CHAOS TEORY AND APPLICATIONS
Volume 01, Number 1, Pages 1–18

DESIGN OF AN INTERFACE FOR RANDOM NUMBER GENERATORS BASED ON INTEGER AND FRACTIONAL ORDER CHAOTIC SYSTEMS

AKIF AKGUL, COSKUN ARSLAN, AND BURAK ARICIOGLU

Abstract. Chaos is one of the most topical subjects in the literature and is applied to various number of different fields such as system identification, optimization, brain functions identification, secure communication, encryption and random number generation. In this work, a user interface is designed for generation of random numbers based on fractional and integer order chaotic systems. In order to evaluate the randomness of the generated numbers NIST-800-22 and ENT statistical tests are performed. The design interface provides the users a wide range of options. Moreover, the interface is also implemented on a microcomputer that the generated random number can be used in mobile applications. In this way, random numbers that have great importance on applications such as cryptographic and secure communication systems, statistical samplings, computer simulations and designs based on randomness are generated with different ways.

Keywords: Chaos, chaotic systems, fractional order chaotic systems, random number generators, interface design.

1. Introduction

Nowadays smart systems that can be used in house and building automation, industry, energy, medical, transportation and communication system and the security of data transfer of these smart systems have become very important. Multimedia data like text, image and audio can be shared all over the globe very easily but this system has brought some series concern on the shared data security. It is quite possible for the third parties to access to the data and even to change it. In order to prevent third parties to access the data some security tools and technologies have been developed. Among these tools random number generators (RNGs) are one of the most important ones. The RNGs are widely employed in encryption applications as well as in applications such as numerical analysis, game theory, statistics and simulation and so on. Chaos based RNG designs have become a very topical subjects in the literature. Chaos can be simply defined as a discipline that studies order in disorder. The main features of the chaotic signs are sensitivity to the initial conditions, irregularity in time domain, having huge number of harmonics and broader power spectrum like noise [1].

In nowadays chaotic systems are used in many different application fields. Some of these are nonlinear deterministic estimation, biomedical applications, chaotic encryption and stenography, modelling of nonlinear systems, nonlinear filtering, dynamic data compression and coding, weather forecast and random number generation and so on [2-4]. In this study, a design of a user-friendly interface for chaos based random number generation is aimed. There are two types of RNGs, namely, true random number generators (TRNGs) and pseudo random number generators (PRNGs). In RNG based applications, PRNGs are preferred for repeatable random number sequences while TRNGs are preferred for high unpredictability. All the generated numbers with the designed interface are pseudo random numbers. Moreover, the interface will allow generation of random numbers with different methods or chaotic systems. In the literature, there are numerous studies on chaos based random number generation. In some of these studies, discrete chaotic systems [5-7], continuous systems [8-10] or fractional order chaotic systems [11-13] are employed. There are also studies on chaos based TRNGs [6,14-17] and PRNGs [5,7-12]. The setup of the paper is as follows: the chaotic systems used in random number generation are described in Section 2; chaos based random number generation is discussed in Section 3; the designed interface for chaos based random number generation is mentioned in Section 4; the statistical results of the generated random numbers are given in Section 5 and finally conclusion is given in the last section.

2. The Chaotic Systems Used in the Study

In this section, the chaotic systems which are used in the interface to generate random numbers are mentioned. The chaotic systems which are used in the study as follows:

Lorenz chaotic system [18]

$$\begin{aligned} \dot{x} &= a(y-x) \\ \dot{y} &= rx - y - xz \\ \dot{z} &= xy - bz \end{aligned} \tag{2.1}$$

Rössler chaotic system [19]

$$\begin{aligned} \dot{x} &= -(y+z) \\ \dot{y} &= x + ay \\ \dot{z} &= b + z(x-c) \end{aligned} \tag{2.2}$$

Van Der Pol chaotic system [20]

$$\begin{aligned} \dot{x} &= y \\ \dot{y} &= a(1-x^2)y - x^3 + \cos(ct) \end{aligned} \tag{2.3}$$

Aizawa chaotic system

$$\begin{aligned} \dot{x} &= (z-\beta)x - \delta y \\ \dot{y} &= (\delta x) + (z-\beta)y \\ \dot{z} &= \gamma + \alpha z - \frac{z^3}{3} - (x^2+y^2)(1+\varepsilon z) + (\zeta z x^3) \end{aligned} \tag{2.4}$$

Pehlivan G chaotic system

$$\begin{aligned}\dot{x} &= y - x\\ \dot{y} &= ay - xz\\ \dot{z} &= xy - a\end{aligned} \tag{2.5}$$

Chen chaotic system [21]

$$\begin{aligned}\dot{x} &= a(y - x)\\ \dot{y} &= -xz + (c - a)x + cy\\ \dot{z} &= xy - bz\end{aligned} \tag{2.6}$$

Labyrinth chaotic system [22]

$$\begin{aligned}\dot{x} &= \sin(y) - bx\\ \dot{y} &= -\sin(z) - by\\ \dot{z} &= \sin(x) - bz\end{aligned} \tag{2.7}$$

Rucklidge chaotic system [23]

$$\begin{aligned}\dot{x} &= Kx + Ly - yz\\ \dot{y} &= x\\ \dot{z} &= z + y^2\end{aligned} \tag{2.8}$$

Rikitake chaotic system [24-26]

$$\begin{aligned}\dot{x} &= -\mu x + zy\\ \dot{y} &= -\mu x + (z - a)\\ \dot{z} &= 1 - xy\end{aligned} \tag{2.9}$$

A chaotic system with golden proportion equilibria [27]

$$\begin{aligned}\dot{x} &= y - x - az\\ \dot{y} &= xz - x\\ \dot{z} &= -xy - y + b\end{aligned} \tag{2.10}$$

3. Chaos Based Random Number Generation

In the study, random numbers are generated by numerically solving the chaotic system equations and then the obtained numerical solutions which are in decimal format are converted into the binary format by using three different methods described in this section. After the conversion process a certain number of least significant bits (LSBs) are selected for random bit series.

