International Journal of Scientific Engineering and Research (IJSER)
ISSN (Online): 2347-3878
Impact Factor (2024): 7.741

Design of Modified Dual-CLCG Algorithm for Pseudo-Random Bit Generator Using Carrysave Adder

K. Shiva Kumar[1], Ch. Anil kumar[2]

[1]P. G Scholar Vaagdevi College of Engineering, Bollikunta Warangal, Telangana, India

[2]Assisant Professor Vaagdevi College of Engineering, Bollikunta Warangal, Telangana, India

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 congenital 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 it's 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.***

Keywords: Pseudorandom bit generator (PRBG), LCG, VLSI architecture

1. 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. 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. 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 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. Numerous studies on PRBG based on LFSR, 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 generators because of its large prime factorize problem. 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 and. Most of them either consume a large amount of hardware area or high clock latency to mitigate it, a low hardware complexity coupled LCG (CLCG) has been proposed. 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. Despite an improvement in the security, the CLCG method fails the discrete Fourier transform (DFT) test and five other major NIST statistical tests. DFT test finds the periodic patterns in CLCG which shows it as a weak generator. To amend this, Katti et al. 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 2n for n-bit architecture, fails to achieve the maximum length period of 2n (it depends on the number of 0's in the CLCG sequence and is nearly2n−1for 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.

This paper is organized as follows: architectural mapping of the existing dual-CLCG method is performed and The proposed PRBG method along with its randomness properties are discussed. He efficient VLSI architecture of the proposed modified dual-CLCG method. Combined linear congruential generators, as the name implies, are a type of PRNG (pseudorandom number generator) that combine two or more LCGs (linear congruential generators). The combination of two or more LCGs into one random number generator can result in a marked increase in the period length of the generator which makes them better suited for simulating more complex systems. The combined linear congruential generator algorithm is defined as:

$$\mathrm{Xi} \equiv \left(\sum_{j=1}^{k} (-1)^{j-1} Y_{ij}\right) (\bmod\ (m1 - 1))$$

Where m1m1 is the modulus of the LCG, Yi, j Yi, j is the ith input from the jth LCG and XiXi is the ith random generated value. L 'Ecuyer describes a combined linear generator that utilizes two LCGs in Efficient and Portable Combined Random Number Generators for 32-bit processors.

2. Literature Survey

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. J. Stern, "Secret linear congruential generators are not cryptographically secure," The dual-coupled-linear congruential generator (LCG) (dual-CLCG) is a secure pseudorandom bit generator (PRBG) method among various linear feedback shift register (LFSR), LCG, and chaotic-based PRBG methods for generating a pseudorandom bit sequence. The hardware implementation of this method has a bottleneck due to the involvement of inequality equations. Initially, a direct architectural mapping of the dual-CLCG method is performed. Since two inequality equations are involved for coupling, it generates pseudorandom bit at unequal interval of time that leads to large variation in output latency. In addition, it consumes a large area and fails to achieve the maximal period. Hence, to overcome the aforesaid drawbacks, a new efficient PRBG method, i. e., "coupled-variable input LCG (CVLCG), " and its architecture are proposed. The novelty of the proposed method is the coupling of two newly formed variable input LCGs that generates pseudorandom bit at every uniform clock rate, attains maximum length sequence, and reduces one comparator area as compared to the dual-CLCG architecture. The proposed architecture is implemented using Verilog-HDL and prototyped on the commercially available field-programmable gate array (FPGA) device. Furthermore, the sequences are captured through the logic analyzer and evaluated for randomness using the National Institute of Standard and Technology (NIST) standard test tool. The experimental result reports that the proposed PRBG method passes all the randomness tests with a high degree of consistency.

3. 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 and is defined mathematically as follows:

$x_{i+1} = a_1 \times x_i + b_1 \bmod 2^n$ (1)

$y_{i+1} = a_2 \times y_i + b_2 \bmod 2^n$ (2)

$p_{i+1} = a3 \times p_i + b3 \bmod 2^n$ (3)

$q_{i+1} = a4 \times q_i + b4 \bmod 2^n$ (4)

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

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

$$\mathrm{Zi} = \mathrm{Bi} \text{ if } \mathrm{Ci} = 0 \quad (6)$$

Where,

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

Here, a1, b1, a2, b2, a3, b3, a4 and b4 are the constant parameters; x0, y0, p0 and q0 are the initial seeds. Following are the necessary conditions to get the maximum period.

