On the Effectiveness of True Random Number Generators Implemented on FPGAs

Davide Galli, Andrea Galimberti(✉), William Fornaciari, and Davide Zoni

Politecnico di Milano, P.zza L. Da Vinci, 20133 Milan, Italy
`davide11.galli@mail.polimi.it`,
`{andrea.galimberti,william.fornaciari,davide.zoni}@polimi.it`

Abstract. Randomness is at the core of many cryptographic implementations. True random number generators provide unpredictable sequences of numbers by exploiting physical phenomena. This work compares multiple literature proposals of true random number generators targeting FPGAs. The considered TRNGs are obtained as the combinations of three digital noise sources, namely, NLFIRO, PLL-TRNG, and ES-TRNG, and three post-processing techniques, namely, XOR, Von Neumann, and LFSR. The resulting combinations of such components are evaluated in terms of security, throughput, and resource utilization. The experimental results, which were collected on Xilinx Artix-7 FPGAs, highlight the importance of the post-processing stage for security purposes and reveal NLFIRO as the best digital noise source and LFSR as the best post-processing technique, having the highest throughput with excellent security performance without compromising area and power consumption.

Keywords: Field programmable gate arrays · True random number generators · Hardware-based security primitives · Side-channel attacks

1 Introduction

Modern embedded systems at the edge are pervasively deployed in our living environment and they are increasingly in charge of performing critical tasks or managing sensitive data, thus calling for new and stringent privacy and security requirements in addition to the more traditional energy-efficiency ones.

The use of efficient cryptographic primitives represents the de-facto solution to guaranteeing the privacy and security of the exchanged and processed data. However, the security strength of the cryptographic system is directly connected to the quality of the used random numbers, thus highlighting true random

Supported by the EU Horizon 2020 "TEXTAROSSA" project (Grant No. 956831).

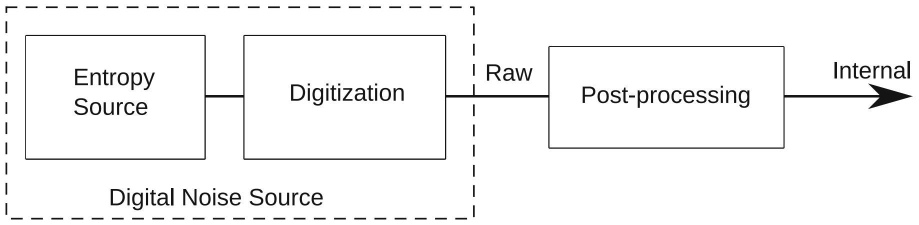

Fig. 1. Baseline architecture of a TRNG.

number generators (TRNGs) as essential components in the security infrastructure [8]. The correct implementation of TRNGs is indeed paramount to ensuring the effective security of the wide array of cryptographic primitives where they are employed, such as traditional [7] and post-quantum [10,12,19,20] key exchange mechanisms, digital signature schemes [5,11,15], and countermeasures to side-channel attacks [3,21]. A TRNG must therefore produce a sequence of numbers such that the generated values are statistically independent, uniformly distributed, and unpredictable.

The architecture of a generic TRNG is depicted in Fig. 1, and it can be split into three main components. They are, respectively, entropy source, digitization, and post-processing, with the latter also referred to as conditioning.

The entropy source is the only component in the architecture that generates true randomness by exploiting some physical phenomenon, while the other components are purely deterministic. Whenever the entropy source generates analog signals, a digitization module is required to transform them into digital form. The combination of an entropy source and a digitization module is also referred to as a digital noise source, which is notably often indicated in the literature as the TRNG itself due to its role in producing the actual entropy underlying the random number generation. The digital noise source outputs the *raw random numbers*, so-called since they are frequently vulnerable to statistical flaws.

Therefore, a post-processing module is used to improve the statistical and security properties of the TRNG. The post-processed output, with increased entropy, is also referred to as *internal random numbers*. Notably, applying conditioning methods is not mandatory for the design of a TRNG, and it can instead be avoided if the entropy produced by the digital noise source is already sufficient for the target application of the TRNG.

