Fuzzy Based Pseudo Random Number Generator used for Wireless Networks

Sattar B. Sadkhan*, Sawsan K. Thamer** and Najwan A. Hassan**
*Department of Computer Science, College of Science, Babel University, Babel-Iraq.
**Department of Computer Science, College of Science, Al-Nahrian University, Baghdad-Iraq.

Abstract

In this paper an adaptive fuzzy system is proposed, it generates the frequency hopping sequence for a spread spectrum communication system. The system learns rules from data and acts as Pseudo Random Number Generator (PRNG) .Thirty sample patterns are the input to Fuzzy PRNG, while the bandwidth is partitioned into number of frequency bins. Each bin used triangular membership function to analyze the input to fuzzy sets, encoding these as fuzzy rules. The input vector matches if-part of a fuzzy rule and fires that rule's output fuzzy set. The fuzzy system tested with 100 and 1025 frequencies and compared it with three other types of PRNG, the fuzzy system had lower values and thus gives a more uniform spread than did the other methods. The fuzzy system was easier to change and harder to intercept.

Keywords: pseudo random number generator, fuzzifaction, inference rules, defuzzifaction.

Introduction

Frequency hopping system spreads a transmitted signal's energy across a bandwidth larger than the minimum required for the signal. It spreads the signal when it changes or hops the transmission frequency many times per second. The transmitter and receiver must stay synchronized. They must both hop to the same frequency at the same time. If an eavesdropper does not know the frequency sequence, the hopped signal looks like low-intensity noise spread over the entire bandwidth. Frequency hopping systems use a pseudo random number generator (PRNG) to produce a random sequence of frequencies [1]. In Fig.(1) shows the signal flow, whatever the code used; both the transmitter and receiver must have a copy of the code or the procedure to generate it.

The proposed system describes a fuzzy rule based PRNG. Fuzzy rules map the distribution of old frequencies to new output frequencies. These rules, with some initial conditions give the output distribution. And the sampling pattern, that gives the previous output distribution, fix the output sequence.

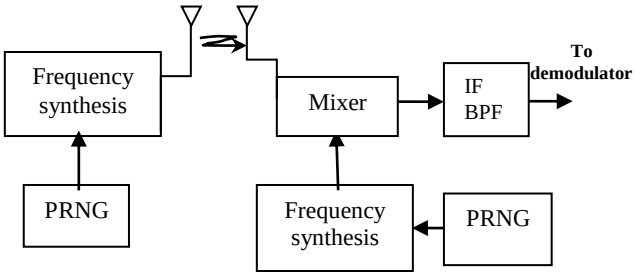

Fig. (1) spread spectrum communications system that frequency hopping.

Background

Several researches in the field of pseudo random number generator were developed and published in the literature Peter J. Pacini & Bart Kosko [2] proposed a system generates the frequency hopping sequence for a spread spectrum communication system. The system learns rules from data and acts as a PRNG.

Andrew Klapper [3], described the Feed Back Shift Registers with Carry operation (FBSRs), implemented, and analyzed with respect to memory requirements, initial loading, period, and distributional properties of their output sequences.

J. Hong [4] proposed a method for stabilizing the input power of a random number generator using fuzzy logic control in crypto module hardware.

Pseudo Random Number Generator (PRNG)

A sequence of numbers is said to be random if the value of each successive number cannot be predicted based on the preceding numbers. Such numbers could be generated using some process normally considered to be random, such as rolling a pair of dice. In practice, it will primarily be interested in arithmetic methods for generating random numbers that can be implemented on a computer. These sequences are referred to as "pseudorandom" [5]. A successful method for generating PRNs yields a number sequence that would appear indistinguishable from a truly random sequence in the sense that no regularities of any kind would appear in the sequence beyond what the laws of chance would predict. In most methods for obtaining a sequence of PRNs, each number in the sequence is used to find the next one using some arithmetic procedure [6].

