Iranian Journal of Electrical and Electronic Engineering

Journal Homepage: ijeee.iust.ac.ir

A Comprehensive Assessment of Random Number Generators on FPGA: Emerging Techniques, Challenges & Future Scope

Bhagyashree Ingle*(C.A.) and Milind Nemade**

Abstract: Computing paradigm has perceived a logical shift from CPU towards application specific GPU, FPGA, CPLD due to slow down of Moore's Law, operating system overheads, serial data processing, memory management, power efficiency and speed. The performance increase of general-purpose CPUs & GPUs is unable to match with the advances in peripheral interfaces, reconfigurable logic deployed in FPGAs provides several exceptional properties that may be able to deliver desired performance. FPGA based digital circuits provide an intermediate arrangement between ASIC and CPU with regards to through-put, latency, portability and design time. True random number generators (TRNG) are expensive, low bandwidth and speed, non-compatible with FPGA or heterogenous architectures. Therefore, design and development of alternative and affordable random-number generators is focused by several researchers. TRNG design in FPGAs is more challenging because it must meet low power and area, high speed and throughput requirements without comprising the statistical quality of the desired results for intended applications. In this paper, an attempt has been made to highlight emerging techniques and challenges associated with FPGA implementation of random number generator. Furthermore, forthcoming techniques with the use of heterogeneous computation using FPGA and python productive multiprocessor system on chip (MPSoC) architecture for generation of random numbers are discussed.

Keywords: True random number generator (TRNG), Field programmable gate array (FPGA), Metastability, Heterogeneous computing, Machine learning resistant.

1 Introduction

THE Internet of things (IoT) will connect several billions of devices and generate 100 terabytes of data per year. This tremendous amount of data is generated due to explosive growth in internet and digital technology devices such as mobile phones, smart watches, tablets, laptops, smart electrical vehicles, surveillance cameras, drones, production monitoring, environment monitoring and many more [1] [2]. Large quantity of data is available that can be collected, analyzed, processed and comprehended. It enables to deploy several applications from communication interfaces, cryptography to complex analytical engines. Encryption / decryption algorithms that provides such needed security to data, completely depends on the randomness of the key. It has been possible due to advancement in the VLSI technology commencing from the transistor to application specific integrated circuits (ASIC) and digital logic gates to computing integrated circuits such as CPU, GPU, FPGA etc. Computing paradigm has perceived a logical shift from CPU towards application specific GPU, FPGA, CPLD due to slow down of Moore's Law, operating system overheads, serial data processing, memory management,

Iranian Journal of Electrical & Electronic Engineering, YYYY.
Paper first received DD MONTH YYYY and accepted DD MONTH YYYY.
* The author is Research Scholar, K. J. Somaiya Institute of Technology, Mumbai, India.
E-mail: bhagyashree.i@somaiya.edu.
** The author is with the Department of Artificial Intelligence and Data Science, K. J. Somaiya Institute of Technology, Mumbai, India.
E-mails: mnemade@somaiya.edu.
Corresponding Author: First. Author.

power efficiency and speed. FPGA based digital circuits provide an intermediate arrangement between ASIC and CPU with regards to through-put, latency, portability and design time. FPGA enables arbitrary different digital circuits such as on-chip memory, combinatorial logic blocks, digital signal processor units and logic gates to be configured through interconnect between them. This system integration on FPGA allows data flow process modelling taking complete advantage of pipeline and parallelism features. A much larger fraction of FPGAs logic can be utilized for actual operations in a dataflow fashion, minimizing data movements between subsequent compute steps. FPGAs are attractive and significant element for higher performance and energy efficiency due to custom datatypes, custom memory hierarchies and connectivity, no instructions, very deep pipelining, abundant parallelism and low power consumption.

Random numbers are used in several important security and authentication protocol applications. Therefore, hardware-based generation of random numbers is essential. A physical entropy source that uses a physical noise source and demonstrates variations in its all or some properties or characteristics is a source of random number. A small deviation in the properties of the source at a certain time gradually effects future conditions in the system. Such a random noise source can be deployed in order to generate an infinite sequence of bits, by converting the analogue values of the numbers into digital form using several techniques. If true randomness does not really exist or becomes difficult to develop or generate, then the design of unpredictable pseudorandom number generators is required. It uses random processing through which sources of uncertainty such as noise are extracted, amplified and digitized. Field-programmable gate arrays (FPGAs) appear to be useful for implementing true random number generators (TRNGs) owing to their internal architecture which provides complete support to bitwise operations. Metastability and jitter offer necessary randomness in FPGAs. It is preferable to design a TRNG that is independent of the technology used to build the FPGA and allows portability between devices. TRNG design in FPGAs is more challenging because it must meet low power and area, high speed and throughput requirements without comprising the statistical quality of the desired results for intended applications. Pseudo random number generators (PRNGs) deployed using software systems results in low quality and throughput. Furthermore, high processing time is also observed owing to highly correlated data and serial execution of programs. The system needs to be capable of generating sequences with better statistical properties [3], pass NIST randomness tests [4] [5], power density spectrum through frequency coefficient estimation [6] and entropy estimation or prediction [7] [8] [9]. Emerging techniques and challenges associated with FPGA implementation of random number generator are discussed in session two and three respectively. Furthermore, heterogeneous computation using FPGA and python productive multiprocessor system on chip (MPSoC) architecture for enabling exciting applications are discussed in future scope.

2 Emerging Techniques

A lot of scholars have been studying TRNGs lately. Device shot and thermal noises are random sources that are utilized to produce genuinely random numbers. A silicon PN diode, amplifier, pulse shaping circuit, discriminating circuit, calculating circuit, random number circuit, and controller circuit make up the conventional true random number production system depicted in figure 1 [10]. The source generating circuit, a silicon PN diode, produces a signal that is entirely dependent on particles that have been identified. Preamplifiers work by amplifying the input signal from the source to produce an amplified signal and increasing the detected input signal's magnitude. The counter circuit's job is to count the number of clock seconds that elapse between two successive pulses in the pulse signal. Based on the outcome of the conversion, the random-number output circuit can verify the binary count values produced by the counter.

