On The Nonlinear Key Generator Design Using Unit-Step and Trace Functions

Raghad Kadhim Salih*

Received on: 11/2/ 2009
Accepted on:4/6/ 2009

Abstract

The paper presents a proposed method with an algorithm which has been written in Matlab language for designing a nonlinear key generator, which is denoted by (US-TR), using unit-step function and trace function from Galois field of order 2^N (i.e. GF(2^N), ($N \geq 2$)) to Galois field of order 2 (GF(2)). The proposed generator produces a binary sequence of period $(2^N - 1)$ where N is a composite number, with high degree of complexity and good randomness properties. The advantage of the new nonlinear generator is the output sequence which has high degree of complexity to increase the security of this generator concerning the designed feature that limit the ability of anti-jammer when it uses as a key in cipher systems or in spread spectrum digital communication system. This paper has useful properties of the trace function. Moreover, Illustrative examples are given for determining the output sequence with its complexity of the proposed generator and good results are obtained.

Keywords: Nonlinear generator, Unit-Step function, Trace function and Complexity.

تصميم مولـــد مفتـــاح لاخطـــي باســـتخدام دوال الوحـــدة والأثـــر

الخلاصة

يقدم البـــحث طريقة مقترحة مع خوارزميـــة كتبت بلغة Matlab لتصميم مولد مفتـــاح غيرخطي يرمـــز لـــه $(US-TR)$ باســـتخدام دالـــة الوحـــدة (Unit-step) ودالـــة الأثـــــر (trace function). المولـــد اللاخطـــي المقتـــرح يولـــد متتابعـــة ثنائيـــة ذات طـــول دورة $(2^N - 1)$حيث N عدد مركب ودرجة تعقيد عالية مع خصائص عشوائية جيدة نسبة لكونه مولد غير خطي . مزايا و كفاءة هذا المولد اللاخطي الجديد تعتمد على درجـــة تعـــقـــــيد متتابعتـــه العالية لزيادة أمنيته و قدرته على مواجهة التداخل و التشويش عند استخدامه كمفتاح في انظمـــة التشفير و في أنظمة الاتصالات الرقمية كانظمة الطيف المنتشر. كما تم ذكر الخصائص المهمة لدالة الأثر. علاوة على ذلك تم إعطاء بعض الأمثلة التوضيحية التي تبين كفاءة المولـــد الجديـــد عن طريق تحديد متتابعته الثنائية الدورية اللاخطية ودرجة تعقيدها و قد تم الحصول على نتائج جيدة .

* Applied Sciences Department, University of Technology/Baghdad

https://doi.org/10.30684/etj.27.15.4
2412-0758/University of Technology-Iraq, Baghdad, Iraq

1. Introduction

A pseudo-noise (PN) generator is a mechanism for generating a PN-sequence of binary or real digits. Pseudo-Noise generators generate (PN) sequences which are used as spectrum-spreading modulations for direct sequence spread spectrum design for digital communication system and as a key in cipher systems [1,2]. The resulting sequence is called pseudo-noise sequence since it is periodic and there is no algorithm using a finite state machine which can produce a truly random sequence [2]. PN-sequences are characterized by three properties; namely: period, complexity and randomness. These properties defined the measure of security for the sequences [3,4]. The complexity of the sequence is one of important properties for security of the information from unauthorized person [2,5].

In 1999 Wes Horner and Wei Chien [6] developed a pseudo-noise code division multiple access (CDMA) spread spectrum , where they inserted an external signal into the channel to evaluate the performance of the system and to make a signal more immune to jamming, in 2006 Laura J. Riely [7] presented the implementations of binary PN Reed-Solomon codes in spread spectrum systems, where she demonstrated all major functions of the spread spectrum modulation technique to perform frequency hop transmission and synchronization and in 2005 Maurice L. Schiff [8] demonstrated how to build and test up to (16) bits of nonlinear PN generator with illustration figures of this PN-generator.

In this work, the proposed generator is a nonlinear PN generator produces a binary sequence of period $(2^N - 1)$ where N is a composite number with high degree of complexity and good randomness properties. The idea of this paper is to increase the complexity of the output sequence of the proposed generator to have enough ability for the security of the information from interceptor and jammer.

