INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2013, VOL. 59, NO. 1, PP. 41–50
Manuscript received January 9, 2013; revised March, 2013. DOI: 10.2478/eletel-2013-0005

Producing Random Bits with Delay-Line-Based Ring Oscillators

Mieczysław Jessa and Łukasz Matuszewski

***Abstract*—One of the sources of randomness for a random bit generator (RBG) is jitter present in rectangular signals produced by ring oscillators (ROs). This paper presents a novel approach for the design of delays used in these oscillators. We suggest using delay elements made on carry4 primitives instead of series of inverters or latches considered in the literature. It enables the construction of many high frequency ring oscillators with different nominal frequencies in the same field programmable gate array (FPGA). To assess the unpredictability of bits produced by RO-based RBG, the restarts mechanism, proposed in earlier papers, was used. The output sequences pass all NIST 800-22 statistical tests for smaller number of ring oscillators than the constructions described in the literature. Due to the number of ROs with different nominal frequencies and the method of construction of carry4 primitives, it is expected that the proposed RBG is more robust to cryptographic attacks than RBGs using inverters or latches as delay element.**

***Keywords*—ring oscillator, jitter, random bit generator, combined generator.**

I. Introduction

ONE of promising techniques for producing random bit sequences that pass all statistical tests is combining XOR bit streams produced by independent generators with weak statistical properties. The generators, called the source generators, can use different sources of randomness. It can be noise generated by a physical system [1]–[5], metastable states [6]–[10], the chaos phenomenon [11]–[19] or jitter produced by ring oscillators [20]–[29]. Mixed solutions that combine various properties of these basic techniques also exist. From a cryptography perspective, analog solutions have several drawbacks. They are: the access to random bits which have to be sent to a Field Programmable Gate Array (FPGA) implementing a cryptographic system, continuous work which can reveal weaknesses of the entropy source, the sensitivity to power attack which degenerate the quality of random source or a small output bit rates (below 1 Mbit/s). Such sources require also developed methods for continuous monitoring to detect imperfections of entropy source caused by internal or external conditions. An alternative seems to be random bit generators (RBGs) which use jitter observed in ring oscillators. Such generators can be easily integrated with cryptographic systems in the same FPGA. The RBGs can produce random bits "on demand" which are not visible outside FPGA. Such generator was proposed by Sunar *at. al.* in 2007 [30] and next modified by Wold and Tan in 2008 [31]. The output bit rate reported by Wold and Petrović achieved 300 Mbit/s, and the output streams passed statistical tests for 25 source generators [32]. A year later, N. Bochard *et al.* showed experimentally that such a generator can produce bit streams that pass statistical tests when no random component is present in the ring generators [33], which indicates that this generator should not be a source of randomness. To distinguish between randomness and pseudo-randomness, M. Jessa and M. Jaworski proposed in 2010 to use the chi-square test and the restart mechanism [34]. Through experiments with real circuits, it was proved that a very small amount of randomness, present in a single-ring oscillator, accumulates as the function of the number of bits combined XOR at the same time instant [34], [35]. The generator of Wold and Tan can provide a bit sequence suitable for cryptographic applications, when we choose every j-th bit from the original sequence, where j is greater than a certain minimum value m_{min} [34], [35]. It significantly reduces the throughput, e.g., from 300 Mbit/s, declared in [32], to approximately 7.14 Mbit/s for 50 source generators [34]. This value depends on the location of the project in the field-programmable gate array and may vary for different types of FPGAs. It was also observed that the frequencies of neighbor ROs may depend on each other when they are implemented in the same FPGA [36]. Therefore, the designers try to place the source generators in the FPGA "suitably" to avoid interactions. This approach usually gives satisfactory results but is impractical for industry. Ring-oscillator-based RBG can also be attacked with the use of frequency injection attack [37]. During this attack, a certain number of ROs lock to frequencies injected into the power supply, eliminating the independence of the generators and significantly reducing the randomness of the output bit stream. A method for eliminating this weakness of the RO-based combined RBG was proposed in 2012 in conference paper [38]. In this paper, we describe wider this method and propose its modification. The goal of the modification is to increase the randomness of a single random source and, consequently, the reduction the value of m_{min}. As previously, the key point of the method is to ensure different nominal frequencies of ROs which prevents coupling between neighbor ROs and locking the frequencies of many ROs to the "injected" frequency. The output bit streams pass the NIST 800-22 tests without any additional post-processing, necessary for analog RBGs.

In Section II, the structure of the combined RBG and a brief theory of RO-based RBG are introduced. The construction of the delay element, used in all ROs, is described in the

M. Jessa is with Poznań University of Technology, Faculty of Electronics and Telecommunications, Polanka 3, 60-965 Poznań, Poland (e-mail: mjessa@et.put.poznan.pl).

L. Matuszewski is with Poznań University of Technology, Faculty of Electronics and Telecommunications, Polanka 3, 60-965 Poznań, Poland (e-mail: lukasz.matuszewski@et.put.poznan.pl).

same section. Section III presents a method and the results of assessing the amount of true randomness present in output signals of the combined RBG using a series of primitives, called carry4. A method for enhancing the randomness of the combined RBG using carry4 is described in Section IV. The results of statistical tests are presented in Section V. The paper ends with the conclusions.

II. Combining XOR Independent Random Bits Produced by Many RO-Based RBGs

A. Jitter Produced by Ring Oscillators as a Source of Randomness

A simple method of producing random bits is sampling the rectangular wave produced by a ring oscillator (Fig. 1). The frequency f_H of the RO is usually several times greater than the frequency f_L of the quartz oscillator. To achieve almost perfectly random bits in the circuit from Fig. 1 it is necessary to sample the ring oscillator at a very low frequency [29]. A D-type flip-flop is triggered by a quartz oscillator that establishes the output bit rate. Ring oscillator shown in Fig. 1 consists of inverter and delay τ connected into a ring. A delay element τ can be realized with an even number of inverters, a chain of latches or a delay line built in many FPGAs.

Ring oscillator from Fig. 1 can be simply implemented in reconfigurable logic as FPGA and CPLD. Thanks for its simplicity, ROs have become the main block in many digital systems, e.g., in voltage-controlled oscillators, serial data transmitters, temperature and voltage sensors on a chip etc. This simple structure and the underlying physical phenomena observed in the RBG from Fig. 1 have been widely studied [20]–[29], [39], [40]. Due to relatively large amount of jitter, present in generated signal, ROs are increasingly used as a source of randomness in hardware true random number generators [20]–[29], [39]. Frequency of signal generated by a single ring oscillator is equal to

$$f_H = \frac{1}{2}\sum_k \frac{1}{d_k}, \tag{1}$$

where d_k is a delay of the k-th component of RO. The expression is true if all components are ideal and delays related with interconnections are ignored. In real circuit this factor is very important and cannot be ignored [36]. Moreover, propagation delays in all circuit paths and gates vary in time, because of shot noise, thermal noise and supply voltage

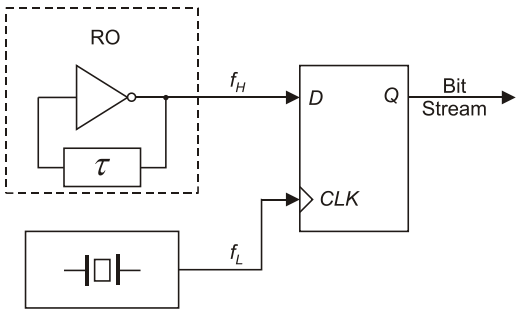

Fig. 1. Jitter oscillator sampling as a method of producing random bits.

instability [27], [39]. Having regards to this phenomenon the equation (1) must be submitted as

$$f_H = \frac{1}{2}\sum_k \frac{1}{d_k + di_k + \Delta dt_k + \Delta dv_k, + \Delta dr_k} \tag{2}$$

where d_k denotes constant delay of the k-th delay element, di_k is the delay of the k-th interconnections in ring oscillator, Δdt_k is the variation of the delay in the k-th component and interconnection caused by variation of temperature, Δdv_k represents the variation of the delay in the k-th component and interconnection caused by supply voltage instability and Δdr_k define others random delays in the k-th element and path in the ring oscillator, e.g., transition spacing or crosstalk's.

