DH-TRNG: A Dynamic Hybrid TRNG with Ultra-High Throughput and Area-Energy Efficiency

Yuan Zhang
Hunan University
Changsha, China
zhangyuanzy@hnu.edu.cn

Kuncai Zhong
Hunan University
Changsha, China
kczhong@hnu.edu.cn

Jiliang Zhang*
Hunan University
Changsha, China
zhangjiliang@hnu.edu.cn

ABSTRACT

As a vital security primitive, the true random number generator (TRNG) is a mandatory component to build roots of trust for any encryption system. However, existing TRNGs suffer from bottlenecks of low throughput and high area-energy consumption. In this work, we propose DH-TRNG, a dynamic hybrid TRNG circuitry architecture with ultra-high throughput and area-energy efficiency. Our DH-TRNG exhibits portability to distinct process FPGAs and passes both NIST and AIS-31 tests without any post-processing. The experiments show it incurs only 8 slices with the highest throughput of 670Mbps and 620Mbps on Xilinx Virtex-6 and Artix-7, respectively. Compared to the state-of-the-art TRNGs, our proposed design has the highest $\frac{\text{Throughput}}{\text{Slices}\cdot\text{Power}}$ with 2.63× increase.

CCS CONCEPTS

• **Hardware → Integrated circuits**; • **Security and privacy → Hardware security implementation**.

KEYWORDS

Hardware Security, True Random Number Generator, Ultra-High Throughput, Area-Energy Efficiency

ACM Reference Format:
Yuan Zhang, Kuncai Zhong, and Jiliang Zhang. 2024. DH-TRNG: A Dynamic Hybrid TRNG with Ultra-High Throughput and Area-Energy Efficiency. In *61st ACM/IEEE Design Automation Conference (DAC '24), June 23–27, 2024, San Francisco, CA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3649329.3656236

1 INTRODUCTION

Serving as a fundamental hardware security primitive, true random number generator (TRNG) is a crucial module in modern information security systems [1]. It extracts unpredictable inherent physical processes and generates completely unforeseeable random bitstreams to build roots of trust [2], which makes it superior to the pseudo-random number generator driven by fixed algorithms in security.

*Corresponding author: Jiliang Zhang.

DAC '24, June 23–27, 2024, San Francisco, CA, USA

ACM ISBN 979-8-4007-0601-1/24/06
https://doi.org/10.1145/3649329.3656236

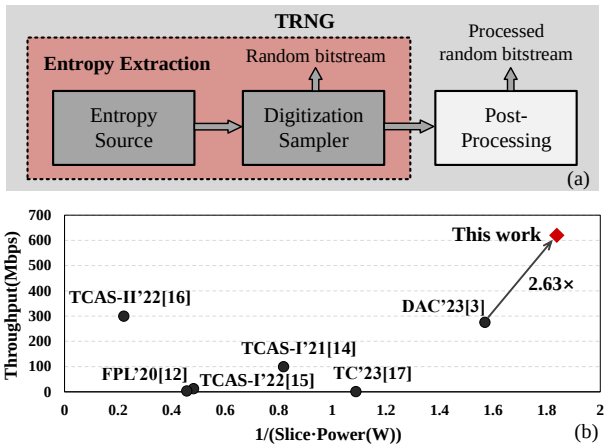


Figure 1: (a) TRNG architecture. (b) Comparison with state-of-the-art TRNGs.

Typically, a TRNG consists of entropy sources, digitization sampler, and post-processing [3], as shown in Figure 1(a). The entropy source generates random events by extracting physical processes, the digitization sampler converts analog quantities into random sequences, and the post-processing transforms raw random numbers into bitstreams that satisfy randomness criteria. For these components, the quality of entropy extraction determines the necessity for post-processing and the performance of TRNG, thus occupying a central position in TRNG designs.

Due to the flexibility and application potential, the Field Programmable Gate Array (FPGA) has been the most prevalent hardware platform for implementing TRNGs. However, the current FPGA-compatible digital TRNGs still suffer from insufficient throughput and area-energy efficiency. The most crucial metric for optimizing TRNGs, denoted as $\frac{\text{Throughput}}{\text{Slices}\cdot\text{Power}}$, presently only attains its maximum value of 432.97 in the recent work [3]. It is inadequate to meet the growing demand for substantial amounts of encrypted data and a high rate of random number generation in emerging scenarios, such as confidential computing, trusted execution environments, blockchain digital signatures, and edge computing.

