Hindawi Publishing Corporation
International Journal of Reconfigurable Computing
Volume 2010, Article ID 879281, 13 pages
doi:10.1155/2010/879281

Research Article

True-Randomness and Pseudo-Randomness in Ring Oscillator-Based True Random Number Generators

Nathalie Bochard, Florent Bernard, Viktor Fischer, and Boyan Valtchanov

CNRS, UMR5516, Laboratoire Hubert Curien, Université de Lyon, 42000 Saint-Etienne, France

Correspondence should be addressed to Nathalie Bochard, nathalie.bochard@univ-st-etienne.fr

Received 26 February 2010; Revised 22 September 2010; Accepted 12 December 2010

Academic Editor: Lionel Torres

The paper deals with true random number generators employing oscillator rings, namely, with the one proposed by Sunar et al. in 2007 and enhanced by Wold and Tan in 2009. Our mathematical analysis shows that both architectures behave identically when composed of the same number of rings and ideal logic components. However, the reduction of the number of rings, as proposed by Wold and Tan, would inevitably cause the loss of entropy. Unfortunately, this entropy insufficiency is masked by the pseudo-randomness caused by XOR-ing clock signals having different frequencies. Our simulation model shows that the generator, using more than 18 ideal jitter-free rings having slightly different frequencies and producing only pseudo-randomness, will let the statistical tests pass. We conclude that a smaller number of rings reduce the security if the entropy reduction is not taken into account in post-processing. Moreover, the designer cannot avoid that some of rings will have the same frequency, which will cause another loss of entropy. In order to confirm this, we show how the attacker can reach a state where over 25% of the rings are locked and thus completely dependent. This effect can have disastrous consequences on the system security.

1. Introduction

True Random Number Generators (TRNGs) are used to generate confidential keys and other critical security parameters (CSPs) in cryptographic modules [1]. Generation of high rate and high quality random bit-stream inside logic devices is difficult because these devices are intended for implementing deterministic data processing algorithms whereas generating true-randomness needs some physical nondeterministic process.

The quality of the generated bit-streams is evaluated using dedicated statistical tests such as FIPS 140-2 [1], NIST 800-22 [2], and Diehard [3]. However, the statistical tests are not able to give a mathematical proof that the generator generates true random numbers and not only pseudo-random numbers that can be employed in attacks [4]. For this reason, Killmann and Schindler [5] propose to characterize the source of randomness from the raw binary signal in order to estimate the entropy in the generator output bit-stream.

The TRNGs implemented in reconfigurable devices usually use metastability [6–8] or the clock jitter [9–12] as the source of randomness. Many of them employ ring oscillators (ROs) as a source of a jittery clock [13–15]. One of principles employed the most frequently in reconfigurable devices is that proposed by Sunar et al. in [15]. This principle was later exploited and modified in [16] and enhanced in [17, 18]. Sources of randomness and randomness extraction in RO-based TRNG were analyzed in [19–21].

In order to increase the entropy of the generated binary raw signal and to make the generator "provably secure", Sunar et al. employ a huge number of ROs [15]. The outputs of 114 supposedly independent ROs are XOR-ed and sampled using a reference clock with a fixed frequency in order to obtain a raw binary signal. This binary signal is then postprocessed using a resilient function depending on the

size of the jitter and the number of ROs employed. The main advantages of the generator of Sunar are

(i) the claimed security level based on a security proof,

(ii) easy (almost "push button") implementation in FPGAs.

Without the security proof, the generator of Sunar et al. can be considered as just one of many existing TRNGs that passes the statistical tests. This security approach is essential for TRNG evaluation according to AIS31 [5] that is accepted as a de facto standard in the field. Unfortunately, the Sunar's security proof is based on at least two assumptions that are impossible or difficult to achieve and/or validate in practice [11]:

(i) the XOR gate is supposed to be infinitely fast in order to maintain the entropy generated in rings;

(ii) the rings are supposed to be independent.

Wold and Tan show in [18] that by adding a flip-flop to the output of each oscillator (before the XOR gate), the generated raw bit-stream will have better statistical properties: the NIST [2] and Diehard [3] tests will pass without postprocessing and with a significantly reduced number of ring oscillators.

It is commonly accepted that contrary to the original design of Sunar et al., the modified architecture proposed by Wold and Tan maintains the entropy of the raw binary signal after the XOR gate if the number of rings is unchanged. However, we believe that several other questions are worthy of investigation. The aim of our paper is to find answers to the following questions and to discuss related problems:

(i) Is the security proof of Sunar valid also for the generator of Wold and Tan?

(ii) What is the entropy of the generated bitstream after the reduction of number of rings?

(iii) How does security enhancement proposed by Fischer et al. in [17] modify the quality of the generated binary raw signal?

(iv) How should the relationship between the rings be taken into account in entropy estimation?

The paper is organized as follows: Section 2 analyzes the composition of the timing jitter of the clock signal generated in ring oscillators. Section 3 deals with the simulation and experimental background of our research. Section 4 compares the behavior of the two generators in simulations and in hardware. Section 5 discusses the impact of the size and type of the jitter on the quality of the raw bit-stream. Section 6 evaluates the dependence between the rings inside the device and its impact on generation of random bit-stream. Section 7 discusses the obtained results and replies to the questions given in the previous paragraph. Section 8 concludes the paper.

2. Ring Oscillators and Timing Jitter

Ring oscillators are free-running oscillators using logic gates. They are easy to implement in logic devices and namely in Field Programmable Gate Arrays (FPGAs). The oscillator consists of a set of delay elements that are chained into a ring. The set of delay elements can be composed of inverting and noninverting elements, while the number of inverting elements has to be an odd number. The period of the signal generated in the RO using ideal components is given by the form

$$T = 2\sum_{i=1}^{k} d_i, \tag{1}$$

where $k \in \{3, 4, 5, \ldots, n\}$ is the number of delay elements and d_i is the delay of the i-th delay element. This expression is simplified in two ways:

(i) the delay d_i is supposed to be constant in time;

(ii) the delays of interconnections are ignored.

In physical devices, the delay d_i varies between two half-periods (i.e., between two instances i and $i+k$) and expression (1) gets the form

$$T = \sum_{i=1}^{2k} d_{(i-1 \bmod k)+1}. \tag{2}$$

In the case of ring oscillators, the variation of the clock period is observed as the clock timing jitter, which can be seen as a composition of the jitter caused by local sources and the jitter coming from global sources, usually from power supply and/or global device environment [19]. When observing rings implemented in real devices and namely, in FPGAs, the delays of interconnections cannot be ignored anymore. For simplicity, we propose to merge them with the gate delays. This approach was validated in [21]. The delay d_i of gate i including interconnection delay between two consecutive gates in the ring oscillator can then be expressed as

$$d_i = D_i + \Delta d_{Li} + \Delta d_{\mathrm{G}i}, \tag{3}$$

where D_i is a constant delay of gate i plus interconnection between gates i and $i + 1 \pmod k$ corresponding to the nominal supply voltage level and nominal temperature of the logic device, Δd_{Li} is the delay variation introduced by local physical events, and $\Delta d_{\mathrm{G}i}$ is the variation of the delay caused by global physical sources such as substrate noise, power supply noise, power supply drifting, and temperature variation.

