On the feasibility of TERO-based true random number generator on Xilinx FPGAs

Naoki Fujieda
Department of Electrical and Electronics Engineering, Faculty of Engineering,
Aichi Institute of Technology, Toyota, Aichi, Japan
nfujieda@aitech.ac.jp

***Abstract*—A True Random Number Generator (TRNG) is an essential component for security applications of FPGAs. Its requirements include small logic area, high throughput, sufficient randomness backed with a mathematical model, and feasibility — ease of implementation. This paper focuses on TRNGs based on a Transition Effect Ring Oscillator (TERO) and presents a three-path configurable TERO (TC-TERO), an improved implementation of TERO that achieves high feasibility with a minimal amount of hardware. According to the evaluation with a Xilinx Artix-7 FPGA, a TC-TERO with a 20-bit configurable parameter only required 40 LUTs. By selecting one of the promising parameters, the proposed TRNG passed AIS-31 Procedure A without post-processing and NIST SP 800-22 with a simple debiasing.**

***Index Terms*—Random number generation, Field programmable gate arrays, Transition effect ring oscillator, AIS-31**

I. Introduction

A True Random Number Generator (TRNG) is an essential component for security applications. It generates a sequence of random numbers, which is typically used as an encryption key and nonce of challenge-response protocols, with various kinds of entropy sources. When implementing a TRNG on Field Programmable Gate Arrays (FPGAs), it is desirable to utilize their own logic elements as an entropy source, rather than external resources such as flash memory cells [13].

Major requirements for TRNGs on FPGAs are fourfold: small logic area, high throughput, sufficient randomness backed with a mathematical model, and feasibility. Logic area is quantified by the number of required logic elements, while throughput means the bit rate of generation. These two features are strongly related to power and energy consumption and comprehensively compared by an area-delay product, the product of the area and the inverse of the throughput.

To guarantee the safety, recent TRNGs have been required to have their own mathematical models that support their randomness. Well-known test suites such as NIST SP 800-22 [14] only assure that there are no flaws in the statistical properties of the generated random numbers. The AIS-31 standard [7], proposed by the Federal Office for Information Security in Germany, requires an evidence that a random number sequence, obtained from physical noise source and not post-processed, has sufficient entropy. A 2016 survey [12] enumerated TRNGs for FPGA devices that complied with this standard.

Feasibility, or ease of implementation, is also an important property for practical TRNGs. Even if logic elements in a circuit block are placed in the same way, its characteristics may vary due to manufacturing variation. TRNGs should be designed to harvest sufficient entropy regardless of them. At least, they should be independent from origin of placement. If this condition is met, we can use the same programming file for all devices with the same part number. Otherwise, we would have to search for a preferable position per device through a repetition of logic synthesis. In the survey [12], this is the criterion that distinguishes score of 2 or more from 1 in feasibility. One solution is to make a TRNG configurable and give a proper parameter to it. It is more preferable to have a mechanism to find a parameter automatically.

In this paper, we focus on TRNGs based on the Transition Effect Ring Oscillator (TERO) [2], [15], which is one of the AIS-31 compliant TRNGs for FPGAs [12]. TERO-based TRNGs have small logic area and high throughput, as well as ones based on the Coherent Sampling Ring Oscillator (COSO) [1], [8], [9]. A disadvantage of TERO is that its characteristics are strongly dependent on origin of placement. Recently, a mechanism to make a COSO-based TRNG configurable has been proposed [11]. Based on knowledge on this mechanism and TERO, we present a highly feasible TERO-based TRNGs on Xilinx FPGAs.

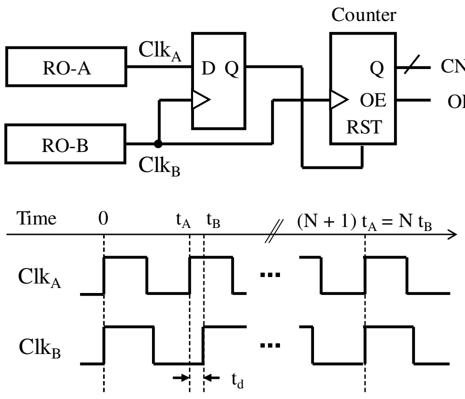

Fig. 1. Architecture and working principle of the COSO-based TRNG.

