ISCC'2005
Isfahan University of Technology
7-8 September 2005

Iranian Society of Cryptology

Rabbit Data Streaming:Architecture & FPGA Implementation

N. Sklavos [I], M. Shahraki [II], and P. Souras [I]

[I]: Electrical & Computer Engineering Dept., University of Patras
nsklavos@ee.upatras.gr

[II]: Electrical Engineering Dept., University of Sistan & Baluchistan
shahraki@rediffmail.com

Abstract

The special needs for cryptography, of both wired and wireless communications, have attracted the researchers' major interest in the design of new encryption algorithms. Rabbit cipher is a latest developed stream cipher, with special features for software performance. Of course, new algorithms are intended to perform efficiently for both software and hardware applications. In this paper, an architecture and the VLSI implementation of Rabbit cipher are proposed. An FPGA device has been used, as the integration platform of the introduced system architecture. Furthermore, both the implementation cost and the performance evaluation concerning the hardware integration are exploited. Finally, comparisons regarding other well-known stream ciphers implementations are given.

Keywords: Rabbit Stream Cipher, Performance Evaluation, Hardware Implementation, FPGA, Chaotic Maps

1. Introduction

Security is a primary requirement of any wired and wireless communication [1]. Encryption algorithms are meant to provide secure communications applications. New encryption algorithms have to operate efficiently in a variety of current and future applications, doing different encryption tasks [2], [3]. All hardware implementations have to be efficient, with the minimum allocated logic gates. This means simplicity in cipher's architectures with enough "clever" data transformation components. A communication protocol implementation, demands low power devices and fast computation components. The ciphers of the near future have to be key agile. Many applications need a small amount of text to be encrypted with keys that are frequently changed. Many well know applications, like IPsec, use this way of algorithm's operation [1].

In order the special requirements for security to be satisfied at a great manner new encryption algorithms have been lately developed [4], [5]. Rabbit stream cipher is a modern stream cipher introduced in 2003 [5]. This cipher has been mainly intended to be used in software applications. However, it is proven very interesting the implementation of it, in hardware platforms.

In this paper, an architecture and the VLSI implementation are presented for the Rabbit stream cipher. According to our knowledge this is the first work on hardware integration, concerning this certain algorithm implementation. An FPGA device has been used as a hardware integration platform for the proposed system architecture. Results concerning both performance and hardware implementation cost are also presented. The synthesis results prove that the proposed system architecture achieves high-speed performance, with low implementation cost at the same time. Finally, comparisons with implementations of other well-known stream ciphers are also given.

This work is organized as follows: in Section 2, the philosophy behind the design of Rabbit stream cipher is described. In Section 3, the proposed system architecture for the hardware implementation is presented. In the next Section 4, the VLSI synthesis results of the proposed architecture are given. In addition, comparisons with other ciphers implementations are presented. Finally, conclusions are discussed in Section 5.

2. Design Philosophy of Rabbit Stream Cipher

Secret key cryptographic systems can be categorized into either block or stream ciphers. Block ciphers are memoryless algorithms that permute N-bit blocks of plaintext data under the influence of the secret key and generate N-bit blocks of encrypted data. Stream ciphers contain internal states and typically operate serially by generating a stream of pseudo-random key bits, the keystream (stream ciphers are also called keystream generators).

One of the latest published stream ciphers is Rabbit. The cipher, introduced by M. Boaesgaard and M. Vesterager on 2003, is inspired by the complex behavior of real valued chaotic maps [5]. Chaotic systems have attracted the cryptographers' interest and have been used in security for different applications before [6], [7]. Although chaotic systems exhibit random like behavior, they are not cryptographically secure in their discretized form [8], [9].

The attention of the cipher introducers was turned to provide an algorithm that takes the advantages of the random-like properties, of real chaotic maps. However, their major intention was to ensure the secure optimal cryptographic properties when discretizing these maps. More analytically, the design was initiated by constructing a chaotic system of coupled non-linear maps. This system was then restricted to be fixed point valued. This means that each variable is represented by an integer type number, where a virtual decimal point is introduced manually. This ensured reproducibility, and made the system analyzable from the binary point of view using well-known cryptographic techniques [10]. The analysis gave reason to some systematic improvements of the equation system, some of which were strictly binary in nature, e.g. adoption of rotations and the XOR operator. Those changes were advantageous for the complexity of the binary functions as well as the performance.

