Exclusive-128 Bit NLFSR Stream Cipher for Wireless Sensor Network Applications

K. J. Jegadish Kumar [#1], K. Chenna Kesava Reddy [*2], S. Salivahanan [#3], S. D. Karthik [#4], V. Praveen [#5]

[#1#3] SSN College of engineering
Anna University,
Kalavakkam, 603110,India
jegadishkj@ssn.edu.in
salivahanans@ssn.edu.in
[*2]Principal, PRRM Engineering College,
Jawaharlal Nehru Technological University,
Shabad, 509217,India
kesavary@yahoo.co.in
[#4]University- Rutgers-the state university of New Jersey,
New Jersey 08901
[#5]Infosys solutions ltd., Chennai, India

***Abstract*-A stream cipher is a common method to protect confidential information from unauthorized person. This kind of cryptosystem is preferred to block ciphers because its implementation in hardware consumes less power and size. Amidst of different stream ciphers, Non-Linear Feedback Shift Register (NLFSR) based one offers the best trade-off between security and hardware capability. This paper describes the stream cipher that generates 128 bit keystream using only NLFSR element as its main function and XOR operation. Hence, this cryptosystem is named as Exclusive-128 NLFSR stream cipher. This cipher consists of different sizes of NLFSRs both in Galois and Fibonacci configuration which offers better trade-off between the algorithm security and hardware capability. The essential design principles in developing the stream cipher are adopted from the A5/1 and modified A5/1 stream ciphers. The proposed stream cipher provides promising efficiency in hardware costs in reconfigurable Field Programmable Gate Array (FPGA) and Application Specific Integrated Circuits (ASIC) implementations.**

Keyword-Cryptography, Stream cipher, NLFSRs, Fibonacci configuration, Galois configuration, FPGA, ASIC.

I. INTRODUCTION

Security is a critical factor for everything in this world; let it be from ordinary short distance communication to large servers that deal with a lot of information. It is essential for the genuineness of our commerce. Any Cryptographic algorithm should not be used as a standard for an unusually long period of time. As the number of hackers and the algorithm reverse engineering techniques are proliferating at a great extent, the modification to the existing algorithms also should be done to make it secure.

Hardware complexity and power consumption are fundamental design criteria for portable communication devices of specific applications like sensor networks, smart cards etc. A stream cipher with modest hardware complexities has enormous significance in the design of stream cipher based cryptosystems. The main goal here is the process of generating the pseudo random bit string. Some of the known stream Ciphers are E0 used in Bluetooth, A5/1 used in GSM communications, RC4 and Grain in RFID applications.

A stream cipher is a symmetric key cipher in which plaintext digits are combined with a pseudo random cipher digit stream (key stream). In a stream cipher, each plaintext digit is encrypted one at a time with the corresponding digit of the key stream, to create a digit of the ciphertext stream. The pseudo random keystream is typically generated serially from a key and an Initialization Vector (IV) or seed value using digital shift registers. The seed value serves as the cryptographic key for decrypting the ciphertext stream.

A stream cipher generates successive elements of the keystream based on an internal state. This state is updated in essentially two ways: if the state changes independently of the plaintext or ciphertext messages, the cipher is classified as a synchronous stream cipher. By contrast, self-synchronizing stream ciphers update their state based on previous ciphertext digits.

Shift Registers are nothing but a simple register which shifts the values and employs some functions to update its bits. LFSR is one of the types which are used for generating Pseudo random sequences. As the name itself suggests, it uses a linear feedback function to update its Most Significant Bit (MSB) or Least Significant Bit (LSB). LFSR is simple, undemanding and easy to implement in both, software and hardware. Mostly used linear function of individual bits is XORed, therefore, naturally it is a shift register whose input bit is driven by

the exclusive-or (XOR) of some bits of the global shift register value. But these LFSRs are not cryptographically secure as the formation of n-bit LFSR can be easily cracked by observing the 2^n consecutive bits of its sequence [1], [2].

