applied sciences

MDPI

Article

A Low-Complexity Start–Stop True Random Number Generator for FPGAs

Łukasz Matuszewski * and Mieczysław Jessa

The Faculty of Computing and Telecommunications, Poznan University of Technology, 60-965 Poznan, Poland; mieczyslaw.jessa@put.poznan.pl

* Correspondence: lukasz.matuszewski@put.poznan.pl

Abstract: This paper introduces a low-complexity start–stop true random number generator (TRNG) utilizing jitter in ring oscillators (ROs). Incorporating phase detectors enhances entropy extraction from the same number of ROs. The raw bits undergo online post-processing using the SHA-1 algorithm, which is widely supported by many programming languages. The output bit streams pass all NIST statistical tests (SP 800-22 and SP-90B). Bits are generated on demand, enhancing security by preventing eavesdropping during continuous bit production. The TRNG maintains its performance regardless of the FPGA manufacturer.

Keywords: random number generator; true randomness; ring oscillators; entropy; restarts; statistical tests; FPGA

Citation: Matuszewski, Ł.; Jessa, M. A Low-Complexity Start–Stop True Random Number Generator for FPGAs. *Appl. Sci.* **2024**, *14*, 5642. https://doi.org/10.3390/app14135642

Academic Editor: Alessandro Lo Schiavo

Received: 15 May 2024
Revised: 21 June 2024
Accepted: 21 June 2024
Published: 28 June 2024

1. Introduction

Random number generators (RNGs) are widely used in many applications [1]. There are two types of RNGs: pseudo-random number generators (PRNGs), which can be described with a deterministic algorithm, and true random number generators (TRNGs), which produce non-deterministic sequences even if the generator's structure is known. Currently, most systems are digital constructions. Therefore, true random number generators are anticipated to be fully digital and integrated into a single chip using a system that utilizes random numbers. This feature is essential for cryptographic systems because it makes it difficult for attackers to retrieve and manipulate the numbers or sequences produced by the TRNG.

The most popular fully digital TRNG solutions, easily implementable in FPGAs alongside cryptographic systems, rely on jitter or metastable states [2–42]. However, the primary disadvantage of existing TRNGs is the need for the further processing of output bits due to biased output sequences and strong correlations between adjacent bits. Consequently, random sequences do not generally pass statistical tests from available packages like Statistical Test Suite described in SP 800-22 by NIST [43], TU01 [44], or DieHarder [45]. Such sequences need to be subjected to post-processing transformation(s), which improve the statistical properties but usually decrease the bit rate. Since statistical tests cannot differentiate between deterministic (PRNG) and non-deterministic (TRNG) generators, an additional mechanism is required to verify the source of the output bits. This determination is challenging, as TRNG signals contain both deterministic and non-deterministic components [11,12,46]. Thus, before a given solution is accepted as TRNG, we must assess the amount of true randomness. This assessment can be made directly by measuring, e.g., jitter in signals produced by entropy sources, or we can attempt to find another measure. Examples of the first approach are measurements made with an oscilloscope or a dedicated device. The second approach is represented by tests from NIST SP 800-90B [47]. The min-entropy can be estimated for sources with independent bits with an identical distribution IID, and for sources that fail to produce IID output [47]. Estimating the entropy of a noise source from a single, even long, output sequence may result in an

overestimate. To address this, we performed a sequence of restarts and dedicated tests. These ensured that the distribution of samples in a restart sequence was independent of its position within the sequence. Knowledge of one restart sequence should not aid in predicting subsequent sequences [47]. During normal exploitation of a TRNG, health tests should be applied to detect deviations from the intended behavior of the entropy source as quickly as possible and with high probability [47]. General rules concerning the construction of random number generators can be found in SP 800-90C [48].

This paper proposes a novel method for producing random numbers with TRNG exploiting ring oscillators (ROs) implemented in an FPGA. The generator uses the concept introduced originally by Wold and Tan [8], described more widely in other publications [10,16]. Our proposition uses only two ROs and additional phase detectors to increase the true randomness harvested from these oscillators. To increase the security of the proposed TRNG, we postulate producing bits on demand, i.e., starting and stopping the work of a TRNG when random bits are needed by a digital system implemented in the same FPGA.

Section 2 contains a description of the proposed solution. The results of the tests are presented in Section 3. The paper ends with the conclusions in Section 4.

2. A Start–Stop TRNG with Two ROs and a Phase Detector

In an RO-based TRNG, the signal from RO is sampled in a flip-flop, using another signal with a lower frequency (Figure 1). Delay τ is usually implemented as a chain of even-number inverters, a chain of latches, or even a delay line [17].

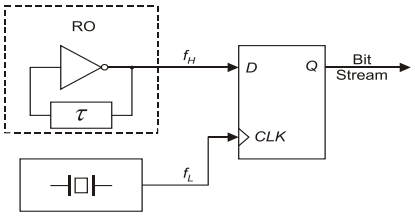

Figure 1. Jitter oscillator sampling as a method of producing random bits.

Using only one ring oscillator does not output a bit string that meets all the statistical tests, even for small sampling frequencies and strong post-processing. To overcome this drawback, Sunar et al. proposed summing the XOR signals from multiple ROs and then sampling the resulting string using a signal with frequency f_L [7]. As shown in [7], the minimum number of ROs is 116. Then, Wold and Tan noticed that the number of ROs used can be significantly reduced if we combine XOR bit streams produced by many low-quality random bit generators called elementary generators (Figure 2) [8]. The TRNG obtained can be considered a combined generator with the XOR combining function, similar to combined PRNG generators.

