

Field programmable gate array implementation frameworks of a variable-length pseudorandom pattern generator

Geethu Remadevi Somanathan, Ramesh Bhakthavatchalu

Department of Electronics and Communication Engineering, Amrita Vishwa Vidyapeetham, Amritapuri, India

Article Info

Article history:

Received Dec 22, 2023

Revised Sep 21, 2024

Accepted Oct 1, 2024

Keywords:

Built-in self-test

Cryptography

Linear feedback shift register

Primitive and non-primitive

polynomials

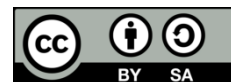
Pseudo-random pattern

generation

ABSTRACT

A pseudorandom pattern generator produces sequences similar to true random sequences. A variable length pseudo-random pattern generator (PRPG), which can be used as a pattern generator for various applications like built-in self-test (BIST) or cryptography, is proposed in this paper. Our work is based on a linear feedback shift register circuit platform. The proposed design can generate patterns corresponding to different characteristic polynomials of any given polynomial degree. These characteristic polynomials are integrated into the linear feedback shift register circuit by providing an option to select the feedback paths of any of these polynomials. This paper implements and evaluates the proposed design for primitive and non-primitive characteristic polynomials of degrees 3 to 15. The circuit generates output patterns of different periods based on user inputs. Compared to other pseudorandom pattern generator circuits, the proposed circuit can generate a large set of patterns and consumes less power. Adequate results from the experiments demonstrate the functionalities and performance of the proposed pattern generator from degrees 3 to 15. The proposed circuit generates pseudorandom patterns that can be used not only for built-in self-test but also for cryptography and wireless communication applications.

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



Corresponding Author:

Geethu Remadevi Somanathan

Department of Electronics and Communication Engineering, Amrita Vishwa Vidyapeetham

Amritapuri, India

Email: geethu.amrita@gmail.com

1. INTRODUCTION

A pseudorandom binary sequence (PRBS) or pseudorandom number is a binary sequence that shows statistical behavior comparable to a genuine random sequence. The pseudo-random pattern generator (PRPG) creates a sequence of integers with similar qualities to random numbers [1]. The sequence of patterns starts from an arbitrary state by using an initial state also known as seed value. Thus, if the starting sequence is known, all the sequences can be regenerated later, making them deterministic. Pseudorandom patterns are used for applications like cryptography [2]–[4], integrated circuit (IC) testing [5]–[12], where, it is used to supply patterns required to test a circuit under test (CUT), information theory [13]–[15] and telecommunication [16], [17]. The maximum length sequence produced by a linear feedback shift register (LFSR) is a typical example. LFSRs are extensively used in many electronic equipment that require the generation of pseudo-random sequences for their operation. Secure communications also use LFSRs as fundamental building blocks for stream ciphers. LFSRs can be used to build robust physical unclonable functions (PUFs) in cryptography applications.

A LFSR generates a pseudo-random sequence based on an initial seed (s_i). The linear function for generating the pseudo random sequence is the characteristic polynomial on the basis of which XOR tap

points for the LFSR are fixed [18]–[21]. The bits in the LFSR which influence the pseudo random sequence generation are called taps. The linear function of LFSR is generated by connecting the XOR function at tap points, which in turn depends on the characteristic polynomial. The polynomial degree determines the number of D flip-flops to be employed and the location of XOR taps (x_i) in the circuit. The output pseudorandom sequences obtained from the LFSR is as per (1):

$$s_k = \sum_{i=0}^n x_i s_{k-1} \quad (1)$$

where s_k is output sequence and $k \geq n$ [19]. The D registers decide bit size in the output sequences, also known as word length. The longest feasible sequence intended by LFSR for a number of bits n is $2^n - 1$ [22]. The polynomial that generates $2^n - 1$ sequences is recognized as a primitive polynomial and those polynomials generating lesser number of sequences are non-primitive in nature. Using characteristic polynomial, different circuit structures of LFSR can be implemented, namely Fibonacci (external or standard), Galois (internal or modular), reseeding, and complete LFSRs [22]–[24] and the proposed design is based on a Fibonacci LFSR structure. The Fibonacci and Galois LFSR structures are duals of each other for a given characteristic's polynomial. Uniformity of the shift register forms the main advantage of a Fibonacci structure [25], [26] whereas, Galois LFSR functions faster than its Fibonacci version due to the placement of taps internal to the shift register, thus resulting in maximum delay equal to that of a single XOR gate. Primitive polynomials, which generate maximal length sequence $2^n - 1$, and non-primitive polynomials, which generate sequences $2^n - 1$ are used in our design. Reseeding configurations enable the circuit to include external seed values and complete LFSR structures generates all $2^n - 1$ sequences by leveraging NOR gates.

Many researchers have experimented with LFSR and numerous variations are also being explored [11], [22], [23], [27]–[32]. The LFSRs are extensively used in IC testing and cryptographic applications since they are capable of producing pseudo random sequences. Normally, the consecutive sequences generated by a LFSR are assured to be least correlated as they differ at more bit positions. But this increased switching activity results in increased power dissipation. For compact and miniature hardware implementations for enhanced portability and performance, the power dissipation should be further reduced. The primary aspect considered for addressing the issue of power dissipation in LFSR is the switching activity. Instead of XOR, XNOR gates can also be exercised as tap points [22], but the former is more commonly used [33], [34]. While the XOR gate taps bring the circuit to a lock state for the LFSR register in all outputs zero state, the presence of XNOR gate causes the circuit to arrive in a lock state for all outputs one. In the proposed design, we have considered XNOR gates in the tap points since the circuit can be driven to the initial state by enabling the clear pin. An analysis of Galois and Fibonacci LFSR behavior is done in [22], [27], where a polynomial is configured to work as a modular and standard structure based on the control signal. While the experiments of [22] are done using XNOR gates, the experiments in [27] use XOR gates in the feedback path. Even though both these works showcase the difference between the patterns generated for both types of LFSRs and the usage of XOR and XNOR, they still need to explore the possibilities of changing the polynomials. This work proposes to provide polynomial level reconfigurability to LFSR architectures. The polynomial level reconfigurability ensures the generation of more numbers of significant output patterns from the LFSRs than [22], [24].

