Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2013, Article ID 285373, 7 pages
http://dx.doi.org/10.1155/2013/285373

Research Article

Random Numbers Generated from Audio and Video Sources

I-Te Chen

Department of Healthcare Administration & Medical Informatics, Kaohsiung Medical University, 100 Shih-Chuan 1st Road, Kaohsiung 80708, Taiwan

Correspondence should be addressed to I-Te Chen; yiter.chen@gmail.com

Received 28 January 2013; Revised 20 March 2013; Accepted 21 March 2013

Academic Editor: Wang Xing-yuan

Random numbers are very useful in simulation, chaos theory, game theory, information theory, pattern recognition, probability theory, quantum mechanics, statistics, and statistical mechanics. The random numbers are especially helpful in cryptography. In this work, the proposed random number generators come from white noise of audio and video (A/V) sources which are extracted from high-resolution IPCAM, WEBCAM, and MPEG-1 video files. The proposed generator applied on video sources from IPCAM and WEBCAM with microphone would be the true random number generator and the pseudorandom number generator when applied on video sources from MPEG-1 video file. In addition, when applying NIST SP 800-22 Rev.1a 15 statistics tests on the random numbers generated from the proposed generator, around 98% random numbers can pass 15 statistical tests. Furthermore, the audio and video sources can be found easily; hence, the proposed generator is a qualified, convenient, and efficient random number generator.

1. Introduction

The security concerning the access of Cloud Database has become a significant up-to-date issue. Cryptography and network security are both fundamental components when accessing Cloud Database safely; furthermore, random numbers play an essential role in cryptography and network security. The random numbers could be generated from two major ways: true random number generators (TRNGs) and pseudorandom number generators (PRNGs). TRNGs often generate random numbers from nature phenomena such as dice, coin flipping, flip-flop circuit, oscillator, electromagnetic wave, thermal noise, and atmospheric noise. As a result, the random numbers generated from TRNGs cannot be reproduced [1]. On the other hand, PRNGs often generate random numbers from mathematical functions such as linear congruential to simulate real randomness, which allow the sender and receiver generating the same random numbers from PRNGs with the same initial value.

Most people access Cloud storage via personal devices; Hence, to create random number generator algorithms from video [2] that could be used in limited computing capacity personal devices would be significant invention. In addition, the result of [2] shows that 98% random numbers generated from IPCAM and WEBCAM can pass National Institute of Standards and Technology (NIST for short) SP 800-22 Rev.1a 15 statistical tests [3], but only 72.8% random numbers generated from a XiangSheng MPEG-1 video files can pass 15 statistical tests. The XiangSheng is a kind of comic dialogue to entertain the audience with ridiculous stories. Furthermore, once adopting the dual-video sources algorithm on the XiangSheng MPEG-1 video file to obtain random numbers, the passing rate against 15 statistical tests will rise up to 97%. Table 1 shows the comparisons of Video Random Number Generator (VRNG) and Dual-Video Random Number Generator (DVRNG). The columns of 2WEBCAM, 2IPCAM, and 2XiangSheng mean that DVRNG came from two video sources, respectively; and the Pure Sound means the single sound source TRNG.

However, passing rate of the random numbers generated from pure sound is almost zero, and using two video sources to generate random numbers is not convenient enough when compared with the random numbers generated from white noise of audio and video—Audio and Video Random

Table 1: Passing rate among VRNG, DVRNG, and pure sound.

	Webcam	Ipcam	XiangSheng	2Webcam	2Ipcam	2XiangSheng	Pure sound
Frequency	0.9900	0.9900	0.5900	1.0000	0.9800	1.0000	0.58
Block frequency	1.0000	1.0000	0.6000	0.9900	0.9900	1.0000	0
Cumulative sums	0.9900	0.9850	0.5850	1.0000	0.9750	1.0000	0.16
Runs	0.9900	0.9900	0.3000	1.0000	0.9900	1.0000	0
Longest-run-of-ones	0.9900	0.9900	0.5800	1.0000	0.9800	1.0000	0
Binary matrix rank	0.9900	0.9800	1.0000	0.9900	1.0000	1.0000	0
FFT (Fourier)	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	0
Nonoverlapping template	0.9886	0.9872	0.9295	0.9882	0.9897	0.9848	0
Overlapping template	0.9800	0.9800	0.7000	0.9900	0.9900	1.0000	0
Universal statistical	1.0000	1.0000	0.0000	1.0000	1.0000	1.0000	0
Approximate entropy	0.9800	0.9900	0.7800	0.9900	1.0000	1.0000	0
The random excursions	0.9853	1.0000	1.0000	1.0000	1.0000	1.0000	0
The random excursions variant	1.0000	1.0000	1.0000	0.9769	0.9899	1.0000	0
The serial	0.9900	0.9800	0.8500	0.9950	0.9950	1.0000	0
The linear complexity	0.9800	0.9900	1.0000	0.9700	0.9700	1.0000	0
Average of above	0.9903	0.9908	0.7276	0.9927	0.9900	0.9990	0.0493
Minimum of above	0.9800	0.9800	0.0000	0.9700	0.9700	0.9848	0

