A New Random Keys Generator Depend on Multi Techniques

Dr. Alaa kadhim
Computer Sciences Department, University of Technology/Baghdad.
Email:Dralaa_cs@yahoo.com
Hussein Abed
Computer Sciences Department, University of Technology/Baghdad.

Revised on:12/5/2014 & Accepted on: 8/1/2015

ABSTRACT

A stream cipher is a symmetric cipher which operates with a time-varying transformation on Individual plaintext digits. By contrast, block ciphers operate with a fixed transformation on large blocks of plaintext digits. Where in operation of Key generator using the LFSR to generate random keys where the shift register is controlled by an external clock. At each time unit, each digit is shifted one stage to the right. The content of the rightmost stage st is output. The new content of the leftmost stage is the feedback bit, st+L.

this paper present design and implementation system of keys generator with nonlinear random of output keys and large moment bits, this system consist of three part from registers , logical circuits and search algorithm using AI work in parallel time to find the result, the LFSRs in registers part (left and right of system)can generate huge moment of bits and pass in randomness test, those bits by the logical circuit and search algorithm convert from linear to nonlinear to get more difficult in break secret keys, finally the keys can used in stream cipher or in block cipher methods.

مولد مفاتيح عشوائي جديد يعتمد على نقنيات متعدده

الخلاصه :

ان التشفير الانسيابي هي عمليه تشفير تناظريه والتي تعمل على اوقات مختلفه لتحويل النصوص . وعلى العكس من ذلك ان التشفير الكتلي يعمل على اساس تشفير الكتل من النصوص . وان عمليه توليد المفاتيح باستخدام التغذيه العكسيه لتوليد مفاتيح عشوائيه خيث ان عمليه التزحيف يتم التحكم بها على اساس اخر بت بحث ان كل وقت يتم تزحيف بت الى الجانب الايمن بينما ان محتويات الجانب الايمن تعتمد على اساس النتيجه او المخرجات . والمحتويات الجديده تعتمد على اساس التغذيه العكسيه.

هذا البحث يقدم عمليه تصميم وتنفيذ نظام لتوليد مفاتيح عشوائيه لاخطيه مع اختلاف المفاتيح العشوائيه وكميه هائله من البتات ان النظام يتكون من جزئين من خلال المسجلات وجزء الدوائر المنطقيه وخوارزميات البحث التي تعمل بشكل متوازي لايجاد النتائج. ان التغذيه العكسيه في المسجلات (للجانب الايمن والايسر) ممكن ان يولد عدد هائل من البت وهي ناجحه باستخدام اختبار العشوائيه هذه البتات بالاعتمادعلى الدوائر المنطقيه وخوارزميات البحث تحول من خطيه الى لاخطيه للحصول على مفتاح سري صعب الاختراق او التدمير والذي من الممكن ان يستخدم في التشفير الانسيابي والتشفير على شكل كتل .

Keywords: cipher, decipher, key generator, polynomials, AI, graph technique

https://doi.org/10.30684/etj.33.3B.5
2412-0758/University of Technology-Iraq, Baghdad, Iraq

INTRODUCTION TO CRYPTOGRAPHY

Cryptography is the science and study of methods of protecting data in computer and communication systems from unauthorized disclosure and modification. The type of cryptography include two type, the Secret key cryptography involves the use of a single key. Given a message (called plaintext) and the key, encryption produces unintelligible data which is about the same length as the plaintext was. Decryption is the reverse of encryption, and uses the same key as encryption.

Public key cryptography is sometimes also referred to as asymmetric cryptography. Public key cryptography is a relatively new field, invented in 1975 unlike secret key cryptography, keys are not shared. Instead, each individual has two keys: a private key that need not be revealed to anyone, and a public key that is preferably known to the entire world.[2][5]

Stream ciphers

encrypt bits individually. This is achieved by adding a bit from a key stream to a plaintext bit. There are synchronous stream ciphers where the key stream depends only on the key, and asynchronous ones where the key stream also depends on the ciphertext.see figure (1)

Type of stream cipher:

A synchronous stream cipher is one in which the key stream is generated **independently** of the plaintext message and of the cipher text. The encryption process of a synchronous stream cipher can be described by the equations: [2]

$$\begin{aligned} \sigma_{i+1} &= f(\sigma_i, k), \\ z_i &= g(\sigma_i, k), \\ c_i &= h(z_i, m_i), \end{aligned}$$

Figure (1) synchronous stream cipher

Where is the **initial** state and may be determined from the key **k**, **f** is the next-state function, **g** is the function which produces the key stream zi, and **h** is the output function which combines the key stream and plaintext **mi** to produce cipher text **ci**. The encryption and decryption processes are depicted in following figure. The OFB mode of a block cipher [4]

