High Performance Architecture for LILI-II Stream Cipher

N. B. Hulle
GHRIEET, Pune
Domkhel Road
Wagholi, Pune

R. D. Kharadkar, Ph.D.
GHRIEET, Pune
Domkhel Road
Wagholi, Pune

S. S. Dorle, Ph.D.
GHRCE, Nagpur
Hingana Road
Nagpur

ABSTRACT
Proposed work presents high performance architecture for LILI-II stream cipher. This cipher uses 128 bit key and 128 bit IV for initialization of two LFSR. Proposed architecture uses single clock for both LFSRs, so this architecture will be useful in high speed communication applications. Presented architecture uses four bit shifting of $LFSR_D$ in single clock cycle without losing any data items from function F_C. Proposed architecture is coded by using VHDL language with CAD tool Xilinx ISE Design Suite 13.2 and targeted hardware is Xilinx Virtex5 FPGA having device xc4vlx60, with package ff1148. Proposed architecture achieved throughput of 224.7 Mbps at 224.7 MHz frequency.

General Terms
Hardware implementation of stream ciphers

Keywords
LILI, Stream cipher, clock controlled, FPGA, LFSR.

1. INTRODUCTION
Message secrecy is one of most important aspect of communication, but especially in wireless environment messages are highly insecure and encryption is must in such environment. Implementations of cryptographic algorithms in hardware usually achieve superior performance when compared with software based ones. It is the growing requirements for high-speed and high-level of secure communications. Security algorithms and their associated keys are more secure, when implemented in hardware because they cannot easily modified by an outside attacker [1]. Reconfigurable devices such as FPGAs are a highly attractive option for a hardware implementation as they provide the flexibility of dynamic system evolution as well as the ability to easily implement a wide range of algorithms [2], [3].

During the last three decades, the art of securing all forms of data has improved to a great extent. The complexity of the functions that can now be performed by the security equipment has increased. The majority of the more effective cipher systems have become far more mathematical intensive and hence their evaluation has become complex and abstract.

The LILI-II stream cipher is LFSR based synchronous stream cipher, designed with larger internal components than previous ciphers in this class. This cipher is improved version of the LILI-128 cipher and capable of providing higher level of security [4]. This stream cipher is less efficient in software because of large size of LFSR and bigger Boolean function, used to increase level of security. It uses 128-bit key and 128 bit Initialization Vector (IV) for initializing LFSRs [5].

Proposed work presents high performance LILI-II stream cipher. This architecture uses same clock for both LFSRs. It is capable of shifting $LFSR_D$ content by one to four stages, depending on value of function F_C in single clock cycle without losing any data from function F_C.

The work described in this article is part of sponsored project by BCUD, Savitribai Phule Pune University, Pune to strengthen the research activity among the academicians. Proposal No. 13ENG001265

2. LILI-II STREAM CIPHER
LILI-II is synchronous stream cipher developed by A. Clark et al. in 2002 by removing existing weaknesses of LILI-128 stream cipher. It consists of two subsystems, clock controlled subsystem and data generation subsystem as shown in Fig. 1.

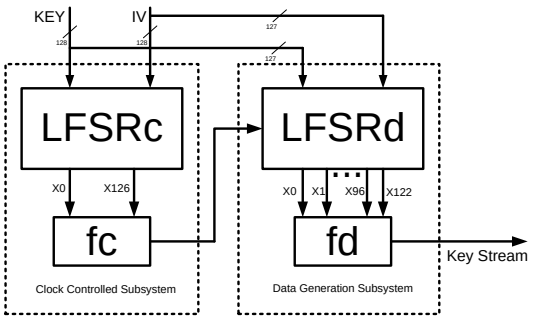

Fig. 1: Block diagram of LILI-II stream cipher

Clock controlled subsystem consists of $LFSR_C$ and function F_C. $LFSR_C$ is 128 bit shift register having primitive feedback polynomial as shown by equation (01) [5].

