True Random Bit Generation from a Double Scroll Attractor

Müştak E. Yalçın *(Student Member, IEEE)*,

Johan A.K. Suykens *(Member, IEEE)*

and Joos Vandewalle *(Fellow, IEEE)*

Katholieke Universiteit Leuven

Department of Electrical Engineering, ESAT-SCD-SISTA

Kasteelpark Arenberg 10, B-3001, Heverlee (Leuven), Belgium

Tel: +32-16-32 96 06, Fax: +32-16-32 19 70

E-mail: {mey,johan.suykens,joos.vandewalle}@esat.kuleuven.ac.be

Abstract

In this paper, a novel true random bit generator (TRBG) based on a double scroll attractor is proposed. The double scroll attractor is obtained from a simple model which is qualitatively similar to Chua's circuit. In order to face the challenge of using the proposed TRBG in cryptography, the proposed TRBG is subjected to statistical tests which are the well-known FIPS-140-1 and Diehard test suite in the area of cryptography. The proposed TRBG successfully passes all these tests and can be implemented in integrated circuits.

1 Introduction

Random number generators are used in many areas including computer simulations, Monte-Carlo techniques in numerical analysis, test problem generation for the performance evaluation of computer algorithms, statistical sampling, stochastic optimization methods (such as simulated annealing), watermarking for image authentication and cryptography. A good random number generation helps to improve the results in these applications and is vital for cryptographic security. Security issues are coming to the fore today because of increasingly demanding security requirements in many new applications on the internet (such as secure e-mail, e-commerce, e-government). The security of cryptographic algorithms depends on generating secret quantities which are generated by random number generators, for passwords, keys, etc. As noted in [12], the use of pseudorandom processes to generate secret quantities can result in pseudo-security. Random numbers can be generated by a random bit generator which can be defined as a device or algorithm whose output is a sequence of statistically independent and unbiased binary digits [31]. To ensure that a random number generator is secure, its output must be statistically indistinguishable from a true random sequence and unpredictable [31, 12].

Random number generators (RNGs) can be classified into three classes which are *true random number generators (TRNGs)*, *pseudorandom number generators (PRNGs)* and *hybrid random number generators (HRNGs)*. A true random number generator operates by measuring unpredictable natural processes such as thermal noise from a semiconductor [9], frequency instability of an oscillator [16], elapsed time between emission of particles during radioactive decay [20] and variations in disk drive response times [10, 22]. These are also called hardware-based generators because of the use of the randomness aspect in the hardware. The processes are chaotic or non-deterministic. Furthermore, they produce continuous time analog signals which are often called noise. As is shown in [1, 29], true random bits/numbers can be generated by these physical noise sources.

Pseudorandom number generators use deterministic processes (also called *deterministic random number generators*) to generate a series of outputs from an initial seed state. PRNGs are much more cost-effective and thousands of times faster than hardware based random number generators. However, because the output is a function of the seed state, the actual entropy of the output can never exceed the entropy of the seed [24, 35]. Hence, the randomness level of the pseudorandom numbers depends on the level of randomness of the seed. Therefore, hybrid random number generators use a random generator as a seed generator and expand it. A seed generator is a hardware-based RNG with or without user's interaction such as random keystrokes, mouse movements or hard drive seek times.

If one summarizes the present state-of-the-art in random number generators, the conclusion is that pseudorandom number generators need a true random number generator in order to improve the quality of the output (e.g. the hardware-based Intel Random Number Generator [9] included in the Intel 8XX series of chip-sets for generating the random seed [36]). Although commercial cryptographic packages have user's interaction to produce a supposedly random seed and do not need additional hardware, these methods are time consuming, inconvenient for the user and cannot be used with automated scripts [22, 24]. Nevertheless, it is quite difficult to distinguish between a signal coming from a non-deterministic source and a chaotic system (Ref. [38] is a recent study which addresses this issue). In this paper, we will group true random number generators into two classes, based on the kind of analog signal they are using and will be called a true random number generator either based on a chaotic or on a non-deterministic system (e.g. while the sources of [9, 20] are known as a non-deterministic system, variations in disk drive response time are known to be a consequence of chaotic air turbulence [16]). The same classification of TRNGs is also made in [29].

The output of chaotic systems can be predicted from the exact initial conditions depending on the first Lyapunov exponent. However, sensitivity with respect to the initial conditions causes unpredictability on the long term. The short term predictability of chaotic time-series was shown in [11, 5] and has been also addressed in the time-series prediction competition reported in [41]. Nevertheless, short term predictability of chaotic time-series is seen as a drawback for using chaotic signals in random number generators which are designed for cryptographic applications. A scheme which is taking into account the *predictability time* is studied in [2]. In [18] a non-invertible function is used as a static output of a chaotic system which also improves the randomness for the output signal. In fact, non-invertibility (also called a *one-way function* in cryptography) is very much used in many areas in cryptography such as public key cryptography, authentication [31] and also for generating pseudorandom numbers [37, 31]. Similar ideas have been used by Stojanovski and Kocarev [39] who define a binary generating partition which can be considered as a one-way function, on the chaotic piecewise-linear one-dimensional map for producing random bits. In a similar way, chaotic maps have been used as a binary information source in telecommunications [3, 4].

One should note that when unpredictability is an important issue, direct use of the output of the chaotic system might give an advantage to the opponent if it is short term predictable. Cryptographic protocols crucially depend on the unpredictability of the secret key which is produced by a random number generator. If an attacker can predict the key's value or the number of keys, the protocol can be broken with much less effort than when truly random keys are used. Therefore, it is vital that the secret keys are generated from a true random source. Statistical tests determine whether the sequence possesses a certain attribute that a truly random sequence would be likely to exhibit. Hence any random number generator which is proposed for use in a cryptographic protocol, must be subject to statistical tests. There are two very well-known statistical test suites which are FIPS-140-1 [31] and the Diehard test suite [30]. FIPS-140-1 is issued by the National Institute of Standards and Technology (NIST) and strong statistical tests are given in the Diehard test suite.

The aim of this paper is to show the usage of continuous-time chaotic systems for generating true random bits. We introduce a straightforward method for generating true random bits and take care of randomness in the sense of unpredictability. Due to the difficulty of proving unpredictability in a theoretical way, the proposed true random bit generator is subjected to statistical tests. We demonstrate that the proposed true random bit generator passes the FIPS-140-1 statistical tests and the full suite of the Diehard test. However, one should notice that none of the statistical tests can prove that the generator is an ideal random number generator (the many tests are in fact necessary conditions but not sufficient).

This paper is organized as follows. In Section 2, the proposed true random bit generator is described in general terms. The following sections describe the details of the system block by block. In Section 3, the employed chaotic oscillator is described. Section 4 explains how a bit sequence is generated by means of the chaotic signal. In Section 5 a de-skewing technique is discussed for eliminating bias in the sequence. Furthermore, it is explained how one of the threshold values is set by measuring entropy. Empirical tests and results are presented in Section 6.

2 A Double Scroll based True Random Bit Generator

In this paper we introduce a new True Random Bit Generator (TRBG) based on a chaotic system. Figure 1 shows the proposed TRBG which consists of a hardware and software parts. The software part can also be integrated in the hardware part using standard digital circuits. For practical reasons, this block is implemented here in software. The hardware part includes a chaotic oscillator and thresholds circuit. The chaotic oscillator is a continuous time third order autonomous system and exhibits a *double scroll* attractor. Double scroll attractors have been observed in Chua's circuit [7]. Since then, the double scroll attractor has been addressed by many researchers [28, 6]. Here, the chaotic oscillator has been used as a source for the proposed RBG which is qualitatively similar to Chua's circuit [8]. The double scroll attractor block which is denoted by $\mathcal{S}_1$ is described in Section 3 together with its circuit implementation.

The state space of the double scroll attractor is partitioned into three subspaces by block $\mathcal{S}_2$ of the hardware part. Two of the subspaces are related with the location of the scrolls in state space. The third subspace is a region where the transition between the scrolls occurs. The bit generator (block $\mathcal{S}_3$) is the first block in the software part. Its inputs correspond to the outputs of block $\mathcal{S}_2$. The block $\mathcal{S}_3$ is basically coding the scrolls (e.g. the output of the bit generator is 0110 which means that the state space trajectory moved from the scroll 0 to the other scroll 1, jumps to itself and then returns to scroll 0). In fact, blocks $\mathcal{S}_2$ and $\mathcal{S}_3$ are discretizing the state variable $x(t)$ in space instead of in time (relying on the ergodicity property of chaos). When the blocks $\mathcal{S}_2$ and $\mathcal{S}_3$ are combined and considered as one block its function is non-invertible. Sampling the chaotic signal in space gives an irregular sampling of the signal in time (Figure 2). Therefore, it becomes very hard to make a prediction for an opponent in cryptography, both a one-step ahead and long term prediction. In this way, the method presented in this paper is a combination of the two ideas mentioned in the introduction and improves the randomness in the sense of unpredictability [18, 2].