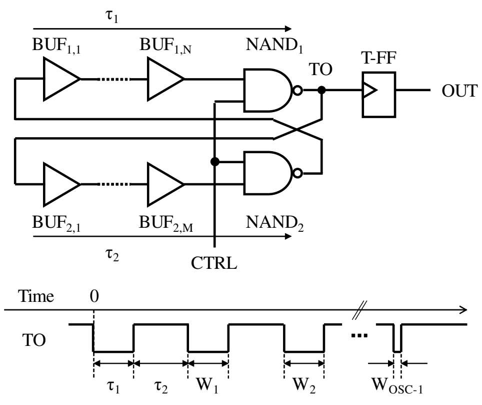

Fig. 2. Architecture and working principle of the TERO-based TRNG.

The contributions of this paper are as follows:

- We present an implementation methodology of the TERO-based TRNG that takes its stochastic model [2] into account and avoids unwanted behavior.
- We propose a *Three-path Configurable Ring Oscillator (TC-RO)*, based on the existing Configurable Ring Oscillator (C-RO) [11], that greatly reduce the number of required logic elements.
- We demonstrate that a TRNG that meets all the requirements — small logic area, high throughput, sufficient randomness, and feasibility — can be realized by applying TC-RO to the TERO-based TRNG (i.e. *TC-TERO*).

We describe them in Sections III, IV, and V, respectively.

II. Existing TRNG Cores

In this section, we outline the organization and the operating principle of existing TRNG circuits, based on COSO (Coherent Sampling Ring Oscillator) and TERO (Transition Effect Ring Oscillator).

A. Coherent Sampling Ring Oscillator

Figure 1 depicts the architecture and the operating principle of the COSO-based TRNG. Its mathematical models have been presented in [1], [9]. The outputs of two ring oscillators, $\mathrm{Clk_A}$ and $\mathrm{Clk_B}$, are connected to the data and the clock inputs of a D flip-flop (D-FF), respectively. The periods of the oscillators are denoted by t_A and t_B. For ease of explanation, we assume that the ratio of the periods is $t_A : t_B = N : (N+1)$, which means $\mathrm{Clk_A}$ goes $t_d = t_B - t_A = t_A/N$ faster than $\mathrm{Clk_B}$ for each cycle. If we suppose the both signals rise at time zero, the D-FF will capture consecutive '1's and '0's before meeting both rising edges again at the time Nt_B. The number of '1's is measured by a counter. Although the expected counter value becomes $N/2 = (t_A/t_d)/2$, the actual counter value varies due to jitter of the signals. COSO utilizes the least significant bits (LSBs) of the counter value as a source of entropy.

The most important parameter for COSO is t_d because it offers a tradeoff between the variation of counter value and the bit rate of generation. A problem of COSO is feasibility, or how to get proper t_d from the pair of clock signals. To manipulate t_d, some similar types of TRNGs utilize a pair of PLLs (Phase-locked Loops) [3] or DCMs (Digital Clock Managers) [6], instead of ring oscillators. Although these TRNGs are more feasible than COSO, jitter obtained from them is relatively small and power consumption tends to be large [12].

B. Transition Effect Ring Oscillator

Figure 2 describes the architecture and the operating principle of the TERO-based TRNG. The ring consists of two branches and each branch has one NAND gate and zero or more buffers (or even number of inverters). The numbers of buffers in the respective branches are denoted by N and M and their delay are denoted by τ_1 and τ_2. Without loss of generality, we can assume that $\tau_1 < \tau_2$. The relative difference of the delay is calculated by $\Delta_r = (\tau_2 - \tau_1)/(\tau_1 + \tau_2)$. When the control signal CTRL rises from '0' to '1' at time zero, the ring begins to oscillate. Logically, a repetition of '0' for τ_1 and '1' for τ_2 will appear at TO (TERO output).

In actual CMOS circuits, gates have an effect of amplifying the relative difference of pulse widths of '0' and '1'. This means, without considering jitter, that the pulse width of '0' becomes shorter every time the pulse goes around the ring. The expected value of the pulse width of '0' after it circles k times, W_k, is calculated by the following formula:

$$W_k = \frac{\tau_1 + \tau_2}{2}(1 - \Delta_r R^k),$$

where R is an amplification factor. The oscillation stops when W_k reaches zero (i.e. $\Delta_r R^k \geq 1$). Such k, which means the number of oscillations N_{OSC}, is measurable by counting the number of rising edges at TO. The distribution of N_{OSC} is modeled by R, Δ_r, and the relative jitter σ_r [2]. The larger σ_r, the larger the variance of the distribution. The T flip-flop (T-FF) in Fig. 2 counts the LSB of N_{OSC} as a source of entropy. Since the model parameters are highly dependent on device and placement, TERO also has a problem on feasibility.

