Hindawi Publishing Corporation
VLSI Design
Volume 2014, Article ID 923618, 9 pages
http://dx.doi.org/10.1155/2014/923618

Research Article

High Throughput Pseudorandom Number Generator Based on Variable Argument Unified Hyperchaos

Kaiyu Wang,[1] Qingxin Yan,[1] Shihua Yu,[2] Xianwei Qi,[1] Yudi Zhou,[1] and Zhenan Tang[1]

[1] *Department of Electronic Engineering, Dalian University of Technology, Gaoxinyuanqu Linggong Road 2, Dalian 116024, China*
[2] *School of Computer Science and Technology, Hulunbuir College, Xuefu Road, Hulunbuir 021008, China*

Correspondence should be addressed to Zhenan Tang; qsyanqingxin@126.com

Received 3 April 2014; Revised 16 June 2014; Accepted 16 June 2014; Published 7 July 2014

Academic Editor: Marcelo Lubaszewski

This paper presents a new multioutput and high throughput pseudorandom number generator. The scheme is to make the homogenized Logistic chaotic sequence as unified hyperchaotic system parameter. So the unified hyperchaos can transfer in different chaotic systems and the output can be more complex with the changing of homogenized Logistic chaotic output. Through processing the unified hyperchaotic 4-way outputs, the output will be extended to 26 channels. In addition, the generated pseudorandom sequences have all passed NIST SP800-22 standard test and DIEHARD test. The system is designed in Verilog HDL and experimentally verified on a Xilinx Spartan 6 FPGA for a maximum throughput of 16.91 Gbits/s for the native chaotic output and 13.49 Gbits/s for the resulting pseudorandom number generators.

1. Introduction

Pseudorandom number (PN) is the 01 sequence which has the randomness similar to noise. It has been widely used in digital communication, cryptography, computer games, and numerical computation [1–3]. Chaos is the phenomenon which shows very complex nonlinear dynamic characteristics in a deterministic system. And it has excellent properties such as nonperiodicity, broad bandwidth, and sensitivity to initial value [4, 5]. So Chaos and PN have a natural link. And compared to other PN sequences like m sequences, and so forth, the PN sequence generated by chaotic system has advantages like larger key space, longer cycle, and so forth.

Currently, researches of chaotic pseudorandom number generator (PRNG) are more focused on the digital implementation of low dimensional chaos such as Logistic chaos, Tent chaos, and Lorenz chaos. While these algorithms have significant advantages in some respects, like simpler construction, fewer resources consuming, and faster computing speed, they also have the fatal weakness that cannot be ignored to PRNG like smaller secret key space, periodic problem, and relatively lower throughput. Therefore, implementing a PRNG based on higher-order chaos equations seems more advantage because the hyperchaos has multiple positive Lyapunov exponent and more controllable parameters and the output of system will have more complex randomness. The hyperchaotic encryption signal is harder to decode than low dimensional encryption signal [6]. And hyperchaos can provide multiple outputs, improve the throughput, and process multiple target signal [7, 8].

In 2002, Lu et al. proposed the unified chaos that can make Lorenz chaos, Lu chaos, and Chen chaos into a unified chaotic system and realize continued transition from one to another [9]. In 2011, Ma and Wang proposed the unified hyperchaos [10]. This algorithm makes the system continued transition from Lorenz hyperchaos through Lu hyperchaos to Chen hyperchaos with one system parameter changing from 0 to 1.

In this paper, we propose a novel variable parameters hyperchaotic PRNG structure which is composed by homogenized Logistic chaos and unified hyperchaos cascade. As [10] proposed the structure that needs to vary the system parameter from 0 to 1 to change chaotic class, and Logistic chaotic output is exactly between 0 and 1, so that they have a natural link. This paper uses the homogenization algorithm proposed in [11] to deal with Logistic output to provide

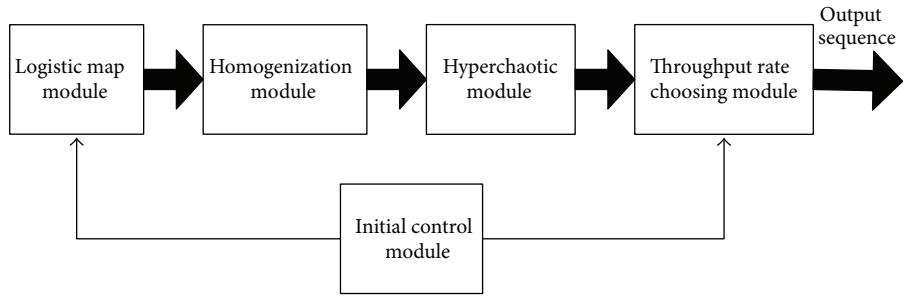

Figure 1: Flowchart of chaotic system.

