DESIGN and IMPLETATION of KEYSTREAM GENERATOR with IMPROVED SECURITY

Vijay Shankar Pendluri, Pankaj Gupta

Wipro Technologies India

vijay_shankarece@yahoo.com, pankaj_gupta96@yahoo.com

Abstract - **This paper provides the design of stream ciphers based on hash functions and an alternating step generator based on clock control. The keystream generators used for the design of stream ciphers uses low hardware and low power based circuits called Linear Feedback Shift Register circuits. The first two stream ciphers use toeplitz hash, CRC hash and keystream generation circuits whereas the third one uses clock controlling mechanism. Analysis is made for the generation of keystream for clock controlled generator (CCG). The irregular clocking in clock controlled alternating step generator provides good security against various cryptographic attacks. The parameters like periodicity, attacktime and throughput of stream ciphers are measured and compared. As per the results, the first stream cipher gives high periodicity and low attacktime, second stream cipher gives low periodicity and high attacktime. The alternating step generator provides good periodicity, high attacktime and good amount of security compared to other stream ciphers. MATLAB tool is used to calculate periodicity, attacktime and throughput whereas XILINX FPGA is used for the design and implementation purpose. Design includes synthesis, simulation, mapping, place route, verification, and timing analysis done for all the stream ciphers. FPGA results of stream ciphers and alternating step generator evaluate their hardware efficiencies. In all the three designs keystream generation plays a major role.**

Keywords--- **Alternating Step Generator, Clock Controlled Generator, Keystream Generator, Linear Feedback Shift Register**

I. INTRODUCTION

Secret key ciphers are classified into 2 types, stream ciphers and block ciphers. In Stream ciphers, a stream or individual data bits vary with respect to time, whereas in block ciphers, a group of data bits vary with respect to time. It includes internal memory in stream ciphers [4] whereas in block ciphers, no internal memory is involved. Stream ciphers are important class of encryption algorithms because of their low hardware complexity and power consumption. They encrypt binary digits of plaintext message at a time, using an encryption transformation that varies with time. Because of their low error propagation criteria, stream ciphers are advantageous in situations where transmission errors are highly probable. A binary additive stream cipher [1] is just an XOR of message bits and keystream. It consists of keystream, plaintext and cipher text. A secret key is given to the generator function in order to generate a keystream sequence. A secret key K is made common between the sender and receiver, thereby, the receiver can get back the plain text by XOR'ing the cipher text with the keystream. Keystream generation plays a major role in the security of the system. Hash functions [5],[10] are commonly used in communication purpose, especially for authentication and integrity verification of message packets. Basically a hash function is a function that maps a message of variable length to a fixed length hash value that is used in authentication purpose. Hence, the complexity of hardware can be very much reduced by using Hash function in a stream cipher. The security of PRNG varies based on hash functions.

A Pseudo Random Number Generator [8] is used in the generation of bit sequences of longer size. Hence it is more important in keystream generation point of view. As per the definition, it is a time based polynomial algorithm that expands small seeds into larger sequence of bits. A strong PRNG can be produced using either LFSR schemes or One-way function based schemes. Because of the low hardware complexity and less power consumption, LFSR schemes are used commonly in stream ciphers. Due to the linearity in their structure, it becomes easy for their attack in LFSR circuits. Two major schemes to destroy the inherent linearity in LFSRs are taking output through non-linear Boolean function and irregularly clocking LFSRs (clock controlled generator) [2],[3],[9]. Boolean functions combine the outputs of several LFSRs to 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 (CCG) where the LFSR generating the keystream is clocked at different intervals. The only drawback of irregular clocking is keystream period gets shortened, but as per the security it gives best results.

A class of bit oriented key stream generator Ω_k, that comes under ASG (Alternating Step Generator) [3],[9] is studied in this paper. ASG belongs to the family of CCG class. Here, the keystreams generated by K generator G_k have high periodicity, high linear complexity and high throughput. It uses three feedback shift register R1, R2 and R3 in the generation of keystream. The first register R1 changes state as a usual FSR (Feedback Shift Register) where as the other 2 registers R2 and R3are clocked using a clocking mechanism. By XOR'ing outputs of shift-registers R1, R2 and R3, the keystream is generated. So, proper

keystream generation improves the security and attacks like correlation attack can be overcome easily.

DESIGN OF STREAM CIPHERS

This chapter deals with the proposals of 3 stream ciphers and the hardware analysis, periodicity, throughput and attack time of the stream ciphers are calculated.