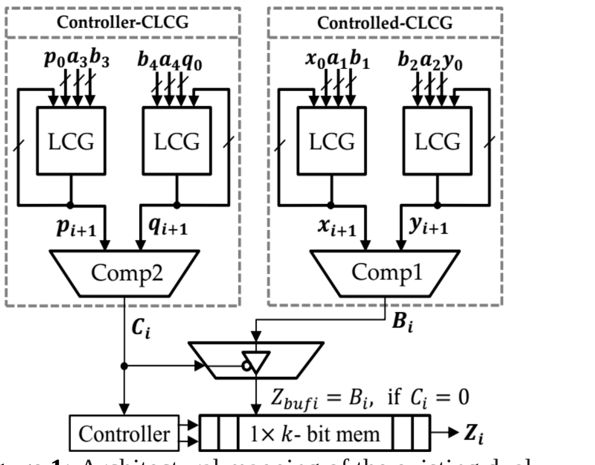

Figure 1: Architectural mapping of the existing dual-CLCG method

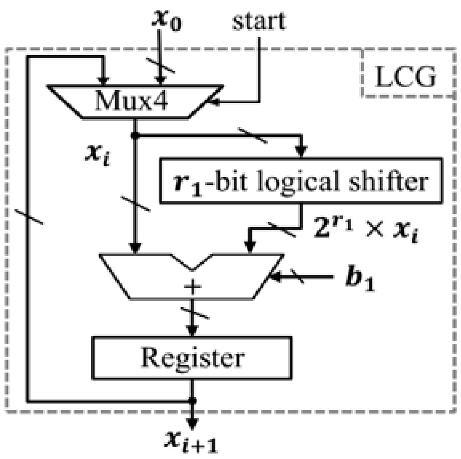

Figure 2: Architecture of the linear congruential generator.

(i) b_1, b_2, b_3 and b_4 are relatively prime with2n (m).
(ii) (a_1-1), (a_2-1), (a_3-1) and (a_4-1) must be divisible by4.

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.

Architecture Mapping of the Existing Dual-CLCG Method

The scope of the work presented in 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-LCG 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 xi+1, the equation (1) can be rewritten as,

$$xi+1 = (a1 \times xi + b1) \bmod 2^n = [(2^{r1}+ 1) xi + b1] \bmod 2^n$$
$$= [(2^{r1} \times xi) +xi+b1] \bmod 2^n$$

The architecture of LCG shown in Fig.2 is implemented with a 3-operand modulo 2n 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_iB_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 consecutive1'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 pseudo random 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 = (0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1);
and
B_i = (1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1);
then,
Z_i= B_i if C_i=0; Z_{bufi}= (1, x, x, 1, x, 0, 0, x, 1, x, 0, x)

In this example, it is observed that the number of 0's and 1'sare equal in every 4-bit patterns. There are two 0's and two1'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,
Ci= (**0**, 1, 1, **0**, 1, **0**, **0**, 1, **0**, 1, **0**, 1);
And
Bi = (**1**, 0, **1**, **1**, **0**, 0, 0, **1**, **0**, 0, 1, 0, **1**, 1, **1**, 1);
then,
Zbu f i= (1, **x**, 1, 1, 0, **x**, **x**, 1, 0, **x**, **x**, **x**, 1, **x**, 1, **x**)

No such uniform pattern can be observed in the sequence Ci 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 (termedas1 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 2n-clock cycles. It can utilize k-flip-flops to store k different bits of B_i while C_i '0'. After 2n clock cycles, it releases these stored bits serially at every two clock cycles (1-bitper two clock cycles). Here, it is assumed that the number of 0's number of 1's 2n−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 2n−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 2n−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,
Area, ADCLCG = 4ALCG +2Acmp+Atri +Amem+Acntrl
Critical path, $T_{DCLCG} = T_{add} + T_{mux}$

Drawbacks of the Existing Dual-CLCG Method and Its Architecture

Although the dual-CLCG method is secure when compared with CLCG and single LCG, 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 2n−1 for randomly chosen n-bit input seed, therefore it needs 2n−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 2n 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 2n−1 for randomly chosen n-bit input seed.
4) The proposed architecture works when it is assumed that the number of 0's 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.

4. Proposed PRBG Method

To overcome the aforesaid shortcomings in the existing dual-CLCG method and its architecture as highlighted, a new PRBG method and its architecture are proposed in this project. 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 (5), This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination. Whereas the other four equations from (1) to (4) are same as in the case of dual-
CLCG method from equation (1) to (4).