2. Galois Field Arithmetic [1,5,9] :

Definition (1):

A finite field F is called a Galois field denoted by GF(q^m) if the number of elements of F is q^m (i.e. the order of F is q^m), where q is a prime number and m is a natural number. q is called the characteristic of the field GF(q^m) .

Definition (2):

It is known that every nonzero element α of GF(q^m) which satisfies the equation :

$$a^{q^m - 1} = 1 \qquad \dots (1)$$

is said to be a primitive element of GF(q^m) if all the powers of α less than (q^m-1) are different . Thus, if α is a primitive element then $\alpha^i \neq 1$, for $0 < i < q^m-1$.

Definition (3) :

The polynomial m_α(z) over GF(q) of minimum degree with respect to all polynomials over GF(q) having a field element α of GF(q^m) as a root, is called the minimum polynomial m_α(z) of α over GF(q), where GF(q) is Galois field of order q.

Example:

Consider the field GF(2^4) obtained by taking the polynomial $m_\alpha(z) = z^4+z+1$ over GF(2) as the modulo polynomial. The powers of α were reduced to polynomials of degree (3) or less in α. Table (1) shows the representation elements of GF(16).

3. Unit-Step Function [10,11]:

The unit step function is defined by:

$$u(k) = \begin{cases} 0 & for \quad k < 0 \\ 1 & for \quad k \ge 0 \end{cases} \quad \ldots (2)$$

It is seen to be a sequence of numbers which is everywhere zero for negative discrete time and everywhere one for nonnegative discrete time. Figure (1) shows a plot of the unit-step function.

4. Trace Function [1,5]:

A fundamental mathematical tool used in investigation of PN generator is a linear mapping from a finite field onto a subfield. This mapping is called the "Trace function" or the "Trace polynomial" denoted by $Tr_q^{q^n}(a)$ where $a \in GF(q^n)$.

The Trace function from $GF(q^n)$ to GF(q) where (q>1) and (n≥2) is defined as [1]:

$$Tr_q^{q^n}(a) = \sum_{i=0}^{n-1} (a)^{q^i} \quad \ldots (3)$$

where $a \in GF(q^n)$.

Example :

In GF(16) represented in table (1), the trace values of α^3 and α^5 are :

$Tr_2^{16}(\alpha^3) = (\alpha^3) + (\alpha^3)^2 + (\alpha^3)^4 + (\alpha^3)^8$
$= \alpha^3 + \alpha^6 + \alpha^{12} + \alpha^9$
$= \alpha^3 + (\alpha^2 + \alpha^3) + (1+ \alpha + \alpha^2 + \alpha^3) + (\alpha + \alpha^3) = 1.$

(See table (1)).

$Tr_2^{16}(\alpha^5) = (\alpha^5) + (\alpha^5)^2 + (\alpha^5)^4 + (\alpha^5)^8$
$= \alpha^5 + \alpha^{10} + \alpha^5 + \alpha^{10} = 0$,

where α^3 and $\alpha^5 \in GF(16)$ and the arithmetic operations over GF(2).

The useful properties of the trace function are:

<u>Property 1</u>: When α is in $GF(q^n)$, $Tr_q^{q^n}(a)$ has values in GF(q) [1].

<u>Property 2:</u> Conjugate field elements which have the same trace value [1,5].

5. The Complexity of a Periodic Sequence :

The complexity of the sequence (or generator) in the cipher and communication systems is the *length of the minimum linear feedback shift register* (LFSR) that can generate the sequence [9,12]. We can characterize the LFSR of length (n) by the characteristic polynomial *f*(*x*) as:

$$f(x) = c_0 + c_1 x + c_2 x^2 + \mathbf{L} + c_{n-1} x^{n-1} + x^n$$

where $c_0, c_1, \ldots, c_{n-1}$ are 0 or 1 [2,3].

So the complexity of the sequence is the degree of the minimal characteristic polynomial that can generate the given sequence. If the entire sequence is known, then the complexity can be determined. We show that, for a sequence with a known complexity (L), the entire sequence is given when 2L consecutive bits are known, where $(2L-1)$ consecutive bits are not enough to determine the sequence uniquely. We need (2L) consecutive bits to deduce the entire sequence since if (2L) consecutive bits are given, then we can write a system of L-equations in the L unknown variables and find its unique solution. This gives the characteristic polynomial of the minimal LFSR that can generate the given sequence. Hence for acceptable security, we need to have a sequence with high complexity [2,5,13].

