Eng. & Technology, Vol.25, No.8, 2007

A Mathematical Development of Gordon, Mills and Welch Generator Using Galois Field and Trace Polynomials

Atheer Jawad Kadhim*
Received on: 24/7 / 2006
Accepted on: 11/ 9/ 2007

Abstract

The paper presents a newly developed method depending on trace polynomial with new algorithm which has been written in Matlab language to modify the Gordon, Mills and Welch generator by increasing the complexity of the output sequence to increase the security of this generator concerning the designed feature that limits the ability of anti-jammer. Good results are obtained compared with Gordon, Mills and Welch generator using Berlekamp-Massey method for determining the complexity of the output sequences. Moreover, this paper has the useful properties of the trace function. Some examples are given of show conciliated the results of this proposed method.

Keywords : Gordon, Mills and Welch Generator, Trace polynomial, Complexity and Sequence.

التطوير الرياضي لمولـــد جوردن - ملز - ويلش باستخدام حـــقل غـــالوا و متـــعددات حدود الأثـــر

الخلاصة

يقدم البـــحث طريقة مطـــورة مع خوارزميـــة جـــديـــدة تعتمد على حـــقل غالـــوا و متـــعددات حـــدود الأثـــر (Trace polynomials) لتطويـــر مولـــد جوردن - ملز - ويلـــش (GMW) اللاخطي عـــن طريـــــق زيادة تعـــقـــيد المتتابعة الناتجة من هذا المولـــد لزيادة أمنيته و قدرته على مواجهة التداخل و التشويش في أنظمة الاتصالات وقـــد حصـــلنا علـــى تعقـــيد جيد بالمقارنة مع مولد الـــ (GMW) من خلال إعطاء بعض الأمثلة التي تبين كفـــاءة المولد الجديد. استخدمت لغة (Matlab) لبرمجة هذه الطريقة كما تم ذكر الخصائص المهمـــة لدالة الأثر.

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 define the measure of security for the sequences [3,4]. The complexity of the sequence is one of important properties for security of the

* Dept. of Applied Sciences, UOT., Baghdad-IRAQ.

http s://doi.org/10.30684/etj.25.8.5
2412-0758/University of Technology-Iraq, Baghdad, Iraq
This is an open access article under the CC BY 4.0 license http://creativecommons.org/licenses/by/4.0

information from unauthorized person.

Gordon, Mills and Welch generator is one of these PN generators. It produces a binary sequence with good period and randomness properties but with relatively small complexity. The idea of this paper is to increase the complexity of the produced sequence but with remaining the same period. The new generator has a sequence with high complexity and good randomness properties. Hence, it has enough ability for the security of the information from interceptor and jammer.

2. The Complexity of The 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 [5,6]. 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.

For a sequence with a known complexity (L), 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 or generator with high complexity [2,7]. There are several methods to determine the complexity of the sequences like Berlekamp-Massey method which is given in the following section, Z-transform and matrices techniques [7].

3. The Berlekamp-Massey Method

Berlekamp-Massey (BM) technique [3,5,6] uses the description based on the synthesis of a shift register where it is used to determine the complexity and the minimal characteristic polynomial that can generate the given sequence .

Berlekamp-Massey method gives [6,7]:

1. The polynomial C(D) .
2. The linear equivalence or the complexity (L) of the sequence .

BM technique is explained in the following algorithm :

Berlekamp-Massey (BM) Algorithm [5,6,7]:

Step 1:

Input :

(1) The period (n) of the sequence (S).

(2) The digits S_i ,i=0,1,…, n-1 of the sequence (S).

Step 2:

Put C(D) =1, B(D) =1, L=0 , b=1 , x=1 and N=0

Step 3:

If N = n , then stop. Otherwise compute (d) :

$$d = S_N + \sum_{i=1}^{L} c_i S_{N-i}$$

Step 4:

If d = 0 , then x = x +1 , and go to (step 7)

Step 5:

If $d \neq 0$ and 2L > N , then

- $C(D) = C(D) - db^{-1} D^x B(D)$
- x=x+1 and go to (step 7)

Step 6:

If $d \neq 0$ and $2L \leq N$, then

- $T(D) = C(D)$
- $C(D) = C(D) - db^{-1} D^x B(D)$
- $L = N+1-L$
- $B(D) = T(D)$
- $b = d$
- $x = 1$

Step 7:

$N = N+1$

and go to (step 3) .