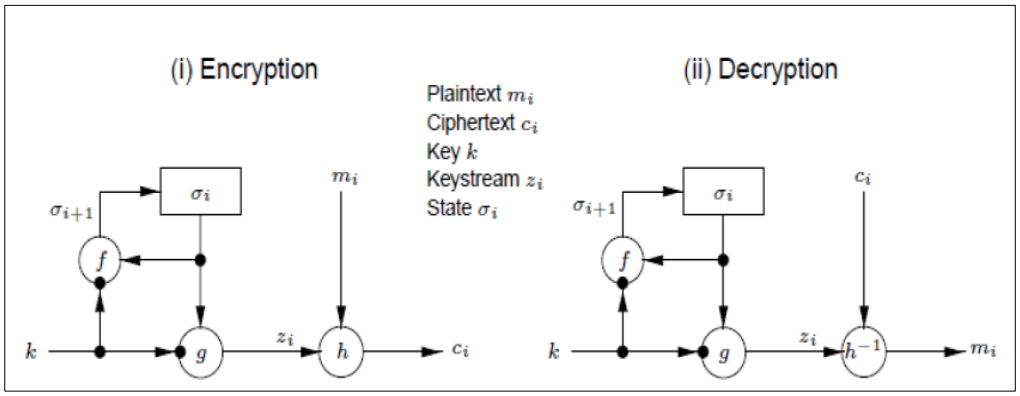


Figure (2) encryption and decryption in stream cipher

A *self-synchronizing* or asynchronous stream cipher is one in which the **key stream** is generated as a function of the key and a fixed number of previous ciphertext digits. The encryption function of a self-synchronizing stream cipher can be described by the equations:

$$\begin{aligned} \sigma_i &= (c_{i-t}, c_{i-t+1}, \cdots, c_{i-1}), \\ z_i &= g(\sigma_i, k), \\ c_i &= h(z_i, m_i), \end{aligned}$$

Figure (3) asynchronous stream cipher

where = (c−t; c−t+1; …. ; c−1) is the (non-secret) *initial state*, k is the *key*, g is the function which produces the *key stream* zi, and h is the *output function* which combines the key stream and plaintext mi to produce ciphertext ci. The encryption and decryption processes are depicted in following Figure. The most common presently-used self-synchronizing stream ciphers are based on block ciphers in 1-bit cipher feedback mode [4][5][6]

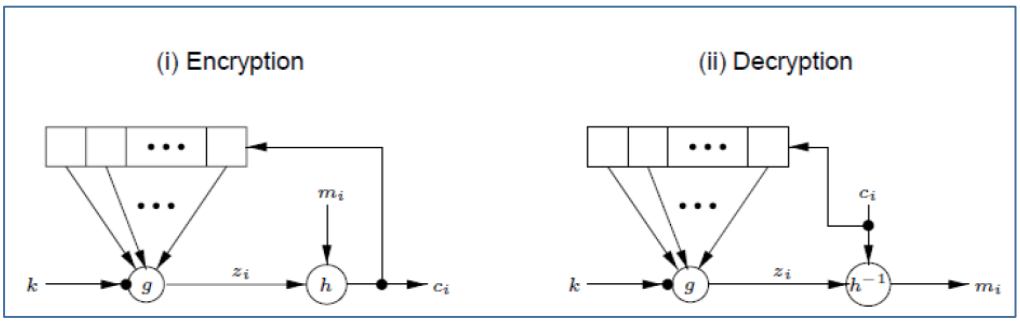


Figure (4) asynchronous encryption and decryption in stream cipher

key generator

The actual encryption and decryption of stream ciphers is extremely simple. The security of stream ciphers hinges entirely on a

"Suitable" key stream s0, s1, s2 ... Since randomness plays a major role, we will first learn about the two types of random number generators (RNG) that are important for us [9].

Trng (True Random Number Generator)

True random number generators (TRNGs) are characterized by the fact that their output cannot be reproduced. TRNGs are based on physical processes. Examples include coin flipping, rolling of dice, semiconductor noise, clock jitter in digital circuits and radioactive decay. In cryptography, TRNGs are often needed for generating session keys, which are then distributed between Alice and Bob, and for other purposes. [10]

(General) Pseudorandom Number Generators (PRNG)

Pseudorandom number generators (PRNGs) generate sequences which are computed from an initial seed value. Often they are computed recursively in the following way: see figure (5)

$$s_0 = \text{seed}$$
$$s_{i+1} = f(s_i), \quad i = 0, 1, \ldots$$

Figure (5) example of PRNG

LFSR & NLFSR (linear feedback shift register and nonlinear feedback shift register)

LFSRs are easily implemented in hardware and many, but certainly not all, stream ciphers make use of LFSRs. A prominent example is the A5/1 cipher, which is standardized for voice encryption in GSM.

As we will see, even though a plain LFSR produces a sequence with good statistical properties, it is cryptographically weak. However, combinations of LFSRs, an LFSR consists of clocked storage elements (flip-flops) and a feedback path. The number of storage elements gives us the degree of the LFSR. In other words, an LFSR with m flip-flops is said to be of degree m. The feedback network computes the input for the last flip-flop as XOR-sum of certain flip-flops in the shift register.
See figure (6).