The *complexity* of the sequence can be determined as follows [3,5,7]:

Let S_i be a binary periodic sequence with period (p) and S(x) be a period polynomial over GF(2) , where:

$$S_i = s_0, s_1, s_2, s_3, \ldots, s_{p-1} ,$$
$$s_i = s_{p+i} , p > 0 , i = 0,1,\ldots, p-1 \quad \ldots (4)$$

$$S(x) = s_0 + s_1 x + s_2 x^2 + \ldots + s_{p-1} x^{p-1}. \quad \ldots (5)$$

Then the complexity of the sequence can be determined as:

$$f^*(x) = \frac{x^p + 1}{\gcd\left(x^p + 1, S(x)\right)}$$
$$= c_n + c_{n-1}x + \ldots + c_1 x^{n-1} + c_0 x^n. \quad \ldots (6)$$

where $\gcd\left(x^p + 1, S(x)\right)$ is the greatest common divisor of $(x^p + 1)$ and $S(x)$, $f^*(x)$ is the reciprocal function of $f(x)$ and $f(x)$ is:

$$f(x) = x^n f^*(\frac{1}{x})$$
$$= c_0 + c_1 x + \ldots + c_{n-1}x^{n-1} + c_n x^n \quad . \quad \ldots (7)$$

where n is the degree of the polynomial $f^*(x)$ and $c_0, c_1, \ldots, c_n$ are the coefficients of $f^*(x)$.

Now, the complexity L can be determined as:

$$L = \deg(f(x)) \quad \ldots (8)$$

where $\deg(f(x))$ is the degree of the characteristic polynomial $f(x)$ and $f(x)$ is the characteristic polynomial of the minimal LFSR that can generate the sequence S_i.

The following algorithm summarizes the steps for the above procedure for determining the ***Complexity*** (L) of the periodic sequence S_i.

<u>Complexity Algorithm:</u>

<u>Step 1:</u>

Input the sequence (S_i) over GF(2) of period (p) ,p>0 and then use S_i to compute the period polynomial S(x) over GF(2) , where:

$$S_i = s_0, s_1, s_2, s_3, \ldots, s_{p-1}$$

and

$$S(x) = s_0 + s_1 x + s_2 x^2 + \ldots + s_{p-1} x^{p-1}.$$

<u>Step 2:</u>

Find the greatest common divisor of the two polynomials $(x^p + 1)$ and $S(x)$ $\left(\gcd(x^p + 1, S(x))\right)$ over GF(2) as follows:

a) Input the two polynomials $(x^p + 1)$ and $S(x)$ where the degree of $x^p + 1$ is greater than $S(x)$.

b) According to the arithmetic operations over GF(2), compute r where r is the remainder of dividing (x^p+1) by $S(x)$ using modulo (2) in operations.

c) If r=0 then :

§ $\gcd\left(x^p+1, S(x)\right) = S(x)$

§ go to (step d)

else

§ Set: $(x^p+1) = S(x)$

$S(x) = r$

§ Go to (step b)

d) End.

Step 3:

Use step (2) to find the reciprocal function $f^*(x)$ of $f(x)$ over GF(2) which is:

$$f^*(x) = \frac{x^p+1}{\gcd\left(x^p+1, S(x)\right)}$$
$$= c_n + c_{n-1}x + \mathbf{L} + c_1x^{n-1} + c_0x^n.$$

Step 4:

Compute the polynomial $f(x)$ from $f^*(x)$ as follows:

$$f(x) = x^n f^*(\frac{1}{x})$$
$$= c_0 + c_1x + \mathbf{L} + c_{n-1}x^{n-1} + c_nx^n$$

where (n) is the degree of the polynomial $f^*(x)$ and $c_0, c_1, \mathbf{K}, c_n$ are the coefficients of $f^*(x)$.

Step 5:

Determine the complexity (L) by:

$$L = \deg(f(x)) = n$$

where $\deg(f(x))$ is the degree of the characteristic polynomial $f(x)$ and $f(x)$ is the characteristic polynomial of the minimal LFSR that can generate the sequence S_i.

Complexity algorithm enables the computation of the complexity accurately for any binary periodic PN sequences produced from linear or nonlinear generators.