Even though the LFSR circuits are commonly used in cryptographic applications, the knowledge of the origin/initial value of LFSR enables the prediction of output sequences. The nature of predictability of the output patterns can be reduced, which increases the security, by making modifications in the existing LFSR circuit. Ramasamy and Samiappan [35] suggests a vertically stacked structure of LFSR that changes LFSR PRBS sequence length and increases the complexity of deciphering the code in security applications. The trade-offs in this design, like an increase in hardware complexity, are addressed in [36], where an inexpensive LFSR-based secured architecture is proposed without affecting testability and the authors present investigative results with a secure methodology that steers to low power and area overhead.

Another application of LFSR is in logic built-in self-test (LBIST), which initially evolved for board-level, system-level, and in-field tests, and has recently become more common. Test pattern generator produces output patterns that can strategically promote fault coverage in a built-in self-test (BIST). An LFSR can be used for the different classes of test patterns: deterministic, algorithmic, exhaustive, pseudo-exhaustive, pseudo-random, and random test patterns, which are not mutually exclusive. LFSRs are the most preferred circuits for test pattern generation (TPG) among the other TPG circuits such as finite state machine (FSMs), counters, and cellular automata, because of the improved hardware area efficiency, enhanced fault coverage and lesser combinational logic requirements. Remodeling an LFSR circuit using weighted patterns can further improve the fault coverage by targeting the random-pattern-resistant (rpr) faults. Along with these

advantages, LFSR circuits address the issues of structural and linear dependencies more precisely than cellular automata (CA) [25].

Several circuits and methods are available for generating pseudorandom patterns. While some generate all patterns at the cost of high-power requirements, some utilize more hardware. Many of the circuits does not discuss the reconfigurability, which can greatly improve the diversity of generated patterns. For a specific polynomial degree n , the proposed design is thus capable of: i) developing different sets of pseudorandom patterns based on different polynomials, or configurability; ii) less area overhead compared to related circuits; and iii) ability to generate patterns while consuming less power. This paper's remaining content is arranged as follows. Section 2 outlines the proposed design with its methodology covered in section 3. Section 4 covers results of functional verification and the field programmable gate array (FPGA) implementation frameworks followed by section 5 which concludes the paper. Table 1 contains tabular data summarizing the current and proposed PRPG.

Table 1. Summary of related works

Related work	Configurability	Length of cycle	Hardware utilization	Power required
[22]	yes	$2^n - 1$	low	high
[24]	yes	$2^n - 1$	low	high
[27]	yes	2^n	high	high
[28]	yes	$2^n - 1$	moderate	high
proposed	yes	$2^n - 1$	low	option to reduce

2. METHOD

A generic circuit of the proposed design for an n -bit polynomial is given in Figure 1. Polynomial degree will determine the number of flip-flops employed in the circuit. The circuit is realized using the Fibonacci architecture for LFSR. The multiplexer block connects the feedback path of polynomial to the register bank.

A 5-bit representation of the design is given in Figure 2. Four polynomials with degree same as number of flip-flops are integrated into circuit using multiplexer. The *pattern selector* input decides the polynomial and thus the generated patterns. *Clear* pins of D Flip-flops share a common connection and ensures the starting sequence of all zero pattern. Table 2 shows the *pattern selector* value, the corresponding polynomial, and output sequences. Depending on whether the polynomial is primitive or non-primitive, the length of sequences will be $2^n - 1$ or, a lesser number respectively.

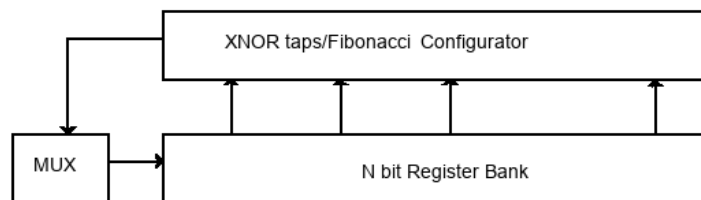


Figure 1. n-bit model of proposed design

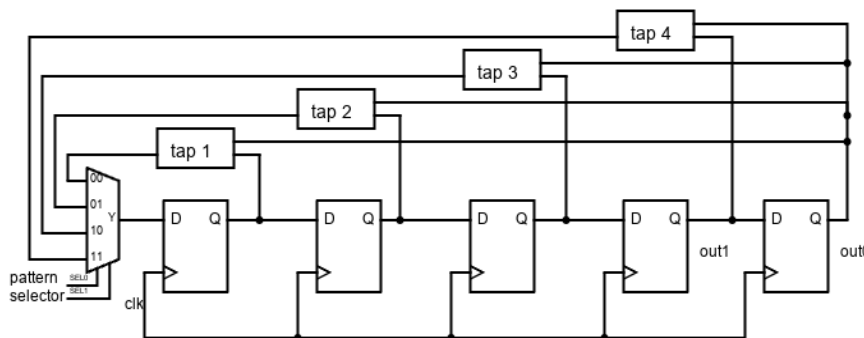


Figure 2. Proposed design configured for polynomial degree 5

Table 2. Output sequences of proposed PRPG configured for polynomial degree 5

Pattern selector	Polynomial	No. of patterns	Generated output patterns
0	$x^5 + x^4 + 1$	21	00,10,18,1c,1e,0f,17,1b,1d,0e,07,13,19,0c,16,0b,15,0a,05,02,01
1	$x^5 + x^3 + 1$	31	00,10,18,1c,0e,07,13,09,04,02,11,08,14,0a,15,1a,1d,1e,0f,17,1b,0d,16,0b,05,12,19,0c,06,03,01
2	$x^5 + x^2 + 1$	31	00,10,18,0c,06,13,09,14,19,0d,16,1b,1d,1e,0f,17,0b,15,0a,05,02,11,08,04,12,19,1c,0e,07,03,01
3	$x^5 + x + 1$	21	00,10,08,14,0a,15,1a,0d,06,13,19,1c,0e,17,1b,1d,1e,0f,07,03,01

