A True Random Number Generator on FPGA with Jitter-Sampling by Ring Generator

Tuan-Kiet Dang, Trong-Thuc Hoang, and Cong-Kha Pham
The University of Electro-Communications (UEC), Tokyo 182-8585, Japan
Email: tuankiet@vlsilab.ee.uec.ac.jp, {hoangtt, phamck}@uec.ac.jp

2024 Asian Hardware Oriented Security and Trust Symposium (AsianHOST) | 979-8-3503-6806-2/24/$31.00 ©2024 IEEE | DOI: 10.1109/ASIANHOST63913.2024.10838391

Abstract—**True Random Number Generator (TRNG) is an essential primitive for extracting random bits from a random entropy source. A robust and reliable TRNG design provides non-deterministic nonces for cryptographic applications, especially in the secret key generation process. This paper presents an accumulating-jitter-based TRNG design that uses a ring generator to sample high-frequency oscillators and obfuscate the sampled bit. Implemented on the Xilinx Artix-7 Field-Programmable Gate Array (FPGA), the proposed design occupies a minimal hardware footprint of 6 SLICEs. This implementation demonstrates a good balance between hardware cost and throughput of 200Mbps. Moreover, the randomness of the output random sequences in the default mode and at power-up are of high quality, passing the random tests SP800-22 and SP800-90B from NIST and AIS31 standards from BSI.**

Index Terms—**TRNGs, jitter, metastability, ring generator**

I. Introduction

Random Number Generators (RNG) are versatile security application tools that provide the essential property of randomness derived from physical phenomena. Many cryptographic systems rely on RNGs for tasks such as authentication algorithms and key-generation mechanisms [1]. Moreover, RNGs can be used for circuit simulation software [2] and arbitrary mask in side channel attack countermeasures [3]. RNGs can be classified into three categories: (a) deterministic RNG (DRNG), also known as pseudo-RNG (PRNG); (b) non-physical true RNGs whose noise source does not require dedicated hardware but instead gains entropy from system data (like timing values, memory, etc.) or user interaction (mouse click, keystrokes, etc.); (c) physical True RNG (TRNG), which exploits physical phenomena from dedicated hardware designs or physical experiments, unlike PRNGs, which can support limited secure applications, a high-quality TRNG with sufficient throughput and favorable statistical properties can meet the stringent demands of most cryptographic algorithms.

Dedicated circuits designed to exploit physical processes to provide a source of randomness for TRNGs are called entropy sources. Entropy can be harvested from many sources. For instance, different physical phenomena produce noise, which can be captured as an entropy source. Brederlow *et al.* [4] proposed an entropy source from the low-frequency voltage noise of a MOS device; however, a quality check algorithm must be applied at the output bit. Teranipoor *et al.* [5] applied another noise source from the power supply with a dynamic voltage feedback tuning circuit to regulate and adjust the voltage levels from multiple power sources. Aside from physical noise, the metastable events of memory cells like flip-flops (FF) and latches have been captured to generate random numbers. A latch-based design from [6] presents a lightweight implementation; however, its throughput is somewhat limited. Also, a limitation of metastability-based TRNG is the stringent implementation of manual routing and symmetric layout to ensure minimum bias effect on the metastable output [7]. Finally, the jitter-based TRNG is an approach to generate random numbers that leverages the inherent unpredictability of timing variations caused by noise or manufacturing imperfections. The entropy from jitter can be digitized by sampling a high-frequency oscillation with a lower-frequency clock. Ronaldo *et al.* proposed extracting jitter through frequency collapse behavior of a multimodal ring oscillator (RO) [8], [9]. Although the throughput of the design is limited, the design was evaluated on both ASIC and FPGA platforms, which is a highlight of this design. This paper shares the same goal of a versatile design on both hardware platforms. For every system to be implemented on ASIC, a prototype on FPGA is required to verify its comprehensive functionality before shipping out for fabrication. A TRNG design that can hold the above requirements and offer portability between ASIC and FPGA platforms can significantly reduce the time cost in the development process. In addition to the unpredictability and versatility, a modern TRNG circuit design is required to be lightweight, robust, and easily implemented with digital components [10]. This work presents a jitter-based TRNG design exploiting the oscillation mechanism of a latch-xor cell [11] whose output is sampled by a ring generator. The ring generator also acts as an obfuscation layer to the digitized bit, increasing the output sequence's randomness. The proposed architecture is implemented on the AMD Artix-7 FPGA and measured thoroughly with a restart test and official random tests to validate its random properties.