Example:

Consider the following sequence over GF(2):

S_i=1011100, where i = 0,1,…,6 and the period p=7 .

By applying complexity algorithm one gets the following results:

The period polynomial S(x) over GF(2) is:

$S(x) = 1 + x^2 + x^3 + x^4$,

$\gcd(x^7+1, S(x)) = x^4 + x^3 + x^2 + 1$,

$$f^*(x) = \frac{x^7+1}{\gcd\left(x^7+1, S(x)\right)}$$
$$= \frac{x^7+1}{x^4+x^3+x^2+1} = x^3 + x^2 + 1$$

,

the minimal characteristic polynomial

$f(x) = x^3 f^*(\frac{1}{x}) = 1 + x + x^3$ and the complexity (L) of the sequence S_i is: $L = \deg(f(x)) = 3$.

6. Proposed Method for Designing a Nonlinear Key Generator Using Unit-Step and Trace Functions:

Unit-step function and trace function from GF(2^N) to GF(2) are proposed for designing a nonlinear generator which produces a pseudo-noise binary sequence of period (2^N – 1), N is a composite integer number, and high complexity with good randomness properties. It is denoted by (US-TR) generator.

Hence, the developing ***US-TR generator*** is defined as :

Let N is a composite integer number, i.e.

$$N = a \times b \qquad \dots (9)$$

Then, *US-TR* sequence generator of period (2^N –1) is defined as :

$$Seq_n = u(n) \oplus_2 \sum_{j=0}^{a-1}\left(\sum_{i=0}^{b-1}(a^n)^{2^{ai}}\right)^{2^{j+1}} \oplus_2 S_n \oplus_2 \mathrm{Re}\,c(S_n)$$

$$,\ n = 0,1,\dots,2^N - 2 \qquad \dots (10)$$

where: $u(n)$ is the unit-step function which is defined in eq.(2), $\oplus_2$ is a modulo 2 addition, $S_n = \mathrm{Tr}_2^{2^N}(a^{n+a+b}) \,|\, \mathrm{Tr}_2^{2^N}(a^{n+b})$ which has period (2^N–1), $\mathrm{Tr}_2^{2^N}(a^{n+a+b}) = \sum_{j=0}^{N-1}(a^{n+a+b})^{2^j}$ and $\mathrm{Tr}_2^{2^N}(a^{n+b}) = \sum_{i=0}^{N-1}(a^{n+b})^{2^i}$, (|) is nonlinear Boolean function "OR" which is defined as (0|0=0, 0|1=1|0=1|1=1), $\mathrm{Re}\,c(S_n)$ is the **<u>rec</u>**iprocal of the sequence (S_n), $n = 0,1,\dots,2^N - 2$ and it is defined as :

$$\mathrm{Re}\,c(S_n) = \mathrm{Re}\,c\left(\mathrm{Tr}_2^{2^N}(a^{n+a+b}) \,|\, \mathrm{Tr}_2^{2^N}(a^{n+b})\right)$$
$$= \mathrm{Re}\,c(S_0, S_1, S_2, \dots, S_{2^N-3}, S_{2^N-2})$$
$$= (S_{2^N-2}, S_{2^N-3}, S_{2^N-4}, \dots, S_1, S_0)$$

, $\alpha^n \in$ GF(2^N), $n = 0,1,\dots,2^N - 2$ and $\mathrm{Tr}_q^{q^d}(a^n)$ is the "Trace function" in eq.(3) from GF(q^d) to GF(q) which is defined as:

$$\mathrm{Tr}_q^{q^d}(a^n) = \sum_{i=0}^{d-1}(a^n)^{q^i}\ .$$

Note that, to find Seq_n in eq.(10) we must find the power-α representation for GF(2^N) elements (i.e. $a^0, a^1, \mathbf{L}, a^{2^N-2}$) using $m_a(z)$ over GF(2) as shown in section (2), where the polynomial $m_\alpha(z)$ is called the minimum polynomial of α over GF(q), (see section (2)).

The following algorithm summarizes the steps for finding the binary sequence of period (2^N-1) of US-TR generator.

<u>*US-TR Algorithm :*</u>

<u>Step 1:</u>

Input :-

(1) The minimum polynomial m_α (z) of the primitive element α of GF(2^N) over GF(2).

(2) The values of N, a and b.