When *pattern selector* value is 3, the circuit is configured to work as the polynomial $x^5 + x^4 + 1$, and since this is a non-primitive polynomial, the number of patterns is 21. For *pattern selector* value 2, the circuit works according to the primitive polynomial $x^5 + x^3 + 1$, and thus generates all $2^n - 1$ patterns *i.e.*, 31. Primitive polynomial $x^5 + x^2 + 1$ is opted when the *pattern selector* is changed to 1 and all $2^n - 1$ outputs are generated. When the *pattern selector* value is revised to 0, the proposed design is configured to the non-primitive polynomial $x^5 + x + 1$ which generates 21 patterns. Thus, the feedback paths given as inputs to the multiplexer are a combination of primitive and non-primitive polynomials for lower degree polynomials. For higher degree polynomials, the feedback paths are that of only primitive polynomials. For polynomial degree 5, the blend of primitive and non-primitive polynomials produces run lengths of 31 and 21 respectively. Similarly, the run lengths vary according to the change in degree of the polynomial for higher-degree characteristic polynomials. The combination of primitive and non-primitive polynomials ensures the variety of patterns created and thus the use of the proposed design in generating ciphertexts in cryptographic applications and for test pattern generation in BIST applications. The experiments were performed from polynomial degree 3 to 15 and performance was analyzed. The pattern length used for the analysis varies from 4 to 32,767. In all these cases, four sets of sequences are obtained at the output by varying *pattern selector*, and the size of the multiplexer is kept same in all cases. For each polynomial degree, the *pattern selector* can choose between a blend of feedback paths of primitive and non-primitive polynomials which ensures all $2^n - 1$ patterns or a lesser number.

3. IMPLEMENTATION

Description of the proposed circuit is done using Verilog hardware description language (HDL) and verification of functionality is executed using Xilinx ModelSim. FPGAs are desirable for prototyping due to the high performance achieved by prompting custom application-specific architectures. Immediate computational speedup and low energy consumption are often achieved by implementing remarkably optimized data paths. The Xilinx Vivado 2015.1 is employed for the synthesis, verification, implementation, and power assessment of the proposed design. The board used for FPGA prototyping is from the Artix-7 family. Also, the board has an internal clock of more than 450MHz so we simulated the proposed circuit for different frequencies: 100, 200, and 500 MHz. For a specific polynomial degree, four different sets of patterns are generated based on four polynomials. The different polynomials used in the experiments along with the number of output sequences generated are mentioned in next section along with the results of the simulation of the proposed design.

4. PERFORMANCE ANALYSIS AND RESULTS

The results of verification of functionality and FPGA implementation of the proposed design are discussed in this section. The functional verification is performed using ModelSim and the results are presented. Implementation in FPGA is done by Xilinx Vivado tool.

4.1. Function verification

Verification of the function of the proposed design was performed for all the cases and polynomials. When the polynomial degree is 5 and the pattern selector value is set to 1, the proposed design is configured to work as given in Figure 3. The polynomial configured for this input selection is $x^5 + x^3 + 1$. Since this is a primitive polynomial, all $2^n - 1$ patterns are generated. The output patterns obtained in functional verification for the above-said input combination are given in Figure 4. In Figure 4, the simulation results corresponding to degree 5 are presented, where the value of the pattern selector is kept as 01. Similarly, the functional verification was performed for all polynomials from 3 to 15 as given in Table 3 details of polynomial 5, is in Table 2. All the polynomial degrees from 3 to 15 contain four polynomials, a mix of primitive and non-primitive polynomials. For a specific degree, when *pattern selector* value increases, the number of patterns generated increases. Also, when the value is 0, the circuit consumes less power as the number of tap points are less.

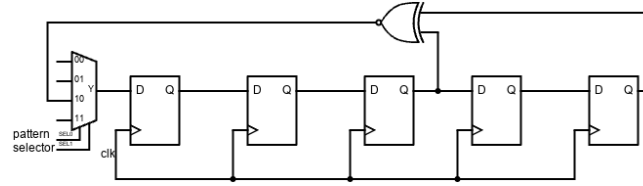


Figure 3. Proposed circuit configured for polynomial degree 5 with *pattern selector* set to 1