Number Generator (AVRNG as short). The AVRNG requires camera and microphone to generate random numbers, which means that AVRNG could not only be applied on personal computer but also on the more and more widespread smartphone and tablet PC.

We will review related works in Section 2, and the main contribution of the proposed algorithms, AVRNG with filter, will be described in Section 3. Section 4 will be a conclusion and followed by references.

2. Related Works

2.1. PRNGs and TRNGs. A linear congruential random number generator [4] represents one of the best-known PRNGs and was firstly broken by Reeds [5] and then by Boyar [6]. Researchers develop the feedback shift register since then [7]. In 2010, Debiao et al., proposed “A Random Number Generator Based on Isogenies Operations” [8]. They used character of elliptic curves to generate random numbers, which is also the PRNG method to pass the NIST SP 800-22 Rev.1a 15 statistical tests. Wang and Yu proposed “A Block Encryption Algorithm Based on Dynamic Sequences of Multiple Chaotic Systems,” in 2009 [9], in which algorithm makes the pseudo-random sequence possess more concealment and noise like characteristic and overcomes the periodic malpractice. Furthermore, Wang et al. proposed a parameter perturbation method based on the good property of the extended one-dimensional smooth map in 2011 [10]. And Wang et al. proposed a serial of random number generators based on chaotic system from then on [11–13]. Wang et al. overcame the periodic problem of random number and said that it is hard to obtain the truly random numbers; therefore, this work will present a method that can be used as both PRNG and TRNG.

Intel attempted to produce true random numbers by taking some of the thermal noise firstly in 1999. But, to amplify thermal noise consumes lots of power; hence, Intel turned to make a random number generator based on only digital hardware in 2008 [14]. Besides, the thermal noise and amplify circuit are not suitable for common computer users. The most popular cryptographically sound random number generator—LavaRnd—was developed in 1996 by Landon Curt Noll, Simon Cooper, and Mel Pleasant. The LavaRnd includes 3 stages: gathering digital chaos, randomizing chaos by digital blender, and outputting random data. Over million people grabbed random numbers from the LavaRnd website [15]. However, digital blender part of LavaRnd computational complexity is too high to use the SHA-1 hash function.

In 2009, Tawfeeq proposed “A Random Number Generator Based on Single-Photon Avalanche Photodiode Dark Counts” [16] which produced nearly 50% 0’s and 50% 1’s. Yamanashi and Yoshikawa proposed “Superconductive Random Number Generator Using Thermal Noises in SFQ Circuits” [17] which used the superconductive single-flux-quantum (SFQ) circuits and thermal noises to produce random numbers. Nevertheless, those two methods also require specific equipment and electronic circuit knowledge to simulate random numbers.

In 2012, Wang et al. proposed “A Novel True Random Number Generator Based on Mouse Movement and a One-Dimensional Chaotic Map” [18] which utilized the x-coordinate to be the length of an iteration segment of their true random numbers and y-coordinate to be the initial value of this iteration segment. As the result, Wang et al. made a uniform distribution random number with average passing rate 68%.

Alsultanny proposed another TRNG called “Random-bit Sequence Generation from Image Data” in 2008 [19]. Alsultanny used simple operation like XOR to generate random bits from images, but only 94% of the generated random bits could pass at least four statistical tests [20] as well as 79% pass all five tests which seems not feasible enough. Though Alsultanny's results are not good enough, this method indeed inspire is us to generate random numbers from video sources.

Tsai et al. proposed the random numbers generated from video [2] and divergence of scaling function [21] in 2009 and 2012. The result of [2] is shown in Table 1. In [2], we cannot produce qualified random numbers from single video or single audio source; further improvement by importing audio and video sources meanwhile applying the NIST SP-800-22 Rev.1a 15 statistical tests [3] to verify the randomness is proposed in this work.

