

An efficient Grain-80 stream cipher with unrolling features to enhance the throughput on hardware platform

Raghavendra Ananth¹, Panduranga Rao Malode Vishwanatha Rao², Narayana Swamy Ramaiah³

¹Department of Electronics and Communications, JAIN Deemed to be University, Bangalore, India

²Department of Computer Science Engineering, FET-JAIN Deemed to be University, Bangalore, India

³Department of Computer Engineering, Sankalchand Patel University Visnagar, Gujarat, India

Article Info

Article history:

Received Aug 1, 2023

Revised Oct 14, 2023

Accepted Oct 25, 2023

Keywords:

Grain-80

Keystream

Shift register

Stream cipher

Throughput

ABSTRACT

The stream cipher is a fundamental component of symmetric cryptography and offers unique implementation speed and scalability advantages. Additionally, the complexity of the cipher algorithm deployment environment forces new, appropriate designs and challenges on the already-existing cipher algorithms. To increase throughput, an efficient Grain-80 stream cipher with unrolling features is designed in this manuscript. The Grain-80 cipher uses an 80-bit key, and a 64-bit initialization vector (IV) and contains two feedback shift registers (linear and non-linear) and an output function. The register balancing and unrolling features of the proposed Grain-80 cipher combine to increase throughput while requiring little additional hardware. Low latency, fast throughput, excellent efficiency, and reduced attack susceptibility are all features of the unrolling architecture. The proposed Grain-80 cipher utilizes <1% chip area and operates at 542.7 MHz on Artix-7 field programmable gate array (FPGA). The proposed Grain-80 cipher improves the operating frequency by 14.85% over conventional Grain-80 cipher. The Grain-80 cipher obtains the throughput of 4.35 Gbps and 8.69 Gbps for unrolling factors 8 and 16, respectively. Lastly, the proposed Grain-80 cipher is compared with existing Grain-80 ciphers with improved throughput and hardware efficiency.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Raghavendra Ananth

Department of Electronics and Communications, JAIN Deemed to be University

Bangalore, India

Email: araghavendra.research@gmail.com

1. INTRODUCTION

The widespread use of online shopping, digital government, virtualization, and big data, collectively with their rapid development, has raised high expectations for the speed and privacy of data handling. Cryptography, which is at the foundation of data security, now faces fresh opportunities and difficulties. Lower implementation costs become required, in particular, to accommodate limited resources, such as mobile broadband networks and radio frequency identification devices (RFID) labels. More robust speed and more substantial security levels are also essential. Stream ciphers distinguish among all types of primitives used in cryptography and are crucial for maintaining data confidentiality [1]. These days, stream ciphers are characterized by simple software and hardware deployment, quick encryption and decryption, and little or no error transmission. As a result, they continue to dominate applications for unique and private organizations, such as diplomatic collaboration, government operations, and defense-related matters. Stream cipher strategies are now a standard part of many worldwide norms and public network protocols for securing sensitive data [2].

Lightweight cryptography (LWC) focuses on cryptographic techniques appropriate for these systems. The primary functions that lightweight block and stream ciphers primarily provide in embedded gadgets are

privacy and data reliability. Stream ciphers have become more popular recently due to numerous research projects looking at secure stream cipher designs. The *eStream* rivalry, sponsored by the European network of excellence for cryptology (ECRYPT) from 2004 to 2008 and encouraged the construction of practical and compact stream ciphers appropriate for widespread deployment, was a turning point in relevant research activities [3], [4]. Physical unclonable functions (PUFs) are frequently presented as a key component of security designs and cryptographic protocols. The secret unknown cipher (SUC) term refers to the digital clone-resistance features. Designing families of secure ciphers with random elements is necessary for SUC. To overcome the limitations of PUFs, a new big class of low-complexity stream ciphers with the same planned level of protection is built utilizing stream cipher-based SUC [4], [5]. To overcome cloud service limitations, an enhanced stream cipher based on rivest cipher-4 (RC-4) with a 128-bit key has been constructed [6].

The *eStream* project was developed by the ECRYPT in 2004. Eight of the 34 stream cipher submissions made it to the final competition. The *eStream* inventory presently contains 7 ciphers because one of the finalists had been effectively cryptanalyzed. According to the *eStream* inventory for 2012, the ciphers Salsa20/12, Rabbit, and sosemanuk, are included in the software background, and Trivium, Mickey 2.0, and Grain-v1 are featured on the hardware side [7]. Grain was developed for hardware implementations that need fewer gates and less power. It includes two 80-bit feedback shift registers (FSRs), one linear and the other non-linear. The cipher was improved and named Grain-v1 after various planned attacks upon its initial form. Grain-128 was created due to additional improvements, such as a rise in the size of the FSRs. Grain-128 later evolved into a new focus for cryptanalysts [8].

An efficient high throughput Grain-80-based stream cipher with unrolling features is designed in this manuscript. The contribution of the proposed Grain-80 cipher is highlighted as follows: the register balancing optimization approach is used in the proposed grain-80 cipher to overcome the drawbacks of the conventional Grain-80 cipher, including frequency and throughput. The unrolling features are used as one of the optimization approaches instead of the pipelining approach in the proposed Grain-80 cipher to enhance the throughput with few additional resources. The simulation results are verified and validated with theoretical values of the original Grain v1 cipher. The performance parameters like frequency, throughput and hardware efficiency of the proposed cipher are improved over existing Grain-80 cipher approaches.