Contributions - This manuscript presents an exploration of the state-of-the-art TRNGs targeting FPGAs, evaluated according to resource utilization, throughput, and security. The goal is to identify the best-performing combinations of digital noise sources and conditioning methods that optimize the three aforementioned quality metrics. The experimental results highlight the combination of an NLFIRO digital noise source and an LFSR post-processing method as the best-performing solution with respect to the throughput and security metrics At the same time, ES-TRNG might be an effective solution in tightly resource-constrained scenarios, albeit with a drastic reduction in the throughput.

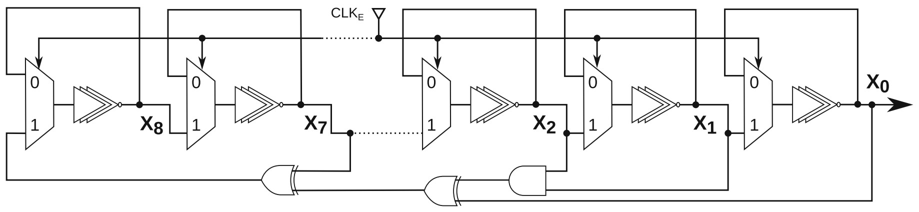

Fig. 2. Architecture of a NLFIRO TRNG.

2 Methodology

This section describes the architectures of the considered TRNGs, detailing separately the three digital noise sources and the three post-processing methods. In particular, the evaluated digital noise sources are the NLFIRO, PLL-based, and edge-sampling TRNGs, while the considered post-processing methods are XOR, Von Neumann, and LFSR conditioning techniques. Notably, all of them are suitable for an FPGA implementation, albeit exploiting different physical phenomena and working principles.

2.1 Digital Noise Sources

NLFIRO TRNG. Non-linear feedback ring oscillators (NLFRO) [16] enhance the design of ring oscillators by incorporating a non-linear feedback function. NLFRO-based TRNGs harvest their high entropy from different sources, namely, noise and variation in delay cells, unpredictable behavior in astable logic elements, and non-linear feedback loops.

Each stage of the ring oscillator presents a multiplexer driven by a clock signal CLK_E, making it possible to reconfigure the ring oscillator between a local loop and an open loop. When the clock is low, the local loop is closed and the signal runs in a ring oscillator composed by an inverter chain. When the clock is high, the loop is open and the entire feedback starts working. CLK_E must have the same frequency and duty cycle of the sampling clock, while a phase shift of $-90°$ is required to leave a sufficient margin for the toggling of the global feedback.

We implement the Fibonacci configuration of the NLFRO with nine stages and three NOT gates as inverter chain, which is referred to as NLFIRO [16]. Figure 2 depicts the architecture of the implemented NLFIRO, whose feedback function is shown in Eq. (1).

$$f(x) = x^7 \oplus (x^2 \cdot x^1) \oplus x^0 \qquad (1)$$

For each TRNG, two NLFIROs are XORed together after being sampled in two D-FFs, and the result of the XOR is registered in another D-FF.

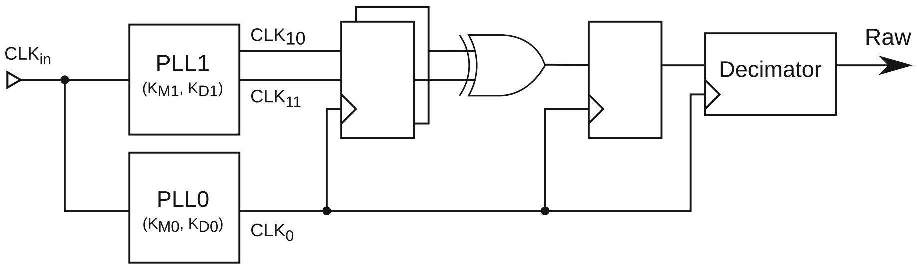

Fig. 3. Architecture of a PLL-based TRNG.

