iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** VI **Month of publication:** June 2024

DOI: https://doi.org/10.22214/ijraset.2024.63240

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue VI June 2024- Available at www.ijraset.com

Implementation of Programmable n-bit Pseudo Random Sequence Generator using 45 nm Technology

Mrs. S. Aruna[1], Ch. Soumya Pranathi[2], A. Siva Madhuri[3], K. Neha[4]
[1]*Assistant Professor,* [2,3,4]*Student, Department of Electronics and Communication Engineering, Maturi Venkata Subba Rao (MVSR) Engineering College, Osmania University, Telangana, India*

Abstract: ***In recent times with the advent of computers, programmers recognized the necessity for a means of introducing randomness into a computer program. Pseudo-random numbers generates necessary values for processes that require randomness, such as creating test signals or for synchronizing transmitting and receiving devices in a spread spectrum transmission. It is not possible to generate truly random numbers from deterministic thing like computers so PRSG is a technique developed to generate random numbers using a computer. Our proposed system is used to implement multiple polynomials of degree 2 to 16 in Incisive and Genus Cadence Tool. This system can be implemented in statistical analysis, and modern-day computer simulations, digital cryptography and generation of OTP.***

Keywords: Pseudorandom Sequence Generator (PRSG), One Time Password (OTP)

I. INTRODUCTION

Pseudorandom Sequence Generator (PRSG) refers to an algorithm that uses mathematical formulas to produce sequences of random numbers. PRSGs generate a sequence of numbers which will approximate the properties of random numbers. Many sequences are generated in a short time and can also be reproduced later, if the beginning point in the sequence is known. Therefore, the numbers are deterministic and efficient. Linear Feedback Shift Registers are shift registers (LFSR) with input bits that are a linear function of their previous state. An LFSR generates pseudo-random bit sequences that can be used in logical circuit testing. The LFSR is driven by XORing/XNORing multiple bits which is popularly termed as Feedback. The primitive polynomial selected determines the feedback path. In our method, XNOR gate is used in the feedback path for the MUX 8x1. The degree of the primitive polynomial is used to tell number of flip-flops used in this method. If the number of bits is given by n, the longest possible sequence length produced can be given by 2^n-1. Primitive polynomial refers to the polynomial that generates the longest possible sequence. Non-Primitive polynomials produce sequence of length should always be lesser than 2^n-1.

II. LITERATURE SURVEY

In the paper, [1] “Programmable Variable Length Pseudo Random Sequence Generator” published in the year 2022 have its features like producing variable length sequence with high speed operation of the circuit using XNOR gates.

Another reference paper cited, [2] “Design of Pseudo Random using Non-Linear Feedback Shift Register” published in the year 2022 has its features like low speed operation. The design in this paper does not support error checking and error correction.

Another reference paper titled, [4] “Low power Programmable Pseudo Random Word Generator” published in the year 2008 has a unique feature as, toggles can be set in advanced i.e., ‘pre-selected toggles’ and used as conventional pseudo random sequence generator.

Another reference paper titled, [3] “A 15-Gb/s 0.0037-mm^2 0.019-Pj/Bit Full-Rate Programmable Multi-Pattern Pseudo-Random Binary Sequence Generator” published in the year 2020, which is suitable for compact designs with high power and less efficient.

III. PROPOSED DESIGN

In this project, we consider polynomial of degree 2 to 16 which further includes one 8x1 Multiplexer, number of flip flops are 16 and 8 XNOR gates are connected as input for the MUX. As shown in the figure .The output of the MUX is given to the first D flip flop d_0 and for remaining flip flops linear shifting takes place. The randomness can be obtained from different polynomial and a particular pattern is selected using pattern selector.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue VI June 2024- Available at www.ijraset.com

IV. METHODOLOGY

The implementation of this project is done in Cadence tool, further synthesis is done in Genus and physical design is implemented with help of Innovus tool. The output sequence depends on number of bits 'n'. A 8x1 is used to vary the output sequences of linear function. Different feedback paths which are based on 8 different polynomials of degree 'n' are given as input to 8x1 MUX. The pseudo code for proposed method for the above figure n=5 is as shown in the Fig. 1.

Pseudo code
For n=5;
If Pattern Selector=0
Output =Sequence of Polynomial $x^5+x^3+x^2+x+1$
If Pattern Selector=1
Output =Sequence of Polynomial x^5+x^4+x+1
If Pattern Selector=2
Output =Sequence of Polynomial $x^5+x^4+x^2+1$
If Pattern Selector=3
Output =Sequence of Polynomial $x^5+x^3+x^2+1$
If Pattern Selector=4
Output =Sequence of Polynomial x^5+x+1
If Pattern Selector=5
Output =Sequence of Polynomial x^5+x^3+1
If Pattern Selector=6
Output =Sequence of Polynomial x^5+x^2+1
If Pattern Selector=7
Output =Sequence of Polynomial $x^5+x^4+x^3+x+1$