To address the above issues, we propose DH-TRNG, a dynamic hybrid true random number generator. To effectively improve the throughput, it exploits multiple entropy sources by dynamic switching of loop state to support higher sampling frequencies. Moreover, it utilizes two reinforcement strategies to further enhance output randomness and reduce entropy unit usage while minimizing the incurred area.

Our main contributions are summarized as follows.

- We propose a hybrid entropy unit capable of dynamically quantifying the randomness of both jitter and metastability. We verify the randomness improvement through theoretical derivation and experiments.
- We propose coupling and feedback strategies to further improve output randomness and reduce entropy unit usage. Then, we introduce a multistage sampling array that generates bitstreams capable of passing the statistical tests without any post-processing.
- We optimize the circuitry area to make the proposed TRNG minimized and portable for FPGAs with various processes. Prominently, it exhibits ultra-high $\frac{\text{Throughput}}{\text{Slices}\cdot\text{Power}}$ with promising application prospects. The source code is available at https://github.com/zy-cshnu/DH-TRNG.

The experimental results demonstrate that the proposed TRNG architecture consumes a total of 8 slices, comprising 23 LUTs, 4 MUXs, and 14 DFFs. In Virtex-6 and Artix-7, our proposed design exhibits an ultra-high throughput of 670Mbps and 620Mbps with a low power consumption of 0.126W and 0.068W, respectively. As depicted in Figure 1(b), compared to the state-of-the-art TRNGs, our DH-TRNG has the highest $\frac{\text{Throughput}}{\text{Slices}\cdot\text{Power}}$ with 2.63× prominent increase.

2 BACKGROUND AND RELATED WORK

This section introduces the randomness principle of jitters and metastability, and presents the related work.

2.1 Oscillation Jitter

Jitter is a typical entropy source induced by external random physical processes, such as thermal noise, scattering noise, power supply fluctuation, and environmental variation, in ring oscillators (ROs). It manifests as uncertain phase noise in the frequency domain. Typically, in the jitter extraction process shown in Figure 2(a), a low-frequency clock signal samples the high-frequency oscillation of entropy sources within a flip-flop to generate random bits. The relation between the phase noise and the oscillation ring order is expressed as [5]:

$$L_{min}\{\Delta f\} = \frac{8N}{3\eta}\cdot\frac{KT}{P}\cdot\left(\frac{V_{DD}}{V}+\frac{V_{DD}}{IR}\right)\cdot\left(\frac{f_0}{\Delta f}\right)^2, \quad (1)$$

where K, T, η, V_{DD}, V, I, and R are constants, f_0 is the frequency of the ring, Δf denotes the offset frequency, P is the power consumption, and N is the order of the ring.

Apparently, increasing the order N can amplify phase noise L_{min}. Nevertheless, it will also reduce frequency f_0 and throughput [4]. Thus, many related works focus on the trade-off schemes to improve performance. For instance, Cui et al. [4] provided a multi-stage feedback RO that increased both N and f_0. Lu et al. [3] designed a multiphase sampler TRNG architecture, which increases the throughput with low resource overhead. However, their basic entropy units exhibit inherent randomness insufficiency, which limits the efficiency improvement of generating random sequences.

2.2 Sampling Metastability

In addition to jitter, metastability, which refers to the unpredictable random output of flip-flops, serves as another ideal entropy source

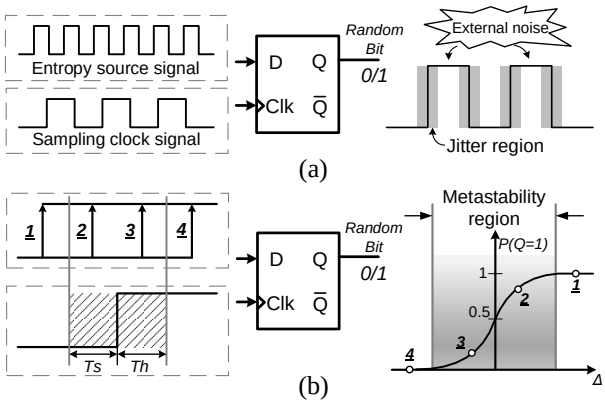


Figure 2: (a) Randomness extraction of jitters. (b) Randomness extraction of metastability.

for true random number generation [6, 7]. As shown in scenarios 2 and 3 of Figure 2(b), when the flip-flop samples unstable intermediate signals within the set/hold time, unpredictable random bits are generated due to timing violations associated with metastability.

