International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 4 Issue 07, July-2015

Design and Implementation of a Secure Stream Cipher for Cryptographic Applications

Felsa Mary Fidus
Dept. of Electronics and Communication
Sree Buddha College of Engineering, Pattoor

Lalmohan K. S
VLSI Deisgn Group
NIELIT, Calicut

Ambika Sekhar
Dept. of Electronics and Communication
Sree Buddha College of Engineering,
Pattoor

***Abstract:*-Stream ciphers based on hash functions are less complex in hardware and provides better security in embedded systems having space constraints. The commonly used Linear Feedback Shift Register (LFSR) structures with inherent linearity can be used along with hash function based Pseudo Random Number Generator (PRNG) structures to increase periodicity and throughput. In such cases the security of the PRNG depends on the hash function used and the way in which it is used. In this paper, the Cyclic Redundancy Check (CRC) hash circuit is combined with the normal LFSR circuit and the keystream is generated by the CRC circuit. The proposed design is implemented for 8 bit, 16 bit and 128 bit key sizes in FPGA platform. The design possess good security and periodicity for the keystream generated. The proposed method is validated in terms of randomness using the statistical test suite provided by National Institute of Standards and Technology (NIST).**

***Keywords* : Stream cipher, Hash functions, LFSR, CRC, security, periodicity**

I. INTRODUCTION

The technological advancements in embedded systems have paved way for its application in numerous practical scenarios. The constraints like hardware complexity and power consumption has become an important criteria in designing many of these embedded systems. Some of these applications demands security as an important concern. The use of cryptographic algorithms and techniques answer this question of security. Stream ciphers in which data encryption is performed as bit by bit transformations are often needed for high date-rate security applications. Stream ciphers are preferred over block ciphers since they require less buffer space, less complex hardware structure and also permits the possibility of real time encryption for time critical applications. Besides this, the different implementation properties of stream ciphers restrict the side channel cryptanalysis.

The strength of the cryptographic cipher depends upon the generation of unpredictable quantities as Keystream for the encryption of plaintext. Stream ciphers with shift registers as pseudo random number generator (PRNG) structures are popular in practice. Linear Feedback Shift Registers (LFSRs) are the simplest kind of feedback shift registers known to produce binary sequences with good pseudorandom properties [1]. An LFSR of length n consists of n registers capable of storing one bit each. In LFSR, input of the final stage is the linear combinations of the outputs of all stages, where the weights of the linear combination are decided by the feedback polynomial. If the feedback polynomial is primitive, the LFSR of length n will produce maximum periodicity 2^n-1. The LFSRs are susceptible to attack due to the inherent linearity in the structure. Due to the linearity and the effective algebraic implementation of LFSRs, the outputs sequences are easily predictable. One-way functions or hash functions can be used as an alternative method for producing cryptographically strong PRNGs [2].

A hash function, is a function that takes some message of any length as input and transforms it into a fixed-length output called a hash value, a message digest, or a digital fingerprint. A cryptographic hash function has additional security properties due to its one-wayness and hence it can be used as part of Keystream generators in synchronous stream ciphers [3]. A simple and secure way of realizing stream cipher from iterated hash functions was designed in [4] and found that found that the complexity of the algorithm depends on the one-way hash function. In embedded systems with space constraints only less space is dedicated for cryptographic security and only few hash functions are reviewed in the literature, which are optimized for such hardware implementations.

In [5], two LFSR based hash functions suitable for hardware implementations were suggested. The design and analysis of stream ciphers with these functions as building blocks for pseudo random sequence generation were discussed in [2]. As per the analysis, the stream cipher using Toeplitz hash found to be efficient in terms of periodicity than the Cyclic Redundancy Check (CRC) hash function. But the attack time of the CRC hash cipher increases at a faster rate and its hardware complexity is

lesser when compared to the Toeplitz hash function cipher. In this paper, the Toeplitz hash function for keystream generation is replaced by CRC hash function and the structure is re-designed without much additional hardware. The new structure is designed in such a way to increase the periodicity and ensures the randomness of the Keystream generated. The proposed structure is designed using VHSIC Hardware Description Language (VHDL) and implemented in an FPGA device. The periodicity and security analysis of the device is performed in MATLAB and the statistical tests for randomness analysis is carried out by the framework provided by NIST.

