Journal of Theoretical and Applied Information Technology
15th April 2017. Vol.95. No 7

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

FPGA-BASED HIGH SPEED BLOWFISH ALGORITHM

[1]SOUFIANE OUKILI, [2]SEDDIK BRI
[1,2]Materials and Instrumentation (MIN), High School of Technology,
Moulay Ismail University, 50040 Meknes, Morocco
E-mail: [1]soufiane.oukili@gmail.com, [2]seddikbri@gmail.com

ABSTRACT

Nowadays, security has become essential element of all systems and applications, due to the rapid growth of information and communication technology. In this context, high speed and high volume secure communications have been a high priority and challenging research area in both fields of mathematics and engineering. In this paper, we present high speed hardware architecture of Blowfish cryptographic algorithm. We had used pipeline technique to allow a parallel processing in order to obtain high throughput. In addition, 5-stage pipeline round of Blowfish algorithm is proposed to increase the speed and the maximum operating frequency. Furthermore, the S-box tables of each round of the algorithm had been implemented in block RAMs to allow parallel data encryption. The proposed design had been successfully implemented in FPGA devices. It improves data throughput by 104%.

Keywords: *Security, Cryptography, Blowfish, Pipeline, High speed; FPGA*

1. INTRODUCTION

The astounding growth of the Internet and computer systems in the last century, have meant that the need for effective security and reliability of data communication, processing and storage is greater than ever. As the fundamental of security information, cryptographic algorithm plays a very important role in the domain of information security. An encryption algorithm, or cipher, is a means of transforming plaintext into ciphertext under the control of a secret or public key. Secret key algorithm (symmetric cryptography) uses the same cryptographic keys for both encryption of plaintext and decryption of ciphertext. Public key algorithm (asymmetric cryptography) uses pairs of keys: public key for encryption with private key for decryption [1-2]. The Data Encryption Standard (DES) was the first modern symmetric key algorithm used for encryption and decryption of digital data. It had been developed in the 1970s at IBM and adopted as a Federal Information Processing Standard (FIPS) by the National Institute of Standards and Technology (NIST) in 1977 [3]. In 1998, the NIST announced a competition for a new encryption algorithm that would be used for protecting sensitive information. This algorithm would replace DES, which was not resistant to known attacks because of the short key length. After all reviews, NIST had chosen an algorithm known as Rijndael. It was developed by two Belgian cryptographers: Dr. Joan Daemen and Dr. Vincent Rijmen. In November 2001, Advanced Encryption Standard (standardized version of Rijndael) becomes a FIPS standard (FIPS-197) [4-5]. Blowfish is a symmetric block cipher algorithm. It had been designed by Bruce Schneier in 1993. It takes 64-bit plaintext and variable-length key, from 32 bits to 448 bits, as inputs and 64-bit ciphertext as an output [6-7]. Blowfish had been identified as a powerful cryptographic algorithm since it can satisfy two basic requirements: high immunity to attacks and relative low algorithm complexity [8]. It is unpatented and no license is required, available free for all uses. Besides, it is suitable and efficient for hardware implementation [9].

There are software and hardware approaches to implement cryptographic Blowfish algorithm. Hardware implementation provides greater physical security and higher speed as compared to software implementation [10]. Because of the increasing requirement to implement cryptographic algorithms in fast rising high-speed network applications combined with physical security, hardware implementation becomes essential.

In this paper, we present high speed hardware architecture and implementation of Blowfish algorithm. We had used pipeline strategy. It modifies the critical path by increasing possible frequency of clock cycle. It consists in parallelizing

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

the data inputs and outputs with the processing by inserting registers. 5-stage pipeline round of Blowfish algorithm is proposed to increase the speed and the maximum operating frequency. In order to reduce more the path delay, we had divided the addition modulo 232 into two additions modulo 216 and each one is executed separately and in a parallel manner by inserting registers in appropriate places. This will further increase the throughput of the algorithm. Furthermore, in each encryption round of the algorithm, S-box tables had been implemented in block RAMs to allow parallel data encryption. Our proposed design was implemented on Xilinx Virtex-5 and Virtex-6 FPGA devices. The FPGAs offer the advantage of hardware speed and software flexibility and programmability.