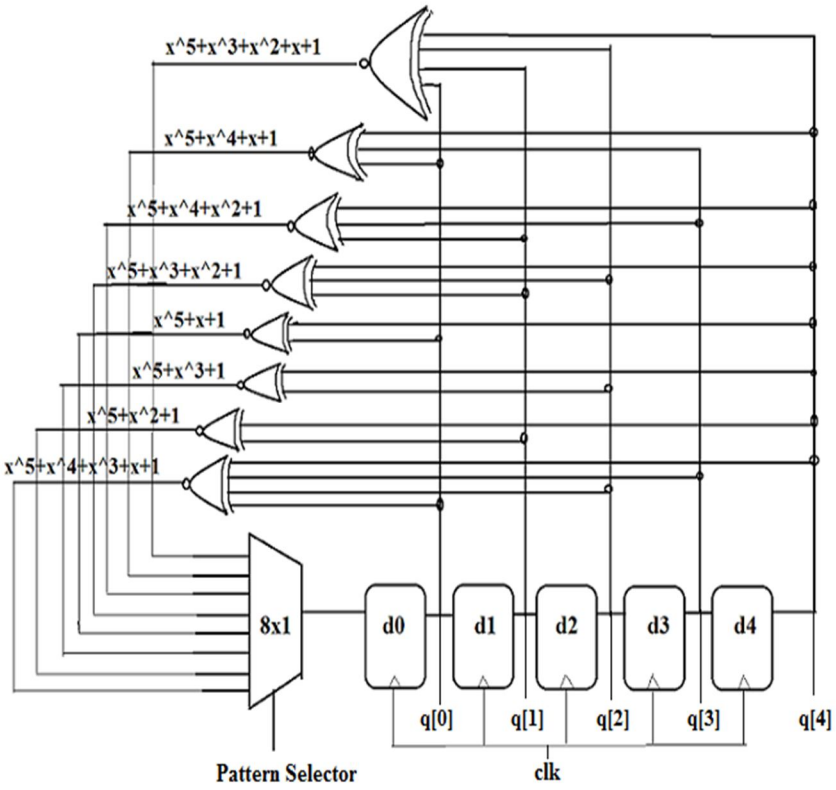


Fig. 1 Proposed Design for n=5 bit

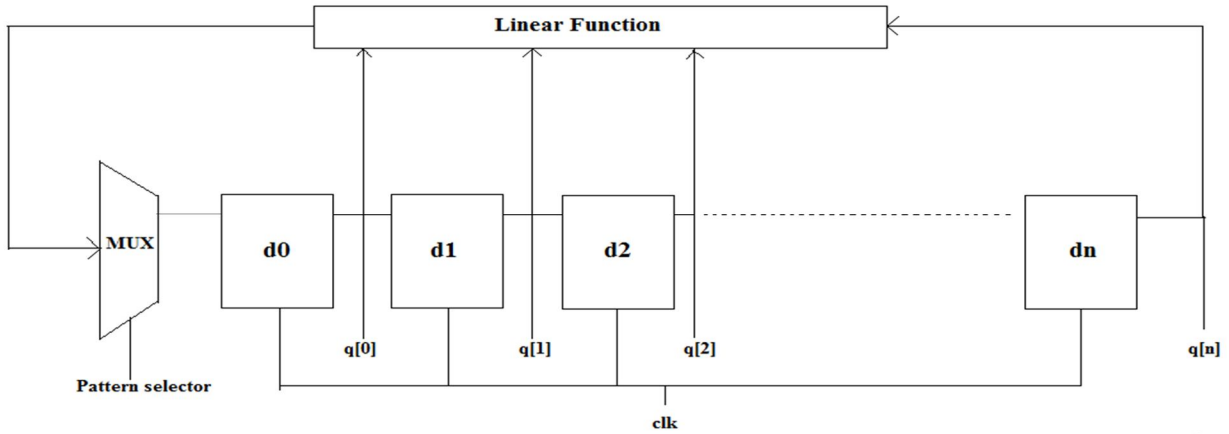


Fig. 2 Proposed Design for n bit

V. RESULT - 1

Table I. Polynomials Chosen And Output Sequence Length

Word Length (n)	Pattern Selector (s)	Polynomial Chosen	Output Sequence Length
2	0	x^2+x+1	3
	1	x^2+x+1	3
	2	x^2+x+1	3
	3	x^2+x+1	3
	4	x^2+x+1	3
	5	x^2+x+1	3
	6	x^2+x+1	3
	7	x^2+x+1	3
3	0	x^3+x+1	7
	1	x^3+x^2+1	7
	2	x^3+x^2+x+1	4
	3	x^3+x+1	7
	4	x^3+x^2+1	7
	5	x^3+x^2+x+1	4
	6	x^3+x^2+1	7
	7	x^3+x^2+x+1	4
4	0	x^4+x^2+x+1	14
	1	x^4+x^3+x+1	12
	2	x^4+x^3+1	15
	3	x^4+x^2+1	6
	4	x^4+x+1	15
	5	$x^4+x^3+x^2+1$	14
	6	x^4+x+1	15
	7	$x^4+x^3+x^2+x+1$	5
5	0	$x^5+x^3+x^2+x+1$	31
	1	x^5+x^4+x+1	8
	2	$x^5+x^4+x^2+1$	30
	3	$x^5+x^3+x^2+1$	12

