applied sciences

MDPI

Article

Design of Synchronized Large-Scale Chaos Random Number Generators and Its Application to Secure Communication

Teh-Lu Liao [1], Pei-Yen Wan [1] and Jun-Juh Yan [2,*]

[1] Department of Engineering Science, National Cheng Kung University, Tainan 701, Taiwan; tlliao@mail.ncku.edu.tw (T.-L.L.); s16637113@stu.edu.tw (P.-Y.W.)
[2] Department of Computer and Communication, Shu-Te University, Kaohsiung 824, Taiwan
* Correspondence: jjyan@stu.edu.tw; Tel.: +886-7-6158000 (ext. 4806)

Received: 8 December 2018; Accepted: 2 January 2019; Published: 7 January 2019

Abstract: This paper is concerned with the design of synchronized large-scale chaos random number generators (CRNGs) and its application to secure communication. In order to increase the diversity of chaotic signals, we firstly introduce additional modulation parameters in the original chaotic Duffing map system to modulate the amplitude and DC offset of the chaotic states. Then according to the butterfly effect, we implement modulated Duffing map systems with different initial values by using the microcontroller and complete the design of large-scale CRNGs. Next, a discrete sliding mode scheme is proposed to solve the synchronization problem of the master-slave large-scale CRNGs. Finally, we integrate the aforementioned results to implement an innovative secure communication system.

Keywords: chaotic Duffing map system; butterfly effect; large-scale CRNG; discrete sliding mode; secure communication

1. Introduction

Chaotic system is a complex nonlinear system. In recent years, it has attracted extensive attention in the field of engineering research. The reason is that chaotic systems have very rich dynamic behavior, unpredictable trajectory, white noise-like broadband and the initial value sensitivity of the butterfly effect. The chaos dynamics in most mechanical engineering systems is undesirable and needs to be suppressed because it will affect the performance or damage the mechanical structure of systems. In 1990, O.G.Y (Ott, Grebogi and Yorke) [1] proposed the study of chaos control. Researchers pointed out that, when parameter values of chaotic systems are slightly modulated, the chaotic behavior will produce huge and unpredictable changes. Several feedback control methods to suppress chaos behavior have been proposed in the literature [2,3]. Although the chaos behavior might be undesired in most mechanical systems, the noise-like behavior of chaos is useful to the application of image encryption and secure communication [4–8]. Due to the practical application of chaos synchronization, since the pioneering research of Pecora and Carroll [9], chaos synchronization has become an interesting research field. Many control methods have been proposed to solve the synchronization problem for chaotic systems, such as $H\infty$ control [10,11], fuzzy sliding mode control [12,13], adaptive control [14,15] and so forth. Among proposed control designs, sliding mode control method is often the first choice for researchers because it is not sensitive to system parameters and external disturbances and with good robustness.

In these two decades, chaotic systems have been widely applied in the areas of communication, medicine and biology to solve several important engineering problems, especially in the security field of communication. In the previous reports [4–6], the authors used the continuous chaotic systems with

well-designed synchronization controllers to achieve the secure communication. However, in practical circuit implementations for continuous chaotic systems, analog components, such as operational amplifier (OPA), resistors, capacitors are necessary. Unfortunately, these analog components are sensitive to environmental conditions and often result in circuit instability. On the other hand, owing to the progress of microcontrollers with digital signal processing (DSP) technology, using microcontrollers to replace the analog circuits has become more and more popular and important. Discrete chaotic systems implemented by the microcontroller are more robust and less susceptible to temperature variation and component aging but also at a lower price. Therefore, to solve the problem of instability of analog circuits, we need to study the discrete chaotic systems realized by microcontrollers. Until now, the applications of discrete chaotic systems to solve image encryption have been proposed in References [7,8]. However, the synchronization control problem of discrete chaotic systems has not been well solved in these studies. In addition, to the best of our knowledge, using discrete chaotic systems to design the synchronized large-scale CRNGs has not been well discussed. Furthermore, recently, due to the advent of quantum computers, many traditional encryption methods may become unsafe. The encryption can be quickly cracked by quantum computers. Because the chaos system has rich dynamic characteristics, it can generate a large amount of random signals in a very short time and many scholars have pointed out that chaotic encryption method may be one of the solutions against the attack of quantum computers [16,17].

For the above reasons, in this paper, we study the design of synchronized large-scale CRNGs and its application to secure communication. In order to simplify the presentation of the paper, we choose Duffing map for this study. However, the proposed modulation approach can also effectively applied to other classes of discrete chaotic systems. First, we introduce a modulation approach by introducing some parameters into the chaotic Duffing map to regulate and increase the diversity of chaotic state responses. Then we implement the proposed modulated Duffing map systems under different initial conditions by the microcontroller. Due to the butterfly effect, the state response of each Duffing map system will be quite different. Therefore, we can get a large amount of random chaotic states and complete the design of large-scale CRNGs. For subsequent communication security application, we use sliding mode control to cope with the synchronization problem of the master-slave large-scale CRNGs. Finally, the above results are integrated to realize an innovative secure communication system to prove the correctness and feasibility of the research.

2. Design of a Large-Scale CRNG

In this paper, we will discuss the design and realization of a large-scale CRNGs and its application to secure communication based on synchronized master-slave large-scale CRNGs. To design the large-scale CRNGs, in the following, we first introduce the modulation of the Duffing map chaotic system such that diversified random chaotic state responses can be obtained for large-scale CRNGs. The original Duffing map system is described by the following system.

