Original scientific paper

https://doi.org/10.33180/InfMIDEM2019.204

Informacije MIDEM
Journal of Microelectronics, Electronic Components and Materials
Vol. 49, No. 2(2019), 79 – 90

Implementation of a Digital TRNG Using Jitter Based Multiple Entropy Source on FPGA

Ali Murat Garipcan, Ebubekir Erdem

Firat University, Department of Computer Engineering, Elazig, Turkey

Abstract: In this study, hardware implementation and evaluation of a true random number generator (TRNG) is presented. For the implementation, Field Programmable Gate Array (FPGA) hardware, in which numerical processes based on an algorithmic basis are carried out, was used. In the system, ring oscillators (ROs) with similar structures were used as a noise source, and true randomness was obtained by sampling the jitter signals originating from the oscillators. However, the most critical cryptographic disadvantage of jitter-based TRNGs is the statistical inadequacy of the system. At this point, in contrast to existing designs, entropy sources derived from the subsets of ROs were used in the sampling and post-processing stage. The statistical quality of the system was improved by using true random numbers/inputs obtained from these entropy sources in the sampling and post-processing stage. With sampling and post-processing inputs, the use of complex post-processing techniques that limit the output bit rate of the generator in the system was not required. Thus, a high-performance adaptable TRNG model with reduced hardware resource consumption is obtained. The statistical validation of the TRNG, which was tested in 6 different scenarios for two separate ring oscillator (RO) architectures and three different operating frequencies, was performed with the NIST 800-22 and AIS31 test packages.

Keywords: Jitter; oscillator rings; FPGA; true random number generators; cryptography.

Uporaba digitalne TRNG na FPGA z uporabo več entropijskih virov na osnovi tresenja

Izvleček: V prispevku je obravavana strojna oprema in ocenitev pravega generatorja naključnih števil (TRNG). Izvedba je bila narejena na FPGA strojni opremi. Za vir šuma je bil uporabljen krožni oscillator, naključnost pa je zagotovljena s tresenjem originalnega signal oscilatorjev. Kriptografska slabost TRNG na osnovi tresenja je njihova statstična nezadostnost. V nasprotju z obstoječimi sistemi so za reševanje tega problema uporabljeni entropijski viri RO v fazi vzorčenja in naknjanje stopnje obdelave. Statistična kvaliteta je bila z uporabo pravih naključnih števil iz entropijskih virov močno izboljšana in to brez uporabe kompleksnih tehnik naknadnega procesiranja. Validacija šestih scenarijev in dveh RO je bila opravljena na testnih paketih NIST 800-22 in AIS31.

Ključne besede: tresenje; krožni oscilator; FPGA; generatov naključnih števil; kriptografija

** Corresponding Author's e-mail: agaripcan6223@gmail.com*

1 Introduction

In computer science, random numbers are used in many different fields such as programming, simulation, statistical sampling, chance games, and cryptography. While simple statistical features are often sufficient, random numbers must meet strict requirements when it comes to cryptography because the understanding of security in cryptographic systems is based on the confidentiality of randomly generated numbers used for performing critical functions in the system. In addition to their excellent statistical properties, random numbers, which are the complementary element of cryptographic systems, should not contain hidden or explicit patterns between their elements and should be unpredictable. Random numbers, which do not meet these characteristic requirements, jeopardize the reliability of cryptographic systems in which they are used.

Generation of random numbers, which provide the characteristic requirements needed in cryptography, forms an essential and challenging problem area [1]. For obtaining these numbers, customized components known as the Random Number Generators (RNGs) are needed. Random numbers are obtained from two separate design classes: PRNG (pseudo RNG) and TRNG (true RNG). A TRNG is a mechanism that generates true random numbers which are difficult to predict and impossible to reproduce, by using physical events/situations as an entropy source. A TRNG design architecture is presented in Figure 1 consists of the noise source, sampler, and post-processing hierarchical components. The uncontrollability of the physical processes used as the noise source makes the outputs of the generator unpredictable and unreproducible. Pure random numbers with a poor statistical quality obtained by the digitalization of noise sources in the system are post-processed and passed to the output.

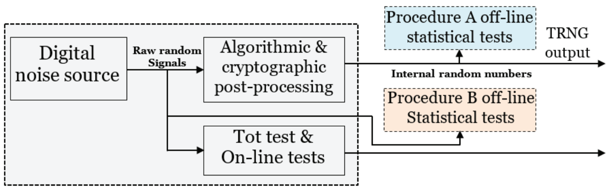

Digital noise source
Raw random Signals
Algorithmic & cryptographic post-processing
Procedure A off-line statistical tests
TRNG output
Internal random numbers
Procedure B off-line Statistical tests
Tot test & On-line tests

Figure 1: Overall design architecture of a true RNG.

The overall design architecture and characteristic behavior of any RNG should coincide with the ideal definition of cryptographic random numbers. In contrast to TRNGs, the PRNG, which corresponds to a standard definition, is a deterministic function. Although their quality of randomness is good, the fact that their outputs appear to be random, yet are predictable limits the use of PRNGs in sensitive cryptographic applications. Although TRNGs are usually slow, costly, and hardware-dependent, they are frequently preferred since they can meet cryptographic requirements.

Random numbers should not be generated on uncontrolled hardware and should not be taken out of the system. If possible, cryptographic systems should be implemented as a whole in a secure computing zone where direct access and programming are not possible on embedded systems [2, 3]. Therefore, the implementation of random number generators is critical. Non-deterministic events/situations on these devices, in which algorithmic processes are performed at the hardware level, adversely affect the behavior of the system. Although non-deterministic processes are minimized by device manufacturers, they cannot be completely eliminated. This situation demonstrates that randomness/noise sources, which are the most critical design components of a TRNG, can be obtained from these devices. Jitter [4-5] and metastability [6-7] on FPGA are frequently used randomness sources, for TRNG designs based on digital design techniques. Especially ROs are used as a source of jitter signals. The clock signals obtained from ROs occur with a deviation from the ideal positions because of the unstable propagation delay in the delay chain. The fact that jitter signals are truly random and easily obtainable has made ROs an important component of TRNGs [8].