The frequency of the ring oscillator can vary in time thanks to changes of Δdt, Δdv and Δdr. This variations are called jitter. Because as central limit theorem states that many independent random variables have normal distribution, jitter in RO's follows with Gaussian distribution. Components which vary in time can be written as [27]

$$\begin{aligned} \Delta dt &= \Delta dt_n + p \cdot \Delta dt_d, \\ \Delta dv &= \Delta dv_n + p \cdot \Delta dv_d, \\ \Delta dr &= \Delta dr_n + p \cdot \Delta dr_d \end{aligned} \tag{3}$$

where Δdt_n, Δdv_n, Δdr_n are nondeterministic components and Δdt_d, Δdv_d, Δdr_d are deterministic components, the value of p is a proportion factor. The nondeterministic elements stands on nondeterministic jitter, which is an ideal source of entropy, through which is possible to produces random bits in sampling process. Both nondeterministic and deterministic jitter increases as a function of delay length and with measurement interval. This jitter accumulation is a consequence of the earlier disturbances [40]. Using equation (2), the instantaneous frequency of ring oscillator with jitter accumulation is

$$f_H = \frac{1}{2}\sum_k \frac{1}{d_k + di_k + \Delta J_a} \tag{4}$$

where

$$\Delta J_a = \sum_k \Delta dt_k, + \Delta dv_k + \Delta dr_k \tag{5}$$

is an accumulated jitter during previous half period of signal with frequency of f_H and k delay elements [27]. It can be divided into deterministic factor and nondeterministic

$$\Delta J_a = \Delta J_{an} + \Delta J_{ad}, \tag{6}$$

where

$$\Delta J_{an} = \sum_k \Delta t_{nk} + \Delta dv_{nk} + \Delta dr_{nk} \tag{7}$$

denotes an accumulated nondeterministic jitter and

$$\Delta J_{ad} = p \cdot \sum_k \Delta t_{dk} + \Delta v_{dk} + \Delta dr_{dk} \tag{8}$$

represents an accumulated deterministic jitter with proportion factor of p. As was mentioned in [40] standard deviation of nondeterministic jitter accumulated in ROs is proportional to square root of time. Deterministic jitter has standard deviation proportional to time. It follows that the deterministic jitter accumulates faster than nondeterministic one.

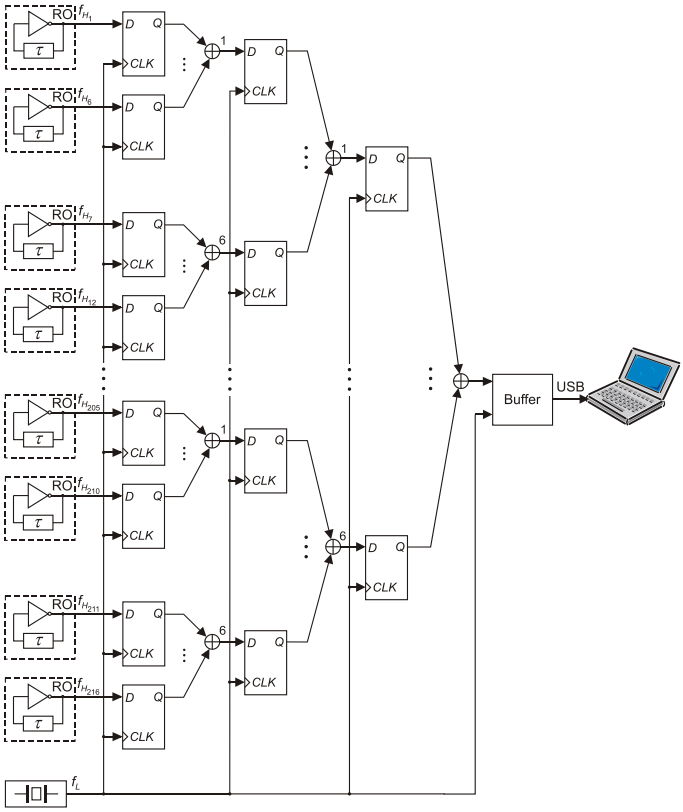

Fig. 2. Combined RO-based RBG.

TABLE I
THE FREQUENCY OF ROS IMPLEMENTED IN VIRTEX-5

K	Inverters as τ		Latches as τ	
	No. of inverters	Frequency [MHz]	No. of latches	Frequency [MHz]
1	2	735	1	667
2	4	645	2	503
3	6	303	3	299
4	8	260	4	210
5	10	164	5	207
6	12	168	6	157
7	14	164	7	130
8	16	132	8	113
9	18	129	9	115
10	20	123	10	96
11	22	113	11	82
12	24	111	12	76
13	26	106	13	72
14	28	107	14	67
15	30	81	15	63
16	32	68	16	62
17	34	51	17	57
18	36	59	18	50
19	38	55	19	49
20	40	49	20	43
21	42	32	21	37
22	44	41	22	32
23	46	23	23	26
24	48	26	24	42
25	50	23	25	12

B. Combined RO-Based RBG

The scheme of combining XOR of K with $K > 1$ bit streams produced by RO-based RBGs is shown in Fig. 2 [35]. The K element set of source generators is divided into groups. The number of elements in each group does not exceed the number n of inputs of a single lookup table (LUT). First, we combine XOR bits produced by source generators belonging to the same group. The output bits are next sampled by the same clock to form a new set of source streams. The new source streams are divided into new groups. The streams inside the same group are combined XOR, and the bits obtained are sampled by the same clock, producing the next source stream, etc. The process ends when one bit is produced. Figure 2 illustrates this process for no more than $6^3 = 216$ source generators ($n = 6$).

The combined RBG produces the first bit with a delay equal to three periods of the quartz oscillator. To assess the quality of the generated sequence, the bits are sent to a personal computer (PC) via a buffer and a USB 2.0 interface. No post-processing is performed in the PC. We assume that the source generators have frequencies $f_{H,l}$, $l = 1, 2, \ldots, K$, which are not lower than f_L, where f_L is the frequency of the quartz oscillator. The placement of all elements depends on the software provided by the manufacturer of the FPGA. We do not perform any manual corrections for the localizations proposed by this software.