The numerical method for the integer order chaotic systems is Runge-Kutta 4 (RK4) method and for the fractional order chaotic systems is Grnwald-Letnikov (GL) method.

3.1. **Numerical Solution of Fractional Order Chaotic System.** Nowadays fractional order integral, differential and integro-differential equations are usually employed in science fields such as physics, chemistry, electric and electronics, thermodynamic and control theory. Fractional order systems and their applications have become a topical subject in the last decades [28-29].

In the literature symbol D is used as integro-differential operator and it can be used to represent integral or differential operations. In Eq. (3.1), it is shown cases where the integro-differential operator represents integral or differential operation. In Eq. (3.1) a positive value of a represents differential while a negative value of a represents integral [30,31].

$$ {}_aD_x^\alpha = \begin{cases} \dfrac{d^a}{dx^a}, & \alpha > 0 \\ 1, & \alpha = 0 \\ \displaystyle\int_a^x (d\tau)^{-\alpha}, & \alpha < 0 \end{cases} \tag{3.1} $$

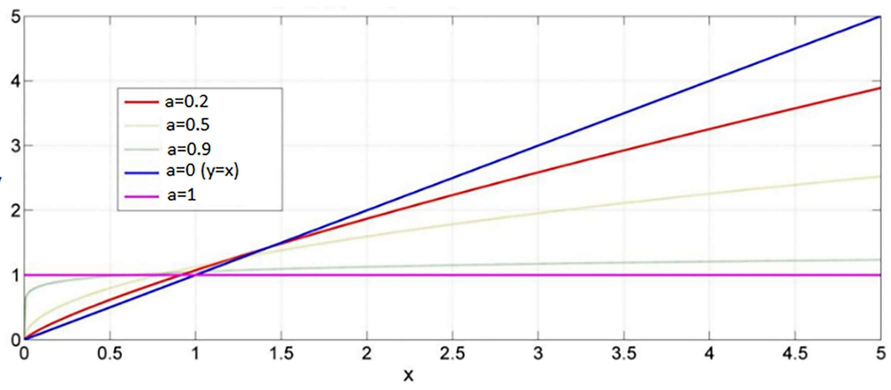


FIGURE 1. Some different order derivatives of f(x)=x [30].

To solve fractional order chaotic system Grnwald Letnikov (GL) definition is employed and it is given in Equation (3.2).

$$ {}_aD_t^\alpha f(t) = \lim_{h\to 0} h^{-\alpha} \sum_{j=0}^{\frac{t-a}{h}} (-1)^j \binom{\alpha}{j} f(t-jh) \tag{3.2} $$

where $\binom{\alpha}{j}$ is the binomial expansion coefficients and

$$ \binom{\alpha}{j} = \frac{\alpha!}{j!(a-j)!} = \frac{\Gamma(\alpha+1)}{\Gamma(j+1)\Gamma(a-j+1)} \tag{3.3} $$

$\Gamma(n)$ is the Gamma function.

$$ \Gamma(n) = (n-1)! \tag{3.4} $$

$$\Gamma(z) = \int_0^\infty t^{z-1} e^{-t} dt \tag{3.5}$$

For numeric solution of the fractional order derivatives the below expression can be written:

$$_{k-\frac{L_m}{h}}D^q_{t_k} f(t) \approx \lim_{h\to 0} h^{-q} \sum_{j=0}^{\frac{t-a}{h}} (-1)^j \binom{q}{j} f(t_k - j) \tag{3.6}$$

where L_m is the memory length and h is the step size. The binomial expansion coefficients can be expressed as:

$$c_j^{(q)} = (-1)^j \binom{q}{j}, \qquad c_0^{(q)} = 1, \qquad c_j^{(q)} = \left(1 - \frac{1+q}{j}\right) c_{j-1}^{(q)}\ ,\ (j = 0, 1, 2 \dots) \tag{3.7}$$

General numeric solution of fractional order system can be written as:

$$_aD_t^q y(t) = f(y(t), t) \tag{3.8}$$

$$y(t_k) = f(y(t), t) h^q - \sum_{j=v}^{k} c_j^{(q)} y(t_{k-j}) \tag{3.9}$$

3.2. **Binary number conversion techniques.** In the studies, time series of the chaotic signals are obtained via numerical algorithms. These obtained numbers are in decimal format and they are converted into binary format. In the binary bit series, the randomness is usually higher at the least significant bits (LSBs). Hence, for the random number generation process the LSBs are selected. How many LSBs are selected can be set in the interface by the user as described in the next section.

For the binary conversion process, there are three different conversion methods in the designed user interface.

Method 1 (Floating point): In this method, the decimal numbers obtained from the time series are converted into 32-bit IEEE 754 floating point number format.