The last block ($\mathcal{S}_4$) in the software part is a de-skewing which is eliminating the bias in the output of the bit generator. This bias depends on the probability density of the chaotic signal itself. Moreover, offset and bandwidth limitations might also effect the bias of the generated sequence [1].

3 Double Scroll Attractor

Recently, in [14] an alternative way for generating double scroll like behavior was proposed by means of a simple circuit implementation. It is given by

$$\mathcal{S}_1 : \quad \dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}f(\mathbf{C}\mathbf{x}) \tag{1}$$

with $\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -a & -a & -a \end{bmatrix}$, $\mathbf{B} = \begin{bmatrix} 0 \\ 0 \\ a \end{bmatrix}$ $\mathbf{C} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$ where

$$f(\zeta) = \begin{cases} 1 & \text{if } \zeta \geq 0 \\ -1 & \text{if } \zeta < 0. \end{cases}$$

and $\mathbf{x} = [x; y; z] \in \mathbb{R}^3$, $\zeta \in \mathbb{R}$. This model produces a double-scroll like attractor for $a = 0.8$ [14]. By taking a generalization of this nonlinearity, it is also possible to obtain n-scroll attractors, similar to the generalized Chua's circuit [40]. The generalized nonlinearity for (1) is given by

$$f(x) = \sum_{i=1}^{M} g_{\frac{(-2i+1)}{2}}(x) + \sum_{i=1}^{N} g_{\frac{(2i-1)}{2}}(x) \quad \text{with} \quad g_\theta(\zeta) = \begin{cases} 1 & \text{if } \zeta \geq \theta \quad \theta > 0 \\ 0 & \text{if } \zeta < \theta \quad \theta > 0 \\ 0 & \text{if } \zeta \geq \theta \quad \theta < 0 \\ -1 & \text{if } \zeta < \theta \quad \theta < 0. \end{cases}$$

An experimental confirmation of a 5-scroll attractor for $M = 2$ and $N = 2$ has been given in [45]. Furthermore, families of scroll grid attractors have been obtained with the same model by taking additional nonlinearities with similar characteristics [45, 6]. In this paper, we use $M = 1$ and $N = 0$ in order to obtain a double scroll attractor. The reason for choosing this model in our application is that the model itself is quite simple. As a result, the implementation can be fully integrated in a CMOS chip with small cost. One of the possible integrations in CMOS chips has been proposed in [15]. Furthermore, two new circuit realizations of this double scroll attractor have been recently proposed based on MOS-only [34] and log domain PNP-only transistors [33]. The latter are very suitable for low-voltage and high frequency operation and can also be implementated in integrated circuits. Hence, a random number generator based on this chaotic system could be included also as a standard part of a system's architecture, as suggested in [12].

Numerous research results have illustrated the advantages of current feedback op-amps (CFOA) for the implementation of chaotic oscillators such as operating at higher frequencies with design flexibility [13]. Therefore a CFOA based circuit realizing the double scroll attractor has been designed for the purpose of this paper. The circuit diagram is given in Figure 3. For $C_x = C_y = C_z = C$, $R_1 = R$, $R_2 = R_4 = R/a$, $V_x = ax$, $V_y = ay$, $V_z = z$ and using the normalized quantity $\tau = \frac{t}{RC}$, this circuit realizes the system (1). The same circuit structure has been successfully used in [45] for realizing families of scroll grid attractors. The CFOAs implemented using AD844 from Analog Devices and required non-linearity is realized by LM311 type comparators. CFOAs and LM311 are supplied under $15V$ DC and passive component values are taken as $C = 1nF$ and $R_1 = 5.1k\Omega$; the values of other passive components have been assigned during the experiment. On the circuit, θ is measured as -0.51 Volt which has been obtained using series connected resistances between ground and -15 Volt. In order to obtain a symmetric double scroll, the value of R_x is set to $74.8k\Omega$, $R_2 = 7k\Omega$ and $R_4 = 11k\Omega$. The corresponding result for these value is shown in Figure 4.

4 Bit Generation

A simple way to generate a bit sequence from a chaotic real valued signal is by using a threshold function [43]

$$\sigma_c(x) = \begin{cases} 0 & \text{if } x < c \\ 1 & \text{if } x \geq c. \end{cases} \tag{2}$$

One obtains binary sequences which are referred to as chaotic binary sequences [43]. Because our chaotic system $\mathcal{S}_1$ generates a continuous time signal, we define two thresholds (c_1 and c_2). Then the state space is divided into three subspaces denoted by $V_0 = \{(x,y,z)|x \leq c_2\}$, $V_T = \{(x,y,z)|c_2 < x < c_1\}$ and $V_1 = \{(x,y,z)|x \geq c_1\}$. Figure 5 shows the subspaces for the values $c_1 = 0$ and $c_2 = -1$. The block $\mathcal{S}_2$ that realizes the two threshold functions, is described by

$$\mathcal{S}_2 : \quad \begin{aligned} \sigma^1(x(t)) &= \begin{cases} 0 & \text{if } x(t) < c_1 \\ 1 & \text{if } x(t) \leq c_1 \end{cases} \\ \sigma^0(x(t)) &= \begin{cases} 0 & \text{if } x(t) > c_2 \\ 1 & \text{if } x(t) \leq c_2. \end{cases} \end{aligned} \tag{3}$$

By discretizing the state space in this way, one can obtain symbolic dynamics that describe the evolution of the system by an infinite sequence of symbols [27, 19]. The two threshold functions are realized by LM311 type comparators (Figure 6) which are of the same type as the comparators to realize the nonlinearity for the double scroll attractor. The outputs of the comparators are directly connected to a Personal Computer via a parallel port.

A bit is generated when the signal passes the subspace V_0 or the subspace V_1 region through the subspace V_T. The bit generator block $\mathcal{S}_3$ is described by

$$\mathcal{S}_3 : \qquad \sigma_i(\sigma^0, \sigma^1) \;=\; \begin{cases} 0 & \text{if } \sigma^0 = 0, \sigma^1 : {}_0\uparrow^1 \\ 1 & \text{if } \sigma^1 = 0, \sigma^0 : {}_0\uparrow^1 \end{cases} \tag{4}$$

where $\sigma^0 : {}_0\uparrow^1$ shows a raising edge (transition from logic 0 to logic 1) of σ^0 and $i \in \{0, 1, 2, ...\}$. Figure 7 shows the corresponding output bits for an input of the block $\mathcal{S}_3$. This bit generation method basically aims at characterizing the jumps in signal $x(t)$ from one scroll to the other or staying at the same scroll.

The threshold value c_1 is set to 0 Volt; c_2 is not determined at this point but is considered as a design parameter to further improve the statistical properties of the bit sequence, as will be further explained in the next section. The signals $y(t)$ and $z(t)$ do not directly effect the bit generation. Therefore, when one compares the proposed RBG with a random number generator using 1D chaotic map, one recognizes that the proposed RBG brings additional difficulties to the opponent in cryptography.