PLL-Based TRNG. A PLL-based TRNG [2,9] exploits the coherent sampling principle. As shown in Fig. 3, which depicts its architecture, a reference clock clk_0 is used to sample two other clock signals clk_{10} and clk_{11}, where clk_{10} is delayed of 90° respect to clk_{11}, in two corresponding D flip-flops (D-FFs). The outputs of such D-FFs are then XORed together and registered in another D-FF. Since the output bits of the latter suffer of a period pattern, they are then decimated by XORing K_D consecutive bits to destroy such pattern and obtain a stream of raw bits.

The reference and sampled frequencies are mutually related according to Eq. (2), where K_{M1}, K_{M0}, K_{D0}, and K_{D1} are the frequency multiplication and division factors.

$$f_1 = f_0 \frac{K_{M1} K_{D0}}{K_{M0} K_{D1}} = f_0 \frac{K_M}{K_D} \tag{2}$$

The output signal of the last D-FF features a pseudo-random pattern with a period $T_Q = K_D/f_0 = K_M/f_1$, which is however removed by the decimator. The output bit-rate R of the generator and its sensitivity S to jitter are defined according to Eq. (3) and Eq. (4), respectively.

$$R = T_Q^{-1} = \frac{f_0}{K_D} \tag{3}$$

$$S = f_1 K_D = f_0 K_M \tag{4}$$

Consequently, R and S are maximized by increasing f_0 and K_M and decreasing K_D. The algorithm proposed in [1] is exploited to find the best parameters for all the PLLs, allowing to obtain a sufficient entropy and bit rate while fulfilling the hardware requirements of the PLL instances.

Edge-Sampling TRNG. The edge-sampling TRNG (ES-TRNG) [18] is characterized by a low resource utilization, with the core TRNG module occupying only 5 LUTs and 6 FFs, as shown by its architectural representation depicted in Fig. 4. Two components devoted to resynchronization and control logic are however also required, in addition to such TRNG core logic.

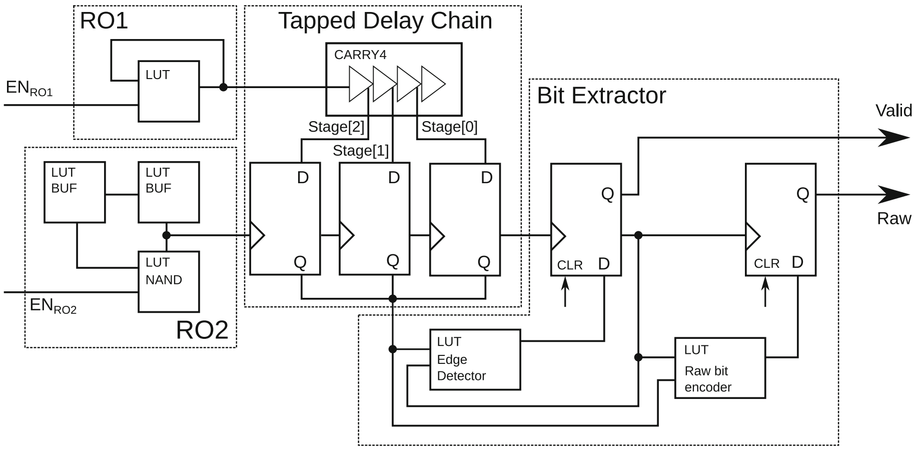

Fig. 4. Architecture of an edge-sampling TRNG.

The ES-TRNG digital noise source exploits the entropy produced by the timing phase jitter of a free-running ring oscillator. The architecture of a ES-TRNG consists of a small ring oscillator `RO1` with an enable signal. The output of `RO1` is then digitalized through a tapped delay chain which slows down the signal and allows saving three different values in three FFs (`Stage[2:0]`). The latter FFs are driven by a larger ring oscillator `RO2`. As the final processing step, a bit extractor module is tasked with extracting a high-entropy bit from the sampled stages `Stage[2:0]`.

The core idea of ES-TRNG is to repeat the sampling at high frequency thanks to `RO2` and focus only on the region around the edges of the `RO1` signal, where the bits have higher entropy.

`RO1` must be reset after generating a new raw bit and be enabled for a period T_A before starting `RO2`. The accumulation time T_A is used by `RO1` to increase the entropy. The designer shall find a trade-off between the required entropy value and the desired throughput.