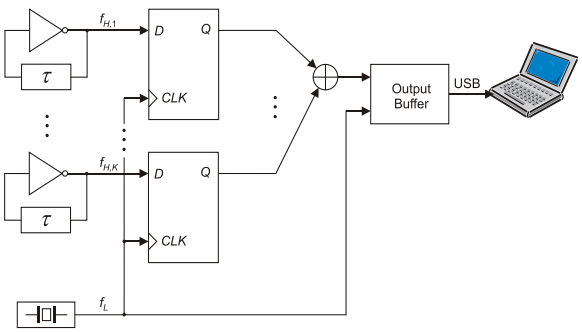

Figure 2. A combined TRNG composed of *K* elementary RO-based TRNGs from Figure 1.

In [9], N. Bochard et al. demonstrated, through simulation experiments, that Wold and Tan's generators can produce sequences passing all statistical tests even when ring oscillators lack jitter. As a result, a fully deterministic source can achieve favorable statistical properties in the bit streams produced by the combined TRNG of Wold and Tan. Therefore, before accepting a solution as a TRNG, it is essential to evaluate the true randomness, especially when utilizing ROs. The structure of the RO-based start–stop TRNG proposed in this paper is shown in Figure 3.

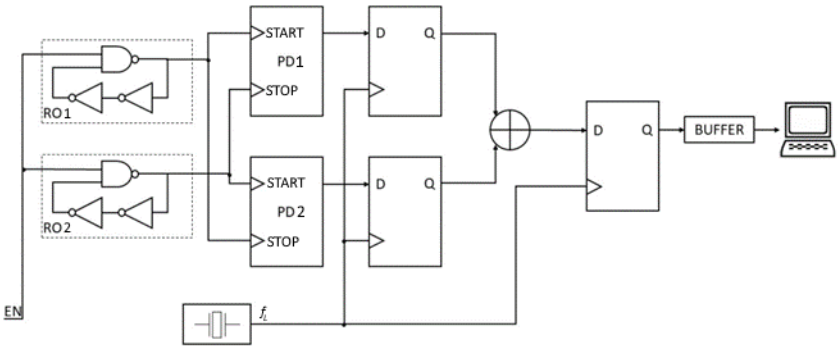

Figure 3. The proposed start–stop TRNG with phase detectors.

The generator uses only two ROs and additional phase detectors. The outputs of the phase detectors are sampled with frequency f_L, with the sampling signal coming from a quartz oscillator. Next, two bitstreams are combined with the XOR function. The start–stop operation provides an external signal EN. For EN = "1", ROs produce rectangular waves. For EN = "0", the generator does not operate. When enabled, the TRNG can provide a specified number of bits on demand, controlled by the number of impulses from the quartz oscillator. The output bits are post-processed using the SHA-1 function.

The jittered signal from a single-ring oscillator can be written as a phase-modulated signal [49]:

$$s(t) = P[\omega t + \varphi(t)], \tag{1}$$

where P denotes the sequence of periodic square pulses, t represents time, $\omega = 2\pi f$ signifies the pulsation of the periodic signal with frequency f, and φ denotes phase fluctuations (jitter). Utilizing Fourier series analysis of the $s(t)$ signal reveals that the rectangular signal shares identical phase properties with its first harmonic [49]. This characteristic significantly simplifies the analysis of a signal affected by phase jitter. Consequently, a sinusoidal signal with jitter can be expressed as follows:

$$s(t) = \sin(\omega t + \varphi(t)), \tag{2}$$

or,

$$s(t) = \sin(\omega t)\cos(\varphi(t)) + \sin(\varphi(t))\cos(\omega t). \tag{3}$$

Due to the slight variations in phase, the cosine of small angles approximates to one, and the sine approximates to the angle value. Thus, Equation (3) can be approximated as follows:

$$s(t) = \sin(\omega t) + \varphi(t)\cos(\omega t). \tag{4}$$

The signal from the kth ring oscillator is as follows:

$$s_k(t) = \sin(\omega_k t) + \varphi_k(t)\cos(\omega_k t). \tag{5}$$

As illustrated in Figure 3, the PD1 produces the signal outlined below:

$$s_1(t) = \sin(\omega_1 t) + (\varphi_1(t) + \varphi_2(t))\cos(\omega_1 t), \tag{6}$$

where ω_1 is the pulsation of RO1. For PD2,

$$s_2(t) = \sin(\omega_2 t) + (\varphi_2(t) + \varphi_1(t))\cos(\omega_2 t), \tag{7}$$

where ω_2 is the pulsation of RO2.

Subsequently, signals $s_1(t)$ and $s_2(t)$ undergo sampling. For simplicity, ideal sampling is assumed using Dirac pulses with a period T_s. The sampled signals are the following:

$$\begin{aligned} d_1(t) &= \sum_{n=-\infty}^{\infty} [\sin(\omega_1 n T_s) + (\varphi_1(nT_s) + \varphi_2(nT_s))\cos(\omega_1 n T_s)]\delta(t - nT_s) \\ d_2(t) &= \sum_{n=-\infty}^{\infty} [\sin(\omega_2 n T_s) + (\varphi_2(nT_s) + \varphi_1(nT_s))\cos(\omega_2 n T_s)]\delta(t - nT_s). \end{aligned} \tag{8}$$