The remainder of this paper is organized as follows. Section II presents the background of the ring generator as an obfuscation layer to the proposed design. Section III describes the combined architecture of the latch-xor cell and the ring generation. Section IV shows the randomness evaluation of the proposed design. Finally, section V concludes the paper.

II. Preliminary Background

Linear Feedback Shift Registers (LFSR) have been the most popular devices for generating and handling pseudo-random sequences. An LFSR consists of a chain n memory element (typically D flip-flop) where the output of a certain flip-flop is fed back into the input of others in a specific pattern. The pattern is represented by a characteristic polynomial $x^n + h_{n-1}x^{n-1} + ... + h_1x + h_0$. If $h_k = 1$, a feedback connection (also called a tap) encompasses this flip-flop by exclusive-or gates. Depending on the implementation of the XOR gates structure, there are two types of LFSR: (a) Fibonacci LFSR and (b) Galois LFSR. The pseudo-random number sequences depend on the polynomial and the initial state of the memory elements. A sequence generated by an n-bit LFSR has a certain period before repeating itself. The characteristic polynomial determines the length of the period with the maximum of $2^n - 1$.

The ring generator is a special form of a LFSR. The structure of an LFSR can be converted to a ring generator form by applying several transformation steps [12]. The ring generator shares some properties with the LFSR, especially in terms of the periodicity of the output sequence. The major differences lie in the modular design and low internal fan-out of the feedback paths, which enhances the operating frequency of the ring generator. Exploiting this characteristic will yield a high throughput scaling proportionally with the operating frequency.

The ring generator is a structurally optimized version of LFSRs, so both circuits share similar functionality. A synchronous ring generator has a deterministic behavior, which renders a predictable bit sequence as any other pseudorandom sequence generated by an LFSR. In other words, with a fixed initial state, the ring generator will generate a determined bit sequence. Therefore, alternating the internal state of an on-duty ring generator will yield an unpredictable output sequence. The design in [10] applied a conventional free-running RO made of a single NAND gate and m-stage inverters. The ring generator samples multiple stages of the RO to populate relatively long intervals of timing jitter into the ring generator. The jitter sampling mechanism forces at least one noisy edge to get sampled; hence, the internal state of the ring generator changes while it is running. The metastability of the flip-flops is another side effect that contributes to the randomness.

The design from [10] was evaluated with different lengths of the ring generator, ranging from 32 to 256 states. However, the given random test results only show the performance of the ring generator having 48 states and above. Therefore, we have investigated and evaluated the design with a smaller size. The design cannot hold its randomness property as the ring size decreases to 16 states. Several factors can affect the performance of a smaller design. Firstly, a smaller size of the ring generator reduces the number of cycles before the ring repeats its internal state transition. The desired noisy edge might not be sampled during the shortened looping cycle, causing the output bit sequence to be similar at some intervals. Secondly, increasing the number of injected signals from the RO reduces the probability of sampling noisy edges due to the decline of RO frequency. To resolve the problems, we proposed sampling multiple independent oscillation sources rather than multiple stages of a single RO. The independent oscillators can achieve higher frequency than a long RO, which benefits the level of randomness since the entropy depends heavily on the mean frequency separation of the oscillation sources and the sampler [13], [14]. Moreover, metastability is a desired effect of TRNG design; increasing the probability of this phenomenon is crucial to the performance of the TRNG. The metastability Mean Time Between Failures (MTBF) indicates the metastability failures can be expressed by the following formula [15]:

$$MTBF = \frac{e^{t_{MET}/C_2}}{C_1 \times f_{SAMPLER} \times f_{SOURCE}} \tag{1}$$

A high MTBF value reduces the chance of metastability problems on the device. With respect to the TRNG, a low MTBF value is desirable. This can be achieved by increasing the frequency of the ring generator and the independent oscillation sources.