5 De-skewing and Setting the Threshold c_2

A natural source of random bits may not give unbiased bits as direct output [23, 21]. Many applications, especially in cryptography, rely on sequences of unbiased bits. There are various techniques to extract unbiased bits from a defective generator with unknown bias. These are called *de-skewing techniques* [31]. These techniques also eliminate the correlation in the output of the natural sources of random bits. This problem is also thought as simulating a fair (unbiased) coin using a biased coin. Von Neumann [32] has probably been the first author to state this problem. He proposed a digital post-processing that balances the distribution of bits. Post-processing converts non-overlapping pairs of bits into output bits by converting the bit pair 0 1 into an output 0, converting 1 0 into an output 1 and pairs 1 1 and 0 0 are discarded [31]. In [10] the authors used the Fast Fourier Transform (FFT) for de-skewing in order to obtain uncorrelated and unbiased numbers from the disk's timing data. Also, cryptographic hash functions are used as another de-skewing technique by taking advantage of their statistical property [31]. In this paper Von Neumann's technique is used because it can easily be integrated into the hardware and it is not decreasing the bit rate too much compared with the other methods. The de-skewing block $\mathcal{S}_4$ which is used in this paper, is described by

$$\mathcal{S}_4: \qquad b_i(\sigma_i, \sigma_{i-1}) = \begin{cases} 0 \quad \text{if } \sigma_i = 0 \ \wedge \ \sigma_{i-1} = 1 \\ 1 \quad \text{if } \sigma_i = 1 \ \wedge \ \sigma_{i-1} = 0. \end{cases} \tag{5}$$

In order to obtain an unbiased bit sequence, the bit generator part is combined with a simple subroutine of the de-skewing block $\mathcal{S}_4$. The proposed TRBG has a single output which is denoted by $\mathcal{B} = \{..., b(i-1), b(i), b(i+1), ...\}$ and $b(i) \in \{0, 1\}$. Statistical properties of the TRBG can be adjusted by the threshold c_2 value which serves as a control parameter for the TRBG. Using the fact in information theory that noise has maximum entropy, the threshold value c_2 is chosen such that the measured entropy of the TRBG is maximal.

The measure-theoretic entropy [17] of the proposed true random bit generator with respect to a partition c_2 is given by

$$h^{c_2} = \lim_{n \to \infty} \frac{H_n^{c_2}}{n}$$

where

$$H_n^{c_2} = -\sum_{\mathcal{B}^n} P(\mathcal{B}^n) \ln P(\mathcal{B}^n) \tag{6}$$

with $P(\mathcal{B}^n)$ the probability of occurrence for binary subsequence (words $\{..., \{b(i), b(i+1), ..., b(i+n-1)\}, \{b(i+n), ..., b(i+2n-1)\}, ...\}$) of length n [27, 17, 39]. $H_n^{c_2}$ achieves its maximum when the n-word sequences in the output sequence $\mathcal{B}$ of the TRBG are equally distributed.

To find the maximum entropy with respect to the threshold c_2, $1,000,000$ bits were generated for 17 candidate thresholds values. Using (6), $\frac{H_n^{c_2}}{n}$ was calculated for $n = 3, 4, ..., 14$ (Figure 8) shows values for $n = 3, 4, ..., 10$. (Figure 9) shows how the proposed TRBG can come close the maximum entropy ($\ln 2$) which might be possible for a true random bit generator. Threshold c_2 is therefore set to -1.44 Volt.

6 Statistical Tests

A large number of statistical tests and whole test suites have been proposed to assess the statistical properties of random number generators. Truly random sequences are e.g. likely to exhibit the same number of 0's and 1's in the sequence. A finite set of statistical tests may only detect defects in statistical properties. Furthermore, the tests cannot verify the unpredictability of the random sequence which is demanded in cryptographic applications. In this paper unpredictability of the sequence is a consequence of having a chaotic signal as a source.

In this section, the proposed random bit generator is faced with statistical tests. The statistical tests are applied to a bit sequence obtained after the de-skewing. As introduced in the previous section, de-skewing maintains bias and correlation and it also maintains the other statistical properties. A better result of the statistical test can be obtained, if a strong de-skewing technique is used such as a cryptographic hash function [31], but these are more time-consuming.

Before we present the test results, the tests that are applied in the next subsection are introduced (details of the tests can be found in [31, 25]):

Monobit test : The purpose of this test is to determine whether the number of 0's and 1's in the bitstream are approximately the same.

Poker test : The poker test determines whether the sequences of length m appear approximately the same number of times in the bitstream. The bitstream is divided into k ($k = \lfloor \frac{n}{m} \rfloor$) non-overlapping parts each of length m. Let n_i be the number of occurrences of the i^{th} type of sequence of length $m = 1, 2, ..., 2^m$. The statistic is then

$$X_3 = \frac{2^m}{k}\left(\sum_{i=1}^{2^m} {n_i}^2\right) - k \tag{7}$$

which approximately follows a χ^2 distribution [25, 31] with $2^m - 1$ degrees of freedom.

Runs test : A run of bitstream is a subsequence of it consisting of consecutive 0's or consecutive 1's. A run of 0's is called a gap, while a run of 1's is called a block. The runs test determines whether the number of runs of various lengths in the bitstream is as expected for a random sequence. The statistic used is

$$X_4 = \sum_{i=1}^{k} \frac{(B_i - e_i)^2}{e_i} + \sum_{i=1}^{k} \frac{(G_i - e_i)^2}{e_i} \tag{8}$$

with $e_i = \frac{(n-i+3)}{2^{i+2}}$ (the expected number of gaps and blocks of length i) where n is the length of the random sequence and B_i, G_i is the number of blocks and gaps, respectively, of length i in the bitstream for each $i = 1, 2, ..., k$. X_4 approximately follows a χ^2 distribution [25, 31] with $2^k - k$ degrees of freedom. It is noted by Knuth [25] that the runs test is a strong test and is used in [44].

6.1 FIPS $140 - 1$ Statistical Tests for Randomness

Federal Information Processing Standards (FIPS) are issued by the National Institute of Standards and Technology (NIST). FIPS $140 - 1$ covers security requirements for cryptographic modules and specifies recommended statistical tests for random number generators. Here a single bitstream of length $20,000$ bits as output from the random number generator is subject to each of the tests. If one of the tests fails, the generator fails the test [31].

- *Monobit test :* The number of 1's (N_1)in the bitstream should satisfy $9654 < N_1 < 10346$.

- *Poker test :* The statistic X_3 is computed for $m = 4$. The poker test is passed if $1.03 < X_3 < 57.4$.

- *Runs test :* The number of blocks B_i and gaps G_i of length i in the bitstream are counted for each $i = 1, ..., 66$. The runs of length greater than 6 are considered to be of length 6. The runs test is passed if the 12 counts B_i and G_i $(i = 1, ..., 6)$ are each within the corresponding interval given in Table 1.

- *long run test :* The long run test is passed if there are no runs of length 34 or more.

The proposed true random bit generator successfully passes all four statistical tests for every run. Table 2 provides results of five consecutive runs for the FIPS $140 - 1$ specified tests.

6.2 Diehard Test Suite

For further testing, we use the software package called Diehard [30] that has been designed for empirical testing. Diehard includes a battery of about 200 instances of fifteen traditional statistical tests which are birthday spacings, overlapping permutations, ranks of 31×31 and 32×32 matrices, ranks of 6×8 matrices, monkey tests on 20-bit Words, monkey tests OPSO, OQSO, DNA, count the 1's in a stream of bytes, count the 1's in specific bytes, parking lot, minimum distance, random spheres, squeeze, overlapping sums, runs, and craps.

The Diehard test suite has been applied here to 80 million bits (`http://www.esat.kuleuven.ac.be/~mey/Ds2RbG/ds2rbg.bin`) generated by the system proposed in this paper. It passes the full suite of Diehard. The complete test results can be further verified at `http://www.esat.kuleuven.ac.be/~mey/Ds2RbG/Ds2RbG.html`

7 Conclusions

This paper addresses a most encouraging application of chaotic signals in engineering which is its use as a source for random number generators. Although RNGs have been presented as a possible application of chaotic signal for several years, up till now the use of a continuous time signal from a chaotic circuit has never been shown with full empirical tests results. In this paper, we presented a random bit generator which uses a double scroll attractor from a simple circuit model. The proposed TRBG is subjected to statistical tests using the well-known tests suites FIPS-140-1 and Diehard in cryptography. The proposed TRBG successfully passes all four statistical tests of the FIPS-140-1 test suite for all runs. A strong and more complete test suite Diehard has been applied to 80 million bits. The system successfully passes the full suite of Diehard test. Another essential result of this paper is that the proposed TRBG implementation can be fully integrated. Therefore, it may become a standard part in hardware.

Interesting further research directions in this field may include:

- One important question might be the bit rate of the proposed RBG which has not been discussed in this paper. Here, we wanted to show that a simple chaotic oscillator can be used for a random bit generator. The bit rate of the proposed RBG depends on the normalized quantity τ of the designed chaotic oscillator. This means that the chaotic oscillator must be designed for a high frequency range when a high bit rate is required. Therefore, this work shows again the importance of high frequency designs of chaotic oscillators for applications like this one.

- Statistical tests have shown that the proposed RBG has good statistical properties. However, unpredictability of the system has not been tested yet. More complex attractors such as n-scrolls or scroll grid attractors may further improve unpredictability at this point.