Majzoobi et al. [7] proved the probability of output settling onto '1' can be accurately modeled by the Gaussian cumulative distribution function and central limit theorem, as expressed in Equation (2):

$$P(out=1) = Q\left(\frac{\Delta}{\sigma}\right), Q(x) = \frac{1}{\sqrt{2\pi}}\int_x^{\infty} e^{\frac{-u^2}{2}}du, \quad (2)$$

where Δ denotes the time difference between the sampling edge and the moment transition occurs, and σ is proportional to the width of the setup/hold time window.

Based on Equation (2), reducing Δ can force the flip-flop into metastability by ensuring that sampling points fall between points 2 and 3 as much as possible, as depicted in Figure 2(b). Inspired by this, Peng et al. [8] proposed RO-driven shift registers that align the delay of symmetric inputs to produce unpredictable equal probability outputs. Sala et al. [9] proposed latched-XOR cells configured by multiple excitation signals to achieve randomness improvement with low hardware area. Nevertheless, both their throughput is constrained due to limitations in entropy collection rate.

3 THE PROPOSED DYNAMIC HYBRID TRNG

This section presents our proposed design. First, we introduce the proposed dynamic hybrid entropy unit. Then, we present two reinforcement strategies for enhancing randomness. Finally, we provide the overall circuitry architecture and the area optimization.

3.1 Dynamic Hybrid Entropy Unit

As discussed in Section 2.1, the randomness of ROs is affected by a trade-off between their order and oscillation frequency. To explore the optimal solution, we experimentally analyze the randomness of ROs with different orders. We utilize NIST SP800-90B to test the parallel XORed ROs with a 100MHz sampling frequency. As shown in Table 1, 9-stage ROs yield the highest min-entropy. However, their long transmission delays make them cannot meet the throughput improvement target. In order to satisfy both high-frequency

properties and sufficient randomness, we propose a dynamic hybrid entropy unit as shown in Figure 3(a).

Table 1: Randomness test of different order oscillation rings.

Stage number	2	3	4	5	6	7
Min-entropy	0.9737	0.9733	0.9756	0.9776	0.9783	0.9831
Stage number	8	9	10	11	12	13
Min-entropy	0.9860	0.9871	0.9842	0.9837	0.9788	0.9735

For the proposed entropy unit, two independent oscillation rings RO1 and RO2 are enabled by signal *En* to generate high-frequency oscillation signals. Two D flip-flops sample nodes R_1 and R_2, producing the final random output through XOR. Due to the influence of external noise, uncertain jitter regions are generated in the transition edge of R_1, as shown in Figure 3(b). When the clock's positive edge samples this offset signal, the randomness of jitters is quantified, producing an unpredictable output Q_1.

Meanwhile, R_1 serves as the activation signal for the MUX in RO2, causing RO2 to realize random dynamic switching between a holding loop and an inverter loop. When R_1=0, R_2 is in the oscillation region. A high-frequency oscillation signal smoothens the original square wave signal and generates short pulses, which expand the transition moment in the time domain. When R_1=1, R_2 is in the holding region, where a rapid transition signal is randomly locked at an uncertain subthreshold state. Thus, the signal collection of Q_2 greatly amplifies the probability of metastability, and the final dynamic hybrid *Out* exhibits considerable randomness.

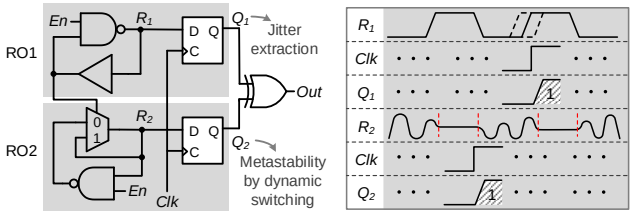


Figure 3: (a) Dynamic hybrid unit structure. (b) Randomness extraction principle.

Next, we present the formal verification of the randomness increase. Ideal random numbers imply an equal generation probability of '0' and '1'. When sampling occurs within the holding region, the random subthreshold signal R_2 causes Δ in Equation (2) to be 0. Hence, the probability of Q_2 nears $\frac{1}{2}$. According to [10], the expected value of *Out* is expressed as:

$$E_{Out} = E(Q_1 \oplus Q_2) = \frac{1}{2} - 2(\mu_1 - \frac{1}{2})(\mu_2 - \frac{1}{2}), \tag{3}$$

where μ_1 and μ_2 are the expected values of Q_1 and Q_2, respectively. In the holding region, R_1 is the definite value 1, and $\mu_2 \approx \frac{1}{2}$. Thus, the expected value is close to $\frac{1}{2}$, exhibiting near-ideal randomness.