I. Stream Cipher1

Stream cipher 1 is a combination of LFSR based keystream (PRNG) circuit [1][5][6] and toeplitz hash circuit.

A. LFSR Based Toeplitz Hash Function

Let a message M consists of length 'm' bits and the output hash be of 'n' bits. In order to generate a matrix of order n*m, we need nm elements. By using Toeplitz criteria, it is possible to achieve the matrix of order m*n by using m+n-1 bits. Consider the initial state of LFSR as (S0, S1, S2, S3, S4) = (0 0 0 1 1) and the LFSR hash polynomial be h(x)= x5 + x2 + 1. It is clear that, the polynomial h(x) degree is 5, n = 5 and let the length of message bits be m=9. According to Toeplitz criteria, it needs only m+n-1 bits, (i.e.,13 bits) in order to generate a matrix. The 13 bit output bits of the LFSR are generated by using the initial states and polynomial equation and are given as (S0, S1,., S12) = (0 0 0 1 1 1 1 1 0 0 1 1 0). The first 5 bits of output bits are written from bottom to top in the first column and rest of bits are written in the first row itself, and by shifting the respective columns right side, a matrix is formed. LFSR o/p sequence = [0 0 0 1 1 1 1 1 0 0 1 1 0]

Toeplitz matrix (5 x 9) = 1 1 1 1 0 0 1 1 0
1 1 1 1 1 0 0 1 1
0 1 1 1 1 1 0 0 1
0 0 1 1 1 1 1 0 0
0 0 0 1 1 1 1 1 1

The basic block diagram of LFSR based Toeplitz matrix is shown below. Here, the control register and shift register combination results the LFSR. By changing input, the LFSR state also changes. If the input is '1', the data is loaded to accumulator else it is not loaded to the accumulator, instead it returns the previous state.

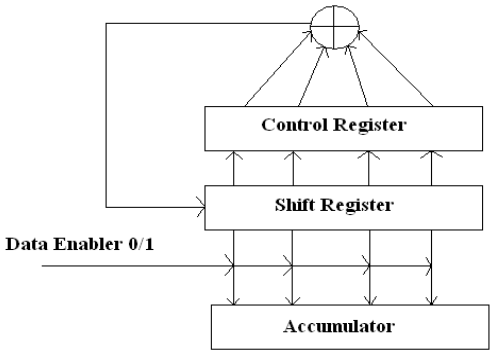


Figure.1 Block diagram of Toeplitz hash

Assume the message of 9 bits be [0 0 0 0 1 0 0 1 0]. Here, the matrix is of order 5*9, and the message is of sixe 9*1. The hash output obtained is by multiplying the matrix with message resulting in 5*1 bits. i.e., the output Hash value is [1 0 1 1 0].

A. Model Of Keystream Generator:

Let the initial key is denoted as 'KEY', hash function as 'HASH'. Let the states generated by function 'F' are S1, S2, S3, ... The first stage output is X1=h(KEY||S1), i.e., concatenation of KEY and initial string. The very next string is generated by the XOR of previous stage output with the KEY and concatenation of current string. The process is repeated and the final keystream is the concatenation of all the stage outputs.

X1=HASH(KEY || S1)

X2= HASH ((KEY $\oplus$X1)||S2)

X3= HASH ((KEY $\oplus$X2)||S3)

……..

Xn= HASH ((KEY $\oplus$Xn-1)||Sn).

The final keystream is given as

KEYSTREAM=X1 ||X2 ||X3 ||……..||Xn

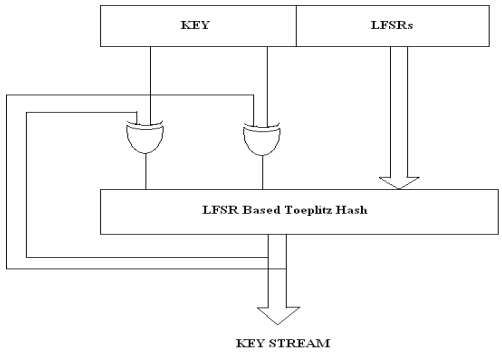


FIGURE .2 PRNG KEYSTREAM GENERATOR

B. Construction of stream cipher1:

Stream cipher 1 is a combines the blocks of LFSR based keystream generator and Toeplitz hash and results in the stream cipher output.

Structure of toeplitz hash is shown below.