After summing modulo 2 (both signals) and applying the Fourier transform, we obtain the following spectrum:

$$\begin{aligned} S(\omega) = \frac{1}{2T_s} \sum_{n=-\infty}^{\infty} \quad & \frac{1}{j}(\delta(\omega - \omega_1 - n\omega_s) - \delta(\omega + \omega_1 - n\omega_s)) \\ & + \Phi_1(\omega - \omega_1 - n\omega_s) + \Phi_1(\omega + \omega_1 - n\omega_s) \\ & + \Phi_2(\omega - \omega_1 - n\omega_s) + \Phi_2(\omega + \omega_1 - n\omega_s) \\ & + \frac{1}{j}(\delta(\omega - \omega_2 - n\omega_s) - \delta(\omega + \omega_2 - n\omega_s)) \\ & + \Phi_2(\omega - \omega_2 - n\omega_s) + \Phi_2(\omega + \omega_2 - n\omega_s) \\ & + \Phi_1(\omega - \omega_1 - n\omega_s) + \Phi_1(\omega + \omega_1 - n\omega_s) - 4\pi C\delta(0). \end{aligned} \tag{9}$$

The value of C is constant. In comparison, the signal spectrum from generator WT2 is as follows:

$$\begin{aligned} SW(\omega) = \frac{1}{2T_s} \sum_{n=-\infty}^{\infty} \quad & \frac{1}{j}(\delta(\omega - \omega_1 - n\omega_s) - \delta(\omega + \omega_1 - n\omega_s)) \\ & + \Phi_1(\omega - \omega_1 - n\omega_s) + \Phi_1(\omega + \omega_1 - n\omega_s) \\ & + \frac{1}{j}(\delta(\omega - \omega_2 - n\omega_s) - \delta(\omega + \omega_2 - n\omega_s)) \\ & + \Phi_2(\omega - \omega_2 - n\omega_s) + \Phi_2(\omega + \omega_2 - n\omega_s) - 4\pi C\delta(0). \end{aligned} \tag{10}$$

On examining Equations (9) and (10), it is evident that, within the proposed generator, the spectral elements associated with phase disturbances traverse to the output twice on two distinct subcarriers. This feature is advantageous as it enables more precise band coverage and makes the signal more like white noise, as depicted in Figure 4. The generators that were tested comprised two source ring oscillators. The sampling frequency was 10 MHz, and the bit rate was 10 Mbit/s.

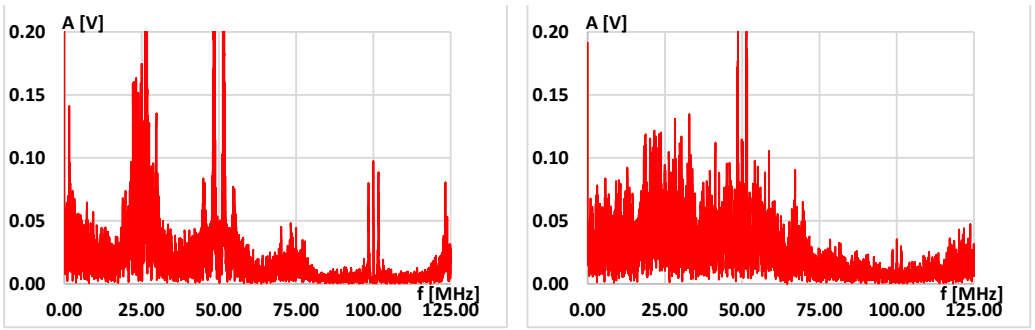

Figure 4. Comparison of spectrum on the output of RO-based generator with two ROs without phase detector (**left side**) and with phase detector (**right side**).

Upon examining the spectra, it is evident that the WT2 generator's spectrum is focused on a few frequencies only. In contrast, the spectrum of the WT2D generator, equipped with a phase detector, is more spread out and effectively covers the entire signal band. This observation aligns with the spectral flatness values calculated using Wiener entropy, as outlined in the following formula [50]:

$$F_l = \frac{\sqrt[N]{\prod_{n=0}^{N-1} x(n)}}{\frac{1}{N}\sum_{n=0}^{N-1} x(n)} = \frac{e^{\frac{1}{N}\sum_{n=0}^{N-1} \ln x(n)}}{\frac{1}{N}\sum_{n=0}^{N-1} x(n)}. \quad (11)$$

The spectrum samples in set N, denoted as $x(n)$, have a spectrum flatness of one when dealing with white noise in the measured signal band. A spectrum flatness value of zero indicates a pure harmonic. When comparing WT2D to WT2, the value of F_l is 0.553671 for WT2 and 0.673340 for WT2D. Thus, better statistical properties of output sequences are expected to be produced by WTD2 than by WTD.

3. The Quality of Bit Streams Produced by the Proposed Generator

The statistical properties analysis was preceded by entropy calculations for output sequences from Wold and Tan's generator using 2-, 3-, and 4-ring oscillators and the proposed generator with two oscillators and two phase detectors. Numerical values were obtained for five sampling frequencies with the TRNG implemented in Virtex 5 XC5VLX50T, as shown in Table 1. The values differed by no more than 16% for other FPGA technologies. WTx represents a TRNG with the construction in Figure 2 using x ROs, while the generator in Figure 3 corresponds to the WT2D symbol.

Table 1. The entropy of output sequences for five values of sampling frequency f_L.

f_L [MHz]	25	15	10	5	1
WT2	0.228335	0.542281	0.654215	0.784981	0.894990
WT3	0.893081	0.970915	0.995306	0.995389	0.998268
WT4	0.994968	0.997237	0.999889	0.999311	0.998281
WT2D	0.999998	0.999999	0.999999	0.999999	0.999999

Next, whether the generators produced IID strings and whether the received strings passed the restart tests described by NIST [47] were checked. The results are shown in Table 2. Symbol Y denotes that the generator passed the tests.