At the same time, when sampling takes place within the oscillation region, the n-order XOR expected value of *Out* is calculated by Equation (4):

$$E_{Out_n} = E(Q_1 \oplus Q_2)_n = \frac{1}{2}\left(1 + ((1-2\mu_1)(1-2\mu_2))^{\frac{n}{2}}\right). \tag{4}$$

Given that $\mu_1, \mu_2 \in (0,1)$ implies $1-2\mu_1, 1-2\mu_2 \in (-1,1)$, the expected value will rapidly converge to $\frac{1}{2}$ with an increase in the XOR number. Thus, employing more parallel entropy units can eliminate the bias and improve the randomness.

Finally, we show the randomness coverage of the proposed unit, which is a vital metric for evaluating the entropy source. According to [11], the randomness coverage of our proposed dynamic hybrid entropy unit through multi-level XOR is derived as:

$$P_{rand} = 1 - \prod_{i=1}^{n}\left((1 - \frac{2aw_i}{T_{ro_i}})(1 - (\tau + 2\varepsilon f_i))\right), \tag{5}$$

where n is the number of XOR. a and w_i are the probability and width of jitters in RO1. T_{ro_i} denotes its oscillation period. τ is the probability of sampling subthreshold signals at the holding region in RO2. ε and f_i denote the transition edge width and oscillation frequency at the oscillation region.

As shown in Equation (5), the values of cumulative multiplication are greatly minimized, and the random coverage by multiple XORs is nearly closer to 1, indicating a significant randomness increase of the proposed design. Based on it, we also do some experiments to show the strength. As depicted in Table 2, compared to 9-stage ROs, dynamic hybrid entropy units exhibit higher min-entropy. Thus, our proposed entropy units successfully achieve both increased randomness and high-frequency characteristics.

Table 2: Comparison of entropy units with 9-stage ROs.

XOR number	9	10	11	12	13
Entropy units	0.9765	0.9803	0.9830	0.9836	0.9853
9-stage ROs	0.9705	0.9751	0.9779	0.9801	0.9813
XOR number	14	15	16	17	18
Entropy units	0.9868	0.9885	0.9896	0.9903	0.9912
9-stage ROs	0.9849	0.9871	0.9873	0.9886	0.9891

3.2 Coupling and Feedback Strategies

Increasing the employed entropy units enhances randomness but leads to higher hardware overhead and power consumption. To improve output randomness while reducing the entropy unit usage, we further apply two reinforcement strategies, which are coupling and feedback strategies.

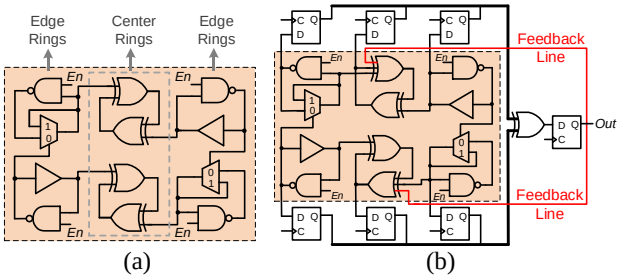


Figure 4: (a) Coupling strategy. (b) Feedback strategy.

In the coupling strategy, two dynamic hybrid entropy units are reversely inserted into two 2-stage XOR rings and create a nested coupling structure, as shown in Figure 4(a). It involves two central rings and four edge rings. Due to the characteristics of XOR

gates, the logic of central rings is variable with inputs. Thus, signals within the central rings are activated by the oscillation signals with random phase noise on both sides. Furthermore, in the central rings, there are independent jitters and overlaps of different frequencies generated by the edge rings in the time domain. This leads to the logical mode of the central ring undergoing disorderly switching, and further causes the gate-level signals within it to exhibit nonperiodic random flips during propagation.

In the feedback strategy, the final output is fed back into central rings to randomize the initial state, as shown in Figure 4(b). For this strategy, multiple D flip-flops sample independent signals generated by various oscillation rings, the sampled random signals are collected by an XOR gate to produce the final output, and an additional flip-flop is introduced to transmit *Out* into the central rings through feedback lines. This will cause random phases to switch in the oscillation signals. Thus, the high-entropy *Out* serves as an activation signal to renewedly randomize the initial state, and further boosts the final output randomness.

The polynomial of a feedback XOR ring is expressed as $f(x) = \sum_{i=1}^{n} x_i^{k(i)}$, where n and $k(i)$ is the stage number of XOR gates and edge rings, respectively. The feedback polynomial of the central ring for achieving multi-ring convergence is $f(x) = x_1^1 + x_2^2 + x_r^{'}$, where x_1^1 and x_2^2 denote signals of edge rings, and $x_r^{'}$ is the feedback of the final random output. Thus, coupling and feedback strategies implement the stacking of random signals and a completely random initial state, greatly improving the chaos of the overall system and enhancing the randomness quality of the final output.