This paper is structured as follows. Section 2 presents a brief background of the Blowfish algorithm. Section 3 gives the relevant works of various authors reported in the literature. Our proposed Blowfish design is presented in Section 4. Section 5 provides results and comparison between our implementations and different reported ones. Finally, conclusion and references are given respectively.

2. BACKGROUND OF BLOWFISH ALGORITHM

Blowfish is a symmetric block cipher. It has a fixed 64-bit data block size and a variable secret key range from 32 bits to 448 bits. The algorithm consists of two parts: key expansion and data encryption. The key expansion converts a key of at most 448 bits into several subkey arrays totaling 4168 bytes. The data encryption occurs via a 16-round Feistel network. Each round consists of a key-dependent permutation, and a key- and data-dependent substitution. All operations are XORs and additions on 32-bit words. The only additional operations are four indexed array data lookups per round [6].

2.1 Key expansion

Blowfish uses a large number of subkeys (eighteen 32-bit P-array and four 32-bit S-boxes with 256 entries each). These subkeys must be calculated before any data encryption or decryption using the Blowfish algorithm. The method is as following:

1: Initialize first the P-array and then the four S-boxes with hexadecimal digits pi.

2: XOR P1 with the first 32-bits of the key, XOR P2 with the second 32-bits of the key, and so on for all bits of the key (up to P18).

3: Encrypt the all zero string with the blowfish algorithm, using the keys described in steps (1) and (2).

4: Replace P1 and P2 with the output of step (3).

5: Encrypt the output of step (3) using the Blowfish algorithm with the modified keys.

6: Replace P3 and P4 with the output of step (5).

7: Continue the process, replacing all elements of the P-array, and then all four S-boxes in order, with the output of the continuously changing Blowfish algorithm.

2.2 Data encryption

As mentioned previously, Blowfish is a Feistel network consisting of 16 rounds, as shown in figure 1. The inputs are 64-bit plaintext and 18 P-array sybkeys (32 bits). The output is a ciphertext (64 bits). The Blowfish algorithm is described in algorithm 1.

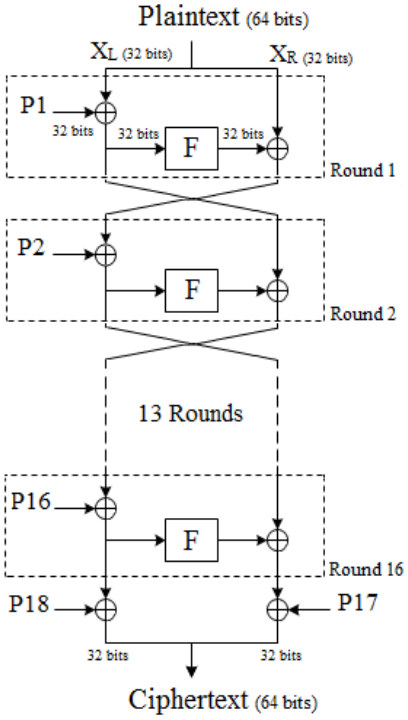

Figure 1. Block diagram of Blowfish algorithm

Journal of Theoretical and Applied Information Technology
15th April 2017. Vol.95. No 7

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Algorithm1. Blowfish

Inputs: Plaintext (64 bits) and P1, P2, …, P18
Output: Ciphertext (64 bits)
1 Divide plaintext into two 32-bit halves: XL, XR
2 **For** i = 1 to 16:
3 Calculate XL = XL XOR Pi
4 Calculate XR = F(XL) XOR XR
5 Swap XL and XR (undo the last swap)
6 **end for**
7 Calculate XR = XR XOR P17
8 Calculate XL = XL XOR P18
9 Recombine XL and XR (ciphertext)
10 **return** ciphertext