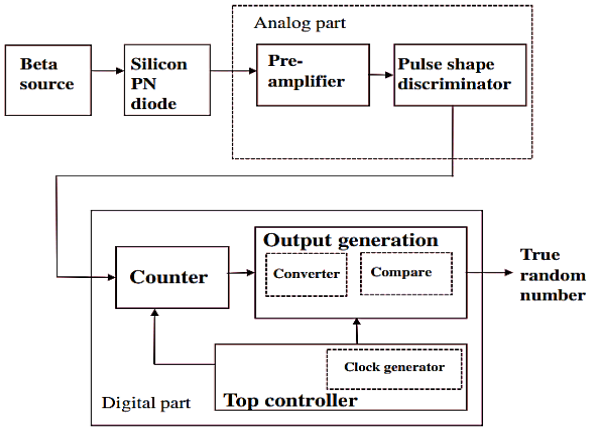

Fig. 1. True random generator conventional system

The algorithm proposed in paper [10] was designed to distinguish between generated pseudo or true random number through cumulative comparison operations. It successfully demonstrated the random number generation using Verilog, LabVIEW system and FPGA. Highly optimized digital hardware circuits can be implemented in FPGA without any associated unused circuitry especially available with controllers. However, additional hardware is required to allow reconfigurability in FPGAs, nevertheless FPGAs are finding importance in several applications. Conventional hardware description languages used are VHDL, Verilog

deployed in prototyping of ASIC. High level synthesis tools are now deployed to reduce design time and skill set through programming using C / C++. With MathWorks 2023's MATLAB HDL Coder, FPGA design is possible. Workflow can be implemented using two input methods: Simulink and MATLAB script. Understanding the I/O types is necessary for the MATLAB script to generate the HDL code. This is required for the pins on the FPGA, also all types must be made interoperable with one another. To enable the mapping of all functions onto hardware, it also needs to identify all libraries and functions that are used [11]. Moreover, heterogeneous computing is now being actively deployed for specialized applications such as graphics processing, cryptographic, compression, media codecs, etc. It is observed that the computational complexity may not always give a perfect substitute, thereby enhancing an opportunity to create an optimized hardware especially suitable for intended applications. Thus, heterogeneous computing with the use of custom reconfigurable hardware along with MPSoC may create tremendous possibility to design effective complex systems.

2.1 TRNGs using Ring Oscillators

Ring oscillator (RO) based TRNGs was demonstrated using two oscillators and D flipflop. The output of high frequency oscillator was sampled by low frequency oscillator through D flipflop. Jitter was used as a source of random noise generated due to thermal noise in oscillator depicted in figure 2 [12]. It was observed that randomness depends upon the jitter and larger frequency separation of the oscillators [13]. TRNG using two oscillators, XOR gate and D flip flop was demonstrated, it clearly showed that the XOR operation leads to better statistical properties of random sequences [14] depicted in figure 3.

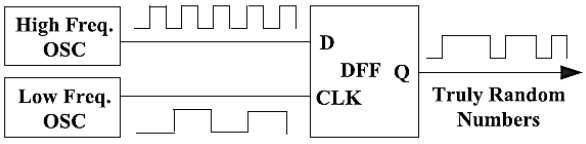

Fig. 2. Ring oscillator based TRNG

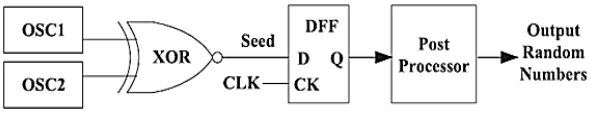

Fig. 3. TRNG using XOR operation

Tetrahedral oscillator was exploited over regular cascade inverter since the ring loop contains several fast and slow loops, perturbation of noise and large jitter. All resulting in improved metastability and randomness [15]. Post processor was deployed to enhance unpredictability and decorrelation of generated output sequences through linear feedback shift registers. The generated sequence approved the NIST [4] [5], entropy [16] and diehard tests [3]. Moreover, the TRNGs / PRNGs depends upon metastability and jitter extracted through ring oscillators, look-up-tables (LUT) and programmable delay lines (PDL) [17] [18] [19] [20] [21]. Availability of digital clock managers (DCM) on recent FPGAs have paved the way for implementation of TRNGs [22] [23]. Programmable frequency clock signals can be obtained using DCM on Xilinx FPGAs [22] [23] [24] and intel chips [24]. Dynamic Partial Reconfiguration port (DRP) of DCMs is incorporated to manipulate parameters, thereby tuning frequency of clock signals [24]. The use of Xilinx DCM to introduce metastability in at least one of the flip flops using two input signals and output clock was demonstrated [24]. The use of carry4 hardware primitive demonstrated enhance randomness along with DSP based post processing scheme that maintains TRNG throughput. The Xilinx Zynq XC7Z020 System on Chip (SoC) is used to implement the suggested TRNG design. DRP of DCM facility is available with Xilinx FPGAs that makes other FPGAs unavailable for implementation clearly indicating importability. Several methods based on FPGA implementation of PRNGs through oscillator rings were analyzed to determine sampling frequency and its efficacy. It was observed that circuit using 48 inverters, 17 D flip flops (DFFs), 31 XORs, and about 100 routing resources generated better randomness [25]. LUT were combined with shift registers and RAM blocks available in FPGAs to achieve randomness [26] [27]. Beat frequency detection (BFD) technique was successfully demonstrated for generation of random numbers that included two oscillators, D flip flop and counters [27] [28] depicted in figure 4 [29].

