2021 IEEE International Symposium on Smart Electronic Systems (iSES) | 978-1-7281-8753-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/iSES52644.2021.00032

Design and FPGA Implementation of High-Speed Area and Power Efficient 64-bit Modified Dual CLCG based Pseudo Random Bit Generator

Krishna Sai Tarun Ramapragada
B.Tech, Department of ECE
National Institute of Technology Tiruchirappalli
Tiruchirappalli, India - 620015
tarunkousik@gmail.com

Ajith Kumar Reddy Netla
B.Tech, Department of ECE
National Institute of Technology Tiruchirappalli
Tiruchirappalli, India - 620015
ajithnetla4@gmail.com

Pavan Kalyan Chattada
B.Tech, Department of ECE
National Institute of Technology Tiruchirappalli
Tiruchirappalli, India - 620015
cpavankalyan37@gmail.com

Bhaskar Manickam
Professor, Department of ECE
National Institute of Technology Tiruchirappalli
Tiruchirappalli, India - 620015
bhaskar@nitt.edu

***Abstract*—In Internet of Things (IOT) based applications, security is given the highest importance and cryptography plays a major role in maintaining the data safety. Encryption and decryption in cryptography requires a Pseudo random bit generator (PRBG) for key generation. Modified dual coupled linear congruential generator (MDCLCG) based PRBG is the highly efficient PRBG algorithm as it clears all 15 National Institute of Standards and Technology (NIST) tests and has maximum period of 2^n for a n-bit design. In this paper, complete design and testing of a 64-bit MDCLCG based PRBG is proposed and it's implementation on Kintex-7 XC7K160TFBG676-3 field-programmable gate array (FPGA) is presented. Main components of MDCLCG based PRBG includes a Three operand adder, Barrel shifter, Comparator and an Encoder. Further, a High speed area efficient three operand adder (HSAEA) is used to improve performance of the proposed 64-bit MDCLCG architecture. It's performance is compared with 64-bit MDCLCG designed using three operand Ultra fast adder (UFA) and three operand Carry save adder (CSA) architectures. The post-implementation results of the proposed 64-bit MDCLCG are carried out and from the analysis, it is reported that the proposed 64-bit MDCLCG designed using HSAEA has 25.1%, 9.2% reduction in Area/Maximum frequency (A/F_{Max}) value when compared to UFA and CSA based 64-bit MDCLCG architectures respectively. Also, it has 17.7%, 4.4% reduction in Power/Maximum frequency (P/F_{Max}) value over UFA and CSA based 64-bit MDCLCG architectures respectively. Moreover, the proposed 64-bit MDCLCG ensures more security than 32-bit MDCLCG proposed in literature as the pseudo random bit sequence has a period of 2^{64} bits instead of 2^{32} bits.**

***Index Terms*—Internet of Things, Cryptography, High speed area efficient three operand adder, Modified dual coupled linear congruential generator, FPGA**

I. Introduction

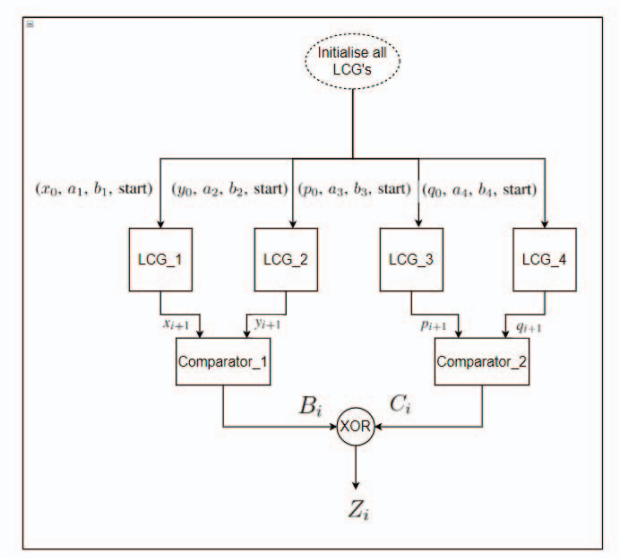
In recent times, Internet of Things (IOT) has become ubiquitous in every domain. IOT refers to many devices connected to internet across the world, all gathering and sharing data [1]–[3]. Besides having many advantages like better speed of operation, easy access; IOT has some sensitive measures pertaining to data safety and security [4], [5]. Therefore, cryptography plays a vital role in protecting data [6]–[8]. In cryptography, plain text is converted to cipher text using encryption at transmitting end and vice-versa using decryption at receiving end to avoid any data breach during transmission.