The delay d_{Li} is dynamically modified by some amount of a random signal $\Delta d_{\mathrm{LG}i}$ (LG-Local Gaussian jitter component) and by some local cross-talks from the neighboring circuitry $\Delta d_{\mathrm{LD}i}$ (LD-Local Deterministic jitter component). The jitter from local sources used in (3) can thus be expressed as

$$\Delta d_{Li} = \Delta d_{\mathrm{LG}i} + \Delta d_{\mathrm{LD}i}. \tag{4}$$

The local Gaussian jitter components $\Delta d_{\mathrm{LG}i}$ coming from individual gates and interconnections are characterized by normal probability distribution $N(\mu_i, \sigma_i^2)$ with the mean value $\mu_i = 0$ and the standard deviation σ_i. We can suppose that these sources are independent. On the other side, the

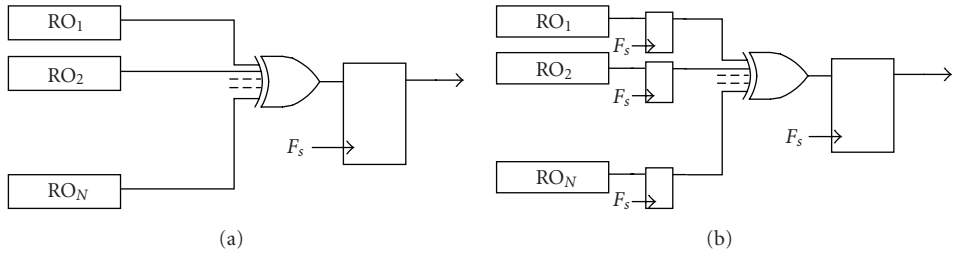


Figure 1: Original TRNG architecture of Sunar et al. (a) and modified architecture of Wold and Tan (b).

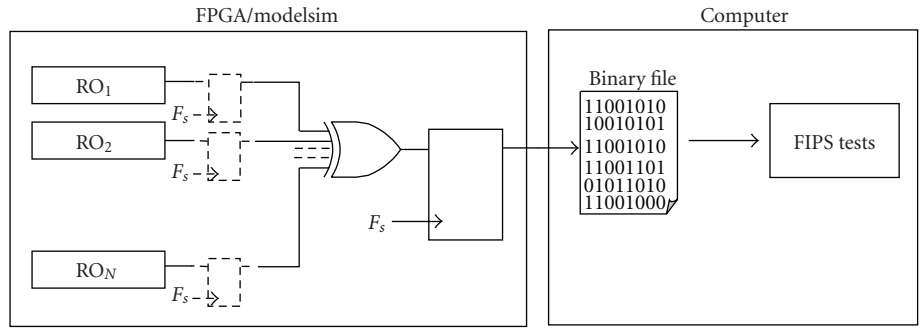


Figure 2: Simulation and experimental platforms.

local deterministic components can feature some mutual dependency, for example, from cross-talks.

Besides being influenced locally, the delays of all logic gates in the device are modified both slowly and dynamically by global jitter sources. The slow changes of the gate delay ΔD (the drift) can be caused by a slow variation of the power supply and/or temperature. The power source noise and some deterministic signal, which can be superposed on the supply voltage, can cause dynamic gate delay modification composed of a Gaussian global jitter component Δd_{GG} and a deterministic global jitter component Δd_{GD}. The overall global jitter from (3) can therefore be expressed as

$$\Delta d_{\mathrm{G}i} = K_i(\Delta D + \Delta d_{\mathrm{GG}} + \Delta d_{\mathrm{GD}}), \tag{5}$$

where $K_i \in [0; 1]$ corresponds to the proportion of the global jitter sources on the given gate delay. This comes from the fact that the amount of the global jitter included in delays of individual logic gates is not necessarily the same for all gates. It is important to note that K_i depends on the power supply voltage, but this dependence may differ for individual gates.

In real physical systems, the switching current of each gate modifies locally and/or globally the voltage level of the power supply, which in turn modifies (again locally and/or globally) the gate delay. This way, the delays of individual gates are not completely independent. We will discuss this phenomenon in Section 6.

3. Simulation and Experimental Setup

The aim of the first part of our work was to compare the behavior of two RO-based TRNGs: the original architecture depicted in Figure 1(a) that was proposed by Sunar et al. in [15] and its modified version presented in Figure 1(b) proposed by Wold and Tan in [18]. Contrary to the strategy adopted in [18], where the behavior of the two generators was compared only in hardware, we propose to compare it on simulation level, too. This approach has two advantages:

(i) the functional simulation results correspond to an ideal behavior of the generator, this way the two underlying mathematical models can be compared;

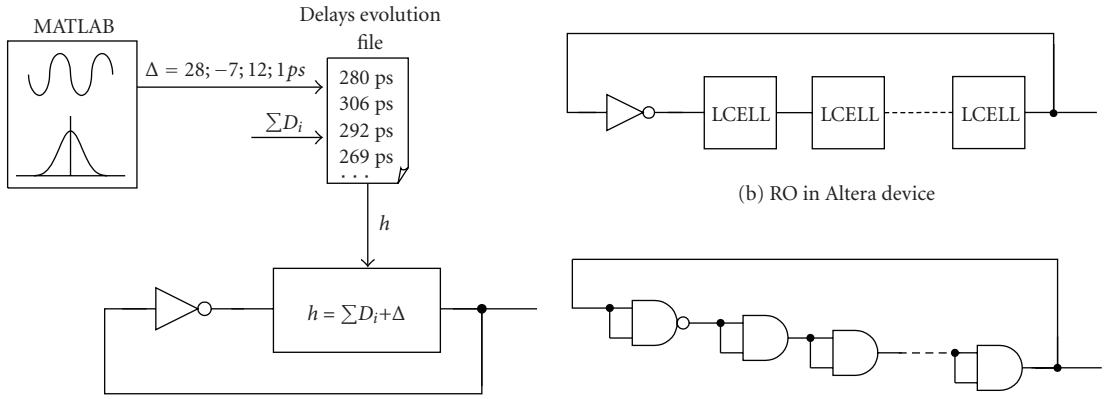
(ii) in contrast with the real hardware, thanks to simulation we can modify the parameters of injected jitter and evaluate the impact of each type of jitter on the quality of the generated bit-stream.

The principle of our simulation platform and experimental platform is depicted in Figure 2. For both platforms, the two generators were described in VHDL language and their architectures differed only in the use of flip-flops on the rings outputs (dashed blocks in Figure 2). The bitstreams obtained at the output of the final sampling flip-flop (before the post-processing) were tested and evaluated for different types and sizes of jitter in simulations and for different numbers of ring oscillators in both simulations and hardware experiments. The output of the TRNG was written into a binary file that was used as an input file in statistical tests.

We avoided the post-processing in the generator of Sunar et al. for two reasons:

(i) the post-processing function can hide imperfections in the generated signal;

(ii) using the same structures, we wanted to compare the two generators more fairly.



(a) RO in simulations

(b) RO in Altera device

(c) RO in Actel device

Figure 3: Implementation of ring oscillators in simulations and in hardware.

At this first level of investigation, we used the statistical tests FIPS 140-2 [1] in order to evaluate the quality of the generated raw bit-streams. We preferred the FIPS tests before the NIST test suite [2], because of the speed and the size of files needed for testing. While using significantly smaller files, the FIPS tests give a good estimation of the quality of the generated raw signal. If these tests do not pass, it is not necessary to go further. As our aim was to test a big set of TRNG configurations, the time and data size constraints were important. However, when the FIPS tests pass, we cannot conclude that the quality of the sequence produced by the TRNG is good. In this case, the generator should be thoroughly inspected.

3.1. Simulation Methodology. In order to compare the two generators on the functional simulation level (i.e., using ideal components), the behavior of ring oscillators was modeled in VHDL by delay elements with dynamically varying delays.