The existing works of grain based stream cipher and their performance realization on different platforms are discussed in this section. Watanabe *et al.* [9] present the Grain version-1 (v1) based stream cipher and its differential cryptoanalysis. The NFSR-based cipher is used to construct the Grain v1 cipher. The work analyses the inverse key initialization process and its impacts. The conditional differential cryptoanalysis, key recovery attack, and weak-key analysis of Grain v1 are discussed in detail. Kazmi *et al.* [10] discuss the Grain and Trivium-based algebraic side-channel attacks (ASCAs). The work integrates the side-channel attacks (SCAs) and algebraic attacks to form the ASCAs. The ASCAs are solved using partial SCA and algebraic cryptoanalysis on Grain/Trivium ciphers. The ASCAs are solved within 28.25 sec for 80-bit stream cipher output without the initialization phase. Hell *et al.* [11] describe the new Grain cipher called Grain128AEAD. The Grain-128 AEAD cipher is an authenticated encryption with associated data (AEAD). The Grain-128AEAD supports a 128-bit key with a 96-bit IV and random sequence to perform the encryption and authentication process. The Grain-128AEAD analyses the correlation attacks, chosen IV attacks, and fault attacks. The cipher operates at 662 MHz and obtains a throughput of 10.59 Gbps on the 32nd parallelization process. Ngo *et al.* [12] explain the bit stream-level-based trivium attacks. The contents of the three lookup tables (LUTs) are altered at the bit-stream level to linearize trivium's non-linear state updating function. The key bits are recovered from 288 keystreams using $2^{19.41}$ operations. Chatterjee *et al.* [13] describe end-to-end encryption using a cryptographic model on FPGA based system. The work uses symmetric methods like triangular encryption (TE), recursive pared parity technique (RPPT), and transformation of bits (TB) techniques for the encryption process. The work realizes the hardware resource utilization and timing analysis with existing approaches with improvements. Ding *et al.* [14] present the new lightweight stream ciphers using an FPGA environment based on chaos. The two non-linear feedback shift registers (NFSRs) and the chaotic system create the new stream cipher model. The work analyses the 16-National Institute of Standards and Technology (NIST) statistical tests to find the p-value. The work also compares the proposed design with existing work with area and throughput improvements. Pan *et al.* [15] explain the Grain based near collision attack (NCA) with fast optimization process using linear programming. The timing complexity is reduced to 27% by merging the internal states-a 79.4% reduction of theoretical timing complexity than conventional Grain v1 cipher.

Perez-Resca *et al.* [16] discuss the format preserving encryption (FPE) for high-data-rate communications. The FPE, FF1 (format (format-preserving feistel), and FF3 algorithms are designed and analysed in detail. The FPE algorithm with counter mode is designed to improve different attacks' confidentiality. The proposed design work is compared with existing work with area and throughput improvements. Tsavos *et al.* [17] describe the reconfigurable logic-based lightweight data security on the FPGA platform. The work analyses The Sober-128, Snow 2.0 and turing ciphers with reconfigurable logic

approaches. The adder and XOR blocks are customized in the above three ciphers to construct a lightweight data security system. The Sober-128, Snow 2.0, and turing ciphers operate at a throughput of 408 Mbps, 2040 Mbps and 788 Mbps, respectively. Dridi *et al.* [18] explain the stream cipher-based secure chaotic generator on the FPGA platform. The design uses chaotic numbers, recursive filters, and mixing techniques for secure pseudo-chaotic number generation. The work analyzes the area resources, throughput, and statistic tests with better efficiency and security improvements. Salam *et al.* [19] present the Grain-128AEAD cipher and its differential fault attacks. The random faults are analyzed using the probabilistic method and recover the initial state with $2^{10.45}$ fault injection. Huang *et al.* [20] discuss the memristive chaotic system with Grain-128a cipher for image encryption and pixel masking. The image encryption methods perform key protection using Grain-128a, pixel masking with dynamic features, and Brownian motion operation. The security analysis of the image encryption system is discussed in detail, including histogram, information entropy, key space, correlation, and robustness analysis.

The manuscript's organization is as follows: the hardware architectures of the conventional and proposed Grain-80 stream cipher are described in detail in section 2. The results of the proposed work are realized in detail with tabulation and performance comparison in section 3. Lastly, it concludes the overall work with improvement in section 4 with a futuristic scope.

2. GRAIN-80 CIPHER

The Grain-80 is one of the synchronous stream ciphers with bit-oriented features. The Keystream output is obtained independently from input data. The cipher uses two shift registers: linear feedback shift register (LFSR) and non-linear feedback shift register (NFSR). The LFSR ensures output balance and an initial time frame assurance for the Keystream. The NFSR adds non-linearity to the Grain-80 cipher and a non-linear output function. The LFSR's output masks the NFSR's input to balance the NFSR's state. The Grain-80 cipher makes it impossible to perform an attack faster than an exhaustive key search; consequently, an optimal attack should necessitate computational complexity of at least 2^{80} .

2.1. Conventional Grain-80 cipher

The Grain-80 cipher is a selected hardware-based *eStream* project presented by Hell *et al.* [21]. The Grain-128 cipher is the extension of Grain-80 version-1 with additional security features presented by Hell *et al.* [22]. The Grain offers better security than other familiar ciphers for hardware applications. The conventional Grain-80 architecture is illustrated in Figure 1. The Grain-80 cipher architecture mainly contains output function $h(x)$, NFSR, and LFSR blocks. The Grain-80 cipher supports an 80-bit key size (k) and 64-bit initialization vector (IV).