<u>Step 2 :</u>

Find the power-α representation for GF(2^N) elements (i.e. $a^0, a^1, \mathbf{L}, a^{2^N-2}$) using

$m_a(z)$ over GF(2) as shown in section (2).

Step 3 :

For all n=0,1,…,2^N-2 compute:- $u(n)$ as in eq.(2) ,

$$S_n = \mathrm{Tr}_2^{2^N}(a^{n+a+b}) \,|\, \mathrm{Tr}_2^{2^N}(a^{n+b}) = \sum_{j=0}^{N-1}(a^{n+a+b})^{2^j} \,|\, \sum_{i=0}^{N-1}(a^{n+b})^{2^i}$$

and

$$\mathrm{Rec}(S_n) = \mathrm{Rec}\left(\mathrm{Tr}_2^{2^N}(a^{n+a+b}) \,|\, \mathrm{Tr}_2^{2^N}(a^{n+b})\right) = (S_{2^N-2}, S_{2^N-3}, S_{2^N-4}, \ldots, S_1, S_0)$$

Step 4 :

For all n=0,1,…,2^N-2, use (step 3) to evaluate Seq_n in Eq.(10) as follows:

$$Seq_0 = u(0) \oplus_2 \sum_{j=0}^{a-1}\left(\sum_{i=0}^{b-1}(a^0)^{2^{ai}}\right)^{2^{j+1}} \oplus_2 S_0 \oplus_2 S_{2^N-2}$$

$$Seq_1 = u(1) \oplus_2 \sum_{j=0}^{a-1}\left(\sum_{i=0}^{b-1}(a)^{2^{ai}}\right)^{2^{j+1}} \oplus_2 S_1 \oplus_2 S_{2^N-3}$$

$$\vdots$$

$$Seq_{2^N-2} = u(2^N-2) \oplus_2 \sum_{j=0}^{a-1}\left(\sum_{i=0}^{b-1}(a^{2^N-2})^{2^{ai}}\right)^{2^{j+1}} \oplus_2 S_{2^N-2} \oplus_2 S_0$$

7. Illustrative Examples :

Example (1) :

Let $N = 4 = 2\times 2 = a\times b$ and $m_\alpha(z) = z^4 + z + 1$ over GF(2) where $\alpha \in GF(2^4)$, then US-TR sequence generator in eq.(10) of period $(2^4 - 1)$ is defined as :

$$Seq_n = u(n) \oplus_2 \sum_{j=0}^{1}\left(\sum_{i=0}^{1}(a^n)^{2^{2i}}\right)^{2^{j+1}} \oplus_2 S_n \oplus_2 \mathrm{Re}\,c(S_n) \quad \ldots (11)$$

where : $u(n)$ is the unit-step function which is defined in eq.(2),

$$S_n = \mathrm{Tr}_2^{2^4}(a^{n+4}) \,|\, \mathrm{Tr}_2^{2^4}(a^{n+2}) = \sum_{j=0}^{3}(a^{n+4})^{2^j} \,|\, \sum_{i=0}^{3}(a^{n+2})^{2^i} ,$$

$$\mathrm{Rec}(S_n) = \mathrm{Rec}\left(\mathrm{Tr}_2^{2^4}(a^{n+4}) \,|\, \mathrm{Tr}_2^{2^4}(a^{n+2})\right) = (S_{14}, S_{13}, S_{12}, \ldots, S_1, S_0),$$

n = 0, 1,…, 14 and $\alpha \in GF(2^4)$.

Table (2) presents the output sequence of US-TR generator in eq.(11) with its period by applying US-TR algorithm.

The degree of complexity of the sequence of period (15) of US-TR generator is computed in table(3) by applying complexity algorithm which is illustrated in section (5).

It is obvious from table (3) that the complexity of US-TR sequence (or generator) is too high (13), since US-TR generator produces a sequence of period (15). So, this generator has enough ability for the security of the confidential information in such a way that its meaning is unintelligible to an unauthorized person from interceptor and jammer, since we need (26) consecutive bits to find the entire sequence and this gives the advantage of the US-TR generator (see the definition of the complexity in section (5)).

