Modified Dual-CLCG Method and Its VLSI Architecture for Pseudorandom Bit Generation

Amit Kumar Panda, *Student Member, IEEE*, and Kailash Chandra Ray, *Member, IEEE*

***Abstract*— Pseudorandom bit generator (PRBG) is an essential component for securing data during transmission and storage in various cryptography applications. Among popular existing PRBG methods such as linear feedback shift register (LFSR), linear congruential generator (LCG), coupled LCG (CLCG), and dual-coupled LCG (dual-CLCG), the latter proves to be more secure. This method relies on the inequality comparisons that lead to generating pseudorandom bit at a non-uniform time interval. Hence, a new architecture of the existing dual-CLCG method is developed that generates pseudo-random bit at uniform clock rate. However, this architecture experiences several drawbacks such as excessive memory usage and high-initial clock latency, and fails to achieve the maximum length sequence. Therefore, a new PRBG method called as "modified dual-CLCG" and its very large-scale integration (VLSI) architecture are proposed in this paper to mitigate the aforesaid problems. The novel contribution of the proposed PRBG method is to generate pseudorandom bit at uniform clock rate with one initial clock delay and minimum hardware complexity. Moreover, the proposed PRBG method passes all the 15 benchmark tests of NIST standard and achieves the maximal period of 2^n. The proposed architecture is implemented using Verilog-HDL and prototyped on the commercially available FPGA device.**

***Index Terms*— Pseudorandom bit generator (PRBG), VLSI architecture, FPGA prototype.**

I. Introduction

SECURITY and privacy over the internet is the most sensitive and primary objective to protect data in various Internet-of-Things (IoT) applications. Millions of devices which are connected to the internet generate big data that can lead to user privacy issues [1], [2]. Also, there are significant security challenges to implement the IoT whose objectives are to connect people-to-things and things-to-things over the internet [3], [4]. The pseudorandom bit generator (PRBG) is an essential component to manage user privacy in IoT enabled resource constraint devices. A high bit-rate, cryptographically secure and large key size PRBG is difficult to attain due to hardware limitations which demands efficient VLSI architecture in terms of randomness, area, latency and power.

The PRBG is assumed to be random if it satisfies the fifteen benchmark tests of National Institute of Standard and Technology (NIST) standard. Linear feedback shift register (LFSR) and linear congruential generator (LCG) are the most common and low complexity PRBGs. However, these PRBGs badly fail randomness tests and are insecure due to its linearity structure [5], [6]. Numerous studies on PRBG based on LFSR [7], chaotic map and congruent modulo are reported in the literature. Among these, Blum-Blum-Shub generator (BBS) is one of the proven polynomial time unpredictable and cryptographic secure key generator because of its large prime factorize problem [8]–[11]. Although it is secure, the hardware implementation is quite challenging for performing the large prime integer modulus and computing the large special prime integer. There are various architectures of BBS PRBG, discussed in [12] and [13]. Most of them either consume a large amount of hardware area or high clock latency [12], [13]; to mitigate it, a low hardware complexity coupled LCG (CLCG) has been proposed in [14] and [15]. The coupling of two LCGs in the CLCG method makes it more secure than a single LCG and chaotic based PRBGs that generates the pseudorandom bit at every clock cycle [14]. Despite an improvement in the security, the CLCG method fails the discrete Fourier transform (DFT) test and five other major NIST statistical tests [16]. DFT test finds the periodic patterns in CLCG which shows it as a weak generator. To amend this, Katti *et al.* [16] proposed another PRBG method, i.e. dual-CLCG that involves two inequality comparisons and four LCGs to generate pseudorandom bit sequence. The dual-CLCG method generates one-bit random output only when it holds inequality equations. Therefore, it is unable to generate pseudorandom bit at every iteration. Hence, designing an efficient architecture is a major challenge to generate random bit in uniform clock time.

To the knowledge of authors, the hardware architecture of the dual-CLCG method is not deeply investigated in the literature and therefore, in the beginning, the architectural mapping of the existing dual-CLCG method is developed to generate the random bit at a uniform clock rate. However, it experiences various drawbacks such as: large usage of flip-flops, high initial clock latency of 2^n for n-bit architecture, fails to achieve the maximum length period of 2^n (it depends on the number of 0's in the CLCG sequence and is nearly 2^{n-1} for randomly chosen n-bit input seed) and also fails five major NIST statistical tests. Hence, to overcome these shortcomings in the existing dual-CLCG method and its architecture, a new PRBG method and its architecture are proposed in this paper.

The manuscript mainly focuses on developing an efficient PRBG algorithm and its hardware architecture in terms of area, latency, power, randomness and maximum length sequence

Manuscript received April 19, 2018; revised August 16, 2018 and September 24, 2018; accepted October 11, 2018. This work was supported by the Council of Scientific and Industrial Research (CSIR), New Delhi, India. This paper was recommended by Associate Editor F. J. Kurdahi. *(Corresponding author: Amit Kumar Panda.)*

The authors are with the Department of Electrical Engineering, Indian Institute of Technology Patna, Patna 801106, India (e-mail: amitee.srf13@iitp.ac.in; kcr@iitp.ac.in).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSI.2018.2876787

for IoT enabled hardware applications. The proposed PRBG method i.e. "modified dual-CLCG" and its VLSI architecture have the following advantages and novel contributions over previous PRBG method. First, a single XOR logic is utilized at the output stage for generating pseudorandom bit at uniform clock rate which leads to lower the hardware cost. Secondly, it generates a maximum length of 2^n pseudorandom bits with one initial clock latency. Third, the proposed modified dual-CLCG method passes all the fifteen benchmark tests of NIST standard and is proved to be polynomial-time unpredictable. The randomness tests are performed using NIST test tool *sts-2.1.2* [17]. Further, the properties of the proposed PRBG method are investigated theoretically by using the probabilistic approach. It shows that the proposed modified dual-CLCG system has the similar security strength of dual-CLCG method and the probabilistic algorithm to obtain the initial seed requires the solution of $n2^{4n}$. The architecture of the existing dual-CLCG method and the proposed modified dual-CLCG method for different word size of n = 8-, 16- and 32- bits are implemented using Verilog HDL and prototyped on commercially available FPGA devices such as Spartan3E XC3S500E and Virtex5 XC5VLX110T. The real-time validation is performed using Xilinx Chipscope.

This paper is organized as follows: architectural mapping of the existing dual-CLCG method is performed and its drawbacks are highlighted in Section-II. The proposed PRBG method along with its randomness properties are discussed in Section-III. Section-IV presents the efficient VLSI architecture of the proposed modified dual-CLCG method. The experimental setup and FPGA prototype of the proposed PRBG method are demonstrated in Section-V. Finally, Section-VI is incorporated to conclude this article.

II. Architectural Mapping of the Existing Dual-CLCG Method

The dual-CLCG method is a dual coupling of four linear congruential generators proposed by Katti *et al.* [16] and is defined mathematically as follows:

$$x_{i+1} \equiv a_1 \times x_i + b_1 \bmod 2^n \tag{1}$$

$$y_{i+1} \equiv a_2 \times y_i + b_2 \bmod 2^n \tag{2}$$

$$p_{i+1} \equiv a_3 \times p_i + b_3 \bmod 2^n \tag{3}$$

$$q_{i+1} \equiv a_4 \times q_i + b_4 \bmod 2^n \tag{4}$$

$$Z_i = \begin{cases} 1 & \text{if } x_{i+1} > y_{i+1} \quad \text{and } p_{i+1} > q_{i+1} \\ 0 & \text{if } x_{i+1} < y_{i+1} \quad \text{and } p_{i+1} < q_{i+1} \end{cases} \tag{5}$$

The output sequence Z_i can also be computed in an alternative way as described in [15], i.e.,

$$Z_i = B_i \quad \text{if } C_i = 0 \tag{6}$$

Where,

$$B_i = \begin{cases} 1, & \text{if } x_{i+1} > y_{i+1} \\ 0, & \text{else} \end{cases}; \quad C_i = \begin{cases} 1, & \text{if } p_{i+1} > q_{i+1} \\ 0, & \text{else} \end{cases} \tag{7}$$

Here, a_1, b_1, a_2, b_2, a_3, b_3, a_4 and b_4 are the constant parameters; x_0, y_0, p_0 and q_0 are the initial seeds. Following are the necessary conditions to get the maximum period.

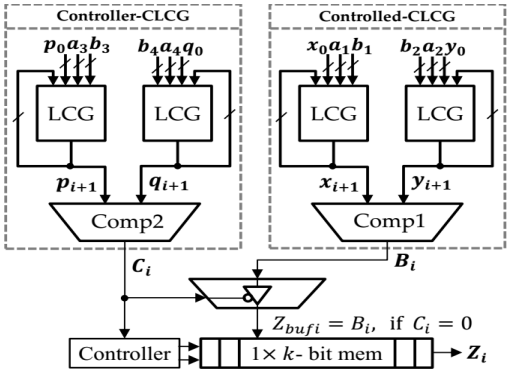

Fig. 1. Architectural mapping of the existing dual-CLCG method.

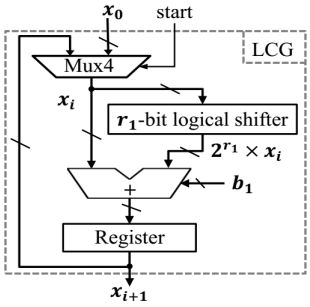

Fig. 2. Architecture of the linear congruential generator.

(i) b_1, b_2, b_3 and b_4 are relatively prime with $2^n(m)$.
(ii) (a_1-1), (a_2-1), (a_3-1) and (a_4-1) must be divisible by 4.

Following points can be observed from the dual-CLCG method i.e.

1. The output of the dual-CLCG method chooses the value of B_i when C_i is 'zero'; else it skips the value of B_i and does not give any binary value at the output.
2. As a result, the dual-CLCG method is unable to generate pseudorandom bit at each iteration.

A. Architecture Mapping of the Existing Dual-CLCG Method

The scope of the work presented in [16] is limited to the algorithmic development. However, it lacks the architectural design of the dual-CLCG method. Hence, a new hardware architecture of the existing dual-CLCG method is developed to generate pseudorandom bit at an equal interval of time for encrypting continuous data stream in the stream cipher. The architecture is designed with two comparators, four LCG blocks, one controller unit and memory (flip-flops) as shown in Fig. 1. The LCG is the basic functional block in the dual-CLCG architecture that involves multiplication and addition processes to compute n-bit binary random number on every clock cycle. The multiplication in the LCG equation can be implemented with shift operation, when a is considered as (2^r+1). Here, r is a positive integer, $1 < r < 2^n$. Therefore, for the efficient computation of x_{i+1}, the equation (1) can be rewritten as,

$$\begin{aligned} x_{i+1} &\equiv (a_1 \times x_i + b_1) \bmod 2^n \equiv [(2^{r_1} + 1)x_i + b_1] \bmod 2^n \\ &\equiv [(2^{r_1} \times x_i) + x_i + b_1] \bmod 2^n \end{aligned}$$

The architecture of LCG shown in Fig. 2 is implemented with a 3-operand modulo 2^n adder, 2×1 n-bit multiplexer

and n-bit register. LCG generates a random n-bit binary equivalent to integer number in each clock cycle. Other three LCG equations can also be mapped to the corresponding architecture similar to the LCG equation (1). To implement the inequality equation, a comparator is used that compares the output of two LCGs. The comparator and two linear congruential generators (LCG) are combined to form a coupled-LCG (CLCG). Two CLCGs are used in the dual-CLCG architecture. One is called controller-CLCG which generates C_i and another one is called controlled-CLCG which generates B_i. To perform the operation of $Z_i = B_i$ if $C_i = 0$, a 1-bit tristate buffer is employed that selects the B_i (output of controlled-CLCG) only when $C_i = 0$ (the output of controller-CLCG) and it does not select the value of B_i while $C_i = 1$. Since the CLCG output (C_i) is random; it selects tristate buffer randomly. Therefore, the direct architectural mapping of the existing dual-CLCG method does not generate random bits in every clock cycle at the output of the tristate buffer. In this case, the overall latency varies accordingly with the number of consecutive 1's between two 0's in the sequence C_i (output of controller-CLCG). This asynchronous generation of pseudo-random bits is applicable only where asynchronous interface is demanded. However, in stream cipher encryption, the key size should be larger than the message size where bitwise operation is performed at every clock cycle. Therefore, a fixed number of flip-flops can be employed at the output stage of dual-CLCG architecture for generating pseudorandom bit in a uniform clock time.

