entropy

MDPI

Article

Transition Probability Test for an RO-Based Generator and the Relevance between the Randomness and the Number of ROs

Yuta Kodera [1,*], Ryoichi Sato [1], Md. Arshad Ali [2], Takuya Kusaka [1] and Yasuyuki Nogami [1]

[1] Graduate School of Natural Science and Technology, Okayama University, Okayama 700-8530, Japan; ryoichi_satoh@s.okayama-u.ac.jp (R.S.); kusaka-t@okayama-u.ac.jp (T.K.); yasuyuki.nogami@okayama-u.ac.jp (Y.N.)

[2] Department of Computer Science and Engineering, Hajee Mohammad Danesh Science and Technology University (HSTU), Dinajpur 5200, Bangladesh; arshad@hstu.ac.bd

* Correspondence: yuta_kodera@okayama-u.ac.jp

Abstract: A ring oscillator is a well-known circuit used for generating random numbers, and interested readers can find many research results concerning the evaluation of the randomness with a packaged test suit. However, the authors think there is room for evaluating the unpredictability of a sequence from another viewpoint. In this paper, the authors focus on Wold's RO-based generator and propose a statistical test to numerically evaluate the randomness of the RO-based generator. The test adopts the state transition probabilities in a Markov process and is designed to check the uniformity of the probabilities based on hypothesis testing. As a result, it is found that the RO-based generator yields a biased output from the viewpoint of the transition probability if the number of ROs is small. More precisely, the transitions $01 \to 01$ and $11 \to 11$ happen frequently when the number l of ROs is less than or equal to 10. In this sense, $l > 10$ is recommended for use in any application, though a packaged test suit is passed. Thus, the authors believe that the proposed test contributes to evaluating the unpredictability of a sequence when used together with available statistical test suits, such as NIST SP800-22.

Keywords: true random number generator; ring oscillator; Markov process; hypothesis testing

Citation: Kodera, Y.; Sato, R.; Ali, M.A.; Kusaka, T.; Nogami, Y. Transition Probability Test for an RO-Based Generator and the Relevance between the Randomness and the Number of ROs. *Entropy* **2022**, *24*, 780. https://doi.org/10.3390/e24060780

Received: 4 April 2022
Accepted: 28 May 2022
Published: 31 May 2022

1. Introduction

The study of finding entropy sources is a traditional and essential topic, with attractive randomness in some applications such as key generation and issuing identifiers in the cryptographic field, for example. In practice, the physical inputs or characteristics of an I/O device on a computer, such as a keyboard or a computer mouse, are well-established sources. However, such inputs are not always ideal; for example, a human-related source such as the input from a keyboard would be affected by the user's intention. Since the entropy source should be truly random, researchers have investigated and developed other methods using physical phenomena to overcome these drawbacks.

Researchers and developers have paid much attention to making random number generators (RNG) using digital circuits compact, so that they can be implemented together with other modules. There are two main types of circuits that can easily cause unstable signals. One of them is called metastability [1], which is an intermediate state between high and low and is dealt with as a malfunction of a circuit in product development. However, it is known to have the ideal characteristics to act as an entropy source, and can be implemented with just a pair of NAND gates on a field programmable gate array (FPGA), for example. However, since some time is consumed to converge the vibration during metastability, a generator using metastability sometimes faces problems with efficiency.

The other circuit is an oscillation circuit, called a ring oscillator (RO), consisting of NOT gates that are aligned in a ring shape. Sunar et al. introduced an RNG using the ROs in [2]. It was designed to mix the output of multiple RO circuits by using the XOR operation,

the output of which is synchronized by an internal clock. Though the construction allows bits to be sampled faster than by using metastability, its randomness was required to be discussed further, since the randomness is easily affected by the number of NOT gates and RO circuits.

Wold et al. [3] extended Sunar's proposed RNG circuit in which the respective output of ROs is synchronized by delay flip flops (D-FFs). Since the D-FFs contribute to improving the randomness of RO-based generators, Wold's construction is now widely adopted as an entropy source in various situations. Research on these RO-based generators has been approached from several viewpoints, such as randomness, security, and energy efficiency. The readers can refer to [4–12] for further results about extensions of the RO-based generators and randomness evaluations, for example.

In this paper, the authors focus on Wold's construction and discuss the distribution property of the generator. Furthermore, the relevance between the number of ROs and the quality of randomness is also considered. More precisely, the target concerning the distribution property is the transition probability of bits introduced in the Markov process. This differs from the elements of several famous statistical tests in the sense that the authors' proposed method discusses the uniformity of a sequence from the relevance of bits at time t and $t+1$, for instance. The authors think this approach contributes to a different aspect of a sequence, together with the currently proposed statistical test suites. In addition, in this paper the authors conduct the same test for generators set up with different numbers of ROs. As a result, it is found that there is a significant relationship between the number of ROs and the randomness property.

This paper is organized as follows: Section 2 introduces the fundamentals related to this work, for example, details of the RO-based generator and statistical tests. Sections 3 and 4 propose a test for an RNG designed using ROs, and give experimental observations for some generators with different numbers of ROs, respectively. Finally, Section 5 concludes this paper.

2. Preliminaries

This paper focused on the probability test for an RO-based generator, as well as the relationship between the randomness property and the number of ROs. This section briefly reviews the idea of a true random number generator based on a ring oscillator, Markov process, and hypothesis testing.

2.1. True Random Number Generator Based on Ring Oscillator

This section briefly reviews the fundamentals of an RNG based on ring oscillators, and related works. An RNG is simply referred to as a generator in this paper.

