Hardware Stream Cipher Based on LFSR and Modular Division Circuit

Deepthi P.P. and P.S. Sathidevi

Abstract—Proposal for a secure stream cipher based on Linear Feedback Shift Registers (LFSR) is presented here. In this method, shift register structure used for polynomial modular division is combined with LFSR keystream generator to yield a new keystream generator with much higher periodicity. Security is brought into this structure by using the Boolean function to combine state bits of the LFSR keystream generator and taking the output through the Boolean function. This introduces non-linearity and security into the structure in a way similar to the Non-linear filter generator. The security and throughput of the suggested stream cipher is found to be much greater than the known LFSR based structures for the same key length.

Keywords—Linear Feedback Shift Register, Stream Cipher, Filter generator, Keystream generator, Modular division circuit

I. Introduction

STREAM ciphers are preferred over block ciphers for encryption in many communication applications owing to their suitability for real time operation. Time critical applications such as multimedia communications provide such an application area wherein stream ciphers will be more advantageous compared to block ciphers. Among various possible stream ciphers, LFSR based structures have gained more popularity thanks to their simple structures and low hardware implementation costs. Hand-held communication devices pose a potential application, where hardware ciphers are highly needful.

The main disadvantage of LFSR based structure is its vulnerability to attack due to inherent linearity in the structure. LFSR based stream ciphers mainly employ two different methods to spoil this linearity. In the first method, non-linearity is introduced by using a suitable cryptographic Boolean function. Combination generators and filter generators are the structures built using Boolean function. In the second method, the LFSR is irregularly clocked to effect non-linearity. The fundamental structures based on this method are step1-step2 generators, alternating step generators, shrinking and self-shrinking generators. Fast correlation attack[1] is one of the most popular attack methods for LFSR based stream ciphers using Boolean function. Shift register based circuit for division modulo an irreducible polynomial over GF(2) is a suggested method for generating hash. Message authentication using shift register based hash is available in the literature [7]. This structure is preferred in many communication applications for message authentication due to the easiness in their hardware implementation. In the proposed stream cipher, the circuit for modulo division is combined with LFSR based keystream generator to develop a stream cipher with much longer periodicity. An LFSR of length L produces a keystream of maximum period $2^L - 1$, when the feedback polynomial is primitive. All stream ciphers based on this structure are also limited to this periodicity. As per the concept of one-time pad, ideal stream ciphers have infinite periodicity and practically all of them need to have very high period. Only possible method to increase the periodicity of generated keystream in conventional LFSR based stream ciphers is to increase the key size L. In the proposed stream cipher, re-seeding of the conventional LFSR keystream generator with shift register circuit for modular division is used to increase the periodicity. The proposed stream cipher is made cryptographically secure by introducing non-linearity in a way similar to that of filter generator. .

The paper is organized as follows. Section II focuses on the fundamental shift register based structures, while Section III discusses the most popular attack method for LFSR based stream ciphers, the fast correlation attack. Section IV discusses the proposal for new keystream generator in detail with the experimental results while Section V gives the hardware structure, properties and analysis of the proposed stream cipher together with experimental results.

II. LFSR based structures for cryptography

Linear Feedback Shift Registers are used as keystream generators in many practical communication systems due to their simple hardware structure. They can produce sequences of large period and good statistical properties. An LFSR of length L consists of L elements capable of storing one bit each. The output of each stage is shifted as input to the next stage. In the LFSR based keystream generator, the input to the final stage is a linear combination of the outputs of all stages wherein, weight of various outputs in the linear combination is specified by the feedback polynomial of the LFSR. An LFSR keystream generator of length L produces maximal length sequence of periodicity 2^L-1 if the feedback polynomial is *primitive.* The output sequences of these keystream generators are easily predictable due to their linearity and hence are not cryptographically strong.

Both authors are with Department of Electronics & Communication Engineering, National Institute of Technology, Calicut, Kerala, India
Corresponding author: Deepthi P.P., 91-495-2286725; fax: ., 91-495-2287250
E-mail: *deepthi@nitc.ac.in*
P.S.Sathidevi : E-mail: sathi@nitc.ac.in

Cryptographically strong pseudo-random sequences are produced by using one or more LFSR structures and introducing non-linearity by some methods. Two major schemes to destroy the inherent linearity in LFSRs are taking output through non-linear Boolean function and irregularly clocking LFSRs. Boolean function can be used to combine the outputs of several LFSRs or outputs of different memory elements of a single LFSR giving rise to nonlinear combination generator and filter generator structures respectively. Step1-step2 generators, alternating step generators, shrinking and self-shrinking generators belong to the category of the clock controlled generators where the LFSR generating the keystream is irregularly clocked.

A shift register based hardware circuit can be used to perform division modulo an irreducible polynomial over GF(2). It is the implementation of well-known Cyclic Redundancy Codes (CRC) which are commonly used as standard error detection mechanism in digital communication systems. The feedback polynomial (g(x)) of the circuit divides the polynomial representing the sequential input data stream (M(x)) and the resultant residue r(x) such that M(x) = g(x) Q(x) + r(x) is left back in the circuit. This circuit could be used to generate hash, which could be used for message authentication. The basic property of a hash generation operation is its one-wayness. When the length of modular division circuit is much less than that of the input it becomes impossible to get the message polynomial M(x) back from the residue r(x).

Combination of an LFSR based keystream generator and the modular division circuit could be used to generate keystreams of large periodicity and increased security.