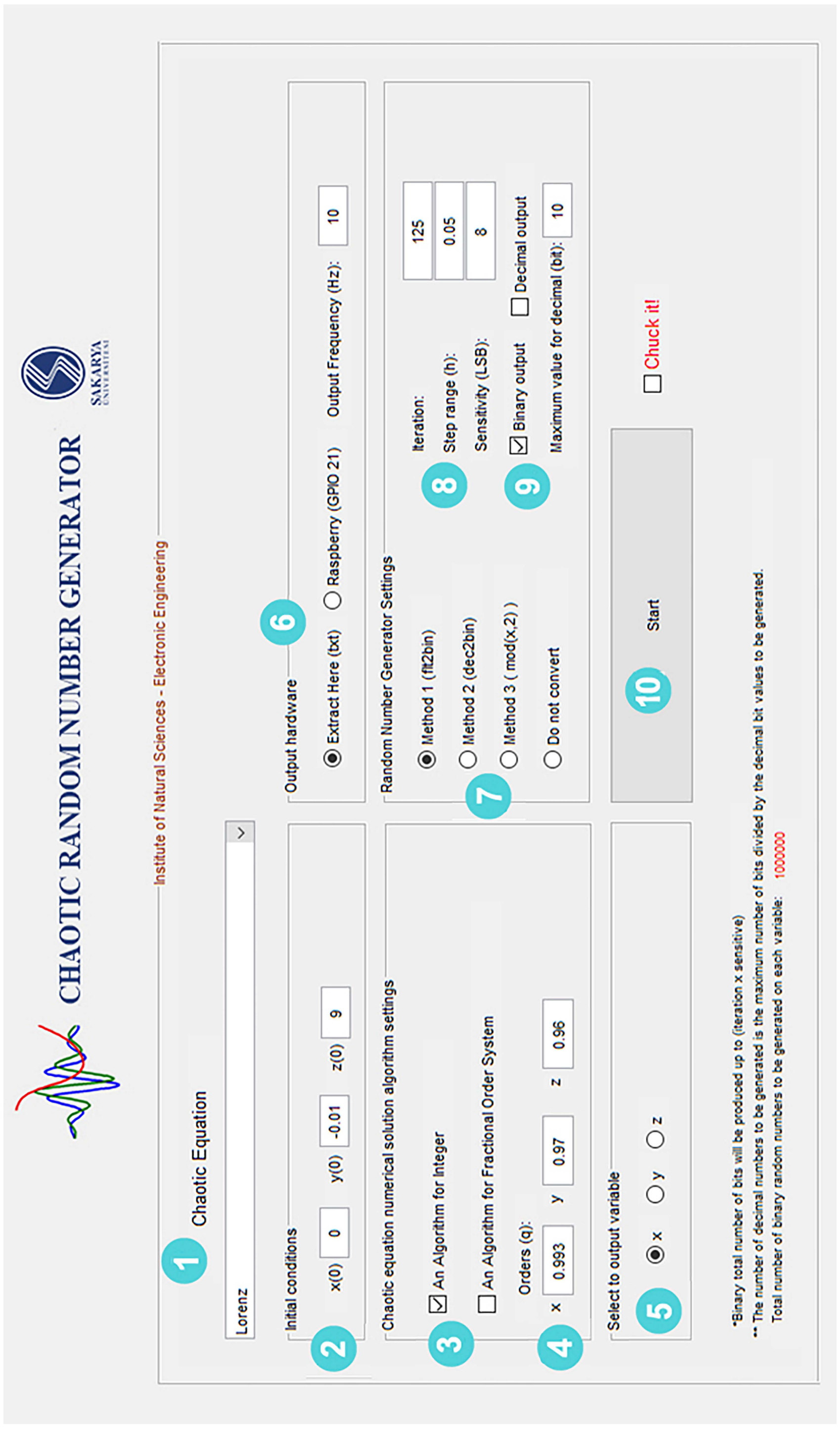
Method 2 (dec2bin): In this method, the negative numbers are converted to positive numbers by shifting the time series. Then, these all positive decimal numbers are converted to integer numbers by multiplication of power of 10. Finally, the integer numbers are converted into binary numbers.

Method 3 (mod2): In this method, only the decimal parts of the time series are considered. For every digit after the decimal point, 0 is written in the bit series if the value of the digit is even while 1 is written in the bit series if the value of the digit is odd. The number of digits can be set in the interface.

4. The Design of an Interface for Chaos Based Random Number Generation

In this part user interface for chaos based random number generation is given. The designed interface allows use of chaotic system with both integer and fractional order, use of different binary conversion methods and selection of any state variable of the chaotic system to generate random numbers. Moreover, in the designed interface new chaotic systems can be added and the order of derivative and initial condition of the existing chaotic systems and newly added one can be easily changed. For the random number generation process, options like conversion methods, number of iterations, step size and the number of LSBs can be set very easily.

In the designed interface, the continuous time chaotic systems are numerically solved with RK4 and GL algorithms for integer and fractional order respectively. In this step, the generated discrete numbers are real numbers. Then, these real decimal numbers are converted to 32-bit binary number according to the selected conversion method. After the conversion process, the LSBs are selected for generated random bit series. The number of selected LSBs can be set in the interface.



