International Journal of
Advanced Computer Science Engineering and Artificial Intelligence
Peer Reviewed, Referred & Indexed Journal
ISSN: 3143-438X www.ijacseai.com Original Research Paper

HIGH-SPEED PSEUDO-RANDOM NUMBER GENERATOR FOR HOMOMORPHIC ENCRYPTION

Rehana Begum, Koppula Manasa
Department of Electronics & Communication Engineering(VLSI system designs), Vaagdevi Engineering College, Warangal, 506005, Telangana, India.
**Correspondence: Koppula Manasa (manasa436@gmail.com)*

ABSTRACT

A silicon-oriented pseudo-random number generator (PRNG) approach for cryptographic processing environments—including hardware platforms supporting homomorphic encryption—is presented in this work. The design consists of a ChaCha-family generation core set up for six rounds (ChaCha6) and a linear-feedback shift-register (LFSR)-driven counter. The ChaCha-based datapath executes nonlinear phrase operations to provide a pseudorandom output stream, while the LFSR-based block supplies the developing counter input. By lowering the round count compared to the twenty-round ChaCha structure, the design goal is to investigate a compact, high-efficiency hardware organization. The functional blocks, operation sequence, technical considerations, and evaluation methods are all described in the article. To show how the frequency, throughput, logic utilisation, register utilisation, power, and delay can be reported, an example performance table is shown. The numerical numbers in that table are not observations from a finished FPGA implementation; rather, they are fictitious samples. Similarly, output data must be processed through the NIST Statistically Test Suite to assess statistical quality; no pass result is provided here. Without specific cryptanalysis, it is not possible to presume that reduced-round ChaCha6 will offer the same level of security as the standardised 20-round ChaCha architecture.

Keywords: Hardware cryptography, FPGA, Verilog HDL, homomorphic encryption, ChaCha, ChaCha6, PRNG, LFSR, and pseudo-random number generator.

I.HISTORICAL

Random and pseudorandom values are important in cryptographic systems because they are used in key generation, nonce construction, masking, protocol initialization, and other security-related operations. Homomorphic encryption (HE) allows selected computations to be performed directly on encrypted data, but practical HE systems have substantial computational and hardware costs [1]–[3], [18]–[19]. The random values required by an HE scheme must satisfy the scheme's security and parameter requirements; therefore, a PRNG used in an HE-enabled hardware platform must be evaluated for both its cryptographic suitability and its implementation cost.

Hardware-oriented ChaCha research provides a useful background for this work. ChaCha is based on add-rotate-XOR (ARX) operations, which can be implemented using modular addition, bitwise XOR, and fixed rotations. Recent FPGA studies have demonstrated hardware implementations of ChaCha20 and ChaCha20–Poly1305, including implementations compatible with TLS 1.3 [13], [14]. These studies show that architectural choices such as datapath organization, resource sharing, and control structure affect hardware area and throughput. Other work has combined a ChaCha20 core with a physical unclonable function (PUF), illustrating how a hardware security primitive can be integrated with a cryptographic core [15]. Recent PRNG research has also explored FPGA implementations beyond conventional shift-register structures. Chaotic-map-based generators and hybrid architectures combining chaotic maps with shift registers have been evaluated for throughput, hardware cost, and statistical properties [16], [17].

Received: 17-10-2026 | Accepted: 21-11-2026 | Published: 27-11-2026

International Journal of
Advanced Computer Science Engineering and Artificial Intelligence
Peer Reviewed, Referred & Indexed Journal
ISSN: 3143-438X www.ijacseai.com Original Research Paper

These approaches are useful comparisons for hardware PRNG design, but their statistical results do not automatically establish cryptographic unpredictability. Similarly, an LFSR is suitable for generating a deterministic counter or state progression sequence, but its linearity means that it should not be treated as a stand-alone cryptographic PRNG [6].

The present architecture investigates a six-round ChaCha-family datapath (ChaCha6) together with an LFSR-based counter. The LFSR is used only to advance the counter-related input, while the nonlinear ARX datapath performs the mixing. ChaCha6 is a custom reduced-round configuration and must not be represented as equivalent in security to the standard 20-round ChaCha construction [4], [5], [20]. The design is therefore an architectural study; its security claims require independent cryptanalysis, and its output quality requires documented statistical testing. NIST SP 800-22 provides a statistical test suite for examining properties of random and pseudorandom bitstreams, but passing statistical tests alone does not prove cryptographic security [7].

