IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 12 January 2024, accepted 6 February 2024, date of publication 13 February 2024, date of current version 20 February 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3365638

RESEARCH ARTICLE

A Novel Design of Random Number Generators Using Chaos-Based Extremum Coding

SHUNSUKE ARAKI[1], JI-HAN WU[2], AND JUN-JUH YAN[2]
[1]Graduate School of Computer Science and Systems Engineering, Kyushu Institute of Technology, Kawazu, Iizuka, Fukuoka 820-8502, Japan
[2]Department of Electronic Engineering, National Chin-Yi University of Technology, Taichung 41107, Taiwan

Corresponding author: Jun-Juh Yan (jjyan@ncut.edu.tw)

This work was supported by the National Science and Technology Council, Taiwan, under Grant NSTC 112-2221-E-167-025.

ABSTRACT This paper proposes a new chaos-based extremum coding method to realize a true random number generator (RNG). Based on the chain rule, we innovatively introduce two parameters into the dynamics of chaotic systems to modulate the speed and amplitude of state responses. Then, by discretizing the continuous modulated chaotic system, the corresponding discrete chaotic system can be obtained, which allows the use of low-cost micro-controllers for implementation, enhancing system stability, and reducing costs. Also, a novel chaos-based extremum coding approach is proposed for generating a random extremum-coded sequence (RECS). Using this RECS to switch and decide modulation parameters significantly improves the randomness of the sequences generated by the RNGs. To highlight the contribution of this RNG design, the randomness and security of the RNGs are evaluated by statistical tests such as NIST, Diehard, and ENT. Through comparisons with recent published works, the results show that the proposed chaotic extremum-coded RNG can demonstrate superior performance with a higher level of randomness.

INDEX TERMS Chaos, discretization, extremum coding, random number generator, statistical test.

I. INTRODUCTION

With the increasing importance of information security, random number generators (RNGs) have received much attention for various encryption algorithms. Random sequences play crucial roles in cryptography, encrypted communication and other information security applications, requiring high levels of randomness and unpredictability [1]. Generally, RNGs can be classified into two main categories: Pseudo RNGs (PRNGs) and True RNGs (TRNGs). The PRNG sequences are periodic, causing the sequence to repeat after a certain cycle, which means PRNGs lack genuine randomness. On the other hand, TRNGs utilize physical phenomena to generate true randomness, but they consume more hardware resources and time, resulting in poorer performance such as slower generation speeds, and higher costs [2], [3], [4]. To enhance the quality of randomness, researchers have engaged in comprehensive exploration of various methodologies. Charalampidis et al. [5] introduced a novel segmented chaotic mapping based on the z-shaped fuzzy number for designing a pseudo-random bit generator. Xu and Tang [6] proposed a method for generating random key streams by utilizing DNA encoding and fuzzy delayed self-feedback chaotic neurons. In the research by Zhao et al. [7], they incorporated the Secure Hash Algorithm (SHA) to derive a 384-bit hash outcome from pure images, employed as raw keys. This approach metamorphoses images into key substrates via hashing, ensuring key unpredictability and security. Ma et al. [8] employed Analog-to-Digital Converters (ADC) to sample analog signals for yielding high-speed digitized random numbers. Yang et al.'s study [9] harnessed the intricate nonlinear dynamic characteristics of hyper-chaos to generate sequences boasting heightened random quality. To forestall the dynamic degradation of digital chaos, Zheng and Hu [10] proposed a method for highly secure stream ciphering grounded in an analog-digital hybrid chaotic system encompassing the Chen chaotic system and the 3D Logistic map. Founded upon the Ring oscillator, Kamadi and Abbas [11] introduced a bona fide random number generator. This technique exploits the noise attributes of physical devices, accomplishing authentic random generation suited to applications demanding pronounced randomness. In Camara et al.'s research [12], predicated on human gait data, they proffered a methodology for genuine random

The associate editor coordinating the review of this manuscript and approving it for publication was Chao-Yang Chen.

number generation. By harnessing data from myriad sensors, they extracted randomness from biological attributes for the purpose of random number generation. In Kiran et al.'s research [13], they utilized the Hem Cubic Map and Ricker's population model to devise a chaos-based pseudorandom number generator. In synthesis, the proposed methods and techniques underscored the indispensability of RNGs in data security. Obviously, researchers introduced chaos, physical device noise, biological attributes, and other modes to design high-quality and exceedingly random sequences for the requisites of diverse applications. Therefore, this study also aims to provide a simpler, lower-cost and higher-quality RNG. To achieve this design goal, a chaos-based extremum coding method is newly presented to implement a reliable true RNG. Chaotic systems inherently possess high sensitivity and unpredictability, meaning even slight differences in initial conditions can lead to dramatic changes in system behavior. Many studies [2], [3], [4], [5], [7], [9], [13] have used these features to design RNGs. In this study, to further promote the randomness quality of RGNs, we not only use chaotic systems but also introduce new modulation parameters (k_m, k_s) to change the response speed and state amplitude of chaotic systems. This design provides a more flexible method that can improve the quality of the RNG. We'll detail that later. By discretizing the continuous modulated chaotic systems, the corresponding discrete chaotic systems can be obtained. Consequently, we can use micro-controllers for implementation and enhancing system stability, and reducing costs. Furthermore, a chaos-based extremum coding technique is presented to generate a RECS, using this RECS to switch modulation parameters significantly improves the randomness of the proposed RNGs.

The rest of this study is organized as follows. Section II describes the main structure for designing chaotic extremum-coded RNGs. It includes the modulation design and the discretization of continuous chaotic systems as well as the extremum coding approach and extremum-coded modulation mechanism. In Section III, to show the contributions of this design, the statistical methods including NIST, Diehard, and ENT are employed to evaluate this RNGs and some comparisons and observations with the previous works are given. Conclusions are provided in Section IV.