	4	x^5+x+1	21
	5	x^5+x^3+1	31
	6	x^5+x^2+1	31
	7	$x^5+x^4+x^3+x+1$	31
6	0	$x^6+x^4+x^3+x+1$	63
	1	$x^6+x^5+x^2+1$	42
	2	$x^6+x^5+x^3+1$	28
	3	x^6+x^4+x+1	42
	4	x^6+x^2+1	14
	5	x^6+x^4+1	14
	6	x^6+x^3+1	9
	7	$x^6+x^4+x^2+x+1$	21
7	0	$x^7+x^5+x^4+x+1$	93
	1	$x^7+x^6+x^3+1$	124
	2	x^7+x^6+x+1	12
	3	$x^7+x^5+x^2+1$	20
	4	x^7+x^3+1	127
	5	x^7+x^5+1	93
	6	x^7+x^4+1	127
	7	$x^7+x^5+x^3+x+1$	127

Word Length (n)	Pattern Selector (s)	Polynomial Chosen	Output Sequence Length
8	0	$x^8+x^6+x^5+x+1$	255
	1	$x^8+x^7+x^4+1$	254
	2	$x^8+x^7+x^2+1$	254
	3	$x^8+x^6+x^3+1$	210
	4	x^8+x^4+1	12
	5	x^8+x^6+1	30
	6	x^8+x^5+1	217
	7	$x^8+x^6+x^4+x^2+1$	10
9	0	$x^9+x^7+x^6+x^2+1$	381
	1	x^9+x^8+x+1	16
	2	$x^9+x^8+x^3+1$	508
	3	$x^9+x^7+x^4+1$	372
	4	x^9+x^5+1	511
	5	x^9+x^7+1	465
	6	x^9+x^6+1	21
	7	$x^9+x^7+x^5+x^3+1$	105
10	0	$x^{10}+x^8+x^7+x^3+1$	105
	1	$x^{10}+x^9+x^2+1$	146
	2	$x^{10}+x^9+x^4+1$	210
	3	$x^{10}+x^8+x^5+1$	1022
	4	$x^{10}+x^6+1$	62
	5	$x^{10}+x^8+1$	42
	6	$x^{10}+x^7+1$	1023
	7	$x^{10}+x^8+x^6+x^4+1$	62

11	0	$x^{11}+x^{9}+x^{8}+x^{4}+1$	2047
	1	$x^{11}+x^{10}+x^{3}+1$	1860
	2	$x^{11}+x^{10}+x^{5}+1$	868
	3	$x^{11}+x^{9}+x^{6}+1$	204
	4	$x^{11}+x^{7}+1$	1533
	5	$x^{11}+x^{9}+1$	2047
	6	$x^{11}+x^{8}+1$	1953
	7	$x^{11}+x^{9}+x^{7}+x^{5}+1$	595
12	0	$x^{12}+x^{10}+x^{9}+x^{5}+1$	2667
	1	$x^{12}+x^{11}+x^{4}+1$	1302
	2	$x^{12}+x^{11}+x^{6}+1$	3810
	3	$x^{12}+x^{10}+x^{7}+1$	930
	4	$x^{12}+x^{8}+1$	28
	5	$x^{12}+x^{10}+1$	126
	6	$x^{12}+x^{9}+1$	45
	7	$x^{12}+x^{10}+x^{8}+x^{6}+1$	30
13	0	$x^{13}+x^{11}+x^{10}+x^{6}+1$	8191
	1	$x^{13}+x^{12}+x^{5}+1$	4088
	2	$x^{13}+x^{12}+x^{7}+1$	1524
	3	$x^{13}+x^{11}+x^{8}+1$	2380
	4	$x^{13}+x^{9}+1$	7161
	5	$x^{13}+x^{11}+1$	1785
	6	$x^{13}+x^{10}+1$	8001
	7	$x^{13}+x^{11}+x^{9}+x^{7}+1$	2555
14	0	$x^{14}+x^{12}+x^{11}+x^{7}+1$	14329
	1	$x^{14}+x^{13}+x^{6}+1$	1778
	2	$x^{14}+x^{13}+x^{8}+1$	2190
	3	$x^{14}+x^{12}+x^{9}+1$	16382
	4	$x^{14}+x^{10}+1$	186
	5	$x^{14}+x^{12}+1$	254
	6	$x^{14}+x^{11}+1$	5115
	7	$x^{14}+x^{12}+x^{10}+x^{8}+1$	254

Word Length (n)	Pattern Selector (s)	Polynomial Chosen	Output Sequence Length
15	0	$x^{15}+x^{13}+x^{12}+x^{8}+1$	24573
	1	$x^{15}+x^{14}+x^{7}+1$	10540
	2	$x^{15}+x^{14}+x^{9}+1$	1020
	3	$x^{15}+x^{13}+x^{10}+1$	10668
	4	$x^{15}+x^{11}+1$	32767
	5	$x^{15}+x^{13}+1$	4599
	6	$x^{15}+x^{12}+1$	63
	7	$x^{15}+x^{13}+x^{11}+x^{9}+1$	7905