In this study, the hardware implementation and evaluation of a TRNG on an FPGA in which ROs were used as the source of noise/randomness are presented. In contrast to the studies in the literature, additional sources of true randomness are used for triggering signal sampling and as an additional input for the post-processing components of the system. In addition to improving the statistical quality of the system, the way in which the entropy sources for additional inputs are implemented further simplifies the system in terms of hardware. Besides to simplifying the system and improving its statistical quality, additional inputs have also eliminated the necessity of complicated post-processing techniques that limit the output bit rate. Therefore, the output bit rate of TRNG is high, although the sampling input is non-periodic. Furthermore, the fact that the additional inputs used in the hierarchical components of the system are truly random made the system cryptographically secure.

The remaining part of the study is organized as follows: Developments in the literature are presented in Section II. In addition to the architectures of the used RO, the conceptual infrastructure of the proposed system is displayed in Section III. Detailed information about the hardware implementation of TRNGs and the results of the experimental analysis are presented in Sections IV and V, respectively. Finally, the study is terminated by presenting the conclusion and recommendations in Section VI.

2 Related works

In cryptography, implementation of an RNG, which is the most important component of the system, in a computation zone where access and manipulation are not possible, is essential regarding the security of the system. Again, the possible attacks on the principles representing the generator and the sub-components of the generator can change the constant theoretical safety limit of TRNGs over time. This causes analytic attacks to occur on the cryptographic system in which the generator is used in a shorter time than expected. By keeping the theoretical safety limit constant, the ability to re-configure implementation platforms to minimize the impact of possible attacks is another important step in system security. For providing these basic requirements, FPGA hardware, in which cryptosystems can be applied as a whole, is a popular platform. Therefore, obtaining the noise source, which is the most critical design component of a TRNG, on these devices is a desired feature [9, 10].

Noise sources such as jitter, metastability, and clock jitter, which mostly emerge as a result of the use of digital devices such as FPGA, are inefficient in terms of cryptographic competencies [11]. In the literature, there are many TRNG designs in which these randomness sources are used. The focus of the designs is to improve the basic design parameters and cryptographic competencies of the generator, as in this study. Some of these studies are as follows:

Phase-Locked Loop (PLL) [4], [12] and multi-ROs [13-14] are used to obtain jitter on digital devices. In [13], 114 free-running ROs, each consisting of 13 inverters, were used in the system. The sampling frequency and output bit rate of the system, in which resilient functions are used as the post-processing technique, are 40 MHz and 2 Mbps, respectively. Due to the complexity of the system, the energy consumption is high, and the output bit rate is low. An improved version of the model proposed by Sunar was proposed by Knuth in [14]. Post-processing was not required in the system, in which the number of inverters and free oscillation ROs was reduced. The most significant deficiency of the system proposed by Wold in [11] was that the generator became insecure due to the entropy loss caused by the reduction of the number of oscillators in the system and occurrence of deterministic randomness. In another study [15], in which the model of Sunar was referenced, a total of 110 free oscillation oscillators each comprising 3 inverters were used in the system. The output bit rate of the system implemented on the Xilinx Virtex II Pro FPGA was measured to be 2.5 Mbs. Other TRNG designs using ROs were also proposed by Kollhenberger and Gaj [16], Golic [17], Dichtl and Golic [18], Tuncer [19], and Avaroglu [8].

In the literature, ROs, in which jitter represents the source of randomness, are also used in Physically Unclonable Function (PUF) based TRNG designs [20-21]. In [20], for two separate PUF circuits in the system, 64 ROs, each consisting of 13 inverters, were used. Random numbers obtained from the RO-PUF (Ring Oscillator-Physically Unclonable Function) implemented on two different FPGA cores against the same query were passed through the post-processing technique, and successful results were obtained. In [21], the query input of the PUF circuit with 128 ROs, each consisting of 3 inverters, was obtained from the logistic map with chaotic behavior. In [4] and [12], instead of ROs, jitter signals obtained by the numerical implementation of analog PLL components were used. The most significant disadvantage of both systems is that a limited number of PLL components can be used on digital devices and a limited number of outputs can be obtained from these components.

A hard-core TRNG design in which the R-S flip-flop is used as a metastability-based source of randomness was presented in [22]. In the system in which the NAND gates of the R-S latch are applied as LUT (look-up table), the outputs of 64, 128, 256 parallelly connected R-S latches were sampled by combining them with the XOR (exclusive OR) operation. The TRNG which did not need post-processing passed NIST tests successfully. Another metastability-based TRNG design was presented in [23]. The system, in which cross-linked NAND gates were used as a source of metastability, was implemented on the Xilinx Virtex XC5VLX50T FPGA chip. The system that reached an output bit rate of 30 Mbps by using the Von Neumann and XOR post-processing techniques passed the NIST tests successfully.

3 Generic architecture of the proposed TRNG

Within the FPGA, each of the digital circuit elements that make up the integrated structure has its own specific unstable time delay. This instability can also be observed on closed or open loop combinational structures, such as ROs, which are used for generating clock signals and which consist of a certain number of delay elements. The temporal deviations caused by the propagation delay occurring on the inverters of the RO cause a period irregularity (jitter) in clock signals [2]. The amount of this irregularity is one of the most important performance metrics of TRNGs and directly affects the quality of the generated random numbers. The jitter occurs in two different ways as deterministic and random. The random jitter which is unpredictable for any sampling time is usually expressed with the Gaussian distribution presented in Equation 1. The jitter occurs as a natural result of flicker, shot and thermal noise which depend on the generation and operating conditions of the logic circuit elements. Periodic irregularity and uncertainty occurring in clock signals due to jitter increases over time as depicted in Figure 2 (a) [2, 9, 19].

$$J_{RJ}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\left(\frac{x^2}{2\sigma^2}\right)} \tag{1}$$

The RO (Figure 2 (b)) is a combinational structure in which an odd number of inverters is sequentially connected forming a delay chain. In addition to their simple combinational definitions, free oscillation ROs have a structure that is easy to implement on digital devices. Therefore, they are frequently used in TRNG designs for obtaining jitter signals.

