system Fig. 11.

The algorithm monitored during encryption is represented as pseudocode in Algorithm 2 According to this algorithm, the image to be encrypted is first introduced into the program. After the dimensions (row = A, column = B) of the image converted to grayscale are calculated, an index value of k = 0 is defined. To edit all row and column values, a nested for loop is created in the form of row (A) and column (B). Each row and column value is converted to an 8-bit binary format. The 8-bit sequences from the entered random numbers (one of the x, y, z-phases) and the 8-bit values of the image are XOR processed. The entire matrix structure of the image is processed. The values obtained as a result of the XOR operation are converted to a decimal form. This is how the image is encrypted. The matrix values before the encoding of the image in Fig. 12 and after the encoding (with x-phase) are shown in Fig. 13. The analysis methods to measure the quality and reliability of the encryption process will be discussed in the next sections.

Algorithm 2
Pseudo code of Image Encryption Algorithm.

```
Input: Tested random numbers and image data
Output: Encrypted image data (8 bit) )
1: START
2: Entering random numbers and image data
3: Determination of image size (A=256, B=256)
4: Convert the image to GrayScale
5: for i=0 → A-1 do
    for j=0 → B-1 do
      Bin=Convert image(i, j) to 8bit k=0
      for m=0 → 7 do
        k=k+1 Encrypted = Bin XOR random numbers
        Encrypted to Decimal
      end for
    end for
  end for
7: EXIT
```

Table 4
Random numbers FIPS 400-1 test results for produced from x, y and z.

FIPS 140-1 Tests	Success Criterion	Value			Result
		x	y	z	
Monobit Test	$9654 < x < 10346$	10014	9929	10071	Successful
Poker Test	$1.03 < x < 57.4$	8.3264	14.5024	8.6464	Successful
Run Test (1)	$2267 \leq x \leq 2733$	2567	2545	2491	Successful
Run Test (2)	$1079 \leq x \leq 1421$	1267	1206	1264	Successful
Run Test (3)	$502 \leq x \leq 748$	615	658	617	Successful
Run Test (4)	$223 \leq x \leq 402$	338	296	316	Successful
Run Test (5)	$90 \leq x \leq 223$	129	165	151	Successful
Long Run Test	$34 > \text{Run}$	13	12	13	Successful

Table 5
ENT test results of random numbers obtained from x, y and z.

Test name	Average	Ideal Results	Result
Arithmetic Mean	127,4990	127,5	Successful
Entropy	7,9995	8	Successful
Correlation	0,0031804	0	Successful
Chi Square	259,041	10% and 90% between	Successful
Monte Carlo	3.1444 (error = 0.0009)	Pi Number	Successful

Fig. 11. Mobil Encryption Application (8 bit result).

4.2. Security analysis

Successful results in the image encoding process do not only depend on the reliability of random numbers, but also they depend on a good encryption algorithm. The good encryption method must be resistant to all types of cryptoanalytical, statistical, and security attacks. For this purpose, the literature introduces some security analysis methods for image encryption. In this section, some statistical tests such as histogram analysis, correlation and entropy catalog, and correlation maps were performed to measure the performance of the encryption process [31,48,49].

4.2.1. Histogram analysis

In this section, a histogram analysis of the source image and the image encoded with each of the x, y, and z phases was performed. The closer the data obtained from an encrypted image is in the histogram, the more difficult the decryption becomes [19]. In Fig. 14, the histogram of the source image has a scatter structure, while the histogram analysis of the encrypted images encoded with random numbers obtained from the x-y-z phases in Fig. 15 shows tight values. This is considered an indication that the encryption process was successful.

4.2.2. Correlation and entropy constants

In this section, entropy and correlation coefficients are investigated. The entropy value must be close to 8 and the correlation value must be very close to 0 for the quality of the random distribution and thus a successful coding within the encoded image [50,51]. Results are displayed. In Table 6, while the correlation values of the source image are around 1, the correlation values of the images encoded separately with each of the x, y, and z phases are very close to 0. Similarly, the entropy value of the encrypted image is very close to 8, indicating that successful encryption against attacks has been achieved. Table 6 contains some of the most recent entropy values found in the literature. This result demonstrates that the encryption system can effectively withstand malicious attacks, and the findings were consistent with previous research.

4.2.3. Correlation maps

This section details the correlation maps of the source image and encoded images. Fig. 16 shows correlation maps of the original image, the image encoded in the x-phase, the y-phase, and the z-phase. While the row, column, and diagonal correlation maps of the source image are diagonally concentrated without encryption, it is seen that the correlation distributions of the encoded images are in a fairly homogeneous distribution. This shows that the images obtained as a result of homogeneous distribution encoding are present in a very random distribution [49].

4.2.4. NPCR and UACI

This section investigated the NPCR (number of pixel change rates) and Unified Average Changing Intensity (UACI) values by comparing the 8-bit and 1-bit encrypted images that are mobile encrypted in Fig. 21. While NPCR expresses how many percent of all pixels change, the UACI value shows ratio of pixels change [31,56]. In previous studies, the NPCR was considered to be a good encryption, with a value greater than 99.6% and a UACI greater than or equal to 30% [57].