The jittered half-period, generated in MatLab Ver. R2008b, is based on (2) to (5). However, we take into account only local Gaussian jitter $\Delta d_{\mathrm{LG}i}$ (the source of true-randomness) and global deterministic jitter Δd_{GD} (the source of pseudo-randomness which can easily be manipulated) in our simulations. This approach was explained in [19]. Both sources of jitter depend a priori on the time t. Thus the simplified equation used in MatLab for generating jittery half-periods over the time t, denoted by $h(t)$, is expressed as

$$h(t) = \sum_{i=1}^{k} (D_i(t) + \Delta \mathrm{d}_{\mathrm{LG}i}(t) + K_i \times \Delta d_{\mathrm{GD}}(t)). \quad (6)$$

The fix part of the generated delay that determines the mean half-period of the ring oscillator is defined as a sum of k delay elements featuring constant delay $D_i(t) = D_i$ for each gate i. The variable delay that is added to the mean half-period is composed of a Gaussian component generated for each gate individually and of a deterministic component generated by the same generator for all gates and rings. The Gaussian delay component $\Delta d_{\mathrm{LG}i}(t)$ can be seen as a stationary process (i.e., mean and variance of $\Delta d_{\mathrm{LG}i}(t)$ do not change over the time t). Thus $\Delta d_{\mathrm{LG}i}(t)$ can be generated using the *normrnd* function in MatLab with mean 0 and standard deviation σ_i. The deterministic component $\Delta d_{\mathrm{GD}}(t)$ applied at time t is calculated in MatLab in the following way:

$$\Delta d_{\mathrm{GD}}(t) = A_{\mathrm{GD}} \times \sin(2\pi F_{\mathrm{GD}} \times t), \quad (7)$$

where A_{GD} and F_{GD} represent the amplitude and frequency of the deterministic signal. Note, that the *sin* function can be replaced with the *square*, *sawtooth*, or other deterministic function. Coefficients K_i are used to simulate the varying influence of $\Delta d_{\mathrm{GD}}(t)$ on each gate i.

Once the parameters k, D_i, K_i, σ_i, A_{GD}, and F_{GD} were set, a separated file containing a stream of half-period values was generated for each ring oscillator. These files were read during the VHDL behavioral simulations performed using the ModelSim SE 6.4 software, as presented in Figure 3(a).

The output signals were sampled using a D flip-flop at the sampling frequency $F_s = 32$ MHz, and the obtained 20,000 samples were written during the simulation to a binary file. Finally, generated sequences were tested using FIPS 140-2 tests.

3.2. Methodology of Testing in Hardware. Enhancements of the generator architecture brought by Wold and Tan were related to the behavior of the XOR gate. In order to compare generators' behavior in two different technologies, we employed one from Altera (the same that was used in [18]) based on Lookup tables (LUT) and one from Actel based on multiplexers. We developed two different modules dedicated to TRNG testing. Each module contained a selected FPGA device, a 16 MHz quartz oscillator and low-noise linear voltage regulators. The modules were plugged into a motherboard containing Cypress USB interface device CY7C6B013A-100AXC powered by an isolated power supply.

The Altera module contained the Cyclone III EP3C25F256C8N device. The noninverting delay elements and one inverter were mapped to LUT-based logic cells (LCELL) from Altera library (see Figure 3(b)). This way, either odd or even number of delay elements could be used in order to tune the frequency in smaller steps. We used

Quartus II software version 9.0 from Altera for mapping the rings into the device. All delay elements were preferably placed in the same logic array block (LAB) in order to minimize the dispersion of parameters.

The Actel Fusion module featured the M7AFS600FGG2-56 FPGA device. The non-inverting delay elements were implemented using AND2 gates from Actel library with two inputs short-connected and one inverter (again from Actel library) was added to close the loop (see Figure 3(c)).

On both hardware platforms, an internal PLL was used to generate the 32 MHz sampling clock F_s. The generated bitstreams were sent to the PC using the USB interface. A 16-bit interface communicating with the Cypress USB controller was implemented inside the FPGA. A Visual C++ application running on the PC read the USB peripheral and wrote data into a binary file that was used by the FIPS 140-2 tests software.

4. Comparison of the Generators' Behavior in Simulations and in Hardware

First, we compared the behavior of both generators in VHDL simulations. The generators used 1 to 20 ROs consisting of $k = 9$ inverters each. To take into account the differences related to the placement and routing of ROs, we supposed that the mean delay D_i of individual gates was slightly different from one ring to another (between 275 ps and 281 ps). The additional Gaussian jitter Δd_{LG_i} has mean 0 and standard deviation $\sigma_i = 30$ ps. No deterministic component was added in this experiment (i.e., $\Delta d_{\mathrm{GD}}(t) = 0$). It is important to note that the same random data files were used for both Sunar's and Wold's generators.

4.1. Simulation Results and Mathematical Analysis of the Generator Behavior. The simulation results for both evaluated generators are presented in Figure 4. Note that the "Monobit" and "Poker" tests succeed if the test result lies in the gray area. The "Run" test succeeds if no run fails. The "Long run" test is not presented in the graphs because it always succeeded.

It can be seen that in all configurations the two versions of the generator gave very similar (almost identical) results. Next, we will explain the reason for this behavior.

Let $b_j(t)$ be the bit sampled at time $t \geq 0$ at the output of the ring oscillator RO_j. This bit depends on the time t, the period $T_j > 0$ generated by RO_j, and the initial phase φ_j of RO_j at time $t = 0$.

Note that T_j, t, and φ_j are expressed in the time domain. This dependency is given by the following relation:

$$b_j(t) = 1 - \left\lfloor \frac{2\left(\varphi_j + t \bmod T_j\right)}{T_j} \right\rfloor. \tag{8}$$

Equation (8) ensures that $b_j(t)$ is a bit (i.e., $b_j(t) \in \{0, 1\}$).

Indeed, by definition of operation mod T_j,

$$0 \leq \left(\varphi_j + t\right) \bmod T_j < T_j, \tag{9}$$

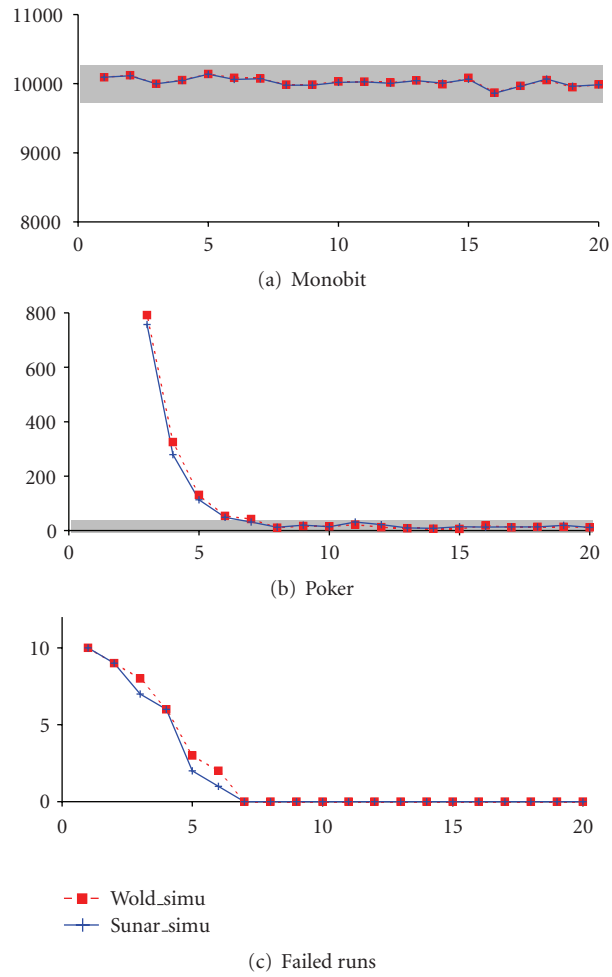


Figure 4: Results of the FIPS 140-2 tests in simulation with 30 ps of Gaussian jitter for Sunar's and Wold's architecture (excluding long runs that always passed), the tests pass if the results are in the gray region or are equal to zero for the "Runs" test.

then

$$0 \leq \frac{\left(\varphi_j + t\right) \bmod T_j}{T_j/2} < 2, \tag{10}$$

and by definition of the floor operation,

$$0 \leq \left\lfloor \frac{2\left(\varphi_j + t \bmod T_j\right)}{T_j} \right\rfloor \leq 1, \tag{11}$$

that means

$$\left\lfloor \frac{2\left(\varphi_j + t \bmod T_j\right)}{T_j} \right\rfloor \in \{0, 1\}, \tag{12}$$

thus

$$b_j(t) = 1 - \left\lfloor \frac{2\left(\varphi_j + t \bmod T_j\right)}{T_j} \right\rfloor \in \{0, 1\}. \tag{13}$$

Equation (8) holds if T_j is constant. Then if $(\varphi_j + t) \bmod T_j < T_j/2$, the sample value will be "1" otherwise it will be "0".