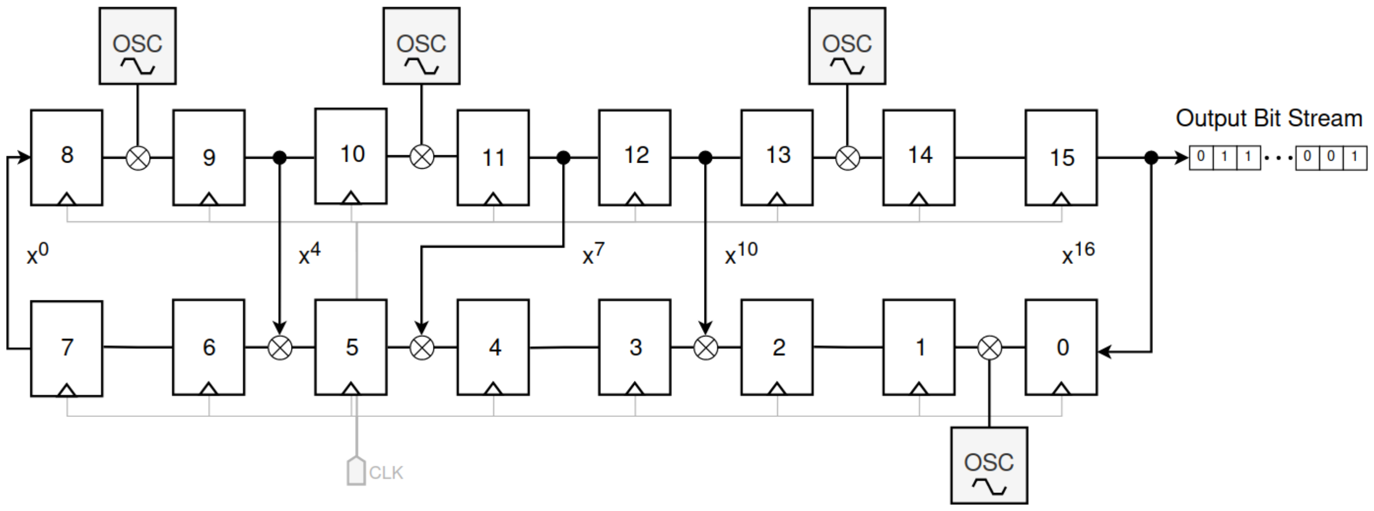

Fig. 1. 16-state ring generator with four oscillator injection points.

III. Proposed TRNG Design

A. Design Overview

To validate the proposed TRNG scheme, we implemented an enhanced architecture with a 16-state ring generator with four independent oscillation sources, as shown in Fig. 1. The oscillators are fed into the ring generator via XOR gates. Applying independent oscillators allows more oscillators to be injected into the ring. The maximum oscillator that can be added is limited by the characteristic polynomial of the ring generator, where the feedback paths occupy between two adjacent flip-flops. Moreover, the randomness of a TRNG design might degrade over time or vary due to different platforms. A general design can use the maximum number of oscillators with the option to deactivate and activate the oscillators, allowing tuning of the random entropy for a balance in performance and power consumption. Furthermore, the position of the design on the FPGA also affects the randomness. The proposed design is placed in the middle of the FPGA chip, surrounded by logic gates of a measurement system. The design in Fig. 1 was tested with the maximum number of oscillators, then, tuning down to four to reduce the hardware costs. The polynomial selection is a significant aspect of designing the proposed TRNG. The polynomials that enable the maximum state transitions of a ring generator before its internal state repeats are called primitive polynomials. The maximum period increases the probability of sampling noisy edges to the ring generator. Besides, the number of coefficients in a polynomial indicates the number of feedback paths. The lower the number of coefficients in a polynomial, the more possible injection points for independent oscillators. With these criteria, we selected the primitive polynomial with five coefficients from [16]. The selected polynomial $P = x^{16} + x^{10} + x^7 + x^4 + x^0$ requires only three feedback paths for the three middle terms, since the paths for x^{16} and x^0 are already used for the creation of the ring.

B. Controlling oscillation source

The oscillation source adapts a cross-coupled XOR and AND gates design, which has been applied as a Physical Unclonable Function (PUF) cell [11] and also utilized as an entropy cell in LX-TRNG structure [17]. In this TRNG design, we adjusted the cell to utilize it as an oscillation source and called it an OSC cell, as shown in Fig. 2. The configuration of the OSC cell has a trigger T, two inputs, I1 and I2, and an output oscillation OSC. The cell exhibits different behaviors based on the triggering sequence of the input signals T, I1 and I2. The principle operation of the OSC cell of the TRNG structure in [17] is the jitter accumulation of the cell's internal oscillation. By terminating the oscillation, a stable or metastable output is captured as a random bit. This technique requires a controller to retrigger the cell to sample one single bit. As a result, the throughput of this controlling mechanism reduces the throughput of the TRNG design. On the contrary, instead of terminating the internal oscillation of the OSC cell, the proposed design utilizes the free running oscillation behavior of the OSC cell. Hence, the excitation sequence to initiate the oscillation of the OSC cell is as follows:

- To reset the OSC cell: set T = 0, I1 = 0 and I2 = 0.
- To arm the oscillation: set I1 = 1 and I2 = 0 or vice versa.
- To initiate the oscillation: set T = 1.