The specific design goals of Rabbit were as follows:

- *Security: the cipher should justify a key size of 128-bit for encrypting up to 2^{64} bytes of plaintext;*
- *Speed: it should be faster than commonly used ciphers.*

The operation process of rabbit is based on a 128-bit secret key, while for each iteration an output block of 128-bit pseudo-random bits is generated. The value of this block is the output of the internal state bits combination. The data are encrypted/decrypted via XORing the pseudo-random bits with the

plaintext/ciphertext. The size of the internal state is 513-bit divided between eight 32-bit state X_j and Counter variables C_j variables. The last bit is used as a counter carry bit. The eight state variables are updated by eight coupled non-linear integer valued functions. The counters secure a lower bound on the period length for the state variables.

The cryptanalysis of Rabbit resulted in the following [5]. To investigate the possibilities for Divide-and-Conquer and Guess-and-Determine types of attacks, an algebraic analysis was performed with special attention on the non-linear parts of the next-state function, as they are the main sources for mixing input bits. No such attacks better than exhaustive key search were found [5]. To verify the resistance against correlation and distinguishing types of attacks, a correlation analysis was performed by calculating the Walsh-Hadamard spectra of the nonlinear parts. Based on the correlation analysis the cipher's introducers do not believe there exists a correlation-type attack, which requires less work than exhaustive key search for an output sequence shorter than 2^{64} bytes [5].

3. Proposed System Architecture

The proposed architecture for the hardware implementation of the studied stream cipher is illustrated in Fig. 1. It consists of a number of separate cores, which are used for the appropriate data transformations according to Rabbit cipher's specifications [5]. The main components of the following architecture are: Counter and Carry Units, G Transformation, Next State Function, Extraction Scheme, Constants Unit and the Data Transformation Round. In addition, an I/O interface Unit is used for the appropriate communication of the proposed system with the external environment.

The basic data transformations are based on 32-bit vectors (n=32-bit), according to cipher's basic functionality [5]. The n-bit data buses are used for the internal data transmission between the system's units.

The internal data state of this stream cipher consists of 512 plus one – bits. The first 256-bit vector of this state is used for the data of variables (x[0],…,x[7]), while the second 256-bit vector is dedicated to counters data (c[0],…,c[7]). In other words the two fundamental data vectors (2x256-bit) are divided to 8 internal states of 32-bit. The last one bit is used a carry-bit. Rabbit operation is also based on a 128-bit key. The key K(128-bit) is divided into eight subkeys (k_0=K[15..0],…,k_7=K[127112]).

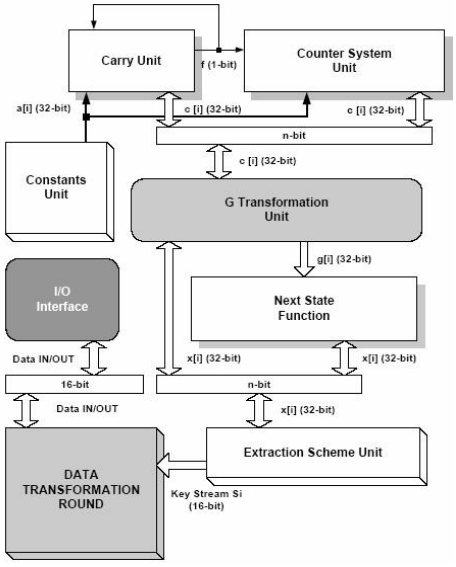

Figure 1: Proposed System Architecture

Data variables (x[0],…,x[7]) are used as the basic data vectors. Their initial values are defined during initialization process as a function of the subkeys values. For example: $x[0]=k_1 \,||\, k_0$. During initialization the counter values are defined also in a similar way, based on the same eight subkeys.

G Transformation Unit, which is shown in Fig. 2, is one of the fundamental units of the system architecture (Fig. 1).

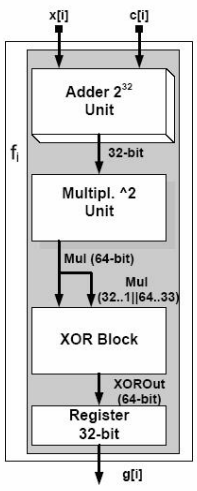

Figure 2: Proposed Architecture of G Transformation Unit

The operation that this unit performs is based on function f_i. This basic function is described by the following equation:

$$f_i = [(X_{j,i}+C_{j,i})^2 \; XOR \; \{(X_{j,i}+C_{j,i})^2 >> 32\}] \; mod \; 2^{32} \qquad (1)$$

where: x,y are 32-bit vectors, and >> denote right bit-wise rotation.