3.3 Overall Architecture and Area Optimization

Based on the aforementioned entropy units and strategies, we propose the DH-TRNG as shown in Figure 5(a). In the entropy source,

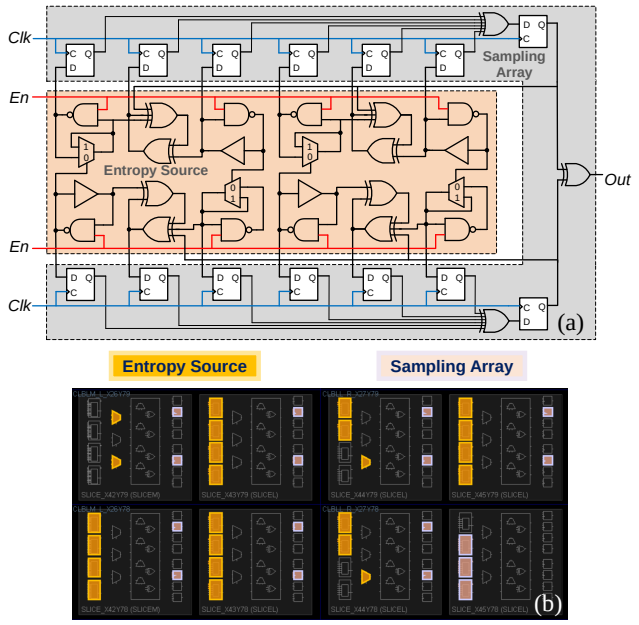


Figure 5: (a) Overall circuitry architecture of DH-TRNG. (b) Implementation with automated placement and routing.

the proposed dynamic hybrid entropy units inserted into central XOR rings form two sets of identical nested coupling structures. In the multistage sampling array, all independent signals generated by the 12 rings are sampled separately through D flip-flops, ultimately generating random output via an XOR tree. Meanwhile, the final output is fed back to the input of the XOR rings in the entropy source to randomize the initial state. The entire circuitry is enabled by the signal *En*, and the *Clk* is provided through the built-in Phase Locked Loop (PLL) in the FPGA. One bit true random number is produced at each clock cycle. Moreover, consider that our DH-TRNG can provide sufficient randomness support as discussed above. It allows for automatic layout and routing in FPGA without any manual intervention and performance loss. Hence, the proposed design also has high portability.

To reduce the area, we further optimize the implementation of DH-TRNG. As the basic programmable unit of FPGA, one slice in Xilinx 6 serials or 7 serials FPGA contains four six-input LUTs, three MUXs, eight DFFs, and other arithmetic logic. All gate circuits can be implemented by configuring the LUT to corresponding logical functions. We constrain all gate cells by type to an appropriate position in a compact square slice array, as shown in Figure 5(b). Given the coordinates of the origin slice, only 8 slices are consumed, including 20 LUTs and 4 MUXs for entropy source structure, 14 DFFs and 3 LUTs for the sampling array. Moreover, the placement of the same type of gates and each slice can be flexibly adjusted. Thus, the area of our proposed DH-TRNG is greatly optimized.

4 EXPERIMENT AND RESULTS

We implement the proposed DH-TRNG on two different process FPGAs to verify its portability, which are Xilinx Virtex-6 FPGA (xc6vlx240t with 45 nm process) and Xilinx Artix-7 FPGA (xc7a100t with 28 nm process). To evaluate its resistance to temperature and voltage variations, we set up an experimental platform illustrated in Figure 6, consisting of a temperature chamber, a DC power supply, a computer, and two FPGA boards. The performance of DH-TRNG is measured by a series of tests on the collected data, such as NIST test, AIS-31 test, deviation test, autocorrelation test, restart test, and PVT test. Finally, we conduct a comprehensive comparison of our proposed design with the state-of-the-art FPGA-based TRNG architectures.

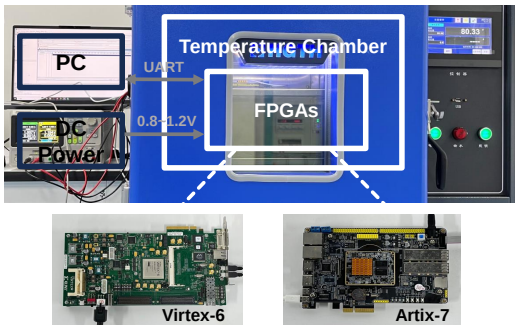


Figure 6: Experimental platform.