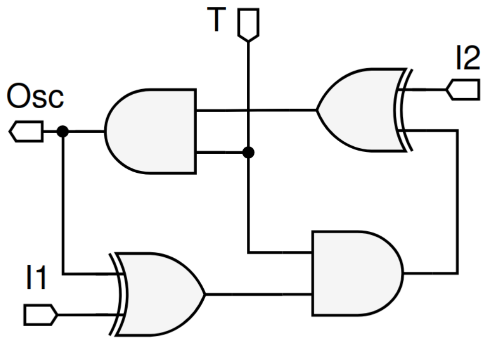

Fig. 2. Independent Oscillation Source from latch cell

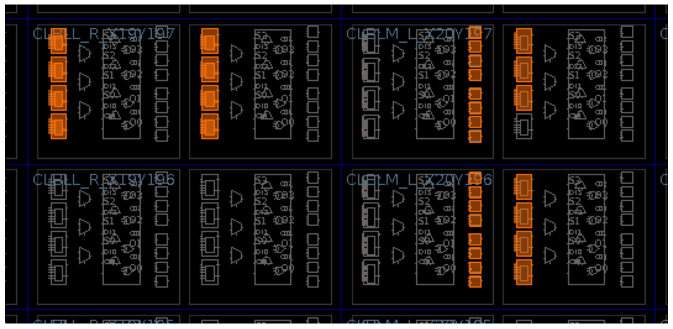

Fig. 3. Placement on Xilinx Artix-7 FPGA

The oscillation frequency depends on the delay t_d of each gate and the total routing delay t_r from LUTs to the switch matrix, thus:

$$OSC_{frequency} = \frac{1}{t_{d_{AND1}} + t_{d_{OR1}} + t_{d_{AND2}} + t_{d_{OR2}} + t_r} \tag{2}$$

C. FPGA Implementation

The proposed TRNG was implemented on Xilinx Artix-7 FPGA. Fig. 3 shows the placement of the TRNG on the FPGA. One OSC cell can fit in two LUT-6s of a SLICE using LUT5 macros four times for the AND and XOR gates. The intra-slice routing of the OSC cell is more relaxed compared to the latch-xor cell [11] and the LX-TRNG [17] because the slight variation in routing delay has minor effects on the probability of sampling the noisy edges. As a result, four OSC cells fit nicely in two top-left SLICEs. Other logic gates for the ring generator and the injection XOR are placed on four SLICE on the right. The total cost of the proposed TRNG on Artix-7 is 15LUTs and 16FFs, resulting in 6 SLICEs. It is worth notice that if the intra-slice connection of the OSC cell is carefully routed, the proposed design can also exploit the weak PUF behavior by utilizing the ring generator as a shift register to output the OSC cell's PUF bit. In this case, a maximum number of OSC cell can be injected to the ring generator for the PUF mode, and enabling a few OSC cells for the TRNG mode. However, the scope of this paper focuses on the TRNG design rather than the unified PUF and TRNG structure.

IV. Performance Evaluation

A. *Startup Validation of The Proposed TRNG*

The initial validation of the random source measures some statistical properties of the proposed TRNG design after start-up due to the possibility of generating correlated sequences. This initial check is essential and has been applied in recent works [8], [10], [18]. To gather the dataset, the first 32-bit random number produced by the TRNG is recorded, and then the circuit is reset to return the state of the ring generator to zero. The process is repeated until 50Mbit of the generated random sequence is collected. Later, this dataset is used for more comprehensive tests with official standard tests.

The random number is collected in multiple 32-bit samples. In the first simple test, we verify the probability of 1s and 0s in each bit position of all samples. Fig. 4 illustrates the proportion of 1s at successive bit-position where the expected value is 50 percent. Additionally, the Hamming weight of all samples in Fig. 5 should be distributed binomially to demonstrate the independence of each generated bit. Another common test for random numbers is autocorrelation [7], which is used to analyze the correlation between a variable and its lag values. A correlation coefficient less than 0.3 indicates the lack of relevance between a variable and its past values [19], [20]. The test result of 1Mbit power-up samples is shown in Fig. 6. The autocorrelation factor (ACF) is examined with the lag value set from 1 to 100, and the resulting coefficient fluctuates around zero with a 95% confidence boundary of approximately ±0.0001. The low ACF value suggests that the correlation analysis attack is ineffective against the proposed TRNG after power-up. Finally, Fig. 7 shows the spatial distribution of 1Mbit (1000×1000) from the start-up dataset. The figure plots a black pixel representing bit-1 and a white pixel as bit-0.