The rest of the paper is organized as follows, section II highlights the mathematical preliminaries and theories related to LFSRs and CRCs. The design and development of the proposed model and its explanation is presented in section III. The simulation, performance analysis and implementation of the proposed design are discussed in section IV and section V concludes the paper.

II. PRELIMINARIES

Linear Feedback Shift Registers are used as keystream generators in many practical applications due to their simple hardware structure. They can produce sequences of large period and good statistical properties. An LFSR of length n over the finite field Fq consists of n stages $[s_{n-1},s_{n-2},\ldots..,s_0]$ with $s_i \in Fq$, and a polynomial

$$p(x)= 1+c_1x+c_2x_2+\ldots..+c_nx_n \text{ over Fq.}$$

The contents of $[s_{n-1}, s_{n-2},\ldots..,s_0]$ is called the state of LFSR and p(x) is called the feedback or connection polynomial. In the Fibonacci representation, the register is controlled by a clock, and at each stepping the elements are moved to the right so that $s_i=s_{i+1}$ for i =0,…..,n-2 and the input to the first stage is a linear combination of outputs of all stages specified by the feedback polynomial. If the feedback polynomial p(x) is a primitive polynomial of degree n each of the non-zero initial states produces an output sequence with the maximum possible period q^{n-1}. Such a LFSR is called a maximum-length LFSR.

The key for LFSR should be at least the starting state, thus the n bits for a length of n in the LFSR. From the shift register equation, with knowledge of 2n consecutive bits, n equations are obtained and can solve for unknown previous states and polynomial coefficients. Due to this linearity the output sequences of these keystream generators are easily predictable and hence are not cryptographically strong.

A hash function is a computationally efficient function mapping binary strings of arbitrary length to binary strings of some fixed length values called hash values [1]. Basically hash functions are used for authentication of messages in cryptography. Most of the basic hash functions are not suitable for hardware implementation when used in a stream cipher because it requires complex iteration operations for generation of output random bits. Hash function derived from the well known cyclic redundancy check codes code has proved to have much less hardware requirement than other hash functions.

CRC is a technique used for error detection in digital data networks. It makes the data secure by using a checksum or cyclic redundancy check. A CRC calculation can be mathematically described as a polynomial division performed over the input data by a generator polynomial g(x).

$$\text{CRC code} = M(x) \bmod g(x) \qquad (1)$$

The cryptographic CRC hash function h(M) is defined as:

$$\text{CRC hash} = M(x)\, x^n \bmod g(x) \qquad (2)$$

where g(x) is the irreducible polynomial of degree n over GF(2), where 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, x^n is the multiplication factor.

The operation of polynomial division over GF(2) can be implemented using a simple LFSR with taps or connections determined by the generator polynomial [6] as shown in Figure 1. This hash function can be implemented using flipflops and xor gates only and this makes it suitable for low-power applications.

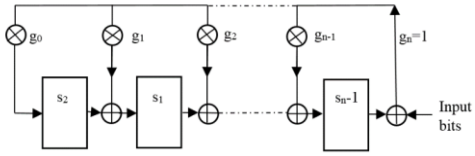

Figure 1 Structure of CRC polynomial division circuit

III. PROPOSEDSYSTEM DESIGN

A cryptographically strong PRNG should generate keystreams that possess a high degree of randomness and the bits generated should be unpredictable. If a PRNG is adequately strong then the chances of tracing the initial states or key of the PRNG from the output keystream are infeasible[7]. The existence of a one-way function introduces these properties in a PRNG resulting in a secure stream cipher. When a random seed is initially applied to a hash function it provides the first set of output bits. If the same seed is used for the next iteration the periodicity of the PRNG is spoiled. If a new seed is used for each of the iterations, the randomness of the input increases resulting in the reduction of throughput. The throughput of a stream cipher is defined as the ratio of number of output random bits produced to the number of input random bits consumed. To avoid these in developing a cryptographically strong PRNG, a good deterministic function has to be found which can randomnly select the input to the one-way function. Based on this concept two designs was developed in [2] identifying a method to generate different random seeds for various iterations.