2.1.1. Random Number Generator and Ring Oscillators

RNGs are typically classified into two main classes [13]. One of them is the deterministic RNGs, called pseudorandom number generators (PRNGs). They work algorithmically with a given seed value. Another class is the non-deterministic RNGs, which often adopt some non-reproducible phenomena to generate an ideal random number sequence. Such an ideal sequence is called the true random number generator (TRNG), and many approaches using physical phenomena, called physical RNGs in what follows, have been proposed as a class of TRNG. However, not every RNG can be dealt with as a TRNG, even if it employs a physical phenomena. In this paper, the authors mainly work on evaluating a well-known physical RNG construction from the viewpoint of the unpredictability of sequences.

A representative construction is to use digital circuits to obtain a sequence of digits including bits. There are several approaches such as using noises or the unstable behavior of logic gates. Among them, an oscillation circuit consisting of odd numbers of NOT gates, called a ring oscillator (RO), is widely adopted and studied. As shown in Figure 1, an RO is a circuit that is composed of odd numbers of NOT gates connected in a ring shape.

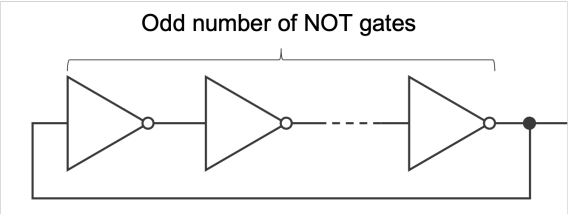

Figure 1. Illustration of a ring oscillator.

Since it can be implemented as a logic circuit in an FPGA, a generator based on ROs is cheaper than one that uses external equipment to observe phenomena to obtain a sequence.

Though external equipment may possibly be interfered with or intermediated by an attacker and leak a sequence as a result, a circuit closed inside an FPGA has an advantage in this regard. In detail, without an adequate environment, data through a circuit are hard to eavesdrop directly. In addition, a generator consisting of a logic circuit is preferred in practical use since it can be embedded with other circuits into a chip.

The output of an RO oscillates due to the recursive input from the right edge NOT gate, as the name stands for. The frequency $f_{t_{\mathrm{NOT}}}$ is known to be given by Equation (1), where t_{pd} and t_{NOT} denote the propagation delay time of a NOT gate and the number of NOT gates, respectively.

$$f_{t_{\mathrm{NOT}}} = \frac{1}{2t_{pd}t_{\mathrm{NOT}}} \tag{1}$$

It is noted that since the propagation delay time changes from high, say t_{pLH}, and high to low, say t_{pHL}, are the same if CMOS devices are used, we can assume $t_{pLH} = t_{pHL}$ and denote the propagation delay time by t_{pd} for simplicity.

Since an RO is not stable because of effects from the external environment, such as thermal noises, for example, the actual oscillation period has time difference T_d from the theoretical oscillation period T, as shown in Figure 2, where $T_d \ll T$.

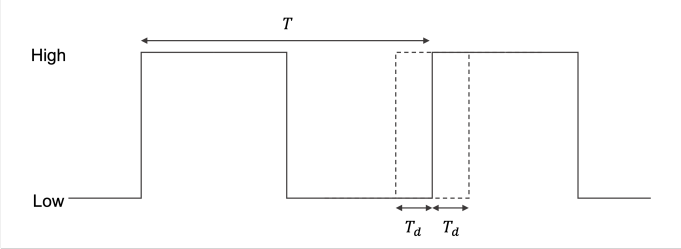

Figure 2. Illustration of an oscillation period in a ring oscillator.

Thus, the oscillation period T_{RO} of the RO is given by $T_{RO} = T \pm T_d$.

This instability is useful for sampling binary symbols, e.g., 0 and 1, and the circuit size can be scalable depending on the number of NOT gates. Therefore, many researchers have focused on ROs to develop a compact and ideal RNG, based on ROs.

2.1.2. Related Works

Sunar et al. introduced ROs to construct a TRNG in [2]. It was designed to mix the output of the respective RO circuit by using the XOR operation, the output of which is synchronized by an internal clock, as shown in Figure 3.

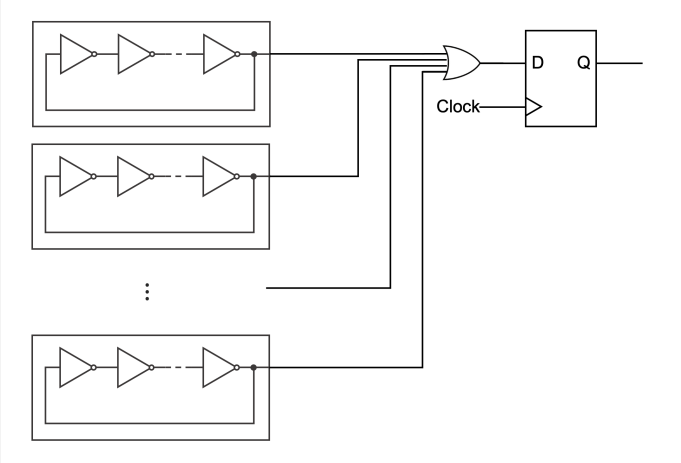

Figure 3. Illustration of the generator proposed by Sunar et al.

In [2], Sunar et al. also discussed several properties, such as the ideal length of a ring, from the theoretical viewpoint.

Based on their construction, Wold et al. [3] extended the circuit to be as shown in Figure 4. Wold et al. successfully enhanced the randomness by inserting D-FFs between ROs and an XOR gate to sample the wave endowed by the ROs. They found that short ROs are better for improving randomness, since the difference in the wave frequency can be easily induced by the restriction of the length. In this paper, the authors mainly deal with Wold's construction to investigate the randomness of continuous digits. In addition, the readers can refer to the results in [4–12] concerning RO-based generators and evaluations for more information.