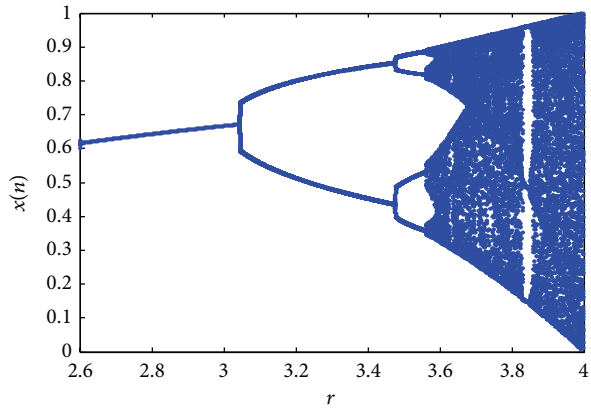

Figure 2: Logistic map bifurcation diagram.

variable parameters to the unified hyperchaos in [10]. With this method we can extend cycle of pseudorandom sequence and increase the complexity of system. And through the simple XOR processing to the output of hyperchaos, the system can generate multiple new pseudorandom sequences, greatly improving throughput. The system is designed in Verilog HDL and experimentally verified on a Xilinx Spartan 6 FPGA for a maximum throughput of 16.91 Gbits/s for the native chaotic output and 13.49 Gbits/s for the resulting PRNG output. And the output channel is increased to 26 roads. The output sequence is shown to pass the NIST SP. 800-22 test suite [12] and DIEHARD test suite to indicate statistical randomness.

This paper is organized as follows: Section 2 discusses the algorithms composed of the variable argument hyperchaos and demonstrates chaotic nature; Section 3 describes the details of its implementation in hardware; Section 4 introduces the test results of the output sequence and the resource consuming after FPGA implementation; Section 5 is conclusion.

2. Proposed Variable Argument Unified Hyperchaotic PRNG

As shown in Figure 1, the proposed variable argument unified hyperchaotic PRNG is mainly composed of five modules. They are the Logistic map module which provides the parameter, homogenization module which homogenizes the output of Logistic map, unified hyperchaotic module, throughput rate choosing module which controls the number of output channel, and initial control module which controls the system. The core algorithms are the Logistic chaos, homogenization algorithm, and unified hyperchaos. Now we will discuss these three algorithms.

2.1. Logistic Chaos. Logistic chaos is one of the most studied chaos systems. It is applied in many chaos systems because of its simple description. The Logistic chaos is described as follows:

$$x_{n+1} = r x_n (1 - x_n)$$
$$0 \le x_n \le 1, \quad 0 \le r \le 4, \quad n = 0, 1, 2, \ldots \tag{1}$$

Iteration of Logistic chaos is affected by parameters r and initial value x_0. Small changes of the two values will lead to significantly different output. When r is in the range $3.569945672 \le r \le 4$, the numbers generated in successive iterations of the mapping become chaotic, and output is always between $[0, 1]$, just as bifurcation diagram Figure 2. We take $r = 4$ to realize in hardware easily.

2.2. Homogenization Algorithm. The Logistic chaotic output in this research is homogenized to make it become a uniform pseudorandom sequence, so that the parameters input into the unified hyperchaos can be more complex and more randomness. Now, we will introduce the transform method.

The IEEE double format consists of three fields: a 52-bit fraction, f; an 11-bit biased exponent, e; and a 1-bit sign, s; then, any real number can be expressed as the following equation:

$$x = ((-1)^s \times 2^{e-1023} \times 1.f)_{10} = (\{s, e, f\})_2. \tag{2}$$

Definition 1. Left-shift b-operation of f, $f_{\leftarrow b}$, is a new fraction obtained by discarding the left-most b bits of f and then padding the result with $b - 1$ bits 0 and 1 bit 1 on the right, if the 51st-bit, 50th-bit, ..., $(51 - b + 1)$th-bit in f equals zeroes, while the $(51 - b)$th-bit in f equals one.

Definition 2. The bit-transformation of f, BT$\{f\}$: in (1), the fraction f can be rewritten in the binary-coded form

$f = f_{51}f_{50}\cdots f_1 f_0$. Parse f into higher 26-bit block f_H and lower 26-bit block f_L is as follows:

$$f_H = f_{51}f_{50}\cdots f_{27}, \qquad f_L = f_{26}f_{25}\cdots f_0. \tag{3}$$

Then, reverse f_L into f_L'; that is,

$$f_L' = f_0 f_1 \cdots f_{26}. \tag{4}$$

Then

$$\mathrm{BT}\{f\} = \left\{f_H \oplus f_L', f_L\right\}. \tag{5}$$

Now, one defines (5) as bit-transformation of f, BT$\{f\}$.

Definition 3. Bit-transformation of real numbers: suppose $x = \{s, e, f\} \in G$, G represent all real numbers. The bit-transformation of $\{s, e, f\}$ is defined by

$$\mathrm{RBT}\{x\} = \left\{0, 1023 - b, \mathrm{BT}\left\{f\right\}_{\leftarrow b}\right\}. \tag{6}$$

Note that a bit-transformation of real numbers is composed of a bit-transformation and a left-shift b-operation, so RBT$\{x\}$ is a multiple-to-one map function.

After the conversion like (6), the pseudorandom sequence can be made uniform. One realizes the homogenization algorithm on FPGA to deal with Logistic chaos. Import the output of Logistic chaos with preprocessing and postprocessing into MATLAB. The result is shown in Figure 3. It has obtained the good effect of homogenization and achieves the goal of the interference transformation, homogenization.