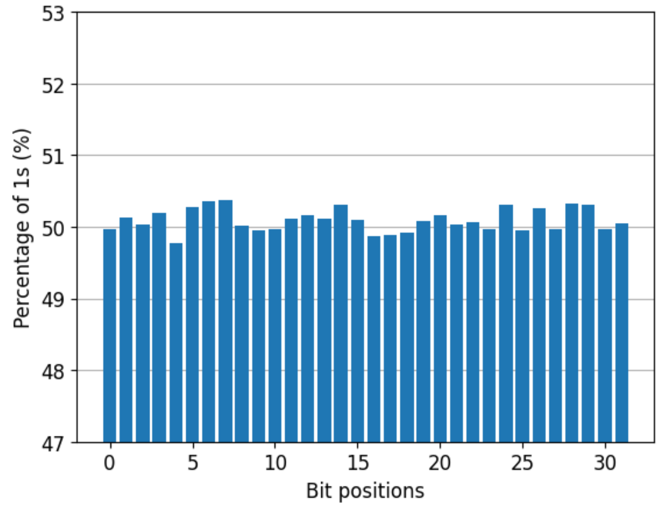

Fig. 4. Proportion of 1s and 0s on successive bit-position in 32-bit samples

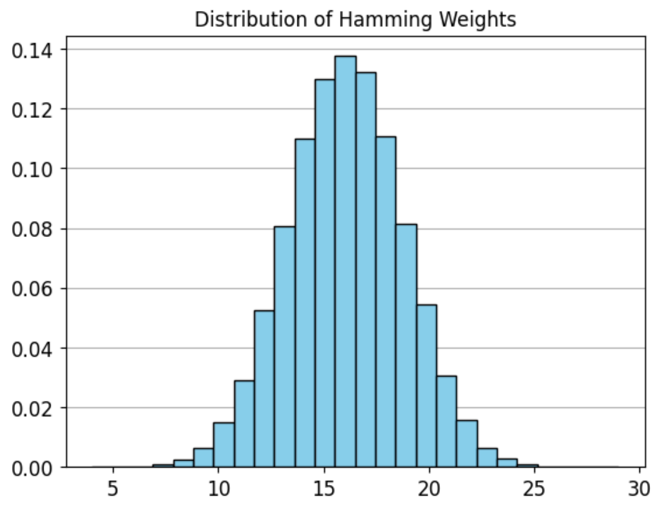

Fig. 5. Distribution of 32-bit samples w.r.t to their Hamming weights.

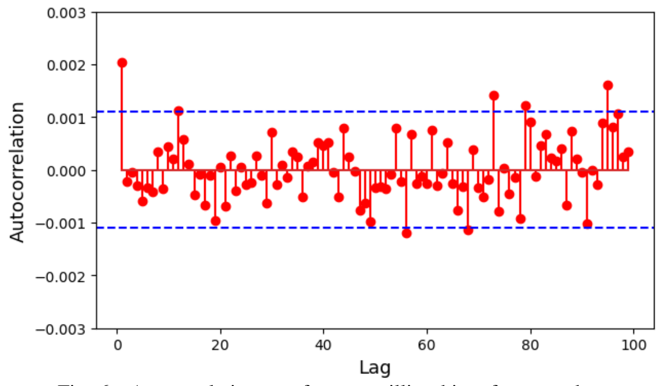

Fig. 6. Autocorelation test for one million bits of startup data

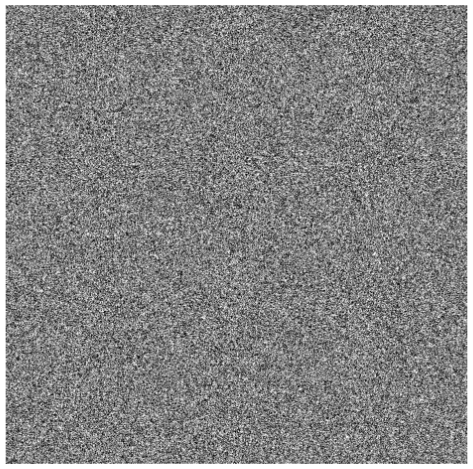

Fig. 7. Spatial distribution of 1Mbit (1000x1000) raw startup data

B. *Comprehensive Random Tests*

1) NIST SP800-22: The NIST SP800-22, or Special Publication 800-22, is a document published by the National Institute of Standards and Technology (NIST) in the United States. It outlines a suite of 15 statistical tests for evaluating

TRNG and PRNG. The test returns a p-value, an indicator of the randomness of a sequence. A sequence with a p-value greater or equal to 0.01 is a good random sequence. In the normal operation mode of the proposed TRNG, we collected a sequence of 50 million random bits. Then, we evaluated both datasets, the newly collected dataset, and the start-up dataset, using the NIST SP800-22 test. The test results are shown in Table I, which demonstrates that the proposed TRNG passes the SP800-22 test.