Some of the common applications of LFSR include Counters, Built In Self Test (BIST) and Encryption. Few techniques to improve the linear complexity of LFSRs include (a) Non Linear Combining Functions: This technique employs the idea of feeding the outputs of all the LFSRs into a Non Linear Boolean Function to create a Combinational Generator. (b) Clock Controlled Generators: This is another method in which the LFSRs are clocked irregularly, controlled by the output of the second LFSR. Some of the generators include Stop and Go Generator, Alternating Step Generator, Shrinking Generator. The Stop and Go generator consist of two LFSRs, and if the output of one of the LFSRs is '1' the other LFSR is clocked otherwise it repeats the last output. The Alternating Step Generator uses three LFSRs namely$LFSR0$, $LFSR1$, $LFSR2$. Here, if the output of the third LFSR is'0', then $LFSR\ 0$ is clocked and if the output of the $LFSR2$ is 1 then the $LFSR1$ is clocked. The final output is the $ExclusiveOR$ of the last bit produced by $LFSR\ 0$ and$LFSR\ 1$. In the shrinking Generator which uses two LFSRs, if the output of one of them is'1', the output of the following LFSR becomes the main output. Else the output is discarded. In all these clock Controlled generators, they are stepped few times to give just a single bit. Their exposure to timing, power and side attacks find Clock Controlled LFSRs ineffective for pseudo random sequence generators in stream ciphers. (c) Filter Generator: This method depends on the idea of feeding all stages of the LFSR to a non linear Boolean function, and the output of the Non Linear Boolean function is considered as the key stream. There are some disadvantages like Fast Correlation Attack and Generalized Inversion Attack. Examples of LFSR based stream ciphers include A5/1 and E0 Stream Ciphers. A5/1 [3] stream Cipher uses three LFSRs that are of different bit length and employing a different feedback functions.

Because of the above specified inefficiencies of LFSRs, we choose NLFSR for Keystream generating purposes. Hence as an alternative NLFSRs can be used in Stream Ciphers instead of LFSRs. The output sequences of NLFSRs are extremely hard to predict and might need $O(2^n)$ [4] bits to predict an n-bit NLFSR. Some of the stream Ciphers using NLFSRs to generate random sequences are GRAIN [5]and VEST [6].

The following describes the related works of certain stream ciphers like GRAIN, VEST and SNOW. GRAIN is a stream cipher with 160 bit states consisting of 80-bit LFSR and 80-bit NLFSR. In this, one bit in each state of LFSR and NLFSR are updated for every bit of ciphertext released by nonlinear filter function. This specifically designed cipher takes 16 rounds that are carried out in parallel to allow faster implementation at the cost of greater hardware consumption. These ciphers were generally designed to be used in hardware implementations. However, the algorithm provides a greater level of security at the same time maintaining a less hardware complexity [5].

VEST is another stream cipher that is designed using four components: a linear counter, a linear counter diffuser, a linear combiner and a bijective non-linear accumulator. The hardware performance of VEST-32 claims that for 256-bit secure 10 Gbits/sec @167MHz 180 nm ASIC technologies and consumes less than 45 K [6]. A word- oriented stream cipher called SNOW is designed in a simple manner that consists of linear feedback shift register ad a finite state machine. The significant design goal of the algorithm is to create a stream cipher that obviously runs faster than AES and consumes lower implementation cost in hardware. However, the author failed to prove its hardware efficiency [7].

II. MATHEMATICAL BACKGROUND

NLFSR is usually implemented in two configurations. They are Fibonacci Configuration and Galois Configuration. In both the configurations, the current state of the output is a nonlinear function of its previous state. In NLFSRs, the bits are numbered as $0, 1, 2, \ldots . n-2, n-1$ from right to left. In Fibonacci Configuration, the feedback is applied to the last bit i.e. $(n-1)^{th}$ bit. The feedback function is the non-linear function of the previous state of the other bits. This is shown in Fig 1.

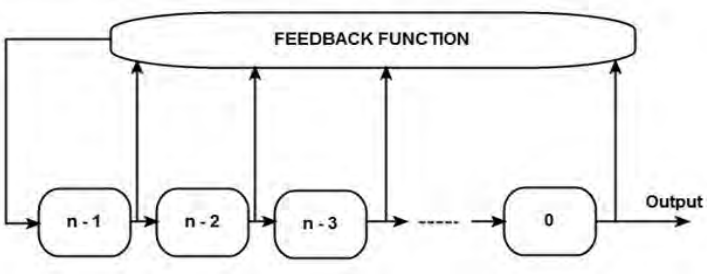


Fig 1.Fibonacci Configuration

In the Galois Configuration, each and every bit is updated based on a pre-defined feedback function as explained in Fig 2. As the feedback is applied to almost every bit, in contrast to Fibonacci Configuration in which the feedback is applied to only the last bit depth of circuits implementing the Galois configuration is much smaller than that of Fibonacci Configuration. Hence the propagation speed and size consumption is significantly reduced in Galois Configuration making it as an attractive part in stream cipher applications.