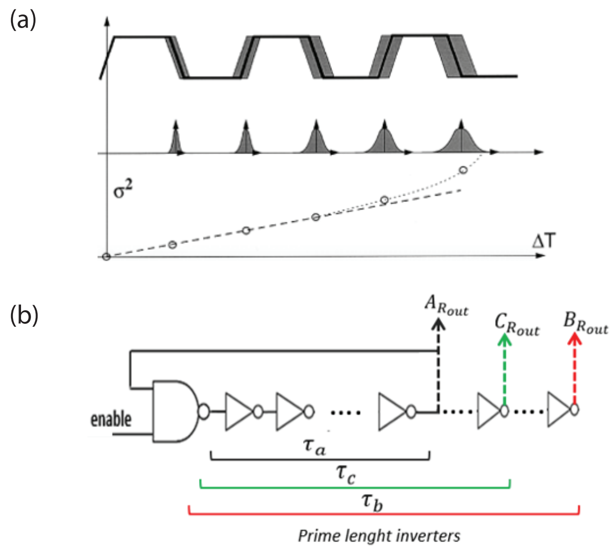


Figure 2: (a) Occurrence of jitter and its time-dependent change, **(b)** combinational structure of the ring oscillator.

The fundamental characteristic of the jitter occurring on ROs is as follows [3, 13]:
The RO-based architecture used as the true randomness source of TRNG is depicted in Figure 3. In the system, $(f_{i1}, f_{i2} .....f_{iN})$ N denotes the number of ROs, and f_i is the ideal square wave signal obtained at each oscillator output. The average period of the f_i signal at any oscillator output T_0 is given by Equation 2, where n is the number of inverters and τ is the delay of a single inverter. The periodic nature of f_i is given by Equation 3.

$$T_0 = 2n\tau \qquad (2)$$

$$f_i(t) = f_i(t + T_0) \qquad (3)$$

However, f_i signals at the oscillator outputs are not in an ideal form due to the instability of the delay occurring on the inverters. In the system in which this unstable delay is represented by $T_{GAUSS,}$ the actual period of the oscillator output signal f_i is given by Equation 4. $T_{GAUSS,}$ which represents the jitter, is the random variable of the system and can take values from $(-T_0/2, T_0/2)$ for any time t. The true randomness of TRNGs required in terms of cryptography is based on the jitter's Gaussian distribution in Equation 1 [3].

$$f_i(t) = f_i(t + T_0 + T_{GAUSS}) \qquad (4)$$

The generic design architecture of the system proposed within the scope of the study is depicted in Figure 3. System consists of three separate oscillator-based hierarchical true randomness sources used for sampling and post-processing inputs together with the noise source. In the system, two separate RO scenarios, taken from [8, 13] consisting of 3 and 13 inverters, respectively, were tested as noise/entropy source. These oscillator scenarios are represented by the block structure in Figure 3 (A). The block structures in Figure 3 (B) and 3 (C) are other entropy sources obtained with a minimum design cost from the noise source of the generator. True random signals obtained from these entropy sources are used as sampling and post-processing inputs of TRNG in the system.

The operation logic of the system can be briefly described as follows: The RO outputs of each entropy source combined with XOR process were sampled through D-type flip-flops to obtain f_s, f_p, and f_k true random outputs. In the system, the non-periodic f_s signal is obtained by sampling from the entropy source in Figure 3 (B). Three different clock signals with a frequency of 50, 100 and 200 MHz are used for sampling. For each sampling scenario, the f_s signal is used as the non-periodic sampling input of the other oscillator clusters and f_p and f_k true random outputs are obtained. In the system, f_k and f_p are obtained from the entropy sources in Figure 3 (A) and (C), respectively, and are raw true random numbers with poor statistical quality. True random outputs of TRNG are obtained by XORing the bit leve f_p and f_k outputs generated in the system at equal times depending on the state of f_s, as in Figure 3 (D).

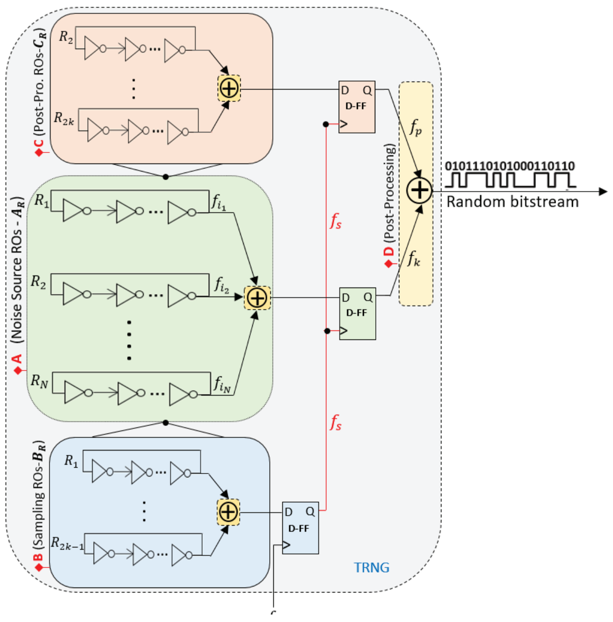


Figure 3: The generic architecture of the proposed system.

The design details of TRNG's oscillator-based entropy sources can be explained as follows. In Figure 3 (A), with the oscillator cluster representing the block structure (noise source) $A_R = \{R_1, R_2, R_3 \ldots R_{114}\}$ and $1 \le k \le 57$. The selection of other RO clusters representing the other block structures of Figures 3 (B) and (C) from this cluster are formed as follows: The RO cluster selected for the sampling frequency (f_s) in Figure 3 (B) is $B_R = \{R_1, R_3, R_5, \ldots R_{2k-1}\}$, and the RO cluster selected for the post-processing input (f_p) in Figure 3 (C) is $C_R = \{R_2, R_4, R_6, \ldots R_{2k}\}$. The use of free-running ROs in the TRNG design was proposed by Sunar in [13]. In the proposed system, it was assumed that the ROs are independent. References [2] and [11] indicate that dependency (also described as phase locking) occurs in 25% of ROs that are supposed to be independent. This caused a loss of entropy in the system. In order to minimize the possible loss of entropy due to this dependence, B_R and C_R oscillator clusters from which additional inputs were obtained were uniformly spread across the whole set of A_R oscillators.

The oscillators in Figure 3 (B) and (C) were converted into a semi-open cycle combinational structure in Figure 2 (a) defined on the noise source, to reduce the energy consumption associated with the system's hardware resource demand. A certain number of the inverters of semi-open cyclical structures are defined on the noise source oscillators, and the oscillation continuity is provided by the feedback loop of these oscillators. In the system, the delay chain of any closed or semi-open cycle RO consists of an odd number of inverters. Therefore, inverters used additionally for semi-open cyclical structures will not change the oscillator outputs at the logic level. However, they will randomly modify the time-dependent periodic irregularity of the output signals. At this point, the basic idea is to increase true randomness time-dependently by making the random behavior of ROs as different as possible from each other and the noise source.