In both Sunar's and Wold's designs, outputs of N ring oscillators are XOR-ed to get one random bit as it is shown in Figure 1.

From the mathematical point of view, XOR-ing the outputs of N ring oscillators in Sunar's architecture is given by

$$S(t) = \bigoplus_{j=1}^{N} b_j(t) \tag{14}$$

or

$$\begin{aligned} S(t) &= \sum_{j=1}^{N} b_j(t) \bmod 2 \\ &= \sum_{j=1}^{N} \left(1 - \left\lfloor \frac{(\varphi_j + t) \bmod T_j}{T_j/2} \right\rfloor\right) \bmod 2. \end{aligned} \tag{15}$$

Thus

$$S(t) = \left(N + \sum_{j=1}^{N} \left\lfloor \frac{(\varphi_j + t) \bmod T_j}{T_j/2} \right\rfloor\right) \bmod 2. \tag{16}$$

This way, the nth bit sampled at time $n \times T_s$ in the D flip-flop is given by (16) for $t = n \times T_s$.

In Wold's architecture, each ring oscillator is sampled at time $n \times T_s$, which gives a set of bits $\{b_1(n \times T_s), b_2(n \times T_s), \ldots, b_N(n \times T_s)\}$. Then all these bits are XOR-ed and the result is sampled at time $(n+1) \times T_s$, giving the output of Wold's TRNG denoted by $W((n+1) \times T_s)$:

$$W((n+1) \times T_s) = \bigoplus_{j=1}^{N} b_j(n \times T_s). \tag{17}$$

Thus using equations (14) and (17), the relation between outputs of Wold's TRNG and Sunar's TRNG is given by

$$W((n+1) \times T_s) = S(n \times T_s). \tag{18}$$

We can conclude that from the mathematical point of view, assuming constant periods of ring oscillators and ideal components, Sunar's and Wold's generators can be described in the same way. The only difference is the right shift of the sequence from the Wold's generator. Note that this conclusion explains also the similar behavior of both generators in simulation as it is shown in Figure 4.

The claim that both ideal generators behave according to the same mathematical model is very important, because it means that the security proof of Sunar can be applied (at least theoretically) to both of them. However, as we will see in the next section, their behavior in hardware is very different.

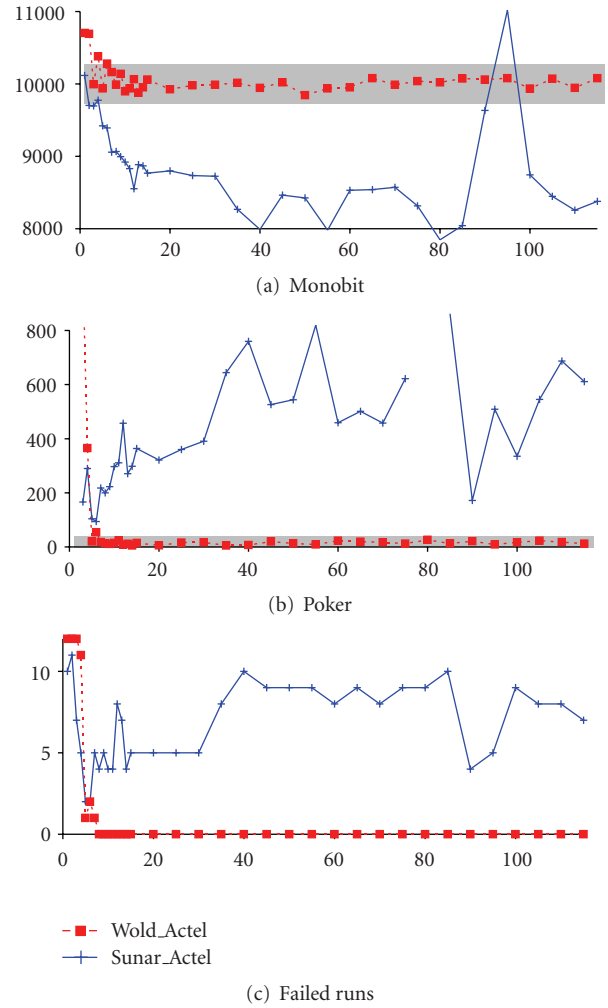


Figure 5: Results of the FIPS 140-2 tests for observed TRNG architectures with varying number of ring oscillators in Actel Fusion device.

4.2. Results Obtained in Hardware. We applied the FIPS 140-2 tests on the raw binary signals generated by the two generators, while incrementing the number of ROs. The results obtained for Actel FPGA are presented in Figure 5 and those obtained for Altera FPGA in Figure 6. The number of ROs varied from 1 to 20 by increments of 1 and from 20 to 115 by increments of 5.

It can be seen that for the Sunar's architecture the tests passed neither for Altera nor for Actel family. However, we can note that the Altera Cyclone III device gave slightly better results. This was probably due to the different behavior of the XOR gate in selected technologies. In the same time, concerning the architecture of Wold, the tests passed for both technologies if the number of ROs was higher than 8. Note that these results confirmed those obtained on standard evaluation boards from Altera and Actel published in [22].

The claims of Wold are thus confirmed in both technologies and various types of boards. However, it is not clear what kind of randomness lets the tests pass. Is it mostly a pseudo-randomness (coming from the sequential behavior

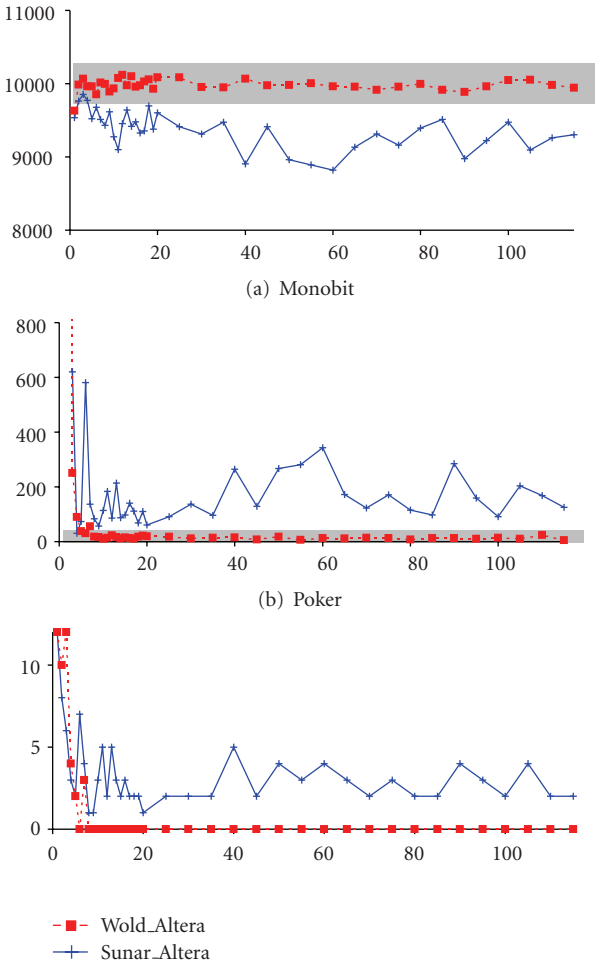


FIGURE 6: Results of the FIPS 140-2 tests for observed TRNG architectures with varying number of ring oscillators in Altera Cyclone III device.