2.2. Statistical Test Suite for Random Number Generators. The qualified random numbers should satisfy unpredictability. For randomness, Menezes et al. proposed five statistical tests—Frequency (Monobit) test, Serial test (two-bit test), Poker test, Runs test, and Autocorrelation test—of random sequences—in Handbook of Applied Cryptography, 1996 [20], but this verification method is not enough to approve the randomness. Therefore, NIST published Special Publication SP-800-22 (A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications) in 2001 and revised in August 2008 and April 2010 [3]. The SP-800-22 Rev.1a developed 15 statistical tests to verify the randomness of random numbers produced by either PRNGs or TRNGs. A qualified random number generator should pass all 15 tests listed in the following:

(1) The Frequency (Monobit) Test

(2) Frequency Test within a Block

(3) The Runs test

(4) Test for the longest-Run-of-Ones in a Block

(5) The Binary Matrix Rank Test

(6) The Discrete Fourier Transform (spectral) Test

(7) The Non-overlapping Template Matching Test

(8) The Overlapping Template Matching Test

(9) Maurer's "Universal Statistical" Test

(10) Linear Complexity Test

(11) The Serial test (two-bit test)

(12) The Approximate Entropy test

(13) The Cumulative Suns (Cusum) test

(14) The Random Excursions test

(15) The Random Excursions Variant test.

The above 15 statistical tests are commonly used for determining whether the random numbers possess some specific characteristics that a true random numbers could possibly exhibit. In addition, the NIST issues the 140 Publication Series to coordinate the requirements and standards for cryptographic modules. The FIPS Pub 140-1 and 140-2 were published in 1994 and 2004, respectively [22]. Revised draft FIPS Pub 140-3 adding on new security features that reflect recent advances in technology and security methods was published in 2009 [23]. FIPS pub 140 serials recommend some statistical tests and FIPS pub 800-90a serials recommend random number generator using Deterministic Random Bit Generator and providing validation system [24]. For more other statistical tests please refer to [22–25].

3. The Proposed Scheme: AVRNG with Filter

Even though random numbers that we generated from single source cannot all pass the NIST SP-800-22 Rev.1a 15 statistical tests. Consequantly this work combines audio and video to develop AVRNGs in order to take advantage of audio's influence. In our experiments, the AVRNG with cartoon video source still cannot pass NIST SP-800-22 Rev.1a 15 statistical tests due to cartoon is artificial color. The artificial color is almost the same in the neighborhood area such as arm or face. Hence, coordinate threshold and discard threshold vt, th, and a new coordinate equation to avoid generating random numbers by capturing pixels in the neighborhood area are introduced.

3.1. AVRNG with Filter Algorithm. Let W and H denote the video frame's width and height and R, G, and B be the color value of red, green, and blue, respectively. We take only from 8 to 15 frames of size 320∗200 pixels per second to ensure the randomness. And the coordinate threshold vt is a filter; if the RGB difference between current coordinate and the previous coordinate is less than vt, then adopt another coordinate. The vt could be set as any image's half variance; thus, we set vt as half variance of the video frame in the 3rd second. And we set the discard threshold $th = 100$. If we cannot get available $RGB\ (x, y)$ coordinate more than th times, then discard this frame and go to the next frame. The algorithm is shown as follows.

(a) Initial value: the initial value can be obtained anyway. In the proposed method, we take average value of the nine pixels around the center pixel of the first frame to be the initial value. Let the coordinate of center pixel be (x_c, y_c), and let the color of (x_c, y_c) be

$$\text{color}_{xc,yc} = R_c \ll 16 + G_c \ll 8 + B_c, \quad (1)$$

where $\ll$ means bit shift.

Then we set the initial value. Let R, G, B, R_1, G_1, B_1, R_2, G_2, and B_2 be zero firstly:

$$\begin{aligned} \text{color}_i = {} & \frac{\text{color}_{xc-1,yc-1} + \text{color}_{xc-1,yc} + \text{color}_{xc-1,yc+1}}{9} \\ & + \frac{\text{color}_{xc,yc-1} + \text{color}_{xc,yc} + \text{color}_{xc,yc+1}}{9} \\ & + \frac{\text{color}_{xc+1,yc-1} + \text{color}_{xc+1,yc} + \text{color}_{xc+1,yc+1}}{9}. \end{aligned} \quad (2)$$

We set initial coordinate (x, y) to be

$$x = (\text{color}_i)\,\%\left(\frac{W}{2}\right) + \frac{W}{4},$$
$$y = (\text{color}_i)\,\%\left(\frac{H}{2}\right) + \frac{H}{4}, \tag{3}$$

where "&" means AND, "⊕" means XOR, and "%" means modular arithmetic.