Example (2) :

Let $N = 6 = 3\times 2 = a\times b$ and $m_\alpha(z)=z^6+z^5+z^2+z+1$ over GF(2) where $\alpha\in GF(2^6)$, then US-TR sequence generator in eq.(10) of period $(2^6 - 1)$ is defined as :

$$Seq_n = u(n) \oplus_2 \sum_{j=0}^{2}\left(\sum_{i=0}^{1}(a^n)^{2^{3i}}\right)^{2^{j+1}} \oplus_2 S_n \oplus_2 \mathrm{Re}c(S_n) \qquad \ldots (12)$$

where : $u(n)$ is the unit-step function which is defined in eq.(2),

$$S_n = \mathrm{Tr}_2^{2^6}(a^{n+5}) \,|\, \mathrm{Tr}_2^{2^6}(a^{n+2}) = \sum_{j=0}^{5}(a^{n+5})^{2^j} \,|\, \sum_{i=0}^{5}(a^{n+2})^{2^i} ,$$

$$\mathrm{Re}c(S_n) = \mathrm{Re}c\left(\mathrm{Tr}_2^{2^6}(a^{n+5}) \,|\, \mathrm{Tr}_2^{2^6}(a^{n+2})\right) = (S_{62}, S_{61}, S_{60}, \ldots, S_1, S_0)$$

, n = 0, 1,…, 62 and $\alpha \in GF(2^6)$.

Table (4) presents the output sequence of US-TR generator in eq.(12) by applying US-TR algorithm.

The degree of complexity of the sequence of period (63) of US-TR generator is computed in table (5) by applying complexity algorithm which is illustrated in section(5).

It is obvious from table (5) that the complexity of US-TR sequence (or generator) is too high (43), since US-TR generator produces a sequence of period (63). So, this generator has enough ability for the security of the secret data in such a way that its meaning is unintelligible to an unauthorized person from interceptor and jammer, since we need (86) consecutive bits to find the entire sequence and this gives the advantage of the US-TR generator (see the definition of the complexity in section (5)).

Example (3) :

Let $N = 8 = 4\times 2 = a\times b$ and $m_\alpha(z) = z^8 + z^6 + z^5 + z + 1$ over GF(2) where $\alpha\in GF(2^8)$, then US-TR sequence generator in eq.(10) of period $(2^8 - 1)$ is defined as :

$$Seq_n = u(n) \oplus_2 \sum_{j=0}^{3}\left(\sum_{i=0}^{1}(a^n)^{2^{4i}}\right)^{2^{j+1}} \oplus_2 S_n \oplus_2 \mathrm{Re}c(S_n) \qquad \ldots (13)$$

where : $u(n)$ is the unit-step function which is defined in eq.(2),

$$S_n = \mathrm{Tr}_2^{2^8}(a^{n+6}) \,|\, \mathrm{Tr}_2^{2^8}(a^{n+2}) = \sum_{j=0}^{7}(a^{n+6})^{2^j} \,|\, \sum_{i=0}^{7}(a^{n+2})^{2^i}$$

,

$$\mathrm{Re}c(S_n) = \mathrm{Re}c\left(\mathrm{Tr}_2^{2^8}(a^{n+6}) \,|\, \mathrm{Tr}_2^{2^8}(a^{n+2})\right) = (S_{254}, S_{253}, S_{252}, \ldots, S_1, S_0)$$

, n = 0, 1,…, 254 and $\alpha \in GF(2^8)$.

Table (6) presents the output sequence of US-TR generator in eq.(13) by applying US-TR algorithm.

The degree of complexity of the sequence of period (255) of US-TR generator is computed in table (7) by applying complexity algorithm which is illustrated in section(5).

It is obvious from table (7) that the complexity of US-TR sequence is too high (73). So, this generator has enough ability for the security of secret information from interceptor and jammer, since we need (146) consecutive bits to find the entire sequence and this gives the advantage of the US-TR generator.

Conclusions

The cipher and communication systems depend on the degree of the security of the key generators to give an acceptable protection from interceptor and jammer to the confidential information from an unauthorized person. So, the paper presents a proposed method for designing a nonlinear key generator, which is denoted by (US-TR), using unit-step function and trace function from $GF(2^N)$ to GF(2), ($N\geq 2$) to produce a binary sequence of period (2^N-1) with high degree of complexity and good randomness properties. From examples (1), (2), (3) and (4) the following results are listed:

- The US-TR generator produces a binary sequence of period (2^N-1); N is a composite number; with good random properties.
- Unit-step and trace functions give an accuracy and consistent to the output sequence of US-TR generator.
- The advantage of the new nonlinear US-TR generator is the complexity of its output sequence which has high degree of complexity to increase the security of this generator from interceptor and concerning the designed feature that limit the ability of anti-jammer

References

[1] Marvin K.S., Jim K.O. , Robert A. S. and Barry K.L., Spread Spectrum Communications, Vol.1, John Wiley & Sons, Inc., USA, 1985.