- One possible cryptanalytic opponent technique might be synchronization of the chaotic system [42]. The aim is basically then to make a one step ahead prediction of the bits sequence by using an identical chaotic system that is driven by a binary signal coming from the proposed RBG. However, even given a full trajectory of the chaotic signal for reaching synchronization this is a challenge in chaotic communications [26]. By using only binary information where a state variable of the trajectory is located as a driver, it is not easy to synchronize two chaotic systems and eventually to predict the next bit.

Acknowledgments

This research work was carried out at the ESAT laboratory and the Interdisciplinary Center of Neural Networks ICNN of the Katholieke Universiteit Leuven, in the framework of the Belgian Programme on Interuniversity Poles of Attraction, initiated by the Belgian State, Prime Minister's Office for Science, Technology and Culture (IUAP P4-02, IUAP P4-24, IUAP-V), the Concerted Action Project MEFISTO of the Flemish Community and the FWO project *Collective Behavior and Optimization: an Interdisciplinary Approach* and ESPRIT IV 27077 (DICTAM).

References

[1] V. Bagini and M. Bucci. A design of reliable true random number generator for cryptographic applications. In Ç. K. Koç and C. Paar, editors, *Cryptographic Hardware and Embedded Systems, First International Workshop*, volume LCNS 1717 of *Lecture Notes in Computer Science*, pages 204–218. Springer-Verlag, 1999.

[2] G. Boffetta, M. Cencini, M. Falcioni, and A. Vulpiani. Predictability : a way to characterize complexity. *Physics Reports*, 356:367–474, 2002.

[3] S. Callegari, A. Cesaroni and G. Setti. Mixed mode unpredictable binary information source exploiting complex dynamics and adaptive median thresholding. In *Proceedings of The 9th Workshop on Nonlinear Dynamics of Electronic Systems (NDES 2001)*, pages 65–68, 2001.

[4] S. Callegari, R. Rovatti and G. Setti. Spectral properties of chaos-based FM signals: theory and simulation results. *IEEE Trans. Circuits and Systems-I*, 50(1):3–15, 2003.

[5] M. Casdagli. Nonlinear prediction of chaotic time series. *Physica D*, 35:335–356, 1989.

[6] G. Chen and T. Ueta. *Chaos in Circuits and Systems.* World Scientific Pub. Co. Singapore, 2002.

[7] L. O. Chua, M. Komuro, and T. Matsumoto. The double scroll family. *IEEE Trans. Circuits and Systems-I*, 33:1072–1118, 1986.

[8] L. O. Chua, W. Wu, A. Huang, and G. Zhong. A universal circuit for studying and genereting chaos-part II: Strange attractors. *IEEE Trans. Circuits and Systems-I*, 40:745–761, 1993.

[9] Intel Corporation. The intel random number generator. Technical report, Intel Corporation, 1999. Available for free download : `http://www.intel.com/design/security/rng/techbrief.htm`.

[10] D. Davis, R. Ihaka, and P. Fenstermacher. Cryptographic randomness from air turbulence in disk drives. In Y. G. Desmedt, editor, *Advances in Cryptology, 14th Annual International Cryptology Conference- CRYPTO '94*, volume LNCS 0839 of *Lecture Notes in Computer Science*, pages 114–120. Springer-Verlag, 1994.

[11] J. Doyne and J. J. Sidorowich. Predicting chaotic time series. *Physical Review Letters*, 59(8):845–848, 1987.

[12] D. Eastlake, S. Crocker, and J. Schiller. RFC 1750: randomness recommendations for security. Technical report, Network Working Group, December 1994.

[13] A. S. Elwakil and M. P. Kennedy. High frequency Wien-type chaotic oscillator. *Electronics Letters*, 34(12):1161–1162, 1998.

[14] A. S. Elwakil and M. P. Kennedy. Construction of classes of circuit-independent chaotic oscillators using passive-only nonlinear devices. *IEEE Trans. Circuits and Systems-I*, 48(3):289–307, 2001.

[15] A. S. Elwakil, K. N. Salama, and M. P. Kennedy. An equation for generating chaos and its monolithic implementation. *Int. J. Bifurcation and Chaos*, 12(12):2885–2896, 2002.

[16] R. C. Fairfield, R. L. Mortenson, and K. B. Coulthart. An LSI random number generator (RNG). In G.R. Blakley and David Chaum, editors, *Advances in Cryptology,Proceedings of CRYPTO 84*, volume LNCS 0196 of *Lecture Notes in Computer Science*, pages 203–230. Springer-Verlag, 1987.

[17] A. M. Fraser. Information and entropy in strange attractors. *IEEE Transactions on Information Theory*, 35(2):245–262, 1989.

[18] J. A. González, L. I. Reyes, J. J. Suarez, L. E. Guerrero, and G. Gutiérrez. A mechanism for randomness. *Physics Letters A*, 295(1):25–34, 2002.

[19] J. Guckenheimer. Toolkit for nonlinear dynamics. *IEEE Trans. Circuits and Systems-I*, 30(8):586–590, 1983.

[20] M. Guide. Concept for a high-performance random number generator based on physical random phenomena. *Frequenz*, 39:187–190, 1985.

[21] T. S. Han and M. Hoshi. Interval algorithm for random number generation. *IEEE Transactions on Information Theory*, 43(2):599–611, 1997.

[22] M. Jakobsson, E. Shriver, B. K. Hillyer, and A. Juels. A practical secure physical random bit generator. In *Proceedings of The Fifth ACM Conference on Computer and Communications Security*, 1998.

[23] A. Juels, M. Jakobsson, E. Shriver, and B. K. Hillyer. How to turn loaded dice into fair coins. *IEEE Transactions on Information Theory*, 46(3):911–921, 2000.

[24] B. Jun and P. Kocher. The intel random number generator. Technical report, White Paper Prepared for Intel Corporation, Cryptography Research Inc., April 1999. Available for free download : `http://www.intel.com/design/security/rng/CRIwp.htm`.

[25] D. E. Knuth. *The Art of Computer Programming*. Addison- Wesley, 1969.

[26] G. Kolumban, M. P. Kennedy, L. O. Chua. The role of synchronization in digital communications using chaos-Part II: Chaotic Modulation of Digital Communications *IEEE Trans. Circuits and Systems-I*, 45(11):1129–1140, 1998.

[27] D. Lind and B. Marcus. *An Introduction to Symbolic Dynamics and Coding*. Cambridge, 1995.

[28] R. N. Madan. *Chua's Circuit: A paradigm for chaos*. World Scientific, Singapore, 1993.

[29] D. P. Maher and R. J. Rance. Random number generators founded on signal and information theory. In Ç. K. Koç and C. Paar, editors, *Cryptographic Hardware and Embedded Systems, First International Workshop*, volume 1717 of *Lecture Notes in Computer Science*, pages 219–230. Springer, 1999.

[30] G. Marsalgia. *Diehard*. `http://stat.fsu.edu/~geo/diehard.html`.

[31] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, 1997. Available for free download : `http://www.cacr.math.uwaterloo.ca/hac/`.

[32] J. Von Neumann. Various techniques used in connection with random digits. *Applied Math Series., Notes by G.E. Forsythe, In National Bureau of Standards*, 12:36–38, 1951.

[33] S. Özoğuz and N. S. Şengör. On the realization of NPN-only log-domain chaotic oscillators. *IEEE Trans. Circuits and Systems-I*, 50(2):291–294, 2003.

[34] A. G. Radwan, A. M. Soliman, and A. El-Sedeek. MOS realization of the double-scroll-like chaotic equation. *IEEE Trans. Circuits and Systems-I*, 50(2):285–288, 2003.

[35] RSA Data Security. Hardware-based random number generation. Technical report, An RSA Data Security White Paper, 1999. Available for free download : `http://www.intel.com/design/security/rng/Intel_RNG_v2.htm`.

[36] RSA Data Security. Using RSA BSAFE crypto-c with intel random number generator. Technical report, An RSA Data Security White Paper, 1999. Available for free download : `http://www.intel.com/design/security/rng/RSA_BSAFE.htm`.

[37] A. Shamir. On the generation of cryptographically strong pseudorandom sequences. *ACM Transactions on Computer systems*, 1(1):38–44, 1983.

[38] M. Small, C. K. Tse. Detecting determinism in time series: The method of surrogate data. *IEEE Trans. Circuits and Systems-I*, 50(5):663–672, 2001.