$$\begin{aligned} x_1(k+1) &= x_2(k) \\ x_2(k+1) &= -0.2x_1(k) + 2.75x_2(k) - x_2^3(k) \end{aligned} \tag{1}$$

where $x_1, x_2(k)$ are the system states. In order to adjust the amplitude and DC offset for state responses, we define new state variables $y_i, i = 1, 2$ satisfying

$$y_i(k) = a_i x_i(k) + d_i \tag{2}$$

where $a_i, i = 1, 2$ are given to adjust the amplitude and the DC offset of state responses, respectively. From Equation (2), we have

$$x_i(k) = \frac{y_i(k) - d_i}{a_i}, i = 1, 2 \tag{3}$$

By (1) and (3), a new modulated Duffing map chaotic system can be obtained as

$$\begin{aligned} y_1(k+1) &= \lambda_1 y_2(k) + \lambda_2 \\ y_2(k+1) &= \beta_1 y_1(k) + \beta_2 y_2^3 + \beta_3 y_2^2 + \beta_4 y_2 + \beta_5 \end{aligned} \tag{4}$$

where

$$\lambda_1 = \frac{a_1}{a_2}, \lambda_2 = -\frac{a_1}{a_2} d_2 + d_1, \beta_1 = -\frac{0.2 a_2}{a_1}$$
$$\beta_2 = -\frac{1}{a_2^2}, \beta_3 = \frac{3 d_2}{a_2^2}, \beta_4 = 2.75 - \frac{3 d_2^2}{a_2^2}, \beta_5 = \frac{0.2 a_2 d_1}{a_1} - 2.75 d_2 + \frac{d_2^3}{a_2^2} + d_2$$

For evaluating the modulation of the system amplitude and DC offset of random number generator, we give the following simulation analysis. In the simulation, the modulation parameters are given as $a_1 = 2, a_2 = 4, d_1 = -5, d_2 = 5$ and the initial conditions are selected as $x_1 = 0.4, x_2 = -0.2$. The simulation results are given in Figures 1–3.

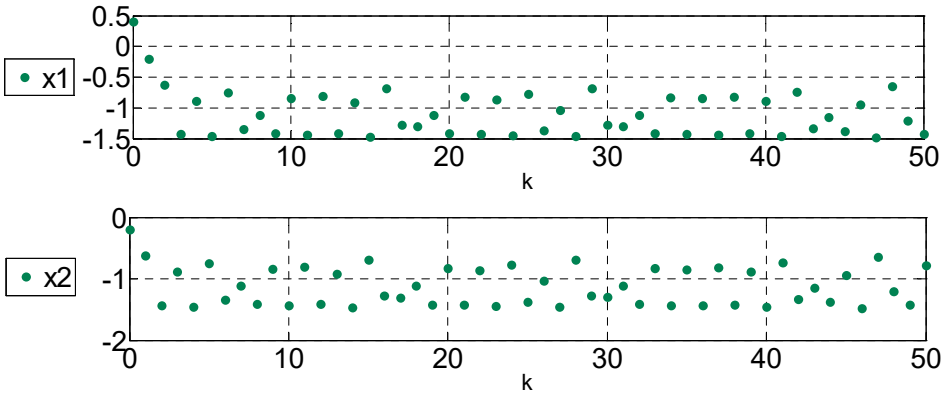

Figure 1. The state responses of the Duffing map chaotic system (before modulation).

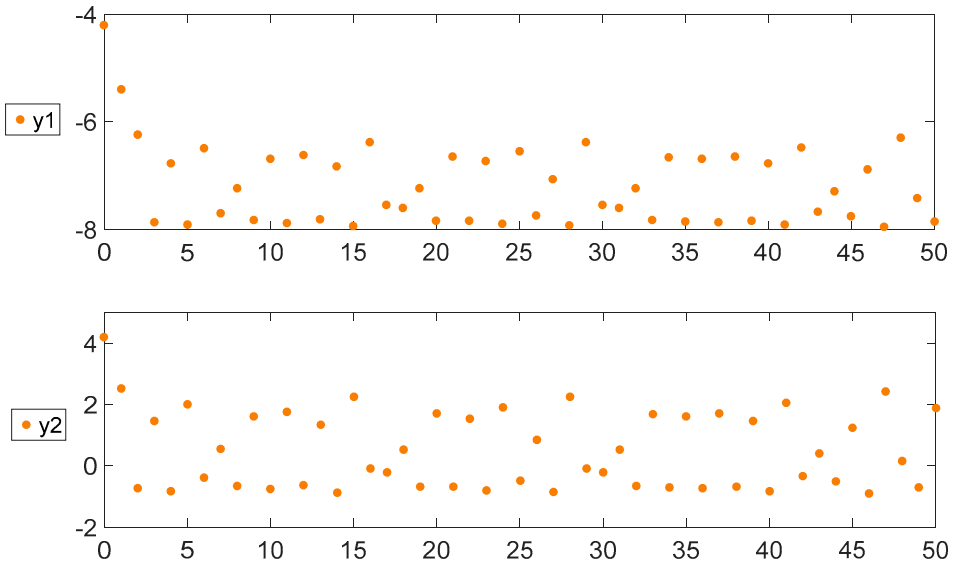

Figure 2. The state responses of the Duffing map chaotic system (after modulation).

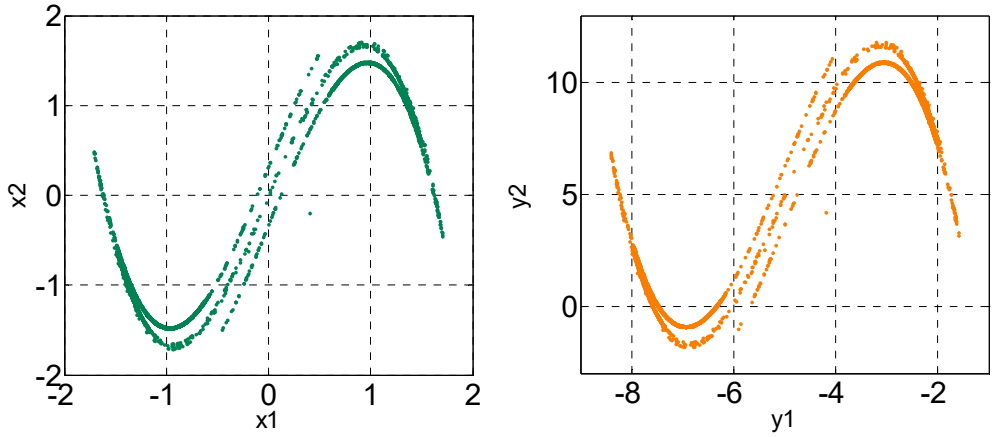

Figure 3. The strange attractors of the original and modulated Duffing map chaotic systems, respectively (left is before modulation, right is after modulation).