2.3. Unified Hyperchaos. The unified hyperchaotic system is shown as the following equations:

$$\dot{x} = (26a + 10)(y - x) \tag{7a}$$

$$\dot{y} = (28 - 44a)x - xz + (29a - 1)y - v \tag{7b}$$

$$\dot{z} = xy - \frac{(8 + a)z}{3} \tag{7c}$$

$$\dot{v} = 0.1(1 - a)yz + ax + 0.2. \tag{7d}$$

Obviously, when the parameter a increases from 0 to 1, the systems (7a), (7b), (7c), and (7d) evolve from hyperchaotic Lorenz system to hyperchaotic Chen system. The maximum Lyapunov exponent (MLE) and the Lyapunov dimension (D_L) are often used to measure a chaotic system in a state of chaos case or period orbit case. It is well known that the MLE and D_L satisfy at least one MLE greater than zero and $2 < D_L < 3$ for chaos case, two MLE greater than zero, and $3 < D_L < 4$ for hyperchaos case. For systems (7a), (7b), (7c), and (7d), when $a \in [0, 1]$, the Lyapunov exponent spectrum and the Lyapunov dimension are shown as Figures 4 and 5. As it is shown in Figure 4, all points from 0 to 1 except $a = 0.14$, there are two MLE greater than zero. And as it is shown in Figure 5, all points from 0 to 1 except $a = 0.14$, $3 < D_L < 4$. It means the system is hyperchaotic system only except individual parameter points. And the individual bad parameter point can be removed by the means of hardware implementation.

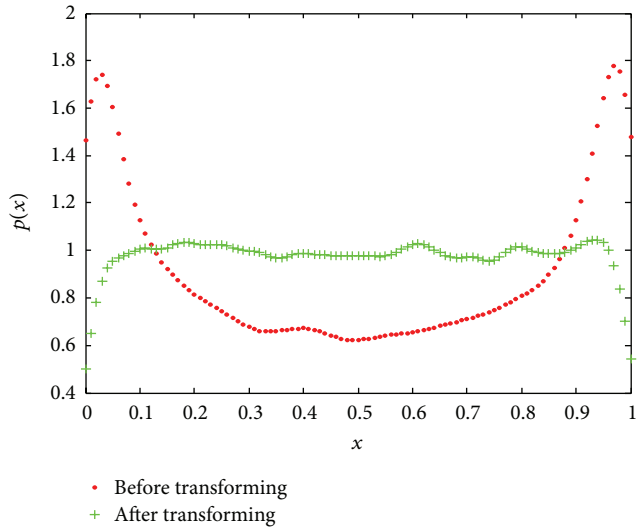

Figure 3: Homogenization effect.

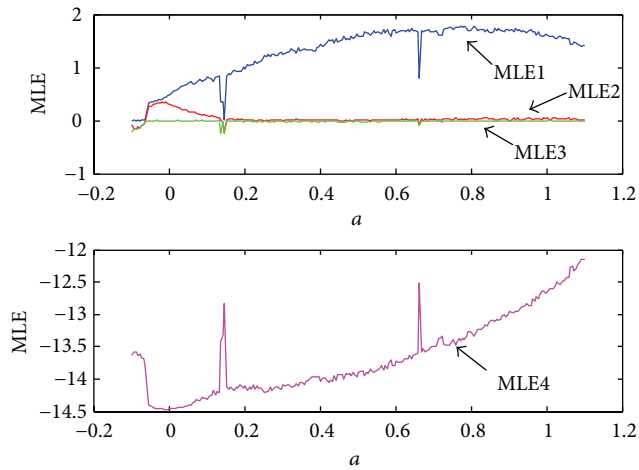

Figure 4: Max Lyapunov exponent spectrum.

In order to see clearly that, when the parameter a increases from 0 to 1, the systems (7a), (7b), (7c), and (7d) evolve from hyperchaotic Lorenz system to hyperchaotic Chen system, we plot the phase diagram with different parameter as shown in Figures 6 and 7.

2.4. Variable Argument Unified Hyperchaotic Algorithm. The variable argument unified hyperchaotic PRNG we proposed is based on the above three algorithms. Logistic chaos generates sequence between 0 and 1. Then, the sequence is processed by homogenization algorithm to be made uniform. After that, the uniform pseudorandom sequence is introduced as changing parameter to be imported into unified hyperchaotic system to control the output. So that the system varies in different state of hyperchaotic system and increases

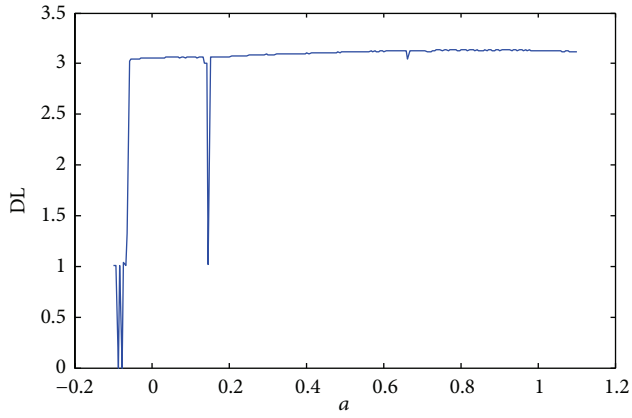

Figure 5: Lyapunov dimension.

the output sequence cycle and has more complex dynamic characteristics and optimizes the statistical properties.