Function F is calculated by Eqs. (1). It divides XL into four eight-bit quarters: a, b, c, and d. These quarters are used as input to the S-boxes. The outputs are added (modulo 232) and XORed to produce the final 32-bit output. This is shown in figure 2. Decryption is exactly same as encryption, except that P1, P2 ... P18 are used in the reverse order.

$$F(X_L) = ((S_{1,a} + S_{2,b} \bmod 2^{32}) \text{ XOR } S_{3,c}) + S_{4,d} \bmod 2^{32} \quad (1)$$

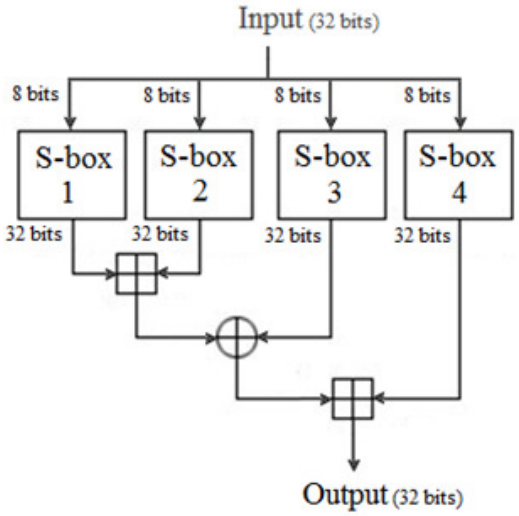

Figure 2. Function F

2.3 Cryptanalysis

Cryptanalysis refers to the process of illegally attempting to recover the plaintext (or the key) that corresponds to a particular ciphertext. Full-round version of Blowfish algorithm is invulnerable against cryptanalysis, to date. Numerous attack schemes have been proposed to break the cryptographic system and extract secret information, but none succeeded.

John Kesley could only break 3-round of Blowfish and his cryptanalysis cannot be extended beyond 3 rounds. Serge Vaudenay examined a simplified variant of Blowfish, with the S-boxes known and not key-dependent. For this variant, a differential attack can recover the P-array with 2^{8r+1}chosen plaintexts, where r is the number of rounds. This attack is impractical in reality and does not work against 8-round Blowfish and higher, since more plaintext is required than can possibly be generated with a 64-bit block cipher [8]. In 1996, Vincent Rijmen proposed a promising attack in his doctoral dissertation, but it can only break 4 rounds of Blowfish and no more [11].

3. RELATED WORKS

Several different methods had been presented in the literature to implement Blowfish algorithm. Cody et al. [12] presented robust implementation of Blowfish in hardware. The design utilizes the simplicity of the algorithm to create a relatively straightforward implementation and uses the core-slow library for worst-case scenario analysis. Kurniawan et al. [13] presented the performance of blowfish algorithm with total time taken for encryption, avalanche effect and throughput from multiple testing scenarios as the parameters. The Blowfish algorithm was implemented on FPGA using VHDL language. The results show that reducing the round of Feistel (F) reduce the total encryption time, give greater throughput and not affect avalanche effect significantly. Kumara and Benakop [14] proposed four different implementations of Blowfish algorithm and analyzed the performance of it with and without Wave Dynamic Differential Logic (WDDL) style to provide security against Differential Power Analysis (DPA) attack. Ahmad and Ismail [15] proposed an improved power-throughput Blowfish algorithm. It was designed with 128-bit block size, which is comprised of parallel blocks of 64-bit inputs that were simultaneously executed. This enables the throughput to be maximized. The parallel blocks share the same S-boxes that are used for Feistel function (F) and were stored in BRAMs. Guerrero and Noras [16] presented fast Blowfish encryption in hardware with a throughput of 1032 Mbps. Sudarshan et al. [17] presented flexible architecture for Blowfish algorithm called Dynamic reconfiguration, Replication, Inner loop pipeline, Loop folding architecture abbreviated as DRIL. DRIL Architecture aims at efficient utilization of hardware through replication and loop folding, higher throughput through replication and inner