According to the simulation results above, it reveals, as expected, the amplitude is modulated by 2 times and 4 times, while the DC offset is shifted by -5 and 5. Furthermore, we use the National Institute of Standards and Technology (NIST) [18] test suite to test the randomness of the modulated chaotic states. In the NIST test, first, we set the test parameters including the sequences length $n = 2 \times 10^7$ bits, the number of subsequences, $m = 20$. Then, we take byte 5 (bits 32~39) of every modulated chaos states y1 and y2 in IEEE754 double-precision format [19] to test the randomness. Finally, all the test results are shown in Table 1. In Table 1, the outcome of the test values is called p value. When p value ≥ 0.01, then it passes the test. From Table 1, the generated numbers of y1 pass all tests and those of y2 pass 13 tests. Therefore, we can conclude that the modulated chaotic states are with good randomness.

Table 1. Randomness test.

Tests	y1	y2
Frequency	0.911413	0.048716
Block Frequency	0.437274	0.637119
Cumulative Sums	0.275709	0.000648
Runs	0.534146	0.162606
Longest Run	0.350485	0.739918
Rank	0.534146	0.739918
FFT	0.834308	0.991468
Nonoverlapping Template	0.048716	0.000026
Overlapping Template	0.739918	0.162606
Universal	0.534146	0.275709
Approximate Entropy	0.911413	0.637119
Random Excursions	0.066882	0.090936
Random Excursions Variant	0.017912	0.025193
Serial	0.048716	0.048716
Linear Complexity	0.637119	0.437274

From discussed above, the random number generator can be implemented by using the modulated Duffing map chaotic system. It is well known that the state response of a chaotic system is very sensitive to initial values and very small differences can lead to very different chaotic state responses, namely the butterfly effect. Based on this feature, we can use the butterfly effect to design large-scale CRNGs. In addition, due to the high capacity and fast computing speed of the digital microcontroller, we can implement this large-scale CRNG with microcontroller and can get a large amount of random numbers in the very short time according to the computing speed of the microcontroller. To show the butterfly effect, we use three sets of random generators (Duffing map systems in (1)) for comparison.

(x_1, x_2) is the random number of the first random number generator, the second is (x_{11}, x_{12}) and the third is (x_{21}, x_{22}). The initial conditions are selected as $(x_{10}, x_{20}) = (0.9, -0.5)$, $(x_{110}, x_{120}) = (0.900001, -0.500012)$, $(x_{210}, x_{220}) = (0.9000011, -0.500001)$, respectively. From the simulation results in Figure 4, it can be found that the initial values with very small difference, as expected, will produce completely different random state responses.

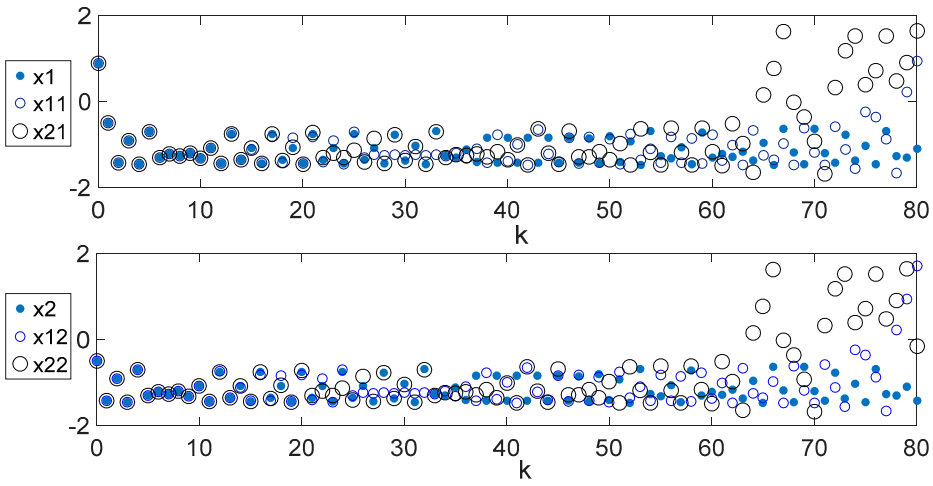

Figure 4. Butterfly effect.

Now we are ready to implement the large-scale CRNG. Its structure is designed as shown in Figure 5. We construct n Duffing maps with the same structure in the microcontroller. Each Duffing map has different initial values. According to the butterfly effect mentioned above, we can get 2n random numbers. It is worth mentioning that the microcontrollers have a large amount of program memory capacity and fast computing speed, which means that we can use the microcontroller to complete the large-scale CRNG in Figure 5 and obtain a large amount of random numbers in a very short period time.

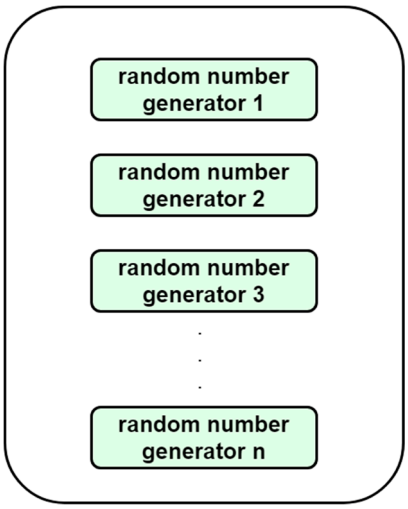

Figure 5. The structure of a large-scale chaos random number generator (CRNG).

In the following, we continue to discuss the realization of the proposed large-scale CRNG in Figure 5. First, we construct ten sets (n = 10) of CRNGs by using the HT32F1765 microcontroller as shown in Figure 6. The HT32F1765 operates at a frequency up to 72MHz with a Flash accelerator to obtain maximum efficiency. It provides 128KB of embedded Flash memory for code/data storage and up to 64 KB of embedded SRAM memory for system operation and application program usage.