To perform the encryption process of a 1-bit image, the threshold operation was first applied to the source image, and imagery consisting of 1-bit values was obtained, as shown in Figs. 17 and 18. After that, the encoded images in Fig. 19 were obtained by XOR for each pixel value with random numbers of x, y, and z phases. When we compare the 8-bit

Fig. 12. Pre-encrypted Matrix value of the image.

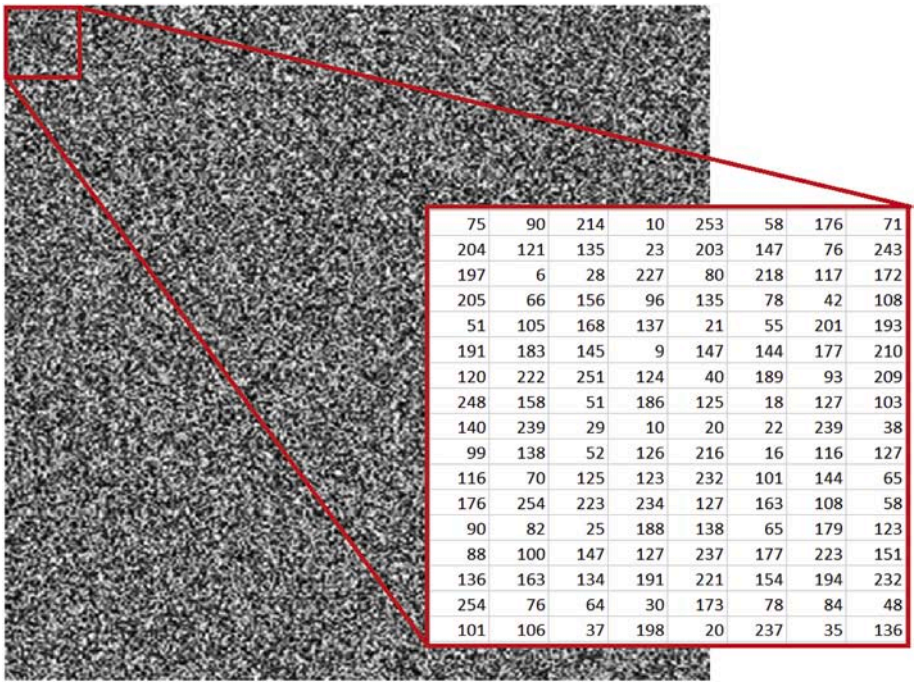

Fig. 13. Pre-encrypted Matrix value of the image.

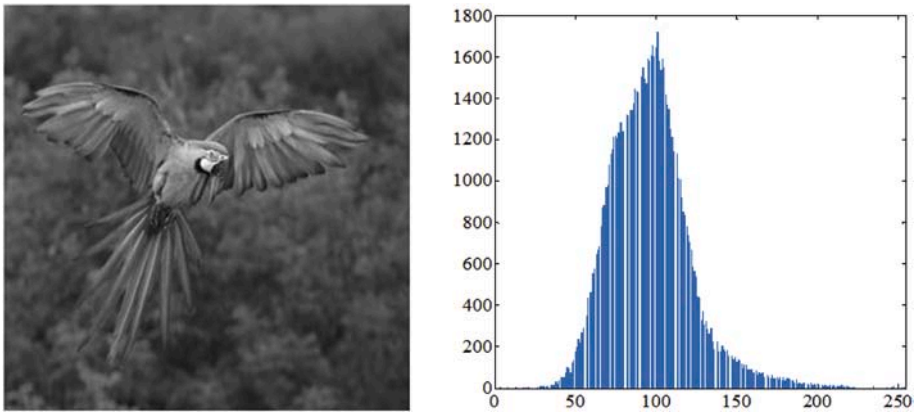

Fig. 14. Source image histogram analysis.