II. DESIGN OF CHAOTIC EXTREMUM-CODED RNGs

In this paper, the structure of proposed chaos-based extremum-coded RNGs is shown in Figure 1. To successfully complete the design of RNGs, the proposed core technologies include an adjustable chaotic system with modulation parameters k_m, k_s, system discretization, and extremum-coded modulation mechanism. When combined with SHA-256, a true RNG can be achieved.

A. THE MODULATION DESIGN OF THE CHAOTIC SYSTEMS

As shown in Figure 1, we firstly discuss the modulation design for a class of continuous chaotic systems. In the following, we introduce the 4-dimention hyper-chaotic system

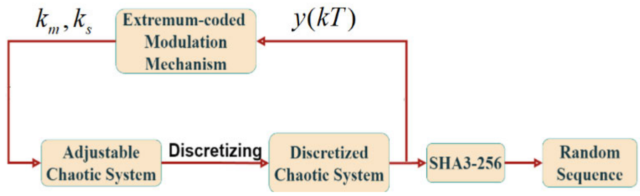

FIGURE 1. The structure of the chaotic extremum-coded RNG.

for discussion, of course, the technology developed here can be easily modified and applied to different chaotic systems. The dynamics of the 4-dimention hyper-chaotic system [14] can be described as follows:

$$\begin{aligned}\frac{dx_1(t)}{dt} &= a(x_2(t) - x_1(t))\\ \frac{dx_2(t)}{dt} &= bx_4(t) + cx_2(t) + dx_1(t) - x_1(t)x_3(t) - x_3(t)x_4(t)\\ \frac{dx_3(t)}{dt} &= fx_3(t) + x_2(t)x_4(t) + x_1(t)x_2(t)\\ \frac{dx_4(t)}{dt} &= gx_2(t) - ex_4(t) - 0.05x_1(t)x_3(t)\end{aligned} \tag{1}$$

The parameters in (1) are given as below.

$$a = 16, b = 45, c = -2, d = 45, f = -4, g = 16, e = 16 \tag{2}$$

We firstly define new state variables $y_i(\tau) = k_m x_i(t)$, $i = 1, 2, 3, 4$ with a new time variable $\tau = \frac{t}{k_s}$. The modulation parameters k_s, k_m are introduced to adjust the response speed and state amplitude of chaotic systems, respectively. Therefore, by using the chain rule [15], one has

$$\begin{aligned}\dot{y}_1(\tau) &= \frac{dy_1(\tau)}{d\tau} = k_m\frac{dx_1(t)}{d\tau}\\ &= k_m\frac{dx_1(t)}{dt}\frac{dt}{d\tau} = k_s(a(k_m x_2(t) - k_m x_1(t)))\\ &= k_s(a(y_2(\tau) - y_1(\tau)))\end{aligned} \tag{3}$$

In the similar way, one can derive

$$\begin{aligned}\dot{y}_2(\tau) &= \frac{dy_2(\tau)}{d\tau} = k_m\frac{dx_2(t)}{dt}\frac{dt}{d\tau}\\ &= k_s(by_4(\tau) + cy_2(\tau) + dy_1(\tau) - \frac{y_1(\tau)y_3(\tau)}{k_m}\\ &\quad - \frac{y_3(\tau)y_4(\tau)}{k_m})\end{aligned} \tag{4}$$

$$\begin{aligned}\dot{y}_3(\tau) &= \frac{dy_3(\tau)}{d\tau} = k_m\frac{dx_3(t)}{dt}\frac{dt}{d\tau}\\ &= k_s(fy_3(\tau) + \frac{y_2(\tau)y_4(\tau)}{k_m} + \frac{y_1(\tau)y_2(\tau)}{k_m})\end{aligned} \tag{5}$$

and

$$\begin{aligned}\dot{y}_4(\tau) &= \frac{dy_4(\tau)}{d\tau} = k_m\frac{dx_1(t)}{dt}\frac{dt}{d\tau}\\ &= k_s(gy_2(\tau) - ey_4(\tau) - 0.05\frac{y_1(\tau)y_3(\tau)}{k_m})\end{aligned} \tag{6}$$

Based on the derivation above, we have obtained a new adjustable chaotic system that allows for the modulation of system's response speed and state magnitude, as shown in (7). For simplicity, in (7) we have omitted the new time variable τ. Clearly, when $k_s = k_m = 1$, the system (7) degenerates into the original system (1).

$$\begin{aligned}
\dot{y}_1 &= k_s a(y_2 - y_1) \\
\dot{y}_2 &= k_s(by_4 + cy_2 + dy_1 - \frac{y_1 y_3}{k_m} - \frac{y_3 y_4}{k_m}) \\
\dot{y}_3 &= k_s(fy_3 + \frac{y_2 y_4}{k_m} + \frac{y_1 y_2}{k_m}) \\
\dot{y}_4 &= k_s(gy_2 - ey_4 - 0.05\frac{y_1 y_3}{k_m})
\end{aligned} \tag{7}$$

To verify the adjustable system (7), we conduct the simulation with initial conditions as $y_1(0) = 2.1k_m$, $y_2(0) = 0$, $y_3(0) = -15.21k_m$, $y_4(0) = -24.74k_m$.

The simulation results are depicted in the Figures 2-4 below. According to the results in Figures 2 and 3, it can be observed that when we assign modulation parameters with $k_s = k_m = 2$, both the response speed and amplitude of the chaotic system (7) are, as expected, twice those of the original system with $k_s = k_m = 1$. The strange attractors are shown in Figure 4, demonstrating that even after system modulation, the chaotic system's strange attractor is surely preserved.