4. Galois Field Arithmetic [1,7] :

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 :

$$a^{q^m-1} = 1$$

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 $0 < i < q^m-1$.

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).

5. Trace polynomial :

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 polynomial" or the "Trace function" denoted by $Tr_q^{q^n}(a)$ where $a \in GF(q^n)$.

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

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

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

Example :

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

$Tr_2^{16}(\alpha^3) = (\alpha^3)+(\alpha^3)^2+(\alpha^3)^4+(\alpha^3)^8$

$= \alpha^3 + \alpha^6 + \alpha^{12} + \alpha^9$

$= \alpha^3+(\alpha^2+\alpha^3)+(1+\alpha+\alpha^2+\alpha^3)$

+

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	$1+\alpha+\alpha^3$		

$(\alpha+\alpha^3)=1$. (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 :

***Property 1*:** When α is in GF(q^n) , $\mathrm{Tr}_q^{q^n}(a)$ has values in GF(q) .

Proof :

$$[\mathrm{Tr}_q^{q^n}(a)]^q = \left(\sum_{j=0}^{n-1} a^{q^j}\right)^q \quad \text{(Definition)}$$

$$= \sum_{j=0}^{n-1} a^{q^{j+1}} \quad \text{(Since the arithmetic operations over GF(q))}$$

$$= \sum_{j=1}^{n-1} a^{q^j} + a$$

$$Since \quad (a \in GF(q^n) \Rightarrow a^{q^n} = a)$$

$$= \mathrm{Tr}_q^{q^n}(a) \quad \text{(Definition)}$$

Therefore , $\mathrm{Tr}_q^{q^n}(a)$ is a field element whose q-th power equals itself . Only elements of GF(q) have this property .

Property 2 :

Conjugate field elements which have the same trace value.

The proof of this property follows from the definition of Trace polynomial and the previous property.

Property 3 : The Trace is linear : for a,b ∈ GF(q) and $a, b \in GF(q^n)$,

$$\mathrm{Tr}_q^{q^n}(aa+bb) = a\mathrm{Tr}_q^{q^n}(a) + b\mathrm{Tr}_q^{q^n}(b)$$

.

Proof :-

$$\mathrm{Tr}(aa+bb) = \sum_{j=0}^{n-1} (aa+bb)^{q^j}$$

(Definition)

$$= \sum_{j=0}^{n-1} (a^{q^j} a^{q^j} + b^{q^j} b^{q^j})$$

(Since the arithmetic operations over GF(2))

$$= \sum_{j=0}^{n-1} a a^{q^j} + \sum_{j=0}^{n-1} b b^{q^j}$$

$$\text{Since } (a \in GF(q) \Rightarrow a^q = a)$$

$$= a\mathrm{Tr}_q^{q^n}(a) + b\mathrm{Tr}_q^{q^n}(b)$$

(Definition).

6. Gordon, Mills and Welch Generator :

Gordon, Mills and Welch (GMW) [1,8,9] suggested the creation of a binary generator depending on "Trace polynomial". When the integer m is a composite, i.e.

$$m = j \times k \qquad \ldots (2)$$

We define a GMW sequence generator of period (2^m –1) as :

$$b_n = \mathrm{Tr}_2^{2^j}\left([\mathrm{Tr}_{2^j}^{2^m}(a^n)]^r\right) \quad \ldots (3)$$

where n=0,1,2,…,2^m–2, α is a primitive element of GF(2^m) , $\mathrm{Tr}_q^{q^d}(a^n)$ is the "Trace polynomial" from GF(q^d) to GF(q) and r is any integer relatively prime to 2^j-1 (i.e. $\gcd(r, 2^j - 1) = 1$), r in the range 0<r< (2^j -1). Figure (1) shows a GMW generator of period (2^m–1) [10].