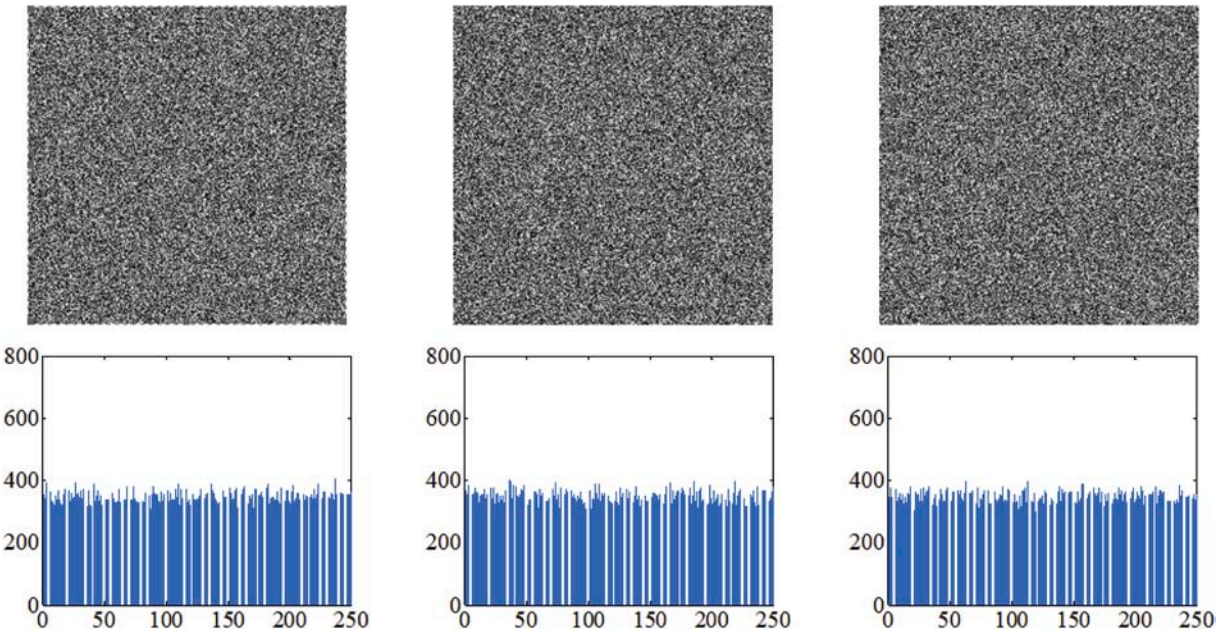

Fig. 15. Encrypted images and histogram analysis of random numbers generated 8 bit encryption by x,y and z phases.

Table 6
Correlation and entropy of source image and encrypted images by x-y-z phases.

Image	Horizontal Correlation	Vertical Correlation	Entropy
Source Image	0,91209	0,88927	6,6297
Encrypted Image (x phase)	−0,005053	0,000252	7,9981
Encrypted Image (y phase)	0,0032142	0,00178	7,9976
Encrypted Image (z phase)	−0,00056	−0,00427	7998
Kaur et al. [52]	0,012	−0,0063	7,9989
Cavusoglu et al. [53]	−0,00197	N/A	7,9637
Chen et al. [54]	−0,0028	0,0171	7,9891
Xu et al. [55])	0,0019	0,0263	7,9974

and 1-bit encodings, the total number of changed pixels of the encrypted 1-bit image and the average of the modified pixels should be the same. Table 8 shows the NPCR and UACI values of 1-bit and 8-bit images encoded in x, y, and z phases. For example, while the pixels of the 8-bit encoded image change at a rate of 99.60% in the x-phase, the change rate of the pixels is displayed as 27.68% compared to the previous values. However, the NPCR and UACI values of the 1-bit encrypted image are the same. The NPCR and UACI values obtained from the four studies are shown in Table 7.

4.2.5. *Key space, sensitivite and encryption quality*

The key domain and the variation are the results of the security analysis, The security keys are crucial for an encryption algorithm for the security of encrypted images against various attacks and brute force attacks. To prove the security of the encryption algorithm, it must be very sensitive to the variation of the secret key and also the length of the key field must be more than 2^{128} to prevent brute force attacks [60]. In the chaotic equation used in encryption, there are a, b, c, d parameters and initial values of x, y, z. If the sensitivity of these 7 values is assumed to be 10^{-14}, the entire key space is greater than its value. This result