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,

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

The pseudorandom bit sequence Zi is obtained by using the congruential modulo-2 equation (5),
Zi= (Bi + Ci) mod2= Bi xorCi (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, a1, b1, a2, b2, a3, b3, a4 and b4 are the constant parameters; x0, y0, p0 and q0 are the initial seeds. The necessary conditions to get the maximum length period are same as the existing dual-CLCG method. The proposed modified dual-CLCG method uses the congruential modulo-2 addition of two different coupled LCG outputs as specified in equation (5). Hence, the congruential modulo-2addition 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, the modulo-2addition of two coupled-LCG outputs in the modified dual-CLCG have also the same maximum length period of 2n for n-bit modulus operand. To perform the modulo-2 addition operation, it takes only single XOR logic. Therefore, by replacing equation (6) in exixting dual CLCG with equation (5), 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 2n. 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.

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

Input: n (positive integer), m 2n.
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 $x_{i1}$, $y_{i1}$, $p_{i1}$, $q_{i1}$ using equation (1), (9), (3) and (4) 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) $Z_i$ $(B_i\ C_i)$ mod2;
6) Return $Z_i$;
```

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

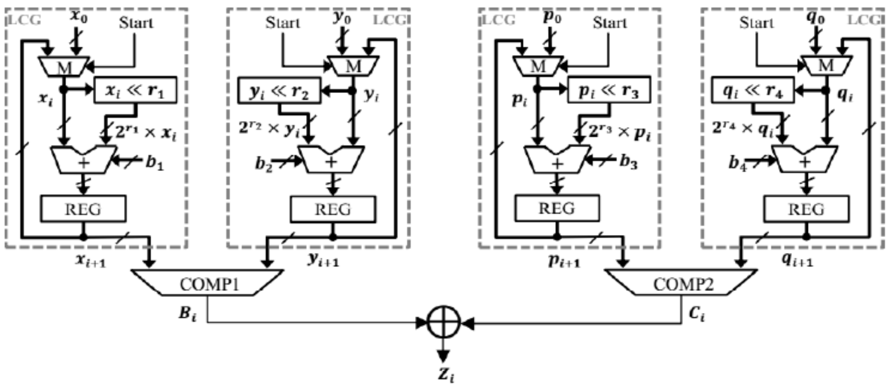

Figure 3: Proposed architecture of the modified dual-CLCG method

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 pseudo random bitate very uniform clock cycle. The first order architecture of the proposed modified dual-CLCG method is shown in Fig.3 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 previous chapter. It involves logical shift operation instead of 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.

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-2n 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,

Area, $A_{prop} = 4\ A_{3oa} + A_{mux} + A_{reg} + 2A_{cmp}+A_X$Critical path, $T_{prop} = T_{3oa} + T_{mux}$

Here, A_{3oa} and 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-2n addition [25]. Therefore, the three-operand modulo-2n 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 (A_{3CSA}) and critical path delay (T_{3CSA}) of carry save adder are evaluated as follows,

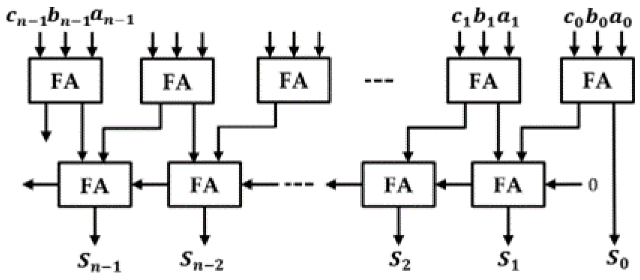

Figure 4: Three-operand modulo-2n carry-save adder.

Area, $A_{3CSA} = (2n - 1)\ A_{FA} = (2n - 1)\ (2A_X + 3A_G)$
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 (log2 n + 1) and therefore, the critical path delay is in the order of O (log2 n). Hence, the overall area (A_{cmp}) and critical path delay (T_{cmp}) of the magnitude comparator are evaluated as follows,

Area, $A_{cmp} = (n - 1)\ [9A_G + 4A_N]$
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,
Area, AMDCLCG
$= 4\ (A_{3oa} + A_{mux} + A_{reg}) + 2A_{cmp} + A_x)$

$= (16n-7)\, A_X + 2\,(27n-15)\, A_G + 12A_N + 4nA_{FF}$
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 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 2n 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 architecture of BBS method [12] has the large output latency of 2n 5 clock cycles due to the use of iterative Montgomery modular multiplier.