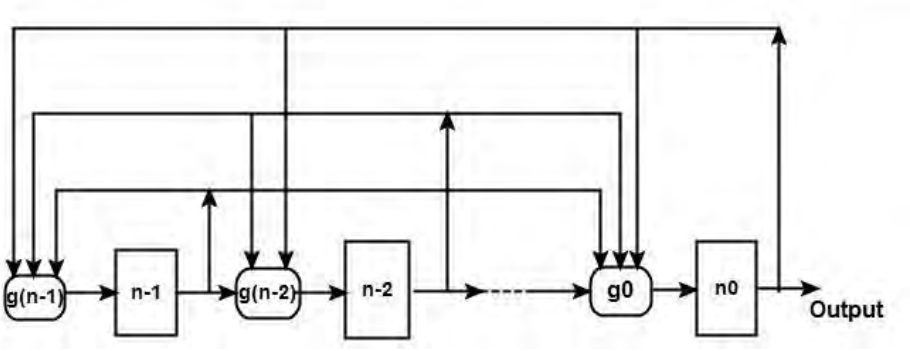


Fig 2.Galois Configuration

A. An Example to Design 32 Bit NLFSR in Fibonacci Configuration

The feedback graph of an 32-bit NLFSR in Fibonacci Configuration is a directed graph with n vertex $\Phi_0,....\Phi_{31}$ which represent the bits $0,\ldots,31$ of the NLFSR, respectively. There is an edge from Φ_i to Φ_j if $i \in dep(f_j)$. The operation substitution, denoted by sub (Φ_i, Φ_j), is defined for any vertex Φ_i which has a single predecessor Φ_j. The substitution sub ($\Phi=\Phi_i$, Φ_j) removes Φ_i from the feedback graph and, for each successor Φ_k of Φ_i, replaces the edge (Φ_i , Φ_k) by an edge (Φ_j ,Φ_k). Given a feedback graph G, the reduced feedback graph of G is a graph obtained by repeatedly applying the substitution to each vertex of G with the input degree 1 until no more substitutions can be applied. It is easy to show that the method of applying the substitution does not affect the resulting reduced feedback graph, i.e. it is unique for a given G. If the feedback graph of a 32-bit NLFSR can be reduced to a single vertex Φ_i, then there exist a non-linear recurrence of order 32 describing the sequence of values of the bit *'i'*[4,7,8].

As an example, consider the 4-bit Fibonacci NLFSR with the feedback function $f_3= \beta_0 \oplus\beta_1 \oplus\beta_2 \oplus\beta_1\beta_3$. The sequence of states is described by the following equations:

$$\lambda_3(t) = \lambda_0(t\text{-}1) \oplus\lambda_1(t\text{-}1) \oplus \lambda_2(t\text{-}1) \oplus\lambda_1(t\text{-}1)\, \lambda_3(t\text{-}1) \quad (1)$$

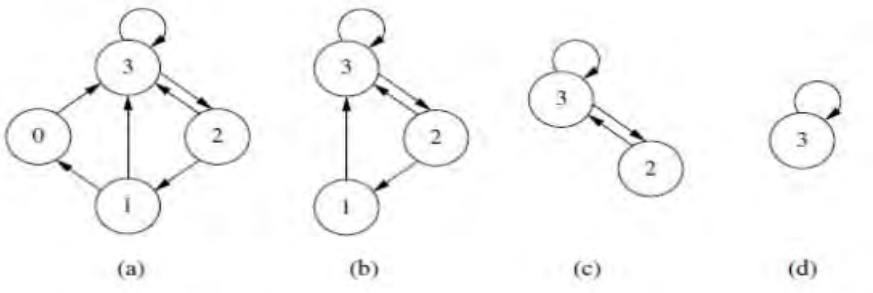
$$\lambda_2 (t) = \lambda_3 (t\text{-}1) \quad (2)$$

$$\lambda_1 (t) = \lambda_2 (t\text{-}1) \quad (3)$$

$$\lambda_0 (t) = \lambda_1 (t\text{-}1) \quad (4)$$

Then the resultant output sequence that NLFSR generates with the period 15 is '111011000101001'.

The feedback graph of this NLFSR shown below in figure below and describes how it is reduced to a single vertex.



