



# **Voice Encryption Using Twin Stream Cipher Algorithm**

**تشفير الصوت باستخدام خوارزمية التوأم الانسيابية**

**Prepared by  
Omar Mejbel Hammad Aljouani  
((401320142))**

**Supervisor  
Dr. Hebah H. O. Nasereddin  
Dr. Abdulkareem O. Ibadi**

**Master Thesis**

**Submitted in Partial Fulfillment of the Requirements of the  
Master Degree in Computer Science**

**Department of Computer Science**

**Faculty of Information Technology**

**Middle East University**

**Amman - Jordan**

**January - 2016**

((بسم الله الرحمن الرحيم))

يَرْفَعِ اللهُ الَّذِينَ آمَنُوا مِنْكُمْ وَالَّذِينَ  
{أَوْتُوا الْعِلْمَ دَرَجَاتٍ

((صدق الله العظيم))

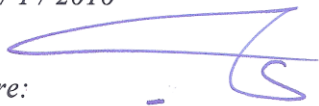
## Authorization Statement

*I, Omar Mejbel Hammad, authorize the Middle East University to provide hard copies or electronic copies of my thesis to libraries, institutions or individuals upon their request.*

*Name: Omar Mejbel Hammad*

*Date: 3 / 1 / 2016*

*Signature:*

A handwritten signature in blue ink, consisting of a large, sweeping horizontal stroke followed by a smaller, more intricate flourish.

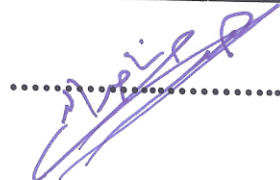
## Examination Committee Decision

This is to certify that the thesis entitled “Voice Encryption Using Twin Stream Cipher Algorithm” was successfully defended and approved on January, 2016

### Examination Committee Members

### Signature

(Supervisor Committee Member)



**Dr. Hebah H. O. Nasereddin**

Associate professor, Faculty of information technology

Middle East University (MEU)

(Head Internal Committee Member)

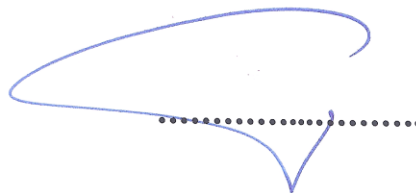


**Dr. Oleg Vladimirovich Viktarov**

Associate professor, Faculty of information technology

Middle East University (MEU)

(External Committee Member)



**Dr. Mohammed Ahmad Alia**

Associate professor, Faculty of Sciences and IT

Al Zaytoonah University (ZUJ)

## **Acknowledgment**

I utilize this opportunity to thank everyone helped me reach this stage and everyone who encourage me during performing this thesis. I want to thank Dr. Hebah H. O. Nasereddin for her guidance and supervision during writing this thesis. Extended thanks are also for my family and friends who encourage me during writing this thesis. I also want to thank everyone who believes that the knowledge is right for everyone.

The greatest thank ever to assistant prof. Abdulkareem O. Ibad, the head of software engineering department at Baghdad College for economic sciences.

## Dedication

اهدي خلاصة جهدي العلمي المتواضع الى :

قرة عيني الرسول محمد عليه افضل الصلاة واتم التسليم .....

..... وطني العراق الجريح .....

..... اخي الشهيد الحاضر الغائب صهيب .....

..... والدي ووالدتي واختي رفاق دربي ومسيرتي .....

..... كل من كان له بصمة بجهدي العلمي هذا .....

..... كل الشهداء الذين استشهدوا برصاص الغدر والخيانة .....

..... كل من كان يدعي لي ويوجهني ويتمنى لي الخير .....

..... جامعة بغداد أخص بها كلية التربية ابن الهيثم .....

..... الاعدادية المركزية للبنين .....

## Table of Contents

<b>AUTHORIZATION STATEMENT.....</b>	<b>II</b>
<b>EXAMINATION COMMITTEE DECISION .....</b>	<b>III</b>
<b>Acknowledgements.....</b>	<b>IV</b>
<b>dedication... ..</b>	<b>V</b>
<b>Table of contents.....</b>	<b>VI</b>
<b>List of Figures.....</b>	<b>VIII</b>
<b>List of Tables.....</b>	<b>IX</b>
<b>List of Abbreviations.....</b>	<b>X</b>
<b>Abstract .....</b>	<b>XI</b>
<b>المخلص .....</b>	<b>XII</b>
<b>Chapter One Introduction.....</b>	<b>1</b>
<b>1 Chapter One Introduction.....</b>	<b>2</b>
1.1 Introduction.....	2
1.1.1 Stream Cipher.....	6
1.1.2 Advantages of Stream Cipher .....	7
1.1.3 Shift Registers .....	8
1.1.4 Linear-Feedback Shift Register.....	9
1.1.5 Feedback with Carry Shift Registers.....	10
1.1.6 Non-Linear Feedback Shift Registers .....	11
1.1.7 Non-linear combiner of LFSRs .....	11
1.1.8 Random Properties of Sequences .....	12
1.2 Research Problem .....	13
1.3 Research Objectives.....	14
1.4 Motivation.....	15
1.5 Significance of Work .....	15
1.6 Thesis outline .....	15
<b>Chapter Two Backgrounds and Literature Review .....</b>	<b>17</b>
<b>2 Chapter Two Background and Literature Review .....</b>	<b>18</b>
2.1 Overview.....	18
2.2 Background .....	18
2.2.1 Stream Cipher Usage.....	19
2.2.2 Stream Cipher Security .....	20

2.2.3	Types of Stream Cipher .....	21
2.2.3.1	Synchronous Stream Ciphers.....	21
2.2.3.2	Self-Synchronizing Stream Ciphers.....	22
2.2.4	Comparison of Stream Cipher.....	22
2.3	Literature review .....	23
<b>Chapter Three</b>	<b>Research Methodology .....</b>	<b>32</b>
<b>3</b>	<b>Chapter Three Research Methodology .....</b>	<b>33</b>
3.1	Introduction.....	33
3.2	The Twin Concept .....	35
3.3	Stream Cipher Algorithm.....	35
3.4	The Proposed Twin Stream Cipher Algorithm .....	36
3.5	The Combining Part.....	36
3.6	The Driving Part .....	38
3.7	The Proposed Twin Stream Cipher Algorithm .....	40
<b>Chapter Four</b>	<b>.....</b>	<b>42</b>
<b>Results and Discussion</b>	<b>.....</b>	<b>42</b>
<b>4</b>	<b>Chapter Four Results and Discussion .....</b>	<b>43</b>
4.1	Overview.....	43
4.2	The Based Algorithm Testing.....	44
4.3	The Twin Algorithm Testing .....	47
<b>Chapter Five</b>	<b>Conclusions and Future Work .....</b>	<b>52</b>
<b>5</b>	<b>Chapter Five Conclusions and Future Work .....</b>	<b>53</b>
5.1	Conclusion .....	53
5.2	Future work.....	55
<b>References</b>	<b>.....</b>	<b>56</b>
<b>Appendix</b>	<b>.....</b>	<b>61</b>



## List of Figures

<b>Figures</b>	<b>Page</b>
Figure 1.1 Cryptology Classification (Ibadi, 2010)	3
Figure 1.2 Stream Cipher (Abdulsalam, 2011)	7
Figure 1.3 Shift Register with Feedback (Hell, Johansson, Maximov and Meier, 2006)	9
Figure 1.4 Linear Feedback Shift Register (Hell, Johansson and Meier, 2006)	9
Figure 1.5 Feedback with Carry Shift Register (Arnault, Berger and Pousse, 2011)	10
Figure 3.1 Methodology Block Diagram	34
Figure 3.2 Twin Algorithm Main Components	35
Figure 3.3 Stream Cipher Algorithm Parts	35
Figure 3.4 The Non-linear Part	37
Figure 3.5 The Driving Part of The Stream Cipher Algorithm	38
Figure 3.6 The Proposed Stream Cipher Algorithm	39
Figure 3.7 The Proposed Twin Stream Cipher Algorithm	40
Figure 3.8 Interface of the Proposed Algorithm	41
Figure 3.9 The Proposed Program Result	41
Figure 4.1 Interface of Twin Stream Cipher Algorithm	43
Figure 4.2 Twin Stream Cipher Algorithm Program Giving Results	43
Figure 4.3 The Statistical Test Program	51
Figure 4.4 The Statistical Test Program Results	51

## List of Tables

<b>Table</b>	<b>Page</b>
Table 2.1: Comparison of Stream Cipher	23
Table 3.1: The Truth Table of The Proposed Function	37
Table 4.1: The Based Algorithm Testing (6003 & 500 bits)	44
Table 4.2: The Based Algorithm Testing (6003 & 500 bits)	45
Table 4.3: New High Speed Stream Cipher Algorithms	46
Table 4.4 The Twin Algorithm Testing (5003 & 800 Bits)	47
Table 4.5 The Twin Algorithm Testing (6000 & 400 Bits)	48
Table 4.6 The Twin Algorithm Testing (24 & 22 Bits)	49
Table 4.7: The Twin Algorithm Testing (6000 & 400 bits)	50

## List of Abbreviations

<b>SR</b>	Shift Register
<b>FSR</b>	Feedback Shift Register
<b>LFSR</b>	Linear Feedback Shift Register
<b>FCSR</b>	Feedback with Carry Shift Register
<b>NLFSR</b>	Non-linear Feedback Shift Registers
<b>FPGA</b>	Field Programmable Gate Arrays
<b>ANF</b>	Algebraic Normal Form
<b>PR</b>	The Bernoulli Trials

## **Abstract**

### **Voice Encryption Using Twin Stream Cipher Algorithm**

Prepared By:

Omar Mejbek Hammad ((401320142))

Supervised By:

Dr. Hebah H. O. Nasereddin.

Dr. Abdelkareem. O. Ibadi.

There are many techniques and methods that are currently used to provide information security. One of these techniques hides information which sends through various media; so that the observer did not feel of the existence of secret information. In this thesis the encryption discussed as a voice encryption by using the twin stream cipher algorithm.

There were two methods for measuring the randomness which need to satisfy the binary strings used as key-stream, the first method examines the hypothesis that the string based on Bernoulli trials where the second examines the strength of a key-stream generator measured the complexity of the strings produced.

This thesis deals with the twin algorithm where it consists of two identical well-tested stream cipher algorithm, then to gather it as a combining section which is a non-linear function. The twining algorithm is responsible to determine the period of the algorithm, generate the key bit sequence, and determine the level of complexity. The proposed stream cipher algorithm based on statistical randomness test, and then a number of output streams generated and tested of various lengths to show that the algorithm has random behavior characteristics. All tests have been achieved is passed as it will show in the following chapters.

**Key Words:** cryptography, voice encryption, twin stream, Bernoulli trials.

## المخلص

### تشفير الصوت باستخدام خوارزمية التوأم الانسيابية

إعداد :

عمر مجبل حماد

إشراف :

د. هبة ناصر الدين

د. عبد الكريم عكلة عبادي

هناك العديد من التقنيات والأساليب التي يتم استخدامها حاليا لتوفير أمن المعلومات. واحدة من هذه التقنيات هي إخفاء المعلومات التي ترسل من خلال وسائل الإعلام المختلفة حتى أن المراقب لا يمكن له ان يلاحظ وجود معلومات سرية. هذه الأطروحة تناقش التشفير على شكل التشفير الصوتي باستخدام خوارزمية تيار الشفرات المزدوجة. هناك طريقتين لقياس العشوائية، الطريقة الأولى مبنية على تجارب برنولي والطريقة الثانية هي ان قوة مولد مفتاح التيار تقيس مدى تعقيد السلاسل المنتجة.

تتناول هذه الأطروحة خوارزمية التوأم حيث انها تتكون من اثنتين من الخوارزميات المتطابقة من تيار الشفرات التي تم اختبارها جيدا، ثم تجميعها لتشكيل معادلة غير خطية. الخوارزمية التوأمة هي المسؤولة عن تحديد فترة الخوارزمية، وتوليد التسلسل في الخوارزمية، وتحديد مستوى التعقيد. خوارزمية تيار الشفرات المقترحة على أساس اختبار العشوائية الإحصائية، ثم عدد من تيارات الإخراج التي تم إنشاؤها واختبارها على أطوال مختلفة لإظهار أن الخوارزمية لديها خصائص سلوكيه عشوائية. وقد اظهرت النتائج نجاح هذه الاختبارات وهذا ما سوف نظهره الفصول اللاحقة.

**كلمات البحث:** الترميز، التشفير الصوتي، تيار التوأم، تجارب برنولي.

# **Chapter One**

## **Introduction**

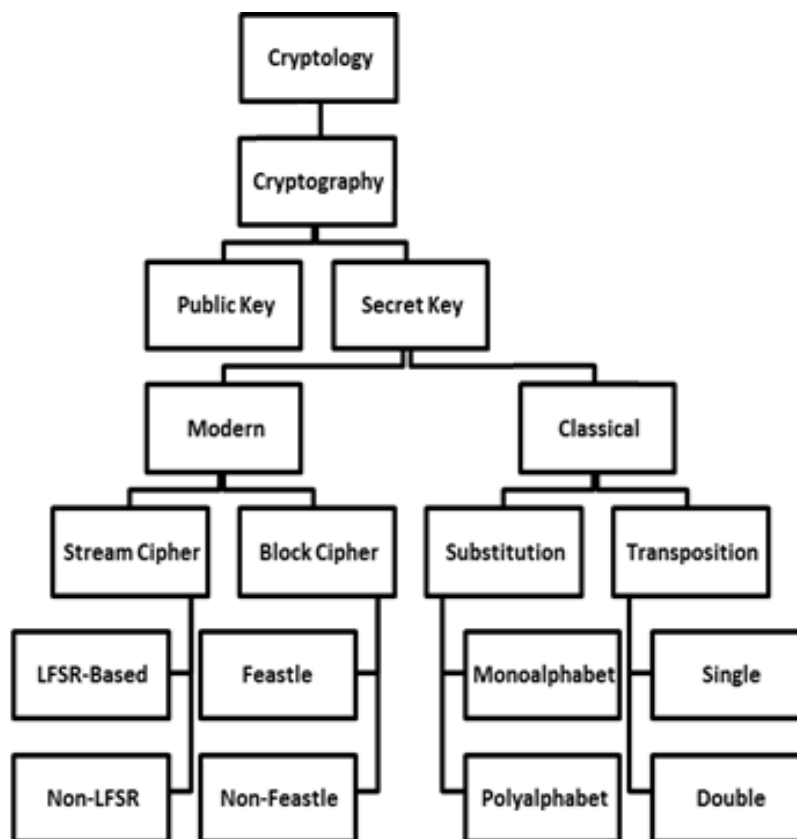
# 1 Chapter One Introduction

## 1.1 Introduction

During the past years, information security has become the most interest issue for many researches who built solutions, techniques and new ideas to ensure the process of transfer information safely through the network without any penetration and detection of the information.

As a result, there are many techniques and methods that are currently used in information security. One of these techniques is to hide information sent through various media; so that the observer does not feel of the existence of secret information within the sent information through the public communication channels. In this thesis the encryption will be discussed as a voice encryption using the twin stream cipher algorithm. The stream cipher is not new algorithm, but it's rarely used so there are a small number of resources that used the twin stream cipher algorithm (Ibadi, 2010).

Cryptography is a method of storing and transmitting data in a particular form so that only those for whom it is intended can read and process it. Cryptography is closely related to the disciplines of cryptology and cryptanalysis. Cryptography includes techniques such as microdots, merging words with images, and other ways to hide information in storage or transit. However, in today's computer-centric world, cryptography is most often associated with scrambling plaintext (ordinary text, sometimes referred to as clear-text) into ciphertext (a process called encryption), then back again (known as decryption). Individuals who practice this field are known as cryptographers (Caesar, G., and John, F., 2005).



**Figure 1.1 Cryptology Classification (Ibadi, 2010)**

The largest problem of the communication and computer security is shielding the secret information from interceptions. Cryptography is the art and science of achieving security by encoding message to make them non-readable. The procedure of transformation the plaintext to ciphertext is called encryption or the ciphering, while the procedure of transformation the ciphertext to plaintext is called decryption or the decipherment. The strength of the cryptosystem or the cipher system can be established on the secrecy of the ciphering algorithm or the secrecy of its parameters.

The security of a cryptosystem usually relies on the secrecy of the keys rather than the supposed secrecy of the algorithm. A strong cryptosystem has a large range of possible keys so that it is not possible to just try all possible keys. A strong cryptosystem will produce ciphertext which appears random to all standard statistical tests and can resist all known methods for breaking codes (Lantronix, Inc, 2010).



Network security is typically handled by a network administrator or system administrator who implements the security policy, network software and hardware needed to protect the network and the resources accessed through the network from unauthorized access and also ensure that employees have adequate access to the network and resources to work. A network security system typically relies on layers of protection and consists of multiple components including networking monitoring and security software in addition to hardware and appliances. All components work together to increase the overall security of the computer network.

The security of the stream cipher don't depend only on the algorithm's secure, even though modern technology allows the encapsulation of algorithm implementations as a coordinated circuits that are resistant to reverse engineering attacks that is include secret parameters called keys. In most applications, except the military or secret government communications, the algorithms are a public knowledge and the security of the cipher is based exclusively on the secrecy of keys (Ghosh, 2011).

Encryption is simply the translation of data into a secret code, and it is considered the most effective way to ensure data security. To read an encrypted file, you must have access to a secret key or password that enables you to decrypt it. Modern encryption is achieved using algorithms with a "key" to encrypt text or other data into digital nonsense and then decrypting it by restoring it to its original form. Encryption is a formula used to turn data into a secret code. Each algorithm uses a string of bits known as a "key" to perform the calculations. The larger a key is (the more bits in the key), the greater number of potential combinations that can be created, thus making it harder to break the code and unscramble the contents (Lantronix, Inc, 2010).

In general, there are two types of the key-based algorithms; symmetric and asymmetric algorithms. In symmetric algorithms, the encryption and the decryption keys are the same. These algorithms called the secret key algorithms the sender and the receiver agree on the key before they communicate securely. Encryption and decryption with symmetric algorithm can be denoted by (Singhal and Rania, 2011).

$E_K(P)$  = ciphertext resulting from encryption of the plaintext  $P$  using key  $K$ .

$D_K(C)$  = plaintext resulting from decryption of the ciphertext  $C$  using key  $K$ .

Symmetric algorithms can be separated into two classes, the first class called the stream algorithms or stream ciphers, where the second class called the block algorithms or block ciphers when to operate on the plaintext in groups of bits, the groups of bits are called blocks. The asymmetric algorithm can be designed when the encryption keys are different of the decryption keys, these algorithms called the public key algorithm (Singhal and Raina, 2011). The key administration considered is significant part of cryptography, as well as the main point of cryptography is to hold the plaintext or the key secret from any unauthorized user eavesdroppers. While the cryptanalysis defined as the science of recovering the plaintext or the key and an attempted cryptanalysis is called the attack.