A non-linear feedback shift register can be easily implemented in hardware or software and is used to create a pseudo-random sequence of numbers for many different applications. [9][10][12]

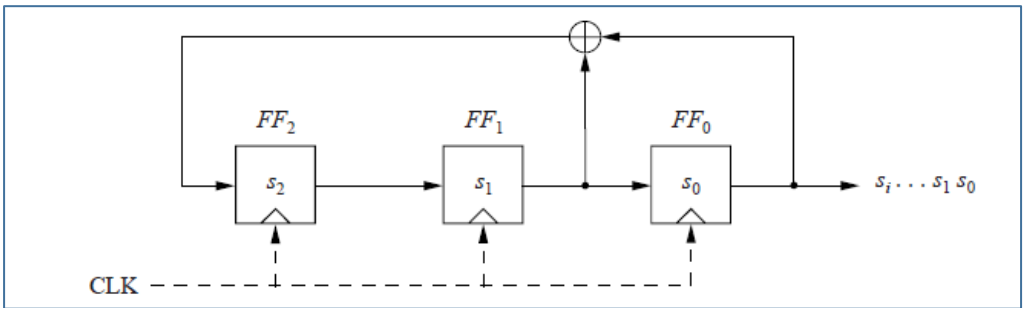


Figure (6) LFSRs

Let's assume the LFSR is initially loaded with the values s0, . . . , sm−1. The next output bit of the LFSR sm, which is also the input to the leftmost flip-flop, can be computed by the XOR-sum of the products of flip-flop outputs and corresponding feedback coefficient.

Nonlinear feedback shift registers (NLFSR) have received much attention in designing numerous cryptographic algorithms such as stream ciphers and lightweight block ciphers to provide security in communication systems. In most cases, NLFSRs which the key stream generator is a shift register with non-linear feedback function .as illustrated in following figure. Function. The simplest nonlinear function is **"AND"** functions, for example:

F=1+X1X2+X2X3+X2X3X4
Where X1X2 are (X1 and X2)

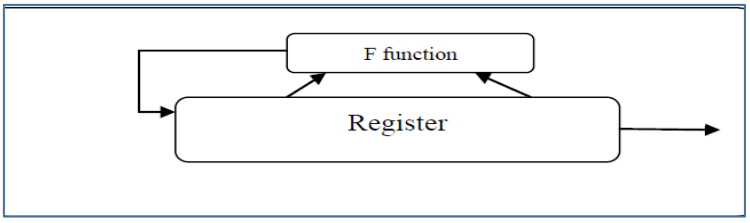


Figure (7) NLFSRs

Testing randomness number generator (five test)

-Frequency test:

The purpose of this test is to determine whether the number of 0's and 1's in s are approximately the same, as would be expected for a random sequence. Let n0, n1 denote the number of 0's and 1's in s, respectively. The statistic used is. Where the equation → **X1 = (n0 − n1)2 /n**

-Serial test (two-bit test)

The purpose of this test is to determine whether the number of occurrences of 00, 01, 10, and 11 as subsequences of s are approximately the same, as would be expected for a random sequence. Let n0, n1 denote the number of 0's and 1's in s, respectively, and let n00, n01, n10, n11 denote the number of occurrences of 00, 01, 10, 11 in s, respectively. Note that n00 + n01 + n10 + n11 = (n − 1) since the subsequences are allowed to overlap. The statistic used is **- Run test:**[1]

$$X_2 = \frac{4}{n-1}\left(n_{00}^2 + n_{01}^2 + n_{10}^2 + n_{11}^2\right) - \frac{2}{n}\left(n_0^2 + n_1^2\right) + 1$$

Poker test

Let m be a positive integer such that [n /m] ≥ 5.(2m), and let k = [n /m]. Divide the sequence **S** into k non-overlapping parts each of length m, and let ni be the number of occurrences of the ith type of sequence of length m, 1 ≤ i ≤ 2m. The poker test determines whether the sequences of length m each appear approximately the same number of times in s, as would be expected for a random sequence. The statistic used is[1]

$$X_3 = \frac{2^m}{k}\left(\sum_{i=1}^{2^m} n_i^2\right) - k$$

-Runs test

The purpose of the runs test is to determine whether the number of runs (of either zeros or ones) of various lengths in the sequence s is as expected for a random sequence. The expected number of gaps (or blocks) of length i in a random sequence of length n is ei = (n−i+3)/2i+2. Let k be equal to the largest integer i for which ei ≥ 5. Let Bi, Gi be the number of blocks and gaps, respectively, of length i in s for each i, 1 ≤ i ≤ k. The statistic used is:**[1]**

$$X_4 = \sum_{i=1}^{K}\frac{(b_i - e_i)^2}{e_i} + \sum_{i=1}^{K}\frac{(g_i - e_i)^2}{e_i}$$

-Autocorrelation test

The purpose of this test is to check for correlations between the sequence s and (non-cyclic) shifted versions of it. Let d be a fixed integer, 1≤ d ≤ [n/2]. The number of bits in s not equal to their d-shifts is[1]

$$A(d) = \sum_{i=0}^{n-d-1} s_i + s_{i+d},$$

Where + denotes the XOR operator. The statistic used is**:[1]**

$$X_5 = 2(A(d) - (n-d/2))/\sqrt{n-d}$$

Which approximately follows an N(0,1) distribution if n − d ≥ 10. Since small values of A(d) are as unexpected as large values of A(d), a two-sided test should be used.

Proposal system (first propose)

The first key generator is building used four stages of logic gate called (system gate 1) that generates in each stage different string of bit. Used the degree of polynomials with 8, 7, and 6.

