VOL. 14, NO. 10, MAY 2019 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

A STUDY OF SEVERAL ALGORITHMS FOR PSEUDO-RANDOM GENERATOR BASED ON FIELD PROGRAMMABLE GATE ARRAY (FPGA)

Mochamad Hafiddin Ruslih, Tito Waluyo Purboyo and Anton Siswo Raharjo Ansori
Faculty of Electrical Engineering, Telkom University, Bandung, Indonesia
E-Mail: hafid.freeccs@gmail.com

ABSTRACT

The Procedural Content Generator (PCG) is a program for game content that uses a random number or pseudo number and value randomization process that produces a variety of random and unexpected game spaces. Many of the games use random number to increase the long duration of the game, and some rely heavily on procedural content creation techniques as makers of pseudo-random generator numbers (PRNG). The article discusses thrcondition of pseudo-random in combined with algorithm or a composite random number maker (CRNG). The aim is to develop and to improve the basic concepts of pseudo itself and discuss various forms of popular algorithms from random number makers and explain some new randomization algorithms. We present a method of comparing algorithms from FPGA that is needed for the implementation of PRNG and maximum operating systems. The randomization algorithms can be used for cryptographic generation, safely in stream ciphers.

Keywords: procedural content generator (PCG), pseudorandom number generator (PRNG).

INTRODUCTION

Pseudorandom Number Generator (PRNG) itself is classified in two classes, the first is the class for RNG hardware, and the second is a hardware algorithm that serves to produce values and numerical sequences that are close to the function of random numbers. The development of itself begins with irregular random generation to produce values that approach the random number [13] [7].

But there is a sequence that is really random and random, it is very important to practice it into a simulation [1]. And in the world of cryptography, it will be useful to make a number of structured sample samples that can test the quality of the pseudo-random itself. For example, we will analyze a value with a sequence of 0-1 obtained from a pseudorandom X = Random sequence, then enter a value of 0 or 1 as an X value, then an unstructured value will be obtained [2]. PRNG has a fairly long period, the maximum length before starting is determined by the size of the country [35]. This will form various sequences evenly distributed by several test samples [3]. Randomization and repetition not only includes content levels, maps, zones, scenarios, and others, but the mechanism and rules of the game are also included in repetition and randomization. This paper will focus on looking at various types of algorithms that can be used and have different randomization capabilities [14] [6].

Properties of Random Numbers: Random numbers have three characteristics. (1) Random numbers have to have a uniform distribution, (2) random numbers that will be taken should have an equal probability, and (3) each random number is independent. Random numbers must be drawn separately from uniform distributions with cumulative distribution function (CDF) and probability density function:

Random number generator (RNG)

Random Numbers Generator (RNG) plays a role in making computers replicate phenomena that occur in nature. Areas such as chemistry and physics might influence the application of simulations. Sampling is unlikely to be ignored in all of these cases; therefore small samples are biased or can be randomly selected from many [31]. Randomization algorithms data can be used to test or test program random data [36]. Cryptography - Random numbers have large systems under the Cryptographic field. Random number Gambling games, integer randomization is one example. From the potential decision making of the pseudorandom system [32].

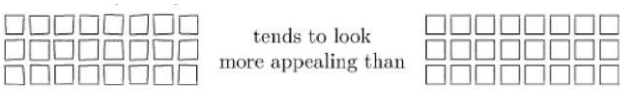

Figure-1. Random Numbers for Aesthetics [35].

Rolling dice, spinning roulette wheels, etc., are a fascinating thing to encounter. It takes some time to finally figure out the result can be seen in Figure-1.

Pseudo-random number generator (PRNG)

The Pseudo-random number is a type of random number that is generated using the initial value in the form of seed [29]. In addition, the PRNG is also known as the best random number number. One example of PRNG has been found in the past, to produce a random and distributed value that is evenly distributed, which means that random values are really well-randomized and perfectly divided and have good periods and times [4]. The PRNG algorithm provides many solutions to problems that occur in number randomization. [33].

