

Article

Combined Pseudo-Random Sequence Generator for Cybersecurity

Volodymyr Maksymovych ¹, Mariia Shabatura ¹, Oleh Harasymchuk ², Ruslan Shevchuk ^{3,4,*}, Pawel Sawicki ⁵ and Tomasz Zajac ³¹ Department of Information Technology Security, Lviv Polytechnic National University, 79013 Lviv, Ukraine² Department of Information Security, Lviv Polytechnic National University, 79013 Lviv, Ukraine³ Department of Computer Science and Automatics, University of Bielsko-Biala, 43-309 Bielsko-Biala, Poland⁴ Department of Computer Science, West Ukrainian National University, 46009 Ternopil, Ukraine⁵ ITSO GmbH, D-10829 Berlin, Germany

* Correspondence: rshevchuk@ath.bielsko.pl

Abstract: Random and pseudo-random number and bit sequence generators with a uniform distribution law are the most widespread and in demand in the market of pseudo-random generators. Depending on the specific field of application, the requirements for their implementation and the quality of the generator's output sequence change. In this article, we have optimized the structures of the classical additive Fibonacci generator and the modified additive Fibonacci generator when they work together. The ranges of initial settings of structural elements (seed) of these generators have been determined, which guarantee acceptable statistical characteristics of the output pseudo-random sequence, significantly expanding the scope of their possible application, including cybersecurity. When studying the statistical characteristics of the modified additive Fibonacci generator, it was found that they significantly depend on the signal from the output of the logic circuit entering the structure. It is proved that acceptable statistical characteristics of the modified additive Fibonacci generator, and the combined generator realized on its basis, are provided at odd values of the module of the recurrent equation describing the work of such generator. The output signal of the combined generator has acceptable characteristics for a wide range of values of the initial settings for the modified additive Fibonacci generator and the classic additive Fibonacci generator. Regarding the use of information security, it is worth noting the fact that for modern encryption and security programs, generators of random numbers and bit sequences and approaches to their construction are crucial and critical.

Keywords: pseudo-random number; pseudo-random sequence generators; authentication; encryption of information



Citation: Maksymovych, V.; Shabatura, M.; Harasymchuk, O.; Shevchuk, R.; Sawicki, P.; Zajac, T. Combined Pseudo-Random Sequence Generator for Cybersecurity. *Sensors* **2022**, *22*, 9700. <https://doi.org/10.3390/s22249700>

Academic Editors: Nikos Fotiou and Marina Gavrilova

Received: 5 November 2022

Accepted: 8 December 2022

Published: 11 December 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Pseudo-random number (PRNG) and bit sequence (PRBSG) generators, in general, pseudo-random sequence generators (PRSG), are used in many fields of science and technology [1–7]. They play an important role in modeling various processes, in solving industrial problems and cybersecurity problems. The relevance of cybersecurity tasks is growing every year. Such tasks include encryption of information, generation of secret keys, authentication, confidentiality and integrity of information. Additionally, an effective solution to these problems is not possible without the use of PRNG and PRBSG.

Random numbers are an important input for many Internets of Things (IoT) functions [8]. In [9], a deep review is given for lightweight random and pseudo-random number generators designed for constrained devices, such as wireless sensor networks and RFID (Radio Frequency Identification) tags, along with a study of a Trifork pseudo-random number generator for constrained devices. In [10], the authors focus on the security and privacy risks in cloud databases and provide a solution for clients who want to generate

the pseudo-random number collaboratively in a distributed way which can be reasonably secure, fast and low-cost to meet the requirements of a cloud database. In [11], the use of evolutionary computation is proposed for designing and optimizing lightweight PRNGs. In [12], the authors introduce a novel approach to implement Pseudo Random Number Generators, by proposing the use of generative adversarial networks (GAN) to train a neural network to behave as a PRNG.

In [13], a text algorithm for watermarks based on a PRNG for use in cryptography was proposed, which has good invisibility and reliability to withstand removal, modification attacks, etc., and can be used in the field of hiding information using cloud computing. An interesting method is proposed in [14], whereby random numbers are generated based on reinforcement learning characteristics that select the optimal behavior considering every possible status up to the point of episode closing to secure the randomness of such random numbers.

A number of questions about the approaches to the construction of PRNG and PRBSG, as well as the requirements for their use in cybersecurity systems are discussed in many works, where the authors explore known methods of constructing such generators, analyze their weaknesses and search for new generation methods [15–24].

Among PRNG, it is possible to allocate additive Fibonacci generators (AFG) to a separate group [25–33], which is characterized by the fact that the sequences at their output have acceptable statistical characteristics. This group of generators is effectively used for mathematical and statistical calculations, as well as in the modeling of various processes. It should be noted that AFG, in contrast to linear congruent generators, can be used in algorithms that are critical to the quality of generated pseudo-random numbers, particular in cryptography. The ever-increasing possibilities for the use of AFG and the new requirements for them pose new challenges to developers in their design and implementation; therefore, in this direction, they are actively working to find new structures of modified additive Fibonacci generators (MAFG) with improved characteristics. In particular [34], the authors propose a five-state real-time generator for the Fibonacci sequence and give formal proof of the correctness of the generator. The proposed five-state Fibonacci sequence generator is optimum in generation steps and is implemented on the smallest known finite state automaton in the number of states. In [35], the researchers sought to determine, between the Fibonacci Random Number Generator and the Gaussian Random Generator, which is better for improving data security in cryptographic software systems.

To assess the quality of PRNG and PRBSG, it is necessary to use statistical testing based on various randomization algorithms and obtain an integral picture of the assessment of the pseudo-random sequence at the generator output. The National Institute of Standards and Technology (NIST), which is part of the American National Standards Institute (ANSI), the International Organization for Standardization (ISO), and national standardization authorities are developing assessment methods, requirements, and standards for PRNG and PRBSG. Among the most popular tests for quality assessment is a set of statistical tests NIST [36–40]. If the generated sequences are aimed to be used in cryptographic applications, it is also necessary to conduct cryptographic testing.

In general, an extremely large number of specialists are involved in the design, construction, quality assessment and application of PRNG and PRBSG.

In comparison with the known ones, the novelty of this work lies in the optimization of the parameters of the MAFG, which includes the possibility of realizing an arbitrary value of the modulus of the recurrent equation of the generated polynomial. Additionally, further improvement was achieved in the output signal characteristics of the pseudo-random generator through the joint work of the MAFG and the classical AFG.

1.1. Related Literature

In [40–49], we and another scientist proposed MAFG, which differs from that reported in [25–36] as follows:

- The possibility of functioning according to recurrent equations, the modules of which can have arbitrary values;
- The presence of an additional logic circuit that can significantly improve the statistical characteristics of the output signals;
- The presence of structures and relevant hardware models, which allows them to be implemented in modern element bases.

Advantages of the new MAFG are demonstrated by examples of their work in accordance with several recurrent equations [40–45].

In connection with the emergence of our new MAFG, a number of questions arise, which include the following:

- The choice of polynomials on the basis of which it is possible to synthesize MAFG with acceptable statistical characteristics;
- Determination of the generator's structural element parameters at which acceptable statistical characteristics are reached;
- The method of joint work of the MAFG with classical AFG in order to further improve the statistical characteristics of the output sequence and increase cryptosecurity;
- The definition and expansion of ranges of structural elements initial installations (seed) at which acceptable statistical characteristics of the output pseudo-random sequence are guaranteed;
- The search for new approaches to the design of the MAFG and methods of their implementation, in particular on the Programmable Logic Device;
- The selection of methods for testing the quality of sequences obtained from such generators for compliance with given conditions.

Table 1 shows a summary of the comparison between the research literature and the work developed in this study. If a column item is marked, the work in that row addresses it. If it is not marked, then the work either does not specify or does not address that item. The items of the table are as follows:

1. Work—contains a reference to research;
2. Repetition period—if the work investigated the repetition period of the generator's output sequence;
3. Statistical characteristics—if the work investigated and showed statistical characteristics of the output sequence of the generator;
4. Ranges of key—if the work investigated the value ranges of key information;
5. Polynomial—if it was possible to use an arbitrary value of the modulus of the recurrent equation of the polynomials forming the algorithm of the generator implemented;
6. Compatible work—if the compatible work of several generators was possible;
7. Hardware implementation—if the work showed the hardware implementation of the generator, in addition to software.

1.2. Purpose of Work

The aim of the work is to optimize the structures of classical AFG and MAFG in their joint work, to determine the ranges of initial settings of structural elements (seed) AFG and MAFG, which guarantee acceptable statistical characteristics of the output pseudo-random sequence.

Table 1. Comparison of related work with the study in this paper.

Work	Repetition Period	Statistical Characteristics	Ranges of Key	Polynomial	Compatible Work	Hardware Implementation
[2]		X				X
[3]	X	X	X			
[8]		X				
[11]						X
[25]		X			X	
[27]		X				
[28]	X	X	X			
[30]		X	X			X
[31]	X					X
[33]	X	X	X		X	
[35]	X	X	X			
This work	X	X	X	X	X	X

2. Materials and Methods

Structure Scheme of the Joint Work of AFG and MAFG and the Principle of Its Work

The generalized scheme of joint work of AFG and MAFG is given in Figure 1, and the corresponding detailed structure scheme is given in Figure 2.

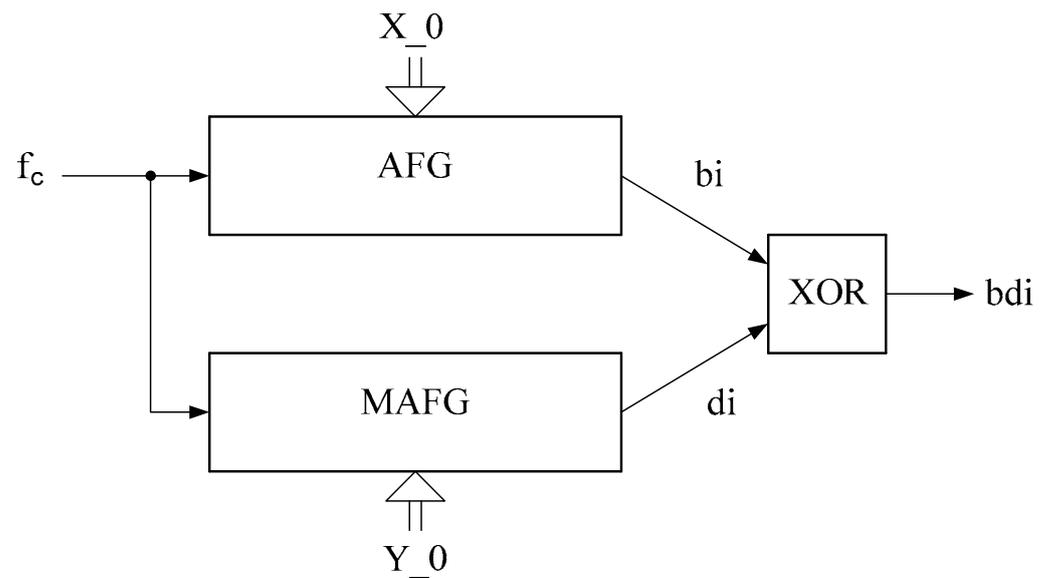


Figure 1. Generalized scheme of the joint work of AFG and MAFG.

The output pseudo-random bit streams AFG— bi and MAFG— di are combined through the logical element XOR, thus forming the output bit sequence— bdi . The inputs AFG and MAFG receive clock pulses— f_c . Before starting the work, the initial values (seed) are written to the AFG and MAFG memory registers X_0 and Y_0 accordingly.

In this paper, the characteristics of the device are investigated, despite the fact that AFG is constructed in accordance with the primitive polynomial

$$GF(2^{30}) = x^{30} + x^{23} + x^2 + x + 1, \quad (1)$$

and MAFG is constructed according to the primitive polynomial

$$GF(2^{20}) = y^{20} + x^3 + 1, \quad (2)$$

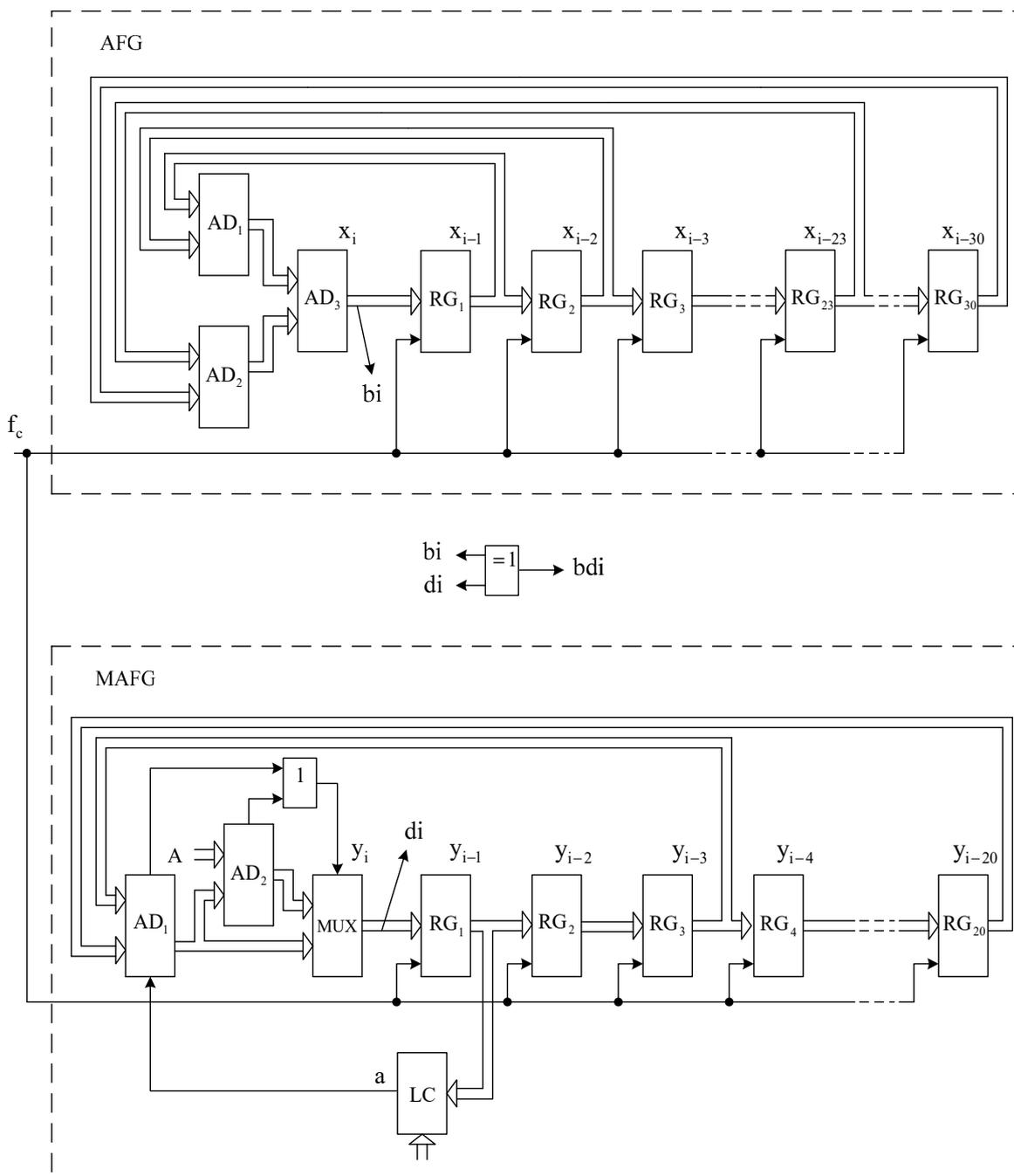


Figure 2. Structure scheme of the joint work of AFG and MAFG.

The choice of polynomials (1) and (2) was made on the basis of studies of many other variants and was based on achieving acceptable statistical characteristics of the output pseudo-random signal, in the full range of possible values X_0 and Y_0 with minimal hardware costs.

AFG consists of adders $AD_1 - AD_3$ and memory registers $RG_1 - RG_{30}$ (Figure 2). According to polynomial (1), AFG operates on a recurrent equation

$$x_i = (x_{i-30} + x_{i-23} + x_{i-2} + x_{i-1}) \bmod m, \tag{3}$$

where x_i is the adder AD_3 output number; $x_{i-1}, x_{i-2}, x_{i-23}, x_{i-30}$ are numbers at register $RG_1, RG_2, RG_{23}, RG_{30}$ outputs, respectively; and m is the modulus of the recurrent equation, determined, in this case, by the number of binary bits of the structural elements of AFG.

The output bit sequence bi is formed at the output of the least significant bit of the adder AD_3 .

The MAFG consists of adders AD_1 and AD_2 , multiplexer MUX, memory registers $RG_1 - RG_{20}$, logic element OR and logic circuit LC .

In accordance with polynomial (2) and the new method of its internal construction [42], MAFG operates on a recurrent equation

$$y_i = (y_{i-20} + y_{i-3} + a) \bmod h, \quad (4)$$

where y_i is the multiplexer MUX output number; y_{i-3} and y_{i-20} are register RG_3 and RG_{20} outputs numbers; h is the the module of the recurrent equation, which is determined, in this case, by the number of binary bits of the structural elements of the MAFG and the value of the control code A that is coming to the second input of the combination adder AD_2 ; and a is the output signal of the logic circuit LC , which is determined by the equation

$$a = a_0 \oplus a_1 \oplus \dots \oplus a_s, \quad (5)$$

where a_i ($i = 0, 1, \dots, s; s \leq n - 1$) are the bit values of the number y_{i-1} , and n is the number of its binary bits.

It should be emphasized that, due to its original structure, the MAFG can operate in accordance with a recurrent equation with an arbitrary value of the module, which will be used in further studies of this work.

The output bit sequence di is formed at the output of the multiplexer MUX's least significant bit.

3. Results and Discussion

3.1. Results of AFG Research

The main task of AFG, which works in pair with MAFG, is to ensure a stable repetition period of output pseudo-random sequence on the whole set of initial values X_0 in registers that are part of it.

It is known [36] that the maximum period of the AFG output sequence is achieved if it works following the primitive polynomial.

As mentioned earlier in this paper, the selected classical AFG works following the primitive polynomial (1) and, therefore, the recurrent Equation (3).

At $m = 2$, the repetition period of the AFG output signal is equal to $2^{30} - 1$ for any initial value X_0 [43].

Figure 3 shows the statistical characteristics of the AFG output signal obtained using statistical tests NIST [37–39]. The results are presented in the statistical portraits. A statistical portrait is an $m \times q$ matrix, where m is the number of binary sequences to be tested and q is the number of statistical tests [37].

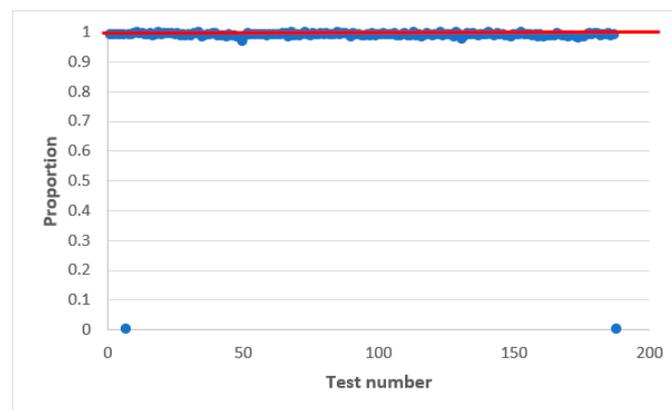


Figure 3. Statistical portrait of the AFG output sequence.

According to the obtained statistical portrait, the proportion of sequences that passed each statistical test is determined. For this, setting the significance level α and the probability values p exceeding the established level α are calculated for each q tests.

It is considered that the generator has passed the test if the value of the coefficient is within $[r_{\max}, r_{\min}]$. The confidence interval limits are determined by the following expression [37]:

$$r_{\max(\min)} = p \pm 3\sqrt{\frac{p(1-\alpha)}{m}}, \quad (6)$$

where $p = 1 - \alpha$.

By substituting the corresponding values, namely $\alpha = 0.01$ and $m = 1000$, we obtain the limits of the confidence interval $0.980561 - 0.999439$. A sequence is said to meet randomness requirements when all test values fall within these limits.

The NIST statistical test set consists of 15 tests, calculating 188 results values [37]. Each of these 188 values is interpreted as the result of a separate test. Each of these 188 values is marked as a dot in the statistical portrait.

The statistical portrait is obtained at the initial value of numbers $X(0) = 7$. Similar statistical portraits were obtained for other values $X(0)$. The output AFG sequence does not pass all the tests from the NIST set, so such a generator is not statistically safe.

3.2. Results of MAFG Research

We will research the characteristics of MAFG for the case of its operation in accordance with the primitive polynomial (2).

Figures 4 and 5 show the dependences of the repetition periods T on the initial values of numbers Y_0 in the registers $RG_1 - RG_{20}$. The value Y_0 is determined by the equation:

$$Y_0 = h^{19}y_{i-1}(0) + h^{18}y_{i-2}(0) + h^{17}y_{i-3}(0) + \dots + hy_{i-19}(0) + y_{i-20}(0), \quad (7)$$

where $y_{i-1}(0), y_{i-2}(0), \dots, y_{i-20}(0)$ are the initial values of the numbers in registers $RG_1, RG_2, \dots, RG_{20}$.

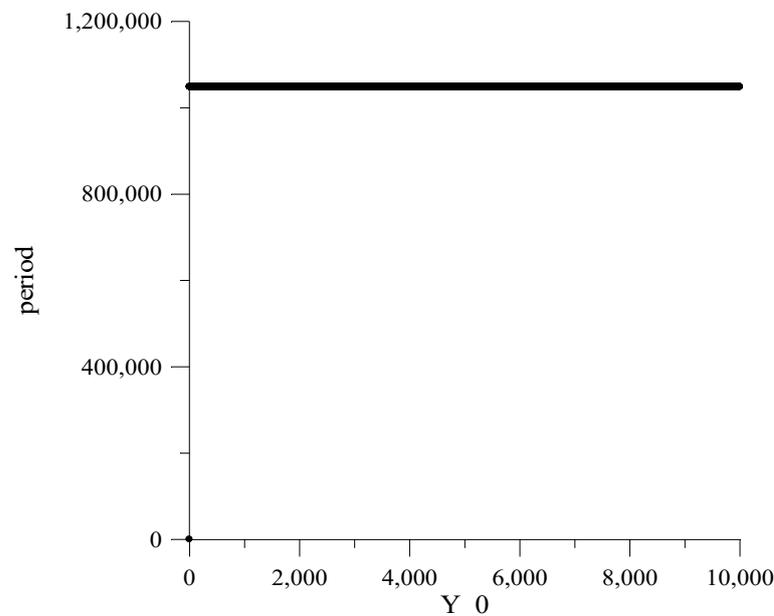


Figure 4. Dependencies of repetition periods for $X_0, h = 2$ and $a = 0$.

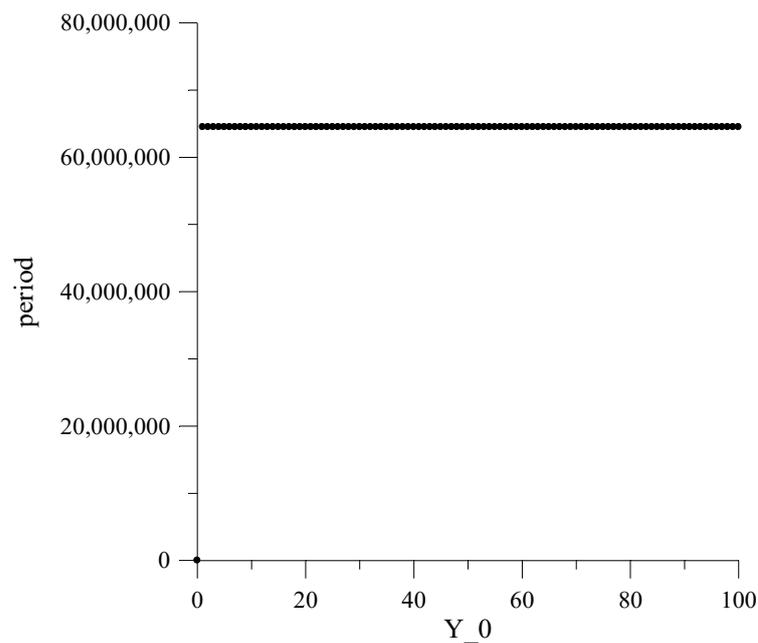


Figure 5. Dependencies of repetition periods for X_0 , $h = 3$ and $a = 0$.

Dependencies are obtained only for two h values and one value of the LC output signal a because obtaining such dependencies for other values of these parameters requires a significant increase in computing resources and machine time.

Here, it should be emphasized that the constancy of the MAFG output sequence repetition period, for the whole set of the initial values Y_0 , is guaranteed only at $h = 2$ and $a = 0$.

Investigations have shown that for other h and a values, the repetition period significantly varies depending on Y_0 , achieving under specific Y_0 small values, thus generating so-called “weak keys”.

This circumstance is the main reason that leads to further research, provided that the MAFG works in conjunction with the classic AFG, which ensure acceptable minimum values of the period for any values Y_0 .

Analyses of the statistical characteristics of the MAFG output signal allowed us to make the following conclusions.

Statistical characteristics at even values of h are much better than at odd values. This is due to a certain asymmetry in the formation of the MAFG output signal di at odd values of h .

In confirmation of this, Figure 6 shows statistical portraits of the MAFG output signal for $Y_0 = 7Y$.

The graphs show statistical portraits of two generators. Figure 6a shows that such a sequence does not meet the randomness requirements since it passed only 2 out of 15 tests (Cumulative Sums and Serial) (see Table 2) of the NIST set. Figure 6b shows that all values are within the confidence intervals; therefore, such a sequence meet the randomness requirements. Table 2 shows the results of the testing calculations.

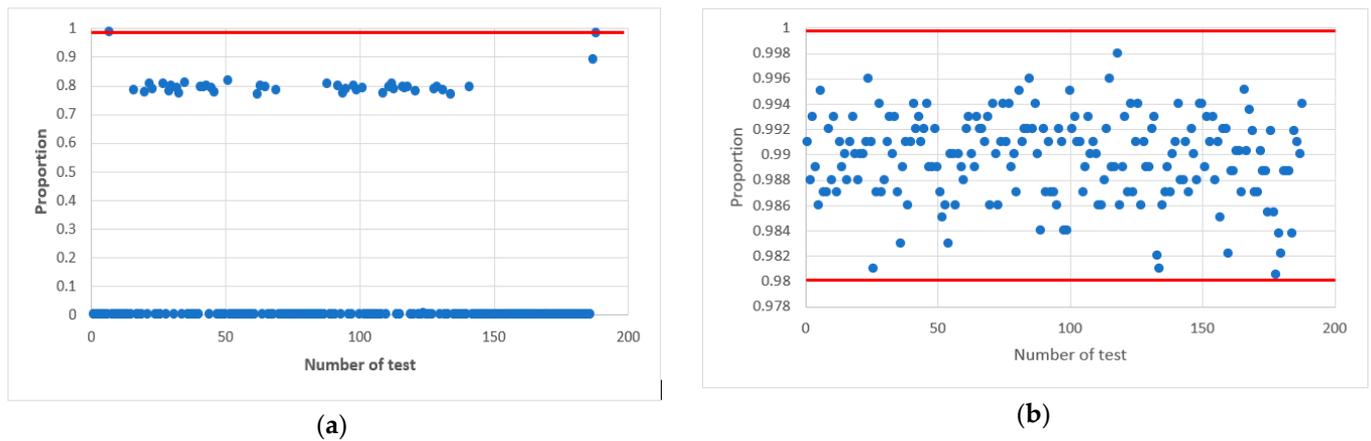


Figure 6. Statistical portrait of the MAFG output sequence: (a) with an odd value $h = 9$, (b) at an even value $h = 10$.

Table 2. Results of testing MAFG by NIST.

№	Statistical Test	Figure 6a		Figure 6b	
		p -Value	Status	p -Value	Status
1	Frequency	0.000000	-	0.291091	+
2	Block Frequency	0.000000	-	0.664168	+
3	Cumulative Sums	0.576961	+	0.950247	+
4	Runs	0.000120	-	0.377007	+
5	Longest Run	0.000750	-	0.350485	+
6	Rank	0.000106	-	0.757790	+
7	FFT	0.000132	-	0.192724	+
8	Non-Overlapping Template	0.000000	-	0.626709	+
9	Overlapping Template	0.000000	-	0.368587	+
10	Universal	0.000000	-	0.289667	+
11	Approximate Entropy	0.000000	-	0.096000	+
12	Random Excursions	0.000000	-	0.958867	+
13	Random Excursions Variant	0.000000	-	0.938062	+
14	Serial	0.128132	+	0.177628	+
15	Linear Complexity	0.000000	-	0.566688	+

It was also found that the statistical characteristics of the MAFG output signal improve when forming the output signal of the LC following the logic equation

$$a = a_0 \oplus a_r \quad (8)$$

where a_0 and a_r are the values of the least and most significant bits of register $RG1$ in the MAFG structure.

In confirmation of this, Figures 7 and 8 shows statistical portraits of the MAFG output signal for $h = 8$, variable initial values $Y_0 = 7, 32, 100$ and different values of the LC output signal: $a = a_0 \oplus a_1$ (Figure 7) and $a = a_0 \oplus a_2$ (Figure 8).

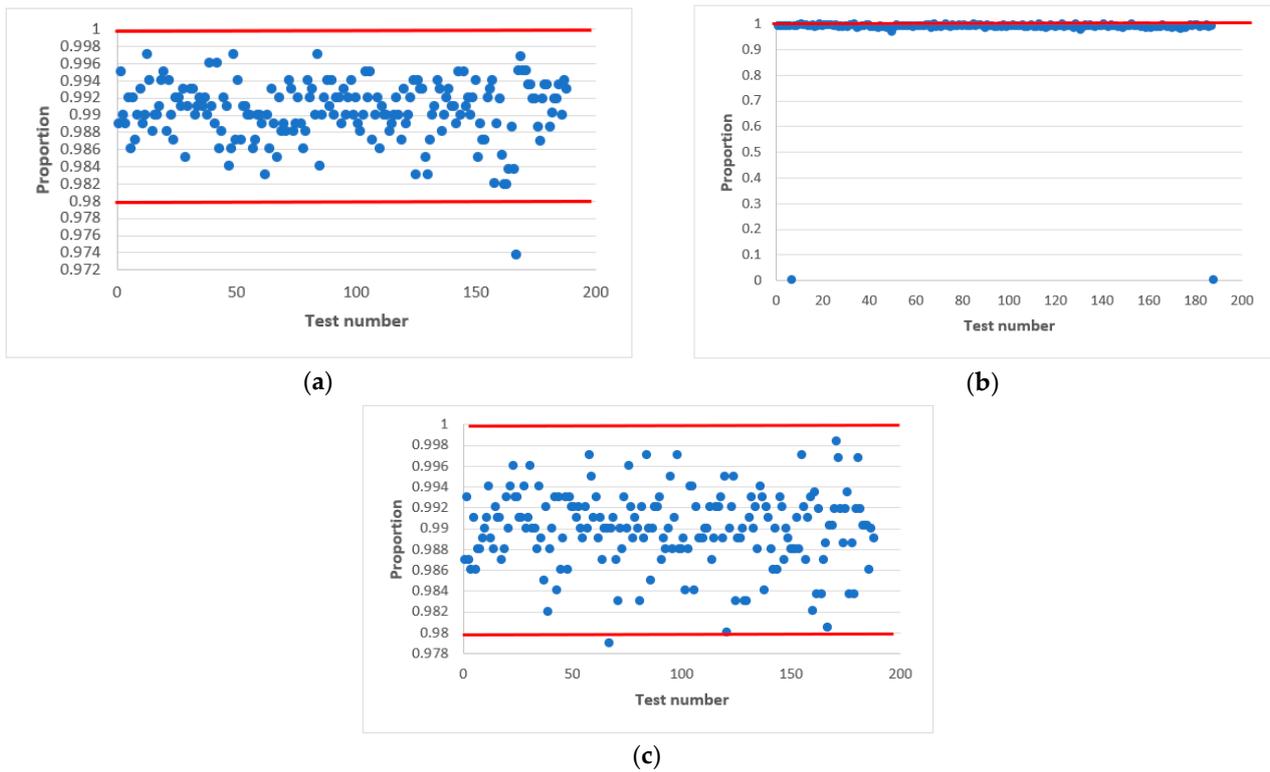


Figure 7. Statistical portrait of the MAFG output sequence for $h = 8$, $a = a_0 \oplus a_1$ with initial values: (a) $Y_0 = 7$, (b) $Y_0 = 32$, (c) $Y_0 = 100$.

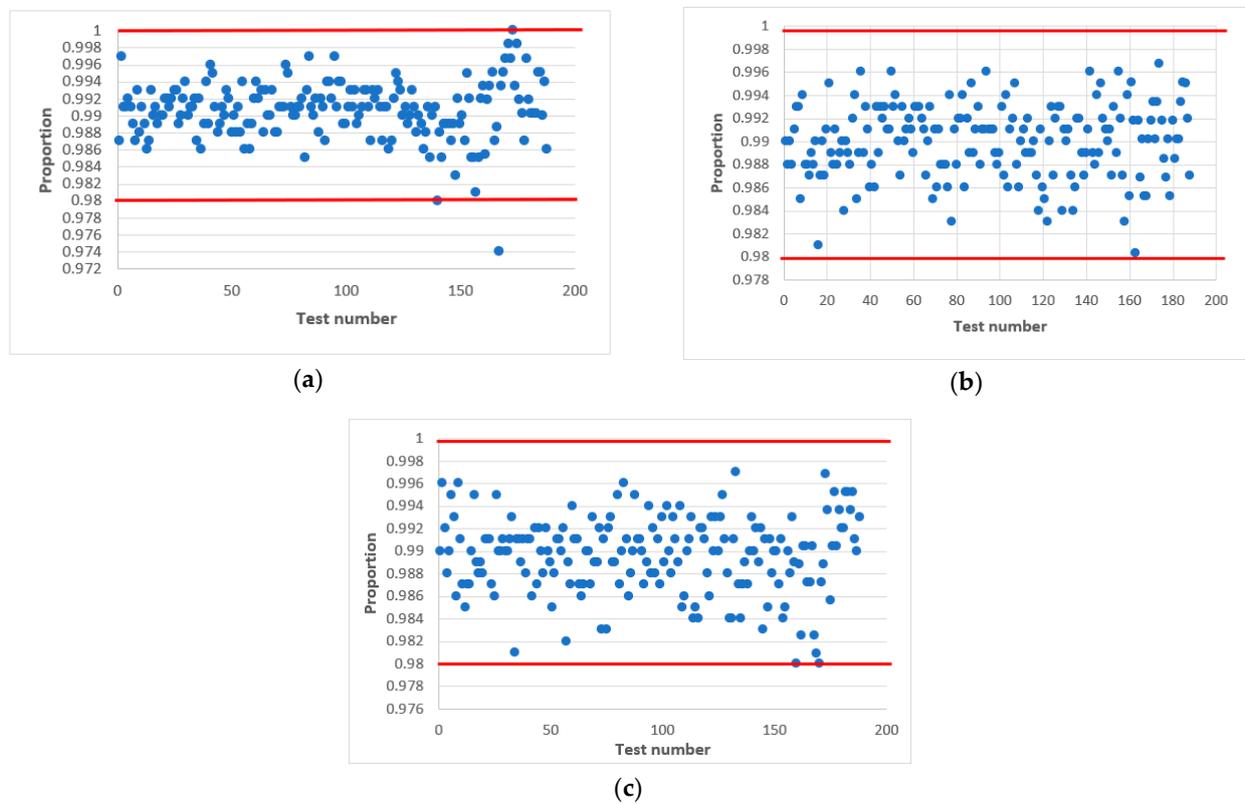


Figure 8. Statistical portrait of the MAFG output sequence for $h = 8$, $a = a_0 \oplus a_2$ with initial values: (a) $Y_0 = 7$, (b) $Y_0 = 32$, (c) $Y_0 = 100$.

Figure 7 shows that by using such initial values, the generator formed a non-random sequence. There are always one or two tests that fail the NIST tests.

Finally, we succeeded in passing all NIST tests at values of 32 and 100. Comparing Figures 7 and 8, we conclude that the quality of the pseudo-random sequence is affected by the value of the output signal of the $LS - a$.

Besides investigating the statistical characteristics of the MAFG, we determined the repetition periods. Table 3 shows some results of the investigation of the MAFG repetition period.

Table 3. Results of the investigation of the MAFG repetition period.

h	$a = 0$	T_{period}	$a = a_0 \oplus a_1$
6	$> 10^9$		$> 10^9$
7	109,531,200		$> 10^9$
8	4,193,300		$> 10^9$
9	193,443,432		$> 10^9$
10	$> 10^9$		$> 10^9$

Therefore, as a result of the research on MAFG statistical characteristics, it is established that they essentially depend on values h and a .

3.3. Results of Research Combined Steam Generator AFG and MAFG

This section presents the results of the research output sequence of pair AFG and MAFG (Figures 1 and 2), with a proven assumption that the improvement of the MAFG statistical characteristics for any initial value Y_0 can be achieved by forming the output signal of the logic circuit LC in such way that the least and most significant bits of the register RG_1 are fed to the input of the LC .

Table 4 shows the parameters of the logic circuit LC following logic Equation (4) for different values h , in which the statistical characteristics of the output signal of the combined generator bdi (Figures 1 and 2) were studied at different initial values X_0 and Y_0 . Figure 9 presents the results of these studies.

Table 4. Parameters of the combined generator.

h Value	LC Parameters	Initial Values
$h = 4$	$a = a_0 \oplus a_1$	
$h = 5$	$a = a_0 \oplus a_2$	
$h = 6$	$a = a_0 \oplus a_2$	$Y_0 = 7, 32, 100$ $X_0 = 7$
$h = 7$	$a = a_0 \oplus a_2$	
$h = 8$	$a = a_0 \oplus a_2$	
$h = 9$	$a = a_0 \oplus a_3$	
$h = 10$	$a = a_0 \oplus a_3$	

The obtained results (Figure 9) confirm the assumption that at even values of h and upon the formation of the LC output signal following Equation (4), the output signal of the combined generator successfully passes all 188 tests of the NIST set at different initial values of structural elements AFG and $MAFG$, which ensure its statistical security as one of the main components of cryptographic stability.

Figure 9 shows the results of testing the combined generator at $h = 10$.

Table 5 shows the results of passing the combined generator at $h = 10$, $a = a_0 \oplus a_3$ with initial values $X_0 = 7, Y_0 = 7$ (generator 1), $X_0 = 7, Y_0 = 32$ (generator 2), $X_0 = 7, Y_0 = 100$ (generator 3).

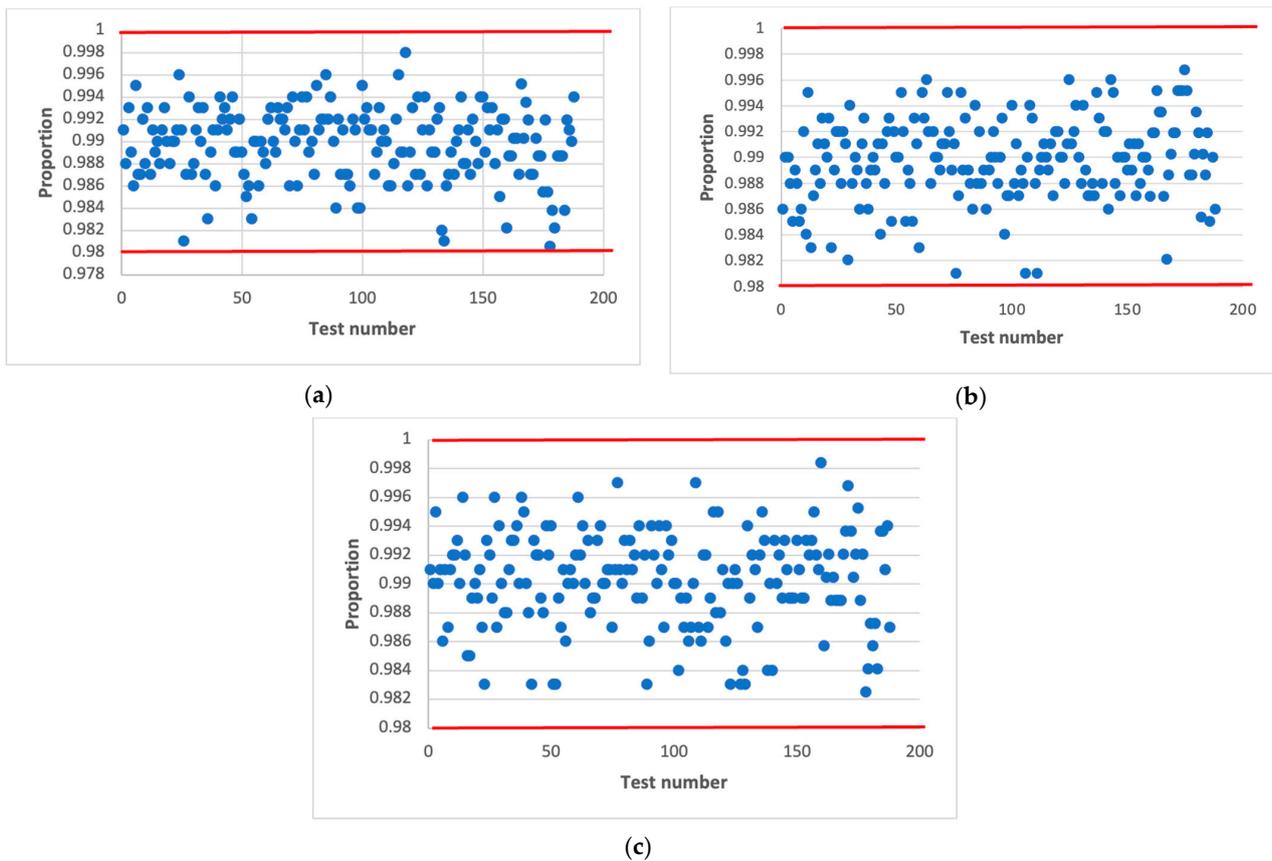


Figure 9. Statistical portrait of the combined generator output sequence for $h = 10, a = a_0 \oplus a_3$ with initial values: (a) $Y_0 = 7$, (b) $Y_0 = 32$, (c) $Y_0 = 100$.

Table 5. p -Values of statistical tests for combined generator.

№	Statistical Test	Generator 1		Generator 2		Generator 3	
		p -Value	Status	p -Value	Status	p -Value	Status
1	Frequency	0.943242	+	0.291091	+	0.291091	+
2	Block Frequency	0.044220	+	0.664168	+	0.664168	+
3	Cumulative Sums	0.576961	+	0.950247	+	0.950247	+
4	Runs	0.000000	+	0.377007	+	0.377007	+
5	Longest Run	0.000000	+	0.350485	+	0.350485	+
6	Rank	0.971006	+	0.757790	+	0.757790	+
7	FFT	0.128132	+	0.192724	+	0.192724	+
8	Non-Overlapping Template	0.000000	+	0.626709	+	0.626709	+
9	Overlapping Template	0.000000	+	0.368587	+	0.368587	+
10	Universal	0.000000	+	0.289667	+	0.289667	+
11	Approximate Entropy	0.000000	+	0.096000	+	0.096000	+
12	Random Excursions	0.000000	+	0.958867	+	0.958867	+
13	Random Excursions Variant	0.253205	+	0.938062	+	0.938062	+
14	Serial	0.000000	+	0.177628	+	0.177628	+
15	Linear Complexity	0.653773	+	0.566688	+	0.566688	+

Figure 10 shows the summary results of testing the combined generator with the parameters displayed in Table 5.

Figure 10 shows that combined generators with an even value of h meet the randomness requirements according to the NIST method in a wide range of initial values X_0 and Y_0 , and with an odd value, they do not. We decided to investigate this issue in more detail.

Further research proved that acceptable statistical characteristics of the output signal of the combined generator could also be provided with odd values of h , but in narrower ranges of initial values of X_0 and Y_0 . This is confirmed by the statistical portraits in Figure 11.

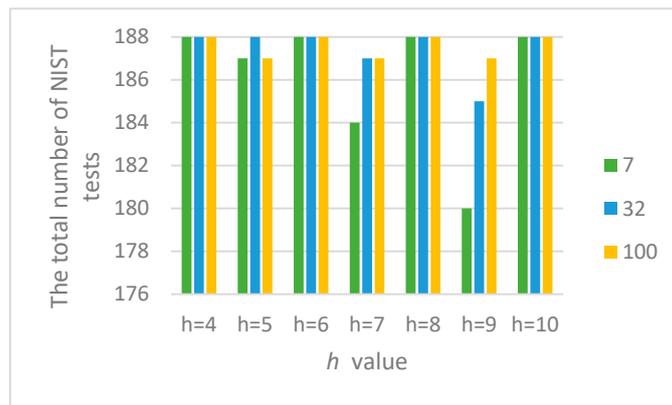


Figure 10. Results of passing the NIST tests with the combined generator.

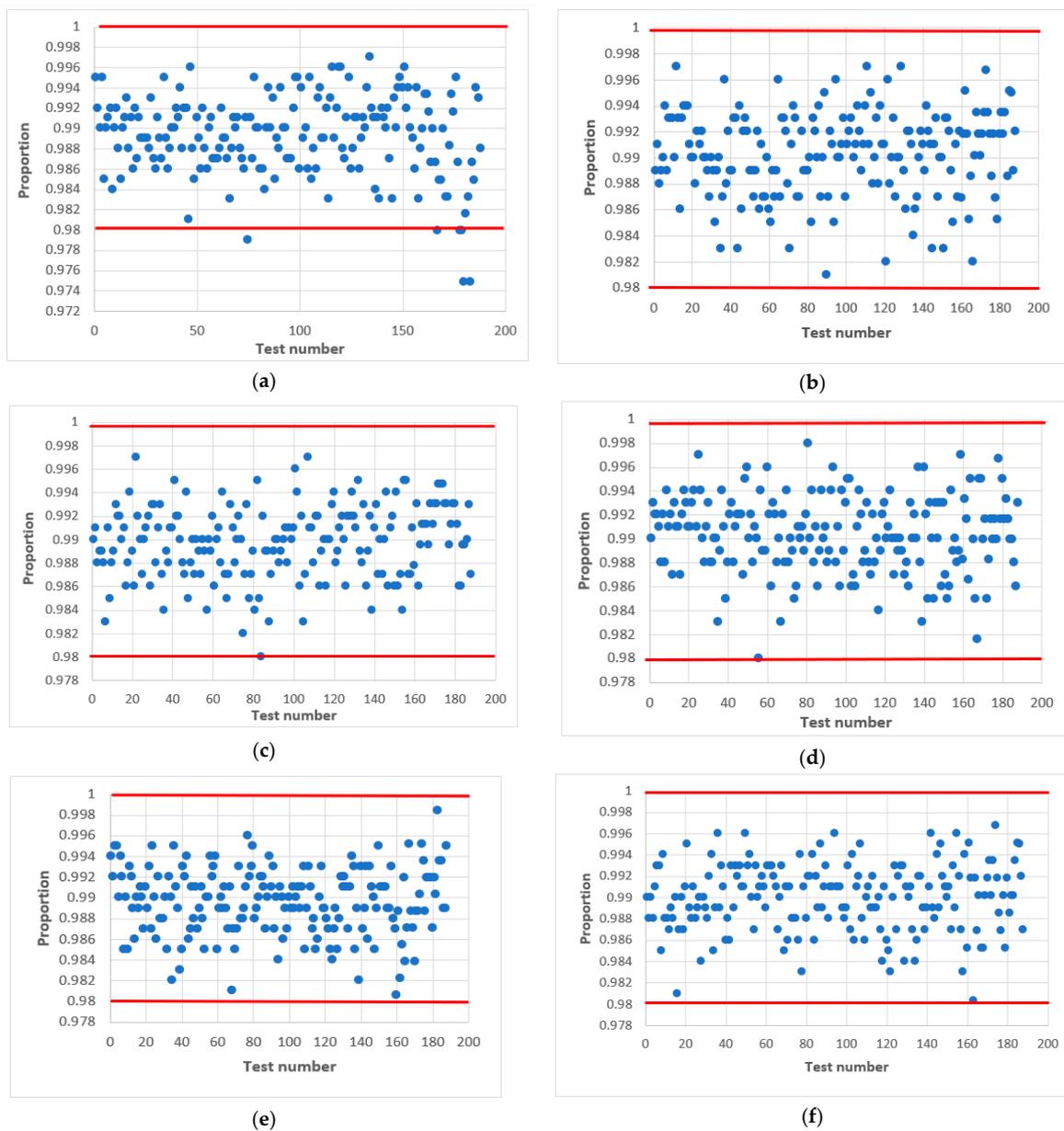


Figure 11. Cont.

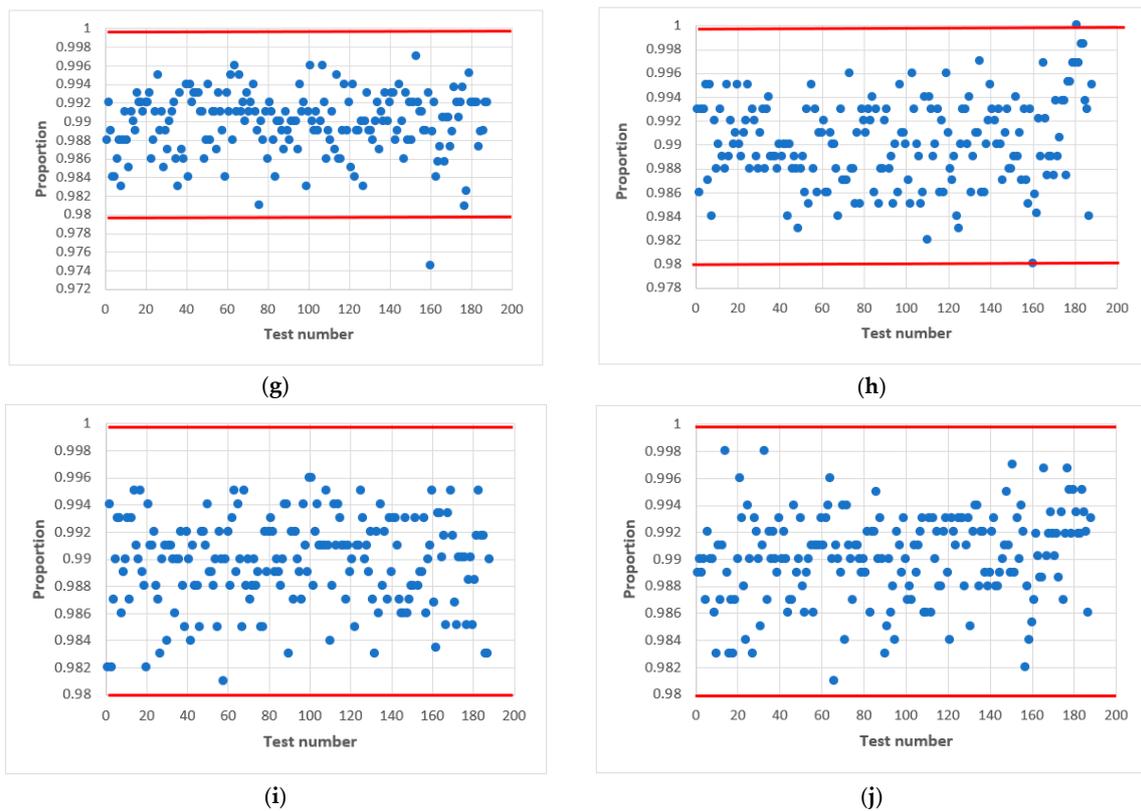


Figure 11. Statistical portrait of the combined generator for $h = 9$, $a = a_0 \oplus a_3$ with initial values: (a) $X_0 = 7, Y_0 = 36$; (b) $X_0 = 32, Y_0 = 36$; (c) $X_0 = 7, Y_0 = 64$; (d) $X_0 = 32, Y_0 = 64$; (e) $X_0 = 7, Y_0 = 71$; (f) $X_0 = 32, Y_0 = 71$; (g) $X_0 = 7, Y_0 = 100$; (h) $X_0 = 32, Y_0 = 100$; (i) $X_0 = 7, Y_0 = 1004$; (j) $X_0 = 32, Y_0 = 1004$.

As can be seen from Figure 10, the combined generator with $h = 9$ has the worst statistical characteristics. By doing additional research, such as varying the initial numbers in X_0 and Y_0 with the condition that $a = a_0 \oplus a_3$ we managed to achieve the passing of all NIST tests.

The results of the tests are shown in Figure 11, where the left side shows the combined generator with initial values $X_0 = 7$, and variables $Y_0 = 36$ (Figure 11a), 64 (Figure 11c), 71 (Figure 11e), 100 (Figure 11g), 1004 (Figure 11i). On the right side is the combined generator with $X_0 = 32$ and variables $Y_0 = 36$ (Figure 11b), 64 (Figure 11d), 71 (Figure 11f), 100 (Figure 11h), 1004 (Figure 11j). The value of the output signal of the LS for both cases is the same $a = a_0 \oplus a_3$.

The statistical portrait Figure 11 shows that the combined generator at $X_0 = 7, Y_0 = 32$ did not pass four NIST tests. This means that the sequence does not meet the randomness requirements. After changing the value of $X_0 = 32$, we see an improvement, and all tests are passed (Figure 11c). The situation is similar to the combined generator $Y_0 = 100$ (Figure 11g,h). Summarizing, we can conclude that the change of the initial numbers X_0 and Y_0 affects the quality of the statistical characteristics of the combined generator.

Thus, as a result of experimental research, we managed to choose the parameters of the combined generator so that it allows the formation of a sequence with acceptable statistical characteristics for odd h .

4. Conclusions

This work offers a new structure of the combined generator consisting of classical AFG and modified AFG – MAFG. The output pulse sequence of the combined generator is formed as the sum modulo 2 of the output sequences of AFG and MAFG.

At certain values of the AFG, internal parameters provide the maximum value of the repetition period of the output sequence for all possible initial settings X_0 of structural elements, thereby providing a repetition period of the combined generator output signal not less than a certain specified value.

It is proved that the statistical characteristics of the MAFG output signal and the combined generator are generally acceptable at odd values of the parameter h , determined by the number of bits of the MAFG registers, and the method of constructing a logic circuit, at which input signals come from the most and least significant bits of one of the MAFG registers. The statistical characteristics of the combined generator output signal are acceptable for a wide range of values of the initial settings AFG – X_0 and MAFG – Y_0 .

The scientific novelty of this paper lies in the further optimization of the MAFG structure with an arbitrary value of the modulus of the recurrent equation, the working principle of which was proposed by the authors in previous works, and the investigation of compatible work with the classical AFG. It allows for significantly improved basic characteristics of additive Fibonacci generators. An essential factor is that the development is aimed, first of all, at the hardware implementation, which allows the high-speed generators to be ensured.

The purpose of further research in this direction will be practical recommendations for the choice of parameters of the combined generator depending on the specific tasks of creating cryptographic or other technical means.

One possible direction for further research in this direction may be to investigate structures with more of AFG and MAFG combined in a different way.

Author Contributions: Conceptualization, V.M.; methodology, V.M., M.S., O.H., P.S. and T.Z.; software, M.S. and O.H.; validation, V.M., P.S. and R.S.; investigation, P.S., O.H., M.S., V.M., T.Z. and R.S.; writing—original draft preparation, V.M., M.S. and O.H.; writing—review and editing, V.M., M.S. and O.H.; funding acquisition, R.S. All authors have read and agreed to the published version of the manuscript.

Funding: The research work reported in this paper was in part supported by the National Centre for Research and Development, Poland under the project no. POIR.04.01.04-00-0048/20.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data cited in this manuscript are available from the published papers or the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Baldanzi, L.; Crocetti, L.; Falaschi, F.; Bertolucci, M.; Belli, J.; Fanucci, L.; Saponara, S. Cryptographically Secure Pseudo-Random Number Generator IP-Core Based on SHA2 Algorithm. *Sensors* **2020**, *20*, 1869. [[CrossRef](#)] [[PubMed](#)]
2. Dichtl, M.; Golić, J.D. High-Speed True Random Number Generation with Logic Gates Only. In *Cryptographic Hardware and Embedded Systems—CHES 2007*; Lecture Notes in Computer Science Book Series; Springer: Berlin/Heidelberg, Germany, 2007; pp. 45–62.
3. Mandrona, M.N.; Maksymovych, V.N. Comparative Analysis of Pseudorandom Bit Sequence Generators. *J. Autom. Inf. Sci.* **2017**, *49*, 78–86. [[CrossRef](#)]
4. Fishman, G.S. Pseudorandom Number Generation. In *Discrete-Event Simulation*; Springer Series in Operations Research; Springer: New York, NY, USA, 2001; pp. 416–451. [[CrossRef](#)]
5. François, M.; Defour, D.; Berthomé, P. A Pseudo-Random Bit Generator Based on Three Chaotic Logistic Maps and IEEE 754-2008 Floating-Point Arithmetic. In *Theory and Applications of Models of Computation—TAMC 2014*; Lecture Notes in Computer Science Book Series; Springer: Cham, Switzerland, 2014; Volume 8402. [[CrossRef](#)]
6. Barker, E.B.; Kelsey, J.M. *Recommendation for Random Number Generation Using Deterministic Random Bit Generators*, ITL Bulletin; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2007. Available online: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=919165 (accessed on 20 November 2022).

7. Shujun, L.; Xuanqin, M.; Yuanlong, C. Pseudo-random Bit Generator Based on Couple Chaotic Systems and Its Applications in Stream-Cipher Cryptography. In *Progress in Cryptology—INDOCRYPT 2001*; Springer: Berlin/Heidelberg, Germany, 2001; pp. 316–329. [[CrossRef](#)]
8. Kietzmann, P.; Schmidt, T.C.; Wählisch, M. A Guideline on Pseudorandom Number Generation (PRNG) in the IoT. *ACM Comput. Surv.* **2022**, *54*, 1–38. [[CrossRef](#)]
9. Orúe, A.B.; Hernández Encinas, L.; Fernández, V.; Montoya, F. A Review of Cryptographically Secure PRNGs in Constrained Devices for the IoT. In Proceedings of the SOCO 2017, ICEUTE 2017, CISIS 2017: International Joint Conference SOCO'17-CISIS'17-ICEUTE'17, León, Spain, 6–8 September 2017; Pérez García, H., Alfonso-Cendón, J., Sánchez González, L., Quintián, H., Corchado, E., Eds.; Advances in Intelligent Systems and Computing Book Series. Springer: Cham, Switzerland, 2018; Volume 649. [[CrossRef](#)]
10. Chen, J.; Miyaji, A.; Su, C. Distributed Pseudo-Random Number Generation and Its Application to Cloud Database. In *ISPEC 2014: Information Security Practice and Experience*; Huang, X., Zhou, J., Eds.; Lecture Notes in Computer Science Book Series; Springer: Cham, Switzerland, 2014; Volume 8434. [[CrossRef](#)]
11. Picek, S.; Yang, B.; Rozic, V.; Vliegen, J.; Winderickx, J.; De Cnudde, T.; Mentens, N. PRNGs for Masking Applications and Their Mapping to Evolvable Hardware. In *Smart Card Research and Advanced Applications—CARDIS 2016*; Lemke-Rust, K., Tunstall, M., Eds.; Lecture Notes in Computer Science Book Series; Springer: Cham, Switzerland, 2017; Volume 10146. [[CrossRef](#)]
12. De Bernardi, M.; Khouzani, M.H.R.; Malacaria, P. Pseudo-Random Number Generation Using Generative Adversarial Networks. In *ECML PKDD 2018: ECML PKDD 2018 Workshops*; Lecture Notes in Computer Science Book Series; Springer: Cham, Switzerland, 2019; Volume 11329. [[CrossRef](#)]
13. Gu, X.-C.; Zhang, M.-X. Design and Implementation of a FPGA Based Gaussian Random Number Generator. *Chin. J. Comput.* **2011**, *34*, 165–173. [[CrossRef](#)]
14. Park, S.; Kim, K.; Kim, K.; Nam, C. Dynamical Pseudo-Random Number Generator Using Reinforcement Learning. *Appl. Sci.* **2022**, *12*, 3377. [[CrossRef](#)]
15. Eastlake, D.; Schiller, J.; Crocker, S. *Randomness Requirements for Security*; The Internet Society: Reston, VA, USA, 2005; pp. 1–48.
16. Barker, E.B.; Kelsey, J.M. *Recommendation for Random Number Generation Using Deterministic Random Bit Generators*; Special Publication (NIST SP); National Institute of Standards and Technology: Gaithersburg, MD, USA, 2015. [[CrossRef](#)]
17. Gutterman, Z.; Pinkas, B.; Reinman, T. Analysis of the Linux random number generator. In Proceedings of the 2006 IEEE Symposium on Security and Privacy (S & P'06), Berkeley/Oakland, CA, USA, 21–24 May 2006; p. 15. [[CrossRef](#)]
18. Ruhault, S. SoK: Security Models for Pseudo-Random Number Generators. *IACR Trans. Symmetric Cryptol.* **2017**, *1*, 506–544. [[CrossRef](#)]
19. Håstad, J.; Impagliazzo, R.; Levin, L.A.; Luby, M. A Pseudorandom Generator from any One-way Function. *SIAM J. Comput.* **1999**, *28*, 1364–1396. [[CrossRef](#)]
20. Braverman, M.; Rao, A.; Raz, R.; Yehudayoff, A. Pseudorandom Generators for Regular Branching Programs. In Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science, Las Vegas, NV, USA, 13–16 October 2010; pp. 40–47. [[CrossRef](#)]
21. Degabriele, J.; Paterson, K.G.; Schuldt, J.C.N.; Woodage, J. Backdoors in Pseudorandom Number Generators: Possibility and Impossibility Results. In *Advances in Cryptology—CRYPTO 2016*; Robshaw, M., Katz, J., Eds.; Lecture Notes in Computer Science Book Series; Springer: Berlin/Heidelberg, Germany, 2016; Volume 9814. [[CrossRef](#)]
22. Barak, B.; Shaltiel, R.; Tromer, E. True Random Number Generators Secure in a Changing Environment. In *Cryptographic Hardware and Embedded Systems—CHES 2003*; Lecture Notes in Computer Science Book Series; Springer: Berlin/Heidelberg, Germany, 2003; Volume 2779, pp. 166–180. [[CrossRef](#)]
23. Saito, M.; Matsumoto, M.; Baccelli, E. *TinyMT32 Pseudorandom Number Generator (PRNG)*; Internet Engineering Task Force: Wilmington, DE, USA, 2020.
24. Maksymovych, V.N.; Mandrona, M.N.; Kostiv, Y.M.; Harasymchuk, O.I. Investigating the Statistical Characteristics of Poisson Pulse Sequences Generators Constructed in Different Ways. *J. Autom. Inf. Sci.* **2017**, *49*, 11–19. [[CrossRef](#)]
25. Cybulski, R. Pseudo-random number generator based on linear congruence and delayed Fibonacci method: Pseudo-random number generator based on linear congruence and delayed Fibonacci method. *Tech. Sci.* **2021**, *24*, 331–349. [[CrossRef](#)]
26. Mascagni, M.; Robinson, M.L.; Pryor, D.V.; Cuccaro, S.A. Parallel Pseudorandom Number Generation Using Additive Lagged-Fibonacci Recursions. In *Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing*; Niederreiter, H., Shiue, P.J.S., Eds.; Lecture Notes in Statistics Book Series; Springer: New York, NY, USA, 1995; Volume 106. [[CrossRef](#)]
27. Mascagni, M.; Srinivasan, A. Parameterizing parallel multiplicative lagged-Fibonacci generators. *Parallel Comput.* **2004**, *30*, 899–916. [[CrossRef](#)]
28. Parker, J.D. The period of the Fibonacci random number generator. *Discret. Appl. Math.* **1988**, *20*, 145–164. [[CrossRef](#)]
29. Orúe López, A.B.; Hernández Encinas, L.; Martín Muñoz, A.; Montoya Vitini, F. A Lightweight Pseudorandom Number Generator for Securing the Internet of Things. *IEEE Access* **2017**, *5*, 27800–27806. [[CrossRef](#)]
30. Bi, Y.; Peterson, G.D.; Warren, G.L.; Harrison, R.J. Poster reception—A reconfigurable supercomputing library for accelerated parallel lagged-Fibonacci pseudorandom number generation. In Proceedings of the 2006 ACM/IEEE Conference on Supercomputing—SC '06, Tampa, FL, USA, 11–17 November 2006. [[CrossRef](#)]
31. Zulfikar, Z.; Away, Y.; Shahnaz Noor, R. FPGA-based Design System for a Two-Segment Fibonacci LFSR Random Number Generator. *Int. J. Electr. Comput. Eng. (IJECE)* **2017**, *7*, 1882. [[CrossRef](#)]

32. Oduwole, H.; Shehu, S.; Adegoke, G.; Onubogu, J. Fibonacci Random Number Generator using Lehmer's Algorithm. *Math. Theory Model.* **2013**, *3*, 56–62.
33. Orue, A.B.; Montoya, F.; Hernández Encinas, L. Trifork, a New Pseudorandom Number Generator Based on Lagged Fibonacci Maps. *J. Comput. Sci. Eng.* **2010**, *2*, 46–51.
34. Kamikawa, N.; Umeo, H. A construction of five-state real-time Fibonacci sequence generator. *Artif. Life Robot.* **2016**, *21*, 531–539. [[CrossRef](#)]
35. Opoku-Mensah, E.; Abilimi, C.A.; Boateng, F.O. Comparative Analysis of Efficiency of Fibonacci Random Number Generator Algorithm and Gaussian Random Number Generator Algorithm in a Cryptographic System. *Comput. Eng. Intell. Syst.* **2013**, *4*, 50–57.
36. Srinivasan, A.; Mascagni, M.; Ceperley, D. Testing parallel random number generators. *Parallel Comput.* **2003**, *29*, 69–94. [[CrossRef](#)]
37. Bassham, L.; Rukhin, A.; Soto, J.; Nechvatal, J.; Smid, M.; Leigh, S.; Levenson, M.; Vangel, M.; Heckert, N.; Banks, D. *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. Special Publication (NIST SP)*; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2010. Available online: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=906762 (accessed on 21 November 2022).
38. Sýs, M.; Říha, Z. Faster Randomness Testing with the NIST Statistical Test Suite. In *Security, Privacy, and Applied Cryptography Engineering—SPACE 2014*; Chakraborty, R.S., Matyas, V., Schaumont, P., Eds.; Lecture Notes in Computer Science Book Series; Springer: Cham, Switzerland, 2014; Volume 8804. [[CrossRef](#)]
39. *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*; NIST SP 800-22 version 1a; NIST: Gaithersburg, MD, USA, 2010. Available online: <http://csrc.nist.gov/publications/nistpubs//SP80022rev1a.pdf> (accessed on 20 November 2022).
40. Maksymovych, V.N.; Mandrona, M.N.; Garasimchuk, O.I.; Kostiv, Y.M. A Study of the Characteristics of the Fibonacci Modified Additive Generator with a Delay. *J. Autom. Inf. Sci.* **2016**, *48*, 76–82. [[CrossRef](#)]
41. Maksymovych, V.; Harasymchuk, O.; Karpinski, M.; Shabatura, M.; Jancarczyk, D.; Kajstura, K. A New Approach to the Development of Additive Fibonacci Generators Based on Prime Numbers. *Electronics* **2021**, *10*, 2912. [[CrossRef](#)]
42. Maksymovych, V.; Shabatura, M.; Harasymchuk, O.; Karpinski, M.; Jancarczyk, D.; Sawicki, P. Development of Additive Fibonacci Generators with Improved Characteristics for Cybersecurity Needs. *Appl. Sci.* **2022**, *12*, 1519. [[CrossRef](#)]
43. Schneier, B. Special Algorithms for Protocols. In *Applied Cryptography*, 2nd ed.; Wiley: Hoboken, NJ, USA, 2015; pp. 527–557. ISBN 0471128457.
44. Gorbenko, A.; Popov, V. Reduction of the uncertainty in feature tracking. *Appl. Intell.* **2018**, *48*, 4626–4645. [[CrossRef](#)]
45. Guan, S.-U.; Zhang, S. Pseudorandom number generation based on controllable cellular automata. *Future Gener. Comput. Syst.* **2004**, *20*, 627–641. [[CrossRef](#)]
46. Mandrona, M.; Maksymovych, V.; Harasymchuk, O.; Kostiv, Y. Generator of pseudorandom bit sequence with increased cryptographic immunity. *Metall. Min. Ind.* **2014**, *6*, 24–28.
47. Maksymovych, V.N.; Harasymchuk, O.I.; Mandrona, M.N. Designing Generators of Poisson Pulse Sequences Based on the Additive Fibonacci Generators. *J. Autom. Inf. Sci.* **2017**, *49*, 1–13. [[CrossRef](#)]
48. Maksymovych, V.; Harasymchuk, O.; Opirskyy, I. The Designing and Research of Generators of Poisson Pulse Sequences on Base of Fibonacci Modified Additive Generator. In *International Conference on Theory and Applications of Fuzzy Systems and Soft Computing—ICCSEEA 2018: Advances in Computer Science for Engineering and Education, Warsaw, Poland, 27–28 August 2018*; Springer: Cham, Switzerland, 2018; Volume 754, pp. 43–53. [[CrossRef](#)]
49. Maksymovych, V.; Mandrona, M.; Harasymchuk, O. Dosimetric Detector Hardware Simulation Model Based on Modified Additive Fibonacci Generator. *Adv. Intell. Syst. Comput.* **2020**, *938*, 162–171. [[CrossRef](#)]