Six polynomials used with right and left with three register on the left and the right at each side register referred to the polynomial degree. This explain with figure (8) the max period of proposed (1) depend on the max value of the register ((2^8-

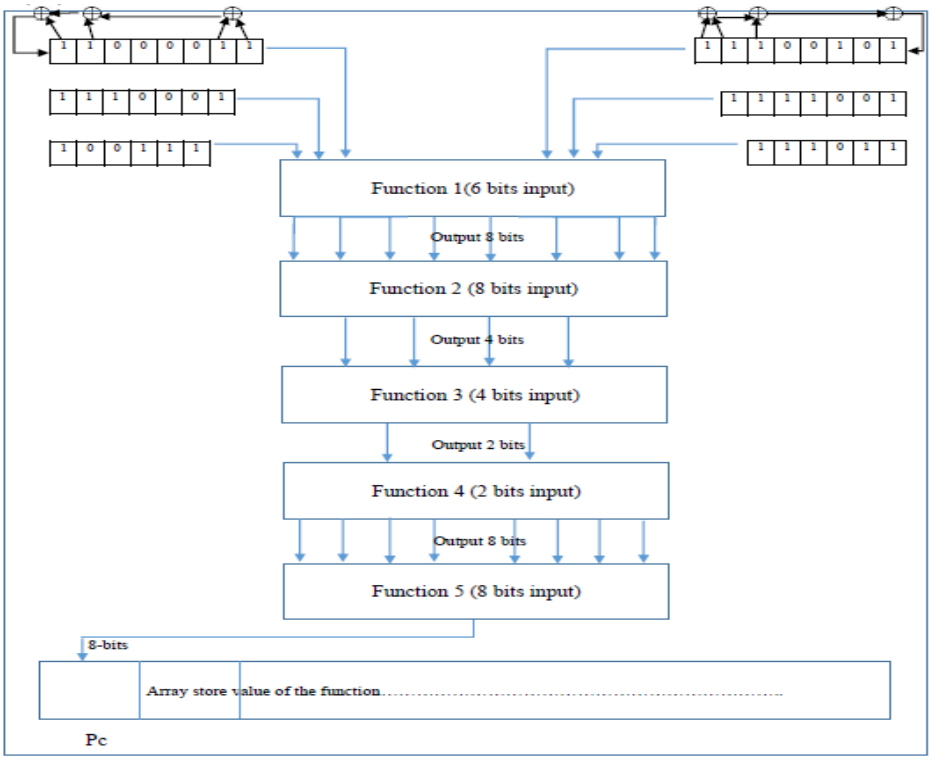


Figure (8): One Round of Architecture of First Proposal

The register side:

The system used six register with three on the left and three on right the system consist of multifunction that can be explained with following:

First register (left)

The first register is polynomials with degree 8 where the initial value is (1, 1, 0, 0, 0, 0, 1, 1) that register gives the polynomial ($x^8+x^7+x^2+x^1+1$) as (LFSR) is used to generate 255 value string of bit where (2^8=265-1)as max period table 1 show the value of the first register.

Table (1) result for the first register

Initial value	Random generator	Five test randomness
11000011	1100001110010100011011110000100010000101111101111110100100000110010001110100010110011010101011011011101011110110010111000001010…… …	• Pass frequency test 0.0039 • Pass serial test 5.9915 • Pass poker test -126.4 • Pass run test 6.221 • Pass autocorrelation test 0.002

Second register (left)

The second register is polynomials with degree 7 where ($x^7+x^3+x^2+x^1+1$) the initial value is (1, 1, 1, 0, 0, 0, 1) as (LFSR)that register is used to generate 255 value string of bit where (2^7=128-1) as max period. Table 2 show the value of the second register:

Table (2) result of the second register

Initial value	Random generator	Five test randomness
1110001	1000111101100111110101010010110110101100001101 1100101011110000000100010010000101110100110100 0001110111111001100100111000110001010001111011 001111101010..……	• Pass frequency test 0.0039 • Pass serial test -764.9 • Pass poker test -126.4 • Pass run test 5.221 • Pass autocorrelation test 1.032

Third register (left)

The third register is polynomials with degree 6 where ($x^6+x^5+x^4+x^1+1$) the result is (1, 1, 0, 0, 1, 1, 1) as (LFSR) that register is used to generate 255 value string of bit where (2^6=64-1)as max period. Table 3 show the value of the third register.

Table (3) result of the third register

Initial value	Random generator	Five test randomness
100111	1110011000100111110000110110101011001011110111 0100100000010100011100110001001111100001101101 0101100101111011101001000000101000111001100010 011111000011……..	• Pass frequency test 0.0039 • Pass serial test -764.9 • Pass poker test -126.4 • Pass run test 4.221 • Pass autocorrelation test 0.032

The right register is the same register of the left except that the Xor between shows the table 4 is different in position between the registers.