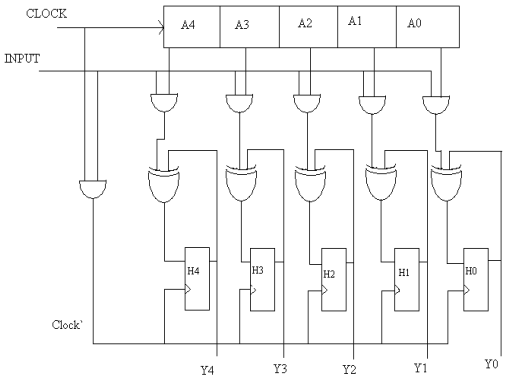


FIGURE .3 LFSR BASED TOEPLITZ HASH

C. *Hardware Implementation*:

Let the states of LFSRT be A0 to A4, and are initialized. Let h(x) =x^5+x^2+1 be the feedback polynomial used here. As the polynomial degree is 5, the output hash will have 5 bits. Therefore m-n=4, i.e., the degree of LFSRS is 4. Let the message is of length 9 bits. The implementation algorithm is as follows: Initially, a clock is given to the LFSR to get the initial states to be loaded to the register. If the Input is high, the states are loaded to the corresponding flip-flops and to the output Y (initial case). Once again if the input is high, the previous output is XORed with the current states and is stored as the present output. If the input is low, the previous output is returned as the output.

D. *Results*

The platform used is MATLAB (for calculating periodicity, attack time) and XILINX ISE 10.1, FPGA (for hardware design). The periodicity is calculated by the formula n*(2^n-1)*($2^{(m-n)}$-1)), here 'm' is input length to hash and 'n' is the degree of polynomial equation. The throughput is given by [n*(2^n-1)*($2^{(m-n)}$-1))]/m and for a simple LFSR based stream cipher it is (2^m- 1)/m. For eg., with n=4, m=8, Period for LFSR is 255 and for the proposed model is 900 and throughput for normal LFSR is 31.8, while that for the current model is 112.5. Attack time is in the order of O(2^k).

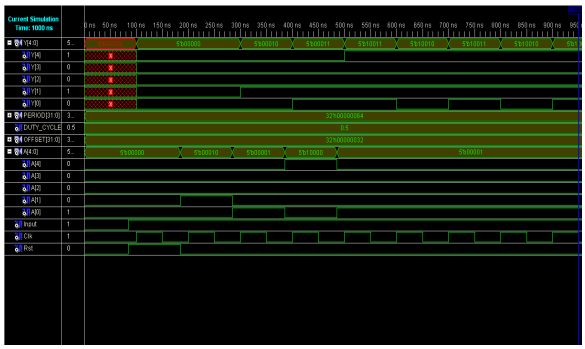
1) *Comparison of Attack Time, Periodicity for various m,n values:*

TABLE1 SHOWS THE PERIODICITY AND ATTACHTIME RESULTS FOR VARIABLE m , n VALUES

Hash o/p length n(bits)	hash i/p length m(bits)	period of key stream	attack time(in s)
4	7	420	0.094
5	8	1085	0.172
5	9	2325	0.812
7	10	6223	0.828
7	11	13335	4.813

From the above table it is observed that the periodicity increases at a high rate and for high values of m,n the attacktime also increases rapidly. Design of the stream cipher can be done using XILINX tool.

The Simulation results of LFSR based Toeplitz are shown below.



I. Stream Cipher2

Stream cipher 2 is a combination of LFSR based filter circuit and polynomial modulo division circuit [1],[6],[7]. Here, CRC hash circuit [1],[5],[7] is generated first and it is given to LFSR filter circuit in order to achieve high security.

A. *Structural block diagram of stream cipher2:*

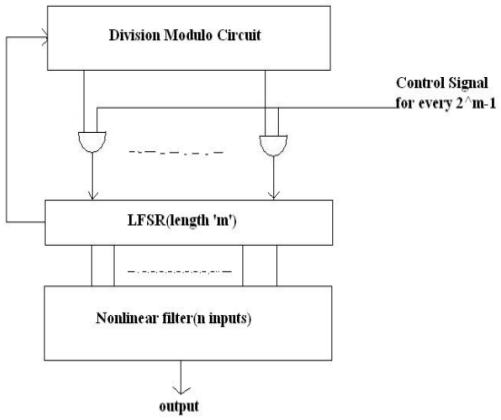


FIGURE 4. STREAM CIPHER2

Division modulo circuit is a CRC circuit that uses a polynomial for the generation of a hash. Here, reseeding mechanism is involved to increase the periodicity and security. Reseeding mechanism is generating a new seed after every 2^m-1 clock cycles. Considering fundamental period of (2^m-1) and through filter generator, another period of (2^n-1), a total of (2^m-1)*(2^n-1) clock cycles, is obtained. i.e., for every (2^m-1)*(2^n-1) clock cycles, it is to be reloaded with initial key.