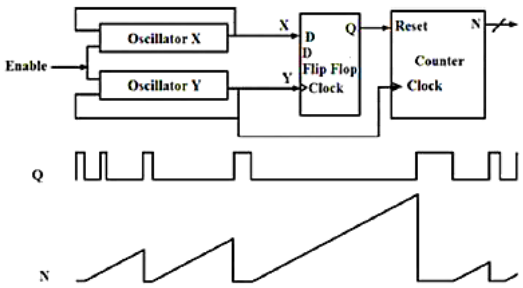

Fig. 4. BFD based TRNG architecture

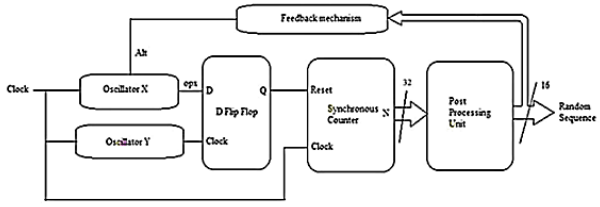

Fig. 5. BFD PRNG architecture using feedback mechanism

The BFD technique based PRNG through XOR gates, D flip flop, synchronous counter, feedback mechanism and post-processing unit was demonstrated with low power dissipation, requires low hardware footprints and passes the entire NIST 800-22 statistical test suite depicted in figure 5 [30]. However, post processing technique along with feedback mechanism was incorporated to achieve better randomness properties [30].

2.2 TRNGs using Chaotic Systems

Chaotic non-linear systems demonstrate physical characteristics namely ergodicity, high sensitivity to initial conditions, true random-like behaviors and broadband. Chaotic systems can be classified into discrete chaotic systems and continuous chaotic systems. The discrete chaotic maps consist of logistic map, the sine map, the tent map, etc. whereas continuous chaotic systems include Lorenz system, Chua's circuit, etc. These chaotic systems are further classified as high dimensional [31] and low dimensional chaos [32] based on complexity. Most of the low dimensional chaotic PRNGs are insecure due to reduced key space and are susceptible to brute force attack [33]. Therefore, hyperchaotic systems that exhibits complex chaotic behaviors, bigger key space and makes it more difficult to predict time series generated numbers are preferred. Bifurcation diagram and the trajectory illustrate the behavior of the chaotic systems. The bifurcation diagram of the 1D logistic map, the sine map, the Henon map demonstrates the small chaotic interval figure 6 [34].

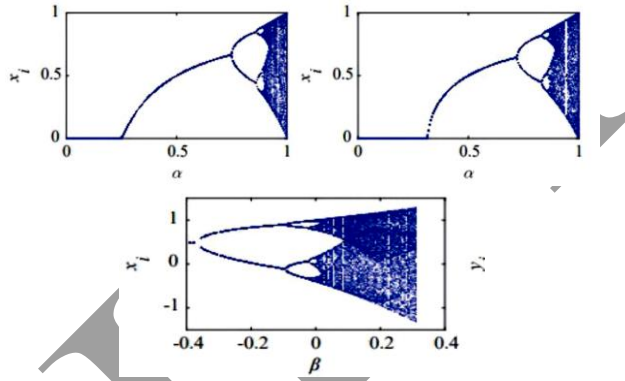

Fig. 6. 1D Logistic map, Sine map and Henon map

Moreover, their chaotic range is noncontinuous that clearly indicates lesser randomness for useful engineering applications [34]. Therefore, it is necessary to evaluate higher dimensional chaotic systems for practical applications. It is worth to mentioned that besides software realization, hardware implementation of chaos is equally realizable [35] [36].

Additionally, two issues are mostly observed in chaotic TRNGs, firstly the effects of finite precision due to use of embedded digital processors and secondly insufficient evaluation criteria resulting in a chaotic system to degenerate into a periodic function. Also, hyperchaotic system bearing higher Lyapunov exponent is desired for cryptographic applications refer table 1 [35] [37] [38] [39] [40] [41]. A new 5-D hyperchaotic dynamo system by introducing two feedback controllers to the existing Rikitake 2-disk dynamo system was demonstrated [44], FPGA implementation was attempted that resulted in better image encryption parameters and better cryptographic parameters. The self-shrinking generator realized through linear feedback shift registers was fused with hyperchaotic system that results in better statistical and cryptographic properties [33]. Several hyperchaotic systems are being proposed for image encryption applications but found to be inefficient in the process of encryption due to weak permutation and diffusion process [45] [46] [47]. Hyperchaotic systems have more complex behaviors and better random-like characteristics than general chaotic systems. Chaotic systems are computationally complex and needs more hardware resources.

Table 1. Lyapunov exponents

Hyperchaotic system	λ1	λ2	λ3	λ4
[33]	2.199	0.071	0	−14.362
[42]	1.402	0.2731	0	−12.863
[43]	0.648	0.153	0	−38.468
[44]	0.1664	0.0335	0	-2.1991

2.3 Gaussian TRNGs Systems

Gaussian random number generators (GRNGs) are mostly applied in advance artificial intelligence, Monte Carlo simulations, cryptographic algorithms, radio frequency identification etc. [48] [49] Box Mullers transformation based random number generators are mostly pseudo random number generators that generates gaussian random numbers through transformation techniques [50] [51] figure 7.

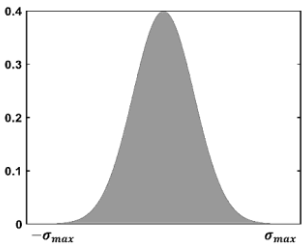

Fig. 7. Gaussian probability distribution function

Major objectives for FPGA implementation of GRNGs is minimum resource utilization, low power dissipation, reconfigurability and efficiency. FPGA-based designs for GRNGs can be categorized as Cumulative Distribution Function (CDF) inversion [52], M Python [53], Ziggurat [54], Central Limit Theorem (CLT) [55], Wallace [56], Box–Muller [50], and Multihat [57]. Conventional Box-Muller based GRNGs is depicted in figure 8. It consists of LFSR based uniform random number generator, block RAMs for three functions and

two multipliers. This conventional method is easy to implement but at the cost of low accuracy [58]. However, the size of the block RAMs increases exponentially due to increase in maximum variance. Therefore, several measures were proposed such as Wallace [56], polynomial approximation and central limit theorem [55] by introducing a greater number of functions. The idea was extended to segmented Box-Muller by stepwise segmenting the interval (0,1) in two parts [50]. Similarly, the idea can be extended to n-segments, thereby generating uniform random numbers R1, R2, …, Rn that may reduce the size of block RAMs effectively. The implementation accuracy and efficiency were proved to be better as compared with other published results [50].