The relationship between the prime number of inverters in a RO and randomness is explained in [13]. Attention was paid to this relationship when increasing the number of inverters. Let τ_a, τ_b and τ_c denote the number of inverters (which must be prime). Then $\tau_c = \tau_a + p$, $\tau_b = \tau_c + k$, and $k \geq p$ (see Figure 2 (a)). In order to minimize the power consumption of ROs in the system, values of k and p were kept at a minimum.

4 Hardware implementation

TRNG scenarios were created with dataflow and schematic design techniques on the Quartus II implementation development platform. In order to measure the real-time performance of the system, the Altera EP-C4GX150 FPGA development board was used during

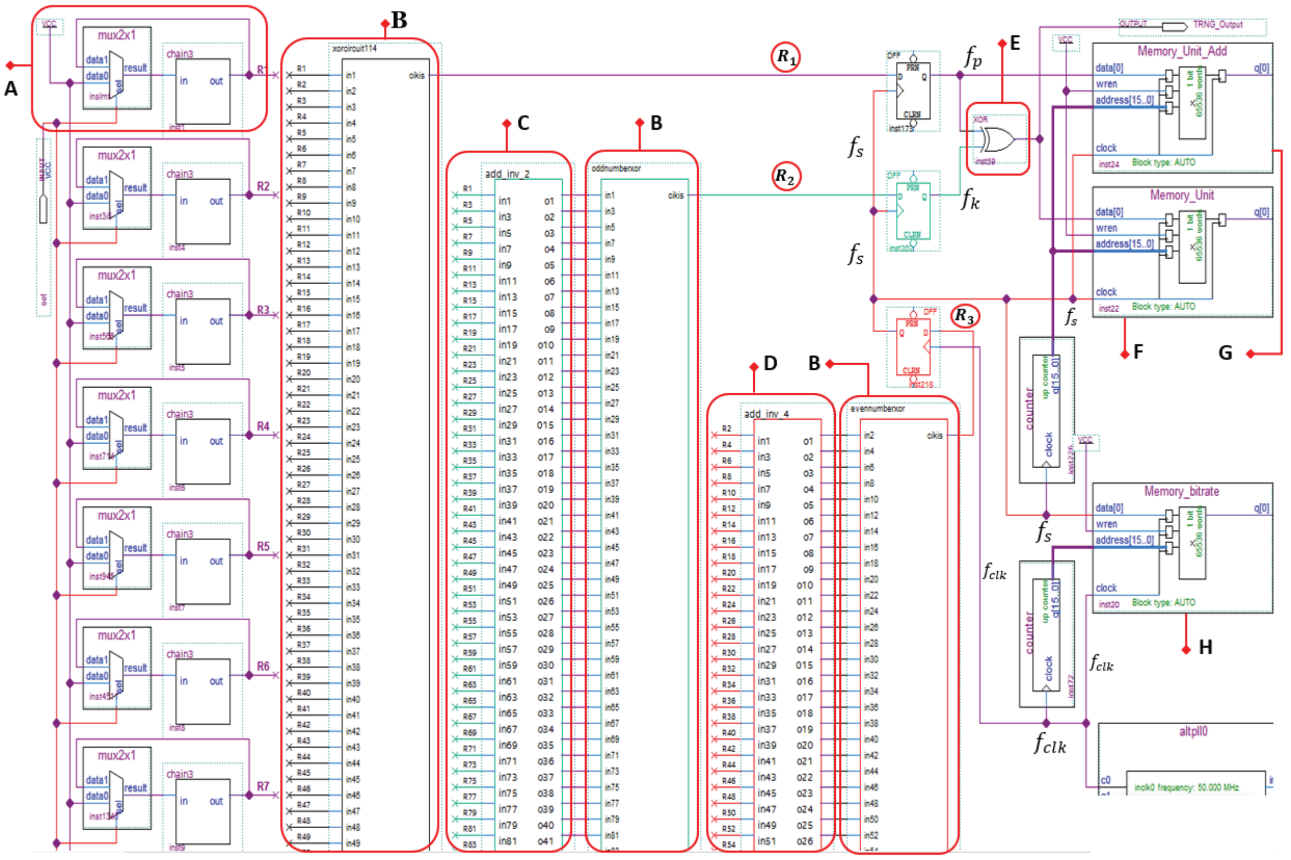


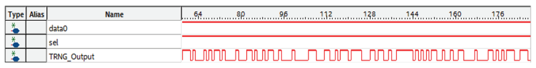
Figure 4: Hardware modeling of the TRNG for a (114,13) scenario

the implementation stage. The overall structure of the proposed system is presented in Figure 3. In the system, two separate RO architectures (114,3) and (114,13) were used as noise sources. The TRNG was tested in six different scenarios for two different noise source architectures and three different operating frequencies. Entropy sources were obtained by modifying the design parameters of these architectures.

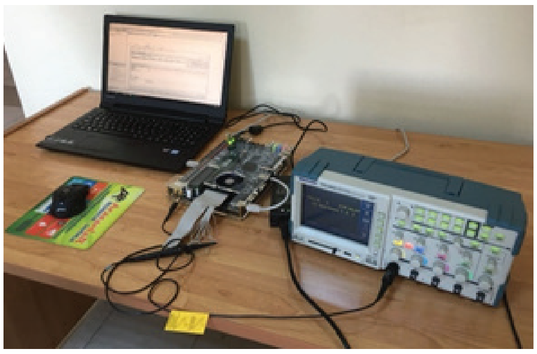
The hardware implementation of a TRNG for the (114,3) oscillator architecture is depicted in Figure 4. Figure 4 (A) is the core oscillator structure which is used as the noise source of the TRNG and that consists of three inverters. 2x1 mux was used as the control variable of the oscillator (Figure 2 (a)). The oscillator output is obtained by applying the output of the mux to the input of the block structure containing the inverters forming the delay chain of the oscillator. The data0 and data1 pins of the mux are the enable and feedback inputs, respectively. The selection pin (sel) of the mux is an excitation signal obtained from the physical environment. For the logic '0' value of the selection pin, the oscillator outputs are constant. For the logic '1' state, the RO is in the feedback position, and the outputs oscillate. The noise source is formed by connecting 114 core structures in parallel. For synchronization of the system, the data0 and sel pins of the muxs at the input of the ring oscillators are common and the data0 pin is connected to the + VCC. In the system, by combining the high oscillating RO outputs, which were used for the noise/randomness source, post-processing input, and sampling operation, with the XOR operation (Figure 4 (B)), R1, R2, and R3 outputs were obtained. Unlike Reference [13] in which a fixed sampling frequency was used, R1 and R2 outputs were sampled with truly random signals (f_s) obtained from R_3.