If the sequence C_i is considered to be known and the zero-one combinations are in a uniform pattern, then the fixed number of flip-flops can be estimated to generate random bit at every uniform clock cycle. For example:

If,

$$C_i = (\underline{\mathbf{0}, 1, 1, \mathbf{0}},\ \ \underline{1, \mathbf{0}, \mathbf{0}, 1},\ \ \underline{\mathbf{0}, 1, \mathbf{0}, 1});$$

and

$$B_i = (\mathbf{1}, 0, 1, \mathbf{1},\ \ 1, \mathbf{0}, \mathbf{0}, 0,\ \ \mathbf{1}, 1, \mathbf{0}, 1);$$

then,

$$Z_i = B_i \text{ if } C_i = 0;\ \ Z_{bufi} = (1, \mathbf{x}, \mathbf{x}, 1,\ \ \mathbf{x}, 0, 0, \mathbf{x},\ \ 1, \mathbf{x}, 0, \mathbf{x})$$

In this example, it is observed that the number of 0's and 1's are equal in every 4-bit patterns. There are two 0's and two 1's in every pattern. Therefore, a pair of two flip-flops (two 2-bit registers) is sufficient to get the random bit in the uniform clock cycle. However, C_i is not known to the user and is not always in a uniform pattern. Consider an example,

If,

$$C_i = (\underline{\mathbf{0}, 1, \mathbf{0}, \mathbf{0}},\ \ \underline{\mathbf{0}, 1, 1, \mathbf{0}},\ \ \underline{\mathbf{0}, 1, 1, 1},\ \ \underline{\mathbf{0}, 1, \mathbf{0}, 1});$$

and

$$B_i = (\mathbf{1}, 0, \mathbf{1}, \mathbf{1},\ \ \mathbf{0}, 0, 0, \mathbf{1},\ \ \mathbf{0}, 0, 1, 0,\ \ \mathbf{1}, 1, \mathbf{1}, 1);$$

then,

$$Z_{bufi} = (1, \mathbf{x}, 1, 1,\ \ 0, \mathbf{x}, \mathbf{x}, 1,\ \ 0, \mathbf{x}, \mathbf{x}, \mathbf{x},\ \ 1, \mathbf{x}, 1, \mathbf{x})$$

No such uniform pattern can be observed in the sequence C_i of the above example. Therefore, the fixed number of flip-flops cannot be estimated in this case.

Considering this problem as mentioned above, k number of bits are generated first and then stored in k-flipflops (termed as $1 \times k$-bit memory) when $C_i =$ '0'. Further, these stored bits are released at every equal interval of clock cycle. If there are k number of 0's in the sequence C_i, then the dual-CLCG architecture can generate maximum of k- random bits in 2^n- clock cycles. It can utilize k- flip-flops to store k different bits of B_i while $C_i =$ '0'. After 2^n clock cycles, it releases these stored bits serially at every two clock cycles (1-bit per two clock cycles). Here, it is assumed that the number of 0's = number of 1's = 2^{n-1} (for n-bit input seed) in the sequence C_i. This architecture may work even if the number of 0's in the sequence C_i are less than 2^{n-1}. However, when the number of 0's are greater than the number of 1's, then this architecture may not work correctly. The maximum length period of dual-CLCG method depends on the number of 0's in C_i and is nearly 2^{n-1} for randomly chosen n-bit input seed.

The maximum combinational path delay in the dual-CLCG architecture is the three-operand adder with multiplexer. The reason is that three-operand adder with multiplexer has highest critical path delay as compared to the comparator. Therefore,

$$\text{Area, } A_{DCLCG} = 4A_{LCG} + 2A_{cmp} + A_{tri} + A_{mem} + A_{cntrl}$$
$$\text{Critical path, } T_{DCLCG} = T_{add} + T_{mux}$$

B. Drawbacks of the Existing Dual-CLCG Method and Its Architecture

Although the dual-CLCG method is secure when compared with CLCG and single LCG [16], it suffers from several drawbacks in terms of hardware and randomness test as mentioned below,

1. It requires control circuit and a large number of flip-flops (referred as memory). It requires k flip-flops for generating k number of pseudorandom bits. In this case, it is assumed $k \approx 2^{n-1}$ for randomly chosen n-bit input seed, therefore it needs 2^{n-1} flip-flops.
2. Initial clock latency (input to first output) depends on the number of clock cycles required to generate k random bits. The dual-CLCG architecture takes 2^n initial clock latency if k is considered as maximum length.
3. Maximum length period of dual-CLCG method depends on the number of 0's in C_i and is nearly 2^{n-1} for randomly chosen n -bit input seed.
4. The proposed architecture works when it is assumed that the number of 0's $\leq$ number of 1's in a maximum length sequence generated from controller-CLCG.
5. The existing dual-CLCG method fails five major NIST randomness tests [16].

III. Proposed PRBG Method

To overcome the aforesaid shortcomings in the existing dual-CLCG method and its architecture as highlighted in Section-II-B, a new PRBG method and its architecture are proposed in this paper. The proposed PRBG method is the modified version of the dual-CLCG method referred as "Modified dual-CLCG", in which the equation (6) of existing dual-CLCG method is replaced with the new equation (12),

whereas the other four equations from (8) to (11) are same as in the case of dual-CLCG method from equation (1) to (4).

A. Proposed Modified Dual-CLCG Method and Its Algorithm

The proposed modified dual-CLCG method generates pseudorandom bits by congruential modulo-2 addition of two coupled linear congruential generator (CLCG) outputs and is mathematically defined as follows,

$$x_{i+1} \equiv a_1 \times x_i + b_1 \bmod 2^n \quad (8)$$
$$y_{i+1} \equiv a_2 \times y_i + b_2 \bmod 2^n \quad (9)$$
$$p_{i+1} \equiv a_3 \times p_i + b_3 \bmod 2^n \quad (10)$$
$$q_{i+1} \equiv a_4 \times q_i + b_4 \bmod 2^n \quad (11)$$

The pseudorandom bit sequence Z_i is obtained by using the congruential modulo-2 equation (12),

$$Z_i \equiv (B_i + C_i) \bmod 2 = B_i \oplus C_i \quad (12)$$

Where,

$$B_i = \begin{cases} 1, & \text{if } x_{i+1} > y_{i+1} \\ 0, & \text{else} \end{cases} \quad \text{and } C_i = \begin{cases} 1, & \text{if } p_{i+1} > q_{i+1} \\ 0, & \text{else} \end{cases}$$

Here, $a_1, b_1, a_2, b_2, a_3, b_3, a_4$ and b_4 are the constant parameters; x_0, y_0, p_0 and q_0 are the initial seeds. The necessary conditions to get the maximum length period are same as the existing dual-CLCG method (as discussed in Section-II). The proposed modified dual-CLCG method uses the congruential modulo-2 addition of two different coupled LCG outputs as specified in equation (12). Hence, the congruential modulo-2 addition does not skip any random bits at the output stage and produces one-bit random output in each iteration. Since, the coupled LCG has the maximal period [16], the modulo-2 addition of two coupled-LCG outputs in the modified dual-CLCG have also the same maximum length period of 2^n for n-bit modulus operand. To perform the modulo-2 addition operation, it takes only single XOR logic. Therefore, by replacing equation (6) with equation (12), the proposed PRBG method can reduce the large memory area used in the existing dual-CLCG method and also can achieve the full-length period of 2^n. The step by step procedure to evaluate the pseudorandom bit sequence in the proposed PRBG method is summarized in the algorithm form in Algorithm 1.

Example of the Proposed Modified Dual-CLCG Method: Let $a_1 = 5$, $b_1 = 5$, $a_2 = 5$, $b_2 = 3$, $a_3 = 5$, $b_3 = 1$, $a_4 = 5$, $b_4 = 7$ and $m = 2^3$. The sequences x_i, y_i, p_i and q_i have a period of 8 and are hence full period. If the initial condition or the seed are $(x_0, y_0, p_0, q_0) = (2, 7, 3, 4)$ then the generated sequences are

$$X_i = (7, 0, 5, 6, 3, 4, 1, 2); \quad P_i = (0, 1, 6, 7, 4, 5, 2, 3);$$
$$Y_i = (6, 1, 0, 3, 2, 5, 4, 7); \quad Q_i = (3, 6, 5, 0, 7, 2, 1, 4);$$

Therefore, the output sequences B_i and C_i are evaluated as,

$$B_i = (1, 0, 1, 1, 1, 0, 0, 0); \quad C_i = (0, 0, 1, 1, 0, 1, 1, 0)$$

The final sequence Z_i generated from the proposed modified dual-CLCG method for $n = 3$- bit is,

$$Z_i = (B_i + C_i) \bmod 2 = B_i \oplus C_i = (1, 0, 0, 0, 1, 1, 1, 0)$$

Algorithm 1 Modified Dual-CLCG Algorithm to Generate Pseudorandom Bit Sequence Z_i

Input: n (positive integer), $m = 2^n$.
Initialization:
$b_1, b_2, b_3, b_4 < m$, such that these are relatively prime with m.
$a_1, a_2, a_3, a_4 < m$ *s.t.* (a_1-1), (a_2-1), (a_3-1) and (a_4-1) must be divisible by 4.
Initial seeds x_0, y_0, p_0 and $q_0 < m$.
Output: Z_i
1. **for** $i = 0$ **to** k
2. Compute x_{i+1}, y_{i+1}, p_{i+1}, q_{i+1} using equation (8), (9), (10) and (11) respectively;
3. **if** $x_{i+1} > y_{i+1}$, **then** $B_i = 1$ **else** $B_i = 0$;
4. **if** $p_{i+1} > q_{i+1}$, **then** $C_i = 1$ **else** $C_i = 0$;
5. $Z_i = (B_i + C_i) \bmod 2$;
6. **Return** Z_i;

Before developing the architecture of the "Modified dual-CLCG" algorithm, the randomness properties are analyzed in the next subsequent sections.