A. . Nonlinear Filter Generator

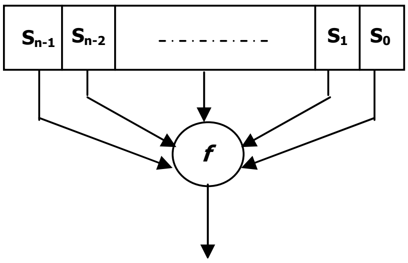

Fig 1 Nonlinear Filter Generator

In a filter generator, the Boolean function combines outputs of different memory elements of a single LFSR, as shown in Fig 1. A detailed analysis of working of the filter generator is given in [3][4][15][16].The period of generated keystream is 2^L-1 or a divisor of 2^L-1 where L is the length of LFSR. The maximum linear complexity is, $L_m = \sum_{i=1}^{m} \binom{L}{i}$ where 'm' is the algebraic order of the Boolean function defining the sequence. When the feedback function for LFSR is primitive and Boolean function '*f*' is balanced, then period of output is exactly 2^L-1.

Two main attack methods for the filter generator are fast correlation attack and generalized inversion attack. Correlation between the Boolean function and a linear function on the same variables as used by the Boolean function is exploited in the fast correlation attack. The fast correlation attack works as an iterative error correction to modify the observed keystream sequence into the corresponding correlated linear transform of the underlying LFSR sequence. Different versions of fast correlation attack use different decoding methods of linear codes [13][5][6].

Another popular attack method is generalized inversion attack[5]. This method gives 100% recovery of keystream and is a very powerful attack for filter generator. But the time complexity of this attack could be increased to great extent by choosing the Boolean function properly to span the full length of the LFSR.

B. Shift Register Based Division Circuit

The hardware circuit used to perform division modulo a polynomial over GF(2) can be implemented efficiently using LFSR. Here CRC operation, which is division modulo an irreducible polynomial over GF(2) is used to generate the cryptographic hash code (CRC code). In a normal CRC calculation, the CRC code is calculated as:

CRC code = M(x) mod g(x)

If g(x) is a polynomial of degree 'n' over GF(2), then 'n' is the hash value size in bits. M is the message to be hashed and M(x) is the message polynomial with degree 'm-1', where 'm' is the message size, and m >> n [7].

The operation of division modulo a polynomial over GF(2) is implemented through a simple LFSR with taps or connections determined by the division polynomial. The state analysis of a mod g(x) circuit can be discussed as below:

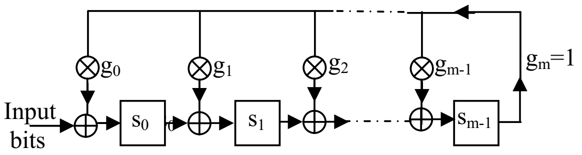

Fig. 2 General block diagram of a mod g(x) circuit

Let the initial contents of the flipflop be $s_0, s_1,..., s_{m\text{-}1}$

The corresponding state polynomial is

$$S(x) = s_0 + s_1 x + ... + s_{m\text{-}1} x^{m\text{-}1} \quad (1)$$

The division polynomial is given by

$$g(x) = g_0 + g_1 x + ... + g_{m\text{-}1} x^{m\text{-}1} + g_m x^m \quad (2)$$

At the 1st clock, the input is a_0. Then the state polynomial after the 1st clock is

$$S'(x) = (a_0 + s_{m\text{-}1} g_0) + (s_0 + s_{m\text{-}1} g_1)\, x + (s_1 + s_{m\text{-}1} g_2)\, x^2 + (s_2 + s_{m\text{-}1} g_3) x^3 + ... + (s_{m-2} + s_{m\text{-}1} g_{m-1})\, x^{m-1} \quad (3)$$

By adding $s_{m\text{-}1}\, x^m$ twice, $S'(x)$ can be written as

$$S'(x) = s_{m\text{-}1}\, g(x) + x\, S(x) + a_0$$

i.e., $S'(x) = [a_0 + x\,S(x)] \mod g(x)$ (4)

Similarly, if the input bit at the 2nd clock is a_1, the state polynomial after the 2nd clock can be written as,

$$S''(x) = [(a_1 + a_0 x) + x\,S(x)] \mod g(x) \quad (5)$$

If there are k input bits, then the state polynomial after the k^{th} clock can be represented as

$$S^k(x) = [(a_{k\text{-}1} + a_{k\text{-}2}\,x + \ldots\ldots\ldots$$
$$\ldots. + a_0 x^{k\text{-}1}) + x\,S(x)] \mod g(x) \quad (6)$$

If the initial state of all the flipflops is zero, then $S(x) = 0$ and the corresponding state polynomial after the kth clock instant can be represented as

$$S^k(x) = [(a_{k\text{-}1} + a_{k\text{-}2}\,x + \ldots + a_0 x^{k\text{-}1})] \mod g(x) \quad (7)$$

Let $m(x)$ be the message polynomial and $m(x)$ is represented as,

$$M(x) = m_0 + m_1 x + \ldots + m_{k\text{-}1}\,x^{k\text{-}1} \quad (8)$$

The 1st input bit is the LSB (coefficient of highest degree) of the message, so

$$a_0 = m_{k\text{-}1}$$
$$a_1 = m_{k\text{-}2}$$
$$\vdots$$
$$a_{k\text{-}2} = m_1$$
$$a_{k\text{-}1} = m_0 \quad (9)$$

The state polynomial after k message bits are given as input can be represented as

$$S^k(x) = (m_0 + m_1\,x + \ldots + m_{k\text{-}1}x^{k\text{-}1}) \mod g(x)$$
$$= M(x) \mod g(x) \quad (10)$$

Depending up on whether the input M(x) is fed from left side or right side, the circuit does either M(x) mod g(x) or M(x). x^n mod g(x).

Fig 3. shows the hardware circuit for a CRC division modulo polynomial g(x) = x^4+ x^3+ 1 over GF(2).