[39] T. Stojanovski and L. Kocarev. Chaos-based random number generators-part I: analysis. *IEEE Trans. Circuits and Systems-I*, 48(3):281–288, 2001.

[40] J. A. K. Suykens, A. Huang, and L. O. Chua. A family of n-scroll attractors from a generalized Chua's circuit. *Archiv für Elektronik und Ubertragungstechnik*, 51(3):131–138, 1997.

[41] J. A. K. Suykens and J. Vandewalle. The K.U.Leuven time series prediction competition. In J. A. K. Suykens and J. Vandewalle, editors, *Nonlinear modeling : Advanced black-Box Techniques*, pages 241–253, 1998.

[42] J. A. K. Suykens, M. E. Yalçın, and J. Vandewalle. Chaotic systems synchronization. In G. Chen and X. Yu, editors, *Chaos Control : Theory and Applications.* Lecture Notes in Control and Information Sciences, Vol. 292, Springer-Verlag, 2003, pp. 117-136.

[43] Kohda T. and Tsuneda A. Statistics of chaotic binary sequences. *IEEE Transactions on Information Theory*, 43:104–112, 1997.

[44] B. Vizvari and G. Kolumban. Quality evaluation of random numbers generated by chaotic sampling phase-locked loops. *IEEE Trans. Circuits and Systems-I*, 45(3):216–224, 1998.

[45] M. E. Yalçın, J. A. K. Suykens, J. Vandewalle and S. Özoğuz. Families of scroll grid attractors. *Int. J. Bifurcation and Chaos*, 12(1):23–41, 2001.

Captions of Figures

Figure 1. Proposed true random bit generator.

Figure 2. Sampling the chaotic signal.

Figure 3. Circuit realizing the double scroll attractor.

Figure 4. Double scroll attractor.

Figure 5. Discretizing the state space for $c_1 = 0$ and $c_2 = -1$.

Figure 6. Additional circuit to generate binary sequences.

Figure 7. A bits sequence from the comparator outputs.

Figure 8. $\frac{H_n^{c2}}{n}$ as a function of the threshold c_2 for $n = 3, ..., 10$.

Figure 9. $\frac{H_n^{c2}}{n}$ for $n = 3, ..., 10$ where $-1.5 \leq c_2 \leq -1.3$.

Captions of Tables

Table 1. Runs test table.

Table 2. Results of FIPS$-140-1$ Specified Tests.

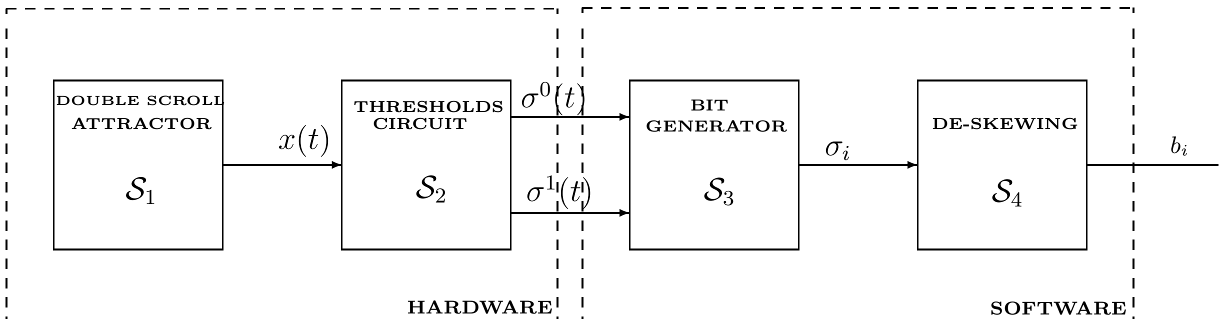

Figure 1: Proposed true random bit generator.

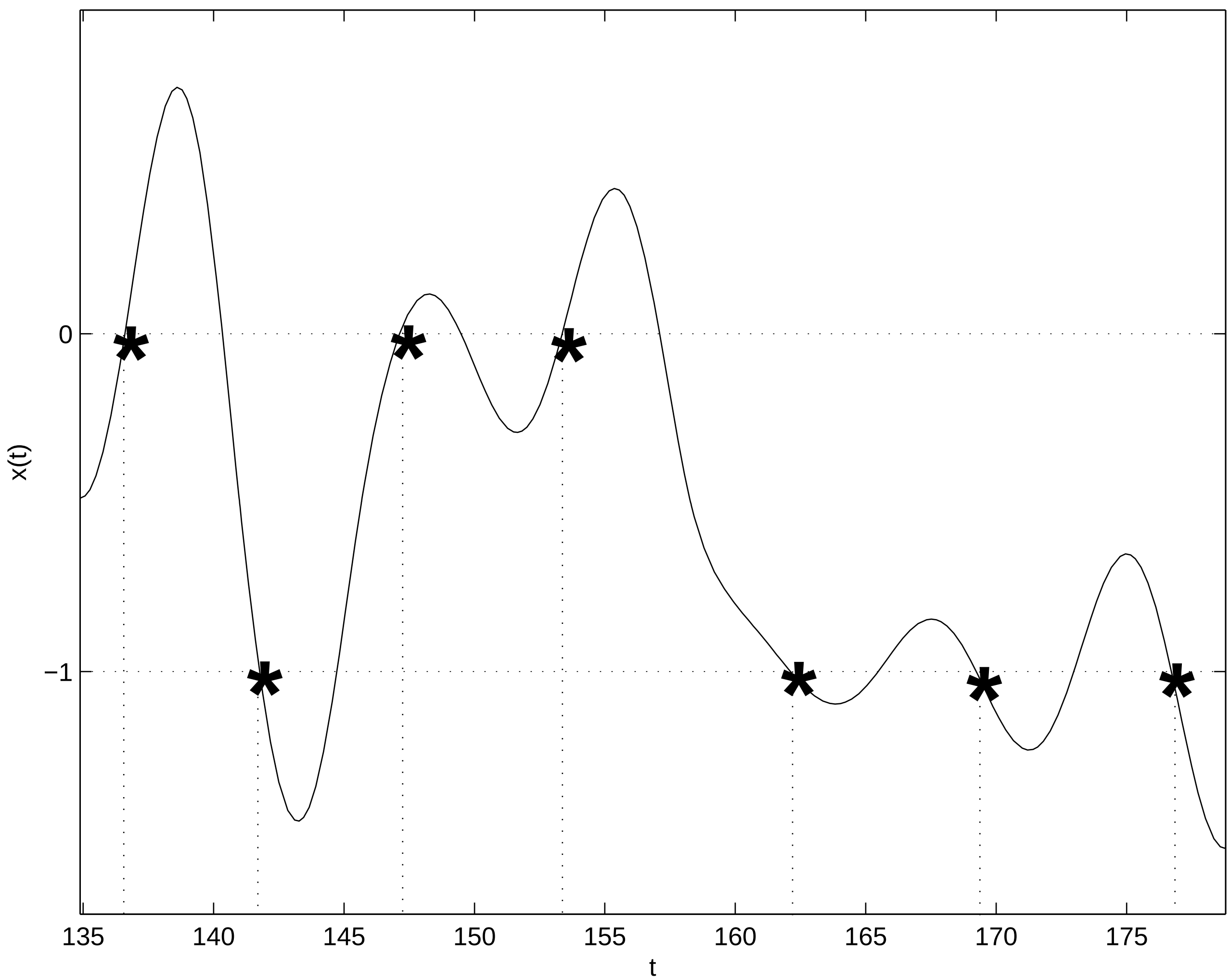

Figure 2: Sampling the chaotic signal.