Table 2. The results of tests described by NIST in SP 800-90B for five sampling frequencies f_L.

f_L [MHz]	25		15		10		5		1	
Tests	**IID**	**RES**	**IID**	**RES**	**IID**	**RES**	**IID**	**RES**	**IID**	**RES**
WT2	—	—	—	—	—	Y	—	Y	—	Y
WT3	—	—	—	—	—	—	—	Y	—	Y
WT4	—	—	—	Y	—	Y	Y	Y	Y	Y
WT2D	—	—	—	Y	Y	Y	Y	Y	Y	Y

In additional studies, only the sampling frequencies for which the WTD2 generator produced IID strings that passed the restart tests from NIST SP-90B across all dominant FPGA technologies were considered. These have successfully integrated true random number generators (TRNGs) into Virtex 5, Spartan-3, Spartan-6, Artix-7, Cyclone V, and Max-10. The maximum output bit rate is limited to 10 Mbit/s. Exceeding this threshold results in output sequences that do not pass at least one of the NIST statistical tests from SP 800-22 and SP 90B. Nonetheless, for most cryptographic applications such as key generation, authentication vector generation, and Initial Vector (IV) generation, a bit rate of several Mbit/s is adequate. Initially, the statistical tests described in SP 800-22 were performed on

raw bits. Subsequently, the same bits were processed using the SHA-1 function from the Hashlib Python Library [51].

The SP 800-22 test suite comprises 15 statistical tests, some with multiple subtests. NIST proposed two testing approaches examining the proportion R_β of sequences that pass a statistical test and analyzing the distribution of p-values computed by the software, which examines the value of P_T [43]. A standard set of parameters proposed by NIST in version 2.1.1 was assumed during the tests. For the significance level $\beta = 0.01$ and 1000 tested sequences, each with a length of 10^6 bits, the minimal passing value for the proportion was approximately $R_\beta = 0.9805$. If $P_T \geq 0.0001$, then p-values can be considered uniformly distributed. The NIST tests were performed for sampling frequencies of $f_L = 10$ MHz, 5 MHz, and 1 MHz. The final results are included in Tables 3–5. Symbol Y denotes that the generator produces sequences that pass all tests, including subtests, and that the distribution of p-values is uniform.

Table 3. The final results of statistical tests described in SP 800-22, $f_L = 10$ MHz.

Test	Raw Data				Raw Data + SHA1			
	WT2	**WT3**	**WT4**	**WT2D**	**WT2**	**WT3**	**WT4**	**WT2D**
1. Frequency	—	—	—	Y	Y	Y	Y	Y
2. Block frequency	—	—	—	—	Y	Y	Y	Y
3. Cumulative sums	—	—	—	Y	Y	Y	Y	Y
4. Runs	—	—	—	—	—	—	Y	Y
5. Longest run	—	—	—	—	Y	Y	Y	Y
6. Rank	—	—	—	Y	Y	Y	Y	Y
7. DFT	—	—	—	—	Y	Y	Y	Y
8. Nonoverlapping template	—	—	—	—	—	Y	Y	Y
9. Overlapping template	—	—	—	—	Y	—	—	Y
10. Universal	—	—	—	—	Y	Y	Y	Y
11. Approximate entropy	—	—	—	Y	Y	Y	Y	Y
12. Random excursions	—	—	—	—	Y	Y	Y	Y
13. Random excursions variants	—	—	—	Y	Y	Y	—	Y
14. Serial	—	—	—	—	Y	Y	Y	Y
15. Linear complexity	Y	Y	Y	Y	Y	Y	Y	Y

Table 4. The final results of statistical tests described in SP 800-22, $f_L = 5$ MHz.

Test	Raw Data				Raw Data + SHA1			
	WT2	**WT3**	**WT4**	**WT2D**	**WT2**	**WT3**	**WT4**	**WT2D**
1. Frequency	—	—	—	Y	Y	Y	Y	Y
2. Block frequency	—	—	—	—	Y	Y	Y	Y
3. Cumulative sums	—	—	—	Y	Y	Y	Y	Y
4. Runs	—	—	—	—	—	—	Y	Y
5. Longest run	—	—	—	—	Y	Y	Y	Y
6. Rank	—	Y	Y	Y	Y	Y	Y	Y
7. DFT	—	—	—	—	Y	Y	Y	Y
8. Nonoverlapping template	—	—	—	—	—	—	Y	Y
9. Overlapping template	—	—	—	—	Y	Y	—	Y
10. Universal	—	—	—	—	Y	Y	Y	Y
11. Approximate entropy	Y	—	—	Y	Y	Y	Y	Y
12. Random excursions	—	—	—	—	Y	Y	Y	Y
13. Random excursions variants	—	—	—	Y	Y	Y	—	Y
14. Serial	—	—	—	—	Y	Y	Y	Y
15. Linear complexity	Y	Y	Y	Y	Y	Y	Y	Y

Table 5. The final results of statistical tests described in SP 800-22, f_L = 1 MHz.