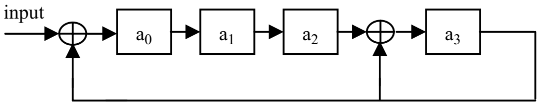

Fig. 3 A Division Modulo Circuit for polynomial g(x) = x^4+ x^3+ 1

A cryptographic variant of this circuit is one in which division polynomial is a part of the secret key and needs to be programmable. Such a circuit can be used for hash generation in communication systems that demand low power consumption and less hardware complexity

III. Cryptanalysis of nonlinear filter generator

Fast Correlation Attack

The idea of a fast correlation attack on nonlinear filter generator was proposed by E. Dawson , L. Simpson[11]. The method is based on the statistical dependence between the keystream generated through the nonlinear filter function, f and the sequence through the best linear approximation of nonlinear filter function, f_l. In the attack, the observed segment of the keystream (Z^N) sequence is considered as a noisy version of a segment of linear transform of the unknown underlying LFSR sequence. Then the problem is viewed as decoding of a Linear Block Code over Binary Symmetric Channel as explained in the references [5][6].

Let f be the nonlinear Boolean function, f_l be the best affine (linear) approximation of f, z^N ($Z^N = \{Z(t)_{t=1}^N\}$) be the observed keystream sequence and d^N ($d^N = \{d(t)_{t=1}^N\}$) be the underlying LFSR sequence, then the keystream can be considered as a noisy version of the unknown underlying LFSR sequence as $Z(t) = f_l(X(t)) \oplus e(t)$, where $f_l(X(t)) = \sum_{j=1}^{n} c_j x_j(t)$ with $c_j \in \{0,1\}$ and $e(t)$ is the noise introduced from BSC.

The following factors are known to the attacker.

- N bits of keystream
- Feedback polynomial of underlying LFSR
- The nonlinear Boolean function f, with Pc > 0.5

Since the nonlinear filter function f is known, the attacker finds the best linear (affine) approximation f_l using the Walsh Transform Technique mentioned in [5][6]. For a Boolean function, $f(x) = f(x_1, x_2 \ldots x_n) : F_{2^n} \to F_2$, the Walsh transform is defined to be the real valued function $F(w)$,

$$F(w) = \sum_{x=0}^{2^n - 1} (-1)^{f(x) \oplus \underline{x}.\underline{w}} \quad (11)$$

where $f(x)$ is the function value for the n –tuple X, $\underline{X} = (x_1, x_2, \ldots x_n)$, $X = x_1 + x_2.2 + \ldots x_n.2^{n\text{-}1}$ *and* $\underline{w} = (w_1, w_2, \ldots w_n)$, $w = w_1 + w_2.2 + \ldots. w_n.2^{n\text{-}1}$. The nonlinearity of $f(x)$ can be obtained from the Walsh transform as,

$$N_f = 2^{n-1} - \frac{1}{2}\max|F(w)|. \quad (12)$$

The probability of correlation between the nonlinear Boolean function and linear approximations is:

$$Pc = \Pr ob(f = f_l) = \frac{2^n - N_f}{2^n} \quad (13)$$

Now, the structure in Fig.1. can be reduced into a linear system (target system) by replacing f with f_l, such that the actual key stream, $Z(t)$ is a noisy version of the output of the target system, $V(t)$, ie. $Z(t) = V(t) + e(t)$.

Let m be the length of the LFSR, U_0 - initial state and N - length of the observed keystream. Then the target system (Fig.4.) can be modeled as a linear block code, C_1 of message length m and code length N , ie, C_1: $[N,m]$. $V = U_0\,G$; G is the generator matrix for C_1 and $G = (\,\boldsymbol{g}_1\,\boldsymbol{g}_2 \cdots \boldsymbol{g}_N\,)$; $g_i - i^{th}$ column of G.

To get the first *'t'* bits of the initial state of LFSR, find all triplets of columns $\boldsymbol{g}_{i1}$, $\boldsymbol{g}_{i2}$, $\boldsymbol{g}_{i3}$ of G such that

$$(\boldsymbol{g}_{i1} + \boldsymbol{g}_{i2} + \boldsymbol{g}_{i3})^T = (\underbrace{\ast,\quad \ast, \ldots,}_{t}\ \underbrace{\ast, 0, 0, \ldots, 0}_{m\text{-}t}); \quad t<m \quad (14)$$

where $\ast$ means an arbitrary value (not all zero).

If there are *'L'* such triplets with k^{th} triple = ($g_{i1}(k)$, $gi2(k)$,$g\ i3(k)$) ; $1 \le k \le L$, then $v_{i1}(k) + v_{i2}(k) + v_{i3}(k)$ is the linear combination of first *'t'* states of the initial state U_0, where $v_{i1}(k)=U_0*g_{i1}(k)$, $v_{i2}(k)=U_0*g_{i2}(k)$, $v_{i3}(k) = U_0*g_{i3}(k)$ and that defines a new *[L, t]* linear block code C_2. The code word in C_2 is,

$v_{i1}(1) + v_{i2}(1) + v_{i3}(1)$, $v_{i1}(2) + v_{i2}(2) + v_{i3}(2)$, . ……. …., $v_{i1}(L) + v_{i2}(L) + v_{i3}(L)$).

This completes the *pre-computation phase* of the attack

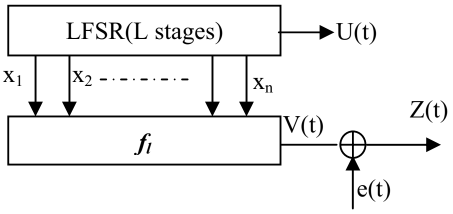

Fig. 4 Target system

Now, with the observed keystream Z, compute a new stream $z_1, z_2,...z_L$ such that,

$$z_k = Z_{i_1}(k) + Z_{i_2}(k) + Z_{i_3}(k); \quad 1 \le k \le L. \qquad (15)$$

This ($z_1, z_2,.....z_L$) is the modified keystream according to the linear approximation of nonlinear Boolean function and is considered as the received word for the code C_2 in binary symmetric channel.

Apply the ML decoding over C_2, to return the codeword in C_2 which is closest to $(z_1, z_2, ... z_L)$. The identified codeword gives the first *'t'* states in U_0. Perform similar parallel iterations to get the other bits in U_0.