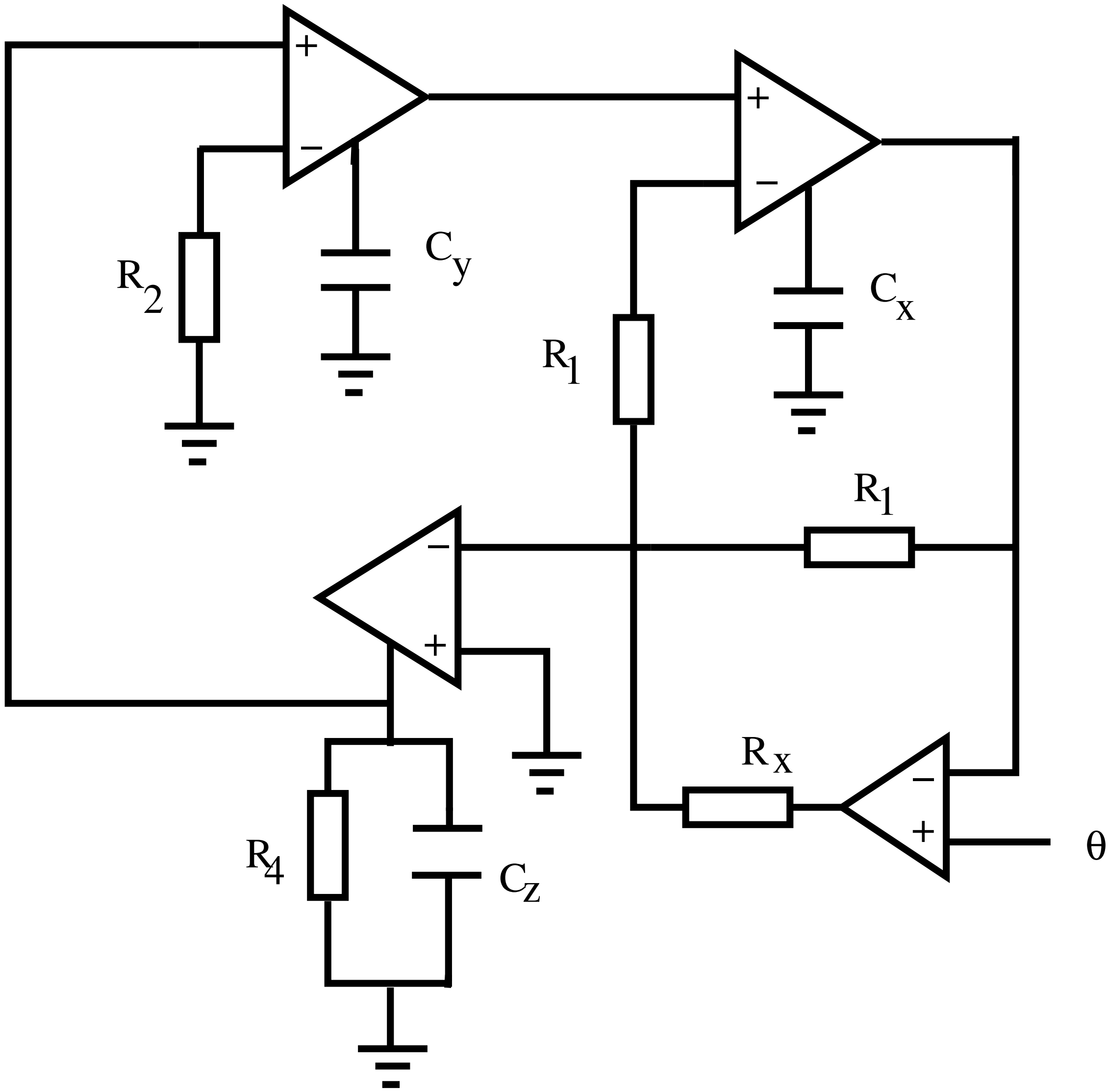

Figure 3: Circuit realizing the double scroll attractor.

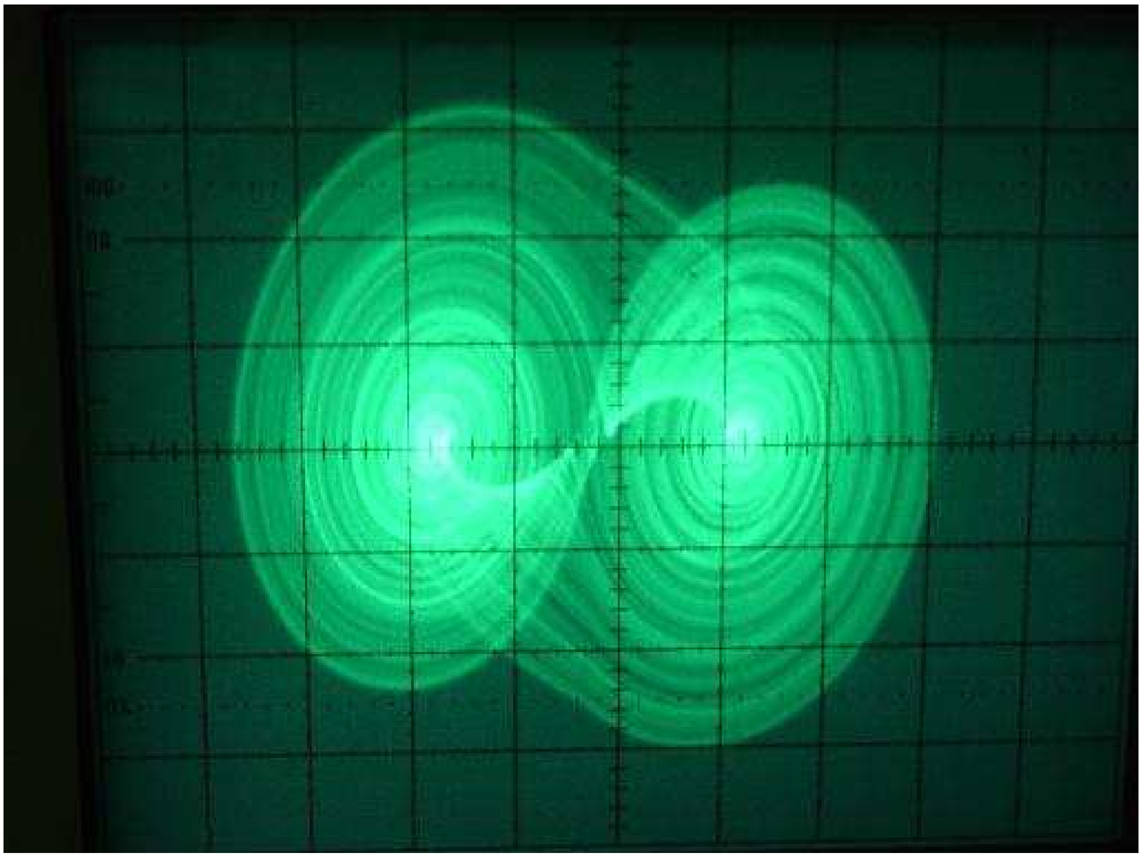

Figure 4: Double scroll attractor.