In Figure 4, R_1, R_2, and R_3 are the combined RO outputs. The oscillator architecture was used for obtaining the combined oscillator outputs R_2 and R_3. Even-numbered outputs of the (114,3) ROs, the noise source, were used for the output R_3 while odd-numbered outputs were used for the output R_2. Before they were obtained, the high oscillation R_2 and R_3 outputs were combined with the XOR operation after they were passed through the block structure as depicted in Figure 4 (C) and (D). In Figure 4 (C), as a result of passing each single oscillator output through two extra inverters, the RO cluster with the (57,5) R_2 post-processing input was obtained. In Figure 4 (D), the oscillator cluster, in which (57,7) R_3 non-periodic sampling signals were attained by using four extra inverters, was obtained. The true random numbers/signals (f_s) in Figure 4 (E) used for sampling the outputs R_1 and R_2 in the system were obtained from the output R_3. For this purpose, the output R_3 was sampled with the help of a type D flip-flop at three different frequencies of 50, 100 and 200 MHz obtained from the PLL for each scenario. The obtained true random numbers and the true random outputs obtained by a synchronized sampling of R1 and R2 outputs were combined with the XOR operation in Figure 4 (E), and the outputs of the TRNG were obtained. The simulation results of the random numbers obtained from the system for the 50 MHz sampling frequency are as shown in Figure 5 (a). The real-time experimental setup of the TRNG in which real-time results are obtained and the real-time variation of the bits produced are depicted in Figures 5 (a) and 5 (b).

(a)



(b)



(c)

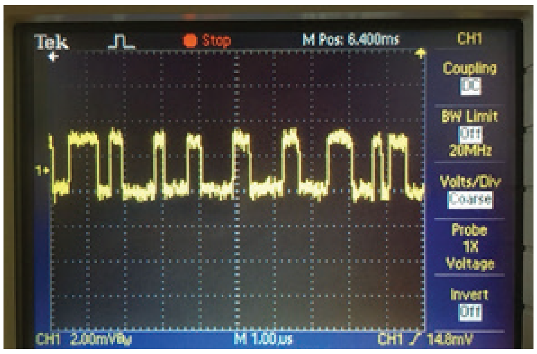


Figure 5: (a) Simulation results**, (b) r**eal-time experimental setup of a TRNG**, (c)** random numbers generated in real time.

In order to measure the statistical adequacy of TRNG, two separate test techniques with different application methods were used. Therefore, two different embedded memory components, such as in Figure 4 (F) and (G), were used to obtain suitable random numbers to each test technique from TRNG. Raw true random numbers (digital noise) sampled from the noise source and post-processed (internal) random numbers were recorded to the memory components, respectively. The width of the memory components using the 16-bit up counter for addressing is 1 bit, and depth is 65536 bits.

Therefore, the sample length of each random number sequence obtained for statistical verification from the TRNG is 65536 bits.

In Figure 4, the non-periodic sampling input (f_s) was used at the same time as the clock signal of the counter circuit and memory component. Thus, truly random numbers at the bit level sampled at equal times in the system are simultaneously recorded the memory cells indicated by the counter. The other embedded memory architecture given in Figure 4 (H) is used to measure the output bit rate of the TRNG. Unlike other memory architectures, the periodic sampling input (f_{clk}) is used as the clock signal of the counter and memory component. Thus, the non-periodic f_s signal is also recorded to the memory component in Figure 4 for output bit rate analysis of TRNG.

5 Experimental results

Cryptographically, one of the most important design evaluation criteria of any TRNG is the process of statistical verification of randomness. This process is essential for the security of TRNGs based on physical noise sources. To what extent the robust randomness criteria are met should be measured by the statistical test tools. There are different test groups developed for the evaluation of randomness. There are different test groups, such as FIPS140, Diehard, Crypt-X, NIST, AIS31, and TestU01 developed to evaluate randomness for TRNGs. However, in the scope of the study, two different hypothesis-based test techniques which has high validity and compelling structure compared to other test groups such as NIST 800-22 and AIS31 were used for statistical verification of TRNG.

Table 1: NIST 800-22 statistical test results

Test Criteria	Stage I		Stage II (sampling true rand. num.)						Stage III (with additional inputs)					
	(114,3)	(114,13)	(114,3)			(114,13)			(114,3)			(114,13)		
	50 MHz	50 MHz	50 MHz	100 MHz	200 MHz	50 MHz	100 MHz	200 MHz	50 MHz	100 MHz	200 MHz	50 MHz	100 MHz	200 MHz
Frequency test	-	-	-	-	-	-	-	-	0.302	0.020	0.576	0.932	0.331	0.403
Frequency test with a block	-	-	-	-	-	-	0.147	0.207	0.077	0.657	0.972	0.515	0.263	0.899
Run test	-	-	-	-	-	-	-	-	0.709	0.0357	0.912	0.112	0.193	0.203
Test for the longest run of ones in a block	-	-	-	-	-	0.019	0.592	0.547	0.668	0.022	0.716	0.256	0.791	0.715
Binary matrix rank	0.457	0.232	0.222	0.621	0.410	0.849	0.602	0.058	0.599	0.917	0.599	0.496	0.308	0.208
Discrete Fourier transform	0.169	-	0.703	0.291	0.561	0.709	0.846	0.630	0.897	0.460	0.818	0.875	0.869	0.962
Non-Overlapping template	-	-	0.894	0.871	0.511	0.200	0.302	0.174	0.825	0.505	0.803	0.725	0.209	0.701
The overlapping template	0.420	0.157	0.222	0.335	0.321	-	0.445	0.056	0.788	0.498	0.055	0.918	0.731	0.394
Universal test	-	0.012	0.039	-	-	0.750	0.015	-	0.813	0.892	0.633	0.564	0.052	0.955
Linear complexity	-	-	0.303	0.540	-	0.837	0.387	0.884	0.866	0.976	0.776	0.506	0.260	0.901
Serial test	-	-	-	-	0.032	0.382	0.423	0.783	0.661	0.812	0.738	0.676	0.344	0.228
	0.219	0.041	0.114	0.545	0.941	0.758	0.613	0.821	0.825	0.980	0.662	0.928	0.667	0.054
Approximate entropy	-	-	-	-	-	-	-	-	0.419	0.451	0.842	0.447	0.165	0.229
Cumulative sums	-	-	-	-	-	-	-	-	0.329	0.292	0.097	0.586	0.139	0.518

The NIST 800-22 test suite consists of 15 different sub-test criteria. The *p-value* (probability value), which corresponds to the randomness probability of the number sequence subjected to the test for each sub-test criterion, is measured. This value is expected to be absolutely greater than parameter *a*, which changes according to the typical importance level [0.001-0.01] of cryptographic applications, for each test criterion [1, 8]. Statistical validation for a random number sequence for any test criterion where this condition is not met is considered to be unsuccessful.