The exponent r in Eq.(3) can be written as [1,10]:

$$r = \sum_{i=1}^{w} 2^{t_i} \qquad \ldots (4)$$

where the t_i 's are distinct integers in the range $0 \le t_i < j$ for all i and w is the number of ones in the base-2 representation of r.

Hence,

$$b_n = \mathrm{T}r_2^{2^j}\left(\prod_{i=1}^{w}\left[\mathrm{T}r_{2^j}^{2^m}(a^n)\right]^{2^{t_i}}\right) \quad \ldots (5)$$

Since the inner trace is a sum of elements from a field of characteristic (2) (i.e. the operations over GF(2)), all cross-product terms disappear when the trace is squared and Eq.(5) reduces to:

$$b_n = \mathrm{T}r_2^{2^j}\left(\prod_{i=1}^{w}\sum_{h=0}^{k-1} a^{n\,2^{jh+t_i}}\right)$$

$$= \mathrm{T}r_2^{2^j}\left(\sum_{h_1=0}^{k-1}\mathbf{K}\sum_{h_w=0}^{k-1} a^{nC(k,r)}\right) \quad \ldots (6)$$

where k is defined in Eq.(2) and

$$C(k,r)=\sum_{i=1}^{w} 2^{jh_i+t_i} \quad \ldots (7)$$

When expanding the outer Trace polynomial in Eq.(6) , cross-product terms again must disappear, and the GMW generator becomes :

$$b_n = \sum_{u=0}^{j-1}\sum_{h_1=0}^{k-1}\mathbf{K}\sum_{h_w=0}^{k-1} a^{nC(k,r)2^u} \quad \ldots (8)$$

The GMW generator producing a binary sequence of period (2^m-1) is computed by the following algorithm:

GMW Algorithm:

Step 1 :

Input:-

(1) The minimum polynomial $m_\alpha(z)$ of the primitive element α of GF(2^m) over GF(2).

(2) The values of j, k and r .

Step 2 :

Find the powers-of-α representation for GF(2^m) elements (i.e. $a^0, a^1, \mathbf{L}, a^{2^m-2}$) from $m_a(z)$ over GF(2) .

Step 3 :

For all n=0,1,…,2^m-2, evaluate b_n:

$$b_n = \sum_{u=0}^{j-1}\sum_{h_1=0}^{k-1}\mathbf{K}\sum_{h_w=0}^{k-1} a^{nC(k,r)2^u}$$

where $C(k,r)=\sum_{i=1}^{w} 2^{jh_i+t_i}$

and use (step 2) to find the sequence of GMW generator .

Example :

A GMW sequence of period (63) is:

$$b_n = \mathrm{T}r_2^{2^3}\left(\left[\mathrm{T}r_{2^3}^{2^6}(a^n)\right]^3\right), \quad n=0,1,\ldots,62 \quad \ldots(9)$$

where $m_a(z)=z^6+z^5+z^2+z+1$ over GF(2), m=6= $3\times 2 = j\times k$ and r=3 is chosen .

Applying the GMW algorithm yields the following sequence :

b_n=000001010010011101011101001 011100011001111110010010111001 110100.

7. Proposed Method for Developing The Gordon, Mills and Welch (GMW) Generator Using Galois Field and Trace polynomial :

The GMW in previous section is a nonlinear generator which produces a pseudo-noise sequence of period (2^m –1) over GF(2), where m is a composite integer as well as the degree of the minimum polynomial

m_α (z) of a primitive element α of GF(2^m) over GF(2).

We are developing the GMW generator by using "Trace polynomial" from GF(q^n) to GF(q) to produce a sequence which has the same period but with complexity higher than GMW generator and it is denoted by (DGMW).