The oscillation of TERO is a behavior when an RS latch transits from the metastable state to one of the stable states. Metastability of latches [4], [5] or flip-flops [10] is also

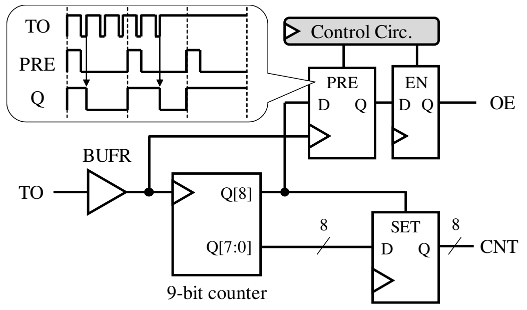

Fig. 3. Proposed implementation of TERO-based TRNG controller.

leveraged by other types of TRNGs. They use the final output after the oscillation, rather than the number of oscillations. However, stochastic models for them have not been established so far.

III. Implementation of TERO-based TRNG for Xilinx FPGAs

The stochastic model of TERO [2] includes two important implications. The first one is that too small Δ_r might result in an infinite oscillation. This unwanted behavior can be avoided by monitoring the number of oscillation by a counter in order to terminate it. The second one is that the relative jitter σ_r becomes larger by making the delay of the ring $\tau_1+\tau_2$ smaller. This means that the number of the buffers should be as small as possible. However, timing constraints in the counter should also be taken into account. If the delay of the ring (i.e. the number of the buffers) is too small, setup time violation will occur. If there is a clock skew among flip-flops in the counter, hold time violation may also occur. Both of them lead to malfunctions of the counter.

Based on the above considerations, we propose an implementation of TERO-based TRNG controller suitable for Xilinx FPGAs. Figure 3 depicts the block diagram of the proposed controller. The input signal TO is the output of the TERO ring shown in Fig. 2. A buffer and a NAND gate in the ring are instantiated as 1-bit and 2-bit LUTs, respectively, with `DONT_TOUCH` attributes. The numbers of buffers in the branches are empirically set to $N = 3$ and $M = 5$. Assuming that all the NAND gates and buffers have the same delay, $\tau_1 : \tau_2 = 4 : 6$ (i.e. $\Delta_r = 0.2$) is expected. The TO signal drives the counter, along with a D-FF for detecting the end of oscillation, through a regional clock buffer (BUFR). The use of a clock buffer is recommended in Xilinx FPGAs to minimize the clock skew [16]. A control circuit and D-FFs of the output signals (CNT and OE) are driven by an external system clock in order to communicate to other circuits.

The point is that the bit width of the internal counter is set to 9 bits. When the counter value reaches 2^8, its MSB (Q[8]) becomes ‘1’ and the counter output (CNT) is set to `0xff`. The D-FF for terminal detection is periodically precharged (PRE) to ‘1.’ Until the counter saturates, the D-FF becomes ‘0’ when TO rises. At the end of each period, another D-FF is enabled (EN) to capture the precharged D-FF as OE. It becomes ‘1’ if the oscillation has ended or the counter has been saturated. In this paper, the cycle time of the PRE signal is set to 40 ns to avoid a metastability in the precharged D-FF.

TABLE I
Arithmetic mean and standard deviation of the counter values.

X \ Y		16	32	48	64	80
4	Avg.	23.21	19.02	60.47	28.84	25.13
	S.D.	0.68	0.30	4.01	0.73	0.46
8	Avg.	61.35	21.87	40.46	167.03	254.97
	S.D.	6.10	0.44	1.22	23.17	0.83

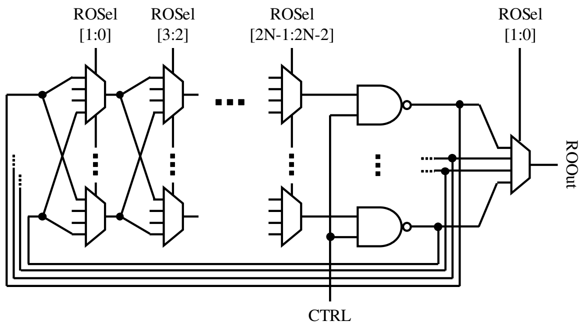

Fig. 4. Architecture of the configurable ring oscillator [11].