2.2 Post-processing Methods

XOR Post-processing. XOR post-processing [6] processes n bits of the raw input stream at a time. In particular, such n bits are XORed together, which reduces the throughput by a factor of n. Taking the parity of n independent bits reduces the internal bias ϵ to $\epsilon_{internal} = 2^{n-1}\epsilon_{raw}^{n}$, resulting into a higher entropy. For the purposes of this work, n was set to 2 for NLFIRO, while XOR post-processing was not applied to PLL-TRNG and ES-TRNG sources.

Von Neumann Post-processing. The Von Neumann conditioning method [13] splits the input bits into non-overlapping pairs and, for pairs whose first and second bits are different, it outputs the first bit, while nothing is output if the two bits hold the same value. The throughput is not constant, but it can never exceed 1/4 of the raw throughput, i.e., the throughput of the digital noise source.

LFSR Post-processing. A linear-feedback shift register (LFSR) is a shift register whose input bit is a linear function of its previous state, and it is defined by a characteristic polynomial. The bit positions that affect the next state are called the taps. The taps are XORed sequentially with the output bit and the new raw bit, and the result is then fed back into the leftmost bit. LFSRs are employed for post-processing purposes due to their good statistical properties. In this work, the raw bits are processed by a LFSR [17] whose feedback function is shown in Eq. (5).

$$\begin{aligned} f(x) = x^{32} \oplus x^{30} \oplus x^{24} \oplus x^{21} \oplus x^{20} \oplus x^{9} \\ \oplus x^{8} \oplus x^{7} \oplus x^{6} \oplus x^{2} \oplus x^{0} \end{aligned} \tag{5}$$

3 Experimental Evaluation

The considered TRNG designs were evaluated according to a set of three quality metrics: area, performance and security. Area expresses the number of LUTs and FFs occupied by the design, performance indicates how many random bits per second are produced by each batch of TRNGs and it is expressed in megabits per second, and security measures how many tests of the NIST suite are passed, as a ratio of the total number of tests. The NIST SP800-22 suite [4] consists of 15 algorithmic tests and 188 subtests, which allow evaluating the unpredictability of the output of a TRNG and therefore its suitability for cryptography applications. Performing the NIST tests required collecting 33 million random bits, split into 33 individual sequences of 1 million bits each. The NIST SP800-22 tests were carried out by employing the C implementation publicly available on the NIST website [14].

3.1 Setup

NLFIRO was evaluated with each of the three post-processing methods as well as without any conditioning, considering bit widths of 1, 8, 32, 64, and 128 bits. PLL-based TRNG was tested without any conditioning, since the raw output of the digital noise source already provided the highest security according to NIST tests, while the bit width was limited to 1 and 4 bits due to the scarce availability of PLL components on the target FPGAs. Finally, ES-TRNG was implemented both without post-processing and with LSFR, since the latter conditioning method showed the best overall metrics of the three discussed ones, with the same bit widths employed as NLFIRO.

All the implemented TRNG designs were instantiated on chips from the Xilinx Artix-7 FPGA family, in particular on the Artix-7 35 (xc7a35tcpg236-1) and 100 (xc7a100tcsg324-1) chips, targeting a clock frequency of 50 MHz.