2) AIS-31: The AIS31 test [21] is also an evaluation guideline for RNGs for cryptographic applications. Proposed by the German Federal Office for Information Security (BSI), this test has been effective in the German certification scheme. BSI and NIST share the same target: harmonizing the SP800-90 and AIS test suites. The AIS31 comprises nine distinct tests denoted as T0-T8. Table III shows the results of the AIS31 test with collected data sequences from start-up and default mode. The T0 (disjointness test) indicates that the random sequence does not contain overlapping segments. Hence, the probability of sampling noisy edges is high enough to prevent the 16-state ring generator from repeating its state cycle. Statistical tests T1-T5 evaluate 257 segments of the data sample. As can be seen, the result passes with a 100% rate for such tests. Tests T6-T8 aim to evaluate the entropy level of the random source. The collected data sequences also pass all these tests by checking the criteria of pass conditions for T6-T8 tests in the second column.

3) NIST SP800-90B: Another recent test suite proposed by NIST is SP800-90B, which can ensure that an entropy source is suitable for secure cryptographic operations. Unlike the former suite, this set of tests focuses on finding the minimum entropy of the random source, a crucial factor for determining the strength and reliability of cryptographic keys generated from the source. This assessment comprises two main tests: the Independent and Identically Distributed (IID) test and the Non-Independent and Non-Identically Distributed (non-IID) test. If the data passes the IID test, it can be used with confidence in cryptographic applications under the assumption of independence and identical distribution. The result analysis of the IID test involves comparing the test statistics to critical values from relevant distributions (e.g., chi-square distribution) to determine if deviations from IID behavior are statistically significant. As shown in Table II, the entropy source of the proposed TRNG is of high quality, and the estimated entropy proves that the proposed TRNG is suitable for cryptographic algorithms.

4) Performance Comparison: Table IV compares the performance of the proposed TRNG with recent designs implemented on FPGA platforms. By exploiting both jitter and metastability phenomena, the proposed design can achieve higher throughput compared to a metastable-based structure LRO [6] with 0.76Mbps. Compared with other jitter-based TRNGs like TROT [22] and mux-based oscillator [23], the proposed structure achieves higher throughput with low hardware overhead. The jitter-accumulate MRCO [7] converges four oscillators to generate random bits at higher throughput. However, it is noticeable that the proposed design has been evaluated with more comprehensive tests, including the restart test.

TABLE I
NIST SP800-22 RANDOM TESTS

Test	Default		Startup	
	Rate	P-value	Rate	P-value
Frequency	100/100	0.035	96/100	0.350
Block Frequency	99/100	0.145	98/100	0.964
Cumulative Sums (2)	97/100	0.035	96/100	0.018
Run	80/100	0.911	100/100	0.044
Longest Run	99/100	0.991	96/100	0.055
Rank	100/100	0.067	100/100	0.955
FFT	98/100	0.759	100/100	0.946
Non-overlap template (148)*	pass	0.384	pass	0.025
Overlapping template	99/100	0.108	99/100	0.455
Maurer's Universal	98/100	0.102	100/100	0.616
Approx. Entropy	100/100	0.798	100/100	0.798
Random Excursions (8)*	pass	0.178	pass	0.091
Random Excursions Variant (18)*	pass	0.028	pass	0.048
Serial (2)	98/100	0.759	100/100	0.081
Linear Complexity	100/100	0.067	100/100	0.066

* Smallest p-values are presented.

TABLE II
SP800-90B IID TEST

Test	Default	Startup
IID Permutation	pass	pass
Chi-square Independence	pass p-value=0.177005	pass p-value = 0.013022
Chi-square goodness of fit	pass p-value=0.111278	pass p-value = 0.602582
Length of the longest repeated substring	pass	pass
Restart	pass	pass
Min. entropy per byte	7.941454	7.949093

TABLE III
AIS-31 RANDOM TEST

Test		Default	Startup
		Pass Rate	
T0	Disjointness	pass	pass
T1	Monobit	257/257	257/257
T2	Poker	257/257	257/257
T3	Run	257/257	257/257
T4	Long Run	257/257	257/257
T5	Autocorrelation	257/257	257/257
		Test Values	
T6-a	Uniform distr. (S<0.025)	0.00309	0.00141
T6-b	Uniform distr. (S<0.025)	0.00353	0.00267
T7-a	Comparative multinomial width = 3 (S <15.13)	0.38642 2.04802	0.04050 2.54899
T7-b	Comparative multinomial width = 4 (S<15.13)	0.42632 0.04050 0.06050 3.97832	0.16928 0.09248 0.21632 0.35378
T8	Entropy (S>7.976)	7.99844	7.99938