(a) Initial graph (b) after sub(Φ_0, Φ_1) (c) after sub(Φ_1, Φ_2) (d) after sub(Φ_2, Φ_3)

Fig 3.Formation of linear feedback function

Reduction steps for the feedback graph of the Fibonacci NLFSR from the example:

1. As in Fig 3 (a) the initial graph Sub (Φ_0, Φ_1) reduces the graph to Fig 3 (b). This is equivalent to substituting $s_0(t)$ by $\lambda_1(t-1)$ into the equation of $\lambda_3(t)$:

$$\lambda_3(t) = \lambda_1(t-2) \oplus\lambda_1(t-1) \oplus\lambda_2(t-1) \oplus\lambda_1(t-1) \oplus\lambda_3(t-1) \quad (5)$$

2. Sub (Φ_1, Φ_2) reduces the graph to Fig 3 (c). This is equivalent to substituting λ_1 (t) by λ_2 (t−1) into the equation of λ_3 (t):

$$\lambda_3(t) = \lambda_2(t-3) \oplus \lambda_2(t-2) \oplus \lambda_2(t-1) \oplus \lambda_2(t-2) \oplus \lambda_3(t-1) \quad (6)$$

3. Sub (Φ_2, Φ_3) reduces the graph to Fig 3 (d). This is equivalent to substituting λ_2 (t) by λ_3 (t −1) into the equation of λ_3 (t):

$$\lambda_3(t) = \lambda_3(t-4) \oplus \lambda_3(t-3) \oplus \lambda_3(t-2) \oplus \lambda_3(t-3)\,\lambda_3(t-1) \quad (7)$$

Hence a 32 bit Non-Linear Feedback Boolean function can be written as:

$$f_{31}=\beta_0 \oplus \beta_2 \oplus \beta_6 \oplus \beta_7 \oplus \beta_{12} \oplus \beta_{17} \oplus \beta_{20} \oplus \beta_{27} \oplus \beta_{30} \oplus \beta_3\beta_9 \oplus \beta_{12}\beta_{15} \oplus \beta_4\beta_5\beta_{16} \quad (8)$$

B Transformation from Fibonacci to Galois Configuration NLFSRs

This section describes the transformation of Fibonacci configuration in to Galois Configuration NLFSRs. The detailed description is given by Elena Dubrova in [4]. Let A_f denotes the set of all product-terms of the Algebraic Normal Function (ANF) of the function f. Throughout the section, P is used to denote a subset of A_f, p denotes as an element of A_f. Let f_a, f_b be feedback functions of bits a and b of an n-bit NLFSR, respectively. The operation shifting, denoted by $f_a \xrightarrow{P} f_b$ moves a set of product-terms from the ANF of f_a to the ANF of f_b. The index of each variable β_i of each product-term in P is changed to $\beta_{(i-a+b)}\ mod n$. The terminal bit τ of an n-bit NLFSR is the bit with the maximal index which satisfies the following condition: For all bits i such that $i < \tau$, the feedback function f_i is of type $f_i = \beta_{i+1}$. Let min index (f) (max index (f)) denote the smallest (largest) index of variables in dep(f). Given a uniform NLFSR with the terminal bit a, a shifting $g_a \rightarrow g_b$, $P \subseteq A_{ga}$, $b < a$, results in an equivalent NLFSR if the transformed NLFSR is uniform. An NLFSR is fully shifted if no product-term of any function g_a can be shifted to a function g_b with the index $b < a$. Note that the fully shifted Galois NLFSR is unique for a given Fibonacci NLFSR. In the following section, the construction of Galois NLFSR is described as in [4,8].

C. Algorithm

Given a uniform n-bit Fibonacci NLFSR N, the fully shifted Galois NLFSR $\hat{N}$ which is equivalent to Nis obtained as follows.

i. First, the terminal bit τ of $\hat{N}$ is computed as:

τ = max (max index (p) − min index (p), ∀ p ∈ $A_g n-1$.

ii. Then each product term p ∈ $A_g n-1$ with min index (p) ≤ (n-1)- τ is shifted to $g_{n-1-\text{min index(p)}}$.

iii. Then each product term p ∈ $A_g n-1$ is shifted to gŕ.

NOTE: for the above case the min-index (p) > ((n-1) – τ).

III.PROPOSED STREAM CIPHER DESIGN

In this section, the formation of ANF of the NLFSR is described with an example giving the method to obtain the feedback functions.