In Stream cipher based applications, encryption is done bit by bit and for each bit at transmission side, one Pseudo random bit for every clock cycle is required [8]. Modified dual coupled linear congruential generator (MDCLCG) is a highly secure Pseudo random bit generator (PRBG) as it generates maximum length of 2^n pseudo random bits for n-bit design and it also satisfies all the fifteen National Institute of Standards and Technology (NIST) tests [9]. So, it can be considered as a better PRBG compared to other methods like Linear congruential generator (LCG), Coupled LCG (CLCG) which do not satisfy all the fifteen NIST tests [9]–[12]. Also, as MDCLCG produces Pseudo random bit at each clock cycle [9], it can be used in stream cipher encryption and decryption for improving security in all IOT and other data security applications.

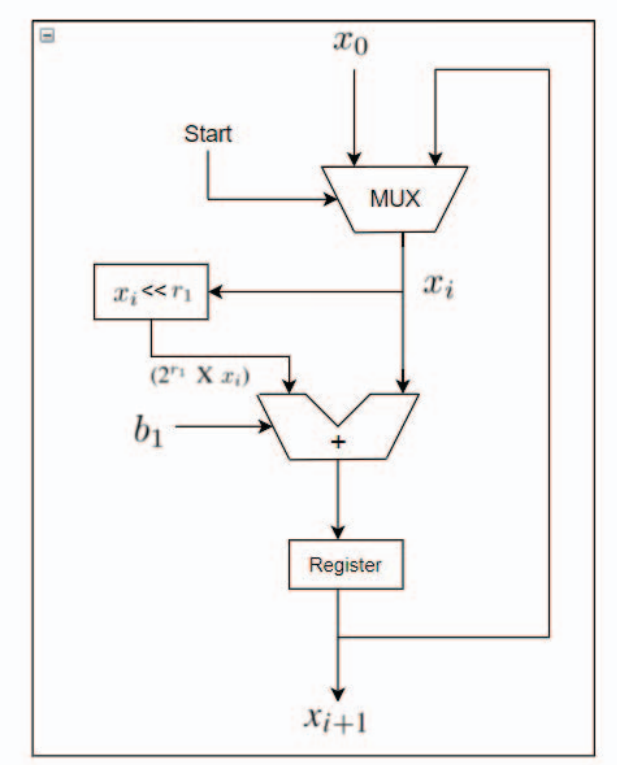
MDCLCG architecture has four LCGs, two comparators and an XOR gate as shown in Fig. 1a [9]. Also, each LCG has a Three operand adder, Barrel shifter, Encoder, Multiplexer, and a Register as shown in Fig. 1b [9]. For better performance of MDCLCG based PRBG, it becomes dire need to design and implement delay, area and power efficient three operand adder, Barrel shifter, Comparator and an Encoder.

A 32-bit MDCLCG based PRBG architecture in [9] is designed using three operand Carry save adder (CSA), High

978-1-7281-8753-2/21/$31.00 ©2021 IEEE
DOI 10.1109/iSES52644.2021.00032



(a) Architecture of Modified Dual CLCG Method



(b) Architecture of LCG_1

Fig. 1: Architecture of Modified Dual CLCG Method and LCG

Speed Area Efficient three operand Adder (HSAEA), Han Carlson adder (HCA) and Hybrid Han Carlson adder (HHCA). However, the 32-bit MDCLCG based PRBG generates random sequence with a period of 2^{32} bits.