Eight variables g[i] values have to be generated in order the Next State Function to start operation. It has to be mentioned that G Transformation Unit and Next State Function are processing together as a coupled system. The function of this system could be described schematically by the graph of the next figure according to Rabbit specifications [5], where c(i) are 32-bit counters, and << denote left bit-wise rotation:

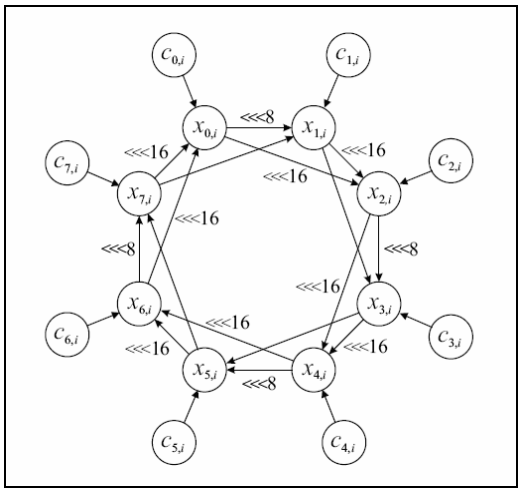

Figure 3: System's Operation Graph

Next State Function is the core of the Rabbit stream cipher. The iterations of the proposed architecture (Fig. 1) are based on this unit data transformations. In Fig. 4, the proposed architecture for the Next State Function is presented. It is a based on a chain-structure, which consists of eight similar building cells. Each basic cell consists of two adder units and two left bit-wise shift registers.

The operation of the G Transformation Unit is based on the usage of counters. These certain counter values c[i] are incremented basically by the Counter System as described bellow.

Counter System Unit defines the counter values based on additions each time, between the used constant a[i], the used carry f, and the previous value of the counter c[i-1]. In order the described additions to be performed, modulo adders 2^{32} are used. The used carry f, each time is derived from the Carry Unit.

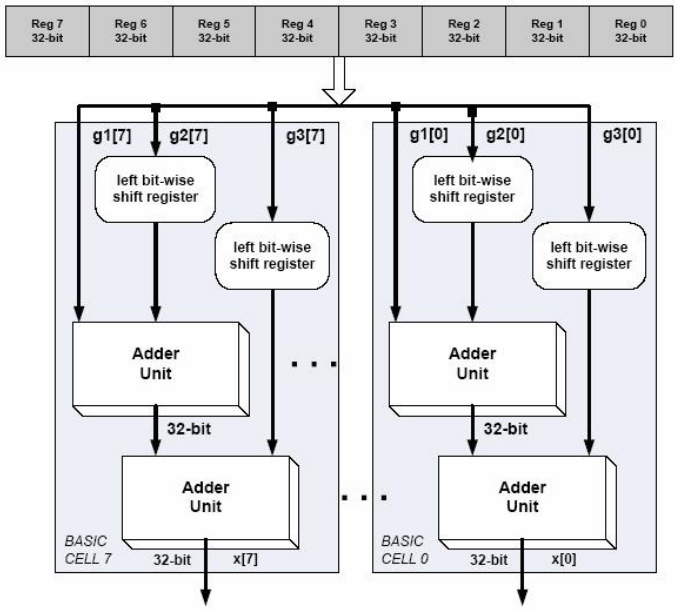

Figure 4: Proposed Architecture for Next State Function

Carry Unit (Fig. 1) is used in order to define the carry-bit φ[i], each time for the Counter System Unit operation. The appropriate values of these carry-bits are determined from the following equation:

$$\varphi[i], j+1 = \begin{cases} 1 & \text{if } c[0],j + a[0] + \varphi([7], i) \geq 2^{32} \wedge j=0 \\ 1 & \text{if } c[i],j + a[i] + \varphi([j-1], i+1) \geq 2^{32} \wedge j>0 \\ 0 & \text{Otherwise} \end{cases} \quad (2)$$

This unit is implemented in a hardware device with a value defined circuit, based on a multiplexer and a comparator. According to the value of the comparator output, the value zero or one is defined for the output of the Counter Carry Unit (Fig. 5).