TABLE IV
COMPARISON WITH OTHER STATE-OF-THE-ART TRNG DESIGNS ON FPGA

Entropy Source	**FPGA Family**	**Resource Cost**	**Throughput [Mbps]**	**Throughput/Slice [Mbps/Slice]**	**Post-processing**	**Restart Test**	**SP800-22**	**SP800-90B**	**AIS-31**
DD-cell [18]	Artix-7	64 SLICEs	225	3.5	no	pass	pass	pass	-
TROT [22]	Artix-7	32LUTs/55FFs 17CARRY4	25	3.125	yes	-	-	pass	pass
MRCO [7]	Spartan-6 Virtex-6	13LUTs/4FFs	350 500	107.69 153.85	no	-	pass	pass	-
RO [23]	Virtex-6/7 Artix-7	1LUT/4FFs 4MUXs/1PLL	100	-	no	-	pass	pass	-
LRO [6]	Spartan-6	4LUTs/3FFs	0.76	0.76	no	-	pass	-	-
LX-TRNG	Spartan-6	36 LUTs	12.5	1.38	yes	-	pass	pass	-
This work	Artix-7	15LUTs/16FFs	200	33.3	no	pass	pass	pass	pass

V. CONCLUSION

This article presents a TRNG based on the ring generator architecture, which acts as a jitter sampler and an obfuscation layer to produce pure randomized sequences. This design leverages the benefits of both the timing jitter of multiple independent oscillation sources and the effect of metastability of flip-flops to create an effective entropy source. The proposed scheme is evaluated with a design combining a 16-state ring generator and four oscillators adapted from a latch-xor cell. The experimental results on FPGA have shown that the proposed design effectively enhances the occurrence probability of jitter and metastability phenomena to the ring generator. Hence, the output sequences measured at start-up and default operation of the TRNG show good randomness in all official random tests from NIST and BSI. The hardware cost on Xilinx Artix-7 is merely 6 SLICEs with 15 LUTs and 16 FFs. The throughput of the TRNG is measured at 200Mpbs, which can be scaled up thanks to the high-speed ring generator architecture. Compared with the performance of state-of-the-art designs, the proposed TRNG achieves a high throughput with low hardware overhead.

REFERENCES

[1] T.-K. Dang, K.-D. Nguyen, B. K.-D.-Nguyen, T.-T. Hoang, and C.-K. Pham, "Realization of Authenticated One-Pass Key Establishment on RISC-V Micro-Controller for IoT Applications," *Future Internet*, vol. 16, no. 5, pp. 1–17, May 2024.

[2] B. Dang, et al., "Physically Transient True Random Number Generators Based on Paired Threshold Switches Enabling Monte Carlo Method Applications," *IEEE Electron Device Letters*, vol. 40, no. 7, pp. 1096–1099, May 2019.

[3] T.-H. Tran, *et al.*, "An Efficient Hiding Countermeasure with Xilinx MMCM Primitive in Spread Mode," in *IEEE Int. Symp. on Circ. and Syst. (ISCAS)*, May 2024, pp. 1–5.

[4] R. Brederlow, R. Prakash, C. Paulus, and R. Thewes, "A Low-power True Random Number Generator Using Random Telegraph Noise of Single Oxide-traps," in *IEEE Int. Solid-State Circ. Conf.*, Feb. 2006, pp. 1666–1675.

[5] F. Tehranipoor, P. Wortman, N. Karimian, W. Yan, and J. A. Chandy, "DVFT: A Lightweight Solution for Power-Supply Noise-Based TRNG Using Dynamic Voltage Feedback Tuning System," *IEEE Trans. on Very Large Scale Integration (VLSI) Syst.*, vol. 26, no. 6, pp. 1084–1097, Mar. 2018.

[6] R. D. Sala, D. Bellizia, and G. Scotti, "A Novel Ultra-Compact FPGA-Compatible TRNG Architecture Exploiting Latched Ring Oscillators," *IEEE Trans. on Circ. and Syst. II: Express Briefs (TCAS-II)*, vol. 69, no. 3, pp. 1672–1676, Oct. 2021.