II.CONNECTED WORK

Research on homomorphic encryption began with the concept of privacy homomorphisms introduced by Rivest, Adleman, and Dertouzos [1]. Gentry later demonstrated the first fully homomorphic encryption construction, establishing a foundation for computation over encrypted data [2]. Brakerski, Gentry, and Vaikuntanathan subsequently presented a leveled fully homomorphic encryption construction based on lattice assumptions [3]. More recent surveys describe how HE/FHE schemes and their hardware accelerators have evolved, and identify computational efficiency, polynomial arithmetic, modular multiplication, memory movement, and accelerator architecture as important implementation challenges [18], [19], [21]. Bian et al. review software–hardware acceleration approaches for FHE, while Zhang et al. provide a systematization of FHE accelerators and discuss design trade-offs [18], [19]. Su et al. (2026) further review hardware implementation approaches, including accelerator architectures and the design of polynomial and integer modular arithmetic units [21]. These studies provide the HE hardware context for a hardware-based random-data source; they do not, by themselves, validate the security of the PRNG proposed here.

For the stream-cipher datapath, Bernstein introduced ChaCha as a variant of Salsa20 using ARX transformations [4]. The IETF specification defines the ChaCha20 and Poly1305 constructions for use in network protocols [5]. Recent hardware work has examined ChaCha-based cryptographic cores in FPGA platforms. Serrano et al. reported a ChaCha20–Poly1305 hardware core compatible with TLS 1.3 [13], and later presented a detailed hardware implementation and evaluation of the AEAD construction [14]. Serrano et al. also combined ChaCha20 with a PUF implemented using latch metastability, demonstrating an approach to integrate a hardware security primitive with a cryptographic core [15]. Rashidi studied a high-performance ChaCha20 hardware structure using a sparse parallel-prefix adder and resource sharing to address delay and area [20]. These works use the standard ChaCha20 construction; their results should not be directly attributed to the six-round ChaCha6 architecture considered in this paper.

Research on hardware PRNGs has investigated FPGA-based chaotic generators and hybrid designs. Dridi et al. presented an FPGA implementation of a PRNG based on coupled chaotic maps and evaluated its hardware performance and statistical behaviour [16]. Kopparthi et al. proposed a digital piecewise-linear chaotic-map architecture combined with XOR shift register stages to improve the generated sequence and reported FPGA and ASIC evaluations with NIST testing [17]. Another FPGA study used elementary cellular automata with a chaotic system for PRNG generation and reported hardware and statistical evaluation [22]. These studies demonstrate alternative hardware approaches, but their randomness tests and

International Journal of
Advanced Computer Science Engineering and Artificial Intelligence
Peer Reviewed, Referred & Indexed Journal
ISSN: 3143-438X www.ijacseai.com Original Research Paper

application claims must be considered separately from cryptographic security analysis.

Linear-feedback shift registers are widely used in digital systems because they require relatively simple shift and XOR feedback logic. Golomb describes the sequence properties of shift registers [6]. However, the linear structure of a basic LFSR makes it unsuitable as an independent cryptographic PRNG. In the proposed architecture, the LFSR is therefore restricted to counter progression, while the ChaCha-family ARX datapath supplies nonlinear processing. The six-round configuration reduces the number of round operations compared with ChaCha20, but also changes the security margin. Consequently, this work should be treated as an architectural investigation until its exact state format, counter behaviour, output function, threat model, and cryptanalytic properties are fully specified and evaluated. Statistical testing based on NIST SP 800-22 can support the assessment of output properties, but cannot replace cryptanalysis [7].

III. SUGGESTED METHODS

An input and initialisation interface, key plus nonce memory, an LFSR-based conventional counter, a ChaCha6 processor core, a result register, and the control unit make up the proposed system. The necessary input words are loaded through the initialisation interface. During block generation, the input state is preserved in the key and nonce corresponding registers. The counter-related state is advanced by the LFSR-based counter in between output blocks. Through 6 rounds of ARX operations, the key, nonce,, counter, and additional state words are combined by the ChaCha6 core. Until the receiving module accepts it, the created block is kept in the output register.