Due to the butterfly effect in the chaotic system, each set of CRNGs has different initial values, so the output responses of the CRNGs of each set will be completely different. When we set up 10 sets of CRNGs with different initial values, we can get 20 random numbers and then according to the IEEE 754 double-precision standard, each random floating-point value is represented with 64 bits. Therefore, we can get 1280 random binary bits ($2 \times 10 \times 64$). At the same time, if necessary, we can also build more chaotic systems in the HT32F1765 microcontroller to get more random bits. By using 8×8 LED matrix, Figure 7 shows 256 random binary signals obtained from the proposed large-scale CRNG.

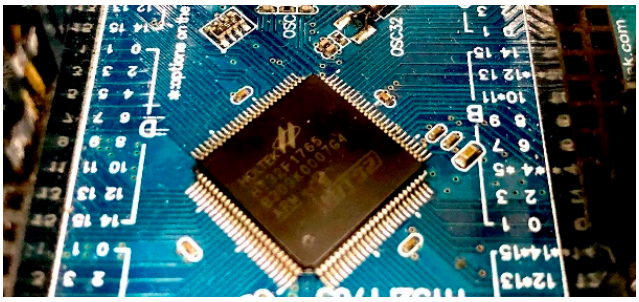

Figure 6. Holtek 32-bit microcontroller.

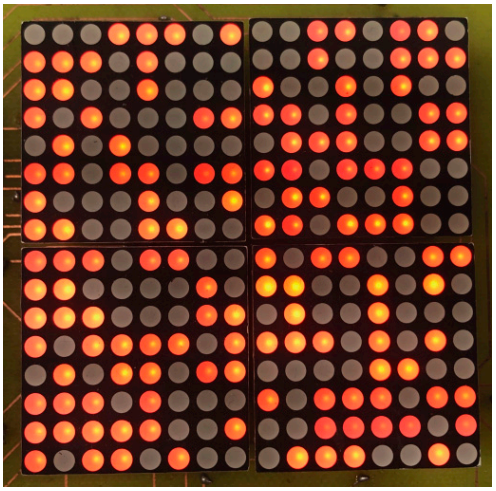

Figure 7. 256 random binary signals.

3. Synchronization of Master-Slave Large-Scale CRNGs

The design of large-scale CRNGs has been completed. We continue to study the synchronization of master and slave large-scale CRNGs. As mentioned above, the large-scale CRNG is composed of n chaotic CRNGs with the same structure, so the synchronization controller for each chaotic CRNGs will also have the same structure. Therefore, we first consider the design of a single master-slave CRNG. The master and slave CRNGs are defined, respectively, as below.

Master random number generator:

$$\begin{aligned} x_1(k+1) &= \lambda_1 x_2(k) + \lambda_2 \\ x_2(k+1) &= \beta_1 x_1(k) + \beta_2 x_2^3(k) + \beta_3 x_2^2(k) + \beta_4 x_2(k) + \beta_5 \end{aligned} \tag{5}$$

Slave random number generator:

$$\begin{aligned} y_1(k+1) &= \lambda_1 y_2(k) + \lambda_2 \\ y_2(k+1) &= \beta_1 y_1(k) + \beta_2 y_2^3(k) + \beta_3 y_2^2(k) + \beta_4 y_2(k) + \beta_5 + u(k) \end{aligned} \tag{6}$$

where x_i and $y_i, i = 1, 2$ are, respectively, the state variables of the master and slave systems. $u(k) \in R$ is the control input introduced to achieve synchronization between the master and slave CRNGs. By defining $e_i(k) = y_i(k) - x_i(k), i = 1, 2$, the error dynamics can be obtained by the following equation:

$$\begin{aligned} e_1(k+1) &= \lambda_1 e_2(k) \\ e_2(k+1) &= \beta_1 e_1(k) + \beta_2(y_2^3(k) - x_2^3(k)) + \beta_3 e_2(k)(y_2(k) + x_2(k)) + \beta_4 e_2(k) + u(k) \end{aligned} \tag{7}$$

In the following, the discrete sliding mode control (DSMC) is utilized to achieve the synchronization. Generally, the DSMC design is composed of two steps. First, we need to select an appropriate switching function for error dynamics (7) such that the sliding motion can ensure the convergence of the error states. Second, we need to propose a DSMC to guarantee the existence of the sliding mode and maintain the error dynamics on the sliding manifold [2]. To achieve the synchronization based on DSMC, a switching function is selected as:

$$s(k) = e_2(k) + \alpha e_1(k) \tag{8}$$

Assuming the error dynamics is already on the sliding manifold ($s(k) = 0$), we have

$$e_2(k) = -\alpha e_1(k) \tag{9}$$

Substituting (9) into (7), we can obtain

$$e_1(k+1) = -\alpha\lambda_1 e_1(k) \tag{10}$$

From (10), we can see that if α is specified to satisfy $|-\alpha\lambda_1| < 1$, than e_1 converges to zero. Furthermore, since $e_2(k) = -\alpha e_1(k)$ in the sliding manifold, we obtain $e_2 = 0$ when $e_1 = 0$. After discussing the selection of the switching surface, we still have to design the controller to ensure that the system can smoothly enter the sliding manifold such that $s(k) = 0$ and $e_2(k) = -\alpha e_1(k)$ can be ensured. The controller design is described as follows. According to (7) and (8), we have

$$\begin{aligned} &s(k+1) \\ &= e_2(k+1) + \alpha e_1(k+1) \\ &= \beta_1 e_1(k) + \beta_2(y_2^3(k) - x_2^3(k)) + \beta_3 e_2(k)(y_2(k) + x_2(k)) \\ &+\beta_4 e_2(k) + u(k) + \alpha\lambda_1 e_2(k) \end{aligned} \tag{11}$$

If the controller $u(k)$ is properly designed as:

$$\begin{aligned} u(k) = &-(\beta_1 e_1(k) + \beta_2(y_2^3(k) - x_2^3(k)) + \beta_3 e_2(k)(y_2(k) + x_2(k)) \\ &+\beta_4 e_2(k) + \alpha\lambda_1 e_2(k)) + \gamma s(k) \end{aligned} \tag{12}$$

where $|\gamma| < 1$. Substituting (12) into (11), we can get

$$s(k+1) = \gamma s(k) \tag{13}$$

Since $|\gamma| < 1$ is specified in (12), the error system can smoothly enter the sliding manifold, that is, $\lim_{k\to\infty} s(k) = 0$. From the above discussion, we can confirm that the system can smoothly enter the sliding mode under the action of the controller (12) and the error states in (7) can also converge to zero as discussed above, that is, the master-slave CRNGs in (5) and (6) can achieve synchronization.