In addition, an infinite oscillation might still occur because of timing violation at the counter. We introduce a timeout period as a fail-safe. The control circuit forcibly sets OE to ‘1’ after the timeout. The period is set to 2 μs, which is certainly longer than the time for saturation of the counter.

We placed this TERO implementation on different places of an FPGA, and measured generated counter values. We set ten different origins of placement (`RLOC_ORIGIN`) by combining two X coordinates (4 and 8) and five Y coordinates (16, 32, 48, 64, and 80). An origin with an X coordinate of i and a Y coordinate of j is expressed as XiYj. The relative position of logic elements to the origin was fixed by `RLOC` constraints. In this paper, circuits were synthesized by Vivado 2019.1 and the target board was Digilent Arty (or Arty A7-35), which included an Artix-7 XC7A35T FPGA. Note that the counter value may saturate at `0xff` = 255.

Table I summarizes the arithmetic mean (Avg.) and the standard deviation (S.D.) of the 2^{16} counter values. The origins of X4Y48 and X8Y16 gave modest counter values, which was well dispersed and suitable for a source of entropy. However, the value was too small and almost constant in six out of ten cases. In the other cases (X8Y64 and X8Y80), the value was too large and sometimes or almost always saturated. These results imply that, even if it is carefully designed and implemented, TERO is still highly dependent on origin of placement.

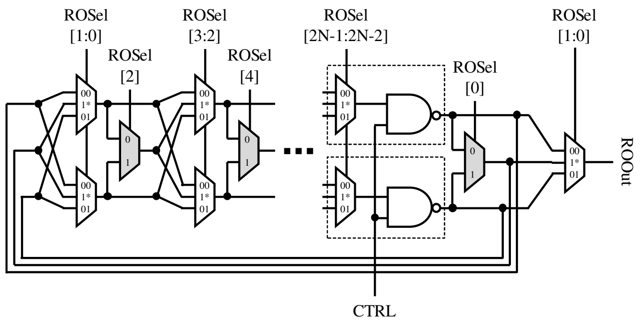

Fig. 5. Architecture of the proposed TC-RO.

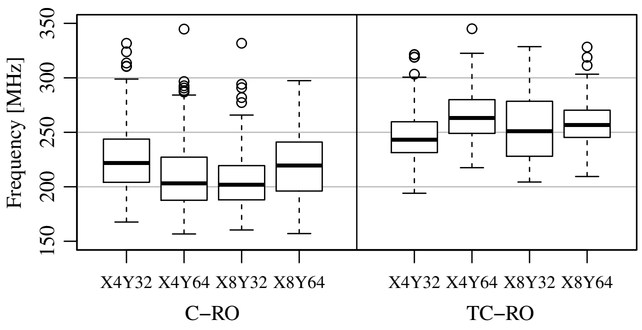

Fig. 6. Distribution of oscillation frequency of C-RO and TC-RO.

IV. Three-path Configurable Ring Oscillator

A. *Configurable Ring Oscillator*

To deal with the problem of feasibility of COSO (explained in Section II.A), a configurable ring oscillator [11] was proposed. Figure 4 describes the architecture of the configurable ring oscillator, denoted as C-RO in this paper. It has four copies of a set of NAND gates and buffers and each buffer is replaced by a 4:1 multiplexer. Each pair of bits of a parameter ROSel (ring oscillator selection) determines which NAND gate or multiplexer in the previous stage is used. The frequency of the ring can be configured by selecting a set of logic elements by the parameter. The output signal (ROOut) is selected by another multiplexer.

When implementing a C-RO from a ring oscillator with N buffers, the number of required 6-input LUTs is $4N + 5$ and the bit width of ROSel becomes $2N$, considering the number of input ports of a 4:1 multiplexer (including selection input) is six. This means that at least two additional LUTs are required to obtain one bit of parameter in the C-RO.

B. *Three-path Configurable Ring Oscillator*

To optimize the C-RO on a Xilinx FPGA, we utilize a multiplexer called F7MUX, which takes outputs of two LUTs as data input. In general, it is used to organize one 7-input LUT from two 6-input LUTs. Another bit of parameter can be obtained by selecting whether outputs of LUTs pass through F7MUX or not.