The combined RBG proposed by Wold and Tan, studied by many authors, uses ROs with the same number of inverters as the delay τ. Its modification, discussed in [35] and [38] exploits as τ latches but it still assumes that the number of latches is the same for the all source generators. The differences in nominal frequencies of ROs result only from technological differences and the length of the connections between elements of each RO. Consequently, the frequencies of the rectangular waves generated by ROs may be very similar. They can also be clustered in groups, especially for a small number of inverters [31], [36]. Due to the non-ideal separation between ROs, the oscillators can synchronize each other using the phenomenon of injection synchronization discovered for the van der Pol oscillator [41]. Similar frequencies are also responsible for the success of injection attack, as described in [37]. To overcome this disadvantage, the ROs should have significantly different frequencies, and the manipulation of τ should be hard. It requires to use ring oscillators with at least different τ in a combined RBG. For example, the first RO can use two inverters or one latch, the second RO can have four inventers or two latches as τ, and so on. Table I contains the frequencies of $K = 25$ ROs implemented in Virtex-5 (XC5VLX50T) manufactured by Xilinx [42]. The values from this table are illustrated in Fig. 3. We have chosen $K = 25$ because this number of ROs was reported in the literature as sufficient to pass the statistical tests for $f_L = 100$ MHz. All the frequencies were measured with a frequency counter.

A greater τ does not always imply a smaller frequency of the RO due to delays introduced by internal connections between slices available in Virtex-5. The length of these connections depends on the placement of elements. In our case, it was determined by the software (ISE Design Suite) provided by Xilinx – the manufacturer of the FPGA. We did not perform any manual corrections for the localizations proposed by this software.

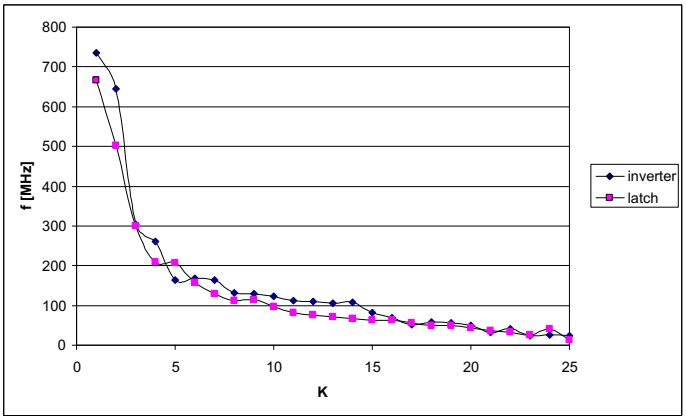

Fig. 3. The frequencies of ROs as a function of K for τ implemented as different numbers of inverters and latches [38].

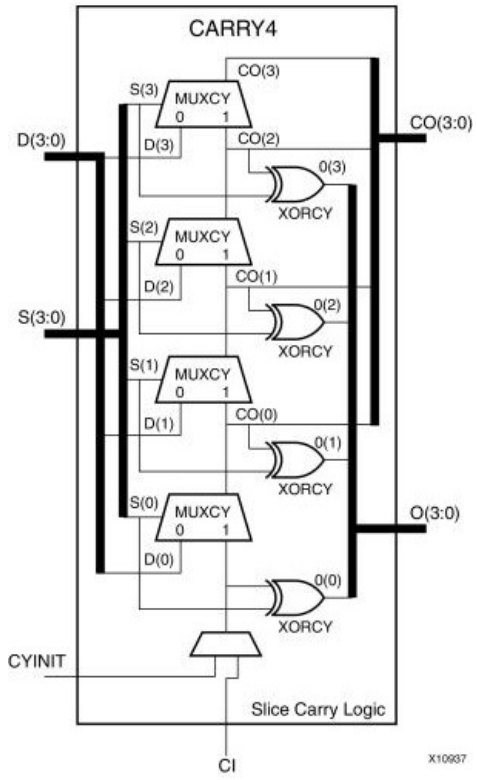

Fig. 4. Xilinx Virtex-5 fast carry chain logic primitive (Carry4) [42].

Table I indicates that the use of inverters or latches as τ is inefficient because the condition $f_H > f_L$ satisfies only 14 ROs with inverters and 9 with latches when $f_L = 100$ MHz. For $f_L = 200$ MHz, we have 4 ROs with inverters and 5 ROs with latches. The above values for K are insufficient to pass the statistical tests by the combined RBG using ROs with significantly different frequencies. We need a different source of delay with a higher granulation.

C. The Use of Carry4 as Delay τ

A series of primitives, called carry4, is available in Xilinx FPGAs [42]. It can be used as a tapped delay line with a slight delay between adjacent taps. The carry4 located in Virtex-5 slices is shown in Fig. 4. It is located in each slice of Virtex-5 FPGA. Carry4 consists of a series of four MUXes and XORs that connect to the other logic [42]. Fast carry chain logic was designed to make arithmetical functions such as adders, subtractors, and comparators faster and easier to implement. As shown in [43], carry4 is also applicable fast carry logic as delay τ in ring oscillators, configured as follows: the output of the inverter from Fig. 2 comes to carry4 input CI, and the delayed signal comes to one of the outputs CO(3...0) (Fig. 4) and to the inverter input from Fig. 1. One carry4 primitive gives four delay taps. If more delay units were required, primitives were connected in series. For ring oscillator number l, $l = 1, 2, \ldots, K$ delay was l delay units. For example, for RO number six, the sixth tap was used. Each RO had a delay line with a different delay length, and thus each RO had significantly different nominal frequency. The reason for using carry chain logic as the τ delay element is the ease of implementation and small delay per one tap, which allows the design of many high frequency ring oscillators.

The main disadvantage of this type of solution is the high area occupancy in FPGA. Nevertheless, the carry4 primitive is an excellent part of FPGA to use as a delay line in RO-based RBGs.

The frequencies of ROs for increasing values of τ are shown in Table II and illustrated in Fig. 5. Similar to inverters and latches, greater τ does not always imply a smaller frequency of the RO. Due to the greater granulation, condition $f_H > f_L$ satisfies 61 ROs for $f_L = 200$ MHz and 37 ROs for $f_L = 300$ MHz.

Consequently, the throughput can be greater than that for RBGs using ROs with different numbers of inverters or latches. The precise value depends on the results of statistical tests and the randomness of the RBG measured by the restarts mechanism.

The price of greater frequency granulation are greater resources used by the FPGA [38]. From a cryptography perspective, the amount of resources devoted to random bit generation is not critical for most of the cryptographic systems because the basic goal is security of the system.