The 80-bit LFSR contains $L_i = l_i, l_{i+1}, \dots, l_{i+79}$ and 80-bit NFSR has $N_i = n_i, n_{i+1}, \dots, n_{i+79}$. The LFSR primitive or feedback polynomial $f(x)$ and NFSR feedback polynomial $g(x)$ are represented using (1) and (2). The LFSR's and NFSR's update functions are additionally defined in (3) and (4) to clear up any confusion:

$$f(x) = x^{80} + x^{67} + x^{57} + x^{42} + x^{29} + x^{18} + 1 \quad (1)$$

$$\begin{aligned} g(x) = & x^{59}x^{52}x^{47}x^{43}x^{35}x^{28} + x^{71}x^{65}x^{59}x^{52}x^{47} + x^{43}x^{35}x^{28}x^{20}x^{17} \\ & + x^{65}x^{59}x^{20}x^{17} + x^{47}x^{43}x^{28}x^{20} + x^{71}x^{52}x^{35}x^{17} + x^{59}x^{52}x^{47} \\ & + x^{35}x^{28}x^{20} + x^{71}x^{65} + x^{47}x^{43} + x^{20}x^{17} + x^{80} + x^{71} + x^{66} \\ & + x^{59} + x^{52} + x^{47} + x^{43} + x^{35} + x^{28} + x^{20} + x^{18} + 1 \end{aligned} \quad (2)$$

$$l_{i+80} = l_i + l_{i+13} + l_{i+23} + l_{i+38} + l_{i+51} + l_{i+62} \quad (3)$$

$$\begin{aligned} n_{i+80} = & n_{i+21}n_{i+28}n_{i+33}n_{i+37}n_{i+45}n_{i+52} + n_{i+9}n_{i+15}n_{i+21}n_{i+28}n_{i+33} \\ & + n_{i+37}n_{i+45}n_{i+52}n_{i+60}n_{i+63} + n_{i+15}n_{i+21}n_{i+60}n_{i+63} \\ & + n_{i+33}n_{i+37}n_{i+52}n_{i+60} + n_{i+9}n_{i+28}n_{i+45}n_{i+63} + n_{i+21}n_{i+28}n_{i+33} \\ & + n_{i+45}n_{i+52}n_{i+60} + n_{i+9}n_{i+15} + n_{i+33}n_{i+37} + n_{i+60}n_{i+63} \\ & + n_i + n_{i+9} + n_{i+14} + n_{i+21} + n_{i+28} + n_{i+33} + n_{i+37} + n_{i+45} \\ & + n_{i+52} + n_{i+60} + n_{i+62} + l_i \end{aligned} \quad (4)$$

Two shift registers represent the Grain-80 cipher's state. The output function $h(x)$ receives the five variables as input from the state. The output function $h(x)$ provides the non-linearity and balanced features. The output function $h(x)$ using LFSR and NFSR is represented using (5) and The output function $H(L_i, N_i)$ is represented in (6) and as follows:

$$h(x) = x_2x_3x_4 + x_1x_2x_4 + x_0x_2x_3 + x_0x_2x_4 + x_1x_2 + x_3x_4 + x_2x_3 + x_0x_2 + x_4 + x_0 \quad (5)$$

$$kso_i = H(L_i, N_i) = \sum_{m \in B} l_{i+m} + h(n_{i+63}, l_{i+64}, l_{i+46}, l_{i+25}, l_{i+3}) \quad (6)$$

where $x_4, x_3, x_2, x_1,$ and x_0 are concerned with tap position of $l_{i+63}, n_{i+64}, n_{i+46}, n_{i+25},$ and n_{i+3} respectively. $B=\{1, 2, 4, 10, 31, 43, 56\}$. The key initialization process starts using key and IV inputs before generating key stream outputs. The key and IV bits are defined as $k_i (0 \leq i \leq 79)$ and $IV_i (0 \leq i \leq 63)$. The key initialization process is organized as follows: the NFSR data is loaded with key bits: $n_i=k_i; (0 \leq i \leq 79)$ followed by LFSR data is loaded with IV bits: $l_i=IV_i; (0 \leq i \leq 63)$. The rest of the LFSR bits (from 64th to 79th) are loaded with ones. The key initialization process takes 160 clock cycles (CC) to load the LFSR (80 CC) and NFSR (80 CC) data and won't produce any Keystream output.