(b) Set threshold: we set vt as half variance of the video frame in the 3rd second. Set the discard threshold $th = 100$ and let $watchdog = 0$.

(c) Get sound coordinate and value: sound source from mpeg1 video file is transferred to 16 bits (2 bytes) stereo format of 44.1 K sampling rate. There are 30 frames/second in a video file; hence, we could obtain around 44100 ∗ 2(stereo) ∗ 2(2bytes)/30 = 5880 bytes sound data in one frame. At first, we get one sound byte from each 5880-byte source; in other word, we synchronize video frame and sound and set $K = 5880$. Since synchronization cannot derive qualified random numbers, various K values ($K = 250$, 500, 1000, 2000, 3000, 4000 and 5000) are adopted to find out the better result as when $K = 500$. Set $K = 500$; let i and j denote random number bit and sound bytes, respectively. The RGB is gotten from coordinate (x, y)

$\text{SN}_1 = \text{sound byte}_j\ [10 + (R * i + (G \ll 2) + B + \text{runcnt})\%\ (K/2)]$;

$\text{SN}_2 = \text{sound byte}_j\ [15 + (R * i + (G \ll 3) + B + \text{runcnt})\%\ (K/2)]$;

$\text{SN}_3 = \text{sound byte}_j\ [20 + (R * i + (G \ll 4) + B + \text{runcnt})\%\ (K/2)]$;

$\text{SN}_4 = \text{sound byte}_j\ [5 + (R * i + (G \ll 1) + B + \text{runcnt})\%\ (K/2)]$;

$\text{SN}_5 = \text{sound byte}_j\ [25 + (R * i + (G \ll 5) + B + \text{runcnt})\%\ (K/2)]$;

$j = j + 1$; move to next K sound bytes.

If we get 100,000 random bits, then we let runcnt = runcnt + 1.

(d) We get R, and G, B from (x, y), and let R_1, G_1, B_1, R_2, G_2, and B_2 be equal to zero firstly. If $(R - R_1)^2 + (G - G_1)^2 + (B - B_1)^2 < vt$, then find another (x, y) via the following equations:

$$x = (x + (R \oplus G) + 1)\ \%\ W,$$
$$y = (y + (G \oplus B) + 1)\ \%\ H, \tag{4}$$

and let $watchdog = watchdog + 1$.

If $watchdog > th$, then we discard this frame, move to the next frame, and run step (d) again.

(e) Get one random bit: $bit\ [i] = 1\&(R \oplus G \oplus B \oplus R_1 \oplus G_1 \oplus B_1 \oplus R_2 \oplus G_2 \oplus B_2 \oplus \text{SN}_1 \oplus \text{SN}_2 \oplus \cdots \oplus \text{SN}_n)$. And let $R_1 = R$, $G_1 = G$, $B_1 = B$.

New coordinate will be

$$x = (((R \oplus x) \ll 4) \oplus (G \oplus y))\ \%\ W,$$
$$y = (((G \oplus x) \ll 4) \oplus (B \oplus y))\ \%\ H. \tag{5}$$

Both x and y having the "≪4" operation are very important. If we omit this operation, the pass rate of statistical test will be down to 10% ~50% rapidly.

(f) Then back to step (c) to get another random bit $bit\ [i]$ until we get a random byte.

(g) Let $R_2 = R$, $G_2 = G$, and $B_2 = B$; back to step (c) to get another random byte.

Figure 1 presents the AVRNG flowchart.

The next step is to prove that the random numbers generated from AVRNG are qualified enough. Obviously, we should apply NIST SP 800-22 Rev.1a—15 statistical tests [3] which are proposed in April 2010 to verify the randomness of proposed AVRNG.

3.2. Result of AVRNG with Filter. First of all, we briefly describe the materials that are used for experiments. The XiangSheng is a kind of comic dialogue to entertain the audience with ridiculous stories we mentioned in Section 1. And the cartoon, *Spirited Away*, is a Japanese animation produced in 2001. It is definitely difficult to produce qualified random numbers from a cartoon image because the original colors have already been artificial. The pictures and result are shown in Figures 2, 3, and 4 and Tables 2 and 3.