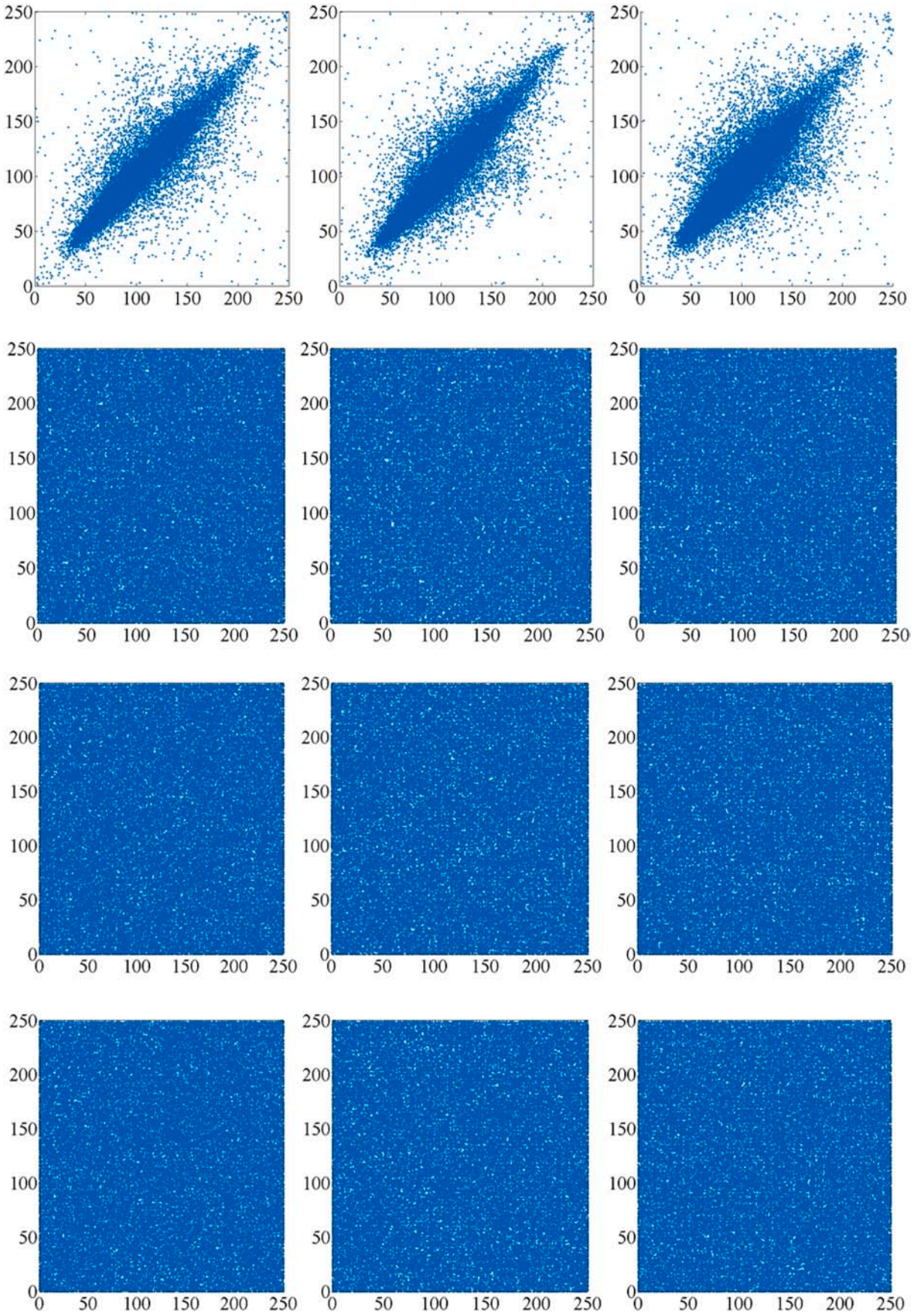

Fig. 16. From top to bottom original view, image encrypted by x-, y- and z-phase's horizontal, vertical, diagonal correlation maps.

Fig. 17. 1 bit conversion of source image

Fig. 18. Mobile encryption application (1 bit view).

Table 7
x-y-z phases encrypted images NPCR and UACI analysis.

Image	NPCR	UACI
x phase encrypted image (8 bit)	99,60	27,68
y phase encrypted image (8 bit)	99,62	27,62
z phase encrypted image (8 bit)	99,62	27,54
Li et al. (8 bit) [58]	99,60	33,43
Zhang et al. (8 bit) [59]	88,99	30,21
Kaur et al. (8 bit) [52]	99,67	33,58
Cavusoglu et al. (8 bit) [53]	99,63	31,63
x phase encrypted image (1 bit)	50,13	50,13
y phase encrypted image (1 bit)	49,88	49,88
z phase encrypted image (1 bit)	50,21	50,21

Table 8
The quality of the encryption algorithm tests.

Encrypted Value	AS	CC	D
X	64430	$-6.4e^{12}$	89821
Y	64791	$-1.18e^{12}$	89831
Z	64766	$-1.1e^{12}$	89825

shows that a small change in the security key leads to a completely different encryption result.The quality of the encryption algorithm is evaluated by three tests; 1) the irregular deviation factor AS, 2) the correlation factor CC and 3) the maximum deviation factor D. Irregular deviation test's expected value for an image with N × M pixels is close to (N × M)/2. If there is no correlation, a CC correlation value is expected to be close to zero. If the maximum deflection factor is D, the expected value for an image with N × M pixels is its proximity to the N × M value (see Fig. 20).

5. Conclusion

In the article study, a microcomputer-based mobile RNG was designed using a nonlinear chaotic system. After verifying that the generated random numbers can be used securely with statistical tests according to NIST 800-22, FIPS 140-1 and ENT, image encryption was applied using these numbers. Security analyses such as histogram analysis, correlation and entropy coefficient, correlation card, UACI, NPCR were used to measure the performance of the encryption algorithm and the reliability of the encrypted image. As a result of all this research and studies, a collective source of information has been provided for studies in areas such as chaotic system analysis methods, chaos-based generation of mobile random numbers, random randomness tests, image encryption, and security analysis of encrypted images. A mobile RNG design has been developed that can be easily used in studies such as encryption, watermarking, and confidential writing where data security is important. In addition to the studies, new and different algorithms for generating random numbers can be presented to make it easier for random numbers to pass random tests. New encryption algorithms can be developed and applied to text, audio, video, and images. In addition, all of these operations can be combined into one interface to provide