TABLE II
THE FREQUENCIES OF ROS WITH DELAY LINES CARRY4 (VIRTEX-5)

K	Frequency [MHz]	K	Frequency [MHz]
1	639	33	339
2	739	34	400
3	599	35	282
4	499	36	402
5	461	37	262
6	462	38	348
7	382	39	331
8	433	40	264
9	418	41	251
10	454	42	375
11	417	43	360
12	355	44	242
13	373	45	221
14	375	46	239
15	400	47	235
16	391	48	229
17	366	49	231
18	341	50	310
19	368	51	234
20	359	52	226
21	383	53	242
22	353	54	224
23	358	55	222
24	370	56	171
25	330	57	219
26	293	58	223
27	325	59	179
28	327	60	214
29	294	61	207
30	328	62	210
31	324	63	274
32	292	64	187

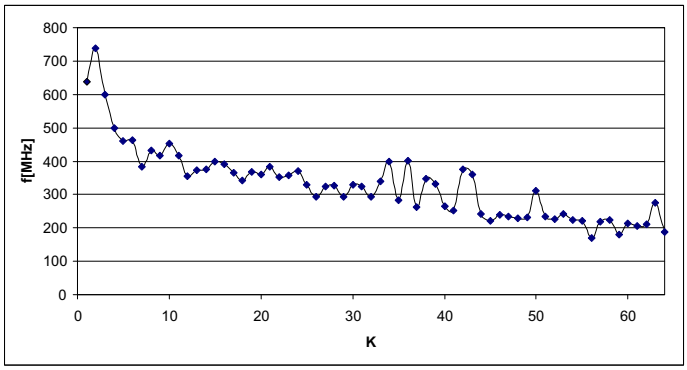

Fig. 5. The frequencies of ROs as a function of K (increasing τ).

III. ASSESSING THE AMOUNT OF RANDOMNESS AND PSEUDO-RANDOMNESS

The source of randomness for ring-oscillator-based RBGs is the accumulated jitter present in the signals of ROs. The jitter contains both noise and a deterministic component. Both components influence the statistical properties of the output sequence, but only the noise component is a source of true randomness. The deterministic component is a source of pseudo-randomness. Therefore, to decide if a source of random bits is suitable for cryptography, we must assess the amount of true randomness and pseudo-randomness present in the output streams. Two basic methods were proposed for distinguishing pseudo randomness and true randomness for ring oscillator-based RBGs. In both methods, we repeat experiments many times starting from identical initial conditions. Because pseudo-randomness is deterministic, it shows identical behavior in each repetition of the experiment. True randomness provides different behavior for identical initial conditions. To ensure identical initial conditions, we replaced the inverters shown in Figs. 1 and 2 with NAND gates triggered by an external signal. In the method described in [26], the authors recorded 1000 restarts of a ring oscillator, Fibonacci ring oscillator (FIRO) and Galois ring oscillator (GARO). For a given oscillator, the curves diverge from each other quickly. The amount of true randomness in the curves obtained was measured by the computation of the standard deviation of the output voltage as a function of time. If this standard deviation was relatively large, then extracting one bit of true randomness by sampling was easy and reliable [26]. The second method does not assess true randomness contained in the output of a single ring oscillator, but it checks the quality of sequences obtained for successive positions of a bit in time sequences produced by restarts [34], [35]. Therefore, this method is more suitable for assessing the randomness of RBGs containing more than one RO. The method checks, with the chi-square test, the distribution of bits in sequences obtained for successive positions of a bit in time sequences produced by restarts. This approach was first proposed in paper [34] and described in detail in [35]. In the experiments, 20000 bits were produced for one restart. The number of restarts was equal to 2048, i.e., it was the same as in [35]. It yields 20000 sequences with $N = 2048$ elements each. If, among the 20000 sequences, there are one or two successive sequences that fail the chi-square test, these failures may be produced by a perfect random source [35]. The value of statistic χ^2 is computed as

$$\chi^2 = \frac{(N_0 - N \cdot P_0)^2}{N \cdot P_0} + \frac{(N_1 - N \cdot P_1)^2}{N \cdot P_1}, \tag{9}$$

where N_0 is the number of zeros in an N-bit sequence and N_1 is the number of ones in this sequence. $N \cdot P_0$ is the expected number of zeros and $N \cdot P_1$ is the expected number of ones. Because $N \cdot P_0 = N \cdot P_1 = N/2$ equation (9) becomes [35]

$$\chi^2 = \frac{\Delta^2}{N}, \tag{10}$$

where Δ is the difference between the number of zeros and ones. For a significance level of $\alpha = 0.01$, the critical χ^2_c value is equal to 6.635. If the value of statistic (10) is smaller than 6.635, there is no reason to reject the hypothesis that zeros and ones occur in the m-th, N-element sequence with the same probability. A sequence composed of $N = 2048$ bits passes the chi-square test if and only if $\Delta < 116.5696\ldots$.Thus, the difference between the number of zeros and ones cannot exceed 116. A 2048-bit sequence does not pass the chi-square test if and only if $\Delta \geq 117$. The probability that the number of zeros differs from the number of ones by at least 117 is $p = 1 - p'$, and p' is the probability that the number of zeros differs from the number of ones by no more than 116. Because N is even, Δ can be only even, including $\Delta = 0$. For a perfect random source, the probability p' can be computed from the

following formula [35]

$$p' = \frac{1}{2^{2048}}[\mathrm{C}_{2048}^{(1024)} + \sum_{i=1}^{58}(\mathrm{C}_{2048}^{(1024-i)} + \mathrm{C}_{2048}^{(1024+i)})] = \\ = 0.990289\ldots \tag{11}$$

Because $p = 1 - p'$ it is also that $p = 0.009711\ldots$. If 2048-bit sequences are equally probable, they fail the chi-square test statistically every $1/p \approx 103$ sequences. Two successive sequences do not pass the same test every $1/p^2 \approx 10604$ sequences and three successive sequences fail every $1/p^3 \approx 10^6$ sequences. Because 10^6 is significantly greater than 20000, among 20000 chi-square test results, there is at most one such case for a perfectly random source. If the sequences are produced by a nonrandom source, 2048-bit sequences fail the chi-square test for three successive values of m, $m = 1, 2, \ldots, 20000$ many times. A computer program searches the results of 20000 chi-square tests for the greatest m for which the 2048-bit sequences fail the chi-square test for m, $m-1$ and $m-2$. For $j > m$ and a given significance level of the test, there is no reason to reject the hypothesis that zeros and ones occur with the same probability in successive 2048-bit sequences. Because the equal probability of zero and one in a 2048-bit sequence produced for any $j > m$ is impossible for a deterministic system, we assume that the elements of this sequence are produced by a random source. The smallest j is equal to $m + 1$ and denoted by m_{min}. If we select every j-th bit from the original sequence as the output, where $j < m_{min}$, we obtain a sequence with initial bits similar to those previously generated by repeating the generation. To avoid this result, we must select every j-th bit with j satisfying $j \geq m_{min}$. We emphasize that the repetition of the generation is typical in cryptography because random numbers are produced when they are needed. The continuous generation of random numbers leads to an undesirable waste of power and simplifies attacks against the RBG. Figures 6–8 present the values of m_{min} for the combined RBG with τ implemented as two inverters, one latch and a delay line with different values [38].