There are four general types of cryptanalytic attacks (Moldovyan, 2008).

- 1. Ciphertext Only Attack:** The attacker knows only the ciphertext and attacks of the knowledge of statistical properties of the language for the plaintext; such as the frequency of a certain symbols or groups of symbols (letters) or by predicting the most possible string of symbols for the plaintext (word).

2. **Known Plaintext Attack:** The attacker knows both the plaintext and the corresponding ciphertext.
3. **Chosen plaintext Attack:** The attacker knows the ciphertext for the plaintext of choice.
4. **Adaptive-Chosen Plaintext Attack:** This is a special situation of a chosen for the plaintext attack. Not only the attacker can choose the plaintext that encrypted, but they can also modify the choice depends on the results of previous encryption.

Different algorithms suggest different degrees of security depend on how hard they are able to know. An algorithm is unconditionally secure if no matter how much ciphertext of cryptanalyst has, there is not enough information to recover the plaintext. In point of fact, only one-time pad is unbreakable given infinite resources. A brute-force attack happened when the unauthorized user trying all possible keys one by one and checking whether the resulting plaintext is meaningful or not, this attack for breakable cryptosystems in a ciphertext-only attack ( Hirota, and Kato, 2005).

### 1.1.1 Stream Cipher

A stream cipher which is defined as the process of encryption that is applied in a binary plaintext and encrypted it one bit at a time interval (t) of a pseudo-random sequence  $K(t)$ , it is combined by utilizing modulo two addition with plaintext bit  $P(t)$ , at time interval (t) to create the ciphertext bit also at time interval (t) which is denoted by  $C(t)$ . The sequence  $K(t)$  is called the key-stream for the stream cipher which is seen in Figure 1.2. The encryption process can be expressed as below (Abdulsalam, 2011).

$$1. C(t) = P(t) \oplus K(t) \quad \dots\dots\dots \text{Equation 1.1, (Singhal and Raina, 2011)}$$

Where  $\oplus$  denotes modulo two addition. The decryption process can be expressed as below:

$$2. \quad P(t) = C(t) \oplus K(t) \quad \dots\dots\dots \text{Equation 1.2, (Singhal and Raina, 2011)}$$

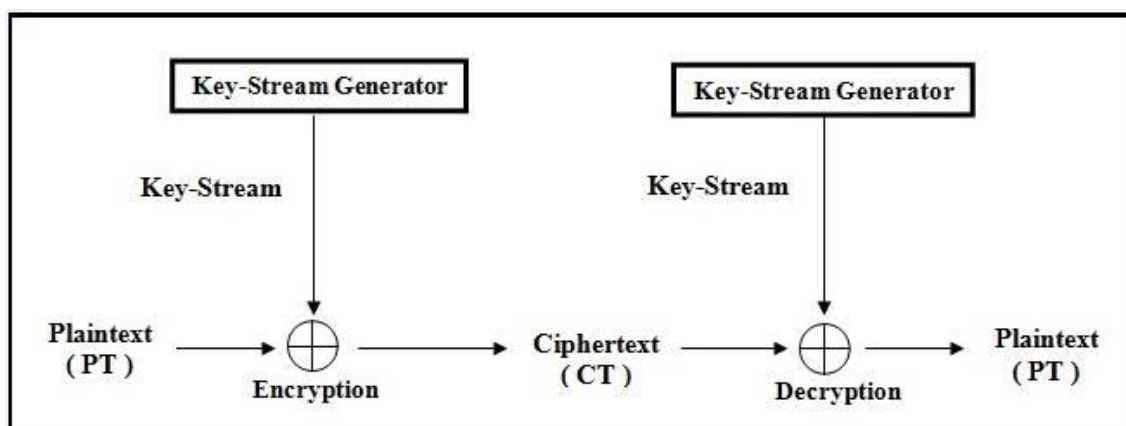
Where :

**(t)** Is the time interval

**K (t)** Is the pseudo-random sequence

**P (t)** is the plaintext bit

**C (t)** is the ciphertext bit



**Figure 1.2 Stream Cipher (Abdulsalam, 2011)**

The equations 1 and 2 indicated that the encrypt and the decrypt needed to produce the same key-stream sequence  $K(t)$ . The key ( $k$ ) for the stream cipher is defined as the initial step to start the generator. Both encrypt and decrypt needed to procedure this key. At the same time, the security of the stream ciphers depends completely on the key-stream generator (Abdulsalam, 2011).

### **1.1.2 Advantages of Stream Cipher**

Stream cipher systems have many advantages, which is made them the mainly extensively used systems, these are:

1. It can be generated a long bit streams from a small number of the initial parameters, so it is extremely able for practical applications.
2. It can be simply operated at speeds above 20 M bit/s, therefore it is enabling real-time enciphering of speech.
3. It depends on a number of shift registers that is cheap relatively.
4. It is relatively insensitive to errors introduced during the transposition. If a bit of the ciphertext is modified during the transposition then only this bit will be deciphered wrongly when the text reaches to the receiver.

### 1.1.3 Shift Registers

The most important component in the stream ciphers is the shift register (SR). The universal structure of the SRs, as shown in Figure 1.3, is that every SR contains  $m$  of stages; every stage can carry one bit. A clock input on each pulse of a clock is controlled the SR; the bits are shifted one stage to the right. The bits which are generated at the stage number 0 form the output of the SR, as the bits are shifted to the right; it is essentially to provide new group of bits to stage  $m-1$  for the SR of length  $m$ . These bits can obtain from the feedback loop contain the module which is calculated the value of the new bit SR with a feedback called the feedback shift register (FSR), (Hell, Johansson, Maximov and Meier, 2006).

It is too obvious that if there are  $m$  numbers of stages then there will be  $2^m$  probable states, since there are  $2^m$  probable states there will be  $2^{2^m}$  possibilities for the function of the feedback. There are three types of FSR, which are LFSR, FCSR, and Non-linear FSR.

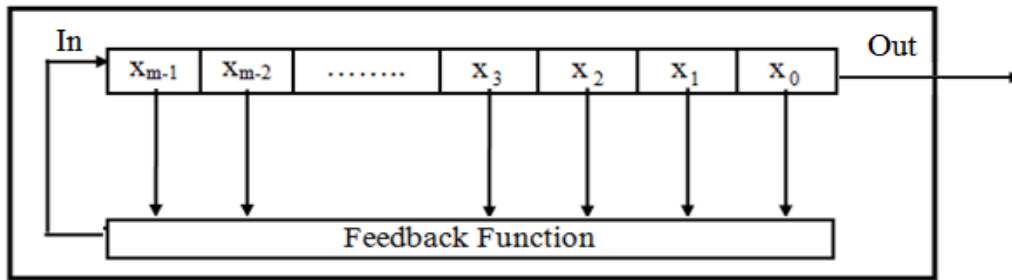


Figure 1.3 Shift Register with Feedback (Hell, Johansson, Maximov and Meier, 2006)

### 1.1.4 Linear-Feedback Shift Register

The linear feedback shift register (LFSR) is distinct as the feedback shift register that's the feedback function  $f(x_0, x_1, \dots, x_{m-1})$  is a linear function, the feedback function can be expressed as (Hell, Johansson and Meier, 2006):

$$F(x_0, x_1, \dots, x_{m-1}) = c_0x_0 + c_1x_1 + \dots + c_{m-1}x_{m-1} = \sum_{i=0}^{m-1} c_i x_i$$

..... Equation 1.3, (Hell, Johansson and Meier, 2006).

The coefficients  $c_0, \dots, c_{m-1}$  can be assumed to be two values 0 or 1, and so determines whether or not a particular stage is linked to the feedback loop or not. Consequently, there are  $2^m$  linear functions in total, as shown in Figure 1.4.

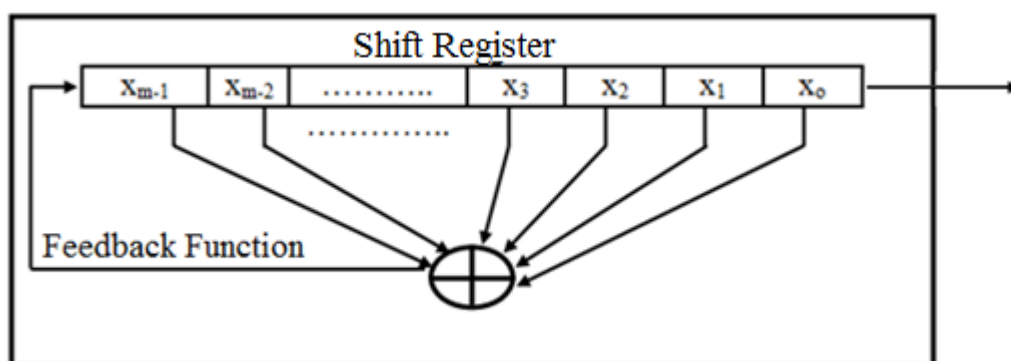


Figure 1.4 Linear Feedback Shift Register (Hell, Johansson and Meier, 2006)

The  $m$  stage in LFSR can be generated sequences with maximum period lengths of  $2^m - 1$ . If the SR is able to produce the maximum length sequences based on if  $f(x)$  is a primitive polynomial or not. The primitive polynomial of the order  $m$  is an irreducible polynomial which is can not to be factorized further, and which has an exponent equal to  $2^m - 1$ . If  $f(x)$  is a primitive polynomial of order  $m$  then the connected shift register will create the maximum length sequence with a period equal to  $2^m - 1$  (Hell et al, 2006).

### 1.1.5 Feedback with Carry Shift Registers

The feedback with carry shift register (FCSR) is alike to (LFSR) that both of them have SRs and feedback functions, but the FCSR has a carry register as shown in Figure 1.5.

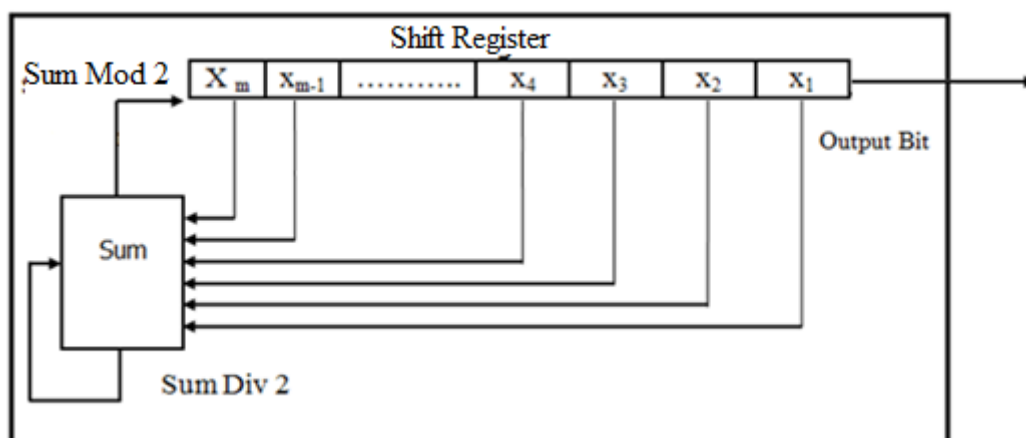


Figure 1.5 Feedback with Carry Shift Register (Arnault, Berger and Pousse, 2011)

Additionally, in place of XO-Ring all the bits in the tap sequence added the bits collectively and added the contents of the carry register. The summation result mod 2 becomes the new bit, and summation div 2 is the new content of the carry register (Arnault, Berger and Pousse, 2011).

### 1.1.6 Non-Linear Feedback Shift Registers

Non-linear FSR is able to be design by defining the non-linear feedback functions. It is easy to visualize more complex feedback sequence than the one used in LFSRs or FCSRs. The problem is that there isn't any numerical theory that can be analyzing. Particularly, here are few problems with non-linear feedback shift register sequences (Chen et al, 2005):

1. There may be biases, i.e. in the output sequence more 1's than 0's or fewer runs than expected.
2. The maximum period of the sequence may be much lower than the predictable.
3. The period of the sequence may be dissimilar for different string values.
4. The sequence may be appeared random for a while, but then "dead end" into a single value. (This can simply be solved by XO-Ring the non-linear function with right-most bits).

On the other side, if there is no theory to analyze the non-linear feedback shift registers for security, there are few tools to cryptanalyst the stream cipher depend on. The non-linear feedback shifts registers can be used in a stream cipher design with careful. In the non-linear feedback shift register, the feedback function can be anything.

### 1.1.7 Non-linear combiner of LFSRs

The essential technique to design the key-stream generator is the LFSRs. It is easy first take one or more LFSRs, usually of dissimilar lengths and with different feedback polynomials. (If the lengths are all relatively prime and the feedback polynomials are all primitives; the whole generator is maximal (length)). The output bit is a function if at all possible a non-linear function of some of the bits of the LFSRs. This function is called



the combining function, while the whole generator is called a combination generator (Deepthi, John and Sathidevi, 2009).

Complications have been added. Some generators have LFSRs clocked at different rates, and sometimes the clocking of one generator depends on the output of the other one. Clock control can feed onward where the output of one LFSR controls the clocking of another, or feedback where the output of one LFSR controls its own clocking (Deepthi, et al, 2009).

### 1.1.8 Random Properties of Sequences

The sequences with a large period have an advantage that is their predictability is far smaller than the sequences with a small period. However, this is not the only principle for cryptographic applications. For instance a particular text with a long period is represented by a sequence of first all zeros and then all ones. It will be completely inappropriate for enciphering since, the ciphertext will be equal to the plaintext as long as only zeros are processed. When ones are created the ciphertext will be equal to the inverted plaintext. Sufficiently random means that succeeding bits cannot be predicted easily, if a given number of bits of the sequence is known. One probable description of a pseudorandom binary sequence was proposed by (Golomb, 1989). Golomb defined a pseudo-random sequences "pseudo-noise sequences" to be a binary sequence of period (P) that satisfy the next three randomness postulates:

**R1:** If P is even and the cycle of a length equal to P should have an equal number of zeros and ones, if P is odd and the number of zeros shall be one less or more than the number of ones.

**R2:** In the cycle of length equal to P, half of the runs have length equal to 1 a quarter have length equal to 2, an eighth have length equal to 3 and in general for every i for

which there are at least  $1/2^i$  of runs have length equal to  $i$ . Moreover, for each one of these lengths there are uniformly many gaps and blocks. A run of length equal to  $r$  is a string of  $r$  the same bits which is both preceded and succeeded by the reverse bit.

**R3:** The out-of-phase auto-correlation is a stable or constant. The auto-correlation function  $C(\tau)$  of a binary sequence  $S_0S_1S_2\dots\dots\dots$  of a period  $P$  is defined by:

$$C(\tau) = \frac{A(\tau) - D(\tau)}{P} \dots\dots\dots \text{Equation 1.4, (Carter, 1989)}$$

Where;

1.  $A(\tau)$  is the position numbers in which  $S_0S_1S_2\dots S_{p-1}$  and  $S_\tau S_{\tau+1}\dots S_{\tau+p-1}$  agree on the key.
2.  $D(\tau)$  is the position numbers in which disagree on the key.

Over and over again one can discover that a sequence is not random, but certify that a sequence is random is a hard mission indeed. As a suggestion, no sequence created by computer ability, in fact, its random but getting sequence that exhibits a lot of the properties for the random numbers. Unluckily, it is regularly not possible to articulate precisely which properties of random numbers are significant for an exacting application (Carter, 1989).

## 1.2 Research Problem

According to the traditional conceptions over secure group communications such as (confidentiality, authenticity and integrity), it will become critical networking issues. So the secure communication between parties takes the greatest attention in recent researches.

There are many problems that occur in the communication process, one of these problems is to produce a pure key bits according to the silence periods in the speech encryption due to the frequently generating zero's or one's to represent the silence between pronounced words during speech.

As the Internet evolves and computer networks become bigger and bigger, network security has become one of the most important factors for companies to consider. By increasing network security, you decrease the chance of privacy spoofing, identity or information theft and so on. Piracy is a big concern to enterprises that are victims of its effects.

Encryption here is so important because it allows a people to securely protect their data. Businesses use it to protect corporate secrets, government use it to secure classified information, and many individuals use it to protect personal information to guard against things like identity theft.

### **1.3 Research Objectives**

The main objective of this research is to develop a new concept of the stream cipher algorithms in voice encryption which used in an advanced application for protecting data and information in network communications.

This research aims to protect and encrypt the voice messages or the voice data by implementing the twin algorithm with computing the period which must be computationally secure and proving that by passing the standard statistical randomness tests. Also this research to design a new LFSR- based stream cipher algorithm and to design a new non-linear compound functions.

## 1.4 Motivation

Encryption information is an attractive problem to solve and to find the best method that increase the security in transferring data, due to the lack of researches about the voice encryption this research has been done. This thesis has two reasons of motivations; the first one is building an algorithm to solve real problem due to the weakness point used always to penetrate algorithms designed then to protect speech data, where the second one is in solving real problem exist in the cost of building the hardware.

## 1.5 Significance of Work

The innovation of the concept of the twin stream cipher algorithm design is the main and the novel designing concept. It is a new approach to create cryptographic algorithms. The concept of this research is to create two symmetric algorithms and to produce a key bit from one algorithm at a time using a random method. The reason of this research is refers for the creating such algorithms to avoid producing pure key bits corresponding to the silence periods in a speech encryption. This research will evaluated by testing a samples after building an application uses by LFSR-based stream cipher.

## 1.6 Thesis outline

**Chapter One** introduces the cryptography, encryption, security, in addition to define the aims, objectives, problem statement, research motivations and the significance of work.

**Chapter Two** introduces some of the recent works that are related to the encryption, stream cipher, and the algorithms used in voice encryption.

**Chapter Three** presented the way that the project has been done with all algorithm details.

**Chapter Four** presented and discussed the experimental results on the thesis.

**Chapter Five** concluded the results and the main points mentioned in the thesis.

---

# **Chapter Two**

## **Backgrounds and Literature Review**

## **2 Chapter Two Background and Literature Review**

### **2.1 Overview**

This chapter will review the previous studies that have done about the voice encryption. Subjects that have been studied in this chapter include: a background about the stream cipher, the usage of stream cipher, types of stream cipher, security, and a comparison between the stream ciphers.

### **2.2 Background**

A stream cipher is defined as a symmetric key cipher wherever plaintext digits are joint with a pseudorandom cipher digit stream (key-stream). In the stream cipher every plaintext digit is encrypted one at a time with the corresponding digit of the key-stream, to offer a digit of the stream cipher. Meanwhile, the encryption of every digit is reliant on the existing case of the cipher; consequently it is moreover recognized as the state cipher (Seddiki, O., et. al, 2014).