The most popular classes of PRNG used to compare with the proposed system are:

- ***Linear Feed Back Shift Register (LFBSR)***

The general form of n-stage LFBSR is shown in Fig. (2), the feedback logic in this case can be written as:

$$f(x_1, x_2, x_3, ..., x_n) = a_0 + a_1 x_1 + a_2 x_2 + ... + a_n x_n \quad .....(1)$$

Where, **a**'s specifies feedback tapings and can only take values of 1 or 0. The values of the **a**'s determine the transition matrix which defines the autonomous behavior of the Feed Back Shift Register (FBSR) [7].

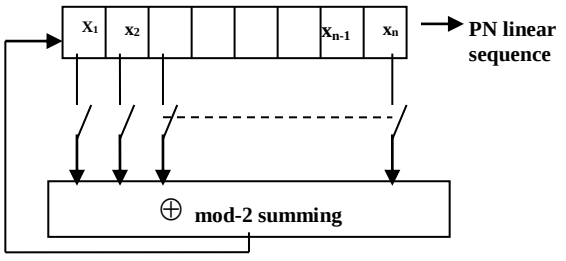

Fig. (2) Linear Feed Back Shift Register.

- ***Non-Linear Feed Back Shift Register (NLFBSR)***

The general form of the combinational feedback function is given by:

$$f(x_1, x_2, x_3, ..., x_n) = a_0 + a_1 x_1 + a_2 x_2 + ... + a_n x_n + a_{n+1} x_1 x_2 + a_{n+2} x_1 x_3 + ... + a_{2n-1} x_1 x_n + a_{2n} x_1 x_2 ... x_n \quad .....(2)$$

Where, a's =0 or 1. Hence there are 2^{2^n} possible feedback functions for n-stage FBSR. Only 2^n of these functions are linear. Also there are $2^{2^n} - 2^{2^n - n - 1}$ FBSRs that have cyclic behavior. It should be noted that a complete theory of NLFBSRs are not very well understood, and it is difficult to generate long periods with good randomness properties. Fig. (3) shows a general model of an **n**-bit NLFBSR [7].

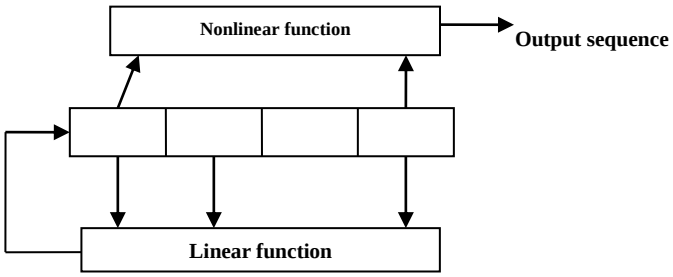

Fig. (3) A Non Linear Feed Back Shift Registers.

- ***Blum Blum Shub Generator***

A popular approach to generate secure pseudo random number is known as the Blum, Blum, and Shub (BBS) generator, named for its developers. It has perhaps the strongest public proof of its cryptographic strength. To generate any set of random numbers, the following steps should be followed [8]:

First, choose two large prime numbers, p and q, that both have a remainder of 3 when divided by 4. That is

$$p \equiv q \equiv 3 \pmod 4 \quad .....(3)$$

Simply means that (p mod 4)=(q mod 4)=3

Next, choose a random number s, such that s is relatively prime to n; this is equivalent to saying that neither p nor q is a factor of s. then the BBS generator produces a sequence of bits B_i according to the following algorithm:

$X_0 = s_2 \bmod n$

For i=1 to α

$X_i = (X_{i-1})^2 \bmod n$

Next i

$B_i = X_i \bmod 2$

Thus, the least significant bit is taken at each iteration.

The Proposed Fuzzy based Pseudo Random Number Generator (FPRNG)

The proposed system describes a fuzzy rule based PRNG. A fuzzy rules map distribution of old frequencies to new output frequencies. These rules, with some initial conditions give the output distribution, and the sampling pattern that gives the output distribution, fix output sequence. Each sequence has an unknown length if it repeated at all. Such sequences are hard for an eavesdropper to predict if he does not know the rules, the initial conditions, or the sampling pattern. A complete communication system includes more subsystem than PRNG. Source coding, encryption, channel coding, modulation schemes, and synchronization all affect how the system performance.