In the following, for evaluating the synchronization effect of master and slave CRNGs, the parameters are given as $a_1 = 1, a_2 = 5\ d_1 = 3, d_2 = -3$. Therefore, the sliding mode control law can be obtained by (12) with $\alpha = 0.3, \gamma = 0.2$. In numerical simulations, the initial conditions are selected as $x_{10} = 3.1, x_{20} = -4.5, y_{10} = -3.3, y_{20} = -4$. The simulation results are shown in Figures 8–11. Figure 8

shows the corresponding state responses of master and controlled slave CRNGs. Figure 9 shows the error response between master and controlled slave CRNGs. Figures 10 and 11, respectively, show the switching function and control input and phase planes of the controlled master-slave CRNG. From the simulation results, it shows the proposed sliding mode control works well and the chaotic behavior of controlled master and slave random number generator can be asymptotically synchronized.

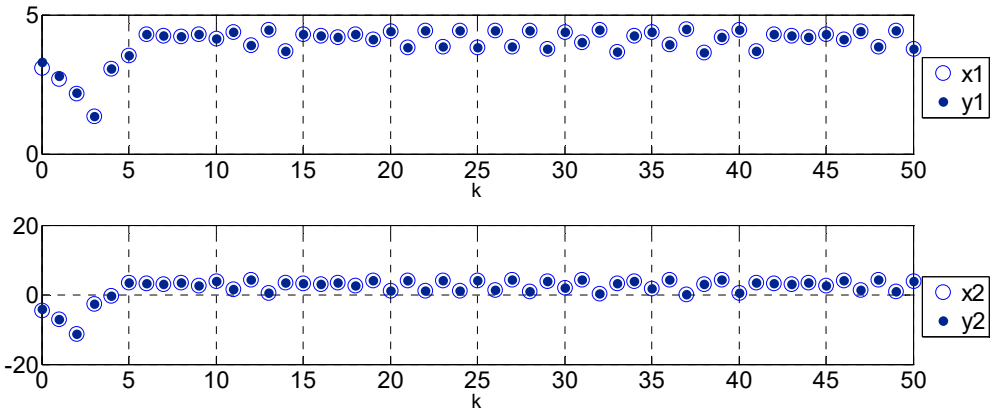

Figure 8. The state responses of the controlled master and slave CRNGs.

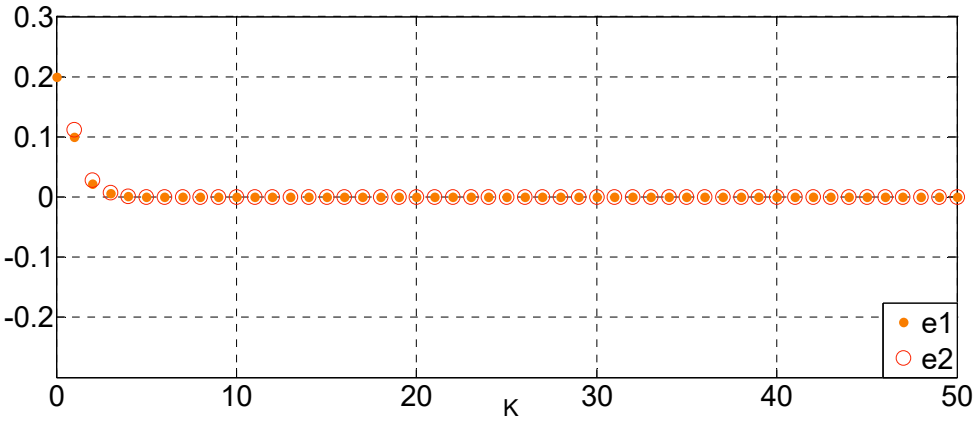

Figure 9. The error responses of the controlled master and slave CRNGs.

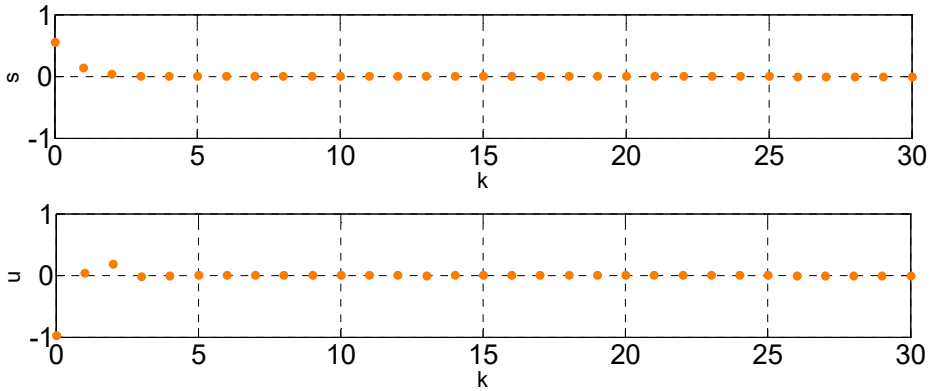

Figure 10. The switching function and control input.