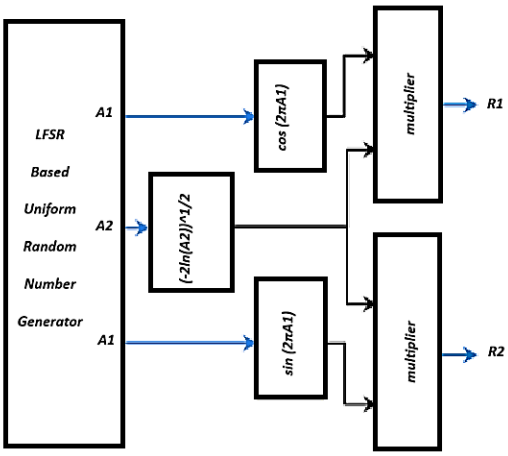

Fig. 8. Conventional Box-Muller based GRNGs

2.4 Machine Learning Resistance PRNGs

PRNGs are more vulnerable to machine learning attacks as compared with TRNGs. Long short-term memory (LSTM) based machine learning technique has been successfully demonstrated in cracking PRNGs [59] [60]. ML attacks can also be used to breach hardware security primitives such as "Physical Unclonable Functions (PUF)" [61]. But the primary security distinction between PUFs and conventional PRNGs is that a typical PRNG can be broken simply by training its output data. A new architecture composed of a PUF and two regular PRNGs is proposed: PRNG1 and PRNG2 [59]. First, the output of PRNG1 is fed into R1 "PUF" to generate an intermediate data R2. The intermediate data R2 is added to the R3 output of the PRNG2 to produce the final R4 output of the new PRNG. Since the input challenge of the PUF in figure 9 is hidden in the R1 chip, the adversary cannot model the secret information of the PUF even using ML techniques [59]. In the new PRNG, the PUF input challenge was masked so that the adversary could not model all elements. When a LSTM attack [62] carried out on the suggested PRNG, the training precision is approximately 52%. In contrast, if the LSTM attack is implemented on a normal PRNG, its training accuracy will be above 99% [59]. However, these PUFs are vulnerable to modelling attacks [63] by application of machine learning algorithms that predicts the response of PUFs [64]. Therefore, it is desired to build PUFs circuits with added complexities such as encryption, hashing, etc that protects from machine learning attacks [65] [66] [67]. An attack on FPGA based PUFs was demonstrated that classified '0' and '1' responses even when encrypted [68] [69]. Hence FPGA implementation of side channel attacks based on deep learning needs bitstream modification by combining them with responses [70].

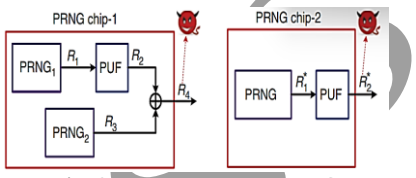

Fig. 9. ML resistant PRNG

3 Discussion & Challenges

Pseudorandom number generators (PRNGs), which use a deterministic process to extend a brief random string into a set of random looking numbers, are adequate for many purposes. PRNGs are divided into two parts cryptographic and non-cryptographic. For cryptographic applications, however, it is critical to create pseudo random bits that are unpredictably recognised even by the most powerful attacker. Furthermore, pseudo random number generator with quality output which is tolerance to attacks can be built, which is called TRNG. PRNGs uses a known algorithm to generate a huge number, which are completely random from a small quantity of unpredictability numbers called seed. The linear congruential generators present in many programming libraries are noncryptographic PRNGs. These libraries statistically acceptable random numbers, but they can't use for cryptographic keying applications. This PRNGs takes input as seed and gives output. The seed used for PRNG is mathematically derived so we can derive the internal state and output of PRNG easily. Attacker can easily predict the outputs if he knew previous outputs therefore security problems are associated with PRNG. A nondeterministic irregular bit generator employment a nondeterministic source to produce randomness. Most of the randomness (random seed) produced by unpredictable natural process such as atmospheric noise, jitter noise and nuclear decay. The major performance of TRNG depends upon the quality of seed produced by entropy source. We can create randomness from within deterministic system. Computers which don't have hardware entropy source try to obtain seed from embedded devices of hardware like keyboard and hard drives, memory, etc. The entropy derived from the sources like keyboard is not at all sufficient, so it is better to have hardware entropy source. Computationally bounded test that can assess a RNG's output and

authoritatively affirm that the output is random. Entropy source should produce output regularly with some frequency. However, some software system stores a seed value in hardware which have security problems, since there are some problems with entropy source. Hardware that provides a well-designed, efficient, and easy-to-use hardware entropy source is the greatest answer to these problems. Random number generators are used to determine optimal input random variables in cryptography, communications, industrial equipment and measurements, Monte Carlo simulations, random filling, mobile and computer games, laboratory testing, application of evolutionary algorithms to characterize networks. Moreover, they are used in applications such as games, lotteries, statistical data analysis, sampling, simulations and testing machine learning algorithms. Technologies generating true or pseudo random numbers needs to be developed in areas such as secure communications, security systems and online banking.

The important observations identified after detailed literature review in FPGA implementation of PRNGs are depicted herewith. Computing paradigm has perceived a logical shift from CPU towards application specific GPU, FPGA, CPLD due to slow down of Moore's Law, operating system overheads, serial data processing, memory management, power efficiency and speed. The increase in performance parameters of general-purpose CPUs & GPUs is unable to match with the newly adopted peripheral interfaces. FPGA based digital circuits provide an intermediate arrangement between ASIC and CPU with regards to through-put, latency, portability and design time. FPGA implementation forms the basis of ASIC implementation. True random number generators are expensive, low bandwidth and speed, non-compatible with FPGA or heterogenous architectures. Ring oscillators deployed using inverters are mostly appropriate for low power and area entitlements and are robust to 1/f noise. Asynchronous modulo counter can be successfully replaced with ring oscillators and PLL to explore device independent operation. LUT-FIFO TRNG uses RAM blocks available in FPGA are used to generate high quality random bit streams however LUT-SR TRNGs do not generate good quality random bitstreams as compared to LUT-FIFO TRNGs. DPR & DCM facilities are available with Xilinx but not in an Altera FPGAs that clearly implies importability. Feedback mechanism from the generated random number enhances metastability. Post processing scheme may be deployed to enhance the statistical characteristics of the generated random numbers. The comparison of TRNGs published in the literature through coherent sampling with self-timed rings and beat frequency oscillators is illustrated in table 2. Hyperchaotic systems have more complex behaviors and better random-like characteristics than general chaotic systems. Chaotic systems are computationally complex and needs more hardware resources. It is observed that the complexity involved is much more that results in lower throughput and higher power dissipation therefore chaos-based TRNGs / PRNGs may not be applied in limited power applications. Table 3 illustrates the comparison of PRNG techniques employed for hardware implementation. Conventional pseudo-random number generator (PRNG) is vulnerable to machine learning (ML) attacks since algorithms are used to generate the random number. FPGA implementation of random number generator for accelerating edge computing algorithms for big data analytics is necessary. Post processing scheme may be deployed to enhance the statistical characteristics of the generated random numbers especially machine learning resistant. Table 4 illustrates PRNG techniques that can be selected for implementation based on the deployed applications.