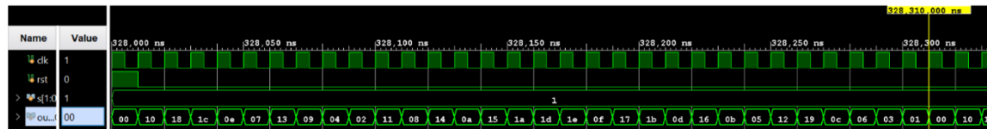


Figure 4. Output for polynomial degree 5 and *pattern selector* 01

Table 3. Polynomial degree 3 to 15 sequences

Polynomial degree	Pattern selector	Polynomial	No. of o/p sequence
3	0	$x^3 + x^2 + x + 1$	4
	1	$x^3 + x^2 + x + 1$	7
	2	$x^3 + x + 1$	7
4	3	$x^3 + x^2 + x + 1$	7
	0	$x^4 + x^2 + 1$	6
	1	$x^4 + x^3 + x + 1$	12
6	2	$x^4 + x^3 + 1$	15
	3	$x^4 + x + 1$	15
	0	$x^6 + x^4 + 1$	14
7	1	$x^6 + x^2 + 1$	14
	2	$x^6 + x^5 + 1$	63
	3	$x^6 + x + 1$	63
8	0	$x^7 + x^5 + 1$	93
	1	$x^7 + x^6 + 1$	127
	2	$x^7 + x^3 + 1$	127
9	3	$x^7 + x + 1$	127
	0	$x^8 + x^4 + 1$	12
	1	$x^8 + x + 1$	63
10	2	$x^8 + x^7 + 1$	63
	3	$x^8 + x^6 + x^5 + x + 1$	255
	0	$x^9 + x^8 + 1$	73
11	1	$x^9 + x + 1$	73
	2	$x^9 + x^4 + 1$	511
	3	$x^9 + x^5 + 1$	511
12	0	$x^{10} + x^4 + 1$	15
	1	$x^{10} + x^8 + 1$	889
	2	$x^{10} + x + 1$	1023
13	3	$x^{10} + x^5 + 1$	1023
	0	$x^{11} + x^{10} + 1$	1533
	1	$x^{11} + x^7 + 1$	1533
14	2	$x^{11} + x^4 + 1$	1533
	3	$x^{11} + x^2 + 1$	2047
	0	$x^{12} + x^8 + 1$	28
15	1	$x^{12} + x^4 + 1$	819
	2	$x^{12} + x^{11} + 1$	3255
	3	$x^{12} + x^7 + x^4 + x^3 + 1$	4095
16	0	$x^{13} + x^9 + 1$	7161
	1	$x^{13} + x^6 + 1$	7665
	2	$x^{13} + x^{12} + 1$	7905
17	3	$x^{13} + x^4 + x^3 + 1$	8191
	0	$x^{14} + x^9 + 1$	254
	1	$x^{14} + x^6 + 1$	5461
18	2	$x^{14} + x^4 + x^3 + 1$	11811
	3	$x^{14} + x^{12} + x^{11} + x + 1$	16383
	0	$x^{15} + x^6 + 1$	35
19	1	$x^{15} + x^9 + 1$	93
	2	$x^{15} + x^{12} + 1$	32767
	3	$x^{15} + x^4 + x^3 + 1$	32767

4.2. FPGA implementation

The proposed design is implemented and analyzed using three different frequencies: 100, 200, and 500 MHz. Table 4 gives insights into the performance of the design for all these different frequencies. worst negative slack (WNS) is the worst negative slack, while total negative slack (TNS) is the total of setup violations for all endpoints in the design. WHS is the worst hold slack, and total hold slack (THS) is the total of hold violations for all of the design's endpoints. When all timing checks (TNS, THS, or TPWS) are met, the total slack is void. The timing metrics of i) Setup or max delay analysis: $WNS > 0\text{ ns}$ and $TNS = 0\text{ ns}$; ii) hold or min delay analysis: $WHS > 0\text{ ns}$ and $THS = 0\text{ ns}$; and iii) worst pulse width slack (WPWS) $> 0\text{ ns}$ and total pulse width negative slack (TPWS) = 0 ns , show the timing closure of the proposed design. The implementation of the proposed design in all three frequencies generated the aforementioned values in timing closure. All the polynomials mentioned in Table 3 were implemented and when the polynomial degree increases from 3 to 15, the total delay shows an increase of 2.446 ns only. Figure 5 shows the increase in total delay for the proposed design for all three frequencies (delay in all cases remains the same with minor variations). Slack values in all the cases also meet the timing requirements. When WNS is a negative value, it shows that the timing constraint is violated, and when $WNS > 0\text{ ns}$ it indicates that the design meets timing constraints, and a positive slack of $x\text{ ns}$ is intended to mean that an extra delay of $x\text{ ns}$ can be tolerated [37].

Table 4. Implementation results for different frequencies

Poly degree	Clk (ns)	Total delay	WNS (ns)	Utilization			Total On chip power (W)	Dynamic power dissipation (W)	Static power dissipation (W)	Junction temp. (°C)
				FF	LUT	IO				
3	10	1.001	8.994	3	2	7	0.075	5%(0.004)	95%(0.072)	25.4
4	10	1.185	8.745	4	2	8	0.084	15%(0.012)	85%(0.072)	25.4
5	10	1.342	8.652	5	3	9	0.087	17%(0.015)	83%(0.072)	25.4
6	10	1.728	8.267	6	3	10	0.09	20%(0.018)	80%(0.072)	25.4
7	10	1.413	8.62	7	3	11	0.093	23%(0.021)	77%(0.072)	25.5
8	10	1.413	8.62	8	3	12	0.096	25%(0.024)	75%(0.072)	25.5
9	10	1.827	8.207	9	3	13	0.099	27%(0.027)	73%(0.072)	25.5
10	10	2.227	7.842	10	3	14	0.102	29%(0.03)	71%(0.072)	25.5
11	10	2.331	7.552	11	3	15	0.105	31%(0.033)	69%(0.072)	25.5
12	10	2.096	7.838	12	4	16	0.108	33%(0.036)	67%(0.072)	25.5
13	10	3.167	6.73	13	4	17	0.111	35%(0.039)	65%(0.072)	25.6
14	10	2.321	7.572	14	3	18	0.114	37%(0.042)	63%(0.072)	25.6
15	10	3.447	6.448	15	3	19	0.117	38%(0.045)	62%(0.072)	25.6
3	20	1.001	18.994	3	2	7	0.074	3%(0.002)	97%(0.072)	25.4
4	20	1.185	18.745	4	2	8	0.078	8%(0.006)	92%(0.072)	25.4
5	20	1.342	18.652	5	3	9	0.079	10%(0.008)	90%(0.072)	25.4
6	20	1.728	18.267	6	3	10	0.081	11%(0.009)	89%(0.072)	25.4
7	20	1.413	18.62	7	3	11	0.082	13%(0.01)	87%(0.072)	25.4
8	20	1.413	18.62	8	3	12	0.084	14%(0.012)	86%(0.072)	25.4
9	20	1.827	18.207	9	3	13	0.085	16%(0.013)	84%(0.072)	25.4
10	20	2.227	17.842	10	3	14	0.087	17%(0.015)	83%(0.072)	25.4
11	20	2.331	17.552	11	3	15	0.088	19%(0.016)	81%(0.072)	25.4
12	20	2.096	17.712	12	4	16	0.09	20%(0.018)	80%(0.072)	25.4
13	20	3.167	16.73	13	4	17	0.091	21%(0.019)	79%(0.072)	25.5
14	20	2.321	17.572	14	3	18	0.093	23%(0.021)	77%(0.072)	25.5
15	20	3.447	16.448	15	3	19	0.094	24%(0.022)	76%(0.072)	25.5
3	50	1.001	48.994	3	2	7	0.072	1%(0.001)	99%(0.072)	25.4
4	50	1.185	48.745	4	2	8	0.074	3%(0.002)	97%(0.072)	25.4
5	50	1.342	48.652	5	3	9	0.075	4%(0.003)	96%(0.072)	25.4
6	50	1.728	48.267	6	3	10	0.075	5%(0.004)	95%(0.072)	25.4
7	50	1.497	48.497	7	3	11	0.076	6%(0.004)	94%(0.072)	25.4
8	50	1.562	48.433	8	3	12	0.077	6%(0.005)	94%(0.072)	25.4
9	50	1.827	48.207	9	3	13	0.077	7%(0.005)	93%(0.072)	25.4
10	50	2.227	47.842	10	3	14	0.078	8%(0.006)	92%(0.072)	25.4
11	50	2.292	47.589	11	3	15	0.078	8%(0.007)	92%(0.072)	25.4
12	50	2.227	47.662	12	4	16	0.079	8%(0.007)	92%(0.072)	25.4
13	50	3.038	46.862	13	4	17	0.08	10%(0.008)	90%(0.072)	25.4
14	50	2.307	47.585	14	3	18	0.08	10%(0.008)	90%(0.072)	25.4
15	50	3.523	46.411	15	3	19	0.081	11%(0.009)	89%(0.072)	25.4