The real-time test setup, in which statistical results were obtained for NIST 800-22, is presented in Figure 5. The test technique was applied in three stages in order to observe the effect of additional true random inputs on statistical results in the system. In the first stage, the combined outputs of the (114,3) and (114,13) ROs were sampled independently with 50, 100 and 200 MHz clock signals. Then, the sampling process was repeated with true random signals/numbers, and the effect of the sampling input on statistical results was observed. In this stage in which partial statistical success was achieved, the successful results presented in Table 1 were obtained by including the post-processing input in the system.

AIS31 test proposed by BSI (German Federal Office for Information Security) was used another statistical validation tool for TRNG. The AIS31 test, which can be used as a statistical verification tool for generators, is also accepted an international standardization process for RNG designs. For this reason, it is frequently used as a popular test technique in recent studies. AIS31 consists of two separate test protocols A and B applied to raw random numbers (digital noise) and internal random numbers (after the post-processing), respectively. In procedure A, the statistical tests cover only the randomness of the bits and they do not cover their unpredictability. In other words, the statistical tests may detect defects of the randomness source, but they cannot verify its randomness. For this reason, in procedure B the entropy tests are added (Coron's test, Collision test etc.) which can verify the unpredictability of the bits and thus the randomness of the source [24, 25].

The test technique consists of a total of 9 separate statistical test criteria, 6 (T0-T5) of these are for procedure A and 3 (T6-T8) for procedure B. Procedure A's T1-T5 test criteria are identical to FIPS-140 tests, another statistical verification tool for TRNGs. Statistical verification fails if more than one test criterion is not met for random numbers in which Procedures A and B are applied. In cases where only one test criterion fails, procedures A and B are repeated for a different random number sequence. If at least one test criterion for the second repetition fails, validation is considered unsuccessful again [26, 27].

As in the NIST 800-22 test, the AIS31 test was applied to the TRNG offline as in Figure 1 and the results in Table 2 were obtained. All tests from the test procedure A (T0-T5) were executed on the internal random numbers, and all tests from the test procedure B (T6-T8) were executed on the raw random numbers. Target random numbers in which A and B procedures are applied in the system are different from each other. Therefore, a second additional memory architecture as in Figure 5 (G) was used to obtain the raw random numbers needed for procedure A. In addition, for procedures A and B, the minimum length of the target random number sequences must be 5.140.000 bits (20000 * 257 = 5140000 bits) and 7.200.000 bits, respectively.

However, the maximum length of each random number sequence obtained from the system for the test is equal to the depth (65536 bits) of 16-bit counter-supported memory units. Quartus II allows export of block memory contents to a text file in address format at run time. In order to obtain sufficient long random number sequence for test, the memory contents of Figure 5 (F) and (G) were consecutively exported to a text file 80 (80*65536=5.242.880) and 110 (110*65536=7.208.960) times respectively. Then, the memory contents in text format were combined in MATLAB and sufficient length two different test files for AIS31 test were obtained. The number of export process is quite high for six different scenarios in the system. Therefore, the AIS31 test was applied only to random numbers obtained from the 50 MHz sampling scenario and test results are given in Table2. In the system, 100 and 200 MHz sampling scenarios are ignored.

Upon examining the results in Table 1 and Table 2, it is observed that the TRNG provides statistical efficiency needed in terms of cryptography for six different scenarios. When the test results were evaluated, it was demonstrated that the proposed system could also be used for cryptographic applications. . In the system, with additional inputs derived from the noise source in order to provide statistical randomness, a TRNG model, which is simple in terms of hardware and which is easy-to-implement with a push button control on digital devices, was obtained. In contrast to complicated post-processing techniques, XOR post-processing technique was used in the system and the output bit rate of the system was not reduced with the post-processing input. Furthermore, the results given in Table 2 show that TRNG has high entropy per bit and its outputs are unpredictable.

Table 2: AIS 31 test results of TRNG for 50 MHz sampling frequency

Procedure		Tests Criteria		(114, 3) Raw random num.	(114, 13) Internal random num.
AIS31	A	T0 (Disjointness test)		Passed	Passed
		T1 (Monobit test)	FIPS 140	Passed	Passed
		T2 (Poker test)		Passed	Passed
		T3 (Run test)		Passed	Passed
		T4 (Long Run test)		Passed	Passed
		T5 (Autocorrelation test)		Passed	Passed
	B	T6 (Uniform distribution test)		Passed	Passed
		T7 (Comparative test for multinomial distributions)		Passed	Passed
		T8 (Entropy test)		Passed	Passed

The output bit rate of TRNG is directly dependent on the frequency of the non-periodic sampling input f_s obtained from the R3 input. Because the one-bit true random output of TRNG occurs dependent on the changes in the logical level of f_s. The f_s sampling signal obtained from both noise source scenarios is also non-periodic in other words, it is truly random. In addition to statistical verification for TRNG, the measurement of the output bit rate, another important evaluation criterion, is based on the off-line analysis of the non-periodic f_s sampling signal in MATLAB.

Random changes of the non-periodic sampling input f_s are recorded to the memory architecture given Figure 5 (H). Thus, 20 different text files, each consisting of 65536 bits, were obtained for the analysis process for two separate scenarios from the memory architecture at run time. In the system, rising-edge triggered D-type flip-flops were used for sampling. Therefore, the output bit rate of the system is determined by looking at the total number of "01" logical transitions randomly occurred in each text file for f_s. The logical validation of the method is as in Figure 6. In Figure 6, the R1 and Q represent the combined noise source outputs (R_1) and raw true random numbers (f_k) obtained from the noise source in Figure 4, respectively. The total number of sampling transitions of f_s for two separate RO scenarios (57,7) and (57,19) is as in Table 3. The f_{clk} is 50 MHz for both RO scenarios in the system.