Bruce Scheiner used these three characteristics as a definition of random number generator:

- The result (output) looks random, meaning that it passes the statistical test of randomness,
- Unpredictable, meaning that even if the algorithm and the previous random numbers are known, the next random numbers must not be easily computable,
- Cannot be reliably reproduced, meaning that if it executed twice with the same exact input, then the result is completely different.

What is meant by random in pseudo-random number is that the numbers are hard to predict. Pseudo-random numbers are generally produced by mathematical formulas and usually generated random numbers can be repeated periodically

Randomization using this generator exists to produce random sequence values that cannot be predicted by anyone (David, 2012). Such random numbers generator is called pseudo-random number generators (PRNG).

SEVERAL METHOD OF PSEUDORANDOM GENERATOR

Of the many review articles and research papers that have been tested, it seems that many of the researchers developed different algorithmic techniques to form an irregular random number. For example, the PRNG algorithm is used to get the result of a series of numeric values that display various pseudo-random properties and random numbers [8] [9]. Pseudo-random it is important in simulations, game environments, and cryptography [30].

Fibonacci: Fibonacci sequence is a sequence that X_{i+1} depends on more than one of the preceding values:

$$X_{i+1} = (X_i + X_{i-1}) \, mod \, \mathrm{c} \quad (1)$$

This generator was talked in early 1950s. Result of some generator testing shown that the sequence produced by Fibonacci sequence is not satisfactorily random. We may also consider generators of the form shown below:

$$X_{y+1} = (X_i + X_{i-1}) \, mod \, c \quad (2)$$

where k is a comparatively large value.

This form was introduced by Green, Smith, and Klem (JACM 6, 1959, 527-537).

Multiple recursive generators: Multiple recursive generators are based on a generalization of linear congruential generators (Christophe and Diethelm, 2009) [30]:

$$X_i = \left(A.X_{i-1} + \cdots + A_i.X_{i-Z^c}\right) mod \, c \quad (3)$$

where,
Z = permanent integer

Therefore, the ith of series relies on the previous k.

Inversive congruential sequences: This method is suggested by Eichenauer and Lehn (Statistische Hefte, 1986) [16].

$$X_{i+1} = (aX_i^{-1} + c) \, mod \, p \quad (4)$$

Here p is prime and X_i ranges over {0, 1, . . . ,$p-1$, ∞}. Inverses are defined by $0^{-1} = \infty, \infty^{-1} = 0$. Therefore, definition of $0^{-1} = \infty$ could simply be replaced by $0^{-1} = 0$ for purposes of implementation.

Blum-blum-shub generator: Linear Congruential Method is vulnerable to attacks if they are used to generate keys in a cryptosystem (Andrey and Berry, 2005). BBS generatordre is a random bit generator and has a very strong cryptographic properties. BBS generator has the following form [34]:

$$X_{i+1} = X_i^2 \quad (5)$$

where,
M = result of two big distinct primes.

The outcome bit is either least significant of X_{i+1} or the parity of X_{i+1}.

Linear congruential generatot (LCG): Linear congruential generators have a function as shown below [18]:

$$f(x) = (Xn + y) \, mod \, z \quad (6)$$

where,
Xn = multiplier,
y = increment,
z = the modulus

The sequence of integers $X_1, X_2, \ldots$ is generated by following function below [18]:

$$X_{i+1} = (aX_1 + C) \, mod \, m \quad (7)$$

$i = 0, 1, \ldots$

The output of the generator will be affected by the selection of a, c, m andX_0. The cycle length of LCM can be maximized by following these three conditions (Knuth, 2002):

- increment y is relatively prime to m,
- $Xn - 1$ is a multiple of every prime dividing m,
- $Xn - 1$ is a multiple of 4 when m is a multiple of 4

After the random integers are being generated from 0 to $m-1$, they can be converted to random numbers by [28]:

$$Ri = \frac{X_i}{m} \quad i = 1, 2, \ldots \tag{8}$$

Fibonacci: Fibonacci sequence is a sequence that X_{i+1} depends on more than one of the preceding values [17]:

$$X_{i+1} = (X_i + X_{i-1}) \, mod \tag{9}$$

This generator was talked in early 1950s. Result of some generator testing shown that the sequence produced by Fibonacci sequence is not satisfactorily random. We may also consider generators of the form shown below [17]:

$$X_{i+1} = (X_i + X_{i-k}) \, mod \tag{10}$$

where k is a comparatively large value. This form was introduced by Green, Smith, and Klem (JACM 6, 1959, 527-537).