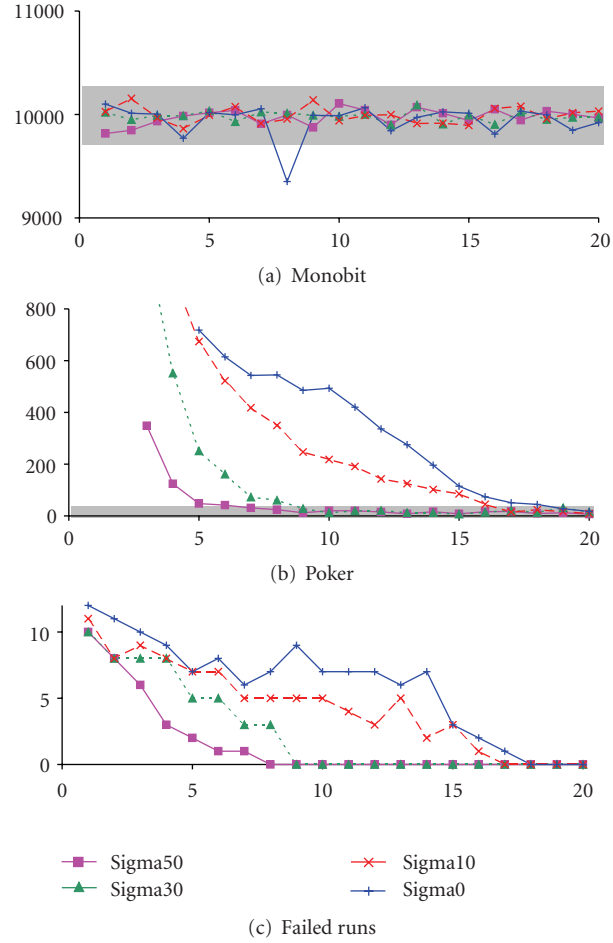


FIGURE 7: Results of the FIPS 140-2 tests in the Wold's TRNG architecture simulations with varying size of injected Gaussian jitter.

of the generator characterized by the internal state evolution) that can theoretically be attacked or a true-randomness that should be employed? The tests are clearly not able to distinguish between them. Again, the simulation can give answers to these questions.

5. Impact of the Size and Type of the Jitter on the Quality of the Raw Bit-Stream

As the architectures of Sunar and Wold have the same ideal behavior, we will analyze only the architecture of Wold and Tan, because its behavior in hardware is closer to the idealized mathematical model. Next, we will study the impact of both Gaussian and deterministic components of the jitter on the generated raw signal.

5.1. Impact of the Size of the Gaussian Jitter on the FIPS 140-2 Tests. Again, we have simulated the behavior of the Wold's architecture with 1 to 20 ROs composed of 9 elements. The half period of each RO was composed of a mean value (frequency of RO-generated clock signal varied between 197,5 MHz and 202 MHz in 250 kHz steps) and of an additional random value (normally distributed with mean 0 and σ_i = 0, 10, 30, and 50 ps for each gate). The random signals were generated in MatLab, independently for each RO. The results are presented in Figure 7.

Two facts can be observed in these figures:

(i) as expected, when the random jitter increases, the tests pass more easily (i.e., with a reduced number of ROs);

(ii) more surprisingly, the tests pass even if the random jitter is not injected at all ($\sigma = 0$), and this for 18 ROs or more.

Thanks to the mathematical model from (16), which generates the same sequences as the simulation tool, we could generate faster long bit-streams in order to perform NIST tests. The results were conclusive: the NIST tests passed with this full deterministic behavior (without any randomness) for only 18 ROs.

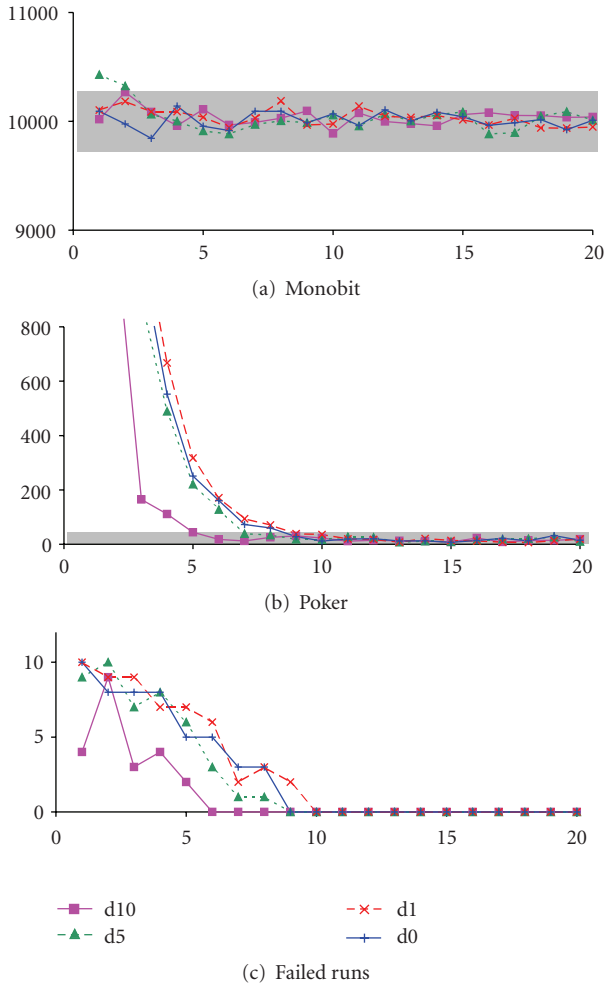


FIGURE 8: Results of the FIPS 140-2 tests in the Wold's TRNG architecture simulations while injecting a Gaussian jitter of 30 ps and a varying deterministic jitter.

The fact that the tests (FIPS and NIST) pass for a few ROs without Gaussian jitter means that both Sunar's and Wold's architectures produce a great amount of pseudo-randomness. We recall that pseudo-randomness in the generated sequence depends on the frequencies of ROs and can be manipulated from outside the chip (e.g., by modulating the power supply or by an electromagnetic interference as it was presented recently at the CHES conference [23]). Furthermore, using a mathematical model (e.g., that from (16)), some patterns can be predicted. The proportion of pseudo-random and true-random components in the generated sequence is thus very important. However, in the solution proposed by Wold, the number of ROs is significantly reduced, because the NIST tests passed. This can be considered as a security-critical attempt for cryptography applications and should certainly be avoided.

5.2. Injecting a Deterministic Jitter. In the next experiments, the Gaussian jitter remained constant ($\sigma_i = 30$ ps per gate) and we applied a sinusoidal deterministic jitter of 3 kHz and 0 to 10 ps in amplitude. The results are presented in Figure 8.

It can be seen that when the deterministic part increases, the tests pass more easily. But there are two problems concerning the deterministic component of the jitter:

(i) the results are strongly dependent on the frequency of the injected signal: depending on the frequency, the output of the TRNG can vary in time in a predictable way;

(ii) the deterministic jitter can be manipulated (for example, an attacker can superimpose a chosen signal that seems to improve randomness so that the tests would pass more easily, but in fact, he can predict some trends in subsequences).

For the above-mentioned reasons, the designer should reduce the pseudo-randomness coming from the deterministic jitter component as much as possible.