3. Hardware Implementation

To implement easily in the hardware, the differential equations (7a), (7b), (7c), and (7d) are discretized. Euler approximation has been shown to provide the best chaotic response, occupy the lowest area, and provide the highest speed compared with Runge-Kutta method and other methods [13]. Therefore, the Euler approximation is applied to discretize the continuous-time systems (7a), (7b), (7c), and (7d) for the digital domain:

$$x_{n+1} = h\left((26a + 10)\left(y_n - x_n\right)\right) + x_n \tag{8a}$$

$$y_{n+1} = h\left((28 - 44a)\, x_n - x_n z_n + (29a - 1)\, y_n - v_n\right) + y_n \tag{8b}$$

$$z_{n+1} = h\left(x_n y_n - \frac{(8 + a)\, z_n}{3}\right) + z_n \tag{8c}$$

$$v_{n+1} = h\left(0.1\,(1 - a)\, y_n z_n + a x_n + 0.2\right) + v_n. \tag{8d}$$

While the chaotic systems are running in finite precision, the fixed-point arithmetic is preferable over floating point mathematics because it requires less hardware resources and computation time. And under the same word length fixed-point format has a higher accuracy [14, 15]. So we select fixed-point format to represent data. Also due to the limited precision, the digital realization of chaotic systems has degradation dynamics and tends to period orbit case, namely, finite precision. Based on the theory [16] proposed that the cycle of chaotic sequence will grow exponentially with growth of format word length, we use 32-bit fixed-point number format to realize the chaotic system to prevent the finite precision effect. In unified hyperchaotic system, the fixed-point two's complement format is used with the 7 most significant bits for sign and integer part and the remaining for the fractional part. But Logistic chaotic system is used with the 1 most significant bit for integer part and the remaining for the fractional part as its output is always positive number.

As shown in Figure 1, the PRNG we proposed has three core algorithms: Logistic, homogenized, and unified hyperchaotic. And they are cascade structure. Therefore, this work employs a pipelined architecture between the three modules, so that the register between these three modules can be updated in each clock and increase hardware utilization efficiency.

Logistic Module is controlled by control module so that the Logistic Module outputs the same value in m clock cycles. As a result of the pipeline structure, unified hyperchaotic module will read an input as a unified hyperchaotic parameter at each rising edge of the clock, so unified hyperchaotic modules will calculate the output with the same parameter in every m clock cycles. So, if Logistic period is n, then the entire system's period is $m * n$.

As unified hyperchaotic module has four dimensional outputs, it could provide operation space for subsequent processing. We add throughput rate choosing module after unified hyperchaotic module to make bitwise operation among the initial four outputs (x, y, z, v). Based on the conclusion proposed in [7] about the fact that doing bitwise XOR operation on chaotic system output can get better PN sequence, in throughput rate choosing module, we do bitwise XOR operation on two different output sequences $A \oplus B$ (like $x \oplus y$, $x \oplus z$, etc.) or three different output sequences $A \oplus B \oplus C$ (like $x \oplus y \oplus z$, $x \oplus z \oplus v$, etc.) or do bitwise XOR operation on A's higher 16 bits and B's lower 16 bits and then merge these 16 bits with A's lower 16 bits $\{A_{\text{high}} \oplus B_{\text{low}}, A_{\text{low}}\}$ (like $\{x_{[31:16]} \oplus y_{[15:0]}, x_{[15:0]}\}$, etc.) to improve the throughput. After these XOR operations, this system can provide up to 26 channels output. And this module can be configured to decide which channel or which several channels can be output.

4. Result

The proposed variable argument unified hyperchaotic PRNGis designed in Xilinx ISE 12.2 environment using Verilog HDL and experimentally verified on a Xilinx Spartan 6 XC6SLX100 FPGA. In order to fully test the output, through controlling the throughput rate choosing module, make all XOR modules work to output all 26 channels and analyze these output data with below tests.

4.1. Phase Diagram. We import the PRNG's output $\{x, y, z, v\}$ into MATLAB as shown in Figure 8. From Figure 8 we can get the system is switching in different chaotic system with the number of iterations increase and parameter change; it effectively improves the complexity of output.

4.2. Correlation. Among the 26 channels output, there are 22 channels which are produced by XOR operation through 4 original outputs (x, y, z, v). To analyze the cross-correlation among these outputs, we import them into MATLAB. Figure 9 shows cross-correlation result between x related outputs $(x \oplus y, x \oplus z, x \oplus v, x \oplus x \oplus z)$ and original x output. The results show that the XOR operation outputs are still correlated to their original channels, as indicated by a peak at zero-lag.