Figure 6 compares the WNS values obtained for the different operational frequencies in our circuit. The WNS values obtained are compared with [22] and [24] and given in Figure 7, which implies that the proposed design has a better WNS which allows enough room for timing closure. Neshvad *et al.* [16] performed the experiments using a clock frequency of 500 MHz and this also is compared and proposed design and is found to be a better candidate. The sum of device static power and design power is the power consumed internally and is known as total on-chip power or thermal power. Figure 8 shows the variation of

total on-chip power for the proposed design across the different frequencies and polynomial degrees mentioned in Table 4. While the polynomial degree increases from 3 to 15, the total on-chip power increases from 0.075 to 0.117 W which is a difference of only 0.042 W. This indicates that when the proposed design works over word lengths of 3 to 15, the expected increase in total on-chip power is as low as 0.042 W. The total on-chip power is compared with [22] and [24] and is given in Figure 9. Static power comes from transistor leakage on all connected voltage rails and the circuits needed for regular device operation. The static power of the proposed design displays a 33% difference when the polynomial degree changes from 3 to 15. Even though the dynamic power dissipation is less in lower degrees, it increases as the polynomial degree is changed to 15. This implies that the static power will be reduced as the polynomial degree is higher. Junction temperature depends on factors like the total power of the device, the cooling system, board selection, and ambience. While changing from degrees 3 to 15, it is observed that the proposed design maintains the junction temperature in the range 25.4 °C to 25.6 °C, which is a nominal value and much less than [22] and [24]. Look-up-tables (LUT) are asynchronous static random-access memory (SRAM) that implement combinational logic, where contents can only be changed during FPGA configuration. The utilization of the design is analyzed and the number of flip-flops required shows a linear increase according to the number of output bits needed, which is justifiable. The number of flip-flops needed is the same as that of the polynomial degree, whereas the LUT required is in the range of 2 to 4. Figure 10 shows the schematic of the circuit obtained for the circuit given in Figure 3.