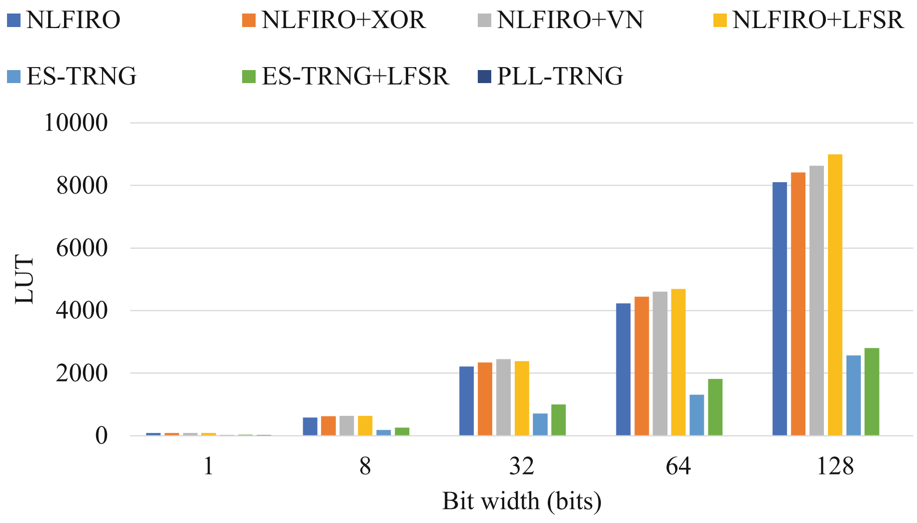

Fig. 5. Resource utilization, expressed as the number of LUTs, of TRNGs combining different digital noise sources and post-processing methods at various bit widths.

Synthesis, implementation, and generation of area reports were all carried out through Xilinx Vivado 2020.2. All designs were automatically placed and routed, without any specific constraints.

3.2 Resource Utilization

Figure 5 shows how the number of LUTs grows circa linearly with respect to the bit width for all three considered digital noise sources. NLFIRO is the TRNG which occupies the largest amount of LUTs. ES-TRNG requires a significantly smaller amount of LUTs, with a maximum of 2806 for a 128-bit LFSR post-processing configuration, but it requires instead the most FF resources, as highlighted by Fig. 6. This high amount of FF is due to the resynchronization circuit of ES-TRNG together with its control circuit, which is not necessary instead for NLFIRO. PLL-based TRNGs require the smallest amount of LUTs and FFs, requiring 21 of both resources with a 1-bit bit width. Even though PLL-based TRNGs consume minimal amounts of LUT and FF resources, it must be noted that PLLs are a very limited resource on FPGAs, and they might also be required in other components of the design.

Focusing on the post-processing methods, LFSR results in the most resource-hungry one, while XOR is the smallest conditioning architecture.

3.3 Throughput

Figure 7 shows the throughput for the considered TRNG designs. NLFIRO is the best performing digital noise sources, providing a raw random bit every clock

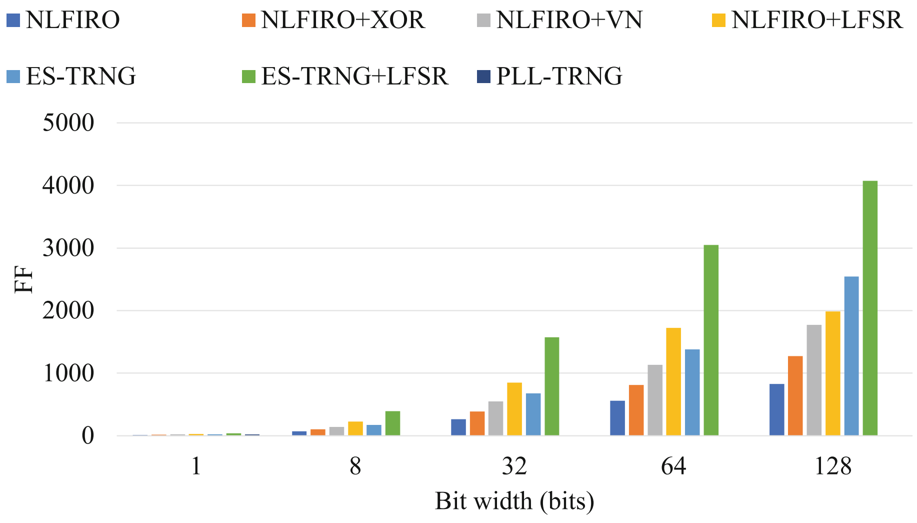

Fig. 6. Resource utilization, expressed as the number of FFs, of TRNGs combining different digital noise sources and post-processing methods at various bit widths.

cycle, eclipsing every other design, with a maximum of 6.4 Gb/s in its best configuration. On the other hand, ES-TRNGs are around 30 times slower than NLFIRO. PLL-based TRNGs have the lowest throughput, moreover their bit width is limited to 4 due to the limited amount of PLLs available on the target FPGAs.