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

loop pipeline, flexibility through dynamic reconfiguration and replication. Rafidah et al. [18] presented a development of an improved power-throughput Blowfish algorithm as an alternative security algorithm. The proposed memory-based method is used to optimize the performance of Blowfish. The performance is analyzed in terms of its architecture, throughput, and power consumption. Kumar and Baskaran [19] proposed low power, area and high throughput 4-stage pipelined implementation of the Blowfish cryptographic algorithm. It combines iterative method and operator-rescheduling methods to reduce the area occupied by the algorithm and partially pipelined method to increase the throughput. Joshi et al. [20] proposed implementation of Blowfish algorithm with modifying its function. They had shown that total time taken for encryption and decryption is reduced and the improvement will not violate the security of the algorithm. Chatterjee et al. [21] presented 3-stage pipeline implementation of Blowfish. Oukili and Bri [22] proposed high throughput efficient hardware architecture of Blowfish algorithm. Pipeline technique was adopted in order to increase the speed and the maximum operating frequency. In addition, the S-boxes tables of each round of the algorithm have been implemented in block RAMs to allow parallel data encryption.

4. PROPOSED BLOWFISH DESIGN

The proposed Blowfish design aimed to increase the throughput and use hardware resource as less as possible. Thus, pipeline technique is adopted. The pipeline strategy modifies the critical path by increasing the possible frequency of clock cycle. It consists in parallelizing the data inputs and outputs with the processing. Consequently, the design is divided into stages. By incrementing the number of these stages, the critical path can be decreased and as a result the speed is increased. The optimum number of pipeline stages and the best placement strategy for pipelining registers are two main factors to achieve an area-throughput design. As mentioned before, Blowfish is a 16-round Feistel network cipher. Therefore, we have inserted 18 64-bit registers before and after each round, forming a one-pass datapath for plaintexts. This is shown in figure 3. Since the 64-bit of plaintexts are processed in parallel, the throughput of the design is directly related to the clock frequency of the pipelined structure.