Multiplewith-carry method: Multiple with-carry method generators are based on a generalization of linear congruential generators (Christophe and Diethelm, 2009) [16]:

$$X_i = \left(a_1 . X_{i-1} + \cdots + a_k . X_{i-k^c}\right) mod \, m \tag{11}$$

where,

k = permanent integer

Therefore, the ith of series relies on the previous k.

Inversive congruential sequences: This method is suggested by Eichenauer and Lehn (Statistische Hefte, 1986) [16].

$$X_{i+1} = (aX_1 + C) \, mod \, p \tag{12}$$

Here p is prime and X_i ranges over {0, 1, . . . ,$p-1$, ∞}. Inverses are defined by $0^{-1} = \infty, \infty^{-1} = 0$. Therefore, definition of $0^{-1} = \infty$ could simply be replaced by $0^{-1} = 0$ for purposes of implementation.

Blum-blum-shub generator: Linear Congruential Method is vulnerable to attacks if they are used to generate keys in a cryptosystem (Andrey and Berry, 2005) [23]. BBS generatordre is a random bit generator and has a very strong cryptographic property. BBS generator has the following form [34]:

$$X_{i+k} = X_i^2 \, mod \, M \tag{13}$$

where,

M = result of two big distinct primes.

The outcome bit is either least significant of X_{i+1} or the parity of X_{i+1}.]

Done in Software that is active on MicroBlaze processors and uses C programming language and TCP / IP library.

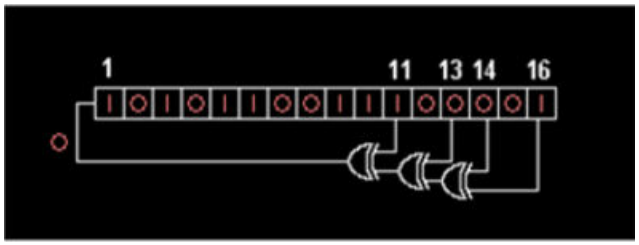

Figure-2. Linear Feedback Shift-Register LFSR with its state diagram [30].

Linear feedback shift-register (LFSR): is used for creating a sequence of a binary bit. This method is used in many hardware applications. LSFR is generating a random number in a fast way [25]. XOR operation is used for generating a random number. This method is used in digital broadcasting and communications that show in Figure-3.

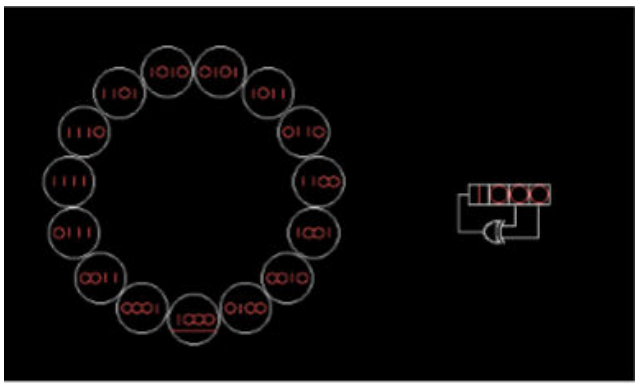

Figure-3. A 16 bit Fibonacci LFSR [30].

Fibonacci LFSRs: Fibonacci LFSRs. It is an algorithm that taps the numbers that in accordance with the provisions of the polynomial number in the table so in recording the maximum value of 65636is obtained does not include the all-zero status. [30]. the bit position is very influential in the next state called tap that show in Figure-4.