Table 3: Number of '01' random sampling transitions in text files for f_s

Fclk	(57,7) RO	Average output bit rate of TRNG	(57, 19) RO	Average output bit rate of TRNG
50 MHz	15.639	15.567 (Mbps)	15.621	15.41 (Mbps)
	15.565		15.516	
	15.562		15.615	
	15.526		15.618	
	15.512		15.506	
	15.599		15.376	
	15.518		15.322	
	15.590		15.396	
	15.642		15.378	
	15.520		15.377	

The total number of logic transitions given in Table 3 also represents the average output bit rate of TRNG. The minimum average output bit rate of TRNG is 30.82 and 61.64 Mbps for 100 and 200 MHz values of f_{clk}, respectively. The performance of the proposed TRNG architecture in terms of output bit rate is considerably higher than the other known oscillator-based studies in the literature. When the comparison results given in Table 4 are examined, it is seen that TRNG is successful in terms of output bit rate, which is another important evaluation criterion besides safety.

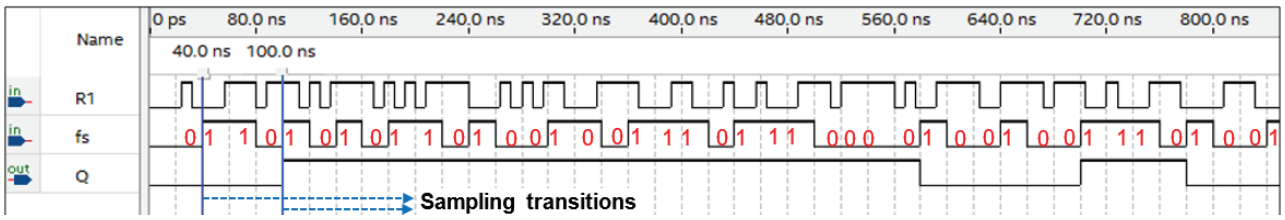


Figure 6: Time-dependent variation of sampling transitions

Table 4: The output bit rate of various FPGA-based TRNGs in the literature

Reference	Oscillator Type	Output bit rate (Mbps)
Avaroğlu et al [8]	Ring+chaos	20.45
Sunar and Stinson [13]	Ring	2.5
Schellekens et al [15]	Ring	2.5
Kohlbrenner and Gaj [16]	Ring	0.5
Fischer et al [17]	PLL	1.0
Dichtl an Golić [18]	Ring	12.5
Tuncer [21]	Ring+chaos	2.17
Hata and Ichikawa [22]	Metastability	12.5
Li et al [23]	Metastability	30.0
Danger et al [28]	Metastability	20.0
Wieczorek and Golofit [29]	FF (flip-flop)	5.0
Istvan et al [30]	Jitter	1.92
Çiçek et al [31]	Chaos	3.2
Tuncer [32]	Ring+chaos	4.77
Koyuncu and Ozcerit [33]	Chaos	58.0
Proposed study (for 200 MHz)	Ring	61.64

6 Conclusions

For TRNGs implemented on digital devices, the entropy of sources of randomness is low. Therefore, it was observed that the statistical quality of random numbers obtained by the pure sampling of ROs in the system was not sufficient to meet the cryptographic competencies. The design architecture was simplified by using true random inputs obtained from the noise source to provide these competencies in the system. The proposed TRNG passed the statistical tests successfully for six different scenarios. The hardware cost of the system, which does not require any additional input from the outside and which does not need the complex post-processing techniques which limit the bit generation rates of the generators, is very low. Therefore, the number of programmable logic elements required for the implementation is less than 1% of the number of the programmable logic elements of the FPGA device used. Thus, it can even be applied to restricted devices.

The designed TRNG has low power consumption, and the final output can pass the NIST800-22 and AIS31 statistical tests, for a minimum output bit rates of 15.41, 30.82 and 61.64 Mbps. The average power consumption of the TRNG for any scenario is 131 milliwatt (mW). This situation can constitute a disadvantage when the increased structural complexity of cryptographic applications depending on security needs is considered. Therefore, the generator can be stopped and operated for critical applications in which energy consumption is important. A true RNG, which can be controlled easily, is fast enough for cryptographic applications and is easily integrable into the system. Besides, the TRNG has an easily adaptable structure for a hierarchically different scenario, frequency, and post-processing techniques. Again, the fact that the additional inputs obtained from entropy sources are truly random made the generator more secure against possible attacks.

7 References

1. Ozkaynak F., "Cryptographically secure random number generator with chaotic additional input", *Nonlinear Dynamics*, 2014;78(3):2015–20. https://doi.org/10.1007/ s11071-014-1591-y
2. Fischer V., "A closer look at security in random number generators design", In International Workshop on Constructive Side-Channel Analysis and Secure Design (pp. 167-182), 2012, Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-29912-4
3. Yang B., "True Random Number Generators for FPGAs", Ph.D. dissertation, Dept. Elect. Eng., Arenberg Doctoral School, Leuven-Belgium, 2018.
4. Fischer V., Drutarovský M., "True random number generator embedded in reconfigurable hardware", In *International Workshop on Cryptographic Hardware and Embedded Systems* (pp. 415-430), 2002, Springer, Berlin, Heidelberg. https://doi.org/
5. Golic´ J.D., "New methods for digital generation and postprocessing of random data," *IEEE Trans. Comput.*, vol.55, no.10, pp.1217– 1229, 2006. https://doi.org/10.1109/tc.2006.164
6. Vasyltsov I., Hambardzumyan E., Kim, Y.S., and Karpinskyy B, "Fast digital TRNG based on metastable ring oscillator", in Proceedings of the 10th International Workshop on Cryptographic Hardware and Embedded Systems (CHES '08), E. Oswald and P. Rohatgi, Eds., vol. 5154 of Lecture Notes in Computer Science, pp. 164–180, Springer, Washington, DC, USA, August 2008. https://doi.org/10.1007/978-3-540-85053-3_11
7. Varchola M., Drutarovsky M., "New high entropy element for FPGA based true random number generators", *in Proceedings of the 12th International Workshop on Cryptographic Hardware and Embedded Systems (CHES '10),* S. Mangard and F.-

X. Standaert, Eds., vol. 6225 of Lecture Notes in Computer Science, pp. 351–365, Springer, Santa Barbara, Calif, USA, August 2010
https://doi.org/10.1007/978-3-642-15031-9_24