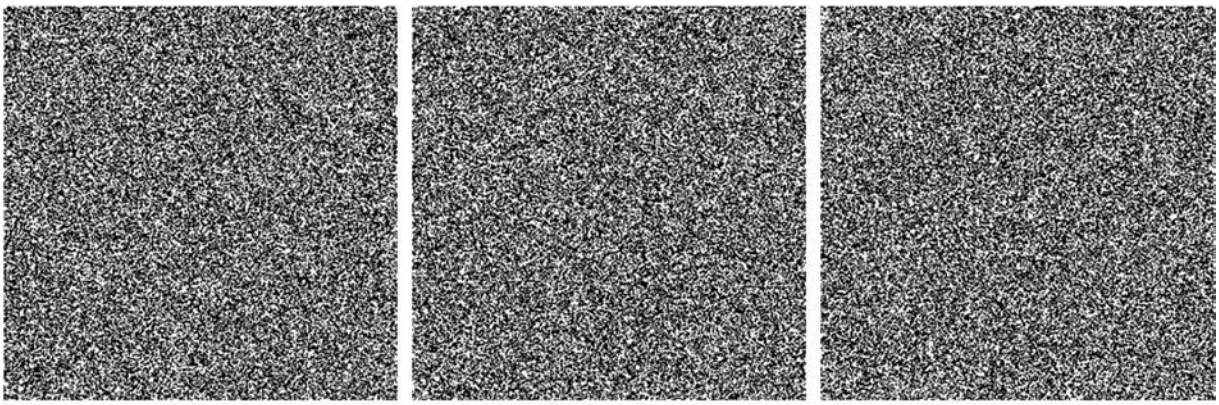

Fig. 19. 1 bit view image encrypted by a) x-phase, b) y-phase, c) z-phase.

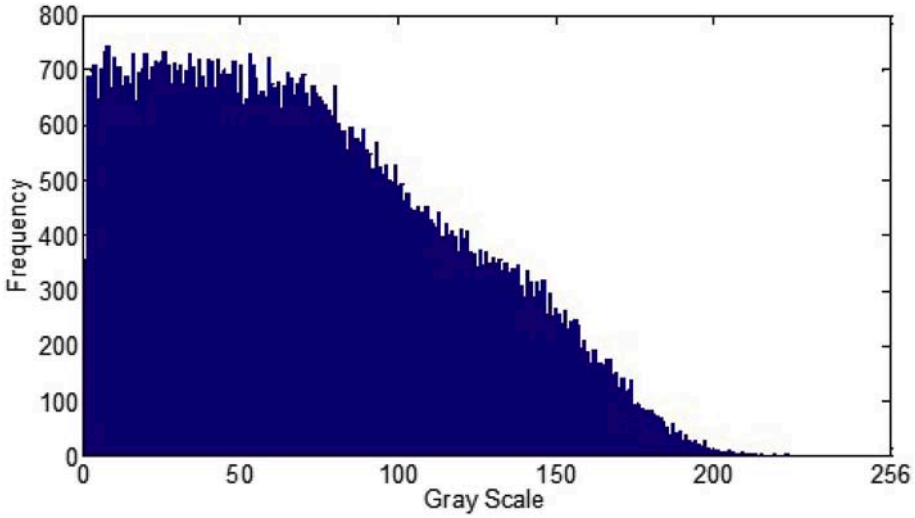

Fig. 20. Histogram of the difference between the original image and the x-encoded image.

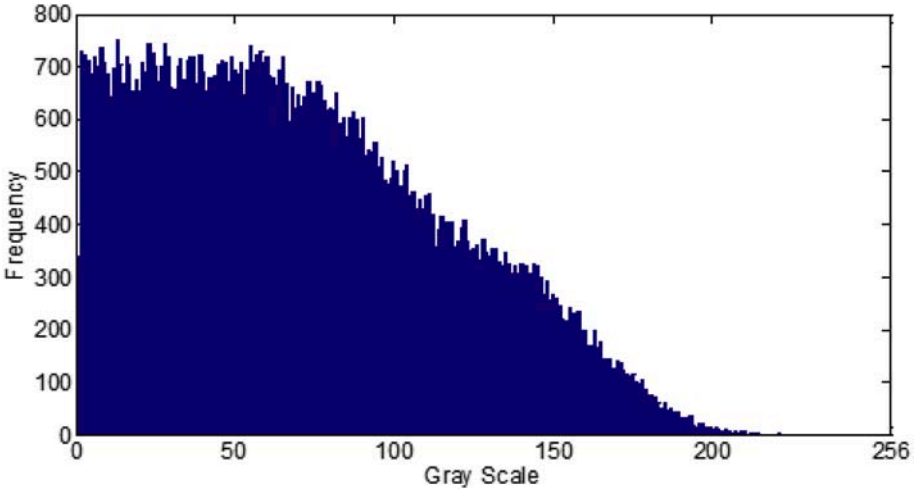

Fig. 21. Histogram of the difference between the original image and the y-encoded image.

comfort to the user.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was supported by the Scientific and the Research Council of Turkey (TUBITAK) under Grant No. 117E284.