B. *Experimental Results:*

The periodicity of proposed structure in comparison with the pairs of polynomials to that of simple LFSR keystream generator for the same key size is given in the table.

TABLE2 RELATES Q(X) KEYSTREAM GENERATION POLYNOMIAL, G(X) DIVISION POLYNOMIALS FOR MAXIMUM PERIODICITY.

Q(x)	G(x)	Periodicity of simple LFSR	Periodicity of filter-generator	Attack time of filter-generator	Periodicity of stream cipher2	Attack time of stream cipher2
X^3+X+1	X^3+x^2+1	7	49	0.35s	1011	0.38s
X^4+X+1	X^4+x^3+1	15	225	0.91s	16,320	54.09s
X^5+X^3+1	X^5+x^4+1	31	961	1.2s	111219	10743.4s
X^7+x^3+1	X^7+X^6+ X^3+X^2+1	127	16129	18.1s	High	High
X^9+X^4+1	X^9+X^4+1	511	261121	128.5s	Very High	Very High

By using the polynomials ($Q(x)= x^9+x^4+1$ and $G1(x)=x^6+x^5+x^4+x^3+1, G2(x) =x^2+x+1$) LFSR and division polynomials respectively, a CRC hash is generated. Once a CRC hash is generated, the output is sent through a non linear filter in order to get the stream cipher output. The simulation of defined model is done by using Verilog on XILINX XC3S1500-4fg676 FPGA.

The expected results are shown below.

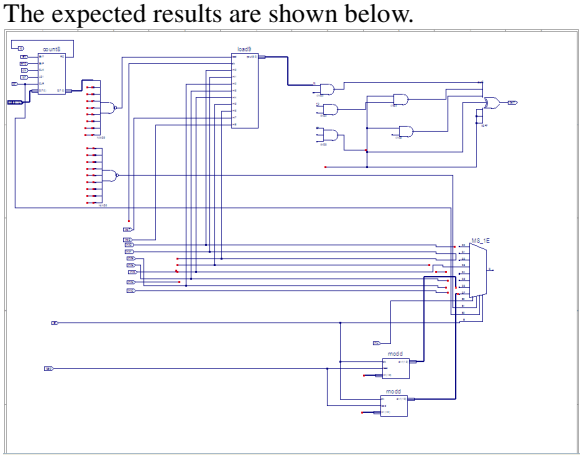


FIGURE.6 RTL SCHEMATIC of STREAM CIPHER2

II. CLOCK CONTROLLED STREAM CIPHER

Clock controlled stream cipher uses irregular clocking as a non-linear function. The main idea behind CCG is to bring back the nonlinearity to keystream generators by controlling the output of one LFSR with other, resulting in the reduction in number of attacks. CCG is classified into two 2 types, the alternating step generator and the shrinking generator. ASG is used in the design point of view in this paper. A clock is given to first LFSR say R1. If the output of first LFSR is high, then second LFSR R2 is clocked. By that time, R3 is not clocked. If the output of first LFSR R1 is low, then third LFSR R3 is clocked and R2 is not clocked. Finally, the resultant keystream is obtained by XOR'ing of outputs of LFSR2 and LFSR3. The generated keystream is XORed with the message bits in order to produce a stream cipher output.