[2] Wikd P., Linear Feedback Shift Registers, www.sss-mag.com/ pdf/ lfsr.pdf , 2004.

[3] Baker , H.J. and Piper, F.C., Cipher Systems: The Protection of Communications, Northwood Publications, London, 1982.

[4] Naoki, S. and Mitsutoshi, H., Modulatable Orthogonal Sequences and Their Application to SSMA Systems, IEEE Trans. on Inf. Theory, Vol.34, No.1, p.93-95, January 1989.

[5] Salih, R.K., Analysis of Pseudo-Noise Generator Designs For Communication Systems Using State-Space Method, M.Sc. Thesis, University of Technology, IRAQ, 2004.

[6] Wes Horner and Wei Chien, Spread Spectrum CDMA PN-Generators Circuits,

www.newwaveinstruments.com/resources/sections /circuit _diagrams.htm - 58k , 1999.
[7] Riely, L.J., PN-Generators, www.et.fhmerseburg.de/labore/ictech /pn.htm-1k, November 19, 2006.
[8] Shiff, M.L., Pseudo-Noise Codes from Spread Spectrum Scene Online, www.sssmag.com/png2.html 13k ,2005 .
[9] Baker, J.M. and Hughs, P.M., Communications Speech and Vision, Jr. Proc.I, Vol.1, IPIDDG 136, P. 115-119, 1989.
[10] James, A.C., Discrete-Time Systems, Englewood Cliffs, Prentice Hall, Inc., N.J., 1973.
[11] Amir Karniel and Gideon F. Inbar, Nonlinear System Description, www.visl.technion.ac.il/karniel/pub/karnielInbar/ModernTechNs/ch21.pdf., 2006.
[12] Massey, J.L., Shift Register Synthesis and BCH Decoding, IEEE Trans. on Information Theory, Vol.IT-15, No.1, P.140-146, 1969.
[13] Viterbi Vdrew, Principles of Spread Spectrum, 2nd edition, Addison-Wesley publishing, Inc., 1995.

Table (1) The representation elements of GF(2^4)

Representations			
Power of α	***Polynomial in α***	***Power of α***	***Polynomial in α***
α^0	1	α^8	$1+\alpha^2$
α^1	α	α^9	$\alpha+\alpha^3$
α^2	α^2	α^{10}	$1+\alpha+\alpha^2$
α^3	α^3	α^{11}	$\alpha+\alpha^2+\alpha^3$
α^4	$1+\alpha$	α^{12}	$1+\alpha+\alpha^2+\alpha^3$
α^5	$\alpha+\alpha^2$	α^{13}	$1+\alpha^2+\alpha^3$
α^6	$\alpha^2+\alpha^3$	α^{14}	$1+\alpha^3$
α^7	$1+\alpha+\alpha^3$		

Table (2) The output sequence with its period of US-TR generator for Ex.(1).

Generators	*Output Sequences*	*Period*
$u(n) \oplus_2 \sum_{j=0}^{1} \left(\sum_{i=0}^{1} (a^n)^{2^{2i}} \right)^{2^{j+1}}$	**111011001010000**	**15**
$S_n = \mathrm{Tr}_2^{2^4}(a^{n+4}) \mid \mathrm{Tr}_2^{2^4}(a^{n+2})$	**011111011111101**	**15**
$\mathrm{Rec}\left(\mathrm{Tr}_2^{2^4}(a^{n+4}) \mid \mathrm{Tr}_2^{2^4}(a^{n+2})\right)$	**101111110111110**	**15**
US-TR generator in eq.(11)	**001011100010011**	**15**

Table (3) The complexity of the sequence of US-TR generator for Ex.(1).