4.1 Randomness Test

We perform three kinds of randomness tests on DH-TRNG.

4.1.1 NIST SP 800-22 Test. It is a widely used test suite to verify the randomness of the random sequence, where the *P-value* exceeding 0.01 indicates the sequences are approximately uniformly distributed. The *Prop.* denotes the pass probability of the test item. We collect 30 sets of 1Mbit random numbers for each FPGA in a standard environment. As shown in Table 3, the test results illustrate that our proposed DH-TRNG passes all test items and exhibits good randomness.

Table 3: Results of NIST SP 800-22 Test.

NIST SP 800-22	Virtex-6		Artix-7	
	P-value	*Prop.*	*P-value*	*Prop.*
Frequency	0.739918	30/30	0.739918	30/30
BlockFrequency	0.100508	29/30	0.407091	29/30
CumulativeSums*	0.180952	30/30	0.462665	30/30
Runs	0.468595	30/30	0.178278	29/30
LongestRun	0.122325	30/30	0.213309	29/30
Rank	0.350485	30/30	0.350485	30/30
FFT	0.739918	30/30	0.468595	30/30
NonOverlappingTemplate*	0.472949	30/30	0.477819	30/30
OverlappingTemplate	0.671779	30/30	0.534146	30/30
Universal	0.350485	30/30	0.299251	29/30
ApproximateEntropy	0.602458	30/30	0.804337	30/30
RandomExcursions*	0.090867	17/17	0.029136	17/17
RandomExcursionsVariant*	0.084577	17/17	0.043234	17/17
Serial*	0.390368	30/30	0.844760	30/30
LinearComplexity	0.178278	29/30	0.407091	30/30

* The p-value is the average of the p-values of all subtests.

4.1.2 NIST SP 800-90B Test. This test focuses on testing the null hypotheses of a digital sequence and employs a distinct method for estimating the statistical distribution and min-entropy of the random sequences. We collect 30 sets of 1Mbit random numbers for each FPGA in a standard environment. The NIST SP 800-90B non-IID test results are shown in Table 4, and the min-entropy of the IID test is 0.994698 (Virtex-6) and 0.995966 (Atrix-7), respectively, demonstrating superior randomness.

Table 4: Results of NIST SP 800-90B Test.

NIST SP 800-90B	Virtex-6		Aritx-7	
	p-max	*h-min*	*p-max*	*h-min*
MCV	0.501841	0.994698	0.501400	0.995966
Collision	0.527344	0.923184	0.521484	0.939304
Markov	4.28E-39	0.995748	3.64E-39	0.997594
Compression	0.5	1	0.5	1
t-Tuple	0.519390	0.945111	0.529343	0.917726
LRS	0.519355	0.945206	0.502963	0.991475
Multi-MCW	0.501042	0.998657	0.501141	0.996713
Lag	0.500465	0.998567	0.501683	0.995153
Multi-MMC	0.500630	0.998183	0.500566	0.998368
LZ78Y	0.501705	0.99509	0.501028	0.997038

* The p-value is the average of the p-values of all subtests.

4.1.3 AIS-31 Test. AIS-31 is another test suit to evaluate the quality of random numbers. This test consists of nine sub-test items from T0 to T8. We collect 7,200,000 bits of random numbers for each FPGA. As shown in Table 5, the test results of AIS-31 exhibit that the random sequences generated by our DH-TRNG pass all test items.

Table 5: Results of AIS-31 Test.

AIS-31	Virtex-6	Artix-7
Disjointness Test (T0)	Pass	Pass
Monobit Tests (T1)*	100%	100%
Poker Tests (T2)*	100%	100%
Run Tests (T3)*	100%	100%
Long Run Test (T4)*	100%	100%
Autocorrelation Test (T5)*	100%	100%
Uniform Distribution Test (T6)	Pass	Pass
Multinomial Distributions (T7)	Pass	Pass
Entropy Test (T8)	Pass	Pass

* The pass rate of this test item.

4.2 Restart Test

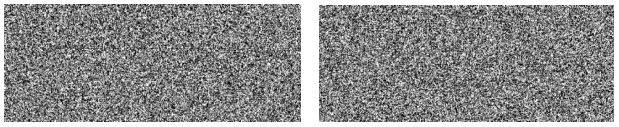
In the restart test, we enable the proposed DH-TRNG and sample the first 32-bit random data six times at the same condition. The obtained data are 0X8E8F7BE6, 0XD448223A, 0X2ED82918, 0X79DA4E4B, 0X51A602A9, and 0XDB9E49EC, respectively. The results show that all sets of sequences are different, which proves that our TRNG is unrepeatable and has true randomness characteristics.