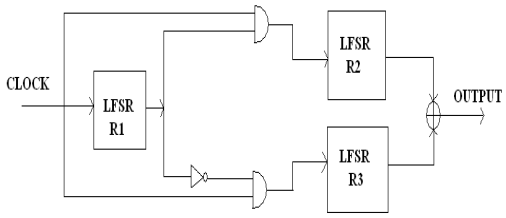


FIGURE 5. CLOCK CONTROLLED GENERATOR

A. Description of a K-generation of clock-controlled alternating step generator (Ω_k)

A generator G_k of the class Ω_k is a binary keystream generator intended for hardware implementation [2],[3],[9]. Every generator G_k is composed of 3 Feedback Shift Registers R1,R2,R3 of lengths l,m,n respectively. Let H={0,1} and K={l,m,n.del1t,del2t}, be arbitrary vectors, where l,m,n are positive integers and deljt(for jϵ{1,2}) gives a decimation function for R1:$\{0,1\}^l$ →varies to $\{1,2,3,\ldots,2^l\}$. For any positive integer i, let R1i,R2i and R3i denote the elements of H^l,H^m,H^n respectively and Xi denotes the elements of $V= (H^l)*(H^m)*(H^n)$. G_k=[K, (f0,f1,f2)], where f0: H^l→H, f1: H^m→H and f2: H^n→H are the feedback functions of R1,R2,R3 respectively. The algorithm is as follows: Clock of R1 controls the clocking of both R2 and R3. At one time t, only one is clocked denotes the ith bit of register R1 by $R1^i(t)$. if $R1^0(t)=1$, the clocking of R2 is performed and the bits $R1^0(t)$, $R1^1(t)$,…. $R1^{w_1-1}(t)$ are the inputs of a clocking unit, otherwise the clocking of R3 is to be performed and the bits $R1^{j0}(t)$, $R1^{j1}(t)$…. ,$R1^{jw_2-1}(t)$ are the inputs. At time t, if $R1^0(t)=1$, R2 is clocked del1t times and R3 is not clocked, otherwise R3 is clocked del2t times and R2 is not clocked, where $Del1t=R1^0(t)[1+2^0\ R1^i{}_0(t)+\ 2^1\ R1^i{}_1(t)+\ldots\ldots+\ 2^{w_1-1}\ R1^i{}_{w1}(t)]$,

$Del1t= \overline{R1}^0(t)[1+2^0\ R1^j{}_0(t)+\ 2^1\ R1^j{}_1(t)+\ldots\ldots+ 2^{w_2-1}\ R1^j{}_{w2}(t)]$ For 0<w1, w2<l, and i0,i1,i2,……iw_1-1, j0,j1,…..jw_2-1ϵ{1,2,…l-1}, Where $\overline{R1}(t)$ is the complement of $R_i(t)$. The output bit is XOR of R2 and R3 if $G_k\epsilon\Omega_k{}^1$ or the XOR of the outputs of R2 and R3 if $G_k\epsilon\Omega_k{}^2$. Keystream $Z=\{Z_t\}_0{}^\infty=Z_0,Z_1,\ldots..Z_\infty$ is the keystream. and Z is given by

$Z= \{R1t \oplus R2\pi1(t) \oplus R3\pi2(t)$ if $G_k\ \epsilon\Omega_k{}^1$,

$\{ R2\pi1(t) \oplus R3\pi2(t)$ if $G_k\epsilon\Omega_k{}^2$.

$R2\pi1(t)= \{R2\pi1(t)\}_0{}^\infty$ ……

Periodicity is given by $Px= 2^l*(2^m-1)*(2^n-1)$

B. Generation of delta function and state diagram:

Keystream Z of a K-generator G_k is a XOR of the irregular decimation of its output sequence. R2t and R3t are governed by the decimation function delt1 and delt2. The secret key comprises the initial state and del1t and

del2t, where the size is proportional to the security. With l≈128, m,n>80, 1<w1,w2<5, gives better security results.

C. *Hardware Implementation*

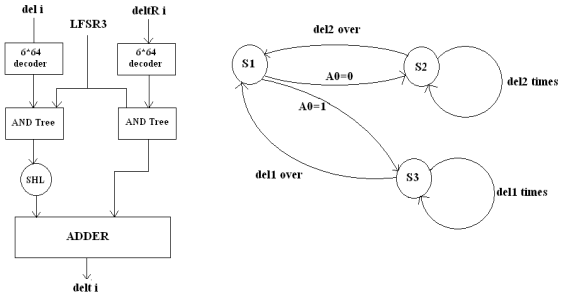


FIGURE.7 delt i FUNCTION GENERATION AND STATE DIAGRAM of CLOCK CONTOLLED CIRCUITRY.

For LFSR based G_k with l=128, the hardware design and implementation is displayed in the figures. The output stream is computed by XOR'ing the register outputs of R1, R2 and R3 if $G_k \in \Omega_k^1$, or $G_k \in \Omega_k^2$ for R2 and R3, for fixed width delt1 and delt2.