In this paper, to enhance the security, 64-bit MDCLCG based PRBG is proposed, the architecture is designed and tested using Verilog Hardware description language (HDL) and implemented on Kintex-7 XC7K160TFBG676-3 field-programmable gate array (FPGA) board using High speed area efficient three operand adder (HSAEA). Also, it's results are compared with 64-bit MDCLCG designed using three operand Ultra fast adder (UFA) and three operand Carry save adder (CSA). In this paper, the other modules such as Barrel Shifter, Comparator are also proposed for 64-bit operand size and are used for the design of 64-bit MDCLCG based PRBG. In the proposed 64-bit MDCLCG, the pseudo random bit sequence has a period of 2^{64} bits.

The paper is organised as follows. Section II discusses the design of each block used in 64-bit MDCLCG based PRBG. Section III presents testing of designed 64 bit MDCLCG based PRBG including randomness testing. Section IV shows the comparison results obtained for various 8-, 16-, 32-, 64-bit adder architectures. Also, the section includes comparison results for the proposed 64-bit MDCLCG based PRBG for various adder architectures. Section V concludes the paper.

II. DESIGN OF 64-BIT MDCLCG

MDCLCG consists of four LCG units and each LCG requires a seed and two other inputs [9]. As shown in Fig. 1b, LCG_1 requires the seed x_0 and other inputs a_1, b_1 besides the start signal. The input a_1 must be of the form $2^{r_1} + 1$. Then, each LCG will generate the sequence in which each new term depends on previous term as shown in (1) and (2) [9].

$$x_{i+1} = (a_1 \text{ x } x_i + b_1) mod 2^n \quad (1)$$

$$x_{i+1} = (2^{r_1} \text{ x } x_i + x_i + b_1) mod 2^n \quad (2)$$

x_1, x_2, x_3,..... are generated as per (2). Similarly, the four LCG units would generate four sequences X_i, Y_i, P_i, Q_i as per (1) and (2). Then, the sequences B_i and C_i in the MDCLCG architecture are generated by comparing X_i with Y_i and P_i with Q_i respectively using the magnitude comparators as shown in Fig. 1a. Finally, The pseudo random bit sequence Z_i is obtained by XOR operation of B_i and C_i ($B_i \oplus C_i$).

There are various blocks involved in the design of MDCLCG and the hardware implementation of every block used in the design of 64-bit MDCLCG is explained in the following subsections.

A. Design of 64-bit Three operand adder

An Efficient implementation of three operand adder has to be designed as it is the main component in LCG as shown in

Fig. 1b. In the design of the proposed 64-bit MDCLCG, a 64-bit High speed area efficient three operand adder (HSAEA) is used and it's performance is compared with 64-bit MDCLCG architectures designed using 64-bit three operand Ultra fast adder (UFA) and 64-bit three operand Carry save adder (CSA).

1) Design of 64-bit High speed area efficient three operand adder: Basic Architecture of High speed area efficient three operand adder (HSAEA) is given in [13]. In the design of 64-bit MDCLCG, HSAEA is used considering the architecture given in [13] and extended it to 64-bit operand size.

2) Design of 64-bit three operand Ultra fast adder: Architecture of two operand Ultra fast adder (UFA) is given in [14]. It is extended for three operand addition using Bit addition logic mentioned in [13]. Also, the design for 8-bit adder in [14] is extended to 64-bit adder. Thus, by employing these two modifications, 64-bit three operand Ultra fast adder (UFA) is designed and it is used in the design of 64-bit MDCLCG.

3) Design of 64-bit three operand Carry save adder: Architecture of three operand Carry Save adder (CSA) is given in [13]. It is extended to 64-bit operand size and used in the design of 64-bit MDCLCG.

B. Design of 64-bit Barrel shifter

In each LCG, multiplication with 2^r is required as shown in Fig. 1b and that is implemented effectively in hardware using a left shift Barrel shifter. The design of Barrel shifter for 64-bit number is shown in Fig. 2. Barrel shifter involved in the design of 64-bit MDCLCG architecture finds 0 to 63 left shift values of the input and select the required output using (64 X 1) - 64 line mux based on the shift value.