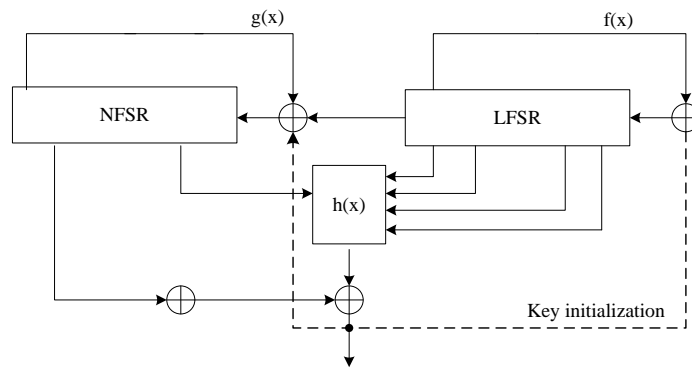


Figure 1. Conventional Grain-80 cipher architecture

2.2. Proposed Grain-80 cipher

The proposed Grain-80 cipher architecture is illustrated in Figure 2. The register balancing approach improves the throughput concerning the LFSR and NFSR in the Grain-80 cipher. The architecture contains 80-bit LFSR, NFSR, five register sets and a control logic unit. The five register sets R_0, R_1, R_2, R_3 and R_4 are considered as a part of the register balancing approach to decompose the shift register data. The LFSR feedback polynomial $f(x)$ is generated using the register set (R_0). The NFSR feedback polynomial $g(x)$ is generated based on the register set (R_1 and R_2). The output function $h(x)$ is generated using the register set (R_3 and R_4). The register sets ($R_0, R_1, R_2, R_3,$ and R_4) are defined using (7) to (11). The $f(x), g(x),$ and $h(x)$ are generated based on the register sets and defined in (12).

$$R_0 = l_1 + l_{14} + l_{24} + l_{39} + l_{52} + l_{63} \quad (7)$$

$$R_1 = l_1 + n_1 + n_{10} + n_{15} + n_{22} + n_{29} + n_{34} + n_{38} + n_{61}n_{64} + n_{34}n_{38} + n_{10}n_{16} + n_{46}n_{53}n_{61} \quad (8)$$

$$R_2 = n_{22}n_{29}n_{34} + n_{10}n_{29}n_{46}n_{64} + n_{34}n_{38}n_{53}n_{61} + n_{46}n_{53}n_{61} + n_{16}n_{22}n_{61}n_{64} + n_{38}n_{46}n_{53}n_{61}n_{64} + n_{10}n_{16}n_{22}n_{29}n_{34} + n_{22}n_{29}n_{34}n_{38}n_{46}n_{53} + n_{46} + n_{53} + n_{61} + n_{63} \quad (9)$$

$$R_3 = n_2 + n_3 + n_{11} + n_{32} + n_{44} + n_{57} + n_{26} + n_{64} \quad (10)$$

$$R_4 = l_4l_{65} + l_{47}l_{65} + l_{65}n_{64} + l_4l_{26}l_{47} + l_4l_{47}l_{65} + l_4l_{47}n_{64} + l_{26}l_{47}l_{64} + l_{47}l_{65}n_{64} \quad (11)$$

$$f(x) = R_0, \quad g(x) = R_1 + R_2, \quad \text{and} \quad h(x) = R_3 + R_4 \quad (12)$$

The control logic unit performs the key initialization using a counter mechanism and shift register updation process. The key initialization process is completed till the valid signal is active. The shift register updation continues until the successive 80-clock cycles produce the updated $h(x)$ output function. After the valid signal goes high, the updated $h(x)$ output is considered the final key stream output (kso). The MSB-bit of the NFSR and LFSR are updated using their polynomials and output function $h(x)$ using (13) as follows:

$$n_{79} = g(x) + h(x) \text{ and } l_{79} = f(x) + h(x) \tag{13}$$

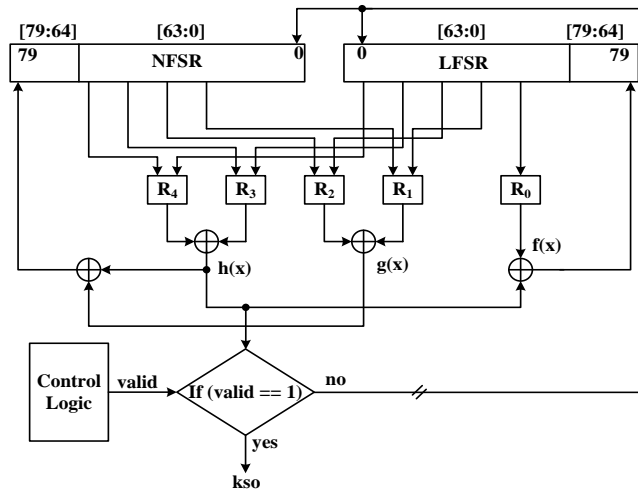


Figure 2. Proposed Grain-80 cipher architecture

2.2.1. Grain-80 cipher with unrolling features

The LFSR and NFSR is updated every clock cycle, so cipher output is estimated at 1-bit/clock. The speed or throughput of the Grain-80 cipher is increased with additional resources (chip area). The throughput-increasing process or unrolling process is achieved by performing the output function $h(x)$ and feedback polynomials $f(x)$ and $g(x)$ multiple times. The unrolling features are activated by considering only the first 64-bit LFSR and NFSR. The last 16-bit shift registers are not considered in this process, which speeds up the process-based scaling factor (up to 16). The proposed Grain-80 cipher with unrolling 2-times ($2x$) architecture is illustrated in Figure 3 as an example.

In the architecture, only register sets, feedback polynomials, and output functions are repeated. The process is repeated as the unrolling factor (u) increases in Grain-80 cipher. The key initialization process is reduced by dividing the 160 CC with an unrolling factor (u). Hence, this approach is entirely appropriate to enhance the throughput.