Test	Raw Data				Raw Data + SHA1			
	WT2	**WT3**	**WT4**	**WT2D**	**WT2**	**WT3**	**WT4**	**WT2D**
1. Frequency	—	—	—	Y	Y	Y	Y	Y
2. Block frequency	—	—	—	—	Y	Y	Y	Y
3. Cumulative sums	—	—	—	Y	Y	Y	Y	Y
4. Runs	—	—	—	—	—	—	Y	Y
5. Longest run	—	—	—	—	Y	Y	Y	Y
6. Rank	—	Y	Y	Y	—	Y	Y	Y
7. DFT	—	—	—	—	Y	Y	Y	Y
8. Nonoverlapping template	—	—	—	—	—	—	—	Y
9. Overlapping template	—	—	—	—	Y	Y	—	Y
10. Universal	—	—	—	—	Y	Y	Y	Y
11. Approximate entropy	Y	Y	—	Y	Y	Y	Y	Y
12. Random excursions	—	—	—	—	Y	Y	Y	Y
13. Random excursions variants	—	—	—	Y	—	—	—	Y
14. Serial	—	—	—	—	Y	Y	Y	Y
15. Linear complexity	Y	Y	Y	Y	Y	Y	Y	Y

The proposed generator yields IID bit sequences that pass the restart tests specified in NIST 800-90B and, following the application of SHA1, all tests outlined in SP800-22. For the generator depicted in Figure 2, achieving the same outcome requires combining at least seven-bit streams using the XOR function, each produced in the circuit shown in Figure 1. This conclusion holds under the condition that the sampling frequency f_L does not exceed 10 MHz and the outcome remains consistent across different FPGA technologies.

Table 6 presents the proportion R_β and the values of P_T for raw data and after processing for WT7 and WT2D implemented in Virtex 5 XC5LVX50T.

Table 6. The values of proportion R_β and the values of P_T for raw data and after their processing for WT2D and WT7, f_L = 10 MHz.

Test	Raw Data				Raw Data + SHA1			
	WT7		**WT2D**		**WT7**		**WT2D**	
	R_β	P_T	R_β	P_T	R_β	P_T	R_β	P_T
1. Frequency	**0.958**	**0.000**	**0.918**	**0.000**	0.986	0.196	0.987	0.745
2. Block frequency	**0.428**	**0.000**	**0.910**	**0.000**	0.989	0.169	0.989	0.326
3. Cumulative sums	**0.949**	**0.000**	0.991	0.534	0.987	0.081	0.991	0.721
4. Runs	0.993	0.435	0.980	0.114	0.992	0.426	0.989	0.982
5. Longest run	**0.000**	**0.000**	0.910	0.000	0.990	0.998	0.983	0.872
6. Rank	0.990	0.175	0.986	0.094	0.992	0.890	0.989	0.739
7. DFT	0.995	0.771	**0.974**	0.007	0.990	0.249	0.991	0.494
8. Non overlapping template	**0.000**	**0.000**	**0.037**	**0.000**	0.987	0.699	0.992	0.133
9. Overlapping template	0.983	0.046	**0.004**	**0.000**	0.988	0.473	0.993	0.220
10. Universal	**0.000**	**0.000**	**0.808**	**0.000**	0.991	0.152	0.991	0.173
11. Approximate entropy	**0.000**	**0.000**	0.991	0.004	0.989	0.684	0.991	0.957
12. Random excursions	**0.938**	**0.000**	**0.955**	**0.000**	0.988	0.188	0.986	0.012
13. Random excursions variants	**0.961**	**0.000**	0.998	0.012	0.993	0.000	0.996	0.077
14. Serial	**0.000**	**0.000**	**0.007**	**0.000**	0.992	0.465	0.992	0.159
15. Linear complexity	0.990	0.028	0.998	0.167	0.989	0.697	0.990	0.420

The worst cases were shown. Bold denotes the results that do not pass the test.

Reliable comparison of existing proposals of TRNGs is very difficult, and the result depends strongly on the assumed criteria. For comparison, TRNGs that meet the following conditions were selected:

I. The generator can be integrated into the same FPGA alongside a digital system that employs random sequences.
II. The authors declare that the generator produces IID sequences and that the restart tests described in the SP800-90B are performed.

III. The authors declare that the generator produces sequences that pass the randomness tests described in NIST 800-22, so that the proportion of R_β of strings that pass the test is within the limits set by NIST for all tests and subtests, and so that the distribution of p-values is uniform.
IV. The authors have specified the FPGA resources for the proposed TRNG.

The criteria selection reflects users' current expectations regarding the methodology and quality of random number sequence generation for cryptographic purposes. The first criterion enhances security by eliminating the necessity to transmit random sequences to the FPGA from an external circuit, which could be more susceptible than the FPGA itself. The second criterion ensures that random sequences originate from phenomena for which non-deterministic processes prevail. The third criterion assures the high statistical integrity of the generated sequences. The fourth criterion excludes TRNGs that provide comparable security while demanding substantial resources or consuming excessive power.

However, the existing literature predominantly features proposals that do not adhere to the first two conditions, which stem from the characteristics and historical development of FPGAs, along with the relatively recent NIST recommendations on entropy sources for generating random sequences [47]. Consequently, the number of proposals validated using the comprehensive methodology outlined by NIST is relatively small [28,29,36,38–42,52]. We refrain from discussing the efficacy of alternative methods for assessing the randomness of generated sequences here. Alternatively, evaluations can employ AIS-31 tests [53,54], tests from the TU01 package [44], or DieHarder [45]. Some authors adopt a hybrid approach, evaluating the entropy source with NIST 800-90B tests while verifying statistical properties using methods such as AIS-31 [37]. This choice highlights the preference for NIST's methodology despite its limited adoption in everyday research practices. Meanwhile, the proposed method by BSI remains in the draft stage [54]. Table 7 compares TRNGs that meet criteria (I)–(IV).

Table 7. A comparison of TRNGs that meet conditions (I)–(IV).