The shapes of the curves in Figs. 6–8 are similar. For all sampling frequencies, the value for m_{min} decreases irregularly with the increase of K. With more source generators, we increase the randomness of the output sequence. For small K, large differences in randomness between combined RBGs with different realizations of τ can be observed for all sampling frequencies. If K is large, a combined RBG with inverters and latches offers similar randomness. The combined RBG using ROs with the delay τ realized with carry chain logic is worse for all f_L. Simultaneously, the same generator produces sequences that pass NIST tests for all sampling frequencies and K significantly smaller than that reported in the literature [38]. It basically indicates that a deterministic component of the accumulated jitter is responsible for the very good statistical properties at the output of this type combined RBG. It also confirms the results of N. Bochard *et al.* that the method of Wold and Tan can provide sequences that pass all statistical tests when ring oscillators do not exhibit any jitter [33]. The result above does not eliminate this generator

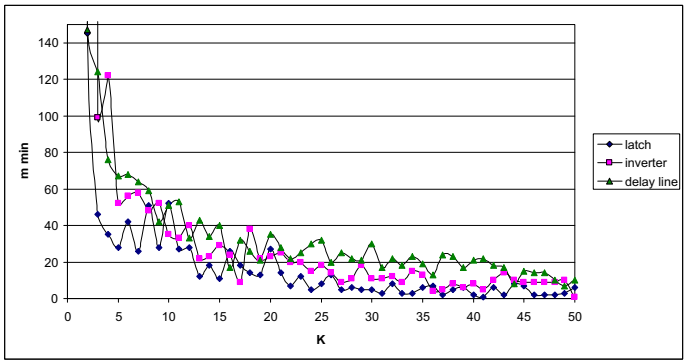

Fig. 6. The values of m_{min} for RBGs from Fig. 2 and three realizations of the delay τ, $f_L = 100$ MHz.

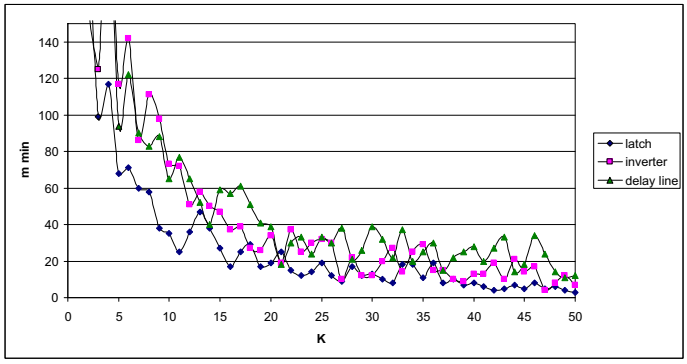

Fig. 7. The values of m_{min} for RBGs from Fig. 2 and three realizations of the delay τ, $f_L = 150$ MHz.

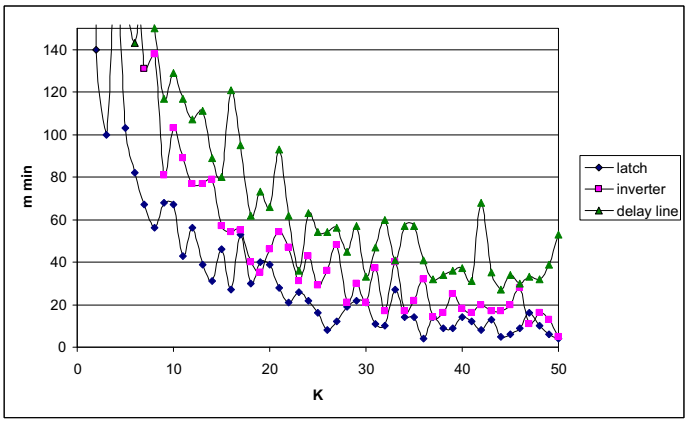

Fig. 8. The values of m_{min} for RBGs from Fig. 2 and three realizations of the delay τ, $f_L = 200$ MHz.

from cryptographic applications. To obtain a random sequence, we choose every j-th element of the sequence produced by the RBG described, where $j \geq m_{min}$. For example, if the combined RBG uses 50 source generators, it is sufficient to choose every 10^{th} bit to obtain a random sequence for $f_L = 100$ MHz. It reduces the bit rate to 10 Mbit/s, but the ROs do not synchronize each other, and the frequency injection attack is significantly hampered.

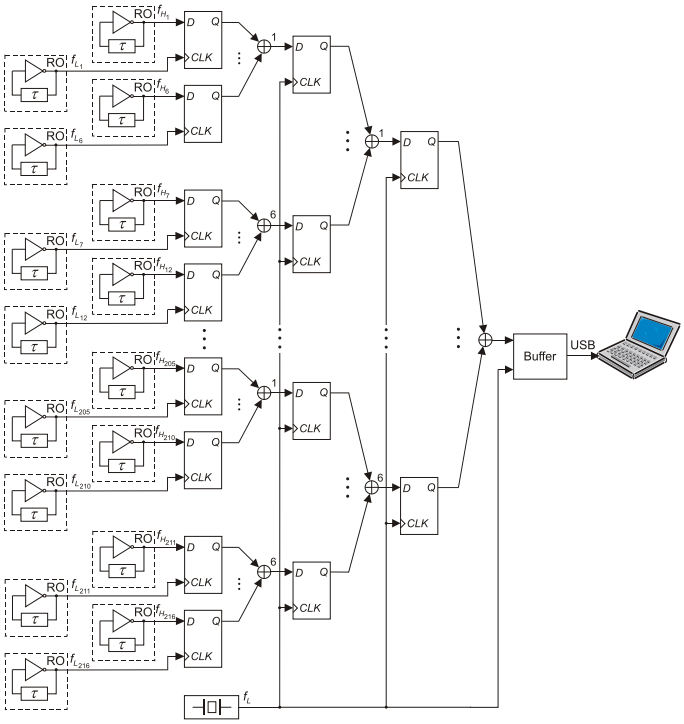

Fig. 9. Combined RO-based RBG with individual sources of sampling frequency.

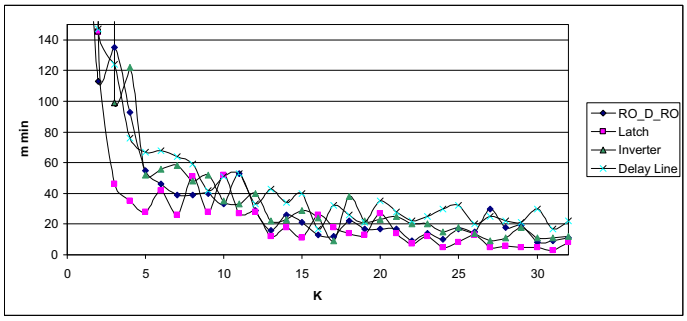

Fig. 10. The values of m_{min} for RBGs from Fig. 9 and four realizations of the delay τ, $f_L = 100$ MHz.

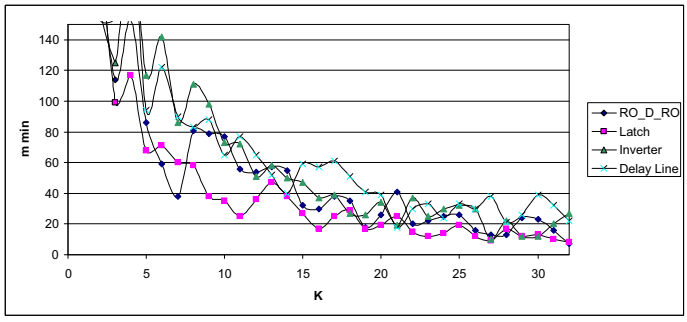

Fig. 11. The values of m_{min} for RBGs from Fig. 9 and four realizations of the delay τ, $f_L = 150$ MHz.