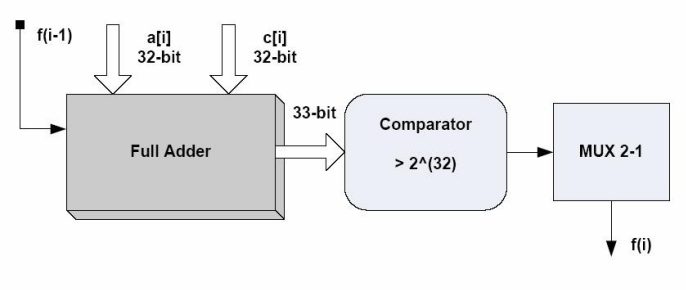

Figure 5: Carry Unit Hardware Design

Constants Unit contains the constants values for the desirable data modifications. Rabbit cipher operation is based on 8x32-bit variables [5]. According to our research the values of these eight variables could be derived from a basic vector (Hex: D34D34D34) and a shift register of Flip-Flops (F/Fs). In this way, an area reduction of 6x32-bit registers is achieved. The basic vector is stored in the Initial Vector Register as it is shown in Fig. 6.

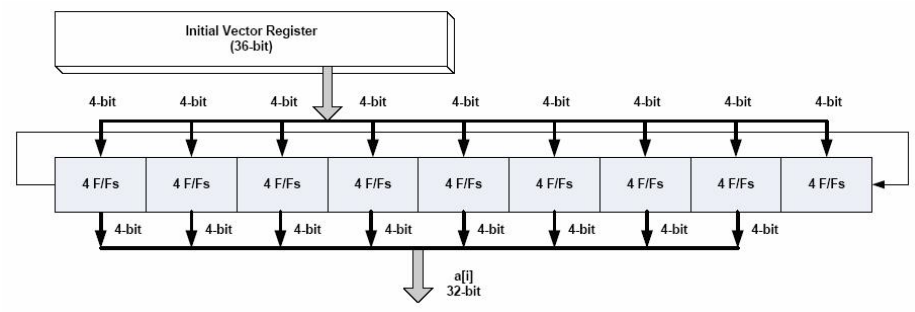

Figure 6: Constants Unit Architecture

After a full iteration the Extraction Scheme generates an output of 128-bit based on a XORing process and a data mixing of the input bits. The proposed architecture for the Extraction Scheme Unit is illustrated in the next figure:

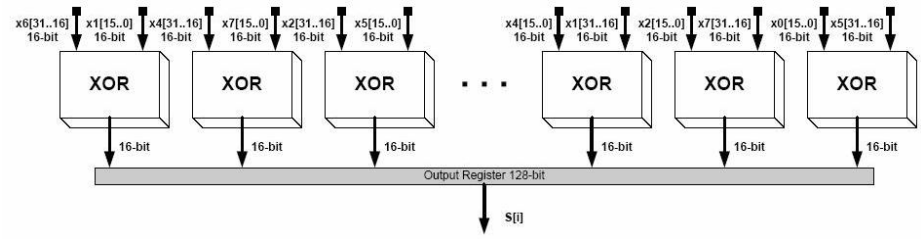

Figure 7: Architecture of the Extraction Scheme Unit

The Data Transformation Round (Fig. 1) is used to implement the Encryption /Decryption Scheme of the stream cipher. It consists of a chain of eight XOR logic gates with 16-bit I/Os. The extracted bits are XOR'ed with the plaintext/cipher text in order encryption/decryption to be performed. A data vector of 128-bit are modified in Data Transformation Round, during one clock cycle according to:

$$C_i = P_i \; XOR \; S_i \quad (Encryption) \quad , \quad P_i = C_i \; XOR \; S_i \quad (Decryption) \tag{3}$$

where C_i and P_i denote the i^{th} ciphertext and plaintext blocks respectively

4. VLSI Synthesis Results & Comparisons

In general, for hardware integrations two different type of devices are used mainly: Application Specific Integrated Circuits (ASIC) and Field Programmable Gate Arrays (FPGA). The performance characteristics of both ASICs and FPGAs are substantially different compared with a general-purpose microprocessor. ASICs and FPGAs have the advantage that can use all the resources for pipelining data transformation, or parallel processing. On the other hand, the internal structure of the microprocessors functional units limits the parallel processing and pipelining transformation. In addition the instruction parallelism level is a factor of great importance that must be taken under consideration for microprocessor performance. Furthermore, the hardware devices of these types can operate on arbitrary size of words, in contrast with processors, that operate only on fixed-sized words.