The pseudorandom key-stream is classically produced successively from a haphazard seed value utilizing digital shift registers. The seed value serves as the cryptographic key for decrypting the ciphertext stream. Stream ciphers denote a different method to symmetric encryption from block ciphers. Block ciphers run on a large block of digits with a secure, unvarying transformation. This difference is not continually clear-cut: in some modes of process, a block cipher primitive is utilized in such a method that it acts efficiently as a stream cipher. Stream ciphers classically perform at a higher speed than block ciphers and have lower hardware complexity. Nevertheless, stream ciphers can be vulnerable to severe security problems if they used

incorrectly. In specific, the same starting state (seed) must never be used twice (Ammar, M., 2014).

### **2.2.1 Stream Cipher Usage**

Stream ciphers are regularly utilized for their speed and effortlessness of application in the hardware, and in applications where plaintext comes in amounts of unknowable distance like a safe wireless connection. If a block cipher (not working in a stream cipher mode) used in this kind of application, the designer would requisite to select either broadcast efficiency or application complexity. Meanwhile, block ciphers cannot work on blocks smaller than their block size. For instance, if a 128-bit block cipher customary distinct 32-bit bursts of plaintext, three quarters of the info conveyed would be padding. Block ciphers have to be utilized in ciphertext stealing or remaining block termination mode to evade padding, while stream ciphers remove this matter by working on the minimum unit that can be transmitted (usually bytes) (Turan, M. S., 2010).

Additional benefit of the stream ciphers in provided cryptography is that the cipher stream can be produced in a distinct box that is focus to strict security measures and fed to other devices such as a radio set, which will achieve the X-or process as a part of their purpose. The latter device can then be intended and utilized in less stringent environments.

RC4 is the most widely utilized stream cipher in software; others include: A5/1, A5/2, Chameleon, FISH, Helix, ISAAC, MUGI, Panama, Phelix, Pike, SEAL, SOBER, SOBER-128 and WAKE (Turan, M. S., 2010).



### 2.2.2 Stream Cipher Security

For the stream cipher to be safe the keys need to have the greatest period and it necessity to be difficult to improve the cipher's key or interior state from the key-stream. Cryptographers similarly request that the key-stream be allowed of even thin biases that would tenancy agreement attackers distinguish the stream from a random noise, and a free of obvious relationships among the key-streams that match up to a related keys or a related cryptographic nonce. This must be right for all keys (there should be no weak keys), and true even if the attacker can know or choose some plaintext or ciphertext (Campbell, J.M., 2015).

As with further attacks in the cryptography, stream cipher occurrences can be certificated, sense they are not essentially applied techniques to break the cipher but designate that the cipher may have additional weaknesses.

Strongly utilizing protected synchronous stream cipher needs that one not ever use again the same key-stream twice; that normally means a different nonce or key have to be provided to every invocation of the cipher. Application designers have also recognized that the most stream ciphers don't offer authenticity, only secrecy: encrypted messages may still have been adapted in transit. Dumpy periods for thee stream ciphers have been an applied worry. For instance, 64-bit block ciphers like DES can be utilized to produce the keys stream in the output feedback (OFB) mode (Katz, J., 2014).

Particular applications utilizing the stream cipher RC4 are attackable due to the weaknesses in RC4's key system routine; new applications must either evade RC4 or make sure all keys are exclusive and ideally unrelated (such as produced by a well-seeded CSPRNG or a cryptographic hash function) and that the initial bytes for the key-stream are discarded (Maitra, S., 2014).

### **2.2.3 Types of Stream Cipher**

The stream cipher produces successive bits of the key-streams based on an inner state. The state of the key-stream is efficient essentially into two ways: if this state changes self-sufficiently of ciphertext or plaintext messages, the ciphers are characterized as a synchronous stream cipher. By alteration, self-synchronising stream ciphers bring up-to-date their state based on the preceding ciphertext digits.

#### **2.2.3.1 Synchronous Stream Ciphers**

In modern stream ciphers, the initial state of the key-stream generator is obtained not only from the key but also from a public initialization vector IV. This IV vector is changed for each new frame encryption, and can be transmitted with no specific protection. Hence, a synchronous stream cipher can be depicted.

In the synchronous stream cipher the stream of pseudo-random digits is produced self-sufficiently for the plaintext and ciphertext messages, and then joint with the plaintext (to encrypt) or the ciphertext (to decrypt). In the greatest communal form, binary digits are utilized (bits), and the key-stream is mutual with the plaintext utilizing the exclusive or process (XOR). This is called a binary additive stream cipher.

In the synchronous stream cipher, the source and receiver have to be precisely in a step for the decryption to be successful. If the digits are added or deleted from the message through the transmission, synchronisation is misplaced. To reinstate synchronisation, numerous offsets may be exasperated systematically to find the right decryption (Cheng, C., 2014).

On the other hand, a digit is tainted in the transmission more than supplementary or lost, lone a single digit in the plaintext is artificial and the mistake does not broadcast to

the other parts of the message. This thing is valuable when the transmission mistake in a high rate; though, it creates less likely the mistake would be noticed deprived of the further mechanisms. Furthermore, due to this property, synchronous stream ciphers are actual vulnerable to the active attacks: if an attacker can change digits in the ciphertext, attackers may be able to make expectable variations to the corresponding plaintext bit; for instance, flipping a bit in the ciphertext reasons the same bit to be flipped in the plaintext (Sharma, P., 2015).

### **2.2.3.2 Self-Synchronizing Stream Ciphers**

Additional method uses several of the previous  $N$  ciphertext digits to calculate the key-stream. Such systems are known as self-synchronizing stream ciphers, asynchronous stream ciphers or ciphertext auto key (CTAK). The knowledge of the self-synchronization was patented on 1946, and has the benefit that the earphones will mechanically synchronise with the key-stream producer after getting  $N$  ciphertext digits, making it easy to improve if digits are dropped or added to the message stream. Single-digit mistakes are limited in their effect, affecting only up to  $N$  plaintext digits. An example of a self-synchronising stream cipher is a block cipher in cipher feedback (CFB) mode (Liu, X., 2014).

### **2.2.4 Comparison of Stream Cipher**

The table below shows a comparison between different types of stream cipher in many parameters like the speed, key length, internal state, and the creation data.

Table 2.1 Comparison of Stream Cipher

Stream Cipher	Creation Date	Speed (cycles per byte)	(bits)			Attack	
			Effective Key-Length	Initialization vector	Internal State	Best Known	Computational Complexity
A5/1	1989	Voice ( $W_{\text{phone}}$ )	54 or 64 (in 2G)	22 (in 2G)	64	Active KPA OR KPA Time-Memory Tradeoff	~2 seconds OR $2^{30.91}$
A5/2	1989	Voice ( $W_{\text{phone}}$ )	54	114	64?	Active	4.6 milliseconds
Achterbahn-128/80	2006	1 (hardware)	80/128	80/128	297/351	Brute force for frame lengths $L \leq 2^{44}$ . Correlation attack for $L \geq 2^{48}$	$2^{80}$ resp. $2^{128}$ for $L \leq 2^{44}$ .
CryptMT	2005	?	Variable	up to 19968	19968	N/A (2008)	N/A (2008)
FISH	1993	?	Variable	?	?	Known-plaintext attack	$2^{11}$
Grain	Pre-2004	?	80	64	160	Key-Derivation	$2^{43}$
HC-256	Pre-2004	4 ( $W_{P4}$ )	256	256	65536	?	?
ISAAC	1996	2.375 ( $W_{64\text{-bit}}$ ) - 4.6875 ( $W_{32\text{-bit}}$ )	8-8288 usually 40-256	N/A	8288	(2006) First-round Weak-Internal-State-Derivation	$4.67 \times 10^{1240}$ (2001)
MUGI	1998–2002	?	128	128	1216	N/A (2002)	$\sim 2^{82}$
PANAMA	1998	2	256	128?	1216?	Hash Collisions (2001)	$2^{82}$
Phelix	Pre-2004	up to 8 ( $W_{x88}$ )	256 + a 128-bit Nonce	128?	?	Differential (2006)	$2^{37}$
Pike	1994	?	Variable	?	?	N/A (2004)	N/A (2004)
Py	Pre-2004	2.6	8-2048? usually 40- 256?	64	8320	Cryptanalytic Theory (2006)	$2^{75}$
Rabbit	2003-Feb	3.7( $W_{P3}$ )-9.7( $W_{ARM7}$ )	128	64	512	N/A (2006)	N/A (2006)
RC4	1987	7 $W_{P5}^{[1]}$	8-2048 usually 40-256	RC4 does not take an IV. If one desires an IV, it must be mixed into the key somehow.	2064	Shamir Initial-Bytes Key-Derivation OR KPA	$2^{13}$ OR $2^{33}$
Salsa20	Pre-2004	4.24 ( $W_{G4}$ ) - 11.84 ( $W_{P4}$ )	256	a 64-bit Nonce + a 64-bit stream position	512	Probabilistic neutral bits method	$2^{251}$ for 8 rounds (2007)
Scream	2002	4 - 5 ( $W_{\text{soft}}$ )	128 + a 128-bit Nonce	32?	64-bit round function	?	?
SEAL	1997	?	?	32?	?	?	?
SNOW	Pre-2003	?	128 OR 256	32	?	?	?
SOBER-128	2003	?	up to 128	?	?	Message Forge	$2^{-6}$
SOSEMANUK	Pre-2004	?	128	128	?	?	?
Trivium	Pre-2004	4 ( $W_{x88}$ ) - 8 ( $W_{LG}$ )	80	80	288	Brute force attack (2006)	$2^{135}$
Turing	2000–2003	5.5 ( $W_{x88}$ )	?	160	?	?	?
VEST	2005	42 ( $W_{ASIC}$ ) - 64 ( $W_{FPGA}$ )	Variable usually 80-256	Variable usually 80-256	256 - 800	N/A (2006)	N/A (2006)
WAKE	1993	?	?	?	8192	CPA & CCA	Vulnerable

## 2.3 Literature review

In the following, some proposals were drawn from the scientific literature for the implementation of search algorithms in the solution of the designing key-stream problem.

### ❖ (Castro, A., et al., 2014)

The authors analyzed the family of the stream ciphers N-viums: Trivium and Bivium. Where presented the Trivium algorithm and its variants. In particular, the non-

linear shift registers (LFSRs) used in the generators, feedback functions, and combination. Because of their size, several research problems remain unanswered: patterns of behavior, algebraic properties, period lengths, and weak keys among others. Finally, they presented reducing a size a variant of these generators for research and applications in cryptology, laying out the formulae of the feedback functions as well as the key bit streams, which was assumed that the properties identified in the reduced sized models would remain invariant in the original ones. The author recommend foster an additional research in the following areas:

- Search for length of the period or cycles.
- Distribution of taps and their changes to determine algebraic properties and personalization of N-viums.
- Algebraic analysis of the non-linear functions used in the models.
- Search for possible weak keys.

#### ❖ (Martin et al., 2006)

The authors proposed a new stream cipher: Grain-128 has been presented, which the design is a new member in the family of Grain stream ciphers, and it is very simple and based on two shift registers, one linear and one non-linear, and an output function. The design parameters have been chosen based on theoretical arguments for linear approximations and other possible attacks.

The size of the key is 128 bits and the size of the IV is 96 bits. Grain-128 is very well suited for hardware environments where low gate count, low power consumption and small chip area are important requirements. One can very easily increase the speed

of the cipher at the expense of extra hardware. To knowledge, there is no 128 bit cipher offering the same security as Grain-128 and a smaller gate count in hardware.

### ❖ (Rachwlik, T., et al., 2012)

The Non-linear Feedback Shift Registers (NLFSR) utilized as a rudimentary in the cryptographic algorithms. Their philosophy is not as hard as that of the Linear Feedback Shift Registers (LFSR). In overall, it is not recognized how to structure of the NLFSRs with the maximum period. The straight method is to seek for such records with proper properties. Moreover, it reconnoitered local statistical of material goods of the binary sequences that generated by NLFSRs of order 25 and 27.

The authors were used NLFSR in Field Programmable Gate Arrays (FPGA) to perform a search of NLFSRs of the order up to  $n = 27$ , the maximum period equal to  $2^n - 1$  and a possibly simple algebraic form of the feedback function. The structure of the Algebraic Normal Form (ANF) of the feedback function was fixed in search. They put the algebraic degree of ANF equal to four and randomly chose linear and higher order terms. The hardware implementation of NLFSRs and verification modules enabled to speed search about 100 times up comparing to software implementation on current PCs. The future task would be to find NLFSRs with bigger number of stages  $n$ . This requires an improvement of searching methods and the use more hardware resources.

### ❖ (Dabrowski, P., et al., 2014)

The authors of this research were utilized the Non-linear Feedback Shift Registers (NLFSRs) to hypothesis the pseudorandom producers for the stream ciphers. Their philosophy is not as hard as that of Linear Feedback Shift Registers (LFSRs).

Generally, it is not identified how the theory of all NLFSRs with the maximum period. The direct process is to examine for such registers with proper properties.

In the paper examined for NLFSRs producing modified de Bruijn sequences. Procedures of the parallel computing have been applied both of software's and hardware's to rapidly the seeking up for the maximum period of the NLFSRs having a honestly simple algebraic of a normal form. The software has been utilized to look for singular form quadratic m-sequences. An enumeration of all quadratic m-sequences up to order  $n = 21$  generated by feedback functions with the term  $x_i + x_{ij}$  and linear terms defined by primitive polynomials has been provided and classified by the weights of these polynomials. The conjecture of Chan, Games and Rushanan has been verified numerically up to order  $n = 29$  for NLFSRs whose feedback functions have the special algebraic normal form.

Hardware search of NLFSRs is a continuation of the previous work. The feedback functions of the registers of orders  $n = 29, 30$  and  $31$  founded are expressed in terms of negations of some arguments; this gives simpler formulae than ANFs and it is suitable for implementation.

### ❖ (Masoodi, F., et al., 2012)

Cryptographically protected the pseudorandom number of producers are of pivotal significance for the stream cipher design and the Linear Feedback Shift Registers (LFSRs) though not protected due to their ingrained linearity are normally utilized as a part of key-stream producers in the stream ciphers due to the worthy statistical properties, great periods and less carrying out costs. LFSRs have continuously established substantial care in the cryptography. Owed to the worthy statistical

properties, great period and less implementation costs, LFSR have been accomplished wide approval in emerging stream ciphers.

This study means to current self-contained and inclusive analysis of the linear feedback shift registers and its applications in the stream ciphers. The research here focused on examining the mechanism of the LFSR, the two applications differences and numerous properties of the LFSR, which played an important role in designing the stream cipher. In the last part of this study, the authors analyzing the security feature of the LFSR depend on stream ciphers and dissimilar methods to make known to non-linearity in the produced sequence, so as to make it cryptographically more safe.

### ❖ (Knellwolf, S., et al., 2012)

The idea of the conditional variance cryptanalysis had been practical to NLFSR-based cryptosystems at ASIACRYPT 2010. The writers of this research enhanced the procedure by utilizing the automatic tools to obtain and analyze the complicated conditions. Utilizing these developments crypt analyzes the stream cipher Trivium and the KATAN group of lightweight blocks ciphers. For both ciphers achieve new cryptanalytic outcomes.

They evaluated the security of Trivium and KATAN with respect to conditional differential cryptanalysis, and used an automatic approach to find and analyze the conditions in terms of polynomial ideals. For reducing Trivium they identified a class of 226 keys that can be distinguished for 961 of 1152 rounds. For reduced KATAN they presented a key recovery attack up to 120 of 254 rounds in a related key scenario of KATAN32, KATAN48 and KATAN64, respectively. KATAN seems to have a comfortable security margin with respect to the approach described in this paper.



### ❖ (Win and kyaw, 2008)

This study is taken into thoughtfulness the issue of cryptanalysis of the stream ciphers. At hand is some stabs require to advance the current attacks on the stream cipher and to create an attempt to characterize the parts of the ciphertext achieved by the encryption of the plaintext in which certain parts of the text are random and the rest of text are non-random. This study offers a tutorial outline to the symmetric cryptography. The elementary data theoretic and the computational characteristics of the classic and the modern cryptographic schemes are offered, followed by an investigation of the solicitation of the cryptography to the security of the VoIP scheme in computers networks utilizing LFSR algorithm. The employment program will be advanced Java 2. LFSR algorithm is suitable for the encryption and the decryption of online streaming info, e.g. VoIP (voice chatting over IP). This study is realized the encryption module of the speech signals to the ciphertext and the decryption module of the ciphertext to the speech signals.

### ❖ (Ashraf D. Elbayoumy and Simon J. Shepherd, 2006)

When the networks are homogeneous, selecting the cipher sort for a packet telephony application is modest. It is obvious that the stream ciphers accomplish better than block ciphers over the landline, circuit-switched network, meanwhile and the losses is negligible in these networks but corruption is not. Similarly, it is also obvious that the block ciphers accomplish better than stream ciphers over the landline, packet-switched networks, subsequently corruption is tiny in these networks but loss is not. Nevertheless, the choice of cipher is not so obvious for a heterogeneous internetwork covering a mix of packet and circuit-switched networks. Moreover, this matter turns out

to be even more confused when heterogeneous internetwork also contains of wireless links. Finding encryption systems will damage the performance in the heterogeneous internetwork because such internetworks have considerable loss and corruption.

Therefore, the error chattels would degrade the individual quality of the packet telephony application. In this study they offer experimental results of the comparing block and the stream ciphers when utilized to protect VoIP in terms of end-to-end delay and individual quality of the perceived voice. They projected a fresh method, which delivers automatic synchronization of the stream ciphers on a per packet basis, deprived of the overhead of an initialization vector in packet headers or deprived of maintaining any state-run of the past-encrypted info. They display that this method mitigates the trade-off among the subjective quality and the confidentiality.

### ❖ (Khaled Merit and Abdel azziz Ouamri, 2012)

Global Systems for the Mobile Communications (GSM) is considered as one of the most frequently utilized cellular technologies in the world. One and only of the objects in the mobile communication systems is the security of the exchanged information. GSM services numerous cryptographic algorithms for the security like A5/1, A5/2 and A5/3. Despite that, these algorithms do not deliver adequate level of the security for the protecting of the confidentiality for GSM. Consequently, it is wanted to upsurge the security by additional encryption approaches. This study presented a voice encryption technique called: “DES with Random permutation and Inversion”, depend on the current voice channel, which overpowers info channel's insufficiencies and solves the problematic of the penetrating, the RPE-LTP vocoder by the encrypted voice. The planned technique full filled an end-to-end protected communication in the GSM;

protect a good compatibility to the all GSM networks, and informal implementation deprived of any alteration in these systems.

### ❖ (Musheer, et al., 2012)

A great dimensional chaotic systems founded mixed key-stream producer is planned to protect the voice info. As the voice-based communication turn out to be comprehensively vital in the areas of application of military, phone banking, voice-conferencing, voice over IP, news telecasting etc. It importantly demands to reserve sensitive voice signals from the unauthorized listening and illegal practice above shared/open networks. To outline the need, the designed key-stream producer employed to work as a symmetric encryption practice to guard voice bit streams over unconfident transmission station. The producer applies the topographies of great dimensional chaos like Chen and Lorenz systems to produce extremely random-like and unpredictable sequences. The encryption key-stream is with dynamism take out from the pre-treated chaotic mixed sequences that are then used to mask the voice bit stream for honesty guard of the voice data. The investigational examines like random-like, auto-correlation signal distribution, the key space and the key-sensitivity demonstrate the efficiency of the planned procedure for safe voice communication.