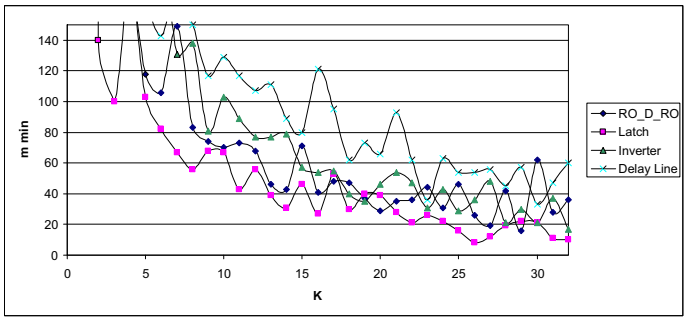

Fig. 12. The values of m_{min} for RBGs from Fig. 9 and four realizations of the delay τ, $f_L = 200$ MHz.

IV. Enhancing the Randomness of a Combined RBG

Although it is expected that RO-based RBG with delay line is significantly more resistant to injection attacks and to the environmental changes compared to solutions described in the literature, its throughput may not be satisfactory in some applications. To increase the value of this parameter, we have to decrease m_{min}. One of the possible methods which can be applied in FPGAs is the change of the source of sampling frequency f_L from quartz oscillator to ring oscillator which offers larger jitter. Because such change does not significantly decrease m_{min}, we have assumed that the output of each RO is sampled by another ring oscillator with significantly smaller frequency, assigned to this RO (Fig. 9). We used pairs of ROs from Table II. The output of oscillator 1 is sampled by oscillator 33, the output of oscillator 2 is sampled by oscillator 34, etc. Generally, the output of the l-th oscillator, is sampled by the $(l + K)$ RO. The values of of m_{min} obtained for modified combined RBG (curve RO_D_RO) and for the combined RBG from Fig. 2, using delay τ implemented as two inverters, one latch and a delay line with different values, are shown in Figs. 10–12.

Comparing RNGs with carrry4, we see that the use of additional ROs decreases m_{min} for almost all K. The greatest gain is observed for the highest sampling frequency and K between 8 and 22. This is particularly promising because the smallest K that ensures at the output sequences that pass all statistical tests is also relatively small [38]. The results of tests obtained for combined RBGs with inverters and latches will not be further considered because of the sensitivity of RBGs to the injection attack.

V. The Statistical Properties of Bit Sequences Produced by a Combined RBG Using ROs with Different Delays

To assess the statistical properties of the binary sequences produced by the combined RBG using the dedicated delay lines and additional ROs as a source of sampling frequencies, we used the statistical tests described in "A statistical Test Suite for Random and Pseudo-Random Number Generators for Cryptographic Applications", document 800-22 prepared by the National Institute of Standards and Technology (NIST) [44]. These tests are often referred to as the NIST 800-22 statistical test suite or, simply, the NIST 800-22 tests. During testing, we applied two approaches proposed by NIST: (1) we examined the proportion R_β of sequences that passed a statistical test, and (2) we examined the distribution of P-values computed by the software; that is, we examined the value of P_T [44]. We also assumed the standard set of

TABLE III
THE RESULTS OF THE NIST 800-22 TESTS FOR COMBINED RBG; $f_L = 100$ MHz

Type of test	$K = 12$		$K = 15$	
	R_β	P_T	R_β	P_T
Frequency	0.9890	0.04010	0.9970	0.35048
Block Frequency	0.9880	0.12962	0.9920	0.62254
Cumulative Sums*	0.9910	0.50619	0.9940	0.59347
Runs	0.9930	0.50421	0.9920	0.39072
Longest Run of Ones	0.9870	0.71567	0.9910	0.72380
Rank	0.9900	0.04450	0.9860	0.50027
Spectral DFT	0.9910	0.22364	0.9890	0.45405
Non-overlapping Temp.*	0.9830	0.05128	0.9830	0.00624
Overlapping Templates	0.9890	0.62869	0.9870	0.22836
Universal	0.9860	0.79439	0.9880	0.06087
Approximate Entropy	0.9910	0.89917	0.9910	0.82727
Random Excursions*	0.9853	0.10282	0.9852	0.15748
Random Exc. Var.**	0.9853	0.04897	0.9869	0.03260
Serial*	0.9870	0.20773	0.9820	0.21900
Linear Complexity	0.9950	0.09542	0.9930	0.62254

TABLE IV
THE RESULTS OF THE NIST 800-22 TESTS FOR COMBINED RBG; $f_L = 150$ MHz

Type of test	$K = 12$		$K = 15$	
	R_β	P_T	R_β	P_T
Frequency	0.9900	0.23803	0.9920	0.46923
Block Frequency	0.9900	0.40649	0.9890	0.24050
Cumulative Sums*	0.9910	0.46541	0.9910	0.06563
Runs	0.9950	0.48270	0.9960	0.81472
Longest Run of Ones	0.9830	0.85304	0.9900	0.41902
Rank	0.9900	0.84293	0.9900	0.08610
Spectral DFT	0.9880	0.18656	0.9850	0.11204
Non-overlapping Temp.*	0.9820	0.02641	0.9830	0.02214
Overlapping Templates	0.9910	0.38211	0.9880	0.75384
Universal	0.9880	0.36858	0.9930	0.10561
Approximate Entropy	0.9930	0.08201	0.9930	0.20773
Random Excursions*	0.9869	0.05910	0.9887	0.28352
Random Exc. Var.**	0.9820	0.02477	0.9823	0.04896
Serial*	0.9890	0.48853	0.9870	0.34886
Linear Complexity	0.9900	0.09833	0.9880	0.88166

TABLE V
THE RESULTS OF THE NIST 800-22 TESTS FOR COMBINED RBG; $f_L = 200$ MHz

Type of test	$K = 12$		$K = 15$	
	R_β	P_T	R_β	P_T
Frequency	0.9890	0.07525	0.9930	0.27846
Block Frequency	0.9890	0.06087	0.9910	0.70341
Cumulative Sums*	0.9850	0.06087	0.9940	0.16170
Runs	0.9890	0.84122	0.9890	0.07905
Longest Run of Ones	0.9900	0.31154	0.9910	0.28402
Rank	0.9910	0.48658	0.9860	0.78110
Spectral DFT	0.9890	0.83256	0.9920	0.94829
Non-overlapping Temp.*	0.9830	0.00731	0.9820	0.00436
Overlapping Templates	0.9890	0.99425	0.9860	0.43359
Universal	0.9900	0.88616	0.9890	0.36028
Approximate Entropy	0.9900	0.75581	0.9900	0.17958
Random Excursions*	0.9838	0.01761	0.9825	0.01862
Random Exc. Var.**	0.9838	**0.00000**	0.9857	0.16628
Serial*	0.9900	0.24549	0.9930	0.84464
Linear Complexity	0.9890	0.89776	0.9870	0.74789

parameters proposed by NIST in v. 1.8. The significance level was $\beta = 0.01$. The frequency f_L was equal to 100 MHz, 150 MHz, and 200 MHz. For greater f_L, e.g., equal to 250 MHz, delays between elements of the cascade from Fig. 1 are greater than the period of the sampling frequency, and the whole structure stops working properly. The results of the tests performed for the three sampling frequencies mentioned are shown in Tables III–V. The minimum passing value for the standard set of parameters was approximately 0.9805. The minimum P_T value was 0.0001. An asterisk * denotes that this test consists of several subtests and that the worst result is shown. For tests marked with **, the minimum passing value for the standard set of parameters was approximately 0.9777.