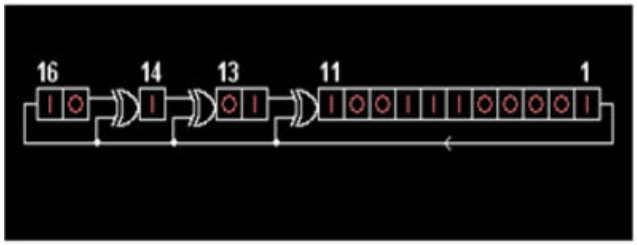

Figure-5. A 4 bit Galois LFSR [30].

Galois LFSR: The LFSR algorithm inthe Galois Configuration there is a Galois configuration if the system is clocked; with the bits not stored and then shifted one position to the right does not change [10]. The Galois LFSR algorithm does not combine everyone, but the tap there is a new input (for example XOR is done in LFSR and there is no XOR gate is serially activated, there is a new input (for example XOR is done in the LFSR and there is no XOR than the whole chain) [30], so that it is possible for each beat to calculated in parallel, increasing execution that show in Figure-5.

REVIEW AND DISCUSSIONS

Table-1 discusses the algorithms used in the pseudo-random generator and includes the author of the reference. There are 17 methods listed in the table in accordance with the above explanation. This table is useful to facilitate the reader in understanding this paper.

Table-1. Resume Algorithm for Pseudo-Random Generator.

Ref No	Methods	Author's Name	Discussion
[18] [28]	Blum Shub Algorithm	Challiz D. Omorog, Bobby D. Gerardo, Ruji P. Medina	Pseudorandom Number Generator sequence which is produced by a generator is a polynomial time randomized for the every fixed initial segment. In this method there are two different generators are defined. They are 1/P generator which is predictable and $X_2 \, mod \, N$ generator is not predictable. The applications of these methods are constructing de Bruijn sequences and public key cryptography.
[24]	Complementary Algorithm	Jianping Hu, Tiefeng Xu, Ping Lin	Complementary-multiply-with-carry is another method, uses the primes of the for $abr + 1$. This method is simple, fast, have a good quality and require an astonishing period. CMWC is used in game development. The inversive congruential generator is a nonlinear congruential PRNG. This generator denoted by ICG.
[21]	Inversive Congruential Generators	Peter Hellekalek	Inversive Congruential Generator (ICG) is an algorithm that must choose modulus m, multiplier x, additive y, an initial value Zo. Then, find and determine the best congruential value. We ensure that this generator with ICG (m, x, y, Z_0)has a different and better-known feature than ICG. In main modulation it is not only a grid structure, but it is also very different from a linear congruential generator (LCG) algorithm.
[22]	Lagged Fibonacci Algorithm	Shankar B R, Karuna Kamath K	Lagged Fibonacci generator algorithm has a equation$Z_n = X_n . \text{y} . \text{k} + Z_n$ l (mode); l > k Where M modules and also prime, l = register length, k = lag. This method used in this computer hardware can be obtained and has that can be implemented into modulo arithmetic with bit truncation storage.
[7] [12]	Linear Congruential Algorithm	Tomas Eriksson, Anderson, Norbert	The Linear Congruential Generator (LCG) is the basis of the PRNG algorithm. The following is the equation for generating a random number, $X_{i+1} = (aX_1 + C) \, mod \, m$where 'Xn' is a sequence of pseudo-random values, and is modulus; 'a.x.c' is a multiplier, and ' c 'is the value of increase. 'Xo' is the seed value. This method is fast and requires less memory. But LCG is not useful for applications where high-quality randomization is needed.
[25]	Linear Feedback Algorithm	Sergey S. Krivenko, Stanislaw A.	Linear Feedback Shift-Register is used for creating a sequence of a binary bit. This method is used in many hardware applications. Linear feedback shift register is generating a random number in a fast way. XOR operation is used for generating a random number. This method is used in digital broadcasting and communications.
[26] [29]	Middle-square Algorithm	Hamed Rahimov, Majid Babaie, Hassan Hassanabadi	Middle-square Algorithm, middle digits of the previous number generate each successive number. First of all, we take a seed value and calculates it square. After squaring it, we select middle digits of that number and take it as the seed value for the next pseudorandom number. This way, middle digits of previous numbers acts as the seed value for the next number.
[24] [30]	Multiply-with-carry method	Jianping Hu, Tiefeng Xu,	Multiple with-carry method generators are based on a generalization of linear congruential generators The Multiply-