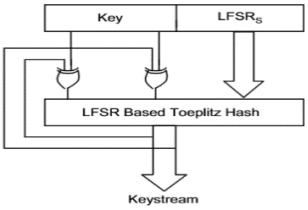

Figure 2 Model using Toeplitz hash

In the stream cipher model using Toeplitz hash[2] as shown in figure 2, the sequence of strings s1,s2,s3,… form the states of a LFSR and the initial state of the LFSR forms a part of the secret key. The LFSR advances its state depending upon the initial state and feedback polynomial used. The key bits are concatenated with the initial state of LFSR and are fed as the first input to the hash function. The periodicity of this model can be expressed as $n(2^n-1)(2^{m-n}-1)$, where m is the length of input to the hash function in bits. The throughput of this model is given by $[n(2^n-1)(2^{m-n}-1)]/m$ which is much larger than that of a simple LFSR based stream cipher given by $(2^m-1)/m$. The security analysis of this cipher model was performed in [8] and found that the security is very low. As per the suggestions in the previous literature if the stream cipher is designed using CRC hash, then the complexity in cryptanalysis is increased much. The hardware has to be designed in such a way to spoil the linearity in the CRC circuit[2] and thereby increasing the periodicity of the generated keystream.

A. Structural block diagram

The proposed model of stream cipher uses a basic Linear Feedback Register to reseed the Cyclic Redundancy Check hash circuit as shown in figure 3. The key is of length n bits and it is equivalent to the number of bits in the plaintext. The key bits are given as the initial seed to the LFSR circuit having n stages. The LFSR connections are specified by a feedback polynomial which yields a maximum length sequence. The taps or connections of the CRC circuit performing modular division are linked as per the generator polynomial selected to ensure enough randomness in the design.

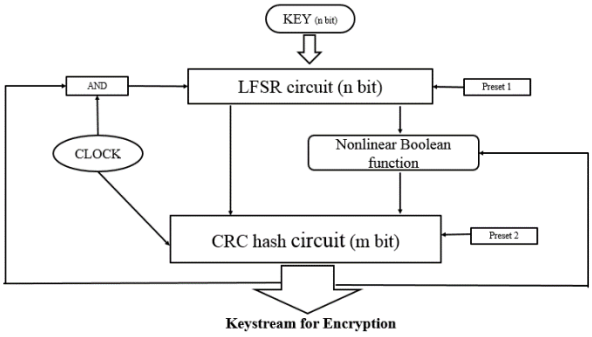

Figure 3 Proposed stream cipher model

The important aspect in the modified stream cipher design was to introduce non-linearity in the generated keystream. This can be achieved by introducing non-linearity in the inputs given to the CRC circuit. A non-linear Boolean function can be used for this purpose. The Boolean functions used for cryptographic applications have important properties like non-linearity, balancedness and correlation immunity [9]. In the proposed design, one of the inputs to the taps at each of the stages of CRC hash circuit are either the linear combination of the output bits of corresponding stages of LFSR and feedback bit CRC hash circuit or the output of the non-linear Boolean function used. When compared to the previous design, the randomness in the proposed method is improved by alternatively giving the LFSR and Boolean function outputs to the CRC hash circuit.