Table (4) the result of all register

Regisy	Reg init array	Result of the registry
1	1100001 1	1100001110010100011011110000100010000101111101111110100100000 1100100011101000101100110101010110110111010111 0.....
2	1110001	1000111101100111110101010010110110101100001101110010101110000 0001000100100001011101001101000001110111111……
3	100111	1110011000100111110000110110101011001011110111010010000001010 0111001100010011111000011011010101100101111……..
4	1110010 1	1010011101010010001101001101100111111110111010010111110100011 1000001101111110001001111001110011010101000 1……..
5	1111001	1001111001010000011100110000110101111100011001111001010000011 1001100001101011111000110011110010100000111 0……..
6	111011	1101110010010100110000111010000001000110110101100111100010111 1101110010010100110000111010000001000110 1……..

Function side

In system gate design the gate used different stages at each stage used different gates that change the value of the initial array in the reg1 to reg6. Six register go through different stages of gate to gets more permutation of the bits where the last stage give the result of the six register this can be explained in the following:

Function 1:

In this stage used different gate that is (XNOR, NOT GATE) where the gate connected in different way to produce 8 bit where accept six bits at each time from the six register. Where the reg1 and reg6 connect with xnor, gate reg2 and reg5 connect with xnor , reg3 and reg4 connect with xnor, the result of reg1 and reg6 then go through not gate,and then the result from the reg3, 4 and 5, 2 go through xnor and then all value then store in array.

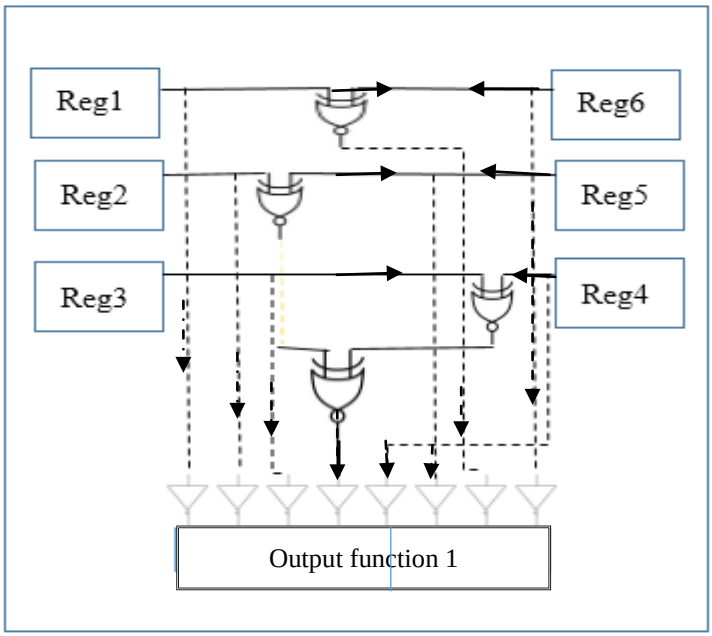


Figure (9): first stage connect logic gate

Where the result from the stage seen in the following table(5)

Table (5) result for the first stage

Function 1	The string of value
6 bit input	00000011101001101100111100001100101100010111110100000101110100010111101101101010011111100111111111111111110011110……

Function 2:

In this stage used the gates that are (NOR, AND GATE) that connected in different way to produce 4 bit where accept 8 bits from the stage one at each time. The connect between the gates seen in the figure(10)

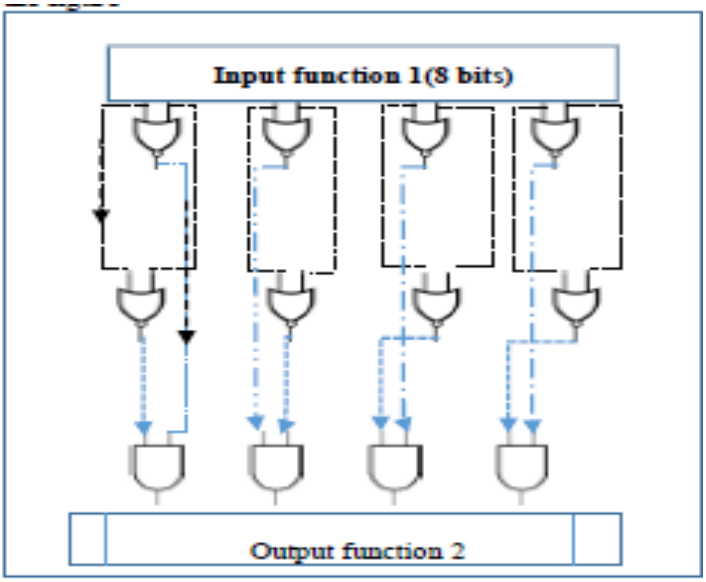


Figure (10): second stage connect logic gate

The result of the stage two is seen in the table(6)

Table (6) result for the second stage

Function 2	**The string of value**
8 bit input	00000000000000000000000011001100000000000000000000000000000000000000000000110000001100110000000000000000000000000011……

Function 3:

In this stage used different gate that is (XNOR, AND GATE) where the gates are connected in different way to produce 2 bit where accept 4 bits from the satge2 at each time. The connect between the gate is seen in the figure (11)