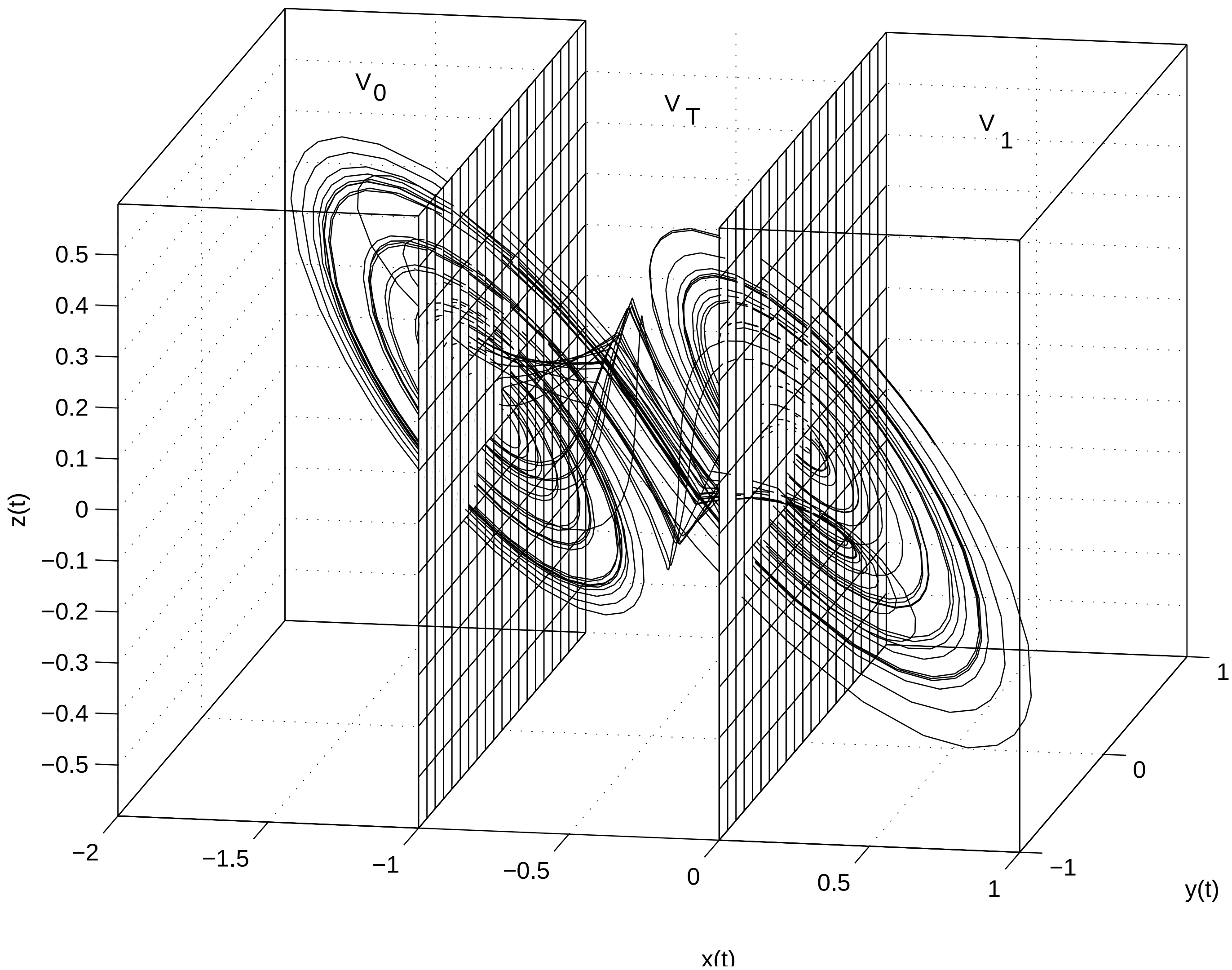

Figure 5: Discretizing the state space for $c_1 = 0$ and $c_2 = -1$.

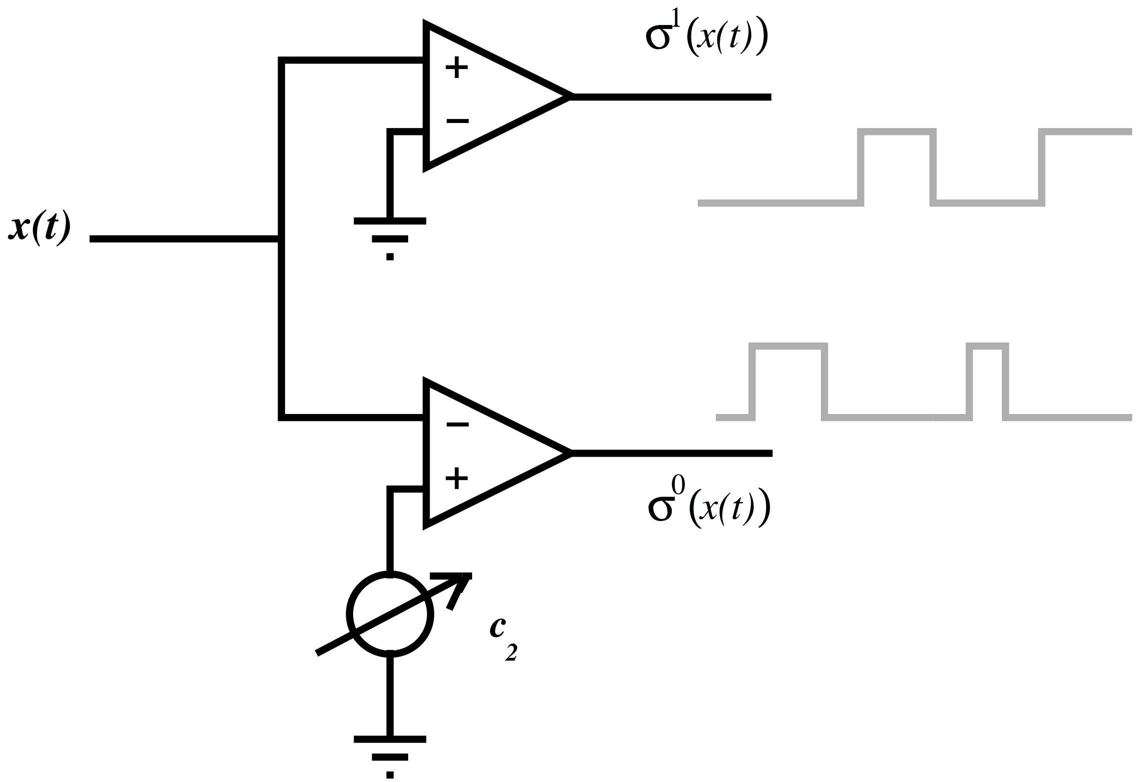

Figure 6: Additional circuit to generate binary sequences.

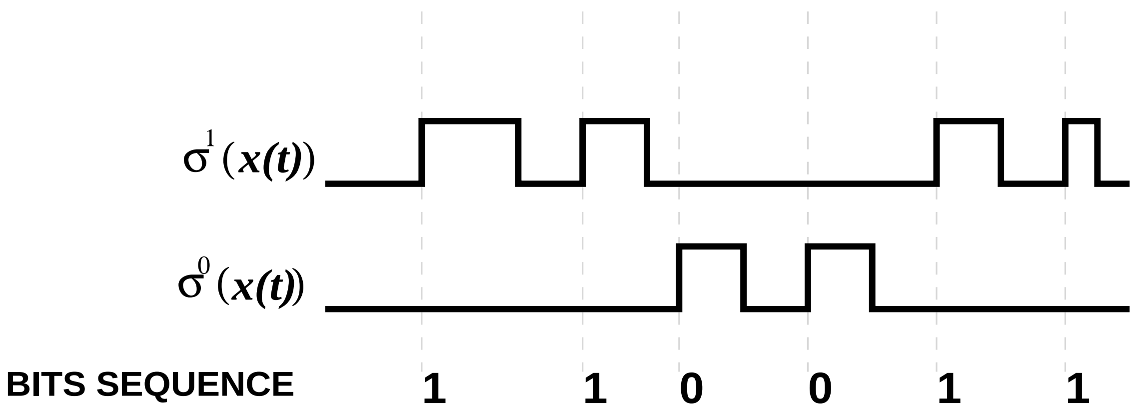

Figure 7: A bits sequence from the comparator outputs.