16	0	$x^{16}+x^{14}+x^{13}+x^{9}+1$	13107
	1	$x^{16}+x^{15}+x^{8}+1$	9362
	2	$x^{16}+x^{15}+x^{10}+1$	49146
	3	$x^{16}+x^{14}+x^{11}+1$	2045
	4	$x^{16}+x^{12}+1$	60
	5	$x^{16}+x^{14}+1$	126
	6	$x^{16}+x^{13}+1$	57337
	7	$x^{16}+x^{14}+x^{12}+x^{10}+1$	434

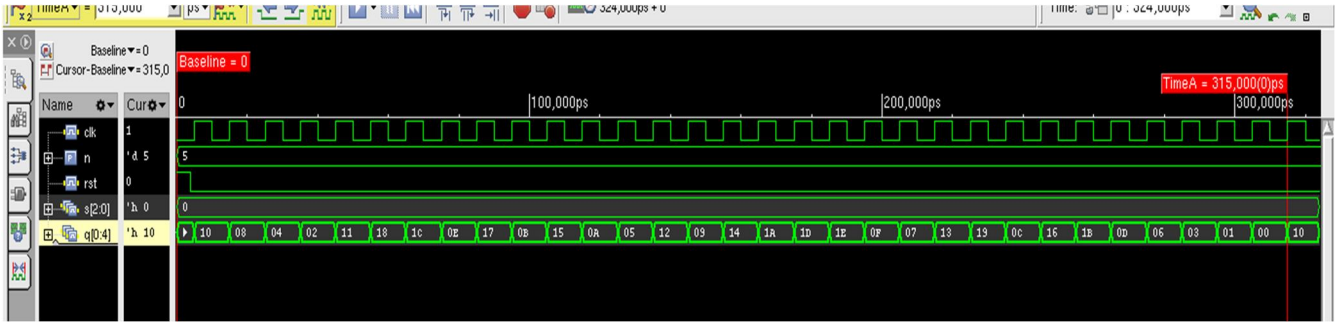


Fig. 3 Output Sequence for s=0, n=5

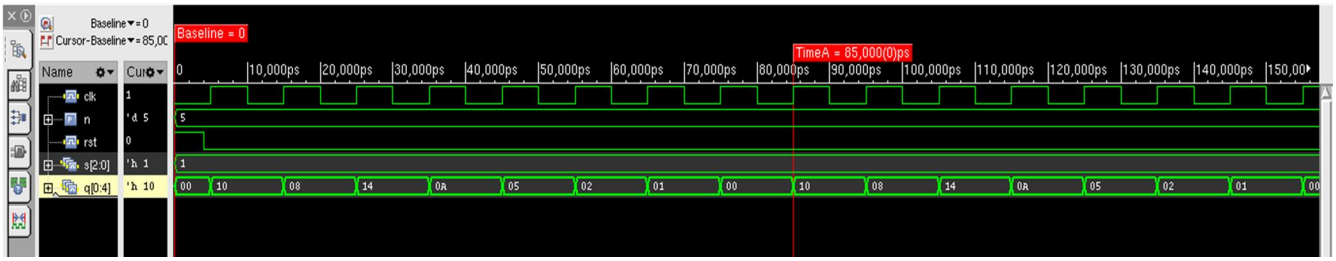


Fig. 4 Output Sequence for s=1, n=5

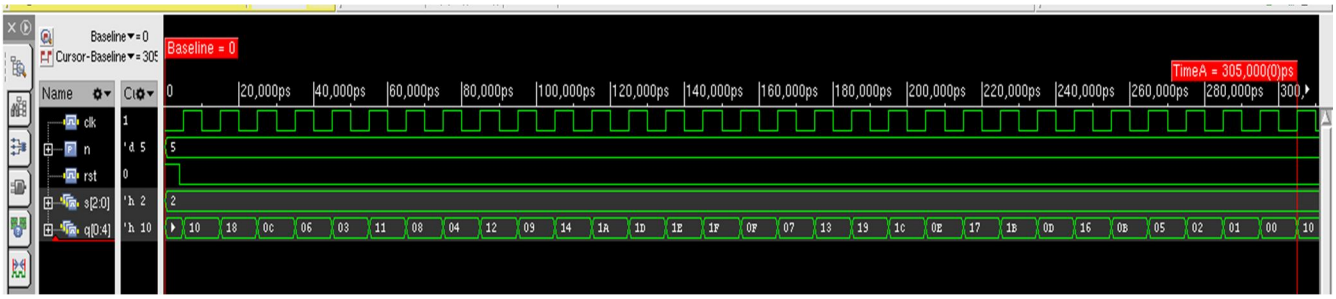


Fig. 5 Output Sequence for s=2, n=5

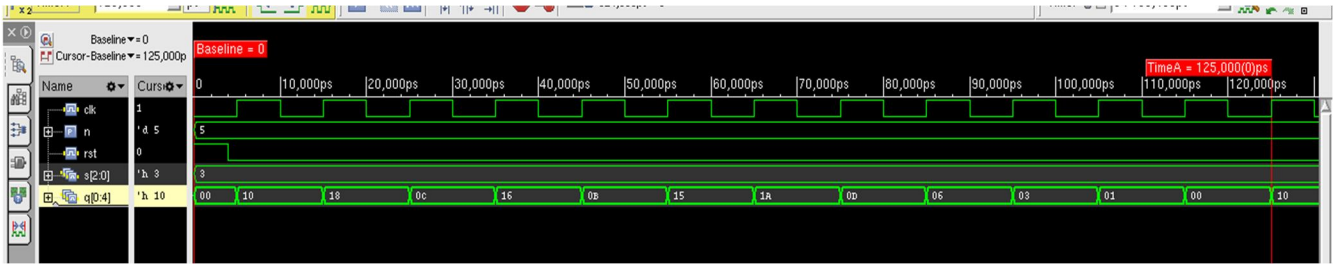


Fig. 6 Output Sequence for s=3, n=5

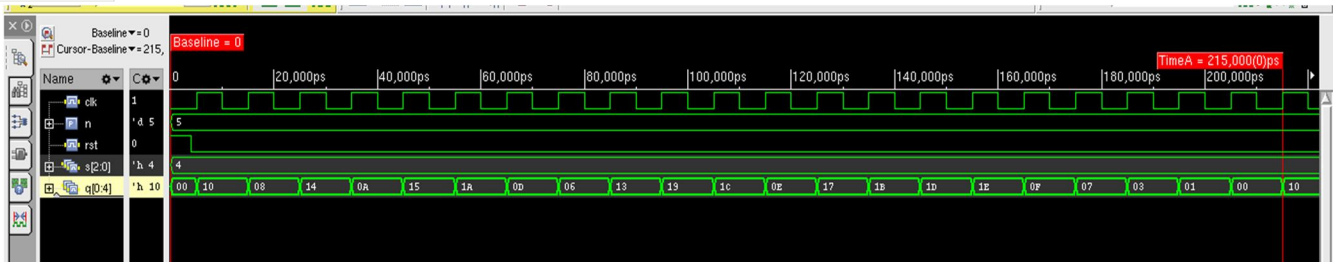


Fig. 7 Output Sequence for s=4, n=5

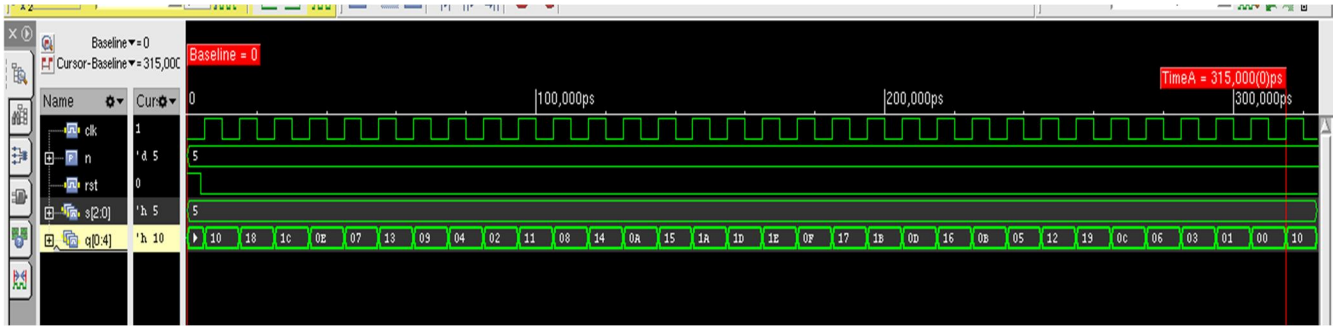


Fig. 8 Output Sequence for s=5, n=5

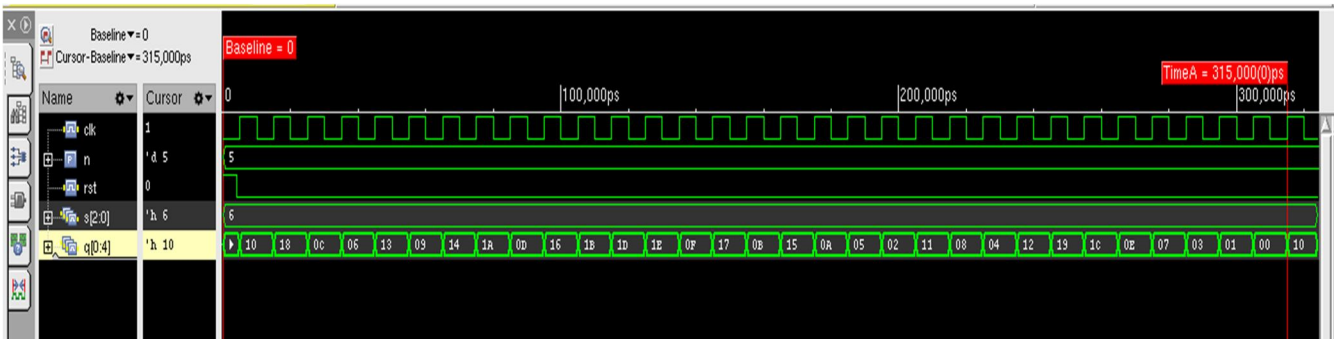


Fig. 9 Output Sequence for s=6, n=5

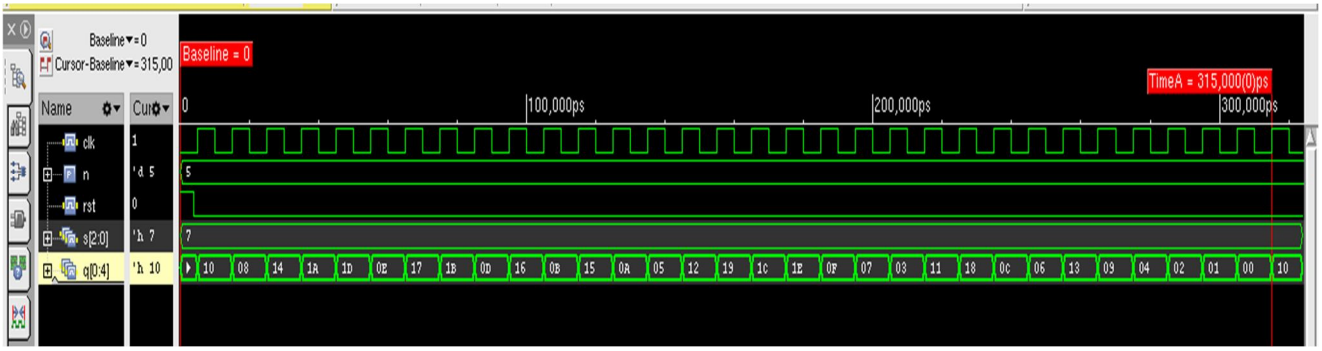


Fig. 10 Output Sequence for s=7, n=5