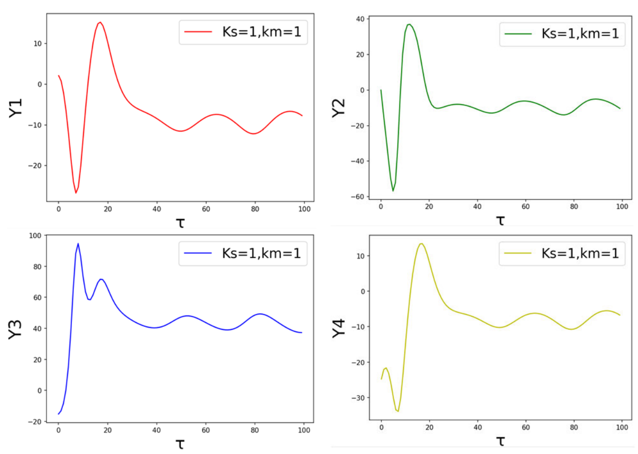

FIGURE 2. The state responses with $k_s = k_m = 1$.

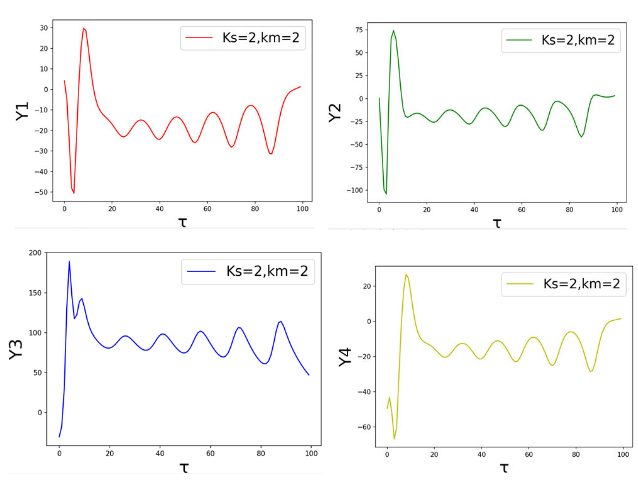

FIGURE 3. The state responses with $k_s = k_m = 2$.

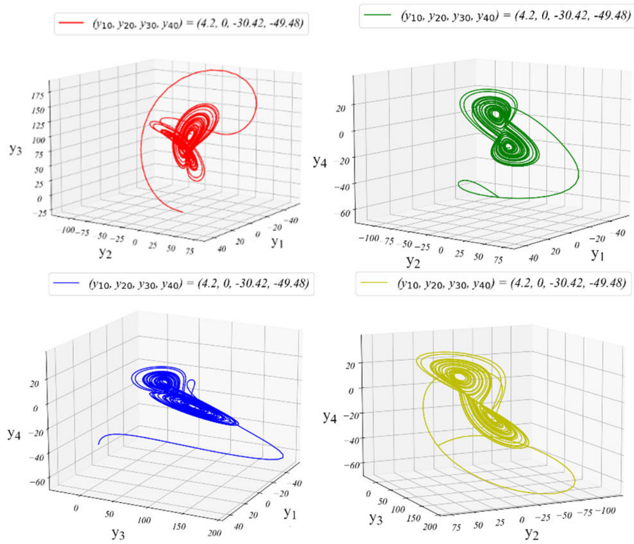

FIGURE 4. The strange attractors with $k_s = k_m = 2$.

B. DISCRETIZATION OF CHAOTIC SYSTEMS

To promote the stability and reduce the implementation cost of the RNGs, we will employ discrete chaotic systems for our design. Consequently, in the following, we outline how to discretize a continuous chaotic system while preserving the chaotic behavior. For continuous nonlinear chaotic systems, their dynamical equations can be generally described as follows:

$$\dot{y}(t) = Ay(t) + Bg(y(t)) \tag{8}$$

where $y(t) \in R^{n\times 1}$ is the state vector, $A \in R^{n\times n}$, $B \in R^{n\times m}$ are system matrices and $g(y(t)) \in R^{m\times 1}$ is the nonlinear vector. Then the corresponding discrete type of system (8) can be described as

$$y((k+1)T) = Gy(kT) + Hg(y(kT)) \tag{9}$$

where $G = e^{AT}$, T is the sampling time and $H = [G - I_n]A^{-1}B$ [16]. In the following, for simplicity, we will use the proposed adjustable chaotic system (7) for our design, however, the results can be easily modified and available for other chaotic systems. Obviously, (7) with the modulating parameters $(k_s, k_m) = (2, 1)$ can be rearranged as the form of (8) with matrices A and B shown as below.

$$A = \begin{bmatrix} -32 & 32 & 0 & 0 \\ 90 & -4 & 0 & 90 \\ 0 & 0 & -8 & 0 \\ 0 & 32 & 0 & -32 \end{bmatrix}; B = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -2 & -2 & 0 & 0 \\ 0 & 0 & 2 & 2 \\ -0.1 & 0 & 0 & 0 \end{bmatrix}$$

Now with sampling time $T = 0.0001$, by using $G = e^{AT}$ and $H = [G - I_n]A^{-1}B$, we can obtain the corresponding matrices G and H shown below.

$$G = \begin{bmatrix} 0.9699 & 0.0315 & 0 & 0.0014 \\ 0.0885 & 0.9989 & 0 & 0.0885 \\ 0 & 0 & 0.992 & 0 \\ 0.0014 & 0.0315 & 0 & 0.9699 \end{bmatrix};$$

$$H = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -0.002 & -0.002 & 0 & 0 \\ 0 & 0 & 0.002 & 0.002 \\ -0.0001 & 0 & 0 & 0 \end{bmatrix}$$

Therefore, the corresponding discrete type of system (7) with the parameters $(a, b, c, d, e, f, g) = (16, 45, -2, 45, 16, -4, 16)$ and $(k_s, k_m) = (2, 1)$ can be obtained as

$$\begin{aligned} y_1(k+1) &= 0.9699y_1(k) + 0.0315y_2(k) + 0.0014y_3(k) \\ y_2(k+1) &= 0.0845y_1(k) + 0.9989y_2(k) + 0.0845y_4(k) \\ &\quad - 0.002y_1(k)y_3(k) - 0.002y_3(k)y_4(k) \\ y_3(k+1) &= 0.992y_3(k) + 0.002y_1(k)y_2(k) + 0.002y_2(k)y_4(k) \\ y_4(k+1) &= 0.001y_1(k) + 0.0315y_2(k) + 0.9699y_4(k) \\ &\quad - 0.0001y_1(k)y_3(k) \end{aligned} \tag{10}$$

In above equation (10), for simplicity, the sampling time T has been omitted. We simulate the responses of strange attractors for the continuous chaotic system (7) with $(k_s, k_m) = (2, 1)$ and the corresponding discrete system (10) with the same initial conditions. From the simulation results shown in Figure 5, it reveals that after the continuous chaotic system (7) can be discretized and its corresponding discrete system (10) still preserve the chaotic behavior.