The LFSR is clocked only when the generated hash output is taking a particular value which can be of designer's choice. Since randomnly changing the input to hash circuit at each instant may adversely affect periodicity of the circuit, the clocking of the LFSR is done at a lesser pace. By using the nonlinear Boolean function the cryptanalysis of the keystream generator becomes difficult. The mathematical model of the proposed stream cipher can be represented as follows :

B. The mathematical model

A hash function basically maps 'm' input bits to 'n' output bits where m>>n. If new 'm' random bits are given as input to the hash function, the throughput decreases. So a part of the input to the hash function is derived from the output of the previous iteration of hash function. Remaining bits are derived from the output of the LFSR. Let 'Key' be the initial random seed and comprises of key bits k1, k2, k3,..kn.

$$Key = k1||k2||\ldots||kn$$

Let $S_0$is the initial state of the LFSR and S_1, S_2,….,S_n be the sequences created by the LFSR based on the feedback polynomial. Each of these sequences will be having subsequences $S_{i,j}$.$U_{m,n}$ is an intermediate output between CRC hash and LFSR circuits. $H_{CRC}$is the CRC hash function and X_0,X_1,….,X_t with subsequences $X_{i,k}$ be the outputs from CRC hash function. Let f be the Boolean logic function [10] consisting of a nonlinear combination of random inputs from CRC and hash function outputs.

$$S_1 = Key \oplus S_0$$
$$U_{1,k} = S_{1,k} \wedge X_{0,k}$$
$$X_1 = H_{CRC}\,[(X_{0,i} \oplus f(X_{0,k}\ldots S_{1,j}))||(X_{0,k} \oplus U_{1,k})]$$
$$X_2 = H_{CRC}\,[(X_{1,i} \oplus f(X_{1,k}\ldots S_{2,j}))||(X_{1,k} \oplus U_{2,k})]$$
$$\ldots\ldots\ldots$$
$$\ldots\ldots\ldots$$
$$X_{n+1} = H_{CRC}\,[(X_{n,i} \oplus f(X_{n,k}\ldots S_{n+1,j}))||(X_{n,k} \oplus U_{n+1,k})] \qquad (3)$$
$$\ldots\ldots\ldots\ldots$$
$$X_t = H_{CRC}[\,(X_{t-1,i} \oplus f(X_{t-1,k}\ldots S_{t,j}))||(X_{t-1,k} \oplus U_{t,k})]$$
$$Keystream = X_1||X_2||\ldots..||X_n||\ldots||X_t$$

IV. SIMULATION, IMPLEMENTATION AND PERFFORMANCE ANALYSIS RESULTS

A. Simulation and FPGA implementation of the design

The circuit shown in figure 3 was implemented in FPGA platform for 8, 16 and 128 key sizes using VHDL in Xilinx ISE design suite 14.3. The hardware for the given block diagram is synthesized with the device XC3S200, package FT256 of SPARTAN 3 series.

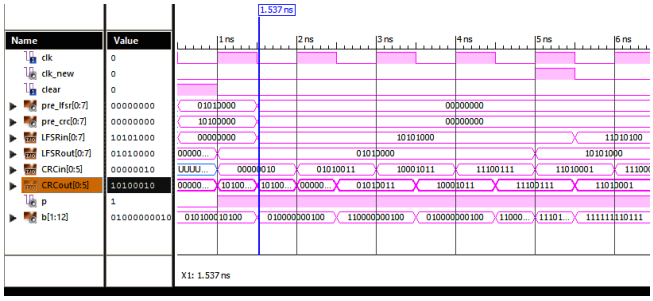

Figure 4 Simulation waveform for 8 bit key size with feedback polynomial $x^8+x^6+x^5+x^4+1$ and generator polynomial $x^8+x^7+x^6+x^5+x^4+x^2+1$ and a random key