Table 2. TRNGs comparison

Parameters	[12]	[71]	[72]	[30]	[24]
Method	RO	BFD	Modi-RO	BFD	DCM
LUTs	32.0	50.0	160.0	29.0	38.0
Registers	48	33	19	20	38
Entropy	0.998	0.931	0.982	0.973	0.998
Complexities	M	Low	Low	Low	M
Portability	Yes	Yes	Yes	Yes	Yes
Tunability	No	Yes	No	Yes	No
Throughput (Mb/s)	4.0	5.0 – 25.0	4.0	5.0 – 25.0	100.0

M: Medium

Table 3.a Comparison of PRNG techniques for hardware implementation

Ref.	Techniques	Speed	Hardware Complexity
[26]	LFSR	High	Low
[73]	Cellular Automata	Medium	Medium
[74]	Fibonacci/Galois LFSR	High	Medium
[49]	Chaos-Based PRNG	Medium	High
[12]	Ring Oscillator Hybrid	Medium	Medium

Table 3.b Comparison of PRNG techniques for security and its applications

Ref.	Techniques	Security	Applications
[26]	LFSR	Low	FPGA, DSP, CRC, ECC
[73]	Cellular Automata	Medium	Image processing, cryptography
[74]	Fibonacci/ Galois LFSR	Medium	Wireless communication, cryptography
[49]	Chaos-Based RNG	High	Secure communication, ML-resistant PRNG
[12]	Ring Oscillator	High	Hardware security, cryptography

Hybrid

Table 4. PRNG technique selection based on applications

Applications	PRNG Choice
High-Speed FPGA/ASIC Systems	LFSR, Cellular Automata (VLSI-Based)
Low-Power IoT & Embedded Systems	LFSR + Chaos-Based PRNG (VLSI-Based)
Machine Learning Security	GAN-Based PRNG, RL-Based PRNG (ML-Based)
Cryptographic Key Generation	Chaos-Based PRNG or ML-Based PRNG
Hardware Security Modules (HSMs)	Hybrid PRNG (Ring Oscillator + Chaos or ML-Based)

4 Future Scope

Currently FPGAs are being deployed to embrace heterogeneity with the aim of providing high performance and energy efficiency. FPGAs have to be programmed through hardware description language VHDL, Verilog, etc. that completely described the circuits designed to execute a particular task. FPGAs are attractive and significant element for higher performance and energy efficiency due to custom datatypes, custom memory hierarchies and connectivity, no instructions, very deep pipelining, abundant parallelism and low power consumption. FPGA enables arbitrary different digital circuits such as on-chip memory, combinatorial logic blocks, digital signal processor units and logic gates to be configured through interconnect between them. Tremendous increase in data and operations, general purpose processors are unable to process the information in real time. Hardware accelerator based on reconfigurable logic such as FPGA can be used to accelerate the some of the tasks and decrease total execution time. These tasks can be meticulously evaluated for execution on hardware (FPGAs) and software (MPSoC) to achieve desired performance. The execution speeds up to 3X has been demonstrated for an integrated platform that host FPGA and MPSoC on same devices. It was achieved through tight communication mechanism between the processor and the reconfigurable logic. Thus, the post processing schemes that may provide much needed machine learning resistant features can be tightly implemented using python productive MPSoC and the TRNG may be hosted on reconfigurable logic.

5 Conclusion

In this paper, an attempt has been made to discuss emerging techniques and challenges associated with FPGA implementation of true / pseudo random number generator along with its future scope. FPGA based digital circuits provide an intermediate arrangement between ASIC and CPU with regards to through-put, latency, portability and design time. True random number generators are expensive, low bandwidth and speed, non-compatible with FPGA or heterogenous architectures. Chaotic systems are computationally complex and needs more hardware resources. Segmented Box-Muller method may be deployed for generation of GRNGS. It is desired to build PUFs circuits with added complexities such as encryption, hashing, etc. that protects from machine learning attacks. Post processing scheme through heterogeneous computing using python productive MPSoC may be deployed to enhance the statistical characteristics of the generated random numbers especially machine learning resistant.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Bhagyashree Ingle the corresponding author's contribution was to accomplish literature survey, writing and editing of the paper. The work was completed under the guidance of the author Milind Nemade and contributed in the overall quality improvement of the paper.

Declaration of generative AI and AI-assisted technologies

The author(s) declare that no generative AI or AI-assisted technologies were used in the writing process of this manuscript.

References

[1] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, "Handbook of Applied Cryptography", Chapter 3, CRC Press, pp. 87–131, 1996.

[2] A. Vassilev and T. A. Hall, "The importance of entropy to information security," Computer, Vol. 47, no. 2, pp. 78–81, Feb. 2014.

[3] G. Marsaglia, The diehard test suite, 2003. Available at http://www.csis.hku.hk/˜dieha rd/

[4] 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," National Institute of Standards & Technology (NIST), Apr. 2010.

Available: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-22r1a.pdf

[5] M. S. Turan et al., "Recommendation for the entropy sources used for random bit generation", (Second DRAFT) NIST Special Publication 800–90B, Jan. 2016.

Available: http://csrc.nist.gov/publications/drafts/800-90/sp800 -90b_second_draft.pdf

[6] P. Grassberger, "Entropy estimates from insufficient samplings", arXiv:physics/0307138, 2003.