The sequences fail only random-excursions-variant test for the P_T value, $f_L = 200$ MHz and $K = 12$. The all tests are satisfied for f_L equal to 100 MHz and 150 MHz for $K \geq 12$, which is significantly better result than that reported in the literature [31], [32], [34], [35], [38]. Perfect results of statistical tests suggest that the choice of greater frequency f_L is better because of the greater throughput. This is only partially true because the throughput is indeed greater, but the true randomness of the output bits can be smaller. For example, if the combined RBG uses 15 source generators, it is sufficient to choose every 22^{nd} bit to obtain a random sequence for $f_L = 100$ MHz. It reduces the bit rate to approximately 4.54 Mbit/s. For $f_L = 200$ MHz, we obtain $m_{min} = 72$ and approximately 2.77 Mbit/s. On the other hand, if we choose $K = 20$, we have $\sim$5.88 Mbit/s for $f_L = 100$ MHz ($m_{min} = 17$) and $\sim$7.14 Mbit/s for $f_L = 200$ MHz ($m_{min} = 28$). In the both cases the throughput obtained for the combined RBG with additional ROs is significantly greater than the throughput observed for the same combined RBG but without additional ROs.

The increase of K over 32 is very limited because of the condition $f_H > f_L$. We can decrease the sampling frequency f_L below 100 MHz but it decreases the output bit rate. If the resources of FPGA are not critical the optimal choice seems to be generator from Fig. 2 with K greater than 80. In this case, we still have that $f_H > f_L = 100$ MHz but m_{min} does not exceed 5. It yields the efficient bit rate at the output of the combined RBG not smaller than 20 Mbit/s for Virtex-5.

VI. CONCLUSION

Combining XOR bits produced at the same time by many independent random number generators is an efficient method for producing random sequences that pass every statistical test. This method requires relatively large resources, and excellent statistical properties can be observed for both deterministic and nondeterministic systems. From a cryptography perspective the elements of output sequence should be unpredictable which require the use of a physical source of randomness. Although the FPGAs are deterministic circuits, we can find phenomena, like jitter in ring oscillators, which can be used to produce unpredictable sequences. Such generator was proposed by Sunar *et. al.* and next modified by Wold and Tan. The use of delay lines as τ in the generator of Wold and Tan has not been considered in the literature. To apply sequences produced by this generator in cryptography, we have to choose only certain bits of the original sequence as output. The throughput is significantly reduced but it is still greater than for the generator of Wold and Tan. It is also expected that RBGs described

in this paper are more robust to injection attack than known solutions of RBGs using ROs. The latter is guaranteed by two elements: the realization of τ with carry chain logic which is robust to different cryptographic attacks, e.g., to power attack, and the distribution of frequencies of ROs over a very wide range. The use of delay lines with different delays for different ROs hampers synchronization between ROs implemented in the same FPGA preventing the quality degradation of RO-based combined RBGs.

The generators described in this paper can be an interesting alternative for cryptographic systems for which the resources are not critical. They can be used, e.g., as generators of initialization vectors for block ciphers, in authentications schemes, as generators of salts for passwords storage or as generators of seeds for pseudo-random number generators. When a cryptographic system is implemented as ASIC, better performance of the RBG is expected. A designer can control some parameters, e.g., the length of internal connections, which is very difficult for FPGAs. Consequently, he/she can ensure better reproducibility of the RBG parameters, smaller correlation between ring oscillators and better robustness to cryptographic attacks.

The goal of future research can be methods of decreasing m_{min} for a given resources. One of the promising technique seems to be compression of bits produced in a time window sliding along the generated bits. This method requires fast and parallel processing. Another method which can potentially decrease m_{min} is the use of another phenomena observed in digital circuits, e.g., the metastable states.

REFERENCES

[1] W. T. Holman, J. A. Connelly, and A. B. Downlatabadi, "An integrated analog/digital random noise source," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 44, pp. 521–528, June 1997.

[2] V. Bagini and M. Bucci, "A Design of reliable true random number generator for cryptographic applications," in *Proceedings of Workshop Cryptographics Hardware Embedded Systems, CHES'1999*, Heidelberg, 1999, pp. 204–218, LNCS 1717.

[3] C. S. Petrie and J. A. Connelly, "The sampling of noise for random generation," *Proceedings of the 50th International Symposium on Circuits and Systems ISCAS'1999*, vol. 6, pp. 26–29, 1999.

[4] M. Bucci, L. Germani, R. Luzzi, P. Tommasimo, A. Trifiletti, and M. Varanonuovo, "A high-speed IC random-number source for smartcard microcontrollers," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 50, pp. 1373–1380, November 2003.

[5] J. Holleman, S. Bridges, B. P. Otis, and C. Diorio, "A 3 μw cmos true random number generator with adaptive floating-gate offset cancellation," *IEEE Journal of Solid-State Circuits*, vol. 43, pp. 1324–1336, May 2008.

[6] M. J. Bellido et al., "A simple binary random number generator: New appoaches for CMOS VLSI," *Proceedings of the 35th Midwest Symposium on Circuits and Systems*, vol. 1, pp. 127–129, 1992.

[7] M. Epstein, L. Hars, R. Krasinski, M. Rosner, and H. Zheng, "Design and implementation of a true random number generator based on digital circuit artifacts," in *Proceedings of Workshop Cryptographics Hardware Embedded Systems, CHES'2003*, 2003, pp. 152–165, LNCS 2779.

[8] I. Vasyltsov, E. Hambardzumyan, Y.-S. Kim, and B. Karpinskyy, "Fast digital RBG based on metastable ring oscillator," in *Proceedings of Workshop Cryptographics Hardware Embedded Systems, CHES'2008*, 2008, pp. 164–180, LNCS 5154.

[9] S. Srinivasan, S. Mathew, V. Erraguntla, and R. Krishnamurthy, "A 4Gbps 0.57pJ/bit Process-Voltage-Temperature Variation Tolerant all-Digital True Random Number Generator in 45nm CMOS," in *Proceedings of 22nd International Conference on VLSI Design*, 2009, pp. 301–306.

[10] M. Varchola and M. Drutarovsky, "New high entropy element for FPGA based true random number generators," in *Proceedings of Workshop Cryptographics Hardware Embedded Systems, CHES'2010*, 2010, pp. 351–365, LNCS 6225.

[11] G. Bernstein and M. A. Lieberman, "Secure random number generation using chaotic circuits," *IEEE Transactions on Circuits and System*, vol. 37, pp. 1157–1164, Spetember 1990.

[12] R. Bernardini and G. Cortelazzo, "Tools for designing chaotic systems for secure random number generation," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 48, pp. 552–564, May 2001.