CHAOTIC RANDOM NUMBER GENERATOR

Institute of Natural Sciences - Electronic Engineering

1 Chaotic Equation: Lorenz

2 Initial conditions: x(0) 0 y(0) -0.01 z(0) 9

3 Chaotic equation numerical solution algorithm settings

☑ An Algorithm for Integer

☐ An Algorithm for Fractional Order System

4 Orders (q): x 0.993 y 0.97 z 0.96

5 Select to output variable: ● x ○ y ○ z

6 Output hardware: ● Extract Here (txt) ○ Raspberry (GPIO 21) Output Frequency (Hz): 10

7 Random Number Generator Settings: ● Method 1 (flt2bin) ○ Method 2 (dec2bin) ○ Method 3 (mod(x,2)) ○ Do not convert

8 Iteration: 125 Step range (h): 0.05 Sensitivity (LSB): 8

9 ☑ Binary output ☐ Decimal output Maximum value for decimal (bit): 10

10 Start ☐ Chuck it!

*Binary total number of bits will be produced up to (iteration x sensitive)

** The number of decimal numbers to be generated is the maximum number of bits divided by the decimal bit values to be generated.

Total number of binary random numbers to be generated on each variable: 1000000

FIGURE 2. The user interface for the chaos based random number generation.

In Fig. 2 the general view of the chaos based random number generator interface is given. In the following a detailed explanation of the interface is given.

(1) Chaotic Equation: In this part a user can select the chaotic system given in Section 2.

(2) Initial Conditions: In this part, the initial values of the selected chaotic system are set.

(3) Chaotic Equation Numerical Solution Algorithm Settings: In this part, numeric solution methods for the chaotic systems are selected. For integer order system the numeric solution method is RK4 (4th order Runge Kutta) method and for fractional order system the numeric solution method is based on GL (Grnwald Letnikov) definition.

(4) Fractional Orders: In this part, the fractional order is set if An Algorithm for Fractional Order System is checked in the interface.

(5) Output Variable Selection: In this part, the state variable of the chaotic system is selected to be used in random number generation process.

(6) Output Hardware: In this part, the location in which the generated random numbers are saved is selected. If Extract Here (txt) is checked, the random numbers are saved in a text file in the interface directory. If Raspberry (GPIO 21) is checked the generated numbers are send to Rasbperrys GPIO 21 pin.

(7) Random Number Generator Settings: In this part, random number generation methods described in Section 3 can be selected. Also, the real numbers can be obtained from the time series of the chaotic system by checking do not convert.

(8) Numerical Solution Settings: In this part, number of iteration and step size is set for the numerical solution of the chaotic system. Moreover, the number of LSBs selected for generation of binary number is set in Sensitivity (LSB) box.

(9) Random Number Output Settings: In this part, the output formats (binary or decimal) of the generated random numbers is set. Maximum value for decimal box represents the number of bit blocks for binary to decimal conversion.

(10) Start Button: This button starts the random number generation processes according to the selected options.

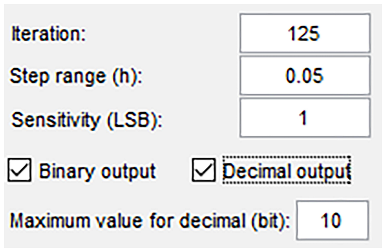

Iteration: 125
Step range (h): 0.05
Sensitivity (LSB): 1
☑ Binary output ☑ Decimal output
Maximum value for decimal (bit): 10

FIGURE 3. Maximum value for decimal box is activated after selecting Decimal Output option.

In random number generation process the obtained numbers from a chaotic system is converted into 32-bit IEEE 754 floating point number. If Binary output is selected in the interface, the number of LSBs are selected from each 32-bit number as set in Sensitivity LSB option. Then, these selected bits are written in a text file or they are sent to digital output of the Raspberry Pi. If Decimal output is selected in the interface, the generated binary random numbers are converted to decimal numbers. For binary to decimal conversion, the number of bit blocks as set in Maximum value for decimal is converted into decimal format. Then the obtained random decimal numbers are written in a text file.

In the interface, the method of conversion to binary numbers can be set with Random Number Generator Settings. In Method 1 the real numbers obtained from the time series of the chaotic systems are converted into 32-bit IEEE 754 floating point number. In Method 2 the time series of the chaotic systems are shifted up such that all the obtained numbers are positive. Then, these positive real numbers are multiplied with powers of 10 to convert all the real numbers into integer numbers. Finally, these integers are converted to binary format. In Method 3 the binary numbers are obtained according to the values of the digits after the decimal point. In Method 4 the numbers obtained from the time series of the chaotic systems are used in the random number generation process without conversion.