[7] D. H. Wolpert and D. R. Wolf, "Estimating functions of probability distributions from a finite set of samples, part 1: Bayes estimators and the Shannon entropy", arXiv preprint comp-gas/9403001, 1994.

[8] P. Hagerty and T. Draper, "Entropy bounds and statistical tests", http://csrc.nist.gov/groups/ST/rbg_workshop_2012/hagerty_entropy_paper.pdf.

[9] J. Kelsey, K. A. McKay, and M. S. Turan, "Predictive models for min-entropy estimation in Cryptographic Hardware and Embedded Systems", Proceedings International Workshop, Saint-Malo (Saint-Malo, France), pp. 373–392, Sept. 2015.

[10] Seongmo Park, Byoung Gun Cho, Taewook Kang, Kyunghwan Park, Youngsu Kwon and Jongbum Kim, "Efficient hardware implementation and analysis of true random number generator based on beta source," ETRI Journal, Vol. 42, no. 4, pp. 518–526, 2020.

[11] Ghada Elsayed, Elsayed Soleit and Somaya Kayed, "FPGA design and implementation for adaptive digital chaotic key generator," Springer Bulletin of the National Research Centre, Vol. 47, no. 122, pp. 1 – 9, 2023.

[12] Dongsheng Liu, Zilong Liu, Lun Li, and Xuecheng Zou, "A Low-Cost Low-Power Ring Oscillator-Based Truly Random Number Generator for Encryption on Smart Cards", IEEE Transactions on Circuits and Systems-II: Express Briefs, Vol. 63, No. 6, pp. 608 - 612, 2016.

[13] C. S. Petrie and J. A. Connelly, "Modelling and simulation of oscillator based random number generators," in Proceedings IEEE International Symposium on Circuits and Systems (ISCAS), pp. 324–327, 1996.

[14] K. Wold and C. H. Tan, "Analysis and enhancement of random number generator in FPGA based on oscillator rings," International Journal on Reconfigurable Computing, Vol. 2009, no. 4, pp. 1–8, Jan. 2009.

[15] E. J. Pankratz and E. Sánchez-Sinencio, "Multiloop high power supply rejection quadrature ring oscillator," IEEE Journal Solid-State Circuits, Vol. 47, no. 9, pp. 2033–2048, Sep. 2012.

[16] S. V. Suresh and W. P. Burleson, "Entropy and energy bounds for metastability based TRNG with lightweight post-processing," IEEE Transaction Circuits System I, Reg. Papers, Vol. 62, no. 7, pp. 1785–1793, Jul. 2015.

[17] H. Martin, P. Peris-Lopez, J. E. Tapiador, and E. San Millan, "A new TRNG based on coherent sampling with self-timed rings," IEEE Transaction on Industrial Informatics, Vol. 12, no. 1, pp. 91–100, Feb. 2016.

[18] N. N. Anandakumar, S. K. Sanadhya, and M. S. Hasmi, "FPGA based true random number generation using programmable delays in oscillator-rings," IEEE Transaction on Circuits Systems II, Exp. Briefs, Vol. 67, no. 3, pp. 570–574, Mar. 2020.

[19] K. Demir and S. Ergün, "Random number generators based on irregular sampling and Fibonacci–Galois ring oscillators," IEEE Transaction on Circuits Systems II, Exp. Briefs, Vol. 66, no. 10, pp. 1718–1722, Oct. 2019.

[20] X. Li, P. Stanwicks, G. Provelengios, R. Tessier, and D. Holcomb, "Jitter-based adaptive true random number generation for FPGAs in the cloud," in Proceedings of International Conference on Field-Programmable Technology (ICFPT), pp. 112–119, 2020.

[21] S. Yang et al., "Lightweight Hybrid Entropy Source True Random Number Generator Based on Jitter and Metastability," in IEEE Transactions on Circuits and Systems II: Express Briefs, Vol. 71, no. 7, pp. 3513-3517, July 2024.

[22] A. P. Johnson, R. S. Chakraborty, and D. Mukhopadyay, "An improved DCM-based tunable true random number generator for Xilinx FPGA," IEEE Transaction Circuits and Systems II, Express Briefs, Vol. 64, no. 4, pp. 452–456, Apr. 2017.

[23] N. Fujieda, M. Takeda, and S. Ichikawa, "An analysis of DCM-based true random number generator," IEEE Transactions on Circuits and Systems II, Exp. Briefs, Vol. 67, no. 6, pp. 1109–1113, Jun. 2020.

[24] Fabio Frustaci, Fanny Spagnolo, Stefania Perri and Pasquale Corsonello, "A High-Speed FPGA-Based True Random Number Generator Using Metastability with Clock Managers," IEEE Transactions on Circuits and Systems-II: Express Briefs, Vol. 70, No. 2, pp. 756 – 760, 2023.

[25] X. Xu and Y. Wang, "High speed true random number generator based on FPGA," in International

Conference on Information Systems Engineering (ICISE), pp. 18-21, 2016.

[26] D. B. Thomas and W. Luk, "The LUT-SR family of uniform random number generators for FPGA architectures," IEEE Transactions on Very Large-Scale Integration (VLSI) Systems, Vol. 21, No. 4, pp. 761–770, 2013.

[27] D. B. Thomas and W. Luk, "High quality uniform random number generation using LUT optimized state-transition matrices," Journal of VLSI Signal Processing, Vol. 47, No. 1, pp. 77–92, 2007.

[28] A. Marghescu, P. Svasta, and E. Simion, "High speed and secure variable probability pseudo/true random number generator using FPGA," in IEEE 21st International Symposium on Design and Technology in Electronic Packaging, pp. 323–328, 2015.

[29] Q. Tang, B. Kim, Y. Lao, K. K. Parhi, and C. H. Kim, "True random number generator circuits based on single- and multi-phase beat frequency detection," in Proceedings of IEEE Custom Integrated Circuits Conference, pp. 1–4, 2014.

[30] G. Morankar, "Low Power True Random Number Generator Through Ring Oscillator for IoT and Smart Card Applications," Iranian Journal of Electrical & Electronics Engineering, Vol. 17, no. 3, 2021.