Hence, the ***DGMW generator*** is defined as :

$$b_n = \mathrm{Tr}_2^{2^j}([\mathrm{Tr}_{2^j}^{2^m}(a^n)]^r) \oplus_2 S_n \oplus_2 \mathrm{Re}c(S_n), \quad n = 0,1,...,2^m - 2 \qquad \ldots (10)$$

where

$S_n = \mathrm{Tr}_2^{2^m}(a^{n+k}) \mid \mathrm{Tr}_2^{2^m}(a^{n+j+k})$, which has period ($2^m$ –1) , $\mathrm{Re}c(S_n)$ is the reciprocal of the sequence ($S_n = \mathrm{Tr}_2^{2^m}(a^{n+k}) \mid \mathrm{Tr}_2^{2^m}(a^{n+j+k})$) , n=0,1,…,$2^m$-2 and it is defined as :

$$\begin{aligned}\mathrm{Re}c(S_n) &= \mathrm{Re}c\left(\mathrm{Tr}_2^{2^m}(a^{n+k}) \mid \mathrm{Tr}_2^{2^m}(a^{n+j+k})\right)\\ &= \mathrm{Re}c(S_0, S_1, S_2, ..., S_{2^m-3}, S_{2^m-2})\\ &= (S_{2^m-2}, S_{2^m-3}, S_{2^m-4}, ..., S_1, S_0)\end{aligned}$$

, $\alpha^n \in$ GF(2^m) , n = 0, 1, 2,…, 2^m-2 , m = j×k is a composite integer, r is an integer in the range (0< r <2^j-1) relatively prime to 2^j-1 (i.e. $\gcd(r, 2^j - 1) = 1$), $\oplus_2$ is a modulo 2 addition, (|) is "OR" function which is defined as (0|0=0, 0|1=1|0=1|1=1) and $\mathrm{Tr}_q^{q^d}(a^n)$ is the "Trace polynomial" in eq.(1) from GF(q^d) to GF(q) which is defined as:

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

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

DGMW Algorithm :

Step 1:

Input:

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

(2) The values of j, k and r .

Step 2 :

Find the power-α representation for GF(2^m) elements (i.e. $a^0, a^1, \mathbf{L}, a^{2^m-2}$) using $m_a(z)$ over GF(2) as shown in section(4).

Step 3 :

For all n=0,1, …,2^m-2 compute:

$$\mathrm{Tr}_2^{2^j}\left([\mathrm{Tr}_{2^j}^{2^m}(a^n)]^r\right) = \mathrm{Tr}_2^{2^j}\left(\sum_{h_1=0}^{k-1} \mathbf{K} \sum_{h_w=0}^{k-1} a^{nC(k,r)}\right)$$

where $C(k,r) = \sum_{i=1}^{w} 2^{jh_i + t_i}$

(See Eq.(4) and Eq.(7)).

Step 4 :

For all n=0,1,…,2^m-2 compute:

$S_n = \mathrm{Tr}_2^{2^m}(a^{n+k}) \mid \mathrm{Tr}_2^{2^m}(a^{n+j+k})$,

$$\begin{aligned}\mathrm{Re}c(S_n) &= \mathrm{Re}c\left(\mathrm{Tr}_2^{2^m}(a^{n+k}) \mid \mathrm{Tr}_2^{2^m}(a^{n+j+k})\right)\\ &= (S_{2^m-2}, S_{2^m-3}, S_{2^m-4}, ..., S_1, S_0)\end{aligned}$$

where

$$\mathrm{Tr}_2^{2^m}(a^{n+k}) = \sum_{i=0}^{m-1} (a^{n+k})^{2^i}$$

and

$$\mathrm{Tr}_2^{2^m}(a^{n+j+k}) = \sum_{d=0}^{m-1} (a^{n+j+k})^{2^d}$$

Step 5 :