B. NIST Test Analysis of the Proposed Modified Dual-CLCG Method for Randomness Evaluation

In this section, the statistical properties of the proposed modified dual-CLCG method are discussed by conducting fifteen benchmark randomness tests of NIST standard. The NIST tests are performed using NIST test tool *sts-2.1.2* [17] on $T = 100$ and 1000 different binary sequences of length 10^6 bits which are generated from the proposed PRBG. All the sequences are generated from different randomly chosen seeds using the parameters $a_1 = 129$, $b_1 = 32177$, $a_2 = 65$, $b_2 = 1533$, $a_3 = 4097$, $b_3 = 571$, $a_4 = 1025$ and $b_4 = 732969$. The size of the modified dual-CLCG method is considered as $n = 24$-, 32- and 40- bit. The initial seeds (x_0, y_0, p_0, q_0) are chosen from a true random source [18]. The threshold level *alpha* (α) is considered as 0.01 for passing the tests. If P-value ≥ 0.01, then the sequence appears to be random with a confidence of 99%. After collecting T number of P-values for each test, a Goodness-of-Fit distributional test is performed to check whether these values are uniformly distributed in [0, 1]. Such computation provides the U-values, and if U-value ≥ 0.0001, then the sequences can be considered uniformly distributed. The NIST test results of the proposed PRBG method are highlighted in Table I. It is observed that the proposed PRBG method for the size of $n \geq 24$- bits passes all the fifteen benchmark randomness tests of NIST standard with a high degree of consistency.

The NIST statistical test results of the proposed modified dual-CLCG method of 24- bit size are further compared with the testing results of other existing PRBG methods in Table II. It is reported that the proposed modified dual-CLCG method is the only method among others, which successfully passes the 'non-periodic templates' (NOT) test. 'NOT' is the most difficult test consisting of 149 subtests that detect irregular occurrences of the possible template pattern [17]. The existing dual-CLCG method fails to meet uniformity requirements for

TABLE I

NIST STATISTICAL TESTS OF THE PROPOSED PRBG METHOD

Sl. No.	TEST	Proposed PRBG *n*=24-bit; *T*= 100		Proposed PRBG *n*=32-bit; *T*= 100		Proposed PRBG *n*=40-bit; *T*= 100		Proposed PRBG *n*=24-bit; *T*= 1000		Proposed PRBG *n*=32-bit; *T*= 1000		Proposed PRBG *n*=40-bit; *T*= 1000	
		#Prop.	*U-value	#Prop.	*U-value	#Prop.	*U-value	#Prop.	*U-value	#Prop.	*U-value	#Prop.	*U-value
01	Frequency (monobit)	0.98	0.366918	0.99	0.437274	1.0	0.816537	0.984	0.856359	0.990	0.825505	0.990	0.002253
02	Block-Frequency	1.0	0.851383	1.0	0.122325	0.99	0.289667	0.992	0.016488	0.996	0.431754	0.992	0.632955
03	Cumulative Sum (F)	0.98	0.096578	0.98	0.616305	1.0	0.595549	0.987	0.958485	0.991	0.842937	0.989	0.350485
	Cumulative Sum (R)	0.99	0.699313	0.99	0.554420	1.0	0.851383	0.986	0.914025	0.990	0.846338	0.992	0.471146
04	Runs	1.0	0.657933	0.99	0.851383	0.99	0.987896	0.991	0.792508	0.991	0.234373	0.991	0.331408
05	Longest Run	0.99	0.798139	0.97	0.437274	0.98	0.289667	0.988	0.116746	0.984	0.081510	0.983	0.765632
06	Binary Rank	0.99	0.058984	0.98	0.262249	0.99	0.678686	0.989	0.292519	0.991	0.897763	0.990	0.554420
07	Spectral (DFT)	0.98	0.037566	1.0	0.062821	0.98	0.719747	0.993	0.820143	0.992	0.248014	0.988	0.098330
08	NOT** (m = 9)	0.96	0.045675	0.96	0.213309	0.96	0.055361	0.982	0.151190	0.980	0.444691	0.982	0.062821
09	OLT	0.96	0.554420	0.98	0.883171	1.0	0.637119	0.985	0.781106	0.988	0.058243	0.990	0.954930
10	Universal	1.0	0.401199	0.99	0.319084	0.97	0.779188	0.988	0.920383	0.986	0.339271	0.986	0.699313
11	Entropy	0.99	0.066882	0.99	0.474986	0.98	0.289667	0.988	0.213309	0.993	0.045971	0.993	0.440975
12	Rand Excursion†	0.9687	0.275709	0.9697	0.020085	0.9692	0.170294	0.9869	0.322085	0.9811	0.826730	0.9857	0.981517
13	Rand Ex. Var.†	0.9687	0.706149	0.9848	0.006196	0.9692	0.337162	0.9837	0.498249	0.9897	0.214283	0.9809	0.142859
14	Serial††	0.99	0.474986	0.99	0.191687	0.99	0.048716	0.988	0.266235	0.985	0.729870	0.989	0.568739
15	Linear Complexity	0.98	0.911413	0.98	0.055361	0.99	0.224821	0.992	0.715679	0.993	0.260930	0.987	0.707513

TABLE II

COMPARISON OF THE PROPOSED AND OTHER EXISTING PRBG METHOD (NIST TEST RESULTS)

Sl. No.	TEST	Proposed PRBG		Dual-CLCG [16]		CLCG [16]		Nonlin. PRBG [19]		24-bit LFSR [19]		128-bit LFSR based PRPG [7]	
		#Prop.	*U-value	#Prop.	*U-value	#Prop.	*U-value	#Prop.	*U-value	#Prop.	*U-value	#Prop.	*U-value
01	Freq. (monobit)	0.98	0.366918	1.00	**<0.0001**	1.0	**0.000000**	1.0000	0.004146	1.0000	0.019857	0.99	0.49439
02	Block-Frequency	1.0	0.851383	0.96	0.0046	0.99	0.350485	1.0000	0.001399	0.9950	0.955835	0.99	0.75975
03	Cum. Sum (F)	0.98	0.096578	1.00	**<0.0001**	1.0	0.002374	1.0000	0.060875	1.0000	**<10E-4**	0.98	0.40119
	Cum. Sum (R)	0.99	0.699313	1.00	**<0.0001**	1.0	0.003201	1.0000	0.060875	1.0000	**<10E-4**	1.00	0.23681
04	Runs	1.0	0.657933	**0.61**	**<0.0001**	**0.88**	**0.000000**	0.9700	0.080519	1.0000	0.626709	0.99	0.97807
05	Longest Run	0.99	0.798139	1.00	0.4190	0.99	0.025193	1.0000	0.585209	1.0000	0.000361	0.99	0.75975
06	Binary Rank	0.99	0.058984	0.99	0.8831	0.98	0.319084	1.0000	0.842937	**0.0000**	**<10E-4**	1.00	0.21331
07	Spectral (DFT)	0.98	0.037566	0.98	0.3838	**0.0**	**0.000000**	0.9900	0.401199	**0.0000**	**<10E-4**	1.00	0.23681
08	NOT** (m = 9)	0.96	0.045675	**0.95**	0.7597	**0.95**	0.304126	**0.9200**	**< 10E-4**	0.8950	**<10E-4**	**0.95**	0.03081
09	OLT	0.96	0.554420	0.98	0.0487	1.0	0.983453	1.0000	0.125927	0.9900	0.474986	0.98	0.79814
10	Universal	1.0	0.401199	0.98	0.8165	0.99	0.319084	0.9750	0.036352	1.0000	**<10E-4**	1.00	0.03292
11	Entropy	0.99	0.066882	0.99	0.4190	1.0	0.779188	1.0000	**< 10E-4**	1.0000	**<10E-4**	0.98	0.24928
12	Rand Excursion†	0.9687	0.275709	0.9785	0.2607	**0.952**	0.624107	0.9861	0.197677	0.9779	0.314428	94.82	0.69931
13	Rand Ex. Var.†	0.9687	0.706149	0.9892	0.683283	0.964	0.340461	0.9861	0.696376	0.9632	0.444405	98.27	0.19168
14	Serial††	0.99	0.474986	1.00	0.6163	1.0	0.213309	1.0000	**< 10E-4**	1.0000	**<10E-4**	1.00	0.16260
15	Linear Complexity	0.98	0.911413	0.99	0.2757	0.99	0.494392	0.9950	0.090936	**0.0000**	**<10E-4**	**0.00**	**0.00000**

#Prop *- represents proportion which is the ratio between the number of successes and the number of trials. For passing the test,* ***Prop*** *value is 0.96.*

****U-value*** *- It reports the output of the Goodness-of-Fit Distributional Test performed on the collected P-values.*

The Non-overlapping template test consists of several sub-tests (for* ***m = 9*, there are* ***149*** *sub-tests): the worst result is reported.*

†Rand. Excursions and Rand. Excursions Var. consists of eight and 18 tests, respectively. ††Serial Test consists of 2 sub-tests: the worst result is reported.

three NIST statistical tests such as 'Frequency', 'Cumulative Sum' and 'Runs'. It also fails to meet the accepted success-trial ratio for 'Runs' and two out of 149 'NOT'. Likewise, CLCG method fails to meet the uniformity requirement for three tests, i.e. 'Frequency', 'Runs' and 'DFT'. Besides this, it fails to meet the accepted success- trial ratio for three tests such as 'Runs', 'DFT' and 'NOT'. In nonlinear PRBG [19], success- trial ratio of 'NOT' test and uniformity requirements

of 'Entropy' and 'Serial' are failed. Further, major NIST tests are failed in 24- bit LFSR method [19]. Similarly, the 128- bit LFSR based weighted pseudorandom test generator [7] also fails to meet both the uniformity requirements and the success-trial ratio of 'linear complexity' test. The NIST test results in Table I and Table II reveal that the proposed modified dual-CLCG method is very efficient for generating high quality pseudo-random bits than the other existing PRBG methods.

C. Measuring Randomness by Linear Complexity Profiles

Linear complexity profile is an important testing method to measure the randomness of a periodic sequence. In the previous subsection, the proposed modified dual-CLCG method has successfully passed the linear complexity test of NIST standard, but unfortunately, it ignores some details of the linear complexity behavior. This subsection presents the computation of linear complexity profile by applying the Berlekamp-Massey algorithm. The linear complexity L_k of a k length binary sequence is the shortest number of stages of a LFSR that generates the same sequence [20]. Intuitively, non-linearity is observed in the sequence when L_k is small and therefore, cryptographically strong sequence must have a high linear complexity. The sequence generated by a PRBG should have these three properties regarding linear complexity:

- Its linear complexity profile graph should be close to the $k/2$- line in its first period.
- Its linear complexity graph should consist of irregular stair cases with average height of 2 and average length of 4 in its first two periods.
- Its linear complexity should be close to its minimal period for $k = 2N$.

Consider the example of generating a maximum length sequence 's' of 32 bits ($k = 32$) by the proposed modified dual-CLCG method of 5-bit word size and then expand it by repeating the first period of 32 bits length. Therefore,

$$s^{32} = (01001111100001000011100001011000)$$
$$s^{64} = (01001111100001000011100001011000\ldots011000)$$

The linear complexity profile of the sequence 's' for its first two periods is illustrated in Fig. 3 which shows that it is quite close to the $k/2$-line and $L_k(s_k)$ stops increasing at 32 (close to the minimal period of the sequence) after $k = 64$ bits. Fig. 4 presents the linear complexity profile of the sequence obtained by the proposed modified dual-CLCG method of 32-bit word length for its first 100000 (10^5) bits. It shows that the linear complexity graph grows approximately as $k/2$ -line with the average height of 2 and the average length of 4 for k-bit sequence and therefore, $L_k(s_k) = 50000$.