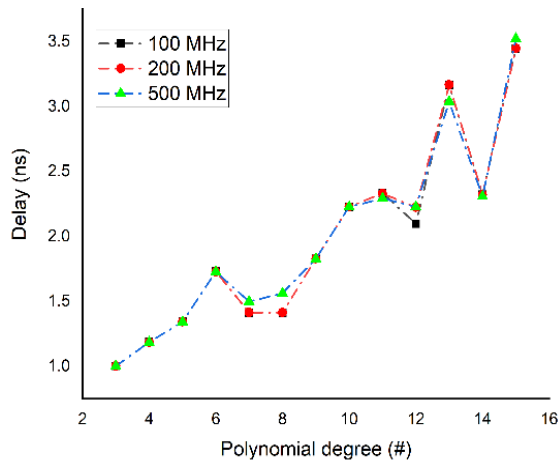


Figure 5. Variation of total delay across different frequencies

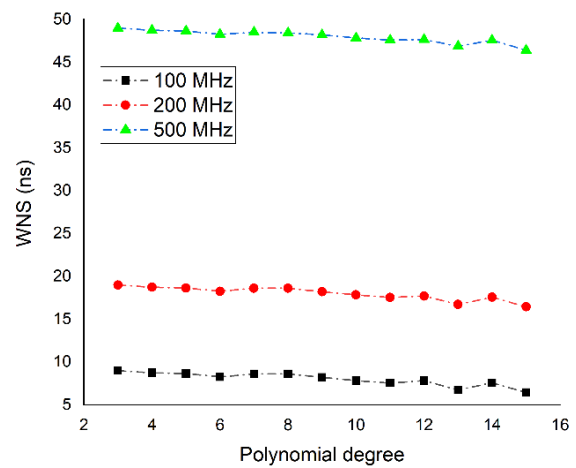


Figure 6. WNS values for all three frequencies

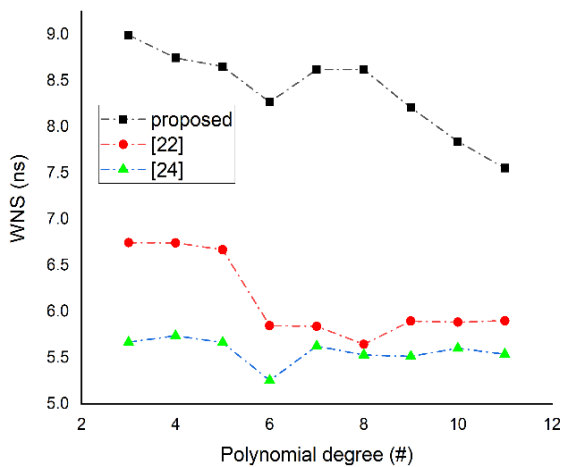


Figure 7. WNS comparison with proposed design

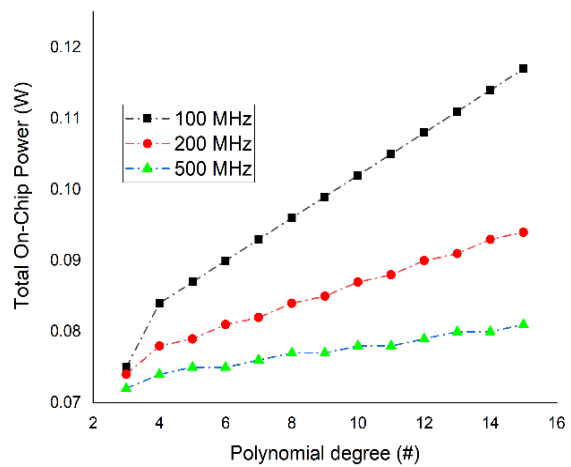


Figure 8. Variation of power with different frequencies

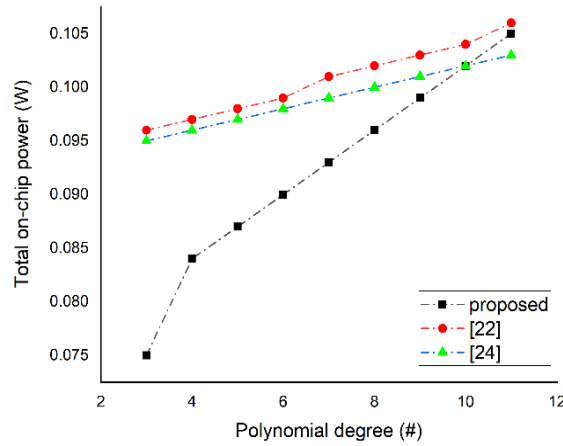


Figure 9. Total on-chip power comparison with existing PRPG

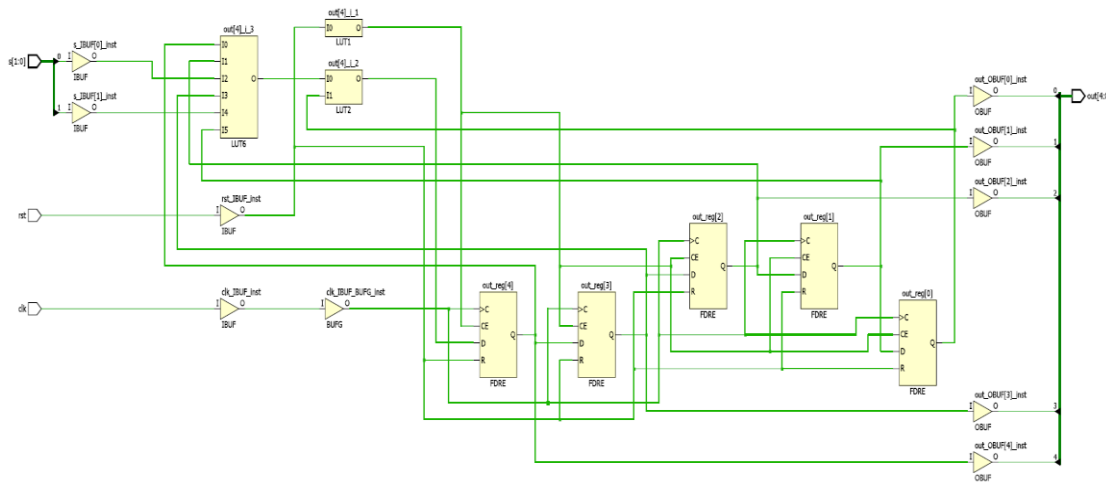


Figure 10. Register transfer level (RTL) Schematic of proposed design working for 5-bit

5. CONCLUSION AND FUTURESCOPE

This work suggests a variable-length pseudo-random sequence generator and circuit is able to generate patterns of 3 to 15-bit and length ranging from 4 to 32,767. Length of sequences is decided by the polynomial which is selected based on *pattern selector* input. The number of patterns generated as well as power required can be decided depending on this input. Feedback paths from primitive as well as non-primitive polynomials are given as input to the multiplexer which aids the generation of maximum patterns i.e. $2^n - 1$ or a lesser number. These pseudorandom patterns can be used for logic BIST as well as cryptography applications. The overall on-chip power decreases as operation frequencies increase, despite the total delay showing little fluctuation across operating frequencies. The difference in power required shows a difference of 0.042 W across polynomial degree 3 to 15, but only 0.009 W across polynomial degree 3 to 15 at 500 MHz. Less WNS values hint at a better timing closure compared to the existing PRPG. Experimental results show that the performance criteria for the proposed PRPG is better compared with the various PRPGs in terms of timing as well as power and is thus a suitable candidate for PRPG applications.