A-FPRNG Characteristics

A FPRNG system has the following characteristics:

- The fuzzy rules map distributions of old output frequencies to new output frequencies.
- The fuzzy rules, with initial conditions and the random consequent, fix the output sequences.
- Each sequence has an unknown length.

B-Input specifications

Frequency synthesizer assumed to have **N** distinct frequencies. The system partitions the bandwidth (B.W) into **n** frequency bins such that each contains **N/n** frequencies, as shown in Fig. (4).

The value of **N** can be changed to any number that does not affect the algorithm or the rules, but **N** should be a multiple of **n,** to give a balanced partition into bins, but **n** must be chosen to be small to limit the number of rules. For that, **n** is chosen to be 5. So the B.W consist of **N** frequencies is portioned into **n** frequency bins each contains **N/n** frequencies.

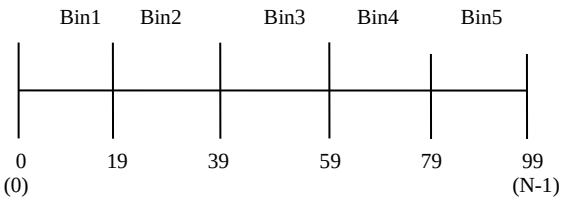

Fig. (4) Frequencies portioned into (5) frequency bins.

Fuzzy system gave a sequence of integers in {0,……,N-1} such that each integer tended to occur equally. In this work **N** is chosen to be **100** frequencies as an example the system work on it, but it can take any number of frequencies without changing the algorithm, and the number of bins was taken 5 bins for simulation.

C-FPRNG Structure

Fig. (5) shows the structure of FPRNG. FPRNG consists of three layers. Each layer performs a specific function. The first layer represents the fuzzifier. Second layer represents inference engine (rule-base). Last layer (third layer), represents the defuzzifier, the output from this layer represents the output frequency.

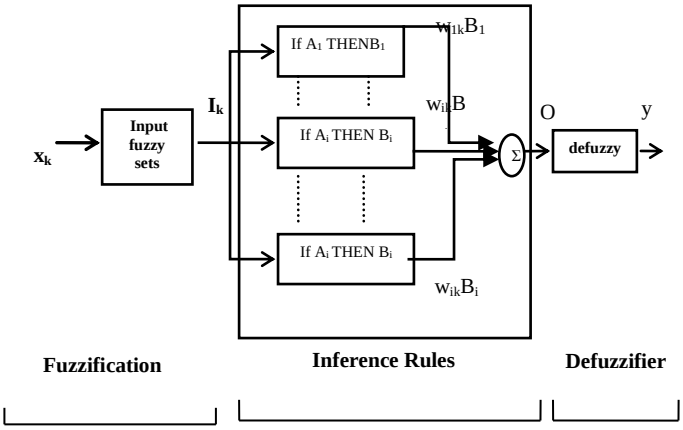

Fig. (5) The structure of proposed system.

1- Fuzzification

A fixed sampling pattern decided which (**30**) is the input to fuzzy system.

f_i=function (f_1,f_2,f_3, . . . ,f_{30}) .....(4)

Where $\mathbf{f_i}$ is fixed sampling pattern of (**30**) output frequencies.

Example:

At time k, it is possible to sample the output frequencies from times (k-1,k-6,k-8,k-14, . . . ,etc).

Since choosing randomly 30 samples, makes the samples appear independent from one step to the next. The number of samples that fall in the frequency bins are five inputs to the fuzzy system, they are stored as a length-5 vector $\mathbf{x_k}$.

The decision to use 30 samples depends on the following two principles in the system design:

1. The first (5) frequency bins are chosen to give about 200 fuzzy rules because number of bins is 5 and three fuzzy sets then $\mathbf{3^5}$ will gave about 243 maximum.
2. Design the input fuzzy sets for each bin as shown in Fig. (6). The medium fuzzy set should have its maximum value at the expected number of hits per bin. Since peaks at **6**, and there are five bins, the number of samples should be **6*5=30**.