While LFSR produces the maximum overall throughput, with no reduction compared to not applying any post-processing, the XOR and Von Neumann conditioning methods reduce instead the throughput. In particular, Von Neumann post-processing is the slowest one.

With respect to the bit width, the throughput increases linearly for all TRNGs.

3.4 Security

Figure 8 reports the experimental results related to the security metric. The bar chart expresses the number of passed NIST tests out of 15. PLL-based TRNGs ended up as the best digital noise sources from the viewpoint of the security metric, being able to pass every test even without any post-processing stage. Such good performance is due to the careful selection of PLL parameters that guarantee a high entropy sensitivity. NLFIRO manages to pass most of the tests, however not being secure enough in absence of post-processing. Finally, the raw entropy of ES-TRNG is excessively weak, with only one NIST test passed if no conditioning is applied.

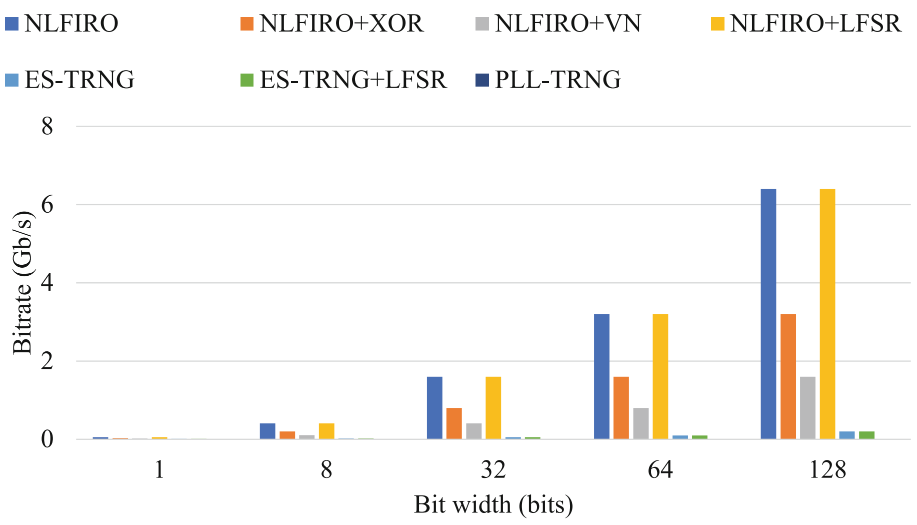

Fig. 7. Throughput, expressed as bit rate in Mb/s, of TRNGs combining different digital noise sources and post-processing methods at various bit widths.

The experimental results highlighted the critical role of the post-processing stage, which is more obvious in the ES-TRNG case. LFSR always allows all the TRNGs to pass 15/15 NIST tests and 188/188 subtests for every bit-width up to 64, while all the other conditioning method always fail at least one subtest in some configuration.

Concerning bit width, the experimental results highlighted a significant drop in security when moving from 64- to 128-bit bit width, with all the TRNGs with the larger bit width failing the majority of the NIST tests and being therefore unsuitable to cryptography applications.

3.5 Overall Results

NLFIRO-based TRNGs provide the best throughput and a high security, albeit at the cost of the largest resource utilization in terms of LUTs and FFs. ES-TRNG-based solutions occupy a smaller area, although producing a significantly smaller throughput and a reduced security, when not applying any post-processing method. PLL-based TRNGs provide the highest security without needing any conditioning, but their bit width, and therefore their throughput, is strongly limited by the amount of PLLs available on the target FPGA chip.