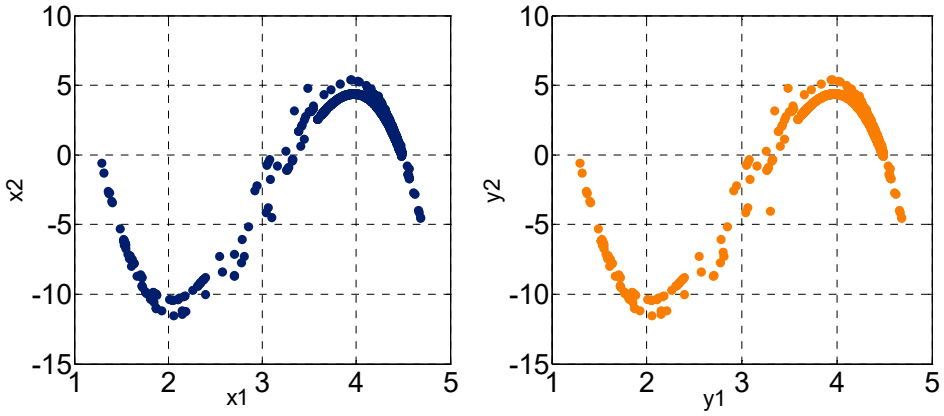

Figure 11. The phase planes of the controlled master and slave CRNGs.

Considering the security of future applications and reducing the complexity of synchronization controller for the master-slave large-scale CRNGs, we divide the controller $u(k)$ (12) into two parts, $u_m(k)$ and $u_s(k)$ satisfying $u(k) = u_m(k) + u_s(k)$, where $u_m(k)$ and $u_s(k)$ are the combination signals of master and slave random number generator, respectively. When the master and slave CRNGs are in different locations, only the information $u_m(k)$ is sent from the master CRNG through the public channel while $u_s(k)$ is generated in the slave CRNG and never appear in the insecure channel. Therefore, the synchronization controller $u(k)$ can be composed in the slave side and the master and slave CRNGs can be synchronized without transmitting full secret information through the public channel.

After completing the synchronization of the single master-slave CRNGs, the large-scale synchronized master-slave CRNGs is given as shown in Figure 12.

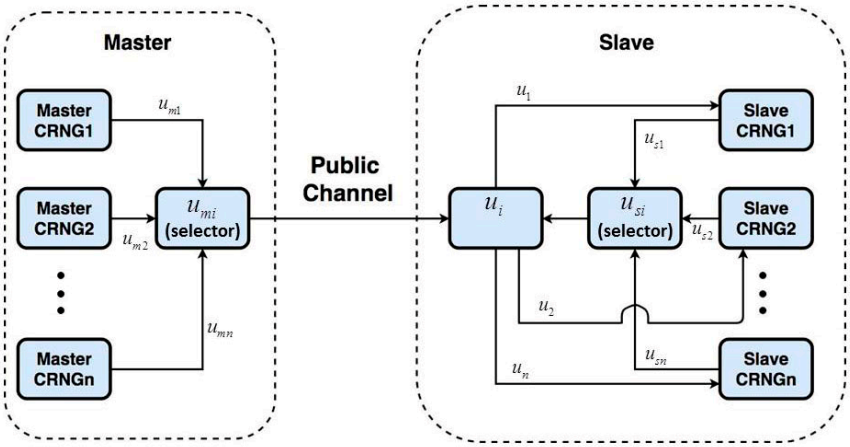

Figure 12. Synchronized large-scale CRNGs.

In Figure 12, First, the master CRNG$_i, i = 1, 2 \ldots, n$, sequentially generates the information $u_{mi}(k)$ required by each RNG$_i$ synchronization controller and in the slave RNG$_i, i = 1, 2 \ldots, n$, also sequentially generates the required information $u_{si}(k)$. $u_{mi}(k)$ and $u_{si}(k)$ are, respectively, time-sharing selected by the selectors in the master and slave sides, then $u_{mi}(k)$ and $u_{si}(k)$ is integrated to form the aforementioned controller ($u_i(k) = u_{mi}(k) + u_{si}(k)$) and then the synchronization between master CRNG$_i$ and slave CRNG$_i$, $i = 1, 2 \ldots, n$ can be guaranteed. Because the computing speed of the microcontroller is extremely fast, the large-scale master-slave CRNGs can also be synchronized quickly.

4. Design of Secure Communication Based on Synchronized Large-Scale CRNGs

In the following, we will discuss how to apply the synchronized master-slave large-scale CRNGs to construct the secure communication system. The design diagram is given as follows:

In Figure 13, the plaintext is sent to the encryption mechanism of the transmitter. The master large-scale CRNG sends the random state to the encryption mechanism and encrypt the original signal. After the encryption is completed, the transmitter transmits the cipher-text and the information $u_{mi}(k)$ to the receiver. At the receiver, combined $u_{si}(k)$ with $u_{mi}(k)$ transmitted from the transmitter, the synchronization controller $u_i(k) = u_{mi}(k) + u_{si}(k)$ is constructed to achieve synchronization of the master and slave large-scale CRNGs. Finally, the decryption mechanism with the synchronization signal can complete the decryption.

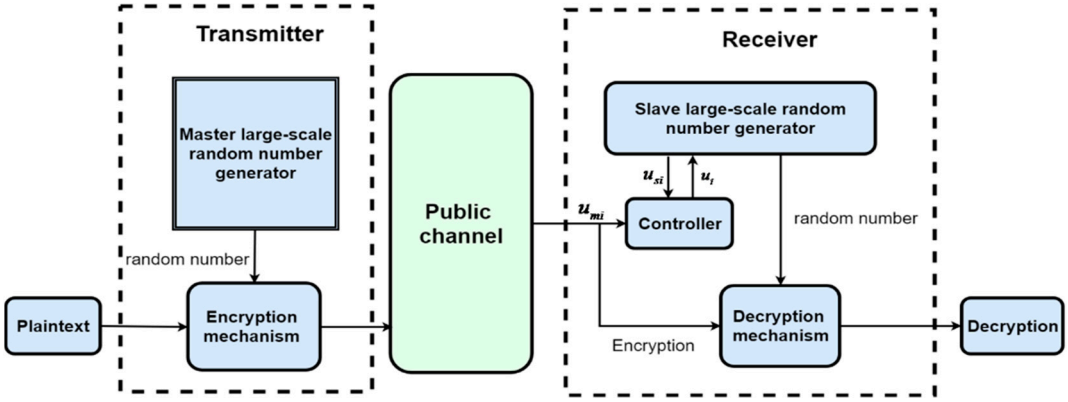

Figure 13. Communication system design.