### ❖ (Backes, M, et al., 2012)

Conveying voice communication in excess of untrusted networks places private information at danger. Even though voice streams are classically encrypted to avoid any unwanted eavesdropping, extra structures of the voice communication code of behavior might still permit eavesdroppers to find out the information on the transmitted content and the speaker. They developed an original method for unveiling the characteristics of

the speakers who take part in encrypted voice communication, exclusively by eavesdropping on the encrypted traffic. Their attitude exploits the idea of the voice activity detection (VAD), an extensively utilized procedure for dropping the bandwidth consumption of the voice traffic. They showed that the reduction of the traffic caused by VAD procedures generates patterns in the encrypted traffic, which in turn of reveal of the patterns of the pauses in the underlying voice stream.

They showed that the patterns they collect are considered as speaker-characteristic, also that they are satisfactory to demoralize the anonymity of the speaker in encrypted voice communication. In an experiential setup with 20 speakers their analysis is capable to in the approved manner recognize an unknown speaker in about 48% of all cases. Their work spreads and takes a broad view in effect work that uses variable bit-rate encoding for recognizing the conversation language and the content of the encrypted voice streams.

---

# **Chapter Three**

## **Research**

### **Methodology**

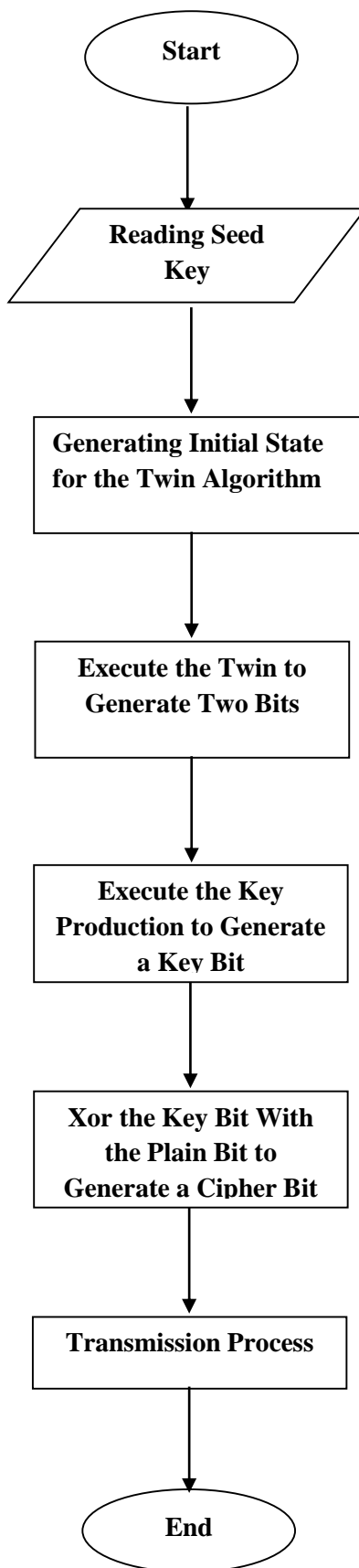
## **3 Chapter Three Research Methodology**

### **3.1 Introduction**

In this chapter a new proposed twin stream cipher algorithm is introduced. Its concept, its main features, and its characteristics are discussed in details. To prove any stream cipher is designed well (working fine), it is necessary to compute its period which must be computationally secure and prove that it has a random behaviour by passing the standard statistical randomness tests.

To prove that the twin stream cipher algorithm is designed well, a single part of the twin must be tested and the produced key sequences must be examined. The following sections will take these tasks in considerations and methodology presents as Figure 3.1.

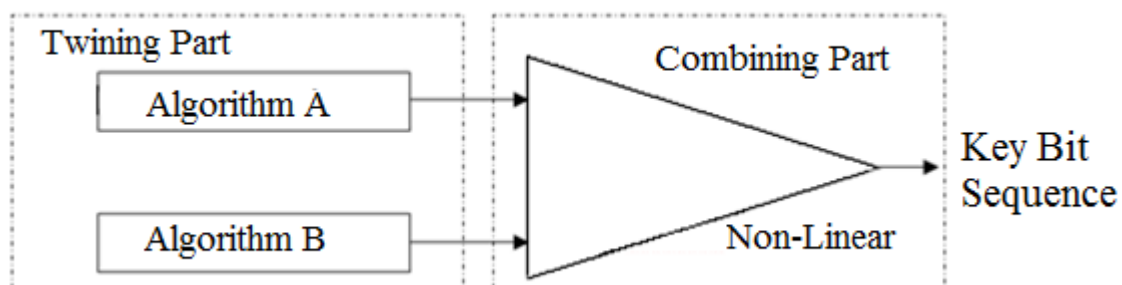
Figure 3.1 starts with reading seed key that generate the initial state for the twin algorithm to execute two bits after will happen an Xor operation for key bit with the plain bit that aimed to generate a cipher bit.



**Figure 3.1 Methodology Block Diagram**

### 3.2 The Twin Concept

Fig. 3.2 shows that the proposed algorithm consists of two main parts they are: the twining part and the combining part.

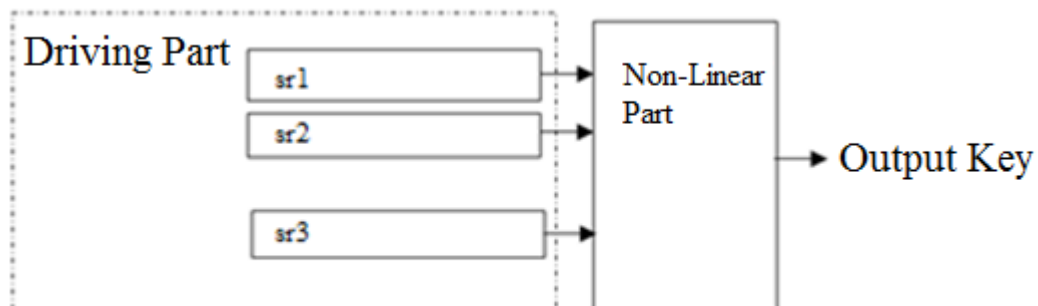


**Figure 3.2 Twin Algorithm Main Components**

The twining part must consist of two identical well-tested stream cipher algorithms (algorithm A and algorithm B). The combining part is a non-linear function its inputs were the twining part outputs and its output is the generated key bit sequence.

### 3.3 Stream Cipher Algorithm

An LFSR-based stream cipher must be combining of two parts; the driving part (linear part) and the non-linear part, as shown in Figure 3.3.



**Figure 3.3 Stream Cipher Algorithm Parts**



The driving part consists of one or more LFSR's. This part is responsible for determining the period of the algorithm. While the non-linear part is one non-linear function or mixed of linear and non-linear functions which are used to generate the key bit sequence. The non-linear part is responsible for the determining the algorithm complexity.

### **3.4 The Proposed Twin Stream Cipher Algorithm**

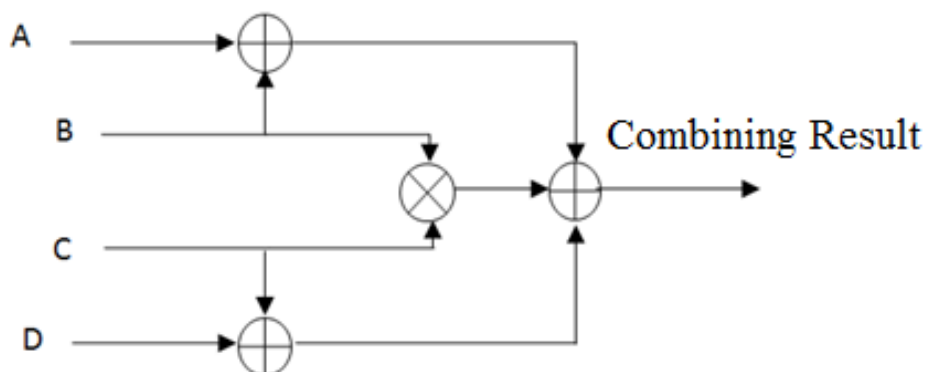
To design a twin stream cipher algorithm we must perform the following:

- a) Design a new LFSR-based stream cipher algorithm or select one of the well-known pretested stream cipher algorithm as a part of the twin.
- b) Design a new non-linear compound function or select one of the well-known pretested one.

The designed stream cipher algorithm must satisfy the conditions of designing a stream cipher algorithm by designing the driving part and the non-linear part. To do so, we must design the non-linear part first to determine the inputs number which correspond the number of LFSR's in the driving part.

### **3.5 The Combining Part**

It is too difficult to design a non-linear function that should generate a balanced sequence and has no correlation with its inputs. For this reason we decided to use the non-linear part of the stream cipher algorithm as the combining part for the twin algorithm. Fig 3.4 discussed the designed non-linear function which is used as non-linear part for the stream cipher algorithm and as combining part for the twin algorithm.



**Figure 3.4 The Non-linear Part**

Table 3.1 shows that the truth table of the function is balanced because the number of 0's and 1's are equal.

**Table 3.1 The Truth Table of the Proposed Function**

A	B	C	D	Output
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

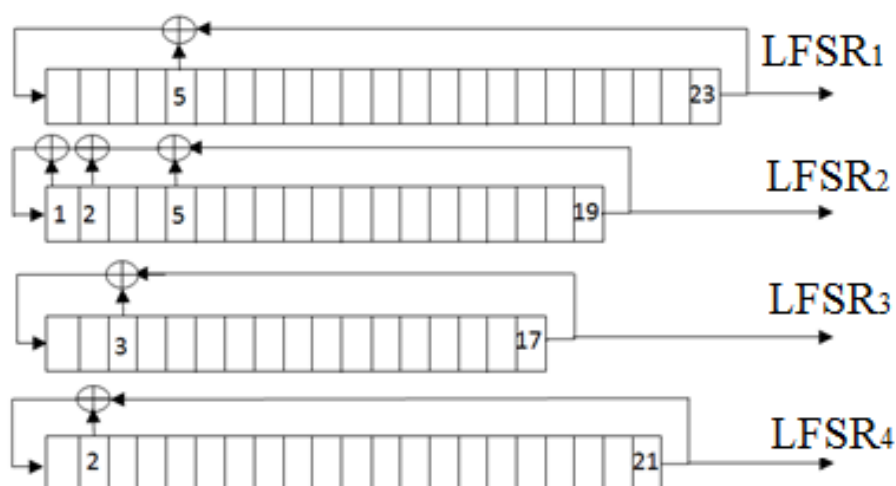
The correlation between the output and any of the input in the truth table are equal to 0.5. The correlation is computed by the following formula:

$$\text{Cor} = \frac{Ac}{Ac + Dc} \dots \dots \dots (\text{Ibadi, 2010})$$

Where Ac is the number of similarity between the output and the input, Dc is the number difference between the output and the input.

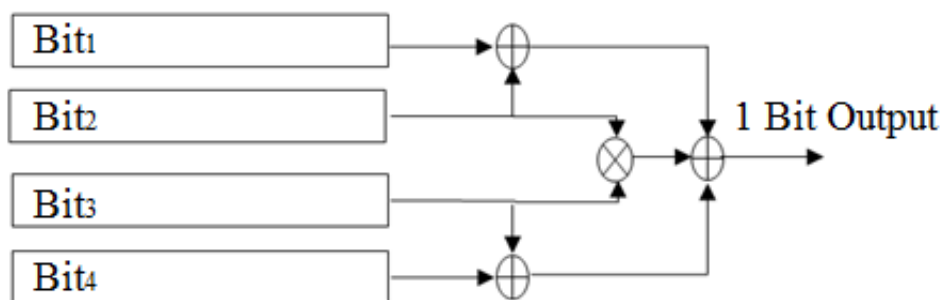
### 3.6 The Driving Part

The proposed function need four input bits to generate one output bit for this reason the driving part of the stream cipher algorithm must have four LFSR's at least. So, a suitable LFSR length must be selected with tapping stages for the feedback function giving maximum period. Figure 3.5 depicts the selected driving part for the stream cipher algorithm.



**Figure 3.5 The Driving Part of the Stream Cipher Algorithm**

The length of the LFSR's are 23, 19, 17, 21 respectively and tapping stages for the feedback functions are (23, 5), (19, 5, 2, 1), (17, 3), (21, 2) as shown in fig. 3.4. The proposed stream cipher algorithm is shown in Figure 3.6.



**Figure 3.6 The Proposed Stream Cipher Algorithm**

The proposed stream cipher algorithm passed on the statistical randomness tests. The following are the summary of the results of testing the algorithm with seed key="AAAAAAAAAA".

Sequence Length is 5003 bits

Significance Level 5%

FREQUENCY TEST

Pass Mark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

The Computed  $\chi^2$  is 0.0881471117329602

FREQUENCY TEST PASSED

SERIAL TEST

Pass Mark  $\chi^2$  is 7.53134432298715 with 3 Degree of Freedom

The Computed  $\chi^2$  is 1.45661735305878

SERIAL TEST PASSED

POKER TEST

Pass Mark  $\chi^2$  is 15.2240701045112 with 8 Degree of Freedom

The Computed  $\chi^2$  is 7.04790857142857

POKER TEST PASSED

RUN TEST

Pass Mark  $\chi^2$  for Run 0's is 19.3913495182024 with 11 Degree of Freedom

The Computed  $\chi^2$  for Run 0's is 14.1060442713864

RUN TEST for RUN 0's PASSED

Pass Mark  $\chi^2$  for Run 1's is 16.6355212541411 with 9 Degree of Freedom

The Computed  $\chi^2$  for Run 1's is 7.36586207455214

RUN TEST for RUN 1's PASSED

### AUTOCORRELATION TEST

Pass Mark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

A (1): The Computed  $\chi^2$  is 0.244902039184326 PASSED

A (2): The Computed  $\chi^2$  is 0.0721855628874225 PASSED

A (3): The Computed  $\chi^2$  is 0.125 PASSED

A (4): The Computed  $\chi^2$  is 4.44108821764353 FAILED

A (5): The Computed  $\chi^2$  is 0.245098039215686 PASSED

A (6): The Computed  $\chi^2$  is 0.480488292975785 PASSED

A (7): The Computed  $\chi^2$  is 0.105884707766213 PASSED

A (8): The Computed  $\chi^2$  is 0.0018018018018018 PASSED

## 3.7 The Proposed Twin Stream Cipher Algorithm

In the design of the proposed twin stream cipher algorithm, the non-linear part is used to be the combining part. If the output bit of the combining part is 0 then the output key bit will be computed from the base algorithm otherwise the output key bit will be computed from the twin algorithm.

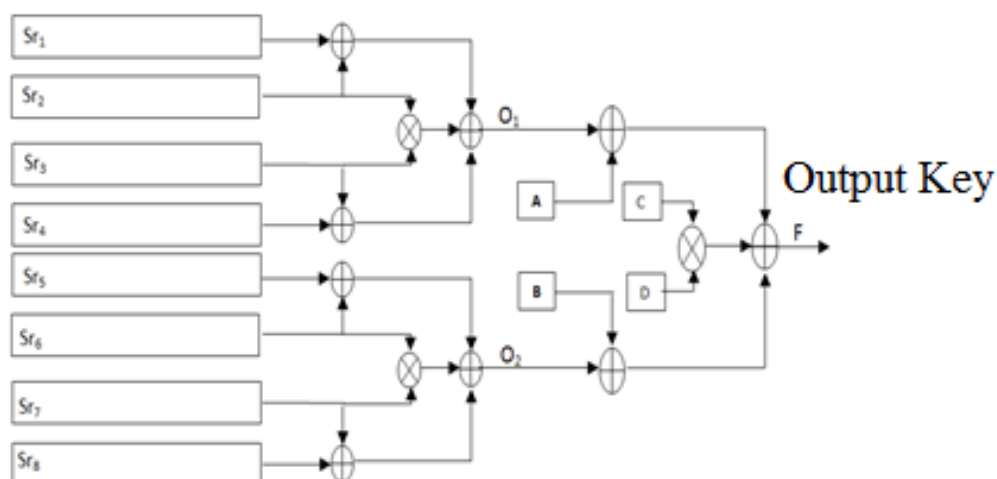


Figure 3.7 The Proposed Twin Stream Cipher Diagram

F = Output key.

A, B, C, and D are computed by the following equations:

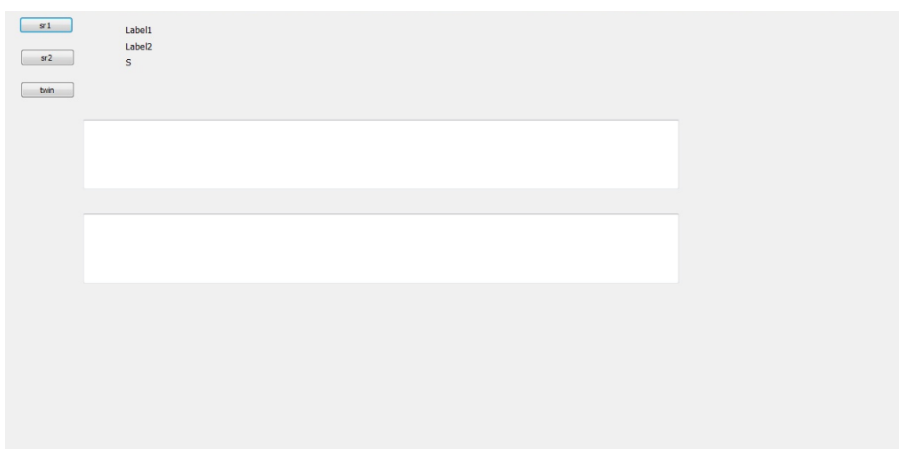
$$A = sr_1[11] \text{ xor } sr_8[7]$$

$$B = sr_3[13] \text{ xor } sr_6[17]$$

$$C = sr_2[11] \text{ xor } sr_7[7]$$

$$D = sr_4[13] \text{ xor } sr_5[13]$$

The produced key bit will be  $O_1$  if  $F=0$  else the key bit will be  $O_2$ .



**Figure 3.8 Interface of the Proposed Algorithm**



**Figure 3.9 The Proposed Program Result**

---

# **Chapter Four**

## **Results and Discussion**

## 4 Chapter Four Results and Discussion

### 4.1 Overview

In this chapter a number of output stream generating and tested of a various length to show that the algorithm has a random behavior and characteristics. Figure 4.1 shows the application interface which used to generate a sequence of output from any algorithm that concludes the twin algorithm.