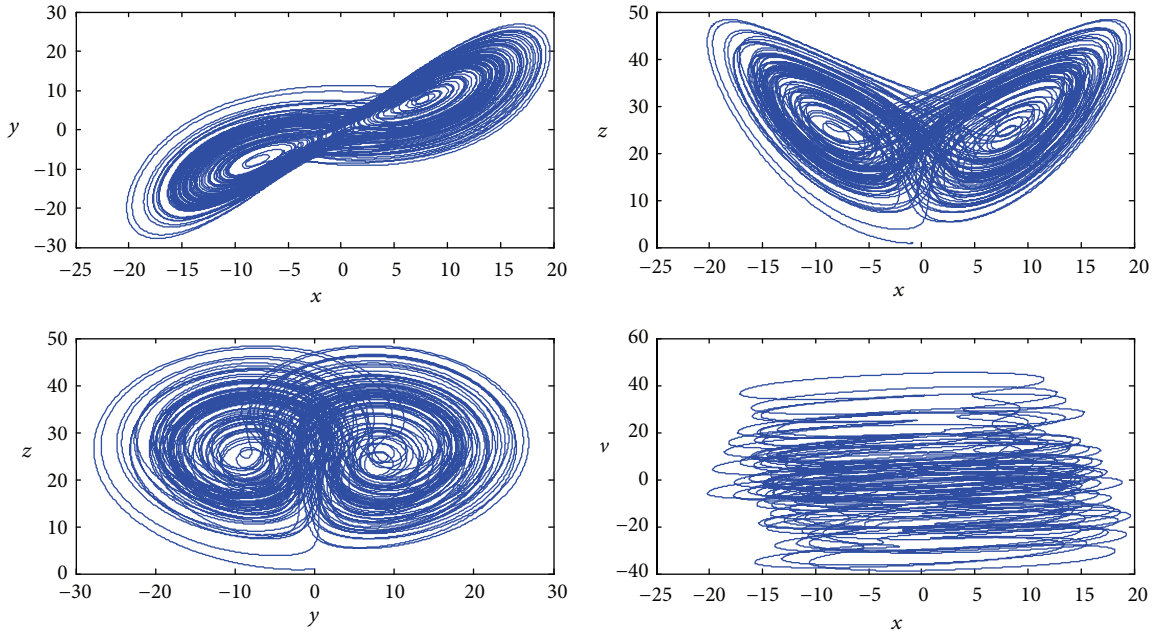

Figure 6: Phase diagram when $a = 0.03$.

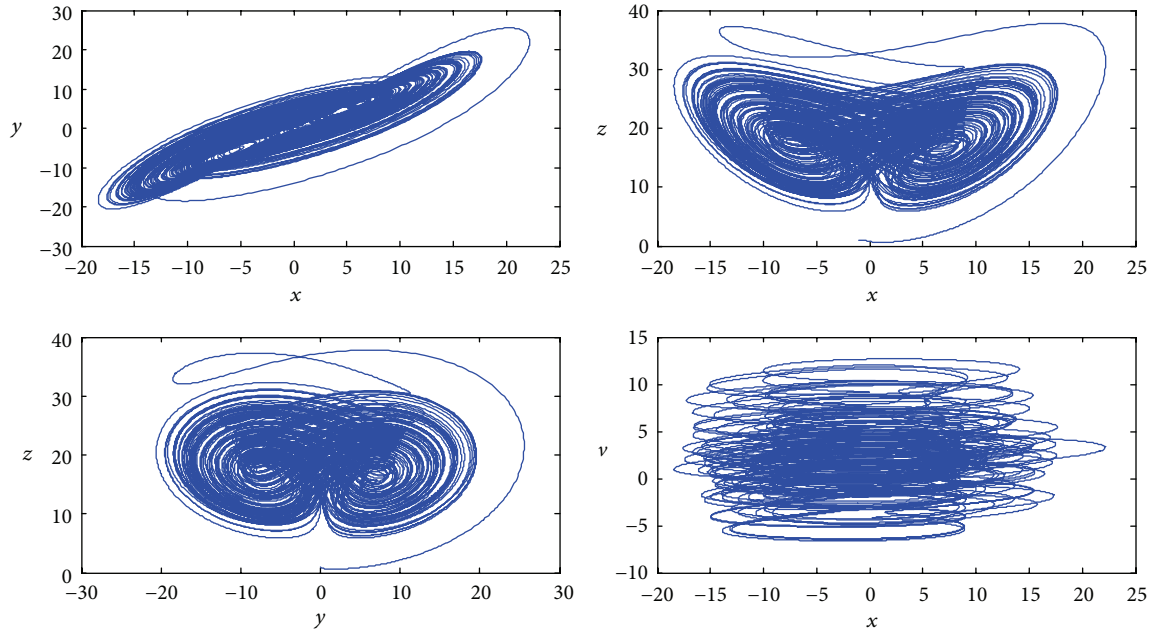

Figure 7: Phase diagram when $a = 0.73$.

However, most peaks are below 0.5, and at other delays, cross-correlation coefficients are below 0.3. About other cross-correlation results, they are similar to the above one.

4.3. Pseudorandom Number Test. If the chaotic system is regarded as a PRNG, not all output bits can meet the requirements of randomness. As in the digital context, it creates an uneven distribution of pseudorandomness across the output bits. The MSBs are not only biased but also highly correlated, while the LSBs show desirable statistical randomness [17]. For this kind of situation, we first test all 32 bits. If the sequence cannot pass the tests, we will discard the highest one. Then, we will test the remaining bits. Repeat these actions until the sequence can pass tests. After tested by NIST SP800-22 test suit and DIEHARD test suit, test results show that in the 26 channels of FPGA output, 12 channels