For all n=0,1,…,2^m-2, evaluate b_n in Eq.(10) by using (step 3) and (step 4) as follows:

$$b_0 = \mathrm{Tr}_2^{2^j}([\mathrm{Tr}_{2^j}^{2^m}(a^0)]^r) \oplus_2 S_0 \oplus_2 S_{2^m-2}$$

$$b_1 = \mathrm{Tr}_2^{2^j}([\mathrm{Tr}_{2^j}^{2^m}(a)]^r) \oplus_2 S_1 \oplus_2 S_{2^m-3}$$

$$\vdots$$

$$b_{2^m-2} = \mathrm{Tr}_2^{2^j}([\mathrm{Tr}_{2^j}^{2^m}(a^{2^m-2})]^r) \oplus_2 S_{2^m-2} \oplus_2 S_0$$

where

$$\mathrm{Tr}_2^{2^j}([\mathrm{Tr}_{2^j}^{2^m}(a^n)]^r) = \sum_{v=0}^{j-1}\left(\sum_{i=0}^{k-1}(a^n)^{2^{ji}}\right)^{2^v r}$$

8. Illustrative Examples :

Example (1) :

Consider the following GMW generator :

$$b_n = \mathrm{Tr}_2^{2^4}([\mathrm{Tr}_{2^4}^{2^8}(a^n)]^{11}) \quad \ldots (11)$$

where n = 0, 1,…, 254 , $\alpha \in GF(2^8)$, $m_\alpha(z) = z^8 + z^6 + z^5 + z + 1$ over GF(2), m=8=4×2= j×k and r=11. The GMW generator is developed using Eq.(10) as :

$$b_n = \mathrm{Tr}_2^{2^4}([\mathrm{Tr}_{2^4}^{2^8}(a^n)]^{11}) \oplus_2 S_n \oplus_2 \mathrm{Rec}(S_n) \quad \ldots (12)$$

where :

$$S_n = \mathrm{Tr}_2^{2^8}(a^{n+2}) \mid \mathrm{Tr}_2^{2^8}(a^{n+6}),$$

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

The following table presents the results obtained by applying DGMW algorithm.

Table (2) The results of DGMW generator for Ex.(1).

Generators	Output Sequences	Period
$\mathrm{Tr}_2^{2^4}([\mathrm{Tr}_{2^4}^{2^8}(a^n)]^{11})$	000101100111110100 111110…11111	255
$\mathrm{Tr}_2^{2^8}(a^{n+2}) \mid \mathrm{Tr}_2^{2^8}(a^{n+6})$	110110111011111111 111011…11101	255
$\mathrm{Rec}\left(\mathrm{Tr}_2^{2^8}(a^{n+2}) \mid \mathrm{Tr}_2^{2^8}(a^{n+6})\right)$	101111111101111101 110100…11011	255
DGMW generator in Eq.(12)	011100100001110110 110001…11001	255

The complexity of GMW and DGMW sequences is computed in Table (3) by applying Berlekamp-Massey (BM) algorithm.

Table (3) The complexity of GMW and DGMW sequences.

The Generators	***BM Method***
	The Complexity (L)
GMW in Eq.(11) $\mathrm{Tr}_2^{2^4}([\mathrm{Tr}_{2^4}^{2^8}(a^n)]^{11})$	16
DGMW in Eq.(12) $\mathrm{Tr}_2^{2^4}([\mathrm{Tr}_{2^4}^{2^8}(a^n)]^{11}) \oplus_2 S_n \oplus_2 \mathrm{Rec}(S_n)$	72

It is obvious from Table (3) that the complexity of DGMW sequence (or generator) is higher than GMW sequence. So, the security of DGMW generator is more than GMW, since in DGMW sequence we need (144) consecutive bits to find the entire sequence while in GMW sequence we need only (32) consecutive bits to find the entire sequence (see the definition of the complexity in section(2)).

Example (2) :

Consider the following GMW generator :

$$b_n = \mathrm{Tr}_2^{2^3}([\mathrm{Tr}_{2^3}^{2^6}(a^n)]^3),$$

where n = 0, 1,…, 62 , $\alpha \in GF(2^6)$, $m_\alpha(z) = z^6+z^5+z^2+z+1$ over GF(2), $m=6=3\times2= j\times k$ and r=3. The GMW generator is developed using Eq.(10) as:

$$b_n = \mathrm{Tr}_2^{2^3}([\mathrm{Tr}_{2^3}^{2^6}(a^n)]^3) \oplus_2 S_n \oplus_2 \mathrm{Rec}(S_n) \quad \ldots(13)$$

where:

$$S_n = \mathrm{Tr}_2^{2^6}(a^{n+2}) \mid \mathrm{Tr}_2^{2^6}(a^{n+5}),$$

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

Table (4) presents the results by applying DGMW algorithm.