ASICs are in general far more expensive devices due to time consuming and high cost fabrication procedure, which is done by expertise industry departments. FPGAs can be bought from anyone, since they are enough cheaper and can be programmed or reconfigured by the designers/researchers. FPGAs have the major advantage that can perform a completely different task/function after simple designers' reconfiguration. ASICs performance is tight and cannot be modified after the chip's fabrication. The offered reconfiguration has a speed penalty in these devices. ASICs have higher speed performance in comparison with FPGAs. This is due to the fact that the reconfiguration in FPGAs causes delays, introduced by the dedicated circuit's parts needed to reconfiguration. In general, any implemented system of digital logic in an FPGA, is slower than the ASIC implementation of the same system.

The proposed system architecture has been implemented by using an FPGA hardware device. The system architecture has been captured by using VHDL and has been synthesized placed and routed using XILINX FPGA Virtex Device [11]. The synthesis results for the proposed implementation are illustrated in the next Table 1.

Table 1: FPGA Implementation Synthesis Results

FPGA DEVICE	XILINX (v300efg456)	
COVERED AREA	**Used / Available**	**Utilization**
IOs	126 / 312	40 %
FGs	2150 / 3072	70 %
CLB Slices	1731 / 6912	34 %
DFFs or Latches	6928 / 7116	97 %
Frequency (F)	55 MHz	

where: D Flip-Flops (DFFs), Configurable Logic Blocks (CLB), and Function Generators (FG)

The basic aim of Rabbit's introducers was to design a secure stream cipher which is highly efficient mainly in software [5]. The performance of Rabbit has been measured for two different types of processors. The encryption/decryption speed of Rabbit is equal to 3.7 clock cycles per byte on a Pentium III processor [5]. For an ARM7 processor the measured performance was 10.5 clock cycles per byte [5].

Rabbit stream cipher performs better in hardware, than in software platforms. According to our research Rabbit stream cipher operates efficiently in applications based on hardware devices. The achieved 55 MHz operation frequency of the proposed FPGA implementation is quite high. Based on this frequency

the throughput value of the proposed integration is equal to 586 Mbps. This value is quite high and so the proposed implementation could applied efficiently in many security applications, such as wireless protocols and networks crypto engines.

According to our knowledge, this is the first work for integrating Rabbit using a hardware device. In order to have a fair and detailed comparison of the proposed implementation, the proposed system is compared with VLSI integrations of other well-known stream ciphers [12], [13], [14]. The results of these comparisons are illustrated in the next Table 2. Although, it has to be mentioned that the following implementation refer to different FPGA families, with different integrated device technology each time.

Table 2: FPGA Implementation Synthesis Results

STREAM CIPHER	PERFORMANCE (Mbps)	FREQUENCY (MHz)	AREA RESOURCES (CLBs)
RC4 [12]	*2.22*	*17.8*	*255*
A5/1 [13]	*188*	*188*	*32*
W7 [13]	*768*	*96*	*608*
Eo [14]	*189*	*189*	*895*
Helix [14]	*418*	*32*	*1024*
Rabbit (Proposed)	*586*	*55*	*2331*

The different design philosophies of the compared ciphers, do not allow making safe comparison's results. Although the above results are illustrated in order to have a complete point of view, regarding the implementations aspects. Finally, the Area-Delay product and the Performance/Area ratio are illustrated in the following Fig. 8, for the above implementations.