Figure 4.1 Interface of Twin Stream Cipher Algorithm



Figure 4.2 Twin Stream Cipher Algorithm Program Giving Results

In the following sections we will test an output sequence of the based algorithm of standard length (5000 bits) with period 1000, to show that it has random properties and then the twin stream cipher will be tested with a different output sequence length and period.



## 4.2 The Based Algorithm Testing

A sequence of length 6003 & 500 bits is generated with an arbitrary seed key (in this test the used seed key="ABCDEFGHJIJ") with period 1000 is tested the result of the test is shown in the

**Table 4.1 The Based Algorithm Testing (6003 & 500 Bits)**

Sequence Length	Frequency Test Pass Mark Chi <sup>2</sup> is The Computed Chi <sup>2</sup> is	Serial Test Pass Mark Chi <sup>2</sup> is The Computed Chi <sup>2</sup> is	Poker Test Pass Mark Chi <sup>2</sup> is The Computed Chi <sup>2</sup> is	Run Test Pass Mark Chi <sup>2</sup> is The Computed Chi <sup>2</sup> is	Autocorrelation Test (d=8) (P=Pass, F=Fail) 3.49
6003	3.49 0.28 Passed	7.53 2.14 Passed	15.22 5.99 Passed	0's is 19.39 0's is 12.78 1's is 20.74 1's is 17.12 T0= Passed T1= Passed	PPPPPPPP
500	3.49 0.392 Passed	7.531 0.72 Passed	15.22 17.69 Failed	0's is 16.63 0's is 15.77 1's is 20.74 1's is 69.448 T0= Passed T1= Failed	PPPPPPPP

The table above concludes all results has been obtained for the base algorithm test in compare with O. Ibadi and Essa research as it shown in the table below, it's obviously that this thesis has a passed results, but on their research there are a failed results.

A sequence of length 6003 & 500 bits is generated with an arbitrary seed key (in this test the used seed key="ALIALIALIM") with period 1000 is tested the result of the test is shown in the

**Table 4.2 The Based Algorithm Testing (6003 & 500 Bits)**

Sequence Length	Frequency Test Pass Mark Chi <sup>2</sup> is The Computed Chi <sup>2</sup> is	Serial test Pass Mark Chi <sup>2</sup> is The Computed Chi <sup>2</sup> is	Poker Test Pass Mark Chi <sup>2</sup> is The Computed Chi <sup>2</sup> is	Run Test Pass Mark Chi <sup>2</sup> is The Computed Chi <sup>2</sup> is	Autocorrelation Test (d=8) (P=Pass, F=Fail) 3.49
6003	3.49 0.28 Passed	7.53 2.14 Passed	15.22 5.99 Passed	0's is 19.39 0's is 12.78 1's is 20.74 1's is 17.12 T0= Passed T1= Passed	PPPPPPPP
500	3.49 0.392 Passed	7.531 0.72 Passed	15.22 17.69 Failed	0's is 16.63 0's is 15.77 1's is 20.74 1's is 69.448 T0= Passed T1= Failed	PPPPPPPP

The table above concludes all results has been obtained for the base algorithm test in compare with O. Ibad and Essa research as it shown in the table below, it's obviously that this thesis has a passed results, but on their research there are a failed results.

**Table 4.3 New High Speed Stream Cipher Algorithms**

Sequence Length	Frequency Test ( $\leq 3.84$ )	Serial Test ( $\leq 7.81$ )	Poker Test ( $\leq 11.1$ )	Run Test T0 Gaps Test T1 Blocks Test	Autocorrelation Test (d=8) (P=Pass, F=Fail)
1000	3.13	7.33	10.16	T0=Pass T1=Pass	PPPPFPFF
5000	3.07	7.7	10.78	T0=Pass T1=Pass	PPPPFPFF
10000	2.78	7.01	10.9	T0=Pass T1=Pass	PPPPPPFF
100000	2.66	6.34	7.87	T0=Pass T1=Pass	PPPPPPPP
1000000	2.10	5.08	4.30	T0=Pass T1=Pass	PPPPPPPP

### 4.3 The Twin Algorithm Testing

A sequence of length 5003 & 800 bits is generated with an arbitrary seed key (in this test the used seed key="GGGGGGGGGG") with period 1000 is tested the result of the test is shown in the following:

**Table 4.4 The Twin Algorithm Testing (5003 & 800 Bits)**

Sequence Length	Frequency Test Pass Mark chi <sup>2</sup> is The Computed Chi <sup>2</sup> is	Serial Test Pass Mark chi <sup>2</sup> is The Computed Chi <sup>2</sup> is	Poker Test Pass Mark chi <sup>2</sup> is The Computed Chi <sup>2</sup> is	Run Test Pass Mark Chi <sup>2</sup> is The Computed Chi <sup>2</sup> is	Autocorrelation Test (d=8) (P=Pass, F=Fail) 3.49
5003	3.49 0.1 Passed	7.53 4.79 Passed	15.22 7.14 Passed	0's is 24.71 0's is 30.26 1's is 20.74 1's is 7.55 T0=FAIL T1=PASS	PPPPPPPP
800	3.49 0.5 Passed	7.53 0.72 Passed	15.22 19.25 Failed	0's is 16.63 0's is 7.50 1's is 20.74 1's is 51.18 T0= Passed T1= Passed	PPPPPPPP

The table above concludes all results has been obtained for the twin algorithm test in compare with O. Ibadi and Essa research as it shown in the table3, it's obviously that this thesis has a passed results, but on their research there are a failed results.

A sequence of length 6000 & 400 bits is generated with an arbitrary seed key (in this test the used seed key="AAAAAAAAAA") with period 1000 is tested the result of the test is shown in the

**Table 4.5 The Twin Algorithm Testing (6000 & 400 Bits)**

Sequence Length	Frequency Test Pass Mark $\chi^2$ is The Computed $\chi^2$ is	Serial Test Pass Mark $\chi^2$ is The Computed $\chi^2$ is	Poker Test Pass Mark $\chi^2$ is The Computed $\chi^2$ is	Run Test Pass Mark $\chi^2$ is The Computed $\chi^2$ is	Autocorrelation Test (d=8) (P=Pass, F=Fail) 3.49
6000	3.49 4133.4 Failed	7.53 6230.154 Failed	15.22 133033.92 Failed	0's is 16.6355 0's is 1056.36 1's is 20.74 1's is 1063.24 T0= Failed T1= Failed	FFFFFFFF
400	3.49 0.16 Passed	7.53 1.24 Passed	15.22 34.37 Failed	0's is 16.6355 0's is 45.722 1's is 20.74 1's is 87.287 T0= Failed T1= Failed	PPPPPPPP

The table above concludes all results has been obtained for the twin algorithm test in compare with O. Ibadi and Essa research as it shown in the table3, it's obviously that this thesis has a passed results, but on their research there are a failed results.

A sequence of length 24 & 22 bits is generated with an arbitrary seed key (in this test the used seed key="DECMECFECD") with period 800 is tested the result of the test is shown in the

**Table 4.6 The Twin Algorithm Testing (24 & 22 Bits)**

Sequence Length	Frequency Test Pass Mark Chi ^2 is The Computed Chi ^2 is	Serial Test Pass Mark Chi ^2 is The Computed Chi^2 is	Poker Test Pass Mark Chi ^2 is The Computed Chi ^2 is	Run Test Pass Mark Chi ^2 is The Computed Chi^2 is	Autocorrelation Test (d=8) (P=Pass, F=Fail) 3.49
24	3.49 24 Failed	7.53 36 Failed	15.22 765 Failed	0's is -1 0's is 0 1's is -1 1's is 0 T0= Passed T1= Passed	FFFFFFFF
22	3.49 22 Failed	7.53 33 Failed	15.22 510 Failed	0's is -1 0's is 0 1's is -1 1's is 0 T0= Passed T1= Passed	FFFFFFFF

The table above concludes all results has been obtained for the twin algorithm test in compare with O. Ibadi and Essa research as it shown in the table3, it's obviously that this thesis has a passed results, but on their research there are a failed results.

A sequence of length 6000 & 400 bits is generated with an arbitrary seed key (in this test the used seed key="ZYXR VWXYZ") with period 10000 is tested the result of the test is shown in the

**Table 4.7 The Twin Algorithm Testing (6000 & 400 bits)**

Sequence Length	Frequency Pass Mark chi <sup>2</sup> is The Computed Chi <sup>2</sup> is	Serial Test Pass Mark chi <sup>2</sup> is The Computed Chi <sup>2</sup> is	Poker Test Pass Mark chi <sup>2</sup> is The Computed Chi <sup>2</sup> is	Run Test Pass Mark chi <sup>2</sup> is The Computed Chi <sup>2</sup> is	Autocorrelation Test (d=8) (P=Pass, F=Fail) 3.49
6000	3.49 24 Passed	7.53 36 Passed	15.22 765 Passed	0's is -1 0's is 0 1's is -1 1's is 0 T0= Passed T1= Passed	FFFFFFFF
400	3.49 0.18 Passed	7.53 1.55 Passed	15.22 6.12 Passed	0's is -1 0's is 0 1's is -1 1's is 0 T0= Passed T1= Passed	PPPPPPPP

The table above concludes all results has been obtained for the twin algorithm test in compare with O. Ibadi and Essa research as it shown in the table3, it's obviously that this thesis has a passed results, but on their research there are a failed results.

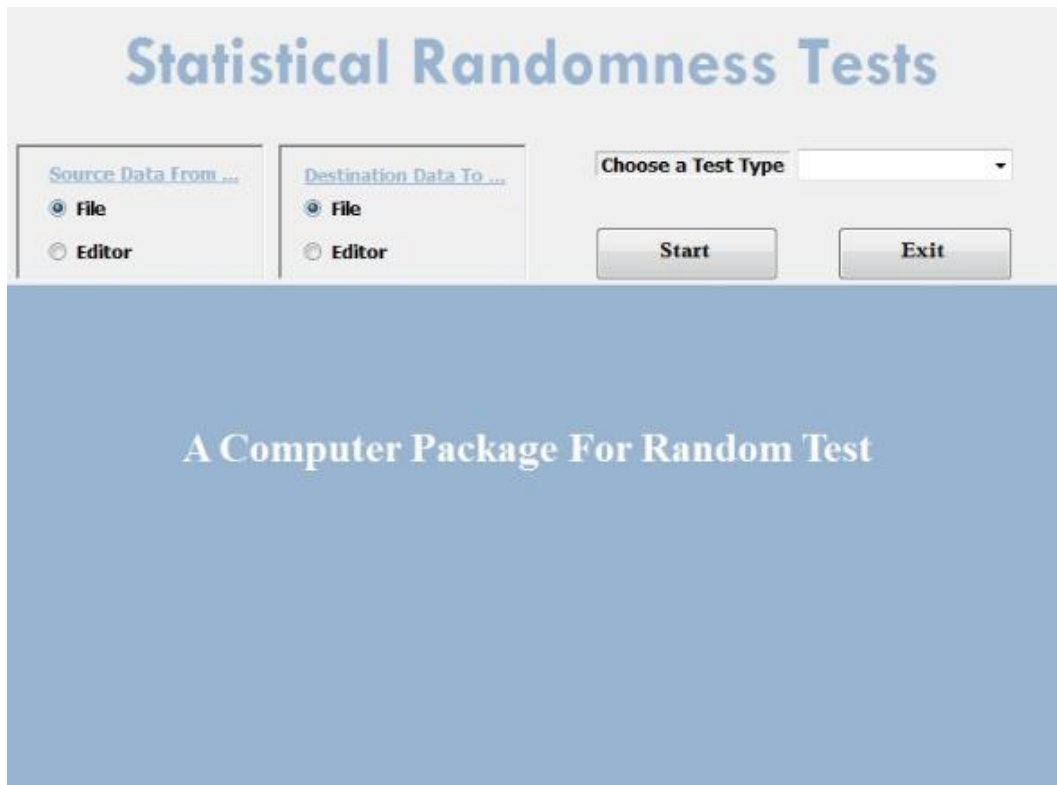


Figure 4.3 The Statistical Test Program

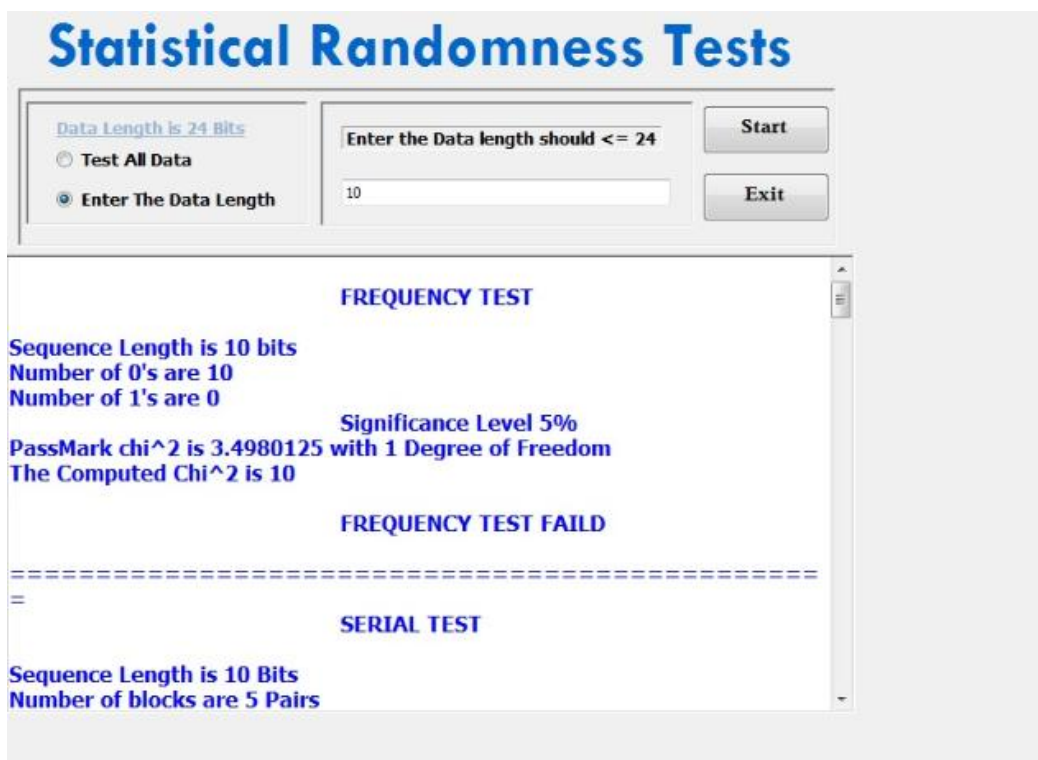


Figure 4.4 The Statistical Test Program Results





# **Chapter Five**

# **Conclusions and**

# **Future Work**

## 5 Chapter Five Conclusions and Future Work

### 5.1 Conclusion

The security of the cipher don't depend only on the algorithm's security, even though modern technology allows the encapsulation of algorithm implementations as a coordinated circuits that are resistant to reverse engineering attacks that is include secret parameters called keys, the algorithms are a public knowledge and the security of the cipher is based exclusively on the secrecy of keys.

Different algorithms suggest different degrees of security depend on how hard they are to break. An algorithm is unconditionally secure if no matter how much ciphertext of cryptanalyst has, there is not enough information to recover the plaintext. In point of fact, only one-time pad is unbreakable given infinite resources.

The twin algorithm contained of two main parts, the twining pat and the combining part. The twining part consisted of two identical well-tasted stream cipher algorithms, while the combining part is a non-linear function.ER

LFSR-based stream cipher must be combining of driving part which is the linear part and the non-linear part. The linear part is used to determine the period of the algorithm, and the non-linear part is used to generate the key bit sequence and the complexity.

Designing the non-linear part by determining the input numbers that corresponding to the number of LFSR which were four with 23,19,17,21 length and (23,5), (19,15,2,1), (17,3), (12,2) as a tapping stages for the feedback function. After that we designed the driving part to satisfy the conditions of designed stream cipher algorithm.

The proposed stream cipher algorithm passed the statistical random tests which are the frequency test, serial test, poker test, run test, and autocorrelation test.

In the twin stream cipher algorithm, the non-linear part used to be as a combining part. The output key bit was computed by the base algorithm if the output bit of the non-linear part was zero and by the twin algorithm if the output bit of the non-linear part wasn't zero.

In the based algorithm test the results of all tests were as the following:

- Frequency test was passed
- Serial test was passed
- Poker test was passed
- Run test for 0 and 1 were passed
- Autocorrelations test from A(1) to A (8) were passed

In the twin algorithm test the results of all tests were as the following:

- Frequency test was passed
- Serial test was passed
- Poker test was passed
- Run test for 0 was failed and 1 was passed
- Autocorrelations test from A(1) to A (8) were passed

## **5.2 Future work**

1. The work of a new application to link this algorithm with other algorithms to strengthen the algorithm and more difficult to break the encryption and access to any information by the attackers.
2. The development of the algorithm and its application in the field of telecommunications and military communications pertaining to state security.
3. Applying the twin algorithm in the block cipher and public cipher.
4. Enhancement this algorithm and applying in the communication network.

## References

- Abderrahim, N. W., Benmansour, F. Z., & Seddiki, O. (2014). A chaotic stream cipher based on symbolic dynamic description and synchronization. *Non-linear Dynamics*, 78(1), 197-207.
- Abdulsalam, A. A., (2011). Key-stream Generator Based On Simulated Annealing. *Journal of Applied Computer Science & Mathematics*, (10).
- Aissa, B., Nadir, D., & Ammar, M. (2014). An approach using stream cipher algorithm for image encryption and decryption. In Sciences and Techniques of Automatic Control and Computer Engineering (STA), 2014 15th International Conference on IEEE, 498-503.
- Arnault, F., Berger, T. P., Pousse, B. (2011). A matrix approach for FCSR automata. *Cryptography and Communications*, 3(2), 109-139.
- Backes, M., Doychev, G., Dürmuth, M., & Köpf, B. (2012). Speaker recognition in encrypted voice streams. *In Computer Security–ESORICS 2010, Springer Berlin Heidelberg*, 512-538.
- ÇALIK, Ç., Turan, M. S., & Özbudak, F. (2010). On feedback functions of maximum length non-linear feedback shift registers. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 93(6), 1226-1231.
- Campbell, J. M. (2015). Simple Secrecy: Analog Stream Cipher for Secure Voice Communication, 9.
- Carter G., (1989). Aspects of Local Linear Complexity. *Ph.D. Thesis, University of London*, 10-15.