Steps for the fast correlation attack can be summarized as follows:

1. Obtain the best linear (affine) approximation of the filter function having maximum P_c.
2. Transform the nonlinear structure of the NLFG into the linear target system.
3. Model the target system as linear block code.
4. To make the attack faster, select some value to *t* such that $t < m$.
5. Get all possible triplets of columns of *G* matrix, such that, sum of those three columns is a column vector with elements other than the first 't' successive elements are zeros.
6. Perform step 5 repeatedly $\left\lceil \frac{m}{t} \right\rceil$ times, shifting (m-t) zero elements cyclically down by 't' bits in each iteration using the equation (14).
7. Using triplets, form LBC, C_2 with code words having bits as the linear combination of specified *'t'* bits of U_0.

Attack phase

Based on the code words in C_2 modify the observed keystream sequence and apply the ML decoding on C_2 by considering the modified keystream$(z_1, z_2,.....z_L)$as the received code word in BSC.

8. Combine the bits obtained from the different iterations to get the possible candidates for the initial state of the target system and choose the one, which give minimum Hamming distance between the actual keystream and the computed keystream.

IV. Proposed Model for Keystream Generator

A. The Model

In this proposed structure, the division modulo circuit is combined with basic LFSR keystream generator to generate a stream of increased periodicity. In this structure, the output stream of fundamental LFSR is passed through the division modulo circuit to generate residue of the output polynomial of the LFSR with respect to g(x), the feedback polynomial of the division modulo circuit. The generated residue is used to reseed the LFSR at the end of its fundamental period, ie., once in every $2^m - 1$ clock cycles, where m is the length of LFSR. The structure is shown in Fig 5.

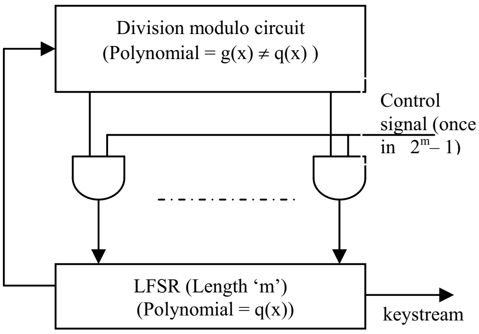

Fig. 5 Proposed model for key stream cipher

Working of the circuit can be explained as follows: Consider an LFSR keystream generator of length 'm' as shown in Figure 6. The feedback polynomial q(x) is given as $q(x) = q_0 + q_1x + q_2x^2 + \ldots\ldots q_mx^m$. Let the initial state of the LFSR be given as $(a_{m-1}, a_{m-2}, \ldots\ldots\ldots\ldots a_1, a_0)$. Then the feedback input bit [3]

$$a_m = a_{m-1}q_1 + a_{m-2}q_2 + \ldots\ldots\ldots a_0q_m. \qquad (16)$$

The output sequence of this LFSR is $(a_0, a_1, \ldots\ldots\ldots\ldots a_{m-1}, a_m \ldots\ldots\ldots)$. Using polynomial notation, the corresponding output polynomial is given by $a(x) = a_0 + a_1x + a_2x^2 + \ldots\ldots.. a_nx^n + \ldots\ldots$

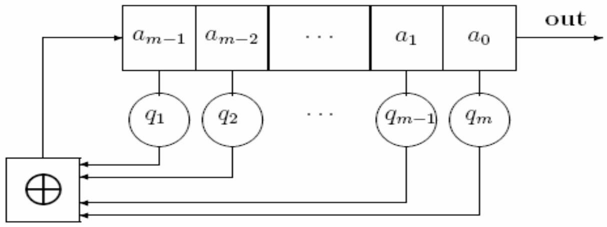

Fig. 6 General LFSR structure

This output polynomial is related to feedback polynomial q(x) as $a(x) = \dfrac{f(x)}{q(x)}$ where the polynomial f(x) is obtained from initial state $\left(a_{m-1}, a_{m-2}, \ldots\ldots\ldots\ldots a_1, a_0\right)$ and coefficients of feedback polynomial $\left(q_0, q_1, \ldots\ldots\ldots\ldots q_{m-1}, q_m\right)$ as,

$$f(x) = \sum_{n=0}^{m-1}\left(\sum_{i=0}^{n} q_i a_{n-i}\right)x^n. \qquad (17)$$

For example with q(x) = x^3+x+1, initial state $\left(a_2, a_1, a_0\right)$ and $f(x) = a_0 + \left(a_0 + a_1\right)x + \left(a_1 + a_2\right)x^2$, output polynomial is

$$a(x) = a_0 + a_1 x + a_2 x^2 + \left(a_0 + a_2\right)x^3 + \left(a_0 + a_1 + a_2\right)x^4 + \left(a_0 + a_1\right)x^5 + \left(a_1 + a_2\right)x^6 + a_0 x^7 + a_1 x^8 + \ldots\ldots. \qquad (18)$$

It could be seen that the sequence is periodic with period 2^m-1 as known.

The output stream of LFSR keystream generator is fed in as input to mod g(x) circuit. At the end of 2^m – 1 cycles, the residue obtained by mod g(x) circuit is fed into the LFSR as the new seed. The input to the mod g(x) circuit is given in such a way that the output bit a_0 of LFSR key stream generator enters first into the modular division circuit, then a_1 etc. So at the end of 2^m – 1 cycles, the polynomial input to mod g(x) circuit is $\hat{a}(x) = a_0 x^{L-1} + a_1 x^{L-2} + \ldots\ldots\ldots\ldots a_{L-1}$ where *L* is the period of output sequence *L* = 2^m – 1. This is the reciprocal polynomial of output polynomial $a(x)$ of LFSR keystream generator for length *L* = 2^m – 1. It can be easily seen that this polynomial $\hat{a}(x)$ is a multiple of feedback polynomial q(x). For the example discussed above,

$$\hat{a}(x) = a_0 x^6 + a_1 x^5 + a_2 x^4 + \left(a_0 + a_2\right)x^3 + \left(a_0 + a_1 + a_2\right)x^2 + \left(a_0 + a_1\right)x + \left(a_1 + a_2\right) \qquad (19)$$

which could be written as,

$$(x^3 + x^2 + 1)\left[a_0 x^3 + \left(a_0 + a_1\right)x^2 + \left(a_0 + a_1 + a_2\right)x + \left(a_0 + a_2\right)\right]$$

Therefore, if the feedback polynomial g(x) of the mod g(x) circuit is same as q(x), output of the mod g(x) circuit will be zero, which should be avoided. Hence g(x) is to be chosen as different from q(x).