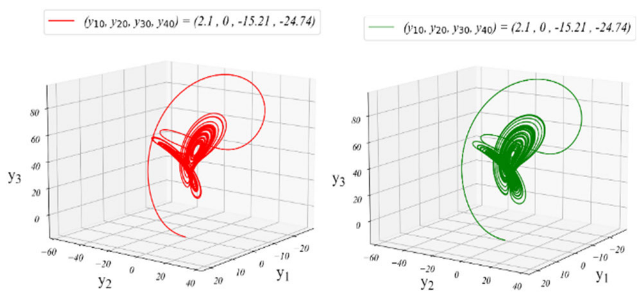

FIGURE 5. Strange attractors (a) the continuous chaotic system (b) the corresponding discrete chaotic system.

As illustrated above, the method described can also be systematically applied to obtain the corresponding discrete system when the system (7) is with different modulated parameters.

C. EXTREMUM CODING AND THE DESIGN OF RNGs

In this paper, the design of the proposed RNG is based on the dynamic behavior of chaotic systems. As mentioned above, chaotic systems exhibit highly sensitive dependence on initial conditions, a phenomenon known as the ''butterfly effect''. In the proposed RNG design, as shown in Figure 1, by utilizing the butterfly effect, we propose an innovative chaotic-based extremum coding approach to generate a random extremum-coded sequence (RECS). Using this RECS for updating modulation parameters of chaotic system (7) notably can enhance the randomness of the sequences generated by the RNGs. Additionally, by incorporating SHA3-256 (Secure Hash Algorithm 3) hashing calculations, we can generate a true RNG. The following sections provide the detailed explanation for extremum-coded modulation mechanism in Figure 1.

1) EXTREMUM-CODED MODULATION MECHANISM

In order to further promote the randomness quality of the dynamic states, we propose an extremum-coded modulation mechanism shown in Figure 6. Figure 6 includes N sets of modulation parameters $k_{si}, k_{mi}, i = 1, 2 \ldots, N$, a multiplexer, and an adjustable chaotic system and a RECS. The RECS is the input of the multiplexer to randomly select different modulation parameters and then modulate the speed and amplitude of state responses of the adjustable chaotic system. The generation rules of the RECS are given in Figure 7. When extremum values are dynamically and randomly generated by the states of the modulated chaotic system, it generates a RECS by the extremum coding rules in Figure 7. The unpredictable random state responses of chaotic systems are used to formulate the rule of extremum coding for generating an unpredictable RECS, and then use this unpredictable RECS as the input of the multiplexer in Figure 6. Therefore, one of N modulation parameters in Figure 6 can be randomly selected to modulate the dynamics of the adjustable chaotic system according to the n-least significant bits satisfying $(2^n \geq N)$ of the RECS. Such design can make the RECS dynamic and random and the updating timing cannot be predicted, which can effectively increase the difficulty of cracking. The proposed extremum coding can be easily realized by using a microcontroller only with programming and explained as below. For a random state response of the chaotic system, we record the peak values of the relative maximum, $P_i, i = 1, 2, \ldots \infty$ and trough values of the relative minimum, $T_i, i = 1, 2, \ldots \infty$ and compare the values of the peaks or troughs, respectively. If $P_{i+1} > P_i$ or $T_{i+1} > T_i$, we store 1, otherwise we store 0 as the new least significant bit (LSB) of RECS. Then using this rule of extremum coding generates the subsequent sequence and the generation flowchart of the RECS is shown in Figure 7.

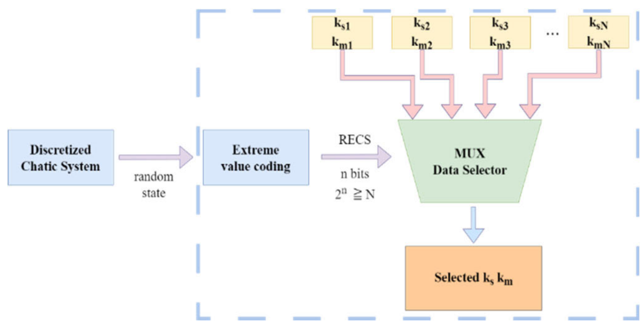

FIGURE 6. The structure of extremum-coded modulation mechanism.

2) REALIZATION OF THE CHAOTIC EXTREMUM-CODED RNGs

With discussions above, we have successfully addressed the various technical components outlined in the architecture of the true RNG presented in Figure 1. This includes the design of adjustable chaotic systems, discretization of

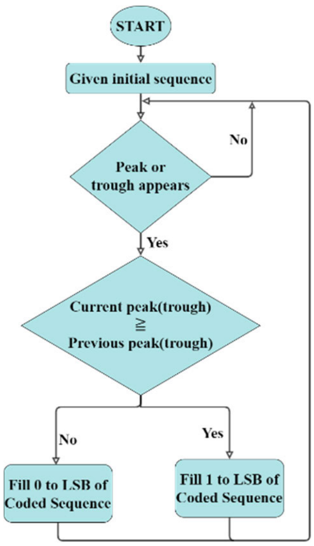

FIGURE 7. The generation of the extremum-coded sequence.

continuous chaotic systems, and the extremum coding mechanism. By combining these results with SHA-256, we can finalize the implementation of the true RNG as depicted in Figure 1.