		Ping Lin	with-carry method is used to create random integer sequence values from the initial set of 2 to 2000 by randomly selecting a seed (Xo) value or Xo. MWC makes integer computer arithmetic simple and refers to fast generation of sequences random number values. With a very large period, ranging from 260 to 22 million, this is the advantage of this method.
[20]	Xorshift	Guang Zeng, Wenbao Han, Wei Sun	Xorshift random number generator is a class of PRNGs. By the combination of many XORshift operations, simple and fastest RNGs can be combined. If the number of combinations is odd then this primitive is Invertible. XORshift requires less code and a small state. This is the fastest in cryptographically-secure random number generators.
[27]	MIX MAX Generator	Konstantin G. Savvidy	MIXMAX is NXN matrix developed as a solution to the problem of determining the integer-valued unimodular maximal matrix generator of PRNs. MIXMAX generator is faster in 24 bit and 48-bit precision and 32 bit Mersenne Twister. Yarrow is an improvement of a PRNG. Yarrow reuses existing building block.
[17]	Counter-based random number generator (CBRNG)	Toni Stojanovski, Johnny Pihl, and LjupcoKocarev	Counter-based random number generator (CBRNG) is used for parallel computation. Counter based PRNGs are quick, need small or no state, and are easy to initialize. They have extensive periods and have an approved set of statistical tests. Full AES and Threefish encryption algorithms are used by CBPRNGs. The fastest PRNGs are including in Philox and Threefry families.
[30]	Galois LFSRs	Vishakha V. Bonde, A. D. Kale	The LFSR algorithm in the Galois Configuration there is a Galois configuration if the system is clocked; with the bits not stored and then shifted one position to the right does not change. The Galois LFSR algorithm does not combine everyone, but the tap there is a new input (for example XOR is done in LFSR and there is no XOR gate is serially activated, there is a new input (for example XOR is done in the LFSR and there is no XOR than the whole chain)
[30]	Fibonacci LFSRs	Vishakha V. Bonde, A. D. Kale	Fibonacci LFSRs. It is an algorithm that taps the numbers that in accordance with the provisions of the polynomial number in the table so in recording the maximum value of 65535 is obtained does not include the all-zero status. The bit position is very influential in the next state called taps that show in Figure-4.
[5] [6] [16]	Combined Random Number Generators	Pierre L'ecuver	The random number generator (CRNG) has many ways to combine the many RNGs that are done. The well-known combined generator was found by Wichmann and Hill (1982), which only gave suggestions for combining several LCG, is a different algorithm. The concept is not difficult, but gives ideas to other researchers. LCG itself has only shown a mathematical improvement from CRNGs. For example, in the case of combining LCG with LCG

CONCLUSIONS

From all the discussions below, it can be concluded that the researcher of this paper, examines various literature reviews of various methodologies and designs used to generate random numbers, and it has been known from all the discussions below, it can be concluded that we have many methods to generate numbers randomly. Each method has advantages and disadvantages. Different ways have been investigated to increase randomization of PRNG. For the Linear Congruential Generator (LCG) number generator is the best effective method. When the value goes up, the greater the order, LCG can be used in simulation or in cryptography. We conclude that only by giving a random value at the beginning, the LCG will generate random numbers for several applications that are entirely dependent on randomness and the value of the Xo or the seed itself.