A. Design of 32 Bit Galois Configurations NLFSR

1) Terminal bit calculation

The 32 bit Fibonacci configuration NLFSR is defined as in equation (8). Initial to the transformation, the terminal bit is calculated as following

$$\tau = \max((16-4), (15-12), (9-3))$$

$$\tau = 12$$

2) Shifting the product terms

In this section one of the feedback functions is derived as an example by considering the product term $\beta_4\beta_5\beta_{16}$. This term satisfies the rule min-index (P) ≤ (n-1) - τ. Hence this must be shifted to $g_{n-1-\text{min-index (P)}}$ i.e. f_{27}. Before shifting the index of each variable β_i of each product-term in P is changed to $\beta_{(i-a+b)}$ mod n. Hence $\beta_4\beta_5\beta_{16}$ gets changed to $\beta_0\beta_1\beta_{12}$. Therefore f_{27} feedback function of Galois becomes

$$f_{27} = \beta_{28} \oplus \beta_0\beta_1\beta_{12} \quad (9)$$

The corresponding remaining feedback functions of 32 bit Galois NLFSR that has the terminal bit τ = 12 are derived as:

$$f_{29} = \beta_{30} \oplus \beta_0 \quad (10)$$
$$f_{28} = \beta_{29} \oplus \beta_0\beta_6 \quad (11)$$
$$f_{25} = \beta_{26} \oplus \beta_0 \quad (12)$$
$$f_{24} = \beta_{25} \oplus \beta_0 \quad (13)$$

$$f_{19} = \beta_{20} \oplus \beta_0 \oplus \beta_0\beta_3 \quad (14)$$
$$f_{14} = \beta_{15} \oplus \beta_0 \quad (15)$$
$$f_{12} = \beta_{13} \oplus \beta_1 \oplus \beta_8 \oplus \beta_{11} \quad (16)$$

3) Initial Key

The initial key is a 128 bit string value that is divided as 32 bits, 31 bits, 25 bits, 32 bits, 4 bits and 4bits, and assigned to each NLFSR of corresponding lengths respectively. The NLFSRs used in the proposed stream cipher is both in Galois and Fibonacci configurations. The Initialization vector (IV) or seed value is derived as the complemented value of the initial keystream.

B. NLFSR in Galois configuration (31-Bit)

As per the algorithm specified in subsection C of section II, the 31 bit Galois NLFSR is derived from 31 bit Fibonacci NLFSR. The 31 bit NLFSR in Fibonacci Configuration is derived as in equation (18):

$$f_{30} = \beta_0 \oplus \beta_3 \oplus \beta_5 \oplus \beta_7 \oplus \beta_{10} \oplus \beta_{16} \oplus \beta_{17} \oplus \beta_{18} \oplus \beta_{19} \oplus \beta_{20} \oplus \beta_{21} \oplus \beta_{24} \oplus \beta_{30} \oplus \beta_5\beta_{15} \oplus \beta_{11}\beta_{18} \oplus \beta_{16}\beta_{22} \oplus \beta_{17}\beta_{21} \oplus \beta_1\beta_2\beta_{19} \oplus \beta_1\beta_{12}\beta_{14}\beta_{17} \oplus \beta_2\beta_5\beta_{13}\beta_{20} \quad (18)$$

The 31 bit Galois NLFSR feedback functions that have the terminal bit τ=18 are derived as:

$$f_{28} = \beta_{29} \oplus \beta_0\beta_3\beta_{11}\beta_{18} \quad (19)$$
$$f_{29} = \beta_{30} \oplus \beta_0\beta_{11}\beta_{13} \oplus \beta_0\beta_1\beta_{18} \quad (20)$$
$$f_{18} = \beta_{19} \oplus \beta_5\beta_9 \oplus \beta_4\beta_{10} \oplus \beta_{18} \oplus \beta_{12} \oplus \beta_9 \oplus \beta_8 \oplus \beta_7 \oplus \beta_6 \oplus \beta_5 \oplus \beta_4 \quad (21)$$
$$f_{19} = \beta_{20} \oplus \beta_0\beta_7 \quad (22)$$
$$f_{20} = \beta_{21} \oplus \beta_0 \quad (23)$$
$$f_{25} = \beta_{26} \oplus \beta_0\beta_{10} \oplus \beta_0 \quad (24)$$
$$f_{23} = \beta_{24} \oplus \beta_0 \quad (25)$$
$$f_{27} = \beta_{28} \oplus \beta_0 \quad (26)$$