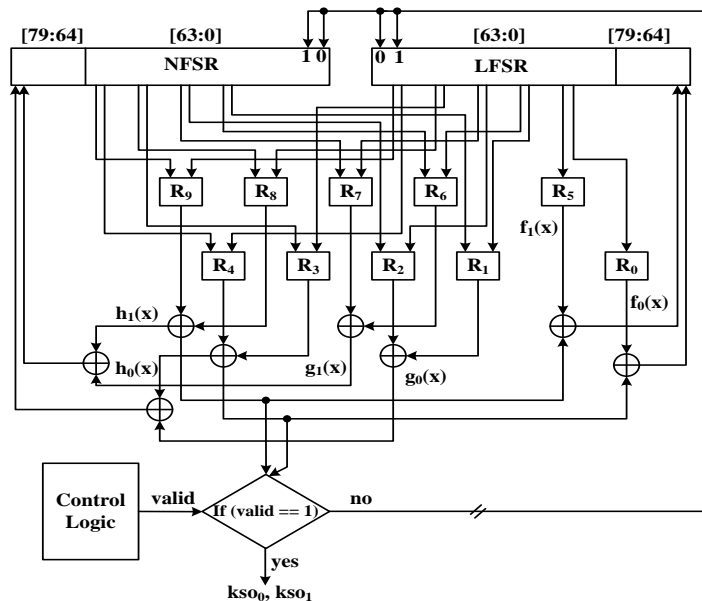


Figure 3. Proposed Grain-80 cipher with unrolling 2-times ($2x$) architecture

3. RESULTS AND DISCUSSION

The results of the proposed Grain-80 stream cipher with unrolling features are discussed in this section. The Grain-80 cipher is designed and synthesized using Verilog-HDL on Artix-7 FPGA (XC7A100T-3CSG324) on the Xilinx ISE environment. The ModelSim simulator is used to generate simulation results. The simulation, synthesis, and performance metrics of the Grain-80 cipher with unrolling features are realized in detail. The simulation results of the proposed Grain-80 cipher are illustrated in Figure 4. The global clock (clk) is activated with low asynchronous reset (rst). Initialize the load input (init) from high to low and define the 80-bit key (key) and 64-bit initialization vector (iv) inputs. The MSB bit of the 80-bit key input and 64-bit iv input is shifted to 1-bit key_i and iv_i on every clock cycle. The corresponding 1-bit key stream output (ks_o) is generated and valid only after the key stream valid (ks_valid) output is high. Once the ks_valid is high, the counter (ks_counter) counts till 80 clock cycles to generate the done signal high and obtain the correct 80-bit key stream output (ks_output) based on the control logic operation of Grain-80 cipher.



Figure 4. Simulation results of proposed Grain-80 cipher

The conventional/proposed Grain-80 cipher and corresponding h(x) function resource utilization are tabulated in Table 1. The h(x) function of conventional Grain-80 cipher utilizes slices of 74, look up table (LUTs) of 48, and operates at 462.12 MHz the h(x) function of the proposed Grain -80 cipher utilizes slices of 80, LUTs of 54, and operates at 609.905 MHz the conventional/proposed Grain-80 cipher utilizes <1% chip area (slices and LUTs) and operates at 462.12 MHz/542.741 MHz, consuming 87 mW of total power on Artix-7 FPGA. The proposed h(x) function and Grain-80 cipher improve the operating frequency by 24.24% and 14.85% than the conventional h(x) function and Grain-80 cipher approach, respectively.

Table 1. Resource results of h (x) function and Grain-80 cipher on Artix-7 FPGA

Resources	Conventional	Proposed
h (x) function of Grain-80 ciphers		
Slices	74	80
LUTs	48	54
Max. frequency (MHz)	462.12	609.905
Grain-80 cipher		
Slices	93	99
LUTs	72	72
Max. frequency (MHz)	462.12	542.741
Power (mW)	87	87

The performance summary of the Grain-80 cipher with unrolling factor on Artix-7 FPGA is tabulated in Table 2. The unrolling factors from 1, 2, 4, 8, and 16 are considered in this work. The Grain-80 supports up to an unrolling factor of 16 due to the size of the NFSR and LFSR. The Grain-80 cipher obtains the same operating of 542.741 MHz for all the unrolling factors (1 to 16). The Grain-80 cipher utilizes the chip area (Slices and LUTs) of <1% for unrolling factors from 1 to 8 and 1% for unrolling factor 16. The Xpower power analyzer is used to estimate the total power and consumes from 87 mW to 130 mW for unrolling factors 1 to 16. The throughput is calculated based on key size (1-bit), operating frequency, and latency. So, throughput=(unrolling factor×key size×frequency)/latency. The throughput of 0.543 Gbps and 8.69 Gbps is obtained for unrolling factors 1 and 16, respectively. The hardware efficiency is measured using throughput/slice and throughput/LUT. The efficiency of 5.49 Mbps/Slice and 7.75 Mbps/Slice is obtained for unrolling factors 1 and 16, respectively. The resource utilization of Grain-80 with an unrolling factor on Artix-7 FPGA is illustrated in Figure 5 the performance metrics like chip area, power, throughput, and efficiency

increase as the unrolling factor increases. The throughput is doubled (2x) for every unrolling factor rise in the Grain-80 cipher.