It is observed that if the q(x), g(x) pair is properly chosen, the residue returned by the mod g(x) circuit pass through all distinct 2^m –1 combinations without collision, making the periodicity of keystream generator equal to (2^m –1)(2^m –1). Thus, the throughput, which is the ratio of number of output random bits to number of input random bits is (2^m –1)/(m-1) for LFSR keystream generator, while it is $(2^m -1)^2$/(m-1) for the proposed model. For the same q(x), g(x) pair different initial states (keys) can provide different periodicity due to the difference in the properties of generated keystream. It is observed from experiments that many of the reciprocal polynomial pairs are able to provide the maximum period of $(2^m -1)^2$ for all the keys. This means a possible good choice for g(x) is the reciprocal polynomial $\hat{q}(x)$ of q(x). In general, the period of the output sequence (L) for a given q(x), g(x) pair and for a given initial state (i.e., key) is given by $(2^m-1)^2 \geq L > (2^m-1)$. This clearly shows that the proposed model provides a large increase in throughput compared to LFSR keystream generator.

A proper pair of polynomials to provide maximum periodicity could be identified by considering the following equation.

$$\hat{a}(x) = g(x)\beta(x) + r(x) \qquad (20)$$

Here, $\hat{a}(x)$ is the reciprocal polynomial of output polynomial for LFSR with feedback polynomial q(x) and g(x) is the division polynomial used in modular division circuit. If the residue r(x) doesn't go back to initial state $\left(a_{m-1}, a_{m-2}, \ldots\ldots\ldots\ldots a_1, a_0\right)$ before reaching the period 2^m –1, for a chosen g(x), then the chosen g(x) can provide maximum period with that particular q(x).

B. Experimental Results

Based on the study of polynomials, the pairs of polynomials identified to give periodicity of $(2^m-1)^2$, without any re-programmability are shown in Table I. The periodicity of proposed structure in comparison to that of simple LFSR keystream generator for the same number of input random bits (same key size) is also given in the table.

TABLE I PAIRS OF Q(X), G(X) POLYNOMIALS FOR MAXIMUM PERIODICITY

q(x) (keystream gen. poly.)	g(x) (division poly.)	Periodicity obtained: $(2^m-1)^2$	Periodicity of simple LFSR: (2^m-1)
$x^3 + x + 1$	$x^3 + x^2 + 1$	49	7
$x^4 + x + 1$	$x^4 + x^3 + 1$	225	15
$x^5 + x^3 + 1$	$x^5 + x^4 + 1$	961	31
$x^6 + x + 1$	$x^6 + x^5 + 1$	3969	63
$x^7 + x^4 + 1$	$x^7 + x^2 + 1$	16129	127
$x^7 + x + 1$	$x^7 + x^5 + 1$	16129	127
$x^7 + x^3 + 1$	$x^7 + x^6 + x^4 + x^3 + x^2 + x + 1$	16129	127
$x^7 + x^6 + 1$	$x^7 + x^4 + x^3 + x + 1$	16129	127
$x^9 + x^4 + 1$	$x^9 + x^2 + 1$	261121	511

Results of simulation of 4 bit LFSR with q(x) = x^4 + x + 1 and g(x) = x^4 + x^3 + 1 are shown in Table II. The entries below each initial key in Table II shows the re-seed keys generated by mod g(x) circuit at the end of every 15 cycles. The fundamental LFSR with feedback polynomial q(x) = x^4+x+1and initial key (a_3, a_2, a_1, a_0) = (1 0 0 1) gives output polynomial,

$$\hat{a}(x) = x^{14} + x^{11} + x^7 + x^6 + x^5 + x^4 + x^2 + 1.$$

This polynomial when divided by g(x) = x^4 + x^3 + 1, in mod g(x) circuit returns the residue r(x) = x + x^2 + x^3, given as (0 1 1 1) in the first row below column corresponding to key (1 0 0 1). This residue (0 1 1 1) is loaded into LFSR as key for next iteration giving output polynomial,

$$\hat{a}(x) = x^{13} + x^{12} + x^{11} + x^{10} + x^8 + x^6 + x^5 + x^2.$$

TABLE II RESIDUES FED AS KEYS AT DIFFERENT STAGES FOR 4 BIT LFSR

Key = 1 0 0 1	Key = 1 0 0 0
0 1 1 1	1 0 1 1
0 0 1 1	1 1 0 1
0 1 1 0	0 0 1 0
1 1 1 1	1 0 1 0
1 0 0 0	0 0 0 1
1 0 1 1	1 1 0 0
1 1 0 1	1 1 1 0
0 0 1 0	0 1 0 0
1 0 1 0	0 1 0 1
0 0 0 1	1 0 0 1
1 1 0 0	0 1 1 1
1 1 1 0	0 0 1 1
0 1 0 0	0 1 1 0
0 1 0 1	1 1 1 1
1 0 0 1	1 0 0 0

TABLE III RESIDUES FED AS KEYS AT DIFFERENT STAGES FOR 7 BIT LFSR

5	54	119	11	98	73	112	47	77	46
10	51	108	14	109	76	19	42	66	43
15	80	105	1	104	87	22	37	71	24
20	85	102	4	91	82	25	32	92	29
17	90	99	31	94	93	28	96	89	18
30	95	35	26	81	88	7	101	86	23
27	68	38	21	84	107	2	106	83	12
40	65	41	16	79	110	13	111	48	9
45	78	44	115	74	97	8	116	53	6
34	75	55	118	69	100	59	113	58	3
39	120	50	121	64	127	62	126	63	
60	125	61	124	67	122	49	123	36	
57	114	56	103	70	117	52	72	33	

The residue returned by mod g(x) circuit for this input polynomial is $r(x) = x^2 + x^3$ given as (0 0 1 1) in the second row below key (1 0 0 1).