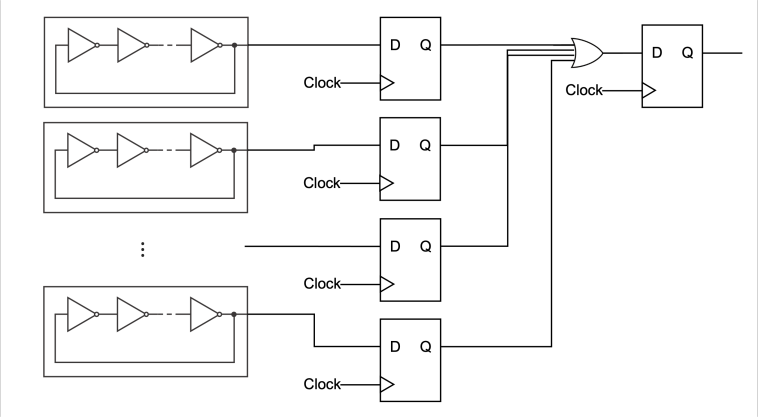

Figure 4. Illustration of the generator proposed by Wold et al.

In a previous work [14], the authors discussed the importance of an XOR gate in a generator based on ROs by approximating the periodicity and investigating the transition probabilities of 2-bit patterns. It was revealed that an XOR gate in an RO-based generator contributes to extending the period length, and the number of ROs is of relevance to the distribution property. In this paper, the authors extend this discussion to the statistical randomness evaluation by using the Markov process and hypothesis testing.

2.2. Markov Process

A Markov process is a stochastic extension of a finite automaton for which state transitions happen probabilistically. It has a memoryless property, which is to say that any additional information about the future behavior of the process cannot be obtained from the past processes in a random process. More precisely, $X_1, X_2, X_3, \ldots$ are random variables and $P\{X \mid Y\}$ denotes the conditional probability of X given Y. Let $S = \{s_1, s_2, s_3, \ldots\}$ be

the state space and let $p_{i,j} = P\{X_n = s_j \mid X_{n-1} = s_i\}$ be the transit probability from s_i to s_j for a positive integer $n > 1$. The memoryless property, referred to as the Markov property, holds the equality as follows:

$$P(X_n = x_n \mid X_{n-1} = x_{n-1}, X_{n-2} = x_{n-2}, \ldots, X_1 = x_1) = P(X_n = x_n \mid X_{n-1} = x_{n-1}), \quad (2)$$

where $x_i \in S$. Such a random process utilizing the Markov property is called a Markov process.

Based on the definition of the Markov process, a Markov chain is defined as a Markov process with discrete time and discrete state space. Thus, a Markov chain is a discrete sequence of states, denoted by S, with random variables $X_1, X_2, X_3, \ldots$ such that the probability of any given state X_n only depends on the current state X_{n-1}, as shown in Equation (2). The process diagram of the Markov chain is a directed graph describing the Markov process. For example, a simple two-state Markov chain can be illustrated as shown in Figure 5.

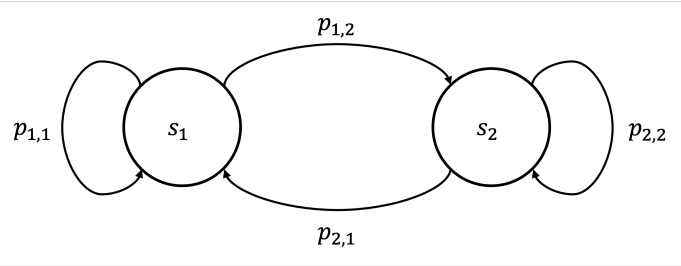

Figure 5. Example of a process diagram.

2.3. Hypothesis Testing and Z-Test

Hypothesis testing is a method of testing whether claims or hypotheses concerning a population are likely to be true. There are two hypotheses: the null hypothesis and an alternative hypothesis. The null hypothesis is a statement about a population parameter, which is assumed to be true. In contradiction to the null hypothesis, the alternative hypothesis is a statement that says the value of the population parameter does not match the value in the null hypothesis.

Hypothesis testing is conducted by following the steps summarized below.

1. State a null hypothesis and alternative hypothesis;
2. Select a random sample from the population;
3. Set a significance level and perform an appropriate statistical test;
4. Decide whether the null hypothesis is valid or not.

The significance level is a criterion to decide the value stated in the null hypothesis. The decision often comes from the outcome of the statistical test, using the p-value, which is the probability of obtaining a sample result under the null hypothesis, which is then compared to the significance level.

A Z-test is a hypothesis test in which the Z-score, also called the Z-statistic, follows a normal distribution. It determines whether the mean of random variables $X_1, X_2, \ldots, X_n$ is equal to a mean m_0 when the variances of $X_i (1 \leq i \leq n)$ are known. It is noted that the test is considered to be accurate if X_k follows the normal distribution, or to be approximately accurate if n is sufficiently large (for example, $n \geq 30$). The test is conducted by assuming that the Z-score follows the standard normal distribution. It is calculated by

$$Z = \frac{\overline{X} - m_0}{\frac{\sigma}{\sqrt{n}}},$$

where $\overline{X} = \sum_{i=1}^{n} X_i / n$ and σ denote the standard deviation.

3. A Test Method and Evaluation Process

This section introduces the details of the test and its evaluation process.

3.1. Background of the Proposed Test

Typically, a sequence generated by PRNGs or TRNGs is evaluated by statistical tests such as TestU01 [15] and NIST SP 800-22 [16]. These are packages of several statistical tests, and users can smoothly check the statistical randomness by running them. For example, the distribution property of a sequence is evaluated by counting the number of bits or comparing a bit pattern with a template. These evaluations are an inseparable part of the distribution property. However, the authors feel that these evaluations cannot fully cover the features of the property.