References

[1] İ. Pehlivan, Yeni Kaotik Sistemler: Elektronik Devre Gerçeklemeleri, Senkronizasyon Ve Güvenli Haberleşme Uygulamaları, Doktora Tezi, Fen Bilimleri Enstitüsü, Sakarya Üniversitesi, 2010.
[2] Murat Tuna, Can Bulent Fidan, A Study on the importance of chaotic oscillators based on FPGA for true random number generating (TRNG) and chaotic systems, J. Facul. Eng. Architect. Gazi Univ. 33 (2) (2018) 469–486. Number: 2 Reporter: Journal of the Faculty of Engineering and Architecture of Gazi University.
[3] K. Ozdemir, S. Kilinc, S. Ozoguz, Random number generator design using continuous-time chaos, in: 2008 *{IEEE} 16th {Signal} {Processing}, {Communication} and {Applications} {Conference}*, 2008, pp. 1–4.
[4] Kalin Su, Dynamic analysis of a chaotic system, Optik 126 (24) (2015) 4880–4886.
[5] Yong-ju Xian, Cheng Xia, Tao-tao Guo, Kun-rong Fu, Chang-biao Xu, Dynamical analysis and {FPGA} implementation of a large range chaotic system with coexisting attractors, Results Phys. 11 (2018) 368–376.
[6] Hegui Zhu, Cheng Zhao, Xiangde Zhang, Lianping Yang, A novel iris and chaos-based random number generator, Comput. Secur. 36 (2013) 40–48.
[7] E. Avaroğlu, M. Türk, Random number generation using multi-mode chaotic attractor, in: 2013 *21st {Signal} {Processing} and {Communications} {Applications} {Conference} ({SIU})*, 2013, pp. 1–4.
[8] Fatih Şahin, Modern {Blok} şifreleme {Algoritmaları}, İstanbul Aydın Üniversitesi Dergisi 7 (26) (2015) 23–40.
[9] Akin Ozdemir, Ihsan Pehlivan, Akif Akgul, Emre Guleryuz, A strange novel chaotic system with fully golden proportion equilibria and its mobile microcomputer-based RNG application, Chin. J. Phys. 56 (6) (2018) 2852–2864. Number: 6 Reporter: Chinese Journal of Physics.
[10] Akif Akgül, Mustafa Zahid Yildiz, Ömer Faruk Boyraz, Emre Güleryüz, Sezgin Kaçar, Bilal Gürevin, Doğrusal olmayan yeni bir sistem ile damar görüntülerinin mikrobilgisayar tabanlı olarak şifrelenmesi, Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi 35 (3) (2020) 1369–1386.
[11] A. Flores-Vergara, E.E. García-Guerrero, E. Inzunza-González, O.R. López-Bonilla, E. Rodríguez-Orozco, J.R. Cárdenas-Valdez, E. Tlelo-Cuautle, Implementing a chaotic cryptosystem in a 64-bit embedded system by using multiple-precision arithmetic, Nonlinear Dynam. 96 (1) (2019) 497–516.
[12] Abraham Flores-Vergara, Everardo Inzunza-González, Enrique Efren García-Guerrero, Oscar Roberto López-Bonilla, Eduardo Rodríguez-Orozco, Juan Miguel Hernández-Ontiveros, José Ricardo Cárdenas-Valdez, Esteban Tlelo-Cuautle, Implementing a chaotic cryptosystem by performing parallel computing on embedded systems with multiprocessors, Entropy 21 (3) (2019) 268.
[13] Fatih Özkaynak, Kriptolojik {Rasgele} {Sayı} Üreteçleri, Türkiye Bilişim Vakfı Bilgisayar Bilimleri ve Mühendisliği Dergisi 8 (2) (2016) 37–45.
[14] Lazaros Moysis, Aleksandra Tutueva, Christos Volos, Denis Butusov, Jesus M. Munoz-Pacheco, Hector Nistazakis, A two-parameter modified logistic map and its application to random bit generation, Symmetry 12 (5) (2020) 829.
[15] Lazaros Moysis, Aleksandra Tutueva, K. Christos, Denis Butusov, A chaos based pseudo-random bit generator using multiple digits comparison, Chaos Theor. Appl. 2 (2) (2020) 58–68.
[16] Lazaros Moysis, Christos Volos, Sajad Jafari, Jesus M. Munoz-Pacheco, Jacques Kengne, Karthikeyan Rajagopal, Ioannis Stouboulos, Modification of the logistic map using fuzzy numbers with application to pseudorandom number generation and image encryption, Entropy 22 (4) (2020) 474.
[17] Mir Mohammad Reza Alavı Milani, Hüseyin Pehlivan, Sahereh Hosein Pour, Kaos tabanlı bir şifreleme yöntemi ve analizi, Akademik Bilisim (2013) 487–493.
[18] Ünal Çavuşoğlu, Yılmaz Uyaroğlu, İhsan Pehlivan, Sürekli {Zamanlı} otonom {Kaotik} devre {Tasarımı} ve sinyal gizleme uygulaması, Gazi Üniversitesi Mühendislik-Mimarlık Fakültesi Dergisi 29 (1) (2014).
[19] A. Akgül, *Yeni {Kaotik} {Sistemler} İle {Rasgele} {Sayi} Üreteci {Tasarımı} Ve Çoklu-{Ortam} {Verilerinin} {Yüksek} {Güvenlikli} {A}ifrelenmesi*. {PhD} {thesis}, Doktora Tezi, Sakarya Üniversitesi, Fen Bilimleri Enstitüsü, Sakarya, 2015.
[20] Metin Varan, Bahar Ulusoy, Ihsan Pehlivan, Bilal Gurevin, Akif Akgul, Nonlinear {Analysis} and {Circuit} {Realization} of {Chaotic} {Aizawa} {System}, 2018, p. 191. Co-chair.
[21] Murat Erhan Çimen, Sezgin Kaçar, Emre Güleryüz, Bilal Gürevin, Akif Akgül, Kaotik bir hareket videosunun yapay sinir ağları ile modellenmesi, Balıkesir Üniversitesi Fen Bilimleri Enstitüsü Dergisi 20 (3) (2018) 23–35.
[22] Akif Akgül, Mustafa Zahid Yıldız, Ömer Faruk Boyraz, Emre Güleryüz, Sezgin Kaçar, Bilal Gürevin, Microcomputer-based encryption of vein images with a non-linear novel system, J. Facul. Eng. Architect. Gazi Univ. 35 (3) (2020) 1369–1385.
[23] Selçuk Coşkun, Ihsan Pehlivan, Akif Akgül, Bilal Gürevin, A new computer-controlled platform for {ADC}-based true random number generator and its applications, Turk. J. Electr. Eng. Comput. Sci. 27 (2) (2019) 847–860.