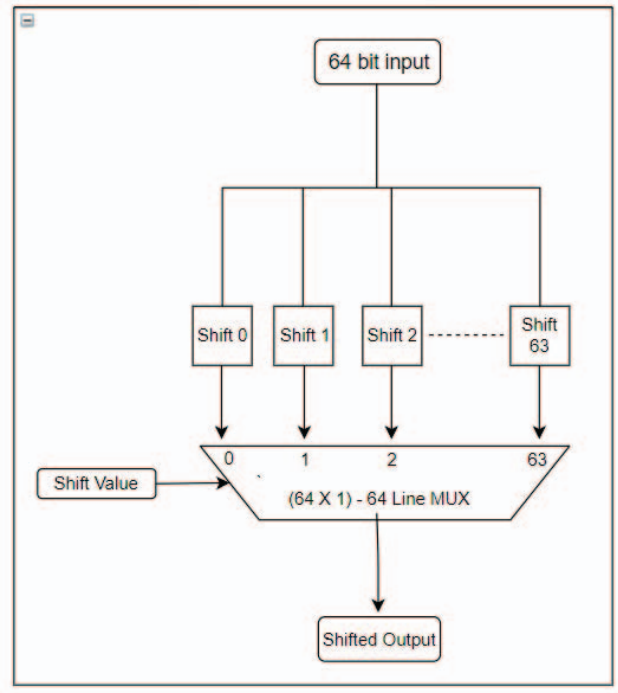


Fig. 2: Design of Barrel Shifter for 64-bit number

C. Design of 64-bit Comparator

Two 64-bit comparators are required to design a 64-bit MDCLCG as shown in Fig. 1a. Instead of using conventional comparator, an 'Adder based comparator' is designed. A > B signal is a Carry out signal of addition of A and one's complement of B.

Consider A and B are two r-bit numbers. Every bit of B is inverted to perform subtraction through one's complement addition. One's complement of B will be equal to $2^r - 1 - B$. When one's complement of B is added to A, result will be A − B + $2^r - 1$. Note that $2^r - 1$ represents the maximum unsigned decimal quantity that can be represented with r-bit binary number. Hence, if A is greater than B, (A − B) > 0 and so the result will be greater than $2^r - 1$ thus resulting in carry out of 1. Similarly, if A is less than or equal to B then carry out will be 0. Thus, the comparator output is the carry out of addition of A and one's complement of B. A portion of 64-bit HSAEA is used for the carry out generation.

D. Design of 64 X 6 Encoder

Input 'a' is constrained to be in the form of $2^r + 1$ (which means apart from Least Significnat Bit (LSB), only 1 bit of 'a' will be active) and this value of r has to be passed to the barrel shifter as shown in Fig. 1b. This is done using a 64 X 6 Encoder. If the LSB of the (64 X 6) encoder's input is fixed as 0 and connecting rest of the bits with a[63:1], encoder returns position of the bit which is active in (a − 1) and this value ($\log_2(a-1) =$ r) is given as shift value to the barrel shifter. In the proposed design of MDCLCG, conventional encoder design is used [15], [16].

III. TESTING OF 64-BIT MDCLCG

A. 64-bit three operand adder testing

To test the adder architecture, $(2^{64})^3$ test vectors are required as there are 3 inputs of 64 bits each. It's not possible to test these many input vectors. So, partition technique is employed to perform testing [17]. By partitioning, first stage is separated from the architecture which has 64 full adders stacked in parallel and other partition will be a two operand 64-bit adder.

For a full adder, a total of 8 input combinations are possible. As the first stage is 64 full adders stacked in parallel, it can be tested using 8 test vectors as each full adder is independent of each other.

The other partition is a 64-bit two operand adder, it is further partitioned into sixteen 4-bit two operand adders. Incoming carry to a particular 4-bit adder partition can be changed by choosing the carry generation from the preceding bits as 0 or 1. So it requires 8192 (2^4 x 2^4 x 16 x 2) test vectors to test this second partition.

B. 64-bit Barrel shifter testing

Left Shift of 64-bit number by any number between 0 and 63 is to be done. No of possible combinations of such 64 bit numbers is 2^{64}. It is a tedious task to test all 64 left shift cases of all those numbers. So $64'h$ FFFF FFFF FFFF FFFF and

$64'h$ 0000 0000 0000 0000 are chosen as inputs and verified for all 64 left shift cases. Correct outputs for these inputs automatically implies that the design is correct for all other combinations too.