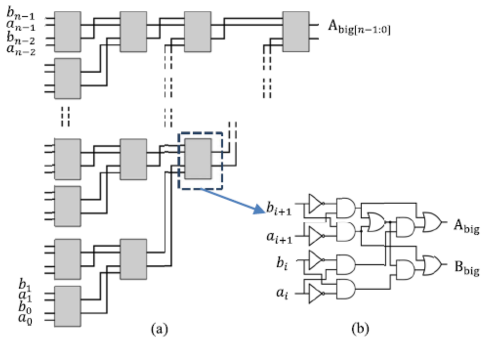

Figure 5: Magnitude comparator (a) n-bit, (b) Logic diagram of 2-bit comparator.

Example of the Proposed Modified Dual-CLCG

Method: Let A=5, b=5, a2=5, b2=3, a3=5, b3=1, a4=5, b4=7, and m 23. 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
$X_i = (7, 0, 5, 6, 3, 4, 1, 2)$; $P_i = (0, 1, 6, 7, 4, 5, 2, 3)$;
$Y_i = (6, 1, 0, 3, 2, 5, 4, 7)$; $Q_i = (3, 6, 5, 0, 7, 2, 1, 4)$;

Therefore, the output sequences B_i and C_i are evaluated as,
$B_i = (1, 0, 1, 1, 1, 0, 0, 0)$; $C_i = (0, 0, 1, 1, 0, 1, 1, 0)$

The final sequence Z_i generated from the proposed modified dual-CLCG method for n = 3-bit is,
$Z_i = (B_i + C_i) \bmod 2 = B_i \oplus C_i = (1, 0, 0, 0, 1, 1, 1, 0)$

5. Results

RTL SCHEMATIC:-The RTL schematic is abbreviated as the register transfer level it denotes the blue print of the architecture and is used to verify the designed architecture to the ideal architecture that we are in need of development. The hdl language is used to convert the description or summery of the architecture to the working summery by use of the coding language i. e verilog, vhdl. The RTL schematic even specifies the internal connection blocks for better analyzing. The figure represented below shows the RTL schematic diagram of the designed architecture.

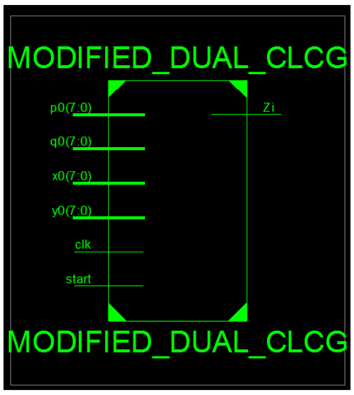

Figure 6: RTL Schematic view of Modified Dual CLCG

Technology Schematic: The technology schematic makes the representation of the architecture in the LUT format, where the LUT is consider as the parameter of area that is used in VLSI to estimate the architecture design. the LUT is consider as an square unit the memory allocation of the code is represented in there LUT s in FPGA.

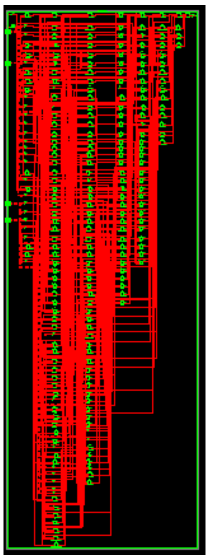

Figure 7: View technology Schematic of Modified Dual CLCG

Simulation

The simulation is the process which is termed as the final verification in respect to its working where as the schematic is the verification of the connections and blocks. The simulation window is launched as shifting from implantation to the simulation on the home screen of the tool, and the simulation window confines the output in the form of the wave forms. Here it has the flexibility of providing the different radix number systems.

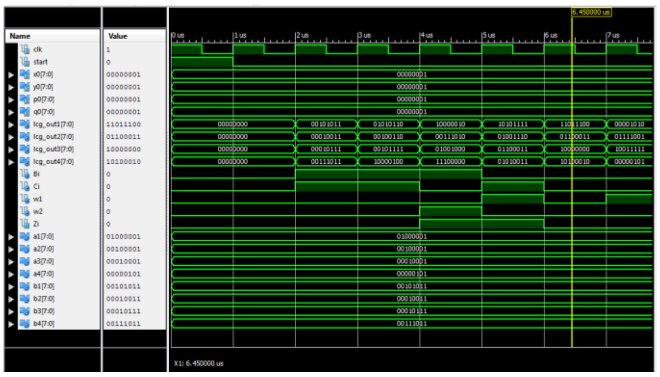
Figure 8: Simulated wave form of Modified Dual CLCG

Table 1: Parameter Comparison Table

Parameter	Existed	Proposed
Delay (ns)	23	7.07
Frequency (Mhz)	96	132

6. Conclusion

Dual-CLCG method involves dual coupling of four LCGs that makes it more secure than LCG based PRBGs. However, itisreportedthatthismethodhasthedrawbackofgeneratingpseudorandom 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 2n for n-bit architecture, large memory consumptions and fails to achieve the maximal period of 2n. 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 2n 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 com-plexity of the proposed architecture of the new modified dual-CLCG method is significantly reduced. 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 8-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 – lfsr based-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] Ding, “Blum-Blum-Shubgenerator, ” IEEE Electron. Lett., vol.33, no.8, p.667, Apr.1997.