US-TR generator in eq.(11)	*Complexity Algorithm*
	The Complexity (L)
$Seq_n = u(n) \oplus_2 \sum_{j=0}^{1} \left(\sum_{i=0}^{1} (a^n)^{2^{2i}} \right)^{2^{j+1}} \oplus_2 S_n \oplus_2 \mathrm{Re}\,c(S_n)$	**13**

Table (4) The output sequence with its period of US-TR generator for Ex.(2).

Generators	**Output Sequences**	**Period**
$u(n) \oplus_2 \sum_{j=0}^{2} \left(\sum_{i=0}^{1} (a^n)^{2^{3i}} \right)^{2^{j+1}}$	**100000010100011100110001001111100001101101010110010111101110100**	**63**
$S_n = \mathrm{Tr}_2^{2^6}(a^{n+5}) \mid \mathrm{Tr}_2^{2^6}(a^{n+2})$	**111111111111101111111011001111111001011110110110101001101101111**	**63**
$\mathrm{Rec}\left(\mathrm{Tr}_2^{2^6}(a^{n+5}) \mid \mathrm{Tr}_2^{2^6}(a^{n+2})\right)$	**111101101100101011011011110100111111100110111111101111111111111**	**63**
US-TR generator in eq.(12)	**100010000111011000010001110100100111010101011111010001111100100**	**63**

Table (5) The complexity of US-TR sequence for Ex.(2).

US-TR generator in eq.(12)	***Complexity Algorithm*** ***The Complexity (L)***
$Seq_n = u(n) \oplus_2 \sum_{j=0}^{2} \left(\sum_{i=0}^{1} (a^n)^{2^{3i}} \right)^{2^{j+1}} \oplus_2 S_n \oplus_2 \mathrm{Re}\,c(S_n)$	**43**

Table (6) The output sequence with its period of US-TR generator of Ex.(3).

Generators	***Output Sequences***	***Period***
$u(n) \oplus_2 \sum_{j=0}^{3} \left(\sum_{i=0}^{1} (a^n)^{2^{4i}} \right)^{2^{j+1}}$	**1110100110010011110000 11…1101110000011111111 0100011100010000**	**255**
$S_n = \mathrm{Tr}_2^{2^8}(a^{n+6}) \mid \mathrm{Tr}_2^{2^8}(a^{n+2})$	**1101101110111111111110 11…1111111111000101110 1111101111111101**	**255**
$\mathrm{Re}\,c\left(\mathrm{Tr}_2^{2^8}(a^{n+6}) \mid \mathrm{Tr}_2^{2^8}(a^{n+2})\right)$	**1011111111011111011101 00…1111101011110111111 1111110111011011**	**255**
US-TR generator in eq.(13)	**1000110111110011010011 00…1101100101100011110 0100000100110110**	**255**

Table (7) The complexity of US-TR sequence of Ex.(3).

US-TR generator in eq.(13)	***Complexity Algorithm*** ***The Complexity (L)***
$Seq_n = u(n) \oplus_2 \sum_{j=0}^{3} \left(\sum_{i=0}^{1} (a^n)^{2^{4i}} \right)^{2^{j+1}} \oplus_2 S_n \oplus_2 \mathrm{Re}\,c(S_n)$	**73**

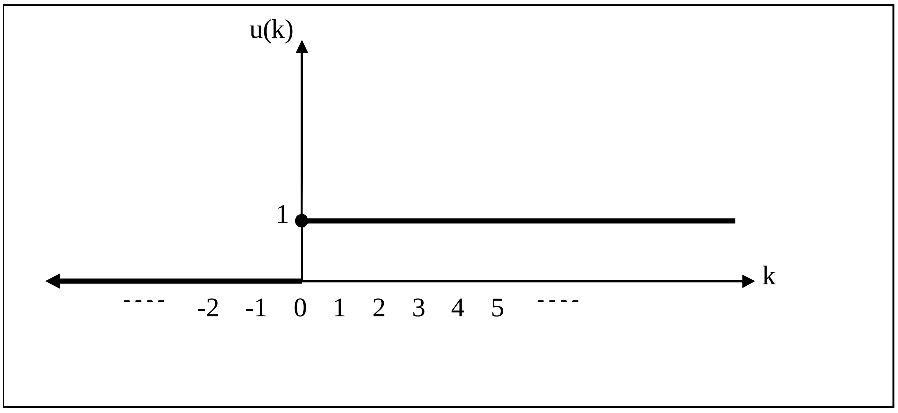

Figure (1)Unit-step function.