5.3. Reducing the Influence of the Deterministic Jitter Component. As we pointed out in [17], the impact of the deterministic jitter on the generated random numbers can significantly be reduced by the use of a reference clock signal featuring the same deterministic global jitter. This fact is not taken into account in any observed TRNG designs [15, 16, 18]. In the following experiments, we used a clock signal generated by another internal ring oscillator as a sampling clock. The generated signal frequency was divided by 8 and then used as a sampling clock (having thus the frequency of about 25 MHz). We implemented the Wold's TRNGs with 1 to 20 ROs. In the first experiment, we did not inject the deterministic jitter component and we let the Gaussian part vary between 0 and 50 ps. The results are presented in Figure 9. Next, we fixed the Gaussian jitter to $\sigma = 30$ ps and we applied a sinusoidal deterministic jitter of 3 kHz and 0 to 10 ps in amplitude. The results are presented in Figure 10.

As expected, increasing the standard deviation of the injected Gaussian jitter component from 0 to 50 ps, the tests passed more easily for both external and internal sampling clocks (see Figures 7 and 9, respectively). It is important to note that for a fixed Gaussian jitter size, tests passed more easily if the sampling was performed with an internal clock signal (i.e., from another ring oscillator), because independent Gaussian jitter components were included in both sampling and sampled signals and they increased the entropy of the generated bit-stream.

In the last experiment, the Gaussian jitter component was constant and the deterministic jitter component varied. The results for external and internal sampling clocks are presented in Figures 8 and 10, respectively. Contrary to the use of an external clock, the influence of the deterministic jitter was strongly reduced when an internal clock was used. As expected, whatever the size of the injected deterministic jitter component, the test results were similar.

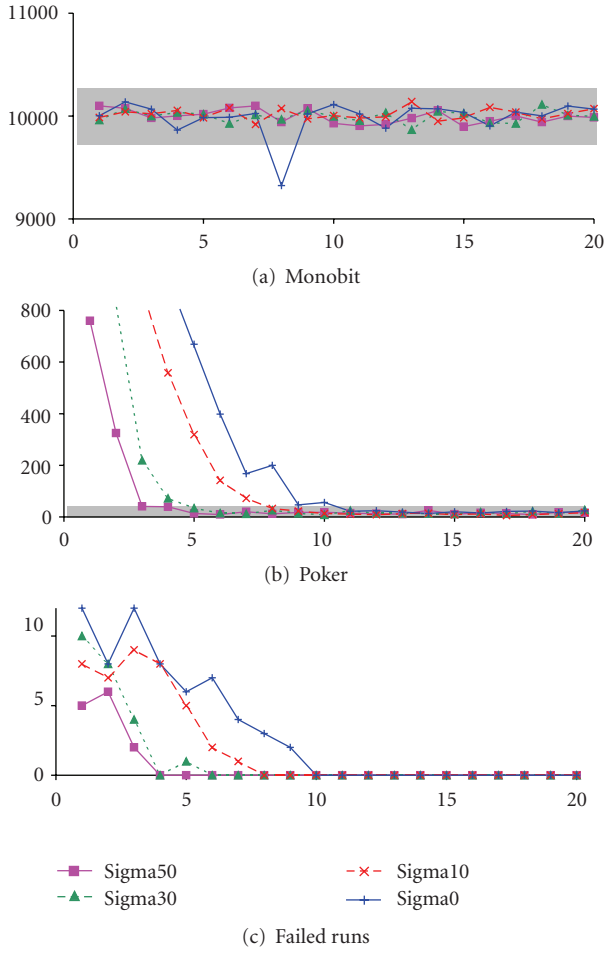


FIGURE 9: Results of the FIPS 140-2 tests of the Wold's TRNG architecture simulations with internal reference clock and varying size of injected Gaussian jitter.

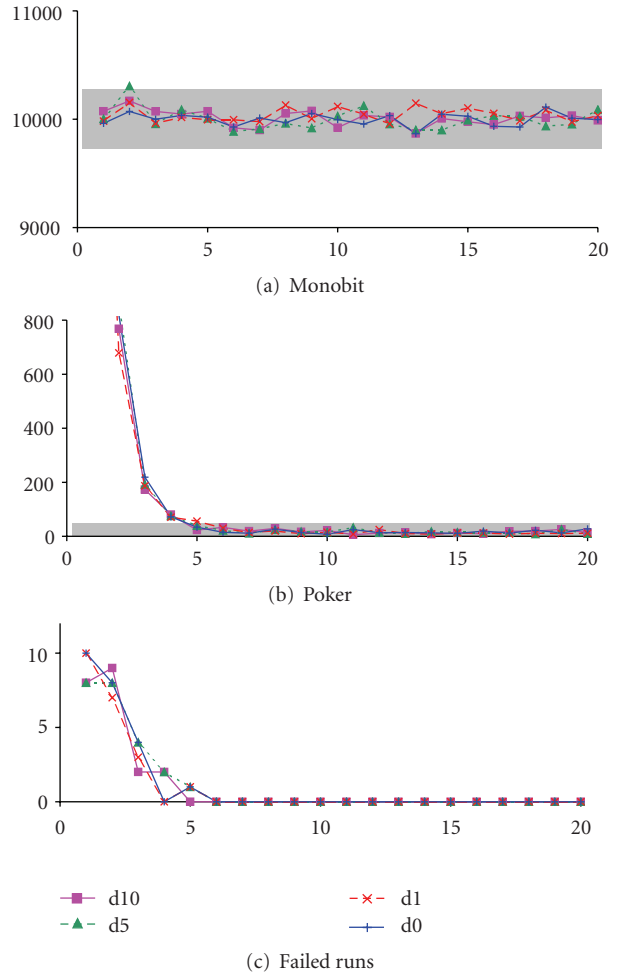


FIGURE 10: Results of the FIPS 140-2 test of the Wold's TRNG architecture simulations while injecting a Gaussian jitter with $\sigma = 30$ ps and a varying size of deterministic jitter, with internal reference clock.

6. Mutual Dependence of Rings and Its Impact on the Quality of the Raw Bit-Stream

The aim of the following experiment was to validate the mutual independence of ring oscillators implemented in FPGA. Note that this mutual independence of rings is a necessary condition for the validity of the security proof from [15].

First, we implemented pairs of 9-element ROs with similar topology (similar routing) in both tested FPGA technologies. The generated clock periods were measured using the high bandwidth (3.5 Ghz) digital oscilloscope LeCroy WavePro 735Zi. The 1.2 V power supply of the FPGA core was replaced with an external variable power supply. The output clock signal periods were measured for the power supply ranging from 0.9 V to 1.3 V.

We can observe in Figure 11 that for the Actel Fusion family the periods of the generated clock signals depend on the power supply in very similar, but not exactly the same way. We note that for a certain voltage interval the periods overlap, while outside this interval they tend to separate slightly. This corresponds to the use of coefficients K_i in Section 2 and to the claim that they depend on the power supply, but differently for each gate or ring.

The difference between the two clock periods as the function of the power supply can be observed in Figure 12. We can notice that it changes from negative to positive values following a monotonously rising curve; but suddenly, in the interval from 1.02 to 1.12 V, it drops almost to zero. The only explanation of this effect is that the rings become locked (otherwise the curve should drop to zero only in one point). The fact that the period difference is not zero is explained later. We can conclude that in the locking range the role of coefficients K_i on determination of the frequency (period) is overruled by some other phenomenon. This is coming probably from the dependence of the frequency of one oscillator on the current peaks caused by rising and falling edges of the second oscillator, as it is explained in the end of Section 2. This phenomenon was not observed in the literature up to now.