- Castro Lechtaler, A., Cipriano, M., García, E., Liporace, J., Maiorano, A., Malvacio, E. (2014). Model design for a reduced variant of a Trivium Type Stream Cipher. *Journal of Computer Science & Technology*, 14.
- Chen, K., Henricksen, M., Millan, W., Fuller, J., Simpson, L., Dawson, E., Moon, S. (2005). Dragon: A fast word based stream cipher. *In Information Security and Cryptology—ICISC. Springer Berlin Heidelberg*, 33-50.
- Dąbrowski, P., Łabuzek, G., Rachwalik, T., Szmidt, J. (2014). Searching for non-linear feedback shift registers with parallel computing. *Information Processing Letters*, 114(5), 268-272.
- Deepthi, P. P., John, D. S., & Sathidevi, P. S. (2009). Design and analysis of a highly secure stream cipher based on linear feedback shift register. *Computers & Electrical Engineering*, 35(2), 235-243.
- Elbayoumy, A. D., & Shepherd, S. J. (2007). Stream or block cipher for securing VoIP. *IJ Network Security*, 5(2), 128-133.
- Fontaine, C., (2015) Synchronous Stream Cipher. *Encyclopedia of Cryptography and Security*. 1274-1275
- G.Julius Caesar, John F. Kennedy, (2005). Security engineering: A Guide to building deniable distributed systems. *Chapter5*, 73.
- Ghosh, A., K., (2011), Data Communication and Computer networks, third edition, 33-36.
- Gupta, S. S., Maitra, S., Paul, G., & Sarkar, S. (2014). (Non-) Random Sequences from (Non-) Random Permutations—Analysis of RC4 Stream Cipher. *Journal of cryptology*, 27(1), 67-108.

- Hell, M., Johansson, T., Maximov, A., & Meier, W. (2006). A stream cipher proposal: Grain-128. *In IEEE International Symposium on Information Theory (ISIT 2006)*, 16-18.
- Hirota, O., Sohma, M., Fuse, M., & Kato, K. (2005). Quantum stream cipher by the Yuen 2000 protocol: Design and experiment by an intensity-modulation scheme. *Physical Review A*, 72(2), 022335.
- Katz, J., & Lindell, Y. (2014). Introduction to modern cryptography. *CRC Press*, 13-18.
- Knellwolf, S., Meier, W., & Naya-Plasencia, M. (2012). Conditional differential cryptanalysis of trivium and KATAN. *In Selected Areas in Cryptography, Springer Berlin Heidelberg*, 200-212.
- lantoronix, Inc, (2010), Encryption and its importance to device networking, 3-4.
- Lin, Z. (2015). On two circuit configurations of non-linear feedback shift registers. *International Journal of Information and Communication Technology*, 7(2-3), 185-201.
- Martin , Hell, Johansson, Willi Meier, (2006). A New Version of Grain-128 with Authentication, 12-17.
- Masoodi, F., Alam, S., & Bokhari, M. U. (2012). An analysis of linear feedback shift registers in stream ciphers. *International Journal of Computer Applications*, 46(17), 46-49.
- Merit, K., & Ouamri, A. (2012). Securing Speech in GSM Networks using DES with Random Permutation and Inversion Algorithm. *arXiv preprint arXiv:1208.2169*

- Moldovyan, N., Moldovyan, A., A., (2008), Data-driven Block Ciphers for Fast Telecommunication Systems. *Published book by Auerbach Publication, 23-27.*
- Musheer, A., Alam, B., & Farooq, O. (2014). Chaos Based Mixed Key-stream Generation for Voice Data Encryption. *arXiv preprint arXiv:1403.4782.*
- Rachwalik, T., Szmids, J., Wicik, R., & Zablocki, J. (2012). Generation of Non-linear Feedback Shift Registers with special-purpose hardware. *In Communications and Information Systems Conference (MCC), 2012 Military IEEE, 1- 4.*
- Singhal, N., & Raina, J. P. S. (2011). Comparative analysis of AES and RC4 algorithms for better utilization. *International Journal of Computer Trends and Technology, 2(6), 177-181.*
- Tanaka, S., Cheng, C. M., Yasuda, T., & Sakurai, K. (2014). Parallelization of QUAD Stream Cipher Using Linear Recurring Sequences on Graphics Processing Units. *In Computing and Networking (CANDAR), 2014 Second International Symposium on IEEE, 543-548.*
- Upadhyay, D., Shah, T., & Sharma, P. (2015). Cryptanalysis of hardware based stream ciphers and implementation of GSM stream cipher to propose a novel approach for designing n-bit LFSR stream cipher. *In VLSI Design and Test (VDATE), 2015 19th International Symposium on, IEEE, 1-6.*
- Win, T. L., & Kyaw, N. C. (2008). Speech Encryption and Decryption Using Linear Feedback Shift Register (LFSR). *World Academy of Science, Engineering and Technology, 48, 463-467.*
- Zhou, J., Au, O. C., Zhai, G., Tang, Y. Y., & Liu, X. (2014). Scalable Compression of Stream Cipher Encrypted Images through Context-Adaptive



Sampling. *Information Forensics and Security, IEEE Transactions on*, 9(11), 1857-1868.

## Appendix

### 1. Test Stream Cipher Algorithm Program

```

Public Class Form1
    Dim i As Integer
    Dim s As String
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        s = ""
        sr1(0) = 1
        sr1(1) = 1
        sr1(2) = 0
        For i = 1 To 7
            shift1()
            s = s + Chr(k + 48)
        Next
        Label1.Text = s

    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        s = ""
        sr2(0) = 0
        sr2(1) = 0
        sr2(2) = 1

        For i = 1 To 7
            shift2()
            s = s + Chr(j + 48)
        Next
        Label2.Text = s

    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click
        s = ""
        sr1(0) = 1
        sr1(1) = 1
        sr1(2) = 0
        sr2(0) = 0
        sr2(1) = 0
        sr2(2) = 1
        Dim m = 0
        For i = 1 To 450
            shift1()

```

```

shift2()
shift3()
' COMMENT
' If (sr1(1) Xor sr2(1) Xor (sr1(0) And sr2(0))) = 1 Then
's = s + Chr(j + 48)
'Else
's = s + Chr(k + 48)
'End If
'm = m Xor sr1(1) Xor sr2(1)
If ii = 0 Then s = s + Chr(j + 48) Else s = s + Chr(k + 48)
If (i Mod 105) = 0 Then s = s + Chr(13)
Next
Label3.Text = s
TextBox1.Text = s
'ListBox1.Text = s

```

End Sub

```

Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
    initialize()
End Sub

```

```

Private Sub Label1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Label1.Click

```

End Sub

```

Private Sub Label2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Label2.Click

```

End Sub

```

Private Sub Label3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Label3.Click

```

End Sub  
End Class

```

Public Class Form1
    Dim i As Integer
    Dim s As String
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        s = ""
        sr1(0) = 1

```

```

sr1(1) = 1
sr1(2) = 0
For i = 1 To 7
    shift1()
    s = s + Chr(k + 48)
Next
Label1.Text = s

```

End Sub

```

Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click

```

```

    s = ""
    sr2(0) = 0
    sr2(1) = 0
    sr2(2) = 1

    For i = 1 To 7
        shift2()
        s = s + Chr(j + 48)
    Next
    Label2.Text = s

```

End Sub

```

Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click

```

```

    s = ""
    sr1(0) = 1
    sr1(1) = 1
    sr1(2) = 0
    sr2(0) = 0
    sr2(1) = 0
    sr2(2) = 1
    Dim m = 0
    For i = 1 To 450
        shift1()
        shift2()
        shift3()
        ' COMMENT
        'If (sr1(1) Xor sr2(1) Xor (sr1(0) And sr2(0))) = 1 Then
        's = s + Chr(j + 48)
        'Else
        's = s + Chr(k + 48)
        'End If
        'm = m Xor sr1(1) Xor sr2(1)
        If ii = 0 Then s = s + Chr(j + 48) Else s = s + Chr(k + 48)
        If (i Mod 105) = 0 Then s = s + Chr(13)
    Next
    Label3.Text = s

```

```

    TextBox1.Text = s
    'ListBox1.Text = s

```

```

End Sub

```

```

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        initialize()
    End Sub

```

```

    Private Sub Label1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Label1.Click

```

```

End Sub

```

```

    Private Sub Label2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Label2.Click

```

```

End Sub

```

```

    Private Sub Label3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Label3.Click

```

```

End Sub
End Class

```

```

Module Module1
    Public sr1(3) As Byte
    Public sr2(3) As Byte
    Public sr3(5) As Byte
    Public ii, j, k As Integer
    Public Sub shift1()
        k = sr1(0) Xor sr1(2)
        sr1(0) = sr1(1)
        sr1(1) = sr1(2)
        sr1(2) = k
    End Sub

```

## 2. Twin Stream Cipher Algorithm Program

```

'Imports System.IO

```

```

Public Class Form1
    Public s1(23), s2(19), s3(17), s4(21) As Byte

```

```

Public t1(23), t2(19), t3(17), t4(21) As Byte
Public sums1, sums2, sums3, sums4 As Byte
Public sumt1, sumt2, sumt3, sumt4 As Byte
Public r1, r2, r3 As Byte
Public iv() As Byte = {1, 0, 0, 0, 1, 0, 1, 1, 0, 1, _
    1, 1, 1, 0, 0, 1, 1, 1, 0, 0, _
    1, 1, 0, 0, 0, 0, 0, 0, 1, 1, _
    1, 0, 1, 1, 1, 1, 1, 1, 1, 1, _
    0, 0, 0, 0, 1, 0, 1, 1, 1, 1, _
    1, 0, 0, 1, 1, 1, 1, 1, 0, 0, _
    0, 1, 0, 1, 0, 1, 0, 1, 0, 1, _
    1, 1, 1, 1, 1, 1, 1, 1, 1, 1}

Public seed As String
Public Sub initialization()
    Dim keyseq(80) As Byte
    Dim i, j, k As Integer
    seed = TextBox1.Text

    For i = 0 To 9
        k = 128
        For j = 1 To 8
            If (Asc(seed(i)) And k) > 0 Then keyseq(i * 8 + j) = 1 Else keyseq(i * 8 + j) =
0
                k = k \ 2
            Next j
        Next i
        For i = 1 To 23
            s1(i) = keyseq(i)
        Next
        For i = 1 To 19
            s2(i) = keyseq(i + 23)
        Next
        For i = 1 To 17
            s3(i) = keyseq(i + 42)
        Next
        For i = 1 To 21
            s4(i) = keyseq(i + 59)
        Next
        For i = 1 To 23
            t1(i) = iv(i - 1)
        Next
        For i = 1 To 19
            t2(i) = iv(i + 23 - 1)
        Next
        For i = 1 To 17
            t3(i) = iv(i + 42 - 1)
        Next
        For i = 1 To 21
            t4(i) = iv(i + 59 - 1)

```

Next

End Sub

```
Public Sub shifts()
    Dim i As Integer
    sums1 = s1(23) Xor s1(5)
    For i = 22 To 1 Step -1
        s1(i + 1) = s1(i)
    Next
    s1(1) = sums1
    sums2 = s2(19) Xor s2(5) Xor s2(2) Xor s3(1)
    For i = 18 To 1 Step -1
    For i = 20 To 1 Step -1
        s4(i + 1) = s4(i)
    Next
    s4(1) = sums4
    r1 = (sums1 Xor sums2) Xor (sums3 Xor sums4) Xor (sums2 And sums3)
```

End Sub

```
Public Sub shiftt()
    Dim i As Integer
    sumt1 = t1(23) Xor t1(5)
    For i = 22 To 1 Step -1
        t1(i + 1) = t1(i)
    Next
    t4(1) = sumt4
    r2 = (sumt1 Xor sumt2) Xor (sumt3 Xor sumt4) Xor (sumt2 And sumt3)
```

End Sub

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
```

```
    Dim i As Integer
    initialization()
    TextBox2.Clear()
    For i = 1 To TextBox3.Text
        shifts()
        TextBox2.Text = TextBox2.Text & r1.ToString
```

Next

End Sub

```
Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button4.Click
```

```
    Dim saveFileDialog1 As New SaveFileDialog()
```

```
    saveFileDialog1.Filter = "txt files (*.txt)|*.txt|All files (*.*)|*.*"
    saveFileDialog1.FilterIndex = 2
    saveFileDialog1.RestoreDirectory = True
```

```

    If saveFileDialog1.ShowDialog() = DialogResult.OK Then
        Dim file As System.IO.StreamWriter
        file =
My.Computer.FileSystem.OpenTextFileWriter(saveFileDialog1.FileName, True)
        file.Write(TextBox2.Text)
        file.Close()

    End If

End Sub

Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
    Dim i As Integer
    initialization()
    TextBox2.Clear()
    For i = 1 To TextBox4.Text
        shiftt()
        TextBox2.Text = TextBox2.Text & r2.ToString
    Next
End Sub

Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click
    Dim i As Integer
    initialization()
    TextBox2.Clear()
    For i = 1 To TextBox5.Text
        shifts()
        shiftt()
        r3 = (r1 Xor t1(11)) Xor (r2 Xor s1(11)) Xor ((s2(7) Xor t2(7)) And (s3(13) Xor
t3(13)) And (s4(5) Xor t4(5)))
        If r3 = 1 Then TextBox2.Text = TextBox2.Text & r1.ToString Else
        TextBox2.Text = TextBox2.Text & r2.ToString
    Next
End Sub

Private Sub Button5_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button5.Click
    End
End Sub

Private Sub TextBox2_TextChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles TextBox2.TextChanged

End Sub

```



```
Private Sub TextBox1_TextChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles TextBox1.TextChanged
```

```
End Sub
End Class
```

### 3. The Statistical Test Program

```
Imports System.IO
Public Class Form1
```

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
```

```
Dim stt As Byte
```

```
If (RadioButton1.Checked = False) And (RadioButton2.Checked = False) Then
    stt = 1
```

```
Else : stt = 0
```

```
End If
```

```
If (RadioButton3.Checked = False) And (RadioButton4.Checked = False) And (stt
= 0) Then
```

```
    stt = 2
```

```
ElseIf stt = 0 Then
```

```
    stt = 0
```

```
End If
```

```
If (ComboBox1.Text = "") And (stt = 0) Then
```

```
    stt = 3
```

```
ElseIf stt = 0 Then
```

```
    stt = 0
```

```
End If
```

```
If (RichTextBox1.Text.Length < 10) And (RadioButton2.Checked) Then
```

```
    stt = 4
```

```
End If
```

```
If stt = 0 Then
```

```
    If RadioButton1.Checked Then
```

```
        Focus()
```

```
        OpenFileDialog1.InitialDirectory = "c:\\"
```

```
        OpenFileDialog1.Title = "Select the Source Data File"
```

```

OpenFileDialog1.ShowDialog()

FNi = OpenFileDialog1.FileName
Dim inf As New FileStream(fni, FileMode.Open, FileAccess.Read,
FileShare.Read)
fs = inf.Length

Me.Hide()

Form2.Label3.Text = "Data Length is " + (fs * 8).ToString + " Bits"

Form2.Show()

Else 'if Editor
fs = RichTextBox1.TextLength
Form2.Label3.Text = "Data Length is " + (fs * 8).ToString + " Bits"
Form2.Show()
Me.Hide()
End If

If RadioButton3.Checked Then
353) fno = InputBox("Enter the output file name ", "Output Window", "", 450,
'Dim outf As New
FileOpen(1, fno, FileMode.Create, FileAccess.Write, FileShare.Write)

End If

Else 'else if stt>0
MsgBox("Please fill the correct options", MsgBoxStyle.OkOnly, "Remind
Box")
End If 'stt=0

End Sub

Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
Application.Exit()
End Sub

Private Sub RadioButton2_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles RadioButton2.Click

```

```

RichTextBox1.Visible = True
Label5.Visible = False
RichTextBox1.Enabled = True
RichTextBox1.Focus()
End Sub

```

```

Private Sub RadioButton1_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles RadioButton1.Click
    RichTextBox1.Enabled = False
End Sub

```

```

Private Sub RadioButton1_CheckedChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles RadioButton1.CheckedChanged
    RichTextBox1.Visible = False
    Label5.Visible = True
End Sub

```

```
End Class
```

```
Imports System.Math
```

```
Imports System.IO
```

```
Module Module1
```

```
    Public fs, n As Integer
```

```
    Public fni, fno As String
```

```
    Public md As Boolean
```

```
    Public tmp() As Byte
```

```
    Public STT As Boolean = False
```

```
    Dim SS As String = ""
```

```

    Public Function FREQUENCY(ByVal prn As Boolean, ByVal No As Integer) As
Double

```

```
        Dim n1 As Integer = 0
```

```
        For i As Integer = 0 To No - 1
```

```
            n1 += tmp(i)
```

```
        Next
```

```
        Dim n0 As Integer = No - n1
```

```
        Dim xcomp As Double = ((n0 - n1) ^ 2 / No)
```

```
        Dim dof As Int16 = 1
```

```
        Dim xtab As Double = 0.5 * ((1.645 + Sqrt(2 * dof - 1)) ^ 2)
```

```
        If prn Then
```

```
            If STT Then
```

```
                Form2.RichTextBox1.Text += Chr(10) + Space(50) + "FREQUENCY TEST"
+ Chr(10) + Chr(10) + "Sequence Length is " + No.ToString + " bits"
```

```
            Else
```

```

Form2.RichTextBox1.Text = Chr(10) + Space(50) + "FREQUENCY TEST"
+ Chr(10) + Chr(10) + "Sequence Length is " + No.ToString + " bits"
STT = False
End If

Form2.RichTextBox1.Text += Chr(10) + "Number of 0's are " + n0.ToString +
Chr(10) + "Number of 1's are " + n1.ToString
Form2.RichTextBox1.Text += Chr(10) + Space(50) + "Significance Level 5%"
Form2.RichTextBox1.Text += Chr(10) + "PassMark chi^2 is " + xtab.ToString
+ " with " + dof.ToString + " Degree of Freedom"
Form2.RichTextBox1.Text += Chr(10) + "The Computed Chi^2 is " +
xcomp.ToString()
If xcomp > xtab Then
Form2.RichTextBox1.Text += Chr(10) + Chr(10) + Space(50) +
"FREQUENCY TEST FAILED"
Form2.RichTextBox1.Text += Chr(10) + Chr(10) +
"=====
Else
Form2.RichTextBox1.Text += Chr(10) + Chr(10) + Space(50) +
"FREQUENCY TEST PASSED"
Form2.RichTextBox1.Text += Chr(10) + Chr(10) +
"=====
End If
Else 'if write to file

If STT Then
SS += Chr(13) + Chr(10) + Space(50) + "FREQUENCY TEST" + Chr(13) +
Chr(10) + Chr(13) + Chr(10) + "Sequence Length is " + No.ToString + " bits"
Else
SS = Chr(13) + Chr(10) + Space(50) + "FREQUENCY TEST" + Chr(13) +
Chr(10) + Chr(13) + Chr(10) + "Sequence Length is " + No.ToString + " bits"
STT = False
End If

SS += Chr(13) + Chr(10) + "Number of 0's are " + n0.ToString + Chr(13) +
Chr(10) + "Number of 1's are " + n1.ToString
SS += Chr(13) + Chr(10) + Space(50) + "Significance Level 5%"
SS += Chr(13) + Chr(10) + "PassMark chi^2 is " + xtab.ToString + " with " +
dof.ToString + " Degree of Freedom"
SS += Chr(13) + Chr(10) + "The Computed Chi^2 is " + xcomp.ToString()
If xcomp > xtab Then
SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) + Space(50) + "FREQUENCY
TEST FAILED"
SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) +
"=====
Else
SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) + Space(50) + "FREQUENCY
TEST PASSED"