TRNG Design Technique	Number		Throughput [Mbit/s]
	LUTs	Flip-Flops	
Self-timed rings [28]	56	19	100
Fibonacci-Galois RO [29]	288	190	400
Metastability+jitter [36]	4	3	0.76
Metastability+jitter [38]	36	0	12.50
Multi-stage feedback ring oscillator [39]	24	2	290
RO with XOR gates [40]	13	4	500
Non-identical ROs [41]	15	13	3.50
Delay-Difference-Cell [42]	256	256	225
Metastability [52]	14	6	25
RO with XOR gates [8]	23	23	10
WTD2 (this work)	11	13	10

Notably, the proposed generator achieves a throughput of up to 10 Mbit/s consistently across various FPGA manufacturing technologies, belonging to the category of generators characterized by low complexity and resource requirements. The specific resource values depend on the layout type. What sets this proposal apart from other known designs meeting criteria (I)–(IV) are its complete scalability and portability across different chips and predominant FPGA technologies. In contrast, other authors' designs (listed in Table 7) are tailored to specific chip types and may not function reliably when implemented on different FPGA technologies. The concept of scalability involves deploying multiple instances of the same TRNG within an FPGA without requiring further adjustments to their placement. For example, by integrating 32 instances of the TRNG shown in Figure 3 into a single FPGA, the bit generation rate can be substantially increased to 320 Mbit/s. However, it is crucial to consider that augmenting the number of instances also amplifies complexity, resource utilization, and power consumption due to the rapid switching of flip-flops. Therefore, while

scalability offers enhanced throughput potential, the fundamental limitation remains the availability of FPGA resources required for building these TRNGs and power consumption, especially in devices such as IoT.

4. Conclusions

This paper introduces a novel approach to generating true random number sequences utilizing two-ring oscillators as primary sources of entropy. Phase detectors complement these oscillators by exploiting phase jitter from both sources, enhancing randomness extraction. The proposed TRNG operates on-demand, seamlessly integrating into a single FPGA alongside any digital system requiring random bits. Additionally, raw bit sequences undergo real-time post-processing using established software implementing the SHA-1 algorithm.

This study evaluates performance across various FPGA technologies, specifically Virtex 5, Spartan-3, Spartan-6, Artix-7, Cyclone V, and Max-10. As described, WT2D finds application across systems necessitating authentic random bit generation, with particular relevance to cryptographic contexts.

Enhancing resilience against attacks, particularly frequency injection attacks, is a promising area for exploration in the field of security. One method involves deploying multiple ring oscillators with significantly distinct nominal frequencies. Another approach entails integrating health tests within the FPGA structure to promptly detect deficiencies in randomness and potential interference in the random number generation process caused by attackers, thereby enhancing overall security. The implementation of multiple WTD2 generators, which includes the random selection of the current source of random bits within the same FPGA, appears to be another promising approach to bolster robustness against cryptographic attacks.

Significantly, the proposed TRNG demonstrates exceptional efficiency metrics. It utilizes less than 1% of resources in the smallest FPGA device and consumes under 260 mW for all sampling frequencies up to 10 MHz. These attributes underscore its suitability for resource-constrained environments and highlight its potential impact in cryptographic applications where robust random number generation is critical.

Author Contributions: Conceptualization, Ł.M. and M.J.; methodology, Ł.M. and M.J.; software, Ł.M.; validation, Ł.M. and M.J.; writing—original draft preparation, M.J.; writing—review and editing, Ł.M.; supervision, M.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Poznań University of Technology, grant number 0314/SBAD/0241.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available in the article.

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

References

1. Menezes, A.J.; van Oorschot, P.C.; Vanstone, S.C. *Handbook of Applied Cryptography*; CRC: Boca Raton, FL, USA, 1997.
2. Petrie, C.S.; Connelly, J.A. Modelling and simulation of oscillator-based random number generators. In Proceedings of the the 47th International Symposium on Circuits and Systems, ISCAS'1996, Atlanta, GA, USA, 12 May 1996; Volume 4, pp. 324–327.
3. Fischer, V.; Drutarovski, M. True random number generator embedded in reconfigurable hardware. In *Cryptographic Hardware and Embedded Systems CHES*; Springer: Berlin/Heidelberg, Germany, 2003; pp. 415–430.
4. Kohlbrenner, P.; Gaj, K. An embedded true random number generator for FPGAs. In Proceedings of the the 2004 ACM/SIGDA 12th International Symposium on Field Programmable Gate Arrays, ACM, Monterey CA, USA, 22–24 February 2004; pp. 71–78.
5. Golić, J.D. New methods for digital generation and postprocessing of random data. *IEEE Trans. Comput.* **2006**, *55*, 1217–1229. [CrossRef]
6. Dichtl, M.; Golić, J.D. High speed true random number generation with logic gates only. In Proceedings of the 9th International Workshop, Cryptographic Hardware and Embedded Systems–CHES, Vienna, Austria, 10–13 September 2007; pp. 45–62.