Similarly 127 distinct residues which are used as reseed keys for a single key simulation with 7 bit shift register is shown in Table III.

V. PROPOSED MODEL FOR STREAM CIPHER

Heart of a stream cipher is a secure keystream generator. So in order to design a stream cipher based on keystream generator proposed in section IV, security should be incorporated into this structure. The security is incorporated in the same way as that of nonlinear filter generator. Here the output is taken through a Boolean function which combines the state bits of the keystream generator in a non-linear way, thus introducing non-linearity and security into the structure.

A. Proposed Stream Cipher Model

In this model, security is introduced into the keystream generator in section IV, by using Boolean function to introduce non-linearity.

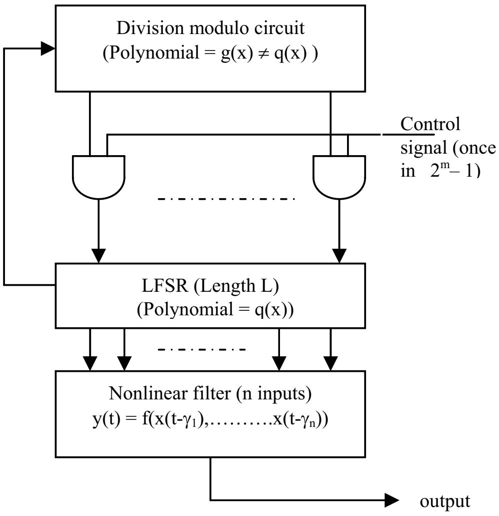
Division modulo circuit
(Polynomial = g(x) ≠ q(x))
Control signal (once in $2^m - 1$)
LFSR (Length L)
(Polynomial = q(x))
Nonlinear filter (n inputs)
$y(t) = f(x(t-\gamma_1),\ldots\ldots\ldots x(t-\gamma_n))$
output

Fig. 7 Structural diagram of proposed stream cipher

The Boolean function combines the state bits of LFSR keystream generator in the same way as that of the filter generator. The structure of this stream cipher is shown in Figure 7.

The periodicity of this stream cipher is same as that of the keystream generator i.e., for an LFSR length of 'm' the period is $(2^m - 1)^2 = 2^{2m} - 2^{m+1} + 1$. For a practical key length of order of m= 128 bits, this is comparable with period of an LFSR based keystream generator of LFSR length '2m' given as $2^{2m} - 1$. Thus compared to a filter generator for the same key size, the proposed stream cipher has a throughput $(2^m - 1)$ times. This means for a given periodicity in key stream output of stream cipher, the number of random key bits required is approximately half, for the proposed structure. It can also be seen that the security of this model is greater than that of the filter generator.

B. Security Analysis for the Proposed Model

Since the output is taken from a filter generator, the attack methods for the proposed model are those of non-linear filter generator. For an attack on the proposed model with fundamental LFSR length 'm', the number of keystream bits required to be known for a successful attack is 2N, where N is the number of bits required for a successful attack on non-linear filter generator. N is calculated as $N = m/C(p_e)$, where $C(p_e)$ is the channel capacity of Binary Symmetric Channel with error probability p_e. The error probability $p_e = 1 - P_c$ where

'P_c' is the correlation probability between the keystream and some linear version of underlying LFSR sequence. For example, consider an LFSR of length m = 10. The number of keystream bits required to be known, for an attack on ordinary filter generator with a practical value of correlation probability $P_c = 0.75$, is 55bits, while that will increase to 110 bits in the proposed stream cipher.

The reason for increase in number of bits required for a successful attack could be explained as below: The two popular attacks on filter generator are generalized inversion attack [3] and fast correlation attack. In generalized inversion attack, a successful attack requires that the guessed state bits are related to other state bits through LFSR linear recursion. With that assumption, unknown state bits are calculated from the guessed state bits. In the fast correlation attack, an approximate linear code structure between state bits and keystream bits are exploited to mount an attack. In the proposed structure, there is a re-seeding through mod g(x) circuit which disturbs this assumed linear recursion. The linear recursion relation between state bits or erroneous linear code structure between state bits and keystream bits will not exist if all the known 'N' keystream bits do not correspond to the same state bits. Assume, for the worst case that keystream bits, which the attacker procured, are tail 'N-k' bits from the stream corresponding to a key and k bits from the stream corresponds to the re-seed of the key. In that case the attack fails due to lack of approximate linear relation between state bits and keystream bits. If the number of known keystream bits are 2N, or more, then there will be anyway, a group of 'N' keystream output bits within this set, such that the required linear recursion for mounting the attack is present between state bits corresponding to these output bits. Take this group of 'N' bits, and mount the attack. The retrieved state could be key itself, or a reseed (residue output of mod g(x) circuit). If the retrieved key is a re-seed, the actual key has to be found by solving equation (20), the required number of times. But this time is comparatively much lesser than the attack time. Hence the advantages of this proposed stream cipher compared to simple nonlinear filter generator can be briefed as below.

(i) Throughput of the system increases by a factor of (2^m –1) times

(ii) Security of the circuit increases in terms of number of bits required for attack, which is double that of filter generator.