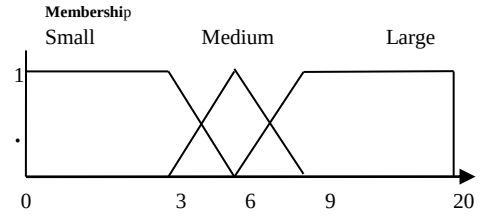

Fig. (6) Input fuzzy sets for each bin.

In designing sampling pattern, it is required to minimize the sequences autocorrelation and minimize the maximum sample time delay.

The 3 fuzzy sets which are used in each bin are: Small, Medium, and Large. Fig. (5) shows these fuzzy sets:

$m_S : \{0,\ldots,29\} \longrightarrow [0,1]$.....(5)

$m_M : \{0,\ldots,29\} \longrightarrow [0,1]$.....(6)

$m_L : \{0,\ldots,29\} \longrightarrow [0,1]$.....(7)

Where

$m_S(u)$ is the degree to which the number (u) is Small.

$m_M(u)$ is the degree to which the number (u) is Medium.

$m_L(u)$ is the degree to which the number (u) is Large.

So, the fuzzy sets, in Fig. (6) map the number of $\mathbf{x_k[j]}$ in each bin into a length-3 fit vector.

($m_S(x_k\,[j])$, $m_M(x_k\,[j])$, $m_L(x_k\,[j])$), of fuzzy unit values.

To find the relative value in membership function for each input to the fuzzy unit, a triangular membership function used which is one of the simplest and more efficient methods used. Fig. (7) shows the triangular membership function which was used:

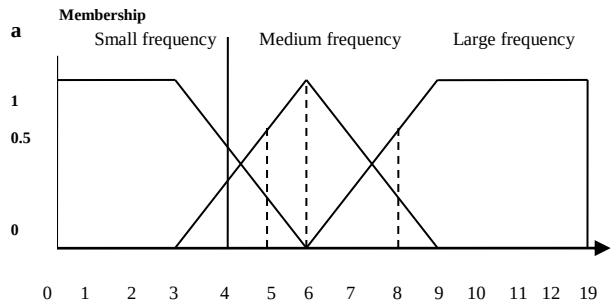

Fig. (7) Triangular Membership function.

And the formula that used in medium frequency is the triangular membership function is [9]:

$$\mu(x) = \begin{cases} \dfrac{a(b-x)}{b-c} & for\ \ b \geq x \leq c, \\ \dfrac{a(d-x)}{d-c} & for\ \ c \geq x \leq d, \\ 0 & otherwise \end{cases}$$

.....(8)

And the formula used in both small and large frequency is trapezoidal membership function is [9]:

$$\mu(x) = \begin{cases} \dfrac{(a-x)e}{a-b} & where\ \ a \leq x \leq b \\ e & when\ \ b \prec x \leq c \\ \dfrac{(d-x)e}{d-c} & when\ \ c \prec x \leq d \\ 0 & otherwise \end{cases}$$

.....(9)

Examples:

If x_k= (4,8,9,5,12)

As discussed before, each number in $\mathbf{x_k}$ is a bin value and for each one it has a fuzzy unit value ($m_S(x_k\,[j])$, $m_M(x_k\,[j])$, $m_L(x_k\,[j])$) then:

- Input value of 4, the expected value per bin, maps to (2/3,1/3,0) where 2/3 membership of 4 in small area, 1/3 membership of 4 in medium area, 0 membership of 4 in large area.
- Input value of 8, the expected value per bin, maps to (0,1/3,2/3).
- Input value of 9, the expected value per bin, maps to (0,0,1).
- Input value of 5, the expected value per bin, maps to (1/3,2/3,0).
- Input value of 12, the expected value per bin, maps to (0,0,1).

The fit vectors of "five integer inputs" form a **(5*3)** matrix $\mathbf{I_k}$

$I_k[j,1]=m_S(x_k[j])$ .....(10)
$I_k[j,2]=m_M(x_k[j])$ .....(11)
$I_k[j,3]=m_L(x_k[j])$ .....(12)

For **n** bins and **p** input fuzzy sets, $\mathbf{I_k}$ would have dimensions **n*p**, where $\mathbf{I_k}$ is the input to the fuzzy rules and **j=1,2,…,5**.