[24] B. Gurevin, M. Yildiz, E. Guleryuz, M.C. Kutlu, O. Sorgun, A chaos based image encryption on LabVIEW, Chaos Theor. Appl. 2 (2) (2020) 69–76.
[25] A. Akhshani, A. Akhavan, A. Mobaraki, S.-C. Lim, Z. Hassan, Pseudo random number generator based on quantum chaotic map, Commun. Nonlinear Sci. Numer. Simulat. 19 (1) (2014) 101–111.
[26] Ünal Çavuşoğlu, Ahmet Zengin, Ihsan Pehlivan, Sezgin Kaçar, A novel approach for strong s-box generation algorithm design based on chaotic scaled zhongtang system, Nonlinear Dynam. 87 (2) (2017) 1081–1094.
[27] Barış Karakaya, Arif Gülten, Mattia Frasca, A true random bit generator based on a memristive chaotic circuit: {Analysis}, design and {FPGA} implementation, Chaos, Solit. Fractals 119 (2019) 143–149.
[28] Ismail Koyuncu, Ahmet Turan Özcerit, The design and realization of a new high speed fpga-based chaotic true random number generator, Comput. Electr. Eng. 58 (2017) 203–214.
[29] Moatsum Alawida, Azman Samsudin, Je Sen Teh, Rami S. Alkhawaldeh, A new hybrid digital chaotic system with applications in image encryption, Signal Process. 160 (2019) 45–58.
[30] Shah Fahd, Mehreen Afzal, Haider Abbas, Waseem Iqbal, Salman Waheed, Correlation power analysis of modes of encryption in {AES} and its countermeasures, Future Generat. Comput. Syst. 83 (2018) 496–509.
[31] Shyamalendu Kandar, Dhaibat Chaudhuri, Apurbaa Bhattacharjee, Bibhas Chandra Dhara, Image encryption using sequence generated by cyclic group, J. Inf. Secur. Appl. 44 (2019) 117–129.
[32] Yingri Su, Yan Wo, Guoqiang Han, Reversible cellular automata image encryption for similarity search, Signal Process. Image Commun. 72 (2019) 134–147.
[33] Mohammad Ali Jafari, Ezzedine Mliki, Akif Akgul, Viet-Thanh Pham, Sifeu Takougang Kingni, Xiong Wang, Sajad Jafari, Chameleon: the most hidden chaotic flow, Nonlinear Dynam. 88 (3) (may 2017) 2303–2317.
[34] Qiang Lai, Akif Akgul, Metin Varan, Jacques Kengne, Alper Turan Erguzel, Dynamic analysis and synchronization control of an unusual chaotic system with exponential term and coexisting attractors, Chin. J. Phys. 56 (6) (2018) 2837–2851.
[35] Nihat Pamuk, Dinamik {Sistemlerde} {Kaotik} {zaman} {Dizilerinin} {Tespiti}, Balıkesir Üniversitesi Fen Bilimleri Enstitüsü Dergisi 15 (1) (2016) 78–92.
[36] Chongxin Liu, Tao Liu, Ling Liu, Kai Liu, A new chaotic attractor, Chaos, Solit. Fractals 22 (5) (2004) 1031–1038.
[37] Buncha Munmuangsaen, Banlue Srisuchinwong, A hidden chaotic attractor in the classical {Lorenz} system, Chaos, Solit. Fractals 107 (2018) 61–66.
[38] Derya Yılmaz, Nihal Fatma Güler, Kaotik zaman serisinin analizi üzerine bir araştırma, Gazi Üniversitesi Mühendislik ve Mimarlık Fakültesi Dergisi 21 (4) (2006) 759–779.
[39] Serdar Ethem Hamamci, Veysel Göğebakan, İbrahim Işık, A new chaotic system with chaos entanglement, in: 2015 23nd Signal Processing and Communications Applications Conference (SIU), IEEE, 2015, pp. 2597–2600.
[40] Atiyeh Bayani, Karthikeyan Rajagopal, Abdul Jalil M Khalaf, Sajad Jafari, G. D. Leutcho, J. Kengne, Dynamical analysis of a new multistable chaotic system with hidden attractor: {Antimonotonicity}, coexisting multiple attractors, and offset boosting, Phys. Lett. (2019).
[41] Yun-Sam Kim, Jong-Chol Kim, Analysis of chaotic vibration of {Shilnikov}-type in rotor with asymmetric and non-linear stiffness, Int. J. Non Lin. Mech. 109 (2019) 132–139.
[42] Zhouchao Wei, Bin Zhu, Jing Yang, Matjaž Perc, Mitja Slavinec, Bifurcation analysis of two disc dynamos with viscous friction and multiple time delays, Appl. Math. Comput. 347 (2019) 265–281.
[43] Fuhong Min, Chuang Li, Lei Zhang, Chunbiao Li, Initial value-related dynamical analysis of the memristor-based system with reduced dimensions and its chaotic synchronization via adaptive sliding mode control method, Chin. J. Phys. 58 (2019) 117–131.
[44] Jinliang Wang, You Li, Shihong Zhong, Xiaojie Hou, Analysis of bifurcation, chaos and pattern formation in a discrete time and space {Gierer} {Meinhardt} system, Chaos, Solit. Fractals 118 (2019) 1–17.
[45] Koyuncu İsmail, Kriptolojik {Uygulamalar} İçin {FPGA} Tabanlı {Yeni} {Kaotik} {Osilatörlerin} Ve {Gerçek} {Rasgele} {Sayı} Üreteçlerinin {Tasarımı} Ve {Gerçeklenmesi} {PhD} {thesis}, Doktora Tezi, Sakarya Üniversitesi, Fen Bilimleri Enstitüsü, 2014.
[46] Fuyan Sun, Shutang Liu, Cryptographic pseudo-random sequence from the spatial chaotic map, Chaos, Solit. Fractals 41 (5) (2009) 2216–2219.
[47] A. Beirami, H. Nejati, W.H. Ali, Zigzag map: a variability-aware discrete-time chaotic-map truly random number generator, Electron. Lett. 48 (24) (2012) 1537–1538.
[48] Yang Liu, Shanyu Tang, Ran Liu, Liping Zhang, Zhao Ma, Secure and robust digital image watermarking scheme using logistic and {RSA} encryption, Expert Syst. Appl. 97 (may 2018) 95–105.
[49] T. Sivakumar, Pu Li, A secure image encryption method using scan pattern and random key stream derived from laser chaos, Opt Laser. Technol. 111 (2019) 196–204.
[50] A. Akhshani, S. Behnia, A. Akhavan, H. Abu Hassan, Z. Hassan, A novel scheme for image encryption based on {2D} piecewise chaotic maps, Opt Commun. 283 (17) (2010) 3259–3266.