C. 64-bit Comparator testing

The comparator which is proposed in this paper in Section II-C takes two inputs of 64 bits. So a total of $(2^{64})^2$ input vectors are possible. It is not possible to carryout testing for these many input vectors. So, partition technique is employed to perform testing which is followed in second partiton of adder testing. So, comparator is divided into 16 partitions of 4-bit each and testing is carried out on these partitions through 8192 (2^4 x 2^4 x 16 x 2) test vectors.

D. 64 X 6 Encoder testing

For a 64 X 6 Encoder a total of 64 input combinations are possible. If input is $64'h$ 0000 0000 0000 0001, then encoded output will be $6'b$ 0. Similarly if input is $64'h$ 8000 0000 0000 0000, then the encoded output will be $6'b$ 111111. Encoded output is verified for all the 64 combinations of inputs thus the functioning of 64 X 6 Encoder is verified.

E. Randomness testing

1) NIST testing: 10 bit streams of 10^6 bits are generated through behavioural simulation of MDCLCG in Xilinx Vivado 2020.2 and NIST randomness test is carried out on Ubuntu machine using NIST Test Suite [18]. An analytical routine has been included in the Test Suite to facilitate the interpretation of results. A file named "finalAnalysisReport" is auto-generated when statistical testing gets completed. The report contains the summary of all empirical results as shown in Fig. 3.

In Fig. 3, the number of rows corresponds to the number of statistical tests applied. The number of columns, are distributed as follows: columns 1-10 correspond to the frequency of P-values, column 11 is the P-value that arises through the application of a chi-square test 11, column 12 is the proportion of binary sequences that passed, and the column 13 is the corresponding statistical test. The data sample obtained from the simulation of the proposed 64-bit MDCLCG cleared critical values for all 15 tests which can be observed in Fig. 3.

2) Auto-correlation testing: A data sample is considered as random if it doesn't hold any patterns in it i.e no similarities should exist throughout the data. Auto-correlation measures the degree of similarity between a sample and delayed version of itself. Auto-correlation value should be almost 0 for non-zero delay in order to consider the data sample as a random data.

Auto-correlation is plotted in MATLAB for a data set of length 10^5 bits generated from MDCLCG simulation in Xilix Vivado 2020.2 and it resembled impulse function i.e the auto-correlation value is almost 0 for non-zero delay. The Auto-correlation plot is shown in Fig. 4.

RESULTS FOR THE UNIFORMITY OF P-VALUES AND THE PROPORTION OF PASSING SEQUENCES

generator is <data/data3.txt>

C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	P-VALUE	PROPORTION	STATISTICAL TEST
3	0	5	0	2	0	0	0	0	0	0.000954	9/10	Frequency
0	1	3	1	1	2	1	0	1	0	0.534146	10/10	BlockFrequency
4	0	4	1	0	0	1	0	0	0	0.004301	9/10	CumulativeSums
0	0	2	2	0	3	0	1	2	0	0.213309	10/10	Runs
1	4	0	1	1	0	1	0	2	0	0.122325	10/10	LongestRun
1	0	1	0	2	2	1	2	1	0	0.739918	10/10	Rank
0	2	1	3	1	2	0	0	0	1	0.350485	10/10	FFT
2	1	1	0	1	1	0	2	0	2	0.739918	9/10	NonOverlappingTemplate
2	0	2	1	0	1	2	0	1	1	0.739918	9/10	OverlappingTemplate
0	0	0	3	1	2	0	2	2	0	0.213309	10/10	Universal
1	1	0	2	2	1	2	0	1	0	0.739918	9/10	ApproximateEntropy
0	0	0	0	1	1	0	0	2	0	----	4/4	RandomExcursions
0	0	1	1	0	0	0	1	0	1	----	4/4	RandomExcursionsVariant
1	0	1	2	0	0	2	2	0	2	0.534146	10/10	Serial
1	2	0	2	0	1	0	0	2	2	0.534146	9/10	LinearComplexity

The minimum pass rate for each statistical test with the exception of the random excursion (variant) test is approximately = 8 for a sample size = 10 binary sequences.

The minimum pass rate for the random excursion (variant) test is approximately = 3 for a sample size = 4 binary sequences.