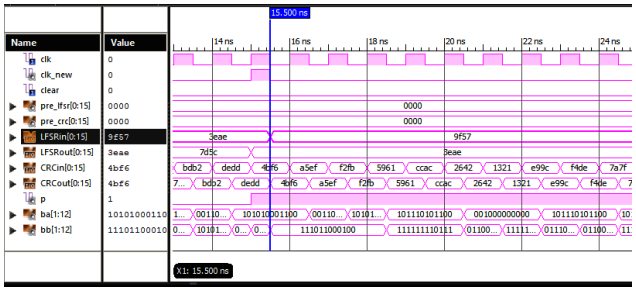

Figure 5 Simulation waveform for 16 bit key size

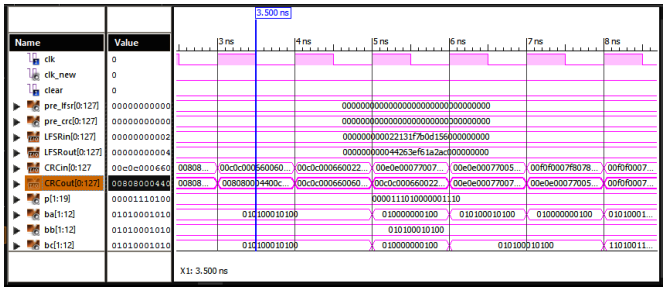

Figure 6 Simulation waveform for 128 bit key size

The architectural layout obtained through RTL viewer for the proposed design is shown in Figure 7. using particular LFSR feedback polynomial suitable for 128 bit key size.

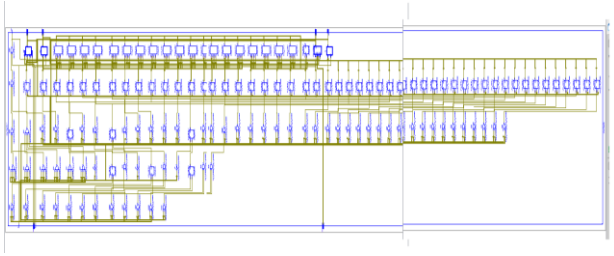

Figure 7 RTL view of the 128 bit final design

Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	299	1920	15%
Number of Slice Flip Flops	513	3840	13%
Number of 4 input LUTs	82	3840	2%
Number of bonded IOBs	771	173	445%
Number of GCLKs	2	8	25%

Table 1 Device Utilization Summary of 128 bit final design

B. Periodicity, throughput and security analysis

The periodicity and security analysis for various key sizes are done using MATLAB. Periodicity of the generated keystream of CRC hash function is computed for different key sizes and initial seeds using MATLAB and it is compared with that of the results obtained from the Toeplitz hash based stream cipher and shrinking generator referenced in [2]. (Table 2) It is evident from the table that the proposed method ensures the generation of keystreams with high periodicity and linear complexity.

Throughput for typical LFSR based stream ciphers is $(2^m-1)/m$, where m is the number of input bits of LFSR. It is possible to increase this throughput by some reseeding mechanisms for LFSR at some clock period. In the proposed design, the CRC hash generating keystream is reseeding by a combination of typical LFSR circuit and a Boolean logic function. The throughput of a stream cipher is defined as the ratio of number of output random bits produced to the number of input random bits consumed.Hence the throughput of the proposed method can be derived as $(n(2^n-1)(2^m-1))/m$, m is the number of hash input bits, which is greater than the already existing methods. By suitably choosing m and n the throughput of the system can be increased to a great extent.

By using a nonlinear Boolean function the cryptanalysis of the keystream generator becomes difficult. The present hash output is modified before using it as input for next iteration. The periodicity of the generated keystream is high compared to the original design. If a few keystream bits from the l^{th}iteration is known, first attack on the hash circuit would be to retrieve the reseed for the l^{th}iteration. But the value to be reseeded is modified at every cycles by the Boolean function, which makes the cryptanalysis difficult. Thus if the value of keystream available for the attacker is not from initial few cycles, it is not possible to deploy correlation attacks or any other methods. A brute force trial is used for the retrieval of original keys and the security increases with brute force attack time. The attack time computed using brute force trials increases exponentially with increasing key size.