7. 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.* **2006**, *56*, 109–119. [CrossRef]
8. Wold, K.; Tan, C.H. Analysis and enhancement of random number generator in FPGA based on oscillator rings. *Int. J. Reconfigurable Comput.* **2009**, *2009*, 501672. [CrossRef]
9. Bochard, N.; Bernard, F.; Fischer, V. Observing the randomness in RO-based TRNG. In Proceedings of the 2009 International Conference on Reconfigurable Computing and FPGAs, Cancun, Mexico, 9–11 December 2009; pp. 237–242.
10. Wold, K.; Petrović, S. Optimizing speed of a true random number generator in FPGA by spectral analysis. In Proceedings of the 2009 Fourth International Conference on Computer Sciences and Convergence Information Technology, ICCIT'09, Seoul, Republic of Korea, 24–26 November 2009; pp. 1105–1110.
11. Valtchanov, B.; Aubert, A.; Bernard, F.; Fischer, V. Modeling and observing the jitter in ring oscillators implemented in FPGAs. In Proceedings of the IEEE Workshop on Design and Diagnostics of Electronic Circuits and Systems, DDECS'08, Bratislava, Slovakia, 16–18 April 2008; pp. 1–6.
12. Valtchanov, B.; Fischer, V.; Aubert, A.; Bernard, F. Characterization of randomness sources in ring oscillator-based true random number generators in FPGAs. In Proceedings of the IEEE Workshop on Design and Diagnostics of Electronic Circuits and Systems, DDECS'10, Vienna, Austria, 14–16 April 2010; pp. 48–53.
13. Jessa, M.; Jaworski, M. Randomness of a combined RBG based on the ring oscillator sampling method. In Proceedings of the International Conference on Signals and Electronic Systems, ICSES'10, Gliwice, Poland, 7–10 September 2010; pp. 323–326.
14. Jessa, M.; Matuszewski, L. Enhancing the Randomness of a Combined True Random Number Generator Based on the Ring Oscillator Sampling Method. In Proceedings of the 2011 International Conference on Reconfigurable Computing and FPGAs, Cancun, Mexico, 30 November–2 December 2011; pp. 274–279.
15. Baudet, M.; Lubicz, D.; Micolod, J.; Tassiaux, A. On the security of oscillator-based random number generators. *J. Cryptol. J. Cryptol.* **2011**, *24*, 398–425. [CrossRef]
16. Wold, K.; Petrović, S. Security properties of oscillator rings in true random number generators. In Proceedings of the 15th Inter-national Symposium on Components, Circuits, Devices and Systems, Tallinn, Estonia, 18–20 April 2012; pp. 145–150.
17. Jessa, M.; Matuszewski, L. Producing random bits with delay-line-based ring oscillators. *Int. J. Electron. Telecommun.* **2013**, *59*, 41–50. [CrossRef]
18. Lubicz, D.; Bochard, N. Towards an Oscillator Based TRNG with a Certified Entropy Rate. *IEEE Trans. Comput.* **2015**, *64*, 1191–1200. [CrossRef]
19. De Micco, L.; Larrondo, H.A. Measuring the jitter of ring oscillators by means of information theory quantifiers. In *Communications in Nonlinear Science and Numerical Simulation*; Elsevier: Amsterdam, The Netherlands, 2017; Volume 43, pp. 139–150, ISSN 1007-5704.
20. Cao, Y.; Chang, C.; 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; pp. 37–42. [CrossRef]
21. Cherkaoui, A.; Fischer, V.; Aubert, A.; Fesquet, L. Comparison of self-timed ring and inverter ring oscillators as entropy sources for FPGAs. In Proceedings of the 2012 Design, Automation & Test in Europe Conference & Exhibition (DATE), Dresden, Germany, 12–16 March 2012; pp. 1325–1330.
22. Cherkaoui, A.; Fischer, V.; Aubert, A.; Fesquet, L. A self-timed ring based true random number generator. In Proceedings of the IEEE 19th International Symposium on Asynchronous Circuits and Systems (ASYNC), Santa Monica, CA, USA, 19–22 May 2013; pp. 99–106.
23. Gimenez, G.; Cherkaoui, A.; Frisch, R.; Fesquet, L. Self-timed Ring based True Random Number Generator: Threat model and countermeasures. In Proceedings of the IEEE 2nd International Verification and Security Workshop (IVSW), Thessaloniki, Greece, 3–5 July 2017.
24. Tao, S.; Yu, Y.; Dubrova, E. FPGA Based True Random Number Generators Using Non-Linear Feedback Ring Oscillators. In Proceedings of the 16th IEEE International New Circuits and Systems Conference (NEWCAS), Montreal, QC, Canada, 24–27 June 2018; pp. 213–216. [CrossRef]
25. Şarkışla, M.A.; Ergün, S. Ring Oscillator Based Random Number Generator Using Wake-up and Shut-down Uncertainties. In Proceedings of the 2018 Asian Hardware Oriented Security and Trust Symposium (AsianHOST), Hong Kong, China, 17–18 December 2018; pp. 104–108. [CrossRef]
26. 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]
27. Lin, J.; Wang, Y.; Zhao, Z.; Hui, C.; Song, Z. A new method for true random number generation based on Galois Ring Oscillator with event sampling Architechture in FPGA. In Proceedings of the 2020 IEEE International Instrumentation and Measurement Technology Conference, Dubrovnik, Croatia, 25–28 May 2020; pp. 1–6.
28. Wang, X.; Liang, H.; Wang, Y.; Yao, L.; Guo, Y.; Yi, M.; Huang, Z.; Qi, H.; Lu, Y. High-throughput portable true random number generator based on jitter-latch structure. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2021**, *68*, 741–750. [CrossRef]
29. Nannipieri, P.; Di Mateo, S.; Baldanzi, L.; Crocetti, L.; Belli, J.; Fanucci, L. True random number generator based on Fibonacci-Galois ring oscillators for FPGA. *Appl. Sci.* **2021**, *11*, 3330. [CrossRef]