The test result in XiangSheng part, minimum value of XiangSheng3, is 0.98 which is the best result of all XiangSheng test samples. We can see that, when adopting AVRNG with filter, both XiangSheng and cartoon become good enough compared with WEBCAM with microphone. From the XiangSheng0 to XiangSheng5 means XiangSheng film XOR from 0 to 5 sound points, respectively. Similarly, from the Cartoon0 to Cartoon5 means cartoon film XOR from 0 to 5 sound points, respectively. The complete results are shown in Tables 2 and 3.

In cartoon part, minimum value of Cartoon3 is 0.98 which is the best result of all cartoon test samples. The result shows that we can generate qualified random numbers from AVRNG with filter no matter from WEBCAM, XiangSheng, or cartoon. The experimental equipment is Logitech Clear Chat Stereo headset's microphone (100–10,000 Hz, Sensitivity −62 dBV/uBAR, −42 dBV/Pascal ±3 dB) [26], and the video devices are 1.3 million pixel WEBCAM of Logitech QuickCam Pro 4000 [27] and BlueEyes's IPCAM BE-1200 [28]. The computer specification of experience is Asus U45J: CPU Intel i5-460 M (2.53 Hz) and 4 G RAM, and the OS is Fedora-18 x64.

4. Conclusions

In this work, the proposed AVRNG with filter could generate random numbers from WEBCAM with microphone (as TRNG) or from video file's frame with sound (as PRNG). Furthermore, the AVRNG adopts the filter 98% random

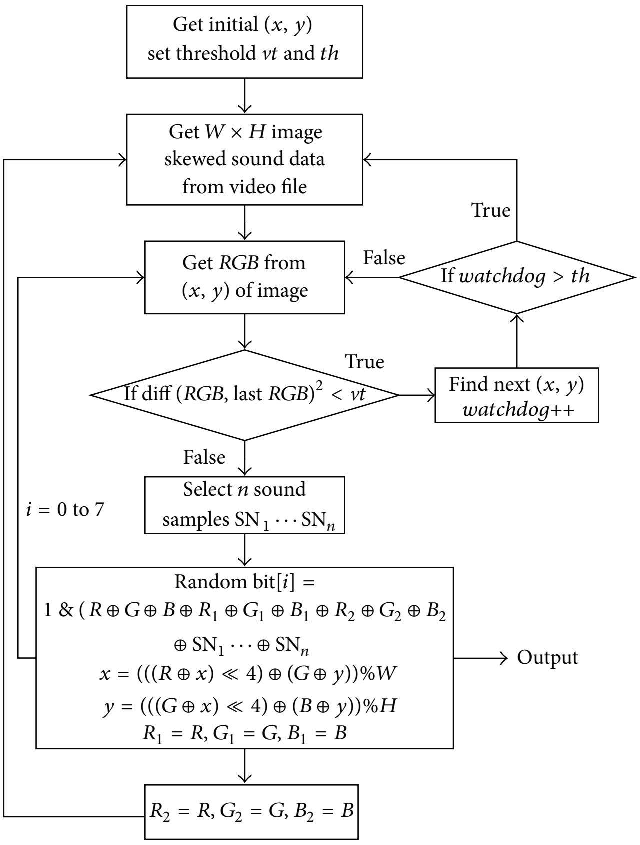

Figure 1: AVRNG with filter flowchart.

Table 2: AVRNG with filter (XiangSheng).

AVRNG with filter	XiangSheng0	XiangSheng1	XiangSheng2	XiangSheng3	XiangSheng4	XiangSheng5
Frequency	0.81	1	0.98	0.98	0.98	0.98
Block frequency	0.98	0.98	0.99	1	1	0.98
Cumulative sums	0.82	0.995	0.98	0.98	0.985	0.98
Runs	0.39	1	1	0.99	0.97	1
Longest-runs-of-ones	0.88	1	0.99	1	0.97	1
Binary matrix rank	0.98	0.99	0.99	1	0.98	0.99
FFT (Fourier)	1	1	1	1	1	1
Nonoverlapping template	0.9801	0.989	0.9899	0.9896	0.9881	0.9893
Overlapping template	0.98	0.98	0.99	1	0.98	1
Universal statistical	1	1	1	1	1	1
Approximate entropy	0.99	1	0.96	1	0.99	0.99
The random excursions	1	0.9886	0.9688	1	0.9861	0.9772
The random excursions variant	1	1	0.9931	1	1	1
The serial	0.995	0.995	0.99	0.996	0.985	0.985
The linear complexity	0.98	1	0.98	0.99	0.98	0.98
Average of above	0.9190	0.9945	0.9868	0.9950	0.9863	0.9901
Minimum of above	0.39	0.98	0.96	0.98	0.97	0.9772