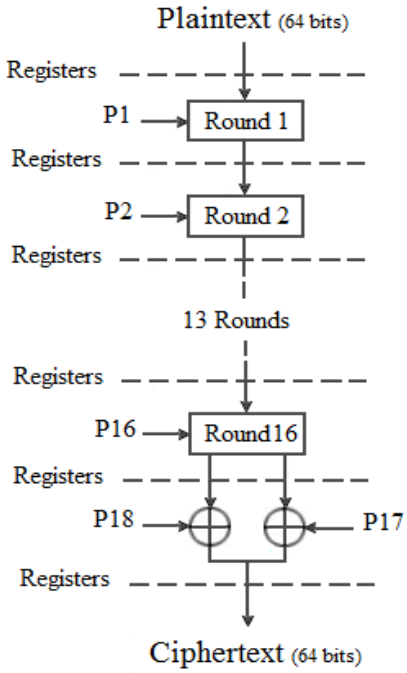

Figure 3. Proposed pipelined Blowfish

Clock frequency of a digital circuit is determined by the delay of the longest path, which is related to the combinational logic depth and routing delay. By analyzing the logic paths, we had noticed that the addition operations in the F function of each round are the most expensive in terms of execution time. To reduce the path delay, we had divided the addition modulo 232 into two additions modulo 216. Each one is executed separately and in a parallel manner by inserting registers in appropriate places. This will further increase the throughput of the algorithm. Note, that the increase of throughput requires an increase in area, as registers are required to store intermediate results. Figure 4 shows the proposed 5-stage pipelined round i. The elements of the P-array (18x32 bits) and the four S-boxes (256x32 bits) subkeys are stored in BRAMs, where the performance can be improved by decreasing the delay into the clock- to-out value of the flip- flop (FF). BRAM is used for storage of larger amount of data. To make a parallel encryption, these S-boxes are duplicated in all 16 rounds. This will tremendously increase the performance of the architecture. The ciphertext takes 82 clock cycles latency, first time only. Then we recover it at each clock cycle. In our proposed architecture, we used pipelining and parallelism in order to break the critical path delay and to obtain high encryption throughput at the expense of area as compared with the based non-pipelined Blowfish algorithm.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

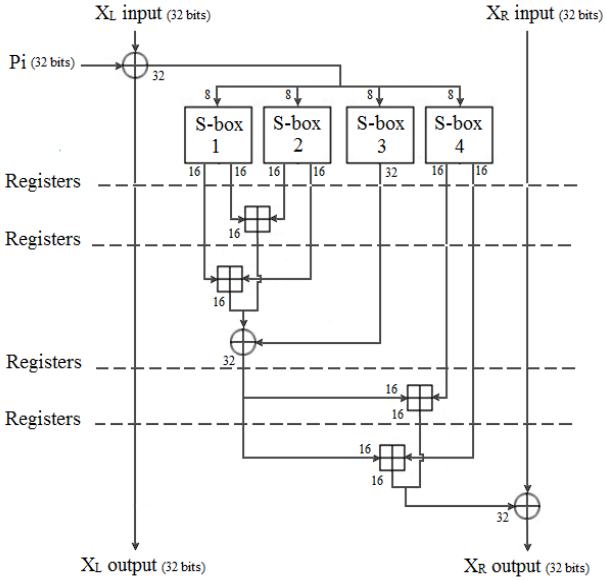

Figure 4. Proposed pipelined round i

Table 1. Devices utilization summaries

Resources	Utilization	
	Virtex-5	Virtex-6
Number of slices	2434/ 34560 7%	1881/ 56880 3%
Number of slice LUTs	4839/ 138240 3%	4770/ 227520 2%
Number of slice registers	6080/ 138240 4%	5942/ 455040 1%
Number of bonded IOBs	579/ 680 85%	579/ 720 80%
Number of block RAMs	74/ 212 34%	73/ 416 17%
Total equivalent cells	14138	-
Minimum period	3.38 ns	2.61 ns
Maximum frequency	312.921 Mhz	383.098 Mhz

5. IMPLEMENTATION SUMMARY AND COMPARISON

FPGA implementation of our proposed Blowfish architecture was established on Virtex-5 (xc5vlx220t) and Virtex-6 (xc6vlx365t) devices using Xilinx ISE Design Suite 14.7 as synthesis and Modelsim 6.1f as simulation tools. The design was described using VHDL language. The design achieves a maximum clock frequency of 312.921 MHz (3.38 ns), a throughput of 20.026 Gbps and an efficiency of 8.23 Mbps/slice on Virtex-5 FPGA and a maximum clock frequency of 383.098 MHz (2.61 ns), a throughput of 24.518 Gbps and an efficiency of 13.03 Mbps/slice on Virtex-6. We employ well-known Eq. (2) and Eq. (3) to calculate the throughput and the efficiency, respectively. The devices utilization summaries are given in table 1.

$$\text{Throughput} = \frac{\text{Number of outputted bits}}{\text{Delay of the critical path}} \quad (2)$$

$$\text{Efficiency} = \frac{\text{Throughput}}{\text{Used Slices}} \quad (3)$$

There are several hardware implementations for the Blowfish algorithm that aimed to achieve the most efficient architectures, by improving high throughput and area-efficient. Table 2 shows the performance figures for some reported architectures up to our best knowledge. It provides values of hardware utilization, maximum frequency, throughput and the increase in throughput of the proposed architecture compared to the reported ones, by a factor of.