2- Inference Rules

Inference rules stage includes fuzzy rule, fuzzy AND operation, adaptive rule generation and combination output fuzzy set.

Fuzzy rules

The fuzzy rules or a fuzzy if-then rule that used in the system is of the form [10]:

IF S_k is A_i Then output is B_i.

Where $\mathbf{S_k}$ is the input to the fuzzy system $\mathbf{x_k[j]}$ and each $\mathbf{A_i}$ is a length **5** vector of fuzzy set values: Small, Medium, and Large.

For example when A_i is (S,L,L,M,L) can be read as:

IF (x_k[1] is Small AND x_k[2] is Large AND x_k[3] is Large AND x_k[4] is Medium AND x_k[5] is Large)

Which is called the antecedent or the if-part for the rule.

Take $\mathbf{A_i}$ as matrix of **5*3** similar to $\mathbf{I_k}$ each of five fuzzy set values in $\mathbf{A_i}$ maps to a unit bit vector

$Small \rightarrow (1,0,0)$

$Medium \rightarrow (0,1,0)$

$Large \rightarrow (0,0,1)$

Then for A_i=(S,L,L,M,L) the matrix which represent it is:

$$A_i = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

At each step, the fuzzy sets that used for input map the five integers input values $\mathbf{x_k[j]}$ to their membership, the degrees to which they belong to the Small, Medium and Large fuzzy sets. Each conjunct in the antecedent or if-part of the rule takes on one of these values.

- ***Fuzzy AND operation***

A fuzzy AND operation takes the minimum of values then the complete [11] "x_k is A_i" take on the value

$$\mu_{A_i}(x_k) = \min_j(\mu_{A_i[j]}(x_k[j])) \quad .....(13)$$

Where **j=1...5**, $\mathbf{x_k}$ input to the fuzzy sets as part of each rule. So each rule looks at how much a numerical input $\mathbf{x_k}$ belongs to fuzzy set $\mathbf{A_i}$ in the if-part of the rule.

$$w_{ik} = \mu_{A_i}(x_k) \quad .....(14)$$

Where $\mathbf{w_{ik}}$ scale the then-part $\mathbf{B_i}$ of each rule.

- ***Adaptive Rule Generation***

The (then-part) or consequent of each rule is a length-5 fit vector. The system used **10000** samples, these samples are random number from the random generator of the computer, again these **10000** samples will give **333** output frequencies every one of them have **30** samples, which is scaled and truncated to give integers between **1** and **5**. These integers are the bin numbers. Like the work of antecedent, the fuzzy system's sampling pattern gives the $\mathbf{x_k}$ of the previous samples. Then if-part $\mathbf{A_i}$ is found closest to $\mathbf{x_k}$, the one with maximum $\mathbf{w_{ik}}$, and the next sample with $\mathbf{A_i}$.

Combination Output Fuzzy Set

The combined output fuzzy set **O** equals to the sum of antecedent and consequent, which can find by the sum of the rule then-parts $\mathbf{B_i}$ multiply with their firing level $\mathbf{w_i}$ .Summing outputs avoids crosstalk and achieves modularity. And this is called fuzzy inference. The fuzzy inference is responsible for drawing conclusions from the knowledge base. Fig. (8) shows the combined output fuzzy set **O** [11].

$$O = \sum_i w_i B_i = [w_1 \ w_2 \ldots\ldots w_i].\begin{bmatrix} B_1 \\ B_2 \\ . \\ . \\ . \\ B_i \end{bmatrix} \quad .....(15)$$

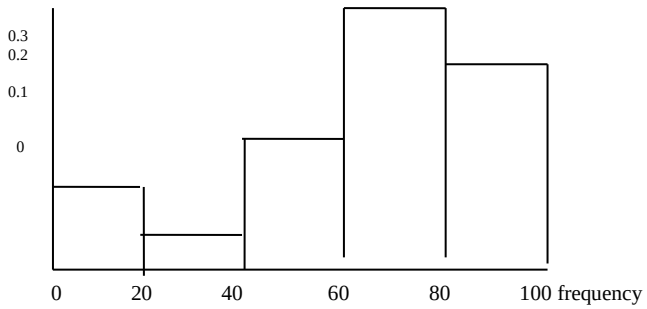

Fig. (8) Combined output fuzzy set O= (0.15,0.1,0.2,0.3,0.25).