III. SECURITY TEST OF CHAOTIC EXTREMUM-CODED RNGs

To highlight the contributions of this design, we subject the generated random sequences from this RNG to rigorous security testing. The statistical methods including NIST, Diehard, and ENT are employed for evaluation. Furthermore, we compare the results with recent works to validate the performance in generating high-quality random sequences. For conducting security testing on the proposed RNG, four sets of modulation parameters are selected ($N = 4$). Therefore, the two least significant bits (LSBs) are employed in the extremum-coded modulation mechanism shown in Figure 6. The generation mechanism of modulation parameters is illustrated in Figure 8. In Figure 8, $x(k) = \left[x_1(k)\ x_2(k)\ x_3(k)\ x_4(k)\right]^T$ is

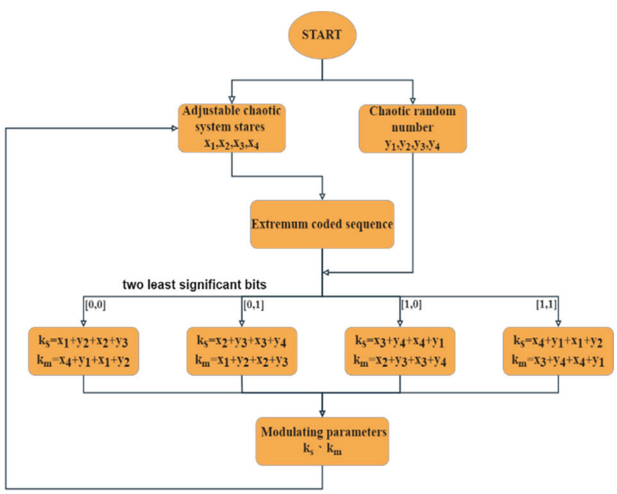

FIGURE 8. The generation mechanism of modulation parameters.

the state vector of the chaotic system. In order to increase the randomness of parameter modulation, we choose another random vector $y(k) = x(k-5) = \left[y_1(k)\ y_2(k)\ y_3(k)\ y_4(k)\right]^T$. When the extremum code sequence updates, new modulation parameters are calculated using the rules suggested in Figure 8. Obviously, in this generation mechanism, the random characteristics of chaotic systems are used to produce modulation parameters, which are subsequently applied to the proposed adjustable chaotic system. Through this design approach, owing to the unpredictable property of chaotic system states, unpredictable modulation parameters can be derived to modulate the chaotic system and the security can be surely promoted. Furthermore, such a design provides high flexibility, allowing for the alteration of modulation parameter generation methods by the designer. The initial values were set as $x_1(0) = 2.1, x_2(0) = 0, x_3(0) = -15.21, x_4(0) = -24.74$ and the initial modulation parameters are $k_s = k_m = 1$, while the initial extremum-coded sequence was established as 01. Following the initiation of system operations, the extremum-coded sequence will be generated according to the encoding scheme as shown in Figure 6, Simultaneously, the two least significant bits will be dynamically updated in a random manner. As indicated by Figure 7, modulation parameters will also undergo random and dynamic updates. Consequently, these random numbers pass through the function of SHA3-256, the designed RNG shown in Figure 1 can generate a randomly binary sequence.

A. STATISTICAL TESTS

In this section, we evaluate the randomness and the security of the proposed RNG by using NIST, Diehard, and ENT statistical tests. The test results are given in Tables 1-3.

TABLE 1. NIST SP 800-22 test results.

Statistical tests	P-value (N=7,360,000bytes)
Frequency	0.939748
Block Frequency	0.958100
Runs	0.741419
Longest Run	0.959669
Rank	0.988893
FFT	0.212713
Non-overlapping Template	0.999418
Overlapping Template	0.725814
Universal	0.970870
Linear Complexity	0.163819
Serial	0.413638
Approximate Entropy	0.831140
Cumulative Sums	0.583478
Random Excursion	0.158152
Random Excursion Variant	0.075151
Sum	9.722021

TABLE 2. Diehard battery test results.

Statistical tests	P-value (N=7,360,000bytes)
Birthdays	0.84235876
Operm5	0.66384363
Rank 32 × 32	0.75987843
Rank 6 × 8	0.42508951
Bitstream	0.95354224
Diehard Opso	0.85522428
Diehard Oqso	0.38695243
DNA	0.65714627
Count 1s-str	0.79351879
Count 1s-byt	0.95578432
Parking Lot	0.10526018
2-d Sphere	0.54022356
3-d Sphere	0.9624542
Squeeze	0.47395784
Diehard Sums	0.20364165
Diehard Runs	0.90273912
Diehard Craps	0.92245721
Marsaglia Tsang GCD	0.66085519
STS Monobit	0.6865481
STS Runs	0.8594987
STS Serial	0.99332692
RGB Bit-Dist	0.9080114
RGB Minimum Distance	0.52417951
RGB Permutations	0.98031507
RGB Lagged Sum	0.99088861
RGB Kstest	0.71662356
DAB Bytedistrib	0.87148689
DAB DCT	0.04537118
DAB Filltree	0.82130046
DAB Filltree2	0.37452769
DAB Monobit-2	0.34838018
Sum	21.1853859

Table 1 shows the NIST SP 800-22 [17] test results. NIST evaluation standard contains 15 items and the test result for every item is called p-value. When p-value is greater than 0.01, it passes the test item. With larger p-value, it means the better randomness of the tested random data. Therefore, from p-values obtained in Table 1, it reveals that the proposed random number not only passes the NIST test but also possesses good randomness. Table 2 shows the Diehard Battery Test [18] results. In this test, the p-value is also used to evaluate the test results. Therefore, from p-values obtained in Table 2, we can also conclude that the proposed random number possesses good randomness. Next, we use the ENT test standard to analyze the quality of random numbers and the test results are shown in Table 3. In the ENT test standard, entropy analysis evaluates the randomness by calculating the entropy value of the random sequence. When the entropy value is closer to 8, it means that the randomness is better. The x^2 Distribution is also called Chi-Squared Test, and the ideal result of the percentage should be between 10% and 90%. In the test of Arithmetic mean value, when the calculated value exceeds 127.5, it means that the randomness of the RNG is better. Monte Carlo value for Pi is the Monte Carlo test. the closer the test result is to π, the better randomness of data is. The test of Serial Correlation coefficients calculates the correlation of random numbers. The closer the test result is to 0, the less correlation between random numbers and the better the quality of random numbers. Therefore, according to the result in Table 3, it shows that the random number sequence generated by the proposed RNG in this study has good randomness quality.