Table (4) The results of DGMW generator for Ex.(2).

Generators	**Output Sequences**	**Period**
$\mathrm{Tr}_2^{2^3}([\mathrm{Tr}_{2^3}^{2^6}(a^n)]^3)$	000001010010011101011101001011100011001111110010010111001110100	63
$\mathrm{Tr}_2^{2^6}(a^{n+2}) \mid \mathrm{Tr}_2^{2^6}(a^{n+5})$	111111111111101111111011001111111001011110110110101001101101111	63
$\mathrm{Rec}\left(\mathrm{Tr}_2^{2^6}(a^{n+2}) \mid \mathrm{Tr}_2^{2^6}(a^{n+5})\right)$	111101101100101011011011110100111111100110111111110111111111111	63
DGMW generator in Eq.(13)	000011000001011001111101110000100101110111111011010001011100100	63

The complexity of GMW and DGMW sequences is computed in table (5) by applying Berlekamp-Massey (BM) algorithm.

Table (5) The complexity of GMW and DGMW sequences.

The Generators	*The Complexity (L)*
GMW in Eq.(9) $\mathrm{Tr}_2^{2^3}([\mathrm{Tr}_{2^3}^{2^6}(a^n)]^3)$	12
DGMW in Eq.(13) $\mathrm{Tr}_2^{2^3}([\mathrm{Tr}_{2^3}^{2^6}(a^n)]^3) \oplus_2 S_n \oplus_2 \mathrm{Re}c(S_n)$	42

It is obvious from table (5) that the complexity of DGMW sequence (or generator) is higher than GMW sequence. So, the security of DGMW generator is more than GMW.

Example (3) :

Consider the following GMW generator :

$$b_n = \mathrm{Tr}_2^{2^2}([\mathrm{Tr}_{2^2}^{2^4}(a^n)]^2), \quad \ldots(14)$$

where n = 0, 1,…, 14 , $\alpha \in GF(2^4)$, $m_\alpha(z) = z^4 + z + 1$ over GF(2) , $m = 4 = 2 \times 2 = j \times k$ and r =2. The GMW generator was developed using Eq.(10) as :

$$b_n = \mathrm{Tr}_2^{2^2}([\mathrm{Tr}_{2^2}^{2^4}(a^n)]^2) \oplus_2 S_n \oplus_2 \mathrm{Re}c(S_n) \quad \ldots(15)$$

where

$$S_n = \mathrm{Tr}_2^{2^4}(a^{n+2}) \mid \mathrm{Tr}_2^{2^4}(a^{n+4}),$$

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

Table (6) presents the sequences of GMW and DGMW generators by applying their algorithms respectively and shows their complexities by using BM algorithm.

Table (6) The sequences of GMW and DGMW generators with their complexities.

	Generators	*Output Sequences*	*Period*	*Complexity (L)*
1	GMW in Eq.(14) $b_n = \mathrm{Tr}_2^{2^2}([\mathrm{Tr}_{2^2}^{2^4}(a^n)]^2)$	000100110101111	15	4
2	DGMW in Eq.(15) $b_n = \mathrm{Tr}_2^{2^2}([\mathrm{Tr}_{2^2}^{2^4}(a^n)]^2) \oplus_2 S_n \oplus_2 \mathrm{Re}c(S_n)$	110100011101100	15	12

It is obvious from Table (6) that the complexity of DGMW sequence (or generator) is higher than GMW sequence. So, the security of DGMW generator is more than GMW.

Table (7) presents a comparison between GMW and DGMW generators depending on the complexity (L) of their sequences by applying BM algorithm when n=0,1,…,4094 , $\alpha \in GF(2^{12})$, $m_a(z) = z^{12} + z^6 + z^4 + z + 1$, $m = 12$, $j = 4$, $k = 3$ *and* $r = 13$.