$$X^{128}+X^{126}+X^{125}+X^{124}+X^{123}+X^{122}+X^{119}+X^{117}+X^{115}+X^{111}+X^{108}+X^{106}+X^{105}+X^{104}+X^{103}+X^{102}+X^{96}+X^{94}+X^{90}+X^{87}+X^{82}+X^{81}+X^{80}+X^{79}+X^{77}+X^{74}+X^{73}+X^{72}+X^{71}+X^{70}+X^{67}+X^{66}+X^{65}+X^{61}+X^{60}+X^{58}+X^{57}+X^{56}+X^{55}+X^{54}+X^{53}+X^{52}+X^{51}+X^{50}+X^{49}+X^{47}+X^{44}+X^{43}+X^{40}+X^{39}+X^{36}+X^{35}+X^{30}+X^{29}+X^{25}+X^{23}+X^{18}+X^{17}+X^{16}+X^{15}+X^{14}+X^{11}+X^{9}+X^{8}+X^{7}+X^{6}+X+1 \quad (01)$$

Function F_C is accepting stage 0 and stage 126 output bit of $LFSR_C$ as input and produces four possible combinations of output. These four combinations of output decide shifting of $LFSR_D$ by one to four stages. Function F_C is given by equation (02).

$$Fc(X_0, X_{126}) = 2X_0 + X_{126} + 1 \quad (02)$$

Data generation subsystem consists of shift register $LFSR_D$ and function F_D. $LFSR_D$ is 127 bit shift register having primitive feedback polynomial shown by equation (03).

$$X^{127}+X^{121}+X^{120}+X^{114}+X^{107}+X^{106}+X^{103}+X^{101}+X^{97}+X^{96}+X^{94}+X^{92}+X^{89}+X^{87}+X^{84}+X^{83}+X^{81}+X^{76}+X^{75}+X^{74}+X^{72}+X^{69}+X^{68}+X^{65}+X^{64}+X^{62}+X^{59}+X^{57}+X^{56}+X^{54}+X^{52}+X^{50}+X^{48}+X^{46}+X^{45}+X^{43}+X^{40}+X^{39}+X^{37}+X^{36}+X^{35}+X^{30}+X^{29}+X^{28}+X^{27}+X^{25}+X^{23}+X^{22}+X^{21}+X^{20}+X^{19}+X^{18}+X^{14}+X^{10}+X^{8}+X^{7}+X^{6}+X^{4}+X^{3}+X^{2}+X+1 \quad (03)$$

$LFSR_D$ shifting is controlled by function F_C. The function F_D takes twelve inputs from $LFSR_D$ stages (0, 1, 3, 7, 12, 20, 30,

44, 65, 80, 96, and 122) and provides one bit output as key stream for each clock cycle. Function F_D is lookup table given in the specification [5] of LILI-II stream cipher.

3. PROPOSED LILI-II ARCHITECTURE

Proposed LILI-II architecture is shown in Fig. 2. It consists of two subsystems, clock controlled subsystem and data generation subsystem. Initial values of the LFSRs in both subsystems are obtained from 128 bit secret key and 128 bit publicly known IV. If publicly known IV is not of 128 bit, then multiple copies will be concatenated or truncated to form 128 bit vector.

Initial state of $LFSR_C$ is obtained by XORing of 128 bit key with 128 bit IV. Similarly, initial state of $LFSR_D$ is obtained by deleting first bit of 128 bit key, last bit of 128 bit IV and XORing the two 127 bit strings.

If cryptanalyst has access to output Z (i) and IV during initialization process, he may recover key for available set of Z (i) and IV. Appropriate security is maintained, when key loading process do not leak information about key during key loading process. For avoiding access of Z (i) to cryptanalyst during key loading process, DFF is used at output. This DFF will not latch the input string till completion of initialization process [5], [6], [7], [8], [9].

Actual initialization process uses LILI-II structure two times for a period of 255 clock cycles each. At the start $LFSR_C$ and $LFSR_D$ are loaded with initial state as explained above and architecture runs to produce 255 bit binary string. This generated 255 bit string is used as initial state of $LFSR_C$ and $LFSR_D$. First 128 bits of generated string are used for $LFSR_C$ and remaining 127 bits for $LFSR_D$ [5], [6].

Second time this architecture runs again to produce 255 bit string by using initial stages from previously generated string.

Now, this time generated 255 bit string is loaded into $LFSR_C$ and $LFSR_D$ stage to begin the key stream generation. As previously, first 128 bits are used for $LFSR_C$ and remaining 127 bits for $LFSR_D$ [5], [6].

Proposed architecture uses two input multiplexor for loading initial values during initialization stage and shifting LFSR content during normal operation. When "Initial Data Load $LFSR_C$" terminal is '1' then initial values are loaded into $LFSR_C$ stages at positive edge of clock, otherwise $LFSR_C$ contents are shifted by one bit towards right at positive edge of each clock.