Suggested future scope is to upscale the multiplexer to accommodate more polynomials in a specific polynomial degree which facilitates more pattern sets for polynomial degree n . Other low-power schemes or toggle reduction methods can be included to analyze further power reduction possibilities. The proposed circuit can be used in several scenarios, among which BIST can be regarded as the primary application. Further research will evaluate the quality of pseudo-random patterns for cryptography applications, which may have stricter requirements than other applications. 15 empirical tests used by the National Institute of Standards and Technology (NIST) can ensure randomness across the bitstream, providing valuable insights into using the proposed design for cryptography applications.





REFERENCES

- [1] D. Xiang, X. Wen, and L. T. Wang, "Low-power scan-based built-in self-test based on weighted pseudorandom test pattern generation and reseeding," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 3, pp. 942–953, Mar. 2017, doi: 10.1109/TVLSI.2016.2606248.
- [2] R. Bezuidenhout, W. Nel, and J. M. Maritz, "Permissionless Blockchain systems as pseudo-random number generators for decentralized consensus," *IEEE Access*, vol. 11, pp. 14587–14611, 2023, doi: 10.1109/ACCESS.2023.3244403.
- [3] N. Babu T, F. Noorbasha, and L. C. Gunnam, "Implementation of high security cryptographic system with improved error correction and detection rate using FPGA," *International Journal of Electrical and Computer Engineering*, vol. 6, no. 2, pp. 602–610, Apr. 2016, doi: 10.11591/ijece.v6i2.9267.
- [4] Y. Wang *et al.*, "A lightweight authentication protocol against modeling attacks based on a novel LFSR-APUF," *IEEE Internet of Things Journal*, vol. 11, no. 1, pp. 283–295, Jan. 2024, doi: 10.1109/JIOT.2023.3314058.
- [5] E. J. McCluskey and S. Bozorgui-Nesbat, "Design for autonomous test," *IEEE Transactions on Circuits and Systems*, vol. 28, no. 11, pp. 1070–1079, Nov. 1981, doi: 10.1109/TCS.1981.1084930.
- [6] I. Pomeranz, "LFSR-based generation of multicycle tests," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 3, pp. 503–507, 2017, doi: 10.1109/TCAD.2016.2587687.
- [7] I. Pomeranz, "LFSR-based test generation for path delay faults," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 2, pp. 345–353, Feb. 2019, doi: 10.1109/TCAD.2018.2812120.
- [8] P. Girard, "Low power testing of VLSI circuits: problems and solutions," in *Proceedings - International Symposium on Quality Electronic Design, ISQED*, 2000, pp. 173–179, doi: 10.1109/ISQED.2000.838871.
- [9] A. Bosio, P. Girard, and A. Virazel, "Test of low power circuits: issues and industrial practices," in *2016 IEEE International Conference on Electronics, Circuits and Systems, ICECS 2016*, Dec. 2016, pp. 524–527, doi: 10.1109/ICECS.2016.7841254.
- [10] A. W. Hakmi *et al.*, "Programmable deterministic built-in self-test," in *Proceedings - International Test Conference*, 2008, pp. 1–9, doi: 10.1109/TEST.2007.4437611.
- [11] V. Shivakumar, C. Senthilpari, and Z. Yusoff, "A low-power and area-efficient design of a weighted pseudorandom test-pattern generator for a test-per-scan built-in self-test architecture," *IEEE Access*, vol. 9, pp. 29366–29379, 2021, doi: 10.1109/ACCESS.2021.3059171.
- [12] I. Pomeranz and S. Venkataraman, "LFSR-based test generation for reduced fail data volume," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 12, pp. 5261–5266, Dec. 2020, doi: 10.1109/TCAD.2020.2971527.
- [13] J. Kaur, A. C. Canto, M. M. Kermani, and R. Azarderakhsh, "Hardware constructions for error detection in WG-29 stream cipher benchmarked on FPGA," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 43, no. 4, pp. 1307–1311, Apr. 2024, doi: 10.1109/TCAD.2023.3338108.
- [14] M. Y. Hsiao, "Single-channel error correction in an F-channel system," *IEEE Transactions on Computers*, vol. C-17, no. 10, pp. 935–943, Oct. 1968, doi: 10.1109/TC.1968.226440.
- [15] Y. Wu, "High-speed LFSR decoder architectures for BCH and GII codes," *IEEE Journal on Selected Areas in Information Theory*, vol. 4, pp. 331–350, 2023, doi: 10.1109/JSAIT.2023.3304235.
- [16] S. Neshvad, S. Chatzinotas, and J. Sachau, "Wideband identification of power network parameters using pseudo-random binary sequences on power inverters," *IEEE Transactions on Smart Grid*, vol. 6, no. 5, pp. 2293–2301, Sep. 2015, doi: 10.1109/TSG.2015.2397552.
- [17] M. Sasmal, T. Joseph, and T. S. Bindiya, "Approximate multiplier design with LFSR-based stochastic sequence generators for edge AI," *IEEE Computer Architecture Letters*, vol. 23, no. 1, pp. 91–94, Jan. 2024, doi: 10.1109/LCA.2024.3379002.
- [18] M. Bushnell and V. D. Agrawal, "Built-in self-test," *Essentials of Electronic Testing for Digital, Memory and Mixed-signal VLSI Circuits*, pp. 489–512, 2000.
- [19] F. Karimzadeh, N. Cao, B. Crafton, J. Romberg, and A. Raychowdhury, "A hardware-friendly approach towards sparse neural networks based on LFSR-generated pseudo-random sequences," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, no. 2, pp. 751–764, Feb. 2021, doi: 10.1109/TCSI.2020.3037028.
- [20] H. Bae *et al.*, "High-speed counter with novel LFSR state extension," *IEEE Transactions on Computers*, vol. 72, no. 3, pp. 893–899, 2023, doi: 10.1109/TC.2022.3187343.
- [21] Y. Chen, Y. Tian, R. Zhou, D. Martinez Castro, D. Guo, and Q. Zhou, "NDSTRNG: non-deterministic sampling-based true random number generator on SoC FPGA systems," *IEEE Transactions on Computers*, vol. 73, no. 5, pp. 1313–1326, May 2024, doi: 10.1109/TC.2024.3365955.
- [22] C. S. Vikranth, K. Rakesh, B. Jagadeesh, D. Mohammad, G. R. Somanathan, and R. Bhakthavatchalu, "Design and analysis of test pattern generator by combining internal and external LFSR," in *Proceedings of the 5th International Conference on Trends in Electronics and Informatics, ICOEI 2021*, Jun. 2021, pp. 240–244, doi: 10.1109/ICOEI51242.2021.9452847.
- [23] M. Goresky and A. M. Klapper, "Fibonacci and Galois representations of feedback-with-carry shift registers," *IEEE Transactions on Information Theory*, vol. 48, no. 11, pp. 2826–2836, Nov. 2002, doi: 10.1109/TIT.2002.804048.
- [24] C. S. Vikranth, D. Mohammad, G. R. Somanathan, and R. Bhakthavatchalu, "Analysis of a novel reseeding pattern generator," in *Proceedings - 2nd International Conference on Smart Electronics and Communication, ICOSEC 2021*, Oct. 2021, vol. 3, pp. 676–682, doi: 10.1109/ICOSEC51865.2021.9591905.
- [25] C. E. Stroud, "Test pattern generation," in *A Designer's Guide to Built-In Self-Test*, 1st ed., vol. 19, New York, USA: Springer New York, 2002, pp. 61–80.
- [26] Zulfikar, Y. Away, and R. S. Noor, "FPGA-based design system for a two-segment fibonacci LFSR random number generator," *International Journal of Electrical and Computer Engineering*, vol. 7, no. 4, pp. 1882–1891, Aug. 2017, doi: 10.11591/ijece.v7i4.pp1882-1891.
- [27] S. K. S, G. R. S, and R. Bhakthavatchalu, "FPGA implementation of parameterizable pattern-generator modules for LBIST applications," in *2023 4th International Conference for Emerging Technology (INCET)*, May 2023, pp. 1–5, doi: 10.1109/INCET57972.2023.10170045.
- [28] S. Abdel-Hafeez, "Programmable feedback shift register," *Circuits, Systems, and Signal Processing*, vol. 42, no. 8, pp. 4784–4808, Mar. 2023, doi: 10.1007/s00034-023-02332-3.
- [29] L. Shaer, T. Sakakini, R. Kanj, A. Chehab, and A. Kayssi, "A low power reconfigurable LFSR," in *Proceedings of the 18th Mediterranean Electrotechnical Conference: Intelligent and Efficient Technologies and Services for the Citizen, MELECON 2016*, Apr. 2016, vol. 36, pp. 1–4, doi: 10.1109/MELCON.2016.7495422.
- [30] J. Lee and N. A. Touba, "LFSR-reseeding scheme achieving low-power dissipation during test," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 2, pp. 396–401, Feb. 2007, doi:





- 10.1109/TCAD.2006.882509.
- [31] M. Puczko, "Low power test pattern generator for BIST," in *2015 Selected Problems of Electrical Engineering and Electronics, WZEE 2015*, Sep. 2016, vol. 73, pp. 1–6, doi: 10.1109/WZEE.2015.7394030.
- [32] R. Kapur, S. Paul, T. J. Snethen, and T. W. Williams, "A weighted random pattern test generation system," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, no. 8, pp. 1020–1025, 1996, doi: 10.1109/43.511581.
- [33] K. Madhulatha, K. Jyothika, S. V. B. Harini, G. R. Somanathan, and R. Bhakthavathchalu, "Reconfigurable linear feedback shift register," in *Proceedings - 4th International Conference on Smart Systems and Inventive Technology, ICSSIT 2022*, Jan. 2022, vol. 656, pp. 641–647, doi: 10.1109/ICSSIT53264.2022.9716565.
- [34] G. Giustolisi, R. Mita, G. Palumbo, and G. Scotti, "A novel clock gating approach for the design of low-power linear feedback shift registers," *IEEE Access*, vol. 10, pp. 99702–99708, 2022, doi: 10.1109/ACCESS.2022.3207151.
- [35] J. Ramasamy and D. Samiappan, "A modified PRBS: vertical stacked LFSR primitive polynomial for secure data communication," *Procedia Computer Science*, vol. 215, pp. 947–954, 2022, doi: 10.1016/j.procs.2022.12.097.
- [36] M. I. Shiny and M. N. Devi, "LFSR-based secured scan design testability techniques," *Procedia Computer Science*, vol. 115, pp. 174–181, 2017, doi: 10.1016/j.procs.2017.09.123.
- [37] J. Vygen, "Slack in static timing analysis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 9, pp. 1876–1885, Sep. 2006, doi: 10.1109/TCAD.2005.858348.

BIOGRAPHIES OF AUTHORS



Geethu Remadevi Somanathan     received B.E. in electronics and communication engineering from MS University in 2003 and M.Tech. in microelectronics and VLSI from National Institute of Technology, Calicut in 2012. Since 2004, she is assistant professor at the Department of Electronics and Communication, Amrita Vishwa Vidyapeetham, Amritapuri, Kerala, India. Her research interests include digital circuits and systems, secured testing, security of logic circuits, testing of VLSI circuits. She can be contacted at email: geethu.amrita@gmail.com.



Ramesh Bhakthavathchalu     received his B.E. from Vellore Institute of Technology in April 1994. He completed his M.E in applied electronics from the College of Engineering, Anna University, Chennai in the year 1998. He served as senior design application engineer in Cirrus Logic Inc., USA, and SynTest Technologies, USA, for 8 years. He has taped out 4 designs during his tenure in VLSI Industry. Currently he is working as professor in ECE Department of Amrita Vishwa Vidyapeetham, Amritapuri, India. His areas of expertise are design for testability, FPGA based system design, VLSI signal processing. He can be contacted at email: ramesh.amrita@gmail.com.