[51] E. Yavuz, R. Yazıcı, M.C. Kasapbaşi, E. Yamaç, Enhanced chaotic key-based algorithm for low-entropy image encryption, in: 2014 22nd *{Signal} {Processing} and {Communications} {Applications} {Conference} ({SIU})*, 2014, pp. 385–388.
[52] M. Kaur, V.I.J.A.Y. Kumar, Efficient image encryption method based on improved lorenz chaotic system, Electron. Lett. 54 (9) (2018) 562–564.
[53] Ünal Çavuşoğlu, Sezgin Kaçar, Ahmet Zengin, Ihsan Pehlivan, A novel hybrid encryption algorithm based on chaos and s-aes algorithm, Nonlinear Dynam. 92 (4) (2018) 1745–1759.
[54] Xiao Chen, Chun-Jie Hu, Adaptive medical image encryption algorithm based on multiple chaotic mapping, Saudi J. Biol. Sci. 24 (8) (2017) 1821–1827.
[55] Lu Xu, Zhi Li, Jian Li, Wei Hua, A novel bit-level image encryption algorithm based on chaotic maps, Opt Laser. Eng. 78 (2016) 17–25.
[56] Yue Wu, Sos Agaian, {NPCR} and {UACI} {Randomness} {Tests} for {Image} {Encryption}, 2011, p. 9.
[57] Padmapriya Praveenkumar, Rengarajan Amirtharajan, K. Thenmozhi, John Bosco Balaguru Rayappan, Pixel scattering matrix formalism for image encryption—a key scheduled substitution and diffusion approach, AEU-Int. J. Electron. Commun. 69 (2) (2015) 562–572.
[58] Zhen Li, Changgen Peng, Liangrong Li, Xiaoyan Zhu, A novel plaintext-related image encryption scheme using hyper-chaotic system, Nonlinear Dynam. 94 (2) (2018) 1319–1333.
[59] Jian Zhang, Dezhi Hou, Honge Ren, Image encryption algorithm based on dynamic dna coding and chen's hyperchaotic system, Math. Probl Eng. (2016) 2016.
[60] Xingyuan Wang, Lin Teng, Xue Qin, A novel colour image encryption algorithm based on chaos, Signal Process. 92 (4) (2012) 1101–1108.