Above fig. gives the $delt^i$ (for i=1,2) delt i circuitry and state diagram of clock controlled circuitry and implementation. Here, each decode sets 1 of the 64 outputs based on index value which is of 6-bit and is given as the decoder input. The LFSR3 (i.e., LFSR R1) content is ANDed to the output of the decoder. A left shift operation is done at any time t, to calculate the values of $delt^i$, the multiplication is done to leftmost bit by 2. $delt^i$ is generated by the addition of weighted bits. The state diagram mentioned above consists of three states S0,S1 andS2. The explanation of the state diagram is as follows: If A0=0, the clocking is done for LFSR2, if A0=1, the clocking is given for LFSR3. It will remain in S2 state till del2 times and it will remain in S3 state till del1 times. If del1 over, it will return from S3 to S1 state, else if del2 over, it returns from S2 state to S1 state.

The RTL schematic of Alternating Step generator is shown below.

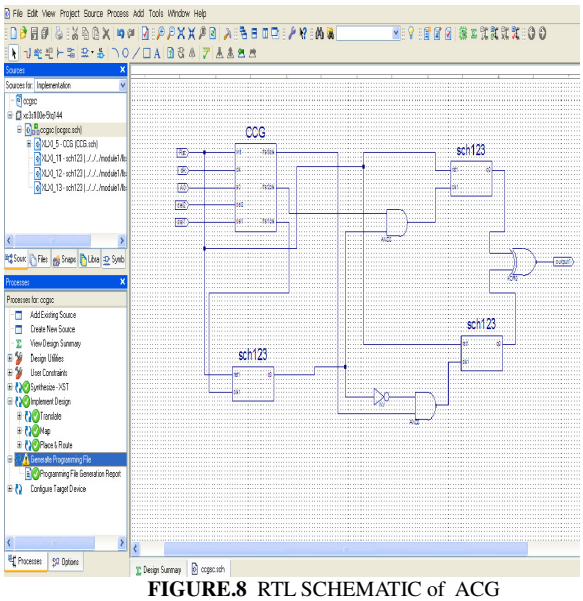


FIGURE.8 RTL SCHEMATIC of ACG

Hardware implementation of proper clock control of 3 LFSRs (CCG block) is shown below. The entire block CCG is used in the above schematic for the control of LFSRs.

RTL schematic of CCG block is shown below:

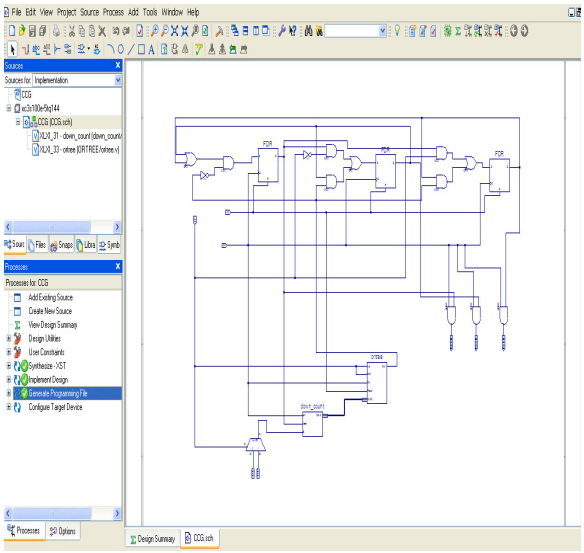


FIGURE .9 RTL SCHEMATIC of CCG

III. COMPARISON RESULTS

Table 3 shows the comparison of various stream ciphers with parameters like periodicity, attacktime, throughput and security are shown below.

TABLE 3. COMPARISON of STREAM CIPHERS

Parameter/stream cipher	Periodicity	Attack-time	Through-put	security
Stream Cipher1	High $n*(2^n-1)*(2^{m-n}-1)$	Medium $O(2^k)$	High $(n(2^n-1)(2^{m-n}-))/m$	Medium
Stream Cipher2	Medium $(2^n-1)*(2^m-1)$	High $O(2^{m+c-n}), C<1$	Medium $(2^n-1)*(2^m-1)/m$	Medium
Alternating Step Generator	High $2^{k}*(2^n-1)*(2^m-1)$	High $(2^k)^2-1$	High $2^{k}*(2^n-1)*(2^m-)/m$	High

A. Comparison of Periodicity and attacktime using MATLAB

1) Periodicity comparison using MATLAB:

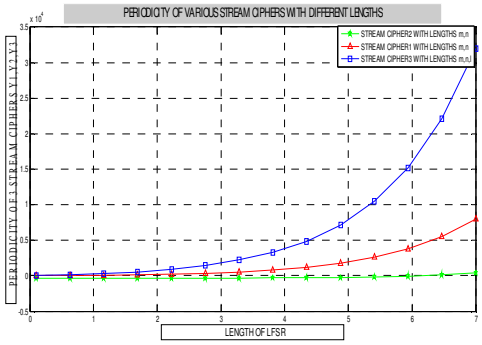


FIGURE .9 PERIODICITY of STREAM CIPHERS

Figure 9 shows the periodicity in clock controlled stream cipher is high compared to other 2 stream ciphers.

2) Attack time comparison using MATLAB:

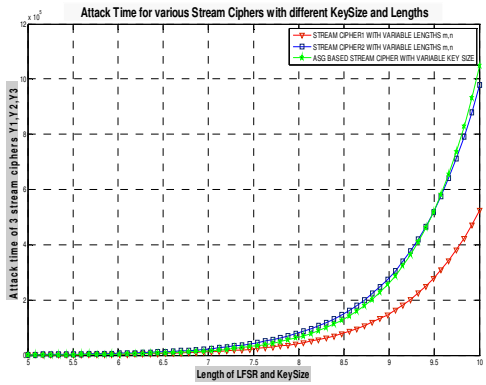


FIGURE .10 Attacktime of STREAM CIPHERS

Figure 10 shows the Attacktime in clock controlled stream cipher is high compared to other 2 stream ciphers.

CONCLUSION

The design of two stream ciphers and the modified clock controlled alternating step generator are made using XILINX FPGA. Periodicity, attacktime and throughput are calculated using MATLAB. This paper includes a class of keystream generators Ω_k intended for the design of hardware. Analysis of generator and its design is given in brief. The clock control introduced in this paper makes cryptanalytic attacks more difficult. As per the results, the periodicity of stream cipher 1 is more compared to the second and the attack time of stream cipher2 is more compared to that of stream cipher 1. It is shown that the keystream of the K-generator G_k, have large period, large linear complexity, high throughput and provides good security compared to the first 2 stream ciphers. Comparison of stream ciphers has been given in terms of periodicity, attacktime, throughput and security.

REFERENCES:

[1] P.P.Deepthi, P.S.Sathidevi,"Design, Implementation and Analysis of Hardware efficient Stream Ciphers using LFSR Based Hash Functions", Science Direct, Computers And Security 28(2009),pp.229-241.

[2] D.Gollmann,W.Chambers,"Clock-Controlled Shift Register" a review, IEEE journal on selected areas of communications, (1989),vol. 7,No.4, pp. 525-533.

[3] C.Gunther, "Alternating Step Generators controlled by De Bruijn Sequences", proceedings of Advances in Cryptography, EUROCRYPT 87,Amsterdam,The Netherlands, 13-15 April, LNCS 309, Springer,Berlin, pp 5-14.

[4] Diffie, W. and Hellman, M.E,"New directions in cryptography", IEEE Transactions on Information Theory, (1976), Vol. IT-22, No. 6, pp.644-654.

[5] Rosiello. A.P.E."Design of a synchronous stream cipher from hash functions", International Journal of computer science and Network Security, (2007), Vol.7,No.8.

[6] P.S.Sathidevi, P.P.Deepthi, "A new hardware efficient stream cipher based on hash functions", International Journal of communication Networks and Distributed Systems 2009-Vol.3, No.4,pp.340-361.

[7] P.P.Deepthi,P.S.Sathidevi, "Hardware Stream Ciphers Based On LFSR and Modular Division Circuit", International Journal of Electronics, Circuits and Systems,2008,Vol.2,No.4

[8] K.C. Zeng, C. H. Yang, D.Y. Wei, and T.R.N. Rao, "Pseudorandom Bit Generators in Stream Cipher Cryptography,"IEEE Computer, Feb.1991,Vol. 24, No.2, pp.8-17.

[9] T.Beth, F.Piper,"The Stop and Go Generator", Springer proceedings of advances in cryptography, EUROCRYPT 84 , 9-11 April, LNCS 209,Newyork,1982,pp.88-92.

[10] William Stallings, "Cryptography and Network Security ",3rd Edition, Pearson Education, Inc;2003.