The Boolean expression of function F_C has three terms $2X_0$, X_{126} and constant 1. The term $2X_0$ has only two possible values, either 00 or 10 (only MSB is changing). The term X_{126} also has two possibilities 0 or 1 and third term is constant 1. The truth table has four terms A_1 ($2X_0$, MSB), A_0 (constant '0', LSB), B_0 (X_{126}), C_0 (constant '1') as shown in Table 1.

Table 1. Truth table for Boolean function F_C

Inputs	Case 1	Case 2	Case 3	Case 4
$A = 2X_0$	0**0**	0**0**	1**0**	1**0**
$B = X_{126}$	0	1	0	1
$C = 1$	1	1	1	1
Sum	1	0	1	0
Carry	0	1	1	10
F_C	001	010	011	100

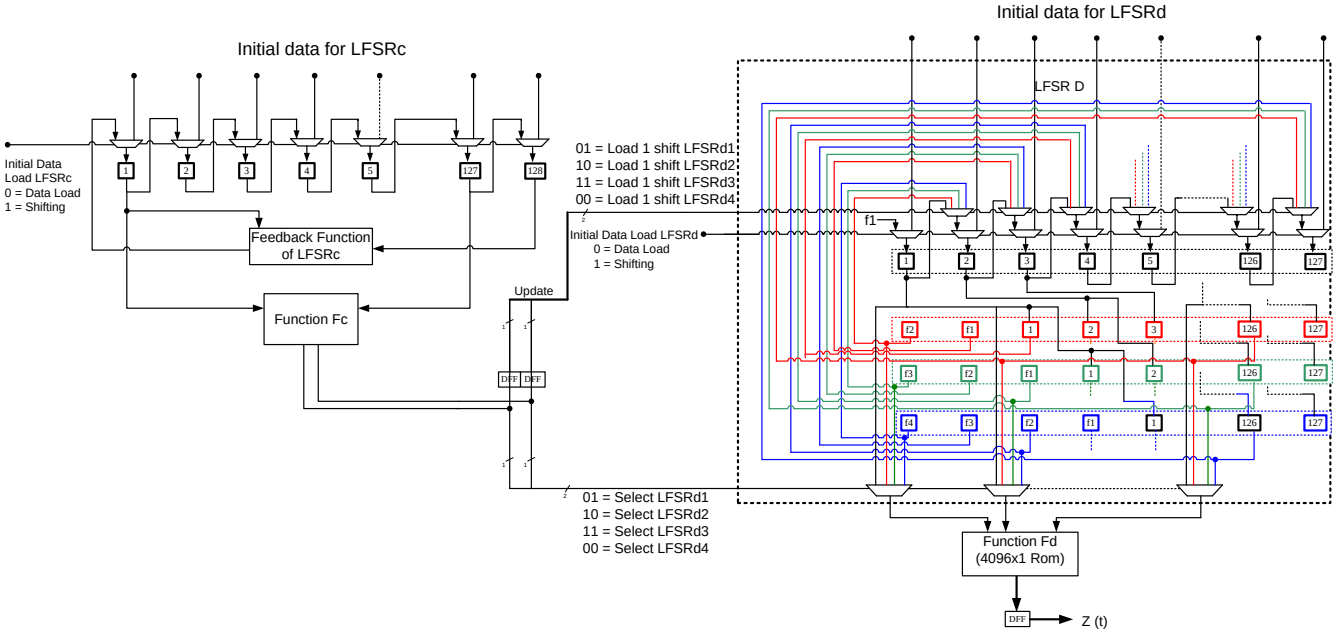

Fig. 2: Proposed Architecture for LILI-II

When output conditions of F_C are 01, 10, 11 and 00 the content of $LFSR_D$ is shifted by one stage, two stages, three stages and four stages respectively in one clock cycle.

The shift register $LFSR_D$ is designed in such a way that, it is capable to shift its content by one to four stages in one clock cycle depending on output of function F_C. Proposed architecture uses $LFSR_D$ having four sets of 127 bit shift registers, named as $LFSR_{D1}$, $LFSR_{D2}$, $LFSR_{D3}$ and $LFSR_{D4}$. $LFSR_{D1}$ to $LFSR_{D4}$ are used for shifting the original content of $LFSR_D$ by one to four bits respectively in single clock cycle depending on output from function F_C. Shifting of $LFSR_D$ by one to four bits is shown in Fig. 3.