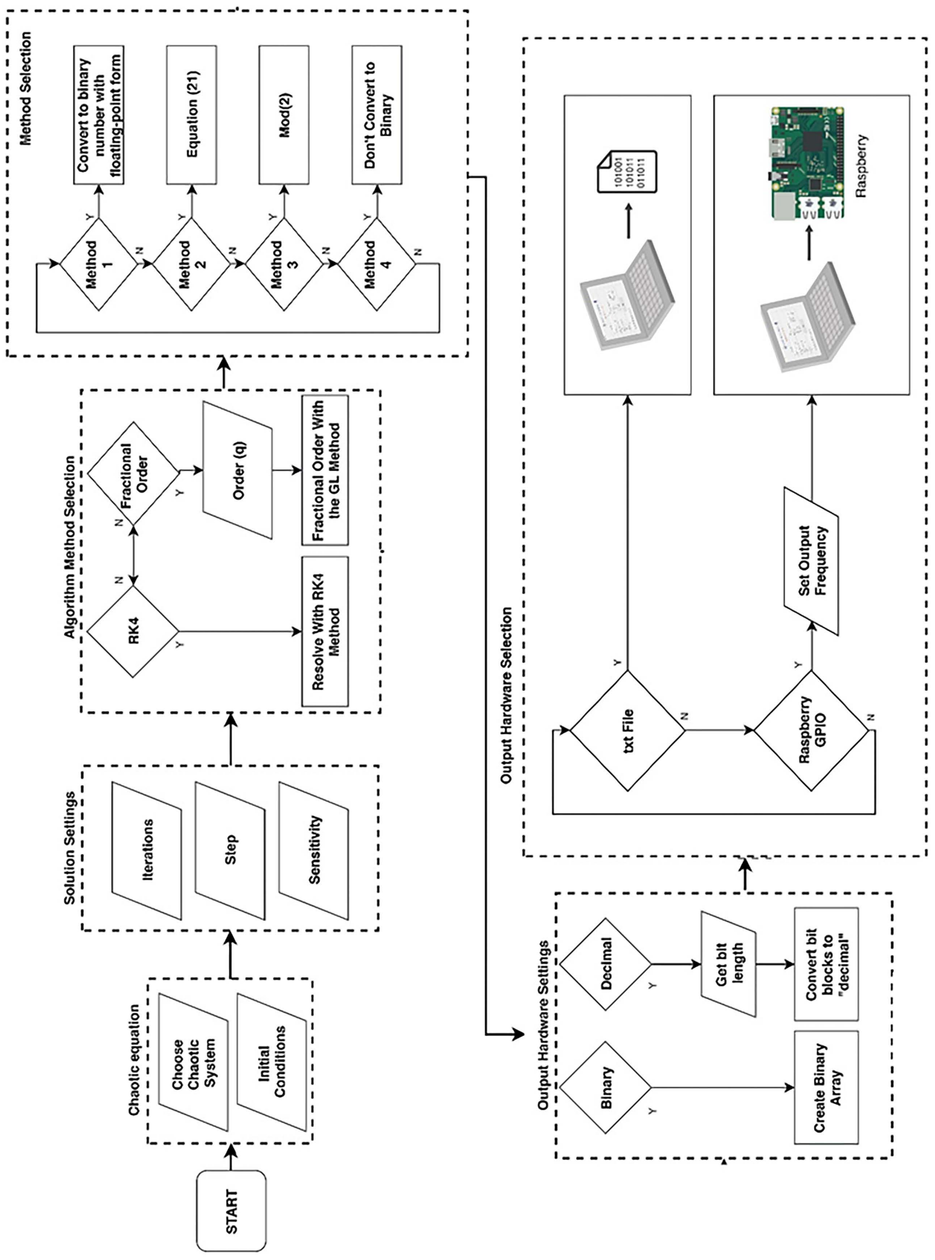


Figure 4. The block diagram of the random number generation interface.

The generated random numbers can be written in a text file or send to the GPIO21 pin of Raspberry Pi according to the selected options in the interface.

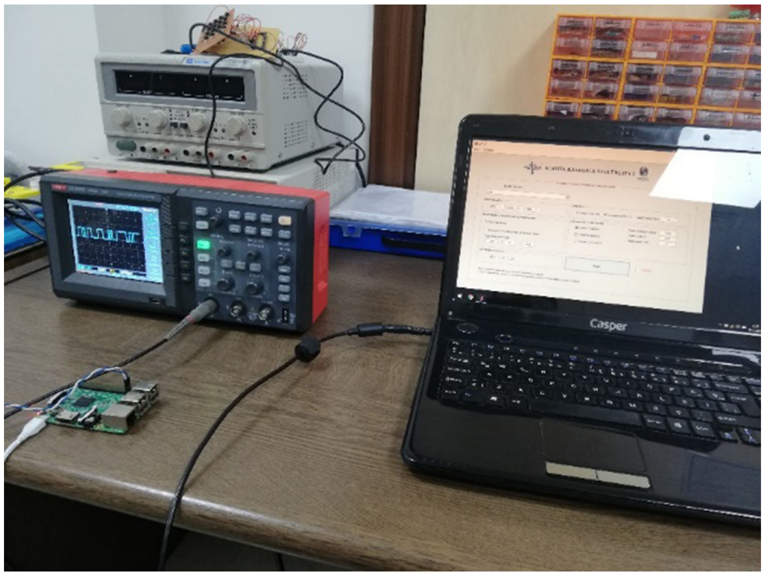


FIGURE 5. Sending the generated random numbers to Raspberry Pi wirelessly.

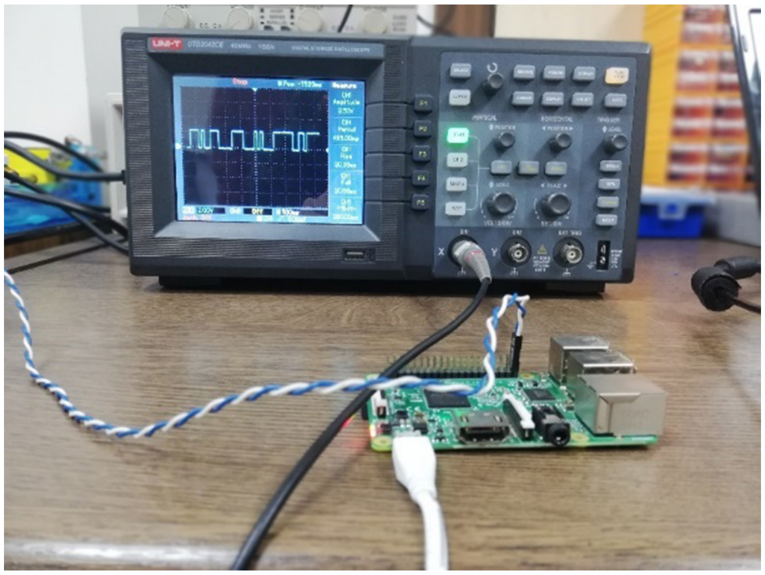


FIGURE 6. The oscilloscope output obtained from Raspberry Pi.