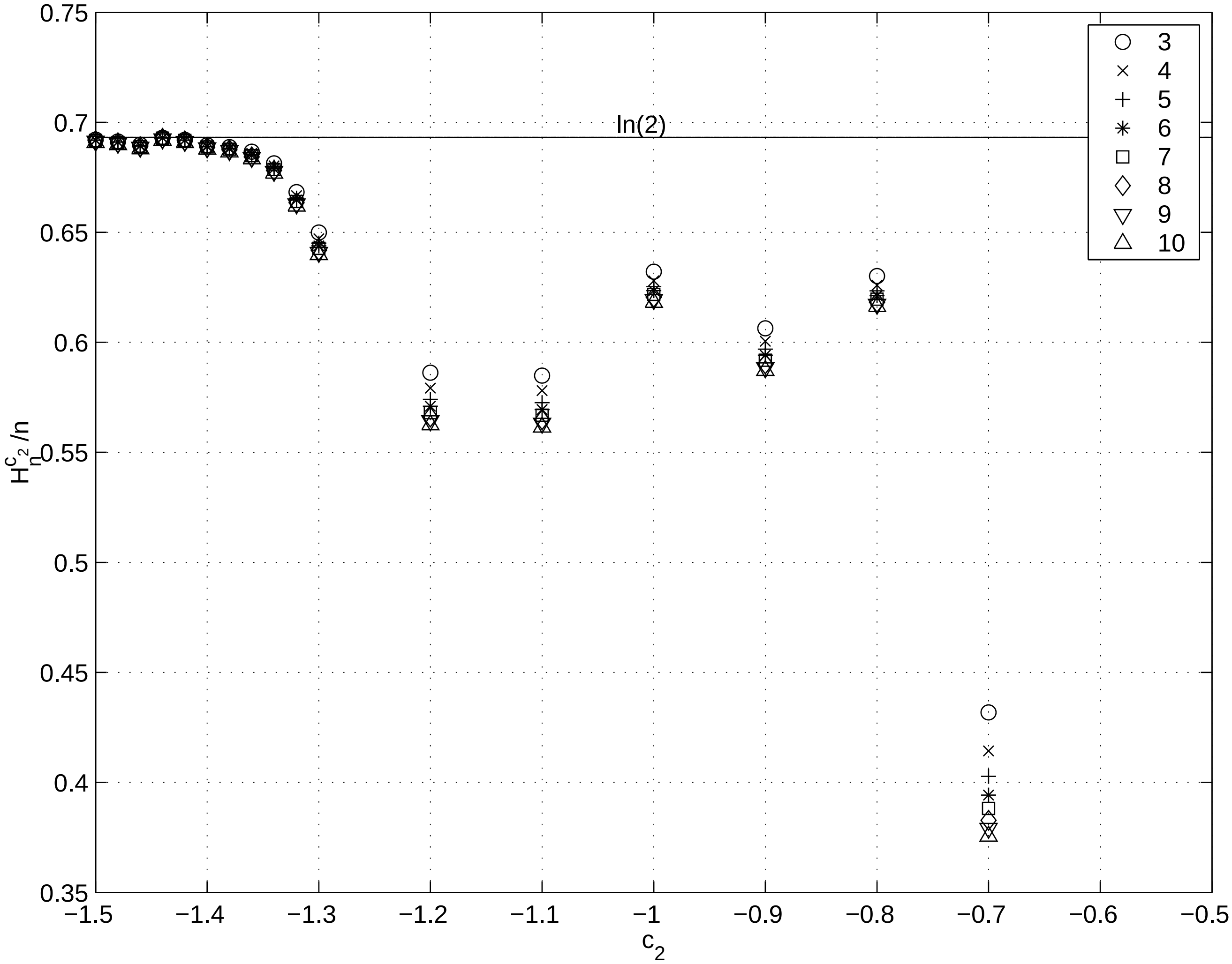

Figure 8: $\frac{H_n^{c_2}}{n}$ as a function of the threshold c_2 for $n = 3, ..., 10$.

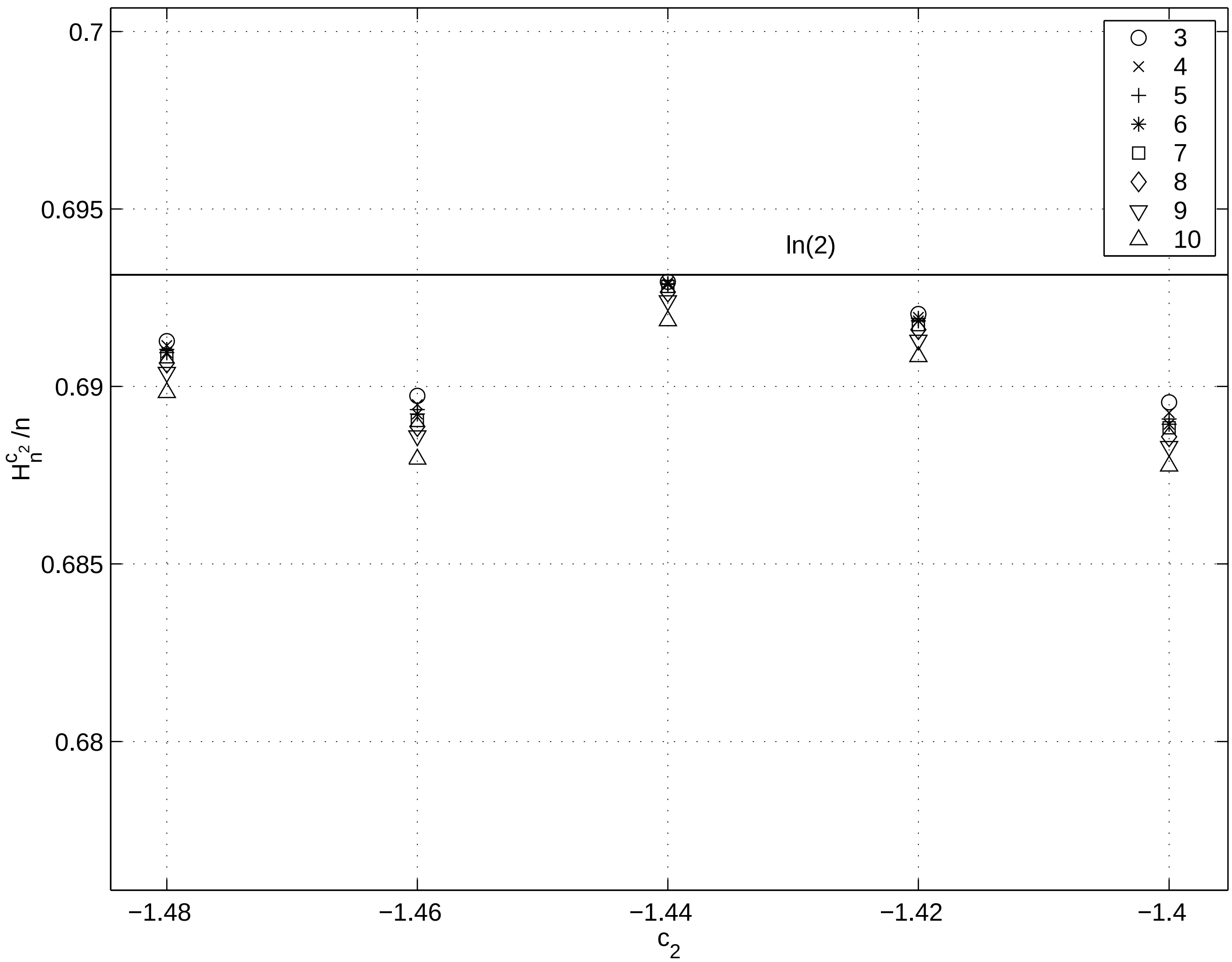

Figure 9: $\frac{H_n^{c_2}}{n}$ for $n = 3, ..., 10$ where $-1.484 \le c_2 \le -1.395$.

Length of run	Required interval
1	$2267 - 2733$
2	$1079 - 1421$
3	$502 - 748$
4	$223 - 402$
5	$90 - 223$
6+	$90 - 223$

Table 1: Runs test table

Run #	Monobit Test	Poker Test	Runs Test		Long Run Test
1	$N_1 = 9943$	$X_3 = 13.9648$	$B_1 = 2614$	$G_1 = 2540$	No long run
			$B_2 = 1201$	$G_2 = 1271$	
			$B_3 = 629$	$G_3 = 624$	
			$B_4 = 313$	$G_4 = 309$	
			$B_5 = 154$	$G_5 = 154$	
			$B_{6+} = 144$	$G_{6+} = 157$	
Passed Test ?	**YES**	**YES**	**YES**		**YES**
2	$N_1 = 9813$	$X_3 = 17.0496$	$B_1 = 2683$	$G_1 = 2559$	No long run
			$B_2 = 1224$	$G_2 = 1230$	
			$B_3 = 614$	$G_3 = 633$	
			$B_4 = 280$	$G_4 = 354$	
			$B_5 = 146$	$G_5 = 155$	
			$B_{6+} = 140$	$G_{6+} = 155$	
Passed Test ?	**YES**	**YES**	**YES**		**YES**
3	$N_1 = 10011$	$X_3 = 18.1248$	$B_1 = 2536$	$G_1 = 2595$	No long run
			$B_2 = 1269$	$G_2 = 1240$	
			$B_3 = 662$	$G_3 = 610$	
			$B_4 = 291$	$G_4 = 299$	
			$B_5 = 149$	$G_5 = 149$	
			$B_{6+} = 149$	$G_{6+} = 163$	
Passed Test ?	**YES**	**YES**	**YES**		**YES**
4	$N_1 = 10052$	$X_3 = 14.9696$	$B_1 = 2513$	$G_1 = 2482$	No long run
			$B_2 = 1215$	$G_2 = 1277$	
			$B_3 = 607$	$G_3 = 634$	
			$B_4 = 343$	$G_4 = 308$	
			$B_5 = 173$	$G_5 = 153$	
			$B_{6+} = 145$	$G_{6+} = 142$	
Passed Test ?	**YES**	**YES**	**YES**		**YES**
5	$N_1 = 10068$	$X_3 = 24.4224$	$B_1 = 2471$	$G_1 = 2498$	No long run
			$B_2 = 1318$	$G_2 = 1289$	
			$B_3 = 609$	$G_3 = 653$	
			$B_4 = 299$	$G_4 = 284$	
			$B_5 = 162$	$G_5 = 155$	
			$B_{6+} = 161$	$G_{6+} = 141$	
Passed Test ?	**YES**	**YES**	**YES**		**YES**

Table 2: Results of $FIPS - 140 - 1$ Specified Tests.