In this context, the authors introduce the transition probability by considering the Markov process for an RO-based generator from previous research [14]. It is noted that the readers can refer to [4], as a work related to this paper. In brief, this paper focuses on the 2-bit patterns and deals with them in the state $S = \{00, 01, 10, 11\}$, with transition probabilities between each other, where each pattern is derived by splitting a sequence of bits from the beginning without any duplication of the index. Furthermore, the authors extend the discussion of this approach to investigate the properties of an RO-based generator in the following sections.

3.2. Design of a Test

This section briefly introduces the assumptions and process of the test, including evaluation.

3.2.1. Assumptions

In this paper, a 100 MHz clock is used to sample a bit to generate a sequence with an RO-based generator. The state considered in the Markov process is a 2-bit pattern. Therefore, the time space is a discrete set $\mathcal{T} = \{2 \times 10^{-8}, 4 \times 10^{-8}, 6 \times 10^{-8}, \ldots\}$.

An RO-based generator is implemented on an FPGA as a combination of lookup tables (LUT) and D-FFs. Hence, both NOT and XOR gates can be expressed by LUTs. Since the wire length causes a difference in RO circuits, this paper intentionally arranges each element so that they can be in the same condition.

3.2.2. Process of the Test and Evaluation

The test conducted in the next section is composed of three steps, as follows:

1. Set the null hypothesis such that the transition probabilities are equal to $1/4$;
2. Repeat the following sampling and preparation step 1000 times:
   (a) Generate a sequence of length 1Kbits on an FPGA, and repeat it 1000 times to obtain a sample sequence of length 1Mbits in total;
   (b) Split the sequence into 2 bits and calculate the transition probabilities;
   (c) Observe the distribution of probabilities (it should follow the normal distribution) and decide the significance level;
   (d) Conduct the Z-test and calculate p-values.
3. Discussion

First, the assumption of the null hypothesis is clear from the fact that the ideal distribution of 2 bits is the uniform distribution, having an apparent probability of $1/2^2$. Since one of the motivations in this work is to investigate the uniformity of transition probabilities from every pattern, the authors propose sampling a sequence of length 1K bits 1000 times to obtain a p-value. By repeating the collection of p-values 1000 times, the authors decide whether the null hypothesis is approved or not.

In addition, the authors' other motivation is to reveal the relationship between the randomness of sequences and the number of ROs in a generator. Several experimental results for different numbers of ROs are introduced in the next section. It is noted that the authors use the Z-test in the following experiments since the number of samples is large

(1000 samples). However, a *t*-test should be utilized when the readers need more strict and practical evaluation.

4. Experimental Results and Considerations

In this section, the authors show the experimental results of the proposed method. For simplicity, the null hypothesis H_0 and alternative hypothesis H_1 throughout the experiment are $H_0 : \mu = 0.25$ and $H_1 : \mu \neq 0.25$, respectively, where μ denotes the mean of transition probabilities. The FPGA board used to implement the RO-based generator was a Nexys A7-100T Artix-7 series [17], with every RO circuit being composed of only three NOT gates.

4.1. Observation

First, let us begin by briefly confirming whether the probability distribution follows the normal distribution. Figures 6–8 shows the histograms of transition probabilities when the numbers of ROs are 2, 10, and 20.

By observing the figures, it is apparent that the probability distribution follows the normal distribution as the expectation gradually closes to 0.25, depending on the increment in the number of ROs. Thus, the proposed method can be considered applicable to an RO-based generator.

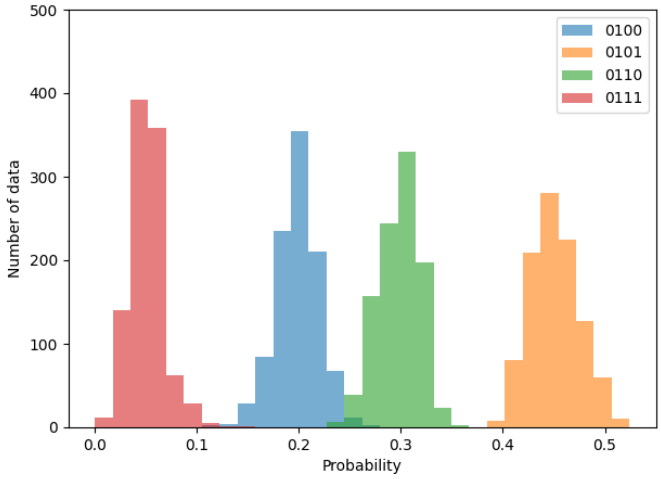

Figure 6. Histogram of transition probabilities when the number of ROs is 2.

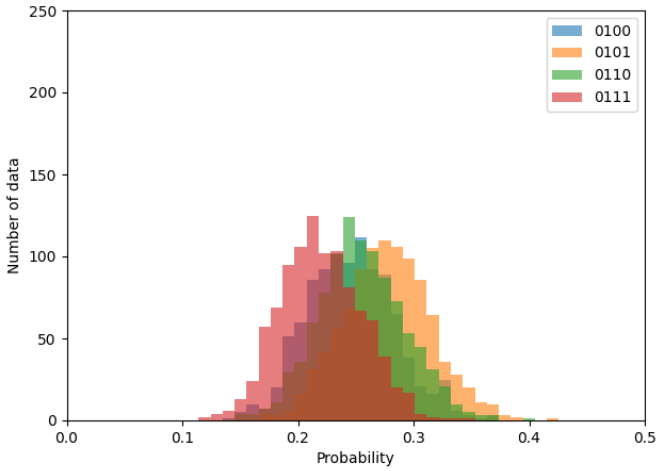

Figure 7. Histogram of transition probabilities when the number of ROs is 10.