5. The oscilloscope output obtained from Raspberry Pi

In this study, chaos based random numbers are generated with the designed interface. In order to evaluate the randomness of the generated numbers, NIST800-22 statistical tests and ENT Monte Carlo test are conducted. In Table 1, the NIST800-22 statistical tests result for the random numbers obtained from the time series of the Lorenz system are given. In Sensitivity (LSB) option in the interface selected as 8 LSB. As it can be seen in the table, random numbers generated from Lorenz system passed all 16 NIST tests successfully for all three random number generation methods described in Section 3.

Table 1. NIST800-22 tests result for random numbers obtained from Lorenz system

Lorenz System	**Method 1**		**Method 2**		**Method 3**	
Statistical Tests	P-value	Result	P-value	Result	P-value	Result
Frequency (Monobit) Test	0.56732	Successful	0.37347	Successful	0.9904256	Successful
Block-Frequency Test	0.01636	Successful	0.88986	Successful	0.3661803	Successful
Cumulative-Sums Test	0.72051	Successful	0.45657	Successful	0.7885117	Successful
Runs Test	0.61263	Successful	0.42209	Successful	0.6100516	Successful
Longest-Run Test	0.39002	Successful	0.09205	Successful	0.4176718	Successful
Binary Matrix Rank Test	0.3404	Successful	0.73107	Successful	0.4901672	Successful
Discrete Fourier Transform Test	0.93418	Successful	0.5147	Successful	0.9195958	Successful
Non-Overlapping Templates Test	0.46464	Successful	0.06464	Successful	0.01634	Successful
Overlapping Templates Test	0.18705	Successful	0.58012	Successful	0.473944	Successful
Maurer's Universal Statistical Test	0.1921	Successful	0.58502	Successful	0.7587417	Successful
Approximate Entropy Test	0.02435	Successful	0.32102	Successful	0.6111174	Successful
Random-Excursions Test	0.79859	Successful	0.60713	Successful	0.4888407	Successful
Random-Excursions Variant Test	0.48922	Successful	0.75212	Successful	0.6371215	Successful
Serial Test-1	0.24493	Successful	0.64047	Successful	0.4685653	Successful
Serial Test-2	0.24669	Successful	0.77561	Successful	0.6927195	Successful
Linear-Complexity Test	0.34569	Successful	0.29519	Successful	0.0710091	Successful

In Table 2, the NIST800-22 statistical tests result for the random numbers obtained from the time series of the fractional ordered Lorenz system are given. In Sensitivity (LSB) option in the interface selected as 8 LSB as in the previous case. As it can be seen in the table, random numbers generated from the fractional ordered Lorenz system passed all 16 NIST tests successfully for all three methods again.

Table 2. NIST800-22 tests result for random numbers obtained from fractional order Lorenz system

Fractional Ordered Lorenz System	**Method 1**		**Method 2**		**Method 3**	
Statistical Tests	P-value	Result	P-value	Result	P-value	Result
Frequency (Monobit) Test	0,54984	Successful	0,0268	Successful	0,4065	Successful
Block-Frequency Test	0,89171	Successful	0,6909	Successful	0,2070	Successful
Cumulative-Sums Test	0,87186	Successful	0,0431	Successful	0,7784	Successful
Runs Test	0,46929	Successful	0,0458	Successful	0,1120	Successful
Longest-Run Test	0,1139	Successful	0,6166	Successful	0,2374	Successful
Binary Matrix Rank Test	0,8626	Successful	0,2378	Successful	0,5560	Successful
Discrete Fourier Transform Test	0,69314	Successful	0,1658	Successful	0,9415	Successful
Non-Overlapping Templates Test	0,02916	Successful	0,0646	Successful	0,0188	Successful
Overlapping Templates Test	0,01659	Successful	0,3872	Successful	0,6761	Successful
Maurer's Universal Statistical Test	0,71419	Successful	0,2072	Successful	0,7160	Successful
Approximate Entropy Test	0,96564	Successful	0,2219	Successful	0,7425	Successful
Random-Excursions Test	0,28161	Successful	0,4409	Successful	0,3325	Successful
Random-Excursions Variant Test	0,37047	Successful	0,3560	Successful	0,4907	Successful
Serial Test-1	0,02469	Successful	0,6084	Successful	0,8528	Successful
Serial Test-2	0,06863	Successful	0,4124	Successful	0,4675	Successful
Linear-Complexity Test	0,99186	Successful	0,96811	Successful	0,56374	Successful

For random number bit series to be considered successful in the NIST800-22 tests, the bit series must be passed successfully all the 16 tests. In Table 3, the NIST800-22 results of the random numbers generated from both integer and fractional order of the chaotic systems used in the study are given.