C. NLFSR in Fibonacci Configuration (25-Bit)

From subsection C of section II, using the algorithm of the transformation, the feedback function of 25 bit Fibonacci configuration is derived as:

$$f_{24} = \beta_0 \oplus \beta_1 \oplus \beta_3 \oplus \beta_5 \oplus \beta_6 \oplus \beta_7 \oplus \beta_9 \oplus \beta_{12} \oplus \beta_{14} \oplus \beta_{15} \oplus \beta_{17} \oplus \beta_{18} \oplus \beta_{22} \oplus \beta_1\beta_6 \oplus \beta_4\beta_{13} \oplus \beta_8\beta_{16} \oplus \beta_{12}\beta_{15} \oplus \beta_5\beta_{11}\beta_{14} \oplus \beta_1\beta_4\beta_{11}\beta_{15} \oplus \beta_2\beta_5\beta_8\beta_{10} \quad (27)$$

D. NLFSR in Fibonacci Configuration (4-Bit)

Using the algorithm in the subsection C of section II, the feedback function of 25 bit Fibonacci configuration is derived as:

$$f_3 = \beta_0 \oplus \beta_1\beta_3 \quad (28)$$

E. NLFSR in Galois Configuration (4-Bit)

Using the algorithm in the subsection C of section II, the feedback function of 4 bit Fibonacci configuration is derived as:

$$f_2 = \beta_3 \oplus \beta_0\beta_2 \quad (29)$$

F. NLFSR in Fibonacci configuration (32-Bit)

The feedback function given in equation (30) of 32-bit Fibonacci configuration is derived using the algorithm in subsection C of section II is,

$$f_{31} = \beta_0 \oplus \beta_2 \oplus \beta_6 \oplus \beta_7 \oplus \beta_{12} \oplus \beta_{17} \oplus \beta_{20} \oplus \beta_{27} \oplus \beta_{30} \oplus \beta_3\beta_9 \oplus \beta_{12}\beta_{15} \oplus \beta_4\beta_5\beta_{16} \quad (30)$$

IV. EXCLUSIVE-128 BIT NLFSR STREAM CIPHER IMPLEMENTATION

The proposed Exclusive-128 NLFSR (Nonlinear Feedback Shift Register) stream cipher is a 128 bit that generates a 128 bit keystream. The cipher is implemented in such a way that each configuration is sandwiched between the other two configurations. The idea of implementing this new stream cipher is adopted from the A5/1 and modified A5/1 stream ciphers [3]. This generated keystream when XORed with the plaintext produces the ciphertext. The 128 bit keystream is produced using six non-linear feedback shift registers, each used in either the Fibonacci or the Galois configuration [8, 9]. The variant size of each configuration is 32 bit Fibonacci, 31 bit Galois, 25 bit Fibonacci, 32 bit Galois, 4 bit Fibonacci and 4 bit Galois NLFSRs. The architecture of the proposed stream cipher is shown in fig.4.