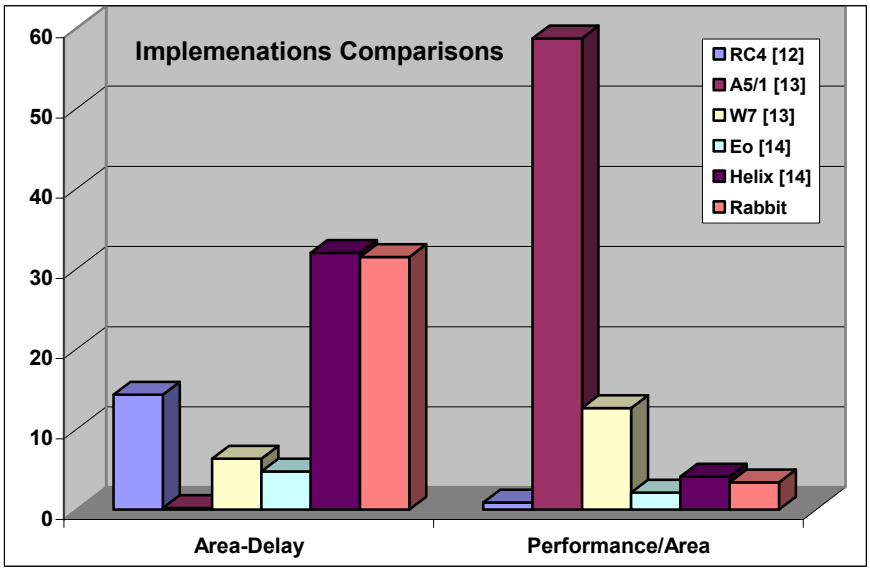

Figure 8: Area-Delay and Performance/Area Graph

5. Conclusions

Security is a primary requirement of any wired and wireless communication [1]. Encryption algorithms are meant to provide secure communications applications. In order the special requirements for security to be satisfied at a great manner, new encryption algorithms have been lately developed. Rabbit stream cipher is a modern stream cipher, which has been mainly intended to be used in software applications. Although, it is proven very interesting the implementation of it, in hardware platforms.

This work focuses on the implementation of Rabbit stream cipher. Especially an architecture and a VLSI implementation have been proposed for this algorithm integration. According to our knowledge this is the first work on hardware integration, concerning Rabbit implementation. An FPGA device has been used as a hardware integration platform. The synthesis results show that the proposed system architecture is comparative in respect to area resources, performance and time delay, with other well-known stream ciphers implementations.

References

[1] B. Schneier, "Applied Cryptography–Protocols, Algorithms and Source Code in C", Second Edition, John Wiley and Sons, New York, 1996.

[2] N. Sklavos, and O. Koufopavlou, “Mobile Communications World: Security Implementations Aspects – A State of the Art”, CSJM Journal: Institute of Mathematics and Computer Science, Vol. 11, Number 2 (32), pp. 168-187, 2003.

[3] N. Sklavos, P. Kitsos, E. Alexopoulos, and O. Koufopavlou, “Open Mobile Alliance (OMA) Security Layer: Architecture Implementation and Performance Evaluation of the Integrity Unit”, Journal of New Generation Computing: Computing Paradigms and Computational Intelligence, Springer-Verlag, Vol. 23, No 1, pp. 77-100, 2005.

[4] N. Sklavos and O. Koufopavlou, “Architectures and VLSI Implementations of the AES-Proposal Rijndael”, IEEE Transactions on Computers, Vol. 51, Issue 12, pp. 1454-1459, 2002.

[5] M. Boesgaard, M. Vesterager, T. Pedersen, J. Christiansen, and O. Scavenius, “Rabbit: A New High-Performance Stream Cipher”, proceedings of Fast Software Encryption (FSE’03), LNCS: 2887, pp. 307-329, Springer-Verlag, 2003.

[6] S. Wolfram, “Cryptography with Cellular Automata”, proceedings of Crypto’85, pp. 429-432, 1985.

[7] G. Jakimoski and L. Kocarev “Chaos and Cryptography: Block Encryption Ciphers Based on Chaotic Maps”, IEEE Transactions on Circuits and Systems-1: Fundamental Theory and Applications, Vol. 48, No 2, pp. 163-169, 2001.

[8] T. Habutso, Y. Nishio, I. Sasase and S. Mori “A secret key cryptosystem by iterating a chaotic map”, Proceedings of the EUROCRYPT‘91, Springer-Berlin, pp.127-140, 1991.

[9] E. Biham “Cryptoanalysis of the chaotic-map cryptosystem suggested at EUROCRYPT-91”, Proceedings of the EUROCRYPT‘91, Springer-Berlin, pp.532-534, 1991.

[10] A. Menezes, P. Oorschot and S. Vanstone, "Handbook of Applied Cryptography", CRC Press LLC, 1997.

[11] Xilinx, San Jose, California, USA, “Virtex, 2.5 V Field Programmable Gate Arrays”, www.xilinx.com, 2005.

[12] P. Hamalainen, M. Hannikainen, T. Hamalainen and J. Saarinen, “Hardware Implementation of the Improved WEP and RC4 Encryption Algorithms for Wireless Terminals”, The European Signal Processing Conference (EUSIPCO'2000), September 5-8, 2000, Tampere, Finland, pp. 2289-2292.

[13] G. Kostopoulos, et. al., “VLSI Implementation of GSM Security: A5/1 and W7 Ciphers”, proceedings of the IEEE Workshop on Wireless Circuits and Systems (WoWCAS'04), Canada, May 21-22, 2004.

[14] M.D. Galanis, et. al., “Comparison of the Hardware Implementation of Stream Ciphers”, IAJIT Journal, Colleges of Computer and Information Society, Vol. 3, Number 1, pp. 1-8, 2006.