D. Properties of the Proposed Modified Dual-CLCG Method

The properties of the CLCG and dual-CLCG appear in [14] and [16] show that the coupling of two/four LCGs makes it more secure. Therefore, the proposed modified dual-CLCG method which involves the dual coupling of four LCGs has also the similar security strength like CLCG/dual-CLCG method. The properties of the proposed modified dual-CLCG method are derived from the complexity based number

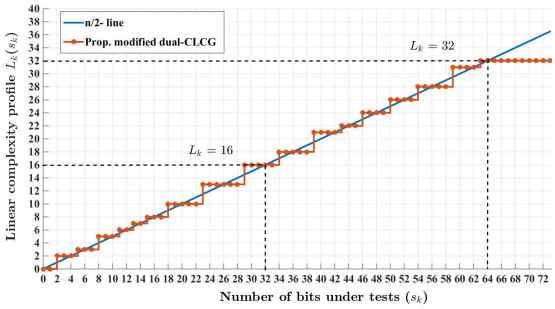

Fig. 3. Linear complexity profiles for first two periods obtained by the proposed modified dual-CLCG method. Here, period N = 32 bits for n = 5-bit.

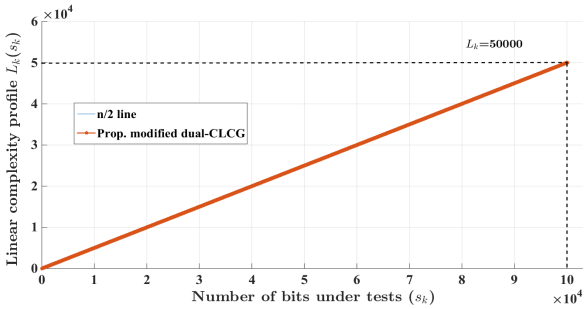

Fig. 4. Linear complexity profiles for first 100000 bits obtained by the proposed modified dual-CLCG method.

theoretic assumptions and arguments by using the probabilistic approach.

The basic property of a PRBG for the application of stream cipher is that the key size should be sufficiently large and must be as long as the message [17]. In case of proposed modified dual-CLCG method, the maximum period occurs only when LCG is considered as maximal period. The sequence of LCG has maximal period if and only if gcd(b, m) = 1 and ($a - 1$) is divisible by 4. Therefore, the sequence of the proposed modified dual-CLCG has maximal period if and only if $\gcd(b_1, m) = \gcd(b_2, m) = \gcd(b_3, m) = \gcd(b_4, m) = 1$ and (a_1-1), (a_2-1), (a_3-1) and (a_4-1) are divisible by 4.

Another important property is that a PRBG must be able to generate a sequence of bits which cannot be distinguished in polynomial time from a truly random distribution even though such a sequence is deterministically generated, given a short truly random seed [21], [22]. Therefore, the probabilistic algorithm to find the next bit and the initial seed must have the negligible probability of success. In the existing CLCG method, two aspects, namely, those of 1) forward and 2) backward unpredictability are addressed with different lemmas. The first aspect is addressed with *Lemma 2.1*, whereby $Pr(B_i = 1 \text{ or } 0) \approx 1/2$ for large m. The second aspect is addressed by quantifying the complexity of solving the CLCG problem, which is exponential in $O(2^{2n})$ [16]. Therefore, these two notions for the proposed PRBG are also addressed with the same mathematical procedures as applied in the existing dual-CLCG method. According to [16, Lemma 2.1],

$$Pr\,(X_i = j) = Pr\,(Y_i = j) = \frac{1}{m}, \quad j \in 0, 1, \cdots, m-1$$

Where, X_i, Y_i, P_i and Q_i are the output of the four different LCGs and is assumed that these are maximal period. Therefore, the probability of getting 0's and 1's in the sequence of both CLCGs are computed as,

$$\begin{aligned} &Pr\,(B_i = 0) \\ &= \sum_{j=0}^{m-1} Pr(X_i = j)Pr(Y_i \geq j) = \frac{1}{m}\sum_{j=0}^{m-1} Pr(Y_i \geq j) \\ &= \frac{1}{m}\sum_{j=0}^{m-1} Pr(Y_i = j) + Pr\,(Y_i = j+1) + \ldots \\ &\quad + Pr\,(Y_i = m-1) \\ &\Rightarrow Pr\,(B_i = 0) = \frac{1}{m}\left[\frac{1}{m}\left(\frac{m(m+1)}{2}\right)\right] = \frac{m+1}{2m} \end{aligned}$$

Similarly, $Pr\,(B_i = 1) = \frac{m-1}{2m}$. Since C_i is another output bit sequence generated from the second CLCG in the proposed modified dual-CLCG method and is obtained in same procedures like B_i, therefore

$$Pr\,(C_i = 0) = Pr\,(B_i = 0) = \frac{m+1}{2m}$$
$$Pr\,(C_i = 1) = Pr\,(B_i = 1) = \frac{m-1}{2m}$$

Further, the probability of getting 0's and 1's in the output sequence of the proposed modified dual-CLCG method can be determined in the following lemma.

Lemma 1: $Pr\,(Z_i = 0) \approx Pr\,(Z_i = 1) \approx 1/2$.

Proof: $Pr\,(Z_i = 0)$ is computed as follows,

$$\begin{aligned} Pr\,(Z_i = 0) = {} & Pr(B_i = 1)Pr(C_i = 1) \\ & + Pr(B_i = 0)Pr(C_i = 0) \end{aligned}$$

Here, B_i and C_i are the two independent random processes that are computed by comparing X_i with Y_i and P_i with Q_i respectively. Hence,

$$Pr\,(Z_i = 0) = \left(\frac{m-1}{2m}\right)\left(\frac{m-1}{2m}\right) + \left(\frac{m+1}{2m}\right)\left(\frac{m+1}{2m}\right)$$
$$Pr\,(Z_i = 0) = \frac{1}{4m^2}\left[2m^2 + 2\right] = \frac{m^2+1}{2m^2}$$

Similarly, the probability of getting 1's is,

$$Pr\,(Z_i = 1) = \frac{m^2-1}{2m^2}$$

From these probabilistic analyses, if m is considered to be very large then $m^2 \gg 1$ and hence

$$Pr\,(Z_i = 0) \approx Pr\,(Z_i = 1) \approx \frac{1}{2}$$

The security strength of the proposed PRBG method depends on the statistical distance between the uniform distribution and the probability distribution of the sequence Z_i of the proposed generator. If the statistical distance $dist\,(D_k, U_k)$ between probability ensembles $\{D_k\}$ and $\{U_k\}$ is negligibly small then the random generator is secure [22], [23]. Therefore, the probability ensembles $\{D_k\}$ and $\{U_k\}$ are called statistically close or statistical indistinguishability if their statistical difference is negligible in k [22]–[24] such that

$$dist\,(D_k, U_k) = \frac{1}{2}\sum_{s\in\{0,1\}^k}\left|Pr\,(S = s) - \frac{1}{2^k}\right| \leq \varepsilon(k) \quad (13)$$

Here D_k and U_k are the probability and uniform distributions respectively over $\{0,1\}^k$ for k-bit sequence. $\{0,1\}^k$ are k-bit segments and S is the random variable that refers to k consecutive outputs of the sequence Z_i in equation (12). If k-bit sequences are generated according to uniform distribution, then the probability of obtaining any sequence is $1/2^k$. For generating the maximum length sequence, the value of $k = m = 2^n$ is chosen. The probability of the occurrence of the random variable S in the proposed modified dual-CLCG can be evaluated by multiplying the individual probability of each outcome in the sequence as described in [15] and [16] and is defined as follows,

$$Pr\,(S = s) = \prod_{i=1}^{k} Pr\,(Z_i = z_i) \quad (14)$$

Here, z_i is the one-bit output of the proposed PRBG method. Practically, Z_i is computed as $Z_i = (B_i + C_i) \bmod 2$, where B_i and C_i are two independent random processes.

Lemma 2: $Pr\,(S = s) \approx \frac{1}{2^k}$.

Proof: The probability of the occurrence of random variable S, $Pr\,(S = s)$ for the proposed generator is computed as,

$$\begin{aligned} Pr\,(S = s) &= \prod_{i=1}^{k} Pr\,(Z_i = z_i) \\ &\approx \prod_{i=1}^{k_1} Pr\,(Z_i = 0)\prod_{i=1}^{k_2} Pr\,(Z_i = 1) \\ &\approx \left(\frac{m^2+1}{2m^2}\right)^{k_1}\left(\frac{m^2-1}{2m^2}\right)^{k_2} \end{aligned} \quad (15)$$

Where, k_1 and k_2 are the total numbers of 0's and 1's count in a proposed PRBG sequence such that $k_1 + k_2 = k$. If m is very large then $m^2 \gg 1$ and hence

$$Pr(S = s) \approx \frac{1}{2^{k_1}}\frac{1}{2^{k_2}} = \frac{1}{2^{k_1+k_2}} = \frac{1}{2^k}$$

Now, the statistical distance between the probability ensembles $\{D_k\}$ and $\{U_k\}$ can be measured for the proposed modified dual-CLCG method by substituting the equation (15) in equation (13).

$$\begin{aligned} &dist\,(D_k, U_k) \\ &= \frac{1}{2}\sum_{s\in\{0,1\}^k}\left|Pr\,(S = s) - \frac{1}{2^k}\right| \\ &\approx \frac{1}{2}\sum_{s\in\{0,1\}^k}\left|\left(\frac{m^2+1}{2m^2}\right)^{k_1}\left(\frac{m^2-1}{2m^2}\right)^{k_2} - \frac{1}{2^k}\right| \\ &\approx \frac{1}{2}\sum_{s\in\{0,1\}^k}\left|\frac{1}{2^k}\left(1+\frac{1}{m^2}\right)^{k_1}\left(1-\frac{1}{m^2}\right)^{k_2} - \frac{1}{2^k}\right| \end{aligned}$$

By using the Binomial theorem, the series $\left(1+\frac{1}{m^2}\right)^{k_1}$ and $\left(1-\frac{1}{m^2}\right)^{k_2}$ in the above equation are expanded and further

simplified as follows,

$$dist\,(D_k, U_k) = \frac{1}{2}\left|\left\{\begin{array}{l}\dfrac{(k_1-k_2)}{m^2} + \dfrac{(k_1-k_2)^2-k}{2m^4} \\ +\ldots+\dfrac{1}{m^{2k}}\end{array}\right\}\right| \quad (16)$$

The 'frequency' test result as reported in Table I shows that the number of 1's and 0's in a sequence of the proposed PRBG is approximately equal for randomly chosen input seed. It assesses the closeness of the fraction of ones to 1/2, i.e., the number of 1's and 0's in a sequence should be approximately equal [17]. If the number of 1's and 0's in the sequence of the proposed PRBG method is considered approximately equal, i.e., $k_1 \approx k_2$, then the $dist\,(D_k, U_k)$ can be computed as,

$$dist\,(D_k, U_k) = \frac{1}{2}\left|\left\{-\frac{k}{2m^4} + \frac{k\,(k-2)}{8m^8} - \ldots + \frac{1}{m^{2k}}\right\}\right| \quad (17)$$

If $k = m$ and if m tends to infinite then the $dist\,(D_k, U_k)$ tends to be zero.

$$\lim_{k\to\infty} dist\,(D_k, U_k) = \lim_{k\to\infty} \frac{1}{2}\left|\left\{-\frac{k}{2m^4} + \ldots + \frac{1}{m^{2k}}\right\}\right| = 0 \quad (18)$$

The polynomial equation (17) can be further simplified to measure the statistical distance between the two distributions for $k_1 \approx k_2$ as,

$$dist\,(D_k, U_k) \approx \frac{1}{2}\left|\left\{-\frac{k}{2m^4} + \frac{k\,(k-2)}{8m^8}\right\}\right| \quad (19)$$

According to equation (13), the probability ensembles $\{D_k\}$ and $\{U_k\}$ are called statistical indistinguishability if their statistical difference is negligible, such that $dist\,(D_k, U_k) \leq \varepsilon(k)$. Here, $\varepsilon(k)$ is a negligible function and in standard practice, ε is negligible if $\varepsilon \leq 2^{-80}$ [24]. Therefore, the equation (19) can be further solved for the negligible function.