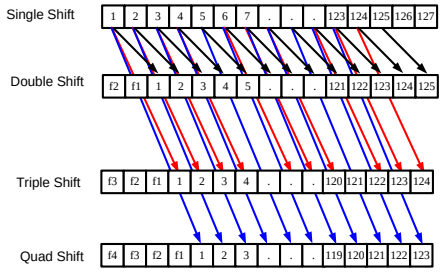

Fig. 3: Shifting of $LFSR_D$ in single clock cycle

Proposed architecture uses multiplexer having two inputs for loading initial values into $LFSR_D$ stages. When "Initial Data Load $LFSR_D$" terminal of $LFSR_{D1}$ is '1' then initial values are loaded into $LFSR_{D1}$ stages. During normal operation this terminal is '0' and provides normal shifting operation. It also consists of multiplexer having four inputs for loading shifted value of $LFSR_{D1}$ to $LFSR_{D4}$ in $LFSR_{D1}$ required for next clock cycle. When select line of multiplexer is 01, 10, 11 or 00, the one bit shifted content of $LFSR_{D1}$, $LFSR_{D2}$, $LFSR_{D3}$ or $LFSR_{D4}$ is loaded into $LFSR_{D1}$.

Architecture presented in this paper uses four set of feedback functions as f1, f2, f3 and f4 as shown in Fig. 4. Feedback function f1 to f4 are useful for shifting of $LFSR_D$ by one to four stages respectively in single clock cycle.

The functions from f1 to f4 are calculated by using current stage of $LFSR_{D1}$ in each clock cycle. For example, first input used by feedback function f1 is X^{127}, same input for feedback functions f2, f3 and f4 is taken from X^{126}, X^{125} and X^{124} respectively. Similarly, the last input for feedback function f1 is X^1, same input for feedback function f2, f3 and f4 is f1, f2 and f3 respectively. This is because f1 is the new value taken as input after first shift and similar for third and fourth shift.

In the Proposed architecture, twelve stages output (0, 1, 3, 7, 12, 20, 30, 44, 65, 80, 96, and 122) of $LFSR_D$ required for function F_D are connected by using four input multiplexers. When output of function F_C is 01, 10, 11 or 00 then $LFSR_{D1}$, $LFSR_{D2}$, $LFSR_{D3}$ and $LFSR_{D4}$ output is selected at multiplexer respectively and delivered to function F_D.

At the same time selected value of $LFSR_D$ (1, 2, 3 or 4) in the current clock cycle needed to be loaded in $LFSR_{D1}$ in next clock cycle, to shift these contents by one to four stages depending on value of function F_C in current clock cycle. The operation mentioned above is performed by storing value of function F_C in the FFs and used in the next clock cycle as shown in Fig. 2.

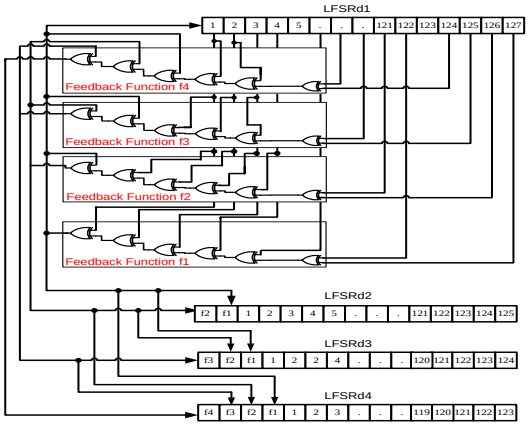

Fig. 4: Feedback function of $LFSR_D$

The function F_D of data generation subsystem is implemented as lookup table in ROM having twelve address lines and each location consists of one bit data. Depending on input address the one bit key stream is available at output.

4. RESULTS AND DISCUSSION

Proposed architecture uses compact architecture for function F_C as compared to existing implementations [3], [5], [6]. Specification of LILI-II [5] and previous implementations [6] uses full adder for implementation of function F_C. We used compact architecture for function F_C as shown by equations (04) and (05).

Previous implementation [6] uses clock pulses mechanism and logic gate AND to shift the content of $LFSR_D$ by one to four states depending on value of function F_C. If $LFSR_{D4}$ of this architecture is clocked four times, it is providing four outputs, but as per specification expected is one. Similarly, this architecture is not using all the values generated by function F_C. For example, when content of $LFSR_D$ is to be shifted by four states, then during four clock cycles $LFSR_D$ is not accepting any value from function F_C.