As can be observed from table 2, the highest throughput reported to our knowledge is 12 Gbps with an efficiency of 9.38 Mbps/slice [22]. By comparing these results with our proposed implementation on a same FPGA board (Virtex-5), we see that ours gives 1.66 times more throughput with an increase of 66.77%. Furthermore, it increases used slices by 90% and reduces the efficiency by 12.79%. The implementation on Virtex-6 FPGA increases the throughput by 104%, the used slices by 46.95% and the efficiency by 38%.

As said before, our proposed Blowfish design aimed to increase the throughput and use hardware resource as less as possible. The reported results show that our proposed architecture provides better performance in terms of throughput than the previous implementations at the cost of increasing the area. Note that the increase of throughput requires an increase in area, as registers are required to store intermediate results.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 2. Hardware utilization, maximum frequency and throughput results

Architectures	Devices	Slices / Standard Cells	Maximum frequency (Mhz)	Throughput (Mbps)	Increase in Throughput*
[12]	SOC (system on a chip)	- / 4996	167	590	41.55
[13]	Virtex4 xc4vlx25	678 / -	-	673	36.43
[14]	-	- / -	13.09	840	29.18
[15]	Virtex6 xc6vlx240T	2348 / -	174	928	26.42
[16]	Altera EPM7128ELC84	- / -	27.02	1032	23.75
[17]	Virtex2 2v1500fg456	77 / -	146.515	1545	15.86
[18]	Zynq-7000	635 / -	324	2183	11.23
[19]	Virtex2 xcv50-bg256	1608 / 5986	167	2670	9.18
[20]	-	- / 4608	-	3680	6.66
[21]	Spartan3E xc3s500e	- / -	295.63	6300	3.89
[22]	Virtex5 xc5vlx220t	1280 / 9525	187.63	12008	2.04
Proposed design	**Virtex5 xc5vlx220t**	**2434 / 14138**	**312.921**	**20026**	**1.22**
	Virtex6 xc6vlx365t	**1881 / -**	**383.098**	**24518**	**-**

6. CONCLUSION

In this paper, we present high throughput Blowfish architecture. Pipelining technique is performed to obtain high throughput than basic structure by inserting registers in optimum placements. Also, we have proposed 5-stage pipelining Blowfish round to break the critical path delay and reach any further speed. Moreover, S-boxes are stored in block RAMs and introduced at each round of Blowfish algorithm to perform a parallel encryption. The implementations were done by FPGA devices. The input can be loaded every clock cycle and after an initial delay of 82 clock cycles, the encrypted data will appear consecutively. The reported results showed that our Blowfish implementations in terms of throughput provide better performance compared to the reported ones.

ACKNOWLEDGMENT

This work is supported by the presidency of Moulay Ismail University, Meknes-Morocco

REFRENCES:

[1] W. Stalling, "Cryptography and Network Security Principles and Practices", Prentice Hall, 4th ed., 2005.

[2] A. Kahate, "Cryptography and Network Security", Tata McGraw Hill, 2nd ed., 2007.

[3] National Institute of Standards and Technology, Federal Information Processing Standards Publication 46-3: Data Encryption Standard, 1999.

[4] National Institute of Standards and Technology, Federal Information Processing Standards Publication 197: Advanced Encryption Standard, 2001.

[5] D. Joan and R. Vincent, "AES Proposal: Rijndael", National Institute of Standards and Technology, 1999. http://csrc.nist.gov/archive/aes/rijndael/Rijndael-ammended.pdf

[6] B. Schneier, "Description of a New Variable-Length Key, 64-bit Block Cipher (Blowfish)", *Cambridge Security Workshop Proceedings*, Cambridge, U. K., December 9-11, 1993, pp. 191-204.

[7] B. Schneier, "Applied Cryptography: Protocols, Algorithms, and Source Code in C, Applied Cryptography", John Wiley and Sons, New York, 1996.