As an example, $\varepsilon = 2^{-100}$ is considered. Suppose, for generating $k = m = 2^n$ pseudorandom bits, then the value of m and size of n can be calculated for the proposed modified dual-CLCG method by solving the following equation:

$$dist\,(D_k, U_k) \approx \frac{1}{2}\left|\left\{-\frac{k}{2m^4} + \frac{k\,(k-2)}{8m^8}\right\}\right| \leq 2^{-100}$$

By solving the above polynomial equation, the value of $k \approx 2^{32}$ is obtained and therefore, the proposed PRBG method is polynomial time unpredictable with an indistinguishability of 2^{-100} (with a distribution that is at a specified distance from the uniform distribution) for $n \geq 32$ bits.

Two other different cases such as $k_1 = 3k_2$ and $k_2 = 3k_1$ can also be considered to measure the statistical distance $dist\,(D_k, U_k)$ between the two distributions. Therefore, the polynomial equation (16) can be further simplified as,

for $k_1 = 3k_2$:

$$dist\,(D_k, U_k) \approx \frac{1}{2}\left|\left\{\frac{k}{2m^2} + \frac{k(k-4)}{8m^4}\right\}\right| \leq 2^{-100} \quad (20)$$

for $k_2 = 3k_1$:

$$dist\,(D_k, U_k) \approx \frac{1}{2}\left|\left\{-\frac{k}{2m^2} + \frac{k(k-4)}{8m^4}\right\}\right| \leq 2^{-100} \quad (21)$$

Hence, the polynomial equations (20) and (21) can be solved for the negligible value of 2^{-100} and the value of $k \approx 2^{98}$ is obtained in both the cases $k_1 = 3k_2$ and $k_2 = 3k_1$. Therefore, the proposed modified dual-CLCG method is polynomial time unpredictable with an indistinguishability of 2^{-100} for $n \geq 98$ bits. Hence, any probabilistic algorithm for finding the next bit has negligible probability of success.

Now consider the problem of determining the initial seed (x_0, y_0, p_0, q_0) of the proposed modified dual-CLCG method. On knowing $a_1, a_2, a_3, a_4, b_1, b_2, b_3, b_4, m$ and q-bits of the output sequence $(Z_1, Z_2, Z_3, \ldots, Z_q)$ of the proposed modified dual-CLCG method, the initial seed (x_0, y_0, p_0, q_0) can be determined by solving the inequality equations. The same mathematical procedure as followed in [16] applies to the proposed modified dual-CLCG system to solve the inequality equation. Every i^{th} output B_i of CLCG in the proposed PRBG holds an inequality equation which is given below

$$x_i > y_i \quad \text{if } B_i = 1; \quad x_i \leq y_i \quad \text{if } B_i = 0$$

Where x_i and y_i are the i^{th} output of LCG that can be mathematically derived from the initial seeds x_0 and y_0 respectively. Therefore,

$$x_i = a_1^i x_0 + b_1 \sum_{j=0}^{i-1} a_1^j \,(\mathrm{mod}\, m)$$

$$y_i = a_2^i y_0 + b_2 \sum_{j=0}^{i-1} a_2^j \,(\mathrm{mod}\, m)$$

It implies that,

$$a_1^i x_0 + b_1 \sum_{j=0}^{i-1} a_1^j \,(\mathrm{mod}\, m) > a_2^i y_0 + b_2 \sum_{j=0}^{i-1} a_2^j \,(\mathrm{mod}\, m) \quad if\ B_i = 1 \quad (22)$$

$$a_1^i x_0 + b_1 \sum_{j=0}^{i-1} a_1^j \,(\mathrm{mod}\, m) \leq a_2^i y_0 + b_2 \sum_{j=0}^{i-1} a_2^j \,(\mathrm{mod}\, m) \quad if\ B_i = 0 \quad (23)$$

Once the inequality equation for given i^{th} output bit B_i is solved, then it is easy to find the initial seeds in the backward direction. The analysis to solve the inequality equation of CLCG is described in [16, Lemma 2.1–Lemma 2.4] with examples. If q-bits of B_i are known then q number of inequalities, E_i can be set up, where $1 \leq i \leq q$. Each inequality equation E_i will have S_i set of solutions (x_j, y_j). The intersection of all the S_i's for $i \in [1, q]$ gives a small set of possible values for the seed. The number of inequalities that are to be solved to obtain an unique solution (x_0, y_0) is given by $q = 1 + \log_2(|S_1|)$. It means $S_1 \cap S_2 \cap S_3 \cap \ldots \cap S_q$ will give an unique solution (x_0, y_0), where $S_1, S_2, \ldots S_q$ are all the set of solutions (x_j, y_j).

As stated in the *Lemma 2.2* of the CLCG properties, the inequality $a_1 x_i + b_1 \leq a_2 y_i + b_2 (\mathrm{mod}\, m)$ and $a_1 x_i + b_1 > a_2 y_i + b_2 (\mathrm{mod}\, m)$ has $m(m+1)/2$ and $m(m-1)/2$ solutions for (x_i, y_i) respectively, if all four LCGs have the full period [16]. Therefore, the complexity of a brute-force search to find the

unique solution (x_0, y_0) in CLCG is

$$\approx \left(\frac{m^2}{2}\right) \times q = \left(\frac{m^2}{2}\right) \times \log_2\left(m^2\right) = m^2 \times n = n2^{2n},$$

where $q = 1 + \log_2(|S_1|)$ and $n = \log_2(m)$. In the proposed PRBG method, a pair of two inequality equations decides the generation of random bits at the output. Therefore, the initial seed (x_0, y_0, p_0, q_0) can be obtained by solving the inequality pair using the above mathematical procedures. Every i^{th} output Z_i of proposed PRBG holds the possible pair of inequality equations as given below:

$[B_i = 1 \text{ and } C_i = 1]$ or $[B_i = 0 \text{ and } C_i = 0]$ if $Z_i = 0$

$[B_i = 1 \text{ and } C_i = 0]$ or $[B_i = 0 \text{ and } C_i = 1]$ if $Z_i = 1$

It implies that,

$$\{x_i > y_i \text{ and } p_i > q_i\} \text{ or } \{x_i \leq y_i \text{ and } p_i \leq q_i\}, \text{ if } Z_i = 0 \tag{24}$$

$$\{x_i > y_i \text{ and } p_i \leq q_i\} \text{ or } \{x_i \leq y_i \text{ and } p_i > q_i\}, \text{ if } Z_i = 1 \tag{25}$$

Where x_i, y_i, p_i and q_i are the i^{th} output of LCGs derived from the initial seeds x_0, y_0, p_0 and q_0 respectively. If $a_1, b_1, a_2, b_2, a_3, b_3, a_4, b_4, m$ and q-bits of the output bit sequence $(Z_1, Z_2, Z_3, \ldots, Z_q)$ are known then the initial condition or the seeds (x_0, y_0, p_0, q_0) of the proposed method cannot be determined by solving the above inequality equations like CLCG method. It means the proposed PRBG method does not give unique solution (x_0, y_0, p_0, q_0) by intersecting the set of solutions of q number of inequality equations. This statement can be justified by taking a suitable example. Consider the same example explained for the proposed PRBG method in Section-III, where Z_i is evaluated from the initial seeds (2, 7, 3, 4) and assume that B_i and C_i are known. The computed value of B_i, C_i and Z_i are given below,

$B_i = (1, 0, 1, 1, 1, 0, 0, 0)$ and $C_i = (0, 0, 1, 1, 0, 1, 1, 0)$

$Z_i = (1, 0, 0, 0, 1, 1, 1, 0)$

In this case, seven inequality equation pairs corresponding to seven consecutive output bits are needed to get a unique solution. The inequality equation pairs E_i for $1 \leq i \leq q$, are

$\{[(5x_0+5) > (5y_0+3)] \bmod 8; [(5p_0+1) \leq (5q_0+7)] \bmod 8\}$

$\{[(x_0+6) \leq (y_0+2)] \bmod 8; [(p_0+6) \leq (q_0+2)] \bmod 8\}$

$\{[(5x_0+3) > (5y_0+5)] \bmod 8; [(5p_0+7) > (5q_0+1)] \bmod 8\}$

$\{[(x_0+4) > (y_0+4)] \bmod 8; [(p_0+4) > (q_0+4)] \bmod 8\}$

$\{[(5x_0+1) > (5y_0+7)] \bmod 8; [(5p_0+5) \leq (5q_0+3)] \bmod 8\}$

$\{[(x_0+2) \leq (y_0+6)] \bmod 8; [(p_0+2) > (q_0+6)] \bmod 8\}$

$\{[(5x_0+7) \leq (5y_0+1)] \bmod 8; [(5p_0+3) > (5q_0+5)] \bmod 8\}$

The inequality equation $x_i > y_i$ has 28 solutions and $x_i \leq y_i$ has 36 solutions. Similarly, inequality equations $p_i > q_i$ and $p_i \leq q_i$ has 28 and 36 solutions respectively. For the first inequality equations pair, there are $28 \times 36 = 1008$ solutions set and the second combination has $36 \times 36 = 1296$ solutions set. Similarly, the third combination has $28 \times 28 = 784$ solutions set. Therefore, the total numbers of solutions for each pair of inequality equations depend on the four different inequality equation conditions. Let S_i and $|S_i|$ are the solution set and the total number of solutions for each inequality equation pair E_i respectively. As the solutions are evaluated using two different combinations of inequality equations, therefore the total solution for each pair of inequality is $|S_i| = |S_{xy}| \times |S_{pq}|$. Where $|S_{xy}|$ is the total number solutions corresponding to the inequality between x_i and y_i. Similarly, $|S_{pq}|$ is the total number solutions corresponding to the inequality between p_i and q_i. Therefore, it has two cases $Z_i = 1$ and $Z_i = 0$ to find the total number of solutions. If $Z_i = 1$, then there are two possible combinations of inequality equation pairs, either ($x_i > y_i$ and $p_i \leq q_i$) or ($x_i \leq y_i$ and $p_i > q_i$). If $Z_i = 1$, then there are two possible combinations of inequality equation pairs, either ($x_i > y_i$ and $p_i > q_i$) or ($x_i \leq y_i$ and $p_i \leq q_i$).

Lemma 3: If four LCGs have full period, then the inequality equation pair ($x_i > y_i$ and $p_i \leq q_i$) has $\frac{m^2(m^2-1)}{2}$ solutions for (x_i, y_i).

Proof: As stated in [16], if the two LCGs x_i and y_i have full period, the inequality equations $x_i > y_i$ and $x_i \leq y_i$ has $m(m-1)/2$ and $m(m+1)/2$ solutions for (x_i, y_i) respectively. Therefore, by using this property, the total number of solutions for the inequality equation pair ($x_i > y_i$ and $p_i \leq q_i$) can be obtained as $\left(\frac{m(m-1)}{2}\right) \times \left(\frac{m(m+1)}{2}\right) = \frac{m^2(m^2-1)}{2}$. Similarly, the number of solutions for other inequality pairs is defined in the following corollaries.