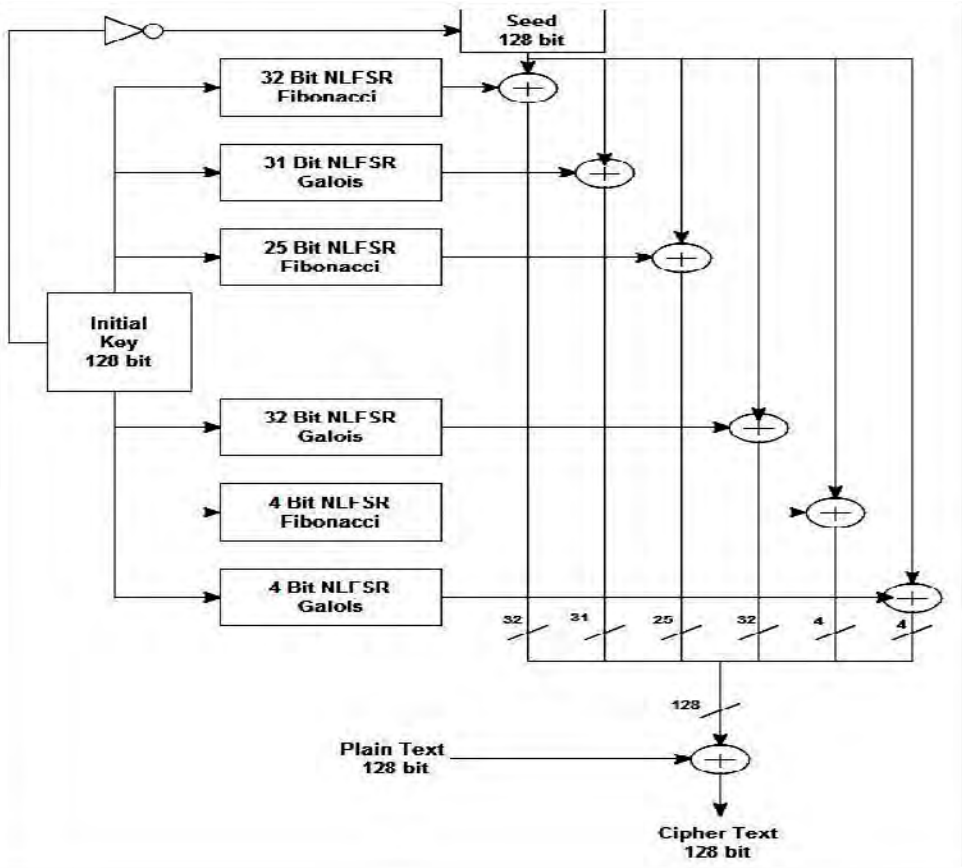


Fig 4. Architecture of Exclusive-128 NLFSR Stream Cipher

In this cipher, each configuration of NLFSR is arranged alternately, and their outputs are XORed with the seed value which is the complement of the initial key. This approach enabled us to encrypt and decrypt when both the key and plaintext vectors are simply 0.

A. Hardware Implementations

The synthesizable VHDL code is written for designing and implementing the cryptographic structure as illustrated in fig.4. At first, the synthesizable VHDL code for each cryptographic architecture is post map simulated, using Modelsim 6.3 to verify the functionality. Then, the FPGA implementation results were extracted with the Xilinx ISE 9.2i tool on a device XC4VLX25 @1.2V, VIRTEX-4 platform with speed grade-12 to determine the memory constraints in slices/area. The XPower Analyzer tool is used to determine the dynamic power consumption. The throughput of each design is calculated using the worst timing constraints by the formula given below.

$$Throughput = \frac{Allowed\ frequency \times Number\ of\ Bits}{Number\ of\ cycles}\ (Bits/sec)$$

Standard Cell 90 nm technology based ASIC implementation was also done to estimate the performance cost in terms of the area and power using SYNOPSYS tools. The Mentor Graphics Modelsim SE PLUS 6.0c is used for simulation and Synopsys Design Compiler version Y-2006.06 for synthesis and power simulation. The foundry typical values of 1.8 Volt for the core voltage and 25 C for the temperature were used, and the suggested wire load model was applied for the power simulation.

The FPGA implementation results of the Exclusive-128 NLFSR stream cipher in Table 1 show that the throughput is incredibly high compared to Grain 128 and SNOW 2.0, and with a compromised reduction in the number of slices and dynamic power consumption.

Table 1: FPGA Implementation of Exclusive-128 NLFSR Stream Cipher

# The Proposed Design Implemented on Virtex 4 XC4VLX25 - 12 FF668, (90 nm Technology)				
Stream Cipher (Key Length)	Max. Work Frequency in MHz	No. of Slices	Dynamic Power Consumption in mW	Throughput in Gbps
# Exclusive NLFSR(128)	455.301	144	11.96	2.591
Grain 128[11]	353.107	44	6.87	0.353
SNOW2.0(128) [12]	5.955	1166	17.77	0.1

The ASIC implementation results of the Exclusive-128 NLFSR stream cipher in Table 2 shows that the throughput is very high compared to Grain 128 and SNOW 2.0 with a compromise reduction in total combinational area and dynamic power consumption. The proposed results show that the total combinational area is 78.7 % increase with respect to Grain 128 and 36.83 % decrease with respect to SNOW 2.0. The dynamic power consumption of the proposed stream cipher is 37.151 % increase with respect to Grain128, and 24.88 % decrease with respect to SNOW 2.0. The stream cipher also shows that the throughput is incomparably very high compared to Grain 128 and 117 % high with respect to SNOW 2.0.