numbers generated from both XiangSheng, and cartoon film could pass 15 statistical tests. Moreover, AVRNG with or without filter takes almost the same time to generate 100,000 random bits. The result is shown in Table 4.

The proposed random numbers generating method requires merely WEBCAM and microphone instead of complex equipment such as electronic circuit or oscillator to generate true random numbers. The results are principally

Table 3: AVRNG with filter (Cartoon).

AVRNG with filter	Cartoon0	Cartoonl	Cartoon2	Cartoon3	Cartoon4	Cartoon5
Frequency	0.3	1	0.99	1	0.98	0.98
Block frequency	0.86	1	1	1	0.98	0.98
Cumulative sums	0.275	1	0.985	1	0.985	0.99
Runs	0.57	0.99	1	1	0.98	0.99
Longest-run-of-ones	0.42	0.99	1	0.98	1	0.97
Binary matrix rank	1	1	0.98	0.98	0.98	0.98
FFT (Fourier)	1	1	1	1	1	1
Nonoverlapping template	0.9666	0.9905	0.9893	0.9901	0.9886	0.989
Overlapping template	0.59	1	0.99	1	0.99	0.98
Universal statistical	0	1	1	1	1	1
Approximate entropy	0.68	0.99	0.98	0.98	0.98	0.99
The random excursions	0	0.975	0.9722	1	1	0.9922
The random excursions variant	0	1	1	1	1	0.9969
The serial	0.915	0.995	0.995	0.995	0.995	0.995
The linear complexity	0.94	0.99	1	1	1	1
Average of above	0.5678	0.9947	0.9921	0.9950	0.9906	0.9889
Minimum of above	0	0.975	0.9722	0.98	0.98	0.97

Figure 2: WEBCAM image.

Figure 3: XiangSheng image.

Figure 4: Cartoon image.

Table 4: Efficiency of AVRNG with/without filter (unit: second).

AVRNG	Without filter	With filter	With/Without filter
XiangSheng	46.0636	47.1470	1.023
Cartoon	52.5758	52.5658	0.999

based on personal computer; consequently, to transfer proposed algorithm upon personal devices such as tablet PC or smartphone so as to prove that these efforts can be widely applied will be the next stage. The random numbers generated from this work could be used in evolutionary algorithm [29, 30]. Hopefully, this work will evolve into an effective adjunctive decision making tool.

Acknowledgments

This work was supported in part by the National Science Council under the Grants NSC 99-2221-E-037 -004 and the NSYSU-KMU under the Grants NSYSUKMU102-P001.

References

[1] D. R. Stinson, *Cryptography: Theory and Practice*, Chapman & Hall/CRC, New York, NY, USA, 3rd edition, 2005.

[2] J. M. Tsai, J. Tzeng, and I. T. Chen, "Random number generated from white noise of video," *ICIC Express Letter*, vol. 6, no. 7, pp. 1827–1832, 2012.

[3] NIST, "A statistical test suite for the validation of random number generators and pseudo random number generators for cryptographic applications," FIPS Special Publication 800-22 Rev.1a, 2010.

[4] J. B. Plumstead, "Inferring a sequence generated by a linear congruence," in *Proceedings of the 23th IEEE Symposium on the Foundations of Computer Science*, pp. 153–159, IEEE, New York, NY, USA, 1982.

[5] J. A. Reeds, "Cracking random number generator," *Cryptologia*, vol. 1, no. 1, pp. 20–26, 1997.

[6] J. Boyar, "Inferring sequences produced by pseudo-random number generators," *Journal of the Association for Computing Machinery*, vol. 36, no. 1, pp. 129–141, 1989.

[7] P. Alfke, "Efficient shift registers, LFSR, counters, and long pseudo-random sequence generators," XAPP 052, (Version 1.1), 1996, http://www.xilinx.com/support/documentation/application_notes/xapp052.pdf.