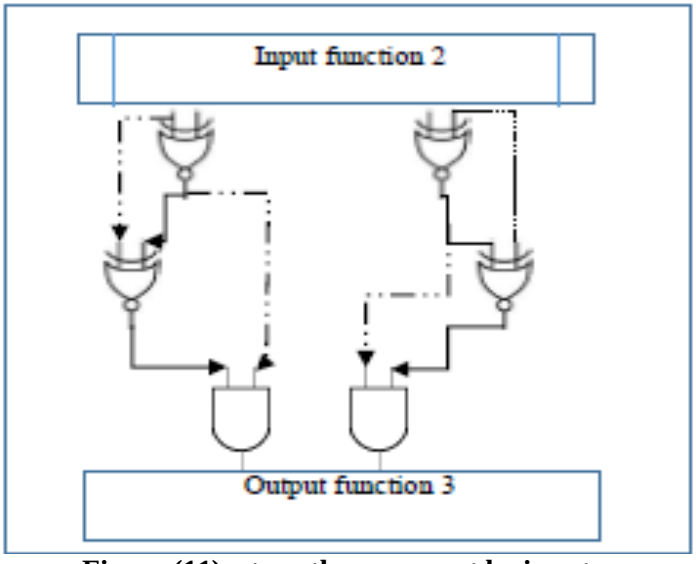


Figure (11): stage three connect logic gate

The result of the stage three seen in the table (7)

Table (7) result for third stage

Function 3	The string of value
4 bit input	0101010101010101010101010101010101010101010101010101010101010101110 10101110101011101010101010101010101010101..……

Function 4:

In this stage used different gate that is (XNOR, AND, NOT GATE) where the gate are connected in different way to produce 8 bit where accept 2 bits from the stage 3 at each time. The connect between the gate is seen in the figure

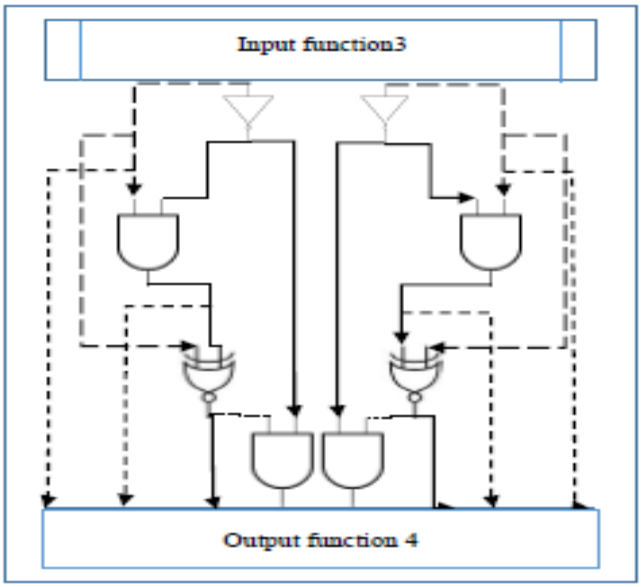


Figure (12): stage 4 connect logic ga

Where the result of the stage four can be seen in the table(8)

Table (8) result for fourth stage

Function 4	The string of value
2 bit input	1111001011110010111100101111001011110010111100101111001011110 0101111001011110010111100101111001011110…..……

AI search technique:

The last step or function in first proposal used the other permutation of bits by using the AI search on the output bits from the function four. in this stage cut 8 bits as the first 8 bits from function 4 that bits go through the AI search (binary search) generate new arrange of bits this explain in the figure below:

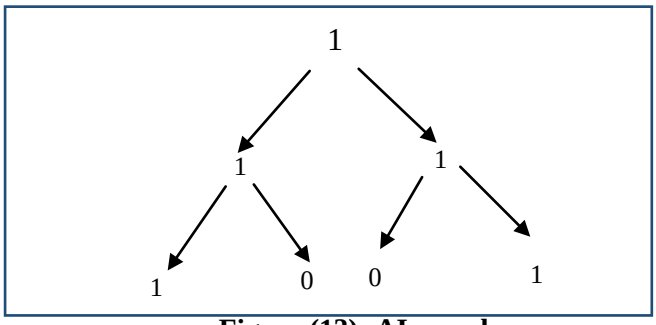


Figure (13): AI search

by reading form right to left in AI search technique where the result for first 8 bits= 11110101 and the last 1 where is number 8 bite in first 8 bits is sequancly add to string of first 8 bits where the finale result = 11101011

This function repeated for all bits in the function four and each time work with 8 bits. When compare between the result of function4 and five can see the different function 4 (value = 11110010) and the result of the function 5 (value = 11101011).