Corollary 1: If four LCGs have full period, then the inequality equation pairs ($x_i \leq y_i$ and $p_i > q_i$) has $\left(\frac{m(m+1)}{2}\right) \times \left(\frac{m(m-1)}{2}\right) = \frac{m^2(m^2-1)}{2}$ solutions for (x_i, y_i).

Corollary 2: If four LCGs have full period, then the inequality equation pairs ($x_i > y_i$ and $p_i > q_i$) has $\left(\frac{m(m-1)}{2}\right) \times \left(\frac{m(m-1)}{2}\right) = \frac{m^2(m-1)^2}{4}$ solutions for (x_i, y_i).

Corollary 3: If four LCGs have full period, then the inequality equation pairs ($x_i \leq y_i$ and $p_i \leq q_i$) has $\left(\frac{m(m+1)}{2}\right) \times \left(\frac{m(m+1)}{2}\right) = \frac{m^2(m+1)^2}{4}$ solutions for (x_i, y_i).

case-1. If $Z_i = 1$: There are two possible combinations of inequality equation pairs, either ($x_i > y_i$ and $p_i \leq q_i$) or ($x_i \leq y_i$ and $p_i > q_i$). In both combinations there are $\frac{m^2(m^2-1)}{2}$ solutions.

case-2. If $Z_i = 0$: There are two possible combinations of inequality equation pairs, either ($x_i > y_i$ and $p_i > q_i$) or ($x_i \leq y_i$ and $p_i \leq q_i$). For ($x_i > y_i$ and $p_i > q_i$) inequality equation pair, it has $\frac{m^2(m-1)^2}{4}$ solutions. Similarly, for ($x_i \leq y_i$ and $p_i \leq q_i$) inequality equation pair, it has $\frac{m^2(m+1)^2}{4}$ solutions.

Let S_i be the solution set for each inequality pair E_i. In the previously mentioned example, there are seven solution sets $S_1, S_2, \ldots S_7$ for the corresponding inequality equation $E_1, E_2, \ldots E_7$. After solving seven inequality pairs, the intersection of sets S_1 through S_7 is $S_1 \cap S_2 \cap S_3 \cap \ldots \cap S_7 = \{(2, 7, 3, 4)\}$. The number of solutions with each step diminishes as $1008 \rightarrow 323 \rightarrow 66 \rightarrow 18 \rightarrow 5 \rightarrow 3 \rightarrow 1$.

Therefore, if the internal states of the comparative LCGs are known, then the complexity of a brute-force search to find

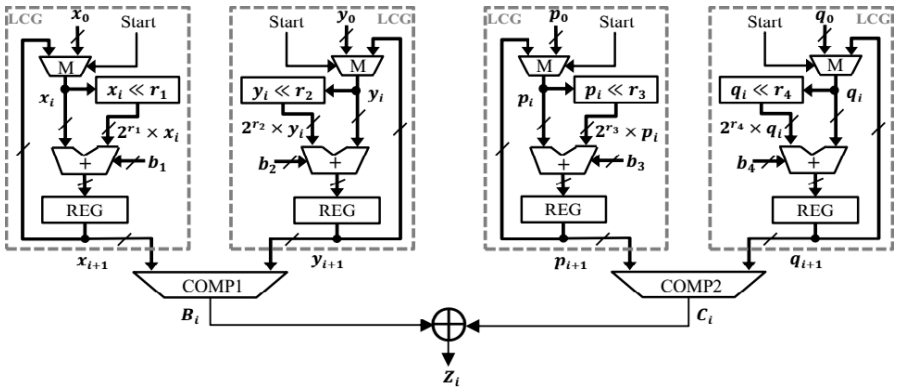

Fig. 5. Proposed architecture of the modified dual-CLCG method.

the unique solution (x_0, y_0, p_0, q_0) in the proposed modified dual-CLCG method is $\approx \frac{m^4}{4} \times q = \frac{m^4}{4} \times \log_2\left(m^2\right) \approx n2^{4n}$, where $q = \log_2\left(|S_{xy}|\right)$ and $n = \log_2(m)$. If the comparator output B_i and C_i in the proposed modified dual-CLCG method are not known corresponding to the output Z_i, then it is not possible to compute the unique solution (x_0, y_0, p_0, q_0) with above theoretical procedures. For each value of Z_i, there are two possible combinations of inequality equation pairs. If we combine the solution sets of each possible inequality pair and further intersect with similar solution sets corresponding to other value of Z_i, then it gives 18 different solutions at the end of the steps. The number of solutions with each step diminishes as $2016 \rightarrow 1020 \rightarrow 488 \rightarrow 256 \rightarrow 120 \rightarrow 46 \rightarrow 26 \rightarrow 18$. It means even after considering all the inequality pairs corresponding to every output bits Z_i, $1 \leq i \leq 2^n$, a brute-force search is not able to find the unique solution for the proposed method.

Therefore, the probabilistic algorithm to obtain the initial seed of the proposed modified dual-CLCG method requires the solution of $n2^{4n}$, if B_i and C_i are known. If the intermediate states B_i and C_i are unknown then it is computationally infeasible to find the initial seed (x_0, y_0, p_0, q_0) for large m, where $m = 2^n$. Therefore, the dual coupling of LCGs and use of XOR logic as post-processing in the proposed modified dual-CLCG method enhance the security strength in the order of $O(n2^{4n})$ as compared to existing dual-CLCG method. Thus, the complexity of a brute-force search to break the proposed modified dual-CLCG system is $n2^{4n}$, whereas it is $n2^{2n}$ for existing dual-CLCG method.

IV. Proposed Architecture of the Modified Dual-CLCG Method and Its Complexity Analysis

This section presents the new efficient VLSI architecture of the proposed modified dual-CLCG method that generates pseudorandom bit at every uniform clock cycle. The first order architecture of the proposed modified dual-CLCG method is shown in Fig. 5 which is developed by mapping the four LCG equations, two inequality equations and one modulo-2 addition. The LCG block is the basic functional structure in the proposed modified dual-CLCG architecture and it is highlighted with dotted box in Fig. 5. The architectural design of LCG block mapped from LCG equation has discussed in the Section-II-A. It involves logical shift operation instead of multiplication and three-operand addition to computes n-bit binary random output on every clock cycle. The proposed modified dual-CLCG architecture consists of four LCG blocks that computes x_{i+1}, y_{i+1}, p_{i+1} and q_{i+1} from x_0, y_0, p_0 and q_0 respectively. The two inequality equations are realized with the two n-bit binary comparators that compare the n-bit binary output x_{i+1} with y_{i+1} and p_{i+1} with q_{i+1} at the same clock time, produces one-bit output B_i and C_i respectively. Further, the modulo-2 addition of the two comparator outputs are realized with XOR logic as shown in Fig. 5 that computes final random bit at every clock rate. Therefore, unlike dual-CLCG architecture which demands complex hardware circuits (buffer, data control and memory), this proposed architecture utilizes only single XOR logic at the output stage.

A. Complexity Analysis of the Proposed Architecture

The LCG block used in the proposed architecture takes the maximum combinational path delay which is contributed by the combination of one adder and multiplexer delay. The proposed architecture of the modified dual-CLCG method consumes an area of four 2x1 n-bit multiplexers, four n-bit registers, four n-bit three-operand modulo-2^n adders and one 1-bit XOR gate. Therefore, the area and the maximum combinational path delay of the proposed modified dual-CLCG architecture are evaluated as follows,

$$\text{Area, } \mathrm{A_{prop}} = 4\left(\mathrm{A_{3oa}} + \mathrm{A_{mux}} + \mathrm{A_{reg}}\right) + 2\mathrm{A_{cmp}} + \mathrm{A_X}$$
$$\text{Critical path, } \mathrm{T_{prop}} = \mathrm{T_{3oa}} + \mathrm{T_{mux}}$$

Here, $\mathrm{A_{3oa}}$ and $\mathrm{T_{3oa}}$ represents the area and critical delay of the three-operand adder. The performance of the proposed architecture depends on the efficient implementation of the three-operand adder and the binary comparator. The carry save adder (CSA) is the most efficient and widely adopted adder technique to perform the three-operand modulo-2^n addition [25]. Therefore, the three-operand modulo-2^n adder in the proposed architecture is implemented by using the carry-save adder which is shown in Fig. 6. In carry-save adder, the three-operand addition is performed in two stages. The first stage is the array of full adders and each of them performs bit wise addition that computes sum and carry bit signal. The second stage is ripple carry adder that computes final sum signal. The area ($\mathrm{A_{3CSA}}$) and critical path delay ($\mathrm{T_{3CSA}}$) of carry

TABLE III
AREA AND TIME COMPLEXITY

Performance / Method	Timing Complexity			Area Complexity
	Initial clock latency (input to first output)	Critical Path	Output Latency	
BBS [12]	$2n$+5 clock	$2(T_{2oa} + T_{mux})$	$2n$+5 clock	$2A_{3oa} + A_{2oa} + A_{mux4} + 3A_{mux2} + 4A_{reg} + A_{FSM}$
CLCG	1 clock	$T_{3oa} + T_{mux}$	1 clock	$2A_{LCG} + A_{cmp} = 2(A_{3oa} + A_{mux} + A_{reg}) + A_{cmp}$
Dual-CLCG	2^n clock	$T_{3oa} + T_{mux}$	2 clock	$4(A_{3oa} + A_{mux} + A_{reg}) + 2A_{cmp} + A_{tri} + A_{mem} + A_{FSM}$
Proposed PRBG Method	1 clock	$T_{3oa} + T_{mux}$	1 clock	$4(A_{3oa} + A_{mux} + A_{reg}) + 2A_{cmp} + A_X$

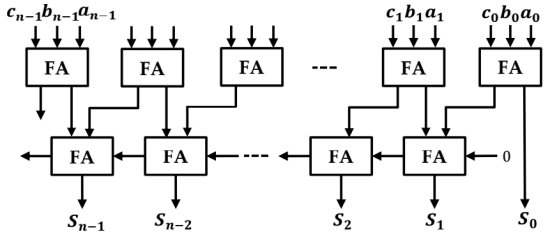

Fig. 6. Three-operand modulo-2^n carry-save adder.

save adder are evaluated as follows,

$$\text{Area, } A_{3CSA} = (2n-1)\,A_{FA} = (2n-1)\,(2A_X+3A_G)$$
$$\text{Critical path, } T_{3CSA} = nT_{FA} = 3T_X + 2(n-1)T_G$$

Similarly, the binary comparator in the proposed modified dual-CLCG architecture is implemented by the magnitude comparator which is the most common comparator technique [26]. The n-bit binary comparator is designed using the 2-bit magnitude comparator (see Fig. 7(a)). The logic diagram of 2-bit magnitude comparator is shown in Fig. 7(b) that computes A > B (A_{big}) and A < B (B_{big}) signals by comparing two 2-bit binary operands. The number of 2-bit comparator stages for the n-bit magnitude comparator is $(\log_2 n + 1)$ and therefore, the critical path delay is in the order of $O(\log_2 n)$. Hence, the overall area (A_{cmp}) and critical path delay (T_{cmp}) of the magnitude comparator are evaluated as follows,

$$\text{Area, } A_{cmp} = (n-1)\,[9A_G + 4A_N]$$
$$\text{Critical path, } T_{cmp} = 4(\log_2 n)T_G$$