Figure 5 describes the architecture of the proposed three-path configurable ring oscillator (TC-RO). It has two copies, rather than four, of a set of NAND gates and buffers. The number of data inputs of a multiplexer becomes three: two of them come from LUTs and the other comes from F7MUX (shown in gray). 0th, 2nd, 4th, ..., or $(2N-2)$th bit of parameter determines which 3:1 multiplexer (upper or lower) is used, which is required in both 3:1 multiplexer and F7MUX. 1st, 3rd, 5th, ..., or $(2N-1)$th bit determines whether output of 3:1 multiplexer or F7MUX to be used.

When organizing a TC-RO from a ring oscillator with N buffers, the number of required 6-input LUTs is $2N + 3$ and the bit width of the parameter becomes $2N + 2$ bits. Since the number of input ports of a 3:1 multiplexer is five, a NAND gate and a 3:1 multiplexer (enclosed by dotted lines) can be packed into a single 6-input LUT. Approximately one additional LUT is required to obtain one bit of parameter in the TC-RO.

C. *Evaluation on Configurability on Oscillation Frequency*

To evaluate the configurability of the TC-RO, we measure the oscillating frequency of a C-RO and a TC-RO with an 8-bit parameter. Based on the considerations in Sections IV.A and IV.B, the number of stages of buffers (N) is set to four in the C-RO and three in the TC-RO. The time to oscillate 50,000 times is measured for each of 2^8 parameters. Four origins, X4Y32, X4Y64, X8Y32, and X8Y64 are examined.

Figure 6 is a box plot on the distribution of oscillating frequency. It was widely distributed in the both oscillators. The probability that a configurable COSO-based TRNG [11] with a specific set of parameters gives proper t_d can be estimated from these data, by choosing a pair of oscillators and their parameters randomly and calculating their t_d. According to our estimation, the probability that t_d became more than 74 (this threshold came from the previous study [11]) was 2.64% for the C-ROs and 3.36% for the TC-ROs. This means the configurability of the TC-RO is not less than the C-RO.

V. Evaluation of TC-TERO-based TRNG

In this section, we apply the TC-RO (described in Section IV) to the implementation of TERO-based TRNG (proposed in Section III) and evaluate it. A TERO organized from the C-RO and the TC-RO is called C-TERO and TC-TERO, respectively. Since the number of buffers in the branches are $N = 3$ and $M = 5$, the C-TERO has a 16-bit parameter, while the TC-TERO has a 20-bit parameter. We selected the same ten origins as evaluated in Section III. Three different devices (boards) were used for evaluation. The device which had been used in the evaluations in Sections III and IV is denoted as Device A; the additional devices are denoted as Devices B and C.

A. *Configurability on Counter Values*

First, we measure the average value of the counter. For each of 2^{16} or 2^{20} parameters, 4,096 counter values were measured and their sum and the sum of their squares were calculated. Arithmetic mean and standard deviation of the counter value is obtained from them. Whether the counter was saturated (i.e. the value of 255 was obtained) at least once was also obtained.

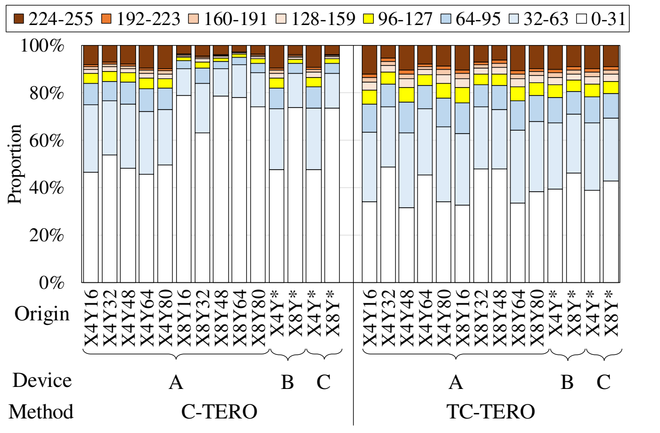

Fig. 7. Distribution of the average counter value of C-TERO and TC-TERO.