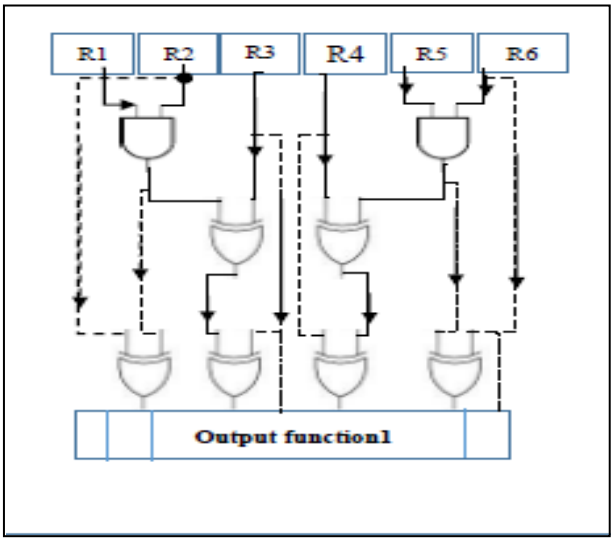
System proposal (second propose)

this system use also four stages to produce different values that is used to generate key using different gate (logic gate that connected in different way in different stage).this system is similar to the first system where the different from the first system in the design of the logic gate of each stage .this can be explain in the following

Function 1:

In this stage used gate like (AND GATE , XOR GATE) to produced 6 bits each time from 6 polynomial registry , the input of the first function is 6 bit from the six register where the output of the first function go to next stage . figure(15) explain the connected logic gate of the first function.

Where the result seen in the table



Figure(14) Connect logic gate

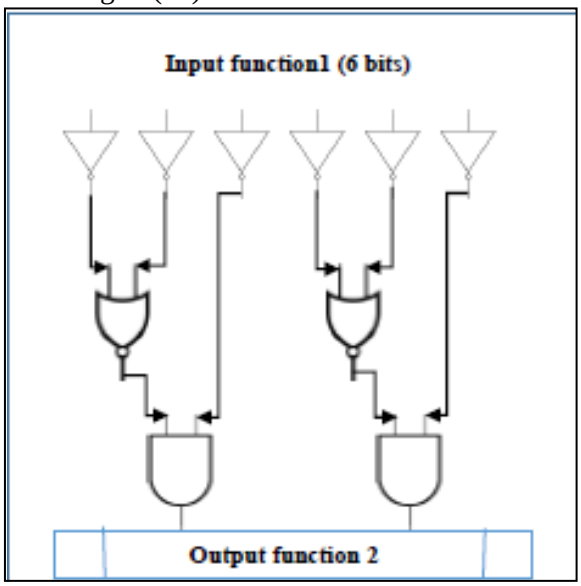
Where the result see in the table(9)

Table (9) result for the stage one

Function 1	**The string of value**
6 bit input	11111110110110110010000100000100111111111011010010010100011000110010 01111001001111010011000011000011011110011…..……

Function 2:

In this stage used gate like (AND GATE , NOR GATE ,NOT GATE) to produced 2 bits each time where accept 6 input from the stage one. the connected of the logic gate seen in the figure(15)



Figure(15) Connect logic gate

Where the result see in the table(10)

Table (10) result for second stage

Function 2	**The string of value**
6 bit input	00001001000010110110100111001010000011101111000001100011100010101000 10111001001000001100100001100101000101111..…..……

Function 3

In this stage used gate like (AND GATE ONLY) to produced 4 bits each time where accept 2 input from the stage two. the connected of the logic gate can be seen in the figure(16)

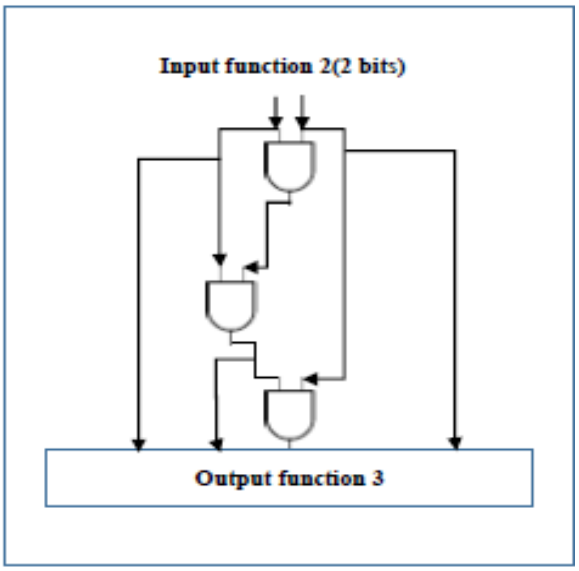


Figure (16) Stage 3 connect logic gate

Where the result can be seen in table (11)

Table (11) result for third stage

Function 3	**The string of value**
2 bit input	0000100100001011011010011100101000001110111100000110001110001010100 0101110010010000011001000011001010001011……..……

Function 4:

In this stage used gate like (XNOR GATE ,NOT GATE) to produced 8 bits each time where accept 4 input from the stage three. the connected of the logic gate see in the figure(17)

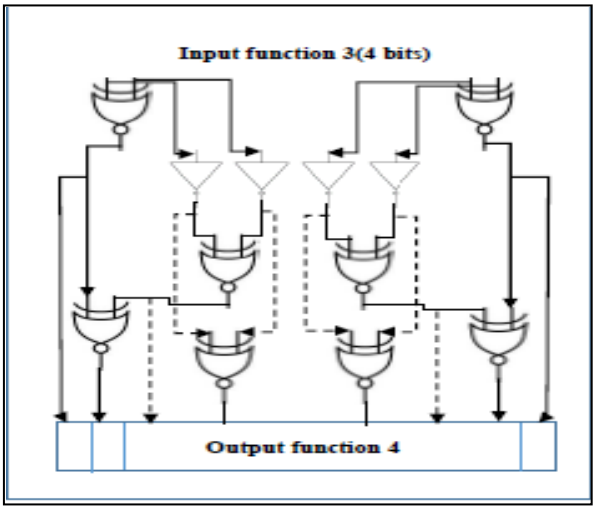


Figure (17): stage 4 connect logic gate

Where the result can be seen in table(12)

Table (12) result for fourth stage

Function 4	**The string of value**
4 bit input	11111111111111111110101000000101111111111111111111010111111111000000010111101011111010100000010111111111.....