Future work includes developing the function, advantages and disadvantages for each pseudo-random numbers generator. By reviewing some of the methods in this paper, it is hoped that there will be other effective methods that can generate random numbers better.

REFERENCES

[1] GU Xiao-chen, ZHANG Min-xuan. 2009. Uniform Random Number Generator using Leap- Ahead LFSR

Architecture. International Conference on Computer and Communications Security.

[2] Jonathan M. Comer, Juan C. Cerda, Chris D. Martinez, and David H. K. 2012. An efficient FPGA random number generator using LFSRs and cellular automata. Hoe 44th IEEE Southeastern Symposium on System Theory University of North Florida, Jacksonville, FL March 11-13.

[3] Pawel Dabal, Ryszard Pelka. 2012. FPGA Implementation of Chaotic Pseudo-Random Bit Generators MIXDES, International Conference. Mixed Design of Integrated Circuits and Systems. May 24-26, Warsaw, Poland.

[4] David B. Thomas, Wayne Luk. 2013. The LUT-SR Family of Uniform Random Number Generators for FPGA Architectures. IEEE transactions on very large scale integration (VLSI) systems. 21(4).

[5] P. L'Ecuyer. 1996. Combined Multiple Recursive Random Number Generators. Operations Research. 44: 816-822.

[6] P. L'Ecuyer. 1999. Tables of Maximally Equidistributed Combined LFSR Generators. Math. Comp. 68: 261-269.

[7] RA. Wichmann and LD. Hill. 1982. An etlicient and portable pseudorandom number generator. Applied Statistics. 1: 188-190.

[8] R. M. Smelik, T. Tutenel, K. J. de Kraker and R. Bidarra. 2010. Integrating procedural generation and manual editing of virtual worlds. In Proceedings of the ACM Foundations of Digital Games. ACM Press.

[9] G. N. Yannakakis and J. Togelius. 2011. Experience-driven Procedural Content Generation. IEEE Transactions on Affective Computing. (to appear)

[10] Toni Stojanovski, Johnny Pihl, and Ljupˇco Kocarev. 2001. Chaos-Based Random Number Generators. IEEE Transactions On Circuits And Systems. 48(3).

[11] L'ecuyer P. 1998. Efficient and Portable Combined Random Number Gnerator. Communications of the ACM. 31(6).

[12] Bani Yassein, Muneer. 2014. A new Dynamic Counter-Based Broadcasting Scheme for Mobile Ad Hoc Network. Departement of Computer Science, Jordan University of Science and Technology.

[13] Challiz D. Omorog, Bobby D. Gerardo. 2018. Enhanced Pseudorandom Number Generator based on Blum-Blum-Shub and Elliptic Curves. IEEE, No 269-275.

[14] Mehrdad Majzoobi, FarinazKoushanfar and Srinivas Devadas. 2011. FPGA-Based True Random Number Generation Using Circuit Metastability with Adaptive Feedback Control. Massachusetts Institute of Technology, CSAIL Cambridge, 17-32.

[15] Guang Zeng, Wenbao Han, Wei Sun. 2007. Improvement of One Type Xorshift Random Number Generators. IEEE, Department of Applied Mathematics Zhengzhou Information Science and Technology Institute, Zhengzhou, China.

[16] C. Alexopoulos, K. Kang, W. R. Lilegdon and D. Goldsman. 1995. Inversive Pseudo-Random Generator: Concepts, Results and Link. Department of Mathematics University of Salzburg A-5020 Salzburg, Austria.

[17] Shankar B. R. and Karuna Kamath K. 2009. Lagged Fibonacci Generators Using Elliptic Curves over Finite Fields. International Conference on Computer Engineering and Technology, Dept. of Mathematical and Computational Sciences, NITK Surathka, India.

[18] Tomas Eriksson, Member, IEEE, John B. Anderson, Life. 2007. Linear Congruential Trellis Source Codes: Design and Analysis. IEEE Trasaction on Communications. 55(9).

[19] Jianping Hu, Tiefeng Xu and Ping Lin. 2005. Low Power Adiabatic Multiplier with Complementary Pass-Transistor Logic. IEEE, Faculty of Information Science and Technology Ningbo University Ningbo, Zhejiang, 3 1521 I, China.

[20] Sergey S, Krivenko. 2014. Many-to-many Linear-feedback Shift Register. IEEEXXXIV International Scientific Conference Electronics and Nanotechnology (ELNANO).

[21] Hamed Rahimov, Majid Babaie, Hassan Hassanabadi. 2011. Improving Middle Square Method RNG Using Chaotic Map. Department of Computer Engineering, Shahrood University of Technology, Shahrood, Iran.

[22] Konstantin G., Savvidy. 2014. The MIXMAX random number generator. Department of Physics and Center for Transcriptional Medicine, Nanjing University, Nanjing, China, April 2.

[23] Elnaz Koopahi and Shahram Etemadi Borujeni. 2016. Secure Scan-based Design Using Blum BlumShub Algorithm. IEEE, Faculty of Computer Engineering, University of Isfahan.

[24] Ritu Maheshwari, Sonam Gupta, Vinita Sharma, Vishakha Chauhan. 2014. VRS algorithm A Novel Approach to Generate Pseudo Random Numbers. Apaji Institute of Mathematics and Applied Computer Technology Banasthali Vidyapith, India-304022.

[25] Vishakha V. Bonde, A. D. Kale. 2015. A Review on Implementation of Random Number Generation based on FPGA. International Journal of Science and Research (IJSR). 4(1).

[26] MELISSA E. O'NEILL. PCG: A Family of Simple Fast Space-Efficient Statistically Good Algorithms for Random Number Generation. ACM Transactions on Mathematical Software. V(N, Article A), Publication date:

[27] Aleksander Dańko. 2009. Improving Pseudo-Random Generators. International Conference on Biometrics and Kansei Engineering.

[28] Pedro Sampaio, Augusto Baffa, Bruno Feij and Mauricio Lana. 2017. A fast approach for automatic generation of populated maps with seed and difficulty control. Brazilian Symposium on Computer Games and Digital Entertainment.

[29] Pawel Daba. 2012. FPGA Implementation of Chaotic Pseudo-Random Bit Generators. International Conference. Mixed Design of Integrated Circuits and Systems, May 24-26.

[30] G. S. Emeralda, Dirgantoro B., Setianingsih C. and Purboyo T.W. 2019. A Review of Pseudo-Random Numbers Generation Techniques. Journal of Engineering and Applied Sciences.

[31] GU Xiao-chen, ZHANG Min-xuan. 2009. Uniform Random Number Generator using Leap- Ahead LFSR Architecture. International Conference on Computer and Communications Security.

[32] Jonathan M. Comer, Juan C. Cerda, Chris D. Martinez and David H. K. 2012. Hoe 44th IEEE Southeastern Symposium on System Theory University of North Florida, Jacksonville, FL March 11-13.

[33] Carlos Arturo Gayoso, C. González, L. Arnone, M. Rabini, Jorge Castiñeira Moreira. 2013. Pseudorandom Number Generator Based on the Residue Number System and its FPGA Implementation. Argentine School of Micro-Nanoelectronics, Technology and Applications.

[34] Sidorenko A., Schoenmakers B. 2005. Concrete Security of the Blum-Blum-Shub Pseudorandom Generator. In Cryptography and Coding: 10th IMA International Conference, Lecture Notes in Computer Science. 3796: 355-375.

[35] M. Bellare, S. Goldwasser and D. Micciancio. Pseudo-Random. Number Generation within Cryptographic Algorithms: the DSS Case.

[36] 1997. In Advances in Cryptology - Crypto 97 Proceedings, Lecture Notes in Computer Science No. 1294. Springer-Verlag.

[37] Y. D. Aprilia, L. Roswan and T. W. Purboyo. 2018. A Review of Several Algoritms for Data Mining. Journal of Engineering and Applied Sciences.