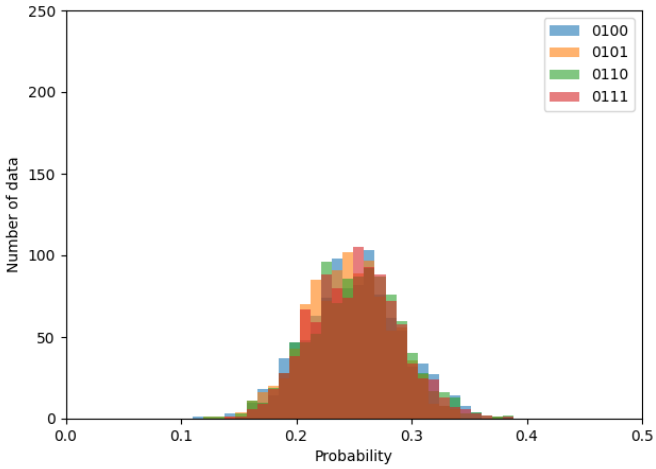

Figure 8. Histogram of transition probabilities when the number of ROs is 20.

4.2. Comparison of p-Values and Considerations

Based on the previous observation, the authors conducted a Z-test and compared the p-values obtained throughout the experiment. It is noted that the significance level is set to 1% to validate the hypothesis more strictly.

Table 1 shows the comparisons of p-values obtained by testing sequences generated with l number of ROs, where $1 \leq l \leq 25$. The element highlighted in red denotes the cases in which the p-value is less than 0.005, and the blue ones the elements greater than or equal to 0.005, respectively. As seen from the table, the null hypothesis tended to be accepted when $l > 10$. Additionally, compared to the other transitions, the null hypotheses for the specific transitions, such as 01 to 01 and 11 to 11 for $l > 10$, are found to be rejected frequently. To check these assumptions further, the authors conducted additional experiments, as given in the next section.

Table 1. Comparisons of p-values.

label	00->00	00->01	00->10	00->11	01->00	01->01	01->10	01->11	10->00	10->01	10->10	10->11	11->00	11->01	11->10	11->11
ro01	0.00000	0.01864	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.02710	0.00000
ro02	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
ro03	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
ro04	0.00000	0.16519	0.00147	0.00000	0.00749	0.00000	0.00000	0.00010	0.00201	0.00000	0.00000	0.00000	0.00000	0.02025	0.10416	0.00000
ro05	0.00000	0.00000	0.00000	0.13086	0.00000	0.00000	0.03604	0.00000	0.00000	0.02970	0.00000	0.00002	0.20960	0.00000	0.00000	0.00000
ro06	0.00000	0.14099	0.00006	0.00000	0.00208	0.00000	0.00000	0.00856	0.10963	0.00000	0.00000	0.41746	0.00000	0.02222	0.32908	0.00000
ro07	0.00000	0.00731	0.00000	0.00004	0.00000	0.00000	0.00130	0.00000	0.00000	0.03426	0.00000	0.00051	0.00000	0.00000	0.01187	0.00000
ro08	0.10281	0.24465	0.17085	0.00734	0.03051	0.15215	0.13458	0.23040	0.10662	0.21383	0.14787	0.01250	0.24931	0.17867	0.08637	0.03173
ro09	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
ro10	0.00000	0.00039	0.00000	0.00001	0.00001	0.00000	0.00000	0.00000	0.00000	0.00046	0.00000	0.00081	0.01576	0.00000	0.00000	0.00000
ro11	0.13946	0.01869	0.00098	0.08739	0.44380	0.46914	0.34012	0.08281	0.44737	0.36448	0.01424	0.02874	0.32228	0.06593	0.47494	0.03191
ro12	0.00385	0.47350	0.15195	0.18107	0.40731	0.11576	0.38456	0.30379	0.14207	0.41450	0.32107	0.48765	0.11917	0.25427	0.12967	0.46923
ro13	0.00000	0.14999	0.00000	0.04430	0.14790	0.03085	0.17533	0.00365	0.00614	0.09182	0.09205	0.12236	0.04304	0.00028	0.14010	0.00004
ro14	0.06962	0.05386	0.35777	0.02385	0.23138	0.00965	0.40748	0.31340	0.18214	0.05755	0.10319	0.00227	0.17186	0.45366	0.40694	0.06193
ro15	0.31498	0.30421	0.02443	0.44159	0.28121	0.00216	0.19350	0.29419	0.39260	0.23912	0.10166	0.31176	0.13531	0.00062	0.08279	0.43614
ro16	0.09703	0.49858	0.37326	0.22738	0.01012	0.00000	0.06982	0.49266	0.00313	0.39870	0.01187	0.20685	0.24796	0.22161	0.29137	0.08422
ro17	0.05079	0.09696	0.24921	0.36462	0.04660	0.00018	0.34975	0.26382	0.01588	0.46235	0.02480	0.15132	0.41270	0.27786	0.16034	0.01318
ro18	0.28229	0.40365	0.47591	0.29814	0.01552	0.11089	0.26885	0.00618	0.40131	0.38623	0.36186	0.46723	0.39206	0.26387	0.00104	0.00001
ro19	0.12650	0.04094	0.11813	0.17239	0.11930	0.00775	0.08286	0.19468	0.11954	0.29621	0.07217	0.45569	0.20664	0.45251	0.04674	0.03825
ro20	0.39509	0.22235	0.34472	0.03331	0.03768	0.05363	0.20799	0.03599	0.39703	0.16654	0.41974	0.35975	0.41668	0.25724	0.14209	0.00392
ro21	0.11472	0.44697	0.29620	0.49856	0.19518	0.00155	0.47680	0.12417	0.41490	0.28599	0.41084	0.19149	0.12177	0.37779	0.48512	0.47740
ro22	0.02919	0.14594	0.02543	0.37998	0.18049	0.01831	0.31116	0.41637	0.43679	0.18288	0.33015	0.46905	0.02866	0.31356	0.40753	0.00073
ro23	0.06241	0.10480	0.01803	0.49077	0.20032	0.00412	0.16405	0.49387	0.46559	0.29056	0.21001	0.25081	0.04402	0.26721	0.00164	0.00316
ro24	0.41387	0.32234	0.33352	0.31339	0.31293	0.33302	0.39807	0.48369	0.04025	0.16643	0.01380	0.26936	0.04839	0.10004	0.05871	0.00211
ro25	0.13377	0.46330	0.49101	0.34563	0.33144	0.00001	0.12796	0.02834	0.12632	0.01423	0.08422	0.05710	0.30308	0.13534	0.30415	0.03729

4.3. Post Evaluations for Consideration

The following two characteristics are found throughout the above comparison; thus, the authors carried out further experiments to confirm the likelihood.