Therefore, the overall area and critical path delay of the proposed architecture of the modified dual-CLCG method can be evaluated as follows,

$$\text{Area, } A_{MDCLCG} = 4\left(A_{3oa} + A_{mux} + A_{reg}\right) + 2A_{cmp} + A_X = (16n-7)\,A_X + 2(27n-15)A_G + 12A_N + 4nA_{FF}$$
$$\text{Critical path, } T_{MDCLCG} = T_{3oa} + T_{mux} = 3T_X + 2nT_G$$

Here, A_G, A_X, A_N and A_{FF} are denoted as area of 2-input basic gate (AND/NAND/OR/NOR), XOR/XNOR gate, NOT gate and flipflop respectively. T_G and T_X denotes the delay of 2-input basic gate (AND/NAND/OR/NOR) and XOR/XNOR gate respectively. Table III summarizes and compares the area and time complexity of the proposed architecture of the modified dual-CLCG method with the architecture of other existing PRBG methods. It reports that

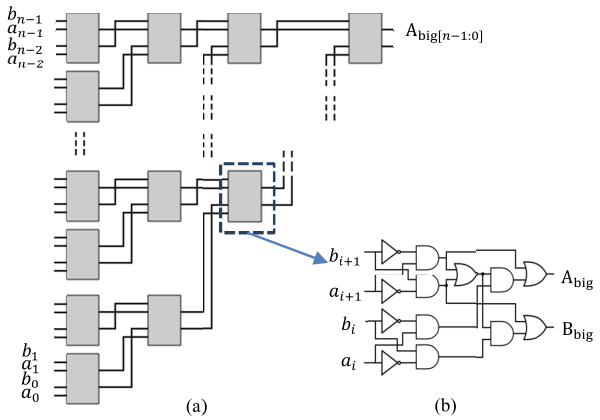

Fig. 7. Magnitude comparator (a) n-bit, (b) Logic diagram of 2-bit comparator.

the critical path delay in all the LCG based methods depend on the three-operand adder circuit. The area of the proposed architecture is considerably less than the dual-CLCG architecture. It generates a one-bit random output at every clock cycle with one initial clock latency. Whereas, dual-CLCG architecture takes 2^n initial clock latency (input to first output delay) to give the first output bit and further, it takes two clock cycles (output to output delay/output latency) to generate one-bit random output. On the other hand, the architecrure of BBS method [12] has the large output latency of $2n$+5 clock cycles due to the use of iterative Montgomery modular multiplier.

V. FPGA PROTOTYPE AND REAL-TIME VALIDATION

The hardware architecture of the existing dual-CLCG and the proposed modified dual-CLCG methods for different word size of n = 8-, 16- and 32- bit are designed using Verilog HDL and prototyped on two different commercially available FPGA devices, such as Spartan3E XC500E and Virtex5 XC5VLX110T to study their resource utilizations. The laboratory experimental setup of the proposed architecture on Virtex5 XC5VLX110T FPGA device is shown in Fig. 8. The real-time pseudorandom bit sequence is captured through USB-JTAG programming cable and displayed on the PC using Xilinx Chipscope analyzer tool. The real-time implementation is performed at the clock frequency of 120 MHz. This clock is provided externally to the FPGA device by using arbitrary function generator (AFG 3252).

A. Real-Time Validation of the Dual-CLCG Architecture

To highlight the initial clock latency (input to first output latency) in the architectural mapping of the existing

TABLE IV
PHYSICAL RESULTS COMPARISON

Methods	Size	Total FFs /Latches		Number of Slices		Number of LUTs		Power at max freq. (in mW)		Power/Freq. (in mW/MHz)		Max. frq. (f_{max}) (in MHz)		T_{clk} ($1/f_{max}$) (in ns)		Initial clock latency (in T_{clk})	o/p to o/p latency (in T_{clk})	NIST Test
		Virtex5	Spartan 3E	Virtex5	Spartan 3E	Virtex5	Spartan 3E	Virtex5	Spartan 3E	Virtex5	Spartan 3E	Virtex5	Spartan 3E	Virtex5	Spartan 3E			
CLCG	32-bit	64	64	134	198	336	373	34.80	17.29	0.1572	0.1629	221.43	106.09	4.516	9.426	01	01	
	16-bit	32	32	50	86	150	166	25.75	13.79	0.0946	0.1054	272.18	130.78	3.674	7.646	01	01	**Fail**
	8-bit	16	16	09	42	36	80	18.02	10.86	0.0491	0.0557	366.84	194.74	2.726	5.135	01	01	
Dual-CLCG	32-bit	**Fail**	**Fail**	**Fail**	**Fail**	**Fail**	**Fail**	**Fail**	**Fail**	**Fail**	**Fail**	**Fail**	**Fail**	**Fail**	**Fail**	2^{32}	02	
	16-bit	**Fail**	**Fail**	**Fail**	**Fail**	**Fail**	**Fail**	**Fail**	**Fail**	**Fail**	**Fail**	**Fail**	**Fail**	**Fail**	**Fail**	2^{16}	02	**Fail**
	8-bit	198	198	199	345	381	414	44.43	25.37	0.1211	0.1303	366.84	194.74	2.726	5.135	2^{8}	02	
Proposed PRBG	32-bit	128	128	248	385	646	737	59.56	29.22	0.2689	0.2754	221.43	106.09	4.516	9.426	01	01	
	16-bit	64	64	93	167	284	320	39.28	19.74	0.1443	0.1509	272.18	130.78	3.674	7.646	01	01	Pass
	8-bit	32	32	22	71	71	135	25.64	14.02	0.0699	0.0719	366.84	194.74	2.726	5.135	01	01	

Fig. 8. Laboratory experimental setup of the proposed architecture of the modified dual-CLCG method of 32-bit word size.

dual- CLCG method, 8-bit design is implemented on the targeted Virtex5 FPGA device and the captured pseudorandom bit sequence on real-time is shown in Fig. 9(c) that validates the behavioral simulation result of Fig. 9(b). The 8-bit architecture is designed with the constant values of $a_1 = 5, b_1 = 1, a_2 = 5, b_2 = 3, a_3 = 9, b_3 = 141, a_4 = 33, b_4 = 79, m = 2^8$ and initial seeds of $(x_0, y_0, p_0, q_0) = (1, 2, 14, 3)$ which generates the sequence as follows,

$$Z_i = [0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 1\ \ 0\ \ 1\ \ 0\ \ 1\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 1 \ldots]$$

It generates a maximum of $2^{n-1} = 2^7 = 128$ random bits and thereafter, the sequence is repeated. Therefore, 1×128 bit size of memory (128 FFs) is required to store the generated bits until it reaches to maximum length period of controller-CLCG. It takes $2^8 = 256$ initial clock cycles as shown in Fig. 9(a) to store the generated random output bits in memory. After 256 clock cycles, it releases the stored 128 bits serially in every 2 clock cycles (1-bit per 2 clock cycles).

B. Real-Time Validation of the Proposed Modified Dual-CLCG Architecture

The proposed architecture of the modified dual-CLCG method is implemented on the same targeted device Virtex5 FPGA to validate the results. For the readers' convenience, the behavioral simulation result (see Fig. 10(a)) of the 8-bit design is validated with Xilinx Chipscope result (see Fig. 10(b)). Further, the 32-bit architecture of the proposed modified dual-CLCG is also validated on the same platform.

The proposed architecture of 8-bit word size is designed with the constant values of $a_1 = 5, b_1 = 1, a_2 = 5, b_2 = 3, a_3 = 9, b_3 = 141, a_4 = 33, b_4 = 79, m = 2^8$ and initial seeds of $(x_0, y_0, p_0, q_0) = (1, 2, 14, 3)$ which generates the sequence as follows,

$$Z_i = [0\ \ 1\ \ 0\ \ 0\ \ 1\ \ 1\ \ 0\ \ 1\ \ 0\ \ 0 \ldots]$$

It generates a maximum of $2^n = 2^8 = 256$ random bits and thereafter, the sequence is repeated. It takes only one clock cycle to generate one random bit with one initial clock latency. Fig. 10(a) shows the behavioral simulation result of the proposed architecture for $n = 8$- bit word size and the real-time captured pseudorandom bit sequence using Xilinx Chipscope analyzer is shown in Fig. 10(b) that validates the behavioral simulation result of Fig. 10(a).

Similarly, the proposed architecture of 32-bit word size is designed with the constant values of $a_1 = 65, b_1 = 117, a_2 = 16385, b_2 = 221, a_3 = 4097, b_3 = 21359, a_4 = 1025, b_4 = 533, m = 2^{32}$ and initial seeds of $(x_0, y_0, p_0, q_0) = (5183, 91356, 39771, 7392)$ which generates the sequence as follows,

$$Z_i = [1\ \ 0\ \ 0\ \ 1\ \ 1\ \ 0\ \ 1\ \ 0\ \ 1\ \ 0 \ldots]$$

It generates a maximum of $2^n = 2^{32} = 4294967296$ random bits and thereafter, the sequence is repeated. Fig. 11(a) shows the behavioral simulation result of the proposed architecture for $n = 32$- bit word size and the real-time captured pseudo-random bit sequence using Xilinx Chipscope analyzer is shown in Fig. 11(b) that validates the behavioral simulation result of Fig. 11(a).

VI. PHYSICAL IMPLEMENTATION RESULTS

The physical implementation results of 8-, 16- and 32- bit size architectures of the proposed modified dual-CLCG method along with other existing PRBG methods such as

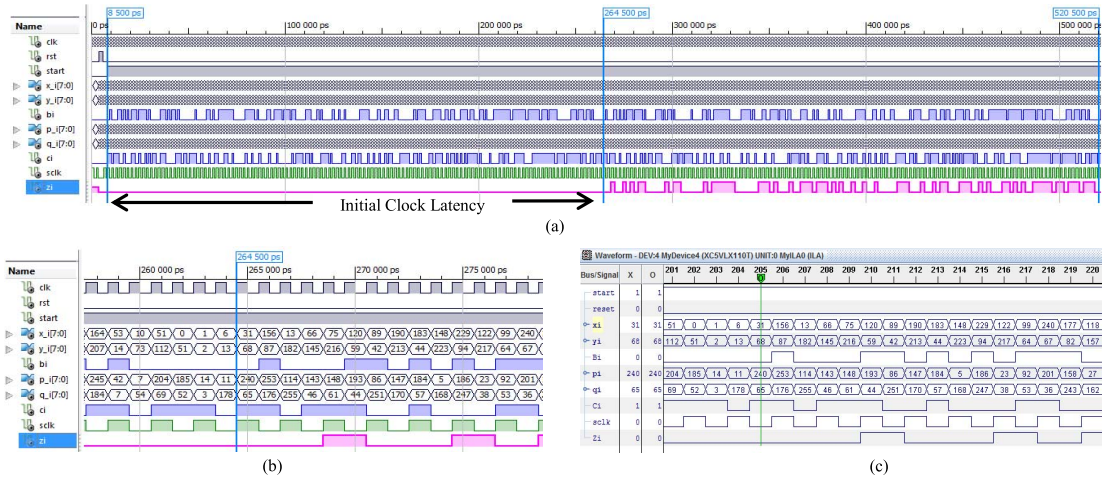

Fig. 9. (a) Behavioral simulation results, (b) Zoom view of (a) and (c) Real-time captured Chipscope result of the dual-CLCG architecture for $n = 8$- bit.