Table 2. Performance summary of the Grain-80 cipher with unrolling factors on Artix-7 FPGA

Resources	Unrolling factor				
	1	2	4	8	16
Max. frequency (MHz)	542.741	542.741	542.741	542.741	542.741
Total power (mW)	87	91	98	108	130
Slices	99	166	300	576	1,121
LUTs	72	125	231	443	849
Throughput (Gbps)	0.543	1.086	2.17	4.35	8.69
Efficiency (Mbps/Slice)	5.49	6.54	7.24	7.54	7.75
Efficiency (Mbps/LUT)	7.54	8.69	9.4	9.81	10.22

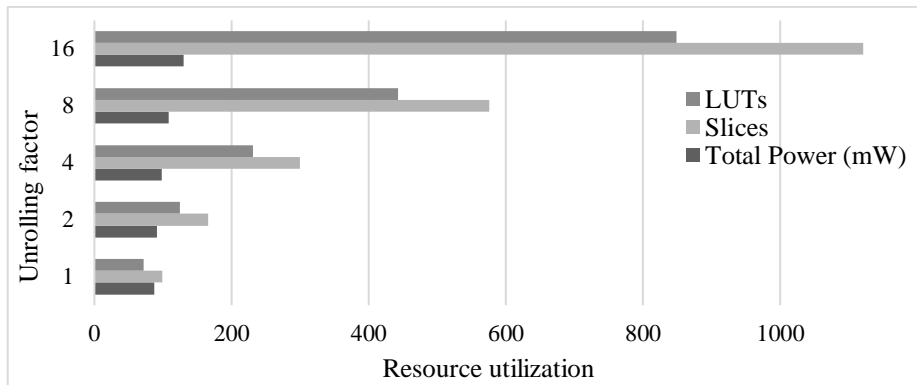


Figure 5. Resource utilization of Grain-80 with unrolling factors on Artix-7 FPGA

The performance realization (unrolling factor v/s throughput) of Grain-80 on different FPGAs like Spartan-6, Artix-7, and Virtex-7 is illustrated in Figure 6. The Spartan-6 FPGA is low-cost, has limited resources, and is used for academic purposes. The Artix-7 FPGA is affordable, has enough resources, and is used for particular applications. The Virtex-7 FPGA is a high-end FPGA with many resources used for advanced complex applications. The proposed Grain-80 cipher obtains the throughput of 367 Mbps, 543 Mbps, and 769 Mbps on Spartan-6, Artix-7, and Virtex-7 FPGAs for unrolling factor 2. Similarly, the Grain-80 cipher obtains the throughput of 5.87 Gbps, 8.69 Gbps, and 12.31 Gbps on Spartan-6, Artix-7, and Virtex-7 FPGAs, respectively, for unrolling factor 16. The operating frequency of Grain-80 cipher is varied from one FPGA to another FPGA. The throughput varies exponentially as the unrolling factor increases, even in FPGAs from low to high.

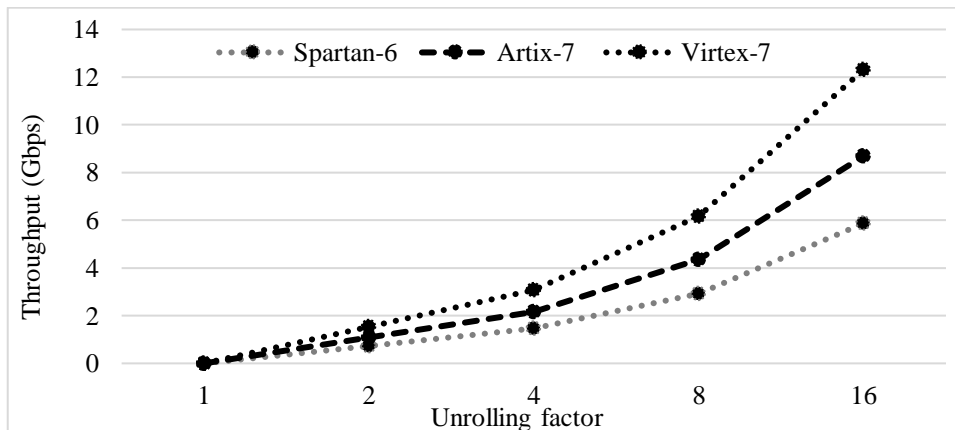


Figure 6. Performance realization of Grain-80 on various FPGAs

The performance comparison of the proposed Grain-80 ciphers with existing Grain-80 ciphers [23]–[25] is in Table 3. The chip area (Slices), operating frequency (Fmax) in MHz, latency (clock cycles), throughput (Mbps), and efficiency (Mbps/Slice) parameters are considered for performance comparison on different FPGAs (Spartan-6/Virtex-7). The proposed Grain-80 cipher reduces the chip area by 68.8% and improves the throughput by 51.78% than the existing Grain-80 cipher [23] on Spartan-6 FPGA. Similarly, the proposed Grain-80 improves the throughput by 9.27% than the existing Grain-80 cipher [24] on Spartan-6 FPGA. The proposed Grain-80 cipher reduces the chip area by 25.56% and improves the throughput of the 9.89% and 32.56 efficiency than the existing Grain-80 cipher [25] on Virtex-7 FPGA. Overall, the proposed Grain-80 cipher improves the performance metrics than existing Grain-80 ciphers on different FPGAs.