Figure 7 plots the distribution of the average counter value. The results of the Devices B and C with the same X coordinates of the origin is summarized in a single bar for each coordinate. In similar to Section III, while the counter value was too large or too small in most cases, a modest value was observed in some cases. According to a preliminary evaluation, we extracted parameters where the average counter value was within a range of 96–127 and the counter was never saturated. Such a counter is likely to be appropriate as a source of entropy. The rate of the extracted parameters was 2.62% in the C-TERO and 5.18% in the TC-TERO, on average. In the C-TERO, the rate of parameters that gave small counter values was increased when the X coordinate was 8, regardless of device. Wires among logic blocks might have affected the characteristics of the ring, as the C-TERO used more logic blocks. Keeping the number of required logic elements small is important not only because it is one of the requirements for TRNGs, but also because it reduces such an effect.

B. Verification with the Stochastic Model of TERO

Next, we verify that the TC-TERO also follows the stochastic model of the TERO [2]. For the rest of the evaluations, we used Device A, the X4Y16 origin, and the parameter where the arithmetic mean of the counter values was 112 and their standard deviation was 7. We obtained the model parameters of TERO from 2^{16} counter values in the way instructed in the literature [2].

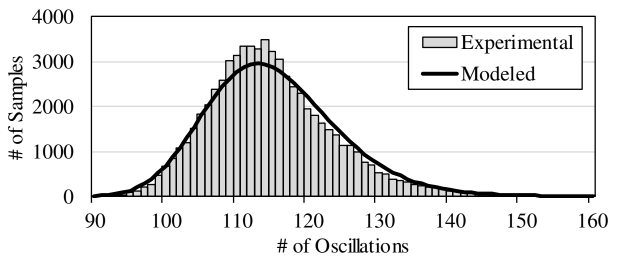

Fig. 8. Distribution of counter values of TC-TERO and the stochastic model.

Fig. 8 compares the modeled distribution to the experimentally obtained one. The model parameters were $R = 1.01908, \sigma_r = 0.00192$, and $\Delta_r = 0.1159$. The outline of the distribution almost agreed, while some errors was observed near the median. The modeling method assumed that the probability that the counter value was not more than the median was 0.5. However, the probability actually was 0.5325 in this experiment. This was a limitation of the model itself. Therefore, the randomness of the TC-TERO can also be justified by the stochastic model of the TERO.

C. Statistical Tests

Finally, we evaluate the quality of random numbers from a TC-TERO-based TRNG. We generated a 1 Gibit of random bit sequence by concatenating LSBs of counter values that were not saturated. The bit rate of generation in this experiment was 1.912 Mbit/s. The number of LUTs and flip-flops used for this TRNG (including the controller) was 40 and 29, respectively.

The generated sequence passed all of the tests of Procedures A and B of AIS-31 [7] without post-processing, even though Procedure A postulated the use of a post-processing method. According to the entropy estimation test, the entropy per one output bit was 0.9993 bit. This means that some bias was observed but not so serious as to be incompliant to AIS-31.

After a simple post-processing where an output sequence of a 4-bit LFSR was XOR-ed, the derived bit sequence passed all of the tests of NIST SP 800-22 test suite [14]. The tests were conducted to 1,073 bitstrings with the parameters recommended in AIS-31 [7]. Without post-processing, some tests failed because of bias. Although the aforementioned hardware amount does not consider post-processing, the additional logic for an LFSR is estimated as almost negligible (a few LUTs and flip-flops).

TABLE II
COMPARISON OF TC-TERO-BASED TRNG WITH OTHER TYPES OF TRNGS.

Type	FPGA Family	Area (LUTs/Regs)	Throughput [Mbit/s]	Statistical Test	Remarks
TC-TERO (This work)	Artix-7	40/29	1.91	AIS-31 Proc. A	
TERO [12]	Spartan-6	39/12	0.63	AIS-31 Test T8	Dependent on Origin
RS Latch [4]	Artix-7	716/974	20.0	NIST SP 800-22	No Statistical Models
Configurable COSO [11]	Spartan-6	108/39*	3.30	AIS-31 Proc. B	
COSO [12]	Spartan-6	18/3	1.22	AIS-31 Test T8	Dependent on Origin
DCM [6]	Virtex-5	19/26	0.21	NIST SP 800-22	Pair of DCMs Required

* Self-calibration circuit is included.