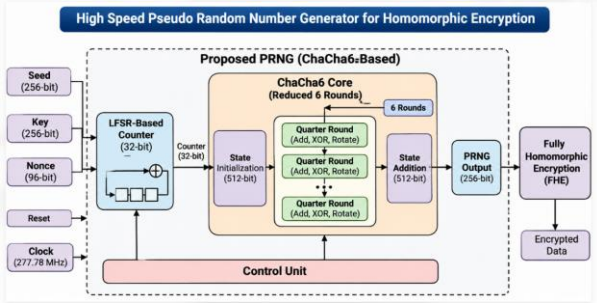

Fig 1: High Speed Pseudo Random Number Generator For Homomorphic Encryption

In accordance with the system's security criteria, the key & nonce have to be provided and handled. The PRNG information path must not repeat a state inside or reuse a nonce in a way that goes against the specifications of the chosen design. Before being implemented, the RTL specification must specify the precise state format, counter size, initialisation constants, output layout, and reseeding policy. There is feedback logic and a shift register in the LFSR-based counter. A feedback bit is computed from chosen taps and the register is shifted at each allowed update. The sequence behaviour and feedback polynomial are determined by the tap locations. Because an all-zero state in a traditional XOR-feedback LFSR can be locked at zero, reset creates a nonzero initial state. Instead of being released as the cryptography output stream, the LFSR output is utilised as a counter-associated input to the ChaCha6. state.

ARX operations and 32-bit words are used in the ChaCha6 core. Modular based addition, a XOR, and fixed revolutions are used in a quarter-round to update four chosen words. The six configured rounds are used for the quarter-round operations. The implementation must adhere to its specified feed-forward protocol and output-serialization rules after the round transformation. The RTL, a standard model, and testbench must all agree on these features. Instead of being referred to as standard ChaCha, every divergence from the normal ChaCha state change should be described as a custom build. Initialisation, state loading, round operation, output-valid creation, and counter advancement are all coordinated by the control unit. An operation for

International Journal of
Advanced Computer Science Engineering and Artificial Intelligence
Peer Reviewed, Referred & Indexed Journal
ISSN: 3143-438X www.ijacseai.com Original Research Paper

new generation is requested by a start input. A ready or busy indication can stop the customer from reading data that isn't complete. The output register records the outcome and declares output-valid once the necessary round operations are completed. The chosen block-generation procedure is then followed to advance the counter.

IV. DESIGN AND IMPLEMENTATION

Separate modules for the round controller, a state registers, top-level interface, LFSR counter, and ChaCha6 quarter-round datapath can be used to define the design in Verilog HDL. Readability is enhanced by modular RTL, which also enables independent block verification. The top-level module establishes connections between the generating core, output interface, state registers, control signals, and initialisation inputs. The design flow starts at the behavioural level with initialisation and reset. It loads the key, nonce,, and initial counter state. The controller moves the state at present into the computation registers when the generation request is approved. The set round operations are executed by the ChaCha6 datapath. The created output is recorded and labelled as valid after it is finished. Before the subsequent block is formed, the counter-related state progresses. Guided tests for reset, initialisation, start/ready handshake, counter progression, validation of output timing, and repeating block creation should all be part of verification. When comparing the RTL and standard output for comparable input conditions, a software model of reference should be utilised. Test vectors should include various key, nonce, & counter patterns as well as boundary values. To ensure that validation of output is not claimed prior to completion as well as that state updates only take place in authorised control states, assertions might be employed. The RTL spec can be synthesised and implemented utilising Xilinx Vivado following functional simulation. Timing analysis, placement, routing, synthesis, and RTL elaboration are all part of the implementation flow. The synthesis or deployment reports are where resource utilisation should be gathered. Instead of estimating the maximum frequency based on the clock period provided by the designer, the timing limitations and accomplished timing should be used. The Vivado based power report with the proper device, normal occurrence, and environment assumptions should be used to determine power. To test for randomness, a sufficiently big output bitstream must be exported from the hardware implementation or verified RTL and used with the NIST Random Test Package. The number of the input sequences of data, sequence length, significant level, p-values, pass percentages, and test setup should all be included in the report. Declaring that all NIST testing pass is invalid unless the assessments have been conducted and their outcomes documented.

V. DISCUSSION AND RESULTS

This section provides an example of how to report the evaluation of the proposed architecture. The numerical numbers below are fictitious samples made for presentation because no verified Vivado implementations report, target FPGA component number, metered power report, or NIST outcome file is given here. They cannot be presented as outcomes of measurements or experiments.