[31] Y. Li, C. Li, S. Liu, Z. Hua, and H. Jiang, ''A 2-D conditional symmetric hyperchaotic map with complete control,'' Nonlinear Dyn., Vol. 109, no. 2, pp. 1155–1165, Jul. 2022.

[32] A. Mansouri and X. Wang, ''A novel one-dimensional chaotic map generator and its application in a new index representation-based image encryption scheme,'' Information Science, Vol. 563, pp. 91–110, Jul. 2021.

[33] Yang Liu, Xiaojun Tong, "Hyperchaotic system-based pseudorandom number generator," IET Information Security, Vol. 10, no. 6, pp. 433 – 441, 2016.

[34] F. Zhang, J. Tang, Z. Zhang, Z. Huang and T. Huang, "An Improved Absolute-Cosine Chaotification Model and Its Simple Application in PRNG," IEEE Access, Vol. 11, pp. 59346 - 59356, 2023.

[35] Swathy, P.S., Thamilmaran, K., "Hyperchaos in SC-CNN based modified canonical Chua's circuit," Nonlinear Dynamics, Vol. 78, no. 4, pp. 2639–2650, 2014.

[36] Kapitaniak, T., Chua, L.O., Zhong, G.Q., "Experimental hyperchaos in coupled Chua's circuits", IEEE Transaction on Circuits and Systems I, Vol. 41, no. 7, pp. 499–503, 1994.

[37] Li, J. H., Liu, H., "Colour image encryption based on advanced encryption standard algorithm with two-dimensional chaotic map", IET Information Security, Vol. 7, no. 4, pp. 265–270, 2013.

[38] Barakat, L. M., Mansingka, A.S., Radwan, A. G., "Hardware stream cipher with controllable chaos generator for colour image encryption", IET Image Processing, Vol. 8, no. 1, pp. 33–43, 2014.

[39] Kocarev, L., "Chaos-based cryptography: a brief overview", IEEE Circuits and Systems Magazine, Vol. 1, no. 3, pp. 6–21, 2001.

[40] Dachselt, F., Schwarz, W., "Chaos and cryptography", IEEE Transaction on Circuits and Systems I, Vol. 48, no. 12, pp. 1498–1509, 2001.

[41] P. S. Paul, M. Sadia, M. R. Hossain, B. Muldrey and M. S. Hasan, "Cascading CMOS-Based Chaotic Maps for Improved Performance and Its Application in Efficient RNG Design," IEEE Access, Vol. 10, pp. 33758-33770, 2022.

[42] Liu, M.H., Feng, J.C., Tse, C.K.: 'A new hyperchaotic system and its circuit implementation', International Journal on Bifurcation Chaos, Vol. 20, no. 4, pp. 1201–1208, 2010.

[43] Wang, J. Z., Chen, Z. Q., Yuan, Z. Z., "The generation of a hyperchaotic system based on a three-dimensional autonomous chaotic system", Chinese Physics, Vol. 15, no. 6, pp. 1216–1225, 2006.

[44] S. Vaidyanathan et al., "A 5-D Multi-Stable Hyperchaotic Two-Disk Dynamo System with No Equilibrium Point: Circuit Design, FPGA Realization and Applications to TRNGs and Image Encryption," IEEE Access, Vol. 9, pp. 81352-81369, 2021.

[45] Jeng, F.G., Huang, W.L., Chen, T.H., "Cryptanalysis and improvement of two hyper-chaos- based image encryption schemes", Signal Processing, Image, Vol. 34, pp. 45–51, 2015.

[46] Özkaynak, F., Özer, A.B., Yavuz, S., "Cryptanalysis of a novel image encryption scheme based on improved hyperchaotic sequences", Optical Communication, Vol. 285, no. 24, pp. 4946–4948, 2012.

[47] Xie, T., Liu, Y.S., Tang, J., "Breaking a novel image fusion encryption algorithm based on DNA sequence operation and hyper-chaotic system", Optical International Journal of Light Electron Optics, Vol. 125, no. 24, pp. 7166–7169, 2014.

[48] R. Serrano, C. Duran, T. Hoang, M. Sarmiento, K. Nguyen, A. Tsukamoto, K. Suzaki, and C. Pham, ‘‘A fully digital true random number generator with entropy source based in frequency collapse,’’ IEEE Access, Vol. 9, pp. 105748–105755, 2021.

[49] B. Paul, S. Pal, A. Agrawal, and G. Trivedi, ‘‘Triple pendulum based nonlinear chaos generator and its applications in cryptography,’’ IEEE Access, Vol. 10, pp. 127073–127093, 2022.

[50] Samir Dahmani, et. al., “An Efficient FPGA-Based Gaussian Random Number Generator Using an Accurate Segmented Box–Muller Method,” IEEE Access, Vol. 11, pp. 64745 – 64757, 2023.

[51] M. Maamoun, H. A. Saadi, S. Dahmani, G. Zerari, N. Chabini, and R. Beguenane, ‘‘An optimized FPGA based Box–Müller Gaussian random number generator architecture for communication applications,’’ in Proceedings IEEE 12th Annual Information Technology, Electronic Mobile Communication Conference, pp. 772–777, Oct. 2021.

[52] R. Gutierrez, V. Torres, and J. Valls, ‘‘Hardware architecture of a Gaussian noise generator based on the inversion method,’’ IEEE Transaction on Circuits and Systems II, Exp. Briefs, Vol. 59, no. 8, pp. 501–505, Aug. 2012.

[53] Y. Li, P. Chow, J. Jiang, M. Zhang, and S. Wei, ‘‘Software/hardware framework for generating parallel Gaussian random numbers based on the Monty Python method,’’ in Proceedings International Conference on Field-Programmable Technology, pp. 190–197, Dec. 2012.

[54] Jianing Su and Jun Han, "An improved Ziggurat-based hardware Gaussian random number generator," 13th IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT), Hangzhou, pp. 1606-1608, 2016.

[55] I. Syafalni, G. Jonatan, N. Sutisna, R. Mulyawan, and T. Adiono, ‘‘Efficient homomorphic encryption accelerator with integrated PRNG using low-cost FPGA,’’ IEEE Access, Vol. 10, pp. 7753–7771, 2022.