Table 2: ASIC Implementation of Exclusive-128 NLFSR Stream Cipher

# The Proposed Designs Implemented in 90 nm Standard CMOS Cell Technology					
Stream Cipher (Key Length)	Clock Frequency in MHz	Total combinational Area in μm^2	No. Gate equivalents	Dynamic power consumption in μW	Throughput in Gbps
#Exclusive - 128 NLFSR (128)	250	10,777.14	1980	2.359	2.173
Grain 128[11]	250	6030.844	1108	1.72	0.25
SNOW2.0 (128)[12]	250	29261.568	5376	9.48	1

V. CONCLUSION

The proposed Exclusive 128 bit NLFSR stream cipher design is simple with mere shift registers, whose basic element is flip-flop and XOR. This promises efficient implementation in reconfigurable FPGA with high throughput owing to parallelism nature. Equivalently, the ASIC implementation also assures the efficiency in hardware costs. Hence, the security algorithm suits well for the Wireless sensor network applications. Testing the algorithm and various attacks models based studies are to be done as future works.

REFERENCES

[1] J. Massey, "Shift-register synthesis and BCH decoding," *IEEE Transactions on Information Theory*, vol. 15, pp. 122–127, 1969.

[2] J.D. Golic, "On the linear complexity of functions of periodic GF(q) sequences," *IEEE Transactions on Information Theory*, vol. 35, no. 1, pp. 69–75, 1989.

[3] Nur Hafiza Zakaria, Kamaruzzaman Seman and Ismail Abdullah, "Modified A5/1 Based Stream Cipher For Secured GSM Communication", IJCSNS International Journal of Computer Science and Network Security, vol.11 No.2, February 2011.

[4] Elena Dubrova,"ATransformation From the Fibonacci to the Galois NLFSRs", *IEEE transactions on information theory*, vol.55, no. 11,pp. 5263-5271, November 2009.

[5] M. Hell, T. Johansson, and W. Meier, "Grain - a stream cipher for constrained environments," citeseer.ist.psu.edu/732342.html.

[6] B. Gittins, H. A. Landman, S. O'Neil, and R. Kelson, "A presentation on VEST hardware performance, chip area measurements, power consumption estimates and benchmarking in relation to the AES, SHA-256 and SHA-512." Cryptology ePrint Archive, Report 2005/415, 2005.Http://eprint.iacr.org/.

[7] Patrik Ekdahl, Thomas Johnsson, "SNOW – A new stream cipher," In proceeding of the first open NESSIE workshop, 2000.

[8] Y. Tarannikov, "New constructions of resilient Boolean function with maximum nonlinearity", Lecture Notes in Computer Science, vol. 2355, pp. 66–77, 2001.

[9] Elena Dubrova,"Finding Matching Initial States for Equivalent NLFSRs in the Fibonacci and the Galois Configurations", IEEE transactions on information theory, vol. 56, no. 6, June 2010.

[10] Andrew Rukhin, Juan Soto," A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications", National Institute of Standards and Technology (NIST), US department of Commerce, April 2010.

[11] J.M. MarmolejoTejada, V. Trujillo Olaya and J. Velasco Medina, "Hardware implementation of Grain-128, Mickey-128, Decim-128 and Trivium," In IEEE conference on ANDESCON 2010, pp. 1 – 6.

[12] P. Leglise, F.X. Standaert, G. Rouvroy and J.J. Quisquater, "Efficient Implementation of Recent Stream Ciphers on reconfigurable Hardware Devices," In 26th Symposium on Information Theory in the Benelux, 2005, pp. 261–268.

[13] I. Mantin, A. Shamir, "A practical attack on RC4," Proceeding of Fast Software Encryption Workshop 2001, (FSE'2001).

[14] M. Matsui, Block encryption algorithm MISTY, Technical report of IEICE,ISEC96-11,1996.

[15] D. McGrew, S. Fluhrer, "The stream cipher LEVIATHAN, Specification and supporting documentation," Proc. Of First open Nessie workshop, seewww.cryptonessie.org.

[16] W. Meier, and O. Staelbach, "Fast correlation attacks on certain stream ciphers," Journal of Cryptology, 1, pp. 159-176, 1989.

[17] A. Menezes, P. van Oorschot, S. Vanstone, "Handbook of Applied Cryptography," CRC Press, 1997.

[18] R. Rivest, "The RC4 encryption algorithm," RSA Data Security, Inc. Mar. 1992.

[19] T. Siegenthaler, "Correlation-immunity of nonlinear combining functions for cryptographic applications," IEEE Trans. On Info. Theory, 30 (1984), pp. 776780.

[20] D. Stinson, "Cryptography, Theory and Practice", CRC Press.

[21] M. Zhang, C. Caroll, A. Chan, "The software-oriented stream cipher SSC2," Proceeding of Fast Software Encryption Workshop 2000, (FSE'2000).