TABLE I: PERFORMANCE PARAMETERS FOR ILLUSTRATION

S. No.	Performance Parameter	Illustrative Value
1	Maximum operating frequency	250 MHz
2	Data throughput	2500 Mbps
3	Slice LUTs	1,250
4	Slice registers	820
5	Total on-chip power	0.85 W
6	Critical path delay	4.0 ns
7	ChaCha rounds	6

5.1 Throughput and Operating Frequency

The maximum operational frequency and throughput for this example are 250 MHz and 2500 Mbps, respectively. An average production rate of ten bits per cycle of the clock is suggested by these two numbers. The number of cycle per created block, output-valid behaviour, and the actual data path width must all be compared to that relationship. Instead of choosing the final throughput on its own,

Received: 17-10-2026 | Accepted: 21-11-2026 | Published: 27-11-2026

International Journal of
Advanced Computer Science Engineering and Artificial Intelligence
Peer Reviewed, Referred & Indexed Journal
ISSN: 3143-438X www.ijacseai.com Original Research Paper

it should be computed from the measured production rate.

5.2 Use of Hardware Resources

1200 Slice LUTs & 820 Slice Registers are listed in the sample table. Combinational operations including XOR, addition, rotational wiring, return logic, and control operations are implemented via LUTs. The state words, counters state, intermediary values, and output data are all stored in registers. The degree of data path sharing, phrase width, pipelining, the target FPGA, and how the ChaCha round are unrolled or repeated all affect how many resources are really used.

5.3 Use of Power

0.85 W is the approximate total on-chip power. Clock frequency, a switching activity, resource usage, device family, voltage, and setup all have an impact on power. The power evaluation report must yield a reliable result for the real implemented design. The report should specify the activity assumptions and indicate whether the value is measured or approximated.

5.4 Delay in the Critical Path

The critical route delay in the scenario is 4.0 ns. The target device type, placement, routing, and time constraints all affect the true delay, which is calculated as the longest restricted path following implementation. When available, the startpoint, endpoint, reasoning delay, routing postponement, and timing slack should all be reported in the final version of the document.

VI. FUTURE SCOPE AND CONCLUSION

This work presented a hardware-driven pseudo-random numbers generator approach that combines a six-round ChaCha-family computing core with an LFSR-based counter. The architecture divides output handling, control, nonlinear creation, and state progression into standardised functional blocks that Verilog HDL can describe and validate. Additionally, an evaluation technique based on FPGA timing and resource utilisation, power evaluation, and statistical testing was described in the article. A finished implementation is not represented by the numerical figures displayed in the Conclusions and Results section; they are merely illustrative. The RTL should be finished, validated against a reliable reference model, synthesised and implemented on a designated FPGA, timing and power data should be gathered, and generated output sequences should be subjected to the NIST test suite. While performing an impartial security analysis, future research should compare different round counts and technology organisations. Without sufficient cryptanalytic evidence, a reduced-round architecture shouldn't be used for security-critical applications.

REFERENCES

[1] R. L. Rivest, L. Adleman, and M. L. Dertouzos, “On data banks and privacy homomorphisms,” in Foundations of Secure Computation, 1978, pp. 169–180.

[2] C. Gentry, “Fully homomorphic encryption using ideal lattices,” in Proc. 41st ACM Symp. Theory of Computing (STOC), 2009, pp. 169–178, doi: 10.1145/1536414.1536440.

[3] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, “(Leveled) fully homomorphic encryption without bootstrapping,” ACM Trans. Comput. Theory, vol. 6, no. 3, Art. no. 13, 2014.

[4] D. J. Bernstein, “ChaCha, a variant of Salsa20,” in 举 .2008. [Online]. Available: https://cr.yp.to/chacha.html

[5] Y. Nir and A. Langley, “ChaCha20 and Poly1305 for IETF protocols,” RFC 8439, Internet Engineering Task Force, Jun. 2018, doi: 10.17487/RFC8439.

[6] S. W. Golomb, Shift Register Sequences. San Francisco, CA, USA: Holden-Day, 1967.

[7] A. Rukhin et al., “A statistical test suite for randomand pseudorandom number generators for cryptographic applications,” NIST Special Publication 800-22 Rev. 1a, Apr. 2010, doi: 10.6028/NIST.SP.800-22r1a.