For further guidelines construct a probability table using the MAPLE program provided in the addendum section of the documentation.

Fig. 3: Report generated from NIST Test Suite

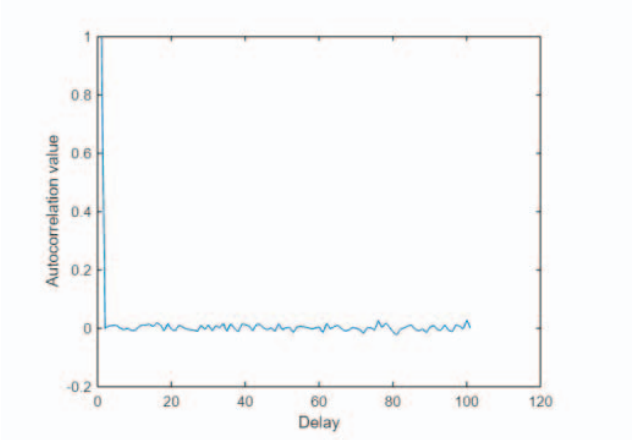


Fig. 4: Auto-correlation plot

IV. Results and Discussion

A. Results for various adder architectures

8-bit, 16-bit, 32-bit and 64-bit HSAEA, UFA and CSA architectures are successfully synthesized and implemented in Xilinx Vivado 2020.2 using Verilog HDL with target device being Kintex-7 XC7K160TFBG676-3 FPGA board. For comparison among various architectures, in this paper, Look-up Table (LUT) Utililization is considered as metric for Area consumption. Post-Implementation Area, Delay and Power metrics of the three adder architectures for various bit operand size are presented in TABLE 1a, TABLE 1b, TABLE 1c respectively.

For better comparison of various adder architectures in terms of all three metrics (Area, Delay and Power) combined together, Area-Delay-Power-Product (A x D x P) parameter is considered. (A x D x P) values for HSAEA, UFA, CSA for various bit operand sizes are presented in TABLE 1d. It is

TABLE 1: Comparison of various metrics for three different adder architectures

(a) Area Utilisation report for various adders

Architecture	LUT Utilization(A)			
	8-bit	*16-bit*	*32-bit*	*64-bit*
HSAEA	0.03%	0.07%	0.18%	0.41%
UFA	0.04%	0.10%	0.23%	0.52%
CSA	0.03%	0.06%	0.13%	0.27%

(b) Delay report for various adders

Architecture	Logic Delay(D)(ns)			
	8-bit	*16-bit*	*32-bit*	*64-bit*
HSAEA	0.258	0.430	0.559	0.602
UFA	0.251	0.301	0.473	0.473
CSA	0.387	0.731	1.419	1.419

(c) Power report for various adders

Architecture	Power(Watts)(P)			
	8-bit	*16-bit*	*32-bit*	*64-bit*
HSAEA	8.666	16.819	34.304	76.18
UFA	8.542	17.178	37.719	80.131
CSA	12.826	25.303	55.22	73.176

(d) Area-Delay-Power Product report for various adders

Architecture	(A x D x P)			
	8-bit	*16-bit*	*32-bit*	*64-bit*
HSAEA	0.06707	0.50625	3.45166	18.80274
UFA	0.08576	0.51705	4.10345	19.70902
CSA	0.14890	1.10978	10.18643	28.03592

reported that HSAEA architecture has less (A x D x P) value when compared to UFA and CSA Architectures for 8-, 16- 32- and 64- bit operand sizes. It has 32.9%, 4.6% reduction in (A x D x P) value when compared to CSA and UFA architectures respectively for 64-bit operand size.

B. Results for various MDCLCG architectures

Incorporating all the designs mentioned in Section II, the proposed 64-bit MDCLCG based PRBG has been successfully synthesized and implemented in Xilinx Vivado 2020.2 using Verilog HDL with target device being Kintex-7 XC7K160TFBG676-3 FPGA board. Behavioral simulation result for the proposed 64-bit MDCLCG which is same for all the three different adder based designs is shown in Fig. 5a. To fit the 64-bit MDCLCG on Kintex-7 xc7K160tfbg676-3 FPGA, four inputs are given per clock cycle and zoom view of simulation result is shown in Fig. 5b. Maximum frequency of Operation, Post-implementation Area and Power metrics of HSAEA, UFA, CSA based 64-bit MDCLCG architectures are presented in TABLE 2.