Table 3. Performance comparison of proposed Grain-80 cipher with existing Grain-80 ciphers [23]–[25]

Resources	Kitsos <i>et al.</i> [23]	Li <i>et al.</i> [24]	Alharbi <i>et al.</i> [25]	This work
FPGA	Spartan-6	Spartan-6	Virtex-7	Spartan-6/Virtex-7
Area (Slices)	318	62	133	99/99
Fmax (MHz)	177	333	693	367/769
Latency (CC)	1	1	161	1/1
Throughput (Mbps)	177	333	693	367/769
Efficiency (Mbps/Slice)	0.556	5.37	5.24	3.71/7.77

4. CONCLUSION AND FUTURE WORK

The Grain-80 stream cipher with unrolling features is designed and implemented on the hardware platform in this manuscript. The architectures of the conventional and proposed Grain-80 stream ciphers are discussed. The Grain-80 cipher is constructed using two shift registers (linear and non-linear) and an output function. The is used to drawbacks of the conventional Grain-80 cipher are overcome and improved in the proposed Grain-80 cipher using the register balancing approach. The unrolling feature is adopted as an optimization approach in Grain-80 cipher to enhance the system throughput with few additional resources on the chip. The simulation results of the Grain-80 cipher are verified and validated with theoretical values. The proposed Grain-80 cipher uses <1% chip area (slices and LUTs) and operates at 542.7 MHz by consuming a power of 87 mW on Artix-7 FPGA. The Grain-80, with an unrolling factor of 16, operates at 8.69 Gbps with an efficiency of 7.75 Mbps/Slice. The proposed Grain-80 cipher is compared with other Grain-80 cipher with improved frequency, throughput and efficiency parameters. In the future, security analysis concerning the attacks of the proposed Grain-80 cipher will be evaluated.




REFERENCES

- [1] M. A. Philip and Vaithiyathan, "A survey on lightweight ciphers for IoT devices," in *2017 International Conference on Technological Advancements in Power and Energy (TAP Energy)*, Dec. 2017, pp. 1–4, doi: 10.1109/TAPENERGY.2017.8397271.
- [2] L. Jiao, Y. Hao, and D. Feng, "Stream cipher designs: a review," *Science China Information Sciences*, vol. 63, no. 3, p. 131101, Mar. 2020, doi: 10.1007/s11432-018-9929-x.
- [3] C. Maniavas, G. Hatzivasilis, K. Fysarakis, and Y. Papaefstathiou, "A survey of lightweight stream ciphers for embedded systems," *Security and Communication Networks*, vol. 9, no. 10, pp. 1226–1246, Jul. 2016, doi: 10.1002/sec.1399.
- [4] A. Babaei and G. Schiele, "Physical unclonable functions in the internet of things: state of the art and open challenges," *Sensors*, vol. 19, no. 14, p. 3208, Jul. 2019, doi: 10.3390/s19143208.
- [5] A. Mars and W. Adi, "New family of stream ciphers as physically clone-resistant VLSI-structures," *Cryptography*, vol. 3, no. 2, p. 11, Apr. 2019, doi: 10.3390/cryptography3020011.
- [6] S. Gnatyuk, M. Iavich, V. Kinzeryavyy, T. Okhrimenko, Y. Burmak, and I. Goncharenko, "Improved secure stream cipher for cloud computing," in *CEUR Workshop Proceedings*, 2020, vol. 2732, pp. 183–197.
- [7] P. R. Hridya and J. Jose, "Cryptanalysis of the Grain family of ciphers: a review," in *2019 International Conference on Communication and Signal Processing (ICCSP)*, Apr. 2019, pp. 0892–0897, doi: 10.1109/ICCSP.2019.8697972.
- [8] A. A. Zadeh and H. M. Heys, "Theoretical simple power analysis of the Grain stream cipher," in *2013 26th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, May 2013, pp. 1–5, doi: 10.1109/CCECE.2013.6567847.
- [9] Y. Watanabe, Y. Todo, and M. Morii, "New conditional differential cryptanalysis for NLFSR-based stream ciphers and application to Grain v1," in *2016 11th Asia Joint Conference on Information Security (AsiaJCIS)*, Aug. 2016, pp. 115–123, doi: 10.1109/AsiaJCIS.2016.26.
- [10] A. R. Kazmi, M. Afzal, M. F. Amjad, H. Abbas, and X. Yang, "Algebraic side channel attack on trivium and Grain ciphers," *IEEE Access*, vol. 5, pp. 23958–23968, 2017, doi: 10.1109/ACCESS.2017.2766234.
- [11] M. Hell, T. Johansson, W. Meier, J. Sönnerup, and H. Yoshida, "An AEAD variant of the Grain stream cipher," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11445 LNCS, 2019, pp. 55–71.
- [12] K. Ngo, E. Dubrova, and M. Moraitis, "Attacking trivium at the bitstream level," in *2020 IEEE 38th International Conference on Computer Design (ICCD)*, Oct. 2020, vol. 2020-October, pp. 640–647, doi: 10.1109/ICCD50377.2020.00110.
- [13] R. Chatterjee, R. Chakraborty, and J. K. Mandal, "Design of cryptographic model for end-to-end encryption in FPGA based systems," in *2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)*, Mar. 2019, pp. 459–465, doi: 10.1109/ICCMC.2019.8819761.