30. Vasyltsov, I.; Hambardzumyan, E.; Kim, Y.-S.; Karpinskyy, B. Fast digital TRNG based on metastable ring oscillator. In Proceedings of the Workshop Cryptograph. Hardware Embedded Systems (CHES), Washington, DC, USA, 10–13 August 2008; pp. 164–180.
31. Bucci, M.; Luzzi, R. Fully digital random bit generators for cryptographic applications. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2008**, *4*, 861–875. [CrossRef]
32. Majzoobi, M.; Koushanfar, F.; Devades, S. FPGA-based true random number generation using circuit metastability with adaptive feedback control. In Proceedings of the Workshop Cryptograph. Hardware Embedded Systems (CHES), Nara, Japan, 28 September–1 October 2011; pp. 17–32.
33. Wieczorek, P.; Golofit, K. Dual-Metastability Time-Competitive True Random Number Generator. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2014**, *61*, 134–145. [CrossRef]
34. Wieczorek, P.Z.; Golofit, K. True Random Number Generator Based on Flip-Flop Resolve Time Instability Boosted by Random Chaotic Source. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2018**, *61*, 134–145. [CrossRef]
35. Kaysici, H.İ.; Ergün, S. Random Number Generator Based on Metastabilities of Ring Oscillators and Irregular Sampling. In Proceedings of the 2020 27th IEEE International Conference on Electronics, Circuits and Systems (ICECS), Glasgow, UK, 23–25 November 2020; pp. 1–4. [CrossRef]
36. Sala, R.D.; Bellizia, D.; Scontti, G. A novel ultra-compact FPGA-compatible TRNG architecture exploiting latched ring oscillators. *IEEE Trans. Circuits Syst. II Express Briefs* **2022**, *3*, 1672–1676.
37. Grujcić, M.; Verbauwhede, I. TROT: A three-edge ring oscillator based true random number generator with time-to-digital conversion. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2022**, *69*, 2435–2448. [CrossRef]
38. Sala, R.D.; Bellizia, D.; Scontti, G. High-throughput FPGA-compatible TRNG architecture exploiting multistimuli metastable cells. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2022**, *12*, 4886–4897. [CrossRef]
39. Cui, H.; Yi, M.; Cao, D.; Yao, L.; Wang, X.; Liang, H.; Huang, Z.; Qi, H.; Ni, T. Design of true random number generator based on multi-stage feedback ring oscillator. *IEEE Trans. Circuits Syst. II Express Briefs* **2022**, *69*, 1752–1756. [CrossRef]
40. Ni, T.; Peng, Q.; Bian, J.; Yao, L.; Huang, Z.; Yan, A.; Wang, S.; Wen, X. Design of True random number generator based on multi-ring convergence oscillator using pulse enhanced randomness. *IEEE Trans. Circuits Syst. I Regul. Pap.* 2023, *in press*.
41. Mahalingam, H.; Rethinam, S.; Janakiraman, S.; Rengarajan, A. Non-identical inverter rings as an entropy source: MIST-90B-verified TRNG architecture on FPGAs for IoT device integrity. *Mathematics* **2023**, *11*, 1049. [CrossRef]
42. Sala, R.D.; Scontti, G. Exploiting the DD-Cell as an ultra-compact entropy source for an FPGA-based Re-configurable PUF-TRNG architecture. *IEEE Access* **2023**, *11*, 86178–86194. [CrossRef]
43. Rukhin, A.; Soto, J.; Nechvatal, J.; Smid, M.; Barker, E.; Leigh, S.; Levenson, M.; Vangel, M.; Banks, D.; Heckert, A.; et al. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. *NIST Special Publication 800-22*. Available online: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-22r1a.pdf (accessed on 5 May 2024).
44. L'Ecuyer, P.; Simard, R. TestU01: A Software Library in ANSIC C for Empirical Testing of Random Number Generators. Software User's Guide. Available online: http://www.iro.umontreal.ca/~lecuyer (accessed on 5 May 2024).
45. Brown, R.G. DieHarder, A Gnu Public License Random Number Tester. Available online: https://rurban.github.io/dieharder/manual/dieharder.pdf (accessed on 5 May 2024).
46. McNeill, J.A.; Ricketts, D.S. *The Designer's Guide to Jitter in Ring Oscillators*; Springer: San Hose, VA, USA, 2009.
47. Turan, M.S.; Barker, E.; Kelsey, J.; McKay, K.; Baish, M.L.; Boyle, M. Recommendation for the Entropy Sources Used in Random Bit Generation. *NIST Special Publication 800-90B*. Available online: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90B.pdf (accessed on 5 May 2024).
48. Barker, E.; Kelsey, J.; McKay, K.; Roginsky, A.; Turan, M.S. Recommendation for Random Bit Generators (RBG) Constructions. *NIST Special Publication 800-90C*. Available online: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90C.3pd.pdf (accessed on 5 May 2024).
49. Trischitta, P.R.; Varma, E.L. *Jitter in Digital Transmission Systems*; Artech House: Norwood, MA, USA, 1989.
50. Madhu, N. Note on measures for spectral flatness. *Electron. Lett.* **2009**, *45*, 1195–1196. [CrossRef]
51. Hashlib Python Library. Available online: https://docs.python.org/3/library/hashlib.html (accessed on 5 May 2024).
52. Peng, Q.; Bian, J.; Huang, Z.; Wang, S.; Yan, A. A Compact TRNG Design for FPGA Based on Metastability of RO-Driven Shift Registers. *ACM Trans. Des. Autom. Electron. Syst.* 2023, *in press*. Available online: https://dl.acm.org/doi/pdf/10.1145/3610295 (accessed on 5 May 2024). [CrossRef]
53. Killmann, W.; Schindler, W. *A Proposal for Functionality Classes for Random Number Generators*; Technical Reference of AIS 20/31; BSI: Bonn, Germany, 2011.
54. Peter, M.; Schindler, W. A Proposal for Functionality Classes for Random Number Generators. *v.2.35–DRAFT*; 2 September 2022. Available online: https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Certification/Interpretations/AIS_31_Functionality_classes_for_random_number_generators_e.pdf?__blob=publicationFile&v=4 (accessed on 5 May 2024).