4.3 Deviation Test

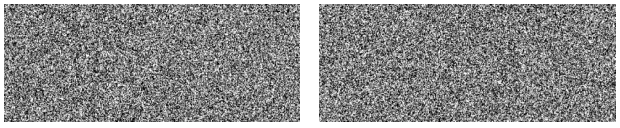
In ideal conditions, a TRNG should have an equal probability of generating 0 and 1. The bias degree of a sequence not only impacts its randomness but also renders a TRNG vulnerable. We quantify the bias degree of random numbers using the following equation:

$$Bias = \frac{|N_1 - N_2|}{N_1 + N_2} \times 100\%, \tag{6}$$

where N_1 and N_2 denote the counts of 1 and 0, respectively. To evaluate our proposed TRNG, we conducted a statistical analysis using 10 sets of generated 1 Mbit random data. The statistical deviations for each FPGA are 0.0075% (Virtex-6) and 0.0069% (Artix-7), respectively. Moreover, we graph the random sequences into a bitstream image, as shown in Figure 7, in which '1' is denoted by a black pixel in the left image and a white pixel in the right image. The uniform distribution of black and white pixels proves that the random numbers generated by our proposed TRNG are uniform and unbiased.



(a) Virtex-6



(b) Artix-7

Figure 7: Bitstream image.

4.4 Autocorrelation Test

The autocorrelation test uses the autocorrelation function (ACF) to test whether a TRNG can produce independent random numbers. According to Karl Pearson's statistical standard, a correlation

coefficient below 0.3 indicates that random sequences are uncorrelated. We set the lag coefficient from 1 to 100 and calculate the autocorrelation of 1Mbit random sequences. As shown in Figure 8, the test results indicate a low autocorrelation, demonstrating our DH-TRNG is resilient to correlation analysis attacks.

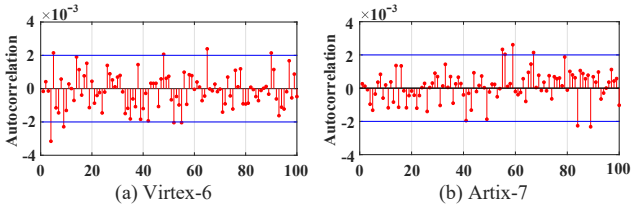


Figure 8: Results of autocorrelation function test.

4.5 Process, Voltages, and Temperatures (PVT) Test

The proposed DH-TRNG is evaluated across a range of temperatures (-20℃ ~ 80℃) and voltages (0.8V ~ 1.2V) on different process FPGAs. We collect 100 different sequences with 1Mbit in each set of experiments for the NIST SP800-90B test and calculate the min-entropy. As shown in Figure 9, the proposed TRNG has the largest min-entropy at 20℃ and 1.0V. Furthermore, when the temperature and voltage vary, the min-entropy presents a slight decrease but remains consistently high, proving our DH-TRNG also has sufficient robustness to PVT fluctuations.

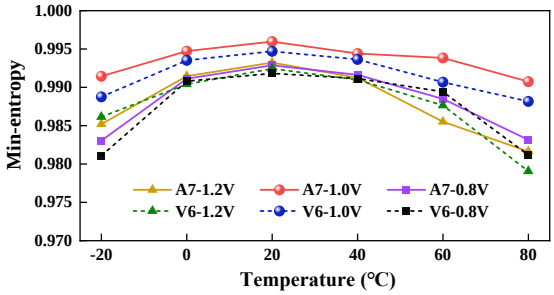


Figure 9: Results of PVT test.

4.6 Comparison with Prior Arts

For a fair and comprehensive evaluation, we compare the proposed design with the state-of-the-art TRNGs in terms of throughput, area, and power on Xilinx Artix-7 FPGA. As shown in Table 6, references [12, 13, 15, 17] have low power consumption but suffer from poor throughput. References [3, 14, 16] have enhanced throughput, and the $\frac{\text{Throughput}}{\text{Slices}\cdot\text{Power}}$ reaching 432.97 in [3]. However, they generally need a high area, and the throughput needs further improvement. Different from them, our proposed entropy units and presented reinforcement strategies sufficiently exploit randomness with minimal hardware area to meet high-frequency sampling. It causes our DH-TRNG to stand out with the highest throughput of 620Mbps while incurring only 8 slices and a total power of 0.068W. Compared with the prior arts, DH-TRNG exhibits absolute dominance in throughput and has the highest $\frac{\text{Throughput}}{\text{Slices}\cdot\text{Power}}$ of 1139.7. Thus, it achieves both ultra-high throughput and area-energy efficiency.