TABLE 3. ENT test results.

Statistical tests	P-value(N=7,360,000bytes) Results
Entropy	7.999977 bits per byte
Optimum compression	7360000 byte file by 0%.
x^2Distribution	For 7360000 samples is 236.81, and randomly would exceed this value 78.69% of the times.
Arithmetic mean value	127.5073 (random=127.5)
Monte Carlo value for Pi	3.141536490 (error 0.00%).
Serial Correlation coefficients	0.000494(totally uncorrelated = 0.0)

B. COMPARISONS WITH RESULTS OF RECENT PUBLISHED WORKS

To emphasize the contribution of the RNG proposed in this paper, we compare our results with the published papers. We perform the NIST, Diehard, and ENT tests according to the same conditions, respectively, set by each paper.

1) COMPARISONS USING NIST TEST

For the NIST test in papers [5], [6], [7], the tested random sequence is with 10^6 bits, and the bit-stream length is set to 100, and the comparison results are given in Table 4. For the papers [8], [9], [10], the tested random number is also with 10^6 bits, but the bit-stream length is set to 1000, and the comparison results are given in Table 5. According to the comparison results, under the individual test conditions, we can see that the proposed RNG in this paper have better results in most test items. Because each RNG designed has its own advantages, our results might not be the best in all test items. However, judging from the sum of the outcomes in all test items, the RNG we proposed is better than others and it can also show that our proposed RNG has better randomness characteristics.

TABLE 4. NIST test with bit-stream length 100.

Statistical tests	P-value(N=10^6bytes) bitstream: 100							
	Charalampidis et al.[5]		Xu et al. [6]		Zhao et al.[7]		This Paper	
	P-value	Prop.	P-value	Prop.	P-value	Result	P-value	Prop.
Frequency	0.883171	99/100	0.678686	99/100	0.8228	Pass	0.534146	100/100
Block Frequency	0.115387	100/100	0.678686	99/100	0.8279	Pass	0.798139	100/100
Cumulative Sums	0.162606	99/100	0.115387	100/100	0.9877	Pass	0.574903	100/100
Runs	0.834308	100/100	0.334538	100/100	0.9155	Pass	0.145326	99/100
Longest Run	0.026948	99/100	0.554420	99/100	0.9398	Pass	0.999438	99/100
Rank	0.678686	100/100	0.514124	100/100	0.0235	Pass	0.350485	99/100
FFT	0.090936	100/100	0.474986	98/100	0.2701	Pass	0.419021	99/100
NonOverlapping Template	0.023545	100/100	0.987896	100/100	0.8840	Pass	0.987896	98/100
Overlapping Template	0.699313	98/100	0.162606	100/100	0.6767	Pass	0.798139	100/100
Universal	0.798139	99/100	0.129620	100/100	0.5964	Pass	0.759756	99/100
Approximate Entropy	0.383827	100/100	0.045675	100/100	0.5138	Pass	0.262249	99/100
Random Excursions	0.262249	58/58	0.900104	100/100	0.1317	Pass	0.484646	67/67
Random Excursions Variant	0.013569	58/58	0.900104	99/100	0.1812	Pass	0.788728	66/67
Serial	0.834308	99/100	0.637119	100/100	0.0404	Pass	0.637119	98/100
Linear Complexity	0.334538	100/100	0.897763	99/100	0.9668	Pass	0.383827	99/100
Sum	**6.14153**		**8.011714**		**8.7783**		**8.923818**	

TABLE 5. NIST test with bit-stream length 1000.

Statistical tests	P-value(N=10^6bytes) bitstream: 1000						
	Ma et al. [8]		Yang et al. [9]		Zheng et al. [10]	This Paper	
	P-value	Prop.	P-value	Prop.	P-value	P-value	Prop.
Frequency	0.3823	991/100	0.8037	984/1000	0.859637	0.979226	994/1000
Block Frequency	0.3121	981/1000	0.0235	981/1000	0.271619	0.991468	989/1000
Cumulative Sums	0.3341	994/1000	0.5859	983/1000	0.688195	0.717714	993/1000
Runs	0.6874	995/1000	0.8000	994/1000	0.375313	0.616305	989/1000
Longest Run	0.2864	984/1000	0.9421	983/1000	0.250558	0.849708	989/1000
Rank	0.4203	983/1000	0.2393	991/1000	0.293952	0.006566	988/1000
FFT	0.2374	991/1000	0.064	991/1000	0.377007	0.122325	992/1000
NonOverlapping Template	0.9342	992/1000	0.5205	990/1000	0.49619	0.992381	990/1000
Overlapping Template	0.2932	986/1000	0.1326	987/1000	0.641284	0.705466	988/1000
Universal	0.7068	981/1000	0.7034	993/1000	0.769527	0.974370	989/1000
Approximate Entropy	0.2381	983/1000	0.4101	989/1000	0.949278	0.006472	988/1000
Random Excursions	0.7231	638/642	0.3931	987/1000	0.456696	0.689019	119/120
Random Excursions Variant	0.7832	638/642	0.4722	990/1000	0.507124	0.922036	122/122
Serial Test	0.2387	981/1000	0.249	986/1000	0.214198	0.676615	995/1000
Linear Complexity	0.3834	995/1000	0.5141	991/1000	0.544254	0.605916	987/1000
Sum	**6.9607**		**6.8535**		**7.694832**	**9.855587**	

2) COMPARISONS USING DIEHARD TEST

For Diehard tests, we use the same data size and related setting conditions in papers [9], [11], [19] to conduct tests, and the results are shown in Table 6. Due to the tests in the compared papers, not every sub-item has a test outcome, so we use the average value of each item to observe and compare. Observing the test results in Table 6, it is observed that this paper has better good performance in most sub-item tests. From the average p-value, it is also better than the results of other papers. Therefore, it can

TABLE 6. Results of diehard test.