3- Defuzzfication

The final step in the proposed system is to defuzzifier fuzzy value to crisp value by using centroid method or Center Of Gravity (COG). This method is best used if there is more than one variable. The formula that used is [12]:

$$y = \frac{\sum_{j=1}^{n} O(j) x_k}{\sum_{j=1}^{n} O(j)} \quad \text{.....(16)}$$

Where **O** is a combined output fuzzy set, $\mathbf{x_k}$ are inputs to the fuzzy system, **n** is a number of frequency bins which it is **5** and **y** is a crisp values which it is the output frequency.

Experimental Results

In this section, a discussion is made for the results of fuzzy system. These results are taken for four runs, each of 100 frequencies. These results are plotted in Fig. (9). Three of them used random initial conditions. The fourth run, shown by dotted line, used all zero initial conditions. This extreme case shows the fuzzy system robustness to initial conditions.

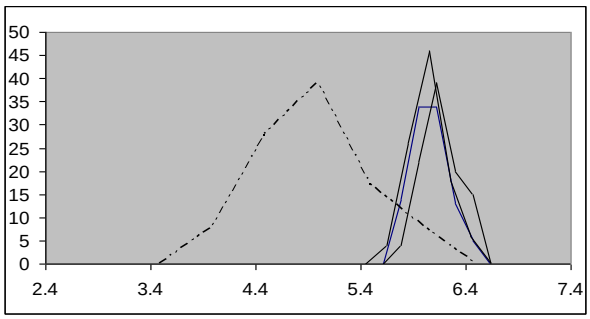

Fig. (9) Chi-squared values for the fuzzy system for 100 frequencies. The dotted line run used all zero initial conditions, the other three used random initial conditions.

A comparison is made between fuzzy system and other conventional systems (Linear FeedBack Shift Register (LFBSR), Non Linear FeedBack Shift Register (NLFBSR) and Blum Blum Shub (BBS)). Each one of them has its maximum length. The linear feedback connection has 33-stage shift register. In the NLFBSR, one of the blocks at least take 33-stage shift register, these two generate a binary pseudo noise sequence of length 2^{33}-1. In BBS system, 2^{33}-1 bits are choosed. The output is sampled every ten clock cycles to get integers in the range {0,…, 1023}. These integers mapped to the transmission frequencies for a frequency hopping system with 1024 frequencies. In the fuzzy system 1025 frequencies has been used.

The chi-squared test is used to compare between the four systems. A lower chi-squared value indicates more uniform sequences. Fig. (10) shows the result of the four systems. The solid line marks the mean chi-squared values for fuzzy system with 1025 frequencies. The dashed line marks the mean chi-squared values for BBS system with 1024 frequencies. The dot-dashed line marks the mean chi-squared values for LFBSR system with 1024 frequencies. The dotted line marks the mean chi-squared values for NLFBSR system with 1024 frequencies. The fuzzy system tended to give a more uniform sequence than the other systems did.

The fuzzy system and other systems differ in structure and in function. A LFBSR, NLFBSR and BBS with m stages maps **m** information bits to a codeword with $\mathbf{2^m-1}$. During the $\mathbf{2^m-1}$ clock cycles required to produce the output sequence, LFBSR and NLFBSR systems take on all bits in the state except the all zero state. So the initial conditions just choose which cyclic shift codeword the system will generate.

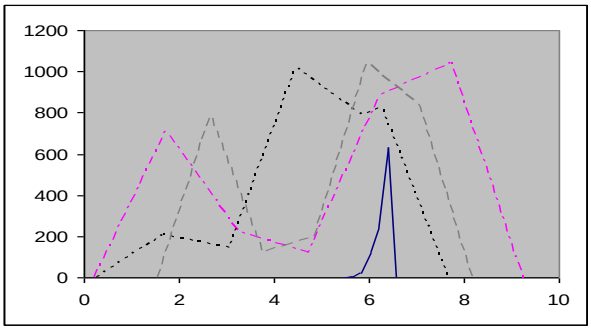

Fig. (10) Mean chi-squared values for the fuzzy system for 1025 frequencies and the other systems with 1024 frequencies.