Table 6: Comparison in terms of throughput, area, and power.

Design	LUTs	DFFs	Slices	Throughput (Mbps)	Power* (W)	$\frac{\text{Throughput}}{\text{Slices}\cdot\text{Power}}$
FPL'20 [12]	40	29	10	1.91	0.043	4.44
TCASII'21 [13]	4	3	1	0.76	0.025	30.40
TCASI'21 [14]	56	19	18	100	0.068	81.70
TCASI'22 [15]	32	55	33	12.5	0.063	6.01
TCASII'22 [16]	38	121	38	300	0.119	66.34
TC'23 [17]	152	16	40	1.25	0.023	1.36
DAC'23 [3]	24	33	13	275.8	0.049	432.97
This work	23	14	8	**620**	0.068	**1139.7**

* The power of all architectures are conducted on Xilinx Artix-7.

5 CONCLUSION

This paper presents a dynamic hybrid TRNG with ultra-high throughput and area-energy efficiency. The proposed dynamic hybrid entropy unit sufficiently exploits both jitters and metastability to enhance randomness for high-frequency sampling. The entropy is further bolstered by the introduced coupling and feedback strategies, enabling the DH-TRNG to pass both NIST and AIS-31 tests incurring only 8 slices without any post-processing. Moreover, our design is portable across different process FPGAs and supports fully automated placement and routing. Compared with state-of-the-art TRNGs, the proposed DH-TRNG achieves the highest throughput and $\frac{\text{Throughput}}{\text{Slices}\cdot\text{Power}}$.

ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (No. U20A20202 and 62122023) and the Hunan Provincial Key Research and Development Project (No. 2023GK2011).

REFERENCES

[1] N. D. Truong et al., "Machine learning cryptanalysis of a quantum random number generator," *IEEE TIFS*, pp. 403–414, 2018.
[2] R. Zhang et al., "A 0.186-pJ per bit latch-based true random number generator featuring mismatch compensation and random noise enhancement." *IEEE JSSC*, pp. 2498–2508, 2022.
[3] Z. Lu et al., "An FPGA-Compatible TRNG with Ultra-High Throughput and Energy Efficiency," in *DAC*, 2023.
[4] J. Cui et al., "Design of true random number generator based on multi-stage feedback ring oscillator," *IEEE TCASII*, pp. 1752-1756, 2021.
[5] A. Hajimiri et al., "Jitter and phase noise in ring oscillators," *IEEE JSSC*, 1999.
[6] C. Tokunaga et al., "True random number generator with a metastability-based quality control," *IEEE JSSC*, pp. 78-85, 2008.
[7] M. Majzoobi et al., "FPGA-based true random number generation using circuit metastability with adaptive feedback control," in *CHES*, 2011.
[8] Q. Peng et al., "A Compact TRNG design for FPGA based on the Metastability of RO-Driven Shift Registers," *ACM TODAES*, 2023.
[9] R. Della Sala et al., "High-throughput FPGA-compatible TRNG architecture exploiting multistimuli metastable cells," *IEEE TCASI*, 2022.
[10] K. Wold et al., "Analysis and enhancement of random number generator in fpga based on oscillator rings," in *IEEE International Conference on Reconfigurable Computing and FPGAs*, 2008.
[11] Y. Liu et al., "A bias-bounded digital true random number generator architecture." *IEEE TCASI*, pp. 133-144, 2017.
[12] N. Fujieda, "On the feasibility of tero-based true random number generator on xilinx fpgas," in *FPL*, 2020.
[13] R. Della Sala et al., "A novel ultra-compact FPGA-compatible TRNG architecture exploiting latched ring oscillators," *IEEE TCASII*, 2022.
[14] X. Wang et al., "High-throughput portable true random number generator based on jitter-latch structure," *IEEE TCASI*, 2021.
[15] M. Gruji´c et al., "Trot: A three-edge ring oscillator based true random number generator with time-to-digital conversion," *IEEE TCASI*, 2022.
[16] F. Frustaci et al., "A high-speed fpga-based true random number generator using metastability with clock managers," *IEEE TCASII*, 2022.
[17] K. Pratihar et al., "Birds of the Same Feather Flock Together: A Dual-Mode Circuit Candidate for Strong PUF-TRNG Functionalities." *IEEE TC*, 2023.