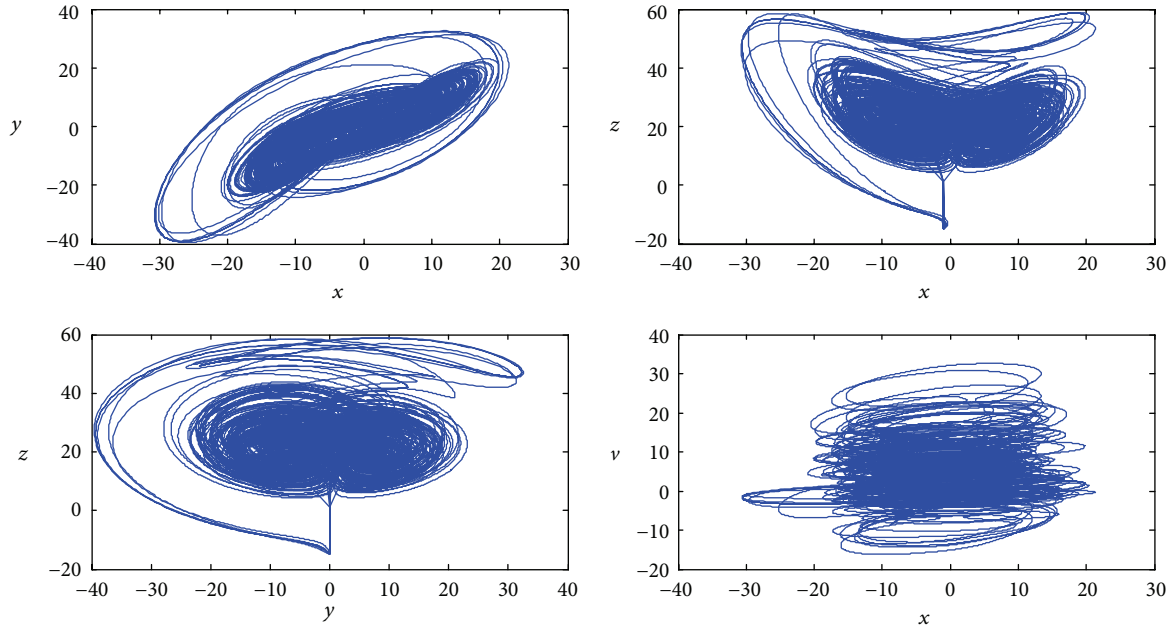

Figure 8: System output phase diagram.

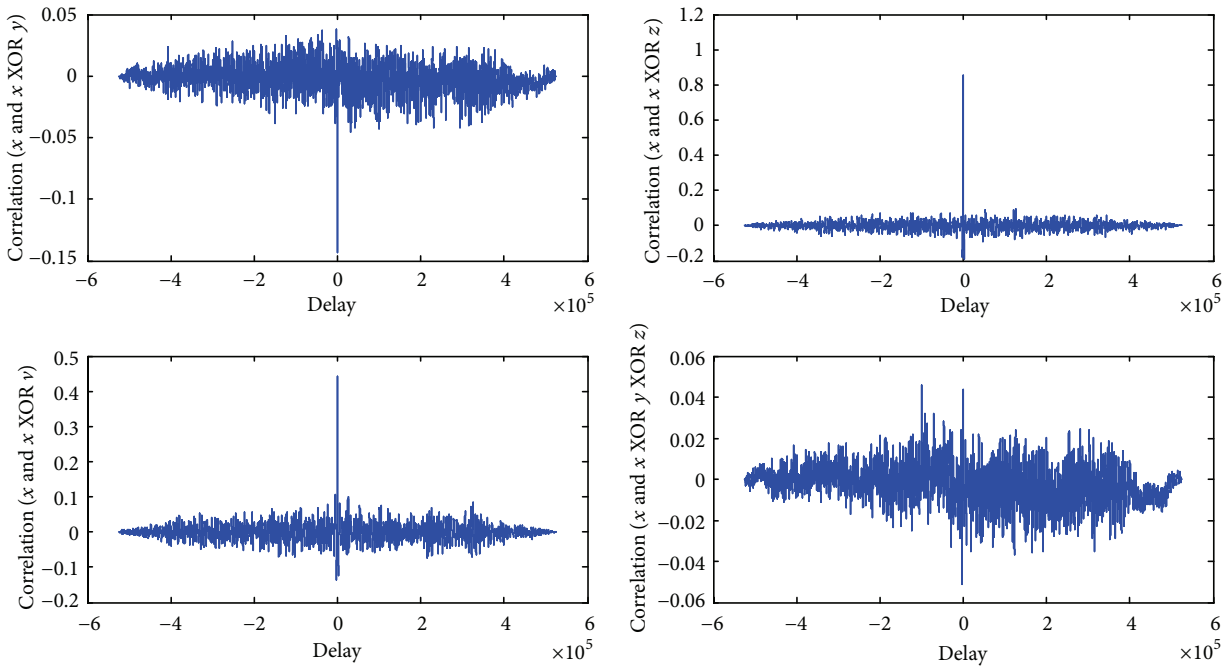

Figure 9: X related cross-correlation diagram.

$\{A_{\text{high}} \oplus B_{\text{low}}, A_{\text{low}}\}$ can pass tests with all 32 bits and other 14 channels can pass with lower 20 bits. So, as a PRNG, we make the system output 32 bits in 12 channels $\{A_{\text{high}} \oplus B_{\text{low}}, A_{\text{low}}\}$ and output lower 20 bits in other 14 channels. And from the test result of DIEHARD, we can get the conclusion that the quality from higher to lower is $\{A_{\text{high}} \oplus B_{\text{low}}, A_{\text{low}}\}$, $A \oplus B \oplus C$, $A \oplus B$ and original 4 outputs.

We take the lower 20 bits $x, y, z \oplus v, x \oplus y \oplus z$ NIST test results and DIEHARD test results as the representative list in this paper, in Tables 1 and 2. The remaining 22 roads also pass the tests but not list in this paper. In conclusion, the PRNG we proposed can provide as high as 26 channels output and 13.49 Gbits/s throughput.