Graph technique

At the last stage in proposal 2 use another function to add more permutation on the bits by using the graph technique on the bits to gets more permutation on the bits this can be explain in the following figure(18)

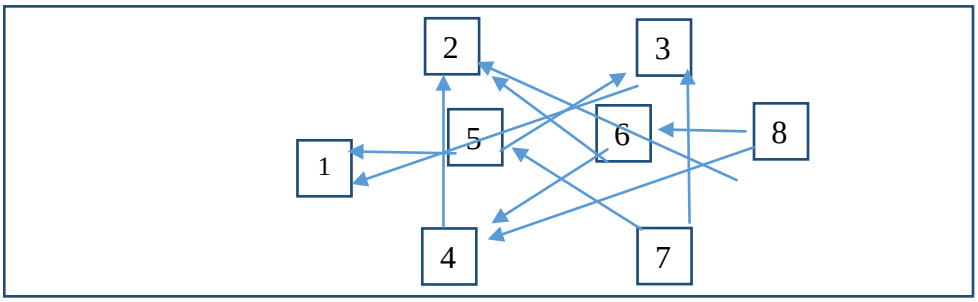


Figure (18): the graph technique first 8 bits

The equation of the permutation the bits depend on the rule that every number (subtraction) between the two number where if the result = 2 or number the result mod 2 = 0 this result rearrange depending on the number that not the result = 2 or mod 2 where the result form the first 8 bits = (1,4) , (2,3) , (6,5) ,(8,1) ,(6,3) . (7, 4), (8, 5)…

Then the first bits in permutation as the result= 1, 4,2,3,8,6,7,5.

Conclusion

1- The LFSR generator have max period of bits after that the value is repeated. So by using new technique to initial new value from the circuit and search algorithm (binary search) to get new initial value.

2- When used the logical gate, AI, and graph technique this get new value of the bits so the max period of register is validate for the randomness test level .

3- Depend on the class testing to generate the bits can easy break but when used another principle for graph theory or search A.I that help to get more coefficient and diffusion .

References

[1] S. W. Golomb. "Shift Register Sequences". Aegean Park Press Laguna Hills, CA, USA 1981.

[2] T. W. Körner, "Codes and Cryptography lecture notes". Available online: http://www.dpmms.cam.ac.uk/~twk/

[3] R. Lidl and H. Niederreiter. "Introduction to Finite Fields and Their Applications Revised Edition". Cambridge University Press, Cambridge; 1994 pp235-239.
[4] J. L. Massey, "Shift-Register Synthesis and BCH Decoding".IEEE Trans. on Information Theory, vol. 15(1), pp. 122-127, Jan 1969.
[5] A. Menezes, P. van Oorschot, and S. Vanstone." Handbook of Applied Cryptography". CRC Press, Inc.; 1997 pp1-4. Available online at http://www.cacr.math.uwaterloo.ca/hac/ 17
[6] S. Matyas and C. Meyer. Cryptography: "A New Dimension in Computer Data Security-A Guide for the Design and Implementation of Secure Systems". John Wiley & Sons, Inc. 1982. pp. 53.
[7] R. Paddock, _"A Guide to Online Information About: Noise/Chaos/Random Numbers and Linear Feedback Shift Registers"._ Circuit Cellar Online: "The Magazine for Computer Applications". http://www.designer iii.com/cco/noise/c89r4.htm, last modi_ed April 17, 2005.
[8] A. Sherman, _"On the Enigma cryptograph and formal dentitions of cryptographic strength", _ Master's thesis (advisor R. Rivest), MIT, 1981.
[9] William Stein, David Joyner, SAGE: "System for Algebra and Geometry Experimentation, Comm. Computer Algebra 39(2005)61-64". SAGE is available for download at http://sage.scipy.org (the article can be downloaded from http://sage.scipy.org/sage/misc/sage_sigsam_updated.pdf)
[10] H. van Tilburg. "Coding Theory at Work in Cryptology and Vice Versa"._ Handbook of Coding Theory, vol. 2. Ed. by V.S. Pless and W.C. Hu_man.
[11] N. G. deBruijn." A combinatorial problem". Indag. Math., 8(1946), pp. 461-467.
[12] E. Dubrova. "A list of maximum period NLFSRs". Cryptology ePrint Archive, 2012/166. www.iacr.org
[13]G. M. "Kyureghyan. Minimal polynomials of the modified de Bruijn sequences". Discrete Applied Math., 156(2008), pp. 1549-1553.
[14]R. Lidl, H. Niederreiter." Introduction to Finite Fields and their Applications (Revisited Edition)". Cambridge University Press, Cambridge, 1994.