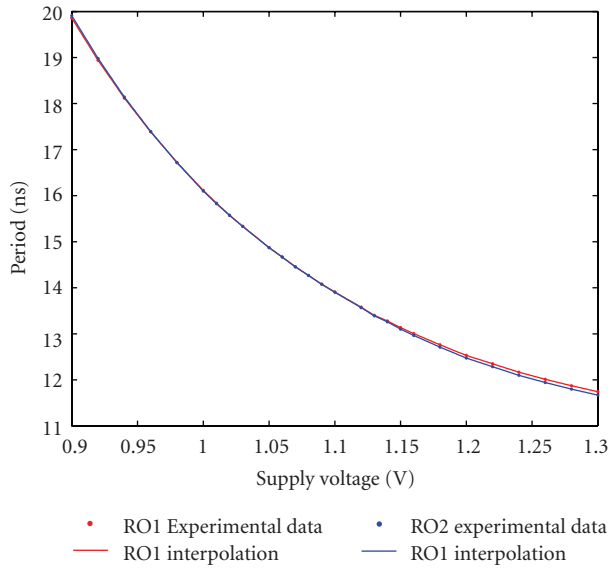


Figure 11: Dependence of clock periods of two rings in Actel Fusion device on the power supply.

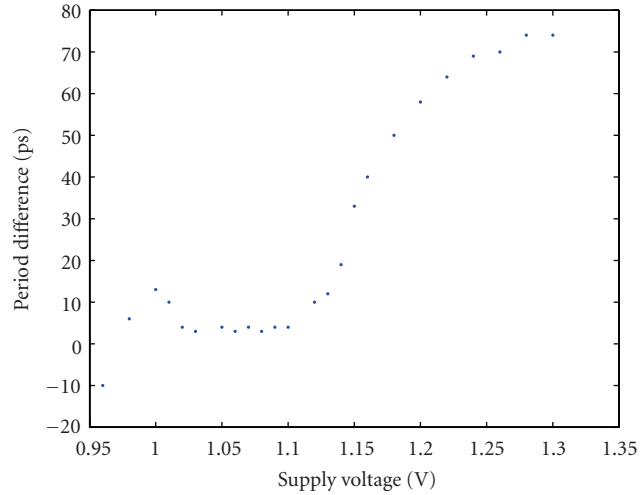


Figure 12: Difference between clock periods of two rings depending on the power supply.

The locking of two rings was also observable on the oscilloscope, as it is depicted in Figure 13 using the screen persistence. We could observe several phenomena during our experiments:

(i) when the rings' frequencies were sufficiently close, it was quite easy to lock the rings by modifying the power supply;

(ii) the locking could be observed for both technologies used;

(iii) although most of the time the phase of the two signals on the oscilloscope was perfectly stable, sometimes they became unlocked for a very short time—this explains why the period difference measured in long time intervals was not exactly zero in the locking zone;

(iv) we could quite easily obtain the state when about 25% of rings were locked—most of them were locked on a dominant frequency and other smaller groups of rings on other frequencies (this phenomenon was observed on all tested cards—five cards for each evaluated technology);

(v) the state when two or more rings were locked on the nominal voltage level (without manipulating the power supply) could also be obtained.

7. Discussion

As it was shown in Section 2, the generator of Sunar et al. and the generator of Wold and Tan are based on the same mathematical model (see (16)), when built using ideal components. This fact means that the proof of Sunar can be valid for both architectures. We recall that the proof of Sunar is based on the entropy estimation based on the jitter measurement before the XOR gate, while supposing that this gate does not reduce the entropy. However, this last assumption is not true for the architecture of Sunar when implemented in physical devices. The architecture of Wold and Tan solves this problem and should be preferred.

Nevertheless, even the new architecture does not eliminate serious doubts about the entropy contents in the raw signal. Unfortunately, this entropy cannot be measured. Applying the theory of Sunar et al., the entropy of the raw binary signal can be estimated knowing the sampling frequency, size of the jitter, and number of independent rings. Supposing that the rings are independent, this theory remains valid for the new generator architecture as we showed in Section 2. For this reason, we can conclude that while reducing number of rings, Wold and Tan reduced unconsciously the entropy of the generated signal. In order to maintain the security level, they should also modify the resilient function, in order to increase the compression ratio and to guarantee the output entropy per bit close to one. Instead, they propose to remove the post-processing, which is clearly a very dangerous action from the point of view of security.

Equation (16) implies that both generators contain some memory element (they are not memoryless in the sense of term used in [5]). This means that besides the true random behavior coming from the Gaussian jitter, they will feature a pseudo-random behavior. This behavior can be described for any time instant t depending on the previous generator state characterized by phases φ_j of N rings that generate clocks with periods T_j. Following the principle of the generator, the presence of this kind of pseudo-randomness in generated bitstream is unavoidable.

There is another source causing the pseudo-randomness in the raw binary signal. It comes from the global deterministic jitter sources. Both above-mentioned sources of pseudo-randomness are dangerous because they can mask the entropy insufficiency (the tests of randomness will pass)

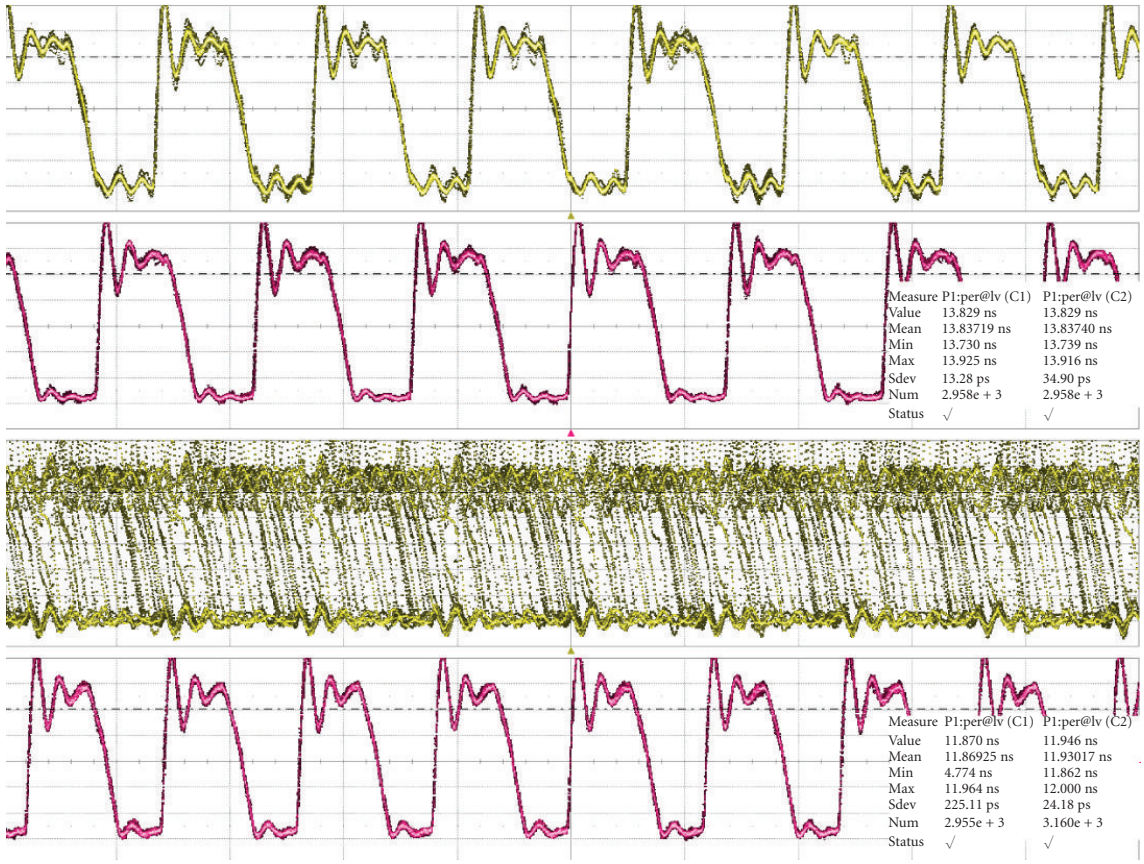


Figure 13: Waveforms of two locked (up) and unlocked (down) rings in Actel Fusion device.

and at the same time they can be manipulated. However, the pseudo-randomness coming from the evolution of the state of the generator described by (16) is more dangerous, because it can have two impacts: the attacker can manipulate the contents of the bit-stream and at the same time the entropy can be reduced.