[7] T. Ni, Q. Peng, J. Bian, L. Yao, Z. Huang, A. Yan, and X. Wen, "MRCO: A Multi-ring Convergence Oscillator-based High-Efficiency True Random Number Generator," in *Asian Hardware Oriented Secu. and Trust Symp. (AsianHOST)*, Dec. 2022, pp. 1–6.

[8] R. Serrano, *et al.*, "A Fully Digital True Random Number Generator With Entropy Source Based in Frequency Collapse," *IEEE Access*, vol. 9, pp. 105 748–105 755, Jul. 2021.

[9] ——, "A Robust and Healthy Against PVT Variations TRNG Based on Frequency Collapse," *IEEE Access*, vol. 10, pp. 41 852–41 862, Apr. 2022.

[10] J. Rajski, M. Trawka, J. Tyszer, and B. Włodarczak, "A Lightweight True Random Number Generator for Root of Trust Applications," *IEEE Trans. on Comp.-Aided Design of Integrated Circ. and Syst.*, vol. 42, no. 9, pp. 2815–2825, Jan. 2023.

[11] R. D. Sala, D. Bellizia, and G. Scotti, "A Lightweight FPGA Compatible Weak-PUF Primitive Based on XOR Gates," *IEEE Trans. on Circ. and Syst. II: Express Briefs (TCAS-II)*, vol. 69, no. 6, pp. 2972–2976, Mar. 2022.

[12] G. Mrugalski, J. Rajski, and J. Tyszer, "High-speed Ring Generators and Compactors of Test Data," in *VLSI Test Symp.*, May 2003, pp. 57–62.

[13] C. S. Petrie and J. A. Connelly, "Modeling and Simulation of Oscillator-based Random Number Generators," in *IEEE Int. Symp. on Circ. and Syst.*, vol. 4, May 1996, pp. 324–327.

[14] D. Liu, Z. Liu, L. Li, and X. Zou, "A Low-Cost Low-Power Ring Oscillator-Based Truly Random Number Generator for Encryption on Smart Cards," *IEEE Trans. on Circ. and Syst. II: Express Briefs (TCAS-II)*, vol. 63, no. 6, pp. 608–612, Feb. 2016.

[15] Altera, "Understanding Metastability in FPGAs," Jul. 2009. [Online]. Available: https://cdrdv2-public.intel.com/650346/wp-01082-quartus-ii-metastability.pdf

[16] J. Rakski and J. Tyszer, "Primitive Polynomials Over GF(2) of Degree up to 660 with Uniformly Distributed Coefficients," *J. of Elec. Testing*, vol. 19, p. 645–657, Dec. 2003.

[17] R. Della Sala, D. Bellizia, and G. Scotti, "High-throughput fpga-compatible trng architecture exploiting multistimuli metastable cells," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 69, no. 12, pp. 4886–4897, 2022.

[18] R. D. Sala and G. Scotti, "Exploiting the DD-Cell as an Ultra-Compact Entropy Source for an FPGA-Based Re-Configurable PUF-TRNG Architecture," *IEEE Access*, vol. 11, pp. 86 178–86 195, Aug. 2023.

[19] X. Wang, H. Liang, Y. Wang, L. Yao, Y. Guo, M. Yi, Z. Huang, H. Qi, and Y. Lu, "High-Throughput Portable True Random Number Generator Based on Jitter-Latch Structure," *IEEE Trans. on Circ. and Syst. I: Regular Papers (TCAS-I)*, vol. 68, no. 2, pp. 741–750, Nov. 2020.

[20] Y. Cao, C.-H. Chang, Y. Zheng, and X. Zhao, "An Energy-efficient True Random Number Generator Based on Current Starved Ring Oscillators," in *Asian Hardware Oriented Secu. and Trust Symp. (AsianHOST)*, Oct. 2017, pp. 37–42.

[21] W. Killmann and W. Schindler, "A Proposal for: Functionality Classes for Random Number Generators," Sep. 2022. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Certification/Interpretations/AIS_31_Functionality_classes_for_random_number_generators_e.html

[22] M. Grujić and I. Verbauwhede, "TROT: A Three-Edge Ring Oscillator Based True Random Number Generator With Time-to-Digital Conversion," *IEEE Trans. on Circ. and Syst. I: Regular Papers (TCAS-I)*, vol. 69, no. 6, pp. 2435–2448, Mar. 2022.

[23] L. Yao, H. Liang, H. Zhang, T. Ni, M. Yi, and Y. Lu, "A Lightweight M_TRNG Design based on MUX Cell Entropy using Multiphase Sampling," in *Asian Hardware Oriented Secu. and Trust Symp. (AsianHOST)*, Dec. 2022, pp. 1–4.