Table 3. NIST800-22 test results of the chaotic system used in the study (8 Bit LSB)

Chaotic Systems	Method 1		Method 2		Method 3	
	RK4	Fractional Order	RK4	Fractional Order	RK4	Fractional Order
Lorenz	**Successful**	**Successful**	**Successful**	**Successful**	**Successful**	**Successful**
Pehlivan	**Successful**	**Successful**	**Successful**	**Successful**	Failed[3]	Failed[6]
Rossler	**Successful**	**Successful**	**Successful**	**Successful**	**Successful**	**Successful**
VanDerPol	**Successful**	**Successful**	**Successful**	**Successful**	**Successful**	Failed[3]
Labyrinth	**Successful**	Failed[1]	**Successful**	**Successful**	Failed[6]	Failed[3]
Rucklidge	**Successful**	**Successful**	Failed[1]	**Successful**	Failed[4]	Failed[2]
Rikitake	**Successful**	Failed[2]	**Successful**	Failed[2]	Failed[4]	Failed[6]
Difflorenz	Failed[2]	Failed[2]	**Successful**	**Successful**	Failed	Failed[5]
Chen	**Successful**	Failed	**Successful**	Failed	**Successful**	Failed
Golden Ratio	**Successful**	Failed	**Successful**	Failed	Failed[1]	Failed
Aizawa	Failed[1]	**Successful**	**Successful**	Failed[5]	Failed[5]	Failed[6]

[1] Passed in 14 tests out of 16
[2] Passed in 13 tests out of 16
[3] Passed in 12 tests out of 16
[4] Passed in 11 tests out of 16
[5] Passed in 10 tests out of 16
[6] Passed in 8 tests out of 16

Finally, in Table 4, ENT Monte Carlo test results of the random numbers generated from both integer and fractional order of the chaotic systems used in the study are given. In the table, the results show the ratio of the Monte Carlo value to the supposed value. It is considered to be successful if this ratio is equal to or greater than 94% [32]. As it can be seen in the table, the all results show the random number generated from both integer and fractional order of the chaotic systems are passed successfully.

TABLE 4. ENT Monte Carlo test results of the chaotic system used in the study (8 Bit LSB)

	Method 1		**Method 2**		**Method 3**	
Chaotic Systems	**RK4**	**Fractional Order**	**RK4**	**Fractional Order**	**RK4**	**Fractional Order**
Lorenz	99,930%	99,999%	99,992%	99,981%	99,988%	99,999%
Pehlivan	99,927%	99,936%	99,985%	99,945%	99,982%	99,947%
Rossler	99,997%	99,969%	99,966%	99,936%	99,928%	99,908%
VanDerPol	99,793%	99,973%	99,980%	99,988%	99,974%	99,977%
Labyrinth	99,963%	99,870%	99,855%	99,964%	99,932%	99,976%
Rucklidge	99,964%	99,951%	99,967%	99,907%	99,944%	99,860%
Rikitake	99,941%	99,915%	99,982%	99,973%	99,952%	99,970%
Difflorenz	99,969%	99,968%	99,988%	99,822%	99,975%	99,933%
Chen	99,942%	99,965%	99,997%	99,966%	99,968%	99,860%
Golden Ratio	99,974%	98,041%	99,947%	96,724%	99,998%	97,832%
Aizawa	99,982%	99,921%	99,799%	99,936%	99,959%	99,980%

6. Conclusion

In this study, a user-friendly interface for integer and fractional order chaotic system based random number generator is designed. For random number generation process there are four different methods. Moreover, the microcomputer-based application of the interface is realized. In order to evaluate the randomness of the generated numbers, the NIST-800-22 and ENT tests are performed. The designed interface enables users to generate chaos based random numbers with four different methods. The generated numbers can be in binary or decimal format and they can be written in a text file or send to a hardware like Raspberry Pi as a signal. This makes mobile RNG application possible. In the interface, a user can select any pre-defined chaotic system and can define new chaotic systems. In addition to this, a user can generate a very huge number of random number sequences by just altering settings like initial condition, order of derivation, generation method etc.

References

[1] Pehlivan, ., (2007). New Chaotic Systems: Electronic Circuit Realizations, Synchronization and Secure Communication Applications. (Ph.D. Thesis), Sakarya University, Sakarya, Turkey.

[2] Kahyaolu, M. B., and S. (2015). The Analysis Of Individual Investor Behaviour within the Frame of Chaos Theory. The Journal of Business Science, 3(1), 38-51.

[3] Arslan, C., Pehlivan, ., Varan, M., and Akgl, A. (2017, September). Simulation and Analog Circuit Implementation FitzHugh-Nagumo (FHN). In 5th International Symposium on Innovative Technologies in Engineering and Science 29-30 September 2017 (ISITES2017 Baku-Azerbaijan).

[4] Yardm, F. E., and Afacan, E. (2010). Simulation of a Communication System Using Lorenz-Based Differential Chaos Shift Keying (DCSK) Model. Journal of the Faculty of Engineering and Architecture of Gazi University, 25(1).

[5] Wang, Y., Liu, Z., Ma, J., and He, H. (2016). A pseudorandom number generator based on piecewise logistic map. Nonlinear Dynamics, 83(4), 2373-2391.

[6] Cicek, I., Pusane, A. E., and Dundar, G. (2014). A novel design method for discrete time chaos based true random number generators. INTEGRATION, the VLSI journal, 47(1), 38-47.