The security of the circuit could be increased much, by making the mod g(x) coefficients programmable or by keeping the mod g(x) coefficients random as a part of the key. In both cases, there will be a slight increase in the number of input random bits. But since the periodicity of the circuit is very high, the throughput is still very high compared to LFSR keystream generator. Since the g(x) polynomial need not be primitive, for a given length of LFSR, say 'm', there are approximately 2^m options for g(x) polynomial. One way is to use the coefficients as a part of the key. Another method is to make g(x) coefficients programmable. Here, a few extra random bits will decide the order in which the g(x) coefficients are to be changed from among a set of possible g(x) coefficients. The coefficients of required polynomials can be stored in a buffer, which outputs the coefficients of the selected polynomial at a pre-determined time. Which set of coefficients are given out at an instant is determined by random bits used as a part of the key. The number of additional random bits required here is not very high. . If the number of polynomials for residue operation is 'N' the number of random bits required to select them randomly is only $\log_2 N$. Now, the attack time for this cipher will be increased much. In fast correlation attack of the filter generator, for each possible solution of the reseed, to get back to the initial state, the randomness in the structure demands the generation of multiple keystreams for comparing with the known keystream, or a brute force trial on the set of possible g(x) coefficients at every instant they are programmed or modified. How many key streams are to be generated for each possible solution, is determined by product of the number of polynomials used for residue operation and number of times the coefficients are programmed per period. This itself shows that a slight increase in the consumed number of random bits increase the security by a large amount. Thus, in the design of a practical structure of proposed stream cipher with re-configurability, the main problem is to decide the extent of re-programmability to be used. The three factors to be considered for taking a decision are (i) Increase in security in terms of increased time for attack. (ii) The throughput in terms of ratio of the number of random bits output to the number of random bits input. (iii) Hardware Complexity.

Another possibility to increase the security of the proposed cipher is to make a variation in the mod g(x) circuit. Instead of using a single g(x) polynomial of length 'm' we can choose a $g_1(x)$ polynomial of degree 'n' less than 'm' and another $g_2(x)$ polynomial degree 'm-n'. Then, as explained before, an attack is mount on non-linear filter generator when 2N bits of key stream known. Thus, either the key or the reseed of the key is retrieved. If the retrieved data is a re-seed in l^{th} re-seed cycle, equation (20) should be solved to get the re-seed in $(l\text{-}1)^{th}$ cycle. Here, the residue r(x) is the relevant bits of the retrieved re-seed. Using 'n' residue bits of $g_1(x)$ and 'm-n' residue bits of $g_2(x)$, we have to solve for 'm' reseed bits of $(l\text{-}1)^{th}$ cycle, by applying equation (20) twice. Here, if $g_2(x)$ is a factor of $g_1(x)$, the 'm-n' reseed bits corresponding to residue of $g_2(x)$ doesn't provide any additional information about the key (re-seed) of previous cycle than that provided by the 'n' reseed bits corresponding to residue of $g_1(x)$. Hence, in order to get 'm' re-seed bits of $(l\text{-}1)^{th}$ cycle, it is required to solve (20) with 'n' residue bits leaving 2^{m-n} options on an average as solution. To get the re-seed key in $(l\text{-}2)^{th}$ cycle, group of relevant 'n' bits from each of this 2^{m-n} possible solution should be taken and solve equation (20). This means equation (20) should be solved 2^{m-n} times, each time leaving 2^{m-n} more possible solutions. Thus, to get re-seed key in $(l\text{-}3)^{th}$ cycle equation (20) should be solved 2^{m-n} x 2^{m-n} times and so on. So, the number of options for the initial state increases to brute force possibility of 2^m-1 very fast. This way the security of stream cipher increases enormously by this change in design.

C. Experimental Results

The increased periodicity and security of the proposed structure is clearly visible from experimental results given in Table IV.

TABLE IV COMPARISON OF PROPOSED STREAM CIPHER WITH FILTER GENERATOR

Length of LFSR	Filter generator		Proposed stream cipher	
	Periodicity (bits)	Attack time	Periodicity (bits)	Attack time
7	127	0.69 sec	25400	12.93 sec
8	255	0.91 sec	102000	434.12 sec
9	511	1.1 sec	613200	9765.56 sec

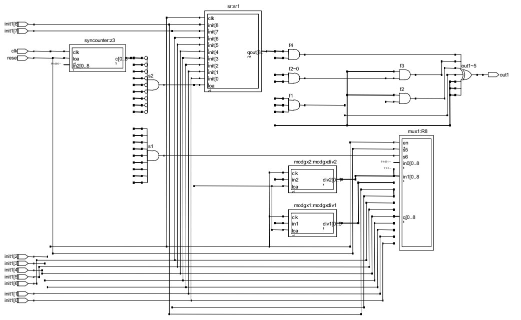

Fig. 8 Schematic diagram of proposed stream cipher from RTL viewer

For the results shown in Table IV, the proposed cipher with re-programmability is implemented. 2 extra random bits are used to program the coefficients of g(x). At predetermined time interval, the coefficients of g(x) is modified. The 2 random bits decide the order in which the coefficients are to be selected from a buffer, which holds 4 sets of g(x) coefficients. The time at which the coefficients are modified are chosen as a suitable multiple of fundamental period of the LFSR. For an LFSR with length m= 7, the coefficients are re-programmed at the end of every 50 periods. Since there are 4 different sets of coefficients, total period is 200 times the fundamental period of the LFSR keystream generator. For m= 8, the coefficients are programmed at the end of 100 fundamental periods etc. The periodicity and security can be increased much with even slight increase in the number of input random bits and circuit hardware complexity.

The proposed cipher is tested for its properties through FPGA implementation also. The architectural layout obtained through RTL viewer for proposed model of non-linear filter generator with no re-configurability is shown in Fig. 8. Here, the LFSR polynomial is taken as $q(x) = x^9 + x^4 + 1$ and the division polynomials are taken as $g_1(x) = x^6 + x^5 + x^4 + x^3 + 1$ and $g_2(x) = x^2 + x + 1$.