- The null hypothesis $H_0 : \mu = 0.25$ is accepted when $l > 10$;
- The null hypothesis concerning the transition probabilities from 01 to 01 and from 11 to 11 are often rejected.

The first assumption shows that the mean of transition probabilities is 0.25 when $l > 10$, which is assumed to be an ideal result for the authors. On the other hand, the second points out that the specific transitions do not happen uniformly.

The authors conduct additional experiments from different viewpoints, as described below, to confirm these assumptions.

1. Comparing the number of sequences that could pass the test (1000 trials);
2. Comparing the difference in the results when the number of NOT gates in an OR circuit is changed (1000 trials).

The experimental results in terms of transition probabilities for each state are introduced in Figures 9–12. The horizontal and vertical axes reflect the number of ROs and the number of sequences of length 1 Mbits for which transition probabilities were able to pass the hypothesis test. As seen from the graphs, the number of sequences will gradually become flat for $l > 10$. However, Figures 10 and 12 also show that the transitions from 01 to 01 and from 11 to 11 are relatively low compared with the other transitions.

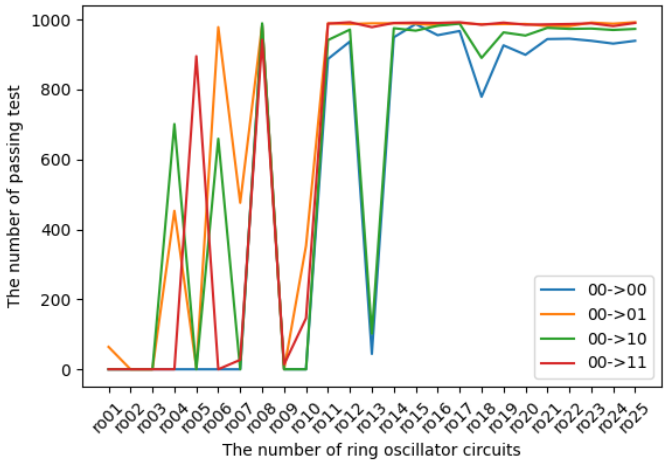

Figure 9. The transition probabilities from 00 (three NOT gates).

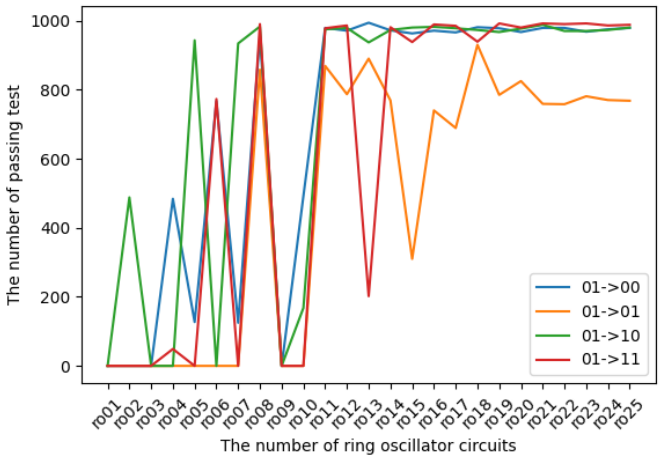

Figure 10. The transition probabilities from 01 (three NOT gates).

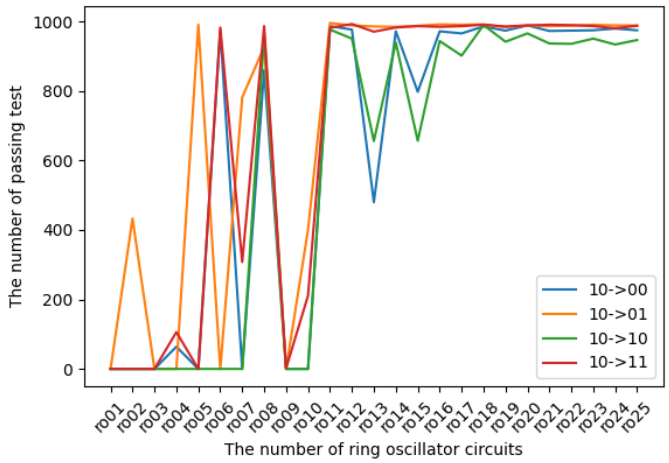

Figure 11. The transition probabilities from 10 (three NOT gates).

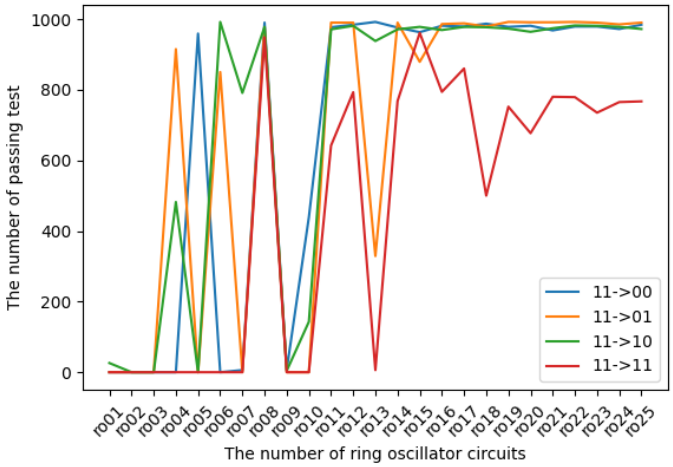

Figure 12. The transition probabilities from 11 (three NOT gates).