Statistical tests	Yang et al [9]	Kamadi et al. [11]	Kamadi et al. with SHA [11]	Paul et al. with BluXor [19]	Paul et al. with MPCG [19]	This Paper
Birthdays	0.5556	0.634	0.387	0.89915249	0.64815728	0.91480126
Operm5	0.3344	0.986	0.799	0.37516036	0.8380907	0.73403751
Rank 32 × 32	0.8186	0.897	0.364	0.96345486	0.90655901	0.67728486
Rank 6 × 8	0.0106	0.345	0.998	0.44332652	0.04603721	0.39865458
Bitstream	0.7802	0.104	0.962	0.98372427	0.64736979	0.6789483
Diehard Opso	0.1997	0.223	0.942	0.95581938	0.92416133	0.56777858
Diehard Oqso		0.737	0.825	0.26745376	0.31318059	0.72915327
DNA	0.0444	0.424	0.807	0.38033601	0.1606891	0.25902449
Count 1s-str	0.0392	0.249	0.095	0.96724261	0.78825851	0.81215304
Count 1s-byt	0.8934	0.318	0.719	0.22539871	0.70303283	0.99418896
Parking Lot	0.2546	0.875	0.774	0.69671967	0.06102235	0.92463763
2-d Sphere	0.9606	0.702	0.503	0.21652778	0.36844792	0.21731157
3-d Sphere	0.7178	0.472	0.942	0.93923026	0.99020443	0.57157328
Squeeze	0.0013	0.98	0.788	0.12831634	0.8215206	0.31284153
Diehard Sums	0.0366	0.042	0.03	0.44893896	0.00339542	0.6389568
Diehard Runs Up	0.0218	0.127	0.63	0.3569768	0.59319437	0.18408594
Diehard Runs Down	0.7207					0.66556938
Diehard Craps	0.0147	0.995	0.397	0.12462753	0.57860814	0.43201711
Marsaglia Tsang GCD				0.89999546	0.36775462	0.85140498
STS Monobit				0.12999497	0.85296739	0.55234949
STS Runs				0.97131711	0.06398866	0.86178132
STS Serial						0.9891612
RGB Bit-Dist				0.43153706	0.33869794	0.99269946
RGB Minimum Distance				0.13589439	0.24214232	0.81624121
RGB Permutations				0.01729007	0.24296868	0.77967143
RGB Lagged Sum				0.09780326	0.45691115	0.97989076
RGB Kstest				0.53462563	0.68976679	0.8887988
DAB Bytedistrib				0.05708488	0.11194593	0.72978658
DAB DCT				0.07248676	0.64705683	0.04220327
DAB Filltree				0.45674104	0.45756793	0.90196605
DAB Monobit-2				0.24737607	0.74744103	0.22497554
Sum	6.4042	9.11	10.962	13.42455301	14.61113885	20.3239482
Average	**0.3767**	**0.536**	**0.6448**	**0.462915621**	**0.503832374**	**0.65561123**

TABLE 7. ENT test.

Statistical tests	Results				
	Camara et al. [12]	Cubic Maps [13]	Ricker's P. Model. [13]	Hayati et al. [20]	This paper
Entropy	7.999985	7.9997	7.9998	1.000000	7.999994
Optimum compression	0%	-	-	-	0%
x^2Distribution	244.55 (66.98%)	287.474	263.369	0.67	269.38 (25.64%)
Arithmetic mean value	127.5467	127.4868	127.4565	0.5000	127.5005
Monte Carlo value for Pi	3.141160791 (error 0.01%)	3.1459	3.1495	3.1415487	3.142428800 (error 0.03%)
Serial Correlation coefficients	0.000331	-0.0011	-0.0005	0.000002	-0.000216

conclude that the RNG proposed in this paper can produce better random number sequences than those in [9], [11], and [19].

3) COMPARISONS USING ENT TEST

Next, we use the ENT test standard to analyze and compare the randomness quality of the proposed RNG with the works in the literature [12], [13], [20], and the test results are shown in Table 7. According to the test results in Table 7 and evaluation criteria mentioned in Section III-A, we can draw the following conclusions. The entropy value of our proposed RNG is better than the results of literature [12], [13], [20]. Also the proposed RNG passes the x^2 Distribution test and it obtains better performances than literature [13], [20] for Arithmetic mean value and Monte Carlo test. As for the Serial Correlation coefficients test, the test result is better literature [12], [13]. Based on the above analysis and comparison, it can be shown that the random number sequence generated by the RNG proposed in this study has good randomness quality.

IV. CONCLUSION

In this paper, we propose an innovative design for a chaos-based random number generator, which successfully introduces the modulation parameters to adjust the dynamics of the chaotic systems. By discretizing the continuous modulated chaotic system, a corresponding discrete chaotic system can be obtained, which can reduce the cost of circuit implementation and maintenance, and also enhance the stability of the RNG. At the same time, we also propose a novel chaos-based extremum encoding method to generate random extremum encoding sequences, and integrate this sequence to complete the design of RNG. In order to highlight the contribution of this paper, we have compared with the recent literature through statistical tests NIST, Diehard and ENT statistical tests. The comparison results also fully demonstrate that the chaotic extremum coding RNG proposed in this paper can exhibit superior performance with a higher level of randomness.