[8] J. Katz and Y. Lindell, Introduction to Modern Cryptography, 3rd ed. Boca Raton, FL, USA: CRC Press, 2020.

[9] D. R. Stinson and M. B. Paterson, Cryptography: Theory and Practice, 4th ed. Boca Raton, FL, USA: CRC Press, 2018.
[10] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, Handbook of Applied Cryptography. Boca Raton, FL, USA: CRC Press, 1996.
[11] P. L. Montgomery, "Modular multiplication without trial division," Math. Comput., vol. 44, no. 170, pp. 519–521, 1985.
[12] C. Peikert, "A decade of lattice cryptography," Found. Trends Theor. Comput. Sci., vol. 10, nos. 4, pp. 283–424, 2016.
[13] R. Serrano, C. Duran, T.-T. Hoang, M. Sarmiento, A. Tsukamoto, K. Suzaki, and C.-K. Pham, "ChaCha20-Poly1305 crypto core compatible with Transport Layer Security 1.3," in Proc. 18th Int. SoC Design Conf. (ISOCC), Jeju Island, South Korea, 2021, pp. 17–18, doi: 10.1109/ISOCC53507.2021.9614016.
[14] R. Serrano, C. Duran, M. Sarmiento, C.-K. Pham, and T.-T. Hoang, "ChaCha20–Poly1305 authenticated encryption with additional data for Transport Layer Security 1.3," Cryptography, vol. 6, no. 2, Art. no. 30, 2022, doi: 10.3390/cryptography6020030.
[15] R. Serrano, C. Duran, M. Sarmiento, T.-K. Dang, T.-T. Hoang, and C.-K. Pham, "A unified PUF and crypto core exploiting the metastability in latches," Future Internet, vol. 14, no. 10, Art. no. 298, 2022, doi: 10.3390/fi14100298.
[16] F. Dridi, S. El Assad, W. El Hadj Youssef, M. Machhout, and A. E. Samhat, "Design, FPGA-based implementation and performance of a pseudo random number generator of chaotic sequences," Adv. Electr. Comput. Eng., vol. 21, no. 2, pp. 41–48, 2021, doi: 10.4316/AECE.2021.02005.
[17] V. R. Kopparthi, A. Kali, S. L. Sabat, K. K. Anumandla, R. Peesapati, and J. S. A. Eyebe Fouda, "Hardware architecture of a digital piecewise linear chaotic map with perturbation for pseudorandom number generation," AEU—Int. J. Electron. Commun., vol. 147, Art. no. 154138, 2022, doi: 10.1016/j.aeue.2022.154138.
[18] S. Bian, R. Mao, Y. Zhu, Y. Fu, Z. Zhang, L. Ding, J. Zhang, B. Zhang, Y. Chen, J. Dong, and Z. Guan, "A survey on software-hardware acceleration for fully homomorphic encryption," J. Electron. Inf. Technol., vol. 46, no. 5, pp. 1790–1805, 2024, doi: 10.11999/JEIT230448.
[19] J. Zhang, X. Cheng, L. Yang, J. Hu, X. Liu, and K. Chen, "SoK: Fully homomorphic encryption accelerators," ACM Comput. Surv., vol. 56, no. 12, Art. no. 316, 2024, doi: 10.1145/3676955.
[20] B. Rashidi, "High-performance hardware structure of ChaCha20 stream cipher based on sparse parallel prefix adder," Int. J. Circuit Theory Appl., vol. 53, no. 5, pp. 2947–2957, 2025, doi: 10.1002/cta.4264.
[21] Y. Su, K. Zhou, W. Zhong, J. Wang, J. Hou, and C. Yang, "A brief survey on hardware implementation of fully homomorphic encryption," Cryptography, vol. 10, no. 4, Art. no. 49, 2026, doi: 10.3390/cryptography10040049.
[22] N. Abbassi, A. Mtibaa, M. Gafsi, and M. A. Hajjaji, "An enhanced ECA/chaotic-based PRNG: Hardware design and implementation," in Proc. 2022 IEEE 21st Int. Conf. on Sciences and Techniques of Automatic Control and Computer Engineering (STA), Sousse, Tunisia, 2022, pp. 249–254, doi: 10.1109/STA56120.2022.10019236.

Received: 17-10-2026 | Accepted: 21-11-2026 | Published: 27-11-2026