```

```

SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) +
"=====
End If

PrintLine(1, SS)
If Not STT Then
    Form2.Button1.Enabled = False
    MsgBox("File write success ", MsgBoxStyle.OkOnly, "OK")
End If

End If

End Function

Public Function SERIAL(ByVal prn As Boolean, ByVal No As Integer) As Double
    No -= (No Mod 2)
    Dim M(.) As Integer = {{0, 0}, {0, 0}}
    Dim i As Integer = 0
    While (i < No - 1)
        M(tmp(i), tmp(i + 1)) += 1
        i += 2
    End While

    Dim xcomp As Double = 0
    For i = 0 To 1
        For j As Int16 = 0 To 1
            xcomp += ((M(i, j) - (No / 8)) ^ 2) / (No / 8)
        Next
    Next

    Dim dof As Int16 = 3
    Dim xtab As Double = 0.5 * ((1.645 + Sqrt(2 * dof - 1)) ^ 2)

    If prn Then
        If STT Then
            Form2.RichTextBox1.Text += Chr(10) + Space(50) + "SERIAL TEST" +
Chr(10) + Chr(10) + "Sequence Length is " + No.ToString + " Bits"

            Else
                Form2.RichTextBox1.Text = Chr(10) + Space(50) + "SERIAL TEST" +
Chr(10) + Chr(10) + "Sequence Length is " + No.ToString + " Bits"
                STT = False
            End If

            Form2.RichTextBox1.Text += Chr(10) + "Number of blocks are " + (No \
2).ToString + " Pairs"
            Form2.RichTextBox1.Text += Chr(10) + "Number of 00 Blocks are " + M(0,
0).ToString + Chr(10) + "Number of 01 Blocks are " + M(0, 1).ToString

```

```

Form2.RichTextBox1.Text += Chr(10) + "Number of 10 Blocks are " + M(1,
0).ToString + Chr(10) + "Number of 11 Blocks are " + M(1, 1).ToString
Form2.RichTextBox1.Text += Chr(10) + Space(50) + "Significance Level 5%"
Form2.RichTextBox1.Text += Chr(10) + "PassMark chi^2 is " + xtab.ToString
+ " with " + dof.ToString + " Degree of Freedom"
Form2.RichTextBox1.Text += Chr(10) + "The Computed Chi^2 is " +
xcomp.ToString()
If xcomp > xtab Then
Form2.RichTextBox1.Text += Chr(10) + Chr(10) + Space(50) + "SERIAL
TEST FAILED"
Form2.RichTextBox1.Text += Chr(10) + Chr(10) +
"=====
Else
Form2.RichTextBox1.Text += Chr(10) + Chr(10) + Space(50) + "SERIAL
TEST PASSED"
Form2.RichTextBox1.Text += Chr(10) + Chr(10) +
"=====
End If
Else 'if write to file

If STT Then
SS += Chr(13) + Chr(10) + Space(50) + "SERIAL TEST" + Chr(13) +
Chr(10) + Chr(13) + Chr(10) + "Sequence Length is " + No.ToString + " Bits"

Else
SS = Chr(13) + Chr(10) + Space(50) + "SERIAL TEST" + Chr(13) + Chr(10)
+ Chr(13) + Chr(10) + "Sequence Length is " + No.ToString + " Bits"
STT = False
End If

SS += Chr(13) + Chr(10) + "Number of blocks are " + (No \ 2).ToString + "
Pairs"
SS += Chr(13) + Chr(10) + "Number of 00 Blocks are " + M(0, 0).ToString +
Chr(13) + Chr(10) + "Number of 01 Blocks are " + M(0, 1).ToString
SS += Chr(13) + Chr(10) + "Number of 10 Blocks are " + M(1, 0).ToString +
Chr(13) + Chr(10) + "Number of 11 Blocks are " + M(1, 1).ToString
SS += Chr(13) + Chr(10) + Space(50) + "Significance Level 5%"
SS += Chr(13) + Chr(10) + "PassMark chi^2 is " + xtab.ToString + " with " +
dof.ToString + " Degree of Freedom"
SS += Chr(13) + Chr(10) + "The Computed Chi^2 is " + xcomp.ToString()
If xcomp > xtab Then
SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) + Space(50) + "SERIAL TEST
FAILED"
SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) +
"=====
Else
SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) + Space(50) + "SERIAL TEST
PASSED"

```

```

"=====
End If
PrintLine(1, SS)
If Not STT Then
    Form2.Button1.Enabled = False
    MsgBox("File write success ", MsgBoxStyle.OkOnly, "OK")
End If

End If
End Function

Public Function fac(ByVal x As Integer) As Integer
    Dim sum As Integer = 1
    For i As Integer = x To 2 Step -1
        sum *= i
    Next

    Return sum

End Function
Function combine(ByVal n As Integer, ByVal k As Integer) As Integer
    Return fac(n) \ (fac(n - k) * fac(k))
End Function
Public Function POCKER(ByVal prm As Boolean, ByVal No As Integer, ByVal b As
Integer) As Double

    Dim dof As Int16 = b

    Dim xtab As Double = 0.5 * ((1.645 + Sqrt(2 * dof - 1)) ^ 2)
    Dim n1(b) As Integer

    For i As Integer = 0 To b
        n1(i) = 0
    Next
    No -= (No Mod b)
    Dim ndx As Integer = 0
    While (ndx < No)
        Dim s As Integer = 0
        For i As Integer = 0 To b - 1
            s += tmp(ndx + i)
        Next
        n1(s) += 1

        ndx += b
    End While

    Dim pow As UInt16
    Dim ci As UInt16
    Dim tx As Double

```

```

Dim t1 As Double
Dim xcomp As Double = 0
For i As Integer = 0 To b
    ci = 1
    pow = ci << b
    tx = combine(b, i) * (1 / pow) * (No / b)
    t1 = ((n1(i) - tx) ^ 2) / tx
    xcomp += t1
Next

If prn Then
    If STT Then
        Form2.RichTextBox1.Text += Chr(10) + Space(50) + "POKER TEST" +
Chr(10) + Chr(10) + "Sequence Length is " + No.ToString + " Bits"

        Else
            Form2.RichTextBox1.Text = Chr(10) + Space(50) + "POKER TEST" +
Chr(10) + Chr(10) + "Sequence Length is " + No.ToString + " Bits"
            STT = False
        End If

        Form2.RichTextBox1.Text += Space(30) + "Block Length is " + b.ToString + "
bits"
        Form2.RichTextBox1.Text += Chr(10) + "Number of blocks are " + (No \
b).ToString + " Groups"
        For i As Integer = 0 To b
            Form2.RichTextBox1.Text += Chr(10) + "Number of Blocks has " +
i.ToString + " 1's are " + n1(i).ToString
        Next

        Form2.RichTextBox1.Text += Chr(10) + Space(50) + "Significance Level 5%"
        Form2.RichTextBox1.Text += Chr(10) + "PassMark chi^2 is " + xtab.ToString
+ " with " + dof.ToString + " Degree of Freedom"
        Form2.RichTextBox1.Text += Chr(10) + "The Computed Chi^2 is " +
xcomp.ToString()
        If xcomp > xtab Then
            Form2.RichTextBox1.Text += Chr(10) + Chr(10) + Space(50) + "POKER
TEST FAILED"
            Form2.RichTextBox1.Text += Chr(10) + Chr(10) +
"=====
"
        Else
            Form2.RichTextBox1.Text += Chr(10) + Chr(10) + Space(50) + "POKER
TEST PASSED"
            Form2.RichTextBox1.Text += Chr(10) + Chr(10) +
"=====
"

        End If
    Else 'if write to file

```



```

If STT Then
    SS += Chr(13) + Chr(10) + Space(50) + "POKER TEST" + Chr(13) +
Chr(10) + Chr(13) + Chr(10) + "Sequence Length is " + No.ToString + " Bits"

Else
    SS = Chr(13) + Chr(10) + Space(50) + "POKER TEST" + Chr(13) + Chr(10)
+ Chr(13) + Chr(10) + "Sequence Length is " + No.ToString + " Bits"
    STT = False
End If

SS += Space(30) + " Block Length is " + b.ToString + " bits"
SS += Chr(13) + Chr(10) + "Number of blocks are " + (No \ b).ToString + "
Groups"
For i As Integer = 0 To b
    SS += Chr(13) + Chr(10) + "Number of Blocks has " + i.ToString + " 1's are "
+ n1(i).ToString
Next

SS += Chr(13) + Chr(10) + Space(50) + "Significance Level 5%"
SS += Chr(13) + Chr(10) + "PassMark chi^2 is " + xtab.ToString + " with " +
dof.ToString + " Degree of Freedom"
SS += Chr(13) + Chr(10) + "The Computed Chi^2 is " + xcomp.ToString()
If xcomp > xtab Then
    SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) + Space(50) + "POKER TEST
FAILED"
    SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) +
"=====
Else
    SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) + Space(50) + "POKER TEST
PASSED"
    SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) +
"=====
End If
PrintLine(1, SS)
If Not STT Then
    Form2.Button1.Enabled = False
    MsgBox("File write success ", MsgBoxStyle.OkOnly, "OK")
End If
End If
End Function
Public Function RUN(ByVal prn As Boolean, ByVal No As Integer) As Double

Dim maxb As Integer = 50000
Dim ni(1, maxb) As Integer

For i As Integer = 0 To 1
    For j As Integer = 0 To maxb
        ni(i, j) = 0

```

Next  
Next

```

Dim old As Byte = tmp(0)
Dim cnt As Integer = 0
Dim ndx As Integer = 0
Dim mx() As Integer = {0, 0}
Dim all As Integer = 0
While (ndx < No - 1)
    ndx += 1
    cnt += 1
    If tmp(ndx) <> old Then
        all += 1
        ni(old, cnt) += 1
        If mx(old) < cnt Then mx(old) = cnt

        old = tmp(ndx)

        cnt = 0

        If ndx = No - 1 Then
            cnt += 1
            all += 1
            ni(tmp(ndx), cnt) += 1
        End If

        ElseIf ndx = No - 1 Then
            cnt += 1
            all += 1
            ni(old, cnt) += 1
        End If

    End While
Dim dof0, dof1 As Integer

dof0 = mx(0) - 1

dof1 = mx(1) - 1

Dim xtab0 As Double = 0.5 * ((1.645 + Sqrt(2 * dof0 - 1)) ^ 2)
Dim xtab1 As Double = 0.5 * ((1.645 + Sqrt(2 * dof1 - 1)) ^ 2)

Dim T() As Double = {0, 0}
For i As Integer = 0 To 1
    For j As Integer = 1 To mx(i)
        Dim mp As Double = n / (2 ^ (j + 2))
        T(i) += ((ni(i, j) - mp) ^ 2) / mp
    Next
Next

```

Next

```

If prn Then
  If STT Then
    Form2.RichTextBox1.Text += Chr(10) + Space(50) + "RUN TEST" +
Chr(10) + Chr(10) + "Sequence Length is " + No.ToString + " Bits"

  Else
    Form2.RichTextBox1.Text = Chr(10) + Space(50) + "RUN TEST" + Chr(10)
+ Chr(10) + "Sequence Length is " + No.ToString + " Bits"
    STT = False
  End If

  For i As Integer = 0 To 1
    For j As Integer = 1 To mx(i)
      If ni(i, j) > 0 Then Form2.RichTextBox1.Text += Chr(10) + " RUN " +
i.ToString + " of length " + j.ToString + " = " + ni(i, j).ToString
    Next
    Form2.RichTextBox1.Text += Chr(10)
  Next

  Form2.RichTextBox1.Text += Chr(10) + Space(50) + "Significance Level 5%"
  Form2.RichTextBox1.Text += Chr(10) + "PassMark chi^2 for Run 0's is " +
xtab0.ToString + " with " + dof0.ToString + " Degree of Freedom"

  Form2.RichTextBox1.Text += Chr(10) + "The Computed Chi^2 fo Run 0's is "
+ T(0).ToString()

  If T(0) > xtab0 Then
    Form2.RichTextBox1.Text += Chr(10) + Chr(10) + Space(50) + "RUN TEST
for RUN 0's FAILED"
    Form2.RichTextBox1.Text += Chr(10) + Chr(10) + Space(40) +
"!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!"
  Else
    Form2.RichTextBox1.Text += Chr(10) + Chr(10) + Space(50) + "RUN TEST
for RUN 0's PASSED"
    Form2.RichTextBox1.Text += Chr(10) + Chr(10) + Space(40) +
"!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!"
  End If

  Form2.RichTextBox1.Text += Chr(10) + Chr(10) + Space(10) + "PassMark
chi^2 for Run 1's is " + xtab1.ToString + " with " + dof1.ToString + " Degree of
Freedom"
  Form2.RichTextBox1.Text += Chr(10) + "The Computed Chi^2 fo Run 1's is "
+ T(1).ToString()

```



```

Else
    SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) + Space(50) + "RUN TEST for
RUN 0's PASSED"
    SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) + Space(40) +
"!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!"
End If

SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) + Space(10) + "PassMark chi^2
for Run 1's is " + xtab1.ToString + " with " + dof1.ToString + " Degree of Freedom"
SS += Chr(13) + Chr(10) + "The Computed Chi^2 fo Run 1's is " +
T(1).ToString()

```

```

If T(1) > xtab1 Then
    SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) + Space(50) + "RUN TEST for
RUN 1's FAILED"
    SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) +
"=====
Else
    SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) + Space(50) + "RUN TEST for
RUN 1's PASSED"
    SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) +
"=====
End If
PrintLine(1, SS)
If Not STT Then
    Form2.Button1.Enabled = False
    MsgBox("File write success ", MsgBoxStyle.OkOnly, "OK")
End If
End If

```

End Function

```

Public Function AUTOCORRELATION(ByVal prn As Boolean, ByVal No As
Integer, ByVal b As Integer) As Double
    Dim dof As Byte = 1

    Dim xtab As Double = 0.5 * ((1.645 + Sqrt(2 * dof - 1)) ^ 2)

    Dim seq(No - 1) As Byte
    For i As Integer = 0 To n - 1
        seq(i) = tmp(i)
    Next
    Dim dis As Integer
    Dim shft As Integer = -1

```

```

If prn Then
  If STT Then
    Form2.RichTextBox1.Text += Chr(13) + Chr(10) + Space(50) +
"AUTOCORRELATION TEST"

  Else
    Form2.RichTextBox1.Text = Chr(13) + Chr(10) + Space(50) +
"AUTOCORRELATION TEST"
    STT = False
  End If
Else
  If STT Then
    SS += Chr(13) + Chr(10) + Space(50) + "AUTOCORRELATION TEST"

  Else
    SS = Chr(13) + Chr(10) + Space(50) + "AUTOCORRELATION TEST"
    STT = False
  End If
End If

For i As Integer = 0 To No - 1
  dis += (seq(i) Xor tmp(i))
Next
Dim agree As Integer = No - dis

Dim xcomp As Double = ((dis - agree) ^ 2) / (No - (shft + 1))

If prn Then
  Form2.RichTextBox1.Text += Chr(10) + Chr(10) + Space(45) + "===== A(
" + (shft + 1).ToString + " ) ====="
  Form2.RichTextBox1.Text += Chr(10) + Chr(10) + "Sequence Length is " +
No.ToString + " Bits"
  Form2.RichTextBox1.Text += Chr(10) + "Number of Agree Bits are " +
agree.ToString
  Form2.RichTextBox1.Text += Chr(10) + "Number of Agree Bits are " +
dis.ToString
  xtab.ToString + " with " + dof.ToString + " Degree of Freedom"
  Form2.RichTextBox1.Text += Chr(10) + "The Computed Chi^2 is " +
xcomp.ToString()
  If xcomp > xtab Then
    Form2.RichTextBox1.Text += Chr(10) + Chr(10) + Space(50) +
"AUTOCORRELATION TEST FAILED"
    Form2.RichTextBox1.Text += Chr(10) + Chr(10) +
"+++++"
  Else
    Form2.RichTextBox1.Text += Chr(10) + Chr(10) + Space(50) +
"AUTOCORRELATION TEST PASSED"

```

```

Form2.RichTextBox1.Text += Chr(10) + Chr(10) +
"+++++"
End If
Else 'if write to file
SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) + Space(45) + "===== A( " +
(shft + 1).ToString + " ) ====="
SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) + "Sequence Length is " +
No.ToString + " Bits"
SS += Chr(13) + Chr(10) + "Number of Agree Bits are " + agree.ToString
SS += Chr(13) + Chr(10) + "Number of Agree Bits are " + dis.ToString

SS += Chr(13) + Chr(10) + Space(50) + "Significance Level 5%"
SS += Chr(13) + Chr(10) + "PassMark chi^2 is " + xtab.ToString + " with " +
dof.ToString + " Degree of Freedom"
SS += Chr(13) + Chr(10) + "The Computed Chi^2 is " + xcomp.ToString()
If xcomp > xtab Then
SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) + Space(50) +
"AUTOCORRELATION TEST FAILED"
SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) +
"+++++"
Else
SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) + Space(50) +
"AUTOCORRELATION TEST PASSED"
SS += Chr(13) + Chr(10) + Chr(13) + Chr(10) +
"+++++"
End If
PrintLine(1, SS)

End If
End While
If ((STT) And (Not prn)) Or ((Not STT) And (Not prn)) Then
Form2.Button1.Enabled = False
MsgBox("File write success ", MsgBoxStyle.OkOnly, "OK")
End If

End Function

Public Function ALL_TESTS(ByVal prn As Boolean, ByVal No As Integer) As
Double
STT = True
FREQUENCY(prn, No)
SERIAL(prn, No)

Dim b As Integer = 8 'Val(InputBox("Enter Block Length", "POKER TEST
BLOCK LENGTH"))

POCKER(prn, No, b)
RUN(prn, No)

```

```

Dim Aut As Integer = 8 'Val(InputBox("How Many shifts", "DETERMINE
NUMBER OF SHIFTS FOR AUTOCORRELATION TEST"))
AUTOCORRELATION(prn, No, Aut)

```

```

End Function
End Module

```

## The Results

### The Based Algorithm Testing

A sequence of length 6000 bits is generated with an arbitrary seed key (in this test the used seed key="ABCDEFGHJIJ") is tested the result of the test is shown in the

following:

Sequence Length is 6003 bits

Significance Level 5%

#### FREQUENCY TEST

Pass Mark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

The Computed  $\chi^2$  is 0.280026653339997

FREQUENCY TEST PASSED

#### SERIAL TEST

Pass Mark  $\chi^2$  is 7.53134432298715 with 3 Degree of Freedom

The Computed  $\chi^2$  is 2.14428523825392

SERIAL TEST PASSED

#### POKER TEST

Pass Mark  $\chi^2$  is 15.2240701045112 with 8 Degree of Freedom

The Computed  $\chi^2$  is 5.997225714285714

POKER TEST PASSED

#### RUN TEST

Pass Mark  $\chi^2$  for Run 0's is 19.3913495182024 with 11 Degree of Freedom

The Computed  $\chi^2$  for Run 0's is 12.7807407229816

RUN TEST for RUN 0's PASSED



Pass Mark  $\chi^2$  for Run 1's is 20.7421553558499 with 9 Degree of Freedom

The Computed  $\chi^2$  for Run 1's is 17.1215256171963

RUN TEST for RUN 1's PASSED

#### AUTOCORRELATION TEST

Pass Mark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

A (1): The Computed  $\chi^2$  is 0.839886704431857 PASSED

A (2): The Computed  $\chi^2$  is 0.12147953374438 PASSED

A (3): The Computed  $\chi^2$  is 0.04816666666667 PASSED

A (4): The Computed  $\chi^2$  is 1.4417402290040013 PASSED

A (5): The Computed  $\chi^2$  is 0.58360120040013 PASSED

A (6): The Computed  $\chi^2$  is 1.44222111055528 PASSED

A (7): The Computed  $\chi^2$  is 0.204302868579053 PASSED

A (8): The Computed  $\chi^2$  is 0.020183486238532 PASSED

#### FREQUENCY TEST

Sequence Length is 6003 bits

Number of 0's are 2981

Number of 1's are 3022

Significance Level 5%

Pass Mark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

The Computed  $\chi^2$  is 0.280026653339997

FREQUENCY TEST PASSED

---

#### SERIAL TEST

Sequence Length is 6002 Bits

Numbers of blocks are 3001 Pairs

Number of 00 Blocks are 750

Number of 01 Blocks are 762

Number of 10 Blocks are 718

Number of 11 Blocks are 771

Significance Level 5%

Pass Mark  $\chi^2$  is 7.53134432298715 with 3 Degree of Freedom

The Computed  $\chi^2$  is 2.14428523825392

**SERIAL TEST PASSED**

---

### **POKER TEST**

Sequence Length is 6000 Bits

Block Length is 8 bits

Numbers of blocks are 750 Groups

Number of Blocks has 0 1's are 4

Number of Blocks has 1 1's are 18

Number of Blocks has 2 1's are 80

Number of Blocks has 3 1's are 160

Number of Blocks has 4 1's are 211

Number of Blocks has 5 1's are 177

Number of Blocks has 6 1's are 69

Number of Blocks has 7 1's are 27

Number of Blocks has 8 1's are 4

Significance Level 5%

Pass Mark  $\chi^2$  is 15.2240701045112 with 8 Degree of Freedom

The Computed  $\chi^2$  is 5.99725714285714

**POKER TEST PASSED**

---

**RUN TEST**

Sequence Length is 6003 Bits

RUN 0 of length 1 = 731

RUN 0 of length 2 = 384

RUN 0 of length 3 = 189

RUN 0 of length 4 = 85

RUN 0 of length 5 = 50

RUN 0 of length 6 = 20

RUN 0 of length 7 = 13

RUN 0 of length 8 = 3

RUN 0 of length 9 = 5

RUN 0 of length 10 = 1

RUN 0 of length 11 = 1

RUN 0 of length 12 = 2

RUN 1 of length 1 = 742

RUN 1 of length 2 = 352

RUN 1 of length 3 = 201

RUN 1 of length 4 = 88

RUN 1 of length 5 = 39

RUN 1 of length 6 = 27

RUN 1 of length 7 = 20

RUN 1 of length 8 = 7

RUN 1 of length 9 = 5

RUN 1 of length 10 = 1

RUN 1 of length 13 = 1

Significance Level 5%

Pass Mark  $\chi^2$  for Run 0's is 19.3913495182024 with 11 Degree of Freedom

The Computed  $\chi^2$  for Run 0's is 12.7807407229816

RUN TEST for RUN 0's PASSED

Pass Mark  $\chi^2$  for Run 1's is 20.7421553558494 with 12 Degree of Freedom

The Computed  $\chi^2$  for Run 1's is 17.1215256171963

RUN TEST for RUN 1's PASSED

---

## **AUTOCORRELATION TEST**

### **A (1)**

Sequence Length is 6003 Bits

Number of Agree Bits are 3037

Number of Agree Bits are 2966

Significance Level 5%

Pass Mark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

The Computed  $\chi^2$  is 0.839886704431856

AUTOCORRELATION TEST PASSED

### **A (2)**

Sequence Length is 6003 Bits

Number of Agree Bits are 3015

Number of Agree Bits are 2988

Significance Level 5%

Pass Mark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

The Computed  $\chi^2$  is 0.121479753374438

**AUTOCORRELATION TEST PASSED**

**A (3)**

Sequence Length is 6003 Bits

Number of Agree Bits are 2993

Number of Agree Bits are 3010

Significance Level 5%

Pass Mark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

The Computed  $\chi^2$  is 0.0481666666666667

**AUTOCORRELATION TEST PASSED**

**A (4)**

Sequence Length is 6003 Bits

Number of Agree Bits are 2955

Number of Agree Bits are 3048

Significance Level 5%

Pass Mark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

The Computed  $\chi^2$  is 1.44174029004834

**AUTOCORRELATION TEST PASSED**

**A (5)**

Sequence Length is 6003 Bits

Number of Agree Bits are 3031

Number of Agree Bits are 2972

Significance Level 5%

Pass Mark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

The Computed  $\chi^2$  is 0.580360120040013

AUTOCORRELATION TEST PASSED

**A (6)**

Sequence Length is 6003 Bits

Number of Agree Bits are 2955

Number of Agree Bits are 3048

Significance Level 5%

Pass Mark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

The Computed  $\chi^2$  is 1.44222111055528

AUTOCORRELATION TEST PASSED

**A (7)**

Sequence Length is 6003 Bits

Number of Agree Bits are 3019

Number of Agree Bits are 2984

Significance Level 5%

Pass Mark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

The Computed Chi<sup>2</sup> is 0.204302868579053

AUTOCORRELATION TEST PASSED

**A (8)**

Sequence Length is 6003 Bits

Number of Agree Bits are 3007

Number of Agree Bits are 2996

Significance Level 5%

Pass Mark chi<sup>2</sup> is 3.4980125 with 1 Degree of Freedom

The Computed Chi<sup>2</sup> is 0.0201834862385321

AUTOCORRELATION TEST PASSED

---

## **The Twin Algorithm Testing**

A sequence of length 16000 bits is generated with an arbitrary seed key (in this test the used seed key="GGGGGGGGGG") is tested the result of the test is shown in the following:

Sequence Length is 5003 bits

Significance Level 5%

FREQUENCY TEST

Pass Mark chi<sup>2</sup> is 3.4980125 with 1 Degree of Freedom

The Computed Chi<sup>2</sup> is 0.105042804474161

FREQUENCY TEST PASSED

SERIAL TEST

Pass Mark chi<sup>2</sup> is 7.53134432298715 with 3 Degree of Freedom

The Computed Chi<sup>2</sup> is 4.79577552805899

SERIAL TEST PASSED

#### POKER TEST

Pass Mark chi<sup>2</sup> is 15.2240701045112 with 8 Degree of Freedom

The Computed Chi<sup>2</sup> is 7.14742857142857

POKER TEST PASSED

#### RUN TEST

Pass Mark chi<sup>2</sup> for Run 0's is 24.7116086077363 with 15 Degree of Freedom

The Computed Chi<sup>2</sup> for Run 0's is 30.2674074395257

RUN TEST for RUN 0's FAILD

Pass Mark chi<sup>2</sup> for Run 1's is 20.7421553558494 with 12 Degree of Freedom

The Computed Chi<sup>2</sup> for Run 1's is 7.55263513327762

RUN TEST for RUN 1's PASSED

#### AUTOCORRELATION TEST

Pass Mark chi<sup>2</sup> is 3.4980125 with 1 Degree of Freedom

A (1): The Computed Chi<sup>2</sup> is 0.3302087290951 PASSED

A (2): The Computed Chi<sup>2</sup> is 0.564027248296981 PASSED

A (3): The Computed Chi<sup>2</sup> is 0.1155625 PASSED

A (4): The Computed Chi<sup>2</sup> is 0.189074317144822PASSED

A (5): The Computed Chi<sup>2</sup> is 0.105075634454307 PASSED

A (6): The Computed Chi<sup>2</sup> is 0.175595424142027 PASSED

A (7): The Computed Chi<sup>2</sup> is 0.0225681420355089 PASSED

A (8): The Computed Chi<sup>2</sup> is 0.517724288840263 PASSED

#### FREQUENCY TEST

Sequence Length is 16003 bits

Number of 0's are 8022

Number of 1's are 7981

Significance Level 5%

Pass Mark chi<sup>2</sup> is 3.4980125 with 1 Degree of Freedom

The Computed Chi<sup>2</sup> is 0.105042804474161



**FREQUENCY TEST PASSED**

---

**SERIAL TEST**

Sequence Length is 16002 Bits

Numbers of blocks are 8001 Pairs

Number of 00 Blocks are 1996

Number of 01 Blocks are 2080

Number of 10 Blocks are 1949

Number of 11 Blocks are 1976

Significance Level 5%

Pass Mark  $\chi^2$  is 7.53134432298715 with 3 Degree of Freedom

The Computed  $\chi^2$  is 4.79577552805899

**SERIAL TEST PASSED**

---

**POKER TEST**

Sequence Length is 16000 Bits

Block Length is 8 bits

Numbers of blocks are 2000 Groups

Number of Blocks has 0 1's are 13

Number of Blocks has 1 1's are 62

Number of Blocks has 2 1's are 210

Number of Blocks has 3 1's are 445

Number of Blocks has 4 1's are 535

Number of Blocks has 5 1's are 462

Number of Blocks has 6 1's are 203

Number of Blocks has 7 1's are 64

Number of Blocks has 8 1's are 6

Significance Level 5%

Pass Mark  $\chi^2$  is 15.2240701045112 with 8 Degree of Freedom

The Computed  $\chi^2$  is 7.14742857142857

POKER TEST PASSED

---

### **RUN TEST**

Sequence Length is 16003 Bits

RUN 0 of length 1 = 2029

RUN 0 of length 2 = 1009

RUN 0 of length 3 = 508

RUN 0 of length 4 = 210

RUN 0 of length 5 = 136

RUN 0 of length 6 = 54

RUN 0 of length 7 = 34

RUN 0 of length 8 = 19

RUN 0 of length 9 = 10

RUN 0 of length 10 = 4

RUN 0 of length 11 = 2

RUN 0 of length 12 = 3

RUN 0 of length 13 = 1

RUN 0 of length 16 = 1

RUN 1 of length 1 = 2033

RUN 1 of length 2 = 987

RUN 1 of length 3 = 516

RUN 1 of length 4 = 244

RUN 1 of length 5 = 111

RUN 1 of length 6 = 59

RUN 1 of length 7 = 39

RUN 1 of length 8 = 16

RUN 1 of length 9 = 6

RUN 1 of length 10 = 4

RUN 1 of length 11 = 3

RUN 1 of length 13 = 1

Significance Level 5%

Pass Mark  $\chi^2$  for Run 0's is 24.7116086077363 with 15 Degree of Freedom

The Computed  $\chi^2$  for Run 0's is 30.2674074395257

RUN TEST for RUN 0's FAILED

Pass Mark  $\chi^2$  for Run 1's is 20.7421553558494 with 12 Degree of Freedom

The Computed  $\chi^2$  for Run 1's is 7.55263513327762

RUN TEST for RUN 1's PASSED

RUN 1's PASSED

---

**AUTOCORRELATION TEST****A (1)**

Sequence Length is 16003 Bits

Number of Agree Bits are 7965

Number of Agree Bits are 8038

Significance Level 5%

Pass Mark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

The Computed  $\chi^2$  is 0.333020872390951

**AUTOCORRELATION TEST PASSED**

**A (2)**

Sequence Length is 16003 Bits

Number of Agree Bits are 8049

Number of Agree Bits are 7954

Significance Level 5%

Pass Mark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

The Computed  $\chi^2$  is 0.564027248296981

**AUTOCORRELATION TEST PASSED**

**A (3)**

Sequence Length is 16003 Bits

Number of Agree Bits are 8023

Number of Agree Bits are 7980

Significance Level 5%

Pass Mark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

The Computed  $\chi^2$  is 0.1155625

AUTOCORRELATION TEST PASSED

**A (4)**

Sequence Length is 16003 Bits

Number of Agree Bits are 8029

Number of Agree Bits are 7974

Significance Level 5%

Pass Mark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

The Computed  $\chi^2$  is 0.189074317144822

AUTOCORRELATION TEST PASSED

**A (5)**

Sequence Length is 16003 Bits

Number of Agree Bits are 7981

Number of Agree Bits are 8022

Significance Level 5%

Pass Mark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

The Computed  $\chi^2$  is 0.105075634454307

AUTOCORRELATION TEST PASSED

**A (6)**

Sequence Length is 16003 Bits

Number of Agree Bits are 7975

Number of Agree Bits are 8028

Significance Level 5%

Pass Mark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

The Computed  $\chi^2$  is 0.175595424142027

AUTOCORRELATION TEST PASSED

A (7)

Sequence Length is 16003 Bits

Number of Agree Bits are 8011

Number of Agree Bits are 7992

Significance Level 5%

Pass Mark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

The Computed  $\chi^2$  is 0.0225681420355089

AUTOCORRELATION TEST PASSED

A (8)

Sequence Length is 16003 Bits

Number of Agree Bits are 8047

Number of Agree Bits are 7956

Significance Level 5%

Pass Mark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

The Computed  $\chi^2$  is 0.517724288840263

AUTOCORRELATION TEST PASSED

---

FREQUENCY TEST

Sequence Length is 400 bits

Number of 0's are 204

Number of 1's are 196

Significance Level 5%

PassMark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom

The Computed  $\chi^2$  is 0.16

FREQUENCY TEST PASSED

=====

SERIAL TEST

Sequence Length is 400 Bits

Number of blocks are 200 Pairs

Number of 00 Blocks are 55

Number of 01 Blocks are 44

Number of 10 Blocks are 50

Number of 11 Blocks are 51

Significance Level 5%

PassMark  $\chi^2$  is 7.53134432298715 with 3 Degree of Freedom

The Computed  $\chi^2$  is 1.24

SERIAL TEST PASSED

=====

POKER TEST

Sequence Length is 400 Bits

Block Length is 8 bits

Number of blocks are 50 Groups

Number of Blocks has 0 1's are 2

Number of Blocks has 1 1's are 1

Number of Blocks has 2 1's are 6

Number of Blocks has 3 1's are 10

Number of Blocks has 4 1's are 14

Number of Blocks has 5 1's are 10

Number of Blocks has 6 1's are 4

Number of Blocks has 7 1's are 1

Number of Blocks has 8 1's are 2

Significance Level 5%

PassMark  $\chi^2$  is 15.2240701045112 with 8 Degree of Freedom

The Computed  $\chi^2$  is 34.3702857142857

POKER TEST FAILD

=====

RUN TEST

Sequence Length is 400 Bits

RUN 0 of length 1 = 41  
RUN 0 of length 2 = 29  
RUN 0 of length 3 = 14  
RUN 0 of length 4 = 4  
RUN 0 of length 5 = 1  
RUN 0 of length 6 = 1  
RUN 0 of length 7 = 1  
RUN 0 of length 9 = 1  
RUN 0 of length 10 = 2

RUN 1 of length 1 = 49  
RUN 1 of length 2 = 18  
RUN 1 of length 3 = 14  
RUN 1 of length 4 = 6  
RUN 1 of length 5 = 2  
RUN 1 of length 7 = 2  
RUN 1 of length 8 = 1  
RUN 1 of length 13 = 1

Significance Level 5%

PassMark  $\chi^2$  for Run 0's is 16.6355212541411 with 9 Degree of Freedom  
The Computed  $\chi^2$  fo Run 0's is 45.72234375

RUN TEST for RUN 0's FAILD

ii

PassMark  $\chi^2$  for Run 1's is 20.7421553558494 with 12 Degree of Freedom  
The Computed  $\chi^2$  fo Run 1's is 87.28779296875

RUN TEST for RUN 1's FAILD

=====

AUTOCORRELATION TEST

===== A( 1===== (

Sequence Length is 400 Bits  
Number of Agree Bits are 214  
Number of Agree Bits are 186

Significance Level 5%

PassMark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom  
The Computed  $\chi^2$  is 1.96491228070175

AUTOCORRELATION TEST PASSED

+++++



=====

A( 2===== (

Sequence Length is 400 Bits  
Number of Agree Bits are 206  
Number of Agree Bits are 194

Significance Level 5%

PassMark chi^2 is 3.4980125 with 1 Degree of Freedom  
The Computed Chi^2 is 0.361809045226131

AUTOCORRELATION TEST PASSED

+++++

=====

A( 3===== (

Sequence Length is 400 Bits  
Number of Agree Bits are 206  
Number of Agree Bits are 194

Significance Level 5%

PassMark chi^2 is 3.4980125 with 1 Degree of Freedom  
The Computed Chi^2 is 0.36272040302267

AUTOCORRELATION TEST PASSED

+++++

=====

A( 4===== (

Sequence Length is 400 Bits  
Number of Agree Bits are 204  
Number of Agree Bits are 196

Significance Level 5%

PassMark chi^2 is 3.4980125 with 1 Degree of Freedom  
The Computed Chi^2 is 0.161616161616162

AUTOCORRELATION TEST PASSED

+++++

=====

A( 5===== (

Sequence Length is 400 Bits  
Number of Agree Bits are 202  
Number of Agree Bits are 198

Significance Level 5%

PassMark chi^2 is 3.4980125 with 1 Degree of Freedom  
The Computed Chi^2 is 0.0405063291139241

## AUTOCORRELATION TEST PASSED

+++++

=====

A( 6===== (

Sequence Length is 400 Bits  
 Number of Agree Bits are 198  
 Number of Agree Bits are 202

Significance Level 5%

PassMark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom  
 The Computed  $\chi^2$  is 0.0406091370558376

## AUTOCORRELATION TEST PASSED

+++++

=====

A( 7===== (

Sequence Length is 400 Bits  
 Number of Agree Bits are 214  
 Number of Agree Bits are 186

Significance Level 5%

PassMark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom  
 The Computed  $\chi^2$  is 1.99491094147583

## AUTOCORRELATION TEST PASSED

+++++

=====

A( 8===== (

Sequence Length is 400 Bits  
 Number of Agree Bits are 204  
 Number of Agree Bits are 196

Significance Level 5%

PassMark  $\chi^2$  is 3.4980125 with 1 Degree of Freedom  
 The Computed  $\chi^2$  is 0.163265306122449

## AUTOCORRELATION TEST PASSED

+++++