C. *Comparison of periodicity and attack time of proposed CRC hash function stream cipher with 2 stream cipher models using Toeplitz hash designed in [2]and shrinking generator of same size*

Type	Input Key size	Periodicity (in bits)	Attack time (in s)
Shrinking generator	8	248	0.0150
	16	65,408	1.5940
Model A	8	1085	0.172
	16	Very high	14181.922
Model B	8	1,00,905	84.5
Proposed CRC hash stream cipher	8	4,65,584	241.259
	16	Very very high	Very high

Table 2 Periodicity and attack time comparison

D. *Randomness analysis*

The randomness of pseudorandom bit sequences are verified using different statistical tests. The keystream generated by the proposed method is tested under the NIST statistical test suite provided by the National Institute of Standards and Technology (NIST)[11].

The NIST test suite consists of 15 different test for the randomness analysis of the input key values. The NIST test suite sts 2.1.1 is used for the study(ANSI C). In the tests, a probability value *p* is produced with significance level, α, is set to 0.01, as suggested for cryptographic applications. A test is considered to be passed if the *p*-value obtained is greater than α. The results of the randomness test are shown in Table 3.

Sl. No	Name of test	*p*-value	Result
1	Runs test	0.7213	Pass
2	Frequency test	0.6438	Pass
3	Longest runs of ones	0.7785	Pass
4	Linear complexity test	0.8154	Pass
5	Approximate Entropy test	0.7521	Pass
6	Maurer's test	0.6510	Pass
7	Cumulative sums test	0.7622	Pass
8	Block Frequency test	9.6937	Pass
9	Overlapping template test	0.5436	Pass
10	Serial test	0.7652	Pass

Table 2 Test results of NIST statistical test suite

V. CONCLUSION

The stream cipher using Linear Feedback Shift Register(LFSR) based Cyclic Redundancy Check(CRC) hash function designed in this paper has proved to be a secure and hardware efficient stream cipher. The keystream generated by the proposed stream cipher shows a significant improvement in performance in terms of periodicity, throughput and randomness. The structure can be easily extended to higher number of bits than for which the experiments are carried out. It is evident from the design that the proposed hash function based stream cipher is having a simple and low hardware complexity structure and hence it is well suited for hardware implementations of embedded system applications demanding less space for cryptographic security.

REFERENCES

[1] J. Menezes, P. C. van Oorschot, and S. A. Vanstone. Handbook of Applied Cryptography. CRC Press, 1997.
[2] Deepthi.P.P, Sathidevi.P.S. Design, implementation and analysis of hardware efficient stream ciphers using LFSR based hash functions. Elsevier Computers and security. 28, 229-241,2009.
[3] M. Bellare, R. Canetti, H. Krawczyk, Keying Hash Functions for Message Authentication Advances in Cryptology, Crypto 96 Proceedings, Lecture Notes in Computer Science Vol. 1109, N. Koblitz ed., Springer-Verlag, 1996.
[4] Angelo P. E. Rosiello, Design of a Synchronous Stream Cipher from Hash Functions, IJCSNS International Journal of Computer Science and Network Security, VOL.7 No.8, August 2007
[5] H. Krawczyk, LFSR-based hashing and authentication, In Advances in Cryptology-Crypto'94, Lecture notes in computer science, vol.839, Springer-Verlag,1994,pp 129-139
[6] S. Lin, D. J. Costello, Error Control Coding - Second Edition, Pearson Education Inc. Pearson Prentice Hall, 2004.
[7] E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid, Recommendation for Key Management – Part 1: General (Revision 3), NIST Special Publication 800-57, July 2012
[8] N.Nisha P. K, Deepthi P P, K.S. Lalmohan, Design and analysis of stream cipher of Low hardware complexity, International Conference on Communication Systems and Network Technologies, 2012
[9] P. Sarkar and S. Maitra, Construction of nonlinear Boolean functions with important cryptographic properties, In Advances in Cryptology - EUROCRYPT 2000, number 1807 in Lecture Notes in Computer Science, pages 491–512. Springer Verlag, 2000.
[10] Y. J. Baek, Yongin-si ,Cryptographic Logic Circuits And Method Of Performing Logic Operations, United States patent application publication, Aug. 16, 2007
[11] A. Rukhin et al, A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, NIST Special Publication 800-22, Revision 1a, 2010