[8] H. Debiao, C. Jianhua, and H. Jin, "A random number generator based on isogenies operations," 2010, http://eprint.iacr.org/2010/094.

[9] X. Y. Wang and Q. Yu, "A block encryption algorithm based on dynamic sequences of multiple chaotic systems," *Communications in Nonlinear Science and Numerical Simulation*, vol. 14, no. 2, pp. 574–581, 2009.

[10] X. Y. Wang, X. Qin, and Y. X. Xie, "Pseudo-random sequences generated by a class of one-dimensional smooth map," *Chinese Physics Letters*, vol. 28, no. 8, Article ID 080501, 2011.

[11] X. Y. Wang and X. Qin, "A new pseudo-random number generator based on CML and chaotic iteration," *Nonlinear Dynamics*, vol. 70, no. 2, pp. 1589–1592, 2012.

[12] X. Y. Wang and Y. X. Xie, "A design of pseudo-random bit generator based on single chaotic system," *International Journal of Modern Physics C*, vol. 23, no. 3, Article ID 1250024, 11 pages, 2012.

[13] X. Y. Wang and L. Yang, "Design of pseudo-random bit generator based on chaotic maps," *International Journal of Modern Physics B*, vol. 26, no. 32, Article ID 12502080, 9 pages, 2012.

[14] G. Taylor and G. Cox, "Behind intel's new random number generator," Semiconductors Processors, 2011, http://spectrum.ieee.org/computing/hardware/behind-intels-new-randomnumber-generator/0.

[15] L. C. Noll, S. Cooper, and M. Pleasant, LavaRnd, 1996, http://www.lavarnd.org.

[16] S. K. Tawfeeq, "A random number generator based on single-photon avalanche photodiode dark counts," *Journal of Lightwave Technology*, vol. 27, no. 24, pp. 5665–5667, 2009.

[17] Y. Yamanashi and N. Yoshikawa, "Superconductive random number generator using thermal noises in SFQ circuits," *IEEE Transactions on Applied Superconductivity*, vol. 19, no. 3, pp. 630–633, 2009.

[18] X. Y. Wang, X. Qin, and L. Teng, "A novel true random number generator based on mouse movement and a one-dimensional chaotic map," *Mathematical Problems in Engineering*, vol. 2012, Article ID 931802, 9 pages, 2012.

[19] Y. A. Alsultanny, "Random-bit sequence generation from image data," *Image and Vision Computing*, vol. 26, no. 4, pp. 592–601, 2008.

[20] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, New York, NY, USA, 1996.

[21] J. Tzeng, I. T. Chen, and J. M. Tsai, "Random number generator designed by the divergence of scaling functions," in *Proceedings of the 5th International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP' 09)*, pp. 1038–1041, Kyoto, Japan, September 2009.

[22] NIST, FIPS PUB 140-2, "Derived test requirements for FIPS PUB 140-2, security requirements for cryptographic modules," Federal Information Processing Standards Publication, 2004.

[23] NIST, Revised draft FIPS 140-3, 2009, http://csrc.nist.gov/publications/drafts/fips140-3/revised-draft-fips140-3_PDF-zip_document-annexA-to-annexG.zip.

[24] NIST, Revised draft FIPS 800-90a, 2012, http://csrc.nist.gov/publications/nistpubs/800-90A/SP800-90A.pdf.

[25] G. Marsaglia, "DIEHARD: a battery of tests of randomness," The preceding description of the DIEHARD executable program that explains the significance of the results, 1995, http://stat.fsu.edu/pub/diehard.

[26] "Logitech ClearChat Stereo Headset," http://www.logitech.com/en-gb/speakers-audio/headphones/devices/349.

[27] Logitech QuickCam Pro 4000, http://www.logitech.com/en-gb/support/269?crid=405.

[28] BlueEyes's IPCAM BE-1200, http://www.blueeyes.com.tw/EN/brochure/BE1200.pdf.

[29] W. H. Ho, J. H. Chou, and C. Y. Guo, "Parameter identification of chaotic systems using improved differential evolution algorithm," *Nonlinear Dynamics*, vol. 61, no. 1-2, pp. 29–41, 2010.

[30] W. H. Ho and C. S. Chang, "Genetic-algorithm-based artificial neural network modeling for platelet transfusion requirements on acute myeloblastic leukemia patients," *Expert Systems with Applications*, vol. 38, no. 5, pp. 6319–6323, 2011.