[7] Patidar, V., Sud, K. K., and Pareek, N. K. (2009). A pseudo random bit generator based on chaotic logistic map and its statistical testing. Informatica, 33(4).

[8] avuolu, ., Kaar, S., Pehlivan, I., and Zengin, A. (2017). Secure image encryption algorithm design using a novel chaos based S-Box. Chaos, Solitons and Fractals, 95, 92-101.

[9] Kaar, S. (2016). Analog circuit and microcontroller based RNG application of a new easy realizable 4D chaotic system. Optik, 127(20), 9551-9561.

[10] Akgul, A., Calgan, H., Koyuncu, I., Pehlivan, I., and Istanbullu, A. (2016). Chaos-based engineering applications with a 3D chaotic system without equilibrium points. Nonlinear dynamics, 84(2), 481-495.

[11] Rajagopal, K., Akgul, A., Jafari, S., Karthikeyan, A., avuolu, ., and Kacar, S. (2019). An Exponential Jerk System: Circuit Realization, Fractional Order and Time Delayed Form with Dynamical Analysis and Its Engineering Application. Journal of Circuits, Systems and Computers, 28(05), 1950087.

[12] Rajagopal, K., Jafari, S., Kacar, S., Karthikeyan, A., and Akgl, A. (2019). Fractional Order Simple Chaotic Oscillator with Saturable Reactors and Its Engineering Applications. Information Technology and Control, 48(1), 115-128.

[13] Rajagopal, K., Akgul, A., Jafari, S., Karthikeyan, A., Cavusoglu, U., and Kacar, S. (2019). An exponential jerk system, its fractional-order form with dynamical analysis and engineering application. Soft Computing, 1-11.

[14] Hu, Y., Liao, X., Wong, K. W., and Zhou, Q. (2009). A true random number generator based on mouse movement and chaotic cryptography. Chaos, Solitons and Fractals, 40(5), 2286-2293.

[15] Karakaya, B., Glten, A., and Frasca, M. (2019). A true random bit generator based on a memristive chaotic circuit: Analysis, design and FPGA implementation. Chaos, Solitons and Fractals, 119, 143-149.

[16] Pareschi, F., Setti, G., and Rovatti, R. (2006, September). A fast chaos-based true random number generator for cryptographic applications. In 2006 Proceedings of the 32nd European Solid-State Circuits Conference (pp. 130-133). IEEE.

[17] Alcin, M., Koyuncu, I., Tuna, M., Varan, M., and Pehlivan, I. (2019). A novel high speed Artificial Neural Networkbased chaotic True Random Number Generator on Field Programmable Gate Array. International Journal of Circuit Theory and Applications, 47(3), 365-378.

[18] Lorenz, E. N. (1963). Deterministic nonperiodic flow. Journal of the atmospheric sciences, 20(2), 130-141.

[19] Rossler, O. E. (1976). An equation for continuous chaos. Physics Letters A, 57(5), 397-398.

[20] Cartwbight, M. L. (1960). Balthazar van der Pol. Journal of the London Mathematical Society, 1(3), 367-376.

[21] Chen, G., and Ueta, T. (1999). Yet another chaotic attractor. International Journal of Bifurcation and chaos, 9(07), 1465-1466.

[22] Sprott, J. C., and Chlouverakis, K. E. (2007). Labyrinth chaos. International Journal of Bifurcation and Chaos, 17(06), 2097-2108.

[23] Rucklidge, A. M. (1992). Chaos in models of double convection. Journal of Fluid Mechanics, 237, 209-229.

[24] Rikitake, T. (1958, January). Oscillations of a system of disk dynamos. In Mathematical Proceedings of the Cambridge Philosophical Society (Vol. 54, No. 1, pp. 89-105). Cambridge University Press.

[25] Ito, K. (1980). Chaos in the Rikitake two-disc dynamo system. Earth and Planetary Science Letters, 51(2), 451-456.

[26] Pehlivan, ., and Uyaroglu, Y. (2007). Rikitake attractor and its synchronization application for secure communication systems. Journal of Applied Sciences, 7(2), 232-236.

[27] Pehlivan, ., and Uyarolu, Y. (2012). A new 3D chaotic system with golden proportion equilibria: Analysis and electronic circuit realization. Computers and Electrical Engineering, 38(6), 1777-1784.

[28] Kilbas, A. A., Srivastava, H. M., and Trujillo, J. J. (2006). Theory and applications of fractional differential equations. North-Holland mathematics studies.

[29] Mathai, A. M., Saxena, R. K., and Haubold, H. J. (2009). The H-function: theory and applications. Springer Science and Business Media.

[30] Korkmaz, M. (2013). Fractional Order PID Controllers, Design, Application and Comparison (M.Sc. Thesis, Seluk University, Konya, Turkey).

[31] Petr, I. (2011). Fractional-order nonlinear systems: modeling, analysis and simulation. Springer Science and Business Media.

[32] Walker, J. (2008). ENT: a pseudorandom number sequence test program. Software and documentation available at/www. fourmilab. ch/random/S.

Department of Electrical and Electronics Engineering, Sakarya University of Applied Sciences,, Sakarya, 54187, Turkey

Email address: `aakgul@sakarya.edu.tr`

Department of Electrical and Electronics Engineering, Sakarya University of Applied Sciences,, Sakarya, 54187, Turkey

Department of Electrical and Electronics Engineering, Sakarya University of Applied Sciences,, Sakarya, 54187, Turkey