[13] T. Stojanovski and L. Kocarev, "Chaos-based random number generators – Part I: Analysis," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 48, pp. 281–288, March 2001.

[14] T. Stojanovski, J. Pihl, and L. Kocarev, "Chaos-based random number generators – Part II: Practical realization," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 48, pp. 382–385, March 2001.

[15] M. E. Yalçin, J. A. K. Suykens, and J. Vandewalle, "True bit generation from a double-scroll attractor," *IEEE Transactions on Circuits and Systems I, Regular Papers*, vol. 51, pp. 1395–1404, July 2004.

[16] S. Callegari, R. Rovatti, and G. Setti, "Embeddable ADC-based true random number generator for cryptographic applications exploiting nonlinear signal processing and chaos," *IEEE Transactions on Signal Processing*, vol. 53, pp. 793–805, February 2005.

[17] ——, "First direct implementation of true random source on programmable hardware," *International Journal of Circuit Theory and Applications*, vol. 33, pp. 1–16, 2005.

[18] M. Drutarovsky and P. Galajda, "Chaos-based true random number generator embedded in a mixed-signal reconfigurable hardware," *Journal of Electrical Engineering*, vol. 57, pp. 218–225, April 2006.

[19] S. Ergün and S. Özog, "Truly random number generator based on a non-autonomous chaotic oscillator," *International Journal of Electronics and Communications*, vol. 61, pp. 235–242, April 2007.

[20] C. S. Petrie and J. A. Connelly, "Modelling and simulation of oscillator-based random number generators," *Proceedings of the 47th International Symposium on Circuits and Systems ISCAS'1996*, vol. 4, pp. 324–327, 1996.

[21] C. S. Petrie and J. L. Connelly, "A noise-based IC rndom number generator for applications in Cryptography," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 47, pp. 615–621, May 2000.

[22] M. Bucci, L. Germani, R. Luzzi, A. Trifiletti, and M. Varnonuovo, "A high-speed oscillator-based truly random number source for cryptographic applications on a smartcard IC," *IEEE Transactions on Computers*, vol. 52, pp. 403–409, April 2003.

[23] B. Jun and B. Kocher, *The Intel random number generator*. San Francisco, CA: Cryptography Research Inc., April 1999, white paper for Intel Corp. Available at: http://www.cryptography.com/resources,whitepapers/IntelRNG.pdf.

[24] P. Kohlbrenner and K. Gay, "An embedded true random number generator for FPGAs," in *Proceedings of the 2004 ACM/SIGDA 12th International Symposium on FPGAs, FPGA'04*, 2004, pp. 71–77.

[25] J. D. Golić, "New methods for digital generation and postprocessing of random data," *IEEE Transactions on Computers*, vol. 55, pp. 1217–1229, October 2006.

[26] M. Dichtl and J. D. Golić, "High speed true random number generation with logic gates only," in *Proceedings of Workshop Cryptographics Hardware Embedded Systems, CHES'2007*, 2007, pp. 45–62, LNCS 4727.

[27] B. Valtchanov, A. Aubert, F. Bernard, and V. Fischer, "Modeling and observing the jitter in ring oscillators implemented in FPGAs," in *Proceedings of IEEE Workshop on Design and Diagnostics of Electronic Circuits and Systems, DDECS'08*, 2008, pp. 1–6.

[28] B. Valtchanov, V. Fischer, A. Aubert, and F. Bernard, "Characterization of randomness sources in ring oscillator-based true random number generators in FPGAs," in *Proceedings of IEEE Workshop on Design and Diagnostics of Electronic Circuits and Systems, DDECS'10*, 2010, pp. 48–53.

[29] M. Baudet, D. Lubicz, J. Micolod, and A. Tassiaux, "On the security of oscillator-based random number generators," *Journal of Cryptology*, vol. 24, pp. 398–425, 2011.

[30] B. Sunar, W. J. Martin, and D. R. Stinson, "A provably secure true random number generator with built-in tolerance to active attacks," *IEEE Transactions on Computers*, vol. 56, pp. 109–119, January 2007.

[31] K. Wold and C. H. Tan, "Analysis and enhancement of random number generator in FPGA based on oscillator rings," *International Journal of Reconfiugurable Computing*, vol. 2009, pp. 1–8, 2009.

[32] K. Wold and S. Petrović, "Optimizing speed of a true random number generator in FPGA by spectral analysis," in *Proceedings of Fourth International Conference on Computer Sciences and Convergence Information Technology, ICCIT'09*, 2009, pp. 1105–1110.

[33] N. Bochard, F. Bernard, and V. Fischer, "Observing the randomness in RO-based RBG," in *Proceedings of International Conference on Reconfigurable Computing and FPGAs, ReConFig 2009*, 2009, pp. 237–242.

[34] M. Jessa and M. Jaworski, "Randomness of a combined RBG based on the ring oscillator sampling method," in *Proceedings of International Conference on Signals and Electronic Systems, ICSES'10*, 2010, pp. 323–326.

[35] M. Jessa and L. Matuszewski, "Enhancing the Randomness of a Combined True Random Number Generator Based on the Ring Oscillator Sampling Method," in *Proceedings of International Conference on ReConFigurable Computing and FPGAs, ReConFig'2011*, 2011, pp. 274–279.

[36] K. Wold and S. Petrović, "Security properties of oscillator rings in true random number generators," in *Proceedings of 15th International Symposium on Components, Circuits, Devices and Systems*, 2012, pp. 145–150.

[37] A. T. Markettos and S. M. Moore, "The frequency injection attack on ring-oscillator-based true random number generators," in *Proceedings of Workshop Cryptographics Hardware Embedded Systems, CHES'2009*, September 2009, pp. 317–331, LNCS 5747.

[38] M. Jessa and L. Matuszewski, "The Use of Delay Lines in a Ring-Oscillator-Based Combined True Random Number Generator," in *Proceedings of International Conference on Signals and Electronic Systems, ICSES'12*, 2012, article 51.

[39] Ü. Güler, S. Ergün, and G. Dündar, "A digital IC random number generator with logic gates only," in *Proc. of 17th IEEE International Conference on Electronics, Circuits, and Systems (ICECS)*, December 2010, pp. 239–242.

[40] A. Hajimiri, S. Limotyraksis, and T. H. Lee, "Jitter and Phase Noise in Ring Oscillators," *IEEE Journal of Solid-State Circuits*, vol. 34, no. 6, June 1999.

[41] J. Guckenheimer and P. Holmes, *Nonlinear Oscillations, Dynamical Systems, and Bifurcation of Vector Fields*. New York: Springer, 1983.

[42] "Virtex-5 Libraries Guide for HDL Designs," http://www.xilinx.com/itp/xilinx10/books/docs/virtex5_hdl/virtex5_hdl.pdf.

[43] C. Favi and E. Charbon, "A 17ps Time-to-Digital Converter Implemented in 65nm FPGA Technology,," in *Proceedings of the ACM/SIGDA international symposium on Field programmable gate arrays, FPGA'09*, Monterey, California, USA, February 2009.

[44] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray, and S. Vo, *A statistical test suite for random and pseudorandom number generators for cryptographic applications*. NIST special publication 800-22, Revised: April 2010, USA, Available at: http://csrc.nist.gov/rng/.