TABLE 2: Max frequency, Area and Power metrics for various 64-bit MDCLCG architectures

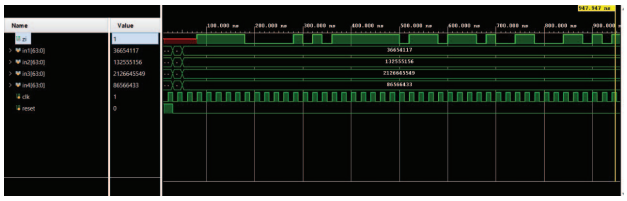
Adder architecture	Max freq(F)	LUT Utilization(A)	Power(P)
HSAEA	225.37 MHz	3.43%	0.292 W
UFA	207.64 MHz	4.21%	0.327 W
CSA	140.11 MHz	2.35%	0.190 W

For better comparison of various 64-bit MDCLCG architectures, Area/Maximum frequency (A/F_{Max}) and Power/Maximum frequency (P/F_{Max}) parameters are introduced. (A/F_{Max}) and (P/F_{Max}) values for various 64-bit MDCLCG architectures are presented in TABLE 3.

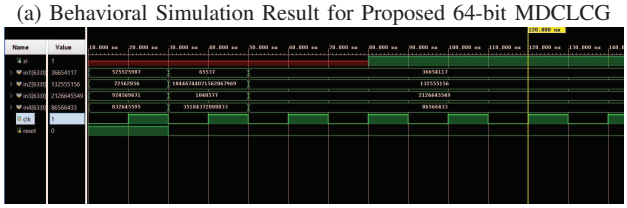
It is worth noting that 64-bit MDCLCG architecture designed using HSAEA has less (A/F_{Max}) and (P/F_{Max}) values when compared to UFA and CSA based 64-bit MDCLCG architectures. It has 25.1%, 9.2% reduction in (A/F_{Max}), and 17.7%, 4.4% reduction in (P/F_{Max}) over 64-bit MDCLCG architectures designed using UFA and CSA respectively. Additionally, it is also observed that HSAEA based 64-bit MDCLCG is 1.6 times faster than CSA based 64-bit MDCLCG and has 18.5% reduction in area when compared to UFA based 64-bit MDCLCG. Proposed 64-bit MDCLCG designed using HSAEA operates at Maximum Frequency of 225.37 MHz and has less LUT Utilization (3.43%).

TABLE 3: (A/F_{Max}) and (P/F_{Max}) values for various 64-bit MDCLCG architectures

Adder architecture	(A/F_{Max}) $(X10^{-8})$	(P/F_{Max}) $(X10^{-9})$
HSAEA	1.522	1.296
UFA	2.032	1.575
CSA	1.677	1.356



(a) Behavioral Simulation Result for Proposed 64-bit MDCLCG



(b) Zoom view of Behavioral Simulation Result for Proposed 64-bit MDCLCG

Fig. 5: Simulation Result for Proposed 64-bit MDCLCG

V. Conclusion

In this paper; the design, implementation and testing of 64-bit MDCLCG has been carried out on Kintex-7 XC7K160TFBG676-3 FPGA. As three operand adder is the main block in the design of MDCLCG, three different three operand adder architectures (HSAEA, UFA, CSA) are considered for the design of the proposed 64-bit MDCLCG and their

perfomance metrics are compared. HSAEA architecture has low Area-Delay-Power product value when compared to UFA and CSA architectures. Also, the proposed 64-bit MDCLCG designed using HSAEA has low (A/F_{Max}) and (P/F_{Max}) values over 64-bit MDCLCG architectures designed using UFA and CSA. Hence, the proposed 64-bit MDCLCG design using HSAEA can be used to improve data security and safety in IOT and all other security applications as it has improvement in terms of Area, Power, Maximum speed of operation and the random sequence has a period of 2^{64} bits. It can also be utilized to accelerate the real-time data acquisition and processing systems such as optical coherence tomography, CT, MRI etc.