REFERENCES

[1] K. Seyhan and S. Akleylek, "Classification of random number generator applications in IoT: A comprehensive taxonomy," *J. Inf. Secur. Appl.*, vol. 71, Dec. 2022, Art. no. 103365.

[2] T. L. Liao, P. Y. Wan, and J.-J. Yan, "Design and synchronization of chaos-based true random number generators and its FPGA implementation," *IEEE Access*, vol. 10, pp. 8279–8286, 2022.

[3] I. Koyuncu and A. Turan Özcerit, "The design and realization of a new high speed FPGA-based chaotic true random number generator," *Comput. Electr. Eng.*, vol. 58, pp. 203–214, Feb. 2017.

[4] I. Koyuncu, M. Tuna, I. Pehlivan, C. B. Fidan, and M. Alçn, "Design, FPGA implementation and statistical analysis of chaos-ring based dual entropy core true random number generator," *Anal. Integr. Circuits Signal Process.*, vol. 102, no. 2, pp. 445–456, Feb. 2020.

[5] N. Charalampidis, C. Volos, L. Moysis, H. E. Nistazakis, and I. Stouboulos, "A novel piecewise chaotic map for image encryption," in *Proc. 11th Int. Conf. Modern Circuits Syst. Technol. (MOCAST)*, Jun. 2022, pp. 1–4.

[6] Y. Xu and M. Tang, "Color image encryption algorithm using DNA encoding and fuzzy single neurons," *IEEE Access*, vol. 10, pp. 127770–127782, 2022.

[7] H. Zhao, S. Xie, J. Zhang, and T. Wu, "A dynamic block image encryption using variable-length secret key and modified Henon map," *Optik*, vol. 230, Mar. 2021, Art. no. 166307.

[8] Y. Ma, T. Chen, J. Lin, J. Yang, and J. Jing, "Entropy estimation for ADC sampling-based true random number generators," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 11, pp. 2887–2900, Nov. 2019.

[9] Z. Yang, Y. Liu, Y. Wu, Y. Qi, F. Ren, and S. Li, "A high speed pseudo-random bit generator driven by 2D-discrete hyperchaos," *Chaos, Solitons Fractals*, vol. 167, Feb. 2023, Art. no. 113039.

[10] J. Zheng and H. Hu, "A highly secure stream cipher based on analog-digital hybrid chaotic system," *Inf. Sci.*, vol. 587, pp. 226–246, Mar. 2022.

[11] A. Kamadi and Z. Abbas, "Implementation of TRNG with SHA-3 for hardware security," *Microelectron. J.*, vol. 123, May 2022, Art. no. 105410.

[12] C. Camara, H. Martín, P. Peris-Lopez, and L. Entrena, "A true random number generator based on gait data for the internet of you," *IEEE Access*, vol. 8, pp. 71642–71651, 2020.

[13] H. E. Kiran, A. Akgul, O. Yildiz, and E. Deniz, "Lightweight encryption mechanism with discrete-time chaotic maps for Internet of Robotic Things," *Integration*, vol. 93, Nov. 2023, Art. no. 102047.

[14] X.-J. Tong, M. Zhang, Z. Wang, Y. Liu, H. Xu, and J. Ma, "A fast encryption algorithm of color image based on four-dimensional chaotic system," *J. Vis. Commun. Image Represent.*, vol. 33, pp. 219–234, Nov. 2015.

[15] G. F. Simmon, *Calculus With Analytic Geometry*. New York, NY, USA: McGraw-Hill, 2007.

[16] K. D. Young, V. I. Utkin, and U. Ozguner, "A control engineer's guide to sliding mode control," *IEEE Trans. Control Syst. Technol.*, vol. 7, no. 3, pp. 328–342, May 1999.

[17] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, and J. Dray, *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*. Gaithersburg, MD, USA: NIST Special, 2010.

[18] G. Marsaglia, "Diehard: A battery of tests of randomness," Dept. Statist., Florida State Univ., 1996. [Online]. Available: http://stat.fsu.edu/pub/diehard/

[19] B. Paul, G. Trivedi, P. Jan, and Z. Nemec, "Efficient PRNG design and implementation for various high throughput cryptographic and low power security applications," in *Proc. 29th Int. Conf. Radioelektronika*, Pardubice, Czech Republic, Apr. 2019, pp. 1–6.

[20] N. Hayati, S. Windarta, M. Suryanegara, B. Pranggono, and K. Ramli, "A novel session key update scheme for LoRaWAN," *IEEE Access*, vol. 10, pp. 89696–89713, 2022.

SHUNSUKE ARAKI received the B.E., M.E., and Dr.Eng. degrees in computer science and system engineering from the Kyushu Institute of Technology, Japan, in 1996, 1998, and 2003, respectively. Since 2000, he has been a Research Associate with the Kyushu Institute of Technology, and become an Associate Professor, in 2018. His main research interests include pseudorandom number generators, digital signatures, and IoT secure protocols.

JI-HAN WU received the B.S. degree in Shu-Te University of Technology, Taiwan, in 2021. He is currently pursuing the master's degree with the Department of Electronic Engineering Science, National Chin-Yi University of Technology, Taichung, Taiwan. His main research interests include chaotic systems, random number generator design, and the Internet of Things application.

JUN-JUH YAN received the B.S. degree in electrical engineering from the National Cheng Kung University, Taiwan, in 1987, the M.S. degree in electrical engineering from the National Central University, Taiwan, in 1992, and the Ph.D. degree in electrical engineering from the National Cheng Kung University, in 1998. Currently, he is a Professor with the Department of Electronic Engineering, National Chin-Yi University of Technology, Taichung, Taiwan. His main research interests include multi-robot dynamic systems, chaotic systems, neural networks, variable-structure control systems, and adaptive control.