[56] D.-U. Lee, W. Luk, J. D. Villasenor, G. Zhang, and P. H. W. Leong, ‘‘A hardware Gaussian noise generator using the Wallace method,’’ IEEE Transaction on Very Large Scale Integrated (VLSI) Systems, Vol. 13, no. 8, pp. 911–920, Aug. 2005.

[57] J. S. Malik and A. Hemani, ‘‘Gaussian random number generation: A survey on hardware architectures,’’ ACM Computer Survey, Vol. 49, no. 3, pp. 1–37, Sep. 2017.

[58] D. B. Thomas, "FPGA Gaussian Random Number Generators with Guaranteed Statistical Accuracy," 2014 IEEE 22nd Annual International Symposium on Field-Programmable Custom Computing Machines, Boston, MA, USA, pp. 149-156, 2014.

[59] Yiming Wen and Weize Yu, “Machine learning-resistant pseudo-random number generator,” IEEE Electronics Letters, Vol. 55, No. 9, pp. 515–517, 2019.

[60] Fischer, T., “Testing cryptographically secure pseudo random number generators with artificial neural networks”, Proc. IEEE International Conference on TrustCom/BigDataSE, New York, NY, USA, August 2018, pp. 1214–1223.

[61] A. P. Johnson, R. S. Chakraborty, and D. Mukhopadhyay, “A PUF-enabled secure architecture for FPGA-based IoT applications,” IEEE Transaction on Multi-Scale Computing System, Vol. 1, No. 2, pp. 110 – 112, 2015.

[62] Jeong, Y.-S., Oh, K., Cho, C.-K., et al. “Pseudo random number generation using LSTMs and irrational numbers”, IEEE International Conference on Big Data and Smart Computing, Shanghai, China, pp. 541–544, 2018.

[63] J. Shi, Y. Lu and J. Zhang, "Approximation attacks on strong PUFs", IEEE Transaction Computer Aided Design Integrated Circuits System, Vol. 39, no. 10, pp. 2138-2151, Oct. 2020.

[64] F. Ganji, S. Tajik and J. P. Seifert, "Why attackers win: On the learnability of XOR arbiter PUFs", Proceedings International Conference Trust Trustworthy Computer, pp. 22-39, 2015.

[65] T. A. Idriss, H. A. Idriss and M. A. Bayoumi, "A Lightweight PUF-Based Authentication Protocol Using Secret Pattern Recognition for Constrained IoT Devices," in IEEE Access, Vol. 9, pp. 80546-80558, 2021.

[66] S. Li, T. Zhang, B. Yu and K. He, "A provably secure and practical PUF-based end-to-end mutual authentication and key exchange protocol for IoT", IEEE Sensors Journal, Vol. 21, no. 4, pp. 5487-5501, Feb. 2021.

[67] V. P. Yanambaka, S. P. Mohanty, E. Kougianos and D. Puthal, "PMsec: Physical unclonable function-based robust and lightweight authentication in the Internet of medical things", IEEE Transaction on Consumer Electronics, Vol. 65, no. 3, pp. 388-397, Aug. 2019.

[68] E. Dubrova, O. Näslund, B. Degen, A. Gawell, and Y. Yu, “CRC-PUF: A Machine Learning Attack Resistant Lightweight PUF Construction”, in 2019

IEEE European Symposium on Security and Privacy Workshops, pp. 264–271, 2019.

[69] Y. Yu, M. Moraitis, and E. Dubrova, “Can Deep Learning Break a True Random Number Generator?”, IEEE Transactions on Circuits and Systems - II - Express Briefs, Vol. 68, no. 5, pp. 1710–1714, 2021.

[70] Y. Yang, M. Moraitis, and E. Dubrova, “Why Deep Learning Makes it Difficult to Keep Secrets in FPGAs”, in DYNAMICS ’20: Proceedings of the 2020 Workshop on Dynamic and Novel Advances in Machine Learning and Intelligent Cyber Security, pp. 1–9, 2022.

[71] G. Morankar, “True random number generator through beat frequency oscillators in FPGA,” Helix Journal, Vol. 8, No. 4, pp. 3442–3447, 2018.

[72] M. A. Şarkışla and S. Ergün, “An area efficient true random number generator based on modified ring oscillators,” in IEEE Asia Pacific Conference on Circuits and Systems (APCCAS), Chengdu, pp. 274 - 278, 2018.

[73] Y. Luo, W. Wang, S. Best, Y. Wang and X. Xu, "A High-Performance and Secure TRNG Based on Chaotic Cellular Automata Topology," in IEEE Transactions on Circuits and Systems I: Regular Papers, Vol. 67, no. 12, pp. 4970-4983, Dec. 2020.

[74] W. Li and X. Yang, "A Parallel and Reconfigurable United Architecture for Fibonacci and Galois LFSR," 2015 7th International Conference on Intelligent Human-Machine Systems and Cybernetics, Hangzhou, China, pp. 203-206, 2015.

Biographies

Bhagyashree Ingle is currently working as Assistant Professor in the Department of Information Technology, K. J. Somaiya Institute of Technology, Mumbai, with seven years of academic and research experience. She is pursuing a Ph.D. in Electronics and Telecommunication from Mumbai University, focusing on VLSI and ML. She received M. E. (Electronics and Telecommunication) from Mumbai university in 2016 and B. E. (Electronics and Telecommunication) from Pune, in 2012. Her research interests include signal processing and VLSI.

Milind Nemade is currently working as Head of Artificial Intelligence and Data Science Department and Professor K. J. Somaiya Institute of Technology, Mumbai. He also worked as Associate Professor and Head of Electronics and Telecommunication Department and served in many administrative positions. In addition, serving actively in KJSIEIT activities, he has participated in many Short-Term Training Programs (STTP) organized by ISTE New Delhi in various colleges. He has presented 2 papers in National and 15 papers in international conferences and published 21 papers in international peer reviewed journals. His research interest includes speech processing and embedded system design. He is a member of IEEE, Life member of ISTE, Associate member of Institution of Engineers and Institute of Electronics and Telecommunication Engineers (IETE).