The conventional systems structure gives fast operation but it leaves a frequency hopping system open to interception. An eavesdropper who can detect **2m+1** bits in the output sequence can build a code generator to produce the same sequence.

The fuzzy system is nonlinear. At each step the uniform distributions conditioned on prior sampled outputs. Unless an eavesdropper knows the fuzzy rules driving the system, he cannot predict the output sequence.

Conclusions

A fuzzy system that used in proposed system produces a sequence that is harder to intercept, and easier to spread over any number of frequencies without changing the algorithm or the fuzzy rules. And the fuzzy spreader is deterministic and does not need randomness to securely spread and de spreader a signal. The eavesdroppers that do not know these parameters cannot enter to the system.

References

[1] Dixon R.C., "spread Spectrum Techniques", New York: Widely Insterscience, 1997.

[2] Peter J. Pacini and Bart Kosko, "daptive Frequency Hopper", Dept. of electrical engineering system, university of southern California, Los Anglos, 1995.

[3] Andrew Klapper, "Feedback Shift Registers, 2-Adic Span, and Combiners with Memory", Anderson Hall, Department of Computer Science, University of Kentucky, 1997.

[4] Hong J., "Mathematical Methods in Sample Surveys", Series on Multivariate Analysis, Vol. 3, 2005.

[5] Robert Ehrlich, "Generation of Random Numbers by Computer", Project PHYSNET, Physics Dept., George Mason University, 2002.

[6] Richard P. Brent, "Note on Marsaglia's Xorshift Random Number Generators", Oxford University, Journal of Statistical Software, Volume 11, Issue 5, August 2004.

[7] Choi K., Cheun K., and Jung T., "The importance of PN Sequences in the Design of Spread Spectrum Systems", IEEE Trans. Commun., 2001.

[8] William Stallings, "Cryptography and Network Security: Principle and Practice", Prentice Hall, second edition, 1999.

[9] Ahmed M. Abraham, "Fuzzy Logic for embedded System Application", Ph.D. senior member, IEEE, 2004.

[10] Kosko B., "Fuzzy systems as universal approximates", IEEE Trans. Computer, vol. 42, no.11, pp1329-1333, 1994.

[11] Kosko B., "Neural Networks and fuzzy system", Englewood Cliffs, prentice- Hall, 1999.

[12] Vojislav Kecman, "Learning and Soft Computing Support Vector Machines, Neural Networks, and Fuzzy Logic Models", A Bradford Book, the MIT Press, 2001.

الخلاصة

النظام الضبابي يدرس ويطبق مولد الاعداد العشوائيه المزيفه المستند الى النظام الضبابي Fuzzy based Pseudo Random Number Generator الذي يولد سلاسل القفز المتردده لنظام اتصالات الطيف المنتشر. حيث استعملت ثلاثون عينه ادخلت الى نظام FPRNG والموجه قسمت الى عدد من صناديق التردد في كل صندوق استعمل وظيفه عضويه مثلثيه triangular membership function لتحليل المدخلات الى المجموعات الضبابيه، شفرت هذه كقواعد ضبابيه. طريقه مركز الجاذبيه هي الطريقه المستخدمه لاعاده التضبيب من القيم الضبابيه الى القيمه الحقيقيه لانتاج العدد في التردد القفاز. كذلك قورنت النتائج المحصله من FPRNG مع الطرق الثلاثه التقليديه الاخرى من مولد الاعداد العشوائيه المزيفه باستخدام Chi Squared Test. بينت المقارنه ان النظام الضبابي لديه قيم أوطا وهكذا يعطي انتشاراً اكثرا توحيداً من عمل الطرق الاخرى. وكذلك بينت ان النظام الضبابي كات اسهل عند التغيير واصلب للاعتراض.