* Increase in throughput of our implementation on Virtex-6 compared to existing techniques (by a factor of)

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[8] B. Schneier, "The Blowfish Encryption Algorithm-One Year Later", *Dr. Dobb's Journal*, 1995. https://www.schneier.com/cryptography/archives/1995/09/the_blowfish_encrypt.html

[9] M.C.J. Lin and Y.L. Lin, "A VLSI Implementation of the Blowfish Encryption/Decryption Algorithm", *Asia and South Pacific Design Automation Conference*, Yokohama, Japan, January 25-28, 2000, pp. 1-2.

[10] S.M. Yoo, D. Kotturi, D.W. Pan and J. Blizzard, "An AES crypto chip using a high-speed parallel pipelined architecture", *Microprocessor and Microsystem*, Vol. 29, No. 7, 2005, pp. 317-326**.**

[11] V. Rijmen, "Cryptanalysis and design of iterated block ciphers", Doctoral dissertation, Katholieke Universiteit Leuven, 1997.

[12] B. Cody, J. Madigan, S. MacDonald and K.W. Hsu, "High Speed SOC Design for Blowfish Cryptographic Algorithm", *IFIP International Conference on Very Large Scale Integration*, Atlanta, USA, October 15-17, 2007, pp. 284-287.

[13] N.P. Kurniawan, P. Yudha and D. Denny, "An implementation of data encryption for Internet of Things using Blowfish algorithm on FPGA", *2nd International Conference on Information and Communication Technology,* Bandung, Indonesia, May 28-30, 2014, pp. 75-79.

[14] S.V. Kumara and P. Benakop, "High throughput and high speed Blowfish algorithm for secure Integrated Circuits", *Anale. Seria Informatic*, Vol. 12, No. 1, 2014, pp. 24-29.

[15] R. Ahmad and W. Ismail, "Performance Comparison of the Improved Power-Throughput AES and Blow fish Algorithms on FPGA", *Lecture Notes in Electrical Engineering*, Vol. 398, 2016, pp. 19-25.

[16] F. Guerrero and J.M. Noras, "Implementing block ciphering algorithms in hardware", *International Journal of Electronics*, Vol. 83, No.5, 1997, pp. 581-598.

[17] T.S.B. Sudarshan, R.A. Mir and S. Vijayalakshmi, "DRIL A Flexible Architecture for Blowfish Encryption Using Dynamic Reconfiguration, Replication, Inner-Loop Pipelining, Loop Folding Techniques", *Advances in Computer Systems Architecture*, Vol. **3740,** 2005, pp. 625-639.

[18] A. Rafidah, A.M. Asrulnizam and I. Widad, "Development of an Improved Power-Throughput Blowfish Algorithm on FPGA", *IEEE 12th International Colloquium on Signal Processing & its Applications,* Melaka, Malaysia, March 4-6, 2016, pp. 237-241.

[19] P.K. Kumar and K. Baskaran, "An ASIC implementation of low power and high throughput blowfish crypto algorithm", *Microelectronics Journal*, Vol. 41, No. 6, 2010, pp. 347-355.

[20] T. Joshi, R. Yadav and U. Malviya, "Design of enhanced speed Blowfish Algorithm for cryptography with merged encryption & decryption in VHDL", *International Journal of Engineering Research and Applications*, Special issue: ICIAC**,** 2014, pp. 68-71.

[21] S.R. Chatterjee, S. Majumder, B. Pramanik and M. Chakraborty, "FPGA Implementation of Pipelined Blowfish Algorithm", *Fifth International Symposium on Electronic System Design*, Mangalore, India, December 15-17**,** 2014, pp. 208-209.

[22] S. Oukili and S. Bri, "High Throughput Parallel Implementation of Blowfish Algorithm", *Applied Mathematics & Information Sciences,* Vol. 10, No. 6, 2016, pp. 2087-2092.