REFERENCES

[1] Dipraj, S. Vishwakarma and J. Singh, "Social Internet of Things: The collaboration of Social Network and Internet of Things and its Future," 2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), 2020, pp. 535-539, doi: 10.1109/ICACCCN51052.2020.9362847.

[2] V. Tyagi and A. Kumar, "Internet of Things and social networks: A survey," 2017 International Conference on Computing, Communication and Automation (ICCCA), 2017, pp. 1268-1270, doi: 10.1109/CCAA.2017.8230013.

[3] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The Internet of things: A survey. Computer networks, 54(15):2787–2805, 2010.

[4] E. P. Yadav, E. A. Mittal and H. Yadav, "IoT: Challenges and Issues in Indian Perspective," 2018 3rd International Conference On Internet of Things: Smart Innovation and Usages (IoT-SIU), 2018, pp. 1-5, doi: 10.1109/IoT-SIU.2018.8519869.

[5] J. Zhou, Z. Cao, X. Dong, and A. V. Vasilakos, "Security and privacy for cloud-based IoT: Challenges," IEEE Commun. Mag., vol. 55, no. 1, pp. 26–33, Jan. 2017.

[6] R. Ostrovsky, Foundations of Cryptography (Lecture Notes). Los Angeles, CA, USA: UCLA, 2010

[7] O. Goldreich, Foundations of Cryptography. New York, NY, USA:Cambridge Univ. Press, 2004.

[8] William Stallings. Cryptography and Network Security, 4/E. Pearson Education India, 2006.

[9] A. K. Panda and K. C. Ray, "Modified Dual-CLCG Method and its VLSI Architecture for Pseudorandom Bit Generation," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 66, no. 3, pp. 989-1002, March 2019, doi: 10.1109/TCSI.2018.2876787.

[10] R. S. Katti and R. G. Kavasseri, "Secure pseudo-random bit sequence generation using coupled linear congruential generators," 2008 IEEE International Symposium on Circuits and Systems, 2008, pp. 2929-2932, doi: 10.1109/ISCAS.2008.4542071.

[11] A. Kumar Panda and K. Chandra Ray, "A coupled variable input LCG method and its VLSI architecture for pseudorandom bit generation," IEEE Trans. Instrum. Meas., vol. 69, no. 4, pp. 1011–1019, Apr. 2020.

[12] R. S. Katti and S. K. Srinivasan, "Efficient hardware implementation of a new pseudo-random bit sequence generator," 2009 IEEE International Symposium on Circuits and Systems, 2009, pp. 1393-1396, doi: 10.1109/ISCAS.2009.5118025.

[13] A. K. Panda, R. Palisetty and K. C. Ray, "High-Speed Area-Efficient VLSI Architecture of Three-Operand Binary Adder," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 67, no. 11, pp. 3944-3953, Nov. 2020, doi: 10.1109/TCSI.2020.3016275.

[14] K. S. Pandey, D. K. B, N. Goel and H. Shrimali, "An Ultra-Fast Parallel Prefix Adder," 2019 IEEE 26th Symposium on Computer Arithmetic (ARITH), 2019, pp. 125-134, doi: 10.1109/ARITH.2019.00034.

[15] Brown, Stephen D. Fundamentals of digital logic with Verilog design / Stephen D. Brown, Zvonko G. Vranesic, 1st ed, McGraw Hill Series in electrical and computer engineering

[16] Samir Palnitkar, Verilog HDL: A Guide to Digital Design and Synthesis, Second Edition, Prentice Hall PTR, 2003

[17] Weste, Harris CMOS VLSI DESIGN, a circuits and systems perspective / Neil H. E. Weste, David Money Harris, 4th ed, Copyright © 2011, 2005, 1993, 1985 Pearson Education, Inc., publishing as Addison-Wesley.

[18] Revised NIST Special Publication 800-22. (Apr. 2010). A Statistica Test Suite for the Validation of Random Number Generators and Pseudo Random Number Generators for Cryptographic Applications. [Online]. Available: https://csrc.nist.gov/publications/detail/sp/800-22/rev-1a/final