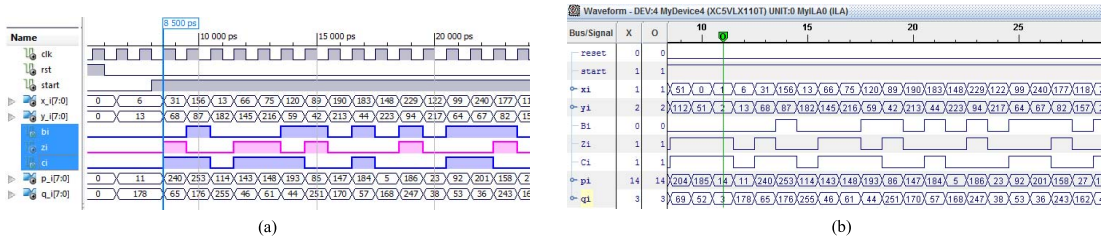

Fig. 10. (a) Behavioral simulation result and (b) Real-time captured Chipscope result of the proposed modified dual-CLCG architecture for $n = 8$- bit.

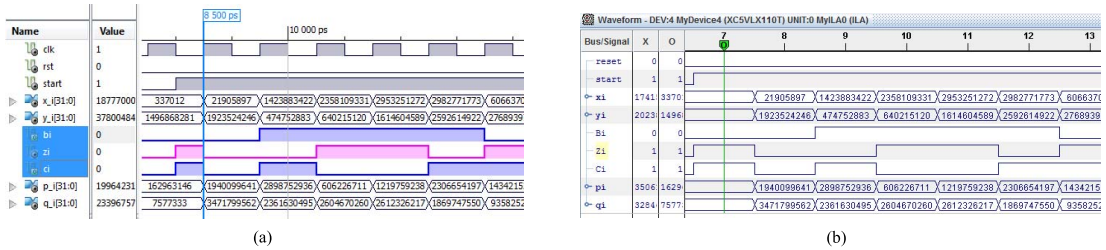

Fig. 11. (a) Behavioral simulation result and (b) Real-time captured Chipscope result of the proposed modified dual-CLCG architecture for $n = 32$-bit.

dual-CLCG and CLCG are highlighted in Table IV. The physical implementation results in Table IV report that the proposed architecture of the modified dual-CLCG method consumes 83.8%, 88.9% and 81.4% less flip-flops, slices and LUTs respectively as compared to the dual-CLCG architecture on Virtex5 FPGA chip for 8-bit design. Similarly, it utilizes 42.3% less power than the dual-CLCG architecture on Virtex5 for 8-bit design. The 8-bit dual-CLCG architecture takes 2^8 initial clock cycles (input to first output latency) to get the first pseudorandom bit and further, it takes two clock cycles (output to output latency) to generate each pseudorandom bit. Whereas, the proposed modified dual-CLCG method takes only one clock cycle to generate one random bit with one initial clock latency. The proposed architecture has the same maximum frequency of CLCG and dual-CLCG architecture because the critical path delay in all three methods depend on the three-operand adder circuit. The implementation of the dual-CLCG architecture failed for the 16- and 32- bit word size on both FPGA devices due to excessive usage of memory ($<2^{15}$) resources on FPGA chip. For 32- bit dual-CLCG architecture, 1×2^{31} bits memory (2^{31} flip-flops) is required which is limited to 69,120 CLB flip-flops and 5,328 Kbits BRAM on Virtex5 XC5VLX110T chip. Similarly, the memory is limited to 9,312 CLB flip-flops and 360 Kbits BRAM on Spartan3E XC3S500E chip. On the other hand, the proposed architecture of the modified dual-CLCG method can be implemented on the commercially available low-end FPGA devices. Therefore, the proposed modified dual-CLCG method is the efficient PRBG method for generating pseudorandom bit at every uniform clock rate because of its advantages such as good randomness, maximal period, less area, low power, low output latency and low initial clock latency.

VII. Conclusion

Dual-CLCG method involves dual coupling of four LCGs that makes it more secure than LCG based PRBGs. However, it is reported that this method has the drawback of generating

pseudorandom bit at non-uniform time interval as it works on the few inequality cases. Hence, a new hardware architecture of the existing dual-CLCG method is developed that generates the pseudorandom bit at uniform clock rate. But, it leads to some other drawbacks such as high initial clock latency of 2^n for n- bit architecture, large memory consumptions and fails to achieve the maximal period of 2^n. Further, to overcome the aforesaid drawbacks, a new modified dual-CLCG method and its architecture are proposed that generate the pseudorandom bits of a maximum length of 2^n at one-bit per clock cycle with one initial clock delay. In this architecture, only a single XOR logic is utilized at the output stage instead of complex hardware circuits (buffer, data control and memory) used in previous dual-CLCG architecture. Thus, the hardware complexity of the proposed architecture of the new modified dual-CLCG method is significantly reduced. Moreover, the proposed modified dual-CLCG method for the size of $n \geq 24$- bits pass all the fifteen benchmark tests of NIST standard with a high degree of consistency and it is polynomial time unpredictable with an indistinguishability of $\varepsilon = 2^{-100}$ for $n \geq 32$- bits. The proposed architecture of the modified dual-CLCG method is prototyped on the commercially available FPGA devices and the results are captured in real-time using Xilinx chipscope for validation. Based on the performance analysis in terms of hardware complexity, randomness and security, it is observed that 32- bit hardware architecture of the proposed modified dual-CLCG method is optimum and can be useful in the area of hardware security and IoT applications.

REFERENCES

[1] 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.

[2] Q. Zhang, L. T. Yang, and Z. Chen, "Privacy preserving deep computation model on cloud for big data feature learning," *IEEE Trans. Comput.*, vol. 65, no. 5, pp. 1351–1362, May 2016.

[3] E. Fernandes, A. Rahmati, K. Eykholt, and A. Prakash, "Internet of Things security research: A rehash of old ideas or new intellectual challenges?" *IEEE Secur. Privacy*, vol. 15, no. 4, pp. 79–84, 2017.

[4] M. Frustaci, P. Pace, G. Aloi, and G. Fortino, "Evaluating critical security issues of the IoT world: Present and future challenges," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2483–2495, Aug. 2018.

[5] E. Zenner, "Cryptanalysis of LFSR-based pseudorandom generators—A survey," Univ. Mannheim, Mannheim, Germany, 2004. [Online]. Available: http://orbit.dtu.dk/en/publications/cryptanalysis-of-lfsrbased-pseudorandom-generators–a-survey(59f7106b-1800-49df-8037-fbe9e0e98ced).html

[6] J. Stern, "Secret linear congruential generators are not cryptographically secure," in *Proc. 28th Annu. Symp. Found. Comput. Sci.*, Oct. 1987, pp. 421–426.

[7] D. Xiang, M. Chen, and H. Fujiwara, "Using weighted scan enable signals to improve test effectiveness of scan-based BIST," *IEEE Trans. Comput.*, vol. 56, no. 12, pp. 1619–1628, Dec. 2007.

[8] L. Blum, M. Blum, and M. Shub, "A simple unpredictable pseudo-random number generator," *SIAM J. Comput.*, vol. 15, no. 2, pp. 364–383, 1986.

[9] W. Thomas Cusick, "Properties of the x2 mod N pseudorandom number generator," *IEEE Trans. Inf. Theory*, vol. 41, no. 4, pp. 1155–1159, Jul. 1995.

[10] C. Ding, "Blum-Blum-Shub generator," *IEEE Electron. Lett.*, vol. 33, no. 8, p. 667, Apr. 1997.

[11] A. Sidorenko and B. Schoenmakers, "Concrete security of the Blum-Blum-Shub pseudorandom generator," in *Cryptography and Coding* (Lecture Notes in Computer Science), vol. 3796. Berlin, Germany: Springer, Nov. 2005, pp. 355–375.

[12] A. K. Panda and C. K. Ray, "FPGA prototype of low latency BBS PRNG," In *Proc. IEEE Int. Symp. Nanoelectron. Inf. Syst. (INIS)*, Indore, India, Dec. 2015, pp. 118–123.

[13] P. P. Lopez and E. S. Millan, "Cryptographically secure pseudorandom bit generator for RFID tags," in *Proc. Int. Conf. Internet Technol. Secured Trans.*, London, U.K., vol. 11, Nov. 2010, pp. 1–6.

[14] R. S. Katti and R. G. Kavasseri, "Secure pseudo-random bit sequence generation using coupled linear congruential generators," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Seattle, WA, USA, May 2008, pp. 2929–2932.

[15] S. Raj Katti and S. Srinivasan, "Efficient hardware implementation of a new pseudo-random bit sequence generator," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Taipei, Taiwan, May 2009, pp. 1393–1396.

[16] R. S. Katti, R. G. Kavasseri, and V. Sai, "Pseudorandom bit generation using coupled congruential generators," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 57, no. 3, pp. 203–207, Mar. 2010.

[17] 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: http://csrc.nist.gov/publications/nistpubs/800-22-rev1.pdf

[18] *Random Integer Generator*. Accessed: Feb. 20, 2018. [Online]. Available: https://www.random.org/integers

[19] T. Addabbo, M. Alioto, A. Fort, A. Pasini, S. Rocchi, and V. Vignoli, "A class of maximum-period nonlinear congruential generators derived from the Rényi chaotic map," *IEEE Trans. Circuits System. I, Ref. Papers*, vol. 54, no. 4, pp. 816–828, Apr. 2007.

[20] J. Massey, "Shift-register synthesis and BCH decoding," *IEEE Trans. Inf. Theory*, vol. IT-15, no. 1, pp. 122–127, Jan. 1969.

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

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

[23] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*. Boca Raton, FL, USA: CRC Press, 2008.

[24] M. Luby, *Pseudo Randomness and Cryptographic Applications*. Princeton, NJ, USA: Princeton Univ. Press, 1996.

[25] T. Kim, W. Jao, and S. Tjiang, "Circuit optimization using carry-save-adder cells," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 17, no. 10, pp. 974–984, Oct. 1998.

[26] S.-W. Cheng, "A high-speed magnitude comparator with small transistor count," in *Proc. ICECS*, vol. 3, 2003, pp. 1168–1171.

Amit Kumar Panda received the M.Sc. degree in electronics from Berhampur University, India, in 2004, and the M.Tech. degree in electronic design and technology from Tezpur Central University, India, in 2009. He is currently pursuing the Ph.D. degree with the Electrical Engineering Department, Indian Institute Technology Patna, India. He served as an Assistant Professor with the Electronics and Communication Engineering Department, Guru Ghasidas Vishwavidyalaya, India, from 2009 to 2012. He was an Assistant Professor with the Centre for Nanotechnology, Central University of Jharkhand, India, from 2012 to 2013. His research interests include very large-scale integration (VLSI) architectural design, VLSI cryptography, and FPGA-based system design.

Kailash Chandra Ray received the B.E. degree in electrical engineering from the Orissa Engineering College, Bhubaneswar, India, in 1997, the M.Tech. degree in electrical engineering from NIT, Jamshedpur, India, in 2000, and the Ph.D. degree in electronics and electrical communication engineering from IIT Kharagpur, India, in 2009. He was a Technical Member Staff (Design Engineer) with CMOS Chips, Bangalore, India, during 2001–2003. He also served as a Lecturer with IIIT Allahabad, India, from 2008 to 2010. He was an Assistant Professor with the Department of Electrical Engineering, Indian Institute Technology Patna, India, from 2010 to 2018, where he has been an Associate Professor since 2018. His research interests include very large-scale integration (VLSI) architectural design, VLSI signal processing, digital VLSI design, hardware design methodologies, FPGA-based system design, CORDIC, and embedded system.