4.4. Hardware Resource Utilization. After the FPGA synthesis, Slice Registers resources utilization rate is 1%, Slice LUTs is 5%, and DSP block is 55%. The designed system can

Table 1: Result of NIST SP800-22 test.

	X P value	Y P value	Z XOR V P value	X XOR Y XOR Z P value
Frequency	0.791780	0.041550	0.979257	0.961716
Block frequency	0.366274	0.648388	0.367493	0.117584
Cumulative sums	0.744906	0.078225	0.664299	0.985758
Runs	0.383170	0.481335	0.872882	0.017885
Longest run of ones	0.753431	0.746675	0.173653	0.222029
Rank	0.123860	0.050112	0.202915	0.849442
FFT	0.168669	0.868803	0.868803	0.934178
Nonperiodic template matchings	0.988648	0.677613	0.896375	0.857033
Overlapping template matchings	0.545254	0.239949	0.392905	0.672648
Universal statistical	0.303032	0.748731	0.096754	0.951902
Approximate entropy	0.977511	0.223329	0.790074	0.634836
Random excursions	0.379538	0.707823	0.490369	0.981034
Random excursions variant	0.509760	0.958761	0.613635	0.937700
Serial	0.380005	0.582563	0.953135	0.876081
linear complexity	0.418562	0.193589	0.352896	0.198838
TEST status	SUCCESS	SUCCESS	SUCCESS	SUCCESS

Length of bit = 1000000, number of bit streams = 100, and confidence level = 99%.

Table 2: Result of DIEHARD test.

	X P value	Y P value	Z XOR V P value	X XOR Y XOR Z P value
BIRTHDAY SPACINGS	0.352976	0.728427	0.767687	0.293804
OVERLAPPING 5-PERMUTATION	1.000000	1.000000	1.000000	0.999995
RANK TEST for 31 × 31 matrices	0.364438	0.610135	0.417598	0.741397
RANK TEST for 32 × 32 matrices	0.594228	0.321434	0.431023	0.514951
RANK TEST for 6 × 8 matrices	0.335467	0.156574	0.227157	0.688417
OVERLAPPING 20-tuples BITSTREAM	0.6421225	0.413186	0.5047115	0.453427
OPSO	0.4453782	0.545113	0.552886	0.470565
OQSO	0.8444143	0.830028	0.720432	0.473571
DNA	0.8535612	0.902319	0.650219	0.597742
COUNT-THE-1's TEST on a stream of bytes	0.3500190	0.210574	0.306527	0.183018
COUNT-THE-1's TEST for specific bytes	0.7102660	0.641749	0.623289	0.490400
PARKING LOT	0.643434	0.068029	0.787648	0.837780
MINIMUM DISTANCE	0.689712	0.467954	0.970672	0.960276
3DSPHERES	0.130624	0.701084	0.231284	0.013265
SQEEZE	0.952062	0.781055	0.454853	0.518254
OVERLAPPING SUMS	0.862138	0.132331	0.913215	0.724329
RUNS	0.578362	0.298342	0.834721	0.302906
CRAPS	1.000000	1.000000	0.679313	0.841946

provide 26-way output with 32 bits. Taking into account the pipeline structure influence, the throughput of system is as high as 16.91 Gbits/s at maximum clock frequency. And its random number throughput rate is as high as 13.49 Gbits/s. Specific numbers of resource consumption and throughput are shown in Table 3. To make the quantitative analysis on the resource consumption and throughput, we adopt the following definition. Gate count is estimated as $\mathrm{Gc} = 8 \times (\mathrm{LUTs} + \mathrm{FFs})$ and the area efficiency is assessed through a figure of merit determined as $\mathrm{FOM} = (N_{\mathrm{PRNG}} \times f_{\mathrm{CLK}})/\mathrm{Gc}$. The PRNG proposed in this paper is compared with several low dimensional chaotic PRNG in Table 4. From Table 4 we can

Table 3: Experimental results on Xilinx Spartan 6 FPGA.

	Experimental results on the Xilinx Spartan 6 FPGA
Registers	1780
LUTs	3068
Fully used LUT-FF pairs	854
DSP48A1s	100
Frequency (MHz)	20.811
Single output bits	32
Number of output channel	26
System throughput (Mb/s)	17314.752
PRNG throughput (Mb/s)	13818.504

Table 4: Comparison with previously reported chaos-based PRNGS.

System		Area (Gc)	T.put	FOM	NIST
Chen et al. [18]	LOG.Map	9622	200	0.02	Pass
Li et al. [19]	LOG.Map	9136	200	0.02	Fail
Chen et al. [20]	LOG.Map	31655	3200	0.10	Pass
This one	Hyp.Chaos	31376	13819	0.44	Pass

get that although this work spent higher hardware resource than low dimensional chaotic PRNG, we get much higher throughput and higher FOM.