Considering, the word length n =5, the design implements the 2 –bit output of 5 different polynomials mentioned in Table.III. Fig. 3 Displays the output sequences for s=0, Fig. 4 Displays the output sequences for s=1, Fig. 5Displays the output sequences for s=2, Fig. 6 Displays the output sequences for s=3, Fig. 7 Displays the output sequences for s=4, Fig. 8 Displays the output sequences for s=5, Fig. 9 Displays the output sequences for s=6, Fig. 10 Displays the output sequences for s=7.

Table II. Sequence Generator Based On The Pattern Selector for n=5

Pattern Selector (s)	Output Sequence Length	Polynomial Chosen	Required Output Patterns
0	31	$x^5+x^3+x^2+x+1$	00,10,08,04,02,11,18,1C,0E,17,0B,15,0A,05,12,09, 14,1A,1D,1E,0F,07,13,19,0C,16,1B,0D,06,03,01
1	8	x^5+x^4+x+1	00,10,08,14,0A,05,02,01
2	30	$x^5+x^4+x^2+1$	00,10,18,0C,06,03,11,08,04,12,09,14,1A,1D,1E, 1F,0F,07,13,19,1C,0E,17,1B,0D,16,0B,05,02,01
3	12	$x^5+x^3+x^2+1$	00,10,18,0C,16,0B,15,1A,0D,06,03,01
4	21	x^5+x+1	00,10,08,14,0A,15,1A,0D,06,13,19, 1C,0E,17,1B,1D,1E,0F,07,03,01
5	31	x^5+x^3+1	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
6	31	x^5+x^2+1	00,10,18,0C,06,13,09,14,1A,0D,16,1B,1D,1E,0F,17, 0B,15,0A,05,02,11,08,04,12,19,1C,0E,07,03,01
7	31	$x^5+x^4+x^3+x+1$	00,10,08,14,1A,1D,0E,17,1B,0D,16,0B,15,0A,05,12, 19,1C,1E,0F,07,03,11,18,0C,06,13,09,04,02,01

VI. BACKEND

A. *Logic Synthesis*

1) It is the main part of the VLSI design flow which shows the RTL code in Schematic.

2) To obtain a Schematic diagram from RTL code, it requires certain files like: **.v file, .lib file, .sdc file.**

3) The Outputs of the Synthesis are Gate Level Netlist, Timing report, Area report, Power report.

B. *Physical Design*

The Physical Design of Pseudo Random Sequence Generator (PRSG) is implemented using Cadence Innovus tools.