In order to verify the design of the above-mentioned secure communication system, we use Holtek's 8-bit (HT66F50) and 32-bit (HT32F1765) microcontrollers to realize the secure communication in Figure 13. The following Figure 14 shows the diagram of the secure communication system. In Figure 14, 8-bit microcontroller is used to generate plaintext and 32-bit one is used to perform large-scale CRNGs and its synchronization.

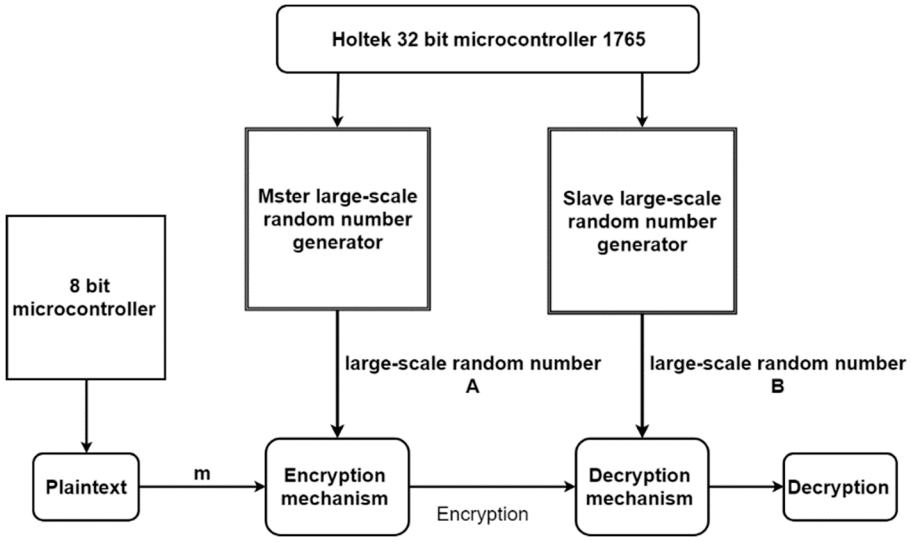

Figure 14. Communication system design diagram.

The encryption and decryption in Figure 14 are realized by using the exclusive-or as shown in Figure 15. When the master and slave large-scale CRNGs reaches synchronization, then the master large-scale random number A and the slave large-scale random number B will be the same. The encryption can be decrypted and the plaintext m can be recovered at the receiver side because $n = m \oplus A \oplus B = m \oplus A \oplus A = m$.

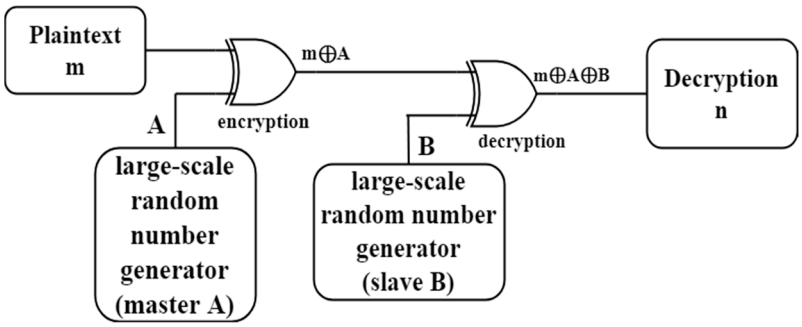

Figure 15. The structure of encryption and decryption mechanism.

After completing the circuit design, we use some basic components to implement the circuit as shown in Figure 16, in which the dot matrix LEDs are used to display master and slave random signals, plaintext, cipher-text and decrypted text, respectively. In Figure 16, we first disable the synchronization controller. We can find that the master random number and the slave random number cannot be synchronized and the encryption cannot be decrypted smoothly. While in Figure 17, we enable the synchronization controller. Therefore, it can be found that the master random number and the slave random number are synchronized, as expected and the encryption can be decrypted.

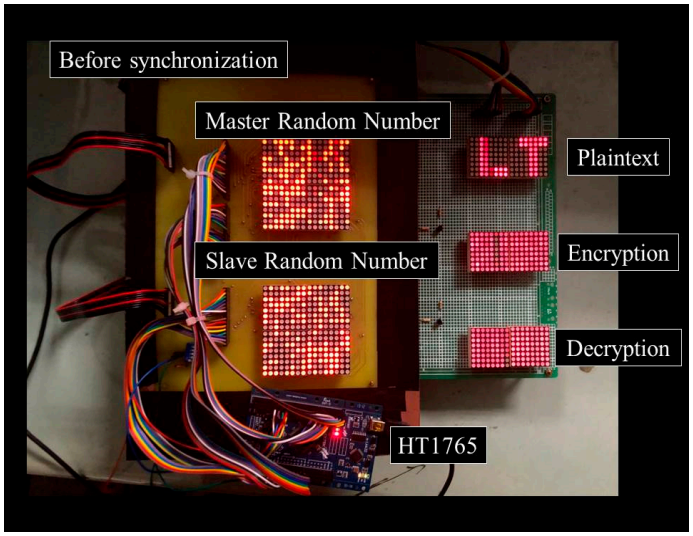

Figure 16. The secure communication system before synchronization.