5. Conclusion

In this paper, we propose a novel variable parameters hyperchaotic PRNG structure which is composed of homogenized Logistic chaos and unified hyperchaos cascade. Take the homogenized Logistic chaotic output as the unified hyperchaotic parameter to make the output sequence in different chaotic system. In this way, system will be more complex and have longer period. At the same time, add a throughput rate control module after the output of the unified hyperchaotic module, through simple XOR processing; the output of the 4 road hyperchaos can be extended to 26 road and greatly improve the throughput of the system. The PRNGis designed in Xilinx ISE 12.2 environment using Verilog HDL and experimentallyverified on a Xilinx Spartan 6 FPGA. The throughput is up to 16.91 Gbits/s for the chaotic system. As a PRNG, it can provide 26 channels output as pseudorandom sequence which all pass NIST SP800-22 test and DIEHARD test. And its random number throughput rate is as high as 13.49 Gbits/s.

Therefore, due to the variable argument unified hyperchaotic PRNG has advantages like high output complexity, multidimensional output, and high throughput rate; it is very suitable for being applied to multiobjective signal processing field like multiobjective control and secure communications, and so forth.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] Q. Rong and Y. Fang, "Pseudo random sequence generator based on variable structure chaos," *Modern Electronics Technique*, vol. 35, no. 11, pp. 64–67, 2012.

[2] H. P. Ren, *The design of chaotic key system based on FPGA [M.S. dissertation]*, Dalian Maritime University, Ganjingzi, China, 2011.

[3] P. Dabal and R. Pelka, "FPGA implementation of chaotic pseudo-random bit generators," in *Proceedings of the 19th International Conference on Mixed Design of Integrated Circuits and Systems (MIXDES '12)*, pp. 260–264, Warsaw, Poland, May 2012.

[4] P. Li, Z. Li, W. A. Halang, and G. Chen, "A multiple pseudorandom-bit generator based on a spatiotemporal chaotic map," *Physics Letters A*, vol. 349, no. 6, pp. 467–473, 2006.

[5] K. Wang, W. Pei, H. Xia, and Y. Cheung, "Pseudo-random number generator based on asymptotic deterministic randomness," *Physics Letters A*, vol. 372, no. 24, pp. 4388–4394, 2008.

[6] H. Wang, L. Cheng, and J.-H. Peng, "Application of hyperchaos to encrypting digital signals," *Journal of Northeast Normal University (Natural Science Edition)*, vol. 32, no. 2, pp. 31–35, 2000.

[7] A. S. Mansingka, M. Affan Zidan, M. L. Barakat, A. G. Radwan, and K. N. Salama, "Fully digital jerk-based chaotic oscillators for high throughput pseudo-random number generators up to 8.77 Gbits/s," *Microelectronics Journal*, vol. 44, no. 9, pp. 744–752, 2013.

[8] F. Jin-Qing, "Several advances in chaos-based communication and research of information security associated with networks," *Journal of Systems Engineering*, vol. 25, no. 6, pp. 725–741, 2010.

[9] J. H. Lu, G. R. Chen, and D. Z. Cheng, "Bridge the gap between the Lorenz system and the Chen system," *International Journal of Bifurcation and Chaos*, vol. 12, no. 12, pp. 2917–2926, 2002.

[10] C. Ma and X. Wang, "Bridge between the hyperchaotic Lorenz system and the hyperchaotic Chen system," *International Journal of Modern Physics B*, vol. 25, no. 5, pp. 711–721, 2011.

[11] S. Li-Yuan, X. Yan-Yu, and Z. Sheng, "How homogenize Chaos-based Pseudo-random sequences," in *Proceedings of the International Conference on Computer Science and Software Engineering (CSSE '08)*, pp. 793–796, December 2008.

[12] A. Rukhin, J. Soto, M. Smid et al., "A statistical test suite for random and pseudorandom number generators for cryptographic applications," NIST Special Publication 800-22, 2010.

[13] M. A. Zidan, A. G. Radwan, and K. N. Salama, "The effect of numerical techniques on differential equation based chaotic generators," in *Proceedings of the 23rd International Conference on Microelectronics (ICM '11)*, pp. 1–4, Hammamet, Tunisia, December 2011.

[14] A. Pande and J. Zambreno, "A chaotic encryption scheme for real-time embedded systems: design and implementation," *Telecommunication Systems*, vol. 52, no. 2, pp. 551–561, 2013.

[15] L. Wang, W. Liu, H. Shi, and J. M. Zurada, "Cellular neural networks with transient chaos," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 54, no. 5, pp. 440–444, 2007.

[16] B. Zhang, *Performance analysis and optimization of chaotic PN sequence [M.S. thesis]*, Hangzhou Dianzi University, Jianggan, China, 2009.

[17] M. L. Barakat, A. S. Mansingka, A. G. Radwan, and K. N. Salama, "Generalized hardware post-processing technique for chaos-based pseudorandom number generators," *ETRI Journal*, vol. 35, no. 3, pp. 448–458, 2013.

[18] S. Chen, T. Hwang, and W. Lin, "Randomness enhancement using digitalized modified logistic map," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 57, no. 12, pp. 996–1000, 2010.

[19] C.-Y. Li, T.-Y. Chang, and C.-C. Huang, "A nonlinear PRNG using digitized logistic map with self-reseeding method," in *Proceedings of the International Symposium on VLSI Design, Automation and Test (VLSI-DAT '10)*, pp. 108–111, April 2010.

[20] S.- L. Chen, T. Hwang, S.-M. Chang, and W.-W. Lin, "A fast digital chaotic generator for secure communication," *International Journal of Bifurcation and Chaos in Applied Sciences and Engineering*, vol. 20, no. 12, pp. 3969–3987, 2010.