Table (7) The complexity of GMW and DGMW generators of period 4095

Generators	*Period*	*Complexity (L)*
GMW with $m_a(z) = z^{12} + z^6 + z^4 + z + 1$, $m = 12$, $j = 4$, $k = 3$ *and* $r = 13$	4095	36
DGMW with $m_a(z) = z^{12} + z^6 + z^4 + z + 1$, $m = 12$, $j = 4$, $k = 3$ *and* $r = 13$	4095	156

9. Conclusion:

The cipher and communication systems depend on the degree of the security of the key generators to give an acceptable protection to the confidential information from the unauthorized person. So, the paper presents a proposed method to increase the security of the Gordon, Mills and Welch (GMW) generator using Trace polynomial and from the Tables (3), (5), (6) and (7) the following results are listed :

- The sequence of DGMW generator has the same period of GMW generator with good random properties.
- The generator DGMW is more secure than GMW generator since it has higher complexity than GMW generator.
- DGMW generator gives a better accuracy and is consistent with the output sequence than GMW generator.

10. 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. , 2002.
3. Baker, H.J. and Piper, F.C., Cipher Systems: The Protection of Communications, Northwood Publications, London, 1982.

4. Naoki, Suehiro and Mitsutoshi, Hatari "Modula table Orthogonal Sequences and Their Application to SSMA Systems", IEEE Trans. on Inf. Theory, Vol.34, No.1, p. 93-95, Janauary 1989.
5. Baker, J.M. and Hughs, P.M. "Communications Speech and Vision". Jr., Proc. I, IPIDDG 136, 1989.
6. Massey, J.L. "Shift Register Synthesis and BCH Decoding". IEEE Trans. on Information Theory, Vol.IT-15, No.3, p.140-146, January, 1969.
7. Salih, R.K. "Analysis of Pseudo-Noise Generator Designs For Communication Systems Using State-Space Method". M.Sc. Thesis, University of Technology, 2004.
8. Vdrew, Viterbi, Principles of Spread Spectrum, Addison-Wesley publishing, Inc., 1995.
9. Maurice, L. Shiff, Pseudo-Noise Codes from Spread Spectrum Scene Online, www.sss-mag.com/png2.html -13k , March 19, 2003 .
10. Laura, J. Riely, PN-Generators, www.et.fh.merseburg.de/labore/ictech/pn.htm-1k, November , 2004.

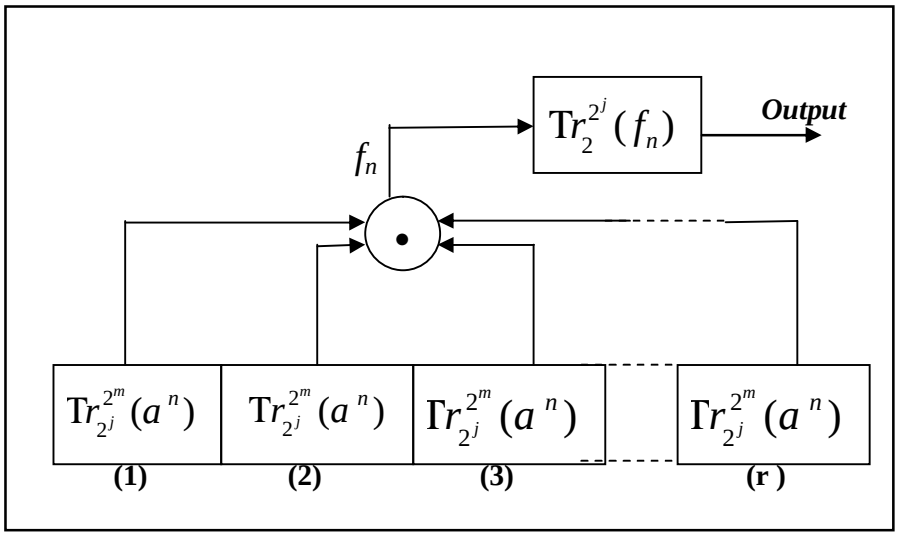

Figure (1) The GMW generator.