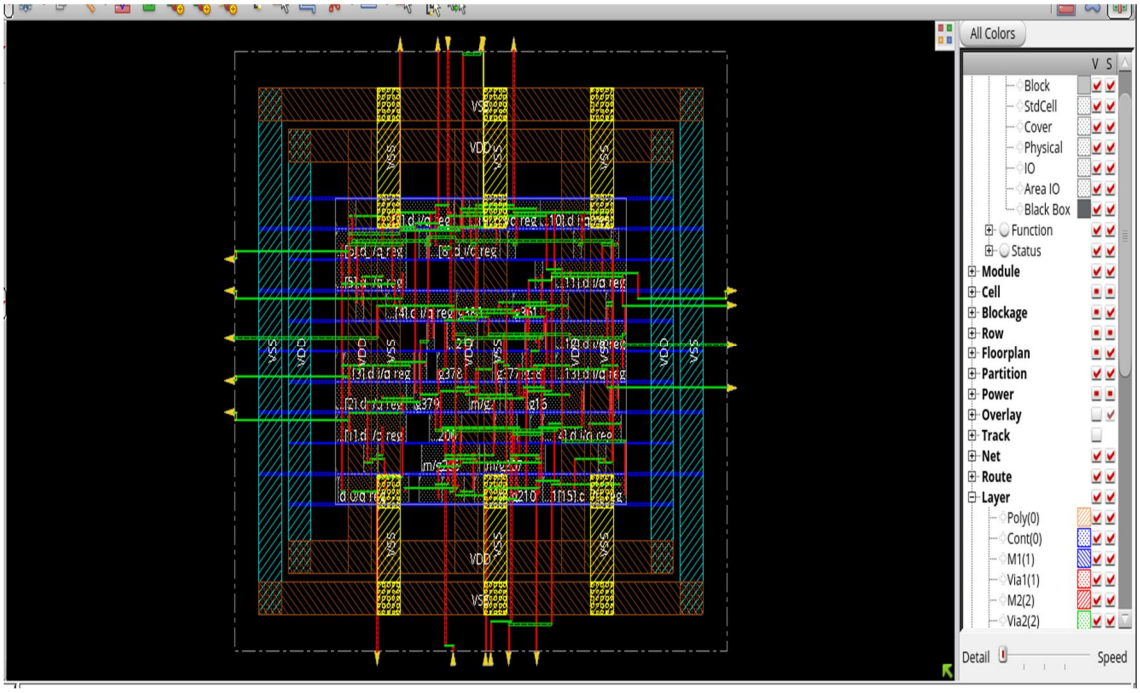


Fig. 11 Physical Design of 16-bit Pseudo Random Sequence Generator

VII. RESULTS – 2

TABLE III. COMPARISON OF PARAMETERS WITH EXISTING METHOD

PARAMETERS	EXISTING METHOD	PROPOSED METHOD
Title of the Project	A 15-Gb/s 0.0037-mm^2 0.019-Pj/Bit Full-Rate Programmable Multi-Pattern Pseudo-Random Binary Sequence Generator	Implementation of Programmable n-bit Pseudo Random Sequence Generator using 45 nm technology
Technology used	45nm	45nm
Supply Voltage	1.1 V	1.08 V
Static Power	8.778 mW	0.1096 mW

TABLE IV. OBTAINED PARAMETERS IN PROPOSED METHOD

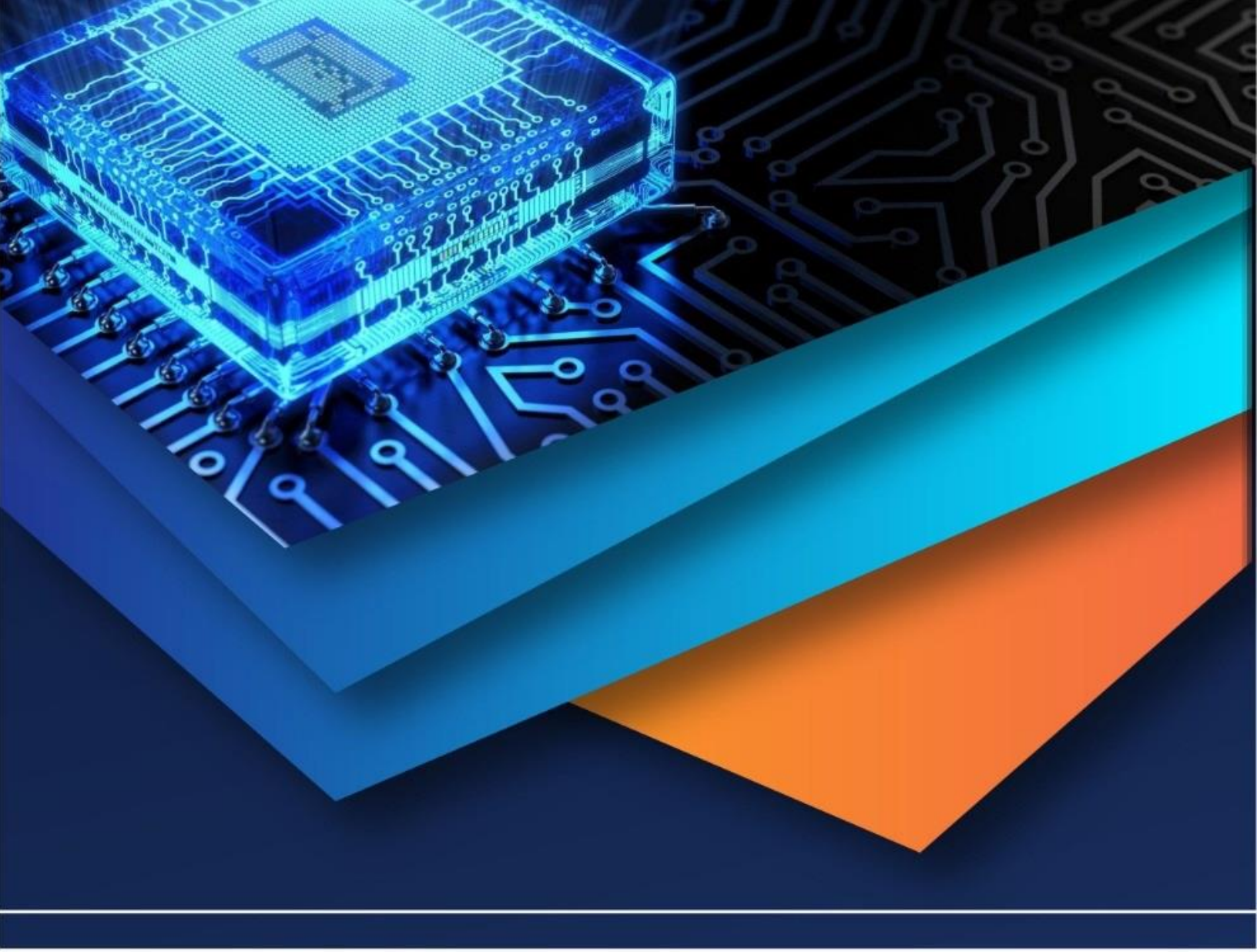
PARAMETERS	PROPOSED METHOD
Title of the Project	Implementation of Programmable n-bit Pseudo Random Sequence Generator using 45 nm technology
Technology used	45nm
Number of bits used i.e.,n	16 bits
Supply Voltage	1.08 V
Total Area occupied	174.078 um^2
Total Power	0.1096 mW