8. Avaroğlu E., Tuncer T., Özer A. B., Ergen B., & Türk M., "A novel chaos-based post-processing for TRNG", *Nonlinear Dynamics*, 81(1-2), 189-199, 2015
https://doi.org/10.1007/s11071-015-1981-9
9. Garipcan A.M., Erdem E., " Hardware design and analysis of ring oscillator based noise source for true random number generators," *presented at the International Artificial Intelligence and Data Processing Symposium (IDAP'18)*, 2018.
https://doi.org/10.1109/IDAP.2018.8620811
10. Buchovecká S., Lórencz R., Kodýtek F., & Buček J., "True random number generator based on ring oscillator PUF circuit", *Microprocessors and Microsystems*, *53*, 33-41, 2017
https://doi.org/10.1016/j.micpro.2017.06.021
11. Bochard N., Bernard F., Fischer V., & Valtchanov B., "True-randomness and pseudo-randomness in ring oscillator-based true random number generators", *International Journal of Reconfigurable Computing*, *2010*.
https://doi.org/10.1155/2010/879281
12. Fischer V., Drutarovský M., Šimka M., Celle F., & Komenského P., "Simple pll-based true random number generator for embedded digital systems", In *Proceedings of IEEE Design and Diagnostics of Electronic Circuits and Systems Workshop–DDECS* (pp. 129-136), 2004.
13. 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.*, vol.56, no.1, pp.109–119, 2007.
https://doi.org/10.1109/tc.2007.250627
14. Wold K., Tan C.H., "Analysis and Enhancement of Random Number Generator in FPGA Based on Oscillator Ring", *International Conference on Reconfigurable Computing and FPGAs*, pp.385-390, 2008
https://doi.org/10.1155/2009/501672
15. Schellekens, D., Preneel, B., Verbauwhede, I., "FPGA vendor agnostic true random number generator", *In: Proc. 16th Int. Conf. Field Programmable Logic and Applications* - FPL 2006
https://doi.org/10.1109/FPL.2006.311206
16. Kohlbrenner P., Gaj K., "An embedded true random number generator for FPGAs", *Proc. ACM/SIGDA 12th Intl. Symp. Field Pro- grammable Gate Arrays (FPGA 2004)*, pp.71–78, ACM, 2004.
https://doi.org/10.1145/968280.968292
17. Fischer V., Drutarovský M., Šimka M., Bochard N. "High performance true random number generator in Altera Stratix FPLDs", *In International Conference on Field Programmable Logic and Applications*, 2004.
https://doi.org/10.1007/978-3-540-30117-2_57
18. Dichtl M., Golic´ J.D., "High-speed true random number generation with logic gates only", *Proc. Cryptographic Hardware and Embedded Systems - CHES 2007*, LNCS 4727, pp.45–62, Springer, 2007.
https://doi.org/10.1007/978-3-540-74735-2_4
19. Tuncer T., "Implementation of duplicate trng on fpga by using two different randomness source", *Elektronika ir Elektrotechnika 21(4),* 35–39, 2015.
https://doi.org/10.5755/j01.eee.21.4.12779
20. Tuncer S. A., "Real-time random number generation with RO-based double PUF", *Informacije MIDEM,* 48(2), 121-128, 2018.
21. Tuncer, T., "The implementation of chaos-based PUF designs in field programmable gate array", *Nonlinear Dynamics*, *86*(2), 975-986, 2016.
https://doi.org/10.1007/s11071-016-2938-3
22. Hata H., Ichikawa S., "FPGA implementation of metastability-based true random number generator", *IEICE TRANSACTIONS on Information and Systems*, *95*(2), 426-436, 2012
https://doi.org/10.1587/transinf.E95.D.426
23. Li C., Wang Q., Jiang J., Guan N., "A metastability-based true random number generator on FPGA", In *ASIC (ASICON), 2017 IEEE 12th International Conference on*(pp. 738-741). IEEE, 2017
https://doi.org/10.1109/ASICON.2017.8252581
24. Killmann W., Schindler W., "A proposal for: Functionality classes for random number generators, version 2.0. Tech. rep.", BSI, Bonn, 2011.
25. Hector (Hardware enabled crypto and randomness) Project, Robustness tests on TRNGs and PUFs Technical Report (V1.0), 2015, available: https://hector-project.eu/downloads/HECTOR-D2.4-PU-M41.pdf
26. Marinakis G., "Design and evaluation of random number generators", *Journal of Applied Mathematics and Bioinformatics*5.3 : 155, 2015.
27. Balasch J., Bernard F., Fischer V., Grujić M., Laban M., Petura O., ... & Yang B., "Design and testing methodologies for true random number generators towards industry certification", *IEEE 23rd European Test Symposium (ETS)*, pp. 1-10, 2018. IEEE.
https://doi.org/10.1109/ETS.2018.8400697
28. Danger J.-L., Guilleya S., Hoogvorsta, P., "High speed true random number generator based on open loop structures in FPGAs", *Microelectronics Journal*, Apr 2, 2009.
https://doi.org/10.1016/j.mejo.2009.02.004
29. Wieczorek P. Z., Gołofit K., "Dual-metastability time-competitive true random number generator", *IEEE Transactions on Circuits and Systems* I: Regular Papers, 61(1), 134-145, 2013.
https://doi.org/10.1109/TCSI.2013.2265952

30. Istvan H., Suciu A., Cret O., "FPGA based TRNG using automatic calibration", *IEEE 5th International Conference on Intelligent Computer Communication and Processing*, pp. 373-376, 2009. https://doi.org/10.1109/ICCP.2009.5284733
31. Cicek I., Pusane A.E., Dundar G., " A novel design method for discrete time chaos based true random number generators", *Integration, the VLSI journal,* 47(1), 38-47, 2014 https://doi.org/10.1016/j.vlsi.2013.06.003
32. Tuncer T., Avaroglu E., Türk M., Ozer A. B., "Implementation of non-periodic sampling true random number generator on FPGA", *Informacije Midem*, *44*(4), 296-302, 2015.
33. Koyuncu I., Ozcerit, A.T., "The design and realization of a new high speed FPGA-based chaotic true random number generator", *Comput. Electr. Eng.* 58, 203–214, 2017. https://doi.org/10.1016/j.compeleceng.2016.07.005

Arrived: 01. 05. 2019
Accepted: 22. 07. 2019