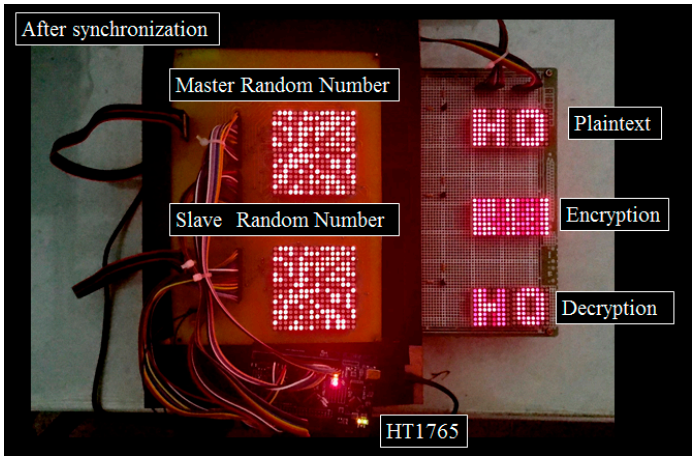

Figure 17. The secure communication system after synchronization.

5. Conclusions

This paper proposes the synchronized master-slave large-scale CRNGs and its application to secure communication. Some parameters are introduced to modulate the amplitude and DC offset of the chaotic states. Then, by using the butterfly effect, we complete the design of large-scale CRNGs. The discrete sliding mode control method is used to solve the synchronization problem of the master and slave large-scale CRNGs. Finally, the above research results are integrated to design an innovative confidential communication system. The proposed simulation, experimental results and the realization of the secure communication system are given to show the effectiveness of the proposed method.

Author Contributions: All authors contributed to the paper. P.-Y.W. wrote the manuscript with the supervision from T.-L.L. and J.-J.Y. is responsible for the hardware design of the secure communication system.

Funding: This work was financially supported by the Ministry of Science and Technology, Taiwan, under grant MOST-105-2221-E-006-103-MY2 and MOST-107-2221-E-366 -002 -MY2.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ott, E.; Grebogi, C.; Yorke, J.A. Controlling Chaos. *Phys. Rev. Lett.* **1990**, *64*, 2837. [CrossRef]
2. Yan, J.J.; Chen, C.Y.; Tsai, J.S.H. Hybrid chaos control of continuous unified chaotic systems using discrete rippling sliding mode control. *Nonlinear Anal. Hybrid Syst.* **2016**, *22*, 276–283. [CrossRef]
3. Alexander, J.; Chen, G.; Gauthier, A. A parameter-perturbation method for chaos control to stabilizing UPOs. *IEEE Trans. Circuits Syst. II Express Briefs* **2015**, *62*, 407–411.
4. Adel Ouannas, A.; Debbouche, N.; Wang, X.; Pham, V.T.; Zehrour, O. Secure Multiple-Input Multiple-Output Communications Based on F–M Synchronization of Fractional-Order Chaotic Systems with Non-Identical Dimensions and Orders. *Appl. Sci.* **2018**, *8*, 1746. [CrossRef]
5. Hoz, M.Z.D.; Acho, L.; Vidal, Y. A modified Chua chaotic oscillator and its application to secure communications. *Appl. Math. Comput.* **2014**, *247*, 712–722.
6. Lin, J.S.; Huang, C.F.; Liao, T.L.; Yan, J.J. Design and implementation of digital secure communication based on synchronized chaotic systems. *Digit. Signal Process.* **2010**, *20*, 229–237. [CrossRef]
7. Hua, Z.; Zhou, Y.; Pun, C.M.; Chen, C.P. 2D Sine Logistic modulation map for image encryption. *Inf. Sci.* **2015**, *297*, 80–94. [CrossRef]
8. Ye, G.; Huang, X. A feedback chaotic image encryption scheme based on both bit-level and pixel-level. *J. Vib. Control* **2015**, *22*, 1171–1180. [CrossRef]
9. Pecora, L.M.; Carroll, T.L. Synchronization in chaotic systems. *Phys. Rev. Lett.* **1990**, *64*, 821–824. [CrossRef] [PubMed]
10. Lee, S.M.; Ji, D.H.; Park, J.H.; Won, S.C. $H\infty$ synchronization of chaotic systems via dynamic feedback approach. *Phys. Lett. A* **2008**, *372*, 4905–4912. [CrossRef]
11. Park, J.H.; Ji, D.H.; Won, S.C.; Lee, S.M. $H\infty$ synchronization of time-delayed chaotic systems. *Appl. Math. Comput.* **2008**, *204*, 170–177.
12. Kuo, C.L. Design of a fuzzy sliding-mode synchronization controller for two different chaos systems. *Comput. Math. Appl.* **2011**, *61*, 2090–2095. [CrossRef]
13. Kuo, C.L.; Yan, J.J. Fuzzy sliding mode control for a class of chaos synchronization with uncertainties. *Int. J. Nonlinear Sci. Numer. Simul.* **2006**, *7*, 333–338.
14. Aghababa, M.P.; Aghababa, H.P. A general nonlinear adaptive control scheme for finite-time synchronization of chaotic systems with uncertain parameters and nonlinear inputs. *Nonlinear Dyn.* **2012**, *69*, 1903–1914. [CrossRef]
15. Liu, B.; Li, J.P.; Zheng, W. Synchronization and Adaptive Anti-Synchronization Control for Lorenz Systems under Channel Noise with Applications. *Asian J. Control* **2013**, *15*, 919–929. [CrossRef]
16. Kautz, R. *Chaos: The Science of Predictable Random Motion*; Oxford University Press: Oxford, UK, 2010.
17. Zhou, Q.; Liao, X.; Wong, K.W.; Yue, H.U.; Xiao, D. True random number generator based on mouse movement and chaotic hash function. *Inf. Sci.* **2009**, *179*, 3442–3450. [CrossRef]

18. Rukhin, A.; Soto, J.; Nechvatal, J.; Smid, M.; Barker, E.; Leigh, S.; Levenson, M.; Vangel, M.; Banks, D.; Heckert, A.; et al. *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*; NIST Special Publication 800-22 Revision 1a; National Institute of Standards & Technology: Gaithersburg, MD, USA, 2010.
19. *IEEE Standard for Binary Floating-Point Arithmetic*; IEEE Std 754-1985; The Institute of Electrical and Electronics Engineers: New York, NY, USA, 1985.