Concerning post-processing methods, LFSR provides the best improvement in security without any reduction in the TRNG throughput, producing however the largest resource utilization. Nevertheless, the difference in LUT and FF resources occupied compared to the other two post-processing methods or to not applying any conditioning is minimal, particularly with respect to the total

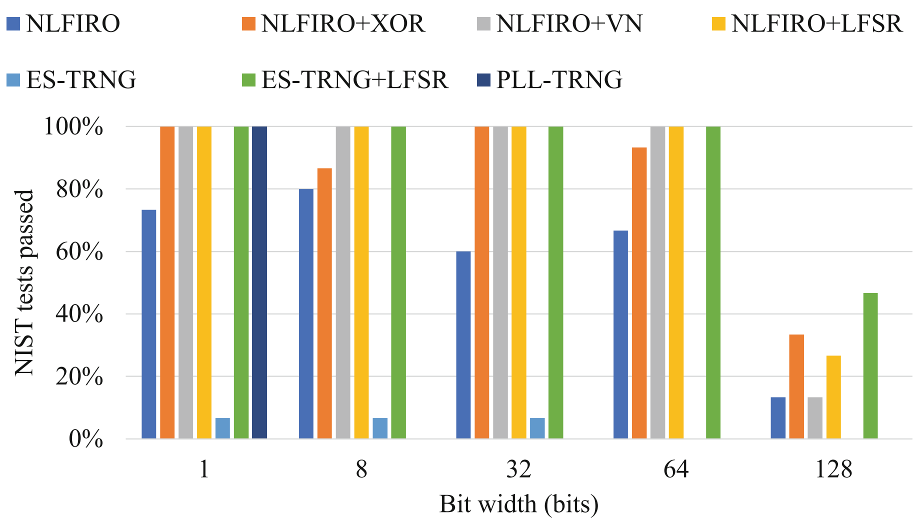

Fig. 8. Security, expressed as the percentage of NIST tests passed, of TRNGs combining different digital noise sources and post-processing methods at various bit widths.

resources available on the target chip and to the area occupied by the other components of the TRNG and of the overall design. On the contrary, applying the XOR and Von Neumann post-processing methods results in a steep reduction in the throughput metric, while not applying any conditioning provides a low security, except for PLL-TRNGs.

Finally, regarding bit-width, the experimental results highlighted a critical drop in security for bit widths larger than 64, i.e., equal to 128, which makes such TRNGs, albeit faster than those with smaller bit widths, not suitable for cryptography applications.

4 Conclusions

This work evaluated different combinations of digital noise sources and post-processing techniques to identify optimal TRNGs that are suitable for cryptographic applications on FPGA targets. In particular, we considered the NLFIRO, PLL-based TRNG and ES-TRNG digital noise sources and the XOR, Von Neumann, and LFSR post-processing methods, while also analyzing the impact of varying the clock frequency and the bit width of the TRNG.

The experimental evaluation was carried out according to three quality metrics, encompassing functional and non-functional requirements. Functional quality metrics included the security and the throughput of the TRNG, while the non-functional metric studied in this work was the resource utilization.

On the one hand, the experimental results highlighted the combination of an NLFIRO digital noise source and an LFSR post-processing method as the best-performing solution with respect to the throughput and security metrics. On the other hand, ES-TRNG-based solutions were shown to be effective choices in tightly resource-constrained scenarios, albeit with a drastic reduction in the throughput. In addition, the results validated the strong positive impact of conditioning methods on the security of the TRNG, while emphasizing the need to limit the TRNG bit width up to 64 bits to avoid a critical drop in the generated entropy.