In the same way, experiments were conducted with the RO-based generator consisting of seven NOT gates. The results are shown in Figures 13–16. Comparing the figures in Figures 9–16, the readers can find similar characteristics in both graphs, and the assumptions mentioned above are considered to be true.

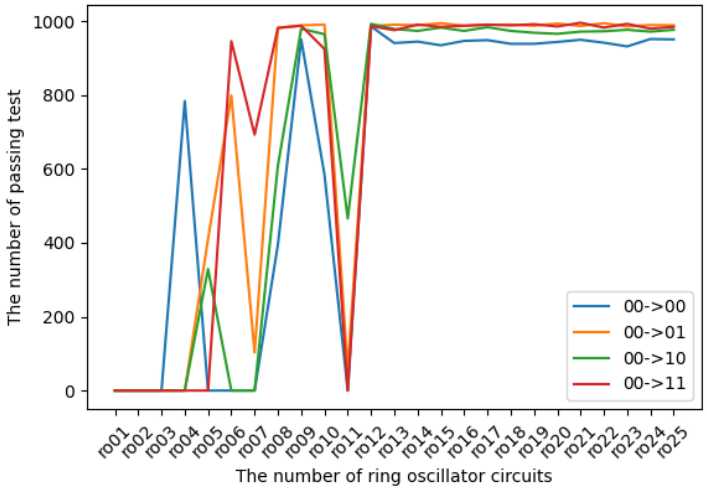

Figure 13. The transition probabilities from 00 (seven NOT gates).

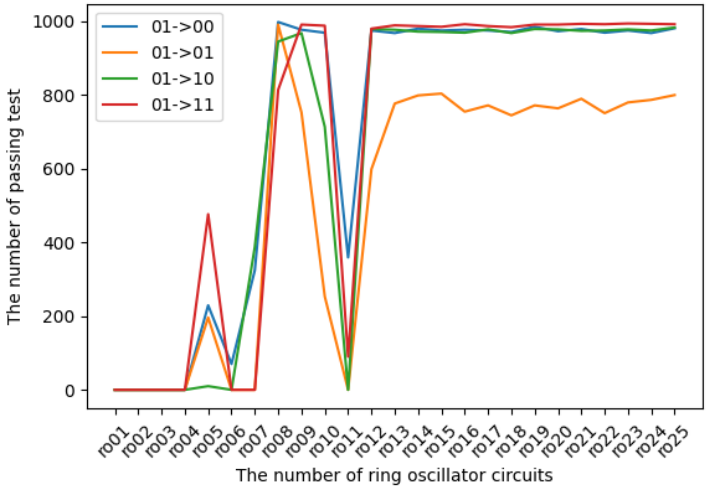

Figure 14. The transition probabilities from 01 (seven NOT gates).

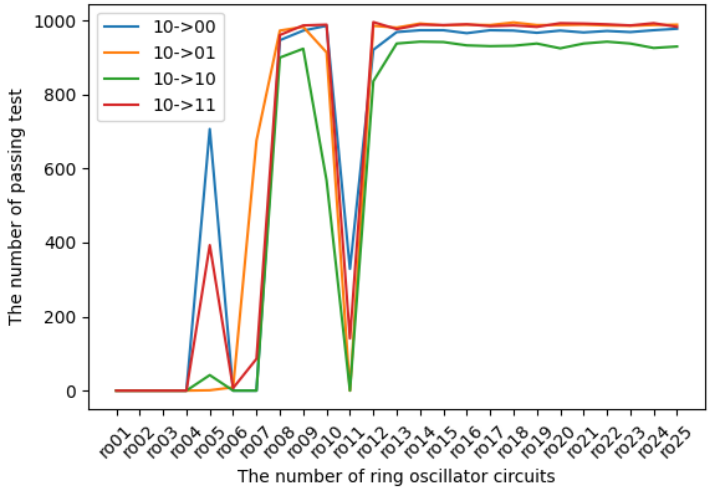

Figure 15. The transition probabilities from 10 (seven NOT gates).

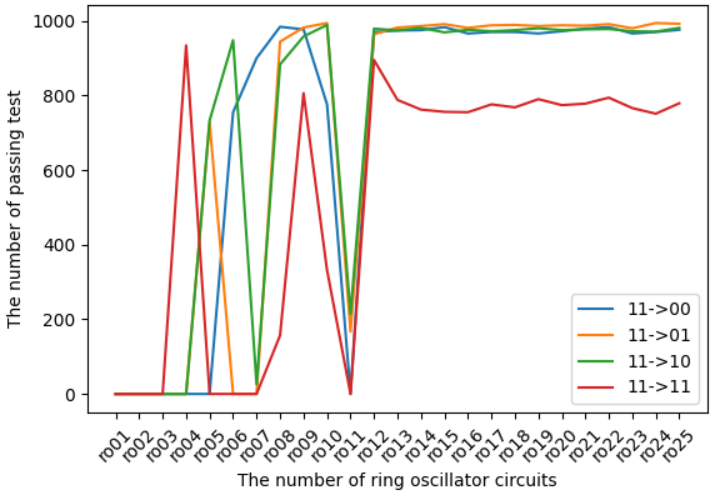

Figure 16. The transition probabilities from 11 (seven NOT gates).

5. Conclusions

This paper focused on an RO-based generator originally proposed by Wold et al.; in addition, the authors proposed a statistical test regarding the state transition probabilities of 2 bits for the generator. The purpose of such a test is to check the uniformity of the respective transition patterns, such as 00, 01, 10, and 11. This statistical test was applied for RO-based generators consisting of different numbers of ROs. As a result, it was found that the randomness of an RO-based generator depends on the number l of ROs, and the result shows that l should be larger than 10.