Implementation [6] do not uses proper loading arrangement after each cycle. For example, if the content of $LFSR_{D4}$ is shifted by four stages, same content need to be shifted in next clock cycle depending on value of function F_C. But previous implementation uses initial contents for shifting of $LFSR_{D1}$, $LFSR_{D2}$ or $LFSR_{D3}$. This may be the serious security issue.

Reconfiguration of LFSR with different feedback functions is one of the good concepts proposed in the previous implementation [6]. This concept may be used for providing different security levels. Feedback logic of proposed architecture is not reconfigurable. Architecture proposed in this paper is capable to shift content of $LFSR_D$ by one to four states in single clock cycle in place of using multiple clock cycles. Similarly, proposed implementation also has facility to load the shifted content in the next cycle and shift this content by requisite number of states depending on value of function F_C.

Proposed implementation is coded by using VHDL language and uses XILINX xc4vlx60 FPGA device with package ff1148. Comparison of synthesis results between previous [6] and proposed implementation is shown in Table 2.

Comparison of simulated result shows that proposed architecture is compact as compared to previous implementations. This architecture achieves maximum throughput of 224.7 Mbps at frequency of 224.7 Mhz. Achieved throughput is more than basic architecture but less than higher versions of FPGA proposed by other authors.

Table 2. Comparisons of hardware resources used

Logic term	Utilized in proposed implementation	Utilized in previous [6] implementation
Function Generators	386	938
Number of Slices	514	469
Number of Slice Flip Flops	261	693
Number of bonded IOBs	260	391
Block RAM	0	1

5. CONCLUSION

Proposed architecture minimizes the hardware requirements as compared to previous implementations and achieves throughput of 224.7 Mbps at clock frequency of 224.7 MHz. This architecture is capable of shifting $LFSR_D$ content by one to four states in single clock cycle, without losing any states from function F_C. This architecture has facility to load shifted content in next clock cycle required for further shifting. It may be more secure than previous architectures.

6. REFERENCES

[1] Noman, A. Al, Sidek, R. M., & Ramli, A. R., "Hardware Implementation of RC4A Stream Cipher", International Journal of Cryptology Research,2009, 1(2), 225–233.

[2] Galanis, M., Kitsos, P., Kostopoulos, G., Sklavos, N., & Goutis, C., "Comparison of the Hardware Implementation of Stream Ciphers", The International Arab Journal of Information Technology, 2005, 2(4), 267–274.

[3] Philippe Leglise, Francois-Xavier Standaert, Gael Rouvroy, Jean- Jacques Quisquater, "Efficient Implementation of Recent Stream Ciphers on Reconfigurable Hardware Devices", In Proc. of the 26th Symposium on Information Theory in the Benelux, May 19th-May 20th, 2005, Brussels, Belgium.

[4] A. Clark, E. Dawson, J. Fuller, J. Golic, H-J. Lee, William Millan, S-J. Moon, and L. Simpson, "The LILI-128 keystream generator", In Selected Areas in Cryptography—SAC 2000, volume 2012 of Lecture Notes in Computer Science. Springer-Verlag, 2000.

[5] Clark, Andrew J. and Fuller, Joanne E. and Golic, Jovan Dj and Dawson, Edward P. and Lee, H-J and Millan, William L. and Moon, Sang-Jae and Simpson, L. R., "The LILI-II Keystream Generator", In Information Security and Privacy, 2002, pp. 25–39.

[6] Kitsos, P., Sklavos, N., & Koufopavlou, O. "A High-Speed Hardware Implementation of the LILI-II Keystream Generator", In 5th International Symposium on Communications Systems, Networks & Digital Signal Processing , 2006, pp. 6–9.

[7] Kircanski, A., & Youssef, A. M., "On the Sliding Property of SNOW 3G and SNOW 2.0", In SAS 2012, LNCS 7707, 2013, Springer-Verlag (Vol. 5, pp. 119–134).

[8] ETSI/SAGE: 'Document 2: Specification of the 3GPP Confidentiality and Integrity Algorithms 128-EEA3 & 128-EUA3: ZUC specification', Version 1.4, 2010. H. Englund and T. Johnson, "A New Distinguisher for Clock Controlled Stream Ciphers", Fast Software Encryption 2005, LNCS Vol 3557, pages 181-195, Springer, 2005.