References

1. Allini, E.N., Petura, O., Fischer, V., Bernard, F.: Optimization of the PLL configuration in a PLL-based TRNG design. In: 2018 Design, Automation Test in Europe Conference Exhibition (DATE), pp. 1265–1270 (2018). https://doi.org/10.23919/DATE.2018.8342209
2. Balasch, J., et al.: Design and testing methodologies for true random number generators towards industry certification. In: 2018 IEEE 23rd European Test Symposium (ETS), pp. 1–10 (2018). https://doi.org/10.1109/ETS.2018.8400697
3. Barenghi, A., Fornaciari, W., Pelosi, G., Zoni, D.: Scramble suit: a profile differentiation countermeasure to prevent template attacks. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. **39**(9), 1778–1791 (2020). https://doi.org/10.1109/TCAD.2019.2926389
4. Bassham, L.E., et al.: SP 800-22 rev. 1a. A statistical test suite for random and pseudorandom number generators for cryptographic applications. Technical report, NIST, Gaithersburg, MD, USA (2010)
5. Bernstein, D.J., Duif, N., Lange, T., Schwabe, P., Yang, B.Y.: High-speed high-security signatures. J. Cryptogr. Eng. **2**(2), 77–89 (2012). https://doi.org/10.1007/s13389-012-0027-1
6. Davies, R.B.: Exclusive OR (XOR) and hardware random number generators (2002). http://www.robertnz.net/pdf/xor2.pdf
7. Diffie, W., Hellman, M.: New directions in cryptography. IEEE Trans. Inf. Theory **22**(6), 644–654 (1976). https://doi.org/10.1109/TIT.1976.1055638
8. Eastlake, D.E., Crocker, S., Schiller, J.I.: Randomness requirements for security. RFC 4086 (2005). https://doi.org/10.17487/RFC4086. https://www.rfc-editor.org/info/rfc4086
9. Fischer, V., Drutarovský, M.: True random number generator embedded in reconfigurable hardware. In: Kaliski, B.S., Koç, K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 415–430. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36400-5_30
10. Galimberti, A., Montanaro, G., Zoni, D.: Efficient and scalable FPGA design of GF(2m) inversion for post-quantum cryptosystems. IEEE Trans. Comput. 1 (2022). https://doi.org/10.1109/TC.2022.3149422
11. Johnson, D., Menezes, A., Vanstone, S.: The elliptic curve digital signature algorithm (ECDSA). Int. J. Inf. Secur. **1**(1), 36–63 (2001). https://doi.org/10.1007/s102070100002
12. McEliece, R.J.: A public-key cryptosystem based on algebraic coding theory. DSN Progress Report, pp. 114–116 (1978)
13. von Neumann, J.: Various techniques used in connection with random digits. In: Householder, A.S., Forsythe, G.E., Germond, H.H. (eds.) Monte Carlo Method, National Bureau of Standards Applied Mathematics Series, vol. 12, chap. 13, pp. 36–38. US Government Printing Office, Washington, DC (1951)

14. NIST: NIST SP 800-22: download documentation and software. https://csrc.nist.gov/projects/random-bit-generation/documentation-and-software
15. Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM **21**(2), 120–126 (1978). https://doi.org/10.1145/359340.359342
16. Tao, S., Yu, Y., Dubrova, E.: FPGA based true random number generators using non-linear feedback ring oscillators. In: 2018 16th IEEE International New Circuits and Systems Conference (NEWCAS), pp. 213–216 (2018). https://doi.org/10.1109/NEWCAS.2018.8585569
17. Wu, X., Ma, Y., Chen, T., Lv, N.: A distinguisher for RNGs with LFSR post-processing. In: Wu, Y., Yung, M. (eds.) Inscrypt 2020. LNCS, vol. 12612, pp. 328–343. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-71852-7_22
18. Yang, B., Rožic, V., Grujic, M., Mentens, N., Verbauwhede, I.: ES-TRNG: a high-throughput, low-area true random number generator based on edge sampling. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2018**(3), 267–292 (2018). https://doi.org/10.13154/tches.v2018.i3.267-292. https://tches.iacr.org/index.php/TCHES/article/view/7276
19. Zoni, D., Galimberti, A., Fornaciari, W.: Efficient and scalable FPGA-oriented design of QC-LDPC bit-flipping decoders for post-quantum cryptography. IEEE Access **8**, 163419–163433 (2020). https://doi.org/10.1109/ACCESS.2020.3020262
20. Zoni, D., Galimberti, A., Fornaciari, W.: Flexible and scalable FPGA-oriented design of multipliers for large binary polynomials. IEEE Access **8**, 75809–75821 (2020). https://doi.org/10.1109/ACCESS.2020.2989423
21. Zoni, D., Barenghi, A., Pelosi, G., Fornaciari, W.: A comprehensive side-channel information leakage analysis of an in-order RISC CPU microarchitecture. ACM Trans. Des. Autom. Electron. Syst. **23**(5) (2018). https://doi.org/10.1145/3212719