Therefore, the authors successfully evaluated the randomness of RO-based generators numerically, and we can conclude from this study that the circuit of RO-based generators becomes complex depending on the increment in ROs. In addition, it tells us that the users have to recognize the bias hidden in the transition probabilities, especially for practical use.

However, since this paper only focused on the 2-bit case and did not formulate the experimental results, the authors would like to utilize larger bit patterns and different boards to explore the characteristic equation in future works.

Author Contributions: Conceptualization, Y.K. and M.A.A.; Data curation, Y.K. and R.S.; Formal analysis, Y.K., R.S., T.K. and Y.N.; Funding acquisition, Y.K. and Y.N.; Investigation, Y.K., R.S., M.A.A. and T.K.; Methodology, Y.K. and M.A.A.; Project administration, Y.K.; Resources, T.K. and Y.N.; Software, R.S., T.K. and Y.N.; Supervision, Y.N.; Validation, Y.K. and R.S.; Visualization, R.S.; Writing—original draft, Y.K.; Writing—review and editing, Y.K., R.S., M.A.A., T.K. and Y.N. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by the JSPS KAKENHI Grant-in-Aid for Research Activity Start-up (20K23327) and the Grant-in-Aid for Challenging Research (Pioneering) (20K20484).

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

References

1. Kleeman, L.; Cantoni, A. Metastable Behavior in Digital Systems. *IEEE Des. Test Comput.* **2008**, *4*, 4–19. [CrossRef]
2. Sunar, B.; Martin, W.J.; Stinson, D.R. A Provably Secure True Random Number Generator with Built-In Tolerance to Active Attacks. *IEEE Trans. Comput.* **2007**, *56*, 109–119. [CrossRef]
3. Wold, K.; Tan, C.H. Analysis and enhancement of random number generator in FPGA based on oscillator rings. In Proceedings of the 2008 International Conference on Reconfigurable Computing and FPGAs, Cancun, Mexico, 3–5 December 2008.
4. Ryabko, B.Y.; Monarev, V.A. Using information theory approach to randomness testing. *J. Stat. Plan. Inference* **2005**, *133*, 95–110. [CrossRef]
5. Marton, K.; Suciu, A.; Ignat, I. Randomness in Digital Cryptography: A Survey. *Rom. J. Inf. Sci. Technol.* **2010**, *13*, 219–240.
6. Bochard, N.; Bernard, F.; Fischer, V.; Valtchanov, B. True-Randomness and Pseudo-Randomness in Ring Oscillator-Based True Random Number Generators. *Int. J. Reconfig. Comput.* **2010**, *2010*, 879281. [CrossRef]
7. Tuncer, T.; Avaroğlu, E.; Türk, M.; Ozer, A.B. Implementation of Non-periodic Sampling True Random Number Generator on FPGA. *J. Microelectron. Electron. Compon. Mater.* **2014**, *44*, 296–302.
8. Cao, Y.; Chang, C.-H.; Zheng, Y.; Zhao, X. An energy-efficient true random number generator based on current starved ring oscillators. In Proceedings of the 2017 Asian Hardware Oriented Security and Trust Symposium (AsianHOST), Beijing, China, 19–20 October 2017. [CrossRef]
9. Anandakumar, N.N.; Sanadhya, S.K.; Hashmi, M.S. FPGA-Based True Random Number Generation Using Programmable Delays in Oscillator-Rings. *IEEE Trans. Circuits Syst. II Express Briefs* **2020**, *67*, 570–574. [CrossRef]
10. Lin, J.; Wang, Y.; Zhao, Z.; Hui, C.; Song, Z. A New Method of True Random Number Generation based on Galois Ring Oscillator with Event Sampling Architecture in FPGA. In Proceedings of the 2020 IEEE International Instrumentation and Measurement Technology Conference (I2MTC), Dubrovnik, Croatia, 25–28 May 2020. [CrossRef]
11. Fujieda, N. On the Feasibility of TERO-Based True Random Number Generator on Xilinx FPGAs. In Proceedings of the 2020 30th International Conference on Field-Programmable Logic and Applications (FPL), Gothenburg, Sweden, 31 August–4 September 2020. [CrossRef]
12. Choi, S.; Shin, Y.; Yoo, H. Analysis of Ring-Oscillator-based True Random Number Generator on FPGAs. In Proceedings of the 2021 International Conference on Electronics, Information, and Communication (ICEIC), Jeju, Korea, 31 January–3 February 2021. [CrossRef]
13. Koç, Ç.K. *Cryptographic Engineering*; Springer: Boston, MA, USA, 2009.
14. Sato, R.; Kodera, Y.; Ali, M.A.; Kusaka, T.; Nogami, Y.; Morelos-Zaragoza, R.H. Consideration for Affects of an XOR in a Random Number Generator Using Ring Oscillators. *Entropy* **2021**, *23*, 1168. [CrossRef] [PubMed]

15. L'Ecuyer, P.; Simard, R. TestU01: AC library for empirical testing of random number generators. *ACM Trans. Math. Softw.* **2007**, *33*, 1–40. [CrossRef]
16. Rukhin, A.; Soto, J.; Nechvatal, J.; Smid, M.; Barker, E.; Leigh, S.; Levenson, M.; Vangel, M.; Banks, D.; Heckert, N.; et al. *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*; NIST Special Publication 800-22 Revision 1a; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2010. Available online: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-22r1a.pdf (accessed on 22 March 2022).
17. Nexys A7 FPGA Board Reference Manual. Available online: https://reference.digilentinc.com/_media/reference/programmable-logic/nexys-a7/nexys-a7_rm.pdf (accessed on 22 March 2022).