VIII. CONCLUSION AND FUTURE SCOPE

The proposed method is able to generate Variable-length Pseudo Random Sequences according to the user's input. As the previous methods of generating a pseudo random sequences are less efficient and dissipate high power as shown in the TABLE IV. This project is implemented with low power and highly efficient pseudo random sequences. The pseudo-random sequence generators are marked by evolution in security, hardware implementations, application-specific, Equalized, interdisciplinary collaboration, and ethical considerations. By Communicating with these challenges and opportunities, PRSG considered to progress as tools for establishing randomness, security, and reliability in diverse computing applications. PRSGs play a crucial role in decision-making processes in diverse sectors, addressing concerns related to bias, fairness, and explicit. Future developments should prioritize ethical principles to be responsible and standardized use of randomness in computational systems. Some of the well known applications are Hardware-based PRSGs, grasping determined hardware components like field-programmable gate arrays (FPGAs) or application-specific integrated circuits (ASICs), provide better performance and efficiency compared to software-based approaches. Future developments primarily focus on optimizing hardware architectures for specific implementations and objectives.

REFERENCES

[1] D. Mohammad, K. Rakesh, G. Remadevi Somanathan and R. Bhakthavatchalu, "Programmable Variable-Length Pseudo-Random Sequence Generator," *2022 Second International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT)*, Bhilai, India, 2022, pp. 1-4, doi: 10.1109/ICAECT54875.2022.9808067.
[2] K. Vooke, N. K. Toramamidi, K. K. Thodeti and S. Singh, "Design of Pseudo-Random Number Generator Using Non-Linear Feedback Shift Register," 2022 First International Conference on Electrical, Electronics, Information and Communication Technologies (ICEEICT), Trichy, India, 2022, pp. 1-5, doi: 10.1109/ICEEICT53079.2022.9768456.
[3] J.Hu,Z.ZhangandQ.Pan,"A15-Gb/s0.0037-mm²0.019-pJ/Bit Full-Rate Programmable Multi-Pattern Pseudo-Random Binary Sequence Generator," in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 67, no. 9, pp. 1499-1503, Sept. 2020, doi:10.1109/TCSII.2020.3008567.
[4] Wei-Z Chen and Guan-Sheng Huang, "Low-Power Programmable Pseudorandom Word Generator and Clock Multiplier Unit for High-Speed SerDes Applications," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 55, no. 6, pp. 1495-1501, July 2008,doi:10.1109/TCSI.2008.916507.
[5] R.S.Durga, C.K.Rashmika, O.N.V.Madhumitha, D.G.Suvetha, B.Tanmai and N. Mohankumar, "Design and Synthesis of LFSR based Random Number Generator,"2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT), 2020, pp. 438-442, doi: 10.1109/ICSSIT48917.2020.9214240.
[6] https://youtu.be/thmnJXfoIYA?si=Z53U8fQnkpQqBnzV
[7] https://youtu.be/QbTedKVxdyU?si=weWAFR8f4NeYwI14



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY