Journal of Engineering Technology and Applied Sciences, 2021 *e-ISSN:* 2548-0391
Received 1 May 2021, Accepted 24 August 2021 *Vol. 6, No. 2, 91-109*
Published online 30 August 2021, Research Article *doi: 10.30931/jetas. 931101*

Citation: Hadj-Brahim, A., Ali-Pacha, H., Ali-Pacha, A., Hadj-Said, N., "Cohabitation of Fibonacci and Galois Modes in One Linear Feedback Shift Register". Journal of Engineering Technology and Applied Sciences 6 (2) 2021 : 91-109.

COHABITATION OF FIBONACCI AND GALOIS MODES IN ONE LINEAR FEEDBACK SHIFT REGISTER

Abderrahmene Hadj Brahim, Hana Ali-Pacha, Adda Ali-Pacha*, Naima Hadj-Said

Laboratory of Coding and Security of Information
University of Sciences and Technology of Oran, Algeria Usto,
PoBox 1505 El M'Naouer Oran 31000 Algeria
hadj_94@hotmail.com {hana.alipacha, naima.hadjsaid}@univ-usto.dz,
*a.alipacha@gmail.com(*corresponding author)*

Abstract

A linear feedback shift register (LFSR) is the basic element of the pseudo-random generators used to generate a sequence of pseudo-random values for a stream cipher. It consists of several cells; each cell is a flip-flop and a feedback function. The feedback function is a linear polynomial function; this function has a degree equal to the number of cells in the register. The basic elements of the register are connected to each other in two different ways, either in Fibonacci mode or in Galois mode.

In this work, we propose the realization of a specific register which is cohabitation of these two modes (Fibonacci and Galois) in the same register and for the same feedback function, and which will be controlled by a random function for the selection of mode, which will be based on the chaotic logistics map. This specific register gave better results compared to registers with separate modes.

Keywords: LFSR, cryptography, stream cipher, Pseudo-Random generation, Fibonacci mode, Galois mode, logistic map

1. Introduction

The stream cipher is becoming increasingly important in our daily lives[1]–[4]. Some of these systems use Linear Feedback Shift Register (LFSR) to produce a random sequence, which must be XORed with plaintext massages to give encrypted messages. LFSR is an electronic device (flip-flops connected in series)[5] that can be seen as software that produces a sequence of bits that can be seen as a recurring sequence on the Galois filed F_2 of 2 elements (0 and 1).

An LFSR is characterized by its feedback function, which connects its cells to each other in two different modes: Fibonacci and Galois. The most natural mode is the so-called Fibonacci mode. This is called because the Fibonacci sequence is represented in this mode. It updates the first cell of the register, which is on the left then operates by shift for the other cells. In 2002, Goresky and Klapper introduced a completely different mode called the Galois mode [6]. The basic idea of the Galois mode is that the contents of the output cell are re-injected into the input cell and added to the contents of the other cells of the register, and then all the cells are shifted to the output.

The critical difference between the two modes is how the feedback polynomial is interpreted. For a given feedback polynomial, the two modes produce different output sequences. Another difference is their implementation, if you implement an LFSR in a CPU or an FPGA [7], the Galois structure that is more computerized and its cells are updated simultaneously is probably faster and has less latency than Fibonacci mode.

In the last few years, several algorithms have been used LFSR to generate pseudo random sequences such as [8]–[11]. Lv et Tong [8] Proposed a novel method of chaotic image encryption based on LFSR. In this algorithm, The LFSR is combined with chaotic system to generate the key stream where the polynomial of the LFSR has degree of 20. Ayoup et al [9] proposed an efficient selective image encryption. The LFSR in this algorithm is used to generate a matrix where the polynomial of the LFSR has degree of 8. After that this matrix is XORed with the Plain image to obtain the pre-encrypted image. Kareem Jumaa [10] designed a digital image encryption using AES and random number generator. To encrypt the plain image using AES, The LFSR is used to generate 128 bits of the input of the AES. The polynomial of the LFSR in this algorithm has degree of 5. Naim et al [11] proposed a novel satellite image encryption algorithm based on hyperchaotic systems and Josephus problem. The LFSR in this algorithm is used to generate a matrix which used to generate 512 bits of the hash function. Then, these 512 bits are used to generate the key of the whole algorithm. The polynomial of the LFSR in this algorithm has degree of 8. The main disadvantage in these algorithms that the LFSR has a low max period which cause a repetition in the value of the sequence of the LFSR.

In order to improve the period and the complexity of the LFSR sequence, we propose the realization of a specific register, which is cohabitation of these two modes (Fibonacci and Galois) in an LFSR and for the same feedback polynomial; it will be controlled by a random function of mode selection, in our case, based on the logistics map. This specific register gave better results compared to the register with separate modes.

2. Linear feedback shift register

A linear feedback shift register is an electronic device that produces a sequence of random bits [12]. It consists of several cells each cell presented by a flip-flop, the particularity of the connection between cells called feedback function, looks like a linear function, so we can apply a mathematical structure. The LFSRs (Figure 1) have the following characteristics:

- ✓ The output is the contents of the last cell on the right.
- ✓ The period is the sequence length produced before it begins to repeat itself.
- ✓ The feedback function is the sum of "exclusive" operations of some of the bits of the register, the list of these bits is called "the derivation sequence". It is obtained from a chosen polynomial.

✓ They are easy to make in hardware [13].

2.1 How an LFSR works

LFSRs are used as pseudo-random number generators [5], [12]. When properly configured, they reach periods of maximum length; each state will be reached only once until all states are reached. Once each state has been reached, the period will be repeated [13].
In general, LFSRs are built with D flip-flops and, XOR operations. The initial value of the shift register and the feedback function determines the order of output [12].

An LFSR of length L is composed of a shift register containing a sequence of L bits (S_i,..., S_{i+L-1}) and a linear feedback function, as well as a clock controlling the movement data [12].

At each clock tick, the content of the rightmost cell is the output of the register and, the contents of the other cells are shifted to the right, a new bit is calculated by the feedback function and will be placed in the leftmost cell of the register:

$$S_{t+L} = C_1 S_{t+L-1} + C_2 S_{t+L-2} + \cdots + C_{L-1} S_{t+1} + C_L S_t \quad (1)$$

The coefficients C_i are binary.

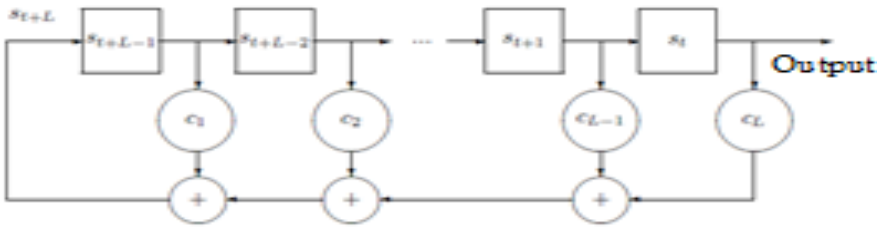

Figure 1. LFSR of length L.

Definition 2.1 [6]: An LFSR whose feedback function is given by the relation:

$$S_{t+L} = C_1 S_{t+L-1} + C_2 S_{t+L-2} + \cdots + C_{L-1} S_{t+1} + C_L S_t \quad (2)$$

Its feedback polynomial f is the following F_2[X] polynomial:

$$f(x) = 1 + C_1 x + C_2 x^2 + \cdots + C_{L-1} x^{L-1} + C_L x^L \quad (3)$$

The sequence $(S_n)_{n \geq n_0}$ is produced by an LFSR whose feedback polynomial is: f (x) If and only if its formal serial development:

$$S(x) = \sum_{n \geq 0} S_n x^n \quad (4)$$

Is written:

$$S(x) = \frac{g(x)}{f(x)} \quad (5)$$

Where g is a polynomial of F_2[X] such that deg(g) < deg(f), and gcd(g_0,f_0)=1. In addition, the polynomial g is determined by the initial state of the register :

$$g(x) = \sum_{i=0}^{L-1} x^i \sum_{j=0}^{i} C_{i-j} S_j \quad (6)$$

It may be noted that such a sequence is ultimately periodic, that is to say, that there exists a pre-period n_0 such that the sequence ($(S_n)_{n \geq n_0}$ is periodic of period $T \leq 2^L - 1$ (there exists an integer i_0 such that $S_i = S_i$+T for all $i \geq i_0$) [6].

The LFSRs have been studied since 1930 in their purely theoretical aspect and are mostly built on a finite field. From 1948 to 1969, LFSRs are used as generators of pseudo-random sequences in cryptosystems since they can generate binary sequences of maximum period. These sequences are called the m-sequences (maximum length sequences). The search for sequences with a very long period becomes a crucial problem in the 1950s. To ensure better security, we must respect three characteristics, in addition to having a maximum period, the m-sequences must verify all the random postulates that give them a good random quality. A pseudo-random generator used by cryptography must be able to [1]:

- Generate sequences of bits that must satisfy the statistical characteristics of truly random sequences.
- To guarantee that if an attacker knows all, or part of the sequence encrypting $S_0, S_1, \dots, S_i$ it is difficult (from a computational point of view) to find the seed.

2.2 Presentation of LFSR modes: Fibonacci & Galois

2.2.1 Fibonacci mode

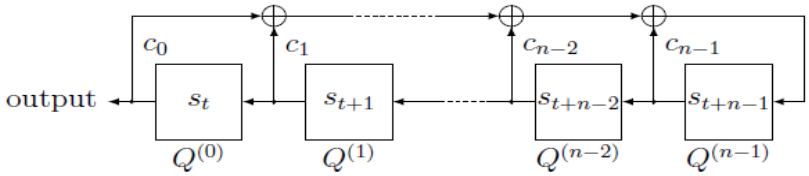

Figure 2. LFSR Fibonacci mode

The Fibonacci model LFSR or just Fibonacci LFSR, named after the 12th century Italian mathematician Leonardo Fibonacci, is the most common used model of LFSR [7], [12] because of it is simple and lightweight hardware implementation which make it so popular in cryptographic systems. A Fibonacci LFSR consists of a number of stages that shift their values with each iteration and a feedback polynomial that returns new values in the first stage, see Figure 2. The memory stages who evaluated by the feedback polynomial are called the taps or tapped positions.

The output sequence S of the LFSR of Figure 2 satisfies the linear recurrence:

$$S_{t+n} = C_{n-1}S_{t+n-1} + C_{n-2}S_{t+n-2} + +C_1S_{t+1} + C_0S_t \quad (7)$$

The feedback polynomial of the LFSR is equivalent to the inverse of the characteristic polynomial of the linear recurrence sequence above:

$$f(x) = 1 + C_{n-1}x + \cdots + C_1x^{n-1} + C_0x^n \quad (8)$$

A Fibonacci LFSR of length n [12], with a feedback polynomial f (x) that updates at each clock interval of t in equation (9).

$$Q_{t+1}{}^{(i)} = \begin{cases} C_{n-1}Q_t{}^{(n-1)} + +C_1Q_t{}^{(1)} + C_0Q_t{}^{(0)} \dots \ if\ i = n-1 \\ Q_t{}^{(i-1)} \ \ Else \end{cases} \quad (9)$$

An example of a Fibonacci mode LFSR with a feedback polynomial [8] $f(x) = \boldsymbol{X^8 + X^4 + X^3 + X^2 + 1}$, is shown in Figure 2.1.

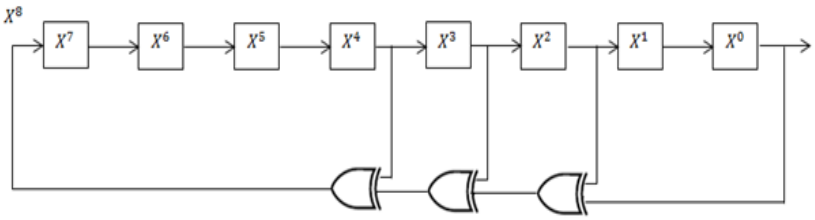

Figure 2.1. LFSR Fibonacci $X^8 + X^4 + X^3 + X^2 + 1$

LFSR Fibonacci is the most common model of LFSR used in systems that require the generation of a pseudo-random sequence. It is these qualities with its simple hardware implementation that make it very popular in cryptographic systems.

2.2.2 Galois mode

The Galois model, named after the 19th century French mathematician Evariste Galois ***[6], [12]***. The function and basic idea behind this model is the same as the Fibonacci model. The critical difference is in how the feedback polynomial is interpreted.

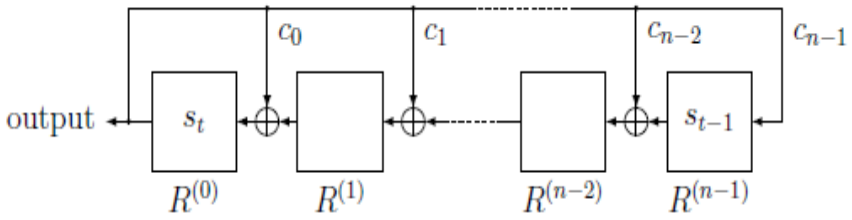

Figure 3. LSFR Galois Mode

An LFSR in Galois configuration, which is also known as **modular**, **internal XORs** as well as **one-to-many LFSR**, is an alternate structure that can generate the same output stream as a conventional LFSR. In the Galois configuration, when the system is clocked, bits that are not tapped are shifted one position to the right unchanged. The taps, on the other hand, are XORed with the output bit before they are stored in the next position. The new output bit is the next input bit. The effect of this is that when the output bit is zero all the bits in the register shift to the right unchanged, and the input bit becomes zero. When the output bit is one, the bits in the top positions all flip (if they are 0, they become 1, and if they are 1, they become 0), and then the entire register is shifted to the right and the input bit becomes 1. The feedback polynomial of the LFSR Galois is defined:

$f(x) = x^n + C_{n-1}x^{n-1} + \cdots + C_1 x + C_0,$

(10)

A Galois LFSR of length *n* [6], with feedback polynomial f (x) that updates at each clock interval of t in equation 11.

$$R_{t+1}{}^{(i)} = \begin{cases} C_{n-1}R_t{}^{(0)}, & if\ i = n-1 \\ R_t{}^{(i+1)} + C_i R_t{}^{(0)} & else \end{cases}$$

(11)

An example of a Galois mode LFSR with a feedback polynomial [14] $f(x) = \boldsymbol{X^8 + X^4 + X^3 + X^2 + 1}$, is shown in Figure 3.1.

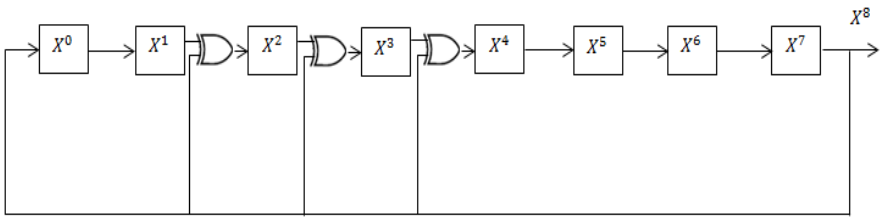

Figure 3.1. LFSR Galois $X^8 + X^4 + X^3 + X^2 + 1$

3. Proposed system: Coexistence of the two modes in one LFSR

We will design a system that will combine the two modes (Fibonacci and Galois) in one register and which will be controlled by a random value T (T = 0 one chooses the mode Galois and if T = 1 one chooses the mode Fibonacci). This value T can be the output of any generator or, as in our case, a value Y_n of the logistic map.

3.1 Logistic Map

Logistics map [15], [16] is a well-known dynamic in non-linear systems theory, defined by equation (12):

$$y_{k+1} = r\, x_k\, (1 - x_k) \quad (12)$$

It gives a perfect explanation of dynamic system behavior. This system was developed by Prof. Pierre François Verhulst (1845) to measure the evolution of a population in a limited environment, later used in 1976 by the biologist Robert May to study the evolution of insect population:

- y_{k+1}: Generation in the future that is proportional to x_k.
- x_k: Previous generation.
- r: Positive constant incorporates all factors related to reproductive, successful overwintering eggs for example, etc.

In order to study this dynamic system and some asymptotic individuals' models, the first thing to do is to draw the parabolic graph $y = r\, x\, (1 - x)$, and the diagonal $y = x$.

The operation that we will follow to draw the iterative form y_{k+1} according to x_k is simply summarized as follows:

- Starting from an initial value x_0 of the x-axis, we reach the function with a vertical; the function takes the value $y_1 = r\, x_0\, (1 - x_0)$,
- From horizontal $y_1 = r\, x_0\, (1 - x_0)$, of the previous point, we join the line $y = x$;
- We represent the abscissa of the intersection with the vertical line $x=x_0$; we have $y_1 = x_1$
- From the x_1 value of the x-axis, we reach the function with a vertical; the function takes the value $y_2 = r\, x_1\, (1 - x_1)$,; and so on.

We take $r = 3.9$ and, $x_0=0.01$ for the logistics map, the previous operations for 100 iterations are represented in Figure 4.

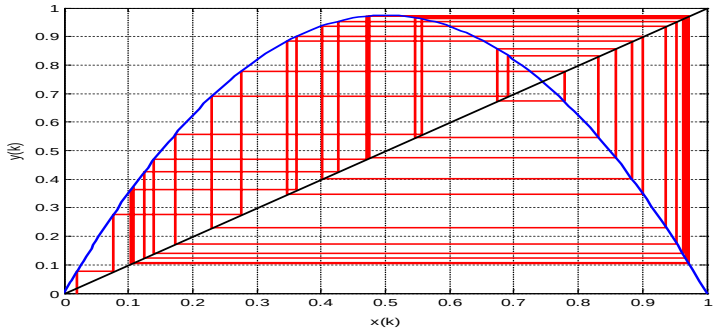

Figure 4a. Evolution of y_k in function of x_k

Figure 4 shows two signals y_k generated from the logistic map in chaotic mode (r = 3.9), one with an initial condition x_0 = 0.1 and the other with x_0= 0.100000000000001 very close to 0.1.

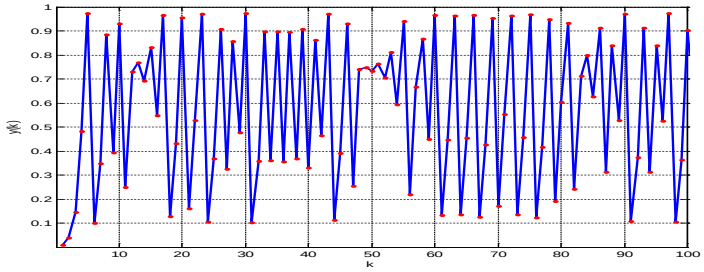

Figure 4b. Chaotic regime in the function of *k*

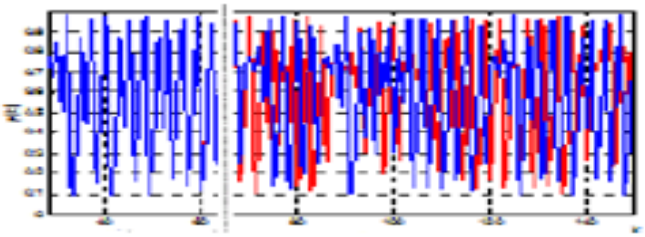

Figure 4c. Sensitivity to initial conditions

We note that a very small error on the knowledge of the initial state x0 in the phase space will be rapidly amplified and gives us two widely different signals. Quantitatively, the growth of error is locally exponential for highly chaotic systems (sensitivity to initial conditions).
It should be noted that the initial condition error in this case is 10^{-15} and this is the smallest value because Matlab works with only 52 bits but the system can be sensitive to smaller values than 10^{-15} depending on the work environment.

3.2 Diagram of "Cohabitation mode"

The scheme of the Fibonacci / Galois "mode cohabitation" is proposed with the generating polynomial [14]:$f(x) = \boldsymbol{X^8 + X^4 + X^3 + X^2 + 1}$.

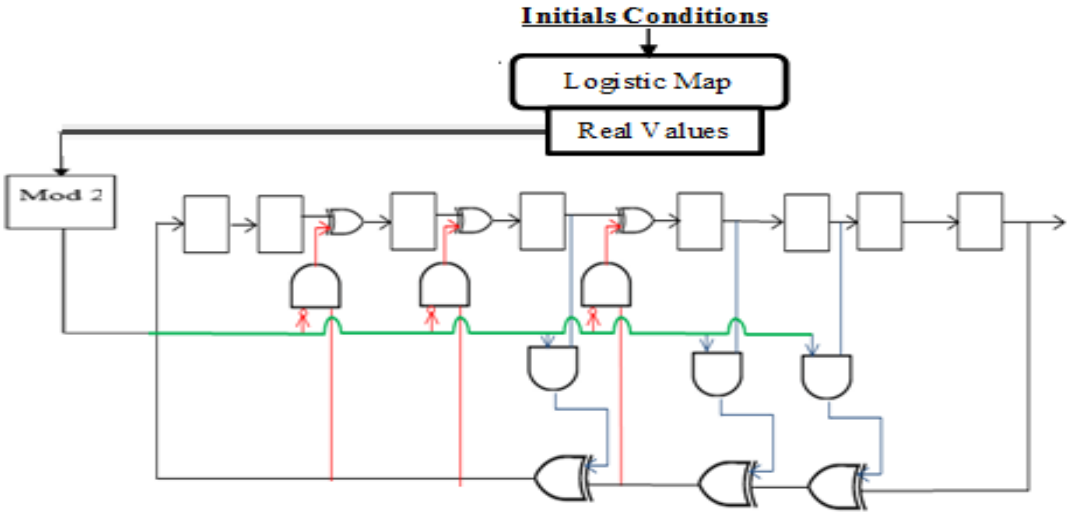

Figure 5. LFSR in cohabitation mode with $X^8 + X^4 + X^3 + X^2 + 1$

The result of (Mod 2) is equal to either 0 or 1, if (Mod 2) = 0 one chooses the mode Galois and, if (Mod 2) = 1 one chooses the mode Fibonacci.

4. Results and interpretations

One of the big problem in the pseudo random generator is to determine whether the generator is random or not, since there are no fix or universal tests that can evaluate the random of generator. One of the best test is to the study the properties of the numbers it generates, a good random generator should generate a sequence of numbers with properties of unpredictability and independence and follows a certain distribution (uniform in cryptography, Gaussian in telecommunications, etc.) [17], [18]. Another evaluation of the random quality of a generator is the control of the properties of the sequence that it generates by compare the performance of the generator studied with to theoretical ones.

We will present some tests used to evaluate the performance of our generators such as entropy test, average test, or spectral test.

A. Characteristics of the working computer

The application was created from a PC HP pavilion 15 Notebook:

- ✓ Memory: 4096MB RAM
- ✓ Processor : intel® Core™ i3-3120M CPU @ 2.50GHZ
- ✓ Operating System: Windows 7 Ultimate 32-bit Edition
- ✓ Graphics Card: Intel® HD Graphics 4000
- ✓ Total memory ≈ 2734 MB

For the implementation of our application, we used the C/C++ programming language.

B. Tests of different generators with different modes

We will take the following values as initial conditions of the logistic map (eq.12), and this is valid throughout the paper:

1. X_0=0.1, μ=3.9999, F=10^7.

We multiply the result of the logistic sequence with F and take the integer part (by the function floor) and we do the modulo 2 which will give a result equal to either 1 or 0.

$$T_n = Mod\ (F * \mu X_{n-1}(1 - X_{n-1}), 2) \quad (13)$$

In addition, we will take the following assumptions:

2. The feedback function is : $\boldsymbol{f(x) = X^8 + X^4 + X^3 + X^2 + 1}$
3. The seed of the cells of the register is 10101010.

4.1 T_n vector of the logistic map

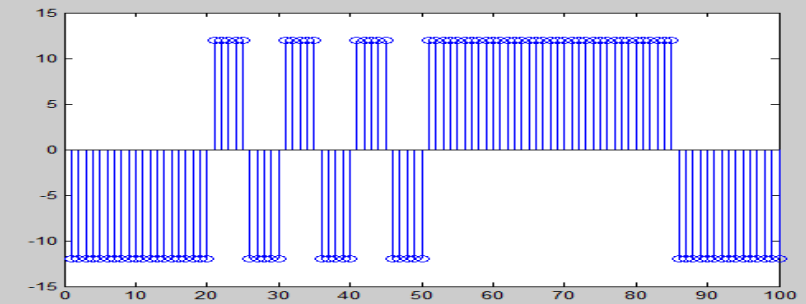

Figure 6. Representation of T_n as a NRZ signal

Figure. 6 represents the first 100 values of the logistic sequence T_n resulting from equation 13 in the form of a representation of an NRZ signal. The 70000 values generated by Equation 13 are summarized in Table 1.

Table 1. T_n Values Generated by Equation 13

	T_n =1	T_n =0
Number	35100	34900
Rate in %	50.14%	49.85%

This specific register of the cohabitation of the two modes (Fibonacci and Galois), uses the Fibonacci mode as well as the Galois mode.

4.2 Status of flip-flops of the register

For a seed =10101010, the LFSR with the polynomial $\boldsymbol{f(x) = X^8 + X^4 + X^3 + X^2 + 1}$ which is a primitive polynomial of degree 8, has a maximum period equal to **255** ($= \boldsymbol{2^8 - 1}$) in both modes: Fibonacci and Galois.

To read the states of the LFSR flip-flops, read the values line by line.

We start with the first line and, from left to right, then and the second line always from left to right, and so on until the end of the last value in the last line of the matrix, we return to the first value of the first line.

Table 2a. Status of flip-flops register in Fibonacci mode

$$\begin{pmatrix}
170-213-234-245-250-125-62-159-79-167-83-41-20-10-133-66\\
33-144-200-228-242-249-252-254-255-127-63-31-15-135-67-161\\
208-232-244-122-61-30-143-199-99-177-88-44-22-11-5-2\\
1-128-64-32-16-136-196-226-113-56-28-142-71-35-145-72\\
164-210-133-116-58-29-14-7-3-129-192-96-48-152-76-38\\
147-73-36-146-201-100-178-217-236-118-59-157-78-39-19-9\\
4-130-65-160-80-168-212-106-181-218-109-182-91-173-214-107\\
53-154-77-166-211-105-52-26-13-134-195-225-240-248-124-190\\
223-111-183-219-237-246-123-189-94-175-215-235-117-186-93-46\\
23-139-69-34-17-8-132-194-97-176-216-108-54-27-141-198\\
227-241-120-60-158-207-231-115-57-156-206-103-51-25-140-70\\
163-209-104-180-90-45-150-75-37-18-137-68-162-81-40-148\\
74-165-82-169-84-42-149-202-229-114-185-220-238-119-187-221\\
110-55-155-205-230-243-121-188-222-239-247-251-253-126-191-95\\
47-151-203-101-50-153-204-102-179-89-172-86-43-21-138-197\\
98-49-24-12-6-131-193-224-112-184-92-174-87-171-85
\end{pmatrix}$$

Table 2b. Status of flip-flops register in Galois mode

$$\begin{pmatrix}
170-85-146-73-156-78-39-171-237-206-103-139-253-198-99-137\\
252-126-63-167-235-205-222-111-143-255-199-219-213-210-105-140\\
70-35-169-236-118-59-165-234-117-130-65-152-76-38-19-177\\
224-112-56-28-14-7-187-229-202-101-138-69-154-77-158-79\\
159-247-195-217-212-106-53-162-81-144-72-36-18-9-188-94\\
47-175-239-207-223-215-211-209-208-104-52-26-13-190-95-151\\
243-193-216-108-54-27-181-226-113-128-64-32-16-8-4-2\\
1-184-92-46-23-179-225-200-100-50-25-180-90-45-174-87\\
147-241-192-96-48-24-12-6-3-185-228-114-57-167-82-41\\
172-86-43-173-238-119-131-249-196-98-49-160-80-40-20-10\\
5-186-93-150-75-157-246-123-133-250-125-134-67-153-244-122\\
61-166-83-145-240-120-60-30-15-191-231-203-221-214-107-141\\
254-127-135-251-197-218-109-142-71-155-245-194-97-136-68-34\\
17-176-88-44-22-11-189-230-115-129-248-124-62-31-183-227\\
201-220-110-55-163-233-204-102-51-161-232-116-58-29-182-91\\
149-242-121-132-66-33-168-84-42-21-178-89-148-74-37
\end{pmatrix}$$

The values (Table 2) of the states of the flip-flops of the register different from one mode to another.

Table 3. Period of the cohabitation mode for different seeds

Initial state of the register	Length of the period		
	Fibonacci	Galois	Cohabitation
10101010	255	255	19990
11001110			19990
10010011			470
01001101			19990
11100010			753
00001111			451

Table 3 gives the length of the period of the specific register in mode cohabitation for different seeds. The results presented are satisfactory and show that an LFSR in mode cohabitation has a much longer period than a separate LFSR mode (Fibonacci or Galois).

4.3 Frequency test

The most natural test is that of the frequencies of occurrence of each digit, for a real random sequence, a particular number has no reason to be more or less represented than another. In other words, the frequencies of each digit must eventually come close to 10%. Obviously, the same thing is expected from our pseudo-random generator: this is the first test to which it is subjected.

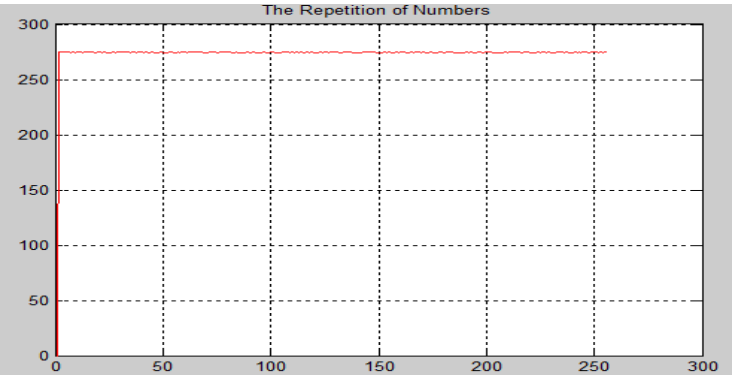

Figure 7. Frequencies of occurrence of each value from 0 to 255 for LFSR cohabitation mode

It can be seen in the graph of figure 7 that all the values from 0 to 255 are represented approximately with the same frequency of occurrence. Therefore, our generator passes the frequency test.

4.4 Entropy test

Shannon entropy is a mathematical function that intuitively corresponds to the amount of information contained or delivered by a source of information. For a source, which is a discrete random variable *X* comprising n symbols, each symbol x_i having a probability P_i to appear, the entropy H of the source X is defined as:

$$H(x) = -\sum_{i=1}^{n} P_i . log_2 (P_i) \quad (14a)$$

We pose $$P_i = \frac{k_i}{n} \quad (14b)$$

With i varying from 0 to 255, and n and the number of values generated in our case (n = 256 * 256 = 65536), k_i corresponds to the occurrence frequency of each number i. A logarithm based on 2 is usually used because the entropy then has the bit/symbol units. On the other hand, consider a source that has an alphabet of 256 characters. If all these characters are equiprobable, the entropy associated with each character is **$\log_2(256) = \log_2(2^8) = 8$ bits**, which means that it takes 8 bits to transmit a character thus, its entropy is equal to **8 bits**.

Table 4. Value of LFSR entropy for different modes

Nature of Source	Entropy in bits/symbols		
	Fibonacci	Galois	Cohabitation
Entropy	**7.994353**	**7.994351**	**7.990974**

The ratio that of the source is 99.88% of a source that delivers equiprobable characters. Therefore, our generator passes the entropy test.

4.5. Mean, variance, and autocorrelation factor tests

We must test the distribution of the numbers produced in the sequence in its interval of operation, we have calculated the three operators: the mean, the variance, and the autocorrelation function of these numbers. In the ideal case [19], and for a random variable μ which follows a uniform distribution over an interval [0; 1], the following three values must be found:

- **Mean of the numbers** (15a)
  $\bar{u} = \frac{1}{n}\sum_{i=1}^{n} u_i = \frac{1}{2} = 0.5$
- **Variance of the numbers** (15b)
  $$v = \frac{1}{n}\sum_{i=1}^{n} u_i^2 - \bar{u}^2 = \frac{1}{12} = 0.0833 \ldots$$
- **Autocorrelation Factor**(14c)
  $$E(u_i u_{i+1}) = \frac{1}{n}\sum_{i=1}^{n=1} u_i\, u_{i+1} = \frac{1}{4} = 0.25$$
  In our case, one pose:
  $\boldsymbol{u_i = \frac{x_i}{m}}$, ∀ i = 1, ..., n,
  (16)

m : The largest value or the value of the modulo.

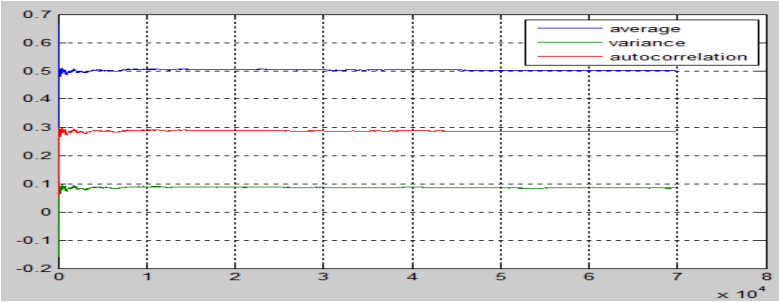

Figure 8. Mean, Variance and autocorrelation factor Tests for LFSR en cohabitation de mode

The results obtained are close to the ideal case results. Figure 8 confirms us that, the numbers generated by LFSR in cohabitation mode, have a random behavior.

4.6. Spectral analysis

Knuth describes [19] the spectral test as the most discriminating of all. Indeed, no proven bad generator could succeed. Very simple, the method consists in studying the distribution of the values generated in a dimension k (2D or 3D) to check the quality. In fact, all generators suffer from a Marsaglia effect (this is because we do not generate all the real numbers, but only fractions are generated).
In general, the spectral test makes it possible to determine the deviation d between two lines. At the most, this gap is small at most the generator is of good quality.

Dimension 2 (2D): Two consecutive values will be the coordinates of a point on the plane. One looks if; the points are uniformly distributed in a square.

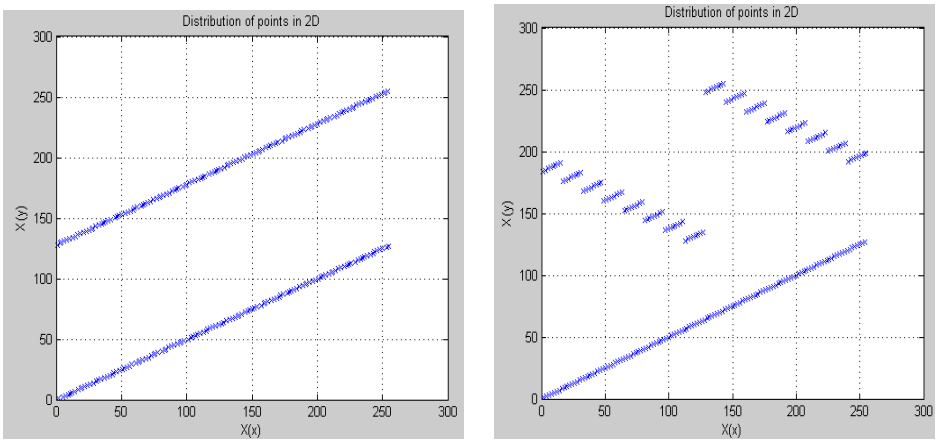

Figure 9a. Spectral Test of dimension 2D for LFSR respectively in, Fibonacci and Galois

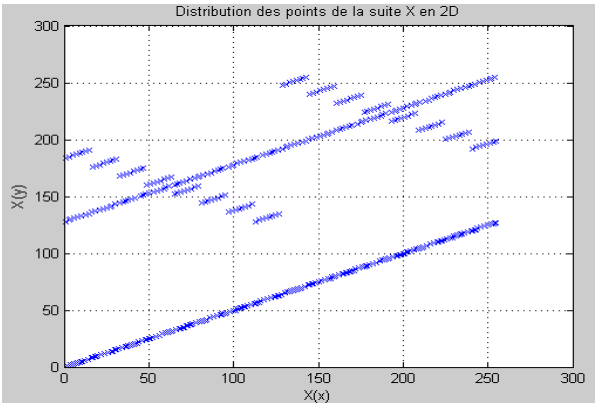

Figure 9b. Spectral Test of dimension 2D for LFSR en cohabitation de mode

Dimension 3 (3D): Three consecutive values will be the coordinates of a point in space. One looks if; the points are distributed evenly in a cube. By turning the cube, one sees the undesirable effect: plans of Marsaglia. It can be seen below that the points are located on planes.

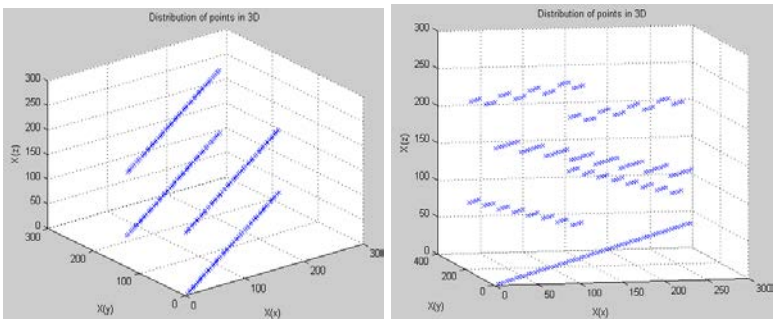

Figure 10a. Spectral Test of dimension 3D for LFSR respectively in Fibonacci and Galois

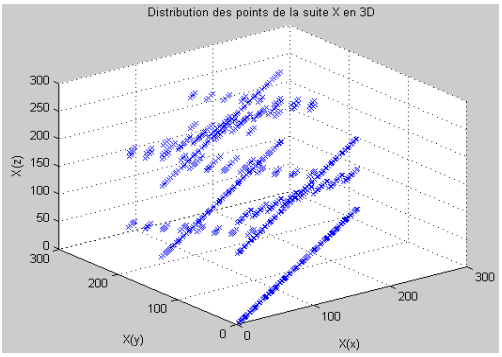

Figure 10b. Spectral Test of dimension 3D for LFSR en cohabitation de mode

- The tests of 3D and 2D give a distribution of the values on spaced lines (we speak of the Marsaglia effect), and they are not distributed in the entire surface.

We note that the LFSR in mode cohabitation (Figures 9b and 10b) inherits LFSR spectra from two separate modes (Figures 9a and 10a).

4.5. Stream cipher with LFSR in cohabitation mode

We propose to use the stream cipher [20] which consists to produce a key stream sequence, generated an LFSR in cohabitation mode, which will be XORed to the plaintext. Therefore, the cipher text will be obtained.

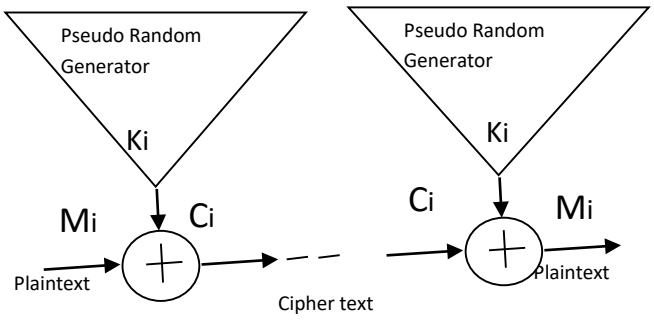

Figure 11. Stream Cipher

$\mathbf{m_i}$: : plaintext, $\mathbf{c_i}$: cipher text, $\mathbf{k_i}$: key stream generated by LFSR in cohabitation mode. This type of generator generates key streams k_1, k_2, k_3, ..., k_i. These key streams are XORed to the plaintext m_1, m_2, m_3,..., m_i, to produce the cipher text [20].

$$c_i = m_i \oplus k_i \qquad (17a)$$

For the decryption, the cipher text is XORed with an identical keystreams, to retrieve the plaintext:

$$m_i = c_i \oplus k_i = (m_i \oplus k_i) \oplus k_i = m_i \qquad (17b)$$

Any continuous synchronous encryption algorithm uses keys (secrets) and generates the same key stream [20] used for encryption and decryption, this flow is generated independently of the flow of the message.

4.6. Implementation

We propose to use LFSR in cohabitation mode for stream cipher, to generate a data stream that will be XORed with an image. We take the following assumptions:

1. The feedback function is : $\boldsymbol{f(x) = X^8 + X^4 + X^3 + X^2 + 1}$
2. Initial conditions of the logistic sequence: the seed of the cells of the register is 11001110.
3. X(0)=0.1, μ=3.9999, F=10^7.
4. The image is "Cameraman" of 256 * 256 pixels.
5.

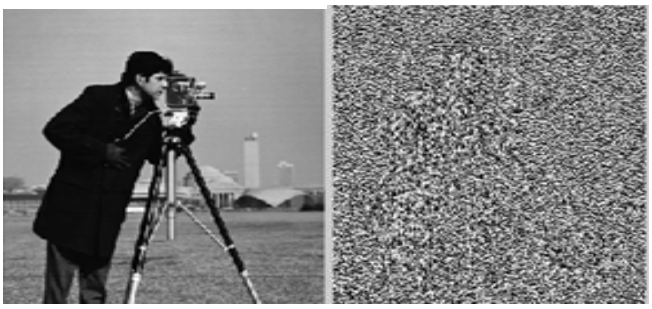

Figure 12. Plaintext and Encrypted Image

Histogram of the Images [21], [22]**:** For a monochrome image, that is to say with a single component, the histogram is defined as a discrete function that maps to each value intensity, the number of pixels of this value. The determination of the histogram is carried out by counting the number of pixel intensity for each of image. The histogram can then be seen as a probability density. The histograms are resistant to a number of transformations on the image. They are invariant to rotations and translations, and to a lesser extent to changes of point of view, and to changes of scale. Referring to the results obtained, we can clearly see that the plaintext image differs substantially from the corresponding ciphered one. Moreover, the histogram of the ciphered image is uniform which makes it difficult to extract the pixels statistical nature of the plaintext image.

The histograms of the plaintext and the cipher images of the "cameraman" show that the proposed cryptosystem works in a correct way.

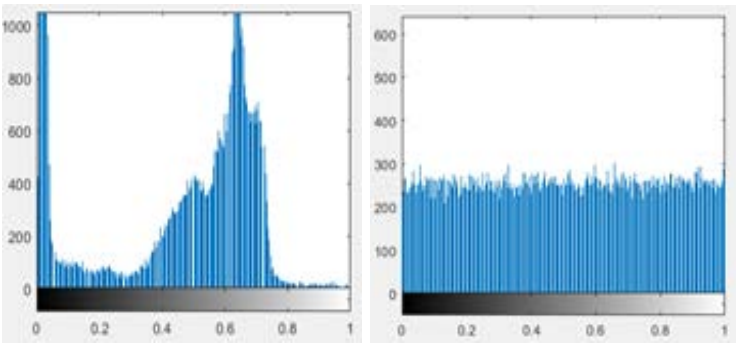

Figure 13. Histogram of plaintext and encrypted images

Entropy: From figures of the histogram of encrypted image, one note has a uniform histogram, which means that the gray levels have the same number of occurrences and hence the entropy is the maximum. Therefore, a grayscale image, where each pixel is represented by 8 bits, must have entropy for the encrypted image, the closest possible 8 bits/pixel.

Table 5. The entropy of the ciphered images

Cameraman Image	Entropy of Images	
	Plaintext	Cipher
Entropy	**7.009716**	**7.997222**

The obtained value is very close to the theoretical one (99.97%). Referring to the results, we can clearly see that the plaintext images differ significantly from her corresponding encrypted. Moreover, the histogram of the encrypted images is quite uniform which makes it difficult for the statistical extraction of pixels of the plaintext image.

The computation of the entropy of the images encrypted by the LFSR in cohabitation mode reveals that, the proposed crypto-system functions in a correct way.

Correlation of the Adjacent Pixels [23]: In probability and in statistics, to study the correlation between two random variables or numerical statistics is to study the strength of the bond that can exist between these variables. The searched link is an affine relationship, it is the linear regression. For example, we calculate the correlation coefficient between two sets of the same length (typical case: a regression). Assume we have the following table of values: $X(x_1, \ldots, x_n)$ and $Y(y_1, \ldots, y_n)$ of each of the two series.

A measure of this correlation is obtained by calculating **the linear correlation coefficient of Bravais-Pearson** [23]**.**

For the correlation coefficient linking these two sets, we apply the following formula:

$$Coef(X,Y) = \frac{cov(X,Y)}{\sqrt{D(X).\sqrt{DY}}} \quad (18a)$$

Covariance between x and y is given as follows:

$$cov(X,Y) = \frac{1}{N}\sum_{i=1}^{N}\big((X_i - E(X)\big).(Y_i - E(Y))) \quad (18b)$$

The average of X is :

$$E(X) = \frac{1}{N}\sum_{i=1}^{N} X_i \quad (18c)$$

The average of Y is :

$$E(Y) = \frac{1}{N}\sum_{i=1}^{N} Y_i \quad (18e)$$

The standard deviation of X is :

$$D(X) = \frac{1}{N}\sum_{i=1}^{N}(X_i - E(X))^2 \quad (18f)$$

The standard deviation of Y is :

$$D(Y) = \frac{1}{N}\sum_{i=1}^{N}(Y_i - E(Y))^2 \quad (18g)$$

The correlation coefficient is between -1 and 1. Intermediate values provide information on the degree of linear dependence between two variables. The closer the coefficient is close to extreme values -1 and 1, the closer the correlation between variables is strong we simply use the term "highly correlated" to describe the two variables. A correlation equal to 0 means that the variables are not correlated. To test the correlation coefficient, we selected randomly **1500 pairs** of two adjacent pixels in both encrypted and clear pictures.

Value of the pixel at the position (x, y)

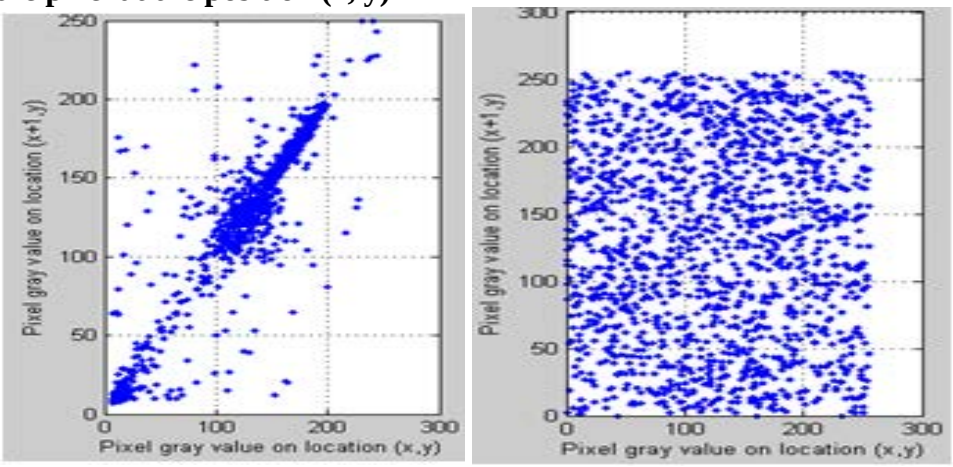

Figure.14. Correlation between the horizontally adjacent pixels of respectively the plaintext image and the encrypted image

Figure 14 shows the correlation between two horizontally adjacent pixels of the image clear and encrypted. We see that the neighboring pixels in the image have a clear correlation (**coeff = 0.95324**), while the encrypted will have one little correlation (**coeff = -0.0048**). This low correlation between two neighboring pixels in the encrypted image makes the attack of our cryptosystem difficult.

In addition, it is clear that in the image clear, several lines can be adjusted to scatter but among all these lines can be retained which has a remarkable property giving rise to the right of the form $Y = a\,X + b$ representing a **linear correlation**.

5. Conclusion

We have created a new mode for LFSR generators "cohabitation mode" that connects two modes (Fibonacci and Galois) together in the same LFSR and based on the logistic map. The length of the periodicity of this generator in "cohabitation mode" is much greater than those of the separate modes.

We have tested this generator with (Frequency Test, Entropy Test, Average Test, Variance Test, Spectral Test, ...) The results are satisfactory. We also encrypted an image with this generator and tested this encryption. The results of the encryption tests are satisfactory.

However, the implementation of the proposed LFSR need more hardware than the classical ones.

Secret key field: In the proposed algorithm, the secret key field is set as follows:

$$ST = \{x_0, \mu, F, K, D, T\}.$$

The initial state of the logistics map x_0=0.1, μ=3.9999, F=10^7, the encryption key can be represented by the following fields:

- ✓ x_0,
- ✓ μ,
- ✓ F : scalar
- ✓ K : starting point or the starting moment k, where we begin to do the encryption.
- ✓ D: Initial state of the register (8 flip-flops = 8 bits)
- ✓ T Mode (Fibonacci / Galois), Fibonacci mode T=0 or T=1 (1 bit)

Where x_0, μ, are double-precision numbers. K are integer constants. If the precision of calculating x_0, μ, is 10^{-16}, and K ∈ [1, 1000].

Therefore, the key space is larger than 2^8x 2^1x 10^{16} x 10^{16} x 10^7 x 10^3=10^{42} (with $10^3 \approx 2^{10}$) in this case we will have a key field of the order of 2^{149}.

We have 149 bits larger of key, this number is huge. Therefore, the encryption algorithm has a very large key space to withstand all kinds of brute force attacks.

References

[1] Schneier, B., "Applied cryptography-protocols, algorithms and source code in C", John Wiley & Sounds, Inc, New York, Second Edition, (1996),.

[2] Menezes, A. J., Oorschot, P. C. V., Vanstone, S. A., "Handbook of applied cryptography" (1997) by CRC Press LLC.

[3] Gutub, A. A.-A., Al-Haidari, F., Al-Kahsah, K. M., Hamodi, J. "e-Text watermarking: utilizing "Kashida" extensions in Arabic language electronic writing", J. Emerg. Technol. Web Intell. 2(1) (2010) : 48‑55.

[4] Gutub, A., Al-Qurashi, A., "Secure shares generation via m-blocks partitioning for counting-based secret sharing", J. Eng. Res. 8(3) (2020) : 91‑117.

[5] George, M., Alfke, P., "Linear feedback shift registers in virtex devices (application note)", http://www.xilinx.com/bvdocs/appnotes/xapp210.pdf.

[6] Goresky, M., Klapper, A., "Fibonacci and galois representations of feedback withcarry shift registers", IEEE Transactions on Information Theory, 48(11) (2002) : 2826-2836.

[7] Stackoverflow, "Galois vs Fibonacci LFSR, more computer-friendly but what else?", [online], nov 2011.

[8] Lv, Y., Tong, X., "A novel method of chaotic image encryption based on LFSR", in 2009 International Conference on Management and Service Science, Beijing, China, sept. (2009) : 1‑4. doi: 10.1109/ICMSS.2009.5302775.

[9] Ayoup, A. M., Hussein, A. H., Attia, M. A. A., “Efficient selective image encryption”, Multimed. Tools Appl. 75(24) (2016): 17171‑17186.
[10] Kareem Jumaa, N., “Digital Image Encryption using AES and Random Number Generator”, Iraqi J. Electr. Electron. Eng. 14(1) (2018) : 80‑89.
[11] Naim, M., Ali Pacha, A., Serief, C., “A novel satellite image encryption algorithm based on hyperchaotic systems and Josephus problem”, Adv. Space Res. 67(7) (2021) : 2077‑2103.
[12] Golomb, S. W., “Shift register sequences”, Aegean Park Press, Laguna Hills, CA, (1982).
[13] Nyathi, J., Delgado-Frias, J. G., Lowe, J., “A high-performance, hybrid wave-pipelined linear feedback shift register with skew tolerant clocks,” 46th IEEE Midwest Symposium on Circuits and Systems, Cairo, Egypt, In Press, Dec. (2003).
[14] Mirella, A. M., Stratulat, M., “Study of software implementation for linear feedback shift register based on 8th degree irreducible polynomials”, International Journal Of Computers 8 (2014) : 46-55.
[15] Devaney, R. L., “A first course in chaotic dynamical systems theory and experiment”, New York: Westview Press, 1992. Aug 09, 2021. [online]: http://search.ebscohost.com/login.aspx?direct=true&scope=site&db=nlebk&db=nlabk&AN=421133.
[16] Gleick, J., “Chaos: Making a New Science”, Albin Michel edition, (1987).
[17] Al-Roithy, B. O., Gutub, A. A., “Trustworthy image security via involving binary and chaotic gravitational searching within PRNG selections”, Int. J. Comput. Sci. Netw. Secur. 20(12) (2020) : 167‑176.
[18] Al-Qurashi, A., Gutub, A., “Reliable secret key generation for counting-based secret sharing”, J. Comput. Sci. Comput. Math. 8(4) (2018) 87‑101.
[19] Knuth, D.E., “The art of computer programming”, Addison-Wesley, Reading, MA, third edition, (1998).
[20] Berbain, C., “Analysis and design of stream algorithm Analysis and design of stream algorithm - in French language”, PhD thesis, University Paris 7. Diderot, supported on 10.2007, (2007).
[21] Gutub, A., Al-Shaarani, F., “Efficient implementation of multi-image secret hiding based on LSB and DWT steganography comparisons”, Arab. J. Sci. Eng. 45(4) (2020) : 2631‑2644.
[22] AlKhodaidi, T., Gutub, A., “Refining image steganography distribution for higher security multimedia counting-based secret-sharing”, Multimed. Tools Appl. 80(1) (2021) : 1143‑1173.
[23] Chen, G., Mao, Y., Chui, C. K., “A symmetric image encryption scheme based on 3D chaotic cat maps”, Chaos Solitons Fractals 21(3) (2004) : 749‑761.