- [14] L. Ding, C. Liu, Y. Zhang, and Q. Ding, "A new lightweight stream cipher based on chaos," *Symmetry*, vol. 11, no. 7, p. 853, Jul. 2019, doi: 10.3390/sym11070853.
- [15] S. Pan, Y. Wu, and L. Wang, "Optimizing fast near collision attack on Grain using linear programming," *IEEE Access*, vol. 7, pp. 181191–181201, 2019, doi: 10.1109/ACCESS.2019.2959334.
- [16] A. Perez-Resca, M. Garcia-Bosque, C. Sanchez-Azqueta, and S. Celma, "A new method for format preserving encryption in high-data rate communications," *IEEE Access*, vol. 8, pp. 21003–21016, 2020, doi: 10.1109/ACCESS.2020.2968816.
- [17] M. Tsavos, N. Sklavos, and G. P. Alexiou, "Lightweight security data streaming, based on reconfigurable logic, for FPGA platform," in *Proceedings - Euromicro Conference on Digital System Design, DSD 2020*, Aug. 2020, pp. 277–280, doi: 10.1109/DSD51259.2020.00052.
- [18] F. Dridi, S. E. Assad, W. E. H. Youssef, M. Machhout, and R. Lozi, "The design and FPGA-based implementation of a stream cipher based on a secure chaotic generator," *Applied Sciences (Switzerland)*, vol. 11, no. 2, pp. 1–19, Jan. 2021, doi: 10.3390/app11020625.
- [19] I. Salam, T. H. Ooi, L. Xue, W. C. Yau, J. Pieprzyk, and R. C. W. Phan, "Random differential fault attacks on the lightweight authenticated encryption stream cipher Grain-128AEAD," *IEEE Access*, vol. 9, pp. 72568–72586, 2021, doi: 10.1109/ACCESS.2021.3078845.
- [20] L. Huang, Y. Sun, J. Xiang, and L. Wang, "Image encryption based on a novel memristive chaotic system, Grain-128a algorithm and dynamic pixel masking," *Journal of Systems Engineering and Electronics*, vol. 33, no. 3, pp. 534–550, Jun. 2022, doi: 10.23919/JSEE.2022.000053.
- [21] M. Hell, T. Johansson, and W. Meier, "Grain: a stream cipher for constrained environments," *International Journal of Wireless and Mobile Computing*, vol. 2, no. 1, pp. 86–93, 2007, doi: 10.1504/IJWMC.2007.013798.
- [22] M. Hell, T. Johansson, A. Maximov, and W. Meier, "A stream cipher proposal: Grain-128," in *IEEE International Symposium on Information Theory - Proceedings*, Jul. 2006, pp. 1614–1618, doi: 10.1109/ISIT.2006.261549.
- [23] P. Kitsos, N. Sklavos, G. Provelengios, and A. N. Skodras, "FPGA-based performance analysis of stream ciphers ZUC, Snow3g, Grain V1, Mickey V2, Trivium and E0," *Microprocessors and Microsystems*, vol. 37, no. 2, pp. 235–245, Mar. 2013, doi: 10.1016/j.micpro.2012.09.007.
- [24] B. Li, M. Liu, and D. Lin, "FPGA implementations of Grain v1, Mickey 2.0, trivium, lizard and plantlet," *Microprocessors and Microsystems*, vol. 78, p. 103210, Oct. 2020, doi: 10.1016/j.micpro.2020.103210.
- [25] F. Alharbi, M. K. Hameed, A. Chowdhury, A. Khalid, A. Chattopadhyay, and I. T. Javed, "Analysis of area-efficiency vs. unrolling for eStream hardware portfolio stream ciphers," *Electronics*, vol. 9, no. 11, p. 1935, Nov. 2020, doi: 10.3390/electronics9111935.

BIOGRAPHIES OF AUTHORS






Raghavendra Ananth    hold a bachelor's degree in Electronics and communication engineering from HMSIT College, Tumkur and Master's degree in VLSI Design and embedded systems from REVA ITM College, Bangalore, Karnataka. He is a Research scholar in the Department of Electronics and Communications, JAIN Deemed to be University, Bangalore. His areas of research include IoT security, VLSI Frontend design and FPGA. He can be contacted at email: araghavendra.research@gmail.com.



Panduranga Rao Malode Vishwanatha Rao    obtained his Ph.D. from the National Institute of Technology Karnataka, Mangalore, India. He has completed a Master of Technology in computer science and a Bachelor of Engineering in electronics and communication. He works as a Professor in Jain (Deemed to be University) Bengaluru, India. His research interests are in the field of real-time and embedded systems on Linux platforms. He has published various research papers in journals and conferences across India. He visited JAPAN in 2008 for the IEEE international conference in Okinawa Island. He has authored two reference books on Linux Internals. He is a Life member of the Indian Society for Technical Education and IAENG. In the past three years, he has published 12 Indian patents. Here, three patents are stepping towards award/grant status. He can be contacted at email: r.panduranga@jainuniversity.ac.in.



Narayana Swamy Ramaiah    received a Ph.D. from PRIST University, Tamilnadu, in the year 2016, M.Tech. in the year 2004 from Visvesvaraya Technological University, Karnataka and a B.E. in the year 2002 from Bangalore University, Karnataka. He has over 16 years of experience in teaching, research and industry. He has published over 45 papers in peer-reviewed journals. His research areas include IoT (agriculture), blockchain, artificial intelligence and machine learning, and cloud computing. He works as a Professor Department of CE and Dean of academics at Sankalchand Patel University Visnagar, Gujarat. He can be contacted at email: narayanaswamy.ramaiah@gmail.com.