VI. CONCLUSION

A hardware stream cipher based on LFSR and polynomial modular division circuit is proposed in this paper. LFSR keystream generator and modular division circuit are combined in such a way that the randomness properties of the LFSR sequences in terms of runs and auto-correlation measures are not disturbed. In the proposed stream ciphers there is a significant improvement in throughput, which is the ratio of number of random bits produced to the number of random bits consumed, compared with the existing standard counterpart. Also, there is a great increase in security in terms of the time for successful attack and the number of keystream bits required to be known for mounting the attack. The structure can be easily extended to higher number of bits than that for which the experiments are carried out. The theoretical analysis of security (time for successful attack) and time period employed in this work are most general. Hence it can be guaranteed that the improvement in these properties of the stream cipher will be clearly available for any length of LFSR chosen. A proper design exercise on higher length of LFSR can give rise to very attractive stream ciphers for use in handheld communication devices and in other communication applications, which demand low hardware complexity and real time operation.

REFERENCES

[1] W. Meier, and O. Staffelbach , “Fast correlation attacks on stream ciphers, *Advances in Cryptology, EUROCRYPT88, Lecture Notes in Computer Science, vol.*330, Springer-Verlag, 1988, pp. 301-314.

[2] T. Siegenthaler, “Correlation-immunity of nonlinear combining functions for cryptographic applications”, *IEEE Trans. on Information Theory*, *vol. IT 30*, 1984,pp. 776- 780.

[3] Mark Goresky, Andrew Klapper, “Algebraic Shift Register Sequences”

[4] Markus Dichtl, “On Nonlinear Filter Generators”**,** *Proceedings of Fast Software Encryption Workshop 1997*

[5] F. J¨onsson and T. Johansson, A Fast Correlation Attack on LILI-128**,** *Information Processing Letters* Vol 81, N. 3, Pages 127–132, 2001.

[6] V. Chepyzhov, T. Johansson, and B. Smeets*,* A simple algorithm for fast correlation attacks on stream ciphers**,** *Fast Software Encryption, FSE’2000*, to appear in Lecture Notes in Computer Science, Springer-Verlag, 2000.

[7] Hugo Krawczyk, “LFSR based hashing and authentication” Proceedings of the 14th Annual International Cryptology Conference on Advances in Cryptology, Lecture Notes In Computer Science; Vol. 839, pp. 129 – 139, 1994

[8] J. Dj. GoliC and M. MihaljeviC, “A generalized correlation attack on a class of stream ciphers based on the Levenshtein distance”, *Journal of Cryptology, vo1.3*(3), 1991, pp. 201-212.

[9] Patrik Ekdahl, “On LFSR based stream ciphers,analysis and design”,*Phd Thesis,Department of Information Technology, Lund University, Sweden,* October ,2003

[10] V. Chepyzhov and B. Smeets. “On a fast correlation attack on certain stream ciphers” *Advances in Cryptology - EUROCRYPT'91*, *Lecture Notes in Computer Science*, no. 547 ,*Springer-Verlag, 1991*, pp 176-185.

[11] E. Dawson , L. Simpson, “ Analysis and Design Issues for Synchronous Stream Ciphers”. Information Security research Centre, Queensland University of Technology.

[12] Sarbani Palit, Bimal K “ Some statistical attacks on stream cipher cryptosystems” *Journal of Indian Statistical Association, vol.42,*May 2004, pp 1-34

[13] T. Johansson and F. Jonsson “ Improved fast correlation attack on stream ciphers via convolutional codes” *Advances in Cryptology - EUROCRYPT'99*, *Lecture Notes in Computer Science*, no: 1592, *Springer-Verlag, 1999,*pages 347-362.

[14] P. Sarkar and S. Maitra. Construction of nonlinear Boolean functions with important cryptographic properties. *In Advances in Cryptology - EUROCRYPT 2000*, LNCS 1807, pages 485–506. Springer Verlag, 2000..

[15] R.A. Rueppel. " Analysis and Design of stream ciphers" *Springer-Verlag, 1986.*

[16] A. Menezes, P. van Oorschot, and S. Vanstone, "Handbook of Applied Cryptography" *CRC Press, 1996.*

[17] Harald Niederreiter, "Coding theory and Cryptology," *Lecture Notes Series, Institute for Mathematical Sciences, National university of Singapore, Singapore university Press*

[18] Jovan Dj Golic, Andrew Clark, and Ed Dawson, "Generalized inversion attack on nonlinear filter generators" *IEEE Trans. on Computers*, *vol. 49*, No.10, October 2000, pp. 1100- 1108.

[19] M. Zhang, "Maximum correlation analysis of nonlinear combining functions in stream ciphers.", *Journal of Cryptology*, *vol 13*(3), 2000, pp .301-313.

Deepthi P.P. received B.Tech Degree in Electronics & Communication Engineering from N.S.S.College of Engg, Palakkad (Calicut University) in 1991, M.Tech Degree in Instrumentation from Indian Institute of Science, Bangalore in 1997. Currently doing Research at National Institute of Technology Calicut in the field of Secure Communication, she has been working as Faculty in institutions under IHRD, Thiruvanthapuram from 1992 to 2001 and in the Department of Electronics & Communication Engineering, National Institute of Technology Calicut from 2001 onwards. Her current interests include Cryptography, Signal Processing with Security Applications, Information Theory and Coding Theory.

P.S. Sathidevi received B.Tech Degree in Electronics Engineering from Regional Engineering College, Calicut (Calicut University) in 1985, M.Tech Degree in Electronics from Cochin University of Science and Technology, Cochin in 1987 and Ph.D from Regional Engineering College Calicut (Calicut University) in 2003 in the field of Speech and Audio Processing. She has been working as Lecturer in the Dept. of Electronics Engineering, National Institute of Technology Calicut (Formerly REC Calicut) from 1990 to 2003 and as Asst. Professor from 2003 to 2007 and as Professor from 2007 onwards. Her current interests include Cryptography, Perceptual audio coding, Speech Coding, Image Coding, Wavelet based speech enhancement techniques for the hearing impaired and Computational Auditory Scene Analysis (CASA).

.