For example, by modulating the power supply and thus changing the periods T_j, the attacker can control the pseudo-random behavior of the generator to some extent (mutual relations between clock periods) and the state can be reached where the rings are locked. This way, the effective number of usable (independent) rings is reduced. As in the case of the generator of Wold and Tan, the reduced number of rings will lower the entropy of the generated signal and at the same time the generator's pseudo-random behavior will be simpler and thus easier to guess.

8. Conclusions

As it was shown, the generator of Wold and Tan follows the same mathematical model as that of Sunar et al. The security proof of Sunar can thus be applied (theoretically) also in this case. Because the generator of Wold and Tan gives much better binary raw signal in hardware, it should be preferred. However, in order to assure that the proof of Sunar will hold, the number of rings should not be reduced as proposed in [18] only because the tests passed. As we showed, the generator using more than 18 ideal jitter-free rings having slightly different frequencies and producing only the manipulable pseudo-randomness will always let the tests pass.

In an ideal case (i.e., when the rings are independent) and following the proof of Sunar, the number of rings is defined by the size of the Gaussian component of the jitter and by the reference clock frequency, both in relationship with the post-processing resilient function. However, even if the number of rings remains high, some of them could be locked and the effective number of exploitable rings could be significantly lower. In this case, which is easy to obtain in real FPGAs and which can concern as much as 25% of rings or more, the entropy of the generated raw signal would be much lower than expected and the generator would be predictable or manipulable.

The locking of rings depends on their topology (placement and routing) and on the technology used. The probability of locking could perhaps be reduced by a careful placement and routing on a per-device basis or by an independent powering of all rings. Applying the first approach, the designer loses the main advantage of this class of TRNGs—device-independent design. The second approach is impossible to apply in FPGAs. Another strategy

can consist in detection of locking of rings in order to stop the generation of random numbers. However, the complexity of the detection circuitry would rise with the square of number of rings and it would thus limit the practical use of the generator, which is already penalized by the fact that the number of rings is considerable.

Although locking of rings can have disastrous consequences on the security of TRNGs based on ring oscillators, this phenomenon is not yet observed in the literature. For this reason, it should be studied extensively in the future.

Acknowledgments

This paper is an extended version of the conference paper [22] presented at ReConFig09. The research was partially supported by a Grant from the French National Research Agency-ANR-09-SEGI-013 and by the Rhone-Alpes Region and Saint-Etienne Metropole, France.

References

[1] Information Technology Laboratory, "FIPS 140-2: Security Requirements for Cryptographic Modules," Special Publication 140-2, May 2001.

[2] A. Rukhin, J. Soto, J. Nechvatal et al., "A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications," NIST Special Publication 800-22, 2001, http://csrc.nist.gov/.

[3] G. Marsaglia, "DIEHARD: Battery of Tests of Randomness," 1996, http://stat.fsu.edu/pub/diehard/.

[4] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, New York, NY, USA, 1997, http://www.cacr.math.uwaterloo.ca/hac/.

[5] W. Killmann and W. Schindler, *AIS 31: Functionality Classes and Evaluation Methodology for True (Physical) Random Number Generators, Version 3.1*, Bundesamt fur Sicherheit in der Informationstechnik (BSI), Bonn, Germany, 2001.

[6] I. Vasyltsov, E. Hambardzumyan, Y.-S. Kim, and B. Karpinskyy, "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.

[7] B. Fechner and A. Osterloh, "A meta-level true random number generator," *International Journal of Critical Computer-Based Systems*, vol. 1, no. 1–3, pp. 267–279, 2010.

[8] M. Varchola and M. Drutarovsky, "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.

[9] V. Fischer and M. Drutarovský, "True random number generator embedded in reconfigurable hardware," in *Proceedings of Cryptographic Hardware and Embedded Systems (CHES '02)*, B. S. Kaliski Jr., Ç. K. Koç, and C. Paar, Eds., vol. 2523 of *Lecture Notes in Computer Science*, pp. 415–430, Springer, Redwood Shores, Calif, USA, 2003.

[10] K. H. Tsoi, K. H. Leung, and P. H. W. Leong, "Compact FPGA-based true and pseudo random number generators," in *Proceedings of the 11th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM '03)*, pp. 51–61, 2003.

[11] M. Dichtl and J. D. Golić, "High-speed true random number generation with logic gates only," in *Proceedings of the 9th International Workshop on Cryptographic Hardware and Embedded Systems (CHES '07)*, P. Paillier and I. Verbauwhede, Eds., vol. 4727 of *Lecture Notes in Computer Science*, pp. 45–62, Springer, Vienna, Austria, September 2007.

[12] J.-L. Danger, S. Guilley, and P. Hoogvorst, "Fast true random generator in FPGAs," in *Proceedings of IEEE North-East Workshop on Circuits and Systems (NEWCAS '07)*, pp. 506–509, Montreal, Canada, August 2007.

[13] T. E. Tkacik, "A hardware random number generator," in *Proceedings of the Cryptographic Hardware and Embedded Systems (CHES '02)*, B. S. Kaliski Jr., Ç. K. Koç, and C. Paar, Eds., vol. 2523 of *Lecture Notes in Computer Science*, pp. 450–453, Springer, Redwood Shores, Calif, USA, 2003.

[14] P. Kohlbrenner and K. Gaj, "An embedded true random number generator for FPGAs," in *Proceedings of the 12th ACM/SIGDA International Symposium on Field Programmable Gate Arrays (FPGA '04)*, vol. 12, pp. 71–78, Monterey, Calif, USA, February 2004.

[15] 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*, pp. 109–119, 2007.

[16] D. Schellekens, B. Preneel, and I. Verbauwhede, "FPGA vendor agnostic true random number generator," in *Proceedings of the 16th International Conference on Field Programmable Logic and Applications (FPL '06)*, pp. 139–144, August 2006.

[17] V. Fischer, F. Bernard, N. Bochard, and M. Varchola, "Enhancing security of ring oscillator-based TRNG implemented in FPGA," in *Proceedings of the International Conference on Field Programmable Logic and Applications (FPL '08)*, pp. 245–250, September 2008.

[18] K. Wold and C. H. Tan, "Analysis and enhancement of random number generator in FPGA based on oscillator rings," *International Journal of Reconfigurable Computing*, vol. 2009, Article ID 501672, 8 pages, 2009.

[19] B. Valtchanov, A. Aubert, F. Bernard, and V. Fischer, "Modeling and observing the jitter in ring oscillators implemented in FPGAs," in *Proceedings of the 11th IEEE Workshop on Design and Diagnostics of Electronic Circuits and Systems (DDECS ')*, pp. 158–163, Bratislava, Slovakia, April 2008.

[20] V. Rožić and I. Verbauwhede, "Random numbers generation: investigation of narrowtransitions suppression on FPGA," in *Proceedings of the 19th International Conference on Field Programmable Logic and Applications (FPL '09)*, M. Danek, J. Kadlec, and B. Nelson, Eds., pp. 699–702, Prague, Czech Republic, August-September 2009.

[21] 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 the 13th IEEE Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS '10)*, pp. 48–53, Vienna, Austria, April 2010.

[22] N. Bochard, F. Bernard, and V. Fischer, "Observing the randomness in RO-based TRNG," in *Proceedings of the International Conference on ReConFigurable Computing and FPGAs (ReConFig '09)*, pp. 237–242, December 2009.

[23] A. T. Markettos and S. W. Moore, "The frequency injection attack on ring-oscillator-based true random number generators," in *Proceedings of the 11th International Workshop on Cryptographic Hardware and Embedded Systems (CHES '09)*, C. Clavier and K. Gaj, Eds., vol. 5747 of *Lecture Notes in Computer Science*, pp. 317–331, Springer, Lausanne, Switzerland, September 2009.