Through the evaluations in this section, we have demonstrated that TC-TERO realized a TRNG that met all of the major requirements. Comparison with other types of TRNGs is summarized in Table II. TC-TERO is one of the strong candidates for TRNGs on Xilinx FPGAs, as well as the configurable COSO [11]. In a practical aspect, it is important that their operating principles are totally different (as explained in Section II), considering a risk that either of them will come out to be compromised in the future.

VI. Conclusion

In this paper, we examined TERO-based TRNGs on Xilinx FPGAs and proposed (1) an implementation methodology that considers the stochastic model and (2) the TC-TERO that achieves both high configurability and small amount of additional hardware.

Our future work includes integration with a mechanism to find an appropriate parameter automatically. A self-calibration method similar to the C-RO [11] might be adopted. It is also important to examine the dependency on operating conditions such as supply voltage and temperature, because some of them can be manipulated by an attacker.

References

[1] F. Bernard, V. Fischer, and B. Valtchanov, "Mathematical Model of Physical RNGs Based on Coherent Sampling," *Tarta Mountains Mathematical Publications*, vol. 45, no. 1, pp. 1–14, 2010.

[2] F. Bernard, P. Haddad, V. Fischer, and J. Nicolai, "From Physical to Stochastic Modeling of a TERO-based TRNG," *Journal of Cryptology*, vol. 32, no. 2, pp. 435–458, 2019.

[3] V. Fischer and M. Drutarovsky, "True Random Number Generator Embedded in Reconfigurable Hardware," in *Proc. 3rd Workshop on Cryptographic Hardware and Embedded Systems*, 2002, pp. 415–430.

[4] N. Fujieda and S. Ichikawa, "A latch-latch composition of metastability-based true random number generator for Xilinx FPGAs," *IEICE Electronics Express*, vol. 15, no. 10, pp. 20 180 386:1–20 180 386:12, 2018.

[5] H. Hata and S. Ichikawa, "FPGA implementation of metastability-based true random number generator," *IEICE Transactions on Information & Systems*, vol. E95-D, no. 2, pp. 426–436, 2012.

[6] A. P. Johnson, R. S. Chakraborty, and D. Mukhopadyay, "An Improved DCM-Based Tunable True Random Number Generator for Xilinx FPGA," *IEEE Transaction on Circuits and Systems II: Express Briefs*, vol. 64, no. 4, pp. 452–456, 2017.

[7] W. Killmann and W. Schindler, *A proposal for: Functionality classes for random number generators, version 2.0*, Federal Office for Information Security, 2011.

[8] P. Kohlbrenner and K. Gaj, "An embedded true random number generator for FPGAs," in *Proc. 12th International Symposium on Field Programmable Gate Arrays*, 2004, pp. 71–78.

[9] Y. Lao, Q. Tang, C. H. Kim, and K. K. Parhi, "Beat Frequency Detector–Based High-Speed True Random Number Generators: Statistical Modeling and Analysis," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 13, no. 1, pp. 1–25, 2016.

[10] M. Majzoobi, F. Koushanfar, and S. Devadas, "FPGA-Based True Random Number Generation Using Circuit Metastability with Adaptive Feedback Control," in *Proc. 13th Workshop on Cryptographic Hardware and Embedded Systems*, 2011, pp. 17–32.

[11] A. Peetermans, V. Rožić, and I. Verbauwhede, "A Highly-Portable True Random Number Generator based on Coherent Sampling," in *Proc. 29th International Conference on Field Programmable Logic and Applications*, 2019, pp. 218–224.

[12] O. Petura, U. Mureddu, N. Bochard, V. Fischer, and L. Bossuet, "A survey of AIS-20/31 compliant TRNG cores suitable for FPGA devices," in *Proc. 26th International Conference on Field Programmable Logic and Applications*, 2016, pp. 1–10.

[13] B. Ray and A. Milenković, "True Random Number Generation Using Read Noise of Flash Memory Cells," *IEEE Transactions on Electron Devices*, vol. 65, no. 3, pp. 963–969, 2018.

[14] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray, and S. Vo, *A statistical test suite for random and pseudorandom number generators for cryptographic applications*, NIST Special Publication 800–22, Rev. 1a, 2010.

[15] M. Varchola and M. Drutarovsky, "New high entropy element for FPGA based true random number generators," in *Proc. 12th Workshop on Cryptographic Hardware and Embedded Systems*, 2010, pp. 351–365.

[16] Xilinx Inc